Binary Emission Discovery Calculus Theory Manuscript

Finite Kernel Theory, Concrete Instances, Proof Obligations, and Formalization Contract

Multi-file LaTeX Theory Manuscript

This manuscript presents the BEDC finite-kernel theory, its concrete instances, its proof obligations, and its formalization-facing contract in one maintained source tree. Core theory and implementation-facing material remain separated by parts and by source files.

Project Preface

This manuscript presents the Binary Emission Discovery Calculus as one maintained LaTeX source tree with a single compiled document.

The organizing principle is:

$$\text{finite-kernel theory} + \text{proof obligations} + \text{separate implementation contracts.}$$

Layer discipline. The theoretical core uses BEDC terminology and derived-interface notation. Implementation-facing text is placed in a separate part. This prevents the host language of a later proof assistant from contaminating the object theory.

Closure and verification. This manuscript separates theoretical closure from formal verification. A BEDC object may be theoretically closed at a stated scope when the paper supplies its source, classifier, exactness, stability, and ledger witnesses. A formalization marker records the independent verification status of those witnesses in a proof assistant. Machine checking strengthens reliability and auditability; it is not an object-language formation rule for closure. The discipline is recorded in chapter 9203.

Proof boundary. The proof boundary is the finite-kernel proof path:

$$\mathsf{msame} \to \mathsf{hsame} \to \mathsf{Ext/Cont} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{PackagePolicy/GapPolicy} \to \mathsf{CompGap} \to \mathsf{NameCert}.$$

Large derived interfaces such as $\mathsf{Real}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, and $\mathsf{Fold}^{\uparrow}$ remain thin interfaces whose full reconstruction is licensed only by separate naming certificates beyond the finite kernel.

External model posture. External semantic models remain explanatory artifacts. They do not alter the BEDC object theory and they do not replace the internal proof obligations recorded in the active chapters.

Reader's Roadmap

This manuscript presents the BEDC theory, its proof targets, and its implementation-facing material in one document while keeping their roles distinct.

How to read this manuscript

Reader goal	Recommended route
Understand the theory	Read the Preface, the finite-kernel core, and the hardening and proof-obligation parts. The core gives the object theory; the later parts record the active refinements and proof obligations.
Follow the theorem chain	Read the proof-sprint part, then chapter 19. The chain chapters state the finite-kernel theorems; the dependency chapter records the order in which they compose.
Understand the governance and proof posture	Read the Project Governance part and Chapter 9466. These chapters record the active contract and the amendment policy.
Prepare implementation work	Read the implementation-facing contract and Appendix A. The contract is external guidance for formalization work, not part of the core object theory.
Extend the book	Add new material to the relevant thematic chapter and update the registry, correspondence table, and amendment policy when the active reading changes.

Layer warning

This manuscript intentionally separates four layers:

1. **Core theory**: clean-room BEDC object-theory specification.
2. **Theorem chain**: theorem statements and proofs for the finite kernel.
3. **Implementation-facing contract**: external guidance for the proof-assistant encoding.
4. **Project governance**: editing, registry, and maintenance rules.

Terms such as "model", "theorem", "table", "implementation", and "module" belong to the explanatory metalanguage unless explicitly introduced as BEDC-derived interfaces.

Proof boundary

The proof boundary is not the reconstruction of all traditional mathematics. It is the finite-kernel proof path:

$$\mathsf{msame} \to \mathsf{hsame} \to \mathsf{Ext/Cont} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{PackagePolicy/GapPolicy} \to \mathsf{CompGap} \to \mathsf{NameCert}.$$

Large derived interfaces such as $\mathsf{Real}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, and $\mathsf{Fold}^{\uparrow}$ remain thin interfaces whose full reconstruction is licensed only by future naming certificates beyond the finite kernel.

Contents

Project Preface **i**

Reader's Roadmap **ii**

I Finite Kernel Theory **1**

1 Orientation and Layer Discipline **2**
1.1 Purpose of This Core 2
1.2 Boundary of This Core 2

2 Raw Marks and Mark-Sameness **3**

3 Emission Histories and History-Sameness **5**

4 Relational Extension and Continuation **7**
4.1 Relational extension 7
4.2 Relational continuation 9

5 Asking Events and Policies **21**
5.1 Asking events 21
5.2 Asking policies 21

6 Generated Probe Bundles and Signature Events **24**
6.1 Bundle grammar 24
6.2 Signature generation 34
6.3 Signature sameness 44

7 Packages and Package Policies **46**

8 Gap Policies, Coverage, Separation, and Composition **47**
8.1 Gap policy 47
8.2 Composite gaps 48

9 Typed Naming Certificates **52**

10 First Derived Interfaces **54**
10.1 Unary repetition 54
10.2 Stable transformations 54

11 Thin Seal Interface **56**

12 External Binary Model **57**
12.1 Purpose of the model . . . 57

13 Finite Theorem Manifest **60**

14 Boundary of the Finite Kernel **61**

Summary **62**

15 The Signature Package Instance **63**
15.1 Signature tokens . . . 63

16 The Concrete Signature Gap Policy **65**

17 The First Concrete Globalize Instance **67**

II Proof Sprint **69**

18 Finite Kernel Proof Target **70**
18.1 Purpose and scope . . . 70
18.2 Marks and mark-sameness . . . 70
18.3 Histories and history-sameness . . . 71
18.4 Relational extension . . . 72
18.5 Relational continuation . . . 72
18.6 Asking events and bundles . . . 74
18.7 Signatures and signature-sameness . . . 75
18.8 Packaging and gap policies . . . 76
18.9 Composite gap ledgers . . . 77
18.10 Name certificates . . . 77
18.11 First derived interface seeds . . . 77
18.12 Thin completion interface . . . 78
18.13 External semantic model . . . 78
18.14 Finite theorem target . . . 79
18.15 Conclusion . . . 79

19 Finite-Kernel Theorem Dependency DAG **80**
19.1 Milestone DAG . . . 80
19.2 Milestone A: marks . . . 80
19.3 Milestone B: histories . . . 81
19.4 Milestone C: relational extension and continuation . . . 81
19.5 Milestone D: asking and signatures . . . 81
19.6 Milestone E: signature sameness . . . 81
19.7 Milestones F–G: packages and gaps . . . 82
19.8 Milestone H: composite gap . . . 82
19.9 Milestone I: naming certificates . . . 82
19.10 Execution rule . . . 82

III Proof Standing **83**

20 Closed Generation and Internal Sameness **84**

21 Extension and Continuation Proof Spine **87**
21.1 Relational extension . . . 87
21.2 Relational continuation . . . 88

22 Signature Proof Spine **96**

23 Package and Gap Proof Spine **98**

24 Naming Certificate Proof Instances **100**
24.1 The unary repetition certificate . . . 100
24.2 The continuation certificate for addition-like behavior . . . 100

IV Hardening **102**

25 Package and Gap Policy Expansion **103**
25.1 Package policy as a public interface . . . 103
25.2 Gap policy as provenance interface . . . 104
25.3 Composite gaps . . . 104

26 Typed Naming Certificate Instances **106**
26.1 Certificate fields . . . 106
26.2 Unary repetition as $\mathsf{Nat}^{\uparrow}$ seed . . . 108
26.3 Continuation as $\mathsf{Add}^{\uparrow}$ seed . . . 109
26.4 Descent certificate for stable transformations . . . 109

27 Machine-Checkable Milestones **114**
27.1 Milestone A: history kernel . . . 114
27.2 Milestone B: signature kernel . . . 114
27.3 Milestone C: package and gap kernel . . . 115
27.4 Milestone D: first derived interface seeds . . . 115
27.5 Settled-kernel criterion . . . 115

V Concrete Instances **116**

28 A Concrete Naming Certificate for $\mathsf{Nat}^{\uparrow}$ **117**
28.1 Unary histories . . . 117
28.2 The certificate . . . 122

29 A Concrete Naming Certificate for $\mathsf{Add}^{\uparrow}$ **127**
29.1 Unary continuation . . . 127
29.2 The certificate . . . 128

30 A Concrete Naming Certificate for $\mathsf{Mul}^{\uparrow}$ **137**
30.1 Unary product rows . . . 137

31 A Concrete Naming Certificate for $\mathsf{Int}^{\uparrow}$ 141
31.1 Carrier definition . . . 141
31.2 Difference-pair context stability . . . 149
31.3 The certificate . . . 153
31.4 Rat Denominator Sibling Bridge . . . 166
31.5 Rat Denominator Sibling Link . . . 166

32 A Concrete Naming Certificate for $\mathsf{Bool}^{\uparrow}$ 175
32.1 Roadmap template status . . . 175
32.2 Carrier definition . . . 176
32.3 The certificate . . . 176

33 A Concrete Naming Certificate for $\mathsf{Option}^{\uparrow}$ 214
33.1 Roadmap template status . . . 214
33.2 Carrier definition . . . 215
33.3 The certificate . . . 215
33.4 Payload descent image classifier transported visible readback . . . 277
33.5 Payload descent image classifier reflective visible pair coherence . . . 278
33.6 Payload descent image classifier reflective transitive normalized public endpoint readback . . . 285
33.7 Composite image classifier visible payload classification . . . 291
33.8 Exclusive classifier branch readback . . . 296
33.9 Payload descent image classifier pair readback . . . 300
33.10 Payload descent image classifier visible payload exactness . . . 307
33.11 Payload descent image classifier visible pair coherence . . . 308
33.12 Payload descent image classifier public constructor normal form . . . 310
33.13 Composite image classifier public readback single-valuedness . . . 314
33.14 Composite image classifier normalized public branch certificate . . . 323
33.15 Composite image classifier normalized constructor normal form . . . 325
33.16 Composite image classifier normalized visible target-payload classification . . . 332
33.17 Composite image classifier normalized public readback completion . . . 343
33.18 Composite image classifier normalized one-sided public completion . . . 346
33.19 Tagged option map relation constructor normal form . . . 349
33.20 Tagged option map relation shared-source target normal form . . . 350
33.21 Tagged option map relation shared-target source reflection normal form . . . 351
33.22 Tagged option public endpoint visibility . . . 356
33.23 Tagged option present-chain endpoint classification . . . 356
33.24 Composite image classifier normalized factored public completion . . . 357
33.25 Tagged option composite factorization intermediate classifier determinacy . . . 363
33.26 Option composite image classifier normalized present intermediate payload determinacy . . . 365
33.27 Option history classifier unary two-sided context . . . 365
33.28 Tagged option map relation R-corner classifier coherence . . . 367
33.29 Tagged option payload descent image target classifier closure . . . 369
33.30 Option composite image classifier normalized present intermediate classifier determinacy . . . 370
33.31 Option composite image classifier normalized intermediate factor coherence . . . 372
33.32 Option composite image classifier normalized selected intermediate factor uniqueness 373

34 A Concrete Naming Certificate for Prod$^{\uparrow}$ **377**
34.1 Roadmap template status . . . 377
34.2 Carrier definition . . . 378
34.3 The certificate . . . 378

35 A Concrete Naming Certificate for Sum$^{\uparrow}$ **431**
35.1 Roadmap template status . . . 431
35.2 Carrier definition . . . 432
35.3 The certificate . . . 433

36 A Concrete Naming Certificate for List$^{\uparrow}$ **477**
36.1 Roadmap template status . . . 477
36.2 Carrier definition . . . 478
36.3 The certificate . . . 478
36.4 List map shape . . . 490
36.5 Singleton source length shape exhaustion . . . 547
36.6 Public append readback . . . 552
36.7 Bordered singleton payload readback . . . 561
36.8 Public reverse readback . . . 562
36.9 Public reverse and append . . . 565

37 A Concrete Naming Certificate for Rat$^{\uparrow}$ **569**
37.1 Carrier definition . . . 569
37.2 The certificate . . . 575

38 Certificate Obligations for Real$^{\uparrow}$ **597**
38.1 Carrier definition . . . 597
38.2 The certificate-obligation schema . . . 598
38.3 Real algebra and order boundary . . . 609
38.4 Real L10 current phase exit audit pullback . . . 817
38.5 StreamName and RegSeqRat dependency route . . . 982

39 A Concrete Naming Certificate for Complex$^{\uparrow}$ **1078**
39.1 Carrier definition . . . 1078
39.2 The certificate . . . 1081

40 A Concrete Naming Certificate for Monoid$^{\uparrow}$ **1089**
40.1 Carrier definition . . . 1089
40.2 Concrete history instance . . . 1089
40.3 The certificate . . . 1091

41 A Concrete Naming Certificate for Group$^{\uparrow}$ **1101**
41.1 Carrier definition . . . 1101
41.2 Concrete singleton history group instance . . . 1101
41.3 Concrete singleton empty-history group instance . . . 1108
41.4 Group concrete history instance surface . . . 1109
41.5 Singleton inverse image exactness . . . 1113
41.6 Concrete singleton equation solver . . . 1116
41.7 Concrete singleton equation solver determinacy . . . 1117

41.8 The certificate . . . 1119
41.9 Singleton contextual continuation obstruction . . . 1132
41.10 Centralizer transport . . . 1132
41.11 Centralizer normalizer orbit relation . . . 1138
41.12 Normalizer orbit classifier stability . . . 1141
41.13 Normalizer action on the centralizer . . . 1142
41.14 Center orbit collapse . . . 1148
41.15 Singleton continuation endpoint equivalence . . . 1149
41.16 Singleton two-sided product-unit exactness . . . 1151
41.17 Concrete singleton continuation group instance . . . 1155
41.18 Singleton conjugation terminal collapse . . . 1157
41.19 Singleton conjugation collapse . . . 1158
41.20 Concrete singleton normalizer action certificate . . . 1158
41.21 Unary append inverse obstruction . . . 1167
41.22 Terminal continuation boundary . . . 1172
41.23 Finite-history group-law witness . . . 1174
41.24 Singleton contextual exact visibility . . . 1175
41.25 Singleton two-conjugator orbit coverage . . . 1175
41.26 Center abelian normal package . . . 1181

42 A Concrete Naming Certificate for $\mathsf{AbGroup}^{\uparrow}$ 1185
42.1 Carrier definition . . . 1185
42.2 Concrete singleton history abelian group instance . . . 1185
42.3 The certificate . . . 1186
42.4 Concrete singleton empty-history abelian group instance . . . 1186
42.5 Public closure record . . . 1196

43 A Concrete Naming Certificate for $\mathsf{Ring}^{\uparrow}$ 1202
43.1 Carrier definition . . . 1202
43.2 Concrete singleton history ring instance . . . 1202
43.3 Concrete singleton empty-history ring instance . . . 1203
43.4 Singleton Empty-History Ring Instance . . . 1206
43.5 Empty-History Ground Schema Unlock . . . 1207
43.6 The certificate . . . 1208

44 A Concrete Naming Certificate for $\mathsf{CommRing}^{\uparrow}$ 1226
44.1 Carrier definition . . . 1226
44.2 Concrete singleton history commutative ring instance . . . 1226
44.3 Concrete singleton empty-history commutative ring instance . . . 1228
44.4 The certificate . . . 1235
44.5 CommRing quadratic polarization package . . . 1245
44.6 Public closure record . . . 1258
44.7 Singleton Cont History Instance for $\mathsf{CommRing}^{\uparrow}$ . . . 1259

45 A Concrete Naming Certificate for $\mathsf{Field}^{\uparrow}$ 1269
45.1 Carrier definition . . . 1269
45.2 Concrete singleton empty-history field schema instance . . . 1269
45.3 RatUp boundary for the singleton field schema . . . 1295

45.4 RatUp continuation closure for the FieldUp boundary . . . 1300
45.5 The certificate . . . 1332
45.6 Field affine equation laws . . . 1353
45.7 Product factor equation exactness . . . 1357
45.8 Signed square-factor consequences . . . 1365
45.9 Zero-apartness transport through certified products . . . 1372
45.10 Field square apartness inverse descent . . . 1376
45.11 Concrete Rat denominator continuation instance . . . 1378
45.12 Rat denominator append-continuation semigroup ledger bridge . . . 1382
45.13 Field square apartness inverse descent . . . 1385
45.14 Rat denominator empty-unit continuation monoid instance . . . 1391
45.15 Rat denominator unit endpoint exactness . . . 1393
45.16 Concrete Rat denominator external-unit package . . . 1398
45.17 Rat denominator strict monoid boundary . . . 1410
45.18 Rat denominator empty-unit cancellation exactness . . . 1417
45.19 Rat denominator external-unit envelope cancellation exactness . . . 1425
45.20 Rat denominator context-pair action laws . . . 1455
45.21 RatUp FieldUp exit support rows . . . 1472
45.22 Field affine explicit inverse . . . 1488
45.23 Five-factor inverse reversal . . . 1513
45.24 Apart-zero continuation exactness . . . 1611
45.25 Terminal history field instance . . . 1652

46 A Concrete Naming Certificate for $\mathsf{Module}^{\uparrow}$ 1655
46.1 Carrier definition . . . 1655
46.2 Concrete singleton empty-history module instance . . . 1655
46.3 The certificate . . . 1660

47 A Concrete Naming Certificate for $\mathsf{VecSpace}^{\uparrow}$ 1679
47.1 Carrier definition . . . 1679
47.2 Concrete singleton history vector space instance . . . 1679
47.3 The certificate . . . 1684

48 A Concrete Naming Certificate for $\mathsf{LinMap}^{\uparrow}$ 1689
48.1 Carrier definition . . . 1689
48.2 Concrete singleton empty-history linear map instance . . . 1690
48.3 The certificate . . . 1704

49 A Concrete Naming Certificate for $\mathsf{Mat}^{\uparrow}$ 1734
49.1 Carrier definition . . . 1734
49.2 Concrete singleton empty-history matrix instance . . . 1734
49.3 The certificate . . . 1750

50 A Concrete Naming Certificate for $\mathsf{Poly}^{\uparrow}$ 1777
50.1 Concrete singleton zero-polynomial instance . . . 1777
50.2 Carrier definition . . . 1778
50.3 Concrete singleton history polynomial instance . . . 1778
50.4 The certificate . . . 1783

51 A Concrete Naming Certificate for FPS$^{\uparrow}$ 1871
51.1 Carrier definition . . . 1871
51.2 Concrete singleton zero formal power series instance . . . 1871
51.3 The certificate . . . 1874

52 A Concrete Naming Certificate for Preorder$^{\uparrow}$ 1909
52.1 Carrier definition . . . 1909
52.2 Concrete unary prefix preorder instance . . . 1909
52.3 Concrete singleton empty preorder instance . . . 1910
52.4 The certificate . . . 1913

53 A Concrete Naming Certificate for POSet$^{\uparrow}$ 1923
53.1 Carrier definition . . . 1923
53.2 Concrete unary prefix POSet instance . . . 1925
53.3 The certificate . . . 1932
53.4 Poset root obligation surface . . . 1941
53.5 Poset root comparison surface . . . 1943
53.6 Poset prefix schema target surface . . . 1944
53.7 Poset TotalOrder and Lattice handoff . . . 1945

54 A Concrete Naming Certificate for TotalOrder$^{\uparrow}$ 1948
54.1 Carrier definition . . . 1948
54.2 Concrete singleton history total order instance . . . 1948
54.3 Concrete unary prefix total order instance . . . 1950
54.4 Concrete Singleton History Instance . . . 1962
54.5 OrderUp Interval Collapse . . . 1965
54.6 Finite Extrema Interval Uniqueness . . . 1967
54.7 OrderUp Phase Exit . . . 1969
54.7.1 OrderUp obligation set . . . 1970
54.8 The certificate . . . 1977
54.9 Public audit boundary . . . 1980
54.10 Root obligation surface . . . 1981
54.11 Unary Prefix History Instance . . . 1983

55 A Concrete Naming Certificate for Lattice$^{\uparrow}$ 1986
55.1 Carrier definition . . . 1986
55.2 Concrete singleton prefix lattice instance . . . 1986
55.3 The certificate . . . 1993
55.4 Carrier definition . . . 2013
55.5 Concrete singleton prefix lattice instance . . . 2013
55.6 The certificate . . . 2029

56 A Concrete Naming Certificate for Interval$^{\uparrow}$ 2041
56.1 Carrier definition . . . 2041
56.2 The certificate . . . 2044

57 A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$ 2047
57.1 Carrier definition . . . 2047
57.2 The certificate . . . 2047

58 A Concrete Naming Certificate for $\mathsf{S}^{1\uparrow}$ 2173
58.1 Carrier obligation surface . . . 2173
58.2 The certificate . . . 2182
58.3 The certificate . . . 2195

59 A Concrete Naming Certificate for $\mathsf{ContinuousMap}^{\uparrow}$ 2220
59.1 Carrier definition . . . 2220
59.2 The certificate . . . 2265

60 A Concrete Naming Certificate for $\mathsf{Compact}^{\uparrow}$ 2304
60.1 Carrier definition . . . 2304
60.2 The certificate . . . 2322

61 A Concrete Naming Certificate for $\mathsf{Cat}^{\uparrow}$ 2336
61.1 Carrier definition . . . 2336
61.2 The certificate . . . 2391
61.3 The certificate . . . 2404

62 A Concrete Naming Certificate for $\mathsf{Functor}^{\uparrow}$ 2434
62.1 Carrier definition . . . 2434
62.2 The certificate . . . 2452

63 A Concrete Naming Certificate for $\mathsf{NatTrans}^{\uparrow}$ 2472
63.1 Carrier definition . . . 2472
63.2 The certificate . . . 2486

64 A Concrete Naming Certificate for $\mathsf{Prime}^{\uparrow}$ 2510
64.1 Multiplicative structure on unary histories . . . 2510
64.2 The prime predicate . . . 2536
64.3 Effective primality testing . . . 2539
64.4 Euclid's theorem on the infinitude of primes . . . 2545
64.5 The fundamental theorem of arithmetic . . . 2548
64.6 The certificate . . . 2551

65 A Concrete Naming Certificate for $\mathsf{CplxLim}^{\uparrow}$ 2556
65.1 Modulus and complex distance on Hist . . . 2556
65.2 Regular complex sequences . . . 2560
65.3 The complex-limit predicate . . . 2560
65.4 Closure laws on regular sequences . . . 2561
65.5 Componentwise limit assembly . . . 2565
65.6 The certificate . . . 2566

66 A Concrete Naming Certificate for ConvRad$^{\uparrow}$ 2573
66.1 Power series carrier and partial sums . . . 2573
66.2 Geometric bound predicate . . . 2574
66.3 Convergence radius predicate . . . 2577
66.4 Opening Boundary . . . 2579
66.5 Closure laws . . . 2579
66.6 The certificate . . . 2581
66.7 Cauchy–Hadamard Boundary . . . 2583

67 A Concrete Naming Certificate for CplxDiff$^{\uparrow}$ 2591
67.1 The complex difference quotient . . . 2591
67.2 Pointwise complex differentiability . . . 2595
67.3 Closure laws for complex differentiability . . . 2597
67.4 Real-component partial derivatives . . . 2598
67.5 Cauchy–Riemann equations, constructive form . . . 2599
67.6 The certificate . . . 2599

68 A Naming Seed for Holomorphic$^{\uparrow}$ 2604
68.1 Open-disk carrier . . . 2604
68.2 Holomorphic predicate . . . 2611
68.3 Core theorem obligations . . . 2611
68.4 The seed fields . . . 2615

69 A Concrete Naming Certificate for CplxSer$^{\uparrow}$ 2618
69.1 Term sequences and partial sums . . . 2618
69.2 Absolute convergence . . . 2621
69.3 Convergence and uniqueness . . . 2624
69.4 Comparison, ratio, and root tests . . . 2624
69.5 Cauchy product . . . 2625
69.6 Closure laws . . . 2625
69.7 The certificate . . . 2626

70 A Concrete Naming Certificate for Dirichlet$^{\uparrow}$ 2631
70.1 Dirichlet term and partial sum . . . 2631
70.2 Abscissa of absolute convergence . . . 2636
70.3 Convergence and holomorphicity in the half-plane . . . 2636
70.4 Euler product for multiplicative coefficients . . . 2637
70.5 The certificate . . . 2637

71 A Concrete Naming Certificate for ZetaBasic$^{\uparrow}$ on $\Re(s) > 1$ 2639
71.1 The constant-coefficient Dirichlet sum . . . 2639
71.2 Convergence and abscissa . . . 2642
71.3 The Euler product over primes . . . 2642
71.4 Non-vanishing on the half-plane . . . 2642
71.5 Holomorphicity on the half-plane . . . 2642
71.6 The certificate . . . 2642

72 A Concrete Naming Certificate for CritStrip$^{\uparrow}$ 2645
72.1 The strip and critical-line predicates . . . 2645
72.2 Compact sub-strips and boundary witnesses . . . 2646
72.3 Closure laws . . . 2647
72.4 The certificate . . . 2648

73 A Concrete Naming Certificate for ZetaZeros$^{\uparrow}$ 2650
73.1 ζ-evaluation and zero predicate . . . 2650
73.2 Trivial zeros . . . 2651
73.3 Non-trivial zeros . . . 2651
73.4 Closure laws . . . 2651
73.5 The certificate . . . 2652

74 A Conditional Certificate Boundary for RealAnalytic$^{\uparrow}$ 2656
74.1 Constructive π via Leibniz acceleration . . . 2656
74.2 Constructive exponential . . . 2658
74.3 Constructive logarithm on positives . . . 2661
74.4 Constructive sine and cosine . . . 2663
74.5 Certificate target . . . 2664

75 A Concrete Naming Certificate for Gamma$^{\uparrow}$ 2672
75.1 The pole locus and the Gamma domain . . . 2672
75.2 Gamma at unary positive integers via factorial . . . 2675
75.3 Gamma extension via Weierstrass product . . . 2675
75.4 Functional equation $\Gamma(s+1) = s\,\Gamma(s)$ . . . 2676
75.5 Reflection formula boundary certificate . . . 2676
75.6 Certificate target . . . 2677

76 A Concrete Naming Certificate for CplxAnalytic$^{\uparrow}$ 2681
76.1 Complex exponential via Euler decomposition . . . 2681
76.2 Complex logarithm with principal branch . . . 2686
76.3 Complex sine and cosine . . . 2687
76.4 Complex powers $z \mapsto a^z$ and n^{-s} . . . 2687
76.5 Certificate target . . . 2688

77 A Concrete Naming Certificate for CplxTopo$^{\uparrow}$ 2690
77.1 Open disks . . . 2690
77.2 Open sets and finite intersections . . . 2694
77.3 Compact subsets via dyadic ε-nets . . . 2695
77.4 Density witnesses and clustering points . . . 2695
77.5 Domain compatibility for analytic continuation . . . 2696
77.6 Certificate target . . . 2696

78 A Concrete Naming Certificate for Magma$^{\uparrow}$ 2703
78.1 Carrier definition . . . 2703
78.2 The certificate . . . 2709

79 A Concrete Naming Certificate for Semigroup$^{\uparrow}$ 2712
79.1 Carrier definition . . . 2712
79.2 The certificate . . . 2713

80 A Concrete Naming Certificate for Subgroup$^{\uparrow}$ 2717
80.1 Carrier definition . . . 2717
80.2 The certificate . . . 2717

81 A Concrete Naming Certificate for Ideal$^{\uparrow}$ 2747
81.1 Carrier definition . . . 2747
81.2 The certificate . . . 2747
81.3 Standard bridge . . . 2774

82 A Concrete Naming Certificate for QuotientGroup$^{\uparrow}$ 2777
82.1 Carrier definition . . . 2777
82.2 The certificate . . . 2780

83 A Concrete Naming Certificate for QuotientRing$^{\uparrow}$ 2792
83.1 Carrier definition . . . 2792
83.2 The certificate . . . 2792

84 A Concrete Naming Certificate for Determinant$^{\uparrow}$ 2799
84.1 Carrier definition . . . 2799
84.2 The certificate . . . 2801

85 A Concrete Naming Certificate for Eigen$^{\uparrow}$ 2804
85.1 Carrier definition . . . 2804
85.2 The certificate . . . 2807

86 A Concrete Naming Certificate for InnerProduct$^{\uparrow}$ 2808
86.1 Carrier definition . . . 2808
86.2 The certificate . . . 2809
86.3 Parallelogram norm seed . . . 2819
86.4 Consumer threshold surface . . . 2827
86.5 Two-vector Gram boundary . . . 2829
86.6 Root unblock obligation basis . . . 2831
86.7 InnerProduct root scalar consumer frontier . . . 2833
86.8 Public NameCert export . . . 2834

87 A Concrete Naming Certificate for TensorProduct$^{\uparrow}$ 2838
87.1 Carrier definition . . . 2838
87.2 The certificate . . . 2840

88 A Concrete Naming Certificate for Topology$^{\uparrow}$ 2844
88.1 Carrier definition . . . 2844
88.2 The certificate . . . 2844
88.3 Public Open Derivation Trees . . . 2867
88.4 Scoped obligation triple . . . 2872

89 A Concrete Naming Certificate for Norm$^{\uparrow}$ 2877
89.1 Carrier definition . . . 2877
89.2 The certificate . . . 2878

90 A Concrete Naming Certificate for Banach$^{\uparrow}$ 2882
90.1 Carrier definition . . . 2882
90.2 The certificate . . . 2887
90.3 Banach limit surface certificate . . . 2901

91 A Concrete Naming Certificate for Hilbert$^{\uparrow}$ 2932
91.1 Carrier definition . . . 2932
91.2 The certificate . . . 2933

92 A Concrete Naming Certificate for Measure$^{\uparrow}$ 2946
92.1 Carrier definition . . . 2946
92.2 The certificate . . . 2953
92.3 Continuity from below row . . . 2963
92.4 Countable-name root surface . . . 2982
92.5 Measure root countable obligation triple . . . 3009
92.6 Measure root real-stream regularity front . . . 3013
92.7 Measure root finite-support consumer front . . . 3021
92.8 Measure root event boundary package . . . 3023
92.9 Root premeasure event-value bridge . . . 3026
92.10 Measure root public face exactness . . . 3030
92.11 Measure root real-stream event obligations . . . 3034
92.12 Measure root countable relative-difference bridge . . . 3036
92.13 Measure root real-stream premeasure bridge . . . 3038
92.14 Measure BHist root obligation basis . . . 3040
92.15 Measure root finite disjoint event envelope . . . 3043
92.16 Measure root event-value closure . . . 3045

93 A Concrete Naming Certificate for Filter$^{\uparrow}$ 3056
93.1 Carrier definition . . . 3056
93.2 The certificate . . . 3060

94 A Concrete Naming Certificate for FunctionalAnalysis$^{\uparrow}$ 3064
94.1 Carrier definition . . . 3064
94.2 The certificate . . . 3066

95 A Concrete Naming Certificate for Completion$^{\uparrow}$ 3070
95.1 Carrier definition . . . 3070
95.2 The certificate . . . 3074

96 A Concrete Naming Certificate for Manifold$^{\uparrow}$ 3077
96.1 Carrier definition . . . 3077
96.2 Atlas Obligation Inventory . . . 3081
96.3 The certificate . . . 3083
96.4 Chart Coordinate Transport . . . 3088
96.5 Scoped Manifold Boundary . . . 3094

96.6 Terminal Standard Bridge 3096

97 A Concrete Naming Certificate for Bundle$^{\uparrow}$ 3098
97.1 Carrier definition 3098
97.2 The certificate 3099

98 A Concrete Naming Certificate for Homology$^{\uparrow}$ 3103
98.1 Carrier definition 3103
98.2 The certificate 3110

99 A Concrete Naming Certificate for Cohomology$^{\uparrow}$ 3113
99.1 Carrier definition 3113
99.2 The certificate 3119

100 A Concrete Naming Certificate for Padic$^{\uparrow}$ 3121
100.1 Carrier definition 3121
100.2 The certificate 3129

101 A Concrete Naming Certificate for Adele$^{\uparrow}$ 3132
101.1 Carrier definition 3132
101.2 The certificate 3139

102 A Concrete Naming Certificate for ModN$^{\uparrow}$ 3142
102.1 Carrier definition 3142
102.2 The certificate 3142

103 A Concrete Naming Certificate for LFunction$^{\uparrow}$ 3146
103.1 Carrier definition 3146
103.2 The certificate 3149

104 A Concrete Naming Certificate for Automorphic$^{\uparrow}$ 3152
104.1 Carrier definition 3152
104.2 The certificate 3156

105 A Concrete Naming Certificate for CatLimit$^{\uparrow}$ 3160
105.1 Carrier definition 3160
105.2 The certificate 3160

106 A Concrete Naming Certificate for CatColimit$^{\uparrow}$ 3171
106.1 Carrier definition 3171
106.2 The certificate 3171

107 A Concrete Naming Certificate for Adjunction$^{\uparrow}$ 3180
107.1 Carrier definition 3180
107.2 The certificate 3193

108 A Concrete Naming Certificate for Monad$^{\uparrow}$ 3200
108.1 Carrier definition 3200
108.2 The certificate 3203

109A Concrete Naming Certificate for Yoneda$^{\uparrow}$ 3205
109.1Carrier definition . . . 3205
109.2The certificate . . . 3207

110A Concrete Naming Certificate for EquivCat$^{\uparrow}$ 3211
110.1Carrier definition . . . 3211
110.2The certificate . . . 3214

111A Concrete Naming Certificate for Set$^{\uparrow}$ 3216
111.1Carrier definition . . . 3216
111.2The certificate . . . 3216

112A Concrete Naming Certificate for FinSet$^{\uparrow}$ 3220
112.1Carrier definition . . . 3220
112.2The certificate . . . 3220

113A Concrete Naming Certificate for Subtype$^{\uparrow}$ 3235
113.1Carrier definition . . . 3235
113.2The certificate . . . 3235

114A Concrete Naming Certificate for Empty$^{\uparrow}$ 3241
114.1Carrier definition . . . 3241
114.2The certificate . . . 3241

115A Concrete Naming Certificate for Unit$^{\uparrow}$ 3245
115.1Carrier definition . . . 3245
115.2The certificate . . . 3247

116A Concrete Naming Certificate for Permutation$^{\uparrow}$ 3249
116.1Carrier definition . . . 3249
116.2The certificate . . . 3250

117A Concrete Naming Certificate for SymGroup$^{\uparrow}$ 3255
117.1Carrier definition . . . 3255
117.2The certificate . . . 3255

118A Concrete Naming Certificate for Graph$^{\uparrow}$ 3258
118.1Carrier definition . . . 3258
118.2The certificate . . . 3259

119A Concrete Naming Certificate for Tree$^{\uparrow}$ 3265
119.1Carrier definition . . . 3265
119.2The certificate . . . 3268

120A Concrete Naming Certificate for Seq$^{\uparrow}$ 3277
120.1Carrier definition . . . 3277
120.2The certificate . . . 3278

121 A Concrete Naming Certificate for Series↑ 3280
121.1 Carrier definition . . . 3280
121.2 The certificate . . . 3281

122 A Concrete Naming Certificate for Derivative↑ 3284
122.1 Carrier definition . . . 3284
122.2 The certificate . . . 3289

123 A Concrete Naming Certificate for Integral↑ 3291
123.1 Carrier definition . . . 3291
123.2 The certificate . . . 3292

124 A Concrete Naming Certificate for Fourier↑ 3295
124.1 Carrier definition . . . 3295
124.2 The certificate . . . 3295

125 A Concrete Naming Certificate for Residue↑ 3299
125.1 Carrier definition . . . 3299
125.2 The certificate . . . 3301

126 A Concrete Naming Certificate for StreamName↑ 3303
126.1 Carrier definition . . . 3303
126.2 The certificate . . . 3308
126.3 StreamName constant finite-window boundary . . . 3338
126.4 StreamName constant-window transport totality . . . 3343
126.5 StreamName Real and RegSeqRat handoff . . . 3344
126.6 StreamName Real-RegSeq window source exhaustion . . . 3373
126.7 StreamName finite-window Real handoff coverage . . . 3374
126.8 StreamName Real seal factorization boundary . . . 3380
126.9 StreamName terminal source lock . . . 3382
126.10 StreamName RegSeqRat Pointwise Handoff . . . 3394
126.11 StreamName RegSeqRat RealUp Tail-Selector Transport . . . 3410
126.12 StreamName Tail-Window Coverage Packet . . . 3419
126.13 StreamName Common-Refinement Source Determinacy . . . 3428
126.14 StreamName Common-Refinement Public Exactness . . . 3432
126.15 StreamName Synchronized Window Union . . . 3433
126.16 StreamName finite-window lattice meet . . . 3439
126.17 StreamName lattice-meet terminal factorization . . . 3443
126.18 StreamName RegSeqRat common-window transport . . . 3446
126.19 Public obligation surface . . . 3472
126.20 StreamName Finite-Window Regular Seal Non-Escape . . . 3481
126.21 StreamName Lattice Dependency . . . 3486
126.22 StreamName RegularCauchyName Lattice Link . . . 3490
126.23 StreamName RegSeqRat Real Lattice Dependency . . . 3491
126.24 StreamName finite-window transport obligations . . . 3502
126.25 StreamName finite-window nonempty-index handoff . . . 3507
126.26 StreamName finite-window budget handoff . . . 3510
126.27 StreamName Real-RegSeq shared-tail budget route . . . 3515

126.28 StreamName Common-Refinement Terminal Real Seal . . . 3516
126.29 StreamName Open-Phase Terminal Route . . . 3523
126.30 StreamName Open-Phase Terminal Exactness . . . 3525
126.31 StreamName Open-Phase Window Budget Routes . . . 3530
126.32 StreamName Open-Phase Dependency Lattice . . . 3534
126.33 StreamName Open-Phase Exit Boundary . . . 3538
126.34 StreamName L10 Real-Completion Sibling Spine . . . 3558
126.35 StreamName Real-Completion Gateway Package . . . 3558
126.36 StreamName Real-completion four-object exit route . . . 3568
126.37 StreamName finite-observation section packet . . . 3585
126.38 StreamName RegSeqRat Real budget factorization . . . 3588
126.39 StreamName Tail-Compatible Finite-Window Packet . . . 3592
126.40 StreamName finite-request section packet . . . 3599
126.41 StreamName finite-request RealUp seal lock . . . 3607
126.42 StreamName RegSeqRat Real seal handoff . . . 3609
126.43 StreamName finite-request section pullback . . . 3611
126.44 StreamName finite-section pullback exactness . . . 3615
126.45 StreamName finite-section reindex gluing . . . 3625
126.46 StreamName RegSeqRat Real public seal boundary . . . 3626
126.47 StreamName RegSeqRat Dyadic Real obligation handoff . . . 3629
126.48 StreamName terminal window fusion . . . 3637
126.49 StreamName Finite-Cover Support Removal Global Obstruction . . . 3665
126.50 StreamName finite-window diagonal section . . . 3668
126.51 StreamName L10 gateway obligation surface . . . 3669
126.52 Pointwise Certified Source . . . 3672
126.53 Finite Observation Section Cofinality . . . 3674

127 A Concrete Naming Certificate for TotallyBounded$^{\uparrow}$ 3682
127.1 Carrier definition . . . 3682
127.2 The certificate . . . 3684

128 A Concrete Naming Certificate for CompleteMetric$^{\uparrow}$ 3690
128.1 Carrier definition . . . 3690
128.2 The certificate . . . 3691

129 A Concrete Naming Certificate for CompactMetric$^{\uparrow}$ 3697
129.1 Carrier definition . . . 3697
129.2 The certificate . . . 3697

130 A Metric-Carrier Interface for ContinuousMap$^{\uparrow}$ 3710
130.1 Carrier definition . . . 3710
130.2 The certificate . . . 3723

131 A Concrete Naming Certificate for Eq$^{\uparrow}$ 3741
131.1 Carrier definition . . . 3741
131.2 The certificate . . . 3744

132 A Concrete Naming Certificate for Func$^{\uparrow}$ 3745
132.1 Carrier definition . . . 3745
132.2 The certificate . . . 3747

133 A Concrete Naming Certificate for RiemannianMetric$^{\uparrow}$ 3749
133.1 Carrier definition . . . 3749
133.2 The certificate . . . 3751

134 A Concrete Naming Certificate for Connection$^{\uparrow}$ 3753
134.1 Carrier definition . . . 3753
134.2 The certificate . . . 3755

135 A Concrete Naming Certificate for Curvature$^{\uparrow}$ 3757
135.1 Carrier definition . . . 3757
135.2 The certificate . . . 3761

136 A Concrete Naming Certificate for DiffForm$^{\uparrow}$ 3777
136.1 Carrier definition . . . 3777
136.2 The certificate . . . 3778
136.3 DiffForm degree-probe support . . . 3786
136.4 DiffForm degree-probe alignment . . . 3788
136.5 DiffForm root-unblock obligations . . . 3790
136.6 DiffForm root export threshold . . . 3815
136.7 Boundary exhaustion for DiffFormUp . . . 3818
136.8 DiffForm root consumer threshold certificate . . . 3820
136.9 DiffForm root degree-classifier package . . . 3828
136.10 DiffForm root consumer-readable package . . . 3831
136.11 DiffForm root consumer entry . . . 3833
136.12 DiffForm root consumer factorization . . . 3836
136.13 DiffForm root consumer export packet . . . 3837
136.14 DiffForm consumer boundary obligations . . . 3839

137 A Concrete Naming Certificate for DeRham$^{\uparrow}$ 3842
137.1 Carrier definition . . . 3842
137.2 The certificate . . . 3842

138 A Concrete Naming Certificate for Symplectic$^{\uparrow}$ 3856
138.1 Carrier definition . . . 3856
138.2 The certificate . . . 3857

139 A Concrete Naming Certificate for Contact$^{\uparrow}$ 3862
139.1 Carrier definition . . . 3862
139.2 The certificate . . . 3863

140 A Concrete Naming Certificate for LieGroup$^{\uparrow}$ 3867
140.1 Carrier definition . . . 3867
140.2 The certificate . . . 3867
140.3 Carrier definition . . . 3879
140.4 The certificate . . . 3879

141A Concrete Naming Certificate for LieAlgebra$^{\uparrow}$ 3892
141.1Carrier definition . . . 3892
141.2The certificate . . . 3892

142A Concrete Naming Certificate for ExpMap$^{\uparrow}$ 3899
142.1Carrier definition . . . 3899
142.2The certificate . . . 3901

143A Concrete Naming Certificate for AdjointRep$^{\uparrow}$ 3904
143.1The certificate . . . 3904

144A Concrete Naming Certificate for RootSystem$^{\uparrow}$ 3908
144.1Carrier definition . . . 3908
144.2The certificate . . . 3909

145A Concrete Naming Certificate for WeylGroup$^{\uparrow}$ 3915
145.1Carrier definition . . . 3915
145.2The certificate . . . 3915

146A Concrete Naming Certificate for BilinForm$^{\uparrow}$ 3922
146.1Carrier definition . . . 3922
146.2The certificate . . . 3927

147A Concrete Naming Certificate for Clifford$^{\uparrow}$ 3931
147.1Carrier definition . . . 3931
147.2The certificate . . . 3933

148A Concrete Naming Certificate for SpinGroup$^{\uparrow}$ 3936
148.1Carrier definition . . . 3936
148.2The certificate . . . 3943

149A Concrete Naming Certificate for PinGroup$^{\uparrow}$ 3959
149.1Carrier definition . . . 3959
149.2The certificate . . . 3961

150A Concrete Naming Certificate for PreSheaf$^{\uparrow}$ 3988
150.1Carrier definition . . . 3988
150.2The certificate . . . 3991

151A Concrete Naming Certificate for Sheaf$^{\uparrow}$ 3993
151.1Carrier definition . . . 3993
151.2The certificate . . . 3993
151.3Displayed base change surface . . . 4016
151.4Sheaf base-change functoriality . . . 4018
151.5Root obligations for downstream certificates . . . 4035
151.6Identity-cover root package . . . 4039
151.7Downstream consumption boundary . . . 4065
151.8Carrier definition . . . 4067
151.9The certificate . . . 4067

151.10 Downstream consumption boundary . . . 4090
151.11 Displayed base change surface . . . 4094
151.12 Sheaf base-change functoriality . . . 4097
151.13 Identity-cover root package . . . 4105
151.14 Root obligations for downstream certificates . . . 4138

152 A Concrete Naming Certificate for RingedSpace$^{\uparrow}$ 4143
152.1 Carrier definition . . . 4143
152.2 The certificate . . . 4146

153 A Concrete Naming Certificate for Scheme$^{\uparrow}$ 4147
153.1 Carrier definition . . . 4147
153.2 The certificate . . . 4148

154 A Concrete Naming Certificate for AffineVar$^{\uparrow}$ 4150
154.1 Carrier definition . . . 4150
154.2 The certificate . . . 4150

155 A Concrete Naming Certificate for ProjectiveVar$^{\uparrow}$ 4159
155.1 Carrier definition . . . 4159
155.2 The certificate . . . 4159

156 A Concrete Naming Certificate for Homotopy$^{\uparrow}$ 4165
156.1 Carrier definition . . . 4165
156.2 The certificate . . . 4167

157 A Concrete Naming Certificate for SimplicialSet$^{\uparrow}$ 4173
157.1 Carrier definition . . . 4173
157.2 The certificate . . . 4174

158 A Concrete Naming Certificate for ModelCat$^{\uparrow}$ 4179
158.1 Carrier definition . . . 4179
158.2 The certificate . . . 4182

159 A Concrete Naming Certificate for InfCat$^{\uparrow}$ 4185
159.1 Carrier definition . . . 4185
159.2 The certificate . . . 4185

160 A Concrete Naming Certificate for DerivedCat$^{\uparrow}$ 4190
160.1 Carrier definition . . . 4190
160.2 The certificate . . . 4190

161 A Concrete Naming Certificate for FieldExt$^{\uparrow}$ 4194
161.1 Carrier definition . . . 4194
161.2 The certificate . . . 4196
161.3 Sibling dependency lattice . . . 4210

162A Concrete Naming Certificate for AlgClosure↑ **4213**
162.1 Carrier definition 4213
162.2 The certificate 4215

163A Concrete Naming Certificate for GaloisExt↑ **4219**
163.1 Carrier definition 4219
163.2 The certificate 4221

164A Concrete Naming Certificate for GaloisGroup↑ **4230**
164.1 Carrier definition 4230
164.2 The certificate 4231

165A Concrete Naming Certificate for SplittingField↑ **4242**
165.1 Carrier definition 4242
165.2 The certificate 4242

166A Concrete Naming Certificate for SeparableExt↑ **4247**
166.1 Carrier definition 4247
166.2 The certificate 4247

167A Concrete Naming Certificate for Transcendence↑ **4251**
167.1 Carrier definition 4251
167.2 The certificate 4251

168A Concrete Naming Certificate for NumField↑ **4254**
168.1 Carrier definition 4254
168.2 The certificate 4255

169A Concrete Naming Certificate for RingOfIntegers↑ **4267**
169.1 Carrier definition 4267
169.2 The certificate 4267

170A Concrete Naming Certificate for IdealClass↑ **4273**
170.1 Carrier definition 4273
170.2 The certificate 4274

171A Concrete Naming Certificate for DirichletUnit↑ **4276**
171.1 Carrier definition 4276
171.2 The certificate 4280

172A Concrete Naming Certificate for Dedekind↑ **4284**
172.1 Carrier definition 4284
172.2 The certificate 4285

173A Concrete Naming Certificate for LocalField↑ **4288**
173.1 Carrier definition 4288
173.2 The certificate 4290

174A Concrete Naming Certificate for Cyclotomic$^{\uparrow}$ **4291**
174.1Carrier definition 4291
174.2The certificate 4294

175A Concrete Naming Certificate for EllipticCurve$^{\uparrow}$ **4319**
175.1Carrier definition 4319
175.2The certificate 4319

176A Concrete Naming Certificate for AbelianCat$^{\uparrow}$ **4326**
176.1Carrier definition 4326
176.2The certificate 4328

177A Concrete Naming Certificate for DerivedFunctor$^{\uparrow}$ **4334**
177.1Carrier definition 4334
177.2The certificate 4340

178A Concrete Naming Certificate for SpectralSeq$^{\uparrow}$ **4345**
178.1Carrier definition 4345
178.2The certificate 4347

179A Concrete Naming Certificate for TriangulatedCat$^{\uparrow}$ **4356**
179.1Carrier definition 4356
179.2The certificate 4357

180A Concrete Naming Certificate for HopfAlg$^{\uparrow}$ **4361**
180.1Carrier definition 4361
180.2The certificate 4361

181A Concrete Naming Certificate for MonoidalCat$^{\uparrow}$ **4368**
181.1Carrier definition 4368
181.2The certificate 4368

182A Concrete Naming Certificate for EnrichedCat$^{\uparrow}$ **4373**
182.1Carrier definition 4373
182.2The certificate 4374

183A Concrete Naming Certificate for Topos$^{\uparrow}$ **4378**
183.1Carrier definition 4378
183.2The certificate 4379

184A Concrete Naming Certificate for ProbSpace$^{\uparrow}$ **4382**
184.1Carrier definition 4382
184.2The certificate 4382

185A Concrete Naming Certificate for RandomVar$^{\uparrow}$ **4392**
185.1Carrier definition 4392
185.2The certificate 4392

186A Concrete Naming Certificate for Distribution$^{\uparrow}$ **4415**
186.1Carrier definition 4415
186.2The certificate 4416

187A Concrete Naming Certificate for Independence$^{\uparrow}$ **4433**
187.1Carrier definition 4433
187.2The certificate 4434

188A Concrete Naming Certificate for CondExp$^{\uparrow}$ **4445**
188.1Carrier definition 4445
188.2The certificate 4445

189A Concrete Naming Certificate for MarkovChain$^{\uparrow}$ **4449**
189.1Carrier definition 4449
189.2The certificate 4453

190A Concrete Naming Certificate for Martingale$^{\uparrow}$ **4458**
190.1Carrier definition 4458
190.2The certificate 4460

191A Concrete Naming Certificate for Brownian$^{\uparrow}$ **4468**
191.1Carrier definition 4468
191.2The certificate 4469

192A Concrete Naming Certificate for Entropy$^{\uparrow}$ **4483**
192.1Carrier definition 4483
192.2The certificate 4484

193A Concrete Naming Certificate for Ode$^{\uparrow}$ **4488**
193.1Carrier definition 4488
193.2The certificate 4488

194A Concrete Naming Certificate for Pde$^{\uparrow}$ **4493**
194.1Carrier definition 4493
194.2The certificate 4495

195A Concrete Naming Certificate for DynSystem$^{\uparrow}$ **4499**
195.1Carrier definition 4499
195.2The certificate 4502

196A Concrete Naming Certificate for Ergodic$^{\uparrow}$ **4510**
196.1Carrier definition 4510
196.2The certificate 4510

197A Concrete Naming Certificate for FirstOrder$^{\uparrow}$ **4513**
197.1Carrier definition 4513
197.2The certificate 4513

198A Concrete Naming Certificate for ModelTheory$^{\uparrow}$ 4522
198.1Carrier definition 4522
198.2The certificate 4524

199A Concrete Naming Certificate for Computable$^{\uparrow}$ 4529
199.1Carrier definition 4529
199.2The certificate 4529

200A Concrete Naming Certificate for LambdaCalc$^{\uparrow}$ 4539
200.1Carrier definition 4539
200.2The certificate 4541

201A Concrete Naming Certificate for Partition$^{\uparrow}$ 4557
201.1Carrier definition 4557
201.2The certificate 4557

202A Concrete Naming Certificate for Matroid$^{\uparrow}$ 4561
202.1Carrier definition 4561
202.2The certificate 4561

203A Concrete Naming Certificate for Ramsey$^{\uparrow}$ 4570
203.1Carrier definition 4570
203.2The certificate 4570

204A Concrete Naming Certificate for TopGroup$^{\uparrow}$ 4574
204.1Carrier definition 4574
204.2The certificate 4574
204.3Root source-fiber obligation packet 4590
204.4Root source-fiber export packet 4592
204.5Root operation source packet 4595
204.6Public threshold frontier 4596
204.7TopGroup root threshold downstream package 4599
204.8TopGroup root downstream obligation triad 4601

205A Concrete Naming Certificate for TopVecSpace$^{\uparrow}$ 4605
205.1Carrier definition 4605
205.2The certificate 4606

206A Concrete Naming Certificate for AffineSpace$^{\uparrow}$ 4609
206.1Carrier definition 4609
206.2The certificate 4610

207A Concrete Naming Certificate for ProjectiveSpace$^{\uparrow}$ 4617
207.1Carrier definition 4617
207.2The certificate 4618

208A Concrete Naming Certificate for ConvexSet$^{\uparrow}$ 4620
208.1Carrier definition 4620
208.2The certificate 4620

209A Concrete Naming Certificate for Polytope$^{\uparrow}$ **4631**
209.1Carrier definition 4631
209.2The certificate 4632

210A Concrete Naming Certificate for VectorBundle$^{\uparrow}$ **4636**
210.1Carrier definition 4636
210.2The certificate 4637

211A Concrete Naming Certificate for CStarAlg$^{\uparrow}$ **4640**
211.1Carrier definition 4640
211.2The certificate 4640

212A Concrete Naming Certificate for VonNeumannAlg$^{\uparrow}$ **4644**
212.1Carrier definition 4644
212.2The certificate 4644

213A Concrete Naming Certificate for OperatorIdeal$^{\uparrow}$ **4647**
213.1Carrier definition 4647
213.2The certificate 4648

214A Concrete Naming Certificate for SpectralTheorem$^{\uparrow}$ **4657**
214.1Carrier definition 4657
214.2The certificate 4657

215A Concrete Naming Certificate for Nuclear$^{\uparrow}$ **4661**
215.1Carrier definition 4661
215.2The certificate 4662

216A Concrete Naming Certificate for Factor$^{\uparrow}$ **4665**
216.1Carrier definition 4665
216.2The certificate 4667

217A Concrete Naming Certificate for QuantumState$^{\uparrow}$ **4670**
217.1Carrier definition 4670
217.2The certificate 4670

218A Concrete Naming Certificate for Observable$^{\uparrow}$ **4684**
218.1Carrier definition 4684
218.2The certificate 4685

219A Concrete Naming Certificate for UnitaryGroup$^{\uparrow}$ **4694**
219.1Carrier definition 4694
219.2The certificate 4695

220A Concrete Naming Certificate for DensityMatrix$^{\uparrow}$ **4699**
220.1Carrier definition 4699
220.2The certificate 4699

221A Concrete Naming Certificate for QuantumChannel$^{\uparrow}$ **4705**
221.1Carrier definition 4705
221.2The certificate 4705

222A Concrete Naming Certificate for Entanglement$^{\uparrow}$ **4717**
222.1Carrier definition 4717
222.2The certificate 4718

223A Concrete Naming Certificate for NewtonIteration$^{\uparrow}$ **4720**
223.1Carrier definition 4720
223.2The certificate 4720

224A Concrete Naming Certificate for Interpolation$^{\uparrow}$ **4726**
224.1Carrier definition 4726
224.2The certificate 4728

225A Concrete Naming Certificate for Approximation$^{\uparrow}$ **4730**
225.1Carrier definition 4730
225.2The certificate 4731

226A Concrete Naming Certificate for Fft$^{\uparrow}$ **4735**
226.1Carrier definition 4735
226.2The certificate 4737

227A Concrete Naming Certificate for Quadrature$^{\uparrow}$ **4740**
227.1Carrier definition 4740
227.2The certificate 4740

228A Concrete Naming Certificate for FisherInfo$^{\uparrow}$ **4749**
228.1Carrier definition 4749
228.2The certificate 4753

229A Concrete Naming Certificate for StatManifold$^{\uparrow}$ **4756**
229.1Carrier definition 4756
229.2The certificate 4758

230A Concrete Naming Certificate for Estimator$^{\uparrow}$ **4760**
230.1Carrier definition 4760
230.2The certificate 4760

231A Concrete Naming Certificate for HypothesisTest$^{\uparrow}$ **4763**
231.1Carrier definition 4763
231.2The certificate 4763

232A Concrete Naming Certificate for Bayesian$^{\uparrow}$ **4767**
232.1Carrier definition 4767
232.2The certificate 4767

233A Concrete Naming Certificate for NetworkFlow↑ 4773
233.1Carrier definition 4773
233.2The certificate 4773

234A Concrete Naming Certificate for Matching↑ 4783
234.1Carrier definition 4783
234.2The certificate 4783

235A Concrete Naming Certificate for LPDuality↑ 4790
235.1Carrier definition 4790
235.2The certificate 4790

236A Concrete Naming Certificate for ShortestPath↑ 4805
236.1Carrier definition 4805
236.2The certificate 4805

237A Concrete Naming Certificate for SpanningTree↑ 4810
237.1Carrier definition 4810
237.2The certificate 4812

238A Concrete Naming Certificate for SimplicialComplex↑ 4813
238.1Carrier definition 4813
238.2The certificate 4813

239A Concrete Naming Certificate for PersistentHom↑ 4822
239.1Carrier definition 4822
239.2The certificate 4824

240A Concrete Naming Certificate for Mapper↑ 4826
240.1Carrier definition 4826
240.2The certificate 4828

241A Concrete Naming Certificate for ErrorCode↑ 4829
241.1Carrier definition 4829
241.2The certificate 4829

242A Concrete Naming Certificate for Hash↑ 4832
242.1Carrier definition 4832
242.2The certificate 4832

243A Concrete Naming Certificate for PublicKey↑ 4842
243.1Carrier definition 4842
243.2The certificate 4842

244A Concrete Naming Certificate for ZeroKnowledge↑ 4847
244.1Carrier definition 4847
244.2The certificate 4847

245A Concrete Naming Certificate for CharacterTheory$^{\uparrow}$ **4851**
245.1Carrier definition 4851
245.2The certificate 4853

246A Concrete Naming Certificate for RepresentationRing$^{\uparrow}$ **4856**
246.1Carrier definition 4856
246.2The certificate 4858

247A Concrete Naming Certificate for InducedRep$^{\uparrow}$ **4866**
247.1Carrier definition 4866
247.2The certificate 4867

248A Concrete Naming Certificate for VermaModule$^{\uparrow}$ **4878**
248.1Carrier definition 4878
248.2The certificate 4878

249A Concrete Naming Certificate for Hypergeometric$^{\uparrow}$ **4880**
249.1Carrier definition 4880
249.2The certificate 4882

250A Concrete Naming Certificate for Bessel$^{\uparrow}$ **4885**
250.1Carrier definition 4885
250.2The certificate 4885

251A Concrete Naming Certificate for ThetaFunction$^{\uparrow}$ **4890**
251.1Carrier definition 4890
251.2The certificate 4890

252A Concrete Naming Certificate for ModularForm$^{\uparrow}$ **4894**
252.1Carrier definition 4894
252.2The certificate 4894

253A Concrete Naming Certificate for ClebschGordan$^{\uparrow}$ **4897**
253.1Carrier definition 4897
253.2The certificate 4897

254A Concrete Naming Certificate for Knot$^{\uparrow}$ **4901**
254.1Carrier definition 4901
254.2The certificate 4902

255A Concrete Naming Certificate for JonesPolynomial$^{\uparrow}$ **4910**
255.1Carrier definition 4910
255.2The certificate 4910

256A Concrete Naming Certificate for ThreeManifold$^{\uparrow}$ **4918**
256.1Carrier definition 4918
256.2The certificate 4919

257A Concrete Naming Certificate for Iwasawa$^{\uparrow}$ 4921
257.1Carrier definition 4921
257.2The certificate 4921

258A Concrete Naming Certificate for ClassField$^{\uparrow}$ 4926
258.1Carrier definition 4926
258.2The certificate 4926

259A Concrete Naming Certificate for Regulator$^{\uparrow}$ 4930
259.1Carrier definition 4930
259.2The certificate 4931

260A Concrete Naming Certificate for Stack$^{\uparrow}$ 4935
260.1Carrier definition 4935
260.2The certificate 4937

261A Concrete Naming Certificate for StoneDuality$^{\uparrow}$ 4938
261.1Carrier definition 4938
261.2The certificate 4939

262A Concrete Naming Certificate for GelfandDuality$^{\uparrow}$ 4941
262.1Carrier definition 4941
262.2The certificate 4942

263A Concrete Naming Certificate for PontryaginDuality$^{\uparrow}$ 4947
263.1Carrier definition 4947
263.2The certificate 4947

264A Concrete Naming Certificate for TannakaKrein$^{\uparrow}$ 4950
264.1Carrier definition 4950
264.2The certificate 4951

265A Concrete Naming Certificate for CurryHoward$^{\uparrow}$ 4953
265.1Carrier definition 4953
265.2The certificate 4953

266A Concrete Naming Certificate for RiemannHilbert$^{\uparrow}$ 4961
266.1Carrier definition 4961
266.2The certificate 4962

267A Concrete Naming Certificate for KoszulDuality$^{\uparrow}$ 4965
267.1Carrier definition 4965
267.2The certificate 4967

268A Concrete Naming Certificate for Langlands$^{\uparrow}$ 4969
268.1Carrier definition 4969
268.2The certificate 4970

269A Concrete Naming Certificate for NoetherSymmetry$^{\uparrow}$ 4972
269.1Carrier definition . . . 4972
269.2The certificate . . . 4973

270A Concrete Naming Certificate for GeomQuantization$^{\uparrow}$ 4976
270.1Carrier definition . . . 4976
270.2The certificate . . . 4979

271A Concrete Naming Certificate for DeformQuantization$^{\uparrow}$ 4981
271.1Carrier definition . . . 4981
271.2The certificate . . . 4981

272A Concrete Naming Certificate for ChernWeil$^{\uparrow}$ 4987
272.1Carrier definition . . . 4987
272.2The certificate . . . 4992

273A Concrete Naming Certificate for AtiyahSinger$^{\uparrow}$ 4995
273.1Carrier definition . . . 4995
273.2The certificate . . . 4995

274A Concrete Naming Certificate for HodgeBridge$^{\uparrow}$ 5001
274.1Carrier definition . . . 5001
274.2The certificate . . . 5004

275A Concrete Naming Certificate for MirrorSymmetry$^{\uparrow}$ 5005
275.1Carrier definition . . . 5005
275.2The certificate . . . 5005

276A Concrete Naming Certificate for AxisNat$^{\uparrow}$ 5009
276.1The E0-spine carrier . . . 5009
276.2The certificate . . . 5010
276.3Distinction from Nat$^{\uparrow}$ . . . 5011

277A Concrete Naming Certificate for AxisAdd$^{\uparrow}$ 5013
277.1Continuation on the zero-spine . . . 5013
277.2The certificate . . . 5014
277.3Horizon obligations . . . 5015

278A Concrete Naming Certificate for ZeckendorfNormal$^{\uparrow}$ 5018
278.1The normal-form predicate . . . 5018
278.2The certificate . . . 5019
278.3Horizon: Fibonacci value reading . . . 5022

279A Concrete Naming Certificate for FullAxis$^{\uparrow}$ 5024
279.1The prefix thread . . . 5024
279.2Seal certificate . . . 5024
279.3The boundary marker . . . 5024
279.4The certificate . . . 5025
279.5Horizon obligations . . . 5027

280A Concrete Naming Certificate for TuringMachine$^{\uparrow}$ **5030**
280.1Carrier definition 5030
280.2The certificate 5033

281A Concrete Naming Certificate for RecursiveFn$^{\uparrow}$ **5036**
281.1Carrier definition 5036

282A Concrete Naming Certificate for ComplexityClass$^{\uparrow}$ **5040**
282.1Carrier definition 5040

283A Concrete Naming Certificate for Kolmogorov$^{\uparrow}$ **5044**
283.1Carrier definition 5044

284A Concrete Naming Certificate for BusyBeaver$^{\uparrow}$ **5047**
284.1Carrier definition 5047

285A Concrete Naming Certificate for AbelRuffini$^{\uparrow}$ **5052**
285.1Carrier definition 5052

286A Concrete Naming Certificate for SolvableRadicals$^{\uparrow}$ **5057**
286.1Carrier definition 5057

287A Concrete Naming Certificate for DiffGalois$^{\uparrow}$ **5060**
287.1Carrier definition 5060

288A Concrete Naming Certificate for GoedelIncompleteness$^{\uparrow}$ **5063**
288.1Carrier definition 5063

289A Concrete Naming Certificate for Belief$^{\uparrow}$ **5066**
289.1Carrier definition 5066

290A Concrete Naming Certificate for Policy$^{\uparrow}$ **5072**
290.1Carrier definition 5072

291A Concrete Naming Certificate for DyadicRatCore$^{\uparrow}$ **5078**
291.1Carrier and classifier 5078
291.2Certificate surface 5082

292A Concrete Naming Certificate for CauchyModulus$^{\uparrow}$ **5162**
292.1Carrier and classifier 5162

293A Concrete Naming Certificate for DyadicAbsoluteValue$^{\uparrow}$ **5168**

294A Concrete Naming Certificate for ControlObservability$^{\uparrow}$ **5175**
294.1Carrier definition 5175
294.2The certificate 5175

295A Concrete Naming Certificate for KalmanFilter$^{\uparrow}$ **5179**
295.1Finite carrier 5179

296A Concrete Naming Certificate for LQR$^{\uparrow}$ 5184
296.1Carrier definition . . . 5184
296.2The certificate . . . 5184

297A Concrete Naming Certificate for ControlControllability$^{\uparrow}$ 5190
297.1Carrier definition . . . 5190
297.2The certificate . . . 5191

298A Concrete Naming Certificate for SpectralMeasure$^{\uparrow}$ 5196
298.1Carrier definition . . . 5196
298.2The certificate . . . 5198

299A Concrete Naming Certificate for Holonomy$^{\uparrow}$ 5202
299.1Carrier definition . . . 5202
299.2The certificate . . . 5203

300A Concrete Naming Certificate for KKT$^{\uparrow}$ 5207
300.1Carrier definition . . . 5207
300.2The certificate . . . 5207

301A Concrete Naming Certificate for RegSeqRat$^{\uparrow}$ 5214
301.1Carrier and classifier . . . 5214
301.2Certificate obligations . . . 5215
301.3RegSeqRat Terminal Four-Face Axis Separation . . . 5245

302A Concrete Naming Certificate for IntervalArithmetic$^{\uparrow}$ 5380
302.1Endpoint carrier . . . 5380
302.2Operation ledgers . . . 5381
302.3Public surface . . . 5382

303A Concrete Naming Certificate for ModularArithmetic$^{\uparrow}$ 5386
303.1Residue source . . . 5386

304A Concrete Naming Certificate for ApartnessReal$^{\uparrow}$ 5390
304.1Carrier and classifier . . . 5390
304.2Certificate obligations . . . 5391

305A Concrete Naming Certificate for BraidGroup$^{\uparrow}$ 5397
305.1Carrier And Ledger . . . 5397
305.2Certificate Surface . . . 5398

306A Concrete Naming Certificate for DyadicInterval$^{\uparrow}$ 5402
306.1Carrier and refinement ledger . . . 5402

307A Concrete Naming Certificate for Riccati$^{\uparrow}$ 5409

308A Concrete Naming Certificate for CofibrantReplacement$^{\uparrow}$ 5413
308.1Finite factorization source . . . 5413
308.2Certificate obligations . . . 5414

309 A Locatedness Route Surface for LocatedReal↑ **5417**
309.1 Carrier and classifier 5417

310 A Concrete Naming Certificate for SemidefiniteCone↑ **5425**
310.1 Carrier packet 5425

311 A Concrete Naming Certificate for DyadicBall↑ **5430**
311.1 Finite center-radius carrier 5430

312 A Concrete Naming Certificate for Monodromy↑ **5440**
312.1 Carrier definition 5440
312.2 The certificate 5441

313 A Concrete Naming Certificate for Lyapunov↑ **5445**

314 A Concrete Naming Certificate for DyadicCompletion↑ **5449**
314.1 Carrier Surface 5449

315 A Concrete Naming Certificate for FastCauchySeq↑ **5457**
315.1 Explicit-rate carrier and classifier 5457

316 A Concrete Naming Certificate for SignedDigitStream↑ **5467**
316.1 Finite signed-digit windows 5467

317 A Concrete Naming Certificate for RealBall↑ **5479**
317.1 Finite dyadic-window carrier 5479

318 A Concrete Naming Certificate for EulerLagrange↑ **5484**
318.1 Finite variational packet 5484

319 A Concrete Naming Certificate for GradedAlgebra↑ **5490**
319.1 Finite graded-algebra carrier 5490

320 A Concrete Naming Certificate for NestedInterval↑ **5495**
320.1 Finite dyadic nested packets 5495

321 A Concrete Naming Certificate for FenchelDuality↑ **5502**
321.1 Finite primal-dual packet 5502

322 DyadicPrecision↑ naming certificate construction **5507**
322.1 Finite precision schedule 5507

323 A Concrete Naming Certificate for FastCauchy↑ **5516**
323.1 Finite modulus-normalized Cauchy names 5516

324 A Concrete Naming Certificate for LyapunovStability↑ **5528**
324.1 Carrier definition 5528
324.2 Finite stability surface 5528

325A Concrete Naming Certificate for BousfieldLocalization$^{\uparrow}$ **5531**
325.1Finite localizing packet 5531

326A Concrete Naming Certificate for OptimalTransport$^{\uparrow}$ **5537**
326.1Finite-support transport packets 5537

327A Concrete Naming Certificate for Semiring$^{\uparrow}$ **5545**

328A Concrete Naming Certificate for NestedDyadicInterval$^{\uparrow}$ **5550**
328.1Finite decreasing dyadic chains 5550

329A Concrete Naming Certificate for StateSpaceModel$^{\uparrow}$ **5559**
329.1Finite state-transition carrier 5559
329.2Control handoff surface 5560

330A Concrete Naming Certificate for FiniteVector$^{\uparrow}$ **5564**
330.1Finite-vector carrier 5564

331A Concrete Naming Certificate for Divisibility$^{\uparrow}$ **5571**

332A Concrete Naming Certificate for ModulusOfConvergence$^{\uparrow}$ **5576**

333A Concrete Naming Certificate for CauchyFilter$^{\uparrow}$ **5583**

334A Concrete Naming Certificate for BaireSpace$^{\uparrow}$ **5590**

335A Concrete Naming Certificate for DyadicApproximation$^{\uparrow}$ **5599**
335.1Finite dyadic observation packets 5599
335.2Certificate obligations 5600

336A Concrete Naming Certificate for ComputableReal$^{\uparrow}$ **5637**

337A Concrete Naming Certificate for LagrangianMechanics$^{\uparrow}$ **5645**

338A Finite Observation Filter Surface for CauchyFilter$^{\uparrow}$ **5651**

339A Concrete Naming Certificate for UniformSpace$^{\uparrow}$ **5658**

340A Concrete Naming Certificate for LocatedCauchy$^{\uparrow}$ **5665**

341A Concrete Naming Certificate for PolePlacement$^{\uparrow}$ **5672**

342A Concrete Naming Certificate for ContinuedFraction$^{\uparrow}$ **5676**

343A Concrete Naming Certificate for BishopReal$^{\uparrow}$ **5685**

344A Concrete Naming Certificate for StoppingTime$^{\uparrow}$ **5691**

345A Concrete Naming Certificate for FreeMonoid$^{\uparrow}$ **5695**

346A Concrete Naming Certificate for BitVector$^{\uparrow}$ **5700**

347A Concrete Naming Certificate for RealLimit↑ **5706**

348A Concrete Naming Certificate for ModelPredictiveControl↑ **5716**

349A Concrete Naming Certificate for Trie↑ **5722**

350A Concrete Naming Certificate for RationalStream↑ **5736**

351A Concrete Naming Certificate for RegularLanguage↑ **5747**

352A Concrete Naming Certificate for FiniteMap↑ **5755**
352.1Finite-map source 5755

353A Concrete Naming Certificate for RegularityModulus↑ **5760**

354A Concrete Naming Certificate for RationalInterval↑ **5768**

355A Concrete Naming Certificate for RationalBall↑ **5783**

356A Concrete Naming Certificate for ContextFreeGrammar↑ **5793**

357A Concrete Naming Certificate for ValidatedNumerics↑ **5804**

358A Concrete Naming Certificate for DyadicMesh↑ **5813**

359A Concrete Naming Certificate for RealDiagonalCompletion↑ **5820**

360A Concrete Naming Certificate for RealCauchyCompletion↑ **5828**

361A Concrete Naming Certificate for DyadicBisection↑ **5864**

362A Concrete Naming Certificate for StationaryRationalDiagonal↑ **5871**

363A Concrete Naming Certificate for SignedDigitReal↑ **5877**

364A Concrete Naming Certificate for StationaryRationalCauchy↑ **5888**

365A Concrete Naming Certificate for FiniteWindowEnvelope↑ **5894**

366A Concrete Naming Certificate for DyadicSubdivision↑ **5901**

367A Concrete Naming Certificate for FastConvergentSeries↑ **5908**

368A Concrete Naming Certificate for FastSeriesProduct↑ **5917**

369A Concrete Naming Certificate for UniformModulus↑ **5923**

370A Concrete Naming Certificate for GoldenMeanShift↑ **5951**
370.1Finite binary windows 5951

371A Concrete Naming Certificate for RegularCauchyName↑ **5957**

372A Concrete Naming Certificate for RegularCauchyDiagonal↑ **5994**

373A Concrete Naming Certificate for RegularCauchyFamily↑ **6008**

374A Concrete Naming Certificate for DiagonalLimit↑ **6016**

375A Concrete Naming Certificate for DiagonalModulus↑ **6021**

376A Concrete Naming Certificate for CauchyRegularization↑ **6026**

377A Concrete Naming Certificate for UnitDisk↑ **6033**

378A Concrete Naming Certificate for CauchyCompletionMonad↑ **6040**

379A Concrete Naming Certificate for DyadicTailBound↑ **6062**

380A Concrete Naming Certificate for NullSequence↑ **6072**

381A Concrete Naming Certificate for RegularCauchySubsequence↑ **6081**

382A Concrete Naming Certificate for DyadicSpline↑ **6088**

383A Concrete Naming Certificate for OptionalStopping↑ **6095**

384A Concrete Naming Certificate for FoldMomentKernel↑ **6101**

385A Concrete Naming Certificate for FiniteErrorBudget↑ **6106**

386A Concrete Naming Certificate for CauchyLimitSeal↑ **6115**

386.1 CauchyLimitSeal Synchronizer Dependency 6168
386.2 CauchyLimitSeal regular tail-selector sibling link 6182
386.3 CauchyLimitSeal finite-window lattice dependency 6187
386.4 CauchyLimitSeal root L10 sibling route 6195
386.5 CauchyLimitSeal DyadicRatCore terminal dependency 6200
386.6 CauchyCriterion Lattice Dependency 6205
386.7 CauchyLimitSeal RealCompletionExactBoundary sibling route 6205

387A Concrete Naming Certificate for CauchyTailComparison↑ **6206**

388A Concrete Naming Certificate for CompactUniformContinuity↑ **6215**

389A Concrete Naming Certificate for NestedIntervalIntersection↑ **6241**

390A Concrete Naming Certificate for RegularCauchyLimitModulus↑ **6247**

391A Concrete Naming Certificate for UniformContinuity↑ **6255**

392A Concrete Naming Certificate for CauchyCriterion↑ **6259**

393A Concrete Naming Certificate for StreamDiagonalSelector↑ **6295**

394A Concrete Naming Certificate for LocatedInterval↑ **6299**

395A Concrete Naming Certificate for PicardContraction$^{\uparrow}$ **6317**

396A Concrete Naming Certificate for BanachFixedPoint$^{\uparrow}$ **6343**

397A Concrete Naming Certificate for CompletionUniversalProperty$^{\uparrow}$ **6350**

398A Concrete Naming Certificate for BoundedVariation$^{\uparrow}$ **6357**

399A Concrete Naming Certificate for WeierstrassMTest$^{\uparrow}$ **6367**

400A Concrete Naming Certificate for FanTheorem$^{\uparrow}$ **6376**

401A Concrete Naming Certificate for FibonacciCube$^{\uparrow}$ **6381**

402A Concrete Naming Certificate for HausdorffCompletion$^{\uparrow}$ **6384**

403A Concrete Naming Certificate for CompletionReflection$^{\uparrow}$ **6402**

404A Concrete Naming Certificate for LocatedCut$^{\uparrow}$ **6410**

405A Concrete Naming Certificate for CofinalSubsequence$^{\uparrow}$ **6420**

406A Concrete Naming Certificate for CauchySequenceSpace$^{\uparrow}$ **6425**

407A Concrete Naming Certificate for SqueezeReal$^{\uparrow}$ **6443**

408A Concrete Naming Certificate for FoldDefectStokes$^{\uparrow}$ **6450**

409A Concrete Naming Certificate for CauchyConvolution$^{\uparrow}$ **6454**

410A Concrete Naming Certificate for LipschitzMap$^{\uparrow}$ **6460**

411Regular Cauchy Comparison NameCert Construction **6470**

412A Concrete Naming Certificate for ArzelaAscoli$^{\uparrow}$ **6483**

413A Concrete Naming Certificate for EquicontinuityFamily$^{\uparrow}$ **6489**

414RegularCauchyInterleaving NameCert Construction **6493**

415A Concrete Naming Certificate for CauchyRate$^{\uparrow}$ **6504**

416A Concrete Naming Certificate for RegularCauchyInterleave$^{\uparrow}$ **6514**

417RegularCauchySum NameCert Construction **6520**

418A Concrete Naming Certificate for BinaryExpansion$^{\uparrow}$ **6533**

419A Concrete Naming Certificate for BinaryEndpointNormalization$^{\uparrow}$ **6541**

420A Concrete Naming Certificate for BinaryEndpointUniqueness$^{\uparrow}$ **6544**

421 CauchyPairing NameCert Construction **6547**

422 A Concrete Naming Certificate for StationaryPartitionDiagonal$^{\uparrow}$ **6554**

423 A Concrete Naming Certificate for CauchyCompletionFunctor$^{\uparrow}$ **6558**

424 A Concrete Naming Certificate for ProductMetric$^{\uparrow}$ **6569**

425 A Concrete Naming Certificate for CauchyErrorCertificate$^{\uparrow}$ **6577**

426 A Concrete Naming Certificate for CompletionFunctor$^{\uparrow}$ **6588**

427 A Concrete Naming Certificate for RegularLimitUniqueness$^{\uparrow}$ **6596**

428 DyadicStepFunction NameCert Construction **6609**

429 A Concrete Naming Certificate for RegularCauchyDifference$^{\uparrow}$ **6622**

430 A Concrete Naming Certificate for ArchimedeanReal$^{\uparrow}$ **6629**

431 A Concrete Naming Certificate for ObserverRegular$^{\uparrow}$ **6639**

432 A Concrete Naming Certificate for FibonacciInverseLimit$^{\uparrow}$ **6646**

433 A Concrete Naming Certificate for DyadicRounding$^{\uparrow}$ **6649**

434 A Concrete Naming Certificate for SubshiftOfFiniteType$^{\uparrow}$ **6654**

434.1 Finite adjacency windows 6654

435 A Concrete Naming Certificate for UniformCauchyCriterion$^{\uparrow}$ **6659**

436 A Concrete Naming Certificate for DedekindCut$^{\uparrow}$ **6751**

437 A Concrete Naming Certificate for TransportedStationaryWindowSeal$^{\uparrow}$ **6755**

438 A Concrete Naming Certificate for MonotoneCauchy$^{\uparrow}$ **6795**

439 A Concrete Naming Certificate for IsometricEmbedding$^{\uparrow}$ **6809**

440 A Concrete Naming Certificate for Sobolev$^{\uparrow}$ **6817**

441 A Concrete Naming Certificate for AscoliModulus$^{\uparrow}$ **6844**

442 A Concrete Naming Certificate for TaylorModel$^{\uparrow}$ **6851**

443 Regular Cauchy Limit Classifier NameCert Construction **6857**

444 A Concrete Naming Certificate for CollisionKernel$^{\uparrow}$ **6867**

445 A Concrete Naming Certificate for DyadicMidpoint$^{\uparrow}$ **6875**

446 A Concrete Naming Certificate for DyadicCover$^{\uparrow}$ **6885**

447A Concrete Naming Certificate for CollisionKernelSpectrum$^{\uparrow}$ **6895**

448A Concrete Naming Certificate for CauchyOscillation$^{\uparrow}$ **6901**

449A Concrete Naming Certificate for RealScheduleFusion$^{\uparrow}$ **6924**

450A Concrete Naming Certificate for RealCompleteness$^{\uparrow}$ **6931**

451A Concrete Naming Certificate for FinitePhaseSampling$^{\uparrow}$ **6958**

452A Concrete Naming Certificate for FiniteLebesgueNumber$^{\uparrow}$ **6962**
452.1 Finite Lebesgue-number Phase Real terminal radius readiness 7030

453A Concrete Naming Certificate for DoCalculus$^{\uparrow}$ **7035**

454 LocatedModulusCompletion NameCert Construction **7040**

455A Concrete Naming Certificate for ReachabilityGramian$^{\uparrow}$ **7048**
455.1 Carrier definition 7048
455.2 Certificate surface 7049

456A Concrete Naming Certificate for TransferOperator$^{\uparrow}$ **7051**

457A Concrete Naming Certificate for HankelVandermondeUp **7055**

458A Concrete Naming Certificate for RegularCauchyTailSelector$^{\uparrow}$ **7059**

459A Concrete Naming Certificate for RegularCauchyNegation$^{\uparrow}$ **7066**

460A Concrete Naming Certificate for CauchyDoubleSequence$^{\uparrow}$ **7074**

461A Concrete Naming Certificate for RegularCauchyProduct$^{\uparrow}$ **7082**

462A Concrete Naming Certificate for CauchyModulusMeet$^{\uparrow}$ **7091**

463A Concrete Naming Certificate for UniformLimit$^{\uparrow}$ **7109**

464A Concrete Naming Certificate for CauchyProduct$^{\uparrow}$ **7121**

465A Concrete Naming Certificate for RegularCauchyScale$^{\uparrow}$ **7202**

466A Concrete Naming Certificate for RegularCauchyTriangleBound$^{\uparrow}$ **7214**

467A Concrete Naming Certificate for DyadicEmbedding$^{\uparrow}$ **7220**

468A Concrete Naming Certificate for FiniteMultiHistPacket$^{\uparrow}$ **7227**

469A Concrete Naming Certificate for ContinuationTrace$^{\uparrow}$ **7232**

470A Concrete Naming Certificate for StreamLimitReflection$^{\uparrow}$ **7238**

471A Concrete Naming Certificate for UnaryContinuationCategory$^{\uparrow}$ **7241**

472A Concrete Naming Certificate for TruthTotalReflection$^{\uparrow}$ **7244**

473A Concrete Naming Certificate for CertificateCompiler$^{\uparrow}$ **7252**

474A Concrete Naming Certificate for MultiHistConfig$^{\uparrow}$ **7292**

475A Concrete Naming Certificate for TransportedStationaryDiagonal$^{\uparrow}$ **7297**

476A Concrete Naming Certificate for TypeClassifierMembership$^{\uparrow}$ **7300**

477A Concrete Naming Certificate for ObserverHistoryIdentity$^{\uparrow}$ **7305**

478A Concrete Naming Certificate for CrossHistCausal$^{\uparrow}$ **7309**

479A Concrete Naming Certificate for ObserverHistIdentity$^{\uparrow}$ **7313**

480A Concrete Naming Certificate for KernelCategory$^{\uparrow}$ **7317**

481A Concrete Naming Certificate for NameCertMorphism$^{\uparrow}$ **7319**

482A Concrete Naming Certificate for ContinuationAutomaton$^{\uparrow}$ **7326**

483A Concrete Naming Certificate for SubjectReduction$^{\uparrow}$ **7334**

484A Concrete Naming Certificate for FiniteKernelCategory$^{\uparrow}$ **7337**

485A Concrete Naming Certificate for ContinuationMonad$^{\uparrow}$ **7353**

486A Concrete Naming Certificate for GeneratorClosure$^{\uparrow}$ **7405**

487A Concrete Naming Certificate for ClosureWitnessLedger$^{\uparrow}$ **7415**

488A Continuation-Ledger Certificate for KernelCategory$^{\uparrow}$ **7419**

489A Concrete Naming Certificate for ClassifierTyping$^{\uparrow}$ **7438**

490A Concrete Naming Certificate for HaltingObstruction$^{\uparrow}$ **7443**

491A Concrete Naming Certificate for HaltingDistinction$^{\uparrow}$ **7453**

492A Concrete Naming Certificate for HaltingDistinctionTrace$^{\uparrow}$ **7473**

493A Concrete Naming Certificate for DecidableRefutation$^{\uparrow}$ **7478**

494A Concrete Naming Certificate for SelectorLedger$^{\uparrow}$ **7482**

495A Concrete Naming Certificate for HaltingDistinctionLimit$^{\uparrow}$ **7485**

496A Concrete Naming Certificate for HaltingDiagonal$^{\uparrow}$ **7489**

497A Concrete Naming Certificate for DecidableRefutationBoundary$^{\uparrow}$ **7493**

498A Concrete Naming Certificate for TranscendentalInductionSocket$^{\uparrow}$ **7496**

499A Concrete Naming Certificate for OtherMindsCommitment$^{\uparrow}$ **7500**

500A Concrete Naming Certificate for RefutationBoundary$^{\uparrow}$ **7505**

501Constructive Refutation NameCert Construction **7510**

502A Concrete Naming Certificate for ConstructiveRefutationBoundary$^{\uparrow}$ **7513**

503A Concrete Naming Certificate for WitnessedRefutation$^{\uparrow}$ **7515**

504A Concrete Naming Certificate for FiniteTraceGapSocket$^{\uparrow}$ **7520**

505A Concrete Naming Certificate for CausalCommitment$^{\uparrow}$ **7526**

506A Concrete Naming Certificate for RussellSelfMembershipBoundary$^{\uparrow}$ **7540**

507A Concrete Naming Certificate for StableNegationBoundary$^{\uparrow}$ **7543**

508A Concrete Naming Certificate for FiniteRefutationBoundary$^{\uparrow}$ **7547**

509A Concrete Naming Certificate for LocalityCell$^{\uparrow}$ **7551**

510A Concrete Naming Certificate for GroundRefutation$^{\uparrow}$ **7555**

511A Concrete Naming Certificate for RefutationWitness$^{\uparrow}$ **7560**

512A Concrete Naming Certificate for RefuterTrace$^{\uparrow}$ **7564**

513A Concrete Naming Certificate for ObserverLocalityCell$^{\uparrow}$ **7568**

514ObserverFilterUp NameCert construction **7576**

515A Concrete Naming Certificate for UnwitnessedExistenceClosure$^{\uparrow}$ **7583**

516A Concrete Naming Certificate for EmptyFableLedger$^{\uparrow}$ **7590**

517A Concrete Naming Certificate for RussellBoundary$^{\uparrow}$ **7594**

518A Concrete Naming Certificate for RussellAnalysisLadder$^{\uparrow}$ **7598**

519A Concrete Naming Certificate for InscriptionEvent$^{\uparrow}$ **7602**

520A Concrete Naming Certificate for InscriptionGapLedger$^{\uparrow}$ **7606**

521A Concrete Naming Certificate for ContourIntegralOperation$^{\uparrow}$ **7609**

522A Concrete Naming Certificate for RecursorInducedNameCert$^{\uparrow}$ **7615**

523A Concrete Naming Certificate for TranscendentalSupplyTaxonomy$^{\uparrow}$ **7618**

524 A Concrete Naming Certificate for AuditMembrane↑ 7624

525 A Concrete Naming Certificate for AuthorizedGeneratorRecursor↑ 7629
525.1 AuthorizedGeneratorRecursor displayed route surface 7657

526 A Concrete Naming Certificate for AuditGateBoundary↑ 7744

527 Closed-Term Substitution Boundary NameCert Construction 7755

528 A Concrete Naming Certificate for SupplySocketLedger↑ 7818

529 A Concrete Naming Certificate for RecursorAuthorization↑ 7823

530 A Concrete Naming Certificate for UnaryDirectionBridge↑ 7826

531 ClosedSubstitutionSeal NameCert Construction 7838

532 A Concrete Naming Certificate for BinderBudgetSeal↑ 7841

533 A Concrete Naming Certificate for OperationalMembrane↑ 7846

534 A Concrete Naming Certificate for RecursorGenerator↑ 7850

535 A Concrete Naming Certificate for BoundedMonotoneCauchyWitness↑ 7855

536 A Concrete Naming Certificate for ApophaticSocket↑ 7954

537 A Concrete Naming Certificate for ClosedTermSubstitution↑ 7960

538 A Concrete Naming Certificate for ClosedSubstitutionBoundary↑ 7964

539 A Concrete Naming Certificate for CountableObservationSchedule↑ 7969

540 ClosedTermSubstitutionCompiler NameCert Construction 7976

541 A Concrete Naming Certificate for AuthorizedRecursor↑ 7983

542 A Concrete Naming Certificate for TranscendentalSupplyLedger↑ 7986

543 A Concrete Naming Certificate for ApophaticName↑ 7989

544 A Concrete Naming Certificate for ZCarry↑ 8043

545 A Concrete Naming Certificate for TasteGateAdmission↑ 8046

546 A Concrete Naming Certificate for KernelAuditWitness↑ 8050

547 A Concrete Naming Certificate for KernelAcceptanceAuditWitness↑ 8054

548 A Concrete Naming Certificate for CriticalLineWitness↑ 8066
548.0.1 Fixed-strip source lock 8230

549A Concrete Naming Certificate for KernelAcceptanceWitness$^{\uparrow}$ **8242**

550A Concrete Naming Certificate for CriticalStripZetaZeroWitness$^{\uparrow}$ **8246**

551A Concrete Naming Certificate for HostDelegationSocket$^{\uparrow}$ **8257**

552A Concrete Naming Certificate for TasteGateAuditTrace$^{\uparrow}$ **8261**

553A Concrete Naming Certificate for GeneratorInductionSeal$^{\uparrow}$ **8264**

554A Concrete Naming Certificate for AxisZeckendorfCannotClaim$^{\uparrow}$ **8267**

555A Concrete Naming Certificate for InductionClosureLedger$^{\uparrow}$ **8287**

556A Concrete Naming Certificate for GeneratorFixedPoint$^{\uparrow}$ **8290**

557A Concrete Naming Certificate for KernelAcceptanceTrace$^{\uparrow}$ **8303**

558A Zeckendorf-Variant Naming Certificate for ZCarry$^{\uparrow}$ **8307**

559A Concrete Naming Certificate for FiniteTailDiagonalSeal$^{\uparrow}$ **8311**

560RegularCauchyTailMeet NameCert construction **8317**

561A Concrete Naming Certificate for RealDiagonalWindowCoverage$^{\uparrow}$ **8366**

562A Concrete Naming Certificate for TypedReductionNormalForm$^{\uparrow}$ **8372**

563A Concrete Naming Certificate for TasteGateCompilerWitness$^{\uparrow}$ **8376**

564A Concrete Naming Certificate for TasteGateStability$^{\uparrow}$ **8380**

565A Concrete Naming Certificate for GeneratorInductionBoundary$^{\uparrow}$ **8384**

566A Concrete Naming Certificate for PropextTransportBoundary$^{\uparrow}$ **8387**

567A Zeckendorf Classifier Surface for ZCarry$^{\uparrow}$ **8393**

568A Concrete Naming Certificate for EffectiveReplacementLedger$^{\uparrow}$ **8397**

569A Concrete Naming Certificate for AnalyticContinuationSocket$^{\uparrow}$ **8400**

570A Concrete Naming Certificate for ZetaContinuationSocket$^{\uparrow}$ **8416**

571A Concrete Naming Certificate for DimLiftBoundary$^{\uparrow}$ **8421**

572A Concrete Naming Certificate for UnaryZeroSpineStandardIso$^{\uparrow}$ **8427**

573A Concrete Naming Certificate for RiemannZeroCountLedger$^{\uparrow}$ **8431**

574A Concrete Naming Certificate for CertificatePipelineStageBoundary$^{\uparrow}$ **8435**

575AspectChain NameCert Construction **8438**

576A Concrete Naming Certificate for FiniteWitnessedRefutation$^{\uparrow}$ **8443**

577A Concrete Naming Certificate for AnalyticContinuationOperation$^{\uparrow}$ **8448**

578A Concrete Naming Certificate for ConsciousObserverState$^{\uparrow}$ **8455**

579A Concrete Naming Certificate for BoundedSearchRefutation$^{\uparrow}$ **8467**

580A Concrete Naming Certificate for ZetaContinuationWitness$^{\uparrow}$ **8472**

581A Concrete Naming Certificate for ZetaFunctionalEquationLedger$^{\uparrow}$ **8511**

582A Concrete Naming Certificate for InscribedRoute$^{\uparrow}$ **8514**

583A Concrete Naming Certificate for ContourIntegralSocket$^{\uparrow}$ **8518**

584A Concrete Naming Certificate for ExternalityGate$^{\uparrow}$ **8522**

585A Concrete Naming Certificate for FableBranchWitness$^{\uparrow}$ **8525**

586A Concrete Naming Certificate for ReflectionLimitBoundary$^{\uparrow}$ **8529**

587A Concrete Naming Certificate for ContourSumWindow$^{\uparrow}$ **8533**

588A Concrete Naming Certificate for ZetaContinuationApplication$^{\uparrow}$ **8538**

589A Concrete Naming Certificate for CauchyNet$^{\uparrow}$ **8558**

590A Concrete Naming Certificate for ClassifierMorphism$^{\uparrow}$ **8563**

591A Concrete Naming Certificate for RuntimeCandidateBoundary$^{\uparrow}$ **8568**

592A Concrete Naming Certificate for FreeWillInscriptionCommitment$^{\uparrow}$ **8571**

593A Concrete Naming Certificate for RegularCauchyWitnessSchedule$^{\uparrow}$ **8574**

594A Concrete Naming Certificate for MaxCausalRate$^{\uparrow}$ **8592**

595A Concrete Naming Certificate for ClassifierBoundary$^{\uparrow}$ **8600**

596A Concrete Naming Certificate for SubordinateModulusCover$^{\uparrow}$ **8604**

597A Concrete Naming Certificate for ContinuationTermination$^{\uparrow}$ **8613**

598A Concrete Naming Certificate for TypeCheckingDerivationTrace$^{\uparrow}$ **8617**

599A Concrete Naming Certificate for AxisUnarySpineNormalizer$^{\uparrow}$ **8623**

600A Concrete Naming Certificate for RegularCauchyLimitTransport$^{\uparrow}$ **8626**

601A Concrete Naming Certificate for FiniteCauchyApproximation$^{\uparrow}$ **8635**

602A Concrete Naming Certificate for FullAxisRealRefusal$^{\uparrow}$ **8648**

603A Concrete Naming Certificate for HistTimeStream$^{\uparrow}$ **8651**

604A Concrete Naming Certificate for TranscendentalSupplyInscription$^{\uparrow}$ **8670**

605A Concrete Naming Certificate for StationarySealUniqueness$^{\uparrow}$ **8674**

606A Finite Value-Preservation Certificate for ZeckendorfCarryValue$^{\uparrow}$ **8677**

607A Concrete Naming Certificate for ObservationConservationLedger$^{\uparrow}$ **8681**

608A Concrete Naming Certificate for ZeckendorfCarryNormalization$^{\uparrow}$ **8684**

609A Concrete Naming Certificate for ObserverCoordinateFrame$^{\uparrow}$ **8689**

610A Concrete Naming Certificate for InscriptionPoint$^{\uparrow}$ **8692**

611A Concrete Naming Certificate for RealObservationBudget$^{\uparrow}$ **8696**

612A Concrete Naming Certificate for CauchyDiagonalFusion$^{\uparrow}$ **8712**

613A Concrete Naming Certificate for StationaryDiagonalWindowRoute$^{\uparrow}$ **8715**

614A Concrete Naming Certificate for DecompilerRightInverse$^{\uparrow}$ **8721**

615A Concrete Naming Certificate for ObserverSubjectBoundary$^{\uparrow}$ **8724**

616A Concrete Naming Certificate for ClosedObservationSystem$^{\uparrow}$ **8728**

617A Concrete Naming Certificate for LorentzFrameRate$^{\uparrow}$ **8731**

618A Concrete Naming Certificate for RealDiagonalMeshBudget$^{\uparrow}$ **8741**

619A Concrete Naming Certificate for FiniteNetMinimumFold$^{\uparrow}$ **8750**

620A Concrete Naming Certificate for KernelMorphismUp **8761**

621A Concrete Naming Certificate for BinderContextSubstitutionSeal$^{\uparrow}$ **8771**

622A Concrete Naming Certificate for ForbiddenAxiomAncestry$^{\uparrow}$ **8777**

623A Concrete Naming Certificate for FableMachineClock$^{\uparrow}$ **8782**

624A Concrete Naming Certificate for RegularCauchyZero$^{\uparrow}$ **8785**

625A Concrete Naming Certificate for TypePreservingCompiler$^{\uparrow}$ **8795**

626A Concrete Naming Certificate for StationaryDiagonalWindow$^{\uparrow}$ **8800**

627A Concrete Naming Certificate for GeneratorContinuationAudit$^{\uparrow}$ **8805**

628A Concrete Naming Certificate for ModulusIndexedTailCover$^{\uparrow}$ **8808**

629A Concrete Naming Certificate for NegativeNameBoundary$^{\uparrow}$ **8812**

630A Concrete Naming Certificate for NegativeCertificateBoundary$^{\uparrow}$ **8816**

631A Concrete Naming Certificate for ProductRegularCauchyLimit$^{\uparrow}$ **8821**

632Diagonal Cofinal Tail Name Certificate Construction **8825**

633A Concrete Naming Certificate for CausalContinuitySocket$^{\uparrow}$ **8840**

634A Concrete Naming Certificate for QuotientSoundnessBoundary$^{\uparrow}$ **8843**

635A Concrete Naming Certificate for RealInverse$^{\uparrow}$ **8877**

636A Concrete Naming Certificate for CauchyModulusComparison$^{\uparrow}$ **8883**

637A Concrete Naming Certificate for RealApproximationEnvelope$^{\uparrow}$ **8891**

638A Concrete Naming Certificate for FiniteWitnessRoute$^{\uparrow}$ **8902**

639A Concrete Naming Certificate for CompilerOptimizationEndomorphism$^{\uparrow}$ **8916**

640A Concrete Naming Certificate for ObserverKernelTrace$^{\uparrow}$ **8920**

641A Concrete Naming Certificate for CauchyModulusExtraction$^{\uparrow}$ **8924**

642A Concrete Naming Certificate for ProofTermErasureTrace$^{\uparrow}$ **8937**

643A Concrete Naming Certificate for CrossPerspectiveAlignment$^{\uparrow}$ **8943**

644A Concrete Naming Certificate for RegularCauchyMesh$^{\uparrow}$ **8947**

645A Concrete Naming Certificate for BoundedCauchyCover$^{\uparrow}$ **8951**

646A Concrete Naming Certificate for DiagonalLimitCompatibility$^{\uparrow}$ **8954**

647A Concrete Naming Certificate for ObserverPerspectiveClassifier$^{\uparrow}$ **9075**

648A Concrete Naming Certificate for CompactModulusCover$^{\uparrow}$ **9091**

649A Concrete Naming Certificate for StationaryWindowLattice$^{\uparrow}$ **9095**

650A Concrete Naming Certificate for ChoiceFreeLimitSchedule$^{\uparrow}$ **9098**

651A Concrete Naming Certificate for InscriptionAcceptanceBudget$^{\uparrow}$ **9106**

652A Concrete Naming Certificate for RealCompletionWitnessExtractor$^{\uparrow}$ **9110**

653A Concrete Naming Certificate for RegularCauchySwapBisimulation$^{\uparrow}$ **9118**

654A Concrete Naming Certificate for AxiomQueryLedger$^{\uparrow}$ **9122**

655A Concrete Naming Certificate for MultiHistSuperposition$^{\uparrow}$ **9126**

656A Concrete Naming Certificate for InterHistInvariant$^{\uparrow}$ **9130**

657A Concrete Naming Certificate for ContinuationBigStep$^{\uparrow}$ **9138**

658A Concrete Naming Certificate for RegularCauchyTailFusion$^{\uparrow}$ **9151**

659A Concrete Naming Certificate for NontrivialZeroClassifier$^{\uparrow}$ **9163**

660A Concrete Naming Certificate for RegularCauchySequence$^{\uparrow}$ **9167**

661A Concrete Naming Certificate for DefiniteDescriptionBoundary$^{\uparrow}$ **9185**

662A Concrete Naming Certificate for RealBudgetTailMeet$^{\uparrow}$ **9189**

663A Concrete Naming Certificate for RegularCauchyCofinalLimit$^{\uparrow}$ **9192**

664A Concrete Naming Certificate for ObservationTimeOrder$^{\uparrow}$ **9195**

665A Concrete Naming Certificate for LocalClockBudget$^{\uparrow}$ **9199**

666A Concrete Naming Certificate for SupplyKindRouter$^{\uparrow}$ **9204**

667A Concrete Naming Certificate for SelfReferenceTruthBranch$^{\uparrow}$ **9207**

668A Concrete Naming Certificate for ObservationCoupling$^{\uparrow}$ **9210**

669A Concrete Naming Certificate for PaperLeanDriftWitness$^{\uparrow}$ **9214**

670A Concrete Naming Certificate for CofinalRegularLimitBudget$^{\uparrow}$ **9234**

671A Concrete Naming Certificate for KernelSourceChannelLedger$^{\uparrow}$ **9238**

672A Concrete Naming Certificate for AxisNatReplacementRefusal$^{\uparrow}$ **9243**

673A Concrete Naming Certificate for LimitSelector$^{\uparrow}$ **9246**

674A Concrete Naming Certificate for GroundLoopBoundary$^{\uparrow}$ **9252**

675A Concrete Naming Certificate for PrefixFunctorIdentityNaturalityBridge$^{\uparrow}$ **9256**

676A Concrete Naming Certificate for RegularCauchyProductBudget$^{\uparrow}$ **9259**

677A Concrete Naming Certificate for ChoiceRecipeLedger$^{\uparrow}$ **9268**

678A Concrete Naming Certificate for RegularCauchySelectorBudget$^{\uparrow}$ **9274**

679A Concrete Naming Certificate for CompilerCompositionTrace$^{\uparrow}$ **9287**

680A Concrete Naming Certificate for DyadicCofinalTailSelector$^{\uparrow}$ **9291**

681A Concrete Naming Certificate for TailCofinalitySchedule$^{\uparrow}$ **9294**

682A Concrete Naming Certificate for InscriptionAuditTrace$^{\uparrow}$ **9309**

683A Concrete Naming Certificate for ContinuationTraceNormalForm$^{\uparrow}$ **9313**

684A Concrete Naming Certificate for ContextualClassReading$^{\uparrow}$ **9330**

685A Concrete Naming Certificate for EmptyBoundarySelector$^{\uparrow}$ **9334**

686A Concrete Naming Certificate for TypeCheckingMembershipTrace$^{\uparrow}$ **9339**

687A Concrete Naming Certificate for FableClockOrder$^{\uparrow}$ **9344**

688A Concrete Naming Certificate for ObserverPerspectiveComposition$^{\uparrow}$ **9347**

689A Concrete Naming Certificate for FiniteWindowObserver$^{\uparrow}$ **9350**

690A Concrete Naming Certificate for ProofTermErasureAuditRoute$^{\uparrow}$ **9356**

691A Concrete Naming Certificate for EmptyBoundaryStep$^{\uparrow}$ **9360**

692A Concrete Naming Certificate for AxisZeckendorfCarryBudget$^{\uparrow}$ **9363**

693A Concrete Naming Certificate for RegularCauchyWindowFusion$^{\uparrow}$ **9366**

694A Concrete Naming Certificate for BranchObstruction$^{\uparrow}$ **9376**

695A Concrete Naming Certificate for ChoiceFreeDiagonalSelector$^{\uparrow}$ **9380**

696A Concrete Naming Certificate for AuditGateComposition$^{\uparrow}$ **9387**

697A Concrete Naming Certificate for DiagonalTailSelector$^{\uparrow}$ **9396**

698A Concrete Naming Certificate for CofinalTailBudget$^{\uparrow}$ **9420**

699A Concrete Naming Certificate for RealTailAgreementSeal$^{\uparrow}$ **9425**

700A Concrete Naming Certificate for UnaryDirectionRouter$^{\uparrow}$ **9469**

701A Concrete Naming Certificate for StationaryEmbeddingClassifier$^{\uparrow}$ **9472**

702A Concrete Naming Certificate for RegularCauchyMerge$^{\uparrow}$ **9476**

703A Concrete Naming Certificate for CauchyInverseBudget$^{\uparrow}$ **9480**

704A Concrete Naming Certificate for ContourAffineConcat$^{\uparrow}$ **9490**

705A Concrete Naming Certificate for CriticalStripBarrier$^{\uparrow}$ **9493**

706A Concrete Naming Certificate for ObservableCauchySeal$^{\uparrow}$ **9497**

707A Concrete Naming Certificate for ProofTermErasureClassifier$^{\uparrow}$ **9500**

708RegularCauchyTelescopingBudget NameCert Construction **9503**

709A Concrete Naming Certificate for ClosedGeneratorAudit$^{\uparrow}$ **9511**

710A Concrete Naming Certificate for RecursionAuthorizationLedger$^{\uparrow}$ **9514**

711A Concrete Naming Certificate for GeneratorAuditClosure$^{\uparrow}$ **9520**

712A Concrete Naming Certificate for MetaCICSubjectReductionObligation$^{\uparrow}$ **9523**

713A Concrete Naming Certificate for AxisCarryConfluence$^{\uparrow}$ **9526**

714A Concrete Naming Certificate for TypeLevelSocketExposure$^{\uparrow}$ **9531**

715A Concrete Naming Certificate for ApophaticGateQuestion$^{\uparrow}$ **9535**

716A Concrete Naming Certificate for BetaSubstitutionPreservation$^{\uparrow}$ **9556**

717A Concrete Naming Certificate for MetaCICDischargeObligationLedger$^{\uparrow}$ **9567**

718A Concrete Naming Certificate for SubjectReductionDischarge$^{\uparrow}$ **9570**

719A Concrete Naming Certificate for SubjectReductionDischargeSocket$^{\uparrow}$ **9575**

720A Concrete Naming Certificate for SubjectReductionDischargeLedger$^{\uparrow}$ **9581**

721MetaCIC Subject-Reduction Boundary NameCert Construction **9587**

722A Concrete Naming Certificate for ClosedTermContextSeal$^{\uparrow}$ **9591**

723A Concrete Naming Certificate for MetaCICSubjectReductionTrace$^{\uparrow}$ **9594**

724A Concrete Naming Certificate for MetaCICDischargeObligation$^{\uparrow}$ **9598**

725A Concrete Naming Certificate for SubjectReductionObligation$^{\uparrow}$ **9601**

726A Concrete Naming Certificate for MetaCICReductionObstructionLedger$^{\uparrow}$ **9604**

727A Concrete Naming Certificate for BetaSubstitutionDischarge$^{\uparrow}$ **9607**

728A Concrete Naming Certificate for BetaCongruenceDomainAudit$^{\uparrow}$ **9611**

729A Concrete Naming Certificate for SubjectReductionObligationLedger$^{\uparrow}$ **9614**

730A Concrete Naming Certificate for MetaCICSubjectReductionFrontier$^{\uparrow}$ **9617**

731A Concrete Naming Certificate for MetaCICSubjectReductionObstruction↑ **9619**

732A Concrete Naming Certificate for NormalFalseExclusion↑ **9622**

733A Concrete Naming Certificate for ClosedNormalConsistencyBoundary↑ **9626**

734A Normal-Fragment Certificate for ClosedNormalConsistency↑ **9629**

735A Concrete Naming Certificate for MetaCICConfluenceAuditPacket↑ **9633**

736A Concrete Naming Certificate for ClosedNormalConsistency↑ **9653**

737A Concrete Naming Certificate for MetaCICConfluenceNormalFragment↑ **9655**

738A Concrete Naming Certificate for ClosedConsistencyReductionBoundary↑ **9658**

739A Concrete Naming Certificate for ClosedNormalConsistencyPacket↑ **9661**

740A Concrete Naming Certificate for NormalFormConsistencySeal↑ **9664**

741A Concrete Naming Certificate for ClosedNormalConfluenceSeal↑ **9677**

742A Concrete Naming Certificate for KernelAcceptanceBuildReplay↑ **9682**

743A Concrete Naming Certificate for MetaCICConfluenceAudit↑ **9685**

743.1Carrier 9685

744A Concrete Naming Certificate for ParallelConfluenceAudit↑ **9688**

745A Concrete Naming Certificate for MetaCICConfluenceAuditWitness↑ **9691**

746A Concrete Naming Certificate for MetaCICDecidableAuditPacket↑ **9694**

747A Concrete Naming Certificate for MetacicDecidableCheckerSeal↑ **9698**

748A Concrete Naming Certificate for BoundedBetaComparison↑ **9700**

749A Concrete Naming Certificate for MetaCICDecidableBoundary↑ **9703**

750A Concrete Naming Certificate for MetaCICNormalizationAudit↑ **9717**

751A Concrete Naming Certificate for MetaCICDecidabilityWitness↑ **9733**

752A Concrete Naming Certificate for BoundedNormalEqualityChecker↑ **9744**

753A Concrete Naming Certificate for MetaCICTypedExamplesSeal↑ **9752**

754A Concrete Naming Certificate for BetaStepBoundary↑ **9755**

755A Concrete Naming Certificate for MetaCICNormalizationBudget↑ **9758**

756A Concrete Naming Certificate for TypedExampleGallery↑ **9765**

757A Concrete Naming Certificate for DyadicTailRadiusLedger$^{\uparrow}$ **9768**

758A Concrete Naming Certificate for CandidateEvidenceSNRoute$^{\uparrow}$ **9779**

759A Concrete Naming Certificate for InterHistLocalityLedger$^{\uparrow}$ **9782**

760A Concrete Naming Certificate for TypedBetaCongruenceTrigger$^{\uparrow}$ **9785**

761A Concrete Naming Certificate for MetaCICBetaAudit$^{\uparrow}$ **9788**

762A Concrete Naming Certificate for MetaCICOpenFrontierLedger$^{\uparrow}$ **9791**

763A Concrete Naming Certificate for MetaCICOpenObligationLedger$^{\uparrow}$ **9794**

764A Concrete Naming Certificate for ClosedNormalConsistencyMain$^{\uparrow}$ **9797**

765A Concrete Naming Certificate for MetaCICCriticalPath$^{\uparrow}$ **9800**

766A Concrete Naming Certificate for RealAlgOrder$^{\uparrow}$ **9984**

767A Concrete Naming Certificate for SocketTaxonomyLedger$^{\uparrow}$ **9990**

768A Concrete Naming Certificate for Limsup$^{\uparrow}$ **9993**

769A Concrete Naming Certificate for UniformContinuousExtension$^{\uparrow}$ **10000**

770A Concrete Naming Certificate for JordanContent$^{\uparrow}$ **10003**

771A Concrete Naming Certificate for UniformConvergenceTest$^{\uparrow}$ **10006**

772A Concrete Naming Certificate for NadlerFixedPoint$^{\uparrow}$ **10009**

773A Concrete Naming Certificate for BrouwerianCounterexample$^{\uparrow}$ **10013**

774A Concrete Naming Certificate for FiniteBairstowRootWindow$^{\uparrow}$ **10016**

775A Concrete Naming Certificate for MetaCICTypedExamplesGallery$^{\uparrow}$ **10019**

776A Concrete Naming Certificate for MetaCICAuditSynthesisPacket$^{\uparrow}$ **10027**

777A Concrete Naming Certificate for MetaCICNormalizationAuditSeal$^{\uparrow}$ **10039**

778A Concrete Naming Certificate for RegularCauchyModulusNormalizer$^{\uparrow}$ **10042**

779A Concrete Naming Certificate for SubstitutionAuditMap$^{\uparrow}$ **10056**

780A Concrete Naming Certificate for MetaCICOpenProblemLedger$^{\uparrow}$ **10064**

781A Concrete Naming Certificate for DependentCodomainInversionBoundary$^{\uparrow}$ **10068**

782A Concrete Naming Certificate for FiniteObservationBudgetSelector$^{\uparrow}$ **10071**

783A Concrete Naming Certificate for SubjectReductionBundleTrace$^{\uparrow}$ **10107**

784A Concrete Naming Certificate for SubjectReductionRouteEquivalence$^{\uparrow}$ **10110**

785A Concrete Naming Certificate for SubjectReductionRouteClassifier$^{\uparrow}$ **10113**

786A Concrete Naming Certificate for ClosurePreservationAuditWitness$^{\uparrow}$ **10121**

787A Concrete Naming Certificate for SubjectReductionRoute$^{\uparrow}$ **10124**

788A Concrete Naming Certificate for MetaCICClosurePreservation$^{\uparrow}$ **10128**

789A Concrete Naming Certificate for SrDischargeRouteAudit$^{\uparrow}$ **10131**

790A Concrete Naming Certificate for SubjectReductionDischargeRouteCert$^{\uparrow}$ **10134**

791A Concrete Naming Certificate for MetaCICClosurePreservationAudit$^{\uparrow}$ **10136**

792A Concrete Naming Certificate for ClosurePreservationAuditTrace$^{\uparrow}$ **10139**

793A Concrete Naming Certificate for SubjectReductionRouteAudit$^{\uparrow}$ **10141**

794A Concrete Naming Certificate for ClosurePreservationTrace$^{\uparrow}$ **10144**

795A Concrete Naming Certificate for MetaCICClosureTrace$^{\uparrow}$ **10147**

796A Concrete Naming Certificate for SubjectReductionRouteTriangle$^{\uparrow}$ **10160**

797A Concrete Naming Certificate for ClosedBetaPath$^{\uparrow}$ **10163**

798A Concrete Naming Certificate for SubjectReductionRouteChoice$^{\uparrow}$ **10166**

799A Concrete Naming Certificate for ClosedNormalCanonicity$^{\uparrow}$ **10173**

800A Concrete Naming Certificate for ClosurePreservationAuditSeal$^{\uparrow}$ **10176**

801A Concrete Naming Certificate for ClosurePreservationClassifier$^{\uparrow}$ **10179**

802A Closedness-Route Audit Surface for MetaCICClosurePreservationAudit$^{\uparrow}$ **10182**

803A Concrete Naming Certificate for ClosurePreservationAuditPacket$^{\uparrow}$ **10185**

804A Concrete Naming Certificate for DependentCodomainClosurePreservation$^{\uparrow}$ **10188**

805A Concrete Naming Certificate for ClosurePreservationRouteLedger$^{\uparrow}$ **10192**

806A Concrete Naming Certificate for AuditMapInterface$^{\uparrow}$ **10194**

807A Concrete Naming Certificate for AuditMapFamily$^{\uparrow}$ **10199**

808A Concrete Naming Certificate for AuditMapTemplate$^{\uparrow}$ **10202**

809A Concrete Naming Certificate for TypeClassifierRouteJoin$^{\uparrow}$ **10205**

810A Concrete Naming Certificate for AuditMapFamilyLedger$^{\uparrow}$ **10208**

811MetaCIC Audit Map Family Name Certificate Construction **10211**

812A Concrete Naming Certificate for AuditMapFrontierIndex$^{\uparrow}$ **10215**

813A Concrete Naming Certificate for AuditMapMethodology$^{\uparrow}$ **10219**

814A Concrete Naming Certificate for AuditMapCoverageLedger$^{\uparrow}$ **10222**

815A Concrete Naming Certificate for AuditMapFamilyObligationLedger$^{\uparrow}$ **10225**

816A Concrete Naming Certificate for AuditMapFamilyPacket$^{\uparrow}$ **10228**

817A Family-Level Audit Surface for AuditMapFamily$^{\uparrow}$ **10231**

818A Concrete Naming Certificate for FiniteTraceEvaluator$^{\uparrow}$ **10237**

819A Concrete Naming Certificate for LegalChannel$^{\uparrow}$ **10289**

820A Concrete Naming Certificate for GroundCompilerRecognizerAuditPacket$^{\uparrow}$ **10292**

821A Concrete Naming Certificate for GroundCompilerRecognition$^{\uparrow}$ **10295**

822A Concrete Naming Certificate for GroundCompilerAuditVerdict$^{\uparrow}$ **10300**

823A Concrete Naming Certificate for AuditMapConcreteInventory$^{\uparrow}$ **10304**

824A Concrete Naming Certificate for SubstitutionAuditWindowRoute$^{\uparrow}$ **10307**

825A Concrete Naming Certificate for AuditMapTemplatePacket$^{\uparrow}$ **10310**

826A Concrete Naming Certificate for AuditMapObstructionSocket$^{\uparrow}$ **10313**

827A Concrete Naming Certificate for GroundCompilerAuditPacket$^{\uparrow}$ **10316**

828A Concrete Naming Certificate for GroundCompilerRecognizerBoundary$^{\uparrow}$ **10322**

829A Concrete Naming Certificate for GroundCompilerEventFlowAudit$^{\uparrow}$ **10325**

830A Concrete Naming Certificate for GroundCompilerEventFlowAuditRoute$^{\uparrow}$ **10328**

831A Concrete Naming Certificate for GroundCompilerAuditWitness$^{\uparrow}$ **10333**

832A Concrete Naming Certificate for GroundCompilerEvidenceBoundary$^{\uparrow}$ **10336**

833A Concrete Naming Certificate for AuditMapFrontierPacket$^{\uparrow}$ **10339**

834A Concrete Naming Certificate for AuditMapRouteCompiler$^{\uparrow}$ **10343**

835A Concrete Naming Certificate for FiniteSupport↑ **10346**

836A Concrete Naming Certificate for GroundCompilerAuditMap↑ **10349**

837A Concrete Naming Certificate for ClosedNormalEndpointLedger↑ **10356**

838A Concrete Naming Certificate for RegularCauchyTailSchedule↑ **10358**

839A Concrete Naming Certificate for AuditMapDependencyWeave↑ **10362**

840A Concrete Naming Certificate for SubstitutionContext↑ **10366**

841A Concrete Naming Certificate for TwinSubstrateAuditBridge↑ **10370**

842TwinSubstrateRoundtrip NameCert Construction **10373**

843A Concrete Naming Certificate for TwinSubstrateAuditPacket↑ **10375**

844A Concrete Naming Certificate for ClosedConsistencyGap↑ **10378**

845A Concrete Naming Certificate for TwinSubstrateSynthesis↑ **10381**

846A Concrete Naming Certificate for TwinSubstrateAuditUp **10384**

847A Concrete Naming Certificate for TwinSubstrateAuditCoupling↑ **10387**

848TwinSubstrateAuditSynthesis NameCert construction **10391**

849A Concrete Naming Certificate for AuditReexportNamespace↑ **10394**

850A Concrete Naming Certificate for TwinSubstrateBridge↑ **10397**

851A Concrete Naming Certificate for TwinSubstrateBoundaryRouter↑ **10401**

852A Concrete Naming Certificate for TwinSubstrateAuditHandshake↑ **10404**

853A Concrate Naming Certificate for CauchyDiagonalBudget↑ **10407**

854A Concrete Naming Certificate for CertificateTruthBranchBoundary↑ **10428**

855A Concrete Naming Certificate for BEDCTheoryIdentity↑ **10432**

856A Concrete Naming Certificate for MetaCICTheoryIdentity↑ **10435**

856.1Carrier and classifier 10435

857A Concrete Naming Certificate for BoundedRecursorReplay↑ **10439**

858A Concrete Naming Certificate for ClosedNormalRouteCertificate↑ **10442**

859A Concrete Naming Certificate for BedcSelfTheory↑ **10444**

860A Concrete Naming Certificate for MetaCICIdentity↑ **10447**

861A Concrete Naming Certificate for AxiomDependencyTuple$^{\uparrow}$ **10450**

862A Concrete Naming Certificate for AxiomDependencyMap$^{\uparrow}$ **10465**

863A Concrete Naming Certificate for AxiomDependencyMapLedger$^{\uparrow}$ **10470**

864A Concrete Naming Certificate for RestrictedSurfaceBudget$^{\uparrow}$ **10473**

865A Concrete Naming Certificate for InternalTheoryBoundary$^{\uparrow}$ **10476**

866A Concrete Naming Certificate for MetaCICIdentityCommitment$^{\uparrow}$ **10479**

867A Concrete Naming Certificate for MetaCICIdentityBoundary$^{\uparrow}$ **10482**

868A Concrete Naming Certificate for AxiomDependencyAuditMap$^{\uparrow}$ **10486**

869A Concrete Naming Certificate for BEDCSelfSubstrate$^{\uparrow}$ **10489**
869.1Carrier and certificate surface 10489

870A Concrete Naming Certificate for DiagonalLimitStability$^{\uparrow}$ **10492**

871A Concrete Naming Certificate for TheorySelfClassifier$^{\uparrow}$ **10500**

872A Concrete Naming Certificate for SelfTheoryReflectionLoop$^{\uparrow}$ **10503**

873A Concrete Naming Certificate for LiteralASTEqualityBoundary$^{\uparrow}$ **10506**

874A Concrete Naming Certificate for RegularCauchyFiniteObservationRoute$^{\uparrow}$ **10509**

875A Concrete Naming Certificate for CrossHistCausalRoute$^{\uparrow}$ **10514**

876A Concrete Naming Certificate for CrossHistCausalRate$^{\uparrow}$ **10517**

877A Concrete Naming Certificate for ObserverAccumulationClock$^{\uparrow}$ **10536**

878A Concrete Naming Certificate for FableMachineBoundary$^{\uparrow}$ **10539**

879A Concrete Naming Certificate for BetaReductionAuditTrail$^{\uparrow}$ **10543**

880A Concrete Naming Certificate for TypedExampleAuditBudget$^{\uparrow}$ **10546**

881A Concrete Naming Certificate for BetaCriticalPair$^{\uparrow}$ **10549**

882A Concrete Naming Certificate for ObserverDistinctionTrace$^{\uparrow}$ **10552**

883A Concrete Naming Certificate for PrefixObserver$^{\uparrow}$ **10556**

884A Concrete Naming Certificate for RealTailAgreementTransitivity$^{\uparrow}$ **10560**

885A Concrete Naming Certificate for CauchyModulusRefinement$^{\uparrow}$ **10563**
885.1L10 window lattice pointer 10647

886A Concrete Naming Certificate for CauchySealBudgetSynchronizer$^{\uparrow}$ **10654**

887A Concrete Naming Certificate for RealSealComparison$^{\uparrow}$ **10659**

888A Concrete Naming Certificate for ObserverTraceKernel$^{\uparrow}$ **10662**

889A Concrete Naming Certificate for ObserverLocalityTransport$^{\uparrow}$ **10665**

890A Concrete Naming Certificate for RuntimeReflectionBoundary$^{\uparrow}$ **10670**

891A Concrete Naming Certificate for RealSealCongruence$^{\uparrow}$ **10673**

892A Concrete Naming Certificate for MetaCICNormalizationFrontier$^{\uparrow}$ **10676**

893A Concrete Naming Certificate for RegularCauchyTailFiber$^{\uparrow}$ **10718**

894A Concrete Naming Certificate for ObserverFrameRefusal$^{\uparrow}$ **10732**

895A Concrete Naming Certificate for MultiHistCoupling$^{\uparrow}$ **10735**

896A Concrete Naming Certificate for DiagonalLimitBudget$^{\uparrow}$ **10738**

897A Concrete Naming Certificate for StationaryWindowNaturality$^{\uparrow}$ **10744**

898A Concrete Naming Certificate for CauchyTailWitnessEnvelope$^{\uparrow}$ **10747**

899A Concrete Naming Certificate for MetaClosureObstruction$^{\uparrow}$ **10753**

900A Concrete Naming Certificate for RealCompletionSealClassifierUp **10757**

901A Concrete Naming Certificate for AuditGateFailureWitness$^{\uparrow}$ **10765**

902A Concrete Naming Certificate for CauchyWitnessLedger$^{\uparrow}$ **10779**

902.1 Cauchy witness ledger terminal observer exactness 10809

902.2 Cauchy witness ledger classifier route coverage 10814

903A Concrete Naming Certificate for ObservationBudgetLimiter$^{\uparrow}$ **10815**

904A Concrete Naming Certificate for MetaCICAuditMatrix$^{\uparrow}$ **10830**

905A Concrete Naming Certificate for RealCauchyWitnessGrid$^{\uparrow}$ **10835**

906A Concrete Naming Certificate for RegularCauchyTailBudget$^{\uparrow}$ **10840**

907A Concrete Naming Certificate for PiApplicationAdequacyRoute$^{\uparrow}$ **10850**

908A Concrete Naming Certificate for ObservationScheduleEnvelope$^{\uparrow}$ **10853**

909A Concrete Naming Certificate for RegularCauchyDiagonalMeet$^{\uparrow}$ **10857**

910A Concrete Naming Certificate for SubstitutionGeneratorAudit$^{\uparrow}$ **10861**

911A Concrete Naming Certificate for MinkowskiRateGeometry$^{\uparrow}$ **10865**

912A Concrete Naming Certificate for AuthoredRecordBoundary$^{\uparrow}$ **10871**

913A Concrete Naming Certificate for BudgetedRealSealRoute$^{\uparrow}$ **10875**

914A Concrete Naming Certificate for CompactNetModulusSelector$^{\uparrow}$ **10881**

915A Concrete Naming Certificate for KernelInductiveAcceptance$^{\uparrow}$ **10902**

916A Concrete Naming Certificate for ClosedNormalEndpointTransport$^{\uparrow}$ **10913**

917A Concrete Naming Certificate for TailBudgetCoherence$^{\uparrow}$ **10916**

918A Concrete Naming Certificate for BoundedMonotoneConvergenceSeal$^{\uparrow}$ **10927**

919A Concrete Naming Certificate for FinitePrefixLimitBudget$^{\uparrow}$ **10946**

920A Concrete Naming Certificate for CofinalTailAgreement$^{\uparrow}$ **10961**

921A Concrete Naming Certificate for FiniteTailFiberSchedule$^{\uparrow}$ **10971**

922A Concrete Naming Certificate for MetaCICTypedExamplesApplicationRoute$^{\uparrow}$ **10975**

923A Concrete Naming Certificate for TypedExampleCoverageMatrix$^{\uparrow}$ **10978**

924A Concrete Naming Certificate for ChurchEliminatorRegressionMatrix$^{\uparrow}$ **10987**

925A Concrete Naming Certificate for LayeredRelationCert$^{\uparrow}$ **10991**

926A Concrete Naming Certificate for CompactNetModulusBudget$^{\uparrow}$ **10994**

927A Concrete Naming Certificate for AttentionLedger$^{\uparrow}$ **10997**

928A Concrete Naming Certificate for ObserverInterface$^{\uparrow}$ **11002**

929A Concrete Naming Certificate for CauchyRateBudget$^{\uparrow}$ **11007**

930A Concrete Naming Certificate for FiniteRealRouteCommitment$^{\uparrow}$ **11013**

931A Concrete Naming Certificate for UnaryContMonoid$^{\uparrow}$ **11016**

932A Concrete Naming Certificate for KernelObservationSieve$^{\uparrow}$ **11023**

933A Concrete Naming Certificate for MaxRateReadGate$^{\uparrow}$ **11026**

934A Concrete Naming Certificate for CellularAutomaton$^{\uparrow}$ **11031**

935A Concrete Naming Certificate for MetricSpace Compact-Continuity Source Lock **11036**

936A Concrete Naming Certificate for ZNormal$^{\uparrow}$ **11040**

937A Concrete Naming Certificate for CellularTrustSubstrate$^{\uparrow}$ **11114**

938A Concrete Naming Certificate for CellularOrbitSubstrate$^{\uparrow}$ **11116**

939A Concrete Naming Certificate for LargeModelInscriptionPoint$^{\uparrow}$ **11122**

940A Trust Manifest Surface for CellularTrustSubstrate$^{\uparrow}$ **11127**

941A Concrete Naming Certificate for CellularRowClassifier$^{\uparrow}$ **11134**

942A Concrete Naming Certificate for CellularSubstrate$^{\uparrow}$ **11137**

943A Concrete Naming Certificate for CellularCarrierEncoding$^{\uparrow}$ **11141**

944A Concrete Naming Certificate for BoundedUniversalTrace$^{\uparrow}$ **11149**

945A Concrete Naming Certificate for CellularSubstrateClassifier$^{\uparrow}$ **11157**

946A Concrete Naming Certificate for ClosureUniversalityQuadrant$^{\uparrow}$ **11161**

947A Concrete Naming Certificate for UniversalityClosureQuadrant$^{\uparrow}$ **11164**

948A Concrete Naming Certificate for BHistCellRowEmbedding$^{\uparrow}$ **11168**

949A Concrete Naming Certificate for CookFrontierCoordinate$^{\uparrow}$ **11171**

950A Concrete Naming Certificate for QuadrantSubstrate$^{\uparrow}$ **11176**

951A Concrete Naming Certificate for OracleAugmentedSubstrate$^{\uparrow}$ **11179**

952A Concrete Naming Certificate for CookCompileFrontierWitness$^{\uparrow}$ **11182**

953A Concrete Naming Certificate for UniversalClosedSubstrate$^{\uparrow}$ **11185**

954A Concrete Naming Certificate for CompileFrontierWitness$^{\uparrow}$ **11188**

955A Concrete Naming Certificate for PresburgerSubUniversal$^{\uparrow}$ **11191**

956A Concrete Naming Certificate for OracleSubstrateBoundary$^{\uparrow}$ **11194**

957A Concrete Naming Certificate for ClosedSubstrateQuadrant$^{\uparrow}$ **11197**

958A Concrete Naming Certificate for HaltedTmRoundTrip$^{\uparrow}$ **11200**

959Cook Frontier Witness Name Certificate Construction **11203**

960A Concrete Naming Certificate for FuelIndexedSubstrateBridge$^{\uparrow}$ **11207**

961A Concrete Naming Certificate for DiagonalLimitObserver$^{\uparrow}$ **11210**

962A Concrete Naming Certificate for TotalHostFuelHandoff$^{\uparrow}$ **11214**

963A Concrete Naming Certificate for FuelBoundedPartialTrace$^{\uparrow}$ **11218**

964A Concrete Naming Certificate for StepIndexedTotalHost$^{\uparrow}$ **11221**

965A Concrete Naming Certificate for PromptResponseTrace$^{\uparrow}$ **11226**

966A Concrete Naming Certificate for ObserverResponseAudit$^{\uparrow}$ **11230**

967A Concrete Naming Certificate for LargeModelActivationOrbit$^{\uparrow}$ **11233**

968A Concrete Naming Certificate for CorpusSupplyDistillation$^{\uparrow}$ **11237**

969A Concrete Naming Certificate for LargeModelAuditChannel$^{\uparrow}$ **11244**

970A Concrete Naming Certificate for ReductionFuelBoundary$^{\uparrow}$ **11249**

971A Concrete Naming Certificate for ContextWindowCommitment$^{\uparrow}$ **11255**

972A Concrete Naming Certificate for ModelTrace$^{\uparrow}$ **11258**

973ObserverBoundarySelectorUp NameCert construction **11262**

974A Concrete Naming Certificate for RecursorBranchAudit$^{\uparrow}$ **11265**

975A Concrete Naming Certificate for SolipsismGapRefusal$^{\uparrow}$ **11268**

976A Concrete Naming Certificate for VerifiedOutputHarness$^{\uparrow}$ **11271**

977A Concrete Naming Certificate for RegularCauchyModulusWitnessLedger$^{\uparrow}$ **11275**

978A Concrete Naming Certificate for HaltingTraceClassifier$^{\uparrow}$ **11303**

979A Concrete Naming Certificate for RealWindowSynchronizer$^{\uparrow}$ **11306**

980A Concrete Naming Certificate for RegularCauchyClassifierNormalForm$^{\uparrow}$ **11312**

981A Concrete Naming Certificate for DyadicWindowSeal$^{\uparrow}$ **11315**

982A Concrete Naming Certificate for InscriptionCommitmentSeal$^{\uparrow}$ **11319**

983A Concrete Naming Certificate for CauchySharedTailModulus$^{\uparrow}$ **11322**

984A Concrete Naming Certificate for CellularLocalUpdateRule$^{\uparrow}$ **11332**

985A Concrete Naming Certificate for HaltingFiniteBoundary$^{\uparrow}$ **11335**

986ObserverHsameClassifierUp NameCert construction **11338**

987A Concrete Naming Certificate for CellularVerbRealignment$^{\uparrow}$ **11343**

988A Concrete Naming Certificate for LargeModelCorpusSupply$^{\uparrow}$ **11346**

989A Concrete Naming Certificate for LargeModelContextAudit$^{\uparrow}$ **11349**

990A Concrete Naming Certificate for CauchyTailConsensus$^{\uparrow}$ **11352**

991A Concrete Naming Certificate for RegularCauchyApartnessBudget$^{\uparrow}$ **11355**

992A Concrete Naming Certificate for AxiomRequirementLedger$^{\uparrow}$ **11369**

993A Concrete Naming Certificate for FiniteReflectionTuple$^{\uparrow}$ **11372**

994A Concrete Naming Certificate for RealSelectorComparison$^{\uparrow}$ **11379**

995A Concrete Naming Certificate for CellularPatternCatalog$^{\uparrow}$ **11382**

996A Concrete Naming Certificate for RatClassifierTransportSeal$^{\uparrow}$ **11386**

997A Concrete Naming Certificate for ReflectionRuntimeCheckpoint$^{\uparrow}$ **11389**

998A Concrete Naming Certificate for QuotientStreamRefusal$^{\uparrow}$ **11406**

999A Concrete Naming Certificate for TrivialZeroClassifier$^{\uparrow}$ **11414**

1000A Concrete Naming Certificate for ApophaticKernelBoundary$^{\uparrow}$ **11417**

1001A Concrete Naming Certificate for RealSealAdmissionCut$^{\uparrow}$ **11420**

1002A Concrete Naming Certificate for CertificateAuditGate$^{\uparrow}$ **11426**

1003A Concrete Naming Certificate for NoGlobalSyncBoundary$^{\uparrow}$ **11431**

1004A Concrete Naming Certificate for DyadicTailRadiusComparison$^{\uparrow}$ **11435**

1005A Concrete Naming Certificate for InscriptionRouteAudit$^{\uparrow}$ **11438**

1006A Concrete Naming Certificate for GapClosureBoundary$^{\uparrow}$ **11442**

1007A Concrete Naming Certificate for FiniteObservationInterface$^{\uparrow}$ **11509**

1008A Concrete Naming Certificate for MetaCICRedexFrontier$^{\uparrow}$ **11514**

1009A Concrete Naming Certificate for CofinalRealRequestReuse$^{\uparrow}$ **11521**

1010A Concrete Naming Certificate for UnaryDirectionNormalizer$^{\uparrow}$ **11524**

1011A Concrete Naming Certificate for ObserverLocalityPullback$^{\uparrow}$ **11527**

1012A Concrete Naming Certificate for TermStratumChoiceClosure$^{\uparrow}$ **11530**

1013A Concrete Naming Certificate for InterHistSymmetryBudget$^{\uparrow}$ **11533**

1014A Concrete Naming Certificate for CauchyLimitWitnessLedger$^{\uparrow}$ **11536**

1015 A Concrete Naming Certificate for OracleResponseLedger$^{\uparrow}$ **11539**

1016 A Concrete Naming Certificate for FinitePrefixAutomaton$^{\uparrow}$ **11542**

1017 A Concrete Naming Certificate for SharedTailNormalizer$^{\uparrow}$ **11549**

1018 A Concrete Naming Certificate for AxisBoundaryLimitRefusal$^{\uparrow}$ **11552**

1019 A Concrete Naming Certificate for ContourReversalLedger$^{\uparrow}$ **11555**

1020 A Concrete Naming Certificate for CauchyWindowTransducer$^{\uparrow}$ **11558**

1021 A Concrete Naming Certificate for LargeModelAttentionGate$^{\uparrow}$ **11562**

1022 A Concrete Naming Certificate for RegularCauchyWindowIntersection$^{\uparrow}$ **11565**

1023 A Concrete Naming Certificate for InterHistRateBoundary$^{\uparrow}$ **11569**

1024 A Concrete Naming Certificate for CauchyTailModulusFusion$^{\uparrow}$ **11573**

1025 A Concrete Naming Certificate for ContinuationBisimulation$^{\uparrow}$ **11578**

1026 A Concrete Naming Certificate for KleisliTrace$^{\uparrow}$ **11581**

1027 A Concrete Naming Certificate for LargeModelInscriptionAudit$^{\uparrow}$ **11584**

1028 A Concrete Naming Certificate for RuntimeTruthSeparation$^{\uparrow}$ **11588**

1029 A Concrete Naming Certificate for KernelNormalizationAuditJoin$^{\uparrow}$ **11590**

1030 A Concrete Naming Certificate for FinitePrefixMachine$^{\uparrow}$ **11593**

1031 ObserverLicensedComparisonUp NameCert construction **11597**

1032 ObserverBridgeLedgerUp NameCert construction **11600**

1033 A Concrete Naming Certificate for RealCompletionRouteAudit$^{\uparrow}$ **11604**

1034 CauchySealFunctorUp NameCert construction **11608**

1035 A Concrete Naming Certificate for LocatedCauchyModulus$^{\uparrow}$ **11611**

1036 A Concrete Naming Certificate for LocalStencilObservation$^{\uparrow}$ **11614**

1037 A Concrete Naming Certificate for RelationalFrameAudit$^{\uparrow}$ **11618**

1038 A Concrete Naming Certificate for TarskiTruthRefusal$^{\uparrow}$ **11624**

1039 A Concrete Naming Certificate for RealLimitUniquenessSeal$^{\uparrow}$ **11627**

1040 A Concrete Naming Certificate for RealLimitUniquenessWitness$^{\uparrow}$ **11640**

1041 Concrete Naming Certificate for CofinalDiagonalSeal$^{\uparrow}$ **11643**

1042 Concrete Naming Certificate for ObserverFilterBudgetSelector$^{\uparrow}$ **11650**

1043 Concrete Naming Certificate for ObserverTimeStep$^{\uparrow}$ **11658**

1044 Concrete Naming Certificate for SocketUseLedger$^{\uparrow}$ **11661**

1045 Concrete Naming Certificate for RuleOneTenGliderCatalog$^{\uparrow}$ **11665**

1046 Concrete Naming Certificate for TypingJudgmentClassifier$^{\uparrow}$ **11668**

1047 ObserverDirectionUp NameCert construction **11671**

1048 Concrete Naming Certificate for GroundCompilerRecognizerFlow$^{\uparrow}$ **11674**

1049 Concrete Naming Certificate for AxisCarryRouteSeparation$^{\uparrow}$ **11678**

1050 Concrete Naming Certificate for HostTotalTrace$^{\uparrow}$ **11680**

1051 Concrete Naming Certificate for TailCauchyRouter$^{\uparrow}$ **11683**

1052 Concrete Naming Certificate for InscriptionTraceClassifier$^{\uparrow}$ **11687**

1053 Concrete Naming Certificate for RegSeqObservationBudget$^{\uparrow}$ **11690**

1054 Concrete Naming Certificate for ObserverTraceSeal$^{\uparrow}$ **11696**

1055 Concrete Naming Certificate for CompilerTraceFaithfulness$^{\uparrow}$ **11699**

1056 Concrete Naming Certificate for MetaCICDischargeObstructionLattice$^{\uparrow}$ **11703**

1057 Concrete Naming Certificate for AxiomRefusalLedger$^{\uparrow}$ **11706**

1058 Concrete Naming Certificate for SocketKindClassifier$^{\uparrow}$ **11710**

1059 Concrete Naming Certificate for MachineInterfaceSocket$^{\uparrow}$ **11713**

1060 Concrete Naming Certificate for SchwarzReflectionRoute$^{\uparrow}$ **11717**

1061 Concrete Naming Certificate for ApophaticFarEndSocket$^{\uparrow}$ **11720**

1062 Concrete Naming Certificate for DigestProvenancePacket$^{\uparrow}$ **11723**

1063 Concrete Naming Certificate for MetaCICProofObjectAuditRoute$^{\uparrow}$ **11727**

1064 Concrete Naming Certificate for ApophaticFiberFarEnd$^{\uparrow}$ **11730**

1065 Concrete Naming Certificate for ExternalSupplyBoundary$^{\uparrow}$ **11734**

1066 Concrete Naming Certificate for NonAxiomBoundaryForm$^{\uparrow}$ **11739**

1067 A Concrete Naming Certificate for DigestFiberLedger$^{\uparrow}$ **11742**

1068 A Concrete Naming Certificate for ApophaticFixedPoint$^{\uparrow}$ **11746**

1069 A Concrete Naming Certificate for ApophaticFixedPointFiber$^{\uparrow}$ **11749**

1070 A Concrete Naming Certificate for BoundaryGate$^{\uparrow}$ **11753**

1071 A Concrete Naming Certificate for LocalSelfCenterInscriptionSeal$^{\uparrow}$ **11757**

1072 A Concrete Naming Certificate for ExternalSupplySocket$^{\uparrow}$ **11761**

1073 A Concrete Naming Certificate for NonAxiomAdmission$^{\uparrow}$ **11764**

1074 A Concrete Naming Certificate for BoundaryAdmissionDecision$^{\uparrow}$ **11768**

1075 A Concrete Naming Certificate for GeneratorLocalInductionBoundary$^{\uparrow}$ **11771**

1076 A Concrete Naming Certificate for ExternalSupplyFailureWitness$^{\uparrow}$ **11774**

1077 A Concrete Naming Certificate for ExternalSupplyWitness$^{\uparrow}$ **11777**

1078 A Concrete Naming Certificate for InterInscriptionCoherence$^{\uparrow}$ **11781**

1079 A Concrete Naming Certificate for HashApophaticSeal$^{\uparrow}$ **11784**

1080 A Concrete Naming Certificate for InscriptionHashTrace$^{\uparrow}$ **11787**

1081 A Concrete Naming Certificate for RealCompletionSelectorSeal$^{\uparrow}$ **11790**

1082 A Concrete Naming Certificate for InscribedRouteSelector$^{\uparrow}$ **11805**

1083 A Concrete Naming Certificate for DigestLoopRefusal$^{\uparrow}$ **11808**

1084 A Concrete Naming Certificate for HashLikeFiberBoundary$^{\uparrow}$ **11811**

1085 A Concrete Naming Certificate for DigestInscription$^{\uparrow}$ **11815**

1086 A Concrete Naming Certificate for InscriptionEventReplay$^{\uparrow}$ **11818**

1087 A Concrete Naming Certificate for HashDigestFiberBoundary$^{\uparrow}$ **11821**

1088 ForceInterHistRelationUp NameCert construction **11825**

1089 A Concrete Naming Certificate for LayeredRelationSite$^{\uparrow}$ **11828**

1090 A Relation-Before-Identity Surface for LayeredRelationCert$^{\uparrow}$ **11831**

1091 A Concrete Naming Certificate for LayeredRelationDisplay$^{\uparrow}$ **11835**

1092 A Concrete Naming Certificate for AnalogyCertificateGate$^{\uparrow}$ **11838**

1093 Same-Schema Relation Certificate for LayeredRelationCert$^{\uparrow}$ **11842**

1094 Concrete Naming Certificate for GroundCompilerFlowLevelRoundTrip$^{\uparrow}$ **11846**

1095 Concrete Naming Certificate for CompactCoverShrinkageLedger$^{\uparrow}$ **11849**

1096 Concrete Naming Certificate for KernelInductiveEliminator$^{\uparrow}$ **11852**

1097 Concrete Naming Certificate for CompilerClassifierRoute$^{\uparrow}$ **11855**

1098 Concrete Naming Certificate for LayeredRelationFailureBoundary$^{\uparrow}$ **11858**

1099 Concrete Naming Certificate for LayeredRelationGate$^{\uparrow}$ **11861**

1100 Concrete Naming Certificate for MetaCICOpenProblemWitness$^{\uparrow}$ **11866**

1101 Concrete Naming Certificate for GammaZetaFactorSeal$^{\uparrow}$ **11869**

1102 Concrete Naming Certificate for SubstitutionBoundaryAudit$^{\uparrow}$ **11872**

1103 Concrete Naming Certificate for WitnessedDescentLedger$^{\uparrow}$ **11875**

1104 Stationary Window Seal Exhaustion Name Certificate Construction **11879**

1105 Concrete Naming Certificate for InterHistSynchronizationRefusal$^{\uparrow}$ **11882**

1106 Concrete Naming Certificate for NoGlobalSynchronizationLedger$^{\uparrow}$ **11885**

1107 Concrete Naming Certificate for ExternalSupplyAuditRoute$^{\uparrow}$ **11889**

1108 Concrete Naming Certificate for MetaCICNormalizationCertificateRouter$^{\uparrow}$ **11893**

1109 Concrete Naming Certificate for FocusedMultiHistLedger$^{\uparrow}$ **11909**

1110 Concrete Naming Certificate for CauchyWitnessGluing$^{\uparrow}$ **11912**

1111 Concrete Naming Certificate for UnaryContinuationEndofunctor$^{\uparrow}$ **11918**

1112 Concrete Naming Certificate for CrossHistCausalCone$^{\uparrow}$ **11921**

1113 Concrete Naming Certificate for FiniteCauchyGluingBudget$^{\uparrow}$ **11925**

1114 Concrete Naming Certificate for TwinSubstrateBoundaryClassifier$^{\uparrow}$ **11929**

1115 Concrete Naming Certificate for RegularCauchyIntervalSelector$^{\uparrow}$ **11932**

1116 Concrete Naming Certificate for CauchyLimitFunctor$^{\uparrow}$ **11936**

1117 Concrete Naming Certificate for CauchyWitnessPullback$^{\uparrow}$ **11949**

1118 Concrete Naming Certificate for AxisCarryDiamondRoute$^{\uparrow}$ **11954**

1119 A Concrete Naming Certificate for FiniteTailFilter$^{\uparrow}$ **11958**

1120 A Concrete Naming Certificate for RealCompletionExactBoundaryUp **11979**

1121 A Concrete Naming Certificate for ObserverBudgetSupport$^{\uparrow}$ **11997**

1122 A Concrete Naming Certificate for ObserverSupportLattice$^{\uparrow}$ **12002**

1123 A Concrete Naming Certificate for CellularBoundaryCondition$^{\uparrow}$ **12005**

1124 A Concrete Naming Certificate for RealityConstrainedModelAudit$^{\uparrow}$ **12008**

1125 A Concrete Naming Certificate for OnticState$^{\uparrow}$ **12013**

1126 A Concrete Naming Certificate for ObservationReflectionPacket$^{\uparrow}$ **12021**

1127 A Concrete Naming Certificate for RealityConstrainedApproximationTower$^{\uparrow}$ **12025**

1128 A Concrete Naming Certificate for RealityConstrainedTowerPacket$^{\uparrow}$ **12029**

1129 A Concrete Naming Certificate for PhysicalModelAudit$^{\uparrow}$ **12035**

1130 A Concrete Naming Certificate for CertifiedPhysicalTruth$^{\uparrow}$ **12040**

1131 A Concrete Naming Certificate for OpenPhysicalFit$^{\uparrow}$ **12044**

1132 A Concrete Naming Certificate for RealityConstrainedOpenFit$^{\uparrow}$ **12048**

1133 A Concrete Naming Certificate for RealityConstrainedObjectivity$^{\uparrow}$ **12051**

1134 A Concrete Naming Certificate for ObjectivityRefutationBoundary$^{\uparrow}$ **12055**

1135 A Concrete Naming Certificate for RealityConstrainedPhysicalInduction$^{\uparrow}$ **12059**

1136 A Concrete Naming Certificate for RealityConstrainedExplanation$^{\uparrow}$ **12064**

1137 A Concrete Naming Certificate for OpenFitPacket$^{\uparrow}$ **12067**

1138 A Concrete Naming Certificate for RelationalObjectivity$^{\uparrow}$ **12071**

1139 A Concrete Naming Certificate for PhysicalInductionStabilitySeal$^{\uparrow}$ **12075**

1140 A Concrete Naming Certificate for RealityConstrainedSynthesisSeal$^{\uparrow}$ **12079**

1141 A Concrete Naming Certificate for PhysicalTruthCertificate$^{\uparrow}$ **12082**

1142 A Concrete Naming Certificate for RealityConstrainedTowerCompression$^{\uparrow}$ **12086**

1143 A Concrete Naming Certificate for TowerEquivalence$^{\uparrow}$ **12092**

1144 A Concrete Naming Certificate for RealityConstrainedTruthCert$^{\uparrow}$ **12097**

1145 Concrete Naming Certificate for RealityConstrainedMethodologyLedger$^{\uparrow}$ **12117**

1146 Concrete Naming Certificate for TheoryTransitionCertificate$^{\uparrow}$ **12146**

1147 Concrete Naming Certificate for OnticAdequacyAudit$^{\uparrow}$ **12149**

1148 Concrete Naming Certificate for ApproximationTowerResidue$^{\uparrow}$ **12152**

1149 Concrete Naming Certificate for ReflectiveStability$^{\uparrow}$ **12156**

1150 Concrete Naming Certificate for AnchorStabilityCertificate$^{\uparrow}$ **12160**

1151 Concrete Naming Certificate for ReflectiveInquiry$^{\uparrow}$ **12163**

1152 Concrete Naming Certificate for OnticResidueLedger$^{\uparrow}$ **12169**

1153 Concrete Naming Certificate for OnticTower$^{\uparrow}$ **12174**

1154 Concrete Naming Certificate for OnticResidueLattice$^{\uparrow}$ **12178**

1155 Concrete Naming Certificate for LawCertificate$^{\uparrow}$ **12181**

1156 Concrete Naming Certificate for EmpiricalRegularityPersistence$^{\uparrow}$ **12187**

1157 Concrete Naming Certificate for HostPrimitiveLeakage$^{\uparrow}$ **12192**

1158 Concrete Naming Certificate for RegularCauchyRegularityWitness$^{\uparrow}$ **12197**

1159 Concrete Naming Certificate for MetricTriangleUniformEstimate$^{\uparrow}$ **12204**

1160 Concrete Naming Certificate for CertifiedOnticTower$^{\uparrow}$ **12221**

1161 Concrete Naming Certificate for ReflectionGapBoundary$^{\uparrow}$ **12224**

1162 Concrete Naming Certificate for CompressionLedgerFunctor$^{\uparrow}$ **12228**

1163 Concrete Naming Certificate for RealityConstrainedFailureSurface$^{\uparrow}$ **12232**

1164 Concrete Naming Certificate for RealWindowBudget$^{\uparrow}$ **12236**

1165 Concrete Naming Certificate for ModelSelectionBoundary$^{\uparrow}$ **12249**

1166 Concrete Naming Certificate for AnchorChangeInvariant$^{\uparrow}$ **12253**

1167 Concrete Naming Certificate for FiniteObservationResidueBoundary$^{\uparrow}$ **12256**

1168 Concrete Naming Certificate for OpenFitResidueGate$^{\uparrow}$ **12259**

1169 Concrete Naming Certificate for FiniteObservationRefutation$^{\uparrow}$ **12262**

1170 Concrete Naming Certificate for ScientificIdealizationResidueSeal$^{\uparrow}$ **12267**

1171 Concrete Naming Certificate for CauchyTailModulusSeal$^{\uparrow}$ **12270**

1172 Concrete Naming Certificate for InscriptionEventBufferSeal$^{\uparrow}$ **12275**

1173 Concrete Naming Certificate for CauchyChoiceBoundary$^{\uparrow}$ **12278**

1174 Concrete Naming Certificate for CofinalWindowRealSeal$^{\uparrow}$ **12283**

1175 Concrete Naming Certificate for RegularCauchyWitnessSelector$^{\uparrow}$ **12288**

1176 Concrete Naming Certificate for CompressionDescentAudit$^{\uparrow}$ **12292**

1177 Concrete Naming Certificate for ClosedConsistencyAssembly$^{\uparrow}$ **12295**

1178 Concrete Naming Certificate for RealityConstrainedFinalSynthesisLedger$^{\uparrow}$ **12299**

1179 Concrete Naming Certificate for SliceCategory$^{\uparrow}$ **12303**

1180 Concrete Naming Certificate for RegularCauchyCompletionFrontier$^{\uparrow}$ **12306**

1181 Concrete Naming Certificate for DomainTruthCertificate$^{\uparrow}$ **12314**

1182 Concrete Naming Certificate for PredictiveDescent$^{\uparrow}$ **12317**

1183 Concrete Naming Certificate for ContinuationCountermodel$^{\uparrow}$ **12320**

1184 Concrete Naming Certificate for RealityConstrainedModelErrorBudget$^{\uparrow}$ **12323**

1185 Concrete Naming Certificate for RealityConstrainedScientificMethodology$^{\uparrow}$ **12328**

1186 Concrete Naming Certificate for EmptyFableMachine$^{\uparrow}$ **12331**

1187 Concrete Naming Certificate for RecursorClosureClassifierAudit$^{\uparrow}$ **12334**

1188 Concrete Naming Certificate for LargeModelOutputVerifier$^{\uparrow}$ **12337**

1189 Concrete Naming Certificate for SynchronizationCertificate$^{\uparrow}$ **12341**

1190 Concrete Naming Certificate for MetaCICNormalizationConfluenceBridge$^{\uparrow}$ **12344**

1191 Concrete Naming Certificate for PhysicalNameCertAudit$^{\uparrow}$ **12350**

1192 Concrete Naming Certificate for FiniteObservationReflectionSeal$^{\uparrow}$ **12353**

1193 Concrete Naming Certificate for PhysicalOperationDescent$^{\uparrow}$ **12356**

1194 Concrete Naming Certificate for FiniteHistLocalityPacket$^{\uparrow}$ **12359**

1195 Concrete Naming Certificate for TowerEndpointReflection$^{\uparrow}$ **12364**

1196 Concrete Naming Certificate for AxisCarryRefusalLedger$^{\uparrow}$ **12367**

1197 A Concrete Naming Certificate for TerminationRefusalBoundary$^{\uparrow}$ **12371**

1198 A Concrete Naming Certificate for CofinalModulusSeal$^{\uparrow}$ **12375**

1199 A Concrete Naming Certificate for FiniteWindowRealSealAudit$^{\uparrow}$ **12396**

1200 A Concrete Naming Certificate for RealModulusPurityBoundary$^{\uparrow}$ **12402**

1201 A Concrete Naming Certificate for InterHistTransportSeal$^{\uparrow}$ **12409**

1202 A Concrete Naming Certificate for FiniteWindowRealSeparation$^{\uparrow}$ **12412**

1203 A Concrete Naming Certificate for BorrowedRecursorBoundary$^{\uparrow}$ **12420**

1204 A Concrete Naming Certificate for TailCompatibleCauchySection$^{\uparrow}$ **12423**

1205 A Concrete Naming Certificate for DiagonalLimitRepresentative$^{\uparrow}$ **12428**

1206 A Concrete Naming Certificate for ConsciousUniverseSpace$^{\uparrow}$ **12500**

1207 A Concrete Naming Certificate for SelectedTailSeed$^{\uparrow}$ **12504**

1208 A Concrete Naming Certificate for UnitCalibrationLedger$^{\uparrow}$ **12514**

1209 A Concrete Naming Certificate for ClaimRegistryLayer$^{\uparrow}$ **12518**

1210 A Concrete Naming Certificate for TailModulusAlignment$^{\uparrow}$ **12522**

1211 A Concrete Naming Certificate for DyadicToleranceTriangleLedger$^{\uparrow}$ **12525**

1212 A Concrete Naming Certificate for DyadicModulusLadder$^{\uparrow}$ **12530**

1213 A Concrete Naming Certificate for RealityConstrainedSignatureResidue$^{\uparrow}$ **12533**

1214 A Concrete Naming Certificate for ScientificObject$^{\uparrow}$ **12546**

1215 A Concrete Naming Certificate for FiniteRealSection$^{\uparrow}$ **12551**

1216 A Concrete Naming Certificate for RegistryExportSurface$^{\uparrow}$ **12557**

1217 A Concrete Naming Certificate for StandardBridgeAuditPacket$^{\uparrow}$ **12566**

1218 A Concrete Naming Certificate for ObjectKnowledgeCertificate$^{\uparrow}$ **12572**

1219 A Concrete Naming Certificate for ProofTraceAuditWitness$^{\uparrow}$ **12576**

1220 A Concrete Naming Certificate for RegistryLayer$^{\uparrow}$ **12579**

1221 A Concrete Naming Certificate for MachineInterfaceBoundary$^{\uparrow}$ **12583**

1222 A Concrete Naming Certificate for WritingItemAuditPacket$^{\uparrow}$ **12587**

1223 Concrete Naming Certificate for RegistryConsistencyAudit↑ **12591**

1224 Concrete Naming Certificate for RegistryExportConsistencyGate↑ **12594**

1225 Concrete Naming Certificate for MachineInterfaceFormalizationTarget↑ **12598**

1226 Concrete Naming Certificate for FinitePrefixLimitStability↑ **12602**

1227 Concrete Naming Certificate for AuditWritingDiscipline↑ **12610**

1228 Concrete Naming Certificate for PointwiseModulusLedger↑ **12614**

1229 Concrete Naming Certificate for GapFailureBridgeAudit↑ **12618**

1230 Concrete Naming Certificate for RegularCauchyTailCertificate↑ **12622**

1231 Concrete Naming Certificate for TailCofinalityBudget↑ **12626**

1232 Concrete Naming Certificate for MachineReadableAuditInterface↑ **12631**

1233 Concrete Naming Certificate for FailureCertificate↑ **12636**

1234 Concrete Naming Certificate for AuditTranscriptEnvelope↑ **12640**

1235 Concrete Naming Certificate for WritingItemAudit↑ **12644**

1236 Concrete Naming Certificate for HardProblemClosure↑ **12650**

1237 Concrete Naming Certificate for TypedFailureCertificate↑ **12653**

1238 Concrete Naming Certificate for ProofWitnessChain↑ **12658**

1239 Concrete Naming Certificate for TruthResidueLedger↑ **12661**

1240 Concrete Naming Certificate for TypedGapSocketTaxonomy↑ **12664**

1241 Concrete Naming Certificate for HistoryPrefix↑ **12667**

1242 Concrete Naming Certificate for RealDiagonalRegularitySeal↑ **12670**

1243 Concrete Naming Certificate for TypedSupplySocket↑ **12674**

1244 Concrete Naming Certificate for PrimitiveDistinctionLedger↑ **12679**

1245 Concrete Naming Certificate for DiagonalRegularitySeal↑ **12682**

1246 Concrete Naming Certificate for TheoremGapRegistry↑ **12685**

1247 Concrete Naming Certificate for FocusedLedger↑ **12689**

1248 Concrete Naming Certificate for RefusalRegistry↑ **12694**

1249 A Concrete Naming Certificate for ActiveReadingGate↑ **12698**

1250 A Concrete Naming Certificate for ClosedBoundedIntervalNet↑ **12701**

1251 A Concrete Naming Certificate for CoreRegistryIndex↑ **12710**

1252 A Concrete Naming Certificate for ConceptRegistrySurface↑ **12713**

1253 A Concrete Naming Certificate for ClosedBoundedIntervalUniformModulus↑ **12716**

1254 A Concrete Naming Certificate for BolzanoWeierstrass↑ **12720**

1255 CofinalTailEquivalence NameCert Construction **12748**

1256 A Concrete Naming Certificate for PhilosophySynthesisAudit↑ **12753**

1257 A Concrete Naming Certificate for CompactUniformEpsilonTriangleHandoff↑ **12756**

1258 A Concrete Naming Certificate for AuditSystem↑ **12759**

1259 A Concrete Naming Certificate for AxiomPurityGate↑ **12763**

1260 A Concrete Naming Certificate for RuntimeBoundaryAudit↑ **12767**

1261 A Concrete Naming Certificate for LogicContradictionMetaLoop↑ **12770**

1262 A Concrete Naming Certificate for ForwardBindingGapLedger↑ **12774**

1263 A Concrete Naming Certificate for CofinalStreamTailSelector↑ **12777**

1264 A Concrete Naming Certificate for CofinalModulusNormalizationSeal↑ **12786**

1265 A Concrete Naming Certificate for ClosureStatusReference↑ **12789**

1266 A Concrete Naming Certificate for FiniteObservationLock↑ **12792**

1267 A Concrete Naming Certificate for TraditionRegistry↑ **12795**

1268 A Concrete Naming Certificate for TraditionComparisonBoundary↑ **12798**

1269 A Concrete Naming Certificate for MetaClosureWitnessLedger↑ **12803**

1270 A Concrete Naming Certificate for TraditionScienceRegistryBridge↑ **12806**

1271 A Concrete Naming Certificate for SubjectFreeObservation↑ **12809**

1272 A Concrete Naming Certificate for SubjectIndexRoute↑ **12812**

1273 A Concrete Naming Certificate for ScientificModelBridge↑ **12815**

1274 A Concrete Naming Certificate for UpgradeRegistry↑ **12818**

1275 A Concrete Naming Certificate for RelationalPhysics$^{\uparrow}$ **12822**

1276 A Concrete Naming Certificate for FiniteLimitSealEnvelope$^{\uparrow}$ **12826**

1277 A Concrete Naming Certificate for PhilosophyMapTarget$^{\uparrow}$ **12829**

1278 A Concrete Naming Certificate for DiagonalIndexCofinality$^{\uparrow}$ **12833**

1279 A Concrete Naming Certificate for SocketReport$^{\uparrow}$ **12836**

1280 A Concrete Naming Certificate for StatusDowngrade$^{\uparrow}$ **12840**

1281 EventualTailAgreement NameCert Construction **12844**

1282 A Concrete Naming Certificate for PhilosophyFormalTargetLedger$^{\uparrow}$ **12847**

1283 A Concrete Naming Certificate for SocketStackCertificate$^{\uparrow}$ **12851**

1284 A Concrete Naming Certificate for ObjectConstitutionLedger$^{\uparrow}$ **12854**

1285 A Concrete Naming Certificate for PhysicalLawBridge$^{\uparrow}$ **12857**

1286 A Concrete Naming Certificate for ScienceBridge$^{\uparrow}$ **12863**

1287 A Concrete Naming Certificate for ConceptRegistryObligation$^{\uparrow}$ **12867**

1288 A Concrete Naming Certificate for KernelPhaseRefusal$^{\uparrow}$ **12870**

1289 A Concrete Naming Certificate for VerificationFailureRoadmap$^{\uparrow}$ **12874**

1290 A Concrete Naming Certificate for CausalIntervention$^{\uparrow}$ **12878**

1291 A Concrete Naming Certificate for CrossHistoryCausalDependence$^{\uparrow}$ **12881**

1292 A Concrete Naming Certificate for BedcPhilosophyPackage$^{\uparrow}$ **12884**

1293 A Concrete Naming Certificate for PhysicsFalsificationSurface$^{\uparrow}$ **12888**

1294 A Concrete Naming Certificate for AllowedProofAudit$^{\uparrow}$ **12891**

1295 A Concrete Naming Certificate for ObservationLogicBoundary$^{\uparrow}$ **12894**

1296 A Concrete Naming Certificate for AuditExportFirewall$^{\uparrow}$ **12897**

1297 A Concrete Naming Certificate for FormalConstantEmpiricalValueBoundary$^{\uparrow}$ **12900**

1298 A Concrete Naming Certificate for MultiHistoryPhysicsRoute$^{\uparrow}$ **12904**

1299 A Concrete Naming Certificate for CauchyTailThresholdNormalizer$^{\uparrow}$ **12907**

1300 A Concrete Naming Certificate for PhenomenologyScienceInterface$^{\uparrow}$ **12918**

1301 Concrete Naming Certificate for CompactLebesgueUniformHandoff↑ **12924**

1302 Concrete Naming Certificate for CompactCoverLebesgueLedger↑ **12927**

1303 Concrete Naming Certificate for RealityConstrainedSignatureDescent↑ **12931**

1304 Concrete Naming Certificate for PhenomenologicalFrame↑ **12934**

1305 Concrete Naming Certificate for ProgrammeStrengthLedger↑ **12938**

1306 Concrete Naming Certificate for EpistemicWitness↑ **12943**

1307 Concrete Naming Certificate for PackageMapBoundary↑ **12946**

1308 Concrete Naming Certificate for PhilosophyGapMapLedger↑ **12949**

1309 Concrete Naming Certificate for NameEligibilityProof↑ **12952**

1310 Concrete Naming Certificate for UseRuleLedger↑ **12956**

1311 Concrete Naming Certificate for StructuralAdjacency↑ **12959**

1312 Concrete Naming Certificate for LocalEmpiricalClosureGate↑ **12962**

1313 Concrete Naming Certificate for UseProcessBoundary↑ **12965**

1314 Concrete Naming Certificate for ProgrammeStrengthGradientLedger↑ **12968**

1315 Concrete Naming Certificate for UseProcessLimitLedger↑ **12972**

1316 Concrete Naming Certificate for UseProcessLimit↑ **12975**

1317 Concrete Naming Certificate for CertifiedUseProcess↑ **12978**

1318 Concrete Naming Certificate for PhilosophyCannotClaimRegistry↑ **12981**

1319 Concrete Naming Certificate for CannotClaimExportGate↑ **12985**

1320 Concrete Naming Certificate for PhilosophyGovernanceExport↑ **12990**

1321 Concrete Naming Certificate for PackageGovernanceExport↑ **12993**

1322 Concrete Naming Certificate for PhaseGovernedCertificateRoute↑ **12997**

1323 Concrete Naming Certificate for PackageExportDiscipline↑ **13000**

1324 Concrete Naming Certificate for ReviewAcceptanceGate↑ **13003**

1325 Concrete Naming Certificate for RealWindowClassifierSeal↑ **13006**

1326 Concrete Naming Certificate for PhysicsProgrammeSynthesis↑ **13009**

1327 Concrete Naming Certificate for PhaseRealCompletionExit$^{\uparrow}$ **13012**

1328 Concrete Naming Certificate for ResearchProgrammeExecutionGate$^{\uparrow}$ **13033**

1329 Concrete Naming Certificate for SmugglingPatternClassifier$^{\uparrow}$ **13036**

1330 Concrete Naming Certificate for NoAbsoluteBridgeAudit$^{\uparrow}$ **13039**

1331 Concrete Naming Certificate for ResearchProgrammeRoute$^{\uparrow}$ **13042**

1332 Concrete Naming Certificate for ResearchProgrammeAuditLoop$^{\uparrow}$ **13046**

1333 Concrete Naming Certificate for AuditAxisSeparation$^{\uparrow}$ **13049**

1334 Concrete Naming Certificate for DefinitionTheoremProofAudit$^{\uparrow}$ **13052**

1335 Concrete Naming Certificate for ModulusIndexedDiagonalReadback$^{\uparrow}$ **13055**

1336 Concrete Naming Certificate for InscriptionNoveltySocket$^{\uparrow}$ **13058**

1337 Concrete Naming Certificate for AuditFinalityCertificate$^{\uparrow}$ **13061**

1338 Concrete Naming Certificate for FourFaceExitClassifier$^{\uparrow}$ **13064**

1339 Concrete Naming Certificate for EpistemicOntologyBridge$^{\uparrow}$ **13067**

1340 Concrete Naming Certificate for PhysicalRecordInvariant$^{\uparrow}$ **13070**

1341 Concrete Naming Certificate for BareObjectRefusal$^{\uparrow}$ **13074**

1342 Concrete Naming Certificate for CauchySealInterface$^{\uparrow}$ **13077**

1343 Concrete Naming Certificate for GapSocketBoundary$^{\uparrow}$ **13080**

1344 Concrete Naming Certificate for RegistryNoSmugglingPredicate$^{\uparrow}$ **13082**

1345 Concrete Naming Certificate for RecursorBranchCompleteness$^{\uparrow}$ **13085**

1346 Concrete Naming Certificate for TraceVerdictBoundary$^{\uparrow}$ **13088**

1347 Concrete Naming Certificate for FormalTargetDependencyAudit$^{\uparrow}$ **13092**

1348 Concrete Naming Certificate for ExternalSupplySignatureSeal$^{\uparrow}$ **13095**

1349 Concrete Naming Certificate for ClosedGenerationRefusal$^{\uparrow}$ **13098**

1350 Concrete Naming Certificate for TraditionLandingLedger$^{\uparrow}$ **13101**

1351 Concrete Naming Certificate for DiagonalConvergenceBound$^{\uparrow}$ **13104**

1352 Concrete Naming Certificate for RealCofinalModulusGate$^{\uparrow}$ **13107**

1353 Concrete Naming Certificate for ProofPatternClassifier$^{\uparrow}$ **13110**

1354 Concrete Naming Certificate for CanonicalTailChoice$^{\uparrow}$ **13114**

1355 Concrete Naming Certificate for ClosedObservationSupplyBoundary$^{\uparrow}$ **13124**

1356 Concrete Naming Certificate for RealCompletenessTailChoiceAudit$^{\uparrow}$ **13127**

1357 Concrete Naming Certificate for ObservationFiberDigest$^{\uparrow}$ **13130**

1358 Concrete Naming Certificate for RegistryChangeContinuity$^{\uparrow}$ **13133**

1359 Concrete Naming Certificate for HashLikeBoundaryGate$^{\uparrow}$ **13136**

1360 Concrete Naming Certificate for AuditMapRouteBudget$^{\uparrow}$ **13139**

1361 Concrete Naming Certificate for FiniteObservationCofinalitySelector$^{\uparrow}$ **13142**

1362 Concrete Naming Certificate for ActionCommitment$^{\uparrow}$ **13145**

1363 Concrete Naming Certificate for BHistNonaTupleNameCert$^{\uparrow}$ **13148**

1364 Concrete Naming Certificate for BHistDecaTupleNameCert$^{\uparrow}$ **13153**

1365 Concrete Naming Certificate for BHistOctaTupleNameCert$^{\uparrow}$ **13156**

1366 Concrete Naming Certificate for BHistHendecaTupleNameCert$^{\uparrow}$ **13159**

1367 Concrete Naming Certificate for BHistNonaSequenceNameCert$^{\uparrow}$ **13162**

1368 Concrete Naming Certificate for BHistDecaSequenceNameCert$^{\uparrow}$ **13166**

1369 Concrete Naming Certificate for BHistDodecaTupleNameCert$^{\uparrow}$ **13170**

1370 Concrete Naming Certificate for BHistArity13TupleNameCert$^{\uparrow}$ **13173**

1371 Concrete Naming Certificate for BHistArity13SequenceNameCert$^{\uparrow}$ **13177**

1372 Concrete Naming Certificate for BHistArity14SequenceNameCert$^{\uparrow}$ **13181**

1373 Concrete Naming Certificate for BHistArity15SequenceNameCert$^{\uparrow}$ **13185**

1374 Concrete Naming Certificate for StopCodonZeckendorfSuffix$^{\uparrow}$ **13190**

1375 Concrete Naming Certificate for BHistHeptaSequenceNameCert$^{\uparrow}$ **13195**

1376 Concrete Naming Certificate for BHistHeptaTupleNameCert$^{\uparrow}$ **13200**

1377 Concrete Naming Certificate for BHistHendecaSequenceNameCert$^{\uparrow}$ **13203**

1378 Concrete Naming Certificate for Order$^{\uparrow}$ **13206**

1378.1 Unary comparison packets 13206

1379 Heine-Borel Interval NameCert Construction **13212**

1380 A Concrete Naming Certificate for LocatedApartness↑ **13220**

1381 A Concrete Naming Certificate for BishopRealComparison↑ **13223**

1382 A Concrete Naming Certificate for FilteredCauchyCompletion↑ **13228**

1383 A Concrete Naming Certificate for RegularCauchyEquivalence↑ **13231**

1384 A Concrete Naming Certificate for MonotoneSubsequence↑ **13235**

1385 A Concrete Naming Certificate for SpeckerSequence↑ **13238**

1386 A Concrete Naming Certificate for BisectionRootIsolation↑ **13248**

1387 A Concrete Naming Certificate for DiniTheorem↑ **13255**

1388 A Concrete Naming Certificate for BolzanoWeierstrassSelector↑ **13258**

1389 A Concrete Naming Certificate for RationalDensityReal↑ **13272**

1390 A Concrete Naming Certificate for CauchySequenceEquivalence↑ **13275**

1391 A Concrete Naming Certificate for BoundedRealSequence↑ **13277**

1392 A Concrete Naming Certificate for DecimalExpansion↑ **13288**

1393 A Concrete Naming Certificate for ArchimedeanOrderedField↑ **13303**

1394 A Concrete Naming Certificate for LocatedLimit↑ **13308**

1395 A Concrete Naming Certificate for IntermediateValueBisection↑ **13315**

1396 A Concrete Naming Certificate for RealCauchyModulus↑ **13318**

1397 A Concrete Naming Certificate for DedekindCauchyComparison↑ **13332**

1398 A Concrete Naming Certificate for RegularCauchyCompletionUniversal↑ **13335**

1399 A Concrete Naming Certificate for SternBrocotApproximation↑ **13340**

1400 A Concrete Naming Certificate for CauchyCompletionStability↑ **13343**

1401 CauchyContinuousMapUp NameCert Construction **13346**

1402 A Concrete Naming Certificate for IntermediateValue↑ **13357**

1403 A Concrete Naming Certificate for FilteredCauchyCompletion↑ **13367**

1404 A Concrete Naming Certificate for RegularCauchyEquivalence↑ **13370**

1405 A Concrete Naming Certificate for MonotoneSubsequence$^{\uparrow}$ **13374**

1406 A Concrete Naming Certificate for SpeckerSequence$^{\uparrow}$ **13377**

1407 A Concrete Naming Certificate for BisectionRootIsolation$^{\uparrow}$ **13387**

1408 A Concrete Naming Certificate for BolzanoWeierstrassSelector$^{\uparrow}$ **13394**

1409 A Concrete Naming Certificate for RationalDensityReal$^{\uparrow}$ **13408**

1410 A Concrete Naming Certificate for BishopIntervalCompact$^{\uparrow}$ **13411**

1411 A Concrete Naming Certificate for SeparatedMetric$^{\uparrow}$ **13414**

1412 A Concrete Naming Certificate for CauchySequenceEquivalence$^{\uparrow}$ **13443**

1413 A Concrete Naming Certificate for RealSequenceLimit$^{\uparrow}$ **13445**

1414 A Concrete Naming Certificate for CauchyNullEquivalence$^{\uparrow}$ **13459**

1415 A Concrete Naming Certificate for BoundedRealSequence$^{\uparrow}$ **13462**

1416 A Concrete Naming Certificate for DyadicFloor$^{\uparrow}$ **13473**

1417 A Concrete Naming Certificate for DecimalExpansion$^{\uparrow}$ **13477**

1418 A Concrete Naming Certificate for ContractionMapping$^{\uparrow}$ **13492**

1419 A Concrete Naming Certificate for ArchimedeanOrderedField$^{\uparrow}$ **13502**

1420 A Concrete Naming Certificate for LocatedLimit$^{\uparrow}$ **13507**

1421 A Concrete Naming Certificate for CantorIntersection$^{\uparrow}$ **13514**

1422 A Concrete Naming Certificate for IntermediateValueBisection$^{\uparrow}$ **13520**

1423 A Concrete Naming Certificate for RealCauchyModulus$^{\uparrow}$ **13523**

1424 A Concrete Naming Certificate for DedekindCauchyComparison$^{\uparrow}$ **13537**

1425 A Concrete Naming Certificate for RegularCauchyCompletionUniversal$^{\uparrow}$ **13540**

1426 A Concrete Naming Certificate for CauchyControlSequence$^{\uparrow}$ **13545**

1427 A Concrete Naming Certificate for LocatedSupremum$^{\uparrow}$ **13548**

1428 A Concrete Naming Certificate for DyadicIntervalCover$^{\uparrow}$ **13557**

1429 A Concrete Naming Certificate for DecimalEndpointNormalization$^{\uparrow}$ **13587**

1430 A Concrete Naming Certificate for BoundedRealFamily$^{\uparrow}$ **13591**

1431 A Concrete Naming Certificate for SternBrocotApproximation$^{\uparrow}$ **13596**

1432 Rational interval refinement NameCert construction **13599**

1433 A Concrete Naming Certificate for ClosedIntervalCompactNet$^{\uparrow}$ **13604**

1434 A Concrete Naming Certificate for MonotoneRealSequence$^{\uparrow}$ **13607**

1435 A Concrete Naming Certificate for MonotoneSequenceConvergence$^{\uparrow}$ **13611**

1436 A Concrete Naming Certificate for DiniUniformConvergence$^{\uparrow}$ **13614**

1437 A Concrete Naming Certificate for CompactIntervalFixedPoint$^{\uparrow}$ **13620**

1438 A Concrete Naming Certificate for CauchyConvergenceModulus$^{\uparrow}$ **13628**

1439 A Concrete Naming Certificate for StolzCesaro$^{\uparrow}$ **13641**

1440 A Concrete Naming Certificate for BoundedRealLowerCut$^{\uparrow}$ **13646**

1441 A Concrete Naming Certificate for RegularLocatedCauchyReal$^{\uparrow}$ **13649**

1442 A Concrete Naming Certificate for SequentialContinuity$^{\uparrow}$ **13653**

1443 A Concrete Naming Certificate for SubsequenceLimit$^{\uparrow}$ **13657**

1444 A Concrete Naming Certificate for RegularCauchyReciprocal$^{\uparrow}$ **13660**

1445 A Concrete Naming Certificate for RegularCauchyFilter$^{\uparrow}$ **13669**

1446 A Concrete Naming Certificate for LowerReal$^{\uparrow}$ **13681**

1447 A Concrete Naming Certificate for UpperReal$^{\uparrow}$ **13689**

1448 A Concrete Naming Certificate for ClosedUnitInterval$^{\uparrow}$ **13694**

1449 A Concrete Naming Certificate for BoundedRealFamily$^{\uparrow}$ **13697**

1450 A Concrete Naming Certificate for MetricBall$^{\uparrow}$ **13702**

1451 A Concrete Naming Certificate for MetricClosedBall$^{\uparrow}$ **13708**

1452 A Concrete Naming Certificate for LocatedInfimum$^{\uparrow}$ **13714**

1453 A Concrete Naming Certificate for RegularCauchyDenseEmbedding$^{\uparrow}$ **13723**

1454 A Concrete Naming Certificate for RegularLocatedCauchyReal$^{\uparrow}$ **13726**

1455 A Concrete Naming Certificate for BishopLocatedCompletionBoundary$^{\uparrow}$ **13730**

1456 A Concrete Naming Certificate for CauchyCompletionAdjunction$^{\uparrow}$ **13737**

1457 A Concrete Naming Certificate for DedekindCauchySeal↑ **13741**

1458 SeparableMetricUp NameCert Construction **13744**

1459 A Concrete Naming Certificate for RegularCauchyComparisonTest↑ **13751**

1460 A Concrete Naming Certificate for DarbouxIntegral↑ **13754**
1460.1 Carrier packet 13754
1460.2 Certificate obligations 13755

1461 A Concrete Naming Certificate for RegularCauchyRatioTest↑ **13759**

1462 A Concrete Naming Certificate for ConnectedInterval↑ **13763**

1463 Completion Uniqueness NameCert Construction **13768**

1464 A Concrete Naming Certificate for CauchyCondensation↑ **13771**

1465 A Concrete Naming Certificate for IntervalBisection↑ **13778**

1466 A Concrete Naming Certificate for SequentialContinuity↑ **13784**

1467 A Concrete Naming Certificate for SubsequenceLimit↑ **13788**

1468 A Concrete Naming Certificate for AdjointOperator↑ **13791**
1468.1 Carrier packet 13791
1468.2 Adjunction law 13792
1468.3 Name certificate surface 13792

1469 A Concrete Naming Certificate for BishopCompletionModulus↑ **13794**

1470 RegularCauchyRing NameCert Construction **13801**

1471 A Concrete Naming Certificate for RegularCauchyReciprocal↑ **13810**

1472 CompactLocatedSubcover NameCert Construction **13819**

1473 A Concrete Naming Certificate for UniformCompletion↑ **13822**

1474 A Concrete Naming Certificate for CalkinWilfEnumeration↑ **13830**

1475 A Concrete Naming Certificate for CantorSpace↑ **13833**

1476 A Concrete Naming Certificate for CauchyCompletionCriterion↑ **13839**

1477 A Concrete Naming Certificate for DyadicMax↑ **13847**

1478 A Concrete Naming Certificate for CauchyCondensationTest↑ **13850**

1479 A Concrete Naming Certificate for ArchimedeanIntegerPart↑ **13854**

1480 A Concrete Naming Certificate for RealSequenceLimitLaws↑ **13858**

1481 Concrete Naming Certificate for RegularCauchyRootTest$^{\uparrow}$ **13861**

1482 Concrete Naming Certificate for ChoiceSequence$^{\uparrow}$ **13864**

1483 Concrete Naming Certificate for AlternatingSeriesTest$^{\uparrow}$ **13868**

1484 Concrete Naming Certificate for DyadicDensityReal$^{\uparrow}$ **13872**

1485 Concrete Naming Certificate for CauchyModulusArithmetic$^{\uparrow}$ **13875**

1486 Concrete Naming Certificate for CauchyModulusOfContinuity$^{\uparrow}$ **13891**

1487 Concrete Naming Certificate for RegularCauchyTailEstimate$^{\uparrow}$ **13898**

1488 Concrete Naming Certificate for LocatedCompact$^{\uparrow}$ **13909**

1489 Concrete Naming Certificate for IntervalHalving$^{\uparrow}$ **13914**

1490 Concrete Naming Certificate for PolishSpace$^{\uparrow}$ **13918**

1491 Concrete Naming Certificate for RationalEmbedding$^{\uparrow}$ **13960**

1492 Concrete Naming Certificate for BolzanoWeierstrassBoundary$^{\uparrow}$ **13963**

1493 Concrete Naming Certificate for ExtremeValue$^{\uparrow}$ **13967**

1494 Concrete Naming Certificate for RepresentedSpace$^{\uparrow}$ **13971**

1495 Concrete Naming Certificate for LawlessSequence$^{\uparrow}$ **13983**

1496 Concrete Naming Certificate for AbelTest$^{\uparrow}$ **13990**

1497 Concrete Naming Certificate for DirichletTest$^{\uparrow}$ **13993**

1498 Concrete Naming Certificate for LocatedOpen$^{\uparrow}$ **13997**

1499 Concrete Naming Certificate for RegularCauchyAbs$^{\uparrow}$ **14000**

1500 Concrete Naming Certificate for NormedRealLine$^{\uparrow}$ **14004**

1501 Concrete Naming Certificate for CauchyFilterBasis$^{\uparrow}$ **14007**

1502 Concrete Naming Certificate for RegularCauchyMax$^{\uparrow}$ **14011**

1503 Concrete Naming Certificate for MetricDiameter$^{\uparrow}$ **14015**

1504 Concrete Naming Certificate for DedekindCompleteReal$^{\uparrow}$ **14018**

1505 Concrete Naming Certificate for DedekindCompleteness$^{\uparrow}$ **14021**

1506 Concrete Naming Certificate for MetricCompletion$^{\uparrow}$ **14024**

1507 A Concrete Naming Certificate for ArchimedeanApproximation↑ **14046**

1508 A Concrete Naming Certificate for BishopCompletionComparison↑ **14052**

1509 A Concrete Naming Certificate for CauchyConvergenceCriterion↑ **14057**

1510 A Concrete Naming Certificate for FormalBall↑ **14070**

1511 A Concrete Naming Certificate for CauchyRealPresentation↑ **14085**

1512 A Concrete Naming Certificate for CompletionEmbedding↑ **14088**

1513 A Concrete Naming Certificate for RegularCauchyCompletionSeparated↑ **14093**

1514 A Concrete Naming Certificate for ApartnessSpace↑ **14096**

1515 A Concrete Naming Certificate for SequentiallyCompleteMetric↑ **14103**

1516 A Concrete Naming Certificate for RegularCauchyBounded↑ **14129**

1517 A Concrete Naming Certificate for RealNameClassifier↑ **14132**

1518 A Concrete Naming Certificate for FastCauchySubsequence↑ **14142**

1519 A Concrete Naming Certificate for MetricSubspace↑ **14170**

1520 A Concrete Naming Certificate for BishopRegularCauchyPair↑ **14173**

1521 A Concrete Naming Certificate for MooreSmithConvergence↑ **14177**

1522 A Concrete Naming Certificate for CauchySequenceBounded↑ **14185**

1523 A Concrete Naming Certificate for AitkenDeltaSquared↑ **14192**

1524 A Concrete Naming Certificate for DyadicRoundingWindow↑ **14195**

1525 A Concrete Naming Certificate for FanFunctional↑ **14203**

1526 A Concrete Naming Certificate for RiemannIntegral↑ **14233**

1527 A Concrete Naming Certificate for CauchyCompletionUnit↑ **14241**

1528 A Concrete Naming Certificate for RegularCauchyMin↑ **14246**

1529 A Concrete Naming Certificate for CauchyProductBounded↑ **14258**

1530 A Concrete Naming Certificate for CantorSet↑ **14262**

1531 A Concrete Naming Certificate for PrecompactMetric↑ **14275**

1532 A Concrete Naming Certificate for SpreadLaw↑ **14282**

1533 Concrete Naming Certificate for LocatedMetric$^{\uparrow}$ **14285**

1534 Concrete Naming Certificate for SturmRootIsolation$^{\uparrow}$ **14291**

1535 Concrete Naming Certificate for UltrametricSpace$^{\uparrow}$ **14298**

1536 Concrete Naming Certificate for LowerSemicontinuous$^{\uparrow}$ **14303**

1537 Concrete Naming Certificate for ClosedGraphTheorem$^{\uparrow}$ **14345**

1538 Concrete Naming Certificate for OpenMappingTheorem$^{\uparrow}$ **14349**

1539 Concrete Naming Certificate for CauchyCompletionCounit$^{\uparrow}$ **14352**

1540 Concrete Naming Certificate for CauchyCompletionProduct$^{\uparrow}$ **14355**

1541 Concrete Naming Certificate for LocatedSet$^{\uparrow}$ **14359**

1542 Concrete Naming Certificate for BaireCategory$^{\uparrow}$ **14374**

1543 Concrete Naming Certificate for IntervalDomain$^{\uparrow}$ **14388**

1544 Concrete Naming Certificate for RealSequentialCompleteness$^{\uparrow}$ **14397**

1545 Concrete Naming Certificate for IntervalDomainWayBelow$^{\uparrow}$ **14401**

1546 Concrete Naming Certificate for CauchyRealLocator$^{\uparrow}$ **14406**

1547 Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$ **14410**

1548 Concrete Naming Certificate for CauchySequenceAlgebra$^{\uparrow}$ **14415**

1549 Concrete Naming Certificate for Cauchy$^{\uparrow}$ **14421**

1550 Concrete Naming Certificate for IntegerRealEmbedding$^{\uparrow}$ **14431**

1551 Concrete Naming Certificate for IsometryExtension$^{\uparrow}$ **14435**

1552 Concrete Naming Certificate for CauchySubsequenceCriterion$^{\uparrow}$ **14439**

1553 Concrete Naming Certificate for Liminf$^{\uparrow}$ **14442**

1554 Concrete Naming Certificate for RegulatedRealFunction$^{\uparrow}$ **14451**

1555 Concrete Naming Certificate for UniformCauchyFamily$^{\uparrow}$ **14456**

1556 Concrete Naming Certificate for ConstructiveMetricCompletion$^{\uparrow}$ **14461**

1557 Concrete Naming Certificate for RegulatedIntegral$^{\uparrow}$ **14464**

1558 Concrete Naming Certificate for CauchyFilterLimit$^{\uparrow}$ **14470**

1559 A Concrete Naming Certificate for MetricCompletionFinite$^{\uparrow}$ **14476**

1560 A Concrete Naming Certificate for CompletionDensity$^{\uparrow}$ **14485**

1561 A Concrete Naming Certificate for CauchyNetLimit$^{\uparrow}$ **14488**

1562 A Concrete Naming Certificate for DyadicRatCoreArithmetic$^{\uparrow}$ **14492**

1563 A Concrete Naming Certificate for CauchyShift$^{\uparrow}$ **14495**

1564 A Concrete Naming Certificate for FrechetFilter$^{\uparrow}$ **14499**

1565 A Concrete Naming Certificate for CauchyCompletionUnitNaturality$^{\uparrow}$ **14511**

1566 A Concrete Naming Certificate for CountableDenseSubset$^{\uparrow}$ **14514**

1567 A Concrete Naming Certificate for RealApartnessCompletion$^{\uparrow}$ **14517**

1568 A Concrete Naming Certificate for CauchyReindexing$^{\uparrow}$ **14526**

1569 A Concrete Naming Certificate for CauchyFilterCompletion$^{\uparrow}$ **14530**

1570 A Concrete Naming Certificate for IntervalNewton$^{\uparrow}$ **14568**

1571 A Concrete Naming Certificate for SequentialClosure$^{\uparrow}$ **14575**

1572 A Concrete Naming Certificate for CofinalCauchyTail$^{\uparrow}$ **14586**

1573 A Concrete Naming Certificate for TotallyBoundedCompletion$^{\uparrow}$ **14589**

1574 A Concrete Naming Certificate for PrincipalCauchyFilter$^{\uparrow}$ **14607**

1575 A Concrete Naming Certificate for LocatedCauchyEmbedding$^{\uparrow}$ **14610**

1576 A Concrete Naming Certificate for RealCauchyModulusSelector$^{\uparrow}$ **14613**

1577 A Concrete Naming Certificate for RegularCauchyModulusRealization$^{\uparrow}$ **14619**

1578 A Concrete Naming Certificate for RegularCauchyShift$^{\uparrow}$ **14622**

1579 A Concrete Naming Certificate for LimitPoint$^{\uparrow}$ **14625**

1580 A Concrete Naming Certificate for RealSequenceClusterPoint$^{\uparrow}$ **14628**

1581 A Concrete Naming Certificate for RealSeries$^{\uparrow}$ **14632**

1582 A Concrete Naming Certificate for CauchyLocalization$^{\uparrow}$ **14656**

1583 A Concrete Naming Certificate for RealPowerSeries$^{\uparrow}$ **14659**

1584 A Concrete Naming Certificate for NestedIntervalCompactness$^{\uparrow}$ **14680**

1585 A Concrete Naming Certificate for CauchyCompletionReflector↑ **14687**

1586 A Concrete Naming Certificate for CauchyProductCriterion↑ **14697**

1587 A Concrete Naming Certificate for PositiveReal↑ **14701**
1587.1 Carrier and positivity packet 14701
1587.2 NameCert obligations 14702

1588 A Concrete Naming Certificate for CompleteUniformSpace↑ **14708**

1589 A Concrete Naming Certificate for FastCauchyReal↑ **14712**

1590 A Concrete Naming Certificate for RealCauchyCriterion↑ **14716**

1591 A Concrete Naming Certificate for ClosedBoundedInterval↑ **14720**

1592 A Concrete Naming Certificate for CauchySubnet↑ **14754**

1593 A Concrete Naming Certificate for GershgorinDisk↑ **14761**

1594 A Concrete Naming Certificate for CofinalSequence↑ **14764**

1595 A Concrete Naming Certificate for LocatedCauchySequence↑ **14767**

1596 A Concrete Naming Certificate for IntervalFiniteCoverCertificate↑ **14770**

1597 A Concrete Naming Certificate for DyadicOrder↑ **14774**

1598 A Concrete Naming Certificate for CompleteOrderedReal↑ **14777**

1599 A Concrete Naming Certificate for DedekindMacNeilleCompletion↑ **14780**

1600 A Concrete Naming Certificate for DirectedSubnet↑ **14783**

1601 A Concrete Naming Certificate for FilterBase↑ **14788**

1602 A Concrete Naming Certificate for BaireMetric↑ **14793**

1603 A Concrete Naming Certificate for SchurOrthogonality↑ **14813**

1604 A Concrete Naming Certificate for BaireOneFunction↑ **14818**

1605 A Concrete Naming Certificate for RegularCauchyTailFilter↑ **14825**

1606 A Concrete Naming Certificate for CauchyCompletionMinimality↑ **14829**

1607 A Concrete Naming Certificate for DirectedSet↑ **14839**

1608 A Concrete Naming Certificate for RealAbs↑ **14845**

1609 A Concrete Naming Certificate for EventuallyZeroSequence↑ **14847**

1610 A Concrete Naming Certificate for HeineCantorModulus$^{\uparrow}$ **14851**

1611 A Concrete Naming Certificate for RegularCauchyCriterion$^{\uparrow}$ **14855**

1612 A Concrete Naming Certificate for SeparatedCompletion$^{\uparrow}$ **14862**

1613 A Concrete Naming Certificate for DyadicNormalForm$^{\uparrow}$ **14891**

1614 A Concrete Naming Certificate for Pseudometric$^{\uparrow}$ **14894**

1615 A Concrete Naming Certificate for RiemannSum$^{\uparrow}$ **14911**

1616 A Concrete Naming Certificate for RegularCauchyCofinalRate$^{\uparrow}$ **14917**

1617 A Concrete Naming Certificate for CauchyModulusComposition$^{\uparrow}$ **14920**

1618 A Concrete Naming Certificate for CauchyTailSelectionCriterion$^{\uparrow}$ **14923**

1619 A Concrete Naming Certificate for RealZero$^{\uparrow}$ **14926**

1620 A Concrete Naming Certificate for RealClassifier$^{\uparrow}$ **14933**

1621 A Concrete Naming Certificate for StationaryDyadicStream$^{\uparrow}$ **14947**

1622 A Concrete Naming Certificate for KleeneTree$^{\uparrow}$ **14951**

1623 A Concrete Naming Certificate for DyadicComparison$^{\uparrow}$ **14958**

1624 A Concrete Naming Certificate for RealUniformStructure$^{\uparrow}$ **14962**

1625 A Concrete Naming Certificate for RealIntervalEnclosure$^{\uparrow}$ **14974**

1626 A Concrete Naming Certificate for FilterRefinement$^{\uparrow}$ **14977**

1627 A Concrete Naming Certificate for LocatednessModulus$^{\uparrow}$ **14981**

1628 A Concrete Naming Certificate for RegularCauchyUniformity$^{\uparrow}$ **14985**

1629 A Concrete Naming Certificate for Upcrossing$^{\uparrow}$ **14989**

1629.1 Finite Crossing Carrier 14989

1629.2 Certificate Obligations 14990

1630 A Concrete Naming Certificate for DoobUpcrossing$^{\uparrow}$ **14995**

1631 A Concrete Naming Certificate for ModulusContinuity$^{\uparrow}$ **14998**

1632 A Concrete Naming Certificate for CauchyContinuousExtension$^{\uparrow}$ **15005**

1633 A Concrete Naming Certificate for DyadicTailBall$^{\uparrow}$ **15011**

1634 A Concrete Naming Certificate for WeierstrassApproximation$^{\uparrow}$ **15019**

1635 A Concrete Naming Certificate for CauchyCompletionAssociativity↑ **15023**

1636 Real Equality Name Certificate **15027**

1637 A Concrete Naming Certificate for RiemannStieltjes↑ **15030**

1638 A Concrete Naming Certificate for RealMetric↑ **15040**

1639 A Concrete Naming Certificate for BishopRegularCauchyCompletion↑ **15045**

1640 A Concrete Naming Certificate for UpperSemicontinuous↑ **15060**

1641 A Concrete Naming Certificate for SequentialCompactness↑ **15063**

1642 A Concrete Naming Certificate for CauchyContinuityPrinciple↑ **15068**

1643 A Concrete Naming Certificate for RieszRepresentation↑ **15072**

1643.1 Carrier definition 15072

1643.2 Certificate obligations 15073

1644 A Concrete Naming Certificate for BoundedLinearOperator↑ **15080**

1645 A Concrete Naming Certificate for DyadicIntervalArithmetic↑ **15084**

1646 A Concrete Naming Certificate for RealMetricLocatednessBracket↑ **15088**

1647 A Concrete Naming Certificate for EvenOddCauchyCriterion↑ **15091**

1648 A Concrete Naming Certificate for FractionalPart↑ **15098**

1649 A Concrete Naming Certificate for StableManifold↑ **15101**

1650 A Concrete Naming Certificate for RegularCauchySeparation↑ **15104**

1651 A Concrete Naming Certificate for DenseUniformEmbedding↑ **15108**

1652 A Concrete Naming Certificate for UniformEmbedding↑ **15112**

1653 A Concrete Naming Certificate for DyadicSupremum↑ **15115**

1654 A Concrete Naming Certificate for RegularCauchyMetric↑ **15119**

1655 A Concrete Naming Certificate for MetricTopology↑ **15123**

1656 A Concrete Naming Certificate for FastRegularCauchyEquivalence↑ **15127**

1657 A Concrete Naming Certificate for CauchyUniformNet↑ **15131**

1658 A Concrete Naming Certificate for Dyadic↑ **15134**

1659 A Concrete Naming Certificate for DyadicCauchyTailEnvelope↑ **15141**

1660 A Concrete Naming Certificate for Hyperspace$^{\uparrow}$ **15152**
1660.1 Downstream Dependency Lattice 15178

1661 A Concrete Naming Certificate for LocatedCauchyFilter$^{\uparrow}$ **15182**

1662 A Concrete Naming Certificate for RealUniformEmbedding$^{\uparrow}$ **15188**

1663 A Concrete Naming Certificate for ApartnessTopology$^{\uparrow}$ **15191**

1664 A Concrete Naming Certificate for RealOne$^{\uparrow}$ **15194**

1665 A Concrete Naming Certificate for IshiharaTrick$^{\uparrow}$ **15197**

1666 A Concrete Naming Certificate for MarkovPrincipleBoundary$^{\uparrow}$ **15203**

1667 A Concrete Naming Certificate for NetConvergence$^{\uparrow}$ **15206**

1668 A Concrete Naming Certificate for RealRationalApproximation$^{\uparrow}$ **15215**

1669 RealLocatorUp NameCert Construction **15218**

1670 A Concrete Naming Certificate for RegularCauchyMidpoint$^{\uparrow}$ **15221**

1671 A Concrete Naming Certificate for CauchyRateComparison$^{\uparrow}$ **15224**

1672 A Concrete Naming Certificate for DyadicArchimedean$^{\uparrow}$ **15227**

1673 A Concrete Naming Certificate for CauchyRateRealization$^{\uparrow}$ **15232**

1674 A Concrete Naming Certificate for RealMinMax$^{\uparrow}$ **15236**

1675 A Concrete Naming Certificate for BrouwerBarInduction$^{\uparrow}$ **15239**

1676 A Concrete Naming Certificate for BrouwerOperation$^{\uparrow}$ **15243**

1677 A Concrete Naming Certificate for BrouwerSpread$^{\uparrow}$ **15246**

1678 A Concrete Naming Certificate for Limit$^{\uparrow}$ **15250**

1679 A Concrete Naming Certificate for RegularCauchyAffineCombination$^{\uparrow}$ **15256**

1680 A Concrete Naming Certificate for CauchyRepresentation$^{\uparrow}$ **15261**

1681 A Concrete Naming Certificate for RegularCauchyLocatedOrder$^{\uparrow}$ **15264**

1682 A Concrete Naming Certificate for ToeplitzLemma$^{\uparrow}$ **15267**

1683 A Concrete Naming Certificate for SequentialCompletion$^{\uparrow}$ **15271**

1684 A Concrete Naming Certificate for CauchySchwarzReal$^{\uparrow}$ **15274**

1685 A Concrete Naming Certificate for RealBaireCylinder$^{\uparrow}$ **15279**

1686 A Concrete Naming Certificate for MooreOsgood$^{\uparrow}$ **15282**

1687 A Concrete Naming Certificate for CompactMetricCompletion$^{\uparrow}$ **15286**

1688 A Concrete Naming Certificate for MinimalCauchyFilter$^{\uparrow}$ **15290**

1689 A Concrete Naming Certificate for DoubleCauchyDiagonal$^{\uparrow}$ **15294**

1690 A Concrete Naming Certificate for DyadicCeiling$^{\uparrow}$ **15298**

1691 A Concrete Naming Certificate for FastCauchyCompletion$^{\uparrow}$ **15301**

1692 A Concrete Naming Certificate for LocatedUniformCompletion$^{\uparrow}$ **15305**

1693 A Concrete Naming Certificate for FanBarRoute$^{\uparrow}$ **15309**

1694 A Concrete Naming Certificate for UniformCauchyFilter$^{\uparrow}$ **15313**

1695 A Concrete Naming Certificate for RegularCauchyAddition$^{\uparrow}$ **15316**

1696 A Concrete Naming Certificate for RegularCauchySubtraction$^{\uparrow}$ **15321**

1697 A Concrete Naming Certificate for BoundedRegularCauchy$^{\uparrow}$ **15324**

1698 A Concrete Naming Certificate for BoundedRegularCauchySubsequence$^{\uparrow}$ **15330**

1699 A Concrete Naming Certificate for UniformCauchySubsequence$^{\uparrow}$ **15334**

1700 A Concrete Naming Certificate for AbsoluteConvergence$^{\uparrow}$ **15337**

1701 A Concrete Naming Certificate for ConvergentRealSequenceCauchy$^{\uparrow}$ **15341**

1702 A Concrete Naming Certificate for RegularCauchyMultiplication$^{\uparrow}$ **15345**

1703 A Concrete Naming Certificate for PrimitiveRecursion$^{\uparrow}$ **15348**

1704 A Concrete Naming Certificate for SternBrocot$^{\uparrow}$ **15352**

1705 A Concrete Naming Certificate for EffectiveCauchySequence$^{\uparrow}$ **15356**

1706 A Concrete Naming Certificate for EffectivePolishSpace$^{\uparrow}$ **15361**

1707 A Concrete Naming Certificate for BishopFanModulus$^{\uparrow}$ **15366**

1708 A Concrete Naming Certificate for CauchyQuotientBoundary$^{\uparrow}$ **15370**

1709 Located Real Interval NameCert Construction **15374**

1710 A Concrete Naming Certificate for RegularCauchyApartnessOrder$^{\uparrow}$ **15378**

1711 A Concrete Naming Certificate for LocatedIntervalBisection$^{\uparrow}$ **15382**

1712 Concrete Naming Certificate for LocatedIntervalMidpoint↑ **15386**

1713 Concrete Naming Certificate for LocatedRealNestedIntervals↑ **15389**

1714 Concrete Naming Certificate for SeparableCompletion↑ **15394**

1715 Concrete Naming Certificate for CompletionPreservesSeparable↑ **15397**

1716 Concrete Naming Certificate for CauchyRealOrder↑ **15400**

1717 Concrete Naming Certificate for CauchyRealField↑ **15404**

1718 Concrete Naming Certificate for RegularCauchyFieldDistributivity↑ **15409**

1719 Concrete Naming Certificate for RaabeDuhamelTest↑ **15413**

1720 Concrete Naming Certificate for CauchyRealizerSequence↑ **15416**

1721 Concrete Naming Certificate for RegularCauchyShuffle↑ **15420**

1722 Concrete Naming Certificate for RegularCauchySubsequenceInvariance↑ **15424**

1723 Concrete Naming Certificate for CauchyRealizerTailModulus↑ **15427**

1724 Concrete Naming Certificate for DecimalNormality↑ **15431**

1725 Concrete Naming Certificate for CompletionExtractor↑ **15434**

1726 Concrete Naming Certificate for LocallyCompact↑ **15437**

1727 Concrete Naming Certificate for RegularCauchyZeroDistance↑ **15447**

1728 Concrete Naming Certificate for RealPolynomial↑ **15450**

1729 Concrete Naming Certificate for RealSequenceAlgebra↑ **15454**

1730 Concrete Naming Certificate for LocatedRealComparison↑ **15459**

1731 Concrete Naming Certificate for KuratowskiEmbedding↑ **15463**

1732 Concrete Naming Certificate for HellySelection↑ **15466**

1733 Concrete Naming Certificate for SequenceFilterBridge↑ **15470**

1734 Concrete Naming Certificate for CauchyFilterCluster↑ **15474**

1735 Concrete Naming Certificate for CompleteSeparableMetric↑ **15478**

1736 Concrete Naming Certificate for UniformCauchySeries↑ **15483**

1737 Concrete Naming Certificate for CauchyApproximationSequence↑ **15487**

1738 Concrete Naming Certificate for CauchyTailEquivalence↑ **15490**

1739 Concrete Naming Certificate for UniqueChoicePrinciple↑ **15494**

1740 Concrete Naming Certificate for RegularCauchyInterpolation↑ **15497**

1741 Concrete Naming Certificate for UniformCauchyModulus↑ **15501**

1742 Concrete Naming Certificate for RegularCauchySquare↑ **15506**

1743 Concrete Naming Certificate for KreiselLacombeShoenfield↑ **15509**

1744 Cauchy Modulus Monotone Normalization NameCert Construction **15514**

1745 Concrete Naming Certificate for DedekindCutCauchyBoundary↑ **15518**

1746 Concrete Naming Certificate for CauchyTailDiameter↑ **15522**

1747 Concrete Naming Certificate for RegularCauchyTailComposition↑ **15526**

1748 Concrete Naming Certificate for UniformLimitContinuous↑ **15530**

1749 Concrete Naming Certificate for UniformLimitInterchange↑ **15534**

1750 Concrete Naming Certificate for RegularCauchyTailMonotonicity↑ **15538**

1751 Concrete Naming Certificate for UniformLimitContinuity↑ **15542**

1752 Concrete Naming Certificate for CantorCompleteness↑ **15546**

1753 Concrete Naming Certificate for BishopModulusReal↑ **15549**

1754 Concrete Naming Certificate for RegularCauchyTailModulus↑ **15552**

1755 Concrete Naming Certificate for RegularCauchyTailModulusReuse↑ **15556**

1756 Concrete Naming Certificate for RegularCauchySquareRoot↑ **15559**

1757 Concrete Naming Certificate for BoundedSet↑ **15562**

1758 Concrete Naming Certificate for CauchyNullDifference↑ **15576**

1759 Concrete Naming Certificate for FareySequence↑ **15579**

1760 Concrete Naming Certificate for AdmissibleRepresentation↑ **15605**

1761 Concrete Naming Certificate for DyadicErrorBudget↑ **15608**

1762 Concrete Naming Certificate for BishopModulusProductClosure↑ **15612**

1763 Concrete Naming Certificate for AbelSummation↑ **15616**

1764 Concrete Naming Certificate for UniformLimitMetric↑ **15620**

1765 Concrete Naming Certificate for HardyCesaroMean↑ **15623**

1766 Concrete Naming Certificate for HolderContinuity↑ **15626**

1767 Concrete Naming Certificate for FiniteEpsilonNet↑ **15629**

1768 Concrete Naming Certificate for DyadicToleranceScale↑ **15633**

1769 Concrete Naming Certificate for LocatedRealTailEnvelope↑ **15637**

1770 Concrete Naming Certificate for RegularCauchyDistance↑ **15640**

1771 Concrete Naming Certificate for CauchyRealizer↑ **15643**

1772 Concrete Naming Certificate for RegularCauchySqueeze↑ **15647**

1773 Concrete Naming Certificate for RegularCauchyDiagonalExtraction↑ **15652**

1774 Concrete Naming Certificate for CauchyMajorantSeries↑ **15655**

1775 Concrete Naming Certificate for CauchyContinuousImage↑ **15658**

1776 Concrete Naming Certificate for BishopCompletionInduction↑ **15661**

1777 Concrete Naming Certificate for ContinuedFractionConvergents↑ **15669**

1778 Concrete Naming Certificate for ClosedIntervalCauchyCompletion↑ **15673**

1779 Concrete Naming Certificate for ClosedIntervalCauchyCriterion↑ **15678**

1780 Concrete Naming Certificate for RationalCauchyGap↑ **15682**

1781 Concrete Naming Certificate for FastCauchyModulus↑ **15685**

1782 Concrete Naming Certificate for RegularCauchyCompletionUniqueness↑ **15691**

1783 Concrete Naming Certificate for RegularCauchyComparisonPrinciple↑ **15694**

1784 Concrete Naming Certificate for RealIntegerPart↑ **15699**

1785 Concrete Naming Certificate for NewtonKantorovich↑ **15703**

1786 Concrete Naming Certificate for RealRootBracket↑ **15707**

1787 Concrete Naming Certificate for ApartnessCompletion↑ **15711**

1788 Concrete Naming Certificate for ApartnessCotransitivity↑ **15714**

1789 Concrete Naming Certificate for CauchyNetFilterCorrespondence↑ **15717**

179.0 Concrete Naming Certificate for SeparatedMetricReflection$^{\uparrow}$ **15721**

179.1 Concrete Naming Certificate for CauchyEntourageFilter$^{\uparrow}$ **15727**

179.2 Concrete Naming Certificate for FiniteIntersectionProperty$^{\uparrow}$ **15730**

179.3 Concrete Naming Certificate for CauchyCompletionReflectiveSubcategory$^{\uparrow}$ **15735**

179.4 Concrete Naming Certificate for RegularCauchySequentialCompleteness$^{\uparrow}$ **15739**

179.5 Concrete Naming Certificate for CauchyCompletionTriangleIdentity$^{\uparrow}$ **15745**

179.6 Concrete Naming Certificate for MetricCompletionUniqueness$^{\uparrow}$ **15749**

179.7 Concrete Naming Certificate for DyadicEmbeddingUniqueness$^{\uparrow}$ **15753**

179.8 Concrete Naming Certificate for BisectionRootCauchySeal$^{\uparrow}$ **15756**

179.9 Concrete Naming Certificate for SummableTailModulus$^{\uparrow}$ **15760**

180.0 Concrete Naming Certificate for CauchyDiagonalLimit$^{\uparrow}$ **15764**

180.1 Concrete Naming Certificate for LocatedSequenceModulus$^{\uparrow}$ **15768**

180.2 Rational Interval Enclosure NameCert Construction **15771**

180.3 Concrete Naming Certificate for CauchyFilterNetCorrespondence$^{\uparrow}$ **15775**

180.4 Concrete Naming Certificate for BisectionConvergence$^{\uparrow}$ **15779**

180.5 Concrete Naming Certificate for LocatedCompletionSeal$^{\uparrow}$ **15783**

180.6 Concrete Naming Certificate for BishopRealUniformCompletion$^{\uparrow}$ **15786**

180.7 Concrete Naming Certificate for FiniteBracketingInterval$^{\uparrow}$ **15791**

180.8 Concrete Naming Certificate for LocatedClosedIntervalFiniteNet$^{\uparrow}$ **15795**

180.9 Concrete Naming Certificate for RealSeparability$^{\uparrow}$ **15798**

181.0 Concrete Naming Certificate for RealClamping$^{\uparrow}$ **15808**

181.1 Concrete Naming Certificate for NestedDyadicCauchy$^{\uparrow}$ **15849**

181.2 Concrete Naming Certificate for DyadicIntervalNesting$^{\uparrow}$ **15855**

181.3 Concrete Naming Certificate for DyadicExponentShift$^{\uparrow}$ **15860**

181.4 Concrete Naming Certificate for CauchyRateDominance$^{\uparrow}$ **15864**

181.5 Concrete Naming Certificate for UniformHomeomorphism$^{\uparrow}$ **15867**

1816 A Concrete Naming Certificate for RealExponential$^{\uparrow}$ **15873**

1817 A Concrete Naming Certificate for ExpLogBisection$^{\uparrow}$ **15877**

1818 A Concrete Naming Certificate for CauchyRateLattice$^{\uparrow}$ **15880**

1819 A Concrete Naming Certificate for EntourageBase$^{\uparrow}$ **15884**

1820 A Concrete Naming Certificate for CauchyEquivalenceSetoid$^{\uparrow}$ **15887**

1821 A Concrete Naming Certificate for BishopRegularCauchyLimitSeal$^{\uparrow}$ **15919**

1822 A Concrete Naming Certificate for CompactMetricProduct$^{\uparrow}$ **15922**

1823 A Concrete Naming Certificate for ConvergentSequenceCauchy$^{\uparrow}$ **15926**

1824 A Concrete Naming Certificate for RegularCauchyCompletionEndpoint$^{\uparrow}$ **15930**

1825 A Concrete Naming Certificate for CauchyEstimate$^{\uparrow}$ **15933**

1826 A Concrete Naming Certificate for DiagonalSubsequence$^{\uparrow}$ **15936**

1827 A Concrete Naming Certificate for CesaroConvergence$^{\uparrow}$ **15940**

1828 A Concrete Naming Certificate for RegularCauchyLocationWitness$^{\uparrow}$ **15943**

1829 A Concrete Naming Certificate for CauchyCompletionMultiplication$^{\uparrow}$ **15946**

1830 A Concrete Naming Certificate for RealFunctionUniformCauchy$^{\uparrow}$ **15951**

1831 A Concrete Naming Certificate for DyadicIntervalRefinement$^{\uparrow}$ **15954**

1832 A Concrete Naming Certificate for RegularCauchyMinimum$^{\uparrow}$ **15957**

1833 A Concrete Naming Certificate for RegularCauchyMaximum$^{\uparrow}$ **15961**

1834 A Concrete Naming Certificate for ImplicitFunction$^{\uparrow}$ **15964**

1835 A Concrete Naming Certificate for UniformBoundedness$^{\uparrow}$ **15969**

1836 A Concrete Naming Certificate for RegularCauchyComposition$^{\uparrow}$ **15975**

1837 A Concrete Naming Certificate for RealRootExistence$^{\uparrow}$ **15980**

1838 A Concrete Naming Certificate for BishopCompletionUniversalBoundary$^{\uparrow}$ **15983**

1839 A Concrete Naming Certificate for RealOscillation$^{\uparrow}$ **15987**

1840 A Concrete Naming Certificate for MeanErgodic$^{\uparrow}$ **15991**

1841 A Concrete Naming Certificate for RegularCauchyBall$^{\uparrow}$ **15995**

1842 Concrete Naming Certificate for PrecompactCompletion$^{\uparrow}$ **15998**

1843 Compact Baire NameCert Construction **16002**

1844 Concrete Naming Certificate for CauchyRealCompletionInterface$^{\uparrow}$ **16006**

1845 Concrete Naming Certificate for CauchyCompletionKleisli$^{\uparrow}$ **16010**

1846 Concrete Naming Certificate for CauchyPrecompletion$^{\uparrow}$ **16014**

1847 Concrete Naming Certificate for CauchySpace$^{\uparrow}$ **16018**

1848 Concrete Naming Certificate for UniformEntourageRegularName$^{\uparrow}$ **16027**

1849 Concrete Naming Certificate for RationalRealEmbedding$^{\uparrow}$ **16031**

1850 Concrete Naming Certificate for BishopLocatedCauchyReal$^{\uparrow}$ **16035**

1851 Concrete Naming Certificate for LocatedArchimedean$^{\uparrow}$ **16040**

1852 Concrete Naming Certificate for ApartnessContinuity$^{\uparrow}$ **16044**

1853 Concrete Naming Certificate for CoveringDimension$^{\uparrow}$ **16047**

1854 Concrete Naming Certificate for CauchySubsequenceExtraction$^{\uparrow}$ **16103**

1855 StreamMap NameCert Construction **16107**

1856 Concrete Naming Certificate for BishopIntervalNestLimit$^{\uparrow}$ **16115**

1857 Concrete Naming Certificate for CozeroSet$^{\uparrow}$ **16119**

1858 Concrete Naming Certificate for ZeroSet$^{\uparrow}$ **16122**

1859 Concrete Naming Certificate for LocatedUniformContinuity$^{\uparrow}$ **16125**

1860 Concrete Naming Certificate for SierpinskiSpace$^{\uparrow}$ **16129**

1861 Concrete Naming Certificate for CousinLemma$^{\uparrow}$ **16133**

1862 Concrete Naming Certificate for RegularCauchySubsequenceLimit$^{\uparrow}$ **16137**

1863 Concrete Naming Certificate for DyadicPartition$^{\uparrow}$ **16140**

1864 Concrete Naming Certificate for DyadicLipschitzModulus$^{\uparrow}$ **16143**

1865 Concrete Naming Certificate for SignedDyadicNormalForm$^{\uparrow}$ **16146**

1866 Concrete Naming Certificate for NormalSpace$^{\uparrow}$ **16150**

1867 Concrete Naming Certificate for ProductFilter$^{\uparrow}$ **16159**

186 A Concrete Naming Certificate for CauchyTailBasis$^{\uparrow}$ **16162**

186 A Concrete Naming Certificate for CauchyCompletionContinuation$^{\uparrow}$ **16165**

187 A Concrete Naming Certificate for CauchyTightness$^{\uparrow}$ **16169**

187 A Concrete Naming Certificate for DirichletApproximation$^{\uparrow}$ **16172**

187 Countable choice boundary NameCert construction **16176**

187 Cauchy Diagonal Argument NameCert Construction **16179**

187 A Concrete Naming Certificate for CauchyDifferenceCriterion$^{\uparrow}$ **16182**

187 A Concrete Naming Certificate for GramSchmidt$^{\uparrow}$ **16192**

187 A Concrete Naming Certificate for SeparatedUniformSpace$^{\uparrow}$ **16195**

187 A Concrete Naming Certificate for DyadicCoverRefinement$^{\uparrow}$ **16200**

187 A Concrete Naming Certificate for ExtendedRealLine$^{\uparrow}$ **16203**

187 A Concrete Naming Certificate for KroneckerLemma$^{\uparrow}$ **16206**

188 A Concrete Naming Certificate for DarbouxSum$^{\uparrow}$ **16209**

188 A Concrete Naming Certificate for CauchyRealApartness$^{\uparrow}$ **16212**

188 Rational Interval Cover NameCert Construction **16215**

188 A Concrete Naming Certificate for RoundedIdealCompletion$^{\uparrow}$ **16219**

188 A Concrete Naming Certificate for CompleteBaireSpace$^{\uparrow}$ **16223**

188 A Concrete Naming Certificate for LocatedIntervalCover$^{\uparrow}$ **16227**

188 A Concrete Naming Certificate for CantorFanBarrier$^{\uparrow}$ **16230**

188 A Concrete Naming Certificate for WeaklyCauchySequence$^{\uparrow}$ **16233**

188 A Concrete Naming Certificate for PicardLindelof$^{\uparrow}$ **16237**

189 A Concrete Naming Certificate for UniformCauchySequence$^{\uparrow}$ **16243**

189 A Concrete Naming Certificate for RealOrderTopology$^{\uparrow}$ **16246**

189 A Concrete Naming Certificate for CompactIntervalLocator$^{\uparrow}$ **16250**

189 A Concrete Naming Certificate for UniformCompletionFunctor$^{\uparrow}$ **16254**

189 A Concrete Naming Certificate for NatRealEmbedding$^{\uparrow}$ **16277**

1894 A Concrete Naming Certificate for RegularCauchyReverseTriangle$^{\uparrow}$ **16280**

1895 A Concrete Naming Certificate for CantorFunction$^{\uparrow}$ **16283**

1896 A Concrete Naming Certificate for CauchyBornology$^{\uparrow}$ **16287**

1897 A Concrete Naming Certificate for ContinuousIntervalImage$^{\uparrow}$ **16291**

1898 A Concrete Naming Certificate for CauchyDiagonalCompactness$^{\uparrow}$ **16295**

1899 A Concrete Naming Certificate for PicardLindelofLocalFlow$^{\uparrow}$ **16298**

1900 A Concrete Naming Certificate for RolleTheorem$^{\uparrow}$ **16301**

1901 A Concrete Naming Certificate for SecondCountable$^{\uparrow}$ **16304**

1902 A Concrete Naming Certificate for Rolle$^{\uparrow}$ **16307**

1903 A Concrete Naming Certificate for CompactOpenMetric$^{\uparrow}$ **16310**

1904 A Concrete Naming Certificate for RegularCauchyCancellation$^{\uparrow}$ **16313**

1905 A Concrete Naming Certificate for RealApartnessOrderBridge$^{\uparrow}$ **16316**

1906 A Concrete Naming Certificate for RegulatedFunction$^{\uparrow}$ **16319**

1907 A Concrete Naming Certificate for ConstructiveMeanValue$^{\uparrow}$ **16324**

1908 A Concrete Naming Certificate for CauchyRealLocator$^{\uparrow}$ **16327**

1909 A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$ **16331**

1910 A Concrete Naming Certificate for CauchySequenceAlgebra$^{\uparrow}$ **16336**

1911 A Concrete Naming Certificate for Cauchy$^{\uparrow}$ **16342**

1912 A Concrete Naming Certificate for MeanValueTheorem$^{\uparrow}$ **16352**

1913 A Concrete Naming Certificate for BolzanoCauchy$^{\uparrow}$ **16355**

1914 A Concrete Naming Certificate for BishopMonotoneRealLimit$^{\uparrow}$ **16360**

1915 A Concrete Naming Certificate for CauchyFilterBase$^{\uparrow}$ **16364**

1916 A Concrete Naming Certificate for DyadicIntervalBasis$^{\uparrow}$ **16367**

1917 A Concrete Naming Certificate for DoubleLimitInterchange$^{\uparrow}$ **16370**

1918 A Concrete Naming Certificate for RegularCauchyModulusRealizer$^{\uparrow}$ **16373**

1919 A Concrete Naming Certificate for StoneWeierstrass$^{\uparrow}$ **16379**

192 A Concrete Naming Certificate for CauchyApproximationSystem$^{\uparrow}$ **16384**

193 A Concrete Naming Certificate for CauchyCompletionRetraction$^{\uparrow}$ **16387**

194 A Concrete Naming Certificate for AbelLimitTheorem$^{\uparrow}$ **16391**

195 A Concrete Naming Certificate for SchauderFixedPoint$^{\uparrow}$ **16394**

196 A Concrete Naming Certificate for NestedIntervalReal$^{\uparrow}$ **16397**

197 A Concrete Naming Certificate for HausdorffSpace$^{\uparrow}$ **16403**

198 A Concrete Naming Certificate for RationalNestedIntervalLimit$^{\uparrow}$ **16410**

199 A Concrete Naming Certificate for RegularCauchyModulusStability$^{\uparrow}$ **16414**

200 A Concrete Naming Certificate for CauchyCompletionSplit$^{\uparrow}$ **16418**

201 A Concrete Naming Certificate for MonotoneInverse$^{\uparrow}$ **16422**

202 A Concrete Naming Certificate for RegularCauchyExtensionality$^{\uparrow}$ **16425**

203 A Concrete Naming Certificate for CauchyCompletionTrace$^{\uparrow}$ **16428**

204 A Concrete Naming Certificate for HardyLittlewoodTauberian$^{\uparrow}$ **16432**

205 Located Real NameCert Construction **16435**

206 A Concrete Naming Certificate for CauchyCompletionEmbedding$^{\uparrow}$ **16439**

207 A Concrete Naming Certificate for LocatedRegularization$^{\uparrow}$ **16442**

208 A Concrete Naming Certificate for GrzegorczykReal$^{\uparrow}$ **16445**

209 A Concrete Naming Certificate for CauchyCompletionFunctoriality$^{\uparrow}$ **16448**

210 A Concrete Naming Certificate for IntervalLength$^{\uparrow}$ **16452**

211 A Concrete Naming Certificate for FejerKernel$^{\uparrow}$ **16455**

212 A Concrete Naming Certificate for Parseval$^{\uparrow}$ **16458**

213 A Concrete Naming Certificate for McShaneExtension$^{\uparrow}$ **16475**

214 A Concrete Naming Certificate for RealDyadicEmbedding$^{\uparrow}$ **16480**

215 A Concrete Naming Certificate for RegularCauchyDivision$^{\uparrow}$ **16483**

216 A Concrete Naming Certificate for StreamMerge$^{\uparrow}$ **16486**

217 A Concrete Naming Certificate for DyadicIntervalDiameter$^{\uparrow}$ **16489**

1946 Concrete Naming Certificate for CauchyCompression↑ **16493**

1947 Concrete Naming Certificate for RegularCauchyMap↑ **16497**

1948 Concrete Naming Certificate for RealDyadicOrderEmbedding↑ **16501**

1949 Concrete Naming Certificate for ClosedSubspaceCompletion↑ **16504**

1950 Concrete Naming Certificate for NormedSpace↑ **16508**

1951 Concrete Naming Certificate for CauchyModulusPullback↑ **16523**

1952 Concrete Naming Certificate for CompactOperator↑ **16527**

1953 Concrete Naming Certificate for RapidCauchyName↑ **16533**

1954 Concrete Naming Certificate for CauchyCompletionComparison↑ **16537**

1955 Concrete Naming Certificate for RegularCauchyModulusIndependence↑ **16542**

1956 Concrete Naming Certificate for TietzeExtension↑ **16545**

1957 Concrete Naming Certificate for TernaryExpansion↑ **16548**

1958 Concrete Naming Certificate for CauchyFilterMeet↑ **16551**

1959 Concrete Naming Certificate for FiniteBranchingCauchyTree↑ **16555**

1960 Concrete Naming Certificate for RiemannIntegrable↑ **16559**

1961 Concrete Naming Certificate for PeriodicContinuedFraction↑ **16562**

1962 Concrete Naming Certificate for FanSpace↑ **16565**

1963 Concrete Naming Certificate for Paracompact↑ **16568**

1964 Concrete Naming Certificate for RegularSequenceLimit↑ **16575**

1965 Concrete Naming Certificate for RealConvergenceRate↑ **16579**

1966 Concrete Naming Certificate for BishopRealNormalizer↑ **16582**

1967 Concrete Naming Certificate for EulerPolygon↑ **16586**

1968 Concrete Naming Certificate for CauchyModulusNormalForm↑ **16591**

1969 Concrete Naming Certificate for RealIntervalArithmetic↑ **16594**

1970 Concrete Naming Certificate for BishopIntervalCompletion↑ **16597**

1971 Concrete Naming Certificate for RegularCauchyScalar↑ **16602**

1972 A Concrete Naming Certificate for RoundedDyadicLocator$^{\uparrow}$ **16605**

1973 A Concrete Naming Certificate for ArithmeticGeometricMean$^{\uparrow}$ **16608**

1974 A Concrete Naming Certificate for KroneckerApproximation$^{\uparrow}$ **16612**

1975 A Concrete Naming Certificate for LocatedCutApproximation$^{\uparrow}$ **16615**

1976 A Concrete Naming Certificate for SignedBinaryExpansion$^{\uparrow}$ **16619**

1977 A Concrete Naming Certificate for LesserLimitedOmniscience$^{\uparrow}$ **16622**

1978 A Concrete Naming Certificate for LLPOBoundary$^{\uparrow}$ **16625**

1979 A Concrete Naming Certificate for CompactCauchyCover$^{\uparrow}$ **16628**

1980 A Concrete Naming Certificate for CauchySequence$^{\uparrow}$ **16631**

1981 A Concrete Naming Certificate for CauchyCompletionAlgebra$^{\uparrow}$ **16636**

1981.1 StreamName Readback Route 16645

1982 A Concrete Naming Certificate for CauchyFilterModulusTransfer$^{\uparrow}$ **16647**

1983 A Concrete Naming Certificate for IrrationalWitness$^{\uparrow}$ **16652**

1984 A Concrete Naming Certificate for LocatedRealSequence$^{\uparrow}$ **16656**

1985 A Concrete Naming Certificate for DenseImageCompletion$^{\uparrow}$ **16660**

1986 A Concrete Naming Certificate for CauchyModulusSelection$^{\uparrow}$ **16664**

1987 A Concrete Naming Certificate for FiniteEpsilonWitness$^{\uparrow}$ **16668**

1988 A Concrete Naming Certificate for RealCompletionFiniteRoute$^{\uparrow}$ **16671**

1989 A Concrete Naming Certificate for RealCauchyFilter$^{\uparrow}$ **16675**

1990 A Concrete Naming Certificate for RealCauchySum$^{\uparrow}$ **16678**

1991 A Concrete Naming Certificate for TauberianRemainder$^{\uparrow}$ **16682**

1992 A Concrete Naming Certificate for CauchyRealOrderedField$^{\uparrow}$ **16685**

1993 A Concrete Naming Certificate for RiemannIntegrabilityCriterion$^{\uparrow}$ **16689**

1994 A Concrete Naming Certificate for CauchySpeedup$^{\uparrow}$ **16692**

1995 A Concrete Naming Certificate for HilbertCube$^{\uparrow}$ **16695**

1996 A Concrete Naming Certificate for EquicontinuousCompletion$^{\uparrow}$ **16699**

1997 A Concrete Naming Certificate for MonotoneSubsequenceTheorem$^{\uparrow}$ **16702**

198 A Concrete Naming Certificate for RegularCauchyCofinalSubsequence↑ **16705**

199 A Concrete Naming Certificate for GaugeIntegral↑ **16708**

200 A Concrete Naming Certificate for ProductTopology↑ **16711**

201 A Concrete Naming Certificate for DyadicMetric↑ **16714**

202 A Concrete Naming Certificate for CauchyCompletionEliminator↑ **16717**

203 A Concrete Naming Certificate for SpernerLemma↑ **16720**

204 A Concrete Naming Certificate for SeparatedQuotientMetric↑ **16723**

205 A Concrete Naming Certificate for CompleteMetricProduct↑ **16726**

206 A Concrete Naming Certificate for UniformIntegralLimit↑ **16730**

207 A Concrete Naming Certificate for EngelExpansion↑ **16733**

208 A Concrete Naming Certificate for GeometricSeries↑ **16736**

209 A Concrete Naming Certificate for RegularCauchyFiniteProduct↑ **16739**

210 Dedekind Cauchy Sandwich NameCert Construction **16749**

211 A Concrete Naming Certificate for CutRegularization↑ **16752**

212 A Concrete Naming Certificate for FormalBallCompletion↑ **16756**

213 A Concrete Naming Certificate for FeketeSubadditive↑ **16760**

214 A Concrete Naming Certificate for RealSeriesCauchyCriterion↑ **16764**

215 A Concrete Naming Certificate for CauchySeriesCriterion↑ **16767**

216 A Concrete Naming Certificate for RoundedCauchyFilter↑ **16771**

217 A Concrete Naming Certificate for DyadicShrinkSchedule↑ **16775**

218 A Concrete Naming Certificate for DyadicBinarySearch↑ **16779**

219 A Concrete Naming Certificate for DyadicMetricTriangle↑ **16782**

220 Dyadic Fast Cauchy NameCert Construction **16785**

221 A Concrete Naming Certificate for ClosedIntervalTotalBounded↑ **16788**

222 Extension by Continuity NameCert Construction **16792**

223 A Concrete Naming Certificate for SeriesLimitComparison↑ **16796**

2024 Concrete Naming Certificate for CantorBendixson$^{\uparrow}$ **16800**

2025 Concrete Naming Certificate for CompleteArchimedeanField$^{\uparrow}$ **16803**

2026 Concrete Naming Certificate for EffectiveCompletion$^{\uparrow}$ **16806**

2027 Concrete Naming Certificate for FanContinuityPrinciple$^{\uparrow}$ **16810**

2028 Concrete Naming Certificate for SelectionFreeCauchyReal$^{\uparrow}$ **16814**

2029 Concrete Naming Certificate for LocatedPositiveReal$^{\uparrow}$ **16818**

2030 Concrete Naming Certificate for BrouwerFixedPoint$^{\uparrow}$ **16822**

2031 Concrete Naming Certificate for FiniteChoice$^{\uparrow}$ **16826**

2032 Concrete Naming Certificate for CauchyUniformContinuityPrinciple$^{\uparrow}$ **16830**

2033 Concrete Naming Certificate for UniformPartition$^{\uparrow}$ **16834**

2034 Concrete Naming Certificate for GronwallInequality$^{\uparrow}$ **16838**

2035 Concrete Naming Certificate for DoublingMetric$^{\uparrow}$ **16842**

2036 Concrete Naming Certificate for CauchyCompletionIsometry$^{\uparrow}$ **16845**

2037 Concrete Naming Certificate for CauchyCompletionLift$^{\uparrow}$ **16849**

2038 Concrete Naming Certificate for ArchimedeanBracket$^{\uparrow}$ **16853**

2039 Concrete Naming Certificate for RatioCauchyTransfer$^{\uparrow}$ **16856**

2040 Concrete Naming Certificate for CauchyModulusChoice$^{\uparrow}$ **16860**

2041 Concrete Naming Certificate for DiniDerivative$^{\uparrow}$ **16864**

2042 Concrete Naming Certificate for HyersUlamStability$^{\uparrow}$ **16868**

2043 Concrete Naming Certificate for LocatedSpace$^{\uparrow}$ **16872**

2044 Concrete Naming Certificate for CauchyCompletionOrderEmbedding$^{\uparrow}$ **16878**

2045 Concrete Naming Certificate for AlexandroffCompactification$^{\uparrow}$ **16881**

2046 Concrete Naming Certificate for AlexandrovCompactification$^{\uparrow}$ **16885**

2047 Concrete Naming Certificate for SeriesDivergenceTest$^{\uparrow}$ **16888**

2048 Concrete Naming Certificate for TotalBoundedMetric$^{\uparrow}$ **16891**

2049 Concrete Naming Certificate for OnePointCompactification$^{\uparrow}$ **16896**

2050 Concrete Naming Certificate for ChainConnectedMetric↑ **16899**

2051 Concrete Naming Certificate for CauchyCompletenessModulus↑ **16902**

2052 Concrete Naming Certificate for ArchimedeanModulus↑ **16905**

2053 Concrete Naming Certificate for FirstCountableSpace↑ **16908**

2054 Concrete Naming Certificate for BishopCauchyNormalization↑ **16912**

2055 Concrete Naming Certificate for InverseFunction↑ **16916**

2056 Concrete Naming Certificate for ConstructiveCompletionModulus↑ **16920**

2057 Concrete Naming Certificate for NestedIntervalConvergence↑ **16923**

2058 Concrete Naming Certificate for DecimalReal↑ **16928**

2059 Concrete Naming Certificate for EulerMaclaurin↑ **16931**

2060 Concrete Naming Certificate for RealUniformEntourageBase↑ **16935**

2061 Concrete Naming Certificate for PositiveRealRadiusScale↑ **16938**

2062 Concrete Naming Certificate for BishopRealLineInterface↑ **16941**

2063 Concrete Naming Certificate for LocatedRealField↑ **16950**

2064 Concrete Naming Certificate for RealUniformEntourageRefinement↑ **16953**

2065 Concrete Naming Certificate for KuratowskiCompletion↑ **16957**

2066 Concrete Naming Certificate for CauchyCompletionKleisliAdjunction↑ **16960**

2067 Concrete Naming Certificate for LocatedRealCauchyCriterion↑ **16964**

2068 Concrete Naming Certificate for CauchyConvergenceModulusTailMonotonicity↑ **16967**

2069 Concrete Naming Certificate for JordanVariationDecomposition↑ **16971**

2070 Concrete Naming Certificate for BoundedVariationCompletion↑ **16974**

2071 Concrete Naming Certificate for CauchyTailBound↑ **16977**

2072 Concrete Naming Certificate for RegularCauchySpread↑ **16980**

2073 Concrete Naming Certificate for RealCompletionFunctor↑ **16983**

2074 Concrete Naming Certificate for RealModulusOfConvergence↑ **16987**

2075 Concrete Naming Certificate for SignedDigitCauchyModulus↑ **16999**

207A Concrete Naming Certificate for RegularCauchyProductProjection↑ **17003**

207B Concrete Naming Certificate for FastConvergentRealSequence↑ **17007**

207C Concrete Naming Certificate for PeanoCurve↑ **17010**

207D Concrete Naming Certificate for SturmSequence↑ **17014**

208E Concrete Naming Certificate for PringsheimPositiveRealSeries↑ **17017**

208F Concrete Naming Certificate for BishopIntervalHalving↑ **17021**

208G Concrete Naming Certificate for LengthSpace↑ **17025**

208H Concrete Naming Certificate for ScottReal↑ **17028**

208I Concrete Naming Certificate for RegularCauchyTailSubsequence↑ **17031**

208J Concrete Naming Certificate for RegularCauchyRateExtraction↑ **17034**

208K Concrete Naming Certificate for HausdorffMetric↑ **17038**

208L Concrete Naming Certificate for LowerUpperRealCut↑ **17041**

208M Located choice sequence NameCert construction **17044**

208N Concrete Naming Certificate for DyadicLocatedCut↑ **17047**

209O Concrete Naming Certificate for BishopCauchyLimitCriterion↑ **17050**

209P Concrete Naming Certificate for RegularCauchyReindexInvariance↑ **17054**

209Q Concrete Naming Certificate for KonigLemma↑ **17058**

209R Concrete Naming Certificate for CauchyMetastability↑ **17061**

209S Concrete Naming Certificate for TanneryTheorem↑ **17065**

209T Concrete Naming Certificate for SmythCompletion↑ **17069**

209U Concrete Naming Certificate for RealCauchyModulusRefinement↑ **17073**

209V Concrete Naming Certificate for RegularCauchyCategory↑ **17077**

209W Concrete Naming Certificate for Goursat↑ **17081**

209X Concrete Naming Certificate for EudoxusReal↑ **17085**

210Y Concrete Naming Certificate for HolderInequality↑ **17088**

210Z Concrete Naming Certificate for CompactMetricSubspace↑ **17092**

2102 Concrete Naming Certificate for CauchyCompletionComparisonNaturality$^{\uparrow}$ **17096**

2103 Concrete Naming Certificate for RegularCauchyLipschitzMap$^{\uparrow}$ **17100**

2104 Concrete Naming Certificate for GromovHausdorff$^{\uparrow}$ **17104**

2105 Concrete Naming Certificate for RegularCauchyNonexpansiveMap$^{\uparrow}$ **17108**

2106 Concrete Naming Certificate for DyadicValuation$^{\uparrow}$ **17111**

2107 Concrete Naming Certificate for Subshift$^{\uparrow}$ **17115**

2108 Concrete Naming Certificate for BisectionMethod$^{\uparrow}$ **17118**

2109 Concrete Naming Certificate for AlgebraicReal$^{\uparrow}$ **17121**

2110 Concrete Naming Certificate for BoundedNestedRealChoiceSeal$^{\uparrow}$ **17124**

2111 Concrete Naming Certificate for HurwitzApproximation$^{\uparrow}$ **17128**

2112 Concrete Naming Certificate for CauchyRealLocatedOrderTest$^{\uparrow}$ **17132**

2113 Concrete Naming Certificate for KummerTest$^{\uparrow}$ **17135**

2114 Concrete Naming Certificate for LocatedMetricCompletion$^{\uparrow}$ **17139**

2115 Concrete Naming Certificate for KuratowskiCompletionNaturality$^{\uparrow}$ **17144**

2116 Concrete Naming Certificate for SignedDistance$^{\uparrow}$ **17148**

2117 Cauchy compact NameCert construction **17152**

2118 Concrete Naming Certificate for RegularCauchyLocatedComparison$^{\uparrow}$ **17155**

2119 Concrete Naming Certificate for DyadicDivision$^{\uparrow}$ **17159**

2120 Concrete Naming Certificate for RegularCauchyAverage$^{\uparrow}$ **17163**

2121 Concrete Naming Certificate for DugundjiExtension$^{\uparrow}$ **17167**

2122 Concrete Naming Certificate for RealIntervalDistance$^{\uparrow}$ **17171**

2123 Concrete Naming Certificate for CauchyOscillationCriterion$^{\uparrow}$ **17174**

2124 Concrete Naming Certificate for MetricCompletionUniversalProperty$^{\uparrow}$ **17177**

2125 Concrete Naming Certificate for AbelPartialSummation$^{\uparrow}$ **17181**

2126 Concrete Naming Certificate for SilvermanToeplitz$^{\uparrow}$ **17184**

2127 Concrete Naming Certificate for SeriesTermTest$^{\uparrow}$ **17188**

212.8 Concrete Naming Certificate for NormalConvergence$^{\uparrow}$ **17191**

212.9 Concrete Naming Certificate for RegularCauchyCommonTailFilter$^{\uparrow}$ **17194**

213.0 Concrete Naming Certificate for GaussTest$^{\uparrow}$ **17197**

213.1 Concrete Naming Certificate for SamuelCompletion$^{\uparrow}$ **17200**

213.2 Concrete Naming Certificate for RiemannRearrangement$^{\uparrow}$ **17203**

213.3 Concrete Naming Certificate for OpenIntervalReal$^{\uparrow}$ **17206**

213.4 Concrete Naming Certificate for MonoidalCompletion$^{\uparrow}$ **17209**

213.5 Concrete Naming Certificate for SmythPowerdomain$^{\uparrow}$ **17213**

213.6 Concrete Naming Certificate for RegularCauchyCompletionMonad$^{\uparrow}$ **17217**

213.7 Regular Cauchy Tail Sum NameCert Construction **17221**

213.8 Concrete Naming Certificate for DedekindCutCompletion$^{\uparrow}$ **17224**

213.9 Concrete Naming Certificate for EilenbergMooreCompletionAlgebra$^{\uparrow}$ **17228**

214.0 Concrete Naming Certificate for BishopIntervalEnclosure$^{\uparrow}$ **17233**

214.1 Concrete Naming Certificate for CofinalTailMap$^{\uparrow}$ **17236**

214.2 Concrete Naming Certificate for ArchimedeanCauchyTail$^{\uparrow}$ **17239**

214.3 Concrete Naming Certificate for SternBrocotInterval$^{\uparrow}$ **17243**

214.4 Concrete Naming Certificate for PrincipalFilter$^{\uparrow}$ **17247**

214.5 Concrete Naming Certificate for KuratowskiClosure$^{\uparrow}$ **17250**

214.6 Concrete Naming Certificate for RegularCauchyTailComparison$^{\uparrow}$ **17254**

214.7 Concrete Naming Certificate for CauchyTailIntersection$^{\uparrow}$ **17258**

214.8 Concrete Naming Certificate for YonedaCompletion$^{\uparrow}$ **17262**

214.9 Concrete Naming Certificate for MetricCompletionReflector$^{\uparrow}$ **17266**

215.0 Concrete Naming Certificate for VietorisRipsComplex$^{\uparrow}$ **17269**

215.1 Concrete Naming Certificate for RegularCauchyTranslation$^{\uparrow}$ **17272**

215.2 Concrete Naming Certificate for RoundedIntervalDomain$^{\uparrow}$ **17275**

215.3 Concrete Naming Certificate for CantorCylinderBasis$^{\uparrow}$ **17278**

2154 Concrete Naming Certificate for SamuelCompactification↑ **17281**

2155 Concrete Naming Certificate for UniformConvergenceSeries↑ **17285**

2156 Concrete Naming Certificate for MetricCompletionMonad↑ **17289**

2157 Concrete Naming Certificate for FirstCountableSequentialClosure↑ **17292**

2158 Concrete Naming Certificate for QuasiMetric↑ **17296**

2159 Concrete Naming Certificate for LeibnizAlternatingRemainder↑ **17303**

2160 Concrete Naming Certificate for PeanoContinuum↑ **17307**

2161 Concrete Naming Certificate for CodonWindowSpectralInverse↑ **17311**

2162 Concrete Naming Certificate for RealApproximationScheme↑ **17315**

2163 Concrete Naming Certificate for SqueezeTheorem↑ **17318**

2164 Concrete Naming Certificate for FrechetCompletion↑ **17323**

2165 Concrete Naming Certificate for EmptyBoundarySelectorLedger↑ **17326**

2166 Concrete Naming Certificate for FiniteVitaliCover↑ **17329**

2167 Concrete Naming Certificate for StarRefinementUniformCover↑ **17332**

2168 Concrete Naming Certificate for InverseLimitMetric↑ **17336**

2169 Concrete Naming Certificate for RegularCauchyQuotientRefusal↑ **17339**

2170 Concrete Naming Certificate for SeparatedCompletionReflector↑ **17342**

2171 Concrete Naming Certificate for LocalTimeFiber↑ **17346**

2172 Concrete Naming Certificate for CauchyInterleavingProduct↑ **17349**

2173 Concrete Naming Certificate for DenseExtensionMetric↑ **17352**

2174 Concrete Naming Certificate for CauchyApproximationScheme↑ **17356**

2175 Concrete Naming Certificate for DyadicCompactness↑ **17359**

2176 Concrete Naming Certificate for LocatedBoundedRealSet↑ **17362**

2177 Concrete Naming Certificate for ViscositySolution↑ **17366**

2178 Concrete Naming Certificate for PreReal↑ **17370**

2179 Concrete Naming Certificate for BishopSequence↑ **17373**

2180 Concrete Naming Certificate for CauchyRealEquality↑ **17376**

2181 Concrete Naming Certificate for MetaCICResidualSubstitutionBoundary↑ **17379**

2182 Concrete Naming Certificate for ClosedTypeCheckerSeal↑ **17383**

2183 Concrete Naming Certificate for LocatedBoundedRealSequence↑ **17387**

2184 Concrete Naming Certificate for VitaliCover↑ **17391**

2185 Concrete Naming Certificate for FastCauchySequence↑ **17395**

2186 Concrete Naming Certificate for NormalFamily↑ **17398**

2187 Concrete Naming Certificate for MetricCoveringNumber↑ **17401**

2188 Concrete Naming Certificate for CompactUniformContinuityScheme↑ **17404**

2189 Concrete Naming Certificate for BanachAlgebra↑ **17407**

2190 Concrete Naming Certificate for MooreSmithCompletion↑ **17412**

2191 Concrete Naming Certificate for OpenBall↑ **17416**

2192 Concrete Naming Certificate for MetricCompletionFunctor↑ **17419**

2193 Concrete Naming Certificate for StronglyExtensionalRealMap↑ **17423**

2194 Concrete Naming Certificate for EulerTransform↑ **17426**

2195 Concrete Naming Certificate for BishopCompactnessCriterion↑ **17430**

2196 Concrete Naming Certificate for DiniContinuity↑ **17434**

2197 Concrete Naming Certificate for GroebnerBasis↑ **17439**

2198 Concrete Naming Certificate for CauchyNetHausdorffUniqueness↑ **17444**

2199 Concrete Naming Certificate for CantorCompleteMetric↑ **17447**

2200 Concrete Naming Certificate for UniformLimitTheorem↑ **17450**

2201 Concrete Naming Certificate for RealArchimedeanBoundSchema↑ **17453**

2202 Concrete Naming Certificate for CompactUniformModulusExtractor↑ **17456**

2203 Concrete Naming Certificate for DyadicIntermediateValue↑ **17459**

2204 Concrete Naming Certificate for Egorov↑ **17463**

2205 Concrete Naming Certificate for UniformEntourageSpace↑ **17466**

2206 A Concrete Naming Certificate for RealPrefixDiagnostic↑ **17469**

2207 A Concrete Naming Certificate for BesselInequality↑ **17472**

2208 A Concrete Naming Certificate for RegularFilterRealization↑ **17475**

2209 A Concrete Naming Certificate for SchurMatrixSummability↑ **17478**

2210 A Concrete Naming Certificate for WeakDerivative↑ **17481**

2211 A Concrete Naming Certificate for UniformCompleteSpace↑ **17484**

2212 A Concrete Naming Certificate for MetaCICConfluenceCriticalPairBoundary↑ **17487**

2213 A Concrete Naming Certificate for GeodesicMetric↑ **17493**

2214 A Concrete Naming Certificate for CauchySeriesProduct↑ **17496**

2215 A Concrete Naming Certificate for RegulatedUniformApproximation↑ **17499**

2216 A Concrete Naming Certificate for CauchyLocatedness↑ **17503**

2217 A Concrete Naming Certificate for SobolevEmbedding↑ **17506**

2218 A Concrete Naming Certificate for RellichKondrachov↑ **17509**

2219 A Concrete Naming Certificate for AscoliDiagonal↑ **17512**

2220 A Concrete Naming Certificate for CauchyTailControl↑ **17515**

2221 A Concrete Naming Certificate for CompactOpenFunctionSpace↑ **17518**

2222 A Concrete Naming Certificate for CauchyRegularityModulus↑ **17522**

2223 A Concrete Naming Certificate for FiniteObservationConfluence↑ **17525**

2224 A Concrete Naming Certificate for OscillationModulus↑ **17528**

2225 A Concrete Naming Certificate for SeparableMetricSpace↑ **17531**

2226 A Concrete Naming Certificate for MetaCICNormalAuditSocket↑ **17534**

2227 A Concrete Naming Certificate for RegularCauchyTightening↑ **17538**

2228 A Concrete Naming Certificate for UniformContinuityModulus↑ **17541**

2229 A Concrete Naming Certificate for ClosedIntervalApproximation↑ **17544**

2230 A Concrete Naming Certificate for AbelDirichletComparison↑ **17548**

2231 A Concrete Naming Certificate for BishopLocatedCompletion↑ **17551**

2232 Concrete Naming Certificate for MetricCofinalCompletion$^{\uparrow}$ **17554**

2233 Concrete Naming Certificate for PicardErrorEstimate$^{\uparrow}$ **17557**

2234 Concrete Naming Certificate for CodonSixBitWindow$^{\uparrow}$ **17560**

2235 Concrete Naming Certificate for JensenInequality$^{\uparrow}$ **17563**

2236 Concrete Naming Certificate for FiniteObservationFold$^{\uparrow}$ **17566**

2237 Concrete Naming Certificate for SeparatedPseudometricCompletion$^{\uparrow}$ **17569**

2238 Concrete Naming Certificate for CauchyProductCommutativity$^{\uparrow}$ **17573**

2239 Concrete Naming Certificate for CauchyProductAssociativity$^{\uparrow}$ **17576**

2240 Concrete Naming Certificate for FiniteWindowPullback$^{\uparrow}$ **17579**

2241 Concrete Naming Certificate for SequentialUniformCompletion$^{\uparrow}$ **17582**

2242 Concrete Naming Certificate for HausdorffCompletionUniversal$^{\uparrow}$ **17585**

2243 Concrete Naming Certificate for RegularCauchyFilterRefinement$^{\uparrow}$ **17588**

2244 Concrete Naming Certificate for RealSeriesRatioTest$^{\uparrow}$ **17591**

2245 Concrete Naming Certificate for CodonTransitionMatrix$^{\uparrow}$ **17594**

2246 Concrete Naming Certificate for MartingaleConvergence$^{\uparrow}$ **17597**
2246.1 Finite Convergence Carrier 17597
2246.2 Certificate Obligations 17598

2247 Concrete Naming Certificate for ObservationResidualFit$^{\uparrow}$ **17602**
2247.1 Residual Carrier 17602
2247.2 Residual Obligations 17603

2248 Concrete Naming Certificate for LocatedRealIntervalPartition$^{\uparrow}$ **17605**

2249 Concrete Naming Certificate for NestedIntervalSelection$^{\uparrow}$ **17608**

2250 Concrete Naming Certificate for DenseCauchyEmbedding$^{\uparrow}$ **17611**

2251 Concrete Naming Certificate for RegularCauchyCompletionUniversalProperty$^{\uparrow}$ **17614**

2252 Concrete Naming Certificate for LandauSymbol$^{\uparrow}$ **17618**

2253 Concrete Naming Certificate for DyadicCompactnessExtractor$^{\uparrow}$ **17620**

2254 Concrete Naming Certificate for CofinalDyadicMesh$^{\uparrow}$ **17623**

2255 Concrete Naming Certificate for RealSequenceCauchyCriterion$^{\uparrow}$ **17625**

2256 A Concrete Naming Certificate for CompletionUnitTriangle↑ **17628**

2257 A Concrete Naming Certificate for TotalVariationFunction↑ **17631**

2258 A Concrete Naming Certificate for GroundCompilerAuditRoute↑ **17634**

2259 A Concrete Naming Certificate for CauchyIntegralTest↑ **17637**

2260 A Concrete Naming Certificate for DNAFourPhaseAlphabetBridge↑ **17641**

2261 A Concrete Naming Certificate for UniformSequentialContinuity↑ **17644**

2262 A Concrete Naming Certificate for SmithNormalForm↑ **17647**

2263 A Concrete Naming Certificate for RadonMeasure↑ **17651**
2263.1 Radon Measure Carrier 17651
2263.2 Regularity Obligations 17652

2264 A Concrete Naming Certificate for OrthogonalPolynomial↑ **17654**

2265 A Concrete Naming Certificate for BanachDual↑ **17657**

2266 A Concrete Naming Certificate for IntervalBisectionTree↑ **17659**

2267 A Concrete Naming Certificate for MeagreSet↑ **17662**

2268 A Concrete Naming Certificate for DirichletUniformConvergence↑ **17665**

2269 A Concrete Naming Certificate for RealRootIsolation↑ **17668**

2270 A Concrete Naming Certificate for SpectralStabilization↑ **17671**

2271 A Concrete Naming Certificate for TotallyDisconnectedSpace↑ **17674**

2272 A Concrete Naming Certificate for MetaCICFrontierDependencyMatrix↑ **17677**

2273 A Concrete Naming Certificate for LiouvilleNumber↑ **17680**

2274 A Concrete Naming Certificate for CandidateSNConfluenceBoundary↑ **17683**

2275 A Concrete Naming Certificate for CoveringSpacePathLifting↑ **17687**

2276 A Concrete Naming Certificate for CoveringHomotopyLifting↑ **17690**

2277 A Concrete Naming Certificate for NestedIntervalTheorem↑ **17693**

2278 A Concrete Naming Certificate for HellyTheorem↑ **17696**
2278.1 Finite Convex Intersection Carrier 17697
2278.2 Helly NameCert Surface 17697

2279 A Concrete Naming Certificate for TemperedDistribution↑ 17707
2279.1 Carrier definition 17707
2279.2 The certificate 17708

2280 A Concrete Naming Certificate for DirichletKernel↑ 17711

2281 A Concrete Naming Certificate for BishopCauchyReal↑ 17714

2282 A Concrete Naming Certificate for PointwiseLimit↑ 17718

2283 A Concrete Naming Certificate for RealLogarithm↑ 17721

2284 A Concrete Naming Certificate for LocatedCauchyCompletionFunctor↑ 17724

2285 A Concrete Naming Certificate for WobbleQuotientSpectralTrigger↑ 17727

2286 A Concrete Naming Certificate for CantorIntersectionTheorem↑ 17730

2287 A Concrete Naming Certificate for CauchyCompletionUniversalProperty↑ 17734

2288 A Concrete Naming Certificate for CandidateSetNormalization↑ 17737

2289 A Concrete Naming Certificate for CompletionComparison↑ 17741

2290 A Concrete Naming Certificate for CofinalCompactInterval↑ 17744

2291 A Concrete Naming Certificate for InverseMappingTheorem↑ 17747

2292 A Concrete Naming Certificate for CompactUniformContinuityModulus↑ 17750

2293 A Concrete Naming Certificate for PoissonKernel↑ 17753

2294 A Concrete Naming Certificate for ConstructiveGreenFunction↑ 17756

2295 A Concrete Naming Certificate for FiniteModulusDiagonal↑ 17759

2296 A Concrete Naming Certificate for MedianLeakageSpectrum↑ 17762

2297 A Concrete Naming Certificate for PositiveCone↑ 17765
2297.1 Carrier and cone packet 17765
2297.2 NameCert obligations 17766

2298 A Concrete Naming Certificate for ContextFreePumpingLemma↑ 17768

2299 A Concrete Naming Certificate for CompactUniformModulus↑ 17771

2300 A Concrete Naming Certificate for MichaelSelection↑ 17776

2301 A Concrete Naming Certificate for LebesgueNumberLemma↑ 17779

2302 A Concrete Naming Certificate for UniformBoundednessPrinciple↑ 17783

2303 A Concrete Naming Certificate for MetrizableCompletion↑ **17787**

2304 A Concrete Naming Certificate for EffectiveMetricCompletion↑ **17790**

2305 A Concrete Naming Certificate for CompletionTower↑ **17793**

2306 A Concrete Naming Certificate for CoreStateGraph↑ **17795**

2307 A Concrete Naming Certificate for MetaCICFrontierDependencyMatrix↑ **17798**

2308 A Concrete Naming Certificate for NestedIntervalTheorem↑ **17801**

2309 A Concrete Naming Certificate for HellyTheorem↑ **17804**
2309.1 Finite Convex Intersection Carrier 17805
2309.2 Helly NameCert Surface 17805

2310 A Concrete Naming Certificate for TemperedDistribution↑ **17815**
2310.1 Carrier definition 17815
2310.2 The certificate 17816

2311 A Concrete Naming Certificate for DirichletKernel↑ **17819**

2312 A Concrete Naming Certificate for CauchyCut↑ **17822**

2313 A Concrete Naming Certificate for BishopCauchyReal↑ **17825**

2314 A Concrete Naming Certificate for PointwiseLimit↑ **17829**

2315 A Concrete Naming Certificate for RealLogarithm↑ **17832**

2316 A Concrete Naming Certificate for LocatedCauchyCompletionFunctor↑ **17835**

2317 A Concrete Naming Certificate for WobbleQuotientSpectralTrigger↑ **17838**

2318 A Concrete Naming Certificate for CantorIntersectionTheorem↑ **17841**

2319 A Concrete Naming Certificate for CauchyCompletionUniversalProperty↑ **17845**

2320 A Concrete Naming Certificate for CandidateSetNormalization↑ **17848**

2321 A Concrete Naming Certificate for CompletionComparison↑ **17852**

2322 A Concrete Naming Certificate for CofinalCompactInterval↑ **17855**

2323 A Concrete Naming Certificate for MetrizableCompletion↑ **17858**

2324 A Concrete Naming Certificate for NestedIntervalDiameterZero↑ **17861**

2325 A Concrete Naming Certificate for ClosedParallelDiamondPremiseLedger↑ **17864**

2326 A Concrete Naming Certificate for UniformConvergenceSpace↑ **17868**

2327 Concrete Naming Certificate for LocatedRealIntervalHull↑ **17871**

2328 Concrete Naming Certificate for LipschitzConstant↑ **17874**

2329 Concrete Naming Certificate for PoincareInequality↑ **17878**

2330 Concrete Naming Certificate for TypedExampleReductionTrace↑ **17880**

2331 Concrete Naming Certificate for UniformEntourageBasis↑ **17882**

2332 Concrete Naming Certificate for UniformCompletionMonad↑ **17885**

2333 Concrete Naming Certificate for DirichletSeriesTest↑ **17888**

2334 Concrete Naming Certificate for QuasiUniformSpace↑ **17891**

2335 Concrete Naming Certificate for VolterraIntegralOperator↑ **17899**

2336 Concrete Naming Certificate for NeighborhoodFunction↑ **17903**

2337 Concrete Naming Certificate for MetaCICNormalizationWindowConfluence↑ **17906**

2338 Concrete Naming Certificate for SturmComparison↑ **17909**

2339 Concrete Naming Certificate for MaterialityOrbitEscape↑ **17912**

2340 Concrete Naming Certificate for DominatedConvergence↑ **17915**

2341 Concrete Naming Certificate for CauchyNetRegularization↑ **17921**

2342 Concrete Naming Certificate for ChoiceFreeRegularCauchyModulus↑ **17924**

2343 Concrete Naming Certificate for MinkowskiInequality↑ **17927**

2344 Concrete Naming Certificate for StoneCechCompactification↑ **17930**

2345 Concrete Naming Certificate for UniformCompletionSeparationKernel↑ **17933**

2346 Concrete Naming Certificate for CauchyDedekindBridge↑ **17936**

2347 Concrete Naming Certificate for CaratheodoryConvexHull↑ **17939**

2348 Concrete Naming Certificate for RealApartnessSign↑ **17941**

2349 Concrete Naming Certificate for ArensFortSpace↑ **17944**

2350 Concrete Naming Certificate for CauchyContractionOrbit↑ **17947**

2351 Concrete Naming Certificate for RegulatedCauchyIntegral↑ **17950**

2352 Concrete Naming Certificate for SpectralRadius↑ **17953**

2353 Concrete Naming Certificate for ConnectedSpace↑ **17956**

2354 Concrete Naming Certificate for PointwiseContinuityModulus↑ **17960**

2355 Concrete Naming Certificate for CompleteSeparatedMetric↑ **17963**

2356 Concrete Naming Certificate for MetricProjection↑ **17966**

2357 Concrete Naming Certificate for BorelCantelli↑ **17971**

2358 Concrete Naming Certificate for YoungInequality↑ **17974**

2359 Concrete Naming Certificate for PicardIteration↑ **17977**

2360 Concrete Naming Certificate for SubstrateTraceInvariant↑ **17982**

2361 Concrete Naming Certificate for IsbellTightSpan↑ **17986**

2362 Concrete Naming Certificate for FiniteNetEquicontinuity↑ **17989**

2363 Concrete Naming Certificate for LaguerrePolya↑ **17993**

2364 Concrete Naming Certificate for FiniteAutomaton↑ **17997**

2365 Concrete Naming Certificate for SorgenfreyPlane↑ **18000**

2366 Concrete Naming Certificate for RegularCauchyCantorIntersection↑ **18005**

2367 Concrete Naming Certificate for HausdorffUniformCompletion↑ **18008**

2368 Concrete Naming Certificate for WrrStatePhaseSpace↑ **18011**

2369 Concrete Naming Certificate for SigmaCompact↑ **18014**

2370 Concrete Naming Certificate for BlaschkeSelectionCompactHyperspace↑ **18017**

2371 Concrete Naming Certificate for CellularSubstrateObservationBoundary↑ **18020**

2372 Concrete Naming Certificate for SequentialCompact↑ **18023**

2373 Concrete Naming Certificate for SeparatedMetricCompletion↑ **18043**

2374 Concrete Naming Certificate for OrthogonalProjectionTheorem↑ **18046**

2375 Concrete Naming Certificate for VietorisTopology↑ **18049**

2376 Concrete Naming Certificate for LocatedRegSeqRat↑ **18051**

2377 Concrete Naming Certificate for FiniteRealCover↑ **18057**

2378 Concrete Naming Certificate for RealNullSequence↑ **18060**

2379 A Concrete Naming Certificate for FanModulusBoundary↑ **18062**

2380 A Concrete Naming Certificate for PhysicalInduction↑ **18064**

2381 A Concrete Naming Certificate for CompletionIdempotence↑ **18067**

2382 A Concrete Naming Certificate for MetaCICFrontierCriticalPair↑ **18070**

2383 A Concrete Naming Certificate for FixedPointIteration↑ **18073**

2384 A Concrete Naming Certificate for RaikovCompletion↑ **18076**

2385 A Concrete Naming Certificate for HeineBorelLocatedInterval↑ **18079**

2386 A Concrete Naming Certificate for DyadicCauchyModulusLattice↑ **18082**

2387 A Concrete Naming Certificate for BishopLocatedLimit↑ **18086**

2388 A Concrete Naming Certificate for MetricEmbedding↑ **18090**

2389 A Concrete Naming Certificate for AffineContraction↑ **18093**

2390 A Concrete Naming Certificate for MetricClosure↑ **18096**

2391 A Concrete Naming Certificate for CauchyCriterionEquivalence↑ **18099**

2392 A Concrete Naming Certificate for MetaCICClosurePreservationWitness↑ **18102**

2393 A Concrete Naming Certificate for CauchyCondensationRealSeries↑ **18105**

2394 A Concrete Naming Certificate for VariationOfConstants↑ **18108**

2395 A Concrete Naming Certificate for CompletionUniformFunctorialEnvelope↑ **18111**

2396 A Concrete Naming Certificate for DarbouxProperty↑ **18114**

2397 A Concrete Naming Certificate for LocatedCompleteMetric↑ **18118**

2398 A Concrete Naming Certificate for CauchyCompletionBoundary↑ **18121**

2399 A Concrete Naming Certificate for MetricCompletionDenseEmbedding↑ **18124**

2400 A Concrete Naming Certificate for ClosedPiDomainSubjectReduction↑ **18127**

2401 A Concrete Naming Certificate for FiniteTraceInductionGap↑ **18130**

2402 A Concrete Naming Certificate for OvertReal↑ **18132**

2403 A Concrete Naming Certificate for CertificateMachine↑ **18135**

2404 A Concrete Naming Certificate for GeneratingFunction↑ **18139**

2405 A Concrete Naming Certificate for MetricIdentification↑ **18142**

2406 A Concrete Naming Certificate for EquicontinuityFiniteNetSelector↑ **18145**

2407 A Concrete Naming Certificate for RealBolzano↑ **18148**

2408 A Concrete Naming Certificate for CofinalFilterBase↑ **18153**

2409 A Concrete Naming Certificate for AbsolutelyContinuousFunction↑ **18156**

2410 A Concrete Naming Certificate for EberleinSmulian↑ **18160**

2411 A Concrete Naming Certificate for SorgenfreyLine↑ **18163**

2412 A Concrete Naming Certificate for CauchyTailDiagonal↑ **18166**

2413 A Concrete Naming Certificate for SeparatedLimitUniqueness↑ **18169**

2414 A Concrete Naming Certificate for CertificateMachineBranch↑ **18172**

2415 A Concrete Naming Certificate for SylvesterInertia↑ **18175**

2416 A Concrete Naming Certificate for FundamentalGroup↑ **18178**

2417 A Concrete Naming Certificate for PartitionOfUnity↑ **18181**

2418 A Concrete Naming Certificate for Bornology↑ **18184**

2419 A Concrete Naming Certificate for MetricAnnulus↑ **18189**

2420 A Concrete Naming Certificate for RealArchimedeanProperty↑ **18191**

2421 A Concrete Naming Certificate for LocatedMonotoneConvergence↑ **18194**

2422 A Concrete Naming Certificate for BaireSteinhaus↑ **18198**

2423 A Concrete Naming Certificate for BanachSteinhaus↑ **18201**

2424 A Concrete Naming Certificate for CauchyCompletionReflection↑ **18204**

2425 A Concrete Naming Certificate for CauchyRealArchimedean↑ **18208**

2426 A Concrete Naming Certificate for CauchyCompletionOperator↑ **18210**

2427 A Concrete Naming Certificate for RealCauchyCompleteness↑ **18221**

2428 A Concrete Naming Certificate for ClosedLamDomainSubjectReduction↑ **18225**

2429 A Concrete Naming Certificate for CauchyCompletenessCriterion↑ **18228**

2430 A Concrete Naming Certificate for TailModulusLimitHandoff↑ **18231**

2431 Concrete Naming Certificate for BorelSet↑ 18233

2432 Concrete Naming Certificate for OuterMeasure↑ 18236

2433 Concrete Naming Certificate for LocallyConvexSpace↑ 18239

2434 Concrete Naming Certificate for ForwardSocketLedger↑ 18250

2435 Concrete Naming Certificate for SuccessiveApproximation↑ 18253

2436 Concrete Naming Certificate for RealCauchyDenseEmbedding↑ 18255

2437 Concrete Naming Certificate for RegularCauchyUniqueness↑ 18258

2438 Concrete Naming Certificate for DyadicMeshComposition↑ 18263

2439 Concrete Naming Certificate for RegularCauchyLimitOperator↑ 18266

2440 Concrete Naming Certificate for DnaWindowSixBoundaryTile↑ 18269

2441 Concrete Naming Certificate for LocatedRealIntermediateValue↑ 18272

2442 Concrete Naming Certificate for LocatedRealLine↑ 18275

2443 Concrete Naming Certificate for LocatedUpperBound↑ 18278

2444 Concrete Naming Certificate for UniformEmbeddingCompletion↑ 18283

2445 Concrete Naming Certificate for RegularCauchyCompletionReflector↑ 18286

2446 Concrete Naming Certificate for SchwartzFunction↑ 18290

2446.1 Carrier definition 18290

2446.2 The certificate 18291

2447 Concrete Naming Certificate for AsymptoticEquivalence↑ 18294

2448 Concrete Naming Certificate for RieszLemma↑ 18297

2449 Concrete Naming Certificate for SimplicialApproximation↑ 18300

2450 Concrete Naming Certificate for MooreSmithNet↑ 18303

2451 Concrete Naming Certificate for BishopCompletionCriterion↑ 18306

2452 Concrete Naming Certificate for CauchyInterleavingCriterion↑ 18308

2453 Concrete Naming Certificate for LocatedRealModulus↑ 18311

2454 Concrete Naming Certificate for MonotoneBoundedSequence↑ 18314

2455 Concrete Naming Certificate for AiryFunction↑ 18319

2456 Concrete Naming Certificate for CauchyLocatedRealEquivalence↑ **18322**

2457 Concrete Naming Certificate for CauchyRealLocator↑ **18325**

2458 Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction↑ **18329**

2459 Concrete Naming Certificate for CauchySequenceAlgebra↑ **18334**

2460 Concrete Naming Certificate for Cauchy↑ **18340**

2461 Concrete Naming Certificate for FiniteSimplicialChain↑ **18350**

2462 Concrete Naming Certificate for RealModulusCriterion↑ **18352**

2463 Concrete Naming Certificate for CauchyFilterCriterion↑ **18354**

2464 Concrete Naming Certificate for UniformConvergence↑ **18357**

2465 Concrete Naming Certificate for BishopApproximationSequence↑ **18360**

2466 Concrete Naming Certificate for LaxMilgram↑ **18363**

2467 Concrete Naming Certificate for BanachAlaoglu↑ **18366**

2468 Concrete Naming Certificate for MonotoneCauchySequence↑ **18369**

2469 Concrete Naming Certificate for MetaCICCandidateSetSNHandoff↑ **18372**

2470 Concrete Naming Certificate for FilterConvergence↑ **18376**

2471 Concrete Naming Certificate for BishopCompleteMetric↑ **18380**

2472 Concrete Naming Certificate for FredholmAlternative↑ **18383**

2473 Concrete Naming Certificate for CousinCover↑ **18386**

2474 Concrete Naming Certificate for RealUniformTailComparison↑ **18389**

2475 Concrete Naming Certificate for CauchyChainCriterion↑ **18392**

2476 Concrete Naming Certificate for ClosedBoundedIntervalDarbouxPartition↑ **18394**

2477 Concrete Naming Certificate for FiniteCoverRefinement↑ **18397**

2478 Concrete Naming Certificate for LocatedIntervalCompactness↑ **18400**

2479 Concrete Naming Certificate for LimitComparisonTest↑ **18402**

2480 Concrete Naming Certificate for DenjoyIntegral↑ **18405**

2481 Concrete Naming Certificate for DecidableBetaFrontier↑ **18408**

2482 Concrete Naming Certificate for HardyInequality$^{\uparrow}$ **18412**

2483 Concrete Naming Certificate for CommutativeCStar$^{\uparrow}$ **18415**

2484 Concrete Naming Certificate for MetaCICCriticalPairLedger$^{\uparrow}$ **18418**

2485 Concrete Naming Certificate for RankNullity$^{\uparrow}$ **18421**

2486 Concrete Naming Certificate for BoundedNormalConversionBridge$^{\uparrow}$ **18424**

2487 Concrete Naming Certificate for DyadicComparisonPrinciple$^{\uparrow}$ **18427**

2488 Concrete Naming Certificate for CauchyQuotientCriterion$^{\uparrow}$ **18430**

2489 Concrete Naming Certificate for PicardFixedPointModulus$^{\uparrow}$ **18433**

2490 Concrete Naming Certificate for RectifiableCurve$^{\uparrow}$ **18436**

2491 Concrete Naming Certificate for HeineCantor$^{\uparrow}$ **18438**

2492 Concrete Naming Certificate for RealUniformSpace$^{\uparrow}$ **18442**

2493 Concrete Naming Certificate for BishopRealApartnessTopology$^{\uparrow}$ **18445**

2494 Concrete Naming Certificate for FatouLemma$^{\uparrow}$ **18448**

2495 Concrete Naming Certificate for RadonNikodym$^{\uparrow}$ **18451**

2496 Concrete Naming Certificate for GaussBonnetBoundary$^{\uparrow}$ **18454**

2497 Concrete Naming Certificate for ForceMetaCICSameSchema$^{\uparrow}$ **18457**

2498 Concrete Naming Certificate for RealBoundedSequence$^{\uparrow}$ **18460**

2499 Concrete Naming Certificate for RegularCauchyCompletionCriterion$^{\uparrow}$ **18464**

2500 Concrete Naming Certificate for ConstructiveDiniModulus$^{\uparrow}$ **18470**

2501 Concrete Naming Certificate for PoincareDiskPhaseTransport$^{\uparrow}$ **18473**

2502 Concrete Naming Certificate for MaschkeTheorem$^{\uparrow}$ **18476**

2503 Concrete Naming Certificate for NestedCauchyRealSection$^{\uparrow}$ **18479**

2504 Concrete Naming Certificate for RealUniformCauchyName$^{\uparrow}$ **18482**

2505 Concrete Naming Certificate for ProofObligationGap$^{\uparrow}$ **18485**

2506 Concrete Naming Certificate for MeasurableFunction$^{\uparrow}$ **18488**

2507 Concrete Naming Certificate for MooreSmithCauchyNetFilterBridge$^{\uparrow}$ **18491**

2508 A Concrete Naming Certificate for PuncturedCubeSkeleton↑ **18494**

2509 A Constructive Closed Metric Ball Certificate for ClosedMetricBall↑ **18497**

2510 A Concrete Naming Certificate for BrouwerContinuityPrinciple↑ **18500**

2511 A Concrete Naming Certificate for CauchyHadamard↑ **18503**

2512 A Concrete Naming Certificate for DyadicSubsequence↑ **18506**

2513 A Concrete Naming Certificate for RegularCauchyShiftEquivalence↑ **18509**

2514 A Concrete Naming Certificate for DiagonalCauchySubsequence↑ **18512**

2515 A Concrete Naming Certificate for SixBoundaryTileCoverage↑ **18515**

2516 A Concrete Naming Certificate for HeineBorelFiniteNet↑ **18518**

2517 A Concrete Naming Certificate for HilbertBasis↑ **18525**

2518 A Concrete Naming Certificate for CantorBernstein↑ **18527**

2519 A Concrete Naming Certificate for HyperbolicMetric↑ **18530**

2520 A Concrete Naming Certificate for DenseIsometryExtension↑ **18533**

2521 A Concrete Naming Certificate for CauchyCompletionProductPreservation↑ **18536**

2522 A Concrete Naming Certificate for WindowCoverageTriggerAlgebra↑ **18539**

2523 A Concrete Naming Certificate for BorelFunction↑ **18543**

2524 A Concrete Naming Certificate for ChebyshevPolynomial↑ **18546**

2525 A Concrete Naming Certificate for CellularWindowTrace↑ **18548**

2526 A Concrete Naming Certificate for WeakKonigBoundary↑ **18550**

2527 A Concrete Naming Certificate for CauchyCutBridge↑ **18553**

2528 A Concrete Naming Certificate for BishopCompletionTheorem↑ **18556**

2529 A Concrete Naming Certificate for BaireUltrametric↑ **18559**

2530 A Concrete Naming Certificate for HyperbolicExpansionLedger↑ **18564**

2531 A Concrete Naming Certificate for LiouvilleLedgerConservation↑ **18567**

2532 A Concrete Naming Certificate for BishopLocatedRealCompletion↑ **18570**

2533 A Concrete Naming Certificate for KirszbraunExtension↑ **18574**

2534 Concrete Naming Certificate for HyperbolicGeodesicBoundary$^{\uparrow}$ **18577**

2535 Concrete Naming Certificate for UniformClosure$^{\uparrow}$ **18580**

2536 Concrete Naming Certificate for CauchyFilterLocatedRealExtraction$^{\uparrow}$ **18583**

2537 Concrete Naming Certificate for SignedDyadicExpansion$^{\uparrow}$ **18585**

2538 Concrete Naming Certificate for FrechetSpace$^{\uparrow}$ **18589**

2539 Concrete Naming Certificate for FinitePartitionMesh$^{\uparrow}$ **18593**

2540 Concrete Naming Certificate for ChoiceWindowSequence$^{\uparrow}$ **18596**

2541 Concrete Naming Certificate for ProximinalSet$^{\uparrow}$ **18599**

2542 Concrete Naming Certificate for CompletionUnitComparison$^{\uparrow}$ **18602**

2543 Concrete Naming Certificate for PoincareBoundaryFixedTransport$^{\uparrow}$ **18605**

2544 Concrete Naming Certificate for MetricCompletionNaturality$^{\uparrow}$ **18608**

2545 Concrete Naming Certificate for AsymptoticSequence$^{\uparrow}$ **18611**

2546 Concrete Naming Certificate for FiniteSubcoverCriterion$^{\uparrow}$ **18614**

2547 Concrete Naming Certificate for RieszMarkov$^{\uparrow}$ **18617**

2547.1 Compact positive functional carrier 18617

2547.2 Certificate obligations 18618

2548 Constructive Apartness Certificate for MetricApartness$^{\uparrow}$ **18619**

2549 Concrete Naming Certificate for ContractiveCauchyOrbit$^{\uparrow}$ **18622**

2550 Concrete Naming Certificate for CompletionSubsetClosure$^{\uparrow}$ **18626**

2551 Concrete Naming Certificate for DnaBoundaryTriggerClosure$^{\uparrow}$ **18629**

2552 Concrete Naming Certificate for Ceva$^{\uparrow}$ **18632**

2553 Concrete Naming Certificate for BishopCauchyApproximation$^{\uparrow}$ **18635**

2554 Concrete Naming Certificate for KakutaniFixedPoint$^{\uparrow}$ **18637**

2555 Concrete Naming Certificate for RatCauchyCompletion$^{\uparrow}$ **18640**

2556 Concrete Naming Certificate for DyadicDensityRealEmbeddingComparison$^{\uparrow}$ **18643**

2557 Concrete Naming Certificate for CauchyInterleavingComparison$^{\uparrow}$ **18647**

2558 Concrete Naming Certificate for RegularCauchySubsequenceStability$^{\uparrow}$ **18650**

2559 A Concrete Naming Certificate for SpeckerSequenceBoundary↑ **18653**

2560 A Concrete Naming Certificate for BishopOrderCompletion↑ **18657**

2561 A Concrete Naming Certificate for EffectiveCauchyModulus↑ **18660**

2562 A Concrete Naming Certificate for MetacicRedexChoiceFrontier↑ **18663**

2563 A Concrete Naming Certificate for MetricCompletionFiniteCoproduct↑ **18667**

2564 A Concrete Naming Certificate for LatentCompletion↑ **18670**

2565 A Concrete Naming Certificate for BoundedFunctionFamily↑ **18673**

2566 A Concrete Naming Certificate for CauchyCompletionApartness↑ **18676**

2567 A Concrete Naming Certificate for BishopRegularCutEquivalence↑ **18679**

2568 A Concrete Naming Certificate for DyadicToleranceLadder↑ **18682**

2569 A Concrete Naming Certificate for NestedClosedBall↑ **18685**

2570 A Concrete Naming Certificate for RealBoundedInterval↑ **18692**

2571 A Concrete Naming Certificate for StieltjesMeasureBoundary↑ **18695**

2572 A Concrete Naming Certificate for BanachContraction↑ **18698**

2573 A Concrete Naming Certificate for PeanoExistence↑ **18702**

2574 A Concrete Naming Certificate for LocatedCompletionFunctor↑ **18705**

2575 A Concrete Naming Certificate for RegSeqRatLocatedModulus↑ **18708**

2576 A Concrete Naming Certificate for MorseTheory↑ **18711**

2577 A Concrete Naming Certificate for HyperbolicHorocycle↑ **18714**

2578 A Concrete Naming Certificate for CompleteMetricBaireSurface↑ **18716**

2579 A Concrete Naming Certificate for LocatedHeineBorelSubcover↑ **18719**

2580 A Concrete Naming Certificate for MetaCICTypedNormalizationAudit↑ **18723**

2581 A Concrete Naming Certificate for LocatedCompletionCriterion↑ **18727**

2582 A Concrete Naming Certificate for RegularCauchySequenceSpace↑ **18731**

2583 A Concrete Naming Certificate for CauchyCompletionDensitySource↑ **18735**

2584 A Concrete Naming Certificate for SeparatedCompletionReflection↑ **18738**

2585 Concrete Naming Certificate for ConvergentNet↑ **18741**

2586 Concrete Naming Certificate for MedianClosure↑ **18745**

2587 Concrete Naming Certificate for RegularCauchyReal↑ **18748**

2588 Concrete Naming Certificate for GDeltaSet↑ **18752**

2589 Concrete Naming Certificate for ClosedBallCompactness↑ **18755**

2590 Concrete Naming Certificate for CompactSourceTriangleEstimate↑ **18758**

2591 Concrete Naming Certificate for RealLineConnected↑ **18761**

2592 Concrete Naming Certificate for CauchyProductMetric↑ **18764**

2593 Concrete Naming Certificate for RegularCauchySpaceMorphism↑ **18767**

2594 Concrete Naming Certificate for MetricCompletionComparison↑ **18770**

2595 Concrete Naming Certificate for BaireTwoFunction↑ **18774**

2596 Concrete Naming Certificate for AlexanderDualTriggerAlgebra↑ **18777**

2597 Concrete Naming Certificate for MooreSmithCauchySubnetCriterion↑ **18780**

2598 Concrete Naming Certificate for ReassignmentSupportCompression↑ **18783**

2599 Concrete Naming Certificate for TopologicalClosure↑ **18786**

2600 Concrete Naming Certificate for UniformSpaceCompletion↑ **18789**

2601 Concrete Naming Certificate for RegularCauchySubspace↑ **18792**

2602 Concrete Naming Certificate for MetricReflectionSandwich↑ **18795**

2603 Concrete Naming Certificate for ConvexConjugate↑ **18798**

2604 Concrete Naming Certificate for LocatedApartnessRealBridge↑ **18801**

2605 Concrete Naming Certificate for NormalizationCandidateSocket↑ **18807**

2606 Concrete Naming Certificate for MetaCICSnCandidateInterface↑ **18815**

2607 Concrete Naming Certificate for ArzelaAscoliFiniteModulus↑ **18818**

2608 Concrete Naming Certificate for CompactTotallyBounded↑ **18835**

2609 Concrete Naming Certificate for BaireCategoryTheorem↑ **18838**

2610 Concrete Naming Certificate for RationalBracketRefinement↑ **18842**

2611 Concrete Naming Certificate for LocatedCompactIntervalOscillationBound↑ **18845**

2612 Concrete Naming Certificate for CharacterTable↑ **18848**

2613 Concrete Naming Certificate for DenjoyWolffBoundaryDynamics↑ **18850**

2614 Concrete Naming Certificate for BishopCauchyCriterion↑ **18853**

2615 Concrete Naming Certificate for BishopCauchyModulusEquivalence↑ **18856**

2616 Concrete Naming Certificate for BurnsideOrbitCount↑ **18858**

2617 Concrete Naming Certificate for CauchyCompletenessWitness↑ **18861**

2618 Concrete Naming Certificate for CauchyCompletenessWitnessObligation↑ **18864**

2619 Concrete Naming Certificate for HardyLittlewoodMaximal↑ **18866**

2620 Concrete Naming Certificate for RationalRootInterval↑ **18869**

2621 Concrete Naming Certificate for EnergyBiasedTriggerReliability↑ **18872**

2622 Concrete Naming Certificate for MertensCauchyProductTheorem↑ **18875**

2623 Concrete Naming Certificate for BishopCompletionUniversalComposition↑ **18879**

2624 Concrete Naming Certificate for FiniteModulusCompactness↑ **18882**

2625 Concrete Naming Certificate for LocatedCauchyNet↑ **18885**

2626 Concrete Naming Certificate for AlternativeCodeReassignment↑ **18888**

2627 Concrete Naming Certificate for UniformReflectionKernel↑ **18891**

2628 Concrete Naming Certificate for PhaseLedgerHorocycle↑ **18893**

2629 Concrete Naming Certificate for MooreSpace↑ **18896**

2630 Concrete Naming Certificate for VariationMeasure↑ **18898**

2631 Concrete Naming Certificate for CauchyContinuityCalibration↑ **18901**

2632 Concrete Naming Certificate for MetaCICLocalDiamondWitness↑ **18904**

2633 Concrete Naming Certificate for TangentCone↑ **18907**

2634 Concrete Naming Certificate for WeakTopology↑ **18909**

2635 Concrete Naming Certificate for CondensationTailSelector↑ **18912**

2636 Concrete Naming Certificate for NestedIntervalCauchy↑ **18915**

2637 Concrete Naming Certificate for TypeLike↑ **18918**

2638 Concrete Naming Certificate for CauchyCompleteApartnessSpace↑ **18922**

2639 Concrete Naming Certificate for RegularCauchyTailFunctor↑ **18925**

2640 Concrete Naming Certificate for OperatorNorm↑ **18928**

2641 Concrete Naming Certificate for ContinuousImageCompact↑ **18931**

2642 Concrete Naming Certificate for FiniteCoverLebesgueNumber↑ **18934**

2643 Concrete Naming Certificate for ContractionPrinciple↑ **18937**

2644 Concrete Naming Certificate for CauchyProductConvergence↑ **18940**

2645 Concrete Naming Certificate for CauchyProductMertens↑ **18943**

2646 Concrete Naming Certificate for KreinMilmanFiniteExtreme↑ **18945**

2647 Concrete Naming Certificate for CFunCarrier↑ **18948**

2648 Concrete Naming Certificate for SturmRootCount↑ **18951**

2649 Concrete Naming Certificate for UniformCauchyBicompletion↑ **18954**

2650 Concrete Naming Certificate for DyadicCauchyApproximation↑ **18957**

2651 Concrete Naming Certificate for SequentialCauchyRegularization↑ **18960**

2652 Concrete Naming Certificate for MetaCICDecidableTypingFrontier↑ **18963**

2653 Concrete Naming Certificate for DenseEmbedding↑ **18967**

2654 Concrete Naming Certificate for UniformCauchyCompletion↑ **18970**

2655 Concrete Naming Certificate for ArchimedeanCompletion↑ **18973**

2656 Concrete Naming Certificate for RieszSpace↑ **18976**

2657 Concrete Naming Certificate for ArzelaAscoliFiniteWindowDiagonalBridge↑ **18979**

2658 Concrete Naming Certificate for BishopCauchySequence↑ **18981**

2659 Concrete Naming Certificate for MeirKeelerContraction↑ **18984**

2660 Concrete Naming Certificate for IntervalHalvingModulus↑ **18987**

2661 Concrete Naming Certificate for PostMedianGateShell↑ **18990**

2662 Concrete Naming Certificate for MetaCICDependentCodomainStabilityBoundary↑ **18993**

2663 Concrete Naming Certificate for DyadicSandwich↑ **18996**

2664 Concrete Naming Certificate for CauchyCompletionRecursor↑ **18999**

2665 Concrete Naming Certificate for DedekindReal↑ **19002**

2666 Concrete Naming Certificate for ObservableDiameterDecay↑ **19005**

2667 Concrete Naming Certificate for FastCauchyNormalForm↑ **19008**

2668 Concrete Naming Certificate for Calculus↑ **19011**

2669 Concrete Naming Certificate for FrechetDerivative↑ **19050**

2670 Concrete Naming Certificate for BoundedRealIntervalPartition↑ **19053**

2671 Concrete Naming Certificate for CauchySequenceOrder↑ **19057**

2672 Concrete Naming Certificate for NoetherianRing↑ **19061**
2672.0 Carrier definition 19061

2673 Concrete Naming Certificate for LeviCivitaConnection↑ **19063**

2674 Concrete Naming Certificate for BishopCauchyCompletionFunctor↑ **19066**

2675 Concrete Naming Certificate for RealLocatedOrder↑ **19068**

2676 Concrete Naming Certificate for RealSequenceSpace↑ **19071**

2677 Concrete Naming Certificate for FrechetUrysohn↑ **19075**

2678 Concrete Naming Certificate for HyperbolicBoundaryPhaseAudit↑ **19078**

2679 Concrete Naming Certificate for SardTheoremFiniteJet↑ **19081**

2680 Concrete Naming Certificate for BusemannBoundary↑ **19084**

2681 Concrete Naming Certificate for RegularCauchyCompletionUnit↑ **19086**

2682 Concrete Naming Certificate for SobolevTrace↑ **19089**

2683 Concrete Naming Certificate for NoetherianModule↑ **19092**
2683.0 Carrier definition 19092

2684 Concrete Naming Certificate for LaplaceTransform↑ **19094**

2685 Concrete Naming Certificate for LipschitzExtension↑ **19097**

2686 Concrete Naming Certificate for AnchorSubcubeTrigger↑ **19100**

2687 Concrete Naming Certificate for HeatEquationFiniteStencil↑ **19103**

2688 A Concrete Naming Certificate for HausdorffDistance↑ **19106**

2689 A Concrete Naming Certificate for BoundedVariationRealFunction↑ **19108**

2690 A Concrete Naming Certificate for RegularCauchyLocatedModulus↑ **19112**
2690. Located modulus carrier 19112

2691 A Concrete Naming Certificate for CauchyCompletionUniversalReflector↑ **19115**

2692 A Concrete Naming Certificate for TriggerHypergraphReliability↑ **19119**

2693 A Concrete Naming Certificate for LocatedCompleteInterval↑ **19122**
2693. Located Complete Interval Carrier 19122

2694 A Concrete Naming Certificate for IntervalCauchyCompleteness↑ **19125**
2694. Interval Cauchy Completeness Carrier 19125

2695 A Concrete Naming Certificate for LimitPointCompact↑ **19128**

2696 A Concrete Naming Certificate for ClosedNormalConsistencyWitness↑ **19131**

2697 A Concrete Naming Certificate for FiniteGroupRepresentation↑ **19134**

2698 A Concrete Naming Certificate for KSpace↑ **19137**

2699 A Concrete Naming Certificate for CompactUniformFiniteCoverSelector↑ **19140**

2700 A Concrete Naming Certificate for BishopRealCompletion↑ **19143**

2701 A Concrete Naming Certificate for UniformCompletionUniversalProperty↑ **19146**

2702 A Concrete Naming Certificate for CauchyCompletionStableReflection↑ **19149**

2703 A Concrete Naming Certificate for SetLike↑ **19152**

2704 A Concrete Naming Certificate for SchauderBasis↑ **19191**

2705 A Concrete Naming Certificate for RationalCauchyCompletion↑ **19194**

2706 A Concrete Naming Certificate for CompactMetricClosedBall↑ **19197**

2707 A Concrete Naming Certificate for RegularCauchyProductCriterion↑ **19200**

2708 A Concrete Naming Certificate for BishopMetricSpace↑ **19203**

2709 A Concrete Naming Certificate for RealClosure↑ **19207**

2710 A Concrete Naming Certificate for IntermediateValueTheorem↑ **19210**

2711 A Concrete Naming Certificate for FirstExitChannel↑ **19213**

2712 A Concrete Naming Certificate for WhitneyExtensionFiniteJet↑ **19216**

2713 Concrete Naming Certificate for HausdorffCompletionKernel$^{\uparrow}$ **19219**

2714 Concrete Naming Certificate for RadialWobbleSpectrum$^{\uparrow}$ **19222**

2715 Concrete Naming Certificate for SchwarzPickDiskContraction$^{\uparrow}$ **19225**

2716 Concrete Naming Certificate for PoincareTriangleDefect$^{\uparrow}$ **19228**

2717 Concrete Naming Certificate for CauchyBoundednessCriterion$^{\uparrow}$ **19231**

2718 Concrete Naming Certificate for LocatedMetricSpace$^{\uparrow}$ **19234**

2719 Concrete Naming Certificate for ProperMetricSpace$^{\uparrow}$ **19237**

2720 Concrete Naming Certificate for CompleteMetricSpace$^{\uparrow}$ **19240**

2721 Concrete Naming Certificate for CauchyCompleteOrderedField$^{\uparrow}$ **19243**

2722 Concrete Naming Certificate for CauchyTailDominance$^{\uparrow}$ **19253**

2723 Concrete Naming Certificate for MonotoneCauchySubsequence$^{\uparrow}$ **19258**

2724 Concrete Naming Certificate for CauchySequenceCompletion$^{\uparrow}$ **19261**

2725 Concrete Naming Certificate for BoundedIntervalProjection$^{\uparrow}$ **19264**

2726 Concrete Naming Certificate for MontelTheorem$^{\uparrow}$ **19267**

2727 Concrete Naming Certificate for AntipodalTriggerLedger$^{\uparrow}$ **19270**

2728 Concrete Naming Certificate for EffectiveCauchyCriterion$^{\uparrow}$ **19273**

2729 Concrete Naming Certificate for UniformlyCauchySequence$^{\uparrow}$ **19276**

2730 Concrete Naming Certificate for DyadicMeshScaleNormalization$^{\uparrow}$ **19279**

2731 Concrete Naming Certificate for CompactCompletionExtensionModulus$^{\uparrow}$ **19282**

2732 Concrete Naming Certificate for UniformlyContinuousCompletionExtension$^{\uparrow}$ **19285**

2733 Concrete Naming Certificate for BishopMonotoneRealConvergence$^{\uparrow}$ **19288**

2734 Concrete Naming Certificate for CauchyLimitClassifierLock$^{\uparrow}$ **19291**

2735 Concrete Naming Certificate for DarbouxTheorem$^{\uparrow}$ **19294**

2736 Concrete Naming Certificate for LocatedCompletionUniversalProperty$^{\uparrow}$ **19297**

2737 Concrete Naming Certificate for BooleanTriggerLattice$^{\uparrow}$ **19300**

2738 Concrete Naming Certificate for BishopCauchyComparison$^{\uparrow}$ **19302**

2739 A Concrete Naming Certificate for ClosedObservationTotalHost↑ **19305**

2740 A Concrete Naming Certificate for RegularCauchyAlgebraCoherence↑ **19307**

2741 A Concrete Naming Certificate for CauchySequenceAlgebra↑ **19310**

2742 A Concrete Naming Certificate for CauchySequenceProductStability↑ **19315**

2743 A Concrete Naming Certificate for OresmeHarmonicDivergence↑ **19318**

2744 A Concrete Naming Certificate for Subdifferential↑ **19321**
2744.1 Subdifferential Carrier 19321
2744.2 Subdifferential Certificate 19322

2745 A Concrete Naming Certificate for ProximalMap↑ **19325**
2745.1 ProximalMap Carrier 19325
2745.2 ProximalMap Certificate 19326

2746 A Concrete Naming Certificate for TypedNormalFormDecider↑ **19329**

2747 A Concrete Naming Certificate for ResolventIdentity↑ **19332**

2748 A Concrete Naming Certificate for BoundedIntervalCauchySubsequence↑ **19335**

2749 A Concrete Naming Certificate for BishopCompletion↑ **19339**

2750 A Concrete Naming Certificate for RegularCauchyModulusExtraction↑ **19343**

2751 A Concrete Naming Certificate for CauchySelectionFunction↑ **19346**

2752 A Concrete Naming Certificate for CauchySumLimit↑ **19349**

2753 A Concrete Naming Certificate for CauchyProductLimit↑ **19352**

2754 A Concrete Naming Certificate for RealCompletenessDiagonalSelector↑ **19355**

2755 A Concrete Naming Certificate for LeibnizSeriesTest↑ **19358**

2756 A Concrete Naming Certificate for LipschitzModulus↑ **19361**

2757 A Concrete Naming Certificate for ModulusCauchyCompletion↑ **19364**

2758 A Concrete Naming Certificate for UniformlyLocatedFiniteCover↑ **19366**

2759 Cauchy Condensation Theorem NameCert Construction **19368**

2760 A Concrete Naming Certificate for Sylow↑ **19371**

2761 A Concrete Naming Certificate for BernsteinPolynomial↑ **19375**

2762 A Concrete Naming Certificate for SequentialCauchySelector↑ **19377**

2763 Concrete Naming Certificate for HyperbolicBusemannBoundary↑ **19381**

2764 Concrete Naming Certificate for BooleanAlgebra↑ **19384**

2765 Concrete Naming Certificate for CoverageRankAlgebra↑ **19391**

2766 Concrete Naming Certificate for CauchySequenceOrderCompleteness↑ **19394**

2767 Concrete Naming Certificate for HilleYosida↑ **19397**

2768 Concrete Naming Certificate for CompactUniformOscillation↑ **19400**

2769 Concrete Naming Certificate for RegularCauchyDiagonalWindow↑ **19403**

2770 Concrete Naming Certificate for DunfordFunctionalCalculus↑ **19406**

2771 Concrete Naming Certificate for UniformCompletionCommonWindow↑ **19409**

2772 Concrete Naming Certificate for FiniteMetricSpace↑ **19412**

2773 Concrete Naming Certificate for RegularCauchyTailBound↑ **19415**

2774 Concrete Naming Certificate for HochsterNerveBettiReduction↑ **19418**

2775 Concrete Naming Certificate for BrouwerDegree↑ **19421**

2776 Concrete Naming Certificate for Metacompact↑ **19424**

2777 Concrete Naming Certificate for FiniteRecognizerCarrierObstruction↑ **19427**

2778 Concrete Naming Certificate for RegularCauchyCanonicalModulus↑ **19431**

2779 Concrete Naming Certificate for RegularCauchyCompletionStability↑ **19434**

2780 Concrete Naming Certificate for BoundedMonotoneSubsequence↑ **19437**

2781 Concrete Naming Certificate for DistributionDerivative↑ **19440**

2782 Concrete Naming Certificate for NetClusterPoint↑ **19443**

2783 Concrete Naming Certificate for GreedySpectralClosure↑ **19446**

2784 Concrete Naming Certificate for CauchyCompleteOrderedFieldUniqueness↑ **19449**

2785 Concrete Naming Certificate for MetricCompletionFunctoriality↑ **19453**

2786 Concrete Naming Certificate for NaturalDeduction↑ **19456**

2787 Concrete Naming Certificate for RegularCauchyFiniteSum↑ **19459**

2788 Concrete Naming Certificate for EpsilonDeltaLimit↑ **19462**

2789 A Concrete Naming Certificate for PackingNumber↑ **19465**

2790 A Concrete Naming Certificate for LipschitzFunction↑ **19468**

2791 A Concrete Naming Certificate for CantorTernarySet↑ **19471**

2792 A Concrete Naming Certificate for CyclicTagSystem↑ **19474**

2793 A Concrete Naming Certificate for HausdorffContent↑ **19476**

2794 A Concrete Naming Certificate for CauchyTailFilterBornology↑ **19479**

2795 A Concrete Naming Certificate for SequentialCompletenessCriterion↑ **19482**

2796 A Concrete Naming Certificate for RegularModulus↑ **19484**

2797 A Concrete Naming Certificate for CompactLocatedIntervalSubcover↑ **19486**

2798 A Concrete Naming Certificate for RealAbsoluteValue↑ **19489**

2799 A Concrete Naming Certificate for CUNSpectralAntenna↑ **19492**

2800 A Concrete Naming Certificate for FrechetCauchyCriterion↑ **19495**

2801 A Concrete Naming Certificate for LocatedCutReal↑ **19498**

2802 A Concrete Naming Certificate for DyadicStepApproximation↑ **19501**

2803 A Concrete Naming Certificate for HausdorffUniformSpace↑ **19504**

2804 A Concrete Naming Certificate for SecantMethod↑ **19507**

2805 A Concrete Naming Certificate for CauchyCompletionBind↑ **19510**

2806 A Concrete Naming Certificate for GlobalFiberSpectra↑ **19513**

2807 A Concrete Naming Certificate for MetaCICNormalizationAuditPacket↑ **19516**

2808 A Concrete Naming Certificate for StepFunction↑ **19520**

2809 A Concrete Naming Certificate for SternBrocotContinuedFractionBridge↑ **19523**

2810 A Concrete Naming Certificate for RegularCauchyDiagonalization↑ **19527**

2811 A Concrete Naming Certificate for TotallyBoundedClosure↑ **19531**

2812 A Concrete Naming Certificate for UniformModulusComposition↑ **19534**

2813 A Concrete Naming Certificate for ConstructiveCauchyInterval↑ **19537**

2814 A Concrete Naming Certificate for CauchyCompletionDenseReflector↑ **19540**

2815 A Concrete Naming Certificate for LocallyCompactMetric↑ **19543**

2816 A Concrete Naming Certificate for LocallyCompactPolish↑ **19546**

2817 A Concrete Naming Certificate for SequentialRealCompactness↑ **19550**

2818 A Concrete Naming Certificate for LawlikeSequence↑ **19553**

2819 A Concrete Naming Certificate for MetaCICConfluenceSubstitutionBoundary↑ **19556**

2820 A Concrete Naming Certificate for CauchyCompletionComonad↑ **19560**

2821 A Concrete Naming Certificate for RealDiagonalNonSurjection↑ **19564**

2822 A Concrete Naming Certificate for CauchyNetCompletion↑ **19573**

2823 A Concrete Naming Certificate for RealTightness↑ **19582**

2824 A Concrete Naming Certificate for SchwarzLemma↑ **19585**

2825 A Concrete Naming Certificate for CauchyProductCompletion↑ **19588**

2826 A Concrete Naming Certificate for CantorSpaceCompactness↑ **19591**

2827 A Concrete Naming Certificate for CauchySchlomilchCondensation↑ **19594**

2828 A Concrete Naming Certificate for FiniteOscillationPartition↑ **19597**

2829 A Concrete Naming Certificate for TriggerBlockerDuality↑ **19602**

2830 A Concrete Naming Certificate for FiniteRelation↑ **19605**

2830. Finite-relation source 19605

2831 A Concrete Naming Certificate for RegularCauchyTailEnvelope↑ **19607**

2832 A Concrete Naming Certificate for VariationNorm↑ **19610**

2833 A Concrete Naming Certificate for RegularCauchyEntourageBasis↑ **19613**

2834 A Concrete Naming Certificate for TriggerEnergyPartition↑ **19616**

2835 A Concrete Naming Certificate for ContinuousSupNorm↑ **19619**

2836 A Concrete Naming Certificate for BoundedResidualSubstitutionFrontier↑ **19623**

2837 A Concrete Naming Certificate for LocatedRealContinuity↑ **19626**

2838 A Concrete Naming Certificate for CellularSubstrateClassifierBoundary↑ **19629**

2839 A Concrete Naming Certificate for HadamardProductSeries↑ **19632**

2840 A Concrete Naming Certificate for LipschitzContinuityHandoff↑ **19635**

2841 Concrete Naming Certificate for HyperbolicBoundaryShadow↑ **19638**

2842 Concrete Naming Certificate for LocatedUniformSpace↑ **19641**

2843 Concrete Naming Certificate for ClosedIntervalBisection↑ **19644**

2844 Concrete Naming Certificate for RealSpectrum↑ **19648**

2845 Concrete Naming Certificate for BanachLattice↑ **19651**

2846 Concrete Naming Certificate for HeronSquareRoot↑ **19654**

2847 Concrete Naming Certificate for ContractiveMap↑ **19669**

2848 Concrete Naming Certificate for ClosedBoundedSequence↑ **19673**

2849 Concrete Naming Certificate for CompactContinuityRadiusChoice↑ **19676**

2850 Concrete Naming Certificate for DuhamelPrinciple↑ **19679**

2851 Concrete Naming Certificate for LocatedSequenceLimit↑ **19682**

2852 Concrete Naming Certificate for DedekindGapLocatedCut↑ **19685**

2853 Concrete Naming Certificate for FastCauchyCriterion↑ **19688**

2854 Concrete Naming Certificate for CauchyCompletionKleisliCategory↑ **19691**

2855 Concrete Naming Certificate for MetaCICConfluenceResidualChecker↑ **19694**

2856 Concrete Naming Certificate for DarbouxContinuity↑ **19697**

2857 Concrete Naming Certificate for ConcaveModulus↑ **19701**

2858 Concrete Naming Certificate for SignedDigitIntervalReal↑ **19704**

2859 Concrete Naming Certificate for CayleyTheorem↑ **19708**

2860 Concrete Naming Certificate for CauchyMeshRefinement↑ **19711**

2861 Concrete Naming Certificate for MonotoneConvergenceReal↑ **19714**

2862 Concrete Naming Certificate for CauchyCompletionMonadLaw↑ **19717**

2863 Concrete Naming Certificate for CompleteUniformSpaceFilterbaseCompletion↑ **19730**

2864 Concrete Naming Certificate for MetricCompletionAdjunction↑ **19734**

2865 Concrete Naming Certificate for GeneratorClosureRobustness↑ **19737**

2866 Concrete Naming Certificate for ComputableUniformContinuity↑ **19742**

2867 A Concrete Naming Certificate for RegularCauchyCompletionSeparation↑ **19745**

2868 A Concrete Naming Certificate for WRRHotspotCube↑ **19748**

2869 A Concrete Naming Certificate for NestedCauchyIntervalSelection↑ **19751**

2870 A Concrete Naming Certificate for BoundaryPressureLedger↑ **19754**

2871 A Concrete Naming Certificate for CompletionAdjoint↑ **19757**

2872 A Concrete Naming Certificate for UniformEquicontinuityModulus↑ **19760**

2873 A Concrete Naming Certificate for LimitSpace↑ **19763**

2874 A Concrete Naming Certificate for FrechetKolmogorovCompactness↑ **19766**

2875 A Concrete Naming Certificate for RationalBracketing↑ **19778**

2876 A Concrete Naming Certificate for RealModulusExtraction↑ **19781**

2877 A Concrete Naming Certificate for CauchyStability↑ **19784**

2878 A Concrete Naming Certificate for CauchySequenceTailFilter↑ **19787**

2879 A Concrete Naming Certificate for CauchySequenceTailFilterCompletion↑ **19790**

2880 A Concrete Naming Certificate for LocalizedCauchyRate↑ **19794**

2881 A Concrete Naming Certificate for FiniteOscillationUniformModulus↑ **19797**

2882 A Concrete Naming Certificate for FubiniFiniteRectangle↑ **19800**

2883 A Concrete Naming Certificate for HyperbolicGeodesicFlow↑ **19803**

2884 A Concrete Naming Certificate for LocatedSequence↑ **19806**

2885 A Concrete Naming Certificate for CauchyModulusExtractor↑ **19809**

2886 A Concrete Naming Certificate for RealArithmeticProgression↑ **19813**

2887 A Concrete Naming Certificate for RegularCauchyReflectionLedger↑ **19816**

2888 A Concrete Naming Certificate for TubeLemma↑ **19819**

2889 A Concrete Naming Certificate for FilterLimitBasis↑ **19821**

2890 A Concrete Naming Certificate for PerfectSet↑ **19832**

2891 A Concrete Naming Certificate for RegularCauchyBinaryInterleaving↑ **19835**

2892 A Concrete Naming Certificate for ?? **19838**

2893 Concrete Naming Certificate for HyperbolicVisualMetric↑ **19841**

2894 Concrete Naming Certificate for MetricCauchyNetCluster↑ **19844**

2895 Concrete Naming Certificate for CompactIntervalModulusBridge↑ **19847**

2896 Concrete Naming Certificate for LocatedRealBisectionSearch↑ **19850**

2897 Concrete Naming Certificate for ConditionalTriggerEnsemble↑ **19853**

2898 Concrete Naming Certificate for NormalizationEvidenceBudget↑ **19856**

2899 Concrete Naming Certificate for RationalIntervalSubdivision↑ **19859**

2900 Concrete Naming Certificate for UniformSequentialLimit↑ **19862**

2901 Concrete Naming Certificate for CauchyFilterConvergence↑ **19864**

2902 Concrete Naming Certificate for UniformConvergenceCauchyCriterion↑ **19867**

2903 Concrete Naming Certificate for BoxDimension↑ **19870**

2904 Concrete Naming Certificate for CompleteUltrametric↑ **19873**

2905 Concrete Naming Certificate for PainleveKuratowskiConvergence↑ **19875**

2906 Concrete Naming Certificate for HaarWavelet↑ **19878**

2907 Concrete Naming Certificate for CauchyRegularCompletionAdjunction↑ **19881**

2908 Concrete Naming Certificate for QuasiIsometry↑ **19884**

2909 Concrete Naming Certificate for DyadicClosedIntervalEndpointGrid↑ **19887**

2910 Concrete Naming Certificate for CauchyFilterCompletionCriterion↑ **19890**

2911 Concrete Naming Certificate for ConstructiveUniformLimit↑ **19894**

2912 Concrete Naming Certificate for CompactnessUniformContinuityDiameterDecay↑ **19897**

2913 Concrete Naming Certificate for MetricCompletionIdempotence↑ **19900**

2914 Concrete Naming Certificate for MetaCICFuelTrace↑ **19904**

2915 Concrete Naming Certificate for CauchyStructure↑ **19907**

2916 Concrete Naming Certificate for UniformContinuityComposition↑ **19910**

2917 Concrete Naming Certificate for HadamardThreeCircle↑ **19915**

2918 Concrete Naming Certificate for HadamardThreeLines↑ **19919**

291A Concrete Naming Certificate for CauchyLocatedApartness↑ **19922**

292A Concrete Naming Certificate for RiemannLebesgueLemma↑ **19925**

292A Concrete Naming Certificate for MedianClosureSpectrum↑ **19928**

292A Concrete Naming Certificate for DirectedFilter↑ **19931**

292A Concrete Naming Certificate for OscillationFunction↑ **19934**

292A Concrete Naming Certificate for FilterCompletionUniversalProperty↑ **19937**

292A Concrete Naming Certificate for FiniteChoiceWindow↑ **19941**

292A Concrete Naming Certificate for CauchySequenceField↑ **19943**

292A Concrete Naming Certificate for BishopLocatedCompletionReflector↑ **19946**

292A Concrete Naming Certificate for CellularOrbitSynchronization↑ **19949**

292A Concrete Naming Certificate for FiniteDimensionalHahnBanachSeparation↑ **19952**

293A Concrete Naming Certificate for PiecewiseLinearApproximation↑ **19955**

293A Concrete Naming Certificate for CompactIntervalMaximum↑ **19958**

293A Concrete Naming Certificate for FilterCompletionMonad↑ **19961**

293A Concrete Naming Certificate for MetaCICDecidableNeutralSpine↑ **19964**

293A Concrete Naming Certificate for YoungIntegral↑ **19967**

293A Concrete Naming Certificate for TypeCheckingClassifierMembership↑ **19970**

293A Concrete Naming Certificate for CauchyCompletionReflectorComparison↑ **19973**

293A Concrete Naming Certificate for CompactRegularCauchySubsequence↑ **19975**

293A Concrete Naming Certificate for LindelofSpace↑ **19978**

293A Concrete Naming Certificate for IntervalCoverRefinementTree↑ **19981**

294A Concrete Naming Certificate for NeumannSeries↑ **19984**

294A Concrete Naming Certificate for CauchyWindowStabilityBridge↑ **19987**

294A Concrete Naming Certificate for MetaCICPiAdequacyBudget↑ **19990**

294B inite Cauchy Diagonal NameCert Construction **19993**

294M etaCIC Confluence Residual Envelope NameCert Construction **19996**

2945 Concrete Naming Certificate for HahnBanach↑ **20002**

2946 Concrete Naming Certificate for RealIntervalCompletion↑ **20005**

2947 Concrete Naming Certificate for CompactContinuousImageTotalBounded↑ **20008**

2948 Concrete Naming Certificate for HyperbolicPoissonKernel↑ **20011**

2949 Concrete Naming Certificate for EkelandVariationalPrinciple↑ **20014**

2950 Concrete Naming Certificate for KanExtension↑ **20017**

2951 Concrete Naming Certificate for MonotoneConvergenceSequence↑ **20020**

2952 Concrete Naming Certificate for RegularCauchyPicardSeal↑ **20023**

2953 Concrete Naming Certificate for ArzelaAscoliSelection↑ **20026**

2954 Concrete Naming Certificate for ContractionMappingTheorem↑ **20029**

2955 Concrete Naming Certificate for HilbertAlexanderBlocker↑ **20032**

2956 Concrete Naming Certificate for LocatedCompactIntervalSelection↑ **20035**

2957 Concrete Naming Certificate for CompactUniformRadiusLedger↑ **20038**

2958 Concrete Naming Certificate for LowerSemicontinuityReal↑ **20041**

2959 Concrete Naming Certificate for RegularCauchyLimitSelector↑ **20044**

2960 Concrete Naming Certificate for SubordinateFiniteModulusCover↑ **20046**

2961 Concrete Naming Certificate for LocatedIntermediateValue↑ **20048**

2962 Concrete Naming Certificate for ClosedBoundedIntervalFan↑ **20051**

2963 Concrete Naming Certificate for DiagonalSequence↑ **20054**

2964 Concrete Naming Certificate for WobbleSaturationSeal↑ **20056**

2965 Concrete Naming Certificate for RegularCauchyLimitUniqueness↑ **20059**

2966 Concrete Naming Certificate for FormalBallBasis↑ **20062**

2967 Concrete Naming Certificate for RealConvergentSequence↑ **20065**

2968 Concrete Naming Certificate for LocatedCauchySubsequence↑ **20069**

2969 Concrete Naming Certificate for CompletionReflectorTriangle↑ **20072**

2970 Concrete Naming Certificate for LawvereMetricCompletion↑ **20075**

2971 Concrete Naming Certificate for HyperbolicCrossRatio↑ **20081**

2972 Concrete Naming Certificate for RegularCauchyFilterFunctor↑ **20086**

2973 Concrete Naming Certificate for ArzelaAscoliFiniteEquicontinuous↑ **20089**

2974 Concrete Naming Certificate for PoissonSummation↑ **20092**

2975 Concrete Naming Certificate for RealDecimalNormalForm↑ **20095**

2976 Concrete Naming Certificate for CauchyModulusNormalizer↑ **20098**

2977 Concrete Naming Certificate for MetaCICStrongNormalizationBoundary↑ **20101**

2978 Concrete Naming Certificate for CofinalFilter↑ **20104**

2979 Concrete Naming Certificate for RegularCauchyCompletionFunctor↑ **20107**

2980 Concrete Naming Certificate for CompactUniformFiniteNetRealization↑ **20111**

2981 Concrete Naming Certificate for MetricSequentialCompletion↑ **20114**

2982 Concrete Naming Certificate for SpectralAblationHinge↑ **20117**

2983 Concrete Naming Certificate for SierpinskiCarpet↑ **20120**

2984 Concrete Naming Certificate for SpectralAblationHingeLedger↑ **20123**

2985 Concrete Naming Certificate for BanachContractionPrinciple↑ **20127**

2986 Concrete Naming Certificate for PublicLocalReflectiveCompilation↑ **20131**

2987 Concrete Naming Certificate for MetricEntourage↑ **20134**

2988 Concrete Naming Certificate for LeastUpperBound↑ **20136**

2989 Concrete Naming Certificate for CauchyIntervalNesting↑ **20139**

2990 Concrete Naming Certificate for FormalTopology↑ **20142**

2991 Concrete Naming Certificate for FormalBallRadiusOrder↑ **20145**

2992 Concrete Naming Certificate for ClosedBoundedIntervalCompact↑ **20147**

2993 Concrete Naming Certificate for FiniteSequencePeak↑ **20151**

2994 Concrete Naming Certificate for LocatedIntervalBolzano↑ **20154**

2995 Concrete Naming Certificate for BishopRealLocatedOrder↑ **20157**

2996 Concrete Naming Certificate for NestedCompactIntersection↑ **20160**

299.7 Concrete Naming Certificate for MercerTheorem↑ **20163**

299.8 Concrete Naming Certificate for CauchyCompletionRegularSequence↑ **20166**

299.9 Concrete Naming Certificate for CosetSpace↑ **20169**

300.0 Concrete Naming Certificate for UniformCauchyNet↑ **20172**

300.1 Concrete Naming Certificate for ContourResidueHandoff↑ **20175**

300.2 Concrete Naming Certificate for InitialValueProblem↑ **20178**

300.3 Concrete Naming Certificate for UniformCauchyLimit↑ **20181**

300.4 Concrete Naming Certificate for MedianDepthSealing↑ **20184**

300.5 Concrete Naming Certificate for UniformFixedPoint↑ **20187**

300.6 Concrete Naming Certificate for CauchySchwarzEquality↑ **20190**

300.7 Concrete Naming Certificate for ExtremeValueTheorem↑ **20193**

300.8 Concrete Naming Certificate for BoundedRealOscillation↑ **20196**

300.9 Concrete Naming Certificate for BishopCompactInterval↑ **20199**

301.0 Concrete Naming Certificate for MetricCompletionLeftAdjoint↑ **20203**

301.1 Concrete Naming Certificate for CFun↑ **20206**

301.2 Concrete Naming Certificate for CounterfactualOutcome↑ **20209**

301.3 Concrete Naming Certificate for FiniteSupportRemoval↑ **20212**

301.4 Concrete Naming Certificate for CauchyRemainderEstimate↑ **20215**

301.5 Concrete Naming Certificate for CauchyTailEnvelopeComparison↑ **20225**

301.6 Concrete Naming Certificate for BishopLocatedCompleteness↑ **20228**

301.7 Concrete Naming Certificate for CauchyPrincipalFilter↑ **20231**

301.8 Concrete Naming Certificate for KochCurve↑ **20234**

301.9 Concrete Naming Certificate for ChannelTerminationHaltingBoundary↑ **20237**

302.0 Concrete Naming Certificate for CauchyCompletionUniversalArrow↑ **20240**

302.1 Concrete Naming Certificate for CellularCarrierBHistEquivalence↑ **20243**

302.2 Concrete Naming Certificate for RegularSequenceBishop↑ **20246**

3023 Concrete Naming Certificate for ModulusOfUniformConvergence↑ **20249**

3024 Concrete Naming Certificate for DyadicBisectionSchedule↑ **20252**

3025 Concrete Naming Certificate for RegularCauchyEnclosureFilter↑ **20255**

3026 Concrete Naming Certificate for MorreySpace↑ **20258**

3027 Concrete Naming Certificate for CompactUniformRadiusChoice↑ **20261**

3028 Concrete Naming Certificate for AsymptoticEquicontinuity↑ **20264**

3029 Concrete Naming Certificate for ProkhorovTightProbability↑ **20267**

3030 Concrete Naming Certificate for CauchyTailEstimate↑ **20270**

3031 Concrete Naming Certificate for Circle↑ **20273**

3032 Concrete Naming Certificate for HyperbolicSelfCenterReading↑ **20277**

3033 Concrete Naming Certificate for ClosedIntervalNesting↑ **20280**

3034 Concrete Naming Certificate for HyperconvexMetric↑ **20283**

3035 Concrete Naming Certificate for SummableSequence↑ **20286**

3036 Concrete Naming Certificate for CompactLebesgueNumber↑ **20289**

3037 Concrete Naming Certificate for CompactPolishSpace↑ **20293**

3038 Concrete Naming Certificate for PartitionCoreIncidence↑ **20296**

3039 Concrete Naming Certificate for BaireModulus↑ **20299**

3040 Concrete Naming Certificate for CompactMetricSequentialNet↑ **20302**

3041 Concrete Naming Certificate for BorelHierarchy↑ **20305**

3042 Concrete Naming Certificate for RegularSequenceLocalization↑ **20308**

3043 Concrete Naming Certificate for LocallyUniformConvergence↑ **20311**

3044 Concrete Naming Certificate for BishopMetricCompletion↑ **20314**

3045 Concrete Naming Certificate for LocatedCompactness↑ **20317**

3046 Concrete Naming Certificate for FourierTransform↑ **20320**

3047 Concrete Naming Certificate for CauchyCompletionAdjunctionTriangle↑ **20323**

3048 Concrete Naming Certificate for MonotoneCauchyCriterion↑ **20326**

3049 A Concrete Naming Certificate for RegularCauchyDifferenceBound↑ **20329**

3050 A Concrete Naming Certificate for LocallyCompactHausdorff↑ **20332**

3051 A Concrete Naming Certificate for GleasonCover↑ **20335**

3052 A Concrete Naming Certificate for HeineBorelNetCompactness↑ **20337**

3053 A Concrete Naming Certificate for HausdorffCompletionUnit↑ **20340**

3054 A Concrete Naming Certificate for CauchyCriterionForSeries↑ **20343**

3055 A Concrete Naming Certificate for HyperbolicHarmonicBoundaryMeasure↑ **20346**

3056 A Concrete Naming Certificate for SupportNerveCensusLiftedBetti↑ **20349**

3057 A Concrete Naming Certificate for EquicontinuousFamily↑ **20352**

3058 A Concrete Naming Certificate for BishopRealComparisonModulus↑ **20355**

3059 A Concrete Naming Certificate for ArchimedeanCauchyCompleteField↑ **20358**

3060 A Concrete Naming Certificate for ProductUniformSpace↑ **20361**

3061 A Concrete Naming Certificate for ConstructiveRealLine↑ **20364**

3062 A Concrete Naming Certificate for MontelSpace↑ **20367**

3063 A Concrete Naming Certificate for BaireGenericResidual↑ **20370**

3064 A Concrete Naming Certificate for CountableMetricCompletion↑ **20373**

3065 Cauchy Sequential Continuity NameCert Construction **20376**

3066 A Concrete Naming Certificate for ContinuousSupNormCompletion↑ **20379**

3067 A Concrete Naming Certificate for CofinalSequenceSelector↑ **20382**

3068 A Concrete Naming Certificate for SupportNerveCensus↑ **20385**

3069 A Concrete Naming Certificate for RegularCauchyLocatedLimit↑ **20388**

3070 A Concrete Naming Certificate for CauchyFilterBasisCompletion↑ **20391**

3071 A Concrete Naming Certificate for HorosphereFlowLedger↑ **20394**

3072 A Concrete Naming Certificate for LocatedCauchyFilterCompletion↑ **20397**

3073 A Concrete Naming Certificate for RegularCauchyTailAgreement↑ **20400**

3074 A Concrete Naming Certificate for BoundedCauchyFilterCompletion↑ **20403**

3075 A Concrete Naming Certificate for CompleteTotallyBoundedCompactMetric† **20406**

3076 A Concrete Naming Certificate for SequentiallyCompleteUniformSpace† **20410**

3077 A Concrete Naming Certificate for AIConsciousnessSelfProxyLedger† **20413**

3078 A Concrete Naming Certificate for BishopRegularReal† **20416**

3079 A Concrete Naming Certificate for CodonLiftOrbitClassifier† **20419**

3080 A Concrete Naming Certificate for RegularCauchyWindowCofinality† **20422**

3081 A Concrete Naming Certificate for MetaCICDiamondResidualBudget† **20426**

3082 A Concrete Naming Certificate for WLPOBoundary† **20438**

3083 A Concrete Naming Certificate for LocatedClosedBall† **20441**

3084 A Concrete Naming Certificate for BishopUniformCompletionTheorem† **20444**

3085 A Concrete Naming Certificate for CantorPairing† **20447**

3086 A Concrete Naming Certificate for SeparatedCauchyNetCompletion† **20449**

3087 A Concrete Naming Certificate for RoleSeparationLedger† **20453**

3088 A Concrete Naming Certificate for PrecompactUniform† **20456**

3089 A Concrete Naming Certificate for RegularCauchyOrder† **20459**

3090 A Concrete Naming Certificate for LocatedCauchyRepresentation† **20462**

3091 A Concrete Naming Certificate for BishopCauchyModulusExtraction† **20466**

3092 A Concrete Naming Certificate for HausdorffUniformity† **20469**

3093 A Concrete Naming Certificate for CauchyCutComparison† **20472**

3094 A Concrete Naming Certificate for FiniteMarkovPartition† **20475**

3095 A Concrete Naming Certificate for RegularCauchyPositivePart† **20478**

3096 A Concrete Naming Certificate for Equicontinuity† **20481**

3097 A Concrete Naming Certificate for ChoquetGame† **20500**

3098 A Concrete Naming Certificate for LocatedCoveringNumber† **20503**

3099 A Concrete Naming Certificate for SupportNerveRealization† **20506**

3100 A Concrete Naming Certificate for CauchyDiagonalSelection† **20509**

310A Concrete Naming Certificate for ApproachSpace↑ **20520**

310B Concrete Naming Certificate for MetaCICConfluenceDiamondBoundary↑ **20523**

310C Concrete Naming Certificate for ?? **20526**

310D Concrete Naming Certificate for KuratowskiMeasureNoncompactness↑ **20529**

310E Concrete Naming Certificate for ?? **20532**

310F Concrete Naming Certificate for CauchyCompletenessTheorem↑ **20535**

310G Concrete Naming Certificate for CauchyModulusMajorant↑ **20538**

310H Concrete Naming Certificate for ?? **20540**

310I Concrete Naming Certificate for MetricSeparationReflector↑ **20544**

311A Concrete Naming Certificate for ?? **20547**

311B Concrete Naming Certificate for ?? **20551**

311C Concrete Naming Certificate for ConstructiveCompletionEquivalence↑ **20554**

311D Concrete Naming Certificate for EffectiveModulus↑ **20557**

311E Concrete Naming Certificate for ?? **20560**

311F Concrete Naming Certificate for EffectiveCauchySequenceSpace↑ **20563**

311G Concrete Naming Certificate for FiniteLowerBoundFold↑ **20566**

311H Concrete Naming Certificate for MetaCICDecidableConversionFrontier↑ **20569**

311I Concrete Naming Certificate for MetaCICClosednessDecisionBoundary↑ **20573**

311J Concrete Naming Certificate for EffectiveCauchySpace↑ **20576**

312A Concrete Naming Certificate for RegularCauchyRegularization↑ **20580**

312B Concrete Naming Certificate for ReflexiveBanach↑ **20583**

312C Concrete Naming Certificate for RegularCauchyFilterSpace↑ **20586**

312D Concrete Naming Certificate for CauchyRealLimit↑ **20589**

312E Concrete Naming Certificate for PolishBaireFixedPoint↑ **20592**

312F Concrete Naming Certificate for CompactModulusSelector↑ **20595**

312G Concrete Naming Certificate for TypeCheckingClassifier↑ **20598**

312.7 Concrete Naming Certificate for MengerNobelingEmbedding↑ **20601**

312.8 Concrete Naming Certificate for UniformSpaceCompletionFilter↑ **20604**

312.9 Concrete Naming Certificate for RegularCauchyFilterCompletion↑ **20607**

313.0 Concrete Naming Certificate for CauchyFilterRegularization↑ **20610**

313.1 Concrete Naming Certificate for RealInterval↑ **20613**

313.2 Concrete Naming Certificate for SequentialSpace↑ **20622**

313.3 Concrete Naming Certificate for MetaCICConversionEndpointSeal↑ **20625**

313.4 Concrete Naming Certificate for BishopLocatedCompletionComparison↑ **20628**

313.5 Concrete Naming Certificate for RiemannSumGauge↑ **20631**

313.6 Concrete Naming Certificate for RegularCauchyTransform↑ **20634**

313.7 Concrete Naming Certificate for PoincareDiskDynamics↑ **20637**

313.8 Concrete Naming Certificate for ProbeBundleFiniteMinimumFold↑ **20640**

313.9 Concrete Naming Certificate for HausdorffDimension↑ **20643**

314.0 Concrete Naming Certificate for EffectiveLimitOperator↑ **20646**

314.1 Concrete Naming Certificate for LocatedCauchyFilterLimit↑ **20649**

314.2 Concrete Naming Certificate for PseudometricCompletionUniversalProperty↑ **20652**

314.3 Concrete Naming Certificate for BaireCategoryChoiceFree↑ **20657**

314.4 Concrete Naming Certificate for RiemannIntegrabilityModulus↑ **20661**

314.5 Concrete Naming Certificate for SequentialRealCompletion↑ **20664**

314.6 Concrete Naming Certificate for MetaCICLocalJoinLedger↑ **20667**

314.7 Concrete Naming Certificate for HemicompactSpace↑ **20671**

314.8 Concrete Naming Certificate for DistanceFunctionCompletion↑ **20674**

314.9 Concrete Naming Certificate for MinimalTriggerLiftBundle↑ **20677**

315.0 Concrete Naming Certificate for BishopUniformContinuityModulus↑ **20680**

315.1 Concrete Naming Certificate for CompactContinuousInverse↑ **20683**

315.2 Concrete Naming Certificate for HausdorffRealCompletion↑ **20686**

3153 Concrete Naming Certificate for NeighborhoodFilter† **20689**

3154 Concrete Naming Certificate for HyperbolicBusemannCocycle† **20691**

3155 Concrete Naming Certificate for BishopRealConstruction† **20693**

3156 Concrete Naming Certificate for CauchyIntegralFormula† **20697**

3157 Concrete Naming Certificate for MinimalTriggerHomologyCore† **20700**

3158 Concrete Naming Certificate for CompactOpenTopology† **20703**

3159 Concrete Naming Certificate for PointwiseStreamMetric† **20706**

3160 Concrete Naming Certificate for RegularCauchyTailFilterCofinality† **20721**

3161 Concrete Naming Certificate for BishopRealPrecision† **20724**

3162 Concrete Naming Certificate for HyperbolicVisualShadowMetric† **20727**

3163 Concrete Naming Certificate for LocatedCauchyExtraction† **20730**

3163. Located Cauchy extraction carrier 20730

3164 Concrete Naming Certificate for BanachMazurGame† **20733**

3165 Concrete Naming Certificate for EffectiveMetricSpace† **20736**

3166 Concrete Naming Certificate for ComputablePolishSpace† **20738**

3167 Concrete Naming Certificate for LusinApproximation† **20742**

3168 Concrete Naming Certificate for StreamNameModulus† **20745**

3169 Concrete Naming Certificate for FiniteObservationTraceFilter† **20749**

3170 Concrete Naming Certificate for LocatedCompletion† **20752**

3171 Concrete Naming Certificate for SeparatedReflection† **20755**

3172 Concrete Naming Certificate for ConstructiveLipschitzFlow† **20758**

3173 Concrete Naming Certificate for LebesgueDifferentiation† **20761**

3174 Concrete Naming Certificate for HyperbolicObserverShadowMetric† **20764**

3175 Concrete Naming Certificate for RegularDoubleCauchy† **20767**

3176 Concrete Naming Certificate for DyadicIntervalNewtonBisection† **20772**

3177 Concrete Naming Certificate for KadecKleeProperty† **20775**

3178 Concrete Naming Certificate for RegularCauchyUniformModulus† **20778**

3179 A Concrete Naming Certificate for SpectralRadiusFormula↑ **20786**

3180 A Concrete Naming Certificate for NestedIntervalsTheorem↑ **20789**

3181 A Concrete Naming Certificate for CauchyRealSpread↑ **20791**

3182 A Concrete Naming Certificate for CompactImageClosed↑ **20797**

3183 A Concrete Naming Certificate for FarEndDiagram↑ **20800**

3184 A Concrete Naming Certificate for ScottDomain↑ **20803**

3185 A Concrete Naming Certificate for FarEndDiagrammatics↑ **20806**

3186 A Concrete Naming Certificate for PolishBasis↑ **20809**

3187 A Concrete Naming Certificate for HyperbolicObserverExpansion↑ **20812**

3188 A Concrete Naming Certificate for Met↑ **20815**

3189 A Concrete Naming Certificate for MetricCompletionKleisli↑ **20818**

3190 A Concrete Naming Certificate for LocatedRegularRealCut↑ **20821**

3191 A Concrete Naming Certificate for BoundaryOrbitInventoryPolynomial↑ **20824**

3192 A Concrete Naming Certificate for PrimitiveRecursive↑ **20827**

3193 A Concrete Naming Certificate for CauchyAcceleration↑ **20830**

3194 A Concrete Naming Certificate for BrouwerMonotoneSequence↑ **20835**

3195 A Concrete Naming Certificate for CauchySequenceCriterion↑ **20838**

3196 A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction↑ **20841**

3197 A Concrete Naming Certificate for FrinkMetrization↑ **20846**

3198 A Concrete Naming Certificate for NestedShrinkInterval↑ **20849**

3199 A Concrete Naming Certificate for RegularCauchyUniformContinuity↑ **20855**

3200 A Concrete Naming Certificate for HenstockKurzweil↑ **20859**

3201 A Concrete Naming Certificate for HyperbolicVisualBoundary↑ **20863**

3202 A Concrete Naming Certificate for BishopCompleteReal↑ **20866**

3203 A Concrete Naming Certificate for RatCauchyGapWitness↑ **20869**

3204 A Concrete Naming Certificate for RieszRepresentationFiniteLattice↑ **20872**

3205 A Concrete Naming Certificate for HyperbolicPhaseObserverExpansion$^{\uparrow}$ **20875**

3206 A Concrete Naming Certificate for NowhereDense$^{\uparrow}$ **20878**

3207 A Concrete Naming Certificate for SeparableReal$^{\uparrow}$ **20880**

3208 A Concrete Naming Certificate for BanachFixedPointModulus$^{\uparrow}$ **20882**

3209 A Concrete Naming Certificate for SpreadSpace$^{\uparrow}$ **20885**

3210 A Concrete Naming Certificate for UniformCauchyEmbedding$^{\uparrow}$ **20888**

3211 A Concrete Naming Certificate for DecimalStream$^{\uparrow}$ **20891**

3212 A Concrete Naming Certificate for MinimalTriggerOrbitClassification$^{\uparrow}$ **20894**

3213 A Concrete Naming Certificate for Portmanteau$^{\uparrow}$ **20897**

3214 A Concrete Naming Certificate for MonotoneClass$^{\uparrow}$ **20900**

3215 A Concrete Naming Certificate for FanCompact$^{\uparrow}$ **20903**

3216 A Concrete Naming Certificate for BoundedNormalizationCandidate$^{\uparrow}$ **20906**

3217 A Concrete Naming Certificate for ConstructiveBaireThread$^{\uparrow}$ **20909**

3218 A Concrete Naming Certificate for Mollifier$^{\uparrow}$ **20912**

3219 Arzela–Ascoli Family Compactness NameCert Construction **20915**

3220 SR Route Dependency Audit NameCert Construction **20918**

3221 A Concrete Naming Certificate for SeparatedReflectionUniversal$^{\uparrow}$ **20921**

3222 A Concrete Naming Certificate for RationalIntervalLocator$^{\uparrow}$ **20924**

3223 A Concrete Naming Certificate for HyperbolicGeodesicLamination$^{\uparrow}$ **20929**

3224 A Concrete Naming Certificate for BesovSpace$^{\uparrow}$ **20932**

3225 A Concrete Naming Certificate for LebesgueCoveringDimension$^{\uparrow}$ **20935**

3226 A Concrete Naming Certificate for LagrangeInterpolation$^{\uparrow}$ **20938**

3227 A Concrete Naming Certificate for RieszFischer$^{\uparrow}$ **20941**

3228 A Concrete Naming Certificate for RationalFastLimit$^{\uparrow}$ **20943**

3229 A Concrete Naming Certificate for RegularCauchyCommonRefinement$^{\uparrow}$ **20946**

3230 A Concrete Naming Certificate for OlivierTheorem$^{\uparrow}$ **20949**

3231 Concrete Naming Certificate for JacobiIteration↑ **20952**

3232 Concrete Naming Certificate for ApophaticBoundarySocketStack↑ **20955**

3233 Concrete Naming Certificate for DiniMonotoneCompactUniform↑ **20957**

3234 Concrete Naming Certificate for PartitionCoreIncidenceCertificate↑ **20961**

3235 Concrete Naming Certificate for CauchyMajorantSequence↑ **20964**

3236 Concrete Naming Certificate for MetricBaireSpace↑ **20967**

3237 Concrete Naming Certificate for DNAWindowSixTriggerThreshold↑ **20969**

3238 Concrete Naming Certificate for MetricCompletionKleisli↑ **20972**

3239 Concrete Naming Certificate for LocatedRegularRealCut↑ **20975**

3240 Concrete Naming Certificate for BoundaryOrbitInventoryPolynomial↑ **20978**

3241 Concrete Naming Certificate for PrimitiveRecursive↑ **20981**

3242 Concrete Naming Certificate for CauchyAcceleration↑ **20984**

3243 Concrete Naming Certificate for BrouwerMonotoneSequence↑ **20989**

3244 Concrete Naming Certificate for CauchySequenceCriterion↑ **20992**

3245 Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction↑ **20995**

3246 Concrete Naming Certificate for FrinkMetrization↑ **21000**

3247 Concrete Naming Certificate for NestedShrinkInterval↑ **21003**

3248 Concrete Naming Certificate for AdaptiveCauchyModulus↑ **21009**

3249 Concrete Naming Certificate for ClosedSet↑ **Metric Boundary** **21012**

3250 Concrete Naming Certificate for Met↑ **21016**

3251 Concrete Naming Certificate for NashEquilibrium↑ **21019**

3252 Concrete Naming Certificate for CauchyFilterSubbasis↑ **21022**

3253 Concrete Naming Certificate for BishopCompleteReal↑ **21029**

3254 Cauchy compact extraction NameCert construction **21032**

3255 Concrete Naming Certificate for CompactOpenUniformConvergence↑ **21035**

3256 Concrete Naming Certificate for DefectFlowPartitionSkeleton↑ **21038**

3257 A Concrete Naming Certificate for RegularCauchyChoiceFreeCompletion$^{\uparrow}$ **21041**

3258 A Concrete Naming Certificate for MetaCICParallelDiamondFrontier$^{\uparrow}$ **21050**

3259 A Concrete Naming Certificate for HausdorffCompletionFilterSeparation$^{\uparrow}$ **21151**

3260 A Concrete Naming Certificate for QRDecomposition$^{\uparrow}$ **21154**

3261 A Concrete Naming Certificate for RegularCauchyDiagonalSeal$^{\uparrow}$ **21157**

3262 Modulus of Total Boundedness NameCert Construction **21160**

3263 A Concrete Naming Certificate for PolishCauchyBasis$^{\uparrow}$ **21163**

3264 A Concrete Naming Certificate for SymmetryRestorationStabilizer$^{\uparrow}$ **21166**

3265 A Concrete Naming Certificate for TaggedRealPartition$^{\uparrow}$ **21169**

3266 A Concrete Naming Certificate for FiniteFanBarModulus$^{\uparrow}$ **21172**

3267 A Concrete Naming Certificate for EdelsteinFixedPoint$^{\uparrow}$ **21175**

3268 A Concrete Naming Certificate for CompletionLadder$^{\uparrow}$ **21178**

3269 A Concrete Naming Certificate for LocatedFormalBar$^{\uparrow}$ **21181**

3270 A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$ **21183**

3271 A Concrete Naming Certificate for RealLimitStability$^{\uparrow}$ **21185**

3272 A Concrete Naming Certificate for DecimalIntervalReal$^{\uparrow}$ **21188**

3273 A Concrete Naming Certificate for AttouchWetsTopology$^{\uparrow}$ **21192**

3274 A Concrete Naming Certificate for CantorIntersectionMetric$^{\uparrow}$ **21195**

3275 A Concrete Naming Certificate for TightRealInterval$^{\uparrow}$ **21198**

3276 A Concrete Naming Certificate for CompactUniformModulusTransfer$^{\uparrow}$ **21201**

3277 A Concrete Naming Certificate for RealIntervalApartness$^{\uparrow}$ **21204**

3278 A Concrete Naming Certificate for PartitionBasinPolynomial$^{\uparrow}$ **21207**

3279 A Concrete Naming Certificate for DistanceToSet$^{\uparrow}$ **21210**

3280 A Concrete Naming Certificate for JordanCurve$^{\uparrow}$ **21213**

3281 A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ **21216**

3282 A Concrete Naming Certificate for PointedMetricSpace$^{\uparrow}$ **21219**

3283 Concrete Naming Certificate for PointedCompleteMetricSpace↑ **21222**

3284 Concrete Naming Certificate for IntervalConnectedness↑ **21226**

3285 Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility↑ **21229**

3286 Concrete Naming Certificate for ComputableCompactMetric↑ **21233**

3287 Concrete Naming Certificate for RegularCauchyHausdorffReflection↑ **21236**

3288 Concrete Naming Certificate for WeakLawLargeNumbers↑ **21240**

3289 Concrete Naming Certificate for LocatedSubset↑ **21243**

3290 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **21246**

3291 Concrete Naming Certificate for CauchyApproximationTree↑ **21251**

3292 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **21254**

3293 Concrete Naming Certificate for SeparableSpace↑ **21257**

3294 Concrete Naming Certificate for FormalBallDomain↑ **21260**

3295 Concrete Naming Certificate for DedekindCutStream↑ **21264**

3296 Concrete Naming Certificate for StokesTheorem↑ **21268**

3297 Concrete Naming Certificate for ConstructiveImplicitFunction↑ **21271**

3298 Concrete Naming Certificate for MetricCompletionExistence↑ **21274**

3299 Concrete Naming Certificate for FiniteLimitSeal↑ **21282**

3300 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **21288**

3301 Concrete Naming Certificate for HoeffdingFiniteSum↑ **21291**

3302 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **21294**

3303 Concrete Naming Certificate for CauchyCompletionRoute↑ **21297**

3304 Concrete Naming Certificate for MetaCICProofSearchFrontier↑ **21300**

3305 Concrete Naming Certificate for DyadicIntervalHull↑ **21303**

3306 Concrete Naming Certificate for CellularCarrierSwitchCost↑ **21306**

3307 Concrete Naming Certificate for UniformlyContinuousSequence↑ **21309**

3308 Concrete Naming Certificate for LocatedIntervalHalving↑ **21316**

3309 A Concrete Naming Certificate for BishopLocatedRoot$^{\uparrow}$ **21319**

3310 A Concrete Naming Certificate for RealCompletenessWitness$^{\uparrow}$ **21322**

3311 A Concrete Naming Certificate for SmoothManifold$^{\uparrow}$ **21325**

3312 A Concrete Naming Certificate for MetaCICNormalEndpointBudget$^{\uparrow}$ **21328**

3313 A Concrete Naming Certificate for PolishCauchyBasis$^{\uparrow}$ **21331**

3314 A Concrete Naming Certificate for SymmetryRestorationStabilizer$^{\uparrow}$ **21334**

3315 A Concrete Naming Certificate for TaggedRealPartition$^{\uparrow}$ **21337**

3316 A Concrete Naming Certificate for FiniteFanBarModulus$^{\uparrow}$ **21340**

3317 A Concrete Naming Certificate for EdelsteinFixedPoint$^{\uparrow}$ **21343**

3318 A Concrete Naming Certificate for CompletionLadder$^{\uparrow}$ **21346**

3319 A Concrete Naming Certificate for LocatedFormalBar$^{\uparrow}$ **21349**

3320 A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$ **21351**

3321 A Concrete Naming Certificate for RealLimitStability$^{\uparrow}$ **21353**

3322 A Concrete Naming Certificate for DecimalIntervalReal$^{\uparrow}$ **21356**

3323 A Concrete Naming Certificate for CantorIntersectionMetric$^{\uparrow}$ **21360**

3324 A Concrete Naming Certificate for AttouchWetsTopology$^{\uparrow}$ **21363**

3325 A Concrete Naming Certificate for DistanceToSet$^{\uparrow}$ **21366**

3326 A Concrete Naming Certificate for JordanCurve$^{\uparrow}$ **21369**

3327 A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ **21372**

3328 A Concrete Naming Certificate for PointedMetricSpace$^{\uparrow}$ **21375**

3329 A Concrete Naming Certificate for IntervalConnectedness$^{\uparrow}$ **21378**

3330 A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ **21381**

3331 A Concrete Naming Certificate for RegularCauchyHausdorffReflection$^{\uparrow}$ **21385**

3332 A Concrete Naming Certificate for TightRealInterval$^{\uparrow}$ **21389**

3333 A Concrete Naming Certificate for CompactUniformModulusTransfer$^{\uparrow}$ **21392**

3334 A Concrete Naming Certificate for LocatedSubset$^{\uparrow}$ **21395**

3335 Concrete Naming Certificate for WeakLawLargeNumbers↑ **21398**

3336 Concrete Naming Certificate for CauchyApproximationTree↑ **21401**

3337 Concrete Naming Certificate for UrysohnMetrization↑ **21404**

3338 Concrete Naming Certificate for TZeroReflection↑ **21407**

3339 Concrete Naming Certificate for MetricEmbedding↑ **21410**

3340 Concrete Naming Certificate for AffineContraction↑ **21413**

3341 Concrete Naming Certificate for AlgebraicExpressionLedger↑ **21416**

3342 Concrete Naming Certificate for MetricClosure↑ **21419**

3343 Concrete Naming Certificate for CauchyCriterionEquivalence↑ **21422**

3344 Concrete Naming Certificate for MetaCICClosurePreservationWitness↑ **21425**

3345 Concrete Naming Certificate for CauchyCondensationRealSeries↑ **21428**

3346 Concrete Naming Certificate for VariationOfConstants↑ **21431**

3347 Concrete Naming Certificate for CompletionUniformFunctorialEnvelope↑ **21434**

3348 Concrete Naming Certificate for DarbouxProperty↑ **21437**

3349 Concrete Naming Certificate for LocatedCompleteMetric↑ **21441**

3350 Concrete Naming Certificate for CauchyCompletionBoundary↑ **21444**

3351 Concrete Naming Certificate for MetricCompletionDenseEmbedding↑ **21447**

3352 Concrete Naming Certificate for ClosedPiDomainSubjectReduction↑ **21450**

3353 Concrete Naming Certificate for FiniteTraceInductionGap↑ **21453**

3354 Concrete Naming Certificate for OvertReal↑ **21455**

3355 Concrete Naming Certificate for CertificateMachine↑ **21458**

3356 Concrete Naming Certificate for GeneratingFunction↑ **21462**

3357 Concrete Naming Certificate for MetricIdentification↑ **21465**

3358 Concrete Naming Certificate for EquicontinuityFiniteNetSelector↑ **21468**

3359 Concrete Naming Certificate for RealBolzano↑ **21471**

3360 Concrete Naming Certificate for CofinalFilterBase↑ **21476**

3361 Concrete Naming Certificate for AbsolutelyContinuousFunction↑ **21479**

3362 Concrete Naming Certificate for EberleinSmulian↑ **21483**

3363 Concrete Naming Certificate for SorgenfreyLine↑ **21486**

3364 Concrete Naming Certificate for CauchyTailDiagonal↑ **21489**

3365 Concrete Naming Certificate for SeparatedLimitUniqueness↑ **21492**

3366 Concrete Naming Certificate for CertificateMachineBranch↑ **21495**

3367 Concrete Naming Certificate for SylvesterInertia↑ **21498**

3368 Concrete Naming Certificate for FundamentalGroup↑ **21501**

3369 Concrete Naming Certificate for PartitionOfUnity↑ **21504**

3370 Concrete Naming Certificate for Bornology↑ **21507**

3371 Concrete Naming Certificate for MetricAnnulus↑ **21512**

3372 Concrete Naming Certificate for RealArchimedeanProperty↑ **21514**

3373 Concrete Naming Certificate for LocatedMonotoneConvergence↑ **21517**

3374 Concrete Naming Certificate for BaireSteinhaus↑ **21521**

3375 Concrete Naming Certificate for BanachSteinhaus↑ **21524**

3376 Concrete Naming Certificate for CauchyCompletionReflection↑ **21527**

3377 Concrete Naming Certificate for CauchyCompletionSeparatedReflection↑ **21531**

3378 Concrete Naming Certificate for CauchyRealArchimedean↑ **21534**

3379 Concrete Naming Certificate for CauchyCompletionOperator↑ **21536**

3380 Concrete Naming Certificate for RealCauchyCompleteness↑ **21547**

3381 Concrete Naming Certificate for ClosedLamDomainSubjectReduction↑ **21551**

3382 Concrete Naming Certificate for CauchyCompletenessCriterion↑ **21554**

3383 Concrete Naming Certificate for TailModulusLimitHandoff↑ **21557**

3384 Concrete Naming Certificate for BorelSet↑ **21559**

3385 Concrete Naming Certificate for OuterMeasure↑ **21562**

3386 Concrete Naming Certificate for LocallyConvexSpace↑ **21565**

3387 A Concrete Naming Certificate for ForwardSocketLedger↑ **21576**

3388 A Concrete Naming Certificate for SuccessiveApproximation↑ **21579**

3389 A Concrete Naming Certificate for RealCauchyDenseEmbedding↑ **21581**

3390 A Concrete Naming Certificate for RegularCauchyUniqueness↑ **21584**

3391 A Concrete Naming Certificate for DyadicMeshComposition↑ **21589**

3392 A Concrete Naming Certificate for RegularCauchyLimitOperator↑ **21592**

3393 A Concrete Naming Certificate for DnaWindowSixBoundaryTile↑ **21595**

3394 A Concrete Naming Certificate for LocatedRealIntermediateValue↑ **21598**

3395 A Concrete Naming Certificate for LocatedRealLine↑ **21601**

3396 A Concrete Naming Certificate for LocatedUpperBound↑ **21604**

3397 A Concrete Naming Certificate for UniformEmbeddingCompletion↑ **21609**

3398 A Concrete Naming Certificate for RegularCauchyCompletionReflector↑ **21612**

3399 A Concrete Naming Certificate for SchwartzFunction↑ **21616**

3399.0 Carrier definition 21616

3399.1 The certificate 21617

3400 A Concrete Naming Certificate for AsymptoticEquivalence↑ **21620**

3401 A Concrete Naming Certificate for RieszLemma↑ **21623**

3402 A Concrete Naming Certificate for SimplicialApproximation↑ **21626**

3403 A Concrete Naming Certificate for MooreSmithNet↑ **21629**

3404 A Concrete Naming Certificate for BishopCompletionCriterion↑ **21632**

3405 A Concrete Naming Certificate for CauchyInterleavingCriterion↑ **21634**

3406 A Concrete Naming Certificate for LocatedRealModulus↑ **21637**

3407 A Concrete Naming Certificate for MonotoneBoundedSequence↑ **21640**

3408 A Concrete Naming Certificate for AiryFunction↑ **21645**

3409 A Concrete Naming Certificate for CauchyLocatedRealEquivalence↑ **21648**

3410 A Concrete Naming Certificate for CauchyRealLocator↑ **21651**

3411 A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction↑ **21655**

3412 Concrete Naming Certificate for CauchySequenceAlgebra↑ **21660**

3413 Concrete Naming Certificate for Cauchy↑ **21666**

3414 Concrete Naming Certificate for FiniteSimplicialChain↑ **21676**

3415 Concrete Naming Certificate for RealModulusCriterion↑ **21678**

3416 Concrete Naming Certificate for CauchyFilterCriterion↑ **21680**

3417 Concrete Naming Certificate for UniformConvergence↑ **21683**

3418 Concrete Naming Certificate for BishopApproximationSequence↑ **21686**

3419 Concrete Naming Certificate for LaxMilgram↑ **21689**

3420 Concrete Naming Certificate for BanachAlaoglu↑ **21692**

3421 Concrete Naming Certificate for MonotoneCauchySequence↑ **21695**

3422 Concrete Naming Certificate for MetaCICCandidateSetSNHandoff↑ **21698**

3423 Concrete Naming Certificate for FilterConvergence↑ **21702**

3424 Concrete Naming Certificate for BishopCompleteMetric↑ **21706**

3425 Concrete Naming Certificate for FredholmAlternative↑ **21709**

3426 Concrete Naming Certificate for CousinCover↑ **21712**

3427 Concrete Naming Certificate for RealUniformTailComparison↑ **21715**

3428 Concrete Naming Certificate for CauchyChainCriterion↑ **21718**

3429 Concrete Naming Certificate for ClosedBoundedIntervalDarbouxPartition↑ **21720**

3430 Concrete Naming Certificate for FiniteCoverRefinement↑ **21723**

3431 Concrete Naming Certificate for LocatedIntervalCompactness↑ **21726**

3432 Concrete Naming Certificate for LimitComparisonTest↑ **21728**

3433 Concrete Naming Certificate for DenjoyIntegral↑ **21731**

3434 Concrete Naming Certificate for DecidableBetaFrontier↑ **21734**

3435 Concrete Naming Certificate for HardyInequality↑ **21738**

3436 Concrete Naming Certificate for CommutativeCStar↑ **21741**

3437 Concrete Naming Certificate for MetaCICCriticalPairLedger↑ **21744**

3438 Concrete Naming Certificate for RankNullity↑ **21747**

3439 Concrete Naming Certificate for BoundedNormalConversionBridge↑ **21750**

3440 Concrete Naming Certificate for DyadicComparisonPrinciple↑ **21753**

3441 Concrete Naming Certificate for CauchyQuotientCriterion↑ **21756**

3442 Concrete Naming Certificate for PicardFixedPointModulus↑ **21759**

3443 Concrete Naming Certificate for RectifiableCurve↑ **21762**

3444 Concrete Naming Certificate for HeineCantor↑ **21764**

3445 Concrete Naming Certificate for RealUniformSpace↑ **21768**

3446 Concrete Naming Certificate for BishopRealApartnessTopology↑ **21771**

3447 Concrete Naming Certificate for FatouLemma↑ **21774**

3448 Concrete Naming Certificate for RadonNikodym↑ **21777**

3449 Concrete Naming Certificate for GaussBonnetBoundary↑ **21780**

3450 Concrete Naming Certificate for ForceMetaCICSameSchema↑ **21783**

3451 Concrete Naming Certificate for RealBoundedSequence↑ **21786**

3452 Concrete Naming Certificate for RegularCauchyCompletionCriterion↑ **21790**

3453 Concrete Naming Certificate for RegularCauchyCompletionModulusRealizer↑ **21796**

3454 Concrete Naming Certificate for ConstructiveDiniModulus↑ **21799**

3455 Concrete Naming Certificate for PoincareDiskPhaseTransport↑ **21802**

3456 Concrete Naming Certificate for MaschkeTheorem↑ **21805**

3457 Concrete Naming Certificate for MaschkeProjector↑ **21808**

3458 Concrete Naming Certificate for NestedCauchyRealSection↑ **21811**

3459 Concrete Naming Certificate for RealUniformCauchyName↑ **21814**

3460 Concrete Naming Certificate for ProofObligationGap↑ **21817**

3461 Concrete Naming Certificate for MeasurableFunction↑ **21820**

3462 Concrete Naming Certificate for MooreSmithCauchyNetFilterBridge↑ **21823**

3463 Concrete Naming Certificate for PuncturedCubeSkeleton↑ **21826**

3464 Constructive Closed Metric Ball Certificate for ClosedMetricBall$^{\uparrow}$ **21829**

3465 Concrete Naming Certificate for BrouwerContinuityPrinciple$^{\uparrow}$ **21832**

3466 Concrete Naming Certificate for CauchyHadamard$^{\uparrow}$ **21835**

3467 Concrete Naming Certificate for DyadicSubsequence$^{\uparrow}$ **21838**

3468 Concrete Naming Certificate for RegularCauchyShiftEquivalence$^{\uparrow}$ **21841**

3469 Concrete Naming Certificate for DiagonalCauchySubsequence$^{\uparrow}$ **21844**

3470 Concrete Naming Certificate for SixBoundaryTileCoverage$^{\uparrow}$ **21847**

3471 Concrete Naming Certificate for HeineBorelFiniteNet$^{\uparrow}$ **21850**

3472 Concrete Naming Certificate for HilbertBasis$^{\uparrow}$ **21857**

3473 Concrete Naming Certificate for CantorBernstein$^{\uparrow}$ **21859**

3474 Concrete Naming Certificate for HyperbolicMetric$^{\uparrow}$ **21862**

3475 Concrete Naming Certificate for DenseIsometryExtension$^{\uparrow}$ **21865**

3476 Concrete Naming Certificate for IsometricCompletionExtension$^{\uparrow}$ **21868**

3477 Concrete Naming Certificate for CauchyCompletionProductPreservation$^{\uparrow}$ **21871**

3478 Concrete Naming Certificate for WindowCoverageTriggerAlgebra$^{\uparrow}$ **21874**

3479 Concrete Naming Certificate for BorelFunction$^{\uparrow}$ **21878**

3480 Concrete Naming Certificate for ChebyshevPolynomial$^{\uparrow}$ **21881**

3481 Concrete Naming Certificate for CellularWindowTrace$^{\uparrow}$ **21883**

3482 Concrete Naming Certificate for WeakKonigBoundary$^{\uparrow}$ **21885**

3483 Concrete Naming Certificate for CauchyCutBridge$^{\uparrow}$ **21888**

3484 Concrete Naming Certificate for BishopCompletionTheorem$^{\uparrow}$ **21891**

3485 Concrete Naming Certificate for BaireUltrametric$^{\uparrow}$ **21894**

3486 Concrete Naming Certificate for HyperbolicExpansionLedger$^{\uparrow}$ **21899**

3487 Concrete Naming Certificate for LiouvilleLedgerConservation$^{\uparrow}$ **21902**

3488 Concrete Naming Certificate for BishopLocatedRealCompletion$^{\uparrow}$ **21905**

3489 Concrete Naming Certificate for LocatedDyadicCut$^{\uparrow}$ **21909**

3490 Concrete Naming Certificate for KirszbraunExtension↑ **21912**

3491 Concrete Naming Certificate for HyperbolicGeodesicBoundary↑ **21915**

3492 Concrete Naming Certificate for UniformClosure↑ **21918**

3493 Concrete Naming Certificate for CauchyFilterLocatedRealExtraction↑ **21921**

3494 Concrete Naming Certificate for SignedDyadicExpansion↑ **21923**

3495 Concrete Naming Certificate for FrechetSpace↑ **21927**

3496 Concrete Naming Certificate for FinitePartitionMesh↑ **21931**

3497 Concrete Naming Certificate for ChoiceWindowSequence↑ **21934**

3498 Concrete Naming Certificate for ProximinalSet↑ **21937**

3499 Concrete Naming Certificate for CompletionUnitComparison↑ **21940**

3500 Concrete Naming Certificate for PoincareBoundaryFixedTransport↑ **21943**

3501 Concrete Naming Certificate for MetricCompletionNaturality↑ **21946**

3502 Concrete Naming Certificate for AsymptoticSequence↑ **21949**

3503 Concrete Naming Certificate for FiniteSubcoverCriterion↑ **21952**

3504 Concrete Naming Certificate for RieszMarkov↑ **21955**

3504.1 Compact positive functional carrier 21955

3504.2 Certificate obligations 21956

3505 Constructive Apartness Certificate for MetricApartness↑ **21957**

3506 Concrete Naming Certificate for ContractiveCauchyOrbit↑ **21960**

3507 Concrete Naming Certificate for CompletionSubsetClosure↑ **21964**

3508 Concrete Naming Certificate for DnaBoundaryTriggerClosure↑ **21967**

3509 Concrete Naming Certificate for Ceva↑ **21970**

3510 Concrete Naming Certificate for BishopCauchyApproximation↑ **21973**

3511 Concrete Naming Certificate for KakutaniFixedPoint↑ **21975**

3512 Concrete Naming Certificate for RatCauchyCompletion↑ **21978**

3513 Concrete Naming Certificate for DyadicDensityRealEmbeddingComparison↑ **21981**

3514 Concrete Naming Certificate for CauchyInterleavingComparison↑ **21985**

3515 A Concrete Naming Certificate for RegularCauchySubsequenceStability$^{\uparrow}$ **21988**

3516 A Concrete Naming Certificate for SpeckerSequenceBoundary$^{\uparrow}$ **21991**

3517 A Concrete Naming Certificate for BishopOrderCompletion$^{\uparrow}$ **21995**

3518 A Concrete Naming Certificate for EffectiveCauchyModulus$^{\uparrow}$ **21998**

3519 A Concrete Naming Certificate for MetacicRedexChoiceFrontier$^{\uparrow}$ **22001**

3520 A Concrete Naming Certificate for MetricCompletionFiniteCoproduct$^{\uparrow}$ **22005**

3521 A Concrete Naming Certificate for LatentCompletion$^{\uparrow}$ **22008**

3522 A Concrete Naming Certificate for BoundedFunctionFamily$^{\uparrow}$ **22011**

3523 A Concrete Naming Certificate for CauchyCompletionApartness$^{\uparrow}$ **22014**

3524 A Concrete Naming Certificate for BishopRegularCutEquivalence$^{\uparrow}$ **22017**

3525 A Concrete Naming Certificate for DyadicToleranceLadder$^{\uparrow}$ **22020**

3526 A Concrete Naming Certificate for NestedClosedBall$^{\uparrow}$ **22023**

3527 A Concrete Naming Certificate for RealBoundedInterval$^{\uparrow}$ **22030**

3528 A Concrete Naming Certificate for StieltjesMeasureBoundary$^{\uparrow}$ **22033**

3529 A Concrete Naming Certificate for BanachContraction$^{\uparrow}$ **22036**

3530 A Concrete Naming Certificate for PeanoExistence$^{\uparrow}$ **22040**

3531 A Concrete Naming Certificate for LocatedCompletionFunctor$^{\uparrow}$ **22043**

3532 A Concrete Naming Certificate for RegSeqRatLocatedModulus$^{\uparrow}$ **22046**

3533 A Concrete Naming Certificate for MorseTheory$^{\uparrow}$ **22049**

3534 A Concrete Naming Certificate for HyperbolicHorocycle$^{\uparrow}$ **22052**

3535 A Concrete Naming Certificate for CompleteMetricBaireSurface$^{\uparrow}$ **22054**

3536 A Concrete Naming Certificate for BaireSpaceCompleteness$^{\uparrow}$ **22057**

3537 A Concrete Naming Certificate for MeasurableSpace$^{\uparrow}$ **22060**

3538 A Concrete Naming Certificate for LocatedHeineBorelSubcover$^{\uparrow}$ **22063**

3539 A Concrete Naming Certificate for MetaCICTypedNormalizationAudit$^{\uparrow}$ **22067**

3540 A Concrete Naming Certificate for LocatedCompletionCriterion$^{\uparrow}$ **22071**

3541 Concrete Naming Certificate for RegularCauchySequenceSpace↑ **22075**

3542 Concrete Naming Certificate for CauchyCompletionDensitySource↑ **22079**

3543 Concrete Naming Certificate for SeparatedCompletionReflection↑ **22082**

3544 Concrete Naming Certificate for ConvergentNet↑ **22085**

3545 Concrete Naming Certificate for MedianClosure↑ **22089**

3546 Concrete Naming Certificate for RegularCauchyReal↑ **22092**

3547 Concrete Naming Certificate for GDeltaSet↑ **22096**

3548 Concrete Naming Certificate for ClosedBallCompactness↑ **22099**

3549 Concrete Naming Certificate for CompactSourceTriangleEstimate↑ **22102**

3550 Concrete Naming Certificate for RealLineConnected↑ **22105**

3551 Concrete Naming Certificate for CauchyProductMetric↑ **22108**

3552 Concrete Naming Certificate for RegularCauchySpaceMorphism↑ **22111**

3553 Concrete Naming Certificate for MetricCompletionComparison↑ **22114**

3554 Concrete Naming Certificate for BaireTwoFunction↑ **22118**

3555 Concrete Naming Certificate for AlexanderDualTriggerAlgebra↑ **22121**

3556 Concrete Naming Certificate for MooreSmithCauchySubnetCriterion↑ **22124**

3557 Concrete Naming Certificate for MooreSmithCompletenessCriterion↑ **22127**

3558 Concrete Naming Certificate for ReassignmentSupportCompression↑ **22130**

3559 Concrete Naming Certificate for TopologicalClosure↑ **22133**

3560 Concrete Naming Certificate for UniformSpaceCompletion↑ **22136**

3561 Concrete Naming Certificate for RegularCauchySubspace↑ **22139**

3562 Concrete Naming Certificate for MetricReflectionSandwich↑ **22142**

3563 Concrete Naming Certificate for ConvexConjugate↑ **22145**

3564 Concrete Naming Certificate for LocatedApartnessRealBridge↑ **22148**

3565 Concrete Naming Certificate for LocatedApartnessCompletion↑ **22153**

3566 Concrete Naming Certificate for NormalizationCandidateSocket↑ **22156**

3567 A Concrete Naming Certificate for MetaCICSnCandidateInterface↑ **22164**

3568 A Concrete Naming Certificate for ArzelaAscoliFiniteModulus↑ **22167**

3569 A Concrete Naming Certificate for CompactTotallyBounded↑ **22184**

3570 A Concrete Naming Certificate for BaireCategoryTheorem↑ **22187**

3571 A Concrete Naming Certificate for BaireCategoryArgument↑ **22191**

3572 A Concrete Naming Certificate for RationalBracketRefinement↑ **22194**

3573 A Concrete Naming Certificate for LocatedCompactIntervalOscillationBound↑ **22197**

3574 A Concrete Naming Certificate for CharacterTable↑ **22200**

3575 A Concrete Naming Certificate for DenjoyWolffBoundaryDynamics↑ **22202**

3576 A Concrete Naming Certificate for BishopCauchyCriterion↑ **22205**

3577 A Concrete Naming Certificate for BishopCauchyModulusEquivalence↑ **22208**

3578 A Concrete Naming Certificate for BurnsideOrbitCount↑ **22210**

3579 A Concrete Naming Certificate for CauchyCompletenessWitness↑ **22213**

3580 A Concrete Naming Certificate for CauchyCompletenessWitnessObligation↑ **22216**

3581 A Concrete Naming Certificate for HardyLittlewoodMaximal↑ **22218**

3582 A Concrete Naming Certificate for RationalRootInterval↑ **22221**

3583 A Concrete Naming Certificate for EnergyBiasedTriggerReliability↑ **22224**

3584 A Concrete Naming Certificate for MertensCauchyProductTheorem↑ **22227**

3585 A Concrete Naming Certificate for BishopCompletionUniversalComposition↑ **22231**

3586 A Concrete Naming Certificate for FiniteModulusCompactness↑ **22234**

3587 A Concrete Naming Certificate for LocatedCauchyNet↑ **22237**

3588 A Concrete Naming Certificate for BishopUniformCauchyNet↑ **22240**

3589 A Concrete Naming Certificate for AlternativeCodeReassignment↑ **22243**

3590 A Concrete Naming Certificate for UniformReflectionKernel↑ **22246**

3591 A Concrete Naming Certificate for PhaseLedgerHorocycle↑ **22248**

3592 A Concrete Naming Certificate for MooreSpace↑ **22251**

3593 Concrete Naming Certificate for VariationMeasure↑ **22253**

3594 Concrete Naming Certificate for CauchyContinuityCalibration↑ **22256**

3595 Concrete Naming Certificate for MetaCICLocalDiamondWitness↑ **22259**

3596 Concrete Naming Certificate for TangentCone↑ **22262**

3597 Concrete Naming Certificate for WeakTopology↑ **22264**

3598 Concrete Naming Certificate for CondensationTailSelector↑ **22267**

3599 Concrete Naming Certificate for NestedIntervalCauchy↑ **22270**

3600 Concrete Naming Certificate for TypeLike↑ **22273**

3601 Concrete Naming Certificate for CauchyCompleteApartnessSpace↑ **22277**

3602 Concrete Naming Certificate for RegularCauchyTailFunctor↑ **22280**

3603 Concrete Naming Certificate for OperatorNorm↑ **22283**

3604 Concrete Naming Certificate for ContinuousImageCompact↑ **22286**

3605 Concrete Naming Certificate for CompactMetricContinuousImage↑ **22289**

3606 Concrete Naming Certificate for FiniteCoverLebesgueNumber↑ **22292**

3607 Concrete Naming Certificate for ContractionPrinciple↑ **22295**

3608 Concrete Naming Certificate for CauchyProductConvergence↑ **22298**

3609 Concrete Naming Certificate for CauchyProductMertens↑ **22301**

3610 Concrete Naming Certificate for KreinMilmanFiniteExtreme↑ **22303**

3611 Concrete Naming Certificate for CFunCarrier↑ **22306**

3612 Concrete Naming Certificate for SturmRootCount↑ **22309**

3613 Concrete Naming Certificate for UniformCauchyBicompletion↑ **22312**

3614 Concrete Naming Certificate for DyadicCauchyApproximation↑ **22315**

3615 Concrete Naming Certificate for SequentialCauchyRegularization↑ **22318**

3616 Concrete Naming Certificate for MetaCICDecidableTypingFrontier↑ **22321**

3617 Concrete Naming Certificate for DenseEmbedding↑ **22325**

3618 Concrete Naming Certificate for UniformCauchyCompletion↑ **22328**

3619 A Concrete Naming Certificate for ArchimedeanCompletion↑ **22331**

3620 A Concrete Naming Certificate for RieszSpace↑ **22334**

3621 A Concrete Naming Certificate for ArzelaAscoliFiniteWindowDiagonalBridge↑ **22337**

3622 A Concrete Naming Certificate for BishopCauchySequence↑ **22339**

3623 A Concrete Naming Certificate for MeirKeelerContraction↑ **22342**

3624 A Concrete Naming Certificate for IntervalHalvingModulus↑ **22345**

3625 A Concrete Naming Certificate for PostMedianGateShell↑ **22348**

3626 A Concrete Naming Certificate for MetaCICDependentCodomainStabilityBoundary↑ **22351**

3627 A Concrete Naming Certificate for DyadicSandwich↑ **22354**

3628 A Concrete Naming Certificate for CauchyCompletionRecursor↑ **22357**

3629 A Concrete Naming Certificate for DedekindReal↑ **22360**

3630 A Concrete Naming Certificate for ObservableDiameterDecay↑ **22363**

3631 A Concrete Naming Certificate for FastCauchyNormalForm↑ **22366**

3632 A Concrete Naming Certificate for Calculus↑ **22369**

3633 A Concrete Naming Certificate for ChainRule↑ **22408**

3634 A Concrete Naming Certificate for FrechetDerivative↑ **22411**

3635 A Concrete Naming Certificate for BoundedRealIntervalPartition↑ **22414**

3636 A Concrete Naming Certificate for CauchySequenceOrder↑ **22418**

3637 A Concrete Naming Certificate for NoetherianRing↑ **22422**

3637.0 Carrier definition 22422

3638 A Concrete Naming Certificate for LeviCivitaConnection↑ **22424**

3639 A Concrete Naming Certificate for BishopCauchyCompletionFunctor↑ **22427**

3640 A Concrete Naming Certificate for RealLocatedOrder↑ **22429**

3641 A Concrete Naming Certificate for RealSequenceSpace↑ **22432**

3642 A Concrete Naming Certificate for FrechetUrysohn↑ **22436**

3643 A Concrete Naming Certificate for HyperbolicBoundaryPhaseAudit↑ **22439**

3644 A Concrete Naming Certificate for SardTheoremFiniteJet↑ **22442**

3645. Concrete Naming Certificate for BusemannBoundary↑ **22445**

3646. Concrete Naming Certificate for RegularCauchyCompletionUnit↑ **22447**

3647. Concrete Naming Certificate for SobolevTrace↑ **22450**

3648. Concrete Naming Certificate for NoetherianModule↑ **22453**
3648.1. Carrier definition 22453

3649. Concrete Naming Certificate for LaplaceTransform↑ **22455**

3650. Concrete Naming Certificate for LipschitzExtension↑ **22458**

3651. Concrete Naming Certificate for AnchorSubcubeTrigger↑ **22461**

3652. Concrete Naming Certificate for HeatEquationFiniteStencil↑ **22464**

3653. Concrete Naming Certificate for HausdorffDistance↑ **22467**

3654. Concrete Naming Certificate for BoundedVariationRealFunction↑ **22469**

3655. Concrete Naming Certificate for RegularCauchyLocatedModulus↑ **22473**
3655.1. Located modulus carrier 22473

3656. Concrete Naming Certificate for CauchyCompletionUniversalReflector↑ **22476**

3657. Concrete Naming Certificate for TriggerHypergraphReliability↑ **22480**

3658. Concrete Naming Certificate for LocatedCompleteInterval↑ **22483**
3658.1. Located Complete Interval Carrier 22483

3659. Concrete Naming Certificate for IntervalCauchyCompleteness↑ **22486**
3659.1. Interval Cauchy Completeness Carrier 22486

3660. Concrete Naming Certificate for LimitPointCompact↑ **22489**

3661. Concrete Naming Certificate for ClosedNormalConsistencyWitness↑ **22492**

3662. Concrete Naming Certificate for FiniteGroupRepresentation↑ **22495**

3663. Concrete Naming Certificate for KSpace↑ **22498**

3664. Concrete Naming Certificate for CompactUniformFiniteCoverSelector↑ **22501**

3665. Concrete Naming Certificate for BishopRealCompletion↑ **22504**

3666. Concrete Naming Certificate for UniformCompletionUniversalProperty↑ **22507**

3667. Concrete Naming Certificate for CauchyCompletionStableReflection↑ **22510**

3668. Concrete Naming Certificate for SetLike↑ **22513**

3669 A Concrete Naming Certificate for SchauderBasis↑ **22552**

3670 A Concrete Naming Certificate for RationalCauchyCompletion↑ **22555**

3671 A Concrete Naming Certificate for CompactMetricClosedBall↑ **22558**

3672 A Concrete Naming Certificate for RegularCauchyProductCriterion↑ **22561**

3673 A Concrete Naming Certificate for BishopMetricSpace↑ **22564**

3674 A Concrete Naming Certificate for BishopSpace↑ **22568**

3675 A Concrete Naming Certificate for RealClosure↑ **22571**

3676 A Concrete Naming Certificate for IntermediateValueTheorem↑ **22574**

3677 A Concrete Naming Certificate for FirstExitChannel↑ **22577**

3678 A Concrete Naming Certificate for BoundaryShadowTrigger↑ **22580**

3679 A Concrete Naming Certificate for WhitneyExtensionFiniteJet↑ **22582**

3680 A Concrete Naming Certificate for HausdorffCompletionKernel↑ **22585**

3681 A Concrete Naming Certificate for RadialWobbleSpectrum↑ **22588**

3682 A Concrete Naming Certificate for SchwarzPickDiskContraction↑ **22591**

3683 A Concrete Naming Certificate for PoincareTriangleDefect↑ **22594**

3684 A Concrete Naming Certificate for CauchyBoundednessCriterion↑ **22597**

3685 A Concrete Naming Certificate for LocatedMetricSpace↑ **22600**

3686 A Concrete Naming Certificate for ProperMetricSpace↑ **22603**

3687 A Concrete Naming Certificate for CompleteMetricSpace↑ **22606**

3688 A Concrete Naming Certificate for CauchyCompleteOrderedField↑ **22609**

3689 A Concrete Naming Certificate for CauchyTailDominance↑ **22619**

3690 A Concrete Naming Certificate for MonotoneCauchySubsequence↑ **22624**

3691 A Concrete Naming Certificate for CauchySequenceCompletion↑ **22627**

3692 A Concrete Naming Certificate for BoundedIntervalProjection↑ **22630**

3693 A Concrete Naming Certificate for MontelTheorem↑ **22633**

3694 A Concrete Naming Certificate for AntipodalTriggerLedger↑ **22636**

3695 A Concrete Naming Certificate for EffectiveCauchyCriterion↑ **22639**

3696 A Concrete Naming Certificate for UniformlyCauchySequence↑ **22642**

3697 A Concrete Naming Certificate for DyadicMeshScaleNormalization↑ **22645**

3698 A Concrete Naming Certificate for CompactCompletionExtensionModulus↑ **22648**

3699 A Concrete Naming Certificate for UniformlyContinuousCompletionExtension↑ **22651**

3700 A Concrete Naming Certificate for BishopMonotoneRealConvergence↑ **22654**

3701 A Concrete Naming Certificate for CauchyLimitClassifierLock↑ **22657**

3702 A Concrete Naming Certificate for DarbouxTheorem↑ **22660**

3703 A Concrete Naming Certificate for LocatedCompletionUniversalProperty↑ **22663**

3704 A Concrete Naming Certificate for BooleanTriggerLattice↑ **22666**

3705 A Concrete Naming Certificate for BishopCauchyComparison↑ **22668**

3706 A Concrete Naming Certificate for ClosedObservationTotalHost↑ **22671**

3707 A Concrete Naming Certificate for RegularCauchyAlgebraCoherence↑ **22673**

3708 A Concrete Naming Certificate for CauchySequenceAlgebra↑ **22676**

3709 A Concrete Naming Certificate for CauchySequenceProductStability↑ **22681**

3710 A Concrete Naming Certificate for BishopCauchyProductBound↑ **22684**

3711 A Concrete Naming Certificate for OresmeHarmonicDivergence↑ **22687**

3712 A Concrete Naming Certificate for Subdifferential↑ **22690**
3712.1 Subdifferential Carrier 22690
3712.2 Subdifferential Certificate 22691

3713 A Concrete Naming Certificate for ProximalMap↑ **22694**
3713.1 ProximalMap Carrier 22694
3713.2 ProximalMap Certificate 22695

3714 A Concrete Naming Certificate for TypedNormalFormDecider↑ **22698**

3715 A Concrete Naming Certificate for ResolventIdentity↑ **22701**

3716 A Concrete Naming Certificate for BoundedIntervalCauchySubsequence↑ **22704**

3717 A Concrete Naming Certificate for BishopCompletion↑ **22708**

3718 A Concrete Naming Certificate for RegularCauchyModulusExtraction↑ **22712**

371A Concrete Naming Certificate for CauchySelectionFunction$^{\uparrow}$ **22715**

372A Concrete Naming Certificate for CauchySumLimit$^{\uparrow}$ **22718**

372A Concrete Naming Certificate for CauchyProductLimit$^{\uparrow}$ **22721**

372A Concrete Naming Certificate for RealCompletenessDiagonalSelector$^{\uparrow}$ **22724**

372A Concrete Naming Certificate for LeibnizSeriesTest$^{\uparrow}$ **22727**

372A Concrete Naming Certificate for LipschitzModulus$^{\uparrow}$ **22730**

372A Concrete Naming Certificate for ModulusCauchyCompletion$^{\uparrow}$ **22733**

372A Concrete Naming Certificate for UniformlyLocatedFiniteCover$^{\uparrow}$ **22735**

372Cauchy Condensation Theorem NameCert Construction **22737**

372A Concrete Naming Certificate for Sylow$^{\uparrow}$ **22740**

372A Concrete Naming Certificate for SylowTheorem$^{\uparrow}$ **22744**

373A Concrete Naming Certificate for BernsteinPolynomial$^{\uparrow}$ **22747**

373A Concrete Naming Certificate for SequentialCauchySelector$^{\uparrow}$ **22749**

373A Concrete Naming Certificate for HyperbolicBusemannBoundary$^{\uparrow}$ **22753**

373A Concrete Naming Certificate for BooleanAlgebra$^{\uparrow}$ **22756**

373A Concrete Naming Certificate for CoverageRankAlgebra$^{\uparrow}$ **22763**

373A Concrete Naming Certificate for CauchySequenceOrderCompleteness$^{\uparrow}$ **22766**

373A Concrete Naming Certificate for HilleYosida$^{\uparrow}$ **22769**

373A Concrete Naming Certificate for CompactUniformOscillation$^{\uparrow}$ **22772**

373A Concrete Naming Certificate for RegularCauchyDiagonalWindow$^{\uparrow}$ **22775**

373A Concrete Naming Certificate for RegularCauchyDiagonalModulusSelection$^{\uparrow}$ **22778**

374A Concrete Naming Certificate for DunfordFunctionalCalculus$^{\uparrow}$ **22781**

374A Concrete Naming Certificate for UniformCompletionCommonWindow$^{\uparrow}$ **22784**

374A Concrete Naming Certificate for FiniteMetricSpace$^{\uparrow}$ **22787**

374A Concrete Naming Certificate for RegularCauchyTailBound$^{\uparrow}$ **22790**

374A Concrete Naming Certificate for HochsterNerveBettiReduction$^{\uparrow}$ **22793**

3745 Concrete Naming Certificate for BrouwerDegree↑ **22796**

3746 Concrete Naming Certificate for Metacompact↑ **22799**

3747 Concrete Naming Certificate for FiniteRecognizerCarrierObstruction↑ **22802**

3748 Concrete Naming Certificate for RegularCauchyCanonicalModulus↑ **22806**

3749 Concrete Naming Certificate for RegularCauchyCompletionStability↑ **22809**

3750 Concrete Naming Certificate for BoundedMonotoneSubsequence↑ **22812**

3751 Concrete Naming Certificate for DistributionDerivative↑ **22815**

3752 Concrete Naming Certificate for NetClusterPoint↑ **22818**

3753 Concrete Naming Certificate for GreedySpectralClosure↑ **22821**

3754 Concrete Naming Certificate for CauchyCompleteOrderedFieldUniqueness↑ **22824**

3755 Concrete Naming Certificate for MetricCompletionFunctoriality↑ **22828**

3756 Concrete Naming Certificate for NaturalDeduction↑ **22831**

3757 Concrete Naming Certificate for RegularCauchyFiniteSum↑ **22834**

3758 Concrete Naming Certificate for FiniteSeriesPermutation↑ **22837**

3759 Concrete Naming Certificate for EpsilonDeltaLimit↑ **22840**

3760 Concrete Naming Certificate for EpsilonDeltaContinuity↑ **22843**

3761 Concrete Naming Certificate for PackingNumber↑ **22847**

3762 Concrete Naming Certificate for LipschitzFunction↑ **22850**

3763 Concrete Naming Certificate for CantorTernarySet↑ **22853**

3764 Concrete Naming Certificate for CyclicTagSystem↑ **22856**

3765 Concrete Naming Certificate for HausdorffContent↑ **22858**

3766 Concrete Naming Certificate for CauchyTailFilterBornology↑ **22861**

3767 Concrete Naming Certificate for SequentialCompletenessCriterion↑ **22864**

3768 Concrete Naming Certificate for RegularModulus↑ **22866**

3769 Concrete Naming Certificate for CompactLocatedIntervalSubcover↑ **22868**

3770 Concrete Naming Certificate for RealAbsoluteValue↑ **22871**

377A Concrete Naming Certificate for CUNSpectralAntenna↑ **22874**

377B Concrete Naming Certificate for FrechetCauchyCriterion↑ **22877**

377C Concrete Naming Certificate for LocatedCutReal↑ **22880**

377D Concrete Naming Certificate for DyadicStepApproximation↑ **22883**

377E Concrete Naming Certificate for HausdorffUniformSpace↑ **22886**

377F Concrete Naming Certificate for SecantMethod↑ **22889**

377G Concrete Naming Certificate for CauchyCompletionBind↑ **22892**

377H Concrete Naming Certificate for GlobalFiberSpectra↑ **22895**

377J Concrete Naming Certificate for MetaCICNormalizationAuditPacket↑ **22898**

378A Concrete Naming Certificate for StepFunction↑ **22902**

378B Concrete Naming Certificate for SternBrocotContinuedFractionBridge↑ **22905**

378C Concrete Naming Certificate for RegularCauchyDiagonalization↑ **22909**

378D Concrete Naming Certificate for TotallyBoundedClosure↑ **22913**

378E Concrete Naming Certificate for UniformModulusComposition↑ **22916**

378F Concrete Naming Certificate for ConstructiveCauchyInterval↑ **22919**

378G Concrete Naming Certificate for CauchyCompletionDenseReflector↑ **22922**

378H Concrete Naming Certificate for LocallyCompactMetric↑ **22925**

378J Concrete Naming Certificate for LocallyCompactPolish↑ **22928**

379A Concrete Naming Certificate for SequentialRealCompactness↑ **22932**

379B Concrete Naming Certificate for LawlikeSequence↑ **22935**

379C Concrete Naming Certificate for MetaCICConfluenceSubstitutionBoundary↑ **22938**

379D Concrete Naming Certificate for CauchyCompletionComonad↑ **22942**

379E Concrete Naming Certificate for RealDiagonalNonSurjection↑ **22946**

379F Concrete Naming Certificate for CauchyNetCompletion↑ **22955**

379G Concrete Naming Certificate for RealTightness↑ **22964**

379H Concrete Naming Certificate for SchwarzLemma↑ **22967**

3797 A Concrete Naming Certificate for CauchyProductCompletion↑ **22970**

3798 A Concrete Naming Certificate for CantorSpaceCompactness↑ **22973**

3799 A Concrete Naming Certificate for CauchySchlomilchCondensation↑ **22976**

3800 A Concrete Naming Certificate for FiniteOscillationPartition↑ **22979**

3801 A Concrete Naming Certificate for TriggerBlockerDuality↑ **22984**

3802 A Concrete Naming Certificate for FiniteRelation↑ **22987**

3802. Finite-relation source 22987

3803 A Concrete Naming Certificate for RegularCauchyTailEnvelope↑ **22989**

3804 A Concrete Naming Certificate for VariationNorm↑ **22992**

3805 A Concrete Naming Certificate for RegularCauchyEntourageBasis↑ **22995**

3806 A Concrete Naming Certificate for TriggerEnergyPartition↑ **22998**

3807 A Concrete Naming Certificate for ContinuousSupNorm↑ **23001**

3808 A Concrete Naming Certificate for BoundedResidualSubstitutionFrontier↑ **23005**

3809 A Concrete Naming Certificate for LocatedRealContinuity↑ **23008**

3810 A Concrete Naming Certificate for CellularSubstrateClassifierBoundary↑ **23011**

3811 A Concrete Naming Certificate for HadamardProductSeries↑ **23014**

3812 A Concrete Naming Certificate for LipschitzContinuityHandoff↑ **23017**

3813 A Concrete Naming Certificate for HyperbolicBoundaryShadow↑ **23020**

3814 A Concrete Naming Certificate for LocatedUniformSpace↑ **23023**

3815 A Concrete Naming Certificate for ClosedIntervalBisection↑ **23026**

3816 A Concrete Naming Certificate for RealSpectrum↑ **23030**

3817 A Concrete Naming Certificate for BanachLattice↑ **23033**

3818 A Concrete Naming Certificate for HeronSquareRoot↑ **23036**

3819 A Concrete Naming Certificate for ContractiveMap↑ **23051**

3820 A Concrete Naming Certificate for ClosedBoundedSequence↑ **23055**

3821 A Concrete Naming Certificate for CompactContinuityRadiusChoice↑ **23058**

3822 A Concrete Naming Certificate for DuhamelPrinciple↑ **23061**

3823 Concrete Naming Certificate for LocatedSequenceLimit$^{\uparrow}$ **23064**

3824 Concrete Naming Certificate for DedekindGapLocatedCut$^{\uparrow}$ **23067**

3825 Concrete Naming Certificate for FastCauchyCriterion$^{\uparrow}$ **23070**

3826 Concrete Naming Certificate for CauchyCompletionKleisliCategory$^{\uparrow}$ **23073**

3827 Concrete Naming Certificate for MetaCICConfluenceResidualChecker$^{\uparrow}$ **23076**

3828 Concrete Naming Certificate for DarbouxContinuity$^{\uparrow}$ **23079**

3829 Concrete Naming Certificate for ConcaveModulus$^{\uparrow}$ **23083**

3830 Concrete Naming Certificate for SignedDigitIntervalReal$^{\uparrow}$ **23086**

3831 Concrete Naming Certificate for CayleyTheorem$^{\uparrow}$ **23090**

3832 Concrete Naming Certificate for CauchyMeshRefinement$^{\uparrow}$ **23093**

3833 Concrete Naming Certificate for MonotoneConvergenceReal$^{\uparrow}$ **23096**

3834 Concrete Naming Certificate for CauchyCompletionMonadLaw$^{\uparrow}$ **23099**

3835 Concrete Naming Certificate for CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$ **23112**

3836 Concrete Naming Certificate for MetricCompletionAdjunction$^{\uparrow}$ **23116**

3837 Concrete Naming Certificate for GeneratorClosureRobustness$^{\uparrow}$ **23119**

3838 Concrete Naming Certificate for ComputableUniformContinuity$^{\uparrow}$ **23124**

3839 Concrete Naming Certificate for RegularCauchyCompletionSeparation$^{\uparrow}$ **23127**

3840 Concrete Naming Certificate for WRRHotspotCube$^{\uparrow}$ **23130**

3841 Concrete Naming Certificate for NestedCauchyIntervalSelection$^{\uparrow}$ **23133**

3842 Concrete Naming Certificate for BoundaryPressureLedger$^{\uparrow}$ **23136**

3843 Concrete Naming Certificate for CompletionAdjoint$^{\uparrow}$ **23139**

3844 Concrete Naming Certificate for UniformEquicontinuityModulus$^{\uparrow}$ **23142**

3845 Concrete Naming Certificate for LimitSpace$^{\uparrow}$ **23145**

3846 Concrete Naming Certificate for FrechetKolmogorovCompactness$^{\uparrow}$ **23148**

3847 Concrete Naming Certificate for RationalBracketing$^{\uparrow}$ **23160**

3848 Concrete Naming Certificate for RealModulusExtraction$^{\uparrow}$ **23163**

3849 A Concrete Naming Certificate for CauchyStability$^{\uparrow}$ **23166**

3850 A Concrete Naming Certificate for CauchySequenceTailFilter$^{\uparrow}$ **23169**

3851 A Concrete Naming Certificate for CauchySequenceTailFilterCompletion$^{\uparrow}$ **23172**

3852 A Concrete Naming Certificate for LocalizedCauchyRate$^{\uparrow}$ **23176**

3853 A Concrete Naming Certificate for FiniteOscillationUniformModulus$^{\uparrow}$ **23179**

3854 A Concrete Naming Certificate for FubiniFiniteRectangle$^{\uparrow}$ **23182**

3855 A Concrete Naming Certificate for TonelliFiniteRectangle$^{\uparrow}$ **23185**

3856 A Concrete Naming Certificate for HyperbolicGeodesicFlow$^{\uparrow}$ **23187**

3857 A Concrete Naming Certificate for LocatedSequence$^{\uparrow}$ **23190**

3858 A Concrete Naming Certificate for CauchyModulusExtractor$^{\uparrow}$ **23193**

3859 A Concrete Naming Certificate for RealArithmeticProgression$^{\uparrow}$ **23197**

3860 A Concrete Naming Certificate for RegularCauchyReflectionLedger$^{\uparrow}$ **23200**

3861 A Concrete Naming Certificate for TubeLemma$^{\uparrow}$ **23203**

3862 A Concrete Naming Certificate for FilterLimitBasis$^{\uparrow}$ **23205**

3863 A Concrete Naming Certificate for PerfectSet$^{\uparrow}$ **23216**

3864 A Concrete Naming Certificate for RegularCauchyBinaryInterleaving$^{\uparrow}$ **23219**

3865 A Concrete Naming Certificate for ?? **23222**

3866 A Concrete Naming Certificate for HyperbolicVisualMetric$^{\uparrow}$ **23225**

3867 A Concrete Naming Certificate for MetricCauchyNetCluster$^{\uparrow}$ **23228**

3868 A Concrete Naming Certificate for CompactIntervalModulusBridge$^{\uparrow}$ **23231**

3869 A Concrete Naming Certificate for LocatedRealBisectionSearch$^{\uparrow}$ **23234**

3870 A Concrete Naming Certificate for ConditionalTriggerEnsemble$^{\uparrow}$ **23237**

3871 A Concrete Naming Certificate for NormalizationEvidenceBudget$^{\uparrow}$ **23240**

3872 A Concrete Naming Certificate for RationalIntervalSubdivision$^{\uparrow}$ **23243**

3873 A Concrete Naming Certificate for LocatedIntervalSubdivision$^{\uparrow}$ **23246**

3874 A Concrete Naming Certificate for UniformSequentialLimit$^{\uparrow}$ **23249**

3875 A Concrete Naming Certificate for CauchyFilterConvergence↑ **23251**

3876 A Concrete Naming Certificate for UniformConvergenceCauchyCriterion↑ **23254**

3877 A Concrete Naming Certificate for BoxDimension↑ **23257**

3878 A Concrete Naming Certificate for MinkowskiDimension↑ **23260**

3879 A Concrete Naming Certificate for CompleteUltrametric↑ **23263**

3880 A Concrete Naming Certificate for PainleveKuratowskiConvergence↑ **23265**

3881 A Concrete Naming Certificate for HaarWavelet↑ **23268**

3882 A Concrete Naming Certificate for CauchyRegularCompletionAdjunction↑ **23271**

3883 A Concrete Naming Certificate for QuasiIsometry↑ **23274**

3884 A Concrete Naming Certificate for DyadicClosedIntervalEndpointGrid↑ **23277**

3885 A Concrete Naming Certificate for CauchyFilterCompletionCriterion↑ **23280**

3886 A Concrete Naming Certificate for ConstructiveUniformLimit↑ **23284**

3887 A Concrete Naming Certificate for CompactnessUniformContinuityDiameterDecay↑ **23287**

3888 A Concrete Naming Certificate for MetricCompletionIdempotence↑ **23290**

3889 A Concrete Naming Certificate for MetaCICFuelTrace↑ **23294**

3890 A Concrete Naming Certificate for CauchyStructure↑ **23297**

3891 A Concrete Naming Certificate for UniformContinuityComposition↑ **23300**

3892 A Concrete Naming Certificate for HadamardThreeCircle↑ **23305**

3893 A Concrete Naming Certificate for HadamardThreeLines↑ **23309**

3894 A Concrete Naming Certificate for PhragmenLindelof↑ **23312**

3895 A Concrete Naming Certificate for CauchyLocatedApartness↑ **23315**

3896 A Concrete Naming Certificate for RiemannLebesgueLemma↑ **23318**

3897 A Concrete Naming Certificate for MedianClosureSpectrum↑ **23321**

3898 A Concrete Naming Certificate for DirectedFilter↑ **23324**

3899 A Concrete Naming Certificate for OscillationFunction↑ **23327**

3900 A Concrete Naming Certificate for FilterCompletionUniversalProperty↑ **23330**

390 A Concrete Naming Certificate for FiniteChoiceWindow↑ **23334**

390 A Concrete Naming Certificate for CauchySequenceField↑ **23336**

390 A Concrete Naming Certificate for BishopLocatedCompletionReflector↑ **23339**

390 A Concrete Naming Certificate for CellularOrbitSynchronization↑ **23342**

390 A Concrete Naming Certificate for FiniteDimensionalHahnBanachSeparation↑ **23345**

390 A Concrete Naming Certificate for PiecewiseLinearApproximation↑ **23348**

390 A Concrete Naming Certificate for CompactIntervalMaximum↑ **23351**

390 A Concrete Naming Certificate for FilterCompletionMonad↑ **23354**

390 A Concrete Naming Certificate for MetaCICDecidableNeutralSpine↑ **23357**

391 A Concrete Naming Certificate for YoungIntegral↑ **23360**

391 A Concrete Naming Certificate for TypeCheckingClassifierMembership↑ **23363**

391 A Concrete Naming Certificate for CauchyCompletionReflectorComparison↑ **23366**

391 A Concrete Naming Certificate for CompactRegularCauchySubsequence↑ **23368**

391 A Concrete Naming Certificate for LindelofSpace↑ **23371**

391 A Concrete Naming Certificate for LindelofCoverRefinement↑ **23374**

391 A Concrete Naming Certificate for IntervalCoverRefinementTree↑ **23377**

391 A Concrete Naming Certificate for NeumannSeries↑ **23380**

391 A Concrete Naming Certificate for CauchyWindowStabilityBridge↑ **23383**

391 A Concrete Naming Certificate for MetaCICPiAdequacyBudget↑ **23386**

392 Finite Cauchy Diagonal NameCert Construction **23389**

392 MetaCIC Confluence Residual Envelope NameCert Construction **23392**

392 A Concrete Naming Certificate for HahnBanach↑ **23398**

392 A Concrete Naming Certificate for RealIntervalCompletion↑ **23401**

392 A Concrete Naming Certificate for CompactContinuousImageTotalBounded↑ **23404**

392 A Concrete Naming Certificate for HyperbolicPoissonKernel↑ **23407**

392 A Concrete Naming Certificate for EkelandVariationalPrinciple↑ **23410**

3927 Concrete Naming Certificate for KanExtension↑ **23413**

3928 Concrete Naming Certificate for MonotoneConvergenceSequence↑ **23416**

3929 Concrete Naming Certificate for RegularCauchyPicardSeal↑ **23419**

3930 Concrete Naming Certificate for ArzelaAscoliSelection↑ **23422**

3931 Concrete Naming Certificate for ContractionMappingTheorem↑ **23425**

3932 Concrete Naming Certificate for HilbertAlexanderBlocker↑ **23428**

3933 Concrete Naming Certificate for LocatedCompactIntervalSelection↑ **23431**

3934 Concrete Naming Certificate for CompactUniformRadiusLedger↑ **23434**

3935 Concrete Naming Certificate for LowerSemicontinuityReal↑ **23437**

3936 Concrete Naming Certificate for RegularCauchyLimitSelector↑ **23440**

3937 Concrete Naming Certificate for SubordinateFiniteModulusCover↑ **23442**

3938 Concrete Naming Certificate for LocatedIntermediateValue↑ **23444**

3939 Concrete Naming Certificate for ClosedBoundedIntervalFan↑ **23447**

3940 Concrete Naming Certificate for DiagonalSequence↑ **23450**

3941 Concrete Naming Certificate for WobbleSaturationSeal↑ **23452**

3942 Concrete Naming Certificate for RegularCauchyLimitUniqueness↑ **23455**

3943 Concrete Naming Certificate for FormalBallBasis↑ **23458**

3944 Concrete Naming Certificate for RealConvergentSequence↑ **23461**

3945 Concrete Naming Certificate for LocatedCauchySubsequence↑ **23465**

3946 Concrete Naming Certificate for CompletionReflectorTriangle↑ **23468**

3947 Concrete Naming Certificate for LawvereMetricCompletion↑ **23471**

3948 Concrete Naming Certificate for HyperbolicCrossRatio↑ **23477**

3949 Concrete Naming Certificate for RegularCauchyFilterFunctor↑ **23482**

3950 Concrete Naming Certificate for ArzelaAscoliFiniteEquicontinuous↑ **23485**

3951 Concrete Naming Certificate for PoissonSummation↑ **23488**

3952 Concrete Naming Certificate for RealDecimalNormalForm↑ **23491**

3953 Concrete Naming Certificate for CauchyModulusNormalizer↑ **23494**

3954 Concrete Naming Certificate for MetaCICStrongNormalizationBoundary↑ **23497**

3955 Concrete Naming Certificate for CofinalFilter↑ **23500**

3956 Concrete Naming Certificate for RegularCauchyCompletionFunctor↑ **23503**

3957 Concrete Naming Certificate for CompactUniformFiniteNetRealization↑ **23507**

3958 Concrete Naming Certificate for MetricSequentialCompletion↑ **23510**

3959 Concrete Naming Certificate for SpectralAblationHinge↑ **23513**

3960 Concrete Naming Certificate for SierpinskiCarpet↑ **23516**

3961 Concrete Naming Certificate for SpectralAblationHingeLedger↑ **23519**

3962 Concrete Naming Certificate for BanachContractionPrinciple↑ **23523**

3963 Concrete Naming Certificate for PublicLocalReflectiveCompilation↑ **23527**

3964 Concrete Naming Certificate for MetricEntourage↑ **23530**

3965 Concrete Naming Certificate for LeastUpperBound↑ **23532**

3966 Concrete Naming Certificate for CauchyIntervalNesting↑ **23535**

3967 Concrete Naming Certificate for FormalTopology↑ **23538**

3968 Concrete Naming Certificate for FormalBallRadiusOrder↑ **23541**

3969 Concrete Naming Certificate for ClosedBoundedIntervalCompact↑ **23543**

3970 Concrete Naming Certificate for FiniteSequencePeak↑ **23547**

3971 Concrete Naming Certificate for LocatedIntervalBolzano↑ **23550**

3972 Concrete Naming Certificate for BishopRealLocatedOrder↑ **23553**

3973 Concrete Naming Certificate for NestedCompactIntersection↑ **23556**

3974 Concrete Naming Certificate for MercerTheorem↑ **23559**

3975 Concrete Naming Certificate for CauchyCompletionRegularSequence↑ **23562**

3976 Concrete Naming Certificate for CosetSpace↑ **23565**

3977 Concrete Naming Certificate for UniformCauchyNet↑ **23568**

3978 Concrete Naming Certificate for ContourResidueHandoff↑ **23571**

3979 A Concrete Naming Certificate for InitialValueProblem↑ **23574**

3980 A Concrete Naming Certificate for UniformCauchyLimit↑ **23577**

3981 A Concrete Naming Certificate for MedianDepthSealing↑ **23580**

3982 A Concrete Naming Certificate for UniformFixedPoint↑ **23583**

3983 A Concrete Naming Certificate for CauchySchwarzEquality↑ **23586**

3984 A Concrete Naming Certificate for ExtremeValueTheorem↑ **23589**

3985 A Concrete Naming Certificate for BoundedRealOscillation↑ **23592**

3986 A Concrete Naming Certificate for BishopCompactInterval↑ **23595**

3987 A Concrete Naming Certificate for MetricCompletionLeftAdjoint↑ **23599**

3988 A Concrete Naming Certificate for CFun↑ **23602**

3989 A Concrete Naming Certificate for CounterfactualOutcome↑ **23605**

3990 A Concrete Naming Certificate for FiniteSupportRemoval↑ **23608**

3991 A Concrete Naming Certificate for CauchyRemainderEstimate↑ **23611**

3992 A Concrete Naming Certificate for CauchyTailEnvelopeComparison↑ **23621**

3993 A Concrete Naming Certificate for BishopLocatedCompleteness↑ **23624**

3994 A Concrete Naming Certificate for CauchyPrincipalFilter↑ **23627**

3995 A Concrete Naming Certificate for KochCurve↑ **23630**

3996 A Concrete Naming Certificate for ChannelTerminationHaltingBoundary↑ **23633**

3997 A Concrete Naming Certificate for CauchyCompletionUniversalArrow↑ **23636**

3998 A Concrete Naming Certificate for CellularCarrierBHistEquivalence↑ **23639**

3999 A Concrete Naming Certificate for RegularSequenceBishop↑ **23642**

4000 A Concrete Naming Certificate for ModulusOfUniformConvergence↑ **23645**

4001 A Concrete Naming Certificate for DyadicBisectionSchedule↑ **23648**

4002 A Concrete Naming Certificate for DyadicMidpointRefinement↑ **23651**

4003 A Concrete Naming Certificate for RegularCauchyEnclosureFilter↑ **23654**

4004 A Concrete Naming Certificate for MorreySpace↑ **23657**

4005 A Concrete Naming Certificate for MorreyInequality↑ **23660**

4006 A Concrete Naming Certificate for CompactUniformRadiusChoice↑ **23663**

4007 A Concrete Naming Certificate for AsymptoticEquicontinuity↑ **23666**

4008 A Concrete Naming Certificate for ProkhorovTightProbability↑ **23669**

4009 A Concrete Naming Certificate for CauchyTailEstimate↑ **23672**

4010 A Concrete Naming Certificate for Circle↑ **23675**

4011 A Concrete Naming Certificate for HyperbolicSelfCenterReading↑ **23679**

4012 A Concrete Naming Certificate for ClosedIntervalNesting↑ **23682**

4013 A Concrete Naming Certificate for HyperconvexMetric↑ **23685**

4014 A Concrete Naming Certificate for SummableSequence↑ **23688**

4015 A Concrete Naming Certificate for CompactLebesgueNumber↑ **23691**

4016 A Concrete Naming Certificate for CompactPolishSpace↑ **23695**

4017 A Concrete Naming Certificate for PartitionCoreIncidence↑ **23698**

4018 A Concrete Naming Certificate for BaireModulus↑ **23701**

4019 A Concrete Naming Certificate for CompactMetricSequentialNet↑ **23704**

4020 A Concrete Naming Certificate for BorelHierarchy↑ **23707**

4021 A Concrete Naming Certificate for RegularSequenceLocalization↑ **23710**

4022 A Concrete Naming Certificate for LocallyUniformConvergence↑ **23713**

4023 A Concrete Naming Certificate for BishopMetricCompletion↑ **23716**

4024 A Concrete Naming Certificate for LocatedCompactness↑ **23719**

4025 A Concrete Naming Certificate for FourierTransform↑ **23722**

4026 A Concrete Naming Certificate for CauchyCompletionAdjunctionTriangle↑ **23725**

4027 A Concrete Naming Certificate for MonotoneCauchyCriterion↑ **23728**

4028 A Concrete Naming Certificate for RegularCauchyDifferenceBound↑ **23731**

4029 A Concrete Naming Certificate for LocallyCompactHausdorff↑ **23734**

4030 A Concrete Naming Certificate for GleasonCover↑ **23737**

403 A Concrete Naming Certificate for HeineBorelNetCompactness↑ **23739**

403 A Concrete Naming Certificate for HausdorffCompletionUnit↑ **23742**

403 A Concrete Naming Certificate for CauchyCriterionForSeries↑ **23745**

403 A Concrete Naming Certificate for HyperbolicHarmonicBoundaryMeasure↑ **23748**

403 A Concrete Naming Certificate for SupportNerveCensusLiftedBetti↑ **23751**

403 A Concrete Naming Certificate for EquicontinuousFamily↑ **23754**

403 A Concrete Naming Certificate for BishopRealComparisonModulus↑ **23757**

403 A Concrete Naming Certificate for ArchimedeanCauchyCompleteField↑ **23760**

403 A Concrete Naming Certificate for RegularCauchyDifferenceModulus↑ **23763**

404 A Concrete Naming Certificate for ProductUniformSpace↑ **23766**

404 A Concrete Naming Certificate for ConstructiveRealLine↑ **23769**

404 A Concrete Naming Certificate for MontelSpace↑ **23772**

404 A Concrete Naming Certificate for BaireGenericResidual↑ **23775**

404 A Concrete Naming Certificate for CountableMetricCompletion↑ **23778**

404 Cauchy Sequential Continuity NameCert Construction **23781**

404 A Concrete Naming Certificate for ContinuousSupNormCompletion↑ **23784**

404 A Concrete Naming Certificate for CofinalSequenceSelector↑ **23787**

404 A Concrete Naming Certificate for SupportNerveCensus↑ **23790**

404 A Concrete Naming Certificate for RegularCauchyLocatedLimit↑ **23793**

405 A Concrete Naming Certificate for CauchyFilterBasisCompletion↑ **23796**

405 A Concrete Naming Certificate for HorosphereFlowLedger↑ **23799**

405 A Concrete Naming Certificate for LocatedCauchyFilterCompletion↑ **23802**

405 A Concrete Naming Certificate for RegularCauchyTailAgreement↑ **23805**

405 A Concrete Naming Certificate for BoundedCauchyFilterCompletion↑ **23808**

405 A Concrete Naming Certificate for CompleteTotallyBoundedCompactMetric↑ **23811**

405 A Concrete Naming Certificate for SequentiallyCompleteUniformSpace↑ **23815**

4057 Concrete Naming Certificate for AIConsciousnessSelfProxyLedger† **23818**

4058 Concrete Naming Certificate for BishopRegularReal† **23821**

4059 Concrete Naming Certificate for CodonLiftOrbitClassifier† **23824**

4060 Concrete Naming Certificate for RegularCauchyWindowCofinality† **23827**

4061 Concrete Naming Certificate for MetaCICDiamondResidualBudget† **23831**

4062 Concrete Naming Certificate for WLPOBoundary† **23843**

4063 Concrete Naming Certificate for LocatedClosedBall† **23846**

4064 Concrete Naming Certificate for BishopUniformCompletionTheorem† **23849**

4065 Concrete Naming Certificate for CantorPairing† **23852**

4066 Concrete Naming Certificate for SeparatedCauchyNetCompletion† **23854**

4067 Concrete Naming Certificate for RoleSeparationLedger† **23858**

4068 Concrete Naming Certificate for PrecompactUniform† **23861**

4069 Concrete Naming Certificate for RegularCauchyOrder† **23864**

4070 Concrete Naming Certificate for LocatedCauchyRepresentation† **23867**

4071 Concrete Naming Certificate for BishopCauchyModulusExtraction† **23871**

4072 Concrete Naming Certificate for HausdorffUniformity† **23874**

4073 Concrete Naming Certificate for CauchyCutComparison† **23877**

4074 Concrete Naming Certificate for FiniteMarkovPartition† **23880**

4075 Concrete Naming Certificate for RegularCauchyPositivePart† **23883**

4076 Concrete Naming Certificate for Equicontinuity† **23886**

4077 Concrete Naming Certificate for ChoquetGame† **23905**

4078 Concrete Naming Certificate for LocatedCoveringNumber† **23908**

4079 Concrete Naming Certificate for SupportNerveRealization† **23911**

4080 Concrete Naming Certificate for CauchyDiagonalSelection† **23914**

4081 Concrete Naming Certificate for ApproachSpace† **23925**

4082 Concrete Naming Certificate for MetaCICConfluenceDiamondBoundary† **23928**

4083 Concrete Naming Certificate for ?? **23931**

4084 Concrete Naming Certificate for KuratowskiMeasureNoncompactness$^{\uparrow}$ **23934**

4085 Concrete Naming Certificate for ?? **23937**

4086 Concrete Naming Certificate for CauchyCompletenessTheorem$^{\uparrow}$ **23940**

4087 Concrete Naming Certificate for CauchyModulusMajorant$^{\uparrow}$ **23943**

4088 Concrete Naming Certificate for ?? **23945**

4089 Concrete Naming Certificate for MetricSeparationReflector$^{\uparrow}$ **23949**

4090 Concrete Naming Certificate for ?? **23952**

4091 Concrete Naming Certificate for ?? **23956**

4092 Concrete Naming Certificate for ConstructiveCompletionEquivalence$^{\uparrow}$ **23959**

4093 Concrete Naming Certificate for EffectiveModulus$^{\uparrow}$ **23962**

4094 Concrete Naming Certificate for ?? **23965**

4095 Concrete Naming Certificate for EffectiveCauchySequenceSpace$^{\uparrow}$ **23968**

4096 Concrete Naming Certificate for FiniteLowerBoundFold$^{\uparrow}$ **23971**

4097 Concrete Naming Certificate for MetaCICDecidableConversionFrontier$^{\uparrow}$ **23974**

4098 Concrete Naming Certificate for MetaCICClosednessDecisionBoundary$^{\uparrow}$ **23978**

4099 Concrete Naming Certificate for EffectiveCauchySpace$^{\uparrow}$ **23981**

4100 Concrete Naming Certificate for RegularCauchyRegularization$^{\uparrow}$ **23985**

4101 Concrete Naming Certificate for ReflexiveBanach$^{\uparrow}$ **23988**

4102 Concrete Naming Certificate for RegularCauchyFilterSpace$^{\uparrow}$ **23991**

4103 Concrete Naming Certificate for CauchyRealLimit$^{\uparrow}$ **23994**

4104 Concrete Naming Certificate for PolishBaireFixedPoint$^{\uparrow}$ **23997**

4105 Concrete Naming Certificate for CompactModulusSelector$^{\uparrow}$ **24000**

4106 Concrete Naming Certificate for TypeCheckingClassifier$^{\uparrow}$ **24003**

4107 Concrete Naming Certificate for MengerNobelingEmbedding$^{\uparrow}$ **24006**

4108 Concrete Naming Certificate for UniformSpaceCompletionFilter$^{\uparrow}$ **24009**

4109 A Concrete Naming Certificate for RegularCauchyFilterCompletion↑ **24012**

4110 A Concrete Naming Certificate for CauchyFilterRegularization↑ **24015**

4111 A Concrete Naming Certificate for RealInterval↑ **24018**

4112 A Concrete Naming Certificate for SequentialSpace↑ **24027**

4113 A Concrete Naming Certificate for MetaCICConversionEndpointSeal↑ **24030**

4114 A Concrete Naming Certificate for BishopLocatedCompletionComparison↑ **24033**

4115 A Concrete Naming Certificate for RiemannSumGauge↑ **24036**

4116 A Concrete Naming Certificate for RegularCauchyTransform↑ **24039**

4117 A Concrete Naming Certificate for PoincareDiskDynamics↑ **24042**

4118 A Concrete Naming Certificate for ProbeBundleFiniteMinimumFold↑ **24045**

4119 A Concrete Naming Certificate for HausdorffDimension↑ **24048**

4120 A Concrete Naming Certificate for EffectiveLimitOperator↑ **24051**

4121 A Concrete Naming Certificate for LocatedCauchyFilterLimit↑ **24054**

4122 A Concrete Naming Certificate for PseudometricCompletionUniversalProperty↑ **24057**

4123 A Concrete Naming Certificate for BaireCategoryChoiceFree↑ **24062**

4124 A Concrete Naming Certificate for RiemannIntegrabilityModulus↑ **24066**

4125 A Concrete Naming Certificate for SequentialRealCompletion↑ **24069**

4126 A Concrete Naming Certificate for MetaCICLocalJoinLedger↑ **24072**

4127 A Concrete Naming Certificate for HemicompactSpace↑ **24076**

4128 A Concrete Naming Certificate for DistanceFunctionCompletion↑ **24079**

4129 A Concrete Naming Certificate for MinimalTriggerLiftBundle↑ **24082**

4130 A Concrete Naming Certificate for BishopUniformContinuityModulus↑ **24085**

4131 A Concrete Naming Certificate for CompactContinuousInverse↑ **24088**

4132 A Concrete Naming Certificate for HausdorffRealCompletion↑ **24091**

4133 A Concrete Naming Certificate for NeighborhoodFilter↑ **24094**

4134 A Concrete Naming Certificate for HyperbolicBusemannCocycle↑ **24096**

4135 Concrete Naming Certificate for BishopRealConstruction† **24098**

4136 Concrete Naming Certificate for CauchyIntegralFormula† **24102**

4137 Concrete Naming Certificate for MinimalTriggerHomologyCore† **24105**

4138 Concrete Naming Certificate for CompactOpenTopology† **24108**

4139 Concrete Naming Certificate for PointwiseStreamMetric† **24111**

4140 Concrete Naming Certificate for RegularCauchyTailFilterCofinality† **24126**

4141 Concrete Naming Certificate for BishopRealPrecision† **24129**

4142 Concrete Naming Certificate for HyperbolicVisualShadowMetric† **24132**

4143 Concrete Naming Certificate for LocatedCauchyExtraction† **24135**

4143. Located Cauchy extraction carrier 24135

4144 Concrete Naming Certificate for BanachMazurGame† **24138**

4145 Concrete Naming Certificate for EffectiveMetricSpace† **24141**

4146 Concrete Naming Certificate for ComputablePolishSpace† **24143**

4147 Concrete Naming Certificate for LusinApproximation† **24147**

4148 Concrete Naming Certificate for StreamNameModulus† **24150**

4149 Concrete Naming Certificate for FiniteObservationTraceFilter† **24154**

4150 Concrete Naming Certificate for LocatedCompletion† **24157**

4151 Concrete Naming Certificate for SeparatedReflection† **24160**

4152 Concrete Naming Certificate for ConstructiveLipschitzFlow† **24163**

4153 Concrete Naming Certificate for LebesgueDifferentiation† **24166**

4154 Concrete Naming Certificate for HyperbolicObserverShadowMetric† **24169**

4155 Concrete Naming Certificate for RegularDoubleCauchy† **24172**

4156 Concrete Naming Certificate for DyadicIntervalNewtonBisection† **24177**

4157 Concrete Naming Certificate for KadecKleeProperty† **24180**

4158 Concrete Naming Certificate for RegularCauchyUniformModulus† **24183**

4159 Concrete Naming Certificate for SpectralRadiusFormula† **24191**

4160 Concrete Naming Certificate for NestedIntervalsTheorem† **24194**

4161 Concrete Naming Certificate for CauchyRealSpread↑ **24196**

4162 Concrete Naming Certificate for CompactImageClosed↑ **24202**

4163 Concrete Naming Certificate for FarEndDiagram↑ **24205**

4164 Concrete Naming Certificate for ScottDomain↑ **24208**

4165 Concrete Naming Certificate for FarEndDiagrammatics↑ **24211**

4166 Concrete Naming Certificate for PolishBasis↑ **24214**

4167 Concrete Naming Certificate for HyperbolicObserverExpansion↑ **24217**

4168 Concrete Naming Certificate for Met↑ **24220**

4169 Concrete Naming Certificate for MetricCompletionKleisli↑ **24223**

4170 Concrete Naming Certificate for LocatedRegularRealCut↑ **24226**

4171 Concrete Naming Certificate for BoundaryOrbitInventoryPolynomial↑ **24229**

4172 Concrete Naming Certificate for PrimitiveRecursive↑ **24232**

4173 Concrete Naming Certificate for CauchyAcceleration↑ **24235**

4174 Concrete Naming Certificate for BrouwerMonotoneSequence↑ **24240**

4175 Concrete Naming Certificate for CauchySequenceCriterion↑ **24243**

4176 Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction↑ **24246**

4177 Concrete Naming Certificate for FrinkMetrization↑ **24251**

4178 Concrete Naming Certificate for NestedShrinkInterval↑ **24254**

4179 Concrete Naming Certificate for RegularCauchyUniformContinuity↑ **24260**

4180 Concrete Naming Certificate for HenstockKurzweil↑ **24264**

4181 Concrete Naming Certificate for HyperbolicVisualBoundary↑ **24268**

4182 Concrete Naming Certificate for BishopCompleteReal↑ **24271**

4183 Concrete Naming Certificate for RatCauchyGapWitness↑ **24274**

4184 Concrete Naming Certificate for RieszRepresentationFiniteLattice↑ **24277**

4185 Concrete Naming Certificate for HyperbolicPhaseObserverExpansion↑ **24280**

4186 Concrete Naming Certificate for NowhereDense↑ **24283**

4187 A Concrete Naming Certificate for SeparableReal$^{\uparrow}$ **24285**

4188 A Concrete Naming Certificate for BanachFixedPointModulus$^{\uparrow}$ **24287**

4189 A Concrete Naming Certificate for SpreadSpace$^{\uparrow}$ **24290**

4190 A Concrete Naming Certificate for UniformCauchyEmbedding$^{\uparrow}$ **24293**

4191 A Concrete Naming Certificate for DecimalStream$^{\uparrow}$ **24296**

4192 A Concrete Naming Certificate for MinimalTriggerOrbitClassification$^{\uparrow}$ **24299**

4193 A Concrete Naming Certificate for Portmanteau$^{\uparrow}$ **24302**

4194 A Concrete Naming Certificate for MonotoneClass$^{\uparrow}$ **24305**

4195 A Concrete Naming Certificate for FanCompact$^{\uparrow}$ **24308**

4196 A Concrete Naming Certificate for BoundedNormalizationCandidate$^{\uparrow}$ **24311**

4197 A Concrete Naming Certificate for ConstructiveBaireThread$^{\uparrow}$ **24314**

4198 A Concrete Naming Certificate for Mollifier$^{\uparrow}$ **24317**

4199 Arzela–Ascoli Family Compactness NameCert Construction **24320**

4200 SR Route Dependency Audit NameCert Construction **24323**

4201 A Concrete Naming Certificate for SeparatedReflectionUniversal$^{\uparrow}$ **24326**

4202 A Concrete Naming Certificate for RationalIntervalLocator$^{\uparrow}$ **24329**

4203 A Concrete Naming Certificate for HyperbolicGeodesicLamination$^{\uparrow}$ **24334**

4204 A Concrete Naming Certificate for BesovSpace$^{\uparrow}$ **24337**

4205 A Concrete Naming Certificate for LebesgueCoveringDimension$^{\uparrow}$ **24340**

4206 A Concrete Naming Certificate for LagrangeInterpolation$^{\uparrow}$ **24343**

4207 A Concrete Naming Certificate for RieszFischer$^{\uparrow}$ **24346**

4208 A Concrete Naming Certificate for RationalFastLimit$^{\uparrow}$ **24348**

4209 A Concrete Naming Certificate for RegularCauchyCommonRefinement$^{\uparrow}$ **24351**

4210 A Concrete Naming Certificate for OlivierTheorem$^{\uparrow}$ **24354**

4211 A Concrete Naming Certificate for JacobiIteration$^{\uparrow}$ **24357**

4212 A Concrete Naming Certificate for ApophaticBoundarySocketStack$^{\uparrow}$ **24360**

4213 A Concrete Naming Certificate for DiniMonotoneCompactUniform$^{\uparrow}$ **24362**

4214 A Concrete Naming Certificate for PartitionCoreIncidenceCertificate$^{\uparrow}$ **24366**

4215 A Concrete Naming Certificate for CauchyMajorantSequence$^{\uparrow}$ **24369**

4216 A Concrete Naming Certificate for MetricBaireSpace$^{\uparrow}$ **24372**

4217 A Concrete Naming Certificate for DNAWindowSixTriggerThreshold$^{\uparrow}$ **24374**

4218 A Concrete Naming Certificate for MetricCompletionKleisli$^{\uparrow}$ **24377**

4219 A Concrete Naming Certificate for LocatedRegularRealCut$^{\uparrow}$ **24380**

4220 A Concrete Naming Certificate for BoundaryOrbitInventoryPolynomial$^{\uparrow}$ **24383**

4221 A Concrete Naming Certificate for PrimitiveRecursive$^{\uparrow}$ **24386**

4222 A Concrete Naming Certificate for CauchyAcceleration$^{\uparrow}$ **24389**

4223 A Concrete Naming Certificate for BrouwerMonotoneSequence$^{\uparrow}$ **24394**

4224 A Concrete Naming Certificate for CauchySequenceCriterion$^{\uparrow}$ **24397**

4225 A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$ **24400**

4226 A Concrete Naming Certificate for FrinkMetrization$^{\uparrow}$ **24405**

4227 A Concrete Naming Certificate for NestedShrinkInterval$^{\uparrow}$ **24408**

4228 A Concrete Naming Certificate for AdaptiveCauchyModulus$^{\uparrow}$ **24414**

4229 A Concrete Naming Certificate for ClosedSet$^{\uparrow}$ **Metric Boundary** **24417**

4230 A Concrete Naming Certificate for Met$^{\uparrow}$ **24421**

4231 A Concrete Naming Certificate for NashEquilibrium$^{\uparrow}$ **24424**

4232 A Concrete Naming Certificate for CauchyFilterSubbasis$^{\uparrow}$ **24427**

4233 A Concrete Naming Certificate for BishopCompleteReal$^{\uparrow}$ **24434**

4234 Cauchy compact extraction NameCert construction **24437**

4235 A Concrete Naming Certificate for CompactOpenUniformConvergence$^{\uparrow}$ **24440**

4236 A Concrete Naming Certificate for DefectFlowPartitionSkeleton$^{\uparrow}$ **24443**

4237 A Concrete Naming Certificate for RegularCauchyChoiceFreeCompletion$^{\uparrow}$ **24446**

4238 A Concrete Naming Certificate for MetaCICParallelDiamondFrontier$^{\uparrow}$ **24455**

4239 A Concrete Naming Certificate for HausdorffCompletionFilterSeparation↑ **24556**

4240 A Concrete Naming Certificate for QRDecomposition↑ **24559**

4241 A Concrete Naming Certificate for RegularCauchyDiagonalSeal↑ **24562**

4242 Modulus of Total Boundedness NameCert Construction **24565**

4243 A Concrete Naming Certificate for PolishCauchyBasis↑ **24568**

4244 A Concrete Naming Certificate for SymmetryRestorationStabilizer↑ **24571**

4245 A Concrete Naming Certificate for TaggedRealPartition↑ **24574**

4246 A Concrete Naming Certificate for FiniteFanBarModulus↑ **24577**

4247 A Concrete Naming Certificate for EdelsteinFixedPoint↑ **24580**

4248 A Concrete Naming Certificate for CompletionLadder↑ **24583**

4249 A Concrete Naming Certificate for LocatedFormalBar↑ **24586**

4250 A Concrete Naming Certificate for CauchyModulusFunctoriality↑ **24588**

4251 A Concrete Naming Certificate for RealLimitStability↑ **24590**

4252 A Concrete Naming Certificate for DecimalIntervalReal↑ **24593**

4253 A Concrete Naming Certificate for AttouchWetsTopology↑ **24597**

4254 A Concrete Naming Certificate for CantorIntersectionMetric↑ **24600**

4255 A Concrete Naming Certificate for TightRealInterval↑ **24603**

4256 A Concrete Naming Certificate for CompactUniformModulusTransfer↑ **24606**

4257 A Concrete Naming Certificate for RealIntervalApartness↑ **24609**

4258 A Concrete Naming Certificate for PartitionBasinPolynomial↑ **24612**

4259 A Concrete Naming Certificate for DistanceToSet↑ **24615**

4260 A Concrete Naming Certificate for JordanCurve↑ **24618**

4261 A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary↑ **24621**

4262 A Concrete Naming Certificate for PointedMetricSpace↑ **24624**

4263 A Concrete Naming Certificate for PointedCompleteMetricSpace↑ **24627**

4264 A Concrete Naming Certificate for IntervalConnectedness↑ **24631**

4265 A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ **24634**

4266 A Concrete Naming Certificate for ComputableCompactMetric$^{\uparrow}$ **24638**

4267 A Concrete Naming Certificate for RegularCauchyHausdorffReflection$^{\uparrow}$ **24641**

4268 A Concrete Naming Certificate for WeakLawLargeNumbers$^{\uparrow}$ **24645**

4269 A Concrete Naming Certificate for LocatedSubset$^{\uparrow}$ **24648**

4270 A Concrete Naming Certificate for RegularCauchyHeineBorelRoute$^{\uparrow}$ **24651**

4271 A Concrete Naming Certificate for CauchyApproximationTree$^{\uparrow}$ **24656**

4272 A Concrete Naming Certificate for BishopCauchyRepresentationComparison$^{\uparrow}$ **24659**

4273 A Concrete Naming Certificate for SeparableSpace$^{\uparrow}$ **24662**

4274 A Concrete Naming Certificate for FormalBallDomain$^{\uparrow}$ **24665**

4275 A Concrete Naming Certificate for DedekindCutStream$^{\uparrow}$ **24669**

4276 A Concrete Naming Certificate for StokesTheorem$^{\uparrow}$ **24673**

4277 A Concrete Naming Certificate for ConstructiveImplicitFunction$^{\uparrow}$ **24676**

4278 A Concrete Naming Certificate for MetricCompletionExistence$^{\uparrow}$ **24679**

4279 A Concrete Naming Certificate for FiniteLimitSeal$^{\uparrow}$ **24687**

4280 A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$ **24693**

4281 A Concrete Naming Certificate for HoeffdingFiniteSum$^{\uparrow}$ **24696**

4282 A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$ **24699**

4283 A Concrete Naming Certificate for CauchyCompletionRoute$^{\uparrow}$ **24702**

4284 A Concrete Naming Certificate for MetaCICProofSearchFrontier$^{\uparrow}$ **24705**

4285 A Concrete Naming Certificate for DyadicIntervalHull$^{\uparrow}$ **24708**

4286 A Concrete Naming Certificate for CellularCarrierSwitchCost$^{\uparrow}$ **24711**

4287 A Concrete Naming Certificate for UniformlyContinuousSequence$^{\uparrow}$ **24714**

4288 A Concrete Naming Certificate for LocatedIntervalHalving$^{\uparrow}$ **24721**

4289 A Concrete Naming Certificate for BishopLocatedRoot$^{\uparrow}$ **24724**

4290 A Concrete Naming Certificate for RealCompletenessWitness$^{\uparrow}$ **24727**

429.1 Concrete Naming Certificate for SmoothManifold↑ **24730**

429.2 Concrete Naming Certificate for MetaCICNormalEndpointBudget↑ **24733**

429.3 Concrete Naming Certificate for PolishCauchyBasis↑ **24736**

429.4 Concrete Naming Certificate for SymmetryRestorationStabilizer↑ **24739**

429.5 Concrete Naming Certificate for KolmogorovContinuity↑ **24742**

429.6 Concrete Naming Certificate for TaggedRealPartition↑ **24745**

429.7 Concrete Naming Certificate for FiniteFanBarModulus↑ **24748**

429.8 Concrete Naming Certificate for EdelsteinFixedPoint↑ **24751**

429.9 Concrete Naming Certificate for CompletionLadder↑ **24754**

430.0 Concrete Naming Certificate for LocatedFormalBar↑ **24757**

430.1 Concrete Naming Certificate for CauchyModulusFunctoriality↑ **24759**

430.2 Concrete Naming Certificate for GaugeSpace↑ **24762**

430.3 Concrete Naming Certificate for RealLimitStability↑ **24766**

430.4 Concrete Naming Certificate for DecimalIntervalReal↑ **24769**

430.5 Concrete Naming Certificate for AttouchWetsTopology↑ **24773**

430.6 Concrete Naming Certificate for CantorIntersectionMetric↑ **24776**

430.7 Concrete Naming Certificate for TightRealInterval↑ **24779**

430.8 Concrete Naming Certificate for CompactUniformModulusTransfer↑ **24782**

430.9 Concrete Naming Certificate for RealIntervalApartness↑ **24785**

431.0 Concrete Naming Certificate for PartitionBasinPolynomial↑ **24788**

431.1 Concrete Naming Certificate for DistanceToSet↑ **24791**

431.2 Concrete Naming Certificate for JordanCurve↑ **24794**

431.3 Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary↑ **24797**

431.4 Concrete Naming Certificate for PointedMetricSpace↑ **24800**

431.5 Concrete Naming Certificate for PointedCompleteMetricSpace↑ **24803**

431.6 Concrete Naming Certificate for IntervalConnectedness↑ **24807**

4317 Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility↑ **24810**

4318 Concrete Naming Certificate for ComputableCompactMetric↑ **24814**

4319 Concrete Naming Certificate for RegularCauchyHausdorffReflection↑ **24817**

4320 Concrete Naming Certificate for WeakLawLargeNumbers↑ **24821**

4321 Concrete Naming Certificate for LocatedSubset↑ **24824**

4322 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **24827**

4323 Concrete Naming Certificate for CauchyApproximationTree↑ **24832**

4324 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **24835**

4325 Concrete Naming Certificate for SeparableSpace↑ **24838**

4326 Concrete Naming Certificate for DedekindCutStream↑ **24841**

4327 Concrete Naming Certificate for StokesTheorem↑ **24845**

4328 Concrete Naming Certificate for MetricCompletionExistence↑ **24848**

4329 Concrete Naming Certificate for FiniteLimitSeal↑ **24856**

4330 Concrete Naming Certificate for FormalBallDomain↑ **24862**

4331 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **24866**

4332 Concrete Naming Certificate for RieszThorinInterpolation↑ **24870**

4333 Concrete Naming Certificate for HoeffdingFiniteSum↑ **24873**

4334 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **24876**

4335 Concrete Naming Certificate for CauchyModulusBridge↑ **24879**

4336 Concrete Naming Certificate for BoundedLinearFunctional↑ **24882**

4337 Concrete Naming Certificate for CompletionModulusCategory↑ **24885**

4338 Concrete Naming Certificate for MarkovTriggerDynamics↑ **24888**

4339 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **24891**

4340 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **24894**

4341 Concrete Naming Certificate for DiniUniformLimitModulus↑ **24897**

4342 Concrete Naming Certificate for SummationByParts↑ **24900**

4343 Concrete Naming Certificate for RealLineOrder↑ **24903**

4344 Concrete Naming Certificate for SchwartzKernel↑ **24906**

4345 Concrete Naming Certificate for SynonymousTransition↑ **24909**

4346 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **24912**

4347 Concrete Naming Certificate for LipschitzRealMap↑ **24915**

4348 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **24918**

4349 Concrete Naming Certificate for Dendrite↑ **24922**

4350 Concrete Naming Certificate for YoungIntegration↑ **24925**

4351 Concrete Naming Certificate for VariationDiminishing↑ **24929**

4352 Concrete Naming Certificate for FrobeniusReciprocity↑ **24932**

4353 Concrete Naming Certificate for NormalOperator↑ **24935**

4354 Concrete Naming Certificate for HausdorffMeasure↑ **24938**

4355 Concrete Naming Certificate for LocallyConnected↑ **24941**

4356 Concrete Naming Certificate for EquicontinuousSequence↑ **24944**

4357 Concrete Naming Certificate for DecimalIntervalReal↑ **24948**

4358 Concrete Naming Certificate for AttouchWetsTopology↑ **24952**

4359 Concrete Naming Certificate for CantorIntersectionMetric↑ **24955**

4360 Concrete Naming Certificate for TightRealInterval↑ **24958**

4361 Concrete Naming Certificate for CompactUniformModulusTransfer↑ **24961**

4362 Concrete Naming Certificate for RealIntervalApartness↑ **24964**

4363 Concrete Naming Certificate for PartitionBasinPolynomial↑ **24967**

4364 Concrete Naming Certificate for DistanceToSet↑ **24970**

4365 Concrete Naming Certificate for JordanCurve↑ **24973**

4366 Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary↑ **24976**

4367 Concrete Naming Certificate for PointedMetricSpace↑ **24979**

4368 Concrete Naming Certificate for PointedCompleteMetricSpace↑ **24982**

4369 A Concrete Naming Certificate for IntervalConnectedness↑ **24986**

4370 A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility↑ **24989**

4371 A Concrete Naming Certificate for ComputableCompactMetric↑ **24993**

4372 A Concrete Naming Certificate for RegularCauchyHausdorffReflection↑ **24996**

4373 A Concrete Naming Certificate for WeakLawLargeNumbers↑ **25000**

4374 A Concrete Naming Certificate for LocatedSubset↑ **25003**

4375 A Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **25006**

4376 A Concrete Naming Certificate for CauchyApproximationTree↑ **25011**

4377 A Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **25014**

4378 A Concrete Naming Certificate for SeparableSpace↑ **25017**

4379 A Concrete Naming Certificate for DedekindCutStream↑ **25020**

4380 A Concrete Naming Certificate for StokesTheorem↑ **25024**

4381 A Concrete Naming Certificate for MetricCompletionExistence↑ **25027**

4382 A Concrete Naming Certificate for FiniteLimitSeal↑ **25035**

4383 A Concrete Naming Certificate for FormalBallDomain↑ **25041**

4384 A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **25045**

4385 A Concrete Naming Certificate for RieszThorinInterpolation↑ **25049**

4386 A Concrete Naming Certificate for HoeffdingFiniteSum↑ **25052**

4387 A Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **25055**

4388 A Concrete Naming Certificate for CauchyModulusBridge↑ **25058**

4389 A Concrete Naming Certificate for MyhillNerode↑ **25061**

4390 A Concrete Naming Certificate for VariationSeminorm↑ **25064**

4391 A Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **25067**

4392 A Concrete Naming Certificate for BoundedLinearFunctional↑ **25070**

4393 A Concrete Naming Certificate for CompletionModulusCategory↑ **25073**

4394 A Concrete Naming Certificate for MarkovTriggerDynamics↑ **25076**

4395 A Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **25079**

4396 A Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **25082**

4397 A Concrete Naming Certificate for DiniUniformLimitModulus↑ **25085**

4398 A Concrete Naming Certificate for SummationByParts↑ **25088**

4399 A Concrete Naming Certificate for RealLineOrder↑ **25091**

4400 A Concrete Naming Certificate for SchwartzKernel↑ **25094**

4401 A Concrete Naming Certificate for SynonymousTransition↑ **25097**

4402 A Concrete Naming Certificate for CentralLimitFiniteWindow↑ **25100**

4403 A Concrete Naming Certificate for LipschitzRealMap↑ **25103**

4404 A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **25106**

4405 A Concrete Naming Certificate for Dendrite↑ **25110**

4406 A Concrete Naming Certificate for YoungIntegration↑ **25113**

4407 A Concrete Naming Certificate for VariationDiminishing↑ **25117**

4408 A Concrete Naming Certificate for FrobeniusReciprocity↑ **25120**

4409 A Concrete Naming Certificate for NormalOperator↑ **25123**

4410 A Concrete Naming Certificate for HausdorffMeasure↑ **25126**

4411 A Concrete Naming Certificate for LocallyConnected↑ **25129**

4412 A Concrete Naming Certificate for BishopCauchyMean↑ **25132**

4413 A Concrete Naming Certificate for BishopFixedPointInterval↑ **25135**

4414 A Concrete Naming Certificate for PrivateCoordinateTrigger↑ **25138**

4415 A Concrete Naming Certificate for LocatedRealComparisonModulus↑ **25141**

4416 A Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **25144**

4417 A Concrete Naming Certificate for HellyBraySelection↑ **25147**

4418 A Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **25151**

4419 A Concrete Naming Certificate for FiniteCauchyFilter↑ **25154**

4420 A Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **25156**

4421 Concrete Naming Certificate for RegulatedConvergence↑ **25159**

4422 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **25162**

4423 Concrete Naming Certificate for CauchyTailRequest↑ **25165**

4424 Concrete Naming Certificate for QuotientLeakageMarkov↑ **25167**

4425 Concrete Naming Certificate for CauchyCriterionUniform↑ **25170**

4426 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **25173**

4427 Concrete Naming Certificate for CauchyModulusSubsequence↑ **25176**

4428 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **25179**

4429 Concrete Naming Certificate for BorsukUlam↑ **25182**

4430 Concrete Naming Certificate for DenjoyYoungSaks↑ **25185**

4431 Concrete Naming Certificate for UniformlyConvexSpace↑ **25189**

4432 Concrete Naming Certificate for HyperbolicTranslationLength↑ **25192**

4433 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **25195**

4434 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **25197**

4435 Concrete Naming Certificate for UniformInverseModulus↑ **25200**

4436 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **25203**

4437 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **25206**

4438 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **25209**

4439 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **25211**

4440 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **25214**

4441 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **25217**

4442 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **25221**

4443 Concrete Naming Certificate for OneSidedLimit↑ **25224**

4444 Concrete Naming Certificate for BertrandSeriesTest↑ **25227**

4445 Concrete Naming Certificate for RationalCanonicalForm↑ **25231**

4446 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **25234**

4447 A Concrete Naming Certificate for ObservabilityGramian↑ **25236**
4447.1 Carrier definition 25236
4447.2 Certificate surface 25237

4448 A Concrete Naming Certificate for HelmholtzDecomposition↑ **25239**

4449 A Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **25242**

4450 A Concrete Naming Certificate for BoundedLinearFunctional↑ **25245**

4451 A Concrete Naming Certificate for CompletionModulusCategory↑ **25248**

4452 A Concrete Naming Certificate for MarkovTriggerDynamics↑ **25251**

4453 A Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **25254**

4454 A Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **25257**

4455 A Concrete Naming Certificate for DiniUniformLimitModulus↑ **25260**

4456 A Concrete Naming Certificate for SummationByParts↑ **25263**

4457 A Concrete Naming Certificate for RealLineOrder↑ **25266**

4458 A Concrete Naming Certificate for SchwartzKernel↑ **25269**

4459 A Concrete Naming Certificate for SynonymousTransition↑ **25272**

4460 A Concrete Naming Certificate for CentralLimitFiniteWindow↑ **25275**

4461 A Concrete Naming Certificate for LipschitzRealMap↑ **25278**

4462 A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **25281**

4463 A Concrete Naming Certificate for Dendrite↑ **25285**

4464 A Concrete Naming Certificate for YoungIntegration↑ **25288**

4465 A Concrete Naming Certificate for VariationDiminishing↑ **25292**

4466 A Concrete Naming Certificate for FrobeniusReciprocity↑ **25295**

4467 A Concrete Naming Certificate for NormalOperator↑ **25298**

4468 A Concrete Naming Certificate for HausdorffMeasure↑ **25301**

4469 A Concrete Naming Certificate for LocallyConnected↑ **25304**

4470 A Concrete Naming Certificate for BishopCauchyMean↑ **25307**

4471 A Concrete Naming Certificate for BishopFixedPointInterval↑ **25310**

4472 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **25313**

4473 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **25316**

4474 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **25319**

4475 Concrete Naming Certificate for HellyBraySelection↑ **25322**

4476 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **25326**

4477 Concrete Naming Certificate for FiniteCauchyFilter↑ **25329**

4478 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **25331**

4479 Concrete Naming Certificate for RegulatedConvergence↑ **25334**

4480 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **25337**

4481 Concrete Naming Certificate for CauchyTailRequest↑ **25340**

4482 Concrete Naming Certificate for QuotientLeakageMarkov↑ **25342**

4483 Concrete Naming Certificate for CauchyCriterionUniform↑ **25345**

4484 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **25348**

4485 Concrete Naming Certificate for CauchyModulusSubsequence↑ **25351**

4486 Concrete Naming Certificate for RegularCauchySequenceComparison↑ **25354**

4487 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **25359**

4488 Concrete Naming Certificate for BorsukUlam↑ **25362**

4489 Concrete Naming Certificate for DenjoyYoungSaks↑ **25365**

4490 Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **25369**

4491 Concrete Naming Certificate for UniformlyConvexSpace↑ **25373**

4492 Concrete Naming Certificate for HyperbolicTranslationLength↑ **25376**

4493 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **25379**

4494 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **25381**

4495 Concrete Naming Certificate for UniformInverseModulus↑ **25384**

4496 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **25387**

4497 Concrete Naming Certificate for FreeGroup↑ **25390**

4498 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **25393**

4499 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **25397**

4500 A Concrete Naming Certificate for LocatedCauchyFilterBasis **25400**

4501 A Concrete Naming Certificate for OneSidedLimit↑ **25404**

4502 A Concrete Naming Certificate for OneSidedCauchyLimit↑ **25407**

4503 A Concrete Naming Certificate for BertrandSeriesTest↑ **25410**

4504 A Concrete Naming Certificate for BertrandTest↑ **25414**

4505 A Concrete Naming Certificate for CauchyContinuousExtensionTheorem↑ **25417**

4506 A Concrete Naming Certificate for HallMarriage↑ **25419**

4507 A Concrete Naming Certificate for RationalCanonicalForm↑ **25421**

4508 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **25424**

4509 A Concrete Naming Certificate for RealizerSpace↑ **25426**

4510 A Concrete Naming Certificate for RuleOneTenCausalCone↑ **25429**

4511 A Concrete Naming Certificate for BaireTreeModulus↑ **25431**

4512 Located Uniform Cauchy Principle NameCert Construction **25434**

4513 A Concrete Naming Certificate for BaireSpaceCompactOpen↑ **25437**

4514 A Concrete Naming Certificate for CompactApartnessProfile↑ **25440**

4515 A Concrete Naming Certificate for RiemannIntegralCauchy↑ **25443**

4516 A Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **25446**

4517 A Concrete Naming Certificate for BoundedRegulatedFunction↑ **25449**

4518 A Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **25452**

4519 A Concrete Naming Certificate for ArensEellsSpace↑ **25456**

4520 A Concrete Naming Certificate for BanachFixedPointStability↑ **25460**

4521 A Concrete Naming Certificate for RealTriangleInequality↑ **25463**

4522 A Concrete Naming Certificate for LowerDarbouxSum↑ **25465**

4523 A Concrete Naming Certificate for BoundedMetricSpace↑ **25468**

4524 Concrete Naming Certificate for LegendreFenchelEnvelope↑ **25471**

4525 Concrete Naming Certificate for PrefixTree↑ **25473**

4526 Concrete Naming Certificate for ChomskyNormalForm↑ **25476**

4527 Concrete Naming Certificate for FiniteFanTheorem↑ **25479**

4528 Concrete Naming Certificate for SpecializationOrder↑ **25483**

4529 Concrete Naming Certificate for FiniteModulusReal↑ **25486**

4530 Concrete Naming Certificate for CompactUniformSubordinateCover↑ **25489**

4531 Concrete Naming Certificate for BishopIntervalContraction↑ **25491**

4532 Concrete Naming Certificate for SeminormedSpace↑ **25496**

4533 Concrete Naming Certificate for ReassignmentMassSubcube↑ **25499**

4534 Concrete Naming Certificate for RealBolzanoWeierstrass↑ **25502**

4535 Concrete Naming Certificate for CandidateSetClosedRoute↑ **25505**

4536 Concrete Naming Certificate for CauchyInterlacing↑ **25508**

4537 Concrete Naming Certificate for BisectionReal↑ **25511**

4538 Concrete Naming Certificate for DarbouxCriterion↑ **25515**

4539 Concrete Naming Certificate for RiemannSumConsistency↑ **25518**

4540 Concrete Naming Certificate for RealLocallyCompact↑ **25521**

4541 Concrete Naming Certificate for CompactUniformModulusStability↑ **25524**

4542 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **25527**

4543 Concrete Naming Certificate for CauchyApproximationTree↑ **25532**

4544 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **25535**

4545 Concrete Naming Certificate for SeparableSpace↑ **25538**

4546 Concrete Naming Certificate for DedekindCutStream↑ **25541**

4547 Concrete Naming Certificate for StokesTheorem↑ **25545**

4548 Concrete Naming Certificate for MetricCompletionExistence↑ **25548**

4549 Concrete Naming Certificate for FiniteLimitSeal↑ **25556**

4550 Concrete Naming Certificate for FormalBallDomain↑ **25562**

4551 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **25566**

4552 Concrete Naming Certificate for RieszThorinInterpolation↑ **25570**

4553 Concrete Naming Certificate for HoeffdingFiniteSum↑ **25573**

4554 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **25576**

4555 Concrete Naming Certificate for CauchyModulusBridge↑ **25579**

4556 Concrete Naming Certificate for MyhillNerode↑ **25582**

4557 Concrete Naming Certificate for VariationSeminorm↑ **25585**

4558 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **25588**

4559 Concrete Naming Certificate for BoundedLinearFunctional↑ **25591**

4560 Concrete Naming Certificate for CompletionModulusCategory↑ **25594**

4561 Concrete Naming Certificate for MarkovTriggerDynamics↑ **25597**

4562 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **25600**

4563 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **25603**

4564 Concrete Naming Certificate for DiniUniformLimitModulus↑ **25606**

4565 Concrete Naming Certificate for SummationByParts↑ **25609**

4566 Concrete Naming Certificate for RealLineOrder↑ **25612**

4567 Concrete Naming Certificate for SchwartzKernel↑ **25615**

4568 Concrete Naming Certificate for SynonymousTransition↑ **25618**

4569 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **25621**

4570 Concrete Naming Certificate for LipschitzRealMap↑ **25624**

4571 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **25627**

4572 Concrete Naming Certificate for Dendrite↑ **25631**

4573 Concrete Naming Certificate for YoungIntegration↑ **25634**

4574 Concrete Naming Certificate for VariationDiminishing↑ **25638**

4575 Concrete Naming Certificate for FrobeniusReciprocity↑ **25641**

4576 A Concrete Naming Certificate for NormalOperator$^{\uparrow}$ **25644**

4577 A Concrete Naming Certificate for HausdorffMeasure$^{\uparrow}$ **25647**

4578 A Concrete Naming Certificate for LocallyConnected$^{\uparrow}$ **25650**

4579 A Concrete Naming Certificate for BishopCauchyMean$^{\uparrow}$ **25653**

4580 A Concrete Naming Certificate for BishopFixedPointInterval$^{\uparrow}$ **25656**

4581 A Concrete Naming Certificate for PrivateCoordinateTrigger$^{\uparrow}$ **25659**

4582 A Concrete Naming Certificate for LocatedRealComparisonModulus$^{\uparrow}$ **25662**

4583 A Concrete Naming Certificate for BoundaryShadowExpansionCocycle$^{\uparrow}$ **25665**

4584 A Concrete Naming Certificate for HellyBraySelection$^{\uparrow}$ **25668**

4585 A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$ **25672**

4586 A Concrete Naming Certificate for FiniteCauchyFilter$^{\uparrow}$ **25675**

4587 A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$ **25677**

4588 A Concrete Naming Certificate for RegulatedConvergence$^{\uparrow}$ **25680**

4589 A Concrete Naming Certificate for LocatedNestedIntervalLimit$^{\uparrow}$ **25683**

4590 A Concrete Naming Certificate for CauchyTailRequest$^{\uparrow}$ **25686**

4591 A Concrete Naming Certificate for QuotientLeakageMarkov$^{\uparrow}$ **25688**

4592 A Concrete Naming Certificate for CauchyCriterionUniform$^{\uparrow}$ **25691**

4593 A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$ **25694**

4594 A Concrete Naming Certificate for CauchyModulusSubsequence$^{\uparrow}$ **25697**

4595 A Concrete Naming Certificate for CauchyCompletionMonadicity$^{\uparrow}$ **25700**

4596 A Concrete Naming Certificate for BorsukUlam$^{\uparrow}$ **25703**

4597 A Concrete Naming Certificate for DenjoyYoungSaks$^{\uparrow}$ **25706**

4598 A Concrete Naming Certificate for UniformlyConvexSpace$^{\uparrow}$ **25710**

4599 A Concrete Naming Certificate for HyperbolicTranslationLength$^{\uparrow}$ **25713**

4600 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope$^{\uparrow}$ **25716**

4601 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection$^{\uparrow}$ **25718**

4602 Concrete Naming Certificate for UniformInverseModulus↑ **25721**

4603 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **25724**

4604 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **25727**

4605 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **25730**

4606 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **25732**

4607 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **25735**

4608 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **25738**

4609 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **25742**

4610 Concrete Naming Certificate for OneSidedLimit↑ **25745**

4611 Concrete Naming Certificate for BertrandSeriesTest↑ **25748**

4612 Concrete Naming Certificate for RationalCanonicalForm↑ **25752**

4613 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **25755**

4614 Concrete Naming Certificate for ObservabilityGramian↑ **25757**

4614.1 Carrier definition 25757

4614.2 Certificate surface 25758

4615 Concrete Naming Certificate for HelmholtzDecomposition↑ **25760**

4616 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **25763**

4617 Concrete Naming Certificate for BoundedLinearFunctional↑ **25766**

4618 Concrete Naming Certificate for CompletionModulusCategory↑ **25769**

4619 Concrete Naming Certificate for MarkovTriggerDynamics↑ **25772**

4620 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **25775**

4621 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **25778**

4622 Concrete Naming Certificate for DiniUniformLimitModulus↑ **25781**

4623 Concrete Naming Certificate for SummationByParts↑ **25784**

4624 Concrete Naming Certificate for RealLineOrder↑ **25787**

4625 Concrete Naming Certificate for SchwartzKernel↑ **25790**

4626 Concrete Naming Certificate for SynonymousTransition↑ **25793**

4627 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **25796**

4628 Concrete Naming Certificate for LipschitzRealMap↑ **25799**

4629 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **25802**

4630 Concrete Naming Certificate for Dendrite↑ **25806**

4631 Concrete Naming Certificate for YoungIntegration↑ **25809**

4632 Concrete Naming Certificate for VariationDiminishing↑ **25813**

4633 Concrete Naming Certificate for FrobeniusReciprocity↑ **25816**

4634 Concrete Naming Certificate for NormalOperator↑ **25819**

4635 Concrete Naming Certificate for HausdorffMeasure↑ **25822**

4636 Concrete Naming Certificate for LocallyConnected↑ **25825**

4637 Concrete Naming Certificate for BishopCauchyMean↑ **25828**

4638 Concrete Naming Certificate for BishopFixedPointInterval↑ **25831**

4639 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **25834**

4640 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **25837**

4641 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **25840**

4642 Concrete Naming Certificate for HellyBraySelection↑ **25843**

4643 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **25847**

4644 Concrete Naming Certificate for FiniteCauchyFilter↑ **25850**

4645 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **25852**

4646 Concrete Naming Certificate for RegulatedConvergence↑ **25855**

4647 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **25858**

4648 Concrete Naming Certificate for CauchyTailRequest↑ **25861**

4649 Concrete Naming Certificate for QuotientLeakageMarkov↑ **25863**

4650 Concrete Naming Certificate for CauchyCriterionUniform↑ **25866**

4651 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **25869**

4652 Concrete Naming Certificate for CauchyModulusSubsequence↑ **25872**

4653 Concrete Naming Certificate for RegularCauchySequenceComparison↑ **25875**

4654 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **25880**

4655 Concrete Naming Certificate for BorsukUlam↑ **25883**

4656 Concrete Naming Certificate for DenjoyYoungSaks↑ **25886**

4657 Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **25890**

4658 Concrete Naming Certificate for UniformlyConvexSpace↑ **25894**

4659 Concrete Naming Certificate for HyperbolicTranslationLength↑ **25897**

4660 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **25900**

4661 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **25902**

4662 Concrete Naming Certificate for UniformInverseModulus↑ **25905**

4663 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **25908**

4664 Concrete Naming Certificate for FreeGroup↑ **25911**

4665 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **25914**

4666 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **25918**

4667 Concrete Naming Certificate for LocatedCauchyFilterBasis **25921**

4668 Concrete Naming Certificate for OneSidedLimit↑ **25925**

4669 Concrete Naming Certificate for BertrandSeriesTest↑ **25928**

4670 Concrete Naming Certificate for RationalCanonicalForm↑ **25932**

4671 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **25935**

4672 Concrete Naming Certificate for RealizerSpace↑ **25937**

4673 Concrete Naming Certificate for RuleOneTenCausalCone↑ **25940**

4674 Concrete Naming Certificate for BaireTreeModulus↑ **25942**

4675 Located Uniform Cauchy Principle NameCert Construction **25945**

4676 Concrete Naming Certificate for BaireSpaceCompactOpen↑ **25948**

4677 Concrete Naming Certificate for CompactApartnessProfile↑ **25951**

4678 Concrete Naming Certificate for RiemannIntegralCauchy↑ **25954**

4679 A Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **25957**

4680 A Concrete Naming Certificate for BoundedRegulatedFunction↑ **25960**

4681 A Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **25963**

4682 A Concrete Naming Certificate for ArensEellsSpace↑ **25967**

4683 A Concrete Naming Certificate for BanachFixedPointStability↑ **25971**

4684 A Concrete Naming Certificate for RealTriangleInequality↑ **25974**

4685 A Concrete Naming Certificate for LowerDarbouxSum↑ **25976**

4686 A Concrete Naming Certificate for BoundedMetricSpace↑ **25979**

4687 A Concrete Naming Certificate for LegendreFenchelEnvelope↑ **25982**

4688 A Concrete Naming Certificate for PrefixTree↑ **25984**

4689 A Concrete Naming Certificate for ChomskyNormalForm↑ **25987**

4690 A Concrete Naming Certificate for FiniteFanTheorem↑ **25990**

4691 A Concrete Naming Certificate for SpecializationOrder↑ **25994**

4692 A Concrete Naming Certificate for FiniteModulusReal↑ **25997**

4693 A Concrete Naming Certificate for CompactUniformSubordinateCover↑ **26000**

4694 A Concrete Naming Certificate for BishopIntervalContraction↑ **26002**

4695 A Concrete Naming Certificate for SeminormedSpace↑ **26007**

4696 A Concrete Naming Certificate for ReassignmentMassSubcube↑ **26010**

4697 A Concrete Naming Certificate for CauchyInterlacing↑ **26013**

4698 A Concrete Naming Certificate for RealLocallyCompact↑ **26016**

4699 A Concrete Naming Certificate for CompactUniformModulusStability↑ **26019**

4700 A Concrete Naming Certificate for CauchyFilterSelector↑ **26022**

4701 A Concrete Naming Certificate for RegularCauchySandwich↑ **26025**

4702 A Concrete Naming Certificate for DynamicProgramming↑ **26029**

4703 Regular Cauchy Even-Odd Substreams NameCert Construction **26032**

4704 A Concrete Naming Certificate for SchwartzSpace↑ **26036**

470.5 A Concrete Naming Certificate for RuleOneTenPrivateSharedDecoration↑ **26039**

470.6 A Concrete Naming Certificate for ConstructiveCantorTheorem↑ **26042**

470.7 Markov-Kakutani Fixed Point NameCert Construction **26045**

470.8 Window-Six Private/Shared Decoration NameCert Construction **26048**

470.9 A Concrete Naming Certificate for SchurLemma↑ **26052**

471.0 A Concrete Naming Certificate for CompactRealIntervalMinimum↑ **26054**

471.1 A Concrete Naming Certificate for SimpleFunctionApproximation↑ **26057**

471.2 A Concrete Naming Certificate for DyadicModulus↑ **26060**

471.3 A Concrete Naming Certificate for CauchyFilterSelector↑ **26063**

471.4 A Concrete Naming Certificate for CauchyModulusSpace↑ **26066**

471.5 A Concrete Naming Certificate for CauchyCompletionLeftExactness↑ **26070**

471.6 A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection↑ **26079**

471.7 A Concrete Naming Certificate for SnakeLemma↑ **26082**

471.8 A Concrete Naming Certificate for RealTailDiameterGauge↑ **26085**

471.9 A Concrete Naming Certificate for EffectiveRealCompactIntervalSearch↑ **26088**

472.0 A Concrete Naming Certificate for MarkedReal↑ **26094**

472.1 A Concrete Naming Certificate for CauchyCompletionDenseEmbedding↑ **26097**

472.2 A Concrete Naming Certificate for RiemannRefinementCriterion↑ **26100**

472.3 A Concrete Naming Certificate for CompactUniformModulusComposition↑ **26103**

472.4 A Concrete Naming Certificate for PrivateSharedDecorationModel↑ **26106**

472.5 A Concrete Naming Certificate for BishopLocatedRealFilter↑ **26108**

472.6 A Concrete Naming Certificate for RegularCauchyDiagonalArgument↑ **26111**

472.7 A Concrete Naming Certificate for PrivateSharedCoordinateDecomposition↑ **26114**

472.8 A Concrete Naming Certificate for RegulatedCauchyModulus↑ **26117**

472.9 A Concrete Naming Certificate for RegularCauchyMinModulus↑ **26120**

473.0 A Concrete Naming Certificate for TaylorTheorem↑ **26127**

4731 A Concrete Naming Certificate for LagrangeInversion$^{\uparrow}$ **26130**

4732 A Concrete Naming Certificate for TaylorRemainder$^{\uparrow}$ **26134**

4733 A Concrete Naming Certificate for ApproximationTower$^{\uparrow}$ **26137**

4734 A Concrete Naming Certificate for ArchimedeanOrderedCauchyCompletion$^{\uparrow}$ **26141**

4735 A Concrete Naming Certificate for ClaimStatusAudit$^{\uparrow}$ **26144**

4736 Located Real Interval Bisection NameCert Construction **26147**

4737 MetaCIC Candidate Normalization Confluence Handoff NameCert Construction **26150**

4738 A Concrete Naming Certificate for CantorIntersectionModulus$^{\uparrow}$ **26198**

4739 A Concrete Naming Certificate for CompactUniformContinuityLedger$^{\uparrow}$ **26201**

4740 A Concrete Naming Certificate for FrechetMetricEmbedding$^{\uparrow}$ **26204**

4741 A Concrete Naming Certificate for CaristiFixedPoint$^{\uparrow}$ **26206**

4742 A Concrete Naming Certificate for WeightedDeformationSpine$^{\uparrow}$ **26209**

4743 A Concrete Naming Certificate for CompactRealInterval$^{\uparrow}$ **26212**

4744 A Concrete Naming Certificate for LocatedCutCompletion$^{\uparrow}$ **26217**

4745 A Concrete Naming Certificate for DedekindEtaCompletion$^{\uparrow}$ **26221**

4746 A Concrete Naming Certificate for GITQuotient$^{\uparrow}$ **26224**

4747 A Concrete Naming Certificate for AubinLions$^{\uparrow}$ **26227**

4748 A Concrete Naming Certificate for CompactUniformPullback$^{\uparrow}$ **26230**

4749 A Concrete Naming Certificate for RegularCauchyGluing$^{\uparrow}$ **26233**

4750 A Concrete Naming Certificate for YoungSymmetrizer$^{\uparrow}$ **26244**

4751 A Concrete Naming Certificate for MetaCICSRKernel$^{\uparrow}$ **26247**

4752 A Concrete Naming Certificate for HInfinityControl$^{\uparrow}$ **26251**

4753 A Concrete Naming Certificate for RealApartnessField$^{\uparrow}$ **26254**

4754 A Concrete Naming Certificate for ModulusOfRegularConvergence$^{\uparrow}$ **26264**

4755 A Concrete Naming Certificate for RHRouteHaltingBoundary$^{\uparrow}$ **26267**

4756 A Concrete Naming Certificate for Premetric$^{\uparrow}$ **26270**

4757 Concrete Naming Certificate for FiniteDimensionalNormEquivalence↑ **26273**

4758 Concrete Naming Certificate for FiniteOrthonormalBasis↑ **26276**

4759 Concrete Naming Certificate for RealCauchyConvergence↑ **26279**

4760 Concrete Naming Certificate for CompactUniformFiniteNet↑ **26282**

4761 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **26284**

4762 Concrete Naming Certificate for CauchyModulusBridge↑ **26287**

4763 Concrete Naming Certificate for MyhillNerode↑ **26290**

4764 Concrete Naming Certificate for VariationSeminorm↑ **26293**

4765 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **26296**

4766 Concrete Naming Certificate for BoundedLinearFunctional↑ **26299**

4767 Concrete Naming Certificate for CompletionModulusCategory↑ **26302**

4768 Concrete Naming Certificate for MarkovTriggerDynamics↑ **26305**

4769 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **26308**

4770 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **26311**

4771 Concrete Naming Certificate for DiniUniformLimitModulus↑ **26314**

4772 Concrete Naming Certificate for SummationByParts↑ **26317**

4773 Concrete Naming Certificate for RealLineOrder↑ **26320**

4774 Concrete Naming Certificate for SchwartzKernel↑ **26323**

4775 Concrete Naming Certificate for SynonymousTransition↑ **26326**

4776 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **26329**

4777 Concrete Naming Certificate for LipschitzRealMap↑ **26332**

4778 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **26335**

4779 Concrete Naming Certificate for Dendrite↑ **26339**

4780 Concrete Naming Certificate for YoungIntegration↑ **26342**

4781 Concrete Naming Certificate for VariationDiminishing↑ **26346**

4782 Concrete Naming Certificate for FrobeniusReciprocity↑ **26349**

4783 Concrete Naming Certificate for NormalOperator$^{\uparrow}$ **26352**

4784 Concrete Naming Certificate for HausdorffMeasure$^{\uparrow}$ **26355**

4785 Concrete Naming Certificate for LocallyConnected$^{\uparrow}$ **26358**

4786 Concrete Naming Certificate for BishopCauchyMean$^{\uparrow}$ **26361**

4787 Concrete Naming Certificate for BishopFixedPointInterval$^{\uparrow}$ **26364**

4788 Concrete Naming Certificate for PrivateCoordinateTrigger$^{\uparrow}$ **26367**

4789 Concrete Naming Certificate for LocatedRealComparisonModulus$^{\uparrow}$ **26370**

4790 Concrete Naming Certificate for BoundaryShadowExpansionCocycle$^{\uparrow}$ **26373**

4791 Concrete Naming Certificate for HellyBraySelection$^{\uparrow}$ **26376**

4792 Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$ **26380**

4793 Concrete Naming Certificate for FiniteCauchyFilter$^{\uparrow}$ **26383**

4794 Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$ **26385**

4795 Concrete Naming Certificate for RegulatedConvergence$^{\uparrow}$ **26388**

4796 Concrete Naming Certificate for LocatedNestedIntervalLimit$^{\uparrow}$ **26391**

4797 Concrete Naming Certificate for CauchyTailRequest$^{\uparrow}$ **26394**

4798 Concrete Naming Certificate for QuotientLeakageMarkov$^{\uparrow}$ **26396**

4799 Concrete Naming Certificate for CauchyCriterionUniform$^{\uparrow}$ **26399**

4800 Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$ **26402**

4801 Concrete Naming Certificate for CauchyModulusSubsequence$^{\uparrow}$ **26405**

4802 Concrete Naming Certificate for CauchyCompletionMonadicity$^{\uparrow}$ **26408**

4803 Concrete Naming Certificate for BorsukUlam$^{\uparrow}$ **26411**

4804 Concrete Naming Certificate for DenjoyYoungSaks$^{\uparrow}$ **26414**

4805 Concrete Naming Certificate for UniformlyConvexSpace$^{\uparrow}$ **26418**

4806 Concrete Naming Certificate for HyperbolicTranslationLength$^{\uparrow}$ **26421**

4807 Concrete Naming Certificate for RegularCauchyCompletionEnvelope$^{\uparrow}$ **26424**

4808 Concrete Naming Certificate for WeaklyLocatedIntervalSelection$^{\uparrow}$ **26426**

4809 A Concrete Naming Certificate for UniformInverseModulus↑ **26429**

4810 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **26432**

4811 A Concrete Naming Certificate for RealCompletionWindowFunctor↑ **26435**

4812 A Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **26438**

4813 A Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **26440**

4814 A Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **26443**

4815 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **26446**

4816 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **26450**

4817 A Concrete Naming Certificate for OneSidedLimit↑ **26453**

4818 A Concrete Naming Certificate for BertrandSeriesTest↑ **26456**

4819 A Concrete Naming Certificate for RationalCanonicalForm↑ **26460**

4820 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **26463**

4821 A Concrete Naming Certificate for ObservabilityGramian↑ **26465**

4821.1 Carrier definition 26465

4821.2 Certificate surface 26466

4822 A Concrete Naming Certificate for HelmholtzDecomposition↑ **26468**

4823 A Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **26471**

4824 A Concrete Naming Certificate for BoundedLinearFunctional↑ **26474**

4825 A Concrete Naming Certificate for CompletionModulusCategory↑ **26477**

4826 A Concrete Naming Certificate for MarkovTriggerDynamics↑ **26480**

4827 A Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **26483**

4828 A Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **26486**

4829 A Concrete Naming Certificate for DiniUniformLimitModulus↑ **26489**

4830 A Concrete Naming Certificate for SummationByParts↑ **26492**

4831 A Concrete Naming Certificate for RealLineOrder↑ **26495**

4832 A Concrete Naming Certificate for SchwartzKernel↑ **26498**

4833 A Concrete Naming Certificate for SynonymousTransition↑ **26501**

4834 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **26504**

4835 Concrete Naming Certificate for LipschitzRealMap↑ **26507**

4836 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **26510**

4837 Concrete Naming Certificate for Dendrite↑ **26514**

4838 Concrete Naming Certificate for YoungIntegration↑ **26517**

4839 Concrete Naming Certificate for VariationDiminishing↑ **26521**

4840 Concrete Naming Certificate for FrobeniusReciprocity↑ **26524**

4841 Concrete Naming Certificate for NormalOperator↑ **26527**

4842 Concrete Naming Certificate for HausdorffMeasure↑ **26530**

4843 Concrete Naming Certificate for LocallyConnected↑ **26533**

4844 Concrete Naming Certificate for BishopCauchyMean↑ **26536**

4845 Concrete Naming Certificate for BishopFixedPointInterval↑ **26539**

4846 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **26542**

4847 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **26545**

4848 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **26548**

4849 Concrete Naming Certificate for HellyBraySelection↑ **26551**

4850 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **26555**

4851 Concrete Naming Certificate for FiniteCauchyFilter↑ **26558**

4852 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **26560**

4853 Concrete Naming Certificate for RegulatedConvergence↑ **26563**

4854 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **26566**

4855 Concrete Naming Certificate for CauchyTailRequest↑ **26569**

4856 Concrete Naming Certificate for QuotientLeakageMarkov↑ **26571**

4857 Concrete Naming Certificate for CauchyCriterionUniform↑ **26574**

4858 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **26577**

4859 Concrete Naming Certificate for CauchyModulusSubsequence↑ **26580**

4860 A Concrete Naming Certificate for RegularCauchySequenceComparison↑ **26583**

4861 A Concrete Naming Certificate for CauchyCompletionMonadicity↑ **26588**

4862 A Concrete Naming Certificate for BorsukUlam↑ **26591**

4863 A Concrete Naming Certificate for DenjoyYoungSaks↑ **26594**

4864 A Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **26598**

4865 A Concrete Naming Certificate for UniformlyConvexSpace↑ **26602**

4866 A Concrete Naming Certificate for HyperbolicTranslationLength↑ **26605**

4867 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **26608**

4868 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **26610**

4869 A Concrete Naming Certificate for UniformInverseModulus↑ **26613**

4870 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **26616**

4871 A Concrete Naming Certificate for FreeGroup↑ **26619**

4872 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **26622**

4873 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **26626**

4874 A Concrete Naming Certificate for LocatedCauchyFilterBasis **26629**

4875 A Concrete Naming Certificate for OneSidedLimit↑ **26633**

4876 A Concrete Naming Certificate for BertrandSeriesTest↑ **26636**

4877 A Concrete Naming Certificate for RationalCanonicalForm↑ **26640**

4878 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **26643**

4879 A Concrete Naming Certificate for RealizerSpace↑ **26645**

4880 A Concrete Naming Certificate for RuleOneTenCausalCone↑ **26648**

4881 A Concrete Naming Certificate for BaireTreeModulus↑ **26650**

4882 Located Uniform Cauchy Principle NameCert Construction **26653**

4883 A Concrete Naming Certificate for BaireSpaceCompactOpen↑ **26656**

4884 A Concrete Naming Certificate for CompactApartnessProfile↑ **26659**

4885 A Concrete Naming Certificate for RiemannIntegralCauchy↑ **26662**

4886 Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **26665**

4887 Concrete Naming Certificate for BoundedRegulatedFunction↑ **26668**

4888 Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **26671**

4889 Concrete Naming Certificate for ArensEellsSpace↑ **26675**

4890 Concrete Naming Certificate for BanachFixedPointStability↑ **26679**

4891 Concrete Naming Certificate for RealTriangleInequality↑ **26682**

4892 Concrete Naming Certificate for LowerDarbouxSum↑ **26684**

4893 Concrete Naming Certificate for BoundedMetricSpace↑ **26687**

4894 Concrete Naming Certificate for LegendreFenchelEnvelope↑ **26690**

4895 Concrete Naming Certificate for PrefixTree↑ **26692**

4896 Concrete Naming Certificate for ChomskyNormalForm↑ **26695**

4897 Concrete Naming Certificate for FiniteFanTheorem↑ **26698**

4898 Concrete Naming Certificate for SpecializationOrder↑ **26702**

4899 Concrete Naming Certificate for FiniteModulusReal↑ **26705**

4900 Concrete Naming Certificate for CompactUniformSubordinateCover↑ **26708**

4901 Concrete Naming Certificate for BishopIntervalContraction↑ **26710**

4902 Concrete Naming Certificate for SeminormedSpace↑ **26715**

4903 Concrete Naming Certificate for ReassignmentMassSubcube↑ **26718**

4904 Concrete Naming Certificate for CauchyInterlacing↑ **26721**

4905 Concrete Naming Certificate for BisectionReal↑ **26724**

4906 Concrete Naming Certificate for DarbouxCriterion↑ **26728**

4907 Concrete Naming Certificate for RealLocallyCompact↑ **26731**

4908 Concrete Naming Certificate for CompactUniformModulusStability↑ **26734**

4909 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **26737**

4910 Concrete Naming Certificate for CauchyApproximationTree↑ **26742**

4911 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **26745**

4912 Concrete Naming Certificate for SeparableSpace↑ **26748**

4913 Concrete Naming Certificate for DedekindCutStream↑ **26751**

4914 Concrete Naming Certificate for StokesTheorem↑ **26755**

4915 Concrete Naming Certificate for MetricCompletionExistence↑ **26758**

4916 Concrete Naming Certificate for FiniteLimitSeal↑ **26766**

4917 Concrete Naming Certificate for FormalBallDomain↑ **26772**

4918 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **26776**

4919 Concrete Naming Certificate for RieszThorinInterpolation↑ **26780**

4920 Concrete Naming Certificate for HoeffdingFiniteSum↑ **26783**

4921 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **26786**

4922 Concrete Naming Certificate for CauchyModulusBridge↑ **26789**

4923 Concrete Naming Certificate for MyhillNerode↑ **26792**

4924 Concrete Naming Certificate for VariationSeminorm↑ **26795**

4925 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **26798**

4926 Concrete Naming Certificate for BoundedLinearFunctional↑ **26801**

4927 Concrete Naming Certificate for CompletionModulusCategory↑ **26804**

4928 Concrete Naming Certificate for MarkovTriggerDynamics↑ **26807**

4929 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **26810**

4930 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **26813**

4931 Concrete Naming Certificate for DiniUniformLimitModulus↑ **26816**

4932 Concrete Naming Certificate for SummationByParts↑ **26819**

4933 Concrete Naming Certificate for RealLineOrder↑ **26822**

4934 Concrete Naming Certificate for SchwartzKernel↑ **26825**

4935 Concrete Naming Certificate for SynonymousTransition↑ **26828**

4936 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **26831**

4937 Concrete Naming Certificate for LipschitzRealMap↑ **26834**

4938 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **26837**

4939 Concrete Naming Certificate for Dendrite↑ **26841**

4940 Concrete Naming Certificate for YoungIntegration↑ **26844**

4941 Concrete Naming Certificate for VariationDiminishing↑ **26848**

4942 Concrete Naming Certificate for FrobeniusReciprocity↑ **26851**

4943 Concrete Naming Certificate for NormalOperator↑ **26854**

4944 Concrete Naming Certificate for HausdorffMeasure↑ **26857**

4945 Concrete Naming Certificate for LocallyConnected↑ **26860**

4946 Concrete Naming Certificate for BishopCauchyMean↑ **26863**

4947 Concrete Naming Certificate for BishopFixedPointInterval↑ **26866**

4948 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **26869**

4949 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **26872**

4950 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **26875**

4951 Concrete Naming Certificate for HellyBraySelection↑ **26878**

4952 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **26882**

4953 Concrete Naming Certificate for FiniteCauchyFilter↑ **26885**

4954 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **26887**

4955 Concrete Naming Certificate for RegulatedConvergence↑ **26890**

4956 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **26893**

4957 Concrete Naming Certificate for CauchyTailRequest↑ **26896**

4958 Concrete Naming Certificate for QuotientLeakageMarkov↑ **26898**

4959 Concrete Naming Certificate for CauchyCriterionUniform↑ **26901**

4960 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **26904**

4961 Concrete Naming Certificate for CauchyModulusSubsequence↑ **26907**

4962 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **26910**

4963 Concrete Naming Certificate for BorsukUlam↑ **26913**

4964 Concrete Naming Certificate for DenjoyYoungSaks↑ **26916**

4965 Concrete Naming Certificate for UniformlyConvexSpace↑ **26920**

4966 Concrete Naming Certificate for HyperbolicTranslationLength↑ **26923**

4967 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **26926**

4968 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **26928**

4969 Concrete Naming Certificate for UniformInverseModulus↑ **26931**

4970 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **26934**

4971 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **26937**

4972 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **26940**

4973 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **26942**

4974 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **26945**

4975 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **26948**

4976 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **26952**

4977 Concrete Naming Certificate for OneSidedLimit↑ **26955**

4978 Concrete Naming Certificate for BertrandSeriesTest↑ **26958**

4979 Concrete Naming Certificate for RationalCanonicalForm↑ **26962**

4980 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **26965**

4981 Concrete Naming Certificate for ObservabilityGramian↑ **26967**

4981.1 Carrier definition 26967

4981.2 Certificate surface 26968

4982 Concrete Naming Certificate for HelmholtzDecomposition↑ **26970**

4983 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **26973**

4984 Concrete Naming Certificate for BoundedLinearFunctional↑ **26976**

4985 Concrete Naming Certificate for CompletionModulusCategory↑ **26979**

4986 Concrete Naming Certificate for MarkovTriggerDynamics↑ **26982**

4987 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **26985**

4988 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **26988**

4989 A Concrete Naming Certificate for DiniUniformLimitModulus↑ **26991**

4990 A Concrete Naming Certificate for SummationByParts↑ **26994**

4991 A Concrete Naming Certificate for RealLineOrder↑ **26997**

4992 A Concrete Naming Certificate for SchwartzKernel↑ **27000**

4993 A Concrete Naming Certificate for SynonymousTransition↑ **27003**

4994 A Concrete Naming Certificate for CentralLimitFiniteWindow↑ **27006**

4995 A Concrete Naming Certificate for LipschitzRealMap↑ **27009**

4996 A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **27012**

4997 A Concrete Naming Certificate for Dendrite↑ **27016**

4998 A Concrete Naming Certificate for YoungIntegration↑ **27019**

4999 A Concrete Naming Certificate for VariationDiminishing↑ **27023**

5000 A Concrete Naming Certificate for FrobeniusReciprocity↑ **27026**

5001 A Concrete Naming Certificate for NormalOperator↑ **27029**

5002 A Concrete Naming Certificate for HausdorffMeasure↑ **27032**

5003 A Concrete Naming Certificate for LocallyConnected↑ **27035**

5004 A Concrete Naming Certificate for BishopCauchyMean↑ **27038**

5005 A Concrete Naming Certificate for BishopFixedPointInterval↑ **27041**

5006 A Concrete Naming Certificate for PrivateCoordinateTrigger↑ **27044**

5007 A Concrete Naming Certificate for LocatedRealComparisonModulus↑ **27047**

5008 A Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **27050**

5009 A Concrete Naming Certificate for HellyBraySelection↑ **27053**

5010 A Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **27057**

5011 A Concrete Naming Certificate for FiniteCauchyFilter↑ **27060**

5012 A Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **27062**

5013 A Concrete Naming Certificate for RegulatedConvergence↑ **27065**

5014 A Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **27068**

5015 A Concrete Naming Certificate for CauchyTailRequest↑ **27071**

5016 A Concrete Naming Certificate for QuotientLeakageMarkov↑ **27073**

5017 A Concrete Naming Certificate for CauchyCriterionUniform↑ **27076**

5018 A Concrete Naming Certificate for CauchyProductCompletionProjection↑ **27079**

5019 A Concrete Naming Certificate for CauchyModulusSubsequence↑ **27082**

5020 A Concrete Naming Certificate for RegularCauchySequenceComparison↑ **27085**

5021 A Concrete Naming Certificate for CauchyCompletionMonadicity↑ **27090**

5022 A Concrete Naming Certificate for BorsukUlam↑ **27093**

5023 A Concrete Naming Certificate for DenjoyYoungSaks↑ **27096**

5024 A Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **27100**

5025 A Concrete Naming Certificate for UniformlyConvexSpace↑ **27104**

5026 A Concrete Naming Certificate for HyperbolicTranslationLength↑ **27107**

5027 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **27110**

5028 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **27112**

5029 A Concrete Naming Certificate for UniformInverseModulus↑ **27115**

5030 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **27118**

5031 A Concrete Naming Certificate for FreeGroup↑ **27121**

5032 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **27124**

5033 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **27128**

5034 A Concrete Naming Certificate for LocatedCauchyFilterBasis **27131**

5035 A Concrete Naming Certificate for OneSidedLimit↑ **27135**

5036 A Concrete Naming Certificate for BertrandSeriesTest↑ **27138**

5037 A Concrete Naming Certificate for RationalCanonicalForm↑ **27142**

5038 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **27145**

5039 A Concrete Naming Certificate for RealizerSpace↑ **27147**

5040 A Concrete Naming Certificate for RuleOneTenCausalCone↑ **27150**

504.1 Concrete Naming Certificate for BaireTreeModulus↑ **27152**

504.2 Located Uniform Cauchy Principle NameCert Construction **27155**

504.3 Concrete Naming Certificate for BaireSpaceCompactOpen↑ **27158**

504.4 Concrete Naming Certificate for CompactApartnessProfile↑ **27161**

504.5 Concrete Naming Certificate for RiemannIntegralCauchy↑ **27164**

504.6 Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **27167**

504.7 Concrete Naming Certificate for BoundedRegulatedFunction↑ **27170**

504.8 Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **27173**

504.9 Concrete Naming Certificate for ArensEellsSpace↑ **27177**

505.0 Concrete Naming Certificate for BanachFixedPointStability↑ **27181**

505.1 Concrete Naming Certificate for RealTriangleInequality↑ **27184**

505.2 Concrete Naming Certificate for LowerDarbouxSum↑ **27186**

505.3 Concrete Naming Certificate for BoundedMetricSpace↑ **27189**

505.4 Concrete Naming Certificate for LegendreFenchelEnvelope↑ **27192**

505.5 Concrete Naming Certificate for PrefixTree↑ **27194**

505.6 Concrete Naming Certificate for ChomskyNormalForm↑ **27197**

505.7 Concrete Naming Certificate for FiniteFanTheorem↑ **27200**

505.8 Concrete Naming Certificate for SpecializationOrder↑ **27204**

505.9 Concrete Naming Certificate for FiniteModulusReal↑ **27207**

506.0 Concrete Naming Certificate for CompactUniformSubordinateCover↑ **27210**

506.1 Concrete Naming Certificate for BishopIntervalContraction↑ **27212**

506.2 Concrete Naming Certificate for SeminormedSpace↑ **27217**

506.3 Concrete Naming Certificate for ReassignmentMassSubcube↑ **27220**

506.4 Concrete Naming Certificate for CauchyInterlacing↑ **27223**

506.5 Concrete Naming Certificate for RealLocallyCompact↑ **27226**

506.6 Concrete Naming Certificate for CompactUniformModulusStability↑ **27229**

506 A Concrete Naming Certificate for CauchyFilterSelector↑ **27232**

506 A Concrete Naming Certificate for RegularCauchySandwich↑ **27235**

506 A Concrete Naming Certificate for DynamicProgramming↑ **27239**

507 Regular Cauchy Even-Odd Substreams NameCert Construction **27242**

507 A Concrete Naming Certificate for SchwartzSpace↑ **27246**

507 A Concrete Naming Certificate for RuleOneTenPrivateSharedDecoration↑ **27249**

507 A Concrete Naming Certificate for ConstructiveCantorTheorem↑ **27252**

507 Markov-Kakutani Fixed Point NameCert Construction **27255**

507 Window-Six Private/Shared Decoration NameCert Construction **27258**

507 A Concrete Naming Certificate for SchurLemma↑ **27262**

507 A Concrete Naming Certificate for CompactRealIntervalMinimum↑ **27264**

507 A Concrete Naming Certificate for SimpleFunctionApproximation↑ **27267**

507 A Concrete Naming Certificate for DyadicModulus↑ **27270**

508 A Concrete Naming Certificate for CauchyFilterSelector↑ **27273**

508 A Concrete Naming Certificate for CauchyModulusSpace↑ **27276**

508 A Concrete Naming Certificate for CauchyCompletionLeftExactness↑ **27280**

508 A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection↑ **27289**

508 A Concrete Naming Certificate for SnakeLemma↑ **27292**

508 A Concrete Naming Certificate for RealTailDiameterGauge↑ **27295**

508 A Concrete Naming Certificate for EffectiveRealCompactIntervalSearch↑ **27298**

508 A Concrete Naming Certificate for CauchyCompletionDenseEmbedding↑ **27304**

508 A Concrete Naming Certificate for RiemannRefinementCriterion↑ **27307**

508 A Concrete Naming Certificate for CompactUniformModulusComposition↑ **27310**

509 A Concrete Naming Certificate for PrivateSharedDecorationModel↑ **27313**

509 A Concrete Naming Certificate for BishopLocatedRealFilter↑ **27315**

509 A Concrete Naming Certificate for TaylorTheorem↑ **27318**

5093 A Concrete Naming Certificate for RegularCauchyGluing† **27321**

5094 A Concrete Naming Certificate for RegularCauchyTailMajorant† **27332**

5095 A Concrete Naming Certificate for RealCantorIntersectionFiniteWindow† **27334**

5096 A Concrete Naming Certificate for CauchySubsequenceSelector† **27336**

5097 A Concrete Naming Certificate for ArtinianRing† **27342**

5097.0 Carrier Definition 27342

5098 A Concrete Naming Certificate for MooreSmithSubnet† **27344**

5099 A Concrete Naming Certificate for FiniteObservationConfluenceBudget† **27347**

5100 A Concrete Naming Certificate for FrobeniusNorm† **27350**

5101 A Concrete Naming Certificate for BoundedBilinearMap† **27353**

5102 A Concrete Naming Certificate for KalmanFilter† **27356**

5102.0 Finite carrier 27356

5103 A Concrete Naming Certificate for ControlRiccati† **27362**

5104 A Concrete Naming Certificate for RoucheTheorem† **27364**

5105 A Concrete Naming Certificate for FiniteIntervalCover† **27368**

5106 A Concrete Naming Certificate for ConstructiveHeineBorel† **27371**

5107 A Concrete Naming Certificate for HyperbolicFixedPointObstruction† **27375**

5108 A Concrete Naming Certificate for HardySpace† **27378**

5109 A Concrete Naming Certificate for FiniteObservationReflection† **27381**

5110 A Concrete Naming Certificate for MobiusTransform† **27384**

5111 A Concrete Naming Certificate for ConstructiveHeineBorelInterval† **27387**

5112 A Concrete Naming Certificate for QuadraticForm† **27390**

5113 A Concrete Naming Certificate for RealStepFunction† **27393**

5114 A Concrete Naming Certificate for RegularCauchyNonzero† **27396**

5115 A Concrete Naming Certificate for RealUniformEntourageBasis† **27399**

5116 A Concrete Naming Certificate for UniformLebesgueNumber† **27402**

5117 A Concrete Naming Certificate for UniformlyCauchyFilter† **27405**

511.8 Concrete Naming Certificate for RealCauchyRate↑ **27409**

511.9 Concrete Naming Certificate for BishopCompletionIdempotenceWitness↑ **27411**

512.0 Concrete Naming Certificate for MetricReverseTriangleInequality↑ **27414**

512.1 Concrete Naming Certificate for UpperRealEnvelope↑ **27417**

512.2 Concrete Naming Certificate for LegitimateClaimAuditLedger↑ **27420**

512.3 Concrete Naming Certificate for PerronFrobenius↑ **27423**

512.4 Concrete Naming Certificate for BishopIntervalCover↑ **27426**

512.5 Concrete Naming Certificate for CauchySubsequenceExtractor↑ **27429**

512.6 Concrete Naming Certificate for ParallelDiamondResidual↑ **27436**

512.7 Concrete Naming Certificate for GramDeterminant↑ **27443**

512.8 Concrete Naming Certificate for ModulusOfUniformContinuity↑ **27445**

512.9 Concrete Naming Certificate for ModulusOfContinuity↑ **27448**

513.0 Concrete Naming Certificate for DirichletBoxApproximation↑ **27451**

513.1 Concrete Naming Certificate for RiemannStieltjesIntegrationParts↑ **27455**

513.2 Concrete Naming Certificate for BishopBarInductionModulus↑ **27457**

513.3 Concrete Naming Certificate for DyadicGridProjection↑ **27461**

513.4 Concrete Naming Certificate for RegularCauchyRoundingProjection↑ **27464**

513.5 Concrete Naming Certificate for FiniteDifference↑ **27467**

513.6 Concrete Naming Certificate for FiniteDimensionalSpectralGap↑ **27471**

513.7 Concrete Naming Certificate for SpectralGap↑ **27474**

513.8 Concrete Naming Certificate for CauchyRootCriterion↑ **27477**

513.9 Concrete Naming Certificate for CauchySequentialCriterion↑ **27480**

514.0 Concrete Naming Certificate for InactiveLayerQuotientDynamics↑ **27483**

514.1 Concrete Naming Certificate for CauchyFilterModulus↑ **27486**

514.2 Concrete Naming Certificate for LocallyConvexCompletion↑ **27490**

514.3 Concrete Naming Certificate for MetaCICNormalEndpointCut↑ **27493**

5144 Concrete Naming Certificate for ExteriorAlgebra$^{\uparrow}$ **27496**

5145 Concrete Naming Certificate for FiniteWindowCompactness$^{\uparrow}$ **27500**

5146 Concrete Naming Certificate for Rademacher$^{\uparrow}$ **27503**

5147 Concrete Naming Certificate for RegularTopologicalSpace$^{\uparrow}$ **27510**

5148 Concrete Naming Certificate for CauchyTailGluing$^{\uparrow}$ **27513**

5149 Concrete Naming Certificate for BishopCauchyCriterionCompletion$^{\uparrow}$ **27516**

5150 Concrete Naming Certificate for SignedDecimalExpansion$^{\uparrow}$ **27519**

5151 Concrete Naming Certificate for BishopCauchyDisplayCertificate$^{\uparrow}$ **27522**

5152 Concrete Naming Certificate for SequentiallyCompactCompletion$^{\uparrow}$ **27525**

5153 Concrete Naming Certificate for LocaleSpace$^{\uparrow}$ **27529**

5154 Concrete Naming Certificate for DiniUniformMonotone$^{\uparrow}$ **27532**

5155 Concrete Naming Certificate for CompleteLattice$^{\uparrow}$ **27537**

5156 Concrete Naming Certificate for PointwiseErgodic$^{\uparrow}$ **27540**

5157 Concrete Naming Certificate for ProofTraceNormalForm$^{\uparrow}$ **27543**

5158 Concrete Naming Certificate for RegularCauchyFunctor$^{\uparrow}$ **27547**

5159 Concrete Naming Certificate for HyperbolicIsometryGroup$^{\uparrow}$ **27550**

5160 Concrete Naming Certificate for BoundedSelfAdjointOperator$^{\uparrow}$ **27553**

5161 Concrete Naming Certificate for CompletelyRegularSpace$^{\uparrow}$ **27556**

5162 Concrete Naming Certificate for LawsonTopology$^{\uparrow}$ **27569**

5163 Concrete Naming Certificate for CandidateMediatedSN$^{\uparrow}$ **27571**

5164 Concrete Naming Certificate for CalculusOfConstructions$^{\uparrow}$ **27576**

5165 Concrete Naming Certificate for PrimeNumberTheoremChebyshevWindow$^{\uparrow}$ **27579**

5166 Concrete Naming Certificate for PriestleyDuality$^{\uparrow}$ **27583**

5167 Concrete Naming Certificate for VitaliConvergence$^{\uparrow}$ **27585**

5168 Concrete Naming Certificate for BishopDyadicSqueeze$^{\uparrow}$ **27589**

5169 Concrete Naming Certificate for GalerkinProjection$^{\uparrow}$ **27592**

5170 A Concrete Naming Certificate for HigsonCompactification↑ **27595**

5171 A Concrete Naming Certificate for HigsonCorona↑ **27598**

5172 A Concrete Naming Certificate for ModulusCompositionSchedule↑ **27601**

5173 A Concrete Naming Certificate for LowDiscrepancySequence↑ **27604**

5174 A Concrete Naming Certificate for MetaCICConfluenceBudgetWitness↑ **27607**

5175 A Concrete Naming Certificate for LawOfTotalVariance↑ **27610**

5176 A Concrete Naming Certificate for FrechetBaireSpace↑ **27614**

5177 A Concrete Naming Certificate for ChannelGeneticCodeBridge↑ **27617**

5178 A Concrete Naming Certificate for FejerTheorem↑ **27622**

5179 A Concrete Naming Certificate for RegularCauchyTailEquivalence↑ **27625**

5180 A Concrete Naming Certificate for HyperbolicObserverShell↑ **27628**

5181 A Concrete Naming Certificate for DiracPointMass↑ **27631**

5182 A Concrete Naming Certificate for DiracSequence↑ **27633**

5183 A Concrete Naming Certificate for LocatedCauchyBracket↑ **27636**

5184 A Concrete Naming Certificate for CompactMetricHyperspace↑ **27639**

5185 A Concrete Naming Certificate for BishopIntervalNetExtraction↑ **27642**

5186 A Concrete Naming Certificate for SigmaAlgebra↑ **27645**

5187 A Concrete Naming Certificate for RegularCauchyModulusCompression↑ **27648**

5188 A Concrete Naming Certificate for QuotientNormedSpace↑ **27651**

5189 A Concrete Naming Certificate for CompactNetRadiusChain↑ **27654**

5190 A Concrete Naming Certificate for DyadicBracketApproximation↑ **27656**

5191 A Concrete Naming Certificate for LogicContradictionRefusal↑ **27660**

5192 A Concrete Naming Certificate for WeakStarTopology↑ **27663**

5193 A Concrete Naming Certificate for ObserverLocalityRateBudget↑ **27666**

5194 Mazur–Ulam Isometry NameCert Construction **27669**

5195 A Concrete Naming Certificate for RegularCauchyBinarySplitting↑ **27672**

5196 A Concrete Naming Certificate for RieszInterpolation↑ **27675**

5197 A Concrete Naming Certificate for Wasserstein↑ **27677**

5198 A Concrete Naming Certificate for CauchyProductRegSeq↑ **27682**

5199 A Concrete Naming Certificate for OsgoodUniqueness↑ **27686**

5200 A Concrete Naming Certificate for GromovHausdorffDistance↑ **27689**

5201 A Concrete Naming Certificate for FrechetRiesz↑ **27694**

5202 A Concrete Naming Certificate for CauchyRateNormalForm↑ **27697**

5203 A Concrete Naming Certificate for FiniteCoverRadiusStability↑ **27700**

5204 A Concrete Naming Certificate for ConstructiveUltrametricCompletion↑ **27703**

5205 A Concrete Naming Certificate for MachineExportBoundary↑ **27707**

5206 A Concrete Naming Certificate for RegularCauchyTailPairing↑ **27715**

5207 A Concrete Naming Certificate for RegulatedMeasure↑ **27718**

5208 A Concrete Naming Certificate for CompactIntervalChoiceFree↑ **27721**

5209 A Concrete Naming Certificate for FiniteMetricEntropy↑ **27724**

5210 A Concrete Naming Certificate for MetaCICCandidateNormalizationRouter↑ **27727**

5211 A Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **27730**

5212 A Concrete Naming Certificate for BoundedLinearFunctional↑ **27733**

5213 A Concrete Naming Certificate for CompletionModulusCategory↑ **27736**

5214 A Concrete Naming Certificate for MarkovTriggerDynamics↑ **27739**

5215 A Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **27742**

5216 A Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **27745**

5217 A Concrete Naming Certificate for DiniUniformLimitModulus↑ **27748**

5218 A Concrete Naming Certificate for SummationByParts↑ **27751**

5219 A Concrete Naming Certificate for RealLineOrder↑ **27754**

5220 A Concrete Naming Certificate for SchwartzKernel↑ **27757**

5221 A Concrete Naming Certificate for SynonymousTransition↑ **27760**

5222 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **27763**

5223 Concrete Naming Certificate for LipschitzRealMap↑ **27766**

5224 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **27769**

5225 Concrete Naming Certificate for Dendrite↑ **27773**

5226 Concrete Naming Certificate for YoungIntegration↑ **27776**

5227 Concrete Naming Certificate for VariationDiminishing↑ **27780**

5228 Concrete Naming Certificate for FrobeniusReciprocity↑ **27783**

5229 Concrete Naming Certificate for NormalOperator↑ **27786**

5230 Concrete Naming Certificate for HausdorffMeasure↑ **27789**

5231 Concrete Naming Certificate for LocallyConnected↑ **27792**

5232 Concrete Naming Certificate for BishopCauchyMean↑ **27795**

5233 Concrete Naming Certificate for BishopFixedPointInterval↑ **27798**

5234 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **27801**

5235 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **27804**

5236 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **27807**

5237 Concrete Naming Certificate for HellyBraySelection↑ **27810**

5238 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **27814**

5239 Concrete Naming Certificate for FiniteCauchyFilter↑ **27817**

5240 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **27819**

5241 Concrete Naming Certificate for RegulatedConvergence↑ **27822**

5242 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **27825**

5243 Concrete Naming Certificate for CauchyTailRequest↑ **27828**

5244 Concrete Naming Certificate for QuotientLeakageMarkov↑ **27830**

5245 Concrete Naming Certificate for CauchyCriterionUniform↑ **27833**

5246 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **27836**

5247 Concrete Naming Certificate for CauchyModulusSubsequence↑ **27839**

5248 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **27842**

5249 Concrete Naming Certificate for BorsukUlam↑ **27845**

5250 Concrete Naming Certificate for DenjoyYoungSaks↑ **27848**

5251 Concrete Naming Certificate for UniformlyConvexSpace↑ **27852**

5252 Concrete Naming Certificate for HyperbolicTranslationLength↑ **27855**

5253 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **27858**

5254 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **27860**

5255 Concrete Naming Certificate for UniformInverseModulus↑ **27863**

5256 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **27866**

5257 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **27869**

5258 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **27872**

5259 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **27874**

5260 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **27877**

5261 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **27880**

5262 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **27884**

5263 Concrete Naming Certificate for OneSidedLimit↑ **27887**

5264 Concrete Naming Certificate for BertrandSeriesTest↑ **27890**

5265 Concrete Naming Certificate for RationalCanonicalForm↑ **27894**

5266 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **27897**

5267 Concrete Naming Certificate for ObservabilityGramian↑ **27899**

5267.1 Carrier definition 27899

5267.2 Certificate surface 27900

5268 Concrete Naming Certificate for HelmholtzDecomposition↑ **27902**

5269 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **27905**

5270 Concrete Naming Certificate for BoundedLinearFunctional↑ **27908**

5271 Concrete Naming Certificate for CompletionModulusCategory↑ **27911**

5272 Concrete Naming Certificate for MarkovTriggerDynamics↑ **27914**

5273 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **27917**

5274 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **27920**

5275 Concrete Naming Certificate for DiniUniformLimitModulus↑ **27923**

5276 Concrete Naming Certificate for SummationByParts↑ **27926**

5277 Concrete Naming Certificate for RealLineOrder↑ **27929**

5278 Concrete Naming Certificate for SchwartzKernel↑ **27932**

5279 Concrete Naming Certificate for SynonymousTransition↑ **27935**

5280 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **27938**

5281 Concrete Naming Certificate for LipschitzRealMap↑ **27941**

5282 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **27944**

5283 Concrete Naming Certificate for Dendrite↑ **27948**

5284 Concrete Naming Certificate for YoungIntegration↑ **27951**

5285 Concrete Naming Certificate for VariationDiminishing↑ **27955**

5286 Concrete Naming Certificate for FrobeniusReciprocity↑ **27958**

5287 Concrete Naming Certificate for NormalOperator↑ **27961**

5288 Concrete Naming Certificate for HausdorffMeasure↑ **27964**

5289 Concrete Naming Certificate for LocallyConnected↑ **27967**

5290 Concrete Naming Certificate for BishopCauchyMean↑ **27970**

5291 Concrete Naming Certificate for BishopFixedPointInterval↑ **27973**

5292 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **27976**

5293 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **27979**

5294 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **27982**

5295 Concrete Naming Certificate for HellyBraySelection↑ **27985**

5296 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **27989**

5297 Concrete Naming Certificate for FiniteCauchyFilter↑ **27992**

5298 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **27994**

5299 A Concrete Naming Certificate for RegulatedConvergence↑ **27997**

5300 A Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **28000**

5301 A Concrete Naming Certificate for CauchyTailRequest↑ **28003**

5302 A Concrete Naming Certificate for QuotientLeakageMarkov↑ **28005**

5303 A Concrete Naming Certificate for CauchyCriterionUniform↑ **28008**

5304 A Concrete Naming Certificate for CauchyProductCompletionProjection↑ **28011**

5305 A Concrete Naming Certificate for CauchyModulusSubsequence↑ **28014**

5306 A Concrete Naming Certificate for RegularCauchySequenceComparison↑ **28017**

5307 A Concrete Naming Certificate for CauchyCompletionMonadicity↑ **28022**

5308 A Concrete Naming Certificate for BorsukUlam↑ **28025**

5309 A Concrete Naming Certificate for DenjoyYoungSaks↑ **28028**

5310 A Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **28032**

5311 A Concrete Naming Certificate for UniformlyConvexSpace↑ **28036**

5312 A Concrete Naming Certificate for HyperbolicTranslationLength↑ **28039**

5313 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **28042**

5314 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **28044**

5315 A Concrete Naming Certificate for UniformInverseModulus↑ **28047**

5316 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **28050**

5317 A Concrete Naming Certificate for FreeGroup↑ **28053**

5318 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **28056**

5319 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **28060**

5320 A Concrete Naming Certificate for LocatedCauchyFilterBasis **28063**

5321 A Concrete Naming Certificate for OneSidedLimit↑ **28067**

5322 A Concrete Naming Certificate for BertrandSeriesTest↑ **28070**

5323 A Concrete Naming Certificate for RationalCanonicalForm↑ **28074**

5324 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **28077**

5325 A Concrete Naming Certificate for RealizerSpace↑ **28079**

5326 A Concrete Naming Certificate for RuleOneTenCausalCone↑ **28082**

5327 A Concrete Naming Certificate for BaireTreeModulus↑ **28084**

5328 Located Uniform Cauchy Principle NameCert Construction **28087**

5329 A Concrete Naming Certificate for BaireSpaceCompactOpen↑ **28090**

5330 A Concrete Naming Certificate for CompactApartnessProfile↑ **28093**

5331 A Concrete Naming Certificate for RiemannIntegralCauchy↑ **28096**

5332 A Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **28099**

5333 A Concrete Naming Certificate for BoundedRegulatedFunction↑ **28102**

5334 A Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **28105**

5335 A Concrete Naming Certificate for ArensEellsSpace↑ **28109**

5336 A Concrete Naming Certificate for BanachFixedPointStability↑ **28113**

5337 A Concrete Naming Certificate for RealTriangleInequality↑ **28116**

5338 A Concrete Naming Certificate for LowerDarbouxSum↑ **28118**

5339 A Concrete Naming Certificate for BoundedMetricSpace↑ **28121**

5340 A Concrete Naming Certificate for LegendreFenchelEnvelope↑ **28124**

5341 A Concrete Naming Certificate for PrefixTree↑ **28126**

5342 A Concrete Naming Certificate for ChomskyNormalForm↑ **28129**

5343 A Concrete Naming Certificate for FiniteFanTheorem↑ **28132**

5344 A Concrete Naming Certificate for SpecializationOrder↑ **28136**

5345 A Concrete Naming Certificate for FiniteModulusReal↑ **28139**

5346 A Concrete Naming Certificate for CompactUniformSubordinateCover↑ **28142**

5347 A Concrete Naming Certificate for BishopIntervalContraction↑ **28144**

5348 A Concrete Naming Certificate for SeminormedSpace↑ **28149**

5349 A Concrete Naming Certificate for ReassignmentMassSubcube↑ **28152**

5350 A Concrete Naming Certificate for RealBolzanoWeierstrass↑ **28155**

5351 Concrete Naming Certificate for CandidateSetClosedRoute↑ **28158**

5352 Concrete Naming Certificate for CauchyInterlacing↑ **28161**

5353 Concrete Naming Certificate for BisectionReal↑ **28164**

5354 Concrete Naming Certificate for DarbouxCriterion↑ **28168**

5355 Concrete Naming Certificate for RealLocallyCompact↑ **28171**

5356 Concrete Naming Certificate for CompactUniformModulusStability↑ **28174**

5357 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **28177**

5358 Concrete Naming Certificate for CauchyApproximationTree↑ **28182**

5359 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **28185**

5360 Concrete Naming Certificate for SeparableSpace↑ **28188**

5361 Concrete Naming Certificate for DedekindCutStream↑ **28191**

5362 Concrete Naming Certificate for StokesTheorem↑ **28195**

5363 Concrete Naming Certificate for MetricCompletionExistence↑ **28198**

5364 Concrete Naming Certificate for FiniteLimitSeal↑ **28206**

5365 Concrete Naming Certificate for FormalBallDomain↑ **28212**

5366 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **28216**

5367 Concrete Naming Certificate for RieszThorinInterpolation↑ **28220**

5368 Concrete Naming Certificate for HoeffdingFiniteSum↑ **28223**

5369 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **28226**

5370 Concrete Naming Certificate for CauchyModulusBridge↑ **28229**

5371 Concrete Naming Certificate for MyhillNerode↑ **28232**

5372 Concrete Naming Certificate for VariationSeminorm↑ **28235**

5373 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **28238**

5374 Concrete Naming Certificate for BoundedLinearFunctional↑ **28241**

5375 Concrete Naming Certificate for CompletionModulusCategory↑ **28244**

5376 Concrete Naming Certificate for MarkovTriggerDynamics↑ **28247**

537A Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **28250**

537B Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **28253**

537C Concrete Naming Certificate for DiniUniformLimitModulus↑ **28256**

538D Concrete Naming Certificate for SummationByParts↑ **28259**

538E Concrete Naming Certificate for RealLineOrder↑ **28262**

538F Concrete Naming Certificate for SchwartzKernel↑ **28265**

538G Concrete Naming Certificate for SynonymousTransition↑ **28268**

538H Concrete Naming Certificate for CentralLimitFiniteWindow↑ **28271**

538I Concrete Naming Certificate for LipschitzRealMap↑ **28274**

538J Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **28277**

538K Concrete Naming Certificate for Dendrite↑ **28281**

538L Concrete Naming Certificate for YoungIntegration↑ **28284**

538M Concrete Naming Certificate for VariationDiminishing↑ **28288**

539N Concrete Naming Certificate for FrobeniusReciprocity↑ **28291**

539O Concrete Naming Certificate for NormalOperator↑ **28294**

539P Concrete Naming Certificate for HausdorffMeasure↑ **28297**

539Q Concrete Naming Certificate for LocallyConnected↑ **28300**

539R Concrete Naming Certificate for BishopCauchyMean↑ **28303**

539S Concrete Naming Certificate for BishopFixedPointInterval↑ **28306**

539T Concrete Naming Certificate for PrivateCoordinateTrigger↑ **28309**

539U Concrete Naming Certificate for LocatedRealComparisonModulus↑ **28312**

539V Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **28315**

539W Concrete Naming Certificate for HellyBraySelection↑ **28318**

540X Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **28322**

540Y Concrete Naming Certificate for FiniteCauchyFilter↑ **28325**

540Z Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **28327**

5403 Concrete Naming Certificate for RegulatedConvergence↑ **28330**

5404 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **28333**

5405 Concrete Naming Certificate for CauchyTailRequest↑ **28336**

5406 Concrete Naming Certificate for QuotientLeakageMarkov↑ **28338**

5407 Concrete Naming Certificate for CauchyCriterionUniform↑ **28341**

5408 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **28344**

5409 Concrete Naming Certificate for CauchyModulusSubsequence↑ **28347**

5410 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **28350**

5411 Concrete Naming Certificate for BorsukUlam↑ **28353**

5412 Concrete Naming Certificate for DenjoyYoungSaks↑ **28356**

5413 Concrete Naming Certificate for UniformlyConvexSpace↑ **28360**

5414 Concrete Naming Certificate for HyperbolicTranslationLength↑ **28363**

5415 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **28366**

5416 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **28368**

5417 Concrete Naming Certificate for UniformInverseModulus↑ **28371**

5418 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **28374**

5419 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **28377**

5420 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **28380**

5421 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **28382**

5422 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **28385**

5423 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **28388**

5424 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **28392**

5425 Concrete Naming Certificate for OneSidedLimit↑ **28395**

5426 Concrete Naming Certificate for BertrandSeriesTest↑ **28398**

5427 Concrete Naming Certificate for RationalCanonicalForm↑ **28402**

5428 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **28405**

5429 A Concrete Naming Certificate for ObservabilityGramian$^{\uparrow}$ **28407**
5429.1 Carrier definition 28407
5429.2 Certificate surface 28408

5430 A Concrete Naming Certificate for HelmholtzDecomposition$^{\uparrow}$ **28410**

5431 A Concrete Naming Certificate for HausdorffCompletionEnvelope$^{\uparrow}$ **28413**

5432 A Concrete Naming Certificate for BoundedLinearFunctional$^{\uparrow}$ **28416**

5433 A Concrete Naming Certificate for CompletionModulusCategory$^{\uparrow}$ **28419**

5434 A Concrete Naming Certificate for MarkovTriggerDynamics$^{\uparrow}$ **28422**

5435 A Concrete Naming Certificate for RegularCauchyLocatedCompletion$^{\uparrow}$ **28425**

5436 A Concrete Naming Certificate for PicardThresholdCauchyHandoff$^{\uparrow}$ **28428**

5437 A Concrete Naming Certificate for DiniUniformLimitModulus$^{\uparrow}$ **28431**

5438 A Concrete Naming Certificate for SummationByParts$^{\uparrow}$ **28434**

5439 A Concrete Naming Certificate for RealLineOrder$^{\uparrow}$ **28437**

5440 A Concrete Naming Certificate for SchwartzKernel$^{\uparrow}$ **28440**

5441 A Concrete Naming Certificate for SynonymousTransition$^{\uparrow}$ **28443**

5442 A Concrete Naming Certificate for CentralLimitFiniteWindow$^{\uparrow}$ **28446**

5443 A Concrete Naming Certificate for LipschitzRealMap$^{\uparrow}$ **28449**

5444 A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$ **28452**

5445 A Concrete Naming Certificate for Dendrite$^{\uparrow}$ **28456**

5446 A Concrete Naming Certificate for YoungIntegration$^{\uparrow}$ **28459**

5447 A Concrete Naming Certificate for VariationDiminishing$^{\uparrow}$ **28463**

5448 A Concrete Naming Certificate for FrobeniusReciprocity$^{\uparrow}$ **28466**

5449 A Concrete Naming Certificate for NormalOperator$^{\uparrow}$ **28469**

5450 A Concrete Naming Certificate for HausdorffMeasure$^{\uparrow}$ **28472**

5451 A Concrete Naming Certificate for LocallyConnected$^{\uparrow}$ **28475**

5452 A Concrete Naming Certificate for BishopCauchyMean$^{\uparrow}$ **28478**

5453 A Concrete Naming Certificate for BishopFixedPointInterval$^{\uparrow}$ **28481**

5454 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **28484**

5455 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **28487**

5456 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **28490**

5457 Concrete Naming Certificate for HellyBraySelection↑ **28493**

5458 Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **28497**

5459 Concrete Naming Certificate for FiniteCauchyFilter↑ **28500**

5460 Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **28502**

5461 Concrete Naming Certificate for RegulatedConvergence↑ **28505**

5462 Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **28508**

5463 Concrete Naming Certificate for CauchyTailRequest↑ **28511**

5464 Concrete Naming Certificate for QuotientLeakageMarkov↑ **28513**

5465 Concrete Naming Certificate for CauchyCriterionUniform↑ **28516**

5466 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **28519**

5467 Concrete Naming Certificate for CauchyModulusSubsequence↑ **28522**

5468 Concrete Naming Certificate for RegularCauchySequenceComparison↑ **28525**

5469 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **28530**

5470 Concrete Naming Certificate for BorsukUlam↑ **28533**

5471 Concrete Naming Certificate for DenjoyYoungSaks↑ **28536**

5472 Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **28540**

5473 Concrete Naming Certificate for UniformlyConvexSpace↑ **28544**

5474 Concrete Naming Certificate for HyperbolicTranslationLength↑ **28547**

5475 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **28550**

5476 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **28552**

5477 Concrete Naming Certificate for UniformInverseModulus↑ **28555**

5478 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **28558**

5479 Concrete Naming Certificate for FreeGroup↑ **28561**

5480 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **28564**

5481 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **28568**

5482 A Concrete Naming Certificate for LocatedCauchyFilterBasis **28571**

5483 A Concrete Naming Certificate for OneSidedLimit↑ **28575**

5484 A Concrete Naming Certificate for BertrandSeriesTest↑ **28578**

5485 A Concrete Naming Certificate for RationalCanonicalForm↑ **28582**

5486 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **28585**

5487 A Concrete Naming Certificate for RealizerSpace↑ **28587**

5488 A Concrete Naming Certificate for RuleOneTenCausalCone↑ **28590**

5489 A Concrete Naming Certificate for BaireTreeModulus↑ **28592**

5490 Located Uniform Cauchy Principle NameCert Construction **28595**

5491 A Concrete Naming Certificate for BaireSpaceCompactOpen↑ **28598**

5492 A Concrete Naming Certificate for CompactApartnessProfile↑ **28601**

5493 A Concrete Naming Certificate for RiemannIntegralCauchy↑ **28604**

5494 A Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **28607**

5495 A Concrete Naming Certificate for BoundedRegulatedFunction↑ **28610**

5496 A Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **28613**

5497 A Concrete Naming Certificate for ArensEellsSpace↑ **28617**

5498 A Concrete Naming Certificate for BanachFixedPointStability↑ **28621**

5499 A Concrete Naming Certificate for RealTriangleInequality↑ **28624**

5500 A Concrete Naming Certificate for LowerDarbouxSum↑ **28626**

5501 A Concrete Naming Certificate for BoundedMetricSpace↑ **28629**

5502 A Concrete Naming Certificate for LegendreFenchelEnvelope↑ **28632**

5503 A Concrete Naming Certificate for PrefixTree↑ **28634**

5504 A Concrete Naming Certificate for ChomskyNormalForm↑ **28637**

5505 A Concrete Naming Certificate for FiniteFanTheorem↑ **28640**

5506 A Concrete Naming Certificate for SpecializationOrder↑ **28644**

5507 A Concrete Naming Certificate for FiniteModulusReal↑ **28647**

5508 A Concrete Naming Certificate for CompactUniformSubordinateCover↑ **28650**

5509 A Concrete Naming Certificate for BishopIntervalContraction↑ **28652**

5510 A Concrete Naming Certificate for SeminormedSpace↑ **28657**

5511 A Concrete Naming Certificate for ReassignmentMassSubcube↑ **28660**

5512 A Concrete Naming Certificate for CauchyInterlacing↑ **28663**

5513 A Concrete Naming Certificate for RealLocallyCompact↑ **28666**

5514 A Concrete Naming Certificate for CompactUniformModulusStability↑ **28669**

5515 A Concrete Naming Certificate for CauchyFilterSelector↑ **28672**

5516 A Concrete Naming Certificate for RegularCauchySandwich↑ **28675**

5517 A Concrete Naming Certificate for DynamicProgramming↑ **28679**

5518 Regular Cauchy Even-Odd Substreams NameCert Construction **28682**

5519 A Concrete Naming Certificate for SchwartzSpace↑ **28686**

5520 A Concrete Naming Certificate for RuleOneTenPrivateSharedDecoration↑ **28689**

5521 A Concrete Naming Certificate for ConstructiveCantorTheorem↑ **28692**

5522 Markov-Kakutani Fixed Point NameCert Construction **28695**

5523 Window-Six Private/Shared Decoration NameCert Construction **28698**

5524 A Concrete Naming Certificate for SchurLemma↑ **28702**

5525 A Concrete Naming Certificate for CompactRealIntervalMinimum↑ **28704**

5526 A Concrete Naming Certificate for SimpleFunctionApproximation↑ **28707**

5527 A Concrete Naming Certificate for DyadicModulus↑ **28710**

5528 A Concrete Naming Certificate for CauchyFilterSelector↑ **28713**

5529 A Concrete Naming Certificate for CauchyModulusSpace↑ **28716**

5530 A Concrete Naming Certificate for CauchyCompletionLeftExactness↑ **28720**

5531 A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection↑ **28729**

5532 Concrete Naming Certificate for BarRecursion↑ **28732**

5533 Concrete Naming Certificate for SnakeLemma↑ **28735**

5534 Concrete Naming Certificate for RealTailDiameterGauge↑ **28738**

5535 Concrete Naming Certificate for EffectiveRealCompactIntervalSearch↑ **28741**

5536 Concrete Naming Certificate for CauchyCompletionDenseEmbedding↑ **28747**

5537 Concrete Naming Certificate for RiemannRefinementCriterion↑ **28750**

5538 Concrete Naming Certificate for CompactUniformModulusComposition↑ **28753**

5539 Concrete Naming Certificate for PrivateSharedDecorationModel↑ **28756**

5540 Concrete Naming Certificate for BishopLocatedRealFilter↑ **28758**

5541 Concrete Naming Certificate for RegularCauchyDiagonalArgument↑ **28761**

5542 Concrete Naming Certificate for PrivateSharedCoordinateDecomposition↑ **28764**

5543 Concrete Naming Certificate for RegulatedCauchyModulus↑ **28767**

5544 Concrete Naming Certificate for RegularCauchyMinModulus↑ **28770**

5545 Concrete Naming Certificate for TaylorTheorem↑ **28777**

5546 Located Real Interval Bisection NameCert Construction **28780**

5547 MetaCIC Candidate Normalization Confluence Handoff NameCert Construction **28783**

5548 Concrete Naming Certificate for CantorIntersectionModulus↑ **28831**

5549 Concrete Naming Certificate for CompactUniformContinuityLedger↑ **28834**

5550 Concrete Naming Certificate for FrechetMetricEmbedding↑ **28837**

5551 Concrete Naming Certificate for CaristiFixedPoint↑ **28839**

5552 Concrete Naming Certificate for WeightedDeformationSpine↑ **28842**

5553 Concrete Naming Certificate for CompactRealInterval↑ **28845**

5554 Concrete Naming Certificate for LocatedCutCompletion↑ **28850**

5555 Concrete Naming Certificate for DedekindEtaCompletion↑ **28854**

5556 Concrete Naming Certificate for GITQuotient↑ **28857**

5557 Concrete Naming Certificate for AubinLions↑ **28860**

5558 Concrete Naming Certificate for CompactUniformPullback↑ **28863**

5559 Concrete Naming Certificate for RegularCauchyGluing↑ **28866**

5560 Concrete Naming Certificate for YoungSymmetrizer↑ **28877**

5561 Concrete Naming Certificate for MetaCICSRKernel↑ **28880**

5562 Concrete Naming Certificate for RealApartnessField↑ **28884**

5563 Concrete Naming Certificate for ModulusOfRegularConvergence↑ **28894**

5564 Concrete Naming Certificate for RHRouteHaltingBoundary↑ **28897**

5565 Concrete Naming Certificate for Premetric↑ **28900**

5566 Concrete Naming Certificate for FiniteDimensionalNormEquivalence↑ **28903**

5567 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **28906**

5568 Concrete Naming Certificate for CauchyModulusBridge↑ **28909**

5569 Concrete Naming Certificate for MyhillNerode↑ **28912**

5570 Concrete Naming Certificate for VariationSeminorm↑ **28915**

5571 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **28918**

5572 Concrete Naming Certificate for BoundedLinearFunctional↑ **28921**

5573 Concrete Naming Certificate for CompletionModulusCategory↑ **28924**

5574 Concrete Naming Certificate for MarkovTriggerDynamics↑ **28927**

5575 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **28930**

5576 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **28933**

5577 Concrete Naming Certificate for DiniUniformLimitModulus↑ **28936**

5578 Concrete Naming Certificate for SummationByParts↑ **28939**

5579 Concrete Naming Certificate for RealLineOrder↑ **28942**

5580 Concrete Naming Certificate for SchwartzKernel↑ **28945**

5581 Concrete Naming Certificate for SynonymousTransition↑ **28948**

5582 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **28951**

5583 Concrete Naming Certificate for LipschitzRealMap↑ **28954**

5584 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$ **28957**

5585 Concrete Naming Certificate for Dendrite$^{\uparrow}$ **28961**

5586 Concrete Naming Certificate for YoungIntegration$^{\uparrow}$ **28964**

5587 Concrete Naming Certificate for VariationDiminishing$^{\uparrow}$ **28968**

5588 Concrete Naming Certificate for FrobeniusReciprocity$^{\uparrow}$ **28971**

5589 Concrete Naming Certificate for NormalOperator$^{\uparrow}$ **28974**

5590 Concrete Naming Certificate for HausdorffMeasure$^{\uparrow}$ **28977**

5591 Concrete Naming Certificate for LocallyConnected$^{\uparrow}$ **28980**

5592 Concrete Naming Certificate for BishopCauchyMean$^{\uparrow}$ **28983**

5593 Concrete Naming Certificate for BishopFixedPointInterval$^{\uparrow}$ **28986**

5594 Concrete Naming Certificate for PrivateCoordinateTrigger$^{\uparrow}$ **28989**

5595 Concrete Naming Certificate for LocatedRealComparisonModulus$^{\uparrow}$ **28992**

5596 Concrete Naming Certificate for BoundaryShadowExpansionCocycle$^{\uparrow}$ **28995**

5597 Concrete Naming Certificate for HellyBraySelection$^{\uparrow}$ **28998**

5598 Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$ **29002**

5599 Concrete Naming Certificate for FiniteCauchyFilter$^{\uparrow}$ **29005**

5600 Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$ **29007**

5601 Concrete Naming Certificate for RegulatedConvergence$^{\uparrow}$ **29010**

5602 Concrete Naming Certificate for LocatedNestedIntervalLimit$^{\uparrow}$ **29013**

5603 Concrete Naming Certificate for CauchyTailRequest$^{\uparrow}$ **29016**

5604 Concrete Naming Certificate for QuotientLeakageMarkov$^{\uparrow}$ **29018**

5605 Concrete Naming Certificate for CauchyCriterionUniform$^{\uparrow}$ **29021**

5606 Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$ **29024**

5607 Concrete Naming Certificate for CauchyModulusSubsequence$^{\uparrow}$ **29027**

5608 Concrete Naming Certificate for CauchyCompletionMonadicity$^{\uparrow}$ **29030**

5609 Concrete Naming Certificate for BorsukUlam$^{\uparrow}$ **29033**

5610 Concrete Naming Certificate for DenjoyYoungSaks↑ **29036**

5611 Concrete Naming Certificate for UniformlyConvexSpace↑ **29040**

5612 Concrete Naming Certificate for HyperbolicTranslationLength↑ **29043**

5613 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **29046**

5614 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **29048**

5615 Concrete Naming Certificate for UniformInverseModulus↑ **29051**

5616 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **29054**

5617 Concrete Naming Certificate for RealCompletionWindowFunctor↑ **29057**

5618 Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **29060**

5619 Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **29062**

5620 Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **29065**

5621 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **29068**

5622 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **29072**

5623 Concrete Naming Certificate for OneSidedLimit↑ **29075**

5624 Concrete Naming Certificate for BertrandSeriesTest↑ **29078**

5625 Concrete Naming Certificate for RationalCanonicalForm↑ **29082**

5626 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **29085**

5627 Concrete Naming Certificate for ObservabilityGramian↑ **29087**

5627.1 Carrier definition 29087

5627.2 Certificate surface 29088

5628 Concrete Naming Certificate for HelmholtzDecomposition↑ **29090**

5629 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **29093**

5630 Concrete Naming Certificate for BoundedLinearFunctional↑ **29096**

5631 Concrete Naming Certificate for CompletionModulusCategory↑ **29099**

5632 Concrete Naming Certificate for MarkovTriggerDynamics↑ **29102**

5633 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **29105**

5634 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **29108**

5635 Concrete Naming Certificate for DiniUniformLimitModulus$^{\uparrow}$ **29111**

5636 Concrete Naming Certificate for SummationByParts$^{\uparrow}$ **29114**

5637 Concrete Naming Certificate for RealLineOrder$^{\uparrow}$ **29117**

5638 Concrete Naming Certificate for SchwartzKernel$^{\uparrow}$ **29120**

5639 Concrete Naming Certificate for SynonymousTransition$^{\uparrow}$ **29123**

5640 Concrete Naming Certificate for CentralLimitFiniteWindow$^{\uparrow}$ **29126**

5641 Concrete Naming Certificate for LipschitzRealMap$^{\uparrow}$ **29129**

5642 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$ **29132**

5643 Concrete Naming Certificate for Dendrite$^{\uparrow}$ **29136**

5644 Concrete Naming Certificate for YoungIntegration$^{\uparrow}$ **29139**

5645 Concrete Naming Certificate for VariationDiminishing$^{\uparrow}$ **29143**

5646 Concrete Naming Certificate for FrobeniusReciprocity$^{\uparrow}$ **29146**

5647 Concrete Naming Certificate for NormalOperator$^{\uparrow}$ **29149**

5648 Concrete Naming Certificate for HausdorffMeasure$^{\uparrow}$ **29152**

5649 Concrete Naming Certificate for LocallyConnected$^{\uparrow}$ **29155**

5650 Concrete Naming Certificate for BishopCauchyMean$^{\uparrow}$ **29158**

5651 Concrete Naming Certificate for BishopFixedPointInterval$^{\uparrow}$ **29161**

5652 Concrete Naming Certificate for PrivateCoordinateTrigger$^{\uparrow}$ **29164**

5653 Concrete Naming Certificate for LocatedRealComparisonModulus$^{\uparrow}$ **29167**

5654 Concrete Naming Certificate for BoundaryShadowExpansionCocycle$^{\uparrow}$ **29170**

5655 Concrete Naming Certificate for HellyBraySelection$^{\uparrow}$ **29173**

5656 Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$ **29177**

5657 Concrete Naming Certificate for FiniteCauchyFilter$^{\uparrow}$ **29180**

5658 Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$ **29182**

5659 Concrete Naming Certificate for RegulatedConvergence$^{\uparrow}$ **29185**

5660 Concrete Naming Certificate for LocatedNestedIntervalLimit$^{\uparrow}$ **29188**

5661 Concrete Naming Certificate for CauchyTailRequest↑ **29191**

5662 Concrete Naming Certificate for QuotientLeakageMarkov↑ **29193**

5663 Concrete Naming Certificate for CauchyCriterionUniform↑ **29196**

5664 Concrete Naming Certificate for CauchyProductCompletionProjection↑ **29199**

5665 Concrete Naming Certificate for CauchyModulusSubsequence↑ **29202**

5666 Concrete Naming Certificate for RegularCauchySequenceComparison↑ **29205**

5667 Concrete Naming Certificate for CauchyCompletionMonadicity↑ **29210**

5668 Concrete Naming Certificate for BorsukUlam↑ **29213**

5669 Concrete Naming Certificate for DenjoyYoungSaks↑ **29216**

5670 Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **29220**

5671 Concrete Naming Certificate for UniformlyConvexSpace↑ **29224**

5672 Concrete Naming Certificate for HyperbolicTranslationLength↑ **29227**

5673 Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **29230**

5674 Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **29232**

5675 Concrete Naming Certificate for UniformInverseModulus↑ **29235**

5676 Concrete Naming Certificate for BoundedCompleteRealInterval↑ **29238**

5677 Concrete Naming Certificate for FreeGroup↑ **29241**

5678 Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **29244**

5679 Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **29248**

5680 Concrete Naming Certificate for LocatedCauchyFilterBasis **29251**

5681 Concrete Naming Certificate for OneSidedLimit↑ **29255**

5682 Concrete Naming Certificate for BertrandSeriesTest↑ **29258**

5683 Concrete Naming Certificate for RationalCanonicalForm↑ **29262**

5684 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **29265**

5685 Concrete Naming Certificate for RealizerSpace↑ **29267**

5686 Concrete Naming Certificate for RuleOneTenCausalCone↑ **29270**

5687 A Concrete Naming Certificate for BaireTreeModulus↑ **29272**

5688 Located Uniform Cauchy Principle NameCert Construction **29275**

5689 A Concrete Naming Certificate for BaireSpaceCompactOpen↑ **29278**

5690 A Concrete Naming Certificate for CompactApartnessProfile↑ **29281**

5691 A Concrete Naming Certificate for RiemannIntegralCauchy↑ **29284**

5692 A Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **29287**

5693 A Concrete Naming Certificate for BoundedRegulatedFunction↑ **29290**

5694 A Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **29293**

5695 A Concrete Naming Certificate for ArensEellsSpace↑ **29297**

5696 A Concrete Naming Certificate for BanachFixedPointStability↑ **29301**

5697 A Concrete Naming Certificate for RealTriangleInequality↑ **29304**

5698 A Concrete Naming Certificate for LowerDarbouxSum↑ **29306**

5699 A Concrete Naming Certificate for BoundedMetricSpace↑ **29309**

5700 A Concrete Naming Certificate for LegendreFenchelEnvelope↑ **29312**

5701 A Concrete Naming Certificate for PrefixTree↑ **29314**

5702 A Concrete Naming Certificate for ChomskyNormalForm↑ **29317**

5703 A Concrete Naming Certificate for FiniteFanTheorem↑ **29320**

5704 A Concrete Naming Certificate for SpecializationOrder↑ **29324**

5705 A Concrete Naming Certificate for FiniteModulusReal↑ **29327**

5706 A Concrete Naming Certificate for CompactUniformSubordinateCover↑ **29330**

5707 A Concrete Naming Certificate for BishopIntervalContraction↑ **29332**

5708 A Concrete Naming Certificate for SeminormedSpace↑ **29337**

5709 A Concrete Naming Certificate for ReassignmentMassSubcube↑ **29340**

5710 A Concrete Naming Certificate for CauchyInterlacing↑ **29343**

5711 A Concrete Naming Certificate for BisectionReal↑ **29346**

5712 A Concrete Naming Certificate for DarbouxCriterion↑ **29350**

5713 Concrete Naming Certificate for RealLocallyCompact↑ **29353**

5714 Concrete Naming Certificate for CompactUniformModulusStability↑ **29356**

5715 Concrete Naming Certificate for RegularCauchyHeineBorelRoute↑ **29359**

5716 Concrete Naming Certificate for CauchyApproximationTree↑ **29364**

5717 Concrete Naming Certificate for BishopCauchyRepresentationComparison↑ **29367**

5718 Concrete Naming Certificate for SeparableSpace↑ **29370**

5719 Concrete Naming Certificate for DedekindCutStream↑ **29373**

5720 Concrete Naming Certificate for StokesTheorem↑ **29377**

5721 Concrete Naming Certificate for MetricCompletionExistence↑ **29380**

5722 Concrete Naming Certificate for FiniteLimitSeal↑ **29388**

5723 Concrete Naming Certificate for FormalBallDomain↑ **29394**

5724 Concrete Naming Certificate for CauchyCompletionReflectiveLocalization↑ **29398**

5725 Concrete Naming Certificate for RieszThorinInterpolation↑ **29402**

5726 Concrete Naming Certificate for HoeffdingFiniteSum↑ **29405**

5727 Concrete Naming Certificate for RegularCauchyLatticeOperations↑ **29408**

5728 Concrete Naming Certificate for CauchyModulusBridge↑ **29411**

5729 Concrete Naming Certificate for MyhillNerode↑ **29414**

5730 Concrete Naming Certificate for VariationSeminorm↑ **29417**

5731 Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **29420**

5732 Concrete Naming Certificate for BoundedLinearFunctional↑ **29423**

5733 Concrete Naming Certificate for CompletionModulusCategory↑ **29426**

5734 Concrete Naming Certificate for MarkovTriggerDynamics↑ **29429**

5735 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **29432**

5736 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **29435**

5737 Concrete Naming Certificate for DiniUniformLimitModulus↑ **29438**

5738 Concrete Naming Certificate for SummationByParts↑ **29441**

5739 A Concrete Naming Certificate for RealLineOrder↑ **29444**

5740 A Concrete Naming Certificate for SchwartzKernel↑ **29447**

5741 A Concrete Naming Certificate for SynonymousTransition↑ **29450**

5742 A Concrete Naming Certificate for CentralLimitFiniteWindow↑ **29453**

5743 A Concrete Naming Certificate for LipschitzRealMap↑ **29456**

5744 A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **29459**

5745 A Concrete Naming Certificate for Dendrite↑ **29463**

5746 A Concrete Naming Certificate for YoungIntegration↑ **29466**

5747 A Concrete Naming Certificate for VariationDiminishing↑ **29470**

5748 A Concrete Naming Certificate for FrobeniusReciprocity↑ **29473**

5749 A Concrete Naming Certificate for NormalOperator↑ **29476**

5750 A Concrete Naming Certificate for HausdorffMeasure↑ **29479**

5751 A Concrete Naming Certificate for LocallyConnected↑ **29482**

5752 A Concrete Naming Certificate for BishopCauchyMean↑ **29485**

5753 A Concrete Naming Certificate for BishopFixedPointInterval↑ **29488**

5754 A Concrete Naming Certificate for PrivateCoordinateTrigger↑ **29491**

5755 A Concrete Naming Certificate for LocatedRealComparisonModulus↑ **29494**

5756 A Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **29497**

5757 A Concrete Naming Certificate for HellyBraySelection↑ **29500**

5758 A Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **29504**

5759 A Concrete Naming Certificate for FiniteCauchyFilter↑ **29507**

5760 A Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **29509**

5761 A Concrete Naming Certificate for RegulatedConvergence↑ **29512**

5762 A Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **29515**

5763 A Concrete Naming Certificate for CauchyTailRequest↑ **29518**

5764 A Concrete Naming Certificate for QuotientLeakageMarkov↑ **29520**

5765 A Concrete Naming Certificate for CauchyCriterionUniform↑ **29523**

5766 A Concrete Naming Certificate for CauchyProductCompletionProjection↑ **29526**

5767 A Concrete Naming Certificate for CauchyModulusSubsequence↑ **29529**

5768 A Concrete Naming Certificate for CauchyCompletionMonadicity↑ **29532**

5769 A Concrete Naming Certificate for BorsukUlam↑ **29535**

5770 A Concrete Naming Certificate for DenjoyYoungSaks↑ **29538**

5771 A Concrete Naming Certificate for UniformlyConvexSpace↑ **29542**

5772 A Concrete Naming Certificate for HyperbolicTranslationLength↑ **29545**

5773 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **29548**

5774 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **29550**

5775 A Concrete Naming Certificate for UniformInverseModulus↑ **29553**

5776 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **29556**

5777 A Concrete Naming Certificate for RealCompletionWindowFunctor↑ **29559**

5778 A Concrete Naming Certificate for RealCompletenessWindowFunctor↑ **29562**

5779 A Concrete Naming Certificate for CompactRealSubsequenceModulus↑ **29564**

5780 A Concrete Naming Certificate for MetaCICNormalEndpointComparator↑ **29567**

5781 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **29570**

5782 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **29574**

5783 A Concrete Naming Certificate for OneSidedLimit↑ **29577**

5784 A Concrete Naming Certificate for BertrandSeriesTest↑ **29580**

5785 A Concrete Naming Certificate for RationalCanonicalForm↑ **29584**

5786 A Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **29587**

5787 A Concrete Naming Certificate for ObservabilityGramian↑ **29589**

5787.1 Carrier definition 29589

5787.2 Certificate surface 29590

5788 A Concrete Naming Certificate for HelmholtzDecomposition↑ **29592**

5789 A Concrete Naming Certificate for HausdorffCompletionEnvelope↑ **29595**

5790 Concrete Naming Certificate for BoundedLinearFunctional↑ **29598**

5791 Concrete Naming Certificate for CompletionModulusCategory↑ **29601**

5792 Concrete Naming Certificate for MarkovTriggerDynamics↑ **29604**

5793 Concrete Naming Certificate for RegularCauchyLocatedCompletion↑ **29607**

5794 Concrete Naming Certificate for PicardThresholdCauchyHandoff↑ **29610**

5795 Concrete Naming Certificate for DiniUniformLimitModulus↑ **29613**

5796 Concrete Naming Certificate for SummationByParts↑ **29616**

5797 Concrete Naming Certificate for RealLineOrder↑ **29619**

5798 Concrete Naming Certificate for SchwartzKernel↑ **29622**

5799 Concrete Naming Certificate for SynonymousTransition↑ **29625**

5800 Concrete Naming Certificate for CentralLimitFiniteWindow↑ **29628**

5801 Concrete Naming Certificate for LipschitzRealMap↑ **29631**

5802 Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow↑ **29634**

5803 Concrete Naming Certificate for Dendrite↑ **29638**

5804 Concrete Naming Certificate for YoungIntegration↑ **29641**

5805 Concrete Naming Certificate for VariationDiminishing↑ **29645**

5806 Concrete Naming Certificate for FrobeniusReciprocity↑ **29648**

5807 Concrete Naming Certificate for NormalOperator↑ **29651**

5808 Concrete Naming Certificate for HausdorffMeasure↑ **29654**

5809 Concrete Naming Certificate for LocallyConnected↑ **29657**

5810 Concrete Naming Certificate for BishopCauchyMean↑ **29660**

5811 Concrete Naming Certificate for BishopFixedPointInterval↑ **29663**

5812 Concrete Naming Certificate for PrivateCoordinateTrigger↑ **29666**

5813 Concrete Naming Certificate for LocatedRealComparisonModulus↑ **29669**

5814 Concrete Naming Certificate for BoundaryShadowExpansionCocycle↑ **29672**

5815 Concrete Naming Certificate for HellyBraySelection↑ **29675**

5816 A Concrete Naming Certificate for DiniFiniteMinimumBudget↑ **29679**

5817 A Concrete Naming Certificate for FiniteCauchyFilter↑ **29682**

5818 A Concrete Naming Certificate for DNAWindowSixTranslationBoundary↑ **29684**

5819 A Concrete Naming Certificate for RegulatedConvergence↑ **29687**

5820 A Concrete Naming Certificate for LocatedNestedIntervalLimit↑ **29690**

5821 A Concrete Naming Certificate for CauchyTailRequest↑ **29693**

5822 A Concrete Naming Certificate for QuotientLeakageMarkov↑ **29695**

5823 A Concrete Naming Certificate for CauchyCriterionUniform↑ **29698**

5824 A Concrete Naming Certificate for CauchyProductCompletionProjection↑ **29701**

5825 A Concrete Naming Certificate for CauchyModulusSubsequence↑ **29704**

5826 A Concrete Naming Certificate for RegularCauchySequenceComparison↑ **29707**

5827 A Concrete Naming Certificate for CauchyCompletionMonadicity↑ **29712**

5828 A Concrete Naming Certificate for BorsukUlam↑ **29715**

5829 A Concrete Naming Certificate for DenjoyYoungSaks↑ **29718**

5830 A Concrete Naming Certificate for ConstructiveBolzanoWeierstrass↑ **29722**

5831 A Concrete Naming Certificate for UniformlyConvexSpace↑ **29726**

5832 A Concrete Naming Certificate for HyperbolicTranslationLength↑ **29729**

5833 A Concrete Naming Certificate for RegularCauchyCompletionEnvelope↑ **29732**

5834 A Concrete Naming Certificate for WeaklyLocatedIntervalSelection↑ **29734**

5835 A Concrete Naming Certificate for UniformInverseModulus↑ **29737**

5836 A Concrete Naming Certificate for BoundedCompleteRealInterval↑ **29740**

5837 A Concrete Naming Certificate for FreeGroup↑ **29743**

5838 A Concrete Naming Certificate for ContinuousRiemannIntegrability↑ **29746**

5839 A Concrete Naming Certificate for RegularCauchyMeshCofinality↑ **29750**

5840 A Concrete Naming Certificate for LocatedCauchyFilterBasis **29753**

5841 A Concrete Naming Certificate for OneSidedLimit↑ **29757**

5842 Concrete Naming Certificate for BertrandSeriesTest↑ **29760**

5843 Concrete Naming Certificate for RationalCanonicalForm↑ **29764**

5844 Concrete Naming Certificate for RuleOneTenPatternClassifier↑ **29767**

5845 Concrete Naming Certificate for RealizerSpace↑ **29769**

5846 Concrete Naming Certificate for RuleOneTenCausalCone↑ **29772**

5847 Concrete Naming Certificate for BaireTreeModulus↑ **29774**

5848 Located Uniform Cauchy Principle NameCert Construction **29777**

5849 Concrete Naming Certificate for BaireSpaceCompactOpen↑ **29780**

5850 Concrete Naming Certificate for CompactApartnessProfile↑ **29783**

5851 Concrete Naming Certificate for RiemannIntegralCauchy↑ **29786**

5852 Concrete Naming Certificate for InactiveAnchorCubicalSupport↑ **29789**

5853 Concrete Naming Certificate for BoundedRegulatedFunction↑ **29792**

5854 Concrete Naming Certificate for CompactnessSequentialEquivalence↑ **29795**

5855 Concrete Naming Certificate for ArensEellsSpace↑ **29799**

5856 Concrete Naming Certificate for BanachFixedPointStability↑ **29803**

5857 Concrete Naming Certificate for RealTriangleInequality↑ **29806**

5858 Concrete Naming Certificate for LowerDarbouxSum↑ **29808**

5859 Concrete Naming Certificate for BoundedMetricSpace↑ **29811**

5860 Concrete Naming Certificate for LegendreFenchelEnvelope↑ **29814**

5861 Concrete Naming Certificate for PrefixTree↑ **29816**

5862 Concrete Naming Certificate for ChomskyNormalForm↑ **29819**

5863 Concrete Naming Certificate for FiniteFanTheorem↑ **29822**

5864 Concrete Naming Certificate for SpecializationOrder↑ **29826**

5865 Concrete Naming Certificate for FiniteModulusReal↑ **29829**

5866 Concrete Naming Certificate for CompactUniformSubordinateCover↑ **29832**

5867 Concrete Naming Certificate for BishopIntervalContraction↑ **29834**

5868 A Concrete Naming Certificate for SeminormedSpace↑ **29839**

5869 A Concrete Naming Certificate for ReassignmentMassSubcube↑ **29842**

5870 A Concrete Naming Certificate for CauchyInterlacing↑ **29845**

5871 A Concrete Naming Certificate for RealLocallyCompact↑ **29848**

5872 A Concrete Naming Certificate for CompactUniformModulusStability↑ **29851**

5873 A Concrete Naming Certificate for CauchyFilterSelector↑ **29854**

5874 A Concrete Naming Certificate for RegularCauchySandwich↑ **29857**

5875 A Concrete Naming Certificate for DynamicProgramming↑ **29861**

5876 Regular Cauchy Even-Odd Substreams NameCert Construction **29864**

5877 A Concrete Naming Certificate for SchwartzSpace↑ **29868**

5878 A Concrete Naming Certificate for RuleOneTenPrivateSharedDecoration↑ **29871**

5879 A Concrete Naming Certificate for ConstructiveCantorTheorem↑ **29874**

5880 Markov-Kakutani Fixed Point NameCert Construction **29877**

5881 Window-Six Private/Shared Decoration NameCert Construction **29880**

5882 A Concrete Naming Certificate for SchurLemma↑ **29884**

5883 A Concrete Naming Certificate for CompactRealIntervalMinimum↑ **29886**

5884 A Concrete Naming Certificate for SimpleFunctionApproximation↑ **29889**

5885 A Concrete Naming Certificate for DyadicModulus↑ **29892**

5886 A Concrete Naming Certificate for CauchyFilterSelector↑ **29895**

5887 A Concrete Naming Certificate for CauchyModulusSpace↑ **29898**

5888 A Concrete Naming Certificate for CauchyCompletionLeftExactness↑ **29902**

5889 A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection↑ **29911**

5890 A Concrete Naming Certificate for SnakeLemma↑ **29914**

5891 A Concrete Naming Certificate for RealTailDiameterGauge↑ **29917**

5892 A Concrete Naming Certificate for EffectiveRealCompactIntervalSearch↑ **29920**

5893 A Concrete Naming Certificate for CauchyCompletionDenseEmbedding↑ **29926**

5894 Concrete Naming Certificate for RiemannRefinementCriterion↑ **29929**

5895 Concrete Naming Certificate for CompactUniformModulusComposition↑ **29932**

5896 Concrete Naming Certificate for PrivateSharedDecorationModel↑ **29935**

5897 Concrete Naming Certificate for BishopLocatedRealFilter↑ **29937**

5898 Concrete Naming Certificate for TaylorTheorem↑ **29940**

5899 Concrete Naming Certificate for RegularCauchyGluing↑ **29943**

5900 Concrete Naming Certificate for RegularCauchyTailMajorant↑ **29954**

5901 Concrete Naming Certificate for RealCantorIntersectionFiniteWindow↑ **29956**

5902 Concrete Naming Certificate for CauchySubsequenceSelector↑ **29958**

5903 Concrete Naming Certificate for RiemannIntegrationCompletion↑ **29964**

5904 Concrete Naming Certificate for CauchySchwarzCompletion↑ **29967**

5905 Concrete Naming Certificate for DCPO↑ **29970**

5906 Concrete Naming Certificate for CauchyDedekindGapRouter↑ **29977**

5907 Concrete Naming Certificate for SequentCalculus↑ **29980**

5908 Concrete Naming Certificate for CauchyDedekindEquivalence↑ **29983**

5909 Concrete Naming Certificate for LuenbergerObserver↑ **29986**

5910 Concrete Naming Certificate for BinaryTree↑ **29993**

5910.1 Carrier definition 29993

5910.2 Obligation surface 29994

5911 Concrete Naming Certificate for SobolevSpace↑ **29996**

5912 Concrete Naming Certificate for MetricCompletionSeparableCore↑ **29999**

5913 Concrete Naming Certificate for BaireMeasurableFunction↑ **30002**

5914 Concrete Naming Certificate for DiscoveryPipelineCalculus↑ **30005**

5915 Concrete Naming Certificate for LipschitzRetraction↑ **30008**

5916 Concrete Naming Certificate for FiniteStream↑ **30011**

5917 Concrete Naming Certificate for PoincareDiskGeodesicProjection↑ **30021**

5918 Concrete Naming Certificate for CalkinWilfTree↑ **30025**

591A Concrete Naming Certificate for TarskiFixedPoint↑ **30028**

592A Concrete Naming Certificate for LocatedRealCompletionUniversal↑ **30031**

592A Concrete Naming Certificate for NiemytzkiPlane↑ **30034**

592A Concrete Naming Certificate for RationalCauchyTelescope↑ **30037**

592A Concrete Naming Certificate for BishopDiagonalCauchyTelescope↑ **30040**

592A Concrete Naming Certificate for HausdorffReflectionCompletionCounit↑ **30044**

592A Concrete Naming Certificate for MetaCICCandidateStrongNormalization↑ **30047**

592A Concrete Naming Certificate for RealDiagonalModulus↑ **30051**

592A Concrete Naming Certificate for CompactUniformHomeomorphism↑ **30054**

592A Concrete Naming Certificate for LocatedMonotoneLimit↑ **30057**

592A Concrete Naming Certificate for HopfRinowFiniteGeodesic↑ **30060**

593A Concrete Naming Certificate for EffectiveCauchyCompletion↑ **30063**

593A Concrete Naming Certificate for EdelsteinContraction↑ **30067**

593A Concrete Naming Certificate for BerkovichClosedDisc↑ **30070**

593Bounded Real Variation NameCert Construction **30073**

593A Concrete Naming Certificate for ProfiniteCompletion↑ **30076**

593A Concrete Naming Certificate for SequentialCompactReal↑ **30079**

593A Concrete Naming Certificate for MetaCICNormalizationFuelBoundary↑ **30082**

593A Concrete Naming Certificate for CompactIntervalModulus↑ **30085**

593A Concrete Naming Certificate for MoscoConvergence↑ **30088**

593A Concrete Naming Certificate for FiniteRealIntervalChain↑ **30092**

594A Concrete Naming Certificate for RegularRealTailEquivalence↑ **30094**

594A Concrete Naming Certificate for BrouwerFixedPointMetric↑ **30100**

594A Concrete Naming Certificate for MetaCICGroundCompilerNormalizationRoute↑ **30103**

594A Concrete Naming Certificate for CauchyModulusCompletionSelector↑ **30106**

594A Concrete Naming Certificate for KuratowskiHyperspace↑ **30109**

5945 A Concrete Naming Certificate for LimitedPrincipleOmniscience↑ **30111**

5946 A Concrete Naming Certificate for HenselLemma↑ **30114**

5947 A Concrete Naming Certificate for LocatedRiemannIntegral↑ **30117**

5948 A Concrete Naming Certificate for KolmogorovQuotient↑ **30120**

5949 A Concrete Naming Certificate for BoundedTypecheckingBudget↑ **30123**

5950 A Concrete Naming Certificate for ScottTopology↑ **30126**

5951 A Concrete Naming Certificate for FiniteRealSupremum↑ **30128**

5952 A Concrete Naming Certificate for HahnDecomposition↑ **30131**

5953 A Concrete Naming Certificate for ConstructivePolishSpace↑ **30134**

5954 A Concrete Naming Certificate for CompactUniformEquicontinuityStability↑ **30137**

5955 A Concrete Naming Certificate for RealEqualityReflection↑ **30140**

5956 A Concrete Naming Certificate for ConstructiveSobolevEnergy↑ **30143**

5957 A Concrete Naming Certificate for GammaConvergence↑ **30145**

5958 A Concrete Naming Certificate for LocatedRealChoiceSequence↑ **30148**

5959 A Concrete Naming Certificate for NestedIntervalCauchySeal↑ **30151**

5960 A Concrete Naming Certificate for Spectrahedron↑ **30153**

5961 A Concrete Naming Certificate for NestedRadicalConvergence↑ **30157**

5962 A Concrete Naming Certificate for BoundedNormalizationFuel↑ **30160**

5963 A Concrete Naming Certificate for RegularCauchyLocatedField↑ **30163**

5964 A Concrete Naming Certificate for CompactLipschitzUniformHandoff↑ **30166**

5965 A Concrete Naming Certificate for RegularSequenceSpace↑ **30169**

5966 Dyadic Interval Net NameCert Construction **30187**

5967 Channel Termination Budget Certificate NameCert Construction **30209**

5968 A Concrete Naming Certificate for CadlagRealFunction↑ **30212**

5969 A Concrete Naming Certificate for AlmostPeriodic↑ **30215**

5970 A Concrete Naming Certificate for MetaCICResidualJoinDeterminacy↑ **30218**

597A Concrete Naming Certificate for BrownianBridge↑ **30221**

597B Concrete Naming Certificate for CauchyModulusFunctor↑ **30224**

597C Concrete Naming Certificate for FiniteModulusCompactUniformEquicontinuity↑ **30226**

597D Concrete Naming Certificate for RealCompletionBoundaryWitness↑ **30229**

597E Concrete Naming Certificate for RegularCauchyMeshGluing↑ **30232**

597F Concrete Naming Certificate for BrouwerSpreadBar↑ **30235**

597G Concrete Naming Certificate for BishopLocatedCauchyCompletionStability↑ **30238**

597H CauchyMatrixUp NameCert Construction **30241**

597J Concrete Naming Certificate for GrothendieckGroup↑ **30244**

598K Concrete Naming Certificate for WeakOmniscienceBoundary↑ **30247**

598L Concrete Naming Certificate for MetricCauchyFilterBridge↑ **30250**

598M Concrete Naming Certificate for CauchyCompletionReflectorTriangleHandoff↑ **30253**

598N Concrete Naming Certificate for CompleteMetricLimitUniqueness↑ **30256**

598P Concrete Naming Certificate for ArchimedeanRealDensity↑ **30274**

598Q Concrete Naming Certificate for CompleteMetricNestedBalls↑ **30277**

598R Concrete Naming Certificate for CompactUniformCoverChain↑ **30280**

598S Concrete Naming Certificate for MonotoneCauchyConvergence↑ **30283**

598T Concrete Naming Certificate for UniformCompletionKleisli↑ **30286**

598V Concrete Naming Certificate for KolmogorovComplexity↑ **30289**

599W Concrete Naming Certificate for RegularCauchyPower↑ **30292**

599X Concrete Naming Certificate for StreamNameExtensionality↑ **30295**

599Y Concrete Naming Certificate for CaratheodoryMeasureExtension↑ **30300**

599Z Concrete Naming Certificate for SubspaceTopology↑ **30303**

599a Concrete Naming Certificate for NormalizationCandidateFrontier↑ **30308**

599b Concrete Naming Certificate for EffrosBorelSpace↑ **30314**

599c Concrete Naming Certificate for CountingMeasure↑ **30316**

599.7 Concrete Naming Certificate for RationalOpenIntervalBasis↑ **30319**

599.8 Concrete Naming Certificate for FreudenthalEnd↑ **30322**

599.9 Concrete Naming Certificate for FiniteWindowBaireCategory↑ **30324**

600.0 Concrete Naming Certificate for FiniteSubcoverRadiusLedger↑ **30327**

600.1 Concrete Naming Certificate for BishopLocatedInverseFunction↑ **30329**

600.2 Concrete Naming Certificate for RegularCauchyDiagonalLimit↑ **30332**

600.3 Finite Cauchy Density Net NameCert Construction **30335**

600.4 Concrete Naming Certificate for RuleOneTenLocalStencilObservation↑ **30347**

600.5 Concrete Naming Certificate for LocatedUniformLimit↑ **30350**

600.6 Concrete Naming Certificate for FiniteIntervalLattice↑ **30353**

600.7 Concrete Naming Certificate for KernelAcceptanceBoundary↑ **30358**

600.8 Concrete Naming Certificate for LocatedCauchyIntervalRefinement↑ **30363**

600.9 Concrete Naming Certificate for KleeneBrouwerOrder↑ **30366**

601.0 Concrete Naming Certificate for DedekindCutArithmetic↑ **30368**

601.1 Concrete Naming Certificate for HausdorffUniformityCompletion↑ **30371**

601.2 Concrete Naming Certificate for CompleteOrderedField↑ **30375**

601.3 Concrete Naming Certificate for ChebyshevEquioscillation↑ **30377**

601.4 Concrete Naming Certificate for CauchyPasting↑ **30381**

601.5 Concrete Naming Certificate for CofinalTailNormalForm↑ **30385**

601.6 Concrete Naming Certificate for LebesgueNumberFiniteCover↑ **30388**

601.7 Concrete Naming Certificate for PhaseDistanceObserverExpansion↑ **30391**

601.8 Concrete Naming Certificate for PushdownAutomaton↑ **30394**

601.9 Concrete Naming Certificate for BaireCategoryCompleteMetric↑ **30397**

602.0 Concrete Naming Certificate for ContractiveCauchyIterator↑ **30400**

602.1 Concrete Naming Certificate for PoincareVisualRadiusLadder↑ **30403**

602.2 Concrete Naming Certificate for CompactUniformTriangleLedger↑ **30406**

6023 Concrete Naming Certificate for CantorIntersectionUniformWindow↑ **30409**

6024 Concrete Naming Certificate for VariationalEquation↑ **30412**

6025 Concrete Naming Certificate for HyperbolicHorofunctionBoundary↑ **30415**

6026 Concrete Naming Certificate for RegularCauchyOscillation↑ **30418**

6027 Concrete Naming Certificate for FiniteWindowConfluence↑ **30421**

6028 Concrete Naming Certificate for BarInductionContinuity↑ **30427**

6029 Concrete Naming Certificate for UniformLimitCauchyCriterion↑ **30429**

6030 Concrete Naming Certificate for BrouwerFiniteChoiceSequenceModulus↑ **30432**

6031 Concrete Naming Certificate for MinkowskiFunctional↑ **30435**

6032 Concrete Naming Certificate for GroundCompilerStrengthMonotonicity↑ **30438**

6033 Concrete Naming Certificate for YoungMeasure↑ **30441**

6034 Concrete Naming Certificate for FiniteSupportStream↑ **30444**

6035 Concrete Naming Certificate for EffectiveCoverCompactness↑ **30447**

6036 Concrete Naming Certificate for FastCauchyWitness↑ **30450**

6037 Concrete Naming Certificate for SingularValueDecomposition↑ **30453**

6038 Concrete Naming Certificate for Rule110GliderPhaseClassifier↑ **30456**

6039 Concrete Naming Certificate for SeparatedCompletionMonad↑ **30459**

6040 Concrete Naming Certificate for MorseBoundary↑ **30462**

6041 Concrete Naming Certificate for RealCompletionUniversalProperty↑ **30465**

6042 Concrete Naming Certificate for RealModulusArithmetic↑ **30472**

6043 Concrete Naming Certificate for FiniteDimensionalCompactness↑ **30475**

6044 Concrete Naming Certificate for ComplexPhaseSurfaceFiber↑ **30478**

6045 Concrete Naming Certificate for LocatedRealCauchyCompleteness↑ **30481**

6046 Concrete Naming Certificate for LocalizedCauchySubsequence↑ **30484**

6047 Concrete Naming Certificate for LebesgueStieltjesMeasure↑ **30487**

6048 Concrete Naming Certificate for CompactModulusRealizer↑ **30490**

6049 A Concrete Naming Certificate for JacksonFiniteApproximation↑ **30493**

6050 A Concrete Naming Certificate for DyadicCauchyOscillationModulus↑ **30496**

6051 A Concrete Naming Certificate for FractionalSobolev↑ **30499**

6052 A Concrete Naming Certificate for CauchyCondensationTailLock↑ **30502**

6053 A Concrete Naming Certificate for BanachContractionEndpoint↑ **30505**

6054 A Concrete Naming Certificate for FiniteIntervalChoice↑ **30508**

6055 A Concrete Naming Certificate for MinimalRealization↑ **30511**

6056 A Concrete Naming Certificate for BishopSpeckerCompletionBoundary↑ **30514**

6057 A Concrete Naming Certificate for BishopRealCompletionAdjoint↑ **30518**

6058 A Concrete Naming Certificate for AbelTheorem↑ **30521**

6059 A Concrete Naming Certificate for LawlikeCauchyReal↑ **30524**

6060 A Concrete Naming Certificate for BoundedInverseTheorem↑ **30527**

6061 A Concrete Naming Certificate for BishopCompletionKleisliTriangle↑ **30530**

6062 A Concrete Naming Certificate for RiemannUniformMesh↑ **30532**

6063 A Concrete Naming Certificate for FiniteWindowMetaCICDiamondEnvelope↑ **30534**

6064 A Concrete Naming Certificate for UniformCauchyModulusExtractor↑ **30538**

6065 A Concrete Naming Certificate for BoundedCauchyInterval↑ **30541**

6066 A Concrete Naming Certificate for AbsoluteContinuity↑ **30544**

6067 A Concrete Naming Certificate for MonotoneBarModulus↑ **30548**

6068 A Concrete Naming Certificate for MetricCompletionAdjoint↑ **30551**

6069 A Concrete Naming Certificate for MetaCICTypedConversionResidualFrontier↑ **30554**

6070 A Concrete Naming Certificate for HyperbolicPattersonSullivanShadow↑ **30557**

6071 A Concrete Naming Certificate for CauchyCondensationIntegralBridge↑ **30560**

6072 A Concrete Naming Certificate for OptionalSampling↑ **30564**

6073 A Concrete Naming Certificate for HyperbolicGromovProduct↑ **30567**

6074 A Concrete Naming Certificate for UniformCauchyWindowExhaustion↑ **30570**

6075 A Concrete Naming Certificate for PointedGromovHausdorff↑ **30573**

6076 A Concrete Naming Certificate for CauchySequenceComplete↑ **30576**

6077 A Concrete Naming Certificate for RiemannSumCauchyNet↑ **30579**

6078 A Concrete Naming Certificate for BoundedCauchySequence↑ **30582**

6079 A Concrete Naming Certificate for CompactOscillationBarModulus↑ **30585**

6080 A Concrete Naming Certificate for EnrichedYoneda↑ **30588**

6081 A Concrete Naming Certificate for LocatedUpperDedekindReal↑ **30590**

6082 A Concrete Naming Certificate for LocatedLowerDedekindReal↑ **30592**

6083 A Concrete Naming Certificate for DilworthDecomposition↑ **30595**

6084 A Concrete Naming Certificate for PointwiseStreamName↑ **30598**

6085 A Concrete Naming Certificate for MetacicCandidateSnHandoff↑ **30602**

6086 A Concrete Naming Certificate for CauchySeparatedReflection↑ **30605**

6087 A Concrete Naming Certificate for FiniteApproximateFixedPoint↑ **30607**

6088 A Concrete Naming Certificate for RealCompletenessGapRouter↑ **30610**

6089 A Concrete Naming Certificate for BishopCompletionSpace↑ **30613**

6090 A Concrete Naming Certificate for HyperbolicLinearizedPhaseDistance↑ **30616**

6091 A Concrete Naming Certificate for NestedIntervalCompleteness↑ **30619**

6092 A Concrete Naming Certificate for GaugeLebesgueNumber↑ **30622**

6093 A Concrete Naming Certificate for EilenbergSteenrodAxioms↑ **30625**

6094 A Concrete Naming Certificate for EquicontinuityModulusSelection↑ **30628**

6095 A Concrete Naming Certificate for EquicontinuityRadiusChoice↑ **30631**

6096 A Concrete Naming Certificate for CompleteMetricCriterion↑ **30634**

6097 A Concrete Naming Certificate for ClassifierFormingLearning↑ **30637**

6098 A Concrete Naming Certificate for LocatedSequenceSupremum↑ **30640**

6099 A Concrete Naming Certificate for ScottContinuous↑ **30643**

6100 A Concrete Naming Certificate for BlackBoxDecompositionCertificate↑ **30646**

610 A Concrete Naming Certificate for BishopLocatedModulus↑ **30649**

610 A Concrete Naming Certificate for FiniteApartnessNet↑ **30652**

610 A Concrete Naming Certificate for MetaCICSubstitutionSpine↑ **30655**

610 A Concrete Naming Certificate for HyperbolicGeodesicFlowLedger↑ **30658**

610 A Concrete Naming Certificate for FiniteMetricCover↑ **30661**

610 A Concrete Naming Certificate for HahnMazurkiewiczBoundary↑ **30664**

610 A Concrete Naming Certificate for ConstantRegSeqRat↑ **30667**

610 A Concrete Naming Certificate for CompactnessCompletionRoute↑ **30670**

610 A Concrete Naming Certificate for NewtonPolygon↑ **30673**

611 A Concrete Naming Certificate for DiniDerivativeWindow↑ **30676**

611 A Concrete Naming Certificate for MetaCICDischargeInversionBoundary↑ **30679**

611 A Concrete Naming Certificate for BishopDiagonalCauchySequence↑ **30682**

611 A Concrete Naming Certificate for CompactUniformRadiusMinimumStability↑ **30685**

611 A Concrete Naming Certificate for FiniteSelectionSequence↑ **30688**

611 A Concrete Naming Certificate for CantorSpaceUniformModulus↑ **30691**

611 A Concrete Naming Certificate for RegularCauchySubsequenceModulus↑ **30694**

611 A Concrete Naming Certificate for CertifiedModelQuality↑ **30697**

611 A Concrete Naming Certificate for RegularCauchyCompletionCounit↑ **30700**

611 A Concrete Naming Certificate for ConstantCauchyName↑ **30703**

612 A Concrete Naming Certificate for KolmogorovExtensionFiniteWindow↑ **30706**

612 A Concrete Naming Certificate for IntermediateValueRealSeal↑ **30709**

612 A Concrete Naming Certificate for HyperbolicAxisShadow↑ **30712**

612 A Concrete Naming Certificate for AbelUniform↑ **30715**

612 A Concrete Naming Certificate for WorldModelClassifier↑ **30718**

612 A Concrete Naming Certificate for FiniteSubcoverLedger↑ **30721**

612 A Concrete Naming Certificate for RealSine↑ **30723**

6127 Concrete Naming Certificate for FiniteMeasure↑ **30727**

6128 Concrete Naming Certificate for LegendrePolynomial↑ **30730**

6129 Concrete Naming Certificate for CauchyFilterSpace↑ **30733**

6130 Concrete Naming Certificate for CauchyCompletionStrength↑ **30736**

6131 Concrete Naming Certificate for DoublingMetricSpace↑ **30739**

6132 Concrete Naming Certificate for RealCauchyFilterLimit↑ **30742**

6133 Concrete Naming Certificate for RealCompletionLimitRouter↑ **30744**

6134 Concrete Naming Certificate for CauchyBornologyBasis↑ **30747**

6135 Concrete Naming Certificate for CompactDiniModulus↑ **30751**

6136 Concrete Naming Certificate for BairePointwiseOscillation↑ **30754**

6137 Concrete Naming Certificate for StolzCesaroTheorem↑ **30757**

6138 Concrete Naming Certificate for ClassifierNoveltyWitness↑ **30759**

6139 Concrete Naming Certificate for RegularCauchyApartnessReflector↑ **30761**

6140 Concrete Naming Certificate for MonotoneSequenceWindowCauchy↑ **30764**

6141 Concrete Naming Certificate for RegularCauchyComparisonNet↑ **30766**

6142 Concrete Naming Certificate for BoundaryFarEndNoncenter↑ **30769**

6143 Concrete Naming Certificate for CompactUniformRadiusTransport↑ **30772**

6144 Concrete Naming Certificate for NemytskiiOperator↑ **30775**

6145 Concrete Naming Certificate for FiniteLipschitzIteration↑ **30779**

6146 Concrete Naming Certificate for ObserverPhaseGroupoid↑ **30782**

6147 Concrete Naming Certificate for PompeiuHausdorffDistanceRealizer↑ **30785**

6148 Concrete Naming Certificate for CompactIntervalOscillationModulus↑ **30788**

6149 Concrete Naming Certificate for ConstructiveRealTrichotomyBoundary↑ **30791**

6150 Concrete Naming Certificate for SeparableUltrametricSpace↑ **30794**

6151 Concrete Naming Certificate for FanTheoremBarModulus↑ **30797**

6152 Concrete Naming Certificate for FanUniformContinuity↑ **30800**

6153 Concrete Naming Certificate for DyadicToleranceComposition↑ **30803**

6154 Concrete Naming Certificate for KuratowskiClosureAxiom↑ **30806**

6155 Concrete Naming Certificate for ClusterFilter↑ **30809**

6156 Concrete Naming Certificate for RiemannIntegralCompletionModulus↑ **30812**

6157 Concrete Naming Certificate for FellTopology↑ **30815**

6158 Concrete Naming Certificate for HeatKernel↑ **30818**

6159 Concrete Naming Certificate for OsgoodCriterion↑ **30821**

6160 Concrete Naming Certificate for CauchyReflectiveEmbedding↑ **30824**

6161 Concrete Naming Certificate for CompactFilterDiniMonotone↑ **30827**

6162 Concrete Naming Certificate for ClosedConsistencyReductionObstruction↑ **30830**

6163 Concrete Naming Certificate for RealIntervalUniformContinuity↑ **30833**

6164 Concrete Naming Certificate for ArithmeticalHierarchy↑ **30836**

6165 Concrete Naming Certificate for HilbertCurve↑ **30839**

6166 Concrete Naming Certificate for CauchyProductBoundary↑ **30842**

6167 Concrete Naming Certificate for UniformLipschitzFamily↑ **30845**

6168 Concrete Naming Certificate for RealIntervalTotalBoundedness↑ **30848**

6169 Concrete Naming Certificate for ClosedParallelDiamondObstruction↑ **30851**

6170 Concrete Naming Certificate for UniformSpaceRealCompletion↑ **30854**

6171 Concrete Naming Certificate for CechCompleteSpace↑ **30857**

6172 Concrete Naming Certificate for LocatedCompactCompletion↑ **30860**

6173 Concrete Naming Certificate for UniformLipschitzCompactVision↑ **30863**

6174 Concrete Naming Certificate for FiniteLipschitzFamily↑ **30867**

6175 Concrete Naming Certificate for TypedGalleryContextProbe↑ **30870**

6176 Concrete Naming Certificate for GaleShapleyStableMatching↑ **30873**

6177 Concrete Naming Certificate for ClosedAppArgumentStability↑ **30876**

6178 Concrete Naming Certificate for ContextSensitiveGrammar↑ **30879**

6179 A Concrete Naming Certificate for BoundedNormalizationFrontier↑ **30882**

6180 A Concrete Naming Certificate for BishopLocatedReal↑ **30885**

6181 A Concrete Naming Certificate for ExplanationComplexityInformation↑ **30888**

6182 A Concrete Naming Certificate for WijsmanTopology↑ **30891**

6183 A Concrete Naming Certificate for RegularCauchySubnetSelector↑ **30894**

6184 A Concrete Naming Certificate for FiniteStateTransducer↑ **30897**

6185 A Concrete Naming Certificate for FarkasLemma↑ **30901**

6186 A Concrete Naming Certificate for BishopIntervalApartnessModulus↑ **30904**

6187 A Concrete Naming Certificate for BairePrefixCompletion↑ **30908**

6188 A Concrete Naming Certificate for MinimalDfa↑ **30911**

6189 A Concrete Naming Certificate for BishopLocatedInterval↑ **30914**

6190 A Concrete Naming Certificate for KleeneKreiselContinuousFunctional↑ **30917**

6191 A Concrete Naming Certificate for PicardLindelofLocalIteration↑ **30920**

6192 A Concrete Naming Certificate for LocatedDyadicReal↑ **30923**

6193 A Concrete Naming Certificate for MomentProblem↑ **30926**

6194 A Concrete Naming Certificate for KleeneTreeBoundary↑ **30929**

6195 A Concrete Naming Certificate for TotalBoundednessModulus↑ **30932**

6196 A Concrete Naming Certificate for HyperbolicSet↑ **30936**

6197 A Concrete Naming Certificate for LocatedDecimalStreamNormal↑ **30939**

6198 A Concrete Naming Certificate for ChurchRosser↑ **30942**

6199 A Concrete Naming Certificate for LocatedIntervalRefinement↑ **30945**

6200 A Concrete Naming Certificate for DifferentialPrivacy↑ **30948**

6201 A Concrete Naming Certificate for UniformCauchySubnet↑ **30951**

6202 A Concrete Naming Certificate for PositiveInterpretabilityClassifier↑ **30954**

6203 A Concrete Naming Certificate for RegularCauchyFilterReal↑ **30957**

6204 A Concrete Naming Certificate for DyadicTotallyBoundedInterval↑ **30960**

6205 A Concrete Naming Certificate for ArithmeticoGeometricSequence↑ **30962**

6206 A Concrete Naming Certificate for InverseFunctionTheorem↑ **30964**

6207 A Concrete Naming Certificate for CompactIntervalUniformModulus↑ **30967**

6208 A Concrete Naming Certificate for RegularCauchyTailGauge↑ **30970**

6209 A Concrete Naming Certificate for PoincareDiskMetric↑ **30973**

6210 A Concrete Naming Certificate for UniformPolygonalApproximation↑ **30976**

6211 A Concrete Naming Certificate for ItoIntegral↑ **30980**

6212 A Concrete Naming Certificate for ItoIsometry↑ **30982**

6213 A Concrete Naming Certificate for MaximumModulusPrinciple↑ **30985**

6214 A Concrete Naming Certificate for UniversalCauchyCompletion↑ **30988**

6215 A Concrete Naming Certificate for MintyBrowderMonotoneOperator↑ **30991**

6216 A Concrete Naming Certificate for CompactUniformDiameterDecay↑ **30994**

6217 A Concrete Naming Certificate for BoundedMonotoneRealModulus↑ **30997**

6218 A Concrete Naming Certificate for MooreSmithL10CompletionNet↑ **31000**

6219 A Concrete Naming Certificate for CauchyModulusEnvelope↑ **31004**

6220 A Concrete Naming Certificate for RealArctangent↑ **31007**

6221 A Concrete Naming Certificate for CompletionDenseRange↑ **31010**

6222 A Concrete Naming Certificate for HadamardFinitePart↑ **31013**

6223 A Concrete Naming Certificate for RegulatedMeshModulus↑ **31016**

6224 A Concrete Naming Certificate for Epigraph↑ **31019**

6225 A Concrete Naming Certificate for PositiveInterpretabilityClassifierPractice↑ **31024**

6226 A Concrete Naming Certificate for CauchyModulusCompleteness↑ **31027**

6227 A Concrete Naming Certificate for SpeckerMonotoneBoundary↑ **31030**

6228 A Concrete Naming Certificate for UniformIntegrability↑ **31033**

6229 A Concrete Naming Certificate for RegularCauchyTailBall↑ **31036**

6230 A Concrete Naming Certificate for LocatedIntervalSupremum↑ **31039**

6231 Concrete Naming Certificate for EulerMethodFiniteStep$^{\uparrow}$ **31043**

6232 Concrete Naming Certificate for RegularCauchySaturation$^{\uparrow}$ **31046**

6233 Concrete Naming Certificate for MarkovReal$^{\uparrow}$ **31048**

6234 Concrete Naming Certificate for CompactEquicontinuousFamilyModulus$^{\uparrow}$ **31051**

6235 Concrete Naming Certificate for CWComplex$^{\uparrow}$ **31054**

6236 Concrete Naming Certificate for HermitePolynomial$^{\uparrow}$ **31057**

6237 Concrete Naming Certificate for HyperbolicMobiusInvariant$^{\uparrow}$ **31060**

6238 Concrete Naming Certificate for CauchyMeshStability$^{\uparrow}$ **31062**

6239 Concrete Naming Certificate for CauchyCompletionPushout$^{\uparrow}$ **31065**

6240 Concrete Naming Certificate for LocatedLowerBound$^{\uparrow}$ **31068**

6241 Concrete Naming Certificate for ClassifierNoveltyLedger$^{\uparrow}$ **31071**

6242 Concrete Naming Certificate for OsgoodModulus$^{\uparrow}$ **31074**

6243 Concrete Naming Certificate for TheoremDnaScore$^{\uparrow}$ **31078**

6244 Concrete Naming Certificate for NoetherNormalization$^{\uparrow}$ **31081**

6245 Concrete Naming Certificate for CoxeterGroup$^{\uparrow}$ **31084**

6246 Concrete Naming Certificate for LipschitzCompletionExtension$^{\uparrow}$ **31087**

6247 Concrete Naming Certificate for OrdinalAnalysis$^{\uparrow}$ **31090**

6248 Concrete Naming Certificate for CartanMatrix$^{\uparrow}$ **31093**

6249 Concrete Naming Certificate for ScopedExplanationLedger$^{\uparrow}$ **31096**

6250 Concrete Naming Certificate for CompactCoverModulus$^{\uparrow}$ **31099**

6251 Concrete Naming Certificate for RealReciprocal$^{\uparrow}$ **31102**

6252 Concrete Naming Certificate for FundamentalTheoremCalculus$^{\uparrow}$ **31105**

6253 Concrete Naming Certificate for LocatedIntervalCompletion$^{\uparrow}$ **31108**

6254 Concrete Naming Certificate for FiniteVariationLedger$^{\uparrow}$ **31111**

6255 Concrete Naming Certificate for MetricCompletionMorphism$^{\uparrow}$ **31114**

6256 Concrete Naming Certificate for LocatedIntervalUniformCauchy$^{\uparrow}$ **31117**

6257 A Concrete Naming Certificate for CompactRealIntervalSearchBasis$^{\uparrow}$ **31121**

6258 A Concrete Naming Certificate for CountablyBasedUniformity$^{\uparrow}$ **31125**

6259 A Concrete Naming Certificate for GaloisConnection$^{\uparrow}$ **31128**

6260 A Concrete Naming Certificate for StoneWeierstrassLattice$^{\uparrow}$ **31131**

6261 A Concrete Naming Certificate for PringsheimConvergence$^{\uparrow}$ **31134**

6262 A Concrete Naming Certificate for OscillationRankFunction$^{\uparrow}$ **31138**

6263 A Concrete Naming Certificate for CompactCoverNerve$^{\uparrow}$ **31141**

6264 A Concrete Naming Certificate for CompactUniformCauchy$^{\uparrow}$ **31144**

6265 A Concrete Naming Certificate for CompleteRealModulus$^{\uparrow}$ **31147**

6266 A Concrete Naming Certificate for RegularCauchyTailFilterBasis$^{\uparrow}$ **31150**

6267 A Concrete Naming Certificate for PoincareDiskBoundaryTransport$^{\uparrow}$ **31153**

6268 A Concrete Naming Certificate for FiniteCauchyNetExtraction$^{\uparrow}$ **31156**

6269 A Concrete Naming Certificate for PoincareRecurrence$^{\uparrow}$ **31159**

6270 A Concrete Naming Certificate for GronwallBellman$^{\uparrow}$ **31162**

6271 A Concrete Naming Certificate for RealUniformCauchyProduct$^{\uparrow}$ **31165**

6272 A Concrete Naming Certificate for PeanoKernel$^{\uparrow}$ **31169**

6273 A Concrete Naming Certificate for BanachFixedPointIterationBudget$^{\uparrow}$ **31172**

6274 A Concrete Naming Certificate for CompactCauchyEquicontinuity$^{\uparrow}$ **31174**

6275 A Concrete Naming Certificate for CompactUniformCauchyModulus$^{\uparrow}$ **31176**

6276 A Concrete Naming Certificate for MengerCoverCompactness$^{\uparrow}$ **31179**

6277 A Concrete Naming Certificate for FiniteVolume$^{\uparrow}$ **31182**

6278 A Concrete Naming Certificate for OrderedCauchyGroup$^{\uparrow}$ **31184**

6279 A Concrete Naming Certificate for RealCompletionWitnessLedger$^{\uparrow}$ **31187**

6280 A Concrete Naming Certificate for RegularCauchyRealReflection$^{\uparrow}$ **31189**

6281 A Concrete Naming Certificate for InitialUniformity$^{\uparrow}$ **31192**

6282 A Concrete Naming Certificate for UniformCompletionReflector$^{\uparrow}$ **31195**

6283 Concrete Naming Certificate for RealCompletionAdjoint↑ **31198**

6284 Concrete Naming Certificate for UnifiedEpistemicObject↑ **31201**

6285 Concrete Naming Certificate for AlexanderSubbaseTheorem↑ **31204**

6286 Concrete Naming Certificate for BishopUniformCauchyCompletion↑ **31206**

6287 Concrete Naming Certificate for PointedCompleteMetric↑ **31209**

6288 Concrete Naming Certificate for CauchyContinuousFunction↑ **31212**

6289 Concrete Naming Certificate for RegularCauchyDensity↑ **31215**

6290 Concrete Naming Certificate for ChoquetCompleteSpace↑ **31218**

6291 Concrete Naming Certificate for SubstrateFlowInvariant↑ **31221**

6292 Concrete Naming Certificate for BishopEquicontinuityPrinciple↑ **31224**

6293 Concrete Naming Certificate for CompactUniformOscillationWindow↑ **31227**

6294 Concrete Naming Certificate for FiniteCoverModulusSelector↑ **31230**

6295 Concrete Naming Certificate for PrimitiveCertifiedRoute↑ **31233**

6296 Concrete Naming Certificate for FiniteDyadicRefinementBudget↑ **31236**

6297 Concrete Naming Certificate for RealCauchyFilterBasis↑ **31240**

6298 Concrete Naming Certificate for BishopCauchyCompletionPair↑ **31243**

6299 Concrete Naming Certificate for JordanDecompositionBoundedVariation↑ **31246**

6300 Concrete Naming Certificate for FiniteOscillationRadiusLedger↑ **31250**

6301 Concrete Naming Certificate for BishopRegularRealSeal↑ **31253**

6302 Concrete Naming Certificate for SyntacticMonoid↑ **31256**

6303 Concrete Naming Certificate for CertifiedExplanationSchematic↑ **31259**

6304 Concrete Naming Certificate for LocatedUpperSemicontinuousEnvelope↑ **31262**

6305 Concrete Naming Certificate for NormalizationByEvaluation↑ **31265**

6306 Concrete Naming Certificate for NeutralReifier↑ **31268**

6307 Concrete Naming Certificate for CayleyHamilton↑ **31271**

6308 Concrete Naming Certificate for StandardBorelSpace↑ **31273**

6309 A Concrete Naming Certificate for FiniteFourierEnergy$^{\uparrow}$ **31276**

6310 A Concrete Naming Certificate for ProximitySpace$^{\uparrow}$ **31279**

6311 A Concrete Naming Certificate for CauchyTailSupremum$^{\uparrow}$ **31282**

6312 A Concrete Naming Certificate for SingularHomology$^{\uparrow}$ **31285**

6313 A Concrete Naming Certificate for KaramataInequality$^{\uparrow}$ **31288**

6314 A Concrete Naming Certificate for FiniteWindowNormalizationAudit$^{\uparrow}$ **31291**

6315 A Concrete Naming Certificate for BishopRegularSequence$^{\uparrow}$ **31294**

6316 A Concrete Naming Certificate for SystemF$^{\uparrow}$ **31297**

6317 A Concrete Naming Certificate for CompilerClosureSelfCompilation$^{\uparrow}$ **31300**

6318 A Concrete Naming Certificate for ChoiceSequenceReal$^{\uparrow}$ **31303**

6319 A Concrete Naming Certificate for HyperbolicObserverPhaseShell$^{\uparrow}$ **31306**

6320 A Concrete Naming Certificate for RegularCauchyDiagonalEmbedding$^{\uparrow}$ **31309**

6321 A Concrete Naming Certificate for MetaCICResidualContextCut$^{\uparrow}$ **31312**

6322 A Concrete Naming Certificate for OpialCondition$^{\uparrow}$ **31315**

6323 A Concrete Naming Certificate for FiniteIntersectionCompactness$^{\uparrow}$ **31318**

6324 A Concrete Naming Certificate for MichaelLine$^{\uparrow}$ **31320**

6325 A Concrete Naming Certificate for AssouadDimension$^{\uparrow}$ **31323**

6326 A Concrete Naming Certificate for HeineBorelFiniteSubcover$^{\uparrow}$ **31326**

6327 A Concrete Naming Certificate for RealModulusComparison$^{\uparrow}$ **31329**

6328 A Concrete Naming Certificate for WhiteheadProduct$^{\uparrow}$ **31331**

6329 A Concrete Naming Certificate for CompactMetricLocatedNetSelection$^{\uparrow}$ **31334**

6330 A Concrete Naming Certificate for TriebelLizorkinSpace$^{\uparrow}$ **31336**

6331 A Concrete Naming Certificate for DyadicRegularBridge$^{\uparrow}$ **31339**

6332 A Concrete Naming Certificate for BHistHendecaEventFlowNameCert$^{\uparrow}$ **31341**

6333 A Concrete Naming Certificate for RegularCauchyLocatedSplit$^{\uparrow}$ **31343**

6334 A Concrete Naming Certificate for RegularRealCauchySelector$^{\uparrow}$ **31346**

6335 A Concrete Naming Certificate for LocalInterHistRateComparison$^{\uparrow}$ **31349**

6336 A Concrete Naming Certificate for CATZeroMetric$^{\uparrow}$ **31352**

6337 Rising Sun Lemma NameCert Construction **31355**

6338 A Concrete Naming Certificate for RegularCauchyNet$^{\uparrow}$ **31357**

6339 A Concrete Naming Certificate for BesicovitchCovering$^{\uparrow}$ **31360**

6340 A Concrete Naming Certificate for ReducedProduct$^{\uparrow}$ **31363**

6341 A Concrete Naming Certificate for AronszajnTree$^{\uparrow}$ **31366**

6342 A Concrete Naming Certificate for ExtFunctor$^{\uparrow}$ **31369**

6343 A Concrete Naming Certificate for UniformCauchyCompletionRealizer$^{\uparrow}$ **31372**

6344 A Concrete Naming Certificate for GateauxDerivative$^{\uparrow}$ **31375**

6344.1 Directional carrier 31375

6345 A Concrete Naming Certificate for LebesgueNumberFiniteNet$^{\uparrow}$ **31378**

6346 A Concrete Naming Certificate for DunfordPettis$^{\uparrow}$ **31381**

6347 A Concrete Naming Certificate for LocatedOrderInterval$^{\uparrow}$ **31384**

6348 A Concrete Naming Certificate for NagataSmirnovMetrization$^{\uparrow}$ **31387**

6349 A Concrete Naming Certificate for PicardCompletionResidualCauchyRealizer$^{\uparrow}$ **31390**

6350 A Concrete Naming Certificate for CoreCompact$^{\uparrow}$ **31393**

6351 A Concrete Naming Certificate for BoundedRiemannConvergence$^{\uparrow}$ **31396**

6352 A Concrete Naming Certificate for ConstructiveDiniTheorem$^{\uparrow}$ **31399**

6353 A Concrete Naming Certificate for BanachOperatorGraphNorm$^{\uparrow}$ **31404**

6354 A Concrete Naming Certificate for ImplicitFunctionTheorem$^{\uparrow}$ **31407**

6355 Picard Iteration Budget NameCert Construction **31410**

6356 Kernel Budget Flow NameCert Construction **31414**

6357 A Concrete Naming Certificate for CompactMetricContinuousImageCompact$^{\uparrow}$ **31417**

6358 A Concrete Naming Certificate for TheoremGenomeEvolution$^{\uparrow}$ **31420**

6359 A Concrete Naming Certificate for ConstructiveUniformCompletion$^{\uparrow}$ **31423**

6360 A Concrete Naming Certificate for KalmanBucyFilter$^{\uparrow}$ **31426**

6361 Concrete Naming Certificate for EffectiveCauchyConvergenceModulus↑ **31428**

6362 Concrete Naming Certificate for RegularCauchyModulusAlgebra↑ **31431**

6363 Concrete Naming Certificate for HeineBorelFiniteInterval↑ **31434**

6364 Concrete Naming Certificate for BishopLocatedIntervalOscillation↑ **31437**

6365 Concrete Naming Certificate for LocatedDyadicCompletion↑ **31440**

6366 Sequentially Located Real NameCert Construction **31443**

6367 Concrete Naming Certificate for UpperLowerRealEnclosure↑ **31446**

6368 Concrete Naming Certificate for FrechetCompleteMetric↑ **31449**

6369 Concrete Naming Certificate for DyadicCauchyCriterion↑ **31452**

6370 Concrete Naming Certificate for BishopCauchyChoiceSequence↑ **31455**

6371 Concrete Naming Certificate for CauchyModulusDiagonal↑ **31458**

6372 Concrete Naming Certificate for TeichmullerSpace↑ **31461**

6373 Concrete Naming Certificate for ClosedObserverAuditTrace↑ **31464**

6374 Concrete Naming Certificate for FiniteDiniModulus↑ **31466**

6375 Concrete Naming Certificate for CompactRealUniformModulus↑ **31469**

6376 Concrete Naming Certificate for WeylCriterion↑ **31472**

6377 Concrete Naming Certificate for AsymptoticRegularity↑ **31475**

6378 Concrete Naming Certificate for PerronIntegral↑ **31478**

6379 Concrete Naming Certificate for BishopCauchyRateCompletion↑ **31481**

6380 Concrete Naming Certificate for ApophaticTLoop↑ **31484**

6381 Concrete Naming Certificate for BishopRegularCauchyEnvelope↑ **31487**

6382 Concrete Naming Certificate for CompletionDenseExtension↑ **31490**

6383 Concrete Naming Certificate for FiniteCoverModulusFunctional↑ **31493**

6384 Concrete Naming Certificate for ChoiceSequenceRealizability↑ **31496**

6385 Concrete Naming Certificate for HyperbolicPhaseHolonomy↑ **31498**

6386 Concrete Naming Certificate for FiniteVariationPartition↑ **31500**

6387 A Concrete Naming Certificate for BishopLocatedZero↑ **31503**

6388 Constructive Cauchy-Filter Boundary NameCert Construction **31506**

6389 A Concrete Naming Certificate for FiniteWindowRegSeqRatHandoff↑ **31508**

6390 A Concrete Naming Certificate for PrueferSequence↑ **31510**

6391 A Concrete Naming Certificate for BishopContractionEndpointSeal↑ **31513**

6392 A Concrete Naming Certificate for CompactDyadicGridRefinement↑ **31516**

6393 A Concrete Naming Certificate for LocatedCantorBendixson↑ **31519**

6394 A Concrete Naming Certificate for NetConvergenceCriterion↑ **31522**

6395 A Concrete Naming Certificate for CompactRadiusChain↑ **31525**

6396 A Concrete Naming Certificate for FreudenthalCompactification↑ **31528**

6397 A Concrete Naming Certificate for LocatedZeroUniqueness↑ **31530**

6398 A Concrete Naming Certificate for CountablyCompact↑ **31533**

6399 A Concrete Naming Certificate for ObservableOpenFilter↑ **31535**

6400 A Concrete Naming Certificate for PellEquation↑ **31538**

6401 A Concrete Naming Certificate for WeakStarCompactness↑ **31541**

6402 A Concrete Naming Certificate for LocatedDistanceFunction↑ **31544**

6403 A Concrete Naming Certificate for HermitianMatrix↑ **31547**

6404 A Concrete Naming Certificate for DiniModulusContinuity↑ **31550**

6405 A Concrete Naming Certificate for FiniteDiagonalModulusScheduler↑ **31553**

6406 A Concrete Naming Certificate for FiniteVariationPath↑ **31556**

6407 A Concrete Naming Certificate for BrzozowskiDerivative↑ **31559**

6408 A Concrete Naming Certificate for FreedmanInequality↑ **31562**

6409 A Concrete Naming Certificate for CompactUniformCoverRead↑ **31565**

6410 A Concrete Naming Certificate for ConstructiveIntermediateValue↑ **31568**

6411 A Concrete Naming Certificate for AsymptoticDensity↑ **31571**

6412 A Concrete Naming Certificate for CandidateDecidabilityTrace↑ **31574**

6413 Concrete Naming Certificate for BishopEquicontinuityModulus↑ **31577**

6414 Concrete Naming Certificate for FaberSchauderSystem↑ **31580**

6415 Concrete Naming Certificate for DyadicBisectionTree↑ **31583**

6416 Concrete Naming Certificate for KalmanDecomposition↑ **31585**

6417 Concrete Naming Certificate for BishopIntervalNestedLocator↑ **31588**

6418 Concrete Naming Certificate for MorseLemma↑ **31591**

6419 Concrete Naming Certificate for MetaCICSubstitutionDiamond↑ **31594**

6420 Concrete Naming Certificate for AbelSummability↑ **31597**

6421 Concrete Naming Certificate for BaireMetricCylinderBasis↑ **31600**

6422 Concrete Naming Certificate for BishopCompletionApproximationTheorem↑ **31603**

6423 Concrete Naming Certificate for LocatedDedekindRealComparison↑ **31606**

6424 Concrete Naming Certificate for CauchyTailMirror↑ **31609**

6425 Concrete Naming Certificate for LocatedCompactImage↑ **31612**

6426 Concrete Naming Certificate for LocatedImageModulus↑ **31615**

6427 Concrete Naming Certificate for RationalIncompletenessWitness↑ **31618**

6428 Concrete Naming Certificate for BaireMetricCompactWindow↑ **31621**

6429 Concrete Naming Certificate for BrouwerInvarianceDomain↑ **31624**

6430 Concrete Naming Certificate for HadamardGapSeries↑ **31627**

6431 Concrete Naming Certificate for ConstructiveIVTModulus↑ **31631**

6432 Concrete Naming Certificate for LocatedSupremumModulus↑ **31634**

6433 Concrete Naming Certificate for MetaCICTypedParallelDiamondBoundary↑ **31636**

6434 Concrete Naming Certificate for BishopLocatedFilterBase↑ **31639**

6435 Concrete Naming Certificate for CauchyCompletionUnitIsometry↑ **31642**

6436 Concrete Naming Certificate for DarbouxSums↑ **31645**

6437 Concrete Naming Certificate for HyperbolicPhaseCurvatureBudget↑ **31648**

6438 Concrete Naming Certificate for AlexandrovUrysohnMetrization↑ **31651**

6439 A Concrete Naming Certificate for CauchySelectionPrinciple↑ **31654**

6440 A Concrete Naming Certificate for RealCompletionSelectionGate↑ **31657**

6441 A Concrete Naming Certificate for BolzanoDedekindLocatedCluster↑ **31660**

6442 A Concrete Naming Certificate for BishopTightCauchyCriterion↑ **31663**

6443 A Concrete Naming Certificate for BishopFiniteIntervalChain↑ **31666**

6444 A Concrete Naming Certificate for CompactUniformModulusBridge↑ **31669**

6445 Sobolev trace boundary NameCert construction **31672**

6446 A Concrete Naming Certificate for LocatedIntervalArithmetic↑ **31675**

6447 A Concrete Naming Certificate for FittingIdeal↑ **31679**

6448 A Concrete Naming Certificate for AlternativeCodeReassignmentSpectra↑ **31681**

6449 A Concrete Naming Certificate for AsymptoticDimension↑ **31683**

6450 A Concrete Naming Certificate for LocatedRealApartness↑ **31686**

6451 A Concrete Naming Certificate for FiniteBishopConvergenceCriterion↑ **31689**

6452 A Concrete Naming Certificate for CauchyTailCriterion↑ **31692**

6453 A Concrete Naming Certificate for CompactNetModulusStability↑ **31695**

6454 A Concrete Naming Certificate for HausdorffMomentProblem↑ **31698**

6455 A Concrete Naming Certificate for UniformDyadicMesh↑ **31701**

6456 A Concrete Naming Certificate for BishopCompletionOperator↑ **31704**

6457 A Concrete Naming Certificate for BishopLocatedRealCompleteness↑ **31707**

6458 A Concrete Naming Certificate for PointwiseEquicontinuityCompactTransfer↑ **31709**

6459 A Concrete Naming Certificate for BishopCauchyConvergenceCriterion↑ **31711**

6460 A Concrete Naming Certificate for RegularCauchyUniformLimit↑ **31715**

6461 A Concrete Naming Certificate for RegularCauchyEqualityCriterion↑ **31718**

6462 A Concrete Naming Certificate for LocatedIntervalFanModulus↑ **31721**

6463 A Concrete Naming Certificate for TukeyOrder↑ **31724**

6464 A Concrete Naming Certificate for SemidecidableOpen↑ **31727**

6465 Concrete Naming Certificate for CauchyCondensationIntegralBound↑ **31729**

6466 Concrete Naming Certificate for CompactUniformTailSelector↑ **31732**

6467 Concrete Naming Certificate for FinePartitionGauge↑ **31736**

6468 Concrete Naming Certificate for JamesSpace↑ **31739**

6469 Concrete Naming Certificate for AbsoluteRetract↑ **31742**

6470 Concrete Naming Certificate for GromovBoundary↑ **31744**

6471 Concrete Naming Certificate for HausdorffYoung↑ **31746**

6472 Concrete Naming Certificate for BishopCauchyModulusCofinality↑ **31749**

6473 Concrete Naming Certificate for BaireUniformSpace↑ **31752**

6474 Concrete Naming Certificate for MetacicAppArgumentStabilityLedger↑ **31755**

6475 Concrete Naming Certificate for CauchyRetractionModulus↑ **31758**

6476 Concrete Naming Certificate for LocatedRealApartnessCotransitivity↑ **31761**

6477 Concrete Naming Certificate for HyperbolicUniformModulus↑ **31764**

6478 Concrete Naming Certificate for LocatedInfimumModulus↑ **31767**

6479 Concrete Naming Certificate for BingMetrization↑ **31770**

6480 Concrete Naming Certificate for ProbabilityGeneratingFunction↑ **31773**

6481 Concrete Naming Certificate for HyperbolicPhaseEnergyLedger↑ **31776**

6482 Concrete Naming Certificate for Hyperfunction↑ **31779**

6483 Concrete Naming Certificate for NestedShrinkingInterval↑ **31782**

6484 Concrete Naming Certificate for CauchySequenceInterleaving↑ **31785**

6485 Concrete Naming Certificate for LocatedUniformCauchyFilter↑ **31788**

6486 Concrete Naming Certificate for KochSnowflake↑ **31790**

6487 Concrete Naming Certificate for HilbertInequality↑ **31793**

6488 Concrete Naming Certificate for FinitelyBranchingTree↑ **31796**

6489 Concrete Naming Certificate for CompactTailSelector↑ **31799**

6490 Concrete Naming Certificate for CompletionUnitDenseImage↑ **31802**

6491 Concrete Naming Certificate for BishopRealExtensionality↑ **31805**

6492 Concrete Naming Certificate for LocatedCoverSelection↑ **31808**

6493 Concrete Naming Certificate for KrasnoselskiiFixedPoint↑ **31811**

6494 Concrete Naming Certificate for FiniteSubstrateMap↑ **31814**

6495 Concrete Naming Certificate for SoberSpace↑ **31817**

6496 Concrete Naming Certificate for RealLocatedSupremum↑ **31821**

6497 Concrete Naming Certificate for HyperbolicAxisProjection↑ **31824**

6498 Concrete Naming Certificate for FrechetFilterBasis↑ **31827**

6499 Concrete Naming Certificate for CompactFiniteModulusGluing↑ **31830**

6500 Concrete Naming Certificate for ClosedIntervalFiniteNet↑ **31832**

6501 Concrete Naming Certificate for BoundedLattice↑ **31835**

6502 Concrete Naming Certificate for RegularCauchyComparisonModulus↑ **31838**

6503 Concrete Naming Certificate for PolarizationIdentity↑ **31841**

6504 Concrete Naming Certificate for ModulusOfSequentialContinuity↑ **31844**

6505 Concrete Naming Certificate for RouthHurwitzCriterion↑ **31847**

6506 Concrete Naming Certificate for EulerCharacteristic↑ **31850**

6507 Concrete Naming Certificate for VietorisHyperspace↑ **31853**

6508 Concrete Naming Certificate for EventInstance↑ **31856**

6509 Concrete Naming Certificate for AscoliCompactOperator↑ **31859**

6510 Concrete Naming Certificate for ConsciousObservationRoute↑ **31862**

6511 Concrete Naming Certificate for EffectiveBishopCompletion↑ **31865**

6512 Concrete Naming Certificate for FilterSubbasisCofinalCompletion↑ **31868**

6513 Concrete Naming Certificate for MooreAronszajnKernel↑ **31871**

6514 Concrete Naming Certificate for BishopBarInductionBoundary↑ **31874**

6515 Concrete Naming Certificate for UniformizableSpace↑ **31877**

6516 Concrete Naming Certificate for LeibnizAlternatingTest↑ **31880**

6517 Concrete Naming Certificate for Rule110CellularTrustSubstrate↑ **31883**

6518 Concrete Naming Certificate for FiniteSectionCompletionInterface↑ **31886**

6519 Concrete Naming Certificate for RegularCauchyCompletionComparisonTriangle↑ **31888**

6520 Concrete Naming Certificate for SequentialFan↑ **31891**

6521 Concrete Naming Certificate for FrechetUrysohnFan↑ **31894**

6522 Concrete Naming Certificate for BishopFanCompactUniform↑ **31896**

6523 Concrete Naming Certificate for PolishCompletion↑ **31899**

6524 Concrete Naming Certificate for SteffensenAcceleration↑ **31902**

6525 Concrete Naming Certificate for ThomaeFunction↑ **31905**

6526 Concrete Naming Certificate for RegularSequenceApartness↑ **31908**

6527 Concrete Naming Certificate for LocatedInverseBranch↑ **31911**

6528 Concrete Naming Certificate for StarRefinementModulus↑ **31915**

6529 Concrete Naming Certificate for FourierSeries↑ **31918**

6530 Concrete Naming Certificate for OpenDenseSequence↑ **31921**

6531 Concrete Naming Certificate for RegularCauchyFusion↑ **31923**

6532 Concrete Naming Certificate for WhitneyEmbeddingFiniteAtlas↑ **31926**

6533 Concrete Naming Certificate for CompactExhaustion↑ **31929**

6534 Concrete Naming Certificate for PhysicalInductionGate↑ **31932**

6535 Concrete Naming Certificate for SemiAlgebraicSet↑ **31935**

6536 Concrete Naming Certificate for FiniteMinimax↑ **31938**

6537 Concrete Naming Certificate for FourierPlancherel↑ **31940**

6538 Concrete Naming Certificate for EffectiveTotalBoundedCompletion↑ **31943**

6539 Concrete Naming Certificate for AscoliEquicontinuityCriterion↑ **31946**

6540 Concrete Naming Certificate for ArithmetizedProvability↑ **31949**

6541 Concrete Naming Certificate for MackeyDecomposition↑ **31952**

6542 Concrete Naming Certificate for DoobConvergence↑ **31955**

6543 Concrete Naming Certificate for MetrizableSpace↑ **31958**

6544 Concrete Naming Certificate for HyperbolicIdealTriangle↑ **31961**

6545 Concrete Naming Certificate for BishopRegularCompletionReflector↑ **31964**

6546 Concrete Naming Certificate for RegularRealOscillationBoundary↑ **31966**

6547 Concrete Naming Certificate for MetaCICConfluenceDiamondLedger↑ **31969**

6548 Concrete Naming Certificate for LocatedCompactOvert↑ **31973**

6549 Concrete Naming Certificate for CountablyCompactMetric↑ **31976**

6550 Concrete Naming Certificate for DirichletUniformTailSelector↑ **31979**

6551 Concrete Naming Certificate for FiniteMetricTree↑ **31982**

6552 Concrete Naming Certificate for MetaCICCandidateDiamondBudget↑ **31985**

6553 Concrete Naming Certificate for UpperHemicontinuity↑ **31988**

6554 Concrete Naming Certificate for UrysohnUniversalMetric↑ **31991**

6555 Concrete Naming Certificate for PointwiseBoundedRealFamily↑ **31995**

6556 Concrete Naming Certificate for Submartingale↑ **31998**

6557 Concrete Naming Certificate for LefschetzFixedPoint↑ **32003**

6558 Concrete Naming Certificate for RefutationGate↑ **32006**

6559 Concrete Naming Certificate for OrbitStabilizer↑ **32009**

6560 Concrete Naming Certificate for PseudotheoryRefutationGate↑ **32012**

6561 Concrete Naming Certificate for RegularCauchySeparatedCompletion↑ **32015**

6562 Concrete Naming Certificate for AbsoluteNeighborhoodRetract↑ **32018**

6563 Concrete Naming Certificate for RegulatedCauchyMap↑ **32020**

6564 Concrete Naming Certificate for RealUniformCauchyFilterCompletion↑ **32023**

6565 Concrete Naming Certificate for BaireOneOscillationRank↑ **32025**

6566 Concrete Naming Certificate for CompactUniformModulusFold↑ **32028**

6567 Concrete Naming Certificate for CompactClosedMap↑ **32031**

6568 Concrete Naming Certificate for RegularSequenceComparison↑ **32034**

656A Concrete Naming Certificate for HyperbolicHoroballShadow↑ **32037**

657A Concrete Naming Certificate for BoundedVariationSequence↑ **32040**

657A Concrete Naming Certificate for DyadicEpsilonNet↑ **32043**

657A Concrete Naming Certificate for AlexandrovTopology↑ **32046**

657A Concrete Naming Certificate for RegularCauchyTriangleInequality↑ **32050**

657A Concrete Naming Certificate for RegularCauchyFiniteMaximum↑ **32053**

657A Concrete Naming Certificate for FermatStationaryPoint↑ **32057**

657A Concrete Naming Certificate for OptionalStieltjesModulus↑ **32060**

657A Concrete Naming Certificate for MetaCICCandidateNormalizationWitness↑ **32063**

657A Concrete Naming Certificate for ConvergenceFilter↑ **32067**

657A Concrete Naming Certificate for SetoidCompletion↑ **32070**

658A Concrete Naming Certificate for FilterLimitCriterion↑ **32073**

658RegularCauchyInterleavingStability NameCert Construction **32076**

658A Concrete Naming Certificate for LowerUpperRealSeparator↑ **32079**

658A Concrete Naming Certificate for KolmogorovRieszFiniteTranslation↑ **32082**

658A Concrete Naming Certificate for CauchyRegularExtraction↑ **32085**

658A Concrete Naming Certificate for ConvexCompactInterval↑ **32088**

658A Concrete Naming Certificate for EffectiveSeparableMetric↑ **32091**

658A Concrete Naming Certificate for BochnerIntegral↑ **32094**

658A Concrete Naming Certificate for PettisMeasurability↑ **32097**

658A Concrete Naming Certificate for SmithVolterraCantor↑ **32100**

659A Concrete Naming Certificate for MuckenhouptWeight↑ **32103**

659A Concrete Naming Certificate for KuratowskiConvergence↑ **32106**

659A Concrete Naming Certificate for ChebyshevInequality↑ **32109**

659A Concrete Naming Certificate for CauchyStabilityModulus↑ **32112**

659A Concrete Naming Certificate for RegularCauchyStabilityModulus↑ **32115**

5945 Concrete Naming Certificate for HyperbolicBoundaryTransport↑ **32118**

5946 Concrete Naming Certificate for EffectiveCompactness↑ **32120**

5947 Concrete Naming Certificate for CompactlySupportedRealFunction↑ **32123**

5948 Concrete Naming Certificate for CompactMetricUniformContinuity↑ **32126**

5949 Concrete Naming Certificate for BishopContractionIteration↑ **32129**

5950 Concrete Naming Certificate for RealOscillationMajorant↑ **32133**

5951 Concrete Naming Certificate for MarkovInequality↑ **32136**

5952 Concrete Naming Certificate for PolarDecomposition↑ **32139**

5953 Concrete Naming Certificate for LocatedBolzanoWeierstrass↑ **32142**

5954 Concrete Naming Certificate for RegularCauchyLiftedLimit↑ **32145**

5955 Concrete Naming Certificate for LocatedSetDistance↑ **32148**

5956 Concrete Naming Certificate for CauchyProductTailBudget↑ **32151**

5957 Concrete Naming Certificate for CompactIntervalFiniteCoverRefinement↑ **32154**

5958 Concrete Naming Certificate for PontryaginMaximumPrinciple↑ **32157**

5959 Concrete Naming Certificate for HyperbolicVisualFlowShadow↑ **32160**

5960 Concrete Naming Certificate for KripkeSpread↑ **32162**

5961 Concrete Naming Certificate for ChoquetSimplex↑ **32165**

5962 Concrete Naming Certificate for DyadicOrderTopology↑ **32168**

5963 Concrete Naming Certificate for RealBorelCode↑ **32170**

5964 Concrete Naming Certificate for CompactFamilyModulusRouter↑ **32173**

5965 Concrete Naming Certificate for DynkinPiLambda↑ **32176**

5966 Concrete Naming Certificate for BishopLocatedUniformConvergence↑ **32179**

5967 Concrete Naming Certificate for MetaCICTypedNormalFormScheduler↑ **32182**

5968 Concrete Naming Certificate for SearchableCompact↑ **32185**

5969 Concrete Naming Certificate for GalerkinErrorEstimate↑ **32188**

5970 Concrete Naming Certificate for TotallyBoundedLocatedSet↑ **32191**

6621 Concrete Naming Certificate for MazurTheorem↑ **32194**

6622 Concrete Naming Certificate for PolishCompletionCriterion↑ **32196**

6623 Concrete Naming Certificate for ChebyshevCenter↑ **32199**

6624 Concrete Naming Certificate for LocatedSequentialCompleteness↑ **32202**

6625 Concrete Naming Certificate for CertifiedModelQualityLedger↑ **32205**

6626 Concrete Naming Certificate for ValuationDomain↑ **32208**

6627 Concrete Naming Certificate for FastCauchyAccelerationSelector↑ **32211**

6628 Concrete Naming Certificate for LikelihoodRatioTest↑ **32214**

6629 Concrete Naming Certificate for SequentialProbabilityRatioTest↑ **32217**

6630 Compact Completion Dense Image Name Certificate **32221**

6631 Concrete Naming Certificate for ClassEquation↑ **32224**

6632 Concrete Naming Certificate for ConstructiveUltrafilterBoundary↑ **32226**

6633 Concrete Naming Certificate for CompactLebesgueNumberLedger↑ **32229**

6634 Concrete Naming Certificate for HankelOperator↑ **32232**

6635 Concrete Naming Certificate for BaireFunctionRank↑ **32235**

6636 Concrete Naming Certificate for BanachContractionRealCompletion↑ **32238**

6637 Concrete Naming Certificate for EffectiveCauchyCompleteness↑ **32241**

6638 Concrete Naming Certificate for PettisIntegral↑ **32244**

6639 Concrete Naming Certificate for LebesgueNumberFunction↑ **32247**

6640 Concrete Naming Certificate for LocatedRealNumber↑ **32250**

6641 Concrete Naming Certificate for CompactUniformLimit↑ **32253**

6642 Concrete Naming Certificate for LocatedCauchyModulusSelector↑ **32256**

6643 Concrete Naming Certificate for CartesianClosedCategory↑ **32259**

6644 Concrete Naming Certificate for RealCompletionWitnessTransducer↑ **32262**

6645 Concrete Naming Certificate for CoarseMetricSpace↑ **32265**

6646 Concrete Naming Certificate for MetricSequentialSpace↑ **32268**

6647 Concrete Naming Certificate for KuratowskiClusterSet↑ **32271**

6648 Concrete Naming Certificate for BishopLocatedSubset↑ **32274**

6649 Concrete Naming Certificate for BernoulliInequality↑ **32277**

6650 Concrete Naming Certificate for DependentEliminatorCoverage↑ **32280**

6651 Concrete Naming Certificate for RealModulusSpace↑ **32283**

6652 Concrete Naming Certificate for CauchyPreservingMap↑ **32286**

6653 Concrete Naming Certificate for RegularCauchyCompleteness↑ **32289**

6654 Concrete Naming Certificate for LocatedApartnessInterval↑ **32292**

6655 Concrete Naming Certificate for AdamsBashforthFiniteStep↑ **32295**

6656 Concrete Naming Certificate for CofinalCauchySubnet↑ **32297**

6657 Concrete Naming Certificate for SemiDecidableRealCut↑ **32300**

6658 Concrete Naming Certificate for CofinalMonotoneSubsequence↑ **32303**

6659 Concrete Naming Certificate for RegularExpression↑ **32306**

6660 Concrete Naming Certificate for CompleteMetricSequentialClosure↑ **32309**

6661 Concrete Naming Certificate for RegSeqRatOperationLedger↑ **32312**

6662 Concrete Naming Certificate for ConstructiveCauchyDini↑ **32315**

6663 Concrete Naming Certificate for ClosedParallelSubstitutionBudget↑ **32318**

6664 Concrete Naming Certificate for InitialTopology↑ **32321**

6665 Concrete Naming Certificate for CompactGraphContinuity↑ **32323**

6666 Concrete Naming Certificate for MetaCICResidualCheckerNormalizationBudget↑ **32326**

6667 Concrete Naming Certificate for SequentialOrderTopology↑ **32329**

6668 Concrete Naming Certificate for HermiteInterpolation↑ **32332**

6669 Concrete Naming Certificate for BirkhoffInterpolation↑ **32335**

6670 Concrete Naming Certificate for BaireDenseOpen↑ **32338**

6671 Concrete Naming Certificate for KKMFiniteCover↑ **32341**

6672 Concrete Naming Certificate for RegularCauchyTailCollapse↑ **32343**

6673 A Concrete Naming Certificate for SuslinTree↑ **32346**

6674 A Concrete Naming Certificate for TaylorPolynomial↑ **32349**

6675 A Concrete Naming Certificate for FiniteDisjunctionStabilization↑ **32353**

6676 A Concrete Naming Certificate for CompactGridModulusExtraction↑ **32356**

6677 A Concrete Naming Certificate for BishopChoiceBar↑ **32359**

6678 A Concrete Naming Certificate for HyperbolicBoundaryShadowTransport↑ **32362**

6679 A Concrete Naming Certificate for MorseBottLemma↑ **32365**

6680 A Concrete Naming Certificate for ChoquetIntegral↑ **32368**

6681 A Concrete Naming Certificate for BishopCountableDenseRationalGrid↑ **32371**

6682 A Concrete Naming Certificate for SignedDigitIntervalBisection↑ **32374**

6683 A Concrete Naming Certificate for ChoiceFreeDisplayedCompletion↑ **32377**

6684 A Concrete Naming Certificate for SignedDigitRealExpansion↑ **32380**

6685 A Concrete Naming Certificate for RegularCauchyArithmetic↑ **32383**

6686 A Concrete Naming Certificate for CauchyOrderCompletion↑ **32386**

6687 A Concrete Naming Certificate for AttouchWetsConvergence↑ **32389**

6688 A Concrete Naming Certificate for MilmanPettis↑ **32392**

6689 A Concrete Naming Certificate for ModulusOfConvexity↑ **32395**

6690 A Concrete Naming Certificate for RegularCauchyDiagonalExchange↑ **32398**

6691 A Concrete Naming Certificate for DyadicRegularErrorEnvelope↑ **32401**

6692 A Concrete Naming Certificate for MetacicClosedBetaTraceEnvelope↑ **32404**

6693 A Concrete Naming Certificate for SkorokhodTopology↑ **32407**

6694 A Concrete Naming Certificate for JordanHolder↑ **32410**

6695 Bishop Cauchy Real Envelope NameCert Construction **32412**

6696 A Concrete Naming Certificate for FiniteUnionBound↑ **32415**

6697 A Concrete Naming Certificate for BanachLimit↑ **32418**

6698 A Concrete Naming Certificate for RiemannSphere↑ **32421**

6699 Cauchy Sign-Flip Budget NameCert Construction 32424

6700 Hyperbolic Horocycle Return Ledger NameCert Construction 32427

6701 A Concrete Naming Certificate for IntervalHalvingTree$^{\uparrow}$ **32430**

6702 A Concrete Naming Certificate for CauchyCondensationIntegralCriterion$^{\uparrow}$ **32433**

6703 A Concrete Naming Certificate for ChernoffFiniteExponentialBound$^{\uparrow}$ **32436**

6704 A Concrete Naming Certificate for EffectiveModulusCompactFamily$^{\uparrow}$ **32439**

6705 A Concrete Naming Certificate for HermiteNormalForm$^{\uparrow}$ **32442**

6706 A Concrete Naming Certificate for LebesgueDifferentiationBasis$^{\uparrow}$ **32445**

6707 A Concrete Naming Certificate for HyperbolicShadowing$^{\uparrow}$ **32448**

6708 A Concrete Naming Certificate for FiniteGaugeCover$^{\uparrow}$ **32451**

6709 A Concrete Naming Certificate for WeierstrassRootTest$^{\uparrow}$ **32454**

6710 A Concrete Naming Certificate for DvoretzkyFiniteDimensionalSection$^{\uparrow}$ **32457**

6711 A Concrete Naming Certificate for FreydMitchellEmbedding$^{\uparrow}$ **32460**

6711.0 Carrier 32460

6712 A Concrete Naming Certificate for SequentialCauchyFilter$^{\uparrow}$ **32462**

6713 A Concrete Naming Certificate for RegularCauchyFilterSeal$^{\uparrow}$ **32465**

6714 A Concrete Naming Certificate for DAlembertRatioTest$^{\uparrow}$ **32468**

6715 A Concrete Naming Certificate for BourbakiWitt$^{\uparrow}$ **32472**

6716 A Concrete Naming Certificate for CompactChainUniformModulus$^{\uparrow}$ **32475**

6717 A Concrete Naming Certificate for RecursivePolishPresentation$^{\uparrow}$ **32478**

6718 A Concrete Naming Certificate for GrothendieckTopology$^{\uparrow}$ **32481**

6719 A Concrete Naming Certificate for SobolevMorreyEmbedding$^{\uparrow}$ **32484**

6720 A Concrete Naming Certificate for CantorCompleteMetricModulus$^{\uparrow}$ **32487**

6721 Real Archimedean Cutoff NameCert Construction 32490

6722 A Concrete Naming Certificate for CauchyMerge$^{\uparrow}$ **32493**

6723 A Concrete Naming Certificate for EventuallyConstantSequence$^{\uparrow}$ **32496**

6724 A Concrete Naming Certificate for SlowlyVaryingSequence$^{\uparrow}$ **32499**

6725 Concrete Naming Certificate for HaltingRefusalLedger↑ **32501**

6726 Concrete Naming Certificate for CauchyNetTailFilter↑ **32503**

6727 Concrete Naming Certificate for BishopRegularCauchyRealizer↑ **32506**

6728 Concrete Naming Certificate for RiemannStieltjesIntegral↑ **32509**

6729 Concrete Naming Certificate for PrecompactFunctionFamily↑ **32511**

6730 Concrete Naming Certificate for RegularCauchyLocatedApartness↑ **32514**

6731 Concrete Naming Certificate for OpenCover↑ **32517**

6732 Concrete Naming Certificate for CompleteProductMetric↑ **32520**

6733 Concrete Naming Certificate for RegularCauchyProductMetric↑ **32522**

6734 Concrete Naming Certificate for HadamardSpace↑ **32524**

6735 Concrete Naming Certificate for MarkovKernel↑ **32527**

6736 Concrete Naming Certificate for ToeplitzSummability↑ **32530**

6737 Concrete Naming Certificate for DoobDecomposition↑ **32534**

6738 Concrete Naming Certificate for ContinuedFractionReal↑ **32537**

6739 Concrete Naming Certificate for LatinSquare↑ **32540**

6740 Concrete Naming Certificate for MayerVietorisSequence↑ **32543**

6741 Concrete Naming Certificate for RealLineOrderTopology↑ **32546**

6742 Concrete Naming Certificate for UltrametricBaireTheorem↑ **32549**

6743 Concrete Naming Certificate for AsplundSpace↑ **32552**

6744 Concrete Naming Certificate for CompactUniformEquicontinuityImage↑ **32555**

6745 Concrete Naming Certificate for MoscoRecoverySequence↑ **32558**

6746 Concrete Naming Certificate for RealCauchyFilterMonadUnit↑ **32561**

6747 Concrete Naming Certificate for ClosedObservationProofTrace↑ **32564**

6748 Concrete Naming Certificate for RegularCauchyTailShift↑ **32567**

6749 Concrete Naming Certificate for BishopCauchyNameSeal↑ **32570**

6750 Concrete Naming Certificate for CompactModulusExtraction↑ **32573**

6751 Concrete Naming Certificate for AlexandroffDuplicate↑ **32576**

6752 Concrete Naming Certificate for FormalBallSpace↑ **32578**

6753 Concrete Naming Certificate for BishopTotallyBoundedCompletionNet↑ **32581**

6754 Concrete Naming Certificate for DicksonLemma↑ **32584**

6755 Concrete Naming Certificate for CechNerve↑ **32587**

6756 Concrete Naming Certificate for CompactOpenExhaustion↑ **32590**

6757 Concrete Naming Certificate for AubinNitsche↑ **32592**

6758 Concrete Naming Certificate for CantorTernaryExpansion↑ **32595**

6759 Concrete Naming Certificate for DyadicWindowReflector↑ **32598**

6760 Concrete Naming Certificate for FanBarInduction↑ **32601**

6761 Concrete Naming Certificate for CauchyApproximationModulus↑ **32604**

6762 Concrete Naming Certificate for RegularCauchyBallBasis↑ **32607**

6763 Concrete Naming Certificate for CauchyMeanValueTheorem↑ **32610**

6764 Concrete Naming Certificate for RungeKuttaFiniteStep↑ **32613**

6765 Concrete Naming Certificate for ClassifierChannelShiftLedger↑ **32616**

6766 Concrete Naming Certificate for ModulatedCauchyCompletion↑ **32619**

6767 Concrete Naming Certificate for LogicalRelation↑ **32621**

6768 Concrete Naming Certificate for CauchyCompletionEqualizer↑ **32623**

6769 Concrete Naming Certificate for HeineCantorFiniteNet↑ **32626**

6770 Concrete Naming Certificate for BishopLocatedMaximum↑ **32629**

6771 Concrete Naming Certificate for FiniteMetricCompactness↑ **32632**

6772 Concrete Naming Certificate for FortissimoSpace↑ **32635**

6773 Concrete Naming Certificate for FiniteSignedMeasure↑ **32638**

6774 Concrete Naming Certificate for RegularCauchyAsymptoticCenter↑ **32641**

6775 Concrete Naming Certificate for FiniteOpenIntervalCover↑ **32644**

6776 Concrete Naming Certificate for PoincareDiskAutomorphism↑ **32647**

677A Concrete Naming Certificate for HedgehogSpace↑ **32650**

677B Concrete Naming Certificate for BishopGraphCompletion↑ **32653**

677C Concrete Naming Certificate for CompletionExactness↑ **32656**

678D Concrete Naming Certificate for TychonoffPlank↑ **32659**

678E Concrete Naming Certificate for DeletedTychonoffPlank↑ **32661**

678F Concrete Naming Certificate for LocatedUniformExtremeValue↑ **32664**

678G Concrete Naming Certificate for BishopLocatedOscillation↑ **32668**

678H Concrete Naming Certificate for DyadicNestedIntervalSelector↑ **32671**

678J Concrete Naming Certificate for FinePartitionRefinement↑ **32675**

678K Concrete Naming Certificate for EnergyRefinedPrivateCoordinate↑ **32678**

678L Concrete Naming Certificate for BoundedOscillation↑ **32681**

678M Concrete Naming Certificate for MetricGraph↑ **32684**

678N Concrete Naming Certificate for KleeneKreiselSpace↑ **32687**

679P Concrete Naming Certificate for WallisProduct↑ **32690**

679Q Concrete Naming Certificate for BishopLocatedMinimum↑ **32693**

679R Concrete Naming Certificate for BishopChoiceSequence↑ **32696**

679S Concrete Naming Certificate for SquareRootContinuityModulus↑ **32699**

679T Concrete Naming Certificate for BishopIntervalMap↑ **32702**

679V Concrete Naming Certificate for SymmetricAlgebra↑ **32705**

679W Concrete Naming Certificate for BishopCompactRegularCover↑ **32708**

679X Concrete Naming Certificate for ModulatedConvergence↑ **32711**

679Y Concrete Naming Certificate for T1Space↑ **32714**

679Z Concrete Naming Certificate for PrueferSurface↑ **32717**

680B Concrete Naming Certificate for OvertSpace↑ **32720**

680C Concrete Naming Certificate for KorovkinTheorem↑ **32722**

680D Concrete Naming Certificate for MetaCICConfluenceInventory↑ **32725**

6803 Concrete Naming Certificate for BrownRepresentability↑ **32728**

6804 Concrete Naming Certificate for CauchyCompleteSpace↑ **32731**

6805 Concrete Naming Certificate for RiemannMapping↑ **32734**

6806 Concrete Naming Certificate for RealNameGauge↑ **32737**

6807 Concrete Naming Certificate for CompactUniformModulusLadder↑ **32739**

6808 Concrete Naming Certificate for TaylorSeries↑ **32741**

6809 Concrete Naming Certificate for RealCompletionModulusEnvelope↑ **32744**

6810 Concrete Naming Certificate for InjectiveEnvelopeMetric↑ **32747**

6811 Concrete Naming Certificate for RealitySignatureErrorBudget↑ **32750**

6812 Concrete Naming Certificate for ConstructiveRiemannMapping↑ **32753**

6813 Concrete Naming Certificate for RegularCauchyCompletionSpace↑ **32757**

6814 Concrete Naming Certificate for GagliardoNirenbergFiniteWindow↑ **32760**

6815 Concrete Naming Certificate for EhresmannConnection↑ **32763**

6816 Concrete Naming Certificate for WijsmanConvergence↑ **32766**

6817 Concrete Naming Certificate for JacksonTheorem↑ **32769**

6818 Concrete Naming Certificate for RegulatedFunctionCompactOscillation↑ **32772**

6819 Concrete Naming Certificate for ArithmeticalContinuum↑ **32775**

6820 Concrete Naming Certificate for RootChoiceSeparation↑ **32778**

6821 Concrete Naming Certificate for DedekindCompletionBoundary↑ **32781**

6822 Concrete Naming Certificate for RegularCauchyFinitePrefixModulus↑ **32784**

6823 Concrete Naming Certificate for MinkowskiQuestionMark↑ **32787**

6824 Concrete Naming Certificate for CompactFanModulusSelector↑ **32790**

6825 Concrete Naming Certificate for KuratowskiZornLemmaBoundary↑ **32793**

6826 Concrete Naming Certificate for ClassifyingSpace↑ **32796**

6827 Concrete Naming Certificate for RealDedekindCompletenessBoundary↑ **32798**

6828 Concrete Naming Certificate for DyadicIntervalCompletion↑ **32800**

6829 A Concrete Naming Certificate for HopfFibration↑ **32803**

6830 A Concrete Naming Certificate for Kelleyfication↑ **32806**

6831 A Concrete Naming Certificate for FiniteObservationTopology↑ **32815**

6832 CantorModulusContinuity name certificate construction 32817

6833 MetaCICClosedNormalJoinCertificate name certificate construction 32820

6834 A Concrete Naming Certificate for MetaCICSubjectReductionSocketCut↑ **32823**

6835 A Concrete Naming Certificate for RegularOpenAlgebra↑ **32826**

6836 A Concrete Naming Certificate for HeineCantorFiniteModulus↑ **32829**

6837 A Concrete Naming Certificate for SobolevPoincareFiniteEnergy↑ **32831**

6838 A Concrete Naming Certificate for PartialityBoundary↑ **32834**

6839 Sierpinski Continuity Classifier NameCert Construction 32837

6840 A Concrete Naming Certificate for CauchyIntersectionPrinciple↑ **32840**

6841 A Concrete Naming Certificate for RegularCauchyClamp↑ **32843**

6842 A Concrete Naming Certificate for DyadicRadiusHalvingLadder↑ **32846**

6843 A Concrete Naming Certificate for ArchimedeanCauchyTailModulus↑ **32849**

6844 A Concrete Naming Certificate for ConsistencyObservationBudget↑ **32852**

6845 A Concrete Naming Certificate for ArchimedeanDyadicApproximation↑ **32855**

6846 A Concrete Naming Certificate for CompactUniformMeshModulus↑ **32857**

6847 A Concrete Naming Certificate for FiniteHausdorffMetricApproximation↑ **32860**

6848 A Concrete Naming Certificate for BishopIntervalSelection↑ **32863**

6849 A Concrete Naming Certificate for RuleOneTenCookFrontierBudget↑ **32866**

6850 A Concrete Naming Certificate for LinearlyOrderedReal↑ **32869**

6851 A Concrete Naming Certificate for WeakCompactness↑ **32872**

6852 A Concrete Naming Certificate for KleeneRecursionTheorem↑ **32875**

6853 A Concrete Naming Certificate for BaireCategoryDenseReflector↑ **32878**

6854 A Concrete Naming Certificate for LocatedCutApartness↑ **32881**

6855 Concrete Naming Certificate for CauchyFilterCategory↑ **32884**

6856 Concrete Naming Certificate for JordanNormalForm↑ **32887**

6857 Concrete Naming Certificate for MetacicResidualDiamondScheduler↑ **32890**

6858 Concrete Naming Certificate for LocalEmpiricalResidue↑ **32893**

6859 Concrete Naming Certificate for MengerCurvature↑ **32896**

6860 Concrete Naming Certificate for DualBasis↑ **32899**

6861 Concrete Naming Certificate for SequentialCoreflection↑ **32902**

6862 Concrete Naming Certificate for GromovHyperbolicMetric↑ **32905**

6863 Concrete Naming Certificate for BrentSalaminPi↑ **32908**

6864 Concrete Naming Certificate for FanModulusRealCompletionBoundary↑ **32911**

6865 Concrete Naming Certificate for BishopIntervalChoice↑ **32914**

6866 Concrete Naming Certificate for PathMetric↑ **32917**

6867 Concrete Naming Certificate for LocatedRange↑ **32920**

6868 Concrete Naming Certificate for BishopCauchyTailModulus↑ **32923**

6869 Concrete Naming Certificate for CauchyDiagonalUniformModulus↑ **32926**

6870 Concrete Naming Certificate for DyadicDecimalComparison↑ **32929**

6871 Concrete Naming Certificate for PringsheimDoubleSequence↑ **32932**

6872 Concrete Naming Certificate for GroundCompilerAuditMapClosure↑ **32935**

6873 Concrete Naming Certificate for CauchyFilterDiagonal↑ **32938**

6874 Concrete Naming Certificate for RealSeparationViaUniformity↑ **32941**

6875 Concrete Naming Certificate for RegularCauchyFilterComparison↑ **32944**

6876 Concrete Naming Certificate for FiniteBetaClosednessIteration↑ **32947**

6877 Concrete Naming Certificate for Wronskian↑ **32949**

6878 Concrete Naming Certificate for LandauKolmogorovInequality↑ **32960**

6879 Concrete Naming Certificate for CauchyNetDiagonal↑ **32963**

6880 Concrete Naming Certificate for PrueferGroup↑ **32966**

6881 Concrete Naming Certificate for CauchyDiagonalCriterion$^{\uparrow}$ **32969**

6882 Concrete Naming Certificate for EgorovTheorem$^{\uparrow}$ **32972**

6883 Concrete Naming Certificate for PolymorphicInstantiationCoverage$^{\uparrow}$ **32975**

6884 Concrete Naming Certificate for UniformlySmoothSpace$^{\uparrow}$ **32979**

6885 Concrete Naming Certificate for PathConnectedSpace$^{\uparrow}$ **32982**

6886 Concrete Naming Certificate for TubularNeighborhood$^{\uparrow}$ **32985**

6887 Concrete Naming Certificate for KolmogorovThreeSeriesBoundary$^{\uparrow}$ **32988**

6888 Concrete Naming Certificate for AumannIntegralBoundary$^{\uparrow}$ **32991**

6889 Concrete Naming Certificate for CompactRegularRealFunction$^{\uparrow}$ **32994**

6890 Concrete Naming Certificate for MetacicClosedTypingFragment$^{\uparrow}$ **32997**

6891 Concrete Naming Certificate for HitMissConvergence$^{\uparrow}$ **33000**

6892 Concrete Naming Certificate for IntervalBisectionSearch$^{\uparrow}$ **33003**

6893 Concrete Naming Certificate for OstrowskiCompletion$^{\uparrow}$ **33006**

6894 Concrete Naming Certificate for BishopSplineSpace$^{\uparrow}$ **33009**

6895 Concrete Naming Certificate for FiniteSupportingHyperplane$^{\uparrow}$ **33012**

6896 Concrete Naming Certificate for WeakConvergenceHilbert$^{\uparrow}$ **33014**

6897 Concrete Naming Certificate for CarlemanInequality$^{\uparrow}$ **33017**

6898 Concrete Naming Certificate for CauchySchwarzCompletion$^{\uparrow}$ **33019**

6899 Concrete Naming Certificate for ArtinRees$^{\uparrow}$ **33022**

6900 Concrete Naming Certificate for FloorFunction$^{\uparrow}$ **33024**

6901 Concrete Naming Certificate for CeilingFunction$^{\uparrow}$ **33027**

6902 Concrete Naming Certificate for CompleteMetricFixedPoint$^{\uparrow}$ **33031**

6903 Concrete Naming Certificate for UniformBar$^{\uparrow}$ **33034**

6904 Concrete Naming Certificate for InverseFunctionModulus$^{\uparrow}$ **33037**

6905 Concrete Naming Certificate for PseudoArc$^{\uparrow}$ **33040**

6906 Concrete Naming Certificate for RegularCauchyOscillationCriterion$^{\uparrow}$ **33043**

690A Concrete Naming Certificate for BergeMaximum↑ **33046**

690B Concrete Naming Certificate for CompactFunctionAlgebra↑ **33048**

690C Concrete Naming Certificate for DyadicNetTree↑ **33050**

690D Concrete Naming Certificate for DedekindCutBoundary↑ **33053**

690E Concrete Naming Certificate for PoincareBendixsonTrap↑ **33056**

690F Concrete Naming Certificate for CompactContinuousImageDiameter↑ **33059**

690G Concrete Naming Certificate for LocatedRationalInterval↑ **33062**

690H Concrete Naming Certificate for ContractiveSequence↑ **33065**

690I Concrete Naming Certificate for DyadicSignedDigitStream↑ **33068**

690J Concrete Naming Certificate for CofinalCauchyFilter↑ **33071**

690K Concrete Naming Certificate for MetaCICDecidableFragment↑ **33073**

690L Concrete Naming Certificate for RealCompletionAdjunction↑ **33076**

690M Concrete Naming Certificate for BishopRegularInterval↑ **33079**

690N Concrete Naming Certificate for HyperbolicObserverPhaseRoute↑ **33082**

690O Concrete Naming Certificate for FrobeniusCoin↑ **33085**

690P Concrete Naming Certificate for MetaCICSubjectReductionDischargeMatrix↑ **33088**

690Q Concrete Naming Certificate for VanDerCorputDifferenceEstimate↑ **33091**

690R Concrete Naming Certificate for VanDerCorputSum↑ **33094**

690S Concrete Naming Certificate for LocatedRealModulusExtractor↑ **33097**

690T Concrete Naming Certificate for FiniteWindowRegSeqRatRefinement↑ **33100**

690U Concrete Naming Certificate for FiniteCoverBarModulus↑ **33103**

690V Concrete Naming Certificate for CofinalMetricCompletion↑ **33106**

690W Concrete Naming Certificate for RegularConditionalDistribution↑ **33109**

690X Concrete Naming Certificate for BishopCauchyLocator↑ **33112**

690Y Concrete Naming Certificate for PipelineDiscoveryClassifier↑ **33115**

690Z Concrete Naming Certificate for DecimalEndpointAmbiguity↑ **33118**

6933 Concrete Naming Certificate for RHZeroWindow↑ **33121**

6934 Concrete Naming Certificate for RegularCauchyAlgebraClosure↑ **33124**

6935 Concrete Naming Certificate for ErdosSzekeresMonotoneSubsequence↑ **33127**

6936 Concrete Naming Certificate for KolmogorovZeroOneLaw↑ **33130**

6937 Concrete Naming Certificate for CauchyRealsWithoutQuotient↑ **33133**

6938 Concrete Naming Certificate for CauchyBishopBridge↑ **33136**

6939 Concrete Naming Certificate for ClosureCapstoneRealization↑ **33139**

6940 Concrete Naming Certificate for ClosedBoundedIntervalFiniteSubcover↑ **33142**

6941 Concrete Naming Certificate for RegularCauchyTailWindowSelector↑ **33145**

6942 Concrete Naming Certificate for BusemannConvexMetric↑ **33148**

6943 Concrete Naming Certificate for KripkeSemantics↑ **33151**

6944 Concrete Naming Certificate for ArcLengthParameter↑ **33154**

6945 Concrete Naming Certificate for TestFunction↑ **33157**

6946 Concrete Naming Certificate for CompactIntervalFanBridge↑ **33160**

6947 Concrete Naming Certificate for BoundedNormalEqualityFrontier↑ **33163**

6948 Concrete Naming Certificate for MerayCauchyReal↑ **33166**

6949 Concrete Naming Certificate for RealCompletionBudgetRouter↑ **33169**

6950 Concrete Naming Certificate for DirichletUniformTest↑ **33172**

6951 Concrete Naming Certificate for HermiteHadamard↑ **33175**

6952 Concrete Naming Certificate for FiniteRankOperator↑ **33178**

6953 Concrete Naming Certificate for CauchyNetConvergenceCriterion↑ **33181**

6954 Concrete Naming Certificate for CauchyCompletionPullback↑ **33184**

6955 Concrete Naming Certificate for ResidualSubstitutionCompatibilityWindow↑ **33187**

6956 Concrete Naming Certificate for BourbakiCauchyFilter↑ **33190**

6957 Concrete Naming Certificate for AzumaHoeffdingFiniteMartingale↑ **33193**

6958 Concrete Naming Certificate for BishopIntervalNesting↑ **33196**

695.9 A Concrete Naming Certificate for CompactFiniteLowerBound↑ **33198**

696.0 A Concrete Naming Certificate for RegularCauchyReindexing↑ **33200**

696.1 A Concrete Naming Certificate for BanachInverseMapping↑ **33203**

696.2 A Concrete Naming Certificate for LocatedIntervalOrder↑ **33206**

696.3 A Concrete Naming Certificate for ChoquetCapacity↑ **33209**

696.4 A Concrete Naming Certificate for RegularCauchyConstant↑ **33212**

696.5 Bishop Real Window NameCert Construction **33215**

696.6 A Concrete Naming Certificate for FractionalIdeal↑ **33218**

696.7 A Concrete Naming Certificate for FiniteBrunnMinkowskiGrid↑ **33221**

696.8 A Concrete Naming Certificate for CauchyProductSeries↑ **33224**

696.9 A Concrete Naming Certificate for MoserIteration↑ **33227**

697.0 A Concrete Naming Certificate for SeparatedCompletionIdempotence↑ **33230**

697.1 A Concrete Naming Certificate for RoundedCut↑ **33233**

697.2 A Concrete Naming Certificate for DyadicCompletionEmbedding↑ **33236**

697.3 A Concrete Naming Certificate for AbsoluteConvergenceCriterion↑ **33239**

697.4 A Concrete Naming Certificate for HarnackInequality↑ **33242**

697.5 Tucker Lemma NameCert Construction **33245**

697.6 Regular Cauchy Located Limit Seal NameCert Construction **33248**

697.7 A Concrete Naming Certificate for ConstructiveNestedRealInterval↑ **33251**

697.8 A Concrete Naming Certificate for RegularCauchyCompletionOperator↑ **33254**

697.9 A Concrete Naming Certificate for CauchyRepletion↑ **33257**

698.0 A Concrete Naming Certificate for HyperbolicPhaseHorocycle↑ **33259**

698.1 A Concrete Naming Certificate for CauchyModulusDominanceCriterion↑ **33261**

698.2 A Concrete Naming Certificate for BirkhoffErgodicFiniteAverage↑ **33264**

698.3 A Concrete Naming Certificate for NestedClosedIntervalSelector↑ **33267**

698.4 A Concrete Naming Certificate for SlaterCondition↑ **33270**

6985 A Concrete Naming Certificate for ClassifierShiftLedgerResidue↑ **33273**

6986 A Concrete Naming Certificate for BishopIntervalRefinementTree↑ **33276**

6987 A Concrete Naming Certificate for LocatedCauchyProductGuard↑ **33279**

6988 A Concrete Naming Certificate for PrincipalCauchyEmbedding↑ **33282**

6989 A Concrete Naming Certificate for RegularCauchyRepletionBoundary↑ **33284**

6990 A Concrete Naming Certificate for MetaCICSubstitutionSpineRealizer↑ **33287**

6991 A Concrete Naming Certificate for TaitReducibility↑ **33289**

6992 A Concrete Naming Certificate for DirectedCofinalSubset↑ **33292**

6993 A Concrete Naming Certificate for BezoutIdentity↑ **33295**

6994 A Concrete Naming Certificate for SchurComplement↑ **33298**

6995 A Concrete Naming Certificate for HyperbolicGeodesicMidpoint↑ **33301**

6996 A Concrete Naming Certificate for RegularCauchyDiagonalModulus↑ **33304**

6997 A Concrete Naming Certificate for FiniteCoverStarRefinementModulus↑ **33307**

6998 A Concrete Naming Certificate for BishopContinuityModulus↑ **33310**

6999 A Concrete Naming Certificate for CantorDiagonalCompactness↑ **33313**

7000 A Concrete Naming Certificate for HolderSpace↑ **33316**

7001 A Concrete Naming Certificate for TranscendentalGapLedger↑ **33318**

7002 RealNameBallFilter NameCert Construction **33321**

7003 A Concrete Naming Certificate for BishopRegularCompletionFunctor↑ **33324**

7004 A Concrete Naming Certificate for CompactUniformCauchyDiagonal↑ **33327**

7005 Bishop Sequence Limit NameCert Construction **33330**

7006 A Concrete Naming Certificate for RegulatedOscillation↑ **33333**

7007 A Concrete Naming Certificate for CauchySubbasisTailFactorization↑ **33336**

7008 A Concrete Naming Certificate for SignChangeBisection↑ **33339**

7009 A Concrete Naming Certificate for EffectiveRegularRealExtensionality↑ **33341**

7010 A Concrete Naming Certificate for OrderedVectorSpace↑ **33344**

7011 Concrete Naming Certificate for BishopRegularRealRepresentation↑ **33346**

7012 Concrete Naming Certificate for LayeredRelationGateQuestion↑ **33349**

7013 Concrete Naming Certificate for RegularCauchyWindowRefinement↑ **33352**

7014 Concrete Naming Certificate for FiniteNetMeshRefinement↑ **33355**

7015 Concrete Naming Certificate for MetricRegularCauchyCompletion↑ **33358**

7016 Concrete Naming Certificate for RegularSequenceEquivalence↑ **33362**

7017 Concrete Naming Certificate for GaussQuadrature↑ **33365**

7018 Concrete Naming Certificate for RegularCauchyLocatedCut↑ **33368**

7019 Concrete Naming Certificate for HyperbolicBoundaryCompactification↑ **33371**

7020 Concrete Naming Certificate for RegularCauchySubsequencePrinciple↑ **33374**

7021 Concrete Naming Certificate for CompactUniformModulusEnvelope↑ **33377**

7022 Concrete Naming Certificate for CauchyNameUniformity↑ **33380**

7023 Concrete Naming Certificate for CofiniteFilter↑ **33383**

7024 Concrete Naming Certificate for SelfCompilationFeedback↑ **33385**

7025 Concrete Naming Certificate for LittlewoodPaley↑ **33388**

7026 Concrete Naming Certificate for LocatedRealConvergenceCriterion↑ **33391**

7027 Concrete Naming Certificate for ArchimedeanFloor↑ **33394**

7028 Concrete Naming Certificate for DiniUniformCauchy↑ **33397**

7029 Concrete Naming Certificate for CauchyLocatedModulusSpectrum↑ **33400**

7030 Concrete Naming Certificate for NagataDimension↑ **33402**

7031 Concrete Naming Certificate for HyperbolicPhaseLedger↑ **33405**

7032 Concrete Naming Certificate for ClosedIntervalUniformLimit↑ **33407**

7033 Concrete Naming Certificate for LusternikSchnirelmannCategory↑ **33410**

7034 Concrete Naming Certificate for HellySelectionCompactVariation↑ **33413**

7035 Concrete Naming Certificate for AuditGateBoundaryRefusal↑ **33416**

7036 Concrete Naming Certificate for DeGiorgiIteration↑ **33419**

7037 Concrete Naming Certificate for ContourReparameterizationLedger↑ **33422**

7038 Concrete Naming Certificate for BoundedStoppingTime↑ **33425**

7039 Concrete Naming Certificate for DenseSubspaceCompletion↑ **33428**

7040 Concrete Naming Certificate for CauchyTowerCollapse↑ **33431**

7041 Concrete Naming Certificate for QuasisymmetricMap↑ **33434**

7042 Concrete Naming Certificate for MetaCICParallelDiamondWitness↑ **33437**

7043 Concrete Naming Certificate for MetrizableUniformity↑ **33440**

7044 Concrete Naming Certificate for CousinFiniteCover↑ **33443**

7045 Concrete Naming Certificate for PowerSeries↑ **33446**

7046 Concrete Naming Certificate for MengerCompactum↑ **33448**

7047 Concrete Naming Certificate for CauchyFixedPointOrbitSeal↑ **33451**

7048 Concrete Naming Certificate for FiniteProductUniformModulus↑ **33454**

7049 Concrete Naming Certificate for MetricSpaceTotalBounded↑ **33457**

7050 Concrete Naming Certificate for LocatedFiniteChoiceSequence↑ **33460**

7051 Concrete Naming Certificate for RationalBallBasis↑ **33462**

7052 Concrete Naming Certificate for CatZeroProjection↑ **33465**

7053 Concrete Naming Certificate for EuclideanDomain↑ **33468**

7054 Concrete Naming Certificate for ConstantRealStream↑ **33471**

7055 Concrete Naming Certificate for MonotoneRegularCauchyLimit↑ **33474**

7056 Concrete Naming Certificate for HyperbolicShearCoordinate↑ **33477**

7057 Concrete Naming Certificate for DoubleArrowSpace↑ **33480**

7058 Concrete Naming Certificate for CountableBaseMetric↑ **33483**

7059 Concrete Naming Certificate for LocatedDyadicApproximation↑ **33486**

7060 Concrete Naming Certificate for FiniteGameMinimax↑ **33489**

7061 Concrete Naming Certificate for TheoremDnaComplexityScoring↑ **33492**

7062 Concrete Naming Certificate for BaireOnePointContinuity↑ **33495**

7063 Concrete Naming Certificate for DyadicDifferentiationBasis↑ **33499**

7064 Concrete Naming Certificate for DyadicDenominatorClearance↑ **33502**

7065 Concrete Naming Certificate for FiniteDifferenceTable↑ **33505**

7066 Concrete Naming Certificate for RegularCauchyWindowSelector↑ **33508**

7067 Concrete Naming Certificate for FΣSet↑ **33511**

7068 Concrete Naming Certificate for BirkhoffJamesOrthogonality↑ **33514**

7069 Concrete Naming Certificate for Arc↑ **33517**

7070 Concrete Naming Certificate for RationalCauchyTailBound↑ **33520**

7071 Concrete Naming Certificate for FinitePrefixBar↑ **33523**

7072 Concrete Naming Certificate for AxisCarryCriticalPair↑ **33526**

7073 Concrete Naming Certificate for DualityMapping↑ **33529**

7074 Concrete Naming Certificate for HadamardMatrix↑ **33531**

7075 Concrete Naming Certificate for FiniteTotalBoundedNet↑ **33534**

7076 Concrete Naming Certificate for RealInverseModulus↑ **33537**

7077 Concrete Naming Certificate for RegulatedSequence↑ **33540**

7078 Concrete Naming Certificate for SupplyTaxonomy↑ **33543**

7079 Concrete Naming Certificate for HadamardDeterminantInequality↑ **33546**

7080 Concrete Naming Certificate for HalfOpenRealInterval↑ **33549**

7081 Concrete Naming Certificate for StreamReindexing↑ **33553**

7082 Concrete Naming Certificate for DedekindCompleteOrderedField↑ **33556**

7083 Concrete Naming Certificate for RegularRealLineAtlas↑ **33559**

7084 Concrete Naming Certificate for RealCompletionUnit↑ **33562**

7085 Concrete Naming Certificate for RegularCauchyArchimedeanBridge↑ **33564**

7086 Concrete Naming Certificate for AbelUniformConvergence↑ **33567**

7087 Concrete Naming Certificate for FiniteRegularCauchySubsequence↑ **33570**

7088 Concrete Naming Certificate for MetaCICTypedResidualBudget↑ **33573**

7089 A Concrete Naming Certificate for RademacherFiniteDifference↑ **33576**

7090 A Concrete Naming Certificate for RademacherTheorem↑ **33579**

7091 A Concrete Naming Certificate for CauchyRealAddition↑ **33582**

7092 A Concrete Naming Certificate for EulerianGraph↑ **33585**

7093 A Concrete Naming Certificate for VitaliCoveringFiniteGauge↑ **33588**

7094 A Concrete Naming Certificate for BishopRealArithmetic↑ **33591**

7095 A Concrete Naming Certificate for WeakStarFiniteWindow↑ **33594**

7096 A Concrete Naming Certificate for BetaSubstitutionPreservationFrontier↑ **33597**

7097 A Concrete Naming Certificate for MatroidDual↑ **33600**

7098 A Concrete Naming Certificate for CertifiedEventFlow↑ **33603**

7099 A Concrete Naming Certificate for FareyMediantInterval↑ **33606**

7100 A Concrete Naming Certificate for PorousSet↑ **33609**

7101 A Concrete Naming Certificate for BoundedRealLimsup↑ **33612**

7102 A Concrete Naming Certificate for CompactSubspaceModulusRestriction↑ **33615**

7103 A Concrete Naming Certificate for CauchyProductSequence↑ **33618**

7104 A Concrete Naming Certificate for CauchyRealMultiplication↑ **33621**

7105 A Concrete Naming Certificate for RegularCauchyCompletionFunctoriality↑ **33624**

7106 A Concrete Naming Certificate for CompactUniformCauchyFamily↑ **33627**

7107 A Concrete Naming Certificate for OneOverNZeroLimit↑ **33630**

7108 A Concrete Naming Certificate for HyperbolicPhaseSpace↑ **33633**

7109 A Concrete Naming Certificate for UniformLocalConnectedness↑ **33636**

7110 A Concrete Naming Certificate for GelfandTransform↑ **33639**

7111 A Concrete Naming Certificate for UniformLipschitzModulus↑ **33642**

7112 A Concrete Naming Certificate for HyperbolicObserverPhase↑ **33645**

7113 A Concrete Naming Certificate for ClosedIntervalCauchyField↑ **33648**

7114 A Concrete Naming Certificate for ProofRelevantNormalJoin↑ **33651**

711.5 A Concrete Naming Certificate for McShaneExtensionReal↑ **33655**

711.6 A Concrete Naming Certificate for BanachSaks↑ **33659**

711.7 A Concrete Naming Certificate for CauchyNullModulus↑ **33662**

711.8 A Concrete Naming Certificate for DyadicCauchyStream↑ **33665**

711.9 A Concrete Naming Certificate for TailWindowFilter↑ **33668**

712.0 A Concrete Naming Certificate for RegularCauchyUniquenessWindow↑ **33671**

712.1 A Concrete Naming Certificate for HyperbolicPhaseGeodesicMidpoint↑ **33674**

712.2 A Concrete Naming Certificate for BishopModulusCompletion↑ **33677**

712.3 A Concrete Naming Certificate for BishopDyadicLocatedReal↑ **33680**

712.4 A Concrete Naming Certificate for CauchyBicategory↑ **33683**

712.5 A Concrete Naming Certificate for CantorNormalForm↑ **33686**

712.6 A Concrete Naming Certificate for DistinctionCertificate↑ **33689**

712.7 A Concrete Naming Certificate for TaoMetastableCauchy↑ **33692**

712.8 A Concrete Naming Certificate for GaugeIntegralCriterion↑ **33695**

712.9 Bishop interval modulus NameCert construction **33697**

713.0 A Concrete Naming Certificate for RegularCauchyPairModulus↑ **33700**

713.1 A Concrete Naming Certificate for DyadicCofinalGrid↑ **33703**

713.2 A Concrete Naming Certificate for RegularCauchySynchronizationModulus↑ **33706**

713.3 A Concrete Naming Certificate for ContinuousCompactRiemannIntegrability↑ **33709**

713.4 A Concrete Naming Certificate for HyperbolicPhaseTransportBudget↑ **33712**

713.5 A Concrete Naming Certificate for RegularCauchyWindowExtraction↑ **33715**

713.6 A Concrete Naming Certificate for ChoiceSequenceBar↑ **33718**

713.7 A Concrete Naming Certificate for CompactOpenEquicontinuousAscoli↑ **33720**

713.8 A Concrete Naming Certificate for HyperbolicPhaseTorsion↑ **33723**

713.9 A Concrete Naming Certificate for MirskyTheorem↑ **33726**

714.0 A Concrete Naming Certificate for CompactIntervalDyadicGrid↑ **33729**

7141 Concrete Naming Certificate for RadonPartition↑ **33732**

7142 Concrete Naming Certificate for CompactIntervalChoiceSequence↑ **33734**

7143 Concrete Naming Certificate for ClosedMetricGraph↑ **33737**

7144 Concrete Naming Certificate for LocatedApartnessField↑ **33739**

7145 Concrete Naming Certificate for CauchyWindowComparisonPrinciple↑ **33742**

7146 Concrete Naming Certificate for WhitneyCoveringFinite↑ **33745**

7147 Concrete Naming Certificate for LebesgueNumberModulus↑ **33748**

7148 Concrete Naming Certificate for HyperbolicWhitneyShadowCover↑ **33751**

7149 Concrete Naming Certificate for ConditionallyPositiveKernel↑ **33754**

7150 Concrete Naming Certificate for ContractionMappingPicard↑ **33757**

7151 Concrete Naming Certificate for MittagLefflerBishop↑ **33760**

7152 Concrete Naming Certificate for MittagLeffler↑ **33763**

7153 Concrete Naming Certificate for Hausdorffization↑ **33766**

7154 Concrete Naming Certificate for FriendlyMetricCompletion↑ **33779**

7155 Concrete Naming Certificate for LocatedIntermediateValueModulus↑ **33782**

7156 Concrete Naming Certificate for CantorSeriesExpansion↑ **33785**

7157 Concrete Naming Certificate for KuratowskiFiniteSet↑ **33788**

7158 Concrete Naming Certificate for KernelCategoryFunctor↑ **33791**

7159 Concrete Naming Certificate for ConstructiveCantorBaire↑ **33794**

7160 Concrete Naming Certificate for CauchyCriterionSeries↑ **33797**

7161 Concrete Naming Certificate for DarbouxOscillation↑ **33800**

7162 Concrete Naming Certificate for FiniteFanCompactness↑ **33802**

7163 Concrete Naming Certificate for UniformConvergenceTheorem↑ **33804**

7164 Concrete Naming Certificate for HamiltonianFlow↑ **33807**

7165 Concrete Naming Certificate for LocatedIntervalFanBarrier↑ **33810**

7166 Concrete Naming Certificate for CompactEmbedding↑ **33814**

7167 A Concrete Naming Certificate for FiniteProofTraceSieve↑ **33817**

7168 A Concrete Naming Certificate for BishopIntervalBisection↑ **33820**

7169 A Concrete Naming Certificate for RegulatedStepFunction↑ **33823**

7170 A Concrete Naming Certificate for HyperbolicShadowBoundary↑ **33826**

7171 A Concrete Naming Certificate for RationalNullSequence↑ **33829**

7172 A Concrete Naming Certificate for CauchyOrderReflection↑ **33832**

7173 A Concrete Naming Certificate for CompactOscillationSelector↑ **33835**

7174 A Concrete Naming Certificate for CompactOscillationVariation↑ **33838**

7175 A Concrete Naming Certificate for FiniteOscillationVariation↑ **33840**

7176 A Concrete Naming Certificate for CompactModulusOscillationRouter↑ **33842**

7177 A Concrete Naming Certificate for RegularCauchyCompletionObject↑ **33845**

7178 A Concrete Naming Certificate for YoungTableau↑ **33848**

7179 A Concrete Naming Certificate for CauchyTailSpace↑ **33850**

7180 CandidateNormalizationDischargeUp NameCert Construction **33853**

7181 A Concrete Naming Certificate for ExactCouple↑ **33856**

7182 A Concrete Naming Certificate for SeminormFamily↑ **33859**

7183 A Concrete Naming Certificate for BihariLaSalle↑ **33862**

7184 A Concrete Naming Certificate for RegularRealTailSelector↑ **33865**

7185 A Concrete Naming Certificate for RegularCauchyDoubleLimitInterchange↑ **33868**

7186 A Concrete Naming Certificate for FiniteKernelClosureGate↑ **33871**

7187 A Concrete Naming Certificate for AbelDirichletTest↑ **33874**

7188 A Concrete Naming Certificate for ResidualSet↑ **33877**

7189 A Concrete Naming Certificate for CompactUniformLocalToGlobalModulus↑ **33880**

7190 A Concrete Naming Certificate for LocatedCauchyLimit↑ **33883**

7191 A Concrete Naming Certificate for EisensteinCriterion↑ **33886**

7192 A Concrete Naming Certificate for CauchyRealModulus↑ **33889**

7193 Concrete Naming Certificate for ArgumentPrinciple↑ **33891**

7194 Concrete Naming Certificate for UniformModulusErrorBudget↑ **33894**

7195 Concrete Naming Certificate for RegularSequenceCompletion↑ **33897**

7196 Concrete Naming Certificate for DyadicNewtonSqrt↑ **33899**

7197 Concrete Naming Certificate for CantorBendixsonRank↑ **33902**

7198 Concrete Naming Certificate for ConstructiveCauchyTailOperator↑ **33905**

7199 Concrete Naming Certificate for HyperbolicConvexDisk↑ **33908**

7200 Concrete Naming Certificate for BishopCompactIntervalModulus↑ **33911**

7201 Concrete Naming Certificate for TailWindowModulusSynchronization↑ **33914**

7202 Concrete Naming Certificate for AbsoluteContinuityModulusReal↑ **33917**

7203 Concrete Naming Certificate for DyadicIntervalEnclosure↑ **33920**

7204 Concrete Naming Certificate for DenseGDelta↑ **33923**

7205 Concrete Naming Certificate for DyadicRoundingProjection↑ **33925**

7206 Concrete Naming Certificate for BishopRealRegularization↑ **33928**

7207 Concrete Naming Certificate for ConstructiveUniformBoundedImage↑ **33931**

7208 Concrete Naming Certificate for CompactUniformEquicontinuityEnvelope↑ **33934**

7209 Concrete Naming Certificate for CofinalReindexing↑ **33937**

7210 Concrete Naming Certificate for BishopCompletionModulusSelection↑ **33940**

7211 Concrete Naming Certificate for ConstructiveRealNormalization↑ **33943**

7212 Concrete Naming Certificate for LawvereMetric↑ **33945**

7213 Concrete Naming Certificate for RegularCauchyModulusComparison↑ **33948**

7214 Concrete Naming Certificate for ClopenAlgebra↑ **33951**

7215 Concrete Naming Certificate for DoobMaximalInequality↑ **33954**

7216 Concrete Naming Certificate for BishopCompletionModulusStability↑ **33957**

7217 Concrete Naming Certificate for RiemannRemovableDiscontinuity↑ **33960**

7218 Concrete Naming Certificate for RegularCauchyModulusDomination↑ **33963**

7219 A Concrete Naming Certificate for MetaCICResidualDiamondWitness↑ **33966**

7220 A Concrete Naming Certificate for MetaCICCriticalOverlapEnvelope↑ **33969**

7221 A Concrete Naming Certificate for SuslinLine↑ **33972**

7222 A Concrete Naming Certificate for DyadicBracketing↑ **33975**

7223 A Concrete Naming Certificate for CompactOscillationEnvelope↑ **33978**

7224 A Concrete Naming Certificate for BishopIntervalNestingMetricCompletion↑ **33981**

7225 A Concrete Naming Certificate for CauchyCriterionForFunctions↑ **33984**

7226 A Concrete Naming Certificate for WhitneyApproximation↑ **33987**

7227 A Concrete Naming Certificate for CompactCauchySubsequence↑ **33990**

7228 A Concrete Naming Certificate for CompactUniformCauchyTailRouter↑ **33992**

7229 A Concrete Naming Certificate for RegularCauchyLimitTheorem↑ **33995**

7230 A Concrete Naming Certificate for RemovableDiscontinuity↑ **33997**

7231 A Concrete Naming Certificate for HyperbolicBoundaryDistortionBudget↑ **34000**

7232 A Concrete Naming Certificate for HyperbolicObserverHorizon↑ **34003**

7233 A Concrete Naming Certificate for CompactNetSelection↑ **34006**

7234 A Concrete Naming Certificate for FiniteSubcoverExtraction↑ **34009**

7235 A Concrete Naming Certificate for MazurLemma↑ **34012**

7236 A Concrete Naming Certificate for SobolevTraceCompactWindow↑ **34015**

7237 A Concrete Naming Certificate for DarstSelection↑ **34019**

7238 A Concrete Naming Certificate for CompactUniformModulusNormalForm↑ **34022**

7239 A Concrete Naming Certificate for RationalDenseEmbedding↑ **34025**

7240 A Concrete Naming Certificate for ProofEntropyLedger↑ **34028**

7241 A Concrete Naming Certificate for CompactIntervalIntermediateValue↑ **34031**

7242 A Concrete Naming Certificate for MetaCICNormalizationSubstitutionBoundary↑ **34033**

7243 A Concrete Naming Certificate for BishopLocatedCantorSpace↑ **34036**

7244 A Concrete Naming Certificate for PoincareHorocycleTransport↑ **34039**

7245 Concrete Naming Certificate for LocallyFiniteCover↑ **34042**

7246 Concrete Naming Certificate for KannanContractionFixedPoint↑ **34046**

7247 Concrete Naming Certificate for FiniteOpenCover↑ **34049**

7248 Concrete Naming Certificate for DyadicBarycentricSubdivision↑ **34052**

7249 Concrete Naming Certificate for CantorSpaceBaireCategory↑ **34055**

7250 Concrete Naming Certificate for BishopHeineBorelInterval↑ **34058**

7251 Concrete Naming Certificate for FiniteShiftAverage↑ **34060**

7252 Concrete Naming Certificate for DarbouxOscillationCriterion↑ **34063**

7253 Concrete Naming Certificate for FinitePrefixStream↑ **34066**

7254 Concrete Naming Certificate for EffectiveCauchyRealCompletion↑ **34073**

7255 Concrete Naming Certificate for CompactModulusSelection↑ **34076**

7256 Concrete Naming Certificate for LocatedDedekindCut↑ **34079**

7257 Shape Theory NameCert Construction **34082**

7258 Concrete Naming Certificate for MetaCICResidualJoinGrid↑ **34085**

7259 Concrete Naming Certificate for RegularCauchyModulusSpace↑ **34088**

7260 Concrete Naming Certificate for CompactUniformSourceTargetTriangleLedger↑ **34091**

7261 Concrete Naming Certificate for TightProbabilityMeasure↑ **34094**

7262 Concrete Naming Certificate for CauchyLimitUniquenessModulus↑ **34097**

7263 Concrete Naming Certificate for PoincareDiskIsometry↑ **34100**

7264 Concrete Naming Certificate for NewtonCotes↑ **34102**

7265 Concrete Naming Certificate for SimpsonRule↑ **34105**

7266 Concrete Naming Certificate for FiniteJacobian↑ **34108**

7267 Concrete Naming Certificate for BishopRegularRealLocator↑ **34111**

7268 Concrete Naming Certificate for ConstructiveLocatedReal↑ **34115**

7269 Concrete Naming Certificate for BarrelledSpace↑ **34118**

7270 Concrete Naming Certificate for RealApartnessLocator↑ **34121**

7271 Concrete Naming Certificate for KoopmanOperator↑ **34124**

7272 Concrete Naming Certificate for LocatedUniformity↑ **34127**

7273 Concrete Naming Certificate for DyadicIntervalBisection↑ **34130**

7274 Concrete Naming Certificate for BeltramiKleinDisk↑ **34132**

7275 Concrete Naming Certificate for MetricRetraction↑ **34135**

7276 Concrete Naming Certificate for DyadicCauchyApartness↑ **34138**

7277 Concrete Naming Certificate for LocatedCompletionUniqueness↑ **34141**

7278 Concrete Naming Certificate for UniformCompletionUnitModulus↑ **34144**

7279 Concrete Naming Certificate for DiscreteGronwall↑ **34147**

7280 Concrete Naming Certificate for PseudotheoryRefutationCountermodel↑ **34150**

7281 Concrete Naming Certificate for RegularCauchyBishopLimit↑ **34153**

7282 Concrete Naming Certificate for FuchsianGroupAction↑ **34155**

7283 Concrete Naming Certificate for ProofNet↑ **34157**

7284 Concrete Naming Certificate for LocatedCauchyIntervalCompletion↑ **34160**

7285 Concrete Naming Certificate for PrincipalBundle↑ **34163**

7286 Concrete Naming Certificate for DyadicCauchyStaircase↑ **34165**

7287 Concrete Naming Certificate for SobolevDual↑ **34168**

7288 Concrete Naming Certificate for CauchyFilterCompleteness↑ **34170**

7289 Concrete Naming Certificate for BishopCompletionFilter↑ **34173**

7290 Concrete Naming Certificate for NestedIntervalPrinciple↑ **34176**

7291 Concrete Naming Certificate for MetaCICResidualCriticalPair↑ **34179**

7292 Concrete Naming Certificate for BishopCompactIntervalCauchyCompleteness↑ **34183**

7293 Concrete Naming Certificate for UrysohnRationalMetric↑ **34187**

7294 Concrete Naming Certificate for DirichletAbelSummability↑ **34190**

7295 Concrete Naming Certificate for ArzelaAscoliFiniteSubfamily↑ **34193**

7296 Concrete Naming Certificate for WallmanCompactification↑ **34196**

7297 A Concrete Naming Certificate for FiniteElementStiffnessMatrix↑ 34199

7298 Baire open-dense intersection NameCert construction 34202

7299 A Concrete Naming Certificate for SequentialLimitPoint↑ 34205

7300 A Concrete Naming Certificate for PeetreKFunctional↑ 34208

7301 A Concrete Naming Certificate for RegulatedRiemannIntegrability↑ 34212

7302 A Concrete Naming Certificate for FinitePowerBoundedOperator↑ 34215

7303 A Concrete Naming Certificate for RealTree↑ 34217

7304 A Concrete Naming Certificate for MartingaleStoppingConvergence↑ 34220

7305 A Concrete Naming Certificate for ObserverExperienceTrace↑ 34223

7306 Regular Cauchy Finite Splice NameCert Construction 34226

7307 A Concrete Naming Certificate for SequentialFilterBridge↑ 34229

7308 A Concrete Naming Certificate for ConstructiveCompactUniformFan↑ 34232

7309 A Concrete Naming Certificate for BishopLocatedRealModulus↑ 34235

7310 Bishop Regular Cauchy Reindexing NameCert Construction 34238

7311 A Concrete Naming Certificate for BalancedTruncation↑ 34241

7312 A Concrete Naming Certificate for ObserverModulusClock↑ 34244

7313 RiemannPartitionUp NameCert construction 34247

7314 A Concrete Naming Certificate for CountableProductMetric↑ 34249

7315 A Concrete Naming Certificate for BauerSimplex↑ 34252

7316 A Concrete Naming Certificate for TopologistSineCurve↑ 34255

7317 A Concrete Naming Certificate for BishopCauchyModulusWitness↑ 34258

7318 A Concrete Naming Certificate for ClaimAuditBoundary↑ 34261

7319 A Concrete Naming Certificate for RegularSequenceRegularizer↑ 34264

7320 A Concrete Naming Certificate for NakayamaLemma↑ 34267

7320.0 Carrier 34267

7321 UniversalCoefficientUp NameCert Construction 34270

7322 A Concrete Naming Certificate for ConstructiveChoiceSequenceSpace↑ 34273

7323 A Concrete Naming Certificate for FiniteProductCover↑ **34276**

7324 A Concrete Naming Certificate for RegularCauchyTailLimit↑ **34279**

7325 A Concrete Naming Certificate for FanTheoremBoundary↑ **34282**

7326 A Concrete Naming Certificate for SobolevExtensionOperator↑ **34285**

7327 A Concrete Naming Certificate for LocatedUniformExtension↑ **34288**

7328 A Concrete Naming Certificate for RegularCauchyModulusNormalization↑ **34291**

7329 A Concrete Naming Certificate for PolynomialResultant↑ **34295**

7330 A Concrete Naming Certificate for CompactUniformContinuityFiniteModulus↑ **34298**

7331 A Concrete Naming Certificate for DyadicMidpointInterval↑ **34301**

7332 A Concrete Naming Certificate for CantorCylinderCompactness↑ **34304**

7333 Rational squeeze NameCert construction **34307**

7334 A Concrete Naming Certificate for IsaacsEquation↑ **34311**

7335 A Concrete Naming Certificate for BlackBoxCertificateDeficit↑ **34314**

7336 A Concrete Naming Certificate for Sheafification↑ **34317**

7337 A Concrete Naming Certificate for MartingaleConvergenceFiniteUpcrossing↑ **34338**

7338 A Concrete Naming Certificate for MartingaleConvergenceWitness↑ **34341**

7339 A Concrete Naming Certificate for MetaCICEtaExpansionAudit↑ **34344**

7340 A Concrete Naming Certificate for MackeyTopology↑ **34347**

7341 A Concrete Naming Certificate for CylinderSet↑ **34350**

7342 A Concrete Naming Certificate for IteratedFunctionSystem↑ **34353**

7343 A Concrete Naming Certificate for RealUpperLowerApproximation↑ **34356**

7344 A Concrete Naming Certificate for EquicontinuousCompactImage↑ **34359**

7345 A Concrete Naming Certificate for LocallyConnectedContinuum↑ **34362**

7346 A Concrete Naming Certificate for MetaCICLocalConfluenceFrontier↑ **34365**

7347 A Concrete Naming Certificate for SmallInductiveDimension↑ **34368**

7348 A Concrete Naming Certificate for FiniteMetricProjection↑ **34371**

7349 A Concrete Naming Certificate for MilnorFibration↑ **34374**

7350 A Concrete Naming Certificate for BoundedConvergence↑ **34377**

7351 A Concrete Naming Certificate for SubordinateModulusUniformHandoff↑ **34380**

7352 A Concrete Naming Certificate for RegularCauchyEmbedding↑ **34383**

7353 A Concrete Naming Certificate for DunfordPettisFiniteWeakCompactness↑ **34386**

7354 A Concrete Naming Certificate for RegularCauchyReciprocalModulus↑ **34389**

7355 A Concrete Naming Certificate for EgyptianFraction↑ **34392**

7356 A Concrete Naming Certificate for CompactMetricEquicontinuity↑ **34395**

7357 A Concrete Naming Certificate for MilmanConverse↑ **34397**

7358 A Concrete Naming Certificate for HellingerDistance↑ **34400**

7359 A Concrete Naming Certificate for TreeAutomaton↑ **34403**

7360 A Concrete Naming Certificate for PositiveDyadicRadius↑ **34406**

7361 A Concrete Naming Certificate for RegularLanguageContinuation↑ **34409**

7362 A Concrete Naming Certificate for RegularCauchyKleisliExtension↑ **34411**

7363 A Concrete Naming Certificate for DiscreteLyapunov↑ **34414**

7364 A Concrete Naming Certificate for StrongLawFiniteWindow↑ **34417**

7365 A Concrete Naming Certificate for SnowflakeMetric↑ **34420**

7366 A Concrete Naming Certificate for DistanceToCompact↑ **34423**

7367 A Concrete Naming Certificate for RegularCauchyReflectionUnit↑ **34426**

7368 A Concrete Naming Certificate for RegularCauchyReflectionCounit↑ **34428**

7369 A Concrete Naming Certificate for CompactUniformSequenceModulus↑ **34431**

7370 A Concrete Naming Certificate for FiniteChainDecomposition↑ **34433**

7371 A Concrete Naming Certificate for LargeInductiveDimension↑ **34436**

7372 A Concrete Naming Certificate for HeineBorelFiniteCover↑ **34439**

7373 A Concrete Naming Certificate for BoundedContinuousImage↑ **34442**

7374 A Concrete Naming Certificate for WellFoundedTreeRank↑ **34445**

7375 A Concrete Naming Certificate for BanachFixedPointConstructive$^{\uparrow}$ **34447**

7376 A Concrete Naming Certificate for FixedPointIndex$^{\uparrow}$ **34450**

7377 A Concrete Naming Certificate for StratifiedSpace$^{\uparrow}$ **34453**

7378 A Concrete Naming Certificate for MetaCICResidualJoinBudget$^{\uparrow}$ **34456**

7379 A Concrete Naming Certificate for RegularCauchySubsequenceDiagonal$^{\uparrow}$ **34459**

7380 A Concrete Naming Certificate for KoksmaHlawkaFiniteError$^{\uparrow}$ **34462**

7381 A Concrete Naming Certificate for VisionCoverageIndex$^{\uparrow}$ **34465**

7382 A Concrete Naming Certificate for HawaiianEarring$^{\uparrow}$ **34468**

7383 A Concrete Naming Certificate for RiemannDarbouxCriterionFinite$^{\uparrow}$ **34471**

7384 A Concrete Naming Certificate for CauchyModulusThinning$^{\uparrow}$ **34474**

7385 A Concrete Naming Certificate for CousinPartition$^{\uparrow}$ **34477**

7386 A Concrete Naming Certificate for BishopIntervalOrder$^{\uparrow}$ **34480**

7387 A Concrete Naming Certificate for RegularCauchySumTailEstimate$^{\uparrow}$ **34483**

7388 A Concrete Naming Certificate for RuleOneTenHaltingWindow$^{\uparrow}$ **34486**

7389 A Concrete Naming Certificate for KTopology$^{\uparrow}$ **34489**

7390 A Concrete Naming Certificate for ProofTraceCompactness$^{\uparrow}$ **34491**

7391 A Concrete Naming Certificate for BishopLocatedIntervalRefinement$^{\uparrow}$ **34494**

7392 A Concrete Naming Certificate for CauchySeparableCompletion$^{\uparrow}$ **34497**

7393 A Concrete Naming Certificate for RegularCauchySandwichLimit$^{\uparrow}$ **34500**

7394 A Concrete Naming Certificate for MiddleThirdCantorSet$^{\uparrow}$ **34503**

7395 A Concrete Naming Certificate for FrechetFilterCompletion$^{\uparrow}$ **34506**

7396 A Concrete Naming Certificate for RealOrderCompletionBoundary$^{\uparrow}$ **34509**

7397 A Concrete Naming Certificate for RapidlyCauchySequence$^{\uparrow}$ **34512**

7398 A Concrete Naming Certificate for BishopRealRegularCompletion$^{\uparrow}$ **34515**

7399 A Concrete Naming Certificate for FiniteObservationSheaf$^{\uparrow}$ **34518**

7400 A Concrete Naming Certificate for DyadicArchimedeanBridge$^{\uparrow}$ **34520**

7401 Concrete Naming Certificate for CompactUniformFunctionSpace↑ **34523**

7402 Concrete Naming Certificate for RegularCauchyModulusJoin↑ **34526**

7403 Concrete Naming Certificate for CompactLocatedMaximumPrinciple↑ **34529**

7404 Concrete Naming Certificate for FiniteCauchyModulusComparison↑ **34532**

7405 Concrete Naming Certificate for MetaCICResidualSubstitutionComparison↑ **34535**

7406 Concrete Naming Certificate for Kirchberger↑ **34538**

7407 Concrete Naming Certificate for MetaCICResidualDiamondBudget↑ **34541**

7408 Concrete Naming Certificate for RegularCauchyQuotientTailEstimate↑ **34544**

7409 Concrete Naming Certificate for MonotoneConvergenceTheorem↑ **34547**

7410 Concrete Naming Certificate for RationalFloorBound↑ **34551**

7411 Concrete Naming Certificate for OscillationSeminorm↑ **34554**

7412 Concrete Naming Certificate for AbelMeanBoundary↑ **34557**

7413 Concrete Naming Certificate for HenstockLemma↑ **34560**

7414 Concrete Naming Certificate for PositiveDiscoveryDebtLedger↑ **34563**

7415 Concrete Naming Certificate for BoundedVariationRiemannSum↑ **34566**

7416 Concrete Naming Certificate for CompactUniformOscillationNerve↑ **34569**

7417 Concrete Naming Certificate for OstrowskiInequality↑ **34572**

7418 Concrete Naming Certificate for PackingDimension↑ **34576**

7419 Concrete Naming Certificate for FrechetCauchyNet↑ **34579**

7420 Concrete Naming Certificate for HausdorffMetricHyperspace↑ **34582**

7421 Concrete Naming Certificate for CofinalSubsequenceFilter↑ **34585**

7422 Concrete Naming Certificate for MetaCICResidualCandidateSelector↑ **34588**

7423 Concrete Naming Certificate for PhaseGeodesicCrossRatio↑ **34591**

7424 Concrete Naming Certificate for StoneSpace↑ **34594**

7425 Concrete Naming Certificate for CauchySubsequenceLimitSeal↑ **34597**

7426 Concrete Naming Certificate for FrechetCauchyFilter↑ **34600**

7427 Concrete Naming Certificate for ModulusCauchyReal↑ **34603**

7428 Concrete Naming Certificate for BornologicalSpace↑ **34606**

7429 Concrete Naming Certificate for CompactOscillationMesh↑ **34609**

7430 Concrete Naming Certificate for DetachableBar↑ **34612**

7431 Concrete Naming Certificate for CourantFischerMinimax↑ **34616**

7432 Concrete Naming Certificate for RealCauchyInterleavingSeal↑ **34618**

7433 Concrete Naming Certificate for KuratowskiClosureComplement↑ **34621**

7434 Concrete Naming Certificate for FinitePhaseCylinder↑ **34624**

7435 Concrete Naming Certificate for HausdorffDimensionFiniteCover↑ **34626**

7436 Concrete Naming Certificate for RegSeqRatTailCanonical↑ **34629**

7437 Concrete Naming Certificate for RaabeTest↑ **34632**

7438 Concrete Naming Certificate for CauchyModulusFan↑ **34635**

7439 Concrete Naming Certificate for QuasiBanach↑ **34638**

7440 Concrete Naming Certificate for CofinalityFilter↑ **34641**

7441 Concrete Naming Certificate for ExtendedMetricSpace↑ **34644**

7442 Concrete Naming Certificate for BishopLocatedIntervalNest↑ **34647**

7443 Concrete Naming Certificate for LocalizedCompactEnclosure↑ **34650**

7444 Concrete Naming Certificate for UniformlyDiscreteMetric↑ **34653**

7445 Concrete Naming Certificate for CompactUniformDyadicRadiusLadder↑ **34655**

7446 Concrete Naming Certificate for CompactChoiceSequence↑ **34658**

7447 Concrete Naming Certificate for HadamardBoundary↑ **34661**

7448 Concrete Naming Certificate for BishopCauchyModulusLimit↑ **34664**

7449 Concrete Naming Certificate for TaggedPartition↑ **34667**

7450 Concrete Naming Certificate for LocatedUniformCauchyLimit↑ **34670**

7451 Concrete Naming Certificate for RealLocatedInfimum↑ **34673**

7452 Concrete Naming Certificate for ProbeBundleCoverFold↑ **34676**

7453 Concrete Naming Certificate for RegularCauchyVariationBound↑ **34679**

7454 Concrete Naming Certificate for DetachableSubset↑ **34682**

7455 Concrete Naming Certificate for BishopModulusConvergence↑ **34685**

7456 Concrete Naming Certificate for CompactUniformModulusCertificate↑ **34688**

7457 Concrete Naming Certificate for BourbakiCompletionFilter↑ **34691**

7458 Concrete Naming Certificate for PositiveSeriesCauchyProduct↑ **34694**

7459 Concrete Naming Certificate for UpperDiniDerivative↑ **34697**

7460 Concrete Naming Certificate for KreinMilmanCompactConvex↑ **34700**

7461 Concrete Naming Certificate for BishopSeparatedFiniteWindowCompletion↑ **34703**

7462 Concrete Naming Certificate for CantorIntervalTrap↑ **34706**

7463 Concrete Naming Certificate for InscriptionEventNovelty↑ **34709**

7464 Concrete Naming Certificate for PositiveRealNormalization↑ **34711**

7465 Concrete Naming Certificate for DyadicClosedIntervalFiniteNet↑ **34715**

7466 Concrete Naming Certificate for KreinRutman↑ **34718**

7467 Concrete Naming Certificate for UpperSemicontinuity↑ **34721**

7468 Concrete Naming Certificate for ParallelReduction↑ **34724**

7469 Concrete Naming Certificate for ClosedInductionTrace↑ **34726**

7470 Concrete Naming Certificate for MetaCICBetaRedexSpine↑ **34729**

7471 Concrete Naming Certificate for BishopLocatedCompletionEmbedding↑ **34732**

7472 Concrete Naming Certificate for UniformContinuityGraphModulus↑ **34734**

7473 Concrete Naming Certificate for BishopLocatedIntervalRefinementMap↑ **34736**

7474 Concrete Naming Certificate for CompactUniformTriangleEstimateLedger↑ **34738**

7475 Concrete Naming Certificate for PlanarGraphMinor↑ **34740**

7476 Concrete Naming Certificate for LebesgueCoveringLemma↑ **34743**

7477 Concrete Naming Certificate for SchurTest↑ **34746**

7477.1 Carrier and finite Schur packet 34746

7477.2 NameCert obligations 34747

7478 A Concrete Naming Certificate for LocatedDedekindRegularCauchyBridge↑ **34749**

7479 A Concrete Naming Certificate for FiniteIntervalPartition↑ **34752**

7480 A Concrete Naming Certificate for DyadicIntervalSubdivision↑ **34755**

7481 A Concrete Naming Certificate for FiniteOpenCoverNerve↑ **34757**

7482 A Concrete Naming Certificate for MinimizingMovement↑ **34760**

7483 A Concrete Naming Certificate for RegularSequenceModulus↑ **34763**

7484 A Concrete Naming Certificate for VisionCoverageLedger↑ **34766**

7485 A Concrete Naming Certificate for LHopitalRule↑ **34769**

7486 A Concrete Naming Certificate for VoronoiCellLedger↑ **34772**

7487 A Concrete Naming Certificate for HorosphereShearLedger↑ **34775**

7488 A Concrete Naming Certificate for BishopCauchyFusion↑ **34778**

7489 A Concrete Naming Certificate for RealArchimedeanWindow↑ **34781**

7490 A Concrete Naming Certificate for DyadicMidpointChain↑ **34784**

7491 A Concrete Naming Certificate for BishopLocatedDyadicInterval↑ **34787**

7492 A Concrete Naming Certificate for BourbakiFilterLimit↑ **34790**

7493 A Concrete Naming Certificate for HyperbolicVisualCrossRatioCocycle↑ **34793**

7494 A Concrete Naming Certificate for CoordinateFunctional↑ **34796**

7495 A Concrete Naming Certificate for CauchyProductCompletionFusion↑ **34799**

7496 A Concrete Naming Certificate for DyadicMeshGauge↑ **34802**

7497 A Concrete Naming Certificate for SplitInterval↑ **34805**

7498 A Concrete Naming Certificate for StreamInterleaving↑ **34808**

7499 A Concrete Naming Certificate for PoincareDiskAutomorphismDerivative↑ **34812**

7500 A Concrete Naming Certificate for MetaCICResidualJoinGauge↑ **34815**

7501 A Concrete Naming Certificate for CantorScheme↑ **34818**

7502 A Concrete Naming Certificate for LipschitzSpace↑ **34821**

7503 A Concrete Naming Certificate for DyadicShadowCone↑ **34823**

7504 A Concrete Naming Certificate for DyadicRealDensity↑ **34826**

7505 A Concrete Naming Certificate for SupportFunction↑ **34829**

7506 A Concrete Naming Certificate for ClosedFiberUniformConvexity↑ **34832**

7507 RegularCauchyDistributiveLaw NameCert Construction **34835**

7508 A Concrete Naming Certificate for PixleyRoySpace↑ **34838**

7509 A Concrete Naming Certificate for RegulatedFunctionFiniteJump↑ **34841**

7510 A Concrete Naming Certificate for BishopArchimedeanCompletion↑ **34844**

7511 A Concrete Naming Certificate for RegularCauchyInverseBoundary↑ **34847**

7512 A Concrete Naming Certificate for DarbouxIntegrable↑ **34850**

7513 A Concrete Naming Certificate for StreamNameTailWindowRouter↑ **34853**

7514 Regular Real Located Modulus NameCert Construction **34855**

7515 Hyperbolic Visual Metric Modulus NameCert Construction **34858**

7516 A Concrete Naming Certificate for TropicalSemiring↑ **34861**

7517 A Concrete Naming Certificate for TotallyBoundedCompletionCriterion↑ **34864**

7518 A Concrete Naming Certificate for BishopRegularCauchyFilter↑ **34867**

7519 A Concrete Naming Certificate for MetaCICTypedSubstitutionWindow↑ **34870**

7520 WeylEquidistribution NameCert Construction **34873**

7521 A Concrete Naming Certificate for TotalBoundedCompletionCompactness↑ **34876**

7522 A Concrete Naming Certificate for SchwartzDistribution↑ **34879**

7523 A Concrete Naming Certificate for JungTheorem↑ **34882**

7524 A Concrete Naming Certificate for MetricDenseEmbedding↑ **34885**

7525 A Concrete Naming Certificate for MartingaleQuadraticVariation↑ **34888**

7526 A Concrete Naming Certificate for PseudocompactSpace↑ **34891**

7526.1 Pseudocompact Carrier 34891

7526.2 Certificate Obligations 34892

7527 A Concrete Naming Certificate for CauchyCompletionIntersection↑ **34894**

7528 A Concrete Naming Certificate for CauchyBornologicalUniformity↑ **34897**

7529 A Concrete Naming Certificate for RegularCauchyApartnessTransfer$^{\uparrow}$ **34900**

7530 A Concrete Naming Certificate for ShadowingLemma$^{\uparrow}$ **34903**

7531 A Concrete Naming Certificate for KatetovExtension$^{\uparrow}$ **34906**

7532 A Concrete Naming Certificate for CofinalTailMesh$^{\uparrow}$ **34909**

7533 A Concrete Naming Certificate for BlackBoxFiberWitness$^{\uparrow}$ **34912**

7534 A Concrete Naming Certificate for HyperbolicBusemannFunction$^{\uparrow}$ **34915**

7535 A Concrete Naming Certificate for UniformCauchyFilterDiagonal$^{\uparrow}$ **34918**

7536 A Concrete Naming Certificate for CauchyBornologyDuality$^{\uparrow}$ **34921**

7537 Birkhoff Distributive Lattice NameCert Construction **34924**

7538 A Concrete Naming Certificate for RoundedRealDomain$^{\uparrow}$ **34927**

7539 A Concrete Naming Certificate for MenelausTheorem$^{\uparrow}$ **34930**

7540 A Concrete Naming Certificate for MetricMidpointConvexity$^{\uparrow}$ **34933**

7541 A Concrete Naming Certificate for CompactUniformFiniteCoverFold$^{\uparrow}$ **34936**

7542 A Concrete Naming Certificate for IntervalImageConnected$^{\uparrow}$ **34940**

7543 A Concrete Naming Certificate for BeurlingLax$^{\uparrow}$ **34944**

7544 A Concrete Naming Certificate for CauchyTailStability$^{\uparrow}$ **34947**

7545 A Concrete Naming Certificate for CriticalStripOscillation$^{\uparrow}$ **34950**

7546 A Concrete Naming Certificate for ClassifierFormationTrace$^{\uparrow}$ **34953**

7547 A Concrete Naming Certificate for RegularCauchyApartnessCotransitivity$^{\uparrow}$ **34956**

7548 A Concrete Naming Certificate for CompactIntervalVariationGrid$^{\uparrow}$ **34959**

7549 A Concrete Naming Certificate for EuclideanAlgorithm$^{\uparrow}$ **34961**

7550 A Concrete Naming Certificate for LocatedIntervalModulusExtraction$^{\uparrow}$ **34964**

7551 A Concrete Naming Certificate for RegularCauchyStolzTransform$^{\uparrow}$ **34967**

7552 A Concrete Naming Certificate for BusemannProjection$^{\uparrow}$ **34970**

7553 A Concrete Naming Certificate for CompactModulusMonotoneConvergence$^{\uparrow}$ **34973**

7554 A Concrete Naming Certificate for CauchyCompletionCategory$^{\uparrow}$ **34976**

7555 A Concrete Naming Certificate for BishopIntervalNestingModulus↑ **34979**

7556 A Concrete Naming Certificate for ChoiceFreeRealSeal↑ **34982**

7557 A Concrete Naming Certificate for FiniteCoverUniformModulusEnvelope↑ **34984**

7558 A Concrete Naming Certificate for DarbouxStieltjesSum↑ **34986**

7559 A Concrete Naming Certificate for CauchyCompletionMonadJoin↑ **34989**

7560 A Concrete Naming Certificate for BoundaryPairing↑ **34992**

7561 A Concrete Naming Certificate for AlexiewiczNorm↑ **34995**

7562 A Concrete Naming Certificate for RealCauchyWindow↑ **34998**

7563 A Concrete Naming Certificate for CauchyConvergenceRate↑ **35001**

7564 A Concrete Naming Certificate for TailWindowFilterRefinement↑ **35004**

7565 A Concrete Naming Certificate for LindebergCondition↑ **35007**

7566 A Concrete Naming Certificate for SeparatedUniformCompletion↑ **35010**

7567 A Concrete Naming Certificate for ClosedMetricSubspace↑ **35013**

7568 A Concrete Naming Certificate for DirichletUniformModulus↑ **35016**

7569 A Concrete Naming Certificate for KuratowskiEmbedding↑ **35019**

7570 A Concrete Naming Certificate for KuratowskiEmbeddingDistanceProfile↑ **35022**

7571 A Concrete Naming Certificate for SequentialCompactnessModulus↑ **35025**

7572 A Concrete Naming Certificate for HausdorffLocatedUniformReflection↑ **35028**

7573 A Concrete Naming Certificate for AffineFilterHull↑ **35031**

7574 A Concrete Naming Certificate for CauchyApproximationSpace↑ **35034**

7575 A Concrete Naming Certificate for FreeTopologicalGroup↑ **35037**

7576 A Concrete Naming Certificate for ArzelaAscoliFiniteNet↑ **35040**

7577 A Concrete Naming Certificate for BishopCauchyModulusCalibration↑ **35043**

7578 A Concrete Naming Certificate for HilbertCubeCompactness↑ **35046**

7579 A Concrete Naming Certificate for CompletionModulusLifting↑ **35049**

7580 A Concrete Naming Certificate for CauchyRealsCompletion↑ **35052**

7581 Concrete Naming Certificate for DependentChoiceBoundary$^{\uparrow}$ **35055**

7582 Concrete Naming Certificate for RealityConstrainedSelectorBoundary$^{\uparrow}$ **35058**

7583 Concrete Naming Certificate for KolmogorovWidth$^{\uparrow}$ **35061**

7584 Concrete Naming Certificate for FiniteApproximationKernel$^{\uparrow}$ **35064**

7585 Concrete Naming Certificate for DecidableBar$^{\uparrow}$ **35066**

7586 Concrete Naming Certificate for CauchyPairLocator$^{\uparrow}$ **35069**

7587 Concrete Naming Certificate for ConstructiveRealModulus$^{\uparrow}$ **35072**

7588 Concrete Naming Certificate for BishopIntervalCantor$^{\uparrow}$ **35074**

7589 Concrete Naming Certificate for MobiusDiskCrossRatio$^{\uparrow}$ **35077**

7590 Concrete Naming Certificate for FrostmanFiniteMeasure$^{\uparrow}$ **35080**

7591 Concrete Naming Certificate for BishopCauchyCompletenessCriterion$^{\uparrow}$ **35083**

7592 Concrete Naming Certificate for FixedBoundaryHyperbolicShadow$^{\uparrow}$ **35086**

7593 Concrete Naming Certificate for CauchyTailExtraction$^{\uparrow}$ **35089**

7594 Concrete Naming Certificate for BishopNestedClosedSetPrinciple$^{\uparrow}$ **35092**

7595 Concrete Naming Certificate for CartanAngularBoundaryInvariant$^{\uparrow}$ **35095**

7596 Concrete Naming Certificate for FiniteProductCompletion$^{\uparrow}$ **35098**

7597 Concrete Naming Certificate for FiniteCoverNerve$^{\uparrow}$ **35101**

7598 Concrete Naming Certificate for ContinuityModulusLattice$^{\uparrow}$ **35105**

7599 Concrete Naming Certificate for PointedSet$^{\uparrow}$ **35108**

7600 Concrete Naming Certificate for FrontdoorCriterion$^{\uparrow}$ **35111**

7601 Concrete Naming Certificate for BishopRealEqualityModulus$^{\uparrow}$ **35114**

7602 Concrete Naming Certificate for RieszBasis$^{\uparrow}$ **35117**

7603 Concrete Naming Certificate for CousinGaugeCover$^{\uparrow}$ **35120**

7604 Concrete Naming Certificate for CompactOscillationModulus$^{\uparrow}$ **35123**

7605 Concrete Naming Certificate for PietschDomination$^{\uparrow}$ **35126**

7606 Concrete Naming Certificate for RegularRectangularModulus$^{\uparrow}$ **35128**

7607 A Concrete Naming Certificate for EquivalenceRelation$^{\uparrow}$ **35131**

7608 A Concrete Naming Certificate for CriticalPairResidualStream$^{\uparrow}$ **35133**

7609 A Concrete Naming Certificate for NonArchimedeanMetric$^{\uparrow}$ **35136**

7610 A Concrete Naming Certificate for RegularCauchyModulusTransfer$^{\uparrow}$ **35139**

7611 A Concrete Naming Certificate for LocatedSequenceInfimum$^{\uparrow}$ **35142**

7612 A Concrete Naming Certificate for DiagonalLimitUniquenessWitness$^{\uparrow}$ **35145**

7613 A Concrete Naming Certificate for DarbouxDerivativeProperty$^{\uparrow}$ **35148**

7614 A Concrete Naming Certificate for LipschitzFreeSpace$^{\uparrow}$ **35151**

7615 A Concrete Naming Certificate for ContractionRealModulus$^{\uparrow}$ **35154**

7616 A Concrete Naming Certificate for LocatedMetricBall$^{\uparrow}$ **35157**

7617 A Concrete Naming Certificate for CandidateFrontierQueue$^{\uparrow}$ **35160**

7618 A Concrete Naming Certificate for BishopRealSequence$^{\uparrow}$ **35163**

7619 A Concrete Naming Certificate for JamesCompactnessCriterion$^{\uparrow}$ **35166**

7620 A Concrete Naming Certificate for ConstructiveRealClosedInterval$^{\uparrow}$ **35169**

7621 A Concrete Naming Certificate for HadamardDirectionalDerivative$^{\uparrow}$ **35172**

7622 A Concrete Naming Certificate for MoreauEnvelope$^{\uparrow}$ **35175**

7622.1 Moreau Envelope Carrier 35175

7622.2 Moreau Envelope Certificate 35176

7623 A Concrete Naming Certificate for MirrorDescent$^{\uparrow}$ **35179**

7624 A Concrete Naming Certificate for HyperbolicHoroballShadowLedger$^{\uparrow}$ **35182**

7625 A Concrete Naming Certificate for EilenbergZilber$^{\uparrow}$ **35185**

7626 A Concrete Naming Certificate for UniformCompletionUniqueness$^{\uparrow}$ **35188**

7627 A Concrete Naming Certificate for FiniteRealPicardIteration$^{\uparrow}$ **35191**

7628 A Concrete Naming Certificate for WeilUniformity$^{\uparrow}$ **35194**

7629 A Concrete Naming Certificate for CapProduct$^{\uparrow}$ **35197**

7630 A Concrete Naming Certificate for WindowSixDecorationDefect$^{\uparrow}$ **35199**

7631 A Concrete Naming Certificate for ConstructiveContractionModulus$^{\uparrow}$ **35202**

7632 Concrete Naming Certificate for MetaCICParallelDiamondScheduler↑ **35205**

7633 Concrete Naming Certificate for SierpinskiGasket↑ **35208**

7634 Concrete Naming Certificate for DoldKan↑ **35211**

7635 Concrete Naming Certificate for StoneCechRemainder↑ **35214**

7636 Concrete Naming Certificate for NewtonInterpolation↑ **35217**

7637 Concrete Naming Certificate for BishopRegularCompletionMonad↑ **35220**

7638 Concrete Naming Certificate for CompactHyperspaceFiniteNet↑ **35223**

7639 Concrete Naming Certificate for WhitneyStratificationFiniteAtlas↑ **35226**

7640 Concrete Naming Certificate for HaarIntegralFiniteGroup↑ **35229**

7641 Concrete Naming Certificate for CauchyRealNegation↑ **35232**

7642 Concrete Naming Certificate for ToeplitzSilverman↑ **35234**

7643 Concrete Naming Certificate for KrullIntersection↑ **35237**

7644 Concrete Naming Certificate for ChoiceFreeDyadicIntervalSelection↑ **35239**

7645 Concrete Naming Certificate for ClosedAtDecisionRefusalBoundary↑ **35242**

7646 Concrete Naming Certificate for RadstromEmbedding↑ **35245**

7647 Concrete Naming Certificate for CompactApartnessLocatedSeparation↑ **35248**

7648 Concrete Naming Certificate for FiniteOscillationEnvelope↑ **35251**

7649 Concrete Naming Certificate for CauchyReblocking↑ **35254**

7650 Concrete Naming Certificate for RegulatedDarbouxSum↑ **35257**

7651 Concrete Naming Certificate for ClosedIntervalCompactness↑ **35260**

7652 Concrete Naming Certificate for RealCompletenessModulusLedger↑ **35263**

7653 Concrete Naming Certificate for CompactIntervalInverseModulus↑ **35266**

7654 Concrete Naming Certificate for GelfandMazur↑ **35269**

7655 Concrete Naming Certificate for IntervalArzelaAscoli↑ **35272**

7656 Concrete Naming Certificate for DunfordSchwartzMeanErgodicFinite↑ **35275**

7657 Concrete Naming Certificate for RegularCauchyBilinearProduct↑ **35278**

7658 Concrete Naming Certificate for SeparatingHyperplane↑ **35281**

7659 Concrete Naming Certificate for HyperbolicIdealQuadrilateral↑ **35284**

7660 Concrete Naming Certificate for BishopRegularRealLimit↑ **35287**

7661 Concrete Naming Certificate for LayeredRelationCertificate↑ **35290**

7662 Concrete Naming Certificate for DeterministicPushdownAutomaton↑ **35293**

7663 Concrete Naming Certificate for HamiltonJacobi↑ **35296**

7664 Concrete Naming Certificate for RegularCauchyFilterUniversalProperty↑ **35299**

7665 Concrete Naming Certificate for PositiveSeriesComparison↑ **35302**

7666 Concrete Naming Certificate for HyperbolicDirichletDomain↑ **35305**

7667 Concrete Naming Certificate for LocatedCauchySubnet↑ **35308**

7668 Concrete Naming Certificate for MedianDepthSealingCorner↑ **35310**

7669 Concrete Naming Certificate for EulerConstant↑ **35312**

7670 Concrete Naming Certificate for AlternatingHarmonic↑ **35315**

7671 Concrete Naming Certificate for UnconditionalBasis↑ **35318**

7672 Concrete Naming Certificate for ForceLikeRelation↑ **35321**

7673 Concrete Naming Certificate for FiniteCardinal↑ **35324**

7674 Concrete Naming Certificate for FiniteLimitCone↑ **35327**

7675 Concrete Naming Certificate for CauchyFilterBoundedMonotone↑ **35330**

7676 Concrete Naming Certificate for TychonoffProductWindow↑ **35333**

7677 Concrete Naming Certificate for CauchyProductTheorem↑ **35335**

7678 Concrete Naming Certificate for KockZoberleinCauchyCompletion↑ **35338**

7679 Concrete Naming Certificate for RiemannStieltjesSum↑ **35341**

7680 Concrete Naming Certificate for SequentialMetricSpace↑ **35344**

7681 Concrete Naming Certificate for EulerianNumber↑ **35347**

7682 Concrete Naming Certificate for MetaCICClosureSubstitutionFrontier↑ **35350**

7683 Concrete Naming Certificate for VitaliCoveringLemma↑ **35353**

7684 Concrete Naming Certificate for RiemannIntegrabilityDiscontinuity† **35356**

7685 Concrete Naming Certificate for FiniteHausdorffMetric† **35360**

7686 Concrete Naming Certificate for AsymptoticCone† **35363**

7687 Concrete Naming Certificate for RiemannTaggedMesh† **35366**

7688 Concrete Naming Certificate for PhysicalTowerRealization† **35369**

7689 Concrete Naming Certificate for BoundedRegSeqRatMap† **35372**

7690 Concrete Naming Certificate for SchoenfliesFiniteJordanWindow† **35375**

7691 Concrete Naming Certificate for RationalIntervalTree† **35378**

7692 Concrete Naming Certificate for TakagiFunction† **35381**

7693 Concrete Naming Certificate for KirchhoffMatrixTree† **35384**

7694 Concrete Naming Certificate for PastingLemma† **35387**

7695 Concrete Naming Certificate for CayleyTreePruferCode† **35390**

7696 Concrete Naming Certificate for LadyzhenskayaInequality† **35393**

7697 Concrete Naming Certificate for SchurFunction† **35396**

7698 Concrete Naming Certificate for HardyLittlewoodMaximalFinite† **35399**

7699 Concrete Naming Certificate for FarEndBoundaryCospan† **35402**

7700 Concrete Naming Certificate for HahnSeries† **35405**

7701 Concrete Naming Certificate for DyadicModulusRegularity† **35408**

7702 Concrete Naming Certificate for IsbellMrowkaPsiSpace† **35410**

7703 Concrete Naming Certificate for BaireClassOneRealFunction† **35413**

7704 Concrete Naming Certificate for StationaryDistribution† **35416**

7705 Concrete Naming Certificate for ReflectionGapAudit† **35419**

7706 Concrete Naming Certificate for DyadicSpread† **35422**

7707 Concrete Naming Certificate for HoarePowerdomain† **35425**

7708 Concrete Naming Certificate for GromovCompactness† **35428**

7709 Concrete Naming Certificate for MetaCICResidualCriticalPairBudget† **35430**

7710 A Concrete Naming Certificate for BarTheorem↑ **35432**

7711 A Concrete Naming Certificate for CauchyCompletionNaturality↑ **35435**

7712 A Concrete Naming Certificate for FrontierCompileHardProblem↑ **35437**

7713 A Concrete Naming Certificate for TuttePolynomial↑ **35440**

7714 A Concrete Naming Certificate for MetaCICSubstitutionDiamondFrontier↑ **35443**

7715 A Concrete Naming Certificate for BareissFractionFreeElimination↑ **35446**

7716 A Concrete Naming Certificate for ProofMiningMetastability↑ **35449**

7717 A Concrete Naming Certificate for RadonTheorem↑ **35452**

7718 A Concrete Naming Certificate for FiniteBorelCantelli↑ **35454**

7719 A Concrete Naming Certificate for Detectability↑ **35457**

7720 A Concrete Naming Certificate for PicardIterationCauchy↑ **35460**

7721 A Concrete Naming Certificate for RegularCauchyHarmonicEstimate↑ **35463**

7722 A Concrete Naming Certificate for CubicSpline↑ **35466**

7723 A Concrete Naming Certificate for RegularCauchyOscillationModulusExtraction↑ **35469**

7724 A Concrete Naming Certificate for RegularCauchyNameNormalForm↑ **35472**

7725 A Concrete Naming Certificate for HyperbolicVisualAngle↑ **35475**

7726 A Concrete Naming Certificate for CauchyRealRegularization↑ **35478**

7727 A Concrete Naming Certificate for TietzeExtensionFiniteStep↑ **35480**

7728 Mankiewicz Extension NameCert Construction **35483**

7729 A Concrete Naming Certificate for NeronModel↑ **35486**

7729.0 Carrier definition 35486

7729.1 The certificate 35487

7730 A Concrete Naming Certificate for CofilteredLimit↑ **35489**

7731 A Concrete Naming Certificate for RegularCauchyMinmaxLattice↑ **35492**

7732 A Concrete Naming Certificate for RegularCauchyBinarySubdivision↑ **35495**

7733 A Concrete Naming Certificate for ContextFreeLanguage↑ **35498**

7734 A Concrete Naming Certificate for RationalCut↑ **35501**

7735 Concrete Naming Certificate for CompactCauchyNet↑ **35504**

7736 Concrete Naming Certificate for MooreMachine↑ **35507**

7737 Concrete Naming Certificate for RealPositiveRadiusReadback↑ **35509**

7738 Concrete Naming Certificate for RegularCauchyCommonModulus↑ **35512**

7739 Concrete Naming Certificate for DyadicTailComparison↑ **35515**

7740 Concrete Naming Certificate for CauchyBicompletion↑ **35518**

7741 Concrete Naming Certificate for CompactOpenCoverBoundary↑ **35521**

7742 Concrete Naming Certificate for FirmNonexpansiveProjection↑ **35524**

7743 Concrete Naming Certificate for RealCauchyDiagonalModulus↑ **35527**

7744 Concrete Naming Certificate for MetaCICExtractionNormalizationSocket↑ **35530**

7745 Concrete Naming Certificate for RealIntervalHalving↑ **35533**

7746 Concrete Naming Certificate for LocatedTotalBoundedness↑ **35536**

7747 Regular Cauchy Uniform Tail NameCert Construction **35539**

7748 Concrete Naming Certificate for BishopPremetric↑ **35542**

7749 Concrete Naming Certificate for ContextSensitiveLanguage↑ **35545**

7750 Concrete Naming Certificate for PoincareMirandaFinite↑ **35548**

7751 Concrete Naming Certificate for UltrafilterBoundary↑ **35551**

7752 Concrete Naming Certificate for FiniteResidualConfluenceWindow↑ **35553**

7753 Concrete Naming Certificate for NeighbourhoodSystem↑ **35556**

7754 Concrete Naming Certificate for CantorIntersectionPrinciple↑ **35559**

7755 Concrete Naming Certificate for BirkhoffContraction↑ **35562**

7756 Concrete Naming Certificate for CauchyModulusCertificate↑ **35565**

7757 Concrete Naming Certificate for InducedCharacter↑ **35568**

7758 Concrete Naming Certificate for CompletionModulusExtractor↑ **35571**

7759 Concrete Naming Certificate for SchauderTychonoff↑ **35574**

7760 Concrete Naming Certificate for BishopLocatedSpace↑ **35576**

7761 A Concrete Naming Certificate for FiniteMetricProjectionSelector↑ **35580**

7762 A Concrete Naming Certificate for CompletionModulusFunctor↑ **35583**

7763 A Concrete Naming Certificate for FiniteProofDagFrontier↑ **35586**

7764 A Concrete Naming Certificate for HyperbolicLimitSet↑ **35589**

7765 A Concrete Naming Certificate for ProperMetricGroupAction↑ **35592**

7766 A Concrete Naming Certificate for BishopLocatedSequence↑ **35595**

7767 A Concrete Naming Certificate for CylinderObject↑ **35598**

7768 A Concrete Naming Certificate for BishopLocatedBanachFixedPoint↑ **35601**

7769 A Concrete Naming Certificate for FiniteCoverMeshChain↑ **35604**

7770 A Concrete Naming Certificate for RegularCauchyTailDifference↑ **35607**

7771 A Concrete Naming Certificate for CaratheodoryExtension↑ **35609**

7772 A Concrete Naming Certificate for StreamRegularCompleteness↑ **35611**

7773 A Concrete Naming Certificate for BishopLocatedDerivative↑ **35614**

7774 A Concrete Naming Certificate for NormedGroup↑ **35617**

7775 A Concrete Naming Certificate for FiniteDirectedNetWindow↑ **35620**

7776 A Concrete Naming Certificate for UlmInvariant↑ **35623**

7777 A Concrete Naming Certificate for RegularCauchyProductTailEstimate↑ **35625**

7778 A Concrete Naming Certificate for SequentialCompletionCriterion↑ **35628**

7779 A Concrete Naming Certificate for LocatedCompactnessModulus↑ **35631**

7780 A Concrete Naming Certificate for UniformCauchyCriterionFunction↑ **35634**

7781 A Concrete Naming Certificate for BishopEquicontinuityDiagonal↑ **35637**

7782 A Concrete Naming Certificate for CalibratedMetricFamily↑ **35640**

7783 Zero-Dimensional Compact Space NameCert Construction **35643**

7784 Dyadic Cantor Window NameCert Construction **35646**

7785 A Concrete Naming Certificate for AscoliLTenCompactFamily↑ **35649**

7786 A Concrete Naming Certificate for WhitneyDecomposition↑ **35651**

7787 Concrete Naming Certificate for CompactFiniteOscillationModulus↑ **35654**

7788 Concrete Naming Certificate for MetricUniformity↑ **35657**

7789 Concrete Naming Certificate for MengerCurve↑ **35660**

7790 Concrete Naming Certificate for ChannelTerminationBoundary↑ **35662**

7791 Concrete Naming Certificate for MetricCoveringDimension↑ **35664**

7792 Concrete Naming Certificate for LowerEnvelopeRegularization↑ **35667**

7793 Concrete Naming Certificate for RegulatedLeftRightLimit↑ **35670**

7794 Concrete Naming Certificate for CauchyCondensationPacket↑ **35673**

7795 Concrete Naming Certificate for NagumoViability↑ **35676**

7796 Concrete Naming Certificate for DecimalExpansionUniqueness↑ **35678**

7797 Concrete Naming Certificate for MetacicResidualSubstitutionSocket↑ **35681**

7798 Concrete Naming Certificate for CotransitiveApartness↑ **35684**

7799 Concrete Naming Certificate for StolzCesaroCauchyTailCompression↑ **35688**

7800 Concrete Naming Certificate for DedekindLocatedInterval↑ **35691**

7801 Concrete Naming Certificate for PoincareDiskHorocycleReturn↑ **35694**

7802 Concrete Naming Certificate for RegularCauchyDominatedTail↑ **35697**

7803 Concrete Naming Certificate for CauchyTailMeet↑ **35700**

7804 Concrete Naming Certificate for FenchelYoungInequality↑ **35703**

7805 Concrete Naming Certificate for HamiltonJacobiBellman↑ **35706**

7806 Concrete Naming Certificate for PoincareHalfPlane↑ **35709**

7807 Concrete Naming Certificate for UniformFilterCompletion↑ **35712**

7808 Concrete Naming Certificate for DushnikMillerDimension↑ **35715**

7809 Concrete Naming Certificate for BishopLocatedCompleteMetric↑ **35718**

7810 Concrete Naming Certificate for CompactModulusEvaluation↑ **35721**

7811 Concrete Naming Certificate for CauchyModulusFusion↑ **35725**

7812 Concrete Naming Certificate for BishopRegularSequenceSplice↑ **35728**

7813 Concrete Naming Certificate for DyadicIntervalSubdivisionTree↑ **35731**

7814 Concrete Naming Certificate for RegularCauchyInduction↑ **35733**

7815 Concrete Naming Certificate for CeaLemma↑ **35736**

7816 Concrete Naming Certificate for FiniteTriggerCube↑ **35739**

7817 Concrete Naming Certificate for EuclideanSpace↑ **35742**

7818 Concrete Naming Certificate for CompactPointwiseBoundedModulus↑ **35745**

7819 Concrete Naming Certificate for FiniteCoverGaugeRadiusChoice↑ **35748**

7820 Concrete Naming Certificate for FrechetUrysohnMetric↑ **35751**

7821 Concrete Naming Certificate for CompactMetricModulusRealizer↑ **35754**

7822 Concrete Naming Certificate for DyadicBracketSchedule↑ **35756**

7823 Concrete Naming Certificate for RiemannCauchyRealName↑ **35759**

7824 Concrete Naming Certificate for SteinerTree↑ **35762**

7825 Concrete Naming Certificate for MetaCICSubjectReductionResidualGap↑ **35765**

7826 Concrete Naming Certificate for MooreOsgoodFiniteWindow↑ **35768**

7827 Concrete Naming Certificate for CompactIntervalUniformContinuity↑ **35771**

7828 Concrete Naming Certificate for AscoliEquicontinuousSubsequence↑ **35776**

7829 Concrete Naming Certificate for BipolarTheorem↑ **35779**

7830 Concrete Naming Certificate for ConnectedCompactInterval↑ **35782**

7831 Concrete Naming Certificate for CanonicalLatentCompletion↑ **35786**

7832 Concrete Naming Certificate for MooreOsgoodLimitInterchange↑ **35789**

7833 Concrete Naming Certificate for BishopFanModulusBoundary↑ **35792**

7834 Concrete Naming Certificate for RealModulusNormalization↑ **35795**

7835 Concrete Naming Certificate for FiniteDyadicMesh↑ **35798**

7836 Concrete Naming Certificate for BoundedPointEvaluationFunctional↑ **35801**

7837 Concrete Naming Certificate for RegulatedPrimitive↑ **35804**

7838 Concrete Naming Certificate for KnuthBendixCompletion↑ **35807**

7839 Concrete Naming Certificate for FiniteModulusTransfer↑ **35810**

7840 Concrete Naming Certificate for QuasiCauchySequence↑ **35813**

7841 Concrete Naming Certificate for FiniteRankDeficiency↑ **35816**

7842 Concrete Naming Certificate for RearrangementInequality↑ **35819**

7843 Concrete Naming Certificate for HyperbolicHoroball↑ **35822**

7844 Concrete Naming Certificate for HamiltonJacobiAction↑ **35825**

7845 Concrete Naming Certificate for EilenbergMooreCategory↑ **35828**

7846 Concrete Naming Certificate for BishopRegularCauchyTailCommonRefinement↑ **35831**

7847 Concrete Naming Certificate for LocallyCompactMetricSpace↑ **35835**

7848 Concrete Naming Certificate for RegularCauchyTailPasting↑ **35838**

7849 Concrete Naming Certificate for LebesgueConstant↑ **35841**

7850 Concrete Naming Certificate for RichmanCompletion↑ **35844**

7851 Concrete Naming Certificate for RegularCauchyPruning↑ **35847**

7852 Concrete Naming Certificate for DyadicMidpointCompletion↑ **35850**

7853 Concrete Naming Certificate for RegularCauchyFilterBasis↑ **35853**

7854 Concrete Naming Certificate for RegularRealGauge↑ **35856**

7855 Concrete Naming Certificate for BuchbergerCriterion↑ **35859**

7856 Concrete Naming Certificate for MetaCICSubstitutionResidualSpine↑ **35862**

7857 Concrete Naming Certificate for BanachLimitFiniteWindow↑ **35865**

7858 Concrete Naming Certificate for RegularCauchyFilterBasisFunctor↑ **35868**

7859 Concrete Naming Certificate for MacLanePentagonCoherence↑ **35871**

7860 Concrete Naming Certificate for CZeroSpace↑ **35874**

7861 Concrete Naming Certificate for CantorIntersectionWindow↑ **35877**

7862 Concrete Naming Certificate for BohrCompactification↑ **35879**

7863 Concrete Naming Certificate for CantorThreadRegularCauchy↑ **35882**

7864 Concrete Naming Certificate for FiniteUniformTailVision↑ **35884**

7865 A Concrete Naming Certificate for CauchyRegularComparisonSeal↑ **35886**

7866 A Concrete Naming Certificate for EulerPhi↑ **35889**

7867 A Concrete Naming Certificate for KhalimskyLine↑ **35892**

7868 A Concrete Naming Certificate for EquicontinuousCompactClosure↑ **35895**

7869 A Concrete Naming Certificate for CompactUniformCoverMembership↑ **35898**

7870 A Concrete Naming Certificate for CurvedSpacetime↑ **35901**

7871 A Concrete Naming Certificate for DyadicNestedIntervalChoiceFree↑ **35904**

7872 A Concrete Naming Certificate for RegularCauchyRateEnvelope↑ **35907**

7873 A Concrete Naming Certificate for HaarMeasure↑ **35910**

7874 A Concrete Naming Certificate for WeilPetersson↑ **35912**

7875 A Concrete Naming Certificate for NormedAlgebra↑ **35915**

7876 A Concrete Naming Certificate for RealRegularSubsequenceCluster↑ **35917**

7877 A Concrete Naming Certificate for RegularCauchyTailMesh↑ **35920**

7878 A Concrete Naming Certificate for DiniUniformModulus↑ **35923**

7879 A Concrete Naming Certificate for MetaCICResidualPairLattice↑ **35926**

7880 A Concrete Naming Certificate for CayleyGraph↑ **35929**

7881 A Concrete Naming Certificate for AlexandrovSpace↑ **35932**

7882 A Concrete Naming Certificate for CompletionDenseExtensionModulus↑ **35935**

7883 A Concrete Naming Certificate for CompactSupportContinuousFunction↑ **35938**

7884 A Concrete Naming Certificate for DyadicIntervalRefinementLadder↑ **35941**

7885 A Concrete Naming Certificate for DifferenceQuotient↑ **35944**

7886 A Concrete Naming Certificate for CauchyCompletionAdjointUnit↑ **35947**

7887 A Concrete Naming Certificate for HyperbolicOperatorCayley↑ **35950**

7888 A Concrete Naming Certificate for CauchyModulusRegularization↑ **35953**

7889 Compact Open Lipschitz Modulus NameCert Construction **35956**

7890 A Concrete Naming Certificate for ErdosSpace↑ **35958**

789A Concrete Naming Certificate for PoincareDiskGeodesicTriangle↑ **35961**

789B Concrete Naming Certificate for PolishZeroDimensional↑ **35964**

789C Concrete Naming Certificate for MetacicResidualDiamondReadback↑ **35967**

789D Concrete Naming Certificate for CompactApartnessModulus↑ **35970**

789E Concrete Naming Certificate for LelekFan↑ **35973**

789F Concrete Naming Certificate for RealArchimedeanFloor↑ **35976**

789G Concrete Naming Certificate for IsoperimetricInequality↑ **35978**

789H Concrete Naming Certificate for EpsilonDeltaSequentialContinuityBridge↑ **35981**

789J Concrete Naming Certificate for FiniteGeodesicConvexity↑ **35984**

790A Concrete Naming Certificate for ClaimAuditWitness↑ **35986**

790B Concrete Naming Certificate for PretopologicalSpace↑ **35989**

790C Concrete Naming Certificate for FiniteNetDiagonalExtraction↑ **35991**

790D Concrete Naming Certificate for RealCompletionDiagonalBudget↑ **35994**

790E Metric Completion Unit NameCert Construction **35997**

790F Compact Uniform Radius Selector NameCert Construction **35999**

790G Located Dyadic Apartness Window NameCert Construction **36001**

790H Concrete Naming Certificate for LipschitzBanach↑ **36003**

790J Concrete Naming Certificate for LocatedCompletionMonad↑ **36006**

790K Concrete Naming Certificate for CauchyCompletionReflectorTriangle↑ **36009**

791A Concrete Naming Certificate for DeckTransformation↑ **36012**

791B Concrete Naming Certificate for DiniDerivativeLowerEnvelope↑ **36014**

791C Concrete Naming Certificate for MoorePlane↑ **36017**

791D Concrete Naming Certificate for EventualEquality↑ **36020**

791E Concrete Naming Certificate for RegularCauchyApartnessLocator↑ **36023**

791F Concrete Naming Certificate for RegularCauchyMajorant↑ **36026**

791G Concrete Naming Certificate for PoincareDiskAreaElement↑ **36029**

7917 Concrete Naming Certificate for PoincareMap↑ **36032**

7918 Concrete Naming Certificate for UpperDiniDerivativeEnvelope↑ **36035**

7919 Concrete Naming Certificate for DyadicMeshLocator↑ **36038**

7920 Concrete Naming Certificate for VerificationCoverageLedger↑ **36041**

7921 Concrete Naming Certificate for SchurProperty↑ **36044**

7922 Concrete Naming Certificate for EventFlowRecognizer↑ **36047**

7923 Concrete Naming Certificate for FiniteWalshTransform↑ **36050**

7924 Concrete Naming Certificate for OsgoodCurve↑ **36053**

7925 Concrete Naming Certificate for BarrBeckMonadicity↑ **36056**

7926 Concrete Naming Certificate for CartanDecomposition↑ **36059**

7927 Concrete Naming Certificate for PoincareDiskBusemannCocycle↑ **36062**

7928 Concrete Naming Certificate for CalderonZygmundOperator↑ **36065**

7929 Concrete Naming Certificate for LebesgueNumberCoverRefinement↑ **36068**

7930 Concrete Naming Certificate for RegularSeqApartnessSeparator↑ **36071**

7931 Concrete Naming Certificate for StoneRepresentationBooleanAlgebra↑ **36074**

7932 Concrete Naming Certificate for CauchyCondensationRemainderGauge↑ **36077**

7933 Concrete Naming Certificate for FiniteDyadicBracketingModulus↑ **36080**

7934 Concrete Naming Certificate for UniformCauchySequenceSpace↑ **36083**

7935 Concrete Naming Certificate for LocatedLimsup↑ **36086**

7936 Concrete Naming Certificate for BishopRealLocatedExtrema↑ **36089**

7937 Concrete Naming Certificate for CompactDerivativeBound↑ **36091**

7938 Concrete Naming Certificate for LocatedSequenceSubspace↑ **36093**

7939 Concrete Naming Certificate for CauchyCondensationFiniteWindow↑ **36095**

7940 Concrete Naming Certificate for LocatedCutGap↑ **36098**

7941 Concrete Naming Certificate for SubsequenceModulusSelection↑ **36102**

7942 Concrete Naming Certificate for DyadicArzelaAscoliModulus↑ **36105**

7943 Concrete Naming Certificate for RieszProduct$^{\uparrow}$ **36108**

7944 Concrete Naming Certificate for RiemannSumMesh$^{\uparrow}$ **36111**

7945 Concrete Naming Certificate for CauchyRealEquivalenceModulus$^{\uparrow}$ **36114**

7946 Concrete Naming Certificate for AtsujiSpace$^{\uparrow}$ **36116**

7947 Concrete Naming Certificate for SequentialBolzanoChoiceFreeSubsequence$^{\uparrow}$ **36119**

7948 Concrete Naming Certificate for DyadicContractionModulus$^{\uparrow}$ **36122**

7949 Concrete Naming Certificate for CauchySubsequenceSelectionModulus$^{\uparrow}$ **36125**

7950 Concrete Naming Certificate for UniformBornology$^{\uparrow}$ **36129**

7951 Concrete Naming Certificate for BrowderFixedPoint$^{\uparrow}$ **36132**

7952 Concrete Naming Certificate for LocatedCutSequence$^{\uparrow}$ **36135**

7953 Concrete Naming Certificate for LongLine$^{\uparrow}$ **36137**

7954 Concrete Naming Certificate for CompactUniformFanModulus$^{\uparrow}$ **36140**

7955 Concrete Naming Certificate for RegularCauchyModulusCofinalRefinement$^{\uparrow}$ **36143**

7956 Concrete Naming Certificate for LocatedRealLowerCut$^{\uparrow}$ **36146**

7957 Concrete Naming Certificate for FiniteOpenCoverSelection$^{\uparrow}$ **36148**

7958 Concrete Naming Certificate for ProfiniteMetricCompletion$^{\uparrow}$ **36151**

7959 Concrete Naming Certificate for ConstructiveBaireCategory$^{\uparrow}$ **36153**

7960 Concrete Naming Certificate for GoursatTheorem$^{\uparrow}$ **36156**

7961 Concrete Naming Certificate for OscillationRank$^{\uparrow}$ **36159**

7962 Concrete Naming Certificate for MetacicResidualFuelRank$^{\uparrow}$ **36162**

7963 Concrete Naming Certificate for DyadicContinuityModulus$^{\uparrow}$ **36165**

7964 Concrete Naming Certificate for RealFunctionSpace$^{\uparrow}$ **36168**

7965 Concrete Naming Certificate for DyadicModulusChoiceFreeRefinement$^{\uparrow}$ **36171**

7966 Concrete Naming Certificate for MetaCICResidualConfluenceFuel$^{\uparrow}$ **36174**

7967 Concrete Naming Certificate for RegSeqRatFunctorialMap$^{\uparrow}$ **36177**

7968 Concrete Naming Certificate for SchottkyGroup$^{\uparrow}$ **36180**

7969 A Concrete Naming Certificate for BromwichInversionWindow↑ **36183**

7970 A Concrete Naming Certificate for AbelPlana↑ **36186**

7971 A Concrete Naming Certificate for GreenTheorem↑ **36189**

7972 A Concrete Naming Certificate for CauchyLimitSeparation↑ **36192**

7973 A Concrete Naming Certificate for KyFanMinimax↑ **36195**

7974 A Concrete Naming Certificate for LawlikeStreamBoundary↑ **36198**

7975 A Concrete Naming Certificate for BishopRealNumber↑ **36201**

7976 A Concrete Naming Certificate for RieszProjection↑ **36203**

7977 A Concrete Naming Certificate for MetaCICResidualCommutationWitness↑ **36206**

7978 A Concrete Naming Certificate for FiniteDimensionalBanach↑ **36209**

7979 A Concrete Naming Certificate for BanachMazurDistance↑ **36212**

7980 A Concrete Naming Certificate for SequentialContinuityReal↑ **36216**

7981 A Concrete Naming Certificate for RasiowaSikorski↑ **36219**

7982 A Concrete Naming Certificate for CompactIntervalChain↑ **36222**

7983 A Concrete Naming Certificate for LocatedRealSpread↑ **36225**

7984 A Concrete Naming Certificate for BishopLocatedCauchyRealComparison↑ **36228**

7985 A Concrete Naming Certificate for Demicontinuity↑ **36231**

7986 A Concrete Naming Certificate for CauchyConvergenceSpace↑ **36234**

7987 A Concrete Naming Certificate for BoundedMonotoneSequence↑ **36237**

7988 A Concrete Naming Certificate for CollectionwiseNormal↑ **36240**

7989 A Concrete Naming Certificate for MetaCICParallelDiamondSelector↑ **36243**

7990 A Concrete Naming Certificate for BarRecursiveModulus↑ **36246**

7991 A Concrete Naming Certificate for PrincipalIdealDomain↑ **36249**

7992 A Concrete Naming Certificate for CauchyModulusSum↑ **36251**

7993 A Concrete Naming Certificate for CauchyExponential↑ **36253**

7994 A Concrete Naming Certificate for UniformEquivalence↑ **36256**

7995 A Concrete Naming Certificate for LocalSelfAudit$^{\uparrow}$ **36259**

7996 A Concrete Naming Certificate for LebesgueCoveringDimensionInduction$^{\uparrow}$ **36262**

7997 A Concrete Naming Certificate for LocatedCauchyCompletionModulus$^{\uparrow}$ **36265**

7998 A Concrete Naming Certificate for HyperbolicHorocycleShadowModulus$^{\uparrow}$ **36268**

7999 A Concrete Naming Certificate for RegularCauchyMidpointConvex$^{\uparrow}$ **36271**

8000 A Concrete Naming Certificate for RegularCauchyConvexCombination$^{\uparrow}$ **36274**

8001 A Concrete Naming Certificate for NerveLemma$^{\uparrow}$ **36278**

8002 A Concrete Naming Certificate for LimsupLiminfGap$^{\uparrow}$ **36281**

8003 A Concrete Naming Certificate for QuasiPolishSpace$^{\uparrow}$ **36284**

8004 A Concrete Naming Certificate for EngelExpansionReal$^{\uparrow}$ **36287**

8005 A Concrete Naming Certificate for ThreeDistanceTheorem$^{\uparrow}$ **36290**

8006 A Concrete Naming Certificate for ClosedSubstitutionSelfCompile$^{\uparrow}$ **36293**

8007 A Concrete Naming Certificate for MobiusInversion$^{\uparrow}$ **36296**

8008 A Concrete Naming Certificate for AnalyticSet$^{\uparrow}$ **36299**

8009 A Concrete Naming Certificate for InjectiveModule$^{\uparrow}$ **36302**

8010 A Concrete Naming Certificate for LocatedRealReciprocal$^{\uparrow}$ **36305**

8011 A Concrete Naming Certificate for CompactUniformModulusCoherence$^{\uparrow}$ **36308**

8012 A Concrete Naming Certificate for StepFunctionUniformApproximation$^{\uparrow}$ **36311**

8013 A Concrete Naming Certificate for GaloisConnectionClosureOperator$^{\uparrow}$ **36314**

8014 A Concrete Naming Certificate for BishopFanCompactnessBoundary$^{\uparrow}$ **36317**

8015 A Concrete Naming Certificate for MetricSequentialLimitPoint$^{\uparrow}$ **36320**

8016 A Concrete Naming Certificate for ConstructiveDarboux$^{\uparrow}$ **36323**

8017 A Concrete Naming Certificate for HyperbolicGeodesicTriangle$^{\uparrow}$ **36326**

8018 A Concrete Naming Certificate for MarkovDecisionProcess$^{\uparrow}$ **36329**

8019 A Concrete Naming Certificate for BishopLocatedRealEnvelope$^{\uparrow}$ **36332**

8020 A Concrete Naming Certificate for LittlewoodPaleyDyadicBlock$^{\uparrow}$ **36335**

8021 Concrete Naming Certificate for FiniteResidualJoinWindow↑ **36338**

8022 Concrete Naming Certificate for OscillationSeparation↑ **36341**

8023 Concrete Naming Certificate for DistributionSpace↑ **36344**

8024 Concrete Naming Certificate for SternBrocotDyadicApproximation↑ **36347**

8025 Concrete Naming Certificate for DenjoyCarleman↑ **36350**

8026 Concrete Naming Certificate for UniformPartitionVariation↑ **36353**

8027 Concrete Naming Certificate for RegularCauchyEpsilonTail↑ **36356**

8028 Concrete Naming Certificate for BaireClassTwoRealFunction↑ **36359**

8029 Concrete Naming Certificate for BaireClassThreeRealFunction↑ **36362**

8030 Concrete Naming Certificate for RegularCauchyAdmission↑ **36365**

8031 Concrete Naming Certificate for LipschitzFixedPoint↑ **36367**

8032 Concrete Naming Certificate for RiemannSumModulus↑ **36370**

8033 Concrete Naming Certificate for CompactMetricSequentialComplete↑ **36373**

8034 Concrete Naming Certificate for LocalObservationDigestGate↑ **36376**

8035 Concrete Naming Certificate for FiniteRealCauchyCluster↑ **36379**

8036 Concrete Naming Certificate for OdeComparison↑ **36382**

8037 Concrete Naming Certificate for CauchyRateSpace↑ **36385**

8038 Concrete Naming Certificate for CompactUniformPointwiseRadiusChain↑ **36387**

8039 Concrete Naming Certificate for ChristoffelFunction↑ **36389**

8040 Concrete Naming Certificate for CauchyNetCriterion↑ **36392**

8041 Concrete Naming Certificate for RegularCauchyCutoff↑ **36395**

8042 Concrete Naming Certificate for LawlikeLocatedRealSequence↑ **36398**

8043 Concrete Naming Certificate for MonotoneBoundedConvergence↑ **36401**

8044 Concrete Naming Certificate for VisionLedgerRoute↑ **36403**

8045 Concrete Naming Certificate for HenstockKurzweilGaugeIntegral↑ **36406**

8046 Concrete Naming Certificate for MetaCICResidualLocalDiamondWitness↑ **36409**

8047 Concrete Naming Certificate for HyperbolicBarycenter↑ **36412**

8048 Concrete Naming Certificate for LocatedClosedSetDistance↑ **36415**

8049 Concrete Naming Certificate for FiniteEpsilonNetChain↑ **36418**

8050 Concrete Naming Certificate for IdempotentCompletionSplitting↑ **36421**

8051 Concrete Naming Certificate for MetaCICResidualConfluenceBudget↑ **36424**

8052 Concrete Naming Certificate for RegularCauchyTailSeal↑ **36427**

8053 Concrete Naming Certificate for LawlikeCauchyModulus↑ **36429**

8054 Concrete Naming Certificate for InfimalConvolution↑ **36432**

8055 Concrete Naming Certificate for MetricEntropy↑ **36435**

8056 Concrete Naming Certificate for MinCostFlow↑ **36438**

8057 Concrete Naming Certificate for MetaCICBoundedNormalEqualityBudget↑ **36441**

8058 Concrete Naming Certificate for UniformlyDiscreteSpace↑ **36444**

8059 Concrete Naming Certificate for RegularCauchySubsequenceCriterion↑ **36446**

8060 Concrete Naming Certificate for LpSpace↑ **36449**

8061 Concrete Naming Certificate for LovaszLocalLemma↑ **36452**

8062 Concrete Naming Certificate for TrotterKato↑ **36455**

8063 Concrete Naming Certificate for BishopCompletionSequentialClosure↑ **36458**

8064 Concrete Naming Certificate for RiemannSumCriterion↑ **36461**

8065 Concrete Naming Certificate for FilterBasisConvergence↑ **36463**

8066 Concrete Naming Certificate for RoughPath↑ **36466**

8067 Concrete Naming Certificate for BaireOneUniformOscillation↑ **36469**

8068 IcShaneIntegralUp NameCert construction **36472**

8069 Concrete Naming Certificate for MengerTheorem↑ **36475**

8070 Concrete Naming Certificate for FiniteTailSeries↑ **36478**

8071 Concrete Naming Certificate for BishopRealCauchyInterleaving↑ **36480**

8072 Concrete Naming Certificate for ConditionalConvergence↑ **36483**

8073 Concrete Naming Certificate for CompactUniformDiagonalNet↑ **36486**

8074 Concrete Naming Certificate for RealOpenBall↑ **36489**

8075 Concrete Naming Certificate for BishopLTenComparisonPrinciple↑ **36492**

8076 Concrete Naming Certificate for DirichletJordanTest↑ **36495**

8077 Concrete Naming Certificate for RealClosedBall↑ **36498**

8078 Concrete Naming Certificate for BishopCut↑ **36501**

8079 Concrete Naming Certificate for FiniteCandidateDiamond↑ **36504**

8080 Concrete Naming Certificate for BirkhoffVonNeumannFinite↑ **36507**

8081 Concrete Naming Certificate for FareyApproximation↑ **36510**

8082 Concrete Naming Certificate for CauchyUniformCriterion↑ **36513**

8083 Concrete Naming Certificate for MetaCICDecidableTypecheckingHandoff↑ **36516**

8084 Concrete Naming Certificate for RegulatedCauchyFilter↑ **36519**

8085 Concrete Naming Certificate for FiniteOpenBallCover↑ **36522**

8086 Concrete Naming Certificate for LocatedLowerCut↑ **36524**

8087 Concrete Naming Certificate for ConfigurationSpace↑ **36527**

8088 Concrete Naming Certificate for BishopLocatedCompactSet↑ **36530**

8089 Concrete Naming Certificate for HyperbolicMarkovPartition↑ **36533**

8090 Concrete Naming Certificate for LocatedRegularCauchyEquivalence↑ **36536**

8091 Concrete Naming Certificate for RegularCauchyWindowInduction↑ **36539**

8092 Concrete Naming Certificate for DyadicRealBracketRefinement↑ **36542**

8093 Concrete Naming Certificate for KleeneRealizability↑ **36545**

8094 Concrete Naming Certificate for LocatedCompactIntervalSequence↑ **36548**

8095 Concrete Naming Certificate for RegularCauchyOscillationWindow↑ **36551**

8096 Concrete Naming Certificate for EllInftySequenceSpace↑ **36554**

8097 Concrete Naming Certificate for CompactCauchyFilter↑ **36557**

8098 Concrete Naming Certificate for HyperbolicSymbolicCoding↑ **36560**

809 Tug-of-war game NameCert construction **36563**

810 A Concrete Naming Certificate for LinearBoundedAutomaton↑ **36566**

810 A Concrete Naming Certificate for IrrationalReal↑ **36569**

810 A Concrete Naming Certificate for StreamDiagonalization↑ **36572**

810 A Concrete Naming Certificate for BackwardDifferentiationFormula↑ **36575**

810 A Concrete Naming Certificate for TypecheckTraceDecider↑ **36578**

810 A Concrete Naming Certificate for JordanOscillationDecomposition↑ **36581**

810 A Concrete Naming Certificate for BaireOneModulus↑ **36584**

810 A Concrete Naming Certificate for FiniteMeasureDensity↑ **36587**

810 A Concrete Naming Certificate for SchreierRefinement↑ **36590**

810 A Concrete Naming Certificate for ConstructiveClosedIntervalLocator↑ **36593**

811 A Concrete Naming Certificate for CantorBendixsonDerivative↑ **36596**

811 A Concrete Naming Certificate for MetaCICResidualDiamondFrontier↑ **36599**

811 A Concrete Naming Certificate for RegularCauchyEquivalenceCriterion↑ **36602**

811 A Concrete Naming Certificate for DescriptionComplexity↑ **36605**

811 A Concrete Naming Certificate for MarkovBlanket↑ **36608**

811 A Concrete Naming Certificate for BinarySignedDigit↑ **36611**

811 A Concrete Naming Certificate for RationalBallFilter↑ **36614**

811 A Concrete Naming Certificate for RationalBallCover↑ **36617**

811 A Concrete Naming Certificate for CompactUniformFinalTriangleClosure↑ **36620**

811 A Concrete Naming Certificate for BishopLocatedChoiceSequence↑ **36623**

812 A Concrete Naming Certificate for BoundedRealUniformContinuityModulus↑ **36626**

812 A Concrete Naming Certificate for SetValuedMap↑ **36629**

812 A Concrete Naming Certificate for AbelPoissonSummability↑ **36632**

812 A Concrete Naming Certificate for StepIndexedTrace↑ **36635**

812 A Concrete Naming Certificate for KurzweilStieltjesIntegral↑ **36638**

8125 Concrete Naming Certificate for LocatedRealBinarySeparator↑ **36641**

8126 Concrete Naming Certificate for MetaCICCriticalPairResidualMatrix↑ **36644**

8127 Concrete Naming Certificate for CauchyCondensationHarmonicBlock↑ **36647**

8128 Concrete Naming Certificate for SchauderBasisProjection↑ **36650**

8129 Concrete Naming Certificate for SearchableSpace↑ **36653**

8130 Concrete Naming Certificate for BishopUniformGauge↑ **36655**

8131 Concrete Naming Certificate for BishopCompletionReflector↑ **36658**

8132 Concrete Naming Certificate for UniformlyCauchyFilterFamily↑ **36661**

8133 Concrete Naming Certificate for ModulusToRegularCauchy↑ **36664**

8134 Concrete Naming Certificate for MetaCICResidualJoinMatrix↑ **36667**

8135 Concrete Naming Certificate for RegularCauchyExtensionModulus↑ **36670**

8136 Concrete Naming Certificate for RegularCauchySubdivision↑ **36673**

8137 Concrete Naming Certificate for BoundedLipschitzDomain↑ **36676**

8138 Concrete Naming Certificate for KolmogorovRieszCriterion↑ **36679**

8139 Concrete Naming Certificate for RegularCauchyBisimulation↑ **36682**

8140 Concrete Naming Certificate for HyperbolicOperatorTrace↑ **36684**

8141 Concrete Naming Certificate for LowerTopology↑ **36686**

8142 Concrete Naming Certificate for DescentData↑ **36689**
8142.1 Descent Carrier 36689

8143 Concrete Naming Certificate for LipschitzExtensionDomain↑ **36692**

8144 Concrete Naming Certificate for LinearRecurrence↑ **36695**

8145 Concrete Naming Certificate for CauchyTailComparisonFunctor↑ **36698**

8146 Concrete Naming Certificate for BishopLocatedDiagonal↑ **36701**
8146.1 Carrier definition 36701
8146.2 Certificate obligations 36702

8147 Concrete Naming Certificate for CrandallLiggettSemigroup↑ **36704**

8148 Concrete Naming Certificate for CompactRegularCauchyCriterion↑ **36707**

8149 Concrete Naming Certificate for BishopEffectiveRealName↑ **36709**

8150 Concrete Naming Certificate for ProjectionValuedMeasure† **36712**

8151 Concrete Naming Certificate for PeterWeyl† **36715**

8152 Concrete Naming Certificate for DifferentiationUnderIntegral† **36718**

8153 Concrete Naming Certificate for MetricCompletionFunctorialSeal† **36721**

8154 Concrete Naming Certificate for HermiteFunction† **36724**

8155 Concrete Naming Certificate for WeierstrassMTestWindow† **36727**

8156 Concrete Naming Certificate for GaussDivergenceTheorem† **36730**

8157 Concrete Naming Certificate for DyadicAbsoluteValueStability† **36733**

8158 Concrete Naming Certificate for MonotoneRealFunction† **36736**

8159 Concrete Naming Certificate for RegularCauchySheaf† **36739**

8160 Concrete Naming Certificate for SphericallyCompleteUltrametric† **36741**

8161 Concrete Naming Certificate for BishopSpread† **36744**
8161.0 Carrier and obligations 36744

8162 Concrete Naming Certificate for HyperbolicPhaseHolonomyIndex† **36747**
8162.0 Carrier and vision route 36747

8163 Concrete Naming Certificate for FiniteModulusBundle† **36750**

8164 Concrete Naming Certificate for RegularCauchyWindowLattice† **36753**

8165 Concrete Naming Certificate for CauchyTailInterleaving† **36756**

8166 Concrete Naming Certificate for BishopRegularCauchyMetric† **36759**

8167 Concrete Naming Certificate for CompactUniformModulusSelector† **36762**

8168 Concrete Naming Certificate for AlexanderDuality† **36765**

8169 Concrete Naming Certificate for BishopRealLocatedness† **36768**

8170 Concrete Naming Certificate for CompactModulusComposition† **36771**

8171 Concrete Naming Certificate for SierpinskiUniversalCurve† **36774**

8172 Concrete Naming Certificate for LowerSemicontinuousRegularization† **36777**

8173 Concrete Naming Certificate for CompactLawFailureEnvelope† **36780**

8174 Concrete Naming Certificate for CompactTotallyBoundedFunctionFamily† **36783**

8175 A Concrete Naming Certificate for RealSegment† **36787**

8176 A Concrete Naming Certificate for ResidueTheorem† **36790**

8177 A Concrete Naming Certificate for ZTransform† **36793**

8178 A Concrete Naming Certificate for HyperbolicObserverPhaseDistanceShell† **36796**

8179 A Concrete Naming Certificate for HilbertSchmidt† **36799**

8180 A Concrete Naming Certificate for WhittakerShannonSampling† **36802**

8181 A Concrete Naming Certificate for BishopCompletionWitness† **36804**

8182 A Concrete Naming Certificate for StrongLawLargeNumbers† **36807**

8183 A Concrete Naming Certificate for ClassifierInformationBudget† **36810**

8184 A Concrete Naming Certificate for SchauderFiniteCompactMap† **36813**

8185 A Concrete Naming Certificate for DyadicNestedIntervalCode† **36816**

8186 A Concrete Naming Certificate for RegularCauchyExponential† **36819**

8187 A Concrete Naming Certificate for LocatednessPrinciple† **36822**

8188 A Concrete Naming Certificate for KolmogorovReflection† **36825**

8189 A Concrete Naming Certificate for MontelNormalFamily† **36828**

8190 A Concrete Naming Certificate for BishopLocatedOscillationModulus† **36831**

8191 A Concrete Naming Certificate for MetaCICResidualChoiceBudget† **36834**

8192 A Concrete Naming Certificate for AbelCesaroTheorem† **36837**

8193 A Concrete Naming Certificate for TailWindowUniformSequence† **36840**

8194 A Concrete Naming Certificate for CompactModulusTailFusion† **36842**

8195 A Concrete Naming Certificate for UniformAbsoluteConvergence† **36844**

8196 A Concrete Naming Certificate for PhaseCurvatureTransport† **36847**

8197 A Concrete Naming Certificate for SignedDyadicInterval† **36850**

8198 A Concrete Naming Certificate for UniformLimitIntegral† **36854**

8199 A Concrete Naming Certificate for CompactIntervalFanModulusSelector† **36857**

8200 A Concrete Naming Certificate for FiniteDimensionalBrouwerFixedPoint† **36860**

8201 Concrete Naming Certificate for DyadicConvergenceModulus$^{\uparrow}$ **36863**

8202 Concrete Naming Certificate for BishopUniformCauchyPrinciple$^{\uparrow}$ **36866**

8203 Concrete Naming Certificate for RegularCauchyTailBudgetComparison$^{\uparrow}$ **36869**

8204 Concrete Naming Certificate for CauchyCompletionKanExtension$^{\uparrow}$ **36872**

8205 Concrete Naming Certificate for UniformLimitDerivative$^{\uparrow}$ **36875**

8206 Concrete Naming Certificate for LedgerModeClassifier$^{\uparrow}$ **36878**

8207 Concrete Naming Certificate for BisectionCauchy$^{\uparrow}$ **36881**

8208 Concrete Naming Certificate for DyadicIntervalChain$^{\uparrow}$ **36885**

8209 Concrete Naming Certificate for RegularCauchyOscillationBound$^{\uparrow}$ **36888**

8210 Concrete Naming Certificate for RegularCauchyCofinalModulus$^{\uparrow}$ **36890**

8211 Concrete Naming Certificate for NavierStokesFiniteEnergy$^{\uparrow}$ **36893**

8212 Concrete Naming Certificate for PriestleySpace$^{\uparrow}$ **36896**

8213 Concrete Naming Certificate for GroupCohomology$^{\uparrow}$ **36899**

8214 Concrete Naming Certificate for DavenportConstant$^{\uparrow}$ **36902**

8215 Concrete Naming Certificate for RegulatedCauchyStepApproximation$^{\uparrow}$ **36905**

8216 Concrete Naming Certificate for FiniteNetTriangleReplay$^{\uparrow}$ **36908**

8217 Concrete Naming Certificate for RegularCauchyCompletionDensity$^{\uparrow}$ **36911**

8218 Concrete Naming Certificate for RealCompletenessDensityWindow$^{\uparrow}$ **36914**

8219 Concrete Naming Certificate for FiniteStreamMerge$^{\uparrow}$ **36917**

8220 Concrete Naming Certificate for DyadicPiecewiseLinearInterpolation$^{\uparrow}$ **36920**

8221 Concrete Naming Certificate for RegularClosedSet$^{\uparrow}$ **36923**

8222 Concrete Naming Certificate for CompactRealIntervalNetSearch$^{\uparrow}$ **36926**

8223 Concrete Naming Certificate for HalesJewett$^{\uparrow}$ **36930**

8224 Concrete Naming Certificate for ChineseRemainder$^{\uparrow}$ **36933**

8225 Concrete Naming Certificate for RationalIntervalPartition$^{\uparrow}$ **36936**

8226 Concrete Naming Certificate for LocatedCompletionAdjunction$^{\uparrow}$ **36939**

8227 A Concrete Naming Certificate for DiagonalLimitModulusCertificate$^{\uparrow}$ **36942**

8228 Fitting Lemma Name Certificate Construction **36945**

8229 A Concrete Naming Certificate for SpecializationTopology$^{\uparrow}$ **36948**

8230 A Concrete Naming Certificate for CuspNeighborhood$^{\uparrow}$ **36951**

8231 A Concrete Naming Certificate for RemezExchange$^{\uparrow}$ **36954**

8232 A Concrete Naming Certificate for SequentialCompletionMorphism$^{\uparrow}$ **36957**

8233 A Concrete Naming Certificate for ChoiceSequenceSpread$^{\uparrow}$ **36960**

8234 A Concrete Naming Certificate for RegulaFalsi$^{\uparrow}$ **36963**

8235 A Concrete Naming Certificate for MetacicTypedEqualityEliminator$^{\uparrow}$ **36966**

8236 A Concrete Naming Certificate for FinitePositiveMinimum$^{\uparrow}$ **36969**

8237 A Concrete Naming Certificate for CauchyModulusLimiter$^{\uparrow}$ **36972**

8238 A Concrete Naming Certificate for ProjectiveLimitMetric$^{\uparrow}$ **36975**

8239 A Concrete Naming Certificate for CompactCompleteFixedPoint$^{\uparrow}$ **36978**

8240 A Concrete Naming Certificate for HautusTest$^{\uparrow}$ **36981**

8241 A Concrete Naming Certificate for ConstructiveOdeFlowBox$^{\uparrow}$ **36984**

8242 A Concrete Naming Certificate for CompactUniformTriangleChain$^{\uparrow}$ **36987**

8243 A Concrete Naming Certificate for ClosedSubspaceComplete$^{\uparrow}$ **36990**

8244 A Concrete Naming Certificate for GrussInequality$^{\uparrow}$ **36993**

8245 A Concrete Naming Certificate for ReedyCategory$^{\uparrow}$ **36996**

8246 A Concrete Naming Certificate for ReedyModelStructure$^{\uparrow}$ **36999**

8247 A Concrete Naming Certificate for NestedIntervalSqueeze$^{\uparrow}$ **37002**

8248 A Concrete Naming Certificate for CoreCompactSpace$^{\uparrow}$ **37005**

8249 A Concrete Naming Certificate for BishopLocatedRealComparator$^{\uparrow}$ **37008**

8250 A Concrete Naming Certificate for TypecheckingClassifierTrace$^{\uparrow}$ **37011**

8251 A Concrete Naming Certificate for CombSpace$^{\uparrow}$ **37014**

8252 A Concrete Naming Certificate for BrouwerContinuityModulusBridge$^{\uparrow}$ **37017**

8253 Concrete Naming Certificate for BanachContractionSequence↑ **37020**

8254 Concrete Naming Certificate for MetacicConfluenceResidualFrontier↑ **37023**

8255 Concrete Naming Certificate for TheoremOfAscoli↑ **37026**

8256 Concrete Naming Certificate for HyperbolicCannonThurstonBoundary↑ **37028**

8257 Concrete Naming Certificate for EntourageFilter↑ **37030**

8258 Concrete Naming Certificate for BishopLocatedCompletionPair↑ **37033**

8259 Concrete Naming Certificate for FiniteCauchyWitnessMatrix↑ **37036**

8260 Finite Uniform Discrepancy NameCert Construction **37039**

8261 Concrete Naming Certificate for SobczykTheorem↑ **37042**

8262 Concrete Naming Certificate for MetacicConfluenceForkCertificate↑ **37044**

8263 Concrete Naming Certificate for CompactModulusLedger↑ **37046**

8264 Concrete Naming Certificate for JordanVariation↑ **37049**

8265 Concrete Naming Certificate for AscoliRealCompletion↑ **37052**

8266 Concrete Naming Certificate for LocatedMonotoneSequence↑ **37055**

8267 Concrete Naming Certificate for PreuniformBase↑ **37058**

8268 Concrete Naming Certificate for BanachContractionModulus↑ **37061**

8269 Concrete Naming Certificate for FiniteEpsilonNetUniformModulus↑ **37064**

8270 Concrete Naming Certificate for FanTheoremModulus↑ **37067**

8271 Concrete Naming Certificate for LebesgueCriterionRiemannIntegrability↑ **37070**

8272 Concrete Naming Certificate for UniformContinuityFanModulus↑ **37073**

8273 Concrete Naming Certificate for UniformCover↑ **37076**

8274 Concrete Naming Certificate for RadoSelectionLemma↑ **37079**

8275 Concrete Naming Certificate for ProofTraceConfluence↑ **37081**

8276 Concrete Naming Certificate for CousinIProblem↑ **37084**

8277 Concrete Naming Certificate for LocatedRealCauchyBoundary↑ **37087**

8278 Concrete Naming Certificate for SequentialArchimedeanReal↑ **37089**

8279 A Concrete Naming Certificate for RealOrderWindow↑ **37092**

8280 A Concrete Naming Certificate for ClarksonInequality↑ **37094**

8281 A Concrete Naming Certificate for SemilocalRegularCauchyModulus↑ **37097**

8282 A Concrete Naming Certificate for CauchyBoundSelector↑ **37100**

8283 A Concrete Naming Certificate for MetaCICConfluenceBudgetMatrix↑ **37103**

8284 A Concrete Naming Certificate for MetaCICCandidatePruningBudget↑ **37106**

8285 A Concrete Naming Certificate for FredholmOperator↑ **37109**

8286 A Concrete Naming Certificate for HenstockGaugePartition↑ **37112**

8287 A Concrete Naming Certificate for RealUniformPartition↑ **37115**

8288 A Concrete Naming Certificate for RegularCauchyProductModulus↑ **37118**

8289 A Concrete Naming Certificate for MartingaleFiltration↑ **37121**

8290 A Concrete Naming Certificate for DyadicUniformCover↑ **37124**

8291 A Concrete Naming Certificate for RationalBallDomain↑ **37126**

8292 A Concrete Naming Certificate for PositiveRealSquareRoot↑ **37129**

8293 A Concrete Naming Certificate for RegularCauchySieve↑ **37132**

8294 A Concrete Naming Certificate for RegularCauchyRateTransform↑ **37134**

8295 A Concrete Naming Certificate for CauchyRealEmbedding↑ **37137**

8296 A Concrete Naming Certificate for MartinBoundary↑ **37140**

8297 A Concrete Naming Certificate for CompactPositiveRealSquareRoot↑ **37143**

8298 A Concrete Naming Certificate for MetacicObserverPhaseLocality↑ **37146**

8299 A Concrete Naming Certificate for UniformlyCauchyRealSequence↑ **37149**

8300 A Concrete Naming Certificate for NestedClosedSets↑ **37152**

8301 A Concrete Naming Certificate for LindenbaumAlgebra↑ **37155**

8302 A Concrete Naming Certificate for RegularCauchyEqualityModulus↑ **37158**

8303 A Concrete Naming Certificate for FiniteContractionOrbitError↑ **37161**

8304 A Concrete Naming Certificate for LocatedSignApartness↑ **37164**

8305 A Concrete Naming Certificate for BorelCantelliSecondLemma↑ **37167**

8306 A Concrete Naming Certificate for FiniteRadonTightness↑ **37170**

8307 A Concrete Naming Certificate for LocatedWeakKonig↑ **37172**

8308 A Concrete Naming Certificate for FinitePrefixObservationInterface↑ **37175**

8309 A Concrete Naming Certificate for BishopRegularCauchyLocator↑ **37178**

8310 A Concrete Naming Certificate for RegularCauchyDiagonalLimitModulus↑ **37181**

8311 A Concrete Naming Certificate for RegularCauchyStreamReflector↑ **37184**

8312 A Concrete Naming Certificate for KolmogorovConsistency↑ **37186**

8313 A Concrete Naming Certificate for ModulusTailSelection↑ **37189**

8314 A Concrete Naming Certificate for NoetherianInduction↑ **37192**

8315 A Concrete Naming Certificate for LocatedCompletionCore↑ **37195**

8316 A Concrete Naming Certificate for MetacicResidualCriticalPairSieve↑ **37198**

8317 A Concrete Naming Certificate for MetaCICCriticalPairResidualScheduler↑ **37201**

8318 A Concrete Naming Certificate for UniformCauchyCompletionBridge↑ **37204**

8319 A Concrete Naming Certificate for RegularLagrangian↑ **37207**

8320 A Concrete Naming Certificate for CauchyCondensationModulus↑ **37210**

8321 A Concrete Naming Certificate for BishopEpsilonDeltaRealEquality↑ **37213**

8322 A Concrete Naming Certificate for FiniteEpsilonNetSelector↑ **37216**

8323 A Concrete Naming Certificate for HyperbolicBigonThinness↑ **37219**

8324 A Concrete Naming Certificate for MetricCompletionEtaLaw↑ **37222**

8325 A Concrete Naming Certificate for AronszajnSmithInvariantSubspace↑ **37225**

8326 A Concrete Naming Certificate for KantorovichRubinsteinDuality↑ **37228**

8327 A Concrete Naming Certificate for FiniteContinuedFraction↑ **37231**

8328 A Concrete Naming Certificate for SphericalCompletion↑ **37234**

8329 A Concrete Naming Certificate for FiniteCoverMultiplicity↑ **37237**

8330 A Concrete Naming Certificate for BishopRegSeqRealEta↑ **37240**

8331 A Concrete Naming Certificate for RealCompletionEtaLaw↑ 37243

8332 A Concrete Naming Certificate for CauchyCompletionDenseEmbeddingComparison↑ 37246

8333 A Concrete Naming Certificate for DyadicRealBall↑ 37249

8334 A Concrete Naming Certificate for BorelCantelliLemma↑ 37251

8335 A Concrete Naming Certificate for FiniteWindowProofIrrelevanceSieve↑ 37254

8336 A Concrete Naming Certificate for CompleteMetricContractionPrinciple↑ 37257

8337 A Concrete Naming Certificate for LocatedRealOrder↑ 37260

8338 A Concrete Naming Certificate for CompactSourceUniformRadiusSelector↑ 37263

8339 A Concrete Naming Certificate for BishopRegularCauchyTwoScale↑ 37266

8340 A Concrete Naming Certificate for DyadicMeshDiameter↑ 37268

8341 A Concrete Naming Certificate for Window-Six Weight Log-Concavity 37271

8342 The Window6 Weight Polynomial: A Finite-Window Positivity Ladder on a Fibonacci Carrier 37273
8342.1 The Ladder . . . 37273
8342.2 The Spectral-Determinant Identification . . . 37274

8343 The Golden-Mean Ruelle Zeta: A Finite Fibonacci Transfer Certificate 37276
8343.1 The Weighted Periodic-Point Zeta . . . 37276
8343.2 The Finite Zeta Certificate . . . 37277
8343.3 The Golden Pole Factorisation . . . 37277

8344 The Golden-Mean Primitive-Orbit Mobius Certificate 37279
8344.1 The Weighted Primitive-Orbit Transform . . . 37279
8344.2 What This Certificate Claims . . . 37280

8345 The Golden-Mean Parry Residue: A Cylinder-Resolved Finite RPF Certificate 37282
8345.0 Cylinder-Resolved Transfer Powers . . . 37282
8345.2 Adjugate and Pressure Shadow . . . 37283
8345.3 Golden Parry Residue . . . 37283

8346 The Golden-Mean Hankel–Turan Discriminant Identity 37285

8347 The Golden-Mean Zeta Denominator as a Fifth-Cyclotomic Pentagon Certificate 37287
8347.1 The Fifth-Cyclotomic Twisted Norm . . . 37287
8347.2 The Pentagon Bridge . . . 37288
8347.3 Scope . . . 37289

8348 The Golden-Mean Shift Equivalence: A Finite Presentation-Invariance Certificate 37290
8348.1 The Rectangular Factors . . . 37290
8348.2 The Presentation-Invariance Certificate . . . 37290

8349 The Golden-Mean Quadratic-Algebra Capstone 37292
8349.1 The Master Norm Identity . . . 37292
8349.2 Powers, Cassini, and Truncated Resolvents . . . 37293
8349.3 Honest Scope Across the Six Facets . . . 37293

8350 The Finite Golden-Mean Zeta Program 37295
8350.1 The Central Norm Identity . . . 37295
8350.2 The Facet Table . . . 37296
8350.3 The Program Boundary . . . 37296

8351 A Concrete Naming Certificate for DyadicTentFunction↑ **37298**

8352 A Concrete Naming Certificate for Grassmannian↑ **37301**

8353 A Concrete Naming Certificate for CompactUniformLocalModulusNerve↑ **37304**

8354 A Concrete Naming Certificate for LocatedCompactMetric↑ **37307**

8355 A Concrete Naming Certificate for BolzanoWeierstrassLocatedSequence↑ **37310**

8356 A Concrete Naming Certificate for CompactIntervalModulusExtractor↑ **37313**

8357 A Concrete Naming Certificate for RealSeriesComparison↑ **37316**

8358 A Concrete Naming Certificate for StirlingFactorialBudget↑ **37319**

8359 A Concrete Naming Certificate for CompactIntervalSupremum↑ **37322**

8360 A Concrete Naming Certificate for MetaCICResidualPeakScheduler↑ **37325**

8361 A Concrete Naming Certificate for UniformConvergencePreservesContinuity↑ **37328**

8362 A Concrete Naming Certificate for FinitelyPresentedModule↑ **37331**

8363 A Concrete Naming Certificate for BishopCauchyClusterModulus↑ **37334**

8364 A Concrete Naming Certificate for PanitchpakdiHyperconvexInjectivity↑ **37337**

8365 A Concrete Naming Certificate for NestedDyadicIntervalFusion↑ **37340**

8366 A Concrete Naming Certificate for MoreraTheorem↑ **37343**

8367 A Concrete Naming Certificate for BishopCompletionUniquenessPrinciple↑ **37346**

8368 A Concrete Naming Certificate for MetaCICCandidateResidualJoinBudget↑ **37349**

8369 A Concrete Naming Certificate for MetaCICAuditSynthesisRouter↑ **37352**

8370 A Concrete Naming Certificate for SchurDecomposition↑ **37355**

8371 A Concrete Naming Certificate for IntegralTest↑ **37358**

8372 A Concrete Naming Certificate for FiniteSelectionCorrespondence↑ **37361**

8373 A Concrete Naming Certificate for SeparatedCauchyCompletion↑ **37363**

8374 A Concrete Naming Certificate for CompactCoverOrderBound↑ **37366**

8375 A Concrete Naming Certificate for DyadicCutStability↑ **37369**

8376 A Concrete Naming Certificate for RegularCauchyCompletionRetraction↑ **37372**

8377 A Concrete Naming Certificate for RegularCauchyTailReparametrization↑ **37375**

8378 A Concrete Naming Certificate for SturmSeparation↑ **37378**

8379 A Concrete Naming Certificate for MetaCICSubjectReductionResidualFence↑ **37381**

8380 A Concrete Naming Certificate for AsymptoticCompactness↑ **37385**

8381 A Concrete Naming Certificate for BishopLocatedBallCover↑ **37387**

8382 A Concrete Naming Certificate for BishopDiagonalCompletion↑ **37390**

8383 A Concrete Naming Certificate for RegularCauchyLimsup↑ **37393**

8384 A Concrete Naming Certificate for CauchyProductTopology↑ **37396**

8385 A Concrete Naming Certificate for SeparatedCauchyCompletionLimit↑ **37399**

8386 A Concrete Naming Certificate for RegularSequenceTailMetric↑ **37402**

8387 A Concrete Naming Certificate for KrawczykOperator↑ **37405**

8388 A Concrete Naming Certificate for IntervalNewtonKrawczyk↑ **37408**

8389 A Concrete Naming Certificate for BishopLocatedConvexHull↑ **37411**

8390 A Concrete Naming Certificate for FiniteIntervalChainSelection↑ **37413**

8391 A Concrete Naming Certificate for LUDecomposition↑ **37416**

8392 A Concrete Naming Certificate for FiniteTailSandwich↑ **37420**

8393 A Concrete Naming Certificate for BishopRegularCompletionModulus↑ **37423**

8394 A Concrete Naming Certificate for CompactUniformGraphModulus↑ **37426**

8395 A Concrete Naming Certificate for BishopLocatedCompactness$^{\uparrow}$ **37429**

8396 A Concrete Naming Certificate for RegularSequenceClusterWitness$^{\uparrow}$ **37432**

8397 A Concrete Naming Certificate for CholeskyDecomposition$^{\uparrow}$ **37436**

8398 A Concrete Naming Certificate for ParacompactLocallyFinite$^{\uparrow}$ **37439**

8399 A Concrete Naming Certificate for MaxFlowMinCut$^{\uparrow}$ **37442**

8400 A Concrete Naming Certificate for MetaCICBoundedConversionFrontier$^{\uparrow}$ **37445**

8401 A Concrete Naming Certificate for BishopUniformStructure$^{\uparrow}$ **37448**

8402 A Concrete Naming Certificate for AbelTransformation$^{\uparrow}$ **37451**

8403 A Concrete Naming Certificate for BishopChainConnected$^{\uparrow}$ **37454**

8404 A Concrete Naming Certificate for ConstructiveRiemannSum$^{\uparrow}$ **37457**

8405 A Concrete Naming Certificate for FeketeSubadditiveLemma$^{\uparrow}$ **37459**

8406 A Concrete Naming Certificate for CompactOpenEquicontinuity$^{\uparrow}$ **37462**

8407 A Concrete Naming Certificate for FiniteOscillationCover$^{\uparrow}$ **37465**

8408 A Concrete Naming Certificate for OscillationDerivative$^{\uparrow}$ **37468**

8409 A Concrete Naming Certificate for SardRegularValue$^{\uparrow}$ **37471**

8410 A Concrete Naming Certificate for SardTheorem$^{\uparrow}$ **37474**

8411 A Concrete Naming Certificate for MetricDerivedSet$^{\uparrow}$ **37477**

8412 A Concrete Naming Certificate for ApartnessCut$^{\uparrow}$ **37480**

8413 A Concrete Naming Certificate for SequentialArchimedeanWindow$^{\uparrow}$ **37483**

8414 A Concrete Naming Certificate for EntourageCompletion$^{\uparrow}$ **37486**

8415 A Concrete Naming Certificate for BoundedRealInfimum$^{\uparrow}$ **37489**

8416 A Concrete Naming Certificate for ConstantRankTheorem$^{\uparrow}$ **37492**

8417 A Concrete Naming Certificate for SpeckerTheorem$^{\uparrow}$ **37495**

8418 A Concrete Naming Certificate for BishopLocatedRealSegment$^{\uparrow}$ **37498**

8419 A Concrete Naming Certificate for VanKampen$^{\uparrow}$ **37501**

8420 A Concrete Naming Certificate for HarmonicFunction$^{\uparrow}$ **37504**

8421 Concrete Naming Certificate for CodonWindowStopFaceClassifier$^{\uparrow}$ **37507**

8422 Concrete Naming Certificate for RiemannIntegrabilityUniformPartition$^{\uparrow}$ **37510**

8423 Concrete Naming Certificate for LocatedBisectionLimit$^{\uparrow}$ **37513**

8424 Concrete Naming Certificate for TotallyBoundedReal$^{\uparrow}$ **37516**

8425 Concrete Naming Certificate for FiniteSubsequenceCompactness$^{\uparrow}$ **37519**

8426 Concrete Naming Certificate for FiniteWindowSubcover$^{\uparrow}$ **37522**

8427 Concrete Naming Certificate for WhitneyExtension$^{\uparrow}$ **37525**

8428 Concrete Naming Certificate for StratifiableSpace$^{\uparrow}$ **37527**

8429 Concrete Naming Certificate for FiniteCauchyCompletenessCriterion$^{\uparrow}$ **37530**

8430 Concrete Naming Certificate for FiniteNetFan$^{\uparrow}$ **37533**

8431 Concrete Naming Certificate for BorelLebesgueCover$^{\uparrow}$ **37536**

8432 Concrete Naming Certificate for ConstructiveRealOscillationGauge$^{\uparrow}$ **37539**

8433 Concrete Naming Certificate for GoodCover$^{\uparrow}$ **37542**

8434 Concrete Naming Certificate for NormalizationBudgetTrace$^{\uparrow}$ **37545**

8435 Concrete Naming Certificate for BishopLocatedRealFunction$^{\uparrow}$ **37548**

8436 Concrete Naming Certificate for GrothendieckSpectralSequence$^{\uparrow}$ **37551**

8437 Concrete Naming Certificate for RegularCauchyCompactnessCriterion$^{\uparrow}$ **37554**

8438 Concrete Naming Certificate for SpectralSequenceComparison$^{\uparrow}$ **37557**

8439 Concrete Naming Certificate for BrouwerChoiceSequence$^{\uparrow}$ **37560**

8440 Concrete Naming Certificate for TotallyBoundedReflection$^{\uparrow}$ **37563**

8441 Concrete Naming Certificate for BishopLocatedUpperCutCriterion$^{\uparrow}$ **37565**

8442 Concrete Naming Certificate for FiniteDyadicBisection$^{\uparrow}$ **37568**

8443 Concrete Naming Certificate for FiniteWindowChoiceTree$^{\uparrow}$ **37571**

8444 Concrete Naming Certificate for BishopLocatedRealSequenceCalibration$^{\uparrow}$ **37574**

8445 Concrete Naming Certificate for FordCircle$^{\uparrow}$ **37577**

8446 Concrete Naming Certificate for KolmogorovMaximalInequality$^{\uparrow}$ **37579**

8447 A Concrete Naming Certificate for OrderConvexSubspace↑ **37582**

8448 A Concrete Naming Certificate for BishopCauchyModulusCalculus↑ **37585**

8449 A Concrete Naming Certificate for FiniteRealizedSeparableBanachSpace↑ **37588**

8450 A Concrete Naming Certificate for Dvoretsky↑ **37591**

8451 A Concrete Naming Certificate for DyadicCauchyDiagonal↑ **37594**

8452 A Concrete Naming Certificate for HairyBall↑ **37597**

8453 A Concrete Naming Certificate for FiniteCauchyFilterBasisRefinement↑ **37600**

8454 A Concrete Naming Certificate for PringsheimTauberian↑ **37603**

8455 A Concrete Naming Certificate for NewmanLemma↑ **37606**

8456 A Concrete Naming Certificate for MetaCICResidualNormalFormBudgetRouter↑ **37609**

8457 A Concrete Naming Certificate for LocatedRealTailMajorant↑ **37612**

8458 A Concrete Naming Certificate for DualSpace↑ **37615**

8459 A Concrete Naming Certificate for SouslinOperation↑ **37617**

8460 A Concrete Naming Certificate for BoundaryShadowCrossRatio↑ **37620**

8461 Bounded Variation Cauchy Completion NameCert Construction **37623**

8462 A Concrete Naming Certificate for NashEmbedding↑ **37626**

8463 A Concrete Naming Certificate for RationalIntervalSequence↑ **37629**

8464 Nikodym Boundedness NameCert Construction **37632**

8465 A Concrete Naming Certificate for KraftPrefixCode↑ **37635**

8466 A Concrete Naming Certificate for FiniteRealObserver↑ **37638**

8467 A Concrete Naming Certificate for SequentiallyCompactRealInterval↑ **37641**

8468 A Concrete Naming Certificate for TailEquivalenceModulus↑ **37644**

8469 A Concrete Naming Certificate for LocatedRealIntervalChain↑ **37647**

8470 A Concrete Naming Certificate for CauchyGluingLemma↑ **37650**

8471 A Concrete Naming Certificate for BishopSequentialLimitCriterion↑ **37653**

8472 A Concrete Naming Certificate for FiniteCauchyTailHandoff↑ **37657**

8473 Concrete Naming Certificate for RegularCauchyRecursion↑ **37660**

8474 Concrete Naming Certificate for FunctionSequence↑ **37663**

8475 Concrete Naming Certificate for SobolevSlobodeckij↑ **37666**

8476 Concrete Naming Certificate for LebesgueDifferentiationTheorem↑ **37669**

8477 Concrete Naming Certificate for FreudenthalSuspension↑ **37672**

8478 Concrete Naming Certificate for BishopLocatedOscillationSpace↑ **37675**

8479 Concrete Naming Certificate for RationalIntervalCoverage↑ **37678**

8480 Concrete Naming Certificate for DividedPowerAlgebra↑ **37681**

8481 Concrete Naming Certificate for SignedDistanceInterval↑ **37684**

8482 Concrete Naming Certificate for BishopDiagonalRegularization↑ **37688**

8483 Concrete Naming Certificate for CompactUniformSplittingLedger↑ **37691**

8484 Concrete Naming Certificate for TotallyBoundedSubsequence↑ **37694**

8485 Concrete Naming Certificate for CauchyApproximationCategory↑ **37697**

8486 Concrete Naming Certificate for RealModulusContinuity↑ **37700**

8487 Concrete Naming Certificate for ExternalSupplyLedgerRoute↑ **37703**

8488 Concrete Naming Certificate for YoulaKuceraParametrization↑ **37706**

8489 Concrete Naming Certificate for BishopUniformSpaceCompletion↑ **37709**

8490 Concrete Naming Certificate for HyperbolicObserverHorizonCocycle↑ **37712**

8491 Concrete Naming Certificate for RegularCauchyEnrichedCompletion↑ **37715**

8492 Concrete Naming Certificate for HomotopyExtensionProperty↑ **37718**

8493 Concrete Naming Certificate for HigmanLemma↑ **37721**

8494 Concrete Naming Certificate for HewittRealcompactification↑ **37725**

8495 Concrete Naming Certificate for CompactOscillationTree↑ **37728**

8496 Concrete Naming Certificate for FiniteDimensionalNormedCompleteness↑ **37731**

8497 Concrete Naming Certificate for CauchyCompletionErrorEstimator↑ **37734**

8498 Concrete Naming Certificate for CompactMetricRegularSequence↑ **37737**

8499 A Concrete Naming Certificate for DunfordPettisOperator↑ **37740**

8500 A Concrete Naming Certificate for DaugavetProperty↑ **37742**

8501 A Concrete Naming Certificate for DyadicRoundingInterval↑ **37744**

8502 A Concrete Naming Certificate for ClosedGraphCertificate↑ **37747**

8503 A Concrete Naming Certificate for CatalanNumber↑ **37750**

8504 A Concrete Naming Certificate for FiniteModulusMesh↑ **37752**

8505 A Concrete Naming Certificate for FiniteTailOscillationBound↑ **37755**

8506 A Concrete Naming Certificate for FrobeniusNormalForm↑ **37758**

8507 A Concrete Naming Certificate for RegularCauchyChoiceFreeLimit↑ **37762**

8508 A Concrete Naming Certificate for NestedIntervalSequence↑ **37765**

8509 A Concrete Naming Certificate for FiniteModulusLowerBoundLedger↑ **37768**

8510 A Concrete Naming Certificate for AumannIntegral↑ **37770**

8511 A Concrete Naming Certificate for BishopDiagonalContinuityBarrier↑ **37773**

8512 A Concrete Naming Certificate for CauchySequenceModulus↑ **37776**

8513 A Concrete Naming Certificate for CompactUniformEquioscillation↑ **37778**

8514 A Concrete Naming Certificate for FiniteModulusStaircase↑ **37781**

8515 A Concrete Naming Certificate for ModulusOfCauchyConvergence↑ **37784**

8516 A Concrete Naming Certificate for BishopRealFunctionSpace↑ **37787**

8517 A Concrete Naming Certificate for UltrametricSkeleton↑ **37790**

8518 A Concrete Naming Certificate for TotallyBoundedCauchyNet↑ **37792**

8519 A Concrete Naming Certificate for ClassifierShiftDiscoveryGate↑ **37795**

8520 A Concrete Naming Certificate for DyadicIntervalHalving↑ **37798**

8521 A Concrete Naming Certificate for HyperbolicHorosphere↑ **37801**

8522 A Concrete Naming Certificate for MacShaneExtension↑ **37804**

8523 A Concrete Naming Certificate for ControllabilityRank↑ **37807**

8524 A Concrete Naming Certificate for BeckChevalley↑ **37810**

8525 A Concrete Naming Certificate for BoundedRegularCauchyCluster↑ **37813**

8526 A Concrete Naming Certificate for FiniteCoverNerveUniformModulus↑ **37816**

8527 A Concrete Naming Certificate for Stabilizability↑ **37819**

8528 A Concrete Naming Certificate for PaleyWiener↑ **37822**

8529 A Concrete Naming Certificate for UniformCompletionIdempotence↑ **37825**

8530 A Concrete Naming Certificate for RegularCauchyLiminf↑ **37828**

8531 A Concrete Naming Certificate for PrecompactMetricSpace↑ **37831**

8532 A Concrete Naming Certificate for CoarseSpace↑ **37834**

8533 A Concrete Naming Certificate for CauchyCompletionCommutation↑ **37837**

8534 A Concrete Naming Certificate for HyperbolicPhaseVisualModulus↑ **37840**

8535 A Concrete Naming Certificate for SteinitzExchange↑ **37843**

8536 A Concrete Naming Certificate for HeytingAlgebra↑ **37846**

8537 A Concrete Naming Certificate for InhabitedLocatedCut↑ **37849**

8538 A Concrete Naming Certificate for ModulusCofinalFilter↑ **37852**

8539 A Concrete Naming Certificate for PreuniformSpace↑ **37855**

8540 A Concrete Naming Certificate for RegularCauchyReindexingModulus↑ **37858**

8541 A Concrete Naming Certificate for MarcinkiewiczInterpolation↑ **37861**

8542 Finite Intersection Closed Intervals NameCert Construction **37864**

8543 A Concrete Naming Certificate for RiemannLocalization↑ **37867**

8544 A Concrete Naming Certificate for BishopIntervalSlopeBound↑ **37870**

8545 A Concrete Naming Certificate for StepFunctionApproximation↑ **37874**

8546 A Concrete Naming Certificate for HyperbolicSectorEnvelope↑ **37877**

8547 A Concrete Naming Certificate for RegularOpenSet↑ **37880**

8548 A Concrete Naming Certificate for GronwallFiniteOde↑ **37883**

8549 A Concrete Naming Certificate for ContinuousDependenceOde↑ **37886**

8550 A Concrete Naming Certificate for BoundedGeometry↑ **37889**

8551 Concrete Naming Certificate for RieszLemmaFiniteSeparation↑ **37892**

8552 Concrete Naming Certificate for FrechetUrysohnCompletion↑ **37895**

8553 Concrete Naming Certificate for AbelUniformTheorem↑ **37898**

8554 Concrete Naming Certificate for LocatedRealSupremum↑ **37901**

8555 Concrete Naming Certificate for DunfordIntegral↑ **37904**

8556 Concrete Naming Certificate for ClosedBoundedIntervalComplete↑ **37907**

8557 Cauchy Interleaving Tree NameCert Construction **37910**

8558 MetaCIC Residual Join Envelope NameCert Construction **37913**

8559 Concrete Naming Certificate for MetaCICBoundedDiamondWindow↑ **37915**

8560 Concrete Naming Certificate for MetaCICBoundedChurchRosserFragment↑ **37918**

8561 Concrete Naming Certificate for TightApartness↑ **37921**

8562 Concrete Naming Certificate for CompletionExponentialLaw↑ **37924**

8563 Concrete Naming Certificate for TraceTheorem↑ **37927**

8563.1 Carrier 37927

8563.2 Obligations 37928

8564 Concrete Naming Certificate for BMOFiniteWindow↑ **37930**

8565 Concrete Naming Certificate for StreamDiagonalizationCompletion↑ **37933**

8566 Concrete Naming Certificate for BishopCauchyCompletionEquivalence↑ **37936**

8567 Concrete Naming Certificate for ProperUniformSpace↑ **37939**

8568 Concrete Naming Certificate for KatetovTongInsertion↑ **37942**

8569 Concrete Naming Certificate for RegularCauchyRateLattice↑ **37945**

8570 Concrete Naming Certificate for HyperbolicBoundaryCrossRatioDistortion↑ **37947**

8571 Concrete Naming Certificate for CauchyCompletionSeparator↑ **37950**

8572 Concrete Naming Certificate for OpialLemma↑ **37953**

8573 Concrete Naming Certificate for RealTailFilter↑ **37956**

8574 Concrete Naming Certificate for LocatedRealSubset↑ **37959**

8575 Concrete Naming Certificate for CauchyBinet↑ **37962**

8576 Concrete Naming Certificate for DarbouxFunction↑ **37965**

8577 Concrete Naming Certificate for LyndonWord↑ **37967**

8578 Concrete Naming Certificate for IshiharaFiniteTestBoundary↑ **37970**

8579 Concrete Naming Certificate for RegularCauchyPairingFunction↑ **37972**

8580 Concrete Naming Certificate for DyadicRationalBallBasis↑ **37975**

8581 Concrete Naming Certificate for SylvesterEquation↑ **37978**

8582 Concrete Naming Certificate for BoundedVariationFinitePartition↑ **37981**

8583 Concrete Naming Certificate for RegularCauchyUniqueLimit↑ **37984**

8584 Concrete Naming Certificate for FiniteClosedIntervalNestedIntersection↑ **37987**

8585 Concrete Naming Certificate for HausdorffCompletionPullback↑ **37991**

8586 Concrete Naming Certificate for MetricCompletionAdmissibleEmbedding↑ **37994**

8587 Concrete Naming Certificate for CauchyCondensationDyadicBlockWitness↑ **37997**

8588 Concrete Naming Certificate for OrthogonalArray↑ **38001**

8589 Concrete Naming Certificate for AronszajnPanitchpakdiInjectiveMetric↑ **38004**

8590 Concrete Naming Certificate for FiniteLipschitzExtension↑ **38007**

8591 Concrete Naming Certificate for FiniteCoverTree↑ **38010**

8592 Concrete Naming Certificate for StieltjesMeasure↑ **38013**

8593 Concrete Naming Certificate for CountableMetricName↑ **38016**

8594 Concrete Naming Certificate for RegularRealSeparationModulus↑ **38019**

8595 Concrete Naming Certificate for HyperbolicSectorNerve↑ **38022**

8596 Concrete Naming Certificate for HyperbolicAreaDefect↑ **38025**

8597 Concrete Naming Certificate for RegularCauchyLatticeSqueeze↑ **38028**

8598 Concrete Naming Certificate for HyperbolicPhaseExpansionCocycle↑ **38031**

8599 Concrete Naming Certificate for ConstructiveCauchyRealCompletionBridge↑ **38033**

8600 Concrete Naming Certificate for BrouwerianSpreadLaw↑ **38038**

8601 Concrete Naming Certificate for PartiallyOrderedGroup↑ **38041**

8602 Concrete Naming Certificate for ModulusCauchyCompression↑ **38044**

8603 Concrete Naming Certificate for FiniteElementMethod↑ **38047**

8604 Concrete Naming Certificate for MetaCICNormalizationRedexFrontier↑ **38050**

8605 Dyadic Barycentric Refinement NameCert Construction **38053**

8606 Concrete Naming Certificate for AtiyahBottFixedPoint↑ **38056**

8607 Concrete Naming Certificate for MirandaTheoremFiniteCube↑ **38059**

8608 Concrete Naming Certificate for CauchyContinuityCriterion↑ **38062**

8609 Concrete Naming Certificate for ChoiceSequenceWindow↑ **38065**

8610 Concrete Naming Certificate for RamanujanPartition↑ **38068**

8611 Concrete Naming Certificate for BarbalatLemma↑ **38071**

8612 Concrete Naming Certificate for DyadicSqueezeLemma↑ **38073**

8613 Concrete Naming Certificate for CompactContractionOrbit↑ **38076**

8614 Concrete Naming Certificate for UniformApartnessSpace↑ **38079**

8615 Concrete Naming Certificate for CauchyCompletenessRate↑ **38082**

8616 Concrete Naming Certificate for UniformCompletionReflection↑ **38085**

8617 Concrete Naming Certificate for CantorNestedInterval↑ **38088**

8618 Concrete Naming Certificate for ConstructiveCauchyRealLine↑ **38090**

8619 Concrete Naming Certificate for DowkerSpace↑ **38094**

8620 Concrete Naming Certificate for UniformEntourageCompletion↑ **38097**

8621 Concrete Naming Certificate for BishopIntervalSearch↑ **38100**

8622 Concrete Naming Certificate for UniformCauchyMap↑ **38103**

8623 Concrete Naming Certificate for RealUniformLimit↑ **38106**

8624 Concrete Naming Certificate for MooreSmithDirectedNet↑ **38109**

8625 Concrete Naming Certificate for CompactUniformCauchyMap↑ **38112**

8626 Concrete Naming Certificate for CauchyCompletionReflectorUnit↑ **38115**

8627 Concrete Naming Certificate for CompleteGeodesicConvexity↑ **38118**

8628 Concrete Naming Certificate for UniformCauchyRegularFamily↑ **38120**

8629 Concrete Naming Certificate for RegularCauchyOrderedField↑ **38123**

8630 Concrete Naming Certificate for FiniteSubcoverModulusBudget↑ **38126**

8631 Concrete Naming Certificate for BishopModulatedCauchySequence↑ **38129**

8632 Concrete Naming Certificate for MetricCategory↑ **38132**

8633 Concrete Naming Certificate for CantorStieltjesMeasure↑ **38135**

8634 Concrete Naming Certificate for MooreMetrization↑ **38138**

8635 Concrete Naming Certificate for MetaCICResidualNormalizationObstruction↑ **38141**

8636 LefschetzHyperplane NameCert Construction **38144**

8637 Concrete Naming Certificate for ProductiveCauchyName↑ **38146**

8638 Concrete Naming Certificate for CauchyTailEvaluationFunctional↑ **38149**

8639 Concrete Naming Certificate for BirkhoffKakutani↑ **38152**

8640 Concrete Naming Certificate for HyperbolicBoundaryPhaseShadow↑ **38154**

8641 Concrete Naming Certificate for UniformTotalBounded↑ **38157**

8642 Concrete Naming Certificate for HeineBorelCoverSelector↑ **38159**

8643 Concrete Naming Certificate for AntiSpecker↑ **38162**

8644 Concrete Naming Certificate for RealCompletionReflector↑ **38165**

8645 Concrete Naming Certificate for SequentialContinuityCriterion↑ **38169**

8646 Concrete Naming Certificate for HausdorffCompletionFunctor↑ **38172**

8647 Concrete Naming Certificate for FaaDiBruno↑ **38175**

8648 Concrete Naming Certificate for KnoppTailCondensation↑ **38178**

8649 Concrete Naming Certificate for RealIntervalCauchyFilter↑ **38181**

8650 Concrete Naming Certificate for CartanDieudonne↑ **38184**

8651 Concrete Naming Certificate for ClarkeSubdifferential↑ **38186**

8651.1 ClarkeSubdifferential Carrier 38186

8651.2 ClarkeSubdifferential Certificate 38187

8652 Concrete Naming Certificate for FiniteLatticePath↑ **38190**

8653 Concrete Naming Certificate for LevyContinuity$^{\uparrow}$ **38193**

8654 Concrete Naming Certificate for DieudonneCompletion$^{\uparrow}$ **38196**

8655 Concrete Naming Certificate for UniformFilterBase$^{\uparrow}$ **38199**

8656 Concrete Naming Certificate for EquicontinuousCauchyFamily$^{\uparrow}$ **38202**

8657 Concrete Naming Certificate for DyadicIntervalNetCompactness$^{\uparrow}$ **38207**

8658 Concrete Naming Certificate for LawvereTheory$^{\uparrow}$ **38210**

8659 Concrete Naming Certificate for LocatedCantorSet$^{\uparrow}$ **38213**

8660 Concrete Naming Certificate for LocatedIntervalLebesgueNumber$^{\uparrow}$ **38216**

8661 Concrete Naming Certificate for Sobrification$^{\uparrow}$ **38219**

8662 Concrete Naming Certificate for HardCertificateLayer$^{\uparrow}$ **38221**

8663 Concrete Naming Certificate for RealCompletionDiagonalRouter$^{\uparrow}$ **38224**

8664 Concrete Naming Certificate for CompactUniformDiagonalModulus$^{\uparrow}$ **38227**

8665 Concrete Naming Certificate for RegularCauchySubsequenceEquivalence$^{\uparrow}$ **38230**

8666 Concrete Naming Certificate for BishopFilteredRegularCompletion$^{\uparrow}$ **38233**

8667 Concrete Naming Certificate for FiniteVariationMeshCompactness$^{\uparrow}$ **38236**

8668 Concrete Naming Certificate for MetaCICNormalizationAuditRow$^{\uparrow}$ **38239**

8669 Concrete Naming Certificate for BishopFiniteTailComparison$^{\uparrow}$ **38242**

8670 Concrete Naming Certificate for IntertwiningOperator$^{\uparrow}$ **38245**

8671 Concrete Naming Certificate for HyperbolicPhaseEnergyTransport$^{\uparrow}$ **38248**

8672 Concrete Naming Certificate for CauchyProductModulusMajorant$^{\uparrow}$ **38251**

8673 Concrete Naming Certificate for UniformlyLocatedSequence$^{\uparrow}$ **38254**

8674 Concrete Naming Certificate for CompactCompletionFunctor$^{\uparrow}$ **38256**

8675 Concrete Naming Certificate for BanachIndicatrix$^{\uparrow}$ **38259**

8676 Concrete Naming Certificate for BishopLocatedComparison$^{\uparrow}$ **38262**

8677 Concrete Naming Certificate for ArchimedeanDyadicIntervalMesh$^{\uparrow}$ **38265**

8678 Concrete Naming Certificate for BishopLocatedCompletionEnvelope$^{\uparrow}$ **38268**

867.9 A Concrete Naming Certificate for EventuallyFilter↑ **38271**

868.0 A Concrete Naming Certificate for CompactModulusExtractionLattice↑ **38274**

868.1 A Concrete Naming Certificate for MonoidAction↑ **38277**

868.2 A Concrete Naming Certificate for FiniteDyadicSubsequenceSelector↑ **38280**

868.3 A Concrete Naming Certificate for KrullDimension↑ **38283**

868.4 A Concrete Naming Certificate for FiveLemma↑ **38285**

868.5 A Concrete Naming Certificate for HyperbolicSectorExpansion↑ **38288**

868.6 A Concrete Naming Certificate for LusinTheorem↑ **38291**

868.7 A Concrete Naming Certificate for RegularCauchyEventualWindow↑ **38294**

868.8 A Concrete Naming Certificate for WindingNumber↑ **38297**

868.9 A Concrete Naming Certificate for AlgorithmWTypeInference↑ **38300**

869.0 A Concrete Naming Certificate for RieszMarkovKakutani↑ **38303**

869.1 A Concrete Naming Certificate for LoewenheimSkolem↑ **38306**

869.2 A Concrete Naming Certificate for EnrichedMetricSpace↑ **38309**

869.3 A Concrete Naming Certificate for CompletionReflectorMonad↑ **38312**

869.4 A Concrete Naming Certificate for MetricSeparatedReflectorUnit↑ **38315**

869.5 A Concrete Naming Certificate for FiniteTriggerFlipGraph↑ **38318**

869.6 A Concrete Naming Certificate for CompleteMetricPicardIterator↑ **38321**

869.7 A Concrete Naming Certificate for CantorDiagonalTail↑ **38323**

869.8 A Concrete Naming Certificate for BanachPicardOrbit↑ **38326**

869.9 A Concrete Naming Certificate for CoareaFormula↑ **38328**

870.0 A Concrete Naming Certificate for EffectiveOpenBallBasis↑ **38331**

870.1 A Concrete Naming Certificate for RadiusOfConvergence↑ **38334**

870.2 A Concrete Naming Certificate for FrechetUrysohnSpace↑ **38336**

870.3 Compact Uniform Barycentric Refinement NameCert Construction **38339**

870.4 A Concrete Naming Certificate for BishopContinuityTheorem↑ **38342**

8705 A Concrete Naming Certificate for CompletionReflectorKleisliExtension↑ **38345**

8706 A Concrete Naming Certificate for BishopCauchyDiagonal↑ **38348**

8707 A Concrete Naming Certificate for RieszSchauderTheory↑ **38351**

8708 A Concrete Naming Certificate for CompactOpenExponentialLaw↑ **38354**

8709 A Concrete Naming Certificate for EilenbergVariety↑ **38358**

8710 A Concrete Naming Certificate for BishopRegularCompletion↑ **38361**

8711 A Concrete Naming Certificate for CompactUniformModulusExtraction↑ **38364**

8712 A Concrete Naming Certificate for CauchyProductUniformCompletion↑ **38367**

8713 A Concrete Naming Certificate for BishopRegularCauchyCofinalStability↑ **38370**

8714 A Concrete Naming Certificate for CauchyRootTest↑ **38373**

8715 A Concrete Naming Certificate for HyperbolicIdealPentagon↑ **38375**

8716 A Concrete Naming Certificate for ConstructiveCauchyCompletionSelector↑ **38378**

8717 A Concrete Naming Certificate for HartmanGrobman↑ **38381**

8718 A Concrete Naming Certificate for MetricPartitionOfUnity↑ **38384**

8719 Bishop Apartness Field Envelope NameCert Construction **38387**

8720 A Concrete Naming Certificate for RegularCauchySeparatedFilter↑ **38390**

8721 A Concrete Naming Certificate for HyperbolicCrossRatioPhase↑ **38392**

8722 A Concrete Naming Certificate for FiniteLipschitzDifferentiability↑ **38395**

8723 A Concrete Naming Certificate for CofinalTailSelectorBudget↑ **38398**

8724 A Concrete Naming Certificate for MetaCICNormalizationCounterexampleBudget↑ **38401**

8725 A Concrete Naming Certificate for PuncturedInterval↑ **38404**

8726 A Concrete Naming Certificate for DyadicIntervalIntersectionLedger↑ **38407**

8727 A Concrete Naming Certificate for ChoiceFreeFixedPointModulus↑ **38410**

8728 A Concrete Naming Certificate for WarsawCircle↑ **38413**

8729 A Concrete Naming Certificate for RegularCauchyLimitWitness↑ **38416**

8730 A Concrete Naming Certificate for BishopSpreadChoice↑ **38419**

8731 Concrete Naming Certificate for SequentialOrderCompactness↑ **38422**

8732 Concrete Naming Certificate for SequentialCompletionReflector↑ **38425**

8733 Concrete Naming Certificate for DenseRationalEmbedding↑ **38428**

8734 Concrete Naming Certificate for LocatedIntervalSeparation↑ **38431**

8735 Concrete Naming Certificate for DeLaValleePoussinCriterion↑ **38434**

8736 Concrete Naming Certificate for DyadicBallBase↑ **38437**

8737 Concrete Naming Certificate for EgoroffFiniteMeasure↑ **38440**

8738 Concrete Naming Certificate for DyadicApartnessScale↑ **38443**

8739 Concrete Naming Certificate for MetaCICResidualLocalConfluenceMatrix↑ **38446**

8740 Concrete Naming Certificate for RegularCauchyRealizer↑ **38449**

8741 Concrete Naming Certificate for LocatedIntervalModulusPartition↑ **38452**

8742 Concrete Naming Certificate for PerfectlyNormalSpace↑ **38454**

8743 Concrete Naming Certificate for PerronMethodFiniteBarrier↑ **38456**

8744 Concrete Naming Certificate for CauchySequenceModulusChain↑ **38459**

8745 Concrete Naming Certificate for GeneticCodeDeformationLedger↑ **38462**

8746 Concrete Naming Certificate for RegularCauchyModulusOfConvergence↑ **38465**

8747 Concrete Naming Certificate for BishopDyadicLocatedIntervalSelection↑ **38468**

8748 Concrete Naming Certificate for LocatedRealLimit↑ **38471**

8749 Concrete Naming Certificate for CauchyFilterSubsequence↑ **38474**

8750 Concrete Naming Certificate for LusinFiniteContinuityWindow↑ **38477**

8751 Concrete Naming Certificate for FiniteOpenCoverNumber↑ **38480**

8752 Concrete Naming Certificate for BrouwerFixedPointTwoD↑ **38483**

8753 Concrete Naming Certificate for ProperMetricCompletionBoundary↑ **38486**

8754 Concrete Naming Certificate for ClosedIntervalFiniteNetSelector↑ **38489**

8755 Concrete Naming Certificate for SignedDyadicRootBracket↑ **38492**

8756 Concrete Naming Certificate for OmegaGenerationBody↑ **38495**

8757 Concrete Naming Certificate for ToeplitzOperator↑ **38497**

8758 Concrete Naming Certificate for RegularSequenceWindowCoherence↑ **38500**

8759 Concrete Naming Certificate for LocatedMetricReflection↑ **38503**

8760 Concrete Naming Certificate for BishopLocatedSupremum↑ **38506**

8761 Concrete Naming Certificate for RationalFinitePlaceProduct↑ **38509**

8762 Concrete Naming Certificate for CauchySequenceRing↑ **38511**

8763 Concrete Naming Certificate for FiniteMetrizableCompletion↑ **38514**

8764 Concrete Naming Certificate for CauchySpaceCompletion↑ **38517**

8765 Concrete Naming Certificate for FinitePrimeWindowGluing↑ **38520**

8766 Concrete Naming Certificate for FiniteMeshRefinementTree↑ **38523**

8767 Concrete Naming Certificate for RegularCauchyWindowPullback↑ **38525**

8768 Concrete Naming Certificate for CompactlyGeneratedWeakHausdorff↑ **38527**

8769 Concrete Naming Certificate for StaticGreenBoundaryFlux↑ **38530**

8770 Concrete Naming Certificate for WeylCauchyCriterion↑ **38533**

8771 Concrete Naming Certificate for DyadicIntervalContinuityModulus↑ **38536**

8772 Concrete Naming Certificate for ZetaZeroTypeFactorization↑ **38538**

8773 Concrete Naming Certificate for BishopCauchyModulusEnvelope↑ **38541**

8774 Concrete Naming Certificate for CauchyCompletionMonadMultiplication↑ **38544**

8775 Concrete Naming Certificate for GelfandTriple↑ **38547**

8776 Concrete Naming Certificate for Prohorov↑ **38550**

8777 Concrete Naming Certificate for ContractiveCauchyRealizer↑ **38553**

8778 Concrete Naming Certificate for CauchyCompletionSeparatedUniversal↑ **38556**

8779 Concrete Naming Certificate for MarkHistContSpine↑ **38559**

8780 Concrete Naming Certificate for BishopRegularCauchyTailBound↑ **38562**

8781 Concrete Naming Certificate for CauchyEntourageCompletion↑ **38565**

8782 Concrete Naming Certificate for LocatedRealFastCauchy↑ **38568**

8783 Concrete Naming Certificate for SequentialPrecompact↑ **38571**

8784 Concrete Naming Certificate for BaireCategoryCompactMetric↑ **38574**

8785 Concrete Naming Certificate for IwasawaDecomposition↑ **38577**

8786 Concrete Naming Certificate for LocatedLimitOperator↑ **38580**

8787 Concrete Naming Certificate for BishopRealInterval↑ **38583**

8788 Concrete Naming Certificate for TubeLemmaFiniteProduct↑ **38586**

8789 Concrete Naming Certificate for BishopCompactIntervalFanModulus↑ **38589**

8790 Concrete Naming Certificate for BishopCompletionComparisonSquare↑ **38592**

8791 Concrete Naming Certificate for PrimeShadowInterface↑ **38596**

8792 Concrete Naming Certificate for DnaWindowTriggerFrontier↑ **38599**

8793 Concrete Naming Certificate for HenstockGaugeIntegral↑ **38601**

8794 Concrete Naming Certificate for NewtonInverseRadiusLedger↑ **38604**

8795 Concrete Naming Certificate for MetaCICNormalizationConfluenceFrontier↑ **38608**

8796 Concrete Naming Certificate for OgdenLemma↑ **38612**

8797 Concrete Naming Certificate for BolzanoWeierstrassFiniteSubsequence↑ **38615**

8798 Concrete Naming Certificate for CompactIntervalFiniteSubcoverBudget↑ **38618**

8799 Concrete Naming Certificate for RhFreeZeroInitiality↑ **38621**

8800 Concrete Naming Certificate for BishopMonotoneConvergenceBoundary↑ **38624**

8801 Located Interval Enclosure NameCert Construction **38627**

8802 Golden-Mean Shift Window-Six NameCert Construction **38630**

8803 Concrete Naming Certificate for ContinuousLattice↑ **38633**

8804 Concrete Naming Certificate for WeierstrassFunction↑ **38636**

8805 Concrete Naming Certificate for PhillipsLemma↑ **38639**

8806 Concrete Naming Certificate for HellySelectionFiniteVariation↑ **38642**

8807 Concrete Naming Certificate for VCDimension↑ **38645**

8808 Concrete Naming Certificate for JordanCanonicalForm↑ **38648**

8809 A Concrete Naming Certificate for FiniteMeshSelection↑ **38651**

8810 A Concrete Naming Certificate for KrylovSubspace↑ **38654**

8811 A Concrete Naming Certificate for ZeckendorfPhaseShell↑ **38657**

8812 A Concrete Naming Certificate for SionMinimax↑ **38660**

8813 A Concrete Naming Certificate for MooreSmithCauchyNet↑ **38663**

8814 A Concrete Naming Certificate for BorsukUlamAntipodal↑ **38666**

8815 A Concrete Naming Certificate for MetricFanTheorem↑ **38668**

8816 A Concrete Naming Certificate for DyadicDistance↑ **38671**

8817 A Concrete Naming Certificate for BoundedDyadicVariation↑ **38674**

8818 A Concrete Naming Certificate for ConstructiveRealApartness↑ **38676**

8819 A Concrete Naming Certificate for RichardsonExtrapolation↑ **38679**

8820 A Concrete Naming Certificate for FiniteVisionMarkerAudit↑ **38682**

8821 A Concrete Naming Certificate for FluxOperatorGeometry↑ **38685**

8822 A Concrete Naming Certificate for CauchyPeanoEulerPolygon↑ **38688**

8823 A Concrete Naming Certificate for LocatedCauchyNetSelector↑ **38692**

8824 A Concrete Naming Certificate for RhRouteClosureFixedStatementSeparation↑ **38694**

8825 A Concrete Naming Certificate for TotalDerivative↑ **38697**

8826 A Concrete Naming Certificate for LorentzBoostDiskTransport↑ **38700**

8827 A Concrete Naming Certificate for BolzanoWeierstrassRate↑ **38703**

8828 A Concrete Naming Certificate for CartanHadamard↑ **38706**

8829 A Concrete Naming Certificate for OmegaRigidityLedger↑ **38709**

8830 A Concrete Naming Certificate for FiniteEndpointSeparationBudget↑ **38712**

8831 A Concrete Naming Certificate for BingNagataSmirnovMetrization↑ **38715**

8832 A Concrete Naming Certificate for MichaelSelectionFiniteRefinement↑ **38718**

8833 A Concrete Naming Certificate for DyadicUniformCoverTransfer↑ **38721**

8834 A Concrete Naming Certificate for EulerMethodConvergence↑ **38724**

8835 Concrete Naming Certificate for LocalizedDarbouxIntermediate↑ **38727**

8836 Concrete Naming Certificate for LocatedNumber↑ **38730**

8837 Concrete Naming Certificate for MetaCICSubjectReductionObstructionWitness↑ **38733**

8838 Concrete Naming Certificate for DarbouxSumRefinement↑ **38736**

8839 Concrete Naming Certificate for RiemannRoch↑ **38739**

8840 Concrete Naming Certificate for CompactFiniteRadiiMeet↑ **38741**

8841 Concrete Naming Certificate for CompactOperatorSpectrum↑ **38743**

8842 Concrete Naming Certificate for UniformCauchyFilterCompletionFunctor↑ **38746**

8843 Concrete Naming Certificate for MellinTransform↑ **38749**

8844 Concrete Naming Certificate for VariationModulus↑ **38752**

8845 Concrete Naming Certificate for TsirelsonSpace↑ **38755**

8846 Concrete Naming Certificate for BetaSignSelection↑ **38757**

8847 Concrete Naming Certificate for CompactUniformModulusRestriction↑ **38759**

8848 Concrete Naming Certificate for PrimeSkewBoundaryDefectPacket↑ **38762**

8849 Concrete Naming Certificate for BishopRegularRealCriterion↑ **38765**

8850 Concrete Naming Certificate for HyperbolicHoroballFiltration↑ **38767**

8851 Concrete Naming Certificate for CompleteMetricSeparability↑ **38769**

8852 Concrete Naming Certificate for TailDiameterCriterion↑ **38772**

8853 Concrete Naming Certificate for DyadicLipschitzSlopePacket↑ **38775**

8854 Concrete Naming Certificate for FiniteNormalizationDiamond↑ **38777**

8855 Concrete Naming Certificate for DualCone↑ **38779**

8856 Concrete Naming Certificate for FiniteCoverOrderNetwork↑ **38781**

8857 Concrete Naming Certificate for DirectMethodMinimizer↑ **38784**

8858 Concrete Naming Certificate for UpwindFlux↑ **38787**

8859 Concrete Naming Certificate for FiniteDependentFamily↑ **38790**

8860 Concrete Naming Certificate for KnasterBucketHandleContinuum↑ **38792**

8861 Concrete Naming Certificate for BumpFunction↑ **38795**

8862 Concrete Naming Certificate for UniformCauchyFunctionLimit↑ **38798**

8863 Concrete Naming Certificate for BlakersMassey↑ **38801**

8864 Concrete Naming Certificate for RHBoundaryKindSeparation↑ **38804**

8865 Concrete Naming Certificate for PrecompactSpace↑ **38807**

8866 Concrete Naming Certificate for BirkhoffProjectiveMetric↑ **38810**

8867 Concrete Naming Certificate for BishopLocatedIntervalRefinementChain↑ **38813**

8868 Concrete Naming Certificate for RHCriticalStripWindowCertificate↑ **38816**

8869 Concrete Naming Certificate for SturmLiouville↑ **38819**

8870 Concrete Naming Certificate for CantorBorelSet↑ **38823**

8871 Concrete Naming Certificate for FellerSemigroup↑ **38826**

8872 Concrete Naming Certificate for RegularCauchyBoundedTail↑ **38829**

8873 Concrete Naming Certificate for CompactIntervalCantorBisection↑ **38832**

8874 Concrete Naming Certificate for RHMirrorOrbitTarget↑ **38835**

8875 Concrete Naming Certificate for CauchyCompletionInductionPrinciple↑ **38838**

8876 Concrete Naming Certificate for FareyIntervalRefinement↑ **38842**

8877 Concrete Naming Certificate for DyadicRealEmbeddingFaithful↑ **38845**

8878 Concrete Naming Certificate for BishopMeanValueInequality↑ **38849**

8879 Concrete Naming Certificate for LocatedUniformQuotient↑ **38852**

8880 Concrete Naming Certificate for QuarticSpectralCommonCarrier↑ **38855**

8881 Concrete Naming Certificate for LocalHalfSplitCertificate↑ **38858**

8882 Concrete Naming Certificate for CantorFan↑ **38861**

8883 Concrete Naming Certificate for UniformTailAscoli↑ **38863**

8884 Concrete Naming Certificate for RegularCauchyMedian↑ **38866**

8885 Concrete Naming Certificate for BishopRegularCauchyTailGluing↑ **38869**

8886 Concrete Naming Certificate for DyadicCompletionFunctor↑ **38872**

887 A Concrete Naming Certificate for EulerMaclaurinFiniteSum↑ **38875**

888 A Concrete Naming Certificate for FiniteRadiusContraction↑ **38878**

889 UniformRegularCompletionScale NameCert Construction **38881**

890 DeckendorfPrimeEdge NameCert Construction **38883**

891 A Concrete Naming Certificate for FiniteIntervalSignSplit↑ **38886**

892 A Concrete Naming Certificate for RegularRealFamilyDiagonal↑ **38889**

893 A Concrete Naming Certificate for BishopLocatedRealApartnessModulus↑ **38892**

894 RieszInterpolationRealUp NameCert Construction **38895**

895 A Concrete Naming Certificate for DyadicCoverModulus↑ **38898**

896 Diffeological Space NameCert Construction **38901**

897 A Concrete Naming Certificate for LocatedDedekindCutCauchyBracket↑ **38903**

898 A Concrete Naming Certificate for FiniteDataAssembly↑ **38906**

899 A Concrete Naming Certificate for FiniteInterfaceSpine↑ **38908**

900 A Concrete Naming Certificate for GrothendieckInequality↑ **38910**

901 A Concrete Naming Certificate for ParticularPointTopology↑ **38913**

902 A Concrete Naming Certificate for CompactUniformModulusNerve↑ **38916**

903 A Concrete Naming Certificate for Hypercover↑ **38919**

904 A Concrete Naming Certificate for Preuniformity↑ **38923**

905 A Concrete Naming Certificate for SchreierSims↑ **38926**

906 A Concrete Naming Certificate for FineUniformity↑ **38929**

907 A Concrete Naming Certificate for LocallyBoundedFunction↑ **38932**

908 A Concrete Naming Certificate for RegularCauchyTailFunctional↑ **38935**

909 A Concrete Naming Certificate for FiniteMembershipClassifier↑ **38938**

910 A Concrete Naming Certificate for CaratheodoryKernelConvergence↑ **38941**

911 A Concrete Naming Certificate for LanczosIteration↑ **38944**

912 A Concrete Naming Certificate for BishopBridgesRegularCompletionTheorem↑ **38947**

8913 Concrete Naming Certificate for RealCompletionCantorSieve↑ **38950**

8914 Concrete Naming Certificate for RapidCauchySubsequence↑ **38953**

8915 Concrete Naming Certificate for MetaCICStandardizationTheorem↑ **38956**

8916 Concrete Naming Certificate for ApartnessOrderedField↑ **38959**

8917 Concrete Naming Certificate for RealCompletionTailLock↑ **38962**

8918 Concrete Naming Certificate for RealCompletionIdempotence↑ **38965**

8919 Concrete Naming Certificate for DoobMeyerDecomposition↑ **38967**

8920 Concrete Naming Certificate for DomainCompletion↑ **38970**

8921 Concrete Naming Certificate for CauchyNameTruncation↑ **38973**

8922 Concrete Naming Certificate for CompleteSeparatedMetricReflector↑ **38976**

8923 Concrete Naming Certificate for IntervalExchange↑ **38979**

8924 Concrete Naming Certificate for MetricKuratowskiEmbedding↑ **38982**

8925 Concrete Naming Certificate for DyadicLocatedModulus↑ **38985**

8926 Concrete Naming Certificate for QuasiArithmeticMean↑ **38988**

8927 Concrete Naming Certificate for DyadicRealInterval↑ **38991**

8928 Concrete Naming Certificate for LocatedUniformCompletionComparison↑ **38994**

8929 Concrete Naming Certificate for BishopCompactImage↑ **38997**

8930 Concrete Naming Certificate for UniformlyLocatedSet↑ **39000**

8931 Concrete Naming Certificate for FiniteCoverIncidenceRealizer↑ **39003**

8932 Concrete Naming Certificate for LocatedDyadicUniformMesh↑ **39006**

8933 Concrete Naming Certificate for CompactModulusMeshHandoff↑ **39008**

8934 Concrete Naming Certificate for BishopLocatedRealReflection↑ **39010**

8935 Concrete Naming Certificate for RealNameEliminator↑ **39013**

8936 Concrete Naming Certificate for RiemannMappingBoundary↑ **39016**

8937 Concrete Naming Certificate for RegularLocatedCut↑ **39018**

8938 Concrete Naming Certificate for SpectralProjection↑ **39021**

8939 A Concrete Naming Certificate for CauchyRegularizationFunctor↑ **39024**

8940 A Concrete Naming Certificate for EndpointBisectionCertificate↑ **39027**

8941 A Concrete Naming Certificate for FiniteDyadicStaircase↑ **39030**

8942 A Concrete Naming Certificate for DirectedCauchyNet↑ **39033**

8943 A Concrete Naming Certificate for OrderedSemiring↑ **39037**

8944 A Concrete Naming Certificate for MetricCompletionFilter↑ **39041**

8945 A Concrete Naming Certificate for BishopLTenCompletionTerminal↑ **39044**

8946 A Concrete Naming Certificate for CauchyApartnessModulus↑ **39047**

8947 A Concrete Naming Certificate for DirichletFunction↑ **39050**

8948 A Concrete Naming Certificate for FiniteImageStreamWindow↑ **39053**

8949 A Concrete Naming Certificate for MeasurementResidueLedger↑ **39056**

8950 A Concrete Naming Certificate for CompactCauchyRealizer↑ **39059**

8951 A Concrete Naming Certificate for CauchyTailTriangle↑ **39062**

8952 A Concrete Naming Certificate for AbelIdentity↑ **39065**

8953 A Concrete Naming Certificate for FiberFarEndDigestBoundary↑ **39068**

8954 A Concrete Naming Certificate for GromovProduct↑ **39071**

8955 A Concrete Naming Certificate for SolvableGroup↑ **39074**

8956 A Concrete Naming Certificate for CauchyCompletionComparisonFunctor↑ **39076**

8957 A Concrete Naming Certificate for CauchyTailSelectorFusion↑ **39079**

8958 A Concrete Naming Certificate for KreinMilman↑ **39082**

8959 A Concrete Naming Certificate for HyperbolicObserverPhaseTransport↑ **39084**

8960 A Concrete Naming Certificate for BishopDiagonalProductCauchy↑ **39087**

8961 A Concrete Naming Certificate for PrueferTransformation↑ **39090**

8962 A Concrete Naming Certificate for BishopLocatedCauchyCriterion↑ **39093**

8963 A Concrete Naming Certificate for LocatedRealCofinalBracket↑ **39096**

8964 A Concrete Naming Certificate for NewtonForwardSeries↑ **39099**

8965 A Concrete Naming Certificate for MetaCICGeneratorClosureAlgebra$^{\uparrow}$ **39101**

8966 A Concrete Naming Certificate for SchauderFiniteNetFixedPoint$^{\uparrow}$ **39104**

8967 A Concrete Naming Certificate for RegularCauchyWindowAssociator$^{\uparrow}$ **39107**

8968 A Concrete Naming Certificate for FiniteObservationDiagonalRouter$^{\uparrow}$ **39110**

8969 A Concrete Naming Certificate for KunnethFormula$^{\uparrow}$ **39113**

8970 A Concrete Naming Certificate for PoincareDuality$^{\uparrow}$ **39116**

8971 A Concrete Naming Certificate for IntervalBisectionChain$^{\uparrow}$ **39119**

8972 A Concrete Naming Certificate for BishopRegularCauchyModulus$^{\uparrow}$ **39122**

8973 A Concrete Naming Certificate for ZeroSignatureFaithfulness$^{\uparrow}$ **39125**

8974 A Concrete Naming Certificate for ChoquetBoundary$^{\uparrow}$ **39128**

8975 A Concrete Naming Certificate for IncidenceAlgebra$^{\uparrow}$ **39130**

8976 A Concrete Naming Certificate for HyperbolicVisualMeasure$^{\uparrow}$ **39133**

8977 A Concrete Naming Certificate for RealClosedField$^{\uparrow}$ **39136**

8978 A Concrete Naming Certificate for FenchelMoreauBiconjugation$^{\uparrow}$ **39139**

8979 A Concrete Naming Certificate for AskSigGapRoute$^{\uparrow}$ **39142**

8980 A Concrete Naming Certificate for HausdorffCompletionAdjointness$^{\uparrow}$ **39145**

8981 A Concrete Naming Certificate for ApollonianGasket$^{\uparrow}$ **39148**

8982 A Concrete Naming Certificate for FiniteIntervalBisectionSelector$^{\uparrow}$ **39151**

8983 A Concrete Naming Certificate for OmegaGenerationBoundaryPacket$^{\uparrow}$ **39154**

8984 A Concrete Naming Certificate for NetAdherence$^{\uparrow}$ **39157**

8985 A Concrete Naming Certificate for KakeyaSet$^{\uparrow}$ **39160**

8986 A Concrete Naming Certificate for StreamRegularLimitSeal$^{\uparrow}$ **39163**

8987 A Concrete Naming Certificate for EpsilonDeltaModulusSelector$^{\uparrow}$ **39165**

8988 A Concrete Naming Certificate for CarlesonBox$^{\uparrow}$ **39168**

8989 A Concrete Naming Certificate for LocatedUpperLowerPair$^{\uparrow}$ **39171**

8990 A Concrete Naming Certificate for BoundedCauchyFilter$^{\uparrow}$ **39174**

8991 Concrete Naming Certificate for ConstructiveLocatedIVT$^{\uparrow}$ **39176**

8992 Concrete Naming Certificate for UniformContractionRealCompletion$^{\uparrow}$ **39179**

8993 Concrete Naming Certificate for ArtinianModule$^{\uparrow}$ **39182**
8993.1 Carrier definition 39182

8994 Concrete Naming Certificate for FinitePrefixCompactness$^{\uparrow}$ **39184**
8994.1 Carrier definition 39184

8995 Concrete Naming Certificate for PrimeParityPhase$^{\uparrow}$ **39186**

8996 Concrete Naming Certificate for LagrangeSpectrum$^{\uparrow}$ **39189**

8997 Concrete Naming Certificate for BishopLocatedDiagonalSequence$^{\uparrow}$ **39192**

8998 Concrete Naming Certificate for UniformCauchyNetBridge$^{\uparrow}$ **39195**

8999 Concrete Naming Certificate for StrongChoquetSpace$^{\uparrow}$ **39197**
8999.1 Strong Choquet Carrier 39197

9000 Concrete Naming Certificate for PrimeSpectralFingerprint$^{\uparrow}$ **39200**
9000.1 Prime Spectral Fingerprint Carrier 39200

9001 Concrete Naming Certificate for DarbouxStieltjesIntegral$^{\uparrow}$ **39203**

9002 Concrete Naming Certificate for FiniteEnergyTower$^{\uparrow}$ **39206**

9003 Concrete Naming Certificate for PartialMetric$^{\uparrow}$ **39209**

9004 Concrete Naming Certificate for RegularCauchyLocatedIntervalModulus$^{\uparrow}$ **39212**

9005 Concrete Naming Certificate for BishopCauchyReblocking$^{\uparrow}$ **39215**

9006 Concrete Naming Certificate for PrimeSkewDefectLedger$^{\uparrow}$ **39218**

9007 Concrete Naming Certificate for RegularSequenceSubsequence$^{\uparrow}$ **39221**

9008 Concrete Naming Certificate for EventualMap$^{\uparrow}$ **39224**

9009 Concrete Naming Certificate for OrliczSpace$^{\uparrow}$ **39227**

9010 Concrete Naming Certificate for DyadicMeshRefinement$^{\uparrow}$ **39230**

9011 Concrete Naming Certificate for TraceClassOperator$^{\uparrow}$ **39233**

9012 Concrete Naming Certificate for RegularVariation$^{\uparrow}$ **39236**

9013 Concrete Naming Certificate for HyperbolicSectorOverlapLedger$^{\uparrow}$ **39238**

9014 Concrete Naming Certificate for BishopSequenceSpace$^{\uparrow}$ **39241**

901.5 A Concrete Naming Certificate for BishopRegularCompletionObserver↑ **39244**

901.6 A Concrete Naming Certificate for DedekindSplitTowerLiftBoundary↑ **39247**

901.7 A Concrete Naming Certificate for FiniteOpenCoverInterval↑ **39250**

901.8 A Concrete Naming Certificate for HyperbolicSectorRefinement↑ **39253**

901.9 A Concrete Naming Certificate for ChomskyHierarchy↑ **39255**

902.0 A Concrete Naming Certificate for KuratowskiFiniteAxioms↑ **39257**

902.1 A Concrete Naming Certificate for CompletionInjectiveExtension↑ **39260**

902.2 A Concrete Naming Certificate for RegularCauchySelectorStability↑ **39263**

902.3 A Concrete Naming Certificate for SpectralSpace↑ **39265**

902.4 A Concrete Naming Certificate for FiniteModulusRegularCauchyCompletion↑ **39268**

902.5 A Concrete Naming Certificate for DyadicTruncationLPRefutationWindow↑ **39271**

902.6 Bishop Separated Real Completion NameCert Construction **39274**

902.7 A Concrete Naming Certificate for LoewnerOrder↑ **39278**

902.8 A Concrete Naming Certificate for ConstructiveMetricSpace↑ **39281**

902.9 A Concrete Naming Certificate for CantorSpaceCompletion↑ **39284**

903.0 A Concrete Naming Certificate for RegularCauchyModulusWitness↑ **39287**

903.1 A Concrete Naming Certificate for MatroidGreedy↑ **39289**

903.2 A Concrete Naming Certificate for RegularCauchyApartnessCriterion↑ **39292**

903.3 A Concrete Naming Certificate for BishopRealIntervalHalving↑ **39295**

903.4 A Concrete Naming Certificate for CauchyFilterCompletionUniversal↑ **39297**

903.5 A Concrete Naming Certificate for SeparatedCompletionUniversalArrow↑ **39300**

903.6 A Concrete Naming Certificate for SeparatedCauchyFilter↑ **39303**

903.7 A Concrete Naming Certificate for OreLocalization↑ **39306**

903.8 A Concrete Naming Certificate for RightBlockFluxReservoirCoefficient↑ **39309**

903.9 A Concrete Naming Certificate for ConvexFunction↑ **39312**

904.0 A Concrete Naming Certificate for FaulhaberFormula↑ **39315**

9041 Concrete Naming Certificate for SeparatedCauchySequence↑ **39318**

9042 Concrete Naming Certificate for RealPrefixDiagonal↑ **39321**

9043 Concrete Naming Certificate for ZeroFibreNormalSheet↑ **39324**

9044 Concrete Naming Certificate for AbsoluteContinuousCurve↑ **39327**

9045 Concrete Naming Certificate for DyadicEmbeddingRegularSequence↑ **39330**

9046 Concrete Naming Certificate for FixedHalfZeroWitness↑ **39333**

9047 Bishop Separated Cauchy Completion Idempotence NameCert Construction 39336

9048 Concrete Naming Certificate for BurnsideLemma↑ **39339**

9049 Concrete Naming Certificate for RegularCauchyRateTransformer↑ **39342**

9050 Concrete Naming Certificate for CylindricalAlgebraicDecomposition↑ **39345**

9051 Concrete Naming Certificate for WindowSixBetaSeed↑ **39348**

9052 Concrete Naming Certificate for CountableCompletionDiagonalRepresentative↑ **39350**

9053 Concrete Naming Certificate for FiniteWindowUniformCompletion↑ **39353**

9054 Concrete Naming Certificate for BanachFixedPointContraction↑ **39356**

9055 Concrete Naming Certificate for Transportability↑ **39359**

9056 Concrete Naming Certificate for LocalGlobalZeroExclusionCertificate↑ **39365**

9057 Concrete Naming Certificate for SequentialApartnessCompletionBoundary↑ **39367**

9058 Concrete Naming Certificate for RiemannSampler↑ **39370**

9059 Concrete Naming Certificate for CantorRealCompletionKernel↑ **39372**

9060 Concrete Naming Certificate for PhaseObserverShellRealization↑ **39375**

9061 Concrete Naming Certificate for FiniteHerbrandDisjunction↑ **39378**

9062 Concrete Naming Certificate for BarInductionCauchyModulus↑ **39381**

9063 Concrete Naming Certificate for CofinalStreamReindexing↑ **39383**

9064 Concrete Naming Certificate for FanCauchyModulus↑ **39386**

9065 Concrete Naming Certificate for LocatedIntervalSampling↑ **39389**

9066 Concrete Naming Certificate for InfoGenerationTrace↑ **39391**

906.7 A Concrete Naming Certificate for CauchyProductCompletionSeal↑ **39394**

906.8 A Concrete Naming Certificate for CarlesonMeasure↑ **39397**

906.9 Cauchy Uniform Continuation NameCert Construction **39400**

907.0 A Concrete Naming Certificate for KKMLemma↑ **39403**

907.1 A Concrete Naming Certificate for HenkinModel↑ **39406**

9071.1 Finite Henkin packet 39406

9071.2 Local certificate surface 39406

907.2 A Concrete Naming Certificate for FiniteIntervalMeshRefinement↑ **39409**

907.3 A Concrete Naming Certificate for CompactIntervalLipschitzEnvelope↑ **39412**

907.4 A Concrete Naming Certificate for BorelMeasure↑ **39415**

907.5 A Concrete Naming Certificate for SolenoidPhase↑ **39418**

907.6 A Concrete Naming Certificate for FrobeniusCharacterFormula↑ **39421**

907.7 A Concrete Naming Certificate for FloydWarshallClosure↑ **39424**

907.8 A Concrete Naming Certificate for FiniteNetSelectionPrinciple↑ **39427**

907.9 A Concrete Naming Certificate for DyadicReciprocalNewton↑ **39430**

908.0 A Concrete Naming Certificate for SerreSwanFiniteProjective↑ **39433**

908.1 A Concrete Naming Certificate for LocatedApartnessSplit↑ **39435**

908.2 A Concrete Naming Certificate for SequentialCompactInterval↑ **39439**

908.3 A Concrete Naming Certificate for RegularCauchySumCriterion↑ **39442**

908.4 A Concrete Naming Certificate for DyadicCauchySandwich↑ **39445**

908.5 A Concrete Naming Certificate for DyadicModulusComparisonBoundary↑ **39448**

908.6 A Concrete Naming Certificate for GaussianElimination↑ **39451**

908.7 A Concrete Naming Certificate for SternBrocotRealCompletion↑ **39454**

908.8 A Concrete Naming Certificate for GelfandNaimark↑ **39457**

908.9 A Concrete Naming Certificate for UnscentedKalmanFilter↑ **39460**

909.0 A Concrete Naming Certificate for FiniteHellySelection↑ **39463**

909.1 Buckingham Pi NameCert Construction **39466**

90 Axis-Zeckendorf Running Coordinate NameCert Construction **39469**

90 A Concrete Naming Certificate for SingletonMetric$^{\uparrow}$ **39472**

90 A Concrete Naming Certificate for UniformOscillationModulus$^{\uparrow}$ **39474**

90 A Concrete Naming Certificate for RegularCauchyContinuityModulus$^{\uparrow}$ **39477**

90 A Concrete Naming Certificate for BirkhoffPolytope$^{\uparrow}$ **39479**

90 A Concrete Naming Certificate for SchauderProjection$^{\uparrow}$ **39482**

90 A Concrete Naming Certificate for RegularCauchyInterleavingMerge$^{\uparrow}$ **39485**

90 A Concrete Naming Certificate for StreamShuffle$^{\uparrow}$ **39488**

91 A Concrete Naming Certificate for CommonCarrierCompletionBoundary$^{\uparrow}$ **39491**

91 A Concrete Naming Certificate for FiniteFourierTailProjector$^{\uparrow}$ **39494**

91 A Concrete Naming Certificate for FiniteObservationBoundary$^{\uparrow}$ **39497**

91 A Concrete Naming Certificate for MorseHomology$^{\uparrow}$ **39500**

91 A Concrete Naming Certificate for NuclearSpace$^{\uparrow}$ **39503**

91 A Concrete Naming Certificate for CauchySequenceMorphism$^{\uparrow}$ **39506**

91 A Concrete Naming Certificate for BishopRealNameRealizer$^{\uparrow}$ **39509**

91 A Concrete Naming Certificate for ParacompactSpace$^{\uparrow}$ **39512**

91 A Concrete Naming Certificate for RegulatedCauchyStepEnvelope$^{\uparrow}$ **39515**

91 A Concrete Naming Certificate for CauchyCompactnessCriterion$^{\uparrow}$ **39518**

91 A Concrete Naming Certificate for NearestPointProjection$^{\uparrow}$ **39521**

91 A Concrete Naming Certificate for MetricApartnessCompletion$^{\uparrow}$ **39524**

91 A Concrete Naming Certificate for ApartnessCompletionReflector$^{\uparrow}$ **39527**

91 A Concrete Naming Certificate for StreamNameRealizer$^{\uparrow}$ **39530**

91 A Concrete Naming Certificate for ObserverCenteredConstructorCertificate$^{\uparrow}$ **39533**

91 A Concrete Naming Certificate for FiniteCoverMesh$^{\uparrow}$ **39536**

91 Menger Sponge NameCert Construction **39539**

91 A Concrete Naming Certificate for TransferFunctionStateSpaceRealization$^{\uparrow}$ **39542**

9118. A Concrete Naming Certificate for CauchyIdeal↑ **39544**

9119. Analytic Tail Budget NameCert Construction **39547**

9120. A Concrete Naming Certificate for ZeckendorfReadoutCoefficientStrata↑ **39550**

9121. A Concrete Naming Certificate for BishopCompactCompletionExtension↑ **39553**

9122. A Concrete Naming Certificate for CompactUniformNetSelection↑ **39556**

9123. A Concrete Naming Certificate for OstrowskiCompletionUniqueness↑ **39559**

9124. A Concrete Naming Certificate for CauchyCompletionReflectionUnit↑ **39562**

9125. A Concrete Naming Certificate for BishopCompletionFunctor↑ **39565**

9126. A Concrete Naming Certificate for CompactModulusPartitionLadder↑ **39568**

9127. A Concrete Naming Certificate for CauchyKovalevskaya↑ **39571**

9128. A Concrete Naming Certificate for RegularCauchyBoundedness↑ **39574**

9128.1. Regular Cauchy boundedness carrier 39574

9129. A Concrete Naming Certificate for CompletionReflector↑ **39577**

9130. A Concrete Naming Certificate for FinitePrefixBarModulusWindow↑ **39580**

9131. A Concrete Naming Certificate for CauchyCompletionStabilizer↑ **39583**

9132. A Concrete Naming Certificate for DarbouxGapRefinement↑ **39586**

9133. A Concrete Naming Certificate for RationalIntervalHull↑ **39589**

9134. A Concrete Naming Certificate for FiniteComparisonBoundary↑ **39592**

9135. A Concrete Naming Certificate for LogarithmicSectionDynamics↑ **39595**

9136. A Concrete Naming Certificate for LocatedSetoid↑ **39598**

9137. A Concrete Naming Certificate for BishopCauchyLocatedLimit↑ **39601**

9138. A Concrete Naming Certificate for RuleOneTenProductionQueue↑ **39603**

9139. A Concrete Naming Certificate for ArithmeticSpectralStringCertificate↑ **39606**

9140. A Concrete Naming Certificate for CompactZeroDimensionalMetric↑ **39608**

9141. A Concrete Naming Certificate for AronszajnAscoli↑ **39611**

9142. A Concrete Naming Certificate for ZeroLedgerRigidity↑ **39613**

9143. A Concrete Naming Certificate for CompactUniformContinuityExtractor↑ **39616**

9144 A Concrete Naming Certificate for HyperbolicBusemannBoundaryBudget↑ 39618

9145 A Concrete Naming Certificate for McNaughtonPiecewiseLinear↑ 39621

9146 A Concrete Naming Certificate for LocatedCauchyModulusExtraction↑ 39624

9147 A Concrete Naming Certificate for CauchyNameModulusExtraction↑ 39627

9148 A Concrete Naming Certificate for SequentiallyTotallyBounded↑ 39630

9149 A Concrete Naming Certificate for InnovationSolderingMetricInterface↑ 39633

9150 A Concrete Naming Certificate for Window6 Forward Deliverable Shapes 39636
9150.0 Closed-Zero Flattening Count Identity . . . 39636
9150.1 Seam-Gluing Parity . . . 39637
9150.2 Saturation-Deficit Extremality . . . 39637
9150.3 Live-Tail Congruence . . . 39638
9150.4 Signed Involution Fixed Point . . . 39638
9150.5 Zero-Mark Galois Closure . . . 39639

9151 A Finite Window6 Skew/Neutral Certificate for the RH No-Neutral-Leak Obligation 39640
9151.0 The Window6 Mirror Involution . . . 39640
9151.1 Fixed Points Are the Neutral Support . . . 39640
9151.2 The Skew/Neutral Split Certificate . . . 39641
9151.3 Relation to the No-Neutral-Leak Obligation . . . 39641

9152 The Fibonacci–Polya Jensen Kernel 39642
9152.0 Degree-Two Discriminant Window . . . 39642
9152.1 Honest Scope . . . 39643

9153 The Fibonacci–Polya Jensen Deformation 39645
9153.0 Degree-Two Lambda Discriminant . . . 39645
9153.1 Honest Scope . . . 39646

9154 The Zeckendorf Mobius Sieve Certificate 39647
9154.0 Squarefree Zeckendorf-Hamming Polynomials . . . 39647
9154.1 Finite Mertens Cancellation . . . 39647
9154.2 Honest Scope . . . 39648

9155 Fibonacci-Cube Ihara Zeta 39649
9155.0 Finite Bass Determinants . . . 39649
9155.1 Honest Scope . . . 39650

9156 The Finite-Window Fibonacci–RH Program 39651
9156.0 The Certificates . . . 39651
9156.1 The Program Boundary . . . 39652

9157 A Concrete Naming Certificate for Euler's Pentagonal Number Theorem 39653

9158 A Concrete Naming Certificate for Euler's Distinct–Odd Partition Identity 39656

9159 A Concrete Naming Certificate for Euler's Partition Pentagonal Recurrence 39659

9160 A Concrete Naming Certificate for IshiharaBoundednessFiniteTest$^{\uparrow}$ **39662**

9161 A Concrete Naming Certificate for FiniteMetricChainBound$^{\uparrow}$ **39665**

9162 A Concrete Naming Certificate for HorosphereBusemannFlowBudget$^{\uparrow}$ **39668**

9163 A Concrete Naming Certificate for RealConvergenceRateExtractor$^{\uparrow}$ **39671**

9164 A Concrete Naming Certificate for PrimaryDecomposition$^{\uparrow}$ **39674**

9165 A Concrete Naming Certificate for CompactOscillationBudget$^{\uparrow}$ **39677**

9166 A Concrete Naming Certificate for CompactConvergenceTheorem$^{\uparrow}$ **39680**

9167 A Concrete Naming Certificate for RegularCauchyDiagonalLimitSeal$^{\uparrow}$ **39683**

9168 A Concrete Naming Certificate for CauchyFilterTelescope$^{\uparrow}$ **39686**

9169 A Concrete Naming Certificate for TotallyBoundedInterval$^{\uparrow}$ **39689**

9170 A Concrete Naming Certificate for LebesgueNumberMesh$^{\uparrow}$ **39692**

9171 A Concrete Naming Certificate for DirectedNetTailWindowCompletion$^{\uparrow}$ **39695**

9172 A Concrete Naming Certificate for RealTailBudgetHandoff$^{\uparrow}$ **39698**

9173 A Concrete Naming Certificate for BendixsonDulac$^{\uparrow}$ **39701**

9174 A Concrete Naming Certificate for RegularRealMorphism$^{\uparrow}$ **39704**

9175 A Concrete Naming Certificate for LowerVietorisCompletion$^{\uparrow}$ **39706**

9176 A Concrete Naming Certificate for GaussMap$^{\uparrow}$ **39709**

9177 A Concrete Naming Certificate for BishopNestedInterval$^{\uparrow}$ **39712**

9178 A Concrete Naming Certificate for CompactRealIntervalModulusWindow$^{\uparrow}$ **39715**

9179 A Concrete Naming Certificate for ParallelogramLaw$^{\uparrow}$ **39718**

9180 A Concrete Naming Certificate for BesicovitchDifferentiationWindow$^{\uparrow}$ **39721**

9181 A Compact Real Interval Choice-Guard Certificate for CompactRealIntervalChoiceGuard$^{\uparrow}$ **39724**

9182 A Concrete Naming Certificate for RegularCauchyTailProductBound$^{\uparrow}$ **39726**

9183 A Concrete Naming Certificate for ChromaticPolynomial$^{\uparrow}$ **39729**

9184 A Concrete Naming Certificate for QuantumGRSectionBundle↑ 39732

9185 A Concrete Naming Certificate for BoundedCauchySubsequenceModulus↑ 39735

9186 A Concrete Naming Certificate for BishopCauchySpace↑ 39738

9187 A Concrete Naming Certificate for TukeyDepth↑ 39741

9188 A Concrete Naming Certificate for HyperbolicPhaseLens↑ 39744

9189 A Concrete Naming Certificate for RegularCauchyInverseGuard↑ 39747

9190 A Concrete Naming Certificate for LocaleFrame↑ 39750

9191 A Concrete Naming Certificate for the Fibonacci-Window Weil Positivity Matrix 39753

VI Concrete Hardening 39756

9192 Internal Domain Judgments and Signature Token Reflection 39757

9193 Internalized Concrete Gaps and Globalize Soundness 39759

9194 Unary Continuation Commutativity and the Additive Name 39764

VII Proof Obligations 39767

9195 Domain Policy and Unary Instance 39768
9195.1 Domain Policy and Transport Obligations . . . 39768
9195.2 Concrete Domain Policy for the Unary Source . . . 39769

9196 Package Token Policy, Uniqueness, and Reflection 39772
9196.1 Package Token Policy and Reflection . . . 39772
9196.2 Inductive Package-Token Policy for Signature Packages . . . 39774
9196.3 Token-Introduction Uniqueness and Package Reflection . . . 39775
9196.4 Token-Uniqueness Modes . . . 39777

9197 Gap Policy 39782
9197.1 Consolidated Signature Gap Policy . . . 39782

9198 Package-Sameness Design and Base Reflection 39791
9198.1 Concrete Package-Sameness Design Contract . . . 39791
9198.2 Base Reflection Schema . . . 39793
9198.3 Optional Closure Reflection Target . . . 39794

9199 Exact Globalize Base Contract 39798
9199.1 Exact Concrete Signature Globalize Under Explicit Policies . . . 39798
9199.2 Exactness Export Contract . . . 39799
9199.3 Checker-Friendly Exactness Predicate . . . 39801

9199.4 Exact Globalize Base Contract . . . 39803

9200 Unary Shift and Commutativity 39808
9200.1 Unary Commutativity Proof Obligations . . . 39808
9200.2 Unary Shift and the Additive License . . . 39809
9200.3 Unary Right-Shift: Micro-Lemmas for the Additive License . . . 39810

9201 Lean Scaffold Contract 39814
9201.1 Lean-Style Scaffold Contract . . . 39814
9201.2 Scaffold excerpt . . . 39814
9201.3 Interpretation . . . 39814
9201.4 Lean Verification Status . . . 39815

VIII Implementation-Facing Contract 39816

9202 Formalization Implementation Contract 39817
9202.1 Layer separation . . . 39817
9202.2 Module targets . . . 39817
9202.3 Public theorem names . . . 39817
9202.4 Implementation choices . . . 39818
9202.5 Proof-sprint order . . . 39818
9202.6 Forbidden public shortcuts . . . 39819

IX Acceptance Discipline 39820

9203 Closure and Verification Discipline 39821
9203.1 Theoretical closure axis . . . 39821
9203.2 External correspondence axis . . . 39822
9203.3 Formal verification axis . . . 39823
9203.4 The verified gate . . . 39823
9203.5 Marker-to-verification correspondence . . . 39824
9203.6 Per-chapter closurestatus block . . . 39825
9203.7 Upgrade obligations . . . 39825
9203.8 Reuse, composition, and retroactive promotion . . . 39826
9203.9 The discipline as research-plan language . . . 39826

9204 Standard Bridge Protocol 39828
9204.1 The six fields of a standard bridge . . . 39828
9204.2 The no-host-leak field as object-level axiom-purity . . . 39829
9204.3 Quotient and closure discipline as bridge prerequisites . . . 39829
9204.4 Conservativity in the presence of theorem preservation . . . 39830
9204.5 Worked sketch: integers . . . 39831
9204.6 Bridge prerequisites table . . . 39831

9205 Cannot-Claim Registry and Failure Witnesses 39832
9205.1 The discipline of negative claims . . . 39832
9205.2 Failure witness type . . . 39832
9205.3 Cannot-claim registry . . . 39833
9205.4 Closure reading convention . . . 39835
9205.5 Anti-confusion register . . . 39835
9205.6 Failure witness as audit input . . . 39835

9206 Closure Reading Convention 39837
9206.1 Read both axes, separately . . . 39837
9206.2 Read closure as a paper claim . . . 39837
9206.3 Read verification as an audit claim . . . 39837
9206.4 Per-object orientation table . . . 39837

X Ground Compiler Discipline 39840

9207 Ground Event Flow 39841
9207.1 Display alphabet and raw events . . . 39841
9207.2 Event flows and boundary information . . . 39842
9207.3 No-hidden-input discipline . . . 39843
9207.4 Recognized derived flows . . . 39844
9207.5 Accepted object flows and the ground theorem . . . 39844
9207.6 Worked event-flow examples . . . 39846
9207.7 Conservativity and closing observations . . . 39847

9208 Zeckendorf-safe Channel Encoding 39849
9208.1 Body encoding and event terminator . . . 39850
9208.2 Legal event code and channel stream . . . 39851
9208.3 Decoders and round-trip theorems . . . 39852
9208.4 Streaming behavior and host-structure independence . . . 39854
9208.5 Conservativity and closing observations . . . 39855

9209 Source-Level Events and Channel-Level Bits 39857
9209.1 Source layer and channel layer . . . 39857
9209.2 No channel-level source rewrite . . . 39858
9209.3 Decode-normalize-compile pipeline . . . 39859
9209.4 Ledger preservation and classifier quotient . . . 39860
9209.5 Encoding and decoding examples for ledger flows . . . 39861
9209.6 Channel substrings versus source events . . . 39862
9209.7 Layer separation theorem and closing observations . . . 39863

9210 Package Structure Is Generated, Not Input 39864
9210.1 Package recognition on event flows . . . 39864
9210.2 Roles, seals, and ledgers . . . 39866
9210.3 Compile-decode invariance . . . 39868
9210.4 Classified packages as quotients . . . 39869
9210.5 Recognition from channel code . . . 39869

9210.6 Conservativity and derived-interface boundary . . . 39871

9211 NameCert Flow Recognition 39872
9211.1 NameCert recognition on event flows . . . 39872
9211.2 Field subflows and seals . . . 39874
9211.3 Name licensing . . . 39875
9211.4 Ledger completeness . . . 39876
9211.5 NameCert code and channel recognition . . . 39877
9211.6 Field soundness events . . . 39878
9211.7 Conservativity and examples . . . 39879

9212 DerivCert and AcceptanceGate Flow Recognition 39882
9212.1 Strength and DerivCert recognition . . . 39882
9212.2 Six fields and field soundness . . . 39884
9212.3 Acceptance flows . . . 39886
9212.4 Recognition from code and round trips . . . 39888
9212.5 Strength monotonicity and reuse . . . 39888
9212.6 Conservativity and examples . . . 39890

9213 Theorem Flow Recognition 39892
9213.1 Theorem recognition . . . 39892
9213.2 Theorem roles . . . 39894
9213.3 Proof, certificate, ledger, status, and site soundness . . . 39895
9213.4 Theorem code and quotients . . . 39897
9213.5 Recognition from code . . . 39898
9213.6 Acceptance and seals . . . 39899
9213.7 Conservativity and examples . . . 39900

9214 Chapter Flow Recognition 39902
9214.1 Chapter recognition . . . 39902
9214.2 Chapter roles . . . 39904
9214.3 Soundness of chapter structure . . . 39905
9214.4 Chapter code and quotients . . . 39907
9214.5 Recognition from code and readable views . . . 39908
9214.6 Dependencies and seals . . . 39910
9214.7 Conservativity and examples . . . 39910

9215 Recognizer Flows 39912
9215.1 Recognizer candidates and roles . . . 39912
9215.2 Recognizer certificates . . . 39913
9215.3 Conservativity and hierarchy . . . 39915
9215.4 Self-description and self-hosting . . . 39917
9215.5 Examples and conservativity . . . 39919

9216 Self-Hosting Compiler Flow 39921
9216.1 Compiler candidates and behavior . . . 39921
9216.2 Compiler certificates . . . 39922
9216.3 Bootstrap boundary . . . 39924

9216.4 Self-compilation and self-hosting . . . 39924
9216.5 Bootstrap ladder . . . 39926
9216.6 Compiler theorem packages . . . 39927
9216.7 Conservativity . . . 39929

9217 Semantic Motif Analysis 39930
9217.1 Motif recognition . . . 39930
9217.2 Basic motif forms . . . 39932
9217.3 Code behavior . . . 39933
9217.4 Profiles, overlap, and prefixes . . . 39934
9217.5 Motif metrics . . . 39935
9217.6 Examples . . . 39936
9217.7 Limits of motif analysis . . . 39937

9218 First Case Studies: Finite Repetition, Add/Fold, List, and Completion 39939
9218.0 Case-study flows . . . 39939
9218.1 Finite repetition skeleton . . . 39940
9218.2 Nat-like skeleton . . . 39941
9218.3 Add and fold skeletons . . . 39942
9218.4 List-like skeleton . . . 39943
9218.5 Completion and FullAxis skeleton . . . 39944
9218.6 Cross-theory comparison . . . 39946
9218.7 Motif reports . . . 39947
9218.8 Conservativity . . . 39948

9219 Theory Metrics and Flow Invariants 39949
9219.1 Admissible metric data . . . 39949
9219.2 Prefix profiles . . . 39950
9219.3 Motif profiles and counts . . . 39951
9219.4 Normal addresses and distances . . . 39952
9219.5 Theory-flow distance and reports . . . 39953
9219.6 Examples and cannot-claims . . . 39955
9219.7 Conservativity . . . 39956

9220 Higher-Layer Case Studies: Completion, Real, Topology, Geometry, Category, and Homology 39957
9220.0 Higher-layer case-study flows . . . 39957
9220.1 Completion and Real-like flows . . . 39958
9220.2 Metric and topology flows . . . 39959
9220.3 Phase, circle, and complex flows . . . 39960
9220.4 Category, functor, and natural transformation motifs . . . 39962
9220.5 Homology as boundary-quotient flow . . . 39963
9220.6 Cross-domain comparison . . . 39963
9220.7 Reports, limits, and conservativity . . . 39964

9221 Compiler Analysis Pipeline 39966
9221.1 Inputs and protocol recognition . . . 39966
9221.2 Pipeline stages . . . 39968
9221.3 Pipeline correctness . . . 39970
9221.4 Report discipline . . . 39971
9221.5 Failure handling . . . 39972
9221.6 Theorem and chapter code analysis . . . 39972
9221.7 Examples and determinacy . . . 39973

9222 Main Theorems and Cannot-Claim Registry 39975
9222.0 Compiler state and hidden input . . . 39975
9222.2 Channel and layer separation . . . 39976
9222.3 Recognized structures and exports . . . 39977
9222.4 Theorem, chapter, and quotient layers . . . 39979
9222.5 Recognizers, compilers, motifs, and metrics . . . 39981
9222.6 Cannot-claim registry . . . 39982
9222.7 Global conservativity and final theorem . . . 39983

9223 Implementation Interface and Verification Targets 39986
9223.1 Implementation layer and formal surface . . . 39986
9223.2 Core implementation modules . . . 39987
9223.3 Channel compiler specification . . . 39989
9223.4 Executable encoder and decoder obligations . . . 39990
9223.5 Public recognition interfaces . . . 39992
9223.6 No-host-leak audit . . . 39993
9223.7 Lean targets and reference executable . . . 39993
9223.8 Implementation reports and adequacy . . . 39995

9224 Minimal Executable Prototype Specification 39997
9224.1 Scope and display representations . . . 39997
9224.2 Encoding and decoding specification . . . 39999
9224.3 Prototype test suite . . . 40000
9224.4 Prototype outputs . . . 40001
9224.5 Prototype correctness . . . 40002
9224.6 Prototype limitations and CLI . . . 40003
9224.7 Output formats and audit . . . 40004
9224.8 Example sessions and conservativity . . . 40005

9225 Source-Event Reporting Prototype 40007
9225.1 Source-event reports . . . 40007
9225.2 Warnings and cannot-claims . . . 40009
9225.3 Report format and examples . . . 40010
9225.4 Soundness and display policy . . . 40011
9225.5 No-host-leak and adequacy . . . 40013
9225.6 Full example report . . . 40014

9226 Source-Level Normalizer Ledger Prototype 40016
9226.1 Adjacent pairs and candidates . . . 40016
9226.2 Candidate policies . . . 40019
9226.3 Recognized normalizer ledgers . . . 40020
9226.4 Examples and reports . . . 40021
9226.5 Soundness, conservativity, and adequacy . . . 40023
9226.6 Full reports . . . 40024

9227 Motif-Report Prototype 40026
9227.1 Motif candidates and recognition . . . 40026
9227.2 Basic motifs . . . 40028
9227.3 Motif reports . . . 40029
9227.4 P_3 pipeline . . . 40030
9227.5 Examples . . . 40031
9227.6 Motif counts . . . 40032
9227.7 Soundness, adequacy, and conservativity . . . 40033
9227.8 Full report examples . . . 40035

9228 Metric-Report Prototype 40036
9228.1 Metric protocols and admissibility . . . 40036
9228.2 Basic metrics . . . 40038
9228.3 Comparison metrics . . . 40039
9228.4 Metric reports . . . 40041
9228.5 Metric claims and cannot-claims . . . 40041
9228.6 P_4 pipeline and examples . . . 40042
9228.7 Adequacy and conservativity . . . 40043
9228.8 Full reports . . . 40044

9229 Package and NameCert Recognition Prototype 40046
9229.1 P_5 prototype interface . . . 40046
9229.2 Package-flow recognition . . . 40047
9229.3 NameCert-flow recognition . . . 40049
9229.4 Package-sourced NameCert flows . . . 40051
9229.5 P_5 reports and soundness . . . 40051
9229.6 Adequacy . . . 40053
9229.7 Examples . . . 40054
9229.8 Conservativity . . . 40054

9230 DerivCert and AcceptGate Recognition Prototype 40056
9230.1 P_6 prototype interface . . . 40056
9230.2 Strength-flow recognition . . . 40057
9230.3 DerivCert-flow recognition . . . 40058
9230.4 Field soundness . . . 40060
9230.5 AcceptGate-flow recognition . . . 40060
9230.6 Accepted object code . . . 40062
9230.7 Strength monotonicity and promotion . . . 40062
9230.8 Reports and soundness . . . 40063
9230.9 Adequacy and examples . . . 40065

9230.10 Conservativity . . . 40066

9231 Theorem and Proof Recognition Prototype 40067
9231.1 P_7 prototype interface . . . 40067
9231.2 Statement and dependency recognition . . . 40068
9231.3 Proof-flow recognition and checking . . . 40069
9231.4 Theorem-flow recognition . . . 40071
9231.5 Theorem-flow soundness and code . . . 40072
9231.6 Abstract propositions and accepted theorem flows . . . 40073
9231.7 Reports and soundness . . . 40074
9231.8 Adequacy and examples . . . 40075
9231.9 Conservativity . . . 40076

9232 Chapter and Manuscript Recognition Prototype 40078
9232.1 P_8 prototype interface . . . 40078
9232.2 Definition and lemma recognition . . . 40079
9232.3 Chapter-flow recognition . . . 40080
9232.4 Chapter soundness and code . . . 40081
9232.5 Manuscript-flow recognition . . . 40083
9232.6 Manuscript soundness and code . . . 40084
9232.7 Reports and soundness . . . 40085
9232.8 Adequacy . . . 40086
9232.9 Examples . . . 40087
9232.10 Conservativity . . . 40087
9232.11 Summary . . . 40088

9233 Self-Hosting Compiler and Global Verification Prototype 40090
9233.1 P_9 prototype interface . . . 40090
9233.2 Compiler certificate-flow . . . 40091
9233.3 Compiler behavior-flow . . . 40092
9233.4 Bootstrap obligations . . . 40093
9233.5 Self-compilation . . . 40094
9233.6 Compiler behavior equivalence . . . 40094
9233.7 Self-hosting compiler-flow . . . 40095
9233.8 Global verification-flow . . . 40096
9233.9 Reports and soundness . . . 40096
9233.10 Adequacy . . . 40097
9233.11 Examples . . . 40098
9233.12 Conservativity and ladder theorem . . . 40099
9233.13 Summary . . . 40100

9234 Programmatic Realization of the Prototype Ladder 40101
9234.1 Executable base . . . 40101
9234.2 Report and metric layers . . . 40102
9234.3 Recognizer virtual machine . . . 40103
9234.4 Certificates, proofs, and acceptance . . . 40104
9234.5 Self-hosting route . . . 40105
9234.6 Implementation architecture . . . 40105

9234.7 Deliverable ladder . . . 40106
9234.8 Summary . . . 40107

9235 Self-Compilation Surface 40108
9235.1 The L1-L4 Stratification . . . 40108
9235.2 Internal Manifest . . . 40109
9235.3 MetaCIC Reconstruction . . . 40111
9235.3.1 Encoding Overview . . . 40112
9235.3.2 Syntax Constructor Table . . . 40112
9235.3.3 Paper vs Mechanization . . . 40112
9235.3.4 Host Fragment Coverage . . . 40113
9235.3.5 Correctness Claims . . . 40113
9235.3.6 Representation vs Mockup . . . 40114
9235.3.7 Positioning Within BEDC . . . 40114
9235.3.8 Stratification Recap . . . 40114
9235.3.9 Fragments and Limitations . . . 40115
9235.3.10 Typing Rules . . . 40115
9235.3.11 Typing Inference Rules Table . . . 40115
9235.3.12 Typing Relation vs Host CIC . . . 40115
9235.3.13 Context Mechanics . . . 40116
9235.3.14 Reduction Layer Summary . . . 40116
9235.3.15 Reduction Relations . . . 40117
9235.3.16 No Eta-Conversion . . . 40117
9235.3.17 Self-Compilation Completeness . . . 40118
9235.3.18 Host CIC Relation . . . 40118
9235.3.19 Gödel-Tarski Boundary . . . 40119
9235.3.20 Host Internalization . . . 40119
9235.3.21 Axiom Purity Guarantee . . . 40119
9235.3.22 Pure CIC Fragment . . . 40120
9235.3.23 Encoding Choice . . . 40120
9235.3.24 Closure Discipline . . . 40120
9235.3.25 Shift Mechanism Detail . . . 40121
9235.3.26 Substitution Operation . . . 40121
9235.3.27 Glossary . . . 40121
9235.3.28 Construction Style . . . 40122
9235.3.29 Derivation Style . . . 40122
9235.3.30 Induction Style . . . 40123
9235.3.31 Namespace Rationale . . . 40123
9235.3.32 Lean Module Topology . . . 40123
9235.3.33 Tactic Discipline . . . 40124
9235.3.34 Build and Audit Process . . . 40124
9235.3.35 Axiom-Free Claim . . . 40124
9235.3.36 Internal Manifest Coverage . . . 40125
9235.3.37 Shift-Aware Metatheory Progress . . . 40125
9235.3.38 Substrate Stack . . . 40126
9235.3.39 Substrate Components Table . . . 40128
9235.3.40 Current State of the Reconstruction . . . 40128

9235.3.4 Evolution Trajectory . . . 40130
9235.3.4.2 Open Targets . . . 40130
9235.3.4.3 Results Overview . . . 40131
9235.3.4.4 Reconstruction Summary . . . 40131
9235.4 Representational Adequacy for the MetaCIC Fragment . . . 40134
9235.4.1 Adequacy Criterion . . . 40134
9235.4.2 Mechanized Witnesses . . . 40134
9235.4.3 Substrate Differences . . . 40135
9235.4.4 Coverage Boundary . . . 40136
9235.5 Bounds: What This Surface Does Not Claim . . . 40136
9235.6 Reflection Bridge to the Host Kernel . . . 40142
9235.6.1 Bridge Form . . . 40142
9235.6.2 Formal Obstacles . . . 40142
9235.6.3 Tractable Subgoals . . . 40143
9235.6.4 Position . . . 40143
9235.7 Shift-Aware Substrate Design . . . 40143
9235.7.1 Substrate Alternatives Surveyed . . . 40143
9235.7.2 Why De Bruijn Alone Is Insufficient . . . 40144
9235.7.3 Mechanized Properties . . . 40145
9235.7.4 Open Algebraic Gap . . . 40146
9235.7.5 Comparison to the Host CIC . . . 40146
9235.8 MetaCIC Confluence Status . . . 40146
9235.8.1 Setting . . . 40147
9235.8.2 Atom-Shape Diamond . . . 40147
9235.8.3 Application, Lambda, and Pi Shapes . . . 40148
9235.8.4 Strategy Notes . . . 40149
9235.8.5 BEDC Scope . . . 40150
9235.9 Normalization Status for the MetaCIC Fragment . . . 40150
9235.9.1 Bounded Normalizer . . . 40151
9235.9.2 Closure Preservation Under Reduction . . . 40151
9235.9.3 Strong Normalization . . . 40152
9235.9.4 Practical Impact . . . 40152
9235.9.5 Comparison with the Host Calculus . . . 40153
9235.9.6 Status Summary . . . 40154
9235.10 MetaCIC Metatheory Roadmap . . . 40154
9235.10.1 Setting . . . 40154
9235.10.2 Mechanized Surface . . . 40155
9235.10.3 Open Statements . . . 40155
9235.10.4 Why These Gaps Are Specific . . . 40156
9235.10.5 Reading Guide . . . 40157
9235.11 BHist Substrate Bridge . . . 40157
9235.11.1 What BHist Is . . . 40157
9235.11.2 Why Bridge BHist to Term . . . 40158
9235.11.3 The Mechanized Embedding . . . 40158
9235.11.4 What the Bridge Gives . . . 40159
9235.11.5 Unproved Boundaries . . . 40159
9235.12 Substrate Stack Summary . . . 40161

9235.12 Layered View . . . 40161
9235.12.1 Layer 0: Term Syntax . . . 40161
9235.12.2 Layer 1: Substrate Mechanics . . . 40162
9235.12.3 Layer 2: Reduction . . . 40162
9235.12.4 Layer 3: Typing . . . 40162
9235.12.5 Layer 4: Metatheory . . . 40162
9235.12.6 Layer 5: Examples and Embeddings . . . 40163
9235.12.7 Cross-Cutting Invariants . . . 40163
9235.12.8 Open Seam . . . 40163
9235.13 Typed Examples Gallery . . . 40163
9235.13.1 Why a gallery . . . 40164
9235.13.2 Sibling index . . . 40164
9235.13.3 Mechanized witnesses . . . 40164
9235.13.4 Boundary of the gallery . . . 40170
9235.13.5 Contributor discipline . . . 40171
9235.14 Metatheory Milestone Consolidation . . . 40171
9235.14.1 What `BEDC.MetaCIC` Achieves . . . 40171
9235.14.2 Comparison with Textbook CIC Metatheory . . . 40174
9235.14.3 Frontier . . . 40175
9235.14.4 Boundary Statement . . . 40176
9235.14.5 Why This Boundary . . . 40176
9235.15 Closed-Term Metatheory Completion . . . 40177
9235.15.1 The Closed-Term Scope Claim . . . 40177
9235.15.2 What This Scope Buys . . . 40178
9235.15.3 The Three-Hypothesis Chain . . . 40179
9235.15.4 Closed Normalization and Candidate Witnesses . . . 40180
9235.15.5 Bounded Normalizer as Operation . . . 40181
9235.15.6 What Remains Outside the Closed-Term Scope . . . 40182
9235.15.7 Meaning for the BEDC Architecture . . . 40182
9235.16 Session-Level Metatheory Claim . . . 40183
9235.16.1 Claim 1: Substrate Invariants . . . 40183
9235.16.2 Claim 2: Subject Reduction in the Closed Scope . . . 40184
9235.16.3 Claim 3: Confluence in the Closed Scope . . . 40185
9235.16.4 Claim 4: Strong Normalization in the Closed Scope . . . 40186
9235.16.5 Claim 5: Decidable Type Checker . . . 40187
9235.16.6 Claim 6: Self-Representability via HostBridge . . . 40188
9235.16.7 Claim 7: Church-Encoded Data Structure Laws . . . 40189
9235.16.8 Claim 8: BetaConv Relation Properties . . . 40190
9235.16.9 Boundary Statement . . . 40191
9235.16.10 Delivered Invariants . . . 40191
9235.17 Session Closing Summary . . . 40192
9235.17.1 Mechanized Theorems Table . . . 40192
9235.17.2 How to Read the Table . . . 40197
9235.17.3 Closed Surface . . . 40198
9235.17.4 Open Frontier . . . 40199
9235.17.5 Why This Surface Closes Here . . . 40200
9235.17.6 Forward-Looking Summary . . . 40200

9235.17 What BEDC Does Not Claim . . . 40201
9235.18 What BEDC.MetaCIC Does Not Claim . . . 40202
9235.18.1 Why This Boundary Is Stated . . . 40202
9235.18.2 Out of Scope by Design . . . 40203
9235.18.3 Comparison With Standard CIC Literature . . . 40205
9235.18.4 Zero-Axiom Invariants Delivered . . . 40207
9235.18.5 Why This Matters for Self-Compilation . . . 40208
9235.18.6 What Would Extend the Scope . . . 40209

XI Capstones 40210

9236 Capstone Results 40211

9237 Compactness Implies Uniform Continuity 40212
9237.1 Uniform modulus from located finite covers . . . 40212
9237.2 The kernel construction . . . 40213
9237.3 Axiom-purity of the uniform-modulus construction . . . 40214

9238 Real Completeness via Regular Cauchy Sequences 40230
9238.1 Regular Cauchy sequences and their limits . . . 40230
9238.2 The limit theorem . . . 40233
9238.3 Constructive content and axiom purity . . . 40246
9238.4 Modulus comparison boundary . . . 40248

9239 The Riemann Hypothesis as a BEDC Conjecture 40261
9239.1 LEVEL 1–5 dependency chain . . . 40261
9239.2 Formal statement of RH in BEDC . . . 40262
9239.3 Carrier-stack reading . . . 40263
9239.4 Witness types a proof would produce . . . 40264
9239.5 Axiom-purity considerations . . . 40265
9239.6 Connection to existing BEDC content . . . 40266
9239.7 Roadmap and scope . . . 40266

9240 Three Boundaries for RH 40268

9241 The RH Zero–Skew EventFlow Target 40272
9241.1 Zero and skew packets . . . 40272
9241.2 EventFlow reading . . . 40273
9241.3 Channel and recognition boundary . . . 40273
9241.4 Mirror closure and orbit collapse . . . 40275
9241.5 Relation to finite analysis routes . . . 40276
9241.6 Analytic continuation as projection recovery . . . 40276

9242 EventFlow Certificate Flow for Primes, Zeta, and RH 40277
9242.1 Boundary before semantics . . . 40277
9242.2 Prime flow as factorization fixed point . . . 40278
9242.3 Zeta as socket recovery flow . . . 40279
9242.4 RH as fixed-output target . . . 40280

9242.5 Fixed-point stratification . . . 40281

9243 Prime-Local Balance and the RH Critical Surface 40282
9243.1 Zeta as a recovered carrier . . . 40282
9243.2 Prime-local channel normalization . . . 40283
9243.3 Prime-balanced form of RH . . . 40285
9243.4 Archimedean completion and the central surface . . . 40286
9243.5 Projection tower reading . . . 40287

9244 Finite Prime Windows and Local–Global RH Gluing 40289
9244.1 Finite Prime Windows . . . 40289
9244.2 Local Defect Rows . . . 40290
9244.3 Carrier Gluing . . . 40291

9245 Prime Window Recursive Tower 40292
9245.1 Prime-Window Layer Data . . . 40292
9245.2 Readback and Recursive Return . . . 40293
9245.3 No-Exception Invariant . . . 40294
9245.4 Prime-Recursive RH Target . . . 40294

9246 Local–Global Zero Exclusion Target 40296
9246.1 Target Statement . . . 40296
9246.2 Finite-Window Algebra . . . 40298
9246.3 Carrier Compatibility Obligations . . . 40298
9246.4 Positive Defect Certificates . . . 40299
9246.5 Obligation Assembly . . . 40300

9247 Finite Hermitian Rouche Start for ξ 40302
9247.1 Prime–Gamma finite packet . . . 40302
9247.2 Determinant approximant . . . 40303
9247.3 Outer-rectangle Rouche certificate . . . 40304
9247.4 Off-line Rouche localization . . . 40305
9247.5 Non-circularity ledger . . . 40305

9248 Prime Fingerprint Separation for RH 40307
9248.1 Prime fingerprint . . . 40307
9248.2 Mirror fingerprints . . . 40308
9248.3 RH as prime-fingerprint indistinguishability . . . 40309

9249 Prime-Skew Boundary Defect Refutation 40311

9250 Zeta as a Self-Similar Recovered Carrier 40315

9251 Zeta–Euler Tick Algebra 40322
9251.1 Prime Ledger and Logarithmic Hamiltonian . . . 40322
9251.2 Trace and Euler Decomposition . . . 40323
9251.3 Prime Tick and Solenoid Phase . . . 40324
9251.4 Zero Closure and Li Radial Residual . . . 40325

9252 Information-Channel Normal Form of RH 40327
9252.1 Information coordinates . . . 40327
9252.2 Prime carrier and cancellation events . . . 40328
9252.3 Dual return . . . 40329

9253 Zeta Zero Type Factorization 40331
9253.1 Zeta Zero Fiber . . . 40331
9253.2 Non-Trivial and Fixed-Half Subtypes . . . 40332
9253.3 Zero Generators . . . 40332
9253.4 Canonical Presentations and Compatibility . . . 40333
9253.5 Zero and Apart-Zero Conflict . . . 40334
9253.6 Omega Factorization . . . 40334

9254 Layered RH Translation 40336
9254.1 Source Carrier and Half Fiber . . . 40336
9254.2 Equivalent Readout Layers . . . 40337
9254.3 Zeta Recovery and Laplace Readout . . . 40338
9254.4 Exception Transport and No-Exception Form . . . 40338
9254.5 Synthesis . . . 40339

9255 RH-Free Zero Generation by Initiality 40341
9255.1 Spectral reformulation and the Hilbert–Pólya wall . . . 40349

9256 Ledger Register and Tick Reflection Positivity 40350
9256.1 Prime–Zeckendorf register arithmetic . . . 40350
9256.2 Finite reflection positivity as a PSD classification . . . 40351
9256.3 Reflection positivity of the OCLSD tick input . . . 40351
9256.4 Carrier-relative Osterwalder–Schrader reconstruction . . . 40352

9257 Pigeonhole Time Recurrence 40353

9258 PZG Finite-Window Self-Reference Recurrence 40360
9258.1 Finite PZG Windows . . . 40360
9258.2 Zero Reads . . . 40362

9259 Resolution Refinement Induction 40363

9260 Golden Lee–Yang Bridge Obligations 40370

9261 Golden Bridge Three-Level Closure 40377

9262 Zeta Ray Lee–Yang Container 40385

9263 Squarefree Prime Polarized Bridge 40391

9264 Xi Jensen Internal Container 40395

9265 Global Weil Positive Container Tower 40398

9266 Xi Time Negative Witness 40405

9267 Balanced Golden Li Closure 40408

9268 Balanced Polarized Lee–Yang CND 40412

9269 Balanced Golden Li CND Closure 40415

9270 Li–Weil Herglotz Target 40422

9271 Pick Kernel Triadic Positivity 40428

9272 Global Observation Triangle Invariant 40434

9273 Xi Log-Potential No-Scale Field 40442

9274 Triadic Pythagorean Obstruction 40444

9275 Fermat Triangle Obstruction Prototype 40451
9275.0 Quadratic Triangle Closure . . . 40451
9275.1 Fermat Classifiers . . . 40452
9275.2 RH Radial Scale as Fermat-Type Defect . . . 40453

9276 Right-Triangle Invariant System 40455

9277 Source-Side Triadic Audit Criterion 40461

9278 Radial Scale Projection System 40464

9279 Golden Weil Gram Induction 40469

9280 Diagonal Schur Square Target 40473

9281 Golden Nyman Square Tower 40477

9282 Mobius Nyman Mollifier Bounds 40483

9283 BEDC–Golden–Zeta Generation Machine 40487

9284 Internal Ruler Constraint Container 40494

9285 Arithmetic Spectral String Route 40500

9286 Thermodynamic No-Scale Route 40503

9287 Copenhagen Boundary Route Audit 40510
9287.1 Faithful Reformulations and Leverage . . . 40510
9287.2 The Three Projections of the Same Wall . . . 40510
9287.3 Checked Finite Brick . . . 40511
9287.4 GAG One-Step Certificate . . . 40513
9287.5 Gomboc Conditional Comparison . . . 40514
9287.6 Bad Examples as First-Class Boundary Data . . . 40515

9288 Unitary Scale Closure Route for RH 40517

9289 Phase, Orthogonal Readout, and Prime Register Residue 40521
9289.1 Unit Phase and Orthogonal Readout . . . 40521
9289.2 Prime Register Residue . . . 40522
9289.3 Constructive Source Chain . . . 40522
9289.4 Centered Zeta Modes . . . 40523
9289.5 RH-Facing Proof Load . . . 40525

9290 Circle-Triangle Generation and Hidden Scale Refusal 40527
9290.0 Generated Time and Orthogonal Readout . . . 40527
9290.1 Prime Registers and Addressed Residue . . . 40528
9290.2 Li Coordinates as Circle-Triangle Reads . . . 40530
9290.3 Conditional RH Route . . . 40531

9291 Memory Filtration and Prime–Zeckendorf Registers 40533
9291.0 Orthogonal Memory Filtration . . . 40533
9291.1 Prime–Zeckendorf–Goedel Register . . . 40535

9292 PZG Phase Readback and Log Energy 40539
9292.0 Log-Energy Readout . . . 40539
9292.1 Finite Self-Description and Residual Ledgers . . . 40540
9292.2 Phase Readback . . . 40544

9293 Zeta Virtual Machine Execution Trace 40546

9294 Zeta Mirror Cancellation and Local Closure 40553

9295 Zeta Zero Readout Faithfulness 40557

9296 Prime-Fiber Zoom Renormalization Tower 40561
9296.0 Prime-Fiber Zoom Systems . . . 40561
9296.1 PZG Energy Compatibility . . . 40563
9296.2 Analytic Prime-Shift Constraint . . . 40564
9296.3 Continuous Exponential Projection . . . 40565

9297 Prime-Fiber Zoom Scale Obstruction Bridge 40568
9297.0 Self-Substitution Interface . . . 40568
9297.1 Reflection-Positive Windows . . . 40569
9297.2 Smooth Branch Gate . . . 40570
9297.3 Zeta-Facing Closure Obligations . . . 40570

9298 Prime-Fiber Zoom NameCert Interface 40572
9298.0 NameCert Surface . . . 40572
9298.1 Ledger-Complete Limits . . . 40573
9298.2 Zeta-Facing Admission . . . 40573

9299 Prime-Shift Analytic Compatibility 40575
9299.0 Four Compatibility Rows . . . 40575

9300 Smooth Prime Classifier Tower 40577
9300.0 Classifier Tower Surface . . . 40577
9300.1 Finite Classifier Separation . . . 40578
9300.2 Cofinal Detection . . . 40578
9300.3 Radial-Scale Refusal . . . 40579
9300.4 Scale Faithfulness Boundary . . . 40580
9300.5 Explicit-Formula Prime Residuals . . . 40581

9301 Probability Amplitude Unit Geometry 40584
9301.0 Amplitude Readout . . . 40584
9301.1 Event Projection . . . 40585
9301.2 Ledger Consequence . . . 40586

9302 Layered Prime Probability Sieve 40587
9302.0 Probability Layers . . . 40587
9302.1 Probability Distribution Curves . . . 40588
9302.2 Completeness and Detection . . . 40589
9302.3 Cofinal Dissipation . . . 40590
9302.4 Conditional RH Interface . . . 40592

9303 Zeta as a Certified Inverse Limit 40593
9303.0 Ontological Rows . . . 40593
9303.1 Coverage . . . 40594
9303.2 RH Interface . . . 40595

9304 Probability Sieve Tower Proof Pattern 40596
9304.0 Local Mechanism . . . 40596
9304.1 Finite Projection Pigeonhole . . . 40597
9304.2 Tower Amplification . . . 40597
9304.3 Certified Limit Projection . . . 40598

9305 Prime-Generated Recursive Sieve 40600
9305.0 Recursive Local Sieve . . . 40600
9305.1 Finite Residual Detection . . . 40601
9305.2 Euler Probability Readout . . . 40601
9305.3 Continuation Interface . . . 40603

9306 All-Prime Object 40604
9306.0 Prime Source Packet . . . 40604
9306.1 Unit Rows . . . 40605
9306.2 Ledger Rows . . . 40606
9306.3 Finite Detection . . . 40607
9306.4 Zeta Source Interface . . . 40607

9307 Finite Prime Circle Universe 40609
9307.0 Index–Prime Certificate . . . 40609
9307.1 Finite Prime Circle . . . 40610
9307.2 Fourier Completeness . . . 40611
9307.3 Quadratic Norm Sieve . . . 40611

9307.5 Leakage Contradiction . . . 40612
9307.6 Admissible Addresses . . . 40613
9307.7 RH Route Interface . . . 40614

9308 Off-Line Zero Finite-or-Tail Address 40615
9308.1 Address Rows . . . 40615
9308.2 Ledger Dichotomy . . . 40616
9308.3 Address Theorem . . . 40617
9308.4 Finite Closed-Invisible Refusal . . . 40618
9308.5 Global Window Bridge . . . 40619

9309 Global Off-Line Zero to Finite NP Window 40622
9309.1 Radial Test Readout . . . 40622
9309.2 Prime-Power Address Window . . . 40623
9309.3 Finite-or-Tail Bridge . . . 40624
9309.4 Compact Window Exhaustion . . . 40625
9309.5 Closed-Packet Consequence . . . 40627

9310 Finite Prime Circle Off-Line Contradiction 40629
9310.1 Legal Zeta Zero Packets . . . 40629
9310.2 Off-Line Residual Refusal . . . 40630
9310.3 Conditional RH Interface . . . 40631

9311 Analytic Continuation Ontological Triple 40633
9311.1 Continuation Certificate . . . 40633
9311.2 Conservative Zeta Continuation . . . 40634
9311.3 Eta Parity Bridge . . . 40635
9311.4 Sieve Ledger Preservation . . . 40636

9312 Off-Line Zero to Prime Sieve Contradiction 40637
9312.1 Contradiction Packet . . . 40637
9312.2 From Spectral Packet to Prime Residual . . . 40638
9312.3 Prime-Generated Detection . . . 40639
9312.4 Contradiction Route . . . 40639
9312.5 Audit Boundary . . . 40640

9313 Prime Projection Certificate 40641
9313.1 Test-Row Projection . . . 40641
9313.2 Main and Correction Null Rows . . . 40642
9313.3 Finite Prime Projection . . . 40643
9313.4 Detection Interface . . . 40644

9314 Prime Sieve Tower Duality 40645
9314.1 Prime-Generated Sieve Tower . . . 40645
9314.2 Local Prime Sieve Surfaces . . . 40646
9314.3 Quadratic Local Unit Sieve . . . 40647
9314.4 Sieve Objects and Refinement . . . 40648
9314.5 Composition and Prime Sieve Factors . . . 40648
9314.6 Duality Energy and Ledger . . . 40649

9314.7 Recursive Prime-Generated Completeness . . . 40650
9314.8 Tower Interface . . . 40651

9315 Prime Sieve Duality Completeness 40652
9315.1 Type and Sieve Coordinates . . . 40652
9315.2 Finite Dual Detection . . . 40653
9315.3 Dual Tower Direction . . . 40654
9315.4 RH Interface . . . 40654

9316 Prime Distribution Curve Projection 40656
9316.0 Classifier Projections . . . 40656
9316.1 Unit and Radial Modes . . . 40657
9316.2 Completeness . . . 40658
9316.3 Conditional Fixed-Half Consequence . . . 40659

9317 Explicit Formula Scale Residual Bridge 40660
9317.0 Compact Test Rows . . . 40660
9317.1 Zero-Orbit Isolation . . . 40661
9317.2 Finite Prime Residual . . . 40662
9317.3 Signed Prime Energy . . . 40663

9318 Fixed Window Dominance Certificate 40666
9318.1 Finite Spectral Windows . . . 40666
9318.2 Dominance Certificates . . . 40668
9318.3 Prime-Side Realization . . . 40668
9318.4 Zeta Target . . . 40670

9319 Damped Spectral Packet Certificate 40672
9319.1 Damped Mirror Filter . . . 40672
9319.2 Negative Zero-Side Packet . . . 40674
9319.3 Prime Residual Interface . . . 40675

9320 Negative Weil Isolator 40677
9320.1 Mirror Pair Geometry . . . 40677
9320.2 Zero-Side Negative Direction . . . 40678
9320.3 Archimedean Independence . . . 40679
9320.4 Prime-Side Negative Residual . . . 40680
9320.5 Fixed-Half Consequence . . . 40681

9321 Paley–Wiener Zero Pair Tail Control 40683
9321.1 Mirror-Odd Local Isolator . . . 40683
9321.2 Finite Window Cancellation . . . 40685
9321.3 Uniform Tail Row . . . 40685

9322 Paley–Wiener Condition Number Bridge 40688
9322.1 Vertical Decay Cost . . . 40688
9322.2 Condition Number Tail Estimate . . . 40689
9322.3 Archimedean Condition Number . . . 40690
9322.4 Condition-Number RH Interface . . . 40691

9323 Circle-Triangle Projection Dictionary 40693
9323.0 Common Generator . . . 40693
9323.1 Classical Projections . . . 40694
9323.2 Dominance and Cofinality Projections . . . 40695
9323.3 Minimal Proof Package . . . 40695

9324 Unresolved Projection Obligation Ledger 40697
9324.0 Open Projection Obligations . . . 40697
9324.1 Conditional Package . . . 40699
9324.2 Compressed Core . . . 40700

9325 Triadic Solenoid Pick Tower 40701
9325.0 Finite Solenoid Sources . . . 40701
9325.1 Finite Herglotz Kernels . . . 40702
9325.2 Disk Kernels and the Limiting Target . . . 40703
9325.3 Moment Positivity Target . . . 40705

9326 Xi Hausdorff Moment Audit 40706
9326.0 Centered Source Moments . . . 40706
9326.1 Finite Hausdorff Audit . . . 40707
9326.2 From Moments to Stieltjes and Pick Kernels . . . 40708
9326.3 Truncated Witnesses . . . 40709

9327 Xi Triangular Moment Certificate Tower 40710
9327.0 Triangular Difference Ledger . . . 40710
9327.1 Bernstein Measure Layer . . . 40711
9327.2 Finite Stieltjes and Herglotz Layer . . . 40712
9327.3 Finite Pick and Hilbert Layer . . . 40712
9327.4 Tower Convergence Target . . . 40713

9328 Final Moment Obligation Core 40715
9328.0 Moment Difference Surface . . . 40715
9328.1 Source Decomposition Target . . . 40716
9328.2 Triadic Norm Decomposition . . . 40716
9328.3 Obligation Boundary . . . 40717

9329 Source-Positive Obligation Frontier 40718
9329.0 Common Observation Coordinate . . . 40718
9329.1 Legal Source and Pick Factorization . . . 40719
9329.2 Moment Positivity Frontier . . . 40721
9329.3 Analytic Identification and Finite Tower . . . 40722
9329.4 Compressed Frontier Package . . . 40723

9330 Zeta BEDC Closure Strength Gate 40725
9330.0 Closure Strength . . . 40725
9330.1 Conditional Consequence . . . 40726

9331 Radial Register Obstruction Frontier 40728
9331.1 Unresolved Circle–Triangle–Register Package . . . 40728
9331.2 Zeta Source Construction Boundary . . . 40729
9331.3 Single-Mode Source Bridge . . . 40731
9331.4 Invariant Separation . . . 40731
9331.5 Compact Gauge Obstruction . . . 40733
9331.6 Ledgered Scale Is Not Pure Closure . . . 40733
9331.7 Conditional Core . . . 40734

9332 Bulk–Boundary Prime-Shadow Interface for RH 40735
9332.1 Bulk and Boundary Data . . . 40735
9332.2 Prime Sieve and Shadows . . . 40736
9332.3 Zero Atoms and Defect Incompatibility . . . 40737
9332.4 Conditional RH Assembly . . . 40738

9333 RH Proof Obligation Package 40739
9333.1 Constructive Target . . . 40739
9333.2 Structural Rows . . . 40740
9333.3 Rigidity Rows . . . 40740
9333.4 Proof Route Ordering . . . 40742
9333.5 Witness Export . . . 40742

9334 Non-Fixed Orbit Targets for RH 40744
9334.1 Candidate Carrier and Orbit Predicates . . . 40744
9334.2 Fixed Section and Orbit Exclusions . . . 40745
9334.3 Constructive Apart Route . . . 40746

9335 Counterexample Program Safety Flow for RH 40747
9335.1 Elementary arithmetic interface . . . 40747
9335.2 Program and prime-register carrier . . . 40748
9335.3 GroundCompiler run recognition . . . 40749
9335.4 Refutation layer . . . 40751
9335.5 Safety fixed point . . . 40752

9336 RH Route Closure and the Halting Boundary 40754
9336.1 Three strata around RH . . . 40754
9336.2 Grounding rank . . . 40755
9336.3 Closure deciders and halting . . . 40756
9336.4 Audit bases . . . 40756
9336.5 Finite human expression maximality . . . 40757

9337 Route Closure and Fixed RH Separation 40760
9337.1 Route Layer and Statement Layer . . . 40760
9337.2 Route Closure Undecidability . . . 40760
9337.3 Fixed RH Independence Requires a Separate Meta-Theorem . . . 40761
9337.4 Finite Counterexample Constraint . . . 40762

9338 Dyadic Truncation Laguerre–Pólya Route Refutation 40764

9339. A Constructive Winding-Certificate Layer for the Argument-Principle Route 40766

9340. Contour Integration as an Operation on $\mathsf{Holomorphic}^{\uparrow}$ 40769
9340.1. Piecewise-linear contour carrier . . . 40769
9340.2. Riemann sum approximation . . . 40770
9340.3. The contour integral . . . 40770
9340.4. Operation laws . . . 40771
9340.5. Cauchy's theorem and the integral formula . . . 40772
9340.6. Operation laws and ledger . . . 40772

9341. Analytic Continuation as an Operation on $\mathsf{Holomorphic}^{\uparrow}$ 40775
9341.1. Domain compatibility . . . 40775
9341.2. The continuation predicate . . . 40776
9341.3. Chain continuation . . . 40776
9341.4. Branch obstruction (schema-only horizon) . . . 40776
9341.5. Concrete continuations . . . 40777
9341.6. Operation laws . . . 40777
9341.7. The operation certificate . . . 40777

9342. The ζ Continuation Application: Extending $\mathsf{ZetaBasic}^{\uparrow}$ to $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ 40780
9342.1. The Dirichlet eta function . . . 40780
9342.2. ζ on the half-plane $\Re(s) > 0$ via eta . . . 40781
9342.3. The pole at $s = 1$ . . . 40781
9342.4. The functional equation . . . 40781
9342.5. ζ on $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ . . . 40782
9342.6. Trivial zeros from the functional equation . . . 40782
9342.7. The application certificate . . . 40783

9343. Exploratory RH Route Ledger 40784
9343.1. Audit Convention . . . 40784
9343.2. Three Boundaries and the Prime-Unitary Surface . . . 40784
9343.3. Located Process Geometry and Type Boundaries . . . 40785
9343.4. Common Carrier and Zeta as a Projection Chain . . . 40785
9343.5. Object Induction, Three Invariants, and No-Split Zeros . . . 40786
9343.6. Finite Prime Windows, Odd Defects, and Even Energy . . . 40786
9343.7. Far-End, Solenoid, and Lee–Yang Readings . . . 40787
9343.8. Dyadic Truncations, Jensen, Hilbert–Polya, and Weil Positivity . . . 40787
9343.9. Lean Decision . . . 40788
9343.10. Summary . . . 40788

9344. The Unary-Direction Bridge $\mathsf{Nat}^{\uparrow} \leftrightarrow \mathsf{AxisNat}^{\uparrow}$ 40790
9344.1. Carrier intersection . . . 40790
9344.2. The bridge interface . . . 40790
9344.3. Standard-model isomorphism (horizon) . . . 40791

9345 A Concrete Naming Certificate for $\mathsf{ZCarry}^{\uparrow}$ 40792
9345.1 The carry classifier . . . 40792
9345.2 Carry is not history-sameness . . . 40792
9345.3 Carry transports normal-form status . . . 40793
9345.4 The certificate . . . 40793

9346 Zeckendorf Golden Phase Projection 40795
9346.1 Window observers and residual fibers . . . 40795
9346.2 Window-six Fibonacci ledger . . . 40796
9346.3 Golden phase projection . . . 40797
9346.4 Physical representation obligations . . . 40803

9347 Window-Six Inverse-Limit Lifting 40828
9347.1 Address tower . . . 40828
9347.2 Finite support and tower continuation . . . 40830
9347.3 Fixed projection shadow . . . 40831
9347.4 Projection-shadow comparison with recovered zeta . . . 40832
9347.5 Operator compatibility . . . 40833
9347.6 RH residual bridge target . . . 40834

9348 Layer-Kernel Residual Induction 40836
9348.1 Linearized residual tower . . . 40836
9348.2 First nonzero layer . . . 40837
9348.3 Return operators and spectral exclusion . . . 40837
9348.4 Layer-kernel refutation . . . 40838
9348.5 Proof obligations . . . 40839

9349 Zeta-to-Zeckendorf Bridge Obligations 40840
9349.1 Bridge interface . . . 40840
9349.2 Neutral-mode obstruction . . . 40841
9349.3 Bridge-to-refutation criterion . . . 40842
9349.4 Finite laboratory status . . . 40842
9349.5 Work order . . . 40843

9350 Cannot-Claim Registry: AxisZeckendorf Cluster 40844
9350.1 Negative theorems backing the registry . . . 40844
9350.2 Registry of non-claims . . . 40845
9350.3 Horizon obligations associated with the registry . . . 40845

9351 Mark–Hist–Ext–Cont Kernel Spine 40847
9351.1 Raw marks . . . 40847
9351.2 Histories . . . 40848
9351.3 Extension and continuation . . . 40849

9352 Ask–Sig–Pkg–Gap–NameCert Kernel Spine 40851
9352.1 Asking and signatures . . . 40851
9352.2 Packages, gaps, and exact globalization . . . 40853
9352.3 Name certification . . . 40854

9353 Finite Data Interface Spine **40858**
9353.1 Interface Template . . . 40858
9353.2 Boolean Endpoint Interface . . . 40859
9353.3 Optional History Interface . . . 40860
9353.4 Tagged Sum Interface . . . 40860
9353.5 Ordered Product Interface . . . 40861
9353.6 Finite List Interface . . . 40862
9353.7 Finite Data Layer . . . 40863

9354 Arithmetic Interface Spine **40864**
9354.1 Unary Natural Histories . . . 40864
9354.2 Unary Continuation Addition . . . 40865
9354.3 Unary Multiplication and Divisibility . . . 40866
9354.4 Unary Order . . . 40867
9354.5 Signed Integers . . . 40868
9354.6 Rational Histories . . . 40868
9354.7 Arithmetic Layer Boundary . . . 40869

9355 Algebra Interface Spine **40870**
9355.1 Algebraic Certificate Template . . . 40870
9355.2 Magma Through Monoid . . . 40871
9355.3 Groups . . . 40872
9355.4 Rings and Fields . . . 40873
9355.5 Modules and Vector Spaces . . . 40874
9355.6 Algebra Layer Boundary . . . 40875

9356 Order Set Type Interface Spine **40876**
9356.1 Relational Certificate Template . . . 40876
9356.2 Preorders and Posets . . . 40877
9356.3 Total Orders and Lattices . . . 40878
9356.4 Set-Like Membership . . . 40879
9356.5 Type-Like Transport . . . 40880
9356.6 L9 Boundary . . . 40880

9357 Countable Completion Interface Spine **40882**
9357.1 Countable Names . . . 40882
9357.2 Dyadic Rational Core . . . 40883
9357.3 Regular Sequences and Real Names . . . 40884
9357.4 Completion . . . 40885
9357.5 Metric and Topology . . . 40886
9357.6 Metric Completion and Maps . . . 40887
9357.7 L10 Boundary . . . 40888

9358 Analysis Interface Spine **40889**
9358.1 Analytic Certificate Template . . . 40889
9358.2 Calculus . . . 40890
9358.3 Real Analytic Functions . . . 40891
9358.4 Complex Analytic Functions . . . 40893

9358.5 Measure and Integration . . . 40894
9358.6 Functional Analysis . . . 40895
9358.7 Gamma Bridge . . . 40896
9358.8 L11 Boundary . . . 40896

9359 Geometry, Category, and Homology Interface Spine . . . 40898
9359.0 Geometry Certificate Template . . . 40898
9359.2 Categories, Functors, and Natural Transformations . . . 40898
9359.3 Categorical Limits . . . 40900
9359.4 Manifolds and Bundles . . . 40900
9359.5 Homology and Cohomology . . . 40901
9359.6 L12 Boundary . . . 40902

9360 Number Theory and Zeta Interface Spine . . . 40903
9360.1 Prime Axis . . . 40903
9360.2 p-Adic and Adele Rows . . . 40904
9360.3 Dirichlet Series and Zeta . . . 40905
9360.4 Critical Strip and Zeta Zeros . . . 40906
9360.5 RH Target Surface . . . 40906
9360.6 L13 Boundary . . . 40907

9361 Omega Generation Structure Interface . . . 40908
9361.1 Licensed Generation Category . . . 40908
9361.2 Encoded Display . . . 40909
9361.3 Pairing by Involution . . . 40910
9361.4 Time as Generated Direction . . . 40911
9361.5 Interface Assembly . . . 40911

9362 Mature Bridge and Omega Rigidity Interface Spine . . . 40913
9362.0 Conservative Bridge Boundary . . . 40913
9362.2 Standard and Mathlib Bridges . . . 40914
9362.3 Conservativity and Verification Reports . . . 40914
9362.4 Phase Discipline and Refutation Boundary . . . 40915
9362.5 Omega Tower . . . 40916
9362.6 Omega-RH Rigidity . . . 40916
9362.7 L14 Boundary . . . 40917

9363 The Finite Kernel Is Already a Category . . . 40918
9363.1 Hom as unary continuation . . . 40918
9363.2 Derived presentations over the unary category . . . 40919

9364 Lean Kernel Acceptance as the Canonical BEDC Audit Witness . . . 40921
9364.1 The setup . . . 40921
9364.2 What the Lean kernel does at acceptance . . . 40922
9364.3 NameCert stamping . . . 40922
9364.4 `#print axioms` as audit ledger closure . . . 40923
9364.5 `lake build` as ledger instantiation . . . 40923
9364.6 Closed-term substitution cross-link . . . 40924

9364.7 Recursor cross-link . . . 40924
9364.8 Why this is not an infinite regress . . . 40925
9364.9 Kernel boundary as apophatic socket . . . 40925
9364.10 Boundary: what kernel acceptance does not certify . . . 40925

9365 Computation as Continuation: BEDC's Operational Semantics 40927
9365.1 Continuation as small-step transition . . . 40928

9366 Classifier-Level Information: Computation, Unfolding, and Discovery 40937
9366.1 Certified classifier states . . . 40937
9366.2 Computation as record growth . . . 40938
9366.3 Discovery as classifier shift . . . 40939
9366.4 Generators and presentation changes . . . 40940
9366.5 Separation principle . . . 40941

9367 Type Checking as Classifier Membership 40943
9367.1 Extension predicate as type-membership . . . 40943
9367.2 Subject reduction from closure laws . . . 40944
9367.3 Curry-Howard as identity . . . 40944

9368 The Halting Problem as a Limit of the Form of Distinction 40947
9368.1 Halting as fixed-point obstruction . . . 40947
9368.2 Halting as open-meta-loop closure . . . 40948
9368.3 Connection to the form of distinction . . . 40948

9369 Channel Termination and the Halting Boundary 40950
9369.1 Local channel termination . . . 40950
9369.2 Bare bitstreams and self-delimitation . . . 40951
9369.3 Channel grammar and finite window geometry . . . 40952
9369.4 The halting boundary . . . 40953

9370 Strong Normalisation versus Turing Completeness: The Total-Language Position of BEDC 40955
9370.1 Setup: total functional language as discipline . . . 40955
9370.2 The expressive gap, made precise . . . 40956
9370.3 Position among total functional languages . . . 40956
9370.4 The forced choice, read structurally . . . 40957
9370.5 Curry-Howard, and what it means here . . . 40957
9370.6 Mapping to BEDC engineering . . . 40958
9370.7 Closure remarks . . . 40958

9371 Compilation as Naming-Certificate Morphism 40960
9371.1 Compiler as classifier-preserving morphism . . . 40960
9371.2 Compositional compilation . . . 40961
9371.3 Optimisation and decompilation . . . 40961

9372 Closed-term substitution as the self-compilation step 40963
9372.1 The setup . . . 40963
9372.2 The Lean witness . . . 40964
9372.3 Why this is the self-compilation step . . . 40965
9372.4 Generator-closure classifier reading . . . 40966
9372.5 Audit gate as the executable form of the classifier . . . 40967
9372.6 Boundary: what this lemma does not claim . . . 40968

9373 The recursor as an authorized generator 40970
9373.1 The setup . . . 40970
9373.2 The kernel principle . . . 40970
9373.3 Generator role . . . 40971
9373.4 NameCert structure . . . 40971
9373.5 Generator-closure classifier view . . . 40972
9373.6 Comparison with classical "by definition" reading . . . 40972
9373.7 Connection to closed-term substitution . . . 40973
9373.8 Audit gate as the executable form of the classifier . . . 40973
9373.9 Boundary: what is not a recursor-induced NameCert . . . 40974

9374 Subject reduction discharge routes: explicit bundle 40976
9374.1 The setup . . . 40976
9374.2 Bundle route . . . 40976
9374.2.1 Definition . . . 40976
9374.2.2 Subject reduction from bundle . . . 40976
9374.3 Checked obstruction ledger . . . 40977
9374.4 Boundary . . . 40977

9375 Closed-MetaCIC consistency assembly: a partial closure map 40979
9375.1 The setup . . . 40979
9375.2 The assembled positive content . . . 40979
9375.2.1 Closed-term strong normalisation . . . 40979
9375.2.2 No closed-normal proof of False . . . 40979
9375.2.3 Closed-normal consistency assembly . . . 40980
9375.3 The unassembled frontier . . . 40980
9375.3.1 Conditional subject reduction . . . 40980
9375.3.2 The closed reduction obstruction . . . 40980
9375.4 The assembly diagram . . . 40981
9375.5 What this partial closure does establish . . . 40981
9375.6 What discharge would buy . . . 40982
9375.7 Apophatic framing . . . 40982
9375.8 Boundary . . . 40982

9376 Closed-normal consistency: BEDC's first unconditional main result 40984
9376.1 The result . . . 40984
9376.2 Why this is a first-class main result . . . 40984
9376.3 Constructive content . . . 40985
9376.4 Relation to classical results . . . 40986
9376.5 What it does not claim . . . 40986

9376.6 The role of the BEDC project . . . 40987
9376.7 Boundary . . . 40987

9377 MetaCIC subject-reduction obstruction: an audit map 40989
9377.1 The setup . . . 40989
9377.2 What is already proven . . . 40989
9377.3 The four discharge obligations . . . 40990
9377.4 The explicit obstruction theorem . . . 40991
9377.5 Critical path . . . 40991
9377.6 Where this fits in BEDC's apophatic framing . . . 40991
9377.7 The dependent-codomain frontier . . . 40991
9377.8 Boundary . . . 40992

9378 MetaCIC Confluence partial-closure audit map 40994
9378.1 The setup . . . 40994
9378.2 What is established . . . 40994
9378.3 What is conditional . . . 40996
9378.4 Comparison with the SR obstruction . . . 40996
9378.5 Apophatic framing . . . 40997
9378.6 Boundary . . . 40997

9379 MetaCIC Decidability partial-closure audit map 41001
9379.1 The setup . . . 41001
9379.2 What is established . . . 41001
9379.3 Normal equality and bounded beta conversion . . . 41003
9379.4 What remains conditional . . . 41003
9379.5 Comparison with other audit maps . . . 41004
9379.6 Apophatic framing . . . 41005
9379.7 Boundary . . . 41006

9380 MetaCIC Substitution module audit map 41007
9380.1 The setup . . . 41007
9380.2 Core operations . . . 41007
9380.2.1 `shift_closed` . . . 41007
9380.2.2 `substitute_closed` . . . 41008
9380.2.3 Closed composition . . . 41008
9380.3 Connection to subject reduction . . . 41009
9380.4 Connection to GeneratorClosureClassifier . . . 41009
9380.5 Apophatic framing . . . 41009
9380.6 Boundary . . . 41010

9381 MetaCIC ClosurePreservation module audit map 41011
9381.1 The setup . . . 41011
9381.2 Established theorems . . . 41011
9381.2.1 `shift_one_preserves_closed_succ` . . . 41011
9381.2.2 `substitute_var_preserves_closed_at` . . . 41012
9381.2.3 `substitute_preserves_closed_at` . . . 41012
9381.2.4 `betaStep_preserves_closed` . . . 41012

9381.2.5 `betaStarStep_preserves_closed` . . . 41012
9381.3 Why ClosurePreservation is unconditional while SR is not . . . 41013
9381.4 Role within MetaCIC . . . 41013
9381.5 Connection to GeneratorClosureClassifier . . . 41013
9381.6 Boundary . . . 41014

9382 MetaCIC TypedExamples module audit map . . . 41016
9382.1 The setup . . . 41016
9382.2 Inventory . . . 41016
9382.3 Representative theorems . . . 41017
9382.4 Role within MetaCIC . . . 41019
9382.5 Connection to other audit maps . . . 41019
9382.6 Boundary . . . 41019

9383 MetaCIC Normalization partial-closure audit map . . . 41021
9383.1 The setup . . . 41021
9383.2 What is established . . . 41021
9383.3 File-by-file audit reading . . . 41022
9383.4 What remains conditional . . . 41023
9383.5 Comparison with SR obstruction and Confluence audit map . . . 41024
9383.6 Apophatic framing and boundary . . . 41024

9384 MetaCIC Beta module audit map . . . 41028
9384.1 The setup . . . 41028
9384.2 Core inductive definition . . . 41028
9384.2.1 BetaStep . . . 41028
9384.2.2 File-level reading . . . 41029
9384.3 Established results . . . 41029
9384.4 Connection to the SR obstruction . . . 41030
9384.5 Apophatic framing and boundary . . . 41031

9385 MetaCIC audit synthesis: closed-CIC critical-path overview . . . 41032
9385.1 The setup . . . 41032
9385.2 The five submodule audit maps . . . 41032
9385.2.1 Reading rule . . . 41032
9385.2.2 Subject reduction . . . 41032
9385.2.3 Confluence . . . 41033
9385.2.4 Normalization . . . 41033
9385.2.5 Substitution . . . 41033
9385.2.6 Decidability . . . 41033
9385.3 The critical path to unconditional closed consistency . . . 41033
9385.4 Status matrix . . . 41034
9385.5 How the maps interact . . . 41035
9385.6 The frontier shape . . . 41036
9385.7 Apophatic discipline applied to meta-theory . . . 41036
9385.8 Boundary . . . 41037

9386 Closed-MetaCIC open problems: consolidated obligation list 41038
9386.1 The setup . . . 41038
9386.2 Subject reduction obligations (4) . . . 41038
9386.3 Confluence remaining questions . . . 41039
9386.4 Normalization remaining questions . . . 41041
9386.5 Decidability remaining questions . . . 41041
9386.6 TypedExamples coverage questions . . . 41042
9386.7 Cross-cutting frontier questions . . . 41043
9386.7.1 Dependent-codomain barrier . . . 41043
9386.7.2 Bundle discharge route . . . 41043
9386.7.3 Normal-form-only consistency . . . 41043
9386.7.4 Substrate extension for dependent-codomain closure . . . 41043
9386.8 Operational priorities . . . 41044
9386.9 Boundary . . . 41044

9387 Three Axiom Families, One Meta-Closure Taxonomy 41050
9387.1 The three axiom families as taxonomy entries . . . 41050
9387.2 The unification taxonomy . . . 41051
9387.3 Effective replacements inside BEDC . . . 41051
9387.4 The axiom-purity gate as refusal of meta closure . . . 41052
9387.5 Assumed bridges and derived bridges . . . 41052

9388 The Closure-Universality Quadrant: A Two-Axis Substrate Map 41063
9388.1 The two axes, formally . . . 41063
9388.2 Quadrant I: universal and closed . . . 41064
9388.3 Quadrant II: universal but not closed . . . 41064
9388.4 Quadrant III: not universal but closed . . . 41065
9388.5 Quadrant IV: neither universal nor closed . . . 41065
9388.6 The quadrant as map . . . 41066
9388.7 Mapping to BEDC engineering . . . 41066
9388.8 Closure remarks . . . 41067

9389 Generator-Closure Certificates and the Two Layers of BEDC: Technical Mechanism for the Strong Externality Claim 41068
9389.1 The two operational layers of BEDC . . . 41068
9389.2 Naming certificates and generator-closure classifiers . . . 41069
9389.3 The strong form of BEDC's induction-style claim . . . 41071
9389.4 GAP plays two simultaneous roles . . . 41071
9389.5 Compression requires memory . . . 41072
9389.6 BEDC's structural description of CIC and the host-Lean delegation . . . 41072
9389.7 The accurate technical claim alongside the strong philosophical claim . . . 41073
9389.8 Standard axiom systems internalise; BEDC externalises . . . 41073
9389.9 The unified technical-and-philosophical claim . . . 41074
9389.10 Closure remarks . . . 41075

9390 The Cellular Automaton Reading: Pattern, Substrate, and the Cosmological Inscription 41077
9390.1 Setup: classification is the mark discipline . . . 41077
9390.2 Rule 110 as the minimal universal cellular substrate . . . 41078
9390.3 Even Rule 110 cannot internalise induction . . . 41079
9390.4 The cosmological reading: universe as cellular substrate . . . 41080
9390.5 The cook-frontier as quantified Goedel boundary . . . 41080
9390.6 Mapping the cellular reading to BEDC engineering . . . 41081
9390.7 Closure remarks . . . 41082

9391 The Cellular-Carrier Alternative: Could BEDC Be Built on the Substrate Directly? 41084
9391.1 Setup: two carriers, one closure premise . . . 41084
9391.2 The equivalence: bit-string encoding . . . 41085
9391.3 What the alternative would change . . . 41085
9391.4 Why the project keeps the BHist carrier . . . 41086
9391.5 What would justify a switch . . . 41086
9391.6 The position read structurally . . . 41086
9391.7 Mapping to BEDC engineering . . . 41087
9391.8 Closure remarks . . . 41087

9392 The twin substrate architecture: MetaCIC and GroundCompiler 41089
9392.1 The setup . . . 41089
9392.2 Why two substrates . . . 41089
9392.3 Audit map family per substrate . . . 41090
9392.3.1 MetaCIC audit family . . . 41090
9392.3.2 GroundCompiler audit map . . . 41090
9392.4 Audit reexport namespaces . . . 41090
9392.4.1 `BEDC.MetaCIC.Audit` . . . 41090
9392.4.2 `BEDC.GroundCompiler.Audit` . . . 41090
9392.5 The cross-substrate synthesis result . . . 41091
9392.6 Apophatic framing of the twin substrate . . . 41091
9392.7 Why not collapse into one substrate . . . 41091
9392.8 Boundary . . . 41092

9393 Closed Observation, the Transcendental Input, and the Inscription Point 41094
9393.1 Closed observational systems and information conservation . . . 41094
9393.2 The unbridgeable gap to universal quantification . . . 41095
9393.3 The minimum transcendental input . . . 41096
9393.4 Naming the source . . . 41096
9393.5 The individual as inscription point . . . 41097
9393.6 Mapping the philosophical content to BEDC's engineering . . . 41097
9393.7 Closure remarks . . . 41098

9394 Transcendental Supply Taxonomy 41100
9394.1 The need for taxonomy . . . 41100
9394.2 Type I: free-floating induction socket . . . 41101
9394.3 Type II: choice and unwitnessed existential . . . 41101

9394.4 Type III: extensional identification . . . 41102
9394.5 Type IV: causal continuity . . . 41102
9394.6 Type V: inter-history coordination . . . 41103
9394.7 Type VI: free-will and inscription novelty . . . 41103
9394.8 Type VII: self-reference closure . . . 41104
9394.9 Non-collapse laws for the seven types . . . 41105
9394.10 Engineering use protocol . . . 41106
9394.11 Unified long table . . . 41107
9394.12 The single T behind seven types . . . 41108

9395 External Supply Without Axioms 41110
9395.1 The boundary problem . . . 41110
9395.2 Axiom, setup field, and socket . . . 41111
9395.3 The admission decision . . . 41111
9395.4 GAP as the memory of boundary pressure . . . 41112
9395.5 Generator-local induction and the refusal of free-floating induction . . . 41113
9395.6 Choice, quotient, and proposition equality . . . 41113
9395.7 Observer growth and selector ledgers . . . 41114
9395.8 Multi-Hist physics without global axioms . . . 41114
9395.9 Audit gates as the enforcement layer . . . 41115
9395.10 The non-axiom theorem schema . . . 41115
9395.11 Forbidden phrasings and replacements . . . 41116
9395.12 Boundary of this chapter . . . 41116
9395.13 Closure . . . 41117

9396 Audit Gates as Transcendental Boundary Enforcement 41119
9396.1 Why audit gates are not bookkeeping . . . 41119
9396.2 `check-axioms.py`: refuse substrate-internal axioms . . . 41120
9396.3 `axiom-purity -strict`: refuse host axiom propagation . . . 41121
9396.4 `conservativity-audit`: gate compression residue . . . 41121
9396.5 `bedc_ci.py audit`: gate paper-Lean drift . . . 41122
9396.6 The gates as a system . . . 41123
9396.7 Comparison: in CIC, gates are absent . . . 41124
9396.8 Gate-to-boundary map . . . 41125
9396.9 Closure . . . 41126

9397 Inscription Event as a Concrete Instance: Free Will, Agency, and NameCert Audit 41128
9397.1 The setup . . . 41128
9397.2 Restating the inscription event . . . 41129
9397.3 A concrete case: a researcher writing a lemma . . . 41129
9397.4 The NameCert structure of the event . . . 41130
9397.5 GAP ledger for inscription . . . 41131
9397.6 Compatibilism as structure . . . 41131
9397.7 Where the account does not apply . . . 41132
9397.8 Implications for AI and mechanized reasoning . . . 41132
9397.9 Concrete-instance summary . . . 41133

9398 The Large Model as Inscription Point: A Structural Comparison 41135
9398.1 Setup: large model as closed observational system . . . 41135
9398.2 Tensor classifiers and certificate extraction . . . 41136
9398.3 Large model inherits the no-induction theorem . . . 41140
9398.4 Inscription role: where the forward-binding comes from . . . 41140
9398.5 Comparison: BEDC kernel, cellular substrate, large model . . . 41141
9398.6 What this changes about AI epistemics . . . 41141
9398.7 Why this comparison is structurally informative . . . 41142
9398.8 Mapping to BEDC engineering . . . 41142
9398.9 Closure remarks . . . 41143

9399 Empty as the Fable Machine: Observer-Sequence Growth and Selector Ledger 41145
9399.1 Empty as zero-information boundary . . . 41145
9399.2 One-step birth: from empty boundary to singleton continuation . . . 41146
9399.3 Observer sequence as finite selector ledger . . . 41147
9399.4 Multi-solution branch space and realized selector . . . 41147
9399.5 Empty as the fable machine . . . 41148
9399.6 No hidden chooser: selector witness vs. classical choice . . . 41148
9399.7 Synthesis: observer-Hist as fable ledger . . . 41149

9400 Apophatic Naming as Methodological Discipline 41151
9400.1 The setup: what is being named . . . 41151
9400.2 The apophatic discipline . . . 41152
9400.3 Comparison with positive naming . . . 41152
9400.4 Not undefined and not an axiom . . . 41153
9400.5 Inventory of BEDC apophatic names . . . 41153
9400.6 Audit gates as enforcement layer . . . 41154
9400.7 God and T are not metaphor . . . 41154
9400.8 Closed system invariant, apophatically restated . . . 41155
9400.9 Implications for proof discipline . . . 41155
9400.10 Boundary . . . 41155
9400.11 Conclusion . . . 41156

9401 The Unique Apophatic Fixed Point and Local Inscription 41157
9401.1 From many sockets to a single far end . . . 41157
9401.2 What $\top$ is not . . . 41158
9401.3 Self-center as local inscription . . . 41158
9401.4 Records side and inscription side as two readings of one event . . . 41159
9401.5 Free will under the two-aspect reading . . . 41159
9401.6 Other minds, time, space, and universe-for-observer as projections . . . 41160
9401.7 Forbidden writings . . . 41160
9401.8 The unique apophatic fixed point principle . . . 41161
9401.9 Closure . . . 41161

9402 Hash-like Apophatic Fixed Point: Digest, Fiber, GAP, and the Universe 41163
9402.1 Why the hash analogy clarifies T . . . 41163
9402.2 Hash as deliberate fiber generator . . . 41164
9402.3 From hash fiber to BEDC GAP . . . 41165

9402.4 The universe is not a global digest . . . 41165
9402.5 The hash-like apophatic fixed point . . . 41166
9402.6 Apophatic infinity as boundary role . . . 41166
9402.7 Illumination surface and fiber/beam reading . . . 41167
9402.8 Self-center on a digest surface . . . 41168
9402.9 Collision and other minds . . . 41168
9402.10 Information conservation under digest use . . . 41169
9402.11 Multi-Hist physics under the hash-like reading . . . 41170
9402.12 Relation to the unique apophatic fixed point chapter . . . 41170
9402.13 Mathematical understanding as fiber navigation . . . 41171
9402.14 Failure modes . . . 41171
9402.15 Cannot-claim registry . . . 41172
9402.16 Implementation-facing packet . . . 41173
9402.17 Non-axiom theorem schema . . . 41173
9402.18 Closure . . . 41174

9403 Diagrammatics of the Apophatic Far End . . . 41176
9403.1 Closed substrate and socket far ends . . . 41176
9403.2 Local digest, fiber, and provenance . . . 41177
9403.3 Local inscription and self-center . . . 41178
9403.4 Many observers and shared far-end commitment . . . 41179
9403.5 Time, space, and universe-for-observer . . . 41179
9403.6 The infinity diagram . . . 41179
9403.7 Forbidden collapses . . . 41180
9403.8 Closure . . . 41180

9404 Cataphatic Process Names and Apophatic Far Ends . . . 41182
9404.1 Value axes and generation axes . . . 41182
9404.2 Located irrationality . . . 41183
9404.3 The apophatic far-end role . . . 41183
9404.4 Level separation . . . 41184
9404.5 Compactification points . . . 41185
9404.6 Axiom-purity boundary . . . 41185
9404.7 Closure . . . 41186

9405 Information Strata and Non-Collapse . . . 41187
9405.1 Value, process, and far-end strata . . . 41187
9405.2 Algebraic and transcendental directions . . . 41188
9405.3 Allowed folding and forbidden flattening . . . 41189

9406 The audit-map family: a BEDC methodology . . . 41192
9406.1 The setup . . . 41192
9406.2 What an audit map is . . . 41193
9406.3 The nine audit map chapters of BEDC.MetaCIC . . . 41193
9406.4 Common structure across audit maps . . . 41194
9406.5 Methodology versus other disciplines . . . 41195
9406.6 When to use this methodology . . . 41195
9406.7 Audit-map family as apophatic naming in action . . . 41195

9406.8 Boundary . . . 41195

9407 BEDC GroundCompiler module audit map 41197
9407.1 The setup . . . 41197
9407.2 Submodule inventory . . . 41197
9407.3 Core statement defs . . . 41198
9407.4 Established theorems . . . 41199
9407.5 Channel boundary discipline . . . 41200
9407.6 Relation to MetaCIC . . . 41201
9407.7 Open obligations and conditional statements . . . 41202
9407.8 Apophatic framing . . . 41202
9407.9 Boundary . . . 41202

9408 Theorem-DNA and Theorem Complexity Scoring 41204
9408.1 Role-complete theorem flows . . . 41204
9408.2 Statement, code, and quotient boundaries . . . 41205
9408.3 Mass vectors and weighted score . . . 41206
9408.4 Mutation and classifier-level information . . . 41207

9409 Theorem-Genome Evolution: Mutation, Recombination, Selection, and Discovery 41209
9409.1 Theorem genomes . . . 41209
9409.2 Complexity space . . . 41210
9409.3 Mutation and discovery . . . 41210
9409.4 Recombination and lineage . . . 41211
9409.5 Selection and potential . . . 41212
9409.6 Gate hierarchy . . . 41212

9410 Automated Theory-Discovery Pipeline Calculus 41214
9410.1 Pipeline states . . . 41214
9410.2 Transition classes . . . 41215
9410.3 Discovery claims . . . 41216
9410.4 Gate theorems . . . 41216
9410.5 Lineage and selection . . . 41218
9410.6 Pipeline adequacy . . . 41218

9411 Compiler-Closure Self-Compilation 41220
9411.1 From theorem DNA to compiler closure . . . 41220
9411.2 Compiler phenotype . . . 41221
9411.3 Certified expression . . . 41221
9411.4 Ledgered feedback . . . 41222
9411.5 Certified self-compilation loop . . . 41223
9411.6 Self-hosting is not self-compilation . . . 41223
9411.7 Discovery self-compilation . . . 41223
9411.8 Compiler-closure gate contract . . . 41224
9411.9 Pipeline adoption boundary . . . 41225

9412 Pseudotheory Refutation Gate Map 41227
9412.0 Coverage table . . . 41228
9412.1 Stateful and stateless refusal . . . 41229
9412.2 Prime asymmetry . . . 41229
9412.3 Carrier-repair boundary . . . 41229

9413 Common Carriers and Reality Projections 41230
9413.0 Reality Interfaces . . . 41230
9413.1 Common Carriers . . . 41231
9413.2 Ledgered Transport . . . 41231
9413.3 Generalized Continuation . . . 41232
9413.4 Noncollapse and Information Dimension . . . 41233
9413.5 Pythagorean Carrier, Scale Fibers, and Unit-Circle Projection . . . 41235
9413.6 Algebraic Extension Directions and Information Dimension . . . 41240
9413.7 Transcendental Extensions, Process Layers, and Apophatic Far Ends . . . 41244
9413.8 Spectral Curves as Common Carriers . . . 41248
9413.9 Recursive Spectral Zero Hierarchy . . . 41254
9413.10 Irreducible Information Layers and Spectral Factors . . . 41257
9413.11 No Far-End Absorption Boundary . . . 41261
9413.12 Finite Prime Tower as Completed Readout . . . 41263
9413.13 Far-End Unity and Prime-Window Normalization . . . 41267
9413.14 Far-End Correction Obligation . . . 41270
9413.15 From Spectral-Curve Laboratory to the RH Rigidity Route . . . 41274
9413.16 Phase, Transform, and Zeta Ladders . . . 41278
9413.17 Generalized Continuation Chain . . . 41281
9413.18 Inherited Invariants of the Zeta Endpoint . . . 41284
9413.19 Ontic Triple Invariants and the Zeta Projection . . . 41290
9413.20 Prime Phase-Radial Readback . . . 41295
9413.21 Observation-Classified Symmetry on the Critical Surface . . . 41301
9413.22 Ontic Core and Minimal Closure . . . 41305
9413.23 Object Induction and the RH Reality Route . . . 41308
9413.24 Triadic Structural Induction . . . 41312
9413.25 Circle–Triangle–Register Grammar . . . 41313
9413.26 Constructor Risk Surfaces . . . 41315
9413.27 Zeta Constructor Preservation Boundary . . . 41316
9413.28 Universal Generation Closure . . . 41317
9413.29 Zero Recursor and Cover Boundary . . . 41319
9413.30 Zero-Fibre Normal Sheet Geometry . . . 41321
9413.31 Time-Fibre Mirror Atomicity . . . 41323
9413.32 Zero Signature Faithfulness and the Mirror Slot Principle . . . 41326
9413.33 Core Commonality and Completion Boundaries . . . 41332
9413.34 Prime-Sieve Odd Exactness . . . 41336
9413.35 Recursive Parity Tower . . . 41341
9413.36 Parity Closure Tower Certificate . . . 41344
9413.37 Prime Charge, Parity, and Phase Dictionary . . . 41347
9413.38 Triadic Closure Tower Program . . . 41350
9413.39 Oriented Triadic Closure Tower Construction . . . 41353

9413.40 Harmonic Triadic Closure Map . . . 41355
9413.41 Zero-Knowledge Triadic Certificate Layer . . . 41358
9413.42 Analytic Projection and Closed Generability Boundary . . . 41361
9413.43 Oriented Triadic Certificate System . . . 41364
9413.44 Triadic Route Taxonomy . . . 41368
9413.45 BEDC RHRoute Formal Bridge . . . 41371
9413.46 Generation Invariant Exclusion Route . . . 41373
9413.47 Triadic Ontic Invariant Readback . . . 41375
9413.48 Zero Packet Refinement Category . . . 41380
9413.49 Prime-Adic Local Slicing . . . 41383
9413.50 Prime-Adic Parity Energy Window . . . 41385
9413.51 Prime-Adic Parity Sphere Invariant . . . 41388
9413.52 Zeta-Like Projection Taxonomy . . . 41390
9413.53 Global Prime Energy and the No-Hair Boundary . . . 41392
9413.54 Far-End Energy Unity Socket . . . 41395
9413.55 Finite-to-Infinite Phase-Energy Tower . . . 41397
9413.56 Infinite Prime Torus and Far-End Compatibility . . . 41400
9413.57 Far-End Identity Boundary . . . 41403
9413.58 Prime Spiral Projection Kernel . . . 41405
9413.59 Prime Resonance Chord and No-Hair Boundary . . . 41407
9413.60 Local Half-Split Certificate . . . 41412
9413.61 Endpoint Bisection Certificate . . . 41414
9413.62 Zeta Resonance Bound Stability . . . 41418
9413.63 Solenoid Lee–Yang Resonance Boundary . . . 41421
9413.64 Zeta Solenoid Lee–Yang Bound . . . 41424
9413.65 Solenoid Lee–Yang Source Gap . . . 41427
9413.66 Fixed-Prime Split Resonance Tower . . . 41429
9413.67 Dedekind Split-Tower Lift . . . 41435
9413.68 Deckendorf Solenoid Selector . . . 41438
9413.69 Conditional RH Sufficiency Boundary . . . 41441
9413.70 BEDC–Langlands Closure Packet . . . 41444
9413.71 BEDC–Langlands Existence Bridge . . . 41448
9413.72 BEDC–Langlands Unramified Local Layer . . . 41452
9413.73 BEDC–Langlands Finite Hecke Projector Certificate . . . 41456
9413.74 BEDC–Langlands Parameter State Certificate . . . 41460
9413.75 BEDC–Langlands Ramified Local State Closure . . . 41463
9413.76 BEDC–Langlands Local Packet Certificate for GL_n . . . 41468
9413.77 BEDC–Langlands Global Converse Bridge for GL_n . . . 41473
9413.78 BEDC–Langlands Tannakian Functorial Closure . . . 41477
9413.79 BEDC–Langlands Stable Multiplicity Formula Certificate . . . 41482
9413.80 BEDC–Langlands General G Existence Bridge . . . 41486
9413.81 BEDC–Langlands Compact Discrete Spectrum Core . . . 41490
9413.82 BEDC–Langlands Stable Trace Formula Certificate . . . 41494
9413.83 BEDC–Langlands Trace-Window Closure . . . 41498
9413.84 BEDC–Langlands Window-to-Global Bridge . . . 41502
9413.85 BEDC–Langlands No-Hidden Residual Master Lemma . . . 41505
9413.86 BEDC–Langlands Exact Globalization Contract . . . 41509

9413.37 BEDC–Langlands Main Closure Theorem . . . 41513
9413.38 BEDC–Langlands Base Case for GL_1 . . . 41516
9413.39 Unified Relation Atlas . . . 41520

9414 Reality-Constrained Approximation Towers 41525
9414.1 Reality-facing signatures . . . 41525
9414.2 Ledgered approximation towers . . . 41528
9414.3 Physical adequacy theorems . . . 41529
9414.4 Interface–constraint–section packets . . . 41532
9414.5 Self-similar operations and exponential readouts . . . 41541
9414.6 Zero-ledger rigidity route . . . 41547
9414.7 Layered projection theory . . . 41553
9414.8 Layered extension theory . . . 41559
9414.9 Cataphatic-apophatic layering . . . 41564
9414.10 Recursive orthogonal-axis normal forms . . . 41565
9414.11 Constructive information generation . . . 41569
9414.12 Quantum measurement as information generation . . . 41575
9414.13 Finite observation and open fit . . . 41578
9414.14 Open-fit theorems . . . 41579
9414.15 Ontic state as stable signature source . . . 41581
9414.16 Signature priority and exactness . . . 41583
9414.17 Residue and refinement . . . 41584
9414.18 Observer-internal access . . . 41585
9414.19 Certified ontic towers . . . 41585
9414.20 Approximation towers and ledgered compression . . . 41586
9414.21 Endpoint and composite ledger . . . 41587
9414.22 Layer-relative validity . . . 41588
9414.23 Hidden structure and tower collapse . . . 41589
9414.24 Polygon-circle tower and tower equivalence . . . 41590
9414.25 Physical induction and open continuation . . . 41591
9414.26 Mathematical and physical induction . . . 41592
9414.27 Raw certificates, reuse, and descent . . . 41593
9414.28 Open empirical continuation . . . 41595
9414.29 Falsification, intuition, and tower induction . . . 41595
9414.30 Objectivity without a privileged observer . . . 41597
9414.31 No global observer . . . 41598
9414.32 Objectivity as invariance . . . 41599
9414.33 Relational physical content . . . 41599
9414.34 Local time and synchronization . . . 41600
9414.35 Observer-independent constants . . . 41601
9414.36 Tower-aware objectivity . . . 41602
9414.37 Objectivity without finality . . . 41602
9414.38 The observation–reflection equivalence thesis . . . 41603
9414.39 Physical and formal reflective instances . . . 41605
9414.40 The equivalence thesis . . . 41606
9414.41 Architectural equivalence . . . 41607
9414.42 Internalization, reflection gaps, and failure surfaces . . . 41607

9414.43 NameCert as bridge concept . . . 41608
9414.44 Reflective towers and open self-description . . . 41609
9414.45 Equivalence diagram . . . 41610
9414.46 Consequences for explanation, realism, and idealization . . . 41611
9414.47 Certificate realism . . . 41613
9414.48 Effective theories . . . 41613
9414.49 Idealization as licensed compression . . . 41614
9414.50 Model pluralism and theory change . . . 41615
9414.51 Failure surfaces and ideal objects . . . 41616
9414.52 Scientific theory as certificate tower . . . 41617
9414.53 Scientific Inquiry as Ledgered Reflection of Hist-Time . . . 41618
9414.54 Scientific Inquiry, Consciousness-Like Reflection, and T-Supply Boundaries . . . 41620
9414.55 The Position of Consciousness . . . 41624
9414.56 Egocentric interface and mentation dynamics . . . 41630
9414.57 Public and Local Reflective Compilation . . . 41634
9414.58 AI Consciousness Candidate Schema . . . 41638
9414.59 Compiler, AI, and Human Consciousness . . . 41644
9414.60 A constructive research program for auditing physical models . . . 41647
9414.61 The model audit protocol . . . 41648
9414.62 Source, signature, and classifier audit . . . 41649
9414.63 Tower, ledger, and stability audit . . . 41650
9414.64 Descent audit . . . 41651
9414.65 Failure certificates and upgrades . . . 41652
9414.66 Audit table and polygon–circle witness . . . 41653
9414.67 Research workflow . . . 41654
9414.68 Final synthesis: physical truth as certified open fit . . . 41655
9414.69 The master decomposition . . . 41656
9414.70 Truth without finality . . . 41656
9414.71 Reality constraint and tower knowledge . . . 41657
9414.72 Reflective physical knowledge . . . 41658
9414.73 Certified physical truth schema . . . 41659
9414.74 Non-erasure discipline . . . 41659
9414.75 Closing thesis . . . 41660

9415 Hard Problems under the BEDC Frame: Causality, Free Will, Other Minds, Mind-Matter, Self-Reference 41662
9415.0 Causality as forward-binding commitment . . . 41662
9415.1 Free will and agency as inscription event . . . 41663
9415.2 Other-minds problem and inter-observer alignment . . . 41664
9415.3 Mind-matter relation as substrate/inscription duality . . . 41664
9415.4 Self-reference, Gödel-Tarski, and the reflection limit . . . 41665
9415.5 The unified table of hard problems . . . 41665
9415.6 Coordination with the three-axioms chapter . . . 41666
9415.7 Closure remarks . . . 41667

9416 Layered Relation Certificates 41668
9416.1 From Analogy to Certificate . . . 41668
9416.2 Certificate Shape . . . 41669
9416.3 Force Cognition as a Refinement Chain . . . 41670
9416.4 MetaCIC Self-Audit as a Refinement Chain . . . 41670
9416.5 Force–MetaCIC Relation . . . 41671
9416.6 Interfaces to Universe, Physics, and Consciousness . . . 41671
9416.7 Gate and Refusal Surface . . . 41672
9416.8 Closure . . . 41673

9417 The Cook Compile-Frontier as Engineering Path: From Principle to Witness 41675
9417.1 Setup: present frontier as engineering coordinate . . . 41675
9417.2 The compile pipeline, stage by stage . . . 41676
9417.3 Structural obstructions to advancing the frontier . . . 41676
9417.4 The epistemic value of crossing the frontier . . . 41676
9417.5 Failure modes of the engineering path . . . 41677
9417.6 The frontier as ongoing project artifact . . . 41677
9417.7 Mapping to BEDC engineering . . . 41678
9417.8 Closure remarks . . . 41678

9418 Observer-Hist Identity: The Implicit Metaphysics of the Finite Kernel 41680
9418.0 Observer history identity ledger . . . 41681
9418.1 Distinction as the source of information . . . 41682
9418.2 Observer as Hist self-reference loop . . . 41682
9418.3 Time as Hist growth . . . 41683
9418.4 No subject: the two-loops architecture . . . 41684
9418.5 Synthesis: the three-fold identity . . . 41685

9419 Inter-Hist Locality: From No God's-Eye View to Light-Speed Constancy 41686
9419.1 Ledger-local relations . . . 41686
9419.2 Multi-Hist configurations . . . 41687
9419.3 Inter-Hist invariants . . . 41688
9419.4 Cross-Hist causal dependence (schema-only) . . . 41688
9419.5 Roadmap toward physics . . . 41690
9419.6 Closing . . . 41693

9420 Consciousness, Universe, Observer: The Phenomenological Frame of the Closed Substrate 41694
9420.1 Finite records-side closure target . . . 41695
9420.2 Consciousness as closed observational system . . . 41696
9420.3 Embodied continuation and the mortality boundary . . . 41696
9420.4 Universe as observation history . . . 41698
9420.5 Closed history network reading . . . 41698
9420.6 Time as monotonic accumulation . . . 41700
9420.7 Focused still-point and information-clock rate . . . 41701
9420.8 Space as inter-observer coordination . . . 41702
9420.9 Love as other-Hist positive continuation . . . 41703
9420.10 Observer identity through history . . . 41704

9420.1 The hard problems in the closed-substrate frame . . . 41704
9420.2 Closure remarks . . . 41706

9421 Observer-Centered Logarithmic Section Dynamics 41707
9421.1 Frame Selection Data . . . 41707
9421.2 Orthogonal Section Step . . . 41708
9421.3 Logarithmic Time Readout . . . 41710
9421.4 Visible Stability and Solenoid Fibres . . . 41710
9421.5 Synthesis . . . 41712
9421.6 Constructor Certificate for OCLSD . . . 41712

9422 Quantum and Relativistic Section-Bundle Boundary 41715
9422.1 Vertical and Horizontal Faces . . . 41715
9422.2 Quantum Fibre Reading . . . 41716
9422.3 Relativistic Base Reading . . . 41717
9422.4 Unified Boundary . . . 41718

9423 Innovation-Soldering Metric Interface 41719
9423.1 Product Rows and Soldering Rows . . . 41719
9423.2 Continuous Innovation Form . . . 41720
9423.3 Tick Reflection and Pythagorean Coding Closure . . . 41721
9423.4 Discrete Stencils and Cartan Boundary . . . 41724
9423.5 Twisted-Holonomy Backreaction: A Degenerate Channel . . . 41725

9424 Innovation Input-Flow Geometry Boundary 41727
9424.1 Discrete Ledger Source . . . 41727
9424.2 Positive Innovation Gram Row . . . 41728
9424.3 Phase Packets and Curvature Budget . . . 41729
9424.4 Reflection-Positive Input Flow . . . 41731
9424.5 Interface Separation . . . 41733

9425 Innovation Input Lineage and Self-Reference Boundary 41734
9425.1 Lineage of the Refinement Row . . . 41734
9425.2 Self-Substitution Input . . . 41735
9425.3 Phase-Polygon Compatibility . . . 41736
9425.4 Externality Displacement . . . 41738
9425.5 Self-Observation and Arithmetic Habitat . . . 41738
9425.6 Parent Automaton and Larger Alphabets . . . 41739

9426 Innovation Input Self-Substitution Correlation Boundary 41741
9426.1 Layered Phase Correlations . . . 41741
9426.2 Carry Recursion . . . 41742
9426.3 Weak Reversibility Criterion . . . 41743
9426.4 Boundary Examples . . . 41744

9427 Innovation Input Finite-Block Reversibility Boundary 41746
9427.1 Signed Blocks . . . 41746
9427.2 Weak Reversibility Layers . . . 41747
9427.3 Finite-Block Carry Recursion . . . 41747
9427.4 Ternary Four-Point Test . . . 41748
9427.5 Layer Separation . . . 41751

9428 Innovation Input Finite-Window Reflection-Positive Boundary 41752
9428.1 Finite-Window Reflection . . . 41752
9428.2 Charged One-Site Obstruction . . . 41754
9428.3 Ternary Neutral RP Failure . . . 41754
9428.4 Quartic Neutral RP Failure . . . 41756
9428.5 Strict Layering . . . 41757

9429 Innovation Input Minimal RP Deficit Search Boundary 41758
9429.1 Q3 Search Space . . . 41758
9429.2 Neutral Window Deficits . . . 41759
9429.3 Link Deficit Gate . . . 41760
9429.4 Known Link Obstructions . . . 41761
9429.5 Exact Correlation Evaluation . . . 41762
9429.6 Search Objects . . . 41763

9430 Innovation Input Anchored RP Golden Palindrome Boundary 41764
9430.1 Anchored Window Discipline . . . 41764
9430.2 Pentary Palindromes . . . 41765
9430.3 Golden Palindrome Branches . . . 41765
9430.4 Anchored Neutral Window Audit . . . 41766

9431 Innovation Input Pentary Anchored Window Boundary 41769
9431.1 Anchored Pentary Window . . . 41769
9431.2 Pythagorean Pentary Row . . . 41771
9431.3 Anchored $L = 2$ Passage . . . 41771
9431.4 Anchored $L = 3$ Failure . . . 41772
9431.5 Anchor Dependence . . . 41773

9432 Innovation Input Pentary Hermitian Link Boundary 41775
9432.1 Pentary Edge Ratios . . . 41775
9432.2 Anchored $L = 3$ Link Diagonals . . . 41776
9432.3 Endpoint Gate . . . 41777
9432.4 Pythagorean Link Recheck . . . 41780

9433 Innovation Input Pentary Odd Endpoint Exclusion 41781
9433.1 Odd Endpoint Coordinates . . . 41781
9433.2 Real First-Correlation Bound . . . 41782
9433.3 Odd Endpoint Exclusion . . . 41784

9434 Innovation Input Nine-ary Real-Sign RP Screen 41786
9434.1 Nine-ary Real-Sign Rows . . . 41786
9434.2 Real Neutral Window . . . 41787
9434.3 Exact Carry Evaluator . . . 41787
9434.4 Two-Window Passage . . . 41788
9434.5 Three-Window Exclusion . . . 41789

9435 Innovation Input Nine-ary Complex Phase Link Formulas 41790
9435.1 Nine-ary Phase Locus . . . 41790
9435.2 Ordinary Link Correlations . . . 41791
9435.3 Anchored Link Diagonals . . . 41791
9435.4 Principal Reflection-Positive Conditions . . . 41793
9435.5 Pentary Contrast . . . 41793

9436 Innovation Input Square Real-Sign Four-Window Bottleneck 41795
9436.1 Twenty-five-ary Real-Sign Locus . . . 41795
9436.2 Four-Window Screen . . . 41796
9436.3 Concrete Witnesses . . . 41797
9436.4 Square-Family Pattern . . . 41798

9437 Innovation Input RP Solenoid Limit Conjecture 41799
9437.1 Finite-Window Horizon . . . 41799
9437.2 Finite Alphabet Finiteness . . . 41803
9437.3 Prime-Register Approximation . . . 41804
9437.4 Limit Matrix Problem . . . 41805

9438 Innovation Input Pentary Golden Hermitian Obstruction 41806
9438.1 Golden Symmetric Word . . . 41806
9438.2 Hermitian Obstruction . . . 41807

9439 Innovation Input Pentary Palindromic Hermitian Reduction 41810
9439.1 Reflection Evenness . . . 41810
9439.2 Palindromic Reflection . . . 41811
9439.3 Pentary Palindromic Q3 Curve . . . 41812
9439.4 Palindromic Link Formulas . . . 41813
9439.5 Known Rows Inside the Reduction . . . 41814

9440 Innovation Input Pentary Palindromic RP Exclusion 41816
9440.1 Literature Positioning . . . 41816
9440.2 Anchored $L = 2$ Matrix . . . 41816
9440.3 Half-Angle Curve . . . 41817
9440.4 Determinant Factorization . . . 41818
9440.5 Anchored $L = 2$ Rigidity . . . 41818
9440.6 Anchored $L = 3$ Exclusion . . . 41819

9441 Innovation Input Self-Similar RP Scaling 41821
9441.1 Self-Substitution Rule . . . 41821
9441.2 Scale Dilation . . . 41821
9441.3 Reflection-Centered Dilation . . . 41823
9441.4 RP Window Scaling . . . 41823
9441.5 Primitive RP Shapes . . . 41824

9442 Innovation Input Primitive RP Roots 41826
9442.1 Scale Root . . . 41826
9442.2 Window Roots . . . 41827
9442.3 Pentary Scale Test . . . 41827
9442.4 Small Pentary Windows . . . 41828
9442.5 Pentary Link Roots . . . 41829
9442.6 Odd-Length Address Formula . . . 41830

9443 Innovation Input Self-Adjoint Hamiltonian Gate 41831
9443.1 Three Operator Rows . . . 41831
9443.2 RP as the Input Source of Self-Adjointness . . . 41832
9443.3 Zeta Upgrade Obligation . . . 41833
9443.4 Pentary Hermitian Locus . . . 41833

9444 Innovation Input Pentary Mirror Decomposition Gate 41835
9444.1 Mirror Coordinates . . . 41835
9444.2 Mirror Transport of Correlations . . . 41837
9444.3 Mirror-Odd Hermitian Defects . . . 41838
9444.4 Palindromic Ideal . . . 41838
9444.5 Rank Form of the Remaining Problem . . . 41839

9445 Innovation Input Pythagorean Local Hermitian Rigidity 41841
9445.1 Antisymmetric Tangent Directions . . . 41841
9445.2 Defect Linearization . . . 41842
9445.3 Local Rigidity . . . 41843

9446 Innovation Input Pentary Hermitian Rigidity Gate 41844
9446.1 Hermitian Defect Row . . . 41844
9446.2 Cross-Entry Equations . . . 41845
9446.3 Finite Algebraic Evaluator . . . 41846
9446.4 Factor Extraction Gate . . . 41846
9446.5 Audit Pattern . . . 41847

9447 Innovation Input Dynamics and Spectral Boundary 41848
9447.1 Spatialized Input and Diffusion Read . . . 41848
9447.2 Relativistic Dispersion . . . 41849
9447.3 Stable Inputs and Spectral Dimension . . . 41850
9447.4 Dynamic Output Table . . . 41851

9448. Innovation Input Thermodynamic Geometry Boundary 41852
9448.1 Coherent Family and Meeting Law . . . 41852
9448.2 Heat Kernel Identity and Curvature Read . . . 41854
9448.3 Local Heat-Geometry Identity . . . 41855
9448.4 Entanglement Entropy Row . . . 41855
9448.5 Modular Hamiltonian and Unruh Row . . . 41856
9448.6 Jacobson Dictionary . . . 41857
9448.7 Scope and Correspondence Ledger . . . 41858

9449. Innovation Input Thermodynamic Geometry Verification 41859
9449.1 Cone Slot and the Two Twelfths . . . 41859
9449.2 Dual Curvature Instruments . . . 41860
9449.3 Regulator Scheme Comparison . . . 41860
9449.4 Three-Dimensional Ricci Tensor Read . . . 41861
9449.5 Homogeneous Next-Order Trace Channel . . . 41861
9449.6 Two-Dimensional Tensor Coefficient Read . . . 41862
9449.7 Spherical Consistency and Three-Dimensional Tensor Read . . . 41863
9449.8 Residual Discrimination by Grid Scaling . . . 41864

9450. Innovation Input Interaction and Slot-Response Boundary 41865
9450.1 First Interaction Row . . . 41865
9450.2 Non-Gaussian Remainder . . . 41866
9450.3 Slot-Response Audit . . . 41866
9450.4 Interaction Correspondence Ledger . . . 41867

9451. Innovation Input Solenoid and Arithmetic Boundary 41868
9451.1 Holonomy Tower . . . 41868
9451.2 Solenoid Exclusion Rows . . . 41870
9451.3 Denominator Clearing and Shape Blindness . . . 41870
9451.4 Scale and Charge Fingerprints . . . 41871
9451.5 Layering Verdict . . . 41872

9452. Innovation Input Modular Arithmetic Boundary 41873
9452.1 Thermodynamic Zeta Rows . . . 41873
9452.2 Complex Refinement and Chirality . . . 41874
9452.3 Class-Group Shape Registers . . . 41875
9452.4 Hecke Flow and Modular Spectra . . . 41876
9452.5 Zero Boundary . . . 41878

9453. Innovation Input Boundary Dynamics and Numerical Rows 41879
9453.1 Geodesic and Boundary Flow . . . 41879
9453.2 Gauss Map and Mayer Operator . . . 41880
9453.3 Boundary Reflection Positivity . . . 41880
9453.4 Bounded Type and Zaremba Boundary . . . 41881
9453.5 Congruence-Coupled Measurement Rows . . . 41881
9453.6 Boundary Obligation Ledger . . . 41883

9454 BEDC.MetaCIC is its own type theory: a framing principle 41884
9454.1 The framing correction . . . 41884
9454.2 What BEDC.MetaCIC positively is . . . 41884
9454.2.1 Closed generating types . . . 41884
9454.2.2 Literal AST type equality . . . 41884
9454.2.3 Structural recursors from closed generators . . . 41885
9454.2.4 Zero axiom, strict purity . . . 41885
9454.2.5 Apophatic external boundary . . . 41885
9454.3 The defining conjunction . . . 41885
9454.4 Why classical CIC is a different theory . . . 41886
9454.5 What comparison is good for . . . 41886
9454.6 What comparison is not good for . . . 41886
9454.7 The label classical CIC . . . 41887
9454.8 Implications for related chapters . . . 41887
9454.9 Apophatic framing of this principle . . . 41887
9454.10 Boundary . . . 41887

9455 BEDC as an axiom dependency map of classical CIC 41889
9455.1 The reframing . . . 41889
9455.2 Three modes of engagement . . . 41889
9455.2.1 Mode A: completely provable . . . 41889
9455.2.2 Mode B: partial discharge . . . 41890
9455.2.3 Mode C: counterexample plus named obligation . . . 41890
9455.3 The mapping is the deliverable . . . 41891
9455.4 How to read a dependency tuple . . . 41891
9455.5 A worked entry: closed application subject reduction . . . 41892
9455.6 Why this is more informative than classical CIC formalisation . . . 41892
9455.7 The mapped axiom families . . . 41893
9455.7.1 Conversion and definitional equality . . . 41893
9455.7.2 Extensional identification . . . 41893
9455.7.3 Choice and unwitnessed existence . . . 41893
9455.7.4 Mathematical induction as transcendental input . . . 41894
9455.8 Permanent and open-ended . . . 41894
9455.9 Cross-reference . . . 41894
9455.10 Boundary . . . 41895

9456 The Legitimate-Claim Problem for Finite Closed Systems 41896
9456.1 Finite expressions and closed claim substrates . . . 41896
9456.2 Claim status . . . 41897
9456.3 Dual audit bases . . . 41898
9456.4 Infinite claims and global deciders . . . 41899
9456.5 The legitimate-claim theorem . . . 41899
9456.6 RH and observer language as tests . . . 41900

9457 Russell Problem Boundary 41902
9457.1 The inherited problem . . . 41902
9457.2 Kernel reading . . . 41902
9457.3 The upward-name reading . . . 41903
9457.4 Boundary . . . 41903
9457.5 Analysis spine . . . 41904

9458 Reasoning by Contradiction: BEDC's Permitted Forms 41905
9458.1 Permitted refutation patterns . . . 41905
9458.2 Forbidden patterns and their meta-closure . . . 41906
9458.3 The boundary as ground vs meta loop . . . 41906

9459 Theory Stability and the TasteGate Framework 41909
9459.0 Carrier display obligations . . . 41909
9459.1 Lean interface . . . 41910
9459.2 Field faithfulness and nontriviality . . . 41910
9459.3 Relationship to the NameCert obligations . . . 41911
9459.4 Cannot-claim registry . . . 41911

9460 Theorem-DNA Scoring TasteGate 41913
9460.1 Theorem-DNA packet . . . 41913
9460.2 Score certificate . . . 41913
9460.3 The strengthened gate . . . 41914
9460.4 Exclusion principles . . . 41915
9460.5 Score and discovery . . . 41915

9461 Self-Reference in BEDC: The Ground Loop, the Reflection, and the Reflection Limit 41917
9461.1 The primitive distinction as generative loop . . . 41917
9461.2 Definition as classifier . . . 41918
9461.3 Internalisation of inductive structure . . . 41919
9461.4 BEDC describing its own meta-system . . . 41919
9461.5 The reflection limit . . . 41920
9461.6 The two loops of BEDC . . . 41920
9461.7 Position of this chapter in the BEDC architecture . . . 41922

9462 Auditable Generativity: A BEDC Philosophy 41927
9462.1 Distinction and History . . . 41927
9462.1.1 Primitive distinction . . . 41928
9462.1.2 Mark sameness . . . 41928
9462.1.3 History formation . . . 41929
9462.1.4 History sameness . . . 41930
9462.1.5 Minimal information carrier . . . 41931
9462.1.6 Prefix order and the preparatory form of time . . . 41931
9462.1.7 Extension, continuation, and unfolding . . . 41932
9462.1.8 Synthesis . . . 41933
9462.2 Classifier-Shift Information . . . 41933
9462.2.1 Compression, ledger, and conservation . . . 41935

9462.2.2 Machine-learning reading . . . 41936
9462.3 Distinction-to-NameCert Certification . . . 41937
9462.3.1 Semantic Assignment . . . 41938
9462.3.2 Hardening . . . 41938
9462.4 Tensor NameCert . . . 41939
9462.4.1 Black-Box Resolution . . . 41941
9462.4.2 Hardening and Gain . . . 41942
9462.5 Formal Verification as Hard-Certificate Layer . . . 41942
9462.5.1 Verification and Discovery . . . 41944
9462.5.2 Verification Transparency . . . 41944
9462.6 Hardenable and Ledger-Only Classifiers . . . 41945
9462.6.1 Classifier Strength Classes . . . 41946
9462.6.2 Priority . . . 41947
9462.6.3 Machine-Learning Reading . . . 41948
9462.7 Black Box as Missing Certificate . . . 41949
9462.7.1 Certificate Deficits . . . 41950
9462.7.2 Resolution . . . 41950
9462.7.3 Practice . . . 41951
9462.8 Black-Box Ledger Theory . . . 41952
9462.8.1 Ledger Kinds . . . 41953
9462.8.2 Obligations . . . 41954
9462.8.3 Repair and Discovery . . . 41954
9462.9 Neural Theorem-DNA . . . 41955
9462.10 Explanation Complexity and Interpretability Information . . . 41959
9462.11 Interpretability Classifier Novelty . . . 41964
9462.11.1 Nonclassifier Changes . . . 41966
9462.11.2 Classifier-Change Modes . . . 41967
9462.11.3 Discovery Grades . . . 41968
9462.11.4 NDNA, Debt, and Hardening . . . 41969
9462.12 Interpretability Classifier Novelty in Practice . . . 41970
9462.13 Positive Interpretability Discovery . . . 41972
9462.13.1 Cost Protocols . . . 41973
9462.13.2 Benefit Types . . . 41974
9462.13.3 Strong and NDNA-Positive Discovery . . . 41974
9462.13.4 Black-Box Resolution and Verification . . . 41975
9462.13.5 Lineage and Anti-Laundering . . . 41975
9462.14 Classifier-Forming Learning . . . 41976
9462.14.1 Separation from Metrics . . . 41977
9462.14.2 Memory, Generalization, and NameCert . . . 41978
9462.14.3 Representation and Transfer . . . 41979
9462.14.4 Effect Certification . . . 41980
9462.14.5 Lineage, Verification, and Anti-Laundering . . . 41981
9462.15 Certified Model Quality . . . 41983
9462.15.1 Separation from Performance . . . 41984
9462.15.2 Debt-Guided Training . . . 41984
9462.16 World-Model Classifier Certification . . . 41985
9462.17 Formal Verification in Certified Interpretability . . . 41987

9462.17 H Hardenable Components . . . 41988
9462.17 L2 Limits and Transcription . . . 41989
9462.17 V3 Verification Ledgers and Grades . . . 41990
9462.17 N4 DNA Verification and Discovery . . . 41991
9462.18 BEDC Decomposition of Machine-Learning Black Boxes . . . 41992
9462.18 D1 Decomposition States . . . 41993
9462.18 D2 Deficit Vectors and Frontiers . . . 41994
9462.18 N3 DNA and Formal Hardening . . . 41995
9462.18 V4 Value and Discovery . . . 41996
9462.18 P5 Practice Illusions . . . 41997
9462.19 No Global Black-Box Eliminator . . . 41998
9462.19 C1 Classifier and Semantic Globalization . . . 41999
9462.19 F2 Formal and Diagonal Residue . . . 42000
9462.19 C3 Cost and Free Explanation . . . 42001
9462.20 Unified Theory Closure . . . 42002
9462.20 U1 Unified Information and Discovery . . . 42003
9462.20 S2 Speakable Model Form . . . 42004
9462.20 S3 Specializations . . . 42005
9462.20 F4 False Discovery and Lineage . . . 42006
9462.20 U5 Unified Closure Theorems . . . 42007
9462.21 Unified Schematic Map . . . 42008
9462.21 B1 Black-Box and Information Schematics . . . 42009
9462.21 L2 Learning, Generalization, and Verification . . . 42010
9462.21 G3 Global and Final Schematics . . . 42011
9462.22 Observation, Subject, and Time . . . 42011
9462.22 I1 Internalizing observation . . . 42011
9462.22 O2 Observer identity . . . 42012
9462.22 O3 Observer as history . . . 42013
9462.22 T4 Time as history growth . . . 42014
9462.22 T5 The arrow of time . . . 42014
9462.22 N6 No global clock . . . 42015
9462.22 M7 Multiple observers and no God's-eye view . . . 42015
9462.22 O8 Observation, distinction, and time . . . 42016
9462.22 C9 Continuation as universal verb . . . 42017
9462.22 S10 Synthesis . . . 42017
9462.23 Objects, Knowledge, and Naming Certificates . . . 42018
9462.23 O1 Objects are not primitive . . . 42018
9462.23 T2 The five fields of a naming certificate . . . 42019
9462.23 P3 Paper certificates and finite-kernel certificates . . . 42019
9462.23 K4 Knowledge as witness . . . 42020
9462.23 P5 Packaging, compression, and gaps . . . 42021
9462.23 S6 Semantic naming and pattern-ledger transport . . . 42021
9462.23 C7 Certificate-organized clusters . . . 42022
9462.23 A8 Acceptance gates and strength . . . 42023
9462.23 S9 Synthesis . . . 42023
9462.24 Action Meaning Certificates . . . 42024
9462.25 Logic, Contradiction, and the Meta Loop . . . 42026

9462.25.1 The BEDC form of proof . . . 42026
9462.25.2 Permitted contradiction . . . 42027
9462.25.3 Prohibited contradiction . . . 42028
9462.25.4 Closed ground loop and open meta loop . . . 42028
9462.25.5 Self-description and truth boundary . . . 42029
9462.25.6 Three axioms, one meta closure . . . 42030
9462.25.7 Witness-preserving replacements . . . 42031
9462.25.8 Axiom-purity . . . 42032
9462.25.9 Halting as the recursive face of the open meta loop . . . 42032
9462.25.10 Type checking as classifier membership . . . 42033
9462.25.11 Synthesis . . . 42034
9462.25.12 Apophatic T-loops . . . 42034
9462.26 Caps, Transcendental Supply, and Boundary Sockets . . . 42037
9462.26.1 Closed observational systems . . . 42037
9462.26.2 Finite records and universal closure . . . 42038
9462.26.3 Forward-binding belief . . . 42039
9462.26.4 Negative certification and positive continuation . . . 42039
9462.26.5 Value direction and formal appearance . . . 42040
9462.26.6 Transcendental supply . . . 42041
9462.26.7 Apophatic naming and inscription . . . 42042
9462.26.8 Caps and audit gates . . . 42043
9462.26.9 Ω-supply sockets . . . 42044
9462.26.10 Seven socket kinds . . . 42044
9462.26.11 Non-collapse of socket kinds . . . 42045
9462.26.12 Socket reports . . . 42046
9462.26.13 Unity of transcendental supply . . . 42047
9462.26.14 Synthesis . . . 42048
9462.27 Consciousness, Universe, and Phenomenological Frame . . . 42048
9462.27.1 Finite focused multi-Hist ledgers . . . 42049
9462.27.2 Records-side consciousness . . . 42049
9462.27.3 Universe as observation-history slice . . . 42050
9462.27.4 Time as monotone accumulation . . . 42051
9462.27.5 Space as inter-observer coordination . . . 42051
9462.27.6 Identity as history trace . . . 42052
9462.27.7 Vision: Materiality as Orbit-Closure and Freedom-like Appearance as Orbit-Escape . . . 42052
9462.27.8 Hard problems as closure questions . . . 42057
9462.27.9 Causality as forward binding . . . 42057
9462.27.10 Free will as inscription event . . . 42058
9462.27.11 Other minds and cross-perspective alignment . . . 42058
9462.27.12 Mind and matter as double reading . . . 42059
9462.27.13 Self-reference as gap-on-gap . . . 42059
9462.27.14 Synthesis . . . 42060
9462.28 Scientific Objects, Physical Objects, and Model Bridges . . . 42060
9462.28.1 Scientific objects . . . 42061
9462.28.2 Physical records and relational content . . . 42061
9462.28.3 Mature mathematics and mature science . . . 42062

9462.28.P4 Physical laws as records and gaps . . . 42062
9462.28.M5 Maximum cross-history rate . . . 42063
9462.28.F6 Formal constants and empirical values . . . 42063
9462.28.M7 Models are not worlds . . . 42064
9462.28.S8 Standard bridges . . . 42064
9462.28.B9 Bridge conservativity and strength . . . 42065
9462.28.R10 Relative bridges . . . 42066
9462.28.P11 Probability Projection Bridge . . . 42066
9462.28.Q12 Quantum and Relativistic Model Bridges . . . 42069
9462.28.Y13 Yang–Mills mass-gap bridge . . . 42076
9462.28.Y14 Yang–Mills tube geometry certificates . . . 42082
9462.28.Y15 Yang–Mills RG scale-tail closure . . . 42087
9462.28.C16 Channel encoding and genetic-code bridge . . . 42090
9462.28.C17 Codons as six-bit finite windows . . . 42096
9462.28.S18 Stop codons as Zeckendorf-legal family suffixes . . . 42098
9462.28.R19 Research program for Zeckendorf-legal termination . . . 42102
9462.28.D20 DNA window-six symbolic-physical dynamics . . . 42106
9462.28.L21 Local tile dynamics of the genetic code . . . 42110
9462.28.C22 Codon-window spectral inverse dynamics . . . 42283
9462.28.B23 BEDC certificate machines, allegory machines, and biological branches . . . 42289
9462.28.C24 Codon-cube boundary research program . . . 42293
9462.28.F25 Failure certificates and cannot-claim entries . . . 42296
9462.28.P26 Prohibited scientific smuggling . . . 42297
9462.28.E27 External mature systems . . . 42297
9462.28.P28 Physics certificate chain . . . 42297
9462.28.S29 Synthesis . . . 42298
9462.29 BEDC and Philosophical Traditions . . . 42299
9462.29.S1 Structural adjacency . . . 42299
9462.29.S2 Spencer-Brown and primitive distinction . . . 42299
9462.29.B3 Brouwer and time intuition . . . 42300
9462.29.B4 Bundle theory, no-self, and cybernetics . . . 42300
9462.29.K5 Kant and conditions of possibility . . . 42301
9462.29.H6 Husserl and constitution . . . 42301
9462.29.W7 Wittgenstein and use . . . 42302
9462.29.W8 Whitehead and process . . . 42302
9462.29.T9 Tarski, Godel, and Muenchhausen . . . 42302
9462.29.R10 Relational physics . . . 42303
9462.29.C11 Composite positioning . . . 42304
9462.29.S12 Synthesis . . . 42304
9462.29.G13 Geometric Algebra as a Closure Model . . . 42305
9462.30 Audit Methodology and Writing Discipline . . . 42307
9462.30.W1 Writing items . . . 42307
9462.30.T2 Theory closure and formal verification . . . 42308
9462.30.C3 Closure status blocks . . . 42309
9462.30.D4 Definitions . . . 42309
9462.30.T5 Theorems and corollaries . . . 42310
9462.30.P6 Proofs . . . 42310

9462.30.7 Gap reports . . . 42311
9462.30.8 Cannot-claim registry . . . 42311
9462.30.9 Failure certificates . . . 42311
9462.30.10 Standard bridges . . . 42312
9462.30.11 Philosophical comparison . . . 42312
9462.30.12 Struggling . . . 42313
9462.30.13 Research template . . . 42313
9462.30.14 Review discipline . . . 42314
9462.30.15 Synthesis . . . 42314
9462.31 Total Synthesis: Auditable Generativity . . . 42314
9462.31.1 Total definition . . . 42315
9462.31.2 Total ontology . . . 42315
9462.31.3 Total epistemology . . . 42316
9462.31.4 Total observation theory . . . 42316
9462.31.5 Total logic . . . 42317
9462.31.6 Total boundary theory . . . 42318
9462.31.7 Total gap and far-end theory . . . 42318
9462.31.8 Total phenomenology . . . 42318
9462.31.9 Total philosophy of science . . . 42319
9462.31.10 Total bridge theory . . . 42320
9462.31.11 Total audit theory . . . 42320
9462.31.12 Total system diagram . . . 42320
9462.31.13 Auditable-generativity theorem . . . 42321
9462.31.14 Status record . . . 42322
9462.32 Research Programme and Implementation Route . . . 42322
9462.32.1 Definition of the research programme . . . 42322
9462.32.2 Strength gradient . . . 42323
9462.32.3 Layer route . . . 42323
9462.32.4 Finite kernel baseline . . . 42324
9462.32.5 Phase bundles . . . 42325
9462.32.6 Off-the-table claims . . . 42325
9462.32.7 Verification reporting . . . 42326
9462.32.8 Failure and downgrade . . . 42326
9462.32.9 Roadmap and cannot-claim . . . 42327
9462.32.10 Multi-history physics route . . . 42327
9462.32.11 Research programme synthesis . . . 42328
9462.32.12 Programme summary . . . 42328
9462.33 Formal Philosophy Package, Export Interface, and Governance . . . 42329
9462.33.1 Object of the philosophy package . . . 42329
9462.33.2 Concept registry . . . 42330
9462.33.3 Theorem map . . . 42330
9462.33.4 Gap map . . . 42331
9462.33.5 Tradition map . . . 42331
9462.33.6 Scientific map . . . 42332
9462.33.7 Philosophy cannot-claim registry . . . 42332
9462.33.8 Amendment policy . . . 42333
9462.33.9 Capstone policy . . . 42333

9462.33.10 Export interface . . . 42333
9462.33.11 Formal target draft . . . 42334
9462.33.12 Package synthesis . . . 42335
9462.33.13 Package compression . . . 42335
9462.34 Core Registries and Total Index . . . 42336
9462.34.1 Registry principles . . . 42336
9462.34.2 Concept registry . . . 42336
9462.34.3 Theorem registry . . . 42337
9462.34.4 Gap registry . . . 42338
9462.34.5 Tradition registry . . . 42339
9462.34.6 Science registry . . . 42340
9462.34.7 Cannot-claim registry . . . 42341
9462.34.8 Upgrade registry . . . 42342
9462.34.9 Subject index . . . 42343
9462.34.10 Registry synthesis . . . 42344
9462.34.11 Registry status . . . 42344
9462.35 Registry Layer, Manuscript Form, and Machine-Facing Interface . . . 42345
9462.35.1 Registry layer . . . 42345
9462.35.2 Manuscript form . . . 42345
9462.35.3 Export interface . . . 42346
9462.35.4 Registry consistency . . . 42346
9462.35.5 Machine-facing interface . . . 42347
9462.35.6 Socket type draft . . . 42347
9462.35.7 Amendment ledger . . . 42348
9462.35.8 Closure-status reference layer . . . 42348
9462.35.9 Interface synthesis . . . 42349
9462.35.10 Interface status . . . 42349

9463 Hyperbolic Phase Geometry of Observer Expansion . . . 42350
9463.1 Phase surfaces . . . 42356
9463.2 Reversible transport and Liouville conservation . . . 42357
9463.3 Poincare disk representation . . . 42358
9463.4 Hyperbolic observer expansion . . . 42358
9463.5 Disk dynamics and the fixed far end . . . 42359
9463.6 Consciousness fixed point as inscription . . . 42359
9463.7 Closure of the geometric layer . . . 42360

XII Open Conjectures . . . 42364

9464 Open Conjectures . . . 42365
9464.1 Finite-BHist-Injection Admits StdBridge . . . 42365

XIII Project Governance . . . 42401

9465 Project Governance . . . 42402

9466 Theory Amendment Policy **42403**
9466.1 Self-Compilation Surface Governance . . . 42403
9466.1.1 Claimed Strata . . . 42403
9466.1.2 Explicitly Not Claimed . . . 42404
9466.1.3 Open Strata . . . 42404
9466.2 Pure-CIC Tactic Discipline . . . 42404
9466.2.1 Admitted Tactics . . . 42404
9466.2.2 Forbidden Tactics . . . 42405
9466.2.3 Audit Enforcement . . . 42405
9466.2.4 Historical Origin . . . 42405
9466.2.5 Position . . . 42405

9467 Roadmap for Re-deriving Mature Mathematics **42406**
9467.1 Research thesis . . . 42406
9467.2 Closure and verification axes . . . 42406
9467.3 Anti-primitive principle . . . 42406
9467.4 Russell-style reconstruction spine . . . 42407
9467.5 Layer index . . . 42407
9467.6 L0–L5 finite kernel checked baseline . . . 42408
9467.7 L4 exact globalization contract . . . 42409
9467.8 L6 finite data certificates . . . 42409
9467.9 L7 arithmetic . . . 42409
9467.10 L8 algebra . . . 42409
9467.11 L9 order, logic, set-like, type-like . . . 42410
9467.12 L10 topology, metric, completion, real . . . 42410
9467.13 L10 topology continued and L11 analysis . . . 42412
9467.14 L12 categories, geometry, homology . . . 42412
9467.15 L13 number theory and spectra . . . 42413
9467.16 L14 conservative bridge to mature systems . . . 42414
9467.17 Cross-domain bridges via the unified kernel . . . 42414
9467.18 Per-object research template . . . 42415
9467.19 Near-term phases . . . 42415
9467.20 Claims explicitly off the table . . . 42417

XIV Appendices **42418**

A Lean Implementation Scaffold **42419**

B Verification Reporting Policy **42420**
B.1 Required fields for a checked report . . . 42420
B.2 Verification status . . . 42420
B.3 Rule 110 substrate witness . . . 42421
B.4 Relation to theoretical closure . . . 42422

Part I

Finite Kernel Theory

Chapter 1

Orientation and Layer Discipline

1.1 Purpose of This Core

The purpose of this core is not to extend the theory toward more traditional mathematical objects. Its purpose is to make the finite kernel more checkable. The kernel is centered on the following generated chain:

$$\mathsf{b0}, \mathsf{b1} \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask} \to \mathsf{Bundle} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{InGap} \to \mathsf{NameCert}.$$

The target is a minimal rule system in which derived interfaces are licensed by certificates rather than assumed.

Principle 1.1 (Layer isolation)**.** *The words used in this text to explain the kernel, such as relation, family, model, property, coverage, proof, and class, belong to the explanatory metalanguage. They are not object-language primitives. The object-language content consists only of the displayed generated judgments and certificates.*

Lean4 ✓ `BEDC.FKernel.Settled.layer isolation policy memory projection`

Principle 1.2 (No inherited mathematical primitives)**.** *No traditional mathematical interface is an internal primitive of the calculus. In particular, equality, function, natural number, addition, limit, real number, circle, and fold are derived interfaces only.*

Principle 1.3 (Definition demotion)**.** *A definition is not an initial source of objects. A definition is a naming certificate for a behavior that has already been generated and stabilized by the kernel.*

Principle 1.4 (Compression requires memory)**.** *Every operation that identifies, packages, hides, projects, seals, or otherwise compresses generated material must carry a gap ledger.*

Lean4 ✓ `BEDC.FKernel.Gap.compression requires memory`

1.2 Boundary of This Core

This document contains the finite kernel specification. The seal and completion interface is retained only as a thin placeholder. Interfaces such as $\mathsf{Real}^{\uparrow}$, $\mathsf{Circle}^{\uparrow}$, and $\mathsf{Fold}^{\uparrow}$ are mentioned only as derived names and motivation, not as fully reconstructed theories.

Chapter 2

Raw Marks and Mark-Sameness

Definition 2.1 (Raw marks)**.** *The only raw mark forms are*

$$\mathsf{b0}, \qquad \mathsf{b1}.$$

They are not the numbers 0 *and* 1*, not truth values, and not elements of a pre-existing set.*

Lean4 (definition) `BEDC.FKernel.Mark.BMark`
Lean4 ✓ `BEDC.FKernel.Mark.BMark generated cases`

Kernel Rule 2.2 (Mark formation)**.** *The judgment* $m : \mathsf{Mark}$ *is generated by exactly two formation rules:*

$$\frac{}{\mathsf{b0} : \mathsf{Mark}} \qquad \frac{}{\mathsf{b1} : \mathsf{Mark}}.$$

Definition 2.3 (Internal mark-sameness)**.** *The judgment* $\mathsf{msame}(m, n)$ *is generated by exactly two rules:*

$$\frac{}{\mathsf{msame}(\mathsf{b0}, \mathsf{b0})} \qquad \frac{}{\mathsf{msame}(\mathsf{b1}, \mathsf{b1})}.$$

No rule generates $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *or* $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$*.*

Lean4 (definition) `BEDC.FKernel.Mark.msame`
Lean4 ✓ `BEDC.FKernel.Mark.msame constructor characterization`
↪ variant `BEDC.FKernel.Mark.msame iff eq`
↪ variant `BEDC.FKernel.Mark.msame generated rules`
↪ variant `BEDC.FKernel.Mark.msame cases`

Closure Principle 2.4 (Closed generation for marks)**.** *The only generated marks are those generated by the two mark-formation rules. The only generated mark-sameness derivations are those generated by the two mark-sameness rules.*

Lemma 2.5 (Mark-sameness reflexivity)**.** *For every generated mark* m*,* $\mathsf{msame}(m, m)$*.*

Lean4 ✓ `BEDC.FKernel.Mark.msame refl`

Proof. By closed generation, m is either $\mathsf{b0}$ or $\mathsf{b1}$. Use the corresponding mark-sameness rule. □

Lemma 2.6 (Mark-sameness symmetry)**.** *If* $\mathsf{msame}(m, n)$*, then* $\mathsf{msame}(n, m)$*.*

Lean4 ✓ `BEDC.FKernel.Mark.msame symm`

Proof. The premise is generated only in the two reflexive cases. In either case, the conclusion is generated by the same rule. □

Lemma 2.7 (Mark-sameness transitivity)**.** *If* $\mathsf{msame}(m, n)$ *and* $\mathsf{msame}(n, p)$*, then* $\mathsf{msame}(m, p)$*.*

Lean4 ✓ `BEDC.FKernel.Mark.msame trans`

Proof. Closed generation leaves only two cases. If the middle mark is b0, then all generated mark-sameness premises force m and p to be b0. The b1 case is identical. □

Theorem 2.8 (Mark-sameness equivalence)**.** *The internal judgment* msame *is reflexive, symmetric, and transitive on generated raw marks.*

Lean4 ✓ `BEDC.FKernel.Mark.msame equivalence`
↪ variant `BEDC.FKernel.Mark.mark sameness equivalence`
↪ variant `BEDC.FKernel.Mark.msame internal equivalence spine`

Proof. Immediate from the preceding three lemmas. □

Theorem 2.9 (Raw mark no-confusion)**.** *The kernel does not generate* $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *or* $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$*.*

Lean4 ✓ `BEDC.FKernel.Mark.msame no confusion`
↪ variant `BEDC.FKernel.Mark.mark no confusion`
↪ variant `BEDC.FKernel.Mark.msame cross iff false`
↪ variant `BEDC.FKernel.Mark.msame cross impossible`

Proof. By the closed-generation principle, mark-sameness has only the two displayed constructors. Neither has cross-mark conclusion. □

Chapter 3

Emission Histories and History-Sameness

Definition 3.1 (Emission histories)**.** *Emission histories are generated by:*

$$\frac{}{\mathsf{e} : \mathsf{Hist}}$$

$$\frac{h : \mathsf{Hist}}{\mathsf{E0}(h) : \mathsf{Hist}} \qquad \frac{h : \mathsf{Hist}}{\mathsf{E1}(h) : \mathsf{Hist}}.$$

The symbols e*,* E0*, and* E1 *are object-language history constructors, not imported list constructors.*

Lean4 (definition) `BEDC.FKernel.Hist.BHist`

Definition 3.2 (History-sameness)**.** *The judgment* $\mathsf{hsame}(h, k)$ *is generated by:*

$$\frac{}{\mathsf{hsame}(\mathsf{e}, \mathsf{e})}$$

$$\frac{\mathsf{hsame}(h, k)}{\mathsf{hsame}(\mathsf{E0}(h), \mathsf{E0}(k))} \qquad \frac{\mathsf{hsame}(h, k)}{\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k))}.$$

There is no rule relating e *to a nonempty history, nor any rule relating* E0*-histories to* E1*-histories.*

Lean4 (definition) `BEDC.FKernel.Hist.hsame`

Lean4 ✓ `BEDC.FKernel.Hist.hsame iff eq`

↪ variant `BEDC.FKernel.Hist.hsame e0 congr`

↪ variant `BEDC.FKernel.Hist.hsame empty left iff`

Closure Principle 3.3 (Closed generation for histories)**.** *History formation and history-sameness are generated only by their displayed rules. Inversion and no-confusion principles are justified by this closed generation.*

Theorem 3.4 (History constructor characterization)**.** *If two generated histories are internally same, then they are both empty, both* E0*-histories with internally same predecessors, or both* E1*-histories with internally same predecessors. Conversely, any of those three constructor-aligned cases gives internal history-sameness.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame constructor characterization`

↪ variant `BEDC.FKernel.Hist.history constructor characterization`

↪ variant `BEDC.FKernel.Hist.hsame e0 left iff`

↪ variant `BEDC.FKernel.Hist.hsame e1 inversion iff`

Proof. For the forward implication, invert the internal sameness derivation and inspect the common outer constructor. For the reverse implication, eliminate the displayed alternatives and their witnesses, then rebuild the corresponding internal sameness judgment. □

Lemma 3.5 (History-sameness reflexivity)**.** *For every generated history h,* $\mathsf{hsame}(h, h)$.

Lean4 ✓ `BEDC.FKernel.Hist.hsame refl`

Proof. Induct on the generation of h. The empty case uses the empty rule. The two extension cases use the induction hypothesis and the corresponding constructor rule for hsame. □

Lemma 3.6 (History-sameness symmetry)**.** *If* $\mathsf{hsame}(h, k)$*, then* $\mathsf{hsame}(k, h)$.

Lean4 ✓ `BEDC.FKernel.Hist.hsame symm`

Proof. Induct on the derivation of $\mathsf{hsame}(h, k)$. Each constructor is symmetric by applying the induction hypothesis under the same history constructor. □

Lemma 3.7 (History-sameness transitivity)**.** *If* $\mathsf{hsame}(h, k)$ *and* $\mathsf{hsame}(k, l)$*, then* $\mathsf{hsame}(h, l)$.

Lean4 ✓ `BEDC.FKernel.Hist.hsame trans`

Proof. Induct on the first derivation. The empty case forces the second derivation to be the empty case by no-confusion. The $\mathsf{E0}$ case forces the second derivation to also be an $\mathsf{E0}$ case; apply the induction hypothesis to the inner histories and rebuild. The $\mathsf{E1}$ case is analogous. □

Theorem 3.8 (History-sameness equivalence)**.** *The internal judgment* hsame *is reflexive, symmetric, and transitive on generated histories.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame equivalence`
↪ variant `BEDC.FKernel.Hist.history sameness equivalence`

Proof. Combine the preceding lemmas. □

Theorem 3.9 (History no-confusion)**.** *The kernel does not generate any of the following cross-constructor identifications:*

$$\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(h)), \quad \mathsf{hsame}(\mathsf{E0}(h), \mathsf{e}), \quad \mathsf{hsame}(\mathsf{e}, \mathsf{E1}(h)),$$

$$\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e}), \quad \mathsf{hsame}(\mathsf{E0}(h), \mathsf{E1}(k)), \quad \mathsf{hsame}(\mathsf{E1}(h), \mathsf{E0}(k)).$$

Lean4 ✓ `BEDC.FKernel.Hist.hsame no confusion complete`

Proof. By closed generation, the only conclusions of hsame have matching outer constructors. The displayed cross-constructor conclusions are not produced by any rule. □

Theorem 3.10 (History extension is not self-same)**.** *No generated history is internally the same as its immediate* $\mathsf{E0}$*- or* $\mathsf{E1}$*-extension:*

$$\forall h : \mathsf{Hist},\ \neg\mathsf{hsame}(\mathsf{E0}(h), h) \qquad \textit{and} \qquad \forall h : \mathsf{Hist},\ \neg\mathsf{hsame}(\mathsf{E1}(h), h).$$

Proof. Induct on the target history. Empty and cross-constructor cases are excluded by history no-confusion. Matching extension cases reduce by the corresponding constructor inversion to the induction hypothesis on the shorter history. □

Lean4 ✓ `BEDC.FKernel.Hist.hsame extension self absurd`

Chapter 4

Relational Extension and Continuation

4.1 Relational extension

Definition 4.1 (Extension relation)**.** *The extension relation* $\mathsf{Ext}(m, h, r)$ *is generated by the two rules:*

$$\frac{}{\mathsf{Ext}(\mathsf{b0}, h, \mathsf{E0}(h))} \qquad \frac{}{\mathsf{Ext}(\mathsf{b1}, h, \mathsf{E1}(h))}.$$

It is a relation, not a function.

```
Lean4 ✓  BEDC.FKernel.Ext.ext generation rules
  ↪ variant  BEDC.FKernel.Ext.ext total
  ↪ variant  BEDC.FKernel.Ext.ext constructor inversion
  ↪ variant  BEDC.FKernel.Ext.ext constructor characterization
  ↪ variant  BEDC.FKernel.Ext.ext result for mark
  ↪ variant  BEDC.FKernel.Ext.ext b0 result hsame
  ↪ variant  BEDC.FKernel.Ext.ext b1 result hsame
  ↪ variant  BEDC.FKernel.Ext.ext mark deterministic from result
  ↪ variant  BEDC.FKernel.Ext.ext source deterministic from result
  ↪ variant  BEDC.FKernel.Ext.ext cross mark result impossible
```

Theorem 4.2 (Extension result constructor characterization)**.** *For all histories* h, t *and mark* m*,*

$$\mathsf{Ext}(m, h, \mathsf{E0}(t)) \iff (m = \mathsf{b0} \text{ and } \mathsf{hsame}(h, t)).$$

For the one-constructor case,

$$\mathsf{Ext}(m, h, \mathsf{E1}(t)) \iff (m = \mathsf{b1} \text{ and } \mathsf{hsame}(h, t)).$$

```
Lean4 ✓  BEDC.FKernel.Ext.ext result constructor iff
```

Proof. Invert the generated extension witness. In the zero-result branch, closed generation forces the emitted mark to be $\mathsf{b0}$ and the source to be the tail up to history sameness; the converse reconstructs the zero extension after eliminating those equalities. The one-result branch is identical with the $\mathsf{E1}$ constructor. □

Lemma 4.3 (Cross-mark extension results are separated)**.** *If* $\mathsf{Ext}(\mathsf{b0}, h, r_0)$*,* $\mathsf{Ext}(\mathsf{b1}, h, r_1)$*, and* $\mathsf{hsame}(r_0, r_1)$*, then contradiction follows.*

Lean4 ✓ `BEDC.FKernel.Ext.ext same source cross mark results not hsame`

Proof. Invert the two extension witnesses. The result histories are $\mathsf{E0}(h)$ and $\mathsf{E1}(h)$, and history no-confusion excludes $\mathsf{hsame}(\mathsf{E0}(h), \mathsf{E1}(h))$. □

Theorem 4.4 (Extension result is not the source)**.** *If* $\mathsf{Ext}(m, h, r)$*, then* $\mathsf{hsame}(r, h)$ *is impossible.*

Lean4 ✓ `BEDC.FKernel.Ext.ext result source not hsame`

Proof. Invert the generated extension witness. The result is either $\mathsf{E0}(h)$ or $\mathsf{E1}(h)$, and both cases are excluded by history constructor no-confusion for a one-constructor self-extension. □

Theorem 4.5 (Extension two-step cycle is impossible)**.** *If* $\mathsf{Ext}(m, h, r)$ *and* $\mathsf{Ext}(n, r, h)$*, then contradiction follows.*

Lean4 ✓ `BEDC.FKernel.Ext.ext two step cycle absurd`

Proof. Use the finite constructor length of histories. Each extension increases this length by one. Two extensions returning from h to h would force the length of h to equal its own double successor, which is impossible by natural-number constructor no-confusion. □

Theorem 4.6 (Extension result injectivity)**.** *If* $\mathsf{Ext}(m, h, r)$ *and* $\mathsf{Ext}(m', h', r)$*, then* $\mathsf{hsame}(h, h')$ *and* $\mathsf{msame}(m, m')$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext result injective pair`

Proof. Inspect the two generated extension witnesses. A common result can only arise from matching extension constructors, so the source histories and emitted marks are internally same. □

Theorem 4.7 (Extension result hsame injectivity)**.** *If* $\mathsf{Ext}(m, h, r)$*,* $\mathsf{Ext}(m', h', r')$*, and* $\mathsf{hsame}(r, r')$*, then* $\mathsf{hsame}(h, h')$ *and* $\mathsf{msame}(m, m')$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext result hsame injective pair`

Proof. Invert both generated extension witnesses. Matching outer constructors reduce the result-sameness witness to source history sameness and diagonal mark-sameness. Mixed outer constructors contradict history no-confusion. □

Lemma 4.8 (Extension respects mark-sameness)**.** *If* $\mathsf{msame}(m, n)$*,* $\mathsf{Ext}(m, h, r)$*, and* $\mathsf{Ext}(n, h, s)$*, then* $\mathsf{hsame}(r, s)$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext respects mark sameness`

Proof. By closed generation for msame, either both marks are $\mathsf{b0}$ or both are $\mathsf{b1}$. In the first case both extension results are $\mathsf{E0}(h)$; in the second case both are $\mathsf{E1}(h)$. Use reflexivity of hsame. □

Theorem 4.9 (Extension determinacy up to history-sameness)**.** *If* $\mathsf{Ext}(m, h, r)$ *and* $\mathsf{Ext}(m, h, s)$*, then* $\mathsf{hsame}(r, s)$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext deterministic`

Proof. Apply the previous lemma with $\mathsf{msame}(m, m)$, using mark-sameness reflexivity. □

4.2 Relational continuation

Definition 4.10 (Continuation relation)**.** *The continuation relation* $\mathsf{Cont}(h, k, r)$ *is generated by:*

$$\frac{}{\mathsf{Cont}(h, \mathsf{e}, h)}$$

$$\frac{\mathsf{Cont}(h, k, r)}{\mathsf{Cont}(h, \mathsf{E0}(k), \mathsf{E0}(r))} \qquad \frac{\mathsf{Cont}(h, k, r)}{\mathsf{Cont}(h, \mathsf{E1}(k), \mathsf{E1}(r))}.$$

It is a relation, not a primitive function.

Lean4 (definition) `BEDC.FKernel.Cont.Cont`

Lean4 ✓ `BEDC.FKernel.Cont.cont iff append`

↪ variant `BEDC.FKernel.Cont.cont relation generated rules`
↪ variant `BEDC.FKernel.Cont.cont right step rules`
↪ variant `BEDC.FKernel.Cont.cont step zero`
↪ variant `BEDC.FKernel.Cont.cont step zero iff`
↪ variant `BEDC.FKernel.Cont.cont step one`
↪ variant `BEDC.FKernel.Cont.cont step one iff`
↪ variant `BEDC.FKernel.Cont.cont step rules iff pair`
↪ variant `BEDC.FKernel.Cont.continuation step rules iff pair`
↪ variant `BEDC.FKernel.Cont.cont step rules pair`
↪ variant `BEDC.FKernel.Cont.continuation step rules`
↪ variant `BEDC.FKernel.Cont.cont step rules inversion pair`
↪ variant `BEDC.FKernel.Cont.cont step result inversions`
↪ variant `BEDC.FKernel.Cont.cont step result witnesses by constructor`
↪ variant `BEDC.FKernel.Cont.cont step result iff pair`
↪ variant `BEDC.FKernel.Cont.continuation step result witness iff pair`
↪ variant `BEDC.FKernel.Cont.continuation step result witnesses pair`
↪ variant `BEDC.FKernel.Cont.cont e0 result witness`
↪ variant `BEDC.FKernel.Cont.cont e1 result witness`
↪ variant `BEDC.FKernel.Cont.cont left unit result`
↪ variant `BEDC.FKernel.Cont.cont intro`
↪ variant `BEDC.FKernel.Cont.cont right constructor inversion`
↪ variant `BEDC.FKernel.Cont.cont relation constructor characterization`
↪ variant `BEDC.FKernel.Cont.cont empty result inversion`
↪ variant `BEDC.FKernel.Cont.cont empty result iff`
↪ variant `BEDC.FKernel.Cont.append eq empty iff`
↪ variant `BEDC.FKernel.Cont.cont e1 result inversion`
↪ variant `BEDC.FKernel.Cont.cont e0 result inversion`
↪ variant `BEDC.FKernel.Cont.cont e0 result iff`
↪ variant `BEDC.FKernel.Cont.cont e1 result iff`
↪ variant `BEDC.FKernel.Cont.cont respects hsame`

Theorem 4.11 (Append nonempty factor equivalence)**.** *For histories* h *and* k*, the appended history is not internally empty exactly when at least one factor is not internally empty:*

$$\neg\mathsf{hsame}(\mathsf{append}(h, k), \mathsf{e}) \iff \neg\mathsf{hsame}(h, \mathsf{e}) \vee \neg\mathsf{hsame}(k, \mathsf{e}).$$

Proof. The forward direction splits on the left factor: the empty branch reduces the obstruction to the right factor, while visible left constructors are separated from e. Conversely, empty-result inversion contradicts either nonempty disjunct. □

Lean4 ✓ `BEDC.FKernel.Cont.append nonempty iff`

Theorem 4.12 (Append visible-left empty exclusion)**.** *For histories* $h, k : \mathsf{BHist}$*, appending any right tail to a visible left endpoint cannot produce an internally empty history: both* $\mathsf{hsame}(\mathsf{append}(\mathsf{E0}(h), k), \mathsf{e}) \to \bot$ *and* $\mathsf{hsame}(\mathsf{append}(\mathsf{E1}(h), k), \mathsf{e}) \to \bot$ *hold.*

Proof. Split on the right tail k. If it is empty, the visible left endpoint remains the outer constructor. If it is visible, append exposes that right-tail constructor as the outer constructor. In every branch, constructor separation from e rules out the displayed history sameness. □

Lean4 ✓ `BEDC.FKernel.Cont.append visible left empty absurd`

Theorem 4.13 (Append visible-right empty exclusion)**.** *For histories* $h, k : \mathsf{BHist}$*, appending a visible right tail cannot produce an internally empty history: both* $\mathsf{hsame}(\mathsf{append}(h, \mathsf{E0}(k)), \mathsf{e}) \to \bot$ *and* $\mathsf{hsame}(\mathsf{append}(h, \mathsf{E1}(k)), \mathsf{e}) \to \bot$ *hold.*

Proof. Appending a visible right tail exposes that visible constructor at the result. Thus the result has outer constructor E0 or E1, respectively.

In both cases, history no-confusion with e rules out the displayed history sameness. □

Lean4 ✓ `BEDC.FKernel.Cont.append visible right empty absurd`

Theorem 4.14 (Append visible-left same-tail tag separation)**.** *For histories* $h_0, h_1, k : \mathsf{BHist}$*, common right-tail append preserves separation between zero-headed and one-headed visible left endpoints:*

$$\mathsf{hsame}(\mathsf{append}(\mathsf{E0}(h_0), k), \mathsf{append}(\mathsf{E1}(h_1), k)) \to \bot.$$

Proof. Induct on the common right tail k. The empty case is direct zero/one constructor no-confusion. In each visible tail step, both appended histories gain the same outer constructor; constructor injectivity peels that shared outer layer and the induction hypothesis applies to the shorter tail. □

Lean4 ✓ `BEDC.FKernel.Cont.append visible left same tail tag separation`

Theorem 4.15 (Append visible-left same-tag tail cancellation)**.** *For histories* $h_0, h_1, k : \mathsf{BHist}$*, common right-tail append cancellation identifies the payloads of equal zero-headed visible-left endpoints, and likewise for equal one-headed visible-left endpoints.*

Proof. Apply append right-cancellation to the common tail, then use constructor injectivity for the matching E0 or E1 head. □

Lean4 ✓ `BEDC.FKernel.Cont.append visible left same tag tail cancel`

Lemma 4.16 (Continuation respects a matching right extension step)**.** *If* $\mathsf{Cont}(h, k, r)$*,* $\mathsf{Ext}(m, k, k')$*, and* $\mathsf{Ext}(m, r, r')$*, then* $\mathsf{Cont}(h, k', r')$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont ext right step`

Proof. Case split on the extension of k by m. In the E0 branch, the matching extension of r is also the zero branch, and the zero continuation step applies. The E1 branch is analogous. □

Theorem 4.17 (Extension as singleton continuation)**.** *For every history h and result r,* $\mathsf{Ext}(\mathsf{b0}, h, r)$ *if and only if* $\mathsf{Cont}(h, \mathsf{E0}(\mathsf{e}), r)$*, and* $\mathsf{Ext}(\mathsf{b1}, h, r)$ *if and only if* $\mathsf{Cont}(h, \mathsf{E1}(\mathsf{e}), r)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.extension as singleton continuation`
Lean4 ✓ `BEDC.FKernel.Cont.ext singleton continuation iff pair`

Proof. For the forward zero branch, invert the extension witness to identify r with $\mathsf{E0}(h)$, use the empty-continuation rule $\mathsf{Cont}(h, \mathsf{e}, h)$, and apply the $\mathsf{E0}$ continuation step rule. The one branch is the same argument with $\mathsf{E1}$.

For the reverse zero branch, invert the continuation witness for $\mathsf{Cont}(h, \mathsf{E0}(\mathsf{e}), r)$. Step-result inversion gives a predecessor continued by e, and history constructor no-confusion forces the singleton history to be the displayed $\mathsf{E0}(\mathsf{e})$ branch. The empty-continuation rule identifies the predecessor with h, so the $\mathsf{b0}$ extension constructor yields $\mathsf{Ext}(\mathsf{b0}, h, r)$. The one branch uses the corresponding $\mathsf{E1}$ inversion and the $\mathsf{b1}$ extension constructor. □

Theorem 4.18 (Continuation step result no-confusion)**.** *No generated history r can be both the result of continuing by* $\mathsf{E0}(k_0)$ *and the result of continuing by* $\mathsf{E1}(k_1)$ *from the same source history. The converse ordering of the two step constructors is equally impossible.*

Lean4 ✓ `BEDC.FKernel.Cont.cont step result no confusion pair`

Proof. Invert the two continuation witnesses. The zero-step witness forces the common result to have outer constructor $\mathsf{E0}$, while the one-step witness forces the same result to have outer constructor $\mathsf{E1}$. Closed generation of histories gives constructor no-confusion. The converse direction is symmetric. □

Theorem 4.19 (Continuation step result is nonempty)**.** *For all histories h, k, neither* $\mathsf{Cont}(h, \mathsf{E0}(k), \mathsf{e})$ *nor* $\mathsf{Cont}(h, \mathsf{E1}(k), \mathsf{e})$ *can hold.*

Lean4 ✓ `BEDC.FKernel.Cont.cont step result not empty pair`

Proof. Unfold the step result shape for Cont. The zero branch would identify e with $\mathsf{E0}(r)$, and the one branch would identify e with $\mathsf{E1}(r)$. Both contradict the generated history constructors. □

Theorem 4.20 (Zero-headed continuation result characterization)**.** *For histories $h, k, r : \mathsf{BHist}$,*

$$\mathsf{Cont}(h, k, \mathsf{E0}(r)) \iff (k = \mathsf{e} \wedge \mathsf{hsame}(h, \mathsf{E0}(r))) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E0}(k_0) \wedge \mathsf{Cont}(h, k_0, r)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont result e0 cases iff`

Proof. Classify the right continuation history k. The empty branch is exactly the right-unit endpoint equality between h and $\mathsf{E0}(r)$. A zero-headed tail exposes its predecessor and removes the matching outer constructor from the continuation equation. A one-headed tail would identify zero-headed and one-headed generated histories. Conversely, the empty branch is the right-unit continuation after the displayed history sameness is eliminated, and the zero-headed branch is rebuilt by the zero step rule. □

Theorem 4.21 (One-headed continuation result characterization)**.** *For histories $h, k, r : \mathsf{BHist}$,*

$$\mathsf{Cont}(h, k, \mathsf{E1}(r)) \iff (k = \mathsf{e} \wedge \mathsf{hsame}(h, \mathsf{E1}(r))) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \wedge \mathsf{Cont}(h, k_0, r)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont result e1 cases iff`

Proof. Classify the right continuation history k. The empty branch is exactly the right-unit endpoint equality between h and $\mathsf{E1}(r)$. A zero-headed tail would identify one-headed and zero-headed generated histories. A one-headed tail exposes its predecessor and removes the matching outer constructor from the continuation equation. Conversely, the empty branch is the right-unit continuation after the displayed history sameness is eliminated, and the one-headed branch is rebuilt by the one step rule. □

Theorem 4.22 (Zero-headed continuation result up to history-sameness)**.** *For histories* $h, k, r, z : \mathsf{BHist}$*, if* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{hsame}(r, \mathsf{E0}(z))$*, then*

$$(k = \mathsf{e} \wedge \mathsf{hsame}(h, \mathsf{E0}(z))) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E0}(k_0) \wedge \mathsf{Cont}(h, k_0, z)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont result hsame e0 cases`

Proof. Transport the continuation result along the displayed history-sameness to obtain $\mathsf{Cont}(h, k, \mathsf{E0}(z))$. The zero-headed result characterization then gives exactly the two alternatives. □

Theorem 4.23 (One-headed continuation result up to history-sameness)**.** *For histories* $h, k, r, z : \mathsf{BHist}$*, if* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{hsame}(r, \mathsf{E1}(z))$*, then*

$$(k = \mathsf{e} \wedge \mathsf{hsame}(h, \mathsf{E1}(z))) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \wedge \mathsf{Cont}(h, k_0, z)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont result hsame e1 cases`

Proof. Transport the continuation result along the displayed history-sameness to obtain $\mathsf{Cont}(h, k, \mathsf{E1}(z))$. The one-headed result characterization then gives exactly the two alternatives. □

Theorem 4.24 (Left zero-headed continuation result characterization)**.** *For histories* $h, k, r : \mathsf{BHist}$*,*

$$\mathsf{Cont}(\mathsf{E0}(h), k, \mathsf{E0}(r)) \iff (k = \mathsf{e} \wedge \mathsf{hsame}(h, r)) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E0}(k_0) \wedge \mathsf{Cont}(\mathsf{E0}(h), k_0, r)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont left e0 result iff`

Proof. Classify the right continuation history k. The empty branch reduces the outer zero result equality to equality of payload histories. A zero-headed k exposes its predecessor and removes the matching outer constructor from the continuation equation. A one-headed k would identify zero-headed and one-headed generated histories. Conversely, the empty branch is the right-unit continuation after the displayed history sameness is eliminated, and the zero-headed branch is rebuilt by the zero step rule. □

Proposition 4.25 (Left zero-headed continuation result visible-tail exposure)**.** *For histories* $h, k, r : \mathsf{BHist}$*, if* $\mathsf{Cont}(\mathsf{E0}(h), k, \mathsf{E0}(r))$ *holds and* k *is not classified with* e*, then the continuation tail has visible zero-headed form:*

$$\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E0}(k_0) \wedge \mathsf{Cont}(\mathsf{E0}(h), k_0, r).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont left e0 result nonempty tail cases`

Proof. Apply the zero-headed result characterization. Its empty-tail branch contradicts the displayed nonempty-tail hypothesis, so only the visible zero-headed tail branch remains. □

Theorem 4.26 (Continuation result tag separation)**.** *There are no histories h, k, r_0, r_1 such that both $\mathsf{Cont}(h, k, \mathsf{E0}(r_0))$ and $\mathsf{Cont}(h, k, \mathsf{E1}(r_1))$ hold.*

Lean4 ✓ `BEDC.FKernel.Cont.cont result tag separation`

Proof. Assume both continuation witnesses for the same inputs h and k. Continuation determinacy gives $\mathsf{hsame}(\mathsf{E0}(r_0), \mathsf{E1}(r_1))$. History no-confusion rules out any generated sameness derivation between a zero-tagged history and a one-tagged history, so the two witnesses cannot coexist. □

Theorem 4.27 (Left one-headed continuation result cases)**.** *If $\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E1}(r))$, then either $k = \mathsf{e}$ and $\mathsf{hsame}(h, r)$, or $k = \mathsf{E1}(k_0)$ for some k_0 and $\mathsf{Cont}(\mathsf{E1}(h), k_0, r)$.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left e1 result cases`

Proof. Classify the right continuation history k. The empty case reduces the result equation to equality of the inner histories. A zero-headed k would force a one-headed history to equal a zero-headed history, contradicting constructor no-confusion. A one-headed k exposes its predecessor k_0 and removes the matching outer constructor from the continuation equation. □

Theorem 4.28 (Left one-headed continuation result characterization)**.** *For histories $h, k, r : \mathsf{BHist}$,*

$$\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E1}(r)) \iff (k = \mathsf{e} \wedge \mathsf{hsame}(h, r)) \vee (\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \wedge \mathsf{Cont}(\mathsf{E1}(h), k_0, r)).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont left e1 result iff`

Proof. The forward direction is the one-headed result case split. For the converse, the empty branch is the right-unit continuation after the displayed history sameness is eliminated, and the one-headed branch is rebuilt by the one step rule. □

Proposition 4.29 (Left one-headed continuation result visible-tail exposure)**.** *For histories $h, k, r : \mathsf{BHist}$, if $\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E1}(r))$ holds and k is not classified with e, then the continuation tail has visible one-headed form:*

$$\exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \wedge \mathsf{Cont}(\mathsf{E1}(h), k_0, r).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont left e1 result nonempty tail cases`

Proof. Apply the one-headed result case split. Its empty-tail branch contradicts the displayed nonempty-tail hypothesis, so the visible one-headed tail branch is the remaining case. □

Theorem 4.30 (Left cross-tag continuation result cases)**.** *If $\mathsf{Cont}(\mathsf{E0}(h), k, \mathsf{E1}(r))$, then $k = \mathsf{E1}(k_0)$ for some k_0 and $\mathsf{Cont}(\mathsf{E0}(h), k_0, r)$. If $\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E0}(r))$, then $k = \mathsf{E0}(k_0)$ for some k_0 and $\mathsf{Cont}(\mathsf{E1}(h), k_0, r)$.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left tag cross result cases`

Proof. In each direction, classify k by generated-history constructors. The empty case and the branch whose tag matches the left source produce impossible outer-constructor equalities. The opposite-tag branch exposes the predecessor of k and peels the matching outer constructor from the result equation. □

Theorem 4.31 (Left cross-tag continuation result characterization)**.** *For histories h, k, r, the two cross-tag result shapes are characterized by the opposite tag in the continuation tail:*

$$\mathsf{Cont}(\mathsf{E0}(h), k, \mathsf{E1}(r)) \iff \exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \wedge \mathsf{Cont}(\mathsf{E0}(h), k_0, r),$$

and

$$\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E0}(r)) \iff \exists k_0 : \mathsf{BHist}.\ k = \mathsf{E0}(k_0) \wedge \mathsf{Cont}(\mathsf{E1}(h), k_0, r).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont left tag cross result iff pair`

Proof. The forward directions are the cross-tag result case split. For the converse directions, eliminate the displayed equation for k and rebuild the outer result constructor using the corresponding continuation step rule. □

Theorem 4.32 (Left same-tag continuation tail alignment)**.** *If two continuations have the same visible left endpoint and the same same-tag visible result, then their right tails align in the same shape. For zero-headed endpoints, both tails are empty or both are zero-headed with* hsame *exposed tails; for one-headed endpoints, both tails are empty or both are one-headed with* hsame *exposed tails.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left same tag tail alignment`

Proof. Apply continuation left cancellation to the two displayed continuations to obtain $\mathsf{hsame}(k, l)$ between their right tails. Split the first continuation by the same-tag result case theorem. In the empty branch, the tail sameness forces the second tail to be empty. In the visible branch, tail sameness transports the second tail into the matching visible constructor and yields the exposed hsame relation between payload tails. □

Theorem 4.33 (Left cross-tag continuation tail alignment)**.** *If two continuations have the same visible left endpoint and the same cross-tag visible result, then both right tails carry the opposite visible tag, and the exposed tails are* hsame*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left tag cross tail alignment`

Proof. Continuation left cancellation compares the two right tails. The cross-tag result case theorem applied to the first continuation forces that tail to have the opposite visible constructor. Transporting the second tail through the tail sameness forces the same constructor there as well, and constructor inversion gives the exposed hsame relation. □

Theorem 4.34 (Continuation right-tail result cases)**.** *For histories $h, k, r : \mathsf{BHist}$, if $\mathsf{Cont}(h, k, r)$, then the right continuation tail has one of the following displayed forms:*

$$\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(r, h),$$

or there exist histories t, u such that

$$\mathsf{hsame}(k, \mathsf{E0}(t)) \wedge \mathsf{hsame}(r, \mathsf{E0}(u)) \wedge \mathsf{Cont}(h, t, u),$$

or there exist histories t, u such that

$$\mathsf{hsame}(k, \mathsf{E1}(t)) \wedge \mathsf{hsame}(r, \mathsf{E1}(u)) \wedge \mathsf{Cont}(h, t, u).$$

Proof. Classify the right continuation tail k. The empty case is the right-unit continuation, so the result is classified with the source. In the zero-headed and one-headed cases, step-result inversion removes the displayed outer constructor from the result and exposes the residual continuation through the tail payload. □

Lean4 ✓ `BEDC.FKernel.Cont.cont right tail result cases`

Lemma 4.35 (Continuation nonempty right-tail result cases)**.** *For histories h, k, r : BHist, if $\mathsf{Cont}(h, k, r)$ and k is not history-same to e, then the empty-tail branch of the right-tail result classification is impossible. Hence either*

$$\exists t, u : \mathsf{BHist},\ \mathsf{hsame}(k, \mathsf{E0}(t)) \wedge \mathsf{hsame}(r, \mathsf{E0}(u)) \wedge \mathsf{Cont}(h, t, u),$$

or

$$\exists t, u : \mathsf{BHist},\ \mathsf{hsame}(k, \mathsf{E1}(t)) \wedge \mathsf{hsame}(r, \mathsf{E1}(u)) \wedge \mathsf{Cont}(h, t, u).$$

Proof. Apply Definition 4.34 to the continuation witness. The empty-tail branch gives $\mathsf{hsame}(k, \mathsf{e})$ and is eliminated by the nonempty hypothesis. The remaining two branches are already the zero-headed and one-headed witness alternatives, so they are returned unchanged. □

Lean4 ✓ `BEDC.FKernel.Cont.cont right tail result nonempty cases`

Lemma 4.36 (Continuation right unit)**.** *For every history h, $\mathsf{Cont}(h, \mathsf{e}, h)$.*

Proof. This is the base rule of Cont. □

Lemma 4.37 (Continuation left unit)**.** *For every history h, $\mathsf{Cont}(\mathsf{e}, h, h)$.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left unit`

Proof. Induct on h. The empty case is the base rule. The extension cases follow by applying the corresponding continuation step rule to the induction hypothesis. □

Theorem 4.38 (Continuation unit laws)**.** *The empty history is a left unit for continuation, and continuation by the empty history returns a history same as the source.*

Lean4 ✓ `BEDC.FKernel.Cont.cont unit laws`

```
↪ variant  BEDC.FKernel.Cont.append empty left
↪ variant  BEDC.FKernel.Cont.append empty right
↪ variant  BEDC.FKernel.Cont.append left unit iff
↪ variant  BEDC.FKernel.Cont.append right unit iff
↪ variant  BEDC.FKernel.Cont.cont left unit iff
↪ variant  BEDC.FKernel.Cont.cont left unit unique
↪ variant  BEDC.FKernel.Cont.cont right unit result
↪ variant  BEDC.FKernel.Cont.cont right unit unique
↪ variant  BEDC.FKernel.Cont.cont unit uniqueness pair
↪ variant  BEDC.FKernel.Cont.cont unit laws with uniqueness
```

Proof. The left-unit statement is the left-unit lemma. For the right-unit statement, compare the given continuation proof with the right-unit lemma by continuation determinacy. □

Theorem 4.39 (Continuation determinacy up to history-sameness)**.** *If $\mathsf{Cont}(h, k, r)$ and $\mathsf{Cont}(h, k, s)$, then $\mathsf{hsame}(r, s)$.*

Lean4 ✓ `BEDC.FKernel.Cont.cont deterministic`

Proof. Induct on the first derivation. In the empty case the second derivation is also the empty case by no-confusion, and the result follows by reflexivity. In the two extension cases, no-confusion forces the second derivation to use the same outer continuation constructor; apply the induction hypothesis and rebuild hsame with the matching history constructor. □

Theorem 4.40 (Continuation result transport)**.** *If* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{hsame}(r, r')$*, then* $\mathsf{Cont}(h, k, r')$*.*

Proof. Unfold Cont as the append-result equation. Symmetry of the result sameness followed by the original continuation equation gives the required append-result equation for r'. □

Lean4 ✓ `BEDC.FKernel.Cont.cont result hsame transport`

Theorem 4.41 (Continuation hsame transport)**.** *If* $\mathsf{hsame}(h, h')$*,* $\mathsf{hsame}(k, k')$*,* $\mathsf{hsame}(r, r')$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{Cont}(h', k', r')$*.*

Proof. Eliminate the two input hsame witnesses to align the left and right continuation inputs. The remaining result endpoint is transported by Theorem 4.40. □

Lean4 ✓ `BEDC.FKernel.Cont.cont hsame transport`

Theorem 4.42 (Continuation transport result classification)**.** *If* $\mathsf{hsame}(h, h')$*,* $\mathsf{hsame}(k, k')$*,* $\mathsf{hsame}(r, r')$*,* $\mathsf{Cont}(h, k, r)$*, and* $\mathsf{Cont}(h', k', s)$*, then* $\mathsf{Cont}(h', k', r')$ *and* $\mathsf{hsame}(r', s)$*.*

Proof. First transport the original continuation witness across the two input endpoints and the result endpoint. The transported witness and the displayed target-side witness now share the same source and tail histories, so continuation determinacy classifies their results. □

Lean4 ✓ `BEDC.FKernel.Cont.cont transport result classified`

Theorem 4.43 (Continuation left-context cancellation up to history-sameness)**.** *If* $\mathsf{Cont}(a, b, r)$*,* $\mathsf{Cont}(a', d, r')$*,* $\mathsf{hsame}(a, a')$*, and* $\mathsf{hsame}(r, r')$*, then* $\mathsf{hsame}(b, d)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont cancel hsame left context`

Proof. Use $\mathsf{hsame}(a, a')$ to align the left context and $\mathsf{hsame}(r, r')$ to align the result. The two continuation derivations then have the same left context and the same result, so left-context cancellation gives $\mathsf{hsame}(b, d)$. □

Theorem 4.44 (Prefixed result common-suffix source determinacy)**.** *For histories* p, a, b, k, r, s : BHist*, if* $\mathsf{Cont}(a, k, r)$*,* $\mathsf{Cont}(b, k, s)$*, and*

$$\mathsf{hsame}(\mathsf{append}(p, r), \mathsf{append}(p, s)),$$

then $\mathsf{hsame}(a, b)$*.*

Proof. Left cancellation removes the visible prefix p from the result comparison. The two continuations then have a common right suffix k and history-same results, so right-context continuation cancellation identifies the sources. □

Lean4 ✓ `BEDC.FKernel.Cont.cont prefixed result common suffix source deterministic`

Theorem 4.45 (Prefixed result common-source tail determinacy)**.** *For histories* p, a, k, l, r, s : BHist*, if* $\mathsf{Cont}(a, k, r)$*,* $\mathsf{Cont}(a, l, s)$*, and*

$$\mathsf{hsame}(\mathsf{append}(p, r), \mathsf{append}(p, s)),$$

then $\mathsf{hsame}(k, l)$*.*

Proof. Left cancellation removes the common visible prefix p from the result comparison. The two continuation witnesses now share source a and have history-same results, so left-context continuation cancellation identifies the two tails. □

Lean4 ✓ `BEDC.FKernel.Cont.cont prefixed result common source tail deterministic`

Theorem 4.46 (Continuation fixed right-unit probe forces empty tail)**.** *For all histories* p, u : BHist*, if*

$$\forall r : \mathsf{BHist}.\ \mathsf{Cont}(p, u, r) \Rightarrow \mathsf{hsame}(r, p),$$

then $\mathsf{hsame}(u, \mathsf{e})$*.*

Proof. Let $r_0 = \mathsf{append}(p, u)$ be the canonical result representative for the displayed continuation tail. By the append presentation of Cont in Definition 4.10, there is a continuation witness

$$\mathsf{Cont}(p, u, r_0).$$

The fixed-probe hypothesis applied to this witness gives $\mathsf{hsame}(r_0, p)$. The base rule of Definition 4.10 also gives the right-unit continuation

$$\mathsf{Cont}(p, \mathsf{e}, p).$$

Apply Definition 4.43 to these two continuations, using reflexivity for $\mathsf{hsame}(p, p)$ and the obtained result comparison $\mathsf{hsame}(r_0, p)$. The conclusion is $\mathsf{hsame}(u, \mathsf{e})$. □

Lean4 ✓ `BEDC.FKernel.Cont.cont fixed right unit probe empty tail`

Theorem 4.47 (Continuation fixed left-unit probe forces empty source)**.** *For all histories* u, p : BHist*, if*

$$\forall r : \mathsf{BHist}.\ \mathsf{Cont}(u, p, r) \Rightarrow \mathsf{hsame}(r, p),$$

then $\mathsf{hsame}(u, \mathsf{e})$*.*

Proof. Let $r_0 = \mathsf{append}(u, p)$ be the canonical result representative for continuing source u by tail p. By the append presentation of Cont in Definition 4.10, there is a continuation witness

$$\mathsf{Cont}(u, p, r_0).$$

The fixed-probe hypothesis applied to this witness gives $\mathsf{hsame}(r_0, p)$. Transporting the canonical witness along this result comparison by Definition 4.40 gives

$$\mathsf{Cont}(u, p, p).$$

By the left-unit clause of Definition 4.38, there is also a continuation witness

$$\mathsf{Cont}(\mathsf{e}, p, p).$$

Apply the right-cancellation clause of Definition 4.52 to these two continuations and the reflexive comparison $\mathsf{hsame}(p, p)$. The conclusion is $\mathsf{hsame}(u, \mathsf{e})$. □

Lean4 ✓ `BEDC.FKernel.Cont.cont fixed left unit probe empty source`

Theorem 4.48 (Continuation prefix cancellation)**.** *If* $\mathsf{Cont}(\mathsf{append}(p, a), f, \mathsf{append}(p, b))$*, then* $\mathsf{Cont}(a, f, b)$*.*

Proof. Read the displayed continuation as equality of append representatives. Append associativity rewrites the right side as $\mathsf{append}(p, \mathsf{append}(a, f))$. Left cancellation of the common prefix p identifies b with $\mathsf{append}(a, f)$, which is exactly the unprefixed continuation. □

Lean4 ✓ `BEDC.FKernel.Cont.cont prefix cancel`

Theorem 4.49 (Continuation prefix equivalence)**.** *For histories* p, a, b, f*,*

$$\mathsf{Cont}(\mathsf{append}(p, a), f, \mathsf{append}(p, b)) \iff \mathsf{Cont}(a, f, b).$$

Proof. The forward direction is continuation prefix cancellation. For the reverse direction, prefix the equality witness for $\mathsf{Cont}(a, f, b)$ by $\mathsf{append}(p, -)$ and use associativity to identify $\mathsf{append}(p, \mathsf{append}(a, f))$ with $\mathsf{append}(\mathsf{append}(p, a), f)$. □

Lean4 ✓ `BEDC.FKernel.Cont.cont prefix iff`

Theorem 4.50 (Append right-context cancellation up to history-sameness)**.** *For histories* $P, Q, a, b : \mathsf{BHist}$*, if* $P \mathsf{hsame} Q$*, then appending the history-same right contexts* P *and* Q *preserves and reflects endpoint history-sameness:*

$$P \mathsf{hsame} Q \Rightarrow (\mathsf{hsame}(\mathsf{append}(a, P), \mathsf{append}(b, Q)) \iff a \mathsf{hsame} b).$$

Proof. Transport the right context along $P \mathsf{hsame} Q$. The forward direction is right append cancellation for the common suffix. Conversely, after the endpoints a and b are identified, the two appended histories are definitionally the same. □

Lean4 ✓ `BEDC.FKernel.Cont.append hsame right context cancel iff`

Theorem 4.51 (Append common-context cancellation up to history-sameness)**.** *For histories* $L, R, P, Q, a, b : \mathsf{BHist}$*, if* $L \mathsf{hsame} R$ *and* $P \mathsf{hsame} Q$*, then the two-sided contextual append comparison is exactly the comparison of the middle histories:*

$$L \mathsf{hsame} R \Rightarrow P \mathsf{hsame} Q \Rightarrow (\mathsf{hsame}(\mathsf{append}(L, \mathsf{append}(a, P)), \mathsf{append}(R, \mathsf{append}(b, Q))) \iff a \mathsf{hsame} b).$$

Proof. First transport the left and right contexts along the two hsame witnesses. In the forward direction, left append cancellation removes the common outer context and right append cancellation removes the common suffix. The reverse direction is immediate after the middle histories are identified. □

Lean4 ✓ `BEDC.FKernel.Cont.append hsame common context cancel iff`

Theorem 4.52 (Continuation cancellation laws)**.** *Continuation admits cancellation on both sides:*

1. *if* $\mathsf{Cont}(h, k, r)$*,* $\mathsf{Cont}(h', k, r')$*, and* $\mathsf{hsame}(r, r')$*, then* $\mathsf{hsame}(h, h')$*;*
2. *if* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{Cont}(h, k', r)$*, then* $\mathsf{hsame}(k, k')$*;*
3. *if* $\mathsf{Cont}(a, b, ab)$*,* $\mathsf{Cont}(ab, c, left)$*,* $\mathsf{Cont}(a, d, ad)$*,* $\mathsf{Cont}(ad, c, right)$*, and* $\mathsf{hsame}(left, right)$*, then* $\mathsf{hsame}(b, d)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont right cancel`

↪ variant `BEDC.FKernel.Cont.cont right cancel eq`

↪ variant `BEDC.FKernel.Cont.cont common suffix cancellation`

↪ variant `BEDC.FKernel.Cont.cont right cancel hsame result`

↪ variant `BEDC.FKernel.Cont.cont cancel common context`

↪ variant `BEDC.FKernel.Cont.continuation right cancellation`

↪ variant `BEDC.FKernel.Cont.cont left cancel`

↪ variant `BEDC.FKernel.Cont.append right cancel`

↪ variant `BEDC.FKernel.Cont.append left cancel`

Proof. For the first clause, use right-suffix cancellation after transporting the result by hsame. The second clause is left-context cancellation: compare the two derivations with the same left history, peel the shared left context, and conclude that the right histories are equal up to hsame. For the third clause, associate the two continuations so both sides expose the same outer context c over the inner continuations from a; the common outer context cancels against the result-sameness, and the remaining left-context cancellation yields $\mathsf{hsame}(b, d)$. □

Theorem 4.53 (Continuation cancellation with history-same right context)**.** *For histories* $a, a', b, b', r, r' : \mathsf{BHist}$*, if* $\mathsf{Cont}(a, b, r)$*,* $\mathsf{Cont}(a', b', r')$*,* $\mathsf{hsame}(b, b')$*, and* $\mathsf{hsame}(r, r')$*, then* $\mathsf{hsame}(a, a')$*.*

Proof. Transport the second right context and result along the two hsame witnesses. The continuations now share right context and result, so right cancellation identifies the two sources. □

Lean4 ✓ `BEDC.FKernel.Cont.cont cancel hsame right context`

Theorem 4.54 (Self-extension by a visible tail is impossible)**.** *For any histories* h *and* t*, neither* $\mathsf{Cont}(h, \mathsf{E0}(t), h)$ *nor* $\mathsf{Cont}(h, \mathsf{E1}(t), h)$ *can hold.*

Lean4 ✓ `BEDC.FKernel.Cont.cont self extension tail absurd`

Proof. Any such continuation would make a visible right tail behave as the empty right unit at h. Right-unit uniqueness identifies the tail with e, contradicting constructor no-confusion for $\mathsf{E0}(t)$ and $\mathsf{E1}(t)$. □

Theorem 4.55 (Mutual continuations cannot start with a visible left tail)**.** *For histories* h, k, t, u*, a continuation* $\mathsf{Cont}(h, \mathsf{E0}(t), k)$ *together with a return continuation* $\mathsf{Cont}(k, u, h)$ *is impossible; the same holds with* $\mathsf{E1}(t)$ *in place of* $\mathsf{E0}(t)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont mutual extension left tail absurd`

Proof. Mutual continuations force both displayed tails to be empty by the concrete tail-empty cancellation theorem. A left tail with outer constructor $\mathsf{E0}$ or $\mathsf{E1}$ cannot be internally same as e. □

Theorem 4.56 (Continuation associativity)**.** *If* $\mathsf{Cont}(h, k, u)$*,* $\mathsf{Cont}(u, l, v)$*,* $\mathsf{Cont}(k, l, w)$*, and* $\mathsf{Cont}(h, w, z)$*, then* $\mathsf{hsame}(v, z)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont assoc relational`

↪ variant `BEDC.FKernel.Cont.cont assoc middle exists`

↪ variant `BEDC.FKernel.Cont.cont assoc forward witness`

↪ variant `BEDC.FKernel.Cont.cont assoc left exists`

↪ variant `BEDC.FKernel.Cont.cont assoc exists hsame`

```
↪variant BEDC.FKernel.Cont.cont assoc hsame
↪variant BEDC.FKernel.Cont.cont assoc unique
↪variant BEDC.FKernel.Cont.cont assoc four
↪variant BEDC.FKernel.Cont.cont assoc five
↪variant BEDC.FKernel.Cont.cont assoc six
↪variant BEDC.FKernel.Cont.cont assoc primary
↪variant BEDC.FKernel.Cont.continuation associativity
↪variant BEDC.FKernel.Cont.continuation associativity up to hsame
↪variant BEDC.FKernel.Cont.continuation associativity spine
↪variant BEDC.FKernel.Cont.cont assoc common witness
↪variant BEDC.FKernel.Cont.cont assoc common witness symmetric
↪variant BEDC.FKernel.Cont.cont assoc up to hsame spine
↪variant BEDC.FKernel.Cont.cont assoc spine symmetric witnesses
```

Proof. Induct on the derivation of $\mathsf{Cont}(h, k, u)$. The empty case reduces both sides to the determinacy of $\mathsf{Cont}(k, l, -)$. In the $\mathsf{E0}$ and $\mathsf{E1}$ cases, no-confusion forces matching outer constructors in the relevant continuation derivations; use the induction hypothesis on inner histories and rebuild hsame under the same outer constructor. □

Theorem 4.57 (Continuation right append decomposition)**.** *If* $\mathsf{Cont}(h, \mathsf{append}(k, l), r)$*, where* append *is the recursive append representative underlying* Cont*, then there exists a generated history* m *such that* $\mathsf{Cont}(h, k, m)$ *and* $\mathsf{Cont}(m, l, r)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont right append decompose`

Proof. Unfold continuation as append equality. Take m to be the append of h and k. The first continuation is definitional, and the second is the reverse direction of append associativity. □

Theorem 4.58 (Continuation left append decomposition)**.** *If* $\mathsf{Cont}(\mathsf{append}(h, k), l, r)$*, where* append *is the recursive append representative underlying* Cont*, then there exists a generated history* m *such that* $\mathsf{Cont}(k, l, m)$ *and* $\mathsf{Cont}(h, m, r)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont left append decompose`

Proof. Unfold continuation as append equality. Take m to be the append of k and l. The first continuation is definitional, and the second is append associativity. □

Corollary 4.59 (Continuation gives an addition-like seed)**.** *Unary repetition histories with relational continuation license an addition-like derived interface* $\mathsf{Add}^{\uparrow}$*, but no primitive addition is introduced into the kernel.*

Lean4 ✓ `BEDC.FKernel.Cont.cont addition like seed`

Chapter 5

Asking Events and Policies

5.1 Asking events

Definition 5.1 (Probe names and evidence tokens)**.** ProbeName *is a syntactic class of probe names.* Ev *is a syntactic class of evidence tokens. Neither is assumed to be a set of functions or a set of proofs.*

Lean4 (definition) `BEDC.FKernel.Ask.ProbeName`
Lean4 (definition) `BEDC.FKernel.Ask.Evidence`

Definition 5.2 (Asking event)**.** *An asking event is a judgment*

$$\mathsf{Ask}(\pi, h, m, \delta),$$

where $\pi : \mathsf{ProbeName}$*,* $h : \mathsf{Hist}$*,* $m : \mathsf{Mark}$*, and* $\delta : \mathsf{Ev}$*. It is read: asking probe name* π *of history* h *emits mark* m *with evidence token* δ*.*

Lean4 (definition) `BEDC.FKernel.Ask.Ask`
Lean4 (definition) `BEDC.FKernel.Ask.AskEvent`
Lean4 ✓ `BEDC.FKernel.Ask.askEvent witness`
↪ variant `BEDC.FKernel.Ask.askEvent components`
↪ variant `BEDC.FKernel.Ask.AskEvent iff exists`
↪ variant `BEDC.FKernel.Ask.askEvent from witness`
↪ variant `BEDC.FKernel.Ask.askEvent evidence witness`

5.2 Asking policies

Definition 5.3 (Asking policy)**.** *An asking policy* $\mathsf{AskPolicy}(\Pi, D)$ *is paper notation for using a checked* $\mathsf{AskPolicy}(D)$ *certificate at bundle* Π*. The bundle is not a field of the certificate; it determines which probe-name occurrences the surrounding signature construction consumes. The certificate provides:*

1. *totality: every admitted history in* D *has generated asking events for each probe name, with the Lean field quantified over all probe names;*
2. *mark determinacy: if* $\mathsf{Ask}(\pi, h, m, \delta)$ *and* $\mathsf{Ask}(\pi, h, n, \theta)$*, then* $\mathsf{msame}(m, n)$*;*
3. *history-respect: if* $\mathsf{hsame}(h, k)$*, then asking results agree up to* msame *whenever both are generated.*

The object-language content is exactly totality, mark determinacy, and history-respect.

Lean4 ✓ `BEDC.FKernel.Ask.AskPolicy iff fields`

↪ variant `BEDC.FKernel.Ask.ask policy fields`
↪ variant `BEDC.FKernel.Ask.askPolicy total event`
↪ variant `BEDC.FKernel.Ask.askPolicy total deterministic pair`
↪ variant `BEDC.FKernel.Ask.askPolicy total respects pair`
↪ variant `BEDC.FKernel.Ask.askPolicy deterministic respects pair`
↪ variant `BEDC.FKernel.Ask.AskPolicy total deterministic pair`
↪ variant `BEDC.FKernel.Ask.ask total from policy`
↪ variant `BEDC.FKernel.Ask.ask total`
↪ variant `BEDC.FKernel.Ask.ask respects history`
↪ variant `BEDC.FKernel.Ask.askPolicy respects history field`

Definition 5.4 (Bundle-local asking policy)**.** *A bundle-local asking policy* $\mathsf{BundleAskPolicy}(\Pi, D)$ *is the finite obligation needed by the signature construction for bundle* Π*. It consists of the following clauses, each restricted to probe-name occurrences* $\pi \in \Pi$*:*

1. *totality on admitted sources: if* $h \in D$*, then there are* $m : \mathsf{Mark}$ *and* $\delta : \mathsf{Ev}$ *with* $\mathsf{Ask}(\pi, h, m, \delta)$*;*
2. *mark determinacy on the bundle: if* $\mathsf{Ask}(\pi, h, m, \delta)$ *and* $\mathsf{Ask}(\pi, h, n, \theta)$*, then* $\mathsf{msame}(m, n)$*;*
3. *history-respect on the bundle: if* $\mathsf{hsame}(h, k)$*,* $\mathsf{Ask}(\pi, h, m, \delta)$*, and* $\mathsf{Ask}(\pi, k, n, \theta)$*, then* $\mathsf{msame}(m, n)$*.*

Thus $\mathsf{AskPolicy}(\Pi, D)$ *is a sufficient global certificate, while* $\mathsf{BundleAskPolicy}(\Pi, D)$ *records exactly the finite probe obligations consumed by signatures through* Π*.*

Lean4 ✓ `BEDC.FKernel.Ask.BundleAskPolicy iff fields`

Theorem 5.5 (Empty bundle-local asking policy)**.** *For every source specification* D*,* $\mathsf{BundleAskPolicy}(\mathsf{Bnil}, D)$ *holds.*

Proof. Each totality, determinacy, and history-respect clause begins from an occurrence proof $\pi \in \mathsf{Bnil}$. Empty-bundle membership inversion eliminates that occurrence, so all three fields are satisfied. □

Lean4 ✓ `BEDC.FKernel.Ask.empty bundleAskPolicy`
Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy empty`

Theorem 5.6 (Bundle-local policy restriction by membership inclusion)**.** *For probe bundles* Π, Θ *and a domain policy* D*, suppose* $\mathsf{BundleAskPolicy}(\Pi, D)$ *holds and every probe-name occurrence of* Θ *is an occurrence of* Π*:*

$$\forall \rho,\ \rho \in \Theta \Rightarrow \rho \in \Pi.$$

Then $\mathsf{BundleAskPolicy}(\Theta, D)$ *holds.*

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy restrict by inclusion`

Proof. Let ρ be a probe-name occurrence of Θ. The displayed membership inclusion gives $\rho \in \Pi$. Apply the totality, mark determinacy, and history-respect fields of $\mathsf{BundleAskPolicy}(\Pi, D)$ at that occurrence. These three precomposed fields are exactly the bundle-local asking policy fields for Θ. □

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy of membership inclusion`

Lemma 5.7 (Bundle-local asking policy tail restriction)**.** *If* $\mathsf{BundleAskPolicy}(\mathsf{Bcons}(\pi, \Pi), D)$ *holds, then* $\mathsf{BundleAskPolicy}(\Pi, D)$ *holds.*

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy tail`

Proof. Each tail occurrence $\rho \in \Pi$ is also an occurrence $\rho \in \mathsf{Bcons}(\pi, \Pi)$. Restrict totality, mark determinacy, and history-respect along this membership inclusion, field by field. □

Theorem 5.8 (Bundle-local asking policy cons decomposition)**.** *For a head probe* π*, a tail bundle* Π*, and a source specification* D*,* $\mathsf{BundleAskPolicy}(\mathsf{Bcons}(\pi, \Pi), D)$ *holds exactly when the head probe satisfies totality, mark determinacy, and history-respect over* D*, and* $\mathsf{BundleAskPolicy}(\Pi, D)$ *holds.*

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy cons iff`

Proof. In the forward direction, use the head occurrence $\pi \in \mathsf{Bcons}(\pi, \Pi)$ to project the three head fields, and restrict the policy to the tail for the remaining field. In the reverse direction, case-analyze an occurrence of $\mathsf{Bcons}(\pi, \Pi)$: the head case uses the three head clauses, while the tail case uses the tail bundle-local policy. □

Theorem 5.9 (Global asking policy restricts to bundle-local policy)**.** *If* $\mathsf{AskPolicy}(\Pi, D)$ *holds, then* $\mathsf{BundleAskPolicy}(\Pi, D)$ *holds.*

Lean4 ✓ `BEDC.FKernel.Ask.askPolicy to bundleAskPolicy`

Proof. Use the totality, mark determinacy, and history-respect clauses supplied by $\mathsf{AskPolicy}(\Pi, D)$. For each probe-name occurrence $\pi \in \Pi$, the globally quantified totality clause gives an asking event for every admitted source history in D; the same global determinacy and history-respect clauses apply to the restricted occurrence π. These three projections are exactly the clauses of $\mathsf{BundleAskPolicy}(\Pi, D)$. □

Lemma 5.10 (Asking determinacy)**.** *Under an asking policy, a fixed probe name applied to a fixed history emits a mark unique up to* msame*.*

Lean4 ✓ `BEDC.FKernel.Ask.ask deterministic`
↪ variant `BEDC.FKernel.Ask.asking determinacy`
↪ variant `BEDC.FKernel.Ask.asking determinacy field`

Proof. This is the mark determinacy field of AskPolicy. □

Chapter 6

Generated Probe Bundles and Signature Events

6.1 Bundle grammar

Definition 6.1 (Probe bundle)**.** *Probe bundles are generated by:*

$$\frac{}{\mathsf{Bnil} : \mathsf{Bundle}}$$

$$\frac{\pi : \mathsf{ProbeName} \qquad \Pi : \mathsf{Bundle}}{\mathsf{Bcons}(\pi, \Pi) : \mathsf{Bundle}}.$$

Bundle membership is the closed predicate generated by cons membership: the head probe of $\mathsf{Bcons}(\pi, \Pi)$ *is a member of that bundle, and tail membership propagates through cons.*

Lean4 (definition) `BEDC.FKernel.Bundle.ProbeBundle`

Lean4 ✓ `BEDC.FKernel.Bundle.bundle generation cases`

↪ variant `BEDC.FKernel.Bundle.probeBundle nil ne cons`

↪ variant `BEDC.FKernel.Bundle.probeBundle no confusion all`

↪ variant `BEDC.FKernel.Bundle.probeBundle cons injective pair`

↪ variant `BEDC.FKernel.Bundle.probeBundle cons injective`

↪ variant `BEDC.FKernel.Bundle.probeBundle cons ne self`

↪ variant `BEDC.FKernel.Bundle.probeBundle two cons ne tail`

↪ variant `BEDC.FKernel.Bundle.probeBundle three cons ne tail`

↪ variant `BEDC.FKernel.Bundle.probeBundle generated induction`

↪ variant `BEDC.FKernel.Bundle.bundle generation cases and induction`

↪ variant `BEDC.FKernel.Bundle.bundle generation tail induction pair`

Lean4 (definition) `BEDC.FKernel.Bundle.InBundle`

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle cons self`

↪ variant `BEDC.FKernel.Bundle.bundle generation head tail pair`

↪ variant `BEDC.FKernel.Bundle.inBundle cons of eq`

↪ variant `BEDC.FKernel.Bundle.inBundle singleton iff`

↪ variant `BEDC.FKernel.Bundle.inBundle nil elim`

↪ variant `BEDC.FKernel.Bundle.inBundle nil false`

↪ variant `BEDC.FKernel.Bundle.inBundle cons tail`

↪ variant `BEDC.FKernel.Bundle.inBundle cons cons tail`

↪ variant BEDC.FKernel.Bundle.inBundle cons cons tail iff of head ne
↪ variant BEDC.FKernel.Bundle.inBundle cons inversion
↪ variant BEDC.FKernel.Bundle.inBundle tail of cons ne
↪ variant BEDC.FKernel.Bundle.inBundle cons tail iff of ne
↪ variant BEDC.FKernel.Bundle.inBundle cons iff
↪ variant BEDC.FKernel.Bundle.inBundle cons cons iff
↪ variant BEDC.FKernel.Bundle.inBundle cons cons inversion
↪ variant BEDC.FKernel.Bundle.inBundle cons cons swap
↪ variant BEDC.FKernel.Bundle.inBundle cons cons comm
↪ variant BEDC.FKernel.Bundle.inBundle cons cons swap iff
↪ variant BEDC.FKernel.Bundle.inBundle cons three rotate
↪ variant BEDC.FKernel.Bundle.inBundle cons three rotate iff
Lean4 ✓ BEDC.FKernel.Bundle.inBundle cons three head or tail
↪ variant BEDC.FKernel.Bundle.inBundle cons four head or tail
↪ variant BEDC.FKernel.Bundle.inBundle cons four rotate iff
↪ variant BEDC.FKernel.Bundle.inBundle nonempty from membership

Definition 6.2 (Bundle append)**.** *The append operation* $\mathsf{BAppend}(\Pi, \Theta)$ *is recursion on the left bundle:*

$$\mathsf{BAppend}(\mathsf{Bnil}, \Theta) = \Theta, \qquad \mathsf{BAppend}(\mathsf{Bcons}(\pi, \Pi), \Theta) = \mathsf{Bcons}(\pi, \mathsf{BAppend}(\Pi, \Theta)).$$

Lean4 (definition) BEDC.FKernel.Bundle.bundleAppend

Definition 6.3 (Probe bundle length)**.** *The length* $\mathsf{BLEN}(\Pi)$ *of a generated probe bundle is recursion on the bundle:*

$$\mathsf{BLEN}(\mathsf{Bnil}) = 0, \qquad \mathsf{BLEN}(\mathsf{Bcons}(\pi, \Pi)) = 1 + \mathsf{BLEN}(\Pi).$$

Lean4 (definition) BEDC.FKernel.Bundle.bundleLength

Theorem 6.4 (Probe bundle zero length equivalence)**.** *A generated probe bundle has length zero exactly when it is the empty bundle* Bnil*.*

Proof. Eliminate the generated bundle. The empty case reduces by computation. In the cons case the length reduces to a successor, which cannot equal zero. □

Lean4 ✓ BEDC.FKernel.Bundle.bundleLength eq zero iff nil

Lemma 6.5 (Probe bundle zero length exactly empty)**.** *For every generated probe bundle* Π*,*

$$\mathsf{BLEN}(\Pi) = 0 \quad \Longleftrightarrow \quad \Pi = \mathsf{Bnil}.$$

Proof. Case split on the generated bundle. The empty case is definitional. In the cons case, the length has successor form, contradicting equality with zero. The reverse implication follows by substituting Bnil. □

Lean4 ✓ BEDC.FKernel.Bundle.bundleLength zero iff nil

Theorem 6.6 (Probe bundle length one singleton equivalence)**.** *A generated probe bundle has length one exactly when it is a singleton* $\mathsf{Bcons}(\pi, \mathsf{Bnil})$ *for some probe* π*.*

Proof. Eliminate the generated bundle. The empty case has length zero, so it cannot have length one. In the cons case, injectivity of successor forces the tail length to be zero; Theorem 6.4 identifies the tail with Bnil. Conversely, substituting a singleton bundle reduces the length equation by computation. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleLength eq one iff singleton`

Theorem 6.7 (Probe bundle length-one member singleton)**.** *If a generated probe bundle has length one and contains probe π, then the bundle is exactly the singleton* $\mathsf{Bcons}(\pi, \mathsf{Bnil})$*.*

Proof. Case split on the generated bundle. The empty case contradicts membership. In the cons case, successor injectivity forces the tail length to be zero, so Theorem 6.4 identifies the tail with Bnil. Membership is then either the head equality, which fixes the singleton head, or tail membership, which is impossible for Bnil. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle length one singleton`

Theorem 6.8 (Probe bundle length-one membership uniqueness)**.** *If a generated probe bundle has length one and contains probes π and ρ, then $\pi = \rho$.*

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle length one unique`

Proof. Apply the length-one member singleton theorem to each membership witness. Both identify the same bundle as a singleton, so constructor injectivity identifies the two singleton heads. □

Theorem 6.9 (Probe bundle successor length cons decomposition)**.** *A generated probe bundle has successor length exactly when it is a cons whose tail has the predecessor length.*

Proof. Eliminate the bundle. The empty case contradicts successor length, while the cons case uses successor injectivity; the converse substitutes the cons equation. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleLength eq succ iff cons`

Theorem 6.10 (Probe bundle append length)**.** *For all probe bundles Π, Θ,*

$$\mathsf{BLEN}(\mathsf{BAppend}(\Pi, \Theta)) = \mathsf{BLEN}(\Pi) + \mathsf{BLEN}(\Theta).$$

Proof. Induct on Π. In the empty case, the append term reduces to Θ and the length equation reduces to the left identity of addition. In the cons case, both sides reduce by one successor, and the tail equation is the induction hypothesis. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleLength append`
Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend length`
Lean4 ✓ `BEDC.FKernel.Bundle.bundleLength bundleAppend`

Lemma 6.11 (Probe bundle nonempty prefix length separation)**.** *For every probe π and probe bundles Π_0, Ω,*

$$\mathsf{BLEN}(\mathsf{BAppend}(\mathsf{Bcons}(\pi, \Pi_0), \Omega)) \neq \mathsf{BLEN}(\Omega).$$

Consequently,

$$\mathsf{BAppend}(\mathsf{Bcons}(\pi, \Pi_0), \Omega) \neq \Omega.$$

Proof. Theorem 6.10 rewrites the left length as $1 + \mathsf{BLEN}(\Pi_0) + \mathsf{BLEN}(\Omega)$. A natural number with an exposed successor constructor cannot be equal to the unextended suffix length by natural-number constructor no-confusion. If the appended bundle itself were equal to Ω, applying BLEN to that equality would contradict the displayed length separation. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend nonempty prefix length separation`

Theorem 6.12 (Probe bundle append fixes the right side exactly at nil left)**.** *For probe bundles* Π *and* Ω*,*

$$\mathsf{BAppend}(\Pi, \Omega) = \Omega$$

if and only if $\Pi = \mathsf{Bnil}$*.*

Proof. Split on Π. The nil case is definitional. In the cons case, the nonempty-prefix length separation theorem says that appending a cons-prefix to Ω cannot equal Ω. The reverse implication follows by reducing the append definition at Bnil. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend eq right iff left nil`

Theorem 6.13 (Probe bundle append fixes the left side exactly at nil right)**.** *For probe bundles* Π *and* Ω*,*

$$\mathsf{BAppend}(\Pi, \Omega) = \Pi$$

if and only if $\Omega = \mathsf{Bnil}$*.*

Proof. Induct on Π. In the empty case, the displayed equation is exactly $\Omega = \mathsf{Bnil}$. In the cons case, constructor injectivity reduces the fixed append equation to the tail. The reverse implication is the right-unit theorem for append. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend eq left iff right nil`

Theorem 6.14 (Probe bundle append right unit)**.** *For every probe bundle* Π*,*

$$\mathsf{BAppend}(\Pi, \mathsf{Bnil}) = \Pi.$$

Proof. Induct on Π. The empty bundle case is definitional. In the cons case, both sides expose the same head probe, and the tail equality is the induction hypothesis. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend nil right`

Theorem 6.15 (Probe bundle append associativity)**.** *For all probe bundles* Π, Θ, Ω*,*

$$\mathsf{BAppend}(\mathsf{BAppend}(\Pi, \Theta), \Omega) = \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)).$$

Proof. Induct on Π. The empty bundle case is definitional. In the cons case, both sides expose the same head probe, and the tail equality is the induction hypothesis. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend assoc`

Theorem 6.16 (Bundle append membership)**.** *For every probe* p *and bundles* Π, Θ*,*

$$p \in \mathsf{BAppend}(\Pi, \Theta) \iff p \in \Pi \text{ or } p \in \Theta.$$

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle bundleAppend iff`

Proof. Induct on Π. The empty case reduces membership to Θ. The cons case splits head membership from tail membership, then applies the induction hypothesis to the tail. □

Theorem 6.17 (Bundle member split)**.** *If a generated probe bundle Π contains probe p, then Π splits as an append prefix followed by a cons whose head is p:*

$$\Pi = \mathsf{BAppend}(\Pi_L, \mathsf{Bcons}(p, \Pi_R))$$

for some generated bundles Π_L and Π_R.

Proof. Induct on the generated bundle. The empty bundle has no membership witness. In the cons case, head membership gives the split with empty left prefix, while tail membership applies the induction hypothesis to the tail and then restores the displayed head as one more constructor in the left prefix. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle member split`

Theorem 6.18 (Bundle member split equivalence)**.** *For a generated probe bundle Π and probe p,*

$$p \in \Pi \quad \Longleftrightarrow \quad \exists \Pi_L, \exists \Pi_R,\ \Pi = \mathsf{BAppend}(\Pi_L, \mathsf{Bcons}(p, \Pi_R)).$$

Proof. The forward direction is Theorem 6.17. Conversely, after substituting the displayed append decomposition, append membership places p in the right summand, where the exposed cons head is exactly p. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle member split iff`

Theorem 6.19 (Bundle append three-way membership split)**.** *For every probe p and bundles Π, Θ, Ω,*

$$p \in \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)) \quad \Longleftrightarrow \quad p \in \Pi \text{ or } p \in \Theta \text{ or } p \in \Omega.$$

Proof. Apply Theorem 6.16 to the outer append, then apply it again to the inner append. The reverse direction rebuilds the same two append membership witnesses from the displayed three-way disjunction. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle bundleAppend three iff`

Theorem 6.20 (Bundle append four-way membership split)**.** *For every probe p and bundles $\Pi, \Theta, \Omega, \Lambda$,*

$$p \in \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \mathsf{BAppend}(\Omega, \Lambda))) \quad \Longleftrightarrow \quad p \in \Pi \text{ or } p \in \Theta \text{ or } p \in \Omega \text{ or } p \in \Lambda.$$

Proof. Apply Theorem 6.16 to the outer append. In the right branch, apply the same membership split to the second append, and then to the third append. The reverse direction rebuilds the nested append membership witnesses in the same order. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle bundleAppend four iff`

Theorem 6.21 (Bundle append three-segment membership rotation)**.** *For every probe p and bundles Π, Θ, Ω,*

$$p \in \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)) \quad \Longleftrightarrow \quad p \in \mathsf{BAppend}(\Theta, \mathsf{BAppend}(\Omega, \Pi)).$$

Proof. For the forward direction, apply Definition 6.19 to the left-hand membership. The resulting cases are $p \in \Pi$, $p \in \Theta$, and $p \in \Omega$.

Apply the same split theorem to the right-hand bundle expression. In the $p \in \Pi$ case use the third disjunct, in the $p \in \Theta$ case use the first disjunct, and in the $p \in \Omega$ case use the second disjunct.

For the reverse direction, apply Definition 6.19 to the right-hand membership. The cases are now $p \in \Theta$, $p \in \Omega$, and $p \in \Pi$.

Apply the split theorem again to the left-hand bundle expression. These three cases return respectively the second, third, and first disjuncts of $p \in \Pi$ or $p \in \Theta$ or $p \in \Omega$. □

Theorem 6.22 (Bundle append membership left residual)**.** *For every probe p and bundles Π, Θ, if*

$$p \in \mathsf{BAppend}(\Pi, \Theta)$$

and $p \notin \Theta$, then $p \in \Pi$.

Proof. Induct on Π. In the empty case, appended membership is membership in Θ, contradicting the exclusion premise. In the cons case, either the probe is the head of Π, or it lies in the appended tail. The head branch is immediate, and the tail branch follows from the induction hypothesis with the same exclusion premise. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle bundleAppend left of not right`

Theorem 6.23 (Bundle append membership right residual)**.** *For every probe p and bundles Π, Θ, if*

$$p \in \mathsf{BAppend}(\Pi, \Theta)$$

and $p \notin \Pi$, then $p \in \Theta$.

Proof. Induct on Π. In the empty case, appended membership is exactly membership in Θ. In the cons case, head membership contradicts the exclusion premise for Π, while tail membership reduces to the induction hypothesis after restricting the exclusion premise to the tail. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle bundleAppend right of not left`

Theorem 6.24 (Bundle-local policy append restriction)**.** *If* $\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Pi, \Theta), D)$ *holds, then both* $\mathsf{BundleAskPolicy}(\Pi, D)$ *and* $\mathsf{BundleAskPolicy}(\Theta, D)$ *hold.*

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy append restriction`

Proof. Use Theorem 6.16. A member of Π maps into $\mathsf{BAppend}(\Pi, \Theta)$ through the left disjunct, and a member of Θ maps into $\mathsf{BAppend}(\Pi, \Theta)$ through the right disjunct. Restrict the totality, mark determinacy, and history-respect fields of the appended bundle-local policy along these two membership maps. □

Theorem 6.25 (Bundle-local policy append gluing)**.** *If* $\mathsf{BundleAskPolicy}(\Pi, D)$ *and* $\mathsf{BundleAskPolicy}(\Theta, D)$ *hold, then* $\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Pi, \Theta), D)$ *holds.*

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy append gluing`

Proof. Use Theorem 6.16. For each probe membership in $\mathsf{BAppend}(\Pi, \Theta)$, split it into a left membership in Π or a right membership in Θ. In the left case, take totality, determinacy, and history-respect from the policy on Π; in the right case, take the same three fields from the policy on Θ. These case analyses assemble the bundle-local asking policy for the appended bundle. □

Theorem 6.26 (Bundle-local policy append equivalence)**.** *For all probe bundles* Π, Θ *and domain* D*,*

$$\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Pi, \Theta), D) \iff (\mathsf{BundleAskPolicy}(\Pi, D) \text{ and } \mathsf{BundleAskPolicy}(\Theta, D)).$$

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy append iff`

Proof. The forward direction is exactly Theorem 6.24. The reverse direction is exactly Theorem 6.25. □

Theorem 6.27 (Bundle-local policy append commutation)**.** *For all probe bundles* Π, Θ *and domain* D*, if*

$$\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Pi, \Theta), D)$$

holds, then

$$\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Theta, \Pi), D)$$

holds.

Lean4 ✓ `BEDC.FKernel.Ask.bundleAskPolicy append comm`

Proof. Restrict the appended policy to obtain the two component policies on Π and Θ. Reapply bundle-append gluing in the opposite order, using the same totality, mark determinacy, and history-respect fields on their respective probe memberships. □

Theorem 6.28 (Probe bundle prefix cancellation)**.** *For any fixed probe bundle* Π *and any probe bundles* Θ, Ω*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{BAppend}(\Pi, \Omega),$$

then $\Theta = \Omega$*.*

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend prefix cancel`

Proof. Induct on Π. The empty case is immediate because both displayed append terms reduce to their suffix bundles. In the cons case, constructor injectivity reduces the equality of cons bundles to equality of their tails, and the induction hypothesis cancels the remaining common prefix. □

Theorem 6.29 (Probe bundle suffix cancellation)**.** *For any probe bundles* Π, Θ, Ω*, if*

$$\mathsf{BAppend}(\Pi, \Omega) = \mathsf{BAppend}(\Theta, \Omega),$$

then $\Pi = \Theta$*.*

Proof. Induct on Π and then case-split Θ. The empty-empty case is immediate. In the empty-cons and cons-empty cases, Lemma 6.11 rules out equality between a bundle obtained by appending a nonempty prefix to Ω and the suffix Ω. In the cons-cons case, the displayed equality is an equality of cons bundles. Constructor injectivity identifies the two head probes and reduces the tail equality to

$$\mathsf{BAppend}(\Pi_0, \Omega) = \mathsf{BAppend}(\Theta_0, \Omega).$$

The induction hypothesis gives $\Pi_0 = \Theta_0$, and the head equality rebuilds $\Pi = \Theta$. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend suffix cancel`

Theorem 6.30 (Probe bundle common context cancellation)**.** *For probe bundles* $\Pi, \Theta, \Omega, \Theta'$*, if*

$$\mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)) = \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta', \Omega)),$$

then $\Theta = \Theta'$*.*

Proof. First cancel the common left bundle Π by prefix cancellation. The remaining equality has the same suffix Ω on both sides, so suffix cancellation identifies the middle bundles. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend common context cancel`

Theorem 6.31 (Probe bundle append middle cancellation)**.** *For probe bundles* Π*,* Θ*,* Ω*, and* Λ*, if*

$$\mathsf{BAppend}(\mathsf{BAppend}(\Pi, \Theta), \Lambda) = \mathsf{BAppend}(\mathsf{BAppend}(\Pi, \Omega), \Lambda),$$

then $\Theta = \Omega$*.*

Proof. First cancel the common suffix Λ using probe-bundle suffix cancellation. This leaves equality of $\mathsf{BAppend}(\Pi, \Theta)$ and $\mathsf{BAppend}(\Pi, \Omega)$. Then cancel the common prefix Π using probe-bundle prefix cancellation. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend middle cancel`

Lemma 6.32 (Probe bundle append empty-result inversion)**.** *For probe bundles* Π, Θ*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{Bnil},$$

then $\Pi = \mathsf{Bnil}$ *and* $\Theta = \mathsf{Bnil}$*.*

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend empty result inversion`

Proof. Proceed by cases on Π. In the empty case the displayed equality reduces to $\Theta = \mathsf{Bnil}$. In the cons case the append definition exposes a cons bundle equal to Bnil, contradicting probe-bundle constructor no-confusion. □

Lemma 6.33 (Probe bundle append cons-result inversion)**.** *For probe bundles* Π, Θ, Ω *and probe* π*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{Bcons}(\pi, \Omega),$$

then either $\Pi = \mathsf{Bnil}$ *and* $\Theta = \mathsf{Bcons}(\pi, \Omega)$*, or there exists a bundle* Π_0 *such that* $\Pi = \mathsf{Bcons}(\pi, \Pi_0)$ *and*

$$\mathsf{BAppend}(\Pi_0, \Theta) = \Omega.$$

Proof. Proceed by cases on Π. In the empty case the displayed equality reduces to $\Theta = \mathsf{Bcons}(\pi, \Omega)$, giving the left alternative. In the cons case the append definition exposes a cons bundle on the left; constructor injectivity identifies its head with π and its appended tail with Ω, giving the required tail bundle Π_0. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend cons result inversion`

Theorem 6.34 (Probe bundle append cons-result iff)**.** *For probe bundles* Π, Θ, Ω *and probe* π*,*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{Bcons}(\pi, \Omega) \iff (\Pi = \mathsf{Bnil} \wedge \Theta = \mathsf{Bcons}(\pi, \Omega)) \vee \exists \Pi_0,\ \Pi = \mathsf{Bcons}(\pi, \Pi_0) \wedge \mathsf{BAppend}(\Pi_0, \Theta$$

Proof. The forward implication is the cons-result inversion lemma. For the converse, split the displayed disjunction. In the empty-prefix case the append computation is exactly the supplied suffix equality. In the cons-prefix case, substitute the displayed prefix and tail equality, then compute the append recursion. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend cons result iff`

Theorem 6.35 (Probe bundle append split uniqueness at fixed left length)**.** *For probe bundles* $\Pi, \Theta, \Pi', \Theta'$*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{BAppend}(\Pi', \Theta')$$

and

$$\mathsf{BLEN}(\Pi) = \mathsf{BLEN}(\Pi'),$$

then $\Pi = \Pi'$ *and* $\Theta = \Theta'$*.*

Proof. Proceed by simultaneous cases on the two left bundles. The empty-empty case reduces the append equality to the suffix equality $\Theta = \Theta'$. The empty-cons and cons-empty cases contradict the displayed length equality by natural-number constructor no-confusion. In the cons-cons case, constructor injectivity on the append equality identifies the two head probes and gives equality of the appended tails; constructor injectivity on the length equality gives equal tail lengths. The induction hypothesis applied to the tails gives equality of the left tails and suffixes, and the head equality rebuilds $\Pi = \Pi'$. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend split unique fixed length`

Theorem 6.36 (Bundle member split uniqueness at fixed prefix length)**.** *For probe bundles* $\Pi, \Pi_L, \Pi_R, \Pi'_L, \Pi'_R$ *and probes* p, q*, if*

$$\Pi = \mathsf{BAppend}(\Pi_L, \mathsf{Bcons}(p, \Pi_R)) \quad \textit{and} \quad \Pi = \mathsf{BAppend}(\Pi'_L, \mathsf{Bcons}(q, \Pi'_R))$$

and $\mathsf{BLEN}(\Pi_L) = \mathsf{BLEN}(\Pi'_L)$*, then*

$$\Pi_L = \Pi'_L \quad \textit{and} \quad p = q \quad \textit{and} \quad \Pi_R = \Pi'_R.$$

Proof. Compose the two displayed decompositions into an equality between the two append forms. Fixed-left-length split uniqueness identifies the prefixes and the residual cons suffixes. Constructor injectivity for the residual conses then identifies the exposed probes and right tails. □

Lean4 ✓ `BEDC.FKernel.Bundle.inBundle member split unique fixed prefix length`

Theorem 6.37 (Probe bundle append split uniqueness at fixed suffix length)**.** *For probe bundles* $\Pi, \Theta, \Pi', \Theta'$*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{BAppend}(\Pi', \Theta')$$

and

$$\mathsf{BLEN}(\Theta) = \mathsf{BLEN}(\Theta'),$$

then $\Pi = \Pi'$ *and* $\Theta = \Theta'$*.*

Proof. Apply append length preservation to the displayed equality. The equality of suffix lengths cancels the right summand in the resulting natural-number length identity, giving $\mathsf{BLEN}(\Pi) = \mathsf{BLEN}(\Pi')$. Fixed-left-length split uniqueness then identifies the two prefixes and the two suffixes. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend split unique fixed suffix length`

Theorem 6.38 (Probe bundle append three-split uniqueness at fixed lengths)**.** *For probe bundles* $\Pi, \Theta, \Omega, \Pi', \Theta', \Omega'$*, if*

$$\mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)) = \mathsf{BAppend}(\Pi', \mathsf{BAppend}(\Theta', \Omega'))$$

and $\mathsf{BLEN}(\Pi) = \mathsf{BLEN}(\Pi')$ *and* $\mathsf{BLEN}(\Theta) = \mathsf{BLEN}(\Theta')$*, then* $\Pi = \Pi'$*,* $\Theta = \Theta'$*, and* $\Omega = \Omega'$*.*

Proof. Apply fixed-left-length split uniqueness to the outer append, using the equality of Π and Π' lengths. This identifies the two first segments and leaves equality of the two residual inner appends. Apply the same split uniqueness theorem to those residual appends with the equality of Θ and Θ' lengths, obtaining equality of the middle and final segments. □

Lean4 ✓ `BEDC.FKernel.Bundle.bundleAppend three split unique fixed lengths`

Theorem 6.39 (Bundle append membership commutativity)**.** *For every probe* p *and bundles* Π, Θ*,*

$$p \in \mathsf{BAppend}(\Pi, \Theta) \iff p \in \mathsf{BAppend}(\Theta, \Pi).$$

Proof. For the forward direction, assume $p \in \mathsf{BAppend}(\Pi, \Theta)$. By Definition 6.16, either $p \in \Pi$ or $p \in \Theta$. If $p \in \Pi$, then Definition 6.16 applied to $\mathsf{BAppend}(\Theta, \Pi)$ gives $p \in \mathsf{BAppend}(\Theta, \Pi)$ from the right disjunct. If $p \in \Theta$, it gives the same conclusion from the left disjunct.

For the reverse direction, assume $p \in \mathsf{BAppend}(\Theta, \Pi)$. Again by Definition 6.16, either $p \in \Theta$ or $p \in \Pi$. In the first case, Definition 6.16 applied to $\mathsf{BAppend}(\Pi, \Theta)$ gives membership from the right disjunct; in the second case, it gives membership from the left disjunct. □

Theorem 6.40 (Bundle append Levi residual decomposition)**.** *For probe bundles* $\Pi, \Theta, \Omega, \Lambda$*, if*

$$\mathsf{BAppend}(\Pi, \Theta) = \mathsf{BAppend}(\Omega, \Lambda),$$

then there exists a probe bundle R *such that either*

$$\Omega = \mathsf{BAppend}(\Pi, R) \quad \textit{and} \quad \Theta = \mathsf{BAppend}(R, \Lambda),$$

or

$$\Pi = \mathsf{BAppend}(\Omega, R) \quad \textit{and} \quad \Lambda = \mathsf{BAppend}(R, \Theta).$$

Proof. Induct on Π, keeping Θ, Ω, and Λ arbitrary. If $\Pi = \mathsf{Bnil}$, the displayed equality reduces to

$$\Theta = \mathsf{BAppend}(\Omega, \Lambda).$$

Take $R = \Omega$. The append definition gives $\Omega = \mathsf{BAppend}(\mathsf{Bnil}, R)$, and the reduced equality gives $\Theta = \mathsf{BAppend}(R, \Lambda)$, so the first alternative holds.

Now let $\Pi = \mathsf{Bcons}(\pi, \Pi_0)$, and split on Ω. If $\Omega = \mathsf{Bnil}$, the displayed equality reduces to

$$\mathsf{BAppend}(\mathsf{Bcons}(\pi, \Pi_0), \Theta) = \Lambda.$$

Take $R = \mathsf{Bcons}(\pi, \Pi_0)$. The append definition gives $\Pi = \mathsf{BAppend}(\mathsf{Bnil}, R)$, and the reduced equality gives $\Lambda = \mathsf{BAppend}(R, \Theta)$, so the second alternative holds.

It remains to consider $\Omega = \mathsf{Bcons}(\omega, \Omega_0)$. The displayed equality is then an equality of two cons bundles:

$$\mathsf{Bcons}(\pi, \mathsf{BAppend}(\Pi_0, \Theta)) = \mathsf{Bcons}(\omega, \mathsf{BAppend}(\Omega_0, \Lambda)).$$

Constructor injectivity gives $\pi = \omega$ and

$$\mathsf{BAppend}(\Pi_0, \Theta) = \mathsf{BAppend}(\Omega_0, \Lambda).$$

Apply the induction hypothesis to this tail equality. In the first case, there is a probe bundle R with

$$\Omega_0 = \mathsf{BAppend}(\Pi_0, R) \quad \text{and} \quad \Theta = \mathsf{BAppend}(R, \Lambda).$$

Using $\pi = \omega$ and the append recursion,

$$\Omega = \mathsf{Bcons}(\omega, \Omega_0) = \mathsf{Bcons}(\pi, \mathsf{BAppend}(\Pi_0, R)) = \mathsf{BAppend}(\mathsf{Bcons}(\pi, \Pi_0), R) = \mathsf{BAppend}(\Pi, R),$$

so the first alternative holds for the original equality.

In the second case, there is a probe bundle R with

$$\Pi_0 = \mathsf{BAppend}(\Omega_0, R) \quad \text{and} \quad \Lambda = \mathsf{BAppend}(R, \Theta).$$

Using $\pi = \omega$ and the append recursion,

$$\Pi = \mathsf{Bcons}(\pi, \Pi_0) = \mathsf{Bcons}(\omega, \mathsf{BAppend}(\Omega_0, R)) = \mathsf{BAppend}(\mathsf{Bcons}(\omega, \Omega_0), R) = \mathsf{BAppend}(\Omega, R),$$

so the second alternative holds for the original equality. □

6.2 Signature generation

Definition 6.41 (Signature event)**.** *The signature event* $\mathsf{Sig}(\Pi, h, s, \Delta)$ *means that bundle* Π *applied to history* h *generates answer-history* s *with evidence* Δ*. It is generated by the rules below. In the checked kernel, the logical predicate is the three-place* $\mathsf{SigRel}(\Pi, h, s)$ *on bundle, source history, and answer history. The displayed evidence ledger is a paper-side annotation attached to the generating derivation.*

Lean4 ✓ `BEDC.FKernel.Sig.sig empty constructor`
- ↪ variant `BEDC.FKernel.Sig.sig cons constructor`
- ↪ variant `BEDC.FKernel.Sig.sig cons inversion`
- ↪ variant `BEDC.FKernel.Sig.sig cons result inversion`
- ↪ variant `BEDC.FKernel.Sig.sig cons tail deterministic`
- ↪ variant `BEDC.FKernel.Sig.sigRel cons inversion`
- ↪ variant `BEDC.FKernel.Sig.sigRel generation cases`
- ↪ variant `BEDC.FKernel.Sig.sig empty inversion`

Kernel Rule 6.42 (Empty signature)**.**

$$\frac{}{\mathsf{Sig}(\mathsf{Bnil}, h, \mathsf{e}, \Delta_0)}$$

where Δ_0 *is the neutral signature evidence token.*

Lemma 6.43 (Empty signature inversion)**.** *If* $\mathsf{Sig}(\mathsf{Bnil}, h, r, \Delta_0)$*, then* $\mathsf{hsame}(r, \mathsf{e})$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig empty inversion`

Proof. Invert the signature derivation. For the empty bundle, only the empty-signature rule applies, so the result history is e. □

Kernel Rule 6.44 (Cons signature)**.** *If*

$$\mathsf{Ask}(\pi, h, m, \delta), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Ext}(m, s, r),$$

then

$$\mathsf{Sig}(\mathsf{Bcons}(\pi, \Pi), h, r, \delta \star \Delta).$$

Here $\delta \star \Delta$ records the head asking evidence together with the tail derivation ledger; it is not a field of the asking policy.

Theorem 6.45 (Cons signature result is nonempty)**.** *If a cons bundle* $\mathsf{Bcons}(\pi, \Pi)$ *generates a result history r, then r cannot be internally the same as* e.

Proof. Invert the cons signature derivation. The final step is an extension witness $\mathsf{Ext}(m, s, r)$. Each extension constructor produces either an $\mathsf{E0}$ or an $\mathsf{E1}$ result, and BEDC history no-confusion separates both constructors from e. □

Lean4 ✓ `BEDC.FKernel.Sig.sig cons result ne empty`

Theorem 6.46 (Signature result length equals bundle length)**.** *If* $\mathsf{Sig}(\Pi, h, r, \Delta)$*, then the external binary length of the answer history r is the probe-bundle length of* Π.

Lean4 ✓ `BEDC.FKernel.Sig.sigRel result length eq bundleLength`

Proof. Induct on the signature derivation. The empty rule gives zero length on both sides. In the cons rule, the final extension adds exactly one outer history constructor, and the bundle is one cons longer than its tail. □

Theorem 6.47 (Empty signature result forces empty bundle)**.** *If* $\mathsf{Sig}(\Pi, h, \mathsf{e}, \Delta)$*, then* $\Pi = \mathsf{Bnil}$.

Lean4 ✓ `BEDC.FKernel.Sig.sigRel empty result bundle nil`

Proof. The result-length theorem gives bundle length zero because e has zero external binary length. Probe-bundle length zero characterizes Bnil. □

Theorem 6.48 (Signature generation over bundle append)**.** *For probe bundles Π, Θ and source history h, if* $\mathsf{Sig}(\Pi, h, s, \Delta)$ *and* $\mathsf{Sig}(\Theta, h, t, \Gamma)$*, then there exist a history u and evidence Λ such that*

$$\mathsf{Cont}(t, s, u) \quad \textit{and} \quad \mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, u, \Lambda).$$

Lean4 ✓ `BEDC.FKernel.Sig.signature generation bundle append`

Proof. Induct on Π. In the empty bundle case, empty-signature inversion identifies s with e; take $u = t$, use the right-unit continuation constructor, and reuse the given signature for Θ. In the cons case, invert the left signature. The tail signature for Π and the signature for Θ feed the induction hypothesis, producing a tail result u_0 with $\mathsf{Cont}(t, s_0, u_0)$ and a signature for $\mathsf{BAppend}(\Pi, \Theta)$. Rebuild the cons signature using the original head asking event and the tail signature. The final extension branch supplies either the $\mathsf{E0}$ or $\mathsf{E1}$ continuation step, yielding $\mathsf{Cont}(t, s, u)$ for the rebuilt result. □

Theorem 6.49 (Signature generation append inversion)**.** *For probe bundles Π, Θ and source history h, if* $\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, u, \Lambda)$*, then there exist histories s, t and evidence tokens Δ, Γ such that* $\mathsf{Sig}(\Pi, h, s, \Delta)$*,* $\mathsf{Sig}(\Theta, h, t, \Gamma)$*, and* $\mathsf{Cont}(t, s, u)$.

Lean4 ✓ `BEDC.FKernel.Sig.signature generation bundle append inversion`

Proof. Induct on Π. In the empty bundle case, the appended bundle is Θ; generate the empty signature for Bnil, take $s = \mathsf{e}$ and $t = u$, reuse the given signature for Θ, and apply the continuation right-unit constructor. In the cons case, invert the appended cons signature. The inversion exposes the head asking event, the tail signature for $\mathsf{BAppend}(\Pi, \Theta)$, and the final extension witness. Apply the induction hypothesis to the tail append signature to obtain signatures for Π and Θ together with a continuation $\mathsf{Cont}(t, s_0, u_0)$. Rebuild the left cons signature from the exposed head asking event, the recovered tail signature for Π, and the final extension witness. The same extension constructor lifts the recovered tail continuation to $\mathsf{Cont}(t, s, u)$ for the whole appended signature. □

Theorem 6.50 (Signature bundle-append decomposition)**.** *For probe bundles Π, Θ and source history h,*

$$\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, u, \Lambda) \iff \exists s, t, \Delta, \Gamma.\ \mathsf{Sig}(\Pi, h, s, \Delta) \wedge \mathsf{Sig}(\Theta, h, t, \Gamma) \wedge \mathsf{Cont}(t, s, u).$$

Lean4 ✓ `BEDC.FKernel.Sig.sigRel bundleAppend decomposition iff`

Proof. The forward implication is append-signature inversion. Conversely, generate the canonical appended signature from the two component signatures. Its continuation has the same displayed endpoints as $\mathsf{Cont}(t, s, u)$, so continuation determinacy identifies the generated result with u. □

Theorem 6.51 (Signature generation append component determinacy)**.** *For probe bundles Π, Θ, domain D, and source history h, assume $\mathsf{BundleAskPolicy}(\Pi, D)$, $\mathsf{BundleAskPolicy}(\Theta, D)$, and $h \in D$. If*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, u, \Lambda),$$

then there exists a history v such that

$$\mathsf{Cont}(t, s, v) \quad \textit{and} \quad \mathsf{hsame}(v, u).$$

Lean4 ✓ `BEDC.FKernel.Sig.signature generation append component determinacy`

Proof. Theorem 6.25 gives $\mathsf{BundleAskPolicy}(\mathsf{BAppend}(\Pi, \Theta), D)$ from the two component policies. Theorem 6.48 applied to the two component signatures produces a history v with $\mathsf{Cont}(t, s, v)$ and a generated signature for the appended bundle. Theorem 6.63, applied to the glued policy, the premise $h \in D$, that constructed appended signature, and the given appended signature for u, gives $\mathsf{hsame}(v, u)$. □

Theorem 6.52 (Signature generation bundle append recomposition determinacy)**.** *For probe bundles Π, Θ, domain D, and source history h, assume an asking policy on D and $h \in D$. If*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Cont}(t, s, u),$$

and $\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, v, \Lambda)$, then

$$\mathsf{hsame}(u, v).$$

Proof. Generate the canonical appended signature from the two component signatures by Theorem 6.48. Its continuation witness has the same endpoints as the displayed $\mathsf{Cont}(t, s, u)$, so continuation determinacy identifies the displayed recomposition with the canonical result. The asking-policy signature determinacy theorem compares the canonical appended signature with the supplied appended signature for v. Transitivity of hsame gives the displayed result. □

Lean4 ✓ `BEDC.FKernel.Sig.sigRel bundleAppend recompose hsame`

Theorem 6.53 (Three-bundle signature generation coherence)**.** *For probe bundles* Π, Θ, Ω *and source history* h*, assume* $\mathsf{Sig}(\Pi, h, s, \Delta)$*,* $\mathsf{Sig}(\Theta, h, t, \Gamma)$*, and* $\mathsf{Sig}(\Omega, h, w, \Xi)$*. Then there exist histories* a, b, u_L, u_R *and evidence ledgers* $\Lambda_{\Pi\Theta}$*,* Λ_L*,* $\Lambda_{\Theta\Omega}$*, and* Λ_R *such that*

1. $\mathsf{Sig}(\mathsf{BAppend}(\Pi, \Theta), h, a, \Lambda_{\Pi\Theta})$ *and* $\mathsf{Cont}(t, s, a)$*;*
2. $\mathsf{Sig}(\mathsf{BAppend}(\mathsf{BAppend}(\Pi, \Theta), \Omega), h, u_L, \Lambda_L)$ *and* $\mathsf{Cont}(w, a, u_L)$*;*
3. $\mathsf{Sig}(\mathsf{BAppend}(\Theta, \Omega), h, b, \Lambda_{\Theta\Omega})$ *and* $\mathsf{Cont}(w, t, b)$*;*
4. $\mathsf{Sig}(\mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)), h, u_R, \Lambda_R)$ *and* $\mathsf{Cont}(b, s, u_R)$*;*
5. $\mathsf{hsame}(u_L, u_R)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.signature generation three bundle coherence`

Proof. Apply Theorem 6.48 to the Π and Θ signatures to obtain a and $\Lambda_{\Pi\Theta}$ with $\mathsf{Cont}(t, s, a)$. Apply the same theorem to the resulting $\mathsf{BAppend}(\Pi, \Theta)$ signature and the Ω signature to obtain u_L and Λ_L with $\mathsf{Cont}(w, a, u_L)$. For the other grouping, first apply Theorem 6.48 to the Θ and Ω signatures to obtain b and $\Lambda_{\Theta\Omega}$ with $\mathsf{Cont}(w, t, b)$, then apply it to the Π signature and the $\mathsf{BAppend}(\Theta, \Omega)$ signature to obtain u_R and Λ_R with $\mathsf{Cont}(b, s, u_R)$. Theorem 4.56, with $h = w$, $k = t$, $l = s$, middle result b, and alternative middle result a, gives $\mathsf{hsame}(u_R, u_L)$. The history-sameness symmetry lemma converts this to the displayed $\mathsf{hsame}(u_L, u_R)$. □

Theorem 6.54 (Signature-sameness closure under bundle append)**.** *For probe bundles* Π, Θ *and histories* h, k*, if* $\mathsf{sameSig}_{\Pi}(h, k)$ *and* $\mathsf{sameSig}_{\Theta}(h, k)$*, then*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k).$$

Proof. Unpack the two signature-sameness witnesses. The Π witnesses give signatures from h and k whose answer histories are hsame; the Θ witnesses give another such pair. Apply Theorem 6.48 to the two h-side signatures and separately to the two k-side signatures. Continuation respect for hsame transports the two answer-history sameness witnesses across the appended continuations, so the appended signatures have hsame results. These appended signatures are the witnesses for $\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$. □

Lean4 ✓ `BEDC.FKernel.Sig.sameSig bundleAppend closure`

Theorem 6.55 (Signature-sameness left cancellation under bundle append)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$ *and* $h, k \in D$*. If*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$$

and $\mathsf{sameSig}_{\Pi}(h, k)$*, then*

$$\mathsf{sameSig}_{\Theta}(h, k).$$

Proof. Invert the appended signature-sameness witnesses using Theorem 6.49. This recovers Π signatures, Θ signatures, and continuation witnesses on the h and k sides. Compare the recovered left Π signatures with the witnesses from $\mathsf{sameSig}_{\Pi}(h, k)$ using Theorem 6.63 at h and at k. Transport one continuation along that left-result sameness. The common right continuation can then be cancelled against the appended result-sameness, yielding hsame for the recovered Θ results. Those recovered Θ signatures are the witnesses for $\mathsf{sameSig}_{\Theta}(h, k)$. □

Lean4 ✓ `BEDC.FKernel.Sig.sameSig bundleAppend left cancellation`

Theorem 6.56 (Signature-sameness right cancellation under bundle append)**.** *Assume* $\mathsf{BundleAskPolicy}(\Theta, D)$ *and* $h, k \in D$*. If*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k)$$

and $\mathsf{sameSig}_{\Theta}(h, k)$*, then*

$$\mathsf{sameSig}_{\Pi}(h, k).$$

Proof. Invert the appended signature-sameness witnesses using Theorem 6.49. This recovers Π signatures, Θ signatures, and continuation witnesses on the h and k sides. Compare the recovered Θ signatures with the supplied witnesses from $\mathsf{sameSig}_{\Theta}(h, k)$ using Theorem 6.63 at h and at k. Determinacy for the supplied and recovered Θ signatures gives hsame between the two recovered Θ continuation contexts. Continuation result transport then aligns one continuation so both continuations have the same left context and the same result up to hsame. The left-context cancellation clause of Theorem 4.52 applies to these aligned continuations and yields hsame for the recovered Π results. Those recovered Π signatures are the witnesses for $\mathsf{sameSig}_{\Pi}(h, k)$. □

Lean4 ✓ `BEDC.FKernel.Sig.sameSig bundleAppend right cancellation`

Theorem 6.57 (Known component residual for signature-sameness under bundle append)**.** *For probe bundles* Π, Θ *and histories* h, k*, the following residual equivalences hold.*

1. *If* $\mathsf{BundleAskPolicy}(\Pi, D)$*,* $h, k \in D$*, and* $\mathsf{sameSig}_{\Pi}(h, k)$*, then*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k) \iff \mathsf{sameSig}_{\Theta}(h, k).$$

2. *If* $\mathsf{BundleAskPolicy}(\Theta, D)$*,* $h, k \in D$*, and* $\mathsf{sameSig}_{\Theta}(h, k)$*, then*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k) \iff \mathsf{sameSig}_{\Pi}(h, k).$$

Proof. For the first equivalence, the forward direction is Theorem 6.55, using the known Π component and the bundle-local policy on Π. The reverse direction combines the known $\mathsf{sameSig}_{\Pi}(h, k)$ with the supplied $\mathsf{sameSig}_{\Theta}(h, k)$ by Theorem 6.54. The second equivalence is symmetric: its forward direction is Theorem 6.56, and its reverse direction is again the append closure theorem. □

Lean4 ✓ `BEDC.FKernel.Sig.sameSig bundleAppend residual iff`

Theorem 6.58 (Signature-sameness common-context residual under bundle append)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*,* $\mathsf{BundleAskPolicy}(\Omega, D)$*,* $h, k \in D$*,* $\mathsf{sameSig}_{\Pi}(h, k)$*, and* $\mathsf{sameSig}_{\Omega}(h, k)$*. Then*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\mathsf{BAppend}(\Theta,\Omega))}(h, k) \iff \mathsf{sameSig}_{\Theta}(h, k).$$

Proof. For the forward direction, cancel the known left component Π from the outer append using the residual equivalence, leaving $\mathsf{sameSig}_{\mathsf{BAppend}(\Theta,\Omega)}(h, k)$. Then cancel the known right component Ω using the right residual equivalence, yielding $\mathsf{sameSig}_{\Theta}(h, k)$.

For the reverse direction, first append the supplied middle witness with the known right witness $\mathsf{sameSig}_{\Omega}(h, k)$, then append the known left witness $\mathsf{sameSig}_{\Pi}(h, k)$ to rebuild the full common-context signature-sameness row. □

Lean4 ✓ `BEDC.FKernel.Sig.sameSig bundleAppend common context iff`

Theorem 6.59 (Cons tail signatures from result-sameness)**.** *If two derivations of* $\mathsf{Sig}(\mathsf{Bcons}(\pi, \Pi), h, r, \Delta)$ *and* $\mathsf{Sig}(\mathsf{Bcons}(\pi, \Pi), h, r', \Theta)$ *have* $\mathsf{hsame}(r, r')$*, then there exist tail results* s, t *such that* $\mathsf{Sig}(\Pi, h, s, \Delta_s)$*,* $\mathsf{Sig}(\Pi, h, t, \Delta_t)$*, and* $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig cons tail hsame from result hsame`

Proof. Invert both cons-signature derivations and their final extension witnesses. Cross-mark extension branches contradict result-sameness by history no-confusion, while matching marks expose matching outer history constructors; constructor injectivity yields hsame of the tail results. □

Theorem 6.60 (Cons head mark determinacy)**.** *Assume an asking policy on D. If $h \in D$ and two cons-signature derivations for* $\mathsf{Bcons}(\pi, \Pi)$ *start from the same source history h, then their head asking events emit internally same marks.*

Lean4 ✓ `BEDC.FKernel.Sig.sig cons head mark determinacy`

↪ variant `BEDC.FKernel.Sig.sig cons head marks same`
↪ variant `BEDC.FKernel.Sig.sig cons result hsame from policy`
↪ variant `BEDC.FKernel.Sig.sig cons tail hsame from policy`
↪ variant `BEDC.FKernel.Sig.sig cons policy result and tail`
↪ variant `BEDC.FKernel.Sig.sig cons head tail determinacy`
↪ variant `BEDC.FKernel.Sig.signature cons head mark determinacy`
↪ variant `BEDC.FKernel.Sig.sig cons witnesses and result hsame from policy`
↪ variant `BEDC.FKernel.Sig.sig cons head mark determinacy pair`

Proof. Invert both cons-signature derivations. Their head premises are asking events for the same probe and source history. Asking-policy determinacy gives internal sameness of the emitted marks, while the tail signature witnesses are retained from the two inverted derivations. □

Theorem 6.61 (Signature existence under an asking policy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If h is admitted by D, then bundle Π generates some signature event from h.*

Lean4 ✓ `BEDC.FKernel.Sig.sig total from policy`

↪ variant `BEDC.FKernel.Sig.signature totality under policy`
↪ variant `BEDC.FKernel.Sig.sigTotalOn from policy`
↪ variant `BEDC.FKernel.Sig.sigTotalOn tail of cons`
↪ variant `BEDC.FKernel.Sig.sigTotalOn cons iff`
↪ variant `BEDC.FKernel.Sig.sig total pair from policy`
↪ variant `BEDC.FKernel.Sig.sig total unique from policy`
↪ variant `BEDC.FKernel.Sig.sig total unique pair from policy`

Proof. Induct on the generated bundle Π. The empty bundle produces the empty answer-history. In the cons case, asking-policy totality supplies the first emitted mark and evidence token; the induction hypothesis supplies the tail signature; the emitted mark chooses the corresponding extension constructor. □

Theorem 6.62 (Bundle-local signature existence)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If $h \in D$, then there are an answer history s and evidence bundle Δ such that* $\mathsf{Sig}(\Pi, h, s, \Delta)$*.*

Proof. Induct on Π. The empty bundle emits the empty signature. In the cons case, the head occurrence satisfies $\pi \in \mathsf{Bcons}(\pi, \Pi)$, so bundle-local totality supplies the head asking event. The tail policy is obtained by restricting membership from $\rho \in \Pi$ to $\rho \in \mathsf{Bcons}(\pi, \Pi)$, and the induction hypothesis supplies the tail signature. The emitted head mark then selects the matching extension constructor. □

Lean4 ✓ `BEDC.FKernel.Sig.sig total from bundle policy`

Theorem 6.63 (Bundle-local signature determinacy)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If* $h \in D$ *and*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, h, t, \Theta),$$

then $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig deterministic from bundle policy`

Proof. Induct on Π. The empty case is reflexive. In the cons case, invert the two signature derivations. Bundle-local mark determinacy applies to the shared head occurrence, and the tail policy follows from tail membership. The induction hypothesis gives sameness of the tail results, and extension respect combines the head mark-sameness with the tail history-sameness. □

Theorem 6.64 (Bundle-local signature source respect)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If* $\mathsf{hsame}(h, k)$ *and*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta),$$

then $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig respects history from bundle policy`

Proof. Induct on Π. The empty case gives empty signatures on both sides. In the cons case, invert both signature derivations. Apply the bundle-local source-history respect field to the shared head occurrence and the premise $\mathsf{hsame}(h, k)$. The tail policy is obtained by restricting bundle membership, and the induction hypothesis applies to the two tail signatures. Extension respect combines the head mark-sameness with the tail history-sameness. □

Theorem 6.65 (Bundle-local history-sameness gives signature-sameness)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If* $h, k \in D$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{sameSig}_\Pi(h, k)$*.*

Proof. Apply bundle-local signature existence to h and to k, obtaining generated signatures

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta).$$

Bundle-local signature source respect transports the premise $\mathsf{hsame}(h, k)$ across these two generated signatures and yields $\mathsf{hsame}(s, t)$. The two signature witnesses together with this generated-history sameness form $\mathsf{sameSig}_\Pi(h, k)$. □

Lean4 ✓ `BEDC.FKernel.Sig.bundle local sameSig of hsame`

Theorem 6.66 (Bundle-local signature-sameness equivalence)**.** *Assume* $\mathsf{BundleAskPolicy}(\Pi, D)$*. If* $h \in D$*, then* $\mathsf{sameSig}_\Pi(h, h)$*; if* $\mathsf{sameSig}_\Pi(h, k)$*, then* $\mathsf{sameSig}_\Pi(k, h)$*; and if* $k \in D$*,* $\mathsf{sameSig}_\Pi(h, k)$*, and* $\mathsf{sameSig}_\Pi(k, l)$*, then* $\mathsf{sameSig}_\Pi(h, l)$*.*

Proof. For reflexivity, bundle-local signature existence gives a signature generated from the admitted history h, and hsame-reflexivity relates that generated answer history to itself. For symmetry, unpack the two signature witnesses, swap their order, and apply hsame-symmetry. For transitivity through k, unpack both signature-sameness hypotheses. The admitted middle history k has two generated signature results; bundle-local signature determinacy identifies them by hsame. Composing with hsame-transitivity connects the left answer history to the right answer history, yielding $\mathsf{sameSig}_\Pi(h, l)$. □

Lean4 ✓ `BEDC.FKernel.Sig.bundle local signature sameness equivalence`

Theorem 6.67 (Signature determinacy under an asking policy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* h *is admitted by* D *and*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, h, t, \Theta),$$

then $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig deterministic`
↪ variant `BEDC.FKernel.Sig.sig cons determinacy spine`

Proof. Induct on the generated bundle Π. The empty case uses reflexivity of hsame. In the cons case, asking determinacy gives msame of the first emitted marks. The induction hypothesis gives hsame of the tail signatures. Extension respects mark-sameness and history-sameness, so the resulting answer histories are hsame. □

Theorem 6.68 (Signature generation respects source history-sameness)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $\mathsf{hsame}(h, k)$*,*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta),$$

then $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sig respects history`

Proof. Induct on Π. The empty case follows by reflexivity. In the cons case, invert the two signature derivations. Asking-policy history-respect transports the head asking events across $\mathsf{hsame}(h, k)$, the induction hypothesis transports the tail signatures, and extension respect combines the resulting mark-sameness and history-sameness for the generated histories. □

Lemma 6.69 (Continuation fixed-length split uniqueness)**.** *Let* s, s', t, t', u, u' *be histories. Suppose*

$$\mathsf{Cont}(t, s, u), \qquad \mathsf{Cont}(t', s', u'), \qquad \mathsf{hsame}(u, u').$$

If s *and* s' *have the same external binary length, then*

$$\mathsf{hsame}(s, s') \quad \textit{and} \quad \mathsf{hsame}(t, t').$$

Proof. Induct simultaneously on the generated histories s and s'. In the empty-empty branch, both continuation witnesses are right-unit witnesses from Definition 4.10, so the displayed result sameness gives $\mathsf{hsame}(t, t')$, while $\mathsf{hsame}(s, s')$ is reflexive. The empty-visible and visible-empty branches contradict equality of the external binary lengths by natural-number constructor no-confusion. In matching visible branches, inversion of the two continuation witnesses exposes matching visible outer constructors on u and u'. Constructor injectivity reduces the displayed result sameness to the two tail continuation results, and successor injectivity reduces the length equality to the tail lengths. The induction hypothesis gives sameness of the visible tails and of t with t', and the same visible constructor rebuilds $\mathsf{hsame}(s, s')$. Opposite visible branches contradict history constructor no-confusion at the result comparison. □

Theorem 6.70 (Signature-sameness exact split under bundle append)**.** *For probe bundles* Π, Θ *and histories* h, k*, if*

$$\mathsf{sameSig}_{\mathsf{BAppend}(\Pi,\Theta)}(h, k),$$

then

$$\mathsf{sameSig}_{\Pi}(h, k) \quad \textit{and} \quad \mathsf{sameSig}_{\Theta}(h, k).$$

Proof. Unpack the appended signature-sameness witness. It gives appended signatures from h and from k with answer histories u_h and u_k satisfying $\mathsf{hsame}(u_h, u_k)$. Apply Definition 6.49 to the appended signature from h. This recovers histories s_h, t_h and evidence ledgers with

$$\mathsf{Sig}(\Pi, h, s_h, \Delta_h), \qquad \mathsf{Sig}(\Theta, h, t_h, \Gamma_h), \qquad \mathsf{Cont}(t_h, s_h, u_h).$$

Apply the same inversion to the appended signature from k, recovering histories s_k, t_k with

$$\mathsf{Sig}(\Pi, k, s_k, \Delta_k), \qquad \mathsf{Sig}(\Theta, k, t_k, \Gamma_k), \qquad \mathsf{Cont}(t_k, s_k, u_k).$$

By Definition 6.46, the two Π results s_h and s_k both have external binary length $\mathsf{BLEN}(\Pi)$. Hence they have the same external binary length. The fixed-length continuation split lemma applied to the two displayed continuation witnesses and to $\mathsf{hsame}(u_h, u_k)$ yields

$$\mathsf{hsame}(s_h, s_k) \quad \text{and} \quad \mathsf{hsame}(t_h, t_k).$$

The recovered Π signatures together with $\mathsf{hsame}(s_h, s_k)$ are exactly a witness for $\mathsf{sameSig}_\Pi(h, k)$, and the recovered Θ signatures together with $\mathsf{hsame}(t_h, t_k)$ are exactly a witness for $\mathsf{sameSig}_\Theta(h, k)$. □

Theorem 6.71 (Three-bundle signature cut inversion)**.** *For probe bundles Π, Θ, Ω, source history h, and a bundle A with*

$$A = \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega)),$$

if

$$\mathsf{Sig}(A, h, u, \Lambda),$$

then there exist histories s, t, w, b, u' and evidence ledgers Δ, Γ, Ξ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Sig}(\Omega, h, w, \Xi),$$

$$\mathsf{Cont}(w, t, b), \qquad \mathsf{Cont}(b, s, u'), \qquad \mathsf{hsame}(u, u').$$

Proof. Apply Definition 6.49 to the given signature for the outer append $A = \mathsf{BAppend}(\Pi, \mathsf{BAppend}(\Theta, \Omega))$. This gives histories s and b, together with evidence ledgers Δ and Ψ, and a bundle B with

$$B = \mathsf{BAppend}(\Theta, \Omega),$$

such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(B, h, b, \Psi), \qquad \mathsf{Cont}(b, s, u).$$

Apply the same append-inversion theorem to the recovered signature for $B = \mathsf{BAppend}(\Theta, \Omega)$. This gives histories t and w, together with evidence ledgers Γ and Ξ, such that

$$\mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Sig}(\Omega, h, w, \Xi), \qquad \mathsf{Cont}(w, t, b).$$

Take $u' = u$. The displayed outer continuation is then exactly $\mathsf{Cont}(b, s, u')$, and $\mathsf{hsame}(u, u')$ follows by reflexivity of history-sameness. These witnesses are the displayed three component signatures and the two continuation cuts. □

Theorem 6.72 (Signature empty-result iff empty bundle)**.** *If $\mathsf{Sig}(\Pi, h, r, \Delta)$, then*

$$\mathsf{hsame}(r, \mathsf{e}) \iff \Pi = \mathsf{Bnil}.$$

Proof. For the forward implication, assume $\mathsf{hsame}(r, \mathsf{e})$ and classify the bundle Π. The empty-bundle case is immediate. In the cons case, write $\Pi = \mathsf{Bcons}(\pi, \Pi_0)$. The given signature premise is then a cons-bundle signature for result r, so Definition 6.45 rules out $\mathsf{hsame}(r, \mathsf{e})$, a contradiction. Hence the cons case is impossible and $\Pi = \mathsf{Bnil}$.

For the reverse implication, assume $\Pi = \mathsf{Bnil}$ and substitute this into the signature premise. Inverting the resulting empty-bundle signature leaves only the empty-signature rule, so the produced result history is internally the same as e, exactly as in Definition 6.43. □

Theorem 6.73 (Duplicate-probe signature contraction obstruction)**.** *For every probe π, probe bundle Π, source history h, and result histories r, s, if*

$$\mathsf{Sig}(\mathsf{Bcons}(\pi, \mathsf{Bcons}(\pi, \Pi)), h, r, \Delta) \qquad \textit{and} \qquad \mathsf{Sig}(\mathsf{Bcons}(\pi, \Pi), h, s, \Gamma),$$

then

$$\neg\, \mathsf{hsame}(r, s).$$

Proof. Apply Definition 6.46 to the two displayed signature derivations. The first result has external binary length $\mathsf{BLEN}(\mathsf{Bcons}(\pi, \mathsf{Bcons}(\pi, \Pi))) = 2 + \mathsf{BLEN}(\Pi)$, while the second has external binary length $\mathsf{BLEN}(\mathsf{Bcons}(\pi, \Pi)) = 1 + \mathsf{BLEN}(\Pi)$ by Definition 6.3. If $\mathsf{hsame}(r, s)$ held, the internal equality reading of history-sameness would identify the two generated histories and therefore identify their external binary lengths. This would force

$$2 + \mathsf{BLEN}(\Pi) = 1 + \mathsf{BLEN}(\Pi),$$

contradicting natural-number successor no-confusion. Hence the two generated signature histories cannot be internally history-same. □

Theorem 6.74 (Signature append cut decomposition uniqueness)**.** *For probe bundles Π, Θ, A, domain D, and source history h, assume $A = \mathsf{BAppend}(\Pi, \Theta)$, $\mathsf{BundleAskPolicy}(\Pi, D)$, $\mathsf{BundleAskPolicy}(\Theta, D)$, and $h \in D$. If*

$$\mathsf{Sig}(A, h, u, \Lambda)$$

has two append-inversion decompositions

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Cont}(t, s, u),$$

and

$$\mathsf{Sig}(\Pi, h, s', \Delta'), \qquad \mathsf{Sig}(\Theta, h, t', \Gamma'), \qquad \mathsf{Cont}(t', s', u),$$

then

$$\mathsf{hsame}(s, s') \quad \textit{and} \quad \mathsf{hsame}(t, t').$$

Proof. The displayed rows are two decompositions of the same appended signature, but component uniqueness is local to the two recovered bundles. Apply Definition 6.63 to the policy $\mathsf{BundleAskPolicy}(\Pi, D)$, the admitted source history $h \in D$, and the two recovered Π signatures

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, h, s', \Delta').$$

This gives $\mathsf{hsame}(s, s')$. Apply the same determinacy theorem to the policy $\mathsf{BundleAskPolicy}(\Theta, D)$, the same admitted source history, and the two recovered Θ signatures

$$\mathsf{Sig}(\Theta, h, t, \Gamma), \qquad \mathsf{Sig}(\Theta, h, t', \Gamma').$$

This gives $\mathsf{hsame}(t, t')$. The appended signature premise, together with $A = \mathsf{BAppend}(\Pi, \Theta)$ and the two continuation witnesses, certifies that both pairs are append-inversion cuts of the same event; the two component comparisons follow from bundle-local determinacy on the fixed source. □

6.3 Signature sameness

Definition 6.75 (Signature-sameness)**.** *Given a bundle* Π*, histories* h, k *are signature-same, written* $\mathsf{sameSig}_\Pi(h, k)$*, if there exist generated signatures*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta),$$

such that $\mathsf{hsame}(s, t)$*.*

Lean4 (definition) `BEDC.FKernel.Sig.SameSig`

Lean4 ✓ `BEDC.FKernel.Sig.sameSig witnesses`

↪ variant `BEDC.FKernel.Sig.signature sameness witnesses`

↪ variant `BEDC.FKernel.Sig.sameSig intro from witnesses`

↪ variant `BEDC.FKernel.Sig.sameSig left witness`

↪ variant `BEDC.FKernel.Sig.sameSig right witness`

↪ variant `BEDC.FKernel.Sig.sameSig hsame witness`

↪ variant `BEDC.FKernel.Sig.sameSig witness pair symm`

↪ variant `BEDC.FKernel.Sig.sameSig refl from witness`

Theorem 6.76 (Empty-bundle signature-sameness)**.** *For all histories* h, k*,* $\mathsf{sameSig}_{\mathsf{Bnil}}(h, k)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sameSig empty bundle`

Proof. Take both generated signatures for Bnil to be the empty signature with result e. The two results are then related by hsame reflexivity, and these witnesses form $\mathsf{sameSig}_{\mathsf{Bnil}}(h, k)$. □

Theorem 6.77 (History-sameness gives signature-sameness under policy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $h, k \in D$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{sameSig}_\Pi(h, k)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sameSig of hsame from ask policy`

↪ variant `BEDC.FKernel.Sig.sameSig of hsame witnesses under policy`

↪ variant `BEDC.FKernel.Sig.signature totality respects hsame witnesses`

↪ variant `BEDC.FKernel.Sig.signature totality witness chain`

Proof. Obtain total signatures for h and k from the asking policy. The same policy transports $\mathsf{hsame}(h, k)$ across signature generation, yielding hsame of the generated signatures. These witnesses form $\mathsf{sameSig}_\Pi(h, k)$. □

Theorem 6.78 (Signature-sameness equivalence)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $h \in D$*, then* $\mathsf{sameSig}_\Pi(h, h)$*; if* $\mathsf{sameSig}_\Pi(h, k)$*, then* $\mathsf{sameSig}_\Pi(k, h)$*; and if* $k \in D$*,* $\mathsf{sameSig}_\Pi(h, k)$*, and* $\mathsf{sameSig}_\Pi(k, l)$*, then* $\mathsf{sameSig}_\Pi(h, l)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.sameSig equivalence`

↪ variant `BEDC.FKernel.Sig.signature sameness equivalence`

↪ variant `BEDC.FKernel.Sig.signature sameness equivalence policy spine`

↪ variant `BEDC.FKernel.Sig.sameSig equivalence under policy`

↪ variant `BEDC.FKernel.Sig.sameSig equivalence policy spine fields`

↪ variant `BEDC.FKernel.Sig.sameSig equivalence from total and determinacy`

↪ variant `BEDC.FKernel.Sig.sameSig refl under policy`

↪ variant `BEDC.FKernel.Sig.sameSig trans under policy`

↪ variant `BEDC.FKernel.Sig.sameSig trans with middle witness`

↪ variant BEDC.FKernel.Sig.sameSig middle witnesses hsame under policy
↪ variant BEDC.FKernel.Sig.sameSig trans explicit witness chain
↪ variant BEDC.FKernel.Sig.sameSig trans witnesses under policy
↪ variant BEDC.FKernel.Sig.sameSig trans chain witnesses under policy
↪ variant BEDC.FKernel.Sig.sameSig three step witness chain under policy
↪ variant BEDC.FKernel.Sig.sameSig four step witness chain under policy
↪ variant BEDC.FKernel.Sig.sameSig four step chain under policy
↪ variant BEDC.FKernel.Sig.sameSig trans from middle determinacy
↪ variant BEDC.FKernel.Sig.sameSig trans witness chain with result
↪ variant BEDC.FKernel.Sig.signature sameness equivalence total determinacy
↪ variant BEDC.FKernel.Sig.sameSig equivalence witness pack
↪ variant BEDC.FKernel.Sig.sameSig equivalence explicit witnesses

Proof. For an admitted h, policy totality produces a generated signature for h, and hsame-reflexivity gives $\mathsf{sameSig}_\Pi(h, h)$. Symmetry reverses the two signature witnesses and uses hsame-symmetry, with no admission hypothesis. For transitivity through k, unpack the two signature-sameness witnesses. They contain two generated signatures from the admitted middle history k. Signature determinacy under $\mathsf{AskPolicy}(\Pi, D)$ and $k \in D$ relates those two middle results; two applications of hsame-transitivity then connect a signature of h with a signature of l. □

Theorem 6.79 (Signature-sameness endpoint replacement)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $h, h', k, k' \in D$*,* $\mathsf{hsame}(h, h')$*,* $\mathsf{sameSig}_\Pi(h, k)$*, and* $\mathsf{hsame}(k, k')$*, then*

$$\mathsf{sameSig}_\Pi(h', k').$$

Proof. By Theorem 6.77, the left endpoint hypothesis gives $\mathsf{sameSig}_\Pi(h, h')$, and the right endpoint hypothesis gives $\mathsf{sameSig}_\Pi(k, k')$. Symmetry from Theorem 6.78 turns the first relation into $\mathsf{sameSig}_\Pi(h', h)$. Transitivity through the middle h yields $\mathsf{sameSig}_\Pi(h', k)$, and transitivity through the middle k yields $\mathsf{sameSig}_\Pi(h', k')$. □

Lean4 ✓ BEDC.FKernel.Sig.sameSig endpoint replacement
↪ variant BEDC.FKernel.Sig.sameSig endpoint replacement under policy

Chapter 7

Packages and Package Policies

Definition 7.1 (Package token and package-sameness)**.** Pkg *is a judgment*

$$\mathsf{Pkg}(\Pi, s, p)$$

meaning that answer-history s under bundle Π is represented by package token p. The internal package-sameness judgment is $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Package.package token and psame generated pair`

Definition 7.2 (Package policy)**.** *A package policy* $\mathsf{PackagePolicy}(\Pi)$ *consists of the following certified fields:*

1. *existence: each generated answer-history s has a package token p with* $\mathsf{Pkg}(\Pi, s, p)$*;*
2. *extensionality: if* $\mathsf{hsame}(s, t)$*,* $\mathsf{Pkg}(\Pi, s, p)$*, and* $\mathsf{Pkg}(\Pi, t, q)$*, then* $\mathsf{psame}(p, q)$*;*
3. *grounding: if* $\mathsf{psame}(p, q)$*, then there are answer histories s, t with* $\mathsf{Pkg}(\Pi, s, p)$*,* $\mathsf{Pkg}(\Pi, t, q)$*, and* $\mathsf{hsame}(s, t)$*;*
4. *visibility: only the package token and its ledger are visible at the package interface.*

Theorem 7.3 (Packages classify signatures)**.** *Under* $\mathsf{PackagePolicy}(\Pi)$*, if* $\mathsf{Pkg}(\Pi, s, p)$ *and* $\mathsf{Pkg}(\Pi, t, q)$*, then* $\mathsf{hsame}(s, t)$ *implies* $\mathsf{psame}(p, q)$*. Conversely, if* $\mathsf{psame}(p, q)$*, then there exist answer histories u, v such that* $\mathsf{Pkg}(\Pi, u, p)$*,* $\mathsf{Pkg}(\Pi, v, q)$*, and* $\mathsf{hsame}(u, v)$*.*

Lean4 ✓ `BEDC.FKernel.Package.packagePolicy classifies signature witnesses`

Proof. The first direction is the extensionality field. The second direction is the grounding field. □

Theorem 7.4 (Package representative for generated answer-history)**.** *Assume* $\mathsf{PackagePolicy}(\Pi)$*. For every generated answer-history s, there exists a package token p such that* $\mathsf{Pkg}(\Pi, s, p)$ *and, for every answer-history t and package token q, if* $\mathsf{hsame}(s, t)$ *and* $\mathsf{Pkg}(\Pi, t, q)$*, then* $\mathsf{psame}(p, q)$*.*

Proof. Let s be a generated answer-history. The existence field of $\mathsf{PackagePolicy}(\Pi)$ supplies a package token p with $\mathsf{Pkg}(\Pi, s, p)$. It remains to verify the displayed universal property. Let t and q be such that $\mathsf{hsame}(s, t)$ and $\mathsf{Pkg}(\Pi, t, q)$. Applying the first direction of Definition 7.3 to $\mathsf{Pkg}(\Pi, s, p)$, $\mathsf{Pkg}(\Pi, t, q)$, and $\mathsf{hsame}(s, t)$ gives $\mathsf{psame}(p, q)$. □

Chapter 8

Gap Policies, Coverage, Separation, and Composition

8.1 Gap policy

Definition 8.1 (Gap membership)**.** *The finite-kernel judgment* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ *means that source history* h *lies in the signature gap ledger of package token* p *under bundle* Π *and source specification* D*.*

Definition 8.2 (Gap policy)**.** *A gap policy* $\mathsf{GapPolicy}(\Pi, D)$ *over source specification* D *consists of:*

1. *provenance/generation: the membership judgment* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ *itself exposes the admitted-source judgment* $\mathsf{InDom}(D, h)$ *and witnesses* s, Δ *with* $\mathsf{Sig}(\Pi, h, s, \Delta)$ *and* $\mathsf{TokIntro}(\Pi, s, p)$*; the generation field of* $\mathsf{GapPolicy}(\Pi, D)$ *records the token-witness projection from such a membership to some* s *with* $\mathsf{TokIntro}(\Pi, s, p)$*;*
2. *coverage: every admitted source history belongs to some package gap;*
3. *separation: if the same admitted source history belongs to the gaps of* p *and* q*, then* $\mathsf{psame}(p, q)$*.*

The ledger remains part of the package interface: a visible package token is used together with its gap ledger, but ledger visibility is not a separate component of $\mathsf{GapPolicy}(\Pi, D)$*.*

Lean4 (definition) `BEDC.FKernel.Gap.GapPolicy`
Lean4 (definition) `BEDC.FKernel.Gap.GapPolicyInterface`
Lean4 ✓ `BEDC.FKernel.Gap.gap generation witness`
↪ variant `BEDC.FKernel.Gap.inGapSig domain witness`
↪ variant `BEDC.FKernel.Gap.inGapSig signature witness`
↪ variant `BEDC.FKernel.Gap.gap policy fields`
↪ variant `BEDC.FKernel.Gap.GapPolicy iff fields`
↪ variant `BEDC.FKernel.Gap.gapPolicy generation field`
↪ variant `BEDC.FKernel.Gap.gapPolicy signature determination`
↪ variant `BEDC.FKernel.Gap.GapPolicyInterface generation projection`

Theorem 8.3 (Gap coverage)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. For every source history* h*, if* $\mathsf{InDom}(D, h)$ *then there exists a package token* p *such that* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap coverage`

Proof. This is the coverage field of GapPolicy, which directly supplies a package token whose finite-kernel gap judgment contains the admitted source history. □

Theorem 8.4 (Gap separation)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. If* $\mathsf{InDom}(D, h)$*,* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$*, and* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h)$*, then* $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap separation`

Proof. This is the separation field of GapPolicy, applied to the two finite-kernel gap memberships of the same admitted source history. □

Theorem 8.5 (Gap separation from memberships)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$*. If* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ *and* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h)$*, then* $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap separation from memberships`

Proof. Either finite-kernel gap membership exposes the admitted-source judgment $\mathsf{InDom}(D, h)$ by the membership definition recorded in Definition 8.2. Apply Theorem 8.4 to this admitted-source witness and to the two memberships over the same source history h. □

Theorem 8.6 (Gap representative for admitted source)**.** *Assume* $\mathsf{GapPolicy}(\Pi, D)$ *and* $\mathsf{InDom}(D, h)$*. Then there exists a package token* p *such that* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ *and, for every package token* q*, if* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h)$ *then* $\mathsf{psame}(p, q)$*.*

Proof. Coverage supplies a package token p with $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$. For any package token q with $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h)$, separation applies to the two finite-kernel gap memberships over the same admitted source history h and gives $\mathsf{psame}(p, q)$. □

Lean4 ✓ `BEDC.FKernel.Gap.gap representative for admitted source`

8.2 Composite gaps

Definition 8.7 (Composite gap)**.** *Suppose an interface* C *packages sources into intermediate visible tokens and an interface* D *packages those intermediate tokens into final visible tokens. The composite gap judgment*

$$\mathsf{CompGap}_{D \circ C}(z, x)$$

means that source x *lies behind final visible token* z *through some intermediate token. Externally this is described by the existence of an intermediate token* y *with*

$$\mathsf{InGap}_C(y, x) \quad \textit{and} \quad \mathsf{InGap}_D(z, y).$$

This is a metalanguage description of the composite ledger policy.

Lean4 (definition) `BEDC.FKernel.Gap.CompGap`
↪ variant `BEDC.FKernel.Gap.compGap iff exists`
↪ variant `BEDC.FKernel.Gap.compGap witness pair iff`
↪ variant `BEDC.FKernel.Gap.composite gap ledger witnesses`
↪ variant `BEDC.FKernel.Gap.composite gap ledger`
Lean4 ✓ `BEDC.FKernel.Gap.compGap intro`
↪ variant `BEDC.FKernel.Gap.compGap left witness`
Lean4 ✓ `BEDC.FKernel.Gap.compGap right witness`

```
↪ variant  BEDC.FKernel.Gap.compGap left right witness pair
↪ variant  BEDC.FKernel.Gap.compGap second first witness
↪ variant  BEDC.FKernel.Gap.compGap right nested witnesses
```

Theorem 8.8 (Composite gap coverage)**.** *If the first interface has gap coverage in the form that every admitted source h has an intermediate token y with* $\mathsf{InGap}_C(y, h)$ *and an admissibility certificate* $\mathsf{InterOk}(y)$*, and if the second interface has gap coverage for every intermediate token satisfying* $\mathsf{InterOk}$*, then the composite interface has gap coverage.*

```
Lean4 ✓  BEDC.FKernel.Gap.compGap coverage from layers
↪ variant  BEDC.FKernel.Gap.compGap composite coverage witness
↪ variant  BEDC.FKernel.Gap.composite gap coverage spine
↪ variant  BEDC.FKernel.Gap.compGap coverage intermediate and final
↪ variant  BEDC.FKernel.Gap.compGap coverage with intermediate ok
↪ variant  BEDC.FKernel.Gap.compGap coverage final witness pair
```

Proof. Given an admitted source, first coverage supplies both an intermediate token whose first gap contains the source and the admissibility certificate for that intermediate token. Second coverage applies to that certificate and produces a final token whose second gap contains the intermediate token. Thus the source belongs to the composite gap of the final token. □

Theorem 8.9 (Composite gap separation)**.** *If the first gap separates two intermediate witnesses over the same source by* $\mathsf{interSame}$*, and the second gap sends* $\mathsf{interSame}$ *intermediate witnesses to* $\mathsf{finalSame}$ *final tokens, then two composite memberships over the same source imply* $\mathsf{finalSame}(z_1, z_2)$*.*

```
Lean4 ✓  BEDC.FKernel.Gap.compGap separation from layers
```

Proof. Unpack the first composite membership into an intermediate witness y_1 with $\mathsf{InGap}_C(y_1, h)$ and $\mathsf{InGap}_D(z_1, y_1)$. Unpack the second composite membership into y_2 with $\mathsf{InGap}_C(y_2, h)$ and $\mathsf{InGap}_D(z_2, y_2)$. First-layer separation gives $\mathsf{interSame}(y_1, y_2)$, and second-layer separation carries this relation across the two second-layer memberships to obtain $\mathsf{finalSame}(z_1, z_2)$. □

Theorem 8.10 (Composite exactness from layered exactness)**.** *Let* Source*,* Inter*, and* Final *be the source, intermediate, and final token types. Let* $\mathsf{SourceOk}$ *and* $\mathsf{InterOk}$ *be admissibility predicates, let* $\mathsf{firstGap}$ *and* $\mathsf{secondGap}$ *be the first and second gap relations, and let* $\mathsf{interSame}$ *and* $\mathsf{finalSame}$ *be the intermediate and final sameness relations. Assume:*

$$\forall x,\ \mathsf{SourceOk}(x) \to \exists y,\ \mathsf{firstGap}(y, x) \text{ and } \mathsf{InterOk}(y),$$

$$\forall y,\ \mathsf{InterOk}(y) \to \exists z,\ \mathsf{secondGap}(z, y),$$

$$\forall x, y_1, y_2,\ \mathsf{SourceOk}(x) \to \mathsf{firstGap}(y_1, x) \to \mathsf{firstGap}(y_2, x) \to \mathsf{interSame}(y_1, y_2),$$

and

$$\forall y_1, y_2, z_1, z_2,\ \mathsf{interSame}(y_1, y_2) \to \mathsf{secondGap}(z_1, y_1) \to \mathsf{secondGap}(z_2, y_2) \to \mathsf{finalSame}(z_1, z_2).$$

Then the composite gap satisfies both conclusions:

$$\forall x,\ \mathsf{SourceOk}(x) \to \exists z,\ \mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z, x)$$

and

$$\forall x, z_1, z_2,\ \mathsf{SourceOk}(x) \to \mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z_1, x) \to \mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z_2, x) \to \mathsf{finalSame}$$

Lean4 ✓ `BEDC.FKernel.Gap.compGap exactness from layers on histories`

Proof. The first conclusion is Theorem 8.8: choose the first-layer intermediate witness and its InterOk certificate, then apply second-layer coverage to obtain a final token. The second conclusion is Theorem 8.9: unpack the two composite witnesses over the same source, use first-layer separation to identify the intermediate witnesses by interSame, and apply second-layer transport to obtain finalSame for the final tokens. □

Theorem 8.11 (Composite representative for admitted source)**.** *Under the same layered coverage and separation hypotheses as Theorem 8.10, if* $\mathsf{SourceOk}(x)$*, then there exists a final token* z *such that*

$$\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z, x)$$

and every final token z' *satisfying*

$$\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z', x)$$

also satisfies $\mathsf{finalSame}(z, z')$*.*

Proof. Composite coverage supplies a final token z with $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z, x)$. For any other final token z' with $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z', x)$, composite separation applies to the two composite memberships over the same admitted source and gives $\mathsf{finalSame}(z, z')$. □

Lean4 ✓ `BEDC.FKernel.Gap.compGap representative for admitted source`

Principle 8.12 (Ledger composition)**.** *The ledger of a composite compression is not the last ledger alone. It is the first ledger together with the pullback of the second ledger along the first visible interface.*

Lean4 ✓ `BEDC.FKernel.Gap.compGap inversion`

↪ variant `BEDC.FKernel.Gap.compGap witness from layers`
↪ variant `BEDC.FKernel.Gap.compGap assoc`
↪ variant `BEDC.FKernel.Gap.compGap assoc forward`
↪ variant `BEDC.FKernel.Gap.compGap assoc witnesses`
↪ variant `BEDC.FKernel.Gap.compGap assoc witness pack`
↪ variant `BEDC.FKernel.Gap.compGap assoc iff witnesses`
↪ variant `BEDC.FKernel.Gap.compGap assoc backward`
↪ variant `BEDC.FKernel.Gap.ledger composition principle`
↪ variant `BEDC.FKernel.Gap.ledger composition principle iff`
↪ variant `BEDC.FKernel.Gap.compGap left nested witnesses`
↪ variant `BEDC.FKernel.Gap.compGap witnesses to right nested`
↪ variant `BEDC.FKernel.Gap.compGap right nested iff witnesses`
↪ variant `BEDC.FKernel.Gap.compGap three witnesses iff`
↪ variant `BEDC.FKernel.Gap.compGap four witnesses iff`
↪ variant `BEDC.FKernel.Gap.compGap pullback witnesses`
↪ variant `BEDC.FKernel.Gap.compGap four right nested iff`
↪ variant `BEDC.FKernel.Gap.ledger composition principle inverse`

Theorem 8.13 (Composite gap intermediate-witness sameness)**.** *Let* Source*,* Inter*, and* Final *be the source, intermediate, and final token types. Let* $\mathsf{SourceOk}$ *be the admitted-source predicate, let* $\mathsf{firstGap}$ *and* $\mathsf{secondGap}$ *be the layered gap relations, and let* $\mathsf{interSame}$ *be the intermediate sameness relation. Assume first-layer separation:*

$$\forall h, y_1, y_2,\ \mathsf{SourceOk}(h) \to \mathsf{firstGap}(y_1, h) \to \mathsf{firstGap}(y_2, h) \to \mathsf{interSame}(y_1, y_2).$$

For an admitted source h *and final tokens* z_1, z_2*, suppose the two composite memberships over* h *are presented with intermediate witnesses* y_1 *and* y_2 *in the sense that*

$$\mathsf{firstGap}(y_1, h) \wedge \mathsf{secondGap}(z_1, y_1)$$

and

$$\mathsf{firstGap}(y_2, h) \wedge \mathsf{secondGap}(z_2, y_2).$$

These two displayed witness pairs exhibit $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z_1, h)$ *and* $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, z_2,$ *by Definition 8.7. If* $\mathsf{SourceOk}(h)$*, then* $\mathsf{interSame}(y_1, y_2)$*.*

Proof. The first displayed witness pair gives $\mathsf{firstGap}(y_1, h)$, and the second gives $\mathsf{firstGap}(y_2, h)$. Applying the first-layer separation hypothesis to the admitted-source certificate for h and to these two first-layer memberships gives $\mathsf{interSame}(y_1, y_2)$. □

Theorem 8.14 (Composite-gap witness inversion is conservative over final representatives)**.** *Under the same layered coverage and separation hypotheses as Definition 8.10, if* $\mathsf{SourceOk}(x)$*,* $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGa}$ *and*

$$\exists y,\ \mathsf{firstGap}(y, x) \wedge \mathsf{secondGap}(w, y),$$

then $\mathsf{finalSame}(z, w)$*.*

Proof. Choose y witnessing the displayed existential. By Definition 8.7, the relations $\mathsf{firstGap}(y, x)$ and $\mathsf{secondGap}(w, y)$ exhibit $\mathsf{CompGap}(\mathsf{firstGap}, \mathsf{secondGap}, w, x)$. Applying Definition 8.9 to the admitted-source certificate for x, to the given composite membership of z, and to this exhibited composite membership of w gives $\mathsf{finalSame}(z, w)$. □

Chapter 9

Typed Naming Certificates

Definition 9.1 (Source specification)**.** *A source specification* SourceSpec *records the generated material from which a proposed name is licensed. It may refer to raw marks, histories, bundles, packages, gaps, or previously licensed names.*

Definition 9.2 (Pattern specification)**.** *A pattern specification* PatternSpec *records the stable behavior observed in the source. Examples include repetition, continuation, signature classification, package stability, or compatible threading.*

Definition 9.3 (Classifier specification)**.** *A classifier specification* ClassifierSpec *records how source material is grouped. At the finite kernel level this is normally given by a bundle, an asking policy, a package policy, and a gap policy.*

Definition 9.4 (Stability certificates)**.** *The stability component* StabilityCert *is typed by modes:*

$$\begin{aligned}
\mathsf{ClosureCert} &: \textit{the source is closed under the named behavior},\\
\mathsf{ReuseCert} &: \textit{the named behavior can be reused as a later tool},\\
\mathsf{DescentCert} &: \textit{the behavior descends through classification},\\
\mathsf{CompositionCert} &: \textit{the behavior composes with previously named behavior},\\
\mathsf{SealCert} &: \textit{the behavior supports compatible-thread sealing}.
\end{aligned}$$

A name may require one or more of these modes.

```
Lean4 (definition)  BEDC.FKernel.NameCert.StabilityMode
  ↪ variant  BEDC.FKernel.NameCert.stabilityMode no confusion
  ↪ variant  BEDC.FKernel.NameCert.stabilityMode pairwise no confusion
  ↪ variant  BEDC.FKernel.NameCert.stabilityMode descent composition seal no confusion
  ↪ variant  BEDC.FKernel.NameCert.stabilityMode exhaustive
```

Definition 9.5 (Ledger policy)**.** *A ledger policy* LedgerPolicy *records what the name hides and how the hidden material is tracked. It must specify whether the ledger is a gap, a composite gap, a tail ledger, a phase ledger, a source ledger, or another derived ledger.*

Definition 9.6 (Paper naming certificate template)**.** *A paper naming certificate for a proposed derived interface N is the five-field template*

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The proposed name N is licensed only if all required fields are supplied.

Definition 9.7 (Lean carrier naming certificate)**.** *At the formal finite-kernel layer, the Lean structure* NameCert(Carrier, Equiv) *is a carrier-and-equivalence certificate over histories. It supplies inhabitedness of the carrier, reflexivity, symmetry, and transitivity for the equivalence relation, and transport of carrier membership along that relation.*

Lean4 (definition) `BEDC.FKernel.NameCert.NameCert`
Lean4 ✓ `BEDC.FKernel.NameCert.nameCert carrier witness`

Theorem 9.8 (Naming certificate classifier transitivity)**.** *For every naming certificate over a history carrier and classifier, the classifier is transitive on finite-kernel histories.*

Proof. The certificate record contains the classifier transitivity field. Projecting that field supplies the required composition of two classifier witnesses. □

Lean4 ✓ `BEDC.FKernel.NameCert.nameCert equiv trans`

Theorem 9.9 (Derived interfaces require certificates)**.** *No interface marked by an upward arrow is licensed without a* NameCert*.*

Proof. This is the formation rule for the paper template: derived names are not object-language primitives. They enter the theory only by the five supplied certificate fields. □

Lean4 ✓ `BEDC.FKernel.NameCert.derived interfaces require certificates`

Principle 9.10 (Limit-like interfaces carry a ledger)**.** *Every certified derived interface carries a ledger policy witness.*

Proof. Case analysis on the semantic naming certificate exposes the source witness in the core certificate. Applying the ledger soundness field to that same source witness gives the required ledger witness. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert ledger policy witness`

Theorem 9.11 (Semantic NameCert presentation weakening)**.** *Let* $S, P, L, P', L' : \mathsf{BHist} \to \mathsf{Prop}$ *and* $C : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*. If*

$$\mathsf{SemanticNameCert}(S, P, L, C)$$

holds, and for every history h *the implications* $P(h) \to P'(h)$ *and* $L(h) \to L'(h)$ *hold, then*

$$\mathsf{SemanticNameCert}(S, P', L', C)$$

holds. The source specification and classifier specification are unchanged.

Proof. By Definition 9.6, the naming template separates the source, pattern, classifier, stability, and ledger roles. The core field of the semantic certificate is the carrier-and-classifier $\mathsf{NameCert}(S, C)$ described by Definition 9.7; it mentions only the source predicate and the classifier relation. Keep this core field. For a source witness $S(h)$, the original pattern soundness field gives $P(h)$, and the pointwise pattern weakening gives $P'(h)$. The same source witness gives $L(h)$ by the original ledger soundness field, and the pointwise ledger weakening gives $L'(h)$. These three fields form the semantic certificate for S, P', L', C. □

Chapter 10

First Derived Interfaces

10.1 Unary repetition

Definition 10.1 (Unary repetition histories)**.** *Unary repetition histories are histories generated from* e *by repeated* E1*-extension.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryRepetitionHistory`

Proposition 10.2 (Unary repetition is closed under continuation)**.** *If h and k are unary repetition histories and* $\mathsf{Cont}(h, k, r)$*, then r is a unary repetition history.*

Lean4 ✓ `BEDC.FKernel.Unary.unary repetition closed under continuation seed`
↪ variant `BEDC.FKernel.Unary.unary repetition closed under continuation`
↪ variant `BEDC.FKernel.Unary.unary append right factor`

Proof. Read $\mathsf{Cont}(h, k, r)$ as append equality and induct on the right unary history k. The empty case returns h. The E0-case is incompatible with unary repetition, and the E1-case follows from the induction hypothesis. □

Corollary 10.3 (Natural-number-like interface seed)**.** *Unary repetition histories with their continuation behavior license a minimal* $\mathsf{Nat}^{\uparrow}$ *interface seed. This is not a full arithmetic theory and does not import primitive natural numbers.*

Lean4 ✓ `BEDC.FKernel.Unary.nat up interface seed`

Corollary 10.4 (Addition-like interface seed)**.** *Continuation of unary repetition histories licenses a minimal* $\mathsf{Add}^{\uparrow}$ *interface seed. Associativity follows from continuation associativity, and the empty history supplies the unit behavior.*

Lean4 ✓ `BEDC.FKernel.Unary.add up interface seed with units`

10.2 Stable transformations

Definition 10.5 (Stable transformation)**.** *A stable transformation between packaged interfaces is a generated transformation behavior that preserves package-sameness and carries a ledger policy.*

Lean4 (definition) `BEDC.FKernel.NameCert.StableTransformation`
Lean4 ✓ `BEDC.FKernel.NameCert.stableTransformation ledger witness`

Theorem 10.6 (Stable transformations descend to packages)**.** *If a generated transformation sends signature-same source histories to package-same target histories, then it descends to a derived transformation interface between package tokens.*

Lean4 ✓ `BEDC.FKernel.NameCert.stableTransformation descent certificate respects`

Proof. The descent condition says that the visible result depends only on the source package, not on the chosen representative source history. Therefore the transformation is licensed at the package level, with its gap ledger recording the source representative information suppressed by the descent. □

Corollary 10.7 (Function-like interfaces are derived)**.** *A* $\mathsf{Func}^{\uparrow}$ *interface is a derived name for stable transformations. It is not an internal primitive.*

Lean4 ✓ `BEDC.FKernel.NameCert.function like interface seed not primitive`

Chapter 11

Thin Seal Interface

Definition 11.1 (Stage interface)**.** *A stage interface* Stage *is a generated visibility level in an iterative observation process. This chapter treats* Stage *as a placeholder for the completion module.*

Lean4 (definition) `BEDC.FKernel.NameCert.StageInterface`

Definition 11.2 (Thread)**.** *A thread* Thread *is a compatible family of stage-level visible data. Compatibility is not developed fully in this finite kernel; it is reserved for the completion module.*

Lean4 (definition) `BEDC.FKernel.NameCert.ThreadFamily`

Definition 11.3 (Seal interface)**.** *A seal event* $\mathsf{Seal}(T, N)$ *licenses a name* N *for a compatible thread* T*, provided a* SealCert *and a ledger policy are supplied.*

Lean4 (definition) `BEDC.FKernel.NameCert.SealEvent`
Lean4 (definition) `BEDC.FKernel.NameCert.SealInterface`
↪ variant `BEDC.FKernel.NameCert.sealInterface thread witness`
Lean4 ✓ `BEDC.FKernel.NameCert.sealInterface certificate witness`

Corollary 11.4 (Limit-like interfaces are derived)**.** *A* $\mathsf{Limit}^{\uparrow}$ *interface is a derived name for seal-certified compatible-thread behavior. It is not primitive in the finite kernel.*

Lean4 ✓ `BEDC.FKernel.NameCert.sealInterface thread carrier witnesses`

Remark 11.5. *The seal layer is intentionally thin in this chapter. Completion and thread sealing are not primitive finite-kernel steps. Real-like and circle-like names may appear in this manuscript only as* NameCert*-governed certificate-obligation schemas, such as* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{S}^{1\uparrow}$*, not as quotient, completion, or topological primitives.*

Chapter 12

External Binary Model

12.1 Purpose of the model

The following model is external. It is a metalanguage interpretation showing that the kernel rules are not empty. It is not part of the object-language.

Definition 12.1 (External model $\mathcal{M}_{\mathsf{bin}}$)**.** *The external model* $\mathcal{M}_{\mathsf{bin}}$ *interprets:*

- *raw marks as two external tokens;*
- *histories as finite binary words;*
- msame *as literal token equality;*
- hsame *as literal word equality;*
- Ext *as appending one token;*
- Cont *as word concatenation;*
- *asking events as externally chosen binary observations;*
- *signatures as finite words of answers;*
- *packages as signature classes;*
- *gaps as inverse images of package tokens;*
- *composite gaps as relational composites of inverse images.*

Lean4 (definition) `BEDC.FKernel.ExternalBinary.Mbin`

Lean4 ✓ `BEDC.FKernel.ExternalBinary.Mbin eq BHist`

↪ variant `BEDC.FKernel.ExternalBinary.external model mbin`

↪ variant `BEDC.FKernel.ExternalBinary.Mbin reuses kernel history`

↪ variant `BEDC.FKernel.ExternalBinary.external model no parallel word input`

↪ variant `BEDC.FKernel.ExternalBinary.external model reuses kernel histories`

↪ variant `BEDC.FKernel.ExternalBinary.external model kernel input reuse`

↪ variant `BEDC.FKernel.ExternalBinary.Mbin cases`

↪ variant `BEDC.FKernel.ExternalBinary.Mbin induction`

↪ variant `BEDC.FKernel.ExternalBinary.external append bit constructor rules`

↪ variant `BEDC.FKernel.ExternalBinary.external append length eq left iff right empty`

↪ variant `BEDC.FKernel.ExternalBinary.external append length eq right iff left empty`

Theorem 12.2 (External append soundness for the finite history kernel)**.** *For the reused history carrier $\mathcal{M}_{\mathsf{bin}} = BHist$, external append has left and right empty units, associativity, right and left cancellation, and additive length.*

```
Lean4 ✓  BEDC.FKernel.ExternalBinary.external model soundness finite kernel
  ↪ variant  BEDC.FKernel.ExternalBinary.external finite kernel soundness
  ↪ variant  BEDC.FKernel.ExternalBinary.external finite kernel soundness full
  ↪ variant  BEDC.FKernel.ExternalBinary.external append unit laws
  ↪ variant  BEDC.FKernel.ExternalBinary.external append empty left
  ↪ variant  BEDC.FKernel.ExternalBinary.external append empty right
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left absorb empty
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left absorb empty iff
  ↪ variant  BEDC.FKernel.ExternalBinary.external word no confusion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append nil result inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append empty result iff
  ↪ variant  BEDC.FKernel.ExternalBinary.external append nil inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append assoc
  ↪ variant  BEDC.FKernel.ExternalBinary.external append assoc four
  ↪ variant  BEDC.FKernel.ExternalBinary.external append assoc five
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right cancel hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right cancel
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left cancel hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append e0 right
  ↪ variant  BEDC.FKernel.ExternalBinary.external append e1 right
  ↪ variant  BEDC.FKernel.ExternalBinary.external append length
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left cancel
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right length zero
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left length zero
  ↪ variant  BEDC.FKernel.ExternalBinary.external length zero iff nil
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right unit unique
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right unit iff
  ↪ variant  BEDC.FKernel.ExternalBinary.external append unit uniqueness pair
```

Proof. The carrier is the kernel history type $BHist$. The append laws are proved by constructor recursion on finite binary words, with the external append operation reusing the continuation append on histories. Empty units and associativity are the unit and associativity laws of that append. Right and left cancellation follow by induction on the common appended context, and the length equation follows by the same recursion that defines binary-word length. □

Theorem 12.3 (External append respects history-sameness)**.** *For all external words a, a', b, b', if* $\mathsf{hsame}(a, a')$ *and* $\mathsf{hsame}(b, b')$*, then*

$$\mathsf{hsame}(append(a, b), append(a', b')).$$

Proof. Unfold external words as $BHist$. The two hsame hypotheses identify the left and right inputs. After those identifications, both appended histories are the same continuation append result, so hsame-reflexivity gives the conclusion. □

```
Lean4 ✓  BEDC.FKernel.ExternalBinary.external append hsame congruence
```

Theorem 12.4 (External append cancellation soundness)**.** *Right and left cancellation for external append are sound for* hsame*: if appending the same right word or the same left word gives internally same results, then the uncancelled words are internally same.*

```
Lean4 ✓  BEDC.FKernel.ExternalBinary.external append cancel hsame pair
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancellation soundness
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel pair
  ↪ variant  BEDC.FKernel.ExternalBinary.external append middle cancel hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append double cancel hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel common context
  ↪ variant  BEDC.FKernel.ExternalBinary.external append middle cancel eq
  ↪ variant  BEDC.FKernel.ExternalBinary.external append right cancel by induction
  ↪ variant  BEDC.FKernel.ExternalBinary.external append left cancel by induction
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel hsame directions by induction
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel hsame nested pair
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel three context hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel hsame triple suffix
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel four context hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append cancel two sided hsame symmetric
```

Proof. The two directions are the right and left hsame-cancellation laws for the external binary append operation. □

Theorem 12.5 (External append bit result inversion)**.** *If an external append result is a zero-bit word, then either the right word is empty and the left word is that zero-bit word, or the right word starts with zero and the tail append is the result tail. The same alternative holds for one-bit words.*

```
Lean4 ✓  BEDC.FKernel.ExternalBinary.external append bit result inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit0 result inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit1 result inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit result tail hsame
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit result cross impossible
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit result cases by right word
  ↪ variant  BEDC.FKernel.ExternalBinary.external append bit result iff
  ↪ variant  BEDC.FKernel.ExternalBinary.external append constructor result inversion
  ↪ variant  BEDC.FKernel.ExternalBinary.external append length zero pair
```

Proof. Case-analyze the right appended word. The empty case returns the left word. Matching bit constructors expose the tail equality by constructor injectivity, and opposite bit constructors are impossible by constructor disjointness. □

Corollary 12.6 (External append bit tail equality)**.** *If* $append(a, b)$ *is a zero-bit result* $0 :: r$ *and* b *is a zero-bit word* $0 :: b_0$*, then* $append(a, b_0) = r$*. If* $append(a, b)$ *is a one-bit result* $1 :: r$ *and* b *is a one-bit word* $1 :: b_1$*, then* $append(a, b_1) = r$*.*

```
Lean4 ✓  BEDC.FKernel.ExternalBinary.external append bit result tail eq pair
```

Proof. In each bit case, substitute the displayed shape of b into the append equality. Constructor injectivity for the matching bit exposes equality of the tails. □

Chapter 13

Finite Theorem Manifest

This chapter lists the finite kernel theorem targets. The listed names are stable labels for theorem obligations inside the kernel specification; they are not object-language names and do not presuppose any implementation layer.

Target	Content
`msame_refl`	Reflexivity of mark-sameness.
`msame_symm`	Symmetry of mark-sameness.
`msame_trans`	Transitivity of mark-sameness.
`mark_no_confusion`	Cross-mark sameness is not generated.
`hsame_refl`	Reflexivity of history-sameness.
`hsame_symm`	Symmetry of history-sameness.
`hsame_trans`	Transitivity of history-sameness.
`history_no_confusion`	Cross-constructor history-sameness is not generated.
`ext_det`	Relational extension is deterministic up to history-sameness.
`cont_det`	Relational continuation is deterministic up to history-sameness.
`cont_left_unit`	Empty history is a left unit for continuation.
`cont_right_unit`	Empty history is a right unit for continuation.
`cont_assoc`	Continuation is associative up to history-sameness.
`sig_det`	Signature generation is deterministic under an asking policy.
`sameSig_equiv`	Signature-sameness is an equivalence under policy assumptions.
`pkg_by_sig`	Packages classify signatures under package policy.
`gap_coverage`	Every admitted source history lies in some gap.
`gap_separation`	Overlap of gaps implies package-sameness.
`compGap_coverage`	Composite gaps have coverage when component gaps do.
`unary_cont_closed`	Unary repetition histories are closed under continuation.
`add_like_assoc`	The addition-like interface seed inherits continuation associativity.

Chapter 14

Boundary of the Finite Kernel

The finite kernel covers the chain

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{Gap} \to \mathsf{NameCert}.$$

It is intentionally bounded: four families of derived constructions sit outside the primitive finite-kernel chain. They may be named in this manuscript only as typed certificate-obligation schemas when their classifiers, stability data, and ledgers are supplied.

1. completion schemas for seal-certified threads;
2. the full arithmetic module for induction, recursion, multiplication, and order;
3. phase-name schemas for circle-like certificates;
4. the fold module for normal representatives and fiber ledgers beyond signature packages.

None of these families is a primitive of the finite kernel; each is admitted only through its own naming certificate, with the certificate's stability fields supplied by the finite-kernel chain.

Summary

This manuscript gives a relation-based finite-kernel target for the Binary Emission Discovery Calculus. It begins with two raw marks and generated histories; defines internal mark- and history-sameness; relationizes extension and continuation; treats probes as asking events; generates bundles and signatures internally; packages signatures under policies; records gaps; composes ledgers; and licenses derived interfaces only through typed naming certificates. Its primitive scope is finite emission, probing, packaging, and ledger composition. Completion and fold constructions remain outside that primitive chain, while real-like and circle-like appearances are typed naming-certificate obligation schemas, not quotient, completion, or topological primitives.

Chapter 15

The Signature Package Instance

The hardening layer introduced PackagePolicy as an abstract interface. This chapter constructs the first concrete instance: package tokens generated directly from answer signatures.

15.1 Signature tokens

Definition 15.1 (Signature token generator)**.** *Fix a generated probe bundle* Π*. For every generated answer history* s*, introduce a formal package token*

$$\mathsf{Tok}_{\Pi}(s).$$

This notation is a naming convention for the package-token event associated with the answer history s*. It is not a host function in the object theory.*

Lean4 (definition) `BEDC.FKernel.Package.SignatureTokenGenerator`
Lean4 ✓ `BEDC.FKernel.Package.SignatureTokenGenerator iff exists`

Definition 15.2 (Signature package relation)**.** *This concrete instance represents visible signature tokens by answer histories: its package-token carrier is* $BHist$*. The notation* $\mathsf{Tok}_{\Pi}(s)$ *names the visible token represented by a history, and token introduction between a signature history* s *and a visible token* p *is the relation* $\mathsf{hsame}(s, p)$*.*

The concrete signature package relation $\mathsf{Pkg}^{\mathrm{sig}}$ *is generated by the rule*

$$\mathsf{Pkg}^{\mathrm{sig}}(\Pi, s, \mathsf{Tok}_{\Pi}(s)).$$

Package-sameness is generated by the rule

$$\mathsf{hsame}(s, t) \quad \Longrightarrow \quad \mathsf{psame}(\mathsf{Tok}_{\Pi}(s), \mathsf{Tok}_{\Pi}(t)).$$

No additional package-sameness constructors are included.

Lean4 ✓ `BEDC.FKernel.Package.signature package relation generated only`

Lemma 15.3 (Signature package existence)**.** *For every generated answer history* s*, there exists a package token* p *such that*

$$\mathsf{Pkg}^{\mathrm{sig}}(\Pi, s, p).$$

Lean4 ✓ `BEDC.FKernel.Package.signature package existence`

Proof. Take $p = \mathsf{Tok}_\Pi(s)$ and apply the signature package rule. □

Lemma 15.4 (Signature package extensionality)**.** *If* $\mathsf{hsame}(s, t)$*, token introduction relates* s *to* p*, and token introduction relates* t *to* q*, then* $\mathsf{psame}(p, q)$*.*

Proof. The two token-introduction premises and the history-sameness premise are exactly the three arguments of the displayed package-sameness constructor, so that constructor yields $\mathsf{psame}(p, q)$ directly. □

Lemma 15.5 (Signature package grounding)**.** *If* $\mathsf{Pkg}^{\mathrm{sig}}(\Pi, s, p)$*, then* p *is generated from the signature history* s *by the displayed token rule.*

Lean4 ✓ `BEDC.FKernel.Package.signature package grounding`

Proof. By closed generation for $\mathsf{Pkg}^{\mathrm{sig}}$, every package derivation is generated from a token $\mathsf{Tok}_\Pi(s)$. No hidden token constructor is present. □

Lemma 15.6 (Signature package sameness reflects history sameness)**.** *For the concrete signature package instance, where package tokens are represented by histories, if* $\mathsf{psame}(p, q)$*, then* $\mathsf{hsame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Package.signature package psame implies hsame`

Proof. Invert the one package-sameness constructor. The displayed token-introduction witnesses identify p and q with the introducing signature histories, and transitivity of hsame composes those witnesses with the constructor proof. □

Theorem 15.7 (Concrete PackagePolicy for signatures)**.** *The relation* $\mathsf{Pkg}^{\mathrm{sig}}$*, together with the token-sameness rule above, satisfies the package existence, package extensionality, and package grounding requirements of* PackagePolicy*.*

Lean4 ✓ `BEDC.FKernel.Package.signature package policy`
↪ variant `BEDC.FKernel.Package.signature package token exists`

Proof. Existence, extensionality, and grounding are exactly the three preceding lemmas. □

Corollary 15.8 (The first concrete package interface)**.** *Every generated probe bundle* Π *admits a concrete package interface whose visible tokens are generated by answer signatures.*

Lean4 ✓ `BEDC.FKernel.Package.first concrete package interface`

Chapter 16

The Concrete Signature Gap Policy

The signature package instance provides visible tokens. We now define the concrete gap ledger behind those tokens.

Definition 16.1 (Signature gap membership)**.** *Fix a probe bundle* Π*, a source domain specification* D*, and the signature package instance of chapter 15, whose visible package relation is* $\mathsf{Pkg}^{\mathrm{sig}}$*. A history* h *lies in the signature gap of a package token* p *when there exist an answer history* s *and evidence ledger* Δ *such that*

$$\mathsf{InDom}(D, h), \qquad \mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{TokIntro}(\Pi, s, p).$$

We write this as

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Lean4 (definition) `BEDC.FKernel.Gap.GapMembership`
Lean4 (definition) `BEDC.FKernel.Gap.SignatureGapMembership`
↪ variant `BEDC.FKernel.Gap.GapMembership iff InGapSig`
↪ variant `BEDC.FKernel.Gap.GapMembership witness pair`
Lean4 ✓ `BEDC.FKernel.Gap.inGapSig iff witnesses`
↪ variant `BEDC.FKernel.Gap.signature gap membership iff`
↪ variant `BEDC.FKernel.Gap.inGapSig intro`
↪ variant `BEDC.FKernel.Gap.inGapSig intro from signature witness`
↪ variant `BEDC.FKernel.Gap.inGapSig elim`
↪ variant `BEDC.FKernel.Gap.inGapSig domain and signature witness`
↪ variant `BEDC.FKernel.Gap.inGapSig domain signature token witnesses`
↪ variant `BEDC.FKernel.Gap.inGapSig domain signature token witness pair`
↪ variant `BEDC.FKernel.Gap.signature gap membership`
Lean4 ✓ `BEDC.FKernel.Gap.inGapSig signature and token witness`
↪ variant `BEDC.FKernel.Gap.inGapSig token witness`
↪ variant `BEDC.FKernel.Gap.inGapSig witnesses`

Lemma 16.2 (Concrete gap coverage)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and package-wide token totality: for every answer history* s *there exists a package token* p *with* $\mathsf{TokIntro}(\Pi, s, p)$*. For every generated history* h *with* $\mathsf{InDom}(D, h)$*, there exists a package token* p *such that*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h).$$

Lean4 ✓ `BEDC.FKernel.Gap.concrete gap coverage`

Proof. By totality in $\mathsf{AskPolicy}(\Pi, D)$, the bundle Π produces some signature event

$$\mathsf{Sig}(\Pi, h, s, \Delta).$$

By token existence, there is a package token p such that

$$\mathsf{TokIntro}(\Pi, s, p).$$

Therefore $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ holds by definition. □

Lemma 16.3 (Concrete gap separation)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. Suppose*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, h).$$

Then $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap separation concrete`

Proof. By the two gap memberships, there are signatures s, t and ledgers Δ, Θ such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \quad \mathsf{TokIntro}(\Pi, s, p),$$

$$\mathsf{Sig}(\Pi, h, t, \Theta), \quad \mathsf{TokIntro}(\Pi, t, q).$$

By signature determinacy under $\mathsf{AskPolicy}(\Pi, D)$, $\mathsf{hsame}(s, t)$. By signature package extensionality, $\mathsf{psame}(p, q)$. □

Theorem 16.4 (Concrete GapPol for signature packages)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackageTokenPolicy}(\Pi)$*, and package-wide token totality: for every answer history* s*, there is a visible package* p *with* $\mathsf{TokIntro}(\Pi, s, p)$*. Then the package relation* $\mathsf{Pkg}^{\mathrm{sig}}$ *together with* $\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h)$ *satisfies coverage and separation for the signature package instance.*

Lean4 ✓ `BEDC.FKernel.Gap.concrete gap policy`
↪ variant `BEDC.FKernel.Gap.gapPolicy coverage signature witness`

Proof. Coverage is supplied by signature generation under $\mathsf{AskPolicy}(\Pi, D)$ together with token existence. Separation is supplied by signature determinacy under $\mathsf{AskPolicy}(\Pi, D)$ and package soundness from $\mathsf{PackageTokenPolicy}(\Pi)$. □

Corollary 16.5 (First concrete globalization ledger)**.** *The pair*

$$(\mathsf{Pkg}^{\mathrm{sig}}, \mathsf{InGap}^{\mathrm{sig}})$$

forms the first concrete finite-kernel globalization interface for the domain D *and probe bundle* Π*, conditional on* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackageTokenPolicy}(\Pi)$*, and package-wide token totality: for every answer history* s*, there exists a visible package* p *with* $\mathsf{TokIntro}(\Pi, s, p)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.first concrete globalization ledger`

Chapter 17

The First Concrete Globalize Instance

This chapter packages the two preceding chapters into a single concrete globalization instance.

Definition 17.1 (Signature globalization instance)**.** *Given a source domain specification D, a generated probe bundle Π, and* $\mathsf{AskPolicy}(\Pi, D)$*, the signature globalization instance is the triple*

$$\mathsf{Glob}^{\mathrm{sig}}(\Pi, D) := (\mathsf{Pkg}^{\mathrm{sig}}, \mathsf{psame}, \mathsf{InGap}^{\mathrm{sig}}).$$

It is read as a visible package layer plus a provenance ledger layer.

Lean4 (definition) `BEDC.FKernel.Gap.signature globalization instance`

Theorem 17.2 (Concrete globalization soundness)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and* $\mathsf{PackageTokenPolicy}(\Pi)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k), \qquad \mathsf{sameSig}(\Pi, h, k),$$

then $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.concrete globalize soundness from sameSig`
↪ variant `BEDC.FKernel.Gap.internalized globalize soundness for witnesses`

Proof. The two gap memberships provide introduced visible tokens for generated signatures of h and k. By $\mathsf{sameSig}(\Pi, h, k)$, there exist generated answer histories s, t and evidence ledgers such that

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta), \qquad \mathsf{hsame}(s, t).$$

The determinacy field of $\mathsf{AskPolicy}(\Pi, D)$ aligns these signatures with the signatures carried by the two gap witnesses. Package soundness from $\mathsf{PackageTokenPolicy}(\Pi)$ gives $\mathsf{psame}(p, q)$. □

Theorem 17.3 (Concrete globalization completeness)**.** *Assume* $\mathsf{PackageTokenPolicy}(\Pi)$*. If*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h), \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k), \qquad \mathsf{psame}(p, q),$$

then $\mathsf{sameSig}(\Pi, h, k)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.concrete globalize completeness sameSig`
↪ variant `BEDC.FKernel.Gap.concrete globalization completeness`

Proof. Unfold the concrete gap memberships. They provide signatures s, t such that h generates s, k generates t, and p, q are introduced from those signatures. By the reflection field of $\mathsf{PackageTokenPolicy}(\Pi)$, package-sameness of the visible tokens is grounded in $\mathsf{hsame}(s, t)$. Hence $\mathsf{sameSig}(\Pi, h, k)$. □

Corollary 17.4 (Globalize equals signature classification in the concrete instance)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$ *and* $\mathsf{PackageTokenPolicy}(\Pi)$*. For introduced tokens with*

$$\mathsf{InGap}^{\mathrm{sig}}(\Pi, D, p, h) \qquad \textit{and} \qquad \mathsf{InGap}^{\mathrm{sig}}(\Pi, D, q, k),$$

the concrete instance satisfies

$$\mathsf{psame}(p, q) \iff \mathsf{sameSig}(\Pi, h, k).$$

Lean4 ✓ `BEDC.FKernel.Gap.globalize equals signature classification concrete`

Remark 17.5 (Why this matters)**.** *The surrounding theory states that Globalize is classification by signatures. This chapter gives the first concrete finite-kernel inhabitant of that claim. It does not yet build real-like, circle-like, or fold-like interfaces; it only makes the finite signature globalization stand.*

Part II

Proof Sprint

Chapter 18

Finite Kernel Proof Target

18.1 Purpose and scope

This text is a proof-sprint refinement of the finite kernel. Its primitive target excludes completion, quotient, topological, and fold constructions. Real-like and circle-like names may still appear as typed certificate-obligation schemas governed by $\mathsf{NameCert}$, not as completed traditional objects.

The finite kernel target is:

$$\mathsf{Mark} \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{Gap} \to \mathsf{NameCert}.$$

The theory-level constraints are:

(i) raw marks carry no numeric interpretation;

(ii) extension, continuation, asking, signature generation, and packaging are treated relationally;

(iii) internal sameness is not host equality;

(iv) every compression must produce a gap ledger;

(v) every derived interface must be licensed by a naming certificate.

Principle 18.1 (Finite-kernel freezing)**.** *The first proof target isolates the finite kernel before completion-like or geometric constructions. A derived interface may be mentioned only as a typed obligation schema with an explicit naming certificate, not as a completed traditional object.*

18.2 Marks and mark-sameness

Definition 18.2 (Raw marks)**.** *The kernel begins with two raw marks:*

$$\mathsf{b0}, \qquad \mathsf{b1}.$$

They are not numbers, truth values, booleans, or elements of a set in the object theory. They are only two distinguishable emission results.

Definition 18.3 (Internal mark-sameness)**.** *The internal sameness judgment on raw marks is generated only by:*

$$\mathsf{msame}(\mathsf{b0}, \mathsf{b0}), \qquad \mathsf{msame}(\mathsf{b1}, \mathsf{b1}).$$

No rule generates $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *or* $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$.

Lemma 18.4 (Mark-sameness is reflexive)**.** *For every generated raw mark m,* $\mathsf{msame}(m, m)$*.*

Proof. There are only two generated raw marks. If $m = \mathsf{b0}$, use the first generator. If $m = \mathsf{b1}$, use the second generator. This case analysis belongs to the generation clauses for raw marks, not to an external numerical interpretation. □

Lemma 18.5 (Mark-sameness is symmetric)**.** *If* $\mathsf{msame}(m, n)$*, then* $\mathsf{msame}(n, m)$*.*

Proof. The only generators of msame are the two diagonal judgments. Reversing either generator yields the same generator. □

Lemma 18.6 (Mark-sameness is transitive)**.** *If* $\mathsf{msame}(m, n)$ *and* $\mathsf{msame}(n, r)$*, then* $\mathsf{msame}(m, r)$*.*

Proof. By closed generation of msame, both premises must be diagonal at the same raw mark. Hence the conclusion is the corresponding diagonal generator. □

Theorem 18.7 (Mark no-confusion)**.** *The judgments* $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ *and* $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$ *are not generated.*

Proof. The closed generation clause for msame lists only the two diagonal generators. There is no cross-mark generator and no rule that can produce a cross-mark judgment from diagonal ones. □

18.3 Histories and history-sameness

Definition 18.8 (Emission histories)**.** *Emission histories are generated by:*

$$\mathsf{e} : \mathsf{Hist}, \qquad h : \mathsf{Hist} \Rightarrow \mathsf{E0}(h) : \mathsf{Hist}, \qquad h : \mathsf{Hist} \Rightarrow \mathsf{E1}(h) : \mathsf{Hist}.$$

The symbol e *denotes the empty emission history. The constructors* $\mathsf{E0}$ *and* $\mathsf{E1}$ *record one further emission of* $\mathsf{b0}$ *or* $\mathsf{b1}$*, respectively.*

Definition 18.9 (Internal history-sameness)**.** *The judgment* $\mathsf{hsame}(h, k)$ *is generated by:*

$$\mathsf{hsame}(\mathsf{e}, \mathsf{e}),$$

$$\mathsf{hsame}(h, k) \Rightarrow \mathsf{hsame}(\mathsf{E0}(h), \mathsf{E0}(k)),$$

$$\mathsf{hsame}(h, k) \Rightarrow \mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k)).$$

There are no cross-constructor generators.

Theorem 18.10 (History-sameness is an equivalence)**.** *The judgment* hsame *is reflexive, symmetric, and transitive on generated histories.*

Proof. Reflexivity is by induction on generated histories. The empty history uses $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. The $\mathsf{E0}$ and $\mathsf{E1}$ cases use the corresponding closure rule.

Symmetry is by induction on a derivation of $\mathsf{hsame}(h, k)$. The empty case is immediate. The constructor cases reduce to the induction hypothesis.

Transitivity is by induction on the first sameness derivation and inversion on the second. Closed generation prevents cross-constructor cases. Thus the only non-empty cases are matching $\mathsf{E0}$-with-$\mathsf{E0}$ or $\mathsf{E1}$-with-$\mathsf{E1}$, where the induction hypothesis applies to the subhistories. □

Theorem 18.11 (History no-confusion)**.** *The following judgments are not generated:*

$$\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(h)), \quad \mathsf{hsame}(\mathsf{e}, \mathsf{E1}(h)), \quad \mathsf{hsame}(\mathsf{E0}(h), \mathsf{E1}(k)),$$

with symmetric variants.

Lean4 ✓ `BEDC.FKernel.Hist.proof sprint history no confusion complete`

Proof. Each would require a generator for a cross-constructor history-sameness judgment. The closed generation clauses for hsame contain no such generator. □

18.4 Relational extension

Definition 18.12 (Extension relation)**.** *Extension is a relation, not a function. The judgment*

$$\mathsf{Ext}(m, h, r)$$

means that extending history h by raw mark m may generate result history r. It is generated by:

$$\mathsf{Ext}(\mathsf{b0}, h, \mathsf{E0}(h)), \qquad \mathsf{Ext}(\mathsf{b1}, h, \mathsf{E1}(h)).$$

Lean4 (definition) `BEDC.FKernel.Ext.Ext`

Theorem 18.13 (Extension determinacy)**.** *If* $\mathsf{Ext}(m, h, r)$ *and* $\mathsf{Ext}(m, h, r')$*, then* $\mathsf{hsame}(r, r')$*.*

Proof. By cases on the generated extension judgment. If $m = \mathsf{b0}$, both results are internally the same as $\mathsf{E0}(h)$. If $m = \mathsf{b1}$, both results are internally the same as $\mathsf{E1}(h)$. The claim follows by reflexivity of hsame. □

Theorem 18.14 (Extension respects internal sameness)**.** *If* $\mathsf{msame}(m, m')$*,* $\mathsf{hsame}(h, h')$*,* $\mathsf{Ext}(m, h, r)$*, and* $\mathsf{Ext}(m', h', r')$*, then* $\mathsf{hsame}(r, r')$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext respects sameness`

Proof. By closed generation of msame, the two marks are internally the same diagonal mark. If that mark is b0, the extension results have the form $\mathsf{E0}(h)$ and $\mathsf{E0}(h')$; the conclusion follows from $\mathsf{hsame}(h, h')$ and the E0-closure rule. The b1 case is analogous. □

18.5 Relational continuation

Definition 18.15 (Continuation relation)**.** *Continuation is a relation, not a function. The judgment*

$$\mathsf{Cont}(h, k, r)$$

means that continuing history h by history k may generate result r. It is generated by:

$$\mathsf{Cont}(h, \mathsf{e}, h),$$

$$\mathsf{Cont}(h, k, r) \Rightarrow \mathsf{Cont}(h, \mathsf{E0}(k), \mathsf{E0}(r)),$$

$$\mathsf{Cont}(h, k, r) \Rightarrow \mathsf{Cont}(h, \mathsf{E1}(k), \mathsf{E1}(r)).$$

Theorem 18.16 (Continuation determinacy)**.** *If* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{Cont}(h, k, r')$*, then* $\mathsf{hsame}(r, r')$*.*

↪variant BEDC.FKernel.Cont.cont deterministic

Proof. Induct on the right-history structure exposed by the first continuation derivation. In the empty case, both results are internally the same as h. In the E0-case, closed generation forces the second derivation to be an E0-case over the same right-history predecessor, and the induction hypothesis applies to the subresults. The E1-case is analogous. □

Theorem 18.17 (Continuation left unit)**.** *For every generated history k,* $\mathsf{Cont}(\mathsf{e}, k, k)$.

Lean4 ✓ BEDC.FKernel.Cont.cont left unit induction spine

Proof. Induct on k. The empty case is the base generator $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. The E0 and E1 cases apply the corresponding right-recursive continuation step to the induction hypothesis. □

Theorem 18.18 (Continuation right unit)**.** *For every generated history h,* $\mathsf{Cont}(h, \mathsf{e}, h)$.

Lean4 ✓ BEDC.FKernel.Cont.cont right unit

Proof. This is the base generator of Cont. □

Theorem 18.19 (Append right nonempty self contradiction)**.** *For histories $h, k : \mathsf{BHist}$, a nonempty right factor cannot leave the append result internally the same as the left factor:*

$$(\mathsf{hsame}(k, \mathsf{e}) \to \bot) \Rightarrow \mathsf{hsame}(\mathsf{append}(h, k), h) \Rightarrow \bot.$$

Proof. Compare $\mathsf{append}(h, k)$ with $\mathsf{append}(h, \mathsf{e})$ using the displayed self-sameness and the right unit for append. Left cancellation forces $k\mathsf{hsamee}$, contradicting the nonempty right-factor hypothesis. □

Lean4 ✓ BEDC.FKernel.Cont.append right nonempty self absurd

Theorem 18.20 (Append left nonempty self contradiction)**.** *For histories $h, k : \mathsf{BHist}$, a nonempty left factor cannot leave the append result internally the same as the right factor:*

$$(\mathsf{hsame}(h, \mathsf{e}) \to \bot) \Rightarrow \mathsf{hsame}(\mathsf{append}(h, k), k) \Rightarrow \bot.$$

Proof. Compare $\mathsf{append}(h, k)$ with $\mathsf{append}(\mathsf{e}, k)$ using the displayed self-sameness and the left unit for append. Right cancellation forces $h\mathsf{hsamee}$, contradicting the nonempty left-factor hypothesis. □

Lean4 ✓ BEDC.FKernel.Cont.append left nonempty self absurd

Theorem 18.21 (Continuation associativity, relational form)**.** *If* $\mathsf{Cont}(a, b, ab)$ *and* $\mathsf{Cont}(b, c, bc)$*, then there exists a generated history abc such that*

$$\mathsf{Cont}(ab, c, abc) \qquad \textit{and} \qquad \mathsf{Cont}(a, bc, abc).$$

Lean4 ✓ BEDC.FKernel.Cont.cont assoc exists

Proof. Use the checked right-recursive append characterization of continuation and the append associativity spine. Equivalently, induct on the right continuation by c: the empty case takes $abc = ab$, and the E0 and E1 cases lift the witness with the matching right-recursive continuation step. □

Theorem 18.22 (Continuation suffix iff)**.** *For histories a, b, f, p, continuation from a by the suffixed history* $\mathsf{append}(f, p)$ *to the suffixed result* $\mathsf{append}(b, p)$ *is equivalent to the unsuffixed continuation from a by f to b:*

$$\mathsf{Cont}(a, \mathsf{append}(f, p), \mathsf{append}(b, p)) \iff \mathsf{Cont}(a, f, b).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont suffix iff`

Proof. For the forward direction, rewrite the suffixed continuation through associativity so that both sides share the right suffix p, then cancel that common suffix. For the reverse direction, expand the unsuffixed continuation and use associativity of history append to rebuild the suffixed continuation. □

Theorem 18.23 (Continuation common-context equivalence up to history-sameness)**.** *For histories L, R, P, Q, a, b, f : BHist, if $L\mathsf{hsame}R$ and $P\mathsf{hsame}Q$, then adding those matched contexts to the continuation source, tail, and result preserves and reflects the core continuation:*

$$L\mathsf{hsame}R \Rightarrow P\mathsf{hsame}Q \Rightarrow \big(\mathsf{Cont}(\mathsf{append}(L, a), \mathsf{append}(f, P), \mathsf{append}(\mathsf{append}(R, b), Q)) \iff \mathsf{Cont}(a, f, b)\big).$$

Proof. Transport the two context comparisons so the left and right contexts match definitionally. The forward direction first removes the common right suffix by the continuation suffix equivalence, then removes the common left prefix by the continuation prefix equivalence. The reverse direction applies the same two equivalences in the opposite order. □

Lean4 ✓ `BEDC.FKernel.Cont.cont hsame common context iff`

18.6 Asking events and bundles

Definition 18.24 (Asking event)**.** *An asking event is a judgment*

$$\mathsf{Ask}(\pi, h, m, \delta),$$

where π is a probe-name, h is a generated history, m is a raw mark emitted as answer, and δ is evidence produced by the asking event. This is not a function application.

Lean4 (definition) `BEDC.FKernel.Ask.Ask`

Definition 18.25 (Probe bundles)**.** *Probe bundles are generated by:*

$$\mathsf{Bnil}, \qquad \Pi \mapsto \mathsf{Bcons}(\pi, \Pi).$$

They are internal generated records of probe-names, not imported finite lists.

Definition 18.26 (Ask policy)**.** *An* $\mathsf{AskPolicy}(\Pi, D)$ *occurrence is paper notation for applying the checked* $\mathsf{AskPolicy}(D)$ *certificate at bundle Π. The certificate gives enough asking events for probe names used by signature generation, with Lean fields quantified over all probe names, and records determinacy up to* msame *plus respect for* hsame.

Lean4 ✓ `BEDC.FKernel.Ask.askPolicy event determinacy respects fields`

18.7 Signatures and signature-sameness

Definition 18.27 (Generated signature)**.** *The signature judgment*

$$\mathsf{Sig}(\Pi, h, s, \Delta)$$

means that asking bundle Π *against history* h *generates answer history* s *with evidence ledger* Δ*. It is generated by an empty rule for* Bnil *and a cons rule using one* Ask*-event followed by the signature of the remaining bundle.*

Lean4 (definition) `BEDC.FKernel.Sig.SigRel`

Theorem 18.28 (Empty signature has no visible output)**.** *For the empty probe bundle, generated signatures cannot have answer history* $\mathsf{E0}(p)$ *or* $\mathsf{E1}(p)$ *for any* $p : \mathsf{BHist}$*.*

Proof. Invert the empty-bundle signature derivation. Its only generated result is the empty history, and BEDC history constructor no-confusion excludes both visible constructors from being empty. □

Lean4 ✓ `BEDC.FKernel.Sig.sig empty visible absurd`

Definition 18.29 (Signature-sameness)**.** *For a bundle* Π*, histories* h, k *are signature-same, written* $\mathsf{sameSig}(\Pi, h, k)$*, when there exist generated signatures*

$$\mathsf{Sig}(\Pi, h, s, \Delta), \qquad \mathsf{Sig}(\Pi, k, t, \Theta),$$

with $\mathsf{hsame}(s, t)$*.*

Theorem 18.30 (Signature determinacy under policy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $h \in D$*,* $\mathsf{Sig}(\Pi, h, s, \Delta)$*, and* $\mathsf{Sig}(\Pi, h, t, \Theta)$*, then* $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.proof sprint signature determinacy under policy`

Proof. By induction on the generated bundle. The empty bundle produces empty signatures. For a cons bundle, determinacy of the first asking event gives msame of the first answer marks, and the induction hypothesis gives hsame of the remaining answer histories. The history constructor closure rules then yield $\mathsf{hsame}(s, t)$. □

Theorem 18.31 (Signature-sameness is an equivalence under policy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. On histories in* D*,* $\mathsf{sameSig}(\Pi, -, -)$ *is reflexive, symmetric, and transitive.*

Lean4 ✓ `BEDC.FKernel.Sig.proof sprint signature sameness equivalence under policy`

Proof. Reflexivity uses totality of signature generation and reflexivity of hsame. Symmetry uses symmetry of hsame. For transitivity, suppose h is signature-same to k, and k is signature-same to l. The two signatures generated for k are internally the same by signature determinacy. Transitivity of hsame then composes the answer-history sameness witnesses. □

18.8 Packaging and gap policies

Definition 18.32 (Package token)**.** *A package token is a visible token licensed for a signature. The judgment*

$$\mathsf{Pkg}(\Pi, s, p)$$

means that signature s under bundle Π licenses package token p.

Definition 18.33 (Package policy)**.** *A* $\mathsf{PackagePolicy}(\Pi)$ *specifies package existence, package extensionality over* hsame*-same signatures, and package grounding: packages are grounded in signature classes, not in arbitrary external choices.*

Definition 18.34 (Gap membership)**.** *The judgment*

$$\mathsf{Gap}(\Pi, D, p, h)$$

means that source history $h \in D$ is recorded in the ledger of package token p. A gap policy relates this membership to generated signatures and package tokens.

Definition 18.35 (Gap policy)**.** *A* $\mathsf{GapPolicy}(\Pi, D)$ *states that h belongs to the gap of p exactly when h generates some signature s licensed by package p, up to the internal sameness policies specified for histories and packages.*

Theorem 18.36 (Gap coverage)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*, and* $\mathsf{GapPolicy}(\Pi, D)$*. Every source history $h \in D$ belongs to the gap of some package token.*

Lean4 ✓ `BEDC.FKernel.Gap.gapPolicy requires ledger witness`
↪ variant `BEDC.FKernel.Gap.gap ledgers not optional`
↪ variant `BEDC.FKernel.Gap.gapPolicy coverage full witnesses`

Proof. By the ask policy, h generates a signature. By the package policy, that signature licenses a package token. By the gap policy, h belongs to the ledger of that package. □

Theorem 18.37 (Gap separation)**.** *Assume the same policies. If one source history belongs to the gaps of two package tokens, then the two package tokens are internally package-same.*

Lean4 ✓ `BEDC.FKernel.Gap.proof sprint gap separation`

Proof. Membership in each gap gives two generated signatures for the same source history. Signature determinacy gives hsame of those signatures. Package extensionality then gives internal package-sameness of the two package tokens. □

Corollary 18.38 (Globalization classifies by signatures)**.** *Under the three policies above, visible package tokens classify source histories by generated signatures, while gap ledgers record the histories compressed into each visible package.*

Lean4 ✓ `BEDC.FKernel.Gap.proof sprint globalization classifies by signatures`

18.9 Composite gap ledgers

Definition 18.39 (Composite gap)**.** *Suppose a first visible interface compresses sources into intermediate tokens and a second visible interface compresses intermediate tokens into final tokens. The composite gap of a final token contains a source exactly when there exists an intermediate token connecting the two gap memberships. Symbolically:*

$$\mathsf{CompGap}_{D\circ C}(z,x) \quad \textit{iff} \quad \exists y,\ \mathsf{Gap}_D(z,y) \textit{ and } \mathsf{Gap}_C(y,x).$$

This is a meta-kernel ledger law, not an object-language primitive.

Theorem 18.40 (Composite gap coverage)**.** *If the first and second gap interfaces satisfy coverage, then the composite gap satisfies coverage.*

Lean4 ✓ `BEDC.FKernel.Gap.compGap coverage from sourced layers`

Proof. Given a source, use coverage of the first interface to obtain an intermediate token. Then use coverage of the second interface to obtain a final token. The intermediate token witnesses composite membership. □

18.10 Name certificates

Definition 18.41 (Typed naming certificate)**.** *A paper name-certificate template is a record*

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}, \mathsf{PatternSpec}, \mathsf{ClassifierSpec}, \mathsf{StabilityCert}, \mathsf{LedgerPolicy}).$$

It licenses a derived name N. A name without such a certificate is not admitted as a derived interface.

Definition 18.42 (Stability certificate modes)**.** *The stability component may include closure, reuse, descent, composition, or seal certificates:*

$$\mathsf{ClosureCert}, \quad \mathsf{ReuseCert}, \quad \mathsf{DescentCert}, \quad \mathsf{CompositionCert}, \quad \mathsf{SealCert}.$$

Different derived interfaces require different stability modes.

Theorem 18.43 (Derived interfaces require certificates)**.** *No derived interface is licensed unless a corresponding* NameCert *is supplied.*

Proof. This is the admission protocol for derived names. A bare traditional name is not a kernel object. It becomes an admitted derived interface only through a certificate specifying source, pattern, classifier, stability, and ledger policy. □

18.11 First derived interface seeds

Definition 18.44 (Unary history)**.** *A unary history is a generated history built from* e *using only* E1*. This is not yet a natural number; it is only a repetition history.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryHistory`

Theorem 18.45 (Nonempty unary histories expose an E_1-tail)**.** *For every history h : BHist, if h is unary and $h$hsamee would be contradictory, then $h = \mathsf{E1}(t)$ for some unary tail t.*

Proof. Split the unary-history cases into the empty case and the visible E1-tail case. The empty branch contradicts the displayed nonempty hypothesis, while the visible branch gives the required tail witness. □

Lean4 ✓ `BEDC.FKernel.Unary.unary history nonempty e1 tail`

Corollary 18.46 (Natural-number-like interface seed)**.** *The family of unary histories licenses a minimal* $\mathsf{Nat}^{\uparrow}$ *seed once supplied with a name certificate whose pattern is repeated one-emission and whose stability certifies closure under one-extension.*

Corollary 18.47 (Addition-like interface seed)**.** *Continuation of unary histories licenses a minimal* $\mathsf{Add}^{\uparrow}$ *seed once supplied with a name certificate whose stability certifies unit and associativity behavior of continuation.*

Remark 18.48. *These are only interface seeds. They do not yet amount to a full arithmetic reconstruction.*

18.12 Thin completion interface

Definition 18.49 (Thin seal interface)**.** *A seal-like interface records the shape of an eventual completion module: staged data, restriction events, compatible threads, a seal event, and a tail ledger. It is not part of the finite kernel proof target.*

Remark 18.50. *Limit-like and fold-like constructions remain outside the primitive finite-kernel proof target. Real-like and circle-like names may appear only as* NameCert*-governed obligation schemas with supplied classifiers and ledgers.*

18.13 External semantic model

Definition 18.51 (Binary external model)**.** *An external model* $\mathcal{M}_{\mathsf{bin}}$ *interprets raw marks as two external tokens, histories as finite binary words,* msame *and* hsame *as literal token and word sameness,* Ext *and* Cont *as ordinary word extension and concatenation, asking events as binary-valued external relations, signatures as generated answer words, packages as signature classes, gaps as inverse images, and composite gaps as relational composition of inverse images.*

Lean4 ✓ `BEDC.FKernel.ExternalBinary.external model append reuses cont`

Theorem 18.52 (External model soundness, finite kernel)**.** *The external model* $\mathcal{M}_{\mathsf{bin}}$ *satisfies the finite kernel rules for marks, histories, internal sameness, extension, continuation, asking events, generated signatures, package policies, gap coverage, gap separation, and composite gap coverage.*

Lean4 ✓ `BEDC.FKernel.ExternalBinary.external model finite kernel soundness`

Proof. Each kernel clause is interpreted by the corresponding operation on finite binary words. Mark and history sameness become literal sameness. Extension and continuation become word extension and concatenation, whose determinacy, unit, and associativity properties are immediate in the external model. Signature policies interpret generated questioning as answer-word construction. Package and gap policies interpret classes as fibers of signatures. Composite gap coverage follows from relational composition of fibers. □

18.14 Finite theorem target

The finite proof target consists of the following named results:

T1. msame reflexivity, symmetry, transitivity.

T2. mark no-confusion.

T3. hsame reflexivity, symmetry, transitivity.

T4. history no-confusion.

T5. Ext determinacy and sameness-respect.

T6. Cont determinacy.

T7. continuation left and right units.

T8. continuation associativity, relational form.

T9. signature determinacy under AskPolicy.

T10. sameSig equivalence under AskPolicy.

T11. gap coverage under the three policies.

T12. gap separation under the three policies.

T13. composite gap coverage.

T14. certificate admission for derived interfaces.

T15. unary history and unary continuation interface seeds.

18.15 Conclusion

The finite-kernel proof target identifies the first theorem set to be checked. The main goal is no longer to expand toward more traditional mathematics, but to verify that the finite kernel is internally coherent: marks have internal sameness, histories have internal sameness, extension and continuation are relational but deterministic up to internal sameness, signatures are generated by event-style asking, packages classify signatures, gaps cover and separate sources, composite gaps preserve coverage, and derived names require certificates.

The finite kernel may be summarized as:

$$\boxed{\mathsf{b0}, \mathsf{b1} \to \mathsf{Hist} \to \mathsf{msame}, \mathsf{hsame} \to \mathsf{Ext}, \mathsf{Cont} \to \mathsf{Ask} \to \mathsf{Sig} \to \mathsf{sameSig} \to \mathsf{Pkg} \to \mathsf{Gap} \to \mathsf{NameCert}.}$$

Beyond this finite target, completion-like, fold-like, geometric, and algebraic constructions are governed by their own certificate schemas. Real-like and circle-like names inside the manuscript are obligation schemas, not primitive finite-kernel exports.

Chapter 19

Finite-Kernel Theorem Dependency DAG

This chapter gives the proof dependency order for the finite-kernel sprint. It is a planning artifact for the proof target, not a new object-theory primitive.

19.1 Milestone DAG

The dependency chain is:

A. Mark layer
↓
B. History layer
↓
C. Extension and continuation
↓
D. Asking and signature generation
↓
E. Signature sameness
↓
F. Package policy
↓
G. Gap policy
↓
H. Composite ledger
↓
I. Naming certificates

The proof effort should not attempt large derived interfaces before Milestones A–G are mechanically checked.

19.2 Milestone A: marks

1. msame-reflexivity.
2. msame-symmetry.

3. msame-transitivity.
4. Mark no-confusion: b0 is not internally the same as b1, and conversely.

19.3 Milestone B: histories

1. hsame-reflexivity.
2. hsame-symmetry.
3. hsame-transitivity.
4. History no-confusion for empty, E0, and E1 constructors.

19.4 Milestone C: relational extension and continuation

1. Ext-determinacy up to hsame.

   Lean4 ✓ `BEDC.FKernel.Ext.proof sprint extension determinacy`
2. Ext-respect for msame and hsame.
3. Cont-determinacy up to hsame.
4. Left and right unit laws for Cont.

   Lean4 ✓ `BEDC.FKernel.Cont.cont right unit proof sprint`
5. Associativity of Cont up to hsame.

19.5 Milestone D: asking and signatures

1. Generated bundle grammar is fixed.
2. Signature generation rules are fixed.
3. Under $\mathsf{AskPolicy}(\Pi, D)$, signatures exist for histories in the domain.
4. Under $\mathsf{AskPolicy}(\Pi, D)$, signatures are deterministic up to hsame.

19.6 Milestone E: signature sameness

1. sameSig-reflexivity under AskPolicy.
2. sameSig-symmetry.
3. sameSig-transitivity under deterministic signatures.
4. Signature sameness is an equivalence relation on the chosen domain.

19.7 Milestones F–G: packages and gaps

1. PackagePolicy existence.
2. PackagePolicy extensionality: internally same signatures yield internally same packages.
3. PackagePolicy grounding: internally same packages are justified by internally same signatures.
4. GapPolicy coverage: every admissible source history appears in some gap.
5. GapPolicy separation: genuinely different packages have disjoint source membership.
6. Packaging classifies by signatures.

19.8 Milestone H: composite gap

For two compression interfaces C and D, prove:

1. composite gap is defined by an intermediate witness;
2. composite coverage follows from coverage of both stages;
3. composite separation requires separation plus a suitable grounding condition.

19.9 Milestone I: naming certificates

The first mechanically useful certificates should be:

1. a unary repetition source certificate for $\mathsf{Nat}^{\uparrow}$;
2. a continuation stability certificate for $\mathsf{Add}^{\uparrow}$;
3. a descent certificate for function-like interfaces;
4. a ledger policy stating what is preserved by each derived name.

19.10 Execution rule

The implementation sprint should stop at the first unverified milestone and not skip ahead. If Milestone A fails, later milestone statements are not repaired by changing the core theory silently; the affected definitions must be amended in place per the policy in Chapter 9466.

Part III

Proof Standing

Chapter 20

Closed Generation and Internal Sameness

The finite kernel relies on a closed-generation reading: judgments are generated only by their displayed constructors. This is not an extra mathematical object inside the theory; it is the proof discipline governing the kernel rules.

Principle 20.1 (Closed generation)**.** *For each primitive judgment family introduced in the finite kernel, the only valid derivations are those generated by its displayed rules. In particular, there are no hidden conversion, coercion, or extensionality rules.*

Definition 20.2 (Mark constructors)**.** *The raw marks are generated by two constructors,* $\mathsf{b0}$ *and* $\mathsf{b1}$*. The internal mark-sameness judgment* $\mathsf{msame}(m, n)$ *is generated by the two displayed reflexive constructors:*

$$\mathsf{msame}(\mathsf{b0}, \mathsf{b0}), \qquad \mathsf{msame}(\mathsf{b1}, \mathsf{b1}).$$

No cross-constructor rule is present.

Theorem 20.3 (Mark no-confusion)**.** *The following judgments are not derivable:*

$$\mathsf{msame}(\mathsf{b0}, \mathsf{b1}), \qquad \mathsf{msame}(\mathsf{b1}, \mathsf{b0}).$$

Proof. By closed generation, every derivation of msame must end in one of the two mark-sameness constructors. The first constructor has target $\mathsf{msame}(\mathsf{b0}, \mathsf{b0})$, and the second has target $\mathsf{msame}(\mathsf{b1}, \mathsf{b1})$. Neither constructor has target $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$ or $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$. Hence the cross judgments are not generated. □

Theorem 20.4 (msame is an internal equivalence)**.** *The judgment* msame *is reflexive, symmetric, and transitive on generated marks.*

Proof. Reflexivity follows by cases on the two generated marks. Symmetry follows because each generated proof of $\mathsf{msame}(m, n)$ has identical endpoints. Transitivity follows by case analysis on the first proof and mark no-confusion on the second. □

Definition 20.5 (History constructors)**.** *The generated histories are built from an empty history* e *and two extension constructors* $\mathsf{E0}$ *and* $\mathsf{E1}$*. The internal history-sameness judgment* $\mathsf{hsame}(h, k)$ *is generated structurally from matching constructors.*

Theorem 20.6 (History no-confusion)**.** *The following judgments are not derivable:*

$$\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(h)), \quad \mathsf{hsame}(\mathsf{e}, \mathsf{E1}(h)), \quad \mathsf{hsame}(\mathsf{E0}(h), \mathsf{E1}(k)), \quad \mathsf{hsame}(\mathsf{E1}(h), \mathsf{E0}(k)).$$

Proof. By closed generation, a derivation of hsame must follow the displayed structural constructors. Empty histories can only be matched with empty histories. A history beginning with E0 can only be matched with a history beginning with E0, and similarly for E1. There is no displayed constructor crossing E0 with E1, nor a constructor crossing either extension with e. Therefore the listed judgments are not generated. □

Lean4 ✓ `BEDC.FKernel.Hist.hsame no confusion`

↪ variant `BEDC.FKernel.Hist.hsame no confusion all`
↪ variant `BEDC.FKernel.Hist.not hsame e0 empty`
↪ variant `BEDC.FKernel.Hist.not hsame e1 empty`
↪ variant `BEDC.FKernel.Hist.hsame e0 e1 iff false`
↪ variant `BEDC.FKernel.Hist.not hsame e1 e0`
↪ variant `BEDC.FKernel.Hist.hsame e1 e0 iff false`
↪ variant `BEDC.FKernel.Hist.history no confusion`
↪ variant `BEDC.FKernel.Hist.history no confusion four cases`
↪ variant `BEDC.FKernel.Hist.history no confusion empty pair`
↪ variant `BEDC.FKernel.Hist.history no confusion expanded pair`
↪ variant `BEDC.FKernel.Hist.hsame no confusion symmetric`
↪ variant `BEDC.FKernel.Hist.hsame no confusion complete`
↪ variant `BEDC.FKernel.Hist.history no confusion empty cross pair`
↪ variant `BEDC.FKernel.Hist.hsame empty iff`

Theorem 20.7 (History constructor kind preservation)**.** *If* $\mathsf{hsame}(h, k)$ *is derivable, then* h *and* k *have the same outer constructor kind: empty with empty,* E0 *with* E0*, or* E1 *with* E1*.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame constructor kind preserved`

Proof. Invert the displayed derivation of $\mathsf{hsame}(h, k)$. Each constructor of the generated judgment relates histories built by the same outer history constructor, so the conclusion follows by cases on that final constructor. □

Theorem 20.8 (hsame is an internal equivalence)**.** *The judgment* hsame *is reflexive, symmetric, and transitive on generated histories.*

Proof. Reflexivity is proved by induction on generated histories. Symmetry is proved by induction on the displayed derivation of hsame. Transitivity is proved by induction on the first derivation, using history no-confusion to invert the second derivation into the matching constructor case. □

Corollary 20.9 (E0-constructor inversion)**.** *For histories* h *and* k*,* $\mathsf{hsame}(\mathsf{E0}(h), \mathsf{E0}(k))$ *is derivable exactly when* $\mathsf{hsame}(h, k)$ *is derivable.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame e0 iff`

Proof. The forward direction inverts the final constructor of the displayed derivation. The reverse direction applies the E0-congruence constructor for hsame. □

Corollary 20.10 (E1-constructor inversion)**.** *For histories* h *and* k*,* $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k))$ *is derivable exactly when* $\mathsf{hsame}(h, k)$ *is derivable.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame e1 iff`

Proof. The forward direction inverts the final constructor of the displayed derivation. The reverse direction applies the E1-congruence constructor for hsame. □

Corollary 20.11 (Constructor inversion)**.** *If* hsame(E0$(h), x$) *is derivable, then* x *is generated through an* E0*-constructor up to the displayed structural form. If* hsame(E1$(h), x$) *is derivable, then* x *is generated through an* E1*-constructor up to the displayed structural form.*

Lean4 ✓ `BEDC.FKernel.Hist.hsame constructor inversion full`

↪ variant `BEDC.FKernel.Hist.hsame constructor inversion`

↪ variant `BEDC.FKernel.Hist.hsame empty inversion`

↪ variant `BEDC.FKernel.Hist.hsame e0 inversion`

↪ variant `BEDC.FKernel.Hist.hsame e1 inversion`

Chapter 21

Extension and Continuation Proof Spine

This chapter records the proof-standing status of the extension and continuation relations. Both are relation-generated behaviors, not primitive functions.

21.1 Relational extension

Definition 21.1 (Extension relation)**.** *The relation* $\mathsf{Ext}(m, h, r)$ *is generated by:*

$$\mathsf{Ext}(\mathsf{b0}, h, \mathsf{E0}(h)), \qquad \mathsf{Ext}(\mathsf{b1}, h, \mathsf{E1}(h)).$$

It expresses that emitting mark m *after history* h *generates result* r*.*

Theorem 21.2 (Extension determinacy up to history-sameness)**.** *If*

$$\mathsf{Ext}(m, h, r) \quad \textit{and} \quad \mathsf{Ext}(m, h, r'),$$

then $\mathsf{hsame}(r, r')$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext determinacy up to hsame spine`

Proof. By case analysis on the generated extension proof. If $m = \mathsf{b0}$, both generated results have the form $\mathsf{E0}(h)$, hence they are history-same by reflexivity. If $m = \mathsf{b1}$, both generated results have the form $\mathsf{E1}(h)$, hence they are history-same by reflexivity. No cross mark case is generated for the same mark input. □

Theorem 21.3 (Extension respects internal sameness)**.** *Suppose* $\mathsf{msame}(m, n)$*,* $\mathsf{hsame}(h, k)$*,* $\mathsf{Ext}(m, h, r)$*, and* $\mathsf{Ext}(n, k, s)$*. Then* $\mathsf{hsame}(r, s)$*.*

Lean4 ✓ `BEDC.FKernel.Ext.ext respects internal sameness spine`

Proof. By induction on the generated proof of $\mathsf{msame}(m, n)$. There are only two cases. In the $\mathsf{b0}$ case, the extension results are generated with outer constructor $\mathsf{E0}$, and the conclusion follows by applying the $\mathsf{E0}$ history-sameness constructor to $\mathsf{hsame}(h, k)$. The $\mathsf{b1}$ case is identical with $\mathsf{E1}$. □

Theorem 21.4 (Extension result source-mark characterization)**.** *If* $\mathsf{Ext}(m, h, r)$ *and* $\mathsf{Ext}(m', h', r')$*, then*

$$\mathsf{hsame}(r, r') \iff \mathsf{hsame}(h, h') \wedge \mathsf{msame}(m, m').$$

Lean4 ✓ `BEDC.FKernel.Ext.ext result hsame source mark hsame iff`

Proof. For the forward direction, use result-history injectivity for generated extensions. For the reverse direction, transport the source histories and marks across internal sameness, then compare the two generated extension results. □

Theorem 21.5 (Extension result is nonempty)**.** *If* $\mathsf{Ext}(m, h, r)$*, then* r *is not history-same to* e*.*

Proof. Case-split the generated extension proof. The $\mathsf{b0}$ case has result $\mathsf{E0}(h)$, which cannot be history-same to e by history constructor no-confusion. The $\mathsf{b1}$ case is identical with $\mathsf{E1}(h)$. □

Lean4 ✓ `BEDC.FKernel.Ext.ext result ne empty`

21.2 Relational continuation

Definition 21.6 (Continuation relation)**.** *The relation* $\mathsf{Cont}(h, k, r)$ *is generated by:*

1. $\mathsf{Cont}(h, \mathsf{e}, h)$*;*
2. *if* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{Cont}(h, \mathsf{E0}(k), \mathsf{E0}(r))$*;*
3. *if* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{Cont}(h, \mathsf{E1}(k), \mathsf{E1}(r))$*.*

It expresses continuation of history h *by history* k *without treating continuation as a primitive function.*

Theorem 21.7 (Continuation determinacy up to history-sameness)**.** *If* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{Cont}(h, k, r')$*, then* $\mathsf{hsame}(r, r')$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont determinacy up to hsame spine`

Proof. By induction on the generated right-history structure in the first continuation proof. In the empty case, the second proof must also be the empty case, hence both results are h. In the $\mathsf{E0}$ case, the second proof must also have right input and result with outer $\mathsf{E0}$ by closed generation; apply the induction hypothesis to the sub-results and then apply the $\mathsf{E0}$ constructor for hsame. The $\mathsf{E1}$ case is analogous. □

Theorem 21.8 (Continuation result history-sameness iff)**.** *If* $\mathsf{Cont}(a, f, r)$ *holds, then for every history* s*,*

$$\mathsf{Cont}(a, f, s) \iff \mathsf{hsame}(s, r).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont result hsame iff`

Proof. For the forward direction, continuation determinacy compares the two results with the same source and tail. For the reverse direction, transport the given continuation result along the displayed history-sameness. □

Theorem 21.9 (Continuation source history-sameness iff)**.** *If* $\mathsf{Cont}(a, f, r)$ *holds, then for every history* a'*,*

$$\mathsf{Cont}(a', f, r) \iff \mathsf{hsame}(a', a).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont source hsame iff`

Proof. For the forward direction, right cancellation compares the two continuations with shared tail and result. For the reverse direction, transport the source endpoint of the given continuation along the displayed history-sameness while keeping the tail and result fixed. □

Theorem 21.10 (Continuation result history-sameness tag separation)**.** *If* $\mathsf{Cont}(h, k, r_0)$ *and* $\mathsf{Cont}(h, k, r_1)$*, and if* $\mathsf{hsame}(r_0, \mathsf{E0}(z))$ *while* $\mathsf{hsame}(r_1, \mathsf{E1}(o))$*, then contradiction.*

Lean4 ✓ `BEDC.FKernel.Cont.cont result hsame tag separation`

Proof. Continuation determinacy gives $\mathsf{hsame}(r_0, r_1)$. Composing this equality with the two displayed tag classifications yields $\mathsf{hsame}(\mathsf{E0}(z), \mathsf{E1}(o))$, contradicting generated-history no-confusion between the zero and one constructors. □

Theorem 21.11 (Continuation step result history-sameness no-confusion pair)**.** *The two one-step continuation result shapes are disjoint up to history-sameness:*

$$\mathsf{Cont}(h, \mathsf{E0}(k_0), r_0), \quad \mathsf{Cont}(h, \mathsf{E1}(k_1), r_1), \quad \mathsf{hsame}(r_0, r_1)$$

is impossible, and the same holds with the displayed zero and one steps exchanged.

Lean4 ✓ `BEDC.FKernel.Cont.cont step result hsame no confusion pair`

Proof. Invert the zero-step continuation witness to expose $r_0 = \mathsf{E0}(a)$ and invert the one-step witness to expose $r_1 = \mathsf{E1}(b)$. The displayed $\mathsf{hsame}(r_0, r_1)$ then becomes $\mathsf{hsame}(\mathsf{E0}(a), \mathsf{E1}(b))$, which is ruled out by generated-history no-confusion. The exchanged direction is identical with the zero and one constructors swapped. □

Theorem 21.12 (Continuation unit laws)**.** *For every generated history* h*,* $\mathsf{Cont}(h, \mathsf{e}, h)$*. Moreover, for every generated history* k*,* $\mathsf{Cont}(\mathsf{e}, k, k)$*, and if* $\mathsf{Cont}(h, \mathsf{e}, r)$*, then* $\mathsf{hsame}(r, h)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont right unit iff`
↪variant `BEDC.FKernel.Cont.cont unit laws spine`

Proof. The right unit is the base constructor. The left unit is proved by induction on k. The empty case is immediate. If $k = \mathsf{E0}(k')$, use the induction hypothesis for k' and rebuild with the $\mathsf{E0}$ constructor. The $\mathsf{E1}$ case is analogous. The uniqueness form follows by continuation determinacy against the base constructor. □

Theorem 21.13 (Continuation right-unit law endpoint emptiness)**.** *For every generated history* u*, the universal right-unit result law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{Cont}(d, u, r) \to \mathsf{hsame}(r, d)$$

holds if and only if $u\mathsf{hsamee}$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont right unit law endpoint empty iff`

Proof. If the universal law holds, apply it to the left-unit continuation $\mathsf{Cont}(\mathsf{e}, u, u)$ to obtain $u\mathsf{hsamee}$. Conversely, after the endpoint is identified with e, any continuation $\mathsf{Cont}(d, \mathsf{e}, r)$ is a right-unit continuation, so right-unit result uniqueness gives $\mathsf{hsame}(r, d)$. □

Theorem 21.14 (Continuation left-unit law endpoint emptiness)**.** *For every generated history u, the universal left-unit result law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{Cont}(u, d, r) \to \mathsf{hsame}(r, d)$$

holds if and only if $u\mathsf{hsamee}$.

Lean4 ✓ `BEDC.FKernel.Cont.cont left unit law endpoint empty iff`

Proof. If the universal law holds, apply it to the right-unit continuation $\mathsf{Cont}(u, \mathsf{e}, u)$ to obtain $u\mathsf{hsamee}$. Conversely, after the endpoint is identified with e, any continuation $\mathsf{Cont}(\mathsf{e}, d, r)$ is a left-unit continuation, so the left-unit result theorem gives $\mathsf{hsame}(r, d)$. □

Theorem 21.15 (Continuation eone right-unit probe forces emptiness)**.** *For every generated history u, the fixed one-headed probe law*

$$\forall r : \mathsf{BHist}.\ \mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, r) \to \mathsf{hsame}(r, \mathsf{E1}(\mathsf{e}))$$

forces $u\mathsf{hsamee}$.

Lean4 ✓ `BEDC.FKernel.Cont.cont e1 right unit probe forces empty`

Proof. Apply the probe law to the canonical continuation $\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, \mathsf{append}(\mathsf{E1}(\mathsf{e}), u))$. Continuation result transport gives $\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, \mathsf{E1}(\mathsf{e}))$, and right-unit uniqueness then forces the right endpoint u to be empty. □

Theorem 21.16 (Mutual continuation collapses to history-sameness)**.** *If* $\mathsf{Cont}(h, l, k)$ *and* $\mathsf{Cont}(k, r, h)$*, then* $\mathsf{hsame}(h, k)$.

Lean4 ✓ `BEDC.FKernel.Cont.cont mutual extension hsame`

Proof. Compose the two continuation equations into a cycle from h back to h with tail $\mathsf{append}(l, r)$. Right-unit uniqueness forces this composite tail to be empty. Empty append inversion forces l to be empty, so the first continuation reduces to $\mathsf{hsame}(h, k)$. □

Theorem 21.17 (Mutual continuation has empty tails)**.** *If* $\mathsf{Cont}(h, l, k)$ *and* $\mathsf{Cont}(k, r, h)$*, then*

$$\mathsf{hsame}(l, \mathsf{e}) \quad \textit{and} \quad \mathsf{hsame}(r, \mathsf{e}).$$

Proof. Compose the two continuations into a cycle from h back to h whose tail is $\mathsf{append}(l, r)$. Right-unit uniqueness forces that composite tail to be empty, and empty-append inversion identifies both component tails with e. □

Lean4 ✓ `BEDC.FKernel.Cont.cont mutual extension tails empty`

Theorem 21.18 (Composite endpoint empty tails iff)**.** *For histories* $a, b, c, f, g : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(a, f, b) \qquad \textit{and} \qquad \mathsf{Cont}(b, g, c),$$

then the composite endpoints close exactly when the two exposed tails are empty:

$$\mathsf{hsame}(c, a) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Proof. Compose the two continuation ledgers into $\mathsf{Cont}(a, \mathsf{append}(f, g), c)$. If $c\mathsf{hsame}a$, transport the result to obtain a cycle $\mathsf{Cont}(a, \mathsf{append}(f, g), a)$; right-unit uniqueness and empty-append inversion identify f and g with e.

Conversely, if both exposed tails are empty, the first continuation reduces b to a and the second reduces c to b by determinacy against right-unit continuations. Transitivity gives $c\mathsf{hsame}a$. □

Lean4 ✓ `BEDC.FKernel.Cont.cont composite endpoint empty tails iff`

Theorem 21.19 (Nested continuation with visible middle cannot end empty)**.** *For histories* $l, tail, r, u, v : \mathsf{BHist}$*, neither branch*

$$\mathsf{Cont}(l, \mathsf{E0}(tail), u), \qquad \mathsf{Cont}(u, r, v), \qquad \mathsf{hsame}(v, \mathsf{e})$$

nor branch

$$\mathsf{Cont}(l, \mathsf{E1}(tail), u), \qquad \mathsf{Cont}(u, r, v), \qquad \mathsf{hsame}(v, \mathsf{e})$$

can hold.

Proof. In either branch, the outer continuation and $v\mathsf{hsame}\mathsf{e}$ force $\mathsf{append}(u, r)\mathsf{hsame}\mathsf{e}$. Empty-append inversion gives $u\mathsf{hsame}\mathsf{e}$. The middle continuation then forces $\mathsf{append}(l, \mathsf{E0}(tail))\mathsf{hsame}\mathsf{e}$ or $\mathsf{append}(l, \mathsf{E1}(tail))\mathsf{hsame}\mathsf{e}$, and a second empty-append inversion identifies the headed constructor with e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.FKernel.Cont.cont nested visible middle result absurd`

Theorem 21.20 (Nested continuation empty-result inversion)**.** *For histories* $a, b, c, f, g : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, g, c), \qquad c\mathsf{hsame}\mathsf{e},$$

then all exposed endpoints and tails of the nested continuation are empty:

$$a\mathsf{hsame}\mathsf{e}, \qquad f\mathsf{hsame}\mathsf{e}, \qquad b\mathsf{hsame}\mathsf{e}, \qquad g\mathsf{hsame}\mathsf{e}.$$

Proof. Transport the outer continuation result along $c\mathsf{hsame}\mathsf{e}$ and apply empty-result inversion to $\mathsf{Cont}(b, g, \mathsf{e})$, identifying b and g with e. With the middle endpoint identified as empty, apply the same empty-result inversion to the inner continuation $\mathsf{Cont}(a, f, b)$ to identify a and f with e. □

Lean4 ✓ `BEDC.FKernel.Cont.cont nested empty result inversion`

Theorem 21.21 (Mutual continuation forbids visible right tails)**.** *If* $\mathsf{Cont}(h, l, k)$ *holds, then neither* $\mathsf{Cont}(k, \mathsf{E0}(r), h)$ *nor* $\mathsf{Cont}(k, \mathsf{E1}(r), h)$ *can hold.*

Proof. Apply the empty-tail theorem to the two mutual continuations. In the $\mathsf{E0}(r)$ branch it would identify $\mathsf{E0}(r)$ with e, contradicting history constructor no-confusion. The $\mathsf{E1}(r)$ branch is identical with the one-constructor no-confusion theorem. □

Lean4 ✓ `BEDC.FKernel.Cont.cont mutual extension right tail absurd`

Theorem 21.22 (Triangular continuation cycle forbids a visible left tail)**.** *If*

$$\mathsf{Cont}(a, \mathsf{E1}(k), b), \qquad \mathsf{Cont}(b, g, c), \qquad \mathsf{Cont}(c, h, a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(\mathsf{E1}(k), g), c)$. This composite and the return continuation $\mathsf{Cont}(c, h, a)$ form a mutual continuation cycle, so the composite tail is empty. Empty-append inversion identifies $\mathsf{E1}(k)$ with e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle left visible tail absurd`

Theorem 21.23 (Triangular continuation cycle forbids a zero left tail)**.** *If*

$$\mathsf{Cont}(a, \mathsf{E0}(k), b), \qquad \mathsf{Cont}(b, g, c), \qquad \mathsf{Cont}(c, h, a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(\mathsf{E0}(k), g), c)$. This composite and the return continuation $\mathsf{Cont}(c, h, a)$ form a mutual continuation cycle, so the composite tail is empty. Empty-append inversion identifies $\mathsf{E0}(k)$ with e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle left zero tail absurd`

Theorem 21.24 (Triangular continuation cycle forbids a visible middle tail)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, \mathsf{E1}(k), c), \qquad \mathsf{Cont}(c, h, a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(f, \mathsf{E1}(k)), c)$. This composite and the return continuation form a mutual continuation cycle, so the composite tail is empty. Empty-append inversion identifies the middle visible tail $\mathsf{E1}(k)$ with e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle middle visible tail absurd`

Theorem 21.25 (Triangular continuation cycle forbids a zero middle tail)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, \mathsf{E0}(k), c), \qquad \mathsf{Cont}(c, h, a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(f, \mathsf{E0}(k)), c)$. This composite and the return continuation form a mutual continuation cycle, so the composite tail is empty. Empty-append inversion identifies the middle zero tail $\mathsf{E0}(k)$ with e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle middle zero tail absurd`

Theorem 21.26 (Triangular continuation cycle forbids a visible right tail)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, g, c), \qquad \mathsf{Cont}(c, \mathsf{E1}(k), a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(f, g), c)$. The composite continuation and the visible return tail give a mutual continuation with a one-headed right tail, which is forbidden by right-tail mutual-extension cancellation. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle right visible tail absurd`

Theorem 21.27 (Triangular continuation cycle forbids a zero right tail)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, g, c), \qquad \mathsf{Cont}(c, \mathsf{E0}(k), a)$$

all hold, then contradiction follows.

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(f, g), c)$. The composite continuation and the zero return tail give a mutual continuation with a zero-headed right tail, which is forbidden by right-tail mutual-extension cancellation. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle right zero tail absurd`

Theorem 21.28 (Triangular continuation cycle has empty tails)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, g, c), \qquad \mathsf{Cont}(c, h, a)$$

all hold, then all three continuation tails are empty and the adjacent endpoints collapse:

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Compose the first two continuations into $\mathsf{Cont}(a, \mathsf{append}(f, g), c)$. This composite and the return continuation $\mathsf{Cont}(c, h, a)$ form a mutual continuation cycle, so the composite tail and the return tail are empty. Empty-append inversion splits the composite tail into $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$. Comparing the first two continuation ledgers with right-unit continuations at a and b gives the adjacent endpoint equalities. □

Lean4 ✓ `BEDC.FKernel.Cont.cont triangle cycle tails empty`

Theorem 21.29 (Associativity up to history-sameness)**.** *Suppose*

$$\mathsf{Cont}(a, b, ab), \quad \mathsf{Cont}(ab, c, left), \quad \mathsf{Cont}(b, c, bc), \quad \mathsf{Cont}(a, bc, right).$$

Then $\mathsf{hsame}(left, right)$.

Lean4 ✓ `BEDC.FKernel.Cont.cont assoc up to hsame spine`

Proof. Use the right-recursive append associativity spine underlying Cont: first identify each continuation proof with its append result, then apply associativity for recursion on the right history c. The two displayed composites are therefore the same append tree up to hsame. □

Theorem 21.30 (Composite continuation tail uniqueness)**.** *If*

$$\mathsf{Cont}(h, f, k), \quad \mathsf{Cont}(k, g, r), \quad \mathsf{Cont}(h, tail, r),$$

then $\mathsf{hsame}(tail, \mathsf{append}(f, g))$.

Lean4 ✓ `BEDC.FKernel.Cont.cont composite tail unique`

Proof. Compose the first two continuation witnesses using associativity to obtain a direct witness from h to r with tail $\mathsf{append}(f, g)$. Left cancellation for continuation then identifies this composite tail with the given direct tail. □

Theorem 21.31 (Composite continuation tail iff)**.** *If*

$$\mathsf{Cont}(a, f, b), \quad \mathsf{Cont}(b, g, c),$$

then for every displayed tail t,

$$\mathsf{Cont}(f, g, t) \iff \mathsf{Cont}(a, t, c).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont composite tail iff`

Proof. Forwards, reassociate the two displayed continuation equations: the tail continuation $\mathsf{Cont}(f, g, t)$ turns the path from a through t into the same append tree as the path from a through f and then g. Conversely, the direct continuation $\mathsf{Cont}(a, t, c)$ shares its source and target with the composite of $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, c)$, so composite-tail uniqueness identifies t with $\mathsf{append}(f, g)$ and hence gives $\mathsf{Cont}(f, g, t)$. □

Theorem 21.32 (Composite continuation left factor)**.** *If*

$$\mathsf{Cont}(b, g, c), \quad \mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(a, fg, c),$$

then $\mathsf{Cont}(a, f, b)$.

Lean4 ✓ `BEDC.FKernel.Cont.cont composite left factor`

Proof. The displayed composite continuation $\mathsf{Cont}(a, fg, c)$ and the factor continuation $\mathsf{Cont}(f, g, fg)$ rewrite c as $\mathsf{append}(\mathsf{append}(a, f), g)$. Compare this with $\mathsf{Cont}(b, g, c)$ and cancel the common right suffix g; the middle object is therefore hsame to $\mathsf{append}(a, f)$, and the reflexive continuation from a by f transports to b. □

Theorem 21.33 (Composite continuation right factor)**.** *If*

$$\mathsf{Cont}(a, f, b), \quad \mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(a, fg, c),$$

then $\mathsf{Cont}(b, g, c)$.

Lean4 ✓ `BEDC.FKernel.Cont.cont composite right factor`

Proof. Invert the three continuation witnesses. The result endpoint c is $\mathsf{append}(a, \mathsf{append}(f, g))$, while the desired right factor endpoint is $\mathsf{append}(\mathsf{append}(a, f), g)$. BEDC append associativity identifies these two histories and supplies the displayed continuation from b by g to c. □

Theorem 21.34 (Composite continuation factorization iff)**.** *If* $\mathsf{Cont}(f, g, fg)$*, then a displayed composite continuation from a to c by fg is equivalent to the existence of an intermediate history b carrying a by f to b and b by g to c:*

$$\mathsf{Cont}(a, fg, c) \iff \exists b : \mathsf{BHist},\ \mathsf{Cont}(a, f, b) \wedge \mathsf{Cont}(b, g, c).$$

Lean4 ✓ `BEDC.FKernel.Cont.cont composite factorization iff`

Proof. The forward direction chooses the canonical endpoint $b = \mathsf{append}(a, f)$ and applies right-factor recovery for the displayed composite. Conversely, any displayed intermediate factorization composes by rewriting the middle endpoint with $\mathsf{Cont}(a, f, b)$ and applying associativity of append against the fixed witness $\mathsf{Cont}(f, g, fg)$. □

Theorem 21.35 (Composite continuation canonical middle public readback)**.** *If* $\mathsf{Cont}(f, g, fg)$ *and* $\mathsf{Cont}(a, fg, c)$*, then the canonical middle history* $\mathsf{append}(a, f)$ *displays both factors,*

$$\mathsf{Cont}(a, f, \mathsf{append}(a, f)) \quad \textit{and} \quad \mathsf{Cont}(\mathsf{append}(a, f), g, c),$$

and any other displayed middle b *with* $\mathsf{Cont}(a, f, b)$ *and* $\mathsf{Cont}(b, g, c)$ *satisfies* $\mathsf{hsame}(\mathsf{append}(a, f), b)$*.*

Lean4 ✓ `BEDC.FKernel.Cont.cont composite canonical middle public readback`

Proof. The left factor is the defining continuation for append. The right factor is the composite right-factor theorem applied to this canonical left factor, the tail composite $\mathsf{Cont}(f, g, fg)$, and the displayed composite $\mathsf{Cont}(a, fg, c)$. For another middle b, inverting $\mathsf{Cont}(a, f, b)$ identifies b with $\mathsf{append}(a, f)$. □

Theorem 21.36 (Continuation factorization result history-sameness iff)**.** *If*

$$\mathsf{Cont}(a, b, m), \qquad \mathsf{Cont}(m, c, r),$$

then for every direct result s *through the appended tail,*

$$\mathsf{Cont}(a, \mathsf{append}(b, c), s) \iff \mathsf{hsame}(s, r).$$

Proof. The two displayed continuation witnesses compose to a direct continuation from a by $\mathsf{append}(b, c)$ to r, using append associativity. The result history-sameness characterization for a fixed continuation then identifies exactly the direct results s with $\mathsf{hsame}(s, r)$. □

Lean4 ✓ `BEDC.FKernel.Cont.cont factorization result hsame iff`

Theorem 21.37 (Continuation factorization middle exactness)**.** *If*

$$\mathsf{Cont}(a, f, b), \qquad \mathsf{Cont}(b, g, c),$$

then a displayed alternative middle b' *carries both factors exactly when it is history-same to the original middle:*

$$\mathsf{Cont}(a, f, b') \wedge \mathsf{Cont}(b', g, c) \iff \mathsf{hsame}(b', b).$$

Proof. Forwards, determinacy of $\mathsf{Cont}(a, f, _)$ compares the alternative first factor with the original first factor. Conversely, a history-sameness witness from b' to b transports the first factor's result and the second factor's source, yielding the two displayed continuation witnesses. □

Lean4 ✓ `BEDC.FKernel.Cont.cont factorization middle hsame iff`

Corollary 21.38 (Addition-like behavior requires no primitive addition)**.** *The continuation relation supplies a finite-kernel seed for addition-like derived interfaces. The additive interface is not primitive; it may be licensed only by a naming certificate whose stability component cites continuation unit and associativity.*

Lean4 ✓ `BEDC.FKernel.Cont.continuation addition like seed has unit assoc`

Theorem 21.39 (Continuation ezero right-unit probe forces emptiness)**.** *For every generated history* u*, the fixed zero-headed probe law*

$$\forall r : \mathsf{BHist}.\ \mathsf{Cont}(\mathsf{E0}(\mathsf{e}), u, r) \to \mathsf{hsame}(r, \mathsf{E0}(\mathsf{e}))$$

forces $u\mathsf{hsamee}$*.*

Proof. Apply the probe law to the canonical continuation $\mathsf{Cont}(\mathsf{E0}(\mathsf{e}), u, \mathsf{append}(\mathsf{E0}(\mathsf{e}), u))$. Continuation result transport gives $\mathsf{Cont}(\mathsf{E0}(\mathsf{e}), u, \mathsf{E0}(\mathsf{e}))$, and right-unit uniqueness then forces the right endpoint u to be empty. □

Chapter 22

Signature Proof Spine

This chapter turns asking events and generated bundles into signature events. No probe is treated as a function.

Definition 22.1 (Bundle generation)**.** *Probe bundles are generated by* Bnil *and* $\mathsf{Bcons}(\pi, \Pi)$*. They are not imported as external finite lists.*

Definition 22.2 (Asking policy)**.** $\mathsf{AskPolicy}(\Pi, D)$ *is paper notation for using the checked* $\mathsf{AskPolicy}(D)$ *certificate at bundle* Π*. The certificate gives totality of asking, determinacy of asking up to* msame*, and respect for* hsame *on histories in* D*. Its Lean fields quantify over each probe name available to signature generation, so* Π *marks the bundle site where the certificate is used rather than a field that restricts the certificate.*

Definition 22.3 (Signature event)**.** $\mathsf{Sig}(\Pi, h, s, \Delta)$ *means that asking bundle* Π *against history* h *generates answer history* s *with evidence ledger* Δ*. It is generated by the empty-bundle rule and the cons-bundle rule.*

Lean4 ✓ `BEDC.FKernel.Sig.signature event generation pack`

Theorem 22.4 (Signature determinacy)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. If* $h \in D$*,* $\mathsf{Sig}(\Pi, h, s, \Delta)$*, and* $\mathsf{Sig}(\Pi, h, t, \Theta)$*, then* $\mathsf{hsame}(s, t)$*.*

Lean4 ✓ `BEDC.FKernel.Sig.signature determinacy proof spine`
↪ variant `BEDC.FKernel.Sig.signature determinacy spine`
↪ variant `BEDC.FKernel.Sig.sig cons full witness determinacy`

Proof. Induct on the generated bundle Π. For the empty bundle, both answer histories are empty. For a cons bundle, determinacy of the first asking event gives internally same output marks, while the induction hypothesis gives internally same tail signatures. The result follows by the matching history constructor. □

Definition 22.5 (Signature sameness)**.** $\mathsf{sameSig}(\Pi, h, k)$ *holds when there exist generated signatures* s, t *for* h, k *under* Π *with* $\mathsf{hsame}(s, t)$*.*

Theorem 22.6 (Signature sameness equivalence)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*. Then* $\mathsf{sameSig}(\Pi, -, -)$ *is reflexive, symmetric, and transitive on histories in* D*.*

Lean4 ✓ `BEDC.FKernel.Sig.signature sameness equivalence proof spine`

Proof. For reflexivity, generate a signature s of h by totality and use $\mathsf{hsame}(s, s)$. Symmetry swaps the two witnessing signatures and uses symmetry of hsame. For transitivity, choose witnesses s_h, s_k for h, k and t_k, t_l for k, l. Since both s_k and t_k are signatures of k, signature determinacy gives $\mathsf{hsame}(s_k, t_k)$. Transitivity of hsame gives $\mathsf{hsame}(s_h, t_l)$. □

Chapter 23

Package and Gap Proof Spine

The finite globalization interface is not a single primitive. It stands on three policies: asking, packaging, and gaps.

Definition 23.1 (Package policy)**.** $\mathsf{PackagePolicy}(\Pi)$ *states that signatures generated from histories admitted through* $\mathsf{AskPolicy}(\Pi, D)$ *and used by* $\mathsf{GapPolicy}(\Pi, D)$ *admit package tokens, that internally same signatures yield internally same packages, and that package sameness is grounded in internally same signatures.*

Definition 23.2 (Gap policy)**.** $\mathsf{GapPolicy}(\Pi, D)$ *states that gap membership is determined by signatures and package tokens: a history is in the gap of a package precisely when it generates a signature belonging to that package.*

Theorem 23.3 (Gap coverage)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*, and* $\mathsf{GapPolicy}(\Pi, D)$*. If* $h \in D$*, then there exists a package token* p *such that* $\mathsf{InGap}(\Pi, D, p, h)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap coverage from policy layers`

Proof. By asking totality, generate a signature s of h. By package existence, generate a package token p for s. By the gap policy, h is a member of the gap of p. □

Theorem 23.4 (Gap separation)**.** *Assume* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*, and* $\mathsf{GapPolicy}(\Pi, D)$*. If* $h \in D$*,* $\mathsf{InGap}(\Pi, D, p, h)$*, and* $\mathsf{InGap}(\Pi, D, q, h)$*, then* $\mathsf{psame}(p, q)$*.*

Lean4 ✓ `BEDC.FKernel.Gap.gap separation witnesses from policy layers`

Proof. By the gap policy, gap membership yields signatures s, t of h packaged by p, q. Signature determinacy gives $\mathsf{hsame}(s, t)$. Package extensionality then yields $\mathsf{psame}(p, q)$. □

Theorem 23.5 (Globalization classifies by signatures)**.** *Under* $\mathsf{AskPolicy}(\Pi, D)$*,* $\mathsf{PackagePolicy}(\Pi)$*, and* $\mathsf{GapPolicy}(\Pi, D)$*, package tokens classify histories in* D *by signature sameness: histories with the same signature inhabit internally same packages, and histories in internally same grounded packages have internally same signatures.*

Lean4 ✓ `BEDC.FKernel.Package.packagePolicy classifies signatures`

↪ variant `BEDC.FKernel.Package.packages classify signatures`

↪ variant `BEDC.FKernel.Package.PackagePolicy grounding witness`

↪ variant `BEDC.FKernel.Package.packagePolicy classification directions`

↪ variant `BEDC.FKernel.Package.packages classify signatures directional`

Proof. The first direction follows from package extensionality. The second follows from package grounding. The gap policy connects package membership back to source histories, ensuring that the classification is not a detached visible label but a provenance-preserving package. □

Definition 23.6 (Composite gap)**.** *Given compression interfaces $C : X \rightsquigarrow Y$ and $D : Y \rightsquigarrow Z$, define a composite gap by*

$$\mathsf{CompGap}_{D \circ C}(z, x)$$

when there exists an intermediate visible item y such that x lies in the C-gap of y and y lies in the D-gap of z.

Theorem 23.7 (Composite gap coverage)**.** *If the C-gap covers each source item and the D-gap covers each C-visible item, then the composite gap covers each source item.*

Lean4 ✓ `BEDC.FKernel.Gap.proof spine composite gap coverage`

Proof. Given a source item x, choose a C-visible item y whose gap contains x. Then choose a D-visible item z whose gap contains y. By definition of composite gap, x lies in the composite gap of z. □

Corollary 23.8 (Ledger composition principle)**.** *A composite compression has a ledger made of the first compression ledger plus the pullback of the second compression ledger along the first visible interface.*

Lean4 ✓ `BEDC.FKernel.Gap.ledger composition principle proof standing`

Chapter 24

Naming Certificate Proof Instances

This chapter records the first proof-standing instances of the naming protocol. These are still interface seeds; they are not full reconstructions of traditional arithmetic.

24.1 The unary repetition certificate

Definition 24.1 (Unary repetition source)**.** *The unary source consists of histories generated from* e *by repeated* $\mathsf{E1}$*-extension.*

Definition 24.2 (Unary classifier)**.** *Two unary histories are classified together when their generation pattern is internally the same under* hsame*. The classifier is intentionally weaker than a full arithmetic equality; it only records the repetition pattern available at this stage.*

Theorem 24.3 (Unary repetition closure)**.** *The unary source is closed under* $\mathsf{E1}$*-extension.*

Lean4 ✓ `BEDC.FKernel.Unary.unary e1 closed`

Proof. By definition, unary histories are generated by repeated $\mathsf{E1}$-extension. Extending such a history by $\mathsf{E1}$ gives another history generated by repeated $\mathsf{E1}$-extension. □

Corollary 24.4 (License for $\mathsf{Nat}^{\uparrow}$)**.** *A name* $\mathsf{Nat}^{\uparrow}$ *is licensed once its certificate records unary source, unary repetition pattern, unary classifier, closure under* $\mathsf{E1}$*, and the ledger of the underlying histories.*

24.2 The continuation certificate for addition-like behavior

Definition 24.5 (Unary continuation pattern)**.** *The unary continuation pattern is the restriction of* Cont *to unary source histories.*

Theorem 24.6 (Unary continuation closure)**.** *If* h *and* k *are unary histories and* $\mathsf{Cont}(h, k, r)$*, then* r *is unary up to* hsame*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary continuation closure up to hsame`

Proof. Induct on the generation of h. If $h = \mathsf{e}$, then r is internally same as k, hence unary. If $h = \mathsf{E1}(h')$, the continuation rule reduces the result to $\mathsf{E1}(r')$ where r' is obtained by continuing h' by k. Apply the induction hypothesis and close under $\mathsf{E1}$. The $\mathsf{E0}$-case is absent for unary histories. □

Theorem 24.7 (Additive unary continuation transport)**.** *For unary histories h and k with* $\mathsf{Cont}(h, k, r)$*, if* $h\mathsf{hsame}h'$*,* $k\mathsf{hsame}k'$*, and* $r\mathsf{hsame}r'$*, then h', k', and r' are unary and* $\mathsf{Cont}(h', k', r')$*.*

Proof. Transport the two unary inputs across hsame, transport the continuation relation across the same endpoints and result, and close the transported result by unary continuation closure. □

Lean4 ✓ `BEDC.Derived.AddUp.AddUnaryContinuation hsame transport`

Theorem 24.8 (Addition-like unit)**.** *The empty history acts as a left unit for unary continuation, and as a right unit up to* hsame*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary addition like unit with certificate`
↪ variant `BEDC.FKernel.Unary.unary addition like unit spine`
↪ variant `BEDC.FKernel.Unary.addition like unit laws`

Proof. The left unit is the continuation left-unit theorem. The right unit is continuation right unit up to hsame, restricted to unary histories. □

Theorem 24.9 (Addition-like associativity)**.** *Unary continuation is associative up to* hsame*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary cont assoc with closure`

Proof. Restrict the continuation associativity theorem to unary histories. Unary continuation closure ensures that all intermediate histories remain within the unary source up to hsame. □

Corollary 24.10 (License for $\mathsf{Add}^{\uparrow}$)**.** *A name* $\mathsf{Add}^{\uparrow}$ *is licensed once its certificate records unary source, unary continuation pattern, descent to unary classes, unit behavior, associativity behavior, and composite ledger policy.*

↪ variant `BEDC.FKernel.Unary.add up licensed not primitive`

Remark 24.11 (Interface seed, not full arithmetic)**.** *The licenses for* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Add}^{\uparrow}$ *are first interface seeds. They do not yet assert a full induction principle, full recursion scheme, commutativity of addition, multiplication, order, or semiring structure.*

Part IV

Hardening

Chapter 25

Package and Gap Policy Expansion

The BEDC kernel separates asking, packaging, and gap recording. This chapter refines the public obligations of package and gap policies without changing the core declarations.

25.1 Package policy as a public interface

Definition 25.1 (Package-policy interface)**.** *A package policy for a bundle* Π *is a public contract specifying how generated answer histories are represented by visible package tokens. It is not a quotient object and it is not a set of equivalence classes. It consists of three obligations:*

1. ***Existence:*** *every admitted signature history has an admitted package token.*
2. ***Extensionality:*** *internally same signature histories admit internally same package tokens.*
3. ***Grounding:*** *internally same package tokens are justified by internally same admitted signatures.*

```
Lean4 ✓  BEDC.FKernel.Package.PackagePolicy iff fields
  ↪ variant  BEDC.FKernel.Package.packagePolicy field witnesses
  ↪ variant  BEDC.FKernel.Package.packagePolicy token exists
  ↪ variant  BEDC.FKernel.Package.packagePolicy nonempty token
  ↪ variant  BEDC.FKernel.Package.packagePolicy fields
  ↪ variant  BEDC.FKernel.Package.packagePolicy extensionality witness
```

Principle 25.2 (No quotient shortcut)**.** *A package token is not introduced as a quotient class. It is introduced through a package policy. Quotient-like behavior is then a theorem, not a primitive.*

```
Lean4 ✓  BEDC.FKernel.Package.packagePolicy no quotient shortcut
```

Theorem 25.3 (Package tokens are signature-facing)**.** *Assume a package policy for* Π*. If two source histories produce internally same signatures under* Π*, then their visible package tokens are internally same under the package-sameness relation licensed by the package policy.*

```
Lean4 ✓  BEDC.FKernel.Package.package tokens signature facing
  ↪ variant  BEDC.FKernel.Package.package tokens are signature facing
```

Proof. The source histories first produce signatures. The internal sameness of signatures triggers the extensionality field of the package policy. Hence the package tokens assigned to these signatures are package-same. No quotient object is invoked. □

25.2 Gap policy as provenance interface

Definition 25.4 (Gap-policy interface)**.** *A gap policy for a package policy over domain D specifies a provenance membership relation*

$$\mathsf{InGap}_{\Pi,D}(p, h),$$

read as: history h is recorded in the gap ledger behind visible package token p. It must satisfy three obligations:

1. ***Coverage:*** *every admitted source history belongs to at least one visible gap ledger.*
2. ***Separation:*** *if a source history belongs to two package ledgers, then those packages are package-same.*
3. ***Signature determination:*** *membership is justified by a signature event and a package event for that signature.*

Lean4 ✓ `BEDC.FKernel.Gap.gapPolicy provenance interface fields`

↪ variant `BEDC.FKernel.Gap.GapPolicyInterface iff fields`

↪ variant `BEDC.FKernel.Gap.gapPolicy provenance coverage generation pair`

↪ variant `BEDC.FKernel.Gap.gapPolicy coverage separation generation`

↪ variant `BEDC.FKernel.Gap.gap policy provenance interface`

↪ variant `BEDC.FKernel.Gap.gapPolicy provenance signature token witnesses`

Theorem 25.5 (Gap ledgers are not optional)**.** *For any nontrivial package policy, a gap policy is required to recover provenance. Without it, the visible package token has no internal record of which histories were compressed into it.*

Lean4 ✓ `BEDC.FKernel.Gap.gapPolicy requires provenance ledger`

Proof. Packaging replaces source histories by visible tokens. If a visible token is used without a gap ledger, then the theory no longer contains an internal witness of the histories that led to the token. This violates the ledger-for-compression principle. Hence a gap policy is part of the admissible globalization interface. □

Corollary 25.6 (Globalization has three layers)**.** *A well-formed finite globalization interface is not merely a packaging operation. It consists of an asking policy, a package policy, and a gap policy.*

Lean4 ✓ `BEDC.FKernel.Gap.globalization has three layers`

25.3 Composite gaps

Definition 25.7 (Composite gap ledger)**.** *Let C compress histories into intermediate visible tokens and let D compress those intermediate tokens into final visible tokens. The composite gap relation is generated by a two-step witness:*

$$\mathsf{CompGap}_{D\circ C}(z, h)$$

when there exists an intermediate token y such that h is in the C-gap behind y, and y is in the D-gap behind z.

Lean4 ✓ `BEDC.FKernel.Gap.hardening composite gap ledger witnesses`

Theorem 25.8 (Composite coverage)**.** *If C has coverage on the source domain and D has coverage on the intermediate image, then $D \circ C$ has composite coverage on the source domain.*

Lean4 ✓ `BEDC.FKernel.Gap.hardening composite coverage`

↪ variant `BEDC.FKernel.Gap.compGap coverage exact`

↪ variant `BEDC.FKernel.Gap.compGap coverage with intermediate`

↪ variant `BEDC.FKernel.Gap.compGap coverage with source and intermediate`

↪ variant `BEDC.FKernel.Gap.compGap coverage sourced witness`

Proof. Let h be an admitted source history. By coverage of C, there exists an intermediate token y with h in the C-gap behind y. By coverage of D, there exists a final token z with y in the D-gap behind z. Thus the pair (y, z) generates a composite gap witness placing h behind z. □

Remark 25.9. *Composite gap ledgers are the formal expression of the guiding slogan that a compressed result must retain a source map. The map may be direct or staged; the staged case is handled by composite gaps.*

Chapter 26

Typed Naming Certificate Instances

The BEDC paper template specifies the five fields of a derived-interface naming certificate. This chapter records explicit instance templates. These are not complete reconstructions of traditional mathematics. They are interface seeds whose role is to show how the naming mechanism should be used.

26.1 Certificate fields

Definition 26.1 (Typed naming certificate, expanded template)**.** *A typed paper certificate template for a derived interface N is a record*

$$\mathsf{NameCert}(N) = (\mathsf{SourceSpec}_N, \mathsf{PatternSpec}_N, \mathsf{ClassifierSpec}_N, \mathsf{StabilityCert}_N, \mathsf{LedgerPolicy}_N).$$

The fields have the following intended roles:

1. $\mathsf{SourceSpec}_N$*: declares the permitted source histories or visible tokens.*
2. $\mathsf{PatternSpec}_N$*: states the behavior pattern being named.*
3. $\mathsf{ClassifierSpec}_N$*: states how the behavior is packaged or identified.*
4. $\mathsf{StabilityCert}_N$*: records closure, reuse, descent, composition, or seal stability.*
5. $\mathsf{LedgerPolicy}_N$*: specifies which gap ledger records the compression performed by the name.*

Lean4 (definition) `BEDC.FKernel.NameCert.SemanticNameCert`

Lean4 ✓ `BEDC.FKernel.NameCert.NameCert iff semantic fields`

Theorem 26.2 (Semantic NameCert pattern-ledger witness)**.** *Every semantic naming certificate contains a source history whose pattern condition and ledger policy both hold.*

Proof. Eliminate the semantic certificate to expose its core naming certificate and the pattern and ledger soundness fields. The core certificate supplies a source history satisfying the source specification. Applying the two soundness fields to that same source witness yields the required pattern and ledger witnesses. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert pattern ledger witness`

Theorem 26.3 (Semantic NameCert pattern-ledger transport)**.** *If a semantic naming certificate classifies a source history h with a history k, then the pattern condition and ledger policy both hold at k.*

Proof. Use carrier transport from the core naming certificate along the classifier judgment to obtain the source specification at k. The semantic pattern and ledger soundness fields then apply at the transported history. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert pattern ledger transport`

Theorem 26.4 (Semantic NameCert classifier chain transport)**.** *If a semantic naming certificate classifies h with k and k with r, then a source witness at h transports the pattern condition and ledger policy to r.*

Proof. Use transitivity from the core naming certificate to compose the classifier judgments into a classifier judgment from h to r. The pattern-ledger transport theorem then applies to this composite classifier judgment and the source witness at h. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert classifier chain transport`

Theorem 26.5 (Semantic NameCert classifier left-confluence transport)**.** *If a semantic naming certificate classifies a shared left endpoint h with k and with r, then a source witness at k transports the pattern condition and ledger policy to r.*

Proof. Use symmetry from the core naming certificate to reverse the classifier judgment from h to k, then use transitivity with the classifier judgment from h to r. This gives a classifier judgment from k to r, so pattern-ledger transport applies to the source witness at k. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert classifier left confluence transport`

Theorem 26.6 (Semantic NameCert classifier right-confluence transport)**.** *If a semantic naming certificate classifies h and k with a shared right endpoint r, then a source witness at h transports the pattern condition and ledger policy to k.*

Proof. Use symmetry from the core naming certificate to reverse the classifier judgment from k to r, giving a classifier judgment from r to k. Then use transitivity with the classifier judgment from h to r. This gives a classifier judgment from h to k, so pattern-ledger transport applies to the source witness at h. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert classifier right confluence transport`

Theorem 26.7 (Semantic NameCert classifier reverse-chain transport)**.** *If a semantic naming certificate classifies h with k and k with r, then a source witness at r transports the pattern condition and ledger policy back to h.*

Proof. Use transitivity from the core naming certificate to compose the two classifier judgments into a classifier judgment from h to r. Symmetry reverses this composite into a classifier judgment from r to h, and the pattern-ledger transport theorem applies to the source witness at r. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert classifier reverse chain transport`

Theorem 26.8 (Semantic NameCert classifier diamond transport)**.** *If a semantic naming certificate classifies a with b, classifies a with c, and classifies b with d, then a source witness at c transports the pattern condition and ledger policy to d.*

Proof. Use symmetry to reverse the classifier judgment from a to c. Transitivity then carries c to b through a, and another transitivity step carries c to d through b. The pattern-ledger transport theorem applies to this composite classifier judgment and the source witness at c. □

Lean4 ✓ `BEDC.FKernel.NameCert.semanticNameCert classifier diamond transport`

Theorem 26.9 (Carrier self semantic lifting)**.** *Let* $C : \mathsf{BHist} \to \mathsf{Prop}$ *and* $R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*. Any core certificate* $\mathsf{core} : \mathsf{NameCert}(C, R)$ *induces a semantic certificate whose core field is* core *and whose pattern and ledger soundness fields are identity implications over the same carrier. This is the carrier-as-pattern and carrier-as-ledger paper certificate target, distinct from* `BEDC.FKernel.NameCert.NameCert_iff_semantic_fields`*. The target is*

$$\mathsf{SemanticNameCert}(C, C, C, R).$$

Proof. Choose carrier C, classifier R, pattern C, and ledger C. Keep the supplied certificate as the core field. Stability under classifier transport is exactly the carrier stability field already contained in that core certificate, and both semantic soundness fields are filled by the identity implication from a carrier witness of $C(h)$ to the same proposition $C(h)$. □

Lean4 ✓ `BEDC.FKernel.NameCert.NameCert carrier self semantic lifting`

Theorem 26.10 (Eqtype class semantic NameCert)**.** *For any BEDC history anchor* a*, the anchor class* $h \mapsto \mathsf{hsame}(h, a)$ *is a semantic naming certificate when the source, pattern, and ledger predicates are this same class and the classifier is* hsame*.*

Proof. The anchor itself inhabits the class by reflexivity of hsame. Reflexivity, symmetry, and transitivity of hsame supply the core classifier laws. Carrier transport reverses the classifier comparison and composes it with the displayed anchor comparison. The pattern and ledger fields are the identity implications over the same anchor class. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClass semanticNameCert`

26.2 Unary repetition as $\mathsf{Nat}^{\uparrow}$ seed

Definition 26.11 (Unary-source specification)**.** *The unary-source specification is the class of histories generated by repeated extension with* b1 *from the empty history. It is not called a natural number object internally. It is a source pattern of repeated emission.*

Lean4 ✓ `BEDC.FKernel.Unary.UnarySourceSpec induction`

Definition 26.12 (Unary pattern)**.** *The unary pattern is the stable behavior of repeated* b1*-extension together with the ability to distinguish the empty history from an extended unary history by the generated history constructors.*

Lean4 ✓ `BEDC.FKernel.Unary.UnaryPatternSpec step`
↪ variant `BEDC.FKernel.Unary.unary pattern induction with discrimination`

Definition 26.13 ($\mathsf{Nat}^{\uparrow}$ certificate seed)**.** *The name* $\mathsf{Nat}^{\uparrow}$ *is licensed only when the following fields are supplied:*

1. *source: unary histories;*
2. *pattern: repeated* b1*-extension;*
3. *classifier: internal history-sameness restricted to unary histories;*
4. *stability: closure under further* b1*-extension and reuse as a counting interface;*

5. *ledger: the originating unary history behind each visible counting token.*

↪variant BEDC.FKernel.Unary.nat up interface seed

Corollary 26.14 ($\mathsf{Nat}^{\uparrow}$ is not primitive)**.** *The interface* $\mathsf{Nat}^{\uparrow}$ *is licensed by a naming certificate over unary histories. It is not an internal primitive of the kernel.*

Lean4 ✓ BEDC.FKernel.Unary.nat up certificate seed not primitive

26.3 Continuation as $\mathsf{Add}^{\uparrow}$ seed

Definition 26.15 (Continuation pattern)**.** *The continuation pattern is the relation* $\mathsf{Cont}(h, k, r)$*, read as: the history* h *continued by the history* k *yields result* r*, up to internal history-sameness.*

Lean4 ✓ BEDC.FKernel.Cont.continuation pattern stability fields

Definition 26.16 ($\mathsf{Add}^{\uparrow}$ certificate seed)**.** *The name* $\mathsf{Add}^{\uparrow}$ *is licensed only when the following fields are supplied:*

1. *source: pairs of unary histories;*
2. *pattern: relational continuation;*
3. *classifier: internal history-sameness of continuation results;*
4. *stability: left unit, right unit, determinacy up to* hsame*, and associativity up to* hsame*;*
5. *ledger: the pair of source unary histories and the continuation witness.*

Theorem 26.17 (Addition-like behavior is a certificate, not an operation)**.** *The interface* $\mathsf{Add}^{\uparrow}$ *is not admitted by writing a binary operation symbol. It is admitted by a naming certificate whose stability field contains unit and associativity witnesses for relational continuation.*

↪variant BEDC.FKernel.Unary.add up licensed not primitive

Proof. The kernel contains histories and the relation Cont. It does not contain a primitive binary operation called addition. If continuation is shown to be stable on the unary source pattern and its behavior descends to the unary classification, then a derived interface may be named $\mathsf{Add}^{\uparrow}$. The name records the evidence; it does not introduce a primitive operation. □

26.4 Descent certificate for stable transformations

Definition 26.18 (Descent certificate)**.** *A descent certificate for a transformation pattern states that the pattern respects the classifier on the source and the classifier on the target. In explanatory notation, if* a *and* a' *are source-same, then their transformed outputs are target-same.*

Lean4 (definition) BEDC.FKernel.NameCert.DescentCertificate
Lean4 ✓ BEDC.FKernel.NameCert.StableTransformation descentCertificate exists

Theorem 26.19 (Descent certificate respects source sameness)**.** *If a descent certificate maps a source classifier to a target classifier, then source-same inputs have target-same mapped outputs.*

Lean4 ✓ `BEDC.FKernel.NameCert.descentCertificate respects`
↪ variant `BEDC.FKernel.NameCert.stableTransformation descent respects and ledger`
↪ variant `BEDC.FKernel.NameCert.StableTransformation descent certificate and respects`
↪ variant `BEDC.FKernel.NameCert.StableTransformation descentCertificate respects projection`
↪ variant `BEDC.FKernel.NameCert.descentCertificate respects witness`
↪ variant `BEDC.FKernel.NameCert.descentCertificate respects pair with map`

Proof. Unpack the descent certificate and apply its respect field to the source-sameness witness. □

Theorem 26.20 (Stable transformations descend to packages)**.** *If visible source package tokens introduce p and q, and the source classifier identifies them, then any supplied stability witness that respects the package classifier transports that identification to the target package classifier.*

Lean4 ✓ `BEDC.FKernel.NameCert.stableTransformation descends to packages`
↪ variant `BEDC.FKernel.Package.stable transformations descend to packages`

Proof. Apply the supplied descent witness to the two visible token introductions and the source package-sameness proof. □

Corollary 26.21 (Function-like interfaces require descent)**.** *A function-like derived interface cannot be licensed by a raw transformation alone. It requires a descent certificate.*

Lean4 ✓ `BEDC.FKernel.NameCert.function like interfaces require descent`

Corollary 26.22 (Function-like interfaces are derived)**.** *A function-like interface obtained from a stable transformation carries a descent certificate, a respect proof, and a ledger witness.*

Lean4 ✓ `BEDC.FKernel.NameCert.function like interfaces are derived`
↪ variant `BEDC.FKernel.NameCert.stableTransformation descent and ledger`
↪ variant `BEDC.FKernel.NameCert.stableTransformation respects and ledger`
↪ variant `BEDC.FKernel.NameCert.stableTransformation respects with descent certificate`
↪ variant `BEDC.FKernel.NameCert.StableTransformation descent respects ledger pair`
↪ variant `BEDC.FKernel.NameCert.function like interfaces derived core`
↪ variant `BEDC.FKernel.NameCert.function like interface seed is derived`

Theorem 26.23 (Descent-certificate composition for stable transformations)**.** *Let $S, R, Q : \mathsf{BHist} \to \mathsf{Prop}$ be source, intermediate, and target carrier specifications, and let*

$$\sim_S, \sim_R, \sim_Q : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$$

be their classifiers. Assume the classifier roles are licensed by

$$\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S), \qquad \mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R), \qquad \mathsf{cert}_Q : \mathsf{NameCert}(Q, \sim_Q).$$

Let $T, U : \mathsf{BHist} \to \mathsf{BHist}$ have carrier-preservation fields

$$\rho_T : \forall h.\ S(h) \to R(T(h)) \qquad \text{and} \qquad \rho_U : \forall r.\ R(r) \to Q(U(r)).$$

Assume descent certificates

$$\delta_T : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_R)$$

and

$$\delta_U : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_R, \sim_Q)$$

whose map fields are T and U. Then the composite map $U \circ T$ preserves carriers by

$$\rho_{U \circ T}(h, Sh) = \rho_U(T(h), \rho_T(h, Sh))$$

and carries a descent certificate

$$\delta_{U \circ T} : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_Q)$$

with map field $h \mapsto U(T(h))$. Thus any source-classifier witness $h \sim_S k$ gives the target-classifier witness $U(T(h)) \sim_Q U(T(k))$.

Moreover, if

$$\tau_T : \mathsf{StableTransformation}(\mathsf{BHist}, \mathsf{BHist}, L_T, \sim_S, \sim_R)$$

and

$$\tau_U : \mathsf{StableTransformation}(\mathsf{BHist}, \mathsf{BHist}, L_U, \sim_R, \sim_Q)$$

have map fields T and U, then their composite is a stable transformation

$$\mathsf{StableTransformation}(\mathsf{BHist}, \mathsf{BHist}, L_T \times L_U, \sim_S, \sim_Q)$$

whose ledger witness is the ordered pair of the two input ledger witnesses. In gap-ledger cases this ordered pair is the witness-level form of the composite ledger policy of Definition 25.7 and Definition 8.12.

Lean4 ✓ `BEDC.FKernel.NameCert.StableTransformation descentCertificate composition`

Proof. The three displayed naming certificates place S, R, and Q in the typed certificate template of Definition 26.1; the descent argument uses the respect fields of the two descent certificates.

Define $\delta_{U \circ T}$ with map field $h \mapsto U(T(h))$. Given $h \sim_S k$, the respect field of δ_T gives

$$T(h) \sim_R T(k).$$

Applying the respect field of δ_U to this intermediate classifier witness gives

$$U(T(h)) \sim_Q U(T(k)).$$

By Definition 26.18, this is exactly the respect field required for a descent certificate from $\sim_S$ to $\sim_Q$. Carrier preservation is the displayed composite of the two carrier-preservation fields.

For the stable-transformation statement, use the same composite map. If $h \sim_S k$, the respect field of τ_T gives $T(h) \sim_R T(k)$, and the respect field of τ_U gives $U(T(h)) \sim_Q U(T(k))$. The ledger field of Definition 10.5 supplies witnesses $\ell_T : L_T$ and $\ell_U : L_U$; pair them as $(\ell_T, \ell_U) : L_T \times L_U$. This gives the ledger field for the composite stable transformation. When the two ledgers are gap ledgers, the same pair records the first-stage ledger and the second-stage ledger pulled back along the intermediate interface, which is the composite ledger policy cited above. □

Theorem 26.24 (NameCert composition for stable transformations)**.** *Let $S, R, Q, P_S, P_R, P_Q, L_S, L_R, L_Q :$ $\mathsf{BHist} \to \mathsf{Prop}$, and let*

$$\sim_S, \sim_R, \sim_Q : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$$

be classifier specifications. Assume semantic carrier certificates

$$\mathcal{C}_S : \mathsf{SemanticNameCert}(S, P_S, L_S, \sim_S), \qquad \mathcal{C}_R : \mathsf{SemanticNameCert}(R, P_R, L_R, \sim_R), \qquad \mathcal{C}_Q : \mathsf{SemanticNameCert}(Q$$

Let $T, U : \mathsf{BHist} \to \mathsf{BHist}$ *have carrier-preservation fields*

$$\rho_T : \forall h.\ S(h) \to R(T(h)), \qquad \rho_U : \forall r.\ R(r) \to Q(U(r)).$$

Let Λ_T *and* Λ_U *be ledger witness types, and let*

$$\tau_T : \mathsf{StableTransformation}(\mathsf{BHist}, \mathsf{BHist}, \Lambda_T, \sim_S, \sim_R)$$

and

$$\tau_U : \mathsf{StableTransformation}(\mathsf{BHist}, \mathsf{BHist}, \Lambda_U, \sim_R, \sim_Q)$$

be stable transformations with map fields T *and* U*. Let*

$$M_T, M_U : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$$

be their two-stage ledger predicates, with ledger soundness fields

$$\lambda_T : \forall h.\ S(h) \to M_T(h, T(h)), \qquad \lambda_U : \forall r.\ R(r) \to M_U(r, U(r)).$$

Define source-level composite pattern and ledger predicates by

$$\mathsf{CompStab}_{T,U}(h) \iff Q(U(T(h))) \wedge \forall k,\ h \sim_S k \to U(T(h)) \sim_Q U(T(k))$$

and

$$\mathsf{CompLedger}_{T,U}(h) \iff \exists r,\ r = T(h) \wedge M_T(h, r) \wedge M_U(r, U(r)).$$

Then

$$\mathsf{SemanticNameCert}(S, \mathsf{CompStab}_{T,U}, \mathsf{CompLedger}_{T,U}, \sim_S)$$

holds. Its stability component is the composite descent witness for $U \circ T$*, and its ledger component is the composite ledger policy.*

Proof. Use the core field of $\mathcal{C}_S$ as the core field of the displayed semantic certificate. By Definition 9.7, this is exactly the carrier-and-classifier certificate for S and $\sim_S$. The core fields of $\mathcal{C}_R$ and $\mathcal{C}_Q$ place the intermediate and target classifiers in the same typed certificate discipline of Definition 26.1.

For pattern soundness, fix h with $Sh : S(h)$. Carrier preservation gives $\rho_T(h, Sh) : R(T(h))$, and then $\rho_U(T(h), \rho_T(h, Sh)) : Q(U(T(h)))$. If $h \sim_S k$, the respect field of τ_T gives $T(h) \sim_R T(k)$. Applying the respect field of τ_U gives $U(T(h)) \sim_Q U(T(k))$. Hence $\mathsf{CompStab}_{T,U}(h)$.

For ledger soundness at the same source history, choose the intermediate witness $r = T(h)$. The field λ_T gives $M_T(h, T(h))$. The field ρ_T gives $R(T(h))$, so λ_U gives $M_U(T(h), U(T(h)))$. These witnesses prove $\mathsf{CompLedger}_{T,U}(h)$.

The respect calculation is the descent certificate required by Definition 26.18: a source-classifier witness is first transported along T and then along U. By Definition 9.4, this fills the $\mathsf{DescentCert}$ and $\mathsf{CompositionCert}$ modes of the stability component. The ledger proof records the first-stage ledger together with the second-stage ledger pulled back along the intermediate witness. In gap-ledger cases this is the two-step witness of Definition 25.7, and Definition 8.12 identifies it as the composite ledger policy. □

Theorem 26.25 (Semantic NameCert classifier replacement)**.** *Let* $S, P, L : \mathsf{BHist} \to \mathsf{Prop}$ *and let* $C, C' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be classifier specifications. Assume*

$$\mathcal{N} : \mathsf{SemanticNameCert}(S, P, L, C)$$

and assume classifier comparison fields

$$\alpha : \forall h, k,\ C'(h, k) \to C(h, k), \qquad \beta : \forall h, k,\ C(h, k) \to C'(h, k).$$

Then

$$\mathsf{SemanticNameCert}(S, P, L, C')$$

holds. The resulting certificate also has the same pattern-ledger transport consequence: whenever $C'(h, k)$ *and* $S(h)$ *hold, both* $P(k)$ *and* $L(k)$ *hold.*

Proof. By Definition 26.1, a semantic naming certificate separates its core carrier-classifier certificate from its pattern and ledger soundness fields. Keep the source, pattern, and ledger soundness fields of $\mathcal{N}$ unchanged. It remains only to replace the core classifier certificate.

The original core field is a $\mathsf{NameCert}(S, C)$ in the sense of Definition 9.7. Define the new core field over C' as follows. The inhabited source witness is the same one carried by the original core. For reflexivity at a source history h, apply the old reflexivity field to obtain $C(h, h)$ and then apply β. For symmetry, convert a witness $C'(h, k)$ to $C(h, k)$ by α, apply the old symmetry field, and convert the result back by β. For transitivity, convert both C' witnesses to C by α, apply the old transitivity field, and convert the composite classifier witness back by β. For carrier transport, convert $C'(h, k)$ to $C(h, k)$ by α and then use the old carrier-transport field to carry $S(h)$ to $S(k)$.

These four classifier laws together with the unchanged source witness form $\mathsf{NameCert}(S, C')$. Combining this new core field with the unchanged pattern and ledger soundness fields gives $\mathsf{SemanticNameCert}(S, P, L, C')$. Finally, if $C'(h, k)$ and $S(h)$ hold, the new carrier-transport field gives $S(k)$, and the unchanged pattern and ledger soundness fields give $P(k)$ and $L(k)$. This is the same transport consequence recorded in Definition 26.3 for the replacement classifier. □

Chapter 27

Machine-Checkable Milestones

This chapter converts the proof-sprint plan into a milestone schedule. It is implementation-facing, but it is written without depending on any particular proof assistant syntax.

27.1 Milestone A: history kernel

Protocol 27.1 (Milestone A)**.** *Milestone A is complete when the following facts have been checked:*

1. msame *is reflexive, symmetric, and transitive.*
2. *raw marks satisfy no-confusion:* b0 *and* b1 *are not cross-identified.*
3. hsame *is reflexive, symmetric, and transitive.*
4. *history constructors satisfy no-confusion.*
5. Ext *is deterministic up to* hsame*.*
6. Cont *is deterministic up to* hsame*.*
7. Cont *has left and right units.*
8. Cont *is associative up to* hsame*.*

27.2 Milestone B: signature kernel

Protocol 27.2 (Milestone B)**.** *Milestone B is complete when the following facts have been checked:*

1. *bundles are generated by* Bnil *and* Bcons*;*
2. *signature events are generated by the empty and cons signature rules;*
3. *under* AskPolicy(Π, D)*, signature generation is deterministic up to* hsame*;*
4. *under* AskPolicy(Π, D)*,* sameSig *is reflexive, symmetric, and transitive on* D*.*

```
Lean4 ✓  BEDC.FKernel.Settled.settledKernelCriterion milestoneB signature kernel
  ↪ variant  BEDC.FKernel.Settled.settledKernelCriterion signature determinacy projection
  ↪ variant  BEDC.FKernel.Settled.settledKernelCriterion sameSig equivalence projection
  ↪ variant  BEDC.FKernel.Settled.settledKernelCriterion bundle generation projection
  ↪ variant  BEDC.FKernel.Sig.sigRel generation constructor pair
```

27.3 Milestone C: package and gap kernel

Protocol 27.3 (Milestone C)**.** *Milestone C is complete when the following facts have been checked:*

1. *package policies provide existence, extensionality, and grounding;*
2. *gap policies provide coverage, separation, and signature determination;*
3. *packages classify by signatures;*
4. *gap ledgers are determined by signatures and package tokens;*
5. *composite gap ledgers satisfy composite coverage.*

Lean4 ✓ `BEDC.FKernel.Settled.settledKernelCriterion milestoneC package gap kernel`

27.4 Milestone D: first derived interface seeds

Protocol 27.4 (Milestone D)**.** *Milestone D is complete when the following certificates have been constructed:*

1. *the unary repetition certificate for* $\mathsf{Nat}^{\uparrow}$*;*
2. *the continuation certificate for* $\mathsf{Add}^{\uparrow}$*;*
3. *a descent-certificate template for function-like derived interfaces.*

27.5 Settled-kernel criterion

Definition 27.5 (Settled-kernel criterion)**.** *The finite kernel becomes a settled basis for further development once Milestones A, B, and C have been checked in a concrete formal environment and the statements of Milestone D are internally consistent with the checked kernel.*

Lean4 (definition) `BEDC.FKernel.Settled.SettledKernelCriterion`

Lean4 ✓ `BEDC.FKernel.Settled.settledKernelCriterion from current targets`

↪ variant `BEDC.FKernel.Settled.settledKernelCriterion globalize exactness projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion ext cont projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion composite gap projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion history kernel projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion package gap projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion package gap kernel projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion gap coverage projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion namecert projection`
↪ variant `BEDC.FKernel.Settled.settledKernelCriterion function like descent projection`

Remark 27.6. *Until then, large derived constructions such as real certificates, phase geometry, fold normalization, and arithmetic beyond addition should remain non-blocking appendices governed by explicit naming-certificate obligations.*

Part V

Concrete Instances

Chapter 28

A Concrete Naming Certificate for $\mathsf{Nat}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for $\mathsf{Nat}^{\uparrow}$ at the paper-proof level.

The unary carrier and successor rows recorded here are consumed by chapter 29 for continuation-based addition and by chapter 31 for signed-pair balancing. $\mathsf{Add}^{\uparrow}$ reads addition as continuation over $\mathsf{Nat}^{\uparrow}$ unary histories, while $\mathsf{Int}^{\uparrow}$ reads signed pairs by balancing two $\mathsf{Nat}^{\uparrow}$ source spines.

The same unary rows also feed later finite schedule consumers. In chapter 7585, the unary depth ledger is read only as bounded repetition of the existing $\mathsf{Nat}^{\uparrow}$ empty and one-extension rows, so bar-window coverage does not add a separate natural-number primitive.

28.1 Unary histories

Definition 28.1 (Unary history judgment)**.** *The unary history judgment* $\mathsf{Unary}(h)$ *is generated by:*

1. $\mathsf{Unary}(\mathsf{e})$*;*
2. *if* $\mathsf{Unary}(h)$*, then* $\mathsf{Unary}(\mathsf{E1}(h))$*.*

No rule generates $\mathsf{Unary}(\mathsf{E0}(h))$*.*

```
Lean4 ✓  BEDC.FKernel.Unary.unary history judgment generators
  ↪ variant  BEDC.FKernel.Unary.unary history judgment
  ↪ variant  BEDC.FKernel.Unary.unary repetition histories generators
  ↪ variant  BEDC.FKernel.Unary.unary empty
  ↪ variant  BEDC.FKernel.Unary.unary history cases
  ↪ variant  BEDC.FKernel.Unary.unary e1 inversion
  ↪ variant  BEDC.FKernel.Unary.unary history e1 iff
  ↪ variant  BEDC.FKernel.Unary.unary history induction
  ↪ variant  BEDC.FKernel.Unary.unary histories induction from generators
  ↪ variant  BEDC.FKernel.Unary.unary history closed generation spine
  ↪ variant  BEDC.FKernel.Unary.unary double e1 closed
  ↪ variant  BEDC.FKernel.Unary.unary history e0 iff false
  ↪ variant  BEDC.FKernel.Unary.unary history not e0 head
  ↪ variant  BEDC.FKernel.Unary.unary append e0 left absurd
```

Lemma 28.2 (Unary closure under one-extension)**.** *If* $\mathsf{Unary}(h)$*, then* $\mathsf{Unary}(\mathsf{E1}(h))$*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary closure under one extension`

Proof. This is the second generator of the unary judgment. □

Lemma 28.3 (Unary no-zero-extension)**.** *No judgment* $\mathsf{Unary}(\mathsf{E0}(h))$ *is generated.*

Lean4 ✓ `BEDC.FKernel.Unary.unary no zero extension`
↪variant `BEDC.FKernel.Unary.unary history hsame zero absurd`

Proof. By closed generation for Unary, the only extension rule uses $\mathsf{E1}$. No $\mathsf{E0}$-extension constructor is displayed. □

Theorem 28.4 (Nat unary terminal empty classifier)**.** *Any accepted* $\mathsf{Nat}^{\uparrow}$ *unary-history read whose displayed continuation tail is* e *classifies to the empty endpoint route of the unary carrier. Conversely, any nonempty accepted* $\mathsf{Nat}^{\uparrow}$ *read must arise through repeated* $\mathsf{E1}$ *extension from the empty endpoint, never through an* $\mathsf{E0}$ *branch. Downstream* $\mathsf{Add}^{\uparrow}$ *source-spine reads therefore use this empty classifier as the base case before continuation tails are consumed as addition rows.*

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryTerminal empty classifier`

Proof. For the terminal case, unfold the unary history judgment of Definition 28.1. The displayed empty tail is the endpoint route, so the read exposes the base generator $\mathsf{Unary}(\mathsf{e})$ and no predecessor row.

For a nonempty accepted read, use closed generation of the same judgment. The only displayed extension rule is Definition 28.2; Definition 28.3 removes every $\mathsf{E0}$-headed branch. Hence each nonempty accepted history is obtained by iterating $\mathsf{E1}$ from e.

The addition source spine of Definition 28.7 consumes exactly this dichotomy. Its base continuation is the empty classifier route, and each nonempty continuation step is a one-extension row available to the $\mathsf{Add}^{\uparrow}$ certificate. □

Theorem 28.5 (Unary prefix totality)**.** *If h and k are unary histories, then either h continues by a unary tail to k, or k continues by a unary tail to h.*

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryPrefix total`

Proof. Induct on the unary generation of h. The empty case uses k itself as the tail. In the $\mathsf{E1}$ case, split k; the empty and $\mathsf{E0}$ cases are immediate by the continuation unit and unary no-zero rule. When $k = \mathsf{E1}(k')$, apply the induction hypothesis to h' and k', then lift the resulting unary tail across the common $\mathsf{E1}$ boundary by the unary one-extension append law. □

Theorem 28.6 (Unary prefix directed common upper)**.** *If h and k are unary histories, then there is a unary history u such that both h and k continue to u by unary tails.*

Proof. Apply unary prefix totality. If h continues to k, choose k as the common upper and use the empty tail on the k-side. If k continues to h, choose h as the common upper and use the empty tail on the h-side. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryPrefix directed common upper`

Theorem 28.7 (Nat source spine for addition)**.** *Every* $\mathsf{Add}^{\uparrow}$ *continuation read over unary histories factors through the* $\mathsf{Nat}^{\uparrow}$ *unary carrier before it is consumed as addition. Concretely, the source side must display the unary judgment of Definition 28.1, closure under one-extension from Definition 28.2, prefix totality from Definition 28.5, and the common-upper row of Definition 28.6; only after those rows are present may the continuation be read by chapter 29 as an* $\mathsf{Add}^{\uparrow}$ *operation.*

Proof. Read the unary source first. The judgment of Definition 28.1 supplies exactly the empty history and one-extension histories, while Definition 28.2 keeps the carrier closed under the successor step used by addition.

Next use Definition 28.5 and Definition 28.6. They make any two unary inputs comparable through unary tails and then place them under a displayed unary common upper, so the continuation has a source spine before any addition row is consumed.

Finally pass that displayed spine to chapter 29. Since $\mathsf{Add}^{\uparrow}$ reads addition as continuation over $\mathsf{Nat}^{\uparrow}$ unary histories, any continuation lacking the unary carrier, closure, prefix totality, and common-upper rows is not an addition read from the $\mathsf{Nat}^{\uparrow}$ source. □

Lean4 ✓ `BEDC.Derived.NatUp.NatAddSourceSpine forward link`

Theorem 28.8 (Nat unary chain for addition and integer balance)**.** *Every* $\mathsf{Add}^{\uparrow}$ *continuation read and every* $\mathsf{Int}^{\uparrow}$ *signed-pair balance read consumes the same* $\mathsf{Nat}^{\uparrow}$ *unary source chain before it can be interpreted as addition or signed balancing. The shared chain consists of the unary carrier judgment of Definition 28.1, closure under one-extension from Definition 28.2, prefix totality from Definition 28.5, and the directed common-upper row of Definition 28.6.*

Lean4 ✓ `BEDC.Derived.NatUp.NatAddIntUnaryChainOrphanLink`

Proof. For an $\mathsf{Add}^{\uparrow}$ read, apply Definition 28.7. The continuation may be consumed by chapter 29 only after the unary carrier, one-extension closure, prefix totality, and common-upper rows have been displayed.

For an $\mathsf{Int}^{\uparrow}$ read, use the signed-pair presentation of chapter 31. Each positive and negative side is a $\mathsf{Nat}^{\uparrow}$ unary component, so the same carrier judgment and closure rows license the two components before balancing is inspected.

The prefix totality and common-upper rows are still shared after the signed-pair packaging. Thus addition and signed balancing both enter through the same finite unary chain, and their difference is only the downstream consumer that reads the displayed $\mathsf{Nat}^{\uparrow}$ rows. □

Theorem 28.9 (Nat unary dependency route for addition and integer balance)**.** *Every downstream* $\mathsf{Add}^{\uparrow}$ *continuation read and every* $\mathsf{Int}^{\uparrow}$ *signed-pair balancing read must first expose the* $\mathsf{Nat}^{\uparrow}$ *unary carrier, the one-extension closure row, the prefix-total row, and the directed common-upper row. Only after those rows are displayed may the route cite chapter 29 and chapter 31 as the sibling consumers of the same finite unary source.*

Proof. Start with the Nat source spine theorem Definition 28.7. It forces every addition continuation to display the unary judgment, one-extension closure, prefix totality, and common-upper rows before the $\mathsf{Add}^{\uparrow}$ operation is read.

Now apply the add-int unary chain theorem Definition 28.8. The same displayed carrier and closure rows are consumed by the positive and negative sides of an $\mathsf{Int}^{\uparrow}$ signed pair before balancing is inspected.

This dependency is substantive rather than self-reference because the final readers are the sibling chapters chapter 29 and chapter 31. The $\mathsf{Nat}^{\uparrow}$ chapter supplies the unary source rows; the sibling certificates consume those rows as addition and signed-pair balance data. □

Definition 28.10 (Strict unary prefix)**.** *A strict unary prefix from h to k is a unary tail t such that t is not* e *and* $\mathsf{Cont}(h, t, k)$*.*

Lean4 (definition) `BEDC.Derived.NatUp.NatUnaryStrictPrefix`

Lemma 28.11 (Unary continuation tail cases)**.** *If a unary tail t displays* $\mathsf{Cont}(h, t, k)$*, then either* $h\mathsf{hsame}k$ *or h is a strict unary prefix of k.*

Proof. Split the unary tail. The empty tail gives the right-unit continuation and therefore endpoint history-sameness. A zero-headed tail contradicts unary generation. A one-headed tail is nonempty, so the displayed continuation is exactly the witness for $\mathsf{NatUnaryStrictPrefix}(h, k)$. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryPrefix cont tail cases`

Theorem 28.12 (Unary prefix trichotomy into sameness or strict prefix)**.** *If h and k are unary histories, then either* $h\mathsf{hsame}k$*, or h is a strict unary prefix of k, or k is a strict unary prefix of h.*

Proof. Apply unary prefix totality to obtain a unary continuation tail in one direction. If the tail is e, the continuation is the right-unit continuation and the endpoints are history-same. A zero-headed tail contradicts unary generation. A one-headed tail is nonempty, so the same continuation witness gives the corresponding strict unary prefix branch. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryPrefix trichotomy hsame strict`

Lemma 28.13 (Strict unary prefix one-step)**.** *If h is unary, then h is a strict unary prefix of* $\mathsf{E1}(h)$*.*

Proof. Use the tail $\mathsf{E1}(\mathsf{e})$. It is unary, it is not e, and $\mathsf{Cont}(h, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(h))$ holds by the continuation constructor. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix one step`

Theorem 28.14 (Strict unary prefix asymmetry)**.** *No pair of histories can strictly prefix each other through nonempty unary tails.*

Proof. Unpack the two strict witnesses. Mutual continuation makes the endpoints history-same. The right-unit continuation at the source then transports to the forward target, so left cancellation identifies the forward tail with e, contradicting its strictness. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix asymm`

Theorem 28.15 (Strict unary prefix has nonempty target)**.** *No history can be a strict unary prefix of* e*.*

Proof. Unpack the strict-prefix witness. The continuation into e forces the tail to be e, contradicting the nonempty-tail condition. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix empty right absurd`

Lemma 28.16 (Strict unary prefix tail endpoint collapse is impossible)**.** *If a unary tail t is nonempty,* $\mathsf{Cont}(h, t, k)$*, and* $h\mathsf{hsame}k$*, then the displayed continuation is impossible.*

Proof. Transport the right-unit continuation $\mathsf{Cont}(h, \mathsf{e}, h)$ across the endpoint identification $h\mathsf{hsame}k$, giving $\mathsf{Cont}(h, \mathsf{e}, k)$. Left cancellation against the supplied continuation $\mathsf{Cont}(h, t, k)$ identifies t with e, contradicting the strict tail witness. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix tail endpoint hsame absurd`

Theorem 28.17 (Strict unary prefix one-boundary inversion)**.** *If* $\mathsf{E1}(h)$ *is a strict unary prefix of* $\mathsf{E1}(k)$*, then h is a strict unary prefix of k.*

Proof. Unpack the strict witness tail. The empty tail contradicts strictness, and an $\mathsf{E0}$-headed tail contradicts unary generation. For an $\mathsf{E1}(t)$ tail, invert the result boundary of the continuation and commute the visible $\mathsf{E1}$ across the unary tail t, yielding the strict witness $\mathsf{E1}(t)$ from h to k. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix e1 inversion`

Lemma 28.18 (Strict unary prefix one-boundary endpoint collapse is impossible)**.** *If* $\mathsf{E1}(h)$ *is a strict unary prefix of* $\mathsf{E1}(k)$ *and* $h\mathsf{hsame}k$*, then the displayed strict prefix is impossible.*

Proof. Apply one-boundary inversion to obtain a strict unary prefix from h to k. The strict witness tail together with the endpoint identification $h\mathsf{hsame}k$ contradicts the tail endpoint collapse lemma. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix e1 endpoint hsame absurd`

Theorem 28.19 (Strict unary prefix one-boundary congruence)**.** *If* h *is a strict unary prefix of* k*, then* $\mathsf{E1}(h)$ *is a strict unary prefix of* $\mathsf{E1}(k)$*.*

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix e1 congr`

Proof. Unpack the nonempty unary tail witnessing the strict prefix from h to k. The same tail remains unary and nonempty. Commuting append across the visible $\mathsf{E1}$ boundary re-expresses the continuation as one from $\mathsf{E1}(h)$ to $\mathsf{E1}(k)$. □

Theorem 28.20 (Strict unary prefix zero-boundary inversion)**.** *If* $\mathsf{E0}(h)$ *is a strict unary prefix of* $\mathsf{E0}(k)$*, then* h *is a strict unary prefix of* k*.*

Proof. Unpack the strict witness tail. The empty tail contradicts strictness, an $\mathsf{E0}$-headed tail contradicts unary generation, and an $\mathsf{E1}$-headed tail would force a zero-headed continuation result to be one-headed. Hence the premise has no inhabitant, so the displayed implication follows. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix e0 inversion`

Theorem 28.21 (Strict unary prefix append-tail transitivity)**.** *If* h *continues by a nonempty unary tail to* k*, and* k *continues by a unary tail to* l*, then* h *is a strict unary prefix of* l*. The witness tail is* $\mathsf{append}(t_1, t_2)$*, and it also gives* $\mathsf{Cont}(h, \mathsf{append}(t_1, t_2), l)$*.*

Proof. Invert the two continuation witnesses and reassociate append. Unary closure under append gives a unary composite tail, and append empty-result exactness shows that the composite tail cannot be empty because its left component is nonempty. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix append tail trans`

Lemma 28.22 (Strict unary prefix target is unary)**.** *If* h *is unary and* h *is a strict unary prefix of* k*, then* k *is unary.*

Proof. Unpack the strict-prefix witness as a unary tail t with $\mathsf{Cont}(h, t, k)$. Unary continuation closure transports unary generation from h across t to the target k. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix target unary`

Theorem 28.23 (Strict unary prefix composite-tail transitivity)**.** *If* h *is a strict unary prefix of* k *and* k *is a strict unary prefix of* l*, then the two nonempty unary tails compose into a concrete tail from* h *to* l*. This composite tail witnesses both* $\mathsf{Cont}(h, t, l)$ *and the strict unary prefix relation from* h *to* l*.*

Proof. Unpack the two strict-prefix witnesses. Append their unary tails and apply append-tail transitivity. Unary append closure gives the composite tail carrier, and append empty-result exactness shows that the composite tail is nonempty because its left component is nonempty. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix trans composite tail`

Theorem 28.24 (Strict unary prefix continuation transport)**.** *If a nonempty unary tail carries h to k, and $h$hsameh' and $k$hsamek', then h' is a strict unary prefix of k'.*

Proof. Reuse the same tail. The continuation component transports across the two endpoint hsame witnesses, while unary generation and nonemptiness of the tail are unchanged. □

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryStrictPrefix cont hsame transport`

28.2 The certificate

Definition 28.25 (Unary source specification)**.** *The source specification* UnarySource *is the subdiscipline of generated histories governed by* Unary(h)*.*

Lean4 (definition) `BEDC.FKernel.Unary.UnarySourceSpec`
Lean4 ✓ `BEDC.FKernel.Unary.UnarySourceSpec iff unaryHistory`

Definition 28.26 (Unary pattern specification)**.** *The pattern specification* UnaryPattern *is repeated generation by the one-extension constructor* E1 *from* e*.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryPatternSpec`
Lean4 ✓ `BEDC.FKernel.Unary.unary history empty or e1 tail`
↪ variant `BEDC.FKernel.Unary.UnaryPatternSpec fields`
↪ variant `BEDC.FKernel.Unary.UnaryPatternSpec nonempty`

Definition 28.27 (Unary classifier specification)**.** *The classifier specification* UnaryClassifier *identifies unary histories only through the internal history-sameness judgment restricted to unary histories.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryClassifierSpec`
Lean4 ✓ `BEDC.FKernel.Unary.UnaryClassifierSpec iff fields`

Definition 28.28 (Unary stability certificate)**.** *The unary stability certificate* UnaryClosureCert *consists of:*

1. *the base derivation* Unary(e)*;*
2. *closure under* E1*;*
3. *no-zero-extension, recording that* E0 *is not part of the unary generation discipline.*

Lean4 ✓ `BEDC.FKernel.Unary.unary stability certificate fields`
↪ variant `BEDC.FKernel.Unary.unary stability certificate base step nozero pair`
↪ variant `BEDC.FKernel.Unary.unary stability certificate no zero projection`

Definition 28.29 (Unary ledger policy)**.** *The unary ledger policy* UnaryLedger *records the source unary history behind any visible unary class. It does not erase the emitted history; it licenses a derived interface while retaining provenance.*

Lean4 (definition) `BEDC.FKernel.Unary.UnaryLedgerPolicy`

Theorem 28.30 (Concrete $\mathsf{Nat}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}} := (\mathsf{UnarySource}, \mathsf{UnaryPattern}, \mathsf{UnaryClassifier}, \mathsf{UnaryClosureCert}, \mathsf{UnaryLedger})$$

forms a valid naming certificate for the derived interface $\mathsf{Nat}^{\uparrow}$*.*

Proof. The source, pattern, classifier, stability, and ledger fields are explicitly supplied by the preceding definitions. The stability field is justified by unary base generation, unary closure under $\mathsf{E1}$, and unary no-zero-extension. Therefore the naming protocol is satisfied. □

Lean4 ✓ `BEDC.FKernel.Unary.nat up name certificate`

Theorem 28.31 (Semantic $\mathsf{Nat}^{\uparrow}$ naming certificate)**.** *The* $\mathsf{Nat}^{\uparrow}$ *naming certificate lifts to a semantic certificate whose source, pattern, and ledger predicates all preserve the same unary-history witness.*

Proof. The core certificate is the concrete unary $\mathsf{NameCert}$. Pattern soundness and ledger soundness are both identity preservation of the unary history witness, so no information is erased between the source predicate and the two semantic predicates. □

Lean4 ✓ `BEDC.FKernel.Unary.nat up semantic name certificate`

Corollary 28.32 ($\mathsf{Nat}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{Nat}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$*. It is not a primitive object of the finite kernel.*

Lean4 ✓ `BEDC.FKernel.Unary.nat up licensed not primitive`

Theorem 28.33 (NatUp unary standard bridge)**.** *The unary-history certificate for* $\mathsf{Nat}^{\uparrow}$ *carries a conservative standard bridge to finite unary words and their natural-number length readback. The bridge sends* e *to zero, sends* $\mathsf{E1}(h)$ *to the successor of the readback of* h*, admits no* $\mathsf{E0}$ *branch, identifies* $h\mathsf{hsame}k$ *on unary histories exactly when the external readback lengths are equal, and sends a unary continuation* $\mathsf{Cont}(h, t, k)$ *to addition of the readback length of* t *to the readback length of* h*.*

Proof. Define the decode field by unary-generation induction. The base generator reads e as zero. The one-extension generator reads $\mathsf{E1}(h)$ as the successor of the already decoded tail. The no-zero-extension lemma excludes an $\mathsf{E0}$ case, so every decoded value is obtained from a finite unary word and no other BEDC history is admitted by the bridge carrier.

The encode field sends an external finite unary word back to its BEDC unary history by the same two constructors. Induction on the word gives the decode-encode round trip, while induction on the unary-history generator gives the encode-decode round trip up to $h\mathsf{hsame}k$. The semantic NameCert theorem supplies the source, pattern, and ledger preservation needed to keep these witnesses inside the licensed $\mathsf{Nat}^{\uparrow}$ surface.

For exactness, use strict-prefix and continuation transport. If two unary histories are $h\mathsf{hsame}k$, the readback induction transports the same length across the endpoint identification. Conversely, if the readback lengths agree, unary prefix trichotomy leaves only the sameness branch: either strict-prefix branch would add a nonempty unary tail and hence a positive successor length on one side. Thus hsame matches equality of external readback lengths.

For continuation, induct on the unary tail. The empty tail gives addition by zero. A one-headed tail applies the induction hypothesis and then one-extension supplies the successor step on

the result. The zero-headed case is ruled out by unary no-zero-extension. Therefore unary Cont tails read as addition of finite unary lengths, and the theorem-preservation list is limited to these constructor, equality, and continuation clauses.

The no-host-leak field follows because the bridge uses the external natural-number model only in these encode, decode, round-trip, and preservation fields, while the BEDC public surface remains the unary-history NameCert grounded by nat_up_semantic_name_certificate. □

Lean4 ✓ `BEDC.FKernel.Unary.nat up unary standard bridge`
Lean4 ✓ `BEDC.Derived.NatUp.NatUp unary standard bridge`

Theorem 28.34 (NatUp mature unary recursion package)**.** *The unary-history certificate supplies the mature paper-level* $\mathsf{Nat}^{\uparrow}$ *package over BEDC* BHist *and* hsame*. Its zero object is* e*, its successor operation is* E1*, no carried element is a zero-extension, unary generation gives induction, primitive recursion is defined by the empty and one-extension generation cases, and continuation length is addition of the unary tail length. The operation bridges consumed by this package are the already cited* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Mul}^{\uparrow}$*, and* $\mathsf{Order}^{\uparrow}$ *interfaces, not host arithmetic.*

Lean4 ✓ `BEDC.Derived.NatUp.NatUp mature unary recursion package`

Proof. The carrier is the closed unary-generation carrier of the semantic $\mathsf{Nat}^{\uparrow}$ certificate. Its base constructor gives e, its step constructor gives E1, and Definition 28.3 removes every E0 branch from the carried surface.

Induction is the eliminator for that unary-generation predicate. Primitive recursion is obtained by the same cases: assign the empty value at e, and in the one-extension case apply the specified successor clause to the value already computed for the predecessor history.

The endpoint equality used by the recursion clauses is hsame, not equality of host natural numbers. Apply Definition 28.12: the sameness branch is the transport case, and either strict-prefix branch contributes a nonempty unary continuation tail.

Finally, Definition 28.33 reads continuation tails as addition of unary lengths. Induction on the tail gives the empty-tail zero law and the one-headed successor step. Thus the package exposes the NatUp induction, recursion, and continuation-as-addition principles internally over BHist, while arithmetic consumers enter only through the cited $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, and $\mathsf{Order}^{\uparrow}$ interfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$**.**

Constructive story: A natural number begins as a history that has refused every b0-branch: from e one may keep stacking b1-marks, and every such stack is admitted, while the first b0-mark sends the history outside the carrier. Since the kernel already says that two histories are the same only by literal history-equality, the classifier for these unary spines has nothing extra to quotient away; the same unary predicate also tells which sources the certificate covers, which pattern their shape follows, and which raw history remains visible in the ledger. This is why the five name-certificate obligations arrive together: e witnesses that the carrier is nonempty, identity is reflexive, symmetric, and transitive, and a history that is identical to an admitted unary history is admitted as well. A unary history h falls into the signature gap of a $\mathsf{Nat}^{\uparrow}$ token p only when the source-domain first accepts h, the signature relation then pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; after the three checks are unfolded, this says exactly that h is unary and that its b1-depth is the count encoded by p. The standard bridge attached here sends each unary history to a finite unary word and reads the word length back as the ordinary natural-number value, so the outside comparison is a readback of the same spine rather than a new carrier. The chapter registers the construction under the name $\mathsf{Nat}^{\uparrow}$; successor is just one more b1-stack, induction is the closed generation rule for histories, and no Peano axiom, separate $\mathbb{N}$ type, or additional axiom is introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public $\mathsf{Nat}^{\uparrow}$ surface is the unary-history semantic NameCert over BEDC histories together with the paper-side standard bridge to finite unary words and natural-number length readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.FKernel.Unary.nat\_up\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Non-unary external arithmetic theorem inventories and checked standard-model bridge targets are not claimed here.

Upgrade path: no further upgrade – chapter at top of formal axis.

Theorem 28.35 (Unary bridge cut boundary)**.** *Let a consumer use the* $\mathsf{Nat}^{\uparrow}$ *packet through the unary-history certificate of Definition 28.30, the semantic certificate of Definition 28.31, and the standard bridge of Definition 28.33. Then every datum crossing from this chapter to a consumer is forced into one of the following rows:*

1. *the unary source row* $\mathsf{Unary}(h)$*, equivalently* $\mathsf{UnarySource}(h)$*;*
2. *the unary pattern row generated from* e *by repeated* $\mathsf{E1}$*;*
3. *the classifier row* $h\mathsf{hsame}k$ *between unary histories;*
4. *the unary ledger row recording the same admitted history;*
5. *, when the consumer asks for continuation, the BEDC continuation row* $\mathsf{Cont}(h, t, k)$ *with unary source and unary tail.*

The finite unary word, external length readback, encode map, decode map, round-trip witnesses, and bridge preservation witnesses are eliminable bridge cuts: after the consumer-facing projection they leave only the listed BEDC rows. A proposed crossing row whose carrier is a host natural number, a host operation, a non-unary history, or any source coordinate not listed above is not admitted by the $\mathsf{Nat}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.NatUp.UnaryBridgeCutBoundary`

Proof. The concrete certificate of Definition 28.30 displays exactly five BEDC rows: source, pattern, classifier, stability, and ledger. By Definition 28.25, the source row is the judgment $\mathsf{Unary}(h)$. By Definition 28.26, the pattern row is the closed generation spine from e by $\mathsf{E1}$. By Definition 28.27, the classifier row is internal history-sameness restricted to unary histories. By Definition 28.28, the stability row contributes only base generation, one-extension closure, and no-zero exclusion. By Definition 28.29, the ledger row records the same source history rather than a separate external coordinate.

The semantic certificate of Definition 28.31 adds no new consumer carrier. It says that source, pattern, and ledger predicates preserve the same unary-history witness. Therefore any handoff through the semantic certificate projects back to the unary source, unary pattern, classifier, stability, and ledger rows already listed.

It remains to inspect the standard bridge. In Definition 28.33, the finite unary word, length readback, encode field, decode field, and round-trip witnesses are bridge-side interpretation data. The proof of that theorem eliminates them by induction back to the BEDC constructors: e for zero, $\mathsf{E1}$ for successor, and the no-zero rule for the absent $\mathsf{E0}$ branch. Its exactness clause projects external equality of readback lengths back to the classifier row $h\mathsf{hsame}k$, and its continuation clause projects a bridge-side addition reading back to the BEDC row $\mathsf{Cont}(h, t, k)$ with unary tail. Thus these bridge-side names are cuts inside the interpretation, not additional public rows.

Consequently a downstream consumer such as the continuation interface cited in chapter 29 can receive only the unary source, pattern, classifier, ledger, and displayed continuation rows. If a proposed handoff starts from a host natural number, a host operation, a non-unary history, or any coordinate not produced by those rows, then neither the certificate nor the standard bridge has a clause admitting it. Such a row is therefore rejected at the boundary. □

Proposition 28.36 (Unary history judgment codomain mirror boundary)**.** *For the* $\mathsf{Nat}^{\uparrow}$ *certificate of Definition 28.30, the unary history judgment has no independent codomain mirror row. A unary domain row is a single witness* $\mathsf{Unary}(h)$*, and the source, pattern, stability, and ledger fields of Definition 28.25, Definition 28.26, Definition 28.28, and Definition 28.29 all reuse that same displayed history. If a target history* k *is read from the same local packet, then either it is only a classifier comparison* $h\mathsf{hsame}k$*, or it is supplied by an additional unary continuation tail* t *with* $\mathsf{Cont}(h, t, k)$*. In the latter case Definition 28.11 gives exactly the two endpoint cases:* $h\mathsf{hsame}k$*, or* h *is a strict unary prefix of* k*. Hence a non-identity codomain side is not obtained by renaming the unary judgment; it belongs to the continuation or strict-prefix surface.*

Lean4 ✓ `BEDC.Derived.NatUp.NatUnaryHistory codomain mirror boundary`

Proof. Unpack the unary judgment of Definition 28.1. Its generators produce judgments of the form $\mathsf{Unary}(h)$: the empty history is admitted, the one-extension of an admitted history is admitted, and no zero-extension is admitted. Thus the judgment itself names one history at a time.

The certificate fields keep that arity. The source specification of Definition 28.25 is governed by $\mathsf{Unary}(h)$. The pattern specification of Definition 28.26 is repeated one-extension from e, again on the same history. The stability certificate of Definition 28.28 records the base, one-extension, and no-zero facts for this same unary predicate. The ledger policy of Definition 28.29 records the source unary history rather than replacing it by a separate emitted target. These are exactly the rows assembled in Definition 28.30.

The only binary row already present in the unary certificate is the classifier row of Definition 28.27, which compares admitted unary histories by hsame. Reading this as a codomain mirror gives no new target construction: it says only that the displayed target is history-same to the source. To obtain a target history not supplied merely by classifier comparison, one must add a continuation witness $\mathsf{Cont}(h, t, k)$ with a unary tail t. Then Definition 28.11 applies. The empty-tail branch gives endpoint sameness $h\mathsf{hsame}k$, while the nonempty one-tail branch is precisely the strict unary prefix relation of Definition 28.10. Therefore the local chapter supports identity comparison and continuation-generated targets, but it does not support a separate codomain copy of the unary judgment obtained by renaming the same unary row. □

Chapter 29

A Concrete Naming Certificate for $\mathsf{Add}^{\uparrow}$

The addition-like interface is licensed by continuation of unary histories. It is not a primitive operation.

Lean4 ✓ `BEDC.Derived.AddUp.AddUpTasteGate single carrier alignment`

The $\mathsf{Add}^{\uparrow}$ carrier consumes the unary-history source licensed in chapter 28. Its two input histories are $\mathsf{Nat}^{\uparrow}$ unary spines, and $\mathsf{Add}^{\uparrow}$ adds only the displayed Cont result row and continuation ledger.

Theorem 29.1 (AddUp consumes NatUp as a sibling route)**.** *Every accepted* $\mathsf{Add}^{\uparrow}$ *carrier row reads its two source histories through chapter 28 before the displayed* Cont *result row is admitted. Consequently* $\mathsf{Add}^{\uparrow}$ *is a downstream consumer of the* $\mathsf{Nat}^{\uparrow}$ *unary spine, not a kernel-only arithmetic primitive.*

Lean4 ✓ `BEDC.Derived.AddUp.AddUpNatUp sibling dependency route`

Proof. Project the source side of the $\mathsf{Add}^{\uparrow}$ carrier. By Definition 29.7, an additive source is an ordered pair of unary histories together with a continuation witness producing the result row.

The unary hypotheses on the two source histories are precisely the $\mathsf{Nat}^{\uparrow}$ spine discipline cited by chapter 28. The continuation lemmas below then act only after those source histories have already been accepted as unary rows.

Apply Definition 29.5, Definition 29.6, and the continuation-preservation lemma of this section to the displayed Cont witness. These results admit and transport the result row, but they do not create the two source spines. Hence the source read factors through $\mathsf{Nat}^{\uparrow}$ before $\mathsf{Add}^{\uparrow}$ admits its continuation result. □

This dependency order is the horizontal lattice used by the downstream standard bridge and continuation consumers: by Definition 29.1, every additive continuation row first reads the unary source through chapter 28, and only then consumes the displayed Cont witness, bridge readback, and additive ledger rows.

29.1 Unary continuation

Lemma 29.2 (Continuation preserves unary histories)**.** *If* $\mathsf{Unary}(h)$*,* $\mathsf{Unary}(k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{Unary}(r)$*.*

Lean4 ✓ `BEDC.FKernel.Unary.unary cont preserves unary by induction`

Proof. By induction on the generated proof of $\mathsf{Cont}(h, k, r)$. In the empty case, $r = k$, hence $\mathsf{Unary}(r)$ follows from the second premise. In the $\mathsf{E1}$-continuation case, use the induction hypothesis on the subcontinuation and then apply unary closure under $\mathsf{E1}$. The $\mathsf{E0}$-continuation case makes the second input history have shape $\mathsf{E0}(k_0)$, contradicting the premise $\mathsf{Unary}(k)$ by unary no-zero-extension. □

Lemma 29.3 (Unary continuation has no zero result)**.** *If h and k are unary histories, then no continuation $\mathsf{Cont}(h, k, \mathsf{E0}(z))$ can exist.*

Proof. Unary continuation closure sends the two unary inputs and the displayed continuation to a unary result. But a result of shape $\mathsf{E0}(z)$ contradicts the unary no-zero-extension principle. □

Lean4 ✓ `BEDC.FKernel.Unary.unary cont e0 result absurd`

Lemma 29.4 (Unary continuation one-result cases)**.** *If h and k are unary histories and $\mathsf{Cont}(h, k, \mathsf{E1}(r))$, then either $k = \mathsf{e}$ and $h\mathsf{hsameE1}(r)$, or $k = \mathsf{E1}(k_0)$ for a unary tail k_0 and $\mathsf{Cont}(h, k_0, r)$.*

Proof. Apply the continuation one-result inversion. In the empty branch, keep the displayed empty right input and result comparison. In the visible-tail branch, the equality $k = \mathsf{E1}(k_0)$ transports the unary witness for k to the tail witness for k_0, leaving the tail continuation $\mathsf{Cont}(h, k_0, r)$. □

Lean4 ✓ `BEDC.FKernel.Unary.unary cont e1 result cases`

Lemma 29.5 (Unary continuation unit)**.** *For every unary history h, continuation by the empty history and continuation from the empty history produce unary histories internally same to the expected unary source.*

Lean4 ✓ `BEDC.FKernel.Unary.unary cont unit`
↪ variant `BEDC.FKernel.Unary.unary cont unit same pair`

Proof. The left unit is the empty continuation generator. The right unit follows from the continuation unit theorem in the proof-standing layer, restricted to unary histories. Unary closure ensures the resulting histories remain within the unary source discipline. □

Lemma 29.6 (Unary continuation associativity)**.** *Unary continuation is associative up to* hsame *when all participating histories are unary.*

Lean4 ✓ `BEDC.FKernel.Unary.unary continuation associativity`

Proof. This is the continuation associativity theorem specialized to unary histories, together with continuation-preserves-unary-histories for the intermediate results. □

29.2 The certificate

Definition 29.7 (Additive source specification)**.** *The source specification* AddSource *consists of ordered pairs of unary histories together with a continuation witness.*

Lean4 (definition) `BEDC.FKernel.Unary.AddSourceSpec`
↪ variant `BEDC.FKernel.Unary.AddSourceSpec from unary cont`
Lean4 ✓ `BEDC.FKernel.Unary.AddSourceSpec result unary`

Definition 29.8 (Additive pattern specification)**.** *The pattern specification* AddPattern *is unary continuation:*

$$\mathsf{Cont}(h, k, r)$$

with $\mathsf{Unary}(h)$ *and* $\mathsf{Unary}(k)$*.*

Lean4 ✓ `BEDC.FKernel.Unary.additive pattern result unary iff inputs`

Definition 29.9 (Additive classifier specification)**.** *The classifier specification* AddClassifier *classifies additive outputs by* hsame *of the generated result histories.*

Lean4 (definition) `BEDC.FKernel.Unary.AddClassifierSpec`

Definition 29.10 (Additive stability certificate)**.** *The stability certificate* AddStabilityCert *consists of:*

1. *continuation preserves unary histories;*
2. *unary continuation unit;*
3. *unary continuation associativity;*
4. *unary continuation commutativity, by Theorem 9194.6;*
5. *continuation determinacy up to* hsame*.*

Lean4 ✓ `BEDC.FKernel.Unary.additive stability certificate fields`
↪ variant `BEDC.FKernel.Unary.additive stability certificate determinacy field`
↪ variant `BEDC.FKernel.Unary.additive stability certificate unit assoc determinacy`

Definition 29.11 (Additive ledger policy)**.** *The additive ledger policy* AddLedger *records the two input unary histories and the continuation witness generating the output history. For composite additive uses, the ledger composes by the composite gap principle.*

Lean4 (definition) `BEDC.FKernel.Unary.AddLedgerPolicy`
↪ variant `BEDC.FKernel.Unary.AddLedgerPolicy iff unary cont closed`
↪ variant `BEDC.FKernel.Unary.addLedgerPolicy from unary cont closed`
↪ variant `BEDC.FKernel.Unary.unary addition seed from policy`
Lean4 ✓ `BEDC.FKernel.Unary.addLedgerPolicy from unary cont closed`

Theorem 29.12 (Concrete $\mathsf{Add}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{Add}^{\uparrow}} := (\mathsf{AddSource}, \mathsf{AddPattern}, \mathsf{AddClassifier}, \mathsf{AddStabilityCert}, \mathsf{AddLedger})$$

forms a valid naming certificate for the derived interface $\mathsf{Add}^{\uparrow}$*.*

Lean4 ✓ `BEDC.FKernel.Unary.add up name certificate exists`
↪ variant `BEDC.FKernel.Unary.add up name certificate`

Proof. The source, pattern, classifier, stability, and ledger fields are supplied by the preceding definitions. Stability is justified by unary closure, unit, associativity, commutativity via Theorem 9194.6, and continuation determinacy. The ledger policy records the exact continuation witness rather than erasing it. Thus the naming protocol is satisfied. □

Corollary 29.13 ($\mathsf{Add}^{\uparrow}$ is licensed by commutative continuation)**.** *The additive interface* $\mathsf{Add}^{\uparrow}$ *is licensed by commutative unary continuation. It is not a primitive operation of the finite kernel.*

Lean4 ✓ `BEDC.FKernel.Unary.add up licensed not primitive`

Theorem 29.14 (AddUp unary standard bridge)**.** *The public* $\mathsf{Add}^{\uparrow}$ *NameCert surface over unary histories,* Cont*,* AddSource*,* AddPattern*,* AddClassifier*,* AddStabilityCert*, and* AddLedger *carries a conservative paper-level bridge to finite unary words with concatenation and length addition. Carried BEDC histories read back through the* $\mathsf{Nat}^{\uparrow}$ *unary standard bridge of chapter 28; continuation witnesses read back as concatenation witnesses;* hsame *of generated results reads back as equality of unary word lengths; ledger entries record only the two input histories and the continuation witness; and no host natural number or host function is admitted as a BEDC primitive.*

Lean4 ✓ `BEDC.Derived.AddUp.AddUp StdBridge`

Proof. Use Definition 28.33 as the carrier bridge for each unary history appearing in the public $\mathsf{Add}^{\uparrow}$ source. Thus the two input histories and the generated result decode as finite unary words, and the source field of Definition 29.12 supplies no data beyond those two carried histories and the displayed continuation witness.

The continuation witness is read by induction on the right unary tail. The empty tail is the unit case of Definition 29.5; the one-extension tail is the successor step; and Definition 29.6 transports nested tails to ordinary concatenation of finite unary words. The commutative-continuation field listed in Definition 29.10 is therefore available for the public certificate, while the bridge itself reads the ordered continuation witness as the corresponding concatenation witness.

Exactness is the result-classifier clause. By Definition 29.16, two generated results from the same ordered source are related by AddClassifier, and by Definition 29.9 this is hsame of the result histories. Applying the $\mathsf{Nat}^{\uparrow}$ bridge exactness to those result histories reads the comparison as equality of the external unary word lengths, hence as length addition for concatenation.

Finally, the ledger field is the policy of Definition 29.11: it records the two BEDC input histories and the continuation witness that produced the result. The external word model appears only in the bridge readback and exactness fields, so no host natural number, host addition function, or host concatenation function becomes a primitive of the BEDC kernel. □

Theorem 29.15 (Unary continuation monoid activation)**.** *Let the typed naming-certificate template of Definition 9.6 be instantiated on the unary carrier with classifier* hsame*. Assume the core carrier certificate* $\mathsf{NameCert}(\mathsf{Unary}, \mathsf{hsame})$ *from Definition 9.7; take the source, pattern, classifier, and ledger policies to be the continuation policies of Definition 29.7, Definition 29.8, Definition 29.9, and Definition 29.11; assume pattern soundness and ledger soundness for those policies; and assume that* AddStabilityCert *supplies unary closure, empty-history unit, associativity up to* hsame*, and same-source determinacy up to* hsame*. Then*

$$\mathsf{License}(\mathsf{UnaryContMonoid}^{\uparrow}).$$

Lean4 ✓ `BEDC.Derived.AddUp.AddUnaryContinuationMonoid activation package`

Proof. By Definition 9.6, a derived interface is licensed by a certificate once its source, pattern, classifier, stability, and ledger fields are supplied. The assumed $\mathsf{NameCert}(\mathsf{Unary}, \mathsf{hsame})$ provides the carrier-and-classifier core described in Definition 9.7. The source is the ordered unary source with a continuation witness from Definition 29.7; the pattern is unary continuation from Definition 29.8; the classifier is hsame on continuation results from Definition 29.9; and the ledger is the recorded pair of unary inputs together with the continuation witness from Definition 29.11. Pattern soundness sends each source witness to the displayed unary continuation pattern, and ledger soundness records the same source witness in the ledger policy.

The remaining certificate component is stability. Unary closure is the closure field in AddStabilityCert. The empty-history unit is Definition 29.5. Associativity up to hsame is Definition 29.6. Same-source determinacy is the determinacy field of AddStabilityCert, so two continuation results generated from the same ordered unary source pair are classified together by hsame. These fields give a closed carrier, a unit, associative continuation, and deterministic result classification for fixed sources. That is exactly the certificate data for the unary continuation monoid-like interface. Since no order-swap commutativity field is used, the activated upward name is $\mathsf{UnaryContMonoid}^{\uparrow}$. □

Theorem 29.16 (Same-source additive determinacy bridge)**.** *For histories h, k, r, and r', if* $\mathsf{AddSource}(h, k, r)$ *and* $\mathsf{AddSource}(h, k, r')$*, then* $\mathsf{AddClassifier}(r, r')$*.*

Proof. By Definition 29.7, the first source witness supplies unary inputs h and k together with $\mathsf{Cont}(h, k, r)$. The second source witness has the same ordered input pair, so it supplies the same unary inputs together with $\mathsf{Cont}(h, k, r')$. The determinacy field listed in Definition 29.10 applies to these two continuation witnesses and gives $\mathsf{hsame}(r, r')$. By Definition 29.9, AddClassifier classifies additive outputs precisely by hsame of their generated result histories, so this witness is $\mathsf{AddClassifier}(r, r')$. □

Lean4 ✓ `BEDC.FKernel.Unary.AddSourceSpec same source classifier`

Theorem 29.17 (Core additive stability does not imply swapped commutativity)**.** *There is a finite equality-classified carrier with a deterministic total continuation relation Γ such that the carrier* NameCert*, closure, two-sided unit, associativity, same-source determinacy, and the source-to-pattern and source-to-ledger soundness fields of the naming template hold, while the swapped-source additive commutativity field*

$$\Gamma(x, y, r) \quad \textit{and} \quad \Gamma(y, x, s) \quad \Rightarrow \quad r = s$$

fails.

Proof. Let $C = \{e, p, q\}$ and use equality as the classifier. Define $\star : C \times C \to C$ by

$\star$	e	p	q
e	e	p	q
p	p	p	p
q	q	q	q

and let $\Gamma(x, y, r)$ mean $x \star y = r$.

The carrier certificate of Definition 9.7 holds because e inhabits C, equality is reflexive, symmetric, and transitive, and equality transports membership in C. For the additive source role of Definition 29.7, use an ordered pair (x, y) together with the graph witness $\Gamma(x, y, r)$. Take the pattern specification and ledger policy to be this same graph witness, matching the ledger role of Definition 29.11; hence the pattern-soundness and ledger-soundness roles in Definition 9.6 are immediate. The classifier role of Definition 29.9 is literal equality on result elements.

Closure is visible from the table. The row and column for e show $e \star x = x$ and $x \star e = x$ for every $x \in C$. Same-source determinacy holds because Γ is the graph of the function $\star$: if $\Gamma(x, y, r)$ and $\Gamma(x, y, s)$, then $r = x \star y = s$. This is the finite equality-classified analogue of the determinacy field in Definition 29.10.

Associativity follows by cases on the left factor. If $x = e$, then $(e \star y) \star z = y \star z = e \star (y \star z)$. If $x = p$, then $p \star y = p$ and $p \star t = p$ for every $t \in C$, so both $(p \star y) \star z$ and $p \star (y \star z)$ are p. The case $x = q$ is identical with q in place of p.

The swapped-source field fails. The table gives $\Gamma(p, q, p)$ and $\Gamma(q, p, q)$, but the equality classifier does not identify p and q. Thus the fields used for the monoid-like activation in Definition 29.15 do not force the swapped-source additive commutativity field. □

Lean4 ✓ `BEDC.Derived.AddUp.AddUpThreePoint unital graph not swapped commutative`

Theorem 29.18 (Unary continuation monoid activation from stability)**.** *Let* $C : \mathsf{NameCert}(\mathsf{Unary}, \mathsf{hsame})$ *be the core carrier certificate of Definition 9.7. Fix the continuation source, pattern, classifier, and ledger policies of Definition 29.7, Definition 29.8, Definition 29.9, and Definition 29.11. If* $S : \mathsf{AddStabilityCert}$ *supplies unary closure under continuation, empty-history unit, associativity up to* hsame*, and same-source determinacy up to* hsame*, then*

$$\mathsf{License}(\mathsf{UnaryContMonoid}^{\uparrow})$$

holds. No premise asserting $\mathsf{AddCommField}$ *or* $\mathsf{UnaryComm}$ *is required.*

Lean4 ✓ `BEDC.Derived.AddUp.AddUnaryContinuationMonoid activation from stability`

Proof. By Definition 9.6, licensing a derived interface amounts to supplying the source, pattern, classifier, stability, and ledger fields required by its certificate. The carrier-and-classifier core is C, which is a certificate for the unary carrier classified by hsame in the sense of Definition 9.7.

The source field is the ordered unary continuation source: a pair of unary histories together with a witness $\mathsf{Cont}(h, k, r)$. This is the source specification of Definition 29.7. The pattern field is the corresponding unary continuation behavior of Definition 29.8. The classifier field is hsame on generated continuation results, as in Definition 29.9. The ledger field records the same ordered inputs and continuation witness, as in Definition 29.11. Thus source-to-pattern soundness and source-to-ledger soundness follow by unpacking these four policies.

It remains to check the stability component. Unary closure is the closure field supplied by S. The empty-history unit is the unit field supplied by S, matching Definition 29.5. Associativity up to hsame is the associativity field supplied by S, matching Definition 29.6. Same-source determinacy says that for fixed ordered unary inputs h, k, any two witnesses $\mathsf{Cont}(h, k, r)$ and $\mathsf{Cont}(h, k, r')$ produce classified results $\mathsf{hsame}(r, r')$; this is precisely the determinacy field listed in Definition 29.10.

These fields give a closed unary carrier, a two-sided unit, associative continuation, deterministic classification for fixed ordered sources, and an explicit ledger. That is the full certificate for $\mathsf{UnaryContMonoid}^{\uparrow}$. A swapped-source field would compare $\mathsf{Cont}(h, k, r)$ with $\mathsf{Cont}(k, h, s)$; no such comparison is used in the construction, so neither $\mathsf{AddCommField}$ nor $\mathsf{UnaryComm}$ enters the proof. □

Theorem 29.19 (Two-element monoids cannot witness additive noncommutativity)**.** *Let* C *be an equality-classified carrier whose unary source is total. Let* $e, u \in C$ *and assume every* $x \in C$ *satisfies* $x = e$ *or* $x = u$*. Let* $\star : C \times C \to C$ *be a monoid operation with unit* e*, and let the additive continuation relation be the graph of* $\star$*:*

$$\Gamma(x, y, z) \quad \textit{means} \quad x \star y = z.$$

Then for all $x, y, z, z' \in C$*,*

$$\Gamma(x, y, z) \quad \textit{and} \quad \Gamma(y, x, z') \quad \Rightarrow \quad z = z'.$$

Thus no equality-classified monoid with at most two carrier elements and total unary source can witness failure of the swapped-source additive commutativity field measured by Definition 29.9.

Proof. By Definition 29.7, an additive source in this interpretation is an ordered pair together with a graph witness $\Gamma(x, y, z)$. Since the unary source is total, every pair of carrier elements is admissible. By Definition 29.9, the result comparison is the carrier classifier; here that classifier is equality. It is therefore enough to prove $x \star y = y \star x$ for arbitrary $x, y \in C$, because $\Gamma(x, y, z)$ and $\Gamma(y, x, z')$ identify z with $x \star y$ and z' with $y \star x$.

Use the cover $t = e$ or $t = u$ for each carrier element t. If $x = e$, the left-unit law gives $x \star y = e \star y = y$, and the right-unit law gives $y \star x = y \star e = y$. Hence $x \star y = y \star x$. If $y = e$, the same argument with the two roles exchanged gives $x \star y = x = y \star x$.

The only remaining case is $x = u$ and $y = u$. In that case both ordered products are the same expression $u \star u$, so $x \star y = y \star x$ by reflexivity of equality. Combining this product equality with the two graph witnesses gives $z = z'$. The conclusion is exactly the swapped-source comparison not supplied by the monoid-like fields of Definition 29.10. In contrast, Definition 29.17 shows that three carrier elements suffice for an equality-classified unital deterministic graph to fail that comparison. □

Lean4 ✓ `BEDC.Derived.AddUp.AddUpTwoPoint unital graph swapped commutative`

Theorem 29.20 (Unary continuation activation without commutativity)**.** *Let the typed naming-certificate template of Definition 9.6 be instantiated on the unary carrier with classifier* hsame*. Assume a core carrier certificate* $\mathsf{NameCert}(\mathsf{Unary}, \mathsf{hsame})$ *in the sense of Definition 9.7. Use the ordered continuation source, continuation pattern, result classifier, and ledger policies of Definition 29.7, Definition 29.8, Definition 29.9, and Definition 29.11. Assume source-to-pattern soundness and source-to-ledger soundness for those policies. If* $\mathsf{AddStabilityCert}$ *supplies unary closure under continuation, the empty-history unit, associativity up to* hsame*, and same-source determinacy up to* hsame*, then*

$$\mathsf{License}(\mathsf{UnaryContMonoid}^{\uparrow})$$

holds. The implication does not require any swapped-source additive commutativity premise of the form

$$\mathsf{Cont}(h, k, r) \text{ and } \mathsf{Cont}(k, h, s) \Rightarrow \mathsf{hsame}(r, s).$$

Lean4 ✓ `BEDC.Derived.AddUp.AddUnaryContinuation activation without commutativity`

Proof. The naming template of Definition 9.6 licenses a derived interface once its source, pattern, classifier, stability, and ledger fields are supplied. The assumed $\mathsf{NameCert}(\mathsf{Unary}, \mathsf{hsame})$ supplies the carrier inhabitedness, classifier reflexivity, classifier symmetry, classifier transitivity, and carrier transport fields listed in Definition 9.7. Thus the unary carrier and its classifier core are available before any binary behavior is considered.

The source field is the ordered pair of unary histories together with a continuation witness, as in Definition 29.7. The pattern field is the continuation behavior of Definition 29.8. The classifier field is hsame on generated results, as in Definition 29.9. The ledger field records the ordered inputs and the continuation witness, as in Definition 29.11. The assumed soundness fields send every source witness to this pattern witness and to this ledger witness.

It remains only to read the stability fields. Unary closure gives closure of the carrier under the continuation result. The empty-history unit gives the left and right unit clauses for the unary continuation operation, matching Definition 29.5. Associativity up to hsame gives the monoid-like associativity clause, matching Definition 29.6. Same-source determinacy says that two results generated from the same ordered source pair are classified together by hsame; this is exactly the deterministic result-classification role in Definition 29.10.

These fields are precisely the certificate data for $\mathsf{UnaryContMonoid}^{\uparrow}$: an inhabited classified unary carrier, a closed continuation operation, the empty unit, associativity, deterministic classification for fixed ordered sources, and a ledger. A swapped-source field compares the ordered source (h, k) with the different ordered source (k, h). No field in this construction makes that comparison, so the activation of $\mathsf{UnaryContMonoid}^{\uparrow}$ follows without the swapped-source additive commutativity premise. □

Theorem 29.21 (Two-element unital continuation commutativity)**.** *Let U be the unary part of a finite equality-classified carrier with classifier* hsame*, and assume U has exactly the two elements e and u. Let $\star : U \times U \to U$ be a total binary operation with two-sided unit e:*

$$e \star t = t \quad \textit{and} \quad t \star e = t \quad \textit{for every } t \in U.$$

Suppose the continuation relation on U is the graph of $\star$:

$$\mathsf{Cont}(h, k, r) \quad \textit{if and only if} \quad r = h \star k.$$

Then for all $h, k, r, r' \in U$,

$$\mathsf{Cont}(h, k, r) \quad \textit{and} \quad \mathsf{Cont}(k, h, r') \quad \Rightarrow \quad \mathsf{hsame}(r, r').$$

No associativity premise is needed.

Lean4 ✓ `BEDC.Derived.AddUp.AddUpTwoPoint unital continuation commutativity`

Proof. The source role of Definition 29.7 treats a continuation source as an ordered pair together with a continuation witness, and the classifier role of Definition 29.9 compares generated results by hsame. In the equality-classified instance of the carrier certificate of Definition 9.7, proving $\mathsf{hsame}(r, r')$ is the same as proving $r = r'$.

Assume $\mathsf{Cont}(h, k, r)$ and $\mathsf{Cont}(k, h, r')$. Since Cont is the graph of $\star$, these witnesses give $r = h \star k$ and $r' = k \star h$. It is enough to show $h \star k = k \star h$.

Because U has exactly the two elements e and u, each of h and k is either e or u. If $h = e$, then $h \star k = e \star k = k$ by the left-unit law, while $k \star h = k \star e = k$ by the right-unit law. If $k = e$, the same unit laws give $h \star k = h$ and $k \star h = e \star h = h$.

The only remaining case is $h = u$ and $k = u$. Then both ordered products are the identical product $u \star u$, so they are equal by reflexivity. Thus $h \star k = k \star h$ in every case, and the two graph identities give $r = r'$. Equality classification then gives $\mathsf{hsame}(r, r')$.

Relative to the stability fields of Definition 29.10, this is an automatic two-element consequence of graph determinacy and the unit laws, not a separate swapped-source field. Together with Definition 29.17, it identifies three equality classes as the first possible size for such a unital deterministic swapped-source failure. □

Theorem 29.22 (AddUp acceptance bridge fields)**.** *The public* $\mathsf{Add}^{\uparrow}$ *surface carries a paper-level* StandardBridge *to finite unary words with ordinary length addition. Its BEDC side is the UnaryHistory NameCert equipped with* AddSource*,* AddPattern*,* AddClassifier*,* AddStabilityCert*, and* AddLedger*. Its standard side is finite unary words, with empty word as zero and word concatenation read by natural-number length addition. The forward interpretation sends* e *to zero and sends an* E1 *tail to successor of the tail readback. The backward interpretation sends zero to* e *and sends successor to an* E1 *tail. The theorem-preservation field sends unary continuation to ordinary concatenation and therefore to length addition. The reflection field sends standard unary*

concatenation back to Cont. *Host* Nat *and host functions occur only in the bridge codomain and its readback equations, never as a BEDC carrier, classifier, source, stability premise, ledger premise, or proof input.*

Lean4 ✓ `BEDC.Derived.AddUp.AddUp acceptance bridge fields`

Proof. The BEDC source is already fixed by the public certificate of Definition 29.12. Its carrier is UnaryHistory, its source and pattern rows are AddSource and AddPattern, its classifier is AddClassifier, and its stability and ledger rows are AddStabilityCert and AddLedger.

Use the readback pattern of Definition 28.33. The forward map is defined by unary generation: e reads as zero, and E1(h) reads as successor of the readback of h. The backward map is the same induction in the opposite direction, sending zero to e and successor to E1 of the already encoded tail. The zero-extension branch is absent on the BEDC carrier because the carrier is UnaryHistory.

For theorem preservation, apply Definition 29.5 for the empty word and Definition 29.6 for the bracketing comparison. Induction on the right unary word turns a displayed Cont(h, k, r) witness into ordinary concatenation of finite unary words, hence into length addition after readback. The same-source comparison needed for classified results is Definition 29.16.

For reflection, induct on the standard finite unary word used as the right summand. The zero case is the empty continuation unit. The successor case adds one E1 tail and composes with the induction hypothesis through continuation associativity. Thus standard unary concatenation reflects back to a BEDC Cont witness, with result classification supplied by the additive classifier row.

The no-host-leak field is syntactic at this paper site. Host Nat, finite unary words, concatenation, and length addition appear only in the standard codomain, the forward and backward bridge maps, and their round-trip or preservation equations. The BEDC public inputs remain exactly the UnaryHistory carrier, hsame-based classifier, continuation source, stability certificate, and ledger rows of Definition 29.12; no host natural number or host function is accepted as a BEDC carrier, classifier, or proof input. □

Closure status for NameCert$_{\mathsf{Add}^\uparrow}$.

Constructive story: Addition appears once two unary spines are placed in order and the kernel continuation produces the spine that follows them. Thus an admitted row is not an external arithmetic fact but a triple (h, k, r): both inputs are accepted by the unary predicate of Nat$^\uparrow$, the result is unary as well, and r is the displayed continuation of h by k. Two rows name the same addition step only when their three histories match component by component, while the source side covers exactly the unary input pairs with such a continuation, the pattern side records unary closure and the kernel cancellation rows, and the ledger keeps the visible source-result triple aligned with the raw histories that generated it. Because history-equality is identity in each coordinate, the trivial (e, e, e) row supplies an inhabitant and the classifier is automatically reflexive, symmetric, transitive, and closed over the carrier. A triple (h, k, r) enters the signature gap of an Add$^\uparrow$ token p when the source-domain accepts the row, the signature relation pairs it with an answer continuation and evidence ledger, and the token-introduction relation names that answer as p; once unfolded, this says that h and k are unary, that Cont(h, k, r) is the kernel witness, and that the b1-depth of r is the sum encoded by p. The standard unary bridge then reads the same continuation as concatenation of two finite unary words, with word length giving back the equality between the concatenated length and the two input lengths. The chapter registers the construction under the name Add$^\uparrow$; addition is the continuation operation viewed on unary triples, with unit and associativity proved as monoid theorems rather than installed as arithmetic primitives.

Theory closure: TheoryGate(NameCert$_{\mathsf{Add}^\uparrow}$, maturePackageClosed)

Scope closed: The mature Add$^\uparrow$ package is exactly the displayed UnaryHistory carrier and NameCert of Definition 29.12, the standard unary bridge of Definition 29.14, the monoid activation theorem Definition 29.15, the finite swapped-source boundary theorems in this chapter, and the paper-level acceptance bridge fields of Definition 29.22.

Formal verification: FormalStatus(NameCert$_{\mathsf{Add}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.FKernel.Unary.add\_up\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full $\mathsf{Nat}^{\uparrow}$ arithmetic, recursion equations, cancellation, order compatibility, multiplication interaction, and machine-checked standard-bridge closure are outside this $\mathsf{Add}^{\uparrow}$ closure.

Upgrade path: no further upgrade — chapter at top of formal axis

Proposition 29.23 (Zero-result continuation has no same-surface reverse)**.** *In the public* $\mathsf{Add}^{\uparrow}$ *surface of Definition 29.12, the displayed implication*

$$\mathsf{Unary}(h) \quad \textit{and} \quad \mathsf{Unary}(k) \quad \textit{and} \quad \mathsf{Cont}(h, k, \mathsf{E0}(z)) \quad \Rightarrow \quad \bot$$

is a one-way interpretation boundary. More precisely, for fixed unary inputs h *and* k*, no same-surface reverse rule may use only the zero-result absence row to supply the missing zero-result continuation row*

$$\neg\mathsf{Cont}(h, k, \mathsf{E0}(z)) \quad \Rightarrow \quad \mathsf{Cont}(h, k, \mathsf{E0}(z)).$$

Consequently the no-zero-result row cannot be completed into an equivalence row on the visible AddSource*,* AddPattern*,* AddClassifier*, or* AddLedger *surface.*

Lean4 ✓ `BEDC.Derived.AddUp.AddUnaryZeroResult one way boundary`

Proof. Assume h and k are unary histories. By Definition 29.3, every displayed continuation witness

$$\mathsf{Cont}(h, k, \mathsf{E0}(z))$$

is impossible. Hence the zero-result absence row

$$\neg\mathsf{Cont}(h, k, \mathsf{E0}(z))$$

is already forced by the unary source hypotheses.

Suppose a same-surface reverse rule of the displayed form were available. Applying it to the absence row just obtained would produce a continuation witness

$$\mathsf{Cont}(h, k, \mathsf{E0}(z)).$$

Applying Definition 29.3 again to the same unary inputs and this witness gives a contradiction. Thus the reverse rule cannot coexist with the unary continuation calculus.

The certificate rows in this chapter do not contain another route around this obstruction. By Definition 29.7, a source row for an additive output contains an actual continuation witness. By Definition 29.8, the pattern row is that continuation witness together with the unary input discipline. By Definition 29.11, the ledger records the same witness rather than replacing it by an absence statement. Finally, Definition 29.9 compares generated result histories only after a result has been generated. Therefore the absence of a zero-headed continuation is not a hidden source, pattern, classifier, or ledger witness for a zero-headed result; it is exactly the boundary excluding such a row. □

Chapter 30

A Concrete Naming Certificate for $\mathsf{Mul}^{\uparrow}$

The multiplication interface $\mathsf{Mul}^{\uparrow}$ is the repeated-addition consumer of the unary natural-number spine. It sits after $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}$ in the arithmetic chain, uses the concrete history monoid route of $\mathsf{Monoid}^{\uparrow}$, and supplies the multiplication row consumed by the divisibility and prime-number surface of chapter 64.

30.1 Unary product rows

Definition 30.1 (MulUp unary product carrier)**.** *The carrier for* $\mathsf{Mul}^{\uparrow}$ *consists of finite product rows*

$$\mathsf{MulRow}(d, q, n, \lambda)$$

where d, q, and n are $\mathsf{Nat}^{\uparrow}$ *unary histories, λ is a finite additive iteration ledger, and the ledger starts at* e*, reads the multiplier q one unary step at a time, and at each visible step applies the* $\mathsf{Add}^{\uparrow}$ *continuation row that adds one further copy of d to the current partial product. The final ledger endpoint is the result row n.*

Lean4 ✓ `BEDC.Derived.MulUp.TasteGate.MulUpTasteGate single carrier alignment`

Definition 30.2 (MulUp unary product classifier)**.** *Two* $\mathsf{Mul}^{\uparrow}$ *product rows* $\mathsf{MulRow}(d, q, n, \lambda)$ *and* $\mathsf{MulRow}(d', q', n', \lambda')$ *are classified together when the two factor rows satisfy* $d\mathsf{hsame}d'$ *and* $q\mathsf{hsame}q'$*, the result rows satisfy* $n\mathsf{hsame}n'$*, and the two additive iteration ledgers replay the same finite sequence of* $\mathsf{Add}^{\uparrow}$ *continuation steps after transport along those* hsame *comparisons.*

Theorem 30.3 (MulUp unary product NameCert obligations)**.** *The carrier and classifier of Definition 30.1 and Definition 30.2 supply the local* $\mathsf{NameCert}_{\mathsf{Mul}^{\uparrow}}$ *obligation surface over* BHist*: the empty multiplier row is inhabited by the endpoint* e*, classifier reflexivity, symmetry, and transitivity are inherited from* hsame *on the displayed factor and result histories, and transport of a product row along classified factors and result histories replays the same finite additive ledger.*

Proof. For inhabitance, take any unary multiplicand d and the empty multiplier. The additive iteration ledger has no visible addition step and ends at e, so it is an accepted product row.

The classifier clauses are componentwise. Reflexivity, symmetry, and transitivity use the corresponding hsame facts on the two factors and the result, while the ledger component is compared by replaying the same finite list of $\mathsf{Add}^{\uparrow}$ rows after those transports.

For carrier transport, push $d\mathsf{hsame}d'$, $q\mathsf{hsame}q'$, and $n\mathsf{hsame}n'$ through the source, counter, and endpoint coordinates of the ledger. Each intermediate row remains a $\mathsf{Nat}^{\uparrow}$ unary history, and each additive step remains an $\mathsf{Add}^{\uparrow}$ continuation row, so the transported tuple is again a $\mathsf{Mul}^{\uparrow}$ product row. □

Theorem 30.4 (MulUp additive route)**.** *Every accepted* $\mathsf{Mul}^{\uparrow}$ *row factors through the unary source of* $\mathsf{Nat}^{\uparrow}$*, the continuation-addition row of* $\mathsf{Add}^{\uparrow}$*, and the concrete history monoid route of* $\mathsf{Monoid}^{\uparrow}$*. Equivalently, multiplication is not a new kernel operation: it is a finite* Cont *replay in which the multiplier ledger schedules repeated* $\mathsf{Add}^{\uparrow}$ *applications over unary histories.*

Proof. Project an accepted $\mathsf{Mul}^{\uparrow}$ row to its carrier data. The displayed factors and result are unary histories, so the source side is already inside the $\mathsf{Nat}^{\uparrow}$ carrier of chapter 28.

Read the ledger next. Its base row is the empty product endpoint, and every visible multiplier step is an $\mathsf{Add}^{\uparrow}$ continuation row that appends one copy of the multiplicand to the current partial product. Thus the only operation used in the computation is the addition certificate of chapter 29.

Finally, the ledger concatenation of successive continuation rows is exactly the concrete unary-history monoid route displayed by chapter 40. Repacking these three projections gives the claimed factorization and leaves no independent multiplication primitive. □

Theorem 30.5 (MulUp unary product recursion laws)**.** *Every accepted row* $\mathsf{MulRow}(d, q, n, \lambda)$ *satisfies the unary recursion laws for multiplication. When the visible multiplier ledger is empty, the product endpoint is the carrier endpoint* e*. When the visible multiplier ledger has successor form, the row decomposes into the previous product row followed by one* $\mathsf{Add}^{\uparrow}$ *continuation row that adds the multiplicand* d *to the previous partial product.*

These endpoint and successor laws are stable under componentwise hsame *transport of the displayed multiplicand, multiplier, and product histories, and they are stable under replay of the stored* Cont *ledger. Thus* $\mathsf{Mul}^{\uparrow}$ *remains a finite repeated-addition route through* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*, rather than a separate primitive operation.*

Proof. Inspect the visible unary multiplier ledger in the accepted row. In the empty case there is no displayed multiplier step, so the product row is exactly the endpoint row already present in the carrier, namely e.

In the successor case the final visible multiplier step can be peeled from the finite ledger. The prefix ledger is the previous product row, and the peeled terminal row is an $\mathsf{Add}^{\uparrow}$ continuation adding one copy of the multiplicand to that previous partial product.

Transport is componentwise. The classifier supplies the hsame comparisons on the multiplicand, multiplier, and product histories, and continuation replay sends each stored additive row to the transported additive row. Therefore the endpoint and successor decompositions persist after transport and after replay of the same finite ledger. □

Theorem 30.6 (MulUp finite data boundary)**.** *The* $\mathsf{Mul}^{\uparrow}$ *chapter exports only the finite unary product rows of Definition 30.1, their classifier of Definition 30.2, the local* $\mathsf{NameCert}_{\mathsf{Mul}^{\uparrow}}$ *obligations of Definition 30.3, the additive route of Definition 30.4, and the recursion laws of Definition 30.5. The downstream divisibility and* $\mathsf{Prime}^{\uparrow}$ *multiplication consumer may therefore read* NatMul *rows through this repeated-addition boundary, while full semiring or ring algebra, divisibility completeness, and standard bridge checking remain outside this local certificate surface.*

Proof. The exported rows are exactly the carrier, classifier, NameCert obligations, route theorem, and recursion-law theorem just listed. Each row is a finite object: two factor histories, one result

history, and a finite additive ledger whose entries are $\mathsf{Add}^{\uparrow}$ continuation steps over $\mathsf{Nat}^{\uparrow}$ unary histories.

The divisibility and prime-number consumer in chapter 64 reads the same multiplication relation through the NatMul surface, in particular the semantic exit and phase boundary of Definition 64.24 and Definition 64.71. Those consumer rows add divisibility, order, and audit-clean arithmetic boundaries, but they do not change the local repeated-addition carrier defined here.

Since none of the four local exports contains a semiring carrier, a ring carrier, a quotient completion, a divisibility-completeness theorem, or a checked standard bridge, those structures are not claimed by this chapter. □

Theorem 30.7 (MulUp local-to-Full Nat Arithmetic handoff)**.** *The local exports of Definition 30.1, Definition 30.2, Definition 30.3, Definition 30.4, Definition 30.5, and Definition 30.6 form precisely the* $\mathsf{Mul}^{\uparrow}$ *input face read by the mature Full Nat Arithmetic boundary of Definition 64.82. The checked* NatMul *semantic exit,* $\mathsf{Add}^{\uparrow}$ *distributivity rows,* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and divisibility-order consumer rows are downstream consumer rows over that face; they do not add new local carrier coordinates to the repeated-addition certificate.*

Proof. Project the local tuple to its carrier, classifier, NameCert obligations, additive route, recursion laws, and finite-data boundary. These projections expose only unary histories, componentwise hsame comparisons, and finite Cont ledgers whose visible steps are $\mathsf{Add}^{\uparrow}$ continuation rows.

Use the additive route through $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$ to identify the product relation as repeated addition over the same BHist spine. The recursion laws give the empty-multiplier endpoint and successor-step decomposition, while the finite-data boundary restricts the local chapter to those rows and rules out host multiplication or a second carrier.

Finally apply the Prime-side mature boundary of Definition 64.82. That boundary consumes the projected face by adding the checked NatMul semantic exit, the $\mathsf{Add}^{\uparrow}$ distribution rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the divisibility-order rows as consumer structure over the same carrier and classifier. Since these rows are introduced only at the consumer boundary, the local $\mathsf{Mul}^{\uparrow}$ chapter keeps exactly the repeated-addition coordinates listed above. □

Theorem 30.8 (MulUp unary multiplication law bundle)**.** *The concrete unary product carrier over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *supplies exactly the public* $\mathsf{Mul}^{\uparrow}$ *multiplication laws used by downstream arithmetic consumers: totality on unary factors, unary closure of results, functionality up to* hsame, *left and right unit laws for* E1(e), *commutativity up to* hsame, *and associativity up to* hsame.

Proof. Start with the local repeated-addition surface. The carrier of Definition 30.1 and the classifier of Definition 30.2 provide the displayed product rows and the componentwise hsame comparison used to identify equal results.

The local certificate obligations of Definition 30.3 and the recursion laws of Definition 30.5 give the finite endpoint and successor-step account of product formation. The handoff theorem Definition 30.7 identifies this local face as the one consumed by the checked Full Nat Arithmetic boundary.

The Prime-side NatMul rows then package the public laws over that face. Their checked bundle states totality for unary factors, unary result closure, functionality, the two E1(e) unit laws, commutativity, and associativity, all with the result comparisons expressed by hsame. Repacking those rows yields precisely the announced $\mathsf{Mul}^{\uparrow}$ law bundle. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unary mulup certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{Mul}\uparrow}$.

Constructive story: Unary multiplication is generated on the same spine used for natural numbers: the carried inputs and result are unary histories, the multiplication relation is accepted only when a finite $\mathsf{Add}^{\uparrow}$ iteration ledger witnesses the repeated append pattern named by NatMul, and the local NameCert row records that finite surface. Two multiplication presentations are classified by hsame on the unary endpoints and by functionality of the result ledger, so componentwise transport identifies the factor, multiplier, and product rows without importing a semiring carrier or a host multiplication table. The source specification is the unary-history carrier together with the factor, multiplier, and result rows, the pattern specification records totality, result closure, functionality, unit laws, commutativity, associativity, additive distribution, order monotonicity, and divisibility-order rows, and the ledger policy keeps the raw factors, multiplier splits, appended result, unit endpoint, and continuation evidence visible. The full natural-arithmetic mature boundary is the paper-side consumer boundary of Definition 64.82.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Mul}\uparrow}$, maturePackageClosed)

Scope closed: The mature Full Nat Arithmetic boundary includes the unary product carrier of Definition 30.1, classifier of Definition 30.2, local obligations of Definition 30.3, additive route of Definition 30.4, recursion laws of Definition 30.5, finite-data boundary of Definition 30.6, and the BHist unary-history multiplication package of Definition 64.82, including the NatMul semantic exit, checked law audit package, $\mathsf{Add}^{\uparrow}$ distributivity rows, $\mathsf{Order}^{\uparrow}$ monotonicity row, and divisibility-order consumer rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Mul}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PrimeUp.NatMul\_result\_semanticNameCert`

Bridge status: paperBridge

Not claimed: This chapter does not close semiring completion, quotient carriers, host multiplication, or cancellation laws outside the displayed unary-history rows.

Upgrade path: Further closure requires a checked standard bridge for the mature Full Nat Arithmetic surface.

$\mathsf{Int}^{\uparrow}$ is the L7 signed-unary arithmetic surface built from the unary magnitude rows of $\mathsf{Nat}^{\uparrow}$ in chapter 28 and the balancing and continuation rows of $\mathsf{Add}^{\uparrow}$ in chapter 29. $\mathsf{Rat}^{\uparrow}$ in chapter 37 consumes this surface as signed numerator data, while the DyadicRatCore, StreamName, RegSeqRat, and Real route is downstream consumer context rather than the source that defines $\mathsf{Int}^{\uparrow}$.

The IntUp certificate hub routes signed-pair material through the NatUp unary carrier of chapter 28, the AddUp continuation ledger of chapter 29, and the RatUp denominator boundary of chapter 37; the child files below keep those sibling dependencies explicit.

Chapter 31

A Concrete Naming Certificate for $\mathsf{Int}^{\uparrow}$

The integer interface $\mathsf{Int}^{\uparrow}$ is licensed by signed unary histories: a sign component together with a unary magnitude under continuation discipline. It is not a primitive operation of the finite kernel. The construction follows the 5-field NameCert template established by chapter 28 and chapter 29, with the carrier extended by a sign indicator and the stability fields adjusted for sign-aware continuation.

The signed-pair carrier consumes the arithmetic predecessor lattice rather than standing as a kernel island: Nat supplies unary magnitudes, chapter 29 supplies continuation-compatible arithmetic routing, and chapter 37 later reads the signed numerator surface through the same integer carrier. The signed-pair carrier consumes the Nat unary magnitude carrier and Add continuation ledger through chapter 28 and chapter 29, and every later Int classifier row is read as a sign-tagged extension of those two sibling surfaces.

31.1 Carrier definition

Definition 31.1 (Signed unary integer carrier)**.** *The carrier predicate for* $\mathsf{Int}^{\uparrow}$ *uses an existing mark as the sign component and an existing unary history as the magnitude component:*

$$\mathsf{IntCarrier}(s, h) := (s = \mathsf{b0} \lor s = \mathsf{b1}) \land \mathsf{Unary}(h).$$

Lean4 (definition) `BEDC.Derived.IntUp.IntCarrier`
Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier sign cases`

Proof. This is a definitional clause over the finite-kernel mark constructors and the unary-history predicate. □

Theorem 31.2 (Signed carrier is closed under same-sign continuation)**.** *For a fixed sign* $s : \mathsf{BMark}$*, if* $\mathsf{IntCarrier}(s, h)$*,* $\mathsf{IntCarrier}(s, k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{IntCarrier}(s, r)$*.*

Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier continuation closed same sign`

Proof. Keep the sign disjunction supplied by the first carrier witness.

The two carrier witnesses also provide unary magnitude witnesses for h and k. Apply unary continuation closure to those two magnitude witnesses and the displayed continuation ledger $\mathsf{Cont}(h, k, r)$, then pair the resulting unary witness for r with the retained sign clause. □

Theorem 31.3 (Signed carrier transports along mark sameness)**.** *If* $\mathsf{IntCarrier}(s, h)$ *and* $\mathsf{msame}(s, s')$*, then* $\mathsf{IntCarrier}(s', h)$*.*

Proof. The sign-sameness witness identifies the two finite-kernel sign marks. After that identification, the original signed unary carrier witness has the required transported sign. □

Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier sign msame transport`

Theorem 31.4 (Signed carrier transports along history sameness)**.** *For a fixed sign* s*, if* $\mathsf{IntCarrier}(s, h)$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{IntCarrier}(s, k)$*.*

Proof. The carrier splits into the sign clause and the unary magnitude clause. The sign clause is unchanged, while the unary-history witness is transported along $\mathsf{hsame}(h, k)$. □

Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier transport hsame magnitude`

Theorem 31.5 (Integer magnitude transport along history sameness)**.** *For a fixed sign* s*, if* $\mathsf{IntCarrier}(s, h)$ *holds and* h *is internally the same history as* k*, then* $\mathsf{IntCarrier}(s, k)$ *holds.*

Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier magnitude hsame transport`

Proof. The sign disjunction is preserved unchanged, while the unary-history component is transported across $\mathsf{hsame}(h, k)$ by the finite-kernel unary transport theorem. □

Theorem 31.6 (Integer carrier magnitude induction)**.** *For a fixed sign* $s : \mathsf{BMark}$*, any predicate* P *on histories holds for every integer-carried magnitude if it holds at* e *and is preserved by one-step unary extension along* $\mathsf{IntCarrier}(s, -)$*.*

Proof. Unpack $\mathsf{IntCarrier}(s, h)$ into its sign clause and unary-history witness. Apply unary-history induction to the magnitude; the step rebuilds the integer carrier from the fixed sign clause and the current unary witness. □

Lean4 ✓ `BEDC.Derived.IntUp.IntCarrier magnitude induction`

Definition 31.7 (Integer source specification)**.** *The source specification for* $\mathsf{Int}^{\uparrow}$ *is exactly the signed unary integer carrier:*

$$\mathsf{IntSourceSpec}(s, h) := \mathsf{IntCarrier}(s, h).$$

Lean4 (definition) `BEDC.Derived.IntUp.IntSourceSpec`
Lean4 ✓ `BEDC.Derived.IntUp.intup source specification`

Proof. This is a direct definitional identification of the source specification with the carrier predicate. □

Definition 31.8 (Integer classifier specification)**.** *The classifier for two signed unary representatives requires both representatives to be in the integer carrier, their signs to be mark-same, and their magnitudes to be history-same:*

$$\mathsf{IntClassifierSpec}((s, h), (t, k)) := \mathsf{IntCarrier}(s, h) \wedge \mathsf{IntCarrier}(t, k) \wedge \mathsf{msame}(s, t) \wedge \mathsf{hsame}(h, k).$$

Lean4 (definition) `BEDC.Derived.IntUp.IntClassifierSpec`

Definition 31.9 (Integer difference-pair carrier)**.** *The signed-pair presentation of* $\mathsf{Int}^{\uparrow}$ *uses two unary histories, read as a positive counter and a negative counter:*

$$\mathsf{IntPairCarrier}(p, n) := \mathsf{Unary}(p) \wedge \mathsf{Unary}(n).$$

Lean4 (definition) `BEDC.Derived.IntUp.IntPairCarrier`

Theorem 31.10 (Integer difference-pair carrier magnitude-pair induction)**.** *Let* P *be a predicate on two history magnitudes. If* $P(\mathsf{e}, \mathsf{e})$ *holds, if* $P(p, n)$ *is preserved by replacing the positive component with* $\mathsf{E1}(p)$ *whenever both components are unary, and if* $P(p, n)$ *is preserved by replacing the negative component with* $\mathsf{E1}(n)$ *whenever both components are unary, then every carried integer difference-pair satisfies* P*:*

$$\mathsf{IntPairCarrier}(p, n) \Longrightarrow P(p, n).$$

Lean4 ✓ `BEDC.Derived.IntUp.IntPairCarrier magnitude pair induction`

Proof. Unpack $\mathsf{IntPairCarrier}(p, n)$ into unary witnesses for both components. First apply unary-history induction to the negative component to derive $P(\mathsf{e}, n)$ from the base case and the negative-component step. Then apply unary-history induction to the positive component. At each positive successor, use the positive-component step with the current positive unary witness, the fixed negative unary witness, and the induction hypothesis. □

Theorem 31.11 (Integer pair carrier pair induction)**.** *For any predicate* P *on two histories, if* $P(\mathsf{e}, \mathsf{e})$ *holds and* P *is preserved by one-step unary extension of either counter while the other counter remains unary, then* $P(p, n)$ *holds for every carried integer difference pair* $\mathsf{IntPairCarrier}(p, n)$*.*

Proof. Unfold IntPairCarrier into its positive and negative unary counter witnesses. Run unary-history induction over the positive counter; its base case is unary-history induction over the negative counter, and the two step cases are exactly the two supplied one-step closure rules. □

Lean4 ✓ `BEDC.Derived.IntUp.IntPairCarrier pair induction`

Definition 31.12 (Integer difference-pair classifier)**.** *Two carried unary pairs classify together when their cross-additions are history-same:*

$$\mathsf{IntPairClassifier}((p, n), (q, m)) := \mathsf{IntPairCarrier}(p, n) \wedge \mathsf{IntPairCarrier}(q, m) \wedge \mathsf{hsame}(\mathsf{append}(p, m), \mathsf{append}(q, n)).$$

Lean4 (definition) `BEDC.Derived.IntUp.IntPairClassifier`

Theorem 31.13 (Integer pair classifier zero representative criterion)**.** *For all* $p, n : \mathsf{BHist}$*, if* $\mathsf{Unary}(p)$ *and* $\mathsf{Unary}(n)$*, then the ordinary zero representative detects exactly the zero class of the carried signed pair:*

1.

$$\mathsf{IntPairClassifier}((p, n), (\mathsf{e}, \mathsf{e})) \Longleftrightarrow \mathsf{hsame}(p, n);$$

2.

$$\mathsf{IntPairClassifier}((\mathsf{e}, \mathsf{e}), (p, n)) \Longleftrightarrow \mathsf{hsame}(p, n).$$

The ten-field fit is: target ordinary zero class of signed difference pairs; BEDC source IntPairCarrier*,* IntPairClassifier*,* Unary*,* append*,* e*, and* hsame*; carrier carried unary pairs; classifier cross-addition sameness; stability append unit transport; ledger hidden empty counter pair; exactness zero representative iff; downstream consumer* RatUp *numerator-zero and* IntUp *closure; status* ProofTarget*.*

Proof. Unfold IntPairClassifier. In the first forward direction, the cross-addition field reads hsame(append(p, e), appe the right and left unit laws for append reduce this comparison to hsame(p, n).

For the first reverse direction, the unary hypotheses give IntPairCarrier(p, n), and the empty unary witness gives IntPairCarrier(e, e). Transport the assumed hsame(p, n) through the same two unit identifications to obtain the required cross-addition field.

For the second orientation, unfold the classifier for $((\mathsf{e}, \mathsf{e}), (p, n))$. The cross-addition field is hsame(append(e, n), append(p, e)), so the unit laws and symmetry of hsame identify it with hsame(p, n).

The reverse direction again packages the two carrier witnesses and transports hsame(p, n) across the unit laws, reversing the comparison by symmetry when the displayed orientation requires it. □

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier zero representative criterion`

Theorem 31.14 (Integer pair classifier positive append context)**.** *If* a, p, n, q, m : BHist*, a is unary, and* IntPairClassifier$((p, n), (q, m))$*, then*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, p), n), (\mathsf{append}(a, q), m)).$$

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier append positive context`

Proof. Unpack the classifier into the two unary-pair carriers and the cross-addition comparison append(p, m)hsameappend(q, n). Unary append closure puts append(a, p) and append(a, q) back in the carried positive counters. Prefix the cross-addition comparison by a and reassociate both sides by append_assoc, yielding the displayed contextual classifier. □

Theorem 31.15 (Integer pair classifier signed-context invariance)**.** *Let* a, b, p, n, q, m : BHist*. If* Unary(a)*,* Unary(b)*, and* IntPairClassifier$((p, n), (q, m))$*, then*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, p), \mathsf{append}(b, n)), (\mathsf{append}(a, q), \mathsf{append}(b, m))).$$

The ten-field fit is: target simultaneous signed-context transport for integer difference pairs; BEDC source IntPairCarrier*,* IntPairClassifier*,* Unary*,* append*, and* hsame*; carrier carried unary positive and negative counters; classifier cross-addition sameness; stability four unary append-closed endpoints; ledger common signed context* append(a, b)*; exactness preservation of the cross-addition witness; downstream consumer* RatUp *numerator/denominator context transport and the* IntUp *signed-pair arithmetic certificate; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier signed context invariance`

Proof. Unpack IntPairClassifier$((p, n), (q, m))$ into the carried endpoints and the cross-addition witness append(p, m)hsameappend(q, n). Unary append closure applied to the two positive counters and to the two negative counters gives the four carried endpoints required for the displayed classifier.

It remains to transport the cross-addition field. The left target comparison has source append(append(a, p), appe and the right target comparison has source append(append(a, q), append(b, n)). Associativity rewrites these as signed-context words containing the same unary prefix. Commuting unary factors brings the common signed context append(a, b) to the front on both sides.

After this normalization, the remaining suffixes are exactly append(p, m) and append(q, n). Prefix the original hsame comparison by the common context append(a, b), then reassociate in the reverse direction to obtain the required classifier field. □

Theorem 31.16 (Integer difference-pair classifier equivalence fields)**.** *On carried unary pairs,* IntPairClassifier *supplies the expected integer quotient fields:*

1. *reflexivity: for every* IntPairCarrier(p, n), IntPairClassifier$((p, n), (p, n))$*;*
2. *symmetry: from* IntPairClassifier$((p, n), (q, m))$ *one obtains* IntPairClassifier$((q, m), (p, n))$*;*
3. *transitivity: from* IntPairClassifier$((p, n), (q, m))$ *and* IntPairClassifier$((q, m), (r, o))$ *one obtains* IntPairClassifier$((p, n), (r, o))$*;*
4. *componentwise transport: if* $p$hsamep'*,* $n$hsamen'*,* $q$hsameq'*, and* $m$hsamem'*, then a classifier witness for* $((p, n), (q, m))$ *transports to one for* $((p', n'), (q', m'))$ *whenever all four pairs are carried.*

The ten-field fit is: traditional target signed integers as unary difference pairs; BEDC source Unary*,* hsame*, and* append*; carrier* IntPairCarrier*; classifier* IntPairClassifier*; stability componentwise* hsame *transport; ledger the two unary counters; core theorem this block; exactness cross-addition classification; standard bridge the ordinary presentation of integers as* (p, n) *modulo* $p+m = q+n$*; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier equivalence fields`

Proof. For reflexivity, unary append closure shows both cross-additions append(p, n) are unary, and reflexivity of hsame supplies the cross-addition equality. The two carrier witnesses are reused unchanged.

For symmetry, unpack the classifier witness. The carrier witnesses swap positions, and symmetry of hsame reverses the cross-addition comparison between append(p, m) and append(q, n).

For transitivity, suppose the two displayed classifier witnesses give append(p, m)hsameappend(q, n) and append(q, o)hsameappend(r, m). Append the missing contexts to both comparisons so that the shared unary factors are visible.

Using append_assoc, unary append commutativity, and transitivity of hsame, rewrite the composed comparison to the form

$$\mathsf{append}(\mathsf{append}(p, o), m)\mathsf{hsameappend}(\mathsf{append}(r, n), m).$$

The common suffix m is carried by unary closure, so append_right_cancel, equivalently common-context cancellation, removes it and yields append(p, o)hsameappend(r, n). Together with the endpoint carrier witnesses, this is exactly IntPairClassifier$((p, n), (r, o))$.

For componentwise transport, transport unary carrier evidence along the four hsame links. Compatibility of append with hsame transports the cross-addition comparison from append(p, m)hsameappend(q, n) to append(p', m')hsameappend(q', n'). □

Theorem 31.17 (Integer classifier magnitude pair induction)**.** *Any predicate* P *on pairs of histories holds for the two magnitudes of every* IntClassifierSpec *witness if it holds at* (e, e) *and is preserved by one-step unary extension on either magnitude while the other magnitude remains unary.*

Proof. Unpack the classifier into its two signed unary carrier witnesses. These give unary histories for both magnitudes. Run unary-history induction over the left magnitude; its base is supplied by a second unary-history induction over the right magnitude, and its step uses the left extension rule. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec magnitude pair induction`

Theorem 31.18 (Integer classifier endpoint sign cases)**.** *If* $\mathsf{IntClassifierSpec}(x, y)$*, then the two endpoint signs are either both* $\mathsf{b0}$ *or both* $\mathsf{b1}$*, and both endpoint magnitudes are unary:*

$$((x_1 = \mathsf{b0} \wedge y_1 = \mathsf{b0}) \vee (x_1 = \mathsf{b1} \wedge y_1 = \mathsf{b1})) \wedge \mathsf{Unary}(x_2) \wedge \mathsf{Unary}(y_2).$$

Proof. Unpack the classifier into the two signed carrier witnesses, sign sameness, and magnitude sameness. The left carrier splits the left sign into the two finite mark cases and gives the unary witness for the left magnitude. The sign sameness transports the same sign case to the right endpoint, while the right carrier supplies the unary witness for the right magnitude. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec endpoint sign cases`

Theorem 31.19 (Integer classifier cross-sign exclusion)**.** *For all* $h, k : \mathsf{BHist}$*, cross-signed endpoints cannot be classified:*

$$\forall h, k : \mathsf{BHist}, \quad (\mathsf{IntClassifierSpec}((\mathsf{b0}, h), (\mathsf{b1}, k)) \to \mathsf{False}) \wedge (\mathsf{IntClassifierSpec}((\mathsf{b1}, h), (\mathsf{b0}, k)) \to \mathsf{False}).$$

The ten-field fit is: traditional target signed-integer sign separation; BEDC source $\mathsf{IntCarrier}$*,* $\mathsf{IntClassifierSpec}$*, and raw mark no-confusion; carrier signed unary representatives; classifier* $\mathsf{IntClassifierSpec}$*; stability from existing sign and magnitude transport; ledger the sign mark and unary magnitude; core theorem this block; exactness cross-sign exclusion; standard bridge ordinary signed-integer sign disjointness; status* $\mathsf{ProofTarget}$*.*

Proof. Project the sign-sameness component from Definition 31.8. In the first branch, a classifier from $(\mathsf{b0}, h)$ to $(\mathsf{b1}, k)$ supplies $\mathsf{msame}(\mathsf{b0}, \mathsf{b1})$, which contradicts raw mark no-confusion in Definition 2.9.

The second branch is the opposite orientation. A classifier from $(\mathsf{b1}, h)$ to $(\mathsf{b0}, k)$ supplies $\mathsf{msame}(\mathsf{b1}, \mathsf{b0})$, and the exchanged half of Definition 2.9 excludes that mark-sameness judgment. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec cross sign exclusion`

Theorem 31.20 (Integer classifier magnitude transport)**.** *If* $\mathsf{IntClassifierSpec}((s_x, h_x), (s_y, h_y))$*, and the two magnitudes transport to* h'_x *and* h'_y *by* hsame*, then* $\mathsf{IntClassifierSpec}((s_x, h'_x), (s_y, h'_y))$*.*

Proof. Unpack the classifier witness into the two signed unary carrier witnesses, the sign sameness component, and the magnitude sameness component. Transport both carriers across the new magnitude links, keep the sign sameness unchanged, and compose the magnitude links through the original classifier link. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec hsame magnitude transport`

Theorem 31.21 (Integer classifier sign transport)**.** *If* $\mathsf{IntClassifierSpec}((s_x, h_x), (s_y, h_y))$*, and the two signs transport to* s'_x *and* s'_y *by* msame*, then* $\mathsf{IntClassifierSpec}((s'_x, h_x), (s'_y, h_y))$*.*

Proof. Unpack the classifier witness into the two signed unary carrier witnesses, the sign sameness component, and the magnitude sameness component. Transport both carriers across the given sign links, compose the sign links through the original classifier link, and keep the magnitude sameness unchanged. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec msame sign transport`

Theorem 31.22 (Integer classifier sign and magnitude transport)**.** *If* $\mathsf{IntClassifierSpec}((s_x, h_x), (s_y, h_y))$*, the signs transport to* s'_x *and* s'_y *by* msame*, and the magnitudes transport to* h'_x *and* h'_y *by* hsame*, then* $\mathsf{IntClassifierSpec}((s'_x, h'_x), (s'_y, h'_y))$*.*

Proof. First transport the classifier along the two magnitude links, preserving the sign component. Then transport the resulting classifier along the two sign links, preserving the transported magnitude component. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec sign magnitude transport`

Theorem 31.23 (Integer classifier reflexivity)**.** *If* $\mathsf{IntCarrier}(s, h)$*, then* $\mathsf{IntClassifierSpec}((s, h), (s, h))$*.*

Proof. Use the same signed unary carrier witness on both classifier endpoints. The sign component is related by reflexivity of msame, and the magnitude component is related by reflexivity of hsame. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec refl`

Theorem 31.24 (Integer classifier symmetry)**.** *If* $\mathsf{IntClassifierSpec}(x, y)$*, then* $\mathsf{IntClassifierSpec}(y, x)$*.*

Proof. Eliminating the classifier witness exposes carrier evidence for both representatives, sign sameness, and magnitude sameness. The carrier evidence is swapped, while symmetry of msame and hsame reverses the two sameness components. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec symm`

Theorem 31.25 (Integer classifier transitivity)**.** *If* $\mathsf{IntClassifierSpec}(x, y)$ *and* $\mathsf{IntClassifierSpec}(y, z)$*, then* $\mathsf{IntClassifierSpec}(x, z)$*.*

Proof. Eliminating the two classifier witnesses exposes the carrier evidence for the endpoints, together with sign sameness and magnitude sameness along both links. Transitivity of msame composes the sign components, and transitivity of hsame composes the magnitude components. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec trans`

Theorem 31.26 (Integer classifier continuation with the same sign)**.** *If two integer classifier witnesses relate* (s_x, h_x) *to* (s'_x, h'_x) *and* (s_y, h_y) *to* (s'_y, h'_y)*, and the left signs* s_x *and* s_y *are mark-same, then continuation of the two left magnitudes and continuation of the two right magnitudes yield another integer classifier witness.*

Proof. Unpack the classifier witnesses into carrier data, sign sameness, and magnitude sameness. Transport the second left carrier along $\mathsf{msame}(s_x, s_y)$, and transport the second right carrier along the composed sign link $s'_x \sim s_x \sim s_y \sim s'_y$. Carrier closure under same-sign continuation gives the two result carriers, while continuation respects the two magnitude hsame links and gives the result magnitude sameness. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec continuation closed same sign`

Theorem 31.27 (Integer classifier append commutativity with the same sign)**.** *For a fixed sign* $s : \mathsf{BMark}$*, if* h *and* k *are both carried by* $\mathsf{IntCarrier}(s, -)$*, then the two append magnitudes classify each other under the same sign:*

$$\mathsf{IntClassifierSpec}\big((s, \mathsf{append}(h, k)), (s, \mathsf{append}(k, h))\big).$$

Proof. Unpack the two integer carrier witnesses. The shared sign evidence supplies both endpoint carriers. Unary append closure gives the two appended magnitude carriers, reflexivity of msame relates the fixed sign to itself, and unary append commutativity gives $\mathsf{append}(h, k)\mathsf{hsameappend}(k, h)$. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec append comm same sign`

Theorem 31.28 (Integer classifier swaps same-sign append context)**.** *For a fixed sign s : BMark, if $left$, $right$, and a are all carried by* IntCarrier$(s, -)$*, then the classifier identifies the two same-sign context displays*

$$\mathsf{append}(left, \mathsf{append}(a, right))$$

and

$$\mathsf{append}(right, \mathsf{append}(a, left)).$$

The ten-field fit is: traditional target signed-unary context commutativity; BEDC source IntCarrier*,* IntClassifierSpec*, and* append*; carrier same-sign unary representatives; classifier* IntClassifierSpec*; stability from unary append closure and append associativity; ledger the two context magnitudes and the middle magnitude; core theorem this block; exactness same-sign context swap; standard bridge ordinary commutativity of finite unary context addition; status* ProofTarget*.*

Proof. Unpack the three integer carrier witnesses to obtain unary witnesses for the two contexts and the middle magnitude. Unary append closure supplies the two endpoint carrier witnesses. Reassociate the left display, commute the relevant unary append factors, and reassociate back to the right display. The shared sign is related to itself by reflexivity of msame, and the constructed append equality is the magnitude component of the classifier. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec append context swap same sign`

Theorem 31.29 (Integer classifier appends same-sign context)**.** *For a fixed sign s : BMark, if $left$ and $right$ are carried by* IntCarrier$(s, -)$*, and* IntClassifierSpec$((s, a), (s, b))$*, then the classifier identifies the common-context magnitudes:*

$$\mathsf{IntClassifierSpec}((s, \mathsf{append}(left, \mathsf{append}(a, right))), (s, \mathsf{append}(left, \mathsf{append}(b, right)))).$$

Proof. Unpack the classifier witness into signed carrier evidence for a and b, the fixed-sign component, and the magnitude equality $a\mathsf{hsame}b$. Same-sign unary append closure builds the two contextual endpoint carriers. Replacing the middle magnitude along $a\mathsf{hsame}b$ leaves the two contextual append histories equal, while sign reflexivity supplies the sign classifier. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec append context same sign`

Theorem 31.30 (Integer classifier cancels same-sign append context)**.** *For a fixed sign s : BMark, if a and b are carried by* IntCarrier$(s, -)$*, and the integer classifier identifies the two common-context magnitudes*

$$\mathsf{append}(left, \mathsf{append}(a, right)) \quad \textit{and} \quad \mathsf{append}(left, \mathsf{append}(b, right)),$$

then IntClassifierSpec$((s, a), (s, b))$*.*

Proof. The contextual classifier exposes a BEDC history equality between the two append histories. Left append cancellation removes the common left context, and right append cancellation removes the common suffix. The resulting $a\mathsf{hsame}b$, together with the supplied endpoint carriers and reflexivity of msame(s, s), gives the integer classifier. □

Lean4 ✓ `BEDC.Derived.IntUp.IntClassifierSpec cancel append context same sign`

Theorem 31.31 (Integer classifier same-sign common context iff)**.** *For a fixed sign s : BMark, suppose that $left$, $right$, a, and b are all carried by $\mathsf{IntCarrier}(s, -)$. Then common same-sign append context is exact for the integer classifier:*

$$\mathsf{IntClassifierSpec}((s, \mathsf{append}(left, \mathsf{append}(a, right))), (s, \mathsf{append}(left, \mathsf{append}(b, right)))) \iff \mathsf{IntClassifierSpec}((s, a)$$

The ten-field fit is: traditional target common-context exactness for signed unary integers; BEDC source $\mathsf{IntCarrier}$*,* $\mathsf{IntClassifierSpec}$*, and* append*; carrier same-sign unary representatives; classifier* $\mathsf{IntClassifierSpec}$*; stability/common-context exactness; ledger left and right context histories; core theorem this iff; exactness preservation plus reflection; standard bridge ordinary cancellation under equal integer context; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, assume the contextual classifier. Apply Theorem 31.30 with the given endpoint carrier witnesses for a and b to cancel the common left and right unary histories.

For the reverse direction, assume $\mathsf{IntClassifierSpec}((s, a), (s, b))$. Apply Theorem 31.29 with the given carrier witnesses for $left$ and $right$ to append the same unary context on both sides. □

31.2 Difference-pair context stability

The difference-pair presentation of $\mathsf{Int}^{\uparrow}$ must remain stable when both representatives receive the same positive unary context and the same negative unary context. This is the signed-pair form of ordinary quotient stability for the relation $(p, n) \sim (q, m)$ given by cross-addition sameness.

Theorem 31.32 (Integer pair classifier balanced append context)**.** *For a, b, p, n, q, m : BHist, if $\mathsf{Unary}(a)$, $\mathsf{Unary}(b)$, and $\mathsf{IntPairClassifier}((p, n), (q, m))$, then*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, p), \mathsf{append}(n, b)), (\mathsf{append}(a, q), \mathsf{append}(m, b))).$$

The ten-field fit is: traditional target signed-pair integer quotient stability under adding the same positive and negative contexts; BEDC source $\mathsf{IntPairCarrier}$*,* $\mathsf{IntPairClassifier}$*,* Unary*,* append*, and* hsame*; carrier difference pairs; classifier cross-addition sameness; stability balanced context extension; ledger the added positive and negative unary contexts; core theorem this block; exactness preservation of the quotient classifier; standard bridge ordinary $(p, n) \sim (q, m)$ stability under $(p + a, n + b)$; status* $\mathsf{ProofTarget}$*.*

Proof. Unpack the classifier witness into the two difference-pair carrier witnesses and the cross-addition comparison $\mathsf{append}(p, m)\mathsf{hsameappend}(q, n)$. The carrier witnesses give unary evidence for p, n, q, and m.

Unary append closure with the unary contexts a and b supplies all four endpoint carrier components: $\mathsf{append}(a, p)$, $\mathsf{append}(n, b)$, $\mathsf{append}(a, q)$, and $\mathsf{append}(m, b)$.

For the cross-addition component, reassociate the left endpoint by $\mathsf{append_assoc}$:

$$\mathsf{append}(\mathsf{append}(a, p), \mathsf{append}(m, b))\mathsf{hsameappend}(a, \mathsf{append}(\mathsf{append}(p, m), b)).$$

The right endpoint reassociates in the same way:

$$\mathsf{append}(\mathsf{append}(a, q), \mathsf{append}(n, b))\mathsf{hsameappend}(a, \mathsf{append}(\mathsf{append}(q, n), b)).$$

Transport the original comparison through the common left prefix a and the common right suffix b. Transitivity of hsame with the two reassociation steps gives the required cross-addition sameness, and together with the four unary endpoint components this repacks exactly as the displayed $\mathsf{IntPairClassifier}$. □

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier balanced append context`

Theorem 31.33 (Integer pair classifier balanced append context reflection)**.** *For* a, b, p, n, q, m : BHist*, if* Unary(a)*,* Unary(b)*, and*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a,p), \mathsf{append}(n,b)), (\mathsf{append}(a,q), \mathsf{append}(m,b))),$$

then IntPairClassifier$((p,n),(q,m))$*. The ten-field fit is: traditional target signed-pair quotient reflection under adding the same positive and negative contexts; BEDC source* IntPairCarrier*,* IntPairClassifier*,* Unary*,* append*,* hsame*, append associativity, append cancellation, and unary append factorization; carrier difference pairs; classifier cross-addition sameness; stability reflected balanced context cancellation; ledger the removable positive and negative unary contexts; core theorem this reflection block; exactness reflection of the quotient classifier; standard bridge ordinary cancellation of* $(p+a, n+b)$ *back to* (p,n)*; status* ProofTarget*.*

Proof. Unpack the displayed contextual classifier into its two carried endpoints and the cross-addition comparison between the contextual pairs. Unary append factorization applied to those carried endpoints, together with the supplied unary witnesses for a and b, recovers unary evidence for p, n, q, and m.

Reassociate the contextual cross-addition comparison on both sides. The left endpoint has the same hsame-shape as

$$\mathsf{append}(a, \mathsf{append}(\mathsf{append}(p,m), b)),$$

and the right endpoint has the same hsame-shape as

$$\mathsf{append}(a, \mathsf{append}(\mathsf{append}(q,n), b)).$$

Transitivity of hsame with these reassociation witnesses converts the displayed classifier comparison into a comparison with the common left prefix a and the common right suffix b exposed. Append-left cancellation removes the shared prefix and yields a comparison between $\mathsf{append}(\mathsf{append}(p,m),b)$ and $\mathsf{append}(\mathsf{append}(q,n),b)$.

Append-right cancellation then removes the shared suffix b, giving $\mathsf{append}(p,m)\mathsf{hsameappend}(q,n)$. Repack this recovered cross-addition sameness with the recovered carrier witnesses for (p,n) and (q,m) to obtain IntPairClassifier$((p,n),(q,m))$. □

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier balanced append context reflection`

Theorem 31.34 (Integer pair classifier balanced append context exactness)**.** *For* a, b, p, n, q, m : BHist*, if* Unary(a) *and* Unary(b)*, then*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a,p), \mathsf{append}(n,b)), (\mathsf{append}(a,q), \mathsf{append}(m,b))) \iff \mathsf{IntPairClassifier}((p,n),(q,m)).$$

The ten-field fit is: traditional target balanced signed-pair context exactness; BEDC source IntPairCarrier*,* IntPairClassifier*,* Unary*,* append*, and* hsame*; carrier difference pairs; classifier cross-addition sameness; stability common positive and negative unary context transport; ledger the removable positive unary context* a *and removable negative unary context* b*; core theorem this exactness block; exactness the displayed iff; standard bridge ordinary quotient cancellation under simultaneous numerator and denominator context; downstream consumer one-sided context exactness and* $\mathsf{Rat}^{\uparrow}$ *numerator-denominator context cancellation; status* ProofTarget*.*

Proof. For the forward direction, assume the contextual classifier. The hypotheses $\mathsf{Unary}(a)$ and $\mathsf{Unary}(b)$ put the displayed classifier in the scope of Theorem 31.33, which removes the common positive and negative contexts and yields $\mathsf{IntPairClassifier}((p,n),(q,m))$.

For the reverse direction, assume $\mathsf{IntPairClassifier}((p,n),(q,m))$. Applying Theorem 31.32 with the same unary witnesses for a and b transports the classifier through the balanced append context, giving exactly the contextual classifier in the displayed statement. □

Theorem 31.35 (Integer pair classifier one-sided append context exactness)**.** *For* a,p,n,q,m : BHist*, if* $\mathsf{Unary}(a)$*, then both one-sided context laws are exact:*

1. *positive counters:*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a,p),n),(\mathsf{append}(a,q),m)) \iff \mathsf{IntPairClassifier}((p,n),(q,m));$$

2. *negative counters:*

$$\mathsf{IntPairClassifier}((p,\mathsf{append}(n,a)),(q,\mathsf{append}(m,a))) \iff \mathsf{IntPairClassifier}((p,n),(q,m)).$$

The ten-field fit is: traditional target signed-pair integer quotient one-sided context exactness; BEDC source $\mathsf{IntPairCarrier}$*,* $\mathsf{IntPairClassifier}$*,* Unary*,* append*, and* hsame*; carrier difference pairs; classifier cross-addition sameness; stability positive and negative one-sided unary contexts; ledger the common unary context* a *and the empty opposite context; core theorem this pair of equivalences; exactness preservation and reflection through balanced contexts plus componentwise transport; standard bridge ordinary cancellation after adding the same positive or negative integer counter; status* $\mathsf{ProofTarget}$*.*

Proof. For the positive forward implication, assume the contextual classifier on $((\mathsf{append}(a,p),n),(\mathsf{append}(a,q),m))$. The empty history is unary, and the continuation unit laws identify n and m with $\mathsf{append}(n,\mathsf{e})$ and $\mathsf{append}(m,\mathsf{e})$. The componentwise transport field of Theorem 31.16 therefore puts the witness into the balanced-context shape with positive context a and negative context e.

Apply Theorem 31.33 to that balanced-context witness. It cancels the common positive context and the empty negative context, giving $\mathsf{IntPairClassifier}((p,n),(q,m))$.

For the positive reverse implication, assume $\mathsf{IntPairClassifier}((p,n),(q,m))$. Apply Theorem 31.14 with the unary context a to obtain the displayed positive-context classifier.

For the negative forward implication, assume the contextual classifier on $((p,\mathsf{append}(n,a)),(q,\mathsf{append}(m,a)))$. The continuation unit laws identify p and q with $\mathsf{append}(\mathsf{e},p)$ and $\mathsf{append}(\mathsf{e},q)$. Componentwise transport places the witness into the balanced-context shape with positive context e and negative context a.

Apply Theorem 31.33 to cancel those balanced contexts. The result is exactly $\mathsf{IntPairClassifier}((p,n),(q,m))$.

For the negative reverse implication, start from $\mathsf{IntPairClassifier}((p,n),(q,m))$. Apply Theorem 31.32 with the empty positive context and the unary negative context a, then use the continuation unit laws and componentwise transport to replace $\mathsf{append}(\mathsf{e},p)$ and $\mathsf{append}(\mathsf{e},q)$ by p and q. □

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier one sided append context exactness`

Theorem 31.36 (Integer pair classifier iterated balanced append context exactness)**.** *For* a,b,c,d,p,n,q,m : BHist*, if* $\mathsf{Unary}(a)$*,* $\mathsf{Unary}(b)$*,* $\mathsf{Unary}(c)$*, and* $\mathsf{Unary}(d)$*, then*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a,\mathsf{append}(c,p)),\mathsf{append}(\mathsf{append}(n,d),b)),(\mathsf{append}(a,\mathsf{append}(c,q)),\mathsf{append}(\mathsf{append}(m,d),b$$

The ten-field fit is: traditional target stacked signed-pair quotient context exactness; BEDC source IntPairClassifier*,* Unary*,* append*, checked balanced preservation, and checked balanced reflection; carrier difference pairs; classifier cross-addition sameness; stability stacked positive and negative unary context transport; ledger the two positive unary contexts a, c and the two negative unary contexts d, b; core theorem this stacked exactness block; exactness the displayed iff; standard bridge ordinary quotient cancellation for two balanced signed-pair context layers; downstream consumer dyadic-rational mantissa and denominator context normalization; status* ProofTarget*.*

Proof. For the forward direction, assume the displayed contextual classifier. Regard it as a balanced context with outer positive context a, outer negative context b, and inner endpoints $((\mathsf{append}(c,p), \mathsf{append}(n,d)), (\mathsf{append}(c,q), \mathsf{append}(m,d)))$. Apply Theorem 31.33 to remove the outer contexts and yield the inner contextual classifier.

The inner classifier is itself in balanced-context shape with positive context c and negative context d. Applying Theorem 31.33 again cancels this second layer and gives $\mathsf{IntPairClassifier}((p,n),(q,m))$.

For the reverse direction, assume $\mathsf{IntPairClassifier}((p,n),(q,m))$. Apply Theorem 31.32 with contexts c and d to transport the classifier to $((\mathsf{append}(c,p), \mathsf{append}(n,d)), (\mathsf{append}(c,q), \mathsf{append}(m,d)))$.

Applying Theorem 31.32 once more with contexts a and b adds the outer layer, producing exactly the stacked contextual classifier displayed in the statement. □

Definition 31.37 (Int signed-pair carrier)**.** *The root signed-pair carrier for* $\mathsf{Int}^{\uparrow}$ *is the displayed Nat-and-Add row*

$$\mathsf{IntSignedPairCarrier}(a,b) := \mathsf{IntPairCarrier}(a,b),$$

where a and b are unary $\mathsf{Nat}^{\uparrow}$ *magnitude rows and the balancing classifier between (a,b) and (c,d) is read by the additive certificate as the cross-addition comparison $a + d = c + b$. In BEDC notation this is the existing row* $\mathsf{IntPairClassifier}((a,b),(c,d))$*, whose comparison is routed through* $\mathsf{append}(a,d)$ *and* $\mathsf{append}(c,b)$ *under the* $\mathsf{Add}^{\uparrow}$ *continuation ledger.*

Theorem 31.38 (Int signed-pair classifier stability)**.** *The classifier of Definition 31.37 is stable under componentwise* hsame *transport of the four unary magnitude rows and under displayed* Cont *replay of the two cross-addition ledgers. Thus an accepted comparison $(a,b) \sim (c,d)$ remains the same* $\mathsf{Int}^{\uparrow}$ *classifier row after the Nat magnitudes and Add balance witnesses are transported through the local finite ledger.*

Proof. Project the signed-pair carrier through Definition 31.9 and the classifier through Definition 31.12. The carrier rows are unary Nat magnitudes, and the classifier row is the single hsame comparison between the two cross-addition histories.

Transport the four magnitude rows componentwise. The hsame rows preserve the displayed unary histories, while the Cont replay rows rebuild the two additive histories $\mathsf{append}(a,d)$ and $\mathsf{append}(c,b)$ in the transported context.

Apply the balanced-context stability already recorded for the Int pair classifier. The resulting row is again an IntPairClassifier comparison, so no host quotient, primitive subtraction, ordered-ring law, or external integer equality has been introduced. □

Theorem 31.39 (Int additive inverse ledger)**.** *Every accepted* $\mathsf{Int}^{\uparrow}$ *signed-pair row (a,b) carries a displayed inverse ledger row obtained by swapping the two unary components to (b,a). The ledger is finite-row data: it records the Nat magnitudes, the Add balance context, componentwise* hsame *transport, and the local NameCert route, but it does not assert a full group law or ordered-ring interface for* $\mathsf{Int}^{\uparrow}$*.*

Proof. Start from the signed-pair carrier. Both a and b are unary Nat rows, so the swapped pair (b, a) is again admitted by the same carrier definition.

The inverse ledger is the displayed swap of the two coordinates. Its compatibility with the classifier is read through the same cross-addition pattern: the comparison for a pair and its swapped companion is expressed by the Add continuation ledger, not by a primitive subtraction operation.

The structural rows then transport and replay this swap inside the local packet. Since the chapter has no row for a full ordered ring, the theorem records only the inverse ledger surface needed by downstream rational and dyadic consumers. □

Theorem 31.40 (Int Rat and Dyadic source handoff)**.** *Every* $\mathsf{Rat}^{\uparrow}$ *numerator read and every* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa read that consumes* $\mathsf{Int}^{\uparrow}$ *data first receives the signed-pair carrier of Definition 31.37, the Nat magnitude rows, the Add balance ledger, and the inverse ledger of Definition 31.39. The handoff supplies signed mantissa data only as a finite* $\mathsf{Int}^{\uparrow}$ *row before* $\mathsf{Rat}^{\uparrow}$ *denominator normalization,* $\mathsf{DyadicRatCore}^{\uparrow}$ *routing,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal can be inspected.*

Proof. Read the rational boundary first. The $\mathsf{Rat}^{\uparrow}$ carrier consumes an integer numerator row, and the Int chapter exposes that numerator as the signed-pair carrier rather than as a host integer quotient.

Next use the dyadic mantissa route of Definition 31.78. It places the same signed row before the $\mathsf{DyadicRatCore}^{\uparrow}$ mantissa and denominator packet, so dyadic consumers inherit the Nat magnitude and Add balance data already displayed by $\mathsf{Int}^{\uparrow}$.

Finally pass to the L10 route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Those consumers can inspect only the finite signed mantissa handoff and its structural ledger; they receive no primitive subtraction row, full ordered-ring closure, quotient rational equality, selected real limit, or ambient completeness witness. □

31.3 The certificate

Definition 31.41 (Integer history carrier)**.** *The public BEDC-history carrier for* $\mathsf{Int}^{\uparrow}$ *packages the signed unary representative into the visible history tag:*

$$\mathsf{IntHistoryCarrier}(h) := (\exists m : \mathsf{BHist.\ Unary}(m) \wedge \mathsf{hsame}(h, \mathsf{E0}(m))) \vee (\exists m : \mathsf{BHist.\ Unary}(m) \wedge \mathsf{hsame}(h, \mathsf{E1}(m))).$$

The $\mathsf{E0}$ *branch and the* $\mathsf{E1}$ *branch encode the two signed unary presentations as actual BEDC histories.*

Lean4 (definition) `BEDC.Derived.IntUp.IntHistoryCarrier`

The visible signed branch is consumed by chapter 37 as the signed numerator and denominator context, and later by the dyadic-rational carrier through the same $\mathsf{Int}^{\uparrow}$ signed-row surface.

Theorem 31.42 (Integer history carrier signed branch determinacy)**.** *For every history* $h : \mathsf{BHist}$*, if* $\mathsf{IntHistoryCarrier}(h)$*, then exactly one visible signed carrier branch supplies the witness:*

$$\Big(\exists m : \mathsf{BHist.\ Unary}(m) \wedge \mathsf{hsame}(h, \mathsf{E0}(m)) \wedge \forall p : \mathsf{BHist.\ Unary}(p) \to \mathsf{hsame}(h, \mathsf{E1}(p)) \to \mathsf{False}\Big)$$
$$\vee$$
$$\Big(\exists m : \mathsf{BHist.\ Unary}(m) \wedge \mathsf{hsame}(h, \mathsf{E1}(m)) \wedge \forall p : \mathsf{BHist.\ Unary}(p) \to \mathsf{hsame}(h, \mathsf{E0}(p)) \to \mathsf{False}\Big).$$

The ten-field fit is: traditional target signed-integer carrier readback; BEDC source BHist*,* hsame*,* E0*, and* E1*; carrier* IntHistoryCarrier*; classifier not needed; stability from constructor inversion and* hsame *laws; ledger visible sign tag plus unary payload; core theorem this carrier branch determinacy block; exactness exclusive branch determinacy; standard bridge ordinary signed unary integer tag readback; status* ProofTarget*.*

Proof. Split the defining disjunction of $\mathsf{IntHistoryCarrier}(h)$.

In the zero-tag branch, keep the unary payload m and the endpoint witness $\mathsf{hsame}(h, \mathsf{E0}(m))$. If a one-tag carrier witness for the same h were also available, its endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$ would compose with $\mathsf{hsame}(h, \mathsf{E0}(m))$, using symmetry on one side, to produce $\mathsf{hsame}(\mathsf{E0}(m), \mathsf{E1}(p))$. BEDC history constructor no-confusion rules out that mixed visible-tag equality.

In the one-tag branch, keep the unary payload and endpoint witness in the same way. A hypothetical zero-tag witness composes into $\mathsf{hsame}(\mathsf{E1}(m), \mathsf{E0}(p))$, again contradicting constructor no-confusion. Thus the carrier witness determines the visible signed branch before any classifier evidence is used. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryCarrier signed branch determinacy`

Definition 31.43 (Integer history classifier)**.** *The public classifier relates exactly same-tag signed unary histories:*

$$\mathsf{IntHistoryClassifier}(h, k) := (\exists m, n : \mathsf{BHist}.\ \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(h, \mathsf{E0}(m)) \wedge \mathsf{hsame}(k, \mathsf{E0}(n)) \wedge \mathsf{hsame}(m, n)) \vee$$

Lean4 (definition) `BEDC.Derived.IntUp.IntHistoryClassifier`

Theorem 31.44 (Integer history classifier carrier endpoints)**.** *For all histories $h, k : \mathsf{BHist}$, classifier evidence exposes both endpoints as public signed-history carriers:*

$$\forall h, k : \mathsf{BHist}, \quad \mathsf{IntHistoryClassifier}(h, k) \to \mathsf{IntHistoryCarrier}(h) \wedge \mathsf{IntHistoryCarrier}(k).$$

The ten-field fit is: traditional target domain coverage for signed-unary integer equality; BEDC source IntHistoryCarrier*,* IntHistoryClassifier*,* BHist*,* hsame*,* E0*, and* E1*; carrier the public signed-history carrier; classifier the two-branch public classifier; stability extraction of admitted endpoints from classifier evidence; ledger visible sign tag plus unary payload; core theorem this carrier-endpoint block; exactness classifier domain coverage; standard bridge equality only relates admitted signed representatives; status* ProofTarget*.*

Proof. Split the classifier witness into its two visible signed branches.

In the zero branch, expose payloads $m, n : \mathsf{BHist}$, the unary witnesses for m and n, and the endpoint witnesses $\mathsf{hsame}(h, \mathsf{E0}(m))$ and $\mathsf{hsame}(k, \mathsf{E0}(n))$. The carrier witness for h is the zero-tag IntHistoryCarrier branch with payload m, its unary witness, and the displayed endpoint witness. The carrier witness for k is packed in the same way using payload n.

In the one branch, expose the corresponding payloads, unary witnesses, and endpoint witnesses $\mathsf{hsame}(h, \mathsf{E1}(m))$ and $\mathsf{hsame}(k, \mathsf{E1}(n))$. Packing the one-tag carrier branch for h and for k gives the two conjuncts required by the statement. □

Theorem 31.45 (Integer history classifier mixed visible-tag exclusion)**.** *For all $m, n : \mathsf{BHist}$, the public integer-history classifier excludes opposite visible sign tags without requiring the displayed payloads themselves to be externally unary:*

$$\forall m, n : \mathsf{BHist}, \quad (\mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E1}(n)) \to \mathsf{False}) \wedge (\mathsf{IntHistoryClassifier}(\mathsf{E1}(m), \mathsf{E0}(n)) \to \mathsf{False}).$$

The ten-field fit is: traditional target signed-integer mixed-tag separation; BEDC source BHist*,* hsame*,* E0*,* E1*, and* IntHistoryClassifier*; carrier* IntHistoryCarrier*; classifier the public two-branch classifier; stability from constructor no-confusion and* hsame *composition; ledger visible sign tag plus hidden unary payload; core theorem this mixed visible-tag exclusion block; exactness mixed-tag exclusion; standard bridge ordinary signed-unary sign disjointness; status* ProofTarget*.*

Proof. Fix $m, n : \mathsf{BHist}$.

For the endpoint pair $\mathsf{E0}(m), \mathsf{E1}(n)$, split the defining disjunction of IntHistoryClassifier. In the zero-branch witness, the second endpoint is also exposed as some $\mathsf{E0}(q)$, while the displayed endpoint supplies $\mathsf{hsame}(\mathsf{E1}(n), \mathsf{E0}(q))$ up to symmetry. Composing these endpoint witnesses gives a mixed visible-tag history sameness judgment, contradicting BEDC history constructor no-confusion. In the one-branch witness, the first endpoint is also exposed as some $\mathsf{E1}(p)$, and the same composition gives $\mathsf{hsame}(\mathsf{E0}(m), \mathsf{E1}(p))$, again impossible by constructor no-confusion.

For the endpoint pair $\mathsf{E1}(m), \mathsf{E0}(n)$, use the symmetric case split. A zero-branch witness exposes the first endpoint as $\mathsf{E0}(p)$ and composes into $\mathsf{hsame}(\mathsf{E1}(m), \mathsf{E0}(p))$; a one-branch witness exposes the second endpoint as $\mathsf{E1}(q)$ and composes into $\mathsf{hsame}(\mathsf{E0}(n), \mathsf{E1}(q))$. Both conclusions are mixed visible-tag history sameness judgments, so constructor no-confusion closes the two cases. The unary witnesses hidden inside the classifier branches are used only to recognize the public branch witnesses; no unary hypothesis on the displayed payloads m, n is needed. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryClassifier mixed visible tag exclusion`

Theorem 31.46 (Integer history classifier visible branch exactness)**.** *For any* $m, n : \mathsf{BHist}$ *with* $\mathsf{Unary}(m)$ *and* $\mathsf{Unary}(n)$*, the public integer-history classifier reads back the same visible tag precisely as payload history sameness:*

$$(\mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E0}(n)) \iff \mathsf{hsame}(m, n)) \wedge (\mathsf{IntHistoryClassifier}(\mathsf{E1}(m), \mathsf{E1}(n)) \iff \mathsf{hsame}(m, n)).$$

The ten-field fit is: traditional target signed-integer visible-tag readback; BEDC source BHist*; admissible payloads* Unary*; payload relation* hsame*; public classifier* IntHistoryClassifier*; carrier the public integer history carrier; classifier same-tag payload sameness; stability by constructor inversion and* hsame *laws; ledger visible tag plus unary payload; exactness the two iff clauses; standard bridge ordinary signed-integer tag readback; status* ProofTarget*.*

Proof. For the zero-tag forward direction, split a witness of $\mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E0}(n))$ by the two displayed classifier branches. In the zero branch, constructor inversion on the endpoint witnesses reduces the visible histories to payload histories, and transitivity of hsame composes the two endpoint equalities with the stored payload comparison. In the one branch, the endpoint evidence would identify a zero tag with a one tag, contradicting history no-confusion.

The one-tag forward direction is symmetric. The matching one branch gives the payload comparison by constructor inversion and transitivity of hsame; the zero branch is ruled out by the same zero/one history no-confusion.

For the reverse directions, assume $\mathsf{hsame}(m, n)$ together with the displayed unary witnesses for m and n. The zero-tag classifier witness is obtained by choosing the zero branch, using the two unary witnesses, reflexive endpoint witnesses for $\mathsf{E0}(m)$ and $\mathsf{E0}(n)$, and the given payload relation. The one-tag case is identical with E1 in place of E0. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryClassifier visible branch exactness iff`

Theorem 31.47 (Integer history classifier signed branch determinacy)**.** *For all histories* h, k : BHist*, if* $\mathsf{IntHistoryClassifier}(h, k)$*, then exactly one visible signed branch supplies the classifier witnesses:*

$$\Big(\exists m, n : \mathsf{BHist}.\ \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(h, \mathsf{E0}(m)) \wedge \mathsf{hsame}(k, \mathsf{E0}(n)) \wedge \mathsf{hsame}(m, n)$$
$$\wedge\ \forall p, q : \mathsf{BHist}.\ \mathsf{Unary}(p) \to \mathsf{Unary}(q) \to \mathsf{hsame}(h, \mathsf{E1}(p)) \to \mathsf{hsame}(k, \mathsf{E1}(q)) \to \mathsf{hsame}(p, q) \to \mathsf{False}\Big)$$
$$\vee$$
$$\Big(\exists m, n : \mathsf{BHist}.\ \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(h, \mathsf{E1}(m)) \wedge \mathsf{hsame}(k, \mathsf{E1}(n)) \wedge \mathsf{hsame}(m, n)$$
$$\wedge\ \forall p, q : \mathsf{BHist}.\ \mathsf{Unary}(p) \to \mathsf{Unary}(q) \to \mathsf{hsame}(h, \mathsf{E0}(p)) \to \mathsf{hsame}(k, \mathsf{E0}(q)) \to \mathsf{hsame}(p, q) \to \mathsf{False}\Big).$$

The ten-field fit is: traditional target signed integer branch readback; BEDC source BHist*,* hsame*,* E0*,* E1*, and* IntHistoryClassifier*; carrier* IntHistoryCarrier*; classifier* IntHistoryClassifier*; stability inherited from Definition 31.48; ledger the visible sign tag and unary payload; core theorem this signed branch determinacy block; exactness signed branch determinacy; standard bridge ordinary signed-integer sign readback; status* ProofTarget*.*

Proof. Split the defining disjunction of $\mathsf{IntHistoryClassifier}(h, k)$.

In the zero-tag case, keep the exposed unary payloads m, n, the two visible endpoint witnesses $h\mathsf{hsameE0}(m)$ and $k\mathsf{hsameE0}(n)$, and the payload comparison $m\mathsf{hsame}n$. To rule out a one-tag witness for the same endpoints, compose $h\mathsf{hsameE0}(m)$ with the hypothetical $h\mathsf{hsameE1}(p)$, using symmetry on one side as needed. The result is a visible contradiction $\mathsf{hsame}(\mathsf{E0}(m), \mathsf{E1}(p))$ by BEDC constructor no-confusion.

In the one-tag case, keep the exposed unary payloads and endpoint witnesses in the same way. A hypothetical zero-tag witness for the same endpoints composes the endpoint comparisons into $\mathsf{hsame}(\mathsf{E1}(m), \mathsf{E0}(p))$, again contradicting BEDC constructor no-confusion. Thus the classifier witness determines the visible signed branch and leaves only the payload equality inside that branch. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryClassifier signed branch determinacy`

Theorem 31.48 (Concrete IntUp history semantic certificate)**.** *The integer history carrier and classifier form the public semantic naming certificate for the concrete signed unary history presentation:*

$$\mathsf{SemanticNameCert}(\mathsf{IntHistoryCarrier}, \mathsf{IntHistoryCarrier}, \mathsf{IntHistoryCarrier}, \mathsf{IntHistoryClassifier}).$$

Moreover the classifier excludes mixed visible tags:

$$\forall m, n : \mathsf{BHist}, \quad \mathsf{Unary}(m) \to \mathsf{Unary}(n) \to (\mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E1}(n)) \to \mathsf{False}) \wedge (\mathsf{IntHistoryClassifier}(\mathsf{E1}(m),$$

Proof. For inhabitance, choose the zero-tagged empty magnitude $\mathsf{E0}(\mathsf{e})$. The unary base witness gives $\mathsf{Unary}(\mathsf{e})$, and reflexivity of hsame gives the visible endpoint witness.

For reflexivity, split the carrier witness for h. In the E0 branch, reuse the same magnitude on both endpoints and use reflexivity of hsame for the payload comparison. The E1 branch is identical with the one-tagged constructor.

For symmetry, unpack the classifier branch. The two endpoint carrier witnesses are kept in the same visible tag, and symmetry of hsame reverses the payload comparison. Transitivity composes

two same-tag classifier witnesses by composing their payload hsame links through the shared middle payload.

For carrier transport, suppose $\mathsf{IntHistoryCarrier}(h)$ and $\mathsf{IntHistoryClassifier}(h, k)$. In a same-tag branch, compose the classifier endpoint witness for k with the stored unary payload witness, then repack the corresponding E0 or E1 carrier branch.

The semantic pattern and ledger fields are the carrier itself, so both soundness obligations are identity implications from a transported carrier witness. Mixed visible tags are impossible because any mixed classifier branch would compose the visible endpoint witnesses into an $\mathsf{hsame}(\mathsf{E0}(m'), \mathsf{E1}(n'))$ or $\mathsf{hsame}(\mathsf{E1}(m'), \mathsf{E0}(n'))$ judgment, contradicting BEDC constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistory semanticNameCert`

Theorem 31.49 (Integer history classifier same-tag readback)**.** *For all m, n : BHist, the public integer history classifier reads back exactly payload equality on each visible signed tag:*

$$\mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E0}(n)) \iff \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n).$$

The one-tag branch has the same payload criterion:

$$\mathsf{IntHistoryClassifier}(\mathsf{E1}(m), \mathsf{E1}(n)) \iff \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n).$$

The ten-field fit is the following. The traditional target is same-sign equality of signed integers. The BEDC source consists of IntHistoryCarrier*,* IntHistoryClassifier*,* hsame*, and constructor no-confusion. The carrier is the same-tag unary-history carrier. The classifier is the displayed two-branch definition. Stability is given by transport along the payload* hsame*. The ledger records the sign tag together with the unary magnitude. The core theorem is this same-tag readback statement. Exactness is the two displayed iff rows. The downstream consumer is the* $\mathsf{Rat}^{\uparrow}$ *integer numerator readback. The status is* ProofTarget*.*

Proof. Unfold IntHistoryClassifier at same-tag endpoints.

For the zero-tag forward direction, keep the matching E0 branch. The endpoint witnesses have the form $\mathsf{hsame}(\mathsf{E0}(m), \mathsf{E0}(m'))$ and $\mathsf{hsame}(\mathsf{E0}(n), \mathsf{E0}(n'))$, so constructor inversion reads back $\mathsf{hsame}(m, m')$ and $\mathsf{hsame}(n, n')$. Transitivity with the stored payload $\mathsf{hsame}(m', n')$ yields $\mathsf{hsame}(m, n)$, and the same branch supplies $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$ after the endpoint readback. The opposite E1 branch would force an E0 endpoint to be hsame to an E1 endpoint, contradicting constructor no-confusion.

For the zero-tag reverse direction, assume $\mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n)$. Choose the E0 branch with payload witnesses m and n, use reflexivity for the two visible endpoint witnesses, and use the supplied payload $\mathsf{hsame}(m, n)$.

The one-tag forward direction is identical after replacing E0 by E1. The matching branch gives endpoint inversion and payload transport; the mixed branch contradicts E1/E0 no-confusion.

For the one-tag reverse direction, choose the E1 branch with payload witnesses m and n, reflexive visible endpoints, the two unary witnesses, and the supplied payload $\mathsf{hsame}(m, n)$. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryClassifier same tag readback iff`

Theorem 31.50 (Integer canonical difference-pair/history bridge)**.** *For all m, n : BHist, if* $\mathsf{Unary}(m)$ *and* $\mathsf{Unary}(n)$*, then the canonical same-sign difference pairs read back exactly to the corresponding public signed-history classifier branches:*

$$(\mathsf{IntPairClassifier}((m, \mathsf{e}), (n, \mathsf{e})) \iff \mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E0}(n))) \wedge (\mathsf{IntPairClassifier}((\mathsf{e}, m), (\mathsf{e}, n)) \iff \mathsf{IntHisto}$$

The ten-field fit is the following. The traditional target is canonical signed unary integers. The BEDC source consists of IntPairClassifier*,* IntHistoryClassifier*,* BHist*,* hsame*,* append*,* E0*, and* E1*. The carrier is* IntPairCarrier *together with* IntHistoryCarrier*. The classifier is the displayed same-sign bridge. Stability is inherited from the append unit laws and same-tag readback. The ledger records the hidden empty counter and the visible sign tag. The core theorem is this bridge. Exactness is the two iff clauses. The downstream consumer is the* $\mathsf{Rat}^{\uparrow}$ *numerator readback and* $\mathsf{Int}^{\uparrow}$ *closure. The status is* ProofTarget*.*

Proof. Assume $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$. For the positive canonical pair, unfold IntPairClassifier at $((m, \mathsf{e}), (n, \mathsf{e}))$. The carrier clauses reduce to the unary witnesses for m, n, and the two empty counters. The classifier payload is $\mathsf{hsame}(\mathsf{append}(m, \mathsf{e}), \mathsf{append}(n, \mathsf{e}))$, and the right append unit laws identify this with $\mathsf{hsame}(m, n)$.

Apply Theorem 31.49 to the E0 branch. Its readback criterion is exactly the same pair of unary witnesses together with $\mathsf{hsame}(m, n)$, so the positive displayed $\iff$ follows.

For the negative canonical pair, unfold IntPairClassifier at $((\mathsf{e}, m), (\mathsf{e}, n))$. The cross-addition payload is $\mathsf{hsame}(\mathsf{append}(\mathsf{e}, n), \mathsf{append}(\mathsf{e}, m))$. The left append unit laws reduce this to $\mathsf{hsame}(n, m)$, and symmetry of hsame changes the orientation to $\mathsf{hsame}(m, n)$.

Apply Theorem 31.49 to the E1 branch. The same unary witnesses and the reoriented payload $\mathsf{hsame}(m, n)$ are precisely its public-history criterion, giving the negative displayed $\iff$. □

Theorem 31.51 (Integer contextual canonical difference-pair/history bridge)**.** *For* $a, m, n : \mathsf{BHist}$ *with* $\mathsf{Unary}(a)$*,* $\mathsf{Unary}(m)$*, and* $\mathsf{Unary}(n)$*, the positive-context canonical difference pair reads back exactly to the zero-tag public signed-history classifier:*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, m), \mathsf{e}), (\mathsf{append}(a, n), \mathsf{e})) \iff \mathsf{IntHistoryClassifier}(\mathsf{E0}(m), \mathsf{E0}(n)).$$

The negative-context canonical difference pair reads back exactly to the one-tag public signed-history classifier:

$$\mathsf{IntPairClassifier}((\mathsf{e}, \mathsf{append}(m, a)), (\mathsf{e}, \mathsf{append}(n, a))) \iff \mathsf{IntHistoryClassifier}(\mathsf{E1}(m), \mathsf{E1}(n)).$$

The ten-field fit is the following. The traditional target is contextual signed-unary integer readback. The BEDC source consists of IntPairClassifier*,* IntHistoryClassifier*,* Unary*,* append*,* e*,* E0*, and* E1*. The carrier is canonical difference pairs and public signed histories. The classifier is pair cross-addition plus the public same-tag classifier. Stability is one-sided context exactness. The ledger records the hidden common unary context. The core theorem is this bridge. Exactness is the two displayed iff rows. The standard bridge is ordinary cancellation of equal signed unary shifts. The status is* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier contextual canonical pair history bridge iff`

Proof. For the zero-tag row, apply Theorem 31.35 to the common positive context a with canonical endpoints $((m, \mathsf{e}), (n, \mathsf{e}))$. The hypothesis $\mathsf{Unary}(a)$ supplies the context witness for this exactness step. The positive-counter clause strips or introduces the shared left append context, yielding an equivalence with $\mathsf{IntPairClassifier}((m, \mathsf{e}), (n, \mathsf{e}))$.

Apply Theorem 31.50 to the positive canonical pair. The assumptions $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$ supply its endpoint hypotheses, so composition of the two equivalences gives the displayed zero-tag readback.

For the one-tag row, apply Theorem 31.35 to the common negative context a with canonical endpoints $((\mathsf{e}, m), (\mathsf{e}, n))$. The hypothesis $\mathsf{Unary}(a)$ supplies the context witness for this exactness

step. The negative-counter clause strips or introduces the shared right append context, yielding an equivalence with $\mathsf{IntPairClassifier}((\mathsf{e}, m), (\mathsf{e}, n))$.

Apply Theorem 31.50 to the negative canonical pair. The same unary endpoint hypotheses identify this canonical classifier with $\mathsf{IntHistoryClassifier}(\mathsf{E1}(m), \mathsf{E1}(n))$, completing the second displayed equivalence. □

Theorem 31.52 (Integer canonical same-sign pair readback)**.** *For all $m, n : \mathsf{BHist}$, the canonical positive difference-pair classifier reads back exactly to unary payload sameness:*

$$\mathsf{IntPairClassifier}((m, \mathsf{e}), (n, \mathsf{e})) \iff \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n).$$

The canonical negative difference-pair classifier has the same public payload criterion:

$$\mathsf{IntPairClassifier}((\mathsf{e}, m), (\mathsf{e}, n)) \iff \mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n).$$

The ten-field fit is the following. The traditional target is canonical positive and negative signed unary integer equality. The BEDC source consists of IntPairClassifier*,* IntPairCarrier*,* append*,* e*, and* hsame*. The carrier is the canonical difference-pair carrier. The classifier is cross-addition sameness. Stability is supplied by the append unit laws and symmetry of* hsame*. The ledger records the hidden empty counter. The core theorem is this direct readback. Exactness is the two displayed iff rows. The downstream consumer is the* $\mathsf{Rat}^{\uparrow}$ *numerator readback. The status is* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.IntUp.IntPairClassifier canonical same sign readback iff`

Proof. Unfold IntPairClassifier at the positive canonical endpoints $((m, \mathsf{e}), (n, \mathsf{e}))$.

For the positive forward direction, the carrier projections give $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$, with the two hidden counters carried by e. The classifier payload is

$$\mathsf{hsame}(\mathsf{append}(m, \mathsf{e}), \mathsf{append}(n, \mathsf{e})).$$

The right append unit laws reduce this payload to $\mathsf{hsame}(m, n)$.

For the positive reverse direction, assume $\mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n)$. Repack the two canonical carriers from the unary witnesses for m and n together with the empty unary carrier for e. The supplied payload $\mathsf{hsame}(m, n)$ transports through the right append unit laws to the cross-addition payload required by IntPairClassifier.

Unfold IntPairClassifier at the negative canonical endpoints $((\mathsf{e}, m), (\mathsf{e}, n))$.

For the negative forward direction, the carrier projections again give $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$. The classifier payload is

$$\mathsf{hsame}(\mathsf{append}(\mathsf{e}, n), \mathsf{append}(\mathsf{e}, m)).$$

The left append unit laws reduce it to $\mathsf{hsame}(n, m)$, and symmetry of hsame orients the conclusion as $\mathsf{hsame}(m, n)$.

For the negative reverse direction, assume $\mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m, n)$. Symmetry first turns the payload into $\mathsf{hsame}(n, m)$, and the left append unit laws transport it to

$$\mathsf{hsame}(\mathsf{append}(\mathsf{e}, n), \mathsf{append}(\mathsf{e}, m)).$$

Together with the two canonical carriers, this is exactly the negative instance of IntPairClassifier. □

Theorem 31.53 (Integer canonical opposite-sign zero collapse)**.** *For all $m, n : \mathsf{BHist}$, if $\mathsf{Unary}(m)$ and $\mathsf{Unary}(n)$, then the opposite-sign canonical difference pair is classified only at zero:*

$$\mathsf{IntPairClassifier}((m, \mathsf{e}), (\mathsf{e}, n)) \iff \mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e}).$$

The ten-field fit is the following. The traditional target is opposite-sign canonical integer equality. The BEDC source consists of IntPairClassifier*,* IntPairCarrier*,* append*,* e*, and* hsame*. The carrier is the unary difference-pair carrier. The classifier is cross-addition sameness. Stability is supplied by append empty inversion and history empty readback. The ledger records the hidden zero counters. The core theorem is this zero-collapse block. Exactness is canonical opposite-sign collapse. The standard bridge is the ordinary fact that $m + n = 0$ for unary naturals forces both summands to be zero. The status is* ProofTarget*.*

Proof. Assume first that m and n are unary. Unfold IntPairClassifier at $((m, \mathsf{e}), (\mathsf{e}, n))$. The carrier clauses come from the unary hypotheses together with the empty-counter carrier facts. The classifier payload is

$$\mathsf{hsame}(\mathsf{append}(m, n), \mathsf{append}(\mathsf{e}, \mathsf{e})).$$

The append unit law reduces the right endpoint to e.

By empty readback for hsame, the payload gives $\mathsf{append}(m, n) = \mathsf{e}$. Apply append empty inversion to obtain $m = \mathsf{e}$ and $n = \mathsf{e}$. These equalities yield $\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$ by reflexivity and transport.

Conversely, assume $\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$ together with the unary hypotheses. Empty readback turns the two displayed readbacks into $m = \mathsf{e}$ and $n = \mathsf{e}$. Since e is unary, both endpoints (m, e) and (e, n) satisfy IntPairCarrier after transport.

It remains to build the cross-addition payload. Transport along the two empty equalities and the append unit laws reduces it to $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, which is reflexive. Repacking the two carriers and this payload gives $\mathsf{IntPairClassifier}((m, \mathsf{e}), (\mathsf{e}, n))$. □

Theorem 31.54 (Integer zero sign-refined boundary)**.** *The difference-pair zero classifier and the visible history classifier expose different interfaces at zero. The pair-zero class is classified:*

$$\mathsf{IntPairClassifier}((\mathsf{e}, \mathsf{e}), (\mathsf{e}, \mathsf{e})).$$

At the same time, the signed-history classifier rejects the two mixed visible zero displays:

$$(\mathsf{IntHistoryClassifier}(\mathsf{E0}(\mathsf{e}), \mathsf{E1}(\mathsf{e})) \to \mathsf{False}) \wedge (\mathsf{IntHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E0}(\mathsf{e})) \to \mathsf{False}).$$

Consequently, a classifier-reflecting bridge from difference pairs to signed histories must either remain inside one visible branch or choose a canonical zero branch before reflecting the pair-zero class.

Proof. The empty history is unary, so the pair (e, e) satisfies $\mathsf{IntPairCarrier}(\mathsf{e}, \mathsf{e})$. Apply the reflexivity component of Definition 31.16 to this carrier witness. This gives $\mathsf{IntPairClassifier}((\mathsf{e}, \mathsf{e}), (\mathsf{e}, \mathsf{e}))$.

For the signed-history clauses, instantiate Definition 31.45 with $m = \mathsf{e}$ and $n = \mathsf{e}$. The two conjuncts are exactly the exclusions for $\mathsf{IntHistoryClassifier}(\mathsf{E0}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$ and $\mathsf{IntHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E0}(\mathsf{e}))$.

These two facts describe distinct public interfaces. The pair quotient records the integer zero class, while the history certificate keeps the visible sign tag in the endpoint shape unless a bridge explicitly selects a same-branch zero representative. □

Theorem 31.55 (Integer balanced contextual opposite-sign zero collapse)**.** *For all* a, b, m, n : BHist*, if* $\mathsf{Unary}(a)$*,* $\mathsf{Unary}(b)$*,* $\mathsf{Unary}(m)$*, and* $\mathsf{Unary}(n)$*, then the balanced contextual opposite-sign pair collapses exactly at the two zero counters:*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, m), \mathsf{append}(\mathsf{e}, b)), (\mathsf{append}(a, \mathsf{e}), \mathsf{append}(n, b))) \iff \mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e}).$$

The ten-field fit is the following. The traditional target is contextual opposite-sign integer zero collapse. The BEDC source consists of $\mathsf{IntPairClassifier}$*,* Unary*,* append*,* e*, balanced context exactness, and canonical opposite-sign zero collapse. The carrier is carried by unary difference pairs. The classifier is cross-addition sameness. Stability is removal of the shared positive and negative contexts. The ledger records hidden common contexts* a, b *together with zero counters. The core theorem is this balanced contextual zero-collapse block. Exactness is the displayed iff. The standard bridge is ordinary cancellation of equal added contexts before the fact that a unary sum is zero only when both summands are zero. The status is* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, assume the contextual classifier. Apply Theorem 31.34 with $p = m$, $n = \mathsf{e}$, $q = \mathsf{e}$, and $m = n$. The unary hypotheses for a and b remove the balanced positive and negative contexts and reduce the classifier to

$$\mathsf{IntPairClassifier}((m, \mathsf{e}), (\mathsf{e}, n)).$$

Theorem 31.53, applied with the unary hypotheses for m and n, converts this canonical opposite-sign classifier into $\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$.

For the reverse direction, assume $\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$. The canonical opposite-sign zero-collapse theorem first gives $\mathsf{IntPairClassifier}((m, \mathsf{e}), (\mathsf{e}, n))$ from the unary hypotheses for m and n.

Apply Theorem 31.34 again with $p = m$, $n = \mathsf{e}$, $q = \mathsf{e}$, and $m = n$. Reading this equivalence in the reverse direction, the unary hypotheses for a and b transport the canonical classifier into the displayed balanced contextual classifier. □

Theorem 31.56 (Integer contextual opposite-sign zero collapse)**.** *For all* a, m, n : BHist*, if* $\mathsf{Unary}(a)$*,* $\mathsf{Unary}(m)$*, and* $\mathsf{Unary}(n)$*, then the visible-context opposite-sign canonical pair collapses exactly at the shared zero context and zero payloads:*

$$\mathsf{IntPairClassifier}((\mathsf{append}(a, m), \mathsf{e}), (\mathsf{e}, \mathsf{append}(n, a))) \iff \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e}).$$

The ten-field fit is the following. The traditional target is contextual opposite-sign signed-unary zero detection. The BEDC source consists of $\mathsf{IntPairClassifier}$*,* Unary*,* append*,* e*, and* hsame*. The carrier is unary difference pairs. The classifier is cross-addition sameness. Stability is append-empty inversion. The ledger records the shared context together with the two payloads. The core theorem is this contextual zero-collapse block. Exactness is the displayed iff. The standard bridge is the ordinary fact that* $a + m + n + a = 0$ *for unary summands forces all displayed summands to be zero. The status is* $\mathsf{ProofTarget}$*.*

Proof. Assume first the three unary hypotheses and the displayed classifier. Unary append closure makes $\mathsf{append}(a, m)$ and $\mathsf{append}(n, a)$ unary, so Theorem 31.53 applies to the canonical pair built from these two appended counters.

The forward half of that theorem gives $\mathsf{hsame}(\mathsf{append}(a, m), \mathsf{e})$ and $\mathsf{hsame}(\mathsf{append}(n, a), \mathsf{e})$. Empty readback converts these into two empty append equations, and append-empty inversion yields $a = \mathsf{e}$, $m = \mathsf{e}$, and $n = \mathsf{e}$ after reading the shared context from both equations. Transport along these empty equations gives $\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$.

Conversely, assume $\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(n, \mathsf{e})$. Empty readback turns the three readbacks into empty equations for a, m, and n. The append unit laws therefore identify both contextual counters $\mathsf{append}(a, m)$ and $\mathsf{append}(n, a)$ with e.

The reverse half of Theorem 31.53 now applies to these two unary appended counters. Its required empty readbacks are obtained by the preceding transport to e, and the resulting canonical classifier is exactly $\mathsf{IntPairClassifier}((\mathsf{append}(a, m), \mathsf{e}), (\mathsf{e}, \mathsf{append}(n, a)))$. □

Theorem 31.57 (Integer signed-unary standard bridge)**.** *The ordinary signed-unary integer presentation admits a conservative* $\mathsf{StandardBridge}(\mathsf{Int}^{\uparrow}, T_{\mathbb{Z}})$ *to the BEDC public history surface with the following six fields.*

1. *The encode field sends a positive signed-unary magnitude m to* $\mathsf{E0}(m)$ *and a negative signed-unary magnitude m to* $\mathsf{E1}(m)$*.*
2. *The decode field reads the visible tag and unary payload from* $\mathsf{IntHistoryCarrier}$*.*
3. *Carrier reflection is supplied by Definition 31.48.*
4. *Classifier reflection is supplied by Definition 31.49 together with Definition 31.45.*
5. *The theorem-preservation field is supplied by the canonical and contextual difference-pair/history bridge rows Definition 31.50, Definition 31.51, Definition 31.52, Definition 31.53, Definition 31.55, and Definition 31.56.*
6. *The no-host-leak field records that no host equality, quotient, host* Nat*, ordered-ring package, division, or normalization theorem enters the BEDC public surface; all public comparisons remain* $\mathsf{IntHistoryClassifier}$*,* $\mathsf{IntPairClassifier}$*, and* hsame *statements over BEDC histories.*

Lean4 ✓ `BEDC.Derived.IntUp.IntUp StdBridge`

Proof. The encode field is just the public constructor choice for signed unary histories: zero-tag histories carry the positive branch and one-tag histories carry the negative branch. The decode field is the inverse reading of $\mathsf{IntHistoryCarrier}$, whose public witnesses expose exactly a visible tag and a unary payload.

Carrier reflection comes from Definition 31.48. It gives the semantic certificate for the carrier and classifier, so the bridge reads and writes only through the public BEDC history surface.

Classifier reflection is the same-tag readback of Definition 31.49. The mixed visible-tag exclusion of Definition 31.45 rules out the two cross-tag round trips, leaving exactly the signed-unary payload comparison in each visible branch.

The theorem-preservation field is obtained by composing the canonical bridge row, the contextual bridge row, the canonical same-sign readback row, and the opposite-sign zero-collapse rows already proved in this file. These rows cover the canonical same-sign, contextual same-sign, canonical opposite-sign, balanced contextual opposite-sign, and visible-context opposite-sign cases used by the ordinary signed-unary presentation.

No host structure is imported into the public BEDC object during this composition. The ordinary presentation is used only as the reader-facing source of the encode and decode descriptions; the exported carrier, classifier, transport, preservation, and exclusion statements remain the BEDC history and difference-pair statements cited above. □

Theorem 31.58 (Integer history classifier positive append context iff)**.** *For all* $a, m, n : \mathsf{BHist}$*, if* $\mathsf{Unary}(a)$*,* $\mathsf{Unary}(m)$*, and* $\mathsf{Unary}(n)$*, then the public same-sign integer history classifier is invariant*

under adding the same visible unary prefix to both magnitudes:

$$\mathsf{IntHistoryClassifier}(\mathsf{E0}(\mathsf{append}(a,m)),\mathsf{E0}(\mathsf{append}(a,n))) \iff \mathsf{IntHistoryClassifier}(\mathsf{E0}(m),\mathsf{E0}(n)).$$

The one-tag branch satisfies the parallel context law:

$$\mathsf{IntHistoryClassifier}(\mathsf{E1}(\mathsf{append}(a,m)),\mathsf{E1}(\mathsf{append}(a,n))) \iff \mathsf{IntHistoryClassifier}(\mathsf{E1}(m),\mathsf{E1}(n)).$$

The ten-field fit is the following. The traditional target is signed-integer equality under a common unary magnitude context. The BEDC source consists of IntHistoryClassifier*,* Unary*,* append*,* hsame*, and append cancellation. The carrier is the public signed-history carrier. The classifier is the same-tag public classifier. Stability is common unary prefix invariance. The ledger records the common visible unary prefix. The core theorem is this iff statement. Exactness is preservation and reflection of same-tag classifier equality. The standard bridge is ordinary signed unary equality under equal magnitude shifts. The status is* ProofTarget*.*

Proof. Apply Theorem 31.49 to the E0 endpoints. With the unary closure of $\mathsf{append}(a,m)$ and $\mathsf{append}(a,n)$, the left side reads back to $\mathsf{hsame}(\mathsf{append}(a,m),\mathsf{append}(a,n))$ together with the two unary payload witnesses. The right side reads back to $\mathsf{Unary}(m) \wedge \mathsf{Unary}(n) \wedge \mathsf{hsame}(m,n)$.

For the zero-tag reverse direction, use congruence of $\mathsf{append}(a,-)$ on $\mathsf{hsame}(m,n)$ and unary append closure for the two payloads. Repacking by the same-tag readback theorem gives the classifier on $\mathsf{E0}(\mathsf{append}(a,m))$ and $\mathsf{E0}(\mathsf{append}(a,n))$.

For the zero-tag forward direction, read back the left classifier to obtain $\mathsf{hsame}(\mathsf{append}(a,m),\mathsf{append}(a,n))$. Left append cancellation under the common unary prefix a recovers $\mathsf{hsame}(m,n)$, and the original unary assumptions supply the two payload witnesses. Repacking by same-tag readback gives $\mathsf{IntHistoryClassifier}(\mathsf{E0}(m),\mathsf{E0}(n))$.

The one-tag branch repeats the same two moves after replacing E0 by E1. The readback criterion is unchanged on payloads, so append congruence proves the reverse implication, and left append cancellation proves the forward implication before repacking the public classifier. □

Lean4 ✓ `BEDC.Derived.IntUp.IntHistoryClassifier positive append context iff`

Theorem 31.59 (Integer history classifier hsame transport)**.** *For all h,k,h',k' : BHist, the public integer history classifier is stable under replacement of both endpoints by* hsame*-equal representatives:*

$$\mathsf{hsame}(h,h') \to \mathsf{hsame}(k,k') \to \mathsf{IntHistoryClassifier}(h,k) \to \mathsf{IntHistoryClassifier}(h',k').$$

The ten-field fit is the following. The traditional target is signed-integer equality stable under representative replacement. The BEDC source consists of BHist*,* hsame*,* E0*,* E1*, and* IntHistoryClassifier*. The carrier is* IntHistoryCarrier*. The classifier is the two-branch public classifier. The stability endpoint is* hsame *transport. The ledger records the visible sign tag together with the unary payload. The core theorem is this transport statement. Exactness preservation is preservation of the classifier under representative replacement. The standard bridge is replacement of signed-unary representatives. The status is* ProofTarget*.*

Proof. Unfold the classifier witness for $\mathsf{IntHistoryClassifier}(h,k)$ and split its visible signed branch.

In the zero-tag branch, keep the unary payload witnesses m,n, the payload comparison $\mathsf{hsame}(m,n)$, and the stored mixed-tag exclusion. Compose the old endpoint witness $\mathsf{hsame}(h,\mathsf{E0}(m))$ with the symmetry of $\mathsf{hsame}(h,h')$ to obtain $\mathsf{hsame}(h',\mathsf{E0}(m))$. The same composition with $\mathsf{hsame}(k,k')$ gives $\mathsf{hsame}(k',\mathsf{E0}(n))$.

It remains to rebuild the branch exclusion at the transported endpoints. Given a hypothetical one-tag witness from h' and k', compose it back through $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$ to obtain the corresponding one-tag witness from h and k. The stored zero-branch exclusion then rules it out, so the zero-tag classifier branch repacks at h' and k'.

In the one-tag branch, keep the same unary payload data and payload comparison. The supplied endpoint transports compose with the old witnesses to produce $\mathsf{hsame}(h', \mathsf{E1}(m))$ and $\mathsf{hsame}(k', \mathsf{E1}(n))$.

For the transported exclusion, any zero-tag witness at h' and k' composes back to a zero-tag witness at h and k. The stored one-branch exclusion rules it out, and the one-tag classifier branch repacks with the original payload witnesses. □

Theorem 31.60 (IntUp mature signed-unary downstream package)**.** *The carrier* IntHistoryCarrier*, the classifier* IntHistoryClassifier*, the same-tag readback iff, the positive append-context iff,* hsame *transport, the canonical pair/history bridge, the zero-sign refined boundary, and the integer signed-unary standard bridge together form the mature signed-unary* $\mathsf{Int}^{\uparrow}$ *package consumed by* $\mathsf{Rat}^{\uparrow}$ *numerator readback and by later ordered-ring or divisibility consumers. This package imports no host* Nat*, host equality, quotient-normalization primitive, Euclidean division, or full ordered-ring API.*

Proof. The public carrier and classifier are supplied by Definition 31.48. Their same-tag semantic readback is Definition 31.49, and the common positive context law is Definition 31.58.

Representative stability is exactly Definition 31.59. The external pair/history interface is supplied by Definition 31.50, while Definition 31.54 isolates the refined zero-sign boundary used by consumers that inspect signs.

Finally, Definition 31.57 binds these classifier and transport facts to the signed-unary standard bridge. The listed ingredients therefore provide the public $\mathsf{Int}^{\uparrow}$ package used by $\mathsf{Rat}^{\uparrow}$ numerator readback and by later algebraic consumers. None of the cited facts introduces a host Nat equality principle, quotient normalization, Euclidean division, divisibility theory, or a full ordered-ring API. □

Theorem 31.61 (IntUp signed-history RatUp numerator scope)**.** *Every* $\mathsf{Rat}^{\uparrow}$ *numerator readback that consumes an* $\mathsf{Int}^{\uparrow}$ *signed-history row factors through the rational naming certificate of chapter 37. The consumed integer row is still the signed-unary package generated from the* $\mathsf{Nat}^{\uparrow}$ *magnitude surface of chapter 28 and the continuation-compatible additive routing of chapter 29; the rational consumer merely places that displayed signed row beside its positive denominator and ledger rows.*

Consequently the downstream rational read exports no integer quotient primitive, no field equality, and no rational equality row at the $\mathsf{Int}^{\uparrow}$ *boundary. The only exported integer data are the visible sign mark, unary magnitude, zero-sign normalization, append-context transport, and signed-history classifier rows already supplied by the* $\mathsf{Int}^{\uparrow}$ *certificate.*

Proof. Read the signed-history certificate through Definition 31.60. That package exposes the semantic certificate, signed-history classifier rows, append-context and hsame transports, zero-sign boundary, and standard signed-unary bridge.

The rational consumer is then the carrier of chapter 37. It accepts the displayed signed integer row as numerator data and pairs it with the positive denominator and rational ledger rows, so the readback factors through the rational sibling chapter rather than through an ambient quotient construction.

The source rows for the integer side remain the Nat unary magnitude row and the Add continuation route. Since no extra coordinate is introduced at the handoff, the boundary exports neither an integer quotient primitive nor a field-level equality witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{Int}^\uparrow}$**.**

Constructive story: An integer is presented as one sign mark sitting beside one unary magnitude: the mark is b0 or b1, the magnitude is a $\mathsf{Nat}^\uparrow$ spine, and nothing enters the carrier unless both pieces are visible. The classifier first compares signs by mark-equality and magnitudes by history-equality, then folds exactly one ambiguity back together by identifying $(b0, \mathsf{e})$ with $(b1, \mathsf{e})$ so that the two written zeroes name the same integer. With that rule in place, the source is the same signed-unary predicate, the pattern records the sign/magnitude case split, magnitude behavior under append contexts, and the zero-sign collapse, and the ledger transports the visible pair along history-sameness on the magnitude side. The five name-certificate obligations are carried by $(b0, \mathsf{e})$ as the inhabitant, by mark- and history-equality as equivalence relations, and by the zero rule being stable under symmetry and transitivity. A signed pair (s, m) lies in the signature gap of an $\mathsf{Int}^\uparrow$ token p when the source-domain accepts the pair, the signature relation supplies an answer pair with its evidence ledger, and the token-introduction relation names that answer as p; after unfolding, the pair has the sign and magnitude encoded by p, except that either sign may represent zero. The standard bridge reads the same data as a signed unary word and back, which is why $\mathsf{Rat}^\uparrow$ can later consume it for numerator readback and algebraic packages can reuse it without changing the carrier. The chapter registers the construction under the name $\mathsf{Int}^\uparrow$; an integer is a sign mark next to a unary spine, not a quotient of natural-number pairs, and the only imposed equivalence is the explicit zero-sign normalization.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Int}^\uparrow}$, maturePackageClosed)

Scope closed: The mature public surface is the signed-unary downstream package of Definition 31.60: the semantic certificate, classifier readback rows, append-context and hsame transports, pair/history and zero-sign bridges, and the standard signed-unary bridge consumed by $\mathsf{Rat}^\uparrow$ numerator readback and later algebraic consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Int}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.IntUp.IntHistory\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: No full traditional integer algebra, ordered-ring API, Euclidean division, divisibility theory, or quotient-normalization primitive is closed here.

Upgrade path: No further upgrade – chapter at top of formal axis.

Lemma 31.62 (IntUp signed rows feed the RatUp denominator certificate)**.** *The* $\mathsf{Rat}^\uparrow$ *carrier of chapter 37 consumes the* $\mathsf{Int}^\uparrow$ *signed-pair normalization before any field-style rational row is licensed. Concretely, a rational representative presents signed numerator data through the* $\mathsf{Int}^\uparrow$ *carrier and then pairs that row with the positive unary denominator witness; the denominator certificate is read in the same finite packet as the signed numerator, not as an independent field quotient.*

Proof. The signed-pair interface supplies the visible sign and unary-magnitude rows for the numerator, together with the local normalization that identifies the two written zeroes. The $\mathsf{Rat}^\uparrow$ carrier then reads those signed rows beside a positive unary denominator row.

Because the rational row is admitted only after both pieces are displayed, the later field-style rational operations inherit a certified signed representative and a certified positive denominator. No unlisted quotient of integer pairs or ambient field equality is introduced by this dependency. □

Theorem 31.63 (Int signed numerator route into Rat)**.** *Every positive-denominator* $\mathsf{Rat}^\uparrow$ *packet that consumes the* $\mathsf{Int}^\uparrow$ *surface reads signed numerator data only through the displayed signed numerator row, the* $\mathsf{Nat}^\uparrow$ *magnitude row, the* $\mathsf{Add}^\uparrow$ *balancing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The route may then pair that numerator with the positive denominator surface of chapter 37, but it exports no host integer equality, quotient class of integer pairs, primitive subtraction row, or ambient field equality.*

Proof. Start with the integer surface. The $\mathsf{Int}^\uparrow$ carrier presents the sign and magnitude of the numerator as finite signed-pair data, and the magnitude coordinate is read through the unary $\mathsf{Nat}^\uparrow$ rows of chapter 28.

The balancing evidence is routed through chapter 29. Thus a rational consumer sees the numerator only after the signed row and additive balancing row have both appeared in the same finite history packet.

The structural rows preserve that route componentwise. The hsame row transports the signed numerator, magnitude, and balancing coordinates; the Cont row replays the same displayed route; Pkg records the provenance; and the local NameCert row closes the accepted integer surface.

Finally the $\mathsf{Rat}^{\uparrow}$ carrier of chapter 37 pairs that accepted numerator with a positive denominator row. Since neither the integer surface nor the rational pairing row contains host integer equality, quotient classes, primitive subtraction, or ambient field equality, no such data are exported by the dependency. □

Theorem 31.64 (IntUp Rat additive denominator route)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair balancing route consumes the unary magnitude rows of chapter 28 and the continuation-compatible addition rows of chapter 29 before* $\mathsf{Rat}^{\uparrow}$ *reads the signed numerator beside the positive denominator of chapter 37. Thus the rational denominator packet receives an already balanced signed numerator row rather than a fresh integer quotient or a field-level equality witness.*

Proof. Read the $\mathsf{Int}^{\uparrow}$ signed-pair carrier. Its magnitude coordinates are unary rows, so their source is the Nat certificate, and its balancing step is an additive comparison, so its route is through the Add continuation ledger.

Only after those rows are displayed does the rational carrier read numerator data. The $\mathsf{Rat}^{\uparrow}$ packet places the balanced signed numerator beside the positive denominator row in the same finite certificate.

Therefore the denominator route is horizontal through Nat, Add, Int, and Rat packet rows. It does not rebuild integer equality as an ambient quotient and does not import a field-level rational equality before the signed numerator and denominator witnesses are present. □

31.4 Rat Denominator Sibling Bridge

The $\mathsf{Int}^{\uparrow}$ signed-pair route is a substantive sibling dependency for chapter 37. The $\mathsf{Rat}^{\uparrow}$ denominator surface consumes the balanced integer numerator only after the signed-pair route and the additive-denominator route have been displayed, so the rational carrier is not a standalone quotient surface or a field-equality row.

31.5 Rat Denominator Sibling Link

The $\mathsf{Int}^{\uparrow}$ signed-pair rows are the horizontal source for the $\mathsf{Rat}^{\uparrow}$ denominator surface in chapter 37. The signed numerator first passes through Definition 31.62 and the additive denominator route of Definition 31.64; only then does the rational carrier read that balanced integer row beside its positive denominator witness. Thus the denominator and additive-denominator certificates consume an already displayed $\mathsf{Int}^{\uparrow}$ signed-pair route rather than a standalone rational quotient or field-equality row. The precise dependency is that the $\mathsf{Rat}^{\uparrow}$ positive-denominator ledger consumes the $\mathsf{Int}^{\uparrow}$ signed-pair normalization route before it reads the rational denominator carrier in chapter 37.

Theorem 31.65 (Int Rat sibling lattice readback)**.** *Every* $\mathsf{Rat}^{\uparrow}$ *signed-denominator route that consumes* $\mathsf{Int}^{\uparrow}$ *signed-pair data reads the* $\mathsf{Int}^{\uparrow}$ *carrier through the* $\mathsf{Nat}^{\uparrow}$ *rows of chapter 28 and the* $\mathsf{Add}^{\uparrow}$ *rows of chapter 29 before denominator normalization in chapter 37. The rational packet is a downstream consumer of the displayed signed-pair row; it is not a replacement carrier for* $\mathsf{Int}^{\uparrow}$ *and it exports no host integer quotient, primitive subtraction row, or field-level rational equality at the integer boundary.*

Proof. Start with the signed-pair carrier of $\mathsf{Int}^{\uparrow}$. Its sign row is read beside a unary magnitude row, so the magnitude component is already governed by the $\mathsf{Nat}^{\uparrow}$ surface before any rational denominator packet can consume it.

Next read the additive balancing route. The $\mathsf{Add}^{\uparrow}$ rows supply the comparison and balance data used by the signed-pair presentation, so the rational route receives an already balanced integer numerator rather than a new integer construction.

Apply Definition 31.70 and Definition 31.72. They identify the sibling lattice edge from $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}$ through $\mathsf{Int}^{\uparrow}$ into the $\mathsf{Rat}^{\uparrow}$ denominator surface.

Finally use the denominator-side descriptions of Definition 31.64 and Definition 31.66. The $\mathsf{Rat}^{\uparrow}$ packet reads the displayed signed-pair row beside its positive denominator witness only after those Nat and Add rows are in force, so the route is a downstream readback of $\mathsf{Int}^{\uparrow}$, not a replacement for it. □

Theorem 31.66 (IntUp Rat denominator dependency surface)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair denominator route consumed by* $\mathsf{Rat}^{\uparrow}$ *in chapter 37 factors through the signed-pair carrier before the rational denominator packet is read. A rational consumer first receives the displayed sign and unary-magnitude rows of* $\mathsf{Int}^{\uparrow}$*, then the balancing route that normalizes the signed numerator, and only then pairs that signed row with the positive denominator witness used by* $\mathsf{Rat}^{\uparrow}$*.*

Thus the denominator dependency is a horizontal sibling route from the integer signed-pair surface to the rational carrier, not a hidden rational quotient, host field equality, or denominator row independent of the displayed integer packet.

Proof. Start with the signed-pair interface. The integer packet exposes a sign mark and unary magnitude row, together with the local zero-sign normalization and balancing context.

Apply Definition 31.62 and Definition 31.64. These rows show that the rational carrier reads already displayed integer numerator data before it pairs that data with its positive denominator witness.

The sibling link of section 31.5 identifies chapter 37 as the rational consumer of that route. Since no step introduces an independent quotient or field-equality coordinate, the dependency surface is exactly the signed-pair carrier followed by the RatUp denominator packet. □

Theorem 31.67 (IntUp Rat denominator ledger dependency)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair carrier supplies the numerator and denominator pair surface consumed by the positive-denominator* $\mathsf{Rat}^{\uparrow}$ *ledger of chapter 37. A rational denominator consumer reads the signed-pair interface of Definition 31.9 before it can pair the numerator surface with a positive unary denominator row; the route exports no host integer quotient, primitive subtraction row, or field-level rational equality.*

Proof. Project through the signed-pair rows of the $\mathsf{Int}^{\uparrow}$ carrier. The positive and negative unary components expose the numerator surface that the rational packet later consumes beside its positive denominator witness.

Transport is componentwise. The hsame rows preserve the signed-pair components, while the displayed Cont replay keeps the numerator/denominator read inside the finite ledger rather than in a host quotient.

Finally, local $\mathsf{NameCert}$ provenance records that the $\mathsf{Rat}^{\uparrow}$ denominator route reads an already displayed $\mathsf{Int}^{\uparrow}$ signed-pair surface. Hence the dependency is a visible ledger edge from $\mathsf{Int}^{\uparrow}$ to $\mathsf{Rat}^{\uparrow}$, not a detached rational carrier. □

Theorem 31.68 (IntUp signed-pair additive lattice dependency)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair presentation is an additive child of the arithmetic lattice. Its balancing rows consume the unary magnitude*

rows of chapter 28 and the continuation-compatible addition rows of chapter 29; the later $\mathsf{Rat}^{\uparrow}$ *denominator presentation of chapter 37 consumes the resulting signed numerator row rather than rebuilding an independent integer quotient.*

Proof. Read the signed-pair carrier from the integer interface. Its magnitude coordinate is unary, so it is supplied by the Nat row, and its balancing step is additive, so it is routed through the Add continuation ledger.

The rational side then reads the same signed-pair surface as numerator data before pairing it with the denominator row. Thus $\mathsf{Rat}^{\uparrow}$ receives an already balanced $\mathsf{Int}^{\uparrow}$ row, with no separate quotient of integer pairs or field-style rational equality introduced at the integer boundary. □

Lemma 31.69 (Int signed-pair Rat dependency lattice)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair rows consumed by* $\mathsf{Rat}^{\uparrow}$ *are already arithmetic rows over* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Add}^{\uparrow}$*. The sign coordinate is paired with a unary magnitude row from chapter 28, the zero-balancing and signed continuation discipline are routed through chapter 29, and the resulting signed numerator surface is the row later read by the denominator and numerator packets of chapter 37.*

Thus an accepted rational numerator read does not rebuild an integer quotient. It consumes the displayed $\mathsf{Int}^{\uparrow}$ signed pair, whose BHist magnitude, additive balancing, hsame transport, Cont replay, Pkg provenance, and local NameCert row have already been exposed. The sibling edge is therefore Nat source row plus Add balancing row into Int signed pair, then Int signed pair into Rat numerator and denominator readback.

Theorem 31.70 (IntUp Nat Add Rat sibling route)**.** *Every signed-pair* $\mathsf{Int}^{\uparrow}$ *carrier route consumed by a later rational embedding first reads the* $\mathsf{Nat}^{\uparrow}$ *unary rows of chapter 28, the* $\mathsf{Add}^{\uparrow}$ *balance rows of chapter 29, and the* $\mathsf{Rat}^{\uparrow}$ *denominator sibling rows of chapter 37. The rational consumer receives the displayed signed-pair carrier, additive balancing route, and denominator packet; it receives no host integer quotient, primitive subtraction operation, or field-level rational equality from the integer boundary.*

Proof. Read the signed-pair carrier from the integer interface. Its magnitude coordinates are unary histories, so the route begins at the Nat certificate surface, and its balance comparison is routed through the Add continuation ledger.

Apply Definition 31.68 and Definition 31.66. They identify the integer row consumed by the rational side as the already displayed signed-pair packet together with its additive balance rows.

Finally use Definition 31.64. The $\mathsf{Rat}^{\uparrow}$ consumer reads that balanced signed row beside the positive denominator witness, so the sibling route is exactly Nat unary data, Add balance data, and Rat denominator data. No step exports a host integer quotient or primitive subtraction row. □

Theorem 31.71 (IntUp Nat Add Rat Dyadic lattice readback)**.** *Every* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed-mantissa consumer of the* $\mathsf{Int}^{\uparrow}$ *surface first factors through chapter 28, chapter 29, chapter 37, and then chapter 291. The consumer receives signed-unary branch data,* $\mathsf{Add}^{\uparrow}$ *balance rows,* $\mathsf{Rat}^{\uparrow}$ *denominator rows, and dyadic tolerance ledgers; it receives no host integer quotient, host rational quotient, selected real limit, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Apply Definition 31.70. It identifies the integer surface consumed by rational embedding as a displayed signed-pair carrier whose magnitude rows are read through $\mathsf{Nat}^{\uparrow}$, whose balance rows are read through $\mathsf{Add}^{\uparrow}$, and whose denominator-facing route is read through $\mathsf{Rat}^{\uparrow}$.

Next apply Definition 291.2. The $\mathsf{DyadicRatCore}^{\uparrow}$ carrier reads its signed mantissa through the consumed $\mathsf{Rat}^{\uparrow}$ row and its sign and integer provenance through the consumed $\mathsf{Int}^{\uparrow}$ row, while the exponent and denominator ledger stay visible as unary history data.

Finally use Definition 291.80. The L10 dyadic-rational boundary exports exactly the finite dyadic tolerance ledger read by the downstream faces. Since that dependency surface carries no host rational quotient, selected real limit, or $\mathsf{Real}^{\uparrow}$ completeness theorem, and the integer route already excludes host integer quotients, every signed-mantissa consumer factors through the displayed Nat, Add, Rat, and DyadicRatCore rows. □

Proposition 31.72 (Int signed-pair sibling dependency lattice)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair carrier consumes the unary row discipline of* $\mathsf{Nat}^{\uparrow}$ *from chapter 28 and the additive comparison rows of* $\mathsf{Add}^{\uparrow}$ *from chapter 29 before* $\mathsf{Rat}^{\uparrow}$ *denominator routes in chapter 37 can cite the integer surface. Thus the integer packet is a horizontal dependency node: it is not merely a target of rational-denominator readback, but a finite signed-pair ledger whose Nat and Add rows are already in force when Rat consumes it.*

Proposition 31.73 (Int Nat Add Rat dependency surface)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair carrier depends on the unary magnitude rows of* $\mathsf{Nat}^{\uparrow}$ *from chapter 28 and on the additive transport rows of* $\mathsf{Add}^{\uparrow}$ *from chapter 29 before the* $\mathsf{Rat}^{\uparrow}$ *denominator consumer of chapter 37 can use the integer surface. Thus the Rat-facing denominator route reads an already balanced signed-pair ledger rather than treating* $\mathsf{Int}^{\uparrow}$ *as a private rational subcarrier.*

Theorem 31.74 (Int Nat Add Rat sibling lattice route)**.** *Every* $\mathsf{Int}^{\uparrow}$ *signed-pair read consumed by the rational denominator surface is routed through the* $\mathsf{Nat}^{\uparrow}$ *unary rows of chapter 28, the* $\mathsf{Add}^{\uparrow}$ *balance rows of chapter 29, and the* $\mathsf{Rat}^{\uparrow}$ *signed-denominator consumer surface of chapter 37. The route exposes a displayed signed-pair ledger and denominator consumer; it does not introduce a quotient-class primitive, host integer equality, primitive subtraction row, or field-level rational equality at the integer boundary.*

Proof. Begin with the signed-pair interface. The magnitude rows are unary histories supplied by the Nat certificate, while the sign-aware balance row is routed through the Add continuation ledger.

Apply Definition 31.72. It places $\mathsf{Int}^{\uparrow}$ as the horizontal dependency node whose Nat and Add rows are already present before the Rat denominator route cites the integer surface.

Next use Definition 31.70 and Definition 31.64. The rational consumer receives the balanced signed row beside its positive denominator witness, so the downstream surface is the Rat signed-denominator consumer rather than a replacement carrier for Int.

Finally apply Definition 31.66. The dependency stops at the displayed signed-pair row and the denominator packet, leaving no coordinate for a quotient-class primitive or host equality. □

Theorem 31.75 (Int Add Nat sibling route)**.** *Every signed-pair* $\mathsf{Int}^{\uparrow}$ *packet consumes the* $\mathsf{Nat}^{\uparrow}$ *unary rows of chapter 28 and the* $\mathsf{Add}^{\uparrow}$ *balance rows of chapter 29 before the* $\mathsf{Rat}^{\uparrow}$ *denominator routes of chapter 37 can read it. Thus a rational denominator consumer receives an already displayed signed-pair row with Nat magnitude data and Add balance data, not a host integer quotient, primitive subtraction row, or field-level rational equality at the integer boundary.*

Proof. Start with the signed-pair surfaces of the Int chapter. The signed-pair interface and the signed-pair additive lattice dependency expose the sign row, unary magnitude row, and balance context before any Rat-facing denominator read is available.

Apply Definition 31.68 and Definition 31.74. These rows cite the Nat and Add sibling chapters and place the $\mathsf{Int}^{\uparrow}$ carrier between their unary and balance data and the later Rat denominator consumer.

Finally use Definition 31.64 and Definition 31.66. The Rat route reads the balanced signed row beside a positive denominator only after the Nat and Add rows have already been consumed, so the downstream read remains a sibling route through the displayed Int packet. □

Theorem 31.76 (Int sibling lattice pointer)**.** *Every Real-and-Completion consumer of the* $\mathsf{Int}^{\uparrow}$ *signed-pair surface, including the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa route of chapter 291 and the* $\mathsf{Rat}^{\uparrow}$ *denominator routes of chapter 37, reads* $\mathsf{Int}^{\uparrow}$ *only through the displayed* $\mathsf{Nat}^{\uparrow}$ *unary magnitude rows of chapter 28, the* $\mathsf{Add}^{\uparrow}$ *continuation-compatible balancing rows of chapter 29, the* $\mathsf{Rat}^{\uparrow}$ *signed-denominator consumer rows, and the local* $\mathsf{Int}^{\uparrow}$ $\mathsf{NameCert}$ *rows. This lattice admits no host integer equality, quotient representative, primitive subtraction, field-level rational equality at the* $\mathsf{Int}^{\uparrow}$ *boundary, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness operation.*

Proof. Start with the local signed-pair surface. The sibling lattice of Definition 31.72 places every $\mathsf{Int}^{\uparrow}$ magnitude coordinate under the unary rows of chapter 28 and every balancing comparison under the continuation-compatible additive rows of chapter 29. Thus the integer packet reaches its signed-pair form through displayed Nat and Add data, not through a host integer primitive.

Next read the rational consumer boundary. The signed-denominator dependency Definition 31.62 and the additive denominator route Definition 31.64 force a $\mathsf{Rat}^{\uparrow}$ consumer to take the balanced signed numerator beside the positive denominator row of chapter 37. Hence the Rat-facing route can consume the local $\mathsf{Int}^{\uparrow}$ $\mathsf{NameCert}$ rows only after the Nat and Add rows have already been displayed.

Finally pass to the dyadic and completion-facing consumers. The $\mathsf{DyadicRatCore}^{\uparrow}$ chapter chapter 291 names the $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ source certificates in its mantissa and denominator ledger, so a Real-and-Completion reader reaches the integer source only through that same finite route. None of the listed downstream consumers has a coordinate for host integer equality, quotient representatives, primitive subtraction, field-level rational equality at the $\mathsf{Int}^{\uparrow}$ boundary, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so those objects cannot be imported by this sibling lattice pointer. □

Theorem 31.77 (IntUp signed-pair sibling lattice repair)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair carrier is a sibling-lattice node between unary natural arithmetic and rational denominator arithmetic. It consumes the unary source rows of chapter 28 and the balancing rows of chapter 29; it supplies signed numerator, denominator-normalization, and sign-readback rows later read by chapter 37 and chapter 291. Thus the signed-pair construction is not a kernel-only hub: its public route is the finite dependency lattice*

$$\mathsf{Nat}^{\uparrow}, \mathsf{Add}^{\uparrow} \longrightarrow \mathsf{Int}^{\uparrow} \longrightarrow \mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}$$

through displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Read the left side of the route first. The signed-pair carrier uses unary magnitude rows licensed by chapter 28, and it uses balancing data from chapter 29 to identify opposite signed presentations without importing quotient equality.

Now read the right side. The rational packet of chapter 37 consumes the signed numerator and positive denominator discipline, while chapter 291 consumes the dyadic denominator and sign-readback rows used by finite dyadic arithmetic.

The local BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows preserve this route as public data. Therefore $\mathsf{Int}^{\uparrow}$ is a lattice handoff between Nat/Add input rows and Rat/DyadicRatCore consumer rows, not an isolated integer naming site. □

Theorem 31.78 (IntUp dyadic mantissa real route)**.** *Every Real-and-Completion read of an* $\mathsf{Int}^{\uparrow}$ *signed-pair coordinate used as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa first factors through chapter 28, chapter 29, chapter 37, and chapter 291 before any* $\mathsf{Real}^{\uparrow}$ *seal route is read. The route exports the signed-pair mantissa as a BEDC lattice row, not as host integer equality, quotient representatives, primitive subtraction, or ambient real completeness.*

Proof. Start with the signed-pair additive lattice dependency of Definition 31.68. It makes the integer coordinate a Nat-and-Add balanced signed pair and gives the rational side a signed numerator row rather than an independent integer quotient.

Next route that numerator through the existing rational and dyadic mantissa surfaces. The rational row supplies the denominator-facing arithmetic context, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row is the first place where the signed coordinate is read as a dyadic mantissa for the Real-and-Completion chain.

Only after those four chapter surfaces have been crossed may a $\mathsf{Real}^{\uparrow}$ seal consumer read the route. Since none of the preceding rows exports host integer equality, quotient representatives, primitive subtraction, or ambient completeness, the real-side read inherits only the displayed BEDC lattice surface. □

Theorem 31.79 (IntUp dyadic real mantissa route)**.** *Every L10 consumer that reads an* $\mathsf{Int}^{\uparrow}$ *signed-pair coordinate as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa for* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *must pass through* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$ *denominator normalization,* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed-exponent source determinacy, and the finite BEDC source uniqueness rows. The handoff exports a finite signed dyadic source only; it exports no host integer equality, quotient rational representative, primitive subtraction, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 31.78. It forces every Real-and-Completion mantissa read through the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ chapter surfaces before a downstream seal can inspect it.

Apply Definition 31.80. This places the signed-pair zero boundary before rational denominator normalization and before the dyadic mantissa route, so the written sign convention is already part of the finite source row.

Now use Definition 31.81 together with Definition 31.82, Definition 31.83, and Definition 291.22. These rows bind the signed-pair source, rational denominator normalization, and dyadic signed-exponent determinacy before $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ can consume the coordinate.

The resulting handoff is therefore a finite BEDC source route. Since the route contains no coordinate for host integer equality, quotient rational representatives, primitive subtraction, selected limits, or real completeness, none of those objects is exported to the L10 consumer. □

Theorem 31.80 (IntUp dyadic mantissa zero-sign normalization)**.** *Every Real-and-Completion read that uses an* $\mathsf{Int}^{\uparrow}$ *signed-pair coordinate as the mantissa input of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet must first apply the* $\mathsf{Int}^{\uparrow}$ *zero-sign boundary. Thus the two written zero rows determine the same finite signed-mantissa source before* $\mathsf{Rat}^{\uparrow}$ *denominator normalization and* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa routing are read. The normalization is a finite source-row condition only; it does not assert host integer equality, quotient rational representatives, primitive subtraction, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the refined zero-sign boundary Definition 31.54. It identifies the two signed-pair zero presentations as the same $\mathsf{Int}^{\uparrow}$ boundary row before the coordinate is exported to any rational or dyadic consumer.

Next apply the route theorem Definition 31.78. It places every such mantissa read behind the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ surfaces, so the zero-sign boundary is already part of the finite row that enters the dyadic packet.

Finally use source uniqueness Definition 31.81 together with the sibling denominator route Definition 31.82. After the zero rows have collapsed to the same signed-mantissa source, the denominator normalization and dyadic routing cannot split them again. Downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers therefore see only the normalized finite row, with none of the excluded host-level data added. □

Theorem 31.81 (IntUp dyadic mantissa L10 source uniqueness)**.** *Let two L10 Real-and-Completion reads expose an* $\mathsf{Int}^{\uparrow}$ *signed-pair coordinate as the mantissa input of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet. Suppose both reads display the same* $\mathsf{Int}^{\uparrow}$ *signed-pair row, the same* $\mathsf{Rat}^{\uparrow}$ *denominator normalization surface, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa/denominator packet before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers read it. Then the two reads determine the same finite BEDC source row for that signed mantissa handoff. This uniqueness is a statement about the displayed BEDC packet only; it does not assert host integer equality, quotient rational representatives, primitive subtraction, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the route theorem Definition 31.78. It forces each read through the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ chapter surfaces before any Real-and-Completion consumer can inspect a seal boundary.

Next use the sibling route Definition 31.82 and the L10 handoff Definition 31.83. These identify the source of the dyadic packet as the displayed $\mathsf{Int}^{\uparrow}$ signed-pair row together with the displayed $\mathsf{Rat}^{\uparrow}$ denominator normalization surface and the dyadic mantissa route.

Finally apply the scoped source boundary Definition 291.22 and signed-exponent source determinacy Definition 291.59. Those rows say that downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers can read only the visible finite dyadic packet. Since the two reads share that packet before the consumers act, they determine the same finite BEDC source row and no excluded host-level datum can split it into two sources. □

Theorem 31.82 (IntUp DyadicRatCore denominator route sibling repair)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair and denominator rows form the sibling input consumed by chapter 291 on the Real-and-Completion route. A* $\mathsf{DyadicRatCore}^{\uparrow}$ *consumer first reads the* $\mathsf{Int}^{\uparrow}$ *signed-pair carrier through the* $\mathsf{Rat}^{\uparrow}$ *denominator normalization surface of chapter 37, then uses the dyadic mantissa support of Definition 31.78 before any* $\mathsf{Real}^{\uparrow}$ *seal boundary is available.*

Proof. Start with the signed-pair denominator route already displayed for $\mathsf{Rat}^{\uparrow}$. The integer carrier supplies a sign row and unary magnitude row, and the rational denominator surface reads that balanced signed row beside its positive denominator witness.

Next use the dyadic mantissa route. It sends the same signed coordinate through the rational and dyadic core surfaces before the Real-and-Completion chain may read a seal boundary.

Thus the $\mathsf{DyadicRatCore}^{\uparrow}$ chapter consumes a sibling input from $\mathsf{Int}^{\uparrow}$ rather than a detached host integer or field equality. The available route is exactly the signed-pair row, the denominator normalization row, and the dyadic mantissa support. □

Theorem 31.83 (IntUp L10 DyadicRatCore denominator handoff)**.** *Every L10 Real-and-Completion read of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa/denominator packet factors through the displayed* $\mathsf{Int}^{\uparrow}$ *signed-pair row, the* $\mathsf{Rat}^{\uparrow}$ *denominator normalization surface, and the dyadic mantissa route before* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *may consume the packet. The handoff is finite-row data only: it exposes the*

signed-pair row, denominator normalization row, and mantissa route named by Definition 31.82, not host integers, quotient rationals, selected limits, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the sibling route of Definition 31.82. Its source is the displayed $\mathsf{Int}^{\uparrow}$ signed-pair row, and its first consumer is the $\mathsf{Rat}^{\uparrow}$ denominator normalization surface of chapter 37.

Pass next through the dyadic face of chapter 291. The mantissa coordinate is still read as finite signed-pair data, and the denominator coordinate is still the normalized rational denominator row. This is the only input the L10 Real-and-Completion route supplies to the $\mathsf{DyadicRatCore}^{\uparrow}$ packet.

Only after that packet is visible may the downstream consumers read it: $\mathsf{StreamName}^{\uparrow}$ through chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ through chapter 301, and $\mathsf{Real}^{\uparrow}$ through chapter 38. None of these reads adds a host integer, quotient rational, selected limit, or completeness witness; they consume the same finite handoff row. □

Theorem 31.84 (IntUp FieldExt and Rat denominator sibling route)**.** *Every* $\mathsf{FieldExt}^{\uparrow}$ *base-endpoint read that consumes* $\mathsf{Rat}^{\uparrow}$ *denominator data ultimately depends on the* $\mathsf{Int}^{\uparrow}$ *signed-pair denominator route. The consumer in chapter 161 first reads the rational denominator surface of chapter 37; that surface has already consumed the displayed* $\mathsf{Int}^{\uparrow}$ *sign row, unary magnitude row, balancing route, and positive denominator witness before any field-extension endpoint can be named.*

Proof. Start with a $\mathsf{FieldExt}^{\uparrow}$ base-endpoint read whose rational endpoint data includes a denominator row. The field-extension carrier can read that denominator only through the displayed rational surface, so the first dependency is the $\mathsf{Rat}^{\uparrow}$ NameCert chapter.

The rational surface is not independent of $\mathsf{Int}^{\uparrow}$. Its numerator and denominator normalization rows use the signed-pair route of the integer packet before the positive denominator witness is paired with rational data.

Thus the field-extension endpoint consumes the same finite dependency route: $\mathsf{Int}^{\uparrow}$ signed-pair data, $\mathsf{Rat}^{\uparrow}$ denominator normalization, and then the $\mathsf{FieldExt}^{\uparrow}$ endpoint row. No detached host integer, quotient rational, or ambient field equality is introduced between these sibling chapters. □

Proposition 31.85 (IntUp Nat Add Rat Dyadic dependency spine)**.** *The* $\mathsf{Int}^{\uparrow}$ *signed-pair rows consume the unary arithmetic discipline of chapter 28 and the finite additive comparison rows of chapter 29. Those signed rows then feed the denominator surfaces of chapter 37, and the same denominator and mantissa coordinates are read by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *routes of chapter 291. Thus* $\mathsf{Int}^{\uparrow}$ *is the finite arithmetic spine from NatUp and AddUp into RatUp and DyadicRatCore, not an isolated hub or a host-integer equality principle.*

Theorem 31.86 (IntUp Nat Add Rat Dyadic sibling exhaustion)**.** *Every* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *consumer of* $\mathsf{Int}^{\uparrow}$ *signed numerator data first reads the* $\mathsf{Nat}^{\uparrow}$ *magnitude row and the* $\mathsf{Add}^{\uparrow}$ *balance row. Only after those rows are displayed may the consumer read the* $\mathsf{Rat}^{\uparrow}$ *denominator row or the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mantissa and denominator rows. The route exports finite signed-unary dependency data, not a host integer quotient, primitive subtraction row, host rational quotient, selected real limit, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with the rational sibling route. By Definition 31.70, a rational consumer receives the signed-pair carrier only after the $\mathsf{Nat}^{\uparrow}$ unary magnitude rows and the $\mathsf{Add}^{\uparrow}$ balance rows have been read.

The dyadic dependency spine preserves the same order. By Definition 31.85, the signed rows feed the $\mathsf{Rat}^{\uparrow}$ denominator surface and then the $\mathsf{DyadicRatCore}^{\uparrow}$ denominator and mantissa routes; the dyadic rows do not provide an independent source for signed integer data.

Finally apply Definition 31.82. The dyadic consumer reads the $\mathsf{Int}^{\uparrow}$ signed-pair carrier through the rational denominator normalization surface before using the dyadic mantissa support. Hence every Rat or DyadicRatCore consumer factors through Nat magnitude and Add balance rows before denominator or mantissa rows are available. □

Theorem 31.87 (Int signed-pair lattice sibling route)**.** *Every signed-pair read exported by the* $\mathsf{Int}^{\uparrow}$ *hub to downstream* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *consumers factors through the unary magnitude rows of chapter 28 and the balancing row of chapter 29 before any denominator or mantissa data are exposed. Consequently a rational numerator read from chapter 37 and a dyadic mantissa read from chapter 291 receive the same displayed route: first the two* $\mathsf{Nat}^{\uparrow}$ *magnitude rows, then the* $\mathsf{Add}^{\uparrow}$ *balance witness identifying their signed-pair difference, and only then the consumer-specific denominator, scale, or mantissa ledger.*

Proof. Use the $\mathsf{Int}^{\uparrow}$ hub orientation as the boundary condition. It names $\mathsf{Nat}^{\uparrow}$ as the source of unary magnitude data and $\mathsf{Add}^{\uparrow}$ as the balancing surface that makes a signed pair public as integer data.

A $\mathsf{Rat}^{\uparrow}$ consumer cannot inspect a numerator before the signed-pair interface has displayed both magnitudes and the balancing row. Thus its denominator route is downstream of the $\mathsf{Nat}^{\uparrow}, \mathsf{Add}^{\uparrow}$ lattice rather than an alternate source of integer evidence.

The same argument applies to a $\mathsf{DyadicRatCore}^{\uparrow}$ mantissa consumer. The dyadic ledger can read signed mantissa data only after the integer route has opened the two unary magnitudes and the displayed balance witness.

Therefore both downstream consumers share the same finite sibling route through $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}$. Denominator and mantissa coordinates are exposed only after that route has been fixed. □

The BoolUp two-endpoint mark surface is the finite discriminator consumed by chapter 33, chapter 35, and chapter 34; those siblings own payload packaging while BoolUp owns only the visible branch marks.

Chapter 32

A Concrete Naming Certificate for $\mathsf{Bool}^{\uparrow}$

The boolean interface $\mathsf{Bool}^{\uparrow}$ is licensed by raw mark sameness on the two-element basis: b0 read as false, b1 read as true. It is not a primitive proposition or truth-value type; it is licensed only via the underlying Mark structure together with a classifier specification that distinguishes the two marks under msame.

The nullable and tagged split of chapter 33 consumes this two-endpoint surface: the empty endpoint and the one-step endpoint supply the two visible branch markers that $\mathsf{Option}^{\uparrow}$ later separates from payload evidence.

Conversely, chapter 33 uses those two endpoint rows as the sibling lattice discriminator for nullable payloads, while $\mathsf{Bool}^{\uparrow}$ remains only the two-endpoint mark/history certificate and does not own the payload carrier, payload classifier, or option packaging rows.

chapter 35 and chapter 34 consume the same two-endpoint $\mathsf{Bool}^{\uparrow}$ discriminator as finite branch and product control, while $\mathsf{Sum}^{\uparrow}$ and $\mathsf{Prod}^{\uparrow}$ own their payload carriers, pairing rows, and classifiers.

32.1 Roadmap template status

1. **Traditional target**: the two-point boolean object with false and true.
2. **BEDC source**: the finite mark kernel Mark, with b0 and b1 as the displayed points.
3. **Carrier choice**: the mark-level carrier is Mark itself; the history carrier is BoolHistoryCarrier.
4. **Classifier choice**: the classifier is msame on marks and BoolHistoryClassifier on the two licensed histories.
5. **Stability obligations**: mark transport, history-carrier transport, unary closure, endpoint exclusion, classifier symmetry, and classifier transitivity.
6. **Ledger**: no quotient is used; the hidden data are only the generated history witnesses for the two boolean endpoints.
7. **Core theorem list**: source transport, history-carrier transport, history classifier equivalence, constructor exclusions, Theorem 32.4, Theorem 32.5, endpoint readback determinism, Theorem 32.9, Theorem 32.10, Theorem 32.11, and the history core NameCert.
8. **Exactness predicate**: the mark source has exactly the b0 and b1 branches licensed by msame, while the checked history certificate has exactly the e and E1(e) endpoints licensed by hsame.

9. **Standard bridge**: false and true are read through the two displayed marks $\mathsf{b0}, \mathsf{b1}$ together with their displayed history endpoints $\mathsf{e}, \mathsf{E1}(\mathsf{e})$; endpoint coverage and classifier completeness are functional up to msame.

10. **Status**: Theorem 32.42 is the canonical Phase Finite Data public-object scope

    $$\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier})$$

    for $\mathsf{Bool}^{\uparrow}$, with checked public-object status at the $BEDC.Derived.BoolUp$ Lean target bool_history_semantic_ Theorem 32.41 is the generic implication from an arbitrary Bool core certificate, Theorem 32.42 is the closed checked Bool public-object certificate, the generic carrier-self semantic certificate is the paper theorem in Theorem 26.9, and the Lean theorem $BEDC.FKernel.NameCert.NameCert_iff_sema$ records the five core $\mathsf{NameCert}$ fields.

32.2 Carrier definition

Definition 32.1 (Bool carrier)**.** *Use the existing two-constructor mark carrier* Mark *with* $\mathsf{b0}$ *interpreted as false and* $\mathsf{b1}$ *interpreted as true.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolCarrier`

32.3 The certificate

Definition 32.2 (Bool source specification)**.** *The source specification for* $\mathsf{Bool}^{\uparrow}$ *is the mark-level classifier generated by internal sameness to one of the two finite-kernel marks:*

$$\mathsf{BoolSourceSpec}(v) := \mathsf{msame}(v, \mathsf{b0}) \vee \mathsf{msame}(v, \mathsf{b1}).$$

Lean4 (definition) `BEDC.Derived.BoolUp.BoolSourceSpec`
Lean4 (definition) `BEDC.Derived.BoolUp.BoolHistoryCarrier`

Definition 32.3 (Bool endpoint bridge)**.** *The endpoint map* $\mathsf{BoolEndpoint} : \mathsf{BMark} \to \mathsf{BHist}$ *sends the two traditional boolean points from the BEDC mark source into the displayed history endpoints:*

$$\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \qquad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}).$$

The carrier endpoint choice is therefore $\mathsf{BoolHistoryCarrier}$*, and the classifier transport target is* hsame *on the two displayed history endpoints.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolEndpoint`

Theorem 32.4 (Bool endpoint bridge exactness)**.** *For marks* $v, w : \mathsf{BMark}$*, mark sameness holds exactly when the displayed history endpoints are internally the same:*

$$\mathsf{msame}(v, w) \iff \mathsf{hsame}(\mathsf{BoolEndpoint}(v), \mathsf{BoolEndpoint}(w)).$$

Proof. Proceed by cases on v and w. The matching $\mathsf{b0}/\mathsf{b0}$ and $\mathsf{b1}/\mathsf{b1}$ cases reduce both directions to reflexivity. In the mixed cases, the mark-to-history direction has no msame witness by mark no-confusion, and the history-to-mark direction has no hsame witness by the no-confusion between e and $\mathsf{E1}(\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolEndpoint bridge exactness`

Theorem 32.5 (Bool history carrier endpoint coverage)**.** *For every history* $h : \mathsf{BHist}$*, the checked boolean history carrier is exactly the image of the endpoint bridge:*

$$\mathsf{BoolHistoryCarrier}(h) \iff \exists v : \mathsf{BMark},\ \mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)).$$

Proof. In the forward direction, split the carrier witness into its empty and one-endpoint branches. In the empty branch choose $v = \mathsf{b0}$ and use the defining endpoint equality $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$; in the one-endpoint branch choose $v = \mathsf{b1}$ and use $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$.

In the reverse direction, case-analyze the displayed mark v. The $\mathsf{b0}$ case rewrites the endpoint witness to an hsame witness against e and gives the empty carrier branch. The $\mathsf{b1}$ case rewrites it to an hsame witness against $\mathsf{E1}(\mathsf{e})$ and gives the one-endpoint carrier branch. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier endpoint coverage`

Theorem 32.6 (Bool endpoint readback determinism)**.** *For marks* $v, w : \mathsf{BMark}$ *and history* $h : \mathsf{BHist}$*, if* $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ *and* $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(w))$*, then* $\mathsf{msame}(v, w)$*.*

Proof. Compose symmetry of $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ with $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(w))$ to obtain $\mathsf{hsame}(\mathsf{BoolEndpoint}(v), \mathsf{BoolEndpoint}(w))$. The reverse direction of Theorem 32.4 converts this displayed endpoint sameness into $\mathsf{msame}(v, w)$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolEndpoint readback determinism`
↪ variant `BEDC.Derived.BoolUp.BoolEndpoint readback deterministic`

Theorem 32.7 (Bool endpoint readback total uniqueness)**.** *For every history* $h : \mathsf{BHist}$*, if* $\mathsf{BoolHistoryCarrier}(h)$ *holds, then there exists a mark* $v : \mathsf{BMark}$ *such that*

$$\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \quad \textit{and} \quad \mathsf{BoolSourceSpec}(v) \quad \textit{and} \quad \forall w : \mathsf{BMark},\ \mathsf{hsame}(h, \mathsf{BoolEndpoint}(w)) \to \mathsf{msame}(v$$

Proof. Use Theorem 32.5 on the carrier witness for h to obtain a displayed endpoint v and a readback witness $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$. Case analysis on v gives $\mathsf{BoolSourceSpec}(v)$ for both boolean marks. For any other displayed readback w, Theorem 32.6 applied to the two endpoint witnesses gives $\mathsf{msame}(v, w)$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolEndpoint readback total unique`

Theorem 32.8 (Bool classifier readback total uniqueness)**.** *For every history* $h : \mathsf{BHist}$*, if* $\mathsf{BoolHistoryCarrier}(h)$ *holds, then there exists a mark* $v : \mathsf{BMark}$ *such that*

$$\mathsf{BoolSourceSpec}(v) \quad \wedge \quad \mathsf{BoolHistoryClassifier}(h, \mathsf{BoolEndpoint}(v)) \quad \wedge \quad \forall w : \mathsf{BMark},\ \mathsf{BoolHistoryClassifier}(h, \mathsf{BoolEn}$$

The ten-field fit is: traditional target public boolean readback; BEDC source BHist *with* hsame*; carrier* $\mathsf{BoolHistoryCarrier}$*; classifier* $\mathsf{BoolHistoryClassifier}$*; stability obligations endpoint coverage and classifier determinacy; ledger displayed* $\mathsf{BoolEndpoint}(v)$*; core theorem this total uniqueness statement; exactness classifier-visible existence and uniqueness up to* msame*; standard bridge ordinary booleans through* $\mathsf{b0}, \mathsf{b1}$ *endpoints; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Theorem 32.9 to the carrier witness for h. This gives a mark v with $\mathsf{BoolSourceSpec}(v)$ and $\mathsf{BoolHistoryClassifier}(h, \mathsf{BoolEndpoint}(v))$. For uniqueness, unpack this classifier witness and any displayed classifier readback at $\mathsf{BoolEndpoint}(w)$ to expose endpoint witnesses $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ and $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(w))$. Theorem 32.6 then gives $\mathsf{msame}(v, w)$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier readback total unique`

Theorem 32.9 (Bool endpoint bridge classifier completeness)**.** *The endpoint bridge preserves the checked boolean history classifier in both finite certificate directions. For marks* $v, w : \mathsf{BMark}$*,*

$$\mathsf{BoolHistoryClassifier}(\mathsf{BoolEndpoint}(v), \mathsf{BoolEndpoint}(w)) \iff \mathsf{msame}(v, w).$$

Moreover, for every history $h : \mathsf{BHist}$ *satisfying* $\mathsf{BoolHistoryCarrier}(h)$*, there exists a mark* $v : \mathsf{BMark}$ *such that both* $\mathsf{BoolSourceSpec}(v)$ *and* $\mathsf{BoolHistoryClassifier}(h, \mathsf{BoolEndpoint}(v))$ *hold.*

Proof. For the first clause, unfold the classifier at the two displayed endpoints. The endpoint carrier fields are supplied by the two endpoint cases, so the classifier is equivalent to history sameness between the two endpoint images. The endpoint bridge exactness theorem then identifies this condition with $\mathsf{msame}(v, w)$.

For the second clause, apply carrier endpoint coverage to the carrier witness for h and obtain an endpoint v with $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$. Case analysis on v supplies $\mathsf{BoolSourceSpec}(v)$, since the two mark constructors satisfy the source specification by reflexivity. The same history-sameness witness, together with the carrier witness for h and the endpoint carrier witness, is exactly the data required by the definition of the displayed classifier relation. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier endpoint bridge exactness`
Lean4 ✓ `BEDC.Derived.BoolUp.BoolEndpoint classifier exactness`

Theorem 32.10 (Bool carrier classifier endpoint exactness)**.** *For every history* $h : \mathsf{BHist}$*,*

$$\mathsf{BoolHistoryCarrier}(h) \iff \exists v : \mathsf{BMark}.\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolHistoryClassifier}(h, \mathsf{BoolEndpoint}(v)).$$

Proof. The forward direction is the second clause of Theorem 32.9. For the reverse direction, unpack the displayed classifier witness. Its left endpoint carrier field is exactly $\mathsf{BoolHistoryCarrier}(h)$, so the carrier statement follows directly. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier classifier endpoint exactness`

Theorem 32.11 (Bool history classifier endpoint readback)**.** *For histories* $h, k : \mathsf{BHist}$*, if* $\mathsf{BoolHistoryClassifier}(h, k)$ *holds, then there exist marks* $v, w : \mathsf{BMark}$ *such that*

$$\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \quad \wedge \quad \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \quad \wedge \quad \mathsf{msame}(v, w).$$

Proof. Unpack the classifier witness into the two $\mathsf{BoolHistoryCarrier}$ endpoint witnesses and the stored $\mathsf{hsame}(h, k)$ link. Apply Theorem 32.5 to both carrier witnesses, obtaining marks v and w with endpoint readbacks from h and k. Compose the readback for h, the stored $\mathsf{hsame}(h, k)$ witness, and the readback for k to obtain $\mathsf{hsame}(\mathsf{BoolEndpoint}(v), \mathsf{BoolEndpoint}(w))$. The reverse direction of Theorem 32.4 converts this endpoint sameness into $\mathsf{msame}(v, w)$. □

Theorem 32.12 (Bool history classifier mark-readback exactness)**.** *For histories* $h, k : \mathsf{BHist}$*,* $\mathsf{BoolHistoryClassifier}(h, k)$ *holds if and only if there exist marks* $v, w : \mathsf{BMark}$ *satisfying*

$$\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \wedge \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \wedge \mathsf{msame}(v, w).$$

Proof. The forward direction is exactly Theorem 32.11. For the reverse direction, take displayed marks v, w with the two endpoint readbacks and $\mathsf{msame}(v, w)$. Endpoint carrier coverage gives $\mathsf{BoolHistoryCarrier}(\mathsf{BoolEndpoint}(v))$ and $\mathsf{BoolHistoryCarrier}(\mathsf{BoolEndpoint}(w))$. The forward direction of Theorem 32.4 reads $\mathsf{msame}(v, w)$ as $\mathsf{hsame}(\mathsf{BoolEndpoint}(v), \mathsf{BoolEndpoint}(w))$, hence the displayed endpoints satisfy $\mathsf{BoolHistoryClassifier}$. Finally apply endpoint transport for $\mathsf{BoolHistoryClassifier}$ along the two given hsame readbacks to obtain $\mathsf{BoolHistoryClassifier}(h, k)$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier mark readback exactness`

Theorem 32.13 (Bool source-qualified classifier readback exactness)**.** *For histories* h, k : BHist*,* BoolHistoryClassifier(h, k) *holds if and only if there exist marks* v, w : BMark *satisfying*

$$\mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \wedge \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \wedge \mathsf{msame}(v, w).$$

The traditional target is the two-point boolean readback; the BEDC source is BMark *and* BHist*, the carrier is* BoolHistoryCarrier*, the classifier is* BoolHistoryClassifier*, the exactness clause is source-qualified endpoint readback, and the status is* ProofTarget*.*

Proof. For the forward direction, apply Theorem 32.12 to obtain endpoint marks v, w with the two hsame readbacks and msame(v, w). Case analysis on the two displayed marks supplies BoolSourceSpec(v) and BoolSourceSpec(w), since the two constructors are exactly the allowed source marks. For the reverse direction, discard the two BoolSourceSpec witnesses and apply the reverse direction of Theorem 32.12. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier source qualified readback exactness`

Theorem 32.14 (Bool source respects mark sameness)**.** *If* v *satisfies* BoolSourceSpec *and* msame(v, w)*, then* w *also satisfies* BoolSourceSpec*.*

Proof. The source condition is a disjunction of msame$(v, \mathsf{b0})$ and msame$(v, \mathsf{b1})$. In either case, symmetry gives msame(w, v), and transitivity transports the corresponding constructor witness from v to w. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolSourceSpec respects msame`

Theorem 32.15 (Bool source transport along mark sameness)**.** *If* v *and* w *are internally the same mark, then the boolean source predicate transports from* v *to* w*.*

Lean4 ✓ `BEDC.Derived.BoolUp.BoolSourceSpec msame transport`

Proof. The proof eliminates the msame(v, w) witness, reducing w definitionally to v, and reuses the original source-classifier witness. □

Theorem 32.16 (Bool history carrier transport)**.** *If* h *is internally history-same as* k *and* h *lies in the boolean history carrier, then* k *also lies in the boolean history carrier.*

Proof. The carrier witness is either sameness to e or sameness to E1(e). In both cases, symmetry of the link from h to k followed by transitivity transports the endpoint equality to k. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier hsame transport`

Theorem 32.17 (Bool history carrier is unary)**.** *Every boolean history carrier witness determines a unary history.*

Proof. The carrier has two endpoint cases. In the empty endpoint case, unary closure of e transports back along history sameness. In the one-endpoint case, unary closure of E1(e) transports back along the corresponding history-sameness witness. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier unary`

Theorem 32.18 (Bool history carrier excludes zero-extension)**.** *No zero-extension history* $\mathsf{E0}(h)$ *lies in the boolean history carrier.*

Proof. The carrier witness is either sameness to e or sameness to $\mathsf{E1}(\mathsf{e})$. The first case contradicts generated-history separation between $\mathsf{E0}(h)$ and e, and the second contradicts constructor separation between zero- and one-extensions. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier e0 absurd`

Theorem 32.19 (Bool history one-extension tail)**.** *If* $\mathsf{E1}(h)$ *lies in the boolean history carrier, then* h *is internally the same as* e*.*

Proof. The carrier witness is either sameness to e or sameness to $\mathsf{E1}(\mathsf{e})$. The empty endpoint contradicts generated-history no-confusion for $\mathsf{E1}(h)$. The $\mathsf{E1}(\mathsf{e})$ endpoint reduces by one-constructor inversion to $\mathsf{hsame}(h, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier e1 tail empty`

Theorem 32.20 (Bool history double one-extension exclusion)**.** *No history of the form* $\mathsf{E1}(\mathsf{E1}(h))$ *lies in the boolean history carrier.*

Proof. A carrier witness for $\mathsf{E1}(\mathsf{E1}(h))$ forces its tail $\mathsf{E1}(h)$ to be internally the same as e. Generated-history separation between $\mathsf{E1}(h)$ and e eliminates that possibility. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier e1 e1 absurd`

Theorem 32.21 (Bool history one-extension carrier iff)**.** *A one-extension history* $\mathsf{E1}(h)$ *lies in the boolean history carrier exactly when its tail is internally the same as* e*.*

Proof. The forward direction is the tail inversion result above. Conversely, if $\mathsf{hsame}(h, \mathsf{e})$, one-constructor congruence gives $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(\mathsf{e}))$, which is the nonempty endpoint case of the boolean history carrier. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryCarrier e1 iff tail empty`

Definition 32.22 (Bool pattern specification)**.** *The pattern is exactly the two displayed mark constructors, with no history growth.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolPatternSpec`

Definition 32.23 (Bool classifier specification)**.** *The classifier is* msame *on marks, with the two constructors distinguished.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolClassifierSpec`
Lean4 (definition) `BEDC.Derived.BoolUp.BoolHistoryClassifier`
Lean4 ✓ `BEDC.Derived.BoolUp.BoolClassifierSpec constructor separation`

Theorem 32.24 (Bool history classifier excludes zero-growth endpoints)**.** *The boolean history classifier cannot relate a history of the form* $\mathsf{E0}(h)$ *from either endpoint position.*

Proof. Unpack the classifier witness. On the left endpoint, its first carrier field is a boolean history carrier witness for $\mathsf{E0}(h)$; on the right endpoint, its second carrier field is such a witness. The zero-growth carrier exclusion eliminates both cases. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier e0 endpoint absurd pair`

Theorem 32.25 (Bool history classifier one-tail sameness)**.** *If the boolean history classifier relates* $\mathsf{E1}(h)$ *to* $\mathsf{E1}(k)$*, then the tails* h *and* k *are internally history-same.*

Proof. Unpack the classifier witness to expose its underlying $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(k))$ field. One-constructor inversion for generated histories then yields $\mathsf{hsame}(h, k)$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier e1 tail hsame`

Theorem 32.26 (Bool history classifier one-tail empty equivalence)**.** *The boolean history classifier relates* $\mathsf{E1}(h)$ *to* $\mathsf{E1}(k)$ *exactly when both tails* h *and* k *are internally history-same as* e*.*

Proof. For the forward direction, unpack the classifier into its endpoint carrier witnesses and apply the one-tail carrier inversion at both endpoints. For the reverse direction, the tail witnesses rebuild both endpoint carriers, while their composite $\mathsf{hsame}(h, k)$ link lifts through the $\mathsf{E1}$ constructor. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier e1 iff tails empty`

Theorem 32.27 (Bool history classifier one-left endpoints)**.** *The boolean history classifier relates* $\mathsf{E1}(h)$ *to* k *exactly when* h *is internally the same as* e *and* k *is internally the same as* $\mathsf{E1}(\mathsf{e})$*.*

Proof. Forward, the left carrier forces the one-extension tail to be empty, and the exposed classifier sameness transports the right endpoint to $\mathsf{E1}(\mathsf{e})$. Reverse, the endpoint sameness witnesses provide both carrier witnesses, and one-extension congruence composed with the right endpoint symmetry gives the classifier link. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier e1 left iff`

Corollary 32.28 (Bool history classifier one-right endpoints)**.** *The boolean history classifier relates* h *to* $\mathsf{E1}(k)$ *exactly when* h *is internally the same as* $\mathsf{E1}(\mathsf{e})$ *and* k *is internally the same as* e*.*

This is the right-endpoint symmetry corollary of Theorem 32.27 and Theorem 32.33, rather than a separate finite-data certificate target.

Proof. Use classifier symmetry to turn a witness for $\mathsf{BoolHistoryClassifier}(h, \mathsf{E1}(k))$ into a witness for $\mathsf{BoolHistoryClassifier}(\mathsf{E1}(k), h)$, then apply Theorem 32.27. This gives $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e}))$. Conversely, the displayed endpoint witnesses satisfy the left-endpoint theorem for $\mathsf{BoolHistoryClassifier}(\mathsf{E1}(k), h)$, and classifier symmetry returns the right-endpoint classifier witness. □

Theorem 32.29 (Bool history classifier cases)**.** *Every boolean history classifier witness relates two histories in the same displayed endpoint class: either both are internally the same as* e*, or both are internally the same as* $\mathsf{E1}(\mathsf{e})$*.*

Proof. Unpack the classifier witness into endpoint carrier data and the internal hsame link. The left endpoint carrier has exactly two cases. In either case, symmetry of the classifier link followed by transitivity transports the left endpoint class to the right endpoint. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier cases`

Theorem 32.30 (Bool history classifier endpoint exactness)**.** *For BEDC histories* h *and* k*,* $\mathsf{BoolHistoryClassifier}(h, k)$ *holds if and only if either* $\mathsf{hsame}(h, \mathsf{e})$ *and* $\mathsf{hsame}(k, \mathsf{e})$*, or* $\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e}))$ *and* $\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e}))$*.*

Proof. For the forward direction, apply the checked case theorem for BoolHistoryClassifier and read off the displayed endpoint class for both histories. For the reverse direction, the empty-empty branch rebuilds the two boolean-history carrier witnesses from the empty endpoint data and composes the left endpoint link with the symmetric right endpoint link to obtain the classifier $\mathsf{hsame}(h, k)$ witness. The one-one branch similarly rebuilds the two one-endpoint carrier witnesses and composes through $\mathsf{E1}(\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier endpoint exactness`

Theorem 32.31 (Bool history classifier constructor separation)**.** *The boolean history classifier cannot relate* e *to* $\mathsf{E1}(\mathsf{e})$*.*

Proof. Eliminating the classifier witness exposes the underlying hsame witness between e and $\mathsf{E1}(\mathsf{e})$. Generated-history constructor no-confusion rules out such a witness. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier constructor separation`

Theorem 32.32 (Bool history classifier transitivity)**.** *If a boolean history classifier relates h to k, and relates k to r, then it relates h to r.*

Proof. Eliminate the two classifier witnesses. The endpoint carrier witnesses for h and r are retained, while the exposed $\mathsf{hsame}(h, k)$ and $\mathsf{hsame}(k, r)$ links compose by generated-history transitivity. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier trans`

Theorem 32.33 (Bool history classifier symmetry)**.** *If the boolean history classifier relates h to k, then it relates k to h.*

Proof. Eliminate the classifier witness. The two endpoint carrier witnesses exchange places, and the exposed $\mathsf{hsame}(h, k)$ link reverses by generated-history symmetry. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier symm`

Theorem 32.34 (Bool history classifier endpoint transport)**.** *If h is internally the same as h' and k is internally the same as k', then the boolean history classifier transports from (h, k) to (h', k').*

Proof. Eliminate the classifier witness into its two endpoint carrier proofs and the exposed $\mathsf{hsame}(h, k)$ link. Transport each endpoint carrier along its corresponding hsame proof, then compose symmetry of the left endpoint with the stored classifier link and the right endpoint transport. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier hsame transport`

Theorem 32.35 (Bool history classifier empty-left equivalence)**.** *The boolean history classifier relates* e *to k exactly when k is internally the same history as* e*.*

Proof. For the forward direction, eliminate the classifier witness and symmetrize the exposed $\mathsf{hsame}(\mathsf{e}, k)$ link. For the reverse direction, the empty carrier witness supplies the left endpoint, the assumed $\mathsf{hsame}(k, \mathsf{e})$ supplies the right endpoint, and symmetry supplies the classifier link. □

Lean4 ✓ `BEDC.Derived.BoolUp.BoolHistoryClassifier empty left iff`

Corollary 32.36 (Bool history classifier empty-right equivalence)**.** *The boolean history classifier relates h to* e *exactly when h is internally the same history as* e*. This is the empty-endpoint symmetry corollary of Theorem 32.35 and Theorem 32.33, rather than a separate finite-data certificate target.*

Proof. For the forward direction, use classifier symmetry to move the empty endpoint to the left, then apply Theorem 32.35. For the reverse direction, apply Theorem 32.35 to the assumed h-to-e sameness, then use classifier symmetry again. □

Definition 32.37 (Bool stability certificate)**.** *The checked mark-level structural subcertificate for* BoolClassifierSpec *consists of:*

1. *reflexivity of* msame *on* b0 *and* b1
2. *constructor separation: no classifier path identifies* b0 *with* b1
3. *closure under the two mark constructors*

The history-level $\mathsf{Bool}^{\uparrow}$ *stability used by Theorem 32.39 is supplied by Theorem 32.16, Theorem 32.32, Theorem 32.33, Theorem 32.34, and the endpoint exclusion and endpoint case theorems immediately above.*

Lean4 ✓ `BEDC.Derived.BoolUp.bool stability certificate fields`
↪ variant `BEDC.Derived.BoolUp.boolClassifierSpec stability`
↪ variant `BEDC.Derived.BoolUp.bool stability certificate`

Definition 32.38 (Bool mark ledger policy)**.** *The mark-level* BoolLedgerPolicy$(source, visible)$ *records a raw source mark and its visible mark under* msame$(source, visible)$*.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolLedgerPolicy`

Theorem 32.39 (Concrete $\mathsf{Bool}^{\uparrow}$ core history certificate)**.** *The history carrier* BoolHistoryCarrier *and classifier* BoolHistoryClassifier *form a core* NameCert *for* $\mathsf{Bool}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BoolUp.bool history name certificate`

Definition 32.40 (Bool history semantic fields)**.** *The semantic history ledger used for Theorem 32.41 is the carrier-as-ledger predicate* BoolHistoryCarrier*, not the mark-level* BoolLedgerPolicy*. The history-level semantic fields used by the boolean certificate are all the same carrier predicate. For a BEDC history h,*

$$\mathsf{PatternSpec}(h) \quad \textit{and} \quad \mathsf{LedgerPolicy}(h)$$

are both read as BoolHistoryCarrier(h)*, namely that h is internally the same as one of the two visible boolean endpoints:*

$$\mathsf{hsame}(h, \mathsf{e}) \quad \textit{or} \quad \mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})).$$

Theorem 32.41 (BoolUp paper-side semantic certificate target)**.** *The canonical public-object certificate proposition for* $\mathsf{Bool}^{\uparrow}$ *is the following formal implication: for any core witness*

$$\mathsf{core} : \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}),$$

derive

$$\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).$$

Here source, pattern, and ledger are all fixed to BoolHistoryCarrier*, and the classifier is fixed to* BoolHistoryClassifier*. Equivalently, the three history predicates are*

$$\mathsf{SourceSpec} = \mathsf{PatternSpec} = \mathsf{LedgerPolicy} = \mathsf{BoolHistoryCarrier}.$$

Its SemanticNameCert *record decomposes as follows:* core *is the supplied core witness;* pattern_sound *is the identity implication on* BoolHistoryCarrier*,*

$$\forall h, \mathsf{BoolHistoryCarrier}(h) \Rightarrow \mathsf{BoolHistoryCarrier}(h),$$

and ledger_sound *is the identity implication on* BoolHistoryCarrier*,*

$$\forall h, \mathsf{BoolHistoryCarrier}(h) \Rightarrow \mathsf{BoolHistoryCarrier}(h).$$

The displayed proposition is the paper-side lifting principle from a core $\mathsf{Bool}^{\uparrow}$ *history certificate to the public-object certificate scope. Specializing* core *to Theorem 32.39 gives the closed Bool public-object certificate in Theorem 32.42. The implication is the instance of the generic carrier-self semantic construction in Theorem 26.9; the closed Bool site is checked by* $BEDC.Derived.BoolUp.bool_history_sem$

Proof. Start with an arbitrary core witness core : NameCert(BoolHistoryCarrier, BoolHistoryClassifier). Apply Theorem 26.9 to core. Since source, pattern, and ledger are all BoolHistoryCarrier, the semantic pattern and ledger fields are the identity implication on the Bool history carrier predicate. Taking core to be Theorem 32.39, equivalently the checked Lean theorem $BEDC.Derived.BoolUp.bool_history_name$ yields the Bool instance of the displayed implication. □

Theorem 32.42 (Concrete BoolUp semantic certificate)**.** *The concrete BoolUp public-object semantic certificate holds:*

$$\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).$$

Proof. Apply Theorem 26.9 to the concrete core certificate in Theorem 32.39. Since source, pattern, and ledger are all BoolHistoryCarrier, the semantic pattern and ledger fields are the identity implications on the Bool history carrier. □

Lean4 ✓ `BEDC.Derived.BoolUp.bool history semantic name certificate`

Definition 32.43 (Bool public endpoint constructors)**.** *The public false constructor is* BoolEndpoint(b0) = e*, and the public true constructor is* BoolEndpoint(b1) = E1(e)*. Public boolean operations are finite endpoint tables whose input and output rows are drawn only from these two constructor rows. No row in this public surface is a host proposition, a decidable predicate, or an eliminator.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolPublicEndpointConstructors`

Theorem 32.44 (Bool public constructor case readback)**.** *Every public Bool endpoint read is classified by exactly one of the displayed false and true endpoint rows of Definition 32.70. Two public constructor readbacks agree exactly when their underlying marks are related by* msame*.*

Proof. Apply Definition 32.5 to the public read. It supplies a displayed mark v : BMark and a witness that the read history is the endpoint BoolEndpoint(v).

Case analysis on v gives the two public rows BoolEndpoint(b0) = e and BoolEndpoint(b1) = E1(e). Theorem 32.6 gives uniqueness of the readback mark, and Definition 32.4 identifies agreement of endpoint rows with msame of the underlying marks. □

Theorem 32.45 (Bool public surface no-hidden leakage)**.** *A consumer of the public Bool surface can read only the two constructor endpoint rows, endpoint-table operation rows over those constructors, the mark and history classifier rows, and the semantic certificate ledger. It cannot read a host proposition, a decidable predicate, an eliminator, or truth-table completeness.*

Proof. The public constructors are exactly the two endpoint rows in Definition 32.70. The endpoint coverage and constructor case readback statements show that public reads return one of these rows, with agreement controlled by msame and endpoint exactness rather than by host logic.

The semantic certificate in Definition 32.42 records the carrier, classifier, and ledger discipline. Since the displayed public rows contain no host proposition, decidable predicate, eliminator, or truth-table-completeness row, those objects are not available to a surface consumer. □

The endpoint coverage, endpoint bridge classifier completeness, and endpoint readback theorems justify the Bool public-object readback and bridge promised by the roadmap template; they are not internal fields of the SemanticNameCert constructor. Option$^{\uparrow}$consumes this two-row endpoint discipline as its absent/present tag surface in chapter 33.

Corollary 32.46 (Bool$^{\uparrow}$ certificate scope)**.** *Theorem 32.39 cites the checked core history* NameCert *for* BoolHistoryCarrier *and* BoolHistoryClassifier*, namely the BEDC.Derived.BoolUp Lean target* bool_history_name_certificate*. The public-object* SemanticNameCert *scope is checked separately at Theorem 32.42 by the BEDC.Derived.BoolUp Lean target* bool_history_semantic_name_certificate*. Thus the core certificate, the mark-level ledger definition, the generic carrier-as-ledger implication, and the closed semantic certificate occupy separate roles. In each case* Bool$^{\uparrow}$ *is derived from mark and history sameness, and is not primitive.*

Lemma 32.47 (Bool public endpoint forgetful projection boundary)**.** *The public constructor surface of Definition 32.70 admits a BEDC-native forgetful projection. Start with a public constructor readback row and erase the constructor-row name, endpoint-table name, and public-surface annotation while retaining the displayed endpoint history. The retained target data are exactly*

BoolHistoryCarrier, BoolHistoryClassifier, BoolHistoryCarrier *as source, pattern, and ledger*, NameCert(BoolHis

together with the public semantic certificate

SemanticNameCert(BoolHistoryCarrier, BoolHistoryCarrier, BoolHistoryCarrier, BoolHistoryClassifier).

More concretely, each retained public read has a mark v : BMark *and an endpoint witness* hsame(h, BoolEndpoint(v)) *unique up to* msame*. For two retained reads* h, k *with endpoint witnesses through* v, w : BMark*, the projected classifier row* BoolHistoryClassifier(h, k) *holds exactly when* msame(v, w) *holds. The projection retains no constructor-row name, endpoint-table name, host proposition, decidable predicate, eliminator, truth-table-completeness datum, payload carrier, payload classifier, or option-packaging row.*

Proof. The public constructor rows are exactly the two endpoint rows displayed in Definition 32.70. Applying Definition 32.71 to a public read supplies a mark v : BMark and an endpoint witness hsame(h, BoolEndpoint(v)), and Definition 32.6 makes the displayed mark unique up to msame.

The retained carrier coordinate is forced by Definition 32.5: an endpoint witness against BoolEndpoint(v) repacks as BoolHistoryCarrier(h), and every Bool carrier row is covered by such an endpoint witness. Thus erasing the public constructor name leaves a history in the same target carrier rather than a separate codomain carrier.

For the classifier coordinate, take two retained reads h, k with endpoint witnesses through v, w. If $\mathsf{msame}(v, w)$ holds, the reverse direction of Definition 32.12 rebuilds $\mathsf{BoolHistoryClassifier}(h, k)$ from the two endpoint witnesses and the mark comparison. Conversely, if $\mathsf{BoolHistoryClassifier}(h, k)$ holds, Definition 32.12 supplies endpoint witnesses through marks v', w' with $\mathsf{msame}(v', w')$. The readback determinism theorem compares v with v' and w with w', and transitivity of mark sameness gives $\mathsf{msame}(v, w)$. Hence the classifier retained by the projection is exactly the mark-level agreement already visible in the constructor readback.

The certificate coordinates are fixed by Definition 32.40, Definition 32.39, and Definition 32.42. Source, pattern, and ledger are all the same predicate $\mathsf{BoolHistoryCarrier}$, the core name certificate is $\mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier})$, and the public semantic certificate is the displayed carrier-self semantic certificate. Finally, Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness, while the Bool public surface contains no payload carrier, payload classifier, or option-packaging coordinate. Those named rows are therefore erased rather than transported by the forgetful projection. □

Lemma 32.48 (Bool public endpoint constructor classifier alignment boundary)**.** *For histories $h, k : \mathsf{BHist}$, the public Bool endpoint surface aligns its classifier boundary exactly with the two displayed constructor endpoint rows:*

$$\mathsf{BoolHistoryClassifier}(h, k) \iff \exists v, w : \mathsf{BMark},\ e_v, e_w : \mathsf{BHist},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge$$

Under this equivalence, the transported public rows are exactly the two endpoint readback rows against the endpoint histories e_v and e_w determined by $\mathsf{BoolEndpoint}(v)$ and $\mathsf{BoolEndpoint}(w)$, together with the mark-comparison row $\mathsf{msame}(v, w)$. The retained certificate rows are the carrier, classifier, and semantic ledger rows named by the Bool semantic certificate. Rows for host propositions, decidable predicates, eliminators, and truth-table completeness are forgotten by the public endpoint classifier boundary.

Proof. The displayed equivalence is Definition 32.13 read through the explicit endpoint-history witnesses supplied by Definition 32.70. Its forward direction says that a classifier comparison between h and k exposes only source-qualified marks v, w, endpoint histories e_v, e_w with the displayed constructor equations, the two endpoint readback rows $\mathsf{hsame}(h, e_v)$ and $\mathsf{hsame}(k, e_w)$, and the mark-comparison row $\mathsf{msame}(v, w)$. Its reverse direction says that precisely those rows rebuild the Bool history classifier comparison, so no further classifier coordinate is being transported.

By Definition 32.70, the public constructors are only $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The case readback theorem Definition 32.71 therefore identifies every public endpoint read with exactly one of those two rows, and Definition 32.6 makes the chosen mark unique up to msame. The endpoint bridge exactness of Definition 32.4 then makes the row comparison a mark comparison rather than a host-logical comparison.

The certificate side contributes no extra public row. The semantic certificate in Definition 32.42 records the Bool carrier, classifier, and ledger discipline over the same history carrier. Finally, Definition 32.72 states that the public surface does not expose a host proposition, decidable predicate, eliminator, or truth-table completeness row. Hence the boundary transports exactly the two constructor readback rows and their msame classifier comparison, retains only the semantic certificate ledger rows, and forgets the non-public rows listed above. □

Lemma 32.49 (Bool public constructor target-source mirror boundary)**.** *For the public Bool constructor surface of Definition 32.70, the target-facing constructor read and the source-facing carrier*

read are the same finite endpoint packet viewed in opposite directions. Concretely, a public target read through either displayed constructor row $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ *or* $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$ *has a source mirror consisting exactly of the same mark* $v : \mathsf{BMark}$*, an endpoint history witness* $e_v : \mathsf{BHist}$ *satisfying* $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$*, the same readback observation row* $\mathsf{hsame}(h, e_v)$*, the same classifier comparison row* $\mathsf{msame}(v, w)$ *when another constructor read is compared through a mark* $w : \mathsf{BMark}$ *and endpoint history witness* $e_w : \mathsf{BHist}$ *satisfying* $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$*, and the same semantic certificate rows*

$$\mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}), \quad \mathsf{SemanticNameC}$$

Thus the source side is obtained by renaming the displayed target endpoint packet back to its carrier readback packet. The boundary is negative for any larger mirror: there is no separate source-side constructor carrier, result row, termination row, host proposition row, decidable predicate row, eliminator row, truth-table-completeness row, payload carrier row, payload classifier row, or option-packaging row.

Proof. Start with a public target read. By Definition 32.70, the only target constructor rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The case readback theorem Definition 32.71 therefore supplies a displayed mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ with $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$, and identifies the read with the endpoint readback row $\mathsf{hsame}(h, e_v)$. Its agreement clause, together with Definition 32.6 and Definition 32.4, says that any comparison between two such reads is exactly the mark comparison $\mathsf{msame}(v, w)$.

The source mirror is forced by endpoint coverage. The reverse direction of Definition 32.5 repacks the readback row $\mathsf{hsame}(h, e_v)$, for an endpoint witness e_v satisfying $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$, as the source carrier row $\mathsf{BoolHistoryCarrier}(h)$, while the forward direction says that every source carrier row used by this Bool surface is covered by such an endpoint readback. Hence the target row and source row do not form two independent constructors; they are the same displayed history endpoint read as target endpoint or as source carrier.

For compared reads h, k, Definition 32.12 identifies the source-side classifier row $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint readbacks through marks v, w, endpoint history witnesses $e_v, e_w : \mathsf{BHist}$ satisfying $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$ and $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$, and the same comparison row $\mathsf{msame}(v, w)$. The source-qualified form in Definition 32.13 adds only the already displayed source qualification of those two finite marks, so it does not introduce a further target or source coordinate.

The certificate rows are exactly the rows already named by Definition 32.40, Definition 32.39, and Definition 32.42: source, pattern, and ledger are all read as $\mathsf{BoolHistoryCarrier}$, and the classifier is $\mathsf{BoolHistoryClassifier}$. Finally, Definition 32.72, Definition 32.74, and Definition 32.75 rule out the larger mirror coordinates listed in the statement. The target-source mirror therefore closes exactly at the finite endpoint packet and its Bool semantic certificate rows. □

Lemma 32.50 (Bool public endpoint bridge cut-elimination boundary)**.** *The public Bool endpoint constructors of Definition 32.70 admit a bridge cut-elimination boundary over the displayed local rows. For a final consumer read of the public Bool surface, every route row factors through the*

finite packet

$$\begin{gathered}\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}),\\ v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist},\\ ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))),\\ ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))),\\ \mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w),\\ \mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}),\\ \mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).\end{gathered}$$

The two displayed constructor endpoint rows, the endpoint history readback rows, the mark-comparison row, and the carrier, classifier, core certificate, and semantic certificate rows are noneliminable at this boundary: removing any one of them loses either the public endpoint choice, the readback into a history, the classifier comparison, or the certificate surface consumed by the final Bool object. Every other named coordinate appearing in the local public-surface discussion is eliminated or rejected: constructor-row names, endpoint-table names, and public-surface annotations are projected away; host propositions, decidable predicates, eliminators, truth-table-completeness data, payload carriers, payload classifiers, option-packaging rows, separate source-side constructor carriers, result rows, termination rows, and continuation rows are not consumer-visible rows of this Bool endpoint bridge.

Proof. The finite endpoint rows are exactly the two rows displayed in Definition 32.70. Thus any public constructor read begins at either $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ or $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$, and no third constructor, host predicate, or eliminator row is available at the source of the route. Applying Definition 32.71 to the read supplies a mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ with $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$ and $\mathsf{hsame}(h, e_v)$; applying it to a second read supplies $w : \mathsf{BMark}$ and an endpoint history witness $e_w : \mathsf{BHist}$ with $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$ and $\mathsf{hsame}(k, e_w)$. The uniqueness part is controlled by Definition 32.6, so the chosen endpoint mark cannot be replaced by an unlisted host or source coordinate.

The comparison row is also forced. By Definition 32.4, comparison of the two endpoint rows is exactly comparison of their marks by $\mathsf{msame}(v, w)$. Equivalently, Definition 32.12 identifies the history classifier row $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint history readbacks and the same mark-comparison row. Hence a bridge consumer cannot keep an independent comparison route after those readback rows have been exposed; the classifier route has already cut through the endpoint packet.

The carrier and certificate side is fixed by the already displayed projection boundary. By Definition 32.5, the endpoint readback through e_v repacks as the carrier row $\mathsf{BoolHistoryCarrier}(h)$, and every carrier row used here is covered by such an endpoint readback. By Definition 32.74, erasing constructor-row names, endpoint-table names, and public-surface annotations leaves exactly the carrier, classifier, carrier-as-source-pattern-ledger, core $\mathsf{NameCert}$, and public semantic certificate rows. By Definition 32.40, Definition 32.39, and Definition 32.42, these rows are the Bool history carrier, the Bool history classifier, and the carrier-self semantic certificate displayed in the statement.

The remaining possible cuts are negative. Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness from the public consumer surface. Definition 32.75 shows that the classifier boundary transports only the two endpoint readback rows and their msame comparison, while Definition 32.76 shows that target-facing and source-facing reads

are the same finite endpoint packet rather than separate constructor carriers, result rows, termination rows, or continuation rows. Since the local chapter displays no additional host, source, handoff, continuation, payload, or option-packaging coordinate, any such coordinate is rejected as unlisted rather than retained by the bridge.

Consequently the final consumer-visible surface is precisely the finite packet named in the statement. The endpoint rows, readback rows, mark-comparison row, carrier and classifier rows, and the two certificate rows are noneliminable because they are the rows used to reconstruct the endpoint choice, the history read, the classifier comparison, and the public Bool certificate. All other local route names are either projected away by the forgetful boundary or excluded by the public-surface and mirror-boundary theorems. □

Lemma 32.51 (Bool public endpoint equivalence-completion boundary)**.** *The public Bool endpoint constructor surface of Definition 32.70 has a complete equivalence boundary over its displayed local classifier and readback rows. For histories $h, k : \mathsf{BHist}$, the carrier and classifier two-sided part is exactly*

$$\begin{aligned}
\mathsf{BoolHistoryCarrier}(h) &\iff \exists v : \mathsf{BMark},\ \exists e_v : \mathsf{BHist},\ ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))) \wedge \mathsf{hsame}(\\
\mathsf{BoolHistoryClassifier}(h, k) &\iff \exists v, w : \mathsf{BMark},\ \exists e_v, e_w : \mathsf{BHist},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge ((v =
\end{aligned}$$

For marks $v, w : \mathsf{BMark}$ and endpoint-history witnesses $e_v, e_w : \mathsf{BHist}$ satisfying the displayed constructor equations

$$\begin{aligned}
&((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))),\\
&((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))),
\end{aligned}$$

the endpoint-agreement part is exactly

$$\mathsf{msame}(v, w) \iff \mathsf{hsame}(e_v, e_w).$$

The endpoint witnesses are precisely the same visible endpoint rows as the public false and true constructor rows: $v = \mathsf{b0}$ carries $e_v = \mathsf{e}$, and $v = \mathsf{b1}$ carries $e_v = \mathsf{E1}(\mathsf{e})$, with the same alternatives for w, e_w. Thus any displayed forward carrier, classifier, endpoint-agreement, or constructor-readback implication in the local chapter has its reverse implication on the same finite rows.

The remaining displayed directions in the local public-surface discussion are one-way interpretation boundaries. Erasing constructor-row names, endpoint-table names, and public-surface annotations maps a public constructor read to the carrier, classifier, core NameCert*, and public semantic certificate rows, but those retained rows do not reconstruct the erased names or annotations. The no-hidden-leakage row is also one-way: it excludes host propositions, decidable predicates, eliminators, truth-table-completeness data, payload carriers, payload classifiers, option-packaging rows, separate source-side constructor carriers, result rows, termination rows, and continuation rows from the public endpoint surface, and no reverse implication is claimed from their absence to any additional public constructor structure.*

Proof. The carrier row is already two-sided by Definition 32.5. Its forward direction reads a Bool carrier history as one of the two endpoint histories, witnessed by some $e_v : \mathsf{BHist}$ satisfying $(v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))$; its reverse direction repacks the same endpoint readback witness as $\mathsf{BoolHistoryCarrier}(h)$. Since Definition 32.70 displays only the two constructor histories e and $\mathsf{E1}(\mathsf{e})$, this carrier equivalence has no third public endpoint row.

The classifier row is two-sided by Definition 32.13. The forward implication exposes source-qualified marks v, w, endpoint-history witnesses e_v, e_w with their constructor equations, and the mark comparison $\mathsf{msame}(v, w)$. The reverse implication rebuilds $\mathsf{BoolHistoryClassifier}(h, k)$ from

exactly those rows. The more explicit constructor-row version is Definition 32.75, which identifies the endpoint witnesses through the two displayed constructor histories e and $\mathsf{E1}(\mathsf{e})$.

Endpoint agreement is two-sided by Definition 32.4. The case-readback theorem Definition 32.71 then says that a public constructor read is classified by exactly one of the displayed false and true endpoint rows, represented by an endpoint-history witness e_v with its constructor equation, while Definition 32.6 makes the chosen mark unique up to msame. Hence the visible constructor readback direction is completed by the same endpoint-history witnesses, constructor equations, and mark-comparison rows, not by a host proposition or by an external equivalence carrier.

It remains to identify the directions that are not equivalences. The forgetful projection of Definition 32.74 deliberately erases constructor-row names, endpoint-table names, and public-surface annotations. Its target rows are the retained carrier, classifier, $\mathsf{NameCert}$, and public semantic certificate rows fixed by Definition 32.40, Definition 32.39, and Definition 32.42; those rows remember the endpoint-history witness and classifier comparison, not the erased presentation names. This is therefore a one-way interpretation boundary.

The negative rows are one-way for the same local reason. By Definition 32.72, the public surface exposes no host proposition, decidable predicate, eliminator, or truth-table-completeness row. By Definition 32.76, the target-facing and source-facing reads are the same finite endpoint packet rather than separate constructor carriers, result rows, termination rows, or continuation rows. By Definition 32.77, every final consumer route factors through the finite endpoint packet and the Bool semantic certificate rows, while payload and option-packaging coordinates are rejected as unlisted. Thus the local chapter has no unmatched forward classifier or readback implication: the carrier, classifier, endpoint-agreement, and constructor-readback rows are completed by reverse implications on the same visible rows, and the projection and exclusion rows are recorded as one-way interpretation boundaries. □

Lemma 32.52 (Bool public endpoint substrate bisimulation boundary)**.** *The public Bool endpoint constructors of Definition 32.70 admit a substrate relation boundary, not a definitional-equality boundary. The BEDC-local relation certificate has source rows only from the displayed finite endpoint packet*

$$\begin{gathered}
\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}), \\
v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist}, \\
((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))), \quad ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))), \\
\mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w), \\
\mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}), \\
\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).
\end{gathered}$$

Its forward leg sends a public endpoint read through $\mathsf{hsame}(h, e_v)$ *to the carrier row* $\mathsf{BoolHistoryCarrier}(h)$*. Its reverse leg reads every carrier row used by the Bool public surface back through one of the two displayed endpoint rows. Its comparison leg is the reflected equivalence*

$$\begin{aligned}
\mathsf{BoolHistoryClassifier}(h, k) \quad \Longleftrightarrow \quad \exists v, w : \mathsf{BMark},\ e_v, e_w : \mathsf{BHist},\ & ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))) \wedge \\
& ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))) \wedge \\
& \mathsf{hsame}(h, e_v) \wedge \mathsf{hsame}(k, e_w) \wedge \mathsf{msame}(v, w).
\end{aligned}$$

Thus a BHist*-facing reading of the public Bool endpoint surface is a substrate mirror, and any consumer that names a* $\mathsf{Rule110}$*, cellular, binary, or* $\mathsf{FKernel}$ *substrate must state its use as a*

bisimulation or row-embedding relation whose Bool side is exactly the packet above. No such substrate name is available here as a definitional equality with the Bool public constructors. Any host coordinate, source coordinate, transition coordinate, external substrate row, result row, termination row, continuation row, payload row, option-packaging row, or equality row not listed in the packet is rejected at this boundary rather than silently identified with the Bool public surface.

Proof. The local constructor source is exhausted by Definition 32.70: the only public endpoint rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Applying Definition 32.71 to a public read supplies a displayed mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ in one of those two cases, together with the readback row $\mathsf{hsame}(h, e_v)$. The determinacy clause of Definition 32.6 prevents replacement of that mark by an unlisted host or substrate coordinate.

The carrier leg is exactly a mirror leg. By Definition 32.5, the endpoint readback row repacks as $\mathsf{BoolHistoryCarrier}(h)$, and every carrier row used by this Bool surface is covered by such an endpoint readback. Hence the relation runs in both directions between the public endpoint packet and the BHist carrier readback packet, but the two sides are not introduced as a single definitional object; they are the same displayed history read viewed through endpoint and carrier roles.

The comparison leg is the bisimulation clause. For two reads h, k with endpoint witnesses through v, w, Definition 32.12 identifies $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint readbacks and the mark comparison $\mathsf{msame}(v, w)$. The explicit endpoint-row form is the classifier alignment boundary of Definition 32.75. Thus classifier comparison is both preserved and reflected by the endpoint packet: endpoint mark agreement rebuilds the history classifier, and a history-classifier comparison exposes endpoint witnesses and mark agreement.

The certificate rows are fixed by Definition 32.74 and Definition 32.77. After constructor-row names, endpoint-table names, and public-surface annotations are erased, the retained rows are exactly the carrier, classifier, carrier-as-source-pattern-ledger, core NameCert, and public semantic certificate rows displayed in the statement. The target-source mirror theorem Definition 32.76 further says that the source-facing and target-facing reads are opposite views of the same finite endpoint packet, not separate constructor carriers.

Finally, the negative part follows from the same row exhaustion. Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness from the public surface, while Definition 32.77 rejects unlisted source, result, termination, continuation, payload, and option-packaging coordinates. The chapter displays no Rule110, cellular, binary, or FKernel substrate row inside the Bool public endpoint constructors. Therefore any later consumer using one of those substrate names must provide its own target-side rows and relate them to the Bool packet by bisimulation, row embedding, or substrate mirror. The BEDC-local Bool boundary supplies no definitional equality from the public endpoint constructors to such a substrate. □

Theorem 32.53 (Bool finite-data exit route)**.** *The public* $\mathsf{Bool}^{\uparrow}$ *finite-data route consists exactly of the two endpoint constructor rows of Definition 32.70, the constructor case readback of Definition 32.71, the no-hidden-leakage boundary of Definition 32.72, and the semantic certificate scope of Definition 32.73. Any* $\mathsf{Option}^{\uparrow}$ *absent/present tag read,* $\mathsf{Prod}^{\uparrow}$ *sibling-lattice read, or* $\mathsf{List}^{\uparrow}$ *finite-data sibling route that consumes* $\mathsf{Bool}^{\uparrow}$ *passes through those rows and receives no host proposition, eliminator, decidable predicate, or truth-table completeness.*

Proof. The two constructor rows are the public endpoints displayed in Definition 32.70. The case readback theorem then says that every public endpoint read is one of those rows, with agreement measured by msame rather than by an external Boolean predicate.

The no-hidden-leakage theorem removes the extra host-logic surfaces from the public exit, while Definition 32.73 identifies the checked core and semantic certificate roles that remain available to

downstream finite data consumers.

The consumer list is exactly the existing finite-data route through Definition 32.59, Definition 34.110, and Definition 36.214. These routes use the same endpoint rows as absent/present tags, product sibling coordinates, and list finite-data siblings; none adds a primitive Boolean object or a host truth-table principle. □

Lemma 32.54 (Bool public endpoint downstream coverage)**.** *Every accepted local consumer of the public Bool endpoint constructors is covered by the finite alternatives and ledger rows already displayed in this file. More precisely, after the consumer is reduced to its public endpoint readback and Bool certificate scope, its visible rows lie in the finite packet*

$$\begin{gathered}\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}),\\ v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist},\\ ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))), \quad ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))),\\ \mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w),\\ \mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}),\\ \mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).\end{gathered}$$

The alternatives outside this packet remain outside the Bool BEDC public surface: no host proposition, decidable predicate, eliminator, truth-table-completeness row, payload carrier, payload classifier, option-packaging row, separate source-side constructor carrier, result row, termination row, continuation row, external substrate row, or definitional-equality row is accepted as a downstream Bool endpoint constructor row.

Proof. Let a local downstream consumer be accepted by the public Bool endpoint constructor surface. By Definition 32.70, the only public constructor alternatives available at the entry point are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Applying Definition 32.71 to the consumer read supplies a mark $v : \mathsf{BMark}$ and an endpoint-history witness $e_v : \mathsf{BHist}$ in one of those two displayed cases, together with the readback row $\mathsf{hsame}(h, e_v)$. If the consumer compares two accepted reads, the same argument supplies $w : \mathsf{BMark}$, $e_w : \mathsf{BHist}$, and $\mathsf{hsame}(k, e_w)$.

The equivalence-completion boundary of Definition 32.78, read with the explicit transport theorems Definition 32.16 and Definition 32.34, then exhausts the possible downstream classifier data. For the carrier leg, Definition 32.5 gives the two endpoint witnesses displayed above, and Definition 32.16 is the transport citation that carries that endpoint witness along the readback row $\mathsf{hsame}(h, e_v)$ to the usable carrier row $\mathsf{BoolHistoryCarrier}(h)$. For the classifier leg, Definition 32.13 gives source-qualified marks, the two displayed constructor equations for e_v and e_w, the two endpoint readback rows, and the mark comparison $\mathsf{msame}(v, w)$; Definition 32.34 is the transport citation for moving this classifier row along the endpoint readbacks. For the endpoint-agreement leg, Definition 32.4 reduces comparison of endpoint histories to the same mark comparison, with the endpoint-readback movement again governed by Definition 32.34. Hence every accepted carrier, classifier, or agreement row used downstream is already one of the rows in the finite packet above.

The certificate scope contributes only the named Bool certificate rows. By Definition 32.73, the core NameCert over BoolHistoryCarrier and BoolHistoryClassifier and the public SemanticNameCert over the same carrier, pattern, ledger, and classifier roles are separate checked scopes, with $\mathsf{Bool}^{\uparrow}$ derived from mark and history sameness rather than primitive host truth. Thus a downstream consumer receives the carrier, classifier, core certificate, and public semantic certificate rows listed in the packet, and no additional Boolean object is introduced by the certificate citation.

The exclusion clause is the one-way side of the same local boundary. Definition 32.78 records that forgetful projection and no-hidden-leakage directions do not reconstruct erased constructor-row names, endpoint-table names, or public-surface annotations, and do not turn absence of host data into further public constructor structure. Together with Definition 32.72, Definition 32.77, and Definition 32.79, this rules out host propositions, decidable predicates, eliminators, truth-table completeness, payload and option coordinates, separate source, result, termination, continuation, substrate, and definitional-equality rows. Therefore the accepted local downstream consumer is exhausted by the finite alternatives, readback rows, classifier row, and certificate ledger rows displayed in this file, while every excluded alternative remains outside the Bool BEDC public endpoint packet. □

Theorem 32.55 (Bool finite-data bridge interface)**.** *The endpoint bridge of Definition 32.3, endpoint coverage, classifier endpoint exactness, constructor separation, semantic fields, and the certificate of Definition 32.42 form the bridge-ready finite-data interface from* $\mathsf{Bool}^{\uparrow}$ *to the standard two-endpoint Bool surface. Concretely, the interface consists of Definition 32.4, Definition 32.5, Definition 32.30, Definition 32.31, Definition 32.40, and Definition 32.42.*

Lean4 ✓ `BEDC.Derived.BoolUp.BoolUp StdBridge`

Proof. The endpoint bridge supplies the two visible endpoints, and Definition 32.4 identifies endpoint agreement with the kernel mark equality read by the Bool bridge.

Endpoint coverage follows from Definition 32.5. Classifier exactness and constructor separation are supplied by Definition 32.30 and Definition 32.31, so the two constructors are both covered and separated.

The semantic fields of Definition 32.40 and the public certificate of Definition 32.42 package the carrier, pattern, and ledger rows into the finite Bool surface. These are exactly the finite data a bridge layer may consume, without asserting a machine-checked standard bridge in this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$**.**

Constructive story: A Boolean here is a single mark from the kernel's two-element mark type — b0 or b1 — with no further structure layered on top. Whichever mark you start from, you stay there: equality on this tiny type is literal identity, and the four basic facts about it (each mark equals itself, and b0 differs from b1, in either reading order) form the entire distinguishing pattern, with every visible mark being just the raw mark itself. Because the type has only two elements and identity on it is automatically an equivalence, the five name-certificate obligations — that the carrier admits at least one element, and that the classifier is reflexive, symmetric, transitive, and respected by the carrier — fall out together without further work. The signature gap of a Boolean token p collects those marks m for which the source-domain admits m, the signature relation pairs m with an answer mark and an evidence ledger, and the token-introduction relation maps that answer to p; once the trivial pattern and ledger machinery is chased away, this becomes just $m = p$. The chapter registers all of this under the name $\mathsf{Bool}^{\uparrow}$; nothing beyond the kernel's mark type and its equality enters, and full Boolean algebra — operations, eliminators, decidable predicates — stays explicitly outside this closure level.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public Bool surface covers the false and true endpoint constructors, constructor case readback, semantic certificate ledger, no-hidden-leakage statement, and Definition 32.85 as the mature finite-data bridge surface over the displayed mark and history classifiers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.BoolUp.bool\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full boolean algebra or proposition theory: boolean operations, eliminators, decidable predicates, truth-table completeness, and logic connectives as BEDC primitives.

Upgrade path: No further formal-axis upgrade remains for this bridge-checked chapter.

Lemma 32.56 (Bool finite-data classifier alignment boundary)**.** *For histories h, k : BHist, the classifier side of the Bool finite-data bridge interface has the following exact alignment boundary:*

$$\mathsf{BoolHistoryClassifier}(h,k) \iff \exists v, w : \mathsf{BMark},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \mathsf{hsame}(h, \mathsf{BoolEndpoint}$$

Under this equivalence, the transported displayed rows are precisely the two endpoint readback rows $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ *and* $\mathsf{hsame}(k, \mathsf{BoolEndpoint}(w))$*, together with the mark-comparison row* $\mathsf{msame}(v, w)$*. The retained certificate rows are precisely the source, pattern, and ledger rows of Definition 32.40, all read as* $\mathsf{BoolHistoryCarrier}$*. Every row outside the interface listed in Definition 32.85 is forgotten by this alignment boundary; in particular, it exports no host proposition, decidable predicate, eliminator, truth-table-completeness row, payload carrier, payload classifier, or option-packaging row.*

Proof. The displayed equivalence is exactly Definition 32.13. Its forward direction says that a classifier comparison between h and k exposes only source-qualified endpoint marks v, w, the two endpoint readback rows, and the mark comparison $\mathsf{msame}(v, w)$. Its reverse direction says that those rows are already sufficient to rebuild the Bool history classifier comparison, so no additional classifier coordinate is consumed.

By Definition 32.3, the endpoint map has only the two visible target rows $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Endpoint coverage in Definition 32.5 supplies no third readback case, and constructor separation in Definition 32.31 prevents the two constructor rows from being identified by the classifier. Thus the readback part of the boundary transports exactly the displayed endpoint rows and their msame comparison.

The certificate part is fixed by Definition 32.40: source, pattern, and ledger are all the same predicate $\mathsf{BoolHistoryCarrier}$. The public semantic certificate in Definition 32.42 therefore retains only the carrier-classifier ledger named in Definition 32.85. Finally, Definition 32.72 rules out host propositions, decidable predicates, eliminators, and truth-table completeness, and the finite-data interface list contains no payload carrier, payload classifier, or option-packaging row. Those nonlisted rows are therefore forgotten rather than transported. □

Lemma 32.57 (Bool finite-data domain-codomain mirror boundary)**.** *In the finite-data bridge interface of Definition 32.85, the codomain-facing Bool read is the endpoint mirror of the domain-facing carrier read. Concretely, every public codomain read of the standard two-endpoint Bool surface is forced through one of the two rows* $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ *or* $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$*, and every such endpoint row is already a domain-side* $\mathsf{BoolHistoryCarrier}$ *row. Its observation, result, termination, and certificate data are therefore exactly the endpoint exactness row, endpoint coverage row, classifier endpoint exactness row, constructor separation row, semantic field row, and concrete semantic certificate row cited in Definition 32.85. There is no separate codomain carrier, codomain classifier, or codomain certificate row in this Bool finite-data bridge.*

Proof. Start with a public codomain read of the standard two-endpoint Bool surface. By Definition 32.70, the only public constructors of that surface are the two endpoint rows $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The readback theorem Definition 32.71 says that every public endpoint read is classified by exactly one of those rows, and that agreement of such rows is exactly mark agreement through msame.

Each of the two displayed endpoint rows is also a carrier row on the domain side. This is precisely the reverse direction of Definition 32.5: an endpoint witness $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ repacks as $\mathsf{BoolHistoryCarrier}(h)$. The agreement comparison is not a codomain-specific classifier, because Definition 32.4 identifies endpoint agreement with msame on the underlying marks, while

Definition 32.30 and Definition 32.31 say that the history classifier has exactly the two endpoint classes and cannot mix them.

It remains to identify the certificate side. The semantic fields in Definition 32.40 read source, pattern, and ledger all as $\mathsf{BoolHistoryCarrier}$, and Definition 32.42 supplies the public semantic certificate for that same carrier and classifier. Thus reversing attention from the domain read to the codomain read only renames the endpoint row being inspected. It does not create any extra carrier, classifier, ledger, result, termination, or certificate row beyond the finite packet already listed in Definition 32.85. □

Lemma 32.58 (Bool finite-data substrate bisimulation boundary)**.** *Let* Sub *be one of the substrate read names* $\mathsf{Rule110}$*,* $\mathsf{Cellular}$*,* Binary*,* $\mathsf{FKernel}$*, or raw* BHist*. Within the finite-data bridge interface of Definition 32.85, a consumer-facing read from* Sub *into* $\mathsf{Bool}^{\uparrow}$ *has only the following BEDC-local relation certificate. Its source is the named substrate read* Sub*, its target is the Bool finite-data packet, its direction is*

$$\mathsf{Sub}\text{-}\textit{read} \longrightarrow \textit{displayed endpoint readback rows} \longrightarrow \mathsf{Bool}^{\uparrow}\text{-}\textit{consumer},$$

and its relation kind is substrate mirror with bisimulation comparison on the two endpoint observations. Concretely, for histories $h, k : \mathsf{BHist}$ *the observable comparison must factor through displayed marks* $v, w : \mathsf{BMark}$ *as*

$$\mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \wedge \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \wedge \mathsf{msame}(v, w),$$

and this displayed relation is equivalent to $\mathsf{BoolHistoryClassifier}(h, k)$*. The retained certificate rows are precisely* $\mathsf{BoolHistoryCarrier}$*,* $\mathsf{BoolHistoryClassifier}$*,* $\mathsf{BoolHistoryCarrier}$ *as source, pattern, and ledger, and the semantic certificate of Definition 32.42. No accepted read may replace this mirror certificate by definitional equality between* Sub *and* $\mathsf{Bool}^{\uparrow}$*, by equality between a host substrate and the Bool carrier, or by an unlisted Rule110 state, cellular clock, cellular neighborhood, external binary trace, FKernel extension, FKernel continuation, FKernel package, FKernel gap, or raw* BHist *constructor coordinate.*

Proof. The finite-data interface is already enumerated in Definition 32.85. Its rows are endpoint bridge exactness, endpoint coverage, classifier endpoint exactness, constructor separation, the semantic fields, and the concrete semantic certificate. Thus a substrate-facing consumer has no independent row from which to read a Rule110 state, a cellular clock or neighborhood, an external binary trace, an FKernel extension, continuation, package, or gap coordinate, or a raw constructor identity for BHist.

The endpoint part of the certificate is forced by Definition 32.3: the only displayed target rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Endpoint coverage in Definition 32.5 says that every Bool history carrier read reaches one of those endpoint rows, and constructor separation prevents the two endpoint rows from being collapsed by the Bool classifier.

The comparison part is exactly the boundary in Definition 32.86. Its forward direction reads any $\mathsf{BoolHistoryClassifier}(h, k)$ comparison as source-qualified marks v, w, two endpoint readback rows $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ and $\mathsf{hsame}(k, \mathsf{BoolEndpoint}(w))$, and the mark comparison $\mathsf{msame}(v, w)$. Its reverse direction rebuilds the Bool history classifier from precisely those rows. Hence the displayed endpoint observations are preserved and reflected by the same finite comparison row; this is a bisimulation boundary on the local Bool observations, not a definitional equality boundary between substrates.

The direction and mirror reading are fixed by Definition 32.87: a codomain-facing Bool read is the endpoint mirror of the domain-facing carrier read. Therefore a substrate read can be consumed

only after it has been projected to the endpoint readback packet and then read through the Bool carrier and classifier rows. The semantic side contributes no extra substrate coordinate, because Definition 32.40 reads source, pattern, and ledger all as $\mathsf{BoolHistoryCarrier}$, and Definition 32.42 certifies that same carrier and classifier.

Finally, Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness, and the finite interface list contains none of the substrate-specific rows named in the statement. A receiving gate that re-reads the displayed rows must therefore reject any unlisted host or source coordinate. What remains is exactly the finite substrate mirror certificate above: source named by the incoming substrate read, target named by the Bool finite-data packet, direction through the displayed endpoint rows, and bisimulation comparison given by msame under the Bool history classifier. □

Theorem 32.59 (Bool-to-option constructor disjointness route)**.** *The* $\mathsf{Bool}^{\uparrow}$ *empty endpoint* e *and one-step endpoint* $\mathsf{E1}(\mathsf{e})$ *provide the branch discriminator consumed by the nullable and tagged constructor separation of* $\mathsf{Option}^{\uparrow}$ *in chapter 33. Any* $\mathsf{Option}^{\uparrow}$ *absent/present read that routes through these two displayed boolean endpoints must keep the absent branch on the empty row and the present branch on the one-step row; payload evidence cannot identify the two branches.*

Proof. Read the $\mathsf{Bool}^{\uparrow}$ endpoint rows through Definition 32.10. The carrier side has only the displayed $\mathsf{b0}$ and $\mathsf{b1}$ witnesses, and the classifier side is already fixed by $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$.

The $\mathsf{Option}^{\uparrow}$ nullable and tagged surfaces then consume those rows as the absent and present branch markers named in chapter 33. Since the discriminator has already been read from the boolean endpoint classifier, an optional payload row can only live under the visible present marker; it cannot change the marker that selected the branch.

Finally, mixed absent/present identification would require the boolean history classifier to relate e to $\mathsf{E1}(\mathsf{e})$. This is ruled out by Definition 32.31. Thus the route exports exactly the branch separation consumed by $\mathsf{Option}^{\uparrow}$, while leaving the payload carrier, payload classifier, and option packaging rows in the option chapter. □

The $\mathsf{Bool}^{\uparrow}$ truth rows feed the constructor-disjointness and finite-data routes of $\mathsf{Option}^{\uparrow}$ as the displayed absent/present discriminator cited in chapter 33. The dependency uses only the two endpoint histories, their local classifier, componentwise hsame transport, Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$ row, while payload carriers, nullable classifiers, and option equality remain owned by the $\mathsf{Option}^{\uparrow}$ packet.

Theorem 32.60 (Bool finite-data lattice route)**.** *The* $\mathsf{Bool}^{\uparrow}$ *endpoint classifier supplies the binary tag row consumed by the constructor-disjointness surface of* $\mathsf{Option}^{\uparrow}$ *in chapter 33 and by the tagged-branch separation surface of* $\mathsf{Sum}^{\uparrow}$ *in chapter 35. Any finite-data consumer that passes through both sibling chapters reads the same two boolean endpoint rows before it distinguishes absent/present option branches or left/right sum branches.*

Proof. Project the $\mathsf{Bool}^{\uparrow}$ endpoint surface through Definition 32.10. The only displayed tags are the empty endpoint and the one-step endpoint, and their classifier separation is fixed before any sibling payload row is inspected.

The $\mathsf{Option}^{\uparrow}$ consumer reads those tags as the absent/present discriminator, while the $\mathsf{Sum}^{\uparrow}$ consumer reads them as the visible branch separator. In both cases the sibling chapter supplies its own payload or branch carrier after the boolean tag row has already selected the finite-data face.

Thus the lattice route exports exactly the binary tag discipline supplied by $\mathsf{Bool}^{\uparrow}$. It does not add option payload equality, sum branch equality, or a host finite-type equality outside the displayed endpoint classifier. □

Theorem 32.61 (Bool finite-data outbound lattice)**.** *The* $\mathsf{Bool}^{\uparrow}$ *endpoint rows are consumed by* $\mathsf{Option}^{\uparrow}$ *in chapter 33,* $\mathsf{Sum}^{\uparrow}$ *in chapter 35,* $\mathsf{Prod}^{\uparrow}$ *in chapter 34, and* $\mathsf{List}^{\uparrow}$ *in chapter 36 only as finite branch or coordinate discriminators. Payload rows, pairing rows, coproduct branch rows, and spine-recursion rows remain owned by those sibling chapters, so the outbound lattice exports the visible binary endpoint surface without manufacturing a payload, product, sum, or list classifier inside* $\mathsf{Bool}^{\uparrow}$*.*

Proof. Project the endpoint exactness supplied by the $\mathsf{Bool}^{\uparrow}$ carrier. Its public rows distinguish the two finite endpoint cases and do not contain payload data, ordered-pair coordinates, coproduct contents, or recursive spine cells.

The four sibling consumers read this endpoint distinction in different positions. $\mathsf{Option}^{\uparrow}$ uses it as absent/present branch data, $\mathsf{Sum}^{\uparrow}$ as left/right branch data, $\mathsf{Prod}^{\uparrow}$ as finite coordinate-facing discriminator data, and $\mathsf{List}^{\uparrow}$ as branch information at the list spine boundary.

Each sibling then supplies its own carrier rows after the boolean discriminator has been read. Therefore the outbound lattice is a route from $\mathsf{Bool}^{\uparrow}$ to finite-data siblings, not a transfer of sibling-owned payload, pairing, coproduct, or recursion obligations back into the Bool chapter. □

Proposition 32.62 (Bool finite-data dependency lattice)**.** *The* $\mathsf{Bool}^{\uparrow}$ *endpoint discriminator rows are consumed as finite-data tags by the sibling constructors of chapter 33, chapter 35, chapter 34, and chapter 36, while equality and readback routes pass through the finite classifier boundary of chapter 131. Thus the Bool chapter exposes a concrete sibling lattice route: absent/present, left/right, component-presence, list-step, and equality-facing consumers all read the same two endpoint rows before their own payload rows are inspected.*

Theorem 32.63 (Bool finite-data sibling lattice integrity)**.** *The* $\mathsf{Bool}^{\uparrow}$ *finite-data surface supplies endpoint discriminator rows consumed by chapter 33, chapter 35, chapter 34, and chapter 36. The discriminator is the two-endpoint row already named by the* $\mathsf{Bool}^{\uparrow}$ *carrier:* $\mathsf{Option}^{\uparrow}$ *reads it as the absent/present branch,* $\mathsf{Sum}^{\uparrow}$ *reads it as the left/right branch,* $\mathsf{Prod}^{\uparrow}$ *reads it only as a coordinate-facing Boolean component when a product field is Boolean, and* $\mathsf{List}^{\uparrow}$ *reads it only as a finite spine or payload tag when a list cell is Boolean. The* $\mathsf{Bool}^{\uparrow}$ *chapter itself does not own option payload rows, product-coordinate pairing rows, coproduct-injection rows, or list-spine recursion rows.*

Proof. Project the $\mathsf{Bool}^{\uparrow}$ endpoint classifier through Definition 32.10. Its exported data are exactly the two endpoint discriminators and their classifier separation.

The sibling chapters then consume that row in distinct positions. $\mathsf{Option}^{\uparrow}$ and $\mathsf{Sum}^{\uparrow}$ use it as a branch discriminator, while $\mathsf{Prod}^{\uparrow}$ and $\mathsf{List}^{\uparrow}$ can only import it as a Boolean component row inside their own product or spine carriers.

Because payload, coordinate-pairing, coproduct-injection, and list-recursion rows are supplied by the respective sibling carriers, no accepted $\mathsf{Bool}^{\uparrow}$ packet can claim ownership of them. Thus the sibling lattice is visible precisely as a dependency of those finite-data constructors on the Bool endpoint discriminator. □

Theorem 32.64 (Bool finite-data sibling route)**.** *The* $\mathsf{Bool}^{\uparrow}$ *two-mark carrier is the finite endpoint consumed by the constructor-tag routes for* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*, and* $\mathsf{List}^{\uparrow}$*. Every such sibling read factors through the displayed true/false endpoint rows, the local* BHist *carrier, componentwise*

hsame *transport, visible* Cont *replay,* Pkg *provenance, and the* $\mathsf{NameCert}_{\mathsf{Bool}^\uparrow}$ *row before it can be used as an option tag, product projection flag, sum injection tag, or list constructor marker.*

In particular, the horizontal finite-data edge from chapter 32 to chapter 33 is a carrier route, not a host-level case split outside the BEDC packet.

Proof. Project the accepted $\mathsf{Bool}^\uparrow$ carrier. Its only data rows are the two displayed marks together with the transport, replay, provenance, and local naming rows.

The sibling constructors consume those marks as finite tags. The $\mathsf{Option}^\uparrow$ route reads one mark to distinguish absence from presence, while the $\mathsf{Prod}^\uparrow$, $\mathsf{Sum}^\uparrow$, and $\mathsf{List}^\uparrow$ routes use the same finite endpoint discipline for projection, injection, and constructor boundaries.

Since every read is routed through the same BHist, hsame, Cont, Pkg, and NameCert rows, no sibling obtains an external Boolean oracle or an unrecorded host case split. □

Theorem 32.65 ($\mathsf{Bool}^\uparrow$ option-product sibling lattice link)**.** *The* $\mathsf{Bool}^\uparrow$ *finite truth classifier supplies the binary discriminator row consumed by* $\mathsf{Option}^\uparrow$ *constructor disjointness in chapter 33 and by* $\mathsf{Prod}^\uparrow$ *pair-envelope selection in chapter 34. Both consumers read* $\mathsf{Bool}^\uparrow$ *only through the displayed two-endpoint truth classifier, componentwise transport, replay, provenance, and local naming rows. Neither consumer adds host booleans, external equality, option payload carriers, product coordinate carriers, or product-pair equality to the* $\mathsf{Bool}^\uparrow$ *packet.*

Proof. Project the finite truth surface already displayed by the $\mathsf{Bool}^\uparrow$ carrier. The visible data are the two endpoint histories, their classifier row, and the structural rows that transport and replay that finite distinction.

The $\mathsf{Option}^\uparrow$ consumer reads the two endpoints as absent and present branch markers before any payload row is inspected. The payload carrier and nullable payload classifier remain owned by the option chapter, so constructor disjointness imports only the binary discriminator.

The $\mathsf{Prod}^\uparrow$ consumer may read the same finite distinction only when a product envelope asks for a Boolean-facing coordinate or selector row. The pair carrier, coordinate carriers, and pair-envelope equality remain owned by the product chapter.

Thus both sibling routes consume the displayed $\mathsf{Bool}^\uparrow$ truth classifier without manufacturing host booleans or external equality. The exported dependency edge is the finite two-endpoint classifier and nothing stronger. □

Theorem 32.66 (Bool finite-truth sibling dependency route)**.** *The* $\mathsf{Bool}^\uparrow$ *finite-truth surface is consumed downstream only through displayed sibling NameCert routes. Equality endpoints and no-confusion reads cite the* $\mathsf{Eq}^\uparrow$ *boundary of chapter 131, while finite carrier scans cite the unary and list surfaces of chapter 28 and chapter 36. Thus a consumer may use Boolean truth rows as explicit endpoint data, but it may not replace those rows by ambient equality reflection, hidden finite enumeration, or list recursion outside the sibling certificates.*

Proof. Project the accepted $\mathsf{Bool}^\uparrow$ consumer route. Any equality or no-confusion read is routed through the displayed EqType sibling, so the endpoint comparison is read as a finite NameCert dependency rather than as ambient equality reflection.

For carrier scans, the only displayed finite traversal rows are the unary and list surfaces supplied by the Nat and List siblings. A downstream consumer therefore receives Boolean truth data through those certificates, not through an unrecorded enumeration of all Boolean values. □

The finite-data sibling lattice reads $\mathsf{Bool}^\uparrow$ as the two-tag truth classifier consumed by constructor interfaces, not as a global proposition oracle. In chapter 33, the empty and present branches use the two boolean endpoint rows as constructor tags; in chapter 34, component truth readback

uses the same finite classifier to decide which displayed component rows have been exposed. The dependence is therefore a constructor-tag and component-readback dependence on the two $\mathsf{Bool}^{\uparrow}$ endpoints, while $\mathsf{Option}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, and $\mathsf{List}^{\uparrow}$ still own their payload carriers, product rows, list spines, and local naming data.

Proposition 32.67 (Bool finite-data sibling readback)**.** *Every finite-data sibling read of* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*, or* $\mathsf{List}^{\uparrow}$ *that cites* $\mathsf{Bool}^{\uparrow}$ *uses only the two endpoint rows of* $\mathsf{Bool}^{\uparrow}$ *as constructor tags or component truth readback. It does not import a payload carrier, product classifier, list spine, or external truth predicate from the boolean chapter.*

Proof. Read the sibling dependency through the hub inputs. The option and sum surfaces consume the two endpoints as branch tags, the product surface consumes them as component-exposure readback, and the list surface consumes them only through finite constructor routing.

The payload, product, and spine data are then supplied by the receiving chapters. Since the Bool carrier contains only the two endpoint histories and their classifier, no additional payload or external truth predicate can be projected from the Bool row. □

Theorem 32.68 ($\mathsf{Bool}^{\uparrow}$ sibling dependency surface)**.** *The* $\mathsf{Bool}^{\uparrow}$ *finite-truth carrier is the L6 endpoint surface consumed by the adjacent finite-data routes for* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*, and* $\mathsf{CantorSpace}^{\uparrow}$*. The* $\mathsf{Option}^{\uparrow}$ *route of chapter 33 reads the two endpoints as constructor discriminators, the* $\mathsf{Sum}^{\uparrow}$ *and* $\mathsf{Prod}^{\uparrow}$ *routes read them as finite branch and component selectors, the* $\mathsf{List}^{\uparrow}$ *route reads them as finite traversal endpoints, and the* $\mathsf{CantorSpace}^{\uparrow}$ *route reads them as binary coordinate endpoints. Each consumer receives only the displayed Boolean endpoint rows, endpoint classifier,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

No sibling consumer receives host booleans, external equality reflection, hidden enumeration, product-pair equality, option payload data, list-recursion data, or Cantor-space compactness from the $\mathsf{Bool}^{\uparrow}$ *packet.*

Proof. Start with the finite-truth route already exposed by the $\mathsf{Bool}^{\uparrow}$ carrier and its local sibling lattice. The readable rows are the two endpoint histories, the endpoint classifier, componentwise transport, displayed replay, provenance, and local naming.

The option route consumes the endpoints only as absent and present branch discriminators, as in chapter 33. The sum, product, and list routes consume the same finite distinction as branch, component, and traversal data, while their payload, pair, and recursion certificates remain owned by their own packets.

For the Cantor-space endpoint route, the Boolean row supplies only the binary coordinate endpoint. Compactness, product topology, and infinite-choice data are not coordinates of the Boolean carrier, so they cannot be exported through this dependency surface. □

Theorem 32.69 (Bool finite-data product sibling route)**.** *The* $\mathsf{Bool}^{\uparrow}$ *finite truth rows are consumed by the* $\mathsf{Prod}^{\uparrow}$ *paired component classifier through chapter 34. A product consumer may read a Boolean component only as one of the two displayed endpoint rows named by the Bool certificate, then package it with the other product coordinate through the product sibling route.*

Proof. Project the Bool carrier to its two finite truth endpoints. The product sibling route reads those endpoints as component classifier data, while the paired coordinate and product envelope remain supplied by chapter 34.

Thus the Boolean row contributes only its displayed finite endpoint and its local naming evidence. It cannot replace the product classifier, introduce an ambient Boolean algebra, or hide product-coordinate data outside the sibling certificate. □

The $\mathsf{Bool}^{\uparrow}$ endpoint surface supplies the two visible rows consumed by the L6 finite-data constructor lattice. The nullable branch of chapter 33, the tagged alternatives of chapter 35, the paired constructor route of chapter 34, and the finite spine classifier of chapter 36 all read those endpoints as displayed discriminator rows. The route exports only the two endpoint marks and their history classifier, leaving payload carriers, product coordinates, coproduct injections, and list recursion to the corresponding sibling packets.

Definition 32.70 (Bool public endpoint constructors)**.** *The public false constructor is* $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$*, and the public true constructor is* $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$*. Public boolean operations are finite endpoint tables whose input and output rows are drawn only from these two constructor rows. No row in this public surface is a host proposition, a decidable predicate, or an eliminator.*

Lean4 (definition) `BEDC.Derived.BoolUp.BoolPublicEndpointConstructors`

Theorem 32.71 (Bool public constructor case readback)**.** *Every public Bool endpoint read is classified by exactly one of the displayed false and true endpoint rows of Definition 32.70. Two public constructor readbacks agree exactly when their underlying marks are related by* msame*.*

Proof. Apply Definition 32.5 to the public read. It supplies a displayed mark $v : \mathsf{BMark}$ and a witness that the read history is the endpoint $\mathsf{BoolEndpoint}(v)$.

Case analysis on v gives the two public rows $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Theorem 32.6 gives uniqueness of the readback mark, and Definition 32.4 identifies agreement of endpoint rows with msame of the underlying marks. □

Theorem 32.72 (Bool public surface no-hidden leakage)**.** *A consumer of the public Bool surface can read only the two constructor endpoint rows, endpoint-table operation rows over those constructors, the mark and history classifier rows, and the semantic certificate ledger. It cannot read a host proposition, a decidable predicate, an eliminator, or truth-table completeness.*

Proof. The public constructors are exactly the two endpoint rows in Definition 32.70. The endpoint coverage and constructor case readback statements show that public reads return one of these rows, with agreement controlled by msame and endpoint exactness rather than by host logic.

The semantic certificate in Definition 32.42 records the carrier, classifier, and ledger discipline. Since the displayed public rows contain no host proposition, decidable predicate, eliminator, or truth-table-completeness row, those objects are not available to a surface consumer. □

The endpoint coverage, endpoint bridge classifier completeness, and endpoint readback theorems justify the Bool public-object readback and bridge promised by the roadmap template; they are not internal fields of the SemanticNameCert constructor. $\mathsf{Option}^{\uparrow}$consumes this two-row endpoint discipline as its absent/present tag surface in chapter 33.

Corollary 32.73 ($\mathsf{Bool}^{\uparrow}$ certificate scope)**.** *Theorem 32.39 cites the checked core history* NameCert *for* BoolHistoryCarrier *and* BoolHistoryClassifier*, namely the BEDC.Derived.BoolUp Lean target* bool_history_name_certificate*. The public-object* SemanticNameCert *scope is checked separately at Theorem 32.42 by the BEDC.Derived.BoolUp Lean target* bool_history_semantic_name_certificate*. Thus the core certificate, the mark-level ledger definition, the generic carrier-as-ledger implication, and the closed semantic certificate occupy separate roles. In each case* $\mathsf{Bool}^{\uparrow}$ *is derived from mark and history sameness, and is not primitive.*

Lemma 32.74 (Bool public endpoint forgetful projection boundary)**.** *The public constructor surface of Definition 32.70 admits a BEDC-native forgetful projection. Start with a public constructor*

readback row and erase the constructor-row name, endpoint-table name, and public-surface annotation while retaining the displayed endpoint history. The retained target data are exactly

$\mathsf{BoolHistoryCarrier}$, $\mathsf{BoolHistoryClassifier}$, $\mathsf{BoolHistoryCarrier}$ *as source, pattern, and ledger,* $\mathsf{NameCert}(\mathsf{BoolHis}$

together with the public semantic certificate

$\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier})$.

More concretely, each retained public read has a mark $v : \mathsf{BMark}$ *and an endpoint witness* $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ *unique up to* msame*. For two retained reads* h, k *with endpoint witnesses through* $v, w : \mathsf{BMark}$*, the projected classifier row* $\mathsf{BoolHistoryClassifier}(h, k)$ *holds exactly when* $\mathsf{msame}(v, w)$ *holds. The projection retains no constructor-row name, endpoint-table name, host proposition, decidable predicate, eliminator, truth-table-completeness datum, payload carrier, payload classifier, or option-packaging row.*

Proof. The public constructor rows are exactly the two endpoint rows displayed in Definition 32.70. Applying Definition 32.71 to a public read supplies a mark $v : \mathsf{BMark}$ and an endpoint witness $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$, and Definition 32.6 makes the displayed mark unique up to msame.

The retained carrier coordinate is forced by Definition 32.5: an endpoint witness against $\mathsf{BoolEndpoint}(v)$ repacks as $\mathsf{BoolHistoryCarrier}(h)$, and every Bool carrier row is covered by such an endpoint witness. Thus erasing the public constructor name leaves a history in the same target carrier rather than a separate codomain carrier.

For the classifier coordinate, take two retained reads h, k with endpoint witnesses through v, w. If $\mathsf{msame}(v, w)$ holds, the reverse direction of Definition 32.12 rebuilds $\mathsf{BoolHistoryClassifier}(h, k)$ from the two endpoint witnesses and the mark comparison. Conversely, if $\mathsf{BoolHistoryClassifier}(h, k)$ holds, Definition 32.12 supplies endpoint witnesses through marks v', w' with $\mathsf{msame}(v', w')$. The readback determinism theorem compares v with v' and w with w', and transitivity of mark sameness gives $\mathsf{msame}(v, w)$. Hence the classifier retained by the projection is exactly the mark-level agreement already visible in the constructor readback.

The certificate coordinates are fixed by Definition 32.40, Definition 32.39, and Definition 32.42. Source, pattern, and ledger are all the same predicate $\mathsf{BoolHistoryCarrier}$, the core name certificate is $\mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier})$, and the public semantic certificate is the displayed carrier-self semantic certificate. Finally, Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness, while the Bool public surface contains no payload carrier, payload classifier, or option-packaging coordinate. Those named rows are therefore erased rather than transported by the forgetful projection. □

Lemma 32.75 (Bool public endpoint constructor classifier alignment boundary)**.** *For histories* $h, k : \mathsf{BHist}$*, the public Bool endpoint surface aligns its classifier boundary exactly with the two displayed constructor endpoint rows:*

$$\mathsf{BoolHistoryClassifier}(h, k)$$
$$\iff \quad \exists v, w : \mathsf{BMark},\ e_v, e_w : \mathsf{BHist},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \big((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge$$

Under this equivalence, the transported public rows are exactly the two endpoint readback rows against the endpoint histories e_v *and* e_w *determined by* $\mathsf{BoolEndpoint}(v)$ *and* $\mathsf{BoolEndpoint}(w)$*, together with the mark-comparison row* $\mathsf{msame}(v, w)$*. The retained certificate rows are the carrier, classifier, and semantic ledger rows named by the Bool semantic certificate. Rows for host propositions, decidable predicates, eliminators, and truth-table completeness are forgotten by the public endpoint classifier boundary.*

Proof. The displayed equivalence is Definition 32.13 read through the explicit endpoint-history witnesses supplied by Definition 32.70. Its forward direction says that a classifier comparison between h and k exposes only source-qualified marks v, w, endpoint histories e_v, e_w with the displayed constructor equations, the two endpoint readback rows $\mathsf{hsame}(h, e_v)$ and $\mathsf{hsame}(k, e_w)$, and the mark-comparison row $\mathsf{msame}(v, w)$. Its reverse direction says that precisely those rows rebuild the Bool history classifier comparison, so no further classifier coordinate is being transported.

By Definition 32.70, the public constructors are only $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The case readback theorem Definition 32.71 therefore identifies every public endpoint read with exactly one of those two rows, and Definition 32.6 makes the chosen mark unique up to msame. The endpoint bridge exactness of Definition 32.4 then makes the row comparison a mark comparison rather than a host-logical comparison.

The certificate side contributes no extra public row. The semantic certificate in Definition 32.42 records the Bool carrier, classifier, and ledger discipline over the same history carrier. Finally, Definition 32.72 states that the public surface does not expose a host proposition, decidable predicate, eliminator, or truth-table completeness row. Hence the boundary transports exactly the two constructor readback rows and their msame classifier comparison, retains only the semantic certificate ledger rows, and forgets the non-public rows listed above. □

Lemma 32.76 (Bool public constructor target-source mirror boundary)**.** *For the public Bool constructor surface of Definition 32.70, the target-facing constructor read and the source-facing carrier read are the same finite endpoint packet viewed in opposite directions. Concretely, a public target read through either displayed constructor row* $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ *or* $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$ *has a source mirror consisting exactly of the same mark* $v : \mathsf{BMark}$*, an endpoint history witness* $e_v : \mathsf{BHist}$ *satisfying* $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$*, the same readback observation row* $\mathsf{hsame}(h, e_v)$*, the same classifier comparison row* $\mathsf{msame}(v, w)$ *when another constructor read is compared through a mark* $w : \mathsf{BMark}$ *and endpoint history witness* $e_w : \mathsf{BHist}$ *satisfying* $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$*, and the same semantic certificate rows*

$$\mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}), \quad \mathsf{SemanticNameC}$$

Thus the source side is obtained by renaming the displayed target endpoint packet back to its carrier readback packet. The boundary is negative for any larger mirror: there is no separate source-side constructor carrier, result row, termination row, host proposition row, decidable predicate row, eliminator row, truth-table-completeness row, payload carrier row, payload classifier row, or option-packaging row.

Proof. Start with a public target read. By Definition 32.70, the only target constructor rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The case readback theorem Definition 32.71 therefore supplies a displayed mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ with $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$, and identifies the read with the endpoint readback row $\mathsf{hsame}(h, e_v)$. Its agreement clause, together with Definition 32.6 and Definition 32.4, says that any comparison between two such reads is exactly the mark comparison $\mathsf{msame}(v, w)$.

The source mirror is forced by endpoint coverage. The reverse direction of Definition 32.5 repacks the readback row $\mathsf{hsame}(h, e_v)$, for an endpoint witness e_v satisfying $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$, as the source carrier row $\mathsf{BoolHistoryCarrier}(h)$, while the forward direction says that every source carrier row used by this Bool surface is covered by such an endpoint readback. Hence the target row and source row do not form two independent constructors; they are the same displayed history endpoint read as target endpoint or as source carrier.

For compared reads h, k, Definition 32.12 identifies the source-side classifier row $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint readbacks through marks v, w, endpoint history witnesses $e_v, e_w : \mathsf{BHist}$ satisfying $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$ and $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$, and the same comparison row $\mathsf{msame}(v, w)$. The source-qualified form in Definition 32.13 adds only the already displayed source qualification of those two finite marks, so it does not introduce a further target or source coordinate.

The certificate rows are exactly the rows already named by Definition 32.40, Definition 32.39, and Definition 32.42: source, pattern, and ledger are all read as $\mathsf{BoolHistoryCarrier}$, and the classifier is $\mathsf{BoolHistoryClassifier}$. Finally, Definition 32.72, Definition 32.74, and Definition 32.75 rule out the larger mirror coordinates listed in the statement. The target-source mirror therefore closes exactly at the finite endpoint packet and its Bool semantic certificate rows. □

Lemma 32.77 (Bool public endpoint bridge cut-elimination boundary)**.** *The public Bool endpoint constructors of Definition 32.70 admit a bridge cut-elimination boundary over the displayed local rows. For a final consumer read of the public Bool surface, every route row factors through the finite packet*

$$\begin{gathered}
\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}), \\
v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist}, \\
((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))), \\
((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))), \\
\mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w), \\
\mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}), \\
\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).
\end{gathered}$$

The two displayed constructor endpoint rows, the endpoint history readback rows, the mark-comparison row, and the carrier, classifier, core certificate, and semantic certificate rows are noneliminable at this boundary: removing any one of them loses either the public endpoint choice, the readback into a history, the classifier comparison, or the certificate surface consumed by the final Bool object. Every other named coordinate appearing in the local public-surface discussion is eliminated or rejected: constructor-row names, endpoint-table names, and public-surface annotations are projected away; host propositions, decidable predicates, eliminators, truth-table-completeness data, payload carriers, payload classifiers, option-packaging rows, separate source-side constructor carriers, result rows, termination rows, and continuation rows are not consumer-visible rows of this Bool endpoint bridge.

Proof. The finite endpoint rows are exactly the two rows displayed in Definition 32.70. Thus any public constructor read begins at either $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ or $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$, and no third constructor, host predicate, or eliminator row is available at the source of the route. Applying Definition 32.71 to the read supplies a mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ with $((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e})))$ and $\mathsf{hsame}(h, e_v)$; applying it to a second read supplies $w : \mathsf{BMark}$ and an endpoint history witness $e_w : \mathsf{BHist}$ with $((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e})))$ and $\mathsf{hsame}(k, e_w)$. The uniqueness part is controlled by Definition 32.6, so the chosen endpoint mark cannot be replaced by an unlisted host or source coordinate.

The comparison row is also forced. By Definition 32.4, comparison of the two endpoint rows is exactly comparison of their marks by $\mathsf{msame}(v, w)$. Equivalently, Definition 32.12 identifies the history classifier row $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint history readbacks and the same mark-comparison row. Hence a bridge consumer cannot keep an independent comparison

route after those readback rows have been exposed; the classifier route has already cut through the endpoint packet.

The carrier and certificate side is fixed by the already displayed projection boundary. By Definition 32.5, the endpoint readback through e_v repacks as the carrier row $\mathsf{BoolHistoryCarrier}(h)$, and every carrier row used here is covered by such an endpoint readback. By Definition 32.74, erasing constructor-row names, endpoint-table names, and public-surface annotations leaves exactly the carrier, classifier, carrier-as-source-pattern-ledger, core $\mathsf{NameCert}$, and public semantic certificate rows. By Definition 32.40, Definition 32.39, and Definition 32.42, these rows are the Bool history carrier, the Bool history classifier, and the carrier-self semantic certificate displayed in the statement.

The remaining possible cuts are negative. Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness from the public consumer surface. Definition 32.75 shows that the classifier boundary transports only the two endpoint readback rows and their msame comparison, while Definition 32.76 shows that target-facing and source-facing reads are the same finite endpoint packet rather than separate constructor carriers, result rows, termination rows, or continuation rows. Since the local chapter displays no additional host, source, handoff, continuation, payload, or option-packaging coordinate, any such coordinate is rejected as unlisted rather than retained by the bridge.

Consequently the final consumer-visible surface is precisely the finite packet named in the statement. The endpoint rows, readback rows, mark-comparison row, carrier and classifier rows, and the two certificate rows are noneliminable because they are the rows used to reconstruct the endpoint choice, the history read, the classifier comparison, and the public Bool certificate. All other local route names are either projected away by the forgetful boundary or excluded by the public-surface and mirror-boundary theorems. □

Lemma 32.78 (Bool public endpoint equivalence-completion boundary)**.** *The public Bool endpoint constructor surface of Definition 32.70 has a complete equivalence boundary over its displayed local classifier and readback rows. For histories $h, k : \mathsf{BHist}$, the carrier and classifier two-sided part is exactly*

$$\begin{aligned}
\mathsf{BoolHistoryCarrier}(h) &\iff \exists v : \mathsf{BMark},\ \exists e_v : \mathsf{BHist},\ ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))) \wedge \mathsf{hsame}(h \\
\mathsf{BoolHistoryClassifier}(h, k) &\iff \exists v, w : \mathsf{BMark},\ \exists e_v, e_w : \mathsf{BHist},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge ((v =
\end{aligned}$$

For marks $v, w : \mathsf{BMark}$ and endpoint-history witnesses $e_v, e_w : \mathsf{BHist}$ satisfying the displayed constructor equations

$$\begin{aligned}
&((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))),\\
&((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))),
\end{aligned}$$

the endpoint-agreement part is exactly

$$\mathsf{msame}(v, w) \iff \mathsf{hsame}(e_v, e_w).$$

The endpoint witnesses are precisely the same visible endpoint rows as the public false and true constructor rows: $v = \mathsf{b0}$ carries $e_v = \mathsf{e}$, and $v = \mathsf{b1}$ carries $e_v = \mathsf{E1}(\mathsf{e})$, with the same alternatives for w, e_w. Thus any displayed forward carrier, classifier, endpoint-agreement, or constructor-readback implication in the local chapter has its reverse implication on the same finite rows.

The remaining displayed directions in the local public-surface discussion are one-way interpretation boundaries. Erasing constructor-row names, endpoint-table names, and public-surface annotations maps a public constructor read to the carrier, classifier, core $\mathsf{NameCert}$, and public semantic certificate rows, but those retained rows do not reconstruct the erased names or annotations.

The no-hidden-leakage row is also one-way: it excludes host propositions, decidable predicates, eliminators, truth-table-completeness data, payload carriers, payload classifiers, option-packaging rows, separate source-side constructor carriers, result rows, termination rows, and continuation rows from the public endpoint surface, and no reverse implication is claimed from their absence to any additional public constructor structure.

Proof. The carrier row is already two-sided by Definition 32.5. Its forward direction reads a Bool carrier history as one of the two endpoint histories, witnessed by some $e_v : \mathsf{BHist}$ satisfying $(v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))$; its reverse direction repacks the same endpoint readback witness as $\mathsf{BoolHistoryCarrier}(h)$. Since Definition 32.70 displays only the two constructor histories e and $\mathsf{E1}(\mathsf{e})$, this carrier equivalence has no third public endpoint row.

The classifier row is two-sided by Definition 32.13. The forward implication exposes source-qualified marks v, w, endpoint-history witnesses e_v, e_w with their constructor equations, and the mark comparison $\mathsf{msame}(v, w)$. The reverse implication rebuilds $\mathsf{BoolHistoryClassifier}(h, k)$ from exactly those rows. The more explicit constructor-row version is Definition 32.75, which identifies the endpoint witnesses through the two displayed constructor histories e and $\mathsf{E1}(\mathsf{e})$.

Endpoint agreement is two-sided by Definition 32.4. The case-readback theorem Definition 32.71 then says that a public constructor read is classified by exactly one of the displayed false and true endpoint rows, represented by an endpoint-history witness e_v with its constructor equation, while Definition 32.6 makes the chosen mark unique up to msame. Hence the visible constructor readback direction is completed by the same endpoint-history witnesses, constructor equations, and mark-comparison rows, not by a host proposition or by an external equivalence carrier.

It remains to identify the directions that are not equivalences. The forgetful projection of Definition 32.74 deliberately erases constructor-row names, endpoint-table names, and public-surface annotations. Its target rows are the retained carrier, classifier, $\mathsf{NameCert}$, and public semantic certificate rows fixed by Definition 32.40, Definition 32.39, and Definition 32.42; those rows remember the endpoint-history witness and classifier comparison, not the erased presentation names. This is therefore a one-way interpretation boundary.

The negative rows are one-way for the same local reason. By Definition 32.72, the public surface exposes no host proposition, decidable predicate, eliminator, or truth-table-completeness row. By Definition 32.76, the target-facing and source-facing reads are the same finite endpoint packet rather than separate constructor carriers, result rows, termination rows, or continuation rows. By Definition 32.77, every final consumer route factors through the finite endpoint packet and the Bool semantic certificate rows, while payload and option-packaging coordinates are rejected as unlisted. Thus the local chapter has no unmatched forward classifier or readback implication: the carrier, classifier, endpoint-agreement, and constructor-readback rows are completed by reverse implications on the same visible rows, and the projection and exclusion rows are recorded as one-way interpretation boundaries. □

Lemma 32.79 (Bool public endpoint substrate bisimulation boundary)**.** *The public Bool endpoint constructors of Definition 32.70 admit a substrate relation boundary, not a definitional-equality boundary. The BEDC-local relation certificate has source rows only from the displayed finite end-*

point packet

$$\begin{gathered}\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}),\\ v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist},\\ ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))), \quad ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))),\\ \mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w),\\ \mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}),\\ \mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).\end{gathered}$$

Its forward leg sends a public endpoint read through $\mathsf{hsame}(h, e_v)$ *to the carrier row* $\mathsf{BoolHistoryCarrier}(h)$*. Its reverse leg reads every carrier row used by the Bool public surface back through one of the two displayed endpoint rows. Its comparison leg is the reflected equivalence*

$$\begin{aligned}\mathsf{BoolHistoryClassifier}(h, k) \quad \Longleftrightarrow \quad \exists v, w : \mathsf{BMark},\ e_v, e_w : \mathsf{BHist},\ & ((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))) \wedge\\ & ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))) \wedge\\ & \mathsf{hsame}(h, e_v) \wedge \mathsf{hsame}(k, e_w) \wedge \mathsf{msame}(v, w).\end{aligned}$$

Thus a BHist*-facing reading of the public Bool endpoint surface is a substrate mirror, and any consumer that names a* Rule110*, cellular, binary, or* FKernel *substrate must state its use as a bisimulation or row-embedding relation whose Bool side is exactly the packet above. No such substrate name is available here as a definitional equality with the Bool public constructors. Any host coordinate, source coordinate, transition coordinate, external substrate row, result row, termination row, continuation row, payload row, option-packaging row, or equality row not listed in the packet is rejected at this boundary rather than silently identified with the Bool public surface.*

Proof. The local constructor source is exhausted by Definition 32.70: the only public endpoint rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Applying Definition 32.71 to a public read supplies a displayed mark $v : \mathsf{BMark}$ and an endpoint history witness $e_v : \mathsf{BHist}$ in one of those two cases, together with the readback row $\mathsf{hsame}(h, e_v)$. The determinacy clause of Definition 32.6 prevents replacement of that mark by an unlisted host or substrate coordinate.

The carrier leg is exactly a mirror leg. By Definition 32.5, the endpoint readback row repacks as $\mathsf{BoolHistoryCarrier}(h)$, and every carrier row used by this Bool surface is covered by such an endpoint readback. Hence the relation runs in both directions between the public endpoint packet and the BHist carrier readback packet, but the two sides are not introduced as a single definitional object; they are the same displayed history read viewed through endpoint and carrier roles.

The comparison leg is the bisimulation clause. For two reads h, k with endpoint witnesses through v, w, Definition 32.12 identifies $\mathsf{BoolHistoryClassifier}(h, k)$ with the two endpoint readbacks and the mark comparison $\mathsf{msame}(v, w)$. The explicit endpoint-row form is the classifier alignment boundary of Definition 32.75. Thus classifier comparison is both preserved and reflected by the endpoint packet: endpoint mark agreement rebuilds the history classifier, and a history-classifier comparison exposes endpoint witnesses and mark agreement.

The certificate rows are fixed by Definition 32.74 and Definition 32.77. After constructor-row names, endpoint-table names, and public-surface annotations are erased, the retained rows are exactly the carrier, classifier, carrier-as-source-pattern-ledger, core NameCert, and public semantic certificate rows displayed in the statement. The target-source mirror theorem Definition 32.76 further says that the source-facing and target-facing reads are opposite views of the same finite endpoint packet, not separate constructor carriers.

Finally, the negative part follows from the same row exhaustion. Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness from the public surface, while Definition 32.77 rejects unlisted source, result, termination, continuation, payload, and option-packaging coordinates. The chapter displays no Rule110, cellular, binary, or FKernel substrate row inside the Bool public endpoint constructors. Therefore any later consumer using one of those substrate names must provide its own target-side rows and relate them to the Bool packet by bisimulation, row embedding, or substrate mirror. The BEDC-local Bool boundary supplies no definitional equality from the public endpoint constructors to such a substrate. □

Theorem 32.80 (Bool finite-data exit route)**.** *The public* $\mathsf{Bool}^{\uparrow}$ *finite-data route consists exactly of the two endpoint constructor rows of Definition 32.70, the constructor case readback of Definition 32.71, the no-hidden-leakage boundary of Definition 32.72, and the semantic certificate scope of Definition 32.73. Any* $\mathsf{Option}^{\uparrow}$ *absent/present tag read,* $\mathsf{Prod}^{\uparrow}$ *sibling-lattice read, or* $\mathsf{List}^{\uparrow}$ *finite-data sibling route that consumes* $\mathsf{Bool}^{\uparrow}$ *passes through those rows and receives no host proposition, eliminator, decidable predicate, or truth-table completeness.*

Proof. The two constructor rows are the public endpoints displayed in Definition 32.70. The case readback theorem then says that every public endpoint read is one of those rows, with agreement measured by msame rather than by an external Boolean predicate.

The no-hidden-leakage theorem removes the extra host-logic surfaces from the public exit, while Definition 32.73 identifies the checked core and semantic certificate roles that remain available to downstream finite data consumers.

The consumer list is exactly the existing finite-data route through Definition 32.59, Definition 34.110, and Definition 36.214. These routes use the same endpoint rows as absent/present tags, product sibling coordinates, and list finite-data siblings; none adds a primitive Boolean object or a host truth-table principle. □

Lemma 32.81 (Bool public endpoint downstream coverage)**.** *Every accepted local consumer of the public Bool endpoint constructors is covered by the finite alternatives and ledger rows already displayed in this file. More precisely, after the consumer is reduced to its public endpoint readback and Bool certificate scope, its visible rows lie in the finite packet*

$$\begin{gathered}
\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}, \quad \mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e}), \\
v, w : \mathsf{BMark}, \quad e_v, e_w : \mathsf{BHist}, \\
((v = \mathsf{b0} \wedge e_v = \mathsf{e}) \vee (v = \mathsf{b1} \wedge e_v = \mathsf{E1}(\mathsf{e}))), \quad ((w = \mathsf{b0} \wedge e_w = \mathsf{e}) \vee (w = \mathsf{b1} \wedge e_w = \mathsf{E1}(\mathsf{e}))), \\
\mathsf{hsame}(h, e_v), \quad \mathsf{hsame}(k, e_w), \quad \mathsf{msame}(v, w), \\
\mathsf{BoolHistoryCarrier}, \quad \mathsf{BoolHistoryClassifier}, \quad \mathsf{NameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}), \\
\mathsf{SemanticNameCert}(\mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryCarrier}, \mathsf{BoolHistoryClassifier}).
\end{gathered}$$

The alternatives outside this packet remain outside the Bool BEDC public surface: no host proposition, decidable predicate, eliminator, truth-table-completeness row, payload carrier, payload classifier, option-packaging row, separate source-side constructor carrier, result row, termination row, continuation row, external substrate row, or definitional-equality row is accepted as a downstream Bool endpoint constructor row.

Proof. Let a local downstream consumer be accepted by the public Bool endpoint constructor surface. By Definition 32.70, the only public constructor alternatives available at the entry point are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Applying Definition 32.71 to the consumer read supplies a mark $v : \mathsf{BMark}$ and an endpoint-history witness $e_v : \mathsf{BHist}$ in one of those two

displayed cases, together with the readback row $\mathsf{hsame}(h, e_v)$. If the consumer compares two accepted reads, the same argument supplies $w : \mathsf{BMark}$, $e_w : \mathsf{BHist}$, and $\mathsf{hsame}(k, e_w)$.

The equivalence-completion boundary of Definition 32.78, read with the explicit transport theorems Definition 32.16 and Definition 32.34, then exhausts the possible downstream classifier data. For the carrier leg, Definition 32.5 gives the two endpoint witnesses displayed above, and Definition 32.16 is the transport citation that carries that endpoint witness along the readback row $\mathsf{hsame}(h, e_v)$ to the usable carrier row $\mathsf{BoolHistoryCarrier}(h)$. For the classifier leg, Definition 32.13 gives source-qualified marks, the two displayed constructor equations for e_v and e_w, the two endpoint readback rows, and the mark comparison $\mathsf{msame}(v, w)$; Definition 32.34 is the transport citation for moving this classifier row along the endpoint readbacks. For the endpoint-agreement leg, Definition 32.4 reduces comparison of endpoint histories to the same mark comparison, with the endpoint-readback movement again governed by Definition 32.34. Hence every accepted carrier, classifier, or agreement row used downstream is already one of the rows in the finite packet above.

The certificate scope contributes only the named Bool certificate rows. By Definition 32.73, the core NameCert over BoolHistoryCarrier and BoolHistoryClassifier and the public SemanticNameCert over the same carrier, pattern, ledger, and classifier roles are separate checked scopes, with $\mathsf{Bool}^{\uparrow}$ derived from mark and history sameness rather than primitive host truth. Thus a downstream consumer receives the carrier, classifier, core certificate, and public semantic certificate rows listed in the packet, and no additional Boolean object is introduced by the certificate citation.

The exclusion clause is the one-way side of the same local boundary. Definition 32.78 records that forgetful projection and no-hidden-leakage directions do not reconstruct erased constructor-row names, endpoint-table names, or public-surface annotations, and do not turn absence of host data into further public constructor structure. Together with Definition 32.72, Definition 32.77, and Definition 32.79, this rules out host propositions, decidable predicates, eliminators, truth-table completeness, payload and option coordinates, separate source, result, termination, continuation, substrate, and definitional-equality rows. Therefore the accepted local downstream consumer is exhausted by the finite alternatives, readback rows, classifier row, and certificate ledger rows displayed in this file, while every excluded alternative remains outside the Bool BEDC public endpoint packet. □

Theorem 32.82 (Bool CantorSpace endpoint route)**.** *Every displayed* $\mathsf{Bool}^{\uparrow}$ *endpoint row consumed by a* $\mathsf{CantorSpace}^{\uparrow}$ *finite prefix cylinder remains a* $\mathsf{Bool}^{\uparrow}$ *two-endpoint* NameCert *row after* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{List}^{\uparrow}$ *prefix transport. The route exports only the empty and one-step endpoint rows, their history classifier, the transported prefix position, and the local package ledger. It does not introduce a host Boolean, quotient stream equality, branch choice, or compactness claim.*

Proof. Start from the public endpoint surface of $\mathsf{Bool}^{\uparrow}$, where Definition 32.70 displays exactly the two rows e and $\mathsf{E1}(\mathsf{e})$. The case readback theorem Definition 32.71 fixes every public endpoint read to one of these rows, with agreement measured by msame and the endpoint classifier rather than by a host Boolean predicate.

The Cantor prefix consumer in chapter 1475 reads Boolean observations only through finite prefix cylinders. Its StreamName schedule is the window discipline of chapter 126, and its finite prefix spine is a list-indexed transport of displayed endpoint rows. Transport along that schedule changes the inspected position and the finite prefix ledger, not the two endpoint constructors supplied by $\mathsf{Bool}^{\uparrow}$.

Thus the transported row remains a $\mathsf{Bool}^{\uparrow}$ endpoint row after the CantorSpace prefix and StreamName/List bookkeeping have been attached. The no-hidden-leakage theorem Definition 32.72

excludes host Boolean structure, while the CantorSpace chapter supplies only the finite-prefix consumer route and not quotient streams, branch choices, or compactness. □

The $\mathsf{CantorSpace}^{\uparrow}$ finite-prefix surface consumes $\mathsf{Bool}^{\uparrow}$ as the finite endpoint ledger for each $\mathsf{StreamName}^{\uparrow}$ prefix window. The endpoint rows read through Definition 32.70, Definition 32.71, and Definition 32.82 supply only the empty and one-step Boolean histories attached to a displayed prefix coordinate.

Thus the dependency into chapter 1475 is a finite endpoint dependency: $\mathsf{CantorSpace}^{\uparrow}$ owns the prefix cylinder, stream schedule, list-indexed window, and local NameCert route, while $\mathsf{Bool}^{\uparrow}$ supplies the two visible endpoint rows consumed at each inspected coordinate. The route does not export host Boolean predicates, quotient stream equality, branch selection, fan-theorem compactness, open-cover compactness, or ambient product completeness.

Theorem 32.83 ($\mathsf{Bool}^{\uparrow}$ sibling carrier boundary)**.** *The two endpoint histories exported by* $\mathsf{Bool}^{\uparrow}$ *are consumed by* $\mathsf{Option}^{\uparrow}$ *as the nullable discriminator of chapter 33 and by* $\mathsf{Sum}^{\uparrow}$ *as the finite branch discriminator of chapter 35. The* $\mathsf{Bool}^{\uparrow}$ *packet exports only the two endpoint rows and their displayed classifier boundary; it exports no option payload carrier, sum payload carrier, nullable payload classifier, or tagged-sum classifier.*

Proof. Project the public endpoint surface of $\mathsf{Bool}^{\uparrow}$. Its exported rows are exactly the empty endpoint and the one-step endpoint, together with the classifier rows that distinguish those two histories.

The $\mathsf{Option}^{\uparrow}$ consumer reads the same two histories as absent and present tags before any payload row is inspected. The $\mathsf{Sum}^{\uparrow}$ consumer reads them as left and right branch tags before any branch payload row is inspected.

Because payload carriers and payload classifiers are supplied by the option and sum sibling chapters, no accepted $\mathsf{Bool}^{\uparrow}$ read can own those rows. The boundary therefore stops at the two endpoint histories and their finite discriminator role. □

Theorem 32.84 ($\mathsf{Bool}^{\uparrow}$ finite-data hub sibling route)**.** *The* $\mathsf{Bool}^{\uparrow}$ *endpoint rows feed* $\mathsf{Option}^{\uparrow}$ *constructor disjointness and* $\mathsf{Prod}^{\uparrow}$ *component projections through the same finite-data lattice. The* $\mathsf{Option}^{\uparrow}$ *route of chapter 33 reads the two endpoint rows as absent and present branch discriminators, while the* $\mathsf{Prod}^{\uparrow}$ *route of chapter 34 reads them as finite component selectors for product projections. Both consumers receive only the displayed* $\mathsf{Bool}^{\uparrow}$ *endpoint classifier, componentwise transport, replay, provenance, and local naming surface; neither receives host booleans, external equality, option payload data, product coordinate data, or product-pair equality from the* $\mathsf{Bool}^{\uparrow}$ *packet.*

Proof. Project the finite-data lattice already displayed by the $\mathsf{Bool}^{\uparrow}$ carrier. Its available rows are the two endpoint histories, the endpoint classifier, componentwise transport, continuation replay, provenance, and local naming certificate.

The $\mathsf{Option}^{\uparrow}$ sibling consumes only the endpoint distinction needed to separate absent and present constructors. Payload rows, nullable payload classifiers, and option equality remain owned by the option packet named in chapter 33.

The $\mathsf{Prod}^{\uparrow}$ sibling consumes the same finite distinction only as a component-facing selector. Product coordinate carriers, projection rows, and pair equality remain owned by the product packet named in chapter 34.

Thus the hub-level sibling route exports a finite dependency edge from $\mathsf{Bool}^{\uparrow}$ to both consumers without adding any non-Boolean payload, product structure, host equality, or external Boolean principle to the $\mathsf{Bool}^{\uparrow}$ packet. □

Theorem 32.85 (Bool finite-data bridge interface)**.** *The endpoint bridge of Definition 32.3, endpoint coverage, classifier endpoint exactness, constructor separation, semantic fields, and the certificate of Definition 32.42 form the bridge-ready finite-data interface from* $\mathsf{Bool}^{\uparrow}$ *to the standard two-endpoint Bool surface. Concretely, the interface consists of Definition 32.4, Definition 32.5, Definition 32.30, Definition 32.31, Definition 32.40, and Definition 32.42.*

Lean4 ✓ `BEDC.Derived.BoolUp.BoolUp StdBridge`

Proof. The endpoint bridge supplies the two visible endpoints, and Definition 32.4 identifies endpoint agreement with the kernel mark equality read by the Bool bridge.

Endpoint coverage follows from Definition 32.5. Classifier exactness and constructor separation are supplied by Definition 32.30 and Definition 32.31, so the two constructors are both covered and separated.

The semantic fields of Definition 32.40 and the public certificate of Definition 32.42 package the carrier, pattern, and ledger rows into the finite Bool surface. These are exactly the finite data a bridge layer may consume, without asserting a machine-checked standard bridge in this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$**.**

Constructive story: A Boolean here is a single mark from the kernel's two-element mark type — b0 or b1 — with no further structure layered on top. Whichever mark you start from, you stay there: equality on this tiny type is literal identity, and the four basic facts about it (each mark equals itself, and b0 differs from b1, in either reading order) form the entire distinguishing pattern, with every visible mark being just the raw mark itself. Because the type has only two elements and identity on it is automatically an equivalence, the five name-certificate obligations — that the carrier admits at least one element, and that the classifier is reflexive, symmetric, transitive, and respected by the carrier — fall out together without further work. The signature gap of a Boolean token p collects those marks m for which the source-domain admits m, the signature relation pairs m with an answer mark and an evidence ledger, and the token-introduction relation maps that answer to p; once the trivial pattern and ledger machinery is chased away, this becomes just $m = p$. The chapter registers all of this under the name $\mathsf{Bool}^{\uparrow}$; nothing beyond the kernel's mark type and its equality enters, and full Boolean algebra — operations, eliminators, decidable predicates — stays explicitly outside this closure level.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public Bool surface covers the false and true endpoint constructors, constructor case readback, semantic certificate ledger, no-hidden-leakage statement, and Definition 32.85 as the mature finite-data bridge surface over the displayed mark and history classifiers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Bool}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.BoolUp.bool\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full boolean algebra or proposition theory: boolean operations, eliminators, decidable predicates, truth-table completeness, and logic connectives as BEDC primitives.

Upgrade path: No further formal-axis upgrade remains for this bridge-checked chapter.

Lemma 32.86 (Bool finite-data classifier alignment boundary)**.** *For histories* $h, k : \mathsf{BHist}$*, the classifier side of the Bool finite-data bridge interface has the following exact alignment boundary:*

$$\mathsf{BoolHistoryClassifier}(h,k) \iff \exists v, w : \mathsf{BMark},\ \mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \mathsf{hsame}(h, \mathsf{BoolEndpoint}$$

Under this equivalence, the transported displayed rows are precisely the two endpoint readback rows $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ *and* $\mathsf{hsame}(k, \mathsf{BoolEndpoint}(w))$*, together with the mark-comparison row* $\mathsf{msame}(v, w)$*. The retained certificate rows are precisely the source, pattern, and ledger rows of Definition 32.40, all read as* $\mathsf{BoolHistoryCarrier}$*. Every row outside the interface listed in Definition 32.85 is forgotten by this alignment boundary; in particular, it exports no host proposition, decidable predicate, eliminator, truth-table-completeness row, payload carrier, payload classifier, or option-packaging row.*

Proof. The displayed equivalence is exactly Definition 32.13. Its forward direction says that a classifier comparison between h and k exposes only source-qualified endpoint marks v, w, the two endpoint readback rows, and the mark comparison $\mathsf{msame}(v, w)$. Its reverse direction says that those rows are already sufficient to rebuild the Bool history classifier comparison, so no additional classifier coordinate is consumed.

By Definition 32.3, the endpoint map has only the two visible target rows $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Endpoint coverage in Definition 32.5 supplies no third readback case, and constructor separation in Definition 32.31 prevents the two constructor rows from being identified by the classifier. Thus the readback part of the boundary transports exactly the displayed endpoint rows and their msame comparison.

The certificate part is fixed by Definition 32.40: source, pattern, and ledger are all the same predicate $\mathsf{BoolHistoryCarrier}$. The public semantic certificate in Definition 32.42 therefore retains only the carrier-classifier ledger named in Definition 32.85. Finally, Definition 32.72 rules out host propositions, decidable predicates, eliminators, and truth-table completeness, and the finite-data interface list contains no payload carrier, payload classifier, or option-packaging row. Those nonlisted rows are therefore forgotten rather than transported. □

Lemma 32.87 (Bool finite-data domain-codomain mirror boundary)**.** *In the finite-data bridge interface of Definition 32.85, the codomain-facing Bool read is the endpoint mirror of the domain-facing carrier read. Concretely, every public codomain read of the standard two-endpoint Bool surface is forced through one of the two rows* $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ *or* $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$*, and every such endpoint row is already a domain-side* $\mathsf{BoolHistoryCarrier}$ *row. Its observation, result, termination, and certificate data are therefore exactly the endpoint exactness row, endpoint coverage row, classifier endpoint exactness row, constructor separation row, semantic field row, and concrete semantic certificate row cited in Definition 32.85. There is no separate codomain carrier, codomain classifier, or codomain certificate row in this Bool finite-data bridge.*

Proof. Start with a public codomain read of the standard two-endpoint Bool surface. By Definition 32.70, the only public constructors of that surface are the two endpoint rows $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. The readback theorem Definition 32.71 says that every public endpoint read is classified by exactly one of those rows, and that agreement of such rows is exactly mark agreement through msame.

Each of the two displayed endpoint rows is also a carrier row on the domain side. This is precisely the reverse direction of Definition 32.5: an endpoint witness $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ repacks as $\mathsf{BoolHistoryCarrier}(h)$. The agreement comparison is not a codomain-specific classifier, because Definition 32.4 identifies endpoint agreement with msame on the underlying marks, while Definition 32.30 and Definition 32.31 say that the history classifier has exactly the two endpoint classes and cannot mix them.

It remains to identify the certificate side. The semantic fields in Definition 32.40 read source, pattern, and ledger all as $\mathsf{BoolHistoryCarrier}$, and Definition 32.42 supplies the public semantic certificate for that same carrier and classifier. Thus reversing attention from the domain read to the codomain read only renames the endpoint row being inspected. It does not create any extra carrier, classifier, ledger, result, termination, or certificate row beyond the finite packet already listed in Definition 32.85. □

Lemma 32.88 (Bool finite-data substrate bisimulation boundary)**.** *Let* Sub *be one of the substrate read names* $\mathsf{Rule110}$, $\mathsf{Cellular}$, Binary, $\mathsf{FKernel}$, *or raw* BHist. *Within the finite-data bridge interface of Definition 32.85, a consumer-facing read from* Sub *into* $\mathsf{Bool}^{\uparrow}$ *has only the following BEDC-local*

relation certificate. Its source is the named substrate read Sub*, its target is the Bool finite-data packet, its direction is*

$$\mathsf{Sub}\text{-}\textit{read} \longrightarrow \textit{displayed endpoint readback rows} \longrightarrow \mathsf{Bool}^{\uparrow}\text{-}\textit{consumer},$$

and its relation kind is substrate mirror with bisimulation comparison on the two endpoint observations. Concretely, for histories $h, k : \mathsf{BHist}$ *the observable comparison must factor through displayed marks* $v, w : \mathsf{BMark}$ *as*

$$\mathsf{BoolSourceSpec}(v) \wedge \mathsf{BoolSourceSpec}(w) \wedge \mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \wedge \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \wedge \mathsf{msame}(v, w),$$

and this displayed relation is equivalent to $\mathsf{BoolHistoryClassifier}(h, k)$*. The retained certificate rows are precisely* BoolHistoryCarrier*,* BoolHistoryClassifier*,* BoolHistoryCarrier *as source, pattern, and ledger, and the semantic certificate of Definition 32.42. No accepted read may replace this mirror certificate by definitional equality between* Sub *and* $\mathsf{Bool}^{\uparrow}$*, by equality between a host substrate and the Bool carrier, or by an unlisted Rule110 state, cellular clock, cellular neighborhood, external binary trace, FKernel extension, FKernel continuation, FKernel package, FKernel gap, or raw* BHist *constructor coordinate.*

Proof. The finite-data interface is already enumerated in Definition 32.85. Its rows are endpoint bridge exactness, endpoint coverage, classifier endpoint exactness, constructor separation, the semantic fields, and the concrete semantic certificate. Thus a substrate-facing consumer has no independent row from which to read a Rule110 state, a cellular clock or neighborhood, an external binary trace, an FKernel extension, continuation, package, or gap coordinate, or a raw constructor identity for BHist.

The endpoint part of the certificate is forced by Definition 32.3: the only displayed target rows are $\mathsf{BoolEndpoint}(\mathsf{b0}) = \mathsf{e}$ and $\mathsf{BoolEndpoint}(\mathsf{b1}) = \mathsf{E1}(\mathsf{e})$. Endpoint coverage in Definition 32.5 says that every Bool history carrier read reaches one of those endpoint rows, and constructor separation prevents the two endpoint rows from being collapsed by the Bool classifier.

The comparison part is exactly the boundary in Definition 32.86. Its forward direction reads any $\mathsf{BoolHistoryClassifier}(h, k)$ comparison as source-qualified marks v, w, two endpoint readback rows $\mathsf{hsame}(h, \mathsf{BoolEndpoint}(v))$ and $\mathsf{hsame}(k, \mathsf{BoolEndpoint}(w))$, and the mark comparison $\mathsf{msame}(v, w)$. Its reverse direction rebuilds the Bool history classifier from precisely those rows. Hence the displayed endpoint observations are preserved and reflected by the same finite comparison row; this is a bisimulation boundary on the local Bool observations, not a definitional equality boundary between substrates.

The direction and mirror reading are fixed by Definition 32.87: a codomain-facing Bool read is the endpoint mirror of the domain-facing carrier read. Therefore a substrate read can be consumed only after it has been projected to the endpoint readback packet and then read through the Bool carrier and classifier rows. The semantic side contributes no extra substrate coordinate, because Definition 32.40 reads source, pattern, and ledger all as BoolHistoryCarrier, and Definition 32.42 certifies that same carrier and classifier.

Finally, Definition 32.72 excludes host propositions, decidable predicates, eliminators, and truth-table completeness, and the finite interface list contains none of the substrate-specific rows named in the statement. A receiving gate that re-reads the displayed rows must therefore reject any unlisted host or source coordinate. What remains is exactly the finite substrate mirror certificate above: source named by the incoming substrate read, target named by the Bool finite-data packet, direction through the displayed endpoint rows, and bisimulation comparison given by msame under the Bool history classifier. □

The nullable tag surface of $\mathsf{Option}^{\uparrow}$ is read through the finite $b0/b1$ sibling discipline of chapter 32; the child file records the payload and classifier obligations after that tag route is fixed.

Chapter 33

A Concrete Naming Certificate for $\mathsf{Option}^{\uparrow}$

This chapter separates two certificate scopes. The checked BEDC object is the nullable history envelope $\mathsf{OptionHistoryCarrier}_S(h) \equiv \mathsf{hsame}(h, \mathsf{e}) \vee S(h)$ together with the classifier endpoint relation carried by hsame. The constructor-tagged optional value interface for $\mathsf{Option}^{\uparrow}$ is a separate visible-tag certificate over a different carrier presentation; it is not a strict strengthening of the nullable envelope unless the bridge data of Definition 33.103 are supplied, with descent given by Theorem 33.104.

33.1 Roadmap template status

1. **Traditional target**: optional data with one absent value and present values over a source object.
2. **BEDC source**: any licensed source carrier A equipped with its existing name certificate.
3. **Carrier choice**: the typed $\mathsf{OptionCarrier}$ has none and some constructors, while the checked history carrier is the nullable envelope $\mathsf{OptionHistoryCarrier}_S(h) \equiv \mathsf{hsame}(h, \mathsf{e}) \vee S(h)$.
4. **Classifier choice**: the typed $\mathsf{OptionClassifierSpec}$ relates only matching option tags through the source classifier; $\mathsf{OptionHistoryClassifier}$ is the nullable-history classifier consisting of endpoint envelope-carrier witnesses and endpoint hsame. These classifiers live over different carriers unless $\mathsf{TaggedOptionToNullableBridge}(S, \sim_S)$ supplies Definition 33.103.
5. **Stability obligations**: typed optional carrier transport, typed absent/present separation, lifted typed classifier symmetry and transitivity, nullable-history endpoint transport, nullable-history classifier symmetry and transitivity, and for the constructor-tagged optional-history certificate a payload coherence condition converting source-carrier hsame between middle payloads into the source classifier.
6. **Ledger**: the typed option interface records the absent-or-present tag and present payload evidence, while the checked nullable-history envelope ledger records only the raw envelope-carrier witness and the raw-to-visible hsame link.
7. **Core theorem list**: typed carrier cases, typed absent/present no-confusion, present inversion, typed classifier equivalence, tagged same-tag exactness, tagged mixed-tag separation, source-

compatibility for the tagged semantic target, nullable-history carrier transport, nullable-history classifier transport, and the nullable-history semantic certificate.

8. **Exactness predicate**: matching-tag exactness belongs to OptionClassifierSpec; the tagged history classifier has same-tag exactness through visible absent and present endpoints plus mixed-tag separation, while the checked nullable-history classifier exactness is endpoint envelope membership plus endpoint hsame and carries no present constructor tag. The tagged exactness clauses do not by themselves produce nullable envelope membership for a visible present endpoint.

9. **Standard bridge**: the checked nullable envelope has only empty-or-source endpoint readback. Ordinary none and some(a) readback belongs to the typed and tagged option interface under TaggedOptionHistoryCarrier, TaggedOptionHistoryClassifier, same-tag exactness, mixed-tag separation, and the tag-plus-source ledger; using that readback as nullable-envelope evidence is exactly the descent of Theorem 33.104.

10. **Status**: Theorem 33.98 is the checked nullable empty-or-source envelope only; Theorem 33.44 is the Phase Finite Data constructor-tagged $\mathsf{Option}^{\uparrow}$ CheckedCert under a source NameCert and TaggedOptionSourceHsameCompatible; comparison with the nullable carrier presentation uses Definition 33.103.

33.2 Carrier definition

Definition 33.1 (Option carrier)**.** *For a licensed source carrier A, define an inductive carrier with constructors* none *and* some(a)*.*

Lean4 ✓ `BEDC.Derived.OptionUp.OptionCarrier cases`

The checked BEDC-history carrier used below is the nullable semantic carrier $\mathsf{OptionHistoryCarrier}_S(h) \equiv \mathsf{hsame}(h, \mathsf{e}) \vee S(h)$, not a tagged history carrier.

33.3 The certificate

Definition 33.2 (Option source specification)**.** *There are two source scopes. For the checked nullable-history certificate, the source input is a BEDC history predicate $S : \mathsf{BHist} \to \mathsf{Prop}$ and $\mathsf{OptionHistoryCarrier}_S(h)$ is exactly the empty-or-source envelope $\mathsf{hsame}(h, \mathsf{e}) \vee S(h)$. A source* NameCert *belongs only to the typed and constructor-tagged option interface used later in Theorem 33.44; it is not part of the checked nullable-envelope definition displayed here.*

Lean4 (definition) `BEDC.Derived.OptionUp.OptionHistoryCarrier`

Theorem 33.3 (Option Bool tag sibling route)**.** *The constructor-tagged $\mathsf{Option}^{\uparrow}$ interface reads its absent/present tag through the finite $\mathsf{Bool}^{\uparrow}$ tag discipline of chapter 32 before applying the nullable envelope bridge. The checked nullable envelope remains empty-or-source history data: its rows are the $\mathsf{hsame}(h, \mathsf{e})$ absent branch, the source branch $S(h)$, endpoint* hsame *transport, and the local envelope classifier, not an additional Boolean coordinate.*

Proof. Project the source specification of Definition 33.2. It separates the typed and constructor-tagged option interface from the checked nullable-history carrier $\mathsf{OptionHistoryCarrier}_S(h) \equiv \mathsf{hsame}(h, \mathsf{e}) \vee S(h)$.

Read the constructor-tagged interface first. Its absent and present alternatives require a finite tag row, and that sibling row is supplied by the $\mathsf{Bool}^{\uparrow}$ discipline of chapter 32. The tag is consumed before any descent to the nullable envelope.

Finally apply the nullable bridge boundary named in the same source specification. The bridge may translate a visible absent/present read into empty-or-source history evidence, but the checked nullable envelope still records only the envelope witness and its hsame transport. Thus the Bool sibling route classifies the tagged presentation without becoming a new nullable-envelope coordinate. □

Theorem 33.4 (Option history carrier transport)**.** *If h and k are internally history-same, membership of h in* $\mathsf{OptionHistoryCarrier}_S$ *transports to membership of k in the same carrier. In the source branch this does not assume that an arbitrary source classifier is stable under* hsame*; it uses the checked identity characterization of history-sameness on* BHist*,* BEDC.FKernel.Hist.hsame_iff_eq*.*

Proof. Invert the optional carrier witness. In the absent branch, use only symmetry and transitivity of hsame: reverse the displayed $\mathsf{hsame}(h, k)$ link and compose it with the stored $\mathsf{hsame}(h, \mathsf{e})$ witness. In the source branch, convert the displayed $\mathsf{hsame}(h, k)$ witness via BEDC.FKernel.Hist.hsame_iff_eq, rewrite the endpoint, and transport the stored predicate witness from $S(h)$ to $S(k)$. The exported nullable classifier remains exactly $\mathsf{OptionHistoryCarrier}_S(h) \wedge \mathsf{OptionHistoryCarrier}_S(k) \wedge \mathsf{hsame}(h, k)$, so this proof does not export identity reflection as a public equality principle for $\mathsf{Option}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier hsame transport`

Theorem 33.5 (Option history carrier unary prefix closure)**.** *Let p, h : BHist. If p is a unary history and* $\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(h)$ *holds, then*

$$\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(\mathsf{append}(p, h))$$

holds.

Proof. Split the nullable carrier witness for h. In the absent branch, transport the empty unary-history witness along $\mathsf{hsame}(\mathsf{e}, h)$ and close by unary closure under append with the prefix p. In the source branch, the source witness is already a unary-history witness for h, and unary append closure again gives the carrier witness for $\mathsf{append}(p, h)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier unary prefix closed`

Theorem 33.6 (Option history carrier unary suffix closure)**.** *Let p, h : BHist. If p is a unary history and* $\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(h)$ *holds, then*

$$\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(\mathsf{append}(h, p))$$

holds.

Proof. Split the nullable carrier witness for h. In the absent branch, transport the empty unary-history witness along $\mathsf{hsame}(\mathsf{e}, h)$ and close by unary closure under append with the suffix p. In the source branch, the source witness is already a unary-history witness for h, and unary append closure again gives the carrier witness for $\mathsf{append}(h, p)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier unary suffix closed`

Theorem 33.7 (Option history carrier unary append context equivalence)**.** *Let p, q, h : BHist. If p and q are unary histories, then nullable unary option carrier membership is invariant under placing p on the left and q on the right:*

$$\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(\mathsf{append}(p, \mathsf{append}(h, q))) \iff \mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(h).$$

Proof. Split the nullable carrier. In the absent branch, two applications of append_eq_empty_iff recover the uncloaked empty endpoint. In the source branch, apply the unary append context middle equivalence. The reverse direction either rebuilds the contextual unary carrier from an empty middle endpoint or closes the source unary witness under the same two append contexts. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier unary append context iff`

Theorem 33.8 (Option history classifier unary prefix closure)**.** *Let p, h, k : BHist. If p is unary and* $\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(h, k)$ *holds, then the nullable option classifier is preserved by placing p on the left:*

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(\mathsf{append}(p, h), \mathsf{append}(p, k)).$$

Proof. Unpack the nullable classifier into the two endpoint carrier witnesses and the stored $\mathsf{hsame}(h, k)$ link. Apply the unary-prefix carrier closure to both endpoints. The remaining classifier field is the prefixed endpoint sameness, obtained by eliminating the $\mathsf{hsame}(h, k)$ witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier unary prefix closed`

Theorem 33.9 (Option history classifier unary suffix closure)**.** *Let p, h, k : BHist. If p is unary and* $\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(h, k)$ *holds, then the nullable option classifier is preserved by placing p on the right:*

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(\mathsf{append}(h, p), \mathsf{append}(k, p)).$$

Proof. Unpack the nullable classifier into the two endpoint carrier witnesses and the stored $\mathsf{hsame}(h, k)$ link. Apply the unary-suffix carrier closure to both endpoints. The remaining classifier field is the suffixed endpoint sameness, obtained by eliminating the $\mathsf{hsame}(h, k)$ witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier unary suffix closed`

Definition 33.10 (Option pattern specification)**.** *The pattern is absent-or-present generation: one null constructor and one wrapper constructor over source witnesses.*

Lean4 (definition) `BEDC.Derived.OptionUp.OptionHistoryPattern`
Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryPattern carrier iff`

Definition 33.11 (Option classifier specification)**.** *The typed classifier relates* none *only to* none*, and relates* $\mathsf{some}(a)$ *to* $\mathsf{some}(a')$ *exactly when the source classifier relates a and a'. The checked history-level classifier for* OptionHistoryCarrier *is the nullable classifier: both endpoints lie in the empty-or-source envelope, and the endpoints are related by* hsame*.*

Lean4 (definition) `BEDC.Derived.OptionUp.OptionClassifierSpec`
Lean4 ✓ `BEDC.Derived.OptionUp.optionClassifierSpec cases`
Lean4 (definition) `BEDC.Derived.OptionUp.OptionClassifier`
Lean4 (definition) `BEDC.Derived.OptionUp.OptionHistoryClassifier`
Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec none some absurd`
Lean4 ✓ `BEDC.Derived.OptionUp.optionClassifier some iff`
Lean4 ✓ `BEDC.Derived.OptionUp.option classifier inversion`

Theorem 33.12 (Option history carrier source monotonicity)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be source predicates. If* $\forall h.\ S(h) \to T(h)$*, then for every* $h : \mathsf{BHist}$*,* $\mathsf{OptionHistoryCarrier}_S(h)$ *implies* $\mathsf{OptionHistoryCarrier}_T(h)$*. Under the same hypothesis, for every* $h, k : \mathsf{BHist}$*,* $\mathsf{OptionHistoryClassifier}_S(h, k)$ *implies* $\mathsf{OptionHistoryClassifier}_T(h, k)$*.*

Proof. For the carrier clause, split the nullable carrier witness. The absent branch already supplies the same $\mathsf{hsame}(h, \mathsf{e})$ witness for the target envelope. In the source branch, apply the displayed implication to the stored $S(h)$ witness and obtain the required $T(h)$ witness.

For the classifier clause, unfold the nullable classifier data: both endpoints come with carrier witnesses and the classifier stores the same endpoint $\mathsf{hsame}(h, k)$ witness. Apply the carrier clause to the two endpoint carrier witnesses, and keep the stored $\mathsf{hsame}(h, k)$ witness unchanged. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistorySource monotonicity`

Definition 33.13 (Tagged option history carrier)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *by the two constructor-shaped cases:*

1. *absent:* $\mathsf{hsame}(h, \mathsf{e})$*;*
2. *present: there is a payload history* a *such that* $S(a)$ *and* $\mathsf{hsame}(h, \mathsf{E1}(a))$*.*

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier`

Theorem 33.14 (Tagged option history present exactness)**.** *For any BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, a present endpoint in the tagged carrier is exactly a source payload at the visible present tag:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(\mathsf{E1}(h)) \iff \exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, a).$$

Proof. For the forward implication, case-analyze the tagged carrier witness for $\mathsf{E1}(h)$. The absent branch gives $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e})$ and is eliminated by generated-history no-confusion between a present endpoint and the empty endpoint. The present branch gives a payload history a, a source witness $S(a)$, and $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(a))$; applying the one-constructor inversion theorem for hsame strips the visible present tags and yields $\mathsf{hsame}(h, a)$, so a is the required payload. Conversely, given a payload a with $S(a)$ and $\mathsf{hsame}(h, a)$, use the present branch of $\mathsf{TaggedOptionHistoryCarrier}_S$. The endpoint witness is $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(a))$, obtained by applying $e1$ congruence to $\mathsf{hsame}(h, a)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier present exactness`

Theorem 33.15 (Tagged option history carrier present hsame readback)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, p*, if* h *inhabits the tagged option carrier and is history-same to the visible present endpoint* $\mathsf{E1}(p)$*, then the visible payload reads back to a source payload:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \Longrightarrow \exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(p, a).$$

Proof. Case-analyze the tagged option carrier witness for h. The absent branch gives $\mathsf{hsame}(h, \mathsf{e})$; composing its symmetry with $\mathsf{hsame}(h, \mathsf{E1}(p))$ would make e history-same to a present endpoint, contradicting generated-history no-confusion. The present branch gives a payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$. Composing symmetry of $\mathsf{hsame}(h, \mathsf{E1}(p))$ with this endpoint witness yields $\mathsf{hsame}(\mathsf{E1}(p), \mathsf{E1}(a))$, and one-constructor inversion gives $\mathsf{hsame}(p, a)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier present hsame readback`

Theorem 33.16 (Tagged option history carrier present source transport)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, assume* S *is stable along history sameness:*

$$\forall x\, y : \mathsf{BHist}.\ \mathsf{hsame}(x, y) \to S(x) \to S(y).$$

Then a tagged carrier witness at the visible present endpoint $\mathsf{E1}(h)$ *reads back to the source predicate at its displayed payload:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(\mathsf{E1}(h)) \Longrightarrow S(h).$$

Proof. Apply Theorem 33.14 to the carrier witness at $\mathsf{E1}(h)$. It gives a payload a with $S(a)$ and $\mathsf{hsame}(h, a)$. Symmetry turns this into $\mathsf{hsame}(a, h)$, and the displayed source transport sends $S(a)$ to $S(h)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier present source transport`

Theorem 33.17 (Tagged option history carrier e1 source exactness)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, assume* S *is stable along history sameness:*

$$\forall x\, y : \mathsf{BHist}.\ \mathsf{hsame}(x, y) \to S(x) \to S(y).$$

Then the visible present endpoint $\mathsf{E1}(h)$ *inhabits the tagged option carrier exactly when its displayed payload inhabits the source carrier:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(\mathsf{E1}(h)) \Longleftrightarrow S(h).$$

Proof. The forward direction is the present-source readback above. For the reverse direction, use the present branch of the tagged option carrier with payload h, source witness $S(h)$, and reflexivity $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(h))$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier e1 source iff`

For the finite-data certificate template, the traditional target is absent-vs-present option tags; the BEDC source is BHist with hsame; the carrier is TaggedOptionHistoryCarrier; no classifier participates in this obligation. The stability obligation is constructor no-confusion, the ledger exposes the visible tag, the core theorem is this disjointness lemma, exactness is absent/present separation, the standard bridge is none/some, and the site status is ProofTarget.

Lemma 33.18 (Tagged option history carrier constructor separation)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, history* h*, and payload history* a*, an absent endpoint and a present endpoint cannot be witnessed by the same carrier endpoint:*

$$\mathsf{hsame}(h, \mathsf{e}) \wedge S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \quad \Longrightarrow \quad \bot.$$

Proof. Compose symmetry of $\mathsf{hsame}(h, \mathsf{e})$ with $\mathsf{hsame}(h, \mathsf{E1}(a))$ to obtain $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(a))$. BEDC generated-history no-confusion between the empty constructor and the one-step constructor eliminates this endpoint equality. The source witness $S(a)$ records that the present payload is a valid source history but is not needed for the constructor contradiction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier constructor separation`

Definition 33.19 (Tagged option history classifier)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifier* $\sim_S$*, define* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$ *by the two constructor-matching cases:*

1. *absent-absent:* $\mathsf{hsame}(h, \mathsf{e})$ *and* $\mathsf{hsame}(k, \mathsf{e})$*;*
2. *present-present: there are payload histories* a, b *such that* $S(a)$*,* $S(b)$*,* $a \sim_S b$*,* $\mathsf{hsame}(h, \mathsf{E1}(a))$*, and* $\mathsf{hsame}(k, \mathsf{E1}(b))$*.*

Definition 33.20 (Tagged option source hsame compatibility)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifier* $\sim_S$*, define* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$ *to be the property that for all payload histories* a, b*, if* $S(a)$*,* $S(b)$*, and* $\mathsf{hsame}(a, b)$*, then* $a \sim_S b$*.*

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionSourceHsameCompatible`

Theorem 33.21 (Tagged option history classifier endpoint carriers)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k*, if* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$ *holds, then both* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *and* $\mathsf{TaggedOptionHistoryCarrier}_S(k)$ *hold.*

Proof. Case-analyze the tagged classifier witness. The absent-absent branch stores $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$, so the absent carrier clause rebuilds both endpoint carriers. The present-present branch stores payload histories a, b, source witnesses $S(a)$ and $S(b)$, the payload classifier witness $a \sim_S b$, and visible endpoint links $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$. The two present carrier clauses reuse the corresponding payload histories, source witnesses, and endpoint links, so both classified endpoints are tagged option carrier endpoints. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier endpoint carriers`

Theorem 33.22 (Tagged option history classifier transitivity)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifier* $\sim_S$*, if* $\sim_S$ *is transitive, then* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}$ *is transitive.*

Proof. Case-analyze the two tagged classifier witnesses. The absent/absent branch rebuilds the absent classifier branch. The mixed absent/present branches present the shared middle history both as e and as $\mathsf{E1}(a)$, so BEDC generated-history no-confusion eliminates them. In the present/present branch, the two endpoint witnesses for the shared middle history give $\mathsf{hsame}(\mathsf{E1}(b), \mathsf{E1}(c))$, and one-constructor inversion gives $\mathsf{hsame}(b, c)$. Transport along this equality identifies the middle payloads, and source-classifier transitivity composes the two payload classifier witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier trans`

Theorem 33.23 (Tagged option history classifier left confluence)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifier* $\sim_S$*, if* $\sim_S$ *is symmetric and transitive, then any two tagged option classifier witnesses with the same left endpoint compose to a classifier witness between their right endpoints:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k) \wedge \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, r) \Longrightarrow \mathsf{TaggedOptionHistoryClassifier}_S$$

Proof. Reverse the first classifier witness using symmetry of the source classifier. The absent branch swaps the two empty endpoint witnesses. The present branch swaps the two tagged visible endpoints and applies symmetry to the payload classifier witness. The reversed classifier from k to h then composes with the classifier from h to r by tagged option classifier transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier left confluence`

Theorem 33.24 (Tagged option history classifier right confluence)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_S$, if $\sim_S$ is symmetric and transitive, then any two tagged option classifier witnesses with the same right endpoint compose to a classifier witness between their left endpoints:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \wedge \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(r,k) \Longrightarrow \mathsf{TaggedOptionHistoryClassifier}_{S,}$$

Proof. Reverse the second classifier witness using symmetry of the source classifier. The absent branch swaps the empty endpoint witnesses, and the present branch swaps the tagged visible endpoints while reversing the payload classifier witness. The classifier from h to k then composes with the reversed classifier from k to r by tagged option classifier transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier right confluence`

Theorem 33.25 (Tagged option history present-present exactness)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, the tagged option history classifier is exact on two visible present endpoints:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{E1}(h), \mathsf{E1}(k)) \quad \Longleftrightarrow \quad \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h,a) \wedge \mathsf{hsame}(k,b) \wedge a \sim_S$$

Proof. For the forward direction, case-analyze the tagged classifier witness. The absent-absent branch would contain $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e})$ and is eliminated by one-versus-empty no-confusion. The present-present branch gives payload histories a, b, source witnesses $S(a)$ and $S(b)$, a source classifier witness $a \sim_S b$, and endpoint witnesses $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(a))$ and $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(b))$; one-constructor inversion strips the visible $e1$ tags and yields $\mathsf{hsame}(h,a)$ and $\mathsf{hsame}(k,b)$. For the reverse direction, the displayed payload data rebuilds the present-present classifier branch after applying one-constructor congruence for hsame to the two payload endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier present present exactness`

Theorem 33.26 (Tagged option history visible present exactness)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, a transitivity hypothesis*

$$\forall x\, y\, z.\ x \sim_S y \to y \sim_S z \to x \sim_S z,$$

and $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, if $S(h)$ and $S(k)$, then the tagged option history classifier is exact on the two source-visible present endpoints:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{E1}(h), \mathsf{E1}(k)) \quad \Longleftrightarrow \quad h \sim_S k.$$

Proof. For the forward direction, apply Theorem 33.25 to the tagged classifier witness. It provides payload histories a, b, source witnesses $S(a)$ and $S(b)$, endpoint witnesses $\mathsf{hsame}(h,a)$ and $\mathsf{hsame}(k,b)$, and a payload classifier witness $a \sim_S b$. The compatibility hypothesis gives $h \sim_S a$ from $S(h)$, $S(a)$, and $\mathsf{hsame}(h,a)$; symmetry of hsame turns $\mathsf{hsame}(k,b)$ into $\mathsf{hsame}(b,k)$, and compatibility gives $b \sim_S k$ from $S(b)$, $S(k)$, and that symmetric endpoint witness. Two applications of the displayed transitivity hypothesis compose $h \sim_S a$, $a \sim_S b$, and $b \sim_S k$ into $h \sim_S k$.

For the reverse direction, use Theorem 33.25 with payloads h and k. The source witnesses are the assumptions $S(h)$ and $S(k)$, the endpoint witnesses are reflexivity of hsame at h and k, and the payload classifier witness is the assumed $h \sim_S k$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible present exactness`

Theorem 33.27 (Tagged option hsame source compatibility)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*,*

$$\mathsf{TaggedOptionSourceHsameCompatible}(S, \mathsf{hsame})$$

holds.

Proof. Unfold Definition 33.20. Given payload histories a, b, source witnesses $S(a)$ and $S(b)$, and the displayed witness $\mathsf{hsame}(a, b)$, return that same history-sameness witness as the source classifier witness. □

Theorem 33.28 (Tagged option hsame present exactness)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and assume* $\mathsf{source_transport} : \forall x\, y : \mathsf{BHist}.\ \mathsf{hsame}(x, y) \to S(x) \to S(y)$*. For histories* h, k*,*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(\mathsf{E1}(h), \mathsf{E1}(k)) \iff S(h) \wedge S(k) \wedge \mathsf{hsame}(h, k).$$

Proof. Apply Theorem 33.25 with source classifier hsame. In the forward direction, the exactness witness gives payload histories a, b, source witnesses $S(a)$ and $S(b)$, endpoint links $\mathsf{hsame}(h, a)$ and $\mathsf{hsame}(k, b)$, and a payload link $\mathsf{hsame}(a, b)$. Symmetry turns $\mathsf{hsame}(h, a)$ into $\mathsf{hsame}(a, h)$, so the first application of $\mathsf{source_transport}$ sends $S(a)$ to $S(h)$; symmetry turns $\mathsf{hsame}(k, b)$ into $\mathsf{hsame}(b, k)$, so the second application sends $S(b)$ to $S(k)$. Compose $\mathsf{hsame}(h, a)$, $\mathsf{hsame}(a, b)$, and the symmetric form of $\mathsf{hsame}(k, b)$ to obtain $\mathsf{hsame}(h, k)$. In the reverse direction, choose payloads h and k; the two source witnesses are displayed, the endpoint links are reflexivity of hsame, and the payload classifier witness is the displayed $\mathsf{hsame}(h, k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier hsame present exactness`

Definition 33.29 (Tagged option history pattern)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, the visible tag pattern* $\mathsf{TaggedOptionHistoryPattern}_S(h)$ *holds exactly when* h *is either an absent endpoint with* $\mathsf{hsame}(h, \mathsf{e})$ *or a present endpoint with payload history* a *satisfying* $S(a)$ *and* $\mathsf{hsame}(h, \mathsf{E1}(a))$*.*

Definition 33.30 (Tagged option history ledger)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, the visible tag ledger* $\mathsf{TaggedOptionHistoryLedger}_S(h)$ *records the same two endpoint forms as certificate data: the absent record stores* $\mathsf{hsame}(h, \mathsf{e})$*, and the present record stores a payload history* a *together with* $S(a)$ *and* $\mathsf{hsame}(h, \mathsf{E1}(a))$*.*

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionHistoryLedger`

Theorem 33.31 (Tagged option history carrier, pattern, and ledger equivalence)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, the tagged option history carrier is equivalent to both visible certificate presentations:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \iff \mathsf{TaggedOptionHistoryPattern}_S(h)$$

and

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \iff \mathsf{TaggedOptionHistoryLedger}_S(h).$$

Proof. Unfold the three tagged option predicates. In the absent case, each presentation stores exactly the witness $\mathsf{hsame}(h, \mathsf{e})$, so the witness is reused unchanged in both directions. In the present case, each presentation stores the same payload history a together with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$, so the same triple rebuilds the carrier, pattern, and ledger witnesses. The two cases exhaust the tagged option history carrier and the two visible certificate presentations. □

Theorem 33.32 (Tagged option ledger present payload readback)**.** *Template role: traditional target present optional payload recovery; BEDC source* BHist *with* hsame*; carrier* TaggedOptionHistoryCarrier*; classifier unused for this readback; stability obligation carrier transport along* hsame *plus* e1 *inversion; ledger* TaggedOptionHistoryLedger*; core theorem visible present payload readback; exactness visible present endpoint gives a source payload witness; standard bridge* $\mathsf{some}(a)$*; status* ProofTarget*.*

For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, p*, if* $\mathsf{TaggedOptionHistoryLedger}_S(h)$ *and* $\mathsf{hsame}(h, \mathsf{E1}(p))$*, then the visible present endpoint reads back a source payload:*

$$\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(p, a).$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryLedger present payload readback`

Proof. Convert the ledger witness to a tagged carrier witness using Theorem 33.31. Transport that carrier witness across $\mathsf{hsame}(h, \mathsf{E1}(p))$ by Theorem 33.33. The resulting carrier witness at $\mathsf{E1}(p)$ is then inverted by Theorem 33.14, which exposes a payload history a with $S(a)$ and $\mathsf{hsame}(p, a)$. □

Theorem 33.33 (Tagged option history carrier transport)**.** *For any BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{TaggedOptionHistoryCarrier}_S(k)$*.*

Proof. Case-analyze the tagged carrier witness for h. In the absent case, compose the symmetry of $\mathsf{hsame}(h, k)$ with the stored endpoint witness $\mathsf{hsame}(h, \mathsf{e})$ to obtain $\mathsf{hsame}(k, \mathsf{e})$, then rebuild the absent clause. In the present case, keep the same payload a and source witness $S(a)$. Compose the symmetry of $\mathsf{hsame}(h, k)$ with the stored endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(a))$ to obtain $\mathsf{hsame}(k, \mathsf{E1}(a))$, then rebuild the present clause. □

Theorem 33.34 (Tagged option history classifier endpoint transport)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k, h', k'*, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k) \wedge \mathsf{hsame}(h, h') \wedge \mathsf{hsame}(k, k')$$

holds, then

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h', k')$$

holds.

Proof. Case-analyze the tagged classifier witness for h and k. In the absent-absent branch, compose the symmetry of $\mathsf{hsame}(h, h')$ with the stored endpoint witness $\mathsf{hsame}(h, \mathsf{e})$ to obtain $\mathsf{hsame}(h', \mathsf{e})$, and compose the symmetry of $\mathsf{hsame}(k, k')$ with the stored endpoint witness $\mathsf{hsame}(k, \mathsf{e})$ to obtain $\mathsf{hsame}(k', \mathsf{e})$; these two witnesses rebuild the absent-absent classifier branch. In the present-present branch, keep the same payload histories a, b, source witnesses $S(a)$ and $S(b)$, and payload classifier witness $a \sim_S b$. Compose the symmetry of $\mathsf{hsame}(h, h')$ with the stored endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(a))$ to obtain $\mathsf{hsame}(h', \mathsf{E1}(a))$, and compose the symmetry of $\mathsf{hsame}(k, k')$ with the stored endpoint witness $\mathsf{hsame}(k, \mathsf{E1}(b))$ to obtain $\mathsf{hsame}(k', \mathsf{E1}(b))$; these data rebuild the present-present classifier branch. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier hsame transport`

Theorem 33.35 (Tagged option history classifier stability fields)**.** *For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and source certificate* $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, the following five clauses are exactly the non-inhabitation stability fields for the tagged option history carrier and classifier. Carrier inhabitation is the absent endpoint at* e *and is supplied separately in Theorem 33.44:*

1. *carrier-relative reflexivity: if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *holds, then* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,h)$ *holds;*
2. *classifier symmetry: if* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$ *holds, then* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(k,h$ *holds;*
3. *classifier transitivity: if* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$ *and* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(k,u)$ *hold, then* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,u)$ *holds;*
4. *classifier-induced carrier transport: if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *and* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,$ *hold, then* $\mathsf{TaggedOptionHistoryCarrier}_S(k)$ *holds;*
5. *classifier endpoint transport: if* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$, $\mathsf{hsame}(h,h')$, *and* $\mathsf{hsame}(k,k')$ *hold, then* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h',k')$ *holds.*

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier stability fields`

Proof. For reflexivity, case-analyze the carrier witness. The absent case rebuilds the absent classifier branch from the stored empty endpoint witness and reflexivity, symmetry, and transitivity of hsame; the present case keeps the same payload and applies the reflexivity field of cert to the source carrier witness. For symmetry, case-analyze the tagged classifier witness. The absent branch reverses the stored endpoint links by symmetry of hsame, and the present branch reverses the payload classifier with cert.equiv_symm while swapping the two endpoint witnesses.

For transitivity, split the two classifier witnesses by their visible branches. In the absent/absent case, both witnesses expose the shared middle history as an empty endpoint, so transitivity of hsame composes the empty endpoint presentations and rebuilds the absent classifier branch from h to u. In the absent/present case, the first witness presents the middle history as e, while the second presents the same middle history as $\mathsf{E1}(b)$ for some source payload b; symmetry and transitivity of hsame give $\mathsf{hsame}(\mathsf{e},\mathsf{E1}(b))$, contradicting empty-versus-one no-confusion. In the present/absent case, the first witness presents the middle history as $\mathsf{E1}(b)$ and the second presents it as e, so the same no-confusion argument applies with the endpoints exchanged.

In the present/present case, the two middle present endpoint witnesses give $\mathsf{hsame}(\mathsf{E1}(b),\mathsf{E1}(b'))$. One-constructor inversion identifies the middle payload presentations, so the two adjacent source-classifier witnesses have a common middle source after this identification. The field cert.equiv_trans then composes those adjacent witnesses into the required endpoint source-classifier witness.

For carrier transport, case-analyze the classifier witness; the absent branch gives the empty endpoint carrier, and the present branch stores the target payload carrier directly. The classifier endpoint transport field is exactly Theorem 33.34. Same-tag exactness, right-absent exactness, and mixed constructor separation are isolated in Theorems 33.37, 33.38, and 33.40. □

Theorem 33.36 (Tagged option history classifier self exactness)**.** *This theorem fits the ten-field template with traditional target self-classification for constructor-tagged* $\mathsf{Option}^{\uparrow}$*, BEDC source* BHist *carrying a source* NameCert*, carrier* TaggedOptionHistoryCarrier*, classifier* TaggedOptionHistoryClassifier*, stability obligation carrier-relative reflexivity together with endpoint carrier exposure, ledger visible absent/present tag, core theorem self-exactness, exactness predicate the displayed iff, standard bridge* none/some *self-classification, and status* ProofTarget*. For a BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, source certificate* $\mathsf{cert} : \mathsf{NameCert}(S,\sim_S)$*, and history* h*, the tagged option classifier at equal endpoints is exactly the tagged option carrier:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,h) \iff \mathsf{TaggedOptionHistoryCarrier}_S(h).$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier self exactness`

Proof. For the forward direction, apply Theorem 33.21 to the self-classifier witness $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim}$. The endpoint-carrier pair has identical endpoints, so projecting the left carrier gives $\mathsf{TaggedOptionHistoryCarrier}_S(h)$. Equivalently, splitting the self-classifier by visible branch gives either the absent carrier witness $\mathsf{hsame}(h, \mathsf{e})$ or a present payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$.

For the reverse direction, split the carrier witness. An absent carrier witness builds the absent-absent classifier branch. A present carrier witness gives a, $S(a)$, and $\mathsf{hsame}(h, \mathsf{E1}(a))$; use the reflexivity field of cert on $S(a)$ to build the present-present self-classifier branch. This is item 1 of Theorem 33.35 under the source certificate cert. □

Theorem 33.37 (Tagged option history same-tag exactness)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_S$, the classifier $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}$ from Definition 33.19 is exact on equal visible tags:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{e}, k) \iff \mathsf{hsame}(k, \mathsf{e})$$

and

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{E1}(h), \mathsf{E1}(k)) \iff \exists a, b.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h, a) \wedge \mathsf{hsame}(k, b) \wedge a \sim_S b.$$

Proof. For the absent endpoint, case-analyze the tagged classifier. The absent-absent branch stores $\mathsf{hsame}(k, \mathsf{e})$, while the present-present branch would contain $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(a))$ and is eliminated by empty-versus-one no-confusion. Conversely, $\mathsf{hsame}(k, \mathsf{e})$ combines with reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$ to build the absent-absent classifier branch.

For the present endpoint, the absent-absent branch would contain $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e})$ and is eliminated by one-versus-empty no-confusion. The present-present branch gives payload histories a, b, source witnesses $S(a)$ and $S(b)$, a source classifier witness $a \sim_S b$, and endpoint witnesses $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{E1}(a))$ and $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(b))$; one-constructor inversion turns these endpoint witnesses into $\mathsf{hsame}(h, a)$ and $\mathsf{hsame}(k, b)$. Conversely, such payload data rebuilds the present-present classifier branch after applying the one-constructor congruence for hsame to the two payload endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier same tag exactness`

Theorem 33.38 (Tagged option history right absent exactness)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and history h, the tagged option history classifier has the exact right absent endpoint:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, \mathsf{e}) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Proof. For the forward direction, case-analyze the tagged classifier witness. The absent-absent branch directly provides the endpoint witness $\mathsf{hsame}(h, \mathsf{e})$. The present-present branch would force $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(b))$ for some source payload b, contradicting BEDC history no-confusion between the empty endpoint and a one-extension endpoint. For the reverse direction, combine the assumed $\mathsf{hsame}(h, \mathsf{e})$ with reflexivity of hsame at e and rebuild the absent-absent classifier branch. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier right absent exactness`

Theorem 33.39 (Tagged option history left absent exactness)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and history k, the tagged option history classifier has the exact left absent endpoint:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{e}, k) \iff \mathsf{hsame}(k, \mathsf{e}).$$

Proof. Case-analyze the tagged classifier witness. The absent-absent branch directly stores $\mathsf{hsame}(k, \mathsf{e})$. The present-present branch would force $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(a))$ for a source payload a, contradicting BEDC history no-confusion between the empty endpoint and a one-extension endpoint. Conversely, pair $\mathsf{hsame}(k, \mathsf{e})$ with reflexivity of hsame at e and build the absent-absent branch. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier left absent exactness`

Theorem 33.40 (Tagged option history mixed-tag separation)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, both mixed constructor cases are impossible:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{e}, \mathsf{E1}(k)) \to \bot$$

and

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{E1}(h), \mathsf{e}) \to \bot.$$

Proof. Case-analyze the tagged classifier witness. For $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{e}, \mathsf{E1}(k))$, the absent-absent branch forces $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{e})$, while the present-present branch forces an endpoint witness from e to some $\mathsf{E1}(a)$. Both contradict BEDC history no-confusion between empty and one-extension constructors. The case $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(\mathsf{E1}(h), \mathsf{e})$ is identical with the visible endpoints exchanged: the absent-absent branch forces $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e})$, and the present-present branch forces an endpoint witness from e to a present endpoint. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier mixed tag separation`

Theorem 33.41 (Tagged option history carrier-level mixed-tag separation)**.** *This theorem fits the per-object template as follows: the traditional target is absent/present option tag separation; the BEDC source is* BHist *with* hsame*; the carrier is* $\mathsf{TaggedOptionHistoryCarrier}$*; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}$*; the stability obligation is mixed-tag no-confusion under carrier presentations; the ledger is the visible tag; the core theorem is this separation; exactness is absent-vs-present impossibility; the standard bridge is none/some; and the status is* $\mathsf{ProofTarget}$*. For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\mathsf{sameS} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$, histories h, k, and payload history a, the classifier cannot relate endpoints that are merely presented as opposite tags:*

$$\mathsf{hsame}(h, \mathsf{e}) \to S(a) \to \mathsf{hsame}(k, \mathsf{E1}(a)) \to \mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{sameS}}(h, k) \to \bot$$

and

$$S(a) \to \mathsf{hsame}(h, \mathsf{E1}(a)) \to \mathsf{hsame}(k, \mathsf{e}) \to \mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{sameS}}(h, k) \to \bot.$$

Proof. Case-analyze the tagged classifier witness. In the first direction, the absent-absent branch stores an endpoint witness from k to e; composing it with the displayed witness from k to $\mathsf{E1}(a)$ gives empty-versus-one no-confusion. The present-present branch stores an endpoint witness from h to some $\mathsf{E1}(b)$; composing it with the displayed witness from h to e again gives empty-versus-one no-confusion. The second direction is symmetric: the absent-absent branch contradicts the displayed present presentation of h, and the present-present branch contradicts the displayed absent presentation of k. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier carrier mixed tag separation`

Theorem 33.42 (Tagged option history classifier visible-branch inversion)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option branch inversion; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is branch determinism under visible endpoint presentations; the ledger is the visible absent/present tag plus payload; the core theorem is this inversion; exactness is no mixed branch and payload exposure; the standard bridge is* none/some*; and the status is* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k*, the classifier determines the visible branch presented by the left endpoint:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to \mathsf{hsame}(h,\mathsf{e}) \to \mathsf{hsame}(k,\mathsf{e})$$

and, for every payload history a*,*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to S(a) \to \mathsf{hsame}(h,\mathsf{E1}(a)) \to \exists a', b : \mathsf{BHist}.\ S(a') \wedge S(b) \wedge \mathsf{hsame}(a,a') \wedge a' \sim$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible branch inversion`

Proof. For the absent clause, case-analyze the tagged classifier witness. The absent-absent branch already stores $\mathsf{hsame}(k,\mathsf{e})$. The present-present branch stores $\mathsf{hsame}(h,\mathsf{E1}(a))$ for some source payload; composing it with the displayed witness $\mathsf{hsame}(h,\mathsf{e})$ gives empty-versus-one no-confusion.

For the present clause, case-analyze the classifier witness again. The absent-absent branch stores $\mathsf{hsame}(h,\mathsf{e})$ and contradicts the displayed present endpoint $\mathsf{hsame}(h,\mathsf{E1}(a))$. In the present-present branch, obtain payloads a' and b with $S(a')$, $S(b)$, $a' \sim_S b$, $\mathsf{hsame}(h,\mathsf{E1}(a'))$, and $\mathsf{hsame}(k,\mathsf{E1}(b))$. Compose the two presentations of h and invert the one-constructor equality to get $\mathsf{hsame}(a,a')$; the remaining fields are exactly the displayed witnesses. □

Theorem 33.43 (Tagged option classifier branch occupancy equivalence)**.** *This theorem fits the per-object template with traditional target tagged optional history branch occupancy, BEDC source* BHist *with* hsame*, carrier* TaggedOptionHistoryCarrier*, classifier* TaggedOptionHistoryClassifier*, stability obligation branch preservation, ledger visible absent/present occupancy, core theorem this equivalence, exactness constructor no-confusion, standard bridge* none/some*, and status* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, histories* h, k*, and witness* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$*, the visible branch occupancy is preserved:*

$$(\mathsf{hsame}(h,\mathsf{e}) \iff \mathsf{hsame}(k,\mathsf{e})) \wedge ((\exists a.\ S(a) \wedge \mathsf{hsame}(h,\mathsf{E1}(a))) \iff (\exists b.\ S(b) \wedge \mathsf{hsame}(k,\mathsf{E1}(b)))).$$

Proof. Case-analyze the tagged classifier witness. The absent-absent branch gives both empty endpoint witnesses and rules out either present endpoint by empty-versus-one no-confusion. The present-present branch projects the two source payload witnesses and their visible endpoint witnesses, and rules out either empty endpoint by the same no-confusion argument. □

Theorem 33.44 (Tagged option history semantic certificate target)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier, let* $\sim_S$ *be a source classifier on source histories, and assume a source* $\mathsf{NameCert}(S, \sim_S$ *). Then* $\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryCarrier}$ *is the tagged option history semantic certificate.*

Proof. Inhabitation is supplied by the absent carrier witness at e. The five stability fields of the semantic certificate, namely carrier-relative reflexivity, classifier symmetry, classifier transitivity, classifier-induced carrier transport, and classifier endpoint transport, are exactly Theorem 33.35 under the source NameCert. The pattern and ledger soundness fields are identity projections because all three semantic certificate predicates are the tagged option history carrier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier semanticNameCert`

Theorem 33.45 (Tagged option history visible semantic certificate)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier, let* $\sim_S$ *be a source classifier on source histories, and assume a source* $\mathsf{NameCert}(S, \sim_S)$*. Then* $\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryPattern}_S, \mathsf{TaggedOptionHistoryLedger}$ *holds.*

Proof. Apply Theorem 33.44 to obtain the carrier-only semantic certificate for the tagged option history carrier and classifier.

Keep the resulting core NameCert field unchanged. For the pattern and ledger soundness fields, apply Theorem 33.31: each carrier witness yields the visible tag pattern witness and the visible tag ledger witness. These three fields repack as the displayed SemanticNameCert. □

Theorem 33.46 (Tagged option classifier endpoint semantic fields)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier, let Pattern and Ledger be predicates on BEDC histories, let* $\sim_S$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathit{Pattern}, \mathit{Ledger}, \mathsf{TaggedOptionHistoryClassifier}_S$ *be a semantic certificate. For histories* h, k*, every classifier witness* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$ *supplies all four endpoint semantic fields:*

$$\mathit{Pattern}(h) \wedge \mathit{Ledger}(h) \wedge \mathit{Pattern}(k) \wedge \mathit{Ledger}(k).$$

Proof. Apply Theorem 33.21 to the classifier witness. This gives the endpoint carrier witnesses $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ and $\mathsf{TaggedOptionHistoryCarrier}_S(k)$.

Apply the pattern_sound field of cert to each endpoint carrier witness to obtain $\mathit{Pattern}(h)$ and $\mathit{Pattern}(k)$. Apply the ledger_sound field of cert to the same two endpoint carrier witnesses to obtain $\mathit{Ledger}(h)$ and $\mathit{Ledger}(k)$, and assemble the four conclusions in the displayed order. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier endpoint semantic fields`

Corollary 33.47 (Tagged option history certificate over hsame)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier. If a source* $\mathsf{NameCert}(S, \mathsf{hsame})$ *is supplied, then* $\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedO}$ *is the tagged option history semantic certificate with no separate compatibility hypothesis.*

Proof. First apply Theorem 33.44 with source classifier hsame to obtain the carrier-only semantic certificate.

The core NameCert field is reused unchanged. For the pattern and ledger soundness fields, apply Theorem 33.31: every carrier witness converts to the visible pattern witness and to the visible ledger witness. Repack these three fields as the displayed SemanticNameCert.

Apply Theorem 33.44 with source classifier hsame. The required compatibility input is exactly Theorem 33.27. □

Theorem 33.48 (Option standard bridge finite constructor surface)**.** *The public* $\mathsf{Option}^{\uparrow}$ *constructor surface consists exactly of the empty endpoint, the one-tag present endpoint, payload classifier transport, carrier-pattern-ledger equivalence, and the source* hsame *compatibility rows. In this surface, absence is represented only by histories presented as* e*, presence is represented only by histories presented as* $\mathsf{E1}(a)$ *for a source-admitted payload* a*, and the payload comparison exposed to a public consumer is precisely the transported source classifier row.*

No additional host option value, eliminator, recursion principle, or hidden branch coordinate is part of the public bridge. The absent and present rows are separated by constructor no-confusion, and the carrier, pattern, and ledger readings name the same finite two-branch history surface.

Proof. The tagged option semantic certificate is Corollary 33.47; it supplies the public carrier, pattern, ledger, and classifier over hsame.

Source compatibility is exactly Theorem 33.27. The carrier-pattern-ledger equivalence is Theorem 33.31, so a public endpoint has the same visible reading in all three rows.

The empty endpoint and one-tag present endpoint are isolated by the absent and present readback rows, while mixed presentations are excluded by Theorem 33.18 and Theorem 33.41. These no-confusion rows leave exactly the finite constructor surface stated above. □

Lean4 ✓ BEDC.Derived.OptionUp.OptionUp StdBridge

Theorem 33.49 (Tagged option public endpoint semantic scope)**.** *Let S : BHist → Prop be a source carrier, let $\sim_S$ be a source classifier on source histories, and assume a source* NameCert$(S, \sim_S)$*. The public* Option$^{\uparrow}$ *endpoint surface determined by this data consists exactly of the following finite information:*

1. *the empty absent endpoint, represented by* e *up to* hsame*;*
2. *the b1-tagged present endpoint, represented by* E1(a) *for a source payload a with $S(a)$;*
3. *the carrier, pattern, and ledger equivalence for the tagged option history surface;*
4. *the endpoint semantic fields obtained from* TaggedOptionHistoryClassifier$_{S,\sim_S}$*;*
5. *payload classifier transport through the present-present branch; and*
6. *the semantic certificate of Definition 33.44 and, for $\sim_S =$* hsame*, the certificate of Definition 33.47.*

No additional public endpoint data is supplied by a host 'Maybe' type, eliminators, a monad API, or a hidden payload projection.

Proof. The absent and present rows are the two constructor-shaped rows of TaggedOptionHistoryCarrier$_S$. The absent row is the empty BEDC history e up to hsame, and the present row is the one-mark BEDC history E1(a) with its source provenance $S(a)$.

Apply Theorem 33.31 to identify the carrier, pattern, and ledger descriptions row by row. The branch no-confusion theorems Definition 33.40 and Definition 33.41 rule out any mixed absent-present endpoint row.

For semantic data, apply Definition 33.46. This extracts the pattern and ledger fields on both classifier endpoints. Payload readback and transport are exactly the present endpoint rows supplied by Definition 33.15, Definition 33.16, and Definition 33.26.

It remains to justify that the list is exhaustive. A row of BHist satisfying the public carrier data is either the empty endpoint or a one-mark endpoint by the carrier rows; hsame only changes the chosen representative inside the same row; Cont uses the displayed classifier and therefore transports only along the same absent row or along a source-certified present payload row; and the source NameCert$(S, \sim_S)$ is the sole provenance for payload classification. Thus the public endpoint surface has exactly the six displayed components and nothing from a host option eliminator, monad interface, or payload projection outside the visible BEDC history. □

Theorem 33.50 (Option history classifier endpoint transport)**.** *If the optional history classifier relates h to k, and* hsame(h, h') *and* hsame(k, k')*, then it relates h' to k'.*

Proof. Unpack the classifier into its endpoint carrier witnesses and the stored hsame(h, k) link. Transport both carriers along the endpoint sameness links, then compose symmetry of hsame(h, h'), the stored link, and hsame(k, k'). □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier hsame transport`

Theorem 33.51 (Option history classifier symmetry)**.** *For any source predicate on BEDC histories, the optional history classifier is symmetric.*

Proof. Unpack the classifier into endpoint carrier witnesses and the stored $\mathsf{hsame}(h, k)$ link. Swap the endpoint witnesses and reverse the stored link by symmetry of hsame. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier symm`

Theorem 33.52 (Option history classifier transitivity)**.** *For every* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and* $h, k, r : \mathsf{BHist}$, $\mathsf{OptionHistoryClassifier}_S(h, k)$ *and* $\mathsf{OptionHistoryClassifier}_S(k, r)$ *imply* $\mathsf{OptionHistoryClassifier}_S(h, r)$.

Proof. Unpack the two classifier witnesses. The source and target carrier witnesses are preserved, and the underlying hsame links compose across the shared middle history. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier trans`

Theorem 33.53 (Option history classifier self exactness)**.** *For every BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and every history* h*, the nullable option classifier at equal endpoints is exactly the nullable option carrier:*

$$\mathsf{OptionHistoryClassifier}_S(h, h) \iff \mathsf{OptionHistoryCarrier}_S(h).$$

Proof. The forward direction projects the left endpoint carrier from the classifier witness. For the reverse direction, use the displayed carrier witness for both endpoints and close the classifier equality field by reflexivity of hsame at h. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier self exactness`

Theorem 33.54 (Option history classifier empty left iff)**.** *For any source predicate on BEDC histories, the optional history classifier relates the empty history on the left to* k *exactly when* k *is history-same as the empty history.*

Proof. In the forward direction, unpack the classifier and reverse its stored $\mathsf{hsame}(\mathsf{e}, k)$ link. In the reverse direction, both endpoint carrier witnesses are supplied by the absent-history branch, and the classifier link is the symmetric form of the assumed $\mathsf{hsame}(k, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier empty left iff`

Corollary 33.55 (Option history classifier empty right iff)**.** *For any source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and any history* h*, this is the right-endpoint symmetry corollary of Theorem 33.54 and Theorem 33.51, not a separate finite-data certificate target:*

$$\mathsf{OptionHistoryClassifier}_S(h, \mathsf{e}) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Proof. In the forward direction, apply classifier symmetry to reduce to $\mathsf{OptionHistoryClassifier}_S(\mathsf{e}, h)$, then use Theorem 33.54. In the reverse direction, use the reverse implication of Theorem 33.54 to obtain $\mathsf{OptionHistoryClassifier}_S(\mathsf{e}, h)$, then apply classifier symmetry. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier empty right iff`

Theorem 33.56 (Option history classifier empty endpoint equivalence)**.** *For any source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k*, if* $\mathsf{OptionHistoryClassifier}_S(h, k)$ *holds, then the two endpoints agree on absentness:*

$$\mathsf{hsame}(h, \mathsf{e}) \iff \mathsf{hsame}(k, \mathsf{e}).$$

Proof. Unpack the nullable option classifier into the two endpoint carrier witnesses and the stored endpoint equality $\mathsf{hsame}(h, k)$. The forward implication composes the symmetric stored equality with $\mathsf{hsame}(h, \mathsf{e})$. The reverse implication composes the stored equality with $\mathsf{hsame}(k, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier empty endpoint equivalence`

Definition 33.57 (Option source excludes empty)**.** *For a BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*,* $\mathsf{OptionSourceExcludesEmpty}(S)$ *means that every source history satisfying* S *is not history-same to the empty history:*

$$\mathsf{OptionSourceExcludesEmpty}(S) \quad :\iff \quad \forall h.\ S(h) \to \mathsf{hsame}(h, \mathsf{e}) \to \bot.$$

Lean4 (definition) `BEDC.Derived.OptionUp.OptionSourceExcludesEmpty`

Theorem 33.58 (Nullable option source transport exclusion)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* S *is closed under* hsame *transport and* $\mathsf{OptionSourceExcludesEmpty}(S)$ *holds, then a source history cannot transport along* $\mathsf{hsame}(h, k)$ *to a history* k *that is history-same to the empty history:*

$$S(h) \to \mathsf{hsame}(h, k) \to \mathsf{hsame}(k, \mathsf{e}) \to \bot.$$

Proof. Transport the source witness $S(h)$ across $\mathsf{hsame}(h, k)$ to obtain $S(k)$. The empty-exclusion predicate applied at k contradicts the displayed $\mathsf{hsame}(k, \mathsf{e})$ witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionSourceExcludesEmpty hsame transport absurd`

Theorem 33.59 (Nullable option absent-source separation)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$*,* $\mathsf{hsame}(h, \mathsf{e})$*, and* $S(h)$ *hold, then* $\bot$*.*

Proof. Apply the exclusion predicate to the source witness $S(h)$ and the displayed empty endpoint witness $\mathsf{hsame}(h, \mathsf{e})$. □

Per-object fit: traditional target nullable empty-or-source separation under an empty-excluding source, not constructor-tagged none/some readback; BEDC source BHist with hsame; carrier $\mathsf{OptionHistoryCarrier}_S$; classifier unused; status ProofTarget.

Theorem 33.60 (Nullable option carrier empty-or-source partition)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $\mathsf{OptionHistoryCarrier}_S(h)$ *hold, then*

$$(\mathsf{hsame}(h, \mathsf{e}) \wedge \neg S(h)) \vee (S(h) \wedge \neg \mathsf{hsame}(h, \mathsf{e})).$$

Proof. Case-analyze the carrier witness. In the empty branch, keep $\mathsf{hsame}(h, \mathsf{e})$ and use Theorem 33.59 to rule out $S(h)$. In the source branch, keep $S(h)$ and use the same theorem to rule out $\mathsf{hsame}(h, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier exclusive readback`

Theorem 33.61 (Nullable option absent iff not source)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $\mathsf{OptionHistoryCarrier}_S(h)$ *hold, then the absent branch is exactly the negation of the source branch:*

$$\mathsf{hsame}(h, \mathsf{e}) \iff \neg S(h).$$

Proof. For the forward implication, apply the empty-exclusion predicate to the source witness and the displayed empty endpoint. For the reverse implication, case-analyze the carrier witness: the empty branch gives $\mathsf{hsame}(h, \mathsf{e})$ directly, while the source branch contradicts the assumed negation of $S(h)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier absent not source iff`

Theorem 33.62 (Nullable option visible source exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, the nullable option carrier at a visible* $\mathsf{E1}$ *endpoint is exactly the source predicate at that same endpoint:*

$$\mathsf{OptionHistoryCarrier}_S(\mathsf{E1}(h)) \iff S(\mathsf{E1}(h)).$$

Proof. Unfold the nullable carrier into its empty-or-source disjunction. The empty branch would give $\mathsf{hsame}(\mathsf{E1}(h), \mathsf{e})$, contradicting constructor separation for $\mathsf{E1}$ and e. The remaining branch is exactly $S(\mathsf{E1}(h))$, and the reverse direction rebuilds the source side of the carrier disjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier e1 source iff`

Theorem 33.63 (Nullable option classifier one-headed left source exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k*, a nullable option classifier whose left endpoint is visibly one-headed is exactly the displayed source evidence for that endpoint, the right endpoint carrier, and the endpoint classifier:*

$$\mathsf{OptionHistoryClassifier}_S(\mathsf{E1}(h), k) \iff S(\mathsf{E1}(h)) \wedge \mathsf{OptionHistoryCarrier}_S(k) \wedge \mathsf{hsame}(\mathsf{E1}(h), k).$$

Proof. Unpack the classifier into its two endpoint carriers and stored $\mathsf{hsame}(\mathsf{E1}(h), k)$ link. The left carrier is converted by Theorem 33.62. Conversely, the displayed source evidence rebuilds the left carrier through the source branch, and the other two displayed fields repack the classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier e1 left source iff`

Theorem 33.64 (Nullable option zero-headed source exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, the nullable option carrier at a zero-headed endpoint is exactly the source predicate at that endpoint:*

$$\mathsf{OptionHistoryCarrier}_S(\mathsf{E0}(h)) \iff S(\mathsf{E0}(h)).$$

Proof. Unfold the nullable carrier into its empty-or-source disjunction. The empty branch would give $\mathsf{hsame}(\mathsf{E0}(h), \mathsf{e})$, contradicting constructor separation for $\mathsf{E0}$ and e. The source branch is the displayed predicate, and the reverse direction rebuilds the source side of the carrier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier e0 source iff`

Theorem 33.65 (Nullable option carrier branch alignment)**.** *For histories* h, k*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$*,* $\mathsf{OptionHistoryCarrier}_S(h)$*,* $\mathsf{OptionHistoryCarrier}_S(k)$*, and* $\mathsf{hsame}(h, k)$ *hold, then either both endpoints are empty or both lie in the source branch:*

$$(\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \vee (S(h) \wedge S(k)).$$

Proof. Case-analyze the two carrier witnesses. The empty–empty and source–source branches are immediate. In each mixed branch, transport the empty endpoint across $\mathsf{hsame}(h, k)$ and apply $\mathsf{OptionSourceExcludesEmpty}(S)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier hsame branch alignment`

Theorem 33.66 (Nullable option classifier exclusive readback)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $\mathsf{OptionHistoryClassifier}_S(h, k)$ *hold, then either both endpoints are the empty option branch and neither endpoint lies in* S*,*

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge \neg S(h) \wedge \neg S(k),$$

or both endpoints lie in the source branch, neither is empty, and the classifier keeps the endpoint equality:

$$S(h) \wedge S(k) \wedge \neg\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(h, k).$$

Proof. Unpack the classifier into the two endpoint carrier witnesses and the displayed $\mathsf{hsame}(h, k)$ equality. Apply Theorem 33.60 to both endpoint carriers. The empty–empty and source–source cases give the two displayed alternatives. The mixed cases contradict $\mathsf{OptionSourceExcludesEmpty}(S)$ after transporting the empty readback across $\mathsf{hsame}(h, k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier exclusive readback`

Theorem 33.67 (Option history classifier source-left exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $S(h)$ *hold, then the nullable option classifier from* h *to* k *is exactly source membership of* k *together with endpoint sameness:*

$$\mathsf{OptionHistoryClassifier}_S(h, k) \iff S(k) \wedge \mathsf{hsame}(h, k).$$

Proof. Unpack the classifier into endpoint carrier witnesses and the stored $\mathsf{hsame}(h, k)$ link. Since $S(h)$ excludes the empty branch, any empty carrier witness for k would transport across $\mathsf{hsame}(h, k)$ to contradict $\mathsf{OptionSourceExcludesEmpty}(S)$. Hence the right carrier is the source branch $S(k)$, and the stored classifier link supplies $\mathsf{hsame}(h, k)$. Conversely, $S(h)$ and $S(k)$ give the two source carrier branches, and the displayed $\mathsf{hsame}(h, k)$ link reassembles the classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier source left exactness`

Theorem 33.68 (Option history classifier zero-headed left source exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k*, the nullable option classifier from the zero-headed endpoint* $\mathsf{E0}(h)$ *to* k *is exactly source membership of the left endpoint together with the right carrier and endpoint sameness:*

$$\mathsf{OptionHistoryClassifier}_S(\mathsf{E0}(h), k) \iff S(\mathsf{E0}(h)) \wedge \mathsf{OptionHistoryCarrier}_S(k) \wedge \mathsf{hsame}(\mathsf{E0}(h), k).$$

Proof. Unpack the classifier into the two endpoint carrier witnesses and the stored hsame link. Apply Theorem 33.64 to the left carrier witness to read it as $S(\mathsf{E0}(h))$; the right carrier and endpoint link are already classifier fields. Conversely, the displayed three fields rebuild the classifier using the source side of the left carrier disjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier e0 left source iff`

Theorem 33.69 (Option history classifier source-right exactness)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $S(k)$ *hold, then the nullable option classifier from* h *to* k *is exactly source membership of* h *together with endpoint sameness:*

$$\mathsf{OptionHistoryClassifier}_S(h, k) \iff S(h) \wedge \mathsf{hsame}(h, k).$$

Proof. Unpack the classifier into endpoint carrier witnesses and the stored $\mathsf{hsame}(h, k)$ link. Since $S(k)$ excludes the empty branch, any empty carrier witness for h transports back across $\mathsf{hsame}(h, k)$ to contradict $\mathsf{OptionSourceExcludesEmpty}(S)$. Hence the left carrier is the source branch $S(h)$, and the stored classifier link supplies $\mathsf{hsame}(h, k)$. Conversely, $S(h)$ and $S(k)$ give the two source carrier branches, and the displayed $\mathsf{hsame}(h, k)$ link reassembles the classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier source right exactness`

Theorem 33.70 (Nullable option classifier absent-source separation)**.** *For any BEDC history source predicate* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{OptionSourceExcludesEmpty}(S)$ *and* $S(h)$ *hold, then both*

$$\neg\mathsf{OptionHistoryClassifier}_S(h, \mathsf{e})$$

and

$$\neg\mathsf{OptionHistoryClassifier}_S(\mathsf{e}, h)$$

hold.

Proof. For the right-empty classifier, use Corollary 33.55 to turn a classifier witness into $\mathsf{hsame}(h, \mathsf{e})$, then apply $\mathsf{OptionSourceExcludesEmpty}(S)$ to $S(h)$. For the left-empty classifier, use Theorem 33.54 to obtain the same $\mathsf{hsame}(h, \mathsf{e})$ witness and apply the same exclusion predicate. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier absent source separation`

Theorem 33.71 (Option classifier hsame symmetry)**.** *For typed option carrier values* $x, y : \mathsf{OptionCarrier}(\mathsf{BHist})$*, if* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, x, y)$ *holds, then* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, y, x)$ *holds.*

Proof. The proof case-analyzes typed none and some carrier values: the none–none branch returns the existing classifier witness, the mixed branches contradict the typed classifier clauses, and the some–some branch reverses the BEDC history witness by hsame symmetry. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame symm`

Theorem 33.72 (Option classifier hsame $e1$ map equivalence)**.** *For typed option carrier values* $x, y : \mathsf{OptionCarrier}(\mathsf{BHist})$*, mapping both visible payloads through the BEDC history constructor* E1 *preserves and reflects the specialized option classifier:*

$$\mathsf{OptionClassifierSpec}(\mathsf{hsame}, \mathsf{map}(\mathsf{E1}, x), \mathsf{map}(\mathsf{E1}, y)) \iff \mathsf{OptionClassifierSpec}(\mathsf{hsame}, x, y).$$

Proof. Case-analyze the two option endpoints. The none–none branch is unchanged, and the mixed branches contradict the typed classifier clauses. In the some–some branch, the forward direction uses injectivity of E1 up to hsame, and the reverse direction applies constructor congruence for E1. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame e1 map iff`

Theorem 33.73 (Option classifier hsame transport)**.** *For typed option carrier values* x, x', y, y' : OptionCarrier(BHist)*, if* OptionClassifierSpec(hsame, x, y)*,* OptionClassifierSpec(hsame, x, x')*, and* OptionClassifierSpe *hold, then* OptionClassifierSpec(hsame, x', y') *holds.*

Proof. Case-analyze the four option endpoints. Mixed none–some branches contradict one of the displayed classifier witnesses. The none–none branch reuses the existing typed classifier witness. In the some–some branch, compose hsame(x, x') symmetrically with hsame(x, y) and then with hsame(y, y'). □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame transport`

Theorem 33.74 (Option classifier hsame right determinacy)**.** *For histories* h, k, l : BHist*, if* OptionClassifierSpec(hsame, some(h), some(k)) *and* OptionClassifierSpec(hsame, some(h), some(l)) *hold, then* hsame(k, l) *holds.*

Proof. The two typed some–some classifier witnesses unfold to hsame(h, k) and hsame(h, l). Reverse the first witness by hsame symmetry and compose it with the second witness by hsame transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame right deterministic`

Theorem 33.75 (Option classifier hsame left determinacy)**.** *For histories* h, k, l : BHist*, if* OptionClassifierSpec(hsar *and* OptionClassifierSpec(hsame, some(l), some(k)) *hold, then* hsame(h, l) *holds.*

Proof. The two typed some–some classifier witnesses unfold to hsame(h, k) and hsame(l, k). Reverse the second witness by hsame symmetry and compose it with the first witness by hsame transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame left deterministic`

Theorem 33.76 (Option classifier hsame some-right confluence)**.** *For a typed option carrier value* x : OptionCarrier(BHist) *and histories* h, k : BHist*, if* OptionClassifierSpec(hsame, x, some(h)) *and* OptionClassifierSpec(hsame, x, some(k)) *hold, then* hsame(h, k) *holds.*

Proof. Case-analyze the shared endpoint x. If x is none, the first classifier witness is impossible because it relates none to a present endpoint. If $x =$ some(a), the two witnesses are hsame(a, h) and hsame(a, k); reverse the first and compose it with the second. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame some right confluence`

Theorem 33.77 (Option classifier hsame common-right confluence)**.** *For typed option carrier values* x, y : OptionCarrier(BHist) *and a history* a : BHist*, if both* x *and* y *classify to the same present endpoint,*

$$\text{OptionClassifierSpec}(\text{hsame}, x, \text{some}(a)) \quad \textit{and} \quad \text{OptionClassifierSpec}(\text{hsame}, y, \text{some}(a)),$$

then OptionClassifierSpec(hsame, x, y) *holds.*

Proof. Case-analyze x and y. Any none branch contradicts one of the displayed none–some classifier clauses. In the some–some branch, the two classifier witnesses are hsame links from the payloads of x and y to the common payload a; compose the first link with the symmetric of the second. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame common right confluence`

Theorem 33.78 (Option classifier hsame some-right inversion)**.** *For a typed option carrier value* $x : \mathsf{OptionCarrier}(\mathsf{BHist})$ *and a history* $b : \mathsf{BHist}$*, if* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, x, \mathsf{some}(b))$ *holds, then there is a history* $a : \mathsf{BHist}$ *such that* $x = \mathsf{some}(a)$ *and* $\mathsf{hsame}(a, b)$ *holds.*

Proof. Case-analyze x. The none branch contradicts the typed none–some classifier clause. In the some branch, the exposed payload is the required witness and the classifier clause is exactly the displayed hsame witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame some right inversion`

Theorem 33.79 (Option classifier hsame some-left inversion)**.** *For a typed option carrier value* $y : \mathsf{OptionCarrier}(\mathsf{BHist})$ *and a history* $a : \mathsf{BHist}$*, if* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, \mathsf{some}(a), y)$ *holds, then there is a history* $b : \mathsf{BHist}$ *such that* $y = \mathsf{some}(b)$ *and* $\mathsf{hsame}(a, b)$ *holds.*

Proof. Case-analyze y. The none branch contradicts the typed some–none classifier clause. In the some branch, the exposed payload is the required witness and the classifier clause is exactly the displayed hsame witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame some left inversion`

Theorem 33.80 (Option classifier hsame right payload determinacy)**.** *For a typed option carrier value* $x : \mathsf{OptionCarrier}(\mathsf{BHist})$ *and histories* $h, k : \mathsf{BHist}$*, if* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, x, \mathsf{some}(h))$ *and* $\mathsf{OptionClassifierSpec}(\mathsf{hsame}, x, \mathsf{some}(k))$ *hold, then* $\mathsf{hsame}(h, k)$ *holds.*

Proof. Case-analyze the shared left option value x. The none branch contradicts the typed option classifier clause for a none–some pair. In the some branch, the two classifier witnesses unfold to payload equalities from the same left payload to h and to k; reverse the first equality and compose it with the second by hsame transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec hsame right payload deterministic`

Theorem 33.81 (Option classifier symmetry from source symmetry)**.** *Let* A *be an entry type, let* $R : A \to A \to \mathsf{Prop}$ *be a source relation, and let* $x, y : \mathsf{OptionCarrier}(A)$*. If* $\forall a\, b.\ R(a, b) \to R(b, a)$ *and* $\mathsf{OptionClassifierSpec}(R, x, y)$ *hold, then* $\mathsf{OptionClassifierSpec}(R, y, x)$ *holds.*

Proof. Case-analyze x and y. The none–none case is immediate from the none classifier clause. The mixed none–some and some–none cases are impossible by the typed option classifier clauses. In the some–some case, apply the displayed source-symmetry hypothesis to the payload witness. □

Definition 33.82 (Option stability certificate)**.** *The stability certificate separates the typed optional classifier fields from the nullable history-envelope fields.*

For the typed OptionClassifierSpec *classifier, the fields are:*

1. *reflexivity for* none *and for* $\mathsf{some}(a)$ *from source reflexivity;*
2. *symmetry from a source-symmetry hypothesis, as in Theorem 33.81;*
3. *transitivity from a source-transitivity hypothesis;*
4. *no-confusion between* none *and* $\mathsf{some}(a)$ *in either order;*
5. *present inversion, exposing the source classifier witness from a* some–some *classification.*

The checked hsame *specialization is the Lean-marked instance of this symmetry field.*

For the nullable OptionHistoryCarrier *envelope and its* OptionHistoryClassifier*, the certificate fields are:*

1. *carrier transport across* hsame*;*
2. *endpoint transport for the classifier across* hsame *on both endpoints;*
3. *classifier symmetry;*
4. *classifier transitivity;*
5. *ledger transport across raw and visible endpoint* hsame *links.*

Lean4 ✓ `BEDC.Derived.OptionUp.option stability certificate fields`

Theorem 33.83 (Option classifier transitivity)**.** *Let A be an entry type, let $R : A \to A \to \mathsf{Prop}$ be a source classifier relation, and let $x, y, z : \mathsf{OptionCarrier}(A)$. If R is transitive and both $\mathsf{OptionClassifierSpec}(R, x, y)$ and $\mathsf{OptionClassifierSpec}(R, y, z)$ hold, then $\mathsf{OptionClassifierSpec}(R, x, z)$ holds.*

Lean4 ✓ `BEDC.Derived.OptionUp.OptionClassifierSpec trans`

Proof. Proceed by constructor case analysis on x, y, and z. The none–none–none case is immediate, mixed none–some cases contradict the classifier clauses, and the some–some–some case is exactly the supplied transitivity of R. □

Definition 33.84 (Option ledger policy)**.** *The ledger keeps a raw optional-history carrier witness and an* hsame *link from the raw history to the visible history.*

Lean4 (definition) `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy`

Theorem 33.85 (Option history ledger policy empty raw iff)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ and let $v : \mathsf{BHist}$. Then*

$$\mathsf{OptionHistoryLedgerPolicy}(S, \mathsf{Empty}, v) \iff \mathsf{hsame}(v, \mathsf{Empty}).$$

Proof. Unpack the ledger at raw history Empty. A ledger witness stores an $\mathsf{hsame}(\mathsf{Empty}, v)$ link, whose symmetry gives $\mathsf{hsame}(v, \mathsf{Empty})$. Conversely, an $\mathsf{hsame}(v, \mathsf{Empty})$ witness provides the empty carrier branch for the raw endpoint and, by symmetry, the ledger link from Empty to v. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy empty raw iff`

Theorem 33.86 (Option visible ledger carrier)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ and let $\rho, v : \mathsf{BHist}$. If source_transport $: \forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k)$ holds, then $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ implies $\mathsf{OptionHistoryCarrier}_S(v)$.*

Proof. Unpack the raw carrier and raw-to-visible hsame link. The absent case composes the symmetric link with the raw empty-history witness; the present case transports the raw source witness along the stored link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy visible carrier`

Theorem 33.87 (Option ledger policy source visible iff)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ and let $\rho, v : \mathsf{BHist}$. Assume*

$$\forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k).$$

If $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ holds, then

$$S(\rho) \iff S(v).$$

Proof. Unpack the ledger to obtain the raw-visible link $\mathsf{hsame}(\rho, v)$. The forward direction transports a raw source witness along this link. The reverse direction transports a visible source witness along the symmetric link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy source visible iff`

Theorem 33.88 (Option ledger visible source exactness)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *exclude the empty history, and let* $\rho, v : \mathsf{BHist}$*. If* $S(v)$ *holds, then the option ledger over* ρ *and* v *is exactly raw source membership together with the stored raw-visible equality:*

$$\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v) \iff S(\rho) \wedge \mathsf{hsame}(\rho, v).$$

Proof. Unpack the ledger into the raw option carrier and the raw-visible hsame link. If the raw carrier is absent, the link transports the empty witness to v, contradicting empty-source exclusion and $S(v)$. Hence the raw carrier is the source branch, and the link is the displayed equality. Conversely, the raw source witness rebuilds the source branch of the carrier and pairs it with the given raw-visible link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy visible source exactness`

Theorem 33.89 (Option ledger exclusive readback)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *exclude the empty history, and let* $\rho, v : \mathsf{BHist}$*. If* $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *holds, then exactly one branch is read back from the ledger:*

$$(\mathsf{hsame}(\rho, \mathsf{Empty}) \wedge \mathsf{hsame}(v, \mathsf{Empty}) \wedge \neg S(\rho) \wedge \neg S(v)) \vee (S(\rho) \wedge S(v) \wedge \neg \mathsf{hsame}(\rho, \mathsf{Empty}) \wedge \neg \mathsf{hsame}(v, \mathsf{Empty}) \wedge \mathsf{hsame}(\rho,$$

Proof. Unpack the ledger into the raw optional-history carrier and the raw-to-visible hsame link. The raw carrier readback gives either the absent branch or the source branch. In the absent branch, the stored link transfers the empty witness to the visible endpoint, and source exclusion rules out visible source membership. In the source branch, the stored hsame link identifies raw and visible endpoints, so the source witness transports by case analysis on that link; empty witnesses on either endpoint contradict source exclusion. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy exclusive readback`

Theorem 33.90 (Option raw-visible ledger classifier)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and let* $\rho, v : \mathsf{BHist}$*. If* $\mathit{source_transport} : \forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k)$ *holds, then* $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *implies* $\mathsf{OptionHistoryClassifier}_S(\rho, v)$*.*

Proof. Unpack the raw carrier witness and raw-to-visible hsame link. The visible carrier is given by Theorem 33.86, and the stored link supplies the classifier equality field. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy raw visible classifier`

Theorem 33.91 (Option ledger shared-visible raw classifier)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and let* $\rho, \rho', v : \mathsf{BHist}$*. Assume*

$$\forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k).$$

If both $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *and* $\mathsf{OptionHistoryLedgerPolicy}(S, \rho', v)$ *hold, then* $\mathsf{OptionHistoryClassifier}_S($ *holds.*

Proof. Convert both ledger witnesses to raw-visible classifiers by Definition 33.90. Reverse the second classifier by option classifier symmetry, and compose the two classifiers by option classifier transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy shared visible raw classifier`

Theorem 33.92 (Option ledger shared-raw visible classifier)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and let* $\rho, v, v' : \mathsf{BHist}$*. Assume*

$$\forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k).$$

If both $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *and* $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v')$ *hold, then* $\mathsf{OptionHistoryClassifier}_S($ *holds.*

Proof. Convert both ledger witnesses to raw-visible classifiers by Definition 33.90. Reverse the first classifier by option classifier symmetry, and compose it with the second raw-visible classifier by option classifier transitivity. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy shared raw visible classifier`

Theorem 33.93 (Option ledger classifier composition)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source predicate and let* $\rho, v, w : \mathsf{BHist}$*. Assume*

$$\forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k).$$

If $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *and* $\mathsf{OptionHistoryClassifier}_S(v, w)$ *hold, then* $\mathsf{OptionHistoryClassifier}_S(\rho, w)$ *holds.*

Proof. Apply Theorem 33.90 to the ledger policy using the displayed source-transport hypothesis to obtain $\mathsf{OptionHistoryClassifier}_S(\rho, v)$. Compose that raw-visible classifier with the visible-target classifier by Theorem 33.52, yielding $\mathsf{OptionHistoryClassifier}_S(\rho, w)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy classifier extension`

Theorem 33.94 (Option ledger classifier endpoint equivalence)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source predicate and let* $\rho, v, w : \mathsf{BHist}$*. Assume*

$$\forall h\, k.\ \mathsf{hsame}(h, k) \to S(h) \to S(k).$$

If $\mathsf{OptionHistoryLedgerPolicy}(S, \rho, v)$ *holds, then* $\mathsf{OptionHistoryClassifier}_S(\rho, w) \iff \mathsf{OptionHistoryClassifier}_S(v, w)$*. The ten-field fit is the traditional target of nullable optional ledger endpoint replacement, BEDC source* BHist *with* hsame*, carrier* OptionHistoryCarrier*, classifier* OptionHistoryClassifier*, stability given by raw-visible ledger classifier together with classifier symmetry and transitivity, ledger* OptionHistoryLedgerPol *core theorem this endpoint equivalence, exactness as classifier-invariant raw and visible endpoints, standard bridge the nullable option envelope readback, and status* ProofTarget*.*

Proof. Apply Theorem 33.90 to the ledger witness to obtain $\mathsf{OptionHistoryClassifier}_S(\rho, v)$. For the forward direction, use Theorem 33.51 to reverse this raw-visible classifier and then compose it with the assumed $\mathsf{OptionHistoryClassifier}_S(\rho, w)$ by Theorem 33.52. For the reverse direction, apply Theorem 33.93 directly to the ledger witness and the assumed visible-endpoint classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy classifier endpoint equivalence`

Theorem 33.95 (Option ledger history-same transport)**.** *If both raw histories and both visible histories are related by* hsame*, then an option ledger policy transports from the first raw-visible pair to the second.*

Proof. Unpack the ledger into its raw carrier witness and raw-to-visible history link. The raw carrier transports along the raw hsame witness. The transported raw-to-visible link is the composite of the symmetric raw link, the stored ledger link, and the visible link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy hsame transport`

Theorem 33.96 (Option ledger policy source monotonicity)**.** *Let S, T : BHist → Prop be source predicates and let ρ, v : BHist. If every S-source history is also a T-source history, then* OptionHistoryLedgerPolicy(S, *implies* OptionHistoryLedgerPolicy(T, ρ, v)*.*

Proof. Unpack the ledger into its raw optional-history carrier and raw-to-visible hsame link. The absent raw branch is kept unchanged. In the source raw branch, apply the source-monotonicity hypothesis to move the raw witness from S to T, and keep the same raw-to-visible link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy source monotonicity`

Theorem 33.97 (Option ledger policy transitivity)**.** *Let S : BHist → Prop and let ρ, m, v : BHist. If* OptionHistoryLedgerPolicy(S, ρ, m) *and* OptionHistoryLedgerPolicy(S, m, v) *hold, then* OptionHistoryLedgerPolicy(S *holds.*

Proof. Unpack both ledger witnesses. The first witness supplies the raw optional carrier for ρ and the link hsame(ρ, m); the second supplies the link hsame(m, v). Compose the two hsame links and keep the raw carrier witness from the first ledger. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryLedgerPolicy trans`

Theorem 33.98 (Concrete optional-history semantic certificate)**.** *For every source predicate S :* BHist → Prop*, the checked nullable certificate is* SemanticNameCert(OptionHistoryCarrier$_S$, OptionHistoryCarrier$_S$, Op *This semantic certificate does not assert constructor-discriminating history tags; that stronger tagged interface is the preceding paper-side target.*

Proof. For each S, the history carrier contains the empty history and any source history. The classifier records both endpoint envelope-carrier witnesses and the BEDC history equality witness. Reflexivity, symmetry, transitivity, and carrier transport are the corresponding hsame fields, while pattern and ledger soundness are carrier projections for the envelope. □

Corollary 33.99 (OptionUp certificate scope)**.** *Citations to Theorem 33.98 assert only the nullable empty-or-source history envelope. The constructor-tagged history certificate is Theorem 33.44, under a source* NameCert *and* TaggedOptionSourceHsameCompatible$(S, \sim_S)$*, or Corollary 33.47 in the* hsame *specialization. The typed* OptionClassifierSpec *is a separate host-option classifier; it is not a field that licenses the tagged history semantic certificate, and neither certificate implies the other carrier presentation without the bridge data of Definition 33.103.*

Theorem 33.100 (Tagged option history classifier right visible-branch inversion)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option branch inversion from the right endpoint; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the ledger is the visible absent/present tag plus payload; exactness is right-endpoint branch exposure; and the status is* ProofTarget*. For every BEDC history source carrier S :* BHist → Prop*, source classifier $\sim_S$, and histories h, k, the classifier determines the visible branch presented by the right endpoint:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k) \to \mathsf{hsame}(k, \mathsf{e}) \to \mathsf{hsame}(h, \mathsf{e})$$

and, for every payload history a,

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to S(a) \to \mathsf{hsame}(k,\mathsf{E1}(a)) \to \exists b,a' : \mathsf{BHist}.\ S(b) \wedge S(a') \wedge \mathsf{hsame}(a,a') \wedge b \sim_S$$

Proof. For the absent clause, case-analyze the tagged classifier witness. The absent-absent branch already stores $\mathsf{hsame}(h,\mathsf{e})$. The present-present branch stores $\mathsf{hsame}(k,\mathsf{E1}(a'))$ for some right payload; composing it with the displayed witness $\mathsf{hsame}(k,\mathsf{e})$ gives empty-versus-one no-confusion.

For the present clause, case-analyze the classifier witness again. The absent-absent branch stores $\mathsf{hsame}(k,\mathsf{e})$ and contradicts the displayed present endpoint $\mathsf{hsame}(k,\mathsf{E1}(a))$. In the present-present branch, obtain payloads b and a' with $S(b)$, $S(a')$, $b \sim_S a'$, $\mathsf{hsame}(h,\mathsf{E1}(b))$, and $\mathsf{hsame}(k,\mathsf{E1}(a'))$. Compose the two presentations of k and invert the one-constructor equality to get $\mathsf{hsame}(a,a')$; the remaining fields are exactly the displayed witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier right visible branch inversion`

Theorem 33.101 (Tagged option history classifier composite right visible readback)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier and let $\sim_S$ be a transitive source relation. For histories h,k,r, two tagged option classifier steps*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \quad \textit{and} \quad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(k,r)$$

determine the branch of h from the visible branch of the composite right endpoint r:

$$\mathsf{hsame}(r,\mathsf{e}) \Longrightarrow \mathsf{hsame}(h,\mathsf{e}),$$

and, for every payload history a,

$$S(a) \Longrightarrow \mathsf{hsame}(r,\mathsf{E1}(a)) \Longrightarrow \exists b,a' : \mathsf{BHist}.\ S(b) \wedge S(a') \wedge \mathsf{hsame}(a,a') \wedge b \sim_S a' \wedge \mathsf{hsame}(h,\mathsf{E1}(b)).$$

Proof. Compose the two tagged classifier steps using transitivity of $\sim_S$. The result is a single classifier step from h to r. Applying the right visible-branch inversion theorem to that composite step gives the absent readback clause and the present payload witness clause exactly as displayed. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier composite right visible readback`

Theorem 33.102 (Tagged option shared-left visible-right determinacy)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and histories h,k,r,a,b,c, two tagged option classifier witnesses with source classifier* hsame *and shared left endpoint h determine the visible right payloads whenever all three displayed endpoints are present:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h,k) \Longrightarrow \mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h,r) \Longrightarrow \mathsf{hsame}(h,\mathsf{E1}(a)) \Longrightarrow \mathsf{hsame}(k,\mathsf{E1}(b)) \Longrightarrow \mathsf{hsame}(r,\mathsf{E1}(c)) \Longrightarrow \mathsf{hsame}(b,c).$$

Proof. Case-analyze both classifier witnesses. Any absent branch contradicts the displayed present endpoint for h. In the present branches, the two stored presentations of h identify the two left payload witnesses with a by one-constructor inversion. The stored presentations of k and r identify the right payload witnesses with b and c. Since the classifier relation is hsame, transitivity along the two shared-left payload links gives $\mathsf{hsame}(b,c)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier hsame shared left visible right deterministic`

Definition 33.103 (Tagged option-to-nullable bridge hypotheses)**.** *For a BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_S$, define* $\mathsf{TaggedOptionToNullableBridge}(S, \sim_S)$ *to be the pair of conditions:*

1. *present endpoint closure: for all histories a, h, if $S(a)$ and* $\mathsf{hsame}(h, \mathsf{E1}(a))$*, then $S(h)$;*
2. *classifier hsame soundness: for all histories a, b, if $S(a)$, $S(b)$, and $a \sim_S b$, then* $\mathsf{hsame}(a, b)$*.*

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionToNullableBridge`

Theorem 33.104 (Tagged option-to-nullable classifier bridge)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier and let $\sim_S$ be a source classifier. If* $\mathsf{TaggedOptionToNullableBridge}(S, \sim_S)$ *holds, then for every history h,* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *implies* $\mathsf{OptionHistoryCarrier}_S(h)$*, and for every pair of histories h, k,* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$ *implies* $\mathsf{OptionHistoryClassifier}_S(h, k)$*.*

Proof. Unpack the bridge into present endpoint closure and classifier hsame soundness. For the carrier implication, case-analyze $\mathsf{TaggedOptionHistoryCarrier}_S(h)$. The absent branch is already the empty side of $\mathsf{OptionHistoryCarrier}_S(h)$. In the present branch, a payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$ descends to $S(h)$ by present endpoint closure, giving the source side of the nullable envelope.

For the classifier implication, use Theorem 33.131. In the absent-absent branch, the two empty endpoint witnesses give the nullable carriers and compose through e to obtain $\mathsf{hsame}(h, k)$. In the present-present branch, the carrier part descends for both endpoints by present endpoint closure. Classifier hsame soundness converts $a \sim_S b$ into $\mathsf{hsame}(a, b)$; applying $e1$ congruence gives $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(b))$, which composes with the endpoint witnesses $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$ to yield $\mathsf{hsame}(h, k)$. These are exactly the carrier and endpoint-equality fields of $\mathsf{OptionHistoryClassifier}_S(h, k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionToNullableBridge descent`

Theorem 33.105 (Tagged option history classifier induces nullable classification)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier. If two histories are classified by the tagged option classifier over* hsame*, then they are classified by the nullable option classifier whose source side is the tagged option history carrier:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h, k) \Longrightarrow \mathsf{OptionHistoryClassifier}_{\mathsf{TaggedOptionHistoryCarrier}_S}(h, k).$$

Proof. Analyze the tagged classifier. In the absent branch, both endpoints are history-same to e, so the nullable carriers use their absent clauses and the classifier equality composes through e. In the present branch, the payload witnesses a and b rebuild the tagged carrier source clauses for the nullable classifier. The endpoint equality is obtained by composing the left endpoint link, congruence of $\mathsf{E1}$ over $\mathsf{hsame}(a, b)$, and the reverse of the right endpoint link. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier option history classifier`

Theorem 33.106 (Tagged option-to-nullable hsame bridge criterion)**.** *The ten-field fit is: traditional target nullable comparison for constructor-tagged options; BEDC source* BHist *with* hsame*; carrier* TaggedOptionHistoryCarrier*; classifier* $\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}$*; stability obligation present-endpoint absorption; ledger visible absent/present tag; core theorem the bridge criterion; exactness one-way comparison to the nullable envelope; standard bridge none/some readback; status* ProofTarget*. Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier. If*

$$\forall a\, h.\ S(a) \to \mathsf{hsame}(h, \mathsf{E1}(a)) \to S(h),$$

then $\mathsf{TaggedOptionToNullableBridge}(S, \mathsf{hsame})$ *holds. Consequently, for every history* h*,*

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \to \mathsf{OptionHistoryCarrier}_S(h),$$

and for every pair of histories h, k*,*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h, k) \to \mathsf{OptionHistoryClassifier}_S(h, k).$$

Proof. Pair the displayed present-endpoint absorption hypothesis with identity classifier hsame soundness: from $S(a)$, $S(b)$, and $\mathsf{hsame}(a, b)$, the required history-sameness conclusion is the same witness. This gives $\mathsf{TaggedOptionToNullableBridge}(S, \mathsf{hsame})$. The two descent conclusions are exactly Theorem 33.104 specialized to the source classifier hsame. □

Theorem 33.107 (Tagged option public certificate compares with nullable envelope)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier, let* $\sim_S$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, and assume* $\mathsf{TaggedOptionToNullableBridge}(S, \sim_S)$*. Then the constructor-tagged public certificate*

$$\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryPattern}_S, \mathsf{TaggedOptionHistoryLedger}_S, \mathsf{Tagge}$$

holds, and the tagged carrier and classifier compare to the nullable envelope:

$$\forall h.\ \mathsf{TaggedOptionHistoryCarrier}_S(h) \to \mathsf{OptionHistoryCarrier}_S(h),$$

and

$$\forall h\, k.\ \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k) \to \mathsf{OptionHistoryClassifier}_S(h, k).$$

This is one-way compatibility of the constructor-tagged public certificate with the checked nullable envelope; it is not an equality of the two carrier presentations.

Proof. First apply Theorem 33.44 to the source certificate cert, obtaining the carrier-only tagged semantic certificate. The core certificate is reused unchanged.

For the displayed public certificate, use Theorem 33.31 to convert each tagged carrier witness into the visible pattern witness and the visible ledger witness, then repack the core certificate with these two soundness maps.

The two displayed implications are exactly Theorem 33.104, applied to the bridge hypothesis $\mathsf{TaggedOptionToNullableBridge}(S, \sim_S)$. The bridge records one-way carrier and classifier compatibility with the checked nullable envelope; it does not construct a nullable semantic certificate from the tagged one. □

Theorem 33.108 (Tagged option hsame public certificate descends to nullable envelope)**.** *The tenfield fit is: traditional target nullable comparison for a public option object; BEDC source* BHist *with* hsame*; carrier* $\mathsf{TaggedOptionHistoryCarrier}_S$*; classifier* $\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}$*; stability obligation present-endpoint absorption; ledger visible absent/present tag; core theorem the tagged* hsame *semantic certificate; exactness one-way comparison to the nullable envelope; standard bridge none/some readback; status* ProofTarget*. Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(S, \mathsf{hsame})$*, and assume*

$$\forall a\, h.\ S(a) \to \mathsf{hsame}(h, \mathsf{E1}(a)) \to S(h).$$

Then the constructor-tagged public certificate

$$\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryPattern}_S, \mathsf{TaggedOptionHistoryLedger}_S, \mathsf{Tagge}$$

holds, and the tagged carrier and classifier descend to the nullable envelope:

$$\forall h.\ \mathsf{TaggedOptionHistoryCarrier}_S(h) \to \mathsf{OptionHistoryCarrier}_S(h),$$

and

$$\forall h\, k.\ \mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h,k) \to \mathsf{OptionHistoryClassifier}_S(h,k).$$

Proof. The tagged certificate is the hsame certificate corollary Corollary 33.47, applied to cert. The displayed absorption hypothesis gives $\mathsf{TaggedOptionToNullableBridge}(S, \mathsf{hsame})$ by Theorem 33.106. Applying Theorem 33.104 to that bridge datum gives the carrier and classifier implications into the nullable envelope. □

Theorem 33.109 (Tagged option history classifier absent-branch equivalence)**.** *This theorem fits the per-object template as follows: the traditional target is option absent-branch agreement; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is branch determinism; the ledger is the visible absent/present tag; the core theorem is absentness equivalence; exactness is absent endpoints agree under classifier; the standard bridge is* none/some*; and the status is* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k*, classified endpoints agree on absentness:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to \big(\mathsf{hsame}(h,\mathsf{e}) \iff \mathsf{hsame}(k,\mathsf{e})\big).$$

Proof. Assume a classifier witness between h and k. The forward implication is exactly the absent clause of Theorem 33.42: from $\mathsf{hsame}(h,\mathsf{e})$, left visible-branch inversion gives $\mathsf{hsame}(k,\mathsf{e})$. The reverse implication is the absent clause of Theorem 33.100: from $\mathsf{hsame}(k,\mathsf{e})$, right visible-branch inversion gives $\mathsf{hsame}(h,\mathsf{e})$. These two implications assemble the displayed $\iff$. □

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier`
Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier absent branch equivalence`

Theorem 33.110 (Tagged option history classifier nonempty-branch equivalence)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k*, classified endpoints agree on nonempty branch occupancy:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to \big((\mathsf{hsame}(h,\mathsf{e}) \to \bot) \iff (\mathsf{hsame}(k,\mathsf{e}) \to \bot)\big).$$

Proof. The absent-branch equivalence gives $\mathsf{hsame}(h,\mathsf{e}) \iff \mathsf{hsame}(k,\mathsf{e})$. Compose each implication with the corresponding negated endpoint hypothesis: a proof of $\mathsf{hsame}(k,\mathsf{e})$ transports back to $\mathsf{hsame}(h,\mathsf{e})$, and a proof of $\mathsf{hsame}(h,\mathsf{e})$ transports forward to $\mathsf{hsame}(k,\mathsf{e})$. These two contrapositive maps assemble the displayed equivalence of nonempty branches. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier nonempty branch equivalence`

Theorem 33.111 (Tagged option history classifier present-branch equivalence)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, histories* h, k*, and classifier witness* $\gamma : \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$*, classified endpoints agree on presentness:*

$$\big(\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))\big) \iff \big(\exists b : \mathsf{BHist}.\ S(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b))\big).$$

Proof. For the forward implication, unpack a source payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$. Apply the present clause of Theorem 33.42 to the witness γ; the exposed right payload supplies $S(b)$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$, which is the required present witness for k. For the reverse implication, unpack a right payload b with $S(b)$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$. Apply the present clause of Theorem 33.100 to γ; the exposed left payload supplies $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$, giving the required present witness for h. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier present branch equivalence`

Theorem 33.112 (Tagged option left present endpoint fanout)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, r, a, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, r), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)),$$

then both right endpoints are present:

$$(\exists b : \mathsf{BHist}.\ S(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b))) \wedge (\exists c : \mathsf{BHist}.\ S(c) \wedge \mathsf{hsame}(r, \mathsf{E1}(c))).$$

Proof. Case-analyze the two tagged classifier witnesses. In either witness, an absent left branch contradicts the displayed present endpoint for h. The remaining present branch exposes its right payload together with the source witness and the right endpoint equation. Applying this argument to the witnesses ending at k and r gives the two conjuncts. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier left present endpoint fanout`

Theorem 33.113 (Tagged option left present payload fanout)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, r, a, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, r), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)),$$

then both right endpoints expose payloads whose source-side predecessors are the displayed payload a up to hsame*:*

$$\begin{aligned}&(\exists b, a' : \mathsf{BHist}.\ S(b) \wedge S(a') \wedge \mathsf{hsame}(a, a') \wedge a' \sim_S b \wedge \mathsf{hsame}(k, \mathsf{E1}(b)))\\ &\quad \wedge (\exists c, a'' : \mathsf{BHist}.\ S(c) \wedge S(a'') \wedge \mathsf{hsame}(a, a'') \wedge a'' \sim_S c \wedge \mathsf{hsame}(r, \mathsf{E1}(c))).\end{aligned}$$

Proof. Case-analyze the two tagged classifier witnesses. An absent left branch contradicts the visible equation $\mathsf{hsame}(h, \mathsf{E1}(a))$. In each present branch, the classifier exposes a source payload, a target payload, the source and target carrier witnesses, the relation between the payloads, and the target endpoint equation. Constructor inversion for $\mathsf{E1}$ identifies the displayed payload a with the exposed source payload. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier left present payload fanout`

Theorem 33.114 (Tagged option right present endpoint fanout)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, r, a, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(r, k), \qquad \mathsf{hsame}(k, \mathsf{E1}(a)),$$

then both left endpoints are present:

$$(\exists b : \mathsf{BHist}.\ S(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(b))) \wedge (\exists c : \mathsf{BHist}.\ S(c) \wedge \mathsf{hsame}(r, \mathsf{E1}(c))).$$

Proof. Case-analyze the two tagged classifier witnesses. In either witness, an absent right branch contradicts the displayed present endpoint for k. The remaining present branch exposes its left payload together with the source witness and the left endpoint equation. Applying this argument to the witnesses starting at h and r gives the two conjuncts. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier right present endpoint fanout`

Theorem 33.115 (Tagged option right present payload fanout)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, r, a, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(r,k), \qquad \mathsf{hsame}(k, \mathsf{E1}(a)),$$

then both left endpoints expose payloads whose target-side successors are the displayed payload a up to hsame*:*

$$\begin{aligned}&(\exists b, a' : \mathsf{BHist}.\ S(b) \wedge S(a') \wedge \mathsf{hsame}(a, a') \wedge b \sim_S a' \wedge \mathsf{hsame}(h, \mathsf{E1}(b)))\\ &\quad \wedge (\exists c, a'' : \mathsf{BHist}.\ S(c) \wedge S(a'') \wedge \mathsf{hsame}(a, a'') \wedge c \sim_S a'' \wedge \mathsf{hsame}(r, \mathsf{E1}(c))).\end{aligned}$$

Proof. Case-analyze the two tagged classifier witnesses. An absent right branch contradicts the visible equation $\mathsf{hsame}(k, \mathsf{E1}(a))$. In each present branch, the classifier exposes a left payload, a right payload, the two carrier witnesses, the relation between the payloads, and the left endpoint equation. Constructor inversion for $\mathsf{E1}$ identifies the displayed payload a with the exposed right payload. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier right present payload fanout`

Theorem 33.116 (Tagged option present-chain payload witness)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, histories h, k, r, and visible payloads a, c, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(k,r), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsar}$$

then there are payload histories a_0, b, b', c_0 such that

$$S(a_0) \wedge S(b) \wedge S(b') \wedge S(c_0) \wedge \mathsf{hsame}(a, a_0) \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(c_0, c) \wedge a_0 \sim_S b \wedge b' \sim_S c_0.$$

Proof. Case-analyze the two tagged classifier witnesses. Any absent branch contradicts one of the displayed visible present endpoints. In the remaining present-present branches, the two classifier witnesses expose payload legs $a_0 \sim_S b$ and $b' \sim_S c_0$. The two visible presentations of the shared middle endpoint k are identified by the $\mathsf{E1}$ constructor inversion principle, giving $\mathsf{hsame}(b, b')$; the endpoint witnesses for h and r similarly identify a with a_0 and c_0 with c. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier present chain payload witness`

Theorem 33.117 (Tagged option history classifier visible payload witnesses)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, histories h, k, a, b, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then there are payload witnesses $a', b' : \mathsf{BHist}$ such that

$$S(a') \wedge S(b') \wedge \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge a' \sim_S b'.$$

Proof. Case-analyze the tagged classifier witness. The absent branch contradicts the left visible endpoint. In the present branch, the classifier already exposes source payload witnesses a' and b' together with the two endpoint equations and $a' \sim_S b'$. Constructor inversion for the two displayed $\mathsf{E1}$ readings identifies a with a' and b with b'. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible payload witnesses`

Theorem 33.118 (Tagged option history classifier visible payload relation transport)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source relation* $R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and* hsame*-transport field for* R*, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,R}(h,k), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then the displayed payloads satisfy $R(a,b)$*.*

Proof. Apply Theorem 33.117 to obtain payload witnesses a', b' with $\mathsf{hsame}(a,a')$, $\mathsf{hsame}(b,b')$, and $R(a',b')$. The transport field for R, used against the symmetric payload identifications, moves $R(a',b')$ back to the displayed payloads $R(a,b)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible payload relation transport`

Theorem 33.119 (Tagged option hsame visible payload exactness)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k, a, b*, if the tagged option classifier uses* hsame *as its payload classifier and both endpoints are displayed as visible payloads,*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\mathsf{hsame}}(h,k), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then the displayed payloads are themselves hsame*:*

$$\mathsf{hsame}(a,b).$$

Proof. Case-analyze the tagged classifier witness. The absent branch contradicts the visible display of the left endpoint. In the present branch, invert the two displayed $\mathsf{E1}$ endpoint readings and compose the exposed hsame payload leg with those two constructor inversions. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier hsame visible payload exactness`

Theorem 33.120 (Tagged option history classifier presented payload exactness)**.** *For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, transitivity hypothesis* $\forall x\, y\, z : \mathsf{BHist}.\ x \sim_S y \to y \sim_S z \to x \sim_S z$*, and* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, if histories* h, k, a, b *satisfy*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k), \qquad S(a), \qquad S(b), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then the displayed payloads have the same source classifier:

$$a \sim_S b.$$

Proof. Use Theorem 33.34 to transport the classifier witness from (h,k) to the displayed endpoints $(\mathsf{E1}(a), \mathsf{E1}(b))$. Then apply Theorem 33.26 with the displayed transitivity hypothesis, the compatibility hypothesis, and the source witnesses $S(a)$ and $S(b)$. The resulting visible present exactness is precisely $a \sim_S b$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier presented payload exactness`

Theorem 33.121 (Tagged option common-left visible payload classification)**.** *For every source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_S$, assume $\sim_S$ is symmetric and transitive, and assume that* hsame *between source histories classified by S implies $\sim_S$. If two tagged option classifier witnesses share the same left endpoint,*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k), \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,r),$$

and the three visible endpoints expose source payloads

$$S(a), \quad S(b), \quad S(c), \qquad \mathsf{hsame}(h,\mathsf{E1}(a)), \quad \mathsf{hsame}(k,\mathsf{E1}(b)), \quad \mathsf{hsame}(r,\mathsf{E1}(c)),$$

then the two right visible payloads are classified:

$$b \sim_S c.$$

Proof. Case-analyze the two tagged option classifier witnesses. Any absent branch contradicts the visible left endpoint $\mathsf{hsame}(h,\mathsf{E1}(a))$. In the present-present case, the first classifier relates its exposed payloads and the second classifier relates its exposed payloads. Constructor inversion for the visible endpoints identifies the exposed left payloads with a and the right payloads with b and c. Compatibility converts these hsame identifications into source classification, and symmetry plus transitivity compose the chain $b \sim_S a \sim_S c$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier common left visible payload classification`

Theorem 33.122 (Tagged option present payload determinism)**.** *This theorem fits the per-object template as follows: the traditional target is present optional payload determinism; the BEDC source and carrier are* BHist *with* hsame *and the visible present endpoint; the classifier is unused; the stability obligation is constructor inversion; the ledger is the visible present payload; exactness says that the same visible present endpoint has the same payload; and the status is* ProofTarget*. For all BEDC histories $h, a, b : \mathsf{BHist}$, if*

$$\mathsf{hsame}(h,\mathsf{E1}(a)), \qquad \mathsf{hsame}(h,\mathsf{E1}(b)),$$

then the two displayed payloads are the same BEDC history:

$$\mathsf{hsame}(a,b).$$

Proof. Use symmetry of the first endpoint witness and compose it with the second endpoint witness to obtain $\mathsf{hsame}(\mathsf{E1}(a),\mathsf{E1}(b))$. The BEDC constructor inversion principle for the visible present constructor E1 then exposes the unique underlying payload, yielding $\mathsf{hsame}(a,b)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistory present payload determinism`

Theorem 33.123 (Tagged option history carrier e0 absurdity)**.** *For every source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and tail history t, the zero-tagged history cannot inhabit the tagged optional carrier:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(\mathsf{E0}(t)) \Longrightarrow \bot.$$

Proof. Case-analyze the tagged carrier witness. The absent branch asserts $\mathsf{hsame}(\mathsf{E0}(t),\mathsf{e})$, contradicting history no-confusion between E0 and the empty history. The present branch supplies a payload a with $S(a)$ and $\mathsf{hsame}(\mathsf{E0}(t),\mathsf{E1}(a))$, contradicting no-confusion between the two visible tags. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier e0 absurd`

Theorem 33.124 (Tagged option history classifier e0 endpoint absurdity)**.** *For every source carrier S, payload classifier Rel, tail history t, and opposite endpoint h, a zero-tagged endpoint cannot occur on either side of the tagged optional classifier:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,Rel}(\mathsf{E0}(t), h) \Longrightarrow \bot, \qquad \mathsf{TaggedOptionHistoryClassifier}_{S,Rel}(h, \mathsf{E0}(t)) \Longrightarrow \bot.$$

Proof. Split the two endpoint positions. In an absent branch, the zero-tagged endpoint would be hsame to e, contradicting no-confusion for E0 and the empty history. In a present branch, the same endpoint would be hsame to some E1 payload, contradicting no-confusion between the two visible tags. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier e0 endpoint absurd`

Theorem 33.125 (Tagged option history classifier empty-visible exclusion)**.** *For every source carrier S, payload classifier Rel, histories h, k, and payload histories a, b, a tagged optional classifier witness excludes an empty endpoint opposite a visible present endpoint:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,Rel}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{e}) \Longrightarrow \mathsf{hsame}(k, \mathsf{E1}(b)) \Longrightarrow \bot) \wedge (\mathsf{hsame}(k, \mathsf{e}) \Longrightarrow \mathsf{hsame}(h, \mathsf{E}$$

Proof. Case-analyze the classifier witness. In the absent branch both endpoints are hsame to e, so either visible endpoint contradicts constructor no-confusion between e and E1. In the present branch the stored visible endpoint on the opposite side gives the same empty-versus-present contradiction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier empty visible exclusion`

Theorem 33.126 (Tagged option history classifier transported e0 endpoint absurdity)**.** *For every source carrier S, payload classifier Rel, tail history t, and classified endpoints h, k, a tagged optional classifier witness rules out transport of either endpoint to the zero-tagged history:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,Rel}(h, k) \Longrightarrow \neg\mathsf{hsame}(h, \mathsf{E0}(t)) \wedge \neg\mathsf{hsame}(k, \mathsf{E0}(t)).$$

Proof. Case-analyze the classifier witness. In the absent branch, composing the transported endpoint equality with the empty endpoint witness would identify $\mathsf{E0}(t)$ with e. In the present branch, composing with the visible endpoint witness would identify $\mathsf{E0}(t)$ with an E1 payload. Both cases contradict BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier endpoint transport e0 absurd`

Theorem 33.127 (Tagged option history classifier zero-endpoint exclusion)**.** *For every source carrier S, payload classifier Rel, histories h, k, and tail history t, a tagged optional classifier witness excludes zero-tagged histories at both endpoints up to* hsame*:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,Rel}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E0}(t)) \Longrightarrow \bot) \wedge (\mathsf{hsame}(k, \mathsf{E0}(t)) \Longrightarrow \bot).$$

Proof. Case-analyze the classifier witness. In the absent branch, either endpoint is hsame to e, so any additional identification with $\mathsf{E0}(t)$ contradicts no-confusion between E0 and e. In the present branch, the endpoint is hsame to some E1 payload, and the additional identification with $\mathsf{E0}(t)$ contradicts no-confusion between the two visible tags. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier zero endpoint exclusion`

Theorem 33.128 (Tagged option history carrier exclusive branch partition)**.** *This theorem fits the per-object template as follows: the traditional target is tagged optional carrier readback; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is unused; the stability obligation is constructor no-confusion; the ledger is the visible absent/present tag with payload; the core theorem is exclusive carrier partition; exactness is absent-or-present coverage with disjointness; the standard bridge is* none/some*; and the status is* ProofTarget*. For every source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ *holds, then*

$$(\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \vee (\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \neg\mathsf{hsame}(h, \mathsf{e})).$$

Proof. Case-analyze the tagged carrier witness. In the absent branch, the carrier witness gives $\mathsf{hsame}(h, \mathsf{e})$; any present payload witness $S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))$ contradicts Lemma 33.18. In the present branch, the carrier witness gives a payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$; the same constructor-separation lemma rules out $\mathsf{hsame}(h, \mathsf{e})$. These two carrier constructors give the displayed coverage and disjointness. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier exclusive branch partition`

Theorem 33.129 (Tagged option history carrier exclusive branch iff)**.** *For every source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, tagged option carrier membership is equivalent to the disjoint absent-or-present branch description:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \iff (\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \vee (\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h$$

Proof. The forward direction is Theorem 33.128. For the converse, the absent branch directly supplies the empty tagged carrier constructor. In the present branch, the displayed source payload and visible endpoint equality directly supply the present tagged carrier constructor. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier exclusive branch iff`

Theorem 33.130 (Tagged option absent endpoint exactly excludes present payloads)**.** *For every source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, a tagged option carrier witness identifies* h *with the absent endpoint exactly when no present payload witness can classify the same visible endpoint:*

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \implies \Big(\mathsf{hsame}(h, \mathsf{e}) \iff \neg\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))\Big).$$

Proof. Assume the absent endpoint equality. Any present payload presentation would compose with it to identify e with an $\mathsf{E1}$-headed history, contradicting generated-history no-confusion. Conversely, split the tagged carrier witness. The absent branch gives $\mathsf{hsame}(h, \mathsf{e})$ directly, while the present branch is ruled out by the displayed no-present hypothesis. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier absent no present iff`

Theorem 33.131 (Tagged option history classifier branch partition)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option classifier branch partition; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the ledger is the visible absent/present tag with present payload evidence; exactness is complete two-branch endpoint exposure; and the status is* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, histories* h, k*, and classifier witness* $\gamma : \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$*, the endpoints fall into one of the two tagged branches:*

$$(\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \vee (\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge a \sim_S b).$$

Proof. Case-analyze the classifier witness γ. The absent-absent constructor directly provides $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$, giving the left branch. The present-present constructor provides payload histories a, b, source witnesses $S(a)$ and $S(b)$, endpoint witnesses $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$, and the source classifier witness $a \sim_S b$, giving the right branch. These are the only constructors of the tagged classifier witness, so the displayed disjunction exhausts the endpoints. □

Theorem 33.132 (Tagged option history classifier branch exactness)**.** *For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_S$, and histories h, k, the tagged option classifier is exactly the disjoint absent-or-present constructor analysis:*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \iff (\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e})) \vee (\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))$$

Proof. For the forward direction, apply Theorem 33.131 to the displayed classifier witness. For the reverse direction, split the displayed disjunction. In the absent branch, use the absent-absent classifier constructor with the two hsame endpoint witnesses. In the present branch, unpack a, b, the two source-carrier witnesses, the two visible present endpoint witnesses, and the source classifier witness $a \sim_S b$, then rebuild the present-present classifier constructor. These two constructors exhaust the tagged classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier branch exactness`

Theorem 33.133 (Tagged option public certificate branch exactness)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier, let $\sim_S$ be a source classifier, let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ be a source certificate, and assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. Then*

$$\mathsf{SemanticNameCert}(\mathsf{TaggedOptionHistoryCarrier}_S, \mathsf{TaggedOptionHistoryPattern}_S, \mathsf{TaggedOptionHistoryLedger}_S, \mathsf{Tagge}$$

holds, and for all histories h, k and every classifier witness $\gamma : \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$, both endpoints expose all three public semantic fields:

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \wedge \mathsf{TaggedOptionHistoryCarrier}_S(k) \wedge \mathsf{TaggedOptionHistoryPattern}_S(h) \wedge \mathsf{TaggedOptionH}$$

and the classifier is exactly the concrete absent-or-present branch disjunction:

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \iff (\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e})) \vee (\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))$$

Proof. Apply Theorem 33.44 to cert and the displayed source hsame compatibility hypothesis to obtain the public semantic certificate. For a classifier witness γ, apply Theorem 33.21 to obtain the two endpoint carrier witnesses. Convert those carrier witnesses to the visible pattern and ledger witnesses with Theorem 33.31. The concrete absent-or-present equivalence is exactly Theorem 33.132. □

Theorem 33.134 (Tagged option public branch readback is single-valued)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier, let $\sim_S$ be a source classifier. For all histories h, k and every classifier witness $\gamma : \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k)$, the public tag readback is exclusive and single-valued:*

$$(\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e}) \wedge \neg(\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \wedge \neg(\exists b : \mathsf{BHist}.\ S(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)))) \vee (\exists a, b : \mathsf{BHi}$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier public branch readback single valued`

Proof. Apply Theorem 33.132 to the witness γ and split the resulting two concrete branches.

In the absent branch, the branch data give $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. A source-present endpoint for h would combine $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$ with $\mathsf{hsame}(h, \mathsf{e})$, contradicting Lemma 33.18. The same argument at k rules out every source-present endpoint on the right side. This is the left disjunct.

In the present branch, unpack a, b with $S(a)$, $S(b)$, $\mathsf{hsame}(h, \mathsf{E1}(a))$, $\mathsf{hsame}(k, \mathsf{E1}(b))$, and $a \sim_S b$. Lemma 33.18 rules out $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$ from the two visible present endpoints.

It remains to show single-valuedness of the payload readback. Given any other source-present witnesses a' for h and b' for k, apply Theorem 33.122 first to $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h, \mathsf{E1}(a'))$, and then to $\mathsf{hsame}(k, \mathsf{E1}(b))$ and $\mathsf{hsame}(k, \mathsf{E1}(b'))$. The two payload equalities give $\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b')$, completing the right disjunct. □

Theorem 33.135 (Tagged option source weakening)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option source weakening; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability field is source monotonicity for tagged endpoints; the ledger is the visible absent/present tag with present payload evidence; the core theorem is tagged source weakening; exactness is endpoint witness preservation; the standard bridge is constructor tagged option syntax; and the status is* ProofTarget*. Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, and let $\sim_S$ and $\sim_T$ be source classifiers. Assume*

$$\forall h.\ S(h) \to T(h)$$

and

$$\forall a\, b.\ S(a) \to S(b) \to a \sim_S b \to a \sim_T b.$$

Then for every history h,

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \to \mathsf{TaggedOptionHistoryCarrier}_T(h),$$

and for every pair of histories h, k,

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k) \to \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(h, k).$$

Proof. For the carrier implication, split the tagged carrier witness. The absent case has no source payload and is reused unchanged. The present case exposes a payload history a, a source witness $S(a)$, and the endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(a))$; apply the source-carrier implication to obtain $T(a)$ and reuse the same endpoint witness.

For the classifier implication, split the tagged classifier witness. The absent-absent case is reused unchanged. In the present-present case, keep the payload histories a, b and both endpoint witnesses. Apply the source-carrier implication to the two source witnesses to obtain $T(a)$ and $T(b)$, and apply the source-classifier implication to the same two source witnesses and the given $a \sim_S b$ witness to obtain $a \sim_T b$. These fields rebuild the tagged classifier witness for T and $\sim_T$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistorySource weakening`

Theorem 33.136 (Tagged option semantic fields source weakening)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, and let $\sim_S$ and $\sim_T$ be source classifiers. Assume*

$$\forall h.\ S(h) \to T(h)$$

and

$$\forall a\, b.\ S(a) \to S(b) \to a \sim_S b \to a \sim_T b.$$

Then for every history h, the three unary tagged option semantic fields transport from S to T:

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \to \mathsf{TaggedOptionHistoryCarrier}_T(h),$$

$$\mathsf{TaggedOptionHistoryPattern}_S(h) \to \mathsf{TaggedOptionHistoryPattern}_T(h),$$

and

$$\mathsf{TaggedOptionHistoryLedger}_S(h) \to \mathsf{TaggedOptionHistoryLedger}_T(h).$$

The binary classifier field transports separately:

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,k) \to \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(h,k).$$

Proof. Apply Theorem 33.135 to the displayed source-carrier and source-classifier implications. This gives the carrier transport and classifier transport directly. For the pattern field, use Theorem 33.31 to convert a $\mathsf{TaggedOptionHistoryPattern}_S(h)$ witness into the S-carrier witness, transport that carrier witness from S to T, and convert the resulting T-carrier witness back to $\mathsf{TaggedOptionHistoryPattern}_T(h$ The ledger field is identical, using the ledger half of the same carrier-pattern-ledger equivalence. □

Definition 33.137 (Tagged option payload descent relation)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

The relation $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ means that the tagged endpoint h over S is carried to the tagged endpoint k over T by preserving the visible option tag:

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k) \iff (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \vee (\exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(h, \mathsf{E1}(a$$

The displayed $T(\delta.\mathsf{map}(a))$ witness is the carrier preservation field ρ_δ applied to the exposed source payload.

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionMapRel`

Theorem 33.138 (Tagged option map relation visible branch equivalence)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-tagged option payload descent readback; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is tagged option histories; the classifier is the transported tagged option classifier; the stability obligation is visible branch preservation; the ledger is the absent/present tag with the transported payload; the core theorem is this branch equivalence; exactness is branch inversion; the standard bridge is none/some map readback; and the status is* ProofTarget*. Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S, \sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and let*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For all histories h, k, every witness $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ implies both

$$\mathsf{hsame}(h, \mathsf{e}) \iff \mathsf{hsame}(k, \mathsf{e})$$

and

$$(\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \iff (\exists b : \mathsf{BHist}.\ T(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b))).$$

Proof. Split the map-relation witness into its visible branches.

In the absent branch, both endpoint presentations are e. Thus absentness is preserved in either direction, and any present payload witness would identify an empty presentation with a one-constructor presentation, contradicting constructor no-confusion.

In the present branch, the exposed source payload a supplies the left present witness, while $\delta.\mathsf{map}(a)$ together with ρ_δ supplies the right present witness. Either endpoint being absent would again identify e with a stored one-constructor presentation, so absentness agrees in both directions. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel visible branch equivalence`

Theorem 33.139 (Tagged option map relation visible payload readback)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S, \sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and let*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories h, k, u, v, assume

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k).$$

If the visible source endpoint is presented by u and the visible target endpoint is presented by v,

$$S(u) \wedge \mathsf{hsame}(h, \mathsf{E1}(u)) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(v)),$$

then there is an internal transported payload $a : \mathsf{BHist}$ such that

$$S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(u, a) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(a)).$$

Proof. Split the map-relation witness into its absent and present branches.

In the absent branch, the same source endpoint h is presented both as e and as $\mathsf{E1}(u)$. Constructor no-confusion rules out this branch, so no visible source payload can coexist with the absent map witness.

In the present branch, the map relation exposes a payload a with $S(a)$, $T(\delta.\mathsf{map}(a))$, $\mathsf{hsame}(h, \mathsf{E1}(a))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$.

Invert the two one-constructor presentations of h to obtain $\mathsf{hsame}(u, a)$. Invert the two one-constructor presentations of k to obtain $\mathsf{hsame}(v, \delta.\mathsf{map}(a))$. The exposed carrier witnesses and these two payload equalities package the required internal transported payload. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel visible payload readback`

Theorem 33.140 (Tagged option identity payload descent exactness)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-tagged option identity map; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier *specialized to* hsame*; the stability obligation is identity preservation of visible absent and present endpoints; the ledger is the visible tag and unchanged payload; the core theorem is identity exactness for* TaggedOptionMapRel*; exactness is branch analysis; the standard bridge is none/some identity readback; and the status is* ProofTarget*. Let $S : \mathsf{BHist} \to \mathsf{Prop}$. Write δ_{id} for the descent certificate*

$$\delta_{\mathsf{id}} : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{hsame})$$

whose map field is $a \mapsto a$ and whose respects field sends p to p. Let

$$\rho_{\mathsf{id}} : \forall a.\ S(a) \to S(\delta_{\mathsf{id}}.\mathsf{map}(a))$$

be the preservation field that reuses the source witness. Then for all histories h, k,

$$\mathsf{TaggedOptionMapRel}_{\delta_{\mathsf{id}}}(S, S, \rho_{\mathsf{id}}; h, k) \iff \mathsf{TaggedOptionHistoryCarrier}_S(h) \wedge \mathsf{hsame}(h, k).$$

Proof. For the forward direction, split the map relation by its visible branch.

In the absent branch, the witness gives $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. The first field builds the absent carrier for h. Composing $\mathsf{hsame}(h, \mathsf{e})$ with symmetry of $\mathsf{hsame}(k, \mathsf{e})$ gives $\mathsf{hsame}(h, k)$.

In the present branch, the witness exposes a payload a with $S(a)$, $S(\delta_{\mathsf{id}}.\mathsf{map}(a))$, $\mathsf{hsame}(h, \mathsf{E1}(a))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta_{\mathsf{id}}.\mathsf{map}(a)))$. Since $\delta_{\mathsf{id}}.\mathsf{map}(a)$ is a, the first payload witness and the source endpoint presentation build the present carrier for h, and composing the two endpoint presentations gives $\mathsf{hsame}(h, k)$.

For the reverse direction, split the carrier of h by its tagged branch and use the assumed $\mathsf{hsame}(h, k)$.

In the absent branch, compose symmetry of $\mathsf{hsame}(h, k)$ with the absent presentation of h to obtain $\mathsf{hsame}(k, \mathsf{e})$. Together with the original absent presentation, this is the absent branch of $\mathsf{TaggedOptionMapRel}_{\delta_{\mathsf{id}}}$.

In the present branch, keep the exposed payload a and the witness $S(a)$. The field ρ_{id} supplies the target payload witness, and composing symmetry of $\mathsf{hsame}(h, k)$ with $\mathsf{hsame}(h, \mathsf{E1}(a))$ gives $\mathsf{hsame}(k, \mathsf{E1}(a))$. Because the identity descent leaves a unchanged, these fields build the present branch of $\mathsf{TaggedOptionMapRel}_{\delta_{\mathsf{id}}}$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel identity carrier hsame iff`

Theorem 33.141 (Tagged option map relation target carrier)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories $h, k : \mathsf{BHist}$, every concrete map-relation witness

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

exposes the target endpoint as a tagged option carrier:

$$\mathsf{TaggedOptionHistoryCarrier}_T(k).$$

Proof. Split the map-relation witness. In the absent branch, the target endpoint is presented as e, so it satisfies the absent carrier branch. In the present branch, the witness exposes a payload a with $T(\delta.\mathsf{map}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$, which are exactly the payload and endpoint fields of the present carrier branch for T. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel target carrier`

Theorem 33.142 (Tagged option payload descent preserves classification)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories h, h', k, k', if

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h'),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

$$\mathsf{cert} : \mathsf{NameCert}(S, \sim_S),$$

and

$$\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. Split the source classifier witness by its tagged branch and split the two endpoint map-relation witnesses. In the absent classifier branch, both source endpoints are presented as e. A present map-relation branch would also present one of those endpoints as $\mathsf{E1}(a)$, contradicting empty-versus-one no-confusion; hence both map-relation witnesses are absent branches, and the target classifier is rebuilt by the absent-absent constructor for k and k'.

In the present classifier branch, the source witness exposes payloads a, b with $S(a)$, $S(b)$, endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h', \mathsf{E1}(b))$, and classifier witness $a \sim_S b$. The absent map-relation branch is impossible by the same constructor no-confusion argument, so the two map-relation witnesses expose payloads c, d with target endpoints $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(c)))$ and $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(d)))$. Endpoint inversion between $\mathsf{E1}(a)$ and $\mathsf{E1}(c)$, and between $\mathsf{E1}(b)$ and $\mathsf{E1}(d)$, gives source history sameness between the corresponding payloads. Source hsame compatibility turns these sameness witnesses into source classifier witnesses; symmetry and transitivity from cert compose them with $a \sim_S b$ to obtain $c \sim_S d$. Applying $\delta.\mathsf{respects}$ yields $\delta.\mathsf{map}(c) \sim_T \delta.\mathsf{map}(d)$, while ρ_δ supplies both target carrier witnesses. These fields rebuild the present-present target classifier for k and k'. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel preserves classification`

Theorem 33.143 (Tagged option payload descent target branch readback)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories h, h', k, k', assume a source classifier witness

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h'),$$

the two map-relation witnesses

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

a source certificate

$$\mathsf{cert} : \mathsf{NameCert}(S, \sim_S),$$

and

$$\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S).$$

Then the target endpoints carry both the transported classifier and the concrete absent-or-present branch exposure:

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k') \wedge \big((\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})) \vee \exists c, d : \mathsf{BHist}.\ T(c) \wedge T(d) \wedge \mathsf{hsame}(k, \mathsf{E1}(c)$$

Proof. First apply Theorem 33.142 to the source classifier witness, the two map-relation witnesses, the source certificate, and source hsame compatibility. This gives $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

Next apply Theorem 33.132 with carrier T, classifier $\sim_T$, and endpoints k, k' to the transported classifier. The forward direction of that exactness theorem gives exactly the displayed target branch disjunction. Conjoining the transported classifier with this branch exposure gives the stated target readback. □

Theorem 33.144 (Tagged option map relation common-target source classification)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume reflected source readback from visible target equality,

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For histories h, h', k*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h').$$

Proof. Split the two map-relation witnesses. If both witnesses are absent, their source endpoint presentations are $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(h', \mathsf{e})$, so the absent-absent tagged classifier branch gives the conclusion.

In a mixed branch, one witness presents the shared target endpoint k as e, while the other presents it as $\mathsf{E1}(\delta.\mathsf{map}(b))$ for some payload b. Symmetry and transitivity of hsame compose these two presentations into $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(\delta.\mathsf{map}(b)))$ or its reverse, contradicting generated-history empty-versus-one no-confusion. Hence mixed branches cannot occur.

In the present/present branch, the two witnesses expose payloads a, b with $S(a)$ and $S(b)$, source endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h', \mathsf{E1}(b))$, and target endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$. Composing the two target endpoint presentations and inverting the common one-step constructor yields $\mathsf{hsame}(\delta.\mathsf{map}(a), \delta.\mathsf{map}(b))$.

Reflected source readback converts this visible target equality into $a \sim_S b$. Together with the stored source carrier and endpoint witnesses, this rebuilds the present-present source classifier for h and h'. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel common target source classification`

Theorem 33.145 (Tagged option map relation common-target visible source completion)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*. Assume reflected source readback from visible target equality,*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S),$$

and assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For histories* h, h', k*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$$

then the two source endpoints complete visible payload readback in both directions:

1. *for every visible payload* $x : \mathsf{BHist}$*, if* $S(x)$ *and* $\mathsf{hsame}(h, \mathsf{E1}(x))$*, then there exists* $y : \mathsf{BHist}$ *such that*
$$S(y) \wedge \mathsf{hsame}(h', \mathsf{E1}(y)) \wedge x \sim_S y;$$
2. *for every visible payload* $y : \mathsf{BHist}$*, if* $S(y)$ *and* $\mathsf{hsame}(h', \mathsf{E1}(y))$*, then there exists* $x : \mathsf{BHist}$ *such that*
$$S(x) \wedge \mathsf{hsame}(h, \mathsf{E1}(x)) \wedge x \sim_S y.$$

Proof. Apply Theorem 33.144 to the two common-target map-relation witnesses and the reflected source readback hypothesis. This gives $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$.

For a visible payload x at the left source endpoint, the absent classifier branch contradicts $\mathsf{hsame}(h, \mathsf{E1}(x))$. In the present branch, compare the display of h with the branch payload display to obtain $\mathsf{hsame}(x, a)$. Source hsame compatibility gives $x \sim_S a$, and the classifier branch gives $a \sim_S b$. Transitivity in cert_S produces $x \sim_S b$, while the branch already displays b at h'. The right-to-left direction is symmetric: compare the display of h' with the branch payload display, use source hsame compatibility, and compose with the branch classifier through cert_S. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel common target visible source completion`

Theorem 33.146 (Tagged option map relation common-source target classification)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume also $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For histories* h, k, k'*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k')$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. Split the two map-relation witnesses. If both witnesses are absent, their target endpoint presentations are $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$, so the absent-absent tagged classifier branch gives the conclusion.

In a mixed branch, one witness presents the shared source endpoint h as e, while the other presents it as $\mathsf{E1}(a)$ for some payload a. Symmetry and transitivity of hsame compose these two presentations into $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(a))$ or its reverse, contradicting generated-history empty-versus-one no-confusion. Hence mixed branches cannot occur.

In the present/present branch, the two witnesses expose payloads a, b with $S(a)$, $S(b)$, target carrier witnesses $T(\delta.\mathsf{map}(a))$ and $T(\delta.\mathsf{map}(b))$, source endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h, \mathsf{E1}(b))$, and target endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b)))$. Composing the two source endpoint presentations and inverting the common one-step constructor yields $\mathsf{hsame}(a, b)$. Source hsame compatibility converts this to $a \sim_S b$, and $\delta.\mathsf{respects}$ sends it to $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$. Together with the stored target carrier and endpoint witnesses, this rebuilds the present-present target classifier for k and k'. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel common source target classification`

Theorem 33.147 (Tagged option map relation common-source visible target completion)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$ *and* $\mathsf{TaggedOptionSourceHs}$ *). For histories* h, k, k'*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k'),$$

then the two target endpoints complete visible payload readback in both directions:

1. *for every visible payload* $x : \mathsf{BHist}$*, if* $T(x)$ *and* $\mathsf{hsame}(k, \mathsf{E1}(x))$*, then there exists* $y : \mathsf{BHist}$ *such that*
$$T(y) \wedge \mathsf{hsame}(k', \mathsf{E1}(y)) \wedge x \sim_T y;$$
2. *for every visible payload* $y : \mathsf{BHist}$*, if* $T(y)$ *and* $\mathsf{hsame}(k', \mathsf{E1}(y))$*, then there exists* $x : \mathsf{BHist}$ *such that*
$$T(x) \wedge \mathsf{hsame}(k, \mathsf{E1}(x)) \wedge x \sim_T y.$$

Proof. First apply Theorem 33.146 to the two common-source map-relation witnesses and the source hsame compatibility hypothesis. This gives $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

For a visible payload x at the left target endpoint, apply Theorem 33.42 to that target classifier, the witness $T(x)$, and $\mathsf{hsame}(k, \mathsf{E1}(x))$. The visible-branch inversion exposes payloads x_0, y with $T(x_0)$, $T(y)$, $\mathsf{hsame}(x, x_0)$, $x_0 \sim_T y$, and $\mathsf{hsame}(k', \mathsf{E1}(y))$. Target hsame compatibility converts $\mathsf{hsame}(x, x_0)$ to $x \sim_T x_0$, and the transitivity field of cert_T composes it with $x_0 \sim_T y$. The payload y is the required matching visible endpoint.

For a visible payload y at the right target endpoint, apply Theorem 33.100 to the same target classifier, the witness $T(y)$, and $\mathsf{hsame}(k', \mathsf{E1}(y))$. The right visible-branch inversion exposes payloads x, y_0 with $T(x)$, $T(y_0)$, $\mathsf{hsame}(y, y_0)$, $x \sim_T y_0$, and $\mathsf{hsame}(k, \mathsf{E1}(x))$. Target hsame compatibility converts $\mathsf{hsame}(y, y_0)$ to $y \sim_T y_0$; the symmetry and transitivity fields of cert_T then compose $x \sim_T y_0$ with $y_0 \sim_T y$. The payload x gives the reverse matching visible endpoint. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel common source visible target completion`

Theorem 33.148 (Tagged option map relation common-source visible target payload classification)**.** *Under the hypotheses of common-source visible target completion, let x, y be target payloads with $T(x)$ and $T(y)$. If the left target endpoint displays x and the right target endpoint displays y,*

$$\mathsf{hsame}(k, \mathsf{E1}(x)) \wedge \mathsf{hsame}(k', \mathsf{E1}(y)),$$

then the two payloads are classified on the target side:

$$x \sim_T y.$$

Proof. Apply common-source visible target completion to the left displayed payload. This produces a target payload z with $T(z)$, right endpoint display $\mathsf{hsame}(k', \mathsf{E1}(z))$, and $x \sim_T z$. Compare the two right endpoint displays, invert the common $\mathsf{E1}$ constructor, and obtain $\mathsf{hsame}(z, y)$. Target hsame compatibility converts this into $z \sim_T y$, and transitivity in cert_T composes $x \sim_T z$ with $z \sim_T y$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel common source visible target payload classification`

Theorem 33.149 (Tagged option payload descent transports semantic fields)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For histories h, h', k, k', if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

then the two source endpoints satisfy their three unary tagged option semantic fields:

$$\mathsf{TaggedOptionHistoryCarrier}_S(h) \wedge \mathsf{TaggedOptionHistoryPattern}_S(h) \wedge \mathsf{TaggedOptionHistoryLedger}_S(h),$$

$$\mathsf{TaggedOptionHistoryCarrier}_S(h') \wedge \mathsf{TaggedOptionHistoryPattern}_S(h') \wedge \mathsf{TaggedOptionHistoryLedger}_S(h'),$$

and the two target endpoints satisfy the corresponding target fields:

$$\mathsf{TaggedOptionHistoryCarrier}_T(k) \wedge \mathsf{TaggedOptionHistoryPattern}_T(k) \wedge \mathsf{TaggedOptionHistoryLedger}_T(k),$$

$$\mathsf{TaggedOptionHistoryCarrier}_T(k') \wedge \mathsf{TaggedOptionHistoryPattern}_T(k') \wedge \mathsf{TaggedOptionHistoryLedger}_T(k').$$

Moreover, the binary tagged option classifier field descends along the same two map-relation witnesses:

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h') \to \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. Split each instance of Definition 33.137. In an absent branch, the witness stores the presentation of the source endpoint as e and the target endpoint as e; these two presentations build the absent carrier witnesses for the corresponding source and target tagged option endpoints. In a present branch, the witness exposes a payload a with $S(a)$, the target payload witness $T(\delta.\mathsf{map}(a))$ supplied by ρ_δ, and the endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$; these fields build the present carrier witnesses for the source and target endpoints. The same argument applies to the primed map-relation witness.

For each of the four endpoint carriers just obtained, apply Theorem 33.31 to convert the carrier witness into the corresponding visible tag pattern and visible tag ledger witnesses. The binary classifier clause is exactly Theorem 33.142, applied to the displayed source classifier witness, the two map-relation witnesses, the source certificate cert, and the source hsame compatibility hypothesis. □

Theorem 33.150 (Tagged option payload descent total endpoint transport)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For every history h, if $\mathsf{TaggedOptionHistoryCarrier}_S(h)$, then there exists $k : \mathsf{BHist}$ such that

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionHistoryCarrier}_T(k) \wedge \mathsf{TaggedOptionHistoryPattern}_T(k) \wedge \mathsf{TaggedOptionHistoryLedger}_T(k).$$

Proof. Split the source carrier witness. In the absent case, choose $k = \mathsf{e}$. The absent branch of Definition 33.137 gives the map-relation witness, and the absent tagged carrier branch gives the target carrier witness.

In the present case, the source witness exposes a payload a with $S(a)$ and $\mathsf{hsame}(h, \mathsf{E1}(a))$. Choose $k = \mathsf{E1}(\delta.\mathsf{map}(a))$; the present branch of Definition 33.137, together with $\rho_\delta(a)$, gives the map-relation witness and the present target carrier witness.

In both branches, apply Theorem 33.31 to the target carrier witness. This converts it into the corresponding visible tag pattern and visible tag ledger witnesses, yielding the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel total carrier transport`

Theorem 33.151 (Tagged option payload descent total classifier transport)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload-carrier preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ and assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. For every pair of histories h, h', if

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h'),$$

then there exist $k, k' : \mathsf{BHist}$ such that

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k) \wedge \mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k') \wedge \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k'$$

Proof. Split the source classifier witness. In the absent branch, choose $k = \mathsf{e}$ and $k' = \mathsf{e}$. The two absent endpoint presentations build the two absent branches of Definition 33.137.

In the present branch, the source witness exposes payloads a, b with $S(a)$, $S(b)$, endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h', \mathsf{E1}(b))$, and a classifier witness $a \sim_S b$. Choose $k = \mathsf{E1}(\delta.\mathsf{map}(a))$ and $k' = \mathsf{E1}(\delta.\mathsf{map}(b))$. The two present branches of Definition 33.137, using $\rho_\delta(a)$ and $\rho_\delta(b)$, give the required map-relation witnesses.

With the map-relation witnesses in either branch, the target classifier is Theorem 33.142 applied to the source classifier witness, the two map-relation witnesses, cert, and the source hsame compatibility hypothesis. □

Theorem 33.152 (Tagged option payload descent public map certificate)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-tagged option payload descent; the BEDC source is* BHist *with* $\mathsf{NameCert}$ *and* $\mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*; the carrier is tagged option histories; the classifier is the tagged option history classifier; the stability fields are totality, target single-valuedness, and classifier preservation; the ledger is visible absent/present tags and transported payloads; the core theorem is this public map certificate; exactness is* $\mathsf{TaggedOptionMapRel}$ *branch analysis; the standard bridge is none/some payload mapping; and the status is* $\mathsf{ProofTarget}$*. Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. Then the constructor-tagged option map relation satisfies the following public map certificate clauses.*

1. *For every history* h*, if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$*, then there exists* $k : \mathsf{BHist}$ *such that*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionHistoryCarrier}_T(k) \wedge \mathsf{TaggedOptionHistoryPattern}_T(k) \wedge \mathsf{TaggedOptionHistoryLedger}_T(k).$$

2. *For histories* h, k, k'*, if the same source endpoint* h *maps to* k *and* k'*, so that*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k'),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T, \sim_T}(k, k').$$

3. *For histories* h, h', k, k'*, if* h *maps to* k*,* h' *maps to* k'*, and the source endpoints are classified together, so that*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

and

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h'),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. For the endpoint totality clause, apply Theorem 33.150 to the source carrier witness. It supplies the target endpoint, the TaggedOptionMapRel witness, and the target carrier, pattern, and ledger fields.

For the same-source target classification clause, apply Theorem 33.146 to the two map-relation witnesses and the source hsame compatibility hypothesis. This gives the target classifier directly.

For the classifier preservation clause, apply Theorem 33.142 to the source classifier witness, the two map-relation witnesses, the source certificate cert, and the source hsame compatibility hypothesis. □

Theorem 33.153 (Tagged option payload descent composition)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-tagged option map composition; the BEDC source is* BHist *with two* DescentCertificate *maps; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is closure of* TaggedOptionMapRel *under composition; the ledger is visible absent/present tags with transported payloads; exactness is branch analysis; and the status is* ProofTarget*. Let* $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be source classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume also endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $p : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(p))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

For histories h, k, m*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m),$$

then

$$\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; h, m).$$

Proof. Split the first map-relation witness. In the absent case, the source endpoint is presented as e and the intermediate endpoint is presented as e. The second map-relation witness cannot be present, since that would also present the same intermediate endpoint as $\mathsf{E1}(b)$ and would contradict empty-versus-one no-confusion at k. Hence the second witness is absent, and the composed witness is the absent branch from h to m.

In the present case, the first witness exposes a payload a with $S(a)$, $T(\delta.\mathsf{map}(a))$, source endpoint $\mathsf{hsame}(h, \mathsf{E1}(a))$, and intermediate endpoint $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. The second map-relation witness cannot be absent, since that would present k as both a one-constructor endpoint and e.

The remaining second witness is present. It exposes a payload b with $T(b)$, $U(\epsilon.\mathsf{map}(b))$, intermediate endpoint $\mathsf{hsame}(k, \mathsf{E1}(b))$, and final endpoint $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(b)))$.

Invert the two presentations of k to obtain $\mathsf{hsame}(\delta.\mathsf{map}(a), b)$, and apply η_ϵ to the two target payload witnesses. This gives $\mathsf{hsame}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)), \epsilon.\mathsf{map}(b))$.

By one-constructor congruence and transitivity of hsame, the final endpoint presentation for m rewrites from $\mathsf{E1}(\epsilon.\mathsf{map}(b))$ to $\mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$.

Together with $\rho_{\epsilon\circ\delta}(a, Sa)$ and the original source endpoint witness, these fields build the present branch of $\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; h, m)$. □

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionDescentComp`

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel composition closed`

Theorem 33.154 (Tagged option map relation composite visible payload readback)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history carriers, let $\sim_S, \sim_T, \sim_U$ be source classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

For histories h, k, m, a, c, assume

$$\mathsf{TaggedOptionMapRel}_\delta(S, T; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U; k, m).$$

If $S(a)$, $\mathsf{hsame}(h, \mathsf{E1}(a))$, $U(c)$, and $\mathsf{hsame}(m, \mathsf{E1}(c))$, then the final visible payload reads back to the composite payload:

$$\mathsf{hsame}(c, \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))).$$

Proof. Split the first map-relation witness. Its absent branch would present h as e, contradicting the visible presentation $\mathsf{hsame}(h, \mathsf{E1}(a))$.

In the present branch, the first witness exposes a source payload a_0 with $\mathsf{hsame}(h, \mathsf{E1}(a_0))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a_0)))$. One-constructor inversion against $\mathsf{hsame}(h, \mathsf{E1}(a))$ identifies a_0 with a.

Split the second map-relation witness. Its absent branch would present k as e, contradicting the visible presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. In the present branch, the second witness exposes a payload b with $\mathsf{hsame}(k, \mathsf{E1}(b))$ and $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(b)))$. One-constructor inversion at k identifies b with $\delta.\mathsf{map}(a)$, and one-constructor inversion at m reads c back as $\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel composition visible payload readback`

Theorem 33.155 (Tagged option payload descent composition exactness)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be source classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume also endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $p : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(p))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

For histories h, m, the composite relation holds exactly when it factors through a canonical intermediate endpoint:

$$\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; h, m) \iff \exists k : \mathsf{BHist}.\ \mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k) \wedge \mathsf{TaggedOptionMapRe}$$

Proof. For the left-to-right direction, split the composite map-relation witness.

In the absent branch, the composite witness presents both endpoints as e. Choose $k = \mathsf{e}$. The first factor is the absent branch of $\mathsf{TaggedOptionMapRel}_\delta$ using the displayed source endpoint presentation and reflexivity at e, and the second factor is the absent branch of $\mathsf{TaggedOptionMapRel}_\epsilon$ using reflexivity at e and the displayed final endpoint presentation.

In the present branch, the composite witness exposes a payload a, a source witness $S(a)$, the composite payload witness $\rho_{\epsilon\circ\delta}(a, Sa)$, the source endpoint presentation $\mathsf{hsame}(h, \mathsf{E1}(a))$, and the final endpoint presentation $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))))$. Choose $k = \mathsf{E1}(\delta.\mathsf{map}(a))$.

The first factor is the present branch for δ, using $S(a)$, $\rho_\delta(a, Sa)$, the source endpoint presentation, and reflexivity at $\mathsf{E1}(\delta.\mathsf{map}(a))$. The second factor is the present branch for ϵ, using $\rho_\delta(a, Sa)$, $\rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa))$, reflexivity at the intermediate endpoint, and the displayed final endpoint presentation.

For the right-to-left direction, take an intermediate history k and the two displayed factor witnesses. Apply Theorem 33.153 with that intermediate witness and the same endpoint-stability hypothesis η_ϵ to obtain the composite $\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}$ witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel composition factorization iff`

Definition 33.156 (Tagged option payload descent image carrier)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For a target endpoint $k : \mathsf{BHist}$, define the tagged option payload descent image carrier by

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \iff \exists h : \mathsf{BHist}.\ \mathsf{TaggedOptionHistoryCarrier}_S(h) \wedge \mathsf{TaggedOpti}$$

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageCarrier`

Theorem 33.157 (Tagged option payload descent image branch exactness)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For every target endpoint $k : \mathsf{BHist}$*, the payload descent image carrier is exactly the absent endpoint together with the transported present payload endpoints:*

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \iff \mathsf{hsame}(k, \mathsf{e}) \vee \exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}$$

This result is not a variant of a checked theorem: it combines the image-carrier definition with the concrete map relation and supplies coverage and completeness for the image predicate.

Proof. For the forward direction, split the stored image-carrier witness. This exposes a source endpoint h, a source tagged-option carrier witness, and a $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ witness.

Now split the map-relation witness. In the absent branch, the relation gives the target presentation $\mathsf{hsame}(k, \mathsf{e})$, so the left side of the displayed disjunction holds. In the present branch, the relation exposes a payload $a : \mathsf{BHist}$, the source witness $S(a)$, the transported target witness $T(\delta.\mathsf{map}(a))$, and the endpoint presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$, giving the right side.

For the reverse direction, split the displayed disjunction. In the absent case, choose the source endpoint e. The absent tagged carrier constructor and the absent map-relation constructor use reflexivity of hsame at e together with the displayed target endpoint witness to build the image-carrier existential.

In the present case, unpack the payload a, the source witness $S(a)$, the target witness $T(\delta.\mathsf{map}(a))$, and the presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. Choose the source endpoint $\mathsf{E1}(a)$; the present tagged carrier constructor uses $S(a)$ and reflexivity of hsame at $\mathsf{E1}(a)$, while the present map-relation constructor uses the displayed target endpoint witness and the payload witnesses. These data repack exactly into the image-carrier existential. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageCarrier branch exactness`

Theorem 33.158 (Tagged option payload descent image certificate)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. Then the image carrier of Definition 33.156 satisfies the following certificate clauses.*

1. *For every source endpoint* h*, if* $\mathsf{TaggedOptionHistoryCarrier}_S(h)$*, then there exists* $k : \mathsf{BHist}$ *such that*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k).$$

2. *For every endpoint k, if*

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k),$$

then

$$\mathsf{TaggedOptionHistoryCarrier}_T(k) \wedge \mathsf{TaggedOptionHistoryPattern}_T(k) \wedge \mathsf{TaggedOptionHistoryLedger}_T(k).$$

3. *For histories h, h', k, k', if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h'),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k),$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageCarrier certificate`

Proof. For the endpoint existence clause, apply Theorem 33.150 to the displayed source carrier witness. It gives an endpoint k together with the required TaggedOptionMapRel witness; the same source carrier witness and map witness are exactly the existential data required by the image carrier definition.

For the target endpoint-scope clause, split the stored image-carrier witness. This exposes a source endpoint h, a source carrier witness, and a $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ witness.

Split the map-relation witness. In the absent branch, the target presentation is the absent endpoint e; in the present branch, it exposes a payload a with $S(a)$, the transported target payload witness $T(\delta.\mathsf{map}(a))$, and the target presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. These two branches give $\mathsf{TaggedOptionHistoryCarrier}_T(k)$.

Apply Theorem 33.31 to that target carrier witness. The equivalence supplies the visible tag pattern and visible tag ledger fields, yielding the displayed conjunction.

For the classifier clause, use Theorem 33.142 with the source classifier witness, the two map-relation witnesses, cert, and the source hsame compatibility hypothesis. Its conclusion is the required target tagged option classifier. □

Theorem 33.159 (Tagged option payload descent image public readback is single-valued)**.** *This theorem fits the per-object template as follows: the traditional target is image option readback for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is deferred to the image classifier chapter; the stability obligation is branch exactness together with target semantic-field exposure; the ledger is the visible transported payload; the core theorem is public readback single-valuedness; exactness is exclusive absent/present coverage; the standard bridge is none/some image readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For every target endpoint* $k : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k),$$

then

$$\mathsf{TaggedOptionHistoryCarrier}_T(k) \wedge \mathsf{TaggedOptionHistoryPattern}_T(k) \wedge \mathsf{TaggedOptionHistoryLedger}_T(k)$$

and the public branch readback is exclusive and single-valued:

$$(\mathsf{hsame}(k, \mathsf{e}) \wedge \neg\exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))))$$

or

$$\exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \neg\mathsf{hsame}(k, \mathsf{e}) \wedge \forall b : \mathsf{BHist}.\ S(b) \to T(\delta.\mathsf{map}(b)) \to \mathsf{hsame}(k,$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageCarrier public readback single valued`

Proof. Apply the target endpoint-scope clause of Theorem 33.158 to the image carrier witness. This supplies the target carrier, visible pattern, and visible ledger fields for k.

Apply Theorem 33.157 to the same image carrier witness. The resulting disjunction says that k is either presented as e or presented as $\mathsf{E1}(\delta.\mathsf{map}(a))$ for a transported source payload a with the displayed source and target payload witnesses.

In the absent branch, keep $\mathsf{hsame}(k, \mathsf{e})$. If a transported present endpoint also existed, composing the empty presentation with the present presentation would give an empty-versus-E1 contradiction by generated-history no-confusion. Hence no displayed transported present endpoint exists.

In the present branch, keep the payload a, its source and target witnesses, and the presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. A simultaneous presentation $\mathsf{hsame}(k, \mathsf{e})$ would again compose with the present presentation to contradict empty-versus-E1 no-confusion.

It remains to show single-valuedness of the public payload readback. Let b be any other displayed transported payload with source witness, target witness, and presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$. Apply Theorem 33.122 to the two present endpoint presentations of k; the result is exactly $\mathsf{hsame}(\delta.\mathsf{map}(a), \delta.\mathsf{map}(b))$. □

Theorem 33.160 (Tagged option payload descent image composition factorization)**.** *Let* $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be source classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume also endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $p : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(p))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

For every final endpoint $m : \mathsf{BHist}$, the composite image carrier factors through an intermediate tagged option endpoint:

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m) \iff \exists k : \mathsf{BHist}.\ \mathsf{TaggedOptionPayloadDescentImageCarrie}$$

Proof. For the forward direction, unpack the composite image-carrier witness. This exposes a source endpoint h, a $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ witness, and a composite map-relation witness from h to m.

Apply Theorem 33.155 in the forward direction to the composite map-relation witness and η_ϵ. It supplies an intermediate endpoint k, a δ-map witness from h to k, and an ϵ-map witness from k to m. The original source tagged carrier witness together with the δ-map witness repacks as $\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k)$, while the ϵ-map witness is the second factor.

For the reverse direction, unpack the displayed factorization. The first factor exposes a source endpoint h, a $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ witness, and a δ-map witness from h to the intermediate endpoint k; the second factor is the displayed ϵ-map witness from k to m.

Apply Theorem 33.155 in the reverse direction to those two map witnesses. It returns the composite $\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; h, m)$ witness. Repacking this composite map witness with the same source tagged carrier witness gives the composite image-carrier witness at m. □

Definition 33.161 (Tagged option payload descent image classifier)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For target endpoints $k, k' : \mathsf{BHist}$, define the tagged option payload descent image classifier by

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \iff \exists h, h' : \mathsf{BHist}.\ \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,$$

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier`

Theorem 33.162 (Tagged option payload descent image classifier branch exactness)**.** *This theorem fits the per-object template as follows: the traditional target is classified image readback for option payload descent; the BEDC source is* BHist *with* hsame *endpoint presentations; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is branch analysis of the source tagged classifier and the two* TaggedOptionMapRel *witnesses; the ledger is the pair of source endpoints and their transported visible payloads; the core theorem is exact absent/present pair readback; exactness is the displayed equivalence; the standard bridge is* none *and* some *under payload descent; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$ and $\sim_T$ be source classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For target endpoints* $k, k' : \mathsf{BHist}$, *the image classifier is exactly the absent target pair or the transported present-payload pair:*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \iff (\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})) \vee (\exists a, b : \mathsf{BHist}.\ S(a) \wedge S($$

Proof. For the forward direction, unfold Definition 33.161. This exposes source endpoints h, h', a source tagged classifier witness, and the two map-relation witnesses from h to k and from h' to k'.

In the absent source-classifier branch, both source endpoints are presented as e. Splitting the two map-relation witnesses rules out any present map-relation branch by empty-versus-$\mathsf{E1}$ no-confusion, so both target endpoints are presented as e. This gives the absent side of the displayed disjunction.

In the present source-classifier branch, the classifier exposes source payloads p, q with $S(p)$, $S(q)$, $p \sim_S q$, and presentations of h, h' as $\mathsf{E1}(p), \mathsf{E1}(q)$. The two map relations cannot be absent, again by empty-versus-$\mathsf{E1}$ no-confusion, and their present branches expose payloads a, b with $S(a)$, $S(b)$, the target carrier witnesses $T(\delta.\mathsf{map}(a))$, $T(\delta.\mathsf{map}(b))$, and the displayed target presentations.

It remains to identify the source payload relation for a and b. Invert the two $\mathsf{E1}$ presentations around h and h' to obtain $\mathsf{hsame}(p, a)$ and $\mathsf{hsame}(q, b)$. Source hsame compatibility gives $p \sim_S a$ and $q \sim_S b$; symmetry and transitivity from cert compose these with $p \sim_S q$ to obtain $a \sim_S b$. The present side of the displayed disjunction follows.

For the reverse direction, split the displayed disjunction. In the absent case, choose source endpoints e and e, use the absent-absent source classifier constructor, and use the two absent map-relation constructors with the displayed target endpoint presentations.

In the present case, unpack a, b, the two source carrier witnesses, the source classifier witness $a \sim_S b$, the two transported target carrier witnesses, and the two target endpoint presentations. Choose source endpoints $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$; the present-present source classifier constructor and the two present map-relation constructors rebuild exactly the existential data required by the image classifier definition. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier branch exactness`

Theorem 33.163 (Tagged option payload descent image classifier right visible-branch inversion)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, *and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For target endpoints* $k, k' : \mathsf{BHist}$, *the right visible branch determines the matching left image branch. The absent clause is*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{hsame}(k', \mathsf{e}) \to \mathsf{hsame}(k, \mathsf{e}).$$

For every payload history $a : \mathsf{BHist}$, *the present clause is*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to T(a) \to \mathsf{hsame}(k', \mathsf{E1}(a)) \to \exists b, c : \mathsf{BHist}.\ S(b) \wedge S(c) /$$

Proof. Apply Theorem 33.162 to the image-classifier witness. This rewrites the witness into the disjunction whose absent side presents both target endpoints as e and whose present side exposes source payloads b, c together with the transported endpoint presentations.

For the absent clause, split that disjunction. The absent side gives $\mathsf{hsame}(k, \mathsf{e})$ directly. The present side also presents k' as $\mathsf{E1}(\delta.\mathsf{map}(c))$ for some source payload c; composing this with the displayed witness $\mathsf{hsame}(k', \mathsf{e})$ gives empty-versus-E1 no-confusion, so the mixed branch is impossible.

For the present clause, split the same disjunction. The absent side presents k' as e, contradicting the displayed witness $\mathsf{hsame}(k', \mathsf{E1}(a))$ by empty-versus-E1 no-confusion.

In the present side, keep the exposed source payloads b, c. The branch-exactness data already supplies $S(b)$, $S(c)$, $b \sim_S c$, $T(\delta.\mathsf{map}(b))$, $T(\delta.\mathsf{map}(c))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$. Compose the right-endpoint presentations $\mathsf{hsame}(k', \mathsf{E1}(a))$ and $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(c)))$, and invert the E1 constructor to obtain $\mathsf{hsame}(a, \delta.\mathsf{map}(c))$. These fields are exactly the displayed existential witness. The hypothesis $T(a)$ records the visible right payload as a target carrier payload and is preserved as part of the public premise. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier right visible branch inversion`

Theorem 33.164 (Tagged option payload descent image classifier visible payload readback)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$ *and visible target payloads* $u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(k', \mathsf{E1}(v)),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(b)).$$

Proof. Apply Theorem 33.162 to the image-classifier witness. This gives the displayed disjunction whose absent side presents both target endpoints as e and whose present side exposes transported source payloads for both endpoints.

The absent side is impossible under the visible presentations. Its left endpoint presents k as e, while the hypotheses present the same endpoint as $\mathsf{E1}(u)$; empty-versus-E1 no-confusion rejects this branch.

In the present side, keep the exposed source payloads a, b. The branch data already supplies $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, $T(\delta.\mathsf{map}(b))$, and the target endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b)))$.

Compose the displayed visible presentations with these two endpoint presentations around k and k'. Inverting the two E1 constructors gives $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$, so the same a, b provide the required existential witness. The hypotheses $T(u)$ and $T(v)$ record that the displayed payloads are target-carrier payloads. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier visible payload readback`

Theorem 33.165 (Tagged option payload descent image classifier certificate)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k') \wedge \mathsf{Tag}$$

Proof. Unpack the image-classifier witness. This exposes source endpoints h, h', the source classifier witness $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$, and the two stored map-relation witnesses from h to k and from h' to k'.

Apply Theorem 33.21 to the source classifier witness. The two resulting source carrier witnesses, together with the corresponding stored map-relation witnesses, are exactly the existential data required by Definition 33.156 for k and k'.

Apply Theorem 33.142 to the same source classifier witness, the two stored map-relation witnesses, cert, and the source hsame compatibility hypothesis. Its conclusion is $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k'$ and these three outputs give the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier certificate`

Theorem 33.166 (Tagged option payload descent image classifier stability fields)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. The image classifier of Definition 33.161 satisfies the following carrier-relative stability fields.*

1. *For every target endpoint* k*, if*

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k).$$

2. *For target endpoints* k, k'*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k', k).$$

3. *For target endpoints* k, k'*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k').$$

Proof. For carrier-relative reflexivity, unpack the displayed image-carrier witness. This gives a source endpoint h, a source carrier witness $\mathsf{TaggedOptionHistoryCarrier}_S(h)$, and a map witness from h to k. The reflexivity clause of Theorem 33.35 turns the source carrier witness into $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,h)$.

Repack Definition 33.161 with the same source endpoint on both sides. The source classifier witness just built and the same map witness used twice give $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,k)$.

For symmetry, unpack the image-classifier witness. This exposes source endpoints h,h', a source classifier witness $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h,h')$, and map witnesses from h to k and from h' to k'. The symmetry clause of Theorem 33.35 gives the reversed source classifier witness $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h',h)$.

Repack the image-classifier existential with the two source endpoints exchanged and the two map witnesses exchanged. This gives the displayed relation from k' to k.

For endpoint exposure, apply Theorem 33.165 to the image-classifier witness with the same cert and source hsame compatibility hypothesis. Projecting the first two components of its conclusion gives exactly the two displayed image-carrier witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier stability fields`

Definition 33.167 (Tagged option payload descent reflects source from visible target equality)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_S$ *and* $\sim_T$ *be classifiers, and let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist},\mathsf{BHist},\sim_S,\sim_T)$*. Define source reflection from visible target equality by*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S,\sim_S) \iff \forall a,b : \mathsf{BHist}.\ S(a) \to S(b) \to \mathsf{hsame}(\delta.\mathsf{map}(a),\delta.\mathsf{map}(b)) -$$

Lean4 (definition) `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentReflectsSource`

Theorem 33.168 (Tagged option payload descent image classifier reflective transitivity)**.** *Let* $S,T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist},\mathsf{BHist},\sim_S,\sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S,\sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S,\sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S,\sim_S).$$

For target endpoints $k,\ell,m : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,\ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;\ell,m),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,m).$$

Proof. Apply Theorem 33.162 to the two image-classifier witnesses. This rewrites each witness into the absent target pair or the present transported-payload pair, with the middle endpoint ℓ appearing in both branch analyses.

1. In the absent/absent case, the first branch gives $\mathsf{hsame}(k,\mathsf{e})$ and $\mathsf{hsame}(\ell,\mathsf{e})$, while the second gives $\mathsf{hsame}(\ell,\mathsf{e})$ and $\mathsf{hsame}(m,\mathsf{e})$. The absent side of the reverse branch-exactness equivalence rebuilds $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,m)$.

2. In the absent/present case, the first branch presents ℓ as e. The second branch presents the same endpoint as $\mathsf{E1}(\delta.\mathsf{map}(c))$ for some source payload c. Empty-versus- $\mathsf{E1}$ no-confusion makes this case impossible.

3. In the present/absent case, the first branch presents ℓ as $\mathsf{E1}(\delta.\mathsf{map}(b))$ for some source payload b, while the second branch presents ℓ as e. Empty-versus-$\mathsf{E1}$ no-confusion again rules out the mixed case.

4. In the present/present case, the first branch exposes source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, and endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(\ell, \mathsf{E1}(\delta.\mathsf{map}(b)))$. The second branch exposes source payloads c, d with $S(c)$, $S(d)$, $c \sim_S d$, and endpoint presentations $\mathsf{hsame}(\ell, \mathsf{E1}(\delta.\mathsf{map}(c)))$ and $\mathsf{hsame}(m, \mathsf{E1}(\delta.\mathsf{map}(d)))$.

   Compose the two presentations of ℓ and invert the $\mathsf{E1}$ constructor to obtain

$$\mathsf{hsame}(\delta.\mathsf{map}(b), \delta.\mathsf{map}(c)).$$

   The reflection hypothesis turns this visible target equality, together with $S(b)$ and $S(c)$, into $b \sim_S c$.

   Transitivity from cert composes $a \sim_S b$, $b \sim_S c$, and $c \sim_S d$ into $a \sim_S d$. The present side of the reverse branch-exactness equivalence, using the already exposed target carrier witnesses and the endpoint presentations for k and m, rebuilds $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, m)$.

□

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier reflective transitivity`

Theorem 33.169 (Tagged option payload descent image semantic certificate)**.** *This theorem fits the ten-field template as follows: the traditional target is the tagged optional image under certified payload descent; the BEDC source is* BHist *with* $\mathsf{DescentCertificate}$*; the carrier is* $\mathsf{TaggedOptionPayloadDescentImage}$ *the classifier is* $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta)$*; the stability obligations are carrier-relative reflexivity, symmetry, reflective transitivity, and carrier transport; the ledger is the target tagged option ledger* $\mathsf{TaggedOptionHistoryLedger}_T$*; the core theorem is the displayed* $\mathsf{SemanticNameCert}$*; exactness is supplied by image carrier and classifier branch exactness; the standard bridge is none/some under payload descent; and the status is* $\mathsf{ProofTarget}$*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

Then the semantic certificate

$$\mathsf{SemanticNameCert}(\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta), \mathsf{TaggedOptionHistoryPattern}_T, \mathsf{TaggedOptio}$$

holds.

Proof. Build the core $\mathsf{NameCert}$ for the image carrier and image classifier. For inhabitedness, use the absent endpoint e. The reverse direction of Theorem 33.157, applied to the absent branch with reflexivity of hsame, gives the required image carrier witness.

Carrier-relative reflexivity and symmetry are the first two fields of Theorem 33.166. Reflective transitivity is exactly Theorem 33.168 under the displayed source-reflection hypothesis.

For carrier transport, start with a classifier witness from k to k' and an image-carrier witness at k. Apply the endpoint exposure field of Theorem 33.166 to the classifier witness, and project the target endpoint carrier at k'. Together with the preceding fields, this assembles

$$\mathsf{NameCert}(\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S,T,\rho_\delta), \mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho$$

It remains to supply the public pattern and ledger fields. Item 2 of Theorem 33.158 sends every image-carrier witness to the target tagged option carrier, visible pattern, and visible ledger. Project its pattern and ledger components to obtain the two soundness maps required by the displayed SemanticNameCert. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier semantic certificate`

Theorem 33.170 (Tagged option payload descent image classifier mixed visible-tag separation)**.** *This theorem fits the per-object template as follows: the traditional target is image option no-confusion under payload descent; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is branch exactness plus generated-history no-confusion; the ledger is the visible absent/present target tag; the core theorem is the two mixed-case contradictions; exactness permits only absent/absent or present/present target pairs; the standard bridge is* none/some *under payload descent; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, let* $k, k' : \mathsf{BHist}$ *be target endpoints, and let* $u, v : \mathsf{BHist}$ *be target payloads. Both mixed visible-tag clauses hold:*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,k') \to \mathsf{hsame}(k,\mathsf{e}) \to T(v) \to \mathsf{hsame}(k',\mathsf{E1}(v)) \to \bot$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S,T,\rho_\delta;k,k') \to T(u) \to \mathsf{hsame}(k,\mathsf{E1}(u)) \to \mathsf{hsame}(k',\mathsf{e}) \to \bot.$$

Proof. Apply Theorem 33.162 to the image-classifier witness. Branch exactness leaves only the absent/absent target pair or a transported present/present target pair.

For the clause with $\mathsf{hsame}(k,\mathsf{e})$ and $\mathsf{hsame}(k',\mathsf{E1}(v))$, the absent/absent branch contradicts the displayed present endpoint on the right by empty-versus-E1 no-confusion. The present/present branch contradicts the displayed absent endpoint on the left by the same no-confusion principle.

For the clause with $\mathsf{hsame}(k,\mathsf{E1}(u))$ and $\mathsf{hsame}(k',\mathsf{e})$, the absent/absent branch contradicts the displayed present endpoint on the left. The present/present branch contradicts the displayed absent endpoint on the right. These two contradictions exclude all cases supplied by branch exactness. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier mixed visible tag separation`

Theorem 33.171 (Tagged option payload descent image classifier nonempty present-pair inversion)**.** *This theorem fits the per-object template as follows: the traditional target is nonempty option image branch inversion; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is branch exactness plus endpoint non-absence; the ledger is the synchronized transported payload pair; the core theorem is present-pair recovery; exactness is elimination of the absent pair; the standard bridge is* none/some *under payload descent; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let cert : $\mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* k, k' : BHist*, endpoint non-absence on either side forces the synchronized transported present pair. The left non-absence clause is*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \neg\mathsf{hsame}(k, \mathsf{e}) \to \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T($$

The right non-absence clause is

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \neg\mathsf{hsame}(k', \mathsf{e}) \to \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T($$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier nonempty present pair inversion`

Proof. Apply Theorem 33.162 to the image-classifier witness. Branch exactness supplies either the absent/absent target pair or an exposed transported present/present target pair.

For the left non-absence clause, the absent/absent branch contains $\mathsf{hsame}(k, \mathsf{e})$ and therefore contradicts the premise $\neg\mathsf{hsame}(k, \mathsf{e})$. The present/present branch already contains payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the two transported target-carrier witnesses, and the displayed present endpoint presentations. Return those witnesses.

For the right non-absence clause, the same branch exactness split applies. The absent/absent branch contains $\mathsf{hsame}(k', \mathsf{e})$ and contradicts the premise $\neg\mathsf{hsame}(k', \mathsf{e})$, while the present/present branch is exactly the required synchronized transported payload pair. □

Theorem 33.172 (Tagged option payload descent image classifier visible target-payload classification)**.** *This theorem fits the per-object template as follows: the traditional target is target-side payload classification for a visible option image pair; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDes *the stability obligation is visible payload readback plus descent respect; the ledger is the displayed target present payload pair; the core theorem is target-payload classifier recovery; exactness supplies transported source payloads behind the visible targets; the standard bridge is* none/some *under payload descent; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ *and* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*. Assume source compatibility* TaggedOptionSourceHsameC *) and target compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*. For target endpoints* k, k' : BHist *and visible target payloads* u, v : BHist*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'), \quad T(u), \quad T(v), \quad \mathsf{hsame}(k, \mathsf{E1}(u)), \quad \mathsf{hsame}($$

then $u \sim_T v$.

Proof. Apply Theorem 33.164 to the image-classifier witness and the two visible present-payload presentations. This produces source payloads a, b with $S(a)$, $S(b)$, and $a \sim_S b$, together with target carrier witnesses for $\delta.\mathsf{map}(a)$ and $\delta.\mathsf{map}(b)$ and the visible payload readbacks $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$.

Use the $\mathsf{respects}$ field of δ on $a \sim_S b$ to obtain $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$. Target compatibility turns $T(u)$, $T(\delta.\mathsf{map}(a))$, and $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ into $u \sim_T \delta.\mathsf{map}(a)$; it also turns $T(v)$, $T(\delta.\mathsf{map}(b))$, and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$ into $v \sim_T \delta.\mathsf{map}(b)$.

Apply symmetry from cert_T to the second target-compatibility relation, giving $\delta.\mathsf{map}(b) \sim_T v$. Transitivity from cert_T first composes $u \sim_T \delta.\mathsf{map}(a)$ with $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$, and then composes the result with $\delta.\mathsf{map}(b) \sim_T v$. The conclusion is $u \sim_T v$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier visible target payload classification`

33.4 Payload descent image classifier transported visible readback

Theorem 33.173 (Tagged option payload descent image classifier transported visible payload readback)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k', u, u' : \mathsf{BHist}$ *and visible target payloads* $x, y : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'), \qquad \mathsf{hsame}(k, u), \qquad \mathsf{hsame}(k', u'),$$

and

$$T(x) \wedge T(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(x, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(y, \delta.\mathsf{map}(b)).$$

Proof. Apply Theorem 33.204 to the classifier witness at (k, k') and to the two endpoint-normalizing witnesses $\mathsf{hsame}(k, u)$ and $\mathsf{hsame}(k', u')$. This gives the same image classifier at the normalized endpoint pair (u, u').

Now apply Theorem 33.164 to that transported classifier and to the displayed visible payload data $T(x)$, $T(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, and $\mathsf{hsame}(u', \mathsf{E1}(y))$. The returned source histories a, b carry $S(a)$, $S(b)$, $a \sim_S b$, the transported target carrier witnesses, and the two payload readback witnesses. These are exactly the fields of the required existential conclusion. □

Theorem 33.174 (Tagged option payload descent image classifier transported visible payload pair coherence)**.** *This theorem fits the per-object template as follows: the traditional target is transported visible readback coherence for constructor-tagged* $\mathsf{OptionUp}$ *under payload descent; the BEDC source is* BHist *with a* $\mathsf{DescentCertificate}$*; the carrier is* $\mathsf{TaggedOptionPayloadDescentImageCarrier}$*; the classifier is* $\mathsf{TaggedOptionPayloadDescentImageClassifier}$*; the stability obligation is endpoint transport*

plus visible endpoint determinacy; the ledger is the two visible target payload pairs after endpoint normalization; the core theorem is shared source-pair coherence; exactness says one transported image-classifier witness explains both visible presentations; the standard bridge is none/some *payload readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k', u, u' : \mathsf{BHist}$ *and visible target payloads* $x, y, x', y' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'), \qquad \mathsf{hsame}(k, u), \qquad \mathsf{hsame}(k', u'),$$

and

$$T(x) \wedge T(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

and

$$T(x') \wedge T(y') \wedge \mathsf{hsame}(u, \mathsf{E1}(x')) \wedge \mathsf{hsame}(u', \mathsf{E1}(y')),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(x, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(y, \delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(x', \delta.\mathsf{m}$$

Proof. First transport the image-classifier witness. Apply Theorem 33.204 to the classifier at (k, k') and to the endpoint witnesses $\mathsf{hsame}(k, u)$ and $\mathsf{hsame}(k', u')$. This yields the transported image classifier at the normalized endpoint pair (u, u').

Next apply Theorem 33.164 to the transported classifier and the first visible pair $T(x)$, $T(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, and $\mathsf{hsame}(u', \mathsf{E1}(y))$. The resulting source histories a, b carry $S(a)$, $S(b)$, $a \sim_S b$, the two target carrier witnesses, and the readback witnesses $\mathsf{hsame}(x, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(y, \delta.\mathsf{map}(b))$.

Compare the two visible presentations of the left normalized endpoint u. Apply Theorem 33.122 to $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$ to obtain $\mathsf{hsame}(x', x)$. Composing this with $\mathsf{hsame}(x, \delta.\mathsf{map}(a))$ gives $\mathsf{hsame}(x', \delta.\mathsf{map}(a))$.

Apply the same visible-payload determinacy at the right normalized endpoint u'. The witnesses $\mathsf{hsame}(u', \mathsf{E1}(y))$ and $\mathsf{hsame}(u', \mathsf{E1}(y'))$ give $\mathsf{hsame}(y', y)$, and composition with $\mathsf{hsame}(y, \delta.\mathsf{map}(b))$ gives $\mathsf{hsame}(y', \delta.\mathsf{map}(b))$.

The original source pair a, b, together with the first readback witnesses and the two transported primed readback witnesses, forms the required shared source-pair witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier transported visible payload pair coherence`

33.5 Payload descent image classifier reflective visible pair coherence

Theorem 33.175 (Tagged option payload descent image classifier reflective transitive visible payload pair coherence)**.** *This theorem fits the per-object template as follows: the traditional target is reflective transitive visible readback coherence for constructor-tagged* OptionUp *under payload descent; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is reflective*

transitivity plus visible endpoint determinacy; the ledger is the two displayed target payload pairs; the core theorem is the shared-source-pair coherence below; exactness says a transitive visible image-classifier path has a single source payload pair for both visible presentations; the standard bridge is none/some *payload readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let cert : $\mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m : \mathsf{BHist}$ *and visible target payloads* $u, v, u', v' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(m, \mathsf{E1}(v))$$

and

$$T(u') \wedge T(v') \wedge \mathsf{hsame}(k, \mathsf{E1}(u')) \wedge \mathsf{hsame}(m, \mathsf{E1}(v')),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u', \delta.\mathsf{m}$$

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier reflective transitive visible pair coherence`

Proof. First combine the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using the source hsame compatibility and source-reflection hypotheses. This gives the direct image-classifier witness from k to m.

Next apply Theorem 33.164 to the direct witness and the first visible target payload pair. The result is a source pair a, b with $S(a)$, $S(b)$, $a \sim_S b$, transported target carrier facts, and the readback identifications $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$.

It remains to compare the primed visible presentation with the unprimed one. At the left endpoint k, the two witnesses present k as both $\mathsf{E1}(u)$ and $\mathsf{E1}(u')$; E1 no-confusion, symmetry of hsame, and transitivity of hsame give $\mathsf{hsame}(u', \delta.\mathsf{map}(a))$. At the right endpoint m, the same argument applied to $\mathsf{E1}(v)$ and $\mathsf{E1}(v')$ gives $\mathsf{hsame}(v', \delta.\mathsf{map}(b))$. Repack the original source pair together with these four payload identifications. □

Theorem 33.176 (Tagged option payload descent image classifier reflective transitive transported visible payload pair coherence)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step readback coherence for constructor-tagged* OptionUp *under payload descent; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImag *the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity,* hsame *endpoint transport, and visible endpoint determinacy; the ledger is the two normalized visible payload presentations; the core theorem is that one source payload pair explains both*

visible presentations; exactness says endpoint normalization does not change the two-step readback; the standard bridge is none/some *payload readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$ *and visible target payloads* $u, v, u', v' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q)$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(p, \mathsf{E1}(u)) \wedge \mathsf{hsame}(q, \mathsf{E1}(v))$$

and

$$T(u') \wedge T(v') \wedge \mathsf{hsame}(p, \mathsf{E1}(u')) \wedge \mathsf{hsame}(q, \mathsf{E1}(v')),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u', \delta.\mathsf{m}$$

Proof. First combine the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), together with source compatibility and source reflection, to obtain the direct image classifier from k to m.

Next normalize the endpoints. Apply Theorem 33.204 to the direct classifier and the endpoint witnesses $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. This yields the same image-classifier witness at the normalized endpoint pair (p, q).

Apply Theorem 33.164 to the normalized classifier and the first visible payload presentation at (p, q). It returns source histories a, b with $S(a)$, $S(b)$, $a \sim_S b$, target carrier facts for $\delta.\mathsf{map}(a)$ and $\delta.\mathsf{map}(b)$, and the readbacks $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$.

It remains to compare the second normalized visible presentation with the first. At endpoint p, apply Theorem 33.122 to $\mathsf{hsame}(p, \mathsf{E1}(u))$ and $\mathsf{hsame}(p, \mathsf{E1}(u'))$. By symmetry and transitivity of hsame, the resulting one-constructor inversion gives $\mathsf{hsame}(u', \delta.\mathsf{map}(a))$.

At endpoint q, the same visible endpoint determinacy applied to $\mathsf{hsame}(q, \mathsf{E1}(v))$ and $\mathsf{hsame}(q, \mathsf{E1}(v'))$ gives $\mathsf{hsame}(v', v)$. Composing this with $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$ yields $\mathsf{hsame}(v', \delta.\mathsf{map}(b))$.

Repack the original source pair a, b with the two source-carrier witnesses, the source classifier relation, the transported target-carrier witnesses, and the four visible payload readbacks. These are precisely the fields of the displayed existential conclusion. □

Theorem 33.177 (Tagged option payload descent image classifier reflective transitive transported visible target-payload classification)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step option image payload classification; the BEDC source*

is BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity, endpoint transport, visible payload readback, descent respect, and target certificate transitivity; the ledger is the normalized visible target payload pair; the core theorem is visible target payload classification by* $\sim_T$*; exactness says one transported two-step image witness explains the visible target pair; the standard bridge is* none/some *payload classification under descent; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ *and* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*. Assume source compatibility* TaggedOptionSourceHsameC) *and target compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*. Assume also*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$ *and visible target payloads* $u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q)$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(p, \mathsf{E1}(u)) \wedge \mathsf{hsame}(q, \mathsf{E1}(v)),$$

then $u \sim_T v$*.*

Proof. First compose the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using source compatibility and source reflection. This gives the direct image-classifier witness from k to m.

Next normalize the endpoints. Apply Theorem 33.204 to the direct classifier together with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. The result is an image-classifier witness at the normalized visible target endpoint pair (p, q).

Apply Theorem 33.164 to the normalized classifier and the visible payload presentation $\mathsf{hsame}(p, \mathsf{E1}(u))$, $\mathsf{hsame}(q, \mathsf{E1}(v))$. It returns source payloads a, b with $S(a)$, $S(b)$, and $a \sim_S b$, transported target carrier witnesses for $\delta.\mathsf{map}(a)$ and $\delta.\mathsf{map}(b)$, and readbacks $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$.

Use the respects field of δ to obtain $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$. Target compatibility gives $u \sim_T \delta.\mathsf{map}(a)$ and $v \sim_T \delta.\mathsf{map}(b)$. Symmetry from cert_T turns the second relation around, and transitivity from cert_T composes the three target-classifier relations into $u \sim_T v$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier reflective transitive transport visible target payload classification`

Theorem 33.178 (Tagged option payload descent image classifier reflective transitive visible target-payload classification)**.** *This theorem fits the per-object template as follows: the traditional target is target two-step image option target-payload classification; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier*

is TaggedOptionPayloadDescentImageClassifier*; the stability obligation is reflective transitivity plus target classifier descent; the ledger is the two visible outer target payloads; the core theorem is the displayed* $u \sim_T v$*; exactness is direct classifier recovery after composing the path; the standard bridge is* none/some *payload classification; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ *and* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*. Assume source compatibility* TaggedOptionSourceHsameC
) *and target compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*. Assume also*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For $k, \ell, m, u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$T(u), \qquad T(v), \qquad \mathsf{hsame}(k, \mathsf{E1}(u)), \qquad \mathsf{hsame}(m, \mathsf{E1}(v)),$$

then $u \sim_T v$*.*

Proof. First compose the visible image path. Apply Theorem 33.168 to the two image-classifier witnesses, using source hsame compatibility and source reflection. This gives a direct image-classifier witness from k to m.

Now apply Theorem 33.172 to the direct witness, the two target carrier witnesses $T(u)$ and $T(v)$, and the visible endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(u))$ and $\mathsf{hsame}(m, \mathsf{E1}(v))$. The conclusion is exactly $u \sim_T v$. □

Theorem 33.179 (Tagged option payload descent image classifier reflective transitive pair readback is single-valued)**.** *This theorem fits the per-object template as follows: the traditional target is reflective two-step image option readback for constructor-tagged* OptionUp *under payload descent; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is reflective transitivity plus paired branch exactness; the ledger is the outer absent tag or transported outer present pair; the core theorem is the complete absent/present split below; exactness says the outer endpoints inherit direct pair readback; the standard bridge is* none/some *payload readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

hold, then either

$\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{e}) \wedge \neg\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{h}$

or

$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(m, \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge \neg\mathsf{hs}$

Proof. First combine the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using the source hsame compatibility and source-reflection hypotheses. This gives the direct image-classifier witness from k to m.

Now apply Theorem 33.201 to the direct witness at the outer endpoint pair (k, m). Its absent branch is already the displayed $\mathsf{hsame}(k, \mathsf{e})$, $\mathsf{hsame}(m, \mathsf{e})$, and exclusion of every transported present pair at (k, m).

In the present branch, keep the source pair a, b, its source carrier witnesses, the source classifier witness, the transported target carrier facts, the two outer visible presentations, and the two non-absence clauses. The paired single-valuedness clause is exactly the one returned by the direct pair readback theorem for every competing transported pair at (k, m), so the branch repackages unchanged. □

Theorem 33.180 (Tagged option payload descent image classifier reflective transitive transported pair readback is single-valued)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step option image readback; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity and* hsame *endpoint transport; the ledger is the normalized absent pair or transported present pair; the core theorem is the absent/present exclusive split at normalized endpoints; exactness is normalized readback; the standard bridge is* none/some*; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$*, if the image classifiers hold from* k *to* ℓ *and from* ℓ *to* m*, and if* $\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q)$*, then either*

$\mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}) \wedge \neg\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hs}$

or

$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(q, \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge \neg\mathsf{hsa}$

Proof. First compose the two image classifiers. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using source compatibility and source reflection. This gives the direct classifier from k to m.

Next transport the composed classifier along the endpoint identifications. Apply Theorem 33.204 with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. The result is a normalized direct classifier at (p, q).

Finally apply Theorem 33.201 to the normalized direct classifier. Its absent branch gives the normalized empty endpoints and excludes every transported present pair at (p, q); its present branch gives the displayed source pair, transported target carrier witnesses, non-absence of p and q, and uniqueness against every competing transported present pair at (p, q). □

Theorem 33.181 (Tagged option payload descent image classifier reflective transitive transported visible-branch inversion)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step option image branch readback; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity, endpoint transport, and left/right visible-branch inversion; the ledger is normalized absent tags or transported visible payloads; the core theorem is the four displayed one-sided clauses; exactness is direct branch inversion after normalization; the standard bridge is* none/some *readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell), \qquad \mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho$$

then the following four clauses hold at (p, q)*.*

1. *The normalized left absent endpoint forces the normalized right absent endpoint:*

$$\mathsf{hsame}(p, \mathsf{e}) \to \mathsf{hsame}(q, \mathsf{e}).$$

2. *Every normalized left visible payload reads back to a transported right visible endpoint:*

$$\forall u : \mathsf{BHist}.\ T(u) \to \mathsf{hsame}(p, \mathsf{E1}(u)) \to \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.$$

3. *The normalized right absent endpoint forces the normalized left absent endpoint:*

$$\mathsf{hsame}(q, \mathsf{e}) \to \mathsf{hsame}(p, \mathsf{e}).$$

4. *Every normalized right visible payload reads back to a transported left visible endpoint:*

$$\forall u : \mathsf{BHist}.\ T(u) \to \mathsf{hsame}(q, \mathsf{E1}(u)) \to \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.$$

Proof. First compose the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using the source hsame compatibility and source-reflection hypotheses. This gives the direct classifier from k to m.

Next normalize the endpoints. Apply Theorem 33.204 with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$ to obtain the transported classifier at (p, q).

Finally project the one-sided branches from the normalized classifier. The left visible-branch inversion theorem gives the first two displayed clauses, because its absent clause sends $\mathsf{hsame}(p, \mathsf{e})$ to $\mathsf{hsame}(q, \mathsf{e})$ and its present clause reads a visible payload at p into source witnesses a, b and a transported visible endpoint at q.

The right visible-branch inversion theorem gives the remaining two displayed clauses in the same normalized classifier. Its absent clause sends $\mathsf{hsame}(q, \mathsf{e})$ to $\mathsf{hsame}(p, \mathsf{e})$, and its present clause reads a visible payload at q into source witnesses a, b and a transported visible endpoint at p. □

33.6 Payload descent image classifier reflective transitive normalized public endpoint readback

Theorem 33.182 (Tagged option payload descent image classifier reflective transitive normalized image closure)**.** *Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source and target carriers, let $\sim_S$ and $\sim_T$ be source and target classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; p, q).$$

Proof. First compose the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using source compatibility and source reflection. This gives the direct classifier from k to m.

Next normalize the public endpoints. Apply Theorem 33.204 with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. The result is the displayed classifier at the normalized endpoint pair (p, q). □

Theorem 33.183 (Tagged option payload descent image classifier reflective transitive normalized public endpoint readback)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step public readback for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*;*

the classifier is TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity, endpoint transport, carrier exposure, target classifier exposure, and paired branch readback; the ledger is the normalized absent pair or transported present pair; the core theorem is the displayed public endpoint conjunction and exclusive branch split; exactness is normalized public endpoint readback; the standard bridge is* none/some *readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let cert : $\mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; p) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; q) \wedge \mathsf{Tagg}$$

and either

$$\mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}) \wedge \neg\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hs}$$

or

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(q, \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge \neg\mathsf{hsa}$$

Proof. First obtain normalized image closure. Apply Theorem 33.182 to the two image-classifier witnesses and the endpoint equalities. This gives the direct classifier at (p, q).

Apply Theorem 33.165 to the normalized classifier. This exposes the two normalized image carriers at p and q and the target tagged classifier $\mathsf{TaggedOptionHistoryClassifier}_{T, \sim_T}(p, q)$.

Finally apply Theorem 33.201 to the same normalized classifier. Its absent branch gives the normalized empty pair and rules out every transported present pair at (p, q); its present branch gives the displayed source pair, transported target witnesses, endpoint presentations, non-absence clauses, and paired single-valuedness. Conjoin these branch data with the exposed public carrier and classifier fields. □

Theorem 33.184 (Tagged option payload descent image classifier reflective transitive normalized branch occupancy equivalence)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then

$$(\mathsf{hsame}(p, \mathsf{e}) \iff \mathsf{hsame}(q, \mathsf{e})) \wedge ((\exists u : \mathsf{BHist}.\ T(u) \wedge \mathsf{hsame}(p, \mathsf{E1}(u))) \iff (\exists v : \mathsf{BHist}.\ T(v) \wedge \mathsf{hsame}(q, \mathsf{E1}(v)))) \wedge (\mathsf{hsa}$$

Proof. Apply Theorem 33.181 to the two image-classifier witnesses and the endpoint equalities. This gives the four normalized one-sided clauses at (p, q).

The first and third clauses give the empty-occupancy equivalence directly. The left absent clause sends $\mathsf{hsame}(p, \mathsf{e})$ to $\mathsf{hsame}(q, \mathsf{e})$, and the right absent clause sends $\mathsf{hsame}(q, \mathsf{e})$ to $\mathsf{hsame}(p, \mathsf{e})$.

For the forward visible-occupancy direction, unpack a witness $u : \mathsf{BHist}$ with $T(u)$ and $\mathsf{hsame}(p, \mathsf{E1}(u))$. The left visible clause returns source witnesses a, b and a transported right endpoint $\mathsf{hsame}(q, \mathsf{E1}(\delta.\mathsf{map}(b)))$ together with $T(\delta.\mathsf{map}(b))$. Choosing $v = \delta.\mathsf{map}(b)$ gives the right visible witness.

For the reverse visible-occupancy direction, unpack a witness $v : \mathsf{BHist}$ with $T(v)$ and $\mathsf{hsame}(q, \mathsf{E1}(v))$. The right visible clause returns source witnesses a, b and a transported left endpoint $\mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a)))$ together with $T(\delta.\mathsf{map}(a))$. Choosing $u = \delta.\mathsf{map}(a)$ gives the left visible witness.

It remains to exclude simultaneous empty and visible occupancy at each normalized endpoint. If $\mathsf{hsame}(p, \mathsf{e})$ and $\mathsf{hsame}(p, \mathsf{E1}(u))$ both hold, compose them by symmetry and transitivity to obtain $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(u))$, contradicting BEDC history no-confusion via *not_hsame_emp_e*1.

If $\mathsf{hsame}(q, \mathsf{e})$ and $\mathsf{hsame}(q, \mathsf{E1}(v))$ both hold, compose the visible presentation with the empty presentation to obtain $\mathsf{hsame}(\mathsf{E1}(v), \mathsf{e})$, contradicting BEDC history no-confusion via *not_hsame_e*1*_empty*. □

Theorem 33.185 (Tagged option payload descent image classifier reflective transitive normalized nonempty present-pair inversion)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized two-step present-pair recovery for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the obligations are reflective transitivity, endpoint transport, and direct nonempty present-pair inversion; the ledger is the synchronized transported payload pair; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then

$$(\neg\mathsf{hsame}(p, \mathsf{e}) \to \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(p, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(q, \mathsf{E1}$$

Proof. First apply Theorem 33.182 to the two image-classifier witnesses and the endpoint equalities. This gives the direct normalized classifier at (p, q).

For the left non-absence clause, apply Theorem 33.171 to the normalized classifier and the premise $\neg\mathsf{hsame}(p, \mathsf{e})$. The direct inversion returns source witnesses a, b with $S(a)$, $S(b)$, $a \sim_S b$, the transported target-carrier witnesses, and the displayed present presentations at p and q.

For the right non-absence clause, apply the same direct inversion theorem to the normalized classifier and the premise $\neg\mathsf{hsame}(q, \mathsf{e})$. Its right endpoint clause returns the same synchronized transported present-pair data, which is the second displayed conclusion. □

Theorem 33.186 (Tagged option payload descent image classifier reflective transitive normalized visible payload exactness)**.** *This theorem fits the per-object template as follows: the traditional target is normalized two-step visible payload exactness for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the obligations are reflective transitivity, endpoint transport, and direct visible-payload exactness; the ledger is the normalized visible payload endpoints; the core theorem is the displayed equivalence; exactness is source-payload explanation of normalized visible target payloads; the standard bridge is* none/some *payload readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S$ *), and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$ *and visible target payloads* $u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then

$$(T(u) \wedge T(v) \wedge \mathsf{hsame}(p, \mathsf{E1}(u)) \wedge \mathsf{hsame}(q, \mathsf{E1}(v))) \iff$$

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(p, \mathsf{E1}(u)) \wedge \mathsf{hsame}(q, \mathsf{E1}(v)) \wedge \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}($$

Proof. First compose the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using source compatibility and source reflection. This gives the direct image-classifier witness from k to m.

Next normalize the endpoints. Apply Theorem 33.204 to the direct classifier together with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. The result is an image-classifier witness at (p, q).

For the forward direction, apply Theorem 33.211 to the normalized classifier and the visible payload facts at (p, q). Its left-to-right direction keeps the four visible facts and supplies source payloads a, b explaining u and v through $\delta.\mathsf{map}$.

For the reverse direction, unpack the right-hand side and project the first four visible facts. These are exactly $T(u)$, $T(v)$, $\mathsf{hsame}(p, \mathsf{E1}(u))$, and $\mathsf{hsame}(q, \mathsf{E1}(v))$, which form the left-hand side of the displayed equivalence. □

Theorem 33.187 (Tagged option payload descent image classifier reflective transitive normalized mixed visible-tag separation)**.** *This theorem fits the per-object template as follows: the traditional target is normalized no-confusion for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity, endpoint transport, and mixed visible-tag separation; the ledger records the normalized absent/present target tags; the core theorem is the two contradiction clauses; exactness permits only absent/absent or present/present tags after normalization; the standard bridge is* none/some *tag separation; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be source and target classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume source compatibility* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$ *and target payloads* $u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q),$$

then both mixed visible-tag clauses hold:

$$\mathsf{hsame}(p, \mathsf{e}) \to T(v) \to \mathsf{hsame}(q, \mathsf{E1}(v)) \to \bot$$

and

$$T(u) \to \mathsf{hsame}(p, \mathsf{E1}(u)) \to \mathsf{hsame}(q, \mathsf{e}) \to \bot.$$

Proof. First compose the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using source compatibility and source reflection. This gives the direct image-classifier witness from k to m.

Next normalize the endpoints. Apply Theorem 33.204 to the direct classifier together with $\mathsf{hsame}(k, p)$ and $\mathsf{hsame}(m, q)$. The result is an image-classifier witness at (p, q).

Finally apply Theorem 33.170 to the normalized classifier. Its first contradiction clause gives $\mathsf{hsame}(p, \mathsf{e}) \to T(v) \to \mathsf{hsame}(q, \mathsf{E1}(v)) \to \bot$, and its second contradiction clause gives $T(u) \to \mathsf{hsame}(p, \mathsf{E1}(u)) \to \mathsf{hsame}(q, \mathsf{e}) \to \bot$. □

Theorem 33.188 (Tagged option payload descent image classifier reflective transitive normalized visible payload single-valuedness)**.** *This theorem fits the per-object template as follows: the traditional target is deterministic endpoint-normalized* OptionUp *visible payload readback; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are reflective transitivity, endpoint transport, target-payload classification, present-payload determinism, and mixed visible-tag separation; the ledger is a pair of competing normalized visible target presentations; the core theorem is classification by $\sim_T$, endpointwise payload uniqueness, and non-absence of the visible endpoints; exactness says the normalized two-step image readback has a single visible payload pair; the standard bridge is* none/some *public readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source and target carriers, let $\sim_S$ and $\sim_T$ be source and target classifiers, let $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ and $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$. Assume source compatibility TaggedOptionSourceHsameC
) and target compatibility $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*. Assume also*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m, p, q : \mathsf{BHist}$ and visible target payloads $u, v, u', v' : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$\mathsf{hsame}(k, p) \wedge \mathsf{hsame}(m, q)$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(p, \mathsf{E1}(u)) \wedge \mathsf{hsame}(q, \mathsf{E1}(v))$$

and

$$T(u') \wedge T(v') \wedge \mathsf{hsame}(p, \mathsf{E1}(u')) \wedge \mathsf{hsame}(q, \mathsf{E1}(v')),$$

then

$$u \sim_T v \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v') \wedge \neg\mathsf{hsame}(p, \mathsf{e}) \wedge \neg\mathsf{hsame}(q, \mathsf{e}).$$

Proof. First classify the displayed target payloads. Apply Theorem 33.177 to the two image-classifier witnesses, the endpoint normalization witnesses, and the first visible presentation at (p, q). This gives $u \sim_T v$.

Next compare the two public presentations at the left endpoint. Apply Theorem 33.122 to $\mathsf{hsame}(p, \mathsf{E1}(u))$ and $\mathsf{hsame}(p, \mathsf{E1}(u'))$. The result is $\mathsf{hsame}(u, u')$.

The right endpoint is identical. Apply Theorem 33.122 to $\mathsf{hsame}(q, \mathsf{E1}(v))$ and $\mathsf{hsame}(q, \mathsf{E1}(v'))$. The result is $\mathsf{hsame}(v, v')$.

It remains to rule out hidden absent tags at the normalized endpoints. Apply Theorem 33.187 to the two image-classifier witnesses and the endpoint normalization witnesses. Its first contradiction clause, together with $T(v)$ and $\mathsf{hsame}(q, \mathsf{E1}(v))$, proves $\neg\mathsf{hsame}(p, \mathsf{e})$; its second contradiction clause, together with $T(u)$ and $\mathsf{hsame}(p, \mathsf{E1}(u))$, proves $\neg\mathsf{hsame}(q, \mathsf{e})$.

Collect these five conclusions in the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier reflective transitive normalized visible payload single valued`

33.7 Composite image classifier visible payload classification

Theorem 33.189 (Tagged option composite image classifier visible payload classification)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *visible payload classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is visible factorization followed by target-classifier transport; the ledger records final visible payloads and intermediate payload readbacks; the core theorem is the displayed factorization together with $p \sim_T q$ and $x \sim_U y$; exactness is composite classified final payload recovery through the intermediate payloads; the standard bridge is* none/some *functorial payload classification; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, and $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. Assume TaggedOptionSourceHsameCompatible$(S, \sim_S)$*,* TaggedOptionSourceHsameCompatible$(T, \sim_T)$*, and* TaggedOptionSou *). For final endpoints $m, m' : \mathsf{BHist}$ and visible final payloads $x, y : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y)),$$

then there exist $k, k', p, q, a, b : \mathsf{BHist}$ such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m')$;
3. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
4. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
5. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
6. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
7. $p \sim_T q$;
8. $x \sim_U y$.

Proof. Apply Theorem 33.213 to the composite image-classifier witness and the visible final presentations. This supplies k, k', p, q, a, b and the six displayed factorization clauses: the intermediate image-classifier witness, the two ϵ map-relation witnesses, the carrier data, the four visible readbacks, the source classified payload pair, and the two δ payload readbacks.

Use $\delta.\mathsf{respects}$ on $a \sim_S b$ to obtain $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$. Target compatibility for T turns $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ into $p \sim_T \delta.\mathsf{map}(a)$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$ into $q \sim_T \delta.\mathsf{map}(b)$. Symmetry from cert_T reverses the second relation, and transitivity from cert_T composes the three relations into $p \sim_T q$.

Now apply $\epsilon.\mathsf{respects}$ to $p \sim_T q$, giving $\epsilon.\mathsf{map}(p) \sim_U \epsilon.\mathsf{map}(q)$. Compatibility for U turns $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$ into $x \sim_U \epsilon.\mathsf{map}(p)$ and $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$ into $y \sim_U \epsilon.\mathsf{map}(q)$. Symmetry and transitivity from cert_U compose these three relations into $x \sim_U y$.

Package the original six factorization clauses with the two classifier consequences $p \sim_T q$ and $x \sim_U y$. □

Theorem 33.190 (Tagged option composite image classifier public branch classification)**.** *This theorem fits the per-object template as follows: the traditional target is public constructor-branch coverage for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is public branch readback followed by visible factorization and classifier transport; the ledger records absent final endpoints or a factored visible final pair; the core theorem is the displayed exhaustive disjunction; exactness says the composite classifier either has no visible final pair or factors through one classified intermediate pair; the standard bridge is* none/some *functorial payload classification; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\, T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints $m, m' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

holds, then either

$$\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(m', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\, U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y))$$

or there exist $k, k', p, q, x, y, a, b : \mathsf{BHist}$ such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m')$;
3. $U(x)$, $U(y)$, $\mathsf{hsame}(m, \mathsf{E1}(x))$, *and* $\mathsf{hsame}(m', \mathsf{E1}(y))$;
4. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
5. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
6. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
7. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
8. $p \sim_T q$;
9. $x \sim_U y$.

Proof. Apply Theorem 33.201 to the composite image-classifier witness for $\epsilon\circ\delta$. Its public readback splits the final pair into an absent branch and a present branch for the composite transported payloads.

In the absent branch, keep the endpoint witnesses $\mathsf{hsame}(m, \mathsf{e})$ and $\mathsf{hsame}(m', \mathsf{e})$. Suppose a visible final pair x, y also existed with $U(x)$, $U(y)$, $\mathsf{hsame}(m, \mathsf{E1}(x))$, and $\mathsf{hsame}(m', \mathsf{E1}(y))$. Composing the displayed empty presentation of m with the visible presentation of m gives empty-versus-$\mathsf{E1}$ no-confusion, so no such visible final pair can exist.

In the present branch, the pair readback supplies source payloads a, b for the composite descent and final visible presentations of the form $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))))$ and $\mathsf{hsame}(m', \mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(b))))$. Take $x = \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and $y = \epsilon.\mathsf{map}(\delta.\mathsf{map}(b))$ as the visible final payloads.

Now apply Theorem 33.189 to the same composite classifier and to these visible final payloads. The theorem returns intermediate endpoints k, k', intermediate visible payloads p, q, and source payloads explaining p, q through δ, together with the intermediate classifier consequence $p \sim_T q$ and the final classifier consequence $x \sim_U y$.

Package the retained visible final endpoint facts with the returned factorization witnesses: the δ image-classifier factor, the two ϵ map relations, the intermediate carrier data, the source pair, the two δ readbacks, and the two classifier consequences. This is exactly the present disjunct. □

Theorem 33.191 (Tagged option composite image classifier canonical public branch readback)**.** *This theorem fits the per-object template as follows: the traditional target is canonical public branch readback for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is public branch classification followed by visible final-payload determinacy; the ledger records either absent final endpoints or a single visible final payload pair up to* hsame*; the core theorem is the displayed branch split with uniqueness of competing visible final pairs; exactness says the present public branch has a canonical final target reading; the standard bridge is* none/some *functorial payload readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou*). For final endpoints* $m, m' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

holds, then either

$$\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(m', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y))$$

or there exist $k, k', p, q, x, y, a, b : \mathsf{BHist}$ *such that*

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m')$*;*
3. $U(x)$*,* $U(y)$*,* $\mathsf{hsame}(m, \mathsf{E1}(x))$*, and* $\mathsf{hsame}(m', \mathsf{E1}(y))$*;*
4. $T(p)$*,* $T(q)$*,* $U(\epsilon.\mathsf{map}(p))$*, and* $U(\epsilon.\mathsf{map}(q))$*;*
5. $\mathsf{hsame}(k, \mathsf{E1}(p))$*,* $\mathsf{hsame}(k', \mathsf{E1}(q))$*,* $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$*, and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$*;*

6. *$S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, and $T(\delta.\mathsf{map}(b))$;*
7. *$\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;*
8. *$p \sim_T q$;*
9. *$x \sim_U y$;*
10. *$\neg\mathsf{hsame}(m, \mathsf{e})$ and $\neg\mathsf{hsame}(m', \mathsf{e})$;*
11. *for every $x', y' : \mathsf{BHist}$, if*

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(m, \mathsf{E1}(x')) \wedge \mathsf{hsame}(m', \mathsf{E1}(y')),$$

then

$$\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. Apply Theorem 33.190 to the composite image-classifier witness and the displayed certificate and compatibility data. This gives the absent branch or a present branch with witnesses k, k', p, q, x, y, a, b and the listed factorization clauses.

In the absent branch, keep the endpoint presentations $\mathsf{hsame}(m, \mathsf{e})$ and $\mathsf{hsame}(m', \mathsf{e})$ together with the displayed exclusion of visible final pairs. This is exactly the first disjunct.

In the present branch, keep the witnesses and all classification and factorization clauses returned by public branch classification. The visible presentations $\mathsf{hsame}(m, \mathsf{E1}(x))$ and $\mathsf{hsame}(m', \mathsf{E1}(y))$ exclude opposite empty presentations by the empty-versus-$\mathsf{E1}$ no-confusion principle, giving $\neg\mathsf{hsame}(m, \mathsf{e})$ and $\neg\mathsf{hsame}(m', \mathsf{e})$.

It remains to prove canonical readback for a competing visible final pair. Let x', y' satisfy $U(x')$, $U(y')$, $\mathsf{hsame}(m, \mathsf{E1}(x'))$, and $\mathsf{hsame}(m', \mathsf{E1}(y'))$.

Apply Theorem 33.192 to the chosen visible pair x, y and the competing visible pair x', y'. The determinacy conclusion gives $\mathsf{hsame}(x, x')$, $\mathsf{hsame}(y, y')$, and $x' \sim_U y'$.

Package the original present-branch witnesses, the two non-absence conclusions, and this universal competing-pair conclusion into the second disjunct. □

Theorem 33.192 (Tagged option composite image classifier visible payload determinacy)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *final visible-payload determinacy; the BEDC source is* BHist *with two* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent respect, target visible-payload classification,* E1 *payload determinacy, and mixed visible-tag separation; the ledger is the two visible final endpoints; the core theorem is the displayed determinacy, classification, and non-absence conjunction; exactness says visible final payload readings are endpoint-determined and target-classified; the standard bridge is* none/some *final readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints $m, m' : \mathsf{BHist}$ and visible final payloads $x, y, x', y' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y))$$

and

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(m, \mathsf{E1}(x')) \wedge \mathsf{hsame}(m', \mathsf{E1}(y')),$$

then

$$x \sim_U y \wedge x' \sim_U y' \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge \neg\mathsf{hsame}(m, \mathsf{e}) \wedge \neg\mathsf{hsame}(m', \mathsf{e}).$$

Proof. First classify the two visible final pairs. Apply Theorem 33.189 to the composite image-classifier witness and the presentation $U(x)$, $U(y)$, $\mathsf{hsame}(m, \mathsf{E1}(x))$, and $\mathsf{hsame}(m', \mathsf{E1}(y))$. This gives $x \sim_U y$. Apply the same theorem to the second presentation to obtain $x' \sim_U y'$.

Next compare the two readings of each endpoint. Apply Theorem 33.122 to $\mathsf{hsame}(m, \mathsf{E1}(x))$ and $\mathsf{hsame}(m, \mathsf{E1}(x'))$, giving $\mathsf{hsame}(x, x')$. The same theorem applied at m' gives $\mathsf{hsame}(y, y')$.

It remains to rule out hidden absent tags at the final endpoints. Apply Theorem 33.170 to the composite descent certificate $\epsilon \circ \delta$ with target carrier U.

Its first contradiction clause, together with $U(y)$ and $\mathsf{hsame}(m', \mathsf{E1}(y))$, proves $\neg\mathsf{hsame}(m, \mathsf{e})$. Its second contradiction clause, together with $U(x)$ and $\mathsf{hsame}(m, \mathsf{E1}(x))$, proves $\neg\mathsf{hsame}(m', \mathsf{e})$.

Collect the two target-classification conclusions, the two payload-determinacy conclusions, and the two non-absence conclusions in the displayed conjunction. □

33.8 Exclusive classifier branch readback

Theorem 33.193 (Tagged option carrier single-valued visible readback)**.** *This theorem fits the per-object template as follows: the traditional target is tagged optional carrier readback; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is unused; the stability obligation is constructor inversion plus branch exclusion; the ledger is the visible absent/present tag and present payload; the core theorem is carrier single-valued visible readback; exactness is existence, opposite-branch exclusion, and payload uniqueness; the standard bridge is* none/some*; and the status is* ProofTarget*. For every source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and history $h : \mathsf{BHist}$, if $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ holds, then either*

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))$$

or

$$\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \neg\mathsf{hsame}(h, \mathsf{e}) \wedge (\forall a' : \mathsf{BHist}.\ S(a') \to \mathsf{hsame}(h, \mathsf{E1}(a')) \to \mathsf{hsame}(a, a')).$$

Proof. Apply Theorem 33.128 to the carrier witness. This gives the displayed absent branch or a visible present payload branch.

In the absent branch, keep the equality $\mathsf{hsame}(h, \mathsf{e})$ and the negative source-present clause supplied by the carrier partition. These are exactly the two fields required for the first displayed alternative.

In the present branch, keep the payload a, the source witness $S(a)$, the endpoint link $\mathsf{hsame}(h, \mathsf{E1}(a))$, and the non-absence proof supplied by the carrier partition. For any other visible present payload a', apply Theorem 33.122 to the two endpoint links $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(h, \mathsf{E1}(a'))$. The resulting $\mathsf{hsame}(a, a')$ gives the payload uniqueness field, completing the second displayed alternative. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryCarrier single valued visible readback`

Theorem 33.194 (Tagged option history classifier exclusive branch readback)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option classifier readback; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is constructor no-confusion plus endpoint carrier exposure; the ledger is the visible absent/present tags and payloads; the core theorem is exclusive classifier branch readback; exactness is branch coverage with disjointness; the standard bridge is* none/some*; and the status is* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, histories* h, k*, and witness* $\gamma : \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$*, either*

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge (\neg\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \wedge (\neg\exists b : \mathsf{BHist}.\ S(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)))$$

or

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge a \sim_S b \wedge \neg\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\mathsf{hsame}(k, \mathsf{e}).$$

Proof. Apply Theorem 33.132 to the classifier witness γ. This splits the proof into the exact absent-absent branch and the exact present-present branch.

In the absent-absent branch, keep the two displayed endpoint equalities. Apply Theorem 33.21 to γ to obtain carrier witnesses for both endpoints. Then apply Theorem 33.128 to each endpoint carrier. The partition result rules out any source payload whose visible present endpoint is hsame-equal to the displayed absent endpoint, so the two negative existential clauses follow.

In the present-present branch, keep the payload histories a, b, the source witnesses $S(a)$ and $S(b)$, the endpoint equalities, and the source classifier witness $a \sim_S b$ exposed by branch exactness. Apply Theorem 33.21 to γ, and then apply Theorem 33.128 to both endpoint carriers. The supplied present payload witnesses rule out the absent alternatives, so the present alternatives of the two carrier partitions give $\neg\mathsf{hsame}(h, \mathsf{e})$ and $\neg\mathsf{hsame}(k, \mathsf{e})$, completing the displayed present-present branch. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier exclusive branch readback`

Theorem 33.195 (Option history classifier exclusive branch iff)**.** *Let source* $: \mathsf{BHist} \to \mathsf{Prop}$ *exclude the empty history: if* $source(h)$ *and* $\mathsf{hsame}(h, \mathsf{e})$ *hold, then contradiction follows. For all histories* h, k*, the untagged option classifier is exactly the disjoint empty-empty branch or the source-source branch:*

$$\mathsf{OptionHistoryClassifier}_{source}(h, k) \iff (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \vee (source(h) \wedge source(k) \wedge \mathsf{hsame}(h, k)).$$

Proof. Unpack the classifier into the two endpoint carriers and the endpoint classifier $\mathsf{hsame}(h, k)$. Each carrier is either the empty branch or the source branch. The empty-empty and source-source cases give the two displayed alternatives directly. In the mixed cases, transport the empty endpoint equality across $\mathsf{hsame}(h, k)$ and apply the hypothesis that *source* excludes empty. Conversely, the empty-empty branch gives both endpoint carriers and reconstructs $\mathsf{hsame}(h, k)$ by transitivity through e; the source-source branch gives the two source carriers and the supplied endpoint classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier exclusive branch iff`

Theorem 33.196 (Option history classifier source branch iff)**.** *Let source* : $\mathsf{BHist} \to \mathsf{Prop}$ *exclude the empty history. For all histories* h, k*, any nullable option classifier witness between* h *and* k *forces the two endpoints to agree on source membership:*

$$\mathsf{OptionHistoryClassifier}_{source}(h, k) \Longrightarrow (source(h) \Longleftrightarrow source(k)).$$

Proof. Unpack the classifier into the endpoint carriers and the stored $\mathsf{hsame}(h, k)$ link. If $source(h)$ holds and the right endpoint carrier is the empty branch, transport that empty equality back across $\mathsf{hsame}(h, k)$ and contradict the empty-exclusion hypothesis; hence the right endpoint is in the source branch. The converse direction is symmetric, transporting an empty left endpoint across the same classifier link. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier source branch iff`

Theorem 33.197 (Option history classifier cross absent-source exclusion)**.** *Let source* : $\mathsf{BHist} \to \mathsf{Prop}$ *exclude the empty history. For histories* h, k*, any nullable option classifier witness from* h *to* k *rules out the two crossed cases:*

$$\mathsf{OptionHistoryClassifier}_{source}(h, k) \Longrightarrow (source(h) \to \mathsf{hsame}(k, \mathsf{e}) \to \bot) \wedge (source(k) \to \mathsf{hsame}(h, \mathsf{e}) \to \bot).$$

Proof. Unpack the classifier into endpoint carriers and the stored $\mathsf{hsame}(h, k)$ link. In the left crossed case, compose $\mathsf{hsame}(h, k)$ with $\mathsf{hsame}(k, \mathsf{e})$ and apply empty exclusion to $source(h)$. In the right crossed case, use symmetry of $\mathsf{hsame}(h, k)$ before composing with $\mathsf{hsame}(h, \mathsf{e})$ and applying empty exclusion to $source(k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier cross absent source exclusion`

Theorem 33.198 (Tagged option history classifier visible-pair payload exactness)**.** *This theorem fits the per-object template as follows: the traditional target is visible payload readback for constructor-tagged* OptionUp*; the BEDC source is* BHist *with* hsame*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligations are presented-payload exactness, constructor no-confusion, and payload determinism; the ledger is the pair of visible present payloads at the two endpoints; the core theorem is the combined exactness statement below; exactness is the visible pair relation together with uniqueness of the displayed payloads; the standard bridge is* none/some*; and the status is* ProofTarget*. For every BEDC history source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, transitivity hypothesis* $\forall x\, y\, z : \mathsf{BHist}.\ x \sim_S y \to y \sim_S z \to x \sim_S z$*, compatibility hypothesis* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and histories* h, k, a, b*, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad S(a), \qquad S(b), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then

$$a \sim_S b \wedge \neg\mathsf{hsame}(h, \mathsf{e}) \wedge \neg\mathsf{hsame}(k, \mathsf{e}) \wedge (\forall a', b' : \mathsf{BHist}.\ S(a') \to S(b') \to \mathsf{hsame}(h, \mathsf{E1}(a')) \to \mathsf{hsame}(k, \mathsf{E1}(b')) \to \mathsf{hsam}$$

Proof. Apply Theorem 33.120 to the classifier witness, the displayed transitivity hypothesis, the compatibility hypothesis, and the two visible endpoint presentations. This gives the source payload relation $a \sim_S b$.

For the left endpoint, suppose $\mathsf{hsame}(h, \mathsf{e})$. Lemma 33.18 applied to $S(a)$, $\mathsf{hsame}(h, \mathsf{e})$, and $\mathsf{hsame}(h, \mathsf{E1}(a))$ eliminates this case, so $\neg\mathsf{hsame}(h, \mathsf{e})$. The same argument at the right endpoint, using $S(b)$ and $\mathsf{hsame}(k, \mathsf{E1}(b))$, gives $\neg\mathsf{hsame}(k, \mathsf{e})$.

Now let a', b' be any other visible pair with $S(a')$, $S(b')$, $\mathsf{hsame}(h, \mathsf{E1}(a'))$, and $\mathsf{hsame}(k, \mathsf{E1}(b'))$. Apply Theorem 33.122 to the two left endpoint presentations to obtain $\mathsf{hsame}(a, a')$, and apply it again to the two right endpoint presentations to obtain $\mathsf{hsame}(b, b')$. These two determinacy facts give the displayed uniqueness clause, completing the combined visible-pair exactness statement. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible pair payload exactness`

Theorem 33.199 (Tagged option history classifier visible endpoint pair readback)**.** *For a BEDC history source carrier S, source classifier $\sim_S$, and histories h, k, a, b, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad \mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then there are source payloads $a_0, b_0 : \mathsf{BHist}$ such that

$$S(a_0) \wedge S(b_0) \wedge \mathsf{hsame}(a, a_0) \wedge \mathsf{hsame}(b, b_0) \wedge a_0 \sim_S b_0.$$

Proof. Split the tagged classifier. In the absent branch, the left endpoint is simultaneously empty and visibly present, contradicting constructor no-confusion. In the present branch, the classifier already supplies source payloads a_0, b_0, their source witnesses, their visible endpoint equations, and the source relation. Strip the common $\mathsf{E1}$ constructor from the two visible endpoint equalities to obtain $\mathsf{hsame}(a, a_0)$ and $\mathsf{hsame}(b, b_0)$, then package the displayed witnesses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier visible endpoint pair readback`

Theorem 33.200 (Tagged option history classifier transported visible payload readback)**.** *For a BEDC history source carrier S, source classifier $\sim_S$, and histories h, k, u, v, x, y, assume*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \qquad \mathsf{hsame}(h, u), \qquad \mathsf{hsame}(k, v),$$

and assume the transported visible endpoints satisfy

$$\mathsf{hsame}(u, \mathsf{E1}(x)), \qquad \mathsf{hsame}(v, \mathsf{E1}(y)).$$

Then there exist payload histories a, b such that

$$S(a) \wedge S(b) \wedge \mathsf{hsame}(x, a) \wedge \mathsf{hsame}(y, b) \wedge a \sim_S b.$$

Proof. Invert the tagged classifier. The absent branch gives $\mathsf{hsame}(h, \mathsf{e})$; transport this along $\mathsf{hsame}(h, u)$, contradicting $\mathsf{hsame}(u, \mathsf{E1}(x))$ by constructor separation. In the present branch, keep the classifier payload witnesses a, b, their source facts, the endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$, $\mathsf{hsame}(k, \mathsf{E1}(b))$, and $a \sim_S b$. Transport the two endpoint presentations along $\mathsf{hsame}(h, u)$ and $\mathsf{hsame}(k, v)$. Constructor injectivity for $\mathsf{E1}$ then gives $\mathsf{hsame}(x, a)$ and $\mathsf{hsame}(y, b)$, which are the required readback fields. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier transport visible payload readback`

33.9 Payload descent image classifier pair readback

Theorem 33.201 (Tagged option payload descent image classifier pair readback is single-valued)**.** *This theorem fits the per-object template as follows: the traditional target is image option pair readback for constructor-tagged* OptionUp*; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is branch exactness plus constructor no-confusion; the ledger is the pair of transported visible payloads; the core theorem is paired exclusive single-valued readback; exactness is the absent pair or the transported present pair; the standard bridge is* none/some *under payload descent; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let cert : $\mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

holds, then either

$$\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e}) \wedge \neg\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{h}$$

or

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge \neg \mathsf{hs}$$

Proof. Apply Theorem 33.162 to the image-classifier witness. This splits the target pair into the exact absent pair or a transported present pair with source payloads a, b.

In the absent branch, keep the two endpoint presentations $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$. If a transported present pair also existed at the same endpoints, the left endpoint would be presented both as e and as $\mathsf{E1}(\delta.\mathsf{map}(a))$, contradicting empty-versus-E1 no-confusion. This gives the displayed negative pair clause.

In the present branch, keep the exposed payloads a, b, their source witnesses, the source classifier witness, the two transported target witnesses, and the two visible endpoint presentations. A simultaneous absent presentation of either endpoint would contradict the corresponding visible present presentation by the same constructor no-confusion argument, so the two non-absent clauses follow.

It remains to prove single-valuedness for any other transported present pair a', b' at the same endpoints. Apply Theorem 33.122 to the two left endpoint presentations of k to obtain $\mathsf{hsame}(\delta.\mathsf{map}(a), \delta.\mathsf{map}(a'))$.

Apply the same theorem to the two right endpoint presentations of k' to obtain $\mathsf{hsame}(\delta.\mathsf{map}(b), \delta.\mathsf{map}(b'))$. These two equalities are exactly the final paired single-valuedness clause. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier pair readback single valued`

Theorem 33.202 (Tagged option payload descent image classifier branch occupancy equivalence)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For target endpoints* $k, k' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

holds, then

$$(\mathsf{hsame}(k, \mathsf{e}) \iff \mathsf{hsame}(k', \mathsf{e})) \wedge ((\exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))) \iff (\exists b : \mathsf{BHist}.\ S(b) \wedge T($$

Proof. Apply Theorem 33.162 to the image-classifier witness. This exposes either the synchronized absent pair or a synchronized present pair.

In the absent branch, the two stored presentations give both directions of the empty-occupancy equivalence. Any displayed transported payload at either endpoint would present that endpoint both as e and as an $\mathsf{E1}$-constructor, contradicting empty-versus-$\mathsf{E1}$ no-confusion. Hence both present-occupancy sides are false, and the present equivalence follows.

In the present branch, keep the displayed payloads a, b, their source carrier witnesses, their transported target carrier witnesses, and the two endpoint presentations. These data give the forward and reverse directions of the present-occupancy equivalence.

It remains to rule out an empty presentation on either side of the present branch. Such a presentation would compose with the corresponding $\mathsf{E1}(\delta.\mathsf{map}(-))$ presentation at the same endpoint and violate empty-versus-$\mathsf{E1}$ no-confusion. Therefore both empty-occupancy sides are false, so the empty equivalence follows. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier branch occupancy equivalence`

Theorem 33.203 (Tagged option payload descent image classifier public endpoint readback)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$, *and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For target endpoints* $k, k' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

holds, then

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k') \wedge \mathsf{Tag}$$

and either

$$\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e}) \wedge (\neg \exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))) \wedge (\neg \exists b : \mathsf{BHist}.\ S(b) \wedge T(\delta.\mathsf{map}($$

or

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge \neg \mathsf{hs}$$

Proof. Apply Theorem 33.165 to the image-classifier witness. This supplies the two public image-carrier witnesses and the target $\mathsf{TaggedOptionHistoryClassifier}_{T, \sim_T}(k, k')$ witness.

Apply Theorem 33.201 to the same image-classifier witness. This gives the synchronized branch split for the endpoint pair.

In the absent branch, keep the two endpoint presentations $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$. A transported visible presentation at the left endpoint would present k both as e and as $\mathsf{E1}(\delta.\mathsf{map}(a))$, contradicting empty-versus-$\mathsf{E1}$ no-confusion. The same argument at the right endpoint rules out every transported visible presentation of k'.

In the present branch, keep the displayed payloads a, b, their source carrier witnesses, the source classifier witness, the two transported target carrier witnesses, the two visible endpoint presentations, and the two non-absence clauses.

It remains only to separate the paired uniqueness field into public endpoint projections. Given any compatible transported visible pair a', b', apply the paired uniqueness clause supplied by Theorem 33.201. Project its left component for the left endpoint statement and its right component for the right endpoint statement. Together with the certificate fields obtained above, these data give the public endpoint readback package. □

Theorem 33.204 (Tagged option payload descent image classifier endpoint transport)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k', u, u' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k'), \qquad \mathsf{hsame}(k, u), \qquad \mathsf{hsame}(k', u'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; u, u').$$

Proof. Apply Theorem 33.162 to the classifier witness at (k, k'). This exposes either the absent branch or a present branch with source payloads a, b.

In the absent branch, compose $\mathsf{hsame}(k, u)$ with the empty presentation of k to obtain $\mathsf{hsame}(u, \mathsf{e})$, and compose $\mathsf{hsame}(k', u')$ with the empty presentation of k' to obtain $\mathsf{hsame}(u', \mathsf{e})$. These two transported empty presentations are the absent branch for the endpoint pair (u, u'), so the reverse direction of branch exactness gives the image-classifier witness at (u, u').

In the present branch, keep the exposed payloads a, b, the source witnesses, the source relation $a \sim_S b$, the transported target witnesses $T(\delta.\mathsf{map}(a))$ and $T(\delta.\mathsf{map}(b))$, and the two visible present presentations at k and k'. Compose the endpoint transports with these presentations to obtain

$$\mathsf{hsame}(u, \mathsf{E1}(\delta.\mathsf{map}(a))) \qquad \text{and} \qquad \mathsf{hsame}(u', \mathsf{E1}(\delta.\mathsf{map}(b))).$$

The same payload data and the transported visible presentations form the present branch for (u, u'), and the reverse direction of branch exactness gives the required image-classifier witness. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier hsame transport`

Theorem 33.205 (Tagged option payload descent image classifier left visible-branch inversion)**.** *This theorem fits the per-object template as follows: the traditional target is one-sided image option readback from the left visible endpoint; the BEDC source is* BHist *with a* $\mathsf{DescentCertificate}$*; the carrier is* $\mathsf{TaggedOptionPayloadDescentImageCarrier}$*; the classifier is* $\mathsf{TaggedOptionPayloadDescentImageClassifier}$*; the stability obligation is branch exactness plus constructor no-confusion; the ledger is the left visible target payload together with the transported right endpoint; the core theorem is left visible-branch*

inversion; exactness is absent propagation or transported present-payload recovery; the standard bridge is none/some *under payload descent; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$*, the absent clause is*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{hsame}(k, \mathsf{e}) \to \mathsf{hsame}(k', \mathsf{e}).$$

For every payload history $a : \mathsf{BHist}$*, the present clause is*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to T(a) \to \mathsf{hsame}(k, \mathsf{E1}(a)) \to \exists b, c : \mathsf{BHist}.\ S(b) \wedge S(c) \wedge$$

Proof. Apply Theorem 33.162 to the image-classifier witness. This rewrites the witness into the disjunction whose absent side presents both target endpoints as e and whose present side exposes source payloads b, c together with the transported endpoint presentations.

For the absent clause, split that disjunction. The absent side gives $\mathsf{hsame}(k', \mathsf{e})$ directly. The present side also presents k as $\mathsf{E1}(\delta.\mathsf{map}(b))$ for some source payload b; composing this with the displayed witness $\mathsf{hsame}(k, \mathsf{e})$ gives empty-versus-E1 no-confusion, so the mixed branch is impossible.

For the present clause, split the same disjunction. The absent side presents k as e, contradicting the displayed witness $\mathsf{hsame}(k, \mathsf{E1}(a))$ by empty-versus-E1 no-confusion.

In the present side, keep the exposed source payloads b, c. The branch-exactness data already supplies $S(b)$, $S(c)$, $b \sim_S c$, $T(\delta.\mathsf{map}(b))$, $T(\delta.\mathsf{map}(c))$, and $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(c)))$.

Compose the left-endpoint presentations $\mathsf{hsame}(k, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$, and invert the E1 constructor to obtain $\mathsf{hsame}(a, \delta.\mathsf{map}(b))$. These fields are exactly the displayed existential witness. The hypothesis $T(a)$ records the visible left payload as a target carrier payload and is preserved as part of the public premise. □

Theorem 33.206 (Tagged option payload descent image classifier self-exactness)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For every target endpoint* $k : \mathsf{BHist}$*, self-classification by the image classifier is exactly membership in the payload descent image carrier:*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k) \iff \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta;$$

Proof. For the forward direction, apply Theorem 33.162 to the self-classifier witness at (k, k). This exposes either the absent pair or a present pair whose two visible endpoints are both k.

In the absent pair branch, the left endpoint presentation gives $\mathsf{hsame}(k, \mathsf{e})$. The reverse direction of Theorem 33.157 sends this absent carrier branch to $\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k)$.

In the present pair branch, keep the first exposed payload a together with $S(a)$, $T(\delta.\mathsf{map}(a))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. These data form the present carrier branch, so the reverse direction of Theorem 33.157 gives the image-carrier witness at k.

For the reverse direction, apply Theorem 33.157 to the image-carrier witness at k. This exposes either the absent endpoint or a transported present payload at the endpoint.

In the absent carrier branch, use the same empty presentation for both sides of the pair. The reverse direction of Theorem 33.162 then rebuilds the absent/absent image-classifier branch at (k, k).

In the present carrier branch, unpack $a : \mathsf{BHist}$, $S(a)$, $T(\delta.\mathsf{map}(a))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. Use the same payload on both sides; the reflexivity field of cert supplies $a \sim_S a$, and the same transported target witness and endpoint presentation supply both visible endpoints. The reverse direction of classifier branch exactness therefore gives $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier self exactness`

Theorem 33.207 (Tagged option payload descent image classifier composition factorization)**.** *This theorem fits the per-object template as follows: the traditional target is composition of classified option images; the BEDC source is* BHist *with two* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is factoring two composite* TaggedOptionMapRel *witnesses through shared intermediate endpoints; the ledger is the paired intermediate target endpoints and transported visible payloads; the core theorem is classifier-level composition factorization; exactness is the displayed equivalence; the standard bridge is* none/some *functorial image composition; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be source classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume also endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $p : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(p))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

For every pair of final endpoints $m, m' : \mathsf{BHist}$, the composite image classifier factors through paired intermediate tagged option endpoints:

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m') \iff \exists k, k' : \mathsf{BHist}.\ \mathsf{TaggedOptionPayloadDescentIm}$$

Proof. For the forward direction, unfold Definition 33.161. This exposes source endpoints h, h', a source tagged classifier witness, and two composite map-relation witnesses from h to m and from h' to m'.

Apply Theorem 33.155 in the forward direction to the first composite map-relation witness and η_ϵ. It supplies an intermediate endpoint k, a δ-map witness from h to k, and an ϵ-map witness from k to m.

Apply the same exactness theorem to the second composite map-relation witness. This supplies k', a δ-map witness from h' to k', and an ϵ-map witness from k' to m'. Repack the source tagged classifier witness with the two δ-map witnesses to obtain the δ-image-classifier factor, and keep the two ϵ-map witnesses as the displayed endpoint factors.

For the reverse direction, unpack the displayed factorization into endpoints k, k', a δ-image-classifier witness, and the two ϵ-map witnesses from k to m and from k' to m'.

Unfold Definition 33.161 for the δ-image-classifier witness. This exposes source endpoints h, h', a source tagged classifier witness, and δ-map witnesses from h to k and from h' to k'.

Apply Theorem 33.155 in the reverse direction to the left pair of map witnesses, and apply it again to the right pair. The two results are composite $\mathsf{TaggedOptionMapRel}_{\epsilon\circ\delta}$ witnesses from h to m and from h' to m'. Repacking these with the same source tagged classifier witness gives the composite image-classifier witness. □

Theorem 33.208 (Tagged option payload descent image classifier stability package)**.** *This theorem fits the per-object template as follows: the traditional target is certificate-facing stability for the option payload descent image classifier; the BEDC source is* BHist *with a payload descent certificate; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier *the stability obligation is self-exactness, symmetry, transitivity, endpoint transport, and endpoint carrier exposure; the ledger is the classified target endpoints and the two transported source payload presentations; the core theorem is the public stability package; exactness is the self-classifier equivalence and the four closure fields below; the standard bridge is* none/some *under tagged option payload descent; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

Then the option payload descent image classifier supplies the following five certificate-facing fields.

1. *For every target endpoint* $k : \mathsf{BHist}$*,*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k) \iff \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T,$$

2. *For target endpoints* $k, k' : \mathsf{BHist}$*,*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T$$

3. *For target endpoints* $k, \ell, m : \mathsf{BHist}$*,*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell) \to \mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T,$$

4. *For target endpoints* $k, k', u, u' : \mathsf{BHist}$,

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{hsame}(k, u) \to \mathsf{hsame}(k', u') \to \mathsf{TaggedOptionPayl}$$

5. *For target endpoints* $k, k' : \mathsf{BHist}$,

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho$$

Proof. For the self-exactness field, apply Theorem 33.206 at the chosen endpoint k. Its two directions are exactly the displayed equivalence between the self-classifier witness and image-carrier membership.

For symmetry and carrier exposure, apply Theorem 33.166. Its second field gives the reversed classifier witness for classified endpoints, and its third field gives the two image-carrier witnesses exposed by any classifier witness.

For transitivity, apply Theorem 33.168 to the two classifier witnesses at (k, ℓ) and (ℓ, m), using the stated source hsame compatibility and source-reflection hypotheses.

For endpoint transport, apply Theorem 33.204 to the classifier witness at (k, k') and the two displayed hsame endpoint witnesses. Its conclusion is the transported classifier witness at (u, u'). □

Theorem 33.209 (Tagged option payload descent image classifier reflective transitive visible payload readback)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For target endpoints $k, \ell, m : \mathsf{BHist}$ *and visible target payloads* $u, v : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, \ell)$$

and

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; \ell, m)$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(m, \mathsf{E1}(v)),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(b)).$$

Proof. First combine the two image-classifier witnesses. Apply Theorem 33.168 to the witnesses at (k, ℓ) and (ℓ, m), using the displayed source hsame compatibility and source-reflection hypotheses. This gives the direct image-classifier witness from k to m.

Next apply Theorem 33.164 to that direct witness and to the visible endpoint presentations $T(u)$, $T(v)$, $\mathsf{hsame}(k, \mathsf{E1}(u))$, and $\mathsf{hsame}(m, \mathsf{E1}(v))$. The produced source payloads a, b carry $S(a)$, $S(b)$, $a \sim_S b$, the transported target carrier facts, and the two payload readback equalities, which are exactly the displayed existential conclusion. □

Theorem 33.210 (Tagged option payload descent image semantic certificate under reflection)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*, and assume*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

Then the payload descent image carrier and image classifier form the image-level semantic certificate

$\mathsf{SemanticNameCert}(\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta), \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta($

Proof. Build the core NameCert for the image carrier and image classifier. For inhabitation, use the absent branch of Theorem 33.157 at e: reflexivity of hsame gives the displayed absent endpoint, and the reverse direction of branch exactness gives the image-carrier witness at e.

For reflexivity of the core classifier, apply Theorem 33.206 to the image-carrier witness at the endpoint. For symmetry, use the symmetry field of Theorem 33.166.

For transitivity, use Theorem 33.168 with the displayed reflection hypothesis. For carrier respect, use the endpoint carrier exposure field of Theorem 33.166 and project the second image-carrier endpoint from the classified pair.

These five clauses assemble $\mathsf{NameCert}(\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta), \mathsf{TaggedOptionPayload}$

Finally, set the source, pattern, and ledger predicates in SemanticNameCert all equal to the same payload descent image carrier. The pattern and ledger soundness maps are identity projections, so the core NameCert just built repacks as the displayed semantic certificate. □

33.10 Payload descent image classifier visible payload exactness

Theorem 33.211 (Tagged option payload descent image classifier visible payload exactness)**.** *Let* $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$ *and visible target payloads* $u, v : \mathsf{BHist}$*, the visible endpoint classifier is exact:*

$$(\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \wedge T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(k', \mathsf{E1}(v))) \iff$$

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(k', \mathsf{E1}(v)) \wedge \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}$$

Proof. For the forward direction, keep the visible target facts $T(u)$, $T(v)$, $\mathsf{hsame}(k, \mathsf{E1}(u))$, and $\mathsf{hsame}(k', \mathsf{E1}(v))$. Apply Theorem 33.164 to the classifier witness and these visible facts. This supplies source payloads a, b with the displayed source carrier witnesses, source classifier witness, transported target witnesses, and payload identifications.

Combining the retained visible facts with that existential witness gives the right-hand side of the equivalence.

For the reverse direction, unpack the right-hand side. Keep the visible target facts and choose witnesses a, b from the existential clause.

Compose $\mathsf{hsame}(k, \mathsf{E1}(u))$ with the $\mathsf{E1}$ congruence of $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ to obtain $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. The same transitivity and congruence argument at the right endpoint gives $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b)))$.

The source witnesses $S(a)$ and $S(b)$, the relation $a \sim_S b$, the transported target witnesses, and these two transported endpoint presentations form the present branch in Theorem 33.162. Apply its reverse direction to rebuild $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$. Together with the retained visible facts, this gives the left-hand side. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier visible payload exactness`

33.11 Payload descent image classifier visible pair coherence

Theorem 33.212 (Tagged option payload descent image classifier visible payload pair coherence)**.** *This theorem fits the per-object template as follows: the traditional target is visible readback coherence for constructor-tagged* OptionUp *under payload descent; the BEDC source is* BHist *with a* DescentCertificate*; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is visible endpoint determinacy plus payload readback; the ledger is the displayed target payload pair; the core theorem is the shared-source-pair coherence below; exactness says one source pair explains all visible target presentations; the standard bridge is* none/some *payload readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$ *and* $\sim_T$ *be source classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$ *and visible target payloads* $u, v, u', v' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

and

$$T(u) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(u)) \wedge \mathsf{hsame}(k', \mathsf{E1}(v))$$

and

$$T(u') \wedge T(v') \wedge \mathsf{hsame}(k, \mathsf{E1}(u')) \wedge \mathsf{hsame}(k', \mathsf{E1}(v')),$$

then

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u, \delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(v, \delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(u', \delta.\mathsf{m}$$

Proof. Apply Theorem 33.164 to the image-classifier witness and the first visible pair. This supplies source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the transported target witnesses, and the two payload identifications $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$.

Compare the two visible presentations of the left endpoint k. The endpoint is presented both as $\mathsf{E1}(u)$ and as $\mathsf{E1}(u')$; $\mathsf{E1}$ no-confusion and symmetry of hsame give $\mathsf{hsame}(u', u)$, and composition with $\mathsf{hsame}(u, \delta.\mathsf{map}(a))$ gives $\mathsf{hsame}(u', \delta.\mathsf{map}(a))$.

Apply the same comparison at the right endpoint k'. The two presentations $\mathsf{E1}(v)$ and $\mathsf{E1}(v')$ determine $\mathsf{hsame}(v', v)$, and composing with $\mathsf{hsame}(v, \delta.\mathsf{map}(b))$ gives $\mathsf{hsame}(v', \delta.\mathsf{map}(b))$.

The original source payloads a, b, together with the two readback identifications and the two transported primed identifications, form the required shared-source-pair witness. □

Theorem 33.213 (Tagged option composite image classifier visible payload factorization)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *visible payload readback; the BEDC source is* BHist *with two* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is composition factorization plus visible payload readback; the ledger is final visible payloads and intermediate tagged endpoints; the core theorem is the factorized readback below; exactness says composite classified final endpoints are witnessed by one intermediate classified pair; the standard bridge is* none/some *functorial payload readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be source classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $p : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(p))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For final endpoints* $m, m' : \mathsf{BHist}$ *and visible final payloads* $x, y : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y)),$$

then there exist $k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m')$*;*
3. $T(p)$*,* $T(q)$*,* $U(\epsilon.\mathsf{map}(p))$*, and* $U(\epsilon.\mathsf{map}(q))$*;*
4. $\mathsf{hsame}(k, \mathsf{E1}(p))$*,* $\mathsf{hsame}(k', \mathsf{E1}(q))$*,* $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$*, and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$*;*
5. $S(a)$*,* $S(b)$*,* $a \sim_S b$*,* $T(\delta.\mathsf{map}(a))$*, and* $T(\delta.\mathsf{map}(b))$*;*
6. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$*.*

Proof. Apply Theorem 33.207 to the composite image-classifier witness. This supplies intermediate endpoints k, k', the δ image-classifier factor at (k, k'), and the two ϵ map-relation witnesses from k to m and from k' to m'.

Use Theorem 33.138 on the two ϵ map-relation witnesses. The visible final presentations $\mathsf{hsame}(m, \mathsf{E1}(x))$ and $\mathsf{hsame}(m', \mathsf{E1}(y))$ force visible intermediate presentations, so obtain payloads p, q with $T(p)$, $T(q)$, $\mathsf{hsame}(k, \mathsf{E1}(p))$, and $\mathsf{hsame}(k', \mathsf{E1}(q))$.

Apply Theorem 33.139 to the left ϵ map-relation witness, the intermediate presentation $\mathsf{hsame}(k, \mathsf{E1}(p))$, and the final presentation $\mathsf{hsame}(m, \mathsf{E1}(x))$. It identifies x with $\epsilon.\mathsf{map}(p)$ and records the final carrier witness $U(\epsilon.\mathsf{map}(p))$.

Apply the same readback theorem to the right ϵ map-relation witness. This identifies y with $\epsilon.\mathsf{map}(q)$ and records $U(\epsilon.\mathsf{map}(q))$.

Now apply Theorem 33.164 to the δ image-classifier factor and the visible intermediate pair $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$. It gives source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the transported target-carrier witnesses, and $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$.

Package the endpoints k, k', the intermediate payloads p, q, and the source payloads a, b with the retained factorization witnesses and the four readback identifications. These are exactly the six displayed clauses. □

33.12 Payload descent image classifier public constructor normal form

Theorem 33.214 (Tagged option payload descent image classifier public constructor normal form)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-facing public image readback for* OptionUp *payload descent; the BEDC source is* BHist *with one* DescentCertificate *map; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is certificate exposure, endpoint transport, and zero-tag exclusion; the ledger records the two public image endpoints, their target classifier witness, the absence of unused zero endpoints, and the absent-or-present readback split; the core theorem is the displayed constructor normal form; exactness says every classified image pair is either the absent public pair or a single-valued transported present pair; the standard bridge is* none/some *public readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$$

holds, then the following clauses hold simultaneously.

1. *Both endpoints are public image endpoints:*

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_\delta(S, T, \rho_\delta; k').$$

2. *The endpoints are classified by the target tagged option classifier:*

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

3. *Every zero-tag presentation is impossible at either endpoint:*

$$\forall t : \mathsf{BHist}.\ (\mathsf{hsame}(k, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(k', \mathsf{E0}(t)) \to \bot).$$

4. *The public readback is absent-or-present and single-valued: either*

$$\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e}) \wedge \neg\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$$

or

$$\exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b))) \wedge$$

Proof. First expose the public image endpoints and the target classifier. Apply Theorem 33.165 to the image-classifier witness, using the displayed descent certificate, payload preservation, source certificate, and source compatibility. This gives the two image-carrier clauses and the target $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$ witness.

Next obtain the readback split. Apply Theorem 33.201 to the same image-classifier witness and the same source data. The resulting absent branch and transported present branch are exactly the fourth clause.

It remains to exclude zero-tag endpoints. Fix a tail t and suppose $\mathsf{hsame}(k, \mathsf{E0}(t))$. Transport the exposed target classifier along this left-endpoint witness and reflexivity on k', using Theorem 33.34. The transported classifier has $\mathsf{E0}(t)$ as its left endpoint, so Theorem 33.124 closes the contradiction.

For the right endpoint, suppose $\mathsf{hsame}(k', \mathsf{E0}(t))$. Transport the same target classifier along reflexivity on k and the displayed right-endpoint witness, again using Theorem 33.34. The transported classifier has $\mathsf{E0}(t)$ as its right endpoint, and Theorem 33.124 gives the second contradiction. Packaging these two functions for arbitrary t gives the zero-tag exclusion clause. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier public constructor normal form`

Theorem 33.215 (Tagged option composite image carrier zero-endpoint exclusion)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-shape exclusion for composite* OptionUp *image carriers; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is not used in the statement; the stability obligation is branch exactness plus generated-history no-confusion; the ledger records the absence of the unused zero tag; the core theorem is the displayed implication; exactness says every composite image carrier endpoint is absent or one-tagged; the standard bridge is* none/some *readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

For every final endpoint $m : \mathsf{BHist}$ and tail $t : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m)$$

holds, then

$$\mathsf{hsame}(m, \mathsf{E0}(t)) \to \bot.$$

Proof. Apply Theorem 33.157 to the composite image-carrier witness. The branch split says that the final endpoint is either absent or presented by a transported one-tagged payload.

In the absent branch, suppose $\mathsf{hsame}(m, \mathsf{E0}(t))$. The branch presentation gives $\mathsf{hsame}(m, \mathsf{e})$. Compose symmetry of this empty presentation with the assumed zero presentation to obtain $\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(t))$, contradicting empty-versus-E0 generated-history no-confusion.

In the present branch, unpack a source payload a together with $S(a)$, $U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$, and $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))))$. If $\mathsf{hsame}(m, \mathsf{E0}(t))$ also held, symmetry and transitivity of hsame would identify the displayed E1 endpoint with the zero endpoint, contradicting E1-versus-E0 generated-history no-confusion.

The two branches both return a contradiction from $\mathsf{hsame}(m, \mathsf{E0}(t))$, giving the displayed endpoint exclusion. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageCarrier composite zero endpoint exclusion`

Theorem 33.216 (Tagged option composite image classifier zero-endpoint exclusion)**.** *This theorem fits the per-object template as follows: the traditional target is public constructor-shape exclusion for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is public branch classification plus generated-history no-confusion; the ledger records that a final endpoint cannot be zero-tagged; the core theorem is the displayed pair of endpoint contradictions; exactness says the composite public readback has only absent or one-tagged final shapes; the standard bridge is* none/some *public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints $m, m' : \mathsf{BHist}$ and any tail $t : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

holds, then

$$(\mathsf{hsame}(m, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(m', \mathsf{E0}(t)) \to \bot).$$

Proof. Apply Theorem 33.190 to the composite image-classifier witness. Its conclusion splits the final readback into an absent branch and a present branch.

For the left endpoint m, suppose $\mathsf{hsame}(m, \mathsf{E0}(t))$. In the absent branch, the classification gives $\mathsf{hsame}(m, \mathsf{e})$. Symmetry and transitivity of hsame compose this empty presentation with the assumed zero presentation, contradicting history no-confusion between e and $\mathsf{E0}(t)$.

In the present branch, the classification gives a visible final payload x with $\mathsf{hsame}(m, \mathsf{E1}(x))$. Symmetry and transitivity of hsame compose that visible presentation with the assumed zero presentation, contradicting $\mathsf{E1}$-versus-$\mathsf{E0}$ no-confusion.

For the right endpoint m', suppose $\mathsf{hsame}(m', \mathsf{E0}(t))$. In the absent branch, the classification gives $\mathsf{hsame}(m', \mathsf{e})$. Symmetry and transitivity of hsame compose this empty presentation with the assumed zero presentation, contradicting history no-confusion between e and $\mathsf{E0}(t)$.

In the present branch, the classification gives a visible final payload y with $\mathsf{hsame}(m', \mathsf{E1}(y))$. Symmetry and transitivity of hsame compose that visible presentation with the assumed zero presentation, contradicting $\mathsf{E1}$-versus-$\mathsf{E0}$ no-confusion.

Package the two endpoint contradiction functions in the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite zero endpoint exclusion`

33.13 Composite image classifier public readback single-valuedness

Theorem 33.217 (Tagged option composite image classifier public readback single-valuedness)**.** *This theorem fits the per-object template as follows: the traditional target is public readback for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is branch coverage together with visible-payload determinacy; the ledger records the absent final pair or the uniquely determined visible final payload pair; the core theorem is the displayed exhaustive disjunction with target classification, endpoint non-absence, and competing-readback uniqueness; exactness says public final endpoints have a single classified visible payload reading when they are present; the standard bridge is* none/some *public readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou *). For final endpoints* $m, m' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

holds, then either

$$\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(m', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y))$$

or there exist $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y)) \wedge x \sim_U y \wedge \neg\mathsf{hsame}(m, \mathsf{e}) \wedge \neg\mathsf{hsame}(m', \mathsf{e})$$

and

$$\forall x', y' : \mathsf{BHist}.\ U(x') \to U(y') \to \mathsf{hsame}(m, \mathsf{E1}(x')) \to \mathsf{hsame}(m', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. Apply Theorem 33.190 to the composite image-classifier witness, using the same descent certificates, payload preservation fields, endpoint stability, name certificates, and compatibility hypotheses. This gives the absent/present split for the final public endpoints.

In the absent branch, keep exactly the returned endpoint witnesses $\mathsf{hsame}(m, \mathsf{e})$ and $\mathsf{hsame}(m', \mathsf{e})$ together with the returned statement that no visible final payload pair exists. These three facts form the first disjunct.

In the present branch, project the visible final payload pair x, y from the branch-classification conclusion. Its carrier facts and visible endpoint presentations provide the visible part of the second disjunct.

Apply Theorem 33.192 to the same composite image-classifier witness, using the selected visible pair x, y as both readings. Project target classification $x \sim_U y$ and the two non-absence conclusions $\neg\mathsf{hsame}(m, \mathsf{e})$ and $\neg\mathsf{hsame}(m', \mathsf{e})$.

It remains to show single-valued public readback. Let x', y' be any competing visible final payloads with $U(x')$, $U(y')$, $\mathsf{hsame}(m, \mathsf{E1}(x'))$, and $\mathsf{hsame}(m', \mathsf{E1}(y'))$. Apply the visible-payload determinacy theorem again to the selected visible pair x, y and the competing visible pair x', y'. The result gives $\mathsf{hsame}(x, x')$, $\mathsf{hsame}(y, y')$, and $x' \sim_U y'$.

Return the two endpoint-determinacy witnesses and the competing target classification $x' \sim_U y'$ for the universal clause, and combine that clause with the selected visible-payload data, target classification, and non-absence facts. □

Theorem 33.218 (Tagged option composite image classifier nonempty public readback equivalence)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *public present-readback criterion; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are branch coverage plus visible-payload determinacy and mixed-tag separation; the ledger is absent/present final endpoint readback; the core theorem is the nonempty iff plus a unique classified visible pair; exactness says the present public branch is equivalent to a nonempty endpoint; the standard bridge is* none/some *functorial readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

holds, then the following three assertions hold:

1. $\neg\mathsf{hsame}(m, \mathsf{e})$ *if and only if* $\neg\mathsf{hsame}(m', \mathsf{e})$*;*
2. $\neg\mathsf{hsame}(m, \mathsf{e})$ *if and only if there exist* $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y));$$

3. *if* $\neg\mathsf{hsame}(m, \mathsf{e})$*, then there exist* $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(m, \mathsf{E1}(x)) \wedge \mathsf{hsame}(m', \mathsf{E1}(y)) \wedge x \sim_U y$$

and

$$\forall x', y' : \mathsf{BHist}.\ U(x') \to U(y') \to \mathsf{hsame}(m, \mathsf{E1}(x')) \to \mathsf{hsame}(m', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. Apply Theorem 33.217 to the composite image-classifier witness and the displayed descent, preservation, stability, certificate, and compatibility data. This gives an absent branch or a present branch with a selected visible final payload pair and the universal competing-readback clause.

In the absent branch, both endpoint presentations are empty. Hence $\neg\mathsf{hsame}(m, \mathsf{e})$ and $\neg\mathsf{hsame}(m', \mathsf{e})$ are both contradictory, and the returned no-visible-pair clause rules out the existential visible readback in the second assertion.

For the reverse implication in the second assertion, suppose a visible pair x, y exists. The visible-payload determinacy theorem applied to this visible pair against itself gives the non-absence conclusions $\neg\mathsf{hsame}(m, \mathsf{e})$ and $\neg\mathsf{hsame}(m', \mathsf{e})$. These also give the endpoint nonempty equivalence in the visible case.

In the present branch, keep the selected visible pair x, y. The branch already provides $U(x)$, $U(y)$, the two visible endpoint presentations, $x \sim_U y$, the two non-absence facts, and the competing-readback uniqueness clause.

These facts prove both directions of the visible-existence equivalence. Under the nonempty hypothesis, the same selected x, y witness the classified visible pair and carry the universal uniqueness conclusion. □

Theorem 33.219 (Tagged option composite image classifier transported nonempty public readback)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *public readback stable under endpoint normalization; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligation is endpoint* hsame *transport plus public readback exactness; the ledger is transported absent/present final endpoint readback; the core theorem is the transported classifier together with the nonempty iff and unique classified visible pair at the transported endpoints; exactness says transported present readback is equivalent to endpoint nonemptiness; the standard bridge is* none/some *endpoint-normalized public readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. *Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints* $m, m', n, n' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, n) \wedge \mathsf{hsame}(m', n'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; n, n')$$

holds, and the following three assertions hold:

1. $\neg\mathsf{hsame}(n, \mathsf{e})$ *if and only if* $\neg\mathsf{hsame}(n', \mathsf{e})$*;*
2. $\neg\mathsf{hsame}(n, \mathsf{e})$ *if and only if there exist* $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(n, \mathsf{E1}(x)) \wedge \mathsf{hsame}(n', \mathsf{E1}(y));$$

3. *if* $\neg\mathsf{hsame}(n, \mathsf{e})$, *then there exist* $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(n, \mathsf{E1}(x)) \wedge \mathsf{hsame}(n', \mathsf{E1}(y)) \wedge x \sim_U y$$

and

$$\forall x', y' : \mathsf{BHist}.\ U(x') \to U(y') \to \mathsf{hsame}(n, \mathsf{E1}(x')) \to \mathsf{hsame}(n', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. First transport the composite image-classifier witness from (m, m') to (n, n'). Apply Theorem 33.204 with the endpoint witnesses $\mathsf{hsame}(m, n)$ and $\mathsf{hsame}(m', n')$ to obtain the displayed classifier at the transported endpoints.

Next apply Theorem 33.218 to the transported composite classifier at (n, n'), using the same descent certificates, preservation fields, endpoint stability, name certificates, and compatibility hypotheses.

The three projected conclusions are exactly the normalized endpoint nonempty equivalence, the visible-pair existence equivalence, and the classified visible pair with the universal competing-readback clause, all indexed by n, n'. □

Theorem 33.220 (Tagged option composite image classifier normalized public readback single–valuedness)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized public readback for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are* hsame *endpoint transport plus public readback single-valuedness; the ledger records the normalized absent pair or the uniquely determined normalized visible final payload pair; the core theorem is the displayed normalized public certificate and branch split; exactness says normalized public endpoints have one classified visible payload reading when present; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou*). For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then

$$\mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; u) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; u$$

and either

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

or there exist $x, y : \mathsf{BHist}$ *such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \wedge x \sim_U y \wedge \neg\mathsf{hsame}(u, \mathsf{e}) \wedge \neg\mathsf{hsame}(u', \mathsf{e})$$

and

$$\forall x', y' : \mathsf{BHist}.\ U(x') \to U(y') \to \mathsf{hsame}(u, \mathsf{E1}(x')) \to \mathsf{hsame}(u', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. First transport the composite image-classifier witness from (m, m') to (u, u'). Apply Theorem 33.204 with the endpoint witnesses $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$.

Next expose the normalized public certificate fields. Apply Theorem 33.165 to the transported classifier at (u, u'). This gives the two normalized image carriers and $\mathsf{TaggedOptionHistoryClassifier}_{U,\sim_U}(u, u')$.

Finally apply Theorem 33.217 to the transported composite classifier. Its absent branch gives the normalized empty pair and excludes every U-visible final payload pair at (u, u').

In the present branch, keep the visible normalized payloads x, y, their carrier facts, endpoint presentations, target classification, and non-absence clauses. The returned competing-readback clause is already indexed by the normalized endpoints u, u', so conjoin it with the exposed carrier and classifier fields. □

Theorem 33.221 (Tagged option composite image classifier normalized nonempty public readback equivalence)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *public present-readback exactness; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint transport, branch coverage, and visible-payload determinacy; the ledger is the normalized absent/present final endpoint split; the core theorem is the normalized nonempty iff plus a unique classified visible pair; exactness says the normalized present public branch is equivalent to a nonempty normalized endpoint; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\, T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following three assertions hold:

1. $\neg\mathsf{hsame}(u, \mathsf{e})$ *if and only if* $\neg\mathsf{hsame}(u', \mathsf{e})$*;*
2. $\neg\mathsf{hsame}(u, \mathsf{e})$ *if and only if there exist $x, y : \mathsf{BHist}$ such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y));$$

3. *if* $\neg\mathsf{hsame}(u, \mathsf{e})$*, then there exist $x, y : \mathsf{BHist}$ such that*

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \wedge x \sim_U y$$

and

$$\forall x', y' : \mathsf{BHist}.\, U(x') \to U(y') \to \mathsf{hsame}(u, \mathsf{E1}(x')) \to \mathsf{hsame}(u', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. Apply Theorem 33.220 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives the normalized public carrier and classifier certificate, together with either an absent branch or a present branch.

In the absent branch, the theorem gives $\mathsf{hsame}(u, \mathsf{e})$, $\mathsf{hsame}(u', \mathsf{e})$, and the no-visible-pair clause at (u, u'). Therefore both endpoint non-absence claims are contradictory, which proves the normalized endpoint nonempty equivalence in this branch.

For the forward direction of the visible-readback equivalence, the same contradiction from $\mathsf{hsame}(u, \mathsf{e})$ rules out the hypothesis $\neg\mathsf{hsame}(u, \mathsf{e})$. For the reverse direction, any proposed visible pair is excluded by the no-visible-pair clause.

In the present branch, keep the selected visible normalized pair x, y. The branch provides $U(x)$, $U(y)$, the two normalized endpoint presentations, $x \sim_U y$, the two non-absence facts, and the universal competing-readback uniqueness clause.

The non-absence facts prove both directions of $\neg\mathsf{hsame}(u, \mathsf{e})$ if and only if $\neg\mathsf{hsame}(u', \mathsf{e})$. They also prove the forward direction of the visible-readback equivalence by the selected pair x, y, while the reverse direction follows from the branch's $\neg\mathsf{hsame}(u, \mathsf{e})$ conclusion.

Under the hypothesis $\neg\mathsf{hsame}(u, \mathsf{e})$, return the same selected x, y. Their carrier facts, endpoint presentations, target classification, and competing-readback uniqueness clause give the classified visible pair required by the third assertion. □

Theorem 33.222 (Tagged option composite image classifier normalized branch occupancy equivalence)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *public branch occupancy; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint transport, synchronized absent coverage, and visible-payload single-valuedness; the ledger is the normalized empty/visible final endpoint split; the core theorem is synchronized normalized empty occupancy, synchronized normalized visible occupancy, and empty-versus-visible exclusion on both endpoints; exactness says constructor occupancy is invariant under normalized public readback; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou
). For final endpoints $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then

$$(\mathsf{hsame}(u, \mathsf{e}) \iff \mathsf{hsame}(u', \mathsf{e})) \wedge ((\exists x : \mathsf{BHist}.\ U(x) \wedge \mathsf{hsame}(u, \mathsf{E1}(x))) \iff (\exists y : \mathsf{BHist}.\ U(y) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)))) \wedge (\mathsf{h}$$

Proof. Apply Theorem 33.220 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives the normalized public carrier and classifier certificate together with an absent branch or a present branch.

In the absent branch, the returned clauses give $\mathsf{hsame}(u, \mathsf{e})$, $\mathsf{hsame}(u', \mathsf{e})$, and exclusion of every paired visible U-readback at (u, u'). The two empty presentations prove both directions of the empty-occupancy equivalence.

For the visible-occupancy equivalence in the absent branch, unpack a visible witness on either endpoint. On the left, $\mathsf{hsame}(u, \mathsf{e})$ together with $\mathsf{hsame}(u, \mathsf{E1}(x))$ composes to $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(x))$, contradicting *not_hsame_emp_e*1. On the right, $\mathsf{hsame}(u', \mathsf{E1}(y))$ together with $\mathsf{hsame}(u', \mathsf{e})$ composes to $\mathsf{hsame}(\mathsf{E1}(y), \mathsf{e})$, contradicting *not_hsame_e*1*_empty*. Hence both visible sides are false in this branch.

The endpoint-specific exclusion clauses follow from the same contradictions: the left implication uses *not_hsame_emp_e*1, and the right implication uses *not_hsame_e*1*_empty*.

In the present branch, keep the selected visible normalized payloads x, y. Their carrier facts and endpoint presentations give the forward and reverse directions of the visible-occupancy equivalence, while the stored non-absence facts contradict any attempted empty presentation of either endpoint.

It remains only to repack the empty-versus-visible exclusions in the present branch. An empty presentation at u would contradict $\mathsf{hsame}(u, \mathsf{E1}(x))$ via *not_hsame_emp_e*1, and an empty presentation at u' would contradict $\mathsf{hsame}(u', \mathsf{E1}(y))$ via *not_hsame_e*1*_empty*. Therefore the empty-occupancy equivalence and both exclusion implications hold. □

Theorem 33.223 (Tagged option composite image classifier normalized one-sided visible branch synchronization)**.** *This theorem fits the per-object template as follows: the traditional target is one-sided visible readback for composite* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, branch occupancy, visible payload determinacy, and target certificate transitivity; the ledger is one visible endpoint plus the recovered opposite visible endpoint and payload uniqueness; the core theorem is synchronized one-sided visible branch readback; exactness is branch occupancy plus visible payload determinacy; the standard bridge is* none/some *public readback; and the status is* ProofTarget*. Let* $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following two one-sided visible branch clauses hold:

1. *For every* $x : \mathsf{BHist}$*, if* $U(x)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x))$*, then there exists* $y : \mathsf{BHist}$ *such that* $U(y)$*,* $\mathsf{hsame}(u', \mathsf{E1}(y))$*,* $x \sim_U y$*,* $\neg\mathsf{hsame}(u, \mathsf{e})$*,* $\neg\mathsf{hsame}(u', \mathsf{e})$*, and for every* $y' : \mathsf{BHist}$ *with* $U(y')$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$*, both* $\mathsf{hsame}(y, y')$ *and* $x \sim_U y'$ *hold.*
2. *For every* $y : \mathsf{BHist}$*, if* $U(y)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y))$*, then there exists* $x : \mathsf{BHist}$ *such that* $U(x)$*,* $\mathsf{hsame}(u, \mathsf{E1}(x))$*,* $x \sim_U y$*,* $\neg\mathsf{hsame}(u, \mathsf{e})$*,* $\neg\mathsf{hsame}(u', \mathsf{e})$*, and for every* $x' : \mathsf{BHist}$ *with* $U(x')$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x'))$*, both* $\mathsf{hsame}(x, x')$ *and* $x' \sim_U y$ *hold.*

Proof. First apply Theorem 33.222 to the composite image-classifier witness and the endpoint normalization witnesses. This gives visible-existence equivalence between u and u', together with the empty-versus-visible exclusion clauses at both normalized endpoints. For the left-to-right clause, fix x with $U(x)$ and $\mathsf{hsame}(u, \mathsf{E1}(x))$. The forward implication in the visible-existence equivalence supplies some y with $U(y)$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$. Apply Theorem 33.231 to the visible pair (x, y), using the same pair as the competing pair. This returns $x \sim_U y$ and the two non-absence facts $\neg\mathsf{hsame}(u, \mathsf{e})$ and $\neg\mathsf{hsame}(u', \mathsf{e})$. Now let y' be any competing target payload with $U(y')$ and $\mathsf{hsame}(u', \mathsf{E1}(y'))$. Apply the same visible-payload determinacy theorem to the first visible pair (x, y) and the competing pair (x, y'). The determinacy conclusion gives $\mathsf{hsame}(y, y')$, and its classifier component gives $x \sim_U y'$. These facts complete the left-to-right package. For the right-to-left clause, fix y with $U(y)$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$. The reverse implication in the visible-existence equivalence supplies some x with $U(x)$ and $\mathsf{hsame}(u, \mathsf{E1}(x))$. Apply the visible-payload determinacy theorem to the visible pair (x, y), again using the same pair as the competing pair. This returns $x \sim_U y$ and the two non-absence facts for u and u'. Finally let x' be any competing target payload with $U(x')$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$. Apply visible-payload determinacy to the first visible pair (x, y) and the competing pair (x', y). The determinacy conclusion gives $\mathsf{hsame}(x, x')$, and its classifier component gives $x' \sim_U y$. This completes the symmetric package. □

33.14 Composite image classifier normalized public branch certificate

Theorem 33.224 (Tagged option composite image classifier normalized public branch certificate)**.** *This theorem fits the per-object template as follows: the traditional target is a single public-scope certificate for endpoint-normalized absent/present coverage and present-branch ledger factorization; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, visible-payload readback, and target certificate transitivity; the ledger records synchronized branch occupancy together with the exhaustive public absent-or-present certificate; the core theorem is the displayed conjunction; exactness supplies the source and intermediate payloads behind a public visible branch; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget*. Let*

$S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, and $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, and $\mathsf{TaggedOptionSou}$). For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following two assertions hold.

1.

$$(\mathsf{hsame}(u, \mathsf{e}) \iff \mathsf{hsame}(u', \mathsf{e})) \wedge ((\exists x : \mathsf{BHist}.\ U(x) \wedge \mathsf{hsame}(u, \mathsf{E1}(x))) \iff (\exists y : \mathsf{BHist}.\ U(y) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))))$$

2. Either

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

or there exist $x, y, k, k', p, q, a, b : \mathsf{BHist}$ such that

(a) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;

(b) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;

(c) $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y))$;

(d) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;

(e) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;

(f) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;

(g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;

(h) $p \sim_T q$;

(i) $x \sim_U y$;

(j) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;

(k) *for all* $x', y' : \mathsf{BHist}$, *if* $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'$.

Proof. First apply Theorem 33.222 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives the synchronized absent occupancy, synchronized visible occupancy, and the two empty-implies-no-visible clauses. Next apply Theorem 33.228 to the same composite image-classifier witness and the same endpoint normalization witnesses. The theorem returns either the normalized absent public certificate or the visible public factorization with the displayed source, intermediate, target, classifier, non-absence, and uniqueness clauses. Package the occupancy result as the first component and the absent-or-present public certificate as the second component. □

33.15 Composite image classifier normalized constructor normal form

Theorem 33.225 (Tagged option composite image classifier normalized constructor normal form)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized constructor-facing public readback for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint transport, public certificate exposure, zero-tag exclusion, and single-valued visible readback; the ledger records the normalized public image endpoints, their target classifier witness, the absence of zero-tag endpoints, and the absent-or-present readback split; the core theorem is the displayed constructor normal form; exactness says a normalized public endpoint pair is either the absent pair or a single classified visible pair and never a zero-tagged pair; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* $\mathsf{TaggedOptionSou}$). *For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following clauses hold simultaneously.

1. *The normalized endpoints are public image endpoints and are classified by the target tagged option classifier:*

   $$\mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; u) \wedge \mathsf{TaggedOptionPayloadDescentImageCarrier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}$$

2. *Every zero-tag presentation is impossible at either normalized endpoint:*

   $$\forall t : \mathsf{BHist}.\ (\mathsf{hsame}(u, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(u', \mathsf{E0}(t)) \to \bot).$$

3. *The normalized public readback is absent-or-present and single-valued: either*

   $$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

   or there exist $x, y : \mathsf{BHist}$ *such that*

   $$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \wedge x \sim_U y \wedge \neg\mathsf{hsame}(u, \mathsf{e}) \wedge \neg\mathsf{hsame}(u', \mathsf{e})$$

   and

   $$\forall x', y' : \mathsf{BHist}.\ U(x') \to U(y') \to \mathsf{hsame}(u, \mathsf{E1}(x')) \to \mathsf{hsame}(u', \mathsf{E1}(y')) \to \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'.$$

Proof. First transport the composite image-classifier witness from (m, m') to (u, u'). Apply Theorem 33.204 with the endpoint normalization witnesses $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$. This gives the same composite image-classifier statement at the normalized endpoint pair.

Next apply Theorem 33.220 to the original composite classifier and the same endpoint normalization witnesses. Project the two normalized public image carriers, the target classifier witness, and the absent-or-present single-valued readback disjunction.

It remains to rule out zero-tag normalized endpoints. Fix $t : \mathsf{BHist}$ and apply Theorem 33.216 to the transported composite classifier at (u, u') with this tail t. The result gives the two implication clauses $\mathsf{hsame}(u, \mathsf{E0}(t)) \to \bot$ and $\mathsf{hsame}(u', \mathsf{E0}(t)) \to \bot$.

Package these endpoint contradictions for arbitrary t together with the carrier, classifier, and readback clauses already obtained from normalized public readback single-valuedness. □

Theorem 33.226 (Tagged option composite image classifier normalized visible payload exactness)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized visible payload exactness for the composite* OptionUp *image classifier; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, and visible-payload exactness; the ledger records normalized visible endpoints and the source payload pair; the core theorem is the displayed equivalence; exactness says visible normalized endpoints classify exactly when they come from a classified source pair; the standard bridge is* none/some *visible readback; and the status is* ProofTarget*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou
). For final endpoints $m, m' : \mathsf{BHist}$*, normalized endpoints* $u, u' : \mathsf{BHist}$*, and visible target payloads* $x, y : \mathsf{BHist}$*, the normalized visible endpoint criterion is exact:*

$$(\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m') \wedge \mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u') \wedge U(x) \wedge U(y) \wedge \mathsf{hsame}($$

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u') \wedge U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \wedge \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge U(\epsilon$$

Proof. For the forward direction, keep the endpoint normalization witnesses and the visible target facts. Apply Theorem 33.204 to the composite classifier at (m, m') with $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$. This gives the composite image-classifier witness at the normalized endpoint pair (u, u').

Now instantiate Theorem 33.211 with the composite descent certificate $\epsilon \circ \delta$, the target carrier U, and the preservation field $\rho_{\epsilon\circ\delta}$. Its forward direction, applied to the normalized classifier and the visible presentations $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$, supplies source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the two composite target carrier witnesses, and the two payload identifications. Combining these witnesses with the retained endpoint and visible facts gives the right-hand side.

For the reverse direction, unpack the right-hand side. Retain $\mathsf{hsame}(m, u)$, $\mathsf{hsame}(m', u')$, the target carrier witnesses $U(x)$ and $U(y)$, and the visible presentations at (u, u'), and choose source payload witnesses a, b from the existential clause.

Apply the reverse direction of Theorem 33.211 to the composite descent certificate. The retained visible facts together with the chosen source pair build $\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_\epsilon$

Finally transport this classifier back to (m, m'). Use symmetry of the two endpoint hsame witnesses to turn $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$ into $\mathsf{hsame}(u, m)$ and $\mathsf{hsame}(u', m')$, then apply Theorem 33.204. Together with the retained endpoint and visible facts, this is the left-hand side of the displayed equivalence. □

Theorem 33.227 (Tagged option composite image classifier normalized visible payload factorization)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* $\mathsf{OptionUp}$ *visible payload factorization; the BEDC source is* BHist *with composable* $\mathsf{DescentCertificate}$ *maps* δ *and* ϵ*; the carrier is* $\mathsf{TaggedOptionPayloadDescentImageCarrier}$*; the classifier is* $\mathsf{TaggedOptionPayloadDescentImageClassifier}$*; the stability obligations are* hsame *endpoint transport, map-relation composition factorization, and visible branch readback; the ledger records normalized final visible payloads together with intermediate endpoints and source payloads; the core theorem is the existence of* k, k', p, q, a, b *below; exactness says every normalized visible final payload pair factors through one intermediate tagged pair; the standard bridge is* some *payload functorial readback; and the status is* $\mathsf{ProofTarget}$*.*

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* $\mathsf{TaggedOptionSou}$ *). For final endpoints* $m, m' : \mathsf{BHist}$*, normalized endpoints* $u, u' : \mathsf{BHist}$*, and visible final payloads* $x, y : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

then there exist $k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
4. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
5. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
6. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$.

Proof. First transport the composite image-classifier witness from (m, m') to (u, u'). Apply Theorem 33.204 with the endpoint normalization witnesses $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$.

Next unfold the transported composite image classifier through Theorem 33.207. Equivalently, apply the underlying map-relation composition factorization to the two composite map-relation witnesses. This supplies intermediate endpoints k, k', the δ image-classifier factor at (k, k'), and the two ϵ map-relation witnesses from k to u and from k' to u'.

Use Theorem 33.138 on the two ϵ map-relation witnesses. The normalized visible presentations $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$ force visible intermediate presentations, giving payloads p, q with $T(p)$, $T(q)$, $\mathsf{hsame}(k, \mathsf{E1}(p))$, and $\mathsf{hsame}(k', \mathsf{E1}(q))$.

Apply Theorem 33.139 to the left ϵ map-relation witness, the intermediate presentation $\mathsf{hsame}(k, \mathsf{E1}(p))$, and the normalized final presentation $\mathsf{hsame}(u, \mathsf{E1}(x))$. This identifies x with $\epsilon.\mathsf{map}(p)$ and records $U(\epsilon.\mathsf{map}(p))$. Apply the same readback theorem to the right endpoint to identify y with $\epsilon.\mathsf{map}(q)$ and record $U(\epsilon.\mathsf{map}(q))$.

Now apply Theorem 33.164 to the δ image-classifier factor and the visible intermediate pair. It gives source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the transported target-carrier witnesses $T(\delta.\mathsf{map}(a))$ and $T(\delta.\mathsf{map}(b))$, and the payload identifications $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$.

Package k, k', p, q, and a, b with the retained factorization witnesses, the visible readback links, and the source payload data. These are exactly the six displayed clauses. □

Theorem 33.228 (Tagged option composite image classifier normalized public branch factorization)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized public branch factorization for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are normalized public readback, visible payload factorization, and classifier transport through* hsame*-compatible source carriers; the ledger records either the normalized absent pair or a classified factored visible public branch; the core theorem is the displayed exhaustive disjunction; exactness says every present normalized public branch carries both an intermediate classified payload pair and a final classified payload pair; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then either

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

or there exist $k, k', p, q, x, y, a, b : \mathsf{BHist}$ such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y))$;
4. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
5. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
6. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
7. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
8. $p \sim_T q$;
9. $x \sim_U y$;
10. $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
11. *for all* $x', y' : \mathsf{BHist}$, *if* $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'$.

Proof. Apply Theorem 33.220 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives the normalized absent branch or a present normalized visible pair x, y with non-absence and the universal competing-pair uniqueness clause.

In the absent branch, return the two empty endpoint presentations and the exclusion of every U-visible pair at (u, u').

In the present branch, apply Theorem 33.227 to the same composite classifier, the same endpoint normalization witnesses, and the selected visible presentations of u and u'. This supplies intermediate endpoints k, k', intermediate payloads p, q, and source payloads a, b with the six factorization clauses.

It remains to recover the two classifier consequences. Apply $\delta.\mathsf{respects}$ to $a \sim_S b$, then use compatibility for T to turn $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$ into classifier links. Symmetry and transitivity from cert_T compose these links into $p \sim_T q$.

Apply $\epsilon.\mathsf{respects}$ to $p \sim_T q$. Compatibility for U turns $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$ and $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$ into target classifier links, and symmetry and transitivity from cert_U compose them into $x \sim_U y$.

Package the factorization witnesses with $p \sim_T q$, $x \sim_U y$, the retained non-absence facts, and the competing-pair uniqueness clause from normalized public readback single-valuedness. □

Theorem 33.229 (Tagged option composite image classifier normalized source payload exactness)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ and assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. For final endpoints $m, m' : \mathsf{BHist}$, normalized endpoints $u, u' : \mathsf{BHist}$, and visible target payloads $x, y : \mathsf{BHist}$, assume

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u').$$

Then the normalized visible payload facts factor through classified source payloads:

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \iff$$

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)) \wedge \exists a, b : \mathsf{BHist}.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))) \wedge U(\epsilon.\mathsf{map}(\delta.$$

Proof. For the forward direction, transport the composite image-classifier witness from (m, m') to (u, u'). Apply Theorem 33.204 with the two endpoint normalization witnesses.

Now apply Theorem 33.211 to the transported classifier, using the composite descent certificate $\epsilon \circ \delta$ and the preservation field $\rho_{\epsilon\circ\delta}$. The visible presentations at (u, u') produce source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the two composite target carrier witnesses, and the two payload identifications. Conjoin this witness package with the four visible facts.

For the reverse direction, project the four visible facts from the right-hand side and discard the source-payload witness. □

33.16 Composite image classifier normalized visible target-payload classification

Theorem 33.230 (Tagged option composite image classifier normalized visible target-payload classification)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized visible target-payload classification for the composite* OptionUp *image classifier; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImag *the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, visible-payload exactness, descent respect, and target certificate transitivity; the ledger records normalized visible endpoints and their target payload pair; the core theorem is target-side classifier recovery; exactness supplies the source pair behind the visible payloads; the standard bridge is* none/some *visible readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. *Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* TaggedOptionSourceHsameCompatible$(S, \sim_S)$, TaggedOptionSourceHsameCompatible$(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$, *normalized endpoints* $u, u' : \mathsf{BHist}$, *and visible target payloads* $x, y : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

then $x \sim_U y$.

Proof. First apply the forward direction of Theorem 33.226 to the composite image-classifier witness, the two endpoint normalization witnesses, and the two visible target-payload presentations. This returns source witnesses a, b with $S(a)$, $S(b)$, and $a \sim_S b$, together with the target carrier witnesses for $\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and $\epsilon.\mathsf{map}(\delta.\mathsf{map}(b))$ and the payload readbacks $\mathsf{hsame}(x, \epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(y, \epsilon.\mathsf{map}(\delta.\mathsf{map}(b)))$.

Use $\delta.\mathsf{respects}$ on $a \sim_S b$ to obtain $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$. Then use $\epsilon.\mathsf{respects}$ to derive $\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)) \sim_U \epsilon.\mathsf{map}(\delta.\mathsf{map}(b))$.

Target compatibility for U turns the carrier facts for x and $\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and the first payload readback into $x \sim_U \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$. It also turns the carrier facts for y and $\epsilon.\mathsf{map}(\delta.\mathsf{map}(b))$ and the second payload readback into $y \sim_U \epsilon.\mathsf{map}(\delta.\mathsf{map}(b))$.

Apply symmetry from cert_U to the second target-compatibility relation, giving $\epsilon.\mathsf{map}(\delta.\mathsf{map}(b)) \sim_U y$. Transitivity from cert_U first composes $x \sim_U \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ with the composite descent relation, and then composes the result with $\epsilon.\mathsf{map}(\delta.\mathsf{map}(b)) \sim_U y$. The conclusion is $x \sim_U y$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized visible target payload classification`

Theorem 33.231 (Tagged option composite image classifier normalized visible payload determinacy)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *visible-payload determinacy; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, normalized target-payload classification,* E1 *payload determinacy, and empty-versus-present no-confusion; the ledger is the pair of normalized visible endpoints and two visible target payload readings; the core theorem is the displayed classification, determinacy, and non-absence conjunction; exactness says normalized visible readback is single-valued and target-classified; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints $m, m' : \mathsf{BHist}$, normalized endpoints $u, u' : \mathsf{BHist}$, and visible target payloads $x, y, x', y' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

and

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(u, \mathsf{E1}(x')) \wedge \mathsf{hsame}(u', \mathsf{E1}(y')),$$

then

$$x \sim_U y \wedge x' \sim_U y' \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge \neg\mathsf{hsame}(u, \mathsf{e}) \wedge \neg\mathsf{hsame}(u', \mathsf{e}).$$

Proof. First classify the two normalized visible target payload pairs. Apply Theorem 33.230 to the composite image-classifier witness, the endpoint normalization witnesses, and the presentation $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, and $\mathsf{hsame}(u', \mathsf{E1}(y))$. This gives $x \sim_U y$.

Apply the same theorem to the second visible presentation $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, and $\mathsf{hsame}(u', \mathsf{E1}(y'))$. This gives $x' \sim_U y'$.

Next compare the two readings of each normalized endpoint. Apply Theorem 33.122 at u to $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$, giving $\mathsf{hsame}(x, x')$. Apply the same theorem at u' to $\mathsf{hsame}(u', \mathsf{E1}(y))$ and $\mathsf{hsame}(u', \mathsf{E1}(y'))$, giving $\mathsf{hsame}(y, y')$.

It remains to rule out hidden absent tags at the normalized endpoints. If $\mathsf{hsame}(u, \mathsf{e})$ held, composing its symmetry with $\mathsf{hsame}(u, \mathsf{E1}(x))$ would give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(x))$, contradicting BEDC history no-confusion via *not_hsame_emp_e*1.

Likewise, if $\mathsf{hsame}(u', \mathsf{e})$ held, composing its symmetry with $\mathsf{hsame}(u', \mathsf{E1}(y))$ would give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(y))$, again contradicting BEDC history no-confusion via *not_hsame_emp_e*1.

Collect the two target-classification conclusions, the two payload-determinacy conclusions, and the two normalized non-absence conclusions in the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized visible payload determinacy`

Theorem 33.232 (Tagged option composite image classifier normalized visible payload clique)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *visible payload classifier clique; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier*

is TaggedOptionPayloadDescentImageClassifier*; the stability obligations are normalized visible determinacy, target* hsame *compatibility, and* NameCert *transitivity; the ledger is all competing visible endpoint readings; the core theorem is four target-classifier links plus endpoint determinacy and non-absence; exactness says every visible readback at these endpoints belongs to the same target classifier component; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. *Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$, *normalized endpoints* $u, u' : \mathsf{BHist}$, *and visible target payloads* $x, y, x', y' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

and

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(u, \mathsf{E1}(x')) \wedge \mathsf{hsame}(u', \mathsf{E1}(y')),$$

then

$$x \sim_U y \wedge x \sim_U y' \wedge x' \sim_U y \wedge x' \sim_U y' \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge \neg\mathsf{hsame}(u, \mathsf{e}) \wedge \neg\mathsf{hsame}(u', \mathsf{e}).$$

Proof. Apply Theorem 33.231 to the two visible presentations. This yields $x \sim_U y$, $x' \sim_U y'$, $\mathsf{hsame}(x, x')$, $\mathsf{hsame}(y, y')$, and the two normalized non-absence clauses.

The endpoint comparison at u' converts to target classification by the $\mathsf{TaggedOptionSourceHsameCompatible}(U, \sim$) assumption, giving $y \sim_U y'$. Transitivity from cert_U composes $x \sim_U y$ with $y \sim_U y'$, yielding $x \sim_U y'$.

For the opposite source reading, symmetry for hsame turns $\mathsf{hsame}(x, x')$ into $\mathsf{hsame}(x', x)$. Target hsame compatibility for U gives $x' \sim_U x$, and transitivity from cert_U composes this with $x \sim_U y$, yielding $x' \sim_U y$.

Collect the retained determinate link $x \sim_U y$, the two cross-links, the retained determinate link $x' \sim_U y'$, the two endpoint payload equalities, and the two normalized non-absence clauses in the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized visible payload clique`

Theorem 33.233 (Tagged option composite image classifier normalized visible source-pair coherence)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$ and assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. For final endpoints $m, m' : \mathsf{BHist}$, normalized endpoints $u, u' : \mathsf{BHist}$, and visible target payloads $x, y, x', y' : \mathsf{BHist}$, assume

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

and

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(u, \mathsf{E1}(x')) \wedge \mathsf{hsame}(u', \mathsf{E1}(y')).$$

Then there exist $a, b : \mathsf{BHist}$ such that

$$S(a) \wedge S(b) \wedge a \sim_S b$$

and

$$U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))) \wedge U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(b)))$$

and

$$\mathsf{hsame}(x, \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(y, \epsilon.\mathsf{map}(\delta.\mathsf{map}(b))) \wedge \mathsf{hsame}(x', \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(y', \epsilon.\mathsf{map}(\delta.\mathsf{map}(b))).$$

Proof. First apply Theorem 33.229 to the composite image-classifier witness, the endpoint normalizations, and the unprimed visible presentation. The forward direction supplies source payloads a, b with $S(a)$, $S(b)$, $a \sim_S b$, the two composite target carrier witnesses, and the readbacks $\mathsf{hsame}(x, \epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(y, \epsilon.\mathsf{map}(\delta.\mathsf{map}(b)))$.

Next compare the competing visible readings of the same normalized endpoints. Apply Theorem 33.122 at u to $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$, and apply the same theorem at u' to $\mathsf{hsame}(u', \mathsf{E1}(y))$ and $\mathsf{hsame}(u', \mathsf{E1}(y'))$. These give $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$.

Use symmetry and transitivity of hsame to compose the endpoint-comparison witnesses with the unprimed source readbacks. This yields $\mathsf{hsame}(x', \epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$ and $\mathsf{hsame}(y', \epsilon.\mathsf{map}(\delta.\mathsf{map}(b)))$.

Package the same source payloads a, b with their source classification, the two composite target carrier witnesses, and the four readback identifications. □

Theorem 33.234 (Tagged option composite image classifier normalized selected visible public pair canonicality)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *selected visible public-pair canonicality; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are composite descent, endpoint* hsame *transport, normalized target-payload classification, visible-payload determinacy, and empty-versus-present no-confusion; the ledger is one selected normalized visible public pair together with all one-sided competing readings; the core theorem is selected pair classification, same-side uniqueness, cross-side target classification, and non-absence; exactness says normalized visible readback is single-valued around the selected public pair; the standard bridge is* none/ some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$*, normalized endpoints* $u, u' : \mathsf{BHist}$*, and selected visible target payloads* $x, y : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

then

$$x \sim_U y$$

and

$$\forall x' : \mathsf{BHist}.\ U(x') \to \mathsf{hsame}(u, \mathsf{E1}(x')) \to \mathsf{hsame}(x, x') \wedge x' \sim_U y$$

and

$$\forall y' : \mathsf{BHist}.\ U(y') \to \mathsf{hsame}(u', \mathsf{E1}(y')) \to \mathsf{hsame}(y, y') \wedge x \sim_U y'$$

and

$$\neg\mathsf{hsame}(u, \mathsf{e}) \wedge \neg\mathsf{hsame}(u', \mathsf{e}).$$

Proof. Apply Theorem 33.230 to the composite image-classifier witness, the endpoint normalization witnesses, and the selected visible presentation $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, and $\mathsf{hsame}(u', \mathsf{E1}(y))$. This gives the selected target classification $x \sim_U y$.

Let x' be a competing left visible payload with $U(x')$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$. Apply Theorem 33.231 to the two complete visible pairs (x, y) and (x', y). The returned conjunction contains $\mathsf{hsame}(x, x')$ and $x' \sim_U y$, which are exactly the left-side canonicality and target-classification conclusions for x'.

Let y' be a competing right visible payload with $U(y')$ and $\mathsf{hsame}(u', \mathsf{E1}(y'))$. Apply the same determinacy theorem to the two complete visible pairs (x, y) and (x, y'). The returned conjunction contains $\mathsf{hsame}(y, y')$ and $x \sim_U y'$, which are exactly the right-side canonicality and target-classification conclusions for y'.

It remains to record that the selected public endpoints are not absent. If $\mathsf{hsame}(u, \mathsf{e})$ held, composing its symmetry with $\mathsf{hsame}(u, \mathsf{E1}(x))$ would give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(x))$, contradicting BEDC history no-confusion via *not_hsame_emp_e*1.

Likewise, if $\mathsf{hsame}(u', \mathsf{e})$ held, composing its symmetry with $\mathsf{hsame}(u', \mathsf{E1}(y))$ would give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(y))$, again contradicting BEDC history no-confusion via *not_hsame_emp_e*1.

Collect the selected target classification, the two universal one-sided canonicality clauses, and the two non-absence conclusions. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized selected visible public pair canonicality`

Theorem 33.235 (Tagged option composite image classifier normalized one-sided visible public factorization)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *one-sided visible public factorization; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint*

transport, branch synchronization, visible-payload factorization, and intermediate classifier transport; the ledger is one specified normalized visible endpoint together with the opposite public payload, intermediate tagged endpoints, and source payload witnesses; the core theorem is one-sided visible readback with factorization and uniqueness against all competing visible payloads on the other side; exactness says the public endpoint already read by a consumer exposes the concrete source and intermediate payloads behind the synchronized public branch; the standard bridge is none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. *Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following two one-sided visible public factorization clauses hold:

1. *For every* $x : \mathsf{BHist}$, *if* $U(x)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x))$, *then there exist* $y, k, k', p, q, a, b : \mathsf{BHist}$ *such that*
   - *(a)* $U(y)$, $\mathsf{hsame}(u', \mathsf{E1}(y))$, *and* $x \sim_U y$;
   - *(b)* $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   - *(c)* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   - *(d)* $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
   - *(e)* $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
   - *(f)* $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;

(g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$*;*

(h) $p \sim_T q$*;*

(i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$*;*

(j) for every $y' : \mathsf{BHist}$*, if* $U(y')$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$*, then* $\mathsf{hsame}(y, y')$ *and* $x \sim_U y'$*.*

2. *For every* $y : \mathsf{BHist}$*, if* $U(y)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y))$*, then there exist* $x, k, k', p, q, a, b : \mathsf{BHist}$ *such that*

   (a) $U(x)$*,* $\mathsf{hsame}(u, \mathsf{E1}(x))$*, and* $x \sim_U y$*;*

   (b) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*

   (c) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$*;*

   (d) $T(p)$*,* $T(q)$*,* $U(\epsilon.\mathsf{map}(p))$*, and* $U(\epsilon.\mathsf{map}(q))$*;*

   (e) $\mathsf{hsame}(k, \mathsf{E1}(p))$*,* $\mathsf{hsame}(k', \mathsf{E1}(q))$*,* $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$*, and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$*;*

   (f) $S(a)$*,* $S(b)$*,* $a \sim_S b$*,* $T(\delta.\mathsf{map}(a))$*, and* $T(\delta.\mathsf{map}(b))$*;*

   (g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$*;*

   (h) $p \sim_T q$*;*

   (i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$*;*

   (j) for every $x' : \mathsf{BHist}$*, if* $U(x')$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x'))$*, then* $\mathsf{hsame}(x, x')$ *and* $x' \sim_U y$*.*

Proof. Fix a left visible payload x with $U(x)$ and $\mathsf{hsame}(u, \mathsf{E1}(x))$. Apply Theorem 33.223 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. The left-to-right branch supplies a right payload y with $U(y)$, $\mathsf{hsame}(u', \mathsf{E1}(y))$, $x \sim_U y$, the two non-absence facts, and the universal competing-right-payload uniqueness clause.

Apply Theorem 33.227 to the same composite classifier, the same endpoint normalization witnesses, and the specified visible pair (x, y). This supplies intermediate endpoints k, k', intermediate payloads p, q, and source payloads a, b with the factorization clauses through δ and ϵ.

It remains to identify the intermediate payloads as classified. Apply $\delta.\mathsf{respects}$ to $a \sim_S b$. Compatibility for T turns $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ and $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$ into classifier links, and symmetry and transitivity from cert_T compose these links into $p \sim_T q$.

Package y, k, k', p, q, a, b with the visible target facts, the factorization data, $p \sim_T q$, the retained non-absence facts, and the competing-right-payload uniqueness clause. This proves the left one-sided factorization clause.

For the right visible payload y with $U(y)$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$, apply the same branch-synchronization theorem and use its right-to-left branch. It supplies a left payload x with $U(x)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, $x \sim_U y$, the two non-absence facts, and the universal competing-left-payload uniqueness clause.

Apply the normalized visible payload factorization theorem to the visible pair (x, y). The same factorization data k, k', p, q, a, b is obtained, because the specified right payload has now been paired with the synchronized left payload.

As in the left clause, $\delta.\mathsf{respects}$, compatibility for T, and symmetry and transitivity from cert_T compose the source classifier fact and the two readback links into $p \sim_T q$.

Package x, k, k', p, q, a, b with the visible target facts, the factorization data, $p \sim_T q$, the retained non-absence facts, and the competing-left-payload uniqueness clause. This proves the symmetric right one-sided factorization clause. □

Theorem 33.236 (Tagged option composite image classifier normalized selected visible public factorization)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, and $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, and $\mathsf{TaggedOptionSou}$). For final endpoints $m, m' : \mathsf{BHist}$, normalized endpoints $u, u' : \mathsf{BHist}$, and selected visible target payloads $x, y : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y)),$$

then there exist $k, k', p, q, a, b : \mathsf{BHist}$ such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
4. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
5. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
6. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
7. $p \sim_T q$ *and* $x \sim_U y$;
8. $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
9. *for every $x' : \mathsf{BHist}$, if $U(x')$ and $\mathsf{hsame}(u, \mathsf{E1}(x'))$, then $\mathsf{hsame}(x, x')$ and $x' \sim_U y$;*

10. for every $y' : \mathsf{BHist}$*, if* $U(y')$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$*, then* $\mathsf{hsame}(y, y')$ *and* $x \sim_U y'$*.*

Proof. Apply Theorem 33.234 to the composite image-classifier witness, the endpoint normalization witnesses, and the selected visible presentation. This gives $x \sim_U y$, the two selected same-side uniqueness clauses, and the two normalized non-absence facts.

Next apply the left clause of Theorem 33.235 to the selected left payload x. It supplies a synchronized right payload y_0 and witnesses k, k', p, q, a, b satisfying the displayed factorization clauses with y_0 in the right readback position, together with $p \sim_T q$, the non-absence facts, and uniqueness against all competing right payloads.

Use the one-sided right uniqueness clause at the selected payload y. Since $U(y)$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$ hold, it returns $\mathsf{hsame}(y_0, y)$ and $x \sim_U y$. Compose the symmetry of $\mathsf{hsame}(y_0, y)$ with the factorization readback $\mathsf{hsame}(y_0, \epsilon.\mathsf{map}(q))$ to obtain the selected readback $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$.

Keep the left readback $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$ and all remaining factorization clauses unchanged. Package the witnesses k, k', p, q, a, b with the transported right readback, the retained $p \sim_T q$, the selected classification $x \sim_U y$, the canonical non-absence facts, and the two selected uniqueness clauses from the canonicality theorem. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized selected visible public factorization`

Theorem 33.237 (Tagged option composite image classifier normalized visible public factorization is representative-independent)**.** *Let* $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* $\mathsf{TaggedOptionSou}$ *). For final endpoints* $m, m' : \mathsf{BHist}$*, normalized endpoints* $u, u' : \mathsf{BHist}$*, selected visible target payloads* $x, y : \mathsf{BHist}$*, and alternative visible target payloads* $x', y' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

and

$$U(x') \wedge U(y') \wedge \mathsf{hsame}(u, \mathsf{E1}(x')) \wedge \mathsf{hsame}(u', \mathsf{E1}(y')),$$

then there exist $k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
4. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
5. $\mathsf{hsame}(x', \epsilon.\mathsf{map}(p))$ *and* $\mathsf{hsame}(y', \epsilon.\mathsf{map}(q))$;
6. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
7. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
8. $p \sim_T q$, $x \sim_U y$, $x' \sim_U y'$, $x \sim_U y'$, *and* $x' \sim_U y$;
9. $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
10. *for every* $x_L : \mathsf{BHist}$, *if* $U(x_L)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *then* $\mathsf{hsame}(x, x_L)$ *and* $x_L \sim_U y$;
11. *for every* $y_R : \mathsf{BHist}$, *if* $U(y_R)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *then* $\mathsf{hsame}(y, y_R)$ *and* $x \sim_U y_R$.

Proof. Apply Theorem 33.236 to the composite image-classifier witness, the endpoint normalization witnesses, and the selected visible presentation (x, y). This gives witnesses k, k', p, q, a, b with the selected factorization clauses, the selected readbacks through $\epsilon.\mathsf{map}(p)$ and $\epsilon.\mathsf{map}(q)$, the classification $x \sim_U y$, the non-absence facts, and the two universal same-endpoint uniqueness clauses.

Apply the left uniqueness clause to x' and the right uniqueness clause to y'. The visible-presentation hypotheses for x' and y' give $\mathsf{hsame}(x, x')$, $x' \sim_U y$, $\mathsf{hsame}(y, y')$, and $x \sim_U y'$.

Apply Theorem 33.234 to the same composite witness and endpoint normalizations, but use the alternative visible presentation (x', y'). This gives the alternative classification $x' \sim_U y'$.

It remains to transport the selected public readbacks to the alternative representatives. Compose the symmetry of $\mathsf{hsame}(x, x')$ with $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$ to obtain $\mathsf{hsame}(x', \epsilon.\mathsf{map}(p))$, and compose the symmetry of $\mathsf{hsame}(y, y')$ with $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$ to obtain $\mathsf{hsame}(y', \epsilon.\mathsf{map}(q))$.

Package the same witnesses k, k', p, q, a, b with the original selected factorization clauses, the two transported readbacks, the selected and alternative target classifications, the two cross-classifications, and the retained universal uniqueness clauses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized visible public factorization representative independent`

33.17 Composite image classifier normalized public readback completion

Theorem 33.238 (Tagged option composite image classifier normalized public readback completion package)**.** *This theorem fits the per-object template as follows: the traditional target is*

endpoint-normalized public readback completion for functorial OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImag *the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are public branch factorization, one-sided visible public factorization, and visible payload determinacy; the ledger records either the absent normalized public branch or the selected present branch together with all one-sided visible completions; the core theorem is the displayed exhaustive completion package; exactness says any visible observation of one endpoint extends to a classified, factored public pair; the standard bridge is* none/some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. *Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* TaggedOptionSourceHsameCompatible$(S, \sim_S)$, TaggedOptionSourceHsameCompatible$(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then either

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}) \wedge \neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$$

or there exist selected visible payloads $x, y : \mathsf{BHist}$ *such that*

1. $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, $\mathsf{hsame}(u', \mathsf{E1}(y))$, *and* $x \sim_U y$;
2. $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
3. *for all* $x', y' : \mathsf{BHist}$, *if* $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'$;

4. *for every left visible payload* x_L : BHist, *if* $U(x_L)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *then there exist* $y_R, k, k', p, q, a, b : \mathsf{BHist}$ *such that*
   (a) $U(y_R)$, $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *and* $x_L \sim_U y_R$;
   (b) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   (c) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   (d) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
   (e) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x_L, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y_R, \epsilon.\mathsf{map}(q))$;
   (f) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
   (g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
   (h) $p \sim_T q$;
   (i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
   (j) *for every* $y' : \mathsf{BHist}$, *if* $U(y')$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(y_R, y')$ *and* $x_L \sim_U y'$.
5. *for every right visible payload* y_R : BHist, *if* $U(y_R)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *then there exist* $x_L, k, k', p, q, a, b : \mathsf{BHist}$ *such that*
   (a) $U(x_L)$, $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *and* $x_L \sim_U y_R$;
   (b) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   (c) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   (d) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
   (e) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x_L, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y_R, \epsilon.\mathsf{map}(q))$;
   (f) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
   (g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
   (h) $p \sim_T q$;
   (i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
   (j) *for every* $x' : \mathsf{BHist}$, *if* $U(x')$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *then* $\mathsf{hsame}(x_L, x')$ *and* $x' \sim_U y_R$.

Proof. Apply Theorem 33.228 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives either the normalized absent public branch or a present visible public branch with selected payloads x, y and universal visible-pair uniqueness.

In the absent branch, return exactly the two empty endpoint presentations and the exclusion of every visible U-payload pair at (u, u').

In the present branch, retain the selected visible payloads x, y, their target classification, the two non-absence facts, and the universal visible-pair uniqueness clause supplied by public branch factorization.

Apply Theorem 33.235 once to the same composite classifier, endpoint normalization witnesses, descent, preservation, stability, certificate, and compatibility data. The result is a pair of one-sided public factorization clauses.

For an arbitrary left visible payload x_L with $U(x_L)$ and $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, instantiate the left one-sided clause. It supplies a right visible payload y_R and witnesses k, k', p, q, a, b with the classified factorization package, the two public map-relation links, the intermediate and source payload data, the classifier link $p \sim_T q$, the non-absence facts, and the universal competing-right-payload clause.

For an arbitrary right visible payload y_R with $U(y_R)$ and $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, instantiate the right one-sided clause. It supplies a left visible payload x_L and witnesses k, k', p, q, a, b with the symmetric classified factorization package and the universal competing-left-payload clause.

Package the selected present branch data together with these two one-sided completion functions. This gives the displayed present public readback completion package. □

33.18 Composite image classifier normalized one-sided public completion

Theorem 33.239 (Tagged option composite image classifier normalized one-sided public completion)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized one-sided public completion for functorial* OptionUp *image classification; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier *the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are public readback completion, endpoint visibility, empty-versus-present no-confusion, and the* hsame *symmetry and transitivity spine; the ledger records total left and right visible-endpoint consumers with synchronized opposite endpoints; the core theorem is one-sided public completion from any visible normalized endpoint; exactness says the completed opposite endpoint carries the same target classifier and the full descent factorization package; the standard bridge is* some *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$, *if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the following one-sided public completion clauses hold:

1. *for every left visible payload* x_L : BHist, *if* $U(x_L)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *then there exist* $y_R, k, k', p, q, a, b : \mathsf{BHist}$ *such that*
   (a) $U(y_R)$, $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *and* $x_L \sim_U y_R$;
   (b) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   (c) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   (d) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
   (e) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x_L, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y_R, \epsilon.\mathsf{map}(q))$;
   (f) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
   (g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
   (h) $p \sim_T q$;
   (i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
   (j) *for every* y' : BHist, *if* $U(y')$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(y_R, y')$ *and* $x_L \sim_U y'$.
2. *for every right visible payload* y_R : BHist, *if* $U(y_R)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *then there exist* $x_L, k, k', p, q, a, b : \mathsf{BHist}$ *such that*
   (a) $U(x_L)$, $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *and* $x_L \sim_U y_R$;
   (b) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   (c) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   (d) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
   (e) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x_L, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y_R, \epsilon.\mathsf{map}(q))$;
   (f) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
   (g) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
   (h) $p \sim_T q$;
   (i) $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
   (j) *for every* x' : BHist, *if* $U(x')$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *then* $\mathsf{hsame}(x_L, x')$ *and* $x' \sim_U y_R$.

Proof. Apply Theorem 33.238 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives either the normalized absent public branch or a present visible branch carrying the two one-sided completion functions.

For the left clause, fix x_L with $U(x_L)$ and $\mathsf{hsame}(u, \mathsf{E1}(x_L))$. In the absent branch, the supplied $\mathsf{hsame}(u, \mathsf{e})$ composes with the visible endpoint witness, after symmetry if needed, to identify e with $\mathsf{E1}(x_L)$. This contradicts BEDC history no-confusion via *not_hsame_emp_e*1, so the absent branch cannot occur.

In the present branch, instantiate the left one-sided completion function at x_L. It supplies y_R, k, k', p, q, a, b with the target endpoint, the synchronized target classifier, the two-stage descent factorization, the public map-relation links, the source and intermediate payload witnesses, the non-absence facts, and the universal competing-right-payload clause.

For the right clause, fix y_R with $U(y_R)$ and $\mathsf{hsame}(u', \mathsf{E1}(y_R))$. In the absent branch, the supplied $\mathsf{hsame}(u', \mathsf{e})$ composes with the visible endpoint witness, after symmetry if needed, to identify e with $\mathsf{E1}(y_R)$. This contradicts BEDC history no-confusion via *not_hsame_emp_e*1, so the absent branch cannot occur.

In the present branch, instantiate the right one-sided completion function at y_R. It supplies x_L, k, k', p, q, a, b with the symmetric target endpoint, classifier, descent factorization, public map-relation links, non-absence facts, and universal competing-left-payload clause. These are exactly the two displayed one-sided public completion clauses. □

Theorem 33.240 (Tagged option composite image classifier normalized absent public factorization)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *absent public factorization; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* TaggedOptionPayloadDescentImageCarrier*; the classifier is* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint* hsame *transport, map-relation composition factorization, empty-branch preservation, and empty-versus-present no-confusion; the ledger is the normalized absent public pair together with its intermediate endpoints; the core theorem is absent public factorization through the second descent stage; exactness says the intermediate public pair is also absent and has no visible present presentation; the standard bridge is* none *normalized public readback; and the status is* ProofTarget*.*

Let S, T, U : BHist $\to$ Prop be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let cert_S : $\mathsf{NameCert}(S, \sim_S)$*,* cert_T : $\mathsf{NameCert}(T, \sim_T)$*, and* cert_U : $\mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou *). For final endpoints m, m' : BHist and normalized endpoints u, u' : BHist, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}),$$

then there exist k, k' : BHist such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$;
4. $\neg\exists p, q : \mathsf{BHist}.\ T(p) \wedge T(q) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k', \mathsf{E1}(q))$.

Proof. First transport the composite image-classifier witness from (m, m') to (u, u'). Apply Theorem 33.204 with the endpoint normalization witnesses $\mathsf{hsame}(m, u)$ and $\mathsf{hsame}(m', u')$.

Next apply Theorem 33.207 to the transported composite classifier. This supplies intermediate endpoints k, k', the δ image-classifier factor at (k, k'), and the two ϵ map-relation witnesses from k to u and from k' to u'.

Use Theorem 33.138 on the left ϵ map-relation witness. The normalized absent presentation $\mathsf{hsame}(u, \mathsf{e})$ forces $\mathsf{hsame}(k, \mathsf{e})$. Apply the same branch equivalence to the right ϵ map-relation witness and $\mathsf{hsame}(u', \mathsf{e})$ to obtain $\mathsf{hsame}(k', \mathsf{e})$.

It remains to rule out an intermediate visible present pair. If such p, q existed, the presentation $\mathsf{hsame}(k, \mathsf{E1}(p))$ would compose with $\mathsf{hsame}(k, \mathsf{e})$ to identify e with $\mathsf{E1}(p)$, contradicting BEDC history empty-versus-one no-confusion.

Package the witnesses k, k' with the retained δ image-classifier factor, the retained ϵ map-relation witnesses, the two intermediate absent presentations, and the excluded visible-pair clause. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized absent public factorization`

33.19 Tagged option map relation constructor normal form

Theorem 33.241 (Tagged option map relation constructor normal form)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-facing readback for the tagged option map relation; the BEDC source is* BHist *with one* DescentCertificate *map; the carrier is the visible tagged option endpoint shape; the classifier is the source and target history classifiers carried by the descent certificate; the stability obligation is branch preservation, payload readback, and zero-tag exclusion; the ledger records the absent branch, the present branch, and the absence of zero-tag endpoints; the core theorem is the displayed constructor normal form; exactness says a map-relation witness has exactly the empty branch or a transported one-payload branch; the standard bridge is* none/some *map readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories $h, k : \mathsf{BHist}$*, every witness*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

implies the following clauses simultaneously.

1. *Every zero-tag presentation is impossible at either endpoint:*

$$\forall t : \mathsf{BHist}.\ (\mathsf{hsame}(h, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(k, \mathsf{E0}(t)) \to \bot).$$

2. *The absent branch is visible on both sides:*

$$\mathsf{hsame}(h, \mathsf{e}) \iff \mathsf{hsame}(k, \mathsf{e}).$$

3. *The present branch is visible on both sides:*

$$(\exists a : \mathsf{BHist}.\ S(a) \wedge \mathsf{hsame}(h, \mathsf{E1}(a))) \iff (\exists b : \mathsf{BHist}.\ T(b) \wedge \mathsf{hsame}(k, \mathsf{E1}(b))).$$

4. *Visible source and target payloads read back to one transported payload:*

$$\forall u\, v : \mathsf{BHist}.\ S(u) \wedge \mathsf{hsame}(h, \mathsf{E1}(u)) \wedge T(v) \wedge \mathsf{hsame}(k, \mathsf{E1}(v)) \to \exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(u, a) \wedge \mathsf{hsam}$$

Proof. Split the map-relation witness into its absent and present branches. In the absent branch, the stored endpoint presentations are $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$; in the present branch, the stored presentations are $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ for the exposed payload a.

First prove zero-tag exclusion. Fix $t : \mathsf{BHist}$. In the absent branch, any presentation $\mathsf{hsame}(h, \mathsf{E0}(t))$ or $\mathsf{hsame}(k, \mathsf{E0}(t))$ composes with the corresponding empty presentation and contradicts generated-history no-confusion between e and $\mathsf{E0}(t)$. In the present branch, any such presentation composes with the corresponding one-tag presentation and contradicts generated-history no-confusion between $\mathsf{E1}$ and $\mathsf{E0}$.

Next obtain the visible branch clauses by applying Theorem 33.138 to the same map-relation witness. Its two projections are exactly the absent-branch equivalence and the present-branch equivalence in the statement.

Finally let $u, v : \mathsf{BHist}$ be visible source and target payloads with the displayed endpoint presentations. Apply Theorem 33.139 to the map-relation witness and to these four visible payload facts. The produced payload a carries $S(a)$, $T(\delta.\mathsf{map}(a))$, and the two readback witnesses required by the last clause. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel constructor normal form`

33.20 Tagged option map relation shared-source target normal form

Theorem 33.242 (Tagged option map relation shared-source target normal form)**.** *This theorem fits the per-object template as follows: the traditional target is tagged option map determinacy at public target endpoints; the BEDC source is* BHist *with one* DescentCertificate *map; the carrier is the visible tagged option endpoint shape; the classifier uses the source and target history classifiers carried by the descent certificate; the stability obligation is shared-source branch preservation together with payload transport; the ledger records absent and present public endpoints; the core theorem is the displayed two-witness normal form; exactness is absent-or-synchronized-present exhaustion; the standard bridge is* none/some *map readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source and target carriers, let $\sim_S$ and $\sim_T$ be their classifiers, let

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume also $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$. *For histories* h, k, k', *if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k')$$

then either

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e}),$$

or there exist $a, b : \mathsf{BHist}$ *such that*

$$S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(h, \mathsf{E1}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}($$

Proof. Split the first map-relation witness.

In the absent branch, the first witness presents the shared source endpoint as $\mathsf{hsame}(h, \mathsf{e})$ and the target endpoint as $\mathsf{hsame}(k, \mathsf{e})$. Split the second map-relation witness. Its absent branch contributes $\mathsf{hsame}(k', \mathsf{e})$, so the absent alternative follows. Its present branch would present the same source endpoint as $\mathsf{hsame}(h, \mathsf{E1}(b))$ for some payload b. Composing the two source presentations contradicts generated-history empty-versus-one no-confusion, so that branch is impossible.

In the present branch of the first witness, expose a payload a with $S(a)$, $T(\delta.\mathsf{map}(a))$, source presentation $\mathsf{hsame}(h, \mathsf{E1}(a))$, and target presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. Split the second map-relation witness. Its absent branch presents the shared source as $\mathsf{hsame}(h, \mathsf{e})$, which again contradicts empty-versus-one no-confusion after composition with $\mathsf{hsame}(h, \mathsf{E1}(a))$.

It remains to handle the present/present branch. The second witness exposes a payload b with $S(b)$, $T(\delta.\mathsf{map}(b))$, source presentation $\mathsf{hsame}(h, \mathsf{E1}(b))$, and target presentation $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(b)))$.

Invert the two one-tag presentations of the shared source endpoint to obtain $\mathsf{hsame}(a, b)$. Source hsame compatibility converts $S(a)$, $S(b)$, and $\mathsf{hsame}(a, b)$ into $a \sim_S b$. Applying $\delta.\mathsf{respects}$ gives $\delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b)$.

The two target carrier witnesses, the two target endpoint presentations, and this transported payload relation assemble the present-present branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$. Package a and b with the retained source facts, target facts, endpoint presentations, source classification, and target classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel shared source target normal form`

33.21 Tagged option map relation shared-target source reflection normal form

Theorem 33.243 (Tagged option map relation shared-target source reflection normal form)**.** *This theorem fits the per-object template as follows: the traditional target is faithful source readback from a shared visible target endpoint; the BEDC source is* BHist *with one* DescentCertificate *map; the carrier is the visible tagged option endpoint shape; the classifier uses the source history classifier reflected from target payload equality; the stability obligation is branch preservation, target one-constructor inversion, and source reflection; the ledger records absent endpoints and reflected*

present payloads; the core theorem is the displayed two-witness normal form; exactness is absent-or-reflected-present exhaustion; the standard bridge is none/some *map readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume also reflected source readback from visible target equality,

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For histories h, h', k*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$$

then either

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(h', \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}),$$

or there exist $a, b : \mathsf{BHist}$ *such that*

$$S(a) \wedge S(b) \wedge a \sim_S b \wedge T(\delta.\mathsf{map}(a)) \wedge T(\delta.\mathsf{map}(b)) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(h', \mathsf{E1}(b)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}($$

Proof. Split the first map-relation witness.

In the absent branch, the first witness presents the shared target endpoint as $\mathsf{hsame}(k, \mathsf{e})$ and the source endpoint as $\mathsf{hsame}(h, \mathsf{e})$. Split the second map-relation witness. Its absent branch contributes $\mathsf{hsame}(h', \mathsf{e})$, so the absent alternative follows. Its present branch would present the same target endpoint as $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$ for some source payload b, contradicting generated-history empty-versus-one no-confusion.

In the present branch of the first witness, expose a payload a with $S(a)$, $T(\delta.\mathsf{map}(a))$, source presentation $\mathsf{hsame}(h, \mathsf{E1}(a))$, and target presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. Split the second map-relation witness. Its absent branch presents the same target as $\mathsf{hsame}(k, \mathsf{e})$, again contradicting empty-versus-one no-confusion after composition with the stored one-tag target presentation.

It remains to handle the present/present branch. The second witness exposes a payload b with $S(b)$, $T(\delta.\mathsf{map}(b))$, source presentation $\mathsf{hsame}(h', \mathsf{E1}(b))$, and target presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(b)))$.

Invert the two one-tag presentations of the shared target endpoint to obtain $\mathsf{hsame}(\delta.\mathsf{map}(a), \delta.\mathsf{map}(b))$. The reflected source readback hypothesis converts this equality, together with $S(a)$ and $S(b)$, into $a \sim_S b$.

The two source carrier witnesses, the two source endpoint presentations, and this reflected payload relation assemble the present-present branch of $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$. Package a and b with the retained source facts, target facts, endpoint presentations, reflected source classification, and source classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel shared target source reflection normal form`

Theorem 33.244 (Tagged option map relation L-corner classifier coherence)**.** *This theorem fits the per-object template as follows: the traditional target is two-axis option-map endpoint determinacy; the BEDC source is* BHist *with one* DescentCertificate *map; the carrier is the tagged option endpoint shape; the classifiers are the source and target history classifiers; the stability obligations are shared-source target normal form and shared-target source reflection normal form; the ledger records the three map-relation legs around an L-corner; the core theorem is simultaneous source and target endpoint classification; exactness is absent-or-present exhaustion on both axes; the standard bridge is* none/some *map readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$ *and reflected source readback from visible target equality,*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For histories h, h', k, k'*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k'),$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h') \wedge \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. First compare the two legs with shared source endpoint h. Apply Theorem 33.242 to the two map-relation witnesses $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ and $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k')$. The result gives either simultaneous empty presentations of h, k, k' or a present-payload package ending in $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

Next compare the two legs with shared target endpoint k. Apply Theorem 33.243 to the map-relation witnesses $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ and $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$. This gives either simultaneous empty presentations of h, h', k or a present-payload package ending in $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$.

In an absent branch of the shared-source comparison, the empty presentations of k and k' assemble the absent branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$. In its present branch, retain the target classifier supplied by the normal form.

In an absent branch of the shared-target comparison, the empty presentations of h and h' assemble the absent branch of $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$. In its present branch, retain the source classifier supplied by the reflected normal form.

Combining the source classifier obtained from the shared-target comparison with the target classifier obtained from the shared-source comparison yields the displayed conjunction. □

Theorem 33.245 (Tagged option map relation exclusive visible readback is single-valued)**.** *This theorem fits the per-object template as follows: the traditional target is constructor-facing option map readback; the BEDC source is* BHist *with one* DescentCertificate*; the carrier is the visible*

tagged option endpoint shape; the classifiers are the source and target classifiers transported by the descent certificate; the stability obligations are branch preservation, no-confusion, and one-constructor inversion; the ledger separates the absent branch from the transported present branch; the core theorem is exclusive visible readback with uniqueness; exactness is a disjoint absent-or-present map-relation package; the standard bridge is none/some *map readback; and the status is* ProofTarget.

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

For histories $h, k : \mathsf{BHist}$*, every witness*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$$

implies exactly one of the following visible readback branches.

1. *The absent branch holds on both endpoints:*

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge (\forall a' : \mathsf{BHist}.\ S(a') \to \mathsf{hsame}(h, \mathsf{E1}(a')) \to \bot) \wedge (\forall b : \mathsf{BHist}.\ T(b) \to \mathsf{hsame}(k, \mathsf{E1}(b)) \to \bot$$

2. *The present branch has one visible source payload and its transported target payload:*

$$\exists a : \mathsf{BHist}.\ S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge (\mathsf{hsame}(h, \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(k, \mathsf{e}) \to \bot)$$

The two displayed alternatives are disjoint, so the map-relation witness packages the absent case and the single-valued visible present readback without any additional endpoint choice.

Proof. Case-analyze the $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ witness.

In the absent branch, the constructor supplies $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. If a source presentation $\mathsf{hsame}(h, \mathsf{E1}(a'))$ were also visible, generated-history no-confusion between e and $\mathsf{E1}(a')$ gives $\bot$; the same argument at the target endpoint rules out every visible $\mathsf{E1}(b)$ presentation. These four facts assemble the absent alternative.

In the present branch, retain the constructor payload a. Payload preservation gives $T(\delta.\mathsf{map}(a))$, and the constructor gives the endpoint presentations $\mathsf{hsame}(h, \mathsf{E1}(a))$ and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$.

It remains to show non-absence and uniqueness. A competing $\mathsf{hsame}(h, \mathsf{e})$ contradicts $\mathsf{hsame}(h, \mathsf{E1}(a))$ by one-versus-empty no-confusion, and the target endpoint is identical with $\delta.\mathsf{map}(a)$ in place of a.

For any competing source presentation $\mathsf{hsame}(h, \mathsf{E1}(a'))$, one-constructor inversion for $\mathsf{E1}$ yields $\mathsf{hsame}(a, a')$. For any competing target presentation $\mathsf{hsame}(k, \mathsf{E1}(b))$, the same inversion applied to the stored presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$ yields $\mathsf{hsame}(b, \delta.\mathsf{map}(a))$. These clauses assemble the present alternative, and empty-versus-one no-confusion also proves that the two displayed alternatives cannot overlap. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel exclusive visible readback single valued`

Theorem 33.246 (Tagged option composite factorization has a canonical intermediate endpoint)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for both descents:

$$\eta_\delta : \forall x\, y.\ S(x) \to S(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\delta.\mathsf{map}(x), \delta.\mathsf{map}(y))$$

and

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

For histories $h, k, k', m : \mathsf{BHist}$, suppose the same composed endpoint pair factors through both k and k':

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k) \wedge \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k') \wedge \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m).$$

Then either all four visible endpoints are absent,

$$\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{e}),$$

or there is one source payload $a : \mathsf{BHist}$ such that

$$S(a) \wedge T(\delta.\mathsf{map}(a)) \wedge U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}$$

Proof. Split the first δ factor through k.

In the absent branch, h and k are visibly e. The second δ factor through k' cannot be present, because it would present the same source endpoint h as both e and $\mathsf{E1}(a)$. Hence the second δ factor is also absent, giving $\mathsf{hsame}(k', \mathsf{e})$. The first ϵ factor cannot be present for the same no-confusion reason at k, so it is absent and gives $\mathsf{hsame}(m, \mathsf{e})$. These four presentations assemble the absent alternative.

In the present branch, the first δ factor exposes a source payload a with $S(a)$, $T(\delta.\mathsf{map}(a))$, $\mathsf{hsame}(h, \mathsf{E1}(a))$, and $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$.

Split the second δ factor. Its absent branch would present h as e, contradicting the stored one-constructor presentation of h. In its present branch it exposes a payload a' with $S(a')$ and $\mathsf{hsame}(h, \mathsf{E1}(a'))$. One-constructor inversion at h yields $\mathsf{hsame}(a, a')$, and η_δ transports this to $\mathsf{hsame}(\delta.\mathsf{map}(a), \delta.\mathsf{map}(a'))$. Rewriting the intermediate endpoint presentation for k' gives $\mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(a)))$.

Now split the first ϵ factor. Its absent branch would present k as e, contradicting the stored presentation $\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a)))$. In the present branch it exposes an intermediate payload

b with $T(b)$ and final endpoint $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(b)))$. One-constructor inversion at k gives $\mathsf{hsame}(\delta.\mathsf{map}(a), b)$, and η_ϵ transports the map endpoint to $\mathsf{hsame}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)), \epsilon.\mathsf{map}(b))$.

By one-constructor congruence and transitivity of hsame, the final endpoint presentation rewrites to $\mathsf{hsame}(m, \mathsf{E1}(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a))))$. Payload preservation supplies $U(\epsilon.\mathsf{map}(\delta.\mathsf{map}(a)))$, and the retained source, intermediate, and endpoint fields package the present alternative. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel composite factorization canonical intermediate`

33.22 Tagged option public endpoint visibility

Theorem 33.247 (Tagged option classifier endpoints expose visible certificate fields)**.** *This theorem fits the ten-field template as follows: the traditional target is public endpoint visibility for constructor-tagged* $\mathsf{Option}^{\uparrow}$*; the BEDC source is* BHist*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is endpoint carrier exposure plus visible-field projection; the ledger is visible absent/present tag; the core theorem is the displayed endpoint field package; exactness says classified endpoints expose public pattern and ledger witnesses; the standard bridge is* none/some *endpoint visibility; and the status is* ProofTarget*.*

For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, and histories* h, k*, if* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k)$ *holds, then*

$$\mathsf{TaggedOptionHistoryPattern}_S(h) \wedge \mathsf{TaggedOptionHistoryLedger}_S(h) \wedge \mathsf{TaggedOptionHistoryPattern}_S(k) \wedge \mathsf{TaggedOptionH}$$

Proof. Apply Theorem 33.21 to the classifier witness. This gives the endpoint carrier witnesses $\mathsf{TaggedOptionHistoryCarrier}_S(h)$ and $\mathsf{TaggedOptionHistoryCarrier}_S(k)$.

Apply Theorem 33.31 to the carrier witness for h and project the pattern and ledger components. Apply the same theorem to the carrier witness for k and project the same two visible certificate fields. The four projected witnesses assemble into the displayed conjunction. □

↪ variant `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier endpoint semantic fields`

33.23 Tagged option present-chain endpoint classification

Theorem 33.248 (Tagged option present-chain endpoint classification)**.** *This theorem fits the ten-field template as follows: the traditional target is present-chain optional payload classification; the BEDC source is* BHist *with a source* NameCert*; the carrier is* TaggedOptionHistoryCarrier*; the classifier is* TaggedOptionHistoryClassifier*; the stability obligation is transitive payload descent through a shared present endpoint; the ledger is visible present tags; the core theorem is endpoint payload classification; exactness is the visible endpoint readback; the standard bridge is* some*-chain classification; and the status is* ProofTarget*.*

For every BEDC history source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$*, source certificate* $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$*, histories* h, k, r*, and visible payloads* a, c*, assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim$ *). If*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, k), \quad \mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(k, r), \quad \mathsf{hsame}(h, \mathsf{E1}(a)), \quad \mathsf{hsam}$$

then there are endpoint payloads $a_0, c_0 : \mathsf{BHist}$ *such that*

$$S(a_0) \wedge S(c_0) \wedge \mathsf{hsame}(a, a_0) \wedge \mathsf{hsame}(c_0, c) \wedge a_0 \sim_S c_0.$$

Proof. Apply Theorem 33.116 to the two tagged classifier witnesses and the displayed endpoint presentations. This gives payload histories a_0, b, b', c_0 with the four source witnesses, the visible readbacks $\mathsf{hsame}(a, a_0)$, $\mathsf{hsame}(b, b')$, and $\mathsf{hsame}(c_0, c)$, and the two payload classifications $a_0 \sim_S b$ and $b' \sim_S c_0$.

Use $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$ on $S(b)$, $S(b')$, and $\mathsf{hsame}(b, b')$ to obtain the middle payload classification $b \sim_S b'$.

Apply the transitivity field cert.equiv_trans first to $a_0 \sim_S b$ and $b \sim_S b'$, and then to the result and $b' \sim_S c_0$. This derives $a_0 \sim_S c_0$. Package a_0 and c_0 with their source witnesses and endpoint readbacks. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier present chain endpoint classification`

33.24 Composite image classifier normalized factored public completion

Theorem 33.249 (Tagged option composite image classifier normalized factored public readback completion)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, and $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, and $\mathsf{TaggedOptionSou}$). For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then either

1. $\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e})$;

2. $\neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$;
3. *there exist* $k, k' : \mathsf{BHist}$ *such that*
   (a) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
   (b) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
   (c) $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$;
   (d) $\neg\exists p, q : \mathsf{BHist}.\ T(p) \wedge T(q) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k', \mathsf{E1}(q))$.

or there exist $x, y, k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, $\mathsf{hsame}(u', \mathsf{E1}(y))$, *and* $x \sim_U y$;
2. $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$;
3. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
4. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
5. $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;
6. $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;
7. $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;
8. $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;
9. $p \sim_T q$;
10. *for all* $x', y' : \mathsf{BHist}$, *if* $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'$;
11. *for every* $x_L : \mathsf{BHist}$, *if* $U(x_L)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *then* $\mathsf{hsame}(x, x_L)$ *and* $x_L \sim_U y$;
12. *for every* $y_R : \mathsf{BHist}$, *if* $U(y_R)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *then* $\mathsf{hsame}(y, y_R)$ *and* $x \sim_U y_R$.

Proof. Apply Theorem 33.238 to the composite image-classifier witness, the endpoint normalization witnesses, and the displayed descent, preservation, stability, certificate, and compatibility data. This gives an exhaustive absent-or-present public readback completion at (u, u').

In the absent branch, retain the two normalized empty presentations and the exclusion of every visible U-payload pair at (u, u'). Apply Theorem 33.240 to the same composite witness, endpoint normalizations, displayed data, and absent endpoint presentations. This supplies intermediate endpoints k, k' with the δ classifier factor, the two ϵ public map-relation links, the intermediate absent presentations, and the excluded intermediate visible pair. Package these data as the absent factored branch.

In the present branch, retain the selected payloads x, y, their visible target presentations, the target classification $x \sim_U y$, the two non-absence facts, and the universal competing visible-pair uniqueness clause. Apply Theorem 33.236 to the same composite witness, endpoint normalizations, displayed data, and selected visible presentation. This supplies k, k', p, q, a, b with the complete selected visible public factorization and the one-sided competing-payload uniqueness clauses. Package those witnesses together with the retained completion uniqueness clause. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized one sided visible public factorization`

Theorem 33.250 (Tagged option composite image classifier normalized zero-endpoint exclusion)**.** *This theorem fits the per-object template as follows: the traditional target is malformed endpoint rejection for functorial* OptionUp *public readback; the BEDC source is* BHist *with two composable descent maps; the carrier and image classifier are* TaggedOptionPayloadDescentImageClassifier *over the displayed public endpoints; the stability obligations are* hsame *transport plus branch exactness; the ledger exposes the visible absent/present tag together with zero-tag rejection; the core theorem is zero-endpoint exclusion; exactness says the normalized public endpoints are only absent or present; the standard bridge is* none/some*; and the status is* ProofTarget*. Let* $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$*,* $\sim_T$*, and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ *for the descent certificate whose map field is* $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ *and whose respects field sends* $r : a \sim_S b$ *to* $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$*. Define*

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$*,* $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$*, and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$*. Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and* TaggedOptionSou*). For final endpoints* $m, m' : \mathsf{BHist}$ *and normalized endpoints* $u, u' : \mathsf{BHist}$*, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then for every $t : \mathsf{BHist}$*,*

$$(\mathsf{hsame}(u, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(u', \mathsf{E0}(t)) \to \bot).$$

Proof. First transport the composite classifier from (m, m') to (u, u'). Apply Theorem 33.204 to the composite witness and the two normalization witnesses.

Next apply Theorem 33.162 to the transported classifier witness. This leaves the two possible public branches at the normalized endpoints.

In the absent branch, the exactness data gives $\mathsf{hsame}(u, \mathsf{e})$ and $\mathsf{hsame}(u', \mathsf{e})$. Any presentation $\mathsf{hsame}(u, \mathsf{E0}(t))$ or $\mathsf{hsame}(u', \mathsf{E0}(t))$ composes with the corresponding empty presentation and contradicts E0-versus-empty generated-history no-confusion.

In the present branch, exactness exposes payloads $x, y : \mathsf{BHist}$ with $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$. Any zero-tag presentation at either endpoint composes with the corresponding one-tag presentation and contradicts zero-versus-one generated-history no-confusion.

Since the branch analysis covers every transported normalized classifier witness, both normalized endpoints reject every malformed zero-tag tail. □

Lean4 ✓ BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized zero endpoint exclusion

Theorem 33.251 (Tagged option composite image classifier normalized public endpoint shape certificate)**.** *Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source, intermediate, and target carriers, let $\sim_S$, $\sim_T$, and $\sim_U$ be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, and $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, and $\mathsf{TaggedOptionSou}$). For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u'),$$

then the normalized public endpoint shape certificate consists of the following two fields.

1. *Either*
   - *(a)* $\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e})$*;*
   - *(b)* $\neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y))$*;*
   - *(c) there exist $k, k' : \mathsf{BHist}$ such that*
     - *i.* $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*
     - *ii.* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$*;*
     - *iii.* $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$*;*
     - *iv.* $\neg\exists p, q : \mathsf{BHist}.\ T(p) \wedge T(q) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k', \mathsf{E1}(q))$.

   or there exist $x, y, k, k', p, q, a, b : \mathsf{BHist}$ such that
   - *(a)* $U(x)$, $U(y)$, $\mathsf{hsame}(u, \mathsf{E1}(x))$, $\mathsf{hsame}(u', \mathsf{E1}(y))$, *and* $x \sim_U y$*;*
   - *(b)* $\neg\mathsf{hsame}(u, \mathsf{e})$ *and* $\neg\mathsf{hsame}(u', \mathsf{e})$*;*

(c) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;

(d) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;

(e) $T(p)$, $T(q)$, $U(\epsilon.\mathsf{map}(p))$, *and* $U(\epsilon.\mathsf{map}(q))$;

(f) $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, $\mathsf{hsame}(x, \epsilon.\mathsf{map}(p))$, *and* $\mathsf{hsame}(y, \epsilon.\mathsf{map}(q))$;

(g) $S(a)$, $S(b)$, $a \sim_S b$, $T(\delta.\mathsf{map}(a))$, *and* $T(\delta.\mathsf{map}(b))$;

(h) $\mathsf{hsame}(p, \delta.\mathsf{map}(a))$ *and* $\mathsf{hsame}(q, \delta.\mathsf{map}(b))$;

(i) $p \sim_T q$;

(j) *for all* $x', y' : \mathsf{BHist}$, *if* $U(x')$, $U(y')$, $\mathsf{hsame}(u, \mathsf{E1}(x'))$, *and* $\mathsf{hsame}(u', \mathsf{E1}(y'))$, *then* $\mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \wedge x' \sim_U y'$;

(k) *for every* $x_L : \mathsf{BHist}$, *if* $U(x_L)$ *and* $\mathsf{hsame}(u, \mathsf{E1}(x_L))$, *then* $\mathsf{hsame}(x, x_L)$ *and* $x_L \sim_U y$;

(l) *for every* $y_R : \mathsf{BHist}$, *if* $U(y_R)$ *and* $\mathsf{hsame}(u', \mathsf{E1}(y_R))$, *then* $\mathsf{hsame}(y, y_R)$ *and* $x \sim_U y_R$.

2. *For every* $t : \mathsf{BHist}$,

$$(\mathsf{hsame}(u, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(u', \mathsf{E0}(t)) \to \bot).$$

Proof. First apply Theorem 33.249 to the displayed composite witness, endpoint normalization witnesses, descent data, preservation fields, stability field, certificates, and compatibility fields. This gives the factored absent-or-present public readback completion at (u, u').

For the malformed zero-tag field, transport the composite classifier from (m, m') to (u, u'). Apply Theorem 33.204 to the composite witness and the two normalization witnesses.

Apply Theorem 33.162 to the transported classifier witness. In the absent branch, the exactness data gives $\mathsf{hsame}(u, \mathsf{e})$ and $\mathsf{hsame}(u', \mathsf{e})$; any presentation $\mathsf{hsame}(u, \mathsf{E0}(t))$ or $\mathsf{hsame}(u', \mathsf{E0}(t))$ contradicts $\mathsf{E0}$-versus-empty generated-history no-confusion.

In the present branch, exactness exposes payloads $x, y : \mathsf{BHist}$ with $\mathsf{hsame}(u, \mathsf{E1}(x))$ and $\mathsf{hsame}(u', \mathsf{E1}(y))$. Any zero-tag presentation at either endpoint contradicts zero-versus-one generated-history no-confusion. Pair the zero-tag exclusion with the public readback completion field. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized public endpoint shape certificate`

Theorem 33.252 (Tagged option composite image classifier normalized absent intermediate constructor exclusion)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *absent public factorization with intermediate constructor exclusion; the BEDC source is* BHist *with two composable descent maps; the carrier and image classifier are* TaggedOptionPayloadDescentImageCarrier *and* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint* hsame *transport, map-relation composition factorization, empty-branch preservation, and empty-versus-nonempty constructor no-confusion; the ledger is the normalized absent public pair together with constructor-normal intermediate endpoints; the core theorem is absent intermediate constructor exclusion; exactness says each intermediate endpoint rejects every nonempty generated-history presentation; the standard bridge is* none *normalized public readback; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for the second descent:

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Write $\epsilon \circ \delta$ for the descent certificate whose map field is $a \mapsto \epsilon.\mathsf{map}(\delta.\mathsf{map}(a))$ and whose respects field sends $r : a \sim_S b$ to $\epsilon.\mathsf{respects}(\delta.\mathsf{respects}(r))$. Define

$$\rho_{\epsilon\circ\delta}(a, Sa) = \rho_\epsilon(\delta.\mathsf{map}(a), \rho_\delta(a, Sa)).$$

Let $\mathsf{cert}_S : \mathsf{NameCert}(S, \sim_S)$, $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$, *and* $\mathsf{cert}_U : \mathsf{NameCert}(U, \sim_U)$. *Assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$, $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$, *and* TaggedOptionSou). *For final endpoints $m, m' : \mathsf{BHist}$ and normalized endpoints $u, u' : \mathsf{BHist}$, if*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_{\epsilon\circ\delta}(S, U, \rho_{\epsilon\circ\delta}; m, m')$$

and

$$\mathsf{hsame}(m, u) \wedge \mathsf{hsame}(m', u')$$

and

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e}),$$

then there exist $k, k' : \mathsf{BHist}$ such that

1. $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$;
2. $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$;
3. $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$;
4. *for every $t : \mathsf{BHist}$,* $(\mathsf{hsame}(k, \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(k, \mathsf{E1}(t)) \to \bot)$;
5. *for every $t : \mathsf{BHist}$,* $(\mathsf{hsame}(k', \mathsf{E0}(t)) \to \bot) \wedge (\mathsf{hsame}(k', \mathsf{E1}(t)) \to \bot)$.

Proof. First apply Theorem 33.240 to the composite image-classifier witness, the endpoint normalizations, the displayed descent, preservation, stability, certificate, compatibility data, and the two absent normalized presentations. This supplies intermediate endpoints k, k', the δ image-classifier factor, the two ϵ map-relation witnesses, the intermediate absent presentations $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$, and the excluded simultaneous visible intermediate pair.

For the left endpoint, fix $t : \mathsf{BHist}$. A presentation $\mathsf{hsame}(k, \mathsf{E0}(t))$ composes with $\mathsf{hsame}(k, \mathsf{e})$ to identify e with $\mathsf{E0}(t)$, contradicting empty-versus-zero generated-history no-confusion. A presentation $\mathsf{hsame}(k, \mathsf{E1}(t))$ composes with the same empty presentation to identify e with $\mathsf{E1}(t)$, contradicting empty-versus-one generated-history no-confusion.

For the right endpoint, fix $t : \mathsf{BHist}$. A presentation $\mathsf{hsame}(k', \mathsf{E0}(t))$ composes with $\mathsf{hsame}(k', \mathsf{e})$ to identify e with $\mathsf{E0}(t)$, contradicting empty-versus-zero generated-history no-confusion. A presentation $\mathsf{hsame}(k', \mathsf{E1}(t))$ composes with the same empty presentation to identify e with $\mathsf{E1}(t)$, contradicting empty-versus-one generated-history no-confusion.

Package the same k, k' with the retained factorization data, the retained intermediate empty presentations, and the two constructor-exclusion functions. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized absent intermediate constructor exclusion`

Theorem 33.253 (Tagged option composite image classifier normalized present intermediate constructor exclusion)**.** *This theorem fits the per-object template as follows: the traditional target is endpoint-normalized functorial* OptionUp *present public factorization with intermediate constructor exclusion; the BEDC source is* BHist *with two composable descent maps; the carrier and image classifier are* TaggedOptionPayloadDescentImageCarrier *and* TaggedOptionPayloadDescentImageClassifier*; the stability obligations are endpoint* hsame *transport, selected visible public factorization, present-branch readback, and empty/zero-versus-one constructor no-confusion; the ledger is the normalized visible public pair together with constructor-normal intermediate endpoints; the core theorem is present intermediate constructor exclusion; exactness says each selected intermediate endpoint rejects empty and zero-tag generated-history presentations; the standard bridge is* some *normalized public readback; and the status is* ProofTarget.

Let the carriers S, T, U, classifiers $\sim_S, \sim_T, \sim_U$, descents δ, ϵ, preservation fields $\rho_\delta, \rho_\epsilon$, endpoint-stability field η_ϵ, certificates $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$, compatibility fields, endpoints m, m', u, u', and selected visible payloads x, y satisfy exactly the hypotheses of Theorem 33.236. Then there exist $k, k', p, q, a, b : \mathsf{BHist}$ such that

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for k, k', p, q, a, b;*
2. *for every $t : \mathsf{BHist}$, $(\mathsf{hsame}(k, \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(k, \mathsf{E0}(t)) \to \bot)$ and $(\mathsf{hsame}(k', \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(k', \mathsf{E0}(t)) \to \bot)$.*

Proof. Apply Theorem 33.236 to the displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, endpoint, and selected visible payload data. This supplies k, k', p, q, a, b together with the ten selected visible public factorization clauses, in particular $\mathsf{hsame}(k, \mathsf{E1}(p))$ and $\mathsf{hsame}(k', \mathsf{E1}(q))$.

For the left intermediate endpoint, fix $t : \mathsf{BHist}$. A competing presentation $\mathsf{hsame}(k, \mathsf{e})$ composes with $\mathsf{hsame}(k, \mathsf{E1}(p))$ and contradicts empty-versus-one generated-history no-confusion. A competing presentation $\mathsf{hsame}(k, \mathsf{E0}(t))$ composes with $\mathsf{hsame}(k, \mathsf{E1}(p))$ and contradicts zero-versus-one generated-history no-confusion.

For the right intermediate endpoint, the same argument uses $\mathsf{hsame}(k', \mathsf{E1}(q))$. Any presentation $\mathsf{hsame}(k', \mathsf{e})$ contradicts empty-versus-one no-confusion, and any presentation $\mathsf{hsame}(k', \mathsf{E0}(t))$ contradicts zero-versus-one no-confusion.

Package the selected visible public factorization witnesses with these four constructor-exclusion functions for each t. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized present intermediate constructor exclusion`

33.25 Tagged option composite factorization intermediate classifier determinacy

Theorem 33.254 (Tagged option composite factorization intermediate classifier)**.** *This theorem fits the ten-field template as follows: the traditional target is functorial* $\mathsf{Option}^{\uparrow}$ *middle determinacy; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is* $\mathsf{TaggedOptionHistoryCarrier}_T$*; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligation is branch preservation plus middle endpoint classifier construction; the ledger records absent*

or present intermediate endpoint provenance; the core theorem is the displayed classifier conclusion; exactness is absent-or-same-present exhaustion; the standard bridge is equality of the none *and* some *middle option; and the status is* ProofTarget.

Let $S, T, U : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source, intermediate, and target carriers, let* $\sim_S$, $\sim_T$, *and* $\sim_U$ *be their classifiers, and let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$$

and

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U).$$

Assume payload preservation fields

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a))$$

and

$$\rho_\epsilon : \forall b.\ T(b) \to U(\epsilon.\mathsf{map}(b)).$$

Assume endpoint stability for both descents:

$$\eta_\delta : \forall x\, y.\ S(x) \to S(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\delta.\mathsf{map}(x), \delta.\mathsf{map}(y))$$

and

$$\eta_\epsilon : \forall x\, y.\ T(x) \to T(y) \to \mathsf{hsame}(x, y) \to \mathsf{hsame}(\epsilon.\mathsf{map}(x), \epsilon.\mathsf{map}(y)).$$

Let $\mathsf{cert}_T : \mathsf{NameCert}(T, \sim_T)$. *For histories* $h, k, k', m : \mathsf{BHist}$, *suppose the same composed endpoint pair factors through both* k *and* k'*:*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k) \wedge \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, m)$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k') \wedge \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', m).$$

Then the two intermediate tagged-option endpoints are classified:

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. Apply Theorem 33.246 to the two displayed δ factors, the two displayed ϵ factors, and the preservation and stability data. This gives the canonical intermediate endpoint split.

In the absent branch, the canonical split supplies $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$. Use these two empty presentations to build the absent–absent branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

In the present branch, retain the source payload a and the intermediate payload witness $T(\delta.\mathsf{map}(a))$. The canonical split supplies

$$\mathsf{hsame}(k, \mathsf{E1}(\delta.\mathsf{map}(a))) \wedge \mathsf{hsame}(k', \mathsf{E1}(\delta.\mathsf{map}(a))).$$

Use these two one-payload presentations and $\mathsf{cert}_T.\mathsf{equiv_refl}$ at $\delta.\mathsf{map}(a)$ to build the present–present branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

The classifier is therefore determined by the same absent-or-same-present exhaustion that controls the canonical intermediate endpoint presentation. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel composite factorization intermediate classifier`

33.26 Option composite image classifier normalized present intermediate payload determinacy

Theorem 33.255 (Tagged option composite image classifier normalized present intermediate payload determinacy)**.** *Let the carriers S, T, U, classifiers $\sim_S, \sim_T, \sim_U$, descents δ, ϵ, preservation fields $\rho_\delta, \rho_\epsilon$, endpoint-stability field η_ϵ, certificates $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$, compatibility fields, endpoints m, m', u, u', and selected visible payloads x, y satisfy exactly the hypotheses of Theorem 33.236. Then there exist $k, k', p, q, a, b : \mathsf{BHist}$ such that*

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for k, k', p, q, a, b;*
2. *the left present intermediate payload is determined:*

$$\forall r : \mathsf{BHist}.\ \mathsf{hsame}(k, \mathsf{E1}(r)) \to \mathsf{hsame}(p, r);$$

3. *the right present intermediate payload is determined:*

$$\forall s : \mathsf{BHist}.\ \mathsf{hsame}(k', \mathsf{E1}(s)) \to \mathsf{hsame}(q, s).$$

Proof. Apply Theorem 33.236 to the displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, endpoint, and selected visible payload data. This supplies k, k', p, q, a, b together with the ten selected visible public factorization clauses, including $\mathsf{hsame}(k, \mathsf{E1}(p))$ and $\mathsf{hsame}(k', \mathsf{E1}(q))$.

For the left intermediate endpoint, fix $r : \mathsf{BHist}$ and assume $\mathsf{hsame}(k, \mathsf{E1}(r))$. Apply Theorem 33.122 at k to the selected presentation $\mathsf{hsame}(k, \mathsf{E1}(p))$ and the competing presentation $\mathsf{hsame}(k, \mathsf{E1}(r))$. This yields $\mathsf{hsame}(p, r)$.

For the right intermediate endpoint, fix $s : \mathsf{BHist}$ and assume $\mathsf{hsame}(k', \mathsf{E1}(s))$. Apply Theorem 33.122 at k' to the selected presentation $\mathsf{hsame}(k', \mathsf{E1}(q))$ and the competing presentation $\mathsf{hsame}(k', \mathsf{E1}(s))$. This yields $\mathsf{hsame}(q, s)$.

Package the selected visible public factorization witnesses with the two payload-determinacy functions. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized present intermediate payload determinacy`

33.27 Option history classifier unary two-sided context

Theorem 33.256 (Option history carrier unary two-sided context)**.** *Let $p, q, h : \mathsf{BHist}$. If $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(q)$ hold, then*

$$\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(h) \iff \mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}(\mathsf{append}(p, \mathsf{append}(h, q))).$$

Proof. For the forward direction, first apply Theorem 33.6 with suffix q. Then apply Theorem 33.5 with prefix p.

For the reverse direction, split the contextual nullable carrier witness.

In the absent branch, use $\mathsf{hsame_iff_eq}$ to turn the empty contextual endpoint witness into equality. Apply $\mathsf{append_eq_empty_iff}$ to the outer append, and then again to the inner append, recovering $\mathsf{hsame}(h, \mathsf{e})$.

In the source branch, use $\mathsf{unary_append_right_factor}$ on the outer append to recover $\mathsf{UnaryHistory}(\mathsf{append}(h, q))$. Then use $\mathsf{unary_append_left_factor}$ on the inner append to recover $\mathsf{UnaryHistory}(h)$.

These two recovered branches are exactly the nullable carrier alternatives for h. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryCarrier unary two sided context iff`

Theorem 33.257 (Option history classifier unary two-sided context)**.** *This theorem fits the ten-field template as follows: the traditional target is context-stable nullable optional history classification; the BEDC source is* BHist *with* UnaryHistory, append, *and* hsame*; the carrier is* $\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}$*; the classifier is* $\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}$*; the stability obligations are unary prefix and suffix closure together with cancellation; the ledger is the hidden common unary context; the core theorem is the displayed equivalence; exactness is recovery of the uncloaked classifier by append cancellation and unary factor readback; the standard bridge is option values under common unary context; and the status is* ProofTarget*.*

Let $p, q, h, k : \mathsf{BHist}$*. If* $\mathsf{UnaryHistory}(p)$ *and* $\mathsf{UnaryHistory}(q)$ *hold, then*

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(h,k) \iff \mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(\mathsf{append}(p,\mathsf{append}(h,q)),\mathsf{append}(p,\mathsf{appe}$$

Proof. For the forward direction, first apply Theorem 33.9 with suffix q to place the common unary right context on both endpoints. Then apply Theorem 33.8 with prefix p to place the common unary left context.

For the reverse direction, unpack the contextual nullable classifier into its two endpoint carrier witnesses and the contextual hsame field. Each endpoint carrier is read back from the surrounding context. In an absent branch, use hsame_iff_eq and append_eq_empty_iff twice to recover the corresponding uncloaked empty endpoint witness.

In a source branch, use unary_append_right_factor on the outer append and unary_append_left_factor on the inner append to recover $\mathsf{UnaryHistory}(h)$ or $\mathsf{UnaryHistory}(k)$ as required by the nullable carrier.

It remains to recover the uncloaked classifier relation. Convert the contextual hsame field to equality, cancel the common left prefix by append_left_cancel, and then cancel the common right suffix by append_right_cancel. Repack the two recovered endpoint carriers together with the recovered $\mathsf{hsame}(h,k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier unary two sided context iff`

Theorem 33.258 (Option unary context composition law)**.** *Let* $p, q, r, s, h, k : \mathsf{BHist}$*. For histories* u, v, x*, write* $C_{u,v}(x) := \mathsf{append}(u, \mathsf{append}(x, v))$*. If* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{UnaryHistory}(q)$*,* $\mathsf{UnaryHistory}(r)$*, and* $\mathsf{UnaryHistory}(s)$ *hold, then*

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(h,k) \iff \mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(C_{p,q}(C_{r,s}(h)), C_{\mathsf{append}(p,r),\mathsf{append}(s,q)}(k)).$$

Proof. Set $L := \mathsf{append}(p,r)$ and $R := \mathsf{append}(s,q)$. Unary append closure applied to p, r and to s, q gives $\mathsf{UnaryHistory}(L)$ and $\mathsf{UnaryHistory}(R)$. The associativity laws for BEDC append identify $C_{p,q}(C_{r,s}(x))$ with $C_{L,R}(x)$ for every history x; in particular, they provide the hsame witnesses needed to move classifiers between the nested source endpoint and the normalized endpoint.

For the forward direction, apply Theorem 33.257 to the classifier between h and k with unary contexts L and R. This gives

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(C_{L,R}(h), C_{L,R}(k)).$$

Transport the source endpoint along the associativity hsame witness from $C_{L,R}(h)$ to $C_{p,q}(C_{r,s}(h))$. The target endpoint is already $C_{L,R}(k) = C_{\mathsf{append}(p,r),\mathsf{append}(s,q)}(k)$ by the definitions of L and R.

For the reverse direction, start with the displayed classifier and transport its source endpoint back across the same associativity witness. This produces

$$\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(C_{L,R}(h), C_{L,R}(k)).$$

Apply the reverse implication of Theorem 33.257 with contexts L and R. Append cancellation inside that theorem removes the common normalized ledger and recovers $\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}(h, k)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.OptionHistoryClassifier unary context composition iff`

Theorem 33.259 (Option finite-data scoped row exhaustion)**.** *The* $\mathsf{Option}^{\uparrow}$ *finite-data public surface exports only the following rows: nullable and tagged option carrier rows; absent and present branch readbacks; payload-descent image carrier and image-classifier rows; componentwise* hsame *transports;* Cont *routes induced by the displayed branch data; source* NameCert *rows; and the visible* Pkg *provenance already listed by the unary-context, tagged-option semantic, and payload-descent public packages. It exports no host* Option *object, quotient row, hidden payload equality row, or unlogged branch eliminator.*

Proof. The nullable side is exhausted by the moved unary-context theorems above. They only transport $\mathsf{OptionHistoryCarrier}_{\mathsf{UnaryHistory}}$ and $\mathsf{OptionHistoryClassifier}_{\mathsf{UnaryHistory}}$ through common unary ledgers, and their proofs recover the uncloaked rows by append cancellation and constructor-branch readback.

The tagged semantic rows are the explicit carrier, classifier, exactness, and NameCert rows recorded in the tagged option semantic package. Their branch analysis has only the absent constructor and the present constructor carrying a source witness, so projecting any public row yields either e readback or an $\mathsf{E1}(a)$ readback together with the source certificate data.

The payload-descent image rows are exhausted by the public readback and image classifier blocks. A target endpoint is public only through a source tagged endpoint and a displayed TaggedOptionMapRel route; a classified target pair is public only through two such routes and a source tagged classifier witness.

Combining these projections gives row exhaustion. Constructor branch exhaustion accounts for absent and present cases, componentwise hsame transport accounts for endpoint replacement, Cont records only the displayed routes, and Pkg provenance records only the listed certificates and descent data. None of these projections produces a host option value, a quotient representative, a payload equality outside hsame or the source classifier, or a branch eliminator without a logged constructor case. □

The absent/present tag row for $\mathsf{Option}^{\uparrow}$ consumes the finite tag discipline of chapter 32 before adding its payload classifier. The option ledger therefore extends $\mathsf{Bool}^{\uparrow}$ with an optional payload coordinate rather than introducing a second tag carrier.

33.28 Tagged option map relation R-corner classifier coherence

Theorem 33.260 (Tagged option map relation R-corner classifier coherence)**.** *This theorem fits the per-object template as follows: the traditional target is right-corner option-map endpoint determinacy; the BEDC source is* BHist *with one* DescentCertificate*; the carrier is the tagged option endpoint shape; the classifiers are* $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}$ *and* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligations are shared-target source reflection and shared-source target normal form; the ledger records the three R-corner legs; the core theorem is simultaneous source and target endpoint classification; exactness is absent-or-present exhaustion on both axes; the standard bridge is* none/some *map readback; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source and target carriers, let* $\sim_S$ *and* $\sim_T$ *be their classifiers, let*

$$\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T),$$

and assume payload preservation

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Assume $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$ *and reflected source readback from visible target equality,*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\delta(S, \sim_S).$$

For histories h, h', k, k'*, if*

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k),$$

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k),$$

and

$$\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k'),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h') \wedge \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. First compare the two legs with shared target endpoint k. Apply Theorem 33.144 to the map-relation witnesses $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h, k)$ and $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$, using the reflected source readback hypothesis. This gives $\mathsf{TaggedOptionHistoryClassifier}_{S,\sim_S}(h, h')$.

Next compare the two legs with shared source endpoint h'. Apply Theorem 33.146 to the map-relation witnesses $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k)$ and $\mathsf{TaggedOptionMapRel}_\delta(S, T, \rho_\delta; h', k')$, using source hsame compatibility. This gives $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$.

Pairing these two classifier conclusions yields the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionMapRel R corner classifier coherence`

Theorem 33.261 (Tagged option common-right visible payload classification)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history carrier and let* $R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a symmetric and transitive payload classifier. Assume that visible* hsame *between source payloads is reflected into* R*. For histories* h, r, k, a, c, b*, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,R}(h, k) \quad \textit{and} \quad \mathsf{TaggedOptionHistoryClassifier}_{S,R}(r, k),$$

and a, c, b *are displayed source payloads with*

$$\mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(r, \mathsf{E1}(c)), \qquad \mathsf{hsame}(k, \mathsf{E1}(b)),$$

then $R(a, c)$*.*

Proof. Use right-visible branch inversion on both classifiers at the common right presentation $\mathsf{hsame}(k, \mathsf{E1}(b))$. The first classifier exposes a left payload classified with a right payload history-same to b, while the second exposes a right payload chain with the same visible b.

Invert the two visible E1 presentations at h and r to compare the displayed payloads a and c with the exposed classifier payloads. The source-hsame reflection converts these payload comparisons into R-legs. Transitivity composes a to the common visible payload b through the first classifier. The second classifier composes c to the same b; symmetry turns that leg around, and one final transitivity step yields $R(a, c)$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier common right visible payload classificatio`

Theorem 33.262 (Tagged option present-chain visible endpoint classification)**.** *Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history carrier and let* $R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a transitive payload classifier. Assume that visible* hsame *between source payloads is reflected into* R*. For histories* h, k, r, a, c*, if*

$$\mathsf{TaggedOptionHistoryClassifier}_{S,R}(h,k) \quad \textit{and} \quad \mathsf{TaggedOptionHistoryClassifier}_{S,R}(k,r),$$

if $S(a)$ *and* $S(c)$ *hold, and if the endpoints are visibly present as*

$$\mathsf{hsame}(h, \mathsf{E1}(a)), \qquad \mathsf{hsame}(r, \mathsf{E1}(c)),$$

then $R(a, c)$*.*

Proof. Compose the two tagged classifiers by transitivity of R to classify h and r. The absent branch would present h as both e and $\mathsf{E1}(a)$, which is impossible by BEDC constructor no-confusion. In the present branch, invert the two visible $\mathsf{E1}$ presentations to compare a and c with the exposed payload witnesses. The source-hsame reflection supplies the two outer R-legs, and transitivity composes them with the exposed payload classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionHistoryClassifier present chain visible endpoint classification`

33.29 Tagged option payload descent image target classifier closure

Theorem 33.263 (Tagged option payload descent image classifier target classifier closure)**.** *This theorem fits the per-object template as follows: the traditional target is functorial* OptionUp *target classifier closure; the BEDC source is* BHist *with a descent certificate; the carrier is the transported target tagged-option carrier; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligation is branch exactness plus descent respect; the ledger is absent or transported present payload provenance; the core theorem is the displayed classifier implication; exactness is the branch split; the standard bridge is* none/some *classifier closure; and the status is* ProofTarget*.*

Let $S, T : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history carriers, let* $\sim_S$ *and* $\sim_T$ *be classifiers, let* $\delta : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_S, \sim_T)$*, and assume payload preservation*

$$\rho_\delta : \forall a.\ S(a) \to T(\delta.\mathsf{map}(a)).$$

Let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$ *and assume* $\mathsf{TaggedOptionSourceHsameCompatible}(S, \sim_S)$*. For target endpoints* $k, k' : \mathsf{BHist}$*,*

$$\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k') \to \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k').$$

Proof. Apply Theorem 33.162 to the displayed carrier, classifier, descent, preservation, certificate, compatibility, and endpoint data. The resulting branch split is either absent-absent target presentation or transported present-payload presentation.

In the absent branch, the split supplies $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$. Use the absent-absent constructor of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$.

In the present branch, keep the witnesses $a, b : \mathsf{BHist}$, the source carrier witnesses, $a \sim_S b$, the two transported target carrier witnesses, and the two target endpoint presentations. The descent certificate's respect field gives

$$\delta.\mathsf{respects}(a \sim_S b) : \delta.\mathsf{map}(a) \sim_T \delta.\mathsf{map}(b).$$

Use the present-present constructor of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$ with target witnesses $\delta.\mathsf{map}(a)$ and $\delta.\mathsf{map}(b)$, their two T witnesses, the transported classifier witness, and the two endpoint presentations. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier target classifier closure`

33.30 Option composite image classifier normalized present intermediate classifier determinacy

Theorem 33.264 (Tagged option composite image classifier normalized present intermediate classifier determinacy)**.** *This theorem fits the per-object template as follows: the traditional target is classifier-level readback for the intermediate public endpoints in functorial* OptionUp*; the BEDC source is* BHist *with two composable* DescentCertificate *maps; the carrier is the selected intermediate* E1 *endpoint shape; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligations are selected visible public factorization, present-payload determinacy,* $\mathsf{TaggedOptionSourceHsameCompatible}(T, \sim_T)$*, and the equivalence fields of* cert_T*; the ledger records selected intermediate endpoints and competing visible intermediate payloads; the core theorem is classifier-level determinacy; exactness says every competing visible intermediate pair lies in the same* $\sim_T$ *class; the standard bridge is some-payload functorial readback; and the status is* ProofTarget*. Let the carriers* S, T, U*, classifiers* $\sim_S, \sim_T, \sim_U$*, descents* δ, ϵ*, preservation fields* $\rho_\delta, \rho_\epsilon$*, endpoint-stability field* η_ϵ*, certificates* $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$*, compatibility fields, endpoints* m, m', u, u'*, and selected visible payloads* x, y *satisfy exactly the hypotheses of Theorem 33.236. Then there exist* $k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for* k, k', p, q, a, b*;*
2. *the selected intermediate endpoints are classified by*

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k');$$

3. *every competing visible intermediate pair lies in the selected intermediate classifier class:*

$$\forall r, s : \mathsf{BHist}.\ T(r) \to T(s) \to \mathsf{hsame}(k, \mathsf{E1}(r)) \to \mathsf{hsame}(k', \mathsf{E1}(s)) \to r \sim_T s.$$

Proof. Apply Theorem 33.255 to the displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, endpoint, and selected visible payload data. This supplies k, k', p, q, a, b, the ten selected visible public factorization clauses, and the two present-payload determinacy functions.

The present branch of the ten factorization clauses gives $T(p)$, $T(q)$, $\mathsf{hsame}(k, \mathsf{E1}(p))$, $\mathsf{hsame}(k', \mathsf{E1}(q))$, and $p \sim_T q$. These witnesses assemble

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$$

by the present constructor of the tagged option history classifier.

Now fix competing visible intermediate payloads $r, s : \mathsf{BHist}$ with $T(r)$, $T(s)$, $\mathsf{hsame}(k, \mathsf{E1}(r))$, and $\mathsf{hsame}(k', \mathsf{E1}(s))$. The two determinacy functions yield $\mathsf{hsame}(p, r)$ and $\mathsf{hsame}(q, s)$.

Target hsame compatibility converts these to $p \sim_T r$ and $q \sim_T s$. Using the symmetry field of cert_T on $p \sim_T r$ gives $r \sim_T p$; two applications of the transitivity field of cert_T compose $r \sim_T p$, $p \sim_T q$, and $q \sim_T s$ into $r \sim_T s$. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized present intermediate classifier determinacy`

Theorem 33.265 (Tagged option composite image classifier normalized intermediate classifier coverage)**.** *This theorem fits the per-object template as follows: the traditional target is branch-complete functorial* OptionUp *intermediate classifier coverage; the BEDC source is* BHist *with two composable descents; the carrier is the intermediate endpoint pair supplied by normalized public factorization; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligations are normalized public readback completion, absent endpoint factorization, present payload determinacy, and source compatibility for* T*; the ledger records absent intermediate endpoints or selected visible present intermediate payloads; the core theorem is classified intermediate coverage; exactness says each normalized composite classifier has an absent or present intermediate factor whose two intermediate endpoints are classified; the standard bridge is none/some functorial composition; and the status is* ProofTarget*.*

Let the carriers S, T, U*, classifiers* $\sim_S, \sim_T, \sim_U$*, descents* δ, ϵ*, preservation fields* $\rho_\delta, \rho_\epsilon$*, endpoint-stability field* η_ϵ*, certificates* $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$*, compatibility fields, and endpoints* m, m', u, u' *satisfy exactly the hypotheses of Theorem 33.249. Then either*

1. *the normalized public endpoints are absent:*

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e});$$

2. *there is no visible present public pair:*

$$\neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y));$$

3. *there exist* $k, k' : \mathsf{BHist}$ *such that*

   (a) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*

   (b) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$*;*

   (c) $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$*;*

   (d) $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$*;*

   (e) no visible present intermediate pair occurs:

$$\neg\exists p, q : \mathsf{BHist}.\ T(p) \wedge T(q) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k', \mathsf{E1}(q)).$$

or there exist $x, y, k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for* x, y, k, k', p, q, a, b*;*

2. *the selected intermediate endpoints are classified:*

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k');$$

3. every competing visible intermediate payload pair lies in the selected intermediate classifier class:

$$\forall r, s : \mathsf{BHist}.\ T(r) \to T(s) \to \mathsf{hsame}(k, \mathsf{E1}(r)) \to \mathsf{hsame}(k', \mathsf{E1}(s)) \to r \sim_T s.$$

Proof. Apply Theorem 33.249 to the displayed composite witness, endpoint normalization witnesses, descents, preservation fields, stability field, certificates, and compatibility fields. This gives the normalized factored public readback completion at (u, u').

In the absent branch, retain the absent public endpoint presentations, the public visible-pair exclusion, and the intermediate witnesses k, k'. The factored completion supplies the δ image-classifier factor, the two ϵ map-relation links to u, u', the presentations $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$, and the exclusion of visible present intermediate pairs.

Build

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$$

by the absent-absent constructor from $\mathsf{hsame}(k, \mathsf{e})$ and $\mathsf{hsame}(k', \mathsf{e})$. Package this classifier with the retained absent data.

In the present branch, retain the displayed payloads x, y, k, k', p, q, a, b and the selected visible factorization clauses supplied by the factored completion. Apply Theorem 33.264 to the same displayed data and selected visible payloads.

That theorem supplies the classifier

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$$

and the universal competing visible intermediate payload conclusion. Package these two fields with the retained selected visible public factorization clauses. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized intermediate classifier coverage`

33.31 Option composite image classifier normalized intermediate factor coherence

Theorem 33.266 (Tagged option composite image classifier normalized intermediate factor coherence)**.** *Let $T, U : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history carriers, let $\sim_T$ and $\sim_U$ be their classifiers, let*

$$\epsilon : \mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, \sim_T, \sim_U),$$

and assume payload preservation

$$\rho_\epsilon : \forall a.\ T(a) \to U(\epsilon.\mathsf{map}(a)).$$

Assume reflected source readback from visible public equality,

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\epsilon(T, \sim_T).$$

For histories k, l, k', l', u, u', if

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u),$$

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l, u),$$

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u'),$$

and

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l', u'),$$

then

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, l) \wedge \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k', l').$$

Proof. First compare the two intermediate factors over the normalized public endpoint u. Apply Theorem 33.243 to the two map-relation witnesses $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ and $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l, u)$. This gives either simultaneous empty presentations of k, l, u or a present-payload package ending in $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, l)$.

Next compare the two intermediate factors over the normalized public endpoint u'. Apply Theorem 33.243 to the two map-relation witnesses $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$ and $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l', u')$. This gives either simultaneous empty presentations of k', l', u' or a present-payload package ending in $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k', l')$.

In an absent branch of the comparison over u, the empty presentations of k and l assemble the absent branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, l)$. In its present branch, retain the classifier supplied by the reflected normal form.

In an absent branch of the comparison over u', the empty presentations of k' and l' assemble the absent branch of $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k', l')$. In its present branch, retain the classifier supplied by the reflected normal form.

Combining the classifier obtained over u with the classifier obtained over u' yields the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized intermediate factor coherence`

33.32 Option composite image classifier normalized selected intermediate factor uniqueness

Theorem 33.267 (Tagged option composite image classifier normalized selected intermediate factor uniqueness)**.** *Let the carriers S, T, U, classifiers $\sim_S, \sim_T, \sim_U$, descents δ, ϵ, preservation fields $\rho_\delta, \rho_\epsilon$, endpoint-stability field η_ϵ, certificates $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$, compatibility fields, endpoints m, m', u, u', and selected visible payloads x, y satisfy exactly the hypotheses of Theorem 33.236. Assume also reflected source readback from visible public equality,*

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\epsilon(T, \sim_T).$$

Then there exist $k, k', p, q, a, b : \mathsf{BHist}$ such that

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for k, k', p, q, a, b;*
2. *the selected intermediate factors are unique against every competing intermediate factor over the same normalized public endpoints:*

$$\forall l, l' : \mathsf{BHist}.\ \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l, u) \to \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l', u') \to \mathsf{TaggedOptionHistor}$$

Proof. Apply Theorem 33.236 to the displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, endpoint, and selected visible payload data. This supplies k, k', p, q, a, b together with the ten selected visible public factorization clauses.

The second selected factorization clause supplies the two selected ϵ map-relation legs

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$$

and

$$\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u').$$

Fix competing intermediate endpoints $l, l' : \mathsf{BHist}$ with map-relation witnesses from l to u and from l' to u'. Apply Theorem 33.266 to the selected left leg, the competing left leg, the selected right leg, and the competing right leg, using the reflected source readback hypothesis. This gives

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, l) \wedge \mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k', l').$$

Package the selected factorization clauses with this universal comparison against arbitrary competing intermediate factors over u and u'. □

Theorem 33.268 (Tagged option composite image classifier normalized unique intermediate factor coverage)**.** *This theorem fits the per-object template as follows: the traditional target is branch-complete functorial* OptionUp *factor uniqueness; the BEDC source is* BHist *with two composable descent certificate maps; the carrier is the selected intermediate endpoint pair; the classifier is* $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}$*; the stability obligations are normalized readback completion, selected visible factorization, reflected source readback, and factor coherence; the ledger records absent or selected present intermediate provenance; the core theorem is displayed uniqueness coverage; exactness says every admitted intermediate factor over the same normalized public endpoints lies in the same classifier class; the standard bridge is none/some functorial composition; and the status is* ProofTarget.

Let the carriers S, T, U, classifiers $\sim_S, \sim_T, \sim_U$, descents δ, ϵ, preservation fields $\rho_\delta, \rho_\epsilon$, endpoint-stability field η_ϵ, certificates $\mathsf{cert}_S, \mathsf{cert}_T, \mathsf{cert}_U$, compatibility fields, and endpoints m, m', u, u' satisfy exactly the hypotheses of Theorem 33.249. Assume also reflected source readback from visible public equality,

$$\mathsf{TaggedOptionPayloadDescentReflectsSource}_\epsilon(T, \sim_T).$$

Then either

1. *the normalized public endpoints are absent:*

$$\mathsf{hsame}(u, \mathsf{e}) \wedge \mathsf{hsame}(u', \mathsf{e});$$

2. *there is no visible present public pair:*

$$\neg\exists x, y : \mathsf{BHist}.\ U(x) \wedge U(y) \wedge \mathsf{hsame}(u, \mathsf{E1}(x)) \wedge \mathsf{hsame}(u', \mathsf{E1}(y));$$

3. *there exist k, k' : BHist such that*

   (a) $\mathsf{TaggedOptionPayloadDescentImageClassifier}_\delta(S, T, \rho_\delta; k, k')$*;*

   (b) $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k, u)$ *and* $\mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; k', u')$*;*

   (c) $\mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{hsame}(k', \mathsf{e})$*;*

   (d) $\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k')$*;*

(e) *no visible present intermediate pair occurs:*

$$\neg\exists p, q : \mathsf{BHist}.\ T(p) \wedge T(q) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k', \mathsf{E1}(q)).$$

(f) *every admitted intermediate factor over the same normalized public endpoints lies in the selected absent classifier class:*

$$\forall l, l' : \mathsf{BHist}.\ \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l, u) \to \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l', u') \to \mathsf{TaggedOptionH}$$

or there exist $x, y, k, k', p, q, a, b : \mathsf{BHist}$ *such that*

1. *the ten selected visible public factorization clauses of Theorem 33.236 hold for* x, y, k, k', p, q, a, b*;*
2. *the selected intermediate endpoints are classified:*

$$\mathsf{TaggedOptionHistoryClassifier}_{T,\sim_T}(k, k');$$

3. *every admitted intermediate factor over the same normalized public endpoints lies in the selected present classifier class:*

$$\forall l, l' : \mathsf{BHist}.\ \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l, u) \to \mathsf{TaggedOptionMapRel}_\epsilon(T, U, \rho_\epsilon; l', u') \to \mathsf{TaggedOptionHistor}$$

Proof. Apply Theorem 33.265 to the displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, and endpoint data. This gives the normalized branch split with classified intermediate witnesses.

In the absent branch, retain the absent public endpoint presentations, the visible public exclusion, and the witnesses k, k' with their δ image-classifier factor, ϵ map-relation links, absent intermediate presentations, selected classifier, and visible intermediate exclusion.

Fix arbitrary intermediate endpoints $l, l' : \mathsf{BHist}$ mapping to the same normalized public endpoints u, u'. Apply Theorem 33.266 to the selected absent map-relation legs and the competing map-relation legs, using reflected source readback. This yields the required classifier comparison between k, l and between k', l'.

In the present branch, retain the selected visible factorization data and the selected intermediate classifier supplied by the coverage theorem. Apply Theorem 33.267 to the same displayed carrier, classifier, descent, preservation, stability, certificate, compatibility, endpoint, selected visible payload, and reflected source readback data.

The selected uniqueness theorem supplies the universal comparison against arbitrary admitted intermediate factors over u and u'. Package that comparison with the selected visible factorization clauses and the retained intermediate classifier. □

Lean4 ✓ `BEDC.Derived.OptionUp.TaggedOptionPayloadDescentImageClassifier composite normalized unique intermediate factor coverage`

Theorem 33.269 (Option finite-data public exit package)**.** *The rows named by* OptionHistoryCarrier*,* OptionHistoryClassifier*,* OptionHistoryLedgerPolicy*, carrier transport, classifier symmetry and transitivity, absent-source separation, exclusive readback, and ledger endpoint transport assemble the public finite-data exit package for* $\mathsf{Option}^{\uparrow}$*. The public package exposes only the finite* BHist *carrier rows,* hsame *classifier rows,* Cont *transport rows,* Pkg *provenance rows, and the displayed* $\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$ *certificate rows; it exports no host option type, external quotient, or hidden payload selector.*

Proof. The carrier and classifier rows are the unary-context and scoped-row surfaces of Definition 33.256, Definition 33.257, and Definition 33.259. These blocks provide the finite BHist rows, the hsame comparison relation, and the displayed Cont routes used by every public endpoint.

The readback and separation rows are Definition 33.194, Definition 33.197, and the payload-descent image readback blocks. They split absent and present endpoints, rule out mixed source readback, and keep every transported endpoint inside the same finite ledger.

Finally the semantic name certificate option_history_semantic_name_certificate packages these rows as the canonical $\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$ surface. Since each cited block factors through BHist, hsame, Cont, Pkg, and the displayed certificate rows, the public exit has no additional data channel. □

Lean4 ✓ `BEDC.Derived.OptionUp.option history semantic name certificate`

Theorem 33.270 (Option payload-map totality)**.** *Every public payload-map read of an accepted* $\mathsf{Option}^{\uparrow}$ *packet first reads the visible* $\mathsf{Bool}^{\uparrow}$ *tag. If the tag is the displayed absent endpoint, the read returns the displayed none witness and exposes no payload row. If the tag is the displayed present endpoint, the read exposes exactly one payload row through the present-payload classifier and the* $\mathsf{Option}^{\uparrow}$ *classifier, with replay and provenance carried by the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$ *rows used by the public exit package.*

Proof. Begin with the public finite-data exit package of Definition 33.269. It restricts every public read to the displayed finite carrier, classifier, replay, provenance, and certificate rows, so a payload-map consumer has no channel before the visible tag is inspected.

The absent case is fixed by the exclusive branch readback and cross-source separation theorems. The displayed absent endpoint is the none witness, and the visible payload classification theorems rule out any present payload row in that branch.

In the present case, use map relation coherence together with visible payload classification. The intermediate factor coverage and unique intermediate factor statements in this file identify the single present payload row read by the public map relation, and the option classifier records it as the displayed some witness.

Thus the payload-map read is total as a visible $\mathsf{Option}^{\uparrow}$ witness: absent reads return none, present reads return one classified payload row, and both branches stay inside the same displayed public package. □

Closure status for $\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Option}^{\uparrow}$ packet is generated from the finite nullable envelope over $\mathsf{Bool}^{\uparrow}$ tags, the tagged absent and present constructor rows, exclusive branch readback, payload descent image classifiers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local semantic NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$, maturePackageClosed)

Scope closed: The closure binds the nullable envelope, tagged constructor surface, exclusive branch readback, payload descent image classifier, finite tag dependency on $\mathsf{Bool}^{\uparrow}$, and public finite-data exit package for $\mathsf{Option}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Option}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.OptionUp.option\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: The chapter does not close full option monad laws, arbitrary eliminators into certified targets, dependent options, host option equality, or quotient payload selection.

Upgrade path: No further upgrade – chapter at top of formal axis.

$\mathsf{Prod}^{\uparrow}$ finite-data product envelope is consumed by the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ product-window routes of chapter 126 and chapter 301.

Chapter 34

A Concrete Naming Certificate for $\mathsf{Prod}^{\uparrow}$

The product interface $\mathsf{Prod}^{\uparrow}$ has two classifier levels plus the checked envelope. ProdClassifierSpec is the typed finite-pair helper. $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$ is the public BEDC-history certificate classifier induced by ProdPairRep and the source classifiers, while ProdHistoryClassifier is the checked envelope classifier using endpoint hsame. It is licensed only as a derived constructor over the two sources.

The finite-data sibling lattice enters the product chapter through chapter 32, chapter 33, chapter 35, and chapter 36. Bool supplies tag discipline, Option and Sum supply branch-tag boundary cases, and List supplies ordered spine discipline, while Product supplies the two-coordinate continuation envelope.

34.1 Roadmap template status

1. **Traditional target**: ordered product data with one left component and one right component.
2. **BEDC source**: the checked history-envelope certificate uses inhabited source-history predicates L and R; the explicit typed interface uses two licensed source carriers A and B, each with its own name certificate.
3. **Carrier choice**: the paired history carrier records a left history, a right history, and the continuation that combines them.
4. **Classifier choice**: ProdClassifierSpec is the typed finite-pair helper; $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$ is the public history-level classifier from ProdPairRep and source classifiers; ProdHistoryClassifier is the checked envelope classifier using endpoint hsame.
5. **Stability obligations**: component transport, continuation transport, unary closure from components, history-envelope classifier symmetry and transitivity, typed classifier symmetry and transitivity, typed coordinate inversion, ProdPairRepCoherent, and componentwise-to-envelope descent.
6. **Ledger**: no product quotient is used; pairing records both component witnesses, their generated continuation, and the displayed pair ledger soundness needed by the public history-level certificate.

7. **Core theorem list**: append introduction, continuation introduction, carrier transport, unary closure, classifier equivalence, component-envelope inversion, ProdPairRepCoherent criteria, displayed pair ledger soundness, componentwise-to-envelope descent under hsame-sound source classifiers, and the semantic name certificate.
8. **Exactness predicate**: typed classification holds exactly when both finite-pair coordinates classify in their respective sources; public history-level classification holds through displayed ProdPairRep components and source classifiers; envelope classification uses endpoint hsame.
9. **Standard bridge**: the checked history-envelope bridge reads back only as a generated product endpoint with a displayed pair representation. Ordinary ordered-pair and componentwise history readback belong to the componentwise certificate target under $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ and $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$.
10. **Status**: Theorem 34.107 is the checked history-envelope certificate via BEDC.Derived.ProdUp.prod_history_sema The Phase Finite Data public-object product scope is the checked componentwise certificate at Theorem 34.60, supported by the checked componentwise classifier, ProdPairRepCoherent field, displayed-pair ledger, descent comparison, and endpoint-invariance rows.

34.2 Carrier definition

Definition 34.1 (Prod carrier)**.** *For licensed source carriers A and B, define a pair carrier with constructor* $\mathsf{pair}(a, b)$*.*

Lean4 (definition) `BEDC.Derived.ProdUp.ProdCarrier`

34.3 The certificate

Definition 34.2 (Prod source specification)**.** *For the checked history-envelope certificate, the source specification is a pair of inhabited predicates $L, R : \mathsf{BHist} \to \mathsf{Prop}$. The carrier is* $\mathsf{ProdHistoryCarrier}(L, R)$*, and the classifier is* $\mathsf{ProdHistoryClassifier}(L, R)$*; no source* NameCert *is required at this scope.*

For the explicit ordered-product interface, the source data are two licensed carriers with NameCert *witnesses. Those certificates are used for typed component transport, pair-representation coherence, and displayed component-exposure fields.*

Lean4 (definition) `BEDC.Derived.ProdUp.ProdHistoryCarrier`

Theorem 34.3 (Product history carrier append introduction)**.** *If l lies in the left source carrier and r lies in the right source carrier, then* $\mathsf{append}(l, r)$ *lies in the product history carrier.*

Proof. Use l and r as the two component witnesses. The BEDC continuation judgment is witnessed by reflexivity of the defining append result, so the product history carrier is rebuilt from the two source proofs and that continuation witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier append intro`

Theorem 34.4 (Product history carrier continuation introduction)**.** *If l lies in the left source carrier, r lies in the right source carrier, and* $\mathsf{Cont}(l, r, h)$*, then h lies in the product history carrier.*

Proof. Use l and r as the component witnesses. The supplied continuation proof is exactly the result witness required by the product history carrier. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier cont intro`

Theorem 34.5 (Product empty-result component exposure)**.** *If* $\mathsf{ProdHistoryCarrier}(L, R, \mathsf{Empty})$ *holds, then there are component histories* l *and* r *with* $L(l)$ *and* $R(r)$ *such that* $l\mathsf{hsameEmpty}$ *and* $r\mathsf{hsameEmpty}$*.*

Proof. Unpack the product carrier into component histories l and r, their source witnesses, and the continuation proof $\mathsf{Cont}(l, r, \mathsf{Empty})$. BEDC empty-result inversion for continuations forces both component histories to be the empty history, giving the two displayed hsame facts. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier empty result components`

Theorem 34.6 (Product visible-result component cases)**.** *If* $\mathsf{ProdHistoryCarrier}(L, R, \mathsf{E0}(tail))$ *holds, then its right component is either* e *and the left component is* hsame *to the visible result, or the right component is itself* E0*-headed and the stored continuation exposes its tail. The same dichotomy holds for visible result* $\mathsf{E1}(tail)$ *with an* E1*-headed right component.*

Proof. Unpack the product carrier into component histories l and r, source witnesses $L(l)$ and $R(r)$, and the continuation proof $\mathsf{Cont}(l, r, \mathsf{E0}(tail))$ or $\mathsf{Cont}(l, r, \mathsf{E1}(tail))$. Apply the corresponding BEDC continuation visible-result inversion theorem to the stored continuation. It yields exactly the empty-right case or the matching visible-headed right component case. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier visible result cases`

Theorem 34.7 (Product unary history closure)**.** *If a product history is generated from unary left and right histories, then the resulting product history is unary.*

Proof. Invert the product-history carrier into its left history, right history, unary component witnesses, and continuation proof. Unary closure under continuation turns the two unary component witnesses and the stored continuation proof into unary membership of the result history. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier unary`

Theorem 34.8 (Product history carrier transport)**.** *If two product histories are internally the same and the first is generated from a left witness, a right witness, and a continuation proof, then the second is generated from the same component witnesses and the transported continuation proof.*

Proof. Invert the product-history carrier into its left history, right history, component witnesses, and continuation proof. Transport the continuation result across hsame, and rebuild the product-history carrier with the same component witnesses. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier hsame transport`

Theorem 34.9 (Product history carrier unary from components)**.** *If the left and right source predicates each determine unary histories, then every product-history carrier generated from those predicates is unary.*

Proof. Unpack the product-history carrier into component histories l and r, their source witnesses, and the BEDC continuation witness for the result. The component assumptions give unary witnesses for l and r, and BEDC unary continuation closure gives unary membership of the product result. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier unary of components`

Definition 34.10 (Prod pattern specification)**.** *The pattern is paired generation from one A-witness and one B-witness.*

Definition 34.11 (Prod classifier specification)**.** *The typed classifier* ProdClassifierSpec *relates pairs componentwise using the classifiers of A and B. The typed wrapper* ProdClassifier *has the same componentwise content. Separately, the checked history-level scope is* $\mathsf{ProdHistoryClassifier}(L, R)$*: it relates histories h and k exactly when h lies in* $\mathsf{ProdHistoryCarrier}(L, R)$*, k lies in* $\mathsf{ProdHistoryCarrier}(L, R)$*, and* $\mathsf{hsame}(h, k)$ *holds.*

Lean4 (definition) `BEDC.Derived.ProdUp.ProdClassifierSpec`
Lean4 (definition) `BEDC.Derived.ProdUp.ProdHistoryClassifier`

Theorem 34.12 (Product typed classifier coordinate exactness)**.** *For types A and B, component classifiers $R_A : A \to A \to \mathsf{Prop}$ and $R_B : B \to B \to \mathsf{Prop}$, and elements $a, a' : A$ and $b, b' : B$, the product typed classifier is exact on coordinates:*

$$\mathsf{ProdClassifierSpec}(R_A, R_B, \mathsf{pair}(a, b), \mathsf{pair}(a', b')) \iff R_A(a, a') \wedge R_B(b, b').$$

Proof. Unfold ProdClassifierSpec. The forward direction is constructor inversion on the product classifier witness, yielding the two coordinate witnesses. The reverse direction pairs the two displayed coordinate witnesses to rebuild the product classifier witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.prodClassifier iff`

Definition 34.13 (Product typed source carrier)**.** *For source carriers $L, R : \mathsf{BHist} \to \mathsf{Prop}$ and $x : \mathsf{ProdCarrier}(\mathsf{BHist}, \mathsf{BHist})$, define* $\mathsf{ProdTypedCarrier}_{L,R}(x)$ *by the displayed pair decomposition*

$$\mathsf{ProdTypedCarrier}_{L,R}(x) \iff \exists l, r : \mathsf{BHist}.\ x = \mathsf{pair}(l, r) \wedge L(l) \wedge R(r).$$

It is the typed source-carrier predicate for the explicit ordered-product interface: the left coordinate is licensed by L and the right coordinate is licensed by R.

Lean4 (definition) `BEDC.Derived.ProdUp.ProdTypedCarrier`

Theorem 34.14 (Product typed carrier transport from source certificates)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\sim_L, \sim_R$ be source classifiers. If* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$*,* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, $x, y : \mathsf{ProdCarrier}(\mathsf{BHist}, \mathsf{BHist})$,* $\mathsf{ProdTypedCarrier}_{L,R}(x)$*, and* $\mathsf{ProdClassifierSpec}(\sim_L, \sim_R, x, y)$*, then* $\mathsf{ProdTypedCarrier}_{L,R}(y)$*.*

Proof. Invert the typed carrier witness for x to obtain coordinates l and r with $x = \mathsf{pair}(l, r)$, $L(l)$, and $R(r)$. Invert the product carrier y as $\mathsf{pair}(l', r')$. Apply Theorem 34.12 to the classifier witness for the two displayed pairs; this yields source classifier witnesses $l \sim_L l'$ and $r \sim_R r'$. The carrier_respects_equiv fields of cert_L and cert_R transport $L(l)$ to $L(l')$ and $R(r)$ to $R(r')$. Those transported witnesses rebuild $\mathsf{ProdTypedCarrier}_{L,R}(y)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdTypedCarrier transport from source certificates`

Theorem 34.15 (Product history classifier endpoint carrier exposure)**.** *For all source predicates $L, R : \mathsf{BHist} \to \mathsf{Prop}$ and histories $h, k : \mathsf{BHist}$, the following equivalence holds:*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \iff \exists l, r, l', r' : \mathsf{BHist},\ L(l) \wedge R(r) \wedge \mathsf{Cont}(l, r, h) \wedge L(l') \wedge R(r') \wedge \mathsf{Cont}(l', r', k) \wedge \mathsf{hsame}($$

Componentwise equality and componentwise classification belong to ProdClassifierSpec*, not to the history-envelope classifier.*

Proof. For the forward direction of the displayed equivalence, unfold ProdHistoryClassifier. Its two endpoint premises are $\mathsf{ProdHistoryCarrier}(L, R, h)$ and $\mathsf{ProdHistoryCarrier}(L, R, k)$, and each carrier premise unfolds into one left witness, one right witness, the two source-membership proofs, and the continuation proof producing the visible endpoint. Pair those two carrier inversions with the stored $\mathsf{hsame}(h, k)$ witness for the forward direction. Conversely, the displayed data rebuild the two product-history carrier premises and then the classifier witness for the reverse direction of the equivalence. The result is endpoint exposure rather than coordinate projection. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier component inversion`

Theorem 34.16 (Product history classifier common-right component left equality)**.** *For source predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* $h, k, l, l', r : \mathsf{BHist}$*, if*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k), \qquad \mathsf{Cont}(l, r, h), \qquad \mathsf{Cont}(l', r, k),$$

then $\mathsf{hsame}(l, l')$*.*

Proof. The product-history classifier stores the endpoint equality $\mathsf{hsame}(h, k)$. Both displayed continuation witnesses use the same right component r. BEDC common-suffix cancellation for continuations therefore cancels the shared right component across the equal endpoints and yields $\mathsf{hsame}(l, l')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier common right component left hsame`

Theorem 34.17 (Product history source monotonicity)**.** *The roadmap-template fit is: traditional target ordered-product stability under source refinement; BEDC source* BHist *with* Cont*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; stability obligation source inclusion transport; ledger unchanged pair-continuation evidence; core theorem list carrier monotonicity plus classifier monotonicity; exactness predicate preservation of visible product endpoints under source refinement; standard bridge ordinary product source restriction; status* ProofTarget*.*

For predicates $L, R, L', R' : \mathsf{BHist} \to \mathsf{Prop}$*, suppose* $\forall h.\ L(h) \to L'(h)$ *and* $\forall h.\ R(h) \to R'(h)$*. Then every* $\mathsf{ProdHistoryCarrier}(L, R, h)$ *yields* $\mathsf{ProdHistoryCarrier}(L', R', h)$*. Under the same hypotheses, every* $\mathsf{ProdHistoryClassifier}(L, R, h, k)$ *yields* $\mathsf{ProdHistoryClassifier}(L', R', h, k)$*.*

Proof. For the carrier clause, unpack the product carrier into histories l and r, source witnesses $L(l)$ and $R(r)$, and the continuation witness $\mathsf{Cont}(l, r, h)$. Apply the two source inclusions to obtain $L'(l)$ and $R'(r)$, then rebuild $\mathsf{ProdHistoryCarrier}(L', R', h)$ with the same continuation evidence.

For the classifier clause, unfold the stored classifier data. Apply the carrier clause to both endpoint carriers, and keep the stored $\mathsf{hsame}(h, k)$ witness unchanged to rebuild $\mathsf{ProdHistoryClassifier}(L', R', h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistorySource monotonicity`

Theorem 34.18 (Product history classifier right transport)**.** *If a product-history classifier relates* h *to* k*, and* k *is internally the same as* k'*, then the classifier relates* h *to* k'*.*

Proof. Invert the product-history classifier into the left carrier, the right carrier, and the stored $\mathsf{hsame}(h, k)$ witness. Transport the right carrier across $\mathsf{hsame}(k, k')$, then compose $\mathsf{hsame}(h, k)$ with $\mathsf{hsame}(k, k')$ for the classifier witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier right hsame transport`

Theorem 34.19 (Product history classifier endpoint transport)**.** *If a product-history classifier relates h to k, and h is internally the same as h' while k is internally the same as k', then the classifier relates h' to k'.*

Proof. Invert the product-history classifier into endpoint carriers and the stored $\mathsf{hsame}(h, k)$ witness. Transport both endpoint carriers across their respective hsame witnesses, then compose symmetry of $\mathsf{hsame}(h, h')$, the stored link, and $\mathsf{hsame}(k, k')$ to obtain $\mathsf{hsame}(h', k')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier hsame transport`

Theorem 34.20 (Product history classifier symmetry)**.** *If a product-history classifier relates h to k, then it relates k to h.*

Proof. Unpack the classifier into the two endpoint product carriers and the stored $\mathsf{hsame}(h, k)$ link. Exchange the endpoint carriers and reverse the stored history-sameness link by symmetry. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier symm`

Theorem 34.21 (Product history classifier left transport)**.** *If a product-history classifier relates h to k, and h is internally the same as h', then the classifier relates h' to k.*

Proof. Invert the classifier witness. Transport the left carrier through the $\mathsf{hsame}(h, h')$ witness, keep the right carrier, and compose the symmetric $\mathsf{hsame}(h', h)$ link with the stored $\mathsf{hsame}(h, k)$ classifier witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier left hsame transport`

Theorem 34.22 (Product history classifier transitivity)**.** *For source-history predicates L and R on* BHist*, if* $\mathsf{ProdHistoryClassifier}(L, R, h, k)$ *and* $\mathsf{ProdHistoryClassifier}(L, R, k, u)$*, then* $\mathsf{ProdHistoryClassifier}(L, R, h, u)$*.*

Proof. Unpack the endpoint $\mathsf{ProdHistoryCarrier}(L, R)$ witnesses for h and k from the first classifier witness, and for k and u from the second. Keep the h endpoint carrier from the first witness and the u endpoint carrier from the second, compose the stored $\mathsf{hsame}(h, k)$ and $\mathsf{hsame}(k, u)$ links, and rebuild $\mathsf{ProdHistoryClassifier}(L, R, h, u)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier trans`

Theorem 34.23 (Product history classifier continuation congruence)**.** *Let L and R be source-history predicates on* BHist*. Assume the left and right transport fields*

$$\forall x\, y.\ L(x) \wedge \mathsf{hsame}(x, y) \to L(y)$$

and

$$\forall x\, y.\ R(x) \wedge \mathsf{hsame}(x, y) \to R(y).$$

If $L(l)$*,* $R(r)$*,* $\mathsf{hsame}(l, l')$*,* $\mathsf{hsame}(r, r')$*,* $\mathsf{Cont}(l, r, h)$*, and* $\mathsf{Cont}(l', r', h')$*, then* $\mathsf{ProdHistoryClassifier}(L, R, h, h')$*.*

Proof. Apply the displayed left transport field to $L(l)$ and $\mathsf{hsame}(l, l')$ to obtain $L(l')$, and apply the displayed right transport field to $R(r)$ and $\mathsf{hsame}(r, r')$ to obtain $R(r')$. Product history carrier continuation introduction builds endpoint carrier witnesses for h and h' from the two continuation witnesses. Continuation respect for hsame transports the result endpoints across the two component links, yielding $\mathsf{hsame}(h, h')$. These three pieces are exactly the history-envelope classifier fields. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier cont congr`

Definition 34.24 (Product pair representation)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *to mean that* l *is a left source history,* r *is a right source history, and their continuation presents* h*:*

$$\mathsf{ProdPairRep}_{L,R}(h, l, r) \quad :\Longleftrightarrow \quad L(l) \wedge R(r) \wedge \mathsf{Cont}(l, r, h).$$

Lean4 (definition) `BEDC.Derived.ProdUp.ProdPairRep`

Theorem 34.25 (Product pair representation coverage)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*,*

$$\mathsf{ProdHistoryCarrier}(L, R, h) \quad \Longleftrightarrow \quad \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r).$$

Proof. Unfold ProdHistoryCarrier and ProdPairRep. In the forward direction, the carrier witness exposes component histories l and r, source witnesses $L(l)$ and $R(r)$, and the continuation witness $\mathsf{Cont}(l, r, h)$; these are exactly the displayed pair representation witnesses. In the reverse direction, a displayed pair representation supplies the same two component histories, source witnesses, and continuation witness, so it rebuilds the product history carrier witness for h. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep coverage`

Theorem 34.26 (Product pair representation endpoint transport)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k, l, r*, if* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{ProdPairRep}_{L,R}(k, l, r)$ *holds.*

Proof. The per-object fit is: traditional target ordered-pair endpoint stability; BEDC source BHist with Cont; carrier ProdPairRep; classifier endpoint hsame; stability obligation endpoint transport; ledger displayed pair representation; core theorem this transport law; exactness pair presentation preserved by endpoint sameness; standard bridge stable ordered-pair endpoint; status ProofTarget. Unfold ProdPairRep to keep $L(l)$ and $R(r)$, and transport the $\mathsf{Cont}(l, r, h)$ result across $\mathsf{hsame}(h, k)$ using continuation result transport. The transported continuation witness together with the unchanged source witnesses rebuilds $\mathsf{ProdPairRep}_{L,R}(k, l, r)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep hsame transport`

Theorem 34.27 (Product fixed-pair endpoint determinism)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k, l, r*, if* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *and* $\mathsf{ProdPairRep}_{L,R}(k, l, r)$*, then* $\mathsf{hsame}(h, k)$ *holds, and consequently* $\mathsf{ProdHistoryClassifier}(L, R, h, k)$ *holds.*

Proof. Unfold both displayed pair representations. The first supplies $L(l)$, $R(r)$, and $\mathsf{Cont}(l, r, h)$; the second supplies the same component source witnesses and $\mathsf{Cont}(l, r, k)$. Applying continuation determinism to the two continuation witnesses gives $\mathsf{hsame}(h, k)$. The same source witnesses and continuation witnesses rebuild $\mathsf{ProdHistoryCarrier}(L, R, h)$ and $\mathsf{ProdHistoryCarrier}(L, R, k)$, and these endpoint carriers together with the hsame witness rebuild $\mathsf{ProdHistoryClassifier}(L, R, h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier fixed pair determinism`

This statement fixes both component histories; comparison of two different pair presentations of one endpoint is exactly the separate coherence datum in Definition 34.29.

Theorem 34.28 (Product fixed-pair endpoint exactness)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k, l, r*, if* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *holds, then*

$$\mathsf{ProdPairRep}_{L,R}(k, l, r) \iff \mathsf{hsame}(h, k).$$

Proof. The per-object fit is: traditional target ordered-pair endpoint readback; BEDC source BHist with Cont; carrier ProdPairRep; classifier endpoint hsame; stability by endpoint transport and continuation determinism; ledger fixed displayed pair; core theorem fixed-pair endpoint exactness; exactness predicate the displayed iff; standard bridge fixed ordered-pair endpoint. For the forward direction, apply Theorem 34.27 to the displayed base representation and the target representation, then project the $\mathsf{hsame}(h, k)$ conjunct. For the reverse direction, apply Theorem 34.26 to the base representation and the displayed endpoint sameness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep fixed endpoint exactness`

Definition 34.29 (Product pair-representation coherence)**.** *For source classifiers* $\sim_L$ *and* $\sim_R$*, define* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *to mean that any two pair representations of the same product history classify their corresponding components:*

$$\forall h\, l\, r\, l'\, r'.\ \mathsf{ProdPairRep}_{L,R}(h, l, r) \wedge \mathsf{ProdPairRep}_{L,R}(h, l', r') \to (l \sim_L l') \wedge (r \sim_R r').$$

Theorem 34.30 (Product raw pair representation coherence is not automatic)**.** *Let* $\mathsf{TopCarrier}(h) := \mathsf{True}$ *for every* $h : \mathsf{BHist}$*, and take both source classifiers to be* hsame*. Then* $\neg\mathsf{ProdPairRepCoherent}_{\mathsf{TopCarrier},\mathsf{TopCarrier},\mathsf{h}}$ *holds.*

The per-object fit is: traditional target ordered-pair readback; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdPairRep*; classifier* hsame*; stability obligation coherence; ledger displayed pair boundary; core theorem non-automaticity; exactness obstruction; standard bridge raw append ambiguity; status* ProofTarget*.*

Proof. The endpoint $\mathsf{E0}(\mathsf{e})$ has two raw pair presentations under TopCarrier. The left-unit presentation uses components $(\mathsf{e}, \mathsf{E0}(\mathsf{e}))$, and the right-unit presentation uses components $(\mathsf{E0}(\mathsf{e}), \mathsf{e})$. If product pair-representation coherence were automatic for the two hsame classifiers, applying it to these two presentations of the same endpoint would force $\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(\mathsf{e}))$ on the left components, and also $\mathsf{hsame}(\mathsf{E0}(\mathsf{e}), \mathsf{e})$ on the right components. BEDC history no-confusion separates the empty history from a zero-step endpoint, so the forced $\mathsf{hsame}(\mathsf{e}, \mathsf{E0}(\mathsf{e}))$ is impossible. Thus the componentwise ordered-pair certificate needs displayed coherence data and cannot be recovered from raw continuation alone. □

Definition 34.31 (Product componentwise history classifier)**.** *For source-history carriers* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifiers* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim}$ *by the existence of component histories* l_h, r_h, l_k, r_k *such that*

$$\mathsf{ProdPairRep}_{L,R}(h, l_h, r_h) \wedge \mathsf{ProdPairRep}_{L,R}(k, l_k, r_k) \wedge l_h \sim_L l_k \wedge r_h \sim_R r_k.$$

Lean4 (definition) `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier`

Theorem 34.32 (Product componentwise history classifier source refinement)**.** *The per-object fit is: traditional ordered-product classifier stability under source refinement; BEDC source* BHist *with* Cont*; carrier through* ProdPairRep*; classifier* ProdComponentHistoryClassifier*; ledger unchanged displayed pair representations; one-way classifier preservation under carrier and classifier inclusions; status* ProofTarget*.*

For carriers $L, R, L', R' : \mathsf{BHist} \to \mathsf{Prop}$*, classifiers* $\sim_L, \sim_R, \sim'_L, \sim'_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* $h, k : \mathsf{BHist}$*, suppose* $\forall u.\ L(u) \to L'(u)$*,* $\forall u.\ R(u) \to R'(u)$*,* $\forall x\, y.\ x \sim_L y \to x \sim'_L y$*, and* $\forall x\, y.\ x \sim_R y \to x \sim'_R y$*. Then* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k)$ *implies* $\mathsf{ProdComponentHistoryClassifier}_{L',R',\sim'_L,\sim'_R}(h,k)$*.*

Proof. Unpack the componentwise classifier witness into displayed pair representations of h and k with components l_h, r_h, l_k, r_k, together with the two component classifier witnesses. Rebuild each displayed pair representation for L', R' by applying the carrier inclusions to the stored component source witnesses and keeping the same continuation evidence. Transport the two component classifier witnesses through the classifier implications, then repack these two rebuilt displayed pair representations and the transported component witnesses as a $\mathsf{ProdComponentHistoryClassifier}_{L',R',\sim'_L,\sim'_R}(h,k)$ witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier source refinement`

Theorem 34.33 (Product componentwise history classifier endpoint transport)**.** *For source-history carriers* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, source classifiers* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k, h', k'*, if* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k)$*,* $\mathsf{hsame}(h,h')$*, and* $\mathsf{hsame}(k,k')$ *hold, then* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h',k')$ *holds.*

Proof. The per-object fit is: traditional target ordered-pair classifier endpoint stability; BEDC source BHist with Cont and hsame; carrier ProdHistoryCarrier through ProdPairRep; classifier ProdComponentHistory stability obligation endpoint transport; ledger displayed pair representations; core theorem this transport law; exactness classifier preservation under endpoint hsame; standard bridge stable ordered-pair endpoints; status ProofTarget. Unpack the componentwise classifier witness into pair representations for h and k and component classifier witnesses $l_h \sim_L l_k$ and $r_h \sim_R r_k$. Apply Theorem 34.26 to move the displayed representation of h along $\mathsf{hsame}(h,h')$, and apply the same theorem to move the displayed representation of k along $\mathsf{hsame}(k,k')$. The component classifier witnesses are unchanged, so the transported pair representations and the original component witnesses rebuild $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h',k')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier hsame transport`

The componentwise endpoint classifier uses the finite-data sibling classifiers only as classifier patterns: the Bool tag discipline of chapter 32, the Sum endpoint separation of chapter 35, and the List spine stability of chapter 36. The $\mathsf{Prod}^{\uparrow}$ packet pairs two histories componentwise, while those siblings supply the finite tag and constructor-classifier shapes that the product route must not collapse.

Theorem 34.34 (Product pair representation hsame coherence)**.** *For source-history carriers* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, source classifiers* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k, l, r, l', r'*, if* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*,* $\mathsf{ProdPairRep}_{L,R}(h,l,r)$*,* $\mathsf{ProdPairRep}_{L,R}(k,l',r')$*, and* $\mathsf{hsame}(h,k)$ *hold, then* $(l \sim_L l') \wedge (r \sim_R r')$ *holds.*

The per-object fit is: traditional target ordered-pair component comparison; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdPairRep*; classifier pair* $\sim_L, \sim_R$*; stability obligation endpoint-transported coherence; ledger displayed pair representations at internally same product endpoints; core theorem this hsame-coherence bridge; exactness component witnesses compared after endpoint transport; standard bridge ordered-pair readback over stable product endpoints; status* ProofTarget*.*

Proof. Transport the second displayed pair representation from k to h. Concretely, symmetry of $\mathsf{hsame}(h,k)$ gives $\mathsf{hsame}(k,h)$, and Theorem 34.26 sends $\mathsf{ProdPairRep}_{L,R}(k,l',r')$ to $\mathsf{ProdPairRep}_{L,R}(h,l',r')$

without changing the displayed components. Now apply $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ at the single endpoint h to the two displayed representations $\mathsf{ProdPairRep}_{L,R}(h,l,r)$ and $\mathsf{ProdPairRep}_{L,R}(h,l',r')$. The coherence datum returns exactly $(l \sim_L l') \wedge (r \sim_R r')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep hsame coherence`

Theorem 34.35 (Product componentwise history classifier endpoint carriers)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers and let $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be source classifiers. If $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k)$ holds, then both $\mathsf{ProdHistoryCarrier}(L,R,h)$ and $\mathsf{ProdHistoryCarrier}(L,R,k)$ hold. The per-object template role is: traditional target ordered-pair classifier endpoint coverage; BEDC source* BHist *with* Cont*; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; ledger displayed pair representation; core theorem endpoint carrier exposure; exactness classifier endpoints are generated product endpoints; status* ProofTarget*.*

Proof. Unpack the componentwise classifier witness to obtain component histories l_h, r_h, l_k, r_k and displayed pair representations $\mathsf{ProdPairRep}_{L,R}(h,l_h,r_h)$ and $\mathsf{ProdPairRep}_{L,R}(k,l_k,r_k)$. Apply pair-representation coverage to the first displayed pair representation to rebuild $\mathsf{ProdHistoryCarrier}(L,R,h)$ and to the second displayed pair representation to rebuild $\mathsf{ProdHistoryCarrier}(L,R,k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier endpoint carriers`

Theorem 34.36 (Product componentwise classifier empty-source exclusion)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers and let $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be source classifiers. If*

$$(\forall l : \mathsf{BHist}.\ \neg L(l)) \vee (\forall r : \mathsf{BHist}.\ \neg R(r)),$$

then for all histories $h, k : \mathsf{BHist}$,

$$\neg\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k).$$

The ten-field fit is: traditional ordered-product impossibility boundary; BEDC source BHist *with* Cont*; carrier through* ProdPairRep*; classifier* ProdComponentHistoryClassifier*; ledger displayed pair representation; core theorem empty-source exclusion; exactness no public product classification without both source witnesses; status* ProofTarget*.*

Proof. Assume the displayed empty-source disjunction and fix histories h, k together with a componentwise classifier witness. Invert that witness to obtain component histories l_h, r_h, l_k, r_k and displayed pair representations for both endpoints.

The first representation $\mathsf{ProdPairRep}_{L,R}(h,l_h,r_h)$ exposes the source witnesses $L(l_h)$ and $R(r_h)$ carried by the displayed product endpoint. If the left source is empty, applying $\forall l.\ \neg L(l)$ to l_h contradicts $L(l_h)$.

If the right source is empty instead, applying $\forall r.\ \neg R(r)$ to r_h contradicts $R(r_h)$. Thus either side of the empty-source disjunction rules out the assumed public componentwise classification. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier empty source exclusion`

Theorem 34.37 (Product history classifier hsame component cancellation)**.** *For source predicates $L, R : \mathsf{BHist} \to \mathsf{Prop}$ and histories $h, k, l, l', r, r' : \mathsf{BHist}$, suppose*

$$\mathsf{ProdHistoryClassifier}(L,R,h,k), \qquad \mathsf{Cont}(l,r,h), \qquad \mathsf{Cont}(l',r',k).$$

Then the displayed component comparisons determine each other:

$$(\mathsf{hsame}(l, l') \to \mathsf{hsame}(r, r')) \wedge (\mathsf{hsame}(r, r') \to \mathsf{hsame}(l, l')).$$

The ten-field fit is: traditional target ordered-pair component cancellation; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; stability from endpoint* hsame *plus continuation cancellation; ledger displayed pair continuations; core theorem this component-cancellation block; exactness component readback under one known component comparison; standard bridge ordered-pair cancellation; status* ProofTarget*.*

Proof. Unfold the product-history classifier only to expose the stored endpoint comparison $\mathsf{hsame}(h, k)$. The two displayed continuation witnesses already show that h is presented by (l, r) and k is presented by (l', r').

For the first implication, assume $\mathsf{hsame}(l, l')$. Use Definition 4.43 on the two displayed continuations, with the left contexts aligned by $\mathsf{hsame}(l, l')$ and the results aligned by the classifier endpoint comparison $\mathsf{hsame}(h, k)$. The left-context cancellation clause yields $\mathsf{hsame}(r, r')$.

For the second implication, assume $\mathsf{hsame}(r, r')$. Transport the right suffix of one displayed continuation along this comparison so the two continuations have a common suffix. Then apply the common-suffix cancellation clause recorded in Definition 4.52, using the same endpoint comparison $\mathsf{hsame}(h, k)$, to obtain $\mathsf{hsame}(l, l')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier component hsame cancellation`

Theorem 34.38 (Product history classifier empty-left component inversion)**.** *For source predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *and a history* $k : \mathsf{BHist}$*, if* $\mathsf{ProdHistoryClassifier}(L, R, \mathsf{Empty}, k)$ *holds, then there are component histories* l, r, l', r' *such that* $L(l)$*,* $R(r)$*,* $l\mathsf{hsameEmpty}$*,* $r\mathsf{hsameEmpty}$*,* $L(l')$*,* $R(r')$*,* $\mathsf{Cont}(l', r', k)$*, and* $\mathsf{Emptyhsame}k$ *all hold.*

Proof. Invert the product-history classifier into displayed product carrier witnesses for the empty left endpoint and for k. The empty endpoint carrier supplies a continuation $\mathsf{Cont}(l, r, \mathsf{Empty})$, and empty-result inversion for BEDC continuations forces both displayed components to be Empty. Keep the right endpoint continuation witness and the stored classifier endpoint equality to obtain the stated exposure. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier empty left component inversion`

Theorem 34.39 (Product pair-representation coherence obstruction from separated presentations)**.** *The ten-field fit is: traditional target ordered-pair presentation comparison; BEDC source* BHist *with* Cont*; carrier* ProdPairRep*; classifier pair* $\sim_L, \sim_R$*; stability obligation coherence; ledger displayed pair boundary; core theorem separated-presentation obstruction; exactness as failure of component comparison; standard bridge ordered-pair readback; status* ProofTarget*.*

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and let* $h, l, r, l', r' : \mathsf{BHist}$*. Suppose* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *and* $\mathsf{ProdPairRep}_{L,R}(h, l', r')$ *hold. If either* $\neg(l \sim_L l')$ *or* $\neg(r \sim_R r')$ *holds, then*

$$\neg\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$$

holds.

Proof. Assume a coherence witness. Apply it to the two displayed representations of the same product history h. The coherence witness returns both component comparisons, namely $l \sim_L l'$ and $r \sim_R r'$. The separated component clause supplies either the negation of the left comparison or the negation of the right comparison, contradicting the corresponding returned component witness. □

Theorem 34.40 (Product pair coherence consumes finite tag separation)**.** *Before a* $\mathsf{Prod}^{\uparrow}$ *pair-representation coherence witness may publish componentwise classifier fields, it must preserve the same finite tag-separation discipline used by the nullable* $\mathsf{Option}^{\uparrow}$ *surface of chapter 33 and the disjoint-branch classifier surface of chapter 35. Concretely, an absent-or-present payload read cannot be folded into a component comparison unless the* $\mathsf{Option}^{\uparrow}$ *tag row has already separated the nullable case, and a left-or-right branch read cannot be paired across components unless the* $\mathsf{Sum}^{\uparrow}$ *classifier has already excluded mixed tags. The product row therefore receives componentwise classifier fields only after tag-separated finite constructors have fixed which payload comparisons are admissible.*

Proof. Start with the obstruction theorem above. A product coherence witness returns left and right component comparisons only after two displayed pair presentations have been matched at the same product history.

The $\mathsf{Option}^{\uparrow}$ dependency supplies the nullable tag boundary. Its absent and present rows determine whether a payload comparison is available at all, so a product component cannot treat the absent case as an ordinary source payload.

The $\mathsf{Sum}^{\uparrow}$ dependency supplies disjoint-branch separation. Its mixed-tag exclusion prevents a left payload and a right payload from being read as one component classifier row.

Thus the product classifier fields are published only after the two finite constructor disciplines have determined admissible component comparisons. The product packet adds no tag-erasing route of its own. □

Theorem 34.41 (Product componentwise history classifier carrier reflexivity from source certificates)**.** *The ten-field fit is: traditional target ordered-pair classifier reflexivity; BEDC source* BHist *with* Cont *and displayed pair representations; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; stability obligation carrier-relative reflexivity from the two source certificates; ledger one displayed product pair representation; exactness component reflexivity at the same endpoints; standard bridge diagonal ordered-pair comparison; status* ProofTarget*.*

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*. Then for every* $h : \mathsf{BHist}$*,*

$$\mathsf{ProdHistoryCarrier}(L, R, h) \to \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, h).$$

Proof. Unpack the product carrier witness into displayed source histories l and r, source carrier witnesses for $L(l)$ and $R(r)$, and the corresponding pair representation of h. Use that same pair representation for both endpoints of the componentwise classifier. The source certificates provide $l \sim_L l$ and $r \sim_R r$ through their reflexivity fields, and these two component witnesses rebuild $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, h)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier carrier reflexivity from source certificates`

Theorem 34.42 (Product componentwise history classifier symmetry from source certificates)**.** *The ten-field fit is: traditional target ordered-pair classifier symmetry; BEDC source* BHist *with* Cont *and displayed pair representations; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; stability obligation source-certified symmetry for the two component classifiers; ledger displayed pair representations at both endpoints; exactness reversed component comparison; standard bridge symmetric ordered-pair comparison; status* ProofTarget*.*

Let L, R : BHist $\to$ Prop *be source-history carriers, let* $\sim_L, \sim_R$: BHist $\to$ BHist $\to$ Prop *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*. Then for all* h, k : BHist*,*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \to \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(k, h).$$

Proof. Unpack the componentwise classifier witness into displayed pair representations of h and k and component classifier witnesses $l_h \sim_L l_k$ and $r_h \sim_R r_k$. Apply cert_L.equiv_symm and cert_R.equiv_symm to reverse the two component comparisons. Swap the endpoint pair representations and rebuild the componentwise classifier from the reversed left and right witnesses. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier symm from source certificates`

Theorem 34.43 (Product componentwise classifier self-readback exactness)**.** *The ten-field fit is: traditional target diagonal ordered-pair readback; BEDC source* BHist *with* Cont *and displayed pair representations; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; stability obligation endpoint-carrier exposure together with carrier-relative reflexivity from the two source certificates; ledger one visible history used at both endpoints; core theorem self-readback exactness; exactness diagonal classifier truth if and only if product carrier membership; standard bridge ordered-pair self comparison; status* ProofTarget*.*

Let L, R : BHist $\to$ Prop *be source-history carriers, let* $\sim_L, \sim_R$: BHist $\to$ BHist $\to$ Prop *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*. Then every* h : BHist *satisfies*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, h) \iff \mathsf{ProdHistoryCarrier}(L, R, h).$$

Proof. For the forward direction, assume $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, h)$. Apply Theorem 34.35 to this diagonal classifier witness. It returns product-carrier witnesses for both endpoints, and either projection gives $\mathsf{ProdHistoryCarrier}(L, R, h)$.

For the reverse direction, assume $\mathsf{ProdHistoryCarrier}(L, R, h)$. Apply Theorem 34.41 with the two source certificates to the carrier witness. The result is exactly $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, h)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier self readback iff`

Theorem 34.44 (Product componentwise classifier is independent of displayed pair representatives)**.** *The ten-field fit is: traditional target ordered-product componentwise readback well-definedness; BEDC source* BHist *with* Cont *and displayed* ProdPairRep *witnesses; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; stability obligation representative independence under* ProdPairRepCoherent *and source* NameCert *symmetry and transitivity; ledger displayed pair representations at both endpoints; core theorem this law; exactness independence from chosen coherent representatives; standard bridge ordered-pair readback; status* ProofTarget*.*

Let L, R : BHist $\to$ Prop *be source-history carriers, let* $\sim_L, \sim_R$: BHist $\to$ BHist $\to$ Prop *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$*,* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*.*

For all $h, k, l, r, l', r', u, v, u', v' : \mathsf{BHist}$*, if* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$*,* $\mathsf{ProdPairRep}_{L,R}(h, l', r')$*,* $\mathsf{ProdPairRep}_{L,R}(k, u, v)$*, and* $\mathsf{ProdPairRep}_{L,R}(k, u', v')$ *hold, then*

$$(l \sim_L u) \wedge (r \sim_R v) \to (l' \sim_L u') \wedge (r' \sim_R v').$$

Proof. Apply $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ at the source endpoint h to compare the displayed pairs (l', r') and (l, r). This gives $l' \sim_L l$ and $r' \sim_R r$. Apply the same coherence law at the target endpoint k to compare (u, v) with (u', v'), giving $u \sim_L u'$ and $v \sim_R v'$. The supplied component comparison gives $l \sim_L u$ and $r \sim_R v$. Two uses of cert_L.equiv_trans compose $l' \sim_L l \sim_L u \sim_L u'$, and two uses of cert_R.equiv_trans compose $r' \sim_R r \sim_R v \sim_R v'$. The resulting left and right witnesses form the displayed pair of component comparisons. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier representative independence`

Theorem 34.45 (Product componentwise history classifier core fields)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$*,* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. Write* $C(h, k)$ *for* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$*. Then its componentwise endpoint exposure is:*

$$\forall h\, k.\ C(h, k) \to \mathsf{ProdHistoryCarrier}(L, R, h) \wedge \mathsf{ProdHistoryCarrier}(L, R, k).$$

Its five core NameCert *fields for* $\mathsf{ProdHistoryCarrier}(L, R)$ *decompose as follows:*

1. *carrier inhabitation:*
$$\exists h.\ \mathsf{ProdHistoryCarrier}(L, R, h);$$
2. *carrier-relative reflexivity:*
$$\forall h.\ \mathsf{ProdHistoryCarrier}(L, R, h) \to C(h, h);$$
3. *source-certified symmetry:*
$$\forall h\, k.\ C(h, k) \to C(k, h);$$
4. *source-certified transitivity:*
$$\forall h\, k\, u.\ C(h, k) \to C(k, u) \to C(h, u);$$
5. *projection-derived NameCert-shaped classifier-induced carrier transport:*
$$\forall h\, k.\ C(h, k) \to \mathsf{ProdHistoryCarrier}(L, R, h) \to \mathsf{ProdHistoryCarrier}(L, R, k).$$

These five fields are the core NameCert *data used by Theorem 34.61.*

Proof. For carrier inhabitation, cert_L.carrier_inhabited and cert_R.carrier_inhabited supply histories l and r with $L(l)$ and $R(r)$. Applying Theorem 34.3 to those two source carrier witnesses gives $\mathsf{ProdHistoryCarrier}(L, R, \mathsf{append}(l, r))$, so the product carrier is inhabited. For carrier-relative reflexivity, unpack a $\mathsf{ProdHistoryCarrier}(L, R, h)$ witness and apply Theorem 34.41. For symmetry, apply Theorem 34.42 to the componentwise classifier witness. For transitivity, unpack $\mathsf{C}(h, k)$ and $\mathsf{C}(k, u)$ into pair representations (l_h, r_h), (l_k, r_k), (l'_k, r'_k), and (l_u, r_u). Pair-representation coherence at the middle history k gives both $l_k \sim_L l'_k$ and $r_k \sim_R r'_k$. The two source NameCert.equiv_trans

fields then compose the left chain $l_h \sim_L l_k \sim_L l'_k \sim_L l_u$ and the right chain $r_h \sim_R r_k \sim_R r'_k \sim_R r_u$ separately. For endpoint exposure, apply Theorem 34.35 directly to the classifier witness $C(h, k)$. For carrier transport, apply that endpoint exposure to $C(h, k)$, take the right projection $\mathsf{ProdHistoryCarrier}(L, R, k)$, and ignore the already supplied left carrier premise except as the required $\mathsf{NameCert}$ field input. Thus the displayed endpoint clause and five-field tuple are exactly the componentwise product core classifier data. □

Theorem 34.46 (Product componentwise history classifier transitivity from pair coherence)**.** *The ten-field fit is: traditional target ordered-pair classifier composition; BEDC source* BHist *with* Cont *and displayed pair representations; carrier* $\mathsf{ProdHistoryCarrier}$*; classifier* $\mathsf{ProdComponentHistoryClassifier}$*; stability obligation source-certified transitivity together with middle pair-representation coherence; ledger displayed pair representations; exactness component comparison through the shared middle endpoint; standard bridge ordered-pair composition; status* $\mathsf{ProofTarget}$.

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$*,* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. Then for all* $h, k, u : \mathsf{BHist}$*,*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \to \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(k, u) \to \mathsf{ProdComponen}$$

Proof. Unpack the first componentwise classifier witness into displayed pair representations of h and k, and unpack the second into displayed pair representations of k and u. The two representations of the shared middle history k are compared by $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$, giving the middle left and right component comparisons. On the left side, $\mathsf{cert}_L.\mathsf{equiv_trans}$ first composes the h–k comparison with the middle left comparison, and then composes the result with the k–u comparison. On the right side, $\mathsf{cert}_R.\mathsf{equiv_trans}$ performs the same two-step composition through the middle right comparison. Reuse the original displayed pair representations of h and u together with these two composed component comparisons to rebuild $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, u)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier trans from pair coherence`

Finite-data sibling lattice. The componentwise $\mathsf{Prod}^{\uparrow}$ transfer reads the same finite-data sibling lattice as the nullable, tagged, and iterated carriers: chapter 33 supplies the one-sided absent-or-present component case, chapter 35 supplies tagged branch separation for component choice, and chapter 36 supplies iterated component transport when a product certificate is replayed across a finite family. The transfer theorems below use those sibling surfaces only as finite carrier discipline; they do not add any coordinate beyond the displayed product pair rows.

Theorem 34.47 (Bounded hsame product pair-representation coherence)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers. Suppose there are histories* $l_0, r_0 : \mathsf{BHist}$ *such that*

$$\forall l.\ L(l) \to \mathsf{hsame}(l, l_0)$$

and

$$\forall r.\ R(r) \to \mathsf{hsame}(r, r_0).$$

With both source classifiers specialized to hsame*, the product pair-representation coherence condition holds:*

$$\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}.$$

Proof. Take two displayed pair representations of the same product history h, with components (l, r) and (l', r'). The pair-representation witnesses include $L(l)$, $L(l')$, $R(r)$, and $R(r')$. The boundedness hypotheses give $\mathsf{hsame}(l, l_0)$, $\mathsf{hsame}(l', l_0)$, $\mathsf{hsame}(r, r_0)$, and $\mathsf{hsame}(r', r_0)$. By symmetry of hsame, turn $\mathsf{hsame}(l', l_0)$ into $\mathsf{hsame}(l_0, l')$ and $\mathsf{hsame}(r', r_0)$ into $\mathsf{hsame}(r_0, r')$. By transitivity of hsame, compose l through l_0 to obtain $\mathsf{hsame}(l, l')$, and compose r through r_0 to obtain $\mathsf{hsame}(r, r')$. These are exactly the two component classifier conclusions required by $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$. □

Theorem 34.48 (Product pair-representation coherence from classifier-bounded sources)**.** *The ten-field fit is: traditional target ordered-product component readback through bounded source classifiers; BEDC source* BHist *with* Cont*; carriers L and R; classifiers $\sim_L$ and $\sim_R$; stability obligation pair-representation coherence; ledger displayed pair representation; core theorem classifier-bounded coherence; exactness component comparison through the source classifiers; standard bridge source certificates plus center histories; status* ProofTarget*.*

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers, let $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be source classifiers, and assume $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ and $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Suppose there are histories $l_0, r_0 : \mathsf{BHist}$ such that

$$\forall l.\ L(l) \to l \sim_L l_0$$

and

$$\forall r.\ R(r) \to r \sim_R r_0.$$

Then the product pair-representation coherence condition holds:

$$\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}.$$

Proof. Take two displayed pair representations of the same product history h, with components (l, r) and (l', r'). The displayed witnesses contain the source facts $L(l)$, $L(l')$, $R(r)$, and $R(r')$. The boundedness assumptions give $l \sim_L l_0$, $l' \sim_L l_0$, $r \sim_R r_0$, and $r' \sim_R r_0$. Apply the symmetry fields of cert_L and cert_R to turn the second left and right bounds into $l_0 \sim_L l'$ and $r_0 \sim_R r'$. Apply the transitivity fields of the two certificates to compose through the center histories, obtaining $l \sim_L l'$ and $r \sim_R r'$. These two component classifier witnesses are exactly the conclusion required by $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep classifier bounded coherence`

Theorem 34.49 (Singleton product pair-representation coherence)**.** *Fix histories $l_0, r_0 : \mathsf{BHist}$. Let $L(x) :\Longleftrightarrow \mathsf{hsame}(x, l_0)$ and $R(y) :\Longleftrightarrow \mathsf{hsame}(y, r_0)$. With both source classifiers specialized to* hsame*, the singleton source carriers satisfy product pair-representation coherence:*

$$\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}.$$

Proof. Apply Theorem 34.47. The required left bound sends a witness $L(l)$ to the displayed clause $\mathsf{hsame}(l, l_0)$, and the required right bound sends a witness $R(r)$ to the displayed clause $\mathsf{hsame}(r, r_0)$. □

Definition 34.50 (Product displayed pair ledger)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers. The product displayed pair ledger is the predicate*

$$\mathsf{ProdDisplayedPairLedger}_{L,R}(h) \Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r).$$

Theorem 34.51 (Product displayed pair ledger soundness)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers. For every history* $h : \mathsf{BHist}$*,*

$$\mathsf{ProdHistoryCarrier}(L, R, h) \to \mathsf{ProdDisplayedPairLedger}_{L,R}(h).$$

Proof. Assume $\mathsf{ProdHistoryCarrier}(L, R, h)$. Apply the forward direction of Theorem 34.25 to this carrier witness.

The theorem returns histories $l, r : \mathsf{BHist}$ with $\mathsf{ProdPairRep}_{L,R}(h, l, r)$. By Definition 34.50, this existential witness is exactly $\mathsf{ProdDisplayedPairLedger}_{L,R}(h)$. □

Theorem 34.52 (Product displayed pair ledger exactness and transport)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers. Write* $D(h) :\Longleftrightarrow \mathsf{ProdDisplayedPairLedger}_{L,R}(h)$*. For all histories* $h, k : \mathsf{BHist}$*,*

$$D(h) \Longleftrightarrow \mathsf{ProdHistoryCarrier}(L, R, h)$$

and

$$D(h) \wedge \mathsf{hsame}(h, k) \to D(k).$$

Proof. Unfold D and Definition 34.50. For the forward direction of the equivalence, a displayed pair ledger witness is a pair of histories l, r with $\mathsf{ProdPairRep}_{L,R}(h, l, r)$. Apply the reverse direction of Theorem 34.25 to obtain $\mathsf{ProdHistoryCarrier}(L, R, h)$.

For the reverse direction, start with $\mathsf{ProdHistoryCarrier}(L, R, h)$. Apply the forward direction of Theorem 34.25 to obtain histories l, r with $\mathsf{ProdPairRep}_{L,R}(h, l, r)$. Repack this displayed representation by Definition 34.50; this is $D(h)$.

For endpoint transport, assume $D(h)$ and $\mathsf{hsame}(h, k)$. Unfold D to obtain a displayed pair representation $\mathsf{ProdPairRep}_{L,R}(h, l, r)$. Apply Theorem 34.26 along $\mathsf{hsame}(h, k)$, keeping the same displayed component histories l, r, to obtain $\mathsf{ProdPairRep}_{L,R}(k, l, r)$. Repack this representation by Definition 34.50; the result is $D(k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdDisplayedPairLedger exact and transport`

Theorem 34.53 (Product carrier single-valued displayed pair readback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. For every history* $h : \mathsf{BHist}$*, if* $\mathsf{ProdHistoryCarrier}(L, R, h)$ *holds, then there exist* $l, r : \mathsf{BHist}$ *such that*

$$\mathsf{ProdPairRep}_{L,R}(h, l, r) \quad \textit{and} \quad \mathsf{ProdDisplayedPairLedger}_{L,R}(h)$$

and, for all $l', r' : \mathsf{BHist}$*,*

$$\mathsf{ProdPairRep}_{L,R}(h, l', r') \to (l \sim_L l') \wedge (r \sim_R r').$$

Proof. Assume $\mathsf{ProdHistoryCarrier}(L, R, h)$. Apply the forward direction of Theorem 34.25 to choose histories l, r with $\mathsf{ProdPairRep}_{L,R}(h, l, r)$.

By Definition 34.50, the same displayed representation gives $\mathsf{ProdDisplayedPairLedger}_{L,R}(h)$.

It remains to compare any competing representation $\mathsf{ProdPairRep}_{L,R}(h, l', r')$. Apply Definition 34.29 to the assumed $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$, the chosen representation, and the competing representation. The result is exactly $(l \sim_L l') \wedge (r \sim_R r')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdCarrier single valued displayed pair readback`

Definition 34.54 (Product displayed pair ledger chain)**.** *For source carriers* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, the relation* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *is the inductively generated displayed ledger-provenance relation on product histories with two constructors. A single displayed ledger gives a chain:*

$$\frac{\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, z)}{\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)}.$$

A displayed ledger followed by a chain through an intermediate history gives a chain:

$$\frac{\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v) \quad \mathsf{ProdHistoryLedgerChain}_{L,R}(v, z)}{\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)}.$$

Lean4 (definition) `BEDC.Derived.ProdUp.ProdHistoryLedgerChain`

Theorem 34.55 (Product ledger chain envelope closure)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\rho, z : \mathsf{BHist}$*. If* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *holds, then* $\mathsf{ProdHistoryCarrier}(L, R, z)$ *and* $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$ *hold. Moreover, for every* $w : \mathsf{BHist}$ *both history-envelope endpoint-replacement clauses hold:*

$$\mathsf{ProdHistoryClassifier}(L, R, \rho, w) \iff \mathsf{ProdHistoryClassifier}(L, R, z, w)$$

and

$$\mathsf{ProdHistoryClassifier}(L, R, w, \rho) \iff \mathsf{ProdHistoryClassifier}(L, R, w, z).$$

Proof. Induct on the derivation of $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$.

In the one-ledger constructor, Theorem 34.92 gives $\mathsf{ProdHistoryCarrier}(L, R, z)$ from the displayed ledger, and Theorem 34.93 gives $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$. The left endpoint-replacement clause is Theorem 34.97. The right endpoint-replacement clause follows by composing through the same raw-visible classifier with symmetry and transitivity of the product history classifier.

In the cons constructor, write the first displayed ledger as $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ and the remaining chain as $\mathsf{ProdHistoryLedgerChain}_{L,R}(v, z)$. The induction hypothesis supplies $\mathsf{ProdHistoryCarrier}(L, R, z)$, $\mathsf{ProdHistoryClassifier}(L, R, v, z)$, and the two endpoint equivalences from v to z.

The first ledger supplies $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$ by Theorem 34.93. Compose it with the induction classifier by Theorem 34.22 to obtain $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$. Compose the one-ledger endpoint equivalence from ρ to v with the induction endpoint equivalence from v to z for the left clause, and use the same endpoint transport through symmetry for the right clause. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain envelope closure`

Theorem 34.56 (Product ledger chain shared-raw visible classifier)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\rho, z, z' : \mathsf{BHist}$*. If both* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *and* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z')$ *hold, then* $\mathsf{ProdHistoryClassifier}(L, R, z, z')$ *holds.*

Proof. Apply Definition 34.55 to both ledger chains. The first envelope gives $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$ and the second gives $\mathsf{ProdHistoryClassifier}(L, R, \rho, z')$. Reverse the first classifier by product classifier symmetry, then compose it with the second classifier by product classifier transitivity. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain shared raw visible classifier`

Theorem 34.57 (Product ledger chain displayed component readback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. For histories* $\rho, z, l, r : \mathsf{BHist}$*, if* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *and* $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$ *hold, then there exist* $l', r' : \mathsf{BHist}$ *such that*

$$\mathsf{ProdPairRep}_{L,R}(z, l', r') \quad \textit{and} \quad l \sim_L l' \quad \textit{and} \quad r \sim_R r'.$$

Proof. Apply Theorem 34.55 to the ledger chain. It gives $\mathsf{ProdHistoryCarrier}(L, R, z)$ and $\mathsf{ProdHistoryClassifier}(L, R,$ Project the endpoint link $\mathsf{hsame}(\rho, z)$ from this envelope classifier.

Apply Theorem 34.25 to the final carrier to obtain $l', r' : \mathsf{BHist}$ with $\mathsf{ProdPairRep}_{L,R}(z, l', r')$.

Now apply Theorem 34.34 to the coherence witness, the source displayed representation $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$, the final displayed representation $\mathsf{ProdPairRep}_{L,R}(z, l', r')$, and the endpoint link $\mathsf{hsame}(\rho, z)$. It returns exactly $l \sim_L l'$ and $r \sim_R r'$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain displayed component readback`

Theorem 34.58 (Product ledger chain componentwise classifier endpoint equivalence)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, and let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. If* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *holds, then for every* $w : \mathsf{BHist}$ *both public componentwise endpoint-replacement clauses hold:*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(\rho, w) \iff \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(z, w)$$

and

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, \rho) \iff \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, z).$$

Proof. Apply Theorem 34.55 to the ledger chain. It gives $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$, whose stored endpoint link is $\mathsf{hsame}(\rho, z)$.

For the first displayed equivalence, transport a witness of $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(\rho, w)$ along $\mathsf{hsame}(\rho, z)$ on the left endpoint and hsame-reflexivity on w by Theorem 34.33. The reverse direction uses hsame-symmetry on the same endpoint link and hsame-reflexivity on w.

For the second displayed equivalence, transport a witness of $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, \rho)$ along hsame-reflexivity on w and $\mathsf{hsame}(\rho, z)$ on the right endpoint. The reverse direction again uses hsame-reflexivity on w and hsame-symmetry of $\mathsf{hsame}(\rho, z)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain componentwise classifier endpoint equivalence`

Theorem 34.59 (Product componentwise history core certificate)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$*,* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. Then the componentwise classifier carries the core product naming certificate:*

$$\mathsf{NameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}).$$

Proof. Invoke Theorem 34.45. Its five-field conclusion supplies carrier inhabitation, carrier-relative reflexivity, source-certified symmetry, source-certified transitivity, and classifier-induced carrier transport for $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$ over $\mathsf{ProdHistoryCarrier}(L, R)$. Assemble these clauses as the corresponding fields of the core $\mathsf{NameCert}$ record. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier name certificate`

Theorem 34.60 (Product componentwise history classifier semantic name certificate)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. *Then the componentwise classifier forms the semantic naming certificate whose source, pattern, and ledger predicates are all the concrete product history carrier:*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdComponentHis}$$

Proof. The componentwise core name certificate is Theorem 34.59. Its source carrier is $\mathsf{ProdHistoryCarrier}(L, R)$. Since the semantic pattern and ledger predicates are the same concrete carrier, both soundness maps are the identity on product history carrier witnesses. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier semantic name certificate`

Theorem 34.61 (Product componentwise history semantic certificate target)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. *Then the componentwise classifier forms the semantic naming certificate with the product displayed pair ledger:*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdDisplayedPairLedger}_{L,R}, \mathsf{ProdComponent}$$

Proof. Apply Theorem 34.59 to the two source certificates and the pair-representation coherence hypothesis. This gives the core $\mathsf{NameCert}$ certificate for $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$ over $\mathsf{ProdHistoryCarrier}(L, R)$. The semantic pattern carrier is $\mathsf{ProdHistoryCarrier}(L, R)$, so $\mathsf{pattern_sound}$ is identity on the carrier.

Ledger soundness is Theorem 34.51: every product history carrier witness exposes a named $\mathsf{ProdDisplayedPairLedger}_{L,R}$ witness. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier displayed ledger semantic name certificate`

Theorem 34.62 (Product componentwise classifier exposes displayed pair ledgers)**.** *The ten-field fit is: traditional target ordered-product displayed ledger exposure; BEDC source* BHist *with* Cont*; carrier through* $\mathsf{ProdPairRep}$*; classifier* $\mathsf{ProdComponentHistoryClassifier}$*; ledger* $\mathsf{ProdDisplayedPairLedger}$*; core theorem endpoint ledger exposure; exactness classifier endpoints carry displayed pair representatives; standard bridge ordered-pair readback; status* $\mathsf{ProofTarget}$.

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, and let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be component classifiers. For all histories* $h, k : \mathsf{BHist}$, *if*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$$

holds, then

$$\mathsf{ProdDisplayedPairLedger}_{L,R}(h) \wedge \mathsf{ProdDisplayedPairLedger}_{L,R}(k).$$

Proof. Unpack the componentwise classifier witness. Its endpoint data include a displayed pair representation of h by some left and right component histories and a displayed pair representation of k by some left and right component histories.

By Definition 34.50, the first representation is the named ledger witness for h, and the second representation is the named ledger witness for k. These two witnesses give the required conjunction. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdUp StdBridge`

Theorem 34.63 (Product componentwise public certificate package)**.** *The ten-field fit is: traditional target ordered-product public certificate package; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier *through* ProdPairRep*; classifier* ProdComponentHistoryClassifier*; stability inputs* ProdPairRepCoherent *and source* hsame*-soundness; ledger* ProdDisplayedPairLedger*; core theorem the semantic certificate, displayed pair ledger endpoints, and envelope descent together; exactness componentwise endpoint ledgers plus descent to* ProdHistoryClassifier*; standard bridge ordered-pair public readback; status* ProofTarget*. This theorem is not a variant of any one preceding theorem, because it composes certificate existence, ledger endpoints, and envelope descent in the single public-scope package required by the roadmap.*

Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. *Assume also that both source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

Then the following three facts hold:

1.

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdDisplayedPairLedger}_{L,R}, \mathsf{ProdCompon}$$

2. For all histories $h, k : \mathsf{BHist}$,

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \to \mathsf{ProdDisplayedPairLedger}_{L,R}(h) \wedge \mathsf{ProdDisplayedPairLedger}_{L,R}(k$$

3. For all histories $h, k : \mathsf{BHist}$,

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \to \mathsf{ProdHistoryClassifier}(L, R, h, k).$$

Proof. For the semantic certificate clause, apply Theorem 34.61 to cert_L, cert_R, and the $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,}$ witness.

For the displayed ledger clause, fix histories h and k and a witness of $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,$ Apply Theorem 34.62 to that witness; its conclusion is exactly the required pair of displayed ledger endpoints.

For the envelope descent clause, fix histories h and k and a witness of $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h$ Apply Theorem 34.79 with the two hsame-soundness hypotheses, giving $\mathsf{ProdHistoryClassifier}(L, R, h, k)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{Prod}^\uparrow}$**.**

Constructive story: An ordered pair here is made by putting two already certified histories on one visible line: choose a left witness from L and a right witness from R, then let the kernel continuation assemble h by appending the right history above the left one. Nothing else enters the product carrier, and two assembled histories count as the same product exactly when their recovered left components agree by L's classifier and their recovered right components agree by R's, so equality of the joined history is tracked back through the two component views. The certificate therefore uses that same product-history predicate as its source, records in the pattern the displayed pair shape and the cancellation rows that recover both sides, and keeps the raw continuation witness and the visible product history in the same ledger. Since the component classifiers are equivalences and continuation is deterministic, an admitted component pair gives the carrier its witness, componentwise sameness is reflexive, symmetric, and transitive, and anything it recognizes remains an admitted product. A history h lies in the signature gap of a $\mathsf{Prod}^\uparrow$ token p when the source-domain accepts the assembled history, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that

answer as p; unfolded, h must decompose into two component histories whose left and right classifiers identify them with the components encoded by p. The chapter registers this parallel-display construction under the name $\mathsf{Prod}^{\uparrow}$; an ordered pair is the kernel's continuation relation used as two named components in view, with no host tuple type or projection axiom introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{Prod}\uparrow}$, maturePackageClosed)
Scope closed: Definition 34.63 is the public surface packaging Definition 34.61, Definition 34.62, and Definition 34.79.
Formal verification: FormalStatus(NameCert$_{\mathsf{Prod}\uparrow}$, bridgeChecked)
Lean target: `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier\_semantic\_name\_certificate`
Bridge status: bridgeChecked
Not claimed: This chapter does not close dependent products, function-space products, product categories, or host tuple equality.
Upgrade path: No further upgrade – chapter at top of the formal axis.

The componentwise envelope route consumes the visible branch discipline before it forms the **??** pair ledger. The tag separation of chapter 32 and the visible-branch readbacks of chapter 35 and chapter 33 are the finite-data pattern: a product endpoint is readable only after its left and right component histories remain separately displayed, just as those siblings keep their branch or absence rows visible before a classifier is consumed.

Theorem 34.64 (Product ledger visible pair-representation readback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history predicates and let* $\rho, v : \mathsf{BHist}$*. Define the displayed pair ledger* $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$*. If* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ *holds, then both* $D(v)$ *and* $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$ *hold.*

Proof. The per-object fit is: traditional target ordered-product visible ledger readback; BEDC source BHist with Cont and hsame; carrier ProdHistoryCarrier through ProdPairRep; classifier ProdHistoryClassifier; stability endpoint transport; ledger ProdHistoryLedgerPolicy plus displayed pair ledger D; core theorem checked ledger descent; exactness visible endpoint pair representation; standard bridge ordered-pair visible readback; status ProofTarget. Unpack the ledger policy to obtain the raw product carrier and the raw-visible hsame endpoint link. Apply Theorem 34.25 to the raw carrier, giving histories l, r with $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$. Transport this displayed pair representation along the raw-visible endpoint link by Theorem 34.26; the result is the witness for $D(v)$. The classifier clause is exactly Theorem 34.93. □

Theorem 34.65 (Product checked envelope admits displayed pair ledger)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, and assume* $\exists l : \mathsf{BHist}.\ L(l)$ *and* $\exists r : \mathsf{BHist}.\ R(r)$*. Define the displayed pair ledger* $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$*. Then*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdHistoryClassifier}(L, R))$$

holds.

Proof. The per-object fit is: traditional target ordered-product visible ledger; BEDC source BHist with Cont; carrier ProdHistoryCarrier; classifier ProdHistoryClassifier; ledger displayed pair representation; core theorem Theorem 34.107; exactness via pair-representation coverage; standard bridge ordinary ordered pairs; public certificate scope SemanticNameCert; status ProofTarget. Instantiate Theorem 34.107 with the two source-history witnesses. Its checked envelope certificate supplies the NameCert core fields and pattern soundness for source and pattern both equal to $\mathsf{ProdHistoryCarrier}(L, R)$. It remains only to replace the ledger field: given a source witness $\mathsf{ProdHistoryCarrier}(L, R, h)$, apply the forward direction of Theorem 34.25 to obtain histories l and r with $\mathsf{ProdPairRep}_{L,R}(h, l, r)$, which is exactly $D(h)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryCarrier displayed pair semantic name certificate`

Theorem 34.66 (Singleton history source certificate)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$*, and define* $A_{a_0} : \mathsf{BHist} \to \mathsf{Prop}$ *by* $A_{a_0}(x) \iff \mathsf{hsame}(x, a_0)$*. Then* $\mathsf{NameCert}(A_{a_0}, \mathsf{hsame})$ *holds.*

Proof. The carrier is inhabited by a_0 using reflexivity of hsame. The reflexivity, symmetry, and transitivity fields of the certificate are exactly the corresponding fields of hsame. For carrier transport, start with $A_{a_0}(x)$, namely $\mathsf{hsame}(x, a_0)$, and a classifier witness $\mathsf{hsame}(x, y)$. Symmetry turns the classifier witness into $\mathsf{hsame}(y, x)$, and transitivity with $\mathsf{hsame}(x, a_0)$ yields $\mathsf{hsame}(y, a_0)$, which is $A_{a_0}(y)$. □

Lean4 ✓ `BEDC.Derived.SingletonSource.SingletonHistorySource name certificate`

Theorem 34.67 (Singleton componentwise product semantic certificate)**.** *Fix histories* $l_0, r_0 : \mathsf{BHist}$*. Define singleton source carriers* $L(x) :\iff \mathsf{hsame}(x, l_0)$ *and* $R(y) :\iff \mathsf{hsame}(y, r_0)$*. With displayed pair ledger* $D(h) :\iff \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$*, the componentwise product classifier specialized to* hsame *forms the semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hs}}$$

Proof. Apply Theorem 34.66 at l_0 and at r_0 to obtain the two singleton $\mathsf{NameCert}$ records for L and R. Theorem 34.49 supplies the pair-representation coherence hypothesis for these singleton carriers. Apply Theorem 34.61 with the two singleton certificates and this coherence witness. □

Theorem 34.68 (Singleton product componentwise classifier endpoint exactness)**.** *Fix histories* $l_0, r_0 : \mathsf{BHist}$*, and define* $L(x) :\iff \mathsf{hsame}(x, l_0)$ *and* $R(y) :\iff \mathsf{hsame}(y, r_0)$*. Then for all histories* $h, k : \mathsf{BHist}$*,*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k) \iff \mathsf{hsame}(h, \mathsf{append}(l_0, r_0)) \wedge \mathsf{hsame}(k, \mathsf{append}(l_0, r_0)).$$

Proof. For the forward direction, unpack the componentwise classifier into displayed pair representations for h and k and component links between their left and right representatives. The singleton source facts identify the left representatives with l_0 and the right representatives with r_0.

Use transitivity of hsame to compare each displayed component with the fixed components l_0, r_0. The displayed representation of $\mathsf{append}(l_0, r_0)$ is obtained from reflexivity component witnesses and the canonical continuation result. Continuation determinism, applied to the displayed continuations for h and k against this fixed displayed continuation, yields the two endpoint witnesses $\mathsf{hsame}(h, \mathsf{append}(l_0, r_0))$ and $\mathsf{hsame}(k, \mathsf{append}(l_0, r_0))$.

For the reverse direction, start from the displayed representation of $\mathsf{append}(l_0, r_0)$ with components l_0, r_0. Transport this representation along the two endpoint hsame witnesses by Theorem 34.26, obtaining displayed pair representations for h and k with the same components.

The component comparison clauses are the reflexivity witnesses $\mathsf{hsame}(l_0, l_0)$ and $\mathsf{hsame}(r_0, r_0)$. Together with the two transported displayed representations, these witnesses rebuild $\mathsf{ProdComponentHistoryClassifie}$ □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier singleton endpoint exactness`

Theorem 34.69 (Singleton product displayed pair ledger endpoint exactness)**.** *Fix histories* $l_0, r_0 : \mathsf{BHist}$*, and define* $L(x) :\iff \mathsf{hsame}(x, l_0)$ *and* $R(y) :\iff \mathsf{hsame}(y, r_0)$*. With displayed pair ledger* $D(h) :\iff \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$*, every history* $h : \mathsf{BHist}$ *satisfies*

$$D(h) \iff \mathsf{hsame}(h, \mathsf{append}(l_0, r_0)).$$

Proof. For the forward direction, unpack $D(h)$ to obtain histories l, r and a displayed representation $\mathsf{ProdPairRep}_{L,R}(h, l, r)$. Its singleton source clauses give $\mathsf{hsame}(l, l_0)$ and $\mathsf{hsame}(r, r_0)$, while its continuation clause presents h from the displayed components.

Transport the displayed components to l_0, r_0 and compare the transported continuation with the canonical continuation for $\mathsf{append}(l_0, r_0)$. The fixed-pair endpoint determinism of Theorem 34.27, applied after this component transport, yields $\mathsf{hsame}(h, \mathsf{append}(l_0, r_0))$.

For the reverse direction, start from the canonical displayed pair representation of $\mathsf{append}(l_0, r_0)$ with components l_0, r_0. Given $\mathsf{hsame}(h, \mathsf{append}(l_0, r_0))$, symmetry converts it to the orientation needed to move that representation back to h.

Theorem 34.26 carries the canonical representation along this endpoint witness, keeping the displayed components l_0, r_0. The resulting $\mathsf{ProdPairRep}_{L,R}(h, l_0, r_0)$ supplies the existential witnesses for $D(h)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPairRep singleton endpoint exactness iff`

Theorem 34.70 (Product componentwise certificate refines the checked envelope)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers, let $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be their source classifiers, and assume $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. Assume also that the source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

With $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$, both of the following hold:

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R})$$

and, for all histories h, k,

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \to \mathsf{ProdHistoryClassifier}(L, R, h, k).$$

Proof. The first conclusion is Theorem 34.61 applied to the two source certificates and the pair-representation coherence witness. The second conclusion is Theorem 34.79 applied to the two hsame-soundness assumptions. □

Theorem 34.71 (Bounded classifier product certificate refines the checked envelope)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers, let $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be their source classifiers, and assume $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume also that the source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

Suppose there are histories $l_0, r_0 : \mathsf{BHist}$ such that

$$\forall l.\ L(l) \to l \sim_L l_0$$

and

$$\forall r.\ R(r) \to r \sim_R r_0.$$

With displayed pair ledger $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$, the following three clauses hold:

1. *the componentwise public certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,}$$

2. *the checked envelope certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryClass}$$

3. *for all histories h, k,*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k) \to D(h) \wedge D(k) \wedge \mathsf{ProdHistoryClassifier}(L,R,h,k).$$

Proof. Apply Theorem 34.48 to the two source certificates and the two center-history bounds. This yields $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. The componentwise public certificate is Theorem 34.61 applied to the two source certificates and this coherence witness. The checked envelope certificate is Theorem 34.107, using the inhabited carrier fields of cert_L and cert_R as the two source-history witnesses. For the endpoint exposure clause, Theorem 34.62 supplies $D(h)$ and $D(k)$ from the componentwise classifier witness, while Theorem 34.79 supplies $\mathsf{ProdHistoryClassifier}(L,R,h,k)$ from the same witness and the two hsame-soundness hypotheses. Conjoin the three conclusions. □

Theorem 34.72 (Product componentwise certificate descends to the checked envelope)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source-history carriers, let $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be their source classifiers, and assume $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. Assume also that both source classifiers are hsame-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x,y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x,y).$$

With displayed pair ledger $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h,l,r)$, the following three clauses hold:

1. *the componentwise public certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,}$$

2. *the checked envelope certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryCarrier}(L,R), \mathsf{ProdHistoryClass}$$

3. *for all histories h, k,*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h,k) \to D(h) \wedge D(k) \wedge \mathsf{ProdHistoryClassifier}(L,R,h,k).$$

This theorem records componentwise-to-envelope compatibility and displayed ledger endpoint exposure for the ordered-product public scope; classifier equivalence is supplied by the hsame*-specialized Theorem 34.75.*

Proof. Theorem 34.70 supplies the componentwise public certificate and the descent implication from the componentwise classifier to the checked envelope classifier. Theorem 34.62 supplies $D(h)$ and $D(k)$ from the same componentwise classifier witness. Conjoin those two displayed ledger endpoints with the descent implication. The checked envelope certificate is Theorem 34.107: the inhabited carrier fields of cert_L and cert_R provide the left and right carrier witnesses required by that theorem. □

Theorem 34.73 (Product componentwise public scope is stable under endpoint transport)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and assume that both source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

Define the displayed pair ledger by $D(x) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(x, l, r)$*. For histories* $h, k, h', k' : \mathsf{BHist}$*, if*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k), \qquad \mathsf{hsame}(h, h'), \qquad \mathsf{hsame}(k, k')$$

hold, then

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h', k') \wedge D(h') \wedge D(k') \wedge \mathsf{ProdHistoryCarrier}(L, R, h') \wedge \mathsf{ProdHistoryCarrier}(L, R$$

The roadmap fit is: traditional target ordered-product public-scope endpoint stability; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdComponentHistoryClassifier*; stability inputs endpoint transport and* hsame*-sound envelope descent; ledger displayed pair representation* D*; core theorem the transported componentwise classifier plus displayed ledger endpoints and endpoint carrier exposure; exactness displayed ledger plus envelope classifier; standard bridge ordered-pair readback; status* ProofTarget*.*

Proof. Apply Theorem 34.33 to the componentwise classifier witness at (h, k) and the two endpoint hsame links, obtaining $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h', k')$.

Apply Theorem 34.62 to the transported classifier. This gives $D(h')$ and $D(k')$ for the displayed pair ledger.

Apply Theorem 34.35 to the same transported classifier. This exposes $\mathsf{ProdHistoryCarrier}(L, R, h')$ and $\mathsf{ProdHistoryCarrier}(L, R, k')$.

Finally apply Theorem 34.79 with the two hsame-soundness assumptions to the transported classifier, obtaining $\mathsf{ProdHistoryClassifier}(L, R, h', k')$. Conjoin the transported classifier with the two ledger endpoints, the two endpoint carriers, and the envelope classifier. □

Theorem 34.74 (Product envelope displayed-pair component exactness)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *and assume both source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

For histories $h, k, l, r, l', r' : \mathsf{BHist}$*, if* $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *and* $\mathsf{ProdPairRep}_{L,R}(k, l', r')$ *hold, then*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \Longleftrightarrow (l \sim_L l') \wedge (r \sim_R r').$$

Proof. For the forward direction, unpack the checked envelope classifier to obtain the endpoint link $\mathsf{hsame}(h, k)$. Symmetry gives $\mathsf{hsame}(k, h)$, and Theorem 34.26 carries the displayed representation of k with components l', r' back to a displayed representation of h with the same components. The $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ witness then compares this transported representation with the given displayed representation $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ and yields $l \sim_L l'$ and $r \sim_R r'$.

For the reverse direction, use the two displayed representations and the two component links as a witness of $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$. Apply Theorem 34.79 with the two hsame-soundness assumptions to obtain $\mathsf{ProdHistoryClassifier}(L, R, h, k)$. □

Theorem 34.75 (Product hsame componentwise certificate is envelope-equivalent)**.** *Let* L, R : $\mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers. Assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \mathsf{hsame})$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \mathsf{hsame})$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$. *With displayed pair ledger* $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$, *both semantic certificates hold:*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryClassifie}$$

and

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{h}}$$

Moreover, for all histories h, k,

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \Longleftrightarrow \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k).$$

Proof. The checked envelope certificate is Theorem 34.107 applied to cert_L.carrier_inhabited and cert_R.carrier_inhabited. The displayed componentwise certificate is Theorem 34.61 with both source classifiers specialized to hsame and with the given $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$ witness. The classifier equivalence is exactly Theorem 34.82. □

Theorem 34.76 (Bounded hsame product certificates are envelope-equivalent)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \mathsf{hsame})$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \mathsf{hsame})$. *Suppose there are histories* $l_0, r_0 : \mathsf{BHist}$ *such that*

$$\forall l.\ L(l) \to \mathsf{hsame}(l, l_0)$$

and

$$\forall r.\ R(r) \to \mathsf{hsame}(r, r_0).$$

With displayed pair ledger $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$, *both semantic certificates hold:*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryClassifie}$$

and

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{h}}$$

Moreover, for all histories h, k,

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \Longleftrightarrow \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k).$$

Proof. Apply Theorem 34.47 to the two center histories and the two boundedness hypotheses. This gives the coherence witness $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$. Then apply Theorem 34.75 to cert_L, cert_R, and that coherence witness. Its displayed ledger is exactly D, and its conclusion is the two semantic certificates together with the classifier equivalence for arbitrary histories h and k. □

Theorem 34.77 (Singleton hsame product certificates are envelope-equivalent)**.** *This theorem fits the per-object template as follows: the traditional target is the singleton ordered-product bridge; the BEDC source is* BHist *with* Cont *and* hsame*; the carrier is* $\mathsf{ProdHistoryCarrier}(L, R)$*; the classifier*

pair is the envelope classifier and the componentwise classifier; the stability data are the singleton source certificates and pair coherence; the ledger is the displayed pair representation; the core theorem is the pair of semantic certificates together with the classifier equivalence; exactness is the classifier equivalence; the standard bridge is singleton ordered-pair readback; and the status is ProofTarget. *Fix histories* $l_0, r_0 : \mathsf{BHist}$. *Define* $L(x) :\Longleftrightarrow \mathsf{hsame}(x, l_0)$ *and* $R(y) :\Longleftrightarrow \mathsf{hsame}(y, r_0)$. *With displayed pair ledger* $D(h) :\Longleftrightarrow \exists l, r : \mathsf{BHist}.\ \mathsf{ProdPairRep}_{L,R}(h, l, r)$, *both semantic certificates hold:*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryClassifie}$$

and

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), D, \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hs}}$$

Moreover, for all histories h, k,

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \Longleftrightarrow \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k).$$

Proof. Obtain the two singleton source NameCert records from Theorem 34.66, first at l_0 and then at r_0. The singleton definitions of L and R supply the left and right carrier witnesses for the checked envelope. Theorem 34.49 gives the required pair-representation coherence. Applying Theorem 34.75 with these certificates, witnesses, and the coherence proof yields the two semantic certificates and the displayed classifier equivalence. □

Theorem 34.78 (Product componentwise history classifier inversion)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *and the displayed source transitivity laws*

$$\forall a\, b\, c.\ a \sim_L b \to b \sim_L c \to a \sim_L c \qquad \textit{and} \qquad \forall a\, b\, c.\ a \sim_R b \to b \sim_R c \to a \sim_R c.$$

If $\mathsf{ProdPairRep}_{L,R}(h, l, r)$, $\mathsf{ProdPairRep}_{L,R}(k, l', r')$, *and* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$ *hold, then* $l \sim_L l'$ *and* $r \sim_R r'$.

Proof. Unpack the componentwise classifier witness to obtain component histories l_h, r_h, l_k, r_k with pair representations for h and k, together with $l_h \sim_L l_k$ and $r_h \sim_R r_k$. Apply pair-representation coherence at h to the displayed representation (l, r) and the witness representation (l_h, r_h), yielding $l \sim_L l_h$ and $r \sim_R r_h$. Apply coherence at k to the witness representation (l_k, r_k) and the displayed representation (l', r'), yielding $l_k \sim_L l'$ and $r_k \sim_R r'$. The displayed source transitivity hypotheses compose the left chain $l \sim_L l_h \sim_L l_k \sim_L l'$ and the right chain $r \sim_R r_h \sim_R r_k \sim_R r'$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier inversion`

The descent theorem instantiates the roadmap template as traditional ordered-pair descent over the BEDC source BHist with Cont, carrier ProdHistoryCarrier, classifier bridge from componentwise to envelope under hsame-sound source classifiers, ledger by displayed pair representations, and status ProofTarget.

Theorem 34.79 (Product componentwise history classifier descends to envelope)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume the source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y) \qquad \textit{and} \qquad \forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

If $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$ *holds, then* $\mathsf{ProdHistoryClassifier}(L, R, h, k)$ *holds.*

Proof. Unpack the componentwise classifier into component histories l_h, r_h, l_k, r_k, pair representations $\mathsf{ProdPairRep}_{L,R}(h, l_h, r_h)$ and $\mathsf{ProdPairRep}_{L,R}(k, l_k, r_k)$, and component links $l_h \sim_L l_k$ and $r_h \sim_R r_k$. The two hsame-sound assumptions turn the component links into $\mathsf{hsame}(l_h, l_k)$ and $\mathsf{hsame}(r_h, r_k)$. The pair representations expose $\mathsf{Cont}(l_h, r_h, h)$ and $\mathsf{Cont}(l_k, r_k, k)$, so continuation respect for hsame yields $\mathsf{hsame}(h, k)$. Pair-representation coverage rebuilds $\mathsf{ProdHistoryCarrier}(L, R, h)$ and $\mathsf{ProdHistoryCarrier}(L, R, k)$ from the two displayed pair representations, and these two endpoint carriers together with $\mathsf{hsame}(h, k)$ rebuild $\mathsf{ProdHistoryClassifier}(L, R, h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier descends to envelope`

The recovery theorem instantiates the roadmap template as traditional ordered-product envelope-to-componentwise readback over the BEDC source BHist with Cont and hsame, carrier ProdHistoryCarrier, classifier ProdComponentHistoryClassifier, stability via pair-representation coherence, ledger displayed pair representation, core theorem coherent envelope recovery, exactness component readback from envelope, standard bridge ordered-pair readback, and status ProofTarget.

Theorem 34.80 (Product envelope classifier recovers coherent componentwise classifier)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and let* $h, k : \mathsf{BHist}$*. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. If* $\mathsf{ProdHistoryClassifier}(L, R, h, k)$ *holds, then* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$ *holds.*

Proof. Unpack the envelope classifier into endpoint carriers for h and k and the stored endpoint link $\mathsf{hsame}(h, k)$. Apply Theorem 34.25 to the two endpoint carriers to choose displayed pair representations $\mathsf{ProdPairRep}_{L,R}(h, l_h, r_h)$ and $\mathsf{ProdPairRep}_{L,R}(k, l_k, r_k)$.

Apply Theorem 34.34 to the coherence datum, the two displayed pair representations, and the endpoint link. The resulting component links $l_h \sim_L l_k$ and $r_h \sim_R r_k$, together with the displayed pair representations, repack as $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier pair coherent componentwise`

Theorem 34.81 (Product envelope hsame classifier recovers componentwise classifier)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$*. If* $\mathsf{ProdHistoryClassifier}(L, R,$ *holds, then* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k)$ *holds.*

Proof. Unpack the envelope classifier into endpoint carrier witnesses for h and k and an endpoint link $\mathsf{hsame}(h, k)$. Pair-representation coverage for the two endpoint carriers supplies displayed pair representations $\mathsf{ProdPairRep}_{L,R}(h, l_h, r_h)$ and $\mathsf{ProdPairRep}_{L,R}(k, l_k, r_k)$. First reverse the endpoint link to obtain $\mathsf{hsame}(k, h)$. Apply Theorem 34.26 to the displayed representation of k and this reversed link, producing a displayed pair representation of h with the same components l_k, r_k. Pair-representation coherence at h compares the two representations of h and gives $\mathsf{hsame}(l_h, l_k)$ and $\mathsf{hsame}(r_h, r_k)$. These two component links together with the original displayed pair representations for h and k rebuild $\mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier pair coherent componentwise hsame`

Theorem 34.82 (Product envelope and componentwise hsame classifiers coincide)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and assume* $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$*. For all* $h, k : \mathsf{BHist}$*,*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \iff \mathsf{ProdComponentHistoryClassifier}_{L,R,\mathsf{hsame},\mathsf{hsame}}(h, k).$$

Proof. The forward implication is Theorem 34.81. For the reverse implication, apply Theorem 34.79 with both source classifiers specialized to hsame. The two required hsame-soundness premises are discharged by the identity proof on the given component links. Thus the envelope classifier and the componentwise hsame classifier have exactly the same histories under the displayed pair-representation coherence. □

This exactness statement instantiates the roadmap template as traditional ordered-pair equality agreement over the BEDC source BHist with Cont, carrier ProdHistoryCarrier, classifier agreement between the envelope and componentwise hsame classifiers, stability by $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$, ledger through displayed pair representations, and status ProofTarget.

Theorem 34.83 (Product history classifier displayed component hsame exactness)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and assume* $\mathsf{ProdPairRepCoherent}_{L,R,\mathsf{hsame},\mathsf{hsame}}$. *If* $L(l)$, $R(r)$, $\mathsf{Cont}(l, r, h)$, $L(l')$, $R(r')$, *and* $\mathsf{Cont}(l', r', k)$ *hold, then*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k) \iff \mathsf{hsame}(l, l') \wedge \mathsf{hsame}(r, r').$$

Proof. For the forward direction, the displayed component data are $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ and $\mathsf{ProdPairRep}_{L,R}(k, l', r')$. The product history classifier supplies the endpoint link $\mathsf{hsame}(h, k)$, so pair-representation coherence transports the endpoint link back to the two component links. For the reverse direction, the two component links are pushed through the two continuation witnesses by continuation respect for hsame, producing $\mathsf{hsame}(h, k)$. The displayed component data rebuild the two endpoint product carriers, and these three fields are exactly $\mathsf{ProdHistoryClassifier}(L, R, h, k)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryClassifier displayed component hsame iff`

Theorem 34.84 (Product componentwise history classifier displayed exactness)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers. Assume* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *and the displayed source transitivity laws*

$$\forall a\, b\, c.\ a \sim_L b \to b \sim_L c \to a \sim_L c \qquad \textit{and} \qquad \forall a\, b\, c.\ a \sim_R b \to b \sim_R c \to a \sim_R c.$$

If $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ *and* $\mathsf{ProdPairRep}_{L,R}(k, l', r')$, *then*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(h, k) \iff (l \sim_L l') \wedge (r \sim_R r').$$

Proof. For the forward direction, apply Theorem 34.78 to the two displayed pair representations and the assumed componentwise classifier. For the reverse direction, use the displayed pair representations $\mathsf{ProdPairRep}_{L,R}(h, l, r)$ and $\mathsf{ProdPairRep}_{L,R}(k, l', r')$ as the existential witnesses in Definition 34.31; the two displayed component relations supply the remaining conjuncts. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdComponentHistoryClassifier displayed exactness`

Theorem 34.85 (Product component-classifier pullback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers and let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers satisfying the displayed transitivity laws of Definition 34.84. For any public product-envelope consumers* C *and* D, *if their displayed reads of a pair envelope agree on the left component classifier and on the right component classifier, then their reads pull back to the same* $\mathsf{Prod}^{\uparrow}$ *envelope classifier. The pullback is measured through* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$, *descends to* $\mathsf{ProdHistoryClassifier}(L, R, -, -)$ *when the component classifiers are* hsame*-sound, and is recorded by the product stability certificate rows.*

Proof. Project both public consumers to their displayed product rows. The exactness theorem Definition 34.84 rewrites each componentwise product classifier as the conjunction of the two displayed component classifier agreements.

Because C and D agree on both component classifiers, those two conjunctions are the same displayed witness. Repack the two component agreements through Definition 34.31; this gives the common ProdComponentHistoryClassifier row for the two product-envelope reads.

Envelope descent is then Definition 34.79 under the declared hsame-soundness of the component classifiers. The displayed hsame exactness theorem Definition 34.83 identifies the envelope readback with the same component data, and Definition 34.86 records the stability rows that let finite-data consumers reuse the pullback. □

Definition 34.86 (Prod stability certificate)**.** *The checked stability subcertificate consists of:*

1. *componentwise reflexivity obtained from the two source reflexivity fields*
2. *componentwise transitivity obtained from the two source transitivity fields*
3. *coordinate exposure: any* ProdClassifierSpec(R_A, R_B, x, y) *witness yields* $R_A(x.1, y.1)$ *and* $R_B(x.2, y.2)$

Relation-level symmetry is supplied by Theorem 34.88; the source-certificate theorem is obtained from the two equiv_symm *fields, while the* hsame *specialization is Theorem 34.90.*

Lean4 ✓ `BEDC.Derived.ProdUp.prod stability certificate fields`

Theorem 34.87 (Product classifier transitivity at the typed level)**.** *Let* $R_A : A \to A \to \mathsf{Prop}$ *and* $R_B : B \to B \to \mathsf{Prop}$ *be source classifier relations. If both* R_A *and* R_B *are transitive, then for product carriers* x*,* y*, and* z*,*

$$\mathsf{ProdClassifierSpec}(R_A, R_B, x, y) \quad \textit{and} \quad \mathsf{ProdClassifierSpec}(R_A, R_B, y, z)$$

imply ProdClassifierSpec(R_A, R_B, x, z)*.*

Lean4 ✓ `BEDC.Derived.ProdUp.ProdClassifierSpec trans`

Proof. Invert the product classifier witnesses by their constructor. The first components compose by transitivity of R_A, the second components compose by transitivity of R_B, and the paired typed witness rebuilds the product classifier. □

Theorem 34.88 (Product classifier symmetry from source symmetry)**.** *Let* $R_A : A \to A \to \mathsf{Prop}$ *and* $R_B : B \to B \to \mathsf{Prop}$ *be source classifier relations. If*

$$\forall a\, a'.\ R_A(a, a') \to R_A(a', a)$$

and

$$\forall b\, b'.\ R_B(b, b') \to R_B(b', b)$$

hold, then for product carriers x *and* y*,* ProdClassifierSpec(R_A, R_B, x, y) *implies* ProdClassifierSpec(R_A, R_B, y, x)*.*

Proof. Invert the product classifier witness into its two coordinate witnesses. Apply source symmetry for R_A to the first coordinate and source symmetry for R_B to the second coordinate, then rebuild the product classifier with the same constructor in the opposite order. □

Theorem 34.89 (Product classifier symmetry from source certificates)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and let* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$ *be source name certificates. For product carriers* $x, y : \mathsf{ProdCarrier}(\mathsf{BHist}, \mathsf{BHist})$*,* $\mathsf{ProdClassifierSpec}(\sim_L, \sim_R, x, y)$ *implies* $\mathsf{ProdClassifierSpec}(\sim_L, \sim_R, y, x)$*.*

Proof. Invert the product classifier witness into its left and right component classifier proofs. Apply the equiv_symm field of cert_L to reverse the left component, and the equiv_symm field of cert_R to reverse the right component. The product classifier constructor then rebuilds the componentwise witness in the opposite order. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdClassifierSpec symm from nameCert`

Theorem 34.90 (History product classifier is symmetric)**.** *For pairs of finite-kernel histories, if the product classifier relates* x *to* y *componentwise by* hsame*, then it relates* y *to* x*.*

Proof. Invert the componentwise product classifier. Each component witness reverses by symmetry of hsame, and the reversed witnesses rebuild the product classifier in the opposite direction. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdClassifierSpec hsame symm`

Definition 34.91 (Prod ledger policy)**.** *Record a raw product history carrying both component ledgers together with an* hsame *link to the visible history exposed by the ledger.*

Lean4 (definition) `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy`

The product ledger is the binary finite-data counterpart of the tagged-sum and option surfaces of chapter 35 and chapter 33: product component exposure uses the same tag/readback discipline, but pairs two visible components instead of selecting one branch.

Theorem 34.92 (Prod ledger visible carrier)**.** *If a visible history is linked by* hsame *to a raw product-ledger history, then the visible history is itself a product carrier.*

Proof. Unpack the raw product carrier and transport it across the stored hsame link by product-carrier transport. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy visible carrier`

Theorem 34.93 (Prod ledger raw-visible classifier)**.** *If* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ *holds, then* $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$ *holds.*

Proof. Unpack the ledger into the raw product carrier for ρ and the stored $\mathsf{hsame}(\rho, v)$ link. The visible-carrier theorem supplies the product carrier for v. These three witnesses are exactly the product classifier fields for the raw ledger history and the visible history. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy raw visible classifier`

Theorem 34.94 (Prod ledger history-same transport)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, \rho', v, v' : \mathsf{BHist}$ *be histories. If* $\mathsf{hsame}(\rho, \rho')$*,* $\mathsf{hsame}(v, v')$*, and* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ *hold, then* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho', v')$ *holds. The object fit is: traditional target ordered-product ledger stability; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; ledger* ProdHistoryLedgerPolicy*; exactness raw-visible transport; standard bridge displayed ordered-pair ledger; status* ProofTarget*.*

Proof. Unpack the ledger into a raw product carrier for ρ and the stored raw-visible link $\mathsf{hsame}(\rho, v)$. Transport the raw carrier across $\mathsf{hsame}(\rho, \rho')$ by Theorem 34.8. For the visible link, compose symmetry of $\mathsf{hsame}(\rho, \rho')$, the stored raw-visible link, and $\mathsf{hsame}(v, v')$ to obtain $\mathsf{hsame}(\rho', v')$. The transported raw carrier and this composed link rebuild $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho', v')$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy hsame transport`

Theorem 34.95 (Prod ledger classifier composition)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, w : \mathsf{BHist}$ be histories. If $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ holds and $\mathsf{ProdHistoryClassifier}(L, R, v, w)$ holds, then $\mathsf{ProdHistoryClassifier}(L, R, \rho, w)$ holds. The object fit is: traditional target ordered-product ledger propagation; BEDC source* BHist *with* Cont*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; ledger evidence the raw-visible pair; exactness raw-to-visible-to-target composition; status* ProofTarget*.*

Proof. Apply Theorem 34.93 to the ledger policy to obtain $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$. Compose that raw-visible classifier with the visible-target classifier by product history classifier transitivity, yielding $\mathsf{ProdHistoryClassifier}(L, R, \rho, w)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy classifier extension`

Theorem 34.96 (Prod ledger backward classifier composition)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, w : \mathsf{BHist}$ be histories. If $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ holds and $\mathsf{ProdHistoryClassifier}(L, R, w, v)$ holds, then $\mathsf{ProdHistoryClassifier}(L, R, w, \rho)$ holds. The object fit is: traditional target ordered-product ledger back-propagation; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; ledger raw-visible pair evidence; core theorem backward classifier composition; exactness raw-visible symmetry; standard bridge displayed ordered-pair ledger; status* ProofTarget*.*

Proof. Apply Theorem 34.93 to the ledger policy to obtain $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$. Reverse this classifier by Theorem 34.20, giving $\mathsf{ProdHistoryClassifier}(L, R, v, \rho)$. Compose the assumed classifier from w to v with this reversed classifier by Theorem 34.22, yielding $\mathsf{ProdHistoryClassifier}(L, R, w, \rho)$. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy classifier backward composition`

Theorem 34.97 (Prod ledger classifier endpoint equivalence)**.** *Let $L, R : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, w : \mathsf{BHist}$ be histories. If $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ holds, then $\mathsf{ProdHistoryClassifier}(L, R, \rho, w) \iff \mathsf{ProdHistoryClassifier}(L, R, v, w)$. The object fit is: traditional target ordered-product ledger endpoint replacement; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; stability by raw-visible classifier propagation in both directions; ledger* ProdHistoryLedgerPolicy*; core theorem this endpoint equivalence; exactness classifier-invariant raw and visible endpoints; standard bridge displayed ordered-pair ledger.*

Proof. Apply Theorem 34.93 to the ledger policy to obtain $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$. For the forward direction, reverse this raw-visible classifier by Theorem 34.20 and compose it with the assumed $\mathsf{ProdHistoryClassifier}(L, R, \rho, w)$ by Theorem 34.22. For the reverse direction, apply Theorem 34.95 to the same ledger policy and the assumed classifier from v to w. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy classifier endpoint equivalence`

Theorem 34.98 (Prod ledger two-step classifier composition)**.** *Let* L, R : BHist $\to$ Prop *be source carriers and let* ρ, v, w : BHist *be histories. If* ProdHistoryLedgerPolicy(L, R, ρ, v) *and* ProdHistoryLedgerPolicy(L, R, v, w) *hold, then* ProdHistoryClassifier(L, R, ρ, w) *holds. The object fit is: ordered-product ledger chain; BEDC source* BHist *with* Cont*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; ledger evidence raw-visible-visible; exactness by classifier composition.*

Proof. Apply Theorem 34.93 to both ledger policies to obtain classifiers from ρ to v and from v to w. Compose them by Theorem 34.22. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy two step classifier composition`

Theorem 34.99 (Prod ledger two-step classifier endpoint equivalence)**.** *Let* L, R : BHist $\to$ Prop *be source carriers and let* ρ, v, w, z : BHist *be histories. If* ProdHistoryLedgerPolicy(L, R, ρ, v) *and* ProdHistoryLedgerPolicy(L, R, v, w) *hold, then* ProdHistoryClassifier$(L, R, \rho, z) \iff$ ProdHistoryClassifier(L, R, w, z)*. The object fit is: traditional target ordered-product ledger-chain endpoint replacement; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; stability by two raw-visible classifier exposures plus symmetry and transitivity; ledger two composable* ProdHistoryLedgerPolicy *witnesses; core theorem this endpoint equivalence; exactness classifier-invariant endpoints of the ledger chain; standard bridge displayed ordered-pair ledger chain; status* ProofTarget*.*

Proof. Apply Theorem 34.98 to the two ledger witnesses to obtain ProdHistoryClassifier(L, R, ρ, w). For the forward implication, reverse this chain classifier by Theorem 34.20, then compose it with the assumed classifier from ρ to z using Theorem 34.22, yielding ProdHistoryClassifier(L, R, w, z). For the reverse implication, compose the chain classifier from ρ to w with the assumed classifier from w to z by Theorem 34.22. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy two step classifier endpoint equivalence`

Theorem 34.100 (Prod ledger component exposure)**.** *If* ProdHistoryLedgerPolicy(L, R, ρ, v) *holds, then there exist histories* l *and* r *such that* $L(l)$*,* $R(r)$*,* Cont(l, r, ρ)*,* hsame(ρ, v)*,* ProdHistoryCarrier(L, R, v)*, and* ProdHistoryClassifier(L, R, ρ, v) *all hold.*

Proof. Unfold the ledger into the raw product carrier for ρ and the stored raw-visible witness hsame(ρ, v). Unfold the raw carrier to expose component histories l and r with $L(l)$, $R(r)$, and Cont(l, r, ρ). Theorem 34.92 supplies ProdHistoryCarrier(L, R, v), and Theorem 34.93 supplies ProdHistoryClassifier(L, R, ρ, v). These witnesses give the stated component exposure. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy component exposure`

Theorem 34.101 (Product ledger visible empty component exposure)**.** *Let* L, R : BHist $\to$ Prop *be source carriers and let* ρ : BHist *be a raw history. If* ProdHistoryLedgerPolicy$(L, R, \rho,$ Empty$)$ *holds, then there exist* l, r : BHist *such that* $L(l)$*,* $R(r)$*,* $l$hsameEmpty*,* $r$hsameEmpty*, and* ProdHistoryClassifier$(L, R, \rho,$ Empt *all hold. The object fit is: traditional target ordered-product ledger at the empty visible endpoint; BEDC source* BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; stability by raw-visible ledger exposure and empty-result carrier inversion; ledger* ProdHistoryLedgerPolicy*; core theorem this empty-endpoint component exposure; exactness component readback at* Empty*; standard bridge displayed ordered-pair ledger boundary; status* ProofTarget*.*

Proof. Apply Theorem 34.92 to the ledger witness. This places the visible endpoint Empty in ProdHistoryCarrier$(L, R,$ Empty$)$.

Apply Theorem 34.5 to that visible carrier. The result supplies component histories l and r with $L(l)$, $R(r)$, $l$hsameEmpty, and $r$hsameEmpty.

Apply Theorem 34.93 to the same ledger witness. Repack the component witnesses together with the raw-visible classifier to obtain the stated empty-boundary exposure. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerPolicy visible empty component exposure`

Theorem 34.102 (Product ledger chain transitivity)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. If* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, v)$ *and* $\mathsf{ProdHistoryLedgerChain}_{L,R}(v, z)$ *hold, then* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *holds.*

Proof. Induct on the first ledger-chain derivation. A one-ledger derivation is a displayed $\mathsf{ProdHistoryLedgerPolicy}(L,$ composing it with the second chain is exactly the cons constructor. In the cons case, apply the induction hypothesis to concatenate the tail chain with the second chain, then reapply the leading displayed ledger by the cons constructor. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain trans`

Theorem 34.103 (Product ledger-chain boundary classifier decomposition)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, and let* $\rho, z : \mathsf{BHist}$*. If* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *holds, then both boundary clauses hold:*

1. *either* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, z)$ *holds, or there exists* $v : \mathsf{BHist}$ *such that* $\mathsf{ProdHistoryLedgerPolicy}(L, R,$ *and* $\mathsf{ProdHistoryClassifier}(L, R, v, z)$ *hold;*
2. *either* $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, z)$ *holds, or there exists* $v : \mathsf{BHist}$ *such that* $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$ *and* $\mathsf{ProdHistoryLedgerPolicy}(L, R, v, z)$ *hold.*

The object fit is: traditional target finite ordered-product provenance; BEDC source BHist*,* ProdHistoryLedgerPolicy*, and* ProdHistoryLedgerChain*; carrier* ProdHistoryCarrier*; classifier* ProdHistoryClassifier*; ledger finite product ledger chain; core theorem boundary decomposition; exactness endpoint classifier preservation.*

Proof. For the head boundary clause, perform cases on the finite ledger-chain constructor. The one-step constructor is exactly $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, z)$.

In the cons constructor, the displayed first ledger has the form $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$ and the tail has the form $\mathsf{ProdHistoryLedgerChain}_{L,R}(v, z)$. Apply Theorem 34.55 to the tail; its envelope part gives $\mathsf{ProdHistoryClassifier}(L, R, v, z)$, so the witness v exposes the first concrete ledger step and the classifier tail consumed by readback.

For the terminal boundary clause, induct on the chain. The base case again is the displayed one-step ledger $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, z)$.

In the cons step, first apply Theorem 34.93 to the leading policy $\mathsf{ProdHistoryLedgerPolicy}(L, R, \rho, v)$, obtaining $\mathsf{ProdHistoryClassifier}(L, R, \rho, v)$. If the tail is a one-step ledger, the same v is the terminal-boundary witness. If the inductive tail has already exposed a terminal ledger $\mathsf{ProdHistoryLedgerPolicy}(L, R, u, z)$ with classifier prefix $\mathsf{ProdHistoryClassifier}(L, R, v, u)$, compose the leading classifier with that prefix by classifier transitivity to obtain $\mathsf{ProdHistoryClassifier}(L, R, \rho, u)$. The witness u exposes the last concrete ledger step and the accumulated classifier prefix. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain boundary classifier decomposition`

Theorem 34.104 (Product ledger chain concatenation preserves public endpoints)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. If*

$\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, v)$ *and* $\mathsf{ProdHistoryLedgerChain}_{L,R}(v, z)$ *hold, then* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *holds, and for every* $w : \mathsf{BHist}$ *both public componentwise endpoint clauses hold:*

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(\rho, w) \iff \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(z, w)$$

and

$$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, \rho) \iff \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, z).$$

The object fit is: traditional target finite ordered-product ledger concatenation; BEDC source BHist *with* ProdHistoryLedgerPolicy *and* ProdHistoryLedgerChain*; carrier* ProdHistoryCarrier *plus displayed* ProdPairRep*; classifier* ProdComponentHistoryClassifier*; stability finite ledger-chain closure; ledger concatenated displayed-pair provenance; core theorem closure plus endpoint invariance; exactness component readback invariant across concatenated chains; standard bridge concatenated ordered-pair provenance; status* ProofTarget.

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain concatenation componentwise endpoints`

Proof. Apply Theorem 34.102 to the two finite ledger chains to obtain $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$.

With the concatenated chain in hand, invoke Theorem 34.58 directly. The theorem depends only on the finite product ledger chain, so it yields the two public componentwise endpoint equivalences for every test history w. □

Theorem 34.105 (Product public certificate with ledger-chain component readback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. *With the named displayed pair ledger* $\mathsf{ProdDisplayedPairLedger}_{L,R}$ *from Definition 34.50, the following two clauses hold:*

1. *the componentwise public certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdDisplayedPairLedger}_{L,R}, \mathsf{ProdCompon}$$

2. *for all histories* $\rho, z, l, r : \mathsf{BHist}$, *if* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *and* $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$ *hold, then* $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$ *holds and there exist* $l', r' : \mathsf{BHist}$ *such that*

$$\mathsf{ProdPairRep}_{L,R}(z, l', r') \quad \text{and} \quad l \sim_L l' \quad \text{and} \quad r \sim_R r'.$$

The object fit is: traditional target ordered-product public certificate with finite ledger-chain component readback; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier *plus displayed* ProdPairRep*; classifier componentwise source classifiers; stability by envelope closure; ledger* ProdHistoryLedgerChain*; core theorem this combined public-certificate and readback package; exactness envelope classifier and final displayed component witnesses; standard bridge ordered-pair provenance; status* ProofTarget.

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain public certificate component readback`

Proof. The public certificate clause is Theorem 34.61 applied to cert_L, cert_R, and $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$, with the named displayed pair ledger from Definition 34.50.

For the ledger-chain readback clause, fix $\rho, z, l, r : \mathsf{BHist}$ and assume $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ together with $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$. Theorem 34.55 applied to the finite chain supplies $\mathsf{ProdHistoryClassifier}(L, R, \rho$ which is the envelope-classifier part of the conclusion.

It remains to expose the final displayed components. Apply Theorem 34.57 to the same chain and initial displayed pair representation, using the coherence witness. This gives $l', r' : \mathsf{BHist}$ with $\mathsf{ProdPairRep}_{L,R}(z, l', r')$, $l \sim_L l'$, and $r \sim_R r'$; conjoining these witnesses with the envelope classifier gives the required public-object ledger guarantee. □

Theorem 34.106 (Product public certificate with endpoint-stable ledger-chain readback)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$, $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, *and* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. *Assume also that both source classifiers are* hsame*-sound:*

$$\forall x\, y.\ x \sim_L y \to \mathsf{hsame}(x, y)$$

and

$$\forall x\, y.\ x \sim_R y \to \mathsf{hsame}(x, y).$$

For all histories $\rho, z, l, r : \mathsf{BHist}$, *if* $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$ *and* $\mathsf{ProdPairRep}_{L,R}(\rho, l, r)$ *hold, then the following three clauses hold:*

1. *the componentwise public certificate*

   $$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdDisplayedPairLedger}_{L,R}, \mathsf{ProdCompon}$$

2. $\mathsf{ProdHistoryClassifier}(L, R, \rho, z)$ *holds, and there exist* $l', r' : \mathsf{BHist}$ *such that*

   $$\mathsf{ProdPairRep}_{L,R}(z, l', r') \quad \textit{and} \quad l \sim_L l' \quad \textit{and} \quad r \sim_R r';$$

3. *for every* $w : \mathsf{BHist}$, *both endpoint-replacement clauses hold:*

   $$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(\rho, w) \Longleftrightarrow \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(z, w)$$

   and

   $$\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, \rho) \Longleftrightarrow \mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}(w, z).$$

The object fit is: traditional target ordered-product public certificate with finite endpoint-stable ledger-chain readback; BEDC source BHist *with* Cont *and* hsame*; carrier* ProdHistoryCarrier *plus displayed* ProdPairRep*; classifier componentwise source classifiers; stability by ledger-chain endpoint equivalence; ledger* ProdHistoryLedgerChain*; core theorem this combined public certificate, displayed component readback, and endpoint-stable component classifier; exactness final displayed component witnesses and endpoint-invariant public classification; standard bridge ordered-pair provenance; status* ProofTarget.

Proof. First apply Theorem 34.105 with cert_L, cert_R, and $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$. Its certificate clause gives the displayed-pair semantic certificate, and its ledger-chain readback clause applied to the chosen ρ, z, l, r gives the product history classifier together with the final displayed witnesses l', r'.

It remains to make the public componentwise classifier stable at the two ledger-chain endpoints. Apply Theorem 34.58 to $\mathsf{ProdHistoryLedgerChain}_{L,R}(\rho, z)$. This supplies, for every test history w, both endpoint-replacement equivalences. Conjoin these with the certificate and displayed readback conclusions from the first step. □

Lean4 ✓ `BEDC.Derived.ProdUp.ProdHistoryLedgerChain public certificate endpoint invariance`

Theorem 34.107 (Concrete product history-envelope semantic certificate)**.** *For source-history predicates* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, assume* $\exists l : \mathsf{BHist}.\ L(l)$ *and* $\exists r : \mathsf{BHist}.\ R(r)$*. Then*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryClassifie}$$

holds. This is the checked ordered-product history-envelope certificate; source certificates supply the two inhabited-carrier hypotheses through their fields.

Proof. Choose the left and right carrier witnesses first. BEDC continuation joins them into a product history. The history-envelope classifier stores the two endpoint carrier witnesses and an hsame witness between product histories, so the certificate fields follow by hsame reflexivity, symmetry, transitivity, and carrier transport. When source certificates are present, their carrier-inhabited fields provide the same witnesses. □

Lean4 ✓ `BEDC.Derived.ProdUp.prod history semantic name certificate`

Theorem 34.108 (ProdUp obligation surface package)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history predicates and let* $\sim_L, \sim_R : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source carrier witnesses for* L *and* R*, source certificates* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*, componentwise* hsame*-soundness for* $\sim_L$ *and* $\sim_R$*, and the displayed-pair coherence* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. Then the ProdUp surface over* BHist*,* Cont*,* hsame*, and* Pkg *supplies all finite-data obligations listed below, using only BEDC histories and product continuations:*

1. *an inhabited product-history carrier* $\mathsf{ProdHistoryCarrier}(L, R)$*;*
2. *componentwise classifier stability for the displayed product classifier induced by the source classifiers;*
3. *raw-visible ledger exactness for* $\mathsf{ProdHistoryLedgerPolicy}(L, R)$*;*
4. *endpoint-stable ledger-chain readback for* $\mathsf{ProdHistoryLedgerChain}_{L,R}$*;*
5. *the semantic* NameCert *surface for the product history envelope and its public displayed-pair classifier.*

Thus the finite-data consumer may read the product package through BEDC histories, continuation, history sameness, and package evidence, without importing host ordered pairs as primitive data.

Proof. Choose source carrier witnesses for L and R. The ProdHistoryCarrier clause follows by joining the witnesses through Cont, and the same carrier exposure is available from the ledger side by Theorem 34.92.

For the raw-visible ledger surface, apply Theorem 34.93. Its transport stability is Theorem 34.94, so the ledger reads the raw history and the visible history through hsame rather than through any host pair object.

For the finite ledger-chain public surface, apply Theorem 34.106. It packages the componentwise public certificate, the final displayed readback, and the endpoint-stable classifier clauses for every test history.

Finally, apply Theorem 34.107 to obtain the semantic history-envelope certificate. Together with the public displayed-pair certificate supplied by the endpoint-stable ledger-chain theorem, these clauses are exactly the carrier, classifier, ledger, endpoint, and semantic-certificate surface consumed by the finite-data phase. □

Corollary 34.109 (Product history envelope is licensed, not primitive)**.** *The checked theorem licenses only* ProdHistoryCarrier *with* ProdHistoryClassifier *under explicit left and right carrier witnesses. The public ordered-product certificate uses the history-level* $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$ *induced by* ProdPairRep *and source classifiers, together with displayed component exposure,* hsame*-sound descent, and ledger fields from this chapter.*

Theorem 34.110 (Prod finite-data sibling lattice)**.** *Every* $\mathsf{Prod}^{\uparrow}$ *pair-envelope read consumes the finite-data sibling lattice before it exports the ordered pair certificate. The lattice consists of the* $\mathsf{Bool}^{\uparrow}$ *tag discipline of chapter 32, the* $\mathsf{Sum}^{\uparrow}$ *branch partition of chapter 35, and the transport surfaces supplied by chapter 36 and chapter 33. A public product consumer may then read the ordered pair only through the displayed product history carrier, componentwise classifier, displayed-pair ledger, source* NameCert *rows,* hsame *transport,* Cont *continuation, and* Pkg *provenance.*

Proof. Start with the product public finite-data exit package, Definition 34.118. It lists the rows available to the product consumer: the componentwise semantic certificate, displayed-pair ledger endpoints, descent to the product history envelope, source certificates, hsame, Cont, and Pkg provenance.

The sibling lattice fixes how those rows are read as finite data. The BoolUp tag discipline supplies the local tag behavior; the SumUp branch partition supplies the disjoint branch read; and the ListUp and OptionUp transport surfaces supply finite repetition and possible absence without adding host tuple equality.

Apply consumer universality, Definition 34.119. Any public product read invariant under the package rows factors through that finite-data exit package, so the ordered pair certificate is exported only after the sibling lattice has supplied its tag, branch, list, and option inputs. □

Theorem 34.111 (Prod finite-data sibling lattice route)**.** *Every accepted* $\mathsf{Prod}^{\uparrow}$ *carrier consumes* $\mathsf{Bool}^{\uparrow}$ *endpoint discrimination,* $\mathsf{Sum}^{\uparrow}$ *tagged alternatives,* $\mathsf{Option}^{\uparrow}$ *absence and presence rows, and* $\mathsf{List}^{\uparrow}$ *finite-spine rows through componentwise* NameCert *projection before exporting the ordered-pair certificate. The route cites chapter 32, chapter 35, chapter 33, and chapter 36 as sibling finite-data surfaces, not as host product projections.*

Proof. Start with the product finite-data sibling lattice, Definition 34.110. It places product reads behind Bool endpoint tags, Sum branch partitions, Option possible-absence surfaces, and List finite transport before the ordered pair certificate is visible.

Apply the option-sum component route, Definition 34.113. Component reads pass through the optional and tagged alternatives as displayed sibling rows rather than as host projections from a product object.

Finally use the public finite-data exit package. It exports the ordered pair through the displayed product history carrier, componentwise classifier, displayed-pair ledger, source NameCert rows, hsame transport, Cont continuation, and Pkg provenance. Thus the sibling lattice is consumed by componentwise NameCert projection and by no unrecorded tuple oracle. □

Theorem 34.112 (ProdUp BHist pair projection determinacy)**.** *For every accepted* $\mathsf{Prod}^{\uparrow}$ *public pair read, the* BHist *projection is determined by the two displayed component rows together with the product component ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ *row. The finite-data siblings of chapter 32, chapter 35, and chapter 36 may consume the read only after those product rows have fixed the two projections. Thus the product packet cannot export a tag-only* $\mathsf{Sum}^{\uparrow}$ *branch, optional none row, or* $\mathsf{List}^{\uparrow}$ *spine without passing through the corresponding sibling* NameCert *surface.*

Proof. Start from the product componentwise interface. The accepted pair read exposes two component rows and the displayed product ledger before any finite-data consumer receives a public projection.

The Bool sibling supplies only endpoint discrimination for the product route, while the Sum and List siblings supply branch and spine consumers after the component rows have been named. Their NameCert surfaces therefore consume already displayed product projections; they do not replace the pair by a detached sum tag, absent option row, or list spine.

Transport through hsame, replay through Cont, provenance through Pkg, and the local $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ row preserve the same two projections. Hence the public BHist read is determined by the product component rows and has no independent finite-data export outside the sibling certificates. □

Theorem 34.113 (Prod option-sum component sibling route)**.** *Every accepted* $\mathsf{Prod}^{\uparrow}$ *component read factors through the finite-data sibling lattice containing chapter 33 and chapter 35 before a downstream component consumer can treat the pair as a case-split or optional projection. The route reads the product carrier through the displayed component rows, then passes through the* $\mathsf{Option}^{\uparrow}$ *possible-absence surface and the* $\mathsf{Sum}^{\uparrow}$ *branch surface, and only then exposes the component to a finite-data consumer.*

Proof. Start with the product public finite-data exit package of Definition 34.118. It exposes the componentwise semantic certificate, displayed-pair ledger endpoints, descent to the product history envelope, source NameCert rows, hsame, Cont, and Pkg provenance.

Next apply the finite-data sibling lattice of Definition 34.110. That lattice requires the component read to pass through the finite-data siblings before the ordered pair is exported.

The relevant sibling route has two finite-data faces. The $\mathsf{Option}^{\uparrow}$ chapter supplies the possible-absence projection discipline, while the $\mathsf{Sum}^{\uparrow}$ chapter supplies the branch partition used when the downstream consumer treats a component as a case split.

Since the public product package has no host tuple equality and no outside-ledger projection, a component consumer receives only the product rows already routed through those two siblings. Thus every accepted component read factors through the option and sum surfaces before it can be used as an optional projection or case-split input. □

The finite-data readback for $\mathsf{Prod}^{\uparrow}$ consumes the $\mathsf{Sum}^{\uparrow}$ route of chapter 35 and the $\mathsf{Option}^{\uparrow}$ route of chapter 33 componentwise: left and right product coordinates retain their sibling finite-data witnesses, $\mathsf{Bool}^{\uparrow}$ tag discrimination keeps the endpoint cases separated, and the $\mathsf{List}^{\uparrow}$ spine-readback route remains an external ledger rather than a hidden product coordinate.

Proposition 34.114 (Prod finite-data dependency lattice read)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *finite-data read factors through the sibling lattice of chapter 32, chapter 33, chapter 35, and chapter 36 before exporting the ordered-pair* NameCert *rows. The factorization uses Definition 34.110 to expose the tag, possible-absence, branch, and finite-transport surfaces, then uses Definition 34.113 to route component reads through the option and sum faces before a downstream consumer receives the pair certificate.*

Proof. Apply Definition 34.110. It places every public product read behind the BoolUp tag discipline, SumUp branch partition, ListUp finite transport, and OptionUp possible-absence surface.

Next apply Definition 34.113. Component reads must pass through the option and sum sibling route before a consumer treats them as optional projections or case-split inputs.

The ordered-pair NameCert rows are therefore exported only after the four finite-data siblings have supplied the dependency lattice. No host tuple equality or outside component projection is added to the public $\mathsf{Prod}^{\uparrow}$ read. □

Theorem 34.115 (Prod finite-data dependency lattice)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *finite-data read is positioned between the endpoint discrimination of chapter 32, the partiality surface of chapter 33, the tagged branching surface of chapter 35, and the spine-iteration surface of chapter 36. The product packet exports only the ordered-pair envelope after those sibling routes have supplied the finite-data lattice, and it keeps payload carriers, branch tags, list spines, component ledgers,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows inside the existing* $\mathsf{Prod}^{\uparrow}$ *envelope.*

Proof. Start with the componentwise product surface. The public product read exposes left and right component projections through the displayed product history carrier, the componentwise classifier, the displayed-pair ledger, and source NameCert rows.

The sibling lattice supplies the finite-data routes around that envelope. Bool supplies endpoint discrimination, Option supplies possible absence, Sum supplies tagged branching, and List supplies spine iteration; none of these rows replaces the product component ledgers or turns them into host-level tuple equality.

Finally apply the finite-data readiness rows already recorded in the product chapter. They route every public finite-data exit through componentwise projection, hsame transport, Cont replay, Pkg provenance, and the local product naming row, so the read cannot escape the existing $\mathsf{Prod}^{\uparrow}$ envelope. □

Theorem 34.116 (Prod product carrier sibling dependency)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *product-carrier read depends on the finite-data sibling route that places Bool endpoint discrimination from chapter 32, Sum branch exposure from chapter 35, and List spine readback from chapter 36 around the componentwise product envelope. The product packet then exports only ordered component carriers, component ledgers,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows; it does not become an isolated host pair or a detached finite-data object.*

Proof. Project the existing $\mathsf{Prod}^{\uparrow}$ componentwise carrier. Its public read exposes a left component, a right component, the component ledgers, and the structural rows that transport and replay those components.

The sibling finite-data route supplies the surrounding classifiers. Bool fixes endpoint discrimination, Sum supplies tagged branching, and List supplies spine readback for repeated finite data. These rows are dependencies of downstream product consumers, but they do not replace the product envelope.

Thus every product-carrier exit remains componentwise. The exported object is the same $\mathsf{Prod}^{\uparrow}$ carrier with sibling finite-data dependencies recorded in the ledger, not an isolated host pair. □

Theorem 34.117 (Prod option-sum sibling lattice readback)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *ordered-pair read consumes the finite-data sibling lattice of chapter 33 and chapter 35 only through the product component ledgers. The public route first reads the product componentwise interface, displayed-pair ledger, source* NameCert *rows,* hsame *transport,* Cont *continuation, and* Pkg *provenance; it then uses the option and sum sibling surfaces as finite-data consumers of those component rows.*

The product pair is not a renamed option case and not a renamed sum tag. It remains the two-component ledger supplied by $\mathsf{Prod}^{\uparrow}$*, and downstream finite-data consumers may read it only through the displayed component ledgers and the existing ProdUp componentwise interface.*

Proof. Begin with the product componentwise interface. Theorems 34.79 and 34.84 identify the public ordered-pair read with component histories, component classifiers, and displayed pair ledgers inside the product packet.

Next use the product finite-data sibling lattice, Definition 34.110. It places product reads behind the finite-data lattice while keeping the ordered-pair certificate on the product carrier rows.

Now apply the option-sum component sibling route, Definition 34.113. The option and sum chapters supply possible-absence and branch-partition consumers for already displayed component rows; they do not replace the pair ledger by an option tag or a sum branch.

Finally use the public finite-data exit package, Definition 34.118. Since every downstream finite-data read factors through the componentwise certificate, displayed-pair ledger, source certificates, hsame, Cont, and Pkg, the sibling lattice is read back through the product component ledgers and through no host tuple equality or external projection. □

Theorem 34.118 (Prod public finite-data exit package)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and suppose* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$ *are the source certificate rows. Assume also a witness of* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$*. Then the public product finite-data exit package consists of the following rows.*

1. *The componentwise classifier carries the semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdHistoryCarrier}(L, R), \mathsf{ProdComponent}$$

2. *For any displayed componentwise classifier witness between histories* $h, k : \mathsf{BHist}$*, the left and right displayed-pair ledger endpoints are readable as*

$$\mathsf{ProdDisplayedPairLedger}_{L,R}(h) \wedge \mathsf{ProdDisplayedPairLedger}_{L,R}(k).$$

3. *For the same histories* $h, k : \mathsf{BHist}$*, every componentwise classifier witness descends to the checked product history envelope classifier*

$$\mathsf{ProdHistoryClassifier}(L, R, h, k).$$

4. *The obligation surface exposes exactly the source* NameCert *rows, the componentwise classifier row, the displayed-pair ledger row, the descent row, the* hsame *and* Cont *transport rows, and the* Pkg *provenance rows named by Definition 34.108.*

Consequently the public product object reads only BHist*,* hsame*,* Cont*, the displayed-pair ledger, the source certificates, and* Pkg *provenance rows.*

Lean4 ✓ `BEDC.Derived.ProdUp.ProdPublicFiniteDataExitPackage`

Proof. First apply Definition 34.60 to the two source certificates and the $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ witness. This supplies the componentwise semantic certificate row.

Next fix histories $h, k : \mathsf{BHist}$ and a componentwise classifier witness between them. Apply Definition 34.62 to read both displayed-pair ledger endpoints.

Then apply Definition 34.79 with the same componentwise witness and the source soundness rows. This is the descent row from the componentwise classifier into the checked product history envelope.

Finally read the obligation rows from Definition 34.108. The four rows above mention no host tuple equality and no data beyond the listed BHist carrier, hsame, Cont, displayed-pair ledger, source certificate rows, and Pkg provenance, so they repackage exactly the advertised public finite-data product surface. □

Theorem 34.119 (Product finite-data public exit consumer universality)**.** *Let* $L, R : \mathsf{BHist} \to \mathsf{Prop}$ *be source-history carriers, let* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be their source classifiers, and let* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$ *be the source certificate rows. Assume a displayed witness of* $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$.

Every public finite-data consumer of a product history whose reads are invariant under the componentwise semantic certificate, the displayed-pair ledger endpoints, the descent-to-envelope row, hsame, Cont, *and* Pkg *provenance factors through the package displayed in Definition 34.118. The factorization reads only the package rows: source* NameCert *rows, the componentwise classifier certificate, displayed-pair ledger endpoints, descent to the product history envelope,* hsame *transport,* Cont *continuation, and* Pkg *provenance.*

No step of the factorization exposes host tuple equality, outside-ledger component projections, or a quotient of pair representatives.

Proof. Apply Definition 34.118. It displays the candidate rows available to any public finite-data read of the product history: the semantic componentwise certificate, displayed-pair ledger endpoints, descent to the envelope classifier, source certificate rows, hsame, Cont, and Pkg provenance.

Next apply Definition 34.130. The consumer invariance assumptions force every public read to be one of those displayed rows; there is no separate host tuple row and no projection outside the displayed product ledger.

It remains to identify endpoint reads. Use Definition 34.131. Any endpoint datum read by the consumer is already the displayed-pair ledger endpoint supplied by the public package, so endpoint access cannot introduce a second product representation or a quotient of pair representatives.

Thus all public consumer reads are exhausted by the rows of Definition 34.118, and the consumer factors through that package. □

Theorem 34.120 (Prod finite-data sibling route)**.** *Every public finite-data consumer that spans* $\mathsf{Prod}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, *and* $\mathsf{List}^{\uparrow}$ *must read the product component through the displayed-pair ledger of Definition 34.118. The same consumer must separately respect the tagged-envelope route of chapter 35 and the spine route of chapter 36. Thus the three constructor surfaces remain sibling finite-data envelopes: product reads expose pair display and componentwise classification, sum reads expose visible tags and branch classification, and list reads expose spine display and prefix classification.*

Consequently no such consumer may collapse the product, sum, and list envelopes into host tuple equality, host list equality, a common untagged carrier, or a generic finite-data quotient.

Proof. Project the product package rows. By Definition 34.118, a public product read sees the source certificate rows, componentwise classifier certificate, displayed-pair ledger endpoints, descent to the product history envelope, hsame transport, Cont replay, and Pkg provenance.

Now read the sibling constructor routes. The sum chapter supplies a visible tagged-envelope route, while the list chapter supplies a spine route; these routes carry tag and spine ledger rows that are not product displayed-pair rows.

The consumer therefore has three distinct finite-data boundaries available at once. Since pair display, tag display, and spine display are different ledger coordinates, replacing them by host tuple/list equality or by a common quotient would erase the route by which each constructor was admitted. □

Theorem 34.121 (Finite-data product, sum, and list envelope separation)**.** *Any public finite-data consumer spanning* $\mathsf{Prod}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, *and* $\mathsf{List}^{\uparrow}$ *must preserve three distinct envelope rows. The product route reads through the displayed-pair ledger and the componentwise classifier of Definition 34.118.*

The sum route reads through the visible tag and branch classifier of Definition 35.41. The list route reads through the spine bridge and prefix classifier of Definition 36.145.

Consequently no public consumer may replace these rows by host tuple equality, a generic finite-data quotient, or a common untagged carrier. Such a replacement would erase the ledger/classifier boundary that distinguishes pair display, tag display, and spine display.

Proof. Start with the product public exit package and its consumer universality. By Definition 34.118 and Definition 34.119, every public product read factors through source certificate rows, the displayed-pair ledger, the componentwise classifier, descent to the product history envelope, hsame, Cont, and Pkg provenance.

For sums, use the checked tag-separation surface. The theorem Definition 35.41 forces mixed visible tags to be impossible, while the same-tag stability rows keep each accepted read inside the branch classifier attached to its tag.

For lists, use the spine bridge exactness theorem. The displayed spine row and prefix classifier identify the accepted list route through the spine bridge, not through a detached host sequence or an untagged finite-data carrier.

Inspecting the three rows gives different ledger/classifier boundaries: products expose pair display plus componentwise classification, sums expose tag display plus branch classification, and lists expose spine display plus prefix classification. A single host tuple equality, generic quotient, or untagged carrier would identify these boundaries, so it is not a public finite-data consumer for the three constructors at once. □

Theorem 34.122 (Prod Bool endpoint-discriminator route)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *component-route read uses the two endpoint rows licensed by chapter 32 only as a finite discriminator for component position. The left and right positions are tags for routing through the displayed-pair ledger; the payload carriers, component classifiers, source* NameCert *rows,* hsame *transports,* Cont *replays, and* Pkg *provenance remain owned by* $\mathsf{Prod}^{\uparrow}$ *and by the source certificates named in Definition 34.118.*

Consequently a public product consumer cannot replace the componentwise interface by an untagged finite-data quotient or by a host ordered-pair primitive.

Proof. Start with the componentwise interface. The displayed product route reads components through the product history classifier and the displayed-pair ledger, as in Definition 34.62 and Definition 34.79. These rows identify which component position is being read, but the component payload still comes from the source carriers and source classifiers.

Next apply the public finite-data exit package of Definition 34.118. It lists exactly the rows available to a public product consumer: the componentwise semantic certificate, displayed-pair ledger endpoints, descent to the product history envelope, source certificate rows, hsame, Cont, and Pkg provenance.

The two $\mathsf{Bool}^{\uparrow}$ endpoint rows therefore serve only as the finite discriminator for left versus right component position. Since all payload, classifier, transport, replay, and provenance rows remain the product rows displayed above, there is no untagged finite-data quotient and no host ordered-pair primitive available to the consumer. □

Theorem 34.123 (Prod Bool endpoint readback exhaustion)**.** *Let a public* $\mathsf{Prod}^{\uparrow}$ *component read have endpoint discriminator rows agreeing with the left and right* $\mathsf{Bool}^{\uparrow}$ *rows used in Definition 34.122. Then the readback recovers only the displayed component payload, the source* NameCert *rows, the componentwise* hsame *transports, the displayed* Cont *replays, and the* Pkg *provenance listed by Definition 34.118. It cannot introduce an untagged quotient of product data or a host pair coordinate outside the public* $\mathsf{Prod}^{\uparrow}$ *packet.*

Proof. Apply Definition 34.122. The two $\mathsf{Bool}^{\uparrow}$ rows identify only the left or right endpoint position in the displayed product route. They do not carry a product payload, a product classifier, or an independent ordered-pair coordinate.

Next use Definition 34.118. That theorem exhausts the public read surface by the displayed component payload rows, source NameCert rows, hsame transports, Cont replays, and Pkg provenance.

Combining the two statements leaves no unread endpoint data. The discriminator chooses the displayed component route, and the exit package lists every row that can be recovered from that route. Hence a quotient product object or host pair coordinate has no public readback source. □

Theorem 34.124 (Prod hub Bool sibling audit link)**.** *The public* $\mathsf{Prod}^{\uparrow}$ *ordered-pair read consumes* $\mathsf{Bool}^{\uparrow}$ *from chapter 32 only as the finite endpoint discriminator before any component row is exposed. The Bool row distinguishes the left and right endpoint positions of the displayed-pair ledger; payload carriers, component classifiers, source* NameCert *rows,* hsame *transports,* Cont *replays, and* Pkg *provenance remain the product rows named by the* $\mathsf{Prod}^{\uparrow}$ *packet.*

Proof. Read the product hub through its child rows. The componentwise interface and displayed-pair ledger expose endpoint positions before they expose payload components.

Apply the Bool endpoint-discriminator route. It identifies the $\mathsf{Bool}^{\uparrow}$ dependency as the finite left-versus-right tag discipline for those endpoint positions.

The public finite-data exit package then supplies the rest of the product route. Since payload, classifier, transport, replay, and provenance rows are still owned by the product and source certificates, the Bool sibling is a genuine horizontal dependency but not a replacement for the $\mathsf{Prod}^{\uparrow}$ carrier. □

Theorem 34.125 (Prod hub finite-data readiness index)**.** *The finite-data public surface indexed by the* $\mathsf{Prod}^{\uparrow}$ *hub is exhausted by five already displayed routes: the product history envelope, the componentwise interface, the ledger certificate, the* $\mathsf{Option}^{\uparrow}$–$\mathsf{Sum}^{\uparrow}$ *sibling route, and the public finite-data exit package. Equivalently, every public product read factors through* BHist*, the componentwise classifier, displayed-pair ledger rows, source* NameCert *rows,* hsame*,* Cont*, and* Pkg *provenance before it reaches any finite-data consumer.*

The index has no extra host tuple equality, outside-ledger projection, generic finite-data quotient, or untagged carrier beyond the child-file surfaces named by the hub.

Proof. Begin with the history envelope. The carrier and classifier surfaces of Definition 34.1, Definition 34.107, and Definition 34.109 put the product read inside the BEDC history envelope rather than in a host tuple object.

Next use the componentwise interface. The descent and exactness rows of Definition 34.79, Definition 34.84, and Definition 34.61 make component reads pass through the displayed classifier and source certificate rows.

The ledger certificate supplies the finite display route. Apply Definition 34.50, Definition 34.62, and Definition 34.108; these expose the displayed-pair ledger, hsame transport, Cont continuation, and Pkg provenance rows.

Finally pass through the finite-data siblings and public exit package. The sibling coverage is Definition 34.110 and Definition 34.113; the public consumer boundary is Definition 34.118 and Definition 34.119. The finite-data sibling route is grounded in chapter 33, chapter 35, chapter 32, and chapter 36; these chapters supply the constructor-tag, branch, boolean, and spine surfaces consumed by the product public boundary. These rows exhaust the hub-indexed readiness surface, so no additional product consumer coordinate remains. □

Proposition 34.126 (Product, sum, and list finite-data lattice)**.** $\mathsf{Prod}^{\uparrow}$ *consumes the same finite-data constructor discipline as chapter 35 and chapter 36. Its componentwise carrier supplies paired projections,* $\mathsf{Sum}^{\uparrow}$ *supplies branch separation, and* $\mathsf{List}^{\uparrow}$ *supplies spine iteration; therefore any public finite-data read that routes through all three surfaces must preserve the product component ledgers before it may branch or iterate.*

Theorem 34.127 (Prod sum-projection route)**.** *For every accepted* $\mathsf{Prod}^{\uparrow}$ *packet, the two projection rows can be consumed by the* $\mathsf{Sum}^{\uparrow}$ *and* $\mathsf{Option}^{\uparrow}$ *finite-data routes only through the displayed product carrier, component projections, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ *rows. The* $\mathsf{Sum}^{\uparrow}$ *route of chapter 35 may read the projected component as a branch payload only after the product carrier has exposed the ordered pair, and the* $\mathsf{Option}^{\uparrow}$ *route of chapter 33 may read the same component as a possible-presence payload only after the product projection row has been named.*

Proof. Start with the product public finite-data exit package of Definition 34.118. It exposes the displayed pair, component projections, semantic certificate, hsame rows, Cont rows, Pkg provenance, and local naming rows before any external finite-data consumer can read a component.

Next apply the existing sibling lattice through Definition 34.113 and Definition 34.126. These results force branch-style and optional-style reads to pass through the same product component rows rather than through a host tuple projection.

Finally consume the sibling surfaces. The $\mathsf{Sum}^{\uparrow}$ chapter accounts for the branch route, while the $\mathsf{Option}^{\uparrow}$ chapter accounts for the possible-presence route. Since the accepted product packet exports no independent projection oracle outside its carrier, transport, replay, provenance, and NameCert rows, both finite-data consumers receive only the displayed ProdUp route. □

Theorem 34.128 (Prod Bool-Sum-List route audit)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *finite-data read factors through the Bool endpoint discriminator, Sum branch surface, List spine surface, and Option absence surface before it exports the product component ledger. Equivalently, the product consumer reaches component data only after reading chapter 32, chapter 35, chapter 36, chapter 33, the finite-data sibling lattice of Definition 34.110, and the option-sum component route of Definition 34.113.*

Proof. Start with Definition 34.110. It records that the ordered pair certificate is not read as an isolated host tuple: the Bool tag discipline, Sum branch partition, List transport surface, and Option possible-absence surface all precede the public product exit.

Next apply Definition 34.113. It forces every component read through the Option absence surface and the Sum branch surface before the component can be used as an optional projection or case-split input.

The Bool endpoint discriminator supplies the finite endpoint separation, while the List spine surface supplies the finite repeated-data route used by product component ledgers. These routes are the same sibling surfaces cited by the local lattice rows, so they do not add host tuple equality or an outside projection.

Therefore every public finite-data read of $\mathsf{Prod}^{\uparrow}$ reaches the product component ledger only after the Bool, Sum, List, and Option routes have been displayed. □

Theorem 34.129 (Prod finite-data scoped closure record)**.** *The componentwise public certificate, displayed-pair ledger endpoints, descent-to-envelope row,* hsame *and* Cont *transports, and* Pkg *provenance of Definition 34.118 bind the* $\mathsf{Prod}^{\uparrow}$ *finite-data scope entirely inside BEDC histories. More precisely, for source-history carriers* $L, R : \mathsf{BHist} \to \mathsf{Prop}$*, source classifiers* $\sim_L, \sim_R$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, source certificates* $\mathsf{cert}_L : \mathsf{NameCert}(L, \sim_L)$ *and* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*,*

and a $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *witness, the public finite-data package exposes only the componentwise* SemanticNameCert *row, the two displayed-pair ledger endpoints, the descent row into* $\mathsf{ProdHistoryClassifier}(L,R)$*, the* hsame *and* Cont *endpoint-transport rows, and the* Pkg *provenance rows named by the* $\mathsf{Prod}^{\uparrow}$ *obligation surface. No host tuple equality is part of the record.*

Proof. Apply Definition 34.118. Its first row gives the componentwise public certificate for $\mathsf{ProdHistoryCarrier}(L,R)$ under $\mathsf{ProdComponentHistoryClassifier}_{L,R,\sim_L,\sim_R}$, using exactly the two source NameCert rows and the displayed-pair coherence witness.

The second and third rows of that package project the displayed-pair ledger endpoints and the descent-to-envelope comparison for any componentwise classifier witness between histories h, k : BHist. The target of the descent is the checked envelope classifier $\mathsf{ProdHistoryClassifier}(L,R,h,k)$.

The fourth row identifies the remaining public data with the source certificate rows, the componentwise classifier row, the displayed-pair ledger row, the descent row, the hsame and Cont transport rows, and the Pkg provenance rows of Definition 34.108. The checked envelope certificate Definition 34.107 supplies the BEDC history envelope in which those rows land.

Thus the finite-data scope is closed by projections of the public package and the checked envelope certificate. Every named row is a row over BHist, hsame, Cont, NameCert, and Pkg, so the record contains no host tuple equality. □

Theorem 34.130 (Prod finite-data consumer row exhaustion)**.** *Any consumer of the public* $\mathsf{Prod}^{\uparrow}$ *finite-data package can read only the componentwise* SemanticNameCert *row, the displayed-pair ledger endpoints, descent into* ProdHistoryClassifier*, the* hsame/Cont *endpoint transports, the source* NameCert *rows, and the* Pkg *provenance. No host tuple equality, component projection outside the displayed ledger, or quotient of pair representatives is exported.*

Proof. Apply Definition 34.118. This gives the consumer-facing package rows: componentwise public certificate, displayed-pair ledger endpoints, descent to the product-history classifier, endpoint transports, source certificate rows, and provenance rows.

Next apply Definition 34.129 to identify the closed finite-data scope. That theorem says that the public rows are exactly the rows bound inside BEDC histories by the product finite-data record.

The absence claim follows by row exhaustion. Since those two packages list the entire consumer surface and neither package contains host tuple equality, outside-ledger component projection, or a quotient of pair representatives, no consumer can project such data from the public $\mathsf{Prod}^{\uparrow}$ finite-data package. □

Theorem 34.131 (Prod component ledger endpoint determinacy)**.** *Let* L, R : BHist $\to$ Prop *be source-history carriers, let* $\sim_L, \sim_R$: BHist $\to$ BHist $\to$ Prop *be source classifiers, let* cert_L : $\mathsf{NameCert}(L, \sim_L)$ *and* cert_R : $\mathsf{NameCert}(R, \sim_R)$ *be source rows, and let* coh : $\mathsf{ProdPairRepCoherent}_{L,R,\sim_L,\sim_R}$ *be a displayed pair-representative coherence witness. If two* $\mathsf{ProdHistoryLedgerChain}(L,R,\rho,z)$ *readbacks from the same raw product row* ρ *expose component pairs* (l,r) *and* (l',r') *through* ProdPairRep*, then the public finite-data package forces* $l \sim_L l'$ *and* $r \sim_R r'$*. Consequently the two visible endpoints are related by* $\mathsf{ProdHistoryClassifier}(L,R)$*, and no further host-level tuple equality is needed.*

Proof. Apply the displayed component readback to each of the two ledger-chain readbacks. Both readbacks start from the same raw product row ρ, so the only component witnesses they expose are the two displayed ProdPairRep pairs named in the ledger.

Use the coherence witness coh to compare those two displayed pairs componentwise. This yields the source classifier rows $l \sim_L l'$ and $r \sim_R r'$ inside the same BEDC history envelope as the source NameCert rows.

Finally repack the two source classifier rows through the product envelope classifier. The consumer-row exhaustion theorem identifies that envelope row as part of the public finite-data surface, so the endpoint comparison lands in $\mathsf{ProdHistoryClassifier}(L, R)$ and exports no outside-ledger component projection. □

Theorem 34.132 (Prod finite-data closure status record)**.** *The scoped row record of Definition 34.129, the consumer-row exhaustion theorem Definition 34.130, and the endpoint determinacy theorem Definition 34.131, together with the checked envelope certificate Definition 34.107, give the Phase Finite Data scoped closure surface for* $\mathsf{Prod}^{\uparrow}$*. The resulting surface is exactly the public finite-data exit package: componentwise semantic certificate rows, displayed-pair ledger endpoints, descent into* $\mathsf{ProdHistoryClassifier}(L, R)$*,* $\mathsf{hsame/Cont}$ *transport rows, source* $\mathsf{NameCert}$ *rows, and* Pkg *provenance.*

Proof. Project first through Definition 34.118. The finite-data scoped record identifies the rows exported by that package and places each row inside the BEDC history envelope rather than in host tuple equality.

Next use Definition 34.130. It says that a consumer of the package has no access to rows beyond the componentwise certificate, the displayed-pair ledger endpoints, the descent row, the endpoint transports, the source certificates, and the provenance rows.

The endpoint determinacy theorem then fixes the visible ledger endpoints by the source classifiers and repacks them through $\mathsf{ProdHistoryClassifier}(L, R)$. The checked envelope certificate Definition 34.107 supplies the semantic-name certificate row for that repacking.

Combining these projections gives the Phase Finite Data scoped closure surface for $\mathsf{Prod}^{\uparrow}$: every exported row is one of the public finite-data rows, every endpoint comparison lands in the checked product-history classifier, and no outside-ledger component projection or host tuple equality is exported. □

Theorem 34.133 (Prod finite-data public bridge consumer)**.** *Every Bool, Option, Sum, or List finite-data consumer of the public* $\mathsf{Prod}^{\uparrow}$ *package factors through the scoped record Definition 34.129, the consumer-row exhaustion theorem Definition 34.130, endpoint determinacy Definition 34.131, and the public finite-data exit package Definition 34.118. Its visible read is therefore exhausted by componentwise semantic certificate rows, displayed-pair ledger endpoints, descent into* $\mathsf{ProdHistoryClassifier}(L, R)$*,* $\mathsf{hsame/Cont}$ *transport, source* $\mathsf{NameCert}$ *rows, and* Pkg *provenance. No such consumer receives host tuple equality or an outside-ledger component projection.*

Proof. First project the public package of Definition 34.118. A Bool, Option, Sum, or List consumer reads the componentwise public certificate, the displayed-pair ledger endpoints, the descent row, the endpoint transports, the source certificate rows, and the provenance rows from that package.

Next apply the scoped record Definition 34.129. It places exactly those rows inside the BEDC history envelope and identifies the product-history classifier as the endpoint classifier available to finite-data consumers.

Then use consumer-row exhaustion and endpoint determinacy. The exhaustion theorem removes every row not listed by the public package, and endpoint determinacy routes visible component endpoints through the product-history classifier rather than through host tuple equality.

Thus the public finite-data bridge available to Bool, Option, Sum, and List consumers is precisely the displayed $\mathsf{Prod}^{\uparrow}$ package route. There is no remaining coordinate from which an outside-ledger projection can be read. □

Lemma 34.134 (Prod finite-data sibling dependency)**.** *The* $\mathsf{Prod}^{\uparrow}$ *finite-data route consumes its sibling packets only through displayed finite rows:* $\mathsf{Bool}^{\uparrow}$ *from chapter 32 supplies tag discrimination for the two component projections,* $\mathsf{Option}^{\uparrow}$ *from chapter 33 supplies endpoint absence and presence discipline for partial component reads,* $\mathsf{Sum}^{\uparrow}$ *from chapter 35 supplies side-tag separation for coproduct-facing projections, and* $\mathsf{List}^{\uparrow}$ *from chapter 36 supplies finite-spine readback for iterated product windows. No sibling is read as an ambient equality, selector, quotient, or external recursion principle.*

Proof. Project the product hub through its displayed finite-data rows. Boolean tags decide which component field is inspected, option endpoints record whether a component read is absent or present, sum tags keep left and right alternatives disjoint, and list spines enumerate the finite sequence of product windows consumed by readback.

These rows are all finite carrier coordinates already named by the product dossier. Transport and replay preserve the displayed rows componentwise, while provenance and the local NameCert only record the same finite route. Since no row stores quotient equality, selected data, or an external recursion principle, the sibling dependency remains confined to the displayed finite-data surface. □

Theorem 34.135 (Prod finite-data real-completion sibling route)**.** *Every finite product pairing used by Real and Completion consumers factors through the* $\mathsf{Prod}^{\uparrow}$ *public finite-data surface before any* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *product-window read is exported. The route first reads the product history carrier, componentwise classifier, displayed-pair ledger, source* NameCert *rows,* hsame *transport,* Cont *continuation, and* Pkg *provenance, and only then presents paired windows to chapter 126 and regular paired coordinate reads to chapter 301.*

In particular, the completion-facing product window is not a host tuple, quotient pair, selected projection, or hidden product stream. It is the displayed finite-data route of Definition 34.110, Definition 34.134, and Definition 34.118 read before the StreamName and RegSeqRat sibling windows are consumed.

Proof. Start with the product public finite-data exit package. By Definition 34.118, a public product read exposes only the componentwise semantic certificate, displayed-pair ledger endpoints, descent to the product history envelope, source certificates, transport, continuation, and provenance rows.

The finite-data sibling route supplies the window discipline. Apply Definition 34.110 and Definition 34.134; they keep product reads behind displayed finite sibling rows before the ordered-pair certificate is exported.

The Real and Completion consumers then read paired windows through chapter 126 and paired regular coordinate rows through chapter 301. Since the product surface has no host tuple equality, quotient pair, selected projection, or hidden stream coordinate, the StreamName and RegSeqRat product-window reads remain downstream of $\mathsf{Prod}^{\uparrow}$. □

Theorem 34.136 (Prod RegSeqRat and StreamName product window)**.** *Whenever a completion-facing consumer reads a pair of stream coordinates or regular-Cauchy coordinates through* $\mathsf{Prod}^{\uparrow}$*, the read factors through the componentwise* SemanticNameCert *row, displayed-pair ledger endpoints,* $\mathsf{ProdHistoryClassifier}^{\uparrow}$ *descent,* hsame *and* Cont *transports, and source* NameCert *rows before* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumes the pair. The product window exports finite paired data only; it does not export host tuple equality, hidden projections, quotient pair representatives, or an ambient product stream.*

Proof. Apply Definition 34.135. It already places Real-and-Completion product reads behind the $\mathsf{Prod}^{\uparrow}$ public finite-data surface before StreamName and RegSeqRat consumers inspect paired windows.

Specialize the downstream readers to the two completion-facing cases. A $\mathsf{StreamName}^{\uparrow}$ product-window read consumes paired stream coordinates through the displayed product endpoint rows, while a $\mathsf{RegSeqRat}^{\uparrow}$ read consumes regular coordinate pairs through the same finite product route.

Now use consumer-row exhaustion from the public finite-data exit package. The componentwise SemanticNameCert row, displayed-pair ledger endpoints, $\mathsf{ProdHistoryClassifier}^{\uparrow}$ descent, source certificates, hsame transport, Cont continuation, and provenance rows are the whole product surface. Thus no host tuple equality, hidden projection, quotient representative, or ambient product stream remains for either downstream reader. □

Theorem 34.137 (Prod real-tail window forward link)**.** *Any* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *packet using two* $\mathsf{StreamName}^{\uparrow}$ *windows reads its pair of finite windows through the* $\mathsf{Prod}^{\uparrow}$ *componentwise interface before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger are consumed. The two windows are therefore a displayed product route: first the product history carrier, componentwise classifier, displayed-pair ledger, source* NameCert *rows,* hsame *transport,* Cont *replay, and* Pkg *provenance of Definition 34.118; then the finite-window rows of chapter 3427; and only after that the regular rational readbacks and shared dyadic tolerance ledger.*

Proof. Start with the product componentwise surface. The componentwise interface and public finite-data exit package expose the left and right component histories, the displayed pair ledger, source certificates, transport, replay, and provenance before a consumer receives a paired read.

Use the real-completion product-window route, Definition 34.135. It already places StreamName and RegSeqRat product-window reads downstream of the displayed $\mathsf{Prod}^{\uparrow}$ finite-data route rather than behind a host tuple or selected projection.

Now read the $\mathsf{RealUniformTailComparison}^{\uparrow}$ carrier. Its two windows W_A, W_B are consumed before the regular rational rows Q_A, Q_B and before the shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger. Therefore a comparison packet that uses two finite windows must first pass through the product component rows that make those windows a displayed pair.

The downstream handoff is exactly the finite route through $\mathsf{Prod}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$. It exports no host tuple equality, quotient stream equality, selected projection, hidden product stream, or ambient real-completeness row. □

Theorem 34.138 (Prod finite-data sibling route)**.** *Every public finite-product carrier route consumed by* $\mathsf{Prod}^{\uparrow}$ *reads the sibling finite-data surfaces before exporting the product classifier. The route uses chapter 32 for* $\mathsf{Bool}^{\uparrow}$ *endpoint discrimination, chapter 33 for* $\mathsf{Option}^{\uparrow}$ *partiality, chapter 35 for* $\mathsf{Sum}^{\uparrow}$ *tagged alternatives, and chapter 36 for* $\mathsf{List}^{\uparrow}$ *spine discipline. No product classifier is exported from a route that omits one of those displayed sibling rows.*

Proof. Start with the product public finite-data exit package. It exports the product classifier only after the displayed product carrier, component rows, history envelope, transport, continuation, provenance, and local NameCert rows have been read.

The finite sibling route supplies the required four surfaces. $\mathsf{Bool}^{\uparrow}$ distinguishes the two endpoint projections, $\mathsf{Option}^{\uparrow}$ records absent and present partial component reads, $\mathsf{Sum}^{\uparrow}$ separates tagged alternatives for case-facing consumers, and $\mathsf{List}^{\uparrow}$ records the finite spine discipline for iterated product windows.

Since the product packet has no host tuple equality, selected projection, quotient carrier, or external recursion coordinate, the classifier cannot bypass those sibling rows. The public route therefore consumes the four finite-data surfaces before the product classifier is exported. □

Theorem 34.139 (Prod Bool Option Sum List forward link)**.** *The* $\mathsf{Prod}^{\uparrow}$ *componentwise carrier is the binary product node in the finite-data lattice whose projections are consumed alongside* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*, and* $\mathsf{List}^{\uparrow}$ *through chapter 32, chapter 33, chapter 35, and chapter 36. Every product classifier read therefore remains a displayed finite-data route rather than an ambient tuple primitive.*

Proof. Read the existing product component rows from the $\mathsf{Prod}^{\uparrow}$ carrier. They expose left and right projections through the componentwise history envelope, classifier fields, transport, continuation, provenance, and local naming rows.

The sibling finite-data routes supply the four adjacent surfaces. $\mathsf{Bool}^{\uparrow}$ distinguishes endpoint positions, $\mathsf{Option}^{\uparrow}$ records optional component presence, $\mathsf{Sum}^{\uparrow}$ separates tagged alternatives, and $\mathsf{List}^{\uparrow}$ records the finite-spine discipline used by iterated product windows.

Since the product carrier exports no host tuple equality, selected projection, quotient product, or unrecorded recursion primitive, the forward link is exactly the displayed route from the product node to those sibling finite-data certificates. □

Theorem 34.140 (Prod finite-data sibling orphan route)**.** *Every* $\mathsf{Prod}^{\uparrow}$ *finite-data read is routed through the sibling lattice consisting of BoolUp endpoint tags from chapter 32, OptionUp optional envelopes from chapter 33, SumUp injections from chapter 35, and ListUp spine and readback rows from chapter 36. The exported product read is therefore a finite* BHist *route with componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* $\mathsf{NameCert}$ *transport. It exposes no host product equality and no tuple primitive outside the displayed finite-data siblings.*

Proof. Start with the endpoint tags. The BoolUp sibling supplies the finite left/right and endpoint-discriminator rows used to distinguish the component positions of a product read.

Next wrap absence and presence through OptionUp and SumUp. Optional envelopes record whether a component slot is present, while sum injections route the displayed component into the finite coproduct surface before any list spine is consumed.

Then read the ListUp spine. The list rows provide the finite order, readback, and replay discipline for the component family, so the product packet is inspected as named finite data rather than as a host tuple.

Finally transport the route through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. These rows preserve the displayed sibling data without adding host product equality or an external tuple primitive. □

The $\mathsf{Prod}^{\uparrow}$ finite-data dependency lattice closes over sibling routes only after the ordered-pair envelope has fixed its two component projections. Bool tag discipline from chapter 32, Option absence and presence routing from chapter 33, Sum side tags from chapter 35, and List finite-spine readback from chapter 36 are consumed as displayed sibling rows around the product component ledger, not as detached host projections.

Lemma 34.141 (Prod finite-data dependency lattice closure)**.** *Every public* $\mathsf{Prod}^{\uparrow}$ *component projection read that cites the finite-data sibling lattice first consumes the product component ledger and then routes Bool endpoint tags, Option absent/present rows, Sum side tags, and List finite-spine readback through the corresponding sibling chapters. The route preserves the same* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ *row, and it exports no ambient tuple equality, unrestricted recursion principle, selected component, or detached finite-data carrier.*

Proof. Begin with the product component ledger fixed by the hub children. The ledger names the left and right component projections before any sibling finite-data read is allowed to consume a tag, absence marker, side marker, or spine row.

Now read the siblings through their own chapters. Bool supplies only endpoint discrimination, Option supplies only the absent/present boundary, Sum supplies only left/right side separation, and List supplies only finite-spine readback.

The structural rows of $\mathsf{Prod}^{\uparrow}$ carry those displayed reads without changing ownership. Thus the dependency lattice closes over finite sibling data while keeping the public ordered-pair certificate inside the product NameCert surface. □

The $\mathsf{Prod}^{\uparrow}$ pair carrier also records a finite-data sibling route through the endpoint, alternative, and spine interfaces already named by the concrete-instance lattice. Bool endpoints are read through chapter 32, tagged alternatives through chapter 35, and finite routing spines through chapter 36; the product packet consumes those rows as dependencies rather than as hidden host projections.

Theorem 34.142 (Prod sibling carrier route)**.** *Every accepted* $\mathsf{Prod}^{\uparrow}$ *pair carrier may consume* $\mathsf{Bool}^{\uparrow}$ *endpoint discrimination,* $\mathsf{Sum}^{\uparrow}$ *tagged alternatives, and* $\mathsf{List}^{\uparrow}$ *finite spine routing only as displayed finite-data sibling dependencies. The carrier exposes those rows through its product ledger, componentwise transport, continuation replay, provenance, and local* NameCert *row, and it does not replace them by host product extensionality or an unrecorded pairing oracle.*

Proof. Read the pair carrier through the existing product ledger. Endpoint discrimination is supplied by the $\mathsf{Bool}^{\uparrow}$ sibling, so a branch decision entering the pair route is a displayed finite row.

Tagged alternatives are then supplied by the $\mathsf{Sum}^{\uparrow}$ sibling. The pair carrier records which side of the product dependency is being consumed without replacing that tag by host-level case analysis.

Finally the finite spine route is supplied by $\mathsf{List}^{\uparrow}$. Componentwise transport, continuation replay, provenance, and the local NameCert row preserve the same finite sibling surface, so the product carrier exports no hidden pairing oracle. □

Theorem 34.143 (Prod sibling carrier route obligation scope)**.** *Any accepted* $\mathsf{Prod}^{\uparrow}$ *sibling route consumes* $\mathsf{Bool}^{\uparrow}$ *endpoint discrimination,* $\mathsf{Sum}^{\uparrow}$ *tagged alternatives, and* $\mathsf{List}^{\uparrow}$ *finite spine routing only through the product carrier ledger, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The dependency route is therefore pinned to chapter 32, chapter 35, and chapter 36; it cannot split those sibling rows into host projections, unrecorded case analysis, or an external pairing oracle.*

Proof. Apply Definition 34.142. It already reads the endpoint, alternative, and spine dependencies as displayed finite-data rows inside the accepted product carrier.

The obligation scope is obtained by naming the same structural rows. The product ledger stores the dependency consumption, componentwise hsame transport carries it across equivalent pair carriers, Cont replay preserves the route, Pkg records the provenance, and the local NameCert row names the whole package.

Thus the Bool, Sum, and List dependencies remain sibling rows of the product certificate. Since no host projection or external pairing coordinate is present, every accepted route factors through the displayed carrier ledger. □

The public $\mathsf{Prod}^{\uparrow}$ ordered-pair carrier is consumed through the finite-data sibling route: Bool endpoint discrimination from chapter 32, Option possible-absence routing from chapter 33, Sum tagged alternatives from chapter 35, and List spine readback from chapter 36.

This ledger records the dependency route as a hub-visible child input while leaving the theorem-free $\mathsf{Prod}^{\uparrow}$ index as an input list. Public $\mathsf{Prod}^{\uparrow}$ reads use the two displayed component projections

before they enter the finite-data sibling lattice generated by chapter 35 and chapter 36. The product packet therefore contributes paired component ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any downstream consumer treats the data as a tagged branch or finite spine.

Definition 34.144 (Prod stream-window carrier)**.** *A* $\mathsf{Prod}^{\uparrow}$ *stream-window carrier for L10 product reads is a finite* BHist *packet*

$$P = (L, R, W, D, Q, H, C, N)$$

with the following displayed rows:

1. *L is the left component row and R is the right component row.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *product window, carrying the finite request schedule through which both components are read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used when the product read is consumed by regular rational windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback row, assembled componentwise from the two component reads over the shared window.*
5. *H records componentwise* hsame *transport for L, R, W, D, Q and for the local classifier row.*
6. *C records displayed* Cont *replay routes for the product window, dyadic tolerance ledger, and regular rational readback.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ *row naming this finite product-window surface.*

The classifier compares accepted packets only through these component rows. It has no coordinate for abstract carrier-equivalence transport or for a product object detached from the displayed StreamName and RegSeqRat windows.

Theorem 34.145 (Prod stream-window NameCert obligations)**.** *For every accepted* $\mathsf{Prod}^{\uparrow}$ *stream-window carrier $P = (L, R, W, D, Q, H, C, N)$, the local* $\mathsf{NameCert}_{\mathsf{Prod}^{\uparrow}}$ *obligation surface consists exactly of left and right component admission, the shared* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback of chapter 301, componentwise* hsame *transport, displayed* Cont *replay, and local naming.*

Proof. Project the carrier of Definition 34.144. The rows L and R give the two component reads, while W is the only public window through which those components are synchronized.

The dyadic ledger D and regular readback row Q are computed from the same component windows. Componentwise transport through H preserves the two component rows, and replay through C exposes the same product-window route to downstream readers.

The local NameCert row N names this finite package. Since the carrier stores no detached product object or abstract carrier-equivalence transport, the obligations are exactly the displayed componentwise product-window rows. □

Theorem 34.146 (Prod RegSeqRat window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *product-window read consumed through* $\mathsf{Prod}^{\uparrow}$ *factors componentwise through the shared* $\mathsf{StreamName}^{\uparrow}$ *window W, the left and right component rows L, R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the product readback row Q. The handoff supplies the finite product surface needed by StreamName and regular-rational consumers; it does not transport an abstract product equivalence in place of the displayed component rows.*

Proof. Start with a downstream $\mathsf{RegSeqRat}^{\uparrow}$ read of the product packet. The only regular rational coordinate is Q, and it is assembled from L and R over the shared request window W.

Use D to audit the component tolerances on the same window. The replay row C exposes the componentwise reads and tolerance checks before the product readback is public.

Transport through H preserves the left row, right row, shared window, dyadic ledger, and product readback separately. Therefore the handoff remains a finite StreamName-to-RegSeqRat product-window route rather than an abstract carrier-equivalence transport. □

The $\mathsf{Sum}^{\uparrow}$ branch partition and visible-branch classifier rows are consumed beside chapter 34 and chapter 36 as finite coproduct/list-product routing data, while this hub exports no host disjoint union semantics.

Chapter 35

A Concrete Naming Certificate for $\mathsf{Sum}^{\uparrow}$

The disjoint sum interface $\mathsf{Sum}^{\uparrow}$ is licensed over two source carriers each endowed with a NameCert. The carrier is the inductive sum with left and right constructors wrapping source-carrier evidence. The classifier is the constructor-discriminating payload relation: it stores visible tag endpoints together with the matching source classifier witness, while actual source membership remains in SumHistoryCarrier and is the evidence used by the semantic certificate. It is not a primitive coproduct; it is licensed only as a derived constructor over the two sources.

35.1 Roadmap template status

1. **Traditional target**: disjoint sum data with left and right tagged injections.
2. **BEDC source**: two licensed source carriers A and B, each with its own name certificate.
3. **Carrier choice**: tagged left carriers use $e0$ histories and tagged right carriers use $e1$ histories.
4. **Classifier choice**: the sum classifier stores tag endpoints plus the matching A or B classifier witness, and excludes mixed tags; source-carrier evidence is supplied by SumHistoryCarrier.
5. **Stability obligations**: tagged carrier transport, source-certified classifier-induced carrier transport, left/right inversion, mixed-tag exclusion, classifier symmetry, classifier transitivity, and endpoint transport.
6. **Ledger**: the tag is visible data; payload evidence is retained from the corresponding source certificate.
7. **Core theorem list**: tagged injections, hsame carrier transport, source-certified classifier-induced carrier transport, left and right inversions, classifier inversion, source-parameter irrelevance of the classifier, generic same-tag exactness, carrier-aware same-tag exactness, mixed-tag exclusion, and the semantic name certificate.
8. **Exactness predicate**: the left-tag and right-tag iff clauses identify classifier-only payload witnesses and the matching source classifier, while carrier-aware exactness combines SumHistoryCarrier with explicit carrier transport along hsame.
9. **Standard bridge**: tagged histories read back as ordinary $\mathsf{inl}(a)$ and $\mathsf{inr}(b)$ values.

10. **Status**: the SemanticNameCert at Theorem 35.108 is CheckedCert; Theorem 35.16 is the paper-side matching-source formulation, and Theorem 35.18 records the witness-level same-tag formulation.

35.2 Carrier definition

Definition 35.1 (Sum carrier)**.** *For licensed source carriers A and B, define an inductive carrier with constructors* $\mathsf{inl}(a)$ *and* $\mathsf{inr}(b)$*.*

Lean4 (definition) `BEDC.Derived.SumUp.SumCarrier`
Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryCarrier`

Theorem 35.2 (Sum history carrier transport)**.** *If h and k are internally the same BEDC histories, then membership of h in the tagged sum-history carrier transfers to membership of k in the same carrier.*

Proof. Invert the sum carrier witness. In the left-tag case, compose symmetry of the given $\mathsf{hsame}(h, k)$ witness with the stored $\mathsf{hsame}(h, e0(l))$ tag witness, and keep the same left source-carrier proof. The right-tag case is identical with $e1(r)$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier hsame transport`

Theorem 35.3 (Sum history carrier branch exclusivity)**.** *For source carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$, no BEDC history h has both a left carrier presentation and a right carrier presentation:*

$$\neg\Big((\exists l : \mathsf{BHist}.\ \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge Left(l)) \wedge (\exists r : \mathsf{BHist}.\ \mathsf{hsame}(h, \mathsf{E1}(r)) \wedge Right(r))\Big).$$

Proof. Assume both presentations. The left witness gives $\mathsf{hsame}(h, \mathsf{E0}(l))$, and the right witness gives $\mathsf{hsame}(h, \mathsf{E1}(r))$. Compose the symmetry of the left endpoint witness with the right endpoint witness to obtain $\mathsf{hsame}(\mathsf{E0}(l), \mathsf{E1}(r))$, then close by BEDC $e0/e1$ no-confusion. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier branch exclusivity`

Theorem 35.4 (Sum history carrier excludes empty)**.** *For any source carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$, the empty history is not a tagged sum-history carrier:*

$$\neg\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{e}).$$

Proof. Eliminate the tagged carrier witness. A left branch would identify e with an $\mathsf{E0}$-tagged payload, and a right branch would identify e with an $\mathsf{E1}$-tagged payload. Both contradict history no-confusion. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier empty absurd`

Theorem 35.5 (Sum history left-tag inversion)**.** *If an $e0$-tagged history belongs to the tagged sum-history carrier, then its payload is history-same to a left carrier witness.*

Proof. Invert the tagged carrier witness. In the left branch, BEDC $e0$-constructor inversion recovers the payload sameness and keeps the stored left carrier proof. In the right branch, the $e0/e1$ no-confusion theorem contradicts the stored tag witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier e0 inversion`

Theorem 35.6 (Sum history right-tag inversion)**.** *If an $e1$-tagged history belongs to the tagged sum-history carrier, then its payload is history-same to a right carrier witness.*

Proof. Invert the tagged carrier witness. In the left branch, the $e1/e0$ no-confusion theorem contradicts the stored tag witness. In the right branch, BEDC $e1$-constructor inversion recovers the payload sameness and keeps the stored right carrier proof. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier e1 inversion`

Theorem 35.7 (Sum history carrier visible-tag exactness)**.** *For source carriers $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$, the visible tags of the tagged sum-history carrier are exactly the corresponding source carrier payloads:*

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l)) \iff \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l, a),$$

and

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E1}(r)) \iff \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r, b).$$

Proof. For the forward implications, apply the checked left-tag and right-tag inversion theorems to the carrier witness. For the reverse implications, start from the payload witness, use the checked tagged-injection constructor for the matching branch, convert the payload endpoint by constructor congruence for hsame, and transport carrier membership along the resulting history sameness by SumHistoryCarrier_hsame_transport. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible tag exactness`

35.3 The certificate

Definition 35.8 (Sum source specification)**.** *The source is the disjoint union of the two source certificates.*

Definition 35.9 (Sum pattern specification)**.** *The pattern is left-or-right injection from the corresponding source witness.*

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier tagged injections`

Definition 35.10 (Sum classifier specification)**.** *The classifier records left-tag endpoint payloads through the A-classifier and right-tag endpoint payloads through the B-classifier, and never relates opposite constructors. Source membership is carried by* SumHistoryCarrier *rather than by this classifier relation.*

Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryClassifier`
Lean4 (definition) `BEDC.Derived.SumUp.SumClassifierSpec`
Lean4 ✓ `BEDC.Derived.SumUp.sum classifier inversion`

Theorem 35.11 (Sum classifier does not carry source membership)**.** *Let* $\mathsf{FalseCarrier}(h)$ *mean* False *for every BEDC history h. Then*

$$\mathsf{SumHistoryClassifier}(\mathsf{FalseCarrier}, \mathsf{FalseCarrier}, \mathsf{hsame}, \mathsf{hsame}, \mathsf{E0}(\mathsf{e}), \mathsf{E0}(\mathsf{e}))$$

holds, while

$$\neg\mathsf{SumHistoryCarrier}(\mathsf{FalseCarrier}, \mathsf{FalseCarrier}, \mathsf{E0}(\mathsf{e})).$$

Proof. Build the classifier by the left-tag branch, using e as both stored left payloads, reflexivity of hsame for the two visible E0(e) endpoints, and reflexivity of hsame for the payload relation. A carrier witness for E0(e) would be either a left branch with some payload satisfying FalseCarrier or a right branch with some payload satisfying FalseCarrier. Both cases unfold to False, so no carrier witness exists. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier no source membership`

Theorem 35.12 (Sum history classifier source-parameter irrelevance)**.** *For source carriers* $Left, Right, LeftAlt, R$ BHist → Prop*, component classifiers* $LeftEq, RightEq$: BHist → BHist → Prop*, and histories* h, k : BHist*,*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \iff \mathsf{SumHistoryClassifier}(LeftAlt, RightAlt, LeftEq, Ri$$

Proof. For the forward direction, case-analyze the classifier witness. In the left-tag branch, keep the same stored left payloads, endpoint tag witnesses, and $LeftEq$ witness, and build the left branch for the alternate source carriers. In the right-tag branch, keep the same stored right payloads, endpoint tag witnesses, and $RightEq$ witness, and build the right branch for the alternate source carriers. For the reverse direction, exchange the two source-carrier pairs and repeat the same branch analysis. No source-carrier proof occurs inside the classifier witness; those proofs belong to SumHistoryCarrier and to certificate transport. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier source parameter irrelevance`

Mixed-tag classifier exclusion is recorded by Theorem 35.41.

Theorem 35.13 (Sum history classifier left-tag inversion)**.** *When the history-level sum classifier uses* hsame *on both components and relates two* $e0$*-tagged histories, the two left payload histories are internally the same.*

Proof. Invert the tagged classifier witness. The left branch gives two $e0$-tag witnesses and a stored component hsame witness; constructor inversion for $e0$ composes them into payload sameness. The right branch contradicts constructor separation between $e0$ and $e1$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier left hsame inversion`

Theorem 35.14 (Sum history classifier right-tag inversion)**.** *When the history-level sum classifier uses* hsame *on both components and relates two* $e1$*-tagged histories, the two right payload histories are internally the same.*

Proof. Invert the tagged classifier witness. The left branch contradicts constructor separation between $e1$ and $e0$. The right branch gives two $e1$-tag witnesses and a stored component hsame witness; constructor inversion for $e1$ composes them into payload sameness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier right hsame inversion`

The same-tag iff clauses below are classifier-only statements; carrier and ledger evidence are supplied separately by SumHistoryCarrier and the ledger policy.

Theorem 35.15 (Sum history classifier same-tag exactness)**.** *For source carriers* $Left, Right$: BHist → Prop*, the history-level sum classifier with* hsame *on both components is exact on each visible tag:*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \mathsf{E0}(l), \mathsf{E0}(l')) \iff \mathsf{hsame}(l, l')$$

and

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \mathsf{E1}(r), \mathsf{E1}(r')) \iff \mathsf{hsame}(r, r').$$

Proof. For the forward directions, apply the left-tag and right-tag inversion theorems above. For the reverse directions, rebuild the matching classifier branch using the given payload hsame witness, the reflexive visible-tag witnesses for $\mathsf{E0}(l)$ and $\mathsf{E0}(l')$ or for $\mathsf{E1}(r)$ and $\mathsf{E1}(r')$, and the corresponding left or right classifier constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier hsame same tag exactness`

Theorem 35.16 (Sum history classifier same-tag source exactness)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, suppose the source classifiers transport along endpoint history sameness:*

$$\forall l, l_0, l', l'_0.\ \mathsf{hsame}(l, l_0) \to \mathsf{hsame}(l', l'_0) \to LeftEq(l_0, l'_0) \to LeftEq(l, l')$$

and

$$\forall r, r_0, r', r'_0.\ \mathsf{hsame}(r, r_0) \to \mathsf{hsame}(r', r'_0) \to RightEq(r_0, r'_0) \to RightEq(r, r').$$

Then the history-level sum classifier is exact for same-tag endpoints through the matching source classifier. This theorem is a classifier-only same-tag exactness statement; it returns the matching source classifier witness and does not assert $Left(l)$*,* $Left(l')$*,* $Right(r)$*, or* $Right(r')$*:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, \mathsf{E0}(l), \mathsf{E0}(l')) \iff LeftEq(l, l')$$

and

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, \mathsf{E1}(r), \mathsf{E1}(r')) \iff RightEq(r, r').$$

Proof. Invert the same-tag classifier witness. In the left-tag forward direction, the right branch is impossible by $e0/e1$ no-confusion. The left branch stores endpoint payloads l_0, l'_0 together with $LeftEq(l_0, l'_0)$; constructor inversion for $e0$ yields $\mathsf{hsame}(l, l_0)$ and $\mathsf{hsame}(l', l'_0)$, and the left transport hypothesis gives $LeftEq(l, l')$. The right-tag forward direction is identical using $e1$-constructor inversion and the right transport hypothesis. For the reverse directions, use the visible endpoints as the stored payloads, the reflexive tag witnesses for E0 or E1, and the supplied matching source-classifier witness to rebuild the corresponding classifier branch. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier source same tag exactness`

Theorem 35.17 (Sum history classifier hsame same-tag carrier exactness)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$*, the* hsame*-specialized history-level sum classifier is exact for same-tag endpoints together with the matching source-carrier evidence:*

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l)) \wedge \mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l')) \wedge \mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hs}$$

and

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E1}(r)) \wedge \mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E1}(r')) \wedge \mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hs}$$

Its roadmap-template role is: traditional target disjoint-sum same-tag payload exactness; BEDC source BHist *with* hsame*; carrier* SumHistoryCarrier*; classifier the* hsame *specialization of* SumHistoryClassifier*; ledger visible tag plus payload carrier witnesses; status* ProofTarget*.*

Proof. For the forward left-tag direction, apply the two $\mathsf{E0}$ carrier inversions to obtain payload witnesses a, a' with $Left(a)$, $Left(a')$, $\mathsf{hsame}(l, a)$, and $\mathsf{hsame}(l', a')$. Apply the left hsame classifier inversion to the same-tag classifier witness to obtain $\mathsf{hsame}(l, l')$. Symmetry and transitivity of hsame compose the displayed endpoint links with $\mathsf{hsame}(l, l')$ to derive $\mathsf{hsame}(a, a')$. The right-tag direction is identical using the two $\mathsf{E1}$ carrier inversions and the right hsame classifier inversion. For the reverse left-tag direction, the displayed witnesses rebuild the two left carrier branches and the left classifier branch. The reverse right-tag direction rebuilds the two right carrier branches and the right classifier branch. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier hsame same tag carrier exactness`

Theorem 35.18 (Sum history classifier generic same-tag exactness)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, the history-level sum classifier is exact on each visible tag:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, \mathsf{E0}(l), \mathsf{E0}(l')) \iff \exists a, a'.\ \mathsf{hsame}(l, a) \wedge \mathsf{hsame}(l', a') \wedge LeftE$$

and

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, \mathsf{E1}(r), \mathsf{E1}(r')) \iff \exists b, b'.\ \mathsf{hsame}(r, b) \wedge \mathsf{hsame}(r', b') \wedge Right$$

Proof. Unfold $\mathsf{SumHistoryClassifier}$. In the left-tag forward direction, the left branch already stores payload witnesses for the two endpoints together with a $LeftEq$ witness; the opposite right branch is impossible by $e0/e1$ no-confusion, and $e0$-constructor inversion identifies the visible endpoints with the stored payload tags. The right-tag forward direction is identical with $e1$-constructor inversion and $RightEq$. For the reverse directions, the displayed payload witnesses rebuild the matching left or right classifier branch by constructor congruence for $e0$ or $e1$, together with the stored source-classifier witness. The surviving branch is constructor-stable, and the impossible branch is discharged by the same tag no-confusion principle. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier generic same tag exactness`

Theorem 35.19 (Sum history classifier left-chain visible payload classification)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *and histories* h, k, r, a, c*, if*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, h, k), \qquad \mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, k, r),$$

and if the outer endpoints are visibly left-tagged as

$$\mathsf{hsame}(h, \mathsf{E0}(a)), \qquad \mathsf{hsame}(r, \mathsf{E0}(c)),$$

then $\mathsf{hsame}(a, c)$*.*

Proof. Compose the two sum classifiers using transitivity of hsame on both branches. Transport the resulting classifier along the two displayed outer endpoint presentations, obtaining a classifier between $\mathsf{E0}(a)$ and $\mathsf{E0}(c)$. Left-tag classifier inversion then identifies the two visible payloads by $\mathsf{hsame}(a, c)$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier left chain visible payload classification`

Theorem 35.20 (Sum history classifier left endpoint fanout)**.** *For source carriers* $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$*, source classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k, r, a*, if*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k), \qquad \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq,$$

then both targets expose left-tag payload witnesses:

$$\begin{aligned}&(\exists a', b.\ \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a', b))\\ &\wedge(\exists a'', c.\ \mathsf{hsame}(a, a'') \wedge \mathsf{hsame}(r, \mathsf{E0}(c)) \wedge LeftEq(a'', c)).\end{aligned}$$

Proof. Case-analyze both sum-classifier witnesses. A right-branch witness would force the shared source endpoint to be simultaneously an E0 endpoint and an E1 endpoint, contradicting constructor separation. The surviving left branches expose the stored source payload, each target payload, the target endpoint equation, and the corresponding $LeftEq$ witness; E0 inversion composes the displayed source endpoint with the stored source tag. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier left endpoint fanout`

Theorem 35.21 (Sum history classifier right endpoint fanout)**.** *For source carriers* $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$*, source classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k, r, a*, if*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k), \qquad \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq,$$

then both targets expose right-tag payload witnesses:

$$\begin{aligned}&(\exists a', b.\ \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a', b))\\ &\wedge(\exists a'', c.\ \mathsf{hsame}(a, a'') \wedge \mathsf{hsame}(r, \mathsf{E1}(c)) \wedge RightEq(a'', c)).\end{aligned}$$

Proof. Case-analyze both sum-classifier witnesses. A left-branch witness would force the shared source endpoint to be simultaneously an E1 endpoint and an E0 endpoint, contradicting constructor separation. The surviving right branches expose the stored source payload, each target payload, the target endpoint equation, and the corresponding $RightEq$ witness; E1 inversion composes the displayed source endpoint with the stored source tag. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier right endpoint fanout`

Theorem 35.22 (Sum history classifier carrier-aware same-tag exactness)**.** *Let* $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume the two carriers transport along* hsame*:*

$$\forall x\, y.\ Left(x) \to \mathsf{hsame}(x, y) \to Left(y)$$

and

$$\forall x\, y.\ Right(x) \to \mathsf{hsame}(x, y) \to Right(y).$$

Then the visible same-tag carrier and classifier evidence is exact for the matching source payloads:

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l)) \wedge \mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l')) \wedge \mathsf{SumHistoryClassifier}(Left, Right$$

and

$$\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E1}(r)) \wedge \mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E1}(r')) \wedge \mathsf{SumHistoryClassifier}(Left, Righ$$

Its roadmap-template role is: traditional target disjoint-sum same-tag payload exactness; BEDC source BHist *with* SumHistoryCarrier *and carrier transport along* hsame*; carrier the tagged sum carrier; classifier* SumHistoryClassifier*; ledger visible tag plus payload evidence; exactness the displayed iff clauses; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier aware same tag exactness`

Proof. For the forward left-tag direction, invert the two displayed carrier witnesses to obtain payloads c, c' with $Left(c)$ and $Left(c')$ plus $\mathsf{hsame}(l, c)$ and $\mathsf{hsame}(l', c')$. Apply Theorem 35.18 to the classifier premise to obtain payloads a, a' with $\mathsf{hsame}(l, a)$, $\mathsf{hsame}(l', a')$, and $LeftEq(a, a')$. Symmetry and transitivity of hsame align c with a and c' with a'. Apply the explicit left carrier-transport field to $Left(c)$ and $\mathsf{hsame}(c, a)$, and to $Left(c')$ and $\mathsf{hsame}(c', a')$. This gives the displayed left payload witnesses. The forward right-tag direction is the same argument with E1, $Right$, $RightEq$, and the explicit right carrier-transport field applied in the same order. For the reverse directions, the displayed payload witnesses rebuild the two tagged carrier premises and the matching classifier branch by the generic same-tag exactness theorem, using the visible tag constructors to supply the endpoint evidence. □

Theorem 35.23 (Sum history classifier branch partition)**.** *For branch carriers* $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$*, branch classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k*, the history-level sum classifier exposes the visible left-or-right constructor analysis:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to (\exists l, l'.\ \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge \mathsf{hsame}(k, \mathsf{E0}(l')) \wedge LeftEq(l, l')$$

In the ten-field fit, the traditional target is disjoint-sum branch readback, the BEDC source is BHist*, the classifier is* SumHistoryClassifier*, the ledger records the visible tag plus payload, and the status is* ProofTarget*.*

Proof. Case-analyze the classifier witness. The left constructor supplies the two E0 endpoint links and the stored $LeftEq$ payload relation, yielding the left disjunct. The right constructor supplies the two E1 endpoint links and the stored $RightEq$ payload relation, yielding the right disjunct. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier branch partition`

Theorem 35.24 (Sum history classifier branch exactness)**.** *For branch carriers* $Left, Right$: $\mathsf{BHist} \to \mathsf{Prop}$*, branch classifiers* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k*, the history-level sum classifier is exactly the visible left-or-right branch disjunction:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \iff (\exists l, l'.\ \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge \mathsf{hsame}(k, \mathsf{E0}(l')) \wedge LeftEq(l, l$$

In the ten-field fit, the exactness target records that classifier witnesses and visible same-branch endpoint data determine each other.

Proof. For the forward direction, apply Theorem 35.74. It case-analyzes the classifier witness and returns the displayed left-or-right branch data.

For the reverse direction, split the displayed disjunction. In the left branch, unpack l, l' together with the two visible left endpoint witnesses and the stored $LeftEq(l, l')$ witness, then rebuild the left classifier constructor. In the right branch, unpack r, r' together with the two visible right endpoint witnesses and the stored $RightEq(r, r')$ witness, then rebuild the right classifier constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier branch iff`

Theorem 35.25 (Sum history carrier visible branch cases)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *holds, then the visible history has one of the two displayed branch witnesses:*

$$(\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \vee (\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r))).$$

In the ten-field fit, the traditional target is disjoint-sum carrier readback, the BEDC source is BHist*, the carrier is* SumHistoryCarrier*, no classifier is needed, the stability obligation is constructor inversion, the ledger records the visible tag plus payload, the core theorem is branch cases, exactness coverage is supplied by the visible branch witnesses, the standard bridge is* inl *and* inr*, and the status is* ProofTarget*.*

Proof. Case-analyze the generated SumHistoryCarrier witness. The left constructor returns a payload l, the source-carrier witness $Left(l)$, and the displayed endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$. The right constructor returns the corresponding payload r, the source-carrier witness $Right(r)$, and the displayed endpoint link $\mathsf{hsame}(h, \mathsf{E1}(r))$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible branch cases`

Theorem 35.26 (Sum history carrier visible branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. Then the sum history carrier is exactly the visible left-or-right branch assertion:*

$$\mathsf{SumHistoryCarrier}(Left, Right, h) \iff (\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \vee (\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r))).$$

In the ten-field fit, the carrier coverage predicate and the visible branch classifier agree on precisely the histories generated by the two sum constructors.

Proof. For the forward direction, apply Definition 35.76. For the reverse direction, split the displayed disjunction. In the left case, unpack the payload l, the source-carrier witness $Left(l)$, and the visible endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$, then rebuild the left carrier constructor. In the right case, unpack the corresponding data for r and rebuild the right carrier constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible branch exactness`

Theorem 35.27 (Sum history carrier exclusive visible branch partition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. Then the sum history carrier has exactly one visible branch presentation:*

$$\mathsf{SumHistoryCarrier}(Left, Right, h) \iff \Big((\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)))\Big) \vee \Big((\exists r.\ R$$

In the ten-field fit, the carrier readback for $\mathsf{Sum}^{\uparrow}$ *is a constructor partition: every carrier history is visibly generated by one sum constructor, and the opposite visible constructor presentation is excluded.*

Proof. For the forward direction, use Definition 35.77 to obtain either a visible left presentation or a visible right presentation. In the left case, the opposite right presentation would contradict Definition 35.3 after reordering the stored endpoint and source-carrier witnesses; the right case is symmetric. For the reverse direction, split the displayed disjunction, forget the negative conjunct, and rebuild $\mathsf{SumHistoryCarrier}(Left, Right, h)$ through Definition 35.77. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier exclusive visible branch partition payload first`

Theorem 35.28 (Sum visible branch payload determinism)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $h, l, l', r, r' : \mathsf{BHist}$ *be histories. Same-branch visible presentations have unique payloads up to* hsame*:*

$$\mathsf{hsame}(h, \mathsf{E0}(l)) \to Left(l) \to \mathsf{hsame}(h, \mathsf{E0}(l')) \to Left(l') \to \mathsf{hsame}(l, l')$$

and

$$\mathsf{hsame}(h, \mathsf{E1}(r)) \to Right(r) \to \mathsf{hsame}(h, \mathsf{E1}(r')) \to Right(r') \to \mathsf{hsame}(r, r').$$

In the ten-field fit, the traditional target is disjoint-sum visible readback determinism; the BEDC source is BHist*; the carrier is* SumHistoryCarrier *presentations; the classifier is* hsame *on visible constructors; the stability obligation is constructor inversion; the ledger records the visible tag plus payload source evidence; the core theorem is same-branch payload uniqueness; exactness is deterministic readback; the standard bridge is* inl *and* inr*; and the status is* ProofTarget*.*

Proof. For the left statement, use symmetry on the first endpoint link and compose it with the second endpoint link, obtaining $\mathsf{hsame}(\mathsf{E0}(l), \mathsf{E0}(l'))$. Constructor inversion for the visible left constructor gives $\mathsf{hsame}(l, l')$. The right statement is identical, composing the two E1 endpoint links and applying right-constructor inversion. The source-carrier witnesses record that both visible presentations are legitimate branch presentations; they are not needed by the constructor inversion itself. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible payload determinism`

Theorem 35.29 (Sum carrier single-valued visible branch readback)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $h : \mathsf{BHist}$*. If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *holds, then the visible tagged payload presentation of* h *is single-valued:*

$$\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge (\forall l'.\ Left(l') \to \mathsf{hsame}(h, \mathsf{E0}(l')) \to \mathsf{hsame}(l, l')) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r$$

or

$$\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)) \wedge (\forall r'.\ Right(r') \to \mathsf{hsame}(h, \mathsf{E1}(r')) \to \mathsf{hsame}(r, r')) \wedge \neg(\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}$$

In the ten-field fit, the traditional target is tagged disjoint-sum readback; the BEDC source is BHist*; the carrier is* SumHistoryCarrier*; the classifier is* hsame *on same-tag visible payloads; stability is constructor inversion plus branch exclusion; the ledger stores the visible tag and payload source evidence; the core theorem is this single-valued readback; exactness is existence, uniqueness, and opposite-tag exclusion; the standard bridge is* inl *and* inr*; and the status is* ProofTarget*.*

Proof. Apply Definition 35.78 to the carrier witness. In the left branch, unpack l, $Left(l)$, the endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$, and the exclusion of right visible presentations. For an arbitrary left visible payload l', apply Definition 35.79 to the two left endpoint links and the two source-carrier witnesses to obtain $\mathsf{hsame}(l, l')$. The right branch is symmetric, using the right payload clause and carrying the exclusion of left visible presentations unchanged. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier single valued visible readback`

Theorem 35.30 (Sum history classifier carrier-aware branch partition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier closure under history sameness:*

$$\forall x\, y.\ Left(x) \to \mathsf{hsame}(x, y) \to Left(y)$$

and

$$\forall x\, y.\ Right(x) \to \mathsf{hsame}(x, y) \to Right(y).$$

If $\mathsf{SumHistoryCarrier}(Left, Right, h)$, $\mathsf{SumHistoryCarrier}(Left, Right, k)$, *and* $\mathsf{SumHistoryClassifier}(Left, Right, Left$ *hold, then a visible classifier side supplies the branch data used by the carrier-backed certificate analysis:*

$$\exists a, b.\ Left(a) \wedge Left(b) \wedge \mathsf{hsame}(h, \mathsf{E0}(a)) \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a, b)$$

or

$$\exists a, b.\ Right(a) \wedge Right(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a, b).$$

Proof. Split the displayed SumHistoryClassifier witness into its visible left or right constructor branch. In the left branch, let the classifier payloads be a, b. Compose the carrier endpoint for h with the classifier endpoint $\mathsf{hsame}(h, \mathsf{E0}(a))$ and apply SumHistoryCarrier_e0_inversion to obtain a source payload l, $Left(l)$, and a payload link $\mathsf{hsame}(l, a)$; the same argument at k gives l', $Left(l')$, and $\mathsf{hsame}(l', b)$. The left source-carrier closure turns these into $Left(a)$ and $Left(b)$. Together with the stored $LeftEq(a, b)$ witness and the two classifier endpoint links, these data give the left disjunct. In the right branch, apply the analogous E1 endpoint composition and SumHistoryCarrier_e1_inversion argument to obtain $Right(a)$ and $Right(b)$ from the right source-carrier closure, then pair them with the stored $RightEq(a, b)$ witness and the two right endpoint links. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier aware branch partition`

Theorem 35.31 (Sum history classifier carrier-aware branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier closure under history sameness:*

$$\forall x\, y.\ Left(x) \to \mathsf{hsame}(x, y) \to Left(y)$$

and

$$\forall x\, y.\ Right(x) \to \mathsf{hsame}(x, y) \to Right(y).$$

Then the carrier-backed classifier assertion

$$\begin{aligned}&\mathsf{SumHistoryCarrier}(Left, Right, h) \wedge \mathsf{SumHistoryCarrier}(Left, Right, k) \wedge \\ &\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k)\end{aligned}$$

is equivalent to the visible branch assertion

$$\begin{aligned}&\big(\exists a, b.\ Left(a) \wedge Left(b) \wedge \mathsf{hsame}(h, \mathsf{E0}(a)) \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a, b)\big) \\ &\vee \\ &\big(\exists a, b.\ Right(a) \wedge Right(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a, b)\big).\end{aligned}$$

Proof. For the forward direction, apply Definition 35.81 with the two displayed source-carrier closure hypotheses. For the reverse direction, split the displayed branch assertion. In the left branch, unpack a, b with their $Left$ witnesses, the two visible E0 endpoint links, and the stored $LeftEq(a, b)$ witness. The same data rebuilds the two SumHistoryCarrier witnesses by the left carrier constructor and rebuilds the matching SumHistoryClassifier witness by the left classifier constructor. The right branch is identical, using the two $Right$ witnesses, the visible E1 endpoint links, $RightEq(a, b)$, and the right carrier and classifier constructors. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier aware branch exactness`

Theorem 35.32 (Sum history source weakening)**.** *Let* $Left, Right, Left', Right' : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. The two refinements have separate premises. Source-carrier inclusion gives carrier refinement:*

$$((\forall h.\ Left(h) \to Left'(h)) \wedge (\forall h.\ Right(h) \to Right'(h))) \to (\mathsf{SumHistoryCarrier}(Left, Right, h) \to \mathsf{SumHistoryCarr}$$

Component-relation weakening gives classifier refinement:

$$((\forall a\, b.\ LeftEq(a,b) \to LeftEq'(a,b)) \wedge (\forall a\, b.\ RightEq(a,b) \to RightEq'(a,b))) \to (\mathsf{SumHistoryClassifier}(Left, Rig$$

In the ten-field fit, the traditional target is disjoint-sum stability under source refinement, the BEDC source is the two BHist *carriers, the carrier is* SumHistoryCarrier*, the component classifier is* SumHistoryClassifier*, the stability datum is source inclusion plus component-relation weakening, the ledger keeps the visible tag unchanged, and the core theorem separates carrier weakening from classifier weakening. The classifier part uses only component-relation weakening because source membership is not a field of* SumHistoryClassifier*. Exactness is preserved by the same endpoint witnesses, the standard bridge is the* inl *and* inr *restriction, and the status is* ProofTarget*.*

Proof. For the carrier claim, assume the displayed pair of source-carrier inclusions and split the SumHistoryCarrier witness into its left and right constructor cases. The endpoint history witness is reused unchanged, while the matching source-carrier inclusion maps the payload membership proof into the refined source carrier. For the classifier claim, assume the displayed pair of component-relation weakenings and split the SumHistoryClassifier witness into its left and right constructor cases. Reuse the same two visible endpoints and apply only the matching component-relation weakening hypothesis to the stored $LeftEq$ or $RightEq$ witness. Source membership is absent from SumHistoryClassifier, so no source-carrier inclusion is needed for the classifier paragraph. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistorySource weakening`

Theorem 35.33 (Sum history source equivalence)**.** *Let* $Left, Right, Left', Right' : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier equivalence*

$$\forall h.\ Left(h) \leftrightarrow Left'(h)$$

and

$$\forall h.\ Right(h) \leftrightarrow Right'(h),$$

and assume component-relation equivalence

$$\forall a\, b.\ LeftEq(a,b) \leftrightarrow LeftEq'(a,b)$$

and

$$\forall a\, b.\ RightEq(a,b) \leftrightarrow RightEq'(a,b).$$

Then the sum carrier and classifier are invariant under the equivalent source presentation:

$$(\forall h.\ \mathsf{SumHistoryCarrier}(Left, Right, h) \leftrightarrow \mathsf{SumHistoryCarrier}(Left', Right', h)) \wedge (\forall h\, k.\ \mathsf{SumHistoryClassifier}(Left, R$$

In the ten-field fit, the traditional target is disjoint-sum source-presentation invariance, the BEDC source is the two BHist *carriers, the carrier is* SumHistoryCarrier*, the classifier is* SumHistoryClassifier*, the stability datum is source and relation equivalence, the ledger keeps the visible tag unchanged, and the core theorem is carrier and classifier equivalence. Exactness uses the unchanged branch witnesses, the standard bridge is* inl *and* inr *readback, and the status is* ProofTarget*.*

Proof. Apply Definition 35.83 from left to right using the forward halves of the two source-carrier equivalences for the carrier clause and the forward halves of the two component-relation equivalences for the classifier clause. This gives the implications from $(Left, Right, LeftEq, RightEq)$ to $(Left', Right', LeftEq', RightEq')$, with the same visible tag and the same branch witnesses. Apply the same weakening theorem from right to left using the reverse halves of the four equivalence hypotheses. Pair the two carrier implications and pair the two classifier implications. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistorySource equivalence`

Theorem 35.34 (Sum history classifier relation weakening)**.** *Let* $Left, Right, LeftAlt, RightAlt$: BHist → Prop *be arbitrary source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq'$: BHist → BHist → Prop *be component classifiers. Assume component-relation weakening*

$$\forall a\, b.\ LeftEq(a, b) \to LeftEq'(a, b)$$

and

$$\forall a\, b.\ RightEq(a, b) \to RightEq'(a, b).$$

Then the sum history classifier relation weakens independently of source-carrier inclusion:

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryClassifier}(LeftAlt, RightAlt, LeftEq', Rig$$

In the ten-field fit, the traditional target is disjoint-sum classifier monotonicity, the BEDC source is BHist*, the classifier is* SumHistoryClassifier*, the stability datum is component-relation weakening, the ledger keeps the visible tag unchanged, exactness is branch partition preservation, and the status is* ProofTarget*.*

Proof. Split the displayed SumHistoryClassifier witness into its visible left or right constructor branch. In each branch, reuse the same visible endpoint witnesses for h and k. The left branch applies the left component-relation weakening hypothesis to the stored $LeftEq$ witness and rebuilds the left tagged classifier witness; the right branch applies the right component-relation weakening hypothesis to the stored $RightEq$ witness and rebuilds the right tagged classifier witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier relation weakening`

Theorem 35.35 (Sum ledger raw-visible branch partition)**.** *Let* $Left, Right$: BHist → Prop *be source carriers and let* ρ, v : BHist *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then the same visible branch and payload appear on both ends of the ledger link:*

$$\big(\exists l.\ Left(l) \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l))\big) \vee \big(\exists r.\ Right(r) \wedge \mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r))\big).$$

In the ten-field fit, the traditional target is disjoint-sum ledger branch readback; the BEDC source is BHist *with* hsame*; the carrier is* SumHistoryCarrier*; no classifier is needed for the direct branch exposure; the ledger is* SumHistoryLedgerPolicy*; exactness is the same raw-visible branch and payload on both endpoints; and the status is* ProofTarget*.*

Proof. Unpack the ledger into the raw SumHistoryCarrier witness and the stored raw-visible $\mathsf{hsame}(\rho, v)$ link. Split the raw carrier. In the left branch, keep the raw payload l and its $Left(l)$ witness, retain the stored raw tag witness $\mathsf{hsame}(\rho, \mathsf{E0}(l))$, and compose symmetry of $\mathsf{hsame}(\rho, v)$ with that raw tag witness to obtain $\mathsf{hsame}(v, \mathsf{E0}(l))$. The right branch uses the same argument with the raw payload r, the $Right(r)$ witness, and the E1 tag. □

Theorem 35.36 (Sum history classifier symmetry under history sameness)**.** *When both component classifiers are* hsame*, the history-level sum classifier is symmetric.*

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier hsame symm`

Proof. Invert the classifier witness. In the left-tagged case, swap the two tagged witnesses and use symmetry of hsame on the left component. The right-tagged case is identical with the right component. □

Theorem 35.37 (Sum history classifier history transport)**.** *If both endpoints of a history-level* $\mathsf{Sum}^{\uparrow}$ *classifier are transported by* hsame*, then the same classifier witness transports to the transported endpoints.*

Proof. Invert the tagged classifier witness. In the left branch, keep the same left component witnesses and compose the reversed endpoint transports with the two stored $e0$-tag witnesses. The right branch is identical using the $e1$-tag witnesses. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier hsame transport`

Theorem 35.38 (Sum history classifier source-certified carrier transport)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume source certificates* $\mathsf{NameCert}(Left, LeftEq)$ *and* $\mathsf{NameCert}(Right, RightEq)$*. If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *and* $\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k)$ *hold, then* $\mathsf{SumHistoryCarrier}(Left, Right, k)$ *holds.*

Proof. Case-analyze the displayed carrier witness and the tagged classifier witness. Mixed source tags are impossible by BEDC $e0/e1$ no-confusion. In the left-left case, let c be the payload supplied by the carrier witness, and let a, a' be the source payloads supplied by the classifier witness. The carrier branch stores $\mathsf{hsame}(h, \mathsf{E0}(c))$ and $Left(c)$, while the classifier branch stores $\mathsf{hsame}(h, \mathsf{E0}(a))$, $\mathsf{hsame}(k, \mathsf{E0}(a'))$, and $LeftEq(a, a')$. Composing the symmetric carrier endpoint witness with the classifier endpoint witness gives $\mathsf{hsame}(\mathsf{E0}(c), \mathsf{E0}(a))$; $e0$ constructor inversion gives $\mathsf{hsame}(c, a)$. The only source-carrier rewrite in this proof is the checked finite-kernel history-sameness identity target BEDC.FKernel.Hist.hsame_iff_eq recorded at Definition 3.2; it turns the payload witness into kernel equality and transports $Left(c)$ to $Left(a)$. After that kernel transport, NameCert.carrier_respects_equiv applies from the left source certificate along the displayed source classifier $LeftEq(a, a')$ to obtain $Left(a')$, before rebuilding the target left tagged carrier using the stored endpoint witness $\mathsf{hsame}(k, \mathsf{E0}(a'))$. The right-right case is identical with payloads aligned through $e1$ constructor inversion and the right source certificate applied along the displayed source classifier $RightEq$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier transport from nameCert`

Theorem 35.39 (Sum classifier endpoint readback under source certificates)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, let* $h, k : \mathsf{BHist}$*, and assume source certificates* $cert_L : \mathsf{NameCert}(Left, LeftEq)$ *and* $cert_R :$

$\mathsf{NameCert}(Right, RightEq)$. *If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *and* $\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, R$
hold, then all of the following assertions hold:

1. $\mathsf{SumHistoryCarrier}(Left, Right, k)$.
2. *The source endpoint h satisfies the visible readback assertion*

$$(\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge (\forall l'.\ Left(l') \to \mathsf{hsame}(h, \mathsf{E0}(l')) \to \mathsf{hsame}(l, l')) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E}$$
$$\vee$$
$$(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)) \wedge (\forall r'.\ Right(r') \to \mathsf{hsame}(h, \mathsf{E1}(r')) \to \mathsf{hsame}(r, r')) \wedge \neg(\exists l.\ Left(l) \wedge \mathsf{hsame}($$

3. *The target endpoint k satisfies the visible readback assertion*

$$(\exists l.\ Left(l) \wedge \mathsf{hsame}(k, \mathsf{E0}(l)) \wedge (\forall l'.\ Left(l') \to \mathsf{hsame}(k, \mathsf{E0}(l')) \to \mathsf{hsame}(l, l')) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(k, \mathsf{E}$$
$$\vee$$
$$(\exists r.\ Right(r) \wedge \mathsf{hsame}(k, \mathsf{E1}(r)) \wedge (\forall r'.\ Right(r') \to \mathsf{hsame}(k, \mathsf{E1}(r')) \to \mathsf{hsame}(r, r')) \wedge \neg(\exists l.\ Left(l) \wedge \mathsf{hsame}($$

4. *The classifier pair satisfies the visible branch assertion*

$$(\exists a, b.\ Left(a) \wedge Left(b) \wedge \mathsf{hsame}(h, \mathsf{E0}(a)) \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a, b))$$
$$\vee$$
$$(\exists a, b.\ Right(a) \wedge Right(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a, b)).$$

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier source certified endpoint readback`

Proof. First obtain the target carrier. Apply Definition 35.38 to the given source carrier witness, the classifier witness, and the two certificates.

The branch-exactness theorem requires closure of $Left$ and $Right$ under hsame. For $Left$, suppose $Left(x)$ and $\mathsf{hsame}(x, y)$. The checked identity BEDC.FKernel.Hist.hsame_iff_eq turns the history-sameness witness into kernel equality, and transport along that equality gives $Left(y)$. The right closure is identical with $Right$.

Apply Definition 35.80 once to the given carrier witness for h and once to the target carrier witness for k. These are exactly the two displayed endpoint readback assertions.

Finally apply the forward direction of Definition 35.82 with the two closure hypotheses to the triple consisting of the source carrier, the target carrier, and the displayed classifier witness. The forward projection gives the displayed branch assertion, and the four components are conjoined in the listed order. □

Theorem 35.40 (Sum history classifier source-certified symmetry)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume source certificates* $\mathsf{NameCert}(Left, LeftEq)$ *and* $\mathsf{NameCert}(Right, RightEq)$. *If* $\mathsf{SumHistoryClassifier}(Left, R$
holds, then $\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, k, h)$ *holds.*

Proof. Invert the tagged classifier witness. In the left branch, swap the two visible $e0$ endpoint witnesses and apply the left source certificate field equiv_symm to the stored $LeftEq$ witness. In the right branch, swap the two visible $e1$ endpoint witnesses and apply the right source certificate field equiv_symm to the stored $RightEq$ witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier symm from nameCert`

Theorem 35.41 (Sum history classifier tag separation)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and let* $h, k, l, r : \mathsf{BHist}$*. Then*

$$\mathsf{hsame}(h, \mathsf{E0}(l)) \to \mathsf{hsame}(k, \mathsf{E1}(r)) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \bot.$$

Proof. Invert the classifier witness. In the left branch k is also E0-tagged, contradicting the displayed E1 endpoint witness. In the right branch h is E1-tagged, contradicting the displayed E0 endpoint witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier mixed tags absurd`

Theorem 35.42 (Sum history classifier reverse tag separation)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and let* $h, k, l, r : \mathsf{BHist}$*. Then*

$$\mathsf{hsame}(h, \mathsf{E1}(r)) \to \mathsf{hsame}(k, \mathsf{E0}(l)) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \bot.$$

Proof. Invert the classifier witness. In the left branch, the classifier forces h to be E0-tagged, contradicting the displayed E1 endpoint witness by E0/E1 no-confusion. In the right branch, the classifier forces k to be E1-tagged, contradicting the displayed E0 endpoint witness by E0/E1 no-confusion. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier reverse tag separation`

Theorem 35.43 (Sum history classifier transitivity)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and let* $h, k, u : \mathsf{BHist}$*. Assume*

$$\forall a\, b\, c : \mathsf{BHist}.\ LeftEq(a, b) \to LeftEq(b, c) \to LeftEq(a, c)$$

and

$$\forall a\, b\, c : \mathsf{BHist}.\ RightEq(a, b) \to RightEq(b, c) \to RightEq(a, c).$$

Then

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, k,$$

Proof. Invert both classifier witnesses. Matching left–left and right–right branches identify the shared tagged middle history by BEDC constructor inversion and compose the corresponding source-classifier witnesses. Mixed branches contradict the $e0/e1$ no-confusion rule. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier trans`

Theorem 35.44 (Sum history classifier source-certified stability fields)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers, and assume source certificates* $\mathsf{NameCert}(Left, LeftEq)$ *and* $\mathsf{NameCert}(Right, RightEq)$*. For the carrier* $\mathsf{SumHistoryCarrier}(Left, Right)$ *and classifier* $\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq)$*, the following stability fields hold:*

1. *carrier-relative reflexivity:*

$$\forall h.\ \mathsf{SumHistoryCarrier}(Left, Right, h) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, h);$$

2. *source-certified symmetry:*

$$\forall h\, k.\ \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, k, h);$$

3. *source-certified transitivity:*

$$\forall h\, k\, u.\ \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, k, u) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, u);$$

4. *classifier-induced carrier transport:*

$$\forall h\, k.\ \mathsf{SumHistoryCarrier}(Left, Right, h) \to \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryCarrier}(Left, Right, k).$$

Proof. For reflexivity, invert the displayed sum carrier witness. In the left branch, use the left source certificate reflexivity field on the source payload and rebuild the $e0$-tagged classifier witness with the same endpoint evidence; the right branch is the same with the right source certificate and $e1$ tag. The symmetry field is Theorem 35.40. The transitivity field is Theorem 35.43 applied with the transitivity fields of the two source certificates. The carrier transport field is Theorem 35.38. □

Lean4 ✓ `BEDC.Derived.SumUp.SumUp StdBridge`

Definition 35.45 (Sum stability certificate)**.** *The displayed checked theorem records the typed structural stability subcertificate:*

1. inl *and* inr *reflexivity*
2. *typed classifier transitivity*
3. *mixed* inl/inr *separation in both directions*

Carrier transport, endpoint transport, and history-level transitivity are supplied by Theorems 35.2, 35.37, and 35.43. Theorem 35.36 is the hsame *specialization of classifier symmetry. For the arbitrary source classifiers used by the concrete semantic certificate, Theorem 35.44 packages carrier-relative reflexivity, symmetry, transitivity, and classifier-induced carrier transport from the two source certificates. For the arbitrary source classifiers used by Theorem 35.108, Theorem 35.2 transports only a displayed carrier witness along* hsame.

Lean4 ✓ `BEDC.Derived.SumUp.sum stability certificate fields`

Theorem 35.46 (Sum classifier transitivity at the typed level)**.** *Let* $R_A : A \to A \to \mathsf{Prop}$ *and* $R_B : B \to B \to \mathsf{Prop}$ *be source classifier relations. If both* R_A *and* R_B *are transitive, then for sum carriers* x*,* y*, and* z*,*

$$\mathsf{SumClassifierSpec}(R_A, R_B, x, y) \quad \textit{and} \quad \mathsf{SumClassifierSpec}(R_A, R_B, y, z)$$

imply $\mathsf{SumClassifierSpec}(R_A, R_B, x, z)$*.*

Lean4 ✓ `BEDC.Derived.SumUp.SumClassifierSpec trans`

Proof. Invert the sum classifier witnesses by their constructors. Matching left and right branches compose through the corresponding source transitivity assumption, while mixed tags are impossible by the classifier clauses. □

Definition 35.47 (Sum ledger policy)**.** *Record a raw sum history carrying the selected injection tag and source ledger together with an* hsame *link to the visible history exposed by the ledger.*

Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryLedgerPolicy`

Theorem 35.48 (Sum ledger visible carrier)**.** *If a visible history is linked by* hsame *to a raw sum-ledger history, then the visible history is itself a sum carrier.*

Proof. Unpack the raw sum carrier and transport it across the stored hsame link by sum-carrier transport. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible carrier`

Theorem 35.49 (Sum ledger raw-visible classifier)**.** *If* $\mathsf{SumHistoryLedgerPolicy}(L, R, \rho, v)$ *holds, then* $\mathsf{SumHistoryClassifier}(L, R, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$ *holds.*

Proof. Unpack the ledger into its raw sum carrier and the stored $\mathsf{hsame}(\rho, v)$ link. The raw carrier determines whether ρ is left-tagged or right-tagged; transport the tag witness across the stored link and use reflexivity of hsame on the selected component to build the matching classifier branch. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy raw visible classifier`

Theorem 35.50 (Sum ledger history-same transport)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $raw, raw', visible, visible' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw, visible)$ $\mathsf{hsame}(raw, raw')$*, and* $\mathsf{hsame}(visible, visible')$ *hold, then*

$$\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw', visible')$$

holds.

Proof. Unpack the ledger into the raw SumHistoryCarrier witness and the stored link $\mathsf{hsame}(raw, visible)$. Transport the raw carrier along $\mathsf{hsame}(raw, raw')$ by Theorem 35.2. For the raw-visible link of the transported ledger, compose symmetry of $\mathsf{hsame}(raw, raw')$, the stored $\mathsf{hsame}(raw, visible)$ link, and $\mathsf{hsame}(visible, visible')$ to obtain $\mathsf{hsame}(raw', visible')$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy hsame transport`

Theorem 35.51 (Sum ledger classifier composition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds and* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, v, w)$ *holds, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, w)$$

holds.

Proof. First apply the raw-visible classifier theorem to the ledger policy, obtaining

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v).$$

Then compose this classifier witness with the given visible-target classifier witness using sum-history classifier transitivity. The branch classifiers are both hsame, so their required transitivity inputs are the left and right instances of BEDC history equality transitivity. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy classifier composition`

Theorem 35.52 (Sum ledger two-step classifier composition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *and* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ *hold, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, w)$$

holds. The roadmap fit is: traditional target disjoint-sum ledger-chain closure; BEDC source BHist *with tagged sum carrier; carrier* SumHistoryCarrier*; classifier* SumHistoryClassifier$(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; stability by two raw-visible ledger exposures and classifier transitivity; ledger two composable* SumHistoryLedgerPolicy *witnesses; core theorem this closure; exactness raw-to-middle-to-visible; standard bridge tagged sum ledger chain; status* ProofTarget*.*

Proof. Apply Theorem 35.89 to the two ledger witnesses, obtaining classifiers from ρ to v and from v to w. Compose them by Theorem 35.43; the branch classifiers are both hsame, so the required transitivity inputs are the left and right instances of BEDC history equality transitivity. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step classifier composition`

Theorem 35.53 (Sum ledger raw-visible branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $raw, visible : \mathsf{BHist}$ *be histories. Then*

$$\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw, visible) \iff (\exists l : \mathsf{BHist}.\ Left(l) \wedge \mathsf{hsame}(raw, \mathsf{E0}(l)) \wedge \mathsf{hsame}(visible, \mathsf{E0}(l$$

Proof. For the forward direction, unpack the ledger into its raw sum carrier and the stored raw-visible link. Split the raw carrier branch.

In the left branch, the carrier supplies $Left(l)$ and $\mathsf{hsame}(raw, \mathsf{E0}(l))$. Compose symmetry of the stored raw-visible link with this raw endpoint witness to obtain $\mathsf{hsame}(visible, \mathsf{E0}(l))$, and assemble the left disjunct.

In the right branch, the same argument uses the witness $Right(r)$ and $\mathsf{hsame}(raw, \mathsf{E1}(r))$ to obtain $\mathsf{hsame}(visible, \mathsf{E1}(r))$ and assemble the right disjunct.

For the reverse direction, split the displayed branch evidence. In the left branch, rebuild the raw sum carrier from $Left(l)$ and $\mathsf{hsame}(raw, \mathsf{E0}(l))$. Compose $\mathsf{hsame}(raw, \mathsf{E0}(l))$ with symmetry of $\mathsf{hsame}(visible, \mathsf{E0}(l))$ to recover the raw-visible ledger link.

The right branch is identical, rebuilding the raw carrier from $Right(r)$ and $\mathsf{hsame}(raw, \mathsf{E1}(r))$, then composing the two endpoint witnesses to recover the stored raw-visible link. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy raw visible branch exactness`

Theorem 35.54 (Sum ledger two-step branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *and* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ *hold, then the whole two-step chain has a visible same-branch payload readback:*

$$(\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l, l')) \vee (\exists r\, r' : \mathsf{BHist}.\ Right(r) \wedge Right(r')$$

The ten-field fit is: traditional target disjoint-sum ledger-chain readback; BEDC source BHist *with* hsame*; carrier* SumHistoryCarrier*; classifier unused in the final exactness statement but supplied by Theorem 35.92; ledger two composable* SumHistoryLedgerPolicy *witnesses; core theorem this branch exactness statement; exactness raw-to-visible same-branch payload readback; standard bridge tagged sum ledger chains; status* ProofTarget*.*

Proof. Apply Theorem 35.93 to each ledger witness. The first application exposes either a left payload visible at ρ and v, or a right payload visible at ρ and v; the second exposes the corresponding alternatives at v and w. Split the four branch cases. In the left-left case, use Theorem 35.79 at the shared middle endpoint v to identify the two left payloads and assemble the left disjunct. The right-right case is identical with the right branch. The mixed left-right and right-left cases contradict generated-history no-confusion between $\mathsf{E0}$ and $\mathsf{E1}$ at the shared middle endpoint v. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step branch exactness`

Theorem 35.55 (Sum ledger two-step endpoint exactness)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, w, l, l', r, r' : \mathsf{BHist}$ be histories. If $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ and $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ hold, then the two-ledger chain has endpoint exactness from ρ to w. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(w, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints of w read back source payload evidence:

$$(\mathsf{hsame}(w, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(w, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

Proof. Apply Theorem 35.92 to the two ledger witnesses and obtain $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame},$ The two mixed endpoint conclusions follow from Theorem 35.41; for the reversed displayed order, use classifier symmetry from Theorem 35.36 before applying the same tag-separation theorem. The left same-tag conclusion is exactly the left endpoint inversion supplied by Theorem 35.13, and the right same-tag conclusion is the corresponding right inversion from Theorem 35.14. For the final visible readback, apply Theorem 35.88 to the second ledger witness, obtaining a visible carrier for w; then apply Theorem 35.7 to the displayed $\mathsf{E0}$ or $\mathsf{E1}$ endpoint of w to extract the matching source payload evidence. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step endpoint exactness`

Definition 35.56 (Sum ledger finite provenance chain)**.** *For source carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$, define $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ as the finite provenance relation generated by the following two clauses. A displayed one-ledger witness $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, z)$ gives $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$. If there is a history $v : \mathsf{BHist}$ such that $\mathsf{SumHistoryLedgerPolicy}(Left, R$ and $\mathsf{SumHistoryLedgerChain}_{Left,Right}(v, z)$ hold, then $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ also holds.*

Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryLedgerChain`

Theorem 35.57 (Sum ledger chain transitivity)**.** *Let* $Left, Right$: BHist → Prop *be source carriers and let* ρ, m, z : BHist *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, m)$ *and* $\mathsf{SumHistoryLedgerChain}_{Left,Righ}$ *hold, then* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds. The ten-field fit is: traditional target finite tagged-sum ledger provenance composition; BEDC source* BHist *with tagged sum carrier; carrier* SumHistoryCarrier*; classifier supplied downstream by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem finite-chain composition; exactness concatenated raw-to-visible provenance; standard bridge tagged sum ledger audit composition; status* ProofTarget*.*

Proof. Induct on the first displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, m)$ witness.

In the one-ledger case, the displayed $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, m)$ is the first finite ledger step. Rebuild the target chain by the cons constructor, using that ledger step as the head and the second displayed chain from m to z as the tail.

In the cons case, the first chain consists of a head ledger from ρ to an intermediate history and a tail chain from that intermediate history to m. Apply the induction hypothesis to compose the tail chain with the second displayed chain from m to z, then rebuild the full chain by the cons constructor with the original head ledger. □

Theorem 35.58 (Sum ledger chain classifier composition)**.** *Let* $Left, Right$: BHist → Prop *be source carriers and let* $raw, visible$: BHist *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(raw, visible)$ *holds, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, raw, visible)$$

holds. The ten-field fit is: traditional target disjoint-sum finite ledger classifier closure; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; stability by raw-visible classification plus classifier transitivity; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this finite-chain classifier closure; exactness finite raw-to-visible classifier reachability; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(raw, visible)$ witness. In the one-ledger case, apply Theorem 35.89. In the cons case, apply the induction hypothesis to the tail chain to obtain the classifier from the middle history to *visible*, then compose it with the first ledger by Theorem 35.91. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain classifier composition`

Theorem 35.59 (Sum ledger chain visible carrier)**.** *Let* $Left, Right$: BHist → Prop *be source carriers and let* ρ, z : BHist *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then*

$$\mathsf{SumHistoryCarrier}(Left, Right, z)$$

holds. The ten-field fit is: traditional target disjoint-sum finite ledger visible-carrier closure; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier not needed for this unary closure; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this carrier closure; exactness final visible endpoint remains in the sum carrier; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ witness. In the one-ledger case, apply Theorem 35.88. In the cons case, use the carrier supplied by the tail induction hypothesis. □

Theorem 35.60 (Sum ledger chain endpoint exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z, l, l', r, r' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then the whole finite ledger provenance from* ρ *to* z *has endpoint exactness. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(z, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints of z *read back matching source payload evidence:*

$$(\mathsf{hsame}(z, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(z, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

The ten-field fit is: traditional target disjoint-sum finite ledger provenance; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier* SumHistoryClassifier$(Left, Right,$ hsame *supplied one ledger at a time by raw-visible classification; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$ *generated by* SumHistoryLedgerPolicy*; core theorem this chain endpoint exactness package; exactness mixed-tag separation, same-tag payload determinacy, and final visible source readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. First apply Theorem 35.98 to the displayed chain, obtaining SumHistoryClassifier$(Left, Right,$ hsame, hsame, The two mixed endpoint clauses follow from Theorem 35.41; for the reversed displayed order, first use classifier symmetry from Theorem 35.36. The same-tag endpoint clauses follow by transporting the classifier along the displayed endpoint hsame links and applying Theorem 35.13 or Theorem 35.14. For the final visible readback, apply Theorem 35.99 to the displayed chain to obtain SumHistoryCarrier$(Left, Right, z)$. Transport this carrier along the displayed hsame endpoint at z and apply the left or right carrier inversion theorem to extract the matching source payload evidence. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain endpoint exactness`

Theorem 35.61 (Sum ledger chain branch readback soundness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z : \mathsf{BHist}$ *be histories. The finite ledger-chain relation forces exact same-branch payload readback:*

$$\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z) \to (\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch readback; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier supplied by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this branch readback soundness; finite ledger-chain witnesses map into same-branch payload readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ witness. In the one-ledger case, apply Theorem 35.93 and choose the same payload at the raw and visible endpoints, with hsame reflexivity for the payload link.

In the cons case, combine the raw-visible branch exactness of the first ledger with the induction hypothesis for the tail chain. The mixed left-right and right-left cases contradict generated-history no-confusion between E0 and E1 at the shared middle endpoint. In the same-branch cases, use Theorem 35.79 at the middle endpoint and hsame transitivity to compose the first payload with the final payload. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain branch exactness`

Theorem 35.62 (Sum ledger chain branch exactness)**.** *Let $Left, Right$: BHist → Prop be source carriers and let ρ, z : BHist be histories. The finite ledger-chain relation is equivalent to exact same-branch payload readback:*

$$\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z) \iff (\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch readback; BEDC source BHist *with* hsame*; source carriers $Left$ and $Right$; carrier* SumHistoryCarrier*; classifier supplied by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this branch exactness; exactness coverage and completeness of the finite ledger-chain relation by same-branch payload readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. For the forward direction, apply Theorem 35.101.

For the reverse direction, split the displayed disjunction. In the left branch, the witnesses $Left(l)$, $\mathsf{hsame}(\rho, \mathsf{E0}(l))$, $\mathsf{hsame}(z, \mathsf{E0}(l'))$, and $\mathsf{hsame}(l, l')$ determine a one-ledger left readback. Use symmetry of $\mathsf{hsame}(l, l')$ and constructor congruence for E0 to transport $\mathsf{hsame}(z, \mathsf{E0}(l'))$ into $\mathsf{hsame}(z, \mathsf{E0}(l))$.

Apply the reverse direction of Theorem 35.93 to obtain $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, z)$, then use the one-ledger constructor of $\mathsf{SumHistoryLedgerChain}_{Left,Right}$.

The right branch is identical, using the witnesses $Right(r)$, $\mathsf{hsame}(\rho, \mathsf{E1}(r))$, $\mathsf{hsame}(z, \mathsf{E1}(r'))$, and $\mathsf{hsame}(r, r')$, with constructor congruence for E1 before applying the same raw-visible branch exactness theorem and the one-ledger chain constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain branch exactness iff`

Theorem 35.63 (Sum ledger chain raw branch forces final branch readback)**.** *Let $Left, Right$: BHist → Prop be source carriers and let ρ, z, l, r : BHist be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then raw visible branch evidence forces the matching final visible branch. The left clause is*

$$Left(l) \to \mathsf{hsame}(\rho, \mathsf{E0}(l)) \to \exists l' : \mathsf{BHist}.\ Left(l') \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l, l') \wedge \neg(\exists r' : \mathsf{BHist}.\ Right(r') \wedge \mathsf{hsa}$$

and the right clause is

$$Right(r) \to \mathsf{hsame}(\rho, \mathsf{E1}(r)) \to \exists r' : \mathsf{BHist}.\ Right(r') \wedge \mathsf{hsame}(z, \mathsf{E1}(r')) \wedge \mathsf{hsame}(r, r') \wedge \neg(\exists l' : \mathsf{BHist}.\ Left(l') \wedge \mathsf{h}$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch forcing; BEDC source BHist *with* hsame*; carrier final* SumHistoryCarrier*; classifier supplied by the chain classifier composition theorem; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this raw-branch forcing readback; exactness same-tag payload determinacy plus mixed-tag exclusion; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Apply Theorem 35.99 to the chain and then Theorem 35.80 to the final carrier. This splits the endpoint z into a left visible presentation with right-presentation exclusion, or a right visible presentation with left-presentation exclusion.

For the left raw premise, keep the left final presentation from the split. The same-tag payload clause of Theorem 35.100, applied to $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and the displayed $\mathsf{hsame}(z, \mathsf{E0}(l'))$, gives $\mathsf{hsame}(l, l')$; combine it with the source witness for l' and the carried right-presentation exclusion.

The right final presentation cannot occur under the left raw premise, because the mixed endpoint clause of Theorem 35.100 contradicts $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(z, \mathsf{E1}(r'))$. The right raw premise is symmetric: the right final presentation supplies the witness and the left-presentation exclusion, while the mixed endpoint clause excludes any left final presentation. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain raw branch final readback`

Theorem 35.64 (Sum ledger preserves visible tag separation)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, l, r : \mathsf{BHist}$ be histories. If $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ holds, then both visible mixed-tag endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l))) \to \bot.$$

Proof. Apply Theorem 35.89 to the ledger policy to obtain $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$. The left-to-right mixed endpoint pair is then discharged by $\mathsf{SumHistoryClassifier_mixed_tags_absurd}$ with the displayed $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(v, \mathsf{E1}(r))$ witnesses. For the symmetric right-to-left pair, use classifier symmetry from Theorem 35.36 and apply the same mixed-tag separation theorem to the exchanged endpoints. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible tag separation`

Theorem 35.65 (Sum ledger same-tag payload exactness)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, l, l', r, r' : \mathsf{BHist}$ be histories. If $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ holds, then the visible same-tag endpoint pairs determine their payloads:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Proof. Apply Theorem 35.89 to obtain the raw-visible $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$ witness. For the left-tagged endpoint pair, transport this witness along the displayed $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(v, \mathsf{E0}(l'))$ links by Theorem 35.37, then apply the left inversion Theorem 35.13 to extract $\mathsf{hsame}(l, l')$. The right-tagged case is identical with Theorem 35.14. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy same tag payload exactness`

Theorem 35.66 (Sum ledger visible payload readback)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, l, r : \mathsf{BHist}$ be histories. If $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ holds, then visible tagged endpoints read back source payload evidence:*

$$(\mathsf{hsame}(v, \mathsf{E0}(l)) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l, a))$$

and

$$(\mathsf{hsame}(v, \mathsf{E1}(r)) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r, b)).$$

Proof. The traditional target is disjoint-sum ledger readback; the BEDC source is BHist; the carrier is SumHistoryCarrier; no classifier is needed for this carrier-readback obligation; the ledger is SumHistoryLedgerPolicy; the exactness principle is visible-tag carrier readback; the status is ProofTarget. Apply Theorem 35.88 to the ledger, transport the visible carrier witness along the displayed $\mathsf{hsame}(v, \mathsf{E0}(l))$ or $\mathsf{hsame}(v, \mathsf{E1}(r))$ endpoint equality, and then apply Theorem 35.7 to extract the left or right source payload witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible payload readback`

Theorem 35.67 (Sum ledger one-step endpoint exactness)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers and let $\rho, v, l, l', r, r' : \mathsf{BHist}$ be histories. If $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ holds, then the one-ledger link has endpoint exactness from ρ to v. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints read back source payload evidence:

$$(\mathsf{hsame}(v, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(v, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

The roadmap ten-field fit is: traditional target disjoint-sum one-ledger endpoint audit; BEDC source BHist *with* hsame*; source carriers $Left$ and $Right$; carrier* SumHistoryCarrier*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; ledger* SumHistoryLedgerPolicy*; core theorem this endpoint exactness package; exactness raw-to-visible tag separation, same-tag payload determinacy, and visible source readback; standard bridge tagged sum ledger readback; status* ProofTarget*.*

Proof. Apply Theorem 35.104 to the ledger witness for the two mixed clauses. Apply Theorem 35.105 to the same ledger witness for the two same-tag payload clauses. Apply Theorem 35.106 to the ledger witness for the two final visible readback clauses. □

Theorem 35.68 (Concrete $\mathsf{Sum}^{\uparrow}$ semantic name certificate)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers with classifiers $LeftEq$ and $RightEq$. If $\mathsf{NameCert}(Left, LeftEq)$ and $\mathsf{NameCert}(Right, RightEq)$ are supplied, then the tagged sum carrier, the tagged sum classifier, and the carrier-as-pattern and carrier-as-ledger tuple form*

$$\begin{aligned}
&\mathsf{SemanticNameCert}(\\
&\quad \mathsf{SumHistoryCarrier}(Left, Right),\\
&\quad \mathsf{SumHistoryCarrier}(Left, Right),\\
&\quad \mathsf{SumHistoryCarrier}(Left, Right),\\
&\quad \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq)).
\end{aligned}$$

Proof. The carrier is represented by $e0$-tagged left histories and $e1$-tagged right histories. Theorem 35.11 marks the split: the classifier delegates within matching tags to the corresponding source certificate, while source-carrier evidence is taken from SumHistoryCarrier when certificate fields require it. Reflexivity is built by tagging the source certificate reflexivity field on the selected component. Symmetry and transitivity invert the tagged classifier witnesses, use the matching source certificate fields inside a left–left or right–right branch, and discharge mixed tags by constructor no-confusion.

The core certificate fields are assembled from Theorem 35.44. That theorem supplies carrier-relative reflexivity, source-certified symmetry, source-certified transitivity, and classifier-induced carrier transport for the tagged carrier and tagged classifier. The inhabited field is separate: take the inhabited witness l from the left source certificate, choose the visible endpoint $\mathsf{E0}(l)$, and rebuild $\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l))$ by the left tagged constructor with the reflexive endpoint witness and the stored $Left(l)$ evidence.

Inside the carrier-transport field supplied by the cited theorem, source-carrier membership is not transported along an arbitrary source NameCert merely from $\mathsf{hsame}(l, a)$. In the left branch, constructor inversion first aligns the tagged payloads and obtains $\mathsf{hsame}(l, a)$; the checked finite-kernel identity BEDC.FKernel.Hist.hsame_iff_eq converts that history-sameness witness to kernel equality before the stored $Left(l)$ evidence is rewritten as $Left(a)$. Only after this equality rewrite does NameCert.*carrier_respects_equiv* from the left source certificate apply along the displayed source classifier witness $LeftEq(a, b)$, yielding $Left(b)$ and hence the target left tagged carrier. The right branch is the same with $RightEq(a, b)$ and the right source certificate. Pattern soundness and ledger soundness are identity implications, since both semantic fields are the same predicate $\mathsf{SumHistoryCarrier}(Left, Right)$. □

Lean4 ✓ `BEDC.Derived.SumUp.sum history semantic name certificate`

Corollary 35.69 ($\mathsf{Sum}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{Sum}^{\uparrow}$ *is licensed as a derived disjoint sum, not as a primitive coproduct.*

Theorem 35.70 (Sum public tagged-payload standard bridge)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *equipped with source classifiers and source* NameCert *rows, the public* $\mathsf{Sum}^{\uparrow}$ *semantic certificate presents exactly the standard finite disjoint-sum interface with two visible injections. A public endpoint is either an* E0*-tagged left payload or an* E1*-tagged right payload. Endpoints with the same tag compare by the corresponding source classifier, and endpoints with mixed tags are separated by constructor no-confusion.*

The exported ledger surface is precisely the visible tag, payload readback, raw-visible branch exactness, and endpoint exactness already recorded by the $\mathsf{Sum}^{\uparrow}$ *rows. No consumer receives an additional coproduct eliminator, hidden branch coordinate, host disjoint-union type, or classifier other than the branchwise tagged classifier.*

Lean4 ✓ `BEDC.Derived.SumUp.SumUp concrete to schema`

Proof. The public certificate is the one supplied by Definition 35.108. Its carrier consists of histories built by the two visible constructors, so constructor inversion gives the two alternatives $\mathsf{E0}(l)$ and $\mathsf{E1}(r)$ and gives no third endpoint form.

Within a left–left comparison, the tagged classifier in Definition 35.108 projects to the left source classifier; within a right–right comparison, it projects to the right source classifier. The branch exactness statement in Definition 35.93 identifies those projections with the raw visible branch rows exported by the ledger.

For mixed tags, Definition 35.104 applies to the visible ledger row and reduces equality or classifier compatibility of opposite tags to constructor no-confusion. Thus a left endpoint and a right endpoint remain separated in the public reading.

Finally, Definition 35.106 recovers the displayed source payload from each visible endpoint, and Definition 35.100 states that the endpoint readback is exact along the public chain. These rows are exactly the finite two-injection interface described above, so the tagged-payload certificate is a paper bridge to the standard disjoint-sum reading. □

Theorem 35.71 (Sum mature finite consumer boundary)**.** *Every finite consumer of the public* $\mathsf{Sum}^{\uparrow}$ *package factors through the tagged payload carrier, visible tag separation, branch exactness, payload readback, endpoint chain exactness, and the public tagged-payload standard bridge. Such a consumer can inspect only whether the exposed endpoint is an* $\mathsf{E0}$*-tagged left payload or an* $\mathsf{E1}$*-tagged right payload, can read the corresponding source evidence through the ledger rows, and can compare same-tag payloads only through the branch classifier exactness already exported by the certificate.*

This finite boundary excludes eliminators into arbitrary certified targets, dependent sums, and categorical coproduct universal properties. Those structures are not consequences of the tagged-payload package surface; they require additional consumer principles beyond the public $\mathsf{Sum}^{\uparrow}$ *certificate.*

Proof. Start with the public tagged-payload bridge in Theorem 35.110. It identifies the public surface with the two visible injections and rules out any hidden carrier coordinate outside the tagged history endpoint.

The finite consumer's branch observation is then forced by Theorem 35.104; mixed visible tags cannot be read as the same public branch. Same-branch comparisons are governed by Theorem 35.93, which is the raw-visible branch exactness row for the ledger.

Payload access is supplied by Theorem 35.106. Chain-level endpoint soundness is supplied by Theorem 35.100, so repeated finite ledger reads still factor through the same tagged endpoint and its source payload evidence.

Combining these rows leaves exactly the finite tagged-payload consumer surface. Since none of the cited rows constructs an arbitrary-target eliminator, dependent sum, or coproduct universal arrow, those stronger interfaces remain outside the mature finite boundary stated here. □

Theorem 35.72 (Sum finite-data scoped-exit package)**.** *The scoped finite-data exit surface of* $\mathsf{Sum}^{\uparrow}$ *is exactly the package consisting of the tagged semantic certificate of Definition 35.108, the selected visible injection tag, left and right branch exactness, the source ledger rows, visible-history descent through endpoint readback, and the mature finite consumer boundary of Definition 35.111. The selected injection tag is the two-branch finite-choice row already licensed by chapter 32;* $\mathsf{Sum}^{\uparrow}$ *consumes that sibling only as visible tag discipline, while the left and right payload rows remain governed by the local sum carrier and classifier. A consumer exiting the* $\mathsf{Sum}^{\uparrow}$ *chapter through this surface may inspect the selected tag, recover the corresponding source payload and source ledger evidence, compare same-tag endpoints through the displayed branch classifier, and replay the visible history descent. It receives no hidden coproduct eliminator, no dependent target recursion, and no classifier beyond the tagged finite-data rows already exported by the semantic certificate.*

Proof. Start with the semantic certificate in Definition 35.108. Its carrier is the tagged finite packet, so each public endpoint enters through an $\mathsf{E0}$ or $\mathsf{E1}$ injection rather than through a host coproduct object.

The selected tag is governed by Definition 35.104. The $\mathsf{Bool}^{\uparrow}$ reference supplies only the finite $b0/b1$ tag reading for this visible branch choice; it does not introduce a host coproduct, an eliminator, or an external Boolean equality. The same-tag clauses are governed by Definition 35.93, while visible payload recovery is governed by Definition 35.106 and endpoint descent by Definition 35.100.

Finally apply the consumer boundary in Definition 35.111. It states that every finite consumer factors through exactly these tagged payload, branch exactness, source ledger, and visible-history rows. Therefore the package is a scoped exit surface for finite data rather than a generic coproduct interface. □

Theorem 35.73 (Sum classifier transitivity at the typed level)**.** *Let $R_A : A \to A \to \mathsf{Prop}$ and $R_B : B \to B \to \mathsf{Prop}$ be source classifier relations. If both R_A and R_B are transitive, then for sum carriers x, y, and z,*

$$\mathsf{SumClassifierSpec}(R_A, R_B, x, y) \quad \textit{and} \quad \mathsf{SumClassifierSpec}(R_A, R_B, y, z)$$

imply $\mathsf{SumClassifierSpec}(R_A, R_B, x, z)$.

Lean4 ✓ `BEDC.Derived.SumUp.SumClassifierSpec trans`

Proof. Invert the sum classifier witnesses by their constructors. Matching left and right branches compose through the corresponding source transitivity assumption, while mixed tags are impossible by the classifier clauses. □

Theorem 35.74 (Sum history classifier branch partition)**.** *For branch carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$, branch classifiers $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$, and histories h, k, the history-level sum classifier exposes the visible left-or-right constructor analysis:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to (\exists l, l'.\ \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge \mathsf{hsame}(k, \mathsf{E0}(l')) \wedge LeftEq(l, l')$$

In the ten-field fit, the traditional target is disjoint-sum branch readback, the BEDC source is BHist*, the classifier is* $\mathsf{SumHistoryClassifier}$*, the ledger records the visible tag plus payload, and the status is* $\mathsf{ProofTarget}$.

Proof. Case-analyze the classifier witness. The left constructor supplies the two $\mathsf{E0}$ endpoint links and the stored $LeftEq$ payload relation, yielding the left disjunct. The right constructor supplies the two $\mathsf{E1}$ endpoint links and the stored $RightEq$ payload relation, yielding the right disjunct. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier branch partition`

Theorem 35.75 (Sum history classifier branch exactness)**.** *For branch carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$, branch classifiers $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$, and histories h, k, the history-level sum classifier is exactly the visible left-or-right branch disjunction:*

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \iff (\exists l, l'.\ \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge \mathsf{hsame}(k, \mathsf{E0}(l')) \wedge LeftEq(l, l$$

In the ten-field fit, the exactness target records that classifier witnesses and visible same-branch endpoint data determine each other.

Proof. For the forward direction, apply Theorem 35.74. It case-analyzes the classifier witness and returns the displayed left-or-right branch data.

For the reverse direction, split the displayed disjunction. In the left branch, unpack l, l' together with the two visible left endpoint witnesses and the stored $LeftEq(l, l')$ witness, then rebuild the left classifier constructor. In the right branch, unpack r, r' together with the two visible right endpoint witnesses and the stored $RightEq(r, r')$ witness, then rebuild the right classifier constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier branch iff`

Theorem 35.76 (Sum history carrier visible branch cases)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *holds, then the visible history has one of the two displayed branch witnesses:*

$$(\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \vee (\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r))).$$

In the ten-field fit, the traditional target is disjoint-sum carrier readback, the BEDC source is BHist*, the carrier is* SumHistoryCarrier*, no classifier is needed, the stability obligation is constructor inversion, the ledger records the visible tag plus payload, the core theorem is branch cases, exactness coverage is supplied by the visible branch witnesses, the standard bridge is* inl *and* inr*, and the status is* ProofTarget*.*

Proof. Case-analyze the generated SumHistoryCarrier witness. The left constructor returns a payload l, the source-carrier witness $Left(l)$, and the displayed endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$. The right constructor returns the corresponding payload r, the source-carrier witness $Right(r)$, and the displayed endpoint link $\mathsf{hsame}(h, \mathsf{E1}(r))$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible branch cases`

Theorem 35.77 (Sum history carrier visible branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. Then the sum history carrier is exactly the visible left-or-right branch assertion:*

$$\mathsf{SumHistoryCarrier}(Left, Right, h) \iff (\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \vee (\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r))).$$

In the ten-field fit, the carrier coverage predicate and the visible branch classifier agree on precisely the histories generated by the two sum constructors.

Proof. For the forward direction, apply Definition 35.76. For the reverse direction, split the displayed disjunction. In the left case, unpack the payload l, the source-carrier witness $Left(l)$, and the visible endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$, then rebuild the left carrier constructor. In the right case, unpack the corresponding data for r and rebuild the right carrier constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible branch exactness`

Theorem 35.78 (Sum history carrier exclusive visible branch partition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers. Then the sum history carrier has exactly one visible branch presentation:*

$$\mathsf{SumHistoryCarrier}(Left, Right, h) \iff \Big((\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l))) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)))\Big) \vee \Big((\exists r.\ R$$

In the ten-field fit, the carrier readback for $\mathsf{Sum}^{\uparrow}$ *is a constructor partition: every carrier history is visibly generated by one sum constructor, and the opposite visible constructor presentation is excluded.*

Proof. For the forward direction, use Definition 35.77 to obtain either a visible left presentation or a visible right presentation. In the left case, the opposite right presentation would contradict Definition 35.3 after reordering the stored endpoint and source-carrier witnesses; the right case is symmetric. For the reverse direction, split the displayed disjunction, forget the negative conjunct, and rebuild $\mathsf{SumHistoryCarrier}(Left, Right, h)$ through Definition 35.77. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier exclusive visible branch partition payload first`

Theorem 35.79 (Sum visible branch payload determinism)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $h, l, l', r, r' : \mathsf{BHist}$ *be histories. Same-branch visible presentations have unique payloads up to* hsame*:*

$$\mathsf{hsame}(h, \mathsf{E0}(l)) \to Left(l) \to \mathsf{hsame}(h, \mathsf{E0}(l')) \to Left(l') \to \mathsf{hsame}(l, l')$$

and

$$\mathsf{hsame}(h, \mathsf{E1}(r)) \to Right(r) \to \mathsf{hsame}(h, \mathsf{E1}(r')) \to Right(r') \to \mathsf{hsame}(r, r').$$

In the ten-field fit, the traditional target is disjoint-sum visible readback determinism; the BEDC source is BHist*; the carrier is* SumHistoryCarrier *presentations; the classifier is* hsame *on visible constructors; the stability obligation is constructor inversion; the ledger records the visible tag plus payload source evidence; the core theorem is same-branch payload uniqueness; exactness is deterministic readback; the standard bridge is* inl *and* inr*; and the status is* ProofTarget*.*

Proof. For the left statement, use symmetry on the first endpoint link and compose it with the second endpoint link, obtaining $\mathsf{hsame}(\mathsf{E0}(l), \mathsf{E0}(l'))$. Constructor inversion for the visible left constructor gives $\mathsf{hsame}(l, l')$. The right statement is identical, composing the two E1 endpoint links and applying right-constructor inversion. The source-carrier witnesses record that both visible presentations are legitimate branch presentations; they are not needed by the constructor inversion itself. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier visible payload determinism`

Theorem 35.80 (Sum carrier single-valued visible branch readback)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $h : \mathsf{BHist}$*. If* $\mathsf{SumHistoryCarrier}(Left, Right, h)$ *holds, then the visible tagged payload presentation of* h *is single-valued:*

$$\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}(l)) \wedge (\forall l'.\ Left(l') \to \mathsf{hsame}(h, \mathsf{E0}(l')) \to \mathsf{hsame}(l, l')) \wedge \neg(\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r$$

or

$$\exists r.\ Right(r) \wedge \mathsf{hsame}(h, \mathsf{E1}(r)) \wedge (\forall r'.\ Right(r') \to \mathsf{hsame}(h, \mathsf{E1}(r')) \to \mathsf{hsame}(r, r')) \wedge \neg(\exists l.\ Left(l) \wedge \mathsf{hsame}(h, \mathsf{E0}$$

In the ten-field fit, the traditional target is tagged disjoint-sum readback; the BEDC source is BHist*; the carrier is* SumHistoryCarrier*; the classifier is* hsame *on same-tag visible payloads; stability is constructor inversion plus branch exclusion; the ledger stores the visible tag and payload source evidence; the core theorem is this single-valued readback; exactness is existence, uniqueness, and opposite-tag exclusion; the standard bridge is* inl *and* inr*; and the status is* ProofTarget*.*

Proof. Apply Definition 35.78 to the carrier witness. In the left branch, unpack l, $Left(l)$, the endpoint link $\mathsf{hsame}(h, \mathsf{E0}(l))$, and the exclusion of right visible presentations. For an arbitrary left visible payload l', apply Definition 35.79 to the two left endpoint links and the two source-carrier witnesses to obtain $\mathsf{hsame}(l, l')$. The right branch is symmetric, using the right payload clause and carrying the exclusion of left visible presentations unchanged. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryCarrier single valued visible readback`

Theorem 35.81 (Sum history classifier carrier-aware branch partition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier closure under history sameness:*

$$\forall x\, y.\ Left(x) \to \mathsf{hsame}(x, y) \to Left(y)$$

and

$$\forall x\, y.\ Right(x) \to \mathsf{hsame}(x, y) \to Right(y).$$

If $\mathsf{SumHistoryCarrier}(Left, Right, h)$, $\mathsf{SumHistoryCarrier}(Left, Right, k)$, *and* $\mathsf{SumHistoryClassifier}(Left, Right, Left$ *hold, then a visible classifier side supplies the branch data used by the carrier-backed certificate analysis:*

$$\exists a, b.\ Left(a) \wedge Left(b) \wedge \mathsf{hsame}(h, \mathsf{E0}(a)) \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a, b)$$

or

$$\exists a, b.\ Right(a) \wedge Right(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a, b).$$

Proof. Split the displayed SumHistoryClassifier witness into its visible left or right constructor branch. In the left branch, let the classifier payloads be a, b. Compose the carrier endpoint for h with the classifier endpoint $\mathsf{hsame}(h, \mathsf{E0}(a))$ and apply SumHistoryCarrier_e0_inversion to obtain a source payload l, $Left(l)$, and a payload link $\mathsf{hsame}(l, a)$; the same argument at k gives l', $Left(l')$, and $\mathsf{hsame}(l', b)$. The left source-carrier closure turns these into $Left(a)$ and $Left(b)$. Together with the stored $LeftEq(a, b)$ witness and the two classifier endpoint links, these data give the left disjunct. In the right branch, apply the analogous E1 endpoint composition and SumHistoryCarrier_e1_inversion argument to obtain $Right(a)$ and $Right(b)$ from the right source-carrier closure, then pair them with the stored $RightEq(a, b)$ witness and the two right endpoint links. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier aware branch partition`

Theorem 35.82 (Sum history classifier carrier-aware branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, let* $LeftEq, RightEq : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier closure under history sameness:*

$$\forall x\, y.\ Left(x) \to \mathsf{hsame}(x, y) \to Left(y)$$

and

$$\forall x\, y.\ Right(x) \to \mathsf{hsame}(x, y) \to Right(y).$$

Then the carrier-backed classifier assertion

$$\begin{aligned}&\mathsf{SumHistoryCarrier}(Left, Right, h) \wedge \mathsf{SumHistoryCarrier}(Left, Right, k) \wedge \\ &\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k)\end{aligned}$$

is equivalent to the visible branch assertion

$$\begin{aligned}&\big(\exists a, b.\ Left(a) \wedge Left(b) \wedge \mathsf{hsame}(h, \mathsf{E0}(a)) \wedge \mathsf{hsame}(k, \mathsf{E0}(b)) \wedge LeftEq(a, b)\big) \\ &\vee \\ &\big(\exists a, b.\ Right(a) \wedge Right(b) \wedge \mathsf{hsame}(h, \mathsf{E1}(a)) \wedge \mathsf{hsame}(k, \mathsf{E1}(b)) \wedge RightEq(a, b)\big).\end{aligned}$$

Proof. For the forward direction, apply Definition 35.81 with the two displayed source-carrier closure hypotheses. For the reverse direction, split the displayed branch assertion. In the left branch, unpack a, b with their $Left$ witnesses, the two visible E0 endpoint links, and the stored $LeftEq(a, b)$ witness. The same data rebuilds the two SumHistoryCarrier witnesses by the left carrier constructor and rebuilds the matching SumHistoryClassifier witness by the left classifier constructor. The right branch is identical, using the two $Right$ witnesses, the visible E1 endpoint links, $RightEq(a, b)$, and the right carrier and classifier constructors. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier carrier aware branch exactness`

Theorem 35.83 (Sum history source weakening)**.** *Let* $Left, Right, Left', Right' : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. The two refinements have separate premises. Source-carrier inclusion gives carrier refinement:*

$$((\forall h.\ Left(h) \to Left'(h)) \wedge (\forall h.\ Right(h) \to Right'(h))) \to (\mathsf{SumHistoryCarrier}(Left, Right, h) \to \mathsf{SumHistoryCarr}$$

Component-relation weakening gives classifier refinement:

$$((\forall a\, b.\ LeftEq(a,b) \to LeftEq'(a,b)) \wedge (\forall a\, b.\ RightEq(a,b) \to RightEq'(a,b))) \to (\mathsf{SumHistoryClassifier}(Left, Rig$$

In the ten-field fit, the traditional target is disjoint-sum stability under source refinement, the BEDC source is the two BHist *carriers, the carrier is* SumHistoryCarrier*, the component classifier is* SumHistoryClassifier*, the stability datum is source inclusion plus component-relation weakening, the ledger keeps the visible tag unchanged, and the core theorem separates carrier weakening from classifier weakening. The classifier part uses only component-relation weakening because source membership is not a field of* SumHistoryClassifier*. Exactness is preserved by the same endpoint witnesses, the standard bridge is the* inl *and* inr *restriction, and the status is* ProofTarget*.*

Proof. For the carrier claim, assume the displayed pair of source-carrier inclusions and split the SumHistoryCarrier witness into its left and right constructor cases. The endpoint history witness is reused unchanged, while the matching source-carrier inclusion maps the payload membership proof into the refined source carrier. For the classifier claim, assume the displayed pair of component-relation weakenings and split the SumHistoryClassifier witness into its left and right constructor cases. Reuse the same two visible endpoints and apply only the matching component-relation weakening hypothesis to the stored $LeftEq$ or $RightEq$ witness. Source membership is absent from SumHistoryClassifier, so no source-carrier inclusion is needed for the classifier paragraph. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistorySource weakening`

Theorem 35.84 (Sum history source equivalence)**.** *Let* $Left, Right, Left', Right' : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be source classifiers. Assume source-carrier equivalence*

$$\forall h.\ Left(h) \leftrightarrow Left'(h)$$

and

$$\forall h.\ Right(h) \leftrightarrow Right'(h),$$

and assume component-relation equivalence

$$\forall a\, b.\ LeftEq(a,b) \leftrightarrow LeftEq'(a,b)$$

and

$$\forall a\, b.\ RightEq(a,b) \leftrightarrow RightEq'(a,b).$$

Then the sum carrier and classifier are invariant under the equivalent source presentation:

$$(\forall h.\ \mathsf{SumHistoryCarrier}(Left, Right, h) \leftrightarrow \mathsf{SumHistoryCarrier}(Left', Right', h)) \wedge (\forall h\, k.\ \mathsf{SumHistoryClassifier}(Left, R$$

In the ten-field fit, the traditional target is disjoint-sum source-presentation invariance, the BEDC source is the two BHist *carriers, the carrier is* SumHistoryCarrier*, the classifier is* SumHistoryClassifier*, the stability datum is source and relation equivalence, the ledger keeps the visible tag unchanged, and the core theorem is carrier and classifier equivalence. Exactness uses the unchanged branch witnesses, the standard bridge is* inl *and* inr *readback, and the status is* ProofTarget*.*

Proof. Apply Definition 35.83 from left to right using the forward halves of the two source-carrier equivalences for the carrier clause and the forward halves of the two component-relation equivalences for the classifier clause. This gives the implications from $(Left, Right, LeftEq, RightEq)$ to $(Left', Right', LeftEq', RightEq')$, with the same visible tag and the same branch witnesses. Apply the same weakening theorem from right to left using the reverse halves of the four equivalence hypotheses. Pair the two carrier implications and pair the two classifier implications. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistorySource equivalence`

Theorem 35.85 (Sum history classifier relation weakening)**.** *Let* $Left, Right, LeftAlt, RightAlt : \mathsf{BHist} \to \mathsf{Prop}$ *be arbitrary source carriers, and let* $LeftEq, RightEq, LeftEq', RightEq' : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be component classifiers. Assume component-relation weakening*

$$\forall a\, b.\ LeftEq(a, b) \to LeftEq'(a, b)$$

and

$$\forall a\, b.\ RightEq(a, b) \to RightEq'(a, b).$$

Then the sum history classifier relation weakens independently of source-carrier inclusion:

$$\mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq, h, k) \to \mathsf{SumHistoryClassifier}(LeftAlt, RightAlt, LeftEq', Righ$$

In the ten-field fit, the traditional target is disjoint-sum classifier monotonicity, the BEDC source is BHist*, the classifier is* SumHistoryClassifier*, the stability datum is component-relation weakening, the ledger keeps the visible tag unchanged, exactness is branch partition preservation, and the status is* ProofTarget*.*

Proof. Split the displayed SumHistoryClassifier witness into its visible left or right constructor branch. In each branch, reuse the same visible endpoint witnesses for h and k. The left branch applies the left component-relation weakening hypothesis to the stored $LeftEq$ witness and rebuilds the left tagged classifier witness; the right branch applies the right component-relation weakening hypothesis to the stored $RightEq$ witness and rebuilds the right tagged classifier witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryClassifier relation weakening`

Theorem 35.86 (Sum ledger raw-visible branch partition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then the same visible branch and payload appear on both ends of the ledger link:*

$$(\exists l.\ Left(l) \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l))) \vee (\exists r.\ Right(r) \wedge \mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r))).$$

In the ten-field fit, the traditional target is disjoint-sum ledger branch readback; the BEDC source is BHist *with* hsame*; the carrier is* SumHistoryCarrier*; no classifier is needed for the direct branch exposure; the ledger is* SumHistoryLedgerPolicy*; exactness is the same raw-visible branch and payload on both endpoints; and the status is* ProofTarget*.*

Proof. Unpack the ledger into the raw SumHistoryCarrier witness and the stored raw-visible $\mathsf{hsame}(\rho, v)$ link. Split the raw carrier. In the left branch, keep the raw payload l and its $Left(l)$ witness, retain the stored raw tag witness $\mathsf{hsame}(\rho, \mathsf{E0}(l))$, and compose symmetry of $\mathsf{hsame}(\rho, v)$ with that raw tag witness to obtain $\mathsf{hsame}(v, \mathsf{E0}(l))$. The right branch uses the same argument with the raw payload r, the $Right(r)$ witness, and the E1 tag. □

Definition 35.87 (Sum ledger policy)**.** *Record a raw sum history carrying the selected injection tag and source ledger together with an* hsame *link to the visible history exposed by the ledger.*

Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryLedgerPolicy`

Theorem 35.88 (Sum ledger visible carrier)**.** *If a visible history is linked by* hsame *to a raw sum-ledger history, then the visible history is itself a sum carrier.*

Proof. Unpack the raw sum carrier and transport it across the stored hsame link by sum-carrier transport. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible carrier`

Theorem 35.89 (Sum ledger raw-visible classifier)**.** *If* $\mathsf{SumHistoryLedgerPolicy}(L, R, \rho, v)$ *holds, then* $\mathsf{SumHistoryClassifier}(L, R, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$ *holds.*

Proof. Unpack the ledger into its raw sum carrier and the stored $\mathsf{hsame}(\rho, v)$ link. The raw carrier determines whether ρ is left-tagged or right-tagged; transport the tag witness across the stored link and use reflexivity of hsame on the selected component to build the matching classifier branch. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy raw visible classifier`

Theorem 35.90 (Sum ledger history-same transport)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $raw, raw', visible, visible' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw, visible)$ $\mathsf{hsame}(raw, raw')$*, and* $\mathsf{hsame}(visible, visible')$ *hold, then*

$$\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw', visible')$$

holds.

Proof. Unpack the ledger into the raw SumHistoryCarrier witness and the stored link $\mathsf{hsame}(raw, visible)$. Transport the raw carrier along $\mathsf{hsame}(raw, raw')$ by Theorem 35.2. For the raw-visible link of the transported ledger, compose symmetry of $\mathsf{hsame}(raw, raw')$, the stored $\mathsf{hsame}(raw, visible)$ link, and $\mathsf{hsame}(visible, visible')$ to obtain $\mathsf{hsame}(raw', visible')$. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy hsame transport`

Theorem 35.91 (Sum ledger classifier composition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds and* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, v, w)$ *holds, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, w)$$

holds.

Proof. First apply the raw-visible classifier theorem to the ledger policy, obtaining

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v).$$

Then compose this classifier witness with the given visible-target classifier witness using sum-history classifier transitivity. The branch classifiers are both hsame, so their required transitivity inputs are the left and right instances of BEDC history equality transitivity. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy classifier composition`

Theorem 35.92 (Sum ledger two-step classifier composition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *and* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ *hold, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, w)$$

holds. The roadmap fit is: traditional target disjoint-sum ledger-chain closure; BEDC source BHist *with tagged sum carrier; carrier* SumHistoryCarrier*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; stability by two raw-visible ledger exposures and classifier transitivity; ledger two composable* SumHistoryLedgerPolicy *witnesses; core theorem this closure; exactness raw-to-middle-to-visible; standard bridge tagged sum ledger chain; status* ProofTarget.

Proof. Apply Theorem 35.89 to the two ledger witnesses, obtaining classifiers from ρ to v and from v to w. Compose them by Theorem 35.43; the branch classifiers are both hsame, so the required transitivity inputs are the left and right instances of BEDC history equality transitivity. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step classifier composition`

Theorem 35.93 (Sum ledger raw-visible branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $raw, visible : \mathsf{BHist}$ *be histories. Then*

$$\mathsf{SumHistoryLedgerPolicy}(Left, Right, raw, visible) \iff (\exists l : \mathsf{BHist}.\ Left(l) \wedge \mathsf{hsame}(raw, \mathsf{E0}(l)) \wedge \mathsf{hsame}(visible, \mathsf{E0}(l$$

Proof. For the forward direction, unpack the ledger into its raw sum carrier and the stored raw-visible link. Split the raw carrier branch.

In the left branch, the carrier supplies $Left(l)$ and $\mathsf{hsame}(raw, \mathsf{E0}(l))$. Compose symmetry of the stored raw-visible link with this raw endpoint witness to obtain $\mathsf{hsame}(visible, \mathsf{E0}(l))$, and assemble the left disjunct.

In the right branch, the same argument uses the witness $Right(r)$ and $\mathsf{hsame}(raw, \mathsf{E1}(r))$ to obtain $\mathsf{hsame}(visible, \mathsf{E1}(r))$ and assemble the right disjunct.

For the reverse direction, split the displayed branch evidence. In the left branch, rebuild the raw sum carrier from $Left(l)$ and $\mathsf{hsame}(raw, \mathsf{E0}(l))$. Compose $\mathsf{hsame}(raw, \mathsf{E0}(l))$ with symmetry of $\mathsf{hsame}(visible, \mathsf{E0}(l))$ to recover the raw-visible ledger link.

The right branch is identical, rebuilding the raw carrier from $Right(r)$ and $\mathsf{hsame}(raw, \mathsf{E1}(r))$, then composing the two endpoint witnesses to recover the stored raw-visible link. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy raw visible branch exactness`

Theorem 35.94 (Sum ledger two-step branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$

and $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ *hold, then the whole two-step chain has a visible same-branch payload readback:*

$$(\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l, l')) \vee (\exists r\, r' : \mathsf{BHist}.\ Right(r) \wedge Right(r')$$

The ten-field fit is: traditional target disjoint-sum ledger-chain readback; BEDC source BHist *with* hsame*; carrier* SumHistoryCarrier*; classifier unused in the final exactness statement but supplied by Theorem 35.92; ledger two composable* SumHistoryLedgerPolicy *witnesses; core theorem this branch exactness statement; exactness raw-to-visible same-branch payload readback; standard bridge tagged sum ledger chains; status* ProofTarget.

Proof. Apply Theorem 35.93 to each ledger witness. The first application exposes either a left payload visible at ρ and v, or a right payload visible at ρ and v; the second exposes the corresponding alternatives at v and w. Split the four branch cases. In the left-left case, use Theorem 35.79 at the shared middle endpoint v to identify the two left payloads and assemble the left disjunct. The right-right case is identical with the right branch. The mixed left-right and right-left cases contradict generated-history no-confusion between E0 and E1 at the shared middle endpoint v. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step branch exactness`

Theorem 35.95 (Sum ledger two-step endpoint exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, w, l, l', r, r' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *and* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, v, w)$ *hold, then the two-ledger chain has endpoint exactness from* ρ *to* w*. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(w, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(w, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints of w *read back source payload evidence:*

$$(\mathsf{hsame}(w, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(w, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

Proof. Apply Theorem 35.92 to the two ledger witnesses and obtain $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame},$ The two mixed endpoint conclusions follow from Theorem 35.41; for the reversed displayed order, use classifier symmetry from Theorem 35.36 before applying the same tag-separation theorem. The left same-tag conclusion is exactly the left endpoint inversion supplied by Theorem 35.13, and the right same-tag conclusion is the corresponding right inversion from Theorem 35.14. For the final visible readback, apply Theorem 35.88 to the second ledger witness, obtaining a visible carrier for w; then apply Theorem 35.7 to the displayed E0 or E1 endpoint of w to extract the matching source payload evidence. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy two step endpoint exactness`

Definition 35.96 (Sum ledger finite provenance chain)**.** *For source carriers* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *as the finite provenance relation generated by the following two clauses. A displayed one-ledger witness* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, z)$ *gives* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$*. If there is a history* $v : \mathsf{BHist}$ *such that* $\mathsf{SumHistoryLedgerPolicy}(Left, R$ *and* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(v, z)$ *hold, then* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *also holds.*

Lean4 (definition) `BEDC.Derived.SumUp.SumHistoryLedgerChain`

Theorem 35.97 (Sum ledger chain transitivity)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, m, z : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, m)$ *and* $\mathsf{SumHistoryLedgerChain}_{Left,Righ}$ *hold, then* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds. The ten-field fit is: traditional target finite tagged-sum ledger provenance composition; BEDC source* BHist *with tagged sum carrier; carrier* $\mathsf{SumHistoryCarrier}$*; classifier supplied downstream by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem finite-chain composition; exactness concatenated raw-to-visible provenance; standard bridge tagged sum ledger audit composition; status* $\mathsf{ProofTarget}$*.*

Proof. Induct on the first displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, m)$ witness.

In the one-ledger case, the displayed $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, m)$ is the first finite ledger step. Rebuild the target chain by the cons constructor, using that ledger step as the head and the second displayed chain from m to z as the tail.

In the cons case, the first chain consists of a head ledger from ρ to an intermediate history and a tail chain from that intermediate history to m. Apply the induction hypothesis to compose the tail chain with the second displayed chain from m to z, then rebuild the full chain by the cons constructor with the original head ledger. □

Theorem 35.98 (Sum ledger chain classifier composition)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $raw, visible : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(raw, visible)$ *holds, then*

$$\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, raw, visible)$$

holds. The ten-field fit is: traditional target disjoint-sum finite ledger classifier closure; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* $\mathsf{SumHistoryCarrier}$*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; stability by raw-visible classification plus classifier transitivity; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this finite-chain classifier closure; exactness finite raw-to-visible classifier reachability; standard bridge tagged sum finite ledger provenance; status* $\mathsf{ProofTarget}$*.*

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(raw, visible)$ witness. In the one-ledger case, apply Theorem 35.89. In the cons case, apply the induction hypothesis to the tail chain to obtain the classifier from the middle history to *visible*, then compose it with the first ledger by Theorem 35.91. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain classifier composition`

Theorem 35.99 (Sum ledger chain visible carrier)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then*

$$\mathsf{SumHistoryCarrier}(Left, Right, z)$$

holds. The ten-field fit is: traditional target disjoint-sum finite ledger visible-carrier closure; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier not needed for this unary closure; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this carrier closure; exactness final visible endpoint remains in the sum carrier; standard bridge tagged sum finite ledger provenance; status* ProofTarget.

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ witness. In the one-ledger case, apply Theorem 35.88. In the cons case, use the carrier supplied by the tail induction hypothesis. □

Theorem 35.100 (Sum ledger chain endpoint exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z, l, l', r, r' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then the whole finite ledger provenance from* ρ *to* z *has endpoint exactness. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(z, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints of z *read back matching source payload evidence:*

$$(\mathsf{hsame}(z, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(z, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

The ten-field fit is: traditional target disjoint-sum finite ledger provenance; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}$ *supplied one ledger at a time by raw-visible classification; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$ *generated by* SumHistoryLedgerPolicy*; core theorem this chain endpoint exactness package; exactness mixed-tag separation, same-tag payload determinacy, and final visible source readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget.

Proof. First apply Theorem 35.98 to the displayed chain, obtaining $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame},$ The two mixed endpoint clauses follow from Theorem 35.41; for the reversed displayed order, first use classifier symmetry from Theorem 35.36. The same-tag endpoint clauses follow by transporting the classifier along the displayed endpoint hsame links and applying Theorem 35.13 or Theorem 35.14. For the final visible readback, apply Theorem 35.99 to the displayed chain to obtain $\mathsf{SumHistoryCarrier}(Left, Right, z)$. Transport this carrier along the displayed hsame endpoint at z and apply the left or right carrier inversion theorem to extract the matching source payload evidence. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain endpoint exactness`

Theorem 35.101 (Sum ledger chain branch readback soundness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z : \mathsf{BHist}$ *be histories. The finite ledger-chain relation forces exact same-branch payload readback:*

$$\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z) \to (\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch readback; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier supplied by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this branch readback soundness; finite ledger-chain witnesses map into same-branch payload readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Induct on the displayed $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ witness. In the one-ledger case, apply Theorem 35.93 and choose the same payload at the raw and visible endpoints, with hsame reflexivity for the payload link.

In the cons case, combine the raw-visible branch exactness of the first ledger with the induction hypothesis for the tail chain. The mixed left-right and right-left cases contradict generated-history no-confusion between E0 and E1 at the shared middle endpoint. In the same-branch cases, use Theorem 35.79 at the middle endpoint and hsame transitivity to compose the first payload with the final payload. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain branch exactness`

Theorem 35.102 (Sum ledger chain branch exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z : \mathsf{BHist}$ *be histories. The finite ledger-chain relation is equivalent to exact same-branch payload readback:*

$$\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z) \iff (\exists l\, l' : \mathsf{BHist}.\ Left(l) \wedge Left(l') \wedge \mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch readback; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier supplied by Theorem 35.98; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this branch exactness; exactness coverage and completeness of the finite ledger-chain relation by same-branch payload readback; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. For the forward direction, apply Theorem 35.101.

For the reverse direction, split the displayed disjunction. In the left branch, the witnesses $Left(l)$, $\mathsf{hsame}(\rho, \mathsf{E0}(l))$, $\mathsf{hsame}(z, \mathsf{E0}(l'))$, and $\mathsf{hsame}(l, l')$ determine a one-ledger left readback. Use symmetry of $\mathsf{hsame}(l, l')$ and constructor congruence for E0 to transport $\mathsf{hsame}(z, \mathsf{E0}(l'))$ into $\mathsf{hsame}(z, \mathsf{E0}(l))$.

Apply the reverse direction of Theorem 35.93 to obtain $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, z)$, then use the one-ledger constructor of $\mathsf{SumHistoryLedgerChain}_{Left,Right}$.

The right branch is identical, using the witnesses $Right(r)$, $\mathsf{hsame}(\rho, \mathsf{E1}(r))$, $\mathsf{hsame}(z, \mathsf{E1}(r'))$, and $\mathsf{hsame}(r, r')$, with constructor congruence for E1 before applying the same raw-visible branch exactness theorem and the one-ledger chain constructor. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain branch exactness iff`

Theorem 35.103 (Sum ledger chain raw branch forces final branch readback)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, z, l, r : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerChain}_{Left,Right}(\rho, z)$ *holds, then raw visible branch evidence forces the matching final visible branch. The left clause is*

$$Left(l) \to \mathsf{hsame}(\rho, \mathsf{E0}(l)) \to \exists l' : \mathsf{BHist}.\ Left(l') \wedge \mathsf{hsame}(z, \mathsf{E0}(l')) \wedge \mathsf{hsame}(l, l') \wedge \neg(\exists r' : \mathsf{BHist}.\ Right(r') \wedge \mathsf{hsa}$$

and the right clause is

$$Right(r) \to \mathsf{hsame}(\rho, \mathsf{E1}(r)) \to \exists r' : \mathsf{BHist}.\ Right(r') \wedge \mathsf{hsame}(z, \mathsf{E1}(r')) \wedge \mathsf{hsame}(r, r') \wedge \neg(\exists l' : \mathsf{BHist}.\ Left(l') \wedge \mathsf{h}$$

The ten-field fit is: traditional target disjoint-sum finite ledger branch forcing; BEDC source BHist *with* hsame*; carrier final* SumHistoryCarrier*; classifier supplied by the chain classifier composition theorem; ledger* $\mathsf{SumHistoryLedgerChain}_{Left,Right}$*; core theorem this raw-branch forcing readback; exactness same-tag payload determinacy plus mixed-tag exclusion; standard bridge tagged sum finite ledger provenance; status* ProofTarget*.*

Proof. Apply Theorem 35.99 to the chain and then Theorem 35.80 to the final carrier. This splits the endpoint z into a left visible presentation with right-presentation exclusion, or a right visible presentation with left-presentation exclusion.

For the left raw premise, keep the left final presentation from the split. The same-tag payload clause of Theorem 35.100, applied to $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and the displayed $\mathsf{hsame}(z, \mathsf{E0}(l'))$, gives $\mathsf{hsame}(l, l')$; combine it with the source witness for l' and the carried right-presentation exclusion.

The right final presentation cannot occur under the left raw premise, because the mixed endpoint clause of Theorem 35.100 contradicts $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(z, \mathsf{E1}(r'))$. The right raw premise is symmetric: the right final presentation supplies the witness and the left-presentation exclusion, while the mixed endpoint clause excludes any left final presentation. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerChain raw branch final readback`

Theorem 35.104 (Sum ledger preserves visible tag separation)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, l, r : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then both visible mixed-tag endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l))) \to \bot.$$

Proof. Apply Theorem 35.89 to the ledger policy to obtain $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$. The left-to-right mixed endpoint pair is then discharged by SumHistoryClassifier_mixed_tags_absurd with the displayed $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(v, \mathsf{E1}(r))$ witnesses. For the symmetric right-to-left pair, use classifier symmetry from Theorem 35.36 and apply the same mixed-tag separation theorem to the exchanged endpoints. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible tag separation`

Theorem 35.105 (Sum ledger same-tag payload exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, l, l', r, r' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then the visible same-tag endpoint pairs determine their payloads:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Proof. Apply Theorem 35.89 to obtain the raw-visible $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame}, \rho, v)$ witness. For the left-tagged endpoint pair, transport this witness along the displayed $\mathsf{hsame}(\rho, \mathsf{E0}(l))$ and $\mathsf{hsame}(v, \mathsf{E0}(l'))$ links by Theorem 35.37, then apply the left inversion Theorem 35.13 to extract $\mathsf{hsame}(l, l')$. The right-tagged case is identical with Theorem 35.14. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy same tag payload exactness`

Theorem 35.106 (Sum ledger visible payload readback)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, l, r : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then visible tagged endpoints read back source payload evidence:*

$$(\mathsf{hsame}(v, \mathsf{E0}(l)) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l, a))$$

and

$$(\mathsf{hsame}(v, \mathsf{E1}(r)) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r, b)).$$

Proof. The traditional target is disjoint-sum ledger readback; the BEDC source is BHist; the carrier is SumHistoryCarrier; no classifier is needed for this carrier-readback obligation; the ledger is SumHistoryLedgerPolicy; the exactness principle is visible-tag carrier readback; the status is ProofTarget. Apply Theorem 35.88 to the ledger, transport the visible carrier witness along the displayed $\mathsf{hsame}(v, \mathsf{E0}(l))$ or $\mathsf{hsame}(v, \mathsf{E1}(r))$ endpoint equality, and then apply Theorem 35.7 to extract the left or right source payload witness. □

Lean4 ✓ `BEDC.Derived.SumUp.SumHistoryLedgerPolicy visible payload readback`

Theorem 35.107 (Sum ledger one-step endpoint exactness)**.** *Let* $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ *be source carriers and let* $\rho, v, l, l', r, r' : \mathsf{BHist}$ *be histories. If* $\mathsf{SumHistoryLedgerPolicy}(Left, Right, \rho, v)$ *holds, then the one-ledger link has endpoint exactness from* ρ *to* v*. First, both mixed endpoint pairs are impossible:*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \bot$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \bot.$$

Second, same-tag endpoint pairs determine their payloads:

$$(\mathsf{hsame}(\rho, \mathsf{E0}(l)) \wedge \mathsf{hsame}(v, \mathsf{E0}(l'))) \to \mathsf{hsame}(l, l')$$

and

$$(\mathsf{hsame}(\rho, \mathsf{E1}(r)) \wedge \mathsf{hsame}(v, \mathsf{E1}(r'))) \to \mathsf{hsame}(r, r').$$

Third, final visible endpoints read back source payload evidence:

$$(\mathsf{hsame}(v, \mathsf{E0}(l')) \to \exists a : \mathsf{BHist}.\ Left(a) \wedge \mathsf{hsame}(l', a))$$

and

$$(\mathsf{hsame}(v, \mathsf{E1}(r')) \to \exists b : \mathsf{BHist}.\ Right(b) \wedge \mathsf{hsame}(r', b)).$$

The roadmap ten-field fit is: traditional target disjoint-sum one-ledger endpoint audit; BEDC source BHist *with* hsame*; source carriers* $Left$ *and* $Right$*; carrier* SumHistoryCarrier*; classifier* $\mathsf{SumHistoryClassifier}(Left, Right, \mathsf{hsame}, \mathsf{hsame})$*; ledger* SumHistoryLedgerPolicy*; core theorem this endpoint exactness package; exactness raw-to-visible tag separation, same-tag payload determinacy, and visible source readback; standard bridge tagged sum ledger readback; status* ProofTarget*.*

Proof. Apply Theorem 35.104 to the ledger witness for the two mixed clauses. Apply Theorem 35.105 to the same ledger witness for the two same-tag payload clauses. Apply Theorem 35.106 to the ledger witness for the two final visible readback clauses. □

Theorem 35.108 (Concrete $\mathsf{Sum}^{\uparrow}$ semantic name certificate)**.** *Let $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ be source carriers with classifiers $LeftEq$ and $RightEq$. If $\mathsf{NameCert}(Left, LeftEq)$ and $\mathsf{NameCert}(Right, RightEq)$ are supplied, then the tagged sum carrier, the tagged sum classifier, and the carrier-as-pattern and carrier-as-ledger tuple form*

$$\begin{aligned}&\mathsf{SemanticNameCert}(\\ &\quad \mathsf{SumHistoryCarrier}(Left, Right),\\ &\quad \mathsf{SumHistoryCarrier}(Left, Right),\\ &\quad \mathsf{SumHistoryCarrier}(Left, Right),\\ &\quad \mathsf{SumHistoryClassifier}(Left, Right, LeftEq, RightEq)).\end{aligned}$$

Proof. The carrier is represented by $e0$-tagged left histories and $e1$-tagged right histories. Theorem 35.11 marks the split: the classifier delegates within matching tags to the corresponding source certificate, while source-carrier evidence is taken from $\mathsf{SumHistoryCarrier}$ when certificate fields require it. Reflexivity is built by tagging the source certificate reflexivity field on the selected component. Symmetry and transitivity invert the tagged classifier witnesses, use the matching source certificate fields inside a left–left or right–right branch, and discharge mixed tags by constructor no-confusion.

The core certificate fields are assembled from Theorem 35.44. That theorem supplies carrier-relative reflexivity, source-certified symmetry, source-certified transitivity, and classifier-induced carrier transport for the tagged carrier and tagged classifier. The inhabited field is separate: take the inhabited witness l from the left source certificate, choose the visible endpoint $\mathsf{E0}(l)$, and rebuild $\mathsf{SumHistoryCarrier}(Left, Right, \mathsf{E0}(l))$ by the left tagged constructor with the reflexive endpoint witness and the stored $Left(l)$ evidence.

Inside the carrier-transport field supplied by the cited theorem, source-carrier membership is not transported along an arbitrary source $\mathsf{NameCert}$ merely from $\mathsf{hsame}(l, a)$. In the left branch, constructor inversion first aligns the tagged payloads and obtains $\mathsf{hsame}(l, a)$; the checked finite-kernel identity $\mathsf{BEDC.FKernel.Hist.hsame_iff_eq}$ converts that history-sameness witness to kernel equality before the stored $Left(l)$ evidence is rewritten as $Left(a)$. Only after this equality rewrite does $\mathsf{NameCert}.carrier_respects_equiv$ from the left source certificate apply along the displayed source classifier witness $LeftEq(a, b)$, yielding $Left(b)$ and hence the target left tagged carrier. The right branch is the same with $RightEq(a, b)$ and the right source certificate. Pattern soundness and ledger soundness are identity implications, since both semantic fields are the same predicate $\mathsf{SumHistoryCarrier}(Left, Right)$. □

Lean4 ✓ `BEDC.Derived.SumUp.sum history semantic name certificate`

Corollary 35.109 ($\mathsf{Sum}^{\uparrow}$ is licensed, not primitive)**.** *$\mathsf{Sum}^{\uparrow}$ is licensed as a derived disjoint sum, not as a primitive coproduct.*

Theorem 35.110 (Sum public tagged-payload standard bridge)**.** *For source carriers $Left, Right : \mathsf{BHist} \to \mathsf{Prop}$ equipped with source classifiers and source $\mathsf{NameCert}$ rows, the public $\mathsf{Sum}^{\uparrow}$ semantic certificate presents exactly the standard finite disjoint-sum interface with two visible injections. A public endpoint is either an $\mathsf{E0}$-tagged left payload or an $\mathsf{E1}$-tagged right payload. Endpoints with the same tag compare by the corresponding source classifier, and endpoints with mixed tags are separated by constructor no-confusion.*

The exported ledger surface is precisely the visible tag, payload readback, raw-visible branch exactness, and endpoint exactness already recorded by the $\mathsf{Sum}^{\uparrow}$ rows. No consumer receives an additional coproduct eliminator, hidden branch coordinate, host disjoint-union type, or classifier other than the branchwise tagged classifier.

Lean4 ✓ `BEDC.Derived.SumUp.SumUp concrete to schema`

Proof. The public certificate is the one supplied by Definition 35.108. Its carrier consists of histories built by the two visible constructors, so constructor inversion gives the two alternatives $\mathsf{E0}(l)$ and $\mathsf{E1}(r)$ and gives no third endpoint form.

Within a left–left comparison, the tagged classifier in Definition 35.108 projects to the left source classifier; within a right–right comparison, it projects to the right source classifier. The branch exactness statement in Definition 35.93 identifies those projections with the raw visible branch rows exported by the ledger.

For mixed tags, Definition 35.104 applies to the visible ledger row and reduces equality or classifier compatibility of opposite tags to constructor no-confusion. Thus a left endpoint and a right endpoint remain separated in the public reading.

Finally, Definition 35.106 recovers the displayed source payload from each visible endpoint, and Definition 35.100 states that the endpoint readback is exact along the public chain. These rows are exactly the finite two-injection interface described above, so the tagged-payload certificate is a paper bridge to the standard disjoint-sum reading. □

Theorem 35.111 (Sum mature finite consumer boundary)**.** *Every finite consumer of the public* $\mathsf{Sum}^{\uparrow}$ *package factors through the tagged payload carrier, visible tag separation, branch exactness, payload readback, endpoint chain exactness, and the public tagged-payload standard bridge. Such a consumer can inspect only whether the exposed endpoint is an* $\mathsf{E0}$*-tagged left payload or an* $\mathsf{E1}$*-tagged right payload, can read the corresponding source evidence through the ledger rows, and can compare same-tag payloads only through the branch classifier exactness already exported by the certificate.*

This finite boundary excludes eliminators into arbitrary certified targets, dependent sums, and categorical coproduct universal properties. Those structures are not consequences of the tagged-payload package surface; they require additional consumer principles beyond the public $\mathsf{Sum}^{\uparrow}$ *certificate.*

Proof. Start with the public tagged-payload bridge in Theorem 35.110. It identifies the public surface with the two visible injections and rules out any hidden carrier coordinate outside the tagged history endpoint.

The finite consumer's branch observation is then forced by Theorem 35.104; mixed visible tags cannot be read as the same public branch. Same-branch comparisons are governed by Theorem 35.93, which is the raw-visible branch exactness row for the ledger.

Payload access is supplied by Theorem 35.106. Chain-level endpoint soundness is supplied by Theorem 35.100, so repeated finite ledger reads still factor through the same tagged endpoint and its source payload evidence.

Combining these rows leaves exactly the finite tagged-payload consumer surface. Since none of the cited rows constructs an arbitrary-target eliminator, dependent sum, or coproduct universal arrow, those stronger interfaces remain outside the mature finite boundary stated here. □

Theorem 35.112 (Sum finite-data scoped-exit package)**.** *The scoped finite-data exit surface of* $\mathsf{Sum}^{\uparrow}$ *is exactly the package consisting of the tagged semantic certificate of Definition 35.108, the selected visible injection tag, left and right branch exactness, the source ledger rows, visible-history descent through endpoint readback, and the mature finite consumer boundary of Definition 35.111. The selected injection tag is the two-branch finite-choice row already licensed by chapter 32;* $\mathsf{Sum}^{\uparrow}$ *consumes that sibling only as visible tag discipline, while the left and right payload rows remain governed by the local sum carrier and classifier. A consumer exiting the* $\mathsf{Sum}^{\uparrow}$ *chapter through this*

surface may inspect the selected tag, recover the corresponding source payload and source ledger evidence, compare same-tag endpoints through the displayed branch classifier, and replay the visible history descent. It receives no hidden coproduct eliminator, no dependent target recursion, and no classifier beyond the tagged finite-data rows already exported by the semantic certificate.

Proof. Start with the semantic certificate in Definition 35.108. Its carrier is the tagged finite packet, so each public endpoint enters through an $\mathsf{E0}$ or $\mathsf{E1}$ injection rather than through a host coproduct object.

The selected tag is governed by Definition 35.104. The $\mathsf{Bool}^{\uparrow}$ reference supplies only the finite $b0/b1$ tag reading for this visible branch choice; it does not introduce a host coproduct, an eliminator, or an external Boolean equality. The same-tag clauses are governed by Definition 35.93, while visible payload recovery is governed by Definition 35.106 and endpoint descent by Definition 35.100.

Finally apply the consumer boundary in Definition 35.111. It states that every finite consumer factors through exactly these tagged payload, branch exactness, source ledger, and visible-history rows. Therefore the package is a scoped exit surface for finite data rather than a generic coproduct interface. □

Theorem 35.113 (Sum finite-data scoped-exit package)**.** *The scoped finite-data exit surface of* $\mathsf{Sum}^{\uparrow}$ *is exactly the package consisting of the tagged semantic certificate of Definition 35.108, the selected visible injection tag, left and right branch exactness, the source ledger rows, visible-history descent through endpoint readback, and the mature finite consumer boundary of Definition 35.111. The selected injection tag is the two-branch finite-choice row already licensed by chapter 32;* $\mathsf{Sum}^{\uparrow}$ *consumes that sibling only as visible tag discipline, while the left and right payload rows remain governed by the local sum carrier and classifier. A consumer exiting the* $\mathsf{Sum}^{\uparrow}$ *chapter through this surface may inspect the selected tag, recover the corresponding source payload and source ledger evidence, compare same-tag endpoints through the displayed branch classifier, and replay the visible history descent. It receives no hidden coproduct eliminator, no dependent target recursion, and no classifier beyond the tagged finite-data rows already exported by the semantic certificate.*

Proof. Start with the semantic certificate in Definition 35.108. Its carrier is the tagged finite packet, so each public endpoint enters through an $\mathsf{E0}$ or $\mathsf{E1}$ injection rather than through a host coproduct object.

The selected tag is governed by Definition 35.104. The $\mathsf{Bool}^{\uparrow}$ reference supplies only the finite $b0/b1$ tag reading for this visible branch choice; it does not introduce a host coproduct, an eliminator, or an external Boolean equality. The same-tag clauses are governed by Definition 35.93, while visible payload recovery is governed by Definition 35.106 and endpoint descent by Definition 35.100.

Finally apply the consumer boundary in Definition 35.111. It states that every finite consumer factors through exactly these tagged payload, branch exactness, source ledger, and visible-history rows. Therefore the package is a scoped exit surface for finite data rather than a generic coproduct interface. □

Theorem 35.114 (Sum finite-data sibling route)**.** *The scoped finite-data exit surface of* $\mathsf{Sum}^{\uparrow}$ *reads its visible branch tag through the same* $b0/b1$ *discipline licensed by chapter 32. Once that visible tag has selected the left or right branch, the selected payload is carried only through the* $\mathsf{Sum}^{\uparrow}$ *semantic certificate of Definition 35.108 and through the existing branch exactness rows, including the single-valued visible readback of Definition 35.80. Thus the sibling route identifies the finite tag surface consumed by* $\mathsf{Sum}^{\uparrow}$ *without adding a host coproduct eliminator, a hidden branch coordinate, or any classifier beyond the displayed tagged finite-data rows.*

Proof. The semantic certificate of Definition 35.108 supplies the carrier and classifier for the tagged sum packet. Its two public entry forms are the displayed injections $\mathsf{E0}(l)$ and $\mathsf{E1}(r)$, so the branch decision is already a visible finite tag rather than an ambient coproduct object.

The tag discipline is the same two-row $b0/b1$ surface recorded by chapter 32. In the $\mathsf{Sum}^{\uparrow}$ chapter this discipline is consumed only as the selected visible branch. The payload itself remains governed by the left or right source classifier inside the local sum certificate.

Apply the visible readback theorem Definition 35.80. It says that the public history carrier has a single visible branch presentation, so the tag that selects the branch also selects the payload row used downstream. The branch exactness rows then compare only same-tag endpoints through the displayed branch classifier.

Therefore the route from finite tag to payload is a sibling finite-data route: it reads the branch through the BoolUp tag surface and then stays inside the SumUp semantic certificate and ledger rows. No hidden branch coordinate, host eliminator, or additional classifier is supplied. □

Theorem 35.115 (Sum finite-data Option route)**.** *The optional value surface of chapter 33 is the one-sided specialization of the same visible $b0/b1$ tag discipline used by $\mathsf{Sum}^{\uparrow}$. $\mathsf{Option}^{\uparrow}$ contributes the none/some finite tag route, while $\mathsf{Sum}^{\uparrow}$ contributes the two-payload branch classifier through Definition 35.114; neither route exports a host coproduct eliminator or hidden branch coordinate.*

Proof. Project the $\mathsf{Sum}^{\uparrow}$ semantic certificate of Definition 35.108. Its visible branch surface is the pair of displayed tags $\mathsf{E0}(l)$ and $\mathsf{E1}(r)$, and the payload is read only after the tag selects the left or right source classifier.

Compare this with the $\mathsf{Option}^{\uparrow}$ surface of chapter 33. The none/some distinction is a one-sided finite tag route: the none case carries no payload, while the some case carries the single payload behind its visible tag.

Use Definition 35.114. The existing BoolUp $b0/b1$ discipline identifies the shared visible tag layer, so the Option route is a sibling finite-data route rather than a new classifier or hidden branch coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Sum}^{\uparrow}}$**.**

Constructive story: A sum value is made by choosing one side and marking that choice directly on the payload: a left payload l from L becomes $\mathsf{E0}(l)$, while a right payload r from R becomes $\mathsf{E1}(r)$. Those two tags are part of the history itself, so the classifier first checks that the tags land on the same branch and only then compares the payloads by the left or right source classifier; a b0-tagged left value and a b1-tagged right value never meet. The same tagged-history predicate is the source, the pattern records the left-versus-right split, the two tag forms, and the no-confusion rows, and the ledger keeps each visible sum history aligned with the raw tagged payload that produced it. Because history-equality and the two source-side classifiers are equivalences, the injected witnesses supply the carrier, the branchwise classifier is reflexive, symmetric, and transitive, and classifier-recognized representatives stay inside the same tagged carrier. A history h lies in the signature gap of a $\mathsf{Sum}^{\uparrow}$ token p when the source-domain accepts the tagged history, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; after the tags are inspected, h is either a left injection whose payload matches the left side encoded by p, or a right injection whose payload matches the right side encoded by p. The chapter registers this tagged-payload construction under the name $\mathsf{Sum}^{\uparrow}$; disjoint union is the same b0/b1 choice used at the kernel level, now carrying a payload, with no separate coproduct type, eliminator, or host disjoint-union construct introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Sum}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public paper scope binds the tagged sum semantic certificate, selected injection tag, left and right branch classifier exactness, source ledger, visible-history descent, licensed public surface, mature finite consumer boundary, scoped finite-data exit package, and finite-data sibling route in Definition 35.114.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Sum}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SumUp.sum\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full sum or coproduct theory: eliminators into arbitrary certified targets, dependent sums, categorical coproduct universal property, distributivity laws, or generic case recursion.

Upgrade path: No further upgrade: chapter is at the top of the formal bridge axis.

Theorem 35.116 (Sum finite-data sibling dependency route)**.** *Every scoped finite-data consumer of* $\mathsf{Sum}^{\uparrow}$ *first reads the visible branch tag through the* $b0/b1$ *branch-mark surface of chapter 32, and only then reads the selected payload from the local* $\mathsf{Sum}^{\uparrow}$ *branch ledger. Its payload packaging is compared with the paired-carrier route of chapter 34:* $\mathsf{Prod}^{\uparrow}$ *keeps both coordinates simultaneously, while* $\mathsf{Sum}^{\uparrow}$ *exposes exactly one branch after the Boolean tag has selected it.*

Proof. Project the accepted $\mathsf{Sum}^{\uparrow}$ certificate. The carrier visible at Definition 35.108 contains a tag row and one payload row, so the consumer cannot inspect a left or right payload before the tag has been read.

The tag is the same finite branch surface used by $\mathsf{Bool}^{\uparrow}$. It contributes only the two visible marks $b0$ and $b1$; it does not add a host eliminator, ambient Boolean equality, or hidden branch coordinate.

Compare this with $\mathsf{Prod}^{\uparrow}$. The product route keeps both payload coordinates under one paired carrier, while the sum route keeps one payload behind a visible tag. Thus the sibling dependency is finite and explicit: $\mathsf{Bool}^{\uparrow}$ supplies the branch mark, $\mathsf{Prod}^{\uparrow}$ supplies the paired-carrier comparison, and $\mathsf{Sum}^{\uparrow}$ retains its own tagged payload certificate. □

Theorem 35.117 (Sum public branch-disjointness route)**.** *Every public* $\mathsf{Sum}^{\uparrow}$ *branch read factors through the visible tag exactness, branch exclusivity, left and right tag inversion, same-tag exactness, and the* $\mathsf{Bool}^{\uparrow}$ *finite discriminator route. A left-branch read exposes only a displayed* $\mathsf{E0}$ *tag together with its left payload classifier row; a right-branch read exposes only a displayed* $\mathsf{E1}$ *tag together with its right payload classifier row. No public consumer can read a left payload as a right payload or read a right payload as a left payload.*

Proof. Start with carrier branch exclusivity from Definition 35.3. It prevents a single displayed sum history from carrying both a left presentation and a right presentation, so a public branch read must choose one visible tag surface.

For same-tag reads, apply the left and right tag inversion theorems Definition 35.13 and Definition 35.14, together with Definition 35.15. These identify the payload classifier only inside the branch selected by the visible tag.

For mixed-tag reads, use Definition 35.41 and Definition 35.42. The finite discriminator supplied by $\mathsf{Bool}^{\uparrow}$ in Definition 32.63 and Definition 32.83 is therefore only a displayed left/right endpoint row; it cannot export the opposite payload carrier or classifier. Thus the public route is branch-disjoint. □

Chapter 36

A Concrete Naming Certificate for $\mathsf{List}^{\uparrow}$

The list interface $\mathsf{List}^{\uparrow}$ is licensed over a source carrier endowed with a $\mathsf{NameCert}$ and a spine-representation coherence witness. The carrier is the inductive sequence built from a nil constructor and a cons constructor consuming one source witness and one shorter list. The classifier is structural sameness lifted from the source classifier through finite spines: nil-with-nil and cons-with-cons under componentwise sameness. It is not a primitive sequence type; it is licensed only as a derived inductive over the source under this coherence datum.

The $\mathsf{List}^{\uparrow}$ spine consumes the constructor-tag and product-envelope surfaces of chapter 32, chapter 33, chapter 35, and chapter 34 before exporting the finite list carrier.

36.1 Roadmap template status

1. **Traditional target**: finite lists over a source object.
2. **BEDC source**: one licensed source carrier A with its name certificate.
3. **Carrier choice**: the carrier is generated by nil and $\mathsf{cons}(a, xs)$.
4. **Classifier choice**: $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}$ is the paper-side generated relation on BEDC histories, relating nil endpoints to nil endpoints and cons endpoints to cons endpoints by source head classification, recursive tail classification, and continuation witnesses. The certificate classifier is the spine-representation classifier from Definition 36.57, linked to the generated relation by Theorem 36.61 and used for the semantic certificate under $\mathsf{ListSpineRepCoherent}_{S,\sim_S}$; the finite-spine helper $\mathsf{ListClassifierSpec}$ remains the checked structural model.
5. **Stability obligations**: structural symmetry from the source $\mathsf{NameCert}$, structural transitivity, list-carrier transport, nil/cons separation, map stability, append stability, reverse stability, and cancellation facts.
6. **Ledger**: list length and constructor shape remain visible; payload classification is delegated to the source certificate.
7. **Core theorem list**: history carrier case analysis, history classifier constructor exactness, list spine representation coverage, classifier transitivity, classifier symmetry from the source $\mathsf{NameCert}$, cons inversion, singleton equivalence, map stability, append stability, source-parametric

left, right, and context cancellation, reverse stability, checked hsame length preservation, represented-spine classifier alignment of Theorem 36.144, and the paper-side source-parametric length preservation of Theorem 36.31.

8. **Exactness predicate**: classification holds exactly when there exist finite spines xs, ys such that $\mathsf{ListSpineRep}_A(h, xs)$, $\mathsf{ListSpineRep}_A(k, ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ hold; $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ makes alternate presentations of the same endpoint compatible for certificate use.

9. **Standard bridge**: the generated carrier reads back through $\mathsf{ListSpineRep}_A$ under $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$; ordinary finite list syntax is the finite-spine presentation witness, not exported carrier equality or a quotient.

10. **Status**: three certificate scopes are kept separate. Theorem 36.68 is the Phase Finite Data unframed public-object ProofTarget for $\mathsf{List}^{\uparrow}$ under $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Theorem 36.163 is the cons-boundary route to that same public ProofTarget when $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ is supplied. Corollary 36.128 is a separate framed carrier certificate under source-carrier hsame compatibility and does not close the unframed public $\mathsf{List}^{\uparrow}$ scope. The finite-spine classifier field inventory is recorded in Definition 36.32, while Theorem 36.31 records only source-parametric length preservation.

36.2 Carrier definition

Definition 36.1 (List finite-spine carrier)**.** *For a source type A,* $\mathsf{ListCarrier}(A)$ *denotes the finite-spine helper with entries of type A;* ListClassifierSpec *uses this helper for structural nil/cons reasoning. The exported BEDC public history carrier is* $\mathsf{ListHistoryCarrier}_S$ *in Definition 36.3.*

36.3 The certificate

Definition 36.2 (List source specification)**.** *The source is finite generated sequencing over the source certificate for A.*

Definition 36.3 (List history carrier)**.** *For a source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$, define* $\mathsf{ListHistoryCarrier}_S(h)$ *as the least history predicate generated by two clauses:*

1. *nil: if* $\mathsf{hsame}(h, \mathsf{e})$*, then* $\mathsf{ListHistoryCarrier}_S(h)$*;*
2. *cons: if there are histories a, t, p with $S(a)$,* $\mathsf{ListHistoryCarrier}_S(t)$*,* $\mathsf{Cont}(a, t, p)$*, and* $\mathsf{hsame}(h, \mathsf{E1}(p))$*, then* $\mathsf{ListHistoryCarrier}_S(h)$*.*

Lean4 (definition) `BEDC.Derived.ListUp.ListHistoryCarrier`

Definition 36.4 (List recursive history classifier)**.** *For a source carrier $S : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_S : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$, define* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(h, k)$ *as the least relation generated by two clauses:*

1. *nil-nil: if* $\mathsf{hsame}(h, \mathsf{e})$ *and* $\mathsf{hsame}(k, \mathsf{e})$*, then* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(h, k)$*;*
2. *cons-cons: if there are histories a, b, t, u, p, q such that $S(a)$, $S(b)$, $a \sim_S b$,* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(t, u)$*,* $\mathsf{Cont}(a, t, p)$*,* $\mathsf{Cont}(b, u, q)$*,* $\mathsf{hsame}(h, \mathsf{E1}(p))$*, and* $\mathsf{hsame}(k, \mathsf{E1}(q))$*, then* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(h, k)$*.*

Lean4 (definition) `BEDC.Derived.ListUp.ListHistoryClassifierRec`
Lean4 (definition) `BEDC.Derived.ListUp.ListHistoryClassifier`

Theorem 36.5 (List recursive history classifier carrier endpoints)**.** *For any source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and source classifier* $\sim_S$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(h,k)$ *holds, then both* $\mathsf{ListHistoryCarrier}_S(h)$ *and* $\mathsf{ListHistoryCarrier}_S(k)$ *hold.*

Proof. Induct on the displayed classifier witness. In the nil-nil clause, the two endpoint witnesses rebuild the nil clause of $\mathsf{ListHistoryCarrier}_S$ on both sides. In the cons-cons clause, the induction hypothesis gives carrier witnesses for the two tails t and u; combine those with the source carrier witnesses $S(a)$ and $S(b)$, the continuation witnesses $\mathsf{Cont}(a,t,p)$ and $\mathsf{Cont}(b,u,q)$, and the endpoint witnesses $\mathsf{hsame}(h,\mathsf{E1}(p))$ and $\mathsf{hsame}(k,\mathsf{E1}(q))$ to rebuild the two cons carrier clauses. □

Lean4 ✓ `BEDC.Derived.ListUp.ListHistoryClassifierRec carrier endpoints`
Lean4 ✓ `BEDC.Derived.ListUp.ListHistoryClassifier carrier endpoints`

Theorem 36.6 (List recursive history classifier constructor exactness)**.** *For any source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_S$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*, and histories* h, k*, the recursive classifier is exactly the constructor disjunction*

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}(h,k) \iff (\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e})) \vee \exists a\, b\, t\, u\, p\, q.\ S(a) \wedge S(b) \wedge a \sim_S b \wedge \mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim}$$

Proof. For the forward implication, induct on the generated classifier witness. The nil-nil constructor gives the left disjunct, and the cons-cons constructor gives the right disjunct with its source head witnesses, recursive tail-classifier witness, continuation witnesses, and visible $e1$ endpoint witnesses. For the reverse implication, the left disjunct is exactly the nil-nil generation clause, and the right disjunct is exactly the cons-cons generation clause. Mixed nil/cons endpoint claims reduce to BEDC history no-confusion: `not_hsame_emp_e1` eliminates nil against cons, and `not_hsame_e1_empty` eliminates cons against nil. □

Theorem 36.7 (List history carrier generated cases)**.** *For any source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and any history* h*, if* $\mathsf{ListHistoryCarrier}_S(h)$*, then either* $\mathsf{hsame}(h,\mathsf{e})$*, or there exist histories* a, t, p *such that* $S(a)$*,* $\mathsf{ListHistoryCarrier}_S(t)$*,* $\mathsf{Cont}(a,t,p)$*, and* $\mathsf{hsame}(h,\mathsf{E1}(p))$*.*

Proof. Apply the induction principle for the least predicate displayed in the preceding definition. The nil constructor gives the left disjunct directly. The cons constructor exposes its witnesses a, t, p together with $S(a)$, the recursive carrier witness for t, the continuation witness $\mathsf{Cont}(a,t,p)$, and the endpoint witness $\mathsf{hsame}(h,\mathsf{E1}(p))$, which are exactly the data required for the right disjunct. □

Lean4 ✓ `BEDC.Derived.ListUp.ListHistoryCarrier generated cases`

Theorem 36.8 (List history carrier transport)**.** *For any source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, if* $\mathsf{hsame}(h,k)$ *and* $\mathsf{ListHistoryCarrier}_S(h)$*, then* $\mathsf{ListHistoryCarrier}_S(k)$*.*

Proof. Proceed by cases on the displayed generation of $\mathsf{ListHistoryCarrier}_S(h)$. In the nil case, compose the symmetry of $\mathsf{hsame}(h,k)$ with the endpoint witness $\mathsf{hsame}(h,\mathsf{e})$ to obtain $\mathsf{hsame}(k,\mathsf{e})$, then use the nil clause. In the cons case, keep the witnesses a, t, p, the source witness $S(a)$, the recursive carrier witness for t, and the continuation witness $\mathsf{Cont}(a,t,p)$. Compose the symmetry of $\mathsf{hsame}(h,k)$ with the endpoint witness $\mathsf{hsame}(h,\mathsf{E1}(p))$ to obtain $\mathsf{hsame}(k,\mathsf{E1}(p))$, then rebuild the cons clause for k. □

Theorem 36.9 (List recursive history classifier reflexivity, symmetry, and endpoint exposure from a source certificate)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. The generated classifier* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}$ *records the following recursive classifier fields here, rather than the entire certificate stability block:*

1. *if* $\mathsf{ListHistoryCarrier}_A(h)$ *holds, then* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,h)$ *holds;*
2. *if* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,k)$ *holds, then* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(k,h)$ *holds;*
3. *if* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,k)$ *holds, then both* $\mathsf{ListHistoryCarrier}_A(h)$ *and* $\mathsf{ListHistoryCarrier}_A(k)$ *hold.*

Lean4 ✓ `BEDC.Derived.ListUp.ListHistoryClassifierRec equivalence fields`

Proof. For reflexivity, induct on the displayed generation of $\mathsf{ListHistoryCarrier}_A(h)$. The nil clause is rebuilt from hsame reflexivity and the given endpoint link. In the cons clause, the source certificate supplies head reflexivity from the head carrier witness, the induction hypothesis supplies tail reflexivity, and the stored Cont witness and endpoint hsame link rebuild the cons-cons classifier clause. For endpoint membership, induct on the displayed generation of $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,k)$. The nil-nil clause rebuilds two nil carrier witnesses. The cons-cons clause combines the two source carrier witnesses, the induction hypotheses for the two tails, the two Cont witnesses, and the two endpoint hsame links to rebuild both cons carrier witnesses. For symmetry, induct on the classifier generation. The nil-nil case swaps the two endpoint hsame witnesses. The cons-cons case uses the symmetry field of cert on the head classifier witness, applies the induction hypothesis to the tail classifier witness, swaps the two Cont witnesses and endpoint links, and rebuilds the cons-cons clause. □

Transitivity for the certificate classifier is the separate spine-coherent result Theorem 36.63.

Definition 36.10 (List pattern specification)**.** *The pattern is nil-or-cons generation with structurally smaller tail.*

Definition 36.11 (List history pattern)**.** *For a source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*, the predicate* $\mathsf{ListHistoryPattern}_S(h)$ *holds when* h *has the visible BEDC shape of a list endpoint: either* $\mathsf{hsame}(h,\mathsf{e})$*, or there are histories* a,t,p *with* $S(a)$*,* $\mathsf{ListHistoryCarrier}_S(t)$*,* $\mathsf{Cont}(a,t,p)$*, and* $\mathsf{hsame}(h,\mathsf{E1}(p))$*.*

Definition 36.12 (List classifier specification)**.** *The finite-spine helper classifier relates nil with nil and cons with cons when heads are source-classified the same and tails are list-classified the same.*

Lean4 (definition) `BEDC.Derived.ListUp.ListClassifierSpec`

Theorem 36.13 (List classifier case exactness)**.** *For any source relation* sameA *on* A *and any finite spines* $xs, ys : \mathsf{ListCarrier}(A)$*,* $\mathsf{ListClassifierSpec}(\mathsf{sameA}, xs, ys)$ *holds exactly in the matching constructor cases: both spines are* nil*, or both spines are* cons *spines with a source-head witness* $\mathsf{sameA}(a,b)$ *and a recursive tail witness* $\mathsf{ListClassifierSpec}(\mathsf{sameA}, xs', ys')$*. It is false for the mixed shapes* nil *against* $\mathsf{cons}(b, ys')$ *and* $\mathsf{cons}(a, xs')$ *against* nil*.*

Proof. Case-analyze the two finite-spine constructors. In the nil-nil case the classifier is the nil clause. In the cons-cons case the classifier unfolds to its head source witness together with the recursive tail classifier witness, and conversely those two witnesses rebuild the cons classifier clause. The nil-cons and cons-nil cases have no classifier clause, so any assumed witness is eliminated by constructor case analysis. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec case exactness`

Lemma 36.14 (List classifier nil-cons no-confusion)**.** *For any source classifier on A,* ListClassifierSpec *never relates* nil *to* $\mathsf{cons}(a, xs)$*, and never relates* $\mathsf{cons}(a, xs)$ *to* nil*.*

Proof. Unfold the structural classifier by list constructors. The only nil clause is the nil-nil clause, and the only cons clause is the cons-cons clause. The two mixed constructor shapes therefore have no classifier constructor, so both directions are discharged directly by case analysis on the impossible witness. □

Theorem 36.15 (List classifier is transitive)**.** *For any source type A, relation $R : A \to A \to \mathsf{Prop}$, and finite spines $xs, ys, zs : \mathsf{ListCarrier}(A)$, if $\forall a\, b\, c.\ R(a, b) \to R(b, c) \to R(a, c)$,* $\mathsf{ListClassifierSpec}(R, xs, ys)$*, and* $\mathsf{ListClassifierSpec}(R, ys, zs)$ *hold, then* $\mathsf{ListClassifierSpec}(R, xs, zs)$ *holds.*

Proof. Induct on the first finite spine. Shape mismatches are impossible by the classifier clauses. In the cons-cons-cons case, the head witnesses compose by source transitivity and the tail witnesses compose by the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec trans`

Theorem 36.16 (History list classifier is transitive)**.** *For lists of finite-kernel histories, if the structural list classifier relates x to y and y to z using* hsame *on heads, then it relates x to z.*

Proof. Induct on the first list. The nil and cons shape mismatches are impossible by the classifier clauses. In the cons-cons-cons case, head witnesses compose by transitivity of hsame, and the tail witnesses compose by the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame trans`

Theorem 36.17 (History list classifier is symmetric)**.** *For lists of finite-kernel histories, if the structural list classifier relates x to y using* hsame *on heads, then it relates y to x.*

Proof. Induct on the first list. Shape mismatches are impossible by the classifier clauses. In the cons-cons case, head witnesses reverse by symmetry of hsame, and tail witnesses reverse by the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame symm`

Theorem 36.18 (History list classifier cons inversion)**.** *For BEDC histories x, y and history-list tails xs, ys, if* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x :: xs, y :: ys)$ *holds, then* $\mathsf{hsame}(x, y)$ *and* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys)$ *hold.*

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame cons inversion`

Proof. Unfold the cons-cons classifier clause and read off its source-head and recursive-tail components. □

Theorem 36.19 (History list classifier one-headed tail transport)**.** *For BEDC histories x, y and history-list tails xs, ys, zs, if* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, zs)$ *and* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{E1}(x) :: xs, \mathsf{E1}(y) :: ys)$ *hold, then*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{E1}(x) :: zs, \mathsf{E1}(y) :: ys)$$

holds.

Proof. Invert the one-headed cons classifier to read off the head comparison $\mathsf{hsame}(\mathsf{E1}(x), \mathsf{E1}(y))$ and the tail classifier from xs to ys. Reverse the classifier from xs to zs and compose it with the tail classifier by history-list transitivity. Reattach the unchanged head comparison by the cons clause. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame e1 cons tail transport`

Theorem 36.20 (Bool-sourced list classifier cons mark readback)**.** *For histories h, k : BHist and finite spines xs, ys, if* $\mathsf{ListClassifierSpec}(\mathsf{BoolHistoryClassifier}, h :: xs, k :: ys)$ *holds, then*

$$(\exists v, w : \mathsf{BMark}.\ \mathsf{hsame}(h, \mathsf{BoolEndpoint}(v)) \wedge \mathsf{hsame}(k, \mathsf{BoolEndpoint}(w)) \wedge \mathsf{msame}(v, w)) \wedge \mathsf{ListClassifierSpec}(\mathsf{BoolHisto}$$

Proof. Invert the cons clause of ListClassifierSpec to obtain the head classifier $\mathsf{BoolHistoryClassifier}(h, k)$ and the recursive tail classifier $\mathsf{ListClassifierSpec}(\mathsf{BoolHistoryClassifier}, xs, ys)$.

Apply Theorem 32.12 to the head classifier. The forward direction supplies endpoints v, w with the two hsame readbacks and $\mathsf{msame}(v, w)$.

Package these endpoint readbacks, mark sameness, and the tail classifier into the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec BoolHistoryClassifier cons mark readback`

Theorem 36.21 (History list classifier singleton equivalence)**.** *For BEDC history singleton lists, structural classification by* hsame *is equivalent to* hsame *of the two displayed histories.*

Proof. Unfold the singleton cons-cons classifier shape. The forward direction keeps the head hsame witness; the reverse direction pairs that head witness with the nil-tail classifier clause. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame singleton iff`

Theorem 36.22 (History list classifier head transport)**.** *For BEDC history lists, if y is internally the same history as z and the structural classifier relates $x :: xs$ to $y :: ys$, then the structural classifier also relates $x :: xs$ to $z :: ys$.*

Proof. Eliminate the cons-cons classifier witness. Compose its head witness $\mathsf{hsame}(x, y)$ with the replacement witness $\mathsf{hsame}(y, z)$, and keep the tail classifier witness unchanged. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame cons head transport`

Theorem 36.23 (List classifier stability under certified entry descent)**.** *Let R, T : BHist $\to$ BHist $\to$ Prop be BEDC history classifiers, and let* cert : $\mathsf{DescentCertificate}(\mathsf{BHist}, \mathsf{BHist}, R, T)$. *If* $\mathsf{ListClassifierSpec}(R, xs, ys)$ *holds, then* $\mathsf{ListClassifierSpec}(T, \mathsf{map}(\mathsf{cert.map}, xs), \mathsf{map}(\mathsf{cert.map}, ys))$ *holds.*

Proof. Induct on the finite list-classifier witness. The nil witness maps to the nil classifier clause. In the cons-cons case, apply descentCertificate_respects to the head witness for cert.map and use the induction hypothesis on the tail witness, then rebuild the cons-cons ListClassifierSpec clause for the mapped lists. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec descentCertificate map`

Theorem 36.24 (History list classifier is stable under $e0$-mapping)**.** *For BEDC history lists, structural classification by* hsame *is preserved when every entry of both lists is mapped through the $e0$ history constructor.*

Proof. Induct on the first list. Nil and cons shape mismatches are impossible by the classifier clauses. In the cons-cons case, apply $e0$-congruence of hsame to the heads and apply the induction hypothesis to the classified tails. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e0`

Theorem 36.25 (History list classifier reflects $e0$-mapping)**.** *For BEDC history lists, if the lists obtained by mapping every entry through the $e0$ history constructor are structurally classified by* hsame*, then the original lists are structurally classified by* hsame*.*

Proof. Induct on the first list. Nil and cons shape mismatches are impossible by the classifier clauses. In the cons-cons case, invert the mapped head witness using the $e0$ constructor characterization for hsame, then apply the induction hypothesis to the mapped-tail classifier witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e0 inversion`

Theorem 36.26 (History list classifier is stable under $e1$-mapping)**.** *For BEDC history lists, structural classification by* hsame *is preserved when every entry of both lists is mapped through the $e1$ history constructor.*

Proof. Induct on the first list. Nil and cons shape mismatches are impossible by the classifier clauses. In the cons-cons case, apply $e1$-congruence of hsame to the heads and apply the induction hypothesis to the classified tails. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1`

Theorem 36.27 (History list classifier reflects $e1$-mapping)**.** *For BEDC history lists, if the lists obtained by mapping every entry through the $e1$ history constructor are structurally classified by* hsame*, then the original lists are structurally classified by* hsame*.*

Proof. Induct on the first list. Nil and cons shape mismatches are impossible by the classifier clauses. In the cons-cons case, invert the mapped head witness using the $e1$ constructor characterization for hsame, then apply the induction hypothesis to the mapped-tail classifier witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1 inversion`

Theorem 36.28 (History list classifier constructor-map embeddings)**.** *For BEDC history lists xs and ys, structural classification by* hsame *is equivalent to structural classification after mapping both lists through the $e0$ history constructor, and likewise equivalent to structural classification after mapping both lists through the $e1$ history constructor.*

Proof. For the $e0$ equivalence, the forward implication is exactly the checked preservation theorem for $e0$-mapping, and the reverse implication is exactly the checked $e0$-mapping inversion theorem. The $e1$ equivalence is obtained in the same way from the checked preservation theorem for $e1$-mapping and the checked $e1$-mapping inversion theorem. Thus each BEDC history constructor preserves and reflects the list classifier, giving a classifier embedding in both constructor-map cases. □

Theorem 36.29 (History list classifier separates mixed mapped tags)**.** *For BEDC history lists, if the list obtained by mapping every entry of x through $e0$ is structurally classified by* hsame *with the list obtained by mapping every entry of y through $e1$, then x and y are both empty.*

Proof. Split both lists by constructors. The nil-nil case is immediate. Nil-cons and cons-nil cases contradict the structural classifier clauses. In the cons-cons case, the head witness would relate an $e0$-tagged history to an $e1$-tagged history, contradicting BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e0 map e1 empty`

Theorem 36.30 (History list classifier preserves length)**.** *For lists of finite-kernel histories, structural classification by* hsame *can only relate lists with equal length.*

Proof. Induct on the first list. Nil can only classify with nil. In the cons case, classifier inversion forces the second list to be cons, and the induction hypothesis gives equality of the tail lengths. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame length eq`

Theorem 36.31 (List classifier preserves length)**.** *For finite-spine presentation witnesses only, write $\ell(xs)$ for their meta-level finite length; this notation is neither a* $\mathsf{Nat}^{\uparrow}$ *operation nor a certificate field. For any source type A, any source relation $sameA : A \to A \to \mathsf{Prop}$, and finite spines $xs, ys : \mathsf{ListCarrier}(A)$, if* $\mathsf{ListClassifierSpec}(sameA, xs, ys)$ *holds, then*

$$\ell(xs) = \ell(ys).$$

Proof. Induct on the first list spine. The nil case can classify only against nil by the mixed-shape clauses of ListClassifierSpec. In the cons case, the classifier can classify only against a cons spine; its tail witness is again a ListClassifierSpec witness for entries related by $sameA$. The induction hypothesis gives equality of tail lengths, and the two cons lengths are equal. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec length eq`

Definition 36.32 (Finite-spine helper stability certificate)**.** *The checked finite-spine fields recorded in this chapter are the marked* ListClassifierSpec *facts: reflexivity, symmetry, source-certificate transitivity, nil/cons separation with cons inversion, constructor-map preservation/reflection with mixed-tag separation, append stability, checked* hsame *left and right cancellation, reverse stability with reverse inversion, and checked* hsame *length preservation. Reverse equivalence, source-parametric left, right, and context cancellation, source-certificate symmetry, source-certificate carrier transport, and source-parametric length preservation are paper-side composite targets built from these checked fields. The public generated-history* $\mathsf{List}^{\uparrow}$ *certificate is obtained only through* ListSpineRep*,* ListSpineBridgeClassifier*,* ListSpineRepCoherent*, and Theorem 36.68.*

Theorem 36.33 (List classifier reflexivity)**.** *If the source classifier is reflexive on A, then* ListClassifierSpec *is reflexive on finite list spines over A.*

Lean4 ✓ `BEDC.Derived.ListUp.list stability reflexive by induction`

Proof. Induct on the finite list spine. The nil spine is classified by the nil clause. In the cons spine, source reflexivity classifies the head and the induction hypothesis classifies the tail. □

Theorem 36.34 (List classifier stability under append)**.** *For any source entry type A, source classifier $sameA : A \to A \to \mathsf{Prop}$, and finite list spines xs, ys, zs, and ws, if* $\mathsf{ListClassifierSpec}(sameA, xs, ys)$ *and* $\mathsf{ListClassifierSpec}(sameA, zs, ws)$ *hold, then* $\mathsf{ListClassifierSpec}(sameA, xs{+}{+}zs, ys{+}{+}ws)$ *holds.*

Proof. Induct on the first classifier witness. The nil clause returns the second classifier witness. In the cons clause, keep the head classifier witness and apply the induction hypothesis to the tail classifier witness and the second classifier witness, then rebuild the cons classifier clause. □

Theorem 36.35 (List classifier stability under append for histories)**.** *For BEDC histories, if two list pairs are structurally classified by* hsame*, then appending the first components and appending the second components preserves structural classification.*

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec append hsame`

Proof. This is the BEDC-history specialization of Theorem 36.34, with the source entry type specialized to BHist and *sameA* specialized to hsame. The empty and cons clauses are exactly the two clauses of the source-parametric append proof. □

Theorem 36.36 (History list classifier left append cancellation)**.** *For BEDC history lists, structural classification of $p{+}{+}x$ with $p{+}{+}y$ using* hsame *on heads determines structural classification of x with y.*

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame append left cancel`

Proof. Induct on the common prefix p. The empty-prefix case is the given classification. In the cons-prefix case, invert the structural classifier and apply the induction hypothesis to the classified tails. □

Theorem 36.37 (Classified list prefix cancellation)**.** *For BEDC history lists, if p and p' are structurally classified by* hsame*, and $p{+}{+}x$ is structurally classified with $p'{+}{+}y$, then x is structurally classified with y.*

Proof. Induct on p. The empty case forces p' to be empty by the prefix classifier, leaving the appended classifier. In the cons case, the prefix classifier and appended classifier both expose matching classified tails, and the induction hypothesis applies to those tails. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame append left cancel classified`

Theorem 36.38 (List classifier stability under reverse)**.** *For BEDC history lists, structural classification by* hsame *is preserved when both lists are reversed.*

Proof. Induct on the first list. Shape mismatches are impossible by the classifier clauses. In the cons-cons case, reverse turns each list into the reversed tail followed by a singleton head; the induction hypothesis handles the tails and append stability combines it with the head witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec reverse hsame`

Theorem 36.39 (Source-parametric list classifier stability under reverse)**.** *For any source type A, source relation $\sim_A : A \to A \to \mathsf{Prop}$, and finite spines $xs, ys : \mathsf{ListCarrier}(A)$, $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ implies $\mathsf{ListClassifierSpec}(\sim_A, \mathsf{reverse}(xs), \mathsf{reverse}(ys))$.*

Proof. Induct on the first spine and case-analyze the second. Mixed shapes are impossible by the classifier clauses. In the cons-cons case, reverse writes both spines as reversed tails followed by singleton heads; the induction hypothesis classifies the reversed tails, and append stability combines that tail witness with the singleton head classifier witness. □

Theorem 36.40 (List classifier reverse inversion)**.** *For BEDC history lists, if the reversed lists are structurally classified by* hsame*, then the original lists are structurally classified by* hsame*.*

Proof. Apply reverse stability to the classified reversed lists, obtaining a classifier on $\mathsf{reverse}(\mathsf{reverse}(xs))$ and $\mathsf{reverse}(\mathsf{reverse}(ys))$. For each finite spine, prove the reverse-reverse identity by induction, using append associativity and append-nil at the cons step. Rewrite both endpoints along those two identities to recover $\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec reverse hsame inversion`

Theorem 36.41 (Source-parametric list classifier reverse equivalence)**.** *For any source type A, source relation $\sim_A : A \to A \to \mathsf{Prop}$, and finite spines $xs, ys : \mathsf{ListCarrier}(A)$, list classification by $\sim_A$ is equivalent before and after reversing both spines:*

$$\mathsf{ListClassifierSpec}(\sim_A, xs, ys) \iff \mathsf{ListClassifierSpec}(\sim_A, \mathsf{reverse}(xs), \mathsf{reverse}(ys)).$$

Proof. The forward direction is Theorem 36.39.

For the reverse direction, apply the same source-parametric reverse stability to the classified reversed spines. This gives a classifier on $\mathsf{reverse}(\mathsf{reverse}(xs))$ and $\mathsf{reverse}(\mathsf{reverse}(ys))$. Reverse-involutivity of finite spines identifies these endpoints with xs and ys, yielding $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec reverse iff`

Theorem 36.42 (List classifier append right cancellation)**.** *For BEDC history lists, if adding the same right suffix to two lists gives a structural* hsame*-classification, then the original lists are structurally classified.*

Proof. Reverse the classified append witnesses. The common right suffix becomes a common left prefix, so left-append cancellation removes it. Reversing the remaining classifier witness returns the original list order. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec append right cancel hsame`

Theorem 36.43 (List classifier right cancellation with equivalent suffixes)**.** *For BEDC history lists, if two right suffixes are structurally classified by* hsame*, and appending those suffixes to x and y gives a structural* hsame*-classification, then x and y are structurally classified.*

Proof. Use symmetry on the suffix classifier and append stability with reflexivity on the left list y to classify $y{+}{+}suffix'$ with $y{+}{+}suffix$. Transitivity aligns the appended classification to a common right suffix, and ordinary right-append cancellation removes it. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec append right cancel with hsame suffix`

Theorem 36.44 (List classifier two-sided cancellation with classified boundaries)**.** *For BEDC history lists p, p', x, y, s, and s', if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}x{+}{+}s, p'{+}{+}y{+}{+}s')$$

hold, then $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$ *holds.*

Proof. Reassociate both appended histories as $(p{+}{+}x){+}{+}s$ and $(p'{+}{+}y){+}{+}s'$. Apply right cancellation with the classified suffixes to obtain $\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}x, p'{+}{+}y)$. Then apply classified left cancellation with the classified prefixes p and p' to obtain $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame append boundary cancel`

Theorem 36.45 (List classifier classified append context equivalence)**.** *For BEDC history lists p, p', x, y, s, and s', if the prefix contexts p and p' are structurally classified by* hsame *and the suffix contexts s and s' are structurally classified by* hsame*, then*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y) \quad \textit{if and only if} \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}x{+}{+}s, p'{+}{+}y{+}{+}s')$$

holds.

Proof. For the forward direction, apply append stability first to the classified prefixes and middle lists, obtaining a classification of $p{+}{+}x$ with $p'{+}{+}y$, and then apply append stability again with the classified suffixes. For the reverse direction, apply the preceding two-sided cancellation theorem to the classified bordered histories, the classified prefixes, and the classified suffixes. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame append context iff`

Theorem 36.46 (List classifier transitivity from a source certificate)**.** *If the source classifier is supplied by a* NameCert*, then the lifted list classifier is transitive on finite list spines.*

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec trans from nameCert`

Proof. Proceed by induction on the first list. The nil case forces the middle list to be nil, then the final list is nil. In the cons case, nil/cons mismatches are impossible by classifier inversion. For matching cons constructors, transitivity of the source certificate handles the heads and the induction hypothesis handles the tails. □

Theorem 36.47 (List classifier symmetry from a source certificate)**.** *For a BEDC history carrier A : $\mathsf{BHist} \to \mathsf{Prop}$, source classifier $\sim_A$, and source certificate* cert : $\mathsf{NameCert}(A, \sim_A)$*, if* $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ *holds for finite list spines xs and ys, then* $\mathsf{ListClassifierSpec}(\sim_A, ys, xs)$ *holds.*

Proof. Induct on the structural list-classifier witness. The nil clause reverses to nil. In the cons clause, apply cert.equiv_symm to the head witness and apply the induction hypothesis to the tail witness; the cons constructor of the lifted classifier rebuilds the reversed structural witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec symm from nameCert`

Theorem 36.48 (List carrier transport from a source certificate)**.** *Let C : $\mathsf{BHist} \to \mathsf{Prop}$ be a source carrier predicate, let $\sim_C$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be a source classifier, and assume* cert : $\mathsf{NameCert}(C, \sim_C)$*. For finite BEDC-history spines xs, ys :* $\mathsf{ListCarrier}(\mathsf{BHist})$*, if every entry of xs satisfies C and* $\mathsf{ListClassifierSpec}(\sim_C, xs, ys)$ *holds, then every entry of ys satisfies C.*

Proof. Induct on the structural classifier derivation. The nil clause has no entries to transport. In the cons clause, the source certificate field NameCert.carrier_respects_equiv transports the head carrier witness across the head classifier witness, and the induction hypothesis transports all tail entries across the tail classifier witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec nameCert carrier transport`
Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec carrier transport from nameCert`

Definition 36.49 (List ledger policy)**.** *Record the finite constructor spine and source-carrier evidence for each visible element; the paper-side* ListHistoryLedger *used in Theorem 36.68 does not carry a separate source ledger.*

Definition 36.50 (List history ledger)**.** *For a source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$*,* $\mathsf{ListHistoryLedger}_S(h)$ *records the visible history endpoint as a finite nil/cons spine: the empty endpoint is recorded by* $\mathsf{hsame}(h, \mathsf{e})$*, and a cons endpoint is recorded by histories* a, t, p *with* $S(a)$*,* $\mathsf{ListHistoryLedger}_S(t)$*,* $\mathsf{Cont}(a, t, p)$*, and* $\mathsf{hsame}(h, \mathsf{E1}(p))$*.*

Theorem 36.51 (List history carrier, pattern, and ledger equivalence)**.** *For every source carrier* $S : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, the three displayed list-history fields determine the same endpoints:*

$$\mathsf{ListHistoryCarrier}_S(h) \iff \mathsf{ListHistoryPattern}_S(h),$$

and

$$\mathsf{ListHistoryCarrier}_S(h) \iff \mathsf{ListHistoryLedger}_S(h).$$

Proof. For the carrier-pattern equivalence, use the displayed generated cases for $\mathsf{ListHistoryCarrier}_S$. The nil case gives $\mathsf{hsame}(h, \mathsf{e})$, which is the nil pattern branch. The cons case exposes histories a, t, p with $S(a)$, $\mathsf{ListHistoryCarrier}_S(t)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$, which is exactly the cons pattern branch. Conversely, each pattern branch is one of the two carrier constructors, using the same hsame, $S(a)$, $\mathsf{Cont}(a, t, p)$, and tail carrier witnesses.

For the carrier-ledger equivalence, induct over the displayed nil/cons generation in each direction. From carrier to ledger, the nil constructor becomes the nil ledger clause, and the cons constructor reuses $S(a)$, $\mathsf{Cont}(a, t, p)$, the endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$, and the induction hypothesis on the tail. From ledger to carrier, the nil ledger clause rebuilds the nil carrier clause, and the cons ledger clause reuses the same source, continuation, endpoint, and tail witnesses, with the induction hypothesis turning the tail ledger witness into the required tail carrier witness. □

Theorem 36.52 (List public finite-data exit package)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. If* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then the public* $\mathsf{List}^{\uparrow}$ *object carries the Phase Finite Data exit package:*

1. $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\sim_A})$.
2. *The public carrier is exact against both the finite pattern predicate and the ledger predicate:*

$$\forall h.\ \mathsf{ListHistoryCarrier}_A(h) \iff \mathsf{ListHistoryPattern}_A(h)$$

and

$$\forall h.\ \mathsf{ListHistoryCarrier}_A(h) \iff \mathsf{ListHistoryLedger}_A(h).$$

3. *Public length is total and well-defined:*

$$\forall h.\ \mathsf{ListHistoryCarrier}_A(h) \to \exists n.\ \mathsf{ListPublicLength}_A(h, n)$$

and

$$\forall h\, n\, m.\ \mathsf{ListPublicLength}_A(h, n) \to \mathsf{ListPublicLength}_A(h, m) \to n = m.$$

4. *Classified endpoints share public length:*

$$\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \to \exists n.\ \mathsf{ListPublicLength}_A(h, n) \wedge \mathsf{ListPublicLength}_A(k, n)$$

and

$$\forall h\, k\, n\, m.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListPublicLength}_A(h, n) \to \mathsf{ListPublicLength}_A(k, m) \to n = m.$$

5. *Every classified endpoint pair exposes finite spine ledgers:*

$$\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h,xs)) \\ \wedge \\ (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k,ys)) \end{array}\right).$$

6. *The same classifier is exactly the nil-nil or cons-cons constructor audit:*

$$\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k) \iff \begin{array}{l} (\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListH} \\ \mathsf{Cont}(a,t,p) \wedge \mathsf{Cont}(b,u,q) \wedge \mathsf{hsame}(h,\mathsf{E1}(p \\ \sim_A (a,b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, \end{array}\right) \end{array}$$

7. *The classified successor component readback package is independent of the chosen six-tuple witness, in the sense of Theorem 36.102.*

Proof. Apply Theorem 36.78 to cert and the cons-boundary coherence witness. Its five displayed fields give the semantic certificate, totality of public length, well-definedness of public length, shared length for classified endpoints, and equality of public length values across classified endpoints.

Apply Theorem 36.51 with $S = A$. This supplies the exactness of $\mathsf{ListHistoryCarrier}_A$ against both $\mathsf{ListHistoryPattern}_A$ and $\mathsf{ListHistoryLedger}_A$.

Next apply Theorem 36.165. This supplies the endpoint spine-ledger exposure for every $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k)$ witness and the nil-or-cons constructor audit for the same classifier.

Finally use Theorem 36.102. It turns the classified successor exactness clause into a witness-choice independent component package for the public object. Repack the three theorem outputs into the seven displayed fields. □

Theorem 36.53 (List finite-data consumer row exhaustion)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. If* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then every public consumer read of the* $\mathsf{List}^{\uparrow}$ *finite-data package factors through exactly the seven rows displayed in Theorem 36.52:*

1. *semantic certification of the lifted list carrier, pattern, ledger, and classifier;*
2. *exactness of the public carrier against the finite carrier-pattern and carrier-ledger predicates;*
3. *totality and uniqueness of public length;*
4. *sharing and uniqueness of public length along classified endpoint pairs;*
5. *finite spine-ledger exposure for each classified endpoint;*
6. *the nil-or-cons constructor audit for the public classifier;*
7. *independence of the classified successor component readback from the chosen six-tuple witness.*

No public finite-data consumer obtains an additional row from host list equality, quotient representatives, or external sequence objects.

Proof. Apply Theorem 36.52. It presents the public finite-data package as the seven displayed rows, so any public consumer read first enters through one of those rows.

The exactness rows have no further public exits. Apply Theorem 36.51: the carrier read is identified with the finite pattern predicate and the ledger predicate, and no separate host-list equality object is introduced by this transport.

The endpoint and constructor rows are likewise exhausted internally. Apply Theorem 36.165 to account for endpoint spine-ledger exposure and the nil-or-cons audit. Then apply Theorem 36.102 to make the successor component readback independent of the chosen six-tuple witness.

These applications cover the semantic certificate, the two exactness rows, the two public length rows, the endpoint exposure row, the constructor audit row, and the successor-component independence row. Since none of the cited clauses exports quotient representatives or external sequence objects, the displayed seven rows form the whole public consumer boundary. □

Theorem 36.54 (List public closure alignment)**.** *For* $\mathsf{List}^{\uparrow}$ *over* BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *the public finite-data exit package and the consumer row exhaustion theorem jointly supply the paper-side Phase Finite Data witness. The witness consists exactly of the carrier-pattern and carrier-ledger exactness rows, the public length rows, the constructor audit, and the successor-component independence package already displayed by Definition 36.52 and exhausted by Definition 36.53. It does not add host list equality, quotient representatives, or an external sequence object.*

Proof. Apply Definition 36.52. Its displayed fields give the finite public object: semantic certification, carrier-pattern and carrier-ledger exactness, public length totality and uniqueness, classified endpoint length agreement, finite spine exposure, the nil-or-cons constructor audit, and successor-component independence.

Apply Definition 36.53. It says that any public consumer read factors through exactly those rows. Hence the roadmap witness is obtained by repackaging the already exposed rows for the $\mathsf{List}^{\uparrow}$ surface over BHist, hsame, Cont, and $\mathsf{NameCert}$.

The cited package and exhaustion theorem explicitly close off host list equality, quotient representatives, and external sequence objects. Therefore the alignment is a paper-side finite-data witness rather than a stronger formal marker or an additional public carrier. □

36.4 List map shape

Theorem 36.55 (History list classifier reflects one-sided $e1$ shape)**.** *For BEDC history lists xs and ys, if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(\mathsf{E1}, xs), ys)$$

holds, then there is a BEDC history list zs such that

$$ys = \mathsf{map}(\mathsf{E1}, zs) \quad \textit{and} \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, zs)$$

hold.

Proof. Induct on xs. The nil case can classify only against nil, so choose $zs = \mathsf{nil}$. In the cons case, the classifier forces ys to be a cons list. Invert the head witness with the $e1$ history-shape inversion theorem to write the head of ys as an $e1$-tagged history and recover the source-head hsame witness. The induction hypothesis shapes the tail and supplies the tail classifier; consing the recovered head onto that tail gives the displayed witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1 left shape`

Theorem 36.56 (History list classifier separates $e1$ from $e0$ mapped tags)**.** *For BEDC history lists xs and ys, if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(\mathsf{E1}, xs), \mathsf{map}(\mathsf{E0}, ys))$$

holds, then $xs = \mathsf{nil}$ and $ys = \mathsf{nil}$.

Proof. Case-split both lists. The nil-nil case is immediate. Nil-cons and cons-nil cases contradict the mixed-shape clauses of ListClassifierSpec. In the cons-cons case, the head witness would relate an $e1$-tagged history to an $e0$-tagged history, contradicting BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1 map e0 empty`

Definition 36.57 (List spine representation)**.** *Let* HistSpine *be the finite-spine carrier whose entries are BEDC histories. For a BEDC history source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$, the relation $\mathsf{ListSpineRep}_A(h, xs)$ has $xs : \mathsf{HistSpine}$ and says that h represents xs by the same nil and cons clauses used by $\mathsf{ListHistoryCarrier}_A$; the source predicate is carried by the clauses:*

1. *nil: $xs = \mathsf{nil}$ and $\mathsf{hsame}(h, \mathsf{e})$;*
2. *cons: $xs = \mathsf{cons}(a, xs')$ and there are histories t, p with $A(a)$, $\mathsf{ListSpineRep}_A(t, xs')$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$.*

The spine-bridge classifier used by the certificate is

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \iff \exists xs, ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs) \wedge \mathsf{ListSpineRep}_A(k, ys) \wedge \mathsf{ListClas}$$

The certificate classifier is this bridge classifier:

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \iff \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k).$$

Thus ListClassifierSpec *remains the finite-spine classifier,* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$ *names the BEDC-history bridge through finite spines, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$ *is the certificate classifier whose endpoints are BEDC histories. The recursive classifier* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{S,\sim_S}$ *from Definition 36.4 is a paper-side generator for carrier and endpoint reasoning.*

Lean4 (definition) `BEDC.Derived.ListUp.ListSpineRep`

Theorem 36.58 (List spine nil-cons no-confusion)**.** *For every BEDC history source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$, history h, source history a, and tail spine $xs : \mathsf{HistSpine}$, $\mathsf{ListSpineRep}_A(h, \mathsf{nil})$ and $\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs))$ are contradictory.*

Proof. Unpack the nil representation to obtain $\mathsf{hsame}(h, \mathsf{e})$. Unpack the cons representation to obtain an endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$ for some continuation package p. Compose the symmetry of the nil endpoint with the cons endpoint to get $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(p))$, then close by BEDC history constructor no-confusion between the empty history and a one-step endpoint. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep nil cons no confusion`

Lemma 36.59 (List spine cons hsame transport)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier. If $A(a)$ holds, the tail history t represents a finite spine xs, the continuation $\mathsf{Cont}(a, t, p)$ holds, and h exposes the cons endpoint $\mathsf{E1}(p)$, then every h' with $\mathsf{hsame}(h, h')$ represents the cons spine:*

$$A(a) \to \mathsf{ListSpineRep}_A(t, xs) \to \mathsf{Cont}(a, t, p) \to \mathsf{hsame}(h, \mathsf{E1}(p)) \to \mathsf{hsame}(h, h') \to \mathsf{ListSpineRep}_A(h', \mathsf{cons}(a, xs)).$$

Proof. Use the cons constructor of Definition 36.57. The head source witness, tail representation, and continuation witness are unchanged. The endpoint witness for h' is obtained by composing $\mathsf{hsame}(h', h)$, the symmetry of the displayed endpoint transport, with the stored cons endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep cons hsame transport`

Theorem 36.60 (List spine cons endpoint shape)**.** *For every source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$, history h, head a, and tail spine $xs : \mathsf{HistSpine}$, a cons spine presentation $\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs))$ exposes its BEDC constructor payload:*

$$\exists t, p : \mathsf{BHist}.\ A(a) \wedge \mathsf{ListSpineRep}_A(t, xs) \wedge \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)).$$

Proof. Invert the displayed cons presentation. The cons constructor of $\mathsf{ListSpineRep}$ stores exactly the head source proof, tail spine presentation, continuation ledger, and one-headed endpoint classifier. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep cons endpoint shape`

Theorem 36.61 (List spine representation exactness)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier and let $\sim_A$ be a source classifier. The spine representation of Definition 36.57 is exact for the generated list carrier and classifier:*

1. *if $\mathsf{ListSpineRep}_A(h, xs)$ holds, then $\mathsf{ListHistoryCarrier}_A(h)$ holds;*
2. *if $\mathsf{ListHistoryCarrier}_A(h)$ holds, then there is a finite spine xs such that $\mathsf{ListSpineRep}_A(h, xs)$ holds;*
3. *for all histories h, k, the spine-induced classifier $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k)$ holds exactly when the recursive generated classifier $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h, k)$ holds.*

Proof. For soundness, induct on the displayed derivation of $\mathsf{ListSpineRep}_A(h, xs)$. The nil representation is exactly the nil generator of $\mathsf{ListHistoryCarrier}_A$, and the cons representation carries the head witness $A(a)$, the recursive representation of the tail, the continuation witness $\mathsf{Cont}(a, t, p)$, and the endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$, so the cons generator rebuilds the carrier.

For completeness, induct on the displayed generation of $\mathsf{ListHistoryCarrier}_A(h)$. The nil carrier chooses $xs = \mathsf{nil}$ and the stored empty endpoint witness. The cons carrier first applies the induction hypothesis to the tail carrier, then chooses $xs = \mathsf{cons}(a, xs')$ and reuses the source, continuation, and endpoint witnesses.

For classifier equivalence, unfold both classifier presentations. A bridge witness exposes finite spines and a $\mathsf{ListClassifierSpec}$ witness; applying carrier soundness to both representation endpoints gives the generated list histories, and induction on the finite-spine classifier derivation rebuilds the nil-nil and cons-cons classifier clauses. Conversely, a generated classifier derivation is handled by induction on its nil/cons clauses: the nil case chooses two nil spines, and the cons case uses the induction hypotheses for the tail classifier together with the stored source head classification and continuation witnesses to construct the finite-spine classifier witness. □

Definition 36.62 (List spine representation coherence)**.** *For a BEDC history source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$ and source classifier $\sim_A$, define $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ as the proposition that every two finite-spine presentations of the same history are classified by the finite spine classifier. Concretely, for every history m and spines $ys, ys' : \mathsf{HistSpine}$, if $\mathsf{ListSpineRep}_A(m, ys)$ and $\mathsf{ListSpineRep}_A(m, ys')$, then $\mathsf{ListClassifierSpec}(\sim_A, ys, ys')$.*

Theorem 36.63 (List history classifier transitivity from spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. If* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ *holds, then the existential spine-representation classifier of Definition 36.57 is transitive: whenever* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k)$ *and* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k, u)$ *hold, then* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, u)$ *holds.*

Proof. Unpack the two history-classifier witnesses. The first gives spines xs, ys with representations of h and k and a witness $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. The second gives spines ys', zs with representations of k and u and a witness $\mathsf{ListClassifierSpec}(\sim_A, ys', zs)$. Apply Definition 36.62 to the two displayed representations of the middle history k to obtain $\mathsf{ListClassifierSpec}(\sim_A, ys, ys')$. The checked finite-spine theorem ListClassifierSpec_trans_from_nameCert, applied with cert, first composes xs with ys' through ys and then composes the result with zs through ys'. Repack the original representation of h, the original representation of u, and the composed finite-spine classifier witness to obtain $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, u)$. □

Theorem 36.64 (List spine classifier preserves represented length)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. Then represented finite spines have equal presentation length along the spine-induced history classifier: whenever* $\mathsf{ListSpineRep}_A(h, xs_0)$*,* $\mathsf{ListSpineRep}_A(k, ys_0)$*, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *hold, then*

$$\ell(xs_0) = \ell(ys_0).$$

Proof. Apply Theorem 36.144 to the two displayed representation witnesses and the spine-induced history classifier, using cert and the coherence hypothesis. This gives $\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$. Theorem 36.31 then reads this finite-spine classifier witness as $\ell(xs_0) = \ell(ys_0)$. □

Theorem 36.65 (List spine representation coverage)**.** *For every BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and history* h*, the generated history carrier is covered exactly by finite spine representations:*

$$\mathsf{ListHistoryCarrier}_A(h) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs).$$

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRepresentation coverage`

Proof. For the forward direction, induct on the generated $\mathsf{ListHistoryCarrier}_A(h)$ witness. The nil constructor is represented by the empty finite spine. The cons constructor uses the induction hypothesis for the tail carrier witness to obtain a tail spine, then prefixes it by the source history and reuses the continuation and endpoint witnesses. For the reverse direction, induct on the $\mathsf{ListSpineRep}_A(h, xs)$ witness. The nil representation rebuilds the nil carrier clause, and the cons representation rebuilds the cons carrier clause from the source witness, the recursively recovered tail carrier witness, the continuation witness, and the displayed $\mathsf{hsame}(h, \mathsf{E1}(p))$ endpoint. □

Theorem 36.66 (List spine bridge certificate fields)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The bridge classifier* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$ *has the certificate fields needed over the carrier* $\mathsf{ListHistoryCarrier}_A$*:*

1. *if* $\mathsf{ListHistoryCarrier}_A(h)$ *holds, then* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, h)$ *holds;*
2. *if* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k)$ *holds, then* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k, h)$ *holds;*

3. *if* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k)$ *and* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k,u)$ *hold, then* $\mathsf{ListSpineBridgeClassifier}_{A,\sim}$ *holds;*
4. *if* $\mathsf{ListHistoryCarrier}_A(h)$ *and* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k)$ *hold, then* $\mathsf{ListHistoryCarrier}_A(k)$ *holds.*

Proof. For carrier-indexed reflexivity, apply the forward direction of Theorem 36.65 to the carrier witness for h. This gives a finite spine xs and a representation witness $\mathsf{ListSpineRep}_A(h,xs)$. Finite-spine reflexivity for $\mathsf{ListClassifierSpec}$, using the reflexivity field of cert on each source entry, repacks with the same representation on both endpoints and gives $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,h)$.

For symmetry, unpack the bridge witness into displayed endpoint spines and a finite-spine classifier witness. Apply Theorem 36.47 with cert to reverse that finite-spine witness, then repack the two displayed representations in the opposite order.

For transitivity, use Theorem 36.63. The coherence hypothesis compares the two finite-spine presentations of the middle history, and the source certificate supplies the finite-spine transitivity used there.

For carrier transport, unpack $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k)$ and take the displayed target representation $\mathsf{ListSpineRep}_A(k,ys)$. The soundness direction of Theorem 36.61, equivalently the reverse direction of Theorem 36.65, converts this representation into $\mathsf{ListHistoryCarrier}_A(k)$. □

Principle 36.67 (Unframed list-spine presentations require an external coherence field)**.** *Definition 36.57 supplies coverage and endpoint presentation data for a chosen finite-spine presentation only. It does not include the comparison datum required by Definition 36.62. For the bridge classifier, the missing datum is exactly a proof that any two cons presentations of the same endpoint have source-classified heads and* $\mathsf{ListClassifierSpec}$*-classified tails. Thus* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ *is an additional certificate hypothesis needed by Theorem 36.63. That comparison has to come from a framed or separated spine representation, from a source presentation whose restrictions prove it directly, or from the cons-boundary coherence condition: Theorem 36.71 supplies the forward construction, and Theorem 36.72 records the equivalence with spine coherence.*

Theorem 36.68 (Conditional $\mathsf{List}^{\uparrow}$ public-object target from the spine bridge)**.** *Given a BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$*, source classifier* $\sim_A$*, source certificate* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and a witness of* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*, define* $\mathsf{ListSource}(A)$ *as the history carrier* $\mathsf{ListHistoryCarrier}_A$*, take the certificate classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$ *to be the spine-induced bridge classifier from Definition 36.57, use Theorem 36.61 as its finite-spine exactness theorem, take* $\mathsf{ListHistoryPattern}_A$ *from Definition 36.11 as the pattern, and take* $\mathsf{ListHistoryLedger}_A$ *from Definition 36.50 as the ledger. The host-list spine model is a presentation witness, not the exported carrier. Under the source* $\mathsf{NameCert}$ *and the spine-coherence witness, these fields form the paper-side target* $\mathsf{SemanticNameCert}(\mathsf{ListHist}$ *for* $\mathsf{List}^{\uparrow}(A)$*. This theorem is the paper-side target and carries no* `\leanchecked` *marker.*

Proof. The source $\mathsf{NameCert}$ supplies the head-level reflexivity, symmetry, transitivity, and carrier transport fields. The generated nil/cons carrier supplies the BEDC histories, and Theorem 36.61 supplies the exact finite-spine presentation through which the checked structural classifier is applied. The reflexivity, symmetry, transitivity, and carrier-transport fields for the bridge classifier are exactly Theorem 36.66. Source-certificate transport is used only for entrywise source-carrier evidence, while the list ledger is the carrier-shaped constructor spine from Definition 36.50. Nil/cons case analysis uses Theorem 36.7 for the generated history carrier and Lemma 36.14 inside the finite-spine bridge, and exactness means that there exist finite spines xs, ys with $\mathsf{ListSpineRep}_A(h,xs)$, $\mathsf{ListSpineRep}_A(k,ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$; Definition 36.62 makes alternate presentations of the same endpoint compatible for certificate use. □

Corollary 36.69 ($\mathsf{List}^{\uparrow}$ conditional public-object target)**.** *By Theorem 36.68,* $\mathsf{List}^{\uparrow}(A)$ *has an unframed public certificate target relative to the supplied source* NameCert *and a witness of* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. Theorem 36.163 gives the unframed public certificate when* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *is supplied, equivalently by Theorem 36.72. Corollary 36.128 supplies a separate framed carrier certificate under source-carrier* hsame *compatibility and does not identify the framed carrier with the unframed* $\mathsf{ListHistoryCarrier}_A$ *used by the public* $\mathsf{List}^{\uparrow}$ *target.*

Definition 36.70 (List cons-boundary coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. Define* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *as the predicate*

$$\forall m\, a\, a'\, t\, t'\, p\, p'\, xs\, xs'.\ A(a) \to A(a') \to \mathsf{ListSpineRep}_A(t, xs) \to \mathsf{ListSpineRep}_A(t', xs') \to \mathsf{Cont}(a, t, p) \to \mathsf{Cont}(a', t', p$$

Lean4 (definition) `BEDC.Derived.ListUp.ListConsBoundaryCoherent`

Theorem 36.71 (List spine coherence from cons-boundary coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. If* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ *holds.*

Proof. Compare two finite-spine presentations of the same history by cases on the displayed spines. The nil-nil case is the nil constructor of ListClassifierSpec. The mixed nil-cons cases contradict Theorem 36.58. In the cons-cons case, both presentations expose the same list endpoint together with displayed head histories, tail spines, tail representations, continuation witnesses, and endpoint witnesses. Definition 36.70 supplies the conjunction $(a \sim_A a') \wedge \mathsf{ListClassifierSpec}(\sim_A, xs, xs')$; its left conjunct is the head classifier witness, and its right conjunct is the tail finite-spine classifier witness. These are the two constructor arguments for $\mathsf{ListClassifierSpec}(\sim_A, -, -)$ on the whole cons spines. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep coherent from cons boundary`

Theorem 36.72 (List spine coherence iff cons-boundary coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. Then* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ *holds if and only if* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds.*

Proof. For the forward implication, assume $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Given two displayed cons presentations of the same endpoint, view them as two $\mathsf{ListSpineRep}_A$ presentations with cons spines. Spine coherence returns a $\mathsf{ListClassifierSpec}(\sim_A, -, -)$ witness relating those two cons spines. The cons-cons branch of Theorem 36.13 inverts that witness and exposes exactly the required head classifier $a \sim_A a'$ together with the tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, xs')$. These are the two fields of Definition 36.70. The reverse implication is precisely Theorem 36.71. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep cons boundary coherence iff`

Theorem 36.73 (List public cons readback determinacy)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. For any* $m, a, a', t, t', p, p' : \mathsf{BHist}$*, if* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*,* $A(a)$*,* $A(a')$*,* $\mathsf{ListHistoryCarrier}_A(t)$*,* $\mathsf{ListHistoryCarrier}_A(t')$*,* $\mathsf{Cont}(a, t, p)$*,* $\mathsf{Cont}(a', t', p')$*,* $\mathsf{hsame}(m, \mathsf{E1}(p))$*, and* $\mathsf{hsame}(m, \mathsf{E1}(p'))$ *hold, then*

$$a \sim_A a' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t').$$

The roadmap ten-field fit is: traditional target finite-list constructor readback; BEDC source BHist*; source carrier A; public carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability supplied by* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*; ledger two cons endpoints; exactness through finite spine representations; standard bridge finite spines; status* ProofTarget*.*

Proof. Apply Theorem 36.65 to the two displayed tail carrier witnesses. This gives finite spines xs and xs' with $\mathsf{ListSpineRep}_A(t, xs)$ and $\mathsf{ListSpineRep}_A(t', xs')$.

Use Definition 36.70 on the two displayed cons presentations of the endpoint m. The hypotheses supply the source membership for a and a', the two finite tail presentations, the two continuation witnesses, and the two endpoint equalities. The resulting conjunction gives $a \sim_A a'$ and $\mathsf{ListClassifierSpec}(\sim_A, xs, xs')$.

Repack the two tail spine representations together with $\mathsf{ListClassifierSpec}(\sim_A, xs, xs')$ through Definition 36.57. This yields $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t')$, which pairs with the head classifier to give the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier public cons readback deterministic`

Theorem 36.74 (Deterministic list spine length readback)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier and let $\sim_A$ be a source classifier. For every history m and finite spines xs and ys, if $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$, $\mathsf{ListSpineRep}_A(m, xs)$, and $\mathsf{ListSpineRep}_A(m, ys)$ hold, then*

$$\ell(xs) = \ell(ys).$$

The roadmap ten-field fit is: traditional target finite-list readback; BEDC source BHist*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* ListClassifierSpec*; ledger finite spine; status* ProofTarget*.*

Proof. Theorem 36.71 turns the cons-boundary coherence witness into $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Apply this coherence datum to the two displayed $\mathsf{ListSpineRep}_A$ presentations of the endpoint m to obtain $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Then Theorem 36.31 gives equality of the two finite spine lengths. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep cons boundary length deterministic`

Definition 36.75 (List public length readback)**.** *For a BEDC history source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$, define $\mathsf{ListPublicLength}_A(h, n)$ by*

$$\mathsf{ListPublicLength}_A(h, n) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs) \wedge \ell(xs) = n.$$

Here ℓ is the finite-spine presentation length. It is not a $\mathsf{Nat}^{\uparrow}$ operation and is not an additional field of the public certificate.

Theorem 36.76 (List public length readback is well-defined under cons-boundary coherence)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, and let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. Assume $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. Then the public length readback has the following two properties.*

1. *If $\mathsf{ListPublicLength}_A(h, n)$ and $\mathsf{ListPublicLength}_A(h, m)$, then $n = m$.*
2. *If $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$, $\mathsf{ListPublicLength}_A(h, n)$, and $\mathsf{ListPublicLength}_A(k, m)$, then $n = m$.*

The roadmap ten-field fit is: traditional target finite-list length readback; BEDC source BHist *with* NameCert*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; ledger finite-spine length witnesses; exactness through* $\mathsf{ListSpineRep}_A$ *and* ListClassifierSpec*; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.ListUp.ListPublicLength well defined`

Proof. For uniqueness at one endpoint, unpack the two $\mathsf{ListPublicLength}_A$ witnesses. They give finite spines xs and ys, representations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(h, ys)$, and displayed equalities $\ell(xs) = n$ and $\ell(ys) = m$. Apply Theorem 36.74 to the two representations of h under the cons-boundary coherence hypothesis, obtaining $\ell(xs) = \ell(ys)$. Chain the inverse of $\ell(xs) = n$, this equality, and $\ell(ys) = m$ to get $n = m$.

For preservation along the public classifier, unpack the two public length witnesses as represented spines xs for h and ys for k, with $\ell(xs) = n$ and $\ell(ys) = m$. First apply Theorem 36.71 to obtain $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Then apply Theorem 36.64, using cert, the coherence witness, the displayed endpoint representations, and the classifier witness $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$, to obtain $\ell(xs) = \ell(ys)$. Chain this length equality with $\ell(xs) = n$ and $\ell(ys) = m$ to get $n = m$. □

Theorem 36.77 (List public length readback totality and classified endpoints)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. Then the public length readback is total on public list histories and supplies common witnesses for classified endpoints:*

1. *If* $\mathsf{ListHistoryCarrier}_A(h)$*, then* $\exists n.\ \mathsf{ListPublicLength}_A(h, n)$*.*
2. *If* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$*, then* $\exists n.\ \mathsf{ListPublicLength}_A(h, n) \wedge \mathsf{ListPublicLength}_A(k, n)$*.*

The roadmap ten-field fit is: traditional target public finite-list length readback; BEDC source BHist *with* $\mathsf{NameCert}$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; ledger finite-spine presentations; exactness through* $\mathsf{ListSpineRep}_A$ *and the public length definition; classified endpoint readback through the spine bridge; status* $\mathsf{ProofTarget}$*.*

Proof. 1. For totality, apply the forward direction of Theorem 36.65 to the displayed $\mathsf{ListHistoryCarrier}_A(h)$ witness. This gives a finite spine xs with $\mathsf{ListSpineRep}_A(h, xs)$. Choose $n = \ell(xs)$ and repack this representation with the reflexive length equality in Definition 36.75.

2. For classified endpoints, unfold $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$ through Definition 36.57. The classifier witness gives finite spines xs and ys, endpoint presentations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and a finite-spine classifier witness $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$.

   Apply Theorem 36.31 to obtain $\ell(xs) = \ell(ys)$. Choose $n = \ell(xs)$. The representation of h gives $\mathsf{ListPublicLength}_A(h, n)$ by reflexivity. The representation of k gives $\mathsf{ListPublicLength}_A(k, n)$ after rewriting the equality $\ell(xs) = \ell(ys)$ in the opposite direction.

   Repack the two public length witnesses under the chosen value n.

□

Theorem 36.78 (List public-object certificate with public length readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. If* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then the public* $\mathsf{List}^{\uparrow}$ *object has the semantic certificate and public length readback package:*

1. $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\sim_A})$*.*
2. $\forall h.\ \mathsf{ListHistoryCarrier}_A(h) \to \exists n.\ \mathsf{ListPublicLength}_A(h, n)$*.*
3. $\forall h\, n\, m.\ \mathsf{ListPublicLength}_A(h, n) \to \mathsf{ListPublicLength}_A(h, m) \to n = m$*.*
4. $\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \to \exists n.\ \mathsf{ListPublicLength}_A(h, n) \wedge \mathsf{ListPublicLength}_A(k, n)$*.*
5. $\forall h\, k\, n\, m.\ \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListPublicLength}_A(h, n) \to \mathsf{ListPublicLength}_A(k, m) \to n = m$*.*

Proof. The roadmap ten-field fit is: traditional target finite-list public certificate with length readback; BEDC source BHist plus source $\mathsf{NameCert}$; carrier $\mathsf{ListHistoryCarrier}_A$; classifier $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$; stability from cons-boundary coherence; ledger $\mathsf{ListHistoryLedger}_A$ plus $\mathsf{ListPublicLength}_A$; core theorem this package; exactness finite-spine length readback; standard bridge finite spines; status $\mathsf{ProofTarget}$.

1. The certificate clause is exactly Theorem 36.163 applied to cert and the cons-boundary coherence witness.
2. The total public length readback clause is the first component of Theorem 36.77.
3. The uniqueness clause is the first component of Theorem 36.76.
4. The classified endpoint common-witness clause is the second component of Theorem 36.77.
5. The equality of endpoint readbacks under the public classifier is the second component of Theorem 36.76.

□

Theorem 36.79 (List public zero length and nonempty readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, and assume* $\mathsf{ListConsBoundaryCoherent}_{A,\mathsf{Rel}}$*. For every history* h*, the public length ledger is exact at zero and exposes a cons readback at nonzero public endpoints:*

1. $\mathsf{ListPublicLength}_A(h, 0)$ *holds if and only if* $\mathsf{hsame}(h, \mathsf{e})$ *holds.*

2. If $\mathsf{ListHistoryCarrier}_A(h)$ *and* $\neg\mathsf{hsame}(h, \mathsf{e})$ *hold, then*

$$\exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \\ \forall n.\ \mathsf{ListPublicLength}_A(h, n) \to n \neq 0 \end{array} \right).$$

The ten-field fit is: traditional target finite-list zero/nonempty readback; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from cons-boundary coherence; ledger* $\mathsf{ListPublicLength}_A$*; core theorem zero/nonempty exactness; exactness nil iff zero and nonempty gives cons; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. For the zero endpoint, unfold Definition 36.75. A public length witness with value 0 gives a finite spine xs with $\mathsf{ListSpineRep}_A(h, xs)$ and $\ell(xs) = 0$. Case-analyze xs. The cons case has successor length, so it cannot have length 0; the remaining nil case is exactly the nil representation, hence it supplies $\mathsf{hsame}(h, \mathsf{e})$.

Conversely, from $\mathsf{hsame}(h, \mathsf{e})$ choose the nil finite spine. The nil clause of $\mathsf{ListSpineRep}_A$ gives $\mathsf{ListSpineRep}_A(h, \mathsf{nil})$, and $\ell(\mathsf{nil}) = 0$ repacks it as $\mathsf{ListPublicLength}_A(h, 0)$.

For the nonempty readback, first turn the cons-boundary hypothesis into $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$ by Theorem 36.71. Apply Theorem 36.162 to the displayed $\mathsf{ListHistoryCarrier}_A(h)$ witness.

The nil alternative contains $\mathsf{hsame}(h, \mathsf{e})$ and is eliminated by the assumed $\neg\mathsf{hsame}(h, \mathsf{e})$. The cons alternative supplies histories a, t, p with $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$.

It remains to show that no public length witness at this endpoint has value 0. Given $\mathsf{ListPublicLength}_A(h, n)$ and an equality $n = 0$, rewrite the witness to obtain $\mathsf{ListPublicLength}_A(h, 0)$. The zero equivalence just proved gives $\mathsf{hsame}(h, \mathsf{e})$, contradicting the nonempty hypothesis. Therefore $n \neq 0$, and the cons witnesses together with this length exclusion are the required existential data. □

Theorem 36.80 (List classified zero length endpoint readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k*, if* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *holds, then:*

1. *The zero public-length facts agree along the classifier:*

$$\mathsf{ListPublicLength}_A(h, 0) \iff \mathsf{ListPublicLength}_A(k, 0).$$

2. *If either endpoint has public length zero, then both endpoints read back as the empty history:*

$$\mathsf{ListPublicLength}_A(h, 0) \vee \mathsf{ListPublicLength}_A(k, 0) \quad \Longrightarrow \quad \mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}).$$

The ten-field fit is: traditional target finite-list classified zero readback; BEDC source BHist *with* NameCert*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence; ledger* $\mathsf{ListPublicLength}_A$*; core theorem zero public-length transport and empty endpoint exposure; exactness by common public witnesses and zero readback; standard bridge finite spines; status* ProofTarget*.*

Proof. Apply Theorem 36.77 to the displayed $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ witness. It supplies a finite value m together with $\mathsf{ListPublicLength}_A(h, m)$ and $\mathsf{ListPublicLength}_A(k, m)$.

For the forward zero transport, assume $\mathsf{ListPublicLength}_A(h, 0)$. The h-side witness at value m and the given zero witness have the same endpoint, so the uniqueness clause of Theorem 36.76 identifies $m = 0$. Rewriting the k-side common witness gives $\mathsf{ListPublicLength}_A(k, 0)$.

For the reverse zero transport, assume $\mathsf{ListPublicLength}_A(k, 0)$. The k-side witness at value m and the given zero witness have the same endpoint, so the uniqueness clause of Theorem 36.76 identifies $m = 0$. Rewriting the h-side common witness gives $\mathsf{ListPublicLength}_A(h, 0)$.

It remains to read back the endpoints. If the displayed disjunction gives $\mathsf{ListPublicLength}_A(h, 0)$, then the first part of Theorem 36.79 gives $\mathsf{hsame}(h, \mathsf{e})$, and the forward zero transport gives $\mathsf{ListPublicLength}_A(k, 0)$. Applying the same zero readback theorem to the k-side gives $\mathsf{hsame}(k, \mathsf{e})$.

If the displayed disjunction gives $\mathsf{ListPublicLength}_A(k, 0)$, the same argument with the reverse zero transport first gives $\mathsf{ListPublicLength}_A(h, 0)$, and then the zero readback theorem gives $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. □

Theorem 36.81 (List public successor length cons readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For every history* h *and every finite length value* n*, if* $\mathsf{ListPublicLength}_A(h, n+1)$ *holds, then*

$$\exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListPublicLength}_A(t, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array} \right).$$

The ten-field fit is: traditional target finite-list successor readback; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier-independent spine exposure; ledger* $\mathsf{ListPublicLength}_A$*; core theorem successor public length inversion; exactness by finite-spine constructor analysis; standard bridge finite spines; status* ProofTarget*.*

Proof. Unfold Definition 36.75. The public length witness gives a finite spine xs with $\mathsf{ListSpineRep}_A(h, xs)$ and $\ell(xs) = n + 1$.

Case-analyze xs. In the nil case, the displayed length equality would force $0 = n + 1$, which is impossible for a finite length value. Thus the represented spine is a cons spine.

The cons representation exposes histories a, t, p with $A(a)$, a tail spine representation for t, the continuation boundary $\mathsf{Cont}(a, t, p)$, and the endpoint presentation $\mathsf{hsame}(h, \mathsf{E1}(p))$. The tail representation supplies $\mathsf{ListHistoryCarrier}_A(t)$ by the carrier coverage direction for represented spines.

The cons length equality identifies the tail-spine length with n. Repack the tail representation and this predecessor length equality through Definition 36.75, obtaining $\mathsf{ListPublicLength}_A(t, n)$. Together with $A(a)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$, these data are the required existential witnesses. □

Theorem 36.82 (List public successor length constructor exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For every history* h *and every finite length value* n*,*

$$\mathsf{ListPublicLength}_A(h, n{+}1) \iff \exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListPublicLength}_A(t, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array} \right).$$

The ten-field fit is: traditional target finite-list public successor constructor law; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier protected by source* $\mathsf{NameCert}$ *and cons-boundary coherence; ledger* $\mathsf{ListPublicLength}_A$*; core theorem constructor readback in both directions; exactness by finite-spine cons formation; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Theorem 36.81. This exposes the head history, tail history, continuation boundary, predecessor public length, and endpoint presentation required by the right-hand side.

For the reverse direction, assume histories a, t, p with $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{ListPublicLength}_A(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$.

Unfold Definition 36.75 for the tail public length. This gives a finite spine xs with $\mathsf{ListSpineRep}_A(t, xs)$ and $\ell(xs) = n$.

Use the cons constructor for $\mathsf{ListSpineRep}_A$ with the displayed source membership $A(a)$, the tail representation, the continuation witness, and the endpoint presentation. It gives $\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs))$.

The represented cons spine has length $\ell(xs) + 1$, so the displayed tail length identifies it with $n + 1$. Repack this cons representation and length identity through Definition 36.75, obtaining $\mathsf{ListPublicLength}_A(h, n + 1)$. □

Theorem 36.83 (List classified successor length cons readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. If* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *and* $\mathsf{ListPublicLength}_A(h, n+1)$ *hold, then*

$$\exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListPublicLength}_A(t, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \end{array} \right).$$

The ten-field fit is: traditional target finite-list classified successor readback; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger* $\mathsf{ListPublicLength}_A$*; core theorem classified successor public-length inversion; exactness by common public-length witnesses and successor spine analysis; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. Use Theorem 36.77 on the displayed $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ witness. It supplies a finite value m together with $\mathsf{ListPublicLength}_A(h, m)$ and $\mathsf{ListPublicLength}_A(k, m)$.

The given witness $\mathsf{ListPublicLength}_A(h, n+1)$ and the h-side witness at value m have the same endpoint. Apply Theorem 36.76 to identify $m = n+1$, and rewrite the k-side witness as $\mathsf{ListPublicLength}_A(k, n+1)$.

Now apply Theorem 36.81 to the endpoint k with length $n+1$. The resulting histories a, t, p satisfy $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{ListPublicLength}_A(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(k, \mathsf{E1}(p))$, which are exactly the displayed existential data. □

Theorem 36.84 (List classified successor component readback)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. *Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. *For histories h, k and every finite length value n, if* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *and* $\mathsf{ListPublicLength}_A(h, n+1)$ *hold, then*

$$\exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\ \mathsf{ListPublicLength}_A(t, n) \wedge \mathsf{ListPublicLength}_A(u, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ a \sim_A b \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u) \end{array} \right).$$

The ten-field fit is: traditional target finite-list classified successor component readback; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger* $\mathsf{ListPublicLength}_A$ *on both tails; core theorem componentwise successor inversion; exactness by public cons endpoint inversion; standard bridge finite spines; status* $\mathsf{ProofTarget}$.

Proof. First obtain the representation coherence needed by the public endpoint inversion. Apply Theorem 36.71 to the assumed $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ witness.

Next read back the public successor endpoint h. Apply Theorem 36.81 to $\mathsf{ListPublicLength}_A(h, n+1)$, obtaining histories a, t, p with $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{ListPublicLength}_A(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$.

Read back the classified successor endpoint k with Theorem 36.83, using the same classifier witness and public length witness. Rename its histories as b, u, q. This gives $A(b)$, $\mathsf{ListHistoryCarrier}_A(u)$, $\mathsf{ListPublicLength}_A(u, n)$, $\mathsf{Cont}(b, u, q)$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$.

Finally apply Theorem 36.155 to the two displayed cons endpoint presentations and the original $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ witness. The conclusion is $a \sim_A b$ together with $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t,$ Repack these with the two readback packages to obtain the displayed existential witnesses. □

Theorem 36.85 (List classified successor component readback determinacy)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier and let $\sim_A$ be a source classifier. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. *For histories $h, k, a, b, t, u, p, q, a', b', t', u', p', q'$ and every finite length value n, suppose the two six-tuples (a, b, t, u, p, q) and (a', b', t', u', p', q') both satisfy the classified successor component package for the same h, k, n:*

$$\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\ \mathsf{ListPublicLength}_A(t, n) \wedge \mathsf{ListPublicLength}_A(u, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ a \sim_A b \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u) \end{array}$$

and

$$\begin{aligned}&A(a') \wedge A(b') \wedge \mathsf{ListHistoryCarrier}_A(t') \wedge \mathsf{ListHistoryCarrier}_A(u') \wedge \\ &\mathsf{ListPublicLength}_A(t', n) \wedge \mathsf{ListPublicLength}_A(u', n) \wedge \\ &\mathsf{Cont}(a', t', p') \wedge \mathsf{Cont}(b', u', q') \wedge \\ &\mathsf{hsame}(h, \mathsf{E1}(p')) \wedge \mathsf{hsame}(k, \mathsf{E1}(q')) \wedge \\ &a' \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t', u')\end{aligned}$$

hold. Then

$$a \sim_A a' \wedge b \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t') \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u').$$

The ten-field fit is: traditional target finite-list classified successor component readback; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence; ledger same public successor endpoints* h, k *with common predecessor length* n*; core theorem witness-choice determinacy; exactness by public cons endpoint readback; standard bridge finite spines; status* ProofTarget*.*

Proof. Apply Theorem 36.73 to the two h-side cons presentations. The first package supplies $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$; the second supplies the corresponding data for a', t', p'. The conclusion gives $a \sim_A a'$ and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t')$.

Apply the same theorem to the two k-side cons presentations. The first package supplies $A(b)$, $\mathsf{ListHistoryCarrier}_A(u)$, $\mathsf{Cont}(b, u, q)$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$; the second supplies the corresponding data for b', u', q'. The conclusion gives $b \sim_A b'$ and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u')$. Repack the four conclusions in the displayed order. □

Definition 36.86 (List source hsame compatibility)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. Define* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$ *as source-carrier completeness for* hsame*:*

$$\forall a\, b.\ A(a) \to A(b) \to \mathsf{hsame}(a, b) \to a \sim_A b.$$

Lean4 (definition) `BEDC.Derived.ListUp.ListSourceHsameCompatible`

Theorem 36.87 (List source hsame compatibility for hsame)**.** *For every BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$*,*

$$\mathsf{ListSourceHsameCompatible}(A, \mathsf{hsame}).$$

Proof. Unfold ListSourceHsameCompatible. For histories a and b, the two source-carrier witnesses $A(a)$ and $A(b)$ are part of the interface, and the final displayed input is already the required $\mathsf{hsame}(a, b)$ classifier witness. Return that witness. □

Theorem 36.88 (List source-classifier weakening)**.** *Let* $A, B : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, and let* $\sim_A$ *and* $\sim_B$ *be source classifiers. The ten-field fit is: traditional target finite-list source refinement; BEDC source* BHist *with* Cont*; source relation carried by* $\sim_A$ *and* $\sim_B$*; carrier generated list/spine bridge; classifier recursive and spine bridge; stability under source/classifier refinement; ledger unchanged finite spine; core theorem the four displayed implications below; exactness preservation under source/classifier refinement; status* ProofTarget*. If source membership and source classification weaken from* A *to* B*,*

$$(\forall a : \mathsf{BHist}.\ A(a) \to B(a)) \wedge (\forall a\, b : \mathsf{BHist}.\ a \sim_A b \to a \sim_B b),$$

then for all histories h, k and finite spines xs,

$$\begin{aligned}
\mathsf{ListHistoryCarrier}_A(h) &\to \mathsf{ListHistoryCarrier}_B(h),\\
\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,k) &\to \mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{B,\sim_B}(h,k),\\
\mathsf{ListSpineRep}_A(h,xs) &\to \mathsf{ListSpineRep}_B(h,xs),\\
\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k) &\to \mathsf{ListSpineBridgeClassifier}_{B,\sim_B}(h,k).
\end{aligned}$$

Proof. Induct separately on the displayed carrier, recursive classifier, spine representation, and bridge witness. The nil clauses are rebuilt directly. In each cons clause, apply the source-carrier inclusion to the head datum, apply the classifier inclusion to the head classifier datum when present, and reuse the same Cont and endpoint hsame witnesses. For a bridge witness, keep the same finite spines, transport the two spine representations by the representation induction, transport the finite-spine classifier entry by entry using the classifier inclusion, and rebuild the bridge witness with those transported components. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSourceClassifier weakening`

Definition 36.89 (Unary NatAdd relation)**.** *For histories u, v, w : BHist, write*

$$\mathsf{NatAdd}(u,v,w) \iff \mathsf{UnaryCont}(u,v,w).$$

This is the unary-continuation additive relation of Definition 9194.2 and Definition 29.8.

Definition 36.90 (List unary length token)**.** *For finite BEDC-history spines, define the unary length token by recursion:*

$$\begin{aligned}
\mathsf{ListUnaryLength}(\mathsf{nil}) &= \mathsf{e},\\
\mathsf{ListUnaryLength}(\mathsf{cons}(a,xs)) &= \mathsf{E1}(\mathsf{ListUnaryLength}(xs)).
\end{aligned}$$

This is the unary-history counterpart of the finite-spine value $\ell(xs)$ used in Definition 36.75; it records the same constructor count as a $\mathsf{Nat}^{\uparrow}$ token.

Lean4 (definition) `BEDC.Derived.ListUp.ListUnaryLength`

Definition 36.91 (List public unary length token)**.** *For a BEDC history source carrier A : BHist $\to$ Prop, define the public unary length token relation by*

$$\mathsf{ListPublicUnaryLength}_A(h,u) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h,xs) \wedge \mathsf{hsame}(\mathsf{ListUnaryLength}(xs),u).$$

Thus the certificate-level endpoint exposes the same unary $\mathsf{Nat}^{\uparrow}$ token that the finite-spine presentation computes internally.

Theorem 36.92 (List classifier preserves unary length token)**.** *Let $\sim_A$: BHist $\to$ BHist $\to$ Prop be a source classifier and let xs, ys : HistSpine be finite BEDC-history spines. If*

$$\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$$

holds, then

$$\mathsf{hsame}(\mathsf{ListUnaryLength}(xs), \mathsf{ListUnaryLength}(ys)).$$

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec unary length token hsame`

Proof. Induct on the displayed finite-spine classifier witness.

In the nil-nil case, both spines are nil, so both unary length tokens reduce to e. Reflexivity of hsame closes the empty-token comparison.

In the cons-cons case, write the spines as $\mathsf{cons}(a, xs_0)$ and $\mathsf{cons}(b, ys_0)$. The classifier witness contains the source head comparison $a \sim_A b$ and the tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$; the induction hypothesis gives

$$\mathsf{hsame}(\mathsf{ListUnaryLength}(xs_0), \mathsf{ListUnaryLength}(ys_0)).$$

The recursive clause of Definition 36.90 turns the two outer length tokens into one-extensions of those tail tokens. Applying the E1-congruence direction of Definition 20.10 gives the required hsame comparison for the two cons spines. □

Theorem 36.93 (List public unary length classifier transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. Then for all histories* h, k *and all unary length tokens* $u : \mathsf{BHist}$*, a classifier witness* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *transports public unary length readback in both directions:*

1. *if* $\mathsf{ListPublicUnaryLength}_A(h, u)$ *holds, then* $\mathsf{ListPublicUnaryLength}_A(k, u)$ *holds;*
2. *if* $\mathsf{ListPublicUnaryLength}_A(k, u)$ *holds, then* $\mathsf{ListPublicUnaryLength}_A(h, u)$ *holds.*

The ten-field fit is: traditional target finite-list length as a $\mathsf{Nat}^{\uparrow}$ *unary token; BEDC source* BHist *with* NameCert*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from represented spine coherence and source-certificate transitivity; ledger finite spine plus unary token; core theorem public unary length classifier transport; exactness by public readback* $\mathsf{ListPublicUnaryLength}_A$*; standard bridge ordinary finite-list length; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.ListUp.ListPublicUnaryLength classifier transport`

Proof. Fix h, k, u and a bridge classifier $C : \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$. Unpack C into finite spines p and q, endpoint presentations $\mathsf{ListSpineRep}_A(h, p)$ and $\mathsf{ListSpineRep}_A(k, q)$, and a finite-spine classifier $\mathsf{ListClassifierSpec}(\sim_A, p, q)$.

For transport from h to k, unpack $\mathsf{ListPublicUnaryLength}_A(h, u)$ as a finite spine xs, a representation $\mathsf{ListSpineRep}_A(h, xs)$, and $\mathsf{hsame}(\mathsf{ListUnaryLength}(xs), u)$. Coherence at the shared endpoint h compares the two presentations of h, giving $\mathsf{ListClassifierSpec}(\sim_A, xs, p)$.

Compose that finite-spine classifier with the bridge classifier $\mathsf{ListClassifierSpec}(\sim_A, p, q)$, using the transitivity clause of cert at each source head. Apply Theorem 36.92 to obtain

$$\mathsf{hsame}(\mathsf{ListUnaryLength}(xs), \mathsf{ListUnaryLength}(q)).$$

Symmetry and transitivity of hsame, together with the original comparison with u, give $\mathsf{hsame}(\mathsf{ListUnaryLength}(q), u$ Repack q and its representation at k as $\mathsf{ListPublicUnaryLength}_A(k, u)$.

For transport from k to h, unpack $\mathsf{ListPublicUnaryLength}_A(k, u)$ as a finite spine ys, a representation $\mathsf{ListSpineRep}_A(k, ys)$, and $\mathsf{hsame}(\mathsf{ListUnaryLength}(ys), u)$. Coherence at the shared endpoint k compares the bridge spine q with ys, giving $\mathsf{ListClassifierSpec}(\sim_A, q, ys)$.

Compose the bridge classifier $\mathsf{ListClassifierSpec}(\sim_A, p, q)$ with this endpoint-coherence classifier, again using source-certificate transitivity headwise. Theorem 36.92 gives

$$\mathsf{hsame}(\mathsf{ListUnaryLength}(p), \mathsf{ListUnaryLength}(ys)).$$

Transitivity with the original comparison to u gives $\mathsf{hsame}(\mathsf{ListUnaryLength}(p), u)$. Repack p and its representation at h as $\mathsf{ListPublicUnaryLength}_A(h, u)$. □

Theorem 36.94 (List public append unary-length classifier congruence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, h', k, k', r, r', u, v, w : \mathsf{BHist}$*, assume*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, h'), \qquad \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k, k'),$$

$$\mathsf{ListPublicAppend}_A(h, k, r), \qquad \mathsf{ListPublicAppend}_A(h', k', r'),$$

and

$$\mathsf{ListPublicUnaryLength}_A(h, u), \qquad \mathsf{ListPublicUnaryLength}_A(k, v), \qquad \mathsf{ListPublicUnaryLength}_A(r, w)$$

hold. Then

$$\mathsf{ListPublicUnaryLength}_A(h', u) \wedge \mathsf{ListPublicUnaryLength}_A(k', v) \wedge \mathsf{ListPublicUnaryLength}_A(r', w).$$

The ten-field fit is: traditional target finite-list append stability for unary length tokens; BEDC source BHist *through* $\mathsf{ListSpineRep}$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from append congruence and unary-length classifier transport; ledger public unary length tokens; core theorem this composite; exactness via public append and public unary length readback; standard bridge ordinary finite-list append length invariance; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Theorem 36.93 to the classifier witness from h to h' and the public unary length token at h. Its forward transport clause gives $\mathsf{ListPublicUnaryLength}_A(h', u)$.

Apply the same theorem to the classifier witness from k to k' and the public unary length token at k. The forward transport clause gives $\mathsf{ListPublicUnaryLength}_A(k', v)$.

Next apply the second clause of Theorem 36.195 to the two endpoint classifier witnesses and the two public append readbacks. This gives

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(r, r').$$

Apply Theorem 36.93 once more to this output classifier and the public unary length token at r. Its forward transport clause gives $\mathsf{ListPublicUnaryLength}_A(r', w)$, and the three transported witnesses form the displayed conjunction. □

Theorem 36.95 (Public list unary length additivity under append)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For histories* $h, k, r : \mathsf{BHist}$ *and finite spines* $xs, ys : \mathsf{HistSpine}$*, assume*

$$\mathsf{ListSpineRep}_A(h, xs), \qquad \mathsf{ListSpineRep}_A(k, ys), \qquad \mathsf{ListSpineRep}_A(r, xs{+}{+}ys).$$

Set

$$L_x = \mathsf{ListUnaryLength}(xs), \qquad L_y = \mathsf{ListUnaryLength}(ys), \qquad L_{xy} = \mathsf{ListUnaryLength}(xs{+}{+}ys).$$

Then

$$\mathsf{NatAdd}(L_y, L_x, L_{xy})$$

and for every $w : \mathsf{BHist}$*,*

$$\mathsf{NatAdd}(L_x, L_y, w) \Longrightarrow \mathsf{hsame}(L_{xy}, w).$$

Proof. By Definition 36.57, the three representation witnesses say that the public endpoints use the displayed finite spines and that the append endpoint uses the spine append $xs{+}{+}ys$. The proof of the additive length claim depends on the spines alone.

A preliminary induction on a finite spine zs gives $\mathsf{Unary}(\mathsf{ListUnaryLength}(zs))$. The nil case is the base unary generator from Definition 28.1; the cons case applies Definition 28.2 to the induction hypothesis.

Now induct on xs to prove $\mathsf{NatAdd}(L_y, L_x, L_{xy})$. In the nil case, $L_x = \mathsf{e}$ and $L_{xy} = L_y$. The preliminary unary induction gives $\mathsf{Unary}(L_y)$, and the right-unit clause of Definition 21.12 gives $\mathsf{Cont}(L_y, \mathsf{e}, L_y)$. Unfolding Definition 36.89 gives $\mathsf{NatAdd}(L_y, \mathsf{e}, L_y)$.

In the cons case, write $xs = \mathsf{cons}(a, xs_0)$ and let $L_0 = \mathsf{ListUnaryLength}(xs_0)$ and $T = \mathsf{ListUnaryLength}(xs_0{+}{+}ys)$ The induction hypothesis gives $\mathsf{NatAdd}(L_y, L_0, T)$, hence $\mathsf{Cont}(L_y, L_0, T)$ after unfolding Definition 36.89. The one-step rule in Definition 4.10 gives $\mathsf{Cont}(L_y, \mathsf{E1}(L_0), \mathsf{E1}(T))$. The preliminary unary induction and Definition 28.2 supply the two unary input witnesses, so this is exactly $\mathsf{NatAdd}(L_y, L_x, L_{xy})$ by the recursive clauses of Definition 36.90.

Finally assume $\mathsf{NatAdd}(L_x, L_y, w)$. Together with the reversed append-aligned witness just proved, Definition 9194.6 compares the two unary continuation results and gives $\mathsf{hsame}(w, L_{xy})$. Symmetry of hsame gives $\mathsf{hsame}(L_{xy}, w)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListUnaryLength append cont`

Theorem 36.96 (Public list unary length is associative under append)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier. For histories $h_x, h_y, h_z, h_{xy}, h_{(xy)z}, h_{yz}, h_{x(yz)} : \mathsf{BHist}$ and finite spines $xs, ys, zs : \mathsf{HistSpine}$, assume*

$$\mathsf{ListSpineRep}_A(h_x, xs), \quad \mathsf{ListSpineRep}_A(h_y, ys), \quad \mathsf{ListSpineRep}_A(h_z, zs),$$

$$\mathsf{ListSpineRep}_A(h_{xy}, xs{+}{+}ys), \quad \mathsf{ListSpineRep}_A(h_{(xy)z}, (xs{+}{+}ys){+}{+}zs),$$

and

$$\mathsf{ListSpineRep}_A(h_{yz}, ys{+}{+}zs), \quad \mathsf{ListSpineRep}_A(h_{x(yz)}, xs{+}{+}(ys{+}{+}zs)).$$

Set

$$L_x = \mathsf{ListUnaryLength}(xs), \quad L_y = \mathsf{ListUnaryLength}(ys), \quad L_z = \mathsf{ListUnaryLength}(zs),$$

$$L_{xy} = \mathsf{ListUnaryLength}(xs{+}{+}ys), \quad L_{yz} = \mathsf{ListUnaryLength}(ys{+}{+}zs),$$

and

$$L_{(xy)z} = \mathsf{ListUnaryLength}((xs{+}{+}ys){+}{+}zs), \quad L_{x(yz)} = \mathsf{ListUnaryLength}(xs{+}{+}(ys{+}{+}zs)).$$

Then

$$\mathsf{hsame}(L_{(xy)z}, L_{x(yz)}).$$

Proof. Apply Definition 36.95 to the witnesses for xs, ys, and $xs{+}{+}ys$, obtaining $\mathsf{NatAdd}(L_y, L_x, L_{xy})$. Apply the same theorem to the witnesses for $xs{+}{+}ys$, zs, and $(xs{+}{+}ys){+}{+}zs$, obtaining $\mathsf{NatAdd}(L_z, L_{xy}, L_{(xy)z})$.

Apply Definition 36.95 to the witnesses for ys, zs, and $ys{+}{+}zs$, obtaining $\mathsf{NatAdd}(L_z, L_y, L_{yz})$. Apply it once more to the witnesses for xs, $ys{+}{+}zs$, and $xs{+}{+}(ys{+}{+}zs)$, obtaining $\mathsf{NatAdd}(L_{yz}, L_x, L_{x(yz)})$.

Unfolding Definition 36.89, these four witnesses are unary continuations for the two parenthesized length composites. The associativity theorem for unary continuation, Definition 29.6, compares the continuation first through L_{xy} with the continuation first through L_{yz}.

The displayed additivity witnesses identify the left parenthesized endpoint with $L_{(xy)z}$ and the right parenthesized endpoint with $L_{x(yz)}$. Symmetry and transitivity of hsame compose those endpoint comparisons with the associativity comparison, yielding $\mathsf{hsame}(L_{(xy)z}, L_{x(yz)})$. □

Theorem 36.97 (List public length classifier transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. Then for all histories* h, k *and all public length values* n*, a classifier witness* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *transports the chosen public length witness across the classified endpoints:*

1. *if* $\mathsf{ListPublicLength}_A(h, n)$ *holds, then* $\mathsf{ListPublicLength}_A(k, n)$ *holds;*
2. *if* $\mathsf{ListPublicLength}_A(k, n)$ *holds, then* $\mathsf{ListPublicLength}_A(h, n)$ *holds.*

The roadmap ten-field fit is: traditional target classifier-stable finite-list length readback; BEDC source BHist *with* NameCert*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; ledger chosen public length witnesses; exactness through* $\mathsf{ListPublicLength}_A$ *and classified endpoint readback; status* ProofTarget.

Proof. Fix h, k, n and a witness $C : \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$.

For transport from h to k, assume $L_h : \mathsf{ListPublicLength}_A(h, n)$. Apply Theorem 36.77 to C to obtain a public length value m with witnesses $\mathsf{ListPublicLength}_A(h, m)$ and $\mathsf{ListPublicLength}_A(k, m)$.

Apply Theorem 36.76 to the same classifier witness C, the given witness L_h, and the endpoint witness $\mathsf{ListPublicLength}_A(k, m)$. This identifies $n = m$. Rewrite the endpoint witness at k along this equality to obtain $\mathsf{ListPublicLength}_A(k, n)$.

For transport from k to h, assume $L_k : \mathsf{ListPublicLength}_A(k, n)$. Apply Theorem 36.77 to C again to obtain a public length value m with witnesses $\mathsf{ListPublicLength}_A(h, m)$ and $\mathsf{ListPublicLength}_A(k, m)$.

Apply Theorem 36.76 to the same classifier witness C, the endpoint witness $\mathsf{ListPublicLength}_A(h, m)$, and the given witness L_k. This identifies $m = n$. Rewrite the endpoint witness at h along this equality to obtain $\mathsf{ListPublicLength}_A(h, n)$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength classifier transport`
Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier public length transport`
Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength bridge transport pair`

Theorem 36.98 (List public length classifier unary token transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. Then for all histories* h, k *and all public length values* n*, a classifier witness* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *transports each finite spine presentation together with its derived unary length token:*

1. *if* $\mathsf{ListSpineRep}_A(h, xs)$ *and* $\ell(xs) = n$*, then there exists* $ys : \mathsf{HistSpine}$ *such that*

$$\mathsf{ListSpineRep}_A(k, ys) \wedge \ell(ys) = n \wedge \mathsf{hsame}(\mathsf{ListUnaryLength}(xs), \mathsf{ListUnaryLength}(ys)).$$

2. *if* $\mathsf{ListSpineRep}_A(k, ys)$ *and* $\ell(ys) = n$*, then there exists* $xs : \mathsf{HistSpine}$ *such that*

$$\mathsf{ListSpineRep}_A(h, xs) \wedge \ell(xs) = n \wedge \mathsf{hsame}(\mathsf{ListUnaryLength}(ys), \mathsf{ListUnaryLength}(xs)).$$

The roadmap ten-field fit is: traditional target classifier-stable finite-list length readback with internal unary evidence; BEDC source BHist *with* NameCert*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; ledger the public finite length value and its BEDC unary token; exactness through* $\mathsf{ListPublicLength}_A$*,* $\mathsf{ListSpineRep}_A$*, and* ListUnaryLength*; status* ProofTarget.

Proof. Fix a classifier witness $C : \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$. Theorem 36.77 applied to C supplies a common public length value m for the classified endpoints.

Unpack Definition 36.75 for those two public witnesses, giving classifier-selected finite spines x_0 and y_0 with representations of h and k and with finite values $\ell(x_0) = m$ and $\ell(y_0) = m$. Unfolding C through the list-spine representation definition selects the same endpoints with a finite-spine classifier witness $\mathsf{ListClassifierSpec}(\sim_A, x_0, y_0)$.

For transport from h to k, assume the displayed public presentation of h is witnessed by xs and $\ell(xs) = n$. Repack it through Definition 36.75 as $\mathsf{ListPublicLength}_A(h, n)$. The k-side selected spine gives $\mathsf{ListPublicLength}_A(k, m)$, and Theorem 36.76 aligns the finite values $n = m$. Rewriting $\ell(y_0) = m$ along this equality gives $\ell(y_0) = n$.

It remains to compare unary tokens. First turn the cons-boundary coherence hypothesis into $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ by Theorem 36.71. Applying that coherence datum to the two h-presentations, xs and x_0, gives $\mathsf{ListClassifierSpec}(\sim_A, xs, x_0)$.

Apply Theorem 36.92 to this comparison to obtain $\mathsf{hsame}(\mathsf{ListUnaryLength}(xs), \mathsf{ListUnaryLength}(x_0))$. Apply the same theorem to the selected endpoint classifier to obtain $\mathsf{hsame}(\mathsf{ListUnaryLength}(x_0), \mathsf{ListUnaryLength}(y_0$

By transitivity of hsame, these combine to $\mathsf{hsame}(\mathsf{ListUnaryLength}(xs), \mathsf{ListUnaryLength}(y_0))$. Taking $ys = y_0$ gives the first existential.

For transport from k to h, use the symmetric argument. The displayed presentation of k repacks as $\mathsf{ListPublicLength}_A(k, n)$; the h-side selected spine gives $\mathsf{ListPublicLength}_A(h, m)$; and Theorem 36.76 aligns $n = m$, hence $\ell(x_0) = n$.

Coherence compares the displayed k-spine with the selected y_0 spine, so Theorem 36.92 gives their unary token comparison.

The selected endpoint classifier gives the reverse comparison from $\mathsf{ListUnaryLength}(y_0)$ to $\mathsf{ListUnaryLength}(x_0)$ by symmetry of hsame after applying Theorem 36.92. Transitivity of hsame combines the two comparisons, and choosing $xs = x_0$ gives the second existential. □

Theorem 36.99 (List classified successor component exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k *and every finite length value* n*,*

$$\left(\begin{array}{l}\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k)\wedge\\ \mathsf{ListPublicLength}_A(h,n+1)\wedge\mathsf{ListPublicLength}_A(k,n+1)\end{array}\right) \iff \exists a\,b\,t\,u\,p\,q : \mathsf{BHist}.\left(\begin{array}{l}A(a)\wedge A(b)\wedge\mathsf{ListHi}\\ \mathsf{ListPublicLength}_A(t,\\ \mathsf{Cont}(a,t,p)\wedge\mathsf{Cont}(\\ \mathsf{hsame}(h,\mathsf{E1}(p))\wedge\mathsf{hs}\\ a\sim_A b\wedge\mathsf{ListSpineH}\end{array}\right.$$

The ten-field fit is: traditional target finite-list classified successor component coverage; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger common successor public lengths on both endpoints; core theorem successor component exactness; exactness by public successor length construction and public-object constructor audit; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Theorem 36.84 to the displayed classifier witness and the public successor length witness for h. It returns histories a, b, t, u, p, q with the two source heads, the two tail carriers, the predecessor public lengths, the two continuation witnesses, the two successor endpoint presentations, the head classifier witness, and the tail classifier witness. These are exactly the six-component package on the right-hand side.

For the reverse direction, assume the displayed six-component package. Apply the reverse direction of Theorem 36.82 to the h-side data $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{ListPublicLength}_A(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$. This gives $\mathsf{ListPublicLength}_A(h, n + 1)$.

Apply the same public successor constructor exactness theorem to the k-side data $A(b)$, $\mathsf{ListHistoryCarrier}_A(u)$, $\mathsf{ListPublicLength}_A(u, n)$, $\mathsf{Cont}(b, u, q)$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$. This gives $\mathsf{ListPublicLength}_A(k, n+1)$.

It remains to recover the classified endpoint relation. Use the cons branch of Theorem 36.165, instantiated with the same A, $\sim_A$, source certificate, and cons-boundary coherence witness. The displayed heads, tails, continuations, endpoint presentations, $a \sim_A b$, and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)$ form exactly its cons-constructor package, so the reverse direction of the audit yields $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h$ Combining this classifier witness with the two public successor length witnesses gives the left-hand side. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier public successor component exactness`
Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier successor component exactness iff`

Theorem 36.100 (List classified successor one-sided component exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k *and every finite length value* n*,*

$$\Big(\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \wedge \mathsf{ListPublicLength}_A(h, n+1)\Big) \iff \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{L} \\ \mathsf{ListPublicLengt}\mathsf{h} \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{C} \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \\ a \sim_A b \wedge \mathsf{ListSpi} \end{array}\right.$$

The ten-field fit is: traditional target finite-list classified successor one-sided component coverage; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger one endpoint successor public length and two predecessor tail lengths; core theorem one-sided successor component exactness; exactness by classified component readback, public successor length construction, and public-object constructor audit; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Theorem 36.84 to the displayed classifier witness and the public successor length witness for h. The resulting histories a, b, t, u, p, q carry the two source heads, the two tail carriers, the predecessor public lengths at n, the two continuation ledgers, the two endpoint presentations through $\mathsf{E1}$, the head classifier witness, and the tail classifier witness. These are exactly the fields displayed on the right-hand side.

For the reverse direction, assume the displayed six-component package. Apply the reverse direction of Theorem 36.82 to the h-side data $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{ListPublicLength}_A(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$. This gives $\mathsf{ListPublicLength}_A(h, n+1)$.

It remains to recover the classified endpoint relation. Use the cons branch of Theorem 36.165, instantiated with the same A, $\sim_A$, source certificate, and cons-boundary coherence witness. The displayed heads, tails, continuations, endpoint presentations, $a \sim_A b$, and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)$ form exactly its cons-constructor package, so the reverse direction of the audit yields $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h$ Combining this classifier witness with the public successor length witness for h gives the left-hand side. □

Theorem 36.101 (List classified successor one-sided length transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k *and every finite length value* n*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \wedge \mathsf{ListPublicLength}_A(h, n+1),$$

then $\mathsf{ListPublicLength}_A(k, n+1)$*. The ten-field fit is: traditional target finite-list classified successor one-sided public-length transport; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger one endpoint successor public length and the classified endpoint successor ledger; core theorem classified one-sided public-length transport; exactness by component readback and public successor length construction; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. Apply the forward direction of Theorem 36.100 to the given classifier witness and the public successor length witness for h. This produces histories a, b, t, u, p, q with the displayed six-history component package.

Project from that package the k-side data $A(b)$, $\mathsf{ListHistoryCarrier}_A(u)$, $\mathsf{ListPublicLength}_A(u, n)$, $\mathsf{Cont}(b, u, q)$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$.

Apply the reverse direction of Theorem 36.82 to these k-side data. The constructor exactness theorem yields $\mathsf{ListPublicLength}_A(k, n + 1)$. □

Theorem 36.102 (List classified successor component unique existence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k *and every finite length value* n*, suppose*

$$\begin{aligned}&\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)\wedge\\&\mathsf{ListPublicLength}_A(h, n + 1) \wedge \mathsf{ListPublicLength}_A(k, n + 1).\end{aligned}$$

Then there exist histories a, b, t, u, p, q *such that*

$$\begin{aligned}&A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u)\wedge\\&\mathsf{ListPublicLength}_A(t, n) \wedge \mathsf{ListPublicLength}_A(u, n)\wedge\\&\mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q)\wedge\\&\mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q))\wedge\\&a \sim_A b \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)\end{aligned}$$

and, for every other six-tuple a', b', t', u', p', q' *satisfying*

$$\begin{aligned}&A(a') \wedge A(b') \wedge \mathsf{ListHistoryCarrier}_A(t') \wedge \mathsf{ListHistoryCarrier}_A(u')\wedge\\&\mathsf{ListPublicLength}_A(t', n) \wedge \mathsf{ListPublicLength}_A(u', n)\wedge\\&\mathsf{Cont}(a', t', p') \wedge \mathsf{Cont}(b', u', q')\wedge\\&\mathsf{hsame}(h, \mathsf{E1}(p')) \wedge \mathsf{hsame}(k, \mathsf{E1}(q'))\wedge\\&a' \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t', u'),\end{aligned}$$

one has

$$a \sim_A a' \wedge b \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t') \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u').$$

The ten-field fit is: traditional target finite-list classified successor unique existence; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger common successor length; core theorem unique existence for the classified successor component; exactness existence plus witness-choice determinacy; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. Apply the forward direction of Theorem 36.99 to the displayed classifier witness and the two public successor length witnesses. This supplies a six-tuple a, b, t, u, p, q satisfying the classified successor component package.

It remains to show that this witness is independent of the chosen component readback. Let a', b', t', u', p', q' be any other six-tuple satisfying the same component package for h, k, n. Apply Theorem 36.85 to the displayed package for a, b, t, u, p, q and the displayed package for a', b', t', u', p', q'.

The determinacy theorem gives $a \sim_A a'$, $b \sim_A b'$, $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t')$, and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u')$ in the required order. Repack the witness package together with this universal comparison clause. □

Theorem 36.103 (List classified successor one-sided component unique existence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* h, k *and every finite length value* n*, suppose*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \wedge \mathsf{ListPublicLength}_A(h, n+1).$$

Then there exist histories a, b, t, u, p, q *such that*

$$\begin{aligned}
&A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\
&\mathsf{ListPublicLength}_A(t, n) \wedge \mathsf{ListPublicLength}_A(u, n) \wedge \\
&\mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \\
&\mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\
&a \sim_A b \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)
\end{aligned}$$

and, for every other six-tuple a', b', t', u', p', q' *satisfying*

$$\begin{aligned}
&A(a') \wedge A(b') \wedge \mathsf{ListHistoryCarrier}_A(t') \wedge \mathsf{ListHistoryCarrier}_A(u') \wedge \\
&\mathsf{ListPublicLength}_A(t', n) \wedge \mathsf{ListPublicLength}_A(u', n) \wedge \\
&\mathsf{Cont}(a', t', p') \wedge \mathsf{Cont}(b', u', q') \wedge \\
&\mathsf{hsame}(h, \mathsf{E1}(p')) \wedge \mathsf{hsame}(k, \mathsf{E1}(q')) \wedge \\
&a' \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t', u'),
\end{aligned}$$

one has

$$a \sim_A a' \wedge b \sim_A b' \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t') \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u').$$

The ten-field fit is: traditional target finite-list one-sided successor component unique existence; BEDC source BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from cons-boundary coherence and source* $\mathsf{NameCert}$*; ledger one public successor endpoint; core theorem one-sided unique existence; exactness by one-sided component exactness plus witness-choice determinacy; standard bridge finite spines; status* $\mathsf{ProofTarget}$*.*

Proof. Apply the forward direction of Theorem 36.100 to the displayed classifier witness and the public successor length witness for h. This supplies histories a, b, t, u, p, q satisfying the six-component package for the same h, k, n.

It remains to compare this witness with any other readback package. Let a', b', t', u', p', q' be another six-tuple satisfying the displayed package for h, k, n. Apply Theorem 36.85 to the package for a, b, t, u, p, q and the package for a', b', t', u', p', q'.

The determinacy theorem gives $a \sim_A a'$, $b \sim_A b'$, $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, t')$, and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(u, u')$ in the displayed order. Repack the existential witness together with this universal comparison clause. □

Definition 36.104 (Pair frame)**.** *For histories* $u, t : \mathsf{BHist}$*, define* $\mathsf{PairFrame}(u, t)$ *by structural recursion on the first BEDC history:*

$$\begin{aligned} \mathsf{PairFrame}(\mathsf{e}, t) &= \mathsf{E0}(t), \\ \mathsf{PairFrame}(\mathsf{E0}(a), t) &= \mathsf{E1}(\mathsf{E0}(\mathsf{PairFrame}(a, t))), \\ \mathsf{PairFrame}(\mathsf{E1}(a), t) &= \mathsf{E1}(\mathsf{E1}(\mathsf{PairFrame}(a, t))). \end{aligned}$$

The outer $\mathsf{E1}$ *marks a nonempty first component, while the inner* $\mathsf{E0}/\mathsf{E1}$ *tag records the constructor of that first component.*

Lean4 (definition) `BEDC.Derived.ListUp.PairFrame`

Lemma 36.105 (Pair frame injectivity)**.** *For histories* $u, u', t, t' : \mathsf{BHist}$*, if*

$$\mathsf{hsame}(\mathsf{PairFrame}(u, t), \mathsf{PairFrame}(u', t')),$$

then $\mathsf{hsame}(u, u') \wedge \mathsf{hsame}(t, t')$*.*

Proof. Proceed by structural induction and case analysis on u and u'. The mixed $\mathsf{e}/\mathsf{E0}$ and $\mathsf{e}/\mathsf{E1}$ cases contradict BEDC constructor no-confusion because $\mathsf{PairFrame}(\mathsf{e}, t)$ begins with $\mathsf{E0}$, while the nonempty cases begin with $\mathsf{E1}$. The mixed $\mathsf{E0}/\mathsf{E1}$ cases contradict no-confusion after inverting the outer $\mathsf{E1}$ and comparing the inner tags. In the matching e/e case, constructor inversion gives $\mathsf{hsame}(t, t')$. In the matching $\mathsf{E0}/\mathsf{E0}$ and $\mathsf{E1}/\mathsf{E1}$ cases, invert the outer $\mathsf{E1}$, then the matching inner tag, and apply the induction hypothesis to the recursive pair-frame equality. Combining the exposed recursive head equality with the constructor equality gives $\mathsf{hsame}(u, u')$, and the induction hypothesis also gives $\mathsf{hsame}(t, t')$. □

Lean4 ✓ `BEDC.Derived.ListUp.PairFrame hsame injective`

Theorem 36.106 (Pair frame empty head iff)**.** *For histories* $u, t : \mathsf{BHist}$*, the pair frame has the empty-head boundary exactly when the head itself is empty:*

$$\mathsf{hsame}(\mathsf{PairFrame}(u, t), \mathsf{E0}(t)) \iff \mathsf{hsame}(u, \mathsf{e}).$$

Proof. Case-split the head u. In the empty case the pair-frame definition is $\mathsf{E0}(t)$, so the displayed classifier is reflexive. In the zero-headed and one-headed cases the pair frame begins with $\mathsf{E1}$, contradicting $\mathsf{E1}/\mathsf{E0}$ no-confusion. Conversely, a classifier $\mathsf{hsame}(u, \mathsf{e})$ substitutes the empty head into the pair-frame definition. □

Lean4 ✓ `BEDC.Derived.ListUp.PairFrame empty head iff`

Definition 36.107 (Framed list endpoint)**.** *For finite BEDC-history spines, define the framed endpoint by recursion on the spine:*

$$\begin{aligned} \mathsf{FramedListEndpoint}(\mathsf{nil}) &= \mathsf{E0}(\mathsf{e}), \\ \mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs)) &= \mathsf{E1}(\mathsf{PairFrame}(a, \mathsf{FramedListEndpoint}(xs))). \end{aligned}$$

Lean4 (definition) `BEDC.Derived.ListUp.FramedListEndpoint`

Theorem 36.108 (Framed list endpoint no-confusion and cons inversion)**.** *The ten-field fit is: traditional target finite-list endpoint readback; BEDC source* BHist *with* hsame*; carrier framed finite spines; classifier endpoint* hsame*; stability obligations nil/cons separation and cons inversion; ledger*

constructor frame; core theorem this block; exactness framed endpoint exposes list shape; standard bridge finite spines; status ProofTarget. *For all histories* $a, b : \mathsf{BHist}$ *and finite spines* xs, ys,

$$\neg\mathsf{hsame}(\mathsf{FramedListEndpoint}(\mathsf{nil}), \mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs))),$$

$$\neg\mathsf{hsame}(\mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs)), \mathsf{FramedListEndpoint}(\mathsf{nil})),$$

and

$$\mathsf{hsame}(\mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs)), \mathsf{FramedListEndpoint}(\mathsf{cons}(b, ys)))$$

implies both $\mathsf{hsame}(a, b)$ *and* $\mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoint}(ys))$.

Proof. The nil endpoint has outer constructor E0, while every cons endpoint has outer constructor E1, so the two mixed cases contradict BEDC history no-confusion in the E0/E1 and E1/E0 directions. For the cons-cons clause, invert the outer E1 equality and obtain equality of the two displayed pair frames. Lemma 36.105 applies: its proof is an induction on the first frame argument, with constructor no-confusion separating the empty and tagged nonempty cases and with recursive inversion exposing the first and second frame arguments. The two projections of that lemma give $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoint}(ys))$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListEndpoint no confusion cons inversion`

Lemma 36.109 (Pair frame congruence)**.** *For histories* $u, u', t, t' : \mathsf{BHist}$, *if* $\mathsf{hsame}(u, u')$ *and* $\mathsf{hsame}(t, t')$, *then*

$$\mathsf{hsame}(\mathsf{PairFrame}(u, t), \mathsf{PairFrame}(u', t')).$$

Proof. Proceed by induction on the displayed witness $\mathsf{hsame}(u, u')$. In the empty case, the goal is the congruence of E0 applied to $\mathsf{hsame}(t, t')$. In the E0 case, apply the induction hypothesis to the inner pair frame and then apply congruence for the two visible constructors E0 and E1. The E1 case is identical except that the inner tag is E1. Thus the frame recursion is stable in the first argument, and the terminal empty branch carries the second-argument hsame witness. □

Lean4 ✓ `BEDC.Derived.ListUp.PairFrame congruence`

Theorem 36.110 (Framed list endpoint classifier exactness)**.** *For finite BEDC-history spines* xs, ys,

$$\mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoint}(ys)) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys).$$

Proof. For the left-to-right direction, argue by induction on the first spine and case analysis on the second. The nil-nil case gives the nil constructor of ListClassifierSpec. The mixed nil-cons and cons-nil cases contradict Theorem 36.108. In the cons-cons case, the same theorem exposes hsame between the heads and hsame between the framed tail endpoints; apply the induction hypothesis to the tail equality and then use the cons constructor of $\mathsf{ListClassifierSpec}(\mathsf{hsame}, -, -)$.

For the right-to-left direction, induct on the displayed $\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys)$ witness. The nil branch is reflexivity of the framed nil endpoint. In the cons branch, the head witness and the induction hypothesis give congruent arguments for PairFrame by Lemma 36.109; one final E1 congruence yields equality of the framed cons endpoints. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListEndpoint classifier exactness`

Theorem 36.111 (Framed list endpoint length preservation)**.** *The ten-field fit is: traditional target finite-list framed endpoint readback; BEDC source* BHist*; carrier finite spines; classifier endpoint* hsame*; stability endpoint exactness; ledger visible framed endpoint; core theorem length preservation; exactness via* ListClassifierSpec*; standard bridge finite spines; status* ProofTarget*. For finite BEDC-history spines xs, ys, if*

$$\mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoint}(ys))$$

holds, then

$$\ell(xs) = \ell(ys).$$

Proof. Apply Theorem 36.110 from left to right to convert the displayed endpoint equality into $\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys)$. Then apply Theorem 36.31 to obtain equality of the finite spine lengths. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListEndpoint length preservation`

Definition 36.112 (Framed list-spine representation)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier. Define $\mathsf{FramedListSpineRep}_A(h, xs)$ to mean that every entry of the finite spine xs satisfies A and that*

$$\mathsf{hsame}(h, \mathsf{FramedListEndpoint}(xs)).$$

Lean4 (definition) `BEDC.Derived.ListUp.FramedListSpineRep`

Theorem 36.113 (Framed list-spine representation hsame transport)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier. If $\mathsf{FramedListSpineRep}_A(h, xs)$ holds and $\mathsf{hsame}(h, h')$ holds, then $\mathsf{FramedListSpineRep}_A(h', xs)$ holds.*

Proof. The entrywise source predicate over xs is unchanged. The endpoint witness for h' is obtained by composing $\mathsf{hsame}(h', h)$, the symmetry of the displayed $\mathsf{hsame}(h, h')$, with the stored witness $\mathsf{hsame}(h, \mathsf{FramedListEndpoint}(xs))$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSpineRep hsame transport`

Theorem 36.114 (Framed list-spine cons endpoint injectivity)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier. If $\mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs))$ and $\mathsf{FramedListSpineRep}_A(k, \mathsf{cons}(b, ys))$ hold, and $\mathsf{hsame}(h, k)$ holds, then the two framed cons endpoints are* hsame*, and their head payloads and tail endpoints are separately* hsame*:*

$$\mathsf{hsame}(\mathsf{E1}(\mathsf{PairFrame}(a, \mathsf{FramedListEndpoint}(xs))), \mathsf{E1}(\mathsf{PairFrame}(b, \mathsf{FramedListEndpoint}(ys))))$$

and

$$\mathsf{hsame}(a, b) \wedge \mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoint}(ys)).$$

Proof. Compose the stored endpoint witness for the first representation, the displayed $\mathsf{hsame}(h, k)$ witness, and the stored endpoint witness for the second representation. This gives hsame between the two cons endpoints. Applying Definition 36.108 to that cons-cons endpoint equality exposes both the head equality $\mathsf{hsame}(a, b)$ and the tail endpoint equality. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSpineRep cons endpoint injective`

Theorem 36.115 (Framed list-spine nil-cons no-confusion)**.** *Let A : BHist $\to$ Prop be a BEDC history source carrier. No single history can be represented both by the framed nil spine and by a framed cons spine:*

$$\mathsf{FramedListSpineRep}_A(h, \mathsf{nil}) \to \mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs)) \to \mathsf{False}.$$

Proof. Unpack the two framed representations and compose their endpoint classifiers through symmetry and transitivity of hsame. This gives $\mathsf{hsame}(\mathsf{FramedListEndpoint}(\mathsf{nil}), \mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs)))$, which contradicts the framed endpoint nil-cons no-confusion theorem. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSpineRep nil cons no confusion`

Theorem 36.116 (Framed list-spine coherence)**.** *Let A : BHist $\to$ Prop be a BEDC history source carrier and let $\sim_A$ be a source classifier. If* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$ *holds, then any two framed presentations of the same history classify their finite spines:*

$$\forall h\, xs\, ys.\ \mathsf{FramedListSpineRep}_A(h, xs) \to \mathsf{FramedListSpineRep}_A(h, ys) \to \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Proof. Lemma 36.105 supplies the required injectivity of PairFrame. The framed list endpoint separates nil and cons by the outer E0/E1 tag, so mixed nil/cons presentations contradict BEDC history no-confusion. In the nil-nil case the finite-spine classifier is the nil constructor. In the cons-cons case, injectivity of the outer E1 endpoint and then injectivity of PairFrame expose hsame between the two heads and hsame between the framed tails. The representation hypotheses give A for both heads, so $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$ supplies the head classifier. Apply the induction hypothesis to the two framed tail presentations, and combine the head classifier with the tail classifier by the cons constructor of ListClassifierSpec. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSpineRep coherence`

Theorem 36.117 (Framed list-spine length determinism)**.** *Let A : BHist $\to$ Prop be a BEDC history source carrier and let $\sim_A$ be a source classifier. If* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*,* $\mathsf{FramedListSpineRep}_A(h, xs)$*, and* $\mathsf{FramedListSpineRep}_A(h, ys)$ *hold, then*

$$\ell(xs) = \ell(ys).$$

The roadmap ten-field fit is: traditional target finite-list framed readback; BEDC source BHist*; carrier* $\mathsf{FramedListHistoryCarrier}_A$*; classifier* ListClassifierSpec*; stability framed endpoint coherence; ledger finite spine; core theorem framed length determinism; exactness through framed spine representation; standard bridge finite spines; status* ProofTarget*.*

Proof. Apply Theorem 36.116 to the compatibility witness and the two displayed framed presentations of h. This gives $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Then apply Theorem 36.31 to read back equality of the two finite spine lengths. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSpineRep length determinism`

Theorem 36.118 (Framed list bridge classifier preserves represented length)**.** *Let A : BHist $\to$ Prop be a BEDC history source carrier and let $\sim_A$ be a source classifier. For histories h, k : BHist and finite spines xs_0, ys_0, if* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*,* $\mathsf{FramedListSpineRep}_A(h, xs_0)$*,* FramedListSpineRep *and* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *hold, then*

$$\ell(xs_0) = \ell(ys_0).$$

The roadmap ten-field fit is: traditional target finite-list framed readback along a bridge classifier; BEDC source BHist*; carrier* $\mathsf{FramedListHistoryCarrier}_A$*; classifier* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}$*; stability framed endpoint coherence and source classifier length soundness; ledger finite spine representatives at both endpoints; core theorem bridge length preservation; exactness by framed spine representation; standard bridge finite spines; status* ProofTarget.

Proof. Unpack the framed bridge classifier witness to finite spines xs, ys with $\mathsf{FramedListSpineRep}_A(h, xs)$, $\mathsf{FramedListSpineRep}_A(k, ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Apply Theorem 36.116 at h to classify xs_0 with xs, and at k to classify ys with ys_0. Apply Theorem 36.31 to the three finite-spine classifier witnesses $xs_0 \sim_A xs$, $xs \sim_A ys$, and $ys \sim_A ys_0$, then chain the three finite-length equalities. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier represented length`

Definition 36.119 (Framed list public length readback)**.** *For a BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{FramedListPublicLength}_A(h, n)$ *by*

$$\mathsf{FramedListPublicLength}_A(h, n) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FramedListSpineRep}_A(h, xs) \wedge \ell(xs) = n.$$

Here ℓ *is the finite-spine presentation length read from the framed spine representation. It is not a separate* $\mathsf{Nat}^{\uparrow}$ *operation and is not an additional field of the framed semantic certificate.*

Lean4 (definition) `BEDC.Derived.ListUp.FramedListPublicLength`

Theorem 36.120 (Framed list public length readback is well-defined)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. Assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. Then the framed public length readback has the following two properties.*

1. *If* $\mathsf{FramedListPublicLength}_A(h, n)$ *and* $\mathsf{FramedListPublicLength}_A(h, m)$*, then* $n = m$*.*
2. *If* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *and* $\mathsf{FramedListPublicLength}_A(h, n)$*, then* $\mathsf{FramedListPublicLength}_A(k, n)$*.*

The roadmap ten-field fit is: traditional target finite-list framed length readback; BEDC source BHist*; carrier* $\mathsf{FramedListHistoryCarrier}_A$*; classifier* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}$*; stability obligation framed endpoint coherence; ledger finite-spine length witness; core theorem well-definedness and classifier preservation; exactness via* FramedListSpineRep*; standard bridge finite spines; status* ProofTarget.

Proof. For uniqueness at one endpoint, unpack the two $\mathsf{FramedListPublicLength}_A$ witnesses. They give finite spines xs and ys, representations $\mathsf{FramedListSpineRep}_A(h, xs)$ and $\mathsf{FramedListSpineRep}_A(h, ys)$, and displayed equalities $\ell(xs) = n$ and $\ell(ys) = m$. Apply Theorem 36.117 to the two framed representations of h, obtaining $\ell(xs) = \ell(ys)$. Chain the inverse of $\ell(xs) = n$, this equality, and $\ell(ys) = m$ to get $n = m$.

For preservation along the framed bridge classifier, unpack the public length witness for h as a finite spine xs_0 with $\mathsf{FramedListSpineRep}_A(h, xs_0)$ and $\ell(xs_0) = n$. Unpack the bridge classifier into finite spines xs, ys with $\mathsf{FramedListSpineRep}_A(h, xs)$, $\mathsf{FramedListSpineRep}_A(k, ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$.

Use Theorem 36.118 with the displayed representative xs_0 at h, the bridge representative ys at k, and the bridge classifier witness to obtain $\ell(xs_0) = \ell(ys)$. Chain this length equality with $\ell(xs_0) = n$, then choose ys as the framed public length witness for k. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength well defined`

Theorem 36.121 (Framed list bridge classifier has common public length)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. If* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$, $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$, *and* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *hold, then* h *and* k *are framed list carrier histories and share a framed public length:*

$$\mathsf{FramedListHistoryCarrier}_A(h) \wedge \mathsf{FramedListHistoryCarrier}_A(k) \wedge \exists n.\ \mathsf{FramedListPublicLength}_A(h, n) \wedge \mathsf{FramedListPublicL}$$

Proof. Use the equivalence-field theorem for the framed bridge classifier to obtain the two carrier endpoints. Unpack the bridge witness into displayed finite spines xs, ys and a finite-spine classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. The list classifier length theorem gives $\ell(xs) = \ell(ys)$, so $\ell(xs)$ is a shared public length for the two displayed framed spine representatives. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier public length total`

Theorem 36.122 (Framed list public length constructor readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For every history* h *and every finite length value* n*, the framed public length readback exposes the finite-spine constructors:*

1. $\mathsf{FramedListPublicLength}_A(h, 0)$ *holds if and only if* $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$ *holds.*
2. *If* $\mathsf{FramedListPublicLength}_A(h, n+1)$ *holds, then*

$$\exists a : \mathsf{BHist},\ xs : \mathsf{HistSpine}.\ A(a) \wedge \mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs)) \wedge \mathsf{FramedListPublicLength}_A(\mathsf{FramedListEndp}$$

The roadmap ten-field fit is: traditional target framed finite-list constructor readback; BEDC source BHist*; carrier* $\mathsf{FramedListHistoryCarrier}_A$*; classifier-independent spine exposure; ledger* $\mathsf{FramedListPublicLength}_A$*; core theorem nil and successor public-length inversion; exactness by finite framed-spine constructor analysis; standard bridge finite spines; status* $\mathsf{ProofTarget}$.

Proof. For the zero forward direction, unfold Definition 36.119. A public length witness with value 0 supplies a finite spine xs with $\mathsf{FramedListSpineRep}_A(h, xs)$ and $\ell(xs) = 0$. Case-analyze xs. The cons case has successor length and cannot have value 0, while the nil case is the required framed nil representation.

For the zero reverse direction, choose the nil finite spine in Definition 36.119. The displayed $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$ witness and $\ell(\mathsf{nil}) = 0$ repack as $\mathsf{FramedListPublicLength}_A(h, 0)$.

For the successor clause, unfold the same definition. A witness for $\mathsf{FramedListPublicLength}_A(h, n+1)$ gives a finite spine xs_0 with $\mathsf{FramedListSpineRep}_A(h, xs_0)$ and $\ell(xs_0) = n+1$. The nil case would force $\ell(\mathsf{nil}) = n+1$, impossible for finite lengths, so the exposed spine has the form $\mathsf{cons}(a, xs)$.

The cons representation already supplies $A(a)$ and $\mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs))$. Its entry predicate restricts to the tail entries of xs, and reflexivity gives $\mathsf{hsame}(\mathsf{FramedListEndpoint}(xs), \mathsf{FramedListEndpoi}$ Together these build $\mathsf{FramedListSpineRep}_A(\mathsf{FramedListEndpoint}(xs), xs)$.

The displayed successor length equality identifies $\ell(xs)$ with n. Repacking the tail representation and this predecessor length through Definition 36.119 gives $\mathsf{FramedListPublicLength}_A(\mathsf{FramedListEndpoint}(xs), n$ so a and xs are the required constructor witnesses. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength constructor endpoint readback`

Theorem 36.123 (Framed list public length successor empty absurd)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. If a history* h *has successor framed public length, then* h *cannot also be the empty framed endpoint:*

$$\mathsf{FramedListPublicLength}_A(h, n+1) \Longrightarrow \mathsf{hsame}(h, \mathsf{FramedListEndpoint}(\mathsf{nil})) \Longrightarrow \mathsf{False}.$$

Proof. Apply constructor readback to the successor public-length witness. It exposes a head a and tail spine xs with a displayed representation of h by the cons endpoint $\mathsf{FramedListEndpoint}(\mathsf{cons}(a, xs))$. Composing this endpoint display with the assumed empty-frame display gives an internal sameness between a cons framed endpoint and the nil framed endpoint. Framed endpoint no-confusion for nil versus cons eliminates this case. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength successor empty absurd`

Theorem 36.124 (Framed list bridge classifier displayed-spine exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. For histories* $h, k : \mathsf{BHist}$ *and finite spines* xs, ys*, if* $\mathsf{FramedListSpineRep}_A(h, xs)$ *and* $\mathsf{FramedListSpineRep}_A(k, ys)$ *hold, then*

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \iff \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

The displayed spine representatives therefore give the exact finite-spine readback of the framed bridge classifier at the two endpoints.

Proof. For the right-to-left implication, use the displayed spines xs and ys as the existential bridge witnesses. The two representation hypotheses supply the endpoint clauses, and the displayed $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ supplies the finite-spine classifier clause.

For the left-to-right implication, unpack the bridge classifier into finite spines xs_0, ys_0 with $\mathsf{FramedListSpineRep}_A(h, xs_0)$, $\mathsf{FramedListSpineRep}_A(k, ys_0)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$. Apply Theorem 36.116 at h to obtain $\mathsf{ListClassifierSpec}(\sim_A, xs, xs_0)$. Apply the same theorem at k to obtain $\mathsf{ListClassifierSpec}(\sim_A, ys, ys_0)$, then use Theorem 36.47 and the certificate cert to reverse this comparison. Two applications of Theorem 36.46 compose $xs \sim_A xs_0$, $xs_0 \sim_A ys_0$, and $ys_0 \sim_A ys$, yielding $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier displayed spine exactness`

Theorem 36.125 (Framed list bridge empty-left source refinement)**.** *Let* $A, B : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers and let* $\sim_A, \sim_B$ *be source classifiers. Assume source refinement preserves carrier membership and head classification:*

$$\forall x : \mathsf{BHist}.\ A(x) \to B(x), \qquad \forall x\, y : \mathsf{BHist}.\ x \sim_A y \to x \sim_B y.$$

Then every framed bridge from the displayed empty-left endpoint $\mathsf{E0}(\mathsf{e})$ *to* k *over* $(A, \sim_A)$ *refines to a framed bridge over* $(B, \sim_B)$*:*

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(\mathsf{E0}(\mathsf{e}), k) \to \mathsf{FramedListBridgeClassifier}_{B,\sim_B}(\mathsf{E0}(\mathsf{e}), k).$$

Proof. Unpack the framed bridge into finite spines xs, ys, two $\mathsf{FramedListSpineRep}$ witnesses, and a $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ witness. The carrier inclusion transports every entry field in both spine representations while preserving the endpoint equalities. Induction on the finite-spine classifier transports each head relation through the classifier inclusion and rebuilds the tail witness. Repack the same two spines with these transported witnesses. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier empty left source refinement`

Theorem 36.126 (Framed list bridge classifier constructor exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. For histories* $h, k : \mathsf{BHist}$*, finite spines* xs, ys*, and source histories* $a, b : \mathsf{BHist}$*, the framed bridge classifier has the following constructor exactness behavior: The roadmap certificate reads: traditional target finite-list framed bridge constructor exactness; BEDC source* BHist *with* $\mathsf{PairFrame}$ *and* $\mathsf{FramedListEndpoint}$*; carrier* $\mathsf{FramedListHistoryCarrier}_A$*; classifier* $\mathsf{FramedListBridgeClassifier}$*; stability from the source* $\mathsf{NameCert}$ *plus* $\mathsf{ListSourceHsameCompatible}$*; ledger displayed framed spines; core theorem nil-nil introduction, mixed nil/cons separation, and cons-cons iff; exactness finite-spine constructor readback; standard bridge finite spines; status* $\mathsf{CheckedCert}$*.*

1. *if* $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$ *and* $\mathsf{FramedListSpineRep}_A(k, \mathsf{nil})$ *hold, then* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *holds;*
2. *if* $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$*,* $\mathsf{FramedListSpineRep}_A(k, \mathsf{cons}(b, ys))$*, and* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *hold, then* $\bot$*;*
3. *if* $\mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs))$*,* $\mathsf{FramedListSpineRep}_A(k, \mathsf{nil})$*, and* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k)$ *hold, then* $\bot$*;*
4. *if* $\mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs))$ *and* $\mathsf{FramedListSpineRep}_A(k, \mathsf{cons}(b, ys))$ *hold, then*

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \iff (a \sim_A b) \wedge \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Proof. In the nil-nil case, the nil branch of Theorem 36.13 gives

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{nil}, \mathsf{nil}).$$

The reverse direction of Theorem 36.124 repackages this finite-spine classifier with the two displayed nil endpoint representatives.

In the nil-cons case, apply Theorem 36.124 to the two displayed endpoint representatives and the bridge witness. This yields

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{nil}, \mathsf{cons}(b, ys)).$$

The mixed nil-cons clause of Theorem 36.13 eliminates that finite-spine classifier.

The cons-nil case is symmetric. Displayed-spine exactness gives

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{nil}),$$

and the mixed cons-nil clause of Theorem 36.13 eliminates this shape.

In the cons-cons case, displayed-spine exactness turns the bridge witness into

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{cons}(b, ys)).$$

The cons-cons clause of Theorem 36.13 projects the source-head classifier $a \sim_A b$ and the tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$.

Conversely, suppose the source-head classifier $a \sim_A b$ and the tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ are given. The cons-cons clause of Theorem 36.13 rebuilds

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{cons}(b, ys)).$$

The reverse direction of Theorem 36.124 repackages this finite-spine classifier as $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h$
□

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier constructor exactness`

Theorem 36.127 (Framed list bridge classifier equivalence fields)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. Define the framed carrier by*

$$\mathsf{FramedListHistoryCarrier}_A(h) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FramedListSpineRep}_A(h, xs).$$

Lean4 (definition) `BEDC.Derived.ListUp.FramedListHistoryCarrier`

Then $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}$ *satisfies the five classifier fields below. Carrier-relative reflexivity:*

$$\mathsf{FramedListHistoryCarrier}_A(h) \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, h).$$

Endpoint exposure for any bridge witness:

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{FramedListHistoryCarrier}_A(h) \wedge \mathsf{FramedListHistoryCarrier}_A(k).$$

Symmetry:

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(k, h).$$

Transitivity:

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(k, u) \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, u)$$

Carrier transport:

$$\mathsf{FramedListHistoryCarrier}_A(h) \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{FramedListHistoryCarrier}_A(k).$$

The ten-field fit is: traditional target finite-list framed bridge equivalence fields; BEDC source BHist*; carrier framed spine representation; classifier* $\mathsf{FramedListBridgeClassifier}_{A,\sim_A}$*; stability obligations the five displayed clauses; ledger finite displayed spines; core theorem this fields theorem; exactness by displayed-spine exactness and framed coherence; standard bridge finite spines; status* ProofTarget*.*

Proof. Endpoint exposure unpacks the bridge witness into finite spines xs, ys with $\mathsf{FramedListSpineRep}_A(h, xs)$, $\mathsf{FramedListSpineRep}_A(k, ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$; the two representation witnesses are exactly the two framed-carrier witnesses.

For carrier-relative reflexivity, choose the displayed spine supplied by the framed-carrier witness at h on both endpoints. Induct over that finite spine. The nil case is the nil constructor of ListClassifierSpec. In the cons case, the entry predicate stored in FramedListSpineRep supplies A for the displayed head, so cert.equiv_refl gives the head classifier witness and the induction hypothesis supplies the tail witness.

For symmetry, unpack the bridge witness, apply Theorem 36.47 to the finite-spine classifier witness, and rebuild the bridge with the two displayed endpoints swapped. For transitivity, unpack bridge witnesses for h, k and k, u. The two middle spines both represent k, so Theorem 36.116 compares them. Compose the first bridge classifier, the middle comparison, and the second bridge classifier by two applications of Theorem 36.46, then rebuild the bridge from the exposed first and last spines. Carrier transport is the right endpoint projection of endpoint exposure, with the displayed left-carrier premise matching the NameCert field shape. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier equivalence fields`

Corollary 36.128 (Framed finite-spine paper certificate)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. Define the framed carrier by*

$$\mathsf{FramedListHistoryCarrier}_A(h) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FramedListSpineRep}_A(h, xs),$$

and define the framed bridge classifier by

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(h, k) \iff \exists xs\, ys.\ \mathsf{FramedListSpineRep}_A(h, xs) \wedge \mathsf{FramedListSpineRep}_A(k, ys) \wedge \mathsf{Lis}$$

Lean4 (definition) `BEDC.Derived.ListUp.FramedListBridgeClassifier`

With pattern and ledger predicates both given by $\mathsf{FramedListHistoryCarrier}_A$*, these fields form the paper certificate target*

$$\mathsf{SemanticNameCert}(\mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListBrid}$$

This certificate is for the framed carrier and framed bridge classifier above. It does not identify $\mathsf{FramedListHistoryCarrier}_A$ *with the unframed* $\mathsf{ListHistoryCarrier}_A$ *used by the public* $\mathsf{List}^{\uparrow}$ *target.*

Proof. The source certificate supplies the entrywise reflexivity, symmetry, transitivity, and carrier transport facts for $\sim_A$. The framed endpoint definition supplies a separated finite-spine readback, and Theorem 36.116 supplies the comparison datum for two presentations of the same history. Reflexivity chooses the displayed spine for a framed carrier witness and classifies it entry by entry using cert. Symmetry and transitivity unpack the two existential framed bridge witnesses and apply the structural finite-spine symmetry and transitivity theorems for ListClassifierSpec; at the shared middle history, framed coherence identifies the two exposed middle spines. Carrier transport follows because a framed bridge witness contains a framed representation of the target history. Since the pattern and ledger predicates are the carrier itself, their soundness fields are identity implications. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier semantic name certificate`

Corollary 36.129 (Framed finite-spine paper certificate for hsame)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*. Define the framed carrier by*

$$\mathsf{FramedListHistoryCarrier}_A(h) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FramedListSpineRep}_A(h, xs),$$

and define the hsame framed bridge classifier by

$$\mathsf{FramedListBridgeClassifier}_{A,\mathsf{hsame}}(h, k) \iff \exists xs\, ys.\ \mathsf{FramedListSpineRep}_A(h, xs) \wedge \mathsf{FramedListSpineRep}_A(k, ys) \wedge$$

With pattern and ledger predicates both given by $\mathsf{FramedListHistoryCarrier}_A$*, these fields form the paper certificate target*

$$\mathsf{SemanticNameCert}(\mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListHistoryCarrier}_A, \mathsf{FramedListBrid}$$

Proof. Apply Corollary 36.128 with source classifier hsame. The compatibility premise is exactly Theorem 36.87, and the remaining source certificate input is the displayed $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$. □

Theorem 36.130 (Framed list public length successor endpoint readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and assume* $\mathsf{ListSourceHsameCompatible}(A, \sim_A)$*. If a history* h *has framed public length* $n+1$*, then* h *exposes a head, tail spine, exact predecessor length, and framed cons endpoint:*

$$\mathsf{FramedListPublicLength}_A(h, n+1) \to \exists a : \mathsf{BHist},\, xs : \mathsf{HistSpine}.\ A(a) \wedge \mathsf{FramedListSpineRep}_A(h, \mathsf{cons}(a, xs)) \wedge \ell(xs) = n \wedge \mathsf{hsame}(h, \mathsf{E1}(\mathsf{PairFrame}(a, \mathsf{FramedListEndpoint}(xs)))).$$

Proof. Apply framed public-length constructor readback to the successor witness. This exposes a, xs, the source row $A(a)$, a displayed framed spine representation of h by $\mathsf{cons}(a, xs)$, and a tail public-length witness at n.

The displayed spine representation restricts its entry predicate to the tail entries of xs, while reflexivity gives the endpoint representation of $\mathsf{FramedListEndpoint}(xs)$ by xs. Thus the tail also has the canonical public length $\ell(xs)$. Well-definedness of framed public length identifies $\ell(xs) = n$. The endpoint row is the cons-endpoint component of the displayed framed spine representation. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicLength successor endpoint readback`

Theorem 36.131 (Framed list bridge one-left source refinement)**.** *Let* $A, B : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers and let* $\sim_A, \sim_B$ *be source classifiers. Assume source refinement preserves carrier membership and head classification:*

$$\forall x : \mathsf{BHist}.\ A(x) \to B(x), \qquad \forall x\, y : \mathsf{BHist}.\ x \sim_A y \to x \sim_B y.$$

Then every framed bridge from the displayed one-headed endpoint $\mathsf{E1}(t)$ *to* k *over* $(A, \sim_A)$ *refines to a framed bridge over* $(B, \sim_B)$*:*

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(\mathsf{E1}(t), k) \to \mathsf{FramedListBridgeClassifier}_{B,\sim_B}(\mathsf{E1}(t), k).$$

Proof. Unpack the framed bridge into finite spines xs, ys, two $\mathsf{FramedListSpineRep}$ witnesses, and a finite-spine classifier. The carrier inclusion transports every entry field in both spine representations without changing the displayed endpoints. Induction on the finite-spine classifier transports each head relation through the classifier inclusion and rebuilds the tail classifier. Repacking the same two spines gives the refined bridge at the visible one-left endpoint. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier e1 left source refinement`

Theorem 36.132 (List spine bridge endpoint exposure and symmetry)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. The compact per-object fit is: traditional target finite-list classifier stability; BEDC source* A *over* BHist *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$*; stability obligations endpoint exposure and symmetry; ledger finite spine representation; core theorem the two displayed clauses below; exactness via* $\mathsf{ListSpineRep}$*; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. For all histories* h *and* k*, the bridge classifier satisfies endpoint exposure:*

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListHistoryCarrier}_A(h) \wedge \mathsf{ListHistoryCarrier}_A(k).$$

Moreover, for any $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, the bridge classifier is symmetric:*

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k, h).$$

Proof. Unpack a bridge witness into finite spines xs and ys, displayed representations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and the finite-spine classifier witness $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Endpoint exposure applies the soundness direction of Theorem 36.61 to the two displayed representations. For symmetry, keep the two representations, apply Theorem 36.47 to the source certificate cert and the finite-spine classifier witness, then rebuild the bridge witness with the represented spines swapped. □

Theorem 36.133 (List spine bridge stability from spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional target finite-list classifier stability; BEDC source* A *over* BHist *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$*; stability obligations reflexivity, symmetry, transitivity, and endpoint closure; ledger finite spine representation; core theorem this block; exactness via* $\mathsf{ListSpineRep}$*; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. Then the bridge classifier satisfies all four clauses: carrier-relative reflexivity, meaning*

$$\mathsf{ListHistoryCarrier}_A(h) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, h);$$

symmetry, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k, h);$$

transitivity, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k, u) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, u);$$

and classifier-induced endpoint closure, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListHistoryCarrier}_A(k).$$

The $\mathsf{NameCert}$*-shaped carrier-transport field follows immediately by weakening this endpoint-closure implication with an unused* $\mathsf{ListHistoryCarrier}_A(h)$ *premise.*

Proof. For reflexivity, use representation coverage to choose a finite-spine representation of h, apply finite-spine reflexivity under the source certificate, and rebuild the bridge witness with the same representation on both endpoints. Symmetry is the second clause of Theorem 36.132. Transitivity is Theorem 36.63 applied to the displayed spine coherence and the source certificate. Endpoint closure is the second endpoint component exposed by Theorem 36.132; carrier transport is obtained by weakening that implication with the unused left-carrier premise. □

Theorem 36.134 (List recursive history classifier transitivity from spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional target finite-list generated classifier stability; BEDC source* A *over* BHist *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}$*; stability obligation transitivity; ledger finite spine representation; core theorem this block; exactness via* $\mathsf{ListSpineRep}$ *and* $\mathsf{ListSpineBridgeClassifier}$*; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. Then for all histories* h, k, u*, if*

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h, k)$$

and

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(k, u)$$

hold, then

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,u)$$

holds.

Proof. Use the classifier equivalence clause of Theorem 36.61 to convert the two recursive classifier witnesses into $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k)$ and $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k,u)$ witnesses. Apply Theorem 36.63 with the displayed spine coherence hypothesis and the source certificate to obtain $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,u)$. The reverse direction of the same exactness equivalence converts this bridge witness back into the recursive classifier witness $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,u)$. □

Theorem 36.135 (List recursive history classifier transitivity from cons-boundary coherence)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, let $\mathsf{cert} : \mathsf{NameCert}(A,\sim_A)$, and assume $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. The ten-field fit is: traditional target finite-list recursive classifier transitivity; BEDC source A over BHist with $\mathsf{NameCert}(A,\sim_A)$; carrier $\mathsf{ListHistoryCarrier}_A$; classifier $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}$; stability obligation transitivity; ledger finite spine representation plus cons-boundary comparison; core theorem this block; exactness through $\mathsf{ListSpineRep}$; standard bridge finite spines; status $\mathsf{ProofTarget}$. Then for all histories h,k,u,*

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,k) \to \mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(k,u) \to \mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h,u)$$

holds.

Proof. First obtain $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ from the displayed cons-boundary coherence by applying Theorem 36.71.

With that spine-coherence witness and the source certificate in hand, apply Theorem 36.134. It gives exactly the recursive classifier transitivity implication for the three histories h,k,u. □

Theorem 36.136 (List spine bridge stability from cons-boundary coherence)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, let $\mathsf{cert} : \mathsf{NameCert}(A,\sim_A)$, and assume $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. The ten-field fit is: traditional target finite-list classifier stability; BEDC source A over BHist with $\mathsf{NameCert}(A,\sim_A)$; carrier $\mathsf{ListHistoryCarrier}_A$; classifier $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$; stability obligations reflexivity, symmetry, transitivity, and endpoint closure; ledger finite spine representation; core theorem this block; exactness via $\mathsf{ListSpineRep}$; standard bridge finite-spine readback; status $\mathsf{ProofTarget}$. Then the bridge classifier satisfies all four clauses: carrier-relative reflexivity, meaning*

$$\mathsf{ListHistoryCarrier}_A(h) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,h);$$

symmetry, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k,h);$$

transitivity, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(k,u) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,u);$$

and classifier-induced endpoint closure, meaning

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h,k) \to \mathsf{ListHistoryCarrier}_A(k).$$

The $\mathsf{NameCert}$*-shaped carrier-transport field follows immediately by weakening this endpoint-closure implication with an unused* $\mathsf{ListHistoryCarrier}_A(h)$ *premise.*

Proof. Theorem 36.71 turns the cons-boundary coherence witness into $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Apply Theorem 36.133 with that spine-coherence witness and the source certificate. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier stability from cons boundary`

Theorem 36.137 (List history semantic fields source refinement)**.** *Let $A, B : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers. The ten-field fit is: traditional target finite-list source refinement; BEDC source* BHist*; carrier* ListHistoryCarrier*, pattern* ListHistoryPattern*, and ledger* ListHistoryLedger *as the three list semantic fields; classifier unused; stability source inclusion; ledger finite spine; core theorem the three displayed implications below; exactness endpoint preservation; standard bridge finite spines; status* ProofTarget*. If the source carrier of A is included in the source carrier of B,*

$$\forall h : \mathsf{BHist}.\ A(h) \to B(h),$$

then for every history h,

$$\mathsf{ListHistoryCarrier}_A(h) \to \mathsf{ListHistoryCarrier}_B(h),$$

$$\mathsf{ListHistoryPattern}_A(h) \to \mathsf{ListHistoryPattern}_B(h),$$

and

$$\mathsf{ListHistoryLedger}_A(h) \to \mathsf{ListHistoryLedger}_B(h).$$

Lean4 ✓ `BEDC.Derived.ListUp.FramedListSemanticFields source refinement`

Proof. Induct on the displayed generation of the carrier and ledger witnesses. The nil case reuses the empty endpoint witness. The cons case applies the source inclusion to the head witness, applies the induction hypothesis to the tail, and rebuilds the same continuation endpoint. For the pattern field, use the displayed nil and cons cases: the nil pattern is unchanged, and the cons pattern uses the source inclusion on the head together with the carrier-refinement clause for the tail. □

Theorem 36.138 (List spine coherence source restriction)**.** *Let $A, B : \mathsf{BHist} \to \mathsf{Prop}$ be BEDC history source carriers, and let $\sim$ be a source classifier. The ten-field fit is: traditional target finite-list source restriction; BEDC source* BHist*; carrier* ListSpineRep*; classifier* ListClassifierSpec($\sim$)*; stability source inclusion; ledger finite spine; core theorem coherence restriction; exactness inherited from ambient spine coherence; standard bridge finite-spine readback; status* ProofTarget*. If the restricted source is included in the ambient source,*

$$\forall h : \mathsf{BHist}.\ A(h) \to B(h),$$

and the ambient source has coherent spine presentations,

$$\mathsf{ListSpineRepCoherent}_{B,\sim},$$

then the restricted source has coherent spine presentations:

$$\mathsf{ListSpineRepCoherent}_{A,\sim}.$$

Proof. Take two A-spine presentations $\mathsf{ListSpineRep}_A(m, xs)$ and $\mathsf{ListSpineRep}_A(m, ys)$ of the same history. Apply Theorem 36.88 with the displayed source inclusion and identity classifier weakening $\forall x\, y : \mathsf{BHist}.\ x \sim y \to x \sim y$. Its spine-representation clause transports both presentations to $\mathsf{ListSpineRep}_B(m, xs)$ and $\mathsf{ListSpineRep}_B(m, ys)$. The ambient coherence hypothesis $\mathsf{ListSpineRepCoherent}_{B,\sim}$ applied to those two transported presentations yields $\mathsf{ListClassifierSpec}(\sim, xs, ys)$, which is exactly the coherence datum required for $\mathsf{ListSpineRepCoherent}_{A,\sim}$. □

Theorem 36.139 (BHist-parametric list classifier append stability)**.** *For any relation* $sameA$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *and finite spines* $xs, ys, zs, ws : \mathsf{ListCarrier}(\mathsf{BHist})$*, if*

$$\mathsf{ListClassifierSpec}(sameA, xs, ys)$$

and

$$\mathsf{ListClassifierSpec}(sameA, zs, ws)$$

hold, then

$$\mathsf{ListClassifierSpec}(sameA, xs{+}{+}zs, ys{+}{+}ws)$$

holds.

Proof. Induct on the first classified spine pair. In the nil case, the first classifier forces the second left spine to be nil, so the conclusion is exactly the second classifier witness. In the cons case, the second left spine is also cons; keep the head classifier witness and apply the induction hypothesis to the classified tails and the second append witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec BHist append`

Theorem 36.140 (BHist-parametric classified list prefix cancellation)**.** *For any relation* $sameA$: $\mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *and finite spines* $p, p', x, y : \mathsf{ListCarrier}(\mathsf{BHist})$*, the ten-field fit is: traditional target finite-list prefix cancellation; BEDC source carried by the history entries of the finite spine; carrier* $\mathsf{ListCarrier}(\mathsf{BHist})$*; classifier* $\mathsf{ListClassifierSpec}(sameA)$*; stability append under classified prefixes; ledger finite spine; core theorem classified left prefix cancellation; exactness inherited from structural nil/cons classification; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. If*

$$\mathsf{ListClassifierSpec}(sameA, p, p')$$

and

$$\mathsf{ListClassifierSpec}(sameA, p{+}{+}x, p'{+}{+}y)$$

hold, then

$$\mathsf{ListClassifierSpec}(sameA, x, y)$$

holds.

Proof. Induct on the prefix classifier witness $\mathsf{ListClassifierSpec}(sameA, p, p')$. In the nil case, both prefixes are empty and the appended classifier is already the required classifier on x and y. In the cons case, invert the appended classifier at the exposed cons heads; its tail is a classifier for the two appended tails. Apply the induction hypothesis to the tail prefix classifier and that tail-appended classifier. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec BHist append left cancel classified`

Theorem 36.141 (Source-parametric classified list suffix cancellation)**.** *For any source type* A*, relation* $sameA : A \to A \to \mathsf{Prop}$*, and finite spines* $x, y, s, s' : \mathsf{ListCarrier}(A)$*, the ten-field fit is: traditional target finite-list suffix cancellation; BEDC source carried by the source entries of the finite spine; carrier* $\mathsf{ListCarrier}(A)$*; classifier* $\mathsf{ListClassifierSpec}(sameA)$*; stability append under classified suffixes; ledger finite spine; core theorem classified right suffix cancellation; exactness inherited from structural nil/cons classification and reverse readback; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. If*

$$\mathsf{ListClassifierSpec}(sameA, s, s')$$

and

$$\mathsf{ListClassifierSpec}(sameA, x{+}{+}s, y{+}{+}s')$$

hold, then

$$\mathsf{ListClassifierSpec}(sameA, x, y)$$

holds.

Proof. Reverse both classified appended spines. Theorem 36.41 turns the second witness into a classifier on $\mathsf{reverse}(s){+}{+}\mathsf{reverse}(x)$ and $\mathsf{reverse}(s'){+}{+}\mathsf{reverse}(y)$, while the suffix classifier becomes a classifier between $\mathsf{reverse}(s)$ and $\mathsf{reverse}(s')$. Apply Theorem 36.140 to cancel these reversed prefixes, obtaining a classifier between $\mathsf{reverse}(x)$ and $\mathsf{reverse}(y)$. Apply Theorem 36.41 again; finite-spine reverse involutivity transports the result back to $\mathsf{ListClassifierSpec}(sameA, x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec append right cancel classified`

Theorem 36.142 (Source-parametric classified append context equivalence)**.** *For any source type A, relation $sameA : A \to A \to \mathsf{Prop}$, and finite spines $p, p', x, y, s, s' : \mathsf{ListCarrier}(A)$, the ten-field fit is: traditional target finite-list two-sided context equivalence; BEDC source carried by the source entries of the finite spine; carrier $\mathsf{ListCarrier}(A)$; classifier $\mathsf{ListClassifierSpec}(sameA)$; stability append under classified prefixes and suffixes; ledger finite spine; core theorem classified append-context equivalence; exactness inherited from structural nil/cons classification; standard bridge finite-spine readback; status* ProofTarget*. If*

$$\mathsf{ListClassifierSpec}(sameA, p, p')$$

and

$$\mathsf{ListClassifierSpec}(sameA, s, s')$$

hold, then

$$\mathsf{ListClassifierSpec}(sameA, x, y) \iff \mathsf{ListClassifierSpec}(sameA, p{+}{+}x{+}{+}s, p'{+}{+}y{+}{+}s')$$

holds.

Proof. For the forward implication, append stability first adds the classified prefixes p and p' to the witness for x and y, and then adds the classified suffixes s and s'. For the reverse implication, use Theorem 36.141 to cancel the classified suffixes from the two contextual spines. The remaining classifier relates $p{+}{+}x$ to $p'{+}{+}y$, so Theorem 36.140 cancels the classified prefixes and yields $\mathsf{ListClassifierSpec}(sameA, x, y)$. □

Theorem 36.143 (BHist-parametric classified append context equivalence)**.** *For any relation $sameA : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ and finite spines $p, p', x, y, s, s' : \mathsf{ListCarrier}(\mathsf{BHist})$, if*

$$\mathsf{ListClassifierSpec}(sameA, p, p')$$

and

$$\mathsf{ListClassifierSpec}(sameA, s, s')$$

hold, then

$$\mathsf{ListClassifierSpec}(sameA, p{+}{+}(x{+}{+}s), p'{+}{+}(y{+}{+}s')) \iff \mathsf{ListClassifierSpec}(sameA, x, y)$$

holds.

Proof. For the left-to-right direction, first cancel the classified prefixes using Theorem 36.140. Then cancel the classified suffixes using Theorem 36.141. For the right-to-left direction, append the classified middle spines to the classified suffixes using Theorem 36.139, and append the classified prefixes to the result using the same append stability. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec BHist append context iff`

Theorem 36.144 (List represented-spine classifier alignment)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The roadmap ten-field fit is: traditional target finite-list readback along represented histories; BEDC source* A *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability supplied by source-certificate transitivity and representation coherence; ledger finite spines* xs_0 *and* ys_0*; core theorem represented-spine classifier alignment; exactness by bridge-classifier unpacking; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. If* $\mathsf{ListSpineRep}_A(h, xs_0)$*,* $\mathsf{ListSpineRep}_A(k, ys_0)$*, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *hold, then*

$$\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$$

holds.

Proof. Unpack the bridge classifier to finite spines xs, ys with $\mathsf{ListSpineRep}_A(h, xs)$, $\mathsf{ListSpineRep}_A(k, ys)$, and $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Coherence at h gives $\mathsf{ListClassifierSpec}(\sim_A, xs_0, xs)$, and coherence at k gives $\mathsf{ListClassifierSpec}(\sim_A, ys, ys_0)$. Apply the source-certificate transitivity theorem $\mathsf{ListClassifierSpec_trans_from_nameCert}$ with cert to compose the three finite-spine classifier links, yielding $\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier represented spine alignment`

Theorem 36.145 (List represented-spine classifier displayed exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional finite-list readback; BEDC source* BHist*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from the source certificate and spine coherence; ledger finite spines* xs_0 *and* ys_0*; core theorem displayed-spine exactness; exactness by the displayed-spine equivalence; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. For histories* $h, k : \mathsf{BHist}$ *and finite BEDC-history spines* xs_0, ys_0*, if*

$$\mathsf{ListSpineRep}_A(h, xs_0)$$

and

$$\mathsf{ListSpineRep}_A(k, ys_0)$$

hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \iff \mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$$

holds.

Proof. For the forward implication, apply Theorem 36.144 to the two displayed representation witnesses and the public classifier witness. For the reverse implication, unfold the bridge classifier definition. The displayed witnesses $\mathsf{ListSpineRep}_A(h, xs_0)$ and $\mathsf{ListSpineRep}_A(k, ys_0)$, together with the finite-spine classifier witness $\mathsf{ListClassifierSpec}(\sim_A, xs_0, ys_0)$, are exactly the existential bridge data, so they repack as $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier displayed spine exactness`

Theorem 36.146 (List represented-spine nil classifier inversion)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional target finite-list nil readback; BEDC source* A *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from* cert *plus spine coherence; ledger finite spines; core theorem nil classifier inversion; exactness by represented-spine alignment; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k : \mathsf{BHist}$ *and a finite BEDC-history spine* ys*, if*

$$\mathsf{ListSpineRep}_A(h, \mathsf{nil}),$$

$$\mathsf{ListSpineRep}_A(k, ys),$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$$

hold, then $ys = \mathsf{nil}$*. Symmetrically, for a finite BEDC-history spine* xs*, if* $\mathsf{ListSpineRep}_A(h, xs)$*,* $\mathsf{ListSpineRep}_A(k, \mathsf{nil})$*, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *hold, then* $xs = \mathsf{nil}$*.*

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier nil classifier inversion`

Proof. Apply Theorem 36.144 to the displayed represented nil spine, the represented spine ys, and the public classifier witness. This gives

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{nil}, ys).$$

Case-analyze the finite spine ys. The nil branch is immediate. In the cons branch, Lemma 36.14 contradicts the mixed nil-cons classifier witness, equivalently the mixed branch excluded by Theorem 36.13. The symmetric clause applies the same alignment theorem to obtain

$$\mathsf{ListClassifierSpec}(\sim_A, xs, \mathsf{nil})$$

and then case-analyzes xs; the nil branch is immediate, and the cons branch is the symmetric mixed case ruled out by the same no-confusion lemma and case exactness theorem. □

Theorem 36.147 (List represented-spine classifier constructor exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional target finite-list readback over* BHist*; BEDC source* A *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from* cert *and spine coherence; ledger finite spines; core theorem represented-spine classifier constructor exactness; exactness by displayed-spine classifier exactness; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k, a, b : \mathsf{BHist}$ *and finite BEDC-history spines* xs, ys*, the public classifier has the following represented-spine constructor behavior:*

1. *if* $\mathsf{ListSpineRep}_A(h, \mathsf{nil})$*,* $\mathsf{ListSpineRep}_A(k, \mathsf{cons}(b, ys))$*, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *hold, then* $\bot$*;*
2. *if* $\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs))$*,* $\mathsf{ListSpineRep}_A(k, \mathsf{nil})$*, and* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$ *hold, then* $\bot$*;*
3. *if* $\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs))$ *and* $\mathsf{ListSpineRep}_A(k, \mathsf{cons}(b, ys))$ *hold, then*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \iff (a \sim_A b) \wedge \mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Proof. For the nil-cons clause, apply the forward direction of Theorem 36.145 to the two represented endpoints and the public classifier witness. This gives

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{nil}, \mathsf{cons}(b, ys)).$$

Lemma 36.14 eliminates that mixed finite-spine classifier. The cons-nil clause uses the same displayed exactness theorem to obtain

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{nil}),$$

and Lemma 36.14 eliminates this mixed shape as well. For the cons-cons clause, apply the same theorem to obtain

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{cons}(b, ys)).$$

The cons-cons branch of Theorem 36.13 then projects the source-head classifier $a \sim_A b$ and the recursive tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Conversely, suppose the source-head classifier $a \sim_A b$ and the recursive tail classifier $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ are given. The cons-cons branch of Theorem 36.13 rebuilds

$$\mathsf{ListClassifierSpec}(\sim_A, \mathsf{cons}(a, xs), \mathsf{cons}(b, ys)).$$

With the two displayed representation witnesses, the reverse direction of Theorem 36.145 repacks this finite-spine classifier as $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier represented spine constructor exactness`

Lemma 36.148 (List represented prefix self-classifier)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and let* $p : \mathsf{BHist}$ *and* $xs : \mathsf{HistSpine}$ *be given. If*

$$\mathsf{ListSpineRep}_A(p, xs)$$

holds, then

$$\mathsf{ListClassifierSpec}(\sim_A, xs, xs)$$

holds.

Proof. Induct on the displayed witness $\mathsf{ListSpineRep}_A(p, xs)$. The nil representation gives the nil clause of $\mathsf{ListClassifierSpec}(\sim_A, xs, xs)$.

In the cons representation, the witness exposes a head a with $A(a)$ and a represented tail. The source reflexivity field of cert gives $a \sim_A a$, and the induction hypothesis gives the tail self-classifier. The cons clause of Definition 36.13 rebuilds $\mathsf{ListClassifierSpec}(\sim_A, xs, xs)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep self classifier`

Theorem 36.149 (List public-prefix append left cancellation)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*. For histories* $p, h, k, t, u : \mathsf{BHist}$ *and finite spines* $xs, ys, zs : \mathsf{HistSpine}$*, suppose*

$$\mathsf{ListSpineRep}_A(p, xs)$$

$$\mathsf{ListSpineRep}_A(h, xs{+}{+}ys), \qquad \mathsf{ListSpineRep}_A(k, xs{+}{+}zs), \qquad \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$$

hold. Then

$$\mathsf{ListClassifierSpec}(\sim_A, ys, zs)$$

holds. Consequently, if $\mathsf{ListSpineRep}_A(t, ys)$ *and* $\mathsf{ListSpineRep}_A(u, zs)$ *also hold, then*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)$$

holds.

Proof. First build the self-classifier for the represented prefix. Apply Definition 36.148 to the displayed witness $\mathsf{ListSpineRep}_A(p, xs)$ to obtain $\mathsf{ListClassifierSpec}(\sim_A, xs, xs)$.

The cons-boundary hypothesis gives $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$ by Definition 36.71. Apply Definition 36.144 to the two displayed representations of h and k, together with the public classifier witness, to obtain

$$\mathsf{ListClassifierSpec}(\sim_A, xs{+}{+}ys, xs{+}{+}zs).$$

Now apply Definition 36.140 to the prefix self-classifier and this appended classifier. The result is $\mathsf{ListClassifierSpec}(\sim_A, ys, zs)$.

For the final public-classifier form, use the reverse implication of Definition 36.145 with the representations $\mathsf{ListSpineRep}_A(t, ys)$ and $\mathsf{ListSpineRep}_A(u, zs)$. Repacking the finite tail classifier through the displayed tail representations yields $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t, u)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineBridgeClassifier public prefix append left cancel`

Theorem 36.150 (List spine bridge source refinement)**.** *Let* $A, B : \mathsf{BHist} \to \mathsf{Prop}$ *be BEDC history source carriers, and let* $\sim_A$ *and* $\sim_B$ *be source classifiers. The ten-field fit is: traditional target finite-list source refinement; BEDC source* BHist *with* ListSpineRep*; carrier* ListHistoryCarrier*; classifier* ListSpineBridgeClassifier*; stability obligation carrier and classifier preservation under source refinement; ledger finite spine representation; core theorem the three displayed implications below; exactness inherited from* ListSpineRep*; standard bridge finite-spine readback; status* ProofTarget*. Assume the source carrier and source classifier of* A *are included in those of* B*:*

$$\forall h : \mathsf{BHist}.\ A(h) \to B(h)$$

and

$$\forall x\, y : \mathsf{BHist}.\ x \sim_A y \to x \sim_B y.$$

Then the displayed list-spine data transport from A *to* B*:*

$$\mathsf{ListSpineRep}_A(h, xs) \to \mathsf{ListSpineRep}_B(h, xs),$$

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListSpineBridgeClassifier}_{B,\sim_B}(h, k),$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k) \to \mathsf{ListSpineHistoryClassifier}_{B,\sim_B}(h, k).$$

Proof. Proceed by induction on the displayed $\mathsf{ListSpineRep}_A(h, xs)$ witness. The nil representation keeps the same endpoint equality. In the cons case, the carrier inclusion turns the recorded source datum $A(a)$ into $B(a)$, while the induction hypothesis transports the tail representation and the continuation and endpoint witnesses are reused unchanged. For the finite-spine classifier component, induct on the displayed $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$ witness and apply the classifier inclusion at each cons head. A bridge witness is then repacked with the transported left and right spines and the transported finite-spine classifier. The certificate classifier clause follows by unfolding that it is the bridge classifier and rebuilding the same endpoints. □

Lemma 36.151 (List spine representation endpoint transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For all histories* h *and* h' *and every finite list spine* xs*,* $\mathsf{ListSpineRep}_A(h, xs)$ *and* $\mathsf{hsame}(h, h')$ *imply* $\mathsf{ListSpineRep}_A(h', xs)$*.*

Proof. Proceed by induction on the displayed finite spine. The empty spine is transported by composing the endpoint empty witness with $\mathsf{hsame}(h, h')$. For a cons spine, keep the recorded head payload and tail representation, and compose the outer history endpoint witness with $\mathsf{hsame}(h, h')$; the payload and tail witnesses are unchanged. □

Theorem 36.152 (List spine bridge endpoint transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. The ten-field per-object fit is: traditional target finite-list classifier stability; BEDC source* A *over* BHist*; source classifier* $\sim_A$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}$*; stability endpoint transport; ledger finite-spine presentation; exactness via* $\mathsf{ListSpineRep}$*; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For all histories* h*,* k*,* h'*, and* k'*,*

$$\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \mathsf{hsame}(h, h') \to \mathsf{hsame}(k, k') \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h', k').$$

Proof. Unpack the bridge witness into finite spines xs and ys, the represented endpoints $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and the finite-spine classifier witness. Apply Lemma 36.151 to transport the two displayed endpoint representations along $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$. The finite-spine classifier witness is independent of the represented endpoints, so the bridge witness is rebuilt over h' and k' with the same spines. □

Theorem 36.153 (List recursive history classifier endpoint transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. The ten-field fit is: traditional target finite-list generated-classifier endpoint stability; BEDC source* BHist*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}$*; stability obligation endpoint transport; ledger finite spine; core theorem this transport law; exactness through Theorem 36.61; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*. For all histories* h*,* k*,* h'*, and* k'*,*

$$\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h, k) \to \mathsf{hsame}(h, h') \to \mathsf{hsame}(k, k') \to \mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h', k').$$

Proof. Convert the displayed recursive classifier witness to $\mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k)$ using the classifier equivalence in Theorem 36.61. Apply Theorem 36.152 to move the two endpoints along $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$. Convert the resulting bridge witness back through the reverse direction of the same exactness theorem to obtain $\mathsf{ListHistoryClassifier}^{\mathrm{rec}}_{A,\sim_A}(h', k')$. □

Theorem 36.154 (List spine bridge nil-cons separation under coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For all histories* h *and* k*, all source histories* a*, and all finite spines* xs*, the bridge classifier separates the nil and cons displayed endpoints:*

$$\mathsf{ListSpineRep}_A(h, \mathsf{nil}) \to \mathsf{ListSpineRep}_A(k, \mathsf{cons}(a, xs)) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \bot.$$

It also separates the symmetric cons-vs-nil orientation:

$$\mathsf{ListSpineRep}_A(h, \mathsf{cons}(a, xs)) \to \mathsf{ListSpineRep}_A(k, \mathsf{nil}) \to \mathsf{ListSpineBridgeClassifier}_{A,\sim_A}(h, k) \to \bot.$$

Proof. Unpack the bridge witness into finite spines u and v, endpoint presentations $\mathsf{ListSpineRep}_A(h, u)$ and $\mathsf{ListSpineRep}_A(k, v)$, and a finite-spine classifier $\mathsf{ListClassifierSpec}(\sim_A, u, v)$. In the nil-vs-cons orientation, apply the coherence datum at endpoint h to relate the displayed nil spine to u, and at endpoint k to relate v to the displayed $\mathsf{cons}(a, xs)$ spine. By Theorem 36.13, the comparison from nil to u forces u to have nil outer shape, and the bridge classifier from u to v then forces v to have nil outer shape. The comparison from v to the displayed $\mathsf{cons}(a, xs)$ spine is therefore a mixed nil-cons classifier witness, contrary to Lemma 36.14. In the cons-vs-nil orientation, the first coherence comparison forces u to have cons outer shape, the bridge classifier forces v to have cons outer shape, and the endpoint coherence comparison against the displayed nil spine is the symmetric impossible mixed classifier witness. □

Theorem 36.155 (List public classifier cons endpoint inversion)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list cons inversion; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from* cert *and spine coherence; ledger finite-spine representatives for the two cons endpoints and the two tails; core theorem public cons inversion; exactness by Theorem 36.147; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k, a, b, t, u, p, q : \mathsf{BHist}$*, if*

$$A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q))$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$$

hold, then

$$\mathsf{Rel}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u).$$

Proof. Apply Theorem 36.65 to the two tail carrier witnesses, choosing finite spines xs and ys with $\mathsf{ListSpineRep}_A(t, xs)$ and $\mathsf{ListSpineRep}_A(u, ys)$. The cons constructor for $\mathsf{ListSpineRep}_A$ combines these tail representations with $A(a)$, $A(b)$, the two continuation witnesses, and the two endpoint witnesses to build displayed representations of h by $\mathsf{cons}(a, xs)$ and k by $\mathsf{cons}(b, ys)$. Apply Theorem 36.147 to those represented cons endpoints and the public classifier witness. Its cons-cons clause yields $\mathsf{Rel}(a, b)$ together with $\mathsf{ListClassifierSpec}(\mathsf{Rel}, xs, ys)$. Repack the two tail representations and this finite-spine classifier witness through the bridge definition of $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$ to obtain the tail classifier $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u)$. □

Theorem 36.156 (List public classifier cons endpoint exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list cons constructor exactness; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from* cert *and spine coherence; ledger finite-spine representatives for the two cons endpoints and the two tails; core theorem public cons endpoint exactness; exactness by Theorem 36.147; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k, a, b, t, u, p, q : \mathsf{BHist}$*, if*

$$A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q))$$

holds, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \iff \mathsf{Rel}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u).$$

Proof. For the forward implication, apply Theorem 36.155 to the displayed cons endpoint data and the public classifier witness. This gives the head relation and the tail public classifier.

For the reverse implication, unpack the tail public classifier into finite spines xs and ys, tail representations $\mathsf{ListSpineRep}_A(t, xs)$ and $\mathsf{ListSpineRep}_A(u, ys)$, and a finite-spine classifier $\mathsf{ListClassifierSpec}(\mathsf{Rel}, xs, ys)$. The cons constructor for $\mathsf{ListSpineRep}_A$ combines these tail representations with $A(a)$, $A(b)$, $\mathsf{Cont}(a, t, p)$, $\mathsf{Cont}(b, u, q)$, $\mathsf{hsame}(h, \mathsf{E1}(p))$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$ to build cons spine representations of h and k. The finite-spine cons clause combines the head witness $\mathsf{Rel}(a, b)$ with the tail classifier witness to obtain the cons-cons $\mathsf{ListClassifierSpec}$ witness. Repack the two cons representations and that finite-spine witness through the bridge definition of $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$. □

Theorem 36.157 (List public classifier nil endpoint inversion)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list nil readback; BEDC source* BHist*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from* cert *and spine coherence; ledger nil finite-spine endpoint; core theorem public nil inversion; exactness by Theorem 36.146; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For all histories* $h, k : \mathsf{BHist}$*, if*

$$\mathsf{hsame}(h, \mathsf{e})$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$$

hold, then

$$\mathsf{hsame}(k, \mathsf{e}).$$

Symmetrically, if

$$\mathsf{hsame}(k, \mathsf{e})$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$$

hold, then

$$\mathsf{hsame}(h, \mathsf{e}).$$

Proof. From the displayed nil endpoint witness $\mathsf{hsame}(h, \mathsf{e})$, use the nil constructor of $\mathsf{ListSpineRep}_A$ to present h by nil. Unfold the bridge definition of $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$, obtaining finite spines xs and ys, endpoint representations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and the finite-spine classifier witness. Apply Theorem 36.146 to the displayed nil representation of h, the represented endpoint for k, and the public classifier witness. The theorem forces $ys = \mathsf{nil}$; unfolding the nil case of $\mathsf{ListSpineRep}_A(k, ys)$ recovers $\mathsf{hsame}(k, \mathsf{e})$. The symmetric clause starts from $\mathsf{hsame}(k, \mathsf{e})$, builds the displayed nil representation of k, applies the symmetric half of the same represented-spine nil inversion to force $xs = \mathsf{nil}$, and unfolds the resulting nil representation of h to recover $\mathsf{hsame}(h, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier nil endpoint inversion`

Theorem 36.158 (List public classifier constructor decomposition)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, and let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier. The ten-field fit is: traditional target finite-list public classifier case analysis; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability constructor*

decomposition; ledger finite-spine constructor witnesses; exactness by ListSpineRep *together with* ListClassifierSpec*; standard bridge finite spines; status* ProofTarget*. For all histories* $h, k : \mathsf{BHist}$*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$$

holds, then

$$\begin{array}{l} (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ \mathsf{Rel}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u) \end{array} \right) \end{array}$$

holds.

Proof. Unfold the public classifier as the spine bridge, obtaining finite spines xs and ys, endpoint representations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and a finite-spine classifier witness $\mathsf{ListClassifierSpec}(\mathsf{Rel}, xs, ys)$. Case-analyze xs and ys. The nil-nil case unfolds the two nil endpoint representations and gives $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. The mixed nil-cons and cons-nil cases contradict Lemma 36.14, equivalently the mixed branches of Theorem 36.13. In the cons-cons case, invert the two spine representations to obtain heads a and b, tails t and u, continuation histories p and q, the source carrier witnesses, the tail carrier witnesses, the continuation witnesses, and the two endpoint links. Invert the finite-spine classifier to obtain $\mathsf{Rel}(a, b)$ and the tail finite-spine classifier. Use Theorem 36.65 for the displayed tail carriers, then repack the tail representations with the tail classifier through the bridge definition to obtain $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u)$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier constructor decomposition`

Theorem 36.159 (List public classifier constructor exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, and let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier. The ten-field fit is: traditional target finite-list public classifier constructor readback; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability constructor exactness; ledger finite-spine constructor witnesses; exactness by* ListSpineRep *together with* ListClassifierSpec*; standard bridge finite spines; status* ProofTarget*. For all histories* $h, k : \mathsf{BHist}$*,*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \iff \begin{array}{l} (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCar} \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsam} \\ \mathsf{Rel}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u) \end{array} \right. \end{array}$$

Proof. For the forward implication, apply Theorem 36.158 to the displayed public classifier witness. This gives exactly the displayed nil-nil alternative or the displayed cons-cons constructor data.

For the reverse implication, split on the displayed disjunction. In the nil-nil branch, the nil constructor of $\mathsf{ListSpineRep}_A$ builds endpoint representations of h and k from $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. The nil constructor of ListClassifierSpec supplies the finite-spine classifier witness, and these three witnesses repack through the bridge definition of $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$.

In the cons-cons branch, unpack the witnesses $a, b, t, u, p, q : \mathsf{BHist}$ together with the source carrier, tail carrier, continuation, endpoint, head relation, and tail public classifier data. Unfold the tail public classifier to obtain finite tail spines, tail endpoint representations, and a finite-spine classifier witness.

Build the two cons spine representations using the displayed heads, tail representations, continuation histories, and endpoint links. The cons constructor of $\mathsf{ListClassifierSpec}$ combines $\mathsf{Rel}(a, b)$ with the tail finite-spine classifier. Repack the two cons endpoint representations and the cons finite-spine classifier through the bridge definition to recover $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$. □

Theorem 36.160 (List public classifier nil-cons endpoint separation)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list nil/cons endpoint separation; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from* cert *and spine coherence; ledger nil endpoint together with a displayed cons endpoint; core theorem public nil-cons endpoint separation; exactness by Theorem 36.157 and Theorem 20.6; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k, a, t, p : \mathsf{BHist}$*, the nil-to-cons orientation is impossible:*

$$\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \\ \mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{E1}(p)) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \end{array} \to \bot.$$

The cons-to-nil orientation is impossible:

$$\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{e}) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \end{array} \to \bot.$$

Proof. In the nil-to-cons orientation, apply Theorem 36.157 to $\mathsf{hsame}(h, \mathsf{e})$ and the displayed public classifier. This yields $\mathsf{hsame}(k, \mathsf{e})$. Compose the symmetry of this endpoint witness with the displayed witness $\mathsf{hsame}(k, \mathsf{E1}(p))$ to obtain $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(p))$, contradicting Theorem 20.6. In the cons-to-nil orientation, the symmetric half of the same nil inversion theorem applied to $\mathsf{hsame}(k, \mathsf{e})$ yields $\mathsf{hsame}(h, \mathsf{e})$. Composing its symmetry with $\mathsf{hsame}(h, \mathsf{E1}(p))$ again gives empty-versus-one history no-confusion. □

Theorem 36.161 (List public classifier cons branch readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list cons branch readback; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from* cert *and spine coherence; ledger one displayed cons endpoint and one classified partner; core theorem public cons branch readback; exactness by Theorem 36.160 and Theorem 36.155; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For histories* $h, k, a, t, p : \mathsf{BHist}$*, if*

$$\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \end{array}$$

holds, then

$$\exists b\, u\, q : \mathsf{BHist}. \left(\begin{array}{l} A(b) \wedge \mathsf{ListHistoryCarrier}_A(u) \wedge \mathsf{Cont}(b, u, q) \wedge \\ \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \mathsf{Rel}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u) \end{array} \right).$$

Proof. Apply Theorem 36.65 to the displayed tail carrier witness for t, choosing a finite spine xs with $\mathsf{ListSpineRep}_A(t, xs)$. The cons constructor for $\mathsf{ListSpineRep}_A$ combines this tail representation with $A(a)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$ to display h as $\mathsf{cons}(a, xs)$. Unfold the public classifier

as the spine bridge, obtaining a represented finite spine for k, and case-analyze that spine. The nil case gives $\mathsf{hsame}(k, \mathsf{e})$, contradicting the cons-to-nil orientation of Theorem 36.160 applied to the displayed cons endpoint of h and the same public classifier witness. In the cons case, inversion of the represented endpoint for k supplies histories b, u, q with $A(b)$, $\mathsf{ListHistoryCarrier}_A(u)$, $\mathsf{Cont}(b, u, q)$, and $\mathsf{hsame}(k, \mathsf{E1}(q))$. Apply Theorem 36.155 to the two displayed cons endpoints and the original public classifier witness. It yields $\mathsf{Rel}(a, b)$ and $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, u)$, which together with the displayed endpoint data give the required existential witness. □

Theorem 36.162 (List public carrier constructor readback under spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{Rel} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *be a source classifier, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$*. The ten-field fit is: traditional target finite-list carrier endpoint readback; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}$*; stability from spine coherence; ledger one generated history endpoint; core theorem public carrier nil-or-cons readback; exactness by generated carrier cases, history no-confusion, spine coverage, and classifier constructor inversion; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. For every history* $h : \mathsf{BHist}$*, if* $\mathsf{ListHistoryCarrier}_A(h)$ *holds, then*

$$\left(\begin{array}{l}\mathsf{hsame}(h, \mathsf{e}) \wedge \\ \neg \exists a\, t\, p : \mathsf{BHist}.\ \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array}\right)\end{array}\right)$$
$$\vee$$
$$\exists a\, t\, p : \mathsf{BHist}.\ \left(\begin{array}{l} A(a) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \\ \neg \mathsf{hsame}(h, \mathsf{e}) \wedge \\ \forall a'\, t'\, p' : \mathsf{BHist}.\ \left(\begin{array}{l} A(a') \wedge \mathsf{ListHistoryCarrier}_A(t') \wedge \mathsf{Cont}(a', t', p') \wedge \\ \mathsf{hsame}(h, \mathsf{E1}(p')) \end{array}\right) \to \Big(\mathsf{Rel}(a, a') \wedge \mathsf{ListSpineHi}\end{array}\right.$$

Proof. Apply Theorem 36.7 to the displayed carrier witness for h. The nil case gives $\mathsf{hsame}(h, \mathsf{e})$. If a cons presentation of the same endpoint also existed, composing the empty endpoint with the displayed $\mathsf{E1}$ endpoint would give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(p))$ for some p, contradicting Theorem 20.6 and yielding the nil alternative.

In the cons case, keep the displayed histories a, t, p with $A(a)$, $\mathsf{ListHistoryCarrier}_A(t)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$. Nonemptiness follows by the same no-confusion argument: an additional $\mathsf{hsame}(h, \mathsf{e})$ witness would compose with $\mathsf{hsame}(h, \mathsf{E1}(p))$ to contradict Theorem 20.6.

It remains to compare any alternate cons presentation $a', t', p' : \mathsf{BHist}$ of the same h. Apply Theorem 36.65 to the two tail carrier witnesses for t and t', obtaining finite tail spines xs and xs' with $\mathsf{ListSpineRep}_A(t, xs)$ and $\mathsf{ListSpineRep}_A(t', xs')$.

Build two cons spine representations of h, one over $\mathsf{cons}(a, xs)$ and one over $\mathsf{cons}(a', xs')$. The assumed $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$ classifies these two presentations by $\mathsf{ListClassifierSpec}(\mathsf{Rel}, \mathsf{cons}(a, xs), \mathsf{cons}(a', xs$

Invert this finite-spine classifier through the cons branch of Theorem 36.13. The inversion yields $\mathsf{Rel}(a, a')$ and $\mathsf{ListClassifierSpec}(\mathsf{Rel}, xs, xs')$.

Repack the two tail representations and the finite-spine tail classifier through Definition 36.57 to obtain $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t, t')$. □

Theorem 36.163 (List public-object certificate from cons-boundary coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. The ten-field per-object fit is: traditional target finite lists; BEDC source* A *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability supplied by the source certificate and* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$*; ledger* $\mathsf{ListHistoryLedger}_A$*; core theorem list Theorem 36.71 and Theorem 36.68; exactness through the spine bridge* $\mathsf{ListSpineRep}_A$*; standard bridge*

finite spines; status ProofTarget *under* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. *The semantic certificate target is*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}).$$

If $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then this semantic certificate target holds.*

Proof. Apply Theorem 36.71 to the cons-boundary coherence witness, obtaining $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. Then apply Theorem 36.68 with the source certificate cert and this spine-coherence witness. The resulting certificate is exactly the displayed semantic target for the list public object. □

Theorem 36.164 (List public-object certificate exposes endpoint spine ledgers)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. *If* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$ *holds, then the displayed public-object certificate target*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\sim_A})$$

holds. Moreover, for all histories h *and* k*, every certificate-classifier witness*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, k)$$

exposes finite-spine endpoint ledgers:

$$\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs)$$

and

$$\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k, ys).$$

Consequently,

$$\mathsf{ListHistoryCarrier}_A(h) \wedge \mathsf{ListHistoryCarrier}_A(k)$$

by Theorem 36.65. The per-object fit is: traditional target finite lists; BEDC source A *with* $\mathsf{NameCert}(A, \sim_A)$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; ledger and exactness finite-spine endpoint exposure; core certificate theorem Theorem 36.163; standard bridge finite spines; status Phase Finite Data.*

Proof. First apply Theorem 36.163 to the source certificate and the cons-boundary coherence witness; this gives the displayed semantic certificate target. For the endpoint ledger claim, unfold $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$ as the bridge classifier. A bridge witness consists of finite spines xs and ys, endpoint presentations $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$, and the finite-spine classifier witness $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. The two endpoint presentations are the required ledgers. Applying the reverse direction of Theorem 36.65 to these representation witnesses recovers $\mathsf{ListHistoryCarrier}_A(h)$ and $\mathsf{ListHistoryCarrier}_A(k)$. □

Theorem 36.165 (List public-object constructor audit)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. *Assume* $\mathsf{ListConsBoundaryCoherent}_{A,\sim_A}$. *The ten-field fit is: traditional target finite-list public constructor audit; BEDC source* BHist *with* Cont*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from the source* $\mathsf{NameCert}(A, \sim_A)$ *and cons-boundary coherence; ledger finite-spine endpoint witnesses; core theorem this constructor audit package; exactness nil/cons constructor equivalence; standard bridge finite spines; status* ProofTarget*. The semantic public-object certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\sim_A})$$

holds. Moreover, for all histories $h, k : \mathsf{BHist}$*, every certificate-classifier witness exposes endpoint spine ledgers:*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs)) \\ \wedge \\ (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k, ys)) \end{array} \right).$$

The same classifier is exactly the nil-nil or cons-cons constructor disjunction:

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k) \iff \begin{array}{l} (\mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryCar} \\ \mathsf{Cont}(a,t,p) \wedge \mathsf{Cont}(b,u,q) \wedge \mathsf{hsame}(h,\mathsf{E1}(p)) \wedge \mathsf{hsam} \\ \sim_A (a,b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(t,u) \end{array} \right. \end{array}$$

Proof. First apply Theorem 36.164 to the source certificate and the cons-boundary coherence witness. This gives the displayed semantic certificate and, for each classifier witness, the two finite-spine endpoint ledgers.

Next apply Theorem 36.159 with source classifier $\sim_A$. Its statement is exactly the displayed nil-nil or cons-cons constructor equivalence for arbitrary histories h and k.

Combining these two outputs gives the public-object certificate, its endpoint ledger witnesses, and the constructor-level classifier audit as one package. □

Theorem 36.166 (List spine bridge visible endpoint transport)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier and let* $\sim_A$ *be a source classifier. Visible endpoint transport holds for the framed list bridge classifier:*

$$\mathsf{FramedListBridgeClassifier}_{A,\sim_A}(\mathsf{E1}(h), \mathsf{E1}(k)) \to \mathsf{hsame}(h,h') \to \mathsf{hsame}(k,k') \to \mathsf{FramedListBridgeClassifier}_{A,\sim_A}(\mathsf{E1}($$

Proof. Unpack the bridge witness into finite spines, framed representations of the two visible endpoints, and the finite-spine classifier witness. Apply visible history congruence to the endpoint equalities $\mathsf{hsame}(h,h')$ and $\mathsf{hsame}(k,k')$, then transport the two framed spine representations along the resulting endpoint equalities. The finite-spine classifier witness is unchanged, so the same spines rebuild the bridge classifier at the transported visible endpoints. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier e1 endpoint hsame transport`

Definition 36.167 (Empty-bounded hsame list source)**.** *For a BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{ListHsameEmptyBoundedSource}(A)$ *by*

$$\mathsf{ListHsameEmptyBoundedSource}(A) \iff \forall a : \mathsf{BHist}.\ A(a) \to \mathsf{hsame}(a,\mathsf{e}).$$

Lean4 (definition) `BEDC.Derived.ListUp.ListHsameEmptyBoundedSource`

Theorem 36.168 (Empty-bounded equal-length list classifier)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *satisfy* $\mathsf{ListHsameEmptyBoundedSource}(A)$*. If every entry of finite spines* xs *and* ys *lies in* A*, and* $\ell(xs) = \ell(ys)$*, then*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, xs, ys)$$

holds.

Proof. Induct on xs and case-analyze ys. The mixed nil/cons cases contradict the displayed length equality. In the cons/cons case, the two head entries are both history-same to e, hence history-same to each other by symmetry and transitivity. The successor length equality gives equal tail lengths, and the induction hypothesis classifies the two tails. □

Lean4 ✓ `BEDC.Derived.ListUp.ListHsameEmptyBoundedSource equal length classifier`

Definition 36.169 (Singleton history source for lists)**.** *For a fixed BEDC history* $a_0 : \mathsf{BHist}$*, define the singleton history source carrier* $A_{a_0} : \mathsf{BHist} \to \mathsf{Prop}$ *by*

$$A_{a_0}(x) \iff \mathsf{hsame}(x, a_0).$$

Lean4 (definition) `BEDC.Derived.ListUp.ListSingletonHistorySource`

Theorem 36.170 (List singleton source framed carrier certificate)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. The ten-field per-object fit is: traditional target framed carrier certificate for finite lists over one BEDC history class; BEDC source* BHist *with* hsame*; carrier* $\mathsf{FramedListHistoryCarrier}_{A_{a_0}}$*; classifier* $\mathsf{FramedListBridgeClassifier}_{A_{a_0},\mathsf{hsame}}$*; stability from* $\mathsf{ListSourceHsameCompatible}(A_{a_0}, \mathsf{hsame})$ *and framed endpoint coherence; ledger finite framed spine; core theorem Corollary 36.128; exactness through framed spine representation; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. The framed semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{FramedListHistoryCarrier}_{A_{a_0}}, \mathsf{FramedListHistoryCarrier}_{A_{a_0}}, \mathsf{FramedListHistoryCarrier}_{A_{a_0}}, \mathsf{FramedLis}$$

hold.

Proof. Theorem 34.66 supplies the source certificate $\mathsf{NameCert}(A_{a_0}, \mathsf{hsame})$. The compatibility predicate $\mathsf{ListSourceHsameCompatible}(A_{a_0}, \mathsf{hsame})$ is immediate: the final displayed input is already the required $\mathsf{hsame}(a, b)$ witness. Apply Corollary 36.128 with the singleton source certificate and this compatibility witness. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSingletonHistorySource framed carrier certificate`

Theorem 36.171 (List singleton source spine coherence)**.** *For any BEDC history* a_0*, let* A_{a_0} *be the singleton history source of Definition 36.169. Then* $\mathsf{ListSpineRepCoherent}_{A_{a_0},\mathsf{hsame}}$ *holds.*

Proof. Compare two singleton-source spine presentations of the same endpoint by cases on the displayed spines. The mixed nil/cons cases contradict Theorem 36.58; the nil/nil case is the nil finite-spine classifier. In the cons/cons case, write the two displayed outer endpoints as $\mathsf{E1}(p)$ and $\mathsf{E1}(p')$. The common endpoint witness gives $\mathsf{hsame}(\mathsf{E1}(p), \mathsf{E1}(p'))$; Theorem 3.4 inverts it to $\mathsf{hsame}(p, p')$. The source facts $A_{a_0}(a)$ and $A_{a_0}(a')$ give $\mathsf{hsame}(a, a_0)$ and $\mathsf{hsame}(a', a_0)$, hence $\mathsf{hsame}(a, a')$ by symmetry and transitivity. Apply Theorem 4.43 to $\mathsf{Cont}(a, t, p)$, $\mathsf{Cont}(a', t', p')$, the head sameness, and the result sameness; this yields $\mathsf{hsame}(t, t')$. Transport the second tail representation along Lemma 36.151, apply the induction hypothesis to the two tail presentations at the aligned endpoint, and rebuild the $\mathsf{ListClassifierSpec}(\mathsf{hsame}, -, -)$ cons witness. □

Definition 36.172 (Hsame-bounded list source)**.** *For a source predicate* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and a center* $a_0 : \mathsf{BHist}$*, define* $\mathsf{ListHsameBoundedSource}(A, a_0)$ *by*

$$\mathsf{ListHsameBoundedSource}(A, a_0) \iff \forall a.\ A(a) \to \mathsf{hsame}(a, a_0).$$

Theorem 36.173 (Hsame-bounded list source spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and* $a_0 : \mathsf{BHist}$*. If* $\mathsf{ListHsameBoundedSource}(A, a_0)$ *holds, then* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$ *holds.*

Proof. Compare two spine presentations of a common endpoint by cases on the two spines. The nil/nil case is classified by the nil finite-spine witness. The mixed nil/cons and cons/nil cases contradict Theorem 36.58. In the cons/cons case, write the two displayed outer endpoints as $\mathsf{E1}(p)$ and $\mathsf{E1}(p')$. The common endpoint witness gives $\mathsf{hsame}(\mathsf{E1}(p), \mathsf{E1}(p'))$, and Theorem 3.4 inverts it to $\mathsf{hsame}(p, p')$. The source facts $A(a)$ and $A(a')$, together with $\mathsf{ListHsameBoundedSource}(A, a_0)$, give $\mathsf{hsame}(a, a_0)$ and $\mathsf{hsame}(a', a_0)$, hence $\mathsf{hsame}(a, a')$ by symmetry and transitivity. Apply Theorem 4.43 to $\mathsf{Cont}(a, t, p)$, $\mathsf{Cont}(a', t', p')$, the head sameness, and the result sameness to obtain $\mathsf{hsame}(t, t')$. Transport the second tail representation along Lemma 36.151, apply the induction hypothesis to the aligned tail presentations, and rebuild the cons witness for $\mathsf{ListClassifierSpec}(\mathsf{hsame}, -, -)$. □

Theorem 36.174 (Hsame-compatible bounded list source spine coherence)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$*, let* Rel *be a source classifier on* BHist*, and let* $a_0 : \mathsf{BHist}$*. If* $\mathsf{ListHsameBoundedSource}(A, a_0)$ *and* $\mathsf{ListSourceHsameCompatible}(A, \mathsf{Rel})$ *hold, then the ten-field per-object fit is: traditional target finite-list readback over a bounded source; BEDC source* BHist *with* hsame*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* Rel *lifted through* $\mathsf{ListClassifierSpec}$*; stability from bounded head comparison plus source hsame compatibility; ledger finite spine; core theorem this coherence law; exactness compatible spine readback; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. Hence*

$$\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$$

holds.

Proof. Compare two spine presentations of a common endpoint by cases on the two spines. The nil/nil case is the nil classifier. The mixed nil/cons and cons/nil cases contradict Theorem 36.58. In the cons/cons case, write the displayed outer endpoints as $\mathsf{E1}(p)$ and $\mathsf{E1}(p')$. The common endpoint witness gives $\mathsf{hsame}(\mathsf{E1}(p), \mathsf{E1}(p'))$, and Theorem 3.4 inverts it to $\mathsf{hsame}(p, p')$. The source facts $A(a)$ and $A(a')$, together with $\mathsf{ListHsameBoundedSource}(A, a_0)$, give $\mathsf{hsame}(a, a_0)$ and $\mathsf{hsame}(a', a_0)$, hence $\mathsf{hsame}(a, a')$ by symmetry and transitivity. Apply $\mathsf{ListSourceHsameCompatible}(A, \mathsf{Rel})$ to obtain $\mathsf{Rel}(a, a')$. Apply Theorem 4.43 to $\mathsf{Cont}(a, t, p)$, $\mathsf{Cont}(a', t', p')$, the head sameness, and the result sameness to obtain $\mathsf{hsame}(t, t')$. Transport the second tail representation along Lemma 36.151, apply the induction hypothesis to the aligned tail presentations, and rebuild the cons witness for $\mathsf{ListClassifierSpec}(\mathsf{Rel}, -, -)$. □

Corollary 36.175 (Hsame-compatible bounded list source unframed certificate)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$*, let* Rel *be a source classifier on* BHist*, and let* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*,* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, and* $\mathsf{ListSourceHsameCompatible}(A, \mathsf{Rel})$ *hold, then*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}})$$

holds.

Proof. Theorem 36.174 supplies $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$ from the bounded-source and source-compatibility hypotheses. Apply Theorem 36.68 with this coherence witness and the displayed source certificate. □

Theorem 36.176 (Hsame-compatible bounded list source endpoint spine ledgers)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$*, let* Rel *be a source classifier on* BHist*, and let* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*,* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, and* $\mathsf{ListSourceHsameCompatible}(A, \mathsf{Rel})$ *hold, then the ten-field fit is: traditional target finite lists over an* hsame*-bounded BEDC history source with a compatible*

source classifier; BEDC source BHist *with* NameCert(A, Rel)*; carrier* ListHistoryCarrier$_A$*; classifier* ListSpineHistoryClassifier$_{A,\mathsf{Rel}}$*; stability from bounded head comparison plus source* hsame *compatibility; ledger finite-spine endpoint exposure; core route Theorem 36.174, Theorem 36.72, and Theorem 36.164; exactness through the finite-spine endpoint representation* ListSpineRep$_A$*; standard bridge finite spines; status Phase Finite Data. The cons-boundary coherence target*

$$\mathsf{ListConsBoundaryCoherent}_{A,\mathsf{Rel}}$$

holds, and the public-object semantic certificate

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}})$$

holds. Moreover, for all histories $h, k : \mathsf{BHist}$*, every witness of*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k)$$

exposes finite spines $xs, ys : \mathsf{HistSpine}$ *with*

$$\mathsf{ListSpineRep}_A(h, xs)$$

and

$$\mathsf{ListSpineRep}_A(k, ys).$$

Consequently,

$$\mathsf{ListHistoryCarrier}_A(h) \wedge \mathsf{ListHistoryCarrier}_A(k)$$

holds.

Proof. Theorem 36.174 turns the bounded-source and source-compatibility hypotheses into ListSpineRepCoherent$_{A,\mathsf{R}}$

The forward implication of Theorem 36.72 converts this spine coherence datum into ListConsBoundaryCoherent$_{A,}$

Apply Theorem 36.164 to the source certificate cert and this cons-boundary witness. Its certificate component is exactly the displayed SemanticNameCert(ListHistoryCarrier$_A$, ListHistoryPattern$_A$, ListHistoryLedge and its endpoint-ledger component supplies the finite spines xs and ys with the two displayed ListSpineRep$_A$ facts. The final carrier conjunction follows from Theorem 36.65. □

Theorem 36.177 (Hsame-compatible bounded list source public-object constructor audit)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$*, let* Rel *be a source classifier on* BHist*, and let* $a_0 : \mathsf{BHist}$*. If* cert : NameCert(A, Rel)*,* ListHsameBoundedSource(A, a_0)*, and* ListSourceHsameCompatible(A, Rel) *hold, then the ten-field fit is: traditional target finite lists over an hsame-bounded source; BEDC source* BHist *with* NameCert(A, Rel)*; carrier* ListHistoryCarrier$_A$*; classifier* ListSpineHistoryClassifier$_{A,\mathsf{Rel}}$*; stability from bounded head comparison plus source hsame compatibility; ledger finite-spine endpoint witnesses; core theorem bounded public constructor audit; exactness nil/cons constructor equivalence; standard bridge finite spines; status* ProofTarget*. The semantic public-object certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}})$$

holds. Moreover, for all histories $h, k : \mathsf{BHist}$*, every certificate-classifier witness exposes endpoint spine ledgers:*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h, k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs)) \\ \wedge \\ (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k, ys)) \end{array} \right).$$

The same classifier is exactly the nil-nil or cons-cons constructor disjunction:

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h,k) \iff \begin{array}{l} (\mathsf{hsame}(h,\mathsf{e}) \land \mathsf{hsame}(k,\mathsf{e})) \\ \lor \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \land A(b) \land \mathsf{ListHistoryCarrier}_A(t) \land \mathsf{ListHistoryCar} \\ \mathsf{Cont}(a,t,p) \land \mathsf{Cont}(b,u,q) \land \mathsf{hsame}(h,\mathsf{E1}(p)) \land \mathsf{hsam} \\ \mathsf{Rel}(a,b) \land \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t,u) \end{array} \right. \end{array}$$

Proof. Theorem 36.174 turns the bounded-source and source-compatibility hypotheses into $\mathsf{ListSpineRepCoherent}_{A,\mathsf{R}}$

The forward implication of Theorem 36.72 converts this spine coherence datum into $\mathsf{ListConsBoundaryCoherent}_{A,}$

Apply Theorem 36.165 to the displayed source certificate and this cons-boundary coherence witness. Its three outputs are exactly the semantic certificate, the endpoint spine ledgers for each classifier witness, and the displayed nil-nil or cons-cons constructor equivalence. □

Theorem 36.178 (Hsame-compatible bounded list source finite-data exit package)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$*, let* Rel *be a source classifier on* BHist*, and let* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{Rel})$*,* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, and* $\mathsf{ListSourceHsameCompatible}(A, \mathsf{Rel})$ *hold, then the* $\mathsf{List}^{\uparrow}$ *public object carries the following finite-data exit fields.*

1. *The semantic certificate* $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpine}$
2. *Public length is total and unique on* $\mathsf{ListHistoryCarrier}_A$*.*
3. *Classified endpoints have a common public length, and any two endpoint length readbacks are equal.*
4. $\mathsf{ListPublicLength}_A(h, 0)$ *holds exactly when* $\mathsf{hsame}(h, \mathsf{e})$ *holds.*
5. *Every nonempty carrier history has a cons readback* $a, t, p : \mathsf{BHist}$ *with* $A(a)$*,* $\mathsf{ListHistoryCarrier}_A(t)$*,* $\mathsf{Cont}(a, t, p)$*,* $\mathsf{hsame}(h, \mathsf{E1}(p))$*, and no zero public length for* h*.*
6. *For every history* h *and finite length value* n*,*

$$\mathsf{ListPublicLength}_A(h, n{+}1) \iff \exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A(a) \land \mathsf{ListHistoryCarrier}_A(t) \land \mathsf{ListPublicLength}_A(t, n) \land \\ \mathsf{Cont}(a, t, p) \land \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array} \right)$$

7. *Every classifier witness exposes finite endpoint spines:*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h,k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs)) \\ \land (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k, ys)) \end{array} \right).$$

8. *The classifier is exactly the nil-nil or cons-cons constructor audit:*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(h,k) \iff \begin{array}{l} (\mathsf{hsame}(h,\mathsf{e}) \land \mathsf{hsame}(k,\mathsf{e})) \\ \lor \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \land A(b) \land \mathsf{ListHistoryCarrier}_A(t) \land \mathsf{ListHistory} \\ \mathsf{Cont}(a,t,p) \land \mathsf{Cont}(b,u,q) \land \mathsf{hsame}(h,\mathsf{E1}(p)) \land \mathsf{hs} \\ \mathsf{Rel}(a,b) \land \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{Rel}}(t,u) \end{array} \right. \end{array}$$

Proof. Theorem 36.174 supplies $\mathsf{ListSpineRepCoherent}_{A,\mathsf{Rel}}$ from the bounded-source and source-compatibility hypotheses.

Use the forward implication of Theorem 36.72 to convert this spine coherence witness into $\mathsf{ListConsBoundaryCoherent}_{A,\mathsf{Rel}}$.

Apply Theorem 36.78 to cert and the cons-boundary coherence witness. This gives the semantic certificate, public-length totality and uniqueness, the common classified endpoint length, and endpoint length equality. Apply Theorem 36.79 with the same coherence witness to obtain the zero-length characterization and the nonempty cons readback.

Theorem 36.82 supplies the successor length constructor equivalence. Finally, Theorem 36.177 supplies the endpoint spine ledgers and the nil-nil or cons-cons constructor audit with head relation $\mathsf{Rel}(a, b)$. □

Corollary 36.179 (Hsame-bounded list source unframed certificate)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$ *and* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, then the ten-field per-object fit is: traditional target finite lists over one internal history class; BEDC source* BHist *with* hsame*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from the source* $\mathsf{NameCert}(A, \mathsf{hsame})$ *and Theorem 36.173; ledger* $\mathsf{ListHistoryLedger}_A$*; core theorem list Theorem 36.173 and Theorem 36.68; exactness through the spine bridge; standard bridge finite spines; status* ProofTarget*. Hence*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}})$$

holds.

Proof. The displayed certificate cert supplies the source $\mathsf{NameCert}(A, \mathsf{hsame})$ input. The bounded-source hypothesis supplies $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$ by Theorem 36.173. Apply Theorem 36.68 with these two inputs. □

Theorem 36.180 (Hsame-bounded list source endpoint spine ledgers)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$ *and* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, then the displayed public-object semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}})$$

holds. Moreover, for all histories h *and* k*, every classifier witness*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k)$$

yields finite spines $xs, ys : \mathsf{HistSpine}$ *with*

$$\mathsf{ListSpineRep}_A(h, xs)$$

and

$$\mathsf{ListSpineRep}_A(k, ys).$$

Consequently,

$$\mathsf{ListHistoryCarrier}_A(h) \wedge \mathsf{ListHistoryCarrier}_A(k)$$

holds. The per-object fit is: traditional target finite lists over one hsame-bounded BEDC history source; BEDC source BHist *with* hsame*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; ledger finite-spine endpoint exposure; core route Theorem 36.173, Theorem 36.72, and Theorem 36.164; standard bridge finite spines; status Phase Finite Data.*

Proof. Theorem 36.173 turns the hsame-bounded source hypothesis into $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$. The forward implication of Theorem 36.72 converts this spine coherence datum into $\mathsf{ListConsBoundaryCoherent}_{A,\mathsf{hsam}}$ Apply Theorem 36.164 to the source certificate cert and this cons-boundary witness. Its certificate component is exactly the displayed $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{L}$ and its endpoint-ledger component supplies the finite spines xs and ys with the two displayed $\mathsf{ListSpineRep}_A$ facts. The final carrier conjunction follows from Theorem 36.65. □

Theorem 36.181 (Hsame-bounded list source public-object constructor audit)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and* $a_0 : \mathsf{BHist}$*. If* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$ *and* $\mathsf{ListHsameBoundedSource}(A, a_0)$*, then the public-object semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A, \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}})$$

holds. Moreover, for all histories $h, k : \mathsf{BHist}$*, every* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k)$ *witness exposes endpoint spine ledgers for* h *and* k*:*

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(h, xs)) \\ \wedge \\ (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_A(k, ys)) \end{array} \right).$$

The same classifier is exactly the nil-nil or cons-cons constructor disjunction:

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k) \iff \begin{array}{l} (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A(a) \wedge A(b) \wedge \mathsf{ListHistoryCarrier}_A(t) \wedge \mathsf{ListHistoryC} \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hs} \\ \mathsf{hsame}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(t, u) \end{array} \right. \end{array}$$

Proof. Define the needed $\mathsf{ListSourceHsameCompatible}(A, \mathsf{hsame})$ witness by taking a source-head comparison $\mathsf{hsame}(a, b)$ and returning that same $\mathsf{hsame}(a, b)$ datum.

Apply Theorem 36.177 with $\mathsf{Rel} := \mathsf{hsame}$, the displayed source certificate, the hsame-bounded source hypothesis, and this compatibility witness.

Its first projection is the displayed $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_A, \mathsf{ListHistoryPattern}_A, \mathsf{ListHistoryLedger}_A$ Its second projection gives the endpoint spine ledgers for every classifier witness, and its third projection is the displayed nil-nil or cons-cons constructor equivalence with head relation $\mathsf{hsame}(a, b)$ and recursive tail classifier $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(t, u)$. □

Theorem 36.182 (List singleton source unframed certificate)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. The ten-field per-object fit is: traditional target finite lists over one BEDC history class; BEDC source* BHist *with* hsame*; carrier* $\mathsf{ListHistoryCarrier}_{A_{a_0}}$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}$*; stability from the singleton source certificate and Theorem 36.171; ledger* $\mathsf{ListHistoryLedger}_{A_{a_0}}$*; core theorem Theorem 36.68; exactness through the spine bridge; standard bridge finite spines; status* $\mathsf{ProofTarget}$*. Then the displayed semantic certificate*

$$\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_{A_{a_0}}, \mathsf{ListHistoryPattern}_{A_{a_0}}, \mathsf{ListHistoryLedger}_{A_{a_0}}, \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsa}}$$

holds.

Proof. Theorem 34.66 supplies $\mathsf{NameCert}(A_{a_0}, \mathsf{hsame})$, and Theorem 36.171 supplies $\mathsf{ListSpineRepCoherent}_{A_{a_0},\mathsf{hsame}}$. Apply Theorem 36.68 directly with these two inputs. □

Theorem 36.183 (List singleton source finite-data certificate package)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. The singleton-source* $\mathsf{List}^{\uparrow}$ *package contains the following concrete public certificate data:*

1. $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_{A_{a_0}}, \mathsf{ListHistoryPattern}_{A_{a_0}}, \mathsf{ListHistoryLedger}_{A_{a_0}}, \mathsf{ListSpineHistoryClassifier}_{A_{a_0}}$
2. $\forall h.\ \mathsf{ListHistoryCarrier}_{A_{a_0}}(h) \to \exists n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h, n)$.

3. $\forall h\, n\, m.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to \mathsf{ListPublicLength}_{A_{a_0}}(h,m) \to n = m.$
4. $\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \to \exists n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k,n).$
5. $\forall h\, k\, n\, m.\ \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \to \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to \mathsf{ListPublicLength}_{A_{a_0}}(k,m) \to n = m.$
6. *For every history h, $\mathsf{ListPublicLength}_{A_{a_0}}(h,0)$ holds if and only if $\mathsf{hsame}(h,\mathsf{e})$ holds.*
7. *For every history h, if $\mathsf{ListHistoryCarrier}_{A_{a_0}}(h)$ and $\neg\mathsf{hsame}(h,\mathsf{e})$ hold, then*

$$\exists a\, t\, p : \mathsf{BHist}.\ \left(\begin{array}{l} A_{a_0}(a) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{Cont}(a,t,p) \wedge \\ \mathsf{hsame}(h,\mathsf{E1}(p)) \wedge \\ \forall n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to n \neq 0 \end{array}\right).$$

8. *For every history h and finite length value n,*

$$\mathsf{ListPublicLength}_{A_{a_0}}(h,n{+}1) \iff \exists a\, t\, p : \mathsf{BHist}.\ \left(\begin{array}{l} A_{a_0}(a) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(t \\ \mathsf{Cont}(a,t,p) \wedge \mathsf{hsame}(h,\mathsf{E1}(p)) \end{array}\right.$$

Proof. Theorem 34.66 supplies $\mathsf{NameCert}(A_{a_0},\mathsf{hsame})$. Theorem 36.171 supplies $\mathsf{ListSpineRepCoherent}_{A_{a_0},\mathsf{hsame}}$ for the same singleton source.

Use the forward implication of Theorem 36.72 to convert this spine coherence witness into $\mathsf{ListConsBoundaryCoherent}_{A_{a_0},\mathsf{hsame}}$.

Apply Theorem 36.78 to $\mathsf{NameCert}(A_{a_0},\mathsf{hsame})$ and the cons-boundary coherence witness. Its five components are exactly the displayed semantic certificate, totality and uniqueness of $\mathsf{ListPublicLength}_{A_{a_0}}$, the common public length for classified endpoints, and equality of endpoint readbacks.

Apply Theorem 36.79 with the same cons-boundary coherence witness. This gives the zero-length characterization and the nonempty cons readback clause.

Finally apply Theorem 36.82 to the source certificate and the cons-boundary coherence witness. The resulting successor law is the last displayed item. □

Theorem 36.184 (List singleton source finite-data exit with constructor audit)**.** *Fix a BEDC history $a_0 : \mathsf{BHist}$ and let A_{a_0} be the singleton history source of Definition 36.169. The singleton-source $\mathsf{List}^{\uparrow}$ finite-data exit object contains the following concrete public certificate data:*

1. $\mathsf{SemanticNameCert}(\mathsf{ListHistoryCarrier}_{A_{a_0}}, \mathsf{ListHistoryPattern}_{A_{a_0}}, \mathsf{ListHistoryLedger}_{A_{a_0}}, \mathsf{ListSpineHistoryClassifier}_{A_{a_0}}$
2. $\forall h.\ \mathsf{ListHistoryCarrier}_{A_{a_0}}(h) \to \exists n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n).$
3. $\forall h\, n\, m.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to \mathsf{ListPublicLength}_{A_{a_0}}(h,m) \to n = m.$
4. $\forall h\, k.\ \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \to \exists n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k,n).$
5. $\forall h\, k\, n\, m.\ \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \to \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to \mathsf{ListPublicLength}_{A_{a_0}}(k,m) \to n = m.$
6. *For every history h, $\mathsf{ListPublicLength}_{A_{a_0}}(h,0)$ holds if and only if $\mathsf{hsame}(h,\mathsf{e})$ holds.*
7. *For every history h, if $\mathsf{ListHistoryCarrier}_{A_{a_0}}(h)$ and $\neg\mathsf{hsame}(h,\mathsf{e})$ hold, then*

$$\exists a\, t\, p : \mathsf{BHist}.\ \left(\begin{array}{l} A_{a_0}(a) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{Cont}(a,t,p) \wedge \\ \mathsf{hsame}(h,\mathsf{E1}(p)) \wedge \\ \forall n.\ \mathsf{ListPublicLength}_{A_{a_0}}(h,n) \to n \neq 0 \end{array}\right).$$

8. *For every history h and finite length value n,*

$$\mathsf{ListPublicLength}_{A_{a_0}}(h, n{+}1) \iff \exists a\, t\, p : \mathsf{BHist}. \left(\begin{array}{l} A_{a_0}(a) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(t \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array} \right.$$

9. *For all histories $h, k : \mathsf{BHist}$, every classifier witness has finite spine ledgers at both endpoints:*

$$\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k) \to \left(\begin{array}{l} (\exists xs : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_{A_{a_0}}(h, xs)) \\ \wedge \\ (\exists ys : \mathsf{HistSpine}.\ \mathsf{ListSpineRep}_{A_{a_0}}(k, ys)) \end{array} \right).$$

10. *For all histories $h, k : \mathsf{BHist}$, the classifier is exactly the nil-nil or cons-cons constructor disjunction:*

$$\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k) \iff \begin{array}{l} (\mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e})) \\ \vee \\ \exists a\, b\, t\, u\, p\, q : \mathsf{BHist}. \left(\begin{array}{l} A_{a_0}(a) \wedge A_{a_0}(b) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{L} \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \\ \mathsf{hsame}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsar}} \end{array} \right. \end{array}$$

Proof. Theorem 36.183 supplies the semantic certificate, public-length totality and uniqueness, common endpoint length, endpoint readback equality, zero-length characterization, nonempty readback, and successor constructor law.

The singleton source definition gives $\mathsf{ListHsameBoundedSource}(A_{a_0}, a_0)$: from $A_{a_0}(a)$, unfold Definition 36.169 to get $\mathsf{hsame}(a, a_0)$.

Theorem 34.66 supplies $\mathsf{NameCert}(A_{a_0}, \mathsf{hsame})$. Apply Theorem 36.181 with $A := A_{a_0}$ and the displayed hsame-bounded source witness. Its endpoint projection gives the finite spine ledgers for every $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k)$ witness, and its constructor projection gives the displayed nil-nil or cons-cons equivalence specialized to A_{a_0} and hsame.

Combining the fields from the finite-data certificate package with these two projections yields the ten displayed components of the singleton-source finite-data exit object. □

36.5 Singleton source length shape exhaustion

Theorem 36.185 (List singleton source public length shape exhaustion)**.** *Fix a BEDC history $a_0 : \mathsf{BHist}$ and let A_{a_0} be the singleton history source of Definition 36.169. For every history h, if $\mathsf{ListHistoryCarrier}_{A_{a_0}}(h)$ holds, then one of the following two disjoint public length shapes is available:*

1. $\mathsf{ListPublicLength}_{A_{a_0}}(h, 0) \wedge \mathsf{hsame}(h, \mathsf{e})$.

2. *There exist a finite length value n and histories $a, t, p : \mathsf{BHist}$ such that*

$$\left(\begin{array}{l} \mathsf{ListPublicLength}_{A_{a_0}}(h, n+1) \wedge A_{a_0}(a) \wedge \\ \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(t, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \end{array} \right).$$

The nil alternative and every displayed cons alternative are disjoint by BEDC history no-confusion between e and $\mathsf{E1}(p)$.

Proof. Apply Theorem 36.183 at the fixed singleton source. Its totality clause gives a finite length value m with $\mathsf{ListPublicLength}_{A_{a_0}}(h, m)$.

Case-analyze the finite length value m.

In the zero case, the zero readback clause of Theorem 36.183 identifies $\mathsf{ListPublicLength}_{A_{a_0}}(h, 0)$ with $\mathsf{hsame}(h, \mathsf{e})$. Pair the given zero public length witness with this endpoint witness to obtain the nil alternative.

In the successor case, write $m = n + 1$. The successor exactness clause of Theorem 36.183 turns $\mathsf{ListPublicLength}_{A_{a_0}}(h, n+1)$ into histories $a, t, p : \mathsf{BHist}$ with $A_{a_0}(a)$, $\mathsf{ListHistoryCarrier}_{A_{a_0}}(t)$, $\mathsf{ListPublicLength}_{A_{a_0}}(t, n)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$. These data are exactly the displayed cons alternative.

For disjointness, suppose the nil alternative and one displayed cons alternative both hold. Compose symmetry of the nil endpoint witness $\mathsf{hsame}(h, \mathsf{e})$ with the cons endpoint witness $\mathsf{hsame}(h, \mathsf{E1}(p))$ to obtain $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(p))$. The BEDC history no-confusion theorem, Theorem 20.6, excludes this empty-versus-one identification. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListHistoryCarrier public length shape exhaustion`

Theorem 36.186 (List singleton source classified shape synchrony)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. For all histories* $h, k : \mathsf{BHist}$*, if* $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k)$ *holds, then exactly one synchronized public endpoint shape is available:*

1.

$$\mathsf{ListPublicLength}_{A_{a_0}}(h, 0) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k, 0) \wedge \mathsf{hsame}(h, \mathsf{e}) \wedge \mathsf{hsame}(k, \mathsf{e}).$$

2. There exist a finite length value n *and histories* $a, b, t, u, p, q : \mathsf{BHist}$ *such that*

$$\left(\begin{array}{l} \mathsf{ListPublicLength}_{A_{a_0}}(h, n+1) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k, n+1) \wedge \\ A_{a_0}(a) \wedge A_{a_0}(b) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(u) \wedge \\ \mathsf{ListPublicLength}_{A_{a_0}}(t, n) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(u, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ \mathsf{hsame}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t, u) \end{array}\right).$$

The nil endpoint shape and every displayed cons endpoint shape are mutually exclusive by BEDC history no-confusion between e *and* $\mathsf{E1}(p)$*.*

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier singleton classified shape synchrony`

Proof. Apply Theorem 36.184 to the fixed singleton source. For the given classifier witness, its public data give a common endpoint length, public-length uniqueness, successor exactness, finite endpoint spine ledgers, and the nil-nil or cons-cons constructor disjunction.

In the nil-nil constructor case, the zero-length characterization in the same certificate turns $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$ into $\mathsf{ListPublicLength}_{A_{a_0}}(h, 0)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(k, 0)$. Pair these two public lengths with the two endpoint witnesses to obtain the first alternative.

In the cons-cons constructor case, the constructor disjunction supplies histories $a, b, t, u, p, q : \mathsf{BHist}$ with both heads in A_{a_0}, both tails in $\mathsf{ListHistoryCarrier}_{A_{a_0}}$, both Cont ledgers, both endpoint readbacks, $\mathsf{hsame}(a, b)$, and the tail classifier $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t, u)$.

Apply the common endpoint length clause of Theorem 36.184 to the tail classifier. This gives a finite value n with $\mathsf{ListPublicLength}_{A_{a_0}}(t, n)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(u, n)$. The reverse direction of

successor exactness transports the two Cont ledgers and these tail lengths to $\mathsf{ListPublicLength}_{A_{a_0}}(h, n+1)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(k, n+1)$.

The endpoint spine ledgers supplied by the certificate give the carrier presentations needed to apply Theorem 36.185 to the classified endpoints h and k. Apply the same theorem to the exposed tails t and u using their carrier witnesses. These four one-endpoint applications attach the public readback shape witnesses to the data supplied by the constructor audit.

In the present cons-cons branch, endpoint nil alternatives would identify e with $\mathsf{E1}(p)$ or with $\mathsf{E1}(q)$ after symmetry and transitivity of hsame, so Theorem 20.6 excludes them. Public-length uniqueness then identifies the resulting endpoint successor lengths with the $n+1$ lengths already obtained from the tail classifier, while the displayed tail public lengths remain the common length data for t and u.

Collect the common length n and the six constructor histories. The displayed data are exactly the second alternative. The same no-confusion argument separates the nil endpoint shape from any cons endpoint shape. □

Theorem 36.187 (List singleton source classified zero length iff empty)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. For all histories* $h, k : \mathsf{BHist}$*, the following conditions are equivalent:*

$$\left(\begin{array}{l}\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k)\wedge\\ \mathsf{ListPublicLength}_{A_{a_0}}(h,0)\wedge\mathsf{ListPublicLength}_{A_{a_0}}(k,0)\end{array}\right) \iff (\mathsf{hsame}(h,\mathsf{E0}(\mathsf{e}))\wedge\mathsf{hsame}(k,\mathsf{E0}(\mathsf{e})))\,.$$

Proof. For the forward direction, start from the classifier witness and the two zero public-length witnesses. The zero-length characterization in Theorem 36.183 turns the h-side witness into $\mathsf{hsame}(h, \mathsf{E0}(\mathsf{e}))$ and the k-side witness into $\mathsf{hsame}(k, \mathsf{E0}(\mathsf{e}))$.

The two endpoint comparisons are exactly the displayed empty endpoint conjunction. The classifier witness is used only to keep the statement tied to classified singleton-source endpoints.

For the reverse direction, assume $\mathsf{hsame}(h, \mathsf{E0}(\mathsf{e})) \wedge \mathsf{hsame}(k, \mathsf{E0}(\mathsf{e}))$. Apply the reverse zero-length characterization from Theorem 36.183 to obtain $\mathsf{ListPublicLength}_{A_{a_0}}(h, 0)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(k, 0)$.

It remains to rebuild the endpoint classifier. Use the nil-nil branch of the constructor-disjunction field in Theorem 36.184; the two empty endpoint comparisons are precisely the nil-nil data. Together with the two reconstructed zero public-length witnesses, this gives the left-hand condition. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier singleton zero public length iff empty`

Theorem 36.188 (List singleton source classified successor length inversion)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. For all histories* $h, k : \mathsf{BHist}$ *and every finite length value* n*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k)\wedge\mathsf{ListPublicLength}_{A_{a_0}}(h,n{+}1)\wedge\mathsf{ListPublicLength}_{A_{a_0}}(k,n{+}1),$$

then there exist histories $a, b, t, u, p, q : \mathsf{BHist}$ *such that*

$$\left(\begin{array}{l}A_{a_0}(a)\wedge A_{a_0}(b)\wedge\mathsf{ListHistoryCarrier}_{A_{a_0}}(t)\wedge\mathsf{ListHistoryCarrier}_{A_{a_0}}(u)\wedge\\ \mathsf{ListPublicLength}_{A_{a_0}}(t,n)\wedge\mathsf{ListPublicLength}_{A_{a_0}}(u,n)\wedge\\ \mathsf{Cont}(a,t,p)\wedge\mathsf{Cont}(b,u,q)\wedge\mathsf{hsame}(h,\mathsf{E1}(p))\wedge\mathsf{hsame}(k,\mathsf{E1}(q))\wedge\\ \mathsf{hsame}(a,b)\wedge\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t,u)\end{array}\right).$$

Proof. Apply Theorem 36.186 to the classifier witness. It gives the synchronized nil endpoint shape or a synchronized cons endpoint shape.

In the nil endpoint shape, the theorem supplies $\mathsf{ListPublicLength}_{A_{a_0}}(h, 0)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(k, 0)$. Public-length uniqueness from Theorem 36.183 identifies these zero witnesses with the displayed successor witnesses $\mathsf{ListPublicLength}_{A_{a_0}}(h, n+1)$ and $\mathsf{ListPublicLength}_{A_{a_0}}(k, n+1)$. The zero-length characterization from the same certificate is incompatible with a successor public endpoint, so the nil branch is empty.

In the cons endpoint shape, the synchrony theorem supplies a finite value m and histories $a, b, t, u, p, q : \mathsf{BHist}$ with the two endpoint public lengths at $m + 1$, both singleton-source heads, both carrier tails, both tail public lengths at m, the two Cont ledgers, the two endpoint readbacks, $\mathsf{hsame}(a, b)$, and the tail classifier $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t, u)$.

Apply public-length uniqueness from Theorem 36.183 to the public length witnesses for h at $m + 1$ and at $n + 1$. The same argument for k gives the identical successor index comparison, and successor injectivity for finite length values identifies m with n.

Transport the two tail public-length witnesses along this length equality. Keeping the constructor histories and all ledger, endpoint, head, and classifier witnesses supplied by the cons shape gives exactly the displayed six-history package. □

Theorem 36.189 (List singleton source classified successor component exactness)**.** *Fix a BEDC history $a_0 : \mathsf{BHist}$ and let A_{a_0} be the singleton history source of Definition 36.169. For all histories $h, k : \mathsf{BHist}$ and every finite length value n, the following conditions are equivalent:*

1. $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k)$, $\mathsf{ListPublicLength}_{A_{a_0}}(h, n+1)$, *and* $\mathsf{ListPublicLength}_{A_{a_0}}(k, n+1)$ *all hold.*
2. *There exist histories $a, b, t, u, p, q : \mathsf{BHist}$ such that*

$$\left(\begin{array}{l} A_{a_0}(a) \wedge A_{a_0}(b) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(u) \wedge \\ \mathsf{ListPublicLength}_{A_{a_0}}(t, n) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(u, n) \wedge \\ \mathsf{Cont}(a, t, p) \wedge \mathsf{Cont}(b, u, q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ \mathsf{hsame}(a, b) \wedge \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t, u) \end{array}\right).$$

Proof. For the forward direction, apply Theorem 36.188 to the classifier witness and the two displayed successor public lengths. The result is exactly the six-history component package in the second condition.

For the reverse direction, start from histories $a, b, t, u, p, q : \mathsf{BHist}$ and the displayed component data. The successor-length equivalence field of Theorem 36.184 applied to h rebuilds $\mathsf{ListPublicLength}_{A_{a_0}}(h, n + 1)$ from a, t, p, the tail public length at n, the $\mathsf{Cont}(a, t, p)$ ledger, and the endpoint readback $\mathsf{hsame}(h, \mathsf{E1}(p))$.

Apply the same successor-length field to k with b, u, q. This rebuilds $\mathsf{ListPublicLength}_{A_{a_0}}(k, n+1)$ from the second tail public length, the second Cont ledger, and the endpoint readback through $\mathsf{E1}(q)$.

It remains to reconstruct the endpoint classifier. Use the cons branch in the constructor-disjunction field of Theorem 36.184 with the same six histories. The supplied head membership, tail carriers, ledgers, endpoint readbacks, head comparison, and tail classifier satisfy its displayed premises, so it yields $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h, k)$.

Together these three reconstructed witnesses give the first condition. □

Theorem 36.190 (List singleton source classified successor endpoint nonempty)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. For all histories* $h, k : \mathsf{BHist}$ *and every finite length value* n*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(h,n{+}1) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k,n{+}1),$$

then

$$\neg\mathsf{hsame}(h,\mathsf{e}) \wedge \neg\mathsf{hsame}(k,\mathsf{e}).$$

The ten-field fit is: traditional target finite-list successor endpoint exclusion; BEDC source BHist *with singleton source* A_{a_0}*; carrier* $\mathsf{ListHistoryCarrier}_{A_{a_0}}$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}$*; stability from public successor length inversion; ledger cons endpoint readbacks through* $\mathsf{E1}(p)$ *and* $\mathsf{E1}(q)$*; core theorem this endpoint exclusion; exactness by BEDC history no-confusion; standard bridge finite spines; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier public successor endpoint nonempty`

Proof. Apply Theorem 36.188 to the classifier witness and the two successor public-length witnesses. It gives histories $a, b, t, u, p, q : \mathsf{BHist}$, in particular the endpoint readbacks $\mathsf{hsame}(h, \mathsf{E1}(p))$ and $\mathsf{hsame}(k, \mathsf{E1}(q))$.

For the first exclusion, suppose $\mathsf{hsame}(h, \mathsf{e})$. Symmetry gives $\mathsf{hsame}(\mathsf{e}, h)$, and transitivity with the readback $\mathsf{hsame}(h, \mathsf{E1}(p))$ gives $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(p))$. The BEDC history no-confusion theorem, Theorem 20.6, rules out this empty-versus-one identification.

The second exclusion is identical with k and q. From $\mathsf{hsame}(k, \mathsf{e})$, symmetry and transitivity with $\mathsf{hsame}(k, \mathsf{E1}(q))$ give $\mathsf{hsame}(\mathsf{e}, \mathsf{E1}(q))$, again contradicting Theorem 20.6. These two exclusions form the displayed conjunction. □

Theorem 36.191 (List singleton source classified double-successor tail nonempty)**.** *Fix a BEDC history* $a_0 : \mathsf{BHist}$ *and let* A_{a_0} *be the singleton history source of Definition 36.169. For all histories* $h, k : \mathsf{BHist}$ *and every finite length value* n*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(h,k) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(h,(n{+}1){+}1) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(k,(n{+}1){+}1),$$

then there exist histories $a, b, t, u, p, q : \mathsf{BHist}$ *such that*

$$\left(\begin{array}{l} A_{a_0}(a) \wedge A_{a_0}(b) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(t) \wedge \mathsf{ListHistoryCarrier}_{A_{a_0}}(u) \wedge \\ \mathsf{ListPublicLength}_{A_{a_0}}(t, n+1) \wedge \mathsf{ListPublicLength}_{A_{a_0}}(u, n+1) \wedge \\ \mathsf{Cont}(a,t,p) \wedge \mathsf{Cont}(b,u,q) \wedge \mathsf{hsame}(h, \mathsf{E1}(p)) \wedge \mathsf{hsame}(k, \mathsf{E1}(q)) \wedge \\ \mathsf{hsame}(a,b) \wedge \mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t,u) \wedge \\ \neg\mathsf{hsame}(t,\mathsf{e}) \wedge \neg\mathsf{hsame}(u,\mathsf{e}) \end{array}\right).$$

The ten-field fit is: traditional target finite-list two-step constructor readback; BEDC source singleton BHist*; carrier* ListHistoryCarrier*; classifier* ListSpineHistoryClassifier*; stability from successor component exactness; ledger exposed cons endpoints and tail public lengths; core theorem this block; exactness by endpoint no-confusion through the cited nonempty theorem; standard bridge finite spines; status* ProofTarget*.*

Proof. Apply Theorem 36.189 at length $n{+}1$ to the classifier witness and the two displayed double-successor public-length witnesses. The forward direction supplies histories $a, b, t, u, p, q : \mathsf{BHist}$ with the singleton heads, tail carriers, tail public lengths at $n + 1$, the two Cont ledgers, the endpoint readbacks, the head comparison, and the tail classifier $\mathsf{ListSpineHistoryClassifier}_{A_{a_0},\mathsf{hsame}}(t, u)$.

Apply Theorem 36.190 to this tail classifier and the two tail public-length witnesses at $n+1$. It gives $\neg\mathsf{hsame}(t,\mathsf{e}) \wedge \neg\mathsf{hsame}(u,\mathsf{e})$.

Keeping the same six histories and appending these two tail exclusions to the successor component package gives the displayed existential conclusion. □

Theorem 36.192 (List represented-spine append-context cancellation)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$, and assume $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. For finite BEDC-history spines $p, p', x, y, s, s' : \mathsf{HistSpine}$ and histories h, k, m, n, suppose*

$$\mathsf{ListClassifierSpec}(\sim_A, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\sim_A, s, s')$$

hold. If

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(x{+}{+}s)), \quad \mathsf{ListSpineRep}_A(k, p'{+}{+}(y{+}{+}s')),$$

and

$$\mathsf{ListSpineRep}_A(m, x), \quad \mathsf{ListSpineRep}_A(n, y)$$

hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k) \iff \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(m,n).$$

Proof. For the forward implication, apply Theorem 36.145 to the represented contextual endpoints h and k. This turns the public classifier witness into

$$\mathsf{ListClassifierSpec}(\sim_A, p{+}{+}(x{+}{+}s), p'{+}{+}(y{+}{+}s')).$$

The classified prefix and suffix hypotheses allow Theorem 36.143 to cancel the displayed finite-spine context, giving $\mathsf{ListClassifierSpec}(\sim_A, x, y)$. Applying the reverse direction of Theorem 36.145 to the represented middle endpoints m and n repacks that witness as $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(m,n)$.

For the reverse implication, apply Theorem 36.145 to the represented middle endpoints m and n. This reads the public middle classifier as $\mathsf{ListClassifierSpec}(\sim_A, x, y)$.

Use Theorem 36.143 in the opposite direction with the same classified prefixes and suffixes to obtain

$$\mathsf{ListClassifierSpec}(\sim_A, p{+}{+}(x{+}{+}s), p'{+}{+}(y{+}{+}s')).$$

The reverse direction of Theorem 36.145 for the represented contextual endpoints h and k repacks the displayed contextual classifier as $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,k)$. □

36.6 Public append readback

Definition 36.193 (List public append readback)**.** *For a BEDC history source carrier $A : \mathsf{BHist} \to \mathsf{Prop}$, define $\mathsf{ListPublicAppend}_A(h,k,r)$ to mean that there exist finite BEDC-history spines $xs, ys : \mathsf{HistSpine}$ such that*

$$\mathsf{ListSpineRep}_A(h, xs), \quad \mathsf{ListSpineRep}_A(k, ys),$$

and

$$\mathsf{ListSpineRep}_A(r, xs{+}{+}ys)$$

hold.

Lemma 36.194 (List spine append representation construction)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For finite BEDC-history spines* $xs, ys : \mathsf{HistSpine}$ *and histories* h, k*, if*

$$\mathsf{ListSpineRep}_A(h, xs) \quad \textit{and} \quad \mathsf{ListSpineRep}_A(k, ys)$$

hold, then there exists a history $r : \mathsf{BHist}$ *such that*

$$\mathsf{ListSpineRep}_A(r, xs{+}{+}ys).$$

Proof. Proceed by induction on the displayed representation $\mathsf{ListSpineRep}_A(h, xs)$.

In the nil case, the displayed spine xs is nil, so $xs{+}{+}ys$ is ys. Reuse the endpoint k together with the given representation $\mathsf{ListSpineRep}_A(k, ys)$.

In the cons case, write $xs = \mathsf{cons}(a, xs_0)$ and let the head constructor provide $A(a)$, a tail endpoint t, a tail representation $\mathsf{ListSpineRep}_A(t, xs_0)$, and a constructor ledger $\mathsf{Cont}(a, t, h)$. The induction hypothesis applied to the tail representation and the endpoint representation of k gives a history r_0 represented by $xs_0{+}{+}ys$.

Use the same head payload a and the constructed tail endpoint r_0 to form a BEDC constructor endpoint r with $\mathsf{Cont}(a, r_0, r)$. The cons constructor for $\mathsf{ListSpineRep}_A$ then represents r by $\mathsf{cons}(a, xs_0{+}{+}ys)$, which is $xs{+}{+}ys$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListSpineRep append representation`

Theorem 36.195 (List public append readback is total and classifier-congruent)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The public append readback relation satisfies:*

1. *if* $\mathsf{ListHistoryCarrier}_A(h)$ *and* $\mathsf{ListHistoryCarrier}_A(k)$ *hold, then there exists a history* $r : \mathsf{BHist}$ *such that*

$$\mathsf{ListPublicAppend}_A(h, k, r) \quad \textit{and} \quad \mathsf{ListHistoryCarrier}_A(r)$$

   hold;

2. *if* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, h')$*,* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k, k')$*,* $\mathsf{ListPublicAppend}_A(h, k, r)$*, and* $\mathsf{ListPublicAppend}_A(h', k', r')$ *hold, then*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(r, r')$$

   holds;

3. *if* $\mathsf{ListPublicAppend}_A(h, k, r)$ *and* $\mathsf{ListPublicAppend}_A(h, k, r')$ *hold, then*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(r, r')$$

   holds.

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicAppend total and classifier congruent`

Proof. For totality, apply Theorem 36.65 to the two public carrier witnesses. This gives finite spines xs and ys with $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, ys)$.

Apply Lemma 36.194 to construct an endpoint r represented by $xs{+}{+}ys$. This representation is precisely the third field of $\mathsf{ListPublicAppend}_A(h, k, r)$, and Theorem 36.65 gives $\mathsf{ListHistoryCarrier}_A(r)$.

For classifier congruence, unpack the two append readbacks as spines xs, ys and xs', ys'. Displayed-spine exactness reads the public classifiers on h, h' and on k, k' as

$$\mathsf{ListClassifierSpec}(\sim_A, xs, xs') \quad \text{and} \quad \mathsf{ListClassifierSpec}(\sim_A, ys, ys').$$

Theorem 36.34 combines these two finite-spine classifiers into $\mathsf{ListClassifierSpec}(\sim_A, xs{+}{+}ys, xs'{+}{+}ys')$. Repack that displayed classifier through the two append endpoint representations by the reverse direction of Theorem 36.145, obtaining $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(r, r')$.

For two outputs over the same inputs, unpack both append readbacks as xs, ys and xs', ys'. Coherence for the two presentations of h gives $\mathsf{ListClassifierSpec}(\sim_A, xs, xs')$, and coherence for the two presentations of k gives $\mathsf{ListClassifierSpec}(\sim_A, ys, ys')$.

Apply Theorem 36.34 and then displayed-spine exactness exactly as in the congruence argument. This classifies the two append outputs r and r'. □

Theorem 36.196 (List public append readback exposes unary-length addition)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. For histories* $h, k, r : \mathsf{BHist}$*, if*

$$\mathsf{ListPublicAppend}_A(h, k, r)$$

holds, then there exist histories $u, v, w : \mathsf{BHist}$ *such that*

1. $\mathsf{ListPublicUnaryLength}_A(h, u)$*,* $\mathsf{ListPublicUnaryLength}_A(k, v)$*, and* $\mathsf{ListPublicUnaryLength}_A(r, w)$ *hold;*
2. $$\mathsf{NatAdd}(v, u, w);$$
3. *for every* $z : \mathsf{BHist}$*,* $$\mathsf{NatAdd}(u, v, z) \to \mathsf{hsame}(w, z).$$

Proof. Unpack $\mathsf{ListPublicAppend}_A(h, k, r)$ as finite BEDC-history spines $xs, ys : \mathsf{HistSpine}$ together with representations of h by xs, of k by ys, and of r by $xs{+}{+}ys$.

Choose

$$u = \mathsf{ListUnaryLength}(xs), \qquad v = \mathsf{ListUnaryLength}(ys), \qquad w = \mathsf{ListUnaryLength}(xs{+}{+}ys).$$

The three public unary-length witnesses follow by pairing the same spine representations with reflexivity of hsame in Definition 36.91.

Apply Theorem 36.95 to the three displayed representations. Its append-aligned clause gives $\mathsf{NatAdd}(v, u, w)$, and its commuted uniqueness clause gives, for every $z : \mathsf{BHist}$, the implication $\mathsf{NatAdd}(u, v, z) \to \mathsf{hsame}(w, z)$. □

Theorem 36.197 (List classifier mixed-tag append-context exactness)**.** *For finite BEDC-history spines* p, p', x, y, s, s'*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. Then mixed visible constructor tags are classified by the finite-spine classifier exactly in the empty middle case:

1. $$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e1, y){+}{+}s')) \iff x = \mathsf{nil} \land y = \mathsf{nil};$$
2. $$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e1, x){+}{+}s), p'{+}{+}(\mathsf{map}(e0, y){+}{+}s')) \iff x = \mathsf{nil} \land y = \mathsf{nil}.$$

Proof. For the $e0/e1$ forward implication, apply Theorem 36.143 with the classified prefixes and suffixes. This cancels the displayed context and leaves

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e0, x), \mathsf{map}(e1, y)).$$

Theorem 36.29 then forces $x = \mathsf{nil}$ and $y = \mathsf{nil}$.

For the $e0/e1$ reverse implication, assume $x = \mathsf{nil}$ and $y = \mathsf{nil}$. The mapped middle spines are both nil, so the nil clause of Definition 36.12 gives the displayed nil classifier. The reverse direction of Theorem 36.143 rebuilds the classified prefix and suffix context.

For the $e1/e0$ forward implication, first cancel the displayed prefix and suffix context by Theorem 36.143. Apply Theorem 36.17 to reverse the resulting middle classifier, obtaining

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e0, y), \mathsf{map}(e1, x)).$$

Theorem 36.29 gives $y = \mathsf{nil}$ and $x = \mathsf{nil}$, hence the required conjunction.

For the $e1/e0$ reverse implication, the assumptions again make both mapped middle spines nil. Use the nil classifier and rebuild the displayed context by the reverse direction of Theorem 36.143. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame mixed tag append context empty iff`

Theorem 36.198 (List represented-spine mixed-tag context separation)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines* $p, p', x, y, s, s' : \mathsf{HistSpine}$ *and histories* h, k*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. If

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(\mathsf{map}(e0, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(k, p'{+}{+}(\mathsf{map}(e1, y){+}{+}s')),$$

hold, and if

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k)$$

holds, then $x = \mathsf{nil}$ *and* $y = \mathsf{nil}$*.*

The ten-field fit is: traditional target finite-list tagged-branch separation; BEDC source A *with* $\mathsf{NameCert}(A, \mathsf{hsame})$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from spine coherence, append-context cancellation, and history constructor no-confusion; ledger visible* $e0/e1$ *branch tags in finite spines; core theorem mixed-tag context separation; exactness by displayed-spine classifier exactness; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Theorem 36.145 to the represented contextual endpoints h and k. The public classifier therefore reads as

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e1, y){+}{+}s')).$$

Now apply Theorem 36.197. Its $e0/e1$ forward implication uses the classified prefix and suffix hypotheses to force $x = \mathsf{nil}$ and $y = \mathsf{nil}$. □

Theorem 36.199 (List represented-spine mixed-tag context exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines* $p, p', x, y, s, s' : \mathsf{HistSpine}$*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. Then mixed visible constructor tags are classified exactly in the empty middle case:

1. *for histories* h_0, k_0*, if*

$$\mathsf{ListSpineRep}_A(h_0, p{+}{+}(\mathsf{map}(e0, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(k_0, p'{+}{+}(\mathsf{map}(e1, y){+}{+}s'))$$

   hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_0, k_0) \iff x = \mathsf{nil} \wedge y = \mathsf{nil};$$

2. *for histories* h_1, k_1*, if*

$$\mathsf{ListSpineRep}_A(h_1, p{+}{+}(\mathsf{map}(e1, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(k_1, p'{+}{+}(\mathsf{map}(e0, y){+}{+}s'))$$

   hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_1, k_1) \iff x = \mathsf{nil} \wedge y = \mathsf{nil}.$$

The ten-field fit is: traditional target finite-list mixed visible-tag exactness; BEDC source A *with* $\mathsf{NameCert}(A, \mathsf{hsame})$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from represented-spine coherence plus append-context cancellation; ledger visible* $e0/e1$ *tags; core theorem mixed-tag exactness; exactness by displayed-spine readback; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$*.*

Proof. For the $e0/e1$ forward implication, apply Theorem 36.145 to the represented contextual endpoints h_0, k_0. The public classifier becomes the displayed contextual classifier, and the $e0/e1$ forward implication of Theorem 36.197 forces $x = \mathsf{nil}$ and $y = \mathsf{nil}$.

For the $e0/e1$ reverse implication, assume $x = \mathsf{nil}$ and $y = \mathsf{nil}$. The $e0/e1$ reverse implication of Theorem 36.197 gives

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e1, y){+}{+}s')).$$

The reverse direction of Theorem 36.145 repacks the displayed contextual classifier through the represented endpoints, producing $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_0, k_0)$.

For the $e1/e0$ forward implication, displayed exactness turns a public classifier on h_1, k_1 into the corresponding contextual finite-spine classifier. The $e1/e0$ forward implication of Theorem 36.197 then gives $x = \mathsf{nil}$ and $y = \mathsf{nil}$.

For the $e1/e0$ reverse implication, the assumptions $x = \mathsf{nil}$ and $y = \mathsf{nil}$ feed the $e1/e0$ reverse implication of Theorem 36.197. The resulting displayed contextual classifier is repacked by represented-spine displayed exactness for the represented endpoints h_1, k_1. □

Theorem 36.200 (List represented-spine mixed-tag same-history emptiness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines* $p, p', x, y, s, s' : \mathsf{HistSpine}$ *and a history* h*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. Then mixed visible constructor tags around two represented-spine readbacks of the same history force both untagged middle payloads to be empty:

1. *if*

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(\mathsf{map}(e0, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(h, p'{+}{+}(\mathsf{map}(e1, y){+}{+}s'))$$

hold, then

$$x = \mathsf{nil} \wedge y = \mathsf{nil};$$

2. *if*

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(\mathsf{map}(e1, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(h, p'{+}{+}(\mathsf{map}(e0, y){+}{+}s'))$$

hold, then

$$x = \mathsf{nil} \wedge y = \mathsf{nil}.$$

The ten-field fit is: traditional target finite-list same-history mixed visible tag no-confusion; BEDC source A with $\mathsf{NameCert}(A, \mathsf{hsame})$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from spine coherence and certificate reflexivity; ledger visible $e0/e1$ branch tags in finite spines; core theorem mixed-tag context exactness at identical endpoints; exactness by represented-spine carrier coverage and public classifier reflexivity; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$.

Proof. Start with either represented-spine presentation of h. By Theorem 36.61, the presentation exposes $\mathsf{ListHistoryCarrier}_A(h)$.

Apply Theorem 36.68 to cert and the coherence witness. The resulting semantic certificate supplies classifier reflexivity for the carrier witness, hence

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, h).$$

For the $e0/e1$ clause, specialize Theorem 36.199 to the two endpoints h and h, using the displayed mixed-tag presentations, the classified prefix and suffix hypotheses, and the reflexive public classifier above. Project the forward implication of the $e0/e1$ branch to obtain $x = \mathsf{nil} \wedge y = \mathsf{nil}$.

The $e1/e0$ clause is the symmetric specialization with the visible tags interchanged. Project the forward implication of that branch, again using the public reflexivity of h, to obtain $x = \mathsf{nil} \wedge y = \mathsf{nil}$. □

Theorem 36.201 (List classifier $e0$ append-context equivalence)**.** *For finite BEDC-history spines p, p', x, y, s, s', suppose the prefix contexts p, p' are structurally classified by* hsame *and the suffix contexts s, s' are structurally classified by* hsame*. Then*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e0, y){+}{+}s')) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y).$$

Proof. In the forward direction, append-context cancellation removes the classified prefixes and suffixes, leaving the $e0$-mapped middle classifier. The $e0$ map-inversion theorem reflects it to the untagged middle classifier. In the reverse direction, $e0$ map stability lifts the middle classifier to the tagged middle spines, and append-context rebuilding restores the classified prefix and suffix contexts. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e0 append context iff`

Theorem 36.202 (List classifier same-tag append-context equivalence)**.** *For finite BEDC-history spines* p, p', x, y, s, s'*, suppose* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$ *and* $\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$*. Then both same-tag contextual classifiers expose exactly the untagged middle classifier:*

1.

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e0, y){+}{+}s')) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y);$$

2.

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e1, x){+}{+}s), p'{+}{+}(\mathsf{map}(e1, y){+}{+}s')) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y).$$

The ten-field fit is: traditional target same-tag finite-spine contextual readback; BEDC source BHist*; carrier finite BEDC-history spines; classifier* $\mathsf{ListClassifierSpec}(\mathsf{hsame})$*; stability from constructor-map stability, constructor-map inversion, and append-context cancellation; ledger visible same-tag finite-spine contexts; core theorem same-tag append-context equivalence; exactness the displayed* $e0$ *and* $e1$ *iff clauses; downstream consumer Theorem 36.203; status* $\mathsf{ProofTarget}$*.*

Proof. For the $e0$ clause, apply Theorem 36.201. Its hypotheses are exactly the classified prefix and suffix assumptions, and its conclusion is the displayed $e0$ contextual equivalence.

For the $e1$ forward direction, first apply Theorem 36.143 with the classified prefixes and suffixes. This cancels the surrounding context and leaves $\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e1, x), \mathsf{map}(e1, y))$.

Theorem 36.27 reflects the $e1$-mapped middle classifier back to $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$.

For the $e1$ reverse direction, start from $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$. Theorem 36.26 transports it to $\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e1, x), \mathsf{map}(e1, y))$.

Finally, the reverse direction of Theorem 36.143 rebuilds the classified prefix and suffix context, giving the displayed contextual classifier. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1 append context iff`

Theorem 36.203 (List represented-spine same-tag context exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines* $p, p', x, y, s, s' : \mathsf{HistSpine}$*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. Then same visible constructor tags expose exactly the untagged payload spine classifier:

1. for histories h_0, k_0*, if*

$$\mathsf{ListSpineRep}_A(h_0, p{+}{+}(\mathsf{map}(e0, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(k_0, p'{+}{+}(\mathsf{map}(e0, y){+}{+}s'))$$

hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_0, k_0) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y);$$

2. for histories h_1, k_1, if

$$\mathsf{ListSpineRep}_A(h_1, p{+}{+}(\mathsf{map}(e1, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(k_1, p'{+}{+}(\mathsf{map}(e1, y){+}{+}s'))$$

hold, then

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_1, k_1) \iff \mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y).$$

The ten-field fit is: traditional target finite-list same-tag payload readback; BEDC source A with $\mathsf{NameCert}(A, \mathsf{hsame})$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from spine coherence, constructor-map stability, and append-context rebuilding; ledger visible $e0/e1$ branch tags in finite spines; core theorem same-tag context exactness; exactness by displayed-spine classifier readback and constructor-map inversion; standard bridge finite-spine readback; status* $\mathsf{ProofTarget}$.

Proof. For the $e0$ forward implication, apply Theorem 36.145 to the two represented contextual endpoints. This reads the public classifier as

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{map}(e0, x){+}{+}s), p'{+}{+}(\mathsf{map}(e0, y){+}{+}s')).$$

Apply Theorem 36.143 with the classified prefixes and suffixes to cancel the displayed context. The result is $\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e0, x), \mathsf{map}(e0, y))$, and Theorem 36.25 reflects it to $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$.

For the $e0$ reverse implication, start with $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$. Theorem 36.24 transports it to the mapped middle spines, and the reverse direction of Theorem 36.143 rebuilds the classified prefix and suffix context.

The reverse direction of Theorem 36.145 then repacks the displayed contextual classifier as $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h_0, k_0)$.

The $e1$ clause is identical with $e1$ in place of $e0$. Its forward direction uses displayed exactness and append-context cancellation to obtain $\mathsf{ListClassifierSpec}(\mathsf{hsame}, \mathsf{map}(e1, x), \mathsf{map}(e1, y))$, then applies Theorem 36.27. Its reverse direction applies Theorem 36.26, rebuilds the classified context, and repacks through displayed exactness for the represented endpoints h_1, k_1. □

Theorem 36.204 (List represented-spine same-tag payload determinacy)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines $p, p', x, y, s, s' : \mathsf{HistSpine}$ and a history h, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. Then same visible constructor tags around two represented-spine readbacks of h determine the untagged payload classifier:

1. if

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(\mathsf{map}(e0, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(h, p'{+}{+}(\mathsf{map}(e0, y){+}{+}s'))$$

hold, then

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y);$$

2. if

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(\mathsf{map}(e1, x){+}{+}s)), \quad \mathsf{ListSpineRep}_A(h, p'{+}{+}(\mathsf{map}(e1, y){+}{+}s'))$$

hold, then

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y).$$

The ten-field fit is: traditional target finite-list same-history readback determinacy; BEDC source A with $\mathsf{NameCert}(A, \mathsf{hsame})$*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$*; stability from spine coherence and certificate reflexivity; ledger visible $e0/e1$ branch tags in finite spines; core theorem same-tag context exactness at identical endpoints; exactness by represented-spine carrier coverage and public classifier reflexivity; standard bridge finite-spine readback; status* ProofTarget*.*

Proof. Start with either displayed represented-spine presentation of h. By Theorem 36.61, the presentation exposes $\mathsf{ListHistoryCarrier}_A(h)$.

Apply Theorem 36.68 to cert and the coherence witness. The resulting semantic certificate supplies classifier reflexivity for the carrier witness, hence

$$\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, h).$$

For the $e0$ clause, specialize Theorem 36.203 to the two endpoints h and h, using the displayed $e0$ presentations, the classified prefix and suffix hypotheses, and the reflexive public classifier above. Project the forward implication of the $e0$ branch to obtain $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$.

The $e1$ clause is the same specialization with $e1$ in place of $e0$. Project the forward implication of the $e1$ branch, again using the public reflexivity of h, to obtain $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x, y)$. □

Theorem 36.205 (List public append unary length classifier transport)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. Assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories $h, h', k, k', r, r' : \mathsf{BHist}$, suppose*

$$\mathsf{ListPublicAppend}_A(h, k, r)$$

holds, and suppose also that

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, h'), \qquad \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k, k'), \qquad \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(r, r'$$

hold. Then there exist unary length histories $u, v, w : \mathsf{BHist}$ such that

1. $\mathsf{ListPublicUnaryLength}_A(h, u)$ *and* $\mathsf{ListPublicUnaryLength}_A(h', u)$ *hold;*
2. $\mathsf{ListPublicUnaryLength}_A(k, v)$ *and* $\mathsf{ListPublicUnaryLength}_A(k', v)$ *hold;*
3. $\mathsf{ListPublicUnaryLength}_A(r, w)$ *and* $\mathsf{ListPublicUnaryLength}_A(r', w)$ *hold;*
4. $$\mathsf{NatAdd}(v, u, w);$$
5. *for every $z : \mathsf{BHist}$,* $$\mathsf{NatAdd}(u, v, z) \to \mathsf{hsame}(w, z).$$

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicAppend public length classifier transport`

Proof. Apply Theorem 36.196 to the displayed public append witness. This gives histories u, v, w with public unary length witnesses for h, k, r, together with $\mathsf{NatAdd}(v, u, w)$ and the commuted uniqueness row $\mathsf{NatAdd}(u, v, z) \to \mathsf{hsame}(w, z)$ for arbitrary z.

Next apply Theorem 36.93 three times, each time using cert and $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$. The classifier from h to h' transports the public unary length witness at h to one at h' with the same token u; the classifier from k to k' does the same for v; and the classifier from r to r' does the same for w.

Finally, repack the original witnesses for h, k, r, the transported witnesses for h', k', r', and the two Nat-addition rows obtained from append additivity. No additional computation is needed, since classifier transport preserves the unary token rather than replacing it. □

36.7 Bordered singleton payload readback

Theorem 36.206 (List classifier bordered singleton payload readback)**.** *For finite BEDC-history lists p, p', s, and s' and histories x and y, if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(x :: \mathsf{nil}){+}{+}s, p'{+}{+}$$

hold, then $\mathsf{hsame}(x, y)$.

Proof. Apply Theorem 36.44 to the displayed prefix classifiers, suffix classifiers, and bordered singleton classification. This cancels the classified boundaries and gives $\mathsf{ListClassifierSpec}(\mathsf{hsame}, x :: \mathsf{nil}, y :: \mathsf{nil})$.

Apply Theorem 36.21 in the forward direction. The singleton classifier witness reads back exactly the payload relation $\mathsf{hsame}(x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame bordered singleton payload readback`

Theorem 36.207 (List classifier bordered zero-headed singleton payload readback)**.** *For finite BEDC-history lists p, p', s, and s' and histories x and y, if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{E0}(x) :: \mathsf{nil}){+}{+}s, p$$

hold, then $\mathsf{hsame}(x, y)$.

Proof. Cancel the classified suffixes from the bordered classifier, then cancel the classified prefixes. The remaining singleton classifier identifies $\mathsf{E0}(x)$ with $\mathsf{E0}(y)$, and zero-headed constructor exactness reads back the payload relation $\mathsf{hsame}(x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame bordered e0 singleton payload readback`

Theorem 36.208 (List classifier bordered one-headed singleton payload readback)**.** *For finite BEDC-history lists p, p', s, and s' and histories x and y, if*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s'), \quad \mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(\mathsf{E1}(x) :: \mathsf{nil}){+}{+}s, p$$

hold, then $\mathsf{hsame}(x, y)$.

Proof. Cancel the classified suffixes from the displayed bordered classifier, then cancel the classified prefixes. The remaining singleton classifier identifies $\mathsf{E1}(x)$ with $\mathsf{E1}(y)$. One-headed constructor exactness reads back the payload relation $\mathsf{hsame}(x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame bordered e1 singleton payload readback`

The ten-field fit is: traditional target finite-list singleton payload readback; BEDC source BHist; carrier finite BEDC-history spines; classifier ListClassifierSpec(hsame, −, −); stability from classified prefix and suffix cancellation; ledger hidden prefix and suffix context; core theorem singleton payload readback; exactness by cancellation and singleton inversion; standard bridge finite-spine payload extraction; status ProofTarget.

Theorem 36.209 (List represented-spine bordered singleton payload readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \mathsf{hsame})$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\mathsf{hsame}}$*. For finite BEDC-history spines* $p, p', s, s' : \mathsf{HistSpine}$ *and histories* $x, y, h, k : \mathsf{BHist}$*, suppose*

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p, p')$$

and

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, s, s')$$

hold. If

$$\mathsf{ListSpineRep}_A(h, p{+}{+}(x :: \mathsf{nil}){+}{+}s), \quad \mathsf{ListSpineRep}_A(k, p'{+}{+}(y :: \mathsf{nil}){+}{+}s')$$

hold, and if $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}(h, k)$ *holds, then* $\mathsf{hsame}(x, y)$*.*

Proof. Apply Theorem 36.145 to the two displayed endpoint representations and the public represented-spine classifier. This reads the endpoint classifier as

$$\mathsf{ListClassifierSpec}(\mathsf{hsame}, p{+}{+}(x :: \mathsf{nil}){+}{+}s, p'{+}{+}(y :: \mathsf{nil}){+}{+}s').$$

The classified prefix and suffix hypotheses now place the displayed finite spines in the scope of Theorem 36.206. Applying that theorem cancels the classified borders and reads back the singleton payload relation $\mathsf{hsame}(x, y)$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListBridgeClassifier bordered singleton payload readback`

The ten-field fit is: traditional target finite-list represented endpoint payload readback; BEDC source A with $\mathsf{NameCert}(A, \mathsf{hsame})$; carrier $\mathsf{ListHistoryCarrier}_A$; classifier $\mathsf{ListSpineHistoryClassifier}_{A,\mathsf{hsame}}$; stability from source certificate plus spine-representation coherence; ledger represented endpoints with classified prefix and suffix borders; core theorem bordered singleton payload readback; exactness by displayed-spine public classifier conversion and finite-spine border cancellation; standard bridge represented finite spines; status ProofTarget.

36.8 Public reverse readback

Definition 36.210 (List public reverse readback)**.** *For a BEDC history source carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$*, define* $\mathsf{ListPublicReverse}_A(h, k)$ *to mean that there exists a finite BEDC-history spine* $xs : \mathsf{HistSpine}$ *such that*

$$\mathsf{ListSpineRep}_A(h, xs)$$

and

$$\mathsf{ListSpineRep}_A(k, \mathsf{reverse}(xs))$$

hold.

Theorem 36.211 (List public reverse readback is classifier-functional and involutive)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, h', k, k' : \mathsf{BHist}$*, the public reverse readback relation satisfies:*

1. *if* $\mathsf{ListPublicReverse}_A(h, k)$ *holds, then* $\mathsf{ListHistoryCarrier}_A(h) \wedge \mathsf{ListHistoryCarrier}_A(k)$*;*
2. *if* $\mathsf{ListPublicReverse}_A(h, k)$ *holds, then* $\mathsf{ListPublicReverse}_A(k, h)$*;*
3. *if* $\mathsf{ListPublicReverse}_A(h, k)$ *and* $\mathsf{ListPublicReverse}_A(h, k')$ *hold, then* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k, k')$*;*
4. *if* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h, h')$*,* $\mathsf{ListPublicReverse}_A(h, k)$*, and* $\mathsf{ListPublicReverse}_A(h', k')$ *hold, then* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k, k')$*.*

Proof. For carrier soundness, unpack the reverse witness as a spine xs with $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, \mathsf{reverse}(xs))$. Apply Theorem 36.61 to each displayed representation to obtain the two public carrier witnesses.

For symmetry, keep the displayed representation of k by $\mathsf{reverse}(xs)$. Finite-spine reverse involutivity identifies $\mathsf{reverse}(\mathsf{reverse}(xs))$ with xs, so the displayed representation of h becomes the required reversed-spine representation from k back to h.

For classifier functionality over a fixed source endpoint, unpack the two reverse witnesses as spines xs and ys. Coherence for the two presentations of h gives $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. Apply Theorem 36.41 to classify $\mathsf{reverse}(xs)$ with $\mathsf{reverse}(ys)$, then repack this finite-spine classifier with the two displayed representations of k and k' by the reverse direction of Theorem 36.145.

For classifier stability along classified source endpoints, unpack the two reverse witnesses as spines xs and ys. The forward direction of Theorem 36.145 reads the public classifier between h and h' as

$$\mathsf{ListClassifierSpec}(\sim_A, xs, ys).$$

Apply Theorem 36.41 to obtain a classifier between $\mathsf{reverse}(xs)$ and $\mathsf{reverse}(ys)$. Repack that finite-spine classifier with the reverse-readback representations of k and k' by the reverse direction of Theorem 36.145. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicReverse functional involutive`
Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicReverse classifier functional`

Theorem 36.212 (List public reverse preserves public length)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, k : \mathsf{BHist}$ *and a finite length value* n*, the public reverse readback relation preserves the public length ledger in both directions:*

1. *if* $\mathsf{ListPublicReverse}_A(h, k)$ *and* $\mathsf{ListPublicLength}_A(h, n)$ *hold, then* $\mathsf{ListPublicLength}_A(k, n)$*;*
2. *if* $\mathsf{ListPublicReverse}_A(h, k)$ *and* $\mathsf{ListPublicLength}_A(k, n)$ *hold, then* $\mathsf{ListPublicLength}_A(h, n)$*.*

Proof. For the forward direction, unpack the reverse witness as a spine xs with $\mathsf{ListSpineRep}_A(h, xs)$ and $\mathsf{ListSpineRep}_A(k, \mathsf{reverse}(xs))$. Unpack the public length witness for h as a spine ys with $\mathsf{ListSpineRep}_A(h, ys)$ and $\ell(ys) = n$.

Spine coherence compares the two displayed representations of h, giving $\mathsf{ListClassifierSpec}(\sim_A, xs, ys)$. The length exactness theorem for classified finite spines identifies $\ell(xs)$ with $\ell(ys)$, and finite-spine reverse length preservation identifies $\ell(\mathsf{reverse}(xs))$ with $\ell(xs)$. Chaining these equalities with $\ell(ys) = n$ gives $\ell(\mathsf{reverse}(xs)) = n$.

Repack the displayed representation $\mathsf{ListSpineRep}_A(k, \mathsf{reverse}(xs))$ together with the resulting length equality through Definition 36.75. This is $\mathsf{ListPublicLength}_A(k, n)$.

For the reverse direction, use the involutive clause of Theorem 36.211 to turn $\mathsf{ListPublicReverse}_A(h,k)$ into $\mathsf{ListPublicReverse}_A(k,h)$. Apply the forward direction just proved to this reversed public-reverse witness and the displayed public length witness for k, obtaining $\mathsf{ListPublicLength}_A(h,n)$. □

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicReverse public length transport`

Theorem 36.213 (List public reverse fixed-length classifier automorphism)**.** *The roadmap ten-field fit is: traditional target finite-list reverse as a fixed-length classifier automorphism; BEDC source* BHist *finite spines; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability from spine coherence and the source* NameCert*; ledger finite spine reversal plus public length; core theorem this fixed-length automorphism package; exactness through reverse readback and length transport; standard bridge finite spines; status* ProofTarget*.*

Let $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, h', k, k' : \mathsf{BHist}$ *and a finite length value* n*, if*

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(h,h') \wedge \mathsf{ListPublicReverse}_A(h,k) \wedge \mathsf{ListPublicReverse}_A(h',k') \wedge \mathsf{ListPublicLength}_A(h,n)$$

holds, then

$$\mathsf{ListPublicLength}_A(k,n) \wedge \mathsf{ListPublicLength}_A(k',n) \wedge \mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k,k') \wedge \mathsf{ListPublicReverse}_A(k,h) \wedge \mathsf{Lis}$$

holds.

Proof. First put both classified source endpoints on the displayed public-length fiber. Apply Theorem 36.97 to the classifier witness between h and h' and the public length witness for h. This gives $\mathsf{ListPublicLength}_A(h',n)$.

Next transport the length through the two public reverse witnesses. Apply Theorem 36.212 to $\mathsf{ListPublicReverse}_A(h,k)$ and $\mathsf{ListPublicLength}_A(h,n)$ to obtain $\mathsf{ListPublicLength}_A(k,n)$. Apply the same theorem to $\mathsf{ListPublicReverse}_A(h',k')$ and the transported length witness for h' to obtain $\mathsf{ListPublicLength}_A(k',n)$.

Now classify the reversed endpoints. The classifier-stability clause of Theorem 36.211, applied to the source classifier witness and the two reverse readback witnesses, produces $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(k,k')$.

Finally, reverse back along each public reverse witness. The involutive clause of Theorem 36.211 sends $\mathsf{ListPublicReverse}_A(h,k)$ to $\mathsf{ListPublicReverse}_A(k,h)$ and sends $\mathsf{ListPublicReverse}_A(h',k')$ to $\mathsf{ListPublicReverse}_A(k',h')$. Combining the two length witnesses, the reversed-endpoint classifier, and these two reverse-back witnesses gives the claimed conjunction. □

The finite-data dependency lattice for $\mathsf{List}^{\uparrow}$ passes through the sibling certificates chapter 32, chapter 33, chapter 34, and chapter 35. Bool supplies the tag rows, Option supplies endpoint absence and presence rows, Product supplies component-envelope rows, and Sum supplies branch-tag rows; the list spine classifier becomes public only after those finite-data rows are available as sibling routes.

Theorem 36.214 (List finite-data sibling route)**.** *The public* $\mathsf{List}^{\uparrow}$ *spine classifier consumes* $\mathsf{Bool}^{\uparrow}$ *tag rows,* $\mathsf{Option}^{\uparrow}$ *endpoint absence/presence rows,* $\mathsf{Prod}^{\uparrow}$ *component envelope rows, and* $\mathsf{Sum}^{\uparrow}$ *branch tag rows before its spine classifier is public. Therefore any accepted list-spine read factors through the finite-data sibling lattice of chapter 32, chapter 33, chapter 34, and chapter 35 before it exports an empty, cons, append, reverse, or payload-bearing spine route.*

Proof. Read the sibling rows in finite-data order. The $\mathsf{Bool}^{\uparrow}$ surface supplies the visible tag discipline, $\mathsf{Option}^{\uparrow}$ separates absence from presence, $\mathsf{Prod}^{\uparrow}$ supplies component envelopes for payload and tail rows, and $\mathsf{Sum}^{\uparrow}$ supplies branch tags for the empty and cons alternatives.

The $\mathsf{List}^{\uparrow}$ spine classifier then reuses those rows as the finite route on which empty, singleton, append, reverse, and payload-bearing reads are displayed. Since the route has already passed through the tag, endpoint, component, and branch surfaces, the spine classifier is public only as a consumer of that sibling lattice. □

36.9 Public reverse and append

Theorem 36.215 (List public reverse turns append into opposite append)**.** *Per-object template: traditional target finite-list reverse anti-homomorphism; BEDC source* BHist *through* ListSpineRep*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability obligations public append congruence plus public reverse functionality; ledger finite spine presentation; core theorem this opposite-append law; exactness via represented-spine classifier displayed exactness; standard bridge ordinary finite-list* $reverse(xs{+}{+}ys) = reverse(ys){+}{+}reverse(xs)$*; status* ProofTarget*.*

Let $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, k, r, h^{\mathrm{rev}}, k^{\mathrm{rev}}, s : \mathsf{BHist}$*, suppose*

$$\mathsf{ListPublicAppend}_A(h, k, r), \quad \mathsf{ListPublicReverse}_A(h, h^{\mathrm{rev}}), \quad \mathsf{ListPublicReverse}_A(k, k^{\mathrm{rev}})$$

and

$$\mathsf{ListPublicAppend}_A(k^{\mathrm{rev}}, h^{\mathrm{rev}}, s)$$

hold. Then there exists a history $s' : \mathsf{BHist}$ *such that*

$$\mathsf{ListPublicReverse}_A(r, s')$$

and

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, s')$$

hold.

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame reverse append antimorphism`

Proof. Unpack the append witness $\mathsf{ListPublicAppend}_A(h, k, r)$ as finite spines xs, ys with h represented by xs, k represented by ys, and r represented by $xs{+}{+}ys$.

Unpack the two reverse witnesses. The public reverse of h supplies a displayed representation of h^{rev} by $\mathsf{reverse}(xs)$ up to the source classifier, and the public reverse of k supplies a displayed representation of k^{rev} by $\mathsf{reverse}(ys)$ up to the same classifier. These comparisons use the functionality clause of Theorem 36.211 together with represented-spine classifier displayed exactness.

Use the finite-spine identity

$$\mathsf{reverse}(xs{+}{+}ys) = \mathsf{reverse}(ys){+}{+}\mathsf{reverse}(xs).$$

The representation of r by $xs{+}{+}ys$ therefore repacks as a public reverse witness from r to an endpoint s' represented by $\mathsf{reverse}(ys){+}{+}\mathsf{reverse}(xs)$.

Finally compare the given opposite-append output s with the repacked endpoint s'. The public append congruence and same-input functionality clauses of Theorem 36.195, applied to k^{rev} and h^{rev} and transported through the reverse functionality comparisons above, give $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, s')$. □

Theorem 36.216 (List classifier e1-map reverse append antimorphism)**.** *For BEDC history lists* $xs, ys, xs^{\mathrm{rev}}, ys^{\mathrm{rev}}$*, if* xs^{rev} *classifies* $reverse(xs)$ *and* ys^{rev} *classifies* $reverse(ys)$ *under* hsame*, then the* E1*-mapped reverse of* $xs{+}{+}ys$ *classifies the* E1*-mapped opposite append* $ys^{\mathrm{rev}}{+}{+}xs^{\mathrm{rev}}$*.*

Proof. Use the finite-list identity $reverse(xs{+}{+}ys) = reverse(ys){+}{+}reverse(xs)$. Symmetry of the two classifier hypotheses gives classifiers from $reverse(ys)$ to ys^{rev} and from $reverse(xs)$ to xs^{rev}; append congruence combines them into a classifier from $reverse(ys){+}{+}reverse(xs)$ to $ys^{\mathrm{rev}}{+}{+}xs^{\mathrm{rev}}$. Transport along the finite-list identity and apply the E1 map classifier theorem. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e1 reverse append antimorphism`

Theorem 36.217 (List classifier e0-map reverse append antimorphism)**.** *For BEDC history lists* $xs, ys, xs^{\mathrm{rev}}, ys^{\mathrm{rev}}$*, if* xs^{rev} *classifies* $reverse(xs)$ *and* ys^{rev} *classifies* $reverse(ys)$ *under* hsame*, then the* E0*-mapped reverse of* $xs{+}{+}ys$ *classifies the* E0*-mapped opposite append* $ys^{\mathrm{rev}}{+}{+}xs^{\mathrm{rev}}$*.*

Proof. Use the finite-list identity $reverse(xs{+}{+}ys) = reverse(ys){+}{+}reverse(xs)$. Symmetry of the two classifier hypotheses gives classifiers from $reverse(ys)$ to ys^{rev} and from $reverse(xs)$ to xs^{rev}; append congruence combines them into a classifier from $reverse(ys){+}{+}reverse(xs)$ to $ys^{\mathrm{rev}}{+}{+}xs^{\mathrm{rev}}$. Transport along the finite-list identity and apply the E0 map classifier theorem. □

Lean4 ✓ `BEDC.Derived.ListUp.ListClassifierSpec hsame map e0 reverse append antimorphism`

Theorem 36.218 (List public reverse append output classifier uniqueness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. The ten-field fit is: traditional target finite-list reverse append output classifier uniqueness; BEDC source* BHist *through* ListSpineRep*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability obligations public append congruence, public reverse functionality, and classifier transitivity; ledger finite spine presentation; core theorem this reverse-output uniqueness law; exactness via represented-spine classifier displayed exactness and public reverse readback functionality; standard bridge ordinary finite-list reverse append readback; status* ProofTarget*.*

For histories $h, k, r, h^{\mathrm{rev}}, k^{\mathrm{rev}}, s, t : \mathsf{BHist}$*, suppose*

$$\mathsf{ListPublicAppend}_A(h, k, r), \quad \mathsf{ListPublicReverse}_A(h, h^{\mathrm{rev}}), \quad \mathsf{ListPublicReverse}_A(k, k^{\mathrm{rev}})$$

and

$$\mathsf{ListPublicAppend}_A(k^{\mathrm{rev}}, h^{\mathrm{rev}}, s), \quad \mathsf{ListPublicReverse}_A(r, t)$$

hold. Then

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, t)$$

holds.

Lean4 ✓ `BEDC.Derived.ListUp.FramedListPublicReverse append output classifier unique`

Proof. Apply Theorem 36.215 to the public append witness for h, k, r, the two public reverse witnesses for h and k, and the displayed opposite public append witness for $k^{\mathrm{rev}}, h^{\mathrm{rev}}, s$. This gives a history $s' : \mathsf{BHist}$ with $\mathsf{ListPublicReverse}_A(r, s')$ and $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, s')$.

Clause 3 of Theorem 36.211, applied to the two public reverse readbacks from the same source r, compares s' with t:

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s', t).$$

Finally compose $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, s')$ with $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s', t)$. The needed transitivity is the spine-induced history classifier transitivity supplied by Theorem 36.63 together with the bridge-classifier presentation in Definition 36.57. □

Theorem 36.219 (List public reverse append length ledger coherence)**.** *The ten-field fit is: traditional target finite-list reverse append law with a public length ledger; BEDC source* BHist *through* ListSpineRep*; carrier* $\mathsf{ListHistoryCarrier}_A$*; classifier* $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}$*; stability obligations public append congruence, public reverse functionality, classifier transitivity, and public length transport; ledger the finite public length value n; core theorem this reverse-append endpoint and length coherence package; exactness via represented-spine classifier displayed exactness, public reverse length preservation, and public length classifier transport; standard bridge ordinary finite-list reverse append with length preservation; status* ProofTarget*.*

Let $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume* $\mathsf{ListSpineRepCoherent}_{A,\sim_A}$*. For histories* $h, k, r, h^{\mathrm{rev}}, k^{\mathrm{rev}}, s, t :$ BHist *and a finite length value n, suppose*

$$\mathsf{ListPublicAppend}_A(h, k, r), \quad \mathsf{ListPublicReverse}_A(h, h^{\mathrm{rev}}), \quad \mathsf{ListPublicReverse}_A(k, k^{\mathrm{rev}})$$

and

$$\mathsf{ListPublicAppend}_A(k^{\mathrm{rev}}, h^{\mathrm{rev}}, s), \quad \mathsf{ListPublicReverse}_A(r, t), \quad \mathsf{ListPublicLength}_A(r, n)$$

hold. Then

$$\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, t) \wedge \mathsf{ListPublicLength}_A(s, n) \wedge \mathsf{ListPublicLength}_A(t, n)$$

holds.

Proof. First apply Theorem 36.218 to the public append witness for h, k, r, the two public reverse witnesses for h and k, the displayed opposite public append witness for $k^{\mathrm{rev}}, h^{\mathrm{rev}}, s$, and the public reverse witness from r to t. This gives $\mathsf{ListSpineHistoryClassifier}_{A,\sim_A}(s, t)$.

Next apply Theorem 36.212 to $\mathsf{ListPublicReverse}_A(r, t)$ and the displayed ledger witness $\mathsf{ListPublicLength}_A(r, n)$. The forward length-preservation clause gives $\mathsf{ListPublicLength}_A(t, n)$.

Finally apply Theorem 36.97 to the classifier witness between s and t. Using the reverse transport direction with the length witness at t yields $\mathsf{ListPublicLength}_A(s, n)$. Combining the classifier witness and the two endpoint length witnesses gives the required conjunction. □

Lean4 ✓ `BEDC.Derived.ListUp.ListUp StdBridge`

Definition 36.220 (List public structural recursion data)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. A public structural recursion datum over the framed list carrier consists of a predicate* $P : \mathsf{BHist} \to \mathsf{Prop}$ *together with the following two clauses:*

1. *for every endpoint h, a witness* $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$ *yields* $P(h)$*;*
2. *for every endpoint h and histories a, t, p, if $A(a)$ holds, if t is represented by a framed tail spine, if $P(t)$ holds, if* $\mathsf{Cont}(a, t, p)$ *holds, and if* $\mathsf{hsame}(h, \mathsf{E1}(p))$ *holds, then $P(h)$.*

Equivalently, the cons clause is allowed to consume a finite tail spine xs and a witness $\mathsf{FramedListSpineRep}_A(t, xs)$ *as the displayed framed tail representation.*

Theorem 36.221 (List public structural recursion principle)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier, let* $\sim_A$ *be a source classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and assume the cons-boundary coherence needed to read each framed cons endpoint as data* $A(a)$*,* $\mathsf{FramedListSpineRep}_A(t, xs)$*,* $\mathsf{Cont}(a, t, p)$*, and* $\mathsf{hsame}(h, \mathsf{E1}(p))$*. For every public structural recursion datum* $(P, \mathsf{nil}_P, \mathsf{cons}_P)$ *in the sense of Definition 36.220, every public framed carrier endpoint satisfies*

$$\mathsf{FramedListHistoryCarrier}_A(h) \to P(h).$$

Lean4 ✓ `BEDC.Derived.ListUp.ListHistoryCarrier structural recursion`

Proof. Unpack the public framed carrier witness as a finite displayed spine xs with $\mathsf{FramedListSpineRep}_A(h, xs)$. It is enough to prove the stronger displayed statement: for every endpoint represented by a framed spine xs, the predicate P holds at that endpoint.

Induct on the displayed framed spine. In the nil case, the representation is $\mathsf{FramedListSpineRep}_A(h, \mathsf{nil})$, so the nil clause nil_P gives $P(h)$ directly.

In the cons case, the representation exposes histories a, t, p and a tail spine xs with $A(a)$, $\mathsf{FramedListSpineRep}_A(t, xs)$, $\mathsf{Cont}(a, t, p)$, and $\mathsf{hsame}(h, \mathsf{E1}(p))$. The induction hypothesis applied to the tail representation gives $P(t)$. Applying the cons clause cons_P to these five pieces of data yields $P(h)$.

The classifier $\sim_A$, the certificate cert, and the boundary coherence witness are not used to add cases; they record that the public framed endpoint being eliminated is the same certificate-bearing surface used by append, reverse, length, and bridge transport. The eliminator itself is the finite-spine induction exposed by $\mathsf{FramedListHistoryCarrier}_A(h)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{List}^\uparrow}$.

Constructive story: A list is a finite spine that either stops at nil or takes one more cons step carrying a payload from the source type A; this is the lone certificate in this run whose carrier is the host inductive `List A` rather than a history predicate, because finite lists are exactly unary growth with data attached at each step. Two such spines are classified together only when they have the same nil-or-cons shape and, at every cons step, their payloads are identified by the source-side equivalence on A. The source is the whole list type, the pattern records the nil/cons case split, append, reverse, length, constructor maps, and endpoint readback, and the ledger keeps the visible list equal to the raw list being structurally inspected. Since the source equivalence is reflexive, symmetric, and transitive, structural induction lifts those three facts to lists, gives the nil inhabitant, and shows that classifier-recognized lists remain in the carrier. A list xs lies in the signature gap of a $\mathsf{List}^\uparrow$ token p when the source-domain accepts xs, the signature relation pairs it with an answer list and evidence ledger, and the token-introduction relation names that answer as p; unfolded, every cons payload of xs must match the corresponding payload encoded by p, and nil must match nil. The finite-spine bridge then reads append, reverse, and length as operations on that same payload-bearing spine rather than as host-list axioms. The chapter registers this payload-bearing-spine construction under the name $\mathsf{List}^\uparrow$; list operations reduce to spine operations, with no quotient principle, sorting API, or arbitrary host-list theory introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{List}^\uparrow}$, maturePackageClosed)

Scope closed: The mature ListUp scope is the finite-spine surface generated by the framed semantic certificate, nil/cons no-confusion, append, reverse, constructor-map, length, endpoint readback, and finite-spine bridge transport.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{List}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.ListUp.FramedListBridgeClassifier\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: The chapter does not close quotient principles, sorting, arbitrary host-list APIs, or a bridgeChecked standard-model correspondence.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 37

A Concrete Naming Certificate for $\mathsf{Rat}^{\uparrow}$

The rational number interface $\mathsf{Rat}^{\uparrow}$ starts from a carrier skeleton: a signed unary integer numerator, drawn from chapter 31, together with a positive unary denominator. Normalized representatives, coprimality, and arithmetic closure are separate source, pattern, classifier, and stability obligations later in the chapter. The classifier is structural equality on the normalized representative, not the equivalence relation $p_1q_2 = p_2q_1$ on unnormalized pairs, which would otherwise demand quotient-type machinery and Quot.sound. It is not a primitive field; it is licensed only via the integer source and the normalization function.

Remark 37.1 (Rational certificate scope)**.** *This chapter records the rational licensing target through normalized integer-positive-natural representatives and structural classifier data. The checked core covers the rational history carrier, rational history classifier fields, ledger carrier, and semantic certificate. Normalized representatives are recorded by the source and pattern obligations, while coprimality and operation closure remain part of the paper specification.*

37.1 Carrier definition

Definition 37.2 (Rat carrier)**.** *Use only the carrier skeleton: a signed unary integer numerator together with a unary denominator history that is not internally the same as* e*:*

$$\mathsf{RatCarrier}(s, n, d) := \mathsf{IntCarrier}(s, n) \wedge \mathsf{Unary}(d) \wedge \neg\mathsf{hsame}(d, \mathsf{e}).$$

Equivalently, this is an $\mathsf{Int}^{\uparrow}$ *numerator with a positive unary-history denominator. At this skeleton layer, positivity is witnessed by writing the denominator as a one-extension of a unary tail; normalized representatives, coprimality, and arithmetic closure are supplied by later source, pattern, and stability obligations.*

Lean4 (definition) `BEDC.Derived.RatUp.RatCarrier`

Theorem 37.3 (Rat carrier positive denominator)**.** *If* $\mathsf{RatCarrier}(s, n, d)$*, then the denominator is internally the same as* $\mathsf{E1}(t)$ *for some unary tail* t*.*

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier positive denominator`

Proof. Unpack the carrier into its integer numerator, unary denominator, and non-emptiness clauses. The unary denominator is either e or a one-extension of a unary tail. The empty case contradicts non-emptiness, while the one-extension case gives the displayed positive-denominator witness. □

Theorem 37.4 (Positive denominator yields a rational carrier)**.** *An* $\mathsf{Int}^{\uparrow}$ *numerator together with a denominator internally same as* $\mathsf{E1}(t)$ *for a unary tail* t *forms a* $\mathsf{Rat}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier of int positive denominator`

Proof. Transport unary closure from the displayed E1-shape to the denominator along hsame. Non-emptiness follows because an E1-history cannot be internally same as e. □

Theorem 37.5 (Rat carrier denominator transport)**.** *If* $\mathsf{RatCarrier}(s, n, d)$ *and* $d\mathsf{hsame}d'$*, then* $\mathsf{RatCarrier}(s, n, d')$*.*

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier denominator hsame transport`

Proof. Extract the positive denominator witness from the rational carrier, transport that witness across denominator sameness, and rebuild the rational carrier from the unchanged integer numerator together with the transported positive denominator. □

Theorem 37.6 (Positive unary denominator transport)**.** *If a denominator history has a positive unary witness and is* hsame*-equivalent to another history, then the other history has the same positive unary denominator witness.*

Proof. Write the positive denominator as an E1-extension of a unary tail. Compose the reverse hsame witness from the target history to the source history with the source witness to the E1-extension, and keep the unary tail witness. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator hsame transport`

Theorem 37.7 (One-extension positive denominator equivalence)**.** *A one-extension history has a positive unary denominator witness exactly when its tail is unary.*

Proof. For the forward direction, eliminate the positive-denominator witness $\mathsf{hsame}(\mathsf{E1}(t), \mathsf{E1}(u))$ with unary tail u. One-constructor inversion gives $\mathsf{hsame}(t, u)$, and unary membership transports back to t. For the reverse direction, use t itself as the displayed positive tail and the reflexive one-extension sameness witness. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator e1 iff unary`

Theorem 37.8 (Left one-extension append positive denominator equivalence)**.** *For histories* $h, t : \mathsf{BHist}$*,*

$$\mathsf{PositiveUnaryDenominator}(\mathsf{append}(\mathsf{E1}(h), t))$$

holds exactly when both h *and* t *are unary histories.*

Proof. If the appended history has a positive unary denominator witness, then it is a unary history. Unary append factorization recovers unary membership of the left factor and of the tail; one-extension inversion turns unary membership of $\mathsf{E1}(h)$ into unary membership of h.

Conversely, unary membership of h makes $\mathsf{E1}(h)$ a positive unary denominator, and positive denominators are closed under appending a unary tail. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append e1 left iff`

Theorem 37.9 (Right one-extension append positive denominator equivalence)**.** *For histories* $h, t : \mathsf{BHist}$*,*

$$\mathsf{PositiveUnaryDenominator}(\mathsf{append}(h, \mathsf{E1}(t)))$$

holds exactly when both h *and* t *are unary histories.*

Proof. The right one-extension unfolds the appended denominator as $\mathsf{E1}(\mathsf{append}(h, t))$. The one-extension positive-denominator equivalence reads this as unary membership of $\mathsf{append}(h, t)$, and unary append factorization is exactly the conjunction of unary membership of h and t. The reverse direction rebuilds the unary append and applies the one-extension positive-denominator equivalence. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append e1 right iff`

Theorem 37.10 (Zero-extension cannot be a positive denominator)**.** *A zero-extension history has no positive unary denominator witness.*

Proof. Unpack the positive denominator witness. It would identify an $\mathsf{E0}$-history with an $\mathsf{E1}$-history, contradicting finite-kernel history constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator e0 absurd`

Theorem 37.11 (Positive unary denominator is not empty)**.** *If a denominator has a positive unary witness, then it is not internally the same as* e*.*

Proof. Unpack the positive denominator as internally the same as $\mathsf{E1}(t)$ for a unary tail t. Composing this witness with an assumed sameness to e would make an $\mathsf{E1}$-history internally the same as e, contradicting history constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator not empty`

Theorem 37.12 (Positive unary denominator is unary and nonempty)**.** *If a denominator history has a positive unary witness, then it is unary and cannot be internally the same as* e*.*

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator unary and nonempty`

Proof. Write the positive denominator as internally the same as $\mathsf{E1}(t)$ with t unary. Unary closure under $\mathsf{E1}$, transported back along history sameness, gives unary membership of the denominator. If the denominator were internally same as e, transitivity would make $\mathsf{E1}(t)$ internally same as e, contradicting constructor separation. □

Theorem 37.13 (Positive unary denominator carrier equivalence)**.** *The rational carrier condition is equivalent to an* $\mathsf{Int}^{\uparrow}$ *numerator together with a denominator that has a positive unary tail witness.*

Proof. Unpack the carrier. A unary denominator that is not hsame-empty cannot be the empty unary history, so unary case analysis gives a one-extension tail. Conversely, a denominator hsame-equal to a one-extension of a unary tail is unary by transport, and it cannot be hsame-empty by constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier iff positive unary denominator`

Theorem 37.14 (Positive denominator append unary tail)**.** *If a denominator has a positive unary witness and the appended tail is unary, then the continued denominator still has a positive unary witness.*

Proof. Write the denominator as internally the same as $\mathsf{E1}(t)$ for a unary tail t, and transport unary membership back to the denominator itself. The appended tail is either e or a one-step unary tail. The empty case keeps the original witness, while the one-step case takes the continued denominator without that final step as the new positive tail; closure follows from unary append closure. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append unary tail`

Theorem 37.15 (Positive unary denominator append zero-tail exclusion)**.** *For histories* d, t : BHist*, the continued denominator* $\mathsf{append}(d, \mathsf{E0}(t))$ *has no positive unary denominator witness.*

Proof. A positive denominator witness first gives unary membership for the whole continued history. Unary append factor extraction then gives unary membership for the right factor $\mathsf{E0}(t)$, contradicting zero-extension separation for unary histories. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append e0 tail absurd`

Theorem 37.16 (Positive unary denominator append one-tail equivalence)**.** *For histories* d, t : BHist*,*

$$\mathsf{PositiveUnaryDenominator}(\mathsf{append}(d, \mathsf{E1}(t))) \iff \mathsf{UnaryHistory}(\mathsf{append}(d, t)).$$

Proof. The forward direction reads unary membership from the positive denominator witness; the one-step shape of continuation identifies this with unary membership of $\mathsf{append}(d, t)$. Conversely, unary membership of $\mathsf{append}(d, t)$ is exactly the positive denominator witness for the one-extension $\mathsf{append}(d, \mathsf{E1}(t))$. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append e1 tail iff unary append`

Theorem 37.17 (Rat carrier append unary denominator closure)**.** *If a rational carrier has denominator* d*, then appending any unary history to* d *preserves rational-carrier membership.*

Proof. Use the carrier equivalence to expose the integer numerator carrier and the positive unary denominator witness. Positive denominators are closed under appending unary tails, and the rational carrier is rebuilt from the same integer numerator and the resulting positive denominator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier append unary denominator closed`

Theorem 37.18 (Rat history carrier is not empty)**.** *If a rational history carrier has denominator* d*, then* d *cannot be internally the same as* e*.*

Proof. Unpack the rational history carrier into its sign, numerator, and rational carrier. The rational carrier exposes a positive unary denominator witness, and positive unary denominators are separated from e. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier not empty`

Theorem 37.19 (Rat history carrier positive denominator equivalence)**.** *A denominator history lies in the rational history carrier exactly when it has a positive unary denominator witness.*

Proof. In the forward direction, unpack the rational history carrier into a sign, numerator, and rational carrier, then apply rational-carrier positivity of the denominator. In the reverse direction, use the concrete integer numerator carrier with sign b0 and empty magnitude, then rebuild the rational carrier from the positive denominator witness. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier iff positive denominator`

Theorem 37.20 (Rat history carrier hsame transport)**.** *If a denominator history lies in the rational history carrier and is transported along* hsame *to another denominator history, then the transported denominator history lies in the rational history carrier.*

Proof. Use the positive denominator equivalence of Definition 37.19. The source carrier gives a positive unary denominator witness, and Definition 37.6 carries that witness across the displayed hsame comparison.

Reapply the reverse direction of Definition 37.19 to rebuild the rational history carrier at the transported denominator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier hsame transport`

Theorem 37.21 (Rat history eone tail unary equivalence)**.** *A one-extension denominator* $\mathsf{E1}(t)$ *lies in the rational history carrier exactly when its tail* t *is unary.*

Proof. For the forward direction, unpack the rational history carrier to obtain the positive denominator witness, then invert the displayed one-extension denominator to transport unary membership to t. For the reverse direction, use the canonical integer numerator with sign b0 and empty magnitude, and use t itself as the positive unary denominator tail. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier e1 tail unary iff`

Theorem 37.22 (Rat history carrier excludes zero-extension)**.** *No zero-extension history* $\mathsf{E0}(t)$ *lies in the rational history carrier.*

Proof. Unpack the rational history carrier into its sign, numerator, and rational carrier. The rational carrier supplies a positive unary denominator witness, and positive unary denominators cannot have zero-extension shape. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier e0 absurd`

Theorem 37.23 (Rat history carrier append unary denominator closure)**.** *If a rational history carrier has denominator* d*, then appending any unary history to* d *preserves rational history carrier membership.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier append unary denominator closed`

Proof. Unpack the rational history carrier into its sign, numerator, and rational carrier data. Apply rational-carrier denominator append closure to the concrete denominator component, then rebuild the existential rational history carrier with the same sign and numerator. □

Theorem 37.24 (Positive denominator append unary prefix)**.** *If* u *is unary and* d *has a positive unary denominator witness, then* $\mathsf{append}(u, d)$ *has a positive unary denominator witness.*

Proof. Write the positive denominator witness as an E1 tail, say d is internally the same as $\mathsf{E1}(t)$ with t unary. Appending the unary prefix u preserves unary structure by unary append closure, and composing the displayed E1 witness with append compatibility gives the required positive witness for $\mathsf{append}(u, d)$. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append unary prefix`

Theorem 37.25 (Positive unary denominator append unary left iff)**.** *If u is unary, then a continued denominator $\mathsf{append}(u, r)$ has a positive unary denominator witness exactly when either r is internally empty and u already has such a witness, or r itself has such a witness.*

Proof. Case-split on the right history r. The empty case reduces the continuation to u. A zero-extension case is impossible because no zero-extension is a positive denominator. In the one-extension case, one-constructor readback gives unary membership of the continued tail, and unary append factorization recovers the right tail witness. Conversely, the empty-right alternative reduces by endpoint equality, while a positive witness on r is preserved by unary prefix closure. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append unary left iff`

Theorem 37.26 (Positive unary denominator append unary left split)**.** *If p is unary and $\mathsf{append}(p, d)$ has a positive unary denominator witness, then either p already has a positive unary denominator witness or d has one.*

Proof. Split on the visible shape of the unary prefix. The empty prefix transports the positive witness across left-unit append. A zero-headed prefix contradicts unary history membership. A one-headed prefix is itself a positive unary denominator. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append unary left split`

Theorem 37.27 (Positive denominator append unary context)**.** *If p and t are unary histories and d has a positive unary denominator witness, then $\mathsf{append}(p, \mathsf{append}(d, t))$ has a positive unary denominator witness.*

Proof. First close the positive denominator under the unary tail t. Then close the resulting positive denominator under the unary prefix p. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append unary context`

Theorem 37.28 (Rat carrier prepend unary denominator closure)**.** *If a rational carrier has denominator d, then prepending any unary history to d preserves rational-carrier membership.*

Proof. Use the carrier equivalence to expose the integer numerator carrier and the positive unary denominator witness. Positive denominators are closed under unary prefix continuation, and the rational carrier is rebuilt from the same integer numerator and the resulting positive denominator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier prepend unary denominator closed`

Theorem 37.29 (Rat history carrier prepend unary denominator closure)**.** *If a rational history carrier has denominator d, then prepending any unary history to d preserves rational history carrier membership.*

Proof. Unpack the rational history carrier into its sign, numerator, and rational carrier data. Apply rational-carrier denominator prepend closure to the concrete denominator component, then rebuild the existential rational history carrier with the same sign and numerator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier prepend unary denominator closed`

Theorem 37.30 (Rat carrier unary denominator context closure)**.** *If a rational carrier has denominator d, then placing a unary prefix and a unary tail around d preserves rational-carrier membership.*

Proof. First close the carrier under appending the unary tail to the denominator. Then close the resulting carrier under prepending the unary prefix to that continued denominator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier unary denominator context closed`

Theorem 37.31 (Rat history carrier unary denominator context closure)**.** *If a rational history carrier has denominator d, then placing a unary prefix and a unary tail around d preserves rational history carrier membership.*

Proof. Unpack the rational history carrier into its sign, numerator, and rational carrier data. Apply rational-carrier unary denominator context closure to the concrete denominator component, then rebuild the existential rational history carrier with the same sign and numerator. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryCarrier unary denominator context closed`

Theorem 37.32 (Rat carrier hsame transport)**.** *If a rational carrier* $\mathsf{RatCarrier}(s, n, d)$ *is transported along* $\mathsf{msame}(s, t)$*,* $\mathsf{hsame}(n, n')$*, and* $\mathsf{hsame}(d, d')$*, then the transported triple is again a rational carrier.*

Lean4 ✓ `BEDC.Derived.RatUp.RatCarrier hsame transport`

Proof. Eliminate the sign sameness, then unpack the rational carrier. The numerator is transported by the integer carrier transport theorem, denominator unary closure is transported along $\mathsf{hsame}(d, d')$, and denominator non-emptiness is preserved by composing $\mathsf{hsame}(d, d')$ with any empty-target witness. □

37.2 The certificate

Definition 37.33 (Rat source specification)**.** *The source is* $\mathsf{Int}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ *restricted to normalized denominator-positive pairs.*

Lean4 (definition) `BEDC.Derived.RatUp.RatSourceSpec`

Theorem 37.34 (Rat source specification yields the carrier)**.** *A rational source witness determines a rational carrier witness: the numerator is an integer carrier, the denominator is unary, and the positive one-tail witness excludes internal sameness with* e*.*

Lean4 ✓ `BEDC.Derived.RatUp.RatSourceSpec to RatCarrier`

Proof. Unpack the source witness. The positive-denominator component gives a unary tail t with $d\mathsf{hsame}e_1(t)$. Unary closure under one-extension and transport along hsame give $\mathsf{Unary}(d)$. If $d\mathsf{hsame}\mathsf{e}$, then symmetry and transitivity give $e_1(t)\mathsf{hsame}\mathsf{e}$, contradicting finite-kernel history no-confusion. □

Definition 37.35 (Rat pattern specification)**.** *The pattern is normalized numerator/denominator construction via a total normalization function on integer and positive-natural inputs.*

Definition 37.36 (Rat classifier specification)**.** *The classifier is structural sameness of normalized numerator and denominator representatives, tracked by mark sameness on signs and history sameness on the numerator and denominator histories.*

Lean4 (definition) `BEDC.Derived.RatUp.RatClassifierSpec`

Theorem 37.37 (Rat classifier reflexivity)**.** *Every rational carrier witness classifies its representative with itself.*

Proof. Reuse the carrier witness for both endpoints. The sign component is witnessed by reflexivity of msame, and the numerator and denominator components are witnessed by reflexivity of hsame. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec refl`

Theorem 37.38 (Rat classifier exposes positive denominators)**.** *If two rational representatives are related by the rational classifier, then both endpoint denominators carry positive unary denominator witnesses.*

Proof. Eliminate the classifier witness to expose the two endpoint rational carriers. Apply the rational-carrier positive-denominator theorem to each endpoint carrier. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec positive denominators`

Theorem 37.39 (Rat classifier transports positive denominators)**.** *If two rational representatives are related by the rational classifier, and each denominator is related by* hsame *to a displayed denominator history, then both displayed denominator histories carry positive unary denominator witnesses.*

Proof. Eliminate the classifier witness to expose the two endpoint rational carriers. Each endpoint carrier gives a positive unary denominator witness, and the two denominator hsame links transport those witnesses to the displayed denominator histories. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec denominator positive transport`

Theorem 37.40 (Rat positive denominator append factor or tail)**.** *If the append of two denominator histories carries a positive unary denominator witness, then either the left factor already carries such a witness or the right tail carries one.*

Proof. Inspect the right tail. The empty case leaves the displayed append equal to the left factor. A zero-headed tail contradicts the positive one-headed denominator witness. For a one-headed tail, the positive witness exposes unary membership of the appended core, and unary append factor recovery gives unary membership of the right core, hence a positive denominator witness for the tail. □

Lean4 ✓ `BEDC.Derived.RatUp.PositiveUnaryDenominator append factor or tail`

Theorem 37.41 (Rat classifier denominator append closure)**.** *If two rational representatives are classified the same, then appending* hsame*-matching unary histories to their denominators preserves the rational classifier relation.*

Proof. Unpack the classifier witness into the two endpoint carriers and the sign, numerator, and denominator sameness fields. Close both endpoint denominator carriers under unary append, transporting the unary-tail witness along the tail sameness for the second endpoint. The resulting denominator sameness follows by eliminating the original denominator and tail sameness witnesses and applying reflexivity to the resulting append history. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec append unary denominators closed`

Theorem 37.42 (Rat classifier denominator prepend closure)**.** *If two rational representatives are classified the same, then prepending* hsame*-matching unary histories to their denominators preserves the rational classifier relation.*

Proof. Unpack the classifier witness into the two endpoint carriers and the sign, numerator, and denominator sameness fields. Close both endpoint denominator carriers under unary prepend, transporting unary membership of the first prefix across prefix sameness for the second endpoint. The resulting denominator sameness follows by eliminating the original denominator and prefix sameness witnesses and applying reflexivity to the resulting append history. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec prepend unary denominators closed`

Theorem 37.43 (Rat classifier unary denominator context closure)**.** *If two rational representatives are classified the same, then surrounding their denominators by* hsame*-matching unary prefixes and unary tails preserves the rational classifier relation.*

Proof. First apply denominator append closure to the classified representatives and the matching unary tails. Then apply denominator prepend closure to the resulting classified representatives and the matching unary prefixes. The sign and numerator witnesses are unchanged, while the denominator endpoint relation is obtained by the two concrete unary-context closure steps. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec append unary denominator context closed`

Theorem 37.44 (Rat classifier unary denominator context exactness)**.** *For* $s_1, s_2 : \mathsf{BMark}$ *and* $n_1, n_2, d_1, d_2, p_1, p_2, t_1, t_2 : \mathsf{BHist}$*, assume* $\mathsf{RatCarrier}(s_1, n_1, d_1)$*,* $\mathsf{RatCarrier}(s_2, n_2, d_2)$*,* $\mathsf{Unary}(p_1)$*,* $\mathsf{Unary}(t_1)$*,* $\mathsf{hsame}(p_1, p_2)$*, and* $\mathsf{hsame}(t_1, t_2)$*. Then*

$$\mathsf{RatClassifierSpec}(s_1, n_1, \mathsf{append}(p_1, \mathsf{append}(d_1, t_1)), s_2, n_2, \mathsf{append}(p_2, \mathsf{append}(d_2, t_2))) \iff \mathsf{RatClassifierSpec}(s_1, n_1, d$$

Proof. Suppose first that the contextual classifier holds. Unpack it to expose the contextual endpoint carriers and the sign, numerator, and denominator sameness fields. Keep the supplied original carrier witnesses for $\mathsf{RatCarrier}(s_1, n_1, d_1)$ and $\mathsf{RatCarrier}(s_2, n_2, d_2)$.

The denominator field gives sameness between the two displayed context denominators. Transport this relation across $\mathsf{hsame}(p_1, p_2)$ and $\mathsf{hsame}(t_1, t_2)$, then cancel the matching prefix and suffix by append cancellation to obtain $\mathsf{hsame}(d_1, d_2)$. Repack the original classifier with the unchanged sign and numerator fields and this cancelled denominator field.

Conversely, suppose the original classifier holds. The unary-context closure theorem, applied with the supplied unary prefix, unary tail, and matching hsame witnesses, gives the contextual classifier directly. □

Theorem 37.45 (Rat classifier append unary prefix denominator split)**.** *If two rational representatives are classified after matching unary prefixes are prepended to their denominators, then each displayed denominator side has a positive unary witness either in its prefix or in its remaining denominator.*

Proof. The classifier exposes positive unary denominator witnesses for the two prefixed denominators. Transport unary membership across the prefix hsame-link, then split each prefixed denominator by the positive unary left-split theorem. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec append unary prefix denominator split`

Theorem 37.46 (Rat classifier transitivity)**.** *If one rational representative is classified the same as a second representative, and the second representative is classified the same as a third representative, then the first and third representatives are classified the same.*

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec trans`

Proof. Unpack both classifier witnesses. The endpoint carriers are preserved, while sign sameness and the two history-sameness components compose by the finite-kernel transitivity theorems for msame and hsame. □

Theorem 37.47 (Rat classifier symmetry)**.** *If one rational representative is classified the same as a second representative, then the second representative is classified the same as the first.*

Proof. Unpack the endpoint carriers and the sign, numerator, and denominator sameness fields. Rebuild the classifier in the reverse direction using symmetry of msame for signs and symmetry of hsame for both history components. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec symm`

Theorem 37.48 (Rat classifier component transport)**.** *If both rational classifier endpoints are transported componentwise by* msame *on signs and* hsame *on numerator and denominator histories, then the rational classifier transports to the componentwise transported representatives.*

Proof. Unpack the rational classifier witness. Transport each endpoint rational carrier by the component transport theorem, then compose the old sign, numerator, and denominator sameness links with the reversed source endpoint transports and the target endpoint transports. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec component transport`

Theorem 37.49 (Rat history classifier one-tail exactness)**.** *For denominator tails* $d, e : \mathsf{BHist}$*, the rational history classifier relates the one-extension denominators exactly when both tails are unary and internally the same:*

$$\mathsf{RatHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)) \iff \mathsf{Unary}(d) \wedge \mathsf{Unary}(e) \wedge \mathsf{hsame}(d, e).$$

Proof. For the forward direction, unpack the classifier into two rational history carrier witnesses and the endpoint hsame link. The one-tail carrier equivalence extracts unary membership of both tails, and E1-constructor inversion gives $\mathsf{hsame}(d, e)$. For the reverse direction, rebuild both rational history carriers from the unary tails and lift $\mathsf{hsame}(d, e)$ by E1-congruence. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier e1 tail unary iff`

Theorem 37.50 (Rat history classifier exposes positive denominators)**.** *If denominator histories* d *and* e *are related by the rational history classifier, then both* d *and* e *carry positive unary denominator witnesses.*

Proof. Unpack the rational history classifier into its two endpoint rational-history carrier witnesses and the stored denominator hsame link. Convert each endpoint carrier to a positive unary denominator witness by the rational history carrier positive-denominator equivalence. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier positive denominators`

Theorem 37.51 (Rat history classifier endpoints are nonempty)**.** *If denominator histories d and e are related by the rational history classifier, then neither endpoint is internally the same as the empty history:*

$$(\mathsf{hsame}(d, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(e, \mathsf{e}) \to \mathsf{False}).$$

Proof. Unpack the rational history classifier into its two endpoint rational-history carrier witnesses. Apply rational-history carrier nonemptiness to each endpoint carrier. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier endpoints not empty`

Theorem 37.52 (Rat history classifier zero-extension endpoint exclusion)**.** *For histories t, d : BHist, zero-extension cannot appear as either endpoint of the rational history classifier:*

$$(\mathsf{RatHistoryClassifier}(\mathsf{E0}(t), d) \to \mathsf{False}) \wedge (\mathsf{RatHistoryClassifier}(d, \mathsf{E0}(t)) \to \mathsf{False}).$$

Proof. For the left endpoint, apply Definition 37.50 to the classifier hypothesis. Its first positive-denominator witness belongs to $\mathsf{E0}(t)$, so Definition 37.10 gives the contradiction.

For the right endpoint, apply Definition 37.50 to the classifier hypothesis again. Its second positive-denominator witness belongs to $\mathsf{E0}(t)$, and Definition 37.10 rules out that endpoint. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier e0 endpoint absurd`

The ten-field fit is: traditional target rational-denominator branch exclusion; BEDC source BHist and RatHistoryClassifier; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability from positive-denominator readback; ledger the rejected zero head; core theorem this block; exactness the two displayed contradictions; standard bridge positive denominators are never zero-headed unary histories; status ProofTarget.

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier zero extension endpoint exclusion`

Theorem 37.53 (Rat history classifier transitivity)**.** *If denominator histories d and e are related by the rational history classifier, and e and f are related by the same classifier, then d and f are related by the rational history classifier.*

Proof. Unpack the two rational history classifier witnesses. Keep the left endpoint carrier from the first witness and the right endpoint carrier from the second witness, then compose the denominator history sameness proofs by transitivity of hsame. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier trans`

Theorem 37.54 (Rat history classifier endpoint transport)**.** *If denominator histories d and e are related by the rational history classifier, and d and e are respectively internally the same as d' and e', then d' and e' are related by the rational history classifier.*

Proof. Unpack the rational history classifier into endpoint rational-history carriers and the stored $\mathsf{hsame}(d, e)$ link. Transport both endpoint carriers across the supplied history-sameness witnesses, then compose the reversed $\mathsf{hsame}(d, d')$ link, the stored classifier link, and $\mathsf{hsame}(e, e')$. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier hsame transport`

Theorem 37.55 (Rat history classifier symmetry)**.** *If denominator histories d and e are related by the rational history classifier, then e and d are related by the rational history classifier.*

Proof. Unpack the rational history classifier witness. Swap the two endpoint carrier witnesses, and reverse the denominator history-sameness proof by symmetry of hsame. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier symm`

Theorem 37.56 (Rat history classifier unary denominator append closure)**.** *If denominator histories d and e are related by the rational history classifier, and unary tails t_d and t_e satisfy* $\mathsf{hsame}(t_d, t_e)$*, then appending the tails to the corresponding denominator histories preserves the rational history classifier.*

Proof. Unpack the rational history classifier into endpoint carriers and denominator hsame. Close the left endpoint carrier under unary denominator append. Transport unary membership from t_d to t_e, close the right endpoint carrier under append, and combine the denominator hsame with the tail hsame by constructor cases on append. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier append unary denominator closed`

Theorem 37.57 (Rat history classifier unary denominator prepend closure)**.** *If denominator histories d and e are related by the rational history classifier, and unary prefixes p_d and p_e satisfy* $\mathsf{hsame}(p_d, p_e)$*, then prepending the prefixes to the corresponding denominator histories preserves the rational history classifier.*

Proof. Unpack the rational history classifier into endpoint carriers and denominator hsame. Close the left endpoint carrier under unary denominator prepend. Transport unary membership from p_d to p_e, close the right endpoint carrier under prepend, and combine the denominator and prefix hsame witnesses by constructor cases on the continuation append operation. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier prepend unary denominator closed`

Theorem 37.58 (Rat history classifier unary denominator context closure)**.** *If denominator histories d and e are related by the rational history classifier, unary prefixes p_d, p_e satisfy* $\mathsf{hsame}(p_d, p_e)$*, and unary tails t_d, t_e satisfy* $\mathsf{hsame}(t_d, t_e)$*, then placing the matching prefix and tail around the corresponding denominators preserves the rational history classifier.*

Proof. First apply unary denominator append closure to the classified denominator histories and the matching tails. Then apply unary denominator prepend closure to the resulting classified denominator histories and the matching prefixes. The endpoint carrier witnesses are closed by the two concrete unary closure steps, and the denominator relation is the composed hsame-respecting append context. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier unary denominator context closed`

Theorem 37.59 (Rat history classifier unary denominator context positive denominators)**.** *If a rational history classifier is closed under matching unary denominator prefixes and tails, then the two contextual denominator endpoints have positive unary denominator witnesses.*

Proof. Apply Definition 37.58 to the original classifier and the matching unary contexts. Then apply Definition 37.50 to the resulting contextual classifier. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier unary denominator context positive denominators`

Theorem 37.60 (Rat history classifier unary context zero-extension endpoint exclusion)**.** *Let* $d, e, p_d, p_e, t_d, t_e, z, z' : \mathsf{BHist}$. *If* $\mathsf{RatHistoryClassifier}(d, e)$ *holds,* p_d, p_e *are unary prefixes with* $\mathsf{hsame}(p_d, p_e)$*, and* t_d, t_e *are unary tails with* $\mathsf{hsame}(t_d, t_e)$*, then the two contextual denominator endpoints cannot be internally the same as zero-extension histories:*

$$(\mathsf{hsame}(\mathsf{append}(p_d, \mathsf{append}(d, t_d)), \mathsf{E0}(z)) \to \mathsf{False}) \wedge (\mathsf{hsame}(\mathsf{append}(p_e, \mathsf{append}(e, t_e)), \mathsf{E0}(z')) \to \mathsf{False}).$$

Proof. Apply Definition 37.59 to the displayed classifier, the matching unary prefix context, and the matching unary tail context. This produces positive unary denominator witnesses for $\mathsf{append}(p_d, \mathsf{append}(d, t_d))$ and $\mathsf{append}(p_e, \mathsf{append}(e, t_e))$.

For the left endpoint, transport the positive unary denominator witness across the displayed hsame link from $\mathsf{append}(p_d, \mathsf{append}(d, t_d))$ to $\mathsf{E0}(z)$. Then apply Definition 37.10 to rule out that zero-extension endpoint.

For the right endpoint, repeat the same transport across the displayed hsame link from $\mathsf{append}(p_e, \mathsf{append}(e, t_e))$ to $\mathsf{E0}(z')$. The positive-denominator zero-head absurdity theorem gives the second contradiction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier unary context zero extension endpoint absurd`

The ten-field fit is: traditional target contextual positive-rational branch exclusion; BEDC source RatHistoryClassifier over BHist; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability from unary context closure and positive-denominator transport; ledger the hidden prefix and tail contexts; core theorem this block; exactness the two contradictions; downstream consumer FieldUp(RatUp) denominator continuation and RealUp stream-prefix readback; status ProofTarget.

Theorem 37.61 (Rat history classifier context one-tail pair readback)**.** *If a rational history classifier is closed under matching unary denominator prefixes and tails, and the resulting left and right endpoints display as one-extension histories, then the displayed tails are unary and internally the same.*

Proof. Apply rational history classifier unary denominator context closure to the original classified denominators and the matching unary contexts. Transport the closed classifier across the two displayed endpoint sameness witnesses, then use rational one-tail classifier exactness to read back the two unary tails and their hsame relation. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier unary context e1 pair readback`

Theorem 37.62 (Rat history classifier continuation context one-tail pair readback)**.** *If the matching unary denominator context is presented by continuation displays and the two outputs are one-extension histories, then their tails are unary and internally the same.*

Proof. Use the two continuation displays and append associativity to identify the outputs with the normal context endpoints, then apply Definition 37.61. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier cont unary context e1 pair readback`

Definition 37.63 (Rat stability certificate)**.** *The stability certificate consists of:*

1. *normalization produces positive denominator*
2. *normalization produces coprime pair*

3. *normalization is idempotent*
4. *structural reflexivity and transitivity*
5. *positive denominator closure under unary prefix continuation*
6. *closure of rational operations after re-normalization*

Definition 37.64 (Rat ledger policy)**.** *Record the raw rational-history carrier and an* hsame *link to the visible denominator history.*

Lean4 (definition) `BEDC.Derived.RatUp.RatHistoryLedgerPolicy`

Theorem 37.65 (Rat ledger visible carrier)**.** *If a raw rational history satisfies the ledger policy for a visible history, then the visible history is itself a rational-history carrier.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible carrier`

Proof. Unpack the ledger policy into a raw rational-history carrier and an hsame link from the raw history to the visible history. Transport the raw carrier across that link. □

Theorem 37.66 (Rat ledger visible positive denominator readback)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds and* $\mathsf{hsame}(v, \mathsf{e1}(t))$ *holds, then* $\mathsf{UnaryHistory}(t)$ *holds.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible positive denominator readback`

Proof. The ledger visible-carrier theorem transports the raw rational-history carrier to v. Transport that carrier once more across $\mathsf{hsame}(v, \mathsf{e1}(t))$ and apply the one-extension rational-history carrier characterization. □

Theorem 37.67 (Rat ledger append unary denominator closure)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds,* $\mathsf{UnaryHistory}(t)$ *holds, and* $\mathsf{hsame}(t, t')$ *holds, then appending those unary denominator tails preserves the rational ledger:*

$$\mathsf{RatHistoryLedgerPolicy}(\mathsf{append}(\rho, t), \mathsf{append}(v, t')).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy append unary denominator closed`

Proof. Unpack the ledger into a raw rational-history carrier and an $\mathsf{hsame}(\rho, v)$ link. The denominator-append carrier theorem closes the raw endpoint under the unary tail, and the two hsame links identify the appended raw and visible histories. □

Theorem 37.68 (Rat ledger prepend unary denominator closure)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds,* $\mathsf{UnaryHistory}(p)$ *holds, and* $\mathsf{hsame}(p, p')$ *holds, then prefixing those unary denominator histories preserves the rational ledger:*

$$\mathsf{RatHistoryLedgerPolicy}(\mathsf{append}(p, \rho), \mathsf{append}(p', v)).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy prepend unary denominator closed`

Proof. Unpack the ledger into a raw rational-history carrier and an $\mathsf{hsame}(\rho, v)$ link. The denominator-prefix carrier theorem closes the raw endpoint under the unary prefix, and the two hsame links identify the prefixed raw and visible histories. □

Theorem 37.69 (Rat ledger unary denominator context closure)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, unary prefixes* p, p' *satisfy* $\mathsf{hsame}(p, p')$*, and unary tails* t, t' *satisfy* $\mathsf{hsame}(t, t')$*, then placing the matching prefix and tail around the raw and visible denominator histories preserves the rational ledger:*

$$\mathsf{RatHistoryLedgerPolicy}(\mathsf{append}(p, \mathsf{append}(\rho, t)), \mathsf{append}(p', \mathsf{append}(v, t'))).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy unary denominator context closed`

Proof. First apply rat ledger append unary denominator closure to the displayed ledger and the matching tails. Then apply rat ledger prepend unary denominator closure to the resulting ledger and the matching prefixes. □

Theorem 37.70 (Rat ledger unary context zero-extension endpoint exclusion)**.** *Let* $\rho, v, p, p', t, t', z, z' :$ BHist*. If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds,* p, p' *are matched unary prefixes with* $\mathsf{hsame}(p, p')$*, and* t, t' *are matched unary tails with* $\mathsf{hsame}(t, t')$*, then the contextual raw and visible ledger endpoints cannot be zero-extension endpoints:*

$$(\mathsf{hsame}(\mathsf{append}(p, \mathsf{append}(\rho, t)), \mathsf{E0}(z)) \to \mathsf{False}) \wedge (\mathsf{hsame}(\mathsf{append}(p', \mathsf{append}(v, t')), \mathsf{E0}(z')) \to \mathsf{False}).$$

Proof. Apply rat ledger unary denominator context closure to obtain the contextual ledger. The raw-visible classifier theorem turns that ledger into a rational history classifier between the contextual raw and visible endpoints. Transport this classifier across a displayed zero-extension equality on either endpoint, then use rational history classifier zero-extension endpoint exclusion. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy unary context zero extension endpoint exclusion`

Theorem 37.71 (Rat ledger history-sameness transport)**.** *If a rational ledger relates raw and visible histories, then transporting the raw endpoint and the visible endpoint along* hsame *yields another rational ledger relation.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy hsame transport`

Proof. Unpack the ledger into a raw carrier and a raw-visible link. Transport the raw carrier across the raw-endpoint link, then compose the reversed raw transport link, the stored ledger link, and the visible-endpoint link. □

Theorem 37.72 (Rat ledger raw-visible classifier)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, then* $\mathsf{RatHistoryClassifier}(\rho, v)$ *holds.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy raw visible classifier`

Proof. Unpack the ledger into the raw rational-history carrier for ρ and the stored $\mathsf{hsame}(\rho, v)$ link. The visible-carrier theorem supplies the rational-history carrier for v, and these witnesses form the raw-visible rational history classifier. □

Theorem 37.73 (Rat ledger shared-raw visible classifier)**.** *If both* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, then* $\mathsf{RatHistoryClassifier}(v, v')$ *holds.*

Proof. Convert both rational ledger witnesses to raw-visible classifiers. Reverse the first classifier by rational classifier symmetry, then compose it with the second raw-visible classifier by rational classifier transitivity. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw visible classifier`

Theorem 37.74 (Rat ledger shared visible zero-extension exclusion)**.** *If both* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, then neither visible denominator endpoint can be displayed as a zero-extension endpoint:*

$$(\mathsf{hsame}(v, \mathsf{E0}(z)) \to \mathsf{False}) \wedge (\mathsf{hsame}(v', \mathsf{E0}(z')) \to \mathsf{False}).$$

Proof. Use the shared-raw visible classifier to obtain $\mathsf{RatHistoryClassifier}(v, v')$. If v is displayed as $\mathsf{E0}(z)$, transport the left endpoint of this classifier to obtain a rational classifier with zero-extension source, which contradicts rational classifier zero-extension endpoint exclusion. The visible endpoint v' is identical: transport the right endpoint to $\mathsf{E0}(z')$ and apply the right branch of the same exclusion theorem. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared visible e0 endpoint absurd`

Theorem 37.75 (Rat ledger shared-raw contextual visible classifier)**.** *If both* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, unary prefixes* $p, p_v, p_{v'}$ *satisfy* $\mathsf{hsame}(p, p_v)$ *and* $\mathsf{hsame}(p, p_{v'})$*, and unary tails* $t, t_v, t_{v'}$ *satisfy* $\mathsf{hsame}(t, t_v)$ *and* $\mathsf{hsame}(t, t_{v'})$*, then the contextual visible denominator endpoints are classified:*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(p_v, \mathsf{append}(v, t_v)), \mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'}))).$$

Proof. Apply Definition 37.69 to the first ledger with the raw context p, t and the visible context p_v, t_v. This produces a ledger from the shared raw contextual endpoint $\mathsf{append}(p, \mathsf{append}(\rho, t))$ to the first contextual visible endpoint.

Apply the same context-closure theorem to the second ledger with the same raw context p, t and the visible context $p_{v'}, t_{v'}$. The two contextual ledger witnesses now share the raw endpoint $\mathsf{append}(p, \mathsf{append}(\rho, t))$.

Finally apply Definition 37.73 to those two contextual ledger witnesses. Its conclusion is exactly the displayed classifier between the two contextual visible denominator endpoints. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw contextual visible classifier`
Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw context visible classifier`

Theorem 37.76 (Rat ledger shared-raw contextual visible endpoint equivalence)**.** *If both* $\mathsf{RatHistoryLedgerPolicy}(\rho,$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, unary prefixes* $p, p_v, p_{v'}$ *satisfy* $\mathsf{hsame}(p, p_v)$ *and* $\mathsf{hsame}(p, p_{v'})$*, and unary tails* $t, t_v, t_{v'}$ *satisfy* $\mathsf{hsame}(t, t_v)$ *and* $\mathsf{hsame}(t, t_{v'})$*, then every target history* w *sees the same classifier fiber over the two contextual visible denominator endpoints:*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(p_v, \mathsf{append}(v, t_v)), w) \iff \mathsf{RatHistoryClassifier}(\mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'})), w).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw context visible endpoint equivalence`

Proof. Apply Definition 37.75 to the two ledger witnesses and the matching unary prefix and tail hypotheses. This gives the classifier edge from $\mathsf{append}(p_v, \mathsf{append}(v, t_v))$ to $\mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'}))$.

For the forward direction, assume the left contextual visible endpoint is classified with w. Reverse the contextual endpoint edge by rational classifier symmetry, then compose it with the left-endpoint-to-w classifier by transitivity. The result is the right-endpoint-to-w classifier.

For the reverse direction, assume the right contextual visible endpoint is classified with w. Compose the contextual endpoint edge itself with the right-endpoint-to-w classifier by transitivity. The result is the left-endpoint-to-w classifier. □

Theorem 37.77 (Rat ledger shared-raw context one-tail readback)**.** *If both* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, unary prefixes* $p, p_v, p_{v'}$ *satisfy* $\mathsf{hsame}(p, p_v)$ *and* $\mathsf{hsame}(p, p_{v'})$*, unary tails* $t, t_v, t_{v'}$ *satisfy* $\mathsf{hsame}(t, t_v)$ *and* $\mathsf{hsame}(t, t_{v'})$*, and the two contextual visible endpoints display as one-extension histories,*

$$\mathsf{hsame}(\mathsf{append}(p_v, \mathsf{append}(v, t_v)), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(\mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'})), \mathsf{E1}(b)),$$

then the displayed tails are unary and internally the same:

$$\mathsf{Unary}(a) \wedge \mathsf{Unary}(b) \wedge \mathsf{hsame}(a, b).$$

Its ten-field fit is: traditional target shared-denominator visible-tail determinacy; BEDC source $\mathsf{Rat}^{\uparrow}$ *ledger over* BHist*; carrier positive unary denominators; classifier* $\mathsf{RatHistoryClassifier}$*; stability from unary context closure and classifier transport; ledger the shared raw denominator plus two visible contexts; core theorem this block; exactness the displayed tail readback; standard bridge normalized denominator readback; status* $\mathsf{ProofTarget}$*.*

Proof. First invoke Definition 37.75 with the two ledger witnesses and the matching raw-to-visible unary contexts. This gives the rational history classifier between the two contextual visible endpoints.

Then apply Definition 37.61 to that classifier and to the two displayed one-extension witnesses. The readback conclusion is exactly $\mathsf{Unary}(a) \wedge \mathsf{Unary}(b) \wedge \mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw context e1 pair readback`

Theorem 37.78 (Rat ledger classifier extension)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds and* $\mathsf{RatHistoryClassifier}(v, t)$ *holds, then* $\mathsf{RatHistoryClassifier}(\rho, t)$ *holds.*

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy classifier extension`

Proof. Convert the raw-visible ledger into the raw-visible classifier, then compose it with the visible-target classifier by transitivity. □

Theorem 37.79 (Rat history ledger classifier endpoint equivalence)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, then every target history* w *is classified from* ρ *exactly when it is classified from* v*:*

$$\mathsf{RatHistoryClassifier}(\rho, w) \leftrightarrow \mathsf{RatHistoryClassifier}(v, w).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy classifier endpoint equivalence`

Proof. The ledger supplies the raw-visible rational classifier. For the raw-to-visible direction, reverse that classifier by symmetry and compose it with the raw-target classifier by transitivity. The visible-to-raw direction is the ledger classifier extension theorem. □

Theorem 37.80 (Rat history ledger displayed context classifier endpoint equivalence)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho,$ *holds, unary prefixes* p_ρ, p_v *satisfy* $\mathsf{hsame}(p_\rho, p_v)$*, unary tails* t_ρ, t_v *satisfy* $\mathsf{hsame}(t_\rho, t_v)$*, and the displayed continuation witnesses* $\mathsf{Cont}(p_\rho, \rho, m_\rho)$*,* $\mathsf{Cont}(m_\rho, t_\rho, out_\rho)$*,* $\mathsf{Cont}(p_v, v, m_v)$*, and* $\mathsf{Cont}(m_v, t_v, out_v)$ *expose the two contextual endpoints, then every target history* w *is classified from the displayed raw endpoint exactly when it is classified from the displayed visible endpoint:*

$$\mathsf{RatHistoryClassifier}(out_\rho, w) \leftrightarrow \mathsf{RatHistoryClassifier}(out_v, w).$$

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context classifier endpoint equivalence`

Proof. First close the ledger under the matching unary denominator context. Apply Definition 37.69 to the original ledger, the prefix match, and the tail match. This gives a ledger between $\mathsf{append}(p_\rho, \mathsf{append}(\rho, t_\rho))$ and $\mathsf{append}(p_v, \mathsf{append}(v, t_v))$.

The four continuation witnesses identify the displayed endpoints with those normalized append endpoints. Associativity of append turns the two-step displays through m_ρ and m_v into hsame links from out_ρ and out_v to the corresponding normalized contextual histories.

Transport the contextual ledger across these two endpoint identifications by Definition 37.71. The transported ledger now relates out_ρ and out_v directly.

Finally apply Definition 37.79 to the transported ledger. Its conclusion is the displayed classifier endpoint equivalence for every target history w. □

Theorem 37.81 (Rat history ledger displayed context positive denominators)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, unary prefixes* p_ρ, p_v *satisfy* $\mathsf{hsame}(p_\rho, p_v)$*, unary tails* t_ρ, t_v *satisfy* $\mathsf{hsame}(t_\rho, t_v)$*, and the displayed continuations* $\mathsf{Cont}(p_\rho, \rho, m_\rho)$*,* $\mathsf{Cont}(m_\rho, t_\rho, out_\rho)$*,* $\mathsf{Cont}(p_v, v, m_v)$*, and* $\mathsf{Cont}(m_v, t_v, out_v)$ *expose the contextual endpoints, then*

$$\mathsf{PositiveUnaryDenominator}(out_\rho) \wedge \mathsf{PositiveUnaryDenominator}(out_v).$$

Proof. Close the ledger under matching unary denominator context by applying Definition 37.69 to the original ledger, the prefix match, and the tail match. This gives a ledger between the two normalized histories $\mathsf{append}(p_\rho, \mathsf{append}(\rho, t_\rho))$ and $\mathsf{append}(p_v, \mathsf{append}(v, t_v))$.

The four continuation witnesses identify out_ρ and out_v with the two displayed two-step continuations through m_ρ and m_v. Associativity of append turns those two displays into hsame links from the normalized contextual histories to the displayed outputs.

Transport the contextual ledger across these two endpoint identifications by Definition 37.71. The resulting ledger relates out_ρ and out_v directly.

Finally read the raw and visible carrier witnesses from the displayed ledger. The carrier equivalence for rational histories gives $\mathsf{PositiveUnaryDenominator}(out_\rho)$, and the visible-carrier projection gives $\mathsf{PositiveUnaryDenominator}(out_v)$. □

↪ variant `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context positive denominators`

Theorem 37.82 (Rat ledger displayed context zero-endpoint exclusion)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, unary prefixes* p_ρ, p_v *satisfy* $\mathsf{hsame}(p_\rho, p_v)$*, unary tails* t_ρ, t_v *satisfy* $\mathsf{hsame}(t_\rho, t_v)$*, and the displayed continuations* $\mathsf{Cont}(p_\rho, \rho, m_\rho)$*,* $\mathsf{Cont}(m_\rho, t_\rho, out_\rho)$*,* $\mathsf{Cont}(p_v, v, m_v)$*, and* $\mathsf{Cont}(m_v, t_v, out_v)$ *expose the contextual endpoints, then neither displayed endpoint can be a zero-extension endpoint:*

$$(\mathsf{hsame}(out_\rho, \mathsf{E0}(z_\rho)) \to \mathsf{False}) \wedge (\mathsf{hsame}(out_v, \mathsf{E0}(z_v)) \to \mathsf{False}).$$

Proof. Apply Definition 37.81 to the displayed contextual ledger data. This gives positive unary denominator witnesses for both out_ρ and out_v.

If out_ρ is displayed as $\mathsf{E0}(z_\rho)$, transport the positive denominator witness along that hsame link and contradict Definition 37.10. The visible endpoint out_v is identical: transport its positive denominator witness along $\mathsf{hsame}(out_v, \mathsf{E0}(z_v))$ and use the same zero-extension exclusion. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context e0 endpoint absurd`

Theorem 37.83 (Rat ledger displayed context empty-endpoint exclusion)**.** *If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, unary prefixes* p_ρ, p_v *satisfy* $\mathsf{hsame}(p_\rho, p_v)$*, unary tails* t_ρ, t_v *satisfy* $\mathsf{hsame}(t_\rho, t_v)$*, and the displayed continuations* $\mathsf{Cont}(p_\rho, \rho, m_\rho)$*,* $\mathsf{Cont}(m_\rho, t_\rho, out_\rho)$*,* $\mathsf{Cont}(p_v, v, m_v)$*, and* $\mathsf{Cont}(m_v, t_v, out_v)$ *expose the contextual endpoints, then neither displayed endpoint can be internally the empty history:*

$$(\mathsf{hsame}(out_\rho, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(out_v, \mathsf{e}) \to \mathsf{False}).$$

Proof. Apply Definition 37.81 to the displayed contextual ledger data. This gives positive unary denominator witnesses for both out_ρ and out_v.

A positive unary denominator is never internally the empty endpoint. Applying that nonempty denominator theorem to each of the two positive witnesses gives the two displayed contradictions. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context empty endpoint absurd`

Definition 37.84 ($\mathsf{Rat}^\uparrow$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Rat}^\uparrow$*.*

Theorem 37.85 (Concrete $\mathsf{Rat}^\uparrow$ naming certificate)**.** *The rational-history carrier/classifier tuple forms a valid semantic naming certificate for* $\mathsf{Rat}^\uparrow$ *at the history level.*

↪ variant `BEDC.Derived.RatUp.rat history semantic name certificate`

Theorem 37.86 (Rat ledger shared-raw contextual zero-extension endpoint exclusion)**.** *Let* $\rho, v, v', p, p_v, p_{v'}, t, t_v, t_{v'}$ BHist*. If* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryLedgerPolicy}(\rho, v')$ *hold, the unary prefixes* $p, p_v, p_{v'}$ *satisfy*

$$\mathsf{hsame}(p, p_v) \wedge \mathsf{hsame}(p, p_{v'}),$$

and the unary tails $t, t_v, t_{v'}$ *satisfy*

$$\mathsf{hsame}(t, t_v) \wedge \mathsf{hsame}(t, t_{v'}),$$

then for all $z, z' : \mathsf{BHist}$ *both contextual visible denominator endpoints are excluded from zero-extension heads:*

$$(\mathsf{hsame}(\mathsf{append}(p_v, \mathsf{append}(v, t_v)), \mathsf{E0}(z)) \to \mathsf{False}) \wedge (\mathsf{hsame}(\mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'})), \mathsf{E0}(z')) \to \mathsf{False}).$$

The ten-field fit is: traditional target RatUp ledger shared-denominator zero-branch exclusion; BEDC source BHist *plus* $\mathsf{RatHistoryLedgerPolicy}$*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from shared raw denominator and context transport; ledger shared raw denominator and context data; core theorem this block; exactness the two contradictions; downstream consumers* RealUp *denominator-context readback and* $\mathsf{FieldUp}$ *denominator continuation; status* $\mathsf{ProofTarget}$*.*

Proof. First invoke Definition 37.75 with the two ledger witnesses and the displayed prefix and tail hsame links. This gives the classifier edge between $\mathsf{append}(p_v, \mathsf{append}(v, t_v))$ and $\mathsf{append}(p_{v'}, \mathsf{append}(v', t_{v'}))$.

For a proposed zero-extension on the first contextual visible endpoint, transport the first classifier endpoint along its displayed hsame link to $\mathsf{E0}(z)$. The transported classifier has a zero-extension left endpoint, so Definition 37.52 gives the contradiction.

For a proposed zero-extension on the second contextual visible endpoint, transport the second classifier endpoint along its displayed hsame link to $\mathsf{E0}(z')$. The transported classifier has a zero-extension right endpoint, and the same endpoint-exclusion theorem gives the second contradiction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy shared raw contextual zero extension endpoint exclusion`

Corollary 37.87 ($\mathsf{Rat}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{Rat}^{\uparrow}$ *is licensed by normalized representatives, not by a quotient of pairs.*

Primitive boundary. $\mathsf{Rat}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 37.88 (Zero-axiom encoding note)**.** *Canonical normalization supplies the zero-axiom encoding without quotient equivalence or* Quot.sound*.*

Theorem 37.89 (Rat representative classifier stability package)**.** *Fix a rational source normalizer* $\mathsf{normalized} : \mathsf{BMark} \to \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$*. For every representative triple* s, n, d *with* $\mathsf{RatSourceSpec}(\mathsf{normalized}, s, n, d)$*, the row* RatSourceSpec_to_RatCarrier *supplies the representative carrier* $\mathsf{RatCarrier}(s, n, d)$*.*

On such representatives the classifier RatClassifierSpec *is stable in the following sense:*

1. RatClassifierSpec *is reflexive on every representative carrier.*
2. RatClassifierSpec *is symmetric on classified representative pairs.*
3. RatClassifierSpec *is transitive along composable representative pairs.*
4. *If signs are related by* msame *and numerator and denominator histories are related by* hsame*, then each classified representative pair transports componentwise to the displayed representatives.*

Thus the representative classifier used by the $\mathsf{Rat}^{\uparrow}$ *name certificate is an equivalence relation on normalized representatives and is stable under the BEDC component sameness relations for signs, numerators, and denominators.*

Proof. Start with a representative source witness $\mathsf{RatSourceSpec}(\mathsf{normalized}, s, n, d)$. The checked row RatSourceSpec_to_RatCarrier converts it into the carrier witness needed by the representative classifier.

The equivalence portion is supplied by three checked rows. Apply RatClassifierSpec_refl to a representative carrier for reflexivity, apply RatClassifierSpec_symm to reverse any classified representative pair, and apply RatClassifierSpec_trans to compose two adjacent classified representative pairs.

For component stability, use RatClassifierSpec_component_transport. Its hypotheses are exactly the displayed msame links on the two signs and the displayed hsame links on the two numerator histories and the two denominator histories, so it moves the classifier edge to the transported representative data.

The statement therefore packages the representative carrier conversion, the three equivalence laws, and component transport as one certificate layer. It records the combined representative-level stability supplied by several checked rows rather than restating any single row. □

Lean4 ✓ `BEDC.Derived.RatUp.RatClassifierSpec representative stability package`

Theorem 37.90 (Rat history semantic name certificate)**.** *The Rat history carrier with its history classifier supplies the semantic name certificate*

$$\mathsf{SemanticNameCert}(\mathsf{RatHistoryCarrier}, \mathsf{RatHistoryCarrier}, \mathsf{RatHistoryCarrier}, \mathsf{RatHistoryClassifier}).$$

Proof. The carrier is inhabited by the one-extension of the empty denominator tail. This is the positive unary denominator witness used by the checked core.

For the core certificate relation, reflexivity is the self-classifier supplied by carrier membership and hsame reflexivity. Symmetry reverses the hsame component of a classifier witness. Transitivity composes the two hsame components and uses carrier transport along hsame to recover the right endpoint carrier. The same carrier transport gives respect of the carrier under a classifier edge.

The semantic source, pattern, and ledger rows are all RatHistoryCarrier. Hence the source-to-pattern and source-to-ledger clauses are identity rows: each source witness is reused unchanged as the pattern witness and as the ledger witness. These are exactly the rows assembled by the Lean theorem. □

Lean4 ✓ `BEDC.Derived.RatUp.rat history semantic name certificate`

The positive-denominator ledger consumes the NatUp unary carrier of chapter 28 together with the signed-pair context supplied by chapter 31. These sibling chapters provide the unary denominator row and the signed numerator and denominator pair surface used by the following ledger readback theorem.

Theorem 37.91 (Rat history ledger displayed context one-tail readback)**.** *For histories* $\rho, v, p_\rho, p_v, t_\rho, t_v, m_\rho, m_v, ou$ BHist*, assume* RatHistoryLedgerPolicy(ρ, v)*, unary prefix and tail hypotheses with* hsame(p_ρ, p_v) *and* hsame(t_ρ, t_v)*, the four continuation displays* Cont(p_ρ, ρ, m_ρ)*,* Cont$(m_\rho, t_\rho, out_\rho)$*,* Cont$(p_v, v, m_v)$*, and* Cont$(m_v, t_v, out_v)$*, and endpoint displays* hsame$(out_\rho, \mathsf{E1}(a))$ *and* hsame$(out_v, \mathsf{E1}(b))$*. Then*

$$\mathsf{Unary}(a) \wedge \mathsf{Unary}(b) \wedge \mathsf{hsame}(a, b).$$

Its ten-field fit is: traditional target displayed rational-denominator one-tail readback; BEDC source $\mathsf{Rat}^{\uparrow}$ *ledger with* Cont *displays; carrier positive unary denominators; classifier* RatHistoryClassifier*; stability from contextual ledger transport and classifier endpoint transport; ledger hidden raw/visible denominator contexts; core theorem this block; exactness the displayed tail readback; standard bridge normalized denominator tail equality; status* ProofTarget*.*

Proof. Apply Definition 37.81 to the ledger witness, the matching unary prefix and tail hypotheses, and the four continuation displays. The visible output out_v is therefore a positive unary denominator, so it carries the self-classifier RatHistoryClassifier(out_v, out_v).

Next apply Definition 37.80 with target out_v. The visible self-classifier transports through the displayed contextual ledger endpoint equivalence and gives RatHistoryClassifier(out_ρ, out_v).

Use Definition 37.54 with the two endpoint displays hsame$(out_\rho, \mathsf{E1}(a))$ and hsame$(out_v, \mathsf{E1}(b))$. This converts the displayed endpoint classifier into RatHistoryClassifier$(\mathsf{E1}(a), \mathsf{E1}(b))$.

Finally apply Definition 37.49 in the forward direction. The one-tail classifier exactness theorem reads back exactly $\mathsf{Unary}(a) \wedge \mathsf{Unary}(b) \wedge \mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context e1 pair readback`

Theorem 37.92 (Rat ledger visible-target lift with positive endpoints)**.** *For histories* ρ, v, w : BHist*, if* RatHistoryLedgerPolicy(ρ, v) *and* RatHistoryClassifier(v, w) *hold, then*

$$\mathsf{RatHistoryClassifier}(\rho, w) \wedge \mathsf{PositiveUnaryDenominator}(\rho) \wedge \mathsf{PositiveUnaryDenominator}(v) \wedge \mathsf{PositiveUnaryDenominator}(u$$

Its ten-field fit is: traditional target ledger-stable rational target readback; BEDC source $\mathsf{Rat}^{\uparrow}$ *over* BHist*; carrier* RatHistoryCarrier*; classifier* RatHistoryClassifier*; stability visible-target classification lifted through the ledger; ledger* RatHistoryLedgerPolicy*; core theorem this block; exactness the displayed lifted classifier plus positive endpoint readback; downstream consumer the* $\mathsf{Rat}^{\uparrow}$ *semantic certificate boundary; status* ProofTarget*.*

Proof. Apply Definition 37.78 to the ledger witness and the visible-target classifier. This lifts the classifier from the visible endpoint v to the raw endpoint ρ, giving $\mathsf{RatHistoryClassifier}(\rho, w)$.

Next apply Definition 37.50 to the lifted classifier. This supplies positive unary denominator witnesses for ρ and w.

For the visible endpoint, apply Definition 37.65 to the original ledger witness. This gives $\mathsf{RatHistoryCarrier}(v)$.

Then use the forward direction of Definition 37.19 to obtain $\mathsf{PositiveUnaryDenominator}(v)$. The lifted classifier and the three positive-denominator witnesses assemble the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible target lift positive endpoints`

Theorem 37.93 (Rat ledger visible-target one-tail pair readback)**.** *For histories* ρ, v, w, a, b : BHist*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$*,* $\mathsf{RatHistoryClassifier}(v, w)$*,* $\mathsf{hsame}(\rho, \mathsf{E1}(a))$*, and* $\mathsf{hsame}(w, \mathsf{E1}(b))$ *hold, then*

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Lift the visible-target classifier through the raw-visible ledger to obtain $\mathsf{RatHistoryClassifier}(\rho, w)$. Transport this classifier across the two displayed one-extension endpoint equalities, producing $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$. Rational one-tail classifier exactness reads back the two unary tails and their internal sameness. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible target e1 pair readback`

Theorem 37.94 (Rat ledger visible-target positive nonempty package)**.** *For histories* ρ, v, w : BHist*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryClassifier}(v, w)$ *hold, then*

$$\begin{aligned}&\mathsf{PositiveUnaryDenominator}(\rho) \wedge \mathsf{PositiveUnaryDenominator}(v) \wedge \mathsf{PositiveUnaryDenominator}(w)\\ &\wedge (\mathsf{hsame}(\rho, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(v, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(w, \mathsf{e}) \to \mathsf{False}).\end{aligned}$$

Proof. Lift the visible-target classifier through the ledger to obtain $\mathsf{RatHistoryClassifier}(\rho, w)$. Positive-denominator readback from this lifted classifier gives the witnesses for ρ and w, while positive-denominator readback from the visible-target classifier gives the witness for v. Applying positive-denominator unary-and-nonempty readback to the three witnesses supplies the three displayed empty-endpoint exclusions. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible target positive nonempty package`

Theorem 37.95 (Rat ledger two-sided endpoint classifier equivalence)**.** *For histories* ρ, v, w : BHist*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, then*

$$(\mathsf{RatHistoryClassifier}(\rho, w) \iff \mathsf{RatHistoryClassifier}(v, w)) \wedge (\mathsf{RatHistoryClassifier}(w, \rho) \iff \mathsf{RatHistoryClassifier}(w, v))$$

Its ten-field fit is: traditional target two-sided $\mathsf{Rat}^{\uparrow}$ *ledger endpoint classifier exactness with conditional endpoint readback; BEDC source* $\mathsf{Rat}^{\uparrow}$ *ledger over* BHist*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from ledger endpoint equivalence, classifier symmetry, and classifier positive-denominator readback; ledger* $\mathsf{RatHistoryLedgerPolicy}$*; core theorem this block; exactness the two iff rows plus conditional* w*-readback; downstream consumer the* $\mathsf{Rat}^{\uparrow}$ *semantic certificate boundary and* $\mathsf{Real}^{\uparrow}$ *rational observation streams; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 37.79 to the ledger witness. This gives the classifier endpoint equivalence from ρ and from v toward every target history w.

For the reversed-argument equivalence, use Definition 37.55 on both sides. A classifier witness from w to ρ becomes one from ρ to w, crosses the endpoint equivalence just obtained, and is then reversed to a classifier witness from w to v. The converse direction is the same argument with v and ρ exchanged.

Finally apply Definition 37.72 to the ledger witness, obtaining $\mathsf{RatHistoryClassifier}(\rho, v)$. Applying Definition 37.50 to that classifier supplies the positive unary denominator witnesses for ρ and v.

For the first conditional w-readback, apply Definition 37.50 to a witness of $\mathsf{RatHistoryClassifier}(v, w)$ and project the right endpoint. This gives $\mathsf{PositiveUnaryDenominator}(w)$ under the visible-target classifier hypothesis.

For the second conditional w-readback, apply the same theorem to a witness of $\mathsf{RatHistoryClassifier}(w, v)$ and project the left endpoint. This gives $\mathsf{PositiveUnaryDenominator}(w)$ under the reversed visible-endpoint classifier hypothesis.

The two equivalences, the positive ρ and v endpoint witnesses, and the two implication rows assemble the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy two sided endpoint classifier equivalence`

Theorem 37.96 (Rat ledger one-tail classifier saturation)**.** *For histories* $\rho, v, t : \mathsf{BHist}$*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho,$ *holds, then*

$$(\mathsf{RatHistoryClassifier}(\rho, \mathsf{E1}(t)) \iff \mathsf{RatHistoryClassifier}(v, \mathsf{E1}(t)))\wedge(\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), \rho) \iff \mathsf{RatHistoryClass}$$

Its ten-field fit is: traditional target rational ledger one-tail readback; BEDC source $\mathsf{Rat}^{\uparrow}$ *ledger over* BHist*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from ledger endpoint classifier equivalence and one-extension denominator readback; ledger raw-visible denominator edge; core theorem this block; exactness the two displayed iff rows plus tail readback; downstream consumers* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *denominator rows and* $\mathsf{Real}^{\uparrow}$ *rational observation streams; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 37.95 with $w = \mathsf{E1}(t)$. Project the forward one-tail classifier row, the reversed one-tail classifier row, and the positive-denominator witnesses for ρ and v.

For the visible target orientation, assume $\mathsf{RatHistoryClassifier}(v, \mathsf{E1}(t))$. The same two-sided endpoint package gives $\mathsf{PositiveUnaryDenominator}(\mathsf{E1}(t))$ under this hypothesis.

Apply Definition 37.7 to the conditional positive denominator for $\mathsf{E1}(t)$. This reads back the tail witness $\mathsf{Unary}(t)$.

For the visible source orientation, assume $\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), v)$. Project the opposite conditional positive-denominator row from the same endpoint package and apply Definition 37.7 again to obtain $\mathsf{Unary}(t)$.

The two classifier equivalences, the two endpoint denominator witnesses, and the two one-tail readback implications assemble the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy e1 tail classifier saturation`
Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy one tail classifier saturation`

Theorem 37.97 (Rat ledger bidirectional target-positive endpoint package)**.** *For histories* $\rho, v, w :$ BHist*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, then*

$$(\mathsf{RatHistoryClassifier}(\rho, w) \iff \mathsf{RatHistoryClassifier}(v, w))\wedge(\mathsf{RatHistoryClassifier}(w, \rho) \iff \mathsf{RatHistoryClassifier}(w, v))$$

Its ten-field fit is: traditional target rational ledger bidirectional endpoint package; BEDC source $\mathsf{Rat}^{\uparrow}$ *ledger over* BHist*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from ledger classifier extension plus classifier symmetry and transitivity; ledger the raw/visible denominator edge; core theorem this block; exactness the two displayed fibers plus positivity; downstream consumer* $\mathsf{Real}^{\uparrow}$ *stream-prefix and the* $\mathsf{Rat}^{\uparrow}$ *semantic certificate boundary; status* $\mathsf{ProofTarget}$.

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy bidirectional target positive endpoint package`

Proof. Apply Definition 37.79 to the ledger witness. This gives the forward-orientation classifier fiber between ρ and v at the common target w.

For the reversed-argument fiber, apply Definition 37.55 before and after the forward fiber. A witness $\mathsf{RatHistoryClassifier}(w, \rho)$ becomes a witness from ρ to w, crosses the endpoint equivalence, and reverses to a witness from w to v; the converse direction follows by the same symmetric transport.

It remains to package endpoint positivity. Assume $\mathsf{RatHistoryClassifier}(v, w)$. Apply Definition 37.78 to the ledger witness and this visible-target classifier, obtaining $\mathsf{RatHistoryClassifier}(\rho, w)$.

Apply Definition 37.50 to the lifted raw-target classifier for the ρ and w witnesses, and apply the same positive-denominator theorem to the original visible-target classifier for the v and w witnesses. These assemble the first implication.

For $\mathsf{RatHistoryClassifier}(w, v)$, first reverse it by Definition 37.55. The resulting $\mathsf{RatHistoryClassifier}(v, w)$ case is exactly the positivity package just constructed, so the second implication follows. □

Theorem 37.98 (Rat ledger classifier saturation unary package)**.** *For histories* $\rho, v, w : \mathsf{BHist}$*, assume* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and at least one of* $\mathsf{RatHistoryClassifier}(\rho, w)$*,* $\mathsf{RatHistoryClassifier}(v, w)$*,* $\mathsf{RatHistoryClassifier}(w, \rho)$*, or* $\mathsf{RatHistoryClassifier}(w, v)$*. Then*

$$\mathsf{RatHistoryClassifier}(\rho, w) \wedge \mathsf{RatHistoryClassifier}(v, w) \wedge \mathsf{RatHistoryClassifier}(w, \rho) \wedge \mathsf{RatHistoryClassifier}(w, v) \wedge \mathsf{PositiveUr}$$

Its ten-field fit is: traditional target ledger-saturated rational classifier endpoint package; BEDC source $\mathsf{Rat}^{\uparrow}$ *over* BHist*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from classifier symmetry/transitivity and ledger endpoint equivalence; ledger* $\mathsf{RatHistoryLedgerPolicy}$*; core theorem this block; exactness the four-orientation saturated package plus unary endpoint readback; downstream consumer the* $\mathsf{Rat}^{\uparrow}$ *semantic certificate boundary and* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *denominator-unit rows; status* $\mathsf{ProofTarget}$.

Proof. First apply Definition 37.95 to the ledger witness. This gives the forward endpoint equivalence $\mathsf{RatHistoryClassifier}(\rho, w) \iff \mathsf{RatHistoryClassifier}(v, w)$, the reversed endpoint equivalence $\mathsf{RatHistoryClassifier}(w, \rho) \iff \mathsf{RatHistoryClassifier}(w, v)$.

For the ledger endpoint denominators, apply Definition 37.72 to the ledger witness and then Definition 37.50 to the obtained $\mathsf{RatHistoryClassifier}(\rho, v)$. This supplies positive unary denominator witnesses for ρ and v.

It remains to turn the supplied orientation into a single classifier involving w and one ledger endpoint. If the supplied classifier is $\mathsf{RatHistoryClassifier}(\rho, w)$, transport it across the forward endpoint equivalence to obtain $\mathsf{RatHistoryClassifier}(v, w)$, and apply Definition 37.55 to obtain both reversed orientations.

If the supplied classifier is $\mathsf{RatHistoryClassifier}(v, w)$, transport it back across the same endpoint equivalence to obtain $\mathsf{RatHistoryClassifier}(\rho, w)$, and again apply Definition 37.55 to obtain the two reversed orientations.

If the supplied classifier is $\mathsf{RatHistoryClassifier}(w, \rho)$, transport it across the reversed endpoint equivalence to obtain $\mathsf{RatHistoryClassifier}(w, v)$, and apply Definition 37.55 to obtain the two forward orientations.

If the supplied classifier is $\mathsf{RatHistoryClassifier}(w, v)$, transport it back across the reversed endpoint equivalence to obtain $\mathsf{RatHistoryClassifier}(w, \rho)$, and apply Definition 37.55 to obtain the two forward orientations.

In every case one of the assembled classifiers has w as an endpoint. Apply Definition 37.50 to that classifier to obtain $\mathsf{PositiveUnaryDenominator}(w)$, keeping the ledger endpoint witnesses for ρ and v. Apply Definition 37.12 to each of the three positive denominator witnesses and project the unary row. The four classifiers, three positive unary denominator witnesses, and three unary endpoint rows assemble the displayed package. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy classifier saturation unary package`

Theorem 37.99 (Rat ledger visible-target zero-extension exclusion)**.** *For histories* $\rho, v, w, z, z' :$ BHist*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and* $\mathsf{RatHistoryClassifier}(v, w)$ *hold, then*

$$(\mathsf{hsame}(\rho, \mathsf{E0}(z)) \to \mathsf{False}) \wedge (\mathsf{hsame}(w, \mathsf{E0}(z')) \to \mathsf{False}).$$

Its ten-field fit is: traditional target $\mathsf{Rat}^{\uparrow}$ *ledger visible-target zero-branch exclusion; BEDC source* $\mathsf{RatHistoryLedgerPolicy}$ *and* $\mathsf{RatHistoryClassifier}$*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability by ledger classifier extension and classifier endpoint transport; ledger the raw-visible edge; core theorem this block; exactness the two contradictions; downstream consumers* $\mathsf{Field}^{\uparrow}$ *denominator continuation and* $\mathsf{Real}^{\uparrow}$ *stream-prefix readback; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 37.92 to the ledger witness and the visible-target classifier. Its first component gives $\mathsf{RatHistoryClassifier}(\rho, w)$, the classifier whose endpoints are the raw denominator and the target denominator.

Suppose first that $\mathsf{hsame}(\rho, \mathsf{E0}(z))$ holds. Transport the lifted classifier along this endpoint equality to obtain $\mathsf{RatHistoryClassifier}(\mathsf{E0}(z), w)$. The left branch of Definition 37.52 rules out that classifier, giving the first contradiction.

Suppose next that $\mathsf{hsame}(w, \mathsf{E0}(z'))$ holds. Transport the same lifted classifier along the target endpoint equality to obtain $\mathsf{RatHistoryClassifier}(\rho, \mathsf{E0}(z'))$. The right branch of Definition 37.52 rules out that classifier, giving the second contradiction. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible target zero extension exclusion`
Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy visible target e0 endpoint absurd`

Theorem 37.100 (Rat ledger saturated denominator-shape package)**.** *For histories* $\rho, v, w, z_\rho, z_v, z_w :$ BHist*, assume* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and at least one of* $\mathsf{RatHistoryClassifier}(\rho, w)$*,* $\mathsf{RatHistoryClassifier}(v, w)$*,* $\mathsf{RatHistoryClassifier}(w, \rho)$*, or* $\mathsf{RatHistoryClassifier}(w, v)$*. Then all of the following hold:*

1. $\mathsf{UnaryHistory}(\rho)$*,* $\mathsf{UnaryHistory}(v)$*, and* $\mathsf{UnaryHistory}(w)$*;*
2. $\mathsf{hsame}(\rho, \mathsf{e}) \to \mathsf{False}$*,* $\mathsf{hsame}(v, \mathsf{e}) \to \mathsf{False}$*, and* $\mathsf{hsame}(w, \mathsf{e}) \to \mathsf{False}$*;*
3. $\mathsf{hsame}(\rho, \mathsf{E0}(z_\rho)) \to \mathsf{False}$*,* $\mathsf{hsame}(v, \mathsf{E0}(z_v)) \to \mathsf{False}$*, and* $\mathsf{hsame}(w, \mathsf{E0}(z_w)) \to \mathsf{False}$*.*

Its ten-field fit is: traditional target saturated rational denominator-shape readback; BEDC source $\mathsf{Rat}^{\uparrow}$ *ledger over* BHist*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from ledger endpoint equivalence, classifier symmetry/transitivity, positive unary denominator readback, and zero-extension exclusion; ledger the raw-visible denominator edge; core theorem this block; exactness the three endpoint shape package; downstream consumers* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *denominator-unit rows and* $\mathsf{Real}^{\uparrow}$ *stream-prefix readback.*

Proof. Unpack the ledger witness. Its raw carrier gives $\mathsf{PositiveUnaryDenominator}(\rho)$, and visible-carrier transport gives $\mathsf{PositiveUnaryDenominator}(v)$. Case-split on the supplied classifier orientation; in each branch the classifier contains a carrier witness for w, which gives $\mathsf{PositiveUnaryDenominator}(w)$.

For each endpoint, apply Definition 37.12; the unary component gives the displayed $\mathsf{UnaryHistory}$ row, and the nonempty component gives the corresponding contradiction from $\mathsf{hsame}(-, \mathsf{e})$.

It remains to exclude zero-extension presentations. Suppose $\mathsf{hsame}(\rho, \mathsf{E0}(z_\rho))$ holds. Transport $\mathsf{PositiveUnaryDenominator}(\rho)$ along this equality and contradict Definition 37.10.

The visible endpoint v is identical: transport $\mathsf{PositiveUnaryDenominator}(v)$ along $\mathsf{hsame}(v, \mathsf{E0}(z_v))$ and apply Definition 37.10. For w, transport $\mathsf{PositiveUnaryDenominator}(w)$ along $\mathsf{hsame}(w, \mathsf{E0}(z_w))$ and apply the same zero-extension exclusion. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy saturated denominator shape package`

Theorem 37.101 (Rat ledger one-tail denominator-shape readback)**.** *For histories* $\rho, v, t, z_\rho, z_v, z_t$: BHist*, assume* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *and at least one of* $\mathsf{RatHistoryClassifier}(\rho, \mathsf{E1}(t))$*,* $\mathsf{RatHistoryClassifier}(v, \mathsf{E}$ $\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), \rho)$*, or* $\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), v)$*. Then all of the following hold:*

1. $\mathsf{UnaryHistory}(\rho)$*,* $\mathsf{UnaryHistory}(v)$*,* $\mathsf{UnaryHistory}(\mathsf{E1}(t))$*, and* $\mathsf{UnaryHistory}(t)$*;*
2. $\mathsf{hsame}(\rho, \mathsf{e}) \to \mathsf{False}$*,* $\mathsf{hsame}(v, \mathsf{e}) \to \mathsf{False}$*, and* $\mathsf{hsame}(\mathsf{E1}(t), \mathsf{e}) \to \mathsf{False}$*;*
3. $\mathsf{hsame}(\rho, \mathsf{E0}(z_\rho)) \to \mathsf{False}$*,* $\mathsf{hsame}(v, \mathsf{E0}(z_v)) \to \mathsf{False}$*, and* $\mathsf{hsame}(\mathsf{E1}(t), \mathsf{E0}(z_t)) \to \mathsf{False}$*.*

Its ten-field fit is: traditional target one-tail rational denominator readback; BEDC source $\mathsf{RatHistoryLedgerPolicy}$*,* $\mathsf{RatHistoryClassifier}$*,* BHist*, and* $\mathsf{E1}$*; carrier* $\mathsf{RatHistoryCarrier}$*; classifier* $\mathsf{RatHistoryClassifier}$*; stability from one-tail saturation and saturated denominator-shape; ledger the raw-visible denominator edge; core theorem this block; exactness tail readback plus endpoint shape exclusions; standard bridge positive rational denominator tail readback; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 37.100 with $w = \mathsf{E1}(t)$. This gives the unary row for ρ, v, and $\mathsf{E1}(t)$, the three nonempty exclusions, and the three zero-extension exclusions.

It remains to read back the tail witness t. Case-split on the supplied classifier orientation. If the classifier is $\mathsf{RatHistoryClassifier}(\rho, \mathsf{E1}(t))$ or $\mathsf{RatHistoryClassifier}(v, \mathsf{E1}(t))$, it already has $\mathsf{E1}(t)$ as the visible target. Apply Definition 37.96 to obtain the one-tail target readback, and project $\mathsf{UnaryHistory}(t)$.

If the classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), \rho)$ or $\mathsf{RatHistoryClassifier}(\mathsf{E1}(t), v)$, the same saturation theorem is applied in the visible-source orientation. Its source readback projects $\mathsf{UnaryHistory}(t)$. Combining this tail row with the package obtained above gives the displayed conjunctions. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy one tail denominator shape readback`

Theorem 37.102 (RatUp bridged closure obligation package)**.** *The public RatUp certificate may be read as a paper-side standard bridge only after the following four witnesses are supplied over concrete BEDC objects:*

1. *a standard rational-constructor map from signed unary numerator histories and positive unary denominator histories into* $\mathsf{RatHistoryCarrier}$*, compatible with* $\mathsf{RatHistoryClassifier}$*;*
2. *normalized representative and readback rows stable under* Cont*,* $\mathsf{UnaryHistory}$*, and* hsame*;*
3. *arithmetic and order law rows descending through* $\mathsf{RatHistoryClassifier}$ *while preserving the positive-denominator ledger;*
4. *a no-hidden-leakage row stating that the ledger hides only numerator-denominator provenance.*

These four rows are exactly the paper-side bridge obligation consumed by RatUp bridgedClosed.

Proof. Unpack the roadmap ten-field template used throughout this chapter. The traditional target is the standard rational object; the BEDC source is RatHistoryCarrier with classifier RatHistoryClassifier; the ledger is the public denominator-shape record; and the status row is the bridge obligation rather than an additional Lean marker.

The existing semantic certificate Definition 37.90 supplies the public carrier/classifier interface. The denominator-shape readback results ending in Definition 37.101 supply the positive-denominator visibility needed for normalized representatives and the ledger side condition. What remains for a standard rational bridge is therefore precisely the displayed rational-constructor compatibility row, the normalized representative/readback row, the arithmetic and order descent row, and the no-hidden-leakage row.

Conversely, once those four witnesses are present, every field demanded by the ten-field bridge reading has a concrete BEDC source: constructor entry, classifier compatibility, stable readback, law descent, positive-denominator ledger preservation, and hidden-information control. Hence the four displayed rows are exactly the paper-side bridge obligation consumed by RatUp bridgedClosed. □

Lean4 ✓ `BEDC.Derived.RatUp.RatUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$**.**

Constructive story: A rational lives, in this construction, as a denominator history together with a hidden but witnessed numerator-and-sign pair: the denominator is required to be a positive unary tail (a history of the form E1(tail) with the tail itself unary, so the denominator is at least one), and the existence of an underlying triple — sign mark, numerator history, denominator — is what licenses the denominator into the carrier in the first place. Two pairs that traditional math would identify, such as $(2, 4)$ and $(1, 2)$, remain distinct denominators here; rather than collapse them into an equivalence class, the classifier asks whether the underlying signed-numerator data satisfies the cross-product condition, and history-sameness then carries that recognition through any context where the two pairs appear interchangeably. Because history-equality is an equivalence and the carrier is preserved by it, the five name-certificate obligations are essentially a free consequence of how the carrier was set up. The signature gap of a $\mathsf{Rat}^\uparrow$ token p collects denominators d for which the source-domain admits d, the signature relation pairs d (together with its implicit numerator and sign) with an answer denominator and an evidence ledger, and the token-introduction relation maps that answer to p — concretely, this means the underlying triple names the same rational as p encodes, modulo the cross-product equivalence. The chapter pairs $\mathsf{Rat}^\uparrow$ with a four-witness bridged-closure obligation package — representative stability, sign normalization, denominator positivity, and downstream-consumer readback — and registers the whole construction under the name $\mathsf{Rat}^\uparrow$; the rational lives entirely as a denominator with witnesses, never as a class of equivalent fractions, and no quotient is ever formed.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$, maturePackageClosed)

Scope closed: The RatUp mature paper scope is the RatHistoryCarrier/RatHistoryClassifier semantic certificate, representative stability, positive denominator-tail readbacks, and the four bridge-obligation witnesses of Definition 37.102.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.RatUp.rat\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full rational field theory, decidable order, density, Archimedean properties, and quotient construction.

Upgrade path: No further upgrade – chapter at top of formal axis.

The $\mathsf{Rat}^\uparrow$ positive-denominator surface is the arithmetic input read by the dyadic real-completion route of chapter 291, without quotient rational equality or host real data.

Theorem 37.103 (RatUp DyadicRatCore denominator handoff)**.** *Every* $\mathsf{DyadicRatCore}^\uparrow$ *consumer of a signed mantissa/exponent packet first reads the normalized* $\mathsf{Rat}^\uparrow$ *positive-denominator surface through the displayed* $\mathsf{Int}^\uparrow$ *numerator, unary positive denominator,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *rows. This route supplies the rational mantissa and denominator evidence consumed by chapter 291, but it adds no quotient rational equality, selected real limit, host equality, or* $\mathsf{Real}^\uparrow$ *completeness coordinate.*

Proof. Project the rational carrier of Definition 37.2. Its visible surface is an $\mathsf{Int}^{\uparrow}$ numerator together with a denominator history that is unary and nonempty, equivalently a positive unary denominator by Definition 37.13.

The denominator rows are stable under the Rat-side transport already recorded in Definition 37.5 and Definition 37.6. Hence a consumer may move the displayed denominator through hsame without changing the positive unary evidence read from the carrier.

Next follow the ledger route. The one-tail and saturated denominator readback theorems in Definition 37.101 expose the unary tail, nonempty endpoint, and zero-extension exclusions that a finite Cont replay can inspect. The Pkg and NameCert rows keep this denominator evidence attached to the Rat provenance rather than turning it into a host rational object.

Finally cite the DyadicRatCore carrier handoff in Definition 291.1 and Definition 291.2. That consumer reads only the mantissa, exponent, denominator ledger, Cont, Pkg, and source certificate rows. Since neither the Rat carrier nor the DyadicRatCore packet contains a quotient rational equality, selected real limit, host equality, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, the displayed positive-denominator route is the full handoff surface. □

Theorem 37.104 (RatUp L10 dyadic denominator gateway)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet whose dyadic face reads a* $\mathsf{Rat}^{\uparrow}$ *positive-denominator surface. Then the denominator row consumed by $\mathcal{E}$ factors through the finite route*

$$\mathsf{Rat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization uses only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows attached to the Rat, dyadic, StreamName, RegSeqRat, and Real faces.*

Consequently the gateway exports no quotient rational equality, quotient stream equality, selected limit, host real equality, choice row, metric-completeness datum, bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with Definition 37.103. It reads the positive unary denominator from the $\mathsf{Rat}^{\uparrow}$ carrier through the displayed numerator, denominator ledger, hsame transport, Cont replay, Pkg provenance, and NameCert rows, and hands exactly that finite surface to the $\mathsf{DyadicRatCore}^{\uparrow}$ consumer.

Next attach the current Phase Real and Completion exit packet. By Definition 38.796, such a packet reads the four local faces objectwise, with support restricted to

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Thus the dyadic face that receives the Rat denominator handoff is the same finite dyadic row available to the Real-and-Completion exit read.

Now push the row through the terminal route. Apply Definition 126.615 to force the public consumer order through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and apply Definition 301.236 to keep the StreamName window, dyadic ledger, RegSeqRat readback, and Real finite-section readback on one common finite source section.

The displayed route therefore carries the Rat positive-denominator row only as finite BEDC support data. The exclusion clauses in the Rat handoff, ordered StreamName gate, RegSeqRat coherence theorem, and Real current-phase threshold remove every non-displayed coordinate: quotient rational equality, quotient stream equality, selected limits, host real equality, choice, metric completeness, bridge data, and $\mathsf{Real}^{\uparrow}$ completeness are not part of the gateway surface. □

The $\mathsf{Real}^{\uparrow}$ index reads its terminal seal surface through the L10 finite-name route of chapter 126, chapter 301, chapter 291, and chapter 1180; it exports no quotient stream, host real equality, selected limit, countable choice, or ambient completeness claim.

Chapter 38

Certificate Obligations for $\mathsf{Real}^{\uparrow}$

The real number interface $\mathsf{Real}^{\uparrow}$ is specified here as a certificate-obligation schema over sealed $\mathsf{RegSeqRat}$ stream names. These names are built from $\mathsf{DyadicRatCore}$ observations through $\mathsf{StreamName}^{\uparrow}$, then sealed with regular-sequence data and a closeness classifier. The classical Dedekind / Cauchy completion is outside this schema because quotient and completion infrastructure is not exported by the mathlib-free BEDC kernel; the interface name is available only through the dyadic rational core, certified stream names, regular-sequence certificates, and their classifier.

38.1 Carrier definition

Definition 38.1 (Real carrier)**.** *Use a sealed* $\mathsf{RegSeqRat}$ *name. Its observations form a* $\mathsf{StreamName}^{\uparrow}$ *stream over* $\mathsf{DyadicRatCore}$*, and its regularity certificate is exposed only through the* $\mathsf{Real}^{\uparrow}$ *seal.*

Definition 38.2 (Constant real history carrier)**.** *A history is in the constant real history carrier when it is internally the same as an* $\mathsf{E1}$*-sealed rational history:*

$$\mathsf{RealConstantHistoryCarrier}(h) \iff \exists d,\ \mathsf{hsame}(h, \mathsf{E1}(d)) \ \wedge\ \mathsf{RatHistoryCarrier}(d).$$

Lean4 (definition) `BEDC.Derived.RealUp.RealConstantHistoryCarrier`

Theorem 38.3 (Constant real carrier at an e-one seal)**.** *For every history* d*,*

$$\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)) \iff \mathsf{RatHistoryCarrier}(d).$$

Proof. Unpack the constant carrier witness. One-constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$ into $\mathsf{hsame}(d, d')$, and rational history carrier transport moves the witness back along that equality. The converse uses d itself as the exposed rational history and reflexivity of hsame. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryCarrier e1 iff rat`

Theorem 38.4 (Constant real carrier unary denominator context closure)**.** *Let* $d, p, t, m, o : \mathsf{Hist}$*. If*

$$\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)),$$

p and t are unary histories, and

$$\mathsf{Cont}(p, d, m), \qquad \mathsf{Cont}(m, t, o),$$

then

$$\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(o)).$$

Proof. Unseal the displayed constant real carrier through Definition 38.3 to obtain $\mathsf{RatHistoryCarrier}(d)$. RatUp unary-denominator context closure gives the rational carrier of $\mathsf{append}(p, \mathsf{append}(d, t))$. The two continuation witnesses and associativity of append identify this normal endpoint with o. Transport the rational carrier along that endpoint identification, and reseal it as a constant real carrier at $\mathsf{E1}(o)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryCarrier cont unary denominator context closed`

Theorem 38.5 (Constant real carrier e-zero exclusion)**.** *For every history tail t,*

$$\mathsf{RealConstantHistoryCarrier}(\mathsf{E0}(t)) \Longrightarrow \bot.$$

Proof. Unpack the constant real carrier witness. It displays $\mathsf{E0}(t)$ as $\mathsf{E1}(d)$ for some rational history d. BEDC history no-confusion between zero-extension and one-extension constructors eliminates that display. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryCarrier e0 absurd`

Theorem 38.6 (Constant real carrier empty exclusion)**.** *The empty history is not a constant real history carrier:*

$$\mathsf{RealConstantHistoryCarrier}(\mathsf{e}) \Longrightarrow \bot.$$

Proof. Unpack the constant real carrier witness. It displays e as an $\mathsf{E1}$-sealed rational history, and BEDC history no-confusion between the empty constructor and the one-extension constructor eliminates that display. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryCarrier empty absurd`

38.2 The certificate-obligation schema

Definition 38.7 (Real source specification)**.** *The source has three supplied layers:* $\mathsf{DyadicRatCore}$ *observations,* $\mathsf{StreamName}^{\uparrow}$ *stream naming, and the* $\mathsf{RegSeqRat}$ *certificate over those observations.*

Definition 38.8 (Real pattern specification)**.** *The pattern is sealed regular-sequence generation: every finite observation is a certified dyadic rational, stream observations are stable pointwise, and the* $\mathsf{RegSeqRat}$ *regularity witness supplies the dyadic Cauchy bound.*

Definition 38.9 (Real classifier specification)**.** *The classifier is* $\mathsf{RegSeqRat}$ *closeness: two sealed real names classify together when their dyadic regular-sequence witnesses supply comparison certificates at every requested dyadic tolerance.*

Definition 38.10 (Constant real history classifier)**.** *Two constant real histories classify together when each exposes an* $\mathsf{E1}$*-sealed rational history and the exposed rational histories classify:*

$$\mathsf{RealConstantHistoryClassifier}(h, k) \iff \exists d, e,\ \mathsf{hsame}(h, \mathsf{E1}(d)) \wedge \mathsf{hsame}(k, \mathsf{E1}(e)) \wedge \mathsf{RatHistoryClassifier}(d, e).$$

Lean4 (definition) `BEDC.Derived.RealUp.RealConstantHistoryClassifier`

Theorem 38.11 (Constant real classifier at e-one seals)**.** *For all histories d and e,*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)) \iff \mathsf{RatHistoryClassifier}(d, e).$$

Proof. Eliminate the two seal witnesses. One-constructor inversion gives the two tail hsame witnesses, and rational classifier transport carries the exposed classifier back to the displayed tails. The reverse direction chooses the same two rational histories and uses reflexivity at both seals. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 iff rat`

Theorem 38.12 (Constant real classifier unary-tail closure)**.** *For all histories d, e and unary history t,*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)) \Longrightarrow \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d \cdot t), \mathsf{E1}(e \cdot t)).$$

Proof. Read the two sealed constant-real endpoints back through Definition 38.11 as a rational history classifier between d and e. The rational classifier is closed under appending the same unary denominator tail to both exposed histories, and the same e-one bridge repacks the appended rational classifier as a constant-real classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append unary tail closed`

Theorem 38.13 (Constant real classifier common-tail cancellation)**.** *For all histories d, e, t,*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d \cdot t), \mathsf{E1}(e \cdot t)) \Longrightarrow d\mathsf{hsame}e.$$

Proof. Read the sealed constant-real classifier through Definition 38.11 as a rational history classifier on the two appended rational endpoints. Its endpoint hsame field identifies $d \cdot t$ with $e \cdot t$, and right cancellation of the common BEDC history tail t exposes $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append common tail cancel`

Theorem 38.14 (Constant real classifier common-head cancellation)**.** *For all histories h, d, e,*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(h \cdot d), \mathsf{E1}(h \cdot e)) \Longrightarrow d\mathsf{hsame}e.$$

Proof. Read the sealed constant-real classifier through Definition 38.11 as a rational history classifier on the two appended rational endpoints. Its endpoint hsame field identifies $h \cdot d$ with $h \cdot e$, and left cancellation of the common BEDC history head h exposes $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append common head cancel`

Theorem 38.15 (Constant real classifier common-context cancellation)**.** *For all histories $left, right, d, e$,*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(left \cdot (d \cdot right)), \mathsf{E1}(left \cdot (e \cdot right))) \Longrightarrow d\mathsf{hsame}e.$$

Proof. First cancel the shared visible left context using the common-head cancellation theorem. This yields $\mathsf{hsame}(d \cdot right, e \cdot right)$. Right append cancellation then removes the shared suffix $right$ and exposes $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append common context cancel`

Theorem 38.16 (Constant real classifier double-seal readback)**.** *If* $\mathsf{RealConstantHistoryClassifier}(h, k)$ *holds and the two endpoints are displayed as* $\mathsf{E1}(\mathsf{E1}(d))$ *and* $\mathsf{E1}(\mathsf{E1}(e))$*, then the exposed tails d and e are unary histories and satisfy* $d\mathsf{hsame}e$*.*

Proof. Transport the classifier along the two displayed endpoint equalities to obtain RealConstantHistoryClassifier(E1
The constant real e-one equivalence reads this as a rational classifier between $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$. Rational e-one tail readback then exposes the two unary tail carriers and the tail equality $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 pair readback`

Theorem 38.17 (Constant real classifier endpoint carriers)**.** *If two constant real histories classify together, then both endpoints are in the constant real history carrier:*

$$\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow \mathsf{RealConstantHistoryCarrier}(h) \wedge \mathsf{RealConstantHistoryCarrier}(k).$$

Proof. Unpack the classifier witness into rational histories d, e, endpoint displays $\mathsf{hsame}(h, \mathsf{E1}(d))$ and $\mathsf{hsame}(k, \mathsf{E1}(e))$, and a rational history classifier between d and e. The rational classifier stores rational carrier witnesses at both exposed endpoints, so pairing those carrier witnesses with the two endpoint displays gives the two constant real history carriers. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier endpoint carriers`

Theorem 38.18 (Constant real classifier endpoint transport)**.** *If constant real histories h, k classify together, and $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, then h' and k' classify together.*

Proof. Unpack the two sealed rational witnesses. Symmetry and transitivity of hsame move the displayed endpoint equalities from h, k to h', k', while the rational classifier witness remains attached to the same sealed tails. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier endpoint transport`

Theorem 38.19 (Constant real classifier e-one tail readback)**.** *If a constant real classifier is displayed at two visible rational seals, then the displayed tails carry the rational history classifier:*

$$\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow \mathsf{hsame}(h, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(k, \mathsf{E1}(e)) \Longrightarrow \mathsf{RatHistoryClassifier}(d,e).$$

Proof. Transport the classifier along the two displayed endpoint equalities to obtain RealConstantHistoryClassifier(E1
The e-one seal readback theorem for constant real classifiers then exposes the rational classifier on the two tails. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 tail readback`

Theorem 38.20 (Constant real classifier e-one denominator package)**.** *For histories h, k, d, e : BHist, suppose*

$$\mathsf{RealConstantHistoryClassifier}(h,k), \qquad h\mathsf{hsame}\mathsf{E1}(d), \qquad k\mathsf{hsame}\mathsf{E1}(e).$$

Then the displayed rational tails satisfy the full denominator shape package:

1. $\mathsf{RatHistoryClassifier}(d,e)$*;*
2. $\mathsf{PositiveUnaryDenominator}(d)$ *and* $\mathsf{PositiveUnaryDenominator}(e)$*;*
3. $\mathsf{UnaryHistory}(d)$ *and* $\mathsf{UnaryHistory}(e)$*;*
4. $d\mathsf{hsame}e \Longrightarrow \bot$ *and* $e\mathsf{hsame}e \Longrightarrow \bot$*.*

In the ten-field fit, the traditional target is constant rational endpoint shape readback; the BEDC source is BHist *e-one rational seals; the carrier is* RealConstantHistoryCarrier*; the classifier is* RealConstantHistoryClassifier*; the stability obligations are tail readback and RatUp denominator projections; the ledger is exposed denominator shape; the core theorem is this package; the exactness predicate is rational classifier plus positive unary nonempty tails; the standard bridge is constant rational real names; and the status is* ProofTarget.

Proof. Apply Definition 38.19 to the real classifier and the two displayed e-one endpoint equalities. This gives $\mathsf{RatHistoryClassifier}(d, e)$.

Next apply Definition 37.50 to that rational classifier. The result supplies the positive unary denominator witnesses for both displayed tails.

Finally apply Definition 37.12 to each positive-denominator witness. The two applications give the unary-history rows and the two exclusions against e, so the listed package follows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 denominator package`

Theorem 38.21 (Constant real classifier unary denominator context closure)**.** *Let* $d, e, p_D, p_E, t_D, t_E, m_D, m_E, o_D,$ BHist*. If*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)),$$

p_D *and* t_D *are unary histories,* $p_D \mathsf{hsame} p_E$*,* $t_D \mathsf{hsame} t_E$*, and the displayed continuation witnesses are*

$$\mathsf{Cont}(p_D, d, m_D), \quad \mathsf{Cont}(m_D, t_D, o_D), \quad \mathsf{Cont}(p_E, e, m_E), \quad \mathsf{Cont}(m_E, t_E, o_E),$$

then

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(o_D), \mathsf{E1}(o_E)).$$

Proof. Use Definition 38.11 to expose the rational classifier on d and e. The RatUp unary-denominator context closure classifies the associativity-normal endpoints $\mathsf{append}(p_D, \mathsf{append}(d, t_D))$ and $\mathsf{append}(p_E, \mathsf{append}(e, t_E))$. The four displayed continuation witnesses identify o_D and o_E with those normal endpoints through associativity of append. Transport the rational classifier along those identifications, then reseal the two endpoints with Definition 38.11. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier cont unary denominator context closed`

Theorem 38.22 (Constant real classifier invalid endpoint exclusions)**.** *For every history* h *and history tail* t*, the constant real-history classifier excludes every endpoint shape that is not an* E1*-sealed rational history:*

1. $\mathsf{RealConstantHistoryClassifier}(\mathsf{E0}(t), h) \Longrightarrow \bot$*;*
2. $\mathsf{RealConstantHistoryClassifier}(h, \mathsf{E0}(t)) \Longrightarrow \bot$*;*
3. $\mathsf{RealConstantHistoryClassifier}(\mathsf{e}, h) \Longrightarrow \bot$*;*
4. $\mathsf{RealConstantHistoryClassifier}(h, \mathsf{e}) \Longrightarrow \bot$*.*

In the ten-field fit, the traditional target is sealed constant-real endpoint no-confusion; the BEDC source is BHist *rational seals; the carrier is* RealConstantHistoryCarrier*; the classifier is* RealConstantHistoryClassifier*, the stability obligations are endpoint carrier projection and carrier exclusions; the ledger is invalid endpoint shape; the core theorem is this block; the exactness predicate is impossible non-*E1 *endpoint; the standard bridge is constant rational real names; and the status is* ProofTarget.

Proof. 1. Apply Definition 38.17 to the classifier witness. Its left projection gives $\mathsf{RealConstantHistoryCarrier}(\mathsf{E0}(t))$.

Apply Definition 38.5.

2. Apply Definition 38.17 to the classifier witness. Its right projection gives RealConstantHistoryCarrier(E0(t)). Apply Definition 38.5.
3. Apply Definition 38.17 to the classifier witness. Its left projection gives RealConstantHistoryCarrier(e). Apply Definition 38.6.
4. Apply Definition 38.17 to the classifier witness. Its right projection gives RealConstantHistoryCarrier(e). Apply Definition 38.6.

□

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier invalid endpoint absurd`

Theorem 38.23 (Constant real classifier equivalence fields)**.** *The constant real history classifier supplies the equivalence fields carried by the constant-real certificate package:*

1. *if* RealConstantHistoryCarrier(h)*, then* RealConstantHistoryClassifier(h, h)*;*
2. *if* RealConstantHistoryClassifier(h, k)*, then* RealConstantHistoryClassifier(k, h)*;*
3. *if* RealConstantHistoryClassifier(h, k) *and* RealConstantHistoryClassifier(k, ℓ)*, then* RealConstantHistoryClassifier(h,

In the ten-field fit, the traditional target is constant rational embedding into Real$^{\uparrow}$*; the BEDC source is BHist rational histories; the carrier is* RealConstantHistoryCarrier*; the classifier is* RealConstantHistoryClassifier*; the stability obligation is the equivalence fields above; the ledger is the sealed rational tail; the core theorem is this displayed package; exactness is inherited from the rational classifier; the standard bridge is constant rational real names; and the status is* ProofTarget*.*

Proof. For reflexivity, unpack RealConstantHistoryCarrier(h) as a sealed rational tail d with hsame(h, E1(d)) and RatHistoryCarrier(d). The rational carrier supplies the reflexive rational classifier on d, using hsame reflexivity on the exposed tail. Packing the same sealed tail twice and using the displayed endpoint witness on both sides gives RealConstantHistoryClassifier(h, h).

For symmetry, unpack RealConstantHistoryClassifier(h, k) into sealed rational tails d, e, endpoint displays for h and k, and RatHistoryClassifier(d, e). Keep the same endpoint displays in the opposite order and apply RatHistoryClassifier_symm to the rational classifier.

For transitivity, unpack the two classifiers into tails d, e and e', f. The two middle displays for k compare E1(e) and E1(e') by hsame symmetry and transitivity; one-constructor inversion turns that comparison into hsame(e, e'). Transport the right rational classifier from e' to e, then compose it with the left rational classifier by RatHistoryClassifier_trans. Repacking the outer endpoint displays for h and ℓ gives the required classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier equivalence fields`

Theorem 38.24 (Constant real semantic name certificate)**.** *The constant real history carrier and classifier form a* SemanticNameCert *whose source, pattern, and ledger predicates are all* RealConstantHistoryCarrier*, and whose classifier is* RealConstantHistoryClassifier*.*

Proof. Use the e-one rational denominator witnessed by e to inhabit the source carrier. The classifier equivalence fields of Definition 38.23 supply reflexivity, symmetry, and transitivity. Endpoint-carrier readback from Definition 38.17 supplies carrier transport along the classifier. Since the pattern and ledger predicates are the same concrete carrier, their soundness rows are direct. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistory semanticNameCert`

Definition 38.25 (Real stream classifiers)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, the stream classifier is pointwise rational-history classification. The finite prefix classifier records the same data recursively through a selected observation index.*

Lean4 (definition) `BEDC.Derived.RealUp.RealStreamClassifier`
Lean4 (definition) `BEDC.Derived.RealUp.RealStreamPrefixClassifier`

Definition 38.26 (Real unary-observation stream classifier)**.** *For rational-observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, the unary-observation stream classifier is pointwise rational-history classification over certified unary BEDC observation histories:*

$$\mathsf{RealUnaryStreamClassifier}(s, t) \iff \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{RatHistoryClassifier}(s(n), t(n)).$$

Lean4 (definition) `BEDC.Derived.RealUp.RealUnaryStreamClassifier`

Theorem 38.27 (Constant rational streams bridge StreamNameUp and RealUp)**.** *For histories $d, e : \mathsf{Hist}$, the RatUp point classifier, the StreamNameUp constant-stream classifier, the RealUp unary-observation stream classifier, and the constant real-history classifier at e-one seals are equivalent:*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e).$$

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \iff \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d$$

$$\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \iff \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e))$$

Proof. For the first equivalence, the forward direction is exactly the constant classifier part of Definition 126.48. Conversely, unfold Definition 126.2 and read the pointwise classifier at the empty unary observation history e using its unary witness; the two constant stream values are d and e.

For the second equivalence, unfold Definition 126.2 and Definition 38.26. Both predicates are the same pointwise requirement over certified unary observation histories, so each direction is identity transport of the displayed witness.

For the last equivalence, unfold Definition 38.26 and $\mathsf{RatConstStream}$. Reading at the empty unary observation gives $\mathsf{RatHistoryClassifier}(d, e)$, while a rational point classifier supplies the same pointwise witness at every unary observation.

Apply Definition 38.11 to identify that rational point classifier with $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e))$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier constant prefix bridge iff`
Lean4 ✓ `BEDC.Derived.RealUp.StreamNameReal constant prefix bridge`
Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier constant streamname bridge`
Lean4 ✓ `BEDC.Derived.RealUp.RealConstantStream streamName bridge`

Theorem 38.28 (Reindexed constant rational streams bridge StreamNameUp and RealUp)**.** *For histories $d, e : \mathsf{Hist}$ and any observation map $r : \mathsf{Hist} \to \mathsf{Hist}$, the RatUp point classifier, the StreamNameUp classifier of reindexed constant streams, the RealUp unary-observation stream classifier of the same reindexed streams, and the constant real-history classifier at e-one seals are equivalent:*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r).$$

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r) \iff \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStre}$$

$$\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r) \iff \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}$$

Proof. For the first equivalence, unfold RatConstStream and Definition 126.2. The forward direction reuses the supplied $\mathsf{RatHistoryClassifier}(d, e)$ at every certified unary observation; the observation map is ignored because both reindexed constant streams still evaluate to their hidden seeds d and e. Conversely, read the stream-name classifier at the empty unary observation history e.

For the second equivalence, unfold Definition 126.2 and Definition 38.26. After expanding the reindexed constant streams, both predicates ask for the same pointwise RatUp classifier over certified unary observations, so each direction transports the displayed pointwise witness unchanged.

For the last equivalence, the forward direction again reads the unary stream classifier at e to recover $\mathsf{RatHistoryClassifier}(d, e)$, then applies Definition 38.11. In the reverse direction, the same theorem unseals the e-one classifier as the RatUp point classifier, and the constant observation maps reuse it at every certified unary observation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantStream reindexed streamName bridge`

The reindexed constant-stream certificate fit has ten fields: traditional target constant rational real names under observation reindexing; BEDC source RatUp plus StreamNameUp plus RealUp; carrier the reindexed constant rational observation maps and e-one real seals; classifier the three displayed bridge rows; stability from ignoring the observation map in constant streams; ledger hidden rational seed plus observation map; core theorem Definition 38.28; exactness the three equivalences; standard bridge target constant rational real names under reindexed observation; downstream use by the RealUp dyadic embedding theorem.

Theorem 38.29 (Independent reindexed constant rational streams bridge StreamNameUp and RealUp)**.** *For histories $d, e : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, the RatUp point classifier, the StreamNameUp classifier of independently reindexed constant streams, the RealUp unary-observation stream classifier of the same streams, and the constant real-history classifier at e-one seals are equivalent:*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q).$$

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q) \iff \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStre}$$

$$\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q) \iff \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}$$

Proof. For the first equivalence, unfold RatConstStream and Definition 126.2. In the forward direction, a RatUp point classifier for d and e supplies the required pointwise classifier at every certified unary observation, because the two reindexed constant streams ignore r and q and return their hidden seeds. In the reverse direction, evaluate the stream-name classifier at the empty unary observation history e.

For the second equivalence, unfold Definition 126.2 and Definition 38.26. The two predicates reduce to the same pointwise RatUp classifier over certified unary observation histories after the constant streams discard their separate observation schedules, so each direction transports the displayed witness unchanged.

For the final equivalence, the forward direction reads the unary stream classifier at e to obtain $\mathsf{RatHistoryClassifier}(d, e)$ and then uses Definition 38.11. Conversely, that theorem turns the e-one real-history classifier back into the RatUp point classifier, which is reused at every certified unary observation for the two independently scheduled constant streams. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantStream independent reindexed streamName bridge`

The independent reindexed constant-stream certificate fit has ten fields: traditional target constant rational real names with separately sampled endpoint observations; BEDC source RatUp

plus StreamNameUp plus RealUp; carrier the two reindexed constant rational observation maps and e-one real seals; classifier the three displayed bridge rows; stability from ignoring both observation maps in constant streams; ledger the two observation maps and the hidden rational seeds; core theorem Definition 38.29; exactness the three equivalences; standard bridge target constant rational real names under independent endpoint observation schedules; downstream use by the RealUp dyadic embedding under independent observation schedules.

Theorem 38.30 (Constant rational stream carriers bridge StreamNameUp and RealUp)**.** *For every history $d : \mathsf{Hist}$, the RatUp point carrier, the StreamNameUp constant-stream carrier, and the RealUp constant real-history carrier at the e-one seal are equivalent:*

$$\mathsf{RatHistoryCarrier}(d) \iff \mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d).$$

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d) \iff \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)).$$

Proof. The first equivalence is the carrier row of Definition 126.55, read symmetrically: the StreamNameUp constant carrier is exactly the RatUp carrier of its hidden constant seed.

For the second equivalence, compose that carrier row with Definition 38.3. A StreamNameUp constant carrier first recovers $\mathsf{RatHistoryCarrier}(d)$, then the e-one theorem seals it as $\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}($ the reverse direction unseals the RealUp constant carrier to $\mathsf{RatHistoryCarrier}(d)$ and repacks it as the StreamNameUp constant carrier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantStreamCarrier streamName bridge`

The carrier certificate fit has ten fields: traditional target constant rational real carrier embedding; BEDC source RatUp histories plus StreamNameUp observations; carrier the displayed constant stream and e-one real seal; classifier inherited from Definition 38.27; stability from RatUp carrier transport and RealConstant carrier transport; ledger the hidden rational seed and observation witness; core theorem Definition 38.30; exactness the two displayed equivalence rows; standard bridge constant rational real name; status $\mathsf{ProofTarget}$.

Theorem 38.31 (Constant rational embedding certificate)**.** *The concrete map*

$$d \longmapsto (\mathsf{RatConstStream}_d, \mathsf{E1}(d))$$

from a RatUp history seed $d : \mathsf{Hist}$ supplies the constant rational embedding into RealUp. Its carrier exactness is the chain

$$\mathsf{RatHistoryCarrier}(d) \iff \mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d) \iff \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)$$

Its classifier exactness, for histories $d, e : \mathsf{Hist}$, is the chain

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \iff \mathsf{RealUnar}$$

The certificate therefore packages the hidden rational seed, the constant StreamNameUp observation map, and the e-one RealUp seal as one concrete BHist-anchored rational-to-real name.

Proof. For carrier exactness, use Definition 38.30. Its two displayed rows identify the RatUp carrier with the StreamNameUp constant-stream carrier, and then identify that carrier with the RealUp constant history carrier at the e-one seal.

For classifier exactness, use Definition 38.27. Its three displayed rows identify the RatUp point classifier, the StreamNameUp constant-stream classifier, the RealUp unary stream classifier, and the RealUp constant history classifier at the paired e-one seals.

The endpoint part of the last row is the Real constant classifier equivalence and transport mechanism: Definition 38.11 turns an e-one sealed RealUp classifier into the underlying RatUp classifier and back, while Definition 38.18 preserves the same classifier under endpoint-equivalent real histories. These facts repack the two chains as one certificate for the displayed map. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant rational embedding certificate`

The constant rational embedding certificate has ten fields: traditional target constant rational embedding; BEDC source RatUp plus StreamNameUp plus RealUp; carrier the constant stream and e-one seal; classifier the displayed equivalence chain; stability inherited from RatUp classifier fields and Real constant endpoint transport; ledger hidden rational seed plus unary observation; core theorem Definition 38.31; exactness the carrier and classifier chains; standard bridge ordinary rational constants; status ProofTarget.

Theorem 38.32 (Reindexed constant rational stream carriers bridge StreamNameUp and RealUp)**.** *For every history d : Hist and any observation map r : Hist $\to$ Hist, the RatUp point carrier, the StreamNameUp carrier of the reindexed constant stream, and the RealUp constant real-history carrier at the e-one seal are equivalent:*

$$\mathsf{RatHistoryCarrier}(d) \iff \mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r).$$

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r) \iff \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)).$$

Proof. Unfold Definition 126.1 and RatConstStream. A carrier of d supplies the carrier at every certified unary observation because the reindexed constant stream still returns the seed d. Conversely, reading the stream carrier at e recovers the RatUp carrier of d. Composing this row with Definition 38.3 gives the e-one RealUp carrier row. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantStreamCarrier reindexed streamName bridge`

Theorem 38.33 (Real constant embedding terminal four-face consumption)**.** *The concrete constant rational embedding*

$$d \longmapsto (\mathsf{RatConstStream}_d, \mathsf{E1}(d))$$

from Definition 38.31 is accepted as a $\mathsf{Real}^{\uparrow}$ *public seal read only through the four Phase Real and Completion faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Its visible support rows are precisely the BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by those faces. The consumption route therefore exports no quotient stream equality, selected host limit, ambient completeness row, metric-completeness datum, countable choice, or standard bridge row.*

Proof. Start with Definition 38.31. It packages the hidden rational seed d, the constant stream $\mathsf{RatConstStream}_d$, and the e-one seal $\mathsf{E1}(d)$ as one concrete BHist-anchored rational-to-real name.

Apply Definition 301.132. The same constant source reaches the local $\mathsf{Real}^{\uparrow}$ seal only after the constant $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius rows, and the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier ledger have been read through their displayed support rows.

Next apply Definition 126.172. This keeps the constant observation surface at the pointwise StreamName boundary and leaves regularity, dyadic seal data, and the Real seal classifier as downstream obligations rather than additional source coordinates.

Finally use Definition 38.534. That terminality theorem reads every accepted public seal or completion-facing route through exactly the four displayed faces with BHist, hsame, Cont, Pkg, and NameCert support. Hence the constant rational embedding is consumed by the same terminal four-face boundary, and the excluded quotient, selected-limit, completeness, choice, metric-completeness, and standard-bridge rows cannot replace any face of the route. □

Theorem 38.34 (Real constant embedding terminal four-face pullback)**.** *Any two accepted presentations of the concrete constant rational embedding which agree on the displayed*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

faces and on the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows determine the same public* $\mathsf{Real}^{\uparrow}$ *seal route. Conversely, any terminal four-face packet with those support rows determines the constant embedding seal read.*

The pullback exports no quotient stream equality, selected host limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, or standard bridge row.*

Proof. First apply Definition 38.33. Each accepted presentation of the constant rational embedding is already consumed only through the four displayed faces and the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 301.132. Agreement on the constant $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint data, the constant $\mathsf{StreamName}^{\uparrow}$ schedule, and the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier ledger gives one visible handoff surface for the $\mathsf{Real}^{\uparrow}$ seal.

Apply Definition 126.172. This keeps the constant observation at the pointwise StreamName boundary and leaves the dyadic, regularity, and Real seal rows as the downstream four-face packet rather than as extra source coordinates.

Finally invoke Definition 38.534. Two accepted reads that agree on the four faces and the listed support rows have the same terminal finite route to the public $\mathsf{Real}^{\uparrow}$ seal. In the reverse direction, a terminal packet carrying exactly those faces and support rows is therefore read as the constant embedding seal route supplied by the handoff theorems. Since none of the cited packets carries quotient stream equality, selected host limits, ambient completeness, metric completeness, countable choice, or a standard bridge row, no such row can replace any face of the pullback. □

Theorem 38.35 (Real constant embedding terminal four-face classifier preservation)**.** *Suppose two accepted presentations of the concrete constant rational embedding agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the constant-window* $\mathsf{StreamName}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier ledger, and the local* $\mathsf{Real}^{\uparrow}$ *seal classifier. Then their public* $\mathsf{Real}^{\uparrow}$ *classifier read is the same terminal four-face classifier route through*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Conversely, any accepted terminal four-face classifier route carrying those support rows determines the classifier read of the constant rational embedding.

This classifier preservation statement exports no quotient stream equality, selected host limit, host real equality, ambient completeness, metric-completeness data, countable choice, or standard bridge row.

Proof. First apply Definition 38.33. It confines the accepted constant rational embedding to the displayed four faces, so the classifier question can only inspect the dyadic endpoint and radius rows, the constant StreamName window, the RegSeqRat ledger, and the local Real seal classifier.

Next use Definition 38.34. Agreement on those four faces and their BHist, hsame, Cont, Pkg, and NameCert support rows gives one terminal route for the public $\mathsf{Real}^{\uparrow}$ seal read, and every accepted terminal packet with the same support rows reads back to that constant embedding route.

Now transport the classifier data along the two handoff boundaries. Apply Definition 301.132 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier ledger that reaches the Real seal, and apply Definition 126.172 to keep the constant observation boundary at the StreamName window.

Finally invoke Definition 38.472. The L10 classifier pullback identifies classifier equality on an accepted four-face route from exactly the displayed classifier rows. Thus the two presentations have the same public $\mathsf{Real}^{\uparrow}$ classifier read, and the reverse terminal route determines the classifier read of the constant rational embedding. The excluded quotient, selected-limit, host-real, completeness, choice, metric-completeness, and standard-bridge rows are absent from every cited route, so they cannot supply an alternative classifier preservation path. □

Theorem 38.36 (Real terminal four-face no direct seal collapse)**.** *If a public* $\mathsf{Real}^{\uparrow}$ *classifier read of the concrete constant rational embedding is accepted, then it factors through the displayed*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

faces with the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Conversely, a packet which omits any of the first three faces is not an accepted public classifier read of the constant embedding.*

Thus the terminal public read excludes quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, standard bridge rows, and any direct $\mathsf{Real}^{\uparrow}$ *seal edge around the dyadic, stream, or regular-sequence faces.*

Proof. First apply Definition 38.33. It gives the route inventory for every accepted constant-embedding public seal read: the dyadic core, stream name, regular sequence, and Real faces are all visible, and their only support rows are the common BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 38.34. The pullback identifies the public seal route uniquely from precisely those faces and rows. Hence a purported public read that bypasses $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, or $\mathsf{RegSeqRat}^{\uparrow}$ has no route coordinate from which the constant embedding seal can be pulled back.

Finally apply Definition 38.35. Classifier preservation transports the same uniqueness statement from the seal route to the public classifier read. Therefore an accepted classifier read factors through all four faces, while a packet missing any of the first three faces is not an accepted constant-embedding classifier read. The excluded quotient stream, selected-limit, host-real, choice, ambient-completeness, metric-completeness, standard-bridge, and direct-seal data are absent from the three cited routes, so none of them supplies a replacement edge to $\mathsf{Real}^{\uparrow}$. □

Theorem 38.37 (Real constant embedding four-face classifier terminality)**.** *For any two accepted concrete constant rational embedding classifier reads, equality of their public* $\mathsf{Real}^{\uparrow}$ *classifier reads holds exactly when their visible terminal rows agree: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the constant-window* $\mathsf{StreamName}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier ledger, the local* $\mathsf{Real}^{\uparrow}$ *seal classifier, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Conversely, any terminal four-face classifier packet carrying precisely those rows determines the public classifier read of the concrete constant rational embedding. This terminality criterion exports no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness row, standard bridge row, or direct $\mathsf{Real}^{\uparrow}$ *seal bypass.*

Proof. For the forward direction, start with the accepted public classifier equality and apply Definition 38.35. Classifier preservation restricts the equality test to the displayed four faces. Then use Definition 38.36: an accepted classifier read cannot pass directly to $\mathsf{Real}^{\uparrow}$ while omitting the dyadic, stream, or regular-sequence faces.

It remains to extract the rows that witness the equality. Apply Definition 38.34. The pullback identifies the public seal route from the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius rows, the constant-window $\mathsf{StreamName}^{\uparrow}$ row, the $\mathsf{RegSeqRat}^{\uparrow}$ ledger, the local $\mathsf{Real}^{\uparrow}$ classifier row, and the common BHist, hsame, Cont, Pkg, and NameCert support. Hence equality of the public classifier read forces agreement on exactly those rows.

For the reverse direction, begin with agreement on the listed terminal rows. The pullback theorem repacks those rows as one terminal four-face packet for the constant embedding seal route. Classifier preservation then transports that repacked packet to the public $\mathsf{Real}^{\uparrow}$ classifier read, so the two accepted presentations have the same classifier endpoint.

The same two theorems also give the converse packet clause: a terminal four-face classifier packet with precisely the displayed rows determines the constant embedding classifier read. The no-direct-seal-collapse theorem excludes quotient stream equality, host real equality, selected limits, countable choice, ambient or metric completeness, standard bridges, and direct Real seal bypasses from serving as an alternative terminal endpoint. □

38.3 Real algebra and order boundary

Definition 38.38 (Concrete RealAlgOrder interface)**.** *The concrete seed for* RealAlgOrder *is the constant rational RealUp subcarrier supplied by Definition 38.31. Its carrier is* RealConstantHistoryCarrier*, its classifier is* RealConstantHistoryClassifier*, and each algebra or order row is a* BHist *relation obtained by applying the corresponding RatUp witness before the e-one RealUp embedding. The rows are:*

1. *addition:* $\mathsf{RatAddRow}(d, e, o)$ *is transported to* $\mathsf{RealAlgAddRow}(\mathsf{E1}(d), \mathsf{E1}(e), \mathsf{E1}(o))$*;*
2. *negation:* $\mathsf{RatNegRow}(d, o)$ *is transported to* $\mathsf{RealAlgNegRow}(\mathsf{E1}(d), \mathsf{E1}(o))$*;*
3. *multiplication:* $\mathsf{RatMulRow}(d, e, o)$ *is transported to* $\mathsf{RealAlgMulRow}(\mathsf{E1}(d), \mathsf{E1}(e), \mathsf{E1}(o))$*;*
4. *absolute value:* $\mathsf{RatAbsRow}(d, o)$ *is transported to* $\mathsf{RealAlgAbsRow}(\mathsf{E1}(d), \mathsf{E1}(o))$*;*
5. *nonnegative comparison:* $\mathsf{RatNonnegativeRow}(d)$ *is transported to* $\mathsf{RealAlgNonnegativeRow}(\mathsf{E1}(d))$*;*
6. *strict apartness:* $\mathsf{RatApartRow}(d, e)$ *is transported to* $\mathsf{RealAlgApartRow}(\mathsf{E1}(d), \mathsf{E1}(e))$.

Thus the interface is a constant-rational boundary object: it names only the operation and order rows whose rational endpoints are already certified by RatUp and whose RealUp endpoints are the constant e-one seals.

Lean4 (definition) `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConcreteInterface`

Theorem 38.39 (RealAlgOrder constant carrier and classifier obligations)**.** *Let* $d, e : \mathsf{Hist}$*. The carrier and classifier face of the constant-rational* RealAlgOrder *boundary consists of the two transported rows*

$$\mathsf{RatHistoryCarrier}(d) \Longrightarrow \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d))$$

and

$$\mathsf{RatHistoryClassifier}(d, e) \Longrightarrow \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)).$$

These rows are stable under the e-one classifier readback of Definition 38.11: every displayed Real constant classifier comparison between e-one endpoints reads back to the RatUp classifier comparison used by the source rows.

Proof. Begin with the embedding certificate. Apply Definition 38.31 to the rational endpoint d; its carrier component is exactly the passage from $\mathsf{RatHistoryCarrier}(d)$ to $\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d))$.

Next read the classifier component of the same certificate at the pair (d, e). The RatUp classifier row is transported first to the constant stream-name comparison and then to the e-one RealUp constant comparison, giving $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e))$.

Finally apply Definition 38.11 in the reverse readback direction. The displayed Real constant classifier comparison is therefore the same rational classifier datum that RatUp classifier transport uses for stability of the source carrier and classifier rows. □

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstantCarrier classifier obligations`

Theorem 38.40 (RealAlgOrder constant algebra row obligations)**.** *Let $d, e, o : \mathsf{Hist}$. The algebra face of the constant-rational* RealAlgOrder *boundary consists of the four transported operation rows*

$$\begin{aligned}
\mathsf{RatAddRow}(d, e, o) &\implies \mathsf{RealAlgAddRow}(\mathsf{E1}(d), \mathsf{E1}(e), \mathsf{E1}(o)),\\
\mathsf{RatNegRow}(d, o) &\implies \mathsf{RealAlgNegRow}(\mathsf{E1}(d), \mathsf{E1}(o)),\\
\mathsf{RatMulRow}(d, e, o) &\implies \mathsf{RealAlgMulRow}(\mathsf{E1}(d), \mathsf{E1}(e), \mathsf{E1}(o)),\\
\mathsf{RatAbsRow}(d, o) &\implies \mathsf{RealAlgAbsRow}(\mathsf{E1}(d), \mathsf{E1}(o)).
\end{aligned}$$

Classifier congruence for these rows is read back through RealConstantHistoryClassifier *to the RatUp classifier transport on the rational endpoints.*

Proof. Use Definition 38.31 to fix the constant-rational bridge for the displayed endpoints. It supplies the e-one RealUp seals for d, e, and o, with carrier and classifier data inherited from their RatUp histories.

For each algebra row, apply the corresponding RatUp witness before embedding. The addition, negation, multiplication, and absolute-value witnesses therefore remain rational ledger rows, while their visible endpoints become the displayed e-one RealUp constants.

It remains to account for classifier congruence. Apply Definition 38.11 to any displayed Real constant classifier comparison among the inputs or output. The result is the RatUp classifier comparison required by RatUp classifier transport, and the embedding certificate repacks the transported witness as the corresponding RealAlgOrder algebra row. □

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant algebra row obligations`

Theorem 38.41 (RealAlgOrder constant order and apartness obligations)**.** *Let $d, e : \mathsf{Hist}$. The order face of the constant-rational* RealAlgOrder *boundary consists of the two transported rows*

$$\mathsf{RatNonnegativeRow}(d) \implies \mathsf{RealAlgNonnegativeRow}(\mathsf{E1}(d))$$

and

$$\mathsf{RatApartRow}(d, e) \implies \mathsf{RealAlgApartRow}(\mathsf{E1}(d), \mathsf{E1}(e)).$$

This boundary records nonnegative and strict-apartness obligations only. It does not assert trichotomy, least-upper-bound completeness, Cauchy completeness, or any total order law beyond the supplied RatUp rows.

Proof. First expose the constant endpoints through Definition 38.31. The same rational-to-Real passage used for the carrier and algebra rows supplies the e-one endpoints on which the nonnegative and apartness rows are displayed.

Next transport the two RatUp order rows. A supplied $\mathsf{RatNonnegativeRow}(d)$ gives the displayed $\mathsf{RealAlgNonnegativeRow}(\mathsf{E1}(d))$, and a supplied $\mathsf{RatApartRow}(d, e)$ gives $\mathsf{RealAlgApartRow}(\mathsf{E1}(d), \mathsf{E1}(e))$.

Finally read classifier movement back through Definition 38.11. The readback produces the RatUp classifier comparisons consumed by RatUp classifier transport, so stability of nonnegative and apartness rows is inherited only on the constant-rational boundary. No trichotomy or completeness datum is present in the transported rows. □

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant order apartness obligations`

Theorem 38.42 (Constant rational RealAlgOrder boundary)**.** *Let $d, e, o : \mathsf{Hist}$. The RatUp carrier, classifier, nonnegative, and apartness rows transported through the constant rational embedding give well-formed* RealAlgOrder *rows on embedded RealUp constants:*

$$\mathsf{RatHistoryCarrier}(d) \Longrightarrow \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d)),$$

$$\mathsf{RatHistoryClassifier}(d, e) \Longrightarrow \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)),$$

and every supplied RatUp row RatAddRow*,* RatNegRow*,* RatMulRow*,* RatAbsRow*,* RatNonnegativeRow*, or* RatApartRow *with output history o transports to the corresponding* RealAlgOrder *row on the embedded constants. These transported rows are stable under* RealConstantHistoryClassifier *on all displayed RealUp constant endpoints.*

Proof. Use Definition 38.31 to expose the carrier and classifier chains from RatUp histories to StreamNameUp constant streams and then to e-one RealUp constants. The carrier row gives $\mathsf{RealConstantHistoryCarrier}(\mathsf{E1}(d))$ from $\mathsf{RatHistoryCarrier}(d)$, and the classifier row gives RealConstantHistoryClassifie from $\mathsf{RatHistoryClassifier}(d, e)$.

For each algebra or order row, apply the corresponding RatUp witness before embedding. Addition, negation, multiplication, absolute value, nonnegative comparison, and strict apartness therefore keep their rational ledger as the hidden source and receive e-one RealUp endpoints as their public boundary.

It remains to check stability under the RealUp constant classifier. Read a classifier comparison back to the RatUp classifier at the e-one endpoint by Definition 38.11. RatUp classifier transport moves the chosen algebra or order witness along the exposed rational comparison, and the constant rational embedding certificate repacks the result as the corresponding RealUp constant row. □

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant rational order boundary`

The intended Lean target shape for this boundary is `BEDC.Derived.RealAlgOrderUp.constant_rational_orc`

Theorem 38.43 (RealAlgOrder constant rational scoped certificate)**.** *The constant-rational* RealAlgOrder *package is scoped exactly by* RatHistoryCarrier*,* RatHistoryClassifier*, the RatUp operation and order rows,* RealConstantHistoryCarrier*,* RealConstantHistoryClassifier*,* BHist*,* hsame*, Definition 38.31, and Definition 38.11. More precisely, it forms a ten-field scoped certificate with the following fields:*

1. *traditional target: the constant rational real algebra and order surface of* RealAlgOrder*;*
2. *BEDC source:* $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow}$ *restricted to RealUp constant rational endpoints over* BHist *and* hsame*;*
3. *carrier:* RealConstantHistoryCarrier*;*

4. *classifier:* RealConstantHistoryClassifier*;*
5. *stability: every displayed Real constant classifier comparison reads back to the RatUp classifier row by Definition 38.11;*
6. *ledger: the hidden rational operation or order row remains the RatUp row, while the visible endpoint is sealed by the e-one RealUp constant embedding of Definition 38.31;*
7. *carrier and classifier theorem: Definition 38.39;*
8. *algebra theorem: Definition 38.40;*
9. *order and apartness theorems: Definition 38.41 and Definition 38.42;*
10. *exactness: no completeness row, trichotomy row, metric row, topology row, or standard-bridge row is consumed.*

Proof. First project the four RealAlgOrder boundary theorems. The carrier and classifier component comes from Definition 38.39; the algebra component comes from Definition 38.40; the order and apartness component comes from Definition 38.41; and the combined boundary comes from Definition 38.42.

Next expose the permitted source for each projected component. The only public Real endpoints are $\mathsf{E1}(d)$-sealed rational histories supplied by Definition 38.31. The only hidden operation and order rows are the RatUp rows named in the concrete interface, and the only ambient history relation used to move them is hsame on BHist.

Finally read every Real constant classifier move back to RatUp through Definition 38.11. This gives exactly the RatUp classifier comparison required by the projected carrier, classifier, algebra, order, and apartness rows, and no proof step asks for completeness, trichotomy, metric structure, topology, or a standard bridge. □

↪ variant BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant rational scoped certificate

Theorem 38.44 (RealAlgOrder constant rational audit exactness)**.** *For constant-rational endpoints* $\mathsf{E1}(d)$*,* $\mathsf{E1}(e)$*, and* $\mathsf{E1}(o)$*, every accepted* RealAlgOrder *carrier, classifier, algebra, nonnegative, and apartness read factors through the constant rational route:*

RatHistoryCarrier, RatHistoryClassifier, *the displayed RatUp operation and order rows,*
RealConstantHistoryCarrier, RealConstantHistoryClassifier, BHist, hsame.

The only cross-layer certificates used by the route are Definition 38.31 and Definition 38.11. Thus the constant-rational interface does not project full non-constant real arithmetic, trichotomy, least-upper-bound completeness, Cauchy completeness, metric or topology rows, quotient equality, or a standard bridge.

Proof. Start with the carrier inventory supplied by Definition 38.39. Its public RealUp endpoints are exactly the e-one constants $\mathsf{E1}(d)$, $\mathsf{E1}(e)$, and $\mathsf{E1}(o)$ whose hidden witnesses are RatUp histories. The carrier and classifier rows therefore pass through RatHistoryCarrier, RatHistoryClassifier, RealConstantHistoryCarrier, and RealConstantHistoryClassifier.

Next apply the row inventories. The algebra rows are those of Definition 38.40; the nonnegative and apartness rows are those of Definition 38.41; and their constant-rational boundary is Definition 38.42. Each row is read by taking the RatUp operation or order witness first and then embedding its output as a RealUp constant endpoint.

Finally compose the scoped certificate of Definition 38.43 with the two RealUp constant readback certificates. The embedding certificate supplies the forward route from rational histories to e-one RealUp constants, and the classifier readback certificate returns any displayed Real constant classifier move to the RatUp classifier. Since this inventory contains no row for non-constant real operations, trichotomy, completeness, metric or topological structure, quotient equality, or a standard bridge, no accepted read can escape the constant rational route. □

The intended Lean target shape for this certificate is `BEDC.Derived.RealAlgOrderUp.constant_rational_sc`

Theorem 38.45 (RealAlgOrder positive-radius window stability)**.** *Let A and B be accepted* RealAlgOrder *constant packets whose endpoints are sealed by the same* StreamName$^{\uparrow}$ *finite-window source, the same* RegSeqRat *seal, the same* Real$^{\uparrow}$ *component row, and the same positive* Rat$^{\uparrow}$ *order-radius row after the displayed* hsame *and* Cont *transports. Then their apartness and order read is the same finite L10 window read. The read is determined by the StreamName finite observation window, the RegSeqRat regularity seal, the RealUp constant endpoint, and the RatUp positive-radius order witness; it cannot import quotient stream equality, host real equality, trichotomy, selected limits, ambient completeness, metric structure, or a standard bridge.*

Proof. Begin with the constant rational order boundary, Definition 38.42. It routes every positive-radius apartness or order read through the RatUp witness before the result is embedded as a RealUp constant endpoint.

Next apply the scoped certificate of Definition 38.43. Since the two packets agree on the StreamName finite-window source, the RegSeqRat seal, the RealUp component row, and the RatUp radius row through hsame and Cont, the accepted read has no remaining coordinate from which to choose a different order window.

Finally use the StreamName and RegSeqRat finite-window handoffs cited by the RealUp seal, including the constant rational embedding and classifier readback certificates. These handoffs identify the L10 observation window as the public source of the RealAlgOrder read and leave no route to quotient equality, host real equality, trichotomy, selected limits, completeness, metric data, or a standard bridge. □

Theorem 38.46 (RealAlgOrder constant apartness metric handoff)**.** *For constant-rational RealUp endpoints carried by the* RealAlgOrder *constant carrier, the checked carrier and classifier rows, algebra rows, and order and apartness rows package a finite* BHist *handoff for downstream apartness and metric consumers. The handoff consists exactly of the two displayed RealUp endpoint rows* E1(d) *and* E1(e)*, their RatUp witness rows, the supplied positive dyadic or apartness row, the* hsame *and* Cont *transport routes, the visible* Pkg *provenance, and the* NameCert *ledger that records the constant rational source.*

Any ApartnessReal$^{\uparrow}$ *or* Met$^{\uparrow}$ *consumer may read this packet only as finite positive-separation data in the sense of Definition 304.6, Definition 304.5, and Definition 304.8. The packet does not export trichotomy, completeness, host real equality, quotient data, or a classifier outside the displayed constant-rational boundary.*

Proof. First project the RealAlgOrder constant boundary rows. The carrier and classifier rows come from Definition 38.39; the operation rows come from Definition 38.40; and the order and apartness rows come from Definition 38.41.

Next apply Definition 38.43. It packages those rows over BHist with the exposed hsame transport, the RatUp source witnesses, the e-one RealUp endpoint seals, and the visible ledger provenance. Thus a consumer sees the two endpoint rows and the finite positive dyadic or apartness row as one bounded packet, rather than as a global real-order principle.

Finally match that packet with the downstream apartness surface. The finite row of Definition 304.6, the handoff of Definition 304.5, and the consumer boundary of Definition 304.8 read exactly the endpoint, radius or apartness, transport, Pkg, and NameCert coordinates named above. Since no cited surface contains trichotomy, completion, host equality, or quotient coordinates, the handoff cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealUp.RealAlgOrderConstantApartnessMetricHandoff`

Theorem 38.47 (RealAlgOrder apartness positive-gap order consumer)**.** *For constant-rational RealUp endpoints carried by the* RealAlgOrder *boundary, a supplied RatUp apartness or nonnegative positive-gap row together with the e-one embedding determines exactly the finite* ApartnessReal$^{\uparrow}$ *separation packet consumed by later* Met$^{\uparrow}$ *and order consumers. The packet consists of the two endpoint rows, the positive dyadic radius or apartness row, the finite window and readback rows, the visible* hsame *and* Cont *transports, the* Pkg *provenance, and the* NameCert *dependency rows.*

The consumer surface exports no trichotomy, completeness, quotient equality, host real inequality, topology row, or non-constant RealUp arithmetic. Every downstream read is restricted to the finite separation packet of Definition 304.6 and the consumer boundary of Definition 304.8.

Proof. Project the constant-rational order and apartness rows from Definition 38.41. The supplied RatUp positive-gap row is then carried upward only through the e-one endpoint embedding, so the visible RealUp endpoints remain the constant-rational rows certified by Definition 38.43.

Next repackage those projected rows through Definition 38.46. Its handoff surface already isolates the two endpoint rows, the positive dyadic radius or apartness row, the hsame and Cont routes, the Pkg provenance, and the NameCert ledger that records the constant rational source.

It remains to identify the packet consumed downstream. The finite row of Definition 304.6 supplies the endpoint, window, readback, and separation-ledger shape. Applying Definition 304.8 then restricts every metric or order consumer to exactly those coordinates, so no excluded order, quotient, host-inequality, completeness, or topology coordinate is available to read. □

Theorem 38.48 (RealAlgOrder constant positive-gap consumer determinacy)**.** *Let two constant-rational* RealAlgOrder *endpoint packets carry the same RatUp source endpoints, the same positive dyadic or apartness row, the same e-one* Real$^{\uparrow}$ *embedding rows, the same finite-window and readback rows, the same* hsame *and* Cont *transport routes, the same* Pkg *provenance, and the same* NameCert *ledger. Then every* ApartnessReal$^{\uparrow}$, Met$^{\uparrow}$, *or order consumer reading those packets reads one and the same finite positive-gap separation packet.*

The determined packet contains exactly the two constant-rational endpoint rows, the shared positive-gap row, the finite-window and readback rows, the visible transports, provenance, and ledger. It exports no trichotomy, completeness, quotient equality, host real inequality, topology row, or non-constant Real$^{\uparrow}$ *arithmetic.*

Proof. Project the RatUp order and apartness rows for the two endpoint packets through Definition 38.41. The hypotheses identify the RatUp endpoints and the positive dyadic or apartness row before any downstream consumer is allowed to read them.

Package the projected rows by Definition 38.43. This places both endpoint displays over the same e-one Real$^{\uparrow}$ embedding rows, the same hsame and Cont routes, the same Pkg provenance, and the same NameCert ledger.

Route the packaged data through Definition 38.46. Its handoff surface has precisely the finite positive-separation coordinates listed in the statement, so the two displays present one packet to apartness and metric consumers.

Finally apply Definition 38.47. That theorem identifies the downstream $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and order consumer surface with the same finite separation packet. Since each cited surface omits trichotomy, completeness, quotient equality, host real inequality, topology, and non-constant real arithmetic, none of those rows is exported by the determinacy claim. □

Theorem 38.49 (RealAlgOrder constant public interface export)**.** *The carrier and classifier obligations of Definition 38.39, the algebra row obligations of Definition 38.40, the order and apartness obligations of Definition 38.41, the rational boundary of Definition 38.42, the scoped certificate of Definition 38.43, and the metric handoff of Definition 38.46 assemble the reusable constant-rational public* RealAlgOrder *interface.*

Every consumer read of this interface factors through RatUp, StreamNameUp, RealUp constant endpoints, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The interface exports no trichotomy, completeness, host real equality, quotient data, non-constant RealUp arithmetic, topology row, or standard bridge.*

Proof. Project the checked carrier and classifier rows, the algebra rows, and the order and apartness rows from the three obligation theorems named above. The rational order boundary identifies these rows as constant-rational rows transported from RatUp through the e-one RealUp endpoint embedding.

Package the projected rows with Definition 38.43. This makes the public surface a scoped packet over BHist, visible hsame and Cont transport, Pkg provenance, and the NameCert ledger.

Finally route apartness and metric consumers through Definition 38.46. Since the handoff reads only the displayed constant endpoints and finite separation rows, none of the excluded full-real, quotient, topology, completeness, or bridge data is available to export. □

Theorem 38.50 (RealAlgOrder constant public order-metric interface)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the* RealAlgOrder *package, every public order read, apartness read, and* $\mathsf{Met}^{\uparrow}$*-facing read factors through the RatUp denominator row, the* E1 $\mathsf{Real}^{\uparrow}$ *endpoint, the transported algebra, order, and apartness rows, the positive-gap separation packet,* hsame*,* Cont*, and* Pkg *provenance.*

The public boundary exports no nonconstant real arithmetic, trichotomy, completeness, topology, quotient carrier, host-real inequality, or standard bridge row. Thus order consumers and metric consumers receive the same finite BHist *constant-rational interface.*

Proof. Begin with Definition 38.49. It packages the carrier, classifier, algebra, order, apartness, rational boundary, and metric handoff rows into one public constant-rational RealAlgOrder surface over RatUp denominators, E1 endpoints, hsame, Cont, Pkg provenance, and the NameCert ledger.

For the metric-facing read, apply Definition 38.46. The handoff identifies the displayed $\mathsf{Met}^{\uparrow}$ surface with the same constant endpoint rows and the same finite positive-separation data.

For the order-facing read, apply Definition 38.47. This turns the positive-gap separation packet into the order-consumer row without adding a new full-real comparison principle.

The determinacy and scope controls come from Definition 38.48 and Definition 38.43. They keep the apartness, metric, and order surfaces tied to the transported rational witness, the finite BHist ledger, and the displayed provenance. Since the cited rows omit nonconstant arithmetic, trichotomy, completeness, topology, quotient carriers, host-real inequalities, and standard bridges, the combined public boundary omits them as well. □

Theorem 38.51 (RealAlgOrder positive apartness nonzero metric boundary)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the* RealAlgOrder *package, suppose the RatUp positive-gap or*

apartness row used by Definition 38.47 is present. Then every $\mathsf{Met}^{\uparrow}$*-facing handoff packet supplied by Definition 38.46 carries a nonempty positive-distance boundary. It cannot be read as an empty-distance witness for the same displayed endpoints.*

The exported packet is restricted to finite BHist *endpoint rows, the radius or apartness row,* hsame *transport,* Cont *routes,* Pkg *provenance, and* NameCert *dependency rows. It exports no trichotomy, full real completeness, quotient carrier, host-real equality, or non-constant* $\mathsf{Real}^{\uparrow}$ *arithmetic.*

Proof. Project the order and apartness rows from Definition 38.41. The supplied RatUp positive-gap row is the same row consumed by Definition 38.47, so the constant endpoints remain tied to their rational witnesses and visible E1 embeddings.

Pass those projected rows through Definition 38.46. The handoff packet is therefore the finite positive-separation packet described by Definition 304.6 and Definition 304.8, with the positive radius or apartness ledger still visible.

Now test the contrary empty-distance read on the metric-facing side. The MetricUp endpoint boundary Definition 57.213 requires a nonempty endpoint boundary from a nonempty distance, while Definition 57.222 rules out reading the same visible positive endpoint evidence as an empty-distance witness. Thus the positive apartness packet can only enter the metric consumer as a nonempty positive-distance boundary.

Finally apply Definition 38.50. It limits the exported rows to finite endpoint histories, radius or apartness evidence, visible hsame and Cont transport, Pkg provenance, and NameCert rows. No excluded full-real or quotient coordinate is available to turn the positive boundary into an empty-distance packet. □

Theorem 38.52 (RealAlgOrder constant positive boundary metric consumer)**.** *A constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoint packet carrying a positive apartness row or a positive-gap row exports exactly the* $\mathsf{Met}^{\uparrow}$ *nonempty-distance consumer boundary. The consumer input consists of finite* BHist *endpoint histories, the positive radius or apartness ledger, visible* hsame *and* Cont *routes,* Pkg *provenance, and the local* NameCert *rows for the constant endpoint packet.*

Thus the exported metric consumer surface contains no empty-distance witness, quotient real equality, trichotomy, selected limit, full real arithmetic row, or ambient completeness principle.

Proof. Project the metric-facing handoff through Definition 38.46. It identifies the $\mathsf{Met}^{\uparrow}$ consumer side with the same constant endpoints and the same positive separation data carried by the $\mathsf{RealAlgOrder}^{\uparrow}$ packet.

The positive packet is the order-facing packet isolated by Definition 38.47. Its determinacy is controlled by Definition 38.48, so the positive gap, apartness row, rational witness, and displayed endpoint histories cannot drift apart.

Now apply the endpoint boundary rows for $\mathsf{Met}^{\uparrow}$. The nonempty-distance boundary of Definition 57.213 accepts the displayed positive endpoint evidence, while Definition 57.222 excludes reading that same evidence as an empty-distance endpoint witness.

Finally use Definition 38.50. It exhausts the public rows by finite endpoint histories, the positive ledger, visible hsame and Cont routes, Pkg provenance, and NameCert rows. Since no quotient, trichotomy, selected-limit, completeness, or full-arithmetic coordinate is exported, the consumer boundary is exactly the nonempty-distance input just described. □

Theorem 38.53 (RealAlgOrder constant rational standard bridge boundary)**.** *The constant-rational public* RealAlgOrder *interface of Definition 38.49, the order-metric public interface of*

Definition 38.50, and the positive-boundary metric consumer of Definition 38.52 assemble a conservative standard bridge from RatUp rational witnesses through RealUp constant endpoints to the $\mathsf{Met}^{\uparrow}$ *nonempty-distance consumer surface.*

The bridge rows are exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, RatUp denominator, RealUp constant endpoint,* RealAlgOrder *order and apartness, and* $\mathsf{Met}^{\uparrow}$ *distance-boundary rows. The packet contains no trichotomy, completed-real equality, full nonconstant arithmetic, topology, quotient carrier, selected limit, completeness, or arbitrary host metric-space coordinate.*

Proof. Project the public constant-rational interface by Definition 38.49. This supplies the RatUp denominator witnesses, their E1 RealUp constant endpoints, the finite BHist endpoint histories, visible hsame and Cont transport, Pkg provenance, and the NameCert rows.

Route the order and metric reads through Definition 38.50. The order, apartness, and metric-facing consumers therefore receive the same displayed constant-rational rows rather than separate carrier, quotient, topology, or completion data.

Identify the metric side with Definition 38.52. It pins the $\mathsf{Met}^{\uparrow}$ consumer surface to the nonempty-distance boundary carried by the positive-gap or apartness ledger over the same endpoints.

Conservativity follows because every row just cited is one of the finite displayed rows named in the statement. None of trichotomy, completed-real equality, full nonconstant arithmetic, topology, quotient carriers, selected limits, completeness, or arbitrary host metric spaces is a coordinate of the packet, so the bridge exports only the displayed RatUp–RealUp–MetricUp boundary. □

Theorem 38.54 (RealAlgOrder standard bridge row conservativity)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the* RealAlgOrder *package, every standard bridge read factors through the displayed* $\mathsf{Rat}^{\uparrow}$ *denominator witness,* $\mathsf{Real}^{\uparrow}$ *embedding row, algebra row, order and apartness row, metric handoff row, finite* BHist *endpoint histories, visible* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Consequently the bridge exports no full non-constant real arithmetic, trichotomy, completeness principle, topology, quotient carrier, or host-real inequality.

Proof. Begin with the scoped constant-rational certificate of Definition 38.43. It fixes the only admissible endpoint histories as E1-embedded rational witnesses and keeps the classifier transport, BHist rows, hsame rows, Cont routes, Pkg provenance, and NameCert ledger visible.

Project the public order and metric rows through Definition 38.50. The algebra, order, apartness, and metric-facing coordinates are therefore read from the same displayed rational witness and its constant $\mathsf{Real}^{\uparrow}$ embedding, not from an ambient real carrier.

For positive metric consumers, apply Definition 38.52. The handoff row is the visible nonempty-distance boundary obtained from the positive gap or apartness ledger over those same endpoints.

Finally use Definition 38.53. Its bridge ledger is exhausted by exactly the rows listed above. Since no row names full non-constant arithmetic, trichotomy, completeness, topology, quotient carriers, or host-real inequalities, every standard bridge read is conservative over the transported $\mathsf{Rat}^{\uparrow}$ rows. □

Theorem 38.55 (RealAlgOrder constant rational bridge endpoint determinacy)**.** *Let P and Q be two constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *standard-bridge packets whose displayed* $\mathsf{Rat}^{\uparrow}$ *source endpoints are the same. Suppose the two packets carry the same transported algebra, order, and apartness rows, the same positive-gap or apartness ledger, and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Then the $\mathsf{Met}^{\uparrow}$ *nonempty-distance consumer endpoints exposed by* P *and* Q *are the same boundary packet. The common exposed rows are exactly the* $\mathsf{Rat}^{\uparrow}$ *denominator witnesses, the* $\mathsf{Real}^{\uparrow}$ *constant endpoints, the* RealAlgOrder *order and apartness rows, and the* $\mathsf{Met}^{\uparrow}$ *distance-boundary rows. No trichotomy, completed-real equality, topology, quotient carrier, selected limit, completeness, or arbitrary host metric-space coordinate is exported.*

Proof. First project both displays through Definition 38.49. Since their $\mathsf{Rat}^{\uparrow}$ source endpoints and displayed histories agree, the public constant-rational packet supplies the same denominator witnesses, the same $\mathsf{Real}^{\uparrow}$ constant endpoints, and the same local provenance rows.

Next read the shared algebra, order, apartness, and metric-facing rows through Definition 38.50. This keeps the two displays on the same transported rational source rather than adding an ambient real carrier or host comparison.

The positive boundary is then identified by Definition 38.52. The common positive-gap or apartness ledger exposes one $\mathsf{Met}^{\uparrow}$ nonempty-distance consumer boundary over the displayed endpoints.

Finally apply Definition 38.53. Its standard bridge coordinates are exactly the denominator, constant endpoint, order, apartness, and metric distance-boundary rows already fixed above. Therefore the two displays expose the same $\mathsf{Met}^{\uparrow}$ consumer endpoint boundary, with no additional real-order, topological, quotient, limit, completeness, or host metric-space coordinate available for them to differ on. □

Theorem 38.56 (RealAlgOrder constant transported classifier congruence)**.** *Let* P *be a constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoint packet whose displayed* $\mathsf{Rat}^{\uparrow}$ *source classifier rows are* $\mathsf{RatHistoryClassifier}(d, d')$, $\mathsf{RatHistoryClassifier}(e, e')$, *and* $\mathsf{RatHistoryClassifier}(o, o')$. *Then every carrier row, algebra row, order row, apartness row, and* $\mathsf{Met}^{\uparrow}$*-facing consumer row exported by the public constant-rational interface is congruent along exactly those three* $\mathsf{Rat}^{\uparrow}$ *classifier rows after applying the* $\mathsf{E1}$ $\mathsf{Real}^{\uparrow}$ *embedding.*

Equivalently, the exported rows of the packet have one classifier witness surface: the carrier endpoints are controlled by the displayed rational classifiers, the algebra outputs are the transported rational algebra outputs, the order and apartness observations are the transported rational order and apartness observations, and every metric consumer reads the resulting constant-rational boundary through the same embedded rows. No trichotomy, completeness theorem, topology, quotient carrier, or nonconstant arithmetic coordinate is introduced.

Proof. First apply Definition 38.39. The constant-rational carrier rows and classifier rows are already displayed as $\mathsf{E1}$-embedded $\mathsf{Rat}^{\uparrow}$ histories, so the only admissible classifier witnesses are the rational rows named in the statement.

Next use Definition 38.40. The addition and multiplication rows are computed on the rational witnesses before returning to the $\mathsf{Real}^{\uparrow}$ carrier, so their exported congruence is transported from the same rational classifier surface.

For the comparison coordinates, apply Definition 38.41. This identifies the order and apartness rows with the displayed rational order and apartness rows after the $\mathsf{E1}$ embedding.

Finally project through Definition 38.50 and Definition 38.55. The public order, apartness, and metric-facing consumer rows therefore share the same transported classifier witnesses and cannot acquire a trichotomy, completeness, topological, quotient, or nonconstant arithmetic coordinate. □

Theorem 38.57 (RealAlgOrder public apartness metric export)**.** *For constant rational real names, the public* $\mathsf{RealAlgOrder}^{\uparrow}$ *export consists exactly of the carrier and classifier rows, algebra rows,*

order and apartness rows, positive-gap metric handoff rows, and the displayed hsame, Cont, Pkg, *and* NameCert *provenance already listed in this chapter. A* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{S}^{1\uparrow}$ *consumer may read only those rows.*

The export supplies no trichotomy row, host real equality, quotient representative, or completeness datum.

Proof. Apply Definition 38.49 to obtain the carrier, classifier, algebra, provenance, and public NameCert rows.

Next apply Definition 38.50 and Definition 38.51. These project the order, apartness, and positive-gap metric handoff rows available to metric and circle consumers.

Finally use Definition 38.56. The transported rational classifier surface is the only congruence source for the displayed rows, so row projection leaves no coordinate from which trichotomy, host real equality, quotient representatives, or completeness could be read. □

Theorem 38.58 (RealAlgOrder public certificate audit boundary)**.** *The public* $\mathsf{RealAlgOrder}^{\uparrow}$ *certificate boundary is exhausted by the constant-rational carrier and classifier rows, the transported* $\mathsf{Rat}^{\uparrow}$ *algebra, order, and apartness rows, the* $\mathsf{Real}^{\uparrow}$ *e-one endpoint embedding, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *classifier readback rows, and the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{S}^{1\uparrow}$ *consumer handoff rows displayed in this chapter. Every public consumer read factors through those rows and their visible* BHist, hsame, Cont, Pkg, *and* NameCert *provenance.*

The boundary exports no trichotomy, completeness, host equality, quotient representative, nonconstant arithmetic coordinate, topology row, or standard bridge.

Proof. Begin with Definition 38.43. It packages the constant-rational carrier, classifier, algebra, order, and apartness rows over the displayed $\mathsf{Rat}^{\uparrow}$ witnesses, $\mathsf{Real}^{\uparrow}$ e-one endpoints, BHist, hsame, Cont, Pkg, and NameCert ledger.

Next apply Definition 38.57. Its public projection identifies the apartness and metric-facing surface with the same finite row set, so a $\mathsf{Met}^{\uparrow}$ or $\mathsf{S}^{1\uparrow}$ consumer receives only the exported positive-gap handoff packet.

The two following handoff surfaces keep the downstream reads inside that packet: Definition 38.59 isolates the metric separation consumer, and Definition 38.66 isolates the $\mathsf{S}^{1\uparrow}$ positive boundary consumer.

Thus row projection exhausts the public certificate boundary by the rows named in the statement. Since the scoped certificate, public export, and two consumer handoff surfaces each omit trichotomy, completeness, host equality, quotient representatives, nonconstant arithmetic coordinates, topology rows, and standard bridges, the assembled public boundary omits them as well. □

Theorem 38.59 (RealAlgOrder apartness metric consumer separation)**.** *For constant rational real names, any* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{S}^{1\uparrow}$ *consumer of the public apartness and metric export of Definition 38.57 can read only the rationally transported carrier, classifier, algebra, order, apartness, positive-gap handoff,* hsame, Cont, Pkg, *and* NameCert *rows displayed by the* $\mathsf{RealAlgOrder}^{\uparrow}$ *packet.*

The same consumer surface does not supply a trichotomy row, completed-real equality, quotient representative, or completeness datum. Apartness and the positive-gap metric boundary are therefore consumer-visible separation rows, not an ambient completed-real order or quotient-equality principle.

Proof. Start with Definition 38.43. The scoped certificate fixes the constant-rational carrier by a $\mathsf{Rat}^{\uparrow}$ witness embedded into $\mathsf{Real}^{\uparrow}$, and it lists the classifier, algebra, order, apartness, transport, continuation, provenance, and local name rows as the available coordinates.

Apply Definition 38.57. Its public projection is exactly the row set available to $\mathsf{Met}^{\uparrow}$ and $\mathsf{S}^{1\uparrow}$ consumers, so a consumer can only project the displayed rows already present in the packet.

Next use Definition 38.51. The apartness-to-positive-gap handoff is a boundary row over the same transported rational witnesses. It adds a metric-facing separation read but no independent completed-real equality, quotient representative, or completeness coordinate.

Finally compare this projected row set with the constructive real algebra boundary recorded by the roadmap constraints for section 9467.12. Since no displayed ledger coordinate has the type of a trichotomy row, completed-real equality, quotient representative, or completeness datum, those data cannot be obtained by reading the apartness or metric rows. □

Theorem 38.60 (RealAlgOrder MetricSpace SOne public interface triad)**.** *For constant rational real names, let a* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness packet be consumed both by the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier and by an accepted* $\mathsf{S}^{1\uparrow}$ *component-and-unit-equation packet. The joint consumer interface factors through exactly the shared* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoint witnesses, the* $\mathsf{Met}^{\uparrow}$ *distance row, the* $\mathsf{S}^{1\uparrow}$ *component and unit-equation rows, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *rows. The triad interface supplies no trichotomy row, completed-real equality, angle quotient, host metric structure, or Real completeness datum.*

Proof. First apply Definition 38.61. The positive-distance handoff repacks the RealAlgOrder apartness row as the $\mathsf{Met}^{\uparrow}$ real-distance component over the same rational endpoint witnesses, real-name transports, hsame rows, continuation routes, provenance, and local certificate.

Next use Definition 38.62. This keeps metric-facing consumption inside the displayed endpoint, apartness, transport, continuation, provenance, local naming, and metric ledger rows. The metric consumer therefore contributes only the public distance and triangle boundary rows, not an ambient host metric or completed-real equality.

Finally apply Definition 38.63. The $\mathsf{S}^{1\uparrow}$ side reads the same RealAlgOrder endpoint witnesses through the real component rows, product carrier, unit-equation witness, componentwise classifier rows, and equation transports. Since the SOne boundary excludes angle quotients and the RealAlgOrder metric boundary excludes trichotomy, completed-real equality, host metric structure, and Real completeness, the combined consumer surface is exactly the displayed triad of RealAlgOrder endpoints, MetricSpace distance, and SOne component-and-equation rows. □

Theorem 38.61 (RealAlgOrder public positive-distance handoff)**.** *For constant rational real names, the positive-gap or apartness row exported by Definition 38.57 repacks as the positive component of the* $\mathsf{Met}^{\uparrow}$ *real-distance public surface of Definition 57.311. The repacking reads only the* $\mathsf{Rat}^{\uparrow}$ *endpoint witnesses, their* $\mathsf{Real}^{\uparrow}$ *transported names, the displayed* hsame *rows,* Cont *routes,* Pkg *provenance, local* NameCert *row, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *budget packet. It does not add a completed-real equality row, a trichotomy row, or a host metric structure.*

Proof. Start from the exported apartness and positive-gap row of Definition 38.57. The row already records the two rational endpoints, their real-name transport, the visible hsame coordinates, the continuation routes, provenance, and local certificate.

Use Definition 38.51 to read this row as a metric-facing positive boundary. The theorem supplies the positive component over the same transported endpoints and the same budget packet, so no new equality or trichotomy coordinate is introduced.

Finally compare the resulting row with Definition 57.311 and Definition 57.313. The $\mathsf{Met}^{\uparrow}$ public surface consumes the positive-distance component as a displayed ledger row, and Definition 38.59 keeps the consumer from reading any stronger completed-real or host-metric structure. □

Theorem 38.62 (RealAlgOrder apartness triangle consumer)**.** *Suppose two adjacent* $\mathsf{Met}^{\uparrow}$ *positive-distance reads are both obtained from* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness packets of Definition 38.57. Then the outer triangle-boundary consumer factors through the same visible endpoint rows, apartness rows,* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and metric triangle ledger rows. The factorization supplies only the public triangle consumer boundary of* $\mathsf{Met}^{\uparrow}$ *and does not upgrade RealAlgOrder apartness into completed-real equality, trichotomy, or an ambient host metric.*

Proof. Apply the public export theorem to each adjacent apartness packet. Each packet exposes a positive-distance read over its displayed rational endpoints, real transport, apartness row, hsame coordinate, continuation route, provenance, and local certificate.

Next apply Definition 38.51 to both reads. The shared endpoint remains visible because both positive components are repacked from the same RealAlgOrder endpoint ledger rather than from a hidden host metric.

Use Definition 57.311 and Definition 57.313 to feed the two displayed positive components into the metric triangle ledger. The triangle consumer reads the outer boundary through the same public metric rows.

Finally invoke Definition 38.59. The consumer surface is limited to visible endpoints, apartness, transport, continuation, provenance, local naming, and triangle-ledger rows, so the outer boundary does not create completed-real equality, trichotomy, or host metric structure. □

Theorem 38.63 (RealAlgOrder SOne unit-equation apartness boundary)**.** *For constant rational real names feeding an accepted* $\mathsf{S}^{1\uparrow}$ *component-and-unit-equation packet, every unit-equation or separation consumer that uses the positive-gap or apartness row exported by Definition 38.57 factors through the same* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoint witnesses, the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311, and the* $\mathsf{S}^{1\uparrow}$ *component and unit-equation rows of chapter 58. The boundary exports no angle quotient, trichotomy row, completed-real equality, host metric structure, or Real completeness datum.*

Proof. Apply Definition 38.57 to the two constant rational endpoint names. The exported row supplies the same endpoint witnesses, real-name transports, apartness surface, continuation routes, provenance, and local naming data used by the RealAlgOrder packet.

Repack the positive-gap row through Definition 38.51. This keeps the metric-facing positive component tied to the displayed RealAlgOrder endpoint witnesses rather than to a hidden host metric.

Use Definition 57.311 and Definition 38.61 to feed that positive component into the public real-distance carrier. The $\mathsf{S}^{1\uparrow}$ consumer then reads only the two real component rows, the product carrier, the unit-equation witness, componentwise classifier rows, and equation transport recorded in chapter 58.

Finally invoke Definition 38.59. The consumer-separation result excludes completed-real equality, trichotomy, quotient representatives, and completeness data from the RealAlgOrder metric surface; the $\mathsf{S}^{1\uparrow}$ carrier rows exclude angle quotients and host metric structure. Therefore the unit-equation consumer factors through exactly the displayed RealAlgOrder, MetricSpace, and SOne rows. □

Theorem 38.64 (RealAlgOrder MetricSpace SOne positive-boundary transport)**.** *For constant rational real names, componentwise* hsame *transport of the* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoint witnesses preserves the positive-boundary row through the three consumer interfaces displayed by the* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *packets. The transported row is the same visible* BHist *boundary with the same* Cont

routes, Pkg *provenance, and local* NameCert *rows: it reads as the* $\mathsf{RealAlgOrder}^{\uparrow}$ *positive apartness boundary, as the* $\mathsf{Met}^{\uparrow}$ *positive-distance row, and as the* $\mathsf{S}^{1\uparrow}$ *component-and-unit-equation row over the same transported endpoints.*

The transported consumer still has no trichotomy row, completed-real equality, angle quotient, host metric structure, or Real completeness coordinate.

Proof. First apply Definition 38.61. The positive-gap row exported by the RealAlgOrder packet is repacked as the $\mathsf{Met}^{\uparrow}$ positive-distance component without changing the rational endpoint witnesses, their hsame transports, the displayed BHist boundary, the Cont routes, the Pkg provenance, or the local NameCert row.

Next apply Definition 38.63. The same transported positive-boundary row feeds the $\mathsf{S}^{1\uparrow}$ component and unit-equation consumer through the displayed real components, product carrier, componentwise classifier rows, and equation transports over those endpoints.

Finally use Definition 38.59. That separation theorem says that the shared consumer surface contains only the listed endpoint, apartness, metric, component, transport, continuation, provenance, and naming rows, so no trichotomy, completed-real equality, angle quotient, host metric, or completeness coordinate is transported with the positive boundary. □

Theorem 38.65 (RealAlgOrder MetricSpace SOne positive-boundary correspondence)**.** *For constant rational real names, the positive-boundary row exported by* $\mathsf{RealAlgOrder}^{\uparrow}$*, the positive-distance row consumed by* $\mathsf{Met}^{\uparrow}$*, and the component-and-unit-equation separation row consumed by* $\mathsf{S}^{1\uparrow}$ *are the same displayed* BEDC *boundary read through three public interfaces. The shared rows are precisely the* $\mathsf{Rat}^{\uparrow}$ *endpoint witnesses, the transported* $\mathsf{Real}^{\uparrow}$ *names, the componentwise* hsame *transports, the* Cont *routes, the* Pkg *provenance, the local* NameCert *row, the* $\mathsf{Met}^{\uparrow}$ *distance row, and the* $\mathsf{S}^{1\uparrow}$ *component and unit-equation rows.*

This correspondence is a statement about public readings of the displayed BEDC *boundary. It is not a host metric equality, not a completed-real equality, and not an import of trichotomy, an angle quotient, host metric structure, or Real completeness.*

Proof. First apply Definition 38.60. This identifies the three public consumers as readings of the same RealAlgOrder endpoint spine: the rational endpoint witnesses, transported real names, hsame rows, continuation routes, provenance, and local naming certificate are shared before any consumer-specific repacking.

Next apply Definition 38.61. The positive-boundary row exported by $\mathsf{RealAlgOrder}^{\uparrow}$ is read by $\mathsf{Met}^{\uparrow}$ as its positive-distance row without adding host metric structure or completed-real equality.

Then apply Definition 38.63. The same positive-boundary row is read by $\mathsf{S}^{1\uparrow}$ through the displayed component rows, product carrier, unit-equation witness, componentwise classifier rows, and equation transports, without adding an angle quotient.

Finally combine the transport statement Definition 38.64 with Definition 38.59. The former keeps the shared boundary stable under the displayed hsame transports; the latter confines the conclusion to the visible endpoint, apartness, metric, component, transport, continuation, provenance, and naming rows. Thus the three readings correspond exactly as public BEDC rows and carry no trichotomy, completed-real equality, angle quotient, host metric structure, or Real completeness coordinate. □

Theorem 38.66 (RealAlgOrder MetricSpace SOne positive boundary)**.** *For constant rational real names, any* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness packet that is consumed simultaneously by* $\mathsf{Met}^{\uparrow}$ *and by* $\mathsf{S}^{1\uparrow}$ *factors through one shared positive-boundary package. The package consists exactly of the common*

rational endpoint witnesses, their $\mathsf{Real}^{\uparrow}$ *constant names, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness and positive-gap rows, the* $\mathsf{Met}^{\uparrow}$ *positive-distance row, the* $\mathsf{S}^{1\uparrow}$ *component product and unit-equation rows, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *rows.*

The package supplies no trichotomy row, quotient equality, host metric structure, compactness, connectedness, or Real completeness datum.

Proof. Begin with Definition 38.43. It fixes the constant-rational endpoints as rational witnesses transported through the $\mathsf{Real}^{\uparrow}$ e-one boundary, and it displays the carrier, classifier, order, apartness, hsame, Cont, Pkg, and local naming rows available to the $\mathsf{RealAlgOrder}^{\uparrow}$ packet.

Next apply Definition 38.61. The apartness or positive-gap row is repacked as the $\mathsf{Met}^{\uparrow}$ positive-distance component without changing the endpoints, transports, continuation routes, provenance, or local certificate.

Use Definition 38.63 for the circle-side consumer. The $\mathsf{S}^{1\uparrow}$ row reads the same real component endpoints, the product carrier, the unit-equation witness, componentwise classifier rows, and equation transports; it does not introduce an angle quotient or an independent equality surface.

Finally compare the two consumer reads with Definition 38.60 and Definition 57.313. The shared row set is exactly the displayed endpoint, positive-distance, product, unit-equation, transport, continuation, provenance, and local certificate package, so no trichotomy, quotient equality, host metric structure, compactness, connectedness, or Real completeness coordinate is present. □

Theorem 38.67 (RealAlgOrder constant apartness SOne unit boundary)**.** *For constant rational real names feeding an accepted* $\mathsf{S}^{1\uparrow}$ *component-and-unit-equation consumer, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness boundary used by that consumer is exactly the constant-rational positive-boundary package of Definition 38.66. It contains the two displayed real component rows, the product carrier, the unit-equation witness, the positive-distance handoff through* $\mathsf{Met}^{\uparrow}$*, the* hsame *and* Cont *transport rows, the* Pkg *provenance row, and the local* NameCert *rows.*

It does not supply an angle quotient, completed-real equality, trichotomy, host metric structure, compactness, connectedness, or Real completeness.

Proof. Project the accepted circle consumer through Definition 38.63. The theorem identifies the $\mathsf{S}^{1\uparrow}$ unit-equation use of RealAlgOrder apartness with the same endpoint witnesses, real component rows, product carrier, unit-equation witness, componentwise classifier rows, and equation transports.

Then feed the underlying apartness row through Definition 38.61. This records the positive-distance component consumed by $\mathsf{Met}^{\uparrow}$ while leaving the rational endpoints, $\mathsf{Real}^{\uparrow}$ names, hsame rows, Cont routes, Pkg provenance, and local NameCert rows unchanged.

Apply Definition 38.66 to identify the common package seen by both consumers. Since that package has only the displayed constant-rational, metric-distance, product, unit-equation, transport, provenance, and certificate rows, an angle quotient or completed-real equality row has no source coordinate from which it can be read. □

Theorem 38.68 (RealAlgOrder metric consumer order stability)**.** *For constant rational real names, every* $\mathsf{Met}^{\uparrow}$ *positive-distance consumer obtained from a* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness packet preserves the original RealAlgOrder order and apartness row under the displayed* $\mathsf{Real}^{\uparrow}$ *classifier transport. The consumer may read the positive-distance component, endpoint witnesses,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row, but the transported order and apartness source remains the same* $\mathsf{RealAlgOrder}^{\uparrow}$ *row certified by the constant-rational interface.*

The stability statement does not assert trichotomy, quotient equality, host metric structure, compactness, connectedness, or Real completeness.

Proof. Start from Definition 38.56. It says that the algebra, order, apartness, and public metric-facing rows are transported from the same rational classifier surface through the displayed $\mathsf{Real}^{\uparrow}$ constant classifier.

Next apply Definition 38.61. The $\mathsf{Met}^{\uparrow}$ consumer receives the positive-distance component by repacking the RealAlgOrder apartness row, so the metric-facing read has the same endpoint witnesses, hsame rows, continuation routes, provenance, and local certificate.

Finally use Definition 38.59 together with Definition 38.66. These results limit the consumer surface to the displayed order, apartness, metric-distance, transport, continuation, provenance, and certificate rows. The order/apartness row is therefore stable under classifier transport, and no trichotomy, quotient equality, host metric, compactness, connectedness, or completeness coordinate is created by the metric consumer. □

Theorem 38.69 (Real constant classifier inner e-zero endpoint absurd)**.** *For histories* $tail, d$: BHist*, the constant real-history classifier excludes an* $\mathsf{E0}$ *rational endpoint even when it is sealed by the outer constant-real* $\mathsf{E1}$ *constructor:*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E0}(tail)), \mathsf{E1}(d)) \Longrightarrow \mathsf{False},$$
$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{E0}(tail))) \Longrightarrow \mathsf{False}.$$

Proof. Expose the sealed constant-real classifier with Definition 38.11. The resulting rational classifier has an $\mathsf{E0}$ endpoint on one side, so the RatUp endpoint exclusion applies in the corresponding orientation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 e0 endpoint absurd`

Theorem 38.70 (Real constant classifier appended e-zero endpoint absurd)**.** *For histories* $head, tail, d$: BHist*, the constant real-history classifier excludes an appended denominator whose visible tail is headed by* $\mathsf{E0}$*, even under the outer constant-real* $\mathsf{E1}$ *seal:*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(head \cdot \mathsf{E0}(tail)), \mathsf{E1}(d)) \Longrightarrow \mathsf{False},$$
$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(head \cdot \mathsf{E0}(tail))) \Longrightarrow \mathsf{False}.$$

Proof. Expose the sealed constant-real classifier with Definition 38.11. The resulting rational classifier supplies a positive unary denominator witness for the appended endpoint. That contradicts Definition 37.15 in the corresponding orientation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 append e0 endpoint absurd`

Theorem 38.71 (Real constant classifier transported inner e-zero endpoint absurd)**.** *For histories* $h, k, tail$: BHist*, any constant real-history classifier excludes an* $\mathsf{E1}(\mathsf{E0}(tail))$ *display on either classified endpoint:*

$$\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{E0}(tail))) \Longrightarrow \mathsf{False}),$$
$$\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{E0}(tail))) \Longrightarrow \mathsf{False}).$$

Proof. For the left endpoint, unpack the constant real classifier only far enough to recover the opposite endpoint display $k\mathsf{hsameE1}(d)$ for some d. Together with the assumed display $h\mathsf{hsameE1}(\mathsf{E0}(tail))$, Definition 38.18 transports the classifier to the exact pair $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E0}(tail)), \mathsf{E1}(d))$. Then Definition 38.69 contradicts this exact sealed-zero shape.

The right endpoint is symmetric. Unpack the classifier to recover $h\mathsf{hsameE1}(d)$ for some d, combine it with the assumed $k\mathsf{hsameE1}(\mathsf{E0}(tail))$, transport to $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{E0}(tail)))$, and apply the opposite orientation of Definition 38.69. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier transported inner e0 endpoint absurd`

Theorem 38.72 (Real constant classifier inner empty endpoint absurd)**.** *For histories d : BHist, the constant real-history classifier excludes an empty rational endpoint even when it is sealed by the outer constant-real* E1 *constructor:*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(d)) \Longrightarrow \mathsf{False},$$
$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}.$$

Proof. Expose the sealed constant-real classifier with Definition 38.11. In the first orientation this gives a rational classifier $\mathsf{RatHistoryClassifier}(\mathsf{e}, d)$; in the second orientation it gives $\mathsf{RatHistoryClassifier}(d, \mathsf{e})$.

Apply Definition 37.51 to the exposed rational classifier in the corresponding orientation. The relevant endpoint nonemptiness clause contradicts $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$ by reflexivity. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier e1 empty endpoint absurd`

Theorem 38.73 (Real constant classifier transported inner empty endpoint absurd)**.** *For histories h, k : BHist, any constant real-history classifier excludes an* $\mathsf{E1}(\mathsf{e})$ *display on either classified endpoint:*

$$\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}),$$
$$\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}).$$

Proof. For the left endpoint, unpack the constant real classifier only far enough to recover the opposite endpoint display $k\mathsf{hsameE1}(d)$ for some d. Combine that display with the assumed $h\mathsf{hsameE1}(\mathsf{e})$ and apply Definition 38.18. This gives the exact transported classifier $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(d))$.

The left orientation of Definition 38.72 contradicts that transported classifier.

For the right endpoint, unpack the classifier to recover $h\mathsf{hsameE1}(d)$ for some d. Combine it with the assumed $k\mathsf{hsameE1}(\mathsf{e})$, transport to $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{e}))$, and apply the right orientation of Definition 38.72. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier transported inner empty endpoint absurd`

Theorem 38.74 (Real constant classifier appended e-zero tail absurd)**.** *For histories $d, e, tail$: BHist, the constant real-history classifier excludes a sealed rational denominator whose visible denominator ends in an appended zero-headed tail:*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d \cdot \mathsf{E0}(tail)), \mathsf{E1}(e)) \Longrightarrow \mathsf{False},$$
$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(e), \mathsf{E1}(d \cdot \mathsf{E0}(tail))) \Longrightarrow \mathsf{False}.$$

Proof. For the left orientation, expose the constant RealUp classifier with Definition 38.11. The resulting RatUp classifier relates $d \cdot \mathsf{E0}(tail)$ to e, so Definition 37.50 gives the left positive-denominator row. This contradicts Definition 37.15.

For the right orientation, expose the same constant RealUp classifier with Definition 38.11. The resulting RatUp classifier relates e to $d \cdot \mathsf{E0}(tail)$, so Definition 37.50 gives the right positive-denominator row. This again contradicts Definition 37.15. □

The ten-field fit is: traditional target constant RealUp denominator endpoint exclusion; BEDC source RealUp constant classifiers plus RatUp denominator shape; carrier BHist; classifier RealConstantHistoryClassi stability by e-one exposure and positive-denominator transport; ledger visible append head and zero-headed tail; core theorem Definition 38.74; exactness impossibility of appended zero-headed denominators; standard bridge finite rational denominator sanity; status ProofTarget.

Theorem 38.75 (Real constant classifier malformed inner denominator exclusions)**.** *For histories* $d, p, z : \mathsf{BHist}$*, the constant real-history classifier excludes every exposed rational denominator that is internally empty or ends with an appended e-zero tail:*

$$\begin{aligned}
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(d)) \Longrightarrow \mathsf{False},\\
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False},\\
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(p \cdot \mathsf{E0}(z)), \mathsf{E1}(d)) \Longrightarrow \mathsf{False},\\
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(p \cdot \mathsf{E0}(z))) \Longrightarrow \mathsf{False}.
\end{aligned}$$

In the ten-field fit, the traditional target is constant-real malformed denominator exclusion; the BEDC source is BHist *e-one rational seals and RatUp positive-denominator shape; the carrier is* RealConstantHistoryCarrier*; the classifier is* RealConstantHistoryClassifier*; the stability obligations are e-one exposure,* hsame *reflexivity, and RatUp denominator projection; the ledger is empty and appended e-zero denominator shapes; the core theorem is this four-way exclusion; the exactness predicate is impossibility of nonpositive exposed denominator tails; the standard bridge is constant rational real names; and the status is* ProofTarget*.*

Proof. For each clause, first expose the sealed constant-real classifier with Definition 38.11. The displayed assumption becomes a RatUp classifier on the two inner histories.

For the two inner-empty clauses, apply Definition 37.51 to the exposed RatUp classifier. In the left orientation the first nonemptiness row contradicts $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$ by reflexivity; in the right orientation the second nonemptiness row gives the same contradiction.

For the left appended e-zero-tail clause, apply Definition 37.50 to the exposed RatUp classifier. The left projection gives a positive unary denominator witness for $p \cdot \mathsf{E0}(z)$, contradicting Definition 37.15.

For the right appended e-zero-tail clause, the same positive-denominator projection gives the witness on the right endpoint $p \cdot \mathsf{E0}(z)$. This is again impossible by Definition 37.15. □

Theorem 38.76 (Real constant classifier transported malformed inner denominator exclusions)**.** *For histories* $h, k, p_h, z_h, p_k, z_k : \mathsf{BHist}$*, a constant real-history classifier excludes transported empty sealed endpoints and transported appended e-zero sealed denominators on either endpoint:*

$$\begin{aligned}
&\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(p_h \cdot \mathsf{E0}(z_h))) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(p_k \cdot \mathsf{E0}(z_k))) \Longrightarrow \mathsf{False}).
\end{aligned}$$

Proof. The two empty endpoint clauses are exactly the transported inner-empty endpoint exclusions from Definition 38.73.

For the left appended e-zero clause, unpack the constant classifier just far enough to recover an opposite endpoint display $k\mathsf{hsameE1}(d)$ for some history d. Together with the assumed display $h\mathsf{hsameE1}(p_h \cdot \mathsf{E0}(z_h))$, Definition 38.18 transports the classifier to $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(p_h \cdot \mathsf{E0}(z_h)), \mathsf{E1}(d))$. The left clause of Definition 38.74 contradicts this transported classifier.

For the right appended e-zero clause, unpack symmetrically to recover $h\mathsf{hsameE1}(d)$ for some history d. Transport the assumed display $k\mathsf{hsameE1}(p_k \cdot \mathsf{E0}(z_k))$ across Definition 38.18, obtaining $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(p_k \cdot \mathsf{E0}(z_k)))$. The right clause of Definition 38.74 gives the contradiction. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier transported malformed inner denominator absurd`

The ten-field fit is: traditional target transported RealUp denominator sanity; BEDC source RealUp constant classifiers plus RatUp positive-denominator shape; carrier BHist; classifier RealConstantHistoryClas stability row endpoint transport; ledger transported empty endpoints and appended e-zero denominator displays; core theorem Definition 38.76; exactness impossibility of empty or zero-headed sealed denominators; downstream bridge RealUp constant-name denominator sanity; status ProofTarget.

Theorem 38.77 (Real constant classifier transported complete malformed inner denominator exclusions)**.** *For histories $h, k, t_h, t_k, p_h, z_h, p_k, z_k$: BHist, a constant real-history classifier excludes every transported malformed sealed denominator boundary: inner empty, inner e-zero, and appended e-zero denominators on either endpoint:*

$$\begin{aligned}
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{E0}(t_h))) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{E0}(t_k))) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(h, \mathsf{E1}(p_h \cdot \mathsf{E0}(z_h))) \Longrightarrow \mathsf{False}),\\
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow (\mathsf{hsame}(k, \mathsf{E1}(p_k \cdot \mathsf{E0}(z_k))) \Longrightarrow \mathsf{False}).
\end{aligned}$$

Proof. Project the first two displayed rows from Definition 38.76. These give the left and right transported inner-empty exclusions.

Project the two e-zero rows from Definition 38.71, using t_h for the left endpoint and t_k for the right endpoint.

Finally project the last two displayed rows from Definition 38.76. Together the empty rows, the e-zero rows, and the appended e-zero rows have exactly the six displayed orientations. □

The ten-field fit is: traditional target RealUp constant malformed sealed-denominator exclusion; BEDC source RealUp constant classifier plus RatUp denominator shape; carrier BHist; classifier RealConstantHistoryClassifier; stability endpoint transport; ledger empty, e-zero, and appended e-zero inner shapes; core theorem Definition 38.77; exactness full transported malformed sealed-denominator impossibility; standard bridge constant rational real names; status ProofTarget.

Theorem 38.78 (Real constant classifier full malformed endpoint exclusion package)**.** *For histories $h, k, tail_h, tail_k, inner_h, inner_k, p_h, p_k, z_h, z_k$: BHist, a constant real-history classifier excludes every malformed outer display and every sealed malformed inner display on either endpoint:*

$$\begin{aligned}
&\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow\\
&\quad (\mathsf{hsame}(h,\mathsf{e}) \Longrightarrow \mathsf{False}) \wedge (\mathsf{hsame}(k,\mathsf{e}) \Longrightarrow \mathsf{False}) \wedge\\
&\quad (\mathsf{hsame}(h, \mathsf{E0}(tail_h)) \Longrightarrow \mathsf{False}) \wedge (\mathsf{hsame}(k, \mathsf{E0}(tail_k)) \Longrightarrow \mathsf{False}) \wedge\\
&\quad (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}) \wedge (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{False}) \wedge\\
&\quad (\mathsf{hsame}(h, \mathsf{E1}(\mathsf{E0}(inner_h))) \Longrightarrow \mathsf{False}) \wedge (\mathsf{hsame}(k, \mathsf{E1}(\mathsf{E0}(inner_k))) \Longrightarrow \mathsf{False}) \wedge\\
&\quad (\mathsf{hsame}(h, \mathsf{E1}(p_h \cdot \mathsf{E0}(z_h))) \Longrightarrow \mathsf{False}) \wedge (\mathsf{hsame}(k, \mathsf{E1}(p_k \cdot \mathsf{E0}(z_k))) \Longrightarrow \mathsf{False}).
\end{aligned}$$

Proof. We prove the ten clauses by grouping the visible endpoint shapes.

1. For the outer empty displays, transport the classifier along $\mathsf{hsame}(h,\mathsf{e})$ or $\mathsf{hsame}(k,\mathsf{e})$ and the reflexive display on the other endpoint. This gives an exact classifier with an e endpoint, so the corresponding empty endpoint clause of Definition 38.22 gives the contradiction.

2. For the outer E0 displays, use the same endpoint transport with $\mathsf{hsame}(h, \mathsf{E0}(tail_h))$ or $\mathsf{hsame}(k, \mathsf{E0}(tail_k))$. The transported classifier has an exact E0 endpoint, and the e-zero endpoint clause of Definition 38.22 excludes it.

3. For the sealed $\mathsf{E1}(\mathsf{E0}(-))$ displays, apply Definition 38.71 directly to the original classifier. Its two projections are exactly the two required exclusions for $inner_h$ and $inner_k$.
4. For the sealed empty displays, unpack the classifier enough to recover an opposite sealed endpoint $k\mathsf{hsameE1}(d)$ or $h\mathsf{hsameE1}(d)$. Endpoint transport then gives an exact classifier $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(d))$ or $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(\mathsf{e}))$. The two orientations of Definition 38.72 exclude these exact sealed-empty classifiers.
5. For the appended e-zero sealed displays, unpack the same opposite sealed endpoint and transport to $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(p_h{\cdot}\mathsf{E0}(z_h)), \mathsf{E1}(d))$ or $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d), \mathsf{E1}(p_k{\cdot}\mathsf{E0}(z_k)))$. The two orientations of Definition 38.74 contradict those transported classifiers.

□

The ten-field fit is: traditional target full malformed constant-real endpoint coverage package; BEDC source constant RealUp classifier endpoint transport and sealed rational endpoint exclusions; carrier BHist; classifier RealConstantHistoryClassifier; stability row outer endpoint transport plus sealed endpoint readback; ledger outer empty, outer e-zero, sealed empty, sealed e-zero, and sealed appended e-zero displays on both sides; core theorem Definition 38.78; exactness no malformed outer or sealed inner constant-real endpoint display on either endpoint; downstream bridge RealUp carrier, NameCert, and dyadic embedding endpoint sanity; status ProofTarget.

Definition 38.79 (Malformed sealed real denominator display)**.** *For a history $h : \mathsf{BHist}$, the predicate* $\mathsf{MalformedRealSealedDenom}(h)$ *records exactly the three sealed denominator displays excluded by the transported RealUp denominator sanity ledger: empty, inner e-zero, and appended e-zero denominators.*

$$\mathsf{MalformedRealSealedDenom}(h) := \mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})) \vee \exists t : \mathsf{BHist},\ \mathsf{hsame}(h, \mathsf{E1}(\mathsf{E0}(t))) \vee \exists p, z : \mathsf{BHist},\ \mathsf{hsame}(h, \mathsf{E1}(p{\cdot}\mathsf{E0}$$

Lean4 (definition) `BEDC.Derived.RealUp.MalformedRealSealedDenom`

Theorem 38.80 (Real constant classifier no malformed sealed denominator displays)**.** *For histories $h, k : \mathsf{BHist}$, a constant real-history classifier excludes malformed sealed denominator displays at both transported endpoints:*

$$\mathsf{RealConstantHistoryClassifier}(h, k) \Longrightarrow (\mathsf{MalformedRealSealedDenom}(h) \Longrightarrow \mathsf{False}) \wedge (\mathsf{MalformedRealSealedDenom}(k)$$

Proof. Unfold MalformedRealSealedDenom on the left endpoint. In the empty-display case, apply the first clause of Definition 38.77.

In the inner e-zero-display case on the left endpoint, use the displayed witness t and apply the third clause of the same theorem.

In the appended e-zero-display case on the left endpoint, use the displayed witnesses p, z and apply the fifth clause of the same theorem.

Unfold MalformedRealSealedDenom on the right endpoint. In the empty-display case, apply the second clause of Definition 38.77.

In the inner e-zero-display case on the right endpoint, use the displayed witness t and apply the fourth clause of the same theorem.

In the appended e-zero-display case on the right endpoint, use the displayed witnesses p, z and apply the sixth clause of the same theorem. □

The ten-field fit is: traditional target reusable transported RealUp denominator sanity predicate; BEDC source the six transported malformed sealed-endpoint exclusions; carrier BHist; classifier RealConstantHistoryClassifier; stability row endpoint transport already concentrated in Definition 38.77; ledger exactly transported empty endpoints, inner e-zero endpoints, and appended e-zero denominator displays; core theorem Definition 38.80; exactness one predicate forbids empty, inner e-zero, and appended e-zero sealed denominator displays on either endpoint; downstream bridge RealUp NameCert and dyadic-embedding denominator sanity; status ProofTarget.

Theorem 38.81 (Real constant classifier no transported malformed sealed denominator displays)**.** *For histories h, k, h', k' : BHist, classifier exclusion of malformed sealed real denominators survives endpoint* hsame *transport:*

$$\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow \mathsf{hsame}(h,h') \Longrightarrow \mathsf{hsame}(k,k') \Longrightarrow (\mathsf{MalformedRealSealedDenom}(h') \Longrightarrow \mathsf{Fals}$$

Proof. Unfold MalformedRealSealedDenom at the transported left endpoint. For each displayed malformed shape of h', compose the endpoint row $\mathsf{hsame}(h,h')$ with that display by hsame transitivity.

In the empty case this gives $\mathsf{hsame}(h, \mathsf{E1}(\mathsf{e}))$. In the inner e-zero case, for the displayed witness t, it gives $\mathsf{hsame}(h, \mathsf{E1}(\mathsf{E0}(t)))$. In the appended e-zero case, for displayed witnesses p, z, it gives $\mathsf{hsame}(h, \mathsf{E1}(p \cdot \mathsf{E0}(z)))$.

These three alternatives are exactly the three clauses of $\mathsf{MalformedRealSealedDenom}(h)$. Apply the left projection of Definition 38.80 to the original classifier and the transported malformed display.

Unfold MalformedRealSealedDenom at the transported right endpoint. For each displayed malformed shape of k', compose the endpoint row $\mathsf{hsame}(k,k')$ with that display by hsame transitivity.

In the empty case this gives $\mathsf{hsame}(k, \mathsf{E1}(\mathsf{e}))$. In the inner e-zero case, for the displayed witness t, it gives $\mathsf{hsame}(k, \mathsf{E1}(\mathsf{E0}(t)))$. In the appended e-zero case, for displayed witnesses p, z, it gives $\mathsf{hsame}(k, \mathsf{E1}(p \cdot \mathsf{E0}(z)))$.

These three alternatives are exactly the three clauses of $\mathsf{MalformedRealSealedDenom}(k)$. Apply the right projection of Definition 38.80 to the original classifier and the transported malformed display. □

The ten-field fit is: traditional target transported constant-real denominator sanity; BEDC source BHist, hsame, the RealUp constant classifier, and the malformed sealed denominator predicate; carrier BHist; classifier RealConstantHistoryClassifier; stability row endpoint hsame transport; ledger transported empty, inner e-zero, and appended e-zero sealed denominators; core theorem Definition 38.81; exactness exclusion of malformed transported endpoints; standard bridge constant rational real names; status ProofTarget.

Definition 38.82 (Real constant malformed endpoint display)**.** *For a history h : BHist, the predicate* $\mathsf{MalformedRealEndpoint}(h)$ *records exactly the reusable endpoint displays excluded by the RealUp constant classifier: an outer empty endpoint, an outer e-zero endpoint, or a sealed malformed denominator display.*

$$\mathsf{MalformedRealEndpoint}(h) := \mathsf{hsame}(h, \mathsf{e}) \vee \exists t : \mathsf{BHist},\ \mathsf{hsame}(h, \mathsf{E0}(t)) \vee \mathsf{MalformedRealSealedDenom}(h).$$

Lean4 (definition) `BEDC.Derived.RealUp.MalformedRealEndpoint`

Theorem 38.83 (Real constant classifier no transported malformed endpoint displays)**.** *For histories h, k, h', k' : BHist, constant real-history classifier exclusion of malformed endpoint displays survives endpoint* hsame *transport:*

$$\mathsf{RealConstantHistoryClassifier}(h,k) \Longrightarrow \mathsf{hsame}(h,h') \Longrightarrow \mathsf{hsame}(k,k') \Longrightarrow (\mathsf{MalformedRealEndpoint}(h') \Longrightarrow \mathsf{False}) \wedge ($$

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier no transported malformed endpoint displays`

Proof. Unfold MalformedRealEndpoint at the transported left endpoint. In the outer empty case, compose $\mathsf{hsame}(h, h')$ with $\mathsf{hsame}(h', \mathsf{e})$ by hsame transitivity. Transport the classifier along this left endpoint display and a reflexive right endpoint display, then apply the left-empty clause of Definition 38.22.

In the outer e-zero case, use the displayed witness t and compose $\mathsf{hsame}(h, h')$ with $\mathsf{hsame}(h', \mathsf{E0}(t))$. Transport the classifier to an exact classifier with left endpoint $\mathsf{E0}(t)$, then apply the left e-zero clause of Definition 38.22.

In the sealed denominator case, apply the left projection of Definition 38.81 to the original classifier, the two endpoint transport rows, and the displayed $\mathsf{MalformedRealSealedDenom}(h')$ witness.

Unfold MalformedRealEndpoint at the transported right endpoint. In the outer empty case, compose $\mathsf{hsame}(k, k')$ with $\mathsf{hsame}(k', \mathsf{e})$ by hsame transitivity. Transport the classifier along a reflexive left endpoint display and this right endpoint display, then apply the right-empty clause of Definition 38.22.

In the outer e-zero case, use the displayed witness t and compose $\mathsf{hsame}(k, k')$ with $\mathsf{hsame}(k', \mathsf{E0}(t))$. Transport the classifier to an exact classifier with right endpoint $\mathsf{E0}(t)$, then apply the right e-zero clause of Definition 38.22.

In the sealed denominator case, apply the right projection of Definition 38.81 to the original classifier, the two endpoint transport rows, and the displayed $\mathsf{MalformedRealSealedDenom}(k')$ witness. □

The ten-field fit is: traditional target RealUp constant endpoint hygiene; BEDC source BHist, hsame, and the RealUp constant classifier; carrier BHist; classifier RealConstantHistoryClassifier; stability row endpoint hsame transport; ledger outer empty, outer e-zero, and malformed sealed denominator displays on transported endpoints; core theorem Definition 38.83; exactness no transported malformed RealUp constant endpoint display on either endpoint; downstream bridge RealUp NameCert endpoint sanity; status ProofTarget.

Theorem 38.84 (Real constant classifier appended e-one tail unary readback)**.** *For histories $d, e, tail_D, tail_E$, a constant RealUp classifier whose two rational denominators are displayed as appended e-one tails reads back both unary factors and the rational endpoint classifier:*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d{\cdot}\mathsf{E1}(tail_D)), \mathsf{E1}(e{\cdot}\mathsf{E1}(tail_E))) \Longrightarrow \mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(tail_D) \wedge \mathsf{UnaryHis}$$

Proof. Use the e-one RealUp/RatUp classifier equivalence to expose the underlying RatUp history classifier. Its positive-denominator fields apply to both displayed denominators. The appended e-one denominator decomposition reads each positive denominator as unary evidence for its head and tail, and the remaining classifier field is the displayed endpoint hsame relation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append e1 tail unary readback`

Theorem 38.85 (Real constant classifier common-head e-one tail readback)**.** *For histories $left, d, e, tail_D, tail_E$, a common visible head in two displayed constant RealUp e-one tail denominators can be read back to the five unary factors and the central denominator comparison:*

$$\begin{aligned}&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(left \cdot (d \cdot \mathsf{E1}(tail_D))), \mathsf{E1}(left \cdot (e \cdot \mathsf{E1}(tail_E))))\\ &\quad \Longrightarrow \mathsf{hsame}(tail_D, tail_E) \Longrightarrow\\ &\quad \mathsf{UnaryHistory}(left) \wedge \mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(e) \wedge \mathsf{UnaryHistory}(tail_D) \wedge \mathsf{UnaryHistory}(tail_E) \wedge \mathsf{hsame}(\end{aligned}$$

Proof. Use associativity of BEDC append to expose the endpoints as appended e-one tails with heads $left \cdot d$ and $left \cdot e$. The appended e-one tail readback theorem gives unary evidence for both composite heads, the two tails, and the displayed endpoint comparison. Split the composite unary heads into their factors. The assumed tail comparison identifies the two e-one tail suffixes, so BEDC append right cancellation removes the shared suffix; append left cancellation then removes the common head and yields $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append common head e1 tail readback`

Theorem 38.86 (Real constant classifier appended e-one matched-tail head readback)**.** *For histories* $d, e, tail_D, tail_E$*, if*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(d \cdot \mathsf{E1}(tail_D)), \mathsf{E1}(e \cdot \mathsf{E1}(tail_E))) \wedge \mathsf{hsame}(tail_D, tail_E),$$

then

$$\mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(tail_D) \wedge \mathsf{UnaryHistory}(e) \wedge \mathsf{UnaryHistory}(tail_E) \wedge \mathsf{hsame}(d, e).$$

Proof. Apply Definition 38.84 to the displayed classifier. This gives the four unary rows for d, $tail_D$, e, and $tail_E$, together with endpoint equality $\mathsf{hsame}(d \cdot \mathsf{E1}(tail_D), e \cdot \mathsf{E1}(tail_E))$.

The assumed tail agreement $\mathsf{hsame}(tail_D, tail_E)$ transports through the e-one constructor and right append congruence. Hence the right endpoint may be read with the same displayed tail as the left endpoint: $\mathsf{hsame}(e \cdot \mathsf{E1}(tail_E), e \cdot \mathsf{E1}(tail_D))$.

Endpoint transport therefore presents a constant-real classifier of $\mathsf{E1}(d \cdot \mathsf{E1}(tail_D))$ against $\mathsf{E1}(e \cdot \mathsf{E1}(tail_D))$. Apply Definition 38.13 with common tail $\mathsf{E1}(tail_D)$ to obtain $\mathsf{hsame}(d, e)$, and combine this with the four unary rows already read back. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append e1 matched tail head readback`

The ten-field fit is: traditional target finite denominator-tail agreement readback for RealUp constants; BEDC source RealUp constant classifiers plus RatUp positive-denominator readback; carrier BEDC histories; classifier RealConstantHistoryClassifier; stability by endpoint transport, e-one congruence, append congruence, and common-tail cancellation; ledger displayed denominator heads and displayed e-one tails; core theorem Definition 38.86; exactness the displayed five-row conjunction; standard bridge finite RegSeqRat denominator-tail readback; status ProofTarget.

Theorem 38.87 (Real constant classifier appended e-one tail denominator package)**.** *For histories* $d, e, tail_D, tail_E, z_D, z_E : \mathsf{BHist}$*, write* $D = d \cdot \mathsf{E1}(tail_D)$ *and* $E = e \cdot \mathsf{E1}(tail_E)$*. If*

$$\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(D), \mathsf{E1}(E)),$$

then the appended rational denominators satisfy the following package:

1. $\mathsf{RatHistoryClassifier}(D, E)$*;*
2. $\mathsf{PositiveUnaryDenominator}(D)$ *and* $\mathsf{PositiveUnaryDenominator}(E)$*;*
3. $\mathsf{UnaryHistory}(d)$*,* $\mathsf{UnaryHistory}(tail_D)$*,* $\mathsf{UnaryHistory}(e)$*, and* $\mathsf{UnaryHistory}(tail_E)$*;*
4. $D\mathsf{hsame}e \Longrightarrow \bot$ *and* $E\mathsf{hsame}e \Longrightarrow \bot$*;*
5. $D\mathsf{hsameE0}(z_D) \Longrightarrow \bot$ *and* $E\mathsf{hsameE0}(z_E) \Longrightarrow \bot$*;*
6. $D\mathsf{hsame}E$*.*

Proof. Apply Definition 38.11 to the displayed constant RealUp classifier. This exposes the underlying $\mathsf{RatHistoryClassifier}(D, E)$ and keeps its classifier field available for the final $D\mathsf{hsame}E$ row.

Next apply Definition 37.50 to the rational classifier. The result supplies positive unary denominator witnesses for both appended tails. Applying Definition 37.9 to each witness decomposes the two appended e-one denominators into the four unary factor rows.

Finally apply Definition 37.12 to the two positive-denominator witnesses to obtain the two exclusions against e. If either appended denominator were history-same to a zero-extension, transport the corresponding positive-denominator witness along that sameness and contradict Definition 37.10. The hsame row is the rational classifier endpoint field, equivalently the checked unary readback theorem above. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstantHistoryClassifier append e1 tail denominator package`

Definition 38.88 (Real constant appended-tail denominator package)**.** *For histories* $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$*, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

The predicate $\mathsf{RealConstantAppendedTailDenominatorPackage}(d, e, tail_D, tail_E, z_D, z_E)$ *is the conjunction of the following six rows:*

1. $\mathsf{RatHistoryClassifier}(D, E)$*;*
2. $\mathsf{PositiveUnaryDenominator}(D)$ *and* $\mathsf{PositiveUnaryDenominator}(E)$*;*
3. $\mathsf{UnaryHistory}(d)$*,* $\mathsf{UnaryHistory}(tail_D)$*,* $\mathsf{UnaryHistory}(e)$*, and* $\mathsf{UnaryHistory}(tail_E)$*;*
4. $D\mathsf{hsame}\mathsf{e} \Longrightarrow \bot$ *and* $E\mathsf{hsame}\mathsf{e} \Longrightarrow \bot$*;*
5. $D\mathsf{hsame}\mathsf{E0}(z_D) \Longrightarrow \bot$ *and* $E\mathsf{hsame}\mathsf{E0}(z_E) \Longrightarrow \bot$*;*
6. $D\mathsf{hsame}E$*.*

Definition 38.89 (Real constant appended-tail full denominator-tail package)**.** *For histories* $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$*, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

The predicate $\mathsf{RealConstantAppendedTailFullDenominatorTailPackage}(d, e, tail_D, tail_E, u_D, u_E, z_D, z_E)$ *is the denominator package Definition 38.88 together with*

1. $\mathsf{hsame}(D, u_D \cdot \mathsf{E0}(z_D)) \Longrightarrow \bot$*;*
2. $\mathsf{hsame}(E, u_E \cdot \mathsf{E0}(z_E)) \Longrightarrow \bot$*;*
3. $\mathsf{hsame}(d, e)$*.*

Theorem 38.90 (Real constant appended e-one tail rational-stream bridge readback)**.** *For histories* $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$ *and observation maps* $r, q : \mathsf{Hist} \to \mathsf{Hist}$*, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the four bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let

$$R = \mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(tail_D) \wedge \mathsf{UnaryHistory}(e) \wedge \mathsf{UnaryHistory}(tail_E) \wedge \mathsf{hsame}(D, E).$$

Let $P(D, E, z_D, z_E)$ *abbreviate* $\mathsf{RealConstantAppendedTailDenominatorPackage}(d, e, tail_D, tail_E, z_D, z_E)$ *from Definition 38.88. Then the bridge rows are linked by*

$$(A \Longleftrightarrow B) \wedge (B \Longleftrightarrow U) \wedge (U \Longleftrightarrow C),$$

and every row reads back the same appended-tail unary readback row:

$$(A \Longrightarrow R) \wedge (B \Longrightarrow R) \wedge (U \Longrightarrow R) \wedge (C \Longrightarrow R),$$

and the same appended-tail denominator package:

$$(A \Longrightarrow P(D, E, z_D, z_E)) \wedge (B \Longrightarrow P(D, E, z_D, z_E)) \wedge (U \Longrightarrow P(D, E, z_D, z_E)) \wedge (C \Longrightarrow P(D, E, z_D, z_E)).$$

Proof. Instantiate Definition 38.29 with the histories D and E and the observation maps r and q. The three resulting equivalences are exactly the displayed links $A \Longleftrightarrow B$, $B \Longleftrightarrow U$, and $U \Longleftrightarrow C$.

The direct readback from C is Definition 38.87, applied to the decompositions $D = d \cdot \mathsf{E1}(tail_D)$ and $E = e \cdot \mathsf{E1}(tail_E)$ with the displayed zero witnesses z_D and z_E. This gives every row of $P(D, E, z_D, z_E)$ at once, with projections taken from Definition 38.88.

The direct readback from A starts with the rational classifier $\mathsf{RatHistoryClassifier}(D, E)$. Apply Definition 37.50 to obtain both positive unary denominator rows. Apply Definition 37.9 to the two positive-denominator witnesses to read back the four unary factors.

Next apply Definition 37.12 to obtain both exclusions against e. If either appended denominator were history-same to $\mathsf{E0}(z_D)$ or $\mathsf{E0}(z_E)$, transport the corresponding positive-denominator witness across that equality and contradict Definition 37.10. The final $D\mathsf{hsame}E$ row is the endpoint sameness field of the rational classifier.

For B, transport along $A \Longleftrightarrow B$ back to A and use the direct rational readback. For U, either transport through $B \Longleftrightarrow U$ and then $A \Longleftrightarrow B$, or transport through $U \Longleftrightarrow C$ and use the direct constant-history readback. The unary row R is the unary-factor and endpoint-sameness part of the same package. Thus every bridge classifier row recovers R and the full package $P(D, E, z_D, z_E)$. □

Theorem 38.91 (Real constant appended-tail bridge denominator package)**.** *For histories* $d, e, tail_D, tail_E, z_D, z_E$ Hist *and observation maps* $r, q : \mathsf{Hist} \to \mathsf{Hist}$*, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \textit{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $P(D, E, z_D, z_E)$ *abbreviate* $\mathsf{RealConstantAppendedTailDenominatorPackage}(d, e, tail_D, tail_E, z_D, z_E)$ *from Definition 38.88. Then every bridge row entails this package:*

$$(A \Longrightarrow P(D, E, z_D, z_E)) \wedge (B \Longrightarrow P(D, E, z_D, z_E)) \wedge (U \Longrightarrow P(D, E, z_D, z_E)) \wedge (C \Longrightarrow P(D, E, z_D, z_E)).$$

Proof. The C row gives the package directly. Apply Definition 38.87 to the displayed decompositions $D = d \cdot \mathsf{E1}(tail_D)$ and $E = e \cdot \mathsf{E1}(tail_E)$, with the endpoint zero witnesses z_D and z_E. The rows are the named projections of Definition 38.88.

The bridge equivalences are those of Definition 38.90. From U, use the final bridge equivalence to obtain C, then apply the preceding package readback.

From B, use the second bridge equivalence and then the final bridge equivalence. This reaches the constant real-history row, so the same package readback supplies $P(D, E, z_D, z_E)$.

From A, use all three bridge equivalences in order. The resulting C row again yields the complete appended-tail denominator package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge denominator package`

Theorem 38.92 (Real constant appended-tail bridge denominator package exactness)**.** *For histories $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $P(D, E, z_D, z_E)$ abbreviate $\mathsf{RealConstantAppendedTailDenominatorPackage}(d, e, tail_D, tail_E, z_D, z_E)$ *from Definition 38.88. Then the denominator package is exact for the bridge rows:*

$$P(D, E, z_D, z_E) \iff A \wedge B \wedge U \wedge C.$$

Proof. Assume $P(D, E, z_D, z_E)$. Its first row is the rational history classifier A. Apply Definition 38.90 to the displayed histories and observation maps. The three bridge equivalences carry A to B, then to U, and then to C, so the four bridge rows hold together.

Conversely assume $A \wedge B \wedge U \wedge C$. Project the final row C and apply Definition 38.87 to the decompositions $D = d \cdot \mathsf{E1}(tail_D)$ and $E = e \cdot \mathsf{E1}(tail_E)$, with the zero witnesses z_D and z_E. This recovers every row of $P(D, E, z_D, z_E)$ as named in Definition 38.88. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge denominator package exactness`

Theorem 38.93 (Real constant appended-tail bridge single-row denominator package exactness)**.** *For histories $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, use the same definitions of D, E, the bridge rows A, B, U, C, and the appended-tail denominator package $P(D, E, z_D, z_E)$ as in Definition 38.92. If $D = d \cdot \mathsf{E1}(tail_D)$ and $E = e \cdot \mathsf{E1}(tail_E)$, then each concrete bridge row is exact for the denominator package:*

$$(A \iff P(D, E, z_D, z_E)) \wedge (B \iff P(D, E, z_D, z_E)) \wedge (U \iff P(D, E, z_D, z_E)) \wedge (C \iff P(D, E, z_D, z_E)).$$

Proof. The row-to-package directions are the four projections of Definition 38.91 under the displayed appended-tail decompositions.

For the package-to-row directions, assume $P(D, E, z_D, z_E)$. Apply Definition 38.92 under the same decompositions. The resulting conjunction $A \wedge B \wedge U \wedge C$ supplies the corresponding row by projection.

Combining the four row-to-package projections with these four projections from the exactness conjunction gives the stated rowwise equivalences for A, B, U, C. □

The single-row denominator bridge fit has ten fields: traditional target faithful constant rational embedding into RealUp; BEDC source RatUp plus StreamNameUp plus RealUp; carrier the appended-tail rational histories D, E and the independently reindexed constant streams; classifier rows A, B, U, C; stability from constant streams ignoring r and q; ledger the e-one denominator tails $tail_D, tail_E$ with heads d, e and zero witnesses z_D, z_E; core theorem Definition 38.90 together with Definition 38.91; exactness predicate $P(D, E, z_D, z_E)$ supplied by Definition 38.92; downstream bridge the dyadic RealUp embedding; status ProofTarget.

Theorem 38.94 (Real constant appended-tail bridge appended e-zero exclusion without tail agreement)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $R(D, E, u_D, u_E, z_D, z_E)$ denote the paired exclusions

$$(\mathsf{hsame}(D, u_D \cdot \mathsf{E0}(z_D)) \Longrightarrow \bot) \wedge (\mathsf{hsame}(E, u_E \cdot \mathsf{E0}(z_E)) \Longrightarrow \bot).$$

Then every bridge row entails $R(D, E, u_D, u_E, z_D, z_E)$:

$$(A \Longrightarrow R(D, E, u_D, u_E, z_D, z_E)) \wedge (B \Longrightarrow R(D, E, u_D, u_E, z_D, z_E)) \wedge (U \Longrightarrow R(D, E, u_D, u_E, z_D, z_E)) \wedge (C \Longrightarrow R(D,$$

Proof. Apply Definition 38.91. For each of the four bridge rows A, B, U, C, this supplies the package $P(D, E, z_D, z_E)$.

Project the two positive-denominator rows of P. If $\mathsf{hsame}(D, u_D \cdot \mathsf{E0}(z_D))$, transport the $\mathsf{PositiveUnaryDenominator}(D)$ witness across the displayed appended e-zero-tail sameness and contradict Definition 37.15.

The same argument applied to E transports the $\mathsf{PositiveUnaryDenominator}(E)$ witness across $\mathsf{hsame}(E, u_E \cdot \mathsf{E0}(z_E))$ and gives the second contradiction. These two contradictions are exactly $R(D, E, u_D, u_E, z_D, z_E)$ for each bridge row. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge appended e0 tail absurd`

Theorem 38.95 (Real constant appended-tail bridge full denominator-tail package)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $P(D, E, z_D, z_E)$ abbreviate $\mathsf{RealConstantAppendedTailDenominatorPackage}(d, e, tail_D, tail_E, z_D, z_E)$ *from Definition 38.88. Let $Q(D, E, u_D, u_E, z_D, z_E)$ abbreviate* $\mathsf{RealConstantAppendedTailFullDenominatorTailPackag}$ *from Definition 38.89. If $\mathsf{hsame}(tail_D, tail_E)$, then every bridge row entails this full denominator-tail package:*

$$(A \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (B \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (U \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (C \Longrightarrow Q(D$$

Proof. Assume $\mathsf{hsame}(tail_D, tail_E)$. Apply Definition 38.91. For each of the four bridge rows A, B, U, C, this supplies the package $P(D, E, z_D, z_E)$.

Independently, Definition 38.94 gives the two appended e-zero-tail exclusions for D and E under each bridge row.

It remains to determine the matched heads. The final row of P gives $D\mathsf{hsame}E$. The tail hypothesis transports through the $\mathsf{E1}$ constructor, and append congruence rewrites $e \cdot \mathsf{E1}(tail_E)$ as $e \cdot \mathsf{E1}(tail_D)$. Thus the endpoint comparison has a common right suffix $\mathsf{E1}(tail_D)$, so BEDC append right cancellation yields $\mathsf{hsame}(d, e)$.

These three consequences together with the already recovered package P are exactly the named projections of $Q(D, E, u_D, u_E, z_D, z_E)$ from Definition 38.89 for each bridge row. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge full denominator tail package`

Theorem 38.96 (Real constant appended-tail bridge full denominator-tail package exactness)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, use the same definitions of D, E, the bridge rows A, B, U, C, and the full package $Q(D, E, u_D, u_E, z_D, z_E)$ as in Definition 38.95. If $\mathsf{hsame}(tail_D, tail_E)$, then the full denominator-tail package is exact for the bridge rows:*

$$Q(D, E, u_D, u_E, z_D, z_E) \iff A \wedge B \wedge U \wedge C.$$

Proof. Assume $Q(D, E, u_D, u_E, z_D, z_E)$. Project its $P(D, E, z_D, z_E)$ component and apply Definition 38.92. This gives the four bridge rows $A \wedge B \wedge U \wedge C$.

Conversely assume $\mathsf{hsame}(tail_D, tail_E)$ and $A \wedge B \wedge U \wedge C$. Apply Definition 38.95 under the tail agreement. The first projected implication has type $A \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)$, and the projected row A supplies its premise. Hence the full package $Q(D, E, u_D, u_E, z_D, z_E)$ holds. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge full denominator tail package exactness`

Theorem 38.97 (Real constant appended-tail bridge common-head full denominator-tail package)**.** *For histories $left, d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = left \cdot (d \cdot \mathsf{E1}(tail_D)) \qquad \text{and} \qquad E = left \cdot (e \cdot \mathsf{E1}(tail_E)).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $P(D, E, z_D, z_E)$ be the appended-tail denominator package of Definition 38.91 applied to the heads $left \cdot d$ and $left \cdot e$. Let $Q(D, E, u_D, u_E, z_D, z_E)$ denote $P(D, E, z_D, z_E)$ together with

1. $\mathsf{hsame}(D, u_D \cdot \mathsf{E0}(z_D)) \Longrightarrow \bot$ *and* $\mathsf{hsame}(E, u_E \cdot \mathsf{E0}(z_E)) \Longrightarrow \bot$;
2. $\mathsf{UnaryHistory}(left)$, $\mathsf{UnaryHistory}(d)$, $\mathsf{UnaryHistory}(e)$, $\mathsf{UnaryHistory}(tail_D)$, *and* $\mathsf{UnaryHistory}(tail_E)$;
3. $\mathsf{hsame}(D, E)$;
4. $\mathsf{hsame}(d, e)$.

If $\mathsf{hsame}(tail_D, tail_E)$, then every bridge row entails this common-head full denominator-tail package:

$$(A \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (B \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (U \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (C \Longrightarrow Q(D$$

Proof. Assume $\mathsf{hsame}(tail_D, tail_E)$. Apply Definition 38.95 with the denominator heads $left \cdot d$ and $left \cdot e$. For each of the four bridge rows A, B, U, C, this supplies the denominator package $P(D, E, z_D, z_E)$, the two appended e-zero-tail exclusions, and the composite head comparison $\mathsf{hsame}(left \cdot d, left \cdot e)$.

The same package supplies unary evidence for the composite heads $left \cdot d$ and $left \cdot e$, and for the two tails. Split the unary append factors in the two composite heads. This gives unary rows for $left$, d, and e, together with the already recovered unary rows for $tail_D$ and $tail_E$.

It remains to expose the central head equality. The displayed comparison $D\mathsf{hsame}E$ has the two endpoints $left \cdot (d \cdot \mathsf{E1}(tail_D))$ and $left \cdot (e \cdot \mathsf{E1}(tail_E))$. The tail hypothesis identifies the two e-one suffixes, and the existing common-tail cancellation gives $\mathsf{hsame}(left \cdot d, left \cdot e)$.

Finally apply append-left cancellation to the common head $left$. This removes the shared visible context and yields $\mathsf{hsame}(d, e)$. Combining these rows gives $Q(D, E, u_D, u_E, z_D, z_E)$ for each bridge classifier row. □

The common-head full denominator-tail fit has ten fields: traditional target constant rational denominator canonicalization under a shared stream context; BEDC source RatUp plus StreamNameUp plus RealUp; carrier the common-head appended-tail rational histories D, E and their reindexed constant streams; classifier rows A, B, U, C; stability from constant-stream reindexing, e-one-tail transport, and append cancellation; ledger the visible head $left$, central heads d, e, tails $tail_D, tail_E$, and zero witnesses z_D, z_E; core theorem list Definition 38.95 and Definition 38.85; exactness predicate $Q(D, E, u_D, u_E, z_D, z_E)$; standard bridge the constant RatUp-to-RealUp stream bridge; status ProofTarget.

Theorem 38.98 (Real constant appended-tail bridge common-head full denominator-tail package exactness)**.** *For histories $left, d, e, tail_D, tail_E, u_D, u_E, z_D, z_E$: Hist and observation maps r, q : Hist $\to$ Hist, use the same definitions of D, E, the bridge rows A, B, U, C, and the common-head package $Q(D, E, u_D, u_E, z_D, z_E)$ as in Definition 38.97. If $\mathsf{hsame}(tail_D, tail_E)$, then the common-head full denominator-tail package is exact for the four bridge rows:*

$$Q(D, E, u_D, u_E, z_D, z_E) \Longleftrightarrow A \wedge B \wedge U \wedge C.$$

Proof. Assume $Q(D, E, u_D, u_E, z_D, z_E)$. Project its appended-tail denominator package $P(D, E, z_D, z_E)$, where the denominator heads are $left \cdot d$ and $left \cdot e$. Apply Definition 38.92 to obtain $A \wedge B \wedge U \wedge C$.

Conversely assume $\mathsf{hsame}(tail_D, tail_E)$ and $A \wedge B \wedge U \wedge C$. Project the row A from the conjunction. Apply Definition 38.97 under the tail agreement; its first projection has type $A \Longrightarrow Q(D, E, u_D, u_E, z_D, z_E)$. The projected row A supplies the premise, so $Q(D, E, u_D, u_E, z_D, z_E)$ holds. □

Theorem 38.99 (Real constant appended-tail bridge single-row full denominator-tail exactness)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E$: Hist and observation maps r, q : Hist $\to$ Hist, use the same definitions of D, E, the bridge rows A, B, U, C, and the full package $Q(D, E, u_D, u_E, z_D, z_E)$ as in Definition 38.95. If $\mathsf{hsame}(tail_D, tail_E)$, then each concrete bridge row is exact for the full denominator-tail package:*

$$(A \Longleftrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (B \Longleftrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (U \Longleftrightarrow Q(D, E, u_D, u_E, z_D, z_E)) \wedge (C \Longleftrightarrow Q($$

Proof. Assume $\mathsf{hsame}(tail_D, tail_E)$. The row-to-package directions are the four projections of Definition 38.95 under this tail agreement.

For the package-to-row directions, assume $Q(D, E, u_D, u_E, z_D, z_E)$. Apply Definition 38.96 under the same tail agreement. The resulting conjunction $A \wedge B \wedge U \wedge C$ supplies the corresponding row by projection.

Combining the four row-to-package projections with these four projections from the exactness conjunction gives the stated rowwise equivalences for A, B, U, C. □

Theorem 38.100 (Real constant appended-tail bridge head-tail agreement equivalence)**.** *For histories $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Then each bridge row identifies denominator-head agreement with displayed e-one-tail agreement:

$$(A \Longrightarrow (\mathsf{hsame}(d, e) \Longleftrightarrow \mathsf{hsame}(tail_D, tail_E))) \wedge (B \Longrightarrow (\mathsf{hsame}(d, e) \Longleftrightarrow \mathsf{hsame}(tail_D, tail_E))) \wedge (U \Longrightarrow (\mathsf{hsame}(d, e$$

Proof. Fix one of the bridge rows A, B, U, C. Apply Definition 38.91 to the same displayed data and the chosen row. This recovers the denominator package $P(D, E, z_D, z_E)$, whose final component is $D\mathsf{hsame}E$.

For the direction from displayed tail agreement to head agreement, assume $\mathsf{hsame}(tail_D, tail_E)$. Transport this equality through the $\mathsf{E1}$ constructor and the right append context $e \cdot (-)$, so the endpoint comparison $D\mathsf{hsame}E$ is read against the common suffix $\mathsf{E1}(tail_D)$. BEDC right append cancellation then yields $\mathsf{hsame}(d, e)$.

For the direction from head agreement to displayed tail agreement, assume $\mathsf{hsame}(d, e)$. Transport this equality through the left append context $(-) \cdot \mathsf{E1}(tail_E)$, so the same endpoint comparison is read with common left head d. BEDC left append cancellation gives $\mathsf{hsame}(\mathsf{E1}(tail_D), \mathsf{E1}(tail_E))$, and e-one constructor inversion gives $\mathsf{hsame}(tail_D, tail_E)$.

The argument uses only the selected row to recover $D\mathsf{hsame}E$, so repeating it for A, B, U, and C gives the four displayed implications. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge head tail agreement iff`

Theorem 38.101 (Real constant appended-tail bridge malformed sealed denominator exclusion)**.** *For histories $d, e, tail_D, tail_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge classifier rows as

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

Let $M(D, E)$ denote the paired malformed sealed denominator exclusion

$$(\mathsf{MalformedRealSealedDenom}(\mathsf{E1}(D)) \Longrightarrow \mathsf{False}) \wedge (\mathsf{MalformedRealSealedDenom}(\mathsf{E1}(E)) \Longrightarrow \mathsf{False}).$$

Then every bridge row entails the same exclusion predicate:

$$(A \Longrightarrow M(D, E)) \wedge (B \Longrightarrow M(D, E)) \wedge (U \Longrightarrow M(D, E)) \wedge (C \Longrightarrow M(D, E)).$$

Proof. Apply Definition 38.91 to the displayed data. For any chosen bridge row among A, B, U, C, this recovers the package $P(D, E, z_D, z_E)$, including the positive-denominator rows and the nonempty rows for D and E.

Unfold Definition 38.79 at $\mathsf{E1}(D)$. The endpoint D has three branches.

1. In the sealed-empty branch, e-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(D), \mathsf{E1}(\mathsf{e}))$ into $D\mathsf{hsame}\mathsf{e}$, contradicting the nonempty row for D from the package.
2. In the sealed inner e-zero branch, use the displayed witness t. E-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(D), \mathsf{E1}(\mathsf{E0}(t)))$ into $D\mathsf{hsame}\mathsf{E0}(t)$. Transport the positive-denominator witness for D across this sameness by Definition 37.6; the transported witness contradicts Definition 37.10.
3. In the sealed appended e-zero branch, use the displayed witnesses p, z. E-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(D), \mathsf{E1}(p \cdot \mathsf{E0}(z)))$ into $D\mathsf{hsame}p \cdot \mathsf{E0}(z)$. Transport the positive-denominator witness for D across this sameness by Definition 37.6; the transported witness contradicts Definition 37.15.

Unfold Definition 38.79 at $\mathsf{E1}(E)$. The endpoint E has the same three branches.

1. In the sealed-empty branch, e-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(E), \mathsf{E1}(\mathsf{e}))$ into $E\mathsf{hsame}\mathsf{e}$, contradicting the nonempty row for E from the package.
2. In the sealed inner e-zero branch, use the displayed witness t. E-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(E), \mathsf{E1}(\mathsf{E0}(t)))$ into $E\mathsf{hsame}\mathsf{E0}(t)$. Transport the positive-denominator witness for E across this sameness by Definition 37.6; the transported witness contradicts Definition 37.10.
3. In the sealed appended e-zero branch, use the displayed witnesses p, z. E-one constructor inversion turns $\mathsf{hsame}(\mathsf{E1}(E), \mathsf{E1}(p \cdot \mathsf{E0}(z)))$ into $E\mathsf{hsame}p \cdot \mathsf{E0}(z)$. Transport the positive-denominator witness for E across this sameness by Definition 37.6; the transported witness contradicts Definition 37.15.

These two endpoint exclusions are exactly $M(D, E)$ for each of the four bridge-row implications. □

Lean4 ✓ `BEDC.Derived.RealUp.RealConstant appended tail bridge malformed sealed denominator exclusion`

Theorem 38.102 (Real constant appended-tail bridge full denominator sanity exactness)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, use the same definitions of D, E, the bridge rows A, B, U, C, the full denominator-tail package $Q(D, E, u_D, u_E, z_D, z_E)$, and the malformed sealed denominator exclusion $M(D, E)$ as in Definition 38.96 and Definition 38.101. Let $S(D, E, u_D, u_E, z_D, z_E)$ denote the combined denominator-sanity package*

$$S(D, E, u_D, u_E, z_D, z_E) := Q(D, E, u_D, u_E, z_D, z_E) \wedge M(D, E).$$

If $\mathsf{hsame}(tail_D, tail_E)$, then the combined package is exact for the bridge rows:

$$S(D, E, u_D, u_E, z_D, z_E) \iff A \wedge B \wedge U \wedge C.$$

Proof. Assume $\mathsf{hsame}(tail_D, tail_E)$.

For the forward direction, assume $S(D, E, u_D, u_E, z_D, z_E)$. Project its $Q(D, E, u_D, u_E, z_D, z_E)$ component and apply Definition 38.96 under the tail agreement. This gives $A \wedge B \wedge U \wedge C$.

For the reverse direction, assume $A \wedge B \wedge U \wedge C$. Apply Definition 38.96 under the same tail agreement to recover $Q(D, E, u_D, u_E, z_D, z_E)$.

It remains to build $M(D, E)$. Apply Definition 38.101 to the displayed data. The first projected implication has type $A \implies M(D, E)$, and the A row supplied by the row conjunction gives $M(D, E)$.

Conjoining the recovered $Q(D, E, u_D, u_E, z_D, z_E)$ with $M(D, E)$ is exactly $S(D, E, u_D, u_E, z_D, z_E)$. □

The malformed sealed bridge denominator certificate fit has ten fields: traditional target constant rational bridge denominator sanity; BEDC source RatUp plus StreamNameUp plus RealUp over BHist; carrier the appended-tail rational histories D, E; classifiers the four concrete rows A, B, U, C; stability from denominator package readback, e-one inversion, and positive-denominator transport; ledger malformed sealed empty, inner e-zero, and appended e-zero displays; core theorem Definition 38.101; exactness the common exclusion predicate for all bridge rows; standard bridge RegSeqRat constant embedding denominator sanity; status $\mathsf{ProofTarget}$.

The bridge-readback certificate fit has ten fields: traditional target faithful constant rational embedding into RealUp; BEDC source RatUp plus StreamNameUp plus RealUp; carrier the appended-tail rational histories D, E and the independently reindexed constant streams; classifier rows A, B, U, C; stability from constant streams ignoring r and q; ledger the appended e-one denominator tails $tail_D, tail_E$ together with heads d, e and the appended e-zero-tail exclusions; core theorem Definition 38.90 together with Definition 38.91, Definition 38.92, and Definition 38.95, with exactness supplied by Definition 38.96; under tail agreement the equivalence $Q(D, E, u_D, u_E, z_D, z_E) \iff A \wedge B \wedge U \wedge C$ recovers R, the denominator package $P(D, E, z_D, z_E)$, the appended e-zero-tail exclusions, and the matched denominator heads from any bridge classifier row; standard bridge RegSeqRat denominator-tail canonicalization; status $\mathsf{ProofTarget}$.

Theorem 38.103 (Real constant-tail carrier and classifier obligations)**.** *For histories $d, e, tail_D, tail_E, z_D, z_E$: Hist and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

The appended-tail histories D and E provide the RealUp constant-tail carrier obligation, and the four bridge rows

$$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$$

$$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$$

provide the classifier obligation through RatUp, StreamNameUp, unary RealUp, and constant-history RealUp.

Proof. The carrier is the concrete BEDC history carrier already used by the constant-tail readback package: D and E are obtained by BEDC append from the visible denominator heads d, e and their displayed e-one tails.

For the classifier obligation, apply Definition 38.90. Its four named rows are exactly A, B, U, and C under the displayed definitions of D and E.

The denominator package readback is sharpened by Definition 38.92. Thus the same concrete carriers support all four classifier faces without changing the history endpoints or the observation maps. □

Theorem 38.104 (Real constant-tail stability and ledger obligations)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E$ Hist and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge rows as

$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$

$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$

The appended-tail denominator package, the head-tail agreement equivalence, and the malformed sealed-denominator exclusion provide the RealUp constant-tail stability and ledger obligations: the ledger records the visible heads d, e, the displayed e-one tails $tail_D, tail_E$, the zero witnesses z_D, z_E, and the appended e-zero witnesses u_D, u_E, while stability is transport along A, B, U, and C.

Proof. Apply Definition 38.92 to identify the denominator package with the simultaneous bridge rows $A \wedge B \wedge U \wedge C$. This supplies the package ledger: rational classifier, positive denominators, unary heads and tails, nonempty exclusions, zero exclusions, and endpoint sameness.

The head-tail transport is Definition 38.100. It states that, under any selected bridge row, agreement of the visible heads and agreement of the displayed e-one tails determine each other.

The sealed-denominator side of the ledger is Definition 38.101. It rules out the sealed empty, sealed inner e-zero, and sealed appended e-zero displays at both endpoints. These rows are stable because the bridge package recovers them from each of A, B, U, and C. □

Theorem 38.105 (Real constant-tail seal and exactness obligation)**.** *For histories $d, e, tail_D, tail_E, u_D, u_E, z_D, z_E$: Hist and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, set*

$$D = d \cdot \mathsf{E1}(tail_D) \qquad \text{and} \qquad E = e \cdot \mathsf{E1}(tail_E).$$

Write the bridge rows as

$A = \mathsf{RatHistoryClassifier}(D, E), \qquad B = \mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q),$

$U = \mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_D \circ r, \mathsf{RatConstStream}_E \circ q), \qquad C = \mathsf{RealConstantHistoryClassifier}(\mathsf{E1}$

If $\mathsf{hsame}(tail_D, tail_E)$, then the combined denominator-sanity exactness theorem supplies the seal and exactness obligation consumed by the RealUp constant-name boundary of chapter 38: the sanity package for D and E is exact for the same four bridge rows A, B, U, C, and therefore no malformed sealed denominator can pass through the constant-tail boundary.

Proof. The exactness source is Definition 38.102. Under the displayed tail agreement, it identifies the combined sanity package $S(D, E, u_D, u_E, z_D, z_E)$ with $A \wedge B \wedge U \wedge C$.

The Q component of S is the full denominator-tail package from Definition 38.95. It contains the denominator package, appended e-zero exclusions, and matched head equality required by the constant-name boundary.

The M component of S is the malformed sealed-denominator exclusion from Definition 38.101. Together these two components give the seal exactness row: every bridge classifier recovers the same sealed-denominator sanity, and the sanity package recovers the bridge classifiers. □

Theorem 38.106 (Real stream classifier prefix closure)**.** *If rational-observation streams are pointwise related by the rational history classifier, then every finite observation prefix is related by the recursively accumulated prefix classifier.*

Proof. Induct on the selected prefix length. The base case is the pointwise classifier at zero. The successor case pairs the induction hypothesis for the shorter prefix with the pointwise classifier at the successor observation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier prefix`

Theorem 38.107 (Real stream prefix classifier hsame transport)**.** *If two rational-observation streams are related pointwise by* hsame *on both sides, then every classified finite prefix transports to the pointwise-related streams.*

Proof. Induct on the prefix length. The base observation is rational classifier transport at index 0. At a successor index, the induction hypothesis transports the shorter prefix and rational classifier transport handles the successor observation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier hsame transport`

Theorem 38.108 (Real stream prefix unary context closure)**.** *If every prefix-side context and every tail-side context is unary pointwise, and the two displayed context streams are pointwise* hsame *on each side, then any classified finite real-stream prefix remains classified after each endpoint is wrapped by the corresponding unary prefix and tail contexts.*

Proof. Induct on the finite prefix length. At the base observation, apply rational history classifier unary-context closure to the endpoint classifier and the four pointwise context hypotheses at index 0. At a successor index, keep the inductively transported shorter prefix and apply the same rational unary-context closure theorem to the successor endpoint classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier unary context closed`

Theorem 38.109 (Real stream prefix classifier reflexivity)**.** *For every rational-observation stream* $x : \mathbb{N} \to \mathsf{Hist}$*, if* $\mathsf{RatHistoryCarrier}(x(i))$ *for every observation index* i*, then* $\mathsf{RealStreamPrefixClassifier}(x, x, n)$ *for every prefix index* n*.*

Proof. Induct on n. At 0, the carrier witness for $x(0)$ supplies the rational history classifier for the endpoint, with hsame reflexivity providing the identity transport on both sides.

At a successor index, pair the induction hypothesis for the shorter prefix with the same rational classifier construction at the new endpoint $n + 1$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier refl`

Theorem 38.110 (Real stream prefix classifier symmetry)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, if* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$*, then* $\mathsf{RealStreamPrefixClassifier}(y, x, n)$*.*

Proof. Induct on n. At 0, apply rational-history classifier symmetry to the base endpoint classifier.

At a successor index, reverse the shorter prefix by the induction hypothesis and reverse the displayed successor endpoint by rational-history classifier symmetry. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier symm`

Theorem 38.111 (Real stream prefix classifier endpoint projection)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, if* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$*, then* $\mathsf{RatHistoryClassifier}(x(n), y(n))$*.*

Proof. Split on n. At 0, the prefix witness is exactly the selected rational-history classifier.

At a successor index, the recursive prefix witness is a pair; its right component is the rational-history classifier at the selected endpoint. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier endpoint`

Theorem 38.112 (Real stream prefix classifier base of successor)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, every nonempty classified finite prefix contains the base observation classifier:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n+1) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, 0).$$

Proof. Induct on n. At 0, the successor prefix is the pair of the base classifier and the first successor endpoint classifier, so project its left component. At a further successor, recursively descend through the left prefix component until the base observation classifier is exposed. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier base of successor`

Lemma 38.113 (Real stream prefix successor inversion)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and every prefix length* n*:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n{+}1) \Longleftrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \mathsf{RatHistoryClassifier}(x(n{+}1), y(n{+}1)).$$

Ten-field fit: traditional target finite prefix successor inversion; BEDC source RealUp prefix classifier over RatUp point classifiers; carrier rational-observation streams; classifier RealStreamPrefixClassifier*; stability inherited from hsame transport and endpoint projection; ledger prefix length and endpoint observation; core theorem this lemma; exactness the displayed iff; standard bridge finite prefix recursion; status* ProofTarget*.*

Proof. Unfold the recursive definition of $\mathsf{RealStreamPrefixClassifier}(x, y, n{+}1)$. In the forward direction, the successor prefix witness is a pair; project its left component to obtain $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ and its right component to obtain the endpoint classifier $\mathsf{RatHistoryClassifier}(x(n+1), y(n+1))$.

Conversely, assume the previous prefix witness and the endpoint rational-history classifier. Pair these two witnesses by the successor constructor of the recursive prefix display, obtaining $\mathsf{RealStreamPrefixClassifier}(x, y, n+1)$. □

Lemma 38.114 (Real stream prefix successor e1 endpoint package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, if* $\mathsf{RealStreamPrefixClassifier}(x, y, n{+}1)$ *and the successor endpoint histories have displays* $x(n+1)\mathsf{hsameE1}(a)$ *and* $y(n+1)\mathsf{hsameE1}(b)$*, then*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge a\mathsf{hsame}b.$$

Proof. The successor prefix witness splits into the predecessor prefix and the successor endpoint classifier. Apply the e1 endpoint readback theorem at index $n+1$ to the endpoint classifier and the two displayed e1 histories; this yields the unary tail witnesses and their history-sameness. Pair that endpoint package with the predecessor prefix. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier successor e1 endpoint package`

Theorem 38.115 (Real stream prefix classifier tail extension)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, m*, an already classified prefix through* n *extends across any finite tail whose endpoints are classified:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \big(\forall k,\ 1 \le k \le m \Longrightarrow \mathsf{RatHistoryClassifier}(x(n{+}k), y(n{+}k))\big) \Longrightarrow \mathsf{RealStreamPrefix}$$

Ten-field fit: traditional target finite prefix continuation from certified endpoints; BEDC source RealUp stream prefixes over RatUp history classifiers; carrier rational-observation streams; classifier RealStreamPrefixClassifier*; stability inherited from each endpoint classifier and recursive prefix pairing; ledger initial prefix length, tail length, and endpoint offsets; core theorem this tail continuation principle; exactness the displayed implication; standard bridge finite prefix recursion; status* ProofTarget*.*

Proof. Induct on m. At $m = 0$, the target prefix is the given prefix through n, since $n + 0 = n$, and no tail endpoint is required.

For the successor step, assume the principle for tail length m and suppose the prefix through n is classified together with endpoint classifiers for all offsets $1 \leq k \leq m + 1$. Restrict the endpoint assumption to offsets $1 \leq k \leq m$ and apply the induction hypothesis. This gives $\mathsf{RealStreamPrefixClassifier}(x, y, n + m)$.

It remains to add the final endpoint. Taking $k = m + 1$ in the endpoint assumption gives $\mathsf{RatHistoryClassifier}(x(n{+}m{+}1), y(n{+}m{+}1))$ after the usual successor arithmetic. Apply the reverse direction of Definition 38.113 to the classified prefix through $n + m$ and this endpoint classifier, obtaining the classified prefix through $n + m + 1$, which is the prefix through $n + (m + 1)$. □

Theorem 38.116 (Real stream prefix tail extension with unary anchor)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a classified prefix through n extends across a classified finite tail of length m, and the left endpoint at the new boundary retains its unary anchor:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n) \Longrightarrow \Big(\forall k < m, \mathsf{RatHistoryClassifier}(x(n{+}k{+}1), y(n{+}k{+}1))\Big) \Longrightarrow \Big(\forall i,\ \mathsf{UnaryHistory}$$

Proof. Induct on m. The empty tail is the given prefix through n, with the unary side condition supplying the endpoint anchor at $x(n)$. For a successor tail, restrict the pointwise tail classifier to the shorter tail, apply the induction hypothesis to obtain the prefix through $n{+}m$, and append the final rational endpoint classifier at $x(n + m + 1), y(n + m + 1)$. The unary side condition supplies the new left endpoint anchor. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier tail extension with unary anchor`

Theorem 38.117 (Real stream prefix tail endpoint context projection)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a classified prefix through n and a classified finite tail of length m determine the endpoint classifier at $n + m$. If that endpoint is then continued through matching unary prefix and tail contexts on both sides, the two displayed continuation results remain rational-history classified.*

Proof. First apply the finite tail principle to obtain $\mathsf{RealStreamPrefixClassifier}(x, y, n{+}m)$ together with the unary anchor at $x(n + m)$. Project the endpoint classifier at $n + m$. Rational unary-denominator context closure places the matching prefix and tail contexts around the two endpoint histories, and the continuation witnesses identify those normal forms with the displayed outputs. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier tail endpoint context projection`

Theorem 38.118 (Real stream prefix classifier truncation)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, any prefix classifier through $m{+}n$ restricts to the prefix classifier through n:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m + n) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, n).$$

Proof. Induct on m. At $m = 0$, the source prefix is already the target prefix, since $0 + n = n$.

At a successor $m + 1$, the recursive definition of $\mathsf{RealStreamPrefixClassifier}(x, y, (m + 1) + n)$ exposes its left component, which is the classified prefix through $m + n$. Apply the induction hypothesis to this exposed prefix to obtain the classified prefix through n. □

Theorem 38.119 (Real stream prefix internal endpoint coverage)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, every classified prefix through $m+n$ exposes the retained endpoint row at n:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n) \Longrightarrow \mathsf{RatHistoryClassifier}(x(n), y(n)).$$

Ten-field fit: traditional target internal finite-prefix observation coverage; BEDC source RealUp prefix classifier plus RatUp endpoint classifier; carrier rational-observation streams; classifier RealStreamPrefixClassifier *stability inherited from truncation and endpoint projection; ledger prefix length, truncated endpoint, and observation index; core theorem this block; exactness classified prefix exposes each retained endpoint row; standard bridge finite rational prefix observation; status* ProofTarget*.*

Proof. Apply Definition 38.118 to the source prefix. This gives $\mathsf{RealStreamPrefixClassifier}(x, y, n)$.

Apply Definition 38.111 to the truncated prefix. The selected endpoint classifier is $\mathsf{RatHistoryClassifier}(x(n), y(n)$ □

Theorem 38.120 (Real stream prefix truncated unary context closure)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if a prefix through $m+n$ is classified, then after descending to the prefix through n, matching unary prefix and tail contexts may be placed around every displayed endpoint:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x', y', n),$$

where $x'(i) = \mathsf{append}(p_X(i), \mathsf{append}(x(i), t_X(i)))$ and $y'(i) = \mathsf{append}(p_Y(i), \mathsf{append}(y(i), t_Y(i)))$, with $p_X(i)$ and $t_X(i)$ unary and with $p_X(i)\mathsf{hsame}p_Y(i)$ and $t_X(i)\mathsf{hsame}t_Y(i)$ for every index i.

Proof. Induct on m to restrict the source prefix from $m+n$ to n. The base case is definitional, and the successor case projects the left component of the recursive prefix witness. Apply the stream-prefix unary-context closure theorem to the truncated prefix and the pointwise matching unary contexts. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated unary context closed`

Theorem 38.121 (Real stream prefix truncated e1 pair readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if the finite prefix through $m+n$ is classified and the selected endpoint at n is displayed as one-headed on both sides,*

$$\mathsf{hsame}(x(n), \mathsf{E1}(d)) \quad \textit{and} \quad \mathsf{hsame}(y(n), \mathsf{E1}(e)),$$

then both tails are unary and internally same:

$$\mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(e) \wedge \mathsf{hsame}(d, e).$$

Proof. First descend the recursive prefix classifier from $m+n$ to n by induction on m. The base case is definitional. In the successor case, the prefix witness at $(m+1)+n$ exposes the witness at $m+n$ as its left component. After the classified prefix at n is obtained, apply the e1 pair readback theorem at the selected endpoint. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated e1 pair readback`

Theorem 38.122 (Real stream prefix truncated continuation endpoint context closure)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a classified prefix through $m+n$ supplies the rational classifier at endpoint n. If the selected left and right endpoints are then continued through unary prefix and tail contexts, with matching* hsame *contexts on the two sides, the two displayed continuation results remain rational-history classified.*

Proof. Induct on m to descend the recursive prefix classifier from $m+n$ to n. Project the rational-history classifier at endpoint n. Apply rational-history classifier unary-denominator context closure to the selected endpoint and the two unary context pairs. The continuation witnesses identify the displayed outputs with the corresponding appended contexts, and associativity of append supplies the final endpoint transports. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated cont endpoint context closed`

Lemma 38.123 (Real stream prefix previous classifier with unary anchor)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if every left observation $x(i)$ is unary, then every successor finite-prefix classifier contains both the previous prefix classifier and the unary anchor at the previous left endpoint:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n+1) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \mathsf{UnaryHistory}(x(n)).$$

Proof. The recursive successor prefix is a pair whose left component is the previous prefix classifier. The unary side condition supplies $\mathsf{UnaryHistory}(x(n))$ at the same previous endpoint. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier previous with unary`

Theorem 38.124 (Real stream prefix add-left previous classifier with unary anchor)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if every left observation $x(i)$ is unary, then any prefix classified through $m+n$ descends to the prefix classified through n, while retaining the unary anchor at $x(n)$:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \mathsf{UnaryHistory}(x(n)).$$

Proof. Induct on m. At $m = 0$, the displayed prefix is the target prefix and the unary side condition supplies the endpoint anchor. At a successor $m+1$, apply Definition 38.123 to peel one recursive prefix layer, then apply the induction hypothesis to the exposed prefix through $m+n$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier add left previous with unary`

Lemma 38.125 (Real stream prefix add-left previous classifier with right unary anchor)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if every right observation $y(i)$ is unary, then any prefix classified through $m+n$ descends to the prefix classified through n, while retaining the unary anchor at $y(n)$:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n) \Longrightarrow \mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge \mathsf{UnaryHistory}(y(n)).$$

Proof. Apply symmetry of the finite-prefix classifier to read the prefix as a classifier from y to x. Then apply the add-left previous theorem with unary anchors on the left stream y. The recovered prefix is finally symmetrized back to a classifier from x to y, while the unary endpoint anchor remains $y(n)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier add left previous with right unary`

Theorem 38.126 (Real stream prefix tail endpoint projection)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, m, if the prefix through n is classified and every endpoint along the finite tail of length m is rational-history classified, then the selected tail endpoint is rational-history classified:*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge (\forall k,\ 1 \le k \le m \Longrightarrow \mathsf{RatHistoryClassifier}(x(n+k), y(n+k))) \Longrightarrow \mathsf{RatHistoryClassif}$$

Ten-field fit: traditional target finite RealUp prefix endpoint readback; BEDC source RealUp stream prefixes over RatUp point classifiers; carrier rational-observation streams; classifier RealStreamPrefixClassifier*; stability by tail extension and endpoint projection; ledger initial prefix length, tail length, and selected endpoint; core theorem this projection; exactness selected endpoint classification; standard bridge RegSeqRat finite observation readback; status* ProofTarget*.*

Proof. Apply Definition 38.115 to the classified prefix through n and the finite tail endpoint hypothesis. This gives $\mathsf{RealStreamPrefixClassifier}(x, y, n + m)$.

Now apply Definition 38.111 at the endpoint index $n + m$. The resulting projection is exactly $\mathsf{RatHistoryClassifier}(x(n + m), y(n + m))$. □

Theorem 38.127 (Real stream prefix truncated endpoint common e-one tail readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, if $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ holds, then the endpoint at n has a common e-one tail readback:*

$$\exists a, b,\ \mathsf{hsame}(x(n), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y(n), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Truncate the displayed prefix classifier to the selected endpoint n and use the denominator package there. Its positive unary denominator rows give unary endpoint witnesses and separate both selected endpoints from e, so unary nonempty readback exposes e-one tails a and b. The truncated e-one pair readback theorem then returns the two unary tail rows and their hsame comparison. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated endpoint common e1 tail readback`

Theorem 38.128 (Real stream prefix probe endpoint determinacy)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ be a finite certified* ProbeBundle *window and let n be a unary observation with $\mathsf{InBundle}(n, \mathcal{P})$. Suppose the window supplies $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ and, for every displayed tail row selected by $\mathcal{P}$ after n, the finite row $\mathsf{RatHistoryClassifier}(x(n+k), y(n+k))$ is present. Then every consumer endpoint selected from $\mathcal{P}$ has a unique* RatHistoryClassifier *readback, and its endpoint admits the same common e-one tail witness after* hsame *transport:*

$$\mathsf{InBundle}(n, \mathcal{P}) \Longrightarrow \forall j,\ \mathsf{InBundle}(j, \mathcal{P}) \Longrightarrow \exists a, b,\ \mathsf{hsame}(x(j), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHis}$$

The statement reads only the displayed finite rows of $\mathcal{P}$; no observation outside the certified probe window is used.

Proof. Restrict the finite ProbeBundle to the selected prefix through n and to the displayed consumer endpoint j. The membership rows $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{InBundle}(j, \mathcal{P})$ are exactly the evidence that these two observations lie inside the certified window.

Apply Definition 38.853 to the selected prefix and the finite tail rows displayed by $\mathcal{P}$. This gives the endpoint RatHistoryClassifier readback for the consumer endpoint, and uniqueness follows because the readback is selected from the same finite bundle row rather than from an external observation.

Finally apply Definition 38.854 to the resulting classified prefix. The hsame transport keeps the two endpoint histories in the same rational-observation class, so the e-one tail witnesses are the common tail witnesses recorded by the finite probe window. □

Theorem 38.129 (Real terminal seal tail-section coherence)**.** *Let $x, y : \mathbb{N} \to \mathsf{Hist}$ be rational-observation streams, let $\mathcal{P}$ be a finite certified* ProbeBundle *endpoint window, and let $\mathcal{T}$ be the same $\mathsf{StreamName}^{\uparrow}$ tail-section package consumed by a finite $\mathsf{Real}^{\uparrow}$ stream prefix handoff. Suppose*

two finite endpoint reads pass through the same prefix row $\mathsf{RealStreamPrefixClassifier}(x, y, n)$*, the same displayed* $\mathsf{InBundle}$ *endpoint rows of* $\mathcal{P}$*, and the same tail-section package* $\mathcal{T}$*. Then the two endpoint reads determine the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route up to the displayed* hsame *transports.*

The seal route is the finite route through the prefix endpoint projection, truncated common e-one tail readback, probe endpoint determinacy, and the certified $\mathsf{StreamName}^{\uparrow}$ *tail-section induction bridge. It does not export quotient stream equality, host sequence equality, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. First apply endpoint projection, Definition 38.853. Both reads use the same prefix row and the same finite tail rows selected by $\mathcal{P}$, so the endpoint $\mathsf{RatHistoryClassifier}$ readback is taken from the same finite endpoint surface.

Next apply truncated endpoint common e-one tail readback, Definition 38.854. The endpoint histories therefore expose common e-one tail witnesses, and the displayed hsame rows transport those witnesses along the same prefix package.

Now use probe endpoint determinacy, Definition 38.855. It identifies the endpoint readback supplied by $\mathcal{P}$ as the unique finite readback for that endpoint inside the certified probe window.

Finally apply the StreamName tail-section induction bridge Definition 126.694. Since both reads consume the same $\mathsf{StreamName}^{\uparrow}$ tail-section package, the terminal seal route is carried over the same finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows. No remaining coordinate can distinguish two terminal $\mathsf{Real}^{\uparrow}$ seal routes. □

Theorem 38.130 (Real stream prefix two-probe endpoint coherence)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, let* $\mathcal{P}$ *and* $\mathcal{Q}$ *be finite certified* $\mathsf{ProbeBundle}$ *windows. Suppose the same unary observation* j *satisfies* $\mathsf{InBundle}(j, \mathcal{P})$ *and* $\mathsf{InBundle}(j, \mathcal{Q})$*, and suppose both windows supply the* $\mathsf{RealStreamPrefixClassifier}$ *row and the finite* $\mathsf{RatHistoryClassifier}$ *tail rows required by Definition 38.855 for the selected endpoint* j*. Then the endpoint readbacks* $\mathsf{RatHistoryClassifier}(x(j), y(j))$ *obtained through* $\mathcal{P}$ *and through* $\mathcal{Q}$ *are the same up to* hsame*, and their common e-one tail witnesses agree up to* hsame *transport:*

$$
\begin{aligned}
&\mathsf{InBundle}(j, \mathcal{P}) \wedge \mathsf{InBundle}(j, \mathcal{Q}) \\
&\quad \textit{and the two displayed endpoint-determinacy packets at } j \\
&\quad \textit{determine histories } a_{\mathcal{P}}, b_{\mathcal{P}}, a_{\mathcal{Q}}, b_{\mathcal{Q}} \textit{ with} \\
&\quad \mathsf{hsame}(a_{\mathcal{P}}, a_{\mathcal{Q}}) \wedge \mathsf{hsame}(b_{\mathcal{P}}, b_{\mathcal{Q}}) \wedge \mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{P}})) \wedge \mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{Q}})) \\
&\quad \wedge\ \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{P}})) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{Q}})).
\end{aligned}
$$

Ten-field fit: traditional target finite RealUp two-window endpoint coherence; BEDC source RealUp stream prefixes over RatUp point classifiers; carrier the two certified $\mathsf{ProbeBundle}$ *windows over the same streams; classifier* $\mathsf{RealStreamPrefixClassifier}$ *at the selected unary observation; stability by* hsame *transitivity across the two endpoint readbacks; ledger the two window memberships, prefix classifiers, tail rows, endpoint row, and common e-one witnesses; core theorem this two-probe coherence statement; exactness the fixed endpoint row* j*; standard bridge StreamName and RegSeqRat finite observation readback; status* $\mathsf{ProofTarget}$*. No observation outside the two certified probe windows is used.*

Proof. Apply Definition 38.855 to $\mathcal{P}$ at the selected observation j. The window membership row and the displayed classifier and tail rows return the endpoint $\mathsf{RatHistoryClassifier}(x(j), y(j))$ together with common e-one tail witnesses $a_{\mathcal{P}}$ and $b_{\mathcal{P}}$.

Apply the same theorem to $\mathcal{Q}$ at the same selected observation j. This gives the endpoint readback through the second finite window and common e-one tail witnesses $a_{\mathcal{Q}}$ and $b_{\mathcal{Q}}$.

Both applications read the fixed endpoint row j of the same streams x and y. Transitivity of hsame identifies the two rational endpoint readbacks, and the same transport identifies the e-one tail witnesses. Since every row used in the argument is selected from either $\mathcal{P}$ or $\mathcal{Q}$, no external observation enters the comparison. □

Theorem 38.131 (Real stream prefix probe-union endpoint readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows, and let $\mathcal{U}$ be their synchronized union window certified by Definition 126.242. Suppose each source window supplies the* RealStreamPrefixClassifier *rows and finite* RatHistoryClassifier *tail rows required by Definition 38.855. Then every unary endpoint j selected by either source window has the same* $\mathsf{RatHistoryClassifier}(x(j), y(j))$ *readback through the union window $\mathcal{U}$, and the common e-one tail witnesses displayed by the source window and by $\mathcal{U}$ agree by* hsame *transport.*

Equivalently, if $\mathsf{InBundle}(j, \mathcal{P}) \vee \mathsf{InBundle}(j, \mathcal{Q})$*, then* $\mathsf{InBundle}(j, \mathcal{U})$ *and the union read supplies histories $a_{\mathcal{U}}, b_{\mathcal{U}}$ with*

$$\mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{U}})) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{U}})) \wedge \mathsf{UnaryHistory}(a_{\mathcal{U}}) \wedge \mathsf{UnaryHistory}(b_{\mathcal{U}}).$$

If j lies in both $\mathcal{P}$ and $\mathcal{Q}$, these witnesses are hsame*-identified with the two source-window witnesses of Definition 38.857; if j lies in only one source window, the union read is the unique displayed readback of that source window.*

Ten-field fit: traditional target finite RealUp union endpoint readback; BEDC source RealUp stream prefixes with RatUp point classifiers; carrier the certified ProbeBundle *union membership rows for $\mathcal{P}, \mathcal{Q}, \mathcal{U}$; classifier* RealStreamPrefixClassifier *at the displayed unary prefix; stability by* hsame *transport of the e-one tails; ledger the finite endpoint ledger, source memberships, synchronized union membership, prefix classifiers, and tail rows; core theorem this probe-union readback statement; exactness the fixed finite endpoint row j; bridge data the StreamName synchronized-window union coverage and RegSeqRat finite observation readback; status* ProofTarget*. No quotient-real equality, completeness theorem, or host real comparison is used.*

Proof. Apply Definition 126.242 to the two certified probe windows. It places every endpoint selected by $\mathcal{P}$ or by $\mathcal{Q}$ into the synchronized union window $\mathcal{U}$ and records the finite union membership row used by the later readback.

When the selected endpoint lies only in $\mathcal{P}$, apply Definition 38.855 to $\mathcal{P}$. The union membership supplied by the synchronized carrier reads the same finite endpoint row, so the union readback is the displayed $\mathcal{P}$ readback transported along hsame.

The case where the endpoint lies only in $\mathcal{Q}$ is identical, with $\mathcal{Q}$ supplying the source readback and the union carrier supplying the same endpoint row in $\mathcal{U}$.

If the endpoint lies in both source windows, first apply endpoint determinacy in each window, then apply Definition 38.857. The coherence row identifies the two rational endpoint readbacks and transports the common e-one tail witnesses through hsame, so the union read has the same displayed tails.

All rows consumed by the argument are finite rows of the two source probe windows or of their synchronized union. Hence the endpoint readback remains a RegSeqRat-style finite observation read and never appeals to a quotient real, completeness principle, or host equality. □

Theorem 38.132 (Real stream prefix probe-union terminal four-face factorization)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows and let $\mathcal{U}$ be their synchronized union window. Suppose the two source windows supply the* RealStreamPrefixClassifier

rows and finite RatHistoryClassifier *tail rows required by Definition 38.858. Then every endpoint readback selected from $\mathcal{U}$ factors through exactly one terminal four-face packet over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, R

The packet has precisely the displayed rows

BHist-*indexed finite probe histories*, hsame-*transport for endpoint tails*, Cont-*continuity rows*, Pkg-*package rows for the synchronized union window*, NameCert-*certification rows for the four visible faces.*

Its $\mathsf{StreamName}^{\uparrow}$ *face is the synchronized carrier $\mathcal{U}$, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the common finite scale window read from the same endpoint ledger, its* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the regular sequence readback, and its* $\mathsf{Real}^{\uparrow}$ *face is the local endpoint seal supplied by the union readback. There is no second terminal packet with the same displayed rows.*

The factorization does not use quotient-real equality, selected limit values, ambient completeness of $\mathsf{Real}^{\uparrow}$*, countable choice, or standard bridge data.*

Proof. First apply Definition 38.858. The synchronized union window supplies the endpoint RatHistoryClassifier readback and the common e-one tail witnesses, all selected from the finite rows of $\mathcal{P}, \mathcal{Q}, \mathcal{U}$.

Next apply Definition 126.242 and Definition 126.558. These two results identify the $\mathsf{StreamName}^{\uparrow}$ carrier row with a terminal finite-request section over the same synchronized window; no endpoint outside $\mathcal{U}$ is introduced.

Now apply Definition 301.239. The RegSeqRat readback and the local Real endpoint seal are thereby read as two faces of a single terminal packet whose support rows are exactly the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 291.83. It supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ common-scale face and proves that the four visible faces form one pullback packet. Terminality gives uniqueness of the packet with those rows, while the construction never introduces quotient-real equality, selected limits, ambient completeness, countable choice, or standard bridge data. □

Theorem 38.133 (Real stream prefix probe-union terminal four-face support minimality)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, finite certified probe windows $\mathcal{P}, \mathcal{Q}$, and their synchronized union $\mathcal{U}$, every accepted terminal packet produced by Definition 38.859 is support-minimal. Deleting or replacing any one of the four faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, or deleting or replacing any displayed* BHist, hsame, Cont, Pkg, *or* NameCert *support row, destroys the accepted probe-union terminal read.*

The missing face or support row cannot be supplied by quotient-real equality, selected limit values, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, metric-completeness rows, aggregate status rows, or standard bridge data. These coordinates remain outside the local probe-union packet and do not substitute for the finite BEDC support rows used by the terminal read.*

Proof. First apply Definition 38.859. It identifies the accepted probe-union read with a unique terminal packet whose visible faces are exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ and whose support is the displayed BHist, hsame, Cont, Pkg, and NameCert row family.

For the finite-window carrier face, apply Definition 126.653 to the synchronized StreamName window underlying $\mathcal{U}$. That result rules out deletion or replacement of the StreamName face and of the finite probe-history rows that certify membership in the accepted window.

For the dyadic request and regular tail faces, apply Definition 301.177. The common finite scale window and the RegSeqRat tail-budget rows are read from the same endpoint ledger, so removing either face, or any of the hsame, Cont, Pkg, or NameCert rows it consumes, leaves no accepted regular tail read for the probe-union endpoint.

Finally apply Definition 38.868 together with Definition 38.741. These pull the Real-side endpoint seal back to the same four-face terminal packet and rule out substitution by host-real equality, selected limits, completeness coordinates, choice principles, aggregate status rows, or bridge evidence. The two-axis roadmap discipline keeps those excluded coordinates separate from the local BEDC support rows, so none of them can repair a missing face or row. □

Theorem 38.134 (Real stream prefix probe-union support-row exchange invariance)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows, and let $\mathcal{U}$ be their synchronized union window. Consider any two presentations of the terminal packet supplied by Definition 38.859. Suppose both presentations preserve the same four visible faces* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *projections.*

Then exchanging the presentation order of those already-displayed support rows does not produce a second terminal packet. The two presentations determine the same terminal packet over $\mathcal{U}$, and for every selected endpoint j they determine the same finite endpoint readback $\mathsf{RatHistoryClassifier}(x(j), y(j))$ *with the same* hsame*-transported common e-one tail witnesses.*

This invariance is local to the displayed BEDC support rows. It is not witnessed by quotient-real equality, selected limit values, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, aggregate status rows, or bridge data, and none of those coordinates can distinguish two presentation orders of the same terminal probe-union packet.*

Proof. First apply Definition 38.858. Both presentations read each selected endpoint from the same synchronized union window $\mathcal{U}$ and the same finite endpoint ledger, so the rational endpoint row and the common e-one tail witnesses agree by the displayed hsame transport.

Next apply Definition 38.859. The two presentations expose the same four faces $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{R}$ and the same BHist, hsame, Cont, Pkg, and NameCert projections. Terminality therefore identifies them as one terminal packet over the synchronized union carrier rather than as two packets distinguished by row order.

Finally apply Definition 38.860. Support minimality says that the accepted terminal read is carried exactly by the displayed faces and support rows. Hence quotient-real equality, selected limits, ambient completeness, countable choice, aggregate status rows, and bridge data cannot replace one of those rows or separate two orderings of the same row family. The endpoint readback equality from the first paragraph is therefore the endpoint readback of the unique terminal packet. □

Theorem 38.135 (Real stream finite-prefix exactness)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$,*

$$\mathsf{RealStreamClassifier}(x, y) \iff \forall n,\ \mathsf{RealStreamPrefixClassifier}(x, y, n).$$

Proof. For the forward direction, apply Definition 38.106. The pointwise stream classifier supplies exactly the rational-history classifier needed at each endpoint of the recursive prefix construction.

For the reverse direction, assume every finite prefix is classified and fix an observation index n. Apply the hypothesis at n, then use Definition 38.111 to recover $\mathsf{RatHistoryClassifier}(x(n), y(n))$. Since n was arbitrary, these endpoint classifiers are precisely $\mathsf{RealStreamClassifier}(x, y)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier finite prefix exactness`

Theorem 38.136 (Real stream classifier symmetry)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, if* $\mathsf{RealStreamClassifier}(x, y)$*, then* $\mathsf{RealStreamClassifier}(y, x)$*.*

Proof. Fix an observation index. The stream classifier supplies the rational-history classifier at that index, and rational-history classifier symmetry reverses it. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier symm`

Theorem 38.137 (Real stream classifier transitivity)**.** *For rational-observation streams* $x, y, z : \mathbb{N} \to \mathsf{Hist}$*, if* $\mathsf{RealStreamClassifier}(x, y)$ *and* $\mathsf{RealStreamClassifier}(y, z)$*, then* $\mathsf{RealStreamClassifier}(x, z)$*.*

Proof. Fix an observation index. The two stream classifiers give the two rational-history classifier witnesses at that index, and rational-history classifier transitivity composes them. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier trans`

Theorem 38.138 (Real stream classifier certificate fields)**.** *For rational-observation streams* $x, y, z : \mathbb{N} \to \mathsf{Hist}$*, the pointwise stream classifier supplies the following stability certificate fields:*

1. *if* $\forall n,\ \mathsf{RatHistoryCarrier}(x(n))$*, then* $\mathsf{RealStreamClassifier}(x, x)$*;*
2. *if* $\mathsf{RealStreamClassifier}(x, y)$*, then for every* n *both* $\mathsf{RatHistoryCarrier}(x(n))$ *and* $\mathsf{RatHistoryCarrier}(y(n))$*;*
3. *if* $\mathsf{RealStreamClassifier}(x, y)$*, then* $\mathsf{RealStreamClassifier}(y, x)$*;*
4. *if* $\mathsf{RealStreamClassifier}(x, y)$ *and* $\mathsf{RealStreamClassifier}(y, z)$*, then* $\mathsf{RealStreamClassifier}(x, z)$*.*

Proof. For reflexivity, fix the stream x and its pointwise rational-history carrier witness. Apply Definition 38.109 at every finite prefix. The reverse direction of Definition 38.135 then repacks these prefix witnesses as $\mathsf{RealStreamClassifier}(x, x)$.

For endpoint carrier readback, unfold the pointwise stream classifier and fix an index n. The classifier at n is a $\mathsf{RatHistoryClassifier}(x(n), y(n))$ witness, and unpacking the rational-history classifier gives both endpoint carrier facts.

The symmetry field is exactly Definition 38.136. The transitivity field is exactly Definition 38.137. These four clauses are therefore available as the certificate-level stability package for the stream classifier. □

Theorem 38.139 (Real stream classifier hsame transport)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, if* $\forall n,\ \mathsf{hsame}(x(n), x'(n))$ *and* $\forall n,\ \mathsf{hsame}(y(n), y'(n))$*, then*

$$\mathsf{RealStreamClassifier}(x, y) \iff \mathsf{RealStreamClassifier}(x', y').$$

Proof. For the forward direction, assume $\mathsf{RealStreamClassifier}(x, y)$. By the forward direction of Definition 38.135, every finite prefix $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ is classified.

For each prefix index n, apply Definition 38.107 with the two pointwise hsame hypotheses. This transports the prefix classifier from (x, y) to (x', y') at the same index.

Since the transported prefix witness is available for every n, the reverse direction of Definition 38.135 reassembles these finite-prefix witnesses into $\mathsf{RealStreamClassifier}(x', y')$.

For the reverse direction, assume $\mathsf{RealStreamClassifier}(x', y')$. The same finite-prefix exactness theorem gives $\mathsf{RealStreamPrefixClassifier}(x', y', n)$ for every n.

For each prefix index n, use the symmetry of the pointwise hsame witnesses to obtain $\mathsf{hsame}(x'(n), x(n))$ and $\mathsf{hsame}(y'(n), y(n))$. Applying Definition 38.107 with these symmetric witnesses transports the prefix classifier from (x', y') back to (x, y).

The reverse direction of Definition 38.135 then reassembles the transported finite-prefix witnesses into $\mathsf{RealStreamClassifier}(x, y)$. □

Theorem 38.140 (Real stream transport with endpoint readback)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, assume* $\forall n,\ \mathsf{hsame}(x(n), x'(n))$*, assume* $\forall n,\ \mathsf{hsame}(y(n), y'(n))$*, and assume* $\mathsf{RealStreamClassifier}(x, y)$*. Then the transported streams carry the following readback package:*

1. $\mathsf{RealStreamClassifier}(x', y')$*;*
2. *for every* n*,* $\mathsf{RealStreamPrefixClassifier}(x', y', n)$*;*
3. *for every* n*,* $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$*.*

Proof. First take the forward implication from Definition 38.139 and apply it to the two pointwise hsame hypotheses and the original stream classifier. This gives $\mathsf{RealStreamClassifier}(x', y')$.

Next use the forward direction of Definition 38.135 on the transported stream classifier. The result is $\mathsf{RealStreamPrefixClassifier}(x', y', n)$ for every finite prefix index n.

Finally fix an index n and apply Definition 38.111 to the transported prefix witness at that index. This recovers the endpoint $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$, completing the three displayed fields of the package. □

Theorem 38.141 (Real stream transported selected zero-endpoint exclusion)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$ *and a selected index* n*, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y),$$

then neither transported selected endpoint can be displayed as a zero-extension endpoint:

$$(\mathsf{hsame}(x'(n), \mathsf{E0}(z_x)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_y)) \to \mathsf{False}).$$

Proof. Transport the pointwise rational-history classifier at index n along the two streamwise hsame hypotheses. The transported classifier exposes positive unary denominator witnesses at $x'(n)$ and $y'(n)$. If either endpoint is displayed as $\mathsf{E0}$, transport the corresponding positive denominator witness to that zero-extension endpoint and apply the rational positive-denominator zero-endpoint contradiction. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transport selected e0 endpoint absurd`

Theorem 38.142 (Real stream classifier hsame one-tail pair readback)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, assume* $\forall n,\ \mathsf{hsame}(x(n), x'(n))$ *and* $\forall n,\ \mathsf{hsame}(y(n), y'(n))$*. If* $\mathsf{RealStreamClassifier}(x, y)$ *and the transported endpoints satisfy* $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ *and* $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$*, then*

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Evaluate the pointwise stream classifier at the selected index n. Transport the resulting rational-history classifier along the two pointwise hsame maps, then transport again to the displayed one-extension endpoints. Rational one-tail classifier exactness exposes the two unary tails and their internal sameness. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier hsame e1 pair readback`

Theorem 38.143 (Real stream classifier selected continuation positive denominators)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, a shared unary suffix* q*, and continuation outputs* xq, yq*, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

then

$$\mathsf{PositiveUnaryDenominator}(xq) \wedge \mathsf{PositiveUnaryDenominator}(yq).$$

Proof. At the selected index, the stream classifier gives $\mathsf{RatHistoryClassifier}(x(n), y(n))$. Close this rational classifier under the common unary suffix q, using reflexive suffix sameness. The two continuation displays identify the closed endpoints with xq and yq. Transport the closed classifier across those displays and read the two positive denominators from the transported rational-history classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation positive denominators`

Theorem 38.144 (Real stream classifier selected continuation zero-endpoint exclusion)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, a shared unary suffix q, continuation outputs xq, yq, and endpoint tails z_x, z_y, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

then neither selected continuation output can be displayed as a zero-headed endpoint:

$$(\mathsf{hsame}(xq, \mathsf{E0}(z_x)) \to \mathsf{False}) \wedge (\mathsf{hsame}(yq, \mathsf{E0}(z_y)) \to \mathsf{False}).$$

Proof. The selected continuation positive-denominator theorem gives positive unary denominator witnesses for xq and yq. Transport either witness along a displayed zero-headed endpoint comparison and apply positive denominator zero-endpoint absurdity. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation e0 endpoint absurd`

Theorem 38.145 (Real stream classifier selected continuation endpoint boundary package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, a shared unary suffix q, continuation outputs xq, yq, and endpoint tails $z_x, z_y : \mathsf{Hist}$, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

then the selected continuation endpoints carry the boundary package

$$\begin{aligned}
&\mathsf{PositiveUnaryDenominator}(xq) \wedge \mathsf{PositiveUnaryDenominator}(yq) \wedge \\
&\mathsf{UnaryHistory}(xq) \wedge \mathsf{UnaryHistory}(yq) \wedge \\
&(\mathsf{hsame}(xq, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(yq, \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(xq, \mathsf{E0}(z_x)) \to \mathsf{False}) \wedge (\mathsf{hsame}(yq, \mathsf{E0}(z_y)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.143 to the selected stream classifier, the shared unary suffix, and the two continuation displays. This gives positive unary denominator witnesses for xq and yq.

Apply Definition 37.12 separately to the two denominator witnesses. These two applications give $\mathsf{UnaryHistory}(xq)$, $\mathsf{UnaryHistory}(yq)$, and the two empty-endpoint exclusions.

Apply Definition 38.144 to the same selected continuation data. This gives the two zero-headed endpoint exclusions. Package the denominator, unary, empty-endpoint, and zero-endpoint rows in the displayed order. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation endpoint boundary package`

Theorem 38.146 (Real stream classifier selected continuation full shape package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, a shared unary suffix q, continuation outputs xq, yq, and histories $a, b, z_x, z_y : \mathsf{Hist}$, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

and

$$\mathsf{hsame}(xq, \mathsf{E1}(a)), \quad \mathsf{hsame}(yq, \mathsf{E1}(b)),$$

then the selected continuation endpoints carry the full displayed shape package:

$$\begin{aligned}&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge \\ &\mathsf{PositiveUnaryDenominator}(xq) \wedge \mathsf{PositiveUnaryDenominator}(yq) \wedge \mathsf{UnaryHistory}(xq) \wedge \mathsf{UnaryHistory}(yq) \wedge \\ &(\mathsf{hsame}(xq, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(yq, \mathsf{e}) \to \mathsf{False}) \wedge \\ &(\mathsf{hsame}(xq, \mathsf{E0}(z_x)) \to \mathsf{False}) \wedge (\mathsf{hsame}(yq, \mathsf{E0}(z_y)) \to \mathsf{False}).\end{aligned}$$

Proof. Apply Definition 38.637 to the RealUp stream classifier, the shared unary suffix, the two continuation outputs, and the two displayed e-one tails. This reads back $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply Definition 38.145 to the same selected continuation data. This supplies both positive denominator rows, both unary endpoint rows, both empty-endpoint exclusions, and both zero-headed endpoint exclusions.

Package the e-one tail readback, the positive denominator witnesses, the unary endpoint rows, the empty-endpoint exclusions, and the zero-endpoint exclusions. The result is exactly the displayed full shape package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation full shape package`

Fit: traditional target selected continuation endpoint shape for RealUp streams; BEDC source RealUp stream classifier plus RatUp denominator readback plus Cont; carrier rational-observation streams with a selected continuation; classifier RealStreamClassifier and RatHistoryClassifier-derived denominator data; stability selected continuation closure and hsame readback; ledger selected index, shared suffix, continuation outputs, and endpoint display tests; core theorem Definition 38.146; exactness the displayed conjunction; standard bridge finite selected rational endpoint shape inside RealUp; status ProofTarget.

Theorem 38.147 (Real stream prefix classifier transitivity)**.** *For rational-observation streams $x, y, z : \mathbb{N} \to \mathsf{Hist}$, if the finite prefix classifier relates x to y through index n and relates y to z through the same index, then it relates x to z through index n.*

Proof. Induct on the prefix index. At zero, use rational history classifier transitivity. At a successor index, combine the induction hypothesis for the shorter prefix with rational history classifier transitivity at the new endpoint. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier trans`

Theorem 38.148 (Real stream prefix one-tail pair readback)**.** *If a classified finite rational-observation prefix displays its selected left and right endpoints as one-extension histories, then the displayed tails are unary and internally the same.*

Proof. Split on the selected observation index. At zero, use the base rational-history classifier; at a successor, use the right endpoint of the prefix classifier. Transport the selected classifier to the displayed one-extension endpoints, then apply rational one-tail classifier exactness. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier e1 pair readback`

Theorem 38.149 (Real stream prefix classifier hsame one-tail pair readback)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, assume* $\forall n,\ \mathsf{hsame}(x(n), x'(n))$ *and* $\forall n,\ \mathsf{hsame}(y(n), y'(n))$*. If* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ *and the transported endpoints satisfy* $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ *and* $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$*, then*

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. First transport the finite prefix classifier from (x, y) to (x', y') using the pointwise hsame hypotheses. The transported prefix has displayed one-extension endpoints at index n, so the one-tail pair readback theorem returns the two unary tail witnesses and $\mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier hsame e1 pair readback`

Theorem 38.150 (Real stream prefix certificate boundary)**.** *For rational-observation streams transported by pointwise* hsame*, the finite prefix classifier supplies a closed BEDC certificate boundary. The same prefix surface gives transitive prefix comparison, one-extension endpoint unary tails, and transported tail* hsame *readback.*

This boundary uses only the displayed finite prefix surface. It does not read host index equality, quotient structure, completion data, $\mathsf{Real}^{\uparrow}$ *completeness, metric structure, order structure, or any standard bridge row.*

Proof. Apply Definition 38.147 to close the comparison surface under transitive prefix composition at the displayed observation index.

Apply Definition 38.148 to read the selected left and right one-extension endpoints back to unary tails and their internal hsame comparison.

Finally apply Definition 38.149. The pointwise hsame hypotheses transport the same finite prefix certificate to the displayed endpoint streams, so the certificate boundary survives the transport without adding any host-side equality, analytic completion, metric, order, or bridge datum. □

Definition 38.151 (Real unary offset bound)**.** *For histories* $k, w : \mathsf{Hist}$*, the unary offset bound is the BEDC continuation witness that* k *is a certified unary observation lying below the certified unary bound* w*:*

$$\mathsf{UnaryOffsetLe}(k, w) \iff \mathsf{UnaryHistory}(k) \wedge \exists \tau : \mathsf{Hist},\ \mathsf{UnaryHistory}(\tau) \wedge \mathsf{Cont}(k, \tau, w).$$

Lean4 (definition) `BEDC.Derived.RealUp.RealUnaryOffsetLe`
Lean4 (definition) `BEDC.Derived.RealUp.UnaryOffsetLe`

Theorem 38.152 (Unary offset continuation transitivity)**.** *For histories* $k, w, u, \tau, \sigma : \mathsf{Hist}$*,*

$$\mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(\tau) \wedge \mathsf{UnaryHistory}(\sigma) \wedge \mathsf{Cont}(k, \tau, w) \wedge \mathsf{Cont}(w, \sigma, u) \Longrightarrow \mathsf{UnaryOffsetLe}(k, u).$$

Proof. Use $\mathsf{append}(\tau, \sigma)$ as the composed offset tail. Unary append closure supplies its unary-history row, and continuation associativity turns the two displayed continuation rows into a continuation from k through that tail to u. Together with the given $\mathsf{UnaryHistory}(k)$ row, these data repack as $\mathsf{UnaryOffsetLe}(k, u)$. □

Lean4 ✓ `BEDC.Derived.RealUp.UnaryOffsetLe cont trans`

Theorem 38.153 (Unary offset append-tail closure)**.** *For histories* $k, w, \tau : \mathsf{Hist}$*,*

$$\mathsf{UnaryOffsetLe}(k, w) \wedge \mathsf{UnaryHistory}(\tau) \Longrightarrow \mathsf{UnaryOffsetLe}(k, \mathsf{append}(w, \tau)).$$

Proof. Unfold the offset witness for $\mathsf{UnaryOffsetLe}(k, w)$. It supplies $\mathsf{UnaryHistory}(k)$ and a unary tail σ with $\mathsf{Cont}(k, \sigma, w)$. Appending the displayed unary history τ to this tail gives $\mathsf{append}(\sigma, \tau)$, whose unary row follows from unary append closure. Continuation associativity turns $\mathsf{Cont}(k, \sigma, w)$ into a continuation from k through $\mathsf{append}(\sigma, \tau)$ to $\mathsf{append}(w, \tau)$, so the same left endpoint repacks as the displayed offset witness. □

Lean4 ✓ `BEDC.Derived.RealUp.UnaryOffsetLe append tail closed`

Definition 38.154 (Real unary stream finite window classifier)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w : \mathsf{Hist}$*, read* a *as the anchor and* w *as the bound. The finite window classifier is the pointwise RatUp classifier over every certified unary offset below* w*:*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \Longleftrightarrow \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(w) \wedge \forall k,\ \mathsf{UnaryOffsetLe}(k, w) \to \mathsf{Ratl}$$

Lean4 (definition) `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier`

Theorem 38.155 (Real unary stream finite-window coverage)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w : \mathsf{Hist}$*,*

$$\mathsf{RealUnaryStreamClassifier}(s, t) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(w) \Longrightarrow \mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w).$$

Proof. Unfold Definition 38.154. The anchor and bound fields are exactly the displayed assumptions $\mathsf{UnaryHistory}(a)$ and $\mathsf{UnaryHistory}(w)$.

Fix an offset k and assume $\mathsf{UnaryOffsetLe}(k, w)$. Unfold Definition 38.151; the offset witness projects $\mathsf{UnaryHistory}(k)$ together with a unary tail whose continuation reaches the bound.

Apply unary append closure to $\mathsf{UnaryHistory}(a)$ and $\mathsf{UnaryHistory}(k)$ to obtain $\mathsf{UnaryHistory}(\mathsf{append}(a, k))$. Evaluating $\mathsf{RealUnaryStreamClassifier}(s, t)$ at this certified observation gives

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, k)), t(\mathsf{append}(a, k))).$$

Since the offset was arbitrary under $\mathsf{UnaryOffsetLe}(k, w)$, these pointwise endpoint classifiers repack into $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier coverage`

Theorem 38.156 (Real stream window selected full shape readback package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, an anchor index* n*, a radius* m*, a bounded offset* k *with* $k \leq m$*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$*,* $\mathsf{hsame}(x(n + k), \mathsf{E1}(a))$*, and* $\mathsf{hsame}(y(n + k), \mathsf{E1}(b))$*. Then the selected endpoints at* $n + k$ *carry the following package:*

1. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$*;*
2. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$*;*
3. $\mathsf{PositiveUnaryDenominator}(x(n + k)) \wedge \mathsf{PositiveUnaryDenominator}(y(n + k))$*;*
4. $\mathsf{UnaryHistory}(x(n + k)) \wedge \mathsf{UnaryHistory}(y(n + k)) \wedge (\mathsf{hsame}(x(n + k), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n + k), \mathsf{e}) \to \mathsf{False})$*;*

5. $(\mathsf{hsame}(x(n+k), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n+k), \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Proof. Unfold the finite window classifier at the selected offset. The bound $k \leq m$ turns the window hypothesis into $\mathsf{RatHistoryClassifier}(x(n+k), y(n+k))$.

Transport this rational classifier across the two displayed one-seal endpoint comparisons by Definition 37.54. The transported classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$, and Definition 37.49 reads back the two unary tails together with $\mathsf{hsame}(a, b)$.

Apply Definition 37.50 to the original endpoint classifier. This gives positive unary denominator witnesses for $x(n+k)$ and $y(n+k)$, and Definition 37.12 converts those witnesses into unary and nonempty endpoint rows.

For the left zero-extension test, transport $\mathsf{PositiveUnaryDenominator}(x(n+k))$ along the proposed comparison $\mathsf{hsame}(x(n+k), \mathsf{E0}(z_X))$ by Definition 37.6. The resulting positive denominator on $\mathsf{E0}(z_X)$ contradicts Definition 37.10.

For the right zero-extension test, repeat the same transport from $y(n+k)$ to $\mathsf{E0}(z_Y)$. The positive-denominator zero-head absurdity theorem gives the second contradiction, completing the listed package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamWindowClassifier selected full shape readback package`

Lemma 38.157 (Real unary stream finite-window endpoint projection)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w, k : \mathsf{Hist}$,

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \mathsf{UnaryOffsetLe}(k, w) \Longrightarrow \mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, k)), t(\mathsf{append}($$

Proof. Unfold Definition 38.154. The window witness contains the universal endpoint row for every offset satisfying $\mathsf{UnaryOffsetLe}(\cdot, w)$.

Apply that row to the displayed offset witness $\mathsf{UnaryOffsetLe}(k, w)$. The result is precisely the RatUp endpoint classifier at $\mathsf{append}(a, k)$. □

Theorem 38.158 (Real unary stream window finite-probe coverage)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, a unary anchor* $a : \mathsf{Hist}$*, a bound* $w : \mathsf{Hist}$*, and a probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*, suppose*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$$

and suppose that every bundled certified unary observation lies below the window bound:

$$\forall n : \mathsf{Hist}, \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{UnaryOffsetLe}(n, w).$$

Then the shifted observation maps export a StreamName finite probe window:

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\lambda n.\, s(\mathsf{append}(a, n)), \lambda n.\, t(\mathsf{append}(a, n)), \mathcal{P}).$$

Proof. Unfold Definition 126.21. Fix a bundled certified unary observation n with $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{UnaryHistory}(n)$.

The admissibility hypothesis consumes exactly these two rows and produces the offset witness $\mathsf{UnaryOffsetLe}(n, w)$. Apply Definition 38.157 to the Real unary window classifier and this offset witness.

The projected endpoint classifier is

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, n)), t(\mathsf{append}(a, n))).$$

This is the required StreamName finite-window row for the shifted observation maps at the chosen bundled unary observation. Since the bundled observation was arbitrary, the displayed rows repack into $\mathsf{RatStreamNameFiniteWindowClassifier}(\lambda n.\, s(\mathsf{append}(a, n)), \lambda n.\, t(\mathsf{append}(a, n)), \mathcal{P})$. □

Theorem 38.159 (Real unary window finite-probe determinacy)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, a unary anchor* $a : \mathsf{Hist}$*, a bound* $w : \mathsf{Hist}$*, and probe bundles* $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$*, suppose*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$$

and suppose every bundled certified unary observation of $\mathcal{P}$ *lies below the window bound:*

$$\forall n : \mathsf{Hist}, \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{UnaryOffsetLe}(n, w).$$

Suppose also that $\mathcal{P}$ *and* $\mathcal{Q}$ *have the same membership rows on certified unary histories:*

$$\forall n : \mathsf{Hist},\ \mathsf{UnaryHistory}(n) \to (\mathsf{InBundle}(n, \mathcal{P}) \iff \mathsf{InBundle}(n, \mathcal{Q})).$$

Then the shifted StreamName finite-window surface is independent of the chosen presentation:

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\lambda n.\, s(\mathsf{append}(a, n)), \lambda n.\, t(\mathsf{append}(a, n)), \mathcal{P}) \iff \mathsf{RatStreamNameFiniteWindo}$$

Proof. Unfold Definition 126.21. For the forward direction, fix a bundled certified unary observation n of $\mathcal{Q}$. The membership-equivalence hypothesis sends its $\mathsf{InBundle}(n, \mathcal{Q})$ row to an $\mathsf{InBundle}(n, \mathcal{P})$ row.

The $\mathcal{P}$ boundedness hypothesis now gives $\mathsf{UnaryOffsetLe}(n, w)$. Apply Definition 38.157 to the shared Real unary window classifier and this bounded-offset witness. The result is the required point row

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, n)), t(\mathsf{append}(a, n))).$$

For the reverse direction, fix a bundled certified unary observation n of $\mathcal{P}$. The same membership-equivalence row also identifies it as a bundled observation of $\mathcal{Q}$, so both presentations expose the same finite probe site.

The boundedness hypothesis for $\mathcal{P}$ supplies $\mathsf{UnaryOffsetLe}(n, w)$. Applying Definition 38.157 again to the shared Real unary window classifier gives the RatUp endpoint classifier at $\mathsf{append}(a, n)$, which is exactly the required $\mathsf{RatStreamNameFiniteWindowClassifier}$ row for $\mathcal{P}$. □

The ten-field fit is: traditional target finite observation-window bridge; BEDC source RealUp plus $\mathsf{StreamNameUp}$ plus RatUp plus $\mathsf{ProbeBundle}$; carrier the shifted observation maps $n \mapsto s(\mathsf{append}(a, n))$ and $n \mapsto t(\mathsf{append}(a, n))$; classifier $\mathsf{RatStreamNameFiniteWindowClassifier}$; stability inherited from the Real window endpoint row; ledger anchor, bound, bundle membership, and offset witness; core theorem Definition 38.158; exactness the bundled offsets are exactly consumed pointwise; standard bridge $\mathsf{RegSeqRat}$ finite-prefix consumption; status $\mathsf{ProofTarget}$.

The ten-field fit is: traditional target finite observation windows for RealUp streams; BEDC source $\mathsf{RealUnaryStreamClassifier}$ plus $\mathsf{RatHistoryClassifier}$ plus $\mathsf{UnaryHistory}$ plus Cont; carrier BHist observation maps; classifier $\mathsf{RealUnaryStreamWindowClassifier}$; stability by unary append closure and pointwise classifier evaluation; ledger anchor, bound, offset, and tail witness; core theorem Definition 38.155; exactness endpoint projection Definition 38.157; standard bridge finite rational observation windows; status $\mathsf{ProofTarget}$.

Theorem 38.160 (Real unary stream finite-window empty bound exactness)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and an anchor history* $a : \mathsf{Hist}$*, the empty finite-window bound is exactly the anchor row:*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{e}) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{RatHistoryClassifier}(s(a), t(a)).$$

Proof. For the forward direction, the window classifier already stores the unary anchor row. The empty history is below the empty bound using the empty unary tail and the continuation identity $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. Projecting the endpoint row at this offset gives $\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, \mathsf{e})), t(\mathsf{append}(a, \mathsf{e})))$, which is the displayed anchor classifier.

For the reverse direction, assume the anchor unary row and the anchor classifier. The bound field is certified by unary emptiness. For any offset k below e, the continuation witness into the empty result forces $k = \mathsf{e}$. The endpoint demanded by the window classifier is therefore the stored anchor classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier empty bound iff`

Lemma 38.161 (Unary offset bound transitivity)**.** *For histories $k, w, u : \mathsf{Hist}$,*

$$\mathsf{UnaryOffsetLe}(k, w) \wedge \mathsf{UnaryOffsetLe}(w, u) \Longrightarrow \mathsf{UnaryOffsetLe}(k, u).$$

Proof. Unfold Definition 38.151 in the first offset row. It gives $\mathsf{UnaryHistory}(k)$ and a unary tail $\tau : \mathsf{Hist}$ with $\mathsf{Cont}(k, \tau, w)$.

Unfold the second offset row. It gives a unary tail $\sigma : \mathsf{Hist}$ with $\mathsf{Cont}(w, \sigma, u)$, and it also certifies the intermediate bound w as a unary history.

Apply unary append closure to the two tails to obtain $\mathsf{UnaryHistory}(\mathsf{append}(\tau, \sigma))$. Continuation associativity composes $\mathsf{Cont}(k, \tau, w)$ with $\mathsf{Cont}(w, \sigma, u)$, and determinacy identifies the composed endpoint with u.

Thus k has the unary tail $\mathsf{append}(\tau, \sigma)$ whose continuation reaches u, so Definition 38.151 repacks this data as $\mathsf{UnaryOffsetLe}(k, u)$. □

Lemma 38.162 (Unary offset bound reflexivity)**.** *For every history $w : \mathsf{Hist}$,*

$$\mathsf{UnaryHistory}(w) \Longrightarrow \mathsf{UnaryOffsetLe}(w, w).$$

Proof. Unfold Definition 38.151. The displayed hypothesis supplies the required unary witness for the left endpoint w.

Choose the tail e. Unary emptiness gives $\mathsf{UnaryHistory}(\mathsf{e})$, and the right-unit continuation law witnesses $\mathsf{Cont}(w, \mathsf{e}, w)$. These three pieces repack as $\mathsf{UnaryOffsetLe}(w, w)$. □

Theorem 38.163 (Real unary stream finite-window endpoint at bound)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and histories $a, w : \mathsf{Hist}$,*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \Longrightarrow \mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, w)), t(\mathsf{append}(a, w))).$$

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier endpoint at bound`

Proof. Unfold Definition 38.154 only far enough to read the stored bound field $\mathsf{UnaryHistory}(w)$.

Apply Definition 38.162 to this bound field. This gives the explicit witness $\mathsf{UnaryOffsetLe}(w, w)$ required for endpoint projection.

Now apply Definition 38.157 with $k = w$. The projected endpoint is exactly the boundary observation $\mathsf{append}(a, w)$. □

The ten-field fit is: traditional target finite observation boundary readback; BEDC source RealUnaryStreamWindowClassifier plus UnaryOffsetLe; carrier BHist observation maps; classifier boundary endpoint RatUp classifier; stability inherited from unary append closure and endpoint projection; ledger the boundary offset w and the empty continuation tail; core theorem Definition 38.163; exactness the bound is itself a certified finite-window offset; standard bridge RegSeqRat finite boundary observation; status ProofTarget.

Theorem 38.164 (Real unary stream finite-window restriction)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w, u : \mathsf{Hist}$*,*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, u) \wedge \mathsf{UnaryOffsetLe}(w, u) \Longrightarrow \mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w).$$

Proof. Unfold Definition 38.154 in the source window. Its anchor field gives $\mathsf{UnaryHistory}(a)$, and the offset witness $\mathsf{UnaryOffsetLe}(w, u)$ projects $\mathsf{UnaryHistory}(w)$ by Definition 38.151.

Fix any offset k with $\mathsf{UnaryOffsetLe}(k, w)$. Apply Definition 38.161 to $\mathsf{UnaryOffsetLe}(k, w)$ and $\mathsf{UnaryOffsetLe}(w, u)$, obtaining $\mathsf{UnaryOffsetLe}(k, u)$.

The universal endpoint row of the source window at bound u then yields

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, k)), t(\mathsf{append}(a, k))).$$

Since the anchor, restricted bound, and every endpoint row have been supplied, Definition 38.154 repacks them as $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier restriction with offset witness`

The ten-field fit is: traditional target shrinkable finite observation windows for RealUp streams; BEDC source RealUnaryStreamWindowClassifier plus UnaryOffsetLe plus Cont; carrier unary anchor and certified unary bounds; classifier the restricted finite-window predicate; stability by offset-bound transitivity and endpoint reuse at the larger bound; ledger anchor, source bound, target bound, offset, and composed tail witnesses; core theorem Definition 38.164; exactness every endpoint below the restricted bound is an endpoint below the source bound; standard bridge RegSeqRat finite observation bounds; status ProofTarget.

Theorem 38.165 (Real unary stream window selected tail-class exactness from window)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w, k : \mathsf{Hist}$*, assume the finite window classifier and a selected certified offset:*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \mathsf{UnaryOffsetLe}(k, w).$$

Then the selected tail class is exact:

$$\begin{aligned}
&\exists u, v : \mathsf{Hist},\ \mathsf{hsame}(s(\mathsf{append}(a, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(a, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\forall u', v' : \mathsf{Hist},\ (\mathsf{hsame}(s(\mathsf{append}(a, k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(a, k)), \mathsf{E1}(v'))) \Longleftrightarrow \\
&\qquad (\mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v')).
\end{aligned}$$

Proof. Project the selected RatUp classifier directly from $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$ at the offset k. Positive denominator readback supplies displayed one-headed representatives for both selected endpoints. Transport the selected rational classifier across these two endpoint displays, then apply the e-one tail classifier exactness theorem to obtain unary tails and their hsame classifier.

For any displayed pair u', v', transport the same selected classifier across the displayed endpoint rows and read back the e-one tails. Comparing the chosen endpoint displays with the arbitrary ones by hsame transitivity and e-one-tail inversion gives $\mathsf{hsame}(u, u')$ and $\mathsf{hsame}(v, v')$. Conversely, tail comparisons $\mathsf{hsame}(u, u')$ and $\mathsf{hsame}(v, v')$ lift by e-one congruence and compose with the chosen endpoint displays, giving the displayed pair. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected tail class exactness from window`

Definition 38.166 (Real stream finite window classifier)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor index n, and a zero-based window radius m, the finite window classifier is the bounded pointwise rational-history classifier*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m) \iff \forall k,\ k \le m \implies \mathsf{RatHistoryClassifier}(x(n+k), y(n+k)).$$

Lean4 (definition) `BEDC.Derived.RealUp.RealStreamWindowClassifier`

Theorem 38.167 (Real stream window classifier hsame transport)**.** *For rational-observation streams $x, y, x', y' : \mathbb{N} \to \mathsf{Hist}$, an anchor index n, and a radius m, assume* $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$*. If every bounded offset $k \le m$ carries both endpoint rows $x(n+k)\mathsf{hsame}x'(n+k)$ and $y(n+k)\mathsf{hsame}y'(n+k)$, then* $\mathsf{RealStreamWindowClassifier}(x', y', n, m)$*.*

Proof. Unfold the finite window classifier and fix an arbitrary offset $k \le m$. The source window hypothesis supplies the endpoint classifier $\mathsf{RatHistoryClassifier}(x(n+k), y(n+k))$.

The offset transport hypothesis gives the two rows $x(n+k)\mathsf{hsame}x'(n+k)$ and $y(n+k)\mathsf{hsame}y'(n+k)$ at the same bounded observation. Apply Definition 37.54 to the source rational classifier and these endpoint rows.

The result is $\mathsf{RatHistoryClassifier}(x'(n+k), y'(n+k))$. Since the offset was arbitrary under the bound $k \le m$, these transported endpoint classifiers repack into $\mathsf{RealStreamWindowClassifier}(x', y', n, m)$. □

The ten-field fit is: traditional target finite observation-window stability for RealUp names; BEDC source RealUp windows plus RatUp endpoint classifier transport; carrier bounded rational-observation windows; classifier RealStreamWindowClassifier; stability endpoint hsame transport; ledger anchor, radius, offset, and two endpoint transports; core theorem Definition 38.167; exactness preservation of the bounded classifier; standard bridge finite prefix stability for regular rational observation names; status ProofTarget.

Theorem 38.168 (Real stream prefix-window exactness at zero)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and every finite offset m,*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m) \iff \mathsf{RealStreamWindowClassifier}(x, y, 0, m).$$

Proof. Induct on m. At $m = 0$, the recursive prefix classifier is exactly the rational-history classifier at observation 0, and the window condition has only the offset $k = 0$.

For the successor step, use Definition 38.113 to split a prefix through $m + 1$ into the prefix through m and the rational-history classifier at the new endpoint. The induction hypothesis converts the shorter prefix into the window condition through offset m.

Given any $k \le m + 1$, split constructively between $k \le m$ and $k = m + 1$. The first case is handled by the shorter window; the endpoint case is exactly the successor rational-history classifier.

Conversely, assume the window condition through $m + 1$. Its restriction to offsets $k \le m$ gives the zero-based window through m, hence by induction the prefix through m. Taking $k = m+1$ gives the endpoint classifier, and the reverse direction of successor inversion pairs these two witnesses into the prefix through $m + 1$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier finite window zero exactness`

Theorem 38.169 (Real stream classifier finite-window coverage)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, m, every pointwise stream classifier covers the finite observation window anchored at n with radius m:*

$$\mathsf{RealStreamClassifier}(x, y) \implies \mathsf{RealStreamWindowClassifier}(x, y, n, m).$$

Proof. Assume $\mathsf{RealStreamClassifier}(x, y)$ and fix an offset $k \leq m$. The stream classifier is pointwise, so evaluating it at the observation index $n + k$ gives $\mathsf{RatHistoryClassifier}(x(n + k), y(n + k))$.

Since the chosen offset was arbitrary under the window bound, these endpoint classifiers assemble exactly into $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$. □

Theorem 38.170 (Real stream classifier zero-window exactness)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$,

$$\mathsf{RealStreamClassifier}(x, y) \iff \forall n,\ \mathsf{RealStreamWindowClassifier}(x, y, n, 0).$$

Proof. For the forward direction, assume $\mathsf{RealStreamClassifier}(x, y)$. For each anchor n, apply Definition 38.169 with radius 0. This gives $\mathsf{RealStreamWindowClassifier}(x, y, n, 0)$ for every anchor.

For the reverse direction, assume every zero-radius window is classified and fix an observation index n. Unfold the window at anchor n and radius 0. Taking offset $k = 0$ uses $0 \leq 0$ and the identity $n + 0 = n$, so the window row returns $\mathsf{RatHistoryClassifier}(x(n), y(n))$.

Since n was arbitrary, these rational-history rows repack as the pointwise stream classifier $\mathsf{RealStreamClassifier}(x, y)$. □

The ten-field fit is: traditional target pointwise equality of countable real names; BEDC source RealUp stream classifier plus RatUp endpoint classifier; carrier rational-observation streams; classifier all zero-radius windows; stability inherited from finite-window stability; ledger anchor index and zero offset; core theorem this exactness; exactness all pointwise observations iff all zero windows; standard bridge finite observation equality of regular rational names; status ProofTarget.

Theorem 38.171 (Real stream finite-window stability package)**.** *For rational-observation streams* $x, x', y, y', z : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, m*, the finite window classifier supplies the following stability rows:*

1. *if* $\forall k,\ k \leq m \Longrightarrow \mathsf{RatHistoryCarrier}(x(n + k))$*, then* $\mathsf{RealStreamWindowClassifier}(x, x, n, m)$*;*
2. *if* $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$*, then* $\mathsf{RealStreamWindowClassifier}(y, x, n, m)$*;*
3. *if* $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$ *and* $\mathsf{RealStreamWindowClassifier}(y, z, n, m)$*, then* RealStreamWindowCla
4. *if* $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$ *and, for every* $k \leq m$*, both* $\mathsf{hsame}(x(n + k), x'(n + k))$ *and* $\mathsf{hsame}(y(n + k), y'(n + k))$ *hold, then* $\mathsf{RealStreamWindowClassifier}(x', y', n, m)$*.*

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamWindowClassifier stability package`

Proof. Unfold the finite window classifier in each row and fix an offset $k \leq m$.

1. The reflexivity row reads the carrier hypothesis at this offset. Pair the carrier with itself, and use hsame reflexivity to build $\mathsf{RatHistoryClassifier}(x(n + k), x(n + k))$.
2. The symmetry row evaluates the source window at the offset and applies Definition 37.55 to reverse the rational-history classifier.
3. The transitivity row evaluates the two source windows at the same offset and applies Definition 37.53 to compose the two rational-history classifiers through the shared middle stream y.
4. The transport row evaluates the source window at the offset and reads the two pointwise hsame hypotheses at the same offset. Apply Definition 37.54 to transport the rational-history classifier to the endpoints $x'(n + k)$ and $y'(n + k)$.

Since the offset was arbitrary under the bound, each row repacks into the corresponding RealStreamWindowClassifier statement. □

The ten-field fit is: traditional target finite observation-window stability; BEDC source RealUp rational-observation streams plus RatUp classifier; carrier the bounded window predicate; classifier RealStreamWindowClassifier; stability the four displayed rows; ledger anchor n, radius m, and offset witnesses; core theorem this block; exactness bounded pointwise rational classification; standard bridge finite rational observation-window agreement; status ProofTarget.

Theorem 38.172 (Real stream finite-window subwindow extraction)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, j, r, m, if*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m)$$

and $j + r \leq m$, then

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, r).$$

Proof. Unfold Definition 38.166 in the source window and fix a shifted offset $k \leq r$. From $k \leq r$ and $j + r \leq m$, bounded-offset monotonicity gives $j + k \leq j + r \leq m$.

Apply the source window at offset $j + k$. This yields the endpoint classifier

$$\mathsf{RatHistoryClassifier}(x(n + (j + k)), y(n + (j + k))).$$

The arithmetic identity $(n+j)+k = n+(j+k)$ identifies this endpoint with the corresponding endpoint of the shifted window. Since the chosen k was arbitrary under the radius r, these endpoint classifiers repack as $\mathsf{RealStreamWindowClassifier}(x, y, n + j, r)$. □

Theorem 38.173 (Real stream finite-window selected zero-subwindow package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, j, m, if*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m)$$

and $j \leq m$, then both

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, 0)$$

and

$$\mathsf{RatHistoryClassifier}(x(n + j), y(n + j))$$

hold.

Proof. Apply Definition 38.172 to the source window with shift j and radius 0. The displayed bound $j \leq m$ is exactly the required inequality $j + 0 \leq m$, so this gives the zero-radius shifted window

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, 0).$$

It remains to read the selected endpoint. Unfold Definition 38.166 for the zero-radius window and take offset $0 \leq 0$. The endpoint index is $(n + j) + 0 = n + j$, so the window row returns $\mathsf{RatHistoryClassifier}(x(n+j), y(n+j))$, as in the zero-window exactness calculation of Definition 38.170. □

The ten-field fit is: traditional target selected finite RealUp observation normalization; BEDC source RealStreamWindowClassifier plus RatHistoryClassifier; carrier rational-observation streams; classifier selected zero-radius finite-window predicate; stability by subwindow extraction at radius 0 and endpoint projection; ledger finite anchor, selected index, source radius, zero radius, and bound witness $j \leq m$; core theorem Definition 38.173; exactness the selected point is read as the unique zero-radius endpoint; standard bridge downstream RegSeqRat selected finite observation normalization; status ProofTarget.

The ten-field fit is: traditional target shifted finite RealUp observation windows; BEDC source RealStreamWindowClassifier plus RatHistoryClassifier; carrier rational-observation streams; classifier shifted finite-window predicate; stability by bounded-offset monotonicity and endpoint reuse; ledger original anchor, shift, radius, source bound, and offset witness; core theorem Definition 38.172; exactness every shifted endpoint is already a source-window endpoint; standard bridge RegSeqRat tail/window stability; status ProofTarget.

Theorem 38.174 (Real stream finite-window nested subwindow extraction)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, j, q, r, s, m, if*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

$j + r \leq m$, and $q + s \leq r$, then

$$\mathsf{RealStreamWindowClassifier}(x, y, (n + j) + q, s).$$

Proof. Apply Definition 38.172 to the source window with shift j and radius r. The bound $j+r \leq m$ produces the first subwindow

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, r).$$

Apply the same subwindow theorem again to this derived window with shift q and radius s. The bound $q + s \leq r$ gives

$$\mathsf{RealStreamWindowClassifier}(x, y, (n + j) + q, s).$$

Every endpoint of the nested window has the form $(n + j) + (q + k)$ with $k \leq s$. The second bound places $q + k$ inside the first subwindow, and the first bound places $j + (q + k)$ inside the original window. Thus the displayed classifier is exactly the repeated finite-window extraction. □

Theorem 38.175 (Real stream nested finite-window endpoint package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, indices n, j, q, r, s, m, and histories $a, b, z_X, z_Y : \mathsf{Hist}$, assume*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

$j + r \leq m$, $q + s \leq r$, $\mathsf{hsame}(x((n + j) + q), \mathsf{E1}(a))$, and $\mathsf{hsame}(y((n + j) + q), \mathsf{E1}(b))$. Then the nested endpoint carries the following package:

1. $\mathsf{RealStreamWindowClassifier}(x, y, (n + j) + q, s)$;
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$;
3. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$;
4. $\mathsf{PositiveUnaryDenominator}(x((n + j) + q)) \wedge \mathsf{PositiveUnaryDenominator}(y((n + j) + q))$;

5. $\mathsf{UnaryHistory}(x((n+j)+q)) \wedge \mathsf{UnaryHistory}(y((n+j)+q)) \wedge (\mathsf{hsame}(x((n+j)+q), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y((n+j)+q), \mathsf{e}) \to \mathsf{False})$;
6. $(\mathsf{hsame}(x((n+j)+q), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y((n+j)+q), \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Proof. First apply Definition 38.174 to the source window with the displayed bounds. This produces the nested finite-window row $\mathsf{RealStreamWindowClassifier}(x, y, (n+j)+q, s)$.

Next read the selected endpoint of that nested window at offset 0. The bound $0 \leq s$ lets Definition 38.156 apply to the nested window, with anchor $(n+j)+q$ and displayed one-extension rows for $x((n+j)+q)$ and $y((n+j)+q)$.

The selected full-shape readback package supplies the endpoint $\mathsf{Rat}^{\uparrow}$ classifier, the positive unary denominator rows, the unary endpoint rows, the two nonempty exclusions, the two zero-headed exclusions for z_X and z_Y, and the tail readback $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$. Repack these rows together with the nested window obtained in the first step. □

Theorem 38.176 (Real stream affine subwindow collapse)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, indices n, j, q, r, s, m, and histories $a, b, z_X, z_Y : \mathsf{Hist}$, assume*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

$j+r \leq m$, $q+s \leq r$, $\mathsf{hsame}(x((n+j)+q), \mathsf{E1}(a))$, and $\mathsf{hsame}(y((n+j)+q), \mathsf{E1}(b))$. Then the nested package obtained by first restricting the source window to $(n+j, r)$ and then to $((n+j)+q, s)$ is the same public finite-window classifier and endpoint read as the one-step subwindow at offset $j+q$ and radius s. In particular, it exposes exactly the same rows:

1. $\mathsf{RealStreamWindowClassifier}(x, y, (n+j)+q, s)$, *equivalently* $\mathsf{RealStreamWindowClassifier}(x, y, n+(j+q), s)$ *under finite-index arithmetic;*
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$;
3. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$;
4. $\mathsf{PositiveUnaryDenominator}(x((n+j)+q)) \wedge \mathsf{PositiveUnaryDenominator}(y((n+j)+q))$;
5. $\mathsf{UnaryHistory}(x((n+j)+q)) \wedge \mathsf{UnaryHistory}(y((n+j)+q)) \wedge (\mathsf{hsame}(x((n+j)+q), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y((n+j)+q), \mathsf{e}) \to \mathsf{False})$;
6. $(\mathsf{hsame}(x((n+j)+q), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y((n+j)+q), \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Moreover, for every $k \leq s$, the nested endpoint family indexed by $(n+j)+(q+k)$ is the same finite observation family as the affine subwindow family indexed by $n+((j+q)+k)$.

Proof. Apply Definition 38.174 to the source window and the displayed bounds. This gives the nested classifier at anchor $(n+j)+q$ and radius s.

Apply Definition 38.172 directly to the source window with shift $j+q$ and radius s. The bounds $j+r \leq m$ and $q+s \leq r$ compose to $j+q+s \leq m$, so the direct affine subwindow is also available at anchor $n+(j+q)$ and radius s.

The finite arithmetic identities $(n+j)+q = n+(j+q)$ and $(n+j)+(q+k) = n+((j+q)+k)$ identify each nested endpoint with the corresponding one-step affine endpoint. Thus the two classifier rows read the same bounded family of $\mathsf{Rat}^{\uparrow}$ endpoint classifiers.

Finally apply Definition 38.175 to the nested anchor. Its endpoint package supplies the $\mathsf{RatHistoryClassifier}$ row, the tail $\mathsf{UnaryHistory}$ and hsame row, the positive-denominator rows, the nonempty exclusions, and the zero-headed exclusions. Transporting those rows along the displayed endpoint equalities repacks exactly the one-step affine subwindow package. □

The ten-field fit is: traditional target nested finite RealUp observation windows; BEDC source RealStreamWindowClassifier over rational-observation streams; carrier rational-observation streams; classifier the finite-window predicate; stability by repeated subwindow extraction and bounded-offset monotonicity; ledger original anchor, first shift, second shift, outer radius, inner radius, and offset witnesses; core theorem Definition 38.174; exactness every nested endpoint is already in the original window; standard bridge RegSeqRat nested tail/window stability; status ProofTarget.

Theorem 38.177 (Constant rational real finite-window exactness)**.** *For histories* d, e, a, w : Hist *with* UnaryHistory(a) *and* UnaryHistory(w)*,*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RealUnaryStreamWindowClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e, a, w).$$

Proof. For the forward direction, assume RatHistoryClassifier(d, e). The constant-stream witness Definition 126.48 supplies RatStreamNameClassifier$(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e)$.

Read this StreamNameUp classifier as the corresponding RealUp unary-observation stream classifier by Definition 38.27. With the displayed unary anchor and bound witnesses, apply Definition 38.155 to obtain the finite-window classifier.

For the reverse direction, assume RealUnaryStreamWindowClassifier$(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e, a, w)$. The empty history is a certified unary offset below w: use the unary witness for w, the unary witness for e, and the continuation row Cont(e, w, w).

Apply Definition 38.157 at the offset e. This yields

$$\mathsf{RatHistoryClassifier}(\mathsf{RatConstStream}_d(\mathsf{append}(a, \mathsf{e})), \mathsf{RatConstStream}_e(\mathsf{append}(a, \mathsf{e}))).$$

Unfold RatConstStream. Both constant streams ignore the observation history append(a, e), so the projected endpoint classifier is exactly RatHistoryClassifier(d, e). □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier constant rational exactness iff`

The ten-field fit is: traditional target constant rational observations inside RealUp finite windows; BEDC source RatUp plus StreamNameUp plus RealUp; carrier constant rational observation streams; classifier the displayed equivalence; stability inherited from constant stream classification and finite-window endpoint projection; ledger hidden rational seed, anchor, bound, and empty offset; core theorem Definition 38.177; exactness the displayed equivalence; standard bridge constant rational finite observation windows; status ProofTarget.

Theorem 38.178 (Real stream shifted prefix-window exactness)**.** *For rational-observation streams* $x, y : \mathbb{N} \to$ Hist *and indices* n, m*, define shifted streams* $x_n, y_n : \mathbb{N} \to$ Hist *by* $x_n(k) = x(n + k)$ *and* $y_n(k) = y(n + k)$*. Then*

$$\mathsf{RealStreamPrefixClassifier}(x_n, y_n, m) \iff \mathsf{RealStreamWindowClassifier}(x, y, n, m).$$

Proof. Apply Definition 38.168 to the shifted streams x_n and y_n. This gives the equivalence between the recursive prefix classifier on the shifted streams and the zero-anchored window classifier RealStreamWindowClassifier$(x_n, y_n, 0, m)$.

Unfold the zero-anchored window classifier. Its endpoint rows are $x_n(0 + k)$ and $y_n(0 + k)$ for bounded offsets $k \leq m$.

By the definitions of the shifted streams and the identity $0 + k = k$, these rows are exactly $x(n + k)$ and $y(n + k)$. Repacking the same bounded endpoint classifiers is precisely RealStreamWindowClassifier(x, y, n, m). □

Theorem 38.179 (Real stream shifted prefix-window seal pullback)**.** *Let $x, y : \mathbb{N} \to$ Hist be rational-observation streams, let $x_n(k) = x(n+k)$ and $y_n(k) = y(n+k)$, and fix a finite radius m. Suppose two downstream* Real$^{\uparrow}$ *finite-window seal uses are restricted to the same shifted window $(x_n, y_n, 0, m)$. If their restricted surfaces agree on the* RatStreamNameFiniteWindowClassifier *rows, the* RegSeqRat *spine rows, the* DyadicRatCoreObservationRow *endpoint observations, and the displayed* hsame/Cont/Pkg/RealLedgerPolicy *rows, then the shifted prefix packet over (x_n, y_n, m) and the unshifted window packet over (x, y, n, m) expose the same finite* BHist *source surface for both seal uses.*

The pullback is only a finite source-surface statement. It exports no quotient data, completion data, metric-completeness row, arithmetic or order closure, Classical representative, or standard bridge datum.

Proof. Apply Definition 38.178. It identifies the shifted prefix classifier over (x_n, y_n, m) with the unshifted finite window classifier over (x, y, n, m), so both presentations name the same bounded endpoint rows.

Next apply Definition 38.288 to the two downstream seal uses over the shifted window. The hypotheses give the required matching RatStreamNameFiniteWindowClassifier rows, RegSeqRat spine rows, DyadicRatCore endpoints, and displayed hsame/Cont/Pkg provenance over that finite packet.

Finally apply Definition 38.355. The shared finite RealLedgerPolicy row identifies the induced Real$^{\uparrow}$ readback packet with the same finite BHist source surface. The preceding exactness step transports this packet back to the unshifted window over (x, y, n, m), and the negative boundary is inherited from the two pullback theorems. □

Theorem 38.180 (Real stream shifted prefix-window classifier composition)**.** *Let $x, y : \mathbb{N} \to$ Hist be rational-observation streams, let $x_n(k) = x(n+k)$ and $y_n(k) = y(n+k)$, and fix a finite radius m. Suppose the shift certificate from (x, y, n, m) to $(x_n, y_n, 0, m)$ carries* RealStreamPrefixClassifier(x_n, y_n, m) *to* RealStreamWindowClassifier(x, y, n, m)*. Suppose also that the shifted window is passed to a downstream finite-window handoff whose displayed rows are the* RatStreamNameFiniteWindowClassifier *row, the* RegSeqRat *spine, the endpoint* DyadicRatCoreObservationRow *witnesses, and the shared* hsame/Cont/Pkg/RealLedgerPolicy *ledger. Then the* RealStreamWindowClassifier *row consumed at (x, y, n, m) is exactly the finite* BHist *classifier row obtained by first reading the shifted prefix classifier over (x_n, y_n, m) and then passing through the* StreamName$^{\uparrow}$-RegSeqRat *finite-window handoff.*

This is only a finite classifier-composition law. It does not introduce a quotient carrier, a completion carrier, host real equality, metric completeness, arithmetic or order closure, Classical representatives, or bridge status.

Proof. First apply Definition 38.178. It identifies the shifted prefix classifier over (x_n, y_n, m) with the unshifted finite-window classifier over (x, y, n, m), so the downstream consumer and the original window read the same bounded endpoint rows.

Next apply Definition 38.179. The hypotheses provide the matching RatStreamNameFiniteWindowClassifier rows, RegSeqRat spine rows, DyadicRatCore endpoints, and displayed hsame/Cont/Pkg/RealLedgerPolicy ledger on the shifted window. The pullback identifies the shifted prefix packet and the unshifted window packet as the same finite BHist source surface.

Finally apply Definition 38.344 together with Definition 38.355. The correspondence factors the finite Real$^{\uparrow}$ window through the public StreamName$^{\uparrow}$ finite-window interface and the RegSeqRat dyadic source rows; the pullback says that these rows repack as the same finite classifier packet. Thus the row consumed at (x, y, n, m) is precisely the row obtained by the shifted-prefix route followed by the StreamName-RegSeqRat handoff, with the negative boundary inherited from the cited finite-window theorems. □

The ten-field fit is: traditional target shifted finite RealUp observation windows; BEDC source RealUp prefix/window classifiers plus RatUp endpoints; carrier rational-observation streams; classifier the displayed equivalence; stability zero-anchor exactness together with the index shift; ledger anchor n, radius m, and offset k; core theorem Definition 38.178; exactness arbitrary-anchor window iff shifted recursive prefix; standard bridge finite RegSeqRat observation windows; status ProofTarget.

Theorem 38.181 (Real unary StreamNameUp finite-window endpoint shape package)**.** *For observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, histories* $a, w, k, u, v, z_s, z_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w),$$

the pointwise transport rows

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)),$$

and the displayed finite-window endpoints

$$\mathsf{hsame}(s'(\mathsf{append}(a, k)), \mathsf{E1}(u)), \quad \mathsf{hsame}(t'(\mathsf{append}(a, k)), \mathsf{E1}(v)).$$

Then the selected transported endpoints carry the bounded RealUp readback package:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(s'(\mathsf{append}(a, k)), t'(\mathsf{append}(a, k)))\wedge\\
&\mathsf{PositiveUnaryDenominator}(s'(\mathsf{append}(a, k))) \wedge \mathsf{PositiveUnaryDenominator}(t'(\mathsf{append}(a, k)))\wedge\\
&\mathsf{UnaryHistory}(s'(\mathsf{append}(a, k))) \wedge \mathsf{UnaryHistory}(t'(\mathsf{append}(a, k)))\wedge\\
&(\mathsf{hsame}(s'(\mathsf{append}(a, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(\mathsf{append}(a, k)), \mathsf{e}) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s'(\mathsf{append}(a, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(\mathsf{append}(a, k)), \mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v).
\end{aligned}$$

Proof. First convert the StreamNameUp classifier into the RealUp unary stream classifier by the reverse direction of Definition 38.551. The carrier rows for s and t are the two fields projected from $\mathsf{RatStreamNameClassifier}(s, t)$, so the classifier row becomes $\mathsf{RealUnaryStreamClassifier}(s, t)$.

Apply Definition 38.553 to the projected carrier rows, the same StreamNameUp classifier, the anchor a, and the bound w. This gives $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. Project the selected offset with Definition 38.157, using $\mathsf{UnaryOffsetLe}(k, w)$, to obtain

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(a, k)), t(\mathsf{append}(a, k))).$$

Unfolding the offset bound supplies $\mathsf{UnaryHistory}(k)$, and unary append closure with $\mathsf{UnaryHistory}(a)$ gives $\mathsf{UnaryHistory}(\mathsf{append}(a, k))$. Evaluate the two pointwise transport rows at this appended observation. Transport the projected RatUp classifier along those two hsame rows by Definition 37.54; the result is

$$\mathsf{RatHistoryClassifier}(s'(\mathsf{append}(a, k)), t'(\mathsf{append}(a, k))).$$

Apply Definition 37.50 to the transported classifier. Then apply Definition 37.12 separately to the two positive-denominator rows. These projections give the positive denominator witnesses, the unary endpoint rows, and the two exclusions against e.

For a displayed zero head on the left endpoint, transport the left positive denominator along $\mathsf{hsame}(s'(\mathsf{append}(a, k)), \mathsf{E0}(z_s))$ by Definition 37.6. This contradicts Definition 37.10. The right endpoint uses the same argument with $t'(\mathsf{append}(a, k))$ and z_t.

Finally, transport the selected RatUp classifier along the two displayed E1 endpoint comparisons and apply Definition 37.49. This reads back $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, and $\mathsf{hsame}(u, v)$, and the collected rows are exactly the stated package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier streamName transported endpoint shape package`

The ten-field fit is: traditional target finite RealUp observation readback; BEDC source StreamNameUp plus RealUp finite windows plus RatUp endpoint classifier; carrier observation maps over BHist; classifier RealUnaryStreamWindowClassifier; stability by hsame transport and RatUp classifier laws; ledger anchor, bound, offset, and transported endpoints; core theorem Definition 38.181; exactness selected endpoint package; standard bridge finite rational observation windows; status ProofTarget.

Theorem 38.182 (Real stream finite-window adjacent concatenation)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, m, r*, if*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m)$$

and

$$\mathsf{RealStreamWindowClassifier}(x, y, n + (m + 1), r),$$

then

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m + 1 + r).$$

Proof. Unfold Definition 38.166 in both source windows and fix an arbitrary offset $k \le m + 1 + r$. The goal is the rational-history classifier at the joined endpoint $n + k$.

Split constructively between the case $k \le m$ and the complementary case in which $k = m+1+j$ for some $j \le r$. In the first case, apply the left source window directly at offset k. This yields

$$\mathsf{RatHistoryClassifier}(x(n + k), y(n + k)).$$

In the second case, apply the right source window at offset j. Its endpoint is

$$\mathsf{RatHistoryClassifier}(x((n + (m + 1)) + j), y((n + (m + 1)) + j)).$$

The arithmetic identity $(n + (m + 1)) + j = n + (m + 1 + j) = n + k$ identifies this endpoint with the selected joined endpoint.

Since every offset $k \le m + 1 + r$ is covered by exactly one of these bounded cases, the collected endpoint classifiers repack as $\mathsf{RealStreamWindowClassifier}(x, y, n, m + 1 + r)$. □

Theorem 38.183 (Real unary stream finite-window adjacent concatenation)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor histories* $a, w, r : \mathsf{Hist}$*, assume*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$$

and

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r).$$

Then

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r))).$$

Proof. Unfold Definition 38.154 in the two source windows. The left source supplies RatUp endpoint classifiers for every k with $\mathsf{UnaryOffsetLe}(k, w)$, and the right source supplies them for every successor-shifted offset below r from the anchor $\mathsf{append}(a, \mathsf{BHist.e1}(w))$.

Fix an offset k with $\mathsf{UnaryOffsetLe}(k, \mathsf{append}(w, \mathsf{BHist.e1}(r)))$. Unfold Definition 38.151; the offset ledger is a displayed Cont decomposition from k to the joined bound. Constructor inspection

of that finite append ledger separates the branch that reaches w from the branch that crosses the visible $\mathsf{BHist.e1}(w)$ successor.

In the left branch, repack the projected continuation as $\mathsf{UnaryOffsetLe}(k, w)$ and apply the left source window. This gives the RatUp classifier at $\mathsf{append}(a, k)$, which is the same endpoint selected by the joined window.

In the right branch, the ledger gives a unary offset j with $\mathsf{UnaryOffsetLe}(j, r)$ and an hsame row identifying $\mathsf{append}(a, k)$ with $\mathsf{append}(\mathsf{append}(a, \mathsf{BHist.e1}(w)), j)$. Apply the right source window at j and transport the RatUp endpoint classifier along that hsame comparison.

The unary rows for a, w, and r come from the two source classifiers and unary successor and append closure. Since the two ledger branches cover exactly the offsets below $\mathsf{append}(w, \mathsf{BHist.e1}(r))$, the collected endpoint classifiers repack as the joined $\mathsf{RealUnaryStreamWindowClassifier}$. □

Theorem 38.184 (Real unary window concatenation left projection)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor histories* $a, w, r : \mathsf{Hist}$*, if*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r))),$$

then

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w).$$

Proof. Unfold Definition 38.154 in the joined window. Its $\mathsf{UnaryHistory}$ rows include the anchor and joined bound, and its endpoint clause gives a $\mathsf{RatHistoryClassifier}$ for each offset below $\mathsf{append}(w, \mathsf{BHist.e1}(r))$.

Fix an offset k with $\mathsf{UnaryOffsetLe}(k, w)$. The left inclusion of the finite append ledger extends this Cont row to a displayed ledger

$$\mathsf{UnaryOffsetLe}(k, \mathsf{append}(w, \mathsf{BHist.e1}(r))).$$

Apply the joined classifier to this extended offset.

The endpoint remains anchored at a, so no endpoint transport is needed: the resulting $\mathsf{RatHistoryClassifier}$ is already the one required for the left window. Repacking these endpoint rows with the inherited $\mathsf{UnaryHistory}$ data gives $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. □

Theorem 38.185 (Real unary window concatenation right projection)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor histories* $a, w, r : \mathsf{Hist}$*, if*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r))),$$

then

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r).$$

Proof. Unfold Definition 38.154 in the joined window and fix an offset j with $\mathsf{UnaryOffsetLe}(j, r)$. The successor shift of this ledger gives a joined offset below $\mathsf{append}(w, \mathsf{BHist.e1}(r))$ by first crossing the visible $\mathsf{BHist.e1}(w)$ row and then following the displayed Cont row for j.

Apply the joined classifier to that shifted offset. Its endpoint is based at a and the shifted ledger, while the right window endpoint is based at $\mathsf{append}(a, \mathsf{BHist.e1}(w))$ and the offset j.

The displayed append associativity row supplies the required hsame and Cont transport between these two endpoint descriptions. Transport the $\mathsf{RatHistoryClassifier}$ along that row, and repack the transported endpoint classifiers with the inherited $\mathsf{UnaryHistory}$ data. This is exactly $\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r)$. □

Theorem 38.186 (Real unary adjacent window concatenation equivalence)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor and bound histories* $a, w, r : \mathsf{Hist}$*,*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r))) \iff \Big(\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)/$$

Proof. For the forward direction, assume the joined finite-window classifier. Apply Definition 38.184 to obtain the left window over the original anchor a. Apply Definition 38.185 to obtain the right window over the shifted anchor $\mathsf{append}(a, \mathsf{BHist.e1}(w))$. Pairing these two projections gives the displayed conjunction.

For the reverse direction, assume the two displayed window classifiers. Apply Definition 38.183 to concatenate the left window and the adjacent right window. The resulting classifier has anchor a and bound $\mathsf{append}(w, \mathsf{BHist.e1}(r))$, which is exactly the joined finite-window classifier. □

Theorem 38.187 (Real unary window associative triple concatenation)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor and bound histories* $a, w, r, q : \mathsf{Hist}$*, assume*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w),$$

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r),$$

and

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(\mathsf{append}(a, \mathsf{BHist.e1}(w)), \mathsf{BHist.e1}(r)), q).$$

Then

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q))).$$

Proof. First concatenate the w-window and the r-window by Definition 38.183. This gives a window from a with displayed bound $\mathsf{append}(w, \mathsf{BHist.e1}(r))$.

Next apply the same adjacent-concatenation theorem to that joined window and the q-window. The third source window is anchored at $\mathsf{append}(\mathsf{append}(a, \mathsf{BHist.e1}(w)), \mathsf{BHist.e1}(r))$, while the second application expects the anchor obtained by appending the joined bound to a.

The displayed associativity row for append supplies the needed hsame endpoint transport between these two anchor descriptions. Transport the RatUp endpoint classifiers along that hsame row, then repack the unary-history and Cont ledgers supplied by the two adjacent concatenations.

The resulting bound is exactly $\mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q))$, so the repackaged endpoint rows prove the displayed triple-window classifier. □

Theorem 38.188 (RealUp concatenated StreamName finite-window feed)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary histories* $a, w, r, q : \mathsf{Hist}$*, assume the three adjacent finite-window feeds*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w),$$

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r),$$

and

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(\mathsf{append}(a, \mathsf{BHist.e1}(w)), \mathsf{BHist.e1}(r)), q)$$

are accepted StreamName-scheduled Real finite-window feeds. Then the concatenated row

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q)))$$

is a single RealUp finite-window feed consumable by Definition 38.314 and Definition 126.289.

Proof. First apply Definition 38.187 to the three accepted adjacent windows. This produces one unary RealUnaryStreamWindowClassifier over the displayed concatenated bound, with all endpoint RatUp classifiers transported through the same hsame, unary-history, append, and finite Cont rows already present in the three source windows.

Next read the scheduling surface through Definition 126.289. The observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ are exactly the StreamName schedule consumed there, and the joined classifier is therefore a finite-window row of the kind passed to Definition 38.314.

The construction introduces no host Nat, quotient representative, completion object, or hidden stream object. Its whole feed is the single concatenated BHist window together with the displayed endpoint classifiers and the finite scheduling ledger. □

Theorem 38.189 (StreamName RegSeqRat RealUp shared finite-window cover)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and unary histories $a, w, r, q : \mathsf{Hist}$, suppose the three adjacent* BHist *pieces satisfy the public finite-window* $\mathsf{StreamName}^{\uparrow}$ *classifier rows of Definition 126.301, the* RegSeqRat *finite-window seal source coverage rows of Definition 38.268, and the* $\mathsf{Real}^{\uparrow}$ *finite-window feed rows of Definition 38.188. Then the concatenated finite window*

$$\mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q))$$

determines one shared public observation package for the three interfaces.

The package surface consists exactly of the StreamName finite-window classifier, the selected $\mathsf{Rat}^{\uparrow}$ *endpoint classifier rows, the finite* ProbeBundle *and* InBundle *ledger rows, the* UnaryHistory *rows, the displayed* hsame *endpoint comparisons, the shared* Cont *transports, and the finite dyadic source rows read by* RegSeqRat*. The* RegSeqRat *seal consumer and the* $\mathsf{Real}^{\uparrow}$ *finite-window feed therefore read the same bounded observations from the same concatenated* BHist *window.*

Proof. First apply Definition 38.188. The three adjacent StreamName-scheduled finite windows are repacked into the displayed concatenated BHist window, and the RealUp feed keeps the same endpoint classifiers, hsame transports, unary-history rows, and finite Cont ledger.

Next apply Definition 126.301. Its structural laws and selected readback rows identify the public StreamName finite-window surface carried by the concatenated window: the classifier, probe bundle, membership rows, unary observations, endpoint comparisons, and finite stability rows are all read from the same ledger.

Finally apply Definition 38.268. The RegSeqRat seal uses exactly the StreamName finite-window interface rows, the dyadic source ledger, and the RealUp finite-window exhaustion rows already exposed by the same concatenated window. Since both downstream consumers cite the same hsame, Cont, UnaryHistory, finite ledger, and endpoint readback rows, there is only one public observation package, with no quotient representative, completion object, host stream carrier, or extra regularity datum. □

Theorem 38.190 (StreamName RegSeqRat RealUp joined-window consumer coherence)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and three adjacent public* $\mathsf{StreamName}^{\uparrow}$ *finite windows over* BHist *anchors $a, w, r, q : \mathsf{Hist}$, let*

$$L = \mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q))$$

and let

$$R = \mathsf{append}(w, \mathsf{BHist.e1}(\mathsf{append}(r, \mathsf{BHist.e1}(q)))).$$

If the three source windows satisfy the public finite-window classifier rows of Definition 126.301, the RegSeqRat *finite-window seal source coverage rows of Definition 38.268, and the* $\mathsf{Real}^{\uparrow}$ *finite-window feed rows of Definition 38.188, then the left-associated joined window L and the right-associated joined window R induce the same public* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* RegSeqRat *finite-window seal source rows, and the same finite-window* $\mathsf{Real}^{\uparrow}$ *feed rows.*

The common consumer surface is exactly the shared BHist *anchor, the three source finite-window rows, their endpoint* $\mathsf{Rat}^{\uparrow}$ *classifiers, the finite* ProbeBundle *and* InBundle *rows, the* UnaryHistory *rows, the displayed* hsame *transports, and the finite* Cont *ledger. No* RegSeqRat *regularity witness,* $\mathsf{Real}^{\uparrow}$ *seal classifier, quotient row, or host index equality is imported from the* $\mathsf{StreamName}^{\uparrow}$ *side.*

Proof. First build the left-associated consumer package. Apply Definition 38.187 to the three adjacent windows, then apply Definition 38.188. This gives the $\mathsf{Real}^{\uparrow}$ feed over L with exactly the endpoint classifiers, UnaryHistory rows, hsame transports, and finite Cont ledger already present in the source windows.

Next apply Definition 38.189. The shared cover reads the StreamName interface, the RegSeqRat seal source rows, and the RealUp feed from that same joined finite window.

For the right-associated reading, use the visible associativity comparison between L and R for append. Transport the public StreamName finite-window rows through the displayed hsame comparison, leaving the three source endpoint classifiers and the finite ledger unchanged.

Finally apply Definition 126.88. Its overlap clause says that two downstream reads sharing the same StreamName finite window, endpoint readback rows, and finite ledger have one source package. Thus the RegSeqRat and RealUp consumers receive the same rows under the two parenthesizations, with no extra regularity, seal-classifier, quotient, or host-index data. □

Theorem 38.191 (RealUp StreamName RegSeqRat triple-window cocycle)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and four adjacent public* $\mathsf{StreamName}^{\uparrow}$ *finite windows over* BHist *anchors $a, w, r, q, u : \mathsf{Hist}$, assume each consecutive triple satisfies the hypotheses of Definition 38.189 and Definition 38.190. Then the two public observation routes obtained by first reassociating the left triple and then the right triple, or by first reassociating the right triple and then the left triple, induce the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* RegSeqRat *seal rows, the same* $\mathsf{Real}^{\uparrow}$ *feed rows, and the same* $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers.*

The common four-window consumer surface also keeps one ProbeBundle/InBundle *ledger, one family of* UnaryHistory *rows, the displayed* hsame *transports, the finite* Cont *ledger, and the* Pkg *provenance for the shared source package at each intermediate joined window. No* RegSeqRat *regularity witness,* $\mathsf{Real}^{\uparrow}$ *completeness row, quotient row, host index equality, or standard bridge data is imported.*

Proof. Start with the left triple. Apply Definition 38.190 to the three windows (w, r, q). The result is a single joined consumer package for the two parenthesizations of that triple, with the same StreamName rows, RegSeqRat seal rows, RealUp feed rows, endpoint classifiers, finite ledger, hsame transports, Cont rows, and Pkg provenance.

Use Definition 38.189 on that joined left package and the fourth window. The shared-window cover identifies the package seen by the next joined-window consumer as the same source package already selected by the left triple, now extended only by the adjacent public finite-window rows.

For the other route, first apply Definition 38.190 to the right triple (r, q, u). Then apply Definition 38.189 to attach the remaining left window. This produces the same kind of joined package, with the same visible rows and no additional source object.

Finally compare the two intermediate source packages. At the overlap, each was obtained by applying the same joined-window consumer coherence to one consecutive triple and the same

shared-window cover to the adjacent triple. Those two clauses identify the common source package at both steps, so the finite StreamName rows, RegSeqRat rows, RealUp rows, endpoint classifiers, bundle ledger, unary-history rows, hsame transports, Cont ledger, and Pkg provenance agree. Since neither theorem imports regularity, completeness, quotient, host-index, or bridge data, the four-window cocycle has exactly the stated negative boundary. □

Theorem 38.192 (RealUp StreamName RegSeqRat five-window pentagon coherence)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and five adjacent public* $\mathsf{StreamName}^{\uparrow}$ *finite windows over* BHist *anchors* $a, w, r, q, u, v : \mathsf{Hist}$*, assume every consecutive triple satisfies the hypotheses of Definition 38.189 and Definition 38.190. Then all five public observation routes obtained by reassociating adjacent triples induce the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* RegSeqRat *seal rows, the same* $\mathsf{Real}^{\uparrow}$ *feed rows, and the same* $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers.*

The common five-window consumer surface also has one ProbeBundle/InBundle *ledger, one family of* UnaryHistory *rows, the displayed* hsame *transports, the finite* Cont *ledger, and the* Pkg *provenance for the shared source package. No* RegSeqRat *regularity witness,* $\mathsf{Real}^{\uparrow}$ *completeness row, quotient row, host-index equality, or standard bridge data is imported.*

Proof. First apply Definition 38.191 to the left four windows (w, r, q, u). It identifies the two routes that move through the left and middle triples, preserving the StreamName rows, RegSeqRat seal rows, RealUp feed rows, endpoint classifiers, finite bundle ledger, unary-history rows, hsame transports, Cont ledger, and Pkg provenance.

Apply the same triple-window cocycle to the right four windows (r, q, u, v). This identifies the routes that move through the middle and right triples, again keeping exactly the same visible rows and finite ledgers.

It remains to compare the two descriptions on their common middle package. Use Definition 38.189 on the overlapping finite-window package. The cover states that the StreamName finite-window rows, the RegSeqRat seal source rows, and the RealUp feed rows are read from the same source package whenever the overlap presents the same endpoint classifiers, bundle rows, unary-history rows, hsame transports, and finite Cont ledger.

Combining the two cocycle comparisons with this shared-window identification connects all five parenthesized public routes around the pentagon. Projecting the listed rows from the common package gives the stated agreement for the StreamName, RegSeqRat, RealUp, RatUp, bundle, unary-history, hsame, Cont, and Pkg surfaces.

The proof uses only the finite-window cocycle and the shared-window cover. Since neither input exposes a regularity witness, completeness row, quotient row, host-index equality, or standard bridge datum, the five-window coherence has the same negative boundary. □

Theorem 38.193 (Real completion boundary unique observation route)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and adjacent unary* BHist *windows* $a, w, r, q : \mathsf{Hist}$ *satisfying the hypotheses of Definition 38.189 and Definition 38.190, every public Real-completion-phase observation read consumed by* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, and* $\mathsf{Real}^{\uparrow}$ *factors through the single concatenated* BHist *window*

$$\mathsf{append}(\mathsf{append}(w, \mathsf{BHist.e1}(r)), \mathsf{BHist.e1}(q)).$$

The visible route consists exactly of that joined window, the StreamName finite-window classifier rows, the RegSeqRat finite-window seal rows, the RealUp finite-window feed rows, the $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers,* UnaryHistory*,* hsame*,* Cont*,* ProbeBundle*,* InBundle*, and* Pkg *provenance.*

No quotient representative, host stream equality, independent regularity witness, RealUp completeness row, or direct RealUp read bypassing the StreamName–RegSeqRat handoff is available at this boundary.

Proof. Apply Definition 38.189. It identifies the common public package over the displayed concatenated BHist window, so the StreamName classifier, RegSeqRat seal source, RealUp feed, endpoint classifiers, bundle rows, unary-history rows, hsame comparisons, Cont transports, and Pkg provenance are read from one bounded package.

Next apply Definition 38.190. The coherence theorem transports that same package across the two visible parenthesizations of the joined window, leaving the downstream StreamName, RegSeqRat, and RealUp consumers attached to the same finite rows.

Project the listed rows from the shared package. Tuple exhaustion leaves no remaining coordinate that could carry a quotient representative, host stream equality, independent regularity witness, RealUp completeness row, or direct RealUp read outside the StreamName–RegSeqRat handoff. Thus the displayed finite route is the unique visible observation route at the Real completion boundary. □

The ten-field fit is: traditional target nested finite observation-window concatenation; BEDC source RealUp finite windows over StreamNameUp observation schedules $s, t : \mathsf{Hist} \to \mathsf{Hist}$; carrier unary BHist anchors and bounds; classifier RatUp endpoint classifiers at displayed append endpoints; stability by hsame endpoint transport; ledger Cont, unary-history rows, and append-associativity readback; core theorem Definition 38.187; exactness each endpoint in the nested bound is read from one of the three source windows; standard bridge finite-prefix consumption by RegSeqRat and RealUp surfaces; status ProofTarget.

The ten-field fit is: traditional target finite observation-window concatenation over unary histories; BEDC source RealUnaryStreamWindowClassifier; carrier RealUp finite windows over StreamNameUp observation schedules $s, t : \mathsf{Hist} \to \mathsf{Hist}$; classifier RatUp endpoint classifiers at displayed append endpoints; stability by unary offset splitting; ledger UnaryHistory, Cont, hsame, and the finite append boundary; core theorem Definition 38.183; exactness every joined endpoint is read from one source window; standard bridge finite stream-prefix concatenation; status ProofTarget.

The ten-field fit is: traditional target finite observation-window concatenation; BEDC source RealStreamWindowClassifier over RatUp endpoint classifiers; carrier rational-observation streams; classifier concatenated finite-window predicate; stability by bounded-offset splitting and endpoint reuse; ledger left radius, right radius, join index, and selected offset; core theorem Definition 38.182; exactness every endpoint in the joined window is covered by one of the two source windows; standard bridge finite rational observation intervals; status ProofTarget.

Definition 38.194 (Real stability certificate)**.** *The stability certificate consists of:*

1. *pointwise stream stability inherited from* $\mathsf{StreamName}^{\uparrow}$ *through Definition 38.273*
2. *transport of the* RegSeqRat *regularity witness along stream classification*
3. *classifier reflexivity, symmetry, and transitivity by dyadic tolerance comparison*
4. *compatibility of constant dyadic embedding with the closeness classifier*
5. *closure of real operations only when supplied as certified* RegSeqRat *transformations*

Definition 38.195 (RegSeqRat stream certificate spine)**.** *A* RegSeqRat *stream certificate spine for a rational-observation stream* $s : \mathsf{Hist} \to \mathsf{Hist}$ *is a rational stream-name carrier* $\mathsf{RatStreamNameCarrier}(s)$ *together with a regularity witness stable under* RealUnaryStreamClassifier*. Its ledger records the unary observation indices, the endpoint* $\mathsf{Rat}^{\uparrow}$ *carriers exposed at those indices, the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifiers read from the stream classifier, and the dyadic-tolerance requests used by the regularity witness.*

The ten-field fit is: traditional target Bishop regular rational sequence; BEDC source BHist, UnaryHistory, $\mathsf{Rat}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$*; carrier* RatStreamNameCarrier *plus the regularity witness; classifier* RealUnaryStreamClassifier*; stability pointwise RatUp reflexivity, symmetry, transitivity, and* hsame *transport; ledger finite observations and tolerance witnesses; core theorem list Definition 38.208 and Definition 38.209; exactness pointwise stream classification; standard bridge RealUp seal; status* ProofTarget.

Lean4 (definition) `BEDC.Derived.RealUp.RegseqratStreamCertificateSpine`

Theorem 38.196 (RegSeqRat carrier and classifier obligations)**.** *Let* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *be rational-observation streams. A* RegSeqRat *stream certificate spine for* s *and* t*, together with a recorded row* RealUnaryStreamClassifier(s, t)*, supplies the carrier and classifier obligations for the* RegSeqRat *interface: the* RatStreamNameCarrier *rows for the two streams, the* RealUnaryStreamClassifier *row as the stream classifier, endpoint* RatHistoryCarrier *rows at every certified unary observation, and the dyadic-tolerance ledger requests named by the regularity witness.*

Proof. Unfold Definition 38.195. The carrier field of each spine is already the required RatStreamNameCarrier row, and the displayed RealUnaryStreamClassifier(s, t) row is the classifier field used by the interface.

For endpoint carrier rows, apply Definition 38.208 to the carried streams and the displayed classifier row. Its endpoint clause returns RatHistoryCarrier$(s(n))$ and RatHistoryCarrier$(t(n))$ for each certified unary observation n.

The remaining ledger entries are exactly those named in the spine definition: the unary observation index, the two endpoint carriers, the pointwise RatUp classifier read from the stream classifier, and the dyadic-tolerance request handled by the regularity witness. □

Theorem 38.197 (RegSeqRat stability and transport obligations)**.** *Let* $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *be carried rational-observation streams with recorded rows* RealUnaryStreamClassifier(s, t) *and* RealUnaryStreamClassifier(t, u)*. If the endpoint transports are pointwise over unary observations, then the* RegSeqRat *obligation set contains reflexivity, symmetry, transitivity, and pointwise* hsame *transport for* RealUnaryStreamClassifier *over those carried streams.*

Proof. Apply Definition 38.273 to the three carried streams, the two displayed classifier rows, and the two pointwise hsame transport rows. The package gives the reflexive classifier on s, the reversed classifier from t to s, and the composite classifier from s to u.

For transport, use Definition 38.275. It repacks the pointwise RatUp transport data along certified unary observations as the transported StreamNameUp classifier row, so the same BEDC point classifiers serve as the RegSeqRat transport obligation. □

Theorem 38.198 (RegSeqRat constant stream and seal obligations)**.** *The* RegSeqRat *obligation set contains constant streams, independently reindexed constant streams, and the* $\mathsf{Real}^{\uparrow}$ *seal export interface. In particular, every carried RatUp point supplies a constant* RegSeqRat *stream spine, compatible RatUp point classifiers supply the independently reindexed constant-stream classifier row, and a sealed* $\mathsf{Real}^{\uparrow}$ *candidate exposing a recorded* RealUnaryStreamClassifier *row exports the classifier required by Definition 38.9.*

Proof. Apply Definition 38.205 to the carried RatUp point. This gives the constant stream carrier row, the reflexive unary stream classifier, and the constant dyadic-tolerance ledger.

Next apply Definition 38.206. The independently reindexed constant streams keep the same endpoint RatUp carrier rows and use the supplied RatUp point classifier as the pointwise stream classifier over every certified unary observation.

Finally apply Definition 38.209. The seal reads the recorded RealUnaryStreamClassifier row and exports exactly the RealUp observation classifier named by Definition 38.9. □

Theorem 38.199 (RegSeqRat carrier obligation package)**.** *For a rational-observation stream* $s : \mathsf{Hist} \to \mathsf{Hist}$*, the carrier obligation for the* RegSeqRat *stream certificate spine is exactly a* $\mathsf{RatStreamNameCarrier}(s)$ *row together with a regularity ledger over certified* UnaryHistory *observations. Each ledger row exposes the endpoint* $\mathsf{RatHistoryCarrier}(s(n))$ *at the certified observation* n *and records the dyadic-tolerance request used by the regularity witness.*

Proof. Unfold Definition 38.195. The carrier field is the displayed $\mathsf{RatStreamNameCarrier}(s)$ row, so projecting the spine gives the stream-name carrier without adding any separate carrier principle.

For a certified unary observation n, read the corresponding ledger row. That row contains the exposed $\mathsf{Rat}^{\uparrow}$ endpoint carrier $\mathsf{RatHistoryCarrier}(s(n))$ and the dyadic-tolerance request attached to the regularity witness at that observation.

Conversely, a stream-name carrier together with these certified ledger rows is precisely the data named by the spine definition. Repacking the carrier row and the ledger rows gives the carrier side of the RegSeqRat obligation package. □

Theorem 38.200 (RegSeqRat classifier obligation package)**.** *For rational-observation maps* $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, the* RegSeqRat *classifier obligation package consists of the reflexive, symmetric, transitive, endpoint-carrier, and pointwise* hsame *transport rows of* RealUnaryStreamClassifier*. These rows are supplied by combining Definition 38.273 with Definition 38.275 and then repacking the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows.*

Proof. Apply Definition 38.273 to the carried streams, the two displayed unary stream classifier rows, and the endpoint hsame transports. Its package gives the reflexive, symmetric, and transitive RealUnaryStreamClassifier rows, and it also exposes the endpoint RatHistoryCarrier rows at every certified unary observation.

Next apply Definition 38.275 to the same stream-name carriers, classifier row, and pointwise transports. This gives the transported $\mathsf{RatStreamNameClassifier}(s', t')$ row.

Both cited theorems unfold the same pointwise RatUp classifier over certified unary observations. Repacking those pointwise rows under Definition 38.195 gives exactly the classifier obligation package. □

Theorem 38.201 (RegSeqRat seal ledger obligation package)**.** *The constant-stream witness, stream classifier stability spine, and* $\mathsf{Real}^{\uparrow}$ *seal consumer jointly provide the remaining* RegSeqRat *obligation list: constant stream, transport, dyadic-tolerance ledger, and* $\mathsf{Real}^{\uparrow}$ *seal-interface obligations. Concretely, the package is obtained from Definition 38.205, Definition 38.208, and Definition 38.209.*

Proof. Use Definition 38.205 for the constant-stream obligation. It supplies the carried constant stream, the reflexive unary stream classifier, and the constant dyadic-tolerance ledger from a single $\mathsf{RatHistoryCarrier}(d)$ row.

Use Definition 38.208 for the transport obligation. It provides reflexivity, symmetry, transitivity, endpoint carrier exposure, and the transported stream-name classifier after pointwise hsame transport.

Finally use Definition 38.209 for the $\mathsf{Real}^{\uparrow}$ seal interface. The seal consumer projects the exposed streams and repackages the recorded stream comparison as the RealUp observation classifier required by Definition 38.9.

Together these three spines account for the constant stream, transport, dyadic-tolerance ledger, and seal-interface entries of the paper-side RegSeqRat obligation list. □

Theorem 38.202 (RegSeqRat carrier and regularity obligations)**.** *A* RegSeqRat *stream certificate spine consists of a* RatStreamNameCarrier *row, pointwise* RatHistoryCarrier *rows over certified* UnaryHistory *observations, a regularity ledger for dyadic-tolerance requests, and constant-stream inhabitation supplied by Definition 38.205.*

Proof. Unfold Definition 38.195. The carrier field is the displayed RatStreamNameCarrier row, and each certified unary observation exposes an endpoint $\mathsf{Rat}^{\uparrow}$ carrier row in the ledger.

The same definition records the dyadic-tolerance requests used by the regularity witness. For inhabitation, apply Definition 38.205 to any supplied $\mathsf{RatHistoryCarrier}(d)$. The theorem builds the constant rational stream, its self classifier, and the constant regularity ledger, which are exactly the listed carrier and regularity obligations. □

Theorem 38.203 (RegSeqRat classifier and transport obligations)**.** *The* RegSeqRat *classifier obligation supplies reflexivity, symmetry, transitivity, endpoint-carrier exposure, and pointwise* hsame *transport for* RealUnaryStreamClassifier *and* RatStreamNameClassifier*.*

Proof. Apply Definition 38.208. Its stability package supplies the reflexive, symmetric, and transitive RealUnaryStreamClassifier rows, and it exposes the endpoint RatHistoryCarrier rows at each certified unary observation.

For the transport row, use Definition 38.275. The pointwise hsame witnesses over unary observations transport the RatUp point classifiers and repack them as the RatStreamNameClassifier row. □

Theorem 38.204 (RegSeqRat seal-interface obligations)**.** *Sealed* $\mathsf{Real}^{\uparrow}$ *candidates exposing* RegSeqRat *stream certificate spines consume the recorded stream comparison as the RealUp classifier row. The ledger records only finite observations,* $\mathsf{Rat}^{\uparrow}$ *classifiers, dyadic-tolerance witnesses, and seal rows.*

Proof. Apply Definition 38.209. A sealed $\mathsf{Real}^{\uparrow}$ candidate exposing two rational stream names reads the recorded RealUnaryStreamClassifier comparison and exports that same row as the classifier required by the RealUp seal.

The allowed ledger entries are those named in Definition 38.279: finite stream observations, dyadic comparison witnesses, regularity certificates, and classifier comparison certificates. In the present seal interface these are precisely the finite observation rows, RatUp classifier rows, dyadic-tolerance witnesses, and seal rows used by the consumer theorem. □

Theorem 38.205 (RegSeqRat constant stream spine)**.** *For every history* $d : \mathsf{Hist}$*, a carrier* $\mathsf{RatHistoryCarrier}(d)$ *supplies a* RegSeqRat *stream certificate spine for* $\mathsf{RatConstStream}_d$*. The spine consists of the* $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d)$ *row, the self* $\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d, \mathsf{R}$ *row, and a constant regularity ledger: at any two certified unary observation indices, both exposed endpoints are the same* $\mathsf{Rat}^{\uparrow}$ *point* d*, and every dyadic-tolerance request is discharged by RatUp classifier reflexivity.*

Proof. Apply the carrier half of Definition 126.55 from right to left to the displayed $\mathsf{RatHistoryCarrier}(d)$. This gives the stream-name carrier row for $\mathsf{RatConstStream}_d$.

For the classifier row, unfold Definition 38.26. A certified unary observation n reads $\mathsf{RatConstStream}_d(n) = d$, so the required pointwise $\mathsf{RatHistoryClassifier}(\mathsf{RatConstStream}_d(n), \mathsf{RatConstStream}_d(n))$ is RatUp classifier reflexivity at the supplied carrier of d.

The regularity ledger is constant. Given any two certified unary observation indices, the point-exactness theorem identifies both exposed endpoints with the same RatUp point d. Thus the two observations have the same carrier, and each dyadic tolerance request compares d with itself, again by RatUp reflexivity. These rows are exactly the data named in Definition 38.195. □

Theorem 38.206 (RegSeqRat reindexed constant stream spine)**.** *For histories d, e : Hist and observation schedules $r, q : \mathsf{Hist} \to \mathsf{Hist}$, $\mathsf{RatHistoryCarrier}(d)$ supplies $\mathsf{RegSeqRat}$ stream certificate spines for both $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_d$. If $\mathsf{RatHistoryClassifier}(d, e)$ is supplied, then the two independently reindexed constant streams $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_e \circ q$ carry compatible $\mathsf{RegSeqRat}$ stream certificate spines and the classifier row $\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStrea}$ $r, \mathsf{RatConstStream}_e \circ q)$.*

Proof. First build the carried stream rows. Apply Definition 126.51 to the carrier row of d to obtain the stream-name carrier for $\mathsf{RatConstStream}_d \circ r$, and apply Definition 38.205 to the same carrier row to obtain the unreindexed constant-stream spine.

The regularity ledger for $\mathsf{RatConstStream}_d \circ r$ is the same constant ledger used in Definition 38.205. At any certified unary observation index, the schedule r is read before the constant stream is evaluated, and the exposed endpoint is still d. Thus every dyadic tolerance request again compares the RatUp point d with itself and is discharged by RatUp classifier reflexivity.

Now suppose $\mathsf{RatHistoryClassifier}(d, e)$ is supplied. Its endpoint carrier rows provide the corresponding constant regularity ledgers for the d-stream and the e-stream. The StreamNameUp classifier row follows from Definition 126.51, while Definition 38.29 reads the same point classifier as the RealUp unary-observation stream classifier for the two independently reindexed constant streams.

Combining these rows gives the two compatible $\mathsf{RegSeqRat}$ spines together with the displayed $\mathsf{RealUnaryStreamClassifier}$ row. No quotient, completion, or host real structure is used. □

Theorem 38.207 (RegSeqRat constant stream seal classifier package)**.** *Let X and Y be sealed $\mathsf{Real}^{\uparrow}$ candidates exposing $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_e \circ q$, respectively. If the two exposed streams carry the compatible $\mathsf{RegSeqRat}$ stream certificate spines supplied by Definition 38.206, and if the seal records the displayed row $\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ$ $q)$, then the seal exports the RealUp observation classifier required by Definition 38.9.*

Proof. Project the two exposed rational-observation streams from the $\mathsf{Real}^{\uparrow}$ seals. By hypothesis, the two streams already carry the $\mathsf{RegSeqRat}$ spines produced for the independently reindexed constant streams.

Use the classifier row supplied by Definition 38.206. It is exactly the $\mathsf{RealUnaryStreamClassifier}(\mathsf{RatConstStream}_d$ $r, \mathsf{RatConstStream}_e \circ q)$ row recorded by the seal.

Finally apply Definition 38.209 to the two spines and the displayed stream classifier. The conclusion is precisely the RealUp observation classifier named in Definition 38.9. □

Theorem 38.208 (RegSeqRat stream classifier stability spine)**.** *Let s, t, u, s', t' : $\mathsf{Hist} \to \mathsf{Hist}$ be rational-observation maps. Suppose*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(u),$$

$$\mathsf{RealUnaryStreamClassifier}(s, t), \quad \mathsf{RealUnaryStreamClassifier}(t, u),$$

and suppose the displayed endpoints transport pointwise over unary observations:

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then the RegSeqRat *stream certificate spine supplies* RealUnaryStreamClassifier(s, s), RealUnaryStreamClassifier(t, s), RealUnaryStreamClassifier(s, u), *the endpoint rows* RatHistoryCarrier$(s(n))$ *and* RatHistoryCarrier$(t(n))$ *at every certified unary observation* n, *and the transported stream-name classifier* RatStreamNameClassifier(s', t').

Proof. Unfold Definition 38.195, Definition 126.1, Definition 126.2, and Definition 38.26. The spine is therefore the existing BEDC unary stream row together with the regularity ledger; no separate classifier is being postulated.

Apply Definition 38.273 to the three carried streams, the two displayed RealUp unary stream classifiers, and the two pointwise hsame transport rows. Its reflexivity, symmetry, and transitivity clauses give the three RealUnaryStreamClassifier conclusions, and its endpoint clause gives the pointwise $\mathsf{Rat}^{\uparrow}$ carrier rows.

The same package returns RatStreamNameClassifier(s, t) and RealUnaryStreamClassifier(s', t'). Combining the transported point classifier with RatUp classifier hsame transport and the transported endpoint carrier rows repacks exactly RatStreamNameClassifier(s', t'). □

Theorem 38.209 (RegSeqRat seal supplies RealUp observation classifier)**.** *Let* X *and* Y *be sealed* $\mathsf{Real}^{\uparrow}$ *candidates exposing rational-observation streams* $s, t : \mathsf{Hist} \to \mathsf{Hist}$. *If the exposed streams satisfy the* RegSeqRat *stream certificate spine and the seal records* RealUnaryStreamClassifier(s, t) *as its stream comparison row, then the sealed candidates export exactly the* RealUnaryStreamClassifier(s, t) *row required by Definition 38.9.*

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratRealupSealConsumesStreamSpine`

Proof. Project the exposed stream data from the two $\mathsf{Real}^{\uparrow}$ seals. The RegSeqRat spine gives the RatStreamNameCarrier rows, the regularity ledger, and the pointwise RatUp classifier row over every certified unary observation.

Now read the seal's stored stream comparison row. By Definition 38.26, this is exactly the pointwise rational-history classifier over the same certified unary observations.

Repack that row under the RealUp seal. The result is the classifier row named by Definition 38.9; no quotient or completion principle is used. □

Theorem 38.210 (RegSeqRat stream spine scoped package)**.** *Let* $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *be rational-observation maps carrying* RatStreamNameCarrier(s), RatStreamNameCarrier(t), *and* RatStreamNameCarrier(u). *Suppose the stream comparison rows* RealUnaryStreamClassifier(s, t) *and* RealUnaryStreamClassifier(t, u) *are recorded, and suppose the endpoint transports are pointwise over unary observations:*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then the RegSeqRat *stream certificate spine supplies the following scoped package.*

1. *For any* $d : \mathsf{Hist}$ *with* RatHistoryCarrier(d), *the constant stream* $\mathsf{RatConstStream}_d$ *inhabits the* RegSeqRat *stream spine.*
2. *The classifier rows* RealUnaryStreamClassifier(s, s), RealUnaryStreamClassifier(t, s), *and* RealUnaryStreamClassifier *are available.*
3. *At every certified unary observation* n, *the endpoint carrier rows* RatHistoryCarrier$(s(n))$ *and* RatHistoryCarrier$(t(n))$ *are available.*
4. *The transported stream-name classifier* RatStreamNameClassifier(s', t') *is available.*

5. *For sealed* $\mathsf{Real}^{\uparrow}$ *candidates exposing* s *and* t*, whose seal records* $\mathsf{RealUnaryStreamClassifier}(s,t)$ *as the stream comparison row, the RealUp seal exports the row required by Definition 38.9.*

Proof. The constant-stream clause is exactly Definition 38.205. It turns any $\mathsf{RatHistoryCarrier}(d)$ into the carried constant stream, its reflexive unary stream classifier, and the constant regularity ledger.

For the stability clauses, apply Definition 38.208 to s, t, u, s', t', the three carried-stream rows, the two displayed classifier rows, and the two pointwise hsame transports. This gives the reflexive, symmetric, and transitive $\mathsf{RealUnaryStreamClassifier}$ rows, the endpoint $\mathsf{Rat}^{\uparrow}$ carrier rows at every certified unary observation, and the transported $\mathsf{RatStreamNameClassifier}(s', t')$ row.

Finally, apply Definition 38.209 to any sealed $\mathsf{Real}^{\uparrow}$ candidates exposing s and t with the recorded stream comparison. The seal consumer repacks that same comparison as the RealUp classifier export.

All rows are obtained by projection, pointwise hsame transport, and repacking of the listed certificate data. No quotient, completion, or host equality principle is used. □

Theorem 38.211 (RegSeqRat spine pointwise hsame transport)**.** *Let* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *be rational-observation maps. Suppose* s *and* t *carry* $\mathsf{RegSeqRat}$ *stream certificate spines, and suppose the stream comparison row* $\mathsf{RealUnaryStreamClassifier}(s,t)$ *is recorded. If the endpoint transports are pointwise over certified unary observations,*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)),$$

then the transported streams s' *and* t' *carry* $\mathsf{RegSeqRat}$ *stream certificate spines and export the transported* $\mathsf{RatStreamNameClassifier}(s', t')$*. Moreover, for sealed* $\mathsf{Real}^{\uparrow}$ *candidates exposing* s' *and* t'*, whose seal records the transported stream comparison row, the seal exports the corresponding RealUp observation classifier required by Definition 38.9.*

Proof. First read the endpoint data from the original spine. The scoped package in Definition 38.210 supplies the carrier rows, the reflexive, symmetric, and transitive classifier rows, and the dyadic ledger entries over certified unary observations.

Next transport the stream-name comparison. Apply Definition 38.275 to the recorded $\mathsf{RealUnaryStreamClassifier}$ row and to the two pointwise hsame transport families. It returns the transported $\mathsf{RatStreamNameClassifier}(s', t')$, including the transported endpoint carrier rows for s' and t'.

These transported carrier rows, together with the original dyadic requests read at the same certified unary observations, repack as the $\mathsf{RegSeqRat}$ stream certificate spines for s' and t'. The construction only changes the endpoint histories by the displayed hsame witnesses; the unary observation indices and tolerance ledger are unchanged.

Finally apply Definition 38.209 to sealed $\mathsf{Real}^{\uparrow}$ candidates exposing s' and t' with the transported stream comparison row. The seal consumer exports exactly the RealUp observation classifier named by Definition 38.9. □

Theorem 38.212 (RegSeqRat stream classifier transport composition)**.** *Let* $s, t, s', t', s'', t'' : \mathsf{Hist} \to \mathsf{Hist}$ *be rational-observation maps. Suppose* s *and* t *carry* $\mathsf{RegSeqRat}$ *stream certificate spines, and suppose the row* $\mathsf{RealUnaryStreamClassifier}(s,t)$ *is recorded. If the two successive endpoint-transport layers are pointwise over every certified unary observation,*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)),$$

and

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s'(n), s''(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t'(n), t''(n)),$$

then s'' and t'' carry RegSeqRat *stream certificate spines and export* $\mathsf{RatStreamNameClassifier}(s'', t'')$*. Moreover, any* $\mathsf{Real}^{\uparrow}$ *seal over s'' and t'' whose stream comparison row is the twice-transported row exports the RealUp observation classifier required by Definition 38.9.*

Proof. Apply Definition 38.211 to the original RegSeqRat stream certificate spines, the recorded $\mathsf{RealUnaryStreamClassifier}(s, t)$ row, and the first pointwise hsame transport layer. This produces transported spines for s' and t' and the row $\mathsf{RatStreamNameClassifier}(s', t')$.

Use those transported spines as the input data for a second application of Definition 38.211. The second pointwise hsame layer carries the endpoint histories from s', t' to s'', t'', so the conclusion gives RegSeqRat stream certificate spines for s'' and t'' and the row $\mathsf{RatStreamNameClassifier}(s'', t'')$.

Finally compose only the named endpoint rows that the two applications expose: the StreamName classifier row is the transported row returned by the second application, and the RealUp classifier row is exported by applying the same seal consumer to the twice-transported stream comparison. No additional quotient, completion, or host equality principle is introduced. □

Theorem 38.213 (RegSeqRat seal finite-window transport package)**.** *Let $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ be certified rational-observation maps, let $\mathcal{P}$ be a* ProbeBundle *finite window, let n be a bundled unary observation in $\mathcal{P}$, and let the selected transported endpoints be displayed as* $\mathsf{E1}(a)$ *and* $\mathsf{E1}(b)$*. Suppose s and t carry* RegSeqRat *stream certificate spines, the row* $\mathsf{RealUnaryStreamClassifier}(s, t)$ *is recorded, and the bundled pointwise transport rows*

$$\forall m \in \mathcal{P},\ \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(s(m), s'(m)), \quad \forall m \in \mathcal{P},\ \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(t(m), t'(m))$$

carry the selected source endpoints to the displayed endpoints $s'(n) = \mathsf{E1}(a)$ and $t'(n) = \mathsf{E1}(b)$. Then the $\mathsf{Real}^{\uparrow}$ *seal over s' and t' exports the finite-window public readback package consisting of the transported* $\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P})$ *row, the selected* $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ *row, the positive unary denominator rows for the displayed endpoints, the endpoint-shape exclusions, one-tail agreement, and the RealUp observation classifier row required by Definition 38.9.*

Proof. Apply Definition 38.211 to the RegSeqRat spines, the recorded stream comparison, and the two bundled pointwise hsame transport families. This gives the transported stream certificate spines for s' and t' and the transported stream-name classifier.

Restrict that transported stream classifier to the bundle $\mathcal{P}$ using the StreamName finite-window stability fields. The selected observation n is already bundled, so the restriction exposes the finite-window classifier row and the selected endpoint histories at $s'(n)$ and $t'(n)$.

Apply the transported selected full endpoint package to the displayed endpoints $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$. It supplies the selected RatHistoryClassifier row, the positive unary denominator rows, the endpoint-shape exclusions, and one-tail agreement for the same selected observation.

Finally repack these rows through the RealUp seal consumer. The consumer keeps the finite-window StreamName row, the selected endpoint ledger, and the transported stream classifier as the public readback surface, and exports the RealUp observation classifier row without adding quotient, completion, or host equality data. □

Fit. Traditional target: finite-window transported rational-stream readback for the real-number seal. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{RegSeqRat} + \mathsf{Real}^{\uparrow} + \mathsf{ProbeBundle}$. Carrier: BHist histories of certified unary rational observations. Classifier: transported RatStreamNameFiniteWindowClassifier

with the selected RatHistoryClassifier endpoint row. Stability: bundled pointwise hsame transport restricted to $\mathcal{P}$. Ledger: positive unary denominators, endpoint-shape exclusions, and one-tail agreement. Core theorems: Definition 38.211 and the StreamName transported finite-window endpoint package. Exactness: the exported rows are exactly the rows consumed by the RealUp seal. Standard bridge: none. Status: paper theorem awaiting Lean realization in BEDC.Derived.RealUp.

Theorem 38.214 (RegSeqRat carrier obligation)**.** *The carrier obligation for* RegSeqRat *is exactly the* RatStreamNameCarrier *row together with the regularity witness over* UnaryHistory *observations. Concretely, for a rational-observation map* $s : \mathsf{Hist} \to \mathsf{Hist}$*, a* RegSeqRat *carrier consists of* RatStreamNameCarrier(s) *and a ledger which, at each certified unary observation index* n*, exposes the endpoint* $\mathsf{Rat}^{\uparrow}$ *carrier row* RatHistoryCarrier$(s(n))$ *and the dyadic-tolerance witness demanded by the stream regularity check.*

Proof. Unfold the spine named in Definition 38.195. Its first component is the RatStreamNameCarrier row, and its ledger is indexed only by certified UnaryHistory observations.

The endpoint carrier rows are supplied by Definition 38.208; the constant case is supplied by Definition 38.205. The public package Definition 38.210 records these rows as the carrier face of the scoped spine.

The seal consumer Definition 38.209 uses no carrier data outside this spine. Thus the carrier obligation is precisely the named stream-name carrier plus the unary-observation regularity ledger. □

Theorem 38.215 (RegSeqRat classifier obligation)**.** *The classifier obligation for* RegSeqRat *is exactly* RealUnaryStreamClassifier *read with* RatStreamNameClassifier *endpoint recovery. For carried streams* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, a classifier row records the pointwise* $\mathsf{Rat}^{\uparrow}$ *comparison over every certified* UnaryHistory *observation and recovers the two endpoint stream-name carrier rows used by* RatStreamNameClassifier(s, t)*.*

Proof. Start from Definition 38.195. The spine does not introduce a separate classifier relation; it stores the RealUp unary stream classifier over the same BEDC unary observations.

Apply Definition 38.208. It reads the pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows from RealUnaryStreamClassifier, recovers the endpoint carrier rows, and repackages the transported data as a RatStreamNameClassifier row. The constant boundary is the reflexive case supplied by Definition 38.205.

The scoped package Definition 38.210 lists the reflexive, symmetric, transitive, and transported classifier rows together. The seal consumer Definition 38.209 confirms that the same row is the classifier exported through the $\mathsf{Real}^{\uparrow}$ interface. □

Theorem 38.216 (RegSeqRat carrier-classifier export package)**.** *For carried rational-observation streams* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, the* RegSeqRat *spine exports exactly the* RatStreamNameCarrier(s) *row, the* RatStreamNameCarrier(t) *row, the* RatStreamNameClassifier(s, t) *row, the* RealUnaryStreamClassifier(s, t) *row, the endpoint* RatHistoryCarrier *and* RatHistoryClassifier *rows over certified* UnaryHistory *observations, and the finite* DyadicRatCore *ledger rows later consumed by the* $\mathsf{Real}^{\uparrow}$ *seal.*

This export package contains no quotient representatives, completion data, metric completeness, $\mathsf{Real}^{\uparrow}$ *completeness, Classical data, or standard bridge data. Its public boundary is the carried stream-name surface, the endpoint RatUp observation surface, the finite dyadic ledger, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal consumption row.*

Proof. Read the carrier rows from Definition 38.195. Apply Definition 38.214 to the carried streams s and t. This supplies the two RatStreamNameCarrier rows and the endpoint RatHistoryCarrier rows indexed by certified UnaryHistory observations.

Read the classifier rows from the same spine. Apply Definition 38.215. It exposes the recorded $\mathsf{RealUnaryStreamClassifier}(s,t)$ row, recovers the pointwise endpoint $\mathsf{RatHistoryClassifier}$ rows, and repackages them as the $\mathsf{RatStreamNameClassifier}(s,t)$ row.

Next read the dyadic side. Apply Definition 38.256 to identify the finite $\mathsf{DyadicRatCore}$ ledger rows and their scoped primitive source. Then apply Definition 38.254 to consume those rows at the $\mathsf{Real}^{\uparrow}$ seal boundary.

The four cited obligations exhaust the displayed package. The carrier and classifier faces are read from the same certificate spine, while the dyadic ledger and the $\mathsf{Real}^{\uparrow}$ seal are consumed as already recorded rows. None of these steps adds a quotient representative, a completion object, a metric completeness principle, a $\mathsf{Real}^{\uparrow}$ completeness principle, Classical data, or standard bridge data as a primitive input. □

Theorem 38.217 (RegSeqRat export determines the RealUp seal source)**.** *For carried rational-observation streams $s,t : \mathsf{Hist} \to \mathsf{Hist}$, the export package of Definition 38.216 determines the exact source face read by the downstream $\mathsf{Real}^{\uparrow}$ seal. That source face consists precisely of the two $\mathsf{RatStreamNameCarrier}$ rows for s and t, the $\mathsf{RatStreamNameClassifier}(s,t)$ row, the pointwise $\mathsf{RealUnaryStreamClassifier}(s,t)$ row, the endpoint $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$ rows over certified $\mathsf{UnaryHistory}$ observations, and the finite $\mathsf{DyadicRatCore}$ ledger rows.*

No quotient representative, completion object, metric completeness principle, or $\mathsf{Real}^{\uparrow}$ completeness datum is part of this seal source. Consequently, any $\mathsf{Real}^{\uparrow}$ seal source read from the $\mathsf{RegSeqRat}$ package factors through the same visible carrier, classifier, endpoint, and dyadic ledger rows.

Proof. First apply Definition 38.216. It identifies the two stream-name carrier rows, the stream-name classifier row, the pointwise RealUnary classifier row, the endpoint RatUp carrier and classifier rows over certified unary observations, and the finite dyadic ledger rows as the complete exported surface.

Next apply Definition 38.256. This fixes the dyadic primitive source of the ledger rows as the scoped $\mathsf{DyadicRatCore}$ data, with no additional completion or quotient object hidden behind the ledger.

Finally apply Definition 38.254. The $\mathsf{Real}^{\uparrow}$ seal consumes the same dyadic ledger together with the same carried stream-name and endpoint observation rows. Since the preceding export package already lists all rows available at the $\mathsf{RegSeqRat}$ boundary, row exhaustion forces every seal-source read to factor through exactly those visible rows and through no separate $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 38.218 (RegSeqRat export finite-window row exhaustion)**.** *For carried rational-observation streams $s,t : \mathsf{Hist} \to \mathsf{Hist}$ and a certified finite $\mathsf{ProbeBundle}$ window, every finite-window row consumed by the downstream $\mathsf{Real}^{\uparrow}$ seal from the $\mathsf{RegSeqRat}$ export factors through exactly the listed $\mathsf{RatStreamNameCarrier}(s)$ and $\mathsf{RatStreamNameCarrier}(t)$ rows, the $\mathsf{RatStreamNameClassifier}(s,t)$ row, the $\mathsf{RealUnaryStreamClassifier}(s,t)$ row, the endpoint $\mathsf{Rat}^{\uparrow}$ carrier and classifier rows, the finite $\mathsf{DyadicRatCore}$ ledger, the displayed BHist rows, the hsame transports, the Cont continuation routes, the Pkg provenance, and the local $\mathsf{NameCert}$ rows.*

Consequently two certified windows which agree on these rows present the same finite-window consumer surface to $\mathsf{Real}^{\uparrow}$. The exhaustion statement exports no quotient representative, completed $\mathsf{Real}^{\uparrow}$ equality, metric-completeness datum, Classical datum, or standard bridge datum.

Proof. First apply Definition 38.216. It identifies the stream-name carrier rows, the stream-name classifier row, the pointwise RealUnary classifier row, the endpoint $\mathsf{Rat}^{\uparrow}$ rows, and the finite dyadic ledger as the complete $\mathsf{RegSeqRat}$ export package.

Next apply Definition 38.217. The downstream $\mathsf{Real}^{\uparrow}$ seal reads its source face from the same exported carrier, classifier, endpoint, and dyadic ledger rows, so it has no independent source coordinate outside the displayed package.

Finally apply Definition 38.228 to the certified finite ProbeBundle window. This restricts the seal read to the displayed BHist, hsame, Cont, Pkg, and local NameCert rows attached to that window.

If two certified windows agree on all listed rows, the three preceding identifications give the same projected consumer surface. Since no cited boundary contains a field for quotient representatives, completed $\mathsf{Real}^{\uparrow}$ equality, metric completeness, Classical data, or standard bridge data, those objects cannot be consumed by the finite-window seal. □

Definition 38.219 (DyadicRatCore observation row)**.** *A DyadicRatCore observation row for a BEDC history d is the package consisting of a* RatHistoryCarrier(d) *witness, a positive unary denominator row* PositiveUnaryDenominator(d)*, and the dyadic-radius token used by the* RegSeqRat *regularity check. Its classifier face is inherited from* RatHistoryClassifier*, together with* hsame *transport of the positive-denominator row.*

Lean4 (definition) `BEDC.Derived.RealUp.DyadicRatCoreObservationRow`

Theorem 38.220 (DyadicRatCore observation-row classifier transport)**.** *Let d, e, d', e' :* Hist*. Suppose d and e carry* DyadicRatCoreObservationRow *packages, suppose* RatHistoryClassifier(d, e) *holds, and suppose* hsame(d, d') *and* hsame(e, e') *hold. If the dyadic-radius tokens for d' and e' are the same certified unary-observation requests as the tokens for d and e, transported along those two* hsame *rows inside the* RegSeqRat *ledger, then d' and e' carry* DyadicRatCoreObservationRow *packages and* RatHistoryClassifier(d', e') *holds.*

Proof. Unpack Definition 38.219. The source rows at d and e consist of rational-history carriers, positive unary denominator rows, and dyadic-radius tokens attached to certified RegSeqRat unary observations.

Apply the RatUp classifier transport to RatHistoryClassifier(d, e) along hsame(d, d') and hsame(e, e'). This gives the transported classifier RatHistoryClassifier(d', e'), and the transported carrier rows are read from the same rational-history endpoint spine.

The denominator component follows by positive-denominator transport. Applying Definition 37.6 to the two hsame rows carries the positive unary denominator rows from d to d' and from e to e'.

Finally, the dyadic-radius component is ledger-attached to the same certified unary-observation requests in the RegSeqRat spine. Transporting the endpoint rows changes neither the request names nor the recorded dyadic-tolerance tokens, so no completion or quotient datum is introduced. Repacking the transported carrier, denominator, token, and classifier rows gives the stated DyadicRatCore observation-row transport. □

Lean4 ✓ `BEDC.Derived.RealUp.DyadicRatCoreObservationRow classifier transport`

The fit is as follows: traditional target dyadic rational observation transport; BEDC source RatUp plus RegSeqRat unary requests over BHist; carrier DyadicRatCoreObservationRow; classifier RatHistoryClassifier; stability transported denominator rows; ledger unchanged dyadic-radius request; core theorem Definition 38.220; exactness transported row plus classifier; standard bridge finite RealUp windows; status ProofTarget.

Theorem 38.221 (DyadicRatCore scoped source package)**.** *The DyadicRatCore source used by* RegSeqRat *is the concrete BEDC scoped source package over* Hist *whose carrier, classifier, stability,*

and ledger faces are all read from the rational-history endpoint spine. For each endpoint history d, the carrier face is RatHistoryCarrier(d) *together with* PositiveUnaryDenominator(d) *and the dyadic-radius token recorded by the* RegSeqRat *regularity request. For endpoints d, e, the classifier face is* RatHistoryClassifier(d, e) *together with* hsame *transport of the positive-denominator rows. The stability face is inherited from the* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier transports and from the pointwise classifier obligation of* RegSeqRat*. The ledger face is exactly the finite family of dyadic-tolerance requests already recorded in the* RegSeqRat *spine.*

Lean4 ✓ `BEDC.Derived.RealUp.DyadicRatCoreObservationRow scoped source package`

Proof. Start with the carrier face. Definition 38.219 names the three endpoint rows: the rational-history carrier, the positive unary denominator, and the dyadic-radius token. The endpoint carrier and the finite dyadic request are supplied by Definition 38.214, while Definition 37.19 converts the carrier row into the positive-denominator row.

Next read the classifier and stability faces. The pointwise stream classifier obligation in Definition 38.215 exposes the RatHistoryClassifier(d, e) endpoint comparison used by the DyadicRatCore source. RatUp carrier and classifier transport carries the rational-history rows along the same hsame comparison, and Definition 37.6 carries the positive-denominator row.

It remains only to identify the ledger. No request outside the RegSeqRat spine is used: the dyadic-radius token is attached to the same certified unary observation entry that supplies the endpoint rational row. Thus the source package has precisely the stated carrier, classifier, stability, and finite ledger faces over Hist. □

Theorem 38.222 (Phase Real dyadic stream RegSeq scope)**.** *Every DyadicRatCore request consumed by the Phase Real* RegSeqRat *spine is exactly a* DyadicRatCoreObservationRow *over* BHist*. Its carrier, positive-denominator, dyadic-radius, classifier, and ledger faces are read from the* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier transports together with the* RegSeqRat *unary observation request. No* $\mathsf{Real}^{\uparrow}$ *completeness row, quotient row, completion object, metric-completeness row, Classical datum, or standard bridge row is part of this DyadicRatCore source scope.*

Proof. Unpack Definition 38.219. A consumed endpoint row is the rational-history carrier, positive unary denominator, and dyadic-radius token attached to the certified unary observation entry of the RegSeqRat spine.

Apply Definition 38.221. Its carrier face is the endpoint $\mathsf{Rat}^{\uparrow}$ row, its classifier face is the RatHistoryClassifier comparison, and its stability face is the same hsame transport used by the RegSeqRat pointwise classifier obligation.

The remaining ledger component is the finite dyadic-tolerance request already named by the RegSeqRat carrier and classifier obligations. Repacking these faces gives exactly the Phase Real DyadicRatCore request scope, and there is no slot in that package for quotient, completion, metric, Classical, or $\mathsf{Real}^{\uparrow}$ completeness data. □

Theorem 38.223 (DyadicRatCore observation row hsame transport)**.** *Let d, e : Hist. If d has a DyadicRatCore observation row in the sense of Definition 38.219 and* hsame(d, e) *holds, then e has a DyadicRatCore observation row. Its* RatHistoryCarrier(e) *component is the* $\mathsf{Rat}^{\uparrow}$ *carrier transport of the carrier component at d, its* PositiveUnaryDenominator(e) *component is transported by Definition 37.6, and its dyadic-radius token is the same certified observation ledger entry of the* RegSeqRat *spine.*

Proof. Unpack the row named by Definition 38.219. It consists of a $\mathsf{RatHistoryCarrier}(d)$ row, a $\mathsf{PositiveUnaryDenominator}(d)$ row, and a dyadic-radius token attached to a certified unary observation entry in the $\mathsf{RegSeqRat}$ regularity ledger.

Transport the carrier face along $\mathsf{hsame}(d, e)$ using the $\mathsf{Rat}^{\uparrow}$ carrier transport already used in Definition 38.221. The compatibility between the rational-history carrier and the positive-denominator row is the one identified by Definition 37.19.

Now apply Definition 37.6 to carry the positive-denominator component from d to e. The endpoint rational row and the certified dyadic request are the rows recorded by Definition 38.214 inside the scoped spine of Definition 38.210.

The dyadic-radius token is not recomputed during transport: it remains attached to the same certified observation ledger entry used by the source package. Thus the transported carrier row, the transported positive-denominator row, and the unchanged ledger token repack as a DyadicRatCore observation row for e. □

Theorem 38.224 (DyadicRatCore observation source transport determinacy)**.** *Let $d, e : \mathsf{Hist}$, and suppose $\mathsf{hsame}(d, e)$ is displayed. Fix one certified $\mathsf{RegSeqRat}$ unary observation request whose source surface at d is a DyadicRatCore observation row. If two transported source packages are obtained at e by carrying that row along the same $\mathsf{hsame}(d, e)$ display through that same certified request, then the two packages expose the same $\mathsf{RatHistoryCarrier}(e)$ face, the same $\mathsf{PositiveUnaryDenominator}(e)$ face, the same dyadic-radius token, and the same $\mathsf{RatHistoryClassifier}$ face. No quotient row, completion row, metric-completeness row, Classical datum, or standard bridge row can separate the two transported source surfaces.*

Proof. Unpack Definition 38.219. The source row at d consists only of the rational-history carrier, the positive unary denominator row, and the dyadic-radius token attached to the named certified $\mathsf{RegSeqRat}$ unary observation request.

Apply Definition 38.223 to each transport along the displayed $\mathsf{hsame}(d, e)$ row. Both transported packages therefore read their $\mathsf{RatHistoryCarrier}(e)$ and $\mathsf{PositiveUnaryDenominator}(e)$ components from the same carrier transport and the same positive-denominator transport.

Next identify the public source faces by Definition 38.221. Its carrier, classifier, stability, and ledger faces are exactly the RatUp endpoint spine, the positive-denominator transport row, and the finite dyadic request ledger; there is no further source face available for a second transported package to choose.

For the classifier face, use Definition 38.220. Since both packages use the same displayed $\mathsf{hsame}(d, e)$ row and the same certified $\mathsf{RegSeqRat}$ request, the transported $\mathsf{RatHistoryClassifier}$ face is the same face in both packages.

The dyadic-radius token remains attached to the original certified unary observation ledger entry throughout the transport. Repacking these common carrier, denominator, token, and classifier faces gives determinacy of the transported source surface, and the excluded quotient, completion, metric, Classical, and bridge rows have no slot in the scoped source package. □

Theorem 38.225 (DyadicRatCore common-exponent transport determinacy)**.** *Let $d, e, d', e', r, s : \mathsf{Hist}$. Suppose d and e carry $\mathsf{DyadicRatCoreObservationRow}$ packages for the same certified $\mathsf{RegSeqRat}$ observation request, with r and s naming the unary-radius histories of that request. Suppose $\mathsf{RatHistoryClassifier}(d, e)$ holds, and suppose $\mathsf{hsame}(d, d')$ and $\mathsf{hsame}(e, e')$ are displayed. If the packages at d' and e' are obtained by transporting the source rows along those displayed hsame rows without changing the certified request ledger, then the two transported packages expose the same $\mathsf{Rat}^{\uparrow}$ carrier faces, the same positive-unary-denominator faces, the same unary radius/request token, and*

the same RatHistoryClassifier(d', e') *surface. No quotient row, completion row, metric-completeness row, Classical datum, or standard bridge row contributes an additional common-exponent choice.*

Proof. First transport the classifier surface. Apply Definition 38.220 to the source comparison RatHistoryClassifier(d, e) and to the displayed rows hsame(d, d') and hsame(e, e'). The transported endpoint comparison is therefore the single surface RatHistoryClassifier(d', e') attached to the same certified RegSeqRat request.

Next transport the two endpoint source rows. Apply Definition 38.223 to the row at d and to the row at e. Because the request ledger is unchanged, the transported rows inherit the same dyadic request token and the same positive-unary-denominator transport rows used by the source package.

Finally compare the transported packages with Definition 38.224. The theorem identifies the $\mathsf{Rat}^{\uparrow}$ carrier face, the positive-denominator face, the dyadic-radius token, and the classifier face for transports through a fixed certified request. These identified faces are exactly the common-exponent surface of the two presentations, and the scoped source package has no field for quotient, completion, metric-completeness, Classical, or standard bridge data. □

Theorem 38.226 (Dyadic observation transport RealUp feed determinacy)**.** *Let* d, e : Hist*, and suppose* hsame(d, e) *is displayed. Fix one certified* RegSeqRat *unary observation request whose source surface at* d *is a* DyadicRatCoreObservationRow*. If two RealUp finite-window feed packages are obtained by first transporting that source to* e *along the same* hsame(d, e) *row and then applying the DyadicRatCore public observation feed, then the two feed packages expose the same RatUp endpoint carrier, the same positive-denominator row, the same dyadic-radius ledger token, the same RealUp finite-window source face, and the same exclusion boundary. No quotient row, completion row, Classical datum, external metric primitive, RealUp completeness claim, or standard bridge row can distinguish the two feeds.*

Proof. Project the transported sources with Definition 38.224. Both source packages at e therefore expose the same RatUp endpoint carrier, the same positive-denominator row, the same dyadic-radius token, and the same RatHistoryClassifier face.

Feed these common source rows through Definition 38.264. The public observation feed consumes exactly the transported DyadicRatCore source, the finite dyadic request ledger, and the RealUp finite-window source face recorded by that ledger.

It remains only to compare the visible fields of the two feeds. Since the input source projections are common and the feed theorem has no slot for a second quotient, completion, Classical, metric, completeness, or bridge datum, field projection gives the same RatUp endpoint carrier, positive-denominator row, dyadic-radius token, RealUp finite-window source face, and exclusion boundary in both packages. □

The intended Lean target shape for this boundary is `BEDC.Derived.RealUp.DyadicRatCore_scoped_source_p`

Theorem 38.227 (DyadicRatCore carrier and classifier obligations)**.** *The carrier and classifier obligations for DyadicRatCore are the certificate boundary whose carrier row is* DyadicRatCoreObservationRow(d) *from Definition 38.219, whose classifier row is* RatHistoryClassifier(d, e) *transported through Definition 38.220, and whose source package is the scoped boundary of Definition 38.221. Concretely, for each endpoint history* d : Hist*, the carrier obligation reads exactly the* RatHistoryCarrier(d) *row, the* PositiveUnaryDenominator(d) *row, and the dyadic-radius token already recorded by the* RegSeqRat *ledger. For each classifier comparison* RatHistoryClassifier(d, e)*, the classifier obligation transports that three-field row along* hsame(d, e) *and repacks no additional completion, quotient, or external metric datum.*

Proof. Unpack the definition of the observation row. The carrier face is precisely the triple named in Definition 38.219: a RatUp endpoint carrier, a positive unary denominator row, and the dyadic-radius token from the regularity ledger.

Now use Definition 38.221. It identifies the source of every such row as the RatUp endpoint history together with the RegSeqRat observation request, so the carrier obligation has no independent object outside that scoped package.

For the classifier face, apply Definition 38.220. The RatHistoryClassifier comparison supplies the hsame transport of the endpoint carrier, denominator row, and dyadic token, which is exactly the DyadicRatCore classifier obligation. □

Theorem 38.228 (RealUp-RegSeqRat finite-window projection)**.** *A finite DyadicRatCore observation package over a certified unary request projects to the exact* RegSeqRat *finite-window observation surface consumed by* $\mathsf{Real}^{\uparrow}$*. The projected surface consists of the* $\mathsf{Rat}^{\uparrow}$ *endpoint carrier, the positive unary denominator row, the dyadic-radius token, the* $\mathsf{StreamName}^{\uparrow}$ *request name, the* RegSeqRat *regularity ledger,* hsame *transport for the endpoint and denominator rows, and finite ledger provenance for the displayed window. It exports no quotient carrier, no completion object, no metric-completeness row, no Classical representative, and no standard bridge datum.*

Proof. Project the finite package to its three visible faces. The package theorem Definition 38.231 identifies these faces as the carrier-classifier face, the finite arithmetic face, and the radius-distance face over the same certified unary request.

The endpoint rows are read from Definition 38.227. They give exactly the $\mathsf{Rat}^{\uparrow}$ carrier, positive unary denominator row, dyadic-radius token, and hsame transport required by the RegSeqRat finite window.

The radius and distance rows are read from Definition 38.230. They keep the dyadic-radius refinement, dyadic distance, absolute-value row, request name, and regularity ledger inside the same finite observation ledger.

Finally repack these shared rows as the RegSeqRat finite-window projection. Since the source package has no coordinate for a quotient carrier, completion object, metric-completeness witness, Classical representative, or standard bridge, the projection passed to $\mathsf{Real}^{\uparrow}$ contains none of those data. □

Theorem 38.229 (DyadicRatCore arithmetic-stability obligations)**.** *The arithmetic-stability obligations for DyadicRatCore are the finite certificate rows required to read dyadic rational endpoint arithmetic through the BEDC source package. They consist of operation rows for dyadic addition, negation, subtraction, multiplication, and positive-denominator division; classifier congruence rows showing that* RatHistoryClassifier *and* hsame *transport carry equivalent inputs to equivalent operation outputs; denominator-preservation rows showing that the* PositiveUnaryDenominator *face remains in the DyadicRatCore carrier after each operation; and finite ledger provenance rows tying every requested operation to the dyadic observation ledger consumed by the* RegSeqRat *regularity check. These are obligations for formal proof targets over the existing RatUp, StreamNameUp,* BHist*,* hsame*, and dyadic ledger interfaces; they do not assert arithmetic laws beyond that certificate inventory.*

Proof. Read every operation input as a DyadicRatCore observation row using Definition 38.227. This exposes the RatUp carrier, the positive denominator row, and the dyadic-radius token at each endpoint.

The operation row is then a finite BEDC request over the same endpoint histories. RatUp supplies the rational endpoint operation, StreamNameUp keeps the requested stream observation

name stable, and the dyadic ledger records the finite observation request consumed downstream by RegSeqRat.

Classifier congruence is the same transport pattern used for the carrier and classifier boundary. The RatHistoryClassifier comparison and hsame denominator transport carry equivalent inputs to the corresponding operation outputs, while denominator preservation keeps the result inside the DyadicRatCore observation-row format. □

Theorem 38.230 (DyadicRatCore radius, distance, and absolute-value obligations)**.** *The monotone dyadic-radius, distance, and absolute-value obligations for DyadicRatCore are the certificate rows consumed by* RegSeqRat *regularity and by the* $\mathsf{Real}^{\uparrow}$ *sealing boundary. The radius obligation states that strengthening a finite dyadic-radius request preserves the same RatUp endpoint carrier and transports only the ledger token. The distance obligation states that dyadic endpoint distance is read from two* DyadicRatCoreObservationRow *inputs, is stable under* RatHistoryClassifier *and* hsame *transport, and records its output in the same finite ledger. The absolute-value obligation is the unary instance of the same pattern: it reads one endpoint row, preserves positive-denominator and classifier compatibility, and exposes the output row required by the* RegSeqRat *regularity spine before* $\mathsf{Real}^{\uparrow}$ *sealing.*

Proof. Begin with the carrier and classifier package of Definition 38.227. It gives the endpoint rows and their RatHistoryClassifier transport inside BHist and hsame.

For radius monotonicity, refine the finite dyadic request in the ledger. The RatUp endpoint carrier and the positive-denominator face remain fixed, while the dyadic-radius token is replaced by the stronger certified request used by the RegSeqRat regularity spine.

For distance and absolute value, use the same operation-row shape as in Definition 38.229. The binary distance row and the unary absolute-value row read only DyadicRatCore carrier data, preserve classifier compatibility under hsame, and expose exactly the finite output rows consumed by the RealUp sealing interface. □

Theorem 38.231 (DyadicRatCore finite observation obligation package)**.** *A finite DyadicRatCore observation package over a certified unary request is exactly the conjunction of the carrier and classifier face, the arithmetic operation face, and the monotone-radius face already displayed in this inventory. Its first component is the* DyadicRatCoreObservationRow *carrier and classifier boundary of Definition 38.227. Its second component is the finite dyadic addition, negation, subtraction, multiplication, and positive-denominator division rows of Definition 38.229. Its third component is the monotone dyadic-radius, dyadic-distance, and absolute-value surface of Definition 38.230. All three components are read over the same* BHist *endpoints,* $\mathsf{Rat}^{\uparrow}$ *carrier rows,* $\mathsf{StreamName}^{\uparrow}$ *request name,* RegSeqRat *regularity ledger,* hsame *transport, and finite observation ledger data. The package has no coordinate for quotient carriers, completion objects, metric completeness, Classical data, or standard bridges.*

Proof. Project a candidate finite observation package to its carrier/classifier coordinate. By Definition 38.227, this coordinate is precisely the DyadicRatCoreObservationRow face over the RatUp endpoint, positive denominator, dyadic-radius token, BHist history, and hsame comparison rows.

Project the same package to its operation coordinate. By Definition 38.229, the only operation rows present are the finite dyadic arithmetic rows whose endpoints, classifier transports, denominator preservation, $\mathsf{StreamName}^{\uparrow}$ request name, and RegSeqRat ledger are the rows already exposed by the first coordinate.

Project the radius-distance coordinate. By Definition 38.230, this coordinate contains only monotone-radius refinement, dyadic distance, and absolute-value rows over the same finite ledger and the same endpoint classifier transport.

Conversely, given those three faces over the same histories, carriers, request name, regularity ledger, hsame transport, and finite observation ledger, repack them as the DyadicRatCore observation package. The repacking has no constructor input for a host quotient carrier, completion object, metric completeness witness, Classical datum, or standard bridge, so none of those objects is exported to RegSeqRat or $\mathsf{Real}^{\uparrow}$. □

Theorem 38.232 (DyadicRatCore distance obligation)**.** *For endpoints d, e : Hist, the DyadicRatCore distance obligation is the finite certificate package consisting of two* DyadicRatCoreObservationRow *witnesses, the displayed* RatHistoryClassifier(d, e) *face, the positive-denominator transports for both endpoints, and the dyadic-radius ledger token used by the* RegSeqRat *regularity request. Under* hsame *transport of both endpoints, the same package transports endpointwise to the corresponding target endpoints.*

Proof. Unpack Definition 38.219. Each endpoint row gives the rational-history carrier, the positive unary denominator, and the dyadic-radius token attached to a certified unary observation ledger entry.

The finite source of these rows is the scoped package of Definition 38.221. Its classifier face is the displayed RatHistoryClassifier(d, e) row, and its endpoint rows are exactly the two observation-row witnesses used by the distance request.

Transport the classifier face with Definition 38.220. Transport each endpoint row with Definition 38.223; the positive-denominator components are carried endpointwise by the same rows.

The dyadic-radius ledger token remains the certified token already recorded by the RegSeqRat regularity request. Thus the transported endpoint rows, transported classifier face, positive-denominator transports, and unchanged ledger token repack as the transported distance-obligation package, with no quotient or completion datum read. □

Theorem 38.233 (DyadicRatCore absolute-value obligation)**.** *For an endpoint d : Hist, the DyadicRatCore absolute-value obligation is the unary-radius face obtained from a single* DyadicRatCoreObservationRow(d). *It preserves the* PositiveUnaryDenominator(d) *component, records the same dyadic-radius request, and is stable under the* hsame *transport supplied by the scoped source package.*

Proof. Project the carrier, positive-denominator, and dyadic-radius components from Definition 38.219. These three projections are the unary-radius face consumed by the absolute-value request.

The positive-denominator component transports along Definition 37.6. The whole observation row transports along Definition 38.223, so the carrier and dyadic-radius components remain aligned with the transported endpoint.

Repack the transported row as the absolute-value obligation face. The scoped source package of Definition 38.221 supplies the hsame transport used here, and no additional arithmetic, quotient, or completion datum is introduced. □

Theorem 38.234 (DyadicRatCore positive-denominator boundary)**.** *For an endpoint d : Hist carrying* DyadicRatCoreObservationRow(d)*, the positive-denominator component supplies* UnaryHistory(d) *and excludes the endpoint shapes d*hsame*e, d*hsameE0(t)*, and d*hsameappend$(u, \mathsf{E0}(t))$ *for all histories u, t : Hist. If d*hsame*e is a displayed classifier endpoint transport, then the transported endpoint e carries the same unary-history witness and the same exclusions against* e*,* E0(t)*, and* append$(u, \mathsf{E0}(t))$*.*

Proof. Unpack Definition 38.219. Its denominator projection is the PositiveUnaryDenominator(d) row, so Definition 37.12 gives both the unary-history witness and the exclusion of $d$hsame*e*.

The zero-extension exclusion is exactly Definition 37.10 applied to the projected positive-denominator row. The appended zero-tail exclusion is Definition 37.15; composing a display $d\mathsf{hsameappend}(u, \mathsf{E0}(t))$ with denominator transport would produce the forbidden positive-denominator witness at the appended zero-tail.

Now let $d\mathsf{hsame}e$ be the displayed classifier transport. Transport the positive-denominator row along Definition 37.6, or equivalently transport the whole observation row along Definition 38.223 and project its denominator component. The three RatUp no-confusion facts applied at e give the same exclusions for the classified endpoint, and the result repacks as the DyadicRatCore endpoint-shape boundary. □

Theorem 38.235 (DyadicRatCore carrier obligation)**.** *The carrier obligation for DyadicRatCore is the ten-field certificate boundary whose traditional target is dyadic rational endpoint observation; whose BEDC source is* $\mathsf{Rat}^{\uparrow}$ *endpoint histories together with* PositiveUnaryDenominator *and the* RegSeqRat *dyadic-radius token; whose carrier is the DyadicRatCore observation row of Definition 38.219; whose classifier is* RatHistoryClassifier *together with* hsame *transport of the denominator row; whose stability face is inherited from the* $\mathsf{Rat}^{\uparrow}$ *endpoint carrier laws and the* RegSeqRat *carrier obligation; whose ledger is the finite dyadic-tolerance request already recorded in the stream spine; whose core theorem is the scoped source package of Definition 38.221; whose exactness clause states that no carrier datum outside the observation row is read; whose standard bridge is the finite RealUp observation window; and whose status is* ProofTarget*.*

Proof. Unpack the scoped source package in Definition 38.221. Its endpoint source is exactly a $\mathsf{Rat}^{\uparrow}$ history observation equipped with the positive unary denominator row and the dyadic-radius token from the certified RegSeqRat request.

Apply Definition 38.214. It supplies the endpoint RatHistoryCarrier row and the finite dyadic-tolerance ledger entry. The denominator component is the RatUp positive-denominator transport face already used by the scoped source package.

The FKernel and Cont unary context laws keep the request name and its finite ledger position fixed while the endpoint history is read. Thus the carrier face is precisely the existing DyadicRatCore observation row, with no extra completion or quotient object. □

Theorem 38.236 (DyadicRatCore classifier obligation)**.** *The classifier obligation for DyadicRatCore is the ten-field certificate boundary whose traditional target is dyadic endpoint comparison; whose BEDC source is the pair of* $\mathsf{Rat}^{\uparrow}$ *endpoint histories read from certified* RegSeqRat *unary observations; whose carrier is the pair of DyadicRatCore observation rows; whose classifier is* RatHistoryClassifier *plus* hsame *transport of* PositiveUnaryDenominator*; whose stability face is the endpointwise classifier transport of* RegSeqRat*; whose ledger is the same finite dyadic-tolerance request on both endpoints; whose core theorem is Definition 38.220; whose exactness clause says that the transported row is repacked from the same carrier, denominator, and token fields; whose standard bridge is the finite RealUp comparison window; and whose status is* ProofTarget*.*

Proof. Start from the classifier face named in Definition 38.221. The only comparison row there is RatHistoryClassifier on the endpoint histories.

Apply Definition 38.215. It recovers the pointwise endpoint comparison from the stream classifier and supplies the same certified unary observation positions on the two streams.

Positive-denominator transport carries the denominator rows along the hsame comparisons, and the FKernel unary context laws keep the finite dyadic request attached to the same observation name. Therefore Definition 38.220 repacks the transported endpoint data as the stated classifier obligation. □

Theorem 38.237 (DyadicRatCore arithmetic-stability obligation)**.** *The arithmetic-stability obligation for DyadicRatCore is the ten-field certificate boundary whose traditional target is closure of dyadic endpoint arithmetic under the rational endpoint interface; whose BEDC source is the* $\mathsf{Rat}^{\uparrow}$ *endpoint histories carrying* PositiveUnaryDenominator *rows and the* RegSeqRat *dyadic-radius token; whose carrier is the DyadicRatCore observation row for each arithmetic input and output endpoint; whose classifier is* RatHistoryClassifier *transported along* hsame*; whose stability face is the RatUp arithmetic context read through FKernel and* Cont *unary laws; whose ledger is the finite dyadic-tolerance request used by the stream regularity check; whose core theorem is Definition 38.221; whose exactness clause says that arithmetic stability uses only endpoint carrier rows, denominator transport, and the dyadic token; whose standard bridge is the finite RealUp arithmetic observation window; and whose status is* ProofTarget*.*

Proof. Read the arithmetic inputs through Definition 38.235. Each input endpoint exposes a rational-history carrier, a positive unary denominator row, and the dyadic-radius ledger token.

The RatUp arithmetic context supplies the endpoint rational operation inside the FKernel unary context. The Cont laws keep the requested unary observation name stable, so the output endpoint is recorded in the same finite dyadic-tolerance ledger format used by Definition 38.214.

Classifier compatibility is supplied by Definition 38.236. Its RatHistoryClassifier row and positive-denominator transport ensure that replacing any input by an hsame endpoint replaces the output by the corresponding transported DyadicRatCore row. □

Theorem 38.238 (DyadicRatCore monotone-radius obligation)**.** *The monotone-radius obligation for DyadicRatCore is the ten-field certificate boundary whose traditional target is preservation of certified dyadic tolerance when a requested unary radius is made stronger; whose BEDC source is the* $\mathsf{Rat}^{\uparrow}$ *endpoint history, the* PositiveUnaryDenominator *row, and the* RegSeqRat *dyadic-radius token; whose carrier is the DyadicRatCore observation row at the endpoint; whose classifier is* RatHistoryClassifier *with* hsame *denominator transport; whose stability face is the FKernel monotonicity of finite unary requests inside the* Cont *context; whose ledger is the finite dyadic-tolerance request ordered by the unary observation radius; whose core theorem is Definition 38.221; whose exactness clause says that strengthening the radius refines only the ledger token and not the endpoint carrier; whose standard bridge is the finite RealUp observation window; and whose status is* ProofTarget*.*

Proof. Begin with the endpoint row supplied by Definition 38.235. Its dyadic-radius token is a finite ledger entry of the RegSeqRat regularity spine, not an independent metric object.

Apply the FKernel finite-request monotonicity inside the Cont unary context. A stronger unary radius refines the same certified observation request and therefore preserves the endpoint $\mathsf{Rat}^{\uparrow}$ carrier and the positive-denominator row.

The classifier side is unchanged by the refinement. If the endpoint is moved along hsame, Definition 38.236 transports the carrier and denominator fields, while the refined dyadic token remains in the same finite ledger family. □

Theorem 38.239 (DyadicRatCore radius-token refinement stability)**.** *Let* $d : \mathsf{Hist}$ *be an endpoint carrying* DyadicRatCoreObservationRow(d)*. If a stronger certified dyadic-radius request refines the ledger token already attached to this row, then the refined request repacks as a* DyadicRatCoreObservationRow(d) *with the same* RatHistoryCarrier(d)*, the same* PositiveUnaryDenominator(d)*, and the same classifier transport behavior under* hsame*. The construction reads no quotient, completion, metric-completeness, or standard-bridge datum.*

Proof. Project the carrier, positive-denominator, and dyadic-radius token fields from Definition 38.219. The first two fields are endpoint data of the original row, while the third field is the finite ledger token that the stronger certified dyadic-radius request refines.

Use Definition 38.230 for radius monotonicity. It changes only the dyadic-radius ledger token: the endpoint $\mathsf{Rat}^{\uparrow}$ carrier and the positive-denominator row remain the source row already projected from $\mathsf{DyadicRatCoreObservationRow}(d)$.

For endpoint transport, apply Definition 38.223 to the whole row and Definition 37.6 to the denominator component. These two transport facts keep the classifier behavior under hsame aligned with the original source row.

Repack the refined dyadic-radius token with the unchanged carrier and denominator fields. The resulting package is again a $\mathsf{DyadicRatCoreObservationRow}(d)$ because its three defining fields are exactly the fields required by Definition 38.219. □

This fit is the local ten-field boundary for dyadic-radius strengthening: DyadicRatCore is the certificate surface, $\mathsf{Rat}^{\uparrow}$ supplies the endpoint carrier, RegSeqRat supplies the finite dyadic ledger, BHist is the endpoint history carrier, hsame is the only endpoint transport relation, the request name remains in the same unary context, the denominator witness is the same positive-unary row, the classifier face is the transported Rat history comparison, the exactness clause forbids quotient or completion data, and the downstream $\mathsf{Real}^{\uparrow}$ finite-window seal consumes the refined row without rebuilding carrier evidence.

Theorem 38.240 (DyadicRatCore distance and absolute-value obligation)**.** *The distance and absolute-value obligation for DyadicRatCore is the ten-field certificate boundary whose traditional target is endpoint stability of dyadic distance and absolute value; whose BEDC source is the pair of* $\mathsf{Rat}^{\uparrow}$ *endpoint histories with* PositiveUnaryDenominator *rows and the* RegSeqRat *dyadic-radius token; whose carrier is the DyadicRatCore observation row for each endpoint and for the distance or absolute-value output; whose classifier is* RatHistoryClassifier *transported by* hsame *with denominator transport; whose stability face is the RatUp unary context law for distance and absolute value inside FKernel and* Cont*; whose ledger is the finite dyadic-tolerance request; whose core theorem is Definition 38.221; whose exactness clause says that distance and absolute value read only the carrier row, classifier row, denominator transport, and dyadic token; whose standard bridge is the finite RealUp metric observation window; and whose status is* ProofTarget*.*

Proof. Use Definition 38.235 to expose the endpoint rows for the inputs. The RatUp source gives the rational-history carriers and positive-denominator rows, while the RegSeqRat spine gives the finite dyadic-tolerance token.

The FKernel and Cont unary context laws read distance and absolute value at the same certified observation level. Their output rows therefore have the same DyadicRatCore shape: rational endpoint carrier, positive denominator, and the requested dyadic-radius token.

For stability, apply Definition 38.236. The RatHistoryClassifier comparison and denominator transport carry input replacement along hsame, and the finite ledger token is preserved by Definition 38.238 when the metric request is read at a stronger unary radius. □

Theorem 38.241 (DyadicRatCore operation-stability obligation)**.** *The DyadicRatCore source exposes addition, negation, and multiplication operation-stability only as* RegSeqRat *ledger-preserving transformations over endpoints carrying DyadicRatCore observation rows. Concretely, if the endpoint histories supplied to one of these operations carry* DyadicRatCoreObservationRow *packages, then the resulting endpoint row is again a* DyadicRatCoreObservationRow *package whose carrier and classifier faces are read through* RatHistoryClassifier *transport and whose positive denominator face*

is transported through PositiveUnaryDenominator*. The operation row preserves the recorded dyadic-radius token of the* RegSeqRat *ledger; it does not introduce quotient, completion, or external metric data.*

Proof. Unpack the scoped source package of Definition 38.221. Its carrier face is the rational-history endpoint row together with the positive-denominator row and the dyadic-radius token recorded in the RegSeqRat ledger.

The operation rows are therefore read at the level of endpoint histories. Apply the RatHistoryClassifier transport face described in Definition 38.220 to carry the rational carrier and classifier data through the addition, negation, or multiplication row.

The denominator component is carried by Definition 37.6. Since the operation stays inside the same certified unary observation request, the dyadic-radius token remains the ledger token already recorded by Definition 38.221.

Repack the transported carrier, classifier, denominator, and ledger token as a DyadicRatCore observation row. The construction uses only the RegSeqRat spine and the RatUp endpoint transport, so no quotient or completion datum is part of the obligation. □

Theorem 38.242 (DyadicRatCore radius monotonicity obligation)**.** *The DyadicRatCore dyadic-radius ledger is ordered by certified* UnaryHistory *refinement. If a certified request in a* RegSeqRat *spine is refined to a stronger unary request, then the refined request keeps the endpoint* DyadicRatCoreObservationRow *visible, with the same RatUp carrier face, the transported positive-denominator face, and the refined dyadic-radius token recorded in the finite ledger bundle.*

Proof. Start from the finite ledger extraction theorem Definition 38.247. It extracts a DyadicRatCore observation row at each certified unary request of a finite RegSeqRat ledger bundle.

For the refined request, use the same finite-bundle induction. The empty bundle has no endpoint row. In the cons step, compare the displayed request with the refined request: the matching head request supplies the endpoint row, and the tail case is handled by the induction hypothesis.

The refined request is then repacked under the scoped source package of Definition 38.221. The carrier and classifier faces are unchanged RatUp endpoint faces, while Definition 37.6 carries the positive-denominator row along the same hsame transport.

The monotone radius claim is exactly the persistence of the dyadic-radius token inside this refined finite ledger. Thus every refined certified unary request still exposes a DyadicRatCore observation row at the endpoint. □

Definition 38.243 (DyadicRatCore radius monotone window packet)**.** *A DyadicRatCore radius monotone window packet is a finite* BHist *window in a* RegSeqRat *spine whose entries are certified* UnaryHistory *requests together with their endpoint* DyadicRatCoreObservationRow *rows, their* PositiveUnaryDenominator *rows, the pointwise* hsame *transports between matching request endpoints, and the* RegSeqRat *ledger token recording the dyadic-radius bound for the window. The packet contains only the RatUp carrier face, the visible denominator rows, the request-order refinement rows, and the ledger provenance needed by the finite completion seal.*

Theorem 38.244 (DyadicRatCore radius window transport)**.** *Let W be a DyadicRatCore radius monotone window packet. If every certified unary request in W is transported pointwise along* hsame *to a request with the same* RegSeqRat *ledger position, then the transported window is again a DyadicRatCore radius monotone window packet. The endpoint* DyadicRatCoreObservationRow*, positive-denominator row, and dyadic-radius ledger token are carried componentwise by the classifier transport for the RatUp endpoint and by the denominator transport.*

Proof. Induct over the finite BHist window. The empty window has no endpoint rows and transports to the empty packet.

In the cons step, apply Definition 38.220 to the head observation row and Definition 37.6 to the head denominator row. The pointwise request transport preserves the displayed ledger position, so the dyadic-radius token is the transported head token.

The induction hypothesis transports the tail packet. Repacking the transported head with the transported tail gives a finite window with the same carrier, denominator, request-order, and ledger fields as Definition 38.243. □

Theorem 38.245 (DyadicRatCore radius window RegSeqRat consumer)**.** *The packet of Definition 38.243 is exactly the finite-window DyadicRatCore face consumed by the* RegSeqRat-*to*-Real$^{\uparrow}$ *seal: a consumer may read the certified unary requests, RatUp endpoint observation rows, positive-denominator rows, pointwise* hsame *transports, and the dyadic-radius ledger token, and it may not read a quotient completion object or an ambient real-number equality from DyadicRatCore alone.*

Proof. Start from the finite ledger extraction theorem Definition 38.247. It exposes the same finite request rows and endpoint observation rows named in the packet definition.

Next use Definition 38.242 to identify the monotone dyadic-radius component under request refinement. The window transport theorem above shows that the same face is stable under the pointwise hsame transport used by downstream stream consumers.

Finally compare the fields with the RegSeqRat finite ledger bundle. The only exported rows are the request window, endpoint observation rows, denominator rows, classifier transports, and radius ledger token. Since none of these fields is a quotient completion object, the finite-window seal reads exactly the displayed packet and nothing outside it. □

Theorem 38.246 (RegSeqRat dyadic observation coverage)**.** *Let s : Hist $\to$ Hist carry a* RegSeqRat *stream certificate spine, and let n be a certified* UnaryHistory *observation index of that spine. Then the exposed endpoint $s(n)$ has a DyadicRatCore observation row in the sense of Definition 38.219. Moreover, if a stream classifier is transported pointwise along* hsame *to a target endpoint $s'(n)$, then the same pointwise transport carries this DyadicRatCore observation row to $s'(n)$.*

Proof. Unfold Definition 38.195. At the certified observation index n, Definition 38.214 supplies the endpoint RatHistoryCarrier($s(n)$) row and the dyadic-tolerance witness recorded in the regularity ledger.

The positive-denominator component is read from the RatUp carrier interface: Definition 37.19 converts the endpoint carrier row into PositiveUnaryDenominator($s(n)$). These three pieces are exactly the DyadicRatCore observation row of Definition 38.219.

For the transported endpoint, apply Definition 38.215 to recover the pointwise RatHistoryClassifier row carried by the stream classifier. Its hsame comparison transports the endpoint carrier/classifier face through the existing RatHistoryClassifier transport, and Definition 37.6 transports the positive-denominator row to $s'(n)$.

The dyadic-radius token is attached to the same certified observation request in the spine ledger, so no additional completion or quotient datum is needed. Thus the transported package is again a DyadicRatCore observation row. □

Theorem 38.247 (DyadicRatCore observation row finite-ledger extraction)**.** *Let s : Hist $\to$ Hist carry a* RegSeqRat *stream certificate spine. For every finite BEDC ledger bundle L of certified* UnaryHistory *observation requests in that spine, there is a finite family indexed by the entries $n \in L$ whose n-th member is a* DyadicRatCoreObservationRow($s(n)$) *in the sense of Definition 38.219. This*

family is read from the DyadicRatCore scoped source package of Definition 38.221; its carrier and classifier faces are precisely the RegSeqRat *carrier and classifier obligations of Definition 38.214 and Definition 38.215.*

Moreover, if a stream classifier transports the spine pointwise to a target stream $s' : \mathsf{Hist} \to \mathsf{Hist}$*, then the same finite family transports endpointwise: for each certified request* $n \in L$*, the transported endpoint row is a* $\mathsf{DyadicRatCoreObservationRow}(s'(n))$*.*

Proof. Induct on the finite ledger bundle L. The empty bundle has no certified observation entry, so the indexed family and its transported endpoint family are empty.

For a cons bundle, write the displayed head request as the certified UnaryHistory observation index n and write L' for the remaining finite bundle. Apply Definition 38.246 to the head request in the RegSeqRat spine. It supplies the row $\mathsf{DyadicRatCoreObservationRow}(s(n))$ by unpacking Definition 38.219, with the carrier datum supplied by Definition 38.214 and the classifier transport datum supplied by Definition 38.215.

The source package identification in Definition 38.221 says that no additional ledger face is used beyond these certified unary observation entries. Therefore the tail induction hypothesis extracts the finite family over L' from the same spine and transports it along the same pointwise classifier.

Adjoin the head row to the tail family. The transported head row is provided by the transported clause of Definition 38.246, and the transported tail is the induction hypothesis. Hence the cons bundle extracts and transports the claimed finite family endpointwise. □

Theorem 38.248 (RegSeqRat observation ledger dyadic endpoint coverage)**.** *Let* $s : \mathsf{Hist} \to \mathsf{Hist}$ *carry a* RegSeqRat *stream certificate spine, and let* L *be a finite BEDC ledger bundle of certified unary* RegSeqRat *observations. Every dyadic endpoint row visible to a downstream* $\mathsf{Real}^{\uparrow}$ *finite-window consumer is one of the rows extracted by Definition 38.247, transported endpointwise along the same pointwise classifier.*

Equivalently, the consumer-visible endpoint surface is exhausted by the finite family of DyadicRatCoreObservation *rows indexed by certified entries* $n \in L$*, together with their pointwise transported endpoint rows. No hidden selector, quotient representative, completion object, or host index can produce an additional dyadic endpoint row.*

Proof. Induct on the finite ledger bundle L. The empty bundle has no certified unary observation entry, so a downstream finite-window consumer has no dyadic endpoint row to read from that ledger.

For a cons bundle, write the head request as n and the tail as L'. Apply Definition 38.246 to the head. It gives the visible $\mathsf{DyadicRatCoreObservationRow}(s(n))$ and its pointwise transported endpoint row.

Apply Definition 38.247 to the tail ledger L'. The induction hypothesis says every endpoint row visible from L' is already in the extracted tail family and transported along the same pointwise classifier.

Adjoin the head row to the extracted tail family. Carrier projection leaves no projection for a hidden selector, quotient representative, completion object, or host index, so the displayed finite family exhausts the downstream consumer-visible dyadic endpoint rows. □

Theorem 38.249 (RegSeqRat constant and transport obligation)**.** *The constant and transport obligations for* RegSeqRat *are the inhabitation of the constant rational-observation stream and pointwise* hsame *transport of stream classifiers. For every* $d : \mathsf{Hist}$ *with* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatConstStream}_d$ *carries a* RegSeqRat *spine. If* s, t *are classified streams and* s', t' *are pointwise*

hsame *transports over certified unary observations, then the transported stream-name classifier* RatStreamNameClassifier(s', t') *is part of the same obligation set.*

Proof. Definition 38.195 identifies the constant case as a spine over a stream-name carrier plus a regularity ledger. Apply Definition 38.205 to the supplied RatHistoryCarrier(d); the resulting ledger is constant at every certified unary observation.

For transport, apply Definition 38.208 to the two endpoint hsame rows. It supplies the transported pointwise $\mathsf{Rat}^{\uparrow}$ classifiers and repackages them as RatStreamNameClassifier(s', t').

The scoped package Definition 38.210 places the constant and transport clauses in one certificate package, and Definition 38.209 shows that these clauses are sufficient for the downstream RealUp seal to consume the stream classifier. □

Theorem 38.250 (RegSeqRat public rational-stream interface)**.** *A* RegSeqRat *public rational-stream interface is exactly the package formed from Definition 38.246, Definition 38.247, Definition 38.249, Definition 38.276, and Definition 38.254. Its public template has the following ten fields:*

1. *the traditional target is the regular rational Cauchy stream spine;*
2. *the BEDC source is DyadicRatCore plus StreamNameUp plus RatUp over* Hist*;*
3. *the carrier is the* RegSeqRat *stream certificate spine;*
4. *the classifier is the pointwise* RealUnaryStreamClassifier *and* RatStreamNameClassifier *row;*
5. *stability is the family of pointwise* hsame *transport rows;*
6. *the ledger consists of finite unary observation requests and dyadic-tolerance tokens;*
7. *the core theorem list is the five cited blocks above;*
8. *exactness says the RealUp seal consumes exactly those finite rows;*
9. *no standard bridge datum is part of the interface;*
10. *the status is a paper-side public interface target.*

Proof. Collect the five cited packages in the displayed order. The first package supplies DyadicRatCore observation coverage at each certified unary request, the second extracts finite ledger families from any selected observation bundle, and the third supplies the constant stream and pointwise transport obligations.

For every selected observation, use Definition 38.247. It turns the finite unary ledger entries and dyadic-tolerance tokens into endpoint DyadicRatCore rows, and its transported clause carries the same rows along the chosen pointwise hsame comparison.

Next use Definition 38.276. It supplies the stream-name interface, the classifier transport row, and the endpoint carrier rows needed to read the RealUnaryStreamClassifier and RatStreamNameClassifier components pointwise.

Finally apply Definition 38.254. The seal obligation identifies the data consumed by RealUp with the same finite observation rows already extracted from the RegSeqRat ledger. Therefore the package exposes the stated public rational-stream interface without quotient, completion, metric-completeness, arithmetic/order, Classical, or standard bridge data. □

Theorem 38.251 (RegSeqRat finite-window seal exactness)**.** *Let* s : Hist $\to$ Hist *carry a* RegSeqRat *rational-observation stream certificate spine, and let* $\mathcal{P}$: ProbeBundle *be a certified finite bundle of unary observation requests. The finite observation rows accepted by the* RegSeqRat-*to*-Real$^{\uparrow}$ *seal*

over $\mathcal{P}$ are exactly the union of the DyadicRatCore endpoint observation-row family extracted by Definition 38.247 and the $\mathsf{StreamName}^{\uparrow}$ *finite-window rows exposed by Definition 126.301.*

Equivalently, a finite window feeds the seal precisely when its carrier, classifier, pointwise stability, dyadic-ledger, and stream-name finite-window rows are present in the public rational-stream interface of Definition 38.250. This exactness claim does not introduce quotient carriers, completion objects, $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, Classical data, or standard bridge status.*

Proof. Expose the public rational-stream package by Definition 38.250. Its carrier, classifier, stability, ledger, and stream-name fields are the only rows made available to the downstream seal.

For the forward direction, take a row accepted by the seal over $\mathcal{P}$. Apply Definition 38.254 to read it as a finite unary observation request with the recorded dyadic-tolerance token and endpoint $\mathsf{Rat}^{\uparrow}$ data. Then apply Definition 38.247 endpointwise to place the endpoint datum in the DyadicRatCore observation-row family, and use Definition 126.301 to read the same request as a stream-name finite-window classifier row.

For the reverse direction, suppose the finite window carries the displayed DyadicRatCore endpoint family together with the $\mathsf{StreamName}^{\uparrow}$ finite-window interface rows. The public package supplies the corresponding carrier, classifier, pointwise stability, and dyadic-ledger fields, so Definition 38.254 gives the seal consumer for those rows.

Finally apply Definition 38.268. It identifies the finite-window source package consumed by the seal with exactly the same DyadicRatCore, dyadic-ledger, and $\mathsf{StreamName}^{\uparrow}$ rows, so no further quotient, completion, metric, Classical, $\mathsf{Real}^{\uparrow}$ completeness, or bridge data can enter the seal interface. □

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratFiniteWindowSealExactness`

Theorem 38.252 (RegSeqRat finite-window seal overlap compatibility)**.** *Let s, t : $\mathsf{Hist} \to \mathsf{Hist}$ carry* RegSeqRat *rational-observation stream certificate spines. Let $\mathcal{P}, \mathcal{Q}$:* ProbeBundle *be certified finite observation windows, and let $\mathcal{R}$:* ProbeBundle *be a certified common sub-bundle of both $\mathcal{P}$ and $\mathcal{Q}$. Suppose the* RegSeqRat*-to-*$\mathsf{Real}^{\uparrow}$ *seal packages over $\mathcal{P}$ and $\mathcal{Q}$ agree after restriction to $\mathcal{R}$ on the StreamName finite-window classifier, the DyadicRatCore observation rows, the unary observation witnesses, and the dyadic ledger tokens.*

Then the two $\mathsf{Real}^{\uparrow}$ *seal readbacks restrict to the same consumer-facing package over $\mathcal{R}$. This common package contains only the finite-window classifier, the pointwise* RatHistoryClassifier *rows, the dyadic observation rows, the regularity ledger, and the* $\mathsf{Real}^{\uparrow}$ *observation classifier; it contains no completion object, quotient carrier, or host-index equality row.*

Proof. First restrict the two finite windows to the common sub-bundle $\mathcal{R}$. By Definition 126.178, the restricted StreamName handoff rows classify the same consumer-facing finite-window tuple over $\mathcal{R}$.

Now read each restricted seal with Definition 38.251. Each readback is therefore exhausted by the StreamName finite-window rows and the DyadicRatCore endpoint observation-row family over the same common bundle.

Identify the dyadic component through Definition 38.247. The assumed agreement of unary witnesses and dyadic ledger tokens gives the same endpoint observation rows on the two restricted packages.

Finally repack the identified rows using Definition 38.253. The resulting consumer-facing $\mathsf{Real}^{\uparrow}$ package over $\mathcal{R}$ is the same on both sides, and the cited exactness boundaries expose no completion, quotient, or host-index equality coordinate. □

Theorem 38.253 (Phase Real seal consumer exactness)**.** *Let s : Hist $\to$ Hist carry a* RegSeqRat *rational-observation stream spine, and let $\mathcal{P}$: ProbeBundle be a certified finite-window package. Every consumer-facing row of the* $\mathsf{Real}^{\uparrow}$ *seal over $\mathcal{P}$ factors through exactly four displayed surfaces: the DyadicRatCore endpoint observation family of Definition 38.219, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the* RegSeqRat *regularity ledger, and the* E1*-sealed* $\mathsf{Real}^{\uparrow}$ *observation row. Conversely, those four surfaces assemble the consumer-facing seal package for the same finite window.*

Thus the Phase Real seal consumer reads precisely the endpoint carrier, positive-denominator, dyadic-radius, stream-name finite-window, regularity ledger, and E1 *seal rows exposed by the certified spine. It does not read a quotient carrier, completion object, Classical choice row, metric-completeness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. For a consumed seal row, first apply Definition 38.251. The row is therefore drawn from the DyadicRatCore endpoint family and the $\mathsf{StreamName}^{\uparrow}$ finite-window classifier rows in the same certified window.

The DyadicRatCore component is the observation family named by Definition 38.219 and scoped by Definition 38.221. The stream-name component is supplied by the public rational-stream interface of Definition 38.250, which also carries the RegSeqRat regularity ledger and the E1 seal row consumed by $\mathsf{Real}^{\uparrow}$.

For the converse direction, start with exactly those four surfaces over $\mathcal{P}$. The public rational-stream interface packages the endpoint carrier, classifier, stability, and regularity rows; the finite-window seal exactness theorem then assembles them as the consumer-facing $\mathsf{Real}^{\uparrow}$ seal package. Since every cited surface is part of this finite BEDC ledger, no quotient, completion, Classical, metric-completeness, or ambient completeness row can be consumed. □

Theorem 38.254 (RegSeqRat dyadic ledger and RealUp seal obligation)**.** *The dyadic-ledger and* $\mathsf{Real}^{\uparrow}$ *seal obligation for* RegSeqRat *consists of finite unary observation indices, the endpoint* $\mathsf{Rat}^{\uparrow}$ *rows read at those indices, the dyadic-tolerance witnesses used by regularity, and the seal export of the recorded* RealUnaryStreamClassifier *row. Whenever sealed* $\mathsf{Real}^{\uparrow}$ *candidates expose streams satisfying the* RegSeqRat *spine and record their stream comparison row, the seal exports the RealUp observation classifier required by Definition 38.9.*

Lean4 ✓ `BEDC.Derived.RealUp.RegSeqRatDyadicLedger real seal obligation`

Proof. Unfold Definition 38.195. Its ledger is finite at the level of requested unary observations and dyadic-tolerance checks, and each ledger entry is backed by endpoint $\mathsf{Rat}^{\uparrow}$ carrier and classifier rows.

The constant ledger is supplied by Definition 38.205. The general endpoint and transport rows are supplied by Definition 38.208, while Definition 38.210 gathers the ledger, classifier, constant, and transport faces as one scoped certificate package.

Finally apply Definition 38.209. The seal reads the already recorded RealUnaryStreamClassifier row and exports exactly the observation classifier required by the RealUp boundary, without adding quotient or completion data. □

Theorem 38.255 (DyadicRatCore obligation boundary package)**.** *The DyadicRatCore obligation boundary consists exactly of the carrier, classifier, arithmetic-stability, monotone-radius, distance and absolute-value, operation-stability, finite-ledger extraction, and* RegSeqRat *seal faces listed in this section. Its carrier face is Definition 38.235; its classifier face is Definition 38.236; its dyadic arithmetic face is Definition 38.237; its monotone-radius face is Definition 38.238; its distance and absolute-value face is Definition 38.240; its operation stability face is Definition 38.241; its finite ledger extraction face is Definition 38.247; and its downstream seal face is Definition 38.254.*

No face in this package reads a quotient, a completion object, Classical data, or an external metric primitive. Every named face is read from BHist*,* hsame*, RatUp endpoint histories, StreamNameUp request names, the DyadicRatCore observation row, and the finite dyadic ledger consumed by* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the carrier and classifier faces first. Apply Definition 38.235 and Definition 38.236. These identify the endpoint rows, their hsame transport, and the RatUp positive-denominator source used by the observation row.

Next read the stability inventory. Apply Definition 38.237, Definition 38.238, Definition 38.240, and Definition 38.241. Together they cover the dyadic arithmetic, radius refinement, distance, absolute-value, and operation rows without adding a separate metric carrier.

Finally read the ledger and seal faces. Apply Definition 38.247 for the finite family of endpoint rows, and apply Definition 38.254 for the RegSeqRat ledger consumed by the $\mathsf{Real}^{\uparrow}$ seal.

The cited theorem list exhausts the displayed boundary: each clause is one of the named faces above, and the proofs of those faces read only BHist, hsame, endpoint histories, request names, observation rows, and finite ledger entries. Hence the package has no quotient, completion object, Classical datum, or external metric primitive as an input. □

Theorem 38.256 (DyadicRatCore scoped primitive dependency package)**.** *Each DyadicRatCore obligation face in Definition 38.255 is scoped over exactly* BHist*,* hsame*, the* $\mathsf{Rat}^{\uparrow}$ *endpoint carrier and classifier rows,* PositiveUnaryDenominator*,* $\mathsf{StreamName}^{\uparrow}$ *unary request names, the DyadicRatCore observation row, and the finite dyadic ledger consumed by* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$*. The carrier, classifier, arithmetic-stability, monotone-radius, distance and absolute-value, operation-stability, finite-ledger extraction, and* RegSeqRat *seal faces use no quotient, completion object, Classical datum, or external metric primitive.*

Proof. Start with the carrier and classifier faces. Apply Definition 38.235 and Definition 38.236. These faces read BHist endpoint histories, hsame transport, the $\mathsf{Rat}^{\uparrow}$ carrier and classifier rows, PositiveUnaryDenominator, and the DyadicRatCore observation row of Definition 38.219.

Next read the dyadic stability faces. Apply Definition 38.237, Definition 38.238, Definition 38.240, and Definition 38.241. Their operation inputs and outputs remain endpoint observation rows, while the radius, distance, and absolute-value clauses use only the positive-denominator and dyadic-radius ledger tokens already present in the scoped source package of Definition 38.221.

The finite-ledger face is supplied by Definition 38.247. It uses $\mathsf{StreamName}^{\uparrow}$ unary request names only to select finitely many endpoint observations, then exposes the same DyadicRatCore observation rows and dyadic ledger entries at those selected names.

Finally apply Definition 38.254. The RegSeqRat seal consumes the recorded stream spine and the finite dyadic ledger exported by the previous faces; the $\mathsf{Real}^{\uparrow}$ observation classifier is therefore read from the existing RealUnaryStreamClassifier row.

These four readings exhaust the face list of Definition 38.255. Every dependency is either a primitive BEDC history, an hsame transport, a $\mathsf{Rat}^{\uparrow}$ endpoint row, a unary request name, an observation-row field, or a finite ledger entry already licensed by the cited face theorem. No cited face introduces a quotient, completion object, Classical datum, or external metric primitive. □

Theorem 38.257 (DyadicRatCore scoped closure boundary)**.** *Every DyadicRatCore obligation face is scoped over RatUp endpoint histories,* PositiveUnaryDenominator*,* RegSeqRat *unary observation requests,* UnaryHistory*,* hsame *transport,* Cont*/FKernel unary context laws,* DyadicRatCoreObservationRow*, and the finite dyadic ledger consumed by the* RegSeqRat *seal. No quotient, completion object, Classical data, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge is part of this scope.*

Proof. Begin with the scoped source package Definition 38.221. It supplies the RatUp endpoint histories, the PositiveUnaryDenominator rows, the observation row, and the dyadic-radius token used by the later ledger.

Next enumerate the obligation faces. Apply Definition 38.235, Definition 38.236, Definition 38.241, Definition 38.238, and Definition 38.240. These faces read only the endpoint rows, UnaryHistory request names, hsame transport, and the Cont/FKernel unary context laws.

Finally read the finite ledger and downstream seal. Apply Definition 38.247 to extract the finite dyadic ledger from the observation row, and apply Definition 38.254 to show that the RegSeqRat seal consumes exactly that finite ledger.

The preceding list exhausts the named faces of Definition 38.255. Since each face has already been bound to the displayed primitive dependency chain, the scoped boundary contains no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.258 (DyadicRatCore public observation package)**.** *The public DyadicRatCore package exports exactly the observation-row carrier, classifier, arithmetic-stability, monotone-radius, distance and absolute-value, operation-stability, finite-ledger extraction, and* RegSeqRat *seal faces of Definition 38.255. Its primitive scope is fixed by Definition 38.256 and Definition 38.257. Thus the public package has no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. First read the face inventory from Definition 38.255. It lists the carrier, classifier, arithmetic-stability, monotone-radius, distance and absolute-value, operation-stability, finite-ledger extraction, and RegSeqRat seal faces as the whole DyadicRatCore boundary.

Next read the primitive dependency boundary from Definition 38.256. That theorem binds every listed face to BHist, hsame, the $\mathsf{Rat}^{\uparrow}$ endpoint rows, PositiveUnaryDenominator, $\mathsf{StreamName}^{\uparrow}$ request names, the DyadicRatCore observation row, and the finite dyadic ledger consumed by RegSeqRat and $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 38.257. Since the public observation package exports only the already listed faces over the already fixed primitive scope, it carries no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.259 (DyadicRatCore public observation hsame transport)**.** *Let* $d, d' : \mathsf{Hist}$*. If* $\mathsf{hsame}(d, d')$ *holds and* d *carries the public DyadicRatCore observation package of Definition 38.258, then the same public package supplies the transported DyadicRatCore observation row at* d'*. The transported row has the* hsame*-transported RatUp carrier face, the* hsame*-transported positive-denominator face, the same certified dyadic ledger entry, and the same no-quotient, no-Classical, no-completion boundary fixed by Definition 38.256 and Definition 38.257.*

Proof. First read the public export surface from Definition 38.258. It exposes only the DyadicRatCore observation-row carrier, classifier, stability, arithmetic, ledger, and seal faces over the scoped primitive package.

Next apply Definition 38.223 to the observation row at d and the comparison $\mathsf{hsame}(d, d')$. This transports the RatUp carrier component and the positive-denominator component to d', while keeping the certified dyadic ledger entry attached to the same RegSeqRat spine.

Finally read the primitive and closure boundaries from Definition 38.256 and Definition 38.257. Because the transport uses only the exported public faces and the already named ledger entry, the transported row inherits the same finite dyadic ledger and introduces no quotient, completion

object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Lean4 ✓ `BEDC.Derived.RealUp.DyadicRatCoreObservationRow public hsame transport`

Theorem 38.260 (Dyadic common-exponent radius monotone)**.** *Let d_0, d_1 : Hist be two public DyadicRatCore observation rows whose endpoints are RatUp rows over positive unary denominators. Then there is a shared finite* BHist *denominator and radius ledger for the pair. Its endpoint reads are obtained by RatUp transport,* hsame *comparison, and* Cont *computation rows, and its common radius is no larger than either displayed input radius in the monotone-radius face of Definition 38.255. The construction uses only the scoped DyadicRatCore source package of Definition 38.256 and the finite-ledger extraction face already exported there.*

Proof. Read the two observation rows through Definition 38.255. Each row supplies a RatUp endpoint, a positive unary denominator, the monotone-radius face, and a finite dyadic ledger entry.

Use the scoped primitive package of Definition 38.256 to compare the two denominator histories in BHist. The hsame and Cont rows give the finite denominator readback needed to place both endpoints over a single displayed exponent ledger.

Finally apply the monotone-radius obligation and the finite-ledger extraction face. They select the common displayed radius inside the existing DyadicRatCore ledger and certify that this radius is bounded by the two input radii, with no quotient, completion object, Classical datum, or external metric primitive. □

Theorem 38.261 (Dyadic window denominator common refinement)**.** *For any finite RealUp observation window whose entries are public DyadicRatCore observation rows, every displayed dyadic row can be replaced by the shared-denominator refinement of Definition 38.260. The resulting finite window has the same public observation package: the RatUp endpoint rows, positive-denominator witnesses,* hsame *transports,* Cont *computations, monotone-radius rows, finite dyadic ledger entries, and scoped boundary from Definition 38.257 are all retained as displayed* BHist *data.*

Proof. Proceed through the finite RealUp window ledger entry by entry. For a singleton entry, the displayed denominator is already a common denominator for that entry, and the public observation package is unchanged.

For the next displayed row, apply Definition 38.260 to the current common ledger and the incoming row. This produces a refined common denominator and a radius bounded by the two displayed radii, while transporting the RatUp endpoint rows and positive-denominator witnesses through the same hsame and Cont rows.

The finite-ledger extraction face then repacks the refined rows into the same public DyadicRatCore observation package. Since the package fields are the carrier, classifier, stability, monotone-radius, and finite-ledger rows already listed in Definition 38.255, replacing each row by its refinement changes no public observation boundary. □

Theorem 38.262 (RegSeqRat dyadic common-radius seal)**.** *The* RegSeqRat *seal consumes exactly the refined finite dyadic ledger of Definition 38.261. From that ledger it exports the same RealUp finite-window input as Definition 38.264: carried RatUp endpoints, positive unary denominators, common-radius rows,* hsame *and* Cont *provenance, and the scoped DyadicRatCore boundary fixed by Definition 38.256.*

Proof. First build the common finite ledger by Definition 38.261. The ledger already contains the displayed endpoints, denominator histories, radius comparisons, and transport rows over BHist.

Next apply the RegSeqRat dyadic ledger seal face named in Definition 38.255. Its input is the finite dyadic ledger, not an ambient completion object, so the refined common denominator and common-radius rows are exactly the rows it reads.

Finally use Definition 38.264 to expose the sealed ledger as the RealUp finite-window input. The primitive and closure boundaries ensure that the export adds no quotient, no Classical datum, no external metric primitive, and no RealUp completeness claim. □

Theorem 38.263 (DyadicRatCore transported public observations feed RealUp windows)**.** *Let* $d, d' : \mathsf{Hist}$*. Suppose* $\mathsf{hsame}(d, d')$ *holds and the public DyadicRatCore observation package of Definition 38.258 supplies the observation row at* d*. Then the* hsame*-transported public observation row at* d' *is available as the same RealUp finite-window exhaustion input. The transported row carries the same finite dyadic ledger, the same RatUp endpoint and positive-denominator faces transported along* hsame*, and the same no-quotient, no-completion, no-Classical, no-external-metric, no-*$\mathsf{Real}^{\uparrow}$ *completeness, and no-standard-bridge boundary fixed by Definition 38.256 and Definition 38.257.*

Proof. First apply Definition 38.259 to the row at d and the comparison $\mathsf{hsame}(d, d')$. This gives the public observation row at d' with the RatUp endpoint face and positive-denominator face transported along hsame, while keeping the certified finite dyadic ledger fixed.

Next apply Definition 38.264. That theorem reads the public DyadicRatCore observation package as exactly the carrier, classifier, stability, and finite-ledger source rows consumed by Definition 38.314. Applying it after transport therefore makes the row at d' the same RealUp finite-window input.

Finally read the primitive and closure boundaries from Definition 38.256 and Definition 38.257. Since the composite uses only the public transport row and the RealUp finite-window consumer already listed in those boundaries, it adds no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.264 (DyadicRatCore public observation feeds RealUp finite-window exhaustion)**.** *The public DyadicRatCore package of Definition 38.258, with scope fixed by Definition 38.256 and Definition 38.257, supplies exactly the carrier, classifier, stability, and finite-ledger source rows consumed by the RealUp finite-window exhaustion obligation of Definition 38.314. It adds no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. First read the public face list from Definition 38.258. The exported surface is the observation-row carrier, classifier, arithmetic-stability, monotone-radius, distance and absolute-value, operation-stability, finite-ledger extraction, and RegSeqRat seal inventory.

Next read the primitive scope from Definition 38.256 and Definition 38.257. These results bind the exported surface to BHist, hsame, the $\mathsf{Rat}^{\uparrow}$ endpoint rows, PositiveUnaryDenominator, $\mathsf{StreamName}^{\uparrow}$ request names, the DyadicRatCore observation row, and the finite dyadic ledger, while excluding quotients, completion objects, Classical data, external metric primitives, $\mathsf{Real}^{\uparrow}$ completeness claims, and standard bridges.

Finally apply Definition 38.314. Its finite-window field consumes the endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, positive unary denominators, nonempty unary endpoint rows, common e-one tail readback, and the DyadicRatCore source rows exposed through the RegSeqRat finite-window dyadic source package. Therefore the public DyadicRatCore observation package supplies exactly the carrier, classifier,

stability, and finite-ledger rows used at the RealUp finite-window boundary, with no extra object or bridge added. □

Lean4 ✓ `BEDC.Derived.RealUp.DyadicRatCoreObservationRow realup window feed`

Theorem 38.265 (DyadicRatCore public observation subwindow stability)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, j, r, m*, suppose*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m)$$

and $j + r \leq m$*. If the public DyadicRatCore package supplies the carrier, classifier, stability, and finite-ledger rows consumed by the RealUp finite-window exhaustion obligation in Definition 38.264, then the certified subwindow*

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, r)$$

produced by Definition 38.172 inherits the same DyadicRatCore observation-row carrier, classifier, stability, and finite dyadic ledger source rows through the $\mathsf{StreamName}^{\uparrow}$ *finite-window public interface of Definition 126.301. The inherited rows still feed Definition 38.314 and add no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. Apply Definition 38.172 to the certified source window with shift j and radius r. The bound $j + r \leq m$ gives the shifted finite-window classifier

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, r).$$

Next read the shifted window through the public finite-window interface of Definition 126.301. The interface changes only the selected finite anchor and radius; the $\mathsf{StreamName}^{\uparrow}$ request names, endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, and bounded finite-ledger rows remain the same rows already exposed by the public DyadicRatCore package.

Finally apply Definition 38.264 to identify the inherited carrier, classifier, stability, and finite-ledger source rows as the rows consumed by Definition 38.314. Since the construction uses only subwindow extraction, the public $\mathsf{StreamName}^{\uparrow}$ finite-window surface, and the DyadicRatCore public observation feed, it introduces no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.266 (RegSeqRat tail-shift regularity stability)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, a certified window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

and indices j, r *with* $j + r \leq m$*, define the tail-shifted streams by*

$$x_j(k) = x(n + j + k), \qquad y_j(k) = y(n + j + k).$$

Then the finite $\mathsf{RegSeqRat}$ *regularity surface for the shifted window* $(x_j, y_j, 0, r)$ *inherits the same endpoint* $\mathsf{DyadicRatCoreObservationRow}$ *carrier, classifier, stability, and finite dyadic ledger rows as the source window. These inherited rows are exactly the rows fed to Definition 38.314, and the tail-shifted surface adds no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. Apply Definition 38.172 to the certified window with shift j and radius r. The bound $j + r \leq m$ gives the shifted finite certificate at anchor $n + j$ and radius r, which is the zero-based certificate for the tail-shifted streams x_j and y_j at window $(0, r)$.

Next apply Definition 38.265. The subwindow keeps the public DyadicRatCore observation-row carrier, classifier, stability, and finite dyadic ledger rows fixed while changing only the finite anchor read by the stream certificate.

Finally read the shifted certificate through Definition 126.301 and close with Definition 38.314. The RealUp finite-window consumer therefore receives the same DyadicRatCore endpoint rows from the tail-shifted RegSeqRat surface that it receives from the original finite window, with no additional primitive or bridge. □

Theorem 38.267 (RegSeqRat tail-shift dyadic-tolerance stability)**.** *For regular rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a certified window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

and indices j, r with $\mathsf{UnaryHistory}(j)$ and $j + r \leq m$, define tail-shifted streams by

$$x_j(k) = x(n + j + k), \qquad y_j(k) = y(n + j + k).$$

If the source window classifies x and y through the displayed RegSeqRat *dyadic-tolerance ledger at every requested unary observation in the window, then the shifted window $(x_j, y_j, 0, r)$ classifies the shifted streams through the transported dyadic-tolerance window. The transported classifier uses only the same* UnaryHistory *request rows,* hsame *endpoint comparisons, finite dyadic ledger rows, and* Cont *provenance already exposed by the source* RegSeqRat *certificate.*

Proof. Project the source certificate into its regularity rows and its dyadic classifier rows. The regularity projection is the input used by Definition 38.266; the classifier projection is the finite list of dyadic-tolerance requests attached to the same unary observation window.

Reindex the finite unary window by the common tail shift j. The hypothesis $j + r \leq m$ identifies every shifted request $k \leq r$ with the source request $n + j + k$, and $\mathsf{UnaryHistory}(j)$ keeps the transported request inside the same unary-history ledger. No new endpoint row is generated; the shifted row is the source row read at the transported index.

Apply Definition 38.266 to keep both shifted streams inside the finite RegSeqRat carrier. Then repack the dyadic-tolerance rows through the same Cont ledger and the same hsame endpoint comparisons. The resulting packet is exactly the classifier for $(x_j, y_j, 0, r)$, with the tolerance window transported by the tail shift and no extra quotient, completion object, external metric primitive, or standard bridge. □

Theorem 38.268 (RegSeqRat seal finite-window source coverage)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a certified source window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

and indices j, r with $j + r \leq m$, the selected bounded subwindow

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j, r)$$

feeds the RegSeqRat *seal with the same finite-window source package as the source window. More precisely, the seal receives the same DyadicRatCore observation-row carrier, classifier, stability rows, finite dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface rows, and* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion rows already exposed at the source window. This coverage uses no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. First apply Definition 38.564 to the source window. This identifies the bounded RegSeqRat observation data with the scoped DyadicRatCore carrier, classifier, stability, and finite dyadic ledger faces at the selected finite window.

Next apply Definition 38.265 with the bound $j + r \leq m$. The subwindow certificate keeps the same DyadicRatCore observation-row carrier, classifier, stability rows, and finite dyadic ledger while changing only the finite anchor. Read that shifted anchor through Definition 126.301 to obtain the matching $\mathsf{StreamName}^{\uparrow}$ finite-window interface rows.

Finally apply Definition 38.264. The subwindow package is therefore exactly the finite-window source input consumed by Definition 38.314. Since every row is obtained from the source exhaustion theorem, the public subwindow stability theorem, the $\mathsf{StreamName}^{\uparrow}$ interface, and the RealUp feed theorem, the seal receives no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.269 (RegSeqRat nested subwindow source coverage)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, a certified source window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

and indices j, r, j', r' *with* $j + r \leq m$ *and* $j' + r' \leq r$*, the nested selected window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n + j + j', r')$$

feeds the RegSeqRat *seal with the same finite-window source package as the original source window. In particular, it exposes the same DyadicRatCore observation-row carrier, classifier, stability rows, finite dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface rows, and* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion rows. This nested coverage introduces no quotient, completion object, Classical datum, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness claim, or standard bridge.*

Proof. Apply Definition 38.265 to the source certificate with the bound $j + r \leq m$. This gives the selected package at anchor $n + j$ and radius r with the same DyadicRatCore observation-row carrier, classifier, stability rows, and finite dyadic ledger rows as the source window.

Apply the same theorem again to the shifted package, now using the bound $j' + r' \leq r$. The second extraction selects the window at anchor $n + j + j'$ and radius r', and it preserves exactly the same observation-row carrier, classifier, stability, and ledger rows because the first selected package already carries those public DyadicRatCore rows.

Finally apply Definition 38.268 and Definition 126.301. These repack the nested certificate as the same RegSeqRat finite-window source package consumed by the seal and by Definition 38.314, with only the finite anchor changed to $n+j+j'$ and with no quotient, completion object, Classical datum, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness claim, or standard bridge. □

Theorem 38.270 (RegSeqRat finite-window seal source exhaustion)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and any certified finite* RegSeqRat-*to*-$\mathsf{Real}^{\uparrow}$ *seal window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m),$$

the accepted source package of the seal is exactly the pair formed by the public $\mathsf{StreamName}^{\uparrow}$ *finite-window interface and the* RegSeqRat/DyadicRatCore *dyadic source-exhaustion rows. The two sides use the same* BHist *carrier,* UnaryHistory *request rows,* ProbeBundle *entries,* InBundle *membership rows,* hsame *comparisons,* Cont *provenance, and finite dyadic ledger rows. No quotient, completion object, Classical datum, metric-completeness witness,* $\mathsf{Real}^{\uparrow}$*-completeness witness, or bridge datum is part of the source package.*

Proof. For the projection direction, apply Definition 38.251. It identifies every seal read with the finite-window source rows carried by the certified window, so the seal source has no row outside the displayed finite packet.

Next apply Definition 126.301. The stream-name component of the projected source is exactly the public finite window interface, including the same unary request rows, probe entries, membership rows, BHist carrier, and hsame comparisons.

Apply Definition 38.268 and Definition 38.314. These identify the remaining projected rows with the RegSeqRat/DyadicRatCore dyadic source-exhaustion rows, the shared Cont provenance, and the finite dyadic ledger consumed by $\mathsf{Real}^{\uparrow}$.

Conversely, start from the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface together with the listed RegSeqRat/DyadicRatCore exhaustion rows. The coverage theorem repacks these rows as the finite-window source package for the seal, and the exactness theorem makes this repacked source the accepted seal source for the same certified window.

Because both readings use only the displayed interface rows, exhaustion rows, continuation provenance, and finite ledger entries, there is no remaining field from which a quotient, completion object, Classical datum, metric-completeness witness, $\mathsf{Real}^{\uparrow}$-completeness witness, or bridge datum could be projected. □

Theorem 38.271 (RegSeqRat finite-window seal composition)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a certified DyadicRatCore public observation row over a finite window* RealStreamWindowClassifier(x *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface rows, the* RegSeqRat *seal source-coverage row, and the* $\mathsf{Real}^{\uparrow}$ *finite-window feed compose to a single finite-window seal package over the same* BHist *carrier,* hsame *comparisons,* Cont *ledger, and finite dyadic observation rows. Every bounded subwindow and every nested bounded subwindow selected inside the source window is read through that same package. The composed seal asserts no quotient carrier, external metric primitive,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or standard-bridge status.*

Proof. Begin with Definition 38.264. It exposes the certified DyadicRatCore observation row as the finite dyadic ledger and RealUp consumer row attached to the displayed source window.

Apply Definition 38.265. The finite window may be restricted without changing the underlying BHist carrier, hsame comparisons, Cont ledger, classifier rows, or finite dyadic ledger rows.

Next use Definition 38.268. This repacks the subwindow observation row together with the $\mathsf{StreamName}^{\uparrow}$ finite-window interface as the source package consumed by the RegSeqRat seal and by the RealUp finite-window feed.

Finally apply Definition 38.269. Nested selected windows are obtained by repeating the same public subwindow extraction, so the seal package remains tied to the original finite dyadic ledger rows and the same BHist, hsame, and Cont data. Since every component is read from the cited finite-window rows, the construction introduces no quotient carrier, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness theorem, or standard-bridge status. □

Theorem 38.272 (DyadicRatCore public observation standard bridge boundary)**.** *The public DyadicRatCore observation package of Definition 38.258, together with* hsame *transport, subwindow stability,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, and the* $\mathsf{Real}^{\uparrow}$ *finite-window feed, supplies a conservative standard bridge from concrete* BHist *and* $\mathsf{Rat}^{\uparrow}$ *dyadic observation rows to the finite dyadic observation interface consumed by* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$*. The bridge uses only the displayed observation-row carrier, classifier, stability rows, finite dyadic ledger, stream-name finite-window interface, and RealUp finite-window consumer. It excludes quotient carriers, completion objects, external metric primitives,* $\mathsf{Real}^{\uparrow}$ *completeness, full DyadicRatCore arithmetic, and any machine-checked bridge status.*

Proof. First read the public surface from Definition 38.258. The exported rows are the concrete observation-row carrier, classifier, stability faces, finite dyadic ledger, and RegSeqRat seal interface over BHist, hsame, and the $\mathsf{Rat}^{\uparrow}$ endpoint histories.

Next apply Definition 38.259. The transport clause shows that replacing a dyadic observation row by an hsame-equivalent row preserves the same finite ledger and the same public observation interface. Apply Definition 38.265 to restrict the interface to any certified finite subwindow without changing the exposed row vocabulary.

Then use Definition 38.269. Its nested window coverage identifies the RegSeqRat seal input with the same finite dyadic source package after repeated subwindow selection, and it reads the needed $\mathsf{StreamName}^{\uparrow}$ rows through the public finite-window interface.

Finally apply Definition 38.264. The finite-window package delivered by the preceding steps is exactly the observation interface consumed by the RealUp finite-window exhaustion row. Since every step factors through the public DyadicRatCore package and its finite-window consumers, the bridge is conservative: it introduces no quotient carrier, completion object, external metric primitive, $\mathsf{Real}^{\uparrow}$ completeness, full DyadicRatCore arithmetic, or machine-checked bridge status. □

Definition 38.272 is a $\mathsf{Real}^{\uparrow}$ local observation-boundary lemma. It records how the public DyadicRatCore observation package, hsame transport, subwindow stability, $\mathsf{StreamName}^{\uparrow}$ finite-window rows, and the $\mathsf{Real}^{\uparrow}$ finite-window feed align the finite dyadic ledger with the RegSeqRat and $\mathsf{Real}^{\uparrow}$ consumer interfaces inside the carrier-obligation subtree.

The lemma does not assign a separate closure axis to $\mathsf{DyadicRatCore}^{\uparrow}$ from the $\mathsf{Real}^{\uparrow}$ certificate-boundary file. The authoritative $\mathsf{DyadicRatCore}^{\uparrow}$ status is the canonical closure block in the DyadicRatCore chapter, where the chapter records its object token, closure grade, verification grade, and standard-bridge status as a single objectwise commitment.

Theorem 38.273 (Real unary stream classifier stability package)**.** *For rational-observation maps* $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, suppose*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(u),$$

$$\mathsf{RealUnaryStreamClassifier}(s,t), \quad \mathsf{RealUnaryStreamClassifier}(t,u),$$

and suppose the endpoint transports are pointwise over unary observations:

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then the same BEDC unary observation rows supply the following stability package:

1. $\mathsf{RatStreamNameClassifier}(s,t)$.
2. $\mathsf{RealUnaryStreamClassifier}(s,s)$, $\mathsf{RealUnaryStreamClassifier}(t,s)$, *and* $\mathsf{RealUnaryStreamClassifier}(s,u)$.
3. *For every certified unary observation* n*, both endpoint RatUp carrier rows* $\mathsf{RatHistoryCarrier}(s(n))$ *and* $\mathsf{RatHistoryCarrier}(t(n))$.
4. $\mathsf{RealUnaryStreamClassifier}(s',t')$.

Proof. Unfold Definition 126.1, Definition 126.2, and Definition 38.26. After this unfolding, both StreamNameUp and RealUp classifier rows are the same pointwise RatUp classifier over histories n equipped with $\mathsf{UnaryHistory}(n)$.

The first conclusion is therefore just the displayed $\mathsf{RealUnaryStreamClassifier}(s,t)$ read under the StreamNameUp classifier name. For $\mathsf{RealUnaryStreamClassifier}(s,s)$, fix a certified observation

n; the carrier row for s gives $\mathsf{RatHistoryCarrier}(s(n))$, and RatUp classifier reflexivity gives the required point classifier at that observation.

For the reversed classifier $\mathsf{RealUnaryStreamClassifier}(t, s)$, read the point classifier from $\mathsf{RealUnaryStreamClassifier}$ and apply RatUp classifier symmetry. For $\mathsf{RealUnaryStreamClassifier}(s, u)$, read the two point classifiers from the assumptions at the same certified observation and compose them by RatUp classifier transitivity.

For endpoint carriers, fix a certified unary observation n and unpack the RatUp classifier obtained from $\mathsf{RealUnaryStreamClassifier}(s, t)$ at that observation. Its endpoint data are precisely $\mathsf{RatHistoryCarrier}(s(n))$ and $\mathsf{RatHistoryCarrier}(t(n))$.

For transport, fix the same certified observation n. Apply $\mathsf{RatHistoryClassifier_hsame_transport}$ to the point classifier from $\mathsf{RealUnaryStreamClassifier}(s, t)$ and to the two pointwise hsame witnesses from s, t to s', t'. Repacking the transported point classifier over all certified unary observations gives $\mathsf{RealUnaryStreamClassifier}(s', t')$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier stability package`

Theorem 38.274 (Real unary stream-name stability package)**.** *For rational-observation maps $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, the stability package of Definition 38.273 is also the StreamNameUp reading of the same unary-observation classifier data. In particular, under the same carrier, classifier, composability, and endpoint-transport hypotheses, one obtains*

$$\mathsf{RatStreamNameClassifier}(s, t)$$

together with the self, symmetry, transitivity, endpoint-carrier, and transport conclusions listed in Definition 38.273.

Proof. Apply Definition 38.273 to the given hypotheses. Its proof unfolds the RatStreamNameUp carrier and classifier rows and identifies them with the same pointwise RatUp classifier used by $\mathsf{RealUnaryStreamClassifier}$ over certified unary observations.

Thus the first component of the package is the StreamNameUp classifier $\mathsf{RatStreamNameClassifier}(s, t)$, while the remaining components are the same RealUp self, reversed, composed, endpoint-carrier, and transported classifier rows from the stability package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier streamName stability package`

Theorem 38.275 (Real unary stream-name transported classifier)**.** *For rational-observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RealUnaryStreamClassifier}(s, t),$$

and suppose the endpoint transports are pointwise over unary observations:

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then $\mathsf{RatStreamNameClassifier}(s', t')$.

Proof. Unfold Definition 126.1, Definition 126.2, and Definition 38.26. The $\mathsf{RealUnaryStreamClassifier}(s, t)$ row supplies the RatUp point classifier at each certified unary observation, while the two displayed carrier hypotheses supply the endpoint stream carriers before transport.

Fix a certified unary observation n with witness $\nu : \mathsf{UnaryHistory}(n)$. Apply Definition 37.54 to the RatUp point classifier read from $\mathsf{RealUnaryStreamClassifier}(s, t)$ at (n, ν) and to the two pointwise hsame witnesses from s, t to s', t'. This gives $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.

Read the endpoint carrier fields of this transported RatUp classifier. The left field gives $\mathsf{RatHistoryCarrier}(s'(n))$ and the right field gives $\mathsf{RatHistoryCarrier}(t'(n))$. Since (n, ν) was arbitrary, these fields build $\mathsf{RatStreamNameCarrier}(s')$ and $\mathsf{RatStreamNameCarrier}(t')$.

The same transported RatUp point classifier, again for arbitrary (n, ν), supplies the pointwise classifier row of $\mathsf{RatStreamNameClassifier}(s', t')$. Repack that pointwise row with the two transported stream carriers to obtain the claimed StreamNameUp classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier streamName transport`

Theorem 38.276 (RegSeqRat stream-name scoped interface)**.** *For rational-observation maps* $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, suppose*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(u),$$

$$\mathsf{RealUnaryStreamClassifier}(s, t), \quad \mathsf{RealUnaryStreamClassifier}(t, u),$$

and suppose the endpoint transports are pointwise over unary observations:

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then these rows determine a scoped RegSeqRat *interface package containing:*

1. $\mathsf{RatStreamNameClassifier}(s, t)$.
2. $\mathsf{RealUnaryStreamClassifier}(s, s)$, $\mathsf{RealUnaryStreamClassifier}(t, s)$, *and* $\mathsf{RealUnaryStreamClassifier}(s, u)$.
3. *For every certified unary observation* n*, endpoint* $\mathsf{Rat}^{\uparrow}$ *carrier rows for* $s(n)$ *and* $t(n)$.
4. $\mathsf{RealUnaryStreamClassifier}(s', t')$.
5. $\mathsf{RatStreamNameClassifier}(s', t')$.

Proof. Apply Theorem 38.273 to the three stream-name carrier rows, the two RealUp unary classifier rows, and the two pointwise hsame transport rows. Its conclusions supply the StreamNameUp classifier for (s, t), the reflexive, reversed, and composed RealUp unary classifier rows, the endpoint $\mathsf{Rat}^{\uparrow}$ carrier rows at every certified unary observation, and the transported RealUp unary classifier for (s', t').

It remains to obtain the transported StreamNameUp classifier. Apply Theorem 38.275 with the same carrier rows for s and t, the same $\mathsf{RealUnaryStreamClassifier}(s, t)$ row, and the same pointwise hsame transports from s, t to s', t'. This yields $\mathsf{RatStreamNameClassifier}(s', t')$, which is the final row of the scoped RegSeqRat interface package. □

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratStreamnameScopedInterface`

Theorem 38.277 (RegSeqRat finite-window seal coverage)**.** *Let* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *be certified rational stream names and let* $W : \mathsf{ProbeBundle}$ *be a finite unary-observation window. The rows exported by Definition 126.301, the scoped* RegSeqRat *interface of Definition 38.276, and the DyadicRatCore observation rows assemble the finite-window seal side consumed by* $\mathsf{Real}^{\uparrow}$*. The assembled package records the bundled unary observations of* W*, the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows for* s *and* t*, the dyadic observation witnesses, and the* RegSeqRat *spine ledger. It claims no* $\mathsf{Real}^{\uparrow}$ *completeness, quotient carrier, arithmetic closure, order closure, metric completion, or standard bridge.*

Proof. First project the finite classifier rows from the StreamNameUp finite-window interface. For every bundled unary observation in W, Definition 126.301 supplies the carried stream-name endpoints and the pointwise $\mathsf{Rat}^{\uparrow}$ classifier row visible at that observation.

Next apply Definition 38.276. Its scoped package recovers the stream-name transport rows, the pointwise endpoint carrier rows, the transported StreamNameUp classifier, and the RegSeqRat spine rows over the same certified names.

Finally attach the DyadicRatCore observation witnesses as the seal ledger. These witnesses are indexed by the same finite window W, so the bundled unary observations, pointwise classifier rows, dyadic observations, and spine ledger repack as exactly the finite boundary consumed by the later $\mathsf{Real}^{\uparrow}$ seal theorem. □

Theorem 38.278 (RegSeqRat finite-window RealUp seal interface)**.** *Let $s, t : \mathsf{Hist} \to \mathsf{Hist}$ be certified rational stream names and let W : ProbeBundle be a finite unary-observation window. The StreamNameUp finite-window interface, the scoped* RegSeqRat *interface, the DyadicRatCore observation rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal rows assemble one* BHist*-anchored* $\mathsf{Real}^{\uparrow}$ *finite-window seal package. Its carrier row is* RatStreamNameCarrier *for s and t; its classifier rows are* RatStreamNameClassifier *and the finite-window classifier over W; its observation face is the finite family of dyadic comparison rows; its spine ledger is the* RegSeqRat *stream-name ledger; and its consumer face is the public* $\mathsf{Real}^{\uparrow}$ *seal row. The package asserts no completeness theorem, quotient carrier, arithmetic closure, order closure, metric completion, Classical data, or trichotomy.*

Proof. First read the finite observation boundary. By Definition 126.301, the certified stream names s and t supply the StreamNameUp carrier rows, the RatStreamNameClassifier row, and the finite-window classifier row for every bundled unary observation in W.

Next read the scoped regular-sequence spine. Theorem 38.276 packages the same stream-name rows as the RegSeqRat interface, while Definition 38.277 attaches the dyadic observation witnesses and the finite spine ledger over the same window.

Finally consume the public seal. Theorem 38.286 identifies the $\mathsf{Real}^{\uparrow}$ consumer row whose visible inputs are precisely the carrier, classifier, dyadic observation, and RegSeqRat ledger rows already assembled. Since none of these sources introduces quotient formation, arithmetic or order closure, metric-completion data, Classical data, or trichotomy, the resulting package is exactly the stated finite-window seal interface. □

The ten-field fit is: traditional target transport of a unary real stream-name classifier; BEDC source StreamNameUp read through the RealUp unary classifier; carrier the transported RatUp endpoint carrier rows for s' and t'; classifier the transported RatUp point classifier; stability pointwise hsame transport along unary observations; ledger the observation witness and the two endpoint transport witnesses; core theorem this block; exactness RatStreamNameClassifier(s', t') rather than only RealUnaryStreamClassifier(s', t'); standard bridge the downstream RealUp certificate boundary; status ProofTarget.

Definition 38.279 (Real ledger policy)**.** *Record finite stream observations, dyadic comparison witnesses, regularity certificates, and classifier comparison certificates when sameness is used.*

Definition 38.280 (RealUp certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the required fields for using the interface name* $\mathsf{Real}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Real}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 38.281 (Zero-axiom encoding note)**.** *The interface avoids quotient machinery. It does not use* Quot.sound*; sameness is carried by dyadic comparison certificates and the* RegSeqRat *classifier.*

Theorem 38.282 (RealUp carrier and classifier obligations)**.** *The* $\mathsf{Real}^{\uparrow}$ *carrier and classifier obligation set consists exactly of the sealed* RegSeqRat *stream spine, the constant-history carrier* RealConstantHistoryCarrier*, the endpoint* $\mathsf{Rat}^{\uparrow}$ *carrier rows, and the classifier row* RealConstantHistoryClassifier *or* RealUnaryStreamClassifier *read through Definition 38.1, Definition 38.2, Definition 38.9, and Definition 38.195. This is the carrier and classifier part of the* $\mathsf{Real}^{\uparrow}$ *obligation boundary.*

Proof. Use the roadmap ten-field template with traditional target $\mathsf{Real}^{\uparrow}$. The BEDC source is the DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ interface over Hist; no quotient carrier is introduced outside those named interfaces.

The carrier choice is the sealed RegSeqRat stream spine together with RealConstantHistoryCarrier and the endpoint $\mathsf{Rat}^{\uparrow}$ carrier rows. Definitions 38.1 and 38.2 name the constant-history side, while Definition 38.195 names the stream spine being sealed.

The classifier choice is read from Definition 38.9: constants use RealConstantHistoryClassifier and unary stream observations use RealUnaryStreamClassifier. Stability obligations, ledger rows, core theorem list, and exactness predicate are not supplied by this block; they are the remaining $\mathsf{Real}^{\uparrow}$ obligations. No standard bridge is claimed, and the status of this theorem is the paper-side carrier and classifier obligation inventory. □

Theorem 38.283 (RealUp stability and transport obligations)**.** *The* $\mathsf{Real}^{\uparrow}$ *stability obligation set is supplied by the constant-history classifier equivalence fields, endpoint transport, unary stream classifier stability, the stream-name transported classifier, and* RegSeqRat *transport. Concretely, the required rows are those supplied by Definition 38.23, Definition 38.18, Definition 38.273, and Definition 38.275, read over the same* Hist*-indexed DyadicRatCore,* RegSeqRat*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *sources.*

Proof. The traditional target is again $\mathsf{Real}^{\uparrow}$, with BEDC source DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ over Hist. The carrier choice and classifier choice are the rows fixed in Definition 38.282; this proof names the stability and transport rows attached to them.

For the constant-history face, apply Definition 38.23 to obtain the reflexive, symmetric, and transitive classifier fields, and apply Definition 38.18 to move the endpoint classifier along hsame rows.

For the unary stream face, apply Definition 38.273. It gives the StreamNameUp classifier, the reflexive, reversed, and composed $\mathsf{Real}^{\uparrow}$ unary classifier rows, endpoint $\mathsf{Rat}^{\uparrow}$ carrier rows, and transport of RealUnaryStreamClassifier along pointwise hsame.

Finally, apply Definition 38.275 to read the transported RealUp unary classifier as the transported StreamNameUp classifier. The ledger and seal are left to the ledger theorem below, the exactness predicate is classifier preservation along the displayed transports, no standard bridge is claimed, and the status is the paper-side stability and transport obligation inventory. □

Theorem 38.284 (RealUp ledger and seal obligations)**.** *The* $\mathsf{Real}^{\uparrow}$ *ledger obligation is exactly the finite unary-observation ledger together with the dyadic-tolerance requests consumed by the* $\mathsf{Real}^{\uparrow}$ *seal. The ledger rows are those named in Definition 38.279, and the seal consumes precisely the scoped* RegSeqRat *spine supplied by Definition 38.210 through Definition 38.209.*

Proof. Use the same roadmap template. The traditional target is $\mathsf{Real}^{\uparrow}$; the BEDC source is the DyadicRatCore and RegSeqRat stream data over Hist, read through $\mathsf{StreamName}^{\uparrow}$ names and endpoint $\mathsf{Rat}^{\uparrow}$ rows. The carrier and classifier choices are the rows fixed by Definition 38.282.

The ledger is the finite policy of Definition 38.279: stream observations, dyadic comparison witnesses, regularity certificates, and classifier comparison certificates. These are the only ledger rows available to the seal.

Apply Definition 38.210 to collect the scoped RegSeqRat carrier, classifier, stability, and finite ledger faces. Then apply Definition 38.209; the seal reads that scoped spine and its dyadic-tolerance requests without adding another source of data.

The core theorem list is the scoped spine theorem followed by the seal consumption theorem. The exactness predicate is equality of consumed data: the $\mathsf{Real}^{\uparrow}$ seal consumes the finite unary-observation ledger and the dyadic-tolerance requests displayed by the spine. No standard bridge is claimed, and the status is the paper-side ledger, seal, and RegSeqRat-consumption obligation inventory. □

Theorem 38.285 (RealUp scoped certificate boundary)**.** *The scoped* $\mathsf{Real}^{\uparrow}$ *certificate boundary is exactly the package formed from Definition 38.282, Definition 38.283, Definition 38.284, Definition 38.335, Definition 38.336, Definition 38.337, and Definition 38.338. Its primitives are DyadicRatCore,* RegSeqRat*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* UnaryHistory*,* hsame *transport,* RealConstantHistoryCarrier*,* RealConstantHistoryClassifier*,* RealUnaryStreamClassifier*, and* RealLedgerPolicy*. Within this boundary every carrier, classifier, stability, transport, ledger, and seal face is read from the same* Hist*-indexed stream spine and finite ledger. The boundary contains no quotient data, completion data, arithmetic or order closure, metric-completeness certificate, or standard-bridge data.*

Proof. We first collect the direct $\mathsf{Real}^{\uparrow}$ obligation faces. Apply Definition 38.282 for the sealed RegSeqRat carrier and the constant-history and unary stream classifiers. Apply Definition 38.283 for hsame transport, endpoint transport, and unary stream classifier stability. Apply Definition 38.284 for the finite ledger and the seal that consumes the scoped stream spine.

We then identify those faces with the stream-prefix obligation inventory. Apply Definition 38.335 for the DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ carrier source. Apply Definition 38.336 for the constant-history and unary stream classifier source. Apply Definition 38.337 for the pointwise hsame stability rows. Apply Definition 38.338 for the finite ledger policy and seal rows.

The two collections name the same data, because every row is indexed by the same Hist stream spine and every ledger entry is one of the finite RealLedgerPolicy entries consumed by the seal. Thus the package is scoped over the displayed BEDC primitives rather than over an external real object.

Nothing in the construction supplies quotient representatives, a completion object, arithmetic or order laws, metric completeness, or a standard bridge. Those data are outside the scoped certificate boundary claimed here. □

Theorem 38.286 (RealUp public certificate export)**.** *The public* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *surface exported to downstream L10 chapters is exactly the package consisting of Definition 38.285, Definition 38.342, the scoped* RegSeqRat *stream-name interface of Definition 38.276, the DyadicRatCore observation rows,* RealConstantHistoryCarrier*,* RealConstantHistoryClassifier*,* RealUnaryStreamClassifier*, and* RealLedgerPolicy*. Downstream chapters may cite this theorem as the public* $\mathsf{Real}^{\uparrow}$ *certificate boundary for carrier, classifier, stability, transport, finite observation, ledger, and seal data. The exported surface contains no completeness theorem, arithmetic or order closure, metric-completeness certificate, or standard bridge.*

Proof. Start with the scoped certificate boundary. Apply Definition 38.285; it gathers the DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, UnaryHistory, hsame transport, constant-history carrier,

constant-history classifier, unary stream classifier, and finite ledger policy as one Hist-indexed $\mathsf{Real}^{\uparrow}$ boundary.

Next attach the finite read surface. Apply Definition 38.342; it says that selected finite $\mathsf{Real}^{\uparrow}$ observation windows read exactly the same carrier/classifier, stability/transport, finite ledger, dyadic source, stream-name, endpoint, and seal rows, and no quotient, completion, arithmetic, order, Classical, or standard-bridge datum.

Finally package the source interfaces exposed to downstream chapters. The scoped RegSeqRat stream-name interface of Definition 38.276 supplies the stream-name classifier, reflexive, reversed, composed, endpoint, and transported classifier rows. The DyadicRatCore observation rows and the named RealConstantHistoryCarrier, RealConstantHistoryClassifier, RealUnaryStreamClassifier, and RealLedgerPolicy rows are therefore precisely the public $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ surface. Since each row is already inside the scoped boundary or the finite read surface, downstream L10 chapters may cite this theorem as the public $\mathsf{Real}^{\uparrow}$ certificate boundary without reconstructing the package. □

Theorem 38.287 (RealUp public seal source determinacy)**.** *Let a downstream L10 use of a sealed* $\mathsf{Real}^{\uparrow}$ *candidate cite Definition 38.286 as its public seal. Then the source rows consumed from that seal are exactly the DyadicRatCore public observation package, the* $\mathsf{StreamName}^{\uparrow}$ *public countable-name interface and finite-window interface, the* RegSeqRat *stream-name scoped interface, and the named* $\mathsf{Real}^{\uparrow}$ *rows* RealConstantHistoryCarrier*,* RealConstantHistoryClassifier*,* RealUnaryStreamClassifier*, and* RealLedgerPolicy*.*

No quotient representative, completion object, metric-completeness row, arithmetic or order row, Classical datum, or standard bridge may be read from that public seal. Thus a downstream argument using the seal is source determinate over the displayed DyadicRatCore, $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, and* $\mathsf{Real}^{\uparrow}$ *rows alone.*

Proof. Start from Definition 38.286. It identifies the public seal as the scoped $\mathsf{Real}^{\uparrow}$ certificate boundary, the selected finite-window read surface, the RegSeqRat stream-name interface, the DyadicRatCore observation rows, and the named carrier, classifier, unary-stream, and ledger rows.

Factor the countable-name and finite-window faces through the public stream-name interfaces. Apply Definition 38.289; its factorization sends the countable-name part through Definition 126.311 and the finite-window part through Definition 126.301.

Project the remaining source rows. Apply Definition 38.258 for the dyadic finite observation package, and apply Definition 38.276 for the regular rational-stream spine with its transported stream-name rows. The only non-factorized $\mathsf{Real}^{\uparrow}$ rows left by the public export are the displayed RealConstantHistoryCarrier, RealConstantHistoryClassifier, RealUnaryStreamClassifier, and RealLedgerPolicy rows.

Finally compare this list with the roadmap ten-field template. The traditional target, BEDC source, carrier, classifier, stability, ledger, core theorem list, exactness, standard-bridge field, and status field are all read from the four public packages and named $\mathsf{Real}^{\uparrow}$ rows above. Since the cited public export and its factorizations explicitly withhold quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data, a downstream use of the seal cannot consume any such row. □

Theorem 38.288 (RegSeqRat finite-window seal source pullback)**.** *Let two downstream L10 seal uses of* $\mathsf{Real}^{\uparrow}$ *cite Definition 38.286. Suppose both uses restrict their public seal to the same certified finite* $\mathsf{StreamName}^{\uparrow}$ *window* U *and there carry matching* RatStreamNameFiniteWindowClassifier *rows, matching* RegSeqRat *spine rows, matching endpoint* DyadicRatCoreObservationRow *witnesses, and matching displayed* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. Then the finite* RegSeqRat

seal-source package induced by either use and the finite $\mathsf{Real}^{\uparrow}$ *readback packet induced by either use are the same public* BHist *source surface over* U*.*

This pullback uses only DyadicRatCore observation rows, the certified $\mathsf{StreamName}^{\uparrow}$ *finite window, the* RegSeqRat *spine rows, displayed* hsame/Cont/Pkg *provenance, and the finite* $\mathsf{Real}^{\uparrow}$ *ledger boundary. It does not export quotient data, completion data, metric-completeness rows, arithmetic rows, order rows, Classical representatives, or standard-bridge rows.*

Proof. First apply Definition 38.287 to both downstream seal uses. Each use can read only the displayed DyadicRatCore, $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and named $\mathsf{Real}^{\uparrow}$ rows from Definition 38.286.

Next factor the common stream-name restriction through Definition 126.88. The matching finite-window classifier rows, probe window, endpoint observation rows, and displayed hsame transports identify one $\mathsf{StreamName}^{\uparrow}$ source surface on U for every induced RegSeqRat handoff.

Now apply Definition 38.355. The matching RegSeqRat spine, DyadicRatCore endpoints, Cont/Pkg provenance, and finite RealLedgerPolicy rows identify the two restricted $\mathsf{Real}^{\uparrow}$ readback packets with that same finite BHist packet.

The finite ledger boundary then repacks the common packet as the public source surface read by both L10 consumers. Since every step has already excluded quotient, completion, metric-completeness, arithmetic, order, Classical, and standard-bridge rows, no additional source row is available in the pullback. □

Theorem 38.289 (RealUp public export through the StreamNameUp public interface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *surface of Definition 38.286 factors its countable-name component through the public* $\mathsf{StreamName}^{\uparrow}$ *export package of Definition 126.311 and factors its finite-window component through Definition 38.345. After these two factorizations, the remaining public* $\mathsf{Real}^{\uparrow}$ *rows are exactly the DyadicRatCore observation rows, the* RegSeqRat *spine rows, the* $\mathsf{Rat}^{\uparrow}$ *endpoint rows,* UnaryHistory*,* hsame *transport,* RealConstantHistoryCarrier*,* RealConstantHistoryClassifier*,* RealUnaryStreamClassifier*, and* RealLedgerPolicy*.*

The factorized public surface still contains no quotient datum, completion datum, arithmetic or order closure, metric-completeness certificate, Classical datum, or standard bridge.

Proof. Start from Definition 38.286. It identifies the public $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ boundary as the scoped $\mathsf{Real}^{\uparrow}$ certificate boundary together with the selected finite-window read surface, the RegSeqRat stream-name interface, the DyadicRatCore observation rows, and the named constant-history, unary-stream, transport, endpoint, and ledger rows.

Next isolate the public stream-name package. Apply Definition 126.311; its countable-name side supplies the RatStreamNameCarrier, RatStreamNameClassifier, RatUp pointwise rows, constant stream rows, reindexing and RatUp-preserving map stability, and unary reindexing composition used by the $\mathsf{Real}^{\uparrow}$ export.

For finite observations, apply Definition 38.345. It identifies the selected finite $\mathsf{Real}^{\uparrow}$ read surface with the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface and leaves only the listed dyadic, RegSeqRat, endpoint, unary-history, hsame, and ledger rows outside the stream-name factor.

Finally use Definition 38.276 and Definition 38.258. The former supplies the regular rational-stream spine and transported stream-name rows consumed by $\mathsf{Real}^{\uparrow}$, while the latter supplies the DyadicRatCore observation package. Thus the public export factors through the two StreamNameUp interfaces and the unfactored rows are precisely the displayed $\mathsf{Real}^{\uparrow}$ carrier, classifier, transport, endpoint, history, and ledger rows. □

Theorem 38.290 (RealUp Phase Real and Completion boundary)**.** *The Phase Real and Completion paper boundary is exactly the composite of the following four public packages:*

1. *the DyadicRatCore public observation package of Definition 38.258;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *public countable-name export package of Definition 126.311 together with the finite-window interface of Definition 126.301;*
3. *the* RegSeqRat *stream-name scoped interface of Definition 38.276;*
4. *the* $\mathsf{Real}^{\uparrow}$ *public certificate export of Definition 38.286 after factorization through the* $\mathsf{StreamName}^{\uparrow}$ *public interface in Definition 38.289.*

Equivalently, its roadmap ten-field row has traditional target regular real completion by dyadic rational streams; BEDC source the DyadicRatCore, $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, and* $\mathsf{Real}^{\uparrow}$ *packages listed above; carrier the DyadicRatCore observations, rational stream names, regular stream spine, and* RealConstantHistoryCarrier *rows; classifier the public* $\mathsf{StreamName}^{\uparrow}$ *classifier, the transported* RegSeqRat *stream classifier, and* RealConstantHistoryClassifier *with* RealUnaryStreamClassifier*; stability the* hsame*, endpoint, stream-name, finite-window, and unary-stream transport rows exposed by those packages; ledger the finite dyadic observation rows and* RealLedgerPolicy*; core theorem list exactly the six displayed source theorems; exactness finite-window and public-export exactness only up to the named classifiers; standard bridge absent; status paper-level phase boundary for the future namespace* BEDC.Derived.RealUp*.*

The boundary contains no $\mathsf{Real}^{\uparrow}$ *completeness theorem, no full* $\mathsf{Real}^{\uparrow}$ *arithmetic or order closure, no metric-completeness certificate, no quotient construction, no Classical datum, and no standard bridge.*

Lean4 ✓ `BEDC.Derived.RealUp.RealUp concrete to schema`

Proof. First collect the four public package faces. Apply Definition 38.258 for the finite dyadic observation source. Apply Definition 126.311 and Definition 126.301 for the public countable-name and finite-window surfaces. Apply Definition 38.276 for the regular rational-stream spine.

Next attach the $\mathsf{Real}^{\uparrow}$ export. Apply Definition 38.286 to read the scoped $\mathsf{Real}^{\uparrow}$ certificate boundary as a public downstream package. Apply Definition 38.289 to factor its countable-name and finite-window components through the public $\mathsf{StreamName}^{\uparrow}$ interfaces, leaving precisely the dyadic, regular-stream, endpoint, unary-history, hsame, carrier, classifier, and ledger rows named in the statement.

Finally repack the collected rows using the roadmap ten-field template. The traditional target, BEDC source, carrier, classifier, stability, ledger, core theorem list, exactness, standard-bridge field, and status field are all read from the cited public packages. Since each cited package explicitly withholds quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data, the composite phase boundary withholds exactly those data as well. □

Theorem 38.291 (RealUp completion phase interface exhaustion)**.** *Every public consumer row for the Phase Real and Completion package of Definition 38.290 factors through exactly one of the four displayed BEDC surfaces: DyadicRatCore public observation rows,* $\mathsf{StreamName}^{\uparrow}$ *pointwise finite-window rows,* RegSeqRat *regular-spine and dyadic-tolerance rows, or the* $\mathsf{Real}^{\uparrow}$ *seal and readback rows. The exhaustion is exclusive at the consumer boundary: dyadic observations are read only through Definition 38.258, stream-name windows only through the public* $\mathsf{StreamName}^{\uparrow}$ *rows of Definition 126.311 and Definition 126.301, regular-spine and dyadic-tolerance data only through Definition 38.276, and seal/readback data only through the* $\mathsf{Real}^{\uparrow}$ *export and seal interfaces of Definition 38.286, Definition 38.289, Definition 38.287, and Definition 38.278.*

No public consumer row in this phase may read completion data, quotient representatives, host stream equality, or a direct $\mathsf{Real}^{\uparrow}$ *read that bypasses the* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *handoff.*

Proof. Begin with the composite boundary of Definition 38.290. Its four cases are the only rows placed in the phase package, so a public consumer row must be dyadic, stream-name finite-window, regular-spine with dyadic tolerance, or $\mathsf{Real}^{\uparrow}$ seal/readback data.

In the dyadic case, apply Definition 38.258. It accounts for the public observation rows and exposes no completion, quotient, host equality, or direct $\mathsf{Real}^{\uparrow}$ read.

In the stream-name case, apply Definition 126.311 together with Definition 126.301. These theorems bind pointwise finite windows through the public $\mathsf{StreamName}^{\uparrow}$ surface and do not introduce regular-spine or seal rows.

In the regular-spine case, apply Definition 38.276 and then Definition 38.278 for the dyadic tolerance rows that hand the finite window to the $\mathsf{Real}^{\uparrow}$ seal. This is the only path from a stream-name read to a $\mathsf{Real}^{\uparrow}$ readback row.

In the seal/readback case, apply Definition 38.286, Definition 38.289, and Definition 38.287. The export factors through the public stream-name interface, and seal-source determinacy prevents any second source from serving the same public read. The excluded completion, quotient, host-equality, and direct-read data are absent from each cited surface, hence absent from the exhausted consumer boundary. □

Theorem 38.292 (RealUp completion phase selected-window determinacy)**.** *Let two public consumer rows of the Phase Real and Completion package select the same certified unary finite window from the same accepted boundary packet. Then both selected reads factor through the same displayed route: the DyadicRatCore public observation package, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface, the* $\mathsf{RegSeqRat}$ *stream-name scoped interface, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback rows.*

The two selected reads are compared only by the displayed hsame *and classifier rows along this route. No quotient representative, host stream equality, completion object, or direct* $\mathsf{Real}^{\uparrow}$ *bypass is available as a second comparison path.*

Proof. Apply Definition 38.291 to the two public consumer rows. Since both rows select a unary finite window, the exhaustion places them in the stream-name branch before any seal/readback row can be read.

Use Definition 126.301 to identify the selected certified window as a public $\mathsf{StreamName}^{\uparrow}$ finite-window read. Then apply Definition 38.276 and Definition 38.278 to repack that same window as the regular-spine input consumed by the $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 38.287. It rules out a second seal source for the same public read, so the two consumers can be compared only through the displayed hsame and classifier rows. The excluded quotient, host-equality, completion, and direct-bypass data are absent by the exhaustion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$**.**

Constructive story: The real-number boundary is built by joining four already displayed surfaces: DyadicRatCore public observations, $\mathsf{StreamName}^{\uparrow}$ countable names and finite windows, the RegSeqRat regular stream spine, and the scoped $\mathsf{Real}^{\uparrow}$ seal/readback rows. The carrier accepts only this composite package of dyadic observations, rational stream names, regularity spine, constant-history rows, endpoint transports, and finite-window ledgers; a completion object, quotient carrier, host equality, arithmetic package, order package, or metric-completeness row is not part of the accepted boundary. Two accepted real packets are classified by the public stream-name classifier, the transported regular-stream classifier, and the real constant/history classifier over the same unary observation and ledger rows, rather than by identifying external Cauchy classes. The source specification is the four-package phase boundary, the pattern specification records dyadic observation transport, public StreamName export, finite-window exactness, regular-spine seal consumption, public real export, endpoint determinacy, and consumer-boundary exhaustion, and the ledger keeps each dyadic, stream, regularity, seal, endpoint, and provenance row visible. The five name-certificate obligations are read through these component certificates: a displayed constant rational embedding inhabits the boundary, the component classifiers provide reflexivity, symmetry, and transitivity, and hsame, endpoint, stream-name, and finite-window transports keep the composite package

inside the same boundary. A packet h lies in the signature gap of a $\mathsf{Real}^{\uparrow}$ token p when the source-domain accepts one of the four public package faces, the signature relation pairs it with an answer real-boundary packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the dyadic observations, rational stream-name rows, regularity spine, and seal/readback entries match the rows encoded by p. The checked bridge marker attaches only to the constant rational embedding and boundary package recorded here; the chapter registers this phase-real construction under $\mathsf{Real}^{\uparrow}$ without closing completeness, full arithmetic, full order, metric completeness, quotient construction, Classical data, or any external bridge stronger than the checked BEDC schema bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 38.290 is the mature paper boundary over DyadicRatCore, $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Real}^{\uparrow}$, binding the dyadic observations, rational stream names, regular stream spine, public $\mathsf{Real}^{\uparrow}$ certificate export, endpoint and finite-window transports, and finite-window ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RealUp.RealUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This block does not close $\mathsf{Real}^{\uparrow}$ completeness, full arithmetic or order, metric completeness, quotient construction, Classical data, or any external bridge stronger than the checked BEDC schema bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 38.293 (Real stream prefix finite-tail transported endpoint denominator package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, indices n, m, and transported endpoint histories $u, v : \mathsf{Hist}$, assume*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n),$$
$$\forall k < m,\ \mathsf{RatHistoryClassifier}(x(n+k+1), y(n+k+1)),$$
$$\forall i,\ \mathsf{UnaryHistory}(x(i)),$$

and

$$\mathsf{hsame}(x(n+m), u), \quad \mathsf{hsame}(y(n+m), v).$$

Then the transported finite-tail boundary carries the endpoint denominator package:

$$\mathsf{RatHistoryClassifier}(u, v) \wedge \mathsf{PositiveUnaryDenominator}(u) \wedge \mathsf{PositiveUnaryDenominator}(v) \wedge \mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge (\mathsf{hsame}(u, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(v, \mathsf{e}) \to \mathsf{False}).$$

Ten-field fit: traditional target finite-tail transported RealUp boundary package; BEDC source RealUp stream prefixes with RatUp finite tail classifiers; carrier rational-observation streams and transported endpoint histories; classifier RealStreamPrefixClassifier *followed by* RatHistoryClassifier*; stability by tail extension with unary anchor, endpoint projection, and endpoint* hsame *transport; ledger base prefix length, tail length, selected boundary, and two endpoint transport rows; core theorem this finite-tail boundary package; exactness classifier, positive denominator, unary, and nonempty rows for u and v; standard bridge RegSeqRat finite boundary observations; status* ProofTarget*.*

Proof. First extend the classified prefix through the displayed finite tail. Apply Definition 38.116 to the prefix through n, the tail classifier row, and the pointwise unary anchor on x. This supplies $\mathsf{RealStreamPrefixClassifier}(x, y, n+m)$, together with the left unary boundary row kept in the construction.

Next apply Definition 38.111 at the boundary index $n+m$. The selected endpoint classifier is $\mathsf{RatHistoryClassifier}(x(n+m), y(n+m))$.

Transport this RatUp classifier along the two displayed hsame rows by Definition 37.54. The transported classifier is $\mathsf{RatHistoryClassifier}(u, v)$.

Finally apply Definition 37.50 to the transported classifier, and then apply Definition 37.12 separately to the two positive-denominator witnesses. These readback rows provide the two positive denominator witnesses, the two unary endpoint witnesses, and the two nonempty exclusions, so the displayed package repacks directly. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier finite tail transported endpoint denominator package`

Theorem 38.294 (Real stream prefix finite-tail transported full endpoint package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, indices n, m, transported endpoint histories $u, v : \mathsf{Hist}$, and histories $a, b, z_U, z_V : \mathsf{Hist}$, assume*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n),$$

$$\forall k < m,\ \mathsf{RatHistoryClassifier}(x(n+k+1), y(n+k+1)),$$

$$\forall i,\ \mathsf{UnaryHistory}(x(i)),$$

and

$$\mathsf{hsame}(x(n+m), u), \quad \mathsf{hsame}(y(n+m), v), \quad \mathsf{hsame}(u, \mathsf{E1}(a)), \quad \mathsf{hsame}(v, \mathsf{E1}(b)).$$

Then the transported finite-tail boundary carries the full endpoint package:

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(u, v) \wedge \mathsf{PositiveUnaryDenominator}(u) \wedge \mathsf{PositiveUnaryDenominator}(v) \wedge \\ &\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge (\mathsf{hsame}(u, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(v, \mathsf{e}) \to \mathsf{False}) \wedge \\ &(\mathsf{hsame}(u, \mathsf{E0}(z_U)) \to \mathsf{False}) \wedge (\mathsf{hsame}(v, \mathsf{E0}(z_V)) \to \mathsf{False}) \wedge \\ &\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).\end{aligned}$$

Ten-field fit: traditional target finite transported rational observation boundary; BEDC source RealUp prefix classifier plus RatUp endpoint classifier and BHist *transports; carrier transported endpoint histories; classifier* RatHistoryClassifier*; stability endpoint* hsame *transport; ledger base prefix, finite tail, transported endpoints, and endpoint display tests; core theorem this block; exactness the displayed full package; standard bridge RegSeqRat finite boundary readback; status* ProofTarget*.*

Proof. First apply Definition 38.293 to the prefix classifier, finite-tail RatUp classifier row, pointwise unary anchor, and two transported endpoint rows. This gives the transported $\mathsf{RatHistoryClassifier}(u, v)$, both positive denominator witnesses, both unary endpoint rows, and both nonempty exclusions.

Next derive the zero-headed exclusions. If u is displayed as $\mathsf{E0}(z_U)$, transport the positive denominator witness for u across that display and apply RatUp positive-denominator zero-headed absurdity. The right endpoint uses the same argument with v and z_V.

Finally transport the transported classifier across the two displayed E1 comparisons. RatUp e-one tail classifier exactness then reads back the two unary tails and their internal sameness, namely $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$. Together with the denominator package and the two zero-headed exclusions, these rows repack as the displayed full endpoint package. □

Theorem 38.295 (RealUp StreamName tail denominator selector handoff)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same StreamName endpoints $s, t :$* BHist $\to$ BHist*, the same finite* ProbeBundle *window $\mathcal{P}$, and the same accepted* RegSeqRat *spine row. Suppose they are connected by a StreamName RegSeqRat RealUp tail-selector transport packet. Let $x, y : \mathbb{N} \to \mathsf{Hist}$ be the transported Real stream-prefix endpoint streams read from the selected finite window, let n, m be the displayed base and finite-tail lengths, and let $u, v : \mathsf{Hist}$ be transported endpoints satisfying*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n),$$

$$\forall k < m,\ \mathsf{RatHistoryClassifier}(x(n+k+1), y(n+k+1)),$$

$$\forall i,\ \mathsf{UnaryHistory}(x(i)),$$

and

$$\mathsf{hsame}(x(n+m), u), \qquad \mathsf{hsame}(y(n+m), v).$$

Then the $\mathsf{Real}^{\uparrow}$ *consumer boundary exposes exactly the shared StreamName public denominator selector selected by Definition 126.207, together with the transported finite-tail endpoint denominator package*

$$\mathsf{RatHistoryClassifier}(u,v) \wedge \mathsf{PositiveUnaryDenominator}(u) \wedge \mathsf{PositiveUnaryDenominator}(v) \wedge \mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge (\mathsf{hsame}(u,\mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(v,\mathsf{e}) \to \mathsf{False}).$$

Equivalently, the handoff is a single finite boundary over the shared StreamName selector, the accepted RegSeqRat *finite row, the transported Real stream-prefix endpoints,* BHist, hsame, Cont, *and* Pkg. *It adds no quotient row, completion object, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or external selector.*

Proof. First apply Definition 126.207 to the transport packet between W_0 and W_1. This identifies the selected finite public denominator selector carried by the StreamName side with the selector read by the finite $\mathsf{Real}^{\uparrow}$ ledger at W_1.

Next apply Definition 38.293 to the transported Real prefix classifier, the finite RatUp tail row, the pointwise unary anchor, and the two endpoint hsame rows. This supplies $\mathsf{RatHistoryClassifier}(u,v)$, the two positive unary denominator rows, the two unary endpoint rows, and the two nonempty endpoint exclusions.

Finally use Definition 38.294 as the readback boundary showing that the same finite-tail hypotheses admit no additional quotient, completion, representative, metric, arithmetic, order, or standard-bridge data. The consumer therefore sees only the shared StreamName denominator selector and the transported endpoint denominator package displayed above. □

Theorem 38.296 (RealUp StreamName tail selector transport determinacy)**.** *Let* H_0 *and* H_1 *be two* $\mathsf{Real}^{\uparrow}$ *StreamName tail-denominator selector handoffs over the same StreamName endpoints* $s, t : \mathsf{BHist} \to \mathsf{BHist}$*, the same finite* ProbeBundle *window* $\mathcal{P}$*, and the same accepted* RegSeqRat *spine row. Suppose both handoffs read their public denominator selector through Definition 126.207. Suppose the first handoff exposes transported Real stream-prefix endpoints* u_0, v_0 *and the second exposes transported Real stream-prefix endpoints* u_1, v_1 *from the same finite row.*

Then the two exposed endpoint denominator packages are classified by the same RatHistoryClassifier *row and expose the same positive-unary denominator rows, unary endpoint rows, and nonempty-endpoint exclusions. In particular the consumer cannot distinguish* H_0 *from* H_1 *by changing the endpoint package: both handoffs reduce to the shared StreamName selector, the same* RegSeqRat *finite spine, and the same transported package over* BHist, hsame, Cont, *and* Pkg. *No quotient, completion, arithmetic or order row, standard bridge row, or external selector row can separate the two handoffs at this boundary.*

Proof. Apply Definition 38.295 to H_0. The result identifies the public consumer boundary of the first handoff with the StreamName denominator selector selected by Definition 126.207 and with the transported finite-tail endpoint denominator package supplied by Definition 38.293.

Apply the same handoff theorem to H_1. The hypotheses have not changed: the StreamName endpoints, the finite ProbeBundle window, and the accepted RegSeqRat spine row are the same, and the public selector is read through the same StreamName theorem. Thus the second endpoint package is read from the same finite-tail denominator package rows as the first.

The endpoint-package theorem records exactly the RatHistoryClassifier row, the positive-unary denominator witnesses, the unary endpoint rows, and the two nonempty-endpoint exclusions. Since

both handoffs consume that same row data, replacing H_0 by H_1 leaves each of those rows fixed at the consumer boundary.

Finally the handoff theorem states the negative boundary: no quotient, completion object, arithmetic or order certificate, standard bridge, or external selector is added. Hence no such row can distinguish the two handoffs, and the exposed endpoint denominator package is determined by the shared StreamName selector and the shared $\mathsf{RegSeqRat}$ finite row. □

Theorem 38.297 (Real stream finite-window adjacent partition)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, m, r*,*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r) \iff \mathsf{RealStreamWindowClassifier}(x, y, n, m) \wedge \mathsf{RealStreamWindowClassif}$$

Proof. For the forward direction, assume the joined window $\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r)$. Apply Definition 38.172 with shift 0 and radius m; the bound $0{+}m \leq m{+}1{+}r$ is immediate, so the left piece is $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$.

Apply Definition 38.172 again with shift $m{+}1$ and radius r. The bound $(m{+}1){+}r \leq m{+}1{+}r$ identifies exactly the right adjacent piece $\mathsf{RealStreamWindowClassifier}(x, y, n + (m + 1), r)$. These two extracted windows form the conjunction.

For the reverse direction, assume the two adjacent pieces. Apply Definition 38.182 to the left piece and the right piece at anchor $n + (m + 1)$. The concatenation theorem repacks the bounded endpoint witnesses into $\mathsf{RealStreamWindowClassifier}(x, y, n, m + 1 + r)$. □

Theorem 38.298 (Real stream finite-window adjacent boundary readback package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, indices* n, m, r*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r)$*,* $\mathsf{hsame}(x(n{+}m), \mathsf{E1}(a))$*, and* $\mathsf{hsame}(y(n{+}m), \mathsf{E1}(b))$*. Then the selected adjacent boundary endpoint at* $n + m$ *carries the following package:*

1. $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$*;*
2. $\mathsf{RealStreamWindowClassifier}(x, y, n + (m + 1), r)$*;*
3. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$*;*
4. $\mathsf{PositiveUnaryDenominator}(x(n + m)) \wedge \mathsf{PositiveUnaryDenominator}(y(n + m))$*;*
5. $\mathsf{UnaryHistory}(x(n + m)) \wedge \mathsf{UnaryHistory}(y(n + m))$*;*
6. $(\mathsf{hsame}(x(n + m), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n + m), \mathsf{e}) \to \mathsf{False})$*;*
7. $(\mathsf{hsame}(x(n + m), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n + m), \mathsf{E0}(z_Y)) \to \mathsf{False})$*;*
8. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$*.*

Proof. Apply the forward direction of Definition 38.297 to the wide window $\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r)$. This produces the left window over (n, m) and the right adjacent window over $(n{+}(m{+}1), r)$.

The selected boundary is the offset m inside the wide window. Since $m \leq m + 1 + r$, Definition 38.156 applies to the wide window with $k = m$ and to the displayed one-extension comparisons at $x(n + m)$ and $y(n + m)$.

That selected readback package supplies the RatUp classifier row at $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$, the positive-denominator rows, the unary endpoint rows, the empty and zero-headed exclusions, and the tail agreement $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$. Repack these rows with the two adjacent windows obtained from the partition. □

The ten-field fit is: traditional target adjacent finite RegSeqRat boundary reads; BEDC source RealUp finite windows over RatUp endpoint classifiers and StreamNameUp observation schedules; carrier rational-observation streams and selected unary tails; classifier RealStreamWindowClassifier with the selected RatUp endpoint package; stability by adjacent-window partition and selected full-shape readback; ledger left finite window, right adjacent window, boundary index, displayed one-extension rows, finite ledger rows, UnaryHistory, and hsame; core theorems Definition 38.297 and Definition 38.156; exactness the displayed adjacent boundary package; standard bridge finite RealUp window consumption; status no completion, quotient, or host-index equality data.

Theorem 38.299 (Real unary stream finite-window adjacent partition)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor histories* $a, w, r : \mathsf{Hist}$*,*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r)))$$
$$\iff \mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r)$$

Proof. For the forward direction, unfold Definition 38.154 in the joined unary window. The displayed append ledger below $\mathsf{append}(w, \mathsf{BHist.e1}(r))$ separates offsets that remain below w from offsets that pass the visible successor $\mathsf{BHist.e1}(w)$.

The first ledger projection supplies every offset required for $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. The second projection supplies an offset below r from the shifted anchor $\mathsf{append}(a, \mathsf{BHist.e1}(w))$, and the carried hsame row identifies that shifted endpoint with the corresponding endpoint of the joined window. The RatUp endpoint classifier transports along this row.

For the reverse direction, assume the left and right unary windows. Apply Definition 38.183. It repacks the two windows, their UnaryHistory rows, and the finite Cont append ledger into the joined classifier over $\mathsf{append}(w, \mathsf{BHist.e1}(r))$. □

Theorem 38.300 (Real unary adjacent-window terminal source boundary)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and unary anchor and bound histories* $a, w, r : \mathsf{Hist}$*, suppose the joined finite window*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, \mathsf{append}(w, \mathsf{BHist.e1}(r)))$$

is consumed through a finite RegSeqRat *handoff and then by a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer. Then the terminal source-boundary projection is exactly the union of the two adjacent source windows supplied by Definition 38.299. Its rows are:*

1. *the left row* $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$*;*
2. *the right row* $\mathsf{RealUnaryStreamWindowClassifier}(s, t, \mathsf{append}(a, \mathsf{BHist.e1}(w)), r)$*;*
3. *the selected* RatHistoryClassifier *endpoint rows for the boundary and terminal subwindow endpoints;*
4. *the bundled* UnaryHistory *and* InBundle *rows;*
5. *the visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *provenance rows; and*
6. *the accepted* RegSeqRat *finite-window spine.*

No quotient row, completion object, host index equality, metric-completeness row, arithmetic or order operation, or standard bridge row is introduced by this terminal projection.

Proof. First apply Definition 38.299 to the joined classifier. This separates the source into the left unary window at anchor a and the right unary window at the shifted anchor $\mathsf{append}(a, \mathsf{BHist.e1}(w))$.

Next feed the two finite windows through the StreamName common-refinement source surface. Use Definition 38.400 and Definition 38.399 to identify the common finite source package read by the StreamName rows, the accepted RegSeqRat handoff, and the finite $\mathsf{Real}^{\uparrow}$ consumer.

Finally apply Definition 38.380 and Definition 38.377. The first theorem removes the route choice through the common-refinement packet; the second identifies the terminal seal projection with exactly the classifier, endpoint, bundle, transport, provenance, ledger, and accepted-spine rows just listed. Its exclusion clause gives the absence of quotient, completion, host-index, metric, arithmetic, order, and bridge data. □

Theorem 38.301 (Real stream window RegSeqRat partition handoff)**.** *Let a joined finite* RealStreamWindowClassif *window be partitioned into adjacent left and right windows. Then any accepted* RegSeqRat *handoff and finite* $\mathsf{Real}^{\uparrow}$ *seal read from the joined window factors through the two adjacent partition rows together with the boundary endpoint package.*

The handoff surface consists of the left and right RealStreamWindowClassifier *rows, the selected* RatHistoryClassifier *boundary endpoint row, the displayed* UnaryHistory *and denominator rows, the finite* hsame *and* Cont *transport rows, the* Pkg *provenance, and the accepted* RegSeqRat *spine. It introduces no quotient row, completion object, host index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.*

Proof. First apply Definition 38.297. The joined finite classifier separates into the left window and the right adjacent window, so the source of the accepted RegSeqRat handoff is already displayed as two finite partition rows.

Next use Definition 38.298. The boundary endpoint package supplies the selected RatHistory classifier row, positive-denominator rows, unary rows, endpoint exclusions, and the boundary hsame comparison needed by the handoff.

Finally apply Definition 38.300. The terminal finite-window $\mathsf{Real}^{\uparrow}$ seal reads only the two adjacent source windows, the boundary endpoint package, the visible transport and provenance rows, and the accepted RegSeqRat spine. The same boundary theorem excludes quotient, completion, host-index, metric-completeness, arithmetic, order, and bridge coordinates. □

The ten-field fit is: traditional target finite adjacent observation windows over unary histories; BEDC source RealUp windows over StreamNameUp schedules $s, t : \mathsf{Hist} \to \mathsf{Hist}$; carrier unary anchor and bound histories; classifier RatUp endpoint classifiers at append-selected endpoints; stability by unary offset partition and adjacent concatenation; ledger UnaryHistory, Cont, hsame, left-window rows, right-window rows, and the visible successor split; core theorem Definition 38.299; exactness the joined unary window is equivalent to its left and right adjacent windows; standard bridge RegSeqRat finite-prefix consumption; status ProofTarget.

The ten-field fit is: traditional target finite adjacent observation intervals; BEDC source RealUp finite windows over RatUp endpoint classifiers; carrier rational-observation streams; classifier RealStreamWindowClassifier; stability by subwindow extraction and concatenation; ledger left radius, right radius, join point, and bounded-offset witnesses; core theorem Definition 38.297; exactness the displayed equivalence between the joined window and its adjacent pieces; standard bridge finite RegSeqRat observation intervals; status ProofTarget.

Theorem 38.302 (Real stream finite-window overlap partition)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$ *and indices* n, m, r*,*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}r) \iff \mathsf{RealStreamWindowClassifier}(x, y, n, m) \wedge \mathsf{RealStreamWindowClassifier}($$

Proof. For the forward direction, assume the wide window $\mathsf{RealStreamWindowClassifier}(x, y, n, m+r)$. Apply Definition 38.172 with shift 0 and radius m; the bound $0 + m \leq m + r$ extracts the left window $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$.

Apply Definition 38.172 again with shift m and radius r. The bound $m+r \leq m+r$ extracts the right overlapping window $\mathsf{RealStreamWindowClassifier}(x, y, n+m, r)$. These two extracted windows form the displayed conjunction.

For the reverse direction, assume the two overlapping windows. Unfold Definition 38.166 and fix a selected offset $k \leq m + r$. It remains to produce the RatUp endpoint classifier at $n + k$.

If $k \leq m$, the left window applies directly at offset k and gives

$$\mathsf{RatHistoryClassifier}(x(n + k), y(n + k)).$$

In the remaining bounded-offset case, write $k = m + j$ with $j \leq r$. The right window applies at offset j and gives

$$\mathsf{RatHistoryClassifier}(x((n + m) + j), y((n + m) + j)).$$

The arithmetic identity $(n+m)+j = n+(m+j) = n+k$ identifies this endpoint with the selected endpoint of the wide window.

The two offset cases cover every $k \leq m+r$, with the boundary endpoint $k = m$ shared by both windows. Hence the collected endpoint classifiers repack as $\mathsf{RealStreamWindowClassifier}(x, y, n, m+r)$. □

Theorem 38.303 (Real stream window concatenation exactness)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, a wide bound w, and a certified finite concatenation spine P over (n, w) whose segment windows are (n_i, r_i), the wide classifier*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w)$$

is equivalent to any certified finite concatenation of the segment classifiers named by P:

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w) \iff \mathsf{ConcatSpine}_P\big(\mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i)\big).$$

The equivalence preserves exactly the scoped $\mathsf{Real}^{\uparrow}$ *read surface: endpoint* $\mathsf{Rat}^{\uparrow}$ *classifiers,* DyadicRatCore *observation rows,* RegSeqRat *spine rows,* $\mathsf{StreamName}^{\uparrow}$ *request names,* UnaryHistory *witnesses,* hsame *transports, and finite ledger rows. It does not use a completion object, metric-completeness data, arithmetic or order closure, quotient data, Classical data, or a bridge row.*

Proof. First convert the concatenation spine to the partition fold. The certified spine rows record exactly the same finite list of anchors, radii, unary append displays, and Cont transitions as the partition spine in Definition 38.308. Repacking these rows changes only the bracketing of the finite segment list, not the displayed observations.

Apply Definition 38.308. It identifies the wide classifier at (n, w) with the finite conjunction of the segment windows named by the spine. The certified concatenation spine is a finite bracketing of the same conjunction, so associating the displayed rows along the spine gives the stated equivalence.

We now identify the public boundary carried by the equivalence. Apply Definition 38.309; both the wide window and the segment presentation expose exactly the endpoint $\mathsf{Rat}^{\uparrow}$ rows, dyadic observation rows, RegSeqRat rows, $\mathsf{StreamName}^{\uparrow}$ names, unary witnesses, hsame transports, and finite ledger rows listed in the statement.

Finally apply Definition 38.310. Its transport rows show that changing from the partition fold to the concatenation spine preserves the same reflexivity, symmetry, transitivity, and pointwise hsame

stability package. Since the three cited surfaces already exclude completion, metric-completeness, arithmetic and order closure, quotient, Classical, and bridge data, the concatenation equivalence introduces none of those rows. □

The ten-field fit is: traditional target overlapping finite rational observation intervals; BEDC source RealStreamWindowClassifier over RatUp endpoint classifiers; carrier rational-observation streams; classifier finite-window predicate; stability by subwindow extraction and bounded-offset splitting; ledger anchor, overlap radius, right radius, shared boundary, and selected offset; core theorem Definition 38.302; exactness the displayed equivalence; standard bridge overlapping finite RegSeqRat observation intervals; status ProofTarget.

Theorem 38.304 (Real stream finite-window overlap boundary readback package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, indices* n, m, r*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamWindowClassifier}(x, y, n, m + r)$*,* $\mathsf{hsame}(x(n + m), \mathsf{E1}(a))$*, and* $\mathsf{hsame}(y(n + m), \mathsf{E1}(b))$*. Then the shared boundary endpoint at* $n + m$ *carries the following package:*

1. $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$*;*
2. $\mathsf{RealStreamWindowClassifier}(x, y, n + m, r)$*;*
3. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$*;*
4. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$*;*
5. $\mathsf{PositiveUnaryDenominator}(x(n + m)) \wedge \mathsf{PositiveUnaryDenominator}(y(n + m))$*;*
6. $\mathsf{UnaryHistory}(x(n+m)) \wedge \mathsf{UnaryHistory}(y(n+m)) \wedge (\mathsf{hsame}(x(n+m), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n+m), \mathsf{e}) \to \mathsf{False})$*;*
7. $(\mathsf{hsame}(x(n + m), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n + m), \mathsf{E0}(z_Y)) \to \mathsf{False})$*.*

Proof. Apply the forward direction of Definition 38.302 to the wide window $\mathsf{RealStreamWindowClassifier}(x, y, n, m+r)$. This produces the left window over (n, m) and the right overlapping window over $(n + m, r)$.

The shared boundary is the selected offset m inside the wide window. Since $m \leq m + r$, Definition 38.156 applies to the wide window with $k = m$ and to the displayed one-extension comparisons at $x(n + m)$ and $y(n + m)$.

That selected readback package gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$, the two unary tail witnesses with $\mathsf{hsame}(a, b)$, the positive denominator rows, the unary and nonempty endpoint rows, and the two zero-head exclusions. Repack these rows with the two windows obtained from the overlap partition. □

The ten-field fit is: traditional target shared boundaries of finite RegSeqRat observation intervals; BEDC source RealUp finite windows, RatUp endpoint classifier, and StreamNameUp observation schedule; carrier rational-observation streams; classifier RealStreamWindowClassifier with the selected boundary RatUp package; stability by overlap partition and selected endpoint readback; ledger anchor, left radius, right radius, shared endpoint, and displayed one-extension witnesses; core theorems Definition 38.302 and Definition 38.156; exactness the displayed boundary package; standard bridge RealUp finite-window exhaustion boundary for RegSeqRat finite observations; status ProofTarget.

Theorem 38.305 (Real stream finite-window two-boundary readback package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, indices* n, m, r, s*, and histories* $a_0, b_0, a_1, b_1, z_{X,0}, z_{Y,0}, z_{X,1}, z_{Y,1} : \mathsf{Hist}$*, assume*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m + 1 + r + 1 + s),$$

with displayed one-extension endpoints

$$\mathsf{hsame}(x(n+m), \mathsf{E1}(a_0)) \wedge \mathsf{hsame}(y(n+m), \mathsf{E1}(b_0))$$

and

$$\mathsf{hsame}(x(n+(m+1)+r), \mathsf{E1}(a_1)) \wedge \mathsf{hsame}(y(n+(m+1)+r), \mathsf{E1}(b_1)).$$

Then the two cut points $n+m$ and $n+(m+1)+r$ carry the following finite readback package:

1. *the three segment windows*

   $\mathsf{RealStreamWindowClassifier}(x, y, n, m)$, $\mathsf{RealStreamWindowClassifier}(x, y, n{+}(m{+}1), r)$, $\mathsf{RealStreamWinc}$

2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a_0), \mathsf{E1}(b_0))$ *and* $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a_1), \mathsf{E1}(b_1))$*;*

3. $\mathsf{PositiveUnaryDenominator}(x(n{+}m))$*,* $\mathsf{PositiveUnaryDenominator}(y(n{+}m))$*,* $\mathsf{PositiveUnaryDenominator}(x(n{+}(m+1)+r))$*, and* $\mathsf{PositiveUnaryDenominator}(y(n+(m+1)+r))$*;*

4. $\mathsf{UnaryHistory}(x(n{+}m))$*,* $\mathsf{UnaryHistory}(y(n{+}m))$*,* $\mathsf{UnaryHistory}(x(n{+}(m{+}1){+}r))$*, and* $\mathsf{UnaryHistory}(y(n{+}(m+1)+r))$*;*

5. *the empty endpoint exclusions*

   $(\mathsf{hsame}(x(n{+}m), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n{+}m), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(x(n{+}(m{+}1){+}r), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n{+}($

6. *the zero-head endpoint exclusions*

   $(\mathsf{hsame}(x(n{+}m), \mathsf{E0}(z_{X,0})) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n{+}m), \mathsf{E0}(z_{Y,0})) \to \mathsf{False}) \wedge (\mathsf{hsame}(x(n{+}(m{+}1){+}r), \mathsf{E0}(z_{X,1})) \to$

7. *the unary tail rows*

   $\mathsf{UnaryHistory}(a_0) \wedge \mathsf{UnaryHistory}(b_0) \wedge \mathsf{hsame}(a_0, b_0) \wedge \mathsf{UnaryHistory}(a_1) \wedge \mathsf{UnaryHistory}(b_1) \wedge \mathsf{hsame}(a_1, b_1).$

Proof. First apply Definition 38.306 to the wide window. The forward direction extracts the left, middle, and right segment windows at (n, m), $(n+(m+1), r)$, and $(n+(m+1)+(r+1), s)$.

The first cut point is the selected offset m of the wide window. Since $m \leq m+1+r+1+s$, Definition 38.156 applies to the wide classifier and to the displayed endpoints at $n+m$.

The second cut point is the selected offset $m+1+r$ of the same wide window. The bound $m+1+r \leq m+1+r+1+s$ lets the same selected full-shape readback theorem apply to the displayed endpoints at $n+(m+1)+r$.

Finally repack the three segment windows with the two selected endpoint packages. The selected readback theorem supplies, at each cut point, the RatHistoryClassifier row, positive denominator rows, unary endpoint rows, empty and zero-head exclusions, and unary tail agreement for the displayed one-extension pair. □

The ten-field fit is: traditional target two cut points of a three-piece finite RegSeqRat observation window; BEDC source RealUp finite windows, RatUp endpoint classifiers, and displayed one-extension endpoints; carrier rational-observation streams; classifier RealStreamWindowClassifier with two selected RatUp readbacks; stability by three-piece partition and selected endpoint readback; ledger anchor, three segment radii, two cut points, and two displayed endpoint pairs; core theorems Definition 38.306 and Definition 38.156; exactness the displayed two-boundary package; standard bridge finite three-piece RegSeqRat observation readback for RealUp; status ProofTarget.

Theorem 38.306 (Real stream finite-window three-piece partition)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, m, r, s,*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r{+}1{+}s) \iff \mathsf{RealStreamWindowClassifier}(x, y, n, m) \wedge \mathsf{RealStreamWindow}$$

Proof. For the forward direction, assume the wide window $\mathsf{RealStreamWindowClassifier}(x, y, n, m + 1 + r + 1 + s)$. Apply Definition 38.297 with right radius $r + 1 + s$ to split it into the left window at (n, m) and the remaining right block at $(n + (m + 1), r + 1 + s)$.

Apply Definition 38.297 again to that remaining block, now with left radius r and right radius s. This gives the middle window at $(n+(m+1), r)$ and the right window at $(n+(m+1)+(r+1), s)$, and the three extracted windows form the displayed conjunction.

For the reverse direction, assume the three displayed windows. Apply the reverse direction of Definition 38.297 to the middle and right windows, producing the combined right block $\mathsf{RealStreamWindowClassifier}(x, y, n + (m + 1), r + 1 + s)$.

Apply the reverse direction of Definition 38.297 once more to the left window and this combined right block. The result is the wide window $\mathsf{RealStreamWindowClassifier}(x, y, n, m{+}1{+}r{+}1{+}s)$. □

The ten-field fit is: traditional target finite RegSeqRat observation chains; BEDC source RealUp windows over RatUp endpoint classifiers and StreamNameUp observation schedules; carrier rational-observation streams; classifier RealStreamWindowClassifier; stability by adjacent partition and concatenation; ledger two cut points and three bounded-offset blocks; core theorem Definition 38.306; exactness equivalence between the wide window and the three chained windows; standard bridge finite observation interval refinement for RealUp; status ProofTarget.

Definition 38.307 (Real stream finite-window partition spine)**.** *For a rational-observation anchor n and a total finite bound w, a nonempty* BEDC *finite partition spine is a finite* BHist*-coded chain P of unary radius histories $a_0, \ldots, a_{q-1} : \mathsf{Hist}$ together with endpoint continuation rows. Its numeric radii r_i are the unary lengths displayed by the a_i, its segment anchors are*

$$n_0 = n, \qquad n_{i+1} = n_i + (r_i + 1),$$

and its accumulated displayed bound is

$$w = r_0 + 1 + r_1 + 1 + \cdots + 1 + r_{q-1}.$$

The singleton spine has accumulated bound r_0 and one segment at the original anchor. The successor spine appends one unary radius a_i by a native unary append *display and a* Cont *row from the previous endpoint anchor to the next endpoint anchor. Thus every segment named by P is a concrete RealUp finite window* $\mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i)$*, and the final accumulated anchor is obtained only by repeated unary* append *and* Cont *composition.*

Theorem 38.308 (Real stream finite-window partition fold)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, a total bound w, and a certified finite partition spine P over (n, w) with segments (n_i, r_i), the total window classifier is equivalent to the conjunction of the segment classifiers named by P:*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w) \iff \bigwedge_{(n_i, r_i) \in P} \mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i).$$

For the singleton spine, the right side is the single segment classifier.

Proof. Proceed by induction on the certified partition spine P.

In the singleton case, the spine definition identifies the total bound with the single segment radius. The right side is the same $\mathsf{RealStreamWindowClassifier}(x, y, n, r_0)$ row, so the displayed equivalence is reflexive.

For the successor step, write the predecessor spine as P_0 with accumulated bound u and final anchor $n + u + 1$ in the displayed unary continuation ledger. The new certified unary radius is r, so the successor spine displays the total bound as $u + 1 + r$ and names the new final segment $\mathsf{RealStreamWindowClassifier}(x, y, n + (u + 1), r)$.

For the forward direction of the successor step, apply Definition 38.297 to the total window $\mathsf{RealStreamWindowClassifier}(x, y, n, u{+}1{+}r)$. This gives the predecessor wide window $\mathsf{RealStreamWindowClassifier}(x$ and the new final segment. Apply the induction hypothesis to the predecessor wide window to obtain every segment classifier named by P_0, then conjoin that list with the new final segment.

For the reverse direction, assume the conjunction over the successor spine. Project the predecessor segment conjunction and the new final segment. The induction hypothesis repacks the predecessor segment conjunction into $\mathsf{RealStreamWindowClassifier}(x, y, n, u)$. Apply the reverse direction of Definition 38.297 to this predecessor window and the new final segment, producing $\mathsf{RealStreamWindowClassifier}(x, y, n, u + 1 + r)$.

The three-segment case is the displayed consistency check: taking a spine with three radii m, r, s reduces the induction twice and gives exactly Definition 38.306. Its two displayed cut points are the first two successor anchors in the finite spine ledger. □

The ten-field fit is: traditional target finite RegSeqRat observation partition coverage; BEDC source RealStreamWindowClassifier plus RatHistoryClassifier endpoint rows and Cont with unary anchors; carrier rational-observation streams; classifier RealStreamWindowClassifier over a certified finite partition spine; stability by adjacent partition, induction on the BHist spine, and unary anchor continuation; ledger total bound, segment radii, accumulated anchors, continuation rows, and endpoint classifiers; core theorem Definition 38.308; exactness the displayed fold equivalence; standard bridge RealUp and RegSeqRat finite observation coverage; status ProofTarget.

Theorem 38.309 (RealUp finite-window partition read surface)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, a total bound w, and a certified finite partition spine P over (n, w) with segments (n_i, r_i), the total window $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$ and the conjunction of segment windows*

$$\bigwedge_{(n_i, r_i) \in P} \mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i)$$

expose the same scoped $\mathsf{Real}^{\uparrow}$ certificate boundary rows. These rows are exactly the endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, positive unary denominator rows, nonempty unary endpoint rows, common e-one tail readback wherever the endpoints are displayed as one-headed, finite RealLedgerPolicy *rows, DyadicRatCore observation rows,* RegSeqRat *spine rows,* $\mathsf{StreamName}^{\uparrow}$ *request names,* UnaryHistory *witnesses, and* hsame *transport rows named by the certified window spine. Neither presentation reads quotient data, a completion object, a metric-completeness certificate, arithmetic or order closure, Classical data, or a standard bridge.*

Proof. First transport between the two window presentations. Apply Definition 38.308; the total classifier at (n, w) is equivalent to the finite conjunction of the segment classifiers named by P.

Now read the total presentation through the scoped boundary. Apply Definition 38.342 to the total window. It identifies the endpoint $\mathsf{Rat}^{\uparrow}$ rows, denominator rows, common e-one tail readback, finite ledger rows, dyadic source rows, stream-name rows, unary history witnesses, and hsame transport rows available at the total window.

For each segment (n_i, r_i) supplied by the partition spine, apply Definition 38.342 again to the segment window $\mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i)$. The partition-spine ledger records that the segment anchors and radii are obtained from the same unary radius histories, append displays, and Cont rows used to display the total bound, so the segment reads name only the finite endpoint and dyadic source rows already named by the total presentation.

It remains to identify the primitive boundary. Apply Definition 38.314 for the endpoint package and dyadic finite-window source exhaustion. Apply Definition 38.392 for the carrier, classifier, stability, ledger, and seal rows of the primitive stream-prefix boundary. These two packages contain exactly the rows listed in the statement and exclude quotient, completion, metric-completeness, arithmetic, order, Classical, and standard-bridge data, so the partitioned and total presentations have the same scoped $\mathsf{Real}^{\uparrow}$ read surface. □

Theorem 38.310 (RealUp finite-window partition stability package)**.** *For rational-observation streams $x, x', y, y', z : \mathbb{N} \to \mathsf{Hist}$, an anchor n, a total bound w, and a certified finite partition spine P over (n, w) with segments (n_i, r_i), the four stability rows of* $\mathsf{RealStreamWindowClassifier}$ *commute with the correspondence between the total window at (n, w) and the segment windows named by P.*

Concretely, both the total presentation $\mathsf{RealStreamWindowClassifier}(-, -, n, w)$ *and the segment presentation*

$$\bigwedge_{(n_i, r_i) \in P} \mathsf{RealStreamWindowClassifier}(-, -, n_i, r_i)$$

expose the same scoped stability package:

1. *reflexivity from endpoint carrier rows for each displayed observation;*
2. *symmetry of the window classifier;*
3. *transitivity through the middle stream y;*
4. *pointwise* hsame *transport from (x, y) to (x', y') over the displayed observations.*

The exposed certificate boundary in either presentation is exactly the scoped read surface of Definition 38.309: endpoint $\mathsf{Rat}^{\uparrow}$ *rows,* $\mathsf{DyadicRatCore}$ *observation rows,* $\mathsf{RegSeqRat}$ *spine rows,* $\mathsf{StreamName}^{\uparrow}$ *request names,* $\mathsf{UnaryHistory}$ *witnesses,* hsame *transport rows, and the finite ledger rows named by the certified partition spine.*

Proof. First identify the two window presentations. Apply Definition 38.308 to the certified spine P. It gives the equivalence between the total classifier at (n, w) and the finite conjunction of segment classifiers at the anchors and radii listed by P.

Next apply Definition 38.171 to the total window. Its four rows give reflexivity from the carrier observations, symmetry, transitivity through y, and pointwise hsame transport from (x, y) to (x', y') on every offset $k \leq w$.

For the segment presentation, apply the same stability package to each segment window $\mathsf{RealStreamWindowClassifier}(-, -, n_i, r_i)$. The spine ledger records that every displayed segment offset is one of the observations in the total window, and the finite fold records the reverse containment by collecting the segment windows back into the total classifier.

Finally apply Definition 38.309. It identifies the public rows exposed by the total presentation and by the partitioned presentation as the same endpoint $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}$, $\mathsf{RegSeqRat}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UnaryHistory}$, hsame, and ledger rows. Therefore transporting any of the four stability rows across the fold does not expose additional data and does not lose any scoped row from the listed $\mathsf{Real}^{\uparrow}$ boundary. □

Theorem 38.311 (RealUp finite-window partition associative read surface)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ over the same* RealStreamWindowClassifier*, let P and Q be certified finite partition spines over the same total window (n, w). Suppose P and Q have the same ordered leaves (n_i, r_i) and differ only by associative rebracketing of the partition spine. Then the two certified presentations expose the same consumer read surface for the window.*

Concretely, the endpoint $\mathsf{Rat}^{\uparrow}$ *classifier rows,* DyadicRatCore *observation rows,* RegSeqRat *spine rows,* $\mathsf{StreamName}^{\uparrow}$ *request names,* UnaryHistory *witnesses,* hsame *transport rows,* Cont *routes, and finite ledger rows determined from P are exactly the rows determined from Q. No completion object, quotient, metric-completeness certificate, Classical choice row, arithmetic or order primitive, or bridge row is exported by either bracketing.*

Proof. Apply Definition 38.308 to the certified spine P. It identifies the total classifier RealStreamWindowClassifier(x with the finite conjunction of the segment classifiers listed by the common ordered leaves.

Apply the same fold theorem to Q. Since the leaves are the same and only the associative spine bracketing changes, both folds land in the same finite conjunction of segment windows.

Now compare the consumer rows through Definition 38.309. The theorem reads each folded presentation as the same endpoint $\mathsf{Rat}^{\uparrow}$, DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, UnaryHistory, hsame, Cont, and finite ledger boundary attached to the displayed leaves.

Finally use Definition 38.310. The stability rows transport across the two fold presentations without exposing any row outside the displayed $\mathsf{Real}^{\uparrow}$ boundary. Thus associative rebracketing of the certified finite partition spine preserves exactly the same consumer read surface. □

Theorem 38.312 (Real stream finite-window partition refinement transitivity)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a total anchor n, a total bound w, and certified finite partition spines P, Q over (n, w), suppose every segment of Q is displayed in the ledger as a consecutive union of segments of P. If every P-segment (n_i, r_i) carries* RealStreamWindowClassifier(x, y, n_i, r_i)*, then every Q-segment (m_j, s_j) carries* RealStreamWindowClassifier(x, y, m_j, s_j)*.*

Equivalently, the finite ledger exposed by the fine partition P refines the coarser partition ledger Q without changing any endpoint $\mathsf{Rat}^{\uparrow}$ *classifier, dyadic source row, stream-name row, unary witness,* hsame *transport row, or continuation ledger entry at the visible boundary.*

The ten-field fit is: name RealStreamWindow *refinement transitivity; BEDC source* $\mathsf{BHist} + \mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} +$ RealStreamWindowClassifier*; carrier the certified finite partition spine; classifier* RealStreamWindowClassifier*; pattern consecutive-union refinement of window spines; stability the four classifier rows exposed by Definition 38.310; ledger the refinement witness from fine rows to coarse rows; exactness the unchanged* $\mathsf{Rat}^{\uparrow}$ *endpoint classifier surface; core theorem Definition 38.312; downstream consumer* $\mathsf{Real}^{\uparrow}$ *finite-window certificate closure.*

Proof. Induct on the coarser spine Q. The empty spine has no segment to certify, and the ledger refinement witness contributes no visible row.

For a nonempty coarser spine, read the head segment (m_j, s_j) as the consecutive subspine of P selected by the refinement ledger. Apply Definition 38.308 to that subspine. The given classifiers on the fine P-segments fold into RealStreamWindowClassifier(x, y, m_j, s_j).

Now repack the visible rows for this coarse segment. Apply Definition 38.309 to identify the endpoint $\mathsf{Rat}^{\uparrow}$ rows, dyadic source rows, stream-name rows, unary witnesses, hsame transport rows, and finite ledger entries exposed by the fine subspine and by the coarse segment as the same scoped read surface.

The stability fields are transported across this repacking by Definition 38.310. Removing the consumed fine subspine leaves the tail of P refining the tail of Q, so the induction hypothesis

certifies every remaining coarse segment. Combining the head certificate with the tail certificates gives the claimed refinement transitivity. □

Theorem 38.313 (Real stream finite-window common refinement exactness)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a total anchor n, a total bound w, and certified finite partition spines P, Q over (n, w), suppose every segment of P and every segment of Q carries* $\mathsf{RealStreamWindowClassifier}(x, y, -, -)$*. Then there is a certified finite partition spine R over (n, w) such that:*

1. *every segment of R carries* $\mathsf{RealStreamWindowClassifier}(x, y, -, -)$*;*
2. *folding the R-segments over each P-block recovers the P presentation, and folding the R-segments over each Q-block recovers the Q presentation;*
3. *the public endpoint rows read from the P, Q, and R presentations are exactly the same* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}$*,* $\mathsf{RegSeqRat}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{UnaryHistory}$*,* hsame*, and finite* Cont *ledger rows.*

Thus two independent certified finite-window presentations of the same RealUp window admit a common refinement whose fold and visible read surface are exact for both presentations.

Proof. Build R by simultaneous induction on the two certified finite spines. The base case is the singleton overlap of the two total-window spines, where the given classifier row is already the required segment row.

In the induction step, compare the first remaining P-segment and Q-segment by their unary endpoint ledgers. If one endpoint is earlier, split the longer current segment at that displayed endpoint; if the endpoints agree, take their common terminal segment. The construction uses only the unary radius histories, accumulated anchors, and Cont rows already present in the two certified spines, so the resulting list is again a certified finite partition spine over (n, w).

For each refined block, apply Definition 38.308. It folds the R-segments contained in a displayed P-block back to the corresponding P-segment classifier, and the same argument folds the R-segments contained in a displayed Q-block back to the corresponding Q-segment classifier.

Next apply Definition 38.312 from R to P and from R to Q. These two refinement readings show that replacing either original presentation by the common refinement does not add or remove any classifier row at the visible boundary.

Finally apply Definition 38.309 to the three presentations and Definition 38.310 to transport the stability rows across the two fold comparisons. The read surfaces are therefore the same endpoint $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}$, $\mathsf{RegSeqRat}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UnaryHistory}$, hsame, and finite Cont ledger rows, which is the asserted exactness. □

Theorem 38.314 (RealUp finite-window exhaustion obligation)**.** *At the RealUp obligation boundary, the finite-window observation field is supplied by the conjunction of* RealStreamClassifier_finite_prefix_endpoint RealStreamPrefixClassifier_truncated_endpoint_denominator_package, RealStreamPrefixClassifier_truncated_endpo *and the* RegSeqRat *finite-window dyadic source exhaustion package. Consequently every selected finite* $\mathsf{Real}^{\uparrow}$ *observation window exposes endpoint* $\mathsf{Rat}^{\uparrow}$ *classifiers, positive unary denominators, nonempty unary endpoint rows, common e-one tail readback, and exactly the* DyadicRatCore *source rows consumed by the* $\mathsf{Real}^{\uparrow}$ *seal, with no data outside the carriers, classifiers, stability rows, and ledger rows listed in Definition 38.285.*

Lean4 ✓ `BEDC.Derived.RealUp.RealupFiniteWindowExhaustionObligation`

Proof. First collect the endpoint denominator face. Apply Definition 38.541 to the selected finite prefix. It supplies the endpoint $\mathsf{Rat}^{\uparrow}$ classifier, the positive unary denominator witnesses, the unary endpoint rows, and the nonempty endpoint exclusions needed at the selected observation.

Next pass to truncated prefixes. Apply the truncated endpoint denominator package recorded by Definition 38.293. This transports the same endpoint denominator data through the finite tail presentation used by the stream-prefix carrier and classifier rows.

Now read the selected one-headed tails. Apply Definition 38.854; the truncated endpoint package gives the hypotheses needed to expose common e-one tails at the two selected endpoint histories.

Finally identify the consumed source rows. Apply Definition 38.564. It states that the selected bounded RegSeqRat observation reads exactly the two DyadicRatCoreObservationRow endpoint sources and the scoped carrier, classifier, stability, and finite ledger faces named by the dyadic source package. These are precisely the rows allowed by Definition 38.285; hence the finite-window observation field consumes no data outside that boundary. □

Theorem 38.315 (RealUp StreamName RegSeqRat dyadic handoff exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *finite-window observation whose StreamName face is a* RegSeqRat *handoff factors through exactly the displayed finite source surface: the* RatStreamNameFiniteWindowClassifier *row, the selected* ProbeBundle*,* InBundle*, and* UnaryHistory *rows, the endpoint* $\mathsf{Rat}^{\uparrow}$ *denominator rows, the accepted* RegSeqRat *spine, the DyadicRatCore source rows, the shown* hsame/Cont/Pkg *ledger, and the* RealLedgerPolicy *row.*

This factorization is finite-row only. It exports no completion object, quotient representative, Classical *representative, metric-completeness certificate, arithmetic or order operation, or standard-bridge row.*

Proof. First apply Definition 38.314. It reduces the selected $\mathsf{Real}^{\uparrow}$ observation to endpoint denominator rows, positive unary denominator witnesses, common e-one tail readback, and the DyadicRatCore source rows supplied by the accepted RegSeqRat finite-window dyadic package.

Next apply Definition 126.268. The StreamName face of the same observation contributes only the finite-window classifier row, the shared ProbeBundle, the selected InBundle/UnaryHistory rows, the pointwise RatHistoryClassifier rows, and the displayed hsame/Cont/Pkg provenance that feeds the accepted spine.

Finally use Definition 126.217. The tail-window packet admits no consumer row beyond the displayed classifier, bundle, unary-history, pointwise rational-history, RegSeqRat, DyadicRatCore, and RealLedgerPolicy rows. Therefore the $\mathsf{Real}^{\uparrow}$ finite-window observation factors through exactly the stated handoff surface, and completion, quotient, Classical representative, metric-completeness, arithmetic or order, and standard-bridge rows remain outside the finite ledger. □

Theorem 38.316 (RealUp refinement envelope reindexed consumer invariance)**.** *Let E be a* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over a fixed* $\mathsf{StreamName}^{\uparrow}$ *source, and let P be a* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window package whose selected rational classifier rows feed a certified window W carried by E. The downstream* RegSeqRat *handoff determined by P and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal read determined by W factor through the same envelope-facing rows:* RatStreamNameFiniteWindowClassifier *rows, the* ProbeBundle*,* InBundle*, and* UnaryHistory *rows, the endpoint denominator rows, the* DyadicRatCore *source rows, the shared* hsame *ledger, and the* RealLedgerPolicy *boundary. Consequently the reindexed handoff and the carried seal expose one public read surface, with no exported host index, quotient witness, completion object, or unrecorded stream row.*

Proof. Begin with the reindexed package P. By Definition 126.155, its selected rational rows are exhausted by the reindexed $\mathsf{StreamName}^{\uparrow}$ finite-window classifier rows, the probe and unary-history rows, the endpoint denominator rows, and the dyadic source rows visible in the window.

Next apply Definition 38.324. The RegSeqRat consumer handoff reads those rows through the same shared hsame ledger and the same RealLedgerPolicy boundary used by the carried $\mathsf{Real}^{\uparrow}$ window.

Finally use Definition 38.332 for the carried window W inside E. That theorem forces both the RegSeqRat handoff and the finite-window $\mathsf{Real}^{\uparrow}$ seal through exactly the rows stored in the refinement envelope.

The listed envelope rows therefore form the common public surface. Since the envelope boundary contains no slot for a host index, quotient witness, completion object, or unrecorded stream datum, none of those objects can be exported by either consumer. □

Theorem 38.317 (RealUp StreamName window determinacy boundary)**.** *Let two finite-window observations be admitted by the* $\mathsf{Real}^{\uparrow}$ *seal through the same selected unary observation and the same* RatStreamNameFiniteWindowClassifier *endpoint package. Then the two observations expose the same endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the same positive denominator witnesses, the same common e-one tail readbacks, and the same* DyadicRatCore *source rows, up to the displayed* hsame *ledger rows permitted by Definition 38.285. No host* Nat *index, host function, quotient, or unrecorded stream object is available to separate the two exposed windows.*

Proof. Begin with the two accepted finite-window observations. Apply Definition 38.320 to each one. The only exposed rows are the selected unary observation, the pointwise RatStreamNameFiniteWindowClassifier row, the endpoint denominator rows, and the dyadic source rows named by the finite-window obligation.

Now use the common StreamName package. The structural laws of Definition 126.25 and the selected full-endpoint package of Definition 126.27 identify the two selected observations and their endpoint classifier rows by the displayed hsame ledger rows. Thus the endpoint $\mathsf{Rat}^{\uparrow}$ classifiers and positive denominator witnesses are the same readable rows of the package.

Finally repack through Definition 38.314. The common endpoint package supplies the hypotheses for the common e-one tail readbacks, and the dyadic source exhaustion supplies the same DyadicRatCore source rows. Since Definition 38.285 permits only these carrier, classifier, stability, and finite ledger rows, the two windows have the same exposed RealUp seal ledger. □

Theorem 38.318 (RealUp finite-window coverage and determinacy package)**.** *Every finite* $\mathsf{Real}^{\uparrow}$ *observation admitted through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window seal factors through one package of public rows: the* RatStreamNameFiniteWindowClassifier *row, the selected* ProbeBundle *and* InBundle *rows, the* UnaryHistory *observation, the endpoint* $\mathsf{Rat}^{\uparrow}$ *denominator rows, the* DyadicRatCore *source rows, the displayed* hsame *ledger, and the finite* RealLedgerPolicy *boundary. Moreover, if two admitted reads use the same selected unary observation and the same endpoint package, then their exposed dyadic source rows agree up to that displayed* hsame *ledger; and any certified unary reindexing of the selected observation preserves exactly the same package.*

Proof. First apply Definition 38.314. It reduces an admitted finite $\mathsf{Real}^{\uparrow}$ window to the endpoint denominator rows, the common e-one tail readback, and the DyadicRatCore source rows visible through the finite seal.

Next apply Definition 38.320. This identifies the StreamName face of the same window: the RatStreamNameFiniteWindowClassifier row, selected probe and bundle rows, unary-history observation, endpoint $\mathsf{Rat}^{\uparrow}$ denominator rows, dyadic source rows, and finite RealLedgerPolicy all sit on the same displayed hsame ledger.

For two reads with the same selected unary observation and endpoint package, use Definition 38.317. The shared StreamName package fixes the endpoint classifiers, denominator witnesses, common e-one tail readbacks, and dyadic source rows up to the displayed hsame ledger.

Finally apply Definition 38.321. Certified unary reindexing transports the selected observation and classifier row while repacking the same endpoint, source, ledger, and finite policy rows. Since no quotient, completion object, host index, or choice datum is exported by these four boundaries, the package is exactly the finite-window coverage and determinacy surface stated above. □

Theorem 38.319 (RealUp open-phase audit boundary)**.** *Every paper-side Phase Real and Completion audit read that reaches* $\mathsf{Real}^{\uparrow}$ *through finite windows factors through exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, the* RegSeqRat *dyadic source exhaustion, the* DyadicRatCore *source rows, the endpoint* $\mathsf{Rat}^{\uparrow}$ *denominator rows, the displayed* hsame, Cont, *and* Pkg *ledger, and the local* NameCert *rows. Conversely this package is sufficient for the L10 exit boundary for those reads. It exports no quotient, selected limit, completion theorem, countable-choice row, metric-completeness datum, arithmetic or order extension, or standard bridge.*

Proof. Apply Definition 38.314 to reduce any accepted $\mathsf{Real}^{\uparrow}$ finite-window read to endpoint denominator rows, common tail readback, and dyadic source rows.

Next apply Definition 38.315 and Definition 38.318. These identify the same read through the StreamName finite-window classifier, RegSeqRat dyadic source exhaustion, DyadicRatCore source rows, endpoint $\mathsf{Rat}^{\uparrow}$ rows, displayed ledger, package provenance, and local naming rows.

The cited StreamName, RegSeqRat, and dyadic source rows already supply the forward package read by the roadmap boundary. The non-export clauses follow from the same finite-window exhaustion: none of quotient formation, selected limit choice, completion closure, countable choice, metric completeness, arithmetic or order extension, or a standard bridge is a displayed row of the package. □

Theorem 38.320 (RealUp StreamName finite-window ledger boundary)**.** *Every finite-window observation admitted by the* $\mathsf{Real}^{\uparrow}$ *seal factors through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, the* $\mathsf{Rat}^{\uparrow}$ *endpoint denominator package, and the* RegSeqRat *dyadic source exhaustion rows. Its ledger is exactly the selected unary observation, the pointwise* RatStreamNameFiniteWindowClassifier *row, the endpoint denominator rows, and the dyadic source rows named by Definition 38.314. No host* Nat *index, host function, quotient, or unrecorded infinite stream object is exposed at this boundary.*

Lean4 ✓ `BEDC.Derived.RealUp.RealupStreamnameFiniteWindowLedgerBoundary`

Proof. Start with an observation accepted by the $\mathsf{Real}^{\uparrow}$ seal. Apply Definition 38.314; it reduces the accepted finite window to the denominator package, truncated endpoint readback, and RegSeqRat dyadic source rows allowed by the scoped seal.

The finite-window stream-name component is supplied by Definition 126.22, Definition 126.25, and Definition 126.27. Together these rows identify the selected unary observation, the bundled pointwise classifier, and the endpoint denominator package read by the window.

Finally use Definition 38.285. That boundary permits only the listed carrier, classifier, stability, and finite ledger rows. The StreamName rows and the dyadic source rows therefore repack into the exact ledger stated above, while host Nat data, host functions, quotients, and unrecorded infinite objects have no exported row to inspect. □

Theorem 38.321 (RealUp finite-window reindexed classifier transport)**.** *Let W be a selected finite* $\mathsf{Real}^{\uparrow}$ *observation window admitted through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window seal ledger. Suppose W' is obtained from W by a certified* $\mathsf{StreamName}^{\uparrow}$ *unary reindexer, and suppose the transported* RatStreamNameFiniteWindowClassifier *row and the selected endpoint denominator package are present at W'. Then the transported window exposes the same* $\mathsf{Real}^{\uparrow}$ *finite seal ledger boundary as W: endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, positive denominator witnesses, common e-one tail readback,*

DyadicRatCore *source rows, unary-history rows,* hsame *transport, and the finite* RealLedgerPolicy*. The transported surface exposes no quotient, completion, host* Nat *index, host function,* Classical *datum, metric-completeness certificate, or standard bridge.*

Proof. Start with the certified unary reindexer. Apply Definition 126.19 to transport the selected $\mathsf{StreamName}^{\uparrow}$ observation rows and their hsame stability ledger from W to W'.

Next apply Definition 126.24. The transported finite-window classifier at W' covers exactly the reindexed unary observation rows and the selected endpoint package, so no host Nat index or host function is exported by the StreamName face.

Now identify the RealUp ledger carried by the source window. By Definition 38.320, the original surface of W contains only the selected unary observation, the RatStreamNameFiniteWindowClassifier row, endpoint $\mathsf{Rat}^{\uparrow}$ classifier rows, positive denominator witnesses, common e-one tail readback, DyadicRatCore source rows, and the finite RealLedgerPolicy rows allowed by the seal.

Finally repack the transported rows through Definition 38.314. The scoped boundary permits only carrier, classifier, stability, source, and finite ledger rows, so the reindexed read surface is the same RealUp finite seal ledger boundary and has no quotient, completion, Classical datum, metric-completeness certificate, or standard bridge to inspect. □

Theorem 38.322 (RealUp finite-window feed reindexing stability)**.** *Let W be a selected finite* $\mathsf{Real}^{\uparrow}$ *observation window admitted through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window seal, and let W' be obtained from W by a certified unary* $\mathsf{StreamName}^{\uparrow}$ *reindexer. Suppose the reindexer preserves the transported* RatStreamNameFiniteWindowClassifier *row at W'. Then the public feed read at W' is the same feed read at W: the selected* UnaryHistory *observation, the* ProbeBundle *and* InBundle *rows of the window, the endpoint* $\mathsf{Rat}^{\uparrow}$ *denominator package, the* RegSeqRat *handoff rows, the* DyadicRatCore *source rows, the displayed* hsame *ledger, and the finite* RealLedgerPolicy *boundary.*

The feed comparison exports no quotient row, completion object, host Nat *index, host function, metric-completeness certificate, or standard bridge claim.*

Proof. First transport the finite StreamName face. Apply Definition 126.19 to move the selected unary observation and its hsame ledger along the certified unary reindexer. Then apply Definition 126.24 to identify the transported finite-window classifier with exactly the reindexed observation rows and endpoint denominator package.

Next pass from classifier transport to the RealUp feed boundary. By Definition 38.321, the reindexed classifier row exposes the same finite seal ledger boundary as the source window, including the endpoint $\mathsf{Rat}^{\uparrow}$ denominator rows, DyadicRatCore source rows, displayed hsame transport, and finite RealLedgerPolicy rows.

The RegSeqRat source face is the same public feed. Apply Definition 38.324 to identify the selected StreamName rational rows with the RegSeqRat dyadic handoff rows read by the finite window.

Finally use Definition 38.314. Its boundary lists all carrier, classifier, source, and finite ledger rows admitted by the selected window, including the ProbeBundle and InBundle entries. Since no admitted row is a quotient, completion, host index, host function, metric-completeness certificate, or standard bridge, the transported feed has exactly the same public surface and no additional exported datum. □

Theorem 38.323 (RealUp StreamName reindexing composition boundary)**.** *Let W be a selected finite* $\mathsf{Real}^{\uparrow}$ *observation window admitted through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window seal ledger. Suppose two certified* $\mathsf{StreamName}^{\uparrow}$ *unary reindexers are applied successively to W, and suppose the same selected endpoint package is also read through the single certified composite reindexer. Then*

the successive read and the composite read expose the same finite seal ledger: selected unary observation rows, the RatStreamNameFiniteWindowClassifier *row, endpoint* $\mathsf{Rat}^{\uparrow}$ *denominator rows, positive denominator witnesses, common e-one tail readback,* DyadicRatCore *source rows,* hsame *transport, and the finite* RealLedgerPolicy*.*

In particular, the composite surface has no additional quotient, completion, host Nat *index, host function, unrecorded stream object,* Classical *datum, metric-completeness certificate, or standard bridge beyond the rows already exposed by the two successive unary transports.*

Proof. Start with the two certified unary reindexers. Apply Definition 126.19 to each reindexing step, and use the $\mathsf{StreamName}^{\uparrow}$ reindexing composition law to identify the successive selected unary observation rows with the rows selected by the single composite reindexer.

Next apply Definition 38.321 to the first transported window and again to the second transported window. The finite classifier, endpoint denominator package, common e-one tail readback, and DyadicRatCore source rows are carried componentwise through the same hsame transport ledger.

Finally compare both transported reads with Definition 38.320. That boundary lists the whole finite $\mathsf{Real}^{\uparrow}$ seal ledger, so the successive read and the composite read have the same public rows and no further host-level reindexing datum to inspect. □

Theorem 38.324 (RealUp StreamName-RegSeqRat reindexed ledger handoff)**.** *Let* $\mathcal{W}$ *be a* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window package whose selected rational classifier rows feed a finite* $\mathsf{Real}^{\uparrow}$ *observation window* W*. Then the package exposes exactly the* $\mathsf{Real}^{\uparrow}$ *finite seal ledger boundary for* W*: the selected unary observation row, the* RatStreamNameFiniteWindowClassifier *row, the endpoint denominator rows, and the* RegSeqRat *dyadic source rows selected by the finite window. It exposes no host* Nat *index, host function, quotient, completion object, or unrecorded infinite stream.*

Proof. Start with the displayed reindexed window package. Apply Definition 126.155 to decompose its visible rows into the selected unary observation, the transported RatStreamNameFiniteWindowClassifier row, the endpoint denominator rows, and the RegSeqRat source rows supplied by the package.

Next use Definition 126.79. It is the common handoff from the StreamName finite-window face to the RegSeqRat dyadic source face, so the selected rational classifier rows are the same rows read by the finite $\mathsf{Real}^{\uparrow}$ observation.

Now apply Definition 38.320. The RealUp seal boundary admits exactly the selected unary observation, the finite StreamName classifier row, the endpoint denominator rows, and the dyadic source rows as public ledger entries.

Finally use Definition 126.156. It excludes the host index, host map, quotient, completion object, and unrecorded stream data from the reindexed package, so the package and the $\mathsf{Real}^{\uparrow}$ finite seal expose the same finite boundary. □

Theorem 38.325 (RealUp finite-window exhaustion subwindow closure)**.** *Whenever a* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion package is exposed for a* $\mathsf{StreamName}^{\uparrow}$ *finite window, every certified subwindow obtained by the* $\mathsf{StreamName}^{\uparrow}$ *finite-window subwindow restriction exposes the same* $\mathsf{Rat}^{\uparrow}$ *endpoint classifier, the same ledger boundary, and the same* $\mathsf{Real}^{\uparrow}$ *read surface restricted to that subwindow. The restricted package reads only the subwindow observation rows, the inherited* RatStreamNameFiniteWindowClassifier *rows, the endpoint denominator rows, and the* DyadicRatCore *source rows already named by the ambient window.*

Proof. Start with the ambient finite-window exhaustion package. Apply Definition 38.314 to identify its $\mathsf{Rat}^{\uparrow}$ endpoint classifiers, positive denominator witnesses, common e-one tail readbacks, and DyadicRatCore source rows.

Restrict the StreamName component by Definition 126.9. The certified subwindow keeps the same pointwise RatStreamNameFiniteWindowClassifier rows and only changes the displayed finite observation domain.

Next use Definition 38.317. It rules out a second endpoint reading for the restricted window once the selected StreamName package and endpoint denominator rows are fixed.

Finally apply Definition 38.320. The ledger boundary remains exactly the selected unary observation, the restricted StreamName classifier row, the endpoint denominator rows, and the dyadic source rows. Repacking these rows gives the same $\mathsf{Real}^{\uparrow}$ read surface restricted to the certified subwindow. □

Theorem 38.326 (RealUp finite-window exhaustion overlap closure)**.** *Let two* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion packages expose certified* $\mathsf{StreamName}^{\uparrow}$ *subwindows. If the two restricted windows have a certified common* $\mathsf{StreamName}^{\uparrow}$ *subwindow overlap, then the overlap exposes the same inherited* RatStreamNameFiniteWindowClassifier *rows, the same endpoint denominator rows, the same common e-one tail readbacks, and the same* DyadicRatCore *source rows, up to the* hsame *ledger permitted by Definition 38.285. The overlap certificate does not expose host* Nat *indices, host functions, quotients, or unrecorded infinite stream data.*

Proof. Apply Definition 38.325 to each ambient finite-window exhaustion package. Each restricted window therefore reads only its subwindow observation rows, inherited RatStreamNameFiniteWindowClassifier rows, endpoint denominator rows, common e-one tail readbacks, and DyadicRatCore source rows.

Now restrict both sides to the certified common $\mathsf{StreamName}^{\uparrow}$ overlap. The overlap has the same selected StreamName window on both readings, so Definition 38.317 identifies the endpoint $\mathsf{Rat}^{\uparrow}$ classifier rows, positive denominator witnesses, common e-one tail readbacks, and dyadic source rows through the displayed hsame ledger.

Finally apply Definition 38.320 on the overlap itself. The only exported ledger rows are the selected unary observation, the overlap RatStreamNameFiniteWindowClassifier rows, the endpoint denominator rows, and the dyadic source rows named by the finite-window exhaustion boundary.

Since Definition 38.285 admits no exported row for host Nat indices, host functions, quotient witnesses, or unrecorded infinite stream data, no further datum can distinguish the two restricted readings on their common overlap. □

Theorem 38.327 (RealUp finite-window two-step restriction coherence)**.** *Suppose a* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion package is exposed for a* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and suppose* U *is a certified* $\mathsf{StreamName}^{\uparrow}$ *subwindow of* W *while* V *is a certified* $\mathsf{StreamName}^{\uparrow}$ *subwindow of* U*. The* $\mathsf{Real}^{\uparrow}$ *read surface obtained by restricting first from* W *to* U *and then from* U *to* V *is the same public finite* BHist *surface as the direct restriction from* W *to* V*. Both readings expose the same* RatStreamNameFiniteWindowClassifier *rows, endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, positive denominator witnesses, common e-one tail readbacks,* DyadicRatCore *source rows,* hsame *ledger, and* RealLedgerPolicy *boundary.*

Proof. Apply Definition 38.325 to the ambient window W and the certified subwindow U. This identifies the U reading as the inherited finite-window package whose public rows are exactly the RatStreamNameFiniteWindowClassifier rows, endpoint denominator rows, common e-one tail readbacks, and dyadic source rows of W restricted to U.

Apply Definition 38.325 again to the U package and the certified subwindow V. The two-step reading therefore has only the inherited V observation rows, the endpoint $\mathsf{Rat}^{\uparrow}$ classifier rows, positive denominator witnesses, common e-one tail readbacks, and DyadicRatCore source rows already named by the ambient package.

For the direct W to V reading, use Definition 38.317. The selected StreamName window and endpoint denominator rows for V are the same in the direct and two-step presentations, so the endpoint classifiers, denominator witnesses, readbacks, and source rows are identified through the displayed hsame ledger.

Finally apply Definition 38.320 to the common V surface. The public finite BHist ledger is exactly the selected observation, the RatStreamNameFiniteWindowClassifier rows, the endpoint rows, the dyadic source rows, and the RealLedgerPolicy boundary. Hence the two-step restriction and the direct restriction expose the same $\mathsf{Real}^{\uparrow}$ read surface. □

Theorem 38.328 (RealUp finite-window partition exactness)**.** *Suppose a* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion package is exposed for a* $\mathsf{StreamName}^{\uparrow}$ *window* W*. Let* $(W_i)_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *subwindows of* W *such that every certified unary observation row of* W *is carried by at least one* W_i*, and such that every pairwise intersection exposed by two selected subwindows is certified by the overlap boundary of Definition 38.326. Then the* $\mathsf{Real}^{\uparrow}$ *read surface assembled from the finite family* $(W_i)_{i\in I}$ *is the same public finite* BHist *surface as the direct read surface of* W*. The two surfaces have the same* RatStreamNameFiniteWindowClassifier *rows, endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, positive denominator witnesses, common e-one tail readbacks,* DyadicRatCore *source rows,* hsame *ledger, and* RealLedgerPolicy *boundary.*

Proof. Apply Definition 38.325 to each certified subwindow W_i. Each local read therefore contains only the inherited unary observation rows, finite-window classifier rows, endpoint denominator rows, common e-one tail readbacks, and dyadic source rows coming from the ambient window W.

On every certified overlap $W_i \cap W_j$, use Definition 38.326. It identifies duplicate observations and their endpoint $\mathsf{Rat}^{\uparrow}$ classifier rows through the same hsame ledger, so the finite family contributes one public observation row for each certified observation of W rather than a separate row for each covering subwindow.

If a selected observation is reached through a smaller certified refinement, Definition 38.327 identifies the refined read with the direct restriction from W. Thus changing the subwindow presentation does not change the visible read surface.

Finally repack the covered observation rows by Definition 38.320. The coverage hypothesis supplies every unary observation row of W, and the overlap and two-step coherence rows remove duplicate presentations. The assembled surface is therefore exactly the direct finite-window surface of W. □

Theorem 38.329 (RealUp finite-window directed refinement seal)**.** *Suppose two* $\mathsf{Real}^{\uparrow}$ *finite-window exhaustion packages are exposed over the same* $\mathsf{StreamName}^{\uparrow}$ *source, with certified windows* W *and* W'*. Suppose also that there is a certified common refinement, given by a finite family of* $\mathsf{StreamName}^{\uparrow}$ *subwindows whose union covers every certified unary observation row of both* W *and* W'*, and whose pairwise overlaps satisfy Definition 38.326. Then the two finite-window presentations expose the same public* BHist *read surface. In particular, they have the same* RatStreamNameFiniteWindowClassifier *rows, endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, positive denominator witnesses, common e-one tail readbacks,* DyadicRatCore *source rows,* hsame *ledger, and* RealLedgerPolicy *boundary, and neither presentation exports host indices, quotient witnesses, or unrecorded infinite stream data.*

Proof. Apply Definition 38.328 to the window W and the certified common refinement. The coverage hypothesis says that the refinement carries every certified unary observation row visible from W, so the refinement assembly is the same public finite BHist surface as the direct W presentation.

Apply Definition 38.328 again to the window W'. The same refinement family covers every certified unary observation row visible from W', so the refinement assembly is also the same public finite BHist surface as the direct W' presentation.

It remains to compare the two uses of the refinement. When a refinement subwindow is reached through either ambient window, use Definition 38.327 to identify the restricted read with the direct refinement read. On intersections between two refinement subwindows, use Definition 38.326 to identify the duplicate classifier rows, denominator witnesses, readbacks, and dyadic source rows through the same hsame ledger.

Finally apply Definition 38.320 to the common refinement surface. Its ledger contains exactly the selected observation rows, endpoint $\mathsf{Rat}^{\uparrow}$ rows, positive denominator witnesses, common e-one tail readbacks, DyadicRatCore rows, and RealLedgerPolicy boundary. Therefore both finite-window presentations expose the same public surface, with no additional host index, quotient, or infinite-stream datum available to distinguish them. □

Definition 38.330 (RealUp finite-window refinement envelope)**.** *A* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over one* $\mathsf{StreamName}^{\uparrow}$ *source is a finite* BHist *package whose rows consist of certified* $\mathsf{StreamName}^{\uparrow}$ *finite windows, a common-refinement package for every ordered pair of carried windows, the overlap-closure witnesses for each refinement family, the two-step restriction coherence witnesses for every displayed subwindow chain, the endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the positive denominator witnesses, the* DyadicRatCore *source rows, the shared* hsame *ledger identifying duplicate presentations, and the* RealLedgerPolicy *boundary. The envelope exports only these finite rows and their* Pkg *provenance.*

Theorem 38.331 (RealUp finite-window refinement envelope coherence)**.** *Let E be a* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over a fixed* $\mathsf{StreamName}^{\uparrow}$ *source. If two certified windows W and W' are carried by E, then the public read surface exposed from W is the same finite* BHist *read surface as the one exposed from W'. The common surface contains exactly the envelope's* RatStreamNameFiniteWindowClassifier *rows, endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier rows, positive denominator witnesses, common e-one tail readbacks,* DyadicRatCore *source rows, shared* hsame *ledger, and* RealLedgerPolicy *boundary, with no host index, quotient witness, or unrecorded infinite stream datum.*

Proof. Choose the common-refinement package in E for the ordered pair (W, W'). The package supplies a finite family of certified subwindows covering every certified unary observation row of both windows, together with the required overlap-closure witnesses.

Apply Definition 38.329 to this pair. The seal reduces the comparison of the two ambient presentations to the public surface assembled from their displayed common refinement.

The assembly through the common refinement is exact by Definition 38.328. Whenever a row is read through an intermediate refinement before reaching the displayed subwindow, Definition 38.327 identifies that reading with the direct restriction used by the refinement surface.

Finally apply Definition 38.320. The only public rows left in the envelope are the selected finite-window observation rows, endpoint $\mathsf{Rat}^{\uparrow}$ classifier rows, positive denominator witnesses, common e-one tail readbacks, dyadic source rows, the shared hsame ledger, and the RealLedgerPolicy boundary. Since that boundary has no slot for a host index, quotient witness, or unrecorded infinite stream datum, the two carried windows expose one public read surface. □

Theorem 38.332 (RealUp finite-window refinement envelope consumer coverage)**.** *Let E be a* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over a fixed* $\mathsf{StreamName}^{\uparrow}$ *source, and let W be any certified window carried by E. Every* RegSeqRat *handoff induced by W, and every finite-window* $\mathsf{Real}^{\uparrow}$ *seal*

read induced by W, factors through exactly the rows carried by E: the StreamName$^{\uparrow}$ *finite-window classifier rows, the* ProbeBundle, InBundle, *and* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the endpoint denominator rows, the* DyadicRatCore *source rows, the shared* hsame *ledger, and the* RealLedgerPolicy *boundary. Conversely, any such row family inside E repacks into the same* RegSeqRat *handoff and finite-window* Real$^{\uparrow}$ *seal surface. No completion object, quotient witness, host-index equality, or unrecorded stream datum is part of this consumer surface.*

Proof. First compare the chosen carried window with the envelope surface. Apply Definition 38.331 with W and any other carried representative used by the consumer. The comparison identifies the consumer-facing read with the envelope's finite public BHist surface, so the read cannot escape the rows named in the envelope.

Next expose the stream-name boundary. By Definition 38.320, a finite window contributes only its classifier rows, the certified probe and unary history rows, the endpoint denominator witnesses, the shared hsame ledger, and the RealLedgerPolicy boundary. These are precisely rows already stored in E.

For a RegSeqRat consumer, apply Definition 126.170. Its pointwise handoff uses the RatHistoryClassifier rows read from the same StreamName$^{\uparrow}$ finite window. Then apply Definition 38.564 to replace the dyadic source requirements by the displayed DyadicRatCore source rows and endpoint denominator rows carried by the envelope.

The reverse packing uses the same two displayed handoff theorems in the opposite direction of construction: the finite-window classifier, ProbeBundle, InBundle, UnaryHistory, pointwise rational classifier, denominator, dyadic-source, hsame, and ledger-policy rows assemble the public consumer interface. Since all cited boundaries are finite BHist rows, there is no constructor slot for completion, quotient identification, host-index equality, or unrecorded stream data. □

Theorem 38.333 (StreamName RegSeqRat window stability through RealUp prefixes)**.** *Let s, t, u, s', t' : BHist $\to$ BHist be* StreamName$^{\uparrow}$ *rows, let $\mathcal{P}$: ProbeBundle be finite, and let E be a* Real$^{\uparrow}$ *finite-window refinement envelope carrying the selected window over $\mathcal{P}$. Suppose the selected window carries the hypotheses needed for the self, symmetric, transitive, and transported outputs of* RatStreamNameFiniteWindowClassifier_stability_fields*: a carrier row for s, finite-window classifiers from s to t and from t to u, and pointwise* hsame *transports from s to s' and from t to t' on the bundled unary observations. Then the four stable windows preserve exactly the* RegSeqRat *handoff of Definition 126.170 and the finite-window* Real$^{\uparrow}$ *seal surface of Definition 38.332. No regularity row, completion object, quotient witness, host-index equality, or unrecorded stream row is added by passing the stable windows through the prefix envelope.*

Proof. First apply the StreamName finite-window stability fields to the selected window. They give the self window on s, the symmetric window from t to s, the transitive window from s to u, and the transported window from s' to t' over the same finite bundle $\mathcal{P}$.

Pass each of these four windows through Definition 126.170. The handoff reads only the StreamName carrier rows, the finite-window classifier row, the UnaryHistory rows, the InBundle rows, and the pointwise rational classifier rows selected by $\mathcal{P}$. Thus the downstream RegSeqRat interface sees the same pointwise packet for each stable window.

Finally apply Definition 38.332 to the envelope E. Its consumer-coverage statement identifies every induced RegSeqRat handoff and every finite-window Real$^{\uparrow}$ seal read with the rows already carried by the prefix envelope: the StreamName classifier rows, bundle and unary-history rows, rational classifier rows, denominator rows, dyadic source rows, shared hsame ledger, and RealLedgerPolicy boundary. Those rows have no constructor slot for regularity, completion, quotient identification, host-index equality, or an unrecorded stream row. □

Theorem 38.334 (RealUp finite-window refinement envelope nonescape)**.** *Let E be a* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over a fixed* $\mathsf{StreamName}^{\uparrow}$ *source and finite* ProbeBundle*. Every consumer of E factors only through the finite StreamName classifier rows, the* ProbeBundle*,* InBundle*, and* UnaryHistory *rows, the rational endpoint denominator rows, the* RegSeqRat *dyadic source rows, the shared* hsame *ledger, and the* RealLedgerPolicy *boundary. The envelope has no consumer-facing slot for quotient stream equality, host index equality, countable choice, selected limits, ambient completeness, or metric-completeness data.*

Proof. First read the envelope through Definition 38.332. The coverage theorem identifies each consumer-facing RegSeqRat handoff and each finite-window $\mathsf{Real}^{\uparrow}$ seal read with the StreamName classifier rows, bundle rows, rational classifier rows, endpoint denominator rows, dyadic source rows, shared hsame ledger, and RealLedgerPolicy boundary already carried by E.

Next pass the StreamName side through Definition 38.333. The prefix handoff keeps the self, symmetric, transitive, and transported windows on the same finite bundle, so the induced RegSeqRat reads remain the same pointwise packet described by Definition 126.170. The dyadic source requirements are exactly the rows supplied by Definition 38.564.

Finally use Definition 38.285. Its scoped certificate boundary says that the public $\mathsf{Real}^{\uparrow}$ prefix certificate is assembled from the recorded finite BHist rows and the ledger boundary, not from an ambient completion surface. Since the named rows are the only rows exposed by the envelope and by the StreamName/RegSeqRat handoff, there is no constructor path by which a consumer can import quotient stream equality, host index equality, countable choice, selected limits, ambient completeness, or metric-completeness data. □

Theorem 38.335 (Real stream-prefix carrier obligation)**.** *A RealUp stream-prefix candidate satisfies the carrier face only when every selected rational-observation endpoint exposes the RatUp endpoint carrier rows through the* RegSeqRat *spine. Concretely, for rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and a certified finite prefix, the candidate carrier must contain the* $\mathsf{RatHistoryCarrier}(x(i))$ *and* $\mathsf{RatHistoryCarrier}(y(i))$ *rows for each certified endpoint i named by the prefix, and those rows must be the endpoint rows read from Definition 38.195, Definition 38.214, and the prefix endpoint and readback blocks of Definition 38.111.*

Proof. Unfold Definition 38.195. Its carrier ledger is indexed by certified unary observations and supplies the rational stream-name carrier together with endpoint rational-history carrier rows.

Apply Definition 38.214 to identify those endpoint rows as the RatUp carrier face demanded by the spine. For a finite prefix, Definition 38.111 selects the endpoint classifier at the displayed prefix index, and the prefix readback blocks keep that selected endpoint attached to the same observation row.

Thus a RealUp stream-prefix carrier cannot use an external carrier datum: the visible endpoint carriers are exactly the RatHistoryCarrier rows projected from the certified unary observation spine and read back at the selected prefix endpoints. □

Theorem 38.336 (Real stream-prefix classifier obligation)**.** *The classifier face required by the RealUp certificate boundary is supplied by the pair of predicates* RealStreamClassifier *and* RealStreamPrefixClassifier*. For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, the global classifier gives every finite prefix by Definition 38.135, each finite prefix exposes its selected endpoint by Definition 38.111, and the global classifier rows are closed under the symmetry and transitivity operations of Definition 38.136 and Definition 38.137.*

Proof. Start with a global $\mathsf{RealStreamClassifier}(x, y)$ row. Apply Definition 38.135 in the forward direction to obtain $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ for every selected finite prefix n.

For a chosen prefix, apply Definition 38.111. This projects the RatUp endpoint classifier $\mathsf{RatHistoryClassifier}(x(n), y(n))$, so the finite classifier carries the same endpoint face required by the certificate boundary.

Conversely, the reverse direction of Definition 38.135 repacks the family of finite prefix classifiers into the global stream classifier. Symmetry and transitivity are then the global operations supplied by Definition 38.136 and Definition 38.137, with endpoint projection returning to the finite face whenever the boundary asks for a selected row. □

Theorem 38.337 (Real stream-prefix stability obligation)**.** *The RealUp stream-prefix stability face is closed under* hsame *transport, unary-context wrapping, and selected continuation packages. More precisely, Definition 38.107, Definition 38.108, Definition 38.139, and Definition 38.146 preserve the endpoint classifier rows used by* RealStreamPrefixClassifier *and repack them as the corresponding global* RealStreamClassifier *rows when all finite prefixes are available.*

Proof. For finite prefixes, apply Definition 38.107 to transport each selected endpoint classifier along the two pointwise hsame comparisons. Apply Definition 38.108 when the endpoints are wrapped by unary prefix and tail contexts; the induction in that theorem keeps every endpoint RatUp classifier in the prefix ledger.

For global stream rows, apply Definition 38.139. It transports the pointwise classifier family and therefore preserves the global classifier face seen by finite-prefix exactness.

For selected continuations, apply Definition 38.146. It supplies the endpoint shape, denominator, unary, and exclusion rows attached to the same selected continuation data. Repacking through Definition 38.135 identifies the finite prefix and global classifier faces, so all displayed stability operations preserve the RealUp classifier rows. □

Theorem 38.338 (Real stream-prefix ledger and seal obligation)**.** *Finite observation windows, dyadic ledger rows, and the* RegSeqRat *seal are sufficient for the visible RealUp seal obligation. The ledger is the finite policy of Definition 38.279; its classifier stability is the package of Definition 38.273; its scoped stream-name interface is Definition 38.276; and the seal is consumed by Definition 38.254 without any quotient or completion datum.*

Proof. Gather the finite observation rows named by Definition 38.279. They record the selected window indices, dyadic comparison witnesses, regularity certificates, and classifier comparison certificates used by the RealUp boundary.

Apply Definition 38.273 to keep the unary stream classifier stable across the recorded observations. Then apply Definition 38.276 to read those observations as the scoped RegSeqRat stream-name interface, including the RatUp endpoint rows exposed by the spine.

Finally, apply Definition 38.254. The theorem consumes exactly the dyadic ledger rows and the scoped stream-name interface, so the visible RealUp seal follows without quotient representatives, metric completion data, or any carrier outside the RegSeqRat spine. □

Theorem 38.339 (RealUp finite-window tail determinacy)**.** *Let W and W' be selected finite* $\mathsf{Real}^{\uparrow}$ *observation windows over the same rational-observation streams. Suppose that both windows carry the same certified unary endpoint row and both expose the common e-one tail readback used by Definition 38.314. Then the* RatHistoryCarrier *endpoint rows, unary denominator witnesses, dyadic source rows, and* RealLedgerPolicy *tail rows selected by W and W' classify to the same finite-window surface.*

Proof. Apply Definition 38.336 to the shared certified unary endpoint. The classifier obligation identifies the selected finite endpoint classifier with the RatUp endpoint rows read from the RegSeqRat spine, so W and W' begin with the same carried RatHistoryCarrier face.

Next apply Definition 38.337. The common e-one tail readback is a selected continuation package over the same unary endpoint row, and stability transports the endpoint classifier through the unary denominator, tail, and hsame rows without creating a second endpoint surface.

Apply Definition 38.338 to the finite dyadic and ledger rows exposed by the two windows. Since the seal consumes only the displayed dyadic source rows, the scoped RegSeqRat interface, and the RealLedgerPolicy tail rows, both windows feed the same visible seal data into the RealUp boundary.

Finally use Definition 38.314. Its finite-window correspondence repacks exactly those endpoint, denominator, dyadic, and tail rows as the public finite-window surface. The two selected windows therefore determine classifier-equivalent finite-window readbacks at the shared e-one tail endpoint. □

Theorem 38.340 (RealUp RegSeqRat window seal restriction)**.** *Let W be a finite* $\mathsf{Real}^{\uparrow}$ *observation window carrying the scoped* RegSeqRat *stream-name interface, the dyadic ledger rows, and the Real ledger tail rows used by the visible seal obligation. If W' is a certified sub-window of W, then W' carries the corresponding seal package by restriction, with the same* RatHistoryCarrier *endpoint rows selected from the* RegSeqRat *spine. The restricted seal exposes no quotient, completion, arithmetic closure, order closure, or metric-completion claim.*

Proof. Start with the visible seal package on W. By Definition 38.338, this package is assembled from the scoped RegSeqRat stream-name interface, the finite dyadic ledger, and the Real ledger tail rows; it does not require quotient or completion data.

Restrict the selected rows to the certified sub-window W'. Finite-window membership for W' reads each selected observation as a member of W, so the same RatHistoryCarrier endpoint rows and dyadic ledger entries are available at the smaller window.

Apply Definition 38.339 to the shared tail readback. The endpoint, denominator, dyadic, and Real ledger tail rows selected by W' classify to the same finite-window surface as their counterparts in W.

Finally repackage only those restricted rows into the seal boundary. Finite window exhaustion supplies no additional arithmetic, order, quotient, or metric-completion component, so the seal over W' is exactly the restricted RegSeqRat-to-$\mathsf{Real}^{\uparrow}$ seal package. □

Theorem 38.341 (RegSeqRat finite-window source-route uniqueness)**.** *Let W be a selected finite* $\mathsf{Real}^{\uparrow}$ *observation window carrying the scoped* RegSeqRat *stream-name interface, the displayed dyadic ledger rows, and the Real seal row. If two consumer derivations pass through the same accepted* RegSeqRat *spine inside W, then they expose the same finite* StreamName *carrier, the same* DyadicRatCore *observation rows, the same* RegSeqRat *classifier and tolerance ledger, and the same* $\mathsf{Real}^{\uparrow}$ *seal row through the displayed* hsame, Cont, *and* Pkg *provenance. No consumer derivation may bypass the* StreamName–RegSeqRat *handoff while still consuming the selected finite-window seal.*

Proof. Apply Definition 38.339 to the shared accepted RegSeqRat tail. The selected endpoint, denominator, dyadic, and Real ledger tail rows are thereby pinned to one finite-window surface.

Next apply Definition 38.340 to each consumer derivation. Since both derivations restrict to the same displayed spine, the visible seal can only read the scoped stream-name interface, the dyadic ledger, and the Real tail rows already fixed by the shared window.

It remains to compare the exposed rows. The StreamName carrier rows agree by the common spine, the DyadicRatCore observation rows agree by the finite dyadic ledger, and the classifier and tolerance entries agree by the displayed hsame and Cont readbacks. The Pkg provenance then repacks the same rows as the unique RegSeqRat-to-Real$^{\uparrow}$ source route for the selected finite seal. □

Theorem 38.342 (RealUp scoped finite-window read surface)**.** *The selected finite* Real$^{\uparrow}$ *observation windows read exactly the carrier/classifier, stability/transport, finite ledger, dyadic source,* RegSeqRat *spine,* StreamName$^{\uparrow}$*,* Rat$^{\uparrow}$*,* UnaryHistory*,* hsame*, and* RealLedgerPolicy *rows named by Definition 38.285. They read no quotient datum, completion datum, metric-completeness certificate, arithmetic or order operation, Classical datum, or standard-bridge datum.*

Lean4 ✓ `BEDC.Derived.RealUp.RealupScopedFiniteWindowReadSurface`

Proof. First collect the finite-window observation face. Apply Definition 38.314: every selected finite Real$^{\uparrow}$ window exposes the endpoint Rat$^{\uparrow}$ classifiers, unary denominator rows, common e-one tail readback, and dyadic source rows consumed by the Real$^{\uparrow}$ seal.

Next identify the structural rows read by that observation field. Apply Definition 38.335 for the DyadicRatCore, RegSeqRat, StreamName$^{\uparrow}$, and Rat$^{\uparrow}$ carrier source. Apply Definition 38.336 for the constant-history and unary stream classifier source. Apply Definition 38.337 for pointwise hsame transport and selected-continuation stability.

Finally attach the ledger and certificate boundary. Apply Definition 38.338 for the finite RealLedgerPolicy rows and seal consumption. Then apply Definition 38.285, which states that these rows are exactly the scoped Real$^{\uparrow}$ certificate boundary and excludes quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data. □

Theorem 38.343 (RealUp common finite-window pullback transitive restriction)**.** *Let W_0 and W_1 be certified finite* Real$^{\uparrow}$ *observation-window packages, let U be a certified common* StreamName$^{\uparrow}$ *subwindow of their displayed* ProbeBundle *windows, and let V be a certified* StreamName$^{\uparrow}$ *subwindow of U. Assume that the restrictions of W_0 and W_1 to U carry matching* RatStreamNameFiniteWindowClassifier *rows, matching* RegSeqRat *spine rows, matching endpoint* DyadicRatCoreObservationRow *witnesses, matching displayed* hsame *transports, matching* Cont *rows, matching* Pkg *provenance, and matching finite* RealLedgerPolicy *rows. Then the two routes to V expose the same finite* BHist *packet and the same finite* RealLedgerPolicy *row: restrict the direct common-window pullback from U to V, or first pull both* Real$^{\uparrow}$ *packages back to U and then restrict that common packet to V.*

This is a finite-window descent theorem only. It exposes no quotient, completion, metric-completeness datum, global Cauchy modulus, arithmetic operation, order operation, Classical representative, or standard bridge row.

Proof. First apply Definition 38.355 on the common subwindow U. The hypotheses name the same StreamName classifier, regular sequence spine, dyadic endpoint observations, endpoint transports, continuation rows, package provenance, and finite real-ledger row, so the pullback over U is a single finite BHist packet with a single RealLedgerPolicy row.

Now compare the two restrictions from U to V. Apply Definition 38.327 to the certified chain $W_i \supseteq U \supseteq V$ for each selected observation window W_i. It identifies the direct restriction of the displayed W_i package to V with the two-step restriction through U, with the same finite packet, classifier rows, endpoint rows, transport ledger, provenance, and real-ledger boundary.

Finally use Definition 38.354 on the endpoint window V. The StreamName-RegSeqRat handoff and the dyadic endpoint rows are already matched after the two-step coherence comparison, so the endpoint readback rows on V and the finite RealLedgerPolicy row are the same for both routes. □

Theorem 38.344 (RealUp StreamName-RegSeqRat finite-window correspondence)**.** *A selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window surface is equivalent, as BEDC public data, to the pair consisting of the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface of Definition 126.301 and the* RegSeqRat *dyadic source exhaustion rows consumed in Definition 38.314. The pair records exactly the finite-window stream classifier, endpoint* $\mathsf{Rat}^{\uparrow}$ *classifiers, unary denominator witnesses, common e-one tail readback, DyadicRatCore source rows,* UnaryHistory *endpoints,* hsame *transport, and finite* RealLedgerPolicy *ledger rows, and it records no quotient, completion, arithmetic, order, metric-completeness, Classical, or standard-bridge datum.*

Proof. For the projection direction, first apply Definition 38.342. It identifies the selected finite $\mathsf{Real}^{\uparrow}$ read surface with the DyadicRatCore, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, unary-history, hsame, and finite ledger rows that the scoped certificate exposes.

Then apply Definition 38.345. The $\mathsf{StreamName}^{\uparrow}$ component factors through the public finite-window interface of Definition 126.301, and the remaining rows are exactly the dyadic source, regular sequence, endpoint, unary, transport, and ledger rows read by Definition 38.314.

For the reconstruction direction, start with the displayed pair. Repack the $\mathsf{StreamName}^{\uparrow}$ finite-window classifier rows through the public interface, and repack the endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, unary denominator witnesses, common e-one tail readback, and DyadicRatCore source rows through the exhaustion obligation.

The two packets share the same finite UnaryHistory endpoints, hsame transport, and RealLedgerPolicy readback. Combining those shared rows gives precisely the scoped boundary named in Definition 38.342, so the reconstructed surface is the original selected finite $\mathsf{Real}^{\uparrow}$ observation-window surface and contains no excluded datum. □

Theorem 38.345 (RealUp finite-window export through StreamNameUp interface)**.** *Every selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window read surface of Definition 38.342 factors its stream-name component through the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface of Definition 126.301. Its stream-name component is exactly the* RatStreamNameFiniteWindowClassifier *surface with classifier-to-window coverage, singleton-window exhaustion, reindexing and point-map coverage, structural laws, the selected endpoint package, and finite-window stability fields.*

After this factorization, the remaining $\mathsf{Real}^{\uparrow}$ *rows are only the DyadicRatCore observation rows,* RegSeqRat *spine rows,* $\mathsf{Rat}^{\uparrow}$ *endpoint rows,* UnaryHistory*,* hsame *transport, and* RealLedgerPolicy*. The exported finite-window surface contains no quotient datum, completion datum, arithmetic or order operation, metric-completeness certificate, Classical datum, or standard bridge.*

Proof. First identify the public stream-name finite-window surface. Apply Definition 126.301: the $\mathsf{StreamName}^{\uparrow}$ component available to downstream $\mathsf{Real}^{\uparrow}$ windows is the RatStreamNameFiniteWindowClassifier surface together with classifier-to-window coverage, singleton-window exhaustion, reindexing and point-map coverage, structural laws, the selected endpoint package, and finite-window stability fields.

Next identify the selected $\mathsf{Real}^{\uparrow}$ read surface. Apply Definition 38.342; it says that selected finite observation windows read the DyadicRatCore source, RegSeqRat spine, $\mathsf{StreamName}^{\uparrow}$ rows, $\mathsf{Rat}^{\uparrow}$ endpoints, UnaryHistory, hsame transport, and finite RealLedgerPolicy rows, while excluding quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data.

The ten-field fit is: traditional target, finite $\mathsf{Real}^{\uparrow}$ observation export; BEDC source, DyadicRatCore plus $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Rat}^{\uparrow}$ over BHist; carrier, selected rational-observation windows; classifier, finite-window pointwise $\mathsf{Rat}^{\uparrow}$ classification; stability, $\mathsf{StreamName}^{\uparrow}$ finite-window stability plus hsame transport; ledger, finite observation index plus RealLedgerPolicy; core the-

orem, this block; exactness, factorization through the public finite-window interface; standard bridge, none; status, proof target.

Finally close directly at the finite-window surface. The $\mathsf{StreamName}^{\uparrow}$ component is exactly the public finite-window interface just identified, and Definition 38.342 supplies the whole local read surface in which that component sits. Removing the $\mathsf{StreamName}^{\uparrow}$ component therefore leaves precisely the DyadicRatCore observation rows, RegSeqRat spine rows, $\mathsf{Rat}^{\uparrow}$ endpoint rows, UnaryHistory, hsame transport, and finite RealLedgerPolicy rows, with the excluded quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data still absent. □

Theorem 38.346 (RealUp StreamName window stability consumption)**.** *Let s, t, s', t' be rational observation maps carried by a finite* ProbeBundle *window, and let the selected unary observation rows be the endpoint rows read by the finite* $\mathsf{Real}^{\uparrow}$ *prefix surface. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier accepts the window for (s, t), suppose the packet carries bundled pointwise* hsame *transports from (s, t) to (s', t'), and suppose the selected transported endpoint package of Definition 126.32 is present.*

Then the transported selected $\mathsf{Real}^{\uparrow}$ *prefix read surface has the same finite* RealLedgerPolicy *boundary as in Definition 38.342. It factors through Definition 38.345 as a finite* BHist *read surface, with no quotient datum, completion datum, metric-completeness certificate, Classical datum, or standard-bridge datum.*

Proof. First apply Definition 126.28 to the accepted $\mathsf{StreamName}^{\uparrow}$ classifier and the bundled pointwise hsame transports. This produces the transported finite-window classifier for (s', t') over the same finite ProbeBundle boundary.

Next apply Definition 126.32 to the selected endpoint rows. The transported package supplies the positive denominator rows, unary shape rows, zero-head exclusion rows, and one-tail agreement rows needed by the selected endpoint surface.

Finally repack these transported rows through Definition 38.345. The scoped read surface of Definition 38.342 then supplies the unchanged finite RealLedgerPolicy boundary, so the transported prefix surface is still a finite BHist interface and none of the excluded quotient, completion, metric, Classical, or bridge data is introduced. □

Theorem 38.347 (RealUp StreamName source-exhaustion equivalence)**.** *For any selected finite* $\mathsf{Real}^{\uparrow}$ *observation window, the selected read surface has exactly the* $\mathsf{StreamName}^{\uparrow}$ *component supplied by Definition 126.301 together with the RegSeqRat and DyadicRatCore finite-window source rows consumed by Definition 38.314. Conversely, the concrete RegSeqRat and DyadicRatCore rows, together with the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package, repack into the* $\mathsf{StreamName}^{\uparrow}$ *component of Definition 38.342.*

This equivalence is internal to the selected finite BHist *window. It does not read a quotient datum, completion datum, metric-completeness certificate, arithmetic or order operation, Classical datum, or standard-bridge datum.*

Proof. Project the selected $\mathsf{Real}^{\uparrow}$ read surface using Definition 38.342. This gives the visible $\mathsf{StreamName}^{\uparrow}$ rows, the RegSeqRat spine rows, the DyadicRatCore source rows, the $\mathsf{Rat}^{\uparrow}$ endpoint rows, UnaryHistory, hsame transport, and the finite RealLedgerPolicy rows, with the excluded data absent.

Next apply Definition 38.345 to the projected stream-name component. It identifies that component with the public RatStreamNameFiniteWindowClassifier package of Definition 126.301.

The remaining finite source rows are read from Definition 38.314: DyadicRatCore supplies the selected dyadic observation rows, and RegSeqRat supplies the spine rows used by the finite window. These rows are concrete BHist entries rather than quotient or completion witnesses.

For the converse direction, start with the public $\mathsf{StreamName}^{\uparrow}$ finite-window package and the concrete RegSeqRat and DyadicRatCore rows just described. The public package reconstructs the same selected stream-name component, while the source rows provide exactly the surrounding finite-window inputs required by Definition 38.342. Repacking these rows therefore recovers the selected $\mathsf{StreamName}^{\uparrow}$ component inside the same $\mathsf{Real}^{\uparrow}$ read surface, and no excluded datum is introduced. □

Theorem 38.348 (RealUp finite-window constant-stream seal factorization)**.** *Let d, e : Hist and let W be a certified finite* ProbeBundle *window. Consider selected finite* $\mathsf{Real}^{\uparrow}$ *observation rows whose* $\mathsf{StreamName}^{\uparrow}$ *component is generated by the constant rational streams* $\mathsf{RatConstStream}_d$ *and* $\mathsf{RatConstStream}_e$*. A displayed* $\mathsf{RatHistoryClassifier}(d, e)$ *together with the constant stream-name package factors the selected read surface through Definition 126.301. After this factorization the remaining visible rows are exactly the DyadicRatCore observation rows, RegSeqRat spine rows,* $\mathsf{Rat}^{\uparrow}$ *endpoint rows,* UnaryHistory *rows,* hsame *transport rows, and finite* RealLedgerPolicy *rows recorded by the selected* $\mathsf{Real}^{\uparrow}$ *window.*

Conversely, those DyadicRatCore, RegSeqRat, $\mathsf{Rat}^{\uparrow}$*, unary-history,* hsame*, and finite* RealLedgerPolicy *rows, together with the constant stream-name package for* $\mathsf{RatHistoryClassifier}(d, e)$*, repack into the selected finite* $\mathsf{Real}^{\uparrow}$ *read surface. The factorization contains no quotient row, completion row, metric-completeness certificate, Classical datum, or standard-bridge datum.*

Proof. First apply Definition 126.61. It converts the displayed $\mathsf{RatHistoryClassifier}(d, e)$ into the constant stream-name package for $\mathsf{RatConstStream}_d$ and $\mathsf{RatConstStream}_e$ without replacing the histories by a quotient or standard model.

Next restrict this package to the finite window W by Definition 126.56. The resulting finite constant-stream rows are exactly the rows accepted by the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface of Definition 126.301.

Apply Definition 38.345 to the selected $\mathsf{Real}^{\uparrow}$ observation window. Its stream-name component therefore factors through the public interface, while Definition 38.347 identifies the unconsumed finite source rows as precisely the DyadicRatCore observations, RegSeqRat spine, $\mathsf{Rat}^{\uparrow}$ endpoints, unary-history rows, hsame transports, and finite RealLedgerPolicy rows.

For the converse direction, start with this same finite list of remaining rows and the constant stream-name package. The public interface reconstructs the selected $\mathsf{StreamName}^{\uparrow}$ component, and the source-exhaustion equivalence supplies exactly the surrounding RegSeqRat, DyadicRatCore, $\mathsf{Rat}^{\uparrow}$, unary-history, transport, and ledger rows needed by the scoped $\mathsf{Real}^{\uparrow}$ read surface. Repacking these displayed rows recovers the selected finite surface, again without quotient, completion, metric-completeness, Classical, or standard-bridge data. □

Theorem 38.349 (StreamName Real seal factorization diamond)**.** *Let $W \subseteq W'$ be finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation windows with displayed prefix refinement. Suppose the windows carry accepted* RegSeqRat *spine rows, a shared DyadicRatCore endpoint ledger,* ProbeBundle *and* InBundle *rows,* UnaryHistory *rows, a finite* Cont *ledger, displayed* hsame *endpoint transports, and* Pkg *provenance. Then restricting the W'* $\mathsf{Real}^{\uparrow}$ *seal factorization to W and factoring the W seal read through the StreamName–*RegSeqRat *source produce the same visible* BHist *source tuple.*

That tuple is exactly the StreamName finite-window row, accepted RegSeqRat *seal row, dyadic endpoint rows, bundle rows, unary-history rows, finite continuation rows, displayed endpoint trans-*

ports, and provenance rows. It contains no completion row, quotient representative, host index equality, Classical choice row, metric-completeness datum, or standard-bridge datum.

Proof. Apply Definition 126.143 to the displayed inclusion $W \subseteq W'$. It identifies the restricted larger factorization with the direct smaller-window factorization over the same bundle, history, transport, continuation, and provenance rows.

Next use Definition 126.149. The two routes have the same public StreamName row, accepted RegSeqRat spine, dyadic endpoint ledger, bundle membership, unary observations, Cont ledger, hsame endpoint transports, and Pkg provenance, so their visible seal source is fixed to one BHist tuple.

Finally, Definition 126.150 rules out any additional source row beyond that tuple, and Definition 38.348 repacks it as the selected finite $\mathsf{Real}^{\uparrow}$ seal surface, without adding completion, quotient, host-equality, Classical, metric-completeness, or standard-bridge data. □

Theorem 38.350 (RealUp finite-window StreamName readback package)**.** *Let W be a certified finite* ProbeBundle *window and let a public* $\mathsf{Real}^{\uparrow}$ *finite-window classifier read be accepted over a* $\mathsf{StreamName}^{\uparrow}$ *source. Then there is a displayed* BHist *packet whose rows are exactly the unary schedule for W, the* $\mathsf{Rat}^{\uparrow}$ *observation endpoints visible in the window, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier row, the* $\mathsf{Real}^{\uparrow}$ *scoped window row, the* Cont *and* hsame *transport ledger connecting the stream read to the real read, and the* Pkg *provenance of the selected source rows. Every public* $\mathsf{Real}^{\uparrow}$ *finite-window classifier read through this* $\mathsf{StreamName}^{\uparrow}$ *source factors through that packet.*

The packet contains no quotient row, completion row, metric-completeness certificate, global Cauchy modulus, arithmetic or order operation, Classical datum, or standard bridge. Its endpoint reads are exactly the displayed $\mathsf{Rat}^{\uparrow}$ *observations and its classifier reads are exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *and scoped* $\mathsf{Real}^{\uparrow}$ *finite-window rows.*

Proof. Begin with the accepted finite-window $\mathsf{Real}^{\uparrow}$ read and project its $\mathsf{StreamName}^{\uparrow}$ component. By Definition 38.345, this component factors through the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface, so the stream-name classifier row is displayed as a finite BHist row rather than as an ambient stream equality.

Next use Definition 38.346. It transports the projected stream-name row across the selected finite window and records the required Cont and hsame ledger entries. These entries are the only transport rows consumed by the scoped $\mathsf{Real}^{\uparrow}$ classifier read.

For constant rational endpoints, apply Definition 38.348. It identifies the endpoint part of the read with the displayed $\mathsf{Rat}^{\uparrow}$ observation rows, the unary schedule, and the surrounding finite RealLedgerPolicy rows. For non-constant accepted stream-name sources, the same packet shape is obtained by reading the $\mathsf{Rat}^{\uparrow}$ endpoints already exposed by the finite StreamName window.

Finally apply Definition 38.399. Any two displayed packets supporting the same public $\mathsf{Real}^{\uparrow}$ classifier read admit the same common finite refinement, hence determine the same visible rows up to the recorded hsame transports. This gives the announced readback packet and shows that every public finite-window classifier read factors through it. □

Theorem 38.351 (RealUp empty-window StreamName boundary)**.** *Let W be a selected finite* $\mathsf{Real}^{\uparrow}$ *observation window whose* ProbeBundle *boundary is empty. Then the* $\mathsf{StreamName}^{\uparrow}$ *component of the* $\mathsf{Real}^{\uparrow}$ *finite-window readback package is exactly the public empty-window vacuity row of Definition 126.310. After this identification, the remaining visible* $\mathsf{Real}^{\uparrow}$ *rows are only the scoped ledger, dyadic source,* UnaryHistory, hsame, *and* RealLedgerPolicy *rows already listed by Definition 38.342.*

The empty-window read contains no quotient row, completion row, metric-completeness certificate, Classical datum, arithmetic or order row, or standard-bridge datum.

Proof. Project the selected finite readback through Definition 38.350. The projected $\mathsf{StreamName}^{\uparrow}$ row is a displayed finite-window classifier row inside the same BHist packet as the unary schedule, endpoint rows, Cont and hsame transport ledger, and Pkg provenance.

Apply Definition 38.345 to factor that stream-name row through the public finite-window interface. Since the ProbeBundle boundary of W is empty, the corresponding public stream-name row is the empty-window vacuity row of Definition 126.310.

It remains to identify what the $\mathsf{Real}^{\uparrow}$ packet still exposes. The scoped read surface theorem Definition 38.342 lists exactly the surviving finite rows: the scoped ledger, dyadic source, UnaryHistory, hsame, and RealLedgerPolicy entries. The same theorem excludes quotient, completion, metric-completeness, Classical, arithmetic, order, and standard-bridge data, and the empty-window projection does not add any of them. □

Theorem 38.352 (RealUp StreamName-RegSeqRat tail-selector determinacy)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. Assume that they display the same public* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier row, the same* RegSeqRat *spine rows, the same DyadicRatCore observation rows, matching* $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers, and the same finite* Cont/hsame *ledger. Then every selected tail-classifier readback extracted from W_0 and every selected tail-classifier readback extracted from W_1 represent the same finite* $\mathsf{Real}^{\uparrow}$ *tail read up to the displayed* hsame *transport. Moreover the two packages expose the same* RealLedgerPolicy *row.*

The conclusion is scoped to the selected finite-window surface. It does not assert equality of quotient completions, metric completions, arithmetic operations, order operations, Classical representatives, or standard bridge data.

Proof. Project both selected packages through Definition 38.350. Each projection has a $\mathsf{StreamName}^{\uparrow}$ component, RegSeqRat spine rows, DyadicRatCore observation rows, $\mathsf{Rat}^{\uparrow}$ endpoint classifiers, a finite Cont/hsame ledger, and a surrounding RealLedgerPolicy row.

Identify the shared $\mathsf{StreamName}^{\uparrow}$ component by Definition 38.347. Under the hypotheses, the two finite-window classifier rows consume the same concrete stream-name source and the same regular-sequence and dyadic observation rows.

The constant and empty boundary theorems remove non-displayed choices. Apply Definition 38.348 to the constant endpoint case and Definition 38.351 to the empty-window case. In both cases the visible tail read is forced by the selected endpoint, stream-name, regular-sequence, dyadic, and ledger rows already present in the projected package.

It remains to compare the two selected tail reads. Compose the displayed hsame transports in the common finite ledger. Since the two packages have the same Cont/hsame ledger and the same RealLedgerPolicy boundary, the transported tail classifiers coincide up to hsame, and the exposed RealLedgerPolicy rows are identical as public finite-window rows. □

Theorem 38.353 (RealUp finite-window StreamName readback uniformity)**.** *Let W be a certified finite* ProbeBundle *window supporting a public* $\mathsf{Real}^{\uparrow}$ *finite-window readback through a* $\mathsf{StreamName}^{\uparrow}$ *source, and let U be either a certified finite subwindow of W or a certified finite common refinement of two such windows. Assume that the restriction to U carries the same public* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* RegSeqRat *spine rows, the same DyadicRatCore observation rows, and the restricted* Cont/hsame *ledger displayed over U. Then the* $\mathsf{Real}^{\uparrow}$ *readback packet over W restricts over U to the same visible* $\mathsf{Rat}^{\uparrow}$ *endpoint rows and to the same finite* RealLedgerPolicy *row.*

Equivalently, certified passage to a displayed finite subwindow or common finite refinement preserves the public endpoint readback and the real-ledger row determined by the StreamName-RegSeqRat-DyadicRatCore surface. The assertion does not identify quotient completions, metric completions, global Cauchy moduli, arithmetic operations, order operations, Classical representatives, or standard bridge data.

Proof. Start with the selected $\mathsf{Real}^{\uparrow}$ read over W and apply Definition 38.350. This displays a single finite BHist packet containing the public $\mathsf{StreamName}^{\uparrow}$ finite-window row, the RegSeqRat spine, the DyadicRatCore observations, the visible $\mathsf{Rat}^{\uparrow}$ endpoints, the restricted Cont/hsame ledger, the Pkg provenance, and the finite RealLedgerPolicy row.

Restrict this displayed packet along the certified subwindow U, or along the certified common refinement when two windows are present. The common-refinement determinacy theorem Definition 38.399 says that any two displayed packets with the same public rows over the refinement determine the same finite surface up to the recorded hsame transports.

The source rows remaining after restriction are exactly the public stream-name row, the regular-sequence spine, and the dyadic observation rows named in the hypotheses. This is the source-exhaustion equivalence of Definition 38.347, while Definition 126.301 identifies the restricted stream-name row as the same public finite-window interface row rather than an ambient stream equality.

With the stream-name source and the restricted transport ledger fixed, the readback package has no remaining displayed choice for the $\mathsf{Rat}^{\uparrow}$ endpoints or for the finite RealLedgerPolicy row. Hence the packet over W restricts to the same visible endpoint rows and the same real-ledger row over U. □

Theorem 38.354 (RealUp StreamName-RegSeqRat uniform window readback)**.** *Let W_0 and W_1 be certified finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages, and let U be a certified finite common refinement of their selected* ProbeBundle *windows. Assume that both packages factor through the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface, that their restrictions to U carry the same* $\mathsf{StreamName}^{\uparrow}$ *classifier row, and that the displayed* RegSeqRat *pointwise seal gives the same regular-sequence spine over the same* BHist *endpoints,* hsame *transports,* Cont *rows, and* Pkg *provenance rows. Then the two restricted* $\mathsf{Real}^{\uparrow}$ *readback packets have the same visible* $\mathsf{Rat}^{\uparrow}$ *endpoint readback rows on every selected finite subwindow of U, and they expose the same finite* RealLedgerPolicy *row.*

The statement is only a finite-window consumer theorem. It identifies the rows read through the StreamName-RegSeqRat handoff and does not identify quotient completions, metric completions, global Cauchy moduli, arithmetic operations, order operations, Classical representatives, or standard bridge data.

Proof. First apply Definition 126.111. The common $\mathsf{StreamName}^{\uparrow}$ classifier row over U and the displayed RegSeqRat pointwise seal determine a single finite-window handoff surface, with the BHist endpoints, hsame transports, Cont rows, and Pkg provenance rows kept in the public packet.

Next compare that handoff surface with the selected $\mathsf{Real}^{\uparrow}$ read surface. Apply Definition 38.344. It identifies the $\mathsf{Real}^{\uparrow}$ finite-window observation surface with the pair consisting of the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface and the RegSeqRat dyadic source exhaustion rows, together with the same endpoint, transport, provenance, and finite-ledger rows.

Finally restrict along each selected finite subwindow of U. The hypotheses now match the input of Definition 38.353: the public stream-name row, regular-sequence spine, dyadic observation rows, and restricted Cont/hsame ledger are the same. Therefore the two consumer packets read back the same visible $\mathsf{Rat}^{\uparrow}$ endpoint rows on each selected finite window and expose the same finite RealLedgerPolicy row. □

Theorem 38.355 (RealUp common finite-window pullback)**.** *Let W_0 and W_1 be certified finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages, and let U be a certified common* $\mathsf{StreamName}^{\uparrow}$ *subwindow of their displayed* ProbeBundle *windows. Assume that the restrictions to U carry the same* RatStreamNameFiniteWindowClassifier *rows, the same* RegSeqRat *spine rows, and endpoint* DyadicRatCoreObservatio *witnesses matched by the displayed* hsame *transports. Assume also that the two selected* $\mathsf{Real}^{\uparrow}$ *read surfaces expose their endpoint rows through the same restricted* BHist *packet and the same finite* Cont/Pkg *provenance ledger over U. Then the restricted* $\mathsf{Real}^{\uparrow}$ *readback packets over U are the same finite* BHist *packet and expose the same finite* RealLedgerPolicy *row.*

The theorem identifies only the finite StreamNameUp-RegSeqRat-DyadicRatCore window shared by the two read surfaces. It does not export quotient completions, metric completions, global Cauchy moduli, arithmetic operations, order operations, Classical representatives, or standard bridge data.

Proof. Restrict both certified observation windows to the common subwindow U. The restricted surfaces still carry exactly the displayed RatStreamNameFiniteWindowClassifier rows, RegSeqRat spine rows, DyadicRatCore endpoint witnesses, hsame transports, Cont rows, and Pkg provenance named in the hypotheses.

Apply the finite-window source-exhaustion obligation of Definition 38.347 to the endpoint rows. It says that, over the selected finite packet, no endpoint readback row remains outside the StreamNameUp-RegSeqRat-DyadicRatCore source family.

Next apply Definition 38.354. The matching stream-name classifier rows and regular-sequence spine rows identify the StreamName-RegSeqRat handoff over U, while the dyadic endpoint witnesses and the displayed hsame transports fix the endpoint side of the handoff.

Finally use the ledger exactness of the selected $\mathsf{Real}^{\uparrow}$ read surface, as in Definition 38.350. The only visible finite packet left over U is the shared BHist packet, and the only finite ledger row exposed by that packet is the same RealLedgerPolicy row. □

Theorem 38.356 (RealUp StreamName RegSeqRat cover seal associativity)**.** *Let K_0, K_1, and K_2 be certified finite* $\mathsf{Real}^{\uparrow}$ *observation windows with certified* $\mathsf{StreamName}^{\uparrow}$ *finite-window restrictions $K_0 \leftarrow K_1 \leftarrow K_2$. Assume that the displayed* RatStreamNameFiniteWindowClassifier *rows, the induced* RegSeqRat *seal rows, and the finite* Cont/hsame/Pkg *ledgers are composable over these two restrictions. Then the* $\mathsf{Real}^{\uparrow}$ *finite-window seal obtained by first pulling the displayed* BHist *seal surface back from K_2 to K_1 and then from K_1 to K_0 is the same displayed* BHist *seal surface as the direct pullback from K_2 to K_0.*

The statement uses only the displayed $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and finite-window ledger rows. It includes no completion, quotient, metric-completeness, Classical, arithmetic, order, or standard bridge datum.*

Proof. Apply Definition 38.355 to the restriction $K_2 \to K_1$ and then to the restriction $K_1 \to K_0$. The two applications identify the intermediate seal with displayed $\mathsf{StreamName}^{\uparrow}$ classifier rows, RegSeqRat spine rows, endpoint witnesses, and the finite Cont/hsame/Pkg ledger on K_1 and K_0.

Apply the same common-window pullback theorem directly to the composed restriction $K_2 \to K_0$. Its input rows are exactly the rows obtained by composing the two certified restrictions: the same StreamName finite-window classifier, the same induced RegSeqRat seal rows, and the same finite transport ledger displayed over K_0.

The common-refinement determinacy theorem Definition 38.399 compares the two displayed packets over the common finite window K_0. Since their StreamName, RegSeqRat, endpoint, transport, and provenance rows agree, the visible $\mathsf{Real}^{\uparrow}$ seal surface is unique up to the recorded hsame transports.

Finally use the tail-reindexing stability of Definition 126.182. It says that the tail rows introduced by the nested restriction are the same pointwise rows as those introduced by the composed restriction. Thus the two pullback routes display the same finite BHist seal surface and the same finite ledger rows. □

Theorem 38.357 (RealUp StreamName RegSeqRat cover identity seal)**.** *Let K be a certified finite* $\mathsf{Real}^{\uparrow}$ *observation window carrying the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window,* RegSeqRat*, and DyadicRatCore seal rows exposed by Definition 38.350. Restrict this displayed seal along the identity certified subwindow $K \leftarrow K$. Then the restricted surface is the same finite* BHist *seal surface, with the same* hsame/Cont/Pkg *provenance ledger and the same finite* RealLedgerPolicy *row.*

The identity restriction exports no quotient, completion, metric-completeness, Classical representative, arithmetic operation, order operation, or standard bridge datum. It only records that the finite StreamNameUp-RegSeqRat-DyadicRatCore read surface is neutral under the certified identity subwindow.

Proof. Start with the finite readback package of Definition 38.350. It displays the $\mathsf{StreamName}^{\uparrow}$ finite-window classifier, the RegSeqRat spine rows, the DyadicRatCore endpoint rows, and the finite hsame/Cont/Pkg ledger on K.

Apply Definition 38.355 to the identity certified subwindow $K \leftarrow K$. The source and target windows are the same certified window, so the pullback reads exactly the same classifier rows, endpoint witnesses, provenance rows, and RealLedgerPolicy entry.

Finally view the identity restriction as the neutral member of the composition law in Definition 38.356. A pullback followed by the identity subwindow, or preceded by it, has the same direct route. Thus the displayed BHist seal surface and its finite ledger are unchanged. □

Theorem 38.358 (RealUp window refinement cover pullback exhaustion)**.** *Let W be a certified finite* $\mathsf{Real}^{\uparrow}$ *observation window. Let $\mathcal{C}_0$ and $\mathcal{C}_1$ be two finite* $\mathsf{StreamName}^{\uparrow}$ *cover presentations of the displayed window of W, and let $\mathcal{R}$ be a certified finite common refinement whose restrictions to the two covers carry the same* RatStreamNameFiniteWindowClassifier *rows, the same* RegSeqRat *spine rows, the same DyadicRatCore observation witnesses, and the same displayed* hsame *transports. Assume that every cover member exposes its* $\mathsf{Rat}^{\uparrow}$ *endpoint rows through the displayed finite* BHist *packet and through the same restricted* Cont/Pkg *provenance ledger. Then the* $\mathsf{Real}^{\uparrow}$ *read surface assembled from $\mathcal{C}_0$, the* $\mathsf{Real}^{\uparrow}$ *read surface assembled from $\mathcal{C}_1$, and the direct read surface of W expose exactly the same finite* BHist *packet.*

The common packet consists of the $\mathsf{StreamName}^{\uparrow}$ *classifier rows, the* RegSeqRat *spine, the DyadicRatCore observation rows, the visible* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, the displayed* hsame/Cont/Pkg *provenance rows, and the finite* RealLedgerPolicy *row. The result does not export quotient completions, metric completions, global Cauchy moduli, arithmetic operations, order operations, Classical representatives, or standard bridge data.*

Proof. Pull both cover presentations back to the certified common refinement $\mathcal{R}$. By Definition 38.355, the restrictions of the two cover-assembled surfaces to each member of $\mathcal{R}$ display the same classifier rows, RegSeqRat spine rows, DyadicRatCore observation witnesses, $\mathsf{Rat}^{\uparrow}$ endpoint rows, and hsame/Cont/Pkg provenance rows.

Assemble those refined packets back along each cover. The cover-seal associativity theorem Definition 38.356 says that refining through $\mathcal{R}$ and then returning along $\mathcal{C}_i$ gives the same displayed finite BHist surface as the direct pullback along that cover. Thus the two assembled cover surfaces expose the same refined packet.

Compare each assembled cover with the direct read surface of W. The identity seal theorem Definition 38.357 identifies the direct surface with the neutral cover presentation on the same certified finite window, so associativity transports the refined packet from either cover route to the direct route.

It remains only to check that no visible row is left outside this packet. Apply the source-exhaustion equivalence Definition 38.347 to the StreamNameUp-RegSeqRat-DyadicRatCore source rows, and then apply the finite-window ledger boundary recorded in Definition 38.350. The endpoint, transport, provenance, and RealLedgerPolicy rows are therefore exactly the shared finite BHist packet named in the statement. □

Theorem 38.359 (RealUp StreamName-RegSeqRat directed-window consumer surface)**.** *Let $\mathcal{D}$ be a finite directed system of certified* $\mathsf{Real}^{\uparrow}$ *observation windows. Assume that every transition map in $\mathcal{D}$ is the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window restriction, and that every endpoint row agrees under the* RegSeqRat *finite-window seal. Then the whole directed system exposes one unique consumer-visible* BHist *packet.*

That packet consists exactly of the $\mathsf{StreamName}^{\uparrow}$ *classifier rows, the* RegSeqRat *spine rows, the DyadicRatCore observation rows, the visible* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, the displayed* hsame/Cont/Pkg *provenance rows, and the finite* RealLedgerPolicy *row. It exports no quotient completion, metric completeness, global Cauchy modulus, order operation, or standard bridge datum.*

Proof. First choose any finite cofinal window of the directed system. Apply Definition 126.111 to place the endpoint rows under the common RegSeqRat finite-window seal, and apply Definition 126.165 to transport the same rows along the displayed $\mathsf{StreamName}^{\uparrow}$ restrictions.

Next compare any two routes through the chosen directed window. The cover-seal associativity theorem Definition 38.356 identifies the iterated restriction with the direct restriction, so the classifier, spine, endpoint, and provenance rows do not depend on the route.

Finally apply Definition 38.358 to the finite family of windows generated by the chosen routes. It says that every cover presentation, every common refinement, and the direct read surface expose the same finite BHist packet. The source list in that theorem is exactly the consumer surface named here, so no quotient completion, metric completeness, global Cauchy modulus, order operation, or standard bridge datum is added. □

Theorem 38.360 (RegSeqRat tail-classifier exactness)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. Assume that they have matching* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, matching* RegSeqRat *spine rows, matching DyadicRatCore observation rows, and the same finite* Cont/hsame *ledger over the visible endpoint and transport entries. Then the selected* RegSeqRat *tail classifier extracted from W_0 is exactly the same selected* RegSeqRat *tail classifier extracted from W_1 as a finite-window readback row, after the displayed* hsame *transports in the common ledger are applied.*

The exact classifier surface is only the finite $\mathsf{StreamName}^{\uparrow}$ *row, the* RegSeqRat *spine, the DyadicRatCore observation rows, the endpoint transport ledger, and the shared* RealLedgerPolicy *provenance. It exposes no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order operation, standard-bridge row, or external regular-sequence selector.*

Proof. Project both selected windows through the finite-window readback package of Definition 38.350. This places the $\mathsf{StreamName}^{\uparrow}$ classifier row, the RegSeqRat spine rows, the DyadicRatCore observation rows, the endpoint classifiers, the finite Cont/hsame ledger, and the RealLedgerPolicy row in one visible finite package.

Apply Definition 38.352. The matching stream-name, regular-sequence, dyadic, endpoint, and ledger rows force the two selected tail readbacks to agree up to the displayed hsame transports, and the same argument identifies the exposed RealLedgerPolicy row.

It remains to record that this agreement is exact on the selected RegSeqRat classifier surface. Use Definition 38.342 together with the finite-window readback package to list every row visible to a consumer. The list is precisely the finite stream-name row, the regular-sequence spine, the dyadic observation rows, the endpoint transport ledger, and the shared provenance; the same scoped surface excludes quotient, completion, Classical, metric-completeness, arithmetic, order, standard-bridge, and external selector data. Hence the selected tail classifier is determined exactly by the common finite rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratTailClassifierExactness`

Theorem 38.361 (RegSeqRat tail-classifier atlas refinement invariance)**.** *Let $\mathcal{A} = \{W_i\}_{i\in I}$ be a finite atlas of selected finite* Real$^{\uparrow}$ *observation-window packages, and let $\mathcal{B} = \{V_a\}_{a\in A}$ be a finite atlas refining $\mathcal{A}$. Suppose each refined chart V_a is certified inside some chart $W_{i(a)}$, and suppose the refinement maps, refined overlaps, and comparison pullbacks preserve the displayed* StreamName$^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows.*

Then the selected RegSeqRat *tail-classifier row descended from $\mathcal{B}$ factors through the selected row descended from $\mathcal{A}$. On the finite union covered by the two atlases these two rows agree after readback to the displayed StreamName and RegSeqRat finite window surface. The comparison uses no quotient, completion, Classical representative, metric-completeness row, arithmetic or order row, standard-bridge row, or hidden selector row.*

Proof. Apply Definition 38.375 to $\mathcal{A}$ and to $\mathcal{B}$. This gives one selected atlas row on each finite union, with chartwise pullback uniqueness for every selected chart.

Next use Definition 38.373 on the certified inclusions $V_a \hookrightarrow W_{i(a)}$. The preservation hypotheses identify each refined selected row with the pullback of the coarse chart row along the displayed finite-window refinement map.

Now group the refined charts by their coarse chart and apply Definition 38.374. The descent obtained by first gluing the refined charts inside each W_i and then gluing the coarse atlas is the same row as the descent obtained by flattening all refined charts and gluing $\mathcal{B}$ directly.

The partition comparison is supplied by Definition 38.367: both atlas rows read the same displayed StreamName, RegSeqRat, DyadicRatCore, unary-history, probe, membership, transport, continuation, package, and policy rows on the common finite union.

It remains to identify the mediator. Both the factored row from $\mathcal{A}$ and the row descended from $\mathcal{B}$ pull back to the same selected RegSeqRat tail classifier on every refined chart V_a. Hence Definition 38.365 identifies them on the union. Since every step used only the displayed rows just named, the excluded quotient, completion, Classical, metric, arithmetic, order, bridge, and hidden-selector data have no projection into the comparison. □

Theorem 38.362 (RegSeqRat tail-classifier directed-refinement normal form)**.** *Let $\{\mathcal{A}_r\}_{r\in R}$ be a finite directed family of finite atlases of selected* Real$^{\uparrow}$ *observation-window packages. Suppose every transition $\mathcal{A}_r \to \mathcal{A}_s$ with $r \le s$, every displayed cofinal refinement, and every comparison pullback preserves the displayed* StreamName$^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. Choose any cofinal refined atlas row and read its selected* RegSeqRat *tail classifier.*

Then that selected classifier agrees with the normal-form tail row on the directed union. The comparison is the unique finite-window row whose pullback to every selected tail window is the displayed chart classifier, and it uses no quotient, completion, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden selector.

Proof. Apply Definition 38.375 to each finite atlas in the directed family. The directed hypotheses give a chartwise selected row on every stage, and each such row is built only from the displayed StreamName, RegSeqRat, dyadic, unary-history, probe, membership, transport, package, and policy data.

For a transition $r \leq s$, use Definition 38.361 and Definition 38.373. They identify the selected row on $\mathcal{A}_s$ with the pullback of the selected row on $\mathcal{A}_r$ along the displayed finite refinement.

Cofinal changes of the directed atlas are reduced to one finite comparison by Definition 38.374, and Definition 38.376 removes the charts outside the selected cofinal row. Thus both the cofinal read and the directed-union normal form have the same pullback on every selected tail window.

Finally apply Definition 38.365. The two finite-window rows are equal because their chart pullbacks agree, and the preserved row inventory listed in the hypotheses leaves no projection for the excluded quotient, completion, Classical, metric, arithmetic, order, bridge, or hidden-selector data. □

Theorem 38.363 (RegSeqRat tail-classifier reindexing normal form)**.** *Let W be a finite* $\mathsf{Real}^{\uparrow}$ *observation-window package, and let $\rho_1, \ldots, \rho_n$ be a finite certified unary reindexing schedule whose adjacent stages preserve the displayed* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. At every stage, read the selected* RegSeqRat *tail classifier from the displayed finite-window surface.*

Then the selected tail row read after any adjacent certified stage, the row read after the composite schedule $\rho_n \circ \cdots \circ \rho_1$, and the row read directly from the normal-form finite-window surface of W are the same finite-window RegSeqRat *tail-classifier row. This normal-form comparison uses only the displayed finite readback rows and introduces no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. Induct on the finite certified schedule. In the empty schedule case, the normal-form read is the direct finite-window read of W; apply Definition 38.360 to obtain the selected RegSeqRat tail row from the displayed StreamName, RegSeqRat, DyadicRatCore, endpoint-transport, ledger, package, and provenance rows.

For the cons step, assume the statement for the initial schedule and append one certified unary reindexer. The induction hypothesis identifies the selected tail row after the initial schedule with the direct normal-form read. Apply Definition 38.370 to the appended stage, using the preserved $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows.

The composite schedule has the same public readback as the stagewise schedule by Definition 38.371. Therefore the row read after the composite schedule agrees with each stagewise selected tail row and with the normal-form finite-window row.

Finally apply Definition 38.365. Any candidate tail row compatible with all stagewise pullbacks factors through the same displayed finite-window surface, so there is only one selected tail classifier. Definition 38.342 supplies the exclusion boundary: quotient, completion, Classical, metric-completeness, arithmetic, order, standard-bridge, and hidden selector rows are outside the scoped finite surface. □

Theorem 38.364 (RegSeqRat tail-classifier normal-form tail coverage)**.** *Let W be a finite* $\mathsf{Real}^{\uparrow}$ *observation-window package, and let $\rho_1, \ldots, \rho_n$ be a finite certified unary reindexing schedule whose adjacent stages preserve the displayed* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. For each stagewise read, for the read after the composite schedule $\rho_n \circ \cdots \circ \rho_1$, and for the direct normal-form finite-window read of W, the selected tail row is covered by one and the same selected* RegSeqRat *classifier row.*

The covering row is the finite-window classifier row exposed by the displayed $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore, endpoint-transport, continuation-ledger, package, and* RealLedgerPolicy *data. The coverage statement has no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden selector row.*

Proof. Apply Definition 38.363. It identifies the selected tail row read at each certified stage, the selected row read after the composite schedule, and the selected row read directly from the normal-form finite-window surface.

Now use Definition 38.360 on that common finite-window surface. The exactness theorem says that the selected tail is not a separate row but the displayed RegSeqRat classifier row read from the shared StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, Pkg, and RealLedgerPolicy data.

It remains to rule out a second mediator for one of the reads. Apply Definition 38.365. Any candidate row compatible with the stagewise, composite, and direct pullbacks factors through the same selected finite-window classifier row, so all selected tails are covered by that single row.

The exclusion boundary is inherited from the exact finite read surface: quotient data, completion data, Classical representatives, metric-completeness certificates, arithmetic or order structure, standard-bridge data, and hidden selector rows are not part of the scoped package. □

Theorem 38.365 (RegSeqRat tail-classifier pullback uniqueness)**.** *Let W_0 and W_1 be finite-window* $\mathsf{Real}^{\uparrow}$ *packages whose displayed* $\mathsf{StreamName}^{\uparrow}$ *classifier rows,* RegSeqRat *spine rows, DyadicRatCore observation rows, endpoint* hsame *transports, and finite* Cont *ledger entries agree as in Definition 38.360. Suppose a candidate tail-classifier row T pulls back to the selected tail of W_0 and also pulls back to the selected tail of W_1 through exactly those displayed rows. Then T is identified with the unique selected* RegSeqRat *tail classifier determined by the common finite-window readback data. No second tail-classifier surface can mediate the finite-window* $\mathsf{Real}^{\uparrow}$ *seal while remaining invisible to the displayed* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* hsame*, and* Cont *rows.*

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratTailClassifierPullbackUniqueness`

Proof. By Definition 38.360, the two selected tails of W_0 and W_1 are the same RegSeqRat tail classifier after the displayed endpoint transports in the shared ledger are applied.

It remains to exclude a hidden mediator. The scoped read surface of Definition 38.342 lists every row that a finite-window $\mathsf{Real}^{\uparrow}$ consumer may read: the stream-name classifier row, the regular-sequence spine, the dyadic observation rows, the endpoint transport ledger, and the finite continuation ledger.

Finally, Definition 38.350 packages exactly that surface as the readback boundary. Since T pulls back through no rows other than those displayed rows, it has no independent slot in the finite package. Therefore the two pullbacks of T factor through the same selected tail, and T is the unique selected RegSeqRat tail classifier available at the finite-window seal. □

Theorem 38.366 (RegSeqRat tail-classifier union exhaustion)**.** *Let $\{W_i\}_{i \in I}$ be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. Assume that every certified common subwindow*

used by the finite cover displays the same $\mathsf{StreamName}^{\uparrow}$ *row, the same* RegSeqRat *spine row, the same DyadicRatCore observation row, matching endpoint* hsame *transports, and the same finite* Cont *ledger entries. Also assume that these certified common subwindows cover the union of the visible windows. Then the union window exposes one selected* RegSeqRat *tail-classifier row.*

Moreover, any candidate RegSeqRat *tail row whose pullback agrees with the selected tail row on every member window factors uniquely through the selected tail-classifier row on the union window. The union read surface adds no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. First restrict the family to a certified common subwindow. By Definition 38.360, the shared $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, endpoint-transport, and finite ledger rows determine the same selected tail classifier on that subwindow.

Next compare two overlaps in the finite cover. Apply Definition 38.365 to the two restricted readback surfaces. Any mediator that pulls back to both selected tails has no independent row outside the displayed finite readback inventory, so the overlap contributes the same mediating tail row.

Now exhaust the finite union. Use Definition 38.326 to repack the certified common subwindows into one finite $\mathsf{Real}^{\uparrow}$ union surface, and use Definition 126.108 to keep the StreamName and RegSeqRat handoff rows fixed across the cover. The repacked surface therefore carries a single selected tail classifier.

Finally let a candidate tail row pull back to every member window. The previous overlap comparison identifies all of its member-window pullbacks with the same selected row, and the union repacking has no further tail-classifier slot. Thus the candidate factors uniquely through the selected RegSeqRat tail row on the finite union window. □

Theorem 38.367 (RegSeqRat tail-classifier partition coherence)**.** *Let* $\{W_i\}_{i\in I}$ *be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages, and let the certified subwindows of this family cover a displayed finite window* W*. Assume that each pairwise overlap displays the same* $\mathsf{StreamName}^{\uparrow}$ *row, the same* RegSeqRat *spine row, the same DyadicRatCore observation row, matching endpoint* hsame *transports, matching finite* Cont *rows, matching* Pkg *provenance, and the same* RealLedgerPolicy *rows.*

Then the assembled finite partition exposes the same selected RegSeqRat *tail-classifier row as the direct selected read of* W*. Moreover, any local* RegSeqRat *tail row whose pullbacks are compatible on all member windows factors uniquely through that direct selected tail. This partition read introduces no quotient representative, completion row, host stream datum, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. Apply Definition 38.328 to the certified finite cover. The public finite BHist partition surface and the direct finite-window read of W therefore have the same displayed $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, endpoint hsame, finite Cont, Pkg, and RealLedgerPolicy rows.

Next apply Definition 38.366 to the same finite family. Since the overlap hypotheses keep the selected tail row fixed on every certified subwindow, the finite union carries one selected RegSeqRat tail classifier.

Finally use Definition 38.365 on the overlap readback surfaces. A local tail row compatible on all member windows has no extra mediator outside the displayed partition rows, so it factors uniquely through the direct selected RegSeqRat tail of W. □

Theorem 38.368 (RegSeqRat tail-classifier finite-cover sheaf condition)**.** *Let $\{W_i\}_{i\in I}$ be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. Suppose every certified overlap displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same selected* RegSeqRat *spine row, the same DyadicRatCore observation row, the same* UnaryHistory*,* ProbeBundle*, and* InBundle *rows, and matching finite* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. If the local selected* RegSeqRat *tail-classifier rows are compatible on all certified overlaps, then they glue to a unique selected global tail row on the finite union window. Each local selected row is the pullback of this global row.*

The glued surface contains only the displayed finite readback rows. It imports no quotient, completion, Classical representative, metric-completeness, order, arithmetic, standard bridge, or hidden selector row.

Proof. First apply Definition 38.360 on each member window. The selected local tail row is thereby determined by the displayed $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows.

Next use Definition 38.366. The finite family has a displayed union readback surface, and every local selected tail row is read through that surface after restriction to its member window.

Compatibility on overlaps is the partition condition of Definition 38.367. It identifies the member-window restrictions with the direct selected read of the finite union, so the union surface carries one candidate global tail row.

Finally apply Definition 38.365. Any other global row with the same member-window pullbacks has no additional mediator outside the displayed finite union rows, hence it is the same selected global tail row. The exclusion sentence follows from the exactness and union exhaustion boundaries: no quotient, completion, Classical representative, metric-completeness, order, arithmetic, standard bridge, or hidden selector row appears in the glued surface. □

Theorem 38.369 (RegSeqRat tail-classifier descent exactness)**.** *Let $\{W_i\}_{i\in I}$ be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages with compatible local selected* RegSeqRat *tail-classifier rows on every certified overlap. Let ρ be a certified unary* $\mathsf{StreamName}^{\uparrow}$ *reindexing schedule from the displayed finite union window to a refined finite-window surface. Suppose ρ preserves the displayed* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows at each certified stage.*

Then the glued global RegSeqRat *tail row descends uniquely through the reindexed finite-window surface. Every local selected row is the pullback of that descended row along its member-window readback, and the descent surface contains no quotient, completion, Classical representative, metric-completeness, arithmetic or order row, standard-bridge row, or hidden selector row.*

Proof. First apply Definition 38.368. The compatible local selected tail-classifier rows glue to a unique global selected RegSeqRat tail row on the finite union window, and every local row is its pullback.

Next transport that global row along the certified schedule. For one unary reindexing step use Definition 38.370; for a finite schedule use Definition 38.371. The displayed $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows are therefore the same rows seen on the refined finite-window surface.

Finally apply Definition 38.365. A second descended tail row with the same local pullbacks would pull back through the same displayed finite readback rows, so it is the transported global row. The same row inventory excludes quotient, completion, Classical representative, metric-completeness, arithmetic, order, standard-bridge, and hidden selector data. □

Theorem 38.370 (RegSeqRat tail-classifier reindexing stability)**.** *Let W_0 and W_1 be finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages, and let ρ be a certified unary reindexer between their displayed* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose the two packages have matching* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows after reindexing by ρ, matching* RegSeqRat *spine rows, matching DyadicRatCore observation rows, matching* UnaryHistory*,* ProbeBundle*, and* InBundle *membership rows, and matching finite* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows.*

Then the RegSeqRat *tail classifier selected after reindexing is the same finite-window readback row for W_0 and W_1. This stability statement uses only the displayed finite rows. It imports no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. First apply Definition 126.155 to the displayed $\mathsf{StreamName}^{\uparrow}$–RegSeqRat handoff surface. The certified unary reindexer ρ identifies the reindexed StreamName classifier rows together with the visible UnaryHistory, ProbeBundle, and InBundle membership rows.

Next apply Definition 126.157. Since the reindexed classifier rows and the finite handoff rows match, every downstream RegSeqRat finite-window consumer reads the same source surface and the same selected input rows from both packages.

Now use Definition 38.360. The common StreamName, RegSeqRat, DyadicRatCore, endpoint-transport, ledger, and provenance rows determine one selected RegSeqRat tail classifier. Then Definition 38.365 rules out a second tail row pulling back through the same displayed finite readback surface.

Finally close the surface with Definition 38.342. Its row inventory is exactly the visible finite-window readback inventory, so quotient, completion, Classical, metric-completeness, arithmetic, order, standard-bridge, and hidden selector rows have no slot in the stabilized reindexed package. □

Theorem 38.371 (RegSeqRat tail-classifier reindexing composition)**.** *Let W_0 and W_1 be finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. Let ρ be a certified unary reindexer from the displayed window of W_0 to an intermediate displayed* $\mathsf{StreamName}^{\uparrow}$ *window, and let σ be a certified unary reindexer from that intermediate window to the displayed window of W_1. Suppose the two packages have matching* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows after first reindexing by ρ and then by σ, matching* RegSeqRat *spine rows, matching DyadicRatCore observation rows, matching* UnaryHistory*,* ProbeBundle*, and* InBundle *membership rows, and matching finite* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows at both displayed reindexing stages.*

Then the RegSeqRat *tail classifier selected after the composite reindexing schedule $\sigma \circ \rho$ is the same finite-window readback row as the classifier obtained by applying the two certified reindexing steps successively. The composition surface is only the displayed finite readback surface; it introduces no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. First apply Definition 126.155 to the ρ-reindexed handoff surface. The displayed $\mathsf{StreamName}^{\uparrow}$ classifier, RegSeqRat spine, UnaryHistory, ProbeBundle, and InBundle membership rows form the intermediate finite readback boundary.

Apply the same exhaustion theorem to the σ-reindexed handoff surface. The intermediate boundary is thereby transported to the final displayed finite-window boundary with the same RegSeqRat spine, DyadicRatCore observation rows, endpoint hsame transports, finite Cont ledger, Pkg row, and RealLedgerPolicy provenance.

Next use Definition 126.157. A consumer reading after $\sigma \circ \rho$ sees exactly the same displayed StreamName and RegSeqRat source rows as a consumer that reads after ρ and then after σ. Thus both schedules feed the tail selector with the same finite-window row inventory.

Now invoke Definition 38.370 for the ρ-stage and again for the σ-stage. Each stage preserves the selected RegSeqRat tail classifier on the displayed finite rows, so the successive readback row agrees with the row exposed by the composite reindexing schedule.

It remains to show that this agreement is the exact public surface. Apply Definition 38.360 to the common StreamName, RegSeqRat, DyadicRatCore, endpoint-transport, ledger, package, and provenance rows. Then Definition 38.365 excludes any second tail-classifier row pulling back through the two-stage schedule.

Finally use Definition 38.342. The scoped row inventory contains only the finite-window readback rows displayed above, so quotient, completion, Classical, metric-completeness, arithmetic, order, standard-bridge, and hidden selector data have no slot in the composite surface. □

Theorem 38.372 (RealUp RegSeqRat tail-window reindexing boundary)**.** *Let W be a finite* $\mathsf{Real}^{\uparrow}$ *observation-window package and let ρ be a certified unary tail map from its displayed window to a reindexed* $\mathsf{StreamName}^{\uparrow}$ *finite window. Suppose the reindexed source rows match the tail-window normal forms: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the selected* ProbeBundle *and* InBundle *membership rows, the* UnaryHistory *observations, the* RegSeqRat *spine and tail-classifier rows, the DyadicRatCore observation rows, the finite* hsame *and* Cont *ledger, the* Pkg *provenance row, and the displayed* RealLedgerPolicy *row.*

Then the finite $\mathsf{Real}^{\uparrow}$ *observation window reindexed through ρ consumes exactly those rows. Its handoff boundary is the StreamName finite-window classifier surface together with the RegSeqRat tail-classifier surface transported by the certified unary tail map. It imports no* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, host-index equality, Classical representative, metric-completeness witness, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector.*

Proof. First apply Definition 126.155. The reindexed window is therefore exhausted by the displayed $\mathsf{StreamName}^{\uparrow}$ classifier, ProbeBundle, InBundle, UnaryHistory, RegSeqRat spine, and finite ledger rows.

Next use Definition 38.370 and Definition 38.371. These two results identify the row obtained by the certified unary tail map with the same selected RegSeqRat tail classifier carried by the displayed tail-window normal form.

Now apply Definition 38.373. It shows that the same projection remains valid after finite-window cover gluing, so the reindexed boundary does not depend on a hidden choice of local tail classifier.

Finally combine the StreamName side with Definition 126.170 and Definition 126.44. These handoff theorems expose exactly the finite StreamName rows that a RegSeqRat or $\mathsf{Real}^{\uparrow}$ consumer may read. Repacking those rows with the RegSeqRat tail normal form gives the stated $\mathsf{Real}^{\uparrow}$ boundary and leaves no slot for seal, completion, quotient, host-index, order, arithmetic, bridge, or hidden selector data. □

Theorem 38.373 (RegSeqRat tail-classifier reindexed finite-cover compatibility)**.** *Let $\{W_i\}_{i\in I}$ be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages with compatible local selected* RegSeqRat *tail-classifier rows on every certified overlap. Let ρ_i be certified unary reindexers for the member windows, and let ρ be the certified unary reindexer induced on the displayed finite union window. Suppose these reindexers preserve the displayed* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*, finite* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy

rows, and suppose the transported overlap rows are the same rows as the overlaps of the reindexed member windows.

Then the selected RegSeqRat *tail classifier obtained by first gluing the finite cover and then reindexing by ρ is the same finite-window readback row as the classifier obtained by reindexing each member window by ρ_i and then applying finite-cover gluing to the reindexed cover. The compatibility surface contains no quotient, completion, Classical representative, metric-completeness, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector row.*

Proof. First glue the original cover with Definition 38.368. The compatible local selected rows determine a unique global selected tail row on the displayed finite union, and each member row is its pullback.

Next transport this global row along ρ. The one-step case is Definition 38.370, and the finite-window schedule is handled by Definition 38.371. Thus the glue-then-reindex construction has a displayed selected row on the reindexed finite union.

Now reindex each member window by ρ_i. The same stability theorem identifies every transported local classifier with the selected classifier on the corresponding reindexed member window, and the transported overlap hypothesis gives the compatibility condition required by Definition 38.368. Therefore the reindexed cover also glues to a selected global tail row.

It remains to compare the two global rows. On each reindexed member window, both constructions pull back to the same transported local selected RegSeqRat tail classifier. Apply Definition 38.365 on the reindexed finite cover to identify the two global rows. The row inventory used in each step is exactly the displayed $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy inventory, so the listed quotient, completion, Classical, metric-completeness, arithmetic, order, standard-bridge, and hidden selector rows are excluded. □

Theorem 38.374 (RegSeqRat tail-classifier nested-cover associativity)**.** *Let $\{W_i\}_{i\in I}$ be a finite family of selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages. For each i, let $\{W_{ij}\}_{j\in J_i}$ be a finite certified subcover of W_i by selected finite windows, and suppose all first-level and second-level overlaps preserve the displayed* $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, *DyadicRatCore,* UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, *and* RealLedgerPolicy *rows.*

Then the selected RegSeqRat *tail-classifier row obtained by first gluing each subcover $\{W_{ij}\}_{j\in J_i}$, then gluing the resulting member windows $\{W_i\}_{i\in I}$, is the same finite-window row as the selected classifier obtained by flattening the family $\{W_{ij}\}_{i,j}$ and gluing that single certified finite cover once. The associativity surface exports no quotient, completion, Classical representative, metric-completeness, arithmetic or order row, standard-bridge row, or hidden regular-sequence selector row.*

Proof. First glue each certified subcover $\{W_{ij}\}_{j\in J_i}$. By Definition 38.368, every compatible subfamily determines a unique selected tail-classifier row on W_i, and the pullback of that row to each W_{ij} is the displayed local classifier.

Next glue the resulting family of member windows. The same finite-cover theorem gives a selected global row over the finite union of the W_i, and the overlap hypotheses ensure that its pullback to each W_i is the row obtained from the corresponding subcover.

Now flatten the two-level family to the certified finite cover $\{W_{ij}\}_{i,j}$. Apply Definition 38.373 to identify the flattened cover with the same displayed finite-window inventory used by the staged construction; the reindexing data here is only the certified inclusion and ordering of the flattened subwindows.

It remains to compare the staged row with the flattened row. Both pull back to the same selected RegSeqRat tail-classifier row on every certified subwindow W_{ij}, so Definition 38.365 identifies the two global rows. Each step used only the displayed StreamName$^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows, leaving no slot for the excluded host or completion data. □

Theorem 38.375 (RegSeqRat tail-classifier atlas descent)**.** *Let $\mathcal{A} = \{W_i\}_{i\in I}$ be a finite atlas of selected finite* Real$^{\uparrow}$ *observation-window packages. Suppose every pairwise overlap $W_i \cap W_j$ and every triple overlap $W_i \cap W_j \cap W_k$ preserves the displayed* StreamName$^{\uparrow}$, RegSeqRat, *DyadicRatCore,* UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, *and* RealLedgerPolicy *rows.*

Then the atlas exposes a single selected RegSeqRat *tail-classifier row on the finite union of its windows. For every chart W_i, the chartwise selected tail-classifier row factors through this atlas row, and this factorisation is unique among finite-window mediators preserving the displayed rows. The descent surface exports no quotient, completion, Classical representative, metric-completeness row, arithmetic or order row, standard-bridge row, or hidden selector row.*

Proof. Apply Definition 38.360 to each selected chart. This identifies the chartwise tail-classifier row with the displayed StreamName$^{\uparrow}$, RegSeqRat, DyadicRatCore, unary-history, probe-bundle, membership, transport, continuation, provenance, and ledger inventory.

Next form the finite atlas union. By Definition 38.366, the union consumes exactly the rows contributed by the selected windows and their certified overlaps. No additional host row is introduced while passing from the chart family to the finite union.

Now decompose the atlas by its chart partition. The coherence result Definition 38.367 says that the selected row obtained from the union is the same row obtained by assembling the partition pieces with their displayed overlap transports.

Triple refinements are handled by Definition 38.374. It identifies the staged gluing over pairwise overlaps with the flattened finite cover over all chart and triple-overlap pieces, so the resulting atlas row is independent of the order in which the certified finite overlaps are read.

It remains to show the universal factorisation claim. A chartwise selected tail pulls back from the atlas row by construction. If another finite-window mediator has the same pullback on every chart while preserving the displayed rows, then Definition 38.365 identifies that mediator with the atlas row. Thus each chartwise tail factors uniquely through the displayed atlas descent row, and the excluded quotient, completion, Classical, metric, arithmetic, order, bridge, and hidden-selector data have no projection in the construction. □

Theorem 38.376 (RegSeqRat tail-classifier subatlas restriction exactness)**.** *Let $\mathcal{A} = \{W_i\}_{i\in I}$ be a finite atlas of selected finite* Real$^{\uparrow}$ *observation-window packages satisfying the atlas-descent hypotheses of Definition 38.375. Let $\mathcal{B} = \{W_j\}_{j\in J}$ be a certified finite subatlas whose union is a certified subwindow of the union of $\mathcal{A}$, and suppose each chart of $\mathcal{B}$ is obtained by restricting one chart of $\mathcal{A}$ along the displayed finite-window inclusion ledger.*

Then the selected RegSeqRat *tail-classifier row obtained by restricting the atlas row of $\mathcal{A}$ to the charts of $\mathcal{B}$ is exactly the row obtained by applying finite-cover descent directly to $\mathcal{B}$. This restriction surface preserves precisely the displayed* StreamName$^{\uparrow}$, RegSeqRat, *DyadicRatCore,* UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, *and* RealLedgerPolicy *rows. It exports no quotient, completion, Classical representative, metric-completeness row, arithmetic or order row, standard-bridge row, or hidden-selector row.*

Proof. Apply Definition 38.375 to the ambient atlas $\mathcal{A}$. This supplies a single selected atlas row whose pullback to each ambient chart is the chartwise RegSeqRat tail-classifier row over the dis-

played $\mathsf{StreamName}^{\uparrow}$, DyadicRatCore, unary-history, probe-bundle, membership, transport, continuation, provenance, and ledger data.

Restrict those pullbacks along the certified inclusions defining the subatlas $\mathcal{B}$. Since each subatlas chart is a finite-window restriction of an ambient chart, the restricted pullbacks still expose only the displayed UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows.

Now apply Definition 38.368 directly to the certified finite cover $\mathcal{B}$. The pairwise compatibility required on $\mathcal{B}$ is inherited from the ambient pairwise and triple overlap hypotheses, so the finite-cover theorem produces a selected RegSeqRat tail-classifier row on the subwindow union.

It remains to compare the restricted ambient row with the row constructed directly on $\mathcal{B}$. Both rows pull back to the same selected tail-classifier on every chart W_j of the subatlas and both preserve the same displayed inventory. Therefore Definition 38.365 identifies them. The comparison has no projection to quotient, completion, Classical, metric, arithmetic, order, standard-bridge, or hidden-selector data because neither construction contains such a row. □

Theorem 38.377 (RealUp terminal refinement seal boundary)**.** *Let E be a* $\mathsf{Real}^{\uparrow}$ *finite-window refinement envelope over a fixed* $\mathsf{StreamName}^{\uparrow}$ *source. Suppose two refined finite-window presentations W and W' are carried by E, are connected by the directed-refinement seal of Definition 38.329, and are then read by a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer. Then every terminal consumer of either presentation reads exactly the same finite source boundary:*

1. *the* RatStreamNameFiniteWindowClassifier *rows;*
2. *the bundled* UnaryHistory *and* InBundle *rows;*
3. *the pointwise* RatHistoryClassifier *rows;*
4. *the accepted* RegSeqRat *finite-window spine;*
5. *the* DyadicRatCore *source rows;*
6. *the visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *ledger rows.*

No terminal read exposes a quotient row, completion object, host Nat *index, host function,* Classical *datum, metric-completeness certificate, arithmetic row, order row, or standard bridge row.*

Proof. First identify both refined presentations with the same envelope boundary. Apply Definition 38.331 to the two windows carried by E. The result gives one public finite BHist surface containing exactly the envelope classifier, endpoint, dyadic-source, hsame, and ledger-policy rows.

Next use the directed-refinement seal. The comparison row supplied by Definition 38.329 forces the presentation read from W and the presentation read from W' through that same finite refinement surface. Thus the two refined windows have no distinct source boundary for a later consumer to inspect.

Now pass this common finite-window surface to the terminal seal consumer. Apply Definition 126.45. It reads the finite StreamName-to-RegSeqRat packet as the selected classifier rows, bundled unary and membership rows, pointwise rational classifier rows, endpoint and denominator rows, the accepted RegSeqRat spine, the dyadic source rows, and the finite Cont and hsame transport ledger.

Finally apply Definition 126.46. The scope readback keeps the source Pkg rows and RealLedgerPolicy boundary attached to the same packet and excludes all rows outside it. Hence no terminal consumer can read quotient data, completion data, host indices, host functions, Classical data, metric-completeness certificates, arithmetic or order rows, or standard bridge rows. □

Theorem 38.378 (RealUp directed-refinement terminality)**.** *Let $\mathcal{F}$ be a directed finite family of StreamName tail-window coverage packets over the same public* $\mathsf{StreamName}^{\uparrow}$ *endpoints and the same accepted* RegSeqRat *finite-window spine. Then the directed common-refinement packet obtained from $\mathcal{F}$ is terminal for finite* $\mathsf{Real}^{\uparrow}$ *consumers: every finite* $\mathsf{Real}^{\uparrow}$ *consumer whose source reads one member of $\mathcal{F}$ factors through this common-refinement packet.*

For any selected subwindow, restriction from the common-refinement packet recovers the same RatStreamNameFiniteWindowClassifier *rows, the same* UnaryHistory *and* InBundle *rows, the same pointwise* RatHistoryClassifier *rows, the same accepted* RegSeqRat *spine, the same* DyadicRatCore *endpoint rows, and the same visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows as the original StreamName finite-window packet.*

The terminal factorization introduces no completion object, quotient row, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.

Proof. First form the common packet. Apply Definition 126.221 to any two members of the directed family; directedness supplies a family member whose selected observation window contains both restrictions, so the generated packet has the same public endpoints and accepted RegSeqRat spine.

For more than two selected windows, iterate the same construction along the finite family. Apply Definition 126.222 at each regrouping step. The resulting packet is independent of the parenthesization, and every selected subwindow restricts back to the same StreamName classifier, unary-history, bundle-membership, pointwise rational-history, accepted spine, dyadic endpoint, continuation, history, package, and ledger-policy rows.

Next pass a finite $\mathsf{Real}^{\uparrow}$ consumer through the packet boundary. Apply Definition 126.224. Every consumer row factors through the displayed tail-window ledger, and equality of the displayed packet gives equality of the finite read surface.

Finally compare this terminal read with the $\mathsf{Real}^{\uparrow}$ seal boundary. Apply Definition 38.377. The terminal consumer therefore reads exactly the rows recovered by the selected subwindow restriction and no object outside that finite surface. In particular, the construction adds no completion object, quotient row, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row. □

Theorem 38.379 (StreamName Real-RegSeq terminal source uniqueness)**.** *Let C be a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package. Suppose two terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumers are reached from C and that their displayed* RegSeqRat *handoff rows expose the same source data:*

1. *the* RatStreamNameFiniteWindowClassifier *rows;*
2. *the* UnaryHistory *and* InBundle *rows;*
3. *the pointwise* RatHistoryClassifier *rows;*
4. *the accepted* RegSeqRat *spine;*
5. *the* DyadicRatCore *rows; and*
6. *the visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows.*

Then the two terminal consumers have the same source-boundary projection. Conversely, equality of the terminal source-boundary projection is exactly equality of the rows listed above.

The projection contains no quotient row, completion object, host index, host function, Classical *datum, metric-completeness row, arithmetic or order row, or standard bridge row.*

Proof. Begin with the two routes from the public $\mathsf{StreamName}^{\uparrow}$ package C to the terminal finite-window consumers. By Definition 38.380, route choice cannot change the finite StreamName source packet read at the terminal interface.

Next identify the packet read by the terminal $\mathsf{Real}^{\uparrow}$ seal. Apply Definition 38.382. The terminal source surface is precisely the displayed StreamName classifier, unary-history, membership, pointwise rational-history, accepted RegSeqRat spine, dyadic-core, transport, package, and real-ledger rows.

For uniqueness of the accepted spine, use Definition 38.365. Any two terminal candidates compatible with those displayed rows factor through the same selected RegSeqRat source surface, so the source-boundary projection is forced to agree.

For the converse, apply Definition 126.170. Equality of the source-boundary projection reads back exactly the StreamName finite-window classifier rows, the unary and membership rows, the pointwise rational classifier rows, the accepted RegSeqRat spine, the dyadic-core rows, and the visible Cont, hsame, Pkg, and RealLedgerPolicy rows.

The coverage theorem also gives the exclusion clause. Since the projection is the finite RegSeqRat source surface and nothing outside it, no quotient row, completion object, host index, host function, Classical datum, metric-completeness row, arithmetic or order row, or bridge row can be recovered from the terminal consumer. □

Theorem 38.380 (StreamName common-refinement terminal route determinacy)**.** *Let C be a certified* $\mathsf{StreamName}^{\uparrow}$ *common-refinement package. Suppose two finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal consumers are reached from C by choosing possibly different common-refinement routes, then passing through the same StreamName finite-window restriction and* RegSeqRat *handoff interface. Then the two terminal reads expose the same source tuple:*

1. *the restricted* RatStreamNameFiniteWindowClassifier *rows;*
2. *the common* ProbeBundle*,* InBundle*, and* UnaryHistory *rows;*
3. *the pointwise* RatHistoryClassifier *rows;*
4. *the accepted* RegSeqRat *spine selected by those rows;*
5. *the displayed dyadic endpoint rows; and*
6. *the displayed* hsame*,* Cont*, and* Pkg *provenance rows.*

The terminal tuple contains no quotient row, completion object, host-index equality, metric-completeness datum, or additional $\mathsf{Real}^{\uparrow}$ *seal classifier outside the packet just listed.*

Proof. Start with the certified common-refinement package C. Apply Definition 126.236. The two routes therefore determine the same StreamName source packet and the same displayed common-refinement provenance rows.

Next restrict that packet to the terminal finite window. By Definition 126.221, both route reads use the same finite-window classifier rows, the same ProbeBundle and InBundle entries, and the same unary-history window.

Now apply Definition 126.170. The restricted StreamName packet hands off to one pointwise rational-history packet, one accepted RegSeqRat spine, and the same dyadic endpoint rows, with the displayed hsame, Cont, and Pkg provenance carried along unchanged.

Finally use Definition 38.382. A terminal finite-window $\mathsf{Real}^{\uparrow}$ seal consumer reads exactly that RegSeqRat source surface and no larger host structure. Since both routes feed the same packet into this coverage theorem, the two terminal reads expose the same listed source tuple and exclude quotient rows, completion objects, host-index equalities, metric-completeness data, and any extra $\mathsf{Real}^{\uparrow}$ seal classifier. □

Theorem 38.381 (StreamName public window prefix determinacy)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, and let B' and B be probe windows. Suppose every certified* UnaryHistory *observation in B' is carried by B. Then every* RegSeqRat *terminal consumer, and every finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal consumer, that reads* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B)$ *through the public* $\mathsf{StreamName}^{\uparrow}$ *interface has the same terminal BEDC readback after restriction to B' as the consumer obtained by reading the certified prefix window B' directly.*

The common restricted readback consists exactly of the pointwise $\mathsf{Rat}^{\uparrow}$ *observation rows, the* InBundle *witnesses, the* hsame *transports, the* Cont *routes, and the* Pkg *provenance rows exposed by the public StreamName interface on B'. If two restricted terminal routes have the same endpoint package, then their finite BEDC readbacks are identical.*

This determinacy statement exports no RegSeqRat *regularity row, no* $\mathsf{Real}^{\uparrow}$ *seal classifier beyond the restricted terminal source surface, no quotient row, no host-index equality, and no completion data.*

Proof. First read the larger public finite-window package. Apply Definition 126.301. It exposes only the finite classifier rows, certified unary-history observations, membership witnesses, pointwise rational observation rows, and visible transport and provenance rows over B.

Next restrict that public packet to the certified prefix window. By Definition 126.302, every UnaryHistory observation carried by B' has the same restricted classifier, membership witness, hsame transport, Cont route, and Pkg provenance as its row inside B.

Now pass the restricted packet through the consumer-facing handoff. Apply Definition 126.170. The RegSeqRat read and the terminal finite-window $\mathsf{Real}^{\uparrow}$ read both see the same pointwise $\mathsf{Rat}^{\uparrow}$ observation rows and the same restricted provenance rows from B'.

Finally use the terminal source comparison theorems. The source uniqueness statement Definition 38.379 identifies any two terminal source projections with the same restricted row inventory, and Definition 38.380 removes the choice of terminal route. Thus two restricted terminal routes with the same endpoint package have the same finite BEDC readback, while all regularity, seal-classifier, quotient, host-index, and completion data remain outside the public prefix surface. □

Theorem 38.382 (RealUp terminal RegSeqRat source coverage)**.** *Every terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through exactly one* RegSeqRat *source surface. That source surface consists of the displayed* RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *and* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore observation rows, and the visible* Cont, hsame, Pkg, *and* RealLedgerPolicy *ledger rows.*

Conversely, any packet carrying exactly those rows repacks into the terminal source surface read by the seal consumer. The coverage contains no quotient row, completion object, host index, host function, Classical *datum, metric-completeness row, arithmetic or order row, or standard bridge row.*

Proof. Start from the terminal read inventory of Definition 38.377. It lists precisely the finite classifier, unary-history, membership, pointwise rational, accepted spine, dyadic-source, transport, package, and ledger rows visible to a terminal seal consumer.

For the source side, apply the finite-window StreamName-to-RegSeqRat handoff from Definition 38.344 and Definition 38.268. These identify the StreamName finite-window classifiers and dyadic observation rows as the same source packet consumed by the RegSeqRat seal.

Now use the tail-classifier exactness package. By Definition 38.360 and Definition 38.364, the accepted RegSeqRat spine is the selected tail-classifier row determined by the displayed StreamName, dyadic, endpoint-transport, package, and RealLedgerPolicy data.

It remains to check uniqueness and repacking. Apply Definition 38.365: any terminal candidate compatible with the displayed rows factors through the same selected RegSeqRat source surface. Conversely, carrying those rows supplies every entry in the inventory above, so the packet repacks into the terminal seal source surface and leaves all excluded quotient, completion, host, Classical, metric, arithmetic, order, and bridge rows unread. □

Theorem 38.383 (RealUp terminal RegSeqRat source minimality)**.** *Every terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through a minimal* BHist *source surface. That surface consists exactly of the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, unary-history membership rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* DyadicRatCore *rows, the endpoint transport rows, and the visible* Cont *and* Pkg *ledger rows.*

Removing any one of those row classes prevents repacking into the terminal source surface read by the $\mathsf{Real}^{\uparrow}$ *seal consumer. The minimal surface contains no quotient row, completion object, host-index equality, metric-completeness datum, or separate seal classifier.*

Proof. Apply Definition 38.382. Its coverage direction gives the listed RegSeqRat source surface as the entire read boundary of any terminal finite-window $\mathsf{Real}^{\uparrow}$ seal consumer, and its converse direction repacks exactly packets carrying those rows.

Uniqueness of the selected source packet is supplied by Definition 38.379. A second candidate surface with the same StreamName finite-window classifier, unary-history, rational-history, accepted-spine, dyadic, transport, and ledger rows is therefore the same terminal source packet.

Route independence is supplied by Definition 38.380. Before a terminal seal consumer reads the packet, every common-refinement route has already fixed the same finite-window classifier rows, membership rows, pointwise rational-history rows, accepted RegSeqRat spine, and dyadic endpoint data.

Minimality follows by constructor and projection inversion on that packet. Omitting a StreamName classifier row removes the finite-window route; omitting a membership or unary-history row removes the window boundary; omitting a pointwise rational-history row removes the rational observation; omitting the accepted spine removes the RegSeqRat source; omitting a dyadic or endpoint transport row removes the endpoint witness; and omitting a visible Cont or Pkg row removes the ledger needed for repacking.

The same inversion shows that no further row can be required. The coverage and determinacy theorems expose no quotient, completion object, host-index equality, metric-completeness datum, or separate terminal seal classifier, so there is no constructor or projection from which such data could be read. □

Theorem 38.384 (RealUp DyadicRatCore StreamName RegSeqRat observation triangle)**.** *For a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *public readback envelope, the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier surface, and the* RegSeqRat *finite-window seal source surface determine the same finite source packet and are mutually recoverable.*

The common packet consists exactly of the RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *and* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* DyadicRatCore *observation rows, the endpoint transport rows, the visible* Cont *and* Pkg *provenance rows, and the* RealLedgerPolicy *rows. It contains no quotient row, completion object, host* Nat *index,* Classical *datum, metric-completeness row, arithmetic or order row, or standard bridge row.*

Proof. First read the dyadic side. Apply Definition 291.23. Its envelope exposes exactly the dyadic observation rows and endpoint transport rows available to the terminal packet, with no quotient or completion object attached.

Next read the finite stream-name side. Apply Definition 126.301. The public finite-window classifier surface supplies the RatStreamNameFiniteWindowClassifier rows, the UnaryHistory and InBundle rows, and the pointwise RatHistoryClassifier rows on the same displayed finite window.

Now read the seal source side. Apply Definition 38.268. The RegSeqRat seal source reads the accepted spine together with the same finite-window classifier, unary-history, membership, pointwise rational, dyadic observation, endpoint transport, Cont, Pkg, and ledger rows.

Finally apply Definition 38.382 and Definition 38.383. Coverage repacks any packet carrying exactly those rows as the terminal source surface, while minimality prevents an additional row class from being part of that terminal source. Hence each of the three projections recovers the same finite source packet, and the excluded quotient, completion, host-index, Classical, metric, arithmetic, order, and bridge rows remain unread. □

Theorem 38.385 (RealUp dyadic StreamName observation-triangle consumer)**.** *Let R be a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer whose source side factors through the triangle of Definition 38.384. Then R exposes exactly the common packet listed there: the* RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *and* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* DyadicRatCore *observation rows, endpoint transport rows, visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows.*

If two such consumer reads agree on the $\mathsf{StreamName}^{\uparrow}$ *finite-window restriction and on the* RegSeqRat *handoff interface, then they expose the same* DyadicRatCore *endpoint rows,* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows. No quotient row, host index, completion datum, arithmetic/order row, or standard bridge row is part of that consumer-facing source packet.*

Proof. Apply Definition 38.384. The factoring hypothesis places the consumer source on the triangle's common packet, so its dyadic readback, finite StreamName restriction, and RegSeqRat handoff are all projections of the same displayed source surface.

Apply Definition 38.382 and Definition 38.383. Coverage repacks the listed rows as a terminal finite-window $\mathsf{Real}^{\uparrow}$ source, while minimality says that no further source row can be demanded by such a consumer.

For two reads, use Definition 38.386. Agreement on the StreamName finite-window restriction and the RegSeqRat handoff fixes the common-refinement splice, hence fixes the DyadicRatCore endpoint rows and the visible Cont, hsame, Pkg, and RealLedgerPolicy rows.

The excluded data have no projection from any of these three inputs. Quotient rows, host indices, completion data, arithmetic/order rows, and standard bridge rows are absent from the triangle packet and remain absent after terminal source coverage and splice uniqueness are applied. □

Theorem 38.386 (RealUp terminal common-refinement splice uniqueness)**.** *Let C be a certified* $\mathsf{StreamName}^{\uparrow}$ *common-refinement packet whose displayed source tuple consists of a fixed* $\mathsf{StreamName}^{\uparrow}$ *source name, a fixed accepted* RegSeqRat *finite-window spine, and fixed* DyadicRatCore *source rows. Suppose two terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumers are reached from C through possibly different common-refinement splices. If both splices use the same StreamName finite-window restriction and the same* RegSeqRat *handoff interface, then the two terminal consumers expose the same finite tuple:*

1. *the* RatStreamNameFiniteWindowClassifier *rows;*
2. *the* ProbeBundle*,* InBundle*, and* UnaryHistory *rows;*
3. *the pointwise* RatHistoryClassifier *rows;*

4. *the accepted* RegSeqRat *spine;*

5. *the* DyadicRatCore *observation rows; and*

6. *the visible* Cont, hsame, Pkg, *and* RealLedgerPolicy *rows.*

This uniqueness statement is only about that terminal source tuple. It does not identify or expose quotient rows, host Nat *indices, host functions,* Classical *data, metric-completeness rows, arithmetic/order rows, or standard bridge rows.*

Proof. First compare the two splice routes before the terminal consumer reads them. Apply Definition 38.380. The possibly different common-refinement routes through C determine one terminal StreamName source packet, one restricted finite-window classifier, one bundled probe and unary-history packet, one pointwise rational-history packet, one accepted RegSeqRat spine, and one displayed dyadic source surface.

Next pass that route-confluent packet through the terminal $\mathsf{Real}^{\uparrow}$ seal. Apply Definition 38.382. The terminal consumer factors through exactly the listed RegSeqRat source surface, and any packet carrying exactly those rows repacks into the same terminal source surface.

Both consumers therefore have the same displayed source tuple as their entire terminal read boundary. Since the coverage theorem also excludes all rows outside that packet, the uniqueness conclusion contains no quotient row, host index, host function, Classical datum, metric-completeness row, arithmetic or order row, or standard bridge row. □

Theorem 38.387 (RealUp finite-window seal cofinality)**.** *Let $\mathcal{F}$ be a finite directed family of terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumers over the same* $\mathsf{StreamName}^{\uparrow}$ *source. Suppose $\mathcal{F}$ has a cofinal common-refinement read C, and suppose the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows of every member of $\mathcal{F}$ agree with the corresponding rows read from C. Then every member of $\mathcal{F}$ and the cofinal read C determine the same terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal source surface.*

The determined source surface consists only of the finite $\mathsf{StreamName}^{\uparrow}$ *restriction, the accepted* RegSeqRat *spine, the* DyadicRatCore *observation rows, and the displayed* hsame, Cont, Pkg, *and* NameCert *provenance rows. It exports no quotient stream, host-index equality, metric-completion datum,* $\mathsf{Real}^{\uparrow}$ *completeness row, or additional seal classifier.*

Proof. First use Definition 38.378. Directed terminality turns the finite family into a single terminal comparison against the cofinal common-refinement read C, so each member is read through the same terminal source route.

Next apply Definition 38.382. The cofinal read and every family member factor through the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows. Apply Definition 38.383 to remove every row not present in that finite source inventory.

It remains to identify the member reads with the cofinal read. For each member, apply Definition 38.386 to the splice through C. The assumed agreement of the displayed rows fixes the terminal splice, hence fixes the terminal source surface.

The last paragraph gives the stated boundary: the four cited theorems only project the finite StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, Pkg, and NameCert rows. There is no constructor in this route that exports quotient streams, host-index equalities, metric-completion data, $\mathsf{Real}^{\uparrow}$ completeness rows, or a further seal classifier. □

Theorem 38.388 (RegSeqRat terminal tail-refinement seal)**.** *Let S be a terminal* $\mathsf{StreamName}^{\uparrow}$ *finite-window source and let T be a* RegSeqRat *stream whose finite-window reads factor through the*

terminal $\mathsf{Real}^{\uparrow}$ *source surface determined by* S*. For every certified tail refinement* C *of* T *over the same public* BHist *window, there is a unique terminal finite-window seal for* C*.*

This seal is determined exactly by the terminal $\mathsf{StreamName}^{\uparrow}$ *source name, the restricted* RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *and* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, and the visible* hsame*,* Cont*,* Pkg*,* NameCert*, and* RealLedgerPolicy *rows. It is invariant under* hsame *transport of the displayed history rows and under* Cont*-splicing along the certified common-refinement route. Its ledger is the one inherited from the terminal* $\mathsf{StreamName}^{\uparrow}$ *source packet, not an independent* RegSeqRat *or* $\mathsf{Real}^{\uparrow}$ *ledger.*

The terminal seal exposes no quotient stream, host-index equality, host function, Classical *datum, metric-completion object, arithmetic or order row, standard bridge row, or separate tail-refinement classifier.*

Proof. First identify the source from which the tail refinement can be read. Apply Definition 38.379 and Definition 38.382. The factoring hypothesis places every finite window of T on the terminal source packet selected by S, and the coverage theorem repacks exactly those displayed rows as the terminal $\mathsf{Real}^{\uparrow}$ source surface.

Next compare the certified refinement route with the terminal source route. Apply Definition 38.380. The common-refinement certificate fixes the same terminal StreamName source name, the same restricted classifier rows, the same unary-history and membership rows, the same pointwise rational-history rows, and the same accepted RegSeqRat spine before the terminal seal is read.

Now splice the route into the terminal $\mathsf{Real}^{\uparrow}$ consumer. Apply Definition 38.386. Agreement on the StreamName restriction and on the RegSeqRat handoff fixes the $\mathsf{DyadicRatCore}^{\uparrow}$ observation rows together with the visible hsame, Cont, Pkg, NameCert, and RealLedgerPolicy rows. Therefore transporting the displayed history data by hsame or changing only the Cont-splice representative cannot produce a second terminal seal.

Finally apply Definition 38.387. Cofinality collapses every certified tail-window refinement of T to the same terminal finite-window source surface. Since the only ledger rows in that surface are the rows already read from the terminal $\mathsf{StreamName}^{\uparrow}$ source packet, the seal inherits that ledger and has no constructor that could export quotient, completion, host, arithmetic, order, bridge, or separate classifier data. □

Theorem 38.389 (RealUp terminal tail-refinement family factorization)**.** *Let* S *be a terminal* $\mathsf{StreamName}^{\uparrow}$ *finite-window source, let* T *be an accepted* RegSeqRat *packet over* S*, and let* $C_1, \ldots, C_n$ *be a finite family of certified tail refinements of* T *over the same public* BHist *window. Suppose the family admits a cofinal certified common refinement* C_* *and that all displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*,* NameCert*, and* RealLedgerPolicy *rows agree after reading along* C_**.*

Then every terminal finite-window $\mathsf{Real}^{\uparrow}$ *seal for a member* C_i *factors through the single cofinal source surface determined by* C_**. This source surface consists exactly of the finite* $\mathsf{StreamName}^{\uparrow}$ *restriction, the accepted* RegSeqRat *spine, the* DyadicRatCore *observation rows, and the displayed provenance rows* hsame*,* Cont*,* Pkg*,* NameCert*, and* RealLedgerPolicy*.*

The factorization exports no quotient stream, host-index equality, host function, Classical *datum, metric-completion object, arithmetic or order row, standard bridge row, or separate tail-refinement classifier.*

Proof. For each family member C_i, apply Definition 38.388. The seal theorem says that the terminal finite-window seal for C_i is read from the terminal $\mathsf{StreamName}^{\uparrow}$ source restriction, the

accepted RegSeqRat spine, the DyadicRatCore$^{\uparrow}$ observation rows, and the visible transport, package, certificate, and ledger rows.

Now use the cofinal certificate C_*. Apply Definition 38.387 to the finite family $C_1, \ldots, C_n$. Cofinality replaces the separate member reads by the single terminal read through C_*, so every member seal has the same finite source surface once the displayed rows have been transported to the cofinal refinement.

It remains to identify the splice rows on that common read. Apply Definition 38.386. The assumed agreement of the RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, hsame, Cont, Pkg, NameCert, and RealLedgerPolicy rows fixes the terminal splice over C_*, hence fixes the factor through the common source surface.

The boundary clause follows from the same three cited results: their constructors expose only the finite StreamName restriction, the accepted RegSeqRat spine, the dyadic observation rows, and the displayed provenance rows. None of them constructs quotient, host, classical, completion, arithmetic, order, bridge, or separate classifier data. □

Theorem 38.390 (RealUp common-refinement seal pullback)**.** *Let C be a certified* StreamName$^{\uparrow}$ *common-refinement package, and let R and R' be finite-window* Real$^{\uparrow}$ *terminal seal consumers reached from C through compatible accepted* RegSeqRat *tail-refinement seals. Then equality of the common-refinement source tuple is equivalent to equality of the visible terminal seal boundary.*

The common-refinement source tuple consists exactly of the restricted RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *rows, the* ProbeBundle *rows, the* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* DyadicRatCore *endpoint rows, the visible* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows, and the certified* StreamName$^{\uparrow}$ *provenance route in C.*

The visible terminal seal boundary consists of the same finite BHist *ledger rows after the terminal* Real$^{\uparrow}$ *consumer read. It contains no completion object, quotient row, host index, metric-completeness datum, arithmetic row, order row, or standard bridge row.*

Proof. First assume equality of the common-refinement source tuple. Apply Definition 38.380 to the two routes out of C. The route comparison identifies the restricted StreamName classifier rows, probe and unary-history rows, bundle membership, pointwise rational-history rows, accepted RegSeqRat spine, dyadic endpoint rows, and visible transport and provenance rows before either terminal consumer reads the packet.

Next pass the identified source tuple through the terminal Real$^{\uparrow}$ surface. Apply Definition 38.382 and Definition 38.388. Coverage says that the terminal consumer can read only the listed finite source rows, and the tail-refinement seal says that compatible accepted RegSeqRat refinements produce the unique terminal seal over those rows. Thus R and R' expose the same visible terminal seal boundary.

Conversely assume equality of the visible terminal seal boundary. Apply Definition 38.379. Equality of the terminal boundary reads back exactly the StreamName classifier rows, unary-history and bundle rows, pointwise rational-history rows, accepted RegSeqRat spine, dyadic endpoint rows, and visible Cont, hsame, Pkg, and RealLedgerPolicy rows.

It remains to identify the common-refinement route itself. Apply Definition 38.387. The two compatible tail-refinement seals are cofinal over the same certified package C, so the finite terminal boundary readback recovers the same common-refinement source tuple. The cited coverage and seal theorems expose no completion, quotient, host, metric, arithmetic, order, or bridge rows, so equality cannot depend on any data outside that tuple. □

Theorem 38.391 (RealUp terminal refinement-family pullback exhaustion)**.** *Let $C_1, \ldots, C_n$ be a finite family of certified* StreamName$^{\uparrow}$ *tail-window coverage packets over the same public* StreamName$^{\uparrow}$ *endpoints and the same accepted* RegSeqRat$^{\uparrow}$ *spine. Assume that each pair C_i, C_j admits the*

common-refinement seal pullback of Definition 38.390. Then every terminal $\mathsf{Real}^{\uparrow}$ *finite-window consumer reached from the family factors through one finite family pullback surface.*

The family pullback surface exposes exactly the restricted RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory *and* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *spine, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the visible* Cont, hsame, *and* Pkg *rows, and the* RealLedgerPolicy *rows. It exposes no completion object, quotient row, host index, host function, metric-completeness datum, arithmetic or order row, or standard bridge row.*

Proof. Induct on the finite family. The empty and singleton cases are the terminal finite-window surfaces supplied by Definition 38.378; there is then no nontrivial pairwise splice to compare.

For the induction step, suppose the first k packets already factor through a single displayed pullback surface and add C_{k+1}. Apply Definition 38.380 to make the route from the accumulated surface to C_{k+1} independent of the chosen member route through the public endpoints and accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine.

Next use Definition 38.386. Pairwise availability of the common-refinement seal pullback fixes the splice between the accumulated surface and C_{k+1}, so all visible classifier, history, bundle, spine, endpoint, transport, package, and ledger rows agree on the same finite surface.

Finally apply Definition 38.390 to identify that surface with the terminal consumer boundary. The cited terminality, route-determinacy, splice-uniqueness, and pullback results expose only the rows listed in the statement, so the factorization cannot read any completion, quotient, host, metric, arithmetic, order, or bridge data. □

Theorem 38.392 (RealUp stream-prefix scoped obligation package)**.** *The four stream-prefix obligation faces in Definition 38.335, Definition 38.336, Definition 38.337, and Definition 38.338 form the scoped RealUp stream-prefix package. Its source is bound to* $\mathsf{Real}^{\uparrow}$ *finite stream-prefix observations over the* RegSeqRat *spine,* $\mathsf{StreamName}^{\uparrow}$ *unary observation names,* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, histories in* BHist, hsame *transport rows, selected* Cont *continuation rows, unary observations, finite-window classifiers, dyadic endpoint ledger rows, and the displayed* $\mathsf{Real}^{\uparrow}$ *seal. The package consumes no quotient, completion object, host function, external metric primitive, or standard bridge.*

Proof. Read the four named obligation theorems field by field and compare them with the ten-field roadmap template.

1. Traditional target: the target is $\mathsf{Real}^{\uparrow}$ finite stream-prefix observations. The carrier, classifier, stability, and ledger-seal faces all refer to selected rational-observation prefixes rather than to an ambient completion object.

2. BEDC source: the source is the RegSeqRat spine over $\mathsf{StreamName}^{\uparrow}$ unary observations and $\mathsf{Rat}^{\uparrow}$ endpoint histories in BHist. Definition 38.335 reads those rows from the certified unary-observation spine.

3. Carrier choice: the carrier face is the visible endpoint carrier data. Apply Definition 38.335; it identifies the selected prefix endpoints with the RatHistoryCarrier rows supplied by the RegSeqRat spine.

4. Classifier choice: the classifier face is the pair RealStreamClassifier and RealStreamPrefixClassifier. Apply Definition 38.336; it relates global stream rows, finite-prefix classifiers, and selected $\mathsf{Rat}^{\uparrow}$ endpoint classifier rows.

5. Stability obligations: the stability face is transport along hsame, unary-context closure, and selected continuation stability. Apply Definition 38.337; it preserves the finite-prefix classifier rows and repacks them as global stream classifier rows when all finite prefixes are present.
6. Ledger: the ledger face is the finite observation ledger together with dyadic endpoint ledger rows. Apply Definition 38.338; it reads finite windows, dyadic comparisons, regularity certificates, and the $\mathsf{Real}^{\uparrow}$ seal from the same scoped stream-name interface.
7. Core theorem list: the package core is exactly the four obligation faces cited in the statement. No fifth obligation theorem is required, because the displayed carrier, classifier, stability, ledger, and seal fields exhaust the stream-prefix boundary.
8. Exactness predicate: exactness is finite-window stream-prefix readback. It is supplied by the classifier face through finite-prefix exactness and by the ledger-seal face through finite-window dyadic source exhaustion.
9. Standard bridge: no standard bridge is claimed. The theorem records a paper-side scoped package over BEDC rows only, not a bridge to classical real analysis or to an external metric construction.
10. Status: the result is the scoped paper package for the RealUp stream-prefix site. It collects the four local obligation faces into one scope without adding a Lean marker or consuming data outside the displayed BEDC source.

□

Theorem 38.393 (RealUp stream-prefix scoped consumer determinacy)**.** *Let two* $\mathsf{Real}^{\uparrow}$ *stream-prefix consumer packets be displayed over the same finite observation window. Suppose they agree on their* $\mathsf{StreamName}^{\uparrow}$ *unary observation rows, their* RegSeqRat *spine rows, their* $\mathsf{Rat}^{\uparrow}$ *endpoint classifier rows, their DyadicRatCore finite ledger rows, their displayed* hsame *transports, their selected* Cont *rows, and their displayed* $\mathsf{Real}^{\uparrow}$ *seal row. Then the two packets determine the same scoped consumer-facing* BHist *surface.*

The determined surface is exactly the stream-prefix surface generated by those rows. It contains no quotient row, no completion object, no metric-completeness datum, no host function, no Classical datum, no arithmetic or order primitive outside the listed rows, and no standard bridge row.

Proof. First bind both packets to the scoped stream-prefix package. Apply Definition 38.392. Its carrier, classifier, stability, ledger, and seal faces list precisely the $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, dyadic, finite-window, and seal rows available to a consumer.

Next read the finite observation boundary. Apply Definition 126.79 for the common StreamName and RegSeqRat handoff, and apply Definition 38.342 for the displayed RealUp window. These theorems identify the public read surface with the listed unary observation rows, spine rows, endpoint classifier rows, finite ledger rows, transports, continuations, and seal data.

Now account for tails and partitions. Apply Definition 38.395 to every selected tail window and use Definition 38.309 for any finite partition presentation. Since the two packets agree on every row named by these read surfaces, their transported endpoints, dyadic entries, ledger entries, continuation routes, and seal rows are the same displayed BHist rows.

Repacking those rows gives one consumer-facing surface. The cited package, handoff, read-surface, and tail-transport theorems expose no field for quotients, completion objects, metric-completeness witnesses, host functions, Classical data, unlisted arithmetic or order primitives, or standard bridge rows, so none of those data can distinguish the two packets. □

Theorem 38.394 (RegSeqRat StreamName common source locality)**.** *Let two scoped* $\mathsf{Real}^{\uparrow}$ *stream-prefix packets be displayed over the same finite observation window. Suppose they agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *unary observation rows, the* $\mathsf{RegSeqRat}$ *spine rows, the* $\mathsf{Rat}^{\uparrow}$ *endpoint classifier rows, the finite* $\mathsf{ProbeBundle}$ *windows, the* $\mathsf{InBundle}$ *membership rows, the displayed* hsame *transports, and the selected* Cont *routes. Then the two packets determine the same common source surface before any* $\mathsf{Real}^{\uparrow}$ *seal row, completion datum, quotient datum, host-equality row, metric-completeness witness, or standard bridge row is read.*

The common source surface is the finite BEDC surface generated by the listed StreamName, RegSeqRat, RatUp, bundle, transport, and continuation rows. Any later $\mathsf{Real}^{\uparrow}$ *seal may consume this surface, but it is not part of the locality claim.*

Proof. Bind both packets to the scoped obligation package of Definition 38.392. The package identifies the only source rows available before the seal: StreamName unary observations, RegSeqRat spine data, RatUp endpoint classifiers, BHist histories, hsame transports, selected Cont routes, and finite-window bundle membership.

Next pass through the handoff boundary. Apply Definition 126.79 to read the common StreamName–RegSeqRat window, and apply Definition 126.82 to keep the handoff source separate from any RealUp seal or completion data.

Finally apply Definition 38.342. Since the two packets agree on every row exposed by that finite read surface, the pre-seal source they present to later consumers is the same. The cited package and read-surface theorems expose no quotient row, host equality row, metric-completeness row, or standard bridge row, so none of those data can enter the local common source. □

Theorem 38.395 (RealUp scoped window tail-transport package)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, indices* n, m*, and transported endpoint histories* $u, v : \mathsf{Hist}$*, assume the Real stream-prefix classifier through* n*,*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n),$$

a classified finite $\mathsf{Rat}^{\uparrow}$ *tail of length* m*,*

$$\forall k < m,\ \mathsf{RatHistoryClassifier}(x(n+k+1), y(n+k+1)),$$

pointwise unary anchors,

$$\forall i,\ \mathsf{UnaryHistory}(x(i)),$$

and transported endpoint rows

$$\mathsf{hsame}(x(n+m), u), \quad \mathsf{hsame}(y(n+m), v).$$

Then the transported tail window remains inside the scoped $\mathsf{Real}^{\uparrow}$ *stream-prefix package. It exposes the transported endpoint* $\mathsf{Rat}^{\uparrow}$ *classifier, positive unary denominator rows for* u *and* v*, unary endpoint rows, nonempty endpoint exclusions, common e-one tail readback whenever the transported endpoints are displayed as one-headed, finite* $\mathsf{RealLedgerPolicy}$ *rows, and the dyadic endpoint source rows consumed by the* $\mathsf{Real}^{\uparrow}$ *seal. It consumes no quotient, completion object, metric-completeness certificate, host function, Classical datum, arithmetic or order primitive outside the listed rows, or standard-bridge datum.*

Proof. First package the ambient scoped source. Apply Definition 38.392. This fixes the carrier, classifier, stability, ledger, and seal faces of the $\mathsf{Real}^{\uparrow}$ stream-prefix site, and it records that those faces do not read quotient, completion, metric, host-function, or standard-bridge data.

Next transport the finite tail to the displayed endpoint boundary. Apply Definition 38.116 to extend the prefix classifier from n to $n+m$ using the classified finite $\mathsf{Rat}^{\uparrow}$ tail and the pointwise unary anchors. Then apply Definition 38.293 to the extended boundary and the two endpoint hsame rows. The result supplies the transported endpoint $\mathsf{Rat}^{\uparrow}$ classifier, the positive unary denominators, the unary endpoint rows, and the nonempty endpoint exclusions.

Now read the window face through the scoped boundary. Apply Definition 38.342. Its finite-window readback contains the endpoint classifier rows, positive unary denominator rows, common e-one tail readback, the finite ledger, and the dyadic source rows used by the $\mathsf{Real}^{\uparrow}$ seal.

The rows just obtained are all rows named by the scoped stream-prefix package, and the finite-window read surface explicitly excludes quotient, completion, metric-completeness, Classical, arithmetic, order, and standard-bridge data outside that boundary. Repacking these two scoped statements with the transported finite-tail endpoint package gives the claimed transported tail window package. □

Theorem 38.396 (RealUp partitioned window tail-transport package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, a total finite bound w, and a certified finite partition spine P over (n, w) with segments (n_i, r_i), let $u_i, v_i : \mathsf{Hist}$ be transported histories at the segment endpoints. Assume the Real stream-prefix classifier through the anchor,*

$$\mathsf{RealStreamPrefixClassifier}(x, y, n),$$

the segment classifier conjunction,

$$\bigwedge_{(n_i,r_i)\in P} \mathsf{RealStreamWindowClassifier}(x, y, n_i, r_i),$$

pointwise unary anchors,

$$\forall j,\ \mathsf{UnaryHistory}(x(j)),$$

and endpoint transport rows for every segment,

$$\forall (n_i, r_i) \in P, \mathsf{hsame}(x(n_i + r_i), u_i) \quad \textit{and} \quad \mathsf{hsame}(y(n_i + r_i), v_i).$$

Then every segment endpoint exposes the transported endpoint $\mathsf{Rat}^{\uparrow}$ classifier, positive unary denominator rows, unary endpoint rows, nonempty endpoint exclusions, common e-one tail readback whenever the transported endpoints are displayed as one-headed, finite $\mathsf{RealLedgerPolicy}$ rows, and DyadicRatCore endpoint source rows listed in Definition 38.395. Moreover, the total window presentation and the segment presentation expose the same finite ledger, DyadicRatCore source rows, $\mathsf{RegSeqRat}$ spine rows, $\mathsf{StreamName}^{\uparrow}$ request names, $\mathsf{UnaryHistory}$ witnesses, and hsame transport rows of Definition 38.309. The combined package consumes no quotient, completion object, metric-completeness certificate, Classical datum, arithmetic or order closure, or standard bridge.

Proof. First fix the ambient scoped boundary. Apply Definition 38.392; it supplies the carrier, classifier, stability, ledger, and seal rows of the $\mathsf{Real}^{\uparrow}$ stream-prefix site from the displayed $\mathsf{RegSeqRat}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, $\mathsf{UnaryHistory}$, Cont, dyadic, and hsame source rows.

Next pass between the total and segmented windows. Apply Definition 38.308 to the certified spine P. The assumed segment classifier conjunction is the segmented presentation of the total window $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$, and the same fold also recovers the segment conjunction from the total window when a consumer starts from the wide observation.

Now identify the read surface. Apply Definition 38.309. It says that the total window and the partition presentation expose exactly the same finite ledger, DyadicRatCore source rows, stream-name requests, unary-history witnesses, and hsame transport rows, with no quotient, completion, metric-completeness, Classical, arithmetic, order, or standard-bridge datum.

For each segment (n_i, r_i) in P, apply Definition 38.395 at the segment anchor with its endpoint histories u_i and v_i. The segment classifier row, the pointwise unary anchors restricted to the segment, and the two endpoint transport rows give exactly the transported endpoint classifier, denominator, unary-endpoint, exclusion, common-tail, finite-ledger, and dyadic source rows named in that theorem.

Finally repack the segmentwise transported endpoint rows with the shared partition read surface. The repacked certificate is a single finite observation package for partitioned $\mathsf{Real}^{\uparrow}$ windows, and every row it displays has already been read from the scoped BEDC boundary named above. □

Theorem 38.397 (RealUp refined partition tail-source determinacy)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, and a finite width w, let P and Q be certified finite partition spines refining the same total $\mathsf{Real}^{\uparrow}$ stream window*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w).$$

Assume that the two refined presentations have matching segment endpoint hsame *rows, matching displayed* RealStreamWindowClassifier *rows on corresponding refined segments, and the same pointwise anchors*

$$\forall j,\ \mathsf{UnaryHistory}(x(j)).$$

Then the two refined presentations expose the same finite source surface: endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, dyadic source rows, RegSeqRat *spine rows,* $\mathsf{StreamName}^{\uparrow}$ *observation names,* Cont *routes,* hsame *transports, and finite ledger rows. Thus the refined partitions determine one BEDC source ledger for the total window, rather than two presentation-dependent ledgers.*

No quotient representative, completion object, metric-completeness witness, Classical datum, host arithmetic or order closure, or standard bridge is read by this determinacy surface.

Proof. First bind both presentations to the same total window. Apply Definition 38.309 to P and again to Q. Each application identifies its refined partition ledger with the same total finite-window read surface, including the $\mathsf{StreamName}^{\uparrow}$ request rows, dyadic source rows, RegSeqRat spine rows, unary-history anchors, Cont routes, and endpoint hsame transports.

Next transport the segment endpoints. Apply Definition 38.395 on each displayed segment. The matching segment classifier rows, endpoint hsame rows, and pointwise unary anchors expose the same endpoint $\mathsf{Rat}^{\uparrow}$ classifiers, dyadic source rows, unary endpoint rows, and finite ledger entries for the two refined presentations.

Finally repack the refined segment data with Definition 38.396. Since both partition packages have already been read from the same total window surface, the repacked endpoint classifiers, dyadic rows, spine rows, observation names, continuation routes, transports, and ledger rows are the same finite BEDC source surface.

The cited packages also carry the exclusion clauses. They have no fields for quotient representatives, completion objects, metric-completeness witnesses, Classical data, host arithmetic or order closure, or standard bridges, so none of those can distinguish the two refined presentations. □

Theorem 38.398 (RealUp finite-window observation determinacy)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, an anchor n, and a finite width w, suppose two BEDC finite-window packages are*

exposed from the same classifier

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w).$$

Assume that both packages read the same $\mathsf{StreamName}^{\uparrow}$ *request rows,* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, DyadicRatCore source rows,* UnaryHistory *witnesses,* Cont *ledger entries, and* hsame *transports named by Definition 38.309. Then their visible endpoint classifiers, positive denominator rows, finite ledger rows, dyadic source rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows classify together by* hsame*. In particular, no second public finite-window observation package with a different visible endpoint surface is available from the same request.*

Proof. First bind the common source boundary. Apply Definition 38.392. This fixes the allowed RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, UnaryHistory, Cont, dyadic, and hsame rows, and it excludes quotient, completion, host-function, external metric, and standard-bridge inputs.

Next identify the window ledger seen by the two packages. Apply Definition 38.309. The total finite window and any certified partition of it expose the same finite ledger, DyadicRatCore source rows, stream-name requests, unary-history witnesses, and endpoint hsame transports. Since the two packages are read from the same classifier and the same request rows, their public ledgers are the same BEDC read surface.

For each displayed segment of a partition, apply Definition 38.395. It reads the endpoint classifiers, positive unary denominator rows, unary endpoint rows, finite ledger entries, and dyadic endpoint source rows from the segment classifier and the endpoint transport rows. The hypotheses identify those rows for the two packages, so the segment endpoints classify together by hsame.

Finally repack the segmentwise hsame endpoint data through Definition 38.396. The repacked finite-window package has only the rows already named by the scoped source and partition read surface. The same exclusion clauses remove quotient, completion, metric-completeness, host-function, Classical, arithmetic or order primitive, and standard-bridge data, so there is no remaining row from which a different public endpoint package could be exposed. □

Theorem 38.399 (RealUp finite-window common-refinement determinacy)**.** *For rational-observation streams* $x, y : \mathbb{N} \to \mathsf{Hist}$*, an anchor* n*, and a finite width* w*, suppose two certified finite-window packages expose read surfaces for the same classifier*

$$\mathsf{RealStreamWindowClassifier}(x, y, n, w).$$

Assume that the two packages admit a common certified partition refinement of that classifier, and that both refined packages read the same $\mathsf{StreamName}^{\uparrow}$ *request rows,* RegSeqRat *spine rows, DyadicRatCore endpoint rows,* UnaryHistory *witnesses,* Cont *ledger entries, and* hsame *transports.*

Then the two refined read surfaces expose classifier-equivalent endpoint rows, positive denominator rows, dyadic source rows, and $\mathsf{Real}^{\uparrow}$ *seal rows. Equivalently, refining the finite window through a shared certified partition does not create a second public endpoint surface for the same observation request.*

Proof. Bind the common scoped source package with Definition 38.392. This fixes the visible RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, UnaryHistory, Cont, dyadic, and hsame rows available to both finite-window packages.

Use Definition 38.309 on the common certified refinement. The theorem identifies the total classifier with its partition read surface, so both packages are read through the same finite ledger, request rows, unary witnesses, endpoint transports, and dyadic source rows.

On each refined segment, apply Definition 38.395. The shared segment classifier and endpoint transports expose the same endpoint classifier rows, positive denominator rows, common-tail rows, finite-ledger rows, and dyadic source rows for the two packages.

Finally repack the segmentwise hsame rows through Definition 38.396. The repacked public finite-window surfaces are built from the same partition ledger and the same transported endpoint data, hence their endpoint, denominator, dyadic source, and $\mathsf{Real}^{\uparrow}$ seal rows classify together by hsame. □

Theorem 38.400 (StreamName RegSeqRat RealUp common handoff cover)**.** *For two public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packages with a certified common refinement, let a* RegSeqRat *handoff package and a finite-window* $\mathsf{Real}^{\uparrow}$ *read surface be induced from the same displayed restrictions. Then all consumer-facing rows factor through one common source package consisting exactly of the selected* RatStreamNameCarrier *rows, the selected* RatStreamNameClassifier *rows,* ProbeBundle*,* InBundle*,* UnaryHistory*, the displayed* hsame *rows, the dyadic ledger rows consumed by the* RegSeqRat *seal, and the finite* $\mathsf{Real}^{\uparrow}$ *ledger boundary.*

The cover contains no quotient data, no completion data, no host index equality, no metric-completeness data, and no standard-bridge data. Thus the StreamName source, the RegSeqRat *handoff, and the finite-window* $\mathsf{Real}^{\uparrow}$ *read surface have a single common BEDC handoff boundary.*

Proof. Start with the two public $\mathsf{StreamName}^{\uparrow}$ finite-window packages. Apply Definition 126.88. It identifies the selected RatStreamNameCarrier and RatStreamNameClassifier rows as the unique stream-name source for the overlapping RegSeqRat handoff surface.

Next apply Definition 126.100. Every RegSeqRat handoff row consumed downstream is exhausted by the selected stream-name rows, ProbeBundle, InBundle, UnaryHistory, the displayed hsame transports, and the dyadic ledger rows used by the seal.

For the finite-window $\mathsf{Real}^{\uparrow}$ side, use Definition 38.314. It reads the public finite-window ledger only from the same observation restrictions, endpoint rows, unary-history witnesses, dyadic rows, and displayed transports.

Finally use Definition 38.399. The certified common refinement prevents the two finite-window packages from exposing a second endpoint, denominator, dyadic source, or $\mathsf{Real}^{\uparrow}$ seal row. Repacking the StreamName source, the exhausted RegSeqRat handoff rows, and the finite $\mathsf{Real}^{\uparrow}$ ledger boundary gives one common source package. Since none of the cited surfaces exports quotient representatives, completion objects, host index equality, metric-completeness witnesses, or standard-bridge rows, the common cover excludes exactly those data. □

Theorem 38.401 (RealUp StreamName regular observation envelope)**.** *Any selected finite-window* $\mathsf{Real}^{\uparrow}$ *certificate boundary read through the StreamName–RegSeqRat common handoff factors through one* BHist*-anchored observation envelope. The envelope consists exactly of finite-window StreamName classifier rows, RegSeqRat pointwise seal rows, selected denominator endpoint rows, and bundle-membership rows. It contains no full* $\mathsf{Real}^{\uparrow}$ *completeness row, no trichotomy row, no quotient-class row, and no open-cover compactness row.*

Proof. Begin with the common handoff cover of Definition 38.400. It binds the selected StreamName rows, ProbeBundle, InBundle, UnaryHistory, displayed hsame transports, RegSeqRat dyadic ledger rows, and finite $\mathsf{Real}^{\uparrow}$ ledger boundary into one BEDC source package.

Apply Definition 38.402. The selected certificate boundary can read only certified StreamName observations, RegSeqRat dyadic source exhaustion, the finite-window ledger boundary, certified subwindow closure, and common-refinement determinacy.

Now use the pointwise StreamName handoff: the finite classifier rows are those of Definition 126.170, their restriction is stable by Definition 126.171, and the singleton reads are the diagonal packets of Definition 126.173. Thus every selected endpoint and denominator row is anchored in the same BHist observation envelope.

The cited rows exhaust the available finite evidence. Full completeness, trichotomy, quotient classes, and open-cover compactness are not fields of the StreamName handoff, the RegSeqRat pointwise packet, or the selected finite-window $\mathsf{Real}^{\uparrow}$ boundary, so none is imported by the factorization. □

Theorem 38.402 (RealUp selected finite-window certificate boundary)**.** *A selected* $\mathsf{Real}^{\uparrow}$ *finite-window certificate for* $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$ *consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *certified observation rows, the* RegSeqRat *dyadic source exhaustion rows, the* $\mathsf{Real}^{\uparrow}$ *finite-window ledger boundary, the certified subwindow-closure rows, and the common-refinement determinacy row of Definition 38.399.*

Equivalently, the selected certificate boundary is the finite BEDC surface generated by those rows and no others. It excludes host Nat *arithmetic data, quotient representatives, real-completeness data, metric-completeness witnesses, external Cauchy structures, and standard-bridge rows.*

Proof. First bind the finite source surface. Apply Definition 38.314. This gives the finite observation requests, RegSeqRat dyadic source rows, $\mathsf{StreamName}^{\uparrow}$ certified rows, unary-history witnesses, and endpoint transport rows available to the selected window.

Next project the window ledger. The ledger boundary is exactly Definition 38.320: it records the finite stream-name observation rows and the $\mathsf{Real}^{\uparrow}$ endpoint rows read from the displayed classifier, without adding host arithmetic or quotient representatives.

The subwindow component is supplied by Definition 38.325. It says that certified subwindows remain inside the same finite exhaustion surface, so the selected certificate cannot acquire a new source row by cutting the window.

Finally attach the common-refinement row from Definition 38.399. Any two certified refinements of the selected window read classifier-equivalent endpoint, denominator, dyadic source, and seal rows. Repacking these four components gives exactly the selected certificate boundary, and the exclusion clauses follow because none of the cited rows exports host Nat data, quotient representatives, completeness data, metric witnesses, or standard-bridge structure. □

Theorem 38.403 (RealUp selected finite-window consumer stability)**.** *Let a selected* $\mathsf{RealStreamWindowClassifier}(x$ *certificate boundary be fixed by Definition 38.402. Any* RegSeqRat *handoff consumer and any finite-window* $\mathsf{Real}^{\uparrow}$ *consumer that read the same selected* $\mathsf{StreamName}^{\uparrow}$ *rows,* RegSeqRat *dyadic rows, endpoint* $\mathsf{Rat}^{\uparrow}$ *rows, unary witnesses,* Cont *ledger entries, and* hsame *transports must expose classifier-equivalent endpoint rows, positive denominator rows, dyadic source rows, finite ledger rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows.*

The stability claim is confined to this selected finite BEDC surface. It does not identify host indices, does not pass to quotient representatives or completion objects, does not use metric-completeness data, does not import Classical data, and does not add standard-bridge rows.

Proof. First use Definition 38.402. It reduces both downstream reads to the same selected finite boundary: certified $\mathsf{StreamName}^{\uparrow}$ observations, RegSeqRat dyadic source exhaustion, endpoint $\mathsf{Rat}^{\uparrow}$ rows, unary witnesses, finite Cont entries, and carried hsame transports.

Next apply Definition 126.82. The RegSeqRat handoff consumer can read the stream-name source and dyadic obligation rows only through the separated BEDC handoff surface, so its visible dyadic and stream rows remain tied to the selected classifier.

Use Definition 38.216 for the carrier export. It identifies the positive denominator rows and dyadic source rows available to the handoff consumer with the classifier rows carried by the same finite package.

Finally compare the finite-window $\mathsf{Real}^{\uparrow}$ consumer by Definition 38.399. Any certified refinement reading the selected rows exposes classifier-equivalent endpoint, denominator, dyadic source, finite-ledger, and seal rows. Since both consumers are projections of the selected boundary, the resulting public surfaces are equivalent by the displayed hsame transports, with no quotient, completion, metric-completeness, host-index, Classical, or standard-bridge row available for alteration. □

Theorem 38.404 (RealUp selected finite-window common boundary exactness)**.** *For a selected* $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$ *boundary, let a* RegSeqRat *handoff consumer and a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer read the same* $\mathsf{StreamName}^{\uparrow}$ *rows,* RegSeqRat *dyadic rows,* DyadicRatCore *ledger entries, endpoint* $\mathsf{Rat}^{\uparrow}$ *rows,* UnaryHistory *witnesses,* Cont *ledger entries, and* hsame *transports. Then the two consumers expose exactly one common BEDC source boundary: coverage is the common handoff cover of Definition 38.400, soundness is the selected certificate boundary of Definition 38.402, and completeness is the consumer stability of Definition 38.403.*

The exact boundary consists of the selected StreamName rows, the dyadic source rows, the DyadicRatCore ledger entries, the RatUp endpoint rows, the unary witnesses, the finite Cont *entries, and the displayed transports. It excludes host* Nat *indices, quotient representatives,* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, Classical data, and standard-bridge rows.*

Proof. First take the common handoff cover of Definition 38.400. It supplies a single source package for the selected StreamName rows, the RegSeqRat dyadic source rows, the displayed unary witnesses, the carried transports, and the finite RealUp ledger boundary.

Next apply Definition 38.402. This identifies every selected certificate row as generated by that finite source surface together with its DyadicRatCore ledger entries, endpoint RatUp rows, finite Cont entries, subwindow closure rows, and common-refinement determinacy. Hence no sound consumer can expose a row outside the displayed boundary.

It remains to identify the common exposure of the two consumers. Use Definition 38.403: any RegSeqRat handoff consumer and finite-window RealUp consumer reading the same displayed rows expose classifier-equivalent endpoint, denominator, dyadic source, finite-ledger, and seal rows. Thus each row covered by the source package is read back by both consumers, and any row exposed by either consumer is already inside the selected certificate boundary.

The exclusion clauses follow from the three cited surfaces. None exports host Nat equality, quotient representatives, RealUp completeness, metric completeness, Classical data, or standard-bridge rows, so the exact common boundary is the whole available BEDC source surface and nothing beyond it. □

Theorem 38.405 (RealUp selected finite-window common boundary witness)**.** *Every downstream selected-window consumer of the* $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$ *boundary factors through one explicit BEDC source package. That package consists exactly of the selected* $\mathsf{StreamName}^{\uparrow}$ *rows, the* RegSeqRat *dyadic rows, the* DyadicRatCore *ledger entries, the* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, the* UnaryHistory *witnesses, the finite* Cont *entries, and the* hsame *transports of Definition 38.404.*

If the consumer is restricted to a certified subwindow or replaced by a certified refinement, the same package remains the common source boundary. The restriction or refinement may discard unread rows, but it cannot introduce a new host index, quotient representative, completion row, metric row, Classical row, or standard-bridge row outside the displayed BEDC source package.

Proof. First use Definition 38.400. It supplies the shared cover by StreamName rows, RegSeqRat dyadic rows, unary witnesses, carried transports, and the finite RealUp ledger boundary.

Next apply Definition 38.402. This identifies the selected certificate as soundly generated from the same finite source surface together with the DyadicRatCore ledger entries, RatUp endpoints, finite Cont entries, and displayed subwindow rows.

For downstream readback, use Definition 38.403. Any selected-window consumer reading the displayed rows exposes only classifier-equivalent endpoint, denominator, dyadic source, finite-ledger, and seal rows, so it factors through the same source package.

Finally apply Definition 38.399. Certified subwindows and refinements keep the endpoint, denominator, dyadic source, ledger, and seal rows within the same finite boundary. Repacking these four steps gives a named witness for the common finite source package and leaves no extra quotient, completion, metric-completeness, Classical, host-index, or standard-bridge row available to any selected-window consumer. □

Theorem 38.406 (RealUp selected StreamName-RegSeqRat-DyadicRatCore boundary)**.** *For any selected* $\mathsf{RealStreamWindowClassifier}(x, y, n, w)$ *boundary, every downstream selected-window* $\mathsf{Real}^{\uparrow}$ *consumer factors through exactly the finite BEDC boundary named in Definition 38.404: the shared* $\mathsf{StreamName}^{\uparrow}$ *rows, the* RegSeqRat *dyadic rows, the* DyadicRatCore *ledger entries, the* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, the* UnaryHistory *witnesses, the finite* Cont *entries, and the* hsame *transports.*

Conversely, those displayed rows repack into both the selected-window certificate boundary of Definition 38.402 and the common boundary witness of Definition 38.405. No part of this boundary contains host Nat *indices, quotient representatives,* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, Classical data, or standard-bridge rows.*

Proof. Begin with the common handoff cover of Definition 38.400. It gives the selected StreamName rows, RegSeqRat dyadic source rows, unary witnesses, carried transports, and finite RealUp ledger surface read by the selected boundary.

Soundness is the selected-window certificate boundary of Definition 38.402. It places the DyadicRatCore ledger entries, RatUp endpoint rows, finite Cont entries, and subwindow rows inside the same finite certificate surface, so no sound selected-window consumer can read outside the displayed BEDC rows.

For downstream readback, apply Definition 38.403. Consumers reading the displayed rows expose only classifier-equivalent endpoint, denominator, dyadic source, finite-ledger, and seal rows, hence their observable boundary factors through the common surface already named by Definition 38.404.

The reverse packing is exactly the witness supplied by Definition 38.405: the same StreamName, RegSeqRat, DyadicRatCore, RatUp, UnaryHistory, Cont, and hsame rows form one selected-window source package. The exclusion clauses are the common exclusions of the cited cover, certificate boundary, stability theorem, and witness, which export no host Nat indices, quotient representatives, completion or metric-completeness rows, Classical data, or standard-bridge rows. □

Theorem 38.407 (RealUp RegSeqRat StreamName seal handoff)**.** *The rows of Definition 38.276, together with the finite-window seal and endpoint readback rows displayed in the* $\mathsf{Real}^{\uparrow}$ *stream-prefix certificate boundary, determine exactly the* $\mathsf{Real}^{\uparrow}$ *seal handoff package. Its exported rows are the* RatStreamNameClassifier *row, the* RealUnaryStreamClassifier *transport row, the certified unary endpoint carriers, the finite-window ledger, and the seal/readback provenance. This handoff is the concrete route through chapter 126, chapter 301, and chapter 291 before the local* $\mathsf{Real}^{\uparrow}$ *seal is read.*

No quotient carrier, host equality, real completeness, arithmetic package, order package, or metric-completeness row is exported by this handoff.

Proof. First project the scoped stream-name interface. Definition 38.276 supplies the RatStreamNameClassifier row, the transported RealUnaryStreamClassifier row, and the certified unary endpoint carrier rows used by the $\mathsf{Real}^{\uparrow}$ stream-prefix certificate. These rows are exactly the sibling chapter surfaces named by chapter 126 and chapter 301, not additional analytic structure.

Next read the finite-window seal boundary. Definition 38.277 and Definition 38.278 identify the finite window, the pointwise dyadic observation rows, and the RegSeqRat spine ledger consumed by the public $\mathsf{Real}^{\uparrow}$ seal. These rows are exactly the sibling chapter surfaces named by chapter 291 and chapter 301, not additional analytic structure.

Finally use the endpoint readback and hidden-data ledger rows of the stream-prefix certificate boundary. They repack the displayed classifier, transport, endpoint, finite-window, and provenance rows as the stated seal handoff package, while excluding every off-table claim: quotient carriers, host equality, completeness, arithmetic, order, and metric-completeness data. □

Theorem 38.408 (RealUp open phase seal exactness)**.** *A finite* $\mathsf{Real}^{\uparrow}$ *seal handoff is accepted precisely when its displayed rows factor through one* BHist *packet containing the four open-phase faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows for the finite unary window;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier and transported stream name rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine ledger over the same scheduled observations;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal, endpoint readback, and provenance ledger.*

The factorization uses the displayed hsame, Cont, Pkg, *and* NameCert *rows, and it exports no quotient carrier, host equality, real completeness, arithmetic package, order package, or metric-completeness row.*

Proof. For the forward direction, begin with an accepted finite $\mathsf{Real}^{\uparrow}$ seal handoff. Theorem 38.407 projects the seal handoff into its stream-name classifier row, transported RealUp unary classifier row, certified unary endpoint carriers, finite-window ledger, and seal/readback provenance.

Now separate the four open-phase faces. Theorem 38.276 supplies the $\mathsf{StreamName}^{\uparrow}$ classifier, transported stream-name rows, and the regular-sequence interface rows. Theorems 38.277 and 38.278 identify the same finite window as the place where the $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the $\mathsf{RegSeqRat}^{\uparrow}$ spine ledger, and the $\mathsf{Real}^{\uparrow}$ seal/readback ledger are read. Thus every accepted handoff factors through one displayed BHist packet with the stated hsame, Cont, Pkg, and NameCert rows.

For the reverse direction, start from such a shared BHist packet. The scoped stream-name interface repacks the $\mathsf{StreamName}^{\uparrow}$ classifier and regularity spine rows, while the finite-window seal coverage theorem attaches the dyadic observation rows to that same window.

Finally apply Definition 38.278 and then Definition 38.407. The first theorem turns the shared finite-window packet into the $\mathsf{Real}^{\uparrow}$ seal-facing package, and the second identifies that package as the finite seal handoff. Since every row used in both directions is one of the displayed four faces, no quotient carrier, host equality, completeness, arithmetic, order, or metric-completeness datum is introduced. □

Theorem 38.409 (RealUp open-phase four-face consumer totality)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *completion-facing, or stream-facing consumer of an accepted finite* $\mathsf{Real}^{\uparrow}$ *seal source factors through exactly the four displayed open-phase faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows over the finite unary window;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows and transported stream-name rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows over the same scheduled observations;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal, endpoint readback, and provenance rows.*

The only structural rows available to the consumer are the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The consumer route has no coordinate for quotient carriers, host equality, ambient completeness, arithmetic packages, order packages, or metric-completeness rows.*

Proof. Project the accepted finite seal source through Definition 38.408. Exactness reads the seal source as one BHist packet whose projections are precisely the dyadic observation face, the stream-name finite-window face, the regular-rational spine face, and the $\mathsf{Real}^{\uparrow}$ seal/readback provenance face.

Now use Definition 38.407. The handoff identifies the seal-facing rows consumed at the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ boundary: classifier rows, transported stream rows, endpoint carriers, finite-window ledger, and provenance. These rows are already contained in the four projected faces.

For the stream-facing side, cite the scoped $\mathsf{StreamName}^{\uparrow}$ finite-window coverage used in the surrounding handoff proof. For the completion-facing side, cite the finite-window $\mathsf{RegSeqRat}^{\uparrow}$ seal rows that place every regularity read on the same scheduled dyadic observations. Both consumers therefore read only rows already present in the same four-face packet.

It remains to exclude off-table data. Coordinate exhaustion leaves only the displayed BHist, hsame, Cont, Pkg, and NameCert structural rows after the four faces are projected. Since none of those rows is a quotient carrier, host equality, ambient completeness datum, arithmetic or order package, or metric-completeness row, no downstream consumer can factor through a fifth face. □

Theorem 38.410 (RealUp finite seal source uniqueness)**.** *Two finite* $\mathsf{Real}^{\uparrow}$ *seal handoffs over the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine ledger, and the same displayed* $\mathsf{Real}^{\uparrow}$ *seal endpoint and readback provenance expose the same finite* BHist *seal source surface when their* hsame*,* Cont*,* Pkg*, and* NameCert *rows agree componentwise.*

The shared source surface is the finite packet carrying the public $\mathsf{StreamName}^{\uparrow}$ *window classifier, the pointwise* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *spine rows, and the displayed* $\mathsf{Real}^{\uparrow}$ *seal and readback ledger. No quotient carrier, host equality, completeness row, arithmetic package, order package, or metric-completeness datum can distinguish the two reads.*

Proof. Project both handoffs through Definition 38.407. The projected rows are the same $\mathsf{StreamName}^{\uparrow}$ classifier, transported unary classifier, endpoint carrier rows, finite-window ledger, and seal/readback provenance because the displayed hsame, Cont, Pkg, and NameCert rows agree componentwise.

Apply Definition 38.408 to each projected handoff. Each accepted read factors through one finite BHist packet with the four open-phase faces: dyadic observations, stream-name classifier rows, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity spine ledger, and the $\mathsf{Real}^{\uparrow}$ seal endpoint/readback provenance.

Use Definition 38.277 and Definition 38.278 on the shared finite window. These theorems identify the dyadic observation rows and the regular-rational spine rows as the same finite seal-facing packet for both reads.

Thus the two handoffs expose one finite BHist seal source surface. Every coordinate available to the seal has already been matched componentwise, and the off-table quotient, host-equality, completeness, arithmetic, order, and metric-completeness data provide no further distinguishing row. □

Theorem 38.411 (RealUp seal source factorization boundary)**.** *Any finite* $\mathsf{Real}^{\uparrow}$ *seal source consumed by downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers factors through the unique four-face* BHist *packet consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, and the* $\mathsf{Real}^{\uparrow}$ *seal and readback provenance rows. Componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *agreement on these rows is sufficient to identify the visible seal source for every such consumer.*

The boundary exports no quotient carrier, host equality, completeness datum, arithmetic package, order package, or metric-completeness row.

Proof. First apply Definition 38.408. It factors an accepted finite seal handoff through the four open-phase faces: dyadic observations, stream-name classifier rows, the regular-rational spine, and the seal/readback provenance.

Next use Definition 38.410. Componentwise agreement of the displayed hsame, Cont, Pkg, and NameCert rows identifies the finite BHist seal source surface for any two reads with the same four faces.

Finally cite Definition 38.407. The handoff theorem says the finite seal-facing rows are exactly the classifier, transport, endpoint, finite-window, and provenance rows consumed by the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ boundary, so quotient carriers, host equality, completeness, arithmetic or order packages, and metric-completeness rows remain outside the consumer-visible source. □

Theorem 38.412 (Real RegSeqRat StreamName seal handoff composition)**.** *An accepted finite* $\mathsf{Real}^{\uparrow}$ *seal read is equivalent, as displayed BEDC data, to the composed L10 open-phase route whose rows are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback ledger. The equivalence is transported only by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

No quotient stream, host equality, selected limit, ambient completeness, arithmetic or order package, metric-completeness row, or hidden seal coordinate is part of the composed route.

Proof. First apply Definition 38.408. It identifies the accepted seal read with one finite BHist packet whose four faces are the dyadic observation face, the stream-name finite-window face, the RegSeqRat regularity face, and the Real seal/readback face.

Next apply Definition 38.409. This shows that every downstream consumer reads only those four faces through the displayed hsame, Cont, Pkg, and NameCert coordinates.

Use Definition 38.411 to identify the consumer-visible source with the same finite seal surface. Then use Definition 301.71 to replace the RegSeqRat-facing seal source by the StreamName, DyadicRatCore, RegSeqRat, and RealUp source-seal route.

The cited theorems expose exactly the rows listed in the statement. Since their exclusion clauses remove quotient streams, host equality, selected limits, ambient completeness, arithmetic and order packages, metric-completeness data, and hidden seal coordinates, the composition has no further row. □

Theorem 38.413 (RealUp open-phase four-face pullback)**.** *Let S and T be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal-source handoffs. They classify as the same consumer-visible handoff precisely when their four open-phase faces classify componentwise: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows classify together, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows classify together, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows classify together, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows classify together. The comparison uses the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates as its only gluing data.*

Equivalently, compatible rows on those four faces determine a unique finite BHist *pullback packet, and any accepted finite seal-source handoff projects to exactly that four-face packet. No*

quotient carrier, host equality, real completeness, arithmetic package, order package, or metric-completeness row is part of the classifier, the pullback packet, or any of its four projections.

Lean4 ✓ `BEDC.Derived.RealUp.RealupOpenPhaseFourFacePullback`

Proof. For the forward direction, start from two accepted finite seal-source handoffs that classify as the same consumer-visible handoff. Apply Definition 38.411 to each handoff. The factorization exposes exactly the dyadic observation face, the stream-name finite-window classifier face, the regularity-spine face, and the $\mathsf{Real}^{\uparrow}$ seal/readback provenance face.

The consumer-visible classification is carried only by the displayed hsame, Cont, Pkg, and NameCert rows. Projecting along the four factorization faces therefore gives componentwise classification of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

For the reverse direction, assume the four projected faces classify componentwise through the displayed gluing rows. Apply Definition 38.408 to read the compatible rows as accepted finite seal-facing packages over the same BHist source shape.

Next apply Definition 38.410. Since the four faces agree componentwise through the displayed hsame, Cont, Pkg, and NameCert rows, the two handoffs expose the same finite seal-source surface. Thus they classify as the same consumer-visible handoff, and the compatible rows determine the unique pullback packet.

The same boundary theorem, together with Definition 38.408, excludes every off-table coordinate. Thus quotient carriers, host equality, real completeness, arithmetic and order packages, and metric-completeness rows cannot appear in the pullback source or in its projected faces. □

Theorem 38.414 (RealUp common-refinement seal consumption)**.** *Let S and T be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal-source handoffs, possibly displayed over different* $\mathsf{StreamName}^{\uparrow}$ *finite windows. Suppose* $\mathsf{StreamName}^{\uparrow}$ *supplies a common finite refinement window for those two displays, and suppose the two handoffs transport to that refinement with the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, the same* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows, and the same displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates. Then every consumer-visible* $\mathsf{Real}^{\uparrow}$ *seal read from either handoff factors through one refined four-face* BHist *packet.*

The refined consumption route is finite and open-phase. It exports no quotient carrier, host equality, selected limit, real-completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness coordinate.

Proof. First transport both accepted handoffs to the supplied $\mathsf{StreamName}^{\uparrow}$ common finite refinement. Apply Definition 38.413 to the transported displays. The matching dyadic observation rows, regularity spine rows, seal/readback provenance rows, and displayed hsame, Cont, Pkg, and NameCert coordinates determine one refined four-face BHist packet.

Next use Definition 38.410. After transport to the common refinement, both accepted handoffs expose the same finite seal-source surface, because their four projected faces agree componentwise through the displayed gluing rows.

Finally apply Definition 38.419 and Definition 38.407. Route totality says every consumer-visible $\mathsf{Real}^{\uparrow}$ seal read is exhausted by an accepted open-phase seal route, and the handoff theorem says those routes are exactly the dyadic observation, finite-window, regularity-spine, and provenance rows read from the refined packet. Since the boundary has no carrier for the off-table coordinates, no consumer read can leave that refined finite packet. □

Theorem 38.415 (RealUp four-face common-window seal uniqueness)**.** *Let S and T be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal-source handoffs. Suppose both handoffs classify componentwise on the same displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, with the same transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows,*

the same $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, the same* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows, and the same displayed* hsame, Cont, Pkg, *and* NameCert *coordinates. Then every consumer-visible* $\mathsf{Real}^{\uparrow}$ *seal read from* S *or* T *factors through the same finite* BHist *route.*

This common route is a finite seal-source route only. It includes no quotient carrier, host equality, selected limit, real-completeness row, arithmetic package, order package, or metric-completeness row.

Proof. Apply Definition 38.413. The shared $\mathsf{StreamName}^{\uparrow}$ finite window and the matching dyadic, regularity-spine, and seal/readback rows identify one common four-face packet through the displayed hsame, Cont, Pkg, and NameCert coordinates.

Use Definition 38.410. Since the two accepted handoffs expose the same four-face packet, their finite seal-source surfaces are the same finite BHist source for consumer-visible reads.

Next apply Definition 38.419. Every consumer-visible $\mathsf{Real}^{\uparrow}$ seal read is exhausted by one of the accepted finite seal-source routes, so no read can bypass the common finite source just identified.

Finally use Definition 38.407. The handoff boundary says that the shared route is read only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ spine and $\mathsf{StreamName}^{\uparrow}$ finite-window rows, together with the dyadic observation and $\mathsf{Real}^{\uparrow}$ provenance faces. Thus quotient carriers, host equality, selected limits, real completeness, arithmetic or order packages, and metric-completeness rows are not part of the consumed route. □

Theorem 38.416 (RealUp RegSeqRat seal readback determinacy)**.** *Let* S *and* T *be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal packets. Suppose they display the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regular stream carrier, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, and the same* $\mathsf{Real}^{\uparrow}$ *seal/readback row. Then every consumer-visible readback route out of* S *or* T *factors through the same finite dyadic observation window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier witness, and the same* $\mathsf{Real}^{\uparrow}$ *seal endpoint.*

The common readback boundary is the four-face packet whose rows are the $\mathsf{DyadicRatCore}^{\uparrow}$ *observations on the displayed window, the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows. It exports no host equality, quotient class, completeness row, arithmetic package, or order package.*

Proof. Apply Definition 38.407 to both accepted packets. The handoff theorem reads their visible data as one package of stream-name classifier rows, transported unary classifier rows, endpoint carriers, finite-window ledger, and seal/readback provenance.

Next use Definition 38.408. Since the accepted packets have the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ carrier, the same $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, and the same $\mathsf{Real}^{\uparrow}$ seal/readback row, exactness restricts both reads to the same four open-phase row set.

Apply Definition 38.410 to identify the consumer-visible source surface. The shared source contains precisely the finite dyadic observation window, the regular-rational classifier witness, and the seal endpoint named by the displayed row.

Now route downstream reads through Definition 38.411. Every consumer of the seal source sees only the displayed schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ spine, and the $\mathsf{Real}^{\uparrow}$ seal provenance, so no off-table host or completeness coordinate can separate the two packets.

Finally apply Definition 38.413. The two readback routes have the same componentwise four-face boundary, hence the same consumer-visible readback boundary, while quotient classes, host equality, arithmetic packages, order packages, and completeness rows remain absent. □

Theorem 38.417 (RealUp RegSeqRat StreamName common-window seal fusion)**.** *Let* S *and* T *be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal handoffs. Suppose they share one displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same componentwise* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows, the same transported* $\mathsf{DyadicRatCore}^{\uparrow}$

observation rows, the same $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows, and the same displayed* hsame, Cont, Pkg, *and* NameCert *coordinates. Then their consumer-visible* $\mathsf{Real}^{\uparrow}$ *seal reads fuse through one finite* BHist *route whose visible boundary consists exactly of those four faces.*

The fused route exports no quotient carrier, host equality, selected limit, real-completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness coordinate.

Proof. Project both accepted handoffs through Definition 38.407. This reads the shared window as the same stream-name schedule together with the transported dyadic observations, the regular-rational spine, and the $\mathsf{Real}^{\uparrow}$ seal/readback provenance.

Next use Definition 38.408. Exactness exposes the accepted handoffs as four open-phase faces over the displayed hsame, Cont, Pkg, and NameCert coordinates, so no consumer-visible seal read has a fifth source row.

Apply Definition 38.413 to the shared finite window and the matching component rows. The pullback identifies one finite BHist route carrying exactly the dyadic observation face, the stream-name finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity face, and the $\mathsf{Real}^{\uparrow}$ seal/readback provenance face.

Finally use Definition 38.410 and Definition 38.416. Source uniqueness identifies the two accepted seal sources, and readback determinacy forces the consumer-visible reads to use the same finite route. Since each cited boundary has no carrier for quotient representatives, host equality, selected limits, real completeness, arithmetic or order packages, Classical choice, or metric completeness, the fused route exports none of those coordinates. □

Theorem 38.418 (RealUp StreamName factorization pullback correspondence)**.** *Let S be an accepted finite* $\mathsf{Real}^{\uparrow}$ *seal handoff. The* $\mathsf{StreamName}^{\uparrow}$ *real-seal factorization boundary holds for S precisely when there exists a* $\mathsf{Real}^{\uparrow}$ *open-phase four-face pullback packet whose projections match the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, and the same* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows.*

The correspondence uses only the displayed hsame, Cont, Pkg, *and* NameCert *rows of the accepted handoff. It introduces no quotient carrier, host equality, real-completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness coordinate.*

Proof. Assume first that the $\mathsf{StreamName}^{\uparrow}$ real-seal factorization boundary holds for S. Apply Definition 126.142 to expose the finite-window stream classifier rows and the real-seal source rows named by the handoff.

Next apply Definition 38.408. Exactness reads the accepted seal rows as the visible $\mathsf{Real}^{\uparrow}$ seal/readback provenance surface over the same finite BHist source.

Now use Definition 38.413. The dyadic observation rows, stream-name classifier rows, regularity-spine rows, and seal/readback provenance rows form the four matching faces, so they determine the required pullback packet.

Conversely, suppose the four-face pullback packet exists with the displayed matching rows. Project its $\mathsf{StreamName}^{\uparrow}$ face, its $\mathsf{RegSeqRat}^{\uparrow}$ face, and its $\mathsf{Real}^{\uparrow}$ seal/readback face through the same hsame, Cont, Pkg, and NameCert coordinates.

Apply Definition 38.411. The projected faces recover exactly the seal-source factorization rows consumed by the $\mathsf{StreamName}^{\uparrow}$ real-seal boundary, and no off-table coordinate can be inserted because neither theorem has a carrier for quotient representatives, host equality, real completeness, arithmetic or order packages, Classical choice, or metric completeness. □

Theorem 38.419 (RealUp open-phase seal route totality)**.** *Every finite* $\mathsf{Real}^{\uparrow}$ *seal consumer route accepted by the open-phase handoff is exhausted by the same four-face packet named in Definition 38.408: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback row. Its only gluing data are the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Thus a downstream consumer replay cannot branch through any fifth visible surface. Quotient representatives, host equality, real completeness, arithmetic packages, order packages, and metric-completeness rows are outside the route.

Proof. Start with an accepted finite seal consumer route. Apply Definition 38.408 to factor its accepted handoff through the four open-phase faces and the displayed transport, continuation, package, and certificate rows.

Next use Definition 38.410. Any two reads with the same four faces and componentwise hsame, Cont, Pkg, and NameCert rows expose the same finite BHist seal source surface.

Finally apply Definition 38.411. Every downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer route out of that seal source is already a route through the same four faces. Since the boundary has no carrier for the off-table rows, the finite packet exhausts the replay. □

Theorem 38.420 (RealUp four-face consumer replay)**.** *Let S and T be accepted finite* $\mathsf{Real}^{\uparrow}$ *seal packages. Suppose they display the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine, the same* $\mathsf{Real}^{\uparrow}$ *seal/readback row, and the same componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then every consumer-visible replay route from S is the same visible route as the corresponding replay route from T.*

The replay comparison is finite: it is made entirely inside the four-face BHist *packet and exports no quotient carrier, host equality, completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness coordinate.*

Proof. Apply Definition 38.419 to the replay routes from S and T. Each route is exhausted by the same four open-phase faces because the displayed observation, stream-window, regularity, seal/readback, transport, continuation, package, and certificate rows agree componentwise.

Use Definition 38.413 to identify the common pullback packet. Componentwise agreement on the four projections determines one finite BHist source, so the two packages have the same projected consumer boundary.

Now apply Definition 38.418. The $\mathsf{StreamName}^{\uparrow}$ factorization boundary and the four-face pullback boundary read the same packet through the same rows. Therefore the replay route visible to any consumer is the same for S and T, and no off-table coordinate can separate the two reads. □

Theorem 38.421 (RealUp seal source classifier correspondence)**.** *For accepted finite* $\mathsf{Real}^{\uparrow}$ *seal handoffs S and T, the consumer-visible seal-source classifier identifies S and T if and only if their four open-phase faces classify componentwise: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity spine rows, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback rows classify through the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

This correspondence is exactly the seal-source read seen by consumers. It has no coordinate for quotient representatives, host equality, real completeness, arithmetic packages, order packages, Classical choice, or metric completeness.

Proof. For the forward direction, assume the consumer-visible seal-source classifier identifies S and T. Apply Definition 38.411 to expose the four faces available to that classifier.

Project the classifier along those faces. Since the visible classifier uses only the displayed hsame, Cont, Pkg, and NameCert coordinates, the dyadic observation rows, stream-window rows, regularity spine rows, and seal/readback rows classify componentwise.

For the reverse direction, assume componentwise classification on the four faces. Apply Definition 38.413 to repack those compatible faces into the unique finite pullback packet.

Next use Definition 38.418 and Definition 38.419. The correspondence reads the pullback packet as the $\mathsf{StreamName}^{\uparrow}$ factorization boundary, and route totality says every consumer replay is exhausted by that packet. Thus the consumer-visible seal-source classifier identifies S and T. □

Theorem 38.422 (RealUp open-phase four-face NameCert admission)**.** *Let P be a finite packet whose four displayed faces are* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows, and* $\mathsf{Real}^{\uparrow}$ *seal/readback rows. Suppose the only gluing coordinates of P are the componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *rows already displayed on those four faces. Then P is admitted as the local* $\mathsf{Real}^{\uparrow}$ *open-phase* NameCert *surface for the seal handoff.*

The admission has no coordinate for quotient carriers, host equality, real completeness, arithmetic packages, order packages, Classical choice, or metric-completeness rows. Every admitted row is one of the four faces or one of the displayed transport, continuation, package, and certificate coordinates.

Proof. First apply Definition 38.408. It reads any accepted seal/readback row in the packet as a finite open-phase seal surface over the same BHist source and over the displayed hsame, Cont, Pkg, and NameCert coordinates.

Next use Definition 38.413. The dyadic observation face, the stream-name finite-window face, the regularity-spine face, and the seal/readback face are compatible along exactly those displayed coordinates, so they form the local four-face pullback packet.

Apply Definition 38.419. It says every consumer-visible route out of the seal handoff is exhausted by the same four-face packet, so no fifth surface is available during admission.

Finally use Definition 38.421. The classifier seen by consumers is the componentwise classifier on the four faces, transported only through hsame, Cont, Pkg, and NameCert. Thus the finite packet is precisely the local $\mathsf{Real}^{\uparrow}$ open-phase NameCert surface, and the excluded coordinates have no carrier in the admission. □

Theorem 38.423 (RealUp open-phase NameCert surface completeness)**.** *A local* $\mathsf{Real}^{\uparrow}$ *open-phase* NameCert *surface for the seal handoff is admitted exactly when it is represented by one finite four-face* BHist *packet whose faces are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback provenance rows. The only gluing coordinates of that packet are the displayed componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

The representing packet is unique among compatible four-face packets. The surface has no quotient carrier, host-equality coordinate, real-completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness row.

Proof. For existence, start with the four displayed faces and their componentwise hsame, Cont, Pkg, and NameCert coordinates. Apply Definition 38.422. It admits that finite packet as the local $\mathsf{Real}^{\uparrow}$ open-phase NameCert surface and leaves no carrier for any off-table row.

For uniqueness, compare two admitted surfaces with the same displayed faces. Use Definition 38.413. Compatible dyadic observation rows, stream-window rows, regularity-spine rows, and seal/readback rows determine exactly one finite BHist pullback packet over the shown coordinates.

The classifier boundary is the same boundary. Apply Definition 38.421. The consumer-side seal-source classifier is precisely the componentwise classifier on the four faces, so no quotient representative, host equality, arithmetic order, or completeness coordinate can refine the surface.

Finally apply Definition 38.419. Every consumer-visible route out of the handoff is exhausted by the four-face packet. Therefore the admitted local NameCert surface is equivalent to that unique finite packet, and the listed excluded rows are outside the open-phase seal surface. □

Theorem 38.424 (RealUp open-phase window splice non-escape)**.** *Let two finite* $\mathsf{Real}^{\uparrow}$ *seal handoffs be displayed over the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window cover, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, compatible* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows, and the same componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *gluing rows. Then every consumer-visible splice between the two handoffs factors through the same four-face* BHist *packet.*

Consequently the splice exports no quotient carrier, host equality, selected limit, real-completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness row. Its whole public boundary is the shared finite packet consisting of the dyadic observation face, the stream-window face, the regularity-spine face, and the seal/readback face.

Proof. Apply Definition 38.413 to the two handoffs. The common stream-window cover, common dyadic ledger, compatible regularity spines, and identical displayed gluing rows determine one finite four-face BHist packet on both sides of the splice.

Next use Definition 38.423. It identifies the admitted local NameCert surface of each handoff with exactly that finite packet, and it lists the only coordinates available to the open-phase seal boundary.

It remains to rule out an escaping consumer route. Apply Definition 38.419 to the splice read, and then use Definition 38.421. The route totality theorem exhausts all consumer-visible reads by the four displayed faces, while the classifier correspondence transports only the componentwise classifier through hsame, Cont, Pkg, and NameCert.

Therefore the splice is a read of the shared packet itself. No quotient carrier, host equality, selected limit, completeness row, arithmetic or order package, Classical-choice row, or metric-completeness row is present in the source packet, so none can be exported by the consumer-visible splice. □

Theorem 38.425 (RealUp open-phase terminal consumer non-escape)**.** *A terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer boundary is accepted exactly when its route factors through the finite four-face packet consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-spine rows, and the* $\mathsf{Real}^{\uparrow}$ *seal/readback rows, glued only by the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Consequently the terminal boundary exports no quotient carrier, host equality, selected limit, completeness row, arithmetic package, order package, Classical-choice row, or metric-completeness coordinate. Every accepted terminal read is a read of the four displayed faces themselves.

Proof. First identify the admissible surface. Apply Definition 38.423. It says the local NameCert surface for an open-phase seal is exhausted by the dyadic observation face, stream-window face, regularity-spine face, seal/readback face, and the displayed hsame, Cont, Pkg, and NameCert coordinates.

Now route the terminal consumer read. Apply Definition 38.419. A consumer-visible terminal route out of the seal has no fifth source row; it must pass through the same four-face packet before any public read is accepted.

It remains to check that the route is stable under the naming and classifier surface. Use Definition 38.421 to transport seal-source classifiers componentwise, and use Definition 38.418 to keep the stream-name factorization inside the displayed finite-window pullback.

Thus both directions are fixed by the same finite packet: every accepted terminal consumer route factors through the four faces, and every route built from those faces is accepted by the open-phase NameCert surface. Since the packet contains no quotient carrier, host equality, selected limit, completeness row, arithmetic or order package, Classical-choice row, or metric-completeness coordinate, none can be exported at the terminal boundary. □

Theorem 38.426 (RealUp open-phase terminal pullback lock)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *open-phase consumer that reads a finite seal through the displayed handoff factors through the same* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The terminal route therefore has no coordinate for quotient real equality, selected host limits, host equality, trichotomy, countable choice, or ambient completeness.*

Proof. First apply Definition 38.423 and Definition 38.419. Together they identify the local seal surface and force every accepted consumer read through it.

The common-refinement consumption and handoff rows are the four finite faces named in Definition 38.425. Use Definition 38.418 to keep the StreamName window, readback, ledger, transports, continuations, provenance, and naming row in the same pullback packet.

Those coordinates exhaust the route. Since none of the cited packets contains quotient real equality, selected host limits, host equality, trichotomy, countable choice, or ambient completeness, the terminal consumer cannot export any of those rows. □

Definition 38.427 (Real RegSeqRat StreamName dependency triple)**.** *A* $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triple is a finite* BHist *packet*

$$(S, Q, D, R, H, C, P, N)$$

whose visible rows have the following shape:

1. *S records the inspected* $\mathsf{StreamName}^{\uparrow}$ *finite windows and their transported stream-name classifier rows;*
2. *Q records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface over the same scheduled windows;*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger read pointwise through those windows;*
4. *R records the* $\mathsf{Real}^{\uparrow}$ *seal row and its finite endpoint readback;*
5. *H records componentwise* hsame *comparisons for the window, readback, dyadic, and seal rows;*
6. *C records the displayed* Cont *route from each public seal read to the preceding window, readback, dyadic, and seal coordinates;*
7. *P records the* Pkg *provenance for the four sibling surfaces;*
8. *N is the local* NameCert *row for this dependency packet.*

The classifier compares dependency triples only through these displayed rows. It has no coordinate for a quotient real, selected host limit, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row.

Theorem 38.428 (Real RegSeqRat StreamName seal-route exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal read from a dependency triple* (S, Q, D, R, H, C, P, N) *is witnessed by the four displayed rows* S, Q, D, R*. The tuple projection inventory is* (S, Q, D, R, H, C, P, N)*, while the source-dependency route is* $D \to S \to Q \to R$*: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D *supplies the endpoint and radius ledger read through the* $\mathsf{StreamName}^{\uparrow}$ *finite window row* S *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row* Q *and the local* $\mathsf{Real}^{\uparrow}$ *seal row* R *are consumed. The factorization is transported by the componentwise* hsame *row* H*, read through the displayed* Cont *route* C*, and packaged by* P, N*.*

Consequently the accepted seal route is exactly the sibling route whose source face is chapter 291, whose finite observation window is scheduled by chapter 126, and whose regular readback is supplied by chapter 301 before the $\mathsf{Real}^{\uparrow}$ *seal is consumed. It exports no quotient real, selected host limit, host equality, ambient completeness coordinate, arithmetic package, order package, or metric-completeness row.*

Proof. Begin with an accepted seal read from (S, Q, D, R, H, C, P, N). The definition of the dependency triple exposes only the finite window row S, regular rational readback Q, dyadic observation ledger D, seal row R, and the transport, route, package, and local naming rows H, C, P, N.

Use Definition 38.407. That handoff identifies the $\mathsf{Real}^{\uparrow}$ seal-facing package as the stream-name classifier row, transported unary classifier row, finite-window ledger, endpoint readback, and seal provenance.

Now read the upstream sibling surfaces. The DyadicRatCore, StreamName, and RegSeqRat chapters named in chapter 291, chapter 126, and chapter 301 provide the source ledger, scheduled window, and regular readback rows used by the handoff. The displayed H, C, P, N rows keep those coordinates in one BHist packet.

Since the packet has no remaining coordinate after R, a public seal read cannot branch to a quotient real, selected host limit, host equality, ambient completeness principle, arithmetic or order package, or metric-completeness row. Thus every accepted read is exhausted by the stated route. □

Theorem 38.429 (Real RegSeqRat StreamName Cauchy-completion interface)**.** *For the current* $\mathsf{Real}^{\uparrow}$ *completion-facing phase, the Cauchy-completion interface exposed by a Real RegSeqRat StreamName dependency triple is exactly the finite source-dependency route through chapter 291, chapter 126, chapter 301, and the local* $\mathsf{Real}^{\uparrow}$ *seal row. Any completion-facing consumer must factor through this route before it reads a* $\mathsf{Real}^{\uparrow}$ *endpoint.*

The interface exports the displayed finite window, regular-rational readback, dyadic observation ledger, seal row, componentwise hsame *transport, displayed* Cont *route, package provenance, and local* $\mathsf{NameCert}$ *row. It exports no quotient stream, selected host limit, representative choice, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row.*

Proof. Start from the dependency triple of Definition 38.427. Its tuple inventory is listed as (S, Q, D, R, H, C, P, N), but its inbound source route begins with the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row D, reads that ledger through the finite $\mathsf{StreamName}^{\uparrow}$ window S, and then reaches the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q before the $\mathsf{Real}^{\uparrow}$ seal row R is consumed in the same finite packet.

Apply the seal-route exhaustion theorem Definition 38.428. It identifies every accepted seal read as the route sourced at D, scheduled through S, read back through Q, and sealed at R, transported by H, read through C, and packaged by P, N.

The local seal row is therefore the completion-facing boundary. Since the packet has no coordinate for quotient streams, selected host limits, representative choice, host equality, completeness,

arithmetic, order, or metric completeness, a consumer cannot read any of those objects before or after the $\mathsf{Real}^{\uparrow}$ endpoint. The only available interface is the stated finite route. □

Theorem 38.430 (Real open-phase dependency triple current surface)**.** *Every current Phase Real and completion-facing* $\mathsf{Real}^{\uparrow}$ *seal consumer of the dependency triple*

$$(S, Q, D, R, H, C, P, N)$$

factors through exactly four displayed faces: the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback face* Q*, and the local* $\mathsf{Real}^{\uparrow}$ *seal face* R*. The factorization is transported only by the displayed componentwise* hsame *row* H*, the* Cont *route row* C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*.*

Conversely, those four faces together with H, C, P, N *assemble the accepted current surface for such consumers. The surface has no coordinate for quotient streams, host equality, selected limits, ambient completeness, arithmetic packages, order packages, or metric-completeness rows.*

Proof. Start with the displayed dependency triple of Definition 38.427. Its carrier rows are exactly S, Q, D, R, H, C, P, N, so any current consumer must begin from those coordinates.

Apply Definition 38.428. It exhausts each accepted seal route by the dyadic observation row D, the finite-window row S, the regular readback row Q, and the local seal row R, while the packet still projects them as S, Q, D, R and transports the route through H, C, P, N.

Apply Definition 38.429. The same route is the completion-facing interface before a $\mathsf{Real}^{\uparrow}$ endpoint is read, so Phase Real and completion RealUp seal consumers share the displayed four-face surface.

For the converse, assemble the accepted surface from the four listed faces and the transport, route, provenance, and certificate rows. The exclusion clause in the dependency triple removes quotient streams, host equality, selected limits, ambient completeness, arithmetic packages, order packages, and metric-completeness rows, leaving no further coordinate to add. □

Theorem 38.431 (Real seal RegSeqRat StreamName dependency exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal consumer that reads the dependency triple of Definition 38.427 factors through exactly the finite rows supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert *row.*

The same consumer exports no quotient stream, selected host limit, representative choice, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row.

Proof. Apply Definition 38.428 to the seal read. It identifies the only accepted route as the path through the finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row, the $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the local $\mathsf{Real}^{\uparrow}$ seal row, transported by hsame, read through Cont, and packaged by Pkg and NameCert.

Apply Definition 38.429. It turns the same finite route into the completion-facing interface and states that every completion-facing consumer must read through that interface before it reaches a $\mathsf{Real}^{\uparrow}$ endpoint.

Finally project the dependency triple itself. Its displayed coordinates are only (S, Q, D, R, H, C, P, N), and its exclusion clause contains no quotient stream, selected host limit, representative choice, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row. Hence there is no non-finite route for the consumer to export. □

Theorem 38.432 (Real finite-window completion boundary)**.** *For any accepted* $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triple* (S, Q, D, R, H, C, P, N)*, every completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite source-dependency route in which the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D *is read through the* $\mathsf{StreamName}^{\uparrow}$ *window row* S *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the local* $\mathsf{Real}^{\uparrow}$ *seal row* R*. The passage is transported by the componentwise* hsame *row* H*, read through the displayed* Cont *route* C*, and packaged by* Pkg *provenance and the local* $\mathsf{NameCert}$ *row* N*.*

No quotient stream, selected host limit, representative choice, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row is exported by this completion-facing boundary.

Proof. Apply Definition 38.428. It gives the only seal-facing route through the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ seal rows, with the displayed H, C, P, N transport, continuity, package, and naming data.

Next apply Definition 38.429. This turns the same finite route into the completion-facing interface and forces every completion consumer to pass through it before reading a $\mathsf{Real}^{\uparrow}$ endpoint.

Finally use the coordinate exclusion clause in Definition 38.427. The accepted packet has only the rows (S, Q, D, R, H, C, P, N), so no quotient stream, selected host limit, representative choice, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row can be exported. □

Theorem 38.433 (Real RegSeqRat StreamName seal-source minimality)**.** *For an accepted* $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triple* (S, Q, D, R, H, C, P, N)*, the current accepted seal source is minimal in its four open-phase faces. Removing any one of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face* Q*, or the local* $\mathsf{Real}^{\uparrow}$ *seal face* R *leaves no accepted current Phase Real or Completion seal source.*

The transport, continuation, package, and local certificate rows H, C, P, N *do not replace a missing face. They only carry and name the route after all four faces are present in the displayed dependency triple.*

Proof. Start from the route exhaustion of Definition 38.428. It places every accepted seal read on the finite source route through D, then S, then Q, and only then R, while H, C, P, N carry that route inside the displayed packet inventory.

Use the current-surface theorem Definition 38.430. Its converse assembles the accepted current surface from precisely the four faces together with H, C, P, N, and its exclusion clause leaves no quotient, host, completeness, arithmetic, order, or metric-completeness coordinate outside them.

Now delete one face. If S is absent, there is no finite-window source for the dyadic ledger read. If Q is absent, there is no regular-rational readback after the scheduled window. If D is absent, there is no dyadic observation source for the seal route. If R is absent, there is no local Real seal for the endpoint read. In each case the Cauchy-completion interface of Definition 38.429 has no accepted endpoint route, so no current seal source remains. □

Theorem 38.434 (Real RegSeqRat StreamName current-phase gate)**.** *Every current Phase Real and Completion consumer of an accepted dependency triple* (S, Q, D, R, H, C, P, N) *must enter through the four-face route*

$$D, \quad S, \quad Q, \quad R$$

before it reads a $\mathsf{Real}^{\uparrow}$ *endpoint. The gate is carried only by the componentwise* hsame *row* H*, the displayed* Cont *route* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

No endpoint read is licensed from quotient streams, selected host limits, representative choices, host equality, ambient completeness, arithmetic or order packages, or metric-completeness rows.

Proof. First apply Definition 38.429. It states that every completion-facing consumer must pass through the finite DyadicRatCore, StreamName, RegSeqRat, and local Real seal route before reading a $\mathsf{Real}^{\uparrow}$ endpoint.

Next apply Definition 38.432. The boundary repeats the same source-entry order for completion-facing consumers while the packet keeps the tuple projection order (S, Q, D, R, H, C, P, N), and identifies H, C, P, N as the only transport, route, provenance, and certificate rows attached to that entry.

Finally apply Definition 38.433. Since no one of the four faces can be removed while retaining an accepted current seal source, the entry route is a gate rather than an optional presentation of the endpoint. □

Theorem 38.435 (Real RegSeqRat StreamName no completion escape)**.** *Completion-facing consumers of an accepted* $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triple cannot bypass the four faces through quotient streams, selected host limits, representative choices, host equality, ambient completeness, arithmetic packages, order packages, or metric-completeness rows. Their only accepted completion route is the displayed source path through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, and the local* $\mathsf{Real}^{\uparrow}$ *seal face.*

Proof. Use route exhaustion to identify the only seal-facing path. By Definition 38.428, that path is the displayed finite source route through D, S, Q, R with H, C, P, N, even though the packet projection inventory is written as (S, Q, D, R, H, C, P, N).

Use the current-phase gate to place every Phase Real and Completion endpoint read behind the same path. A completion-facing consumer therefore cannot reach the endpoint before the four faces have all been read.

It remains to exclude escape coordinates. The finite-window completion boundary of Definition 38.432 and the Cauchy-completion interface of Definition 38.429 both remove quotient streams, selected host limits, representative choices, host equality, ambient completeness, arithmetic and order packages, and metric-completeness rows from the exported boundary. Since minimality also prevents deleting any face, there is no completion route outside the four displayed faces. □

Theorem 38.436 (Real RegSeqRat StreamName endpoint determinacy)**.** *Let* (S, Q, D, R, H, C, P, N) *and* $(S', Q', D', R', H', C', P', N')$ *be accepted* $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. Suppose the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows* D, D' *are read through the advertised* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* S, S' *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q' *using only the* hsame *transport rows in* H, H'*, and suppose this is the same displayed route through chapter 291, chapter 126, and chapter 301.*

Then any public $\mathsf{Real}^{\uparrow}$ *endpoint comparison between the two consumers factors only through the finite packets*

$$(S, Q, D, R, H, C, P, N) \quad \textit{and} \quad (S', Q', D', R', H', C', P', N').$$

In particular the comparison has no coordinate for quotient streams, host real equality, selected host limits, representative choice, ambient completeness, arithmetic or order packages, or metric-completeness rows.

Proof. First apply Definition 38.428 to each accepted consumer. Each endpoint read reaches the $\mathsf{Real}^{\uparrow}$ seal only through its displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row, $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, local seal row, hsame transport, Cont route, Pkg provenance, and local NameCert row.

Next use Definition 38.429. The completion-facing boundary is therefore the same finite route, so an endpoint comparison cannot consult data outside the two displayed packets before it compares the public endpoints.

Finally apply Definition 38.431. That theorem exhausts the coordinate supply for every accepted seal consumer by the rows named in the dependency triple. Since the hypotheses identify the window, readback, and dyadic route through the three sibling chapters, the remaining public comparison is determined by the two finite packets alone, with no quotient-stream, host-equality, selected-limit, representative-choice, ambient-completeness, arithmetic, order, or metric-completeness coordinate available. □

Theorem 38.437 (Real RegSeqRat StreamName finite-window pullback classifier)**.** *Let*

$$X = (S, Q, D, R, H, C, P, N) \quad \textit{and} \quad Y = (S', Q', D', R', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. The two triples classify as the same completion-facing* $\mathsf{Real}^{\uparrow}$ *finite-window packet if and only if all of the following displayed rows agree componentwise:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* S, S'*;*
2. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q'*;*
3. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows* D, D'*;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal rows* R, R'*;*
5. *the componentwise* hsame *transports displayed in* H, H'*;*
6. *the displayed* Cont *routes in* C, C'*;*
7. *the* Pkg *provenance rows* P, P'*;*
8. *the local* $\mathsf{NameCert}$ *rows* N, N'*.*

Equivalently, the classifier for accepted Real RegSeqRat StreamName dependency triples is the finite pullback of the four open-phase faces $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with the displayed transport, route, provenance, and local naming rows. It has no coordinate for quotient streams, host equality, selected limits, ambient completeness, arithmetic or order packages, or metric-completeness rows.*

Proof. For the forward direction, assume the two accepted triples classify as the same completion-facing finite-window packet. Apply Definition 38.428 to both triples. Each accepted route is exhausted by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now apply Definition 38.436. Since a public endpoint comparison cannot consult any coordinate outside those finite packets, equality of the completion-facing packet projects to componentwise agreement of the stream-name window, regular readback, dyadic ledger, real seal, transport, route, provenance, and local naming rows.

For the reverse direction, assume the eight displayed rows agree componentwise. Use Definition 38.413 to repack the four open-phase faces as a single finite pullback packet. The matching hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows provide exactly the gluing data required by that pullback.

Finally read the resulting packet through Definition 38.427. The local dependency triple definition accepts precisely those rows as the completion-facing Real finite-window packet, so the two accepted triples classify the same packet. The exclusion clause in the definition removes quotient

streams, host equality, selected limits, ambient completeness, arithmetic and order packages, and metric-completeness rows from the classifier. □

Theorem 38.438 (Real RegSeqRat StreamName dyadic window lock)**.** *Let*

$$X = (S, Q, D, R, H, C, P, N) \quad \textit{and} \quad Y = (S', Q', D', R', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. Suppose the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* S, S'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q'*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows* D, D'*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows* R, R' *agree componentwise under the shown* hsame *transports in* H, H'*, the displayed* Cont *routes in* C, C'*, the package rows* P, P'*, and the local* NameCert *rows* N, N'*.*

Then every completion-facing $\mathsf{Real}^{\uparrow}$ *consumer comparison between* X *and* Y *factors through exactly the common finite four-face window lock*

$$(S, Q, D, R, H, C, P, N) \simeq (S', Q', D', R', H', C', P', N').$$

That lock is the shared pullback of the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, and the local* $\mathsf{Real}^{\uparrow}$ *seal face, with no quotient stream, selected host limit, host equality, representative choice, ambient completeness principle, arithmetic package, order package, or metric-completeness row.*

Proof. First apply Definition 38.428 to both accepted triples. Each seal-facing route is exhausted by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Next apply Definition 38.429. It turns the same exhausted route into the completion-facing boundary, so a consumer comparison has no access to data before or after the finite route except through the rows shown in the two packets.

Use Definition 38.431 to remove the remaining escape coordinates. Quotient streams, selected limits, representative choices, host equality, ambient completeness, arithmetic and order packages, and metric-completeness rows are not coordinates of the accepted dependency triples.

Finally apply Definition 38.437. The componentwise agreement hypotheses are exactly the pullback agreement conditions for the four open-phase faces and their structural rows. Therefore every completion-facing comparison factors through that common finite window lock and through nothing larger. □

Theorem 38.439 (Real L10 four-face route lock)**.** *Let*

$$X = (S, Q, D, R, H, C, P, N) \quad \textit{and} \quad Y = (S', Q', D', R', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. Every current Phase Real and Completion consumer comparison between* X *and* Y *factors through exactly the four open-phase faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The only comparison structure available to that factorization is the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows in the two packets.*

Thus no quotient stream, host equality, selected limit, ambient completeness, arithmetic package, order package, or metric-completeness row is available to the comparison. The comparison is locked to the local finite route through the dyadic observation ledger, stream-name finite window, regular-rational readback, and local real seal.

Proof. First apply Definition 38.428. It exhausts every accepted seal-facing route by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row, $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, local $\mathsf{Real}^{\uparrow}$ seal row, and the shown hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 38.437. The classifier for accepted Real RegSeqRat StreamName dependency triples is the finite pullback of those four open-phase faces together with the displayed structural rows, so a consumer comparison has no independent coordinate outside that pullback.

Finally apply Definition 38.438. The common finite window lock forces every completion-facing comparison between accepted packets through the shared four-face window. Since the three results already remove quotient streams, host equality, selected limits, ambient completeness, arithmetic and order packages, and metric-completeness rows, the Real L10 comparison route is exactly the four-face route stated above. □

Theorem 38.440 (Real RegSeqRat StreamName terminal common-source lock)**.** *Let*

$$X = (S, Q, D, R, H, C, P, N) \quad \textit{and} \quad Y = (S', Q', D', R', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window faces* S, S'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback faces* Q, Q'*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger faces* D, D'*, the* $\mathsf{Real}^{\uparrow}$ *seal faces* R, R'*, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows* H, H'*,* C, C'*,* P, P'*, and* N, N' *agree componentwise.*

Then every terminal $\mathsf{Real}^{\uparrow}$ *endpoint consumer comparison between* X *and* Y *factors through the same common source packet*

$$(S, Q, D, R, H, C, P, N) \simeq (S', Q', D', R', H', C', P', N').$$

No such terminal comparison can branch through a quotient stream, selected host limit, host equality, ambient completeness principle, arithmetic package, order package, or metric-completeness row hidden behind the endpoint.

Proof. First apply Definition 38.439. It identifies every current Phase Real and Completion consumer comparison with the displayed four-face route through the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and their structural rows.

Next apply Definition 38.437. The componentwise agreement assumptions are precisely the pullback classifier coordinates for the accepted dependency triples, so the two endpoint packets share the same source packet rather than merely matching at the final seal.

Finally apply Definition 38.438. The shared finite window lock is already the common source for completion-facing comparisons. Together with the exclusion clauses in the dependency triple, this prevents a terminal endpoint comparison from branching through quotient streams, selected host limits, host equality, ambient completeness, arithmetic or order packages, or metric-completeness rows. □

Theorem 38.441 (Real RegSeqRat StreamName four-face consumer boundary)**.** *Let*

$$X = (S, Q, D, R, H, C, P, N) \quad \textit{and} \quad Y = (S', Q', D', R', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *RegSeqRat StreamName dependency triples. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window faces,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback faces,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger faces,* $\mathsf{Real}^{\uparrow}$ *seal faces, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows agree componentwise between* X *and* Y*.*

Then every current Phase Real and Completion consumer boundary over $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, completion-facing reads, or stream-facing reads is the same finite boundary packet*

$$(S, Q, D, R, H, C, P, N) \simeq (S', Q', D', R', H', C', P', N').$$

The boundary exposes only the DyadicRatCore ledger face, the StreamName finite-window face, the RegSeqRat readback face, the RealUp seal face, and the displayed hsame, Cont, Pkg, *and* NameCert *rows. Quotient streams, host equality, selected limits, ambient completeness, arithmetic or order packages, and metric-completeness rows are not exported through this consumer boundary.*

Proof. Begin with Definition 38.434. It says that the current Phase Real and Completion consumers can read only the open Real RegSeqRat StreamName interface and the displayed structural rows.

Next use Definition 38.437. The componentwise agreement hypotheses are exactly the finite pullback coordinates of that interface, so Real-facing, completion-facing, and stream-facing reads receive one packet rather than separate witnesses.

Apply Definition 38.439 to identify the available route as the four-face route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Then apply Definition 38.440 to turn the four agreed faces and their displayed rows into the common terminal source.

The exclusion clauses from those route-lock theorems remove every ambient coordinate outside the packet. Hence quotient streams, host equality, selected limits, ambient completeness, arithmetic and order packages, and metric-completeness rows cannot be observed by any current consumer boundary, which leaves exactly the finite boundary packet stated above. □

Definition 38.442 (Real L10 dependency lattice)**.** *The* $\mathsf{Real}^{\uparrow}$ *L10 dependency lattice is the ordered finite certificate route whose source rows are read in this order:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows supplied by chapter 291;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite stream-window rows supplied by chapter 126;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational classifier rows supplied by chapter 301;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal, endpoint readback, and provenance rows.*

The route is finite: it is carried by the displayed BHist *packet, transported only by* hsame*, read only through* Cont*, and named only by the local* NameCert *row before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Theorem 38.443 (Real L10 source route non-escape)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *source-route consumer for the L10 dependency lattice must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic observation rows, then the* $\mathsf{StreamName}^{\uparrow}$ *finite stream-window rows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational classifier rows, and only then the local* $\mathsf{Real}^{\uparrow}$ *seal rows. The consumer may transport these rows by the displayed* hsame *coordinates, read them through the displayed* Cont *route, and package them with the displayed* Pkg *and local* NameCert *rows.*

No accepted source-route consumer has a coordinate for a quotient stream, host completion, countable choice, host real equality, selected host limit, ambient completeness principle, arithmetic package, order package, or metric-completeness row.

Proof. Start with the carrier and source specification in Definition 38.1 and Definition 38.7. The carrier is a sealed RegSeqRat name over DyadicRatCore observations, and the source specification lists exactly the dyadic observation layer, the stream-name layer, and the regular-sequence certificate layer.

Now apply the concrete seal handoff in Definition 38.407. It reads those layers through the sibling chapters chapter 291, chapter 126, and chapter 301, then repacks the result as the local $\mathsf{Real}^{\uparrow}$ seal handoff. This gives the ordered L10 route of Definition 38.442.

Finally use the exhaustion and boundary theorems for the Real RegSeqRat StreamName dependency triple, Definition 38.428 and Definition 38.429. They say that every accepted consumer is exhausted by the finite rows of the displayed packet, with only hsame, Cont, Pkg, and NameCert

structural data. Since none of the cited source, handoff, or dependency-triple blocks exports a quotient stream, host completion, countable choice, host real equality, selected host limit, ambient completeness, arithmetic, order, or metric-completeness coordinate, the consumer has no route to escape the L10 lattice. □

Lean4 ✓ `BEDC.Derived.RealUp.RealL10DependencyLattice source route nonescape`

Theorem 38.444 (Real L10 interface totality)**.** *Every current Phase Real and Completion interface read of an accepted* $\mathsf{Real}^{\uparrow}$ *L10 dependency-lattice packet factors through exactly the four displayed faces of Definition 38.442:* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal and readback rows. The transport allowed between these faces is only the displayed* hsame *data, the displayed* Cont *routes, the displayed* Pkg *provenance, and the local* NameCert *data; no further quotient, completion, host-real, selector, choice, arithmetic, order, or metric-completeness interface row is exported.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10DependencyLatticeInterfaceTotality`

Proof. Start with the ordered source route of Definition 38.443. It confines accepted L10 consumers to the dyadic observation layer, the StreamName finite-window layer, the RegSeqRat regular-rational layer, and the local Real seal layer, with transport only through the displayed structural rows.

Next use the open-phase consumer theorem Definition 38.409. It identifies the current Phase Real and Completion reads with the same four faces, so an interface consumer cannot insert a fifth source row between the dependency lattice and the public open-phase surface.

The two sibling handoff theorems give the middle faces. By Definition 126.132, finite StreamName windows reach the Real handoff only through their displayed window, RegSeqRat, Real-boundary, transport, continuation, provenance, and naming rows. By Definition 301.22, the regular-rational face supplies exactly the seal-facing rows consumed by $\mathsf{Real}^{\uparrow}$.

Finally apply the current-surface theorem for the dependency triple, Definition 38.430. It repackages the same DyadicRatCore, StreamName, RegSeqRat, and local Real rows as the current interface surface. Since every cited route exports only hsame, Cont, Pkg, and NameCert data besides those four faces, the interface read is exhausted by the displayed L10 lattice. □

Theorem 38.445 (Real L10 terminal face order)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *open-phase terminal read consumes the four faces of Definition 38.442 in the order* $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$*. The terminal* $\mathsf{Real}^{\uparrow}$ *seal cannot be read before the dyadic observation rows, the finite-window rows, and the regular-rational classifier rows have been displayed through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Consequently a terminal read has no coordinate through which to consume quotient stream equality, selected limits, host real equality, ambient completeness, or any seal-facing row outside the displayed four-face route.

Lean4 ✓ `BEDC.Derived.RealUp.RealL10TerminalFaceOrder`

Proof. Start from the ordered route in Definition 38.442. The lattice lists the dyadic observation face, the StreamName finite-window face, the RegSeqRat classifier face, and only then the local Real seal face.

Apply the non-escape theorem Definition 38.443. It prevents any accepted source-route consumer from bypassing the dyadic, stream, or regular-rational faces before the local seal is reached.

Next use interface totality, Definition 38.444. Since every open Phase Real and Completion interface read factors through exactly those four faces, a terminal read cannot insert a further interface row between the RegSeqRat classifier and the Real seal.

The middle dependency is exact by Definition 126.349: the stream window read is pulled back to the same dyadic and regular-rational source surface. The final handoff is exact by Definition 301.22: the Real seal consumes the regular-rational source rows already displayed by the finite window.

Coordinate exhaustion then excludes all other terminal readings. The cited routes expose only BHist, hsame, Cont, Pkg, and NameCert data over the four named faces, so quotient stream equality, selected limits, host real equality, ambient completeness, and outside seal data have no terminal coordinate to consume. □

Theorem 38.446 (Real L10 terminal comparison exhaustion)**.** *Let X and Y be accepted terminal* $\mathsf{Real}^{\uparrow}$ *L10 reads. Every completion-facing comparison between X and Y factors through the same four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

displayed in Definition 38.442. The comparison may use only the common BHist *packet, the displayed* hsame *transports, the displayed* Cont *routes, the displayed* Pkg *provenance, and the local* NameCert *coordinates of those faces.*

No completion-facing comparison of X and Y has a coordinate for quotient stream equality, selected limits, host real equality, ambient completeness, arithmetic or order packages, or metric-completeness rows.

Proof. Apply the terminal face order theorem, Definition 38.445, to both accepted reads. It places the dyadic observation face, the finite StreamName window face, the RegSeqRat classifier face, and the Real seal face in the same terminal order for X and Y.

Next apply the four-face route lock, Definition 38.439. A completion-facing comparison cannot leave the displayed route once the two terminal reads have been accepted on that route; its transport and readback data remain confined to BHist, hsame, Cont, Pkg, and NameCert coordinates.

The shared finite surface is supplied by Definition 126.356. It identifies the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal boundary as one pullback surface for the two terminal reads.

Finally use the observation triangle Definition 38.384. The dyadic, stream, and regular-rational observations meet on the same finite triangle before the Real seal is consumed. Since the cited routes export no quotient stream equality, selected limit, host real equality, ambient completeness, arithmetic or order package, or metric-completeness coordinate, the comparison is exhausted by the four displayed faces. □

Theorem 38.447 (Real L10 four-source audit-clean handoff)**.** *Every current Phase Real and Completion exit read of an accepted* $\mathsf{Real}^{\uparrow}$ *packet factors through the ordered four-source packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The only structural transports available to the handoff are the displayed BHist *packets,* hsame *transports,* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows attached to those four sources. In particular, the handoff admits no quotient stream equality, selected limit, host real equality, ambient completeness, arithmetic package, order package, metric-completeness row, or bridge row as an additional source surface.*

Proof. Start from the Real L10 dependency lattice, Definition 38.442. It names the four source faces and places the dyadic observation rows before StreamName windows, RegSeqRat readback, and the local $\mathsf{Real}^{\uparrow}$ seal.

Apply interface totality and terminal face order, Definition 38.444 and Definition 38.445. These results make every current Phase Real and Completion interface read pass through the same ordered packet before any terminal exit can consume the Real seal.

Read the checked source surfaces in that order. The DyadicRatCore face is the standard-source bridge `DyadicRatCoreCarrier_standard_source_bridge`, the StreamName face is supplied by `RatStreamName_certificate_fields`, the RegSeqRat face is the real-seal handoff `RegSeqRatStreamCarrier_rea` and the Real face is the standard bridge `RealUp_StdBridge`.

Each cited surface exports only its local row data into the ordered packet. Thus the common handoff can transport by BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows, and has no coordinate through which to import any extra completion, arithmetic, order, metric, quotient, limit, or bridge surface. □

Theorem 38.448 (Real L10 dependency-lattice exit handoff)**.** *Every current Phase Real and Completion exit read of an accepted* $\mathsf{Real}^{\uparrow}$ *packet factors through exactly the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The handoff surface consists only of the displayed BHist *packets, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and the local* $\mathsf{NameCert}$ *rows attached to those four faces.*

Consequently the exit read has no coordinate for quotient streams, selected limits, host real equality, ambient completeness, arithmetic or order packages, metric-completeness rows, or any other completion-facing structure outside the displayed dependency lattice.

Proof. Start with the dependency lattice of Definition 38.442. It names the only route available to the current exit surface: dyadic observations, finite StreamName windows, RegSeqRat regular-rational readback, and the local $\mathsf{Real}^{\uparrow}$ seal.

Apply source-route non-escape, Definition 38.443. This prevents an accepted exit read from entering through a hidden source, quotient representative, selected limit, host real value, or ambient completeness row before reaching the displayed four faces.

Next use interface totality and terminal face order, Definition 38.444 and Definition 38.445. Together they make every current Phase Real and Completion interface read pass through the same ordered faces, with only BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ transport between them.

Finally apply terminal comparison exhaustion, Definition 38.446. It says that even the terminal comparison surface has no further coordinate beyond the same four-face packet. Thus quotient streams, selected limits, host real equality, ambient completeness, arithmetic or order packages, metric-completeness rows, and hidden completion data are not part of the exit handoff. □

Theorem 38.449 (Real L10 metric boundary non-escape)**.** *Every accepted current Phase Real or Completion consumer of a* $\mathsf{Real}^{\uparrow}$ *L10 packet factors through the four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

before any $\mathsf{Met}^{\uparrow}$ *consumer or metric-completeness consumer can read the packet. The visible route consists only of the DyadicRatCore observation face, the StreamName finite-window face,*

the RegSeqRat regular-rational classifier face, the local $\mathsf{Real}^{\uparrow}$ *seal face, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

The route exports no `MetricDistanceWitness`*, separatedness row, completeness row, Cauchy-limit selector, metric triangle row, or ambient metric-completeness coordinate. Any* $\mathsf{Met}^{\uparrow}$*-facing read is therefore an external consumer boundary for an already sealed* $\mathsf{Real}^{\uparrow}$ *packet, not an input row used to construct the Real seal.*

Proof. First apply the terminal comparison exhaustion theorem, Definition 38.446. It keeps every completion-facing comparison between accepted terminal reads on the same four-face surface, using only the common BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next apply the four-face canonical route, Definition 38.450. It says that every current Phase Real or Completion consumer of the $\mathsf{Real}^{\uparrow}$ packet must pass through the ordered DyadicRatCore, StreamName, RegSeqRat, and local Real faces before the consumer surface is available.

The finite overlap of those faces is the pullback surface of Definition 126.356. Thus the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal boundary are read as one displayed route.

Finally view the $\mathsf{Met}^{\uparrow}$ public interface only as a downstream consumer boundary. Because the cited Real route exports no metric distance witness, separatedness row, completeness row, Cauchy-limit selector, metric triangle row, or ambient metric-completeness coordinate, no metric consumer can supply an input used by the $\mathsf{Real}^{\uparrow}$ L10 seal. □

Theorem 38.450 (Real L10 four-face canonical route)**.** *Every accepted current Phase Real or Completion consumer of a* $\mathsf{Real}^{\uparrow}$ *L10 packet has a canonical finite route through exactly the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The route is displayed only by the common BHist *packet, the shown* hsame *transports, the shown* Cont *routes, the shown* Pkg *provenance, and the local* NameCert *rows.*

If two route presentations have the same four faces and the same displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, then they induce the same consumer surface. No quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness principle, arithmetic or order package, or metric-completeness row can serve as an alternate route.*

Proof. The four faces are the faces of the dependency lattice in Definition 38.442. Apply the non-escape theorem, Definition 38.443, to exclude any route that leaves the dyadic, StreamName, RegSeqRat, and local Real seal sequence.

Next apply interface totality, Definition 38.444. It says that every current Phase Real and Completion interface read factors through exactly those four faces, using only the displayed hsame, Cont, Pkg, and NameCert rows. The terminal order theorem, Definition 38.445, fixes their order as $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$.

For comparisons of two accepted presentations, apply Definition 38.446. It confines every completion-facing comparison to the common BHist, hsame, Cont, Pkg, and NameCert data over the same ordered four faces.

The shared finite consumer surface is then the pullback surface of Definition 126.356 together with the observation triangle of Definition 38.384. These identify the dyadic observations, finite stream windows, regular-rational readback, and Real seal boundary before the consumer surface is read. With no exported coordinate for quotient streams, selected host limits, host real equality, countable choice, ambient completeness, arithmetic or order packages, or metric-completeness rows, no alternate route remains. □

Theorem 38.451 (Real StreamName RegSeqRat seal lattice)**.** *Every completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer that also carries a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row factors through the ordered four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row is only the compatibility row for the already displayed seal route; it does not add a second completion face, quotient stream equality, selected limit, host real equality, ambient completeness, or metric-completeness row.*

Proof. Start with Definition 38.442. Its source route is the ordered finite lattice of dyadic observation rows, finite StreamName windows, regular-rational classifier rows, and the local Real seal.

Apply Definition 38.450. Every current Phase Real or Completion consumer already has a canonical route through exactly those four faces, with only the common BHist, hsame, Cont, Pkg, and NameCert rows as structural data.

For completion-facing comparisons, use Definition 38.446. It keeps both terminal reads on the same four-face route. The shared finite surface is the pullback of Definition 126.356, so the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal boundary remain one displayed route.

The extra $\mathsf{CauchyLimitSeal}^{\uparrow}$ row is therefore read as the compatibility surface supplied by chapter 386. Since the canonical route and terminal comparison already exhaust the consumer surface, the seal row cannot add quotient stream equality, selected limits, host real equality, ambient completeness, or metric-completeness data. □

Theorem 38.452 (RealUp stationary partition normal-form boundary)**.** *Every stationary-rational partition normal-form read used by the Real-completeness diagonal route is accepted by* $\mathsf{Real}^{\uparrow}$ *only through the four displayed faces of Definition 38.442:*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The accepted read consists of DyadicRatCore observation rows, StreamName finite-window rows, RegSeqRat regular-rational row classifiers, and the local $\mathsf{Real}^{\uparrow}$ *seal row, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

In particular, the normal form of Definition 9238.16 may be consumed on the concrete $\mathsf{Real}^{\uparrow}$ *surface only through this four-face route. It supplies no coordinate for quotient stream equality, selected host limits, countable choice, ambient completeness, arithmetic or order packages, or metric-completeness rows.*

Proof. Start with the stationary normal-form theorem in Definition 9238.16. Its finite partition spine reduces every stationary segment ledger to a one-window classifier and then feeds the diagonal representatives through the stationary rational RealUp embedding route.

Apply the Real L10 four-face canonical route of Definition 38.450. Any current Phase Real or Completion consumer of that diagonal read must therefore pass through exactly the displayed DyadicRatCore, StreamName, RegSeqRat, and local RealUp faces, using only the common BHist, hsame, Cont, Pkg, and NameCert rows.

The seal compatibility is the seal lattice of Definition 38.451. It reads the completion-facing seal as compatibility for the already displayed StreamName and RegSeqRat handoff, not as a second completion face.

For comparisons between two stationary partition presentations, use Definition 38.446. It keeps both terminal reads on the same four-face route and leaves no comparison coordinate outside the DyadicRatCore observations, finite StreamName windows, RegSeqRat rows, and local RealUp seal.

Thus the Real-completeness stationary partition normal form lands on the concrete L10 lattice as a finite four-face certificate. The cited routes export no quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, arithmetic or order package, or metric-completeness row, so none of those data can be read from the accepted $\mathsf{Real}^{\uparrow}$ surface. □

Theorem 38.453 (Real L10 stationary diagonal route exhaustion)**.** *For any current Phase Real or Completion consumer, access to a stationary partition diagonal read is admissible exactly when the read factors through the displayed four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The admissible factorization consists of the DyadicRatCore observation face, the StreamName finite-window face, the RegSeqRat regular-rational face, and the local $\mathsf{Real}^{\uparrow}$ *seal face, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Moreover, every comparison between two admissible stationary diagonal reads remains on the same four-face route. The correspondence supplies no quotient stream equality, selected host limit, countable-choice row, ambient completeness theorem, trichotomy row, arithmetic package, order package, or metric-completeness row.

Proof. For the forward direction, start with an admissible stationary partition diagonal read. Apply Definition 9238.16 to put the stationary partition data in its finite normal form, with the diagonal representatives presented through the stationary rational $\mathsf{Real}^{\uparrow}$ embedding route.

Now apply Definition 38.450. The current Phase Real or Completion consumer must read that normal form through exactly the displayed DyadicRatCore, StreamName, RegSeqRat, and local $\mathsf{Real}^{\uparrow}$ faces, using only the common BHist, hsame, Cont, Pkg, and NameCert rows. The seal compatibility is then the lattice handoff of Definition 38.451, so the completion-facing seal is only compatibility for the same displayed StreamName and RegSeqRat handoff.

For the reverse direction, suppose a stationary diagonal packet is already factored through those four faces with the displayed coordinates. The same normal-form theorem supplies the stationary partition diagonal read, and the four-face canonical route repacks the displayed dyadic observations, finite StreamName windows, regular-rational classifiers, and local seal as an admissible current consumer read.

It remains to check comparisons. Apply Definition 38.446. Any comparison between two such stationary diagonal reads stays on the same ordered four-face route and has no comparison coordinate outside the displayed BHist, hsame, Cont, Pkg, and NameCert data.

The route therefore exhausts stationary diagonal admissibility. Since the cited normal form, canonical route, seal lattice, and terminal comparison theorem do not export quotient stream equality, selected host limits, countable choice, ambient completeness, trichotomy, arithmetic or order packages, or metric-completeness rows, none of those data are available through this correspondence. □

Theorem 38.454 (Real L10 diagonal budget four-face ownership)**.** *Any current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read carrying a diagonal-limit row or completion-facing budget row is accepted only when that row is displayed on the same finite packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The budget row is a consumer-side compatibility row over the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

It does not introduce quotient stream equality, selected host limits, countable-choice selectors, ambient completeness, metric-completeness rows, or a second diagonal budget face.

Proof. Start with the terminal $\mathsf{Real}^{\uparrow}$ read and apply Definition 38.457. The accepted terminal surface is identified with the finite packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, using only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next use Definition 38.713. It reads terminal-budget data through the same ordered four-face packet, so a completion-facing budget row cannot sit on a separate Real-side selector or on an additional metric-completeness face.

For the diagonal-limit row, apply Definition 646.75. That theorem places diagonal compatibility on the consumer side of the already displayed budget surface, not on quotient stream equality, selected host limits, countable choice, or ambient completeness.

Finally compare the resulting packet with the StreamName exit route by Definition 38.460. The DyadicRatCore observations, finite StreamName windows, RegSeqRat readback, and local $\mathsf{Real}^{\uparrow}$ seal remain the same terminal route. Hence the diagonal-budget row is owned by the existing four-face finite packet and no second diagonal budget face is exported. □

Theorem 38.455 (Real L10 terminal four-face correspondence)**.** *A terminal* $\mathsf{Real}^{\uparrow}$ *L10 read is accepted exactly when its visible rows factor through the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The factorization uses only displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *coordinates on those four faces.*

Equivalently, any four-face packet with those coordinates repacks as the current terminal $\mathsf{Real}^{\uparrow}$ *seal read. The correspondence exports no quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, arithmetic package, order package, or metric-completeness row.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10TerminalFourFaceCorrespondence`

Proof. For the forward direction, apply Definition 38.450. It routes every current terminal Real and Completion read through the ordered DyadicRatCore, StreamName, RegSeqRat, and Real faces of the dependency lattice.

Next use Definition 38.446. It says that terminal comparison has no surface outside the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates on the same ordered four faces. Thus an accepted terminal read has exactly the visible packet stated above.

For the reverse direction, start with a packet already factored through the four faces and the displayed coordinates. Apply Definition 38.451 to read the StreamName and RegSeqRat handoff as the terminal $\mathsf{Real}^{\uparrow}$ seal surface, with the DyadicRatCore observation ledger kept in the same lattice route.

The reverse repacking adds no hidden coordinate. Since the cited route, comparison, and seal theorems do not export quotient stream equality, selected host limits, countable choice, ambient completeness, arithmetic or order packages, or metric-completeness rows, the repacked terminal read is accepted precisely as the current Real seal read. □

Theorem 38.456 (Real L10 four-face no-completeness exactness)**.** *For every accepted current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read, the exact accepted surface is the four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442, together with displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows on those faces. The read contains no quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, metric-completeness row, or standard bridge datum.*

Proof. Start with the dependency lattice of Definition 38.442. It names the available terminal route as the ordered DyadicRatCore, StreamName, RegSeqRat, and local Real faces.

Apply Definition 38.450. Every accepted current Phase Real and Completion terminal read must factor through those four faces and through the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 38.451. The StreamName and RegSeqRat handoff is read only as seal compatibility for the same displayed route, with the DyadicRatCore observation face and the local $\mathsf{Real}^{\uparrow}$ face kept in the packet.

Finally apply Definition 38.455. The terminal correspondence repacks exactly that four-face surface as the accepted $\mathsf{Real}^{\uparrow}$ read and exports no quotient stream equality, selected host limit, countable choice, ambient completeness, metric-completeness row, or standard bridge datum. □

Theorem 38.457 (Real L10 four-face exactness)**.** *For every current Phase Real or Completion terminal* $\mathsf{Real}^{\uparrow}$ *read, acceptance is equivalent to factorization of its visible rows through the finite four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The packet is exactly the displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *coordinates on those four faces.*

Conversely, every finite packet with precisely those four faces and precisely those displayed coordinates determines the same accepted terminal $\mathsf{Real}^{\uparrow}$ *seal read. The exact boundary exports no quotient stream equality, selected host limit, host real equality, countable-choice selector, ambient completeness principle, arithmetic package, order package, metric-completeness row, or standard bridge row.*

Proof. For the forward direction, start with an accepted current Phase Real or Completion terminal read. Apply Definition 38.450 to route it through the ordered DyadicRatCore, StreamName, RegSeqRat, and local Real faces.

Next apply Definition 38.446. It confines terminal comparison to the common BHist, hsame, Cont, Pkg, and NameCert rows over the same ordered faces. Thus the accepted read supplies no visible coordinate beyond the finite packet in the statement.

For the reverse direction, suppose the visible rows already form such a four-face packet. Apply Definition 38.455 to repack the displayed rows as the terminal $\mathsf{Real}^{\uparrow}$ read, and use Definition 38.451 to read the StreamName and RegSeqRat handoff as the same local Real seal surface.

It remains to check exactness of the boundary. The non-escape theorem, Definition 38.443, excludes any route that leaves the displayed source lattice, while interface totality, Definition 38.444, says that every current Phase Real and Completion interface row is already accounted for by those displayed faces and coordinates. Therefore quotient stream equality, selected host limits, host real equality, countable choice, ambient completeness, arithmetic or order packages, metric-completeness rows, and standard bridge rows are not part of the accepted terminal boundary. □

Theorem 38.458 (Real L10 terminal pullback uniqueness)**.** *Let two current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *reads expose the same DyadicRatCore, StreamName, RegSeqRat, and local*

$\mathsf{Real}^{\uparrow}$ *faces, with the same displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *coordinates. Then the accepted consumer boundary of the two reads is the same finite four-face pullback surface*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The uniqueness claim is only about this finite boundary packet. It does not identify quotient streams, host real values, selected host limits, countable-choice selectors, ambient completeness principles, arithmetic or order packages, metric-completeness rows, or standard bridge rows.

Lean4 ✓ `BEDC.Derived.RealUp.RealL10TerminalPullbackUniqueness`

Proof. Apply Definition 38.457 to each terminal read. Both accepted reads are identified with the finite packet over DyadicRatCore, StreamName, RegSeqRat, and the local Real seal, using only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates.

The two identified packets have the same four faces and the same displayed coordinates by hypothesis. Apply Definition 38.455 to repack either packet as the terminal $\mathsf{Real}^{\uparrow}$ read; the repacking adds no hidden coordinate.

Use Definition 38.451 for seal compatibility. The StreamName and RegSeqRat handoff is the same local $\mathsf{Real}^{\uparrow}$ seal surface on both sides, so the consumer boundary is the common four-face pullback packet.

Finally apply non-escape and interface totality, Definition 38.443 and Definition 38.444. They keep the boundary inside the displayed source lattice and exhaust the current Phase Real and Completion interface rows, excluding quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, arithmetic or order packages, metric-completeness rows, and standard bridge rows. □

Theorem 38.459 (Real L10 terminal source lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal or completion-facing consumer in the L10 route is admitted exactly through the displayed source lock*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The lock consists of finite stream windows, dyadic endpoint and radius rows, the regular-sequence classifier handoff, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local* $\mathsf{Real}^{\uparrow}$ *seal. No quotient stream, host real equality, selected limit, ambient completeness principle, or hidden completion carrier is exported by this terminal route.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10TerminalSourceLock`

Proof. Start with the L10 dependency lattice of Definition 38.442. It displays the four source faces and fixes the structural rows to BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ before the local $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Use the RegSeqRat source lattice pointer through Definition 38.407. The regular-sequence classifier handoff is read only after the StreamName finite windows and the DyadicRatCore observation rows have been displayed.

Next apply the RealUp seal handoff and the four-face route, Definition 38.451 and Definition 38.450. They place every current seal or completion-facing consumer on the same ordered route through DyadicRatCore, StreamName, RegSeqRat, and the local Real seal.

Finally use Definition 38.446 and Definition 38.455. These exhaust terminal comparisons by the displayed finite windows, dyadic endpoint and radius data, regular readback, transport, continuation replay, package provenance, and local naming. Since none of the cited rows carries quotient

streams, host real equality, selected limits, ambient completeness, or a hidden completion carrier, the terminal source lock is exact. □

Theorem 38.460 (Real L10 StreamName exit route correspondence)**.** *A current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read is accepted by the Real L10 dependency lattice exactly when its visible finite packet is the same four-object exit route named by StreamName,*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Equivalently, the accepted packet is the displayed DyadicRatCore observation face, StreamName finite-window face, RegSeqRat regular-rational face, and local $\mathsf{Real}^{\uparrow}$ *seal face, with only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

This correspondence is only a paper-side exit surface. It exports no quotient stream equality, host sequence equality, selected host limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, metric-completeness row, arithmetic package, order package, or standard bridge row.*

Proof. For the forward direction, start with a current Phase Real and Completion terminal $\mathsf{Real}^{\uparrow}$ read accepted by the Real L10 dependency lattice. Apply Definition 38.457 to identify its visible rows with the finite four-face packet over DyadicRatCore, StreamName, RegSeqRat, and the local Real seal.

Next use the terminal source lock of Definition 38.459. It fixes the same terminal read to the displayed source route and prevents any additional completion carrier or host equality coordinate from entering the packet.

Identify this packet with the StreamName exit surface by Definition 126.500. The four objects are the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, and the support rows remain exactly BHist, hsame, Cont, Pkg, and local NameCert.

Finally apply Definition 126.506. It keeps the route as paper-side exit data only, so quotient stream equality, host sequence equality, selected host limits, countable choice, ambient Real completeness, metric completeness, arithmetic or order packages, and standard bridge rows are outside the accepted surface.

For the reverse direction, suppose the visible finite packet is already the StreamName four-object exit route with the displayed support rows. Apply Definition 126.500 to read it as the current Phase Real and Completion exit packet.

Repack that packet through Definition 38.455. The DyadicRatCore observations, StreamName finite windows, RegSeqRat readback, and local Real seal become the terminal $\mathsf{Real}^{\uparrow}$ L10 read, with no hidden coordinate added.

Use Definition 38.451 for the final seal compatibility. It reads the StreamName and RegSeqRat handoff as the same local $\mathsf{Real}^{\uparrow}$ seal surface, so the Real L10 lattice accepts precisely the route named by StreamName and nothing beyond the displayed rows. □

Theorem 38.461 (Real L10 StreamName classifier pullback correspondence)**.** *For any two current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *reads accepted by the Real L10 dependency lattice, the reads have the same Real L10 public boundary exactly when their StreamName four-object classifier pullback agrees on the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and on the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The correspondence is only over this finite public boundary. It does not identify quotient stream equality, host real equality, selected limits, countable-choice rows, completeness data, metric-completeness rows, arithmetic or order packages, or bridge rows.*

Proof. For the forward direction, assume the two terminal reads have the same Real L10 public boundary. Apply Definition 38.457 to each read to expose the DyadicRatCore, StreamName, RegSeqRat, and local Real faces together with the displayed support rows.

Use the boundary equality before any uniqueness claim. By Definition 38.446, a comparison between the two accepted terminal reads has no coordinate outside the displayed BHist, hsame, Cont, Pkg, and NameCert rows over the four named faces. Then Definition 38.455 repacks exactly those exposed rows as the terminal $\mathsf{Real}^{\uparrow}$ read and adds no hidden coordinate.

Therefore the two exposed finite packets have the same DyadicRatCore, StreamName, RegSeqRat, and local Real faces, and they agree componentwise on the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Now apply Definition 126.514. The StreamName classifier pullback is the same four-object public surface because its whole support is precisely that common finite packet.

For the reverse direction, assume the StreamName four-object classifier pullback agrees on the four faces and displayed rows. Apply Definition 126.514 to read this as equality of the public classifier surface, and then use Definition 126.504 to recover the terminal four-object packet.

Repack the recovered packet by Definition 38.455. It supplies the terminal $\mathsf{Real}^{\uparrow}$ boundary from the common DyadicRatCore observations, StreamName finite windows, RegSeqRat readback, local Real seal, and displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally use the seal lattice of Definition 38.451. The repacked boundary is only the local four-face seal surface, so the reverse direction does not claim host real equality, quotient streams, selected limits, ambient completeness, arithmetic or order packages, metric-completeness rows, or bridge rows. Hence agreement of the StreamName classifier pullback gives exactly agreement of the Real L10 public boundary. □

Theorem 38.462 (Real L10 public-boundary classifier reflection)**.** *For any two current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *reads accepted by the Real L10 dependency lattice, equality of their Real L10 public boundary is equivalent to equality of the four-face classifier packet over*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Thus the public Real boundary is precisely the classifier surface formed by those four faces and rows.*

The reflection exports no quotient stream equality, host equality, selected limit, countable-choice row, completeness datum, arithmetic or order package, metric-completeness row, or bridge row.

Proof. For the forward direction, assume the two terminal reads have the same Real L10 public boundary. Apply Definition 38.461 to identify this equality with agreement of the StreamName four-object classifier pullback on the displayed four faces and support rows.

Transport the agreement through Definition 126.514. The StreamName classifier is the same finite classifier packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the same BHist, hsame, Cont, Pkg, and NameCert coordinates.

For the reverse direction, suppose the four-face classifier packet agrees. Use Definition 126.514 to read that packet as equality of the StreamName public classifier surface, then apply Definition 38.458 to recover the unique terminal pullback boundary over the same four faces.

Finally apply Definition 38.461 in the reverse direction. The recovered terminal pullback is exactly the Real L10 public boundary, so equality of the classifier packet and equality of the public boundary determine each other.

The exclusion clauses in the cited Real L10 and StreamName pullback theorems are stable under this reflection. They rule out quotient streams, host equality, selected limits, countable

choice, completeness, arithmetic and order packages, metric-completeness rows, and bridge rows, so the public Real boundary has no coordinate beyond the stated four-face classifier surface. □

Theorem 38.463 (Real L10 terminal-window fusion section compatibility)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *reads accepted by the Real L10 dependency lattice. Suppose their* $\mathsf{StreamName}^{\uparrow}$ *finite windows agree after terminal-window fusion through a certified common refinement that also carries the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, and suppose the induced* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request sections agree after applying the terminal budget factorization to the same seal-facing request. Then $\mathcal{W}_L$ and $\mathcal{W}_R$ have the same Real L10 public boundary.*

Conversely, the Real L10 public boundary of such a terminal read determines exactly the fused $\mathsf{StreamName}^{\uparrow}$ *terminal-window packet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section surface, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger:*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

This compatibility exports no quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic or order packages, or bridge rows.

Proof. For the forward direction, first use Definition 126.623. The certified common refinement makes the two $\mathsf{StreamName}^{\uparrow}$ finite windows one fused terminal packet after transport of the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows.

Next use Definition 301.142. Applied to the common seal-facing finite request, it identifies the RegSeqRat classifier, regularity, readback, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows with the finite request source section for both reads.

These two packets have the same four faces and support rows, so apply Definition 126.514. The fused StreamName terminal packet, the DyadicRatCore endpoint and radius ledger, and the RegSeqRat finite-request section determine one StreamName four-object classifier pullback over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 38.461. Agreement of that StreamName classifier pullback is exactly agreement of the Real L10 public boundary, so the two terminal $\mathsf{Real}^{\uparrow}$ reads have the same public boundary.

For the reverse direction, start from a fixed Real L10 public boundary. By Definition 38.461, it determines the StreamName four-object classifier pullback and therefore the displayed faces and support rows, including the DyadicRatCore endpoint and radius ledger. Reading this classifier through Definition 126.514 recovers the public fused terminal-window surface with that ledger as one determined face.

It remains to isolate the RegSeqRat finite-request section. Apply Definition 301.142 to the seal-facing finite request already present in the public boundary. The terminal budget consumer can factor only through the displayed source section, so the RegSeqRat section surface is determined by the same boundary data.

The same exclusion lists are stable under each cited theorem. None of the constructions introduces quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic or order packages, or bridge rows; hence the Real L10 public boundary determines exactly the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, and the RegSeqRat section surface, with nothing beyond those faces. □

Theorem 38.464 (Real L10 finite-window witness determinacy)**.** *For any current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read accepted by the Real L10 dependency lattice, every finite-window witness consumed by the Real L10 public boundary is determined exactly by the fused* $\mathsf{StreamName}^{\uparrow}$ *terminal-window packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

If two presentations agree on those rows, then they have the same finite-window witness surface. Conversely, the public boundary recovers exactly those rows and no additional coordinate. The determinacy claim excludes quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic or order packages, and bridge rows.

Proof. Start with the terminal-window fusion section compatibility theorem, Definition 38.463. It identifies the Real L10 public boundary with the fused $\mathsf{StreamName}^{\uparrow}$ terminal-window packet, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section surface, and the local $\mathsf{Real}^{\uparrow}$ seal over the displayed support rows.

For two presentations agreeing on those rows, apply Definition 126.623. The common refinement makes their $\mathsf{StreamName}^{\uparrow}$ windows the same fused terminal packet after transport along the same endpoint and radius ledger.

Next apply Definition 301.142. The common seal-facing finite request factors through the same RegSeqRat finite-request source section, so the classifier, regularity, readback, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows are the same finite witness data.

Transport the resulting four-face packet through Definition 38.461. Agreement of the StreamName classifier pullback is exactly agreement of the Real L10 public boundary, so the two presentations have the same finite-window witness surface.

Conversely, begin with the Real L10 public boundary. The reverse direction of Definition 38.463 recovers the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, and the RegSeqRat finite-request section surface. The local $\mathsf{Real}^{\uparrow}$ seal and the displayed BHist, hsame, Cont, Pkg, and NameCert rows are part of the same boundary packet.

Finally use the exclusion clauses from Definition 38.463 and Definition 38.461. They keep quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic or order packages, and bridge rows outside the public boundary, so the finite-window witness surface is exactly the displayed four-face data. □

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FiniteWindowWitnessDeterminacy`

Lemma 38.465 (Real L10 source route foundation)**.** *Every terminal finite-window witness whose visible rows lie in the Real L10 dependency lattice is sourced by the ordered four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates. The foundation route introduces no host equality, quotient stream, selected limit, completeness datum, arithmetic or order package, or bridge row.*

Proof. First apply Definition 38.457. It identifies every accepted terminal read with the finite packet over DyadicRatCore, StreamName, RegSeqRat, and the local Real seal, using only the displayed support rows.

Next use Definition 38.455. The correspondence repacks precisely those displayed rows as the terminal $\mathsf{Real}^{\uparrow}$ read, so the four-face packet is already sufficient for the terminal witness.

Apply the terminal source lock of Definition 38.459. It fixes seal-facing and completion-facing consumers to the ordered source route before any witness is read from the public boundary.

Finally use Definition 38.464. The finite-window witness surface is determined by the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. None of these cited rows carries host equality, quotient streams, selected limits, completeness data, arithmetic or order packages, or bridge rows, so the source foundation is exactly the displayed four-face route. □

Theorem 38.466 (Real L10 source route exactness)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *finite-window witness admitted by the Real L10 dependency lattice is sourced exactly by the ordered rows*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

displayed in Definition 38.442, together with their visible BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *coordinates. This source route is the terminal finite-window face consumed by the L10 public boundary.*

No alternate route through host real equality, quotient streams, selected limits, metric completeness, ambient completeness, arithmetic packages, order packages, or standard bridge rows is available.

Proof. First expose the terminal finite-window face by Definition 38.465. Any accepted terminal witness is already sourced by the ordered four-face packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ seal, with only the displayed support coordinates.

Use the terminal source lock, Definition 38.459. It fixes every seal or completion-facing consumer to the displayed ordered route and excludes hidden completion carriers before any witness is read by the public boundary.

Finally apply Definition 38.464. The public boundary determines exactly the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Combining these clauses leaves no coordinate for host real equality, quotient streams, selected limits, metric completeness, ambient completeness, arithmetic or order packages, or standard bridge rows. Hence the source route for the terminal finite-window witness is exactly the displayed L10 lattice route. □

Theorem 38.467 (Real L10 StreamName-RegSeqRat completion route triangle)**.** *An accepted current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read has the same public boundary whether it is read through the StreamName finite-window classifier pullback, through the RegSeqRat finite-request section, or through the Real L10 source route. The three sides agree exactly on the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and on the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

This triangle exports no quotient stream equality, host real equality, selected limit, countable-choice row, completeness row, arithmetic or order package, metric-completeness row, or bridge row.

Proof. Read the StreamName side first. By Definition 38.461, agreement of the StreamName four-object classifier pullback is exactly agreement of the Real L10 public boundary over the displayed four faces and support rows.

Read the RegSeqRat side next. The finite-request section of Definition 301.142 is the same section used by Definition 38.463, so the regular-rational request contributes only its finite classifier, regularity, readback, transport, provenance, and local naming rows.

Now read the Real side. Definition 38.466 fixes every terminal finite-window witness to the ordered route $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$, and Definition 38.464 says the public boundary recovers exactly that finite-window witness surface.

The three readings therefore present one finite route packet. Since each cited side carries the same exclusion list, no quotient stream equality, host real equality, selected limit, choice row, completeness row, arithmetic or order package, metric-completeness row, or bridge row can be added by moving around the triangle. □

Theorem 38.468 (Real L10 dependency lattice exit totality)**.** *Every current Phase Real and Completion terminal read factors through the ordered dependency lattice*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of Definition 38.442. The factorization exposes the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows for the DyadicRatCore endpoint and radius ledger, the StreamName terminal-window packet, the RegSeqRat finite-request section, and the local Real seal.*

Thus the present terminal surface is exactly the four-face route witnessed by the Real L10 source route, finite-window determinacy, StreamName-RegSeqRat completion triangle, StreamName completion classifier pullback, and RegSeqRat terminal-budget factorization.

Proof. Begin with source route exactness, Definition 38.466. It places every terminal finite-window witness on the ordered four-face route and records the visible support rows.

Use finite-window witness determinacy and the route triangle. Theorems 38.464 and 38.467 identify the public terminal read with the fused StreamName window, DyadicRatCore ledger, RegSeqRat finite-request section, and Real seal.

Finally apply the StreamName classifier pullback and the RegSeqRat budget factorization, Definition 126.514 and Definition 301.142. They provide the two sibling readings of the same finite packet, so every current terminal read factors through the displayed lattice and no further terminal face is needed. □

Theorem 38.469 (Real L10 dependency lattice seal nonescape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal exported by the current Phase Real and Completion terminal surface carries only the ordered four-face lattice and its visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream, selected host limit, countable choice, host real equality, ambient completeness, arithmetic or order package, metric-completeness row, or bridge row.*

Proof. Read the seal through the exit totality theorem. The accepted terminal surface is already the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Use source route exactness and finite-window witness determinacy to fix the public boundary. These theorems leave no coordinate outside the four faces and their displayed support rows.

The route triangle, StreamName classifier pullback, and RegSeqRat terminal-budget factorization carry the same exclusion list on each side of the route. Since none of the cited packets has a coordinate for quotient streams, selected limits, choice, host equality, completeness, arithmetic, order, metric completeness, or bridge data, the exported seal cannot supply one. □

38.4 Real L10 current phase exit audit pullback

Theorem 38.470 (Real L10 four-face status lookup)**.** *Any accepted current Phase Real and Completion exit read through*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

reads four separate local closurestatus records. Each face contributes exactly the closurestatus, formalstatus, leantarget, and bridgestatus rows declared in its own local closurestatus block. The lookup cites those rows objectwise and does not cache their token values in the Real L10 exit audit section.

These four entries are objectwise face records. They are not an aggregate formalstatus token, an aggregate bridge row, or a route-level closurestatus record for the composite exit.

Proof. The four faces are the Real L10 faces named by Definition 38.442. The local records are read from the same concrete chapters that supply the face data: Definition 291.25 for $\mathsf{DyadicRatCore}^{\uparrow}$, Definition 126.311, Definition 126.314, and Definition 126.315 for $\mathsf{StreamName}^{\uparrow}$, Definition 301.22 for $\mathsf{RegSeqRat}^{\uparrow}$, and Definition 38.290 for $\mathsf{Real}^{\uparrow}$.

The cited theorem anchors locate the four local chapters, but this theorem does not copy their token values. The local closurestatus block for each face is the sole source for its closurestatus, formalstatus, leantarget, and bridgestatus rows.

By the two-axis convention of section 9467.2, a closure entry and a verification entry remain attached to the object whose closurestatus block contains them. Reading the four faces therefore produces four objectwise face records and does not create a fifth formalstatus token, route-level bridge row, or composite closurestatus record. □

Theorem 38.471 (Real L10 current phase exit audit pullback)**.** *For any accepted current Phase Real and Completion exit read through the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

of the $\mathsf{Real}^{\uparrow}$ *L10 dependency lattice, the audit boundary is exactly the product of the four local face records. Each face contributes exactly the closurestatus, formalstatus, leantarget, and bridgestatus rows declared in its own local closurestatus block, read by reference from that face and not restated as cached Real L10 status data.*

The exit packet contributes no fifth formal token, no aggregate bridge, no $\mathsf{Real}^{\uparrow}$ *completeness claim, no quotient stream, no selected limit, and no metric-completeness datum. Its only admissible audit read is the displayed four-face product, with each status entry imported by reference from the corresponding local closure record.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10CurrentPhaseExitAuditPullback`

Proof. The four faces are the faces named by Definition 38.442. Apply Definition 38.443 and Definition 38.444 to confine every accepted current Phase Real and Completion interface read to the displayed DyadicRatCore, StreamName, RegSeqRat, and local Real rows.

Next apply Definition 38.457. It identifies acceptance with the finite packet over exactly those four faces, carried only by the displayed BHist, hsame, Cont, Pkg, and local NameCert coordinates. The StreamName exit correspondence, Definition 38.460, names the same packet as the current Phase Real and Completion exit route.

It remains to read the audit status. Apply Definition 38.470. It reads the DyadicRatCore, StreamName, RegSeqRat, and RealUp closurestatus entries as four objectwise face records and excludes any aggregate formalstatus token or aggregate bridge row.

Therefore the exit audit is the product of the four cited local records. Since the exactness and correspondence theorems export no fifth face, no quotient stream, no selected limit, no Real completeness row, no metric-completeness row, and no aggregate bridge coordinate, the exit packet cannot contribute any additional audit datum. □

Theorem 38.472 (Real L10 four-face classifier pullback)**.** *Any accepted Phase Real and Completion classifier read over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

is exactly the pullback of the four displayed classifier faces:

1. *the dyadic endpoint, radius, and distance ledger;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail classifier;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal classifier.*

The pullback uses only BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support along the four faces. Conversely, agreement of these four classifier faces determines the same finite Real classifier route.*

This classifier pullback carries no quotient streams, no host real equality, no selected limits, no countable choice, no ambient completeness, no metric-completeness data, no standard bridge rows, and no aggregate formalstatus tokens.

Proof. Apply Definition 38.457. It presents the accepted Real L10 packet as finite data over the four displayed faces and leaves only the BHist, hsame, Cont, Pkg, and local NameCert coordinates as support for the classifier read.

Apply Definition 38.460. The StreamName finite-window route is the same route used by the current Phase Real and Completion exit packet, so the StreamName classifier face is not an extra carrier and does not introduce a quotient stream or selected limit.

Apply Definition 38.779. It states that classifier agreement across the dyadic ledger, finite StreamName windows, RegSeqRat regularity and common tails, and the local Real seal is exactly the sheaf gluing condition. Hence the four faces determine one finite Real classifier route, and any accepted route restricts back to those four faces.

Finally apply Definition 38.471. The audit row is read as the objectwise product of the same four faces. This keeps the classifier pullback aligned with the audit pullback while keeping the two rows independent: the classifier row does not create host real equality, countable choice, ambient completeness, metric-completeness data, standard bridge rows, or aggregate formalstatus tokens. □

Theorem 38.473 (Real L10 finite-window completion-exit factorization)**.** *Every accepted Phase Real and Completion finite-window exit read of a* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly the four current-phase faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization uses only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows on those faces: the dyadic ledger face, the finite StreamName window face, the RegSeqRat regularity and common-tail face, and the local Real seal face.*

Conversely, matching records on these four local faces, together with matching BHist, hsame, Cont, Pkg, *and* NameCert *support rows, determines the same accepted finite-window Phase Real and Completion exit route.*

The factorization refuses quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus tokens, and standard bridge rows.

Proof. First read the classifier side. Apply Definition 38.472. It identifies every accepted classifier read with the dyadic ledger, finite StreamName window, RegSeqRat regularity and common-tail, and local Real seal faces, carried only by the displayed BEDC support rows.

Next read the audit side. Apply Definition 38.471. The accepted Phase Real and Completion audit route is the product of the same four local face records, with status and bridge coordinates read objectwise from the face records rather than from a route-level cache.

Now compare this product with the completion interface. Apply Definition 38.726. It exhausts the L10 exit interface by exactly $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the same BHist, hsame, Cont, Pkg, and NameCert support rows.

For the converse direction, the four matching local records give the same classifier pullback and the same objectwise audit pullback. The two-axis roadmap discipline keeps the status rows attached to their own faces, so status agreement is used only to identify the face records and not as evidence for classifier agreement.

Finally apply Definition 38.730. If any one of the four faces is removed or replaced, the accepted finite-window exit route is not present. The exclusions named in the statement therefore remain off-table: no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness datum, aggregate formalstatus token, or standard bridge row supplies a substitute route. □

Theorem 38.474 (Real L10 four-face status-classifier square)**.** *Any accepted Phase Real and Completion packet over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

has a commuting status-classifier square. Its horizontal reads are the four classifier faces of Definition 38.472, and its vertical reads are the four local status records of Definition 38.470. Reading the classifier faces first and then reading status gives the same four objectwise records as reading the local status records first and then restricting them along the classifier faces.

The square adds no aggregate formalstatus token, no bridge row for the composite route, no quotient stream, no selected limit, and no completeness datum.

Proof. First locate the common source route. Apply Definition 38.443 and Definition 38.444. They confine every accepted packet to the same DyadicRatCore, StreamName, RegSeqRat, and RealUp faces and exhaust the current Phase Real and Completion interface rows.

Next read the classifier side. Apply Definition 38.472. It identifies the horizontal reads as the dyadic ledger, finite StreamName windows, RegSeqRat regularity and common-tail classifier, and the local Real seal classifier over that route.

Now read the status side. Apply Definition 38.470. It reads the closurestatus, formalstatus, leantarget, and bridgestatus rows from the four local face records, object by object.

Both readings use the same four source faces, so restricting the status read along the classifier faces returns the same four objectwise records as first forming the classifier pullback and then reading the local status records. The two-axis convention of section 9467.2 keeps classifier agreement

and verification status independent, so the square does not create an aggregate formalstatus token, bridge row, quotient stream, selected limit, or completeness datum. □

Theorem 38.475 (Real L10 face status non-collapse)**.** *Any current Phase Real and Completion audit read through the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *preserves the four local closure records as separate face reports. No conjunction of those records can be read as a new formalstatus token, a* bridgeChecked *aggregate, a* $\mathsf{Real}^{\uparrow}$ *completeness claim, or a standard bridge for the composite exit route.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceStatusNoncollapse`

Proof. Apply Definition 38.471. It identifies the current exit audit as the product of the four local face records and states that the exit packet contributes no fifth formal token or aggregate bridge.

The two-axis convention of section 9467.2 reads closure and verification status locally. A face report can therefore be cited as a face report, but the conjunction of face reports is not itself a further formalstatus entry.

Finally use the exactness and non-escape inputs named in the pullback proof: Definition 38.443, Definition 38.444, Definition 38.457, and Definition 38.460. They keep the composite exit route on the four displayed faces and provide no row for RealUp completeness or for a standard bridge over the product. □

Theorem 38.476 (Real L10 face Lean target non-collapse)**.** *Any accepted current Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *preserves each face's Lean-target row and bridge-status row as local records of that face. No* bridgeChecked *face can supply the Lean target or bridge row for another face, and the composite route creates neither a route-level Lean target nor an aggregate bridge row.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceLeantargetNoncollapse`

Proof. Apply Definition 38.471. It reads the accepted exit audit as the product of the four local face records over the displayed Real L10 dependency lattice.

Apply Definition 38.833. It keeps the closure and verification reports objectwise, so a bridgeChecked entry is a record of its own face rather than a credential for a different face.

Now project the target and bridge rows along the same exactness data used by the pullback. The source-route non-escape, interface-totality, four-face exactness, and StreamName exit-correspondence inputs provide only face-local Lean-target and bridge-status coordinates for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not an aggregate target row or an aggregate bridge row.

Since those inputs provide no fifth face, no route-level Lean-target coordinate, and no aggregate bridge coordinate, the local Lean-target and bridge-status rows remain attached to the faces from which they are read. □

Theorem 38.477 (Real L10 face status-target pairing)**.** *In any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, each face's* formalstatus*,* leantarget*, and* bridgestatus *rows are read as one local face record. A status row from one face cannot be paired with the Lean target or bridge row of another face, and no aggregate route row can repair such a mismatch.*

Proof. Apply Definition 38.471. The accepted audit is read as the product of the four local records carried by $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not as a single route credential.

Apply Definition 38.833. It keeps the formalstatus coordinate attached to the face whose closure record supplies it, so the status row cannot be detached and reused as a route-level token.

Apply Definition 38.834. It keeps the leantarget and bridgestatus coordinates attached to the same local face records and supplies no route-level target or aggregate bridge row.

Finally read the three rows through section 9467.2. The roadmap discipline keeps verification status, Lean target, and bridge status as reported coordinates of the displayed face, so a mixed record using one face's status and another face's target or bridge row is outside the accepted audit read. □

Theorem 38.478 (Real L10 face status product readback)**.** *Any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *reads exactly the four face-local triples* (formalstatus, leantarget, bridgestatus) *carried by the corresponding face records. The readback is supported only by the displayed rows*

$$\mathsf{BHist}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{hsame}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Cont}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{hsame}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Cont}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{hsame}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Cont}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{Real}^{\uparrow}}, \mathsf{hsame}_{\mathsf{Real}^{\uparrow}}, \mathsf{Cont}_{\mathsf{Real}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{Real}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}.$$

It does not create an aggregate formalstatus*, aggregate* leantarget*, aggregate bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceStatusProductReadback`

Proof. Apply Definition 38.833. It keeps the status coordinates as local reports of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and supplies no fifth status coordinate.

Apply Definition 38.834. It keeps the leantarget and bridgestatus rows attached to their source faces and supplies no route-level target or aggregate bridge row.

Apply Definition 38.835. It reads formalstatus, leantarget, and bridgestatus together on the same face, so the product readback is exactly the four displayed local triples rather than a pool of exchangeable coordinates.

Finally use section 9467.2. The roadmap keeps paper closure and verification status on separate axes, and it supplies no rule turning the four face-local triples or their support rows BHist, hsame, Cont, Pkg, NameCert into quotient stream equality, a selected limit, a countable-choice row, a metric-completeness datum, an aggregate credential, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.479 (Real L10 face axis independence)**.** *Any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *preserves four independent local coordinates on each face: paper closure, formalstatus, leantarget, and bridgestatus. No coordinate from one face promotes any coordinate of another face, and no conjunction of face coordinates creates route-level closure, route-level formalstatus, route-level leantarget, aggregate bridge status, quotient stream, selected limit, countable-choice row, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.471. The audit is read as a product of the displayed faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, over the Real L10 dependency lattice, with no additional route coordinate.

Apply Definition 38.833. It keeps the paper closure and formalstatus reports face-local and blocks any reading in which a conjunction of local status records becomes route-level closure, route-level formalstatus, an aggregate bridge, or a $\mathsf{Real}^{\uparrow}$ completeness claim.

Apply Definition 38.834. It keeps leantarget and bridgestatus records attached to their source faces and blocks any reading in which one face supplies the Lean target or bridge row for another face.

Finally use the two-axis discipline of section 9467.2. Paper closure, formalstatus, leantarget, and bridgestatus are reported as separate coordinates; the roadmap supplies no rule that turns their conjunction into a quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate bridge, or completed real-completeness certificate. □

Theorem 38.480 (Real L10 face-axis defect localization)**.** *Suppose an accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *does not preserve one of the four independent coordinates named in Definition 38.837. Then the defect is localized to at least one displayed face record or to at least one displayed support row among* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. *No quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate formalstatus, aggregate bridge row, route-level leantarget, or* $\mathsf{Real}^{\uparrow}$ *completeness claim can serve as the defect carrier.*

Proof. Apply Definition 38.837. The admissible coordinate inventory is the displayed facewise list: paper closure, formalstatus, leantarget, and bridgestatus over the four Real L10 faces. The same theorem excludes route-level completion vocabulary and aggregate status vocabulary from that inventory.

Apply Definition 38.833. It keeps paper closure and formalstatus attached to the local closure records of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Thus a defect in either status coordinate is witnessed on at least one displayed face record or on at least one of its displayed support rows $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$.

Apply Definition 38.834. It keeps leantarget and bridgestatus attached to their source faces and supplies no route-level target or aggregate bridge row. Thus a defect in either target coordinate is again witnessed on at least one displayed face record or on at least one displayed support row.

Finally use the two-axis discipline of section 9467.2. Paper closure, formalstatus, leantarget, and bridgestatus are reported as separate axes, and there is no rule that lets a quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate formalstatus, aggregate bridge row, route-level leantarget, or $\mathsf{Real}^{\uparrow}$ completeness claim replace the concrete face or support coordinate where the read failed. □

Theorem 38.481 (Real L10 exit defect face completeness)**.** *Every failed accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is exhausted by the four displayed face records and their displayed support rows* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. *Equivalently, any accepted defect witness is carried either by one of the four face records or by one of those support rows on that face. No quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus, route-level leantarget, aggregate bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim is an additional defect coordinate.*

Proof. Apply Definition 38.837. It supplies the complete coordinate inventory for an accepted audit read: paper closure, formalstatus, leantarget, and bridgestatus, each read locally over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.838. Once one of those coordinates fails to be preserved, the failure is witnessed on a displayed face record or on one of its displayed support rows $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$.

Apply Definition 38.836. The product readback contains exactly the four face-local status, target, and bridge triples supported by the displayed rows, so there is no remaining readback slot in which a hidden fifth status or completion coordinate could be stored.

Finally use the roadmap two-axis discipline of section 9467.2. It separates theory closure from verification status and provides no rule turning quotient-stream data, selected-limit data, countable-choice data, metric-completeness data, aggregate status data, route-level target data, aggregate bridge data, or $\mathsf{Real}^{\uparrow}$ completeness vocabulary into an accepted defect coordinate. □

Theorem 38.482 (Real L10 StreamName terminal status minimality)**.** *For any accepted Phase Real and Completion terminal read through* $\mathsf{StreamName}^{\uparrow}$*, the terminal read satisfies the roadmap exit status coordinate only if the four concrete face chapters* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each supply their own local closurestatus, formalstatus, leantarget, and bridgestatus record at the required local audit-clean grade. If any one face record is absent or below that grade, then no aggregate formalstatus, aggregate bridge token, quotient stream, selected limit, countable-choice row, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim repairs the terminal read.*

Proof. Apply Definition 126.425. The terminal $\mathsf{StreamName}^{\uparrow}$ read is forced to inspect the four concrete sibling status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not a read from a route-level status cache.

Now apply Definition 38.489. A missing face record, or a face record below the required audit-clean grade, blocks the current Phase Real and Completion exit criterion at the object where the defect is read.

Finally apply Definition 38.833 and Definition 38.834. These results keep the closurestatus, formalstatus, leantarget, and bridgestatus coordinates attached to their source faces and provide no aggregate replacement row. Therefore quotient-stream data, selected-limit data, countable-choice data, metric-completeness data, aggregate formalstatus, aggregate bridge tokens, and $\mathsf{Real}^{\uparrow}$ completeness vocabulary cannot repair the failed terminal read. □

Theorem 38.483 (Real L10 terminal four-face defect localization)**.** *Let* $\mathcal{T}$ *be an accepted terminal Phase Real and Completion read through* $\mathsf{StreamName}^{\uparrow}$*. If* $\mathcal{T}$ *fails the roadmap exit criterion, then the failed terminal read has a witness localized to a single displayed face among* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. At that face, the witness is read either from the local closurestatus, formalstatus, leantarget, and bridgestatus record, or from one of the displayed support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *attached to that face.*

The localization has no repair coordinate outside those four faces. Quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge tokens, and $\mathsf{Real}^{\uparrow}$ *completeness do not supply terminal repair data and do not move the witness away from the face where the failed read is displayed.*

Proof. Start with Definition 38.840. It forces every accepted terminal $\mathsf{StreamName}^{\uparrow}$ read to inspect the four local closurestatus, formalstatus, leantarget, and bridgestatus records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.839. A failed accepted exit read is exhausted by those four face records and by the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows on those faces. Hence some face carries the displayed witness.

Now use Definition 38.836. The terminal readback contains exactly the four face-local status, target, and bridge triples, so the witness can be chosen on one of the four objectwise faces rather than on a route-level aggregate.

Finally read the conclusion through section 9467.2. The roadmap separates paper closure, verification status, Lean target, and bridge status into local reported coordinates. It gives no rule that turns quotient stream equality, selected limits, countable choice, metric completeness, aggregate formalstatus, aggregate bridge data, or $\mathsf{Real}^{\uparrow}$ completeness into a terminal repair coordinate. □

Theorem 38.484 (Real L10 regular-Cauchy seal route completeness)**.** *Let $\mathcal{T}$ be an accepted Phase Real and Completion terminal read whose* $\mathsf{Real}^{\uparrow}$ *seal is supplied by an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *route. Then the accepted seal route of $\mathcal{T}$ is exhausted by exactly the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the visible support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *on those faces.*

No fifth route coordinate is exported. In particular, the accepted seal route does not read quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completion data, or standard bridge rows.*

Proof. Apply Definition 301.40. An accepted $\mathsf{RegSeqRat}^{\uparrow}$ route supplying the $\mathsf{Real}^{\uparrow}$ seal exports only the finite regular-source rows and excludes quotient stream equality, selected host limits, host real equality, hidden tail witnesses, and ambient completeness data.

Next use Definition 301.111. The $\mathsf{Real}^{\uparrow}$ consumer read factors through the same finite route rows, so the seal route seen by the terminal Phase Real and Completion read cannot acquire an additional consumer-side completion coordinate.

Use Definition 291.123 for the dyadic tolerance face. The tolerance component of the route is already read through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 38.833, Definition 38.834, and Definition 38.841. These results keep the L10 status, target, bridge, and defect witnesses localized to the four displayed faces and their support rows. Hence the accepted regular-Cauchy seal route has no fifth status coordinate, no independent completeness coordinate, and no standard bridge row outside the displayed four-face route. □

Theorem 38.485 (Real L10 current phase exit certificate)**.** *An accepted Phase Real and Completion exit certificate is exactly the four-face product*

$$\mathsf{DyadicRatCore}^{\uparrow} \times \mathsf{StreamName}^{\uparrow} \times \mathsf{RegSeqRat}^{\uparrow} \times \mathsf{Real}^{\uparrow}$$

read through the Real L10 source-route, interface-totality, four-face exactness, StreamName exit-correspondence, and audit-pullback records. Its certificate content is the cited local closure evidence: each face contributes exactly the closurestatus, formalstatus, leantarget, and bridgestatus rows declared in its own local closurestatus block.

The certificate exports no $\mathsf{Real}^{\uparrow}$ *completeness claim, no quotient stream, no selected limit, no metric-completeness row, no aggregate bridge, and no formalstatus token beyond the four local face reports.*

Proof. Apply Definition 38.471. It identifies the accepted Phase Real and Completion exit audit with the product of the four local face records over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The source-route and interface-totality inputs used there confine the exit read to those faces. The four-face exactness theorem gives the product shape, and the StreamName exit correspondence identifies the same product as the current Phase Real and Completion exit route.

Apply Definition 38.833. It says that the four face reports remain separate audit entries and cannot be fused into a further formalstatus token, aggregate bridge, Real completeness row, quotient stream, selected limit, or metric-completeness row.

Thus the accepted exit certificate is precisely the four-face product read through the named route and exactness records, with only the displayed local closure evidence exported. □

Theorem 38.486 (Real L10 common-tail exit correspondence)**.** *Let an accepted Phase Real and Completion packet be read through the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Assume that its* $\mathsf{RegSeqRat}^{\uparrow}$ *face satisfies the common-tail union classifier determinacy of Definition 301.181. Then the Real exit classifier is determined exactly by the common* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier, and the local* $\mathsf{Real}^{\uparrow}$ *seal row.*

This correspondence is only a four-face finite BEDC exit route. It exports no quotient streams, no host-selected limits, no countable choice, no ambient $\mathsf{Real}^{\uparrow}$ *completeness, no metric-completeness row, and no aggregate formalstatus token.*

Proof. First apply Definition 38.472. It reads an accepted Real L10 classifier route as the pullback of the dyadic ledger, finite StreamName windows, RegSeqRat regularity and common-tail classifier, and the local Real seal classifier.

Next apply Definition 38.460. The StreamName finite-window route is the same source route used by the current Phase Real and Completion exit packet, so the common window is a coordinate of the Real exit read rather than a separate carrier.

Now use Definition 301.181. On the $\mathsf{RegSeqRat}^{\uparrow}$ face, the common-tail window determines one accepted $\mathsf{RegSeqRatClassifier}$ read over the displayed StreamName schedule, dyadic endpoint and radius rows, regularity rows, Real seal-source rows, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ provenance.

Finally apply Definition 38.485. The accepted exit certificate exports exactly the four local face reports and no fifth coordinate. Combining that certificate with the classifier pullback leaves precisely the common StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat common-tail classifier, and local Real seal row as the data determining the Real exit classifier.

The exclusion clauses in the four cited theorems remove the other possible readings: quotient streams, host-selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness rows, and aggregate formalstatus tokens have no displayed row in the four-face route. □

Theorem 38.487 (Real L10 exit refinement stability)**.** *Let an accepted Phase Real and Completion packet be refined by an* hsame*-compatible finite request section. Suppose that the refinement preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary, and the local* $\mathsf{Real}^{\uparrow}$ *seal row, together with the displayed* BHist, Cont, Pkg, *and* $\mathsf{NameCert}$ *support for those four faces. Then the refined packet determines the same Real L10 exit classifier as the original accepted packet.*

Equivalently, the refined read repacks as the same four-face finite BEDC exit route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The stability assertion exports no quotient stream, no selected limit, no countable-choice selector, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim, no metric-completeness row, and no aggregate formalstatus token.*

Proof. First apply Definition 38.486. It says that the Real exit classifier is determined by exactly the common $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ common-tail classifier, and the local $\mathsf{Real}^{\uparrow}$ seal row.

The refinement is hsame-compatible and preserves those four rows. Hence it cannot alter the common-tail classifier read on the $\mathsf{RegSeqRat}^{\uparrow}$ face; this is the determinacy content of Definition 301.181.

Next apply Definition 126.624. The preserved terminal StreamName window, dyadic scale data, common-tail boundary, and local Real seal row are the same finite boundary before and after refinement.

Finally use Definition 38.775. It repacks the preserved DyadicRatCore, StreamName, RegSeqRat, and Real rows as one finite four-face public interface. Therefore the refined packet has the same Real L10 exit classifier and the same finite BEDC exit route as the original accepted packet.

The exclusion clauses in the cited correspondence, fusion, determinacy, and four-face interface theorems leave no place for a quotient stream, selected limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ completeness datum, metric-completeness row, or aggregate formalstatus token to enter the refinement read. □

Theorem 38.488 (Real L10 phase exit criterion is objectwise)**.** *The Phase Real and Completion exit criterion is satisfied only by reading the four local records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Each face must supply its own* theoryclosure, formalstatus, leantarget, *and* bridgestatus *rows from its local closurestatus block, and those rows remain attached to that face.*

The criterion admits no aggregate route-level formalstatus, *no aggregate bridge row, no* $\mathsf{Real}^{\uparrow}$ *completeness claim, no quotient stream, no selected limit, and no metric-completeness datum. In particular, satisfying the current phase exit criterion means presenting the four objectwise face records and nothing further as Real L10 exit status.*

Proof. Apply Definition 38.485. It gives the certificate shape as the four-face product over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with exactly the local closurestatus, formalstatus, leantarget, and bridgestatus rows exported by those faces.

Apply Definition 38.471. It identifies the admissible audit read with objectwise lookup from the four local closurestatus blocks, so the phase criterion is discharged by those local records rather than by cached Real L10 status data.

Apply Definition 38.833. It rules out an aggregate formalstatus token, aggregate bridge row, Real completeness row, quotient stream, selected limit, and metric-completeness datum for the composite exit route.

Finally apply Definition 38.834. The Lean-target and bridge-status rows do not cross from one face to another, and the composite route has no route-level Lean target or bridge coordinate. Therefore the Phase Real and Completion exit criterion is objectwise in the four displayed faces. □

Theorem 38.489 (Real L10 face audit minimum blocker)**.** *If any one of* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *lacks its own local audit-clean verification grade in its closurestatus record, then the four-face product cannot be read as satisfying the current Phase Real and Completion exit criterion.*

This blocker is objectwise. It creates no aggregate route-level formalstatus, *and it does not claim* $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, or metric completeness.*

Proof. Apply Definition 38.488. The current phase exit criterion is discharged only by the four local records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.471. It identifies the admissible audit read with lookup in those local closurestatus records, so a face whose own record is below the audit-clean verification grade leaves the four-face product without the required local coordinate.

Apply Definition 38.833. The missing local verification grade cannot be supplied by an aggregate formalstatus token for the route, and the route does not export Real completeness, quotient streams, selected limits, or metric-completeness data.

Finally apply Definition 38.834. No face transfers its Lean target or bridge-status row to another face, and the composite route has no route-level Lean target or bridge row. Thus any below-audit-clean face blocks satisfaction of the current Phase Real and Completion exit criterion. □

Theorem 38.490 (Real L10 local face-record witness extraction)**.** *For an accepted Phase Real and Completion packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *any claim that the packet satisfies the objectwise exit threshold is witnessed by four local face records. Each record carries its own theory-closure, formalstatus, leantarget, and bridgestatus rows, and the extraction is supported only by the displayed local data*

$$\mathsf{BHist}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{hsame}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Cont}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{hsame}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Cont}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{hsame}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Cont}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{Real}^{\uparrow}}, \mathsf{hsame}_{\mathsf{Real}^{\uparrow}}, \mathsf{Cont}_{\mathsf{Real}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{Real}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}.$$

If the threshold fails, at least one of these four face records or one of its displayed support rows is the local witness of failure.

This extraction creates no aggregate formalstatus*, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, no standard bridge completion, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.471. The accepted packet is read through the current audit pullback from the four displayed faces, so any claimed exit witness must be obtained from the local records of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.494. It identifies satisfaction of the current threshold with the four objectwise local records, each carrying its own theory-closure and audit-clean verification coordinates. Apply Definition 38.836 to read the formalstatus, leantarget, and bridgestatus rows together with the displayed BHist, hsame, Cont, Pkg, and NameCert support on the same face.

For the failure alternative, apply Definition 38.489. A missing or sub-threshold local audit row blocks the four-face product, and the blocker is localized at the responsible face record rather than at a route-level token.

Finally use the noncollapse exclusions cited by the threshold and readback results. They rule out aggregate formalstatus, aggregate bridge, quotient stream equality, selected limits, countable choice, metric completeness, standard bridge completion, and ambient $\mathsf{Real}^{\uparrow}$ completeness as replacement witnesses. Therefore the extracted witness, positive or blocking, is exactly one of the four local face records with its displayed support rows. □

Theorem 38.491 (Real L10 local face record readback)**.** *The four local closurestatus blocks for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *provide the objectwise status threshold required by the current Phase Real and Completion exit criterion. For each face, its own closurestatus block supplies the local theory-closure row and the local* formalstatus *row currently recorded there. The comparison with* auditClean *is made by reading those local records at the citation site, not by copying their token values into the Real L10 exit audit.*

This readback is only a four-face local status threshold. It contributes no aggregate route-level formalstatus*, no aggregate bridge row, no* $\mathsf{Real}^{\uparrow}$ *completeness claim, no quotient stream, no selected limit, no countable-choice coordinate, and no metric-completeness datum.*

Proof. Read the four local closurestatus blocks. The $\mathsf{DyadicRatCore}^{\uparrow}$ record is the block attached to Definition 291.25. The $\mathsf{StreamName}^{\uparrow}$ record is the public interface block attached to Definition 126.311 and Definition 126.314.

The remaining two faces are read in the same local way. The $\mathsf{RegSeqRat}^{\uparrow}$ record is the real-seal handoff block attached to Definition 301.22. The $\mathsf{Real}^{\uparrow}$ record is the phase-boundary block attached to Definition 38.290.

Apply Definition 38.471. The accepted current phase audit reads exactly these four local closurestatus blocks, not a cached Real L10 status row.

Apply Definition 38.488. Since each cited block is read through its own local theory-closure row and its own local formalstatus row, the required comparisons with scopedClosed and auditClean are made face by face. Those four local comparisons supply the objectwise threshold for the current Phase Real and Completion exit criterion.

Finally apply Definition 38.834. The four Lean-target and bridge-status rows remain attached to their own face records, so the readback does not create an aggregate formalstatus, aggregate bridge, Real completeness, quotient stream, selected limit, countable-choice, or metric-completeness row. □

Theorem 38.492 (Real L10 current phase exit witness)**.** *The displayed four-face product* $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *satisfies the Phase Real and Completion exit criterion whenever the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each supply the objectwise local theory-closure threshold and the audit-clean verification threshold. The witness is read through the four local face records, not through cached current face anchors or an aggregate Real L10 row.*

This witness contributes no aggregate route-level formalstatus*, no aggregate bridge row, no* $\mathsf{Real}^{\uparrow}$ *completeness claim, no quotient stream, no selected limit, and no metric-completeness datum.*

Proof. Apply Definition 38.470. It identifies the four local record positions of the displayed product $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$.

Apply Definition 38.494. Under the stated objectwise local theory-closure and audit-clean verification thresholds, the reverse direction supplies the Phase Real and Completion exit criterion for those four face records.

Finally apply Definition 38.834. The local Lean-target and bridge-status rows stay attached to their faces, and the product exports no route-level formalstatus, bridge row, Real completeness claim, quotient stream, selected limit, or metric-completeness datum. □

Theorem 38.493 (Real L10 four-face exit sufficiency)**.** *Suppose the four local closurestatus records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each display the local theory-closure row required by the current Phase Real and Completion criterion and an audit-clean verification row. Then the current Phase Real and Completion exit criterion is satisfied exactly as the four-face product over those records.*

This sufficiency is only the objectwise exit read. It supplies no aggregate route-level formalstatus*, no aggregate bridge row, no quotient stream, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.488. The criterion asks only for the four local records of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with each closure, verification, Lean-target, and bridge-status row kept on its own face.

Apply Definition 38.485. Under the stated local closure and audit-clean verification rows, the accepted exit certificate is the four-face product over exactly those records. Apply Definition 38.471 to read the status coordinates from the four local closurestatus blocks rather than from cached Real L10 route data.

It remains to exclude non-objectwise witnesses. Apply Definition 38.833. It rules out an aggregate formalstatus token, aggregate bridge row, Real completeness row, quotient stream, selected limit, and metric-completeness datum for the composite route.

Finally apply Definition 38.834. The Lean-target and bridge-status rows remain attached to their source faces and do not assemble a route-level target or bridge. Since none of the cited objectwise records supplies a countable-choice row either, the sufficiency claim is exactly the four-face product and nothing further. □

Theorem 38.494 (Real L10 objectwise exit threshold equivalence)**.** *An accepted Phase Real and Completion exit packet satisfies the current exit criterion if and only if the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *each supply the required local theory-closure row and an audit-clean verification row, read objectwise from that face's own closurestatus block.*

This equivalence is exactly the current four-face threshold. It excludes any aggregate route-level formalstatus*, aggregate bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, and* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10ObjectwiseExitThresholdEquivalence`

Proof. For the forward direction, apply Definition 38.488. It says that an accepted packet can satisfy the current Phase Real and Completion exit criterion only through the four local records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.491. The required threshold is read from each face's own closurestatus block, so the local theory-closure row and the local audit-clean verification row are the coordinates that witness the criterion. Apply Definition 38.489; if any one of those local audit-clean rows is absent, the four-face product does not satisfy the current exit criterion.

To keep the witness facewise, apply Definition 38.833. It rules out aggregate route-level formalstatus, aggregate bridge rows, quotient streams, selected limits, countable-choice rows, metric-completeness data, and $\mathsf{Real}^{\uparrow}$ completeness. Apply Definition 38.834 to keep Lean-target and bridge-status rows attached to their source faces.

For the reverse direction, suppose the four local face records each supply the required local theory-closure row and an audit-clean verification row. Apply Definition 38.493. Those four objectwise records satisfy the current Phase Real and Completion exit criterion exactly as the four-face product over the local records.

Finally apply Definition 38.833 and Definition 38.834. The resulting criterion is still the facewise threshold: no aggregate route-level verification token, bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness claim is introduced. □

Theorem 38.495 (Real L10 component bridge latency)**.** *An accepted Phase Real and Completion packet satisfies the current four-face exit threshold exactly when the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *local records each provide the required scoped-closure row and at least* auditClean *verification. The component bridge rows may separately stand at paperBridge or bridgeChecked, as in the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *closurestatus anchors, but none of those component bridge rows supplies an aggregate* bridgeChecked *route credential.*

Consequently a paperBridge component row does not block the current Phase Real and Completion threshold once the local scoped-closure and audit-clean rows are present, and a bridgeChecked component row does not promote the packet to a global bridge-checked Real L10 route.

Proof. For the threshold equivalence, apply Definition 38.494. It identifies the accepted packet's current exit criterion with the four local records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, each read through its own scoped-closure and audit-clean verification rows.

The local records are the displayed component anchors. Apply Definition 38.491. The $\mathsf{DyadicRatCore}^{\uparrow}$ row is read at Definition 291.25, the $\mathsf{StreamName}^{\uparrow}$ row at Definition 126.311 and Definition 126.314, the $\mathsf{RegSeqRat}^{\uparrow}$ row at Definition 301.22, and the $\mathsf{Real}^{\uparrow}$ row at Definition 38.290.

It remains to separate component bridge latency from the phase threshold. Apply Definition 38.833. The component formalstatus and bridgestatus rows stay face-local, so a *paperBridge* row is not a missing audit-clean row and a *bridgeChecked* row is not an aggregate bridgeChecked route credential.

Finally apply Definition 38.834. The Lean-target and bridge-status rows remain attached to their source faces, so the packet gains no global bridge route beyond the objectwise threshold already supplied by the four local scoped-closure and audit-clean rows. □

Theorem 38.496 (Real L10 boundary-face bridge locality)**.** *Let an accepted Phase Real and Completion packet satisfy the objectwise exit threshold of Definition 38.494. If one component bridge row of* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or the same-route* RealCompletionExactBoundaryUp *boundary face is replaced by a stronger local bridge record, and if the replacement remains attached to that face, then the Real L10 exit read is unchanged. The replacement creates no aggregate* bridgeChecked *route credential, no fused standard bridge token, no quotient stream, no selected limit, no host-completeness row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the four roadmap faces, apply Definition 38.495. It separates the local *paperBridge* and *bridgeChecked* component rows from the current four-face exit threshold, so strengthening one displayed bridge row does not change the local scoped-closure and audit-clean records that the Real L10 gate reads.

For the five-face boundary packet, apply Definition 1327.21. It reads $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the same-route RealCompletionExactBoundaryUp face as local factors of the accepted packet, and it keeps the stronger bridge record attached only to the face where it is supplied.

Next apply Definition 1327.23 and Definition 38.834. These results exclude a route-level bridge leakage: the local replacement cannot become an aggregate bridgeChecked credential, a fused standard bridge token, a quotient stream, a selected limit, a host-completeness row, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Finally apply Definition 38.494. The exit read is therefore exactly the same objectwise Phase Real and Completion threshold read as before the component bridge replacement. □

Theorem 38.497 (Real L10 threshold open-phase nonescape)**.** *If a current Phase Real and Completion packet satisfies the objectwise threshold of Definition 38.494, then it licenses exactly the current four-face exit read over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The same threshold does not assert that the phase is globally closed, does not provide $\mathsf{Real}^{\uparrow}$ *completeness, does not add metric completeness, and does not create aggregate* formalstatus*, bridge, selected-limit, quotient-stream, or countable-choice coordinates.*

Proof. Apply Definition 38.494. The threshold is exactly the local theory-closure and audit-clean verification row read from each of the four face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.488. A packet that meets this threshold is read as the objectwise four-face exit certificate, so the licensed content is the current face product and not an additional route-level witness.

The boundary is sharp. By Definition 38.489, any missing or sub-threshold face obstructs the current exit criterion; hence the theorem is not a completion principle that could bypass a local face audit.

Finally apply Definition 38.833 and Definition 38.834. The status, target, and bridge coordinates remain attached to their source faces, so the threshold does not create a global phase closure claim, $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate formalstatus, aggregate bridge, selected-limit, quotient-stream, or countable-choice coordinate. □

Theorem 38.498 (Real L10 finite-cover exit threshold gluing)**.** *Let $\mathcal{U}$ be a finite family of accepted Phase Real and Completion exit packets. Suppose every member of $\mathcal{U}$ satisfies the objectwise exit threshold of Definition 38.494, and suppose that every pairwise overlap agrees on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the family glues to one accepted four-face Phase Real and Completion exit packet satisfying the same objectwise exit threshold.*

The glued exit packet is only the finite four-face packet over those displayed rows. It exports no aggregate route-level formalstatus*, no aggregate bridge row, no quotient stream, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.494 to each member of $\mathcal{U}$. Thus every local packet contributes exactly the four face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the threshold read from the local closurestatus rows of those faces.

We first glue the regular-sequence and real-seal faces. Apply Definition 301.88 to the finite family of $\mathsf{RegSeqRat}^{\uparrow}$ regularity rows and $\mathsf{Real}^{\uparrow}$ seal rows. The overlap agreement hypothesis is precisely the shared regularity, classifier, seal, BHist, hsame, Cont, Pkg, and NameCert surface required for that gluing. Apply Definition 301.80 to identify the resulting seal row on common overlaps.

Next glue the scheduling face. Use Definition 126.240 and Definition 126.265 to pass to a common finite-window presentation of the $\mathsf{StreamName}^{\uparrow}$ schedules. Apply Definition 126.562 to export one schedule row for the finite family, still over the displayed finite-window data and not over a quotient stream or selected limit.

The dyadic face is fixed by the overlap ledger. Apply Definition 291.66 and Definition 291.70. These read the shared endpoint and radius data through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, so the glued packet has the same dyadic endpoint and radius rows on all refined windows.

Combining the glued RegSeqRat/Real, StreamName, and DyadicRatCore rows gives one accepted four-face exit packet by Definition 38.493. Finally apply Definition 38.833 and Definition 38.834. The glued packet remains objectwise: its closure, verification, Lean-target, and bridge rows stay on their source faces and do not form an aggregate route-level status, bridge, quotient, selected-limit, choice, metric-completeness, or $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 38.499 (Real L10 finite-cover exit threshold descent)**.** *Let P be an accepted Phase Real and Completion exit packet, and let $\mathcal{V}$ be a finite certified subcover of P. Suppose every member of $\mathcal{V}$ satisfies the objectwise exit threshold of Definition 38.494. Suppose moreover that each subcover member agrees with P on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then P satisfies the same objectwise exit threshold.*

The descended threshold is only the coarse four-face threshold of P. It does not assert an aggregate route-level formalstatus*, an aggregate bridge row, a quotient stream, a selected limit, a countable-choice row, a metric-completeness datum, or a* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.843 to the finite certified subcover $\mathcal{V}$. The hypotheses on the local threshold and on the shared $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows give one glued accepted four-face packet satisfying the objectwise exit threshold.

We now identify this glued packet with the coarse packet P. The agreement assumptions say that every displayed endpoint, radius, finite-window, regularity, classifier, seal, history, sameness, continuity, package, and NameCert coordinate of the glued packet is the corresponding coordinate of P. Apply Definition 38.494 to read the glued threshold as the four local face records of P.

It remains to keep the descent facewise. Apply Definition 38.833 and Definition 38.834. These prevent the subcover gluing from creating an aggregate status, bridge, quotient-stream, selected-limit, countable-choice, metric-completeness, or $\mathsf{Real}^{\uparrow}$ completeness reading. Hence the coarse packet P carries exactly the same objectwise exit threshold and no route-level strengthening. □

Theorem 38.500 (Real L10 terminal pullback coherence)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read presented through the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route. Suppose its StreamName window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise seal and readback,* $\mathsf{Real}^{\uparrow}$ *seal row, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support agree with a four-face Phase Real and Completion packet P. Then T satisfies the same objectwise exit threshold as P in the sense of Definition 38.494.*

Conversely, any terminal finite-window $\mathsf{Real}^{\uparrow}$ *read satisfying that objectwise threshold and carrying the displayed StreamName, DyadicRatCore, RegSeqRat, Real seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows factors through the StreamName–*RegSeqRat*–*$\mathsf{Real}^{\uparrow}$ *terminal pullback over the four-face Phase Real and Completion packet.*

The coherent terminal read is only this displayed finite-window pullback. It does not assert an aggregate route-level formalstatus*, an aggregate bridge row, a quotient stream, a selected limit, a countable-choice row, a metric-completeness datum, or a* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First read the terminal route. Apply Definition 126.114. The accepted terminal $\mathsf{Real}^{\uparrow}$ read factors through exactly the displayed StreamName window rows, accepted $\mathsf{RegSeqRat}^{\uparrow}$ pointwise seal rows, $\mathsf{Real}^{\uparrow}$ finite-window seal boundary, hsame transports, finite Cont ledger, bundle membership, observations, and Pkg provenance. Apply Definition 126.125 to include the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger in the same finite route.

Now compare the resulting route with the four-face Phase Real and Completion packet P. The agreement assumptions identify the StreamName window, DyadicRatCore tolerance, RegSeqRat seal and readback, Real seal, BHist, hsame, Cont, Pkg, and NameCert coordinates of the terminal read with the corresponding objectwise faces of P. Apply Definition 38.494 to read the exit threshold from those four local closurestatus records.

For the converse direction, start with a terminal finite-window $\mathsf{Real}^{\uparrow}$ read that already carries the objectwise threshold and the displayed finite route rows. Apply Definition 126.125 to recover the finite StreamName–RegSeqRat route, and then apply Definition 126.114 to repack the same rows as the terminal pullback boundary.

It remains to keep the terminal comparison at the same threshold. Apply Definition 38.844 to descend any finite-window cover presentation back to the coarse packet P. Finally apply Definition 38.833 and Definition 38.834. These keep formalstatus, bridge, Lean-target, quotient-

stream, selected-limit, choice, metric-completeness, and $\mathsf{Real}^{\uparrow}$ completeness data attached only to the source faces, so the pullback coherence gives no route-level strengthening. □

Theorem 38.501 (Real L10 terminal cover coordinate exhaustion)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read over the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route. A finite certified cover witnesses the current Phase Real and Completion threshold for T if and only if every cover member and every certified overlap exposes exactly the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

No aggregate formalstatus, *aggregate bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can stand in for any one of these exposed coordinates.*

Proof. For the forward direction, start with a finite certified cover witnessing the terminal threshold for T. Apply Definition 38.845 to identify the terminal read with the four-face Phase Real and Completion packet through the displayed StreamName, DyadicRatCore, RegSeqRat, local Real, and shared support rows.

Apply Definition 38.494. The threshold is objectwise, so each cover member must present the dyadic endpoint and radius, finite-window schedule, regularity and classifier rows, local seal rows, and the common BHist, hsame, Cont, Pkg, and NameCert support. Apply Definition 38.843; the certified overlaps are precisely the rows along which those objectwise faces glue.

For the reverse direction, suppose every member and overlap exposes exactly those coordinates. Apply Definition 38.843 to assemble the memberwise four-face packets over the shared support, and then apply Definition 38.844 to read the glued packet back as the terminal finite-window threshold for T.

It remains to show that no route-level coordinate can replace a displayed face. Apply Definition 38.833 and Definition 38.834. These keep aggregate status, bridge, quotient-stream, selected-limit, countable-choice, metric-completeness, and $\mathsf{Real}^{\uparrow}$ completeness rows outside the finite cover support, so the witness is exhausted by the listed coordinates and by no stronger route credential. □

Theorem 38.502 (Real L10 terminal finite-cover threshold equivalence)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read over the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route, with displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and shared* BHist, hsame, Cont, Pkg, *and* NameCert *rows. Then T satisfies the current Phase Real and Completion objectwise exit threshold if and only if every finite certified cover of the displayed terminal route glues and descends to the same four-face packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal, with the shared* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

The equivalence refuses aggregate route-level formalstatus, *aggregate bridge rows, quotient stream equality, selected limits, countable choice, metric-completeness data, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, apply Definition 38.845. The accepted terminal read is compared with the four-face Phase Real and Completion packet through the displayed StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback and classifier rows, local Real seal, and shared support rows.

Now apply Definition 38.843. Each finite certified cover member carries the objectwise threshold read from the same local faces, and the overlap hypotheses are exactly the shared BHist, hsame,

Cont, Pkg, and NameCert rows. Apply Definition 126.504 to keep the glued cover inside the terminal four-object packet over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

For the reverse direction, suppose every finite certified cover of the displayed terminal route glues and descends to the same four-face packet. Apply Definition 38.844. The descended coarse packet carries the same objectwise threshold as its finite cover members.

It remains to identify this descended packet with the accepted terminal route. Apply Definition 291.107 to read the dyadic endpoint, radius, distance, and local NameCert interface as the DyadicRatCore$^{\uparrow}$ face of the same support packet. Apply Definition 301.93 to read the StreamName, RegSeqRat, DyadicRatCore, and local Real rows as one audit-clean four-face exit surface.

Finally apply Definition 38.845. The terminal read and the descended four-face packet have the same displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert support rows, so the threshold is exactly the current objectwise Phase Real and Completion threshold. The cited noncollapse clauses in the gluing, descent, interface, readiness, and terminal-pullback results leave no aggregate status, bridge row, quotient stream equality, selected limit, choice principle, metric-completeness datum, or ambient Real$^{\uparrow}$ completeness coordinate. □

Theorem 38.503 (Real L10 finite-cover terminal support exhaustion)**.** *Let T be an accepted terminal finite-window* Real$^{\uparrow}$ *read over the public* StreamName$^{\uparrow}$–RegSeqRat$^{\uparrow}$ *route. A finite certified cover witnesses the current Phase Real and Completion threshold for T if and only if its members and certified overlaps expose exactly the* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledger, the* StreamName$^{\uparrow}$ *finite-window schedule, the* RegSeqRat$^{\uparrow}$ *regularity and classifier rows, the local* Real$^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Deleting any one of those displayed rows prevents the altered cover from witnessing the same terminal threshold. No aggregate formalstatus *row, bridge row, quotient stream, selected limit, countable-choice datum, metric-completeness datum, or* Real$^{\uparrow}$ *completeness row replaces the deleted coordinate.*

Proof. For the equivalence of the support inventory with the terminal threshold, combine Definition 38.846 with Definition 38.847. The first result identifies the exact member and overlap coordinates of an accepted terminal cover, while the second reads those coordinates as the current Phase Real and Completion finite-cover threshold for T.

Now delete one displayed row. If the deleted row lies on the RegSeqRat$^{\uparrow}$ audit-threshold side, apply Definition 301.231; the altered cover cannot descend to the same glued audit-threshold read. If the deleted row is one of the shared cover-support rows along the public route, apply Definition 126.665; the cover-descent terminal support packet cannot present the same Real$^{\uparrow}$ seal-visible source.

It remains to keep the dyadic endpoint and radius ledger from being hidden in a different terminal readback. Apply Definition 291.124. The public tolerance read is determined by the finite DyadicRatCore$^{\uparrow}$ ledger, the StreamName window, the RegSeqRat rows, the local Real seal, and the same shared support rows, so deleting any one of those rows removes a coordinate used by the terminal threshold.

Finally use the noncollapse clauses cited in Definition 38.846 and Definition 38.847. They keep aggregate status, bridge data, quotient streams, selected limits, choice, metric-completeness, and ambient Real$^{\uparrow}$ completeness outside the objectwise finite-cover support. Hence none of those rows can replace the deleted coordinate, and the original finite certified cover is support-exhaustive. □

Theorem 38.504 (Real L10 terminal finite-cover face minimality)**.** *Let T be an accepted terminal finite-window* Real$^{\uparrow}$ *read satisfying the current Phase Real and Completion objectwise exit threshold, with displayed* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledger,* StreamName$^{\uparrow}$ *finite-window schedule,*

RegSeqRat$^{\uparrow}$ *regularity and classifier rows, local* Real$^{\uparrow}$ *seal rows, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then every finite certified cover witnessing the terminal finite-cover threshold equivalence for T is face-minimal in the following sense.*

1. *The cover must retain the* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and local* Real$^{\uparrow}$ *faces on the displayed terminal support.*
2. *The cover must retain the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows through the same common support.*
3. *Deleting any one of these faces or support rows makes the resulting subcover inadmissible as a witness for the terminal threshold.*
4. *No aggregate* formalstatus*, bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or ambient* Real$^{\uparrow}$ *completeness row can replace the deleted face or support row.*

Proof. Apply Definition 38.847. The accepted threshold for T is equivalent to gluing and descent of finite certified covers to the same four-face packet over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and the local Real$^{\uparrow}$ seal, with the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 38.845. This identifies the accepted terminal read with that four-face packet, so a cover witnessing the threshold is not merely a route credential: it is a cover whose visible support contains the dyadic ledger, the StreamName finite window, the RegSeqRat classifier and readback rows, the local Real seal, and the shared support rows.

It remains to rule out removing a displayed face. Use Definition 126.407 for the StreamName finite-window subcover, use Definition 301.198 for the RegSeqRat common-tail and cocycle rows, and use Definition 291.107 for the dyadic endpoint, radius, distance, and local NameCert interface. These results make each face visible in the common support consumed by the terminal packet.

Finally apply the noncollapse clauses from Definition 38.847, Definition 38.833, and Definition 38.834. They keep route-level status, bridge, quotient-stream, selected-limit, choice, metric-completeness, and ambient completeness data outside the objectwise face inventory, so none of those rows can compensate for a missing DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, local Real$^{\uparrow}$, or shared support row. □

Theorem 38.505 (Real L10 finite-cover threshold cocycle uniqueness)**.** *Let T be an accepted terminal finite-window* Real$^{\uparrow}$ *read. Let $\mathcal{U}$ and $\mathcal{V}$ be two finite certified covers of T, each satisfying the terminal finite-cover threshold equivalence. Suppose that, on every common refinement of $\mathcal{U}$ and $\mathcal{V}$, the two covers agree on the displayed* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledger, the* StreamName$^{\uparrow}$ *finite-window schedule, the* RegSeqRat$^{\uparrow}$ *regularity and classifier rows, the local* Real$^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then the four-face exit packet glued from $\mathcal{U}$ and the four-face exit packet glued from $\mathcal{V}$ are the same terminal common-refinement packet. This uniqueness statement is only about the displayed finite-cover threshold surface. It does not identify aggregate formalstatus *rows, bridge rows, quotient streams, selected limits, countable-choice data, metric-completeness data, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. First apply Definition 38.847 to both covers. Each cover is therefore a finite certified presentation whose glued packet and descended packet are the terminal four-face exit over the displayed DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and local Real$^{\uparrow}$ faces with the same support rows.

Next pass to any common refinement. Apply Definition 126.628 to the finite-window schedule and Definition 301.243 to the RegSeqRat request sections and local Real seal rows. The refinement exposes a single terminal four-face packet on the common StreamName, RegSeqRat, dyadic, and Real boundary.

The dyadic ledger cannot split into two sources on that refinement. Apply Definition 291.114; the common endpoint and radius rows determine the same BEDC-visible dyadic source ledger and hence the same dyadic face of the exit packet.

Finally apply Definition 38.845 and Definition 38.849. Pullback coherence identifies the common four-face packet with the accepted terminal read T, and face minimality prevents any alternative support row from standing in for a displayed face. Thus the two glued packets coincide as the same terminal common-refinement packet, while aggregate status, bridge, quotient-stream, selected-limit, choice, metric-completeness, and ambient completeness coordinates remain outside the construction. □

Theorem 38.506 (Real L10 terminal finite-cover refinement invariance)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read. Let $\mathcal{V}$ be a finite certified refinement of a finite certified cover $\mathcal{U}$ of T. Suppose that every refinement cell preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then $\mathcal{U}$ witnesses the current Phase Real and Completion terminal finite-cover threshold for T if and only if $\mathcal{V}$ witnesses that same threshold. The equivalence is an objectwise four-face statement: it does not assert equality of aggregate formalstatus *rows, bridge rows, quotient streams, selected limits, choice data, metric-completeness data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Assume first that $\mathcal{U}$ witnesses the terminal finite-cover threshold. Apply Definition 38.847 to $\mathcal{U}$. The cover therefore presents the terminal four-face exit packet over the displayed dyadic, StreamName, RegSeqRat, and local Real faces with the shared support rows.

Transport this packet across the certified refinement. Apply Definition 126.407 to keep the finite-window schedule exact on the terminal subcover, and apply Definition 301.198 to keep the RegSeqRat classifier, readback, and cocycle rows on the common tail. The dyadic endpoint and radius ledger is fixed by Definition 291.114.

Now apply Definition 38.843. Each cell of $\mathcal{V}$ carries the same displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert rows as its parent cell in $\mathcal{U}$, so the refined cells glue to the same terminal exit packet. Then Definition 38.847 reads the glued refined packet as a witness for $\mathcal{V}$.

For the reverse direction, assume that $\mathcal{V}$ witnesses the threshold. Apply Definition 38.844 along the certified refinement map from $\mathcal{V}$ to $\mathcal{U}$. The preservation hypotheses make the descended packet carry exactly the displayed dyadic, StreamName, RegSeqRat, local Real, and shared support rows of $\mathcal{U}$.

Finally apply Definition 38.849 and Definition 38.850. Face minimality prevents the refinement from hiding or replacing any displayed face, and cocycle uniqueness identifies the refined and descended packets on their common terminal support. Thus Definition 38.847 certifies $\mathcal{U}$ exactly when it certifies $\mathcal{V}$.

The construction never promotes route-level data into an objectwise face. Aggregate formalstatus rows, bridge rows, quotient streams, selected limits, choice principles, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the refined terminal finite-cover threshold. □

Theorem 38.507 (Real L10 refinement-support descent noescape)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read. Let $\mathcal{V}$ be a finite certified refinement of a finite certified cover $\mathcal{U}$ of T. Suppose the refinement preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows.*

Then descent from $\mathcal{V}$ to $\mathcal{U}$ preserves exactly this support surface. It introduces no aggregate $\mathsf{formalstatus}$ *row, bridge row, quotient stream, selected limit, choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 38.851. The terminal threshold witnessed on the refined cover is equivalent to the terminal threshold witnessed on the original cover, and the equivalence is already restricted to the objectwise dyadic, StreamName, RegSeqRat, local Real, and shared BEDC support rows.

The descent keeps the three lower faces fixed. Apply Definition 126.407 for the finite-window schedule, apply Definition 301.198 for the RegSeqRat tail, and apply Definition 291.114 for the dyadic endpoint and radius ledger. These certificates leave the descended packet on the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ faces as the refined packet.

It remains to exclude substitute support. Apply Definition 38.849 to prevent any hidden row from standing in for a displayed face, and apply Definition 38.850 to identify the descended and refined terminal packets on their common support. Hence the descent preserves precisely the listed surface and cannot create a route-level credential outside it. □

Theorem 38.508 (Real L10 formal-boundary exit certificate)**.** *An accepted current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read exports exactly the finite four-face boundary*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the displayed support rows

$$\begin{gathered}
\mathsf{BHist}_{\mathsf{DyadicRatCore}^{\uparrow}},\ \mathsf{BHist}_{\mathsf{StreamName}^{\uparrow}},\ \mathsf{BHist}_{\mathsf{RegSeqRat}^{\uparrow}},\ \mathsf{BHist}_{\mathsf{Real}^{\uparrow}},\\
\mathsf{hsame}_{\mathsf{DyadicRatCore}^{\uparrow}},\ \mathsf{hsame}_{\mathsf{StreamName}^{\uparrow}},\ \mathsf{hsame}_{\mathsf{RegSeqRat}^{\uparrow}},\ \mathsf{hsame}_{\mathsf{Real}^{\uparrow}},\\
\mathsf{Cont}_{\mathsf{DyadicRatCore}^{\uparrow}},\ \mathsf{Cont}_{\mathsf{StreamName}^{\uparrow}},\ \mathsf{Cont}_{\mathsf{RegSeqRat}^{\uparrow}},\ \mathsf{Cont}_{\mathsf{Real}^{\uparrow}},\\
\mathsf{Pkg}_{\mathsf{DyadicRatCore}^{\uparrow}},\ \mathsf{Pkg}_{\mathsf{StreamName}^{\uparrow}},\ \mathsf{Pkg}_{\mathsf{RegSeqRat}^{\uparrow}},\ \mathsf{Pkg}_{\mathsf{Real}^{\uparrow}},\\
\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}},\ \mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}},\ \mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}},\ \mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}.
\end{gathered}$$

Its formal boundary is the objectwise product of the four local face records: the DyadicRatCore endpoint and radius ledger, the StreamName terminal-window packet, the RegSeqRat finite-request section, and the local Real seal. This product is not a fifth route-level formalstatus, bridge row, selected limit, quotient stream, host equality, ambient completeness row, metric-completeness datum, arithmetic or order package, or standard bridge.

Proof. First fix the finite terminal witness. Apply Definition 38.464. The accepted terminal read determines exactly the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next fix the route. Apply Definition 38.466. It places every such witness on the ordered path

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Apply Definition 38.467 to identify the StreamName classifier pullback, the RegSeqRat finite-request section, and the Real source route as three readings of the same public boundary.

Now read the audit boundary. Apply Definition 38.471. It identifies the accepted Phase Real and Completion audit as the product of the four local face records, with closurestatus, formalstatus, leantarget, and bridgestatus rows read from the corresponding faces.

Finally apply Definition 38.837. It keeps paper closure, formalstatus, leantarget, and bridgestatus face-local and blocks any reading in which their product becomes a route-level status, aggregate bridge, quotient stream, selected limit, host equality, ambient completeness row, metric-completeness datum, arithmetic or order package, or standard bridge. Thus the certificate is exactly the four-face finite boundary and its objectwise formal record product. □

Theorem 38.509 (Real L10 four-face order rigidity)**.** *Let an accepted current Phase Real and Completion terminal read have precisely the visible faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Then the read is accepted only as the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

No permutation, skipped edge, or bypass through the same four local records is an accepted exit route. In particular, quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus tokens, and standard bridge rows cannot replace or reorder the four-face route.

Proof. First fix the source route. Apply Definition 38.466. It assigns every accepted terminal finite-window witness to the displayed path from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$.

Next compare the middle and terminal readings. Apply Definition 38.467. The StreamName terminal-window classifier, the RegSeqRat finite-request section, and the Real source route are three readings of the same public boundary over the same support rows. A permutation of the faces would change one of these source, middle, or terminal readings, so it is not the accepted triangle.

Now read the audit coordinate. Apply Definition 38.471. The current Phase Real and Completion exit audit is the objectwise product of the four local closurestatus, formalstatus, leantarget, and bridgestatus records, not an aggregate route token that can be reassigned to another order.

Finally exclude bypasses. Apply Definition 38.730. Deleting, substituting, or bypassing any one face leaves no accepted L10 exit route. Thus quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus tokens, and standard bridge rows cannot supply a reordered or shortened route. The only accepted exit route with those four visible faces is the ordered four-face route above. □

Theorem 38.510 (Real L10 terminal order window witness envelope)**.** *An accepted current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *read satisfies the Real L10 exit envelope if and only if both of the following hold:*

1. *the read has the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow};$$

2. *the finite-window witness rows are exactly the rows fixed by Definition 38.464: the fused* $\mathsf{StreamName}^{\uparrow}$ *terminal-window packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The envelope contains no quotient streams, selected limits, countable-choice rows, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, first read the route. Apply Definition 38.509. It forces any accepted terminal read with the four visible faces to use the ordered path

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Next read the window data. Apply Definition 38.464. The public boundary recovers exactly the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

For the reverse direction, assume the ordered route and the finite-window witness rows. Apply Definition 38.508. The route and witness data repack as the four-face formal boundary exported by the accepted Phase Real and Completion terminal read.

Finally apply Definition 38.837. It keeps the certificate face-local, so quotient streams, selected limits, countable-choice rows, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and $\mathsf{Real}^{\uparrow}$ completeness cannot be smuggled into the exit envelope. Hence the envelope is exactly the conjunction of the ordered route and the finite-window witness rows. □

Theorem 38.511 (Real L10 terminal envelope status locality)**.** *An accepted current Phase Real and Completion terminal envelope has status evidence exactly as the objectwise product of the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The ordered route and finite-window witness rows do not create a fifth route-level formalstatus, bridge row, selected limit, quotient stream equality, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First read the terminal envelope. Apply Definition 38.510. It identifies the accepted envelope with the displayed ordered route and the finite-window witness rows fixed by the StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next read the status coordinate. Apply Definition 38.471. The current Phase Real and Completion audit is looked up object by object from the four local face records, with closurestatus, formalstatus, leantarget, and bridgestatus rows remaining local to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now identify the four-face ledger. Apply Definition 126.516 and Definition 126.504. These results keep the dyadic budget, StreamName window, RegSeqRat handoff, and Real seal as the four faces of the same terminal finite boundary rather than as one route-level status credential.

Finally apply Definition 301.60. It reads the RegSeqRat classifier and readback rows as one of the four exposed faces of that terminal boundary. Therefore the terminal envelope carries exactly the four objectwise status records and no fifth route-level formalstatus, bridge row, selected limit, quotient stream equality, countable-choice row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. □

Theorem 38.512 (Real L10 audit-clean four-face exit)**.** *The current Phase Real and Completion exit reaches the verification side exactly when the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

each supply their local auditClean *record through the closurestatus, formalstatus, leantarget, and bridgestatus rows declared in that face's own local closurestatus block. The resulting exit certificate is the four-face product of those local audit-clean records and the terminal route and finite-window witness rows.*

This product creates no $\mathsf{Real}^{\uparrow}$ *completeness claim, metric-completeness datum, quotient stream, host equality, selected limit, countable-choice row, aggregate bridge status, or fifth route-level formal token.*

Proof. First read the terminal status coordinate through Definition 38.511. It identifies the accepted terminal envelope with the objectwise product of the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and it excludes any fifth route-level formalstatus or bridge row.

Next apply Definition 38.471. The roadmap exit audit is looked up through the local closurestatus, formalstatus, leantarget, and bridgestatus rows of those same four faces. Requiring the verification coordinate in each local face record to be auditClean is therefore exactly the four-face verification-side exit criterion.

Now identify the route and witness rows. Apply Definition 38.466 for the ordered route and Definition 38.464 for the terminal finite-window witness. Together they keep the certificate on the displayed DyadicRatCore, StreamName, RegSeqRat, and local Real faces, with only the visible BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally use Definition 38.837. It prevents the four local audit-clean records from becoming $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, quotient streams, host equality, selected limits, countable-choice data, aggregate bridge status, or a route-level formal token. Hence the exit criterion is precisely the four local audit-clean faces together with the existing route and witness rows. □

Theorem 38.513 (Real L10 scoped four-face exit)**.** *The current Phase Real and Completion closure-side exit is reached exactly when the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

each supply their own local scopedClosed *row through that face's local closurestatus block. The resulting exit certificate is the four-face product of those local scoped-closure records together with the terminal route and finite-window witness rows.*

This product creates no aggregate theory closure, completeness claim, quotient stream, selected limit, host real equality, bridge credential, or fifth route-level closure token.

Proof. First read the terminal status coordinate through Definition 38.511. It identifies the accepted terminal envelope with the objectwise product of the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 38.471. The current Phase Real and Completion exit reads closurestatus, formalstatus, leantarget, and bridgestatus from those same local face records. Requiring the closure coordinate in each local face record to be scopedClosed is therefore exactly the four-face closure-side exit criterion.

Now identify the route and witness rows. Apply Definition 38.466 for the ordered route and Definition 38.464 for the terminal finite-window witness. These keep the certificate on the displayed DyadicRatCore, StreamName, RegSeqRat, and local Real faces, with only the visible BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally use Definition 38.837. It keeps the closure coordinate independent from formalstatus and bridgestatus, so the four local scoped-closure rows cannot become an aggregate theory closure,

completeness claim, quotient stream, selected limit, host real equality, bridge credential, or fifth route-level closure token. Hence the exit criterion is precisely the four local scoped-closure faces together with the existing route and witness rows. □

Theorem 38.514 (Real L10 two-axis four-face exit)**.** *The current Phase Real and Completion exit is reached exactly when the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

each supply both their local scopedClosed *row and their local* auditClean *row through that face's own closurestatus, formalstatus, leantarget, and bridgestatus records. The resulting exit certificate is the four-face product of the local closure-side and verification-side rows, together with the terminal route and finite-window witness rows.*

This two-axis product creates no route-level aggregate status, route-level bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First read the closure coordinate through Definition 38.513. It says that the closure-side exit is exactly the four local scopedClosed rows for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the terminal route and finite-window witness rows.

Next read the verification coordinate through Definition 38.512. It says that the verification-side exit is exactly the four local auditClean rows for the same four faces, read from the face-local closurestatus, formalstatus, leantarget, and bridgestatus records.

Now combine the two coordinates objectwise. Apply Definition 38.511. The terminal envelope does not introduce a fifth status source; it only exposes the four face-local records already named by the closure-side and verification-side exit theorems.

Finally use Definition 38.837. The closure row, formal row, Lean target row, and bridge row remain separate coordinates, so the two-axis exit cannot be read as route-level aggregate status, a bridge row, quotient streams, selected limits, countable choice, metric completeness, or $\mathsf{Real}^{\uparrow}$ completeness. Hence the current Phase Real and Completion exit is precisely the four faces carrying both the local scopedClosed and local auditClean rows. □

Theorem 38.515 (RealUp L10 input factorization)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal read or* $\mathsf{Real}^{\uparrow}$ *classifier read that is usable as Phase Real and Completion data factors through the L10 four-object input stack*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization first reads the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows of chapter 291, then the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows of chapter 301, and only then the local* $\mathsf{Real}^{\uparrow}$ *seal or classifier surface. The visible route is transported by the displayed* hsame *rows, replayed by the displayed* Cont *rows, packaged by* Pkg *provenance, and named by the local* NameCert *rows before it is accepted as* $\mathsf{Real}^{\uparrow}$ *data.*

The factorization exports no quotient stream, host real equality, selected limit, countable choice, or ambient completeness coordinate.

Proof. Start with the Real L10 dependency lattice of Definition 38.442. It orders the public input surface by dyadic observations, finite StreamName windows, regular-rational readback, and the local Real seal rows.

Apply the source-route and interface-totality theorems, Definition 38.443 and Definition 38.444. They place every Phase Real and Completion interface read inside the displayed BHist, hsame, Cont, Pkg, and NameCert route over the same four faces.

The sibling route theorem Definition 38.530 supplies the cross-chapter handoff through chapter 291, chapter 126, and chapter 301. Thus a public Real seal read or classifier read reaches the local $\mathsf{Real}^{\uparrow}$ surface only after those sibling rows have been displayed.

Finally use the terminal route theorems Definition 38.670 and Definition 38.675. They exhaust the public exit by the four-object route and the structural rows already listed. Since none of those cited surfaces contains a quotient stream, host real equality, selected limit, countable-choice selector, or ambient completeness coordinate, no such coordinate can be exported by the accepted $\mathsf{Real}^{\uparrow}$ read. □

Theorem 38.516 (RealUp L10 downstream interface non-escape)**.** *Every downstream* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer that uses a public* $\mathsf{Real}^{\uparrow}$ *seal row or public* $\mathsf{Real}^{\uparrow}$ *classifier row must read that row through the same four-object route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The admissible downstream packet consists only of the displayed BHist *histories, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows attached to those four faces.*

The packet exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness row, or metric-completeness row.

Proof. Begin with Definition 38.515. It already forces every public $\mathsf{Real}^{\uparrow}$ seal read and classifier read usable as Phase Real and Completion data through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Now pass to the five downstream faces. A $\mathsf{Met}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, or $\mathsf{ContinuousMap}^{\uparrow}$ consumer can use the $\mathsf{Real}^{\uparrow}$ row only as the public seal or classifier surface supplied by chapter 38. The sibling surfaces chapter 57, chapter 766, chapter 128, chapter 129, and chapter 130 therefore inherit the same Real route rather than a separate analytic entrance.

Finally apply Definition 38.694. The public Real seal exports no quotient stream, selected limit, host real equality, host sequence equality, or ambient completeness theorem; the L10 input theorem also excludes countable choice from the route. Since the downstream consumer receives only that public Real row together with the displayed structural support, it cannot acquire quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, or metric-completeness rows from the Real input boundary. □

Definition 38.517 (Real L10 four-source handoff packet)**.** *The Real L10 four-source handoff packet is the finite packet whose public rows are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows, and the* $\mathsf{Real}^{\uparrow}$ *seal and classifier rows appearing in the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Its support data are componentwise hsame *transports between the displayed rows, displayed* Cont *replay maps for the accepted reads,* Pkg *provenance for each of the four source faces, and the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row naming the resulting Real-facing surface. The packet has no coordinate other than those four source faces and the listed BEDC support rows.*

Theorem 38.518 (Real L10 four-source handoff initiality)**.** *Every admissible Phase Real and Completion public* $\mathsf{Real}^{\uparrow}$ *seal read or classifier read whose support is confined to* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows factors through the Real L10 four-source handoff packet of*

Definition 38.517. If two such factorizations agree on the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback face, the* $\mathsf{Real}^{\uparrow}$ *seal or classifier face, and the displayed support rows, then they determine the same accepted public read.*

Conversely, the four-source handoff packet supplies the initial admissible source for these public reads: any accepted Real-and-Completion read with the listed support rows is obtained from that packet, and no quotient stream, selected limit, host real equality, ambient completeness, metric-completeness, countable-choice, or standard bridge coordinate can supply a competing initial source.

Proof. Begin with Definition 38.515. It sends every admissible Phase Real and Completion public $\mathsf{Real}^{\uparrow}$ seal or classifier read through the ordered four-object route. The definition of the handoff packet then records exactly that route as public source data, together with the BHist, hsame, Cont, Pkg, and local NameCert rows allowed in the statement.

The factorization is determined by the four visible faces. Apply Definition 126.612 to identify the DyadicRatCore observations, StreamName finite windows, RegSeqRat readback surface, and Real seal or classifier row as one exact four-face boundary. Then apply Definition 301.58 to repack any accepted terminal read over the same displayed support rows. Therefore two factorizations that agree on those faces and rows have the same accepted public read.

For existence in the reverse direction, apply Definition 38.519. It says that every Phase Real public read and every Completion public $\mathsf{Real}^{\uparrow}$ read already factors through the named packet, so the packet is an admissible source for every read under consideration.

Finally use Definition 38.520. The packet has no outlet for quotient streams, selected limits, host real equality, ambient completeness, metric-completeness, countable choice, or a standard bridge coordinate. Since the exactness and pullback steps leave only the four faces and the displayed support rows, no external analytic or bridge coordinate can serve as another initial source. □

Theorem 38.519 (Real L10 four-source handoff totality)**.** *Every Phase Real public read and every Completion public* $\mathsf{Real}^{\uparrow}$ *read factors through the Real L10 four-source handoff packet of Definition 38.517. In particular, the accepted read is obtained by first presenting the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face, then the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback face, and finally the* $\mathsf{Real}^{\uparrow}$ *seal or classifier face, with only the displayed* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *support rows.*

Proof. Start with Definition 38.515. It already places every Phase Real and Completion public $\mathsf{Real}^{\uparrow}$ seal or classifier read on the four-object route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 38.516. Downstream consumers receive only the same displayed histories, transports, continuations, package provenance, and local naming rows, so the route is stable when read by MetricSpace, RealAlgOrder, CompleteMetric, CompactMetric, and ContinuousMap surfaces.

Use Definition 38.444 to identify the interface reads with the source lattice coordinates, and use Definition 38.675 to close the terminal public exit over those four faces. These two references show that no additional source face is needed.

Repacking the listed rows gives exactly the packet of Definition 38.517. Thus every admitted public read factors through that named finite packet. □

Theorem 38.520 (Real L10 four-source handoff non-escape)**.** *The Real L10 four-source handoff packet exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness row, or metric-completeness row. Every downstream* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,*

$\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of a public* $\mathsf{Real}^{\uparrow}$ *read receives only the four source faces and the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *support rows.*

Proof. Apply Definition 38.516. It excludes the listed analytic and choice coordinates from every downstream consumer that reads a public Real seal or classifier row.

Now use Definition 38.515. The source of the downstream row is already limited to the four-object route and to the displayed support rows, so an excluded coordinate cannot enter before the downstream handoff.

Finally apply Definition 38.675. The terminal open-phase exit has no fifth face where quotient streams, host real equality, selected limits, countable choice, ambient completeness, or metric-completeness data could be stored. Therefore the named packet is non-escaping. □

Theorem 38.521 (Real L10 four-source handoff obstruction classification)**.** *For a Phase Real and Completion public* $\mathsf{Real}^{\uparrow}$ *read, factorization through the Real L10 four-source handoff packet of Definition 38.517 fails exactly when at least one of the following finite sites is absent or not componentwise transported:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback face;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal or classifier face;*
5. *one of the support rows* BHist, hsame, Cont, Pkg, *or* NameCert.

Equivalently, once these four faces and support rows are present and componentwise transported, the public $\mathsf{Real}^{\uparrow}$ *read factors through the named packet and has no further obstruction. No quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness row, metric-completeness row, or additional bridge coordinate can witness failure.*

Proof. For the failure direction, begin with Definition 38.515. It places every usable Phase Real and Completion public $\mathsf{Real}^{\uparrow}$ seal read or classifier read on the route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed structural support rows.

Use Definition 38.519 to compare the read with the packet of Definition 38.517. If the read does not factor through that packet, the totality theorem leaves no possible missing coordinate except one of the four displayed faces or one of the listed BHist, hsame, Cont, Pkg, and NameCert rows.

The exact location is finite. Apply Definition 126.612 to inspect the DyadicRatCore observation face, the StreamName finite-window face, the RegSeqRat readback face, and the local Real seal or classifier face. Then apply Definition 301.58 to inspect the common support rows used to repack the terminal read. Any remaining failure is therefore exactly one of the five listed sites.

For the converse direction, suppose one listed site is absent or not componentwise transported. Then the read lacks a required public face or a required support row of Definition 38.517, so it cannot factor through that packet.

If none of the five listed sites occurs, Definition 38.519 supplies the factorization through the packet. Finally Definition 38.520 excludes quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness, and additional bridge coordinates as possible obstruction witnesses. □

Theorem 38.522 (Real L10 four-source handoff classifier congruence)**.** *Let* R_0 *and* R_1 *be Phase Real and Completion public* $\mathsf{Real}^{\uparrow}$ *classifier reads that factor through the Real L10 four-source*

handoff packet of Definition 38.517. Suppose the two factorizations agree on the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation face, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback face, the local* $\mathsf{Real}^{\uparrow}$ *seal or classifier face, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then* R_0 *and* R_1 *determine the same finite BEDC public* $\mathsf{Real}^{\uparrow}$ *classifier packet.*

Any classifier disagreement between such reads is therefore one of the finite obstruction sites listed in Definition 38.521. It is not witnessed by quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness, or bridge data.

Proof. First identify the two factorizations. By Definition 38.518, an admissible public $\mathsf{Real}^{\uparrow}$ classifier read supported only by the displayed BHist, hsame, Cont, Pkg, and local NameCert rows factors through the named four-source packet, and two such factorizations with the same four faces and support rows determine the same accepted public read.

Next put the four faces on one support packet. Apply Definition 126.612 to align the DyadicRatCore observation face, the StreamName finite-window face, the RegSeqRat readback face, and the local Real seal or classifier face. Then apply Definition 301.58 to repack the accepted terminal read over the common BHist, hsame, Cont, Pkg, and NameCert support rows.

Now compare classifier coordinates. The classifier coherence theorem Definition 38.740 says that the visible classifier reads over these four faces determine the same finite support packet. Since R_0 and R_1 agree on every face and support row used by that theorem, their public $\mathsf{Real}^{\uparrow}$ classifier reads are the same finite BEDC packet.

It remains to locate any possible disagreement. By Definition 38.520, the handoff packet has no outlet for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness, or bridge data. Therefore Definition 38.521 leaves only the listed finite obstruction sites as possible witnesses for classifier disagreement. □

The $\mathsf{Real}^{\uparrow}$ L10 dependency spine consumes finite-window rows from chapter 126, regular-rational rows from chapter 301, and dyadic-radius rows from chapter 291. This places the current RealUp handoff on the same finite sibling surface used by stream naming, regular rational sequences, and the dyadic core. The $\mathsf{Real}^{\uparrow}$ seal consumes the $\mathsf{StreamName}^{\uparrow}$ finite observation interface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the $\mathsf{RegSeqRat}^{\uparrow}$ regularity carrier of chapter 301. These sibling rows form the visible L10 route read before the local seal is available to downstream Real and Completion consumers. The $\mathsf{Real}^{\uparrow}$ seal face is consumed together with chapter 291, chapter 126, and chapter 301 in the current Phase Real and Completion exit boundary. The consumer surface is Definition 126.500: it reads the Real seal only as one of the four displayed faces, alongside the finite dyadic budget, StreamName window, RegSeqRat readback, and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. The current $\mathsf{Real}^{\uparrow}$ L10 carrier is read through the $\mathsf{StreamName}^{\uparrow}$ finite observation interface of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ regularity certificate of chapter 301. The later $\mathsf{Met}^{\uparrow}$ consumer in chapter 57 uses the resulting $\mathsf{Real}^{\uparrow}$-valued distance interface only after those two sibling certificates are present, so the Real route is a sibling lattice through StreamName and rational regular sequences before it becomes a metric-distance surface.

Theorem 38.523 (Real StreamName RegSeqRat metric-observation factorization)**.** *Every* $\mathsf{Met}^{\uparrow}$*-facing observation of an accepted* $\mathsf{Real}^{\uparrow}$ *finite-window carrier factors through the ordered BEDC observation route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{MetricDistanceWitness}.$$

The first leg reads only the $\mathsf{StreamName}^{\uparrow}$ *finite observation interface and its displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The second leg reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows together with*

the dyadic window rows that lock the finite rational schedule. The third leg exposes the local $\mathsf{Real}^{\uparrow}$ *seal and readback surface. Only after that seal is visible may the downstream* $\mathsf{Met}^{\uparrow}$ *consumer read the* MetricDistanceWitness *ledger.*

Thus the metric observation exports no quotient Cauchy class, host real equality, ambient completeness principle, or direct metric access to an unsealed stream.

Proof. Start with Definition 38.670. It places the accepted Real-facing read on the route from dyadic observations through $\mathsf{StreamName}^{\uparrow}$ finite windows and $\mathsf{RegSeqRat}^{\uparrow}$ regular readback to the local $\mathsf{Real}^{\uparrow}$ seal, using only the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 126.79 and Definition 301.25. The handoff fixes the finite StreamName observation interface, while the lock identifies the regular-rational rows with the dyadic finite-window schedule consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

Now use Definition 38.449. It keeps every metric-facing read outside the Real construction boundary until the local $\mathsf{Real}^{\uparrow}$ seal and readback surface are already visible.

Finally apply the metric witness ledger only at the downstream consumer boundary, as in Definition 57.363. This records the MetricDistanceWitness depth accounting after the Real seal is available, and it supplies no coordinate for quotient Cauchy classes, host real equality, ambient completeness, or direct metric access to an unsealed stream. □

Theorem 38.524 (Real StreamName RegSeqRat metric classifier correspondence)**.** *Every accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observation is classifier-determined by the ordered BEDC route*

$\mathsf{StreamName}^{\uparrow}$ *finite-window classifier* $\to$ $\mathsf{RegSeqRat}^{\uparrow}$ *regularity classifier* $\to$ $\mathsf{Real}^{\uparrow}$ *seal classifier* $\to$ MetricDistanceW

The $\mathsf{StreamName}^{\uparrow}$ *leg fixes the finite schedule and its displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The* $\mathsf{RegSeqRat}^{\uparrow}$ *leg fixes the regularity classifier over the same window rows. The* $\mathsf{Real}^{\uparrow}$ *leg seals the readback surface before the metric witness can be consulted.*

Conversely, a MetricDistanceWitness *read cannot identify two* $\mathsf{Real}^{\uparrow}$ *observations unless their* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows reach the same* $\mathsf{Real}^{\uparrow}$ *seal. The correspondence therefore exports no quotient Cauchy class, host real equality, ambient completeness principle, or direct metric access to an unsealed stream.*

Proof. First apply Definition 38.523. It places every accepted metric-facing observation on the ordered route from $\mathsf{StreamName}^{\uparrow}$ finite windows through $\mathsf{RegSeqRat}^{\uparrow}$ regularity and the $\mathsf{Real}^{\uparrow}$ seal before the MetricDistanceWitness ledger is read.

Next use Definition 126.437. The dependency lattice says that reaching the RegSeqRat readback and Real seal surfaces from a finite StreamName window exposes only the displayed finite route and its BHist, hsame, Cont, Pkg, and NameCert rows.

Then apply Definition 301.103. It identifies the L10 source owners of the schedule, dyadic endpoint rows, regularity rows, classifier rows, and terminal Real seal. Thus the StreamName classifier and the RegSeqRat classifier determine the Real-facing seal read rather than leaving a free ambient coordinate.

Finally use Definition 57.363. The metric consumer can perform its continuation-depth accounting only after the sealed Real read is already present. Therefore a metric witness cannot separate or identify Real observations through any coordinate beyond the shared finite schedule, regularity window, and Real seal; quotient Cauchy classes, host real equality, ambient completeness, and direct access to an unsealed stream remain off the route. □

Theorem 38.525 (Real L10 metric classifier pullback correspondence)**.** *Let R and S be accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observations. Agreement of their* MetricDistanceWitness *classifiers is equivalent to agreement of the displayed four-face route consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal classifier row, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This pullback correspondence does not expose quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge rows, or direct metric access to an unsealed stream.

Proof. For the forward direction, first apply Definition 38.524. It reads agreement of the metric witness classifier back through the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity window, and the local $\mathsf{Real}^{\uparrow}$ seal before any downstream metric ledger is consulted.

Then apply Definition 38.778. The terminal common-tail comparison identifies the same classifier agreement with agreement of the $\mathsf{RegSeqRat}^{\uparrow}$ common-tail packet, the $\mathsf{StreamName}^{\uparrow}$ terminal-window packet, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance ledger, the local $\mathsf{Real}^{\uparrow}$ seal-source row, and the displayed support rows.

Finally use Definition 38.779. It keeps the local agreements compatible on finite covers, so the metric-facing classifier read is exactly the four-face classifier surface rather than an extra metric coordinate.

For the reverse direction, assume the two observations agree on the displayed four-face route and on the support rows BHist, hsame, Cont, Pkg, and NameCert. Repack those rows through Definition 38.775; the result is one public finite pullback interface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ seal.

With the Real seal visible, apply Definition 38.524 again to identify the sealed classifier route seen by the metric consumer. Only at this point may Definition 57.363 be used: it records continuation-depth accounting for the already sealed distance witness and adds no quotient stream equality, host real equality, selected limit, choice, completeness, standard bridge, or unsealed-stream row. □

Theorem 38.526 (Real L10 metric pre-seal non-escape)**.** *Let R be an accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observation. Suppose the* MetricDistanceWitness *classifier of R agrees with the displayed four-face route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, the local* $\mathsf{Real}^{\uparrow}$ *seal classifier, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Then the sealed* $\mathsf{Real}^{\uparrow}$ *boundary contains no BEDC coordinate for quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, metric-completeness rows, standard bridge rows, or direct access to an unsealed stream.*

Proof. First apply Definition 38.523. It places the metric-facing observation on the ordered route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the local $\mathsf{Real}^{\uparrow}$ seal, and only then the MetricDistanceWitness ledger.

Next use Definition 38.524. The metric witness classifier is read back through the finite StreamName schedule, the RegSeqRat regularity classifier, and the Real seal classifier, so the classifier agreement has no independent metric-side carrier.

Now apply Definition 38.525. It identifies that classifier agreement with the four-face pullback over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the local $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows.

The non-escape conclusion is exactly the boundary statement of Definition 38.449: before the sealed Real readback is present, there is no BEDC coordinate for quotient stream equality, selected

limits, countable choice, host real equality, ambient Real completeness, metric-completeness rows, standard bridge rows, or direct access to an unsealed stream.

After that boundary is fixed, Definition 57.363 may be cited only as the downstream continuation-depth ledger. It records metric consumer depth after the sealed Real route is already visible and does not add any of the excluded metric-side completion or choice data. □

Theorem 38.527 (Real L10 metric-distance pullback correspondence)**.** *Let R and S be accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observations. Agreement of their displayed* MetricDistanceWitness *distance rows is equivalent to agreement of the concrete four-face packet consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This distance pullback exports no quotient stream equality, selected limit, host real equality, countable choice, ambient Real completeness, metric-completeness row, standard bridge row, or direct access to an unsealed stream.

Proof. First apply Definition 38.525. It reads the metric-facing classifier agreement back to the four-face Real construction surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the local $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows.

Next route the distance ledger through Definition 38.523. The displayed MetricDistanceWitness distance rows can be consulted only after the StreamName schedule, RegSeqRat regularity rows, dyadic endpoint and radius ledger, and local Real seal have already been reached.

For the reverse direction, begin with componentwise agreement of the four-face packet and support rows. The same classifier pullback correspondence repacks those rows as agreement of the sealed metric-facing route, and the observation factorization then identifies the displayed distance ledger read on both observations.

Finally use Definition 38.526 to discharge the excluded coordinates. Only after this Real seal boundary is visible may Definition 57.363 be cited, and then only as the downstream continuation-depth account for the already sealed distance witness. □

Theorem 38.528 (Real L10 metric witness facewise determinacy)**.** *Let R and S be accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observations. Their* MetricDistanceWitness *classifier reads agree exactly when the displayed four-face route agrees: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail classifier rows; the local* $\mathsf{Real}^{\uparrow}$ *seal classifier; and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Conversely, componentwise agreement of those four faces and support rows supplies the same MetricDistanceWitness *classifier read. This determinacy statement exports no quotient stream equality, selected limit, countable choice, host real equality, ambient completeness, metric-completeness row, standard bridge row, or direct access to an unsealed stream.*

Proof. For the forward direction, apply Definition 38.523. It routes each accepted metric-facing observation through the $\mathsf{StreamName}^{\uparrow}$ finite-window interface, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity surface, the local $\mathsf{Real}^{\uparrow}$ seal, and only then the MetricDistanceWitness ledger.

Next use Definition 38.524. The metric classifier readback is therefore a readback of the StreamName, RegSeqRat, and Real seal classifiers, not a separate metric-side coordinate. Together with Definition 126.613, this places the Real consumer on the sibling L10 exit gate over the displayed four faces.

Now apply Definition 38.525. It identifies agreement of the metric witness classifiers with agreement of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ seal faces, together with the support rows BHist, hsame, Cont, Pkg, and NameCert.

For the reverse direction, start with componentwise agreement on those four faces and support rows. The same pullback correspondence repacks that data as agreement of the sealed Real-facing classifier route, and the classifier correspondence then supplies the identical MetricDistanceWitness read.

Finally apply Definition 38.526. The facewise agreement has no coordinate for quotient stream equality, selected limit, countable choice, host real equality, ambient completeness, metric-completeness rows, standard bridge rows, or direct unsealed-stream access, so the theorem states only determinacy of the finite classifier route. □

Theorem 38.529 (Real L10 metric four-face completeness refusal)**.** *Let R be an accepted metric-facing* $\mathsf{Real}^{\uparrow}$ *finite-window observation whose* MetricDistanceWitness *read agrees with the displayed four-face route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Then the metric witness supplies only the sealed distance and classifier read on that finite route.*

It cannot be reinterpreted as quotient stream equality, selected limit data, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, a standard bridge row, host real equality, or direct access to an unsealed stream.*

Proof. First apply Definition 38.523. The accepted metric-facing observation reaches the distance witness only through the StreamName finite-window schedule, the RegSeqRat regularity surface, the DyadicRatCore endpoint and radius ledger, the local Real seal, and the listed support rows.

Next use the two pullback correspondences, Definition 38.525 and Definition 38.527. They identify the classifier and distance reads with the same four-face packet, so the metric-facing consumer receives no second coordinate beyond the sealed packet.

Now apply Definition 38.528. Agreement of the displayed faces and support rows determines the MetricDistanceWitness classifier read, and conversely that read can be unpacked only to those faces and rows.

Finally combine the no-choice route factorization Definition 126.522 with Definition 38.526. The StreamName route has no coordinate for quotient stream equality, selected limits, countable choice, host sequence equality, metric-completeness data, or standard bridge rows, and the Real metric pre-seal boundary blocks the same data before the local $\mathsf{Real}^{\uparrow}$ seal is visible. Hence the metric witness remains a sealed finite distance and classifier read, not a completeness, choice, quotient, limit, or bridge assertion. □

Theorem 38.530 (Real L10 open-phase sibling route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *public seal read in the L10 open phase factors through the certified countable-name route of chapter 126, the regular rational sequence route of chapter 301, and the dyadic ledger route of chapter 291 before it reaches the local* $\mathsf{Real}^{\uparrow}$ NameCert *row.*

Proof. Start with the ordered dependency lattice of Definition 38.442. It places the dyadic observation face, the finite stream-window face, and the regular-rational classifier face before the local Real seal and readback rows.

Use the source-route and interface-totality theorems for the same lattice. They require every open-phase Real or Completion interface read to remain inside the displayed BHist, hsame, Cont, Pkg, and NameCert route through those sibling surfaces.

The local seal is therefore reached by composing the DyadicRatCore ledger route, the StreamName finite-window route, and the RegSeqRat regular-rational route. This composition does not assert completeness, quotient real equality, selected representatives, or host real equality; it only records the finite sibling route feeding the local $\mathsf{Real}^{\uparrow}$ naming row. □

Theorem 38.531 (Real window synchronizer seal dependency)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read in the L10 open phase that depends on synchronized StreamName windows and RegSeqRat tails must pass through the* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier of Definition 979.1. The read consumes the displayed window, tail, threshold, CauchyLimitSeal row, Real seal row, transport, route, provenance, and local naming rows before the local* $\mathsf{Real}^{\uparrow}$ *seal is public.*

Proof. Start from the open-phase sibling route of Definition 38.530. It already places each public Real seal read over the StreamName, RegSeqRat, and DyadicRatCore sibling surfaces.

When the read additionally requires synchronized windows, apply the RealWindowSynchronizer carrier and obligation rows of Definition 979.1 and Definition 979.7. These rows identify the required window-to-seal packet as $W, R, T, L, S, H, C, P, N$.

Thus the seal read reaches the local Real surface only after the synchronizer has replayed the common StreamName window, RegSeqRat tail, shared threshold, CauchyLimitSeal row, and RealUp seal row through the displayed route. □

Theorem 38.532 (Real tail-budget coherence consumer route)**.** *Every L10 open-phase* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer that uses shared tail budgets must also read the* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet of chapter 917. Its route includes the TailMeet threshold, observation budget, selector budget, shared StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, agreement seal, CauchyLimitSeal row, transport, continuation, provenance, and local naming rows before the consumer receives a Real seal.*

Proof. Use the open-phase dependency lattice to place the consumer on the finite sibling route through StreamName, RegSeqRat, DyadicRatCore, and the local Real seal.

If the same consumer uses shared tail budgets, apply the TailBudgetCoherence public export boundary and bridge interface from Definition 917.13 and Definition 917.15. These theorems expose the public route $M, O, S, W, D, R, A, L, H, C, P, N$.

The Real consumer therefore reads the shared tail-budget packet before receiving the Real seal. The packet contributes only the displayed tail-budget and seal rows, carried by BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance. □

Theorem 38.533 (Real L10 synchronizer non-escape)**.** *The L10 open-phase route obtained by combining Definition 38.531 with Definition 38.532 exports no quotient real equality, selected host limit, host real equality, or ambient completeness principle. It exposes only the finite StreamName, RegSeqRat, DyadicRatCore, RealWindowSynchronizer, TailBudgetCoherence,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *route needed for the local* $\mathsf{Real}^{\uparrow}$ *seal read.*

Proof. First use the synchronizer dependency theorem. It routes every synchronized-window Real seal read through the finite RealWindowSynchronizer packet and excludes detached window, tail, limit-seal, and Real-seal routes.

Next use the tail-budget coherence consumer theorem. It routes every shared-budget consumer through the finite TailBudgetCoherence public packet rather than through an ambient completion object.

Finally combine these two finite packets with the original open-phase sibling route. None of the cited surfaces carries quotient real equality, selected host limits, host real equality, or ambient completeness, so the composite L10 route cannot export those coordinates. □

Theorem 38.534 (Real open-phase four-object terminal route)**.** *Every accepted L10 open-phase* $\mathsf{Real}^{\uparrow}$ *public seal or completion-facing read is determined by the four displayed faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with its* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. If two such reads agree on those four faces and those support rows, then they are the same terminal finite route for the public seal consumption. No quotient stream, selected host limit, ambient completeness principle, metric-completeness datum, countable-choice row, or standard bridge row can replace any one of the four faces.*

Proof. Start with Definition 38.530. It places every accepted open-phase $\mathsf{Real}^{\uparrow}$ seal or completion-facing read on the finite sibling route through StreamName, RegSeqRat, DyadicRatCore, and the local Real naming row before the read reaches the public Real surface.

Next apply Definition 126.500. The same read is the current Phase Real and Completion exit packet, so its displayed boundary consists exactly of the $\mathsf{StreamName}^{\uparrow}$ window face, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, $\mathsf{RegSeqRat}^{\uparrow}$ readback face, $\mathsf{Real}^{\uparrow}$ seal face, and the BHist, hsame, Cont, Pkg, and NameCert support rows.

Use Definition 126.449 to identify two accepted reads that agree on those four faces and support rows. Terminality leaves no second route through which the public seal can be consumed.

Finally use Definition 38.413. The pullback repacks the terminal boundary as a local $\mathsf{Real}^{\uparrow}$ route and preserves the exclusions from the sibling and exit-route theorems, so quotient streams, selected host limits, ambient completeness, metric-completeness data, countable-choice rows, and standard bridge rows cannot stand in for a missing face. □

Theorem 38.535 (Real terminal four-object refusal)**.** *Every accepted L10* $\mathsf{Real}^{\uparrow}$ *public seal or completion-facing read must display the four terminal faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with its* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. A proposed public seal, completion read, or terminal route that omits one of those four faces, or replaces it by quotient stream equality, a selected host limit, ambient completeness, metric-completeness data, countable-choice rows, or a standard bridge row, lies outside the accepted terminal finite route.*

Proof. Apply Definition 38.530. It puts every accepted L10 Real seal read on the finite sibling route through StreamName, RegSeqRat, DyadicRatCore, and the local Real naming row before the read reaches the public surface.

Next apply Definition 126.500. The completion-facing packet has exactly the four displayed object faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Use Definition 126.449. Agreement on the four faces and support rows determines the accepted terminal route, so a packet with an absent face has no accepted coordinate from which that face can be read.

Finally apply Definition 38.413. The pullback repacks the same finite boundary at the $\mathsf{Real}^{\uparrow}$ surface and preserves the exclusions already present in the sibling and exit routes. Hence quotient stream equality, selected host limits, ambient completeness, metric-completeness data, countable-choice rows, and standard bridge rows cannot serve as substitutes for any missing terminal face. □

Theorem 38.536 (Real open-phase objectwise status readback)**.** *Every accepted L10* $\mathsf{Real}^{\uparrow}$ *public seal or completion-facing read reports status objectwise: it must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger status, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window status, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and seal status, and the local* $\mathsf{Real}^{\uparrow}$ *seal status as separate local faces. No route-level formalstatus token, aggregate bridge row, selected limit, quotient stream equality, ambient completeness row, metric-completeness row, or countable-choice row can replace any missing local face status.*

Proof. Start with Definition 38.534. It identifies every accepted public seal or completion-facing read with the four displayed faces and their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows, so the status read already has four local coordinates rather than one route coordinate.

Apply Definition 38.535. A proposed read that omits one of those local faces, or replaces it by quotient stream equality, a selected host limit, ambient completeness, metric completeness, countable choice, or a bridge row, is outside the accepted terminal finite route.

Next use Definition 126.463 and Definition 126.467. These results keep the completion boundary readable only through the displayed facewise records and prevent a local status record from escaping into a global completion or bridge assertion.

The roadmap two-axis discipline is therefore applied only as a status-reading rule. It says how to read each local closure and verification entry once the corresponding face is present; it does not supply evidence for either axis, and it does not create a route-level token that can replace the DyadicRatCore, StreamName, RegSeqRat, or Real status face. □

Theorem 38.537 (Real L10 facewise exit status noncollapse)**.** *A Phase Real and Completion exit read for* $\mathsf{Real}^{\uparrow}$ *is accepted exactly when the four local faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each expose their own closure and status row through the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support. No aggregate route-level formalstatus token, bridge row, quotient stream, selected host limit, ambient completeness row, metric-completeness row, or countable-choice row can replace any one of those four facewise status reads.*

Proof. For the forward direction, apply Definition 38.534. It identifies every accepted exit read with the four-face terminal boundary, so each status read is attached to one of the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and to the common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

For the reverse direction, use Definition 38.536. Once all four local faces expose their own status rows through the shared support, the exit read is precisely the objectwise status readback already permitted for the L10 Real route.

Apply Definition 38.535 to the noncollapse clause. Any packet that omits one local face, or substitutes quotient stream equality, a selected host limit, ambient completeness, metric completeness, countable choice, or a bridge row, is outside the terminal finite route and therefore cannot supply the missing status coordinate.

Finally use Definition 126.500 together with Definition 38.413. These theorems transport the same correspondence across the StreamName exit route and the local Real pullback, so the statement records facewise exit status across the four L10 objects rather than a new verification-axis upgrade. □

The $\mathsf{Real}^{\uparrow}$ seal is a downstream consumer of chapter 126, chapter 301, and chapter 291. The $\mathsf{StreamName}^{\uparrow}$ surface supplies finite observation windows, the $\mathsf{RegSeqRat}^{\uparrow}$ surface supplies the regular rational source row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ surface supplies the dyadic tolerance and readback ledger; only after those three sibling surfaces are in scope does the local $\mathsf{Real}^{\uparrow}$ seal present its public certificate boundary.

Theorem 38.538 (Independent reindexed constant rational embedding certificate)**.** *For histories* $d, e : \mathsf{Hist}$ *and observation maps* $r, q : \mathsf{Hist} \to \mathsf{Hist}$*, the two concrete endpoint assignments*

$$d \longmapsto (\mathsf{RatConstStream}_d \circ r, \mathsf{E1}(d)), \qquad e \longmapsto (\mathsf{RatConstStream}_e \circ q, \mathsf{E1}(e))$$

supply the independently scheduled constant rational embedding certificate into RealUp. The first endpoint has carrier exactness

$$\mathsf{RatHistoryCarrier}(d) \iff \mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r) \iff \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}($$

and the second endpoint has carrier exactness

$$\mathsf{RatHistoryCarrier}(e) \iff \mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_e \circ q) \iff \mathsf{RealConstantHistoryCarrier}(\mathsf{E1}($$

Its classifier exactness is the three-row chain displayed in Definition 38.29: the RatUp classifier of d and e, the StreamNameUp classifier of $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_e \circ q$, the RealUp unary-observation stream classifier of those same streams, and the RealUp constant-history classifier of $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$ are equivalent.

Proof. Apply Definition 38.32 to d with observation map r. This gives the first carrier row, identifying the RatUp carrier of d, the StreamNameUp carrier of $\mathsf{RatConstStream}_d \circ r$, and the RealUp constant carrier of $\mathsf{E1}(d)$.

Apply the same theorem separately to e with observation map q. This gives the second carrier row, identifying the RatUp carrier of e, the StreamNameUp carrier of $\mathsf{RatConstStream}_e \circ q$, and the RealUp constant carrier of $\mathsf{E1}(e)$.

For classifier exactness, apply Definition 38.29. Its three equivalence rows are precisely the classifier rows required for independently scheduled constant streams and their paired e-one seals. □

Lean4 ✓ `BEDC.Derived.RealUp.RealIndependentReindexedConstant rational embedding certificate`

Theorem 38.539 (Real reindexed constant finite-window package)**.** *For histories d, e : Hist, observation maps r, q : Hist $\to$ Hist, and a probe bundle $\mathcal{P}$: ProbeBundle, suppose the bundle contains a certified unary observation:*

$$\exists n : \mathsf{Hist},\ \mathsf{InBundle}(n, \mathcal{P}) \wedge \mathsf{UnaryHistory}(n).$$

Then every RatUp classifier witness $\rho : \mathsf{RatHistoryClassifier}(d, e)$ determines the finite-window constant bridge package:

1. *the finite StreamName observation surface*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q, \mathcal{P});$$

2. *the carrier rows for d and e displayed in Definition 38.538;*
3. *the classifier bridge from ρ through the independently reindexed StreamName row, the RealUp unary-observation stream row, and the RealUp constant-history classifier row of $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$;*
4. *the RegSeqRat-facing ledger whose entries are exactly the* InBundle *rows of $\mathcal{P}$ together with their* UnaryHistory *witnesses and the pointwise RatUp classifier copied from ρ.*

Proof. First build the finite observation row. Apply Definition 126.56 to the constant streams with endpoints d and e and to the inhabited probe bundle $\mathcal{P}$. The reverse direction of that exactness theorem consumes ρ and yields the finite-window classifier for $\mathsf{RatConstStream}_d$ and $\mathsf{RatConstStream}_e$.

The displayed streams are independently reindexed, but constant streams ignore their observation argument. Thus at every bundled certified unary observation n, both composites evaluate as $\mathsf{RatConstStream}_d(r(n)) = d$ and $\mathsf{RatConstStream}_e(q(n)) = e$. The same pointwise classifier ρ therefore supplies the required $\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q, \mathcal{P})$ row.

Next attach the global bridge. Apply Definition 38.538 to d, e, r, q. Its two carrier rows give the RatUp–StreamName–RealUp carrier identifications for the two endpoints, and its classifier rows transport ρ through the independently reindexed StreamName classifier, the RealUp unary-observation stream classifier, and the RealUp constant-history classifier of the two e-one seals.

Finally repack the finite-window row with those carrier and classifier rows. The ledger consumed downstream by RegSeqRat is not a new carrier: it is the visible ProbeBundle spine, each InBundle membership row, the corresponding UnaryHistory witness, and the copied RatUp point classifier at that observation. This is precisely the finite observation surface associated to the checked global constant embedding. □

Theorem 38.540 (Real stream classifier finite-prefix stability package)**.** *For rational-observation streams $x, y, z : \mathbb{N} \to \mathsf{Hist}$ and prefix index n, if*

$$\forall i,\ \mathsf{RatHistoryCarrier}(x(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{RealStreamClassifier}(y, z),$$

then

$$\mathsf{RealStreamPrefixClassifier}(x, x, n) \wedge \mathsf{RealStreamPrefixClassifier}(y, x, n) \wedge \mathsf{RealStreamPrefixClassifier}(x, z, n) \wedge \mathsf{RatHistory}$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the selected prefix index n, and apply the same theorem to $\mathsf{RealStreamClassifier}(y, z)$ at n. These give the two finite-prefix classifier witnesses for x with y and for y with z.

Use Definition 38.109 with the displayed rational-history carrier row for x to obtain $\mathsf{RealStreamPrefixClassifier}(x,$ Apply Definition 38.110 to the x, y prefix witness to obtain $\mathsf{RealStreamPrefixClassifier}(y, x, n)$.

Compose the x, y and y, z prefix witnesses by Definition 38.147, yielding $\mathsf{RealStreamPrefixClassifier}(x, z, n)$. Finally apply Definition 38.111 to this composite prefix to recover $\mathsf{RatHistoryClassifier}(x(n), z(n))$.

Apply the RatUp positive-denominator projection Definition 37.50 to the endpoint classifier. Positive-denominator readback by Definition 37.12 gives $\mathsf{UnaryHistory}(x(n))$ and $\mathsf{UnaryHistory}(z(n))$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier finite prefix unary stability package`

Fit. Traditional target: finite-prefix stability for RealUp streams. BEDC source: RatUp plus RealUp stream prefixes. Carrier: rational-observation streams. Classifier: RealStreamClassifier and RealStreamPrefixClassifier. Stability: prefix reflexivity, symmetry, transitivity, and endpoint projection. Ledger: the selected prefix index n. Core theorem: Definition 38.540. Exactness: the three prefix rows, selected endpoint classifier, and two unary endpoint rows. Standard bridge: finite rational prefix agreement. Status: ProofTarget.

Theorem 38.541 (Real stream finite-prefix endpoint denominator package)**.** *For rational-observation streams $x, y, z : \mathbb{N} \to \mathsf{Hist}$ and prefix index n, if*

$$\forall i,\ \mathsf{RatHistoryCarrier}(x(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{RealStreamClassifier}(y, z),$$

then the selected finite prefixes and endpoint carry the denominator package:

$$\begin{aligned}&\mathsf{RealStreamPrefixClassifier}(x, x, n) \wedge \mathsf{RealStreamPrefixClassifier}(y, x, n) \wedge \mathsf{RealStreamPrefixClassifier}(x, z, n) \wedge \mathsf{RatHist}\\ &\quad \wedge\ \mathsf{PositiveUnaryDenominator}(x(n)) \wedge \mathsf{PositiveUnaryDenominator}(z(n)) \wedge \mathsf{UnaryHistory}(x(n)) \wedge \mathsf{UnaryHis}\\ &\quad\quad \wedge\ (\mathsf{hsame}(x(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(z(n), \mathsf{e}) \to \mathsf{False}).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier finite prefix endpoint denominator package`

Proof. Use finite-prefix reflexivity on the carrier row for x, use $\mathsf{RealStreamClassifier}(x, y)$ to obtain the x, y prefix and symmetrize it to the y, x prefix, and compose the x, y and y, z prefixes to obtain the x, z prefix. Endpoint projection gives $\mathsf{RatHistoryClassifier}(x(n), z(n))$. The rational-history classifier supplies positive denominator witnesses for both endpoints, and the positive denominator readback gives unary membership together with exclusion of e. □

Theorem 38.542 (Real stream finite-prefix transported endpoint denominator package)**.** *For rational-observation streams $x, x', y, z, z' : \mathbb{N} \to \mathsf{Hist}$ and index n, suppose*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{RealStreamClassifier}(y, z), \quad \mathsf{hsame}(x(n), x'(n)), \quad \mathsf{hsame}(z(n), z'(n)).$$

Then the transported selected endpoints carry the denominator package:

$$\mathsf{RatHistoryClassifier}(x'(n), z'(n)) \wedge \mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(z'(n))$$
$$\wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(z'(n)) \wedge (\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(z'(n), \mathsf{e}) \to \mathsf{False}).$$

Proof. Compose the two real stream classifiers pointwise. At the selected index this gives $\mathsf{RatHistoryClassifier}(x(n), z$ Transport that rational history classifier along the displayed hsame rows to obtain $\mathsf{RatHistoryClassifier}(x'(n), z'(n))$. RatUp positive-denominator readback then gives the two positive denominator witnesses, and each positive denominator witness gives unary membership together with exclusion of e. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier finite prefix transported endpoint denominator package`

Theorem 38.543 (Real stream prefix truncation endpoint denominator package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, if*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m + n),$$

then the selected endpoints of the shorter certified prefix carry the complete endpoint-classifier and denominator-shape package:

$$\mathsf{RatHistoryClassifier}(x(n), y(n)) \wedge \mathsf{PositiveUnaryDenominator}(x(n)) \wedge \mathsf{PositiveUnaryDenominator}(y(n)) \wedge \mathsf{UnaryHistory}(x($$

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated endpoint denominator package`

Proof. First descend from the displayed prefix through $m + n$ to the selected prefix through n. This is the constructor induction used in Definition 38.118: the base prefix is already the selected prefix, and each successor step projects the shorter recursive prefix component.

Apply Definition 38.111 to the resulting prefix classifier at n. This projects the endpoint rational-history classifier $\mathsf{RatHistoryClassifier}(x(n), y(n))$, which is kept as the first conjunct of the displayed package.

Apply Definition 37.50 to the endpoint classifier. The two projected witnesses are exactly $\mathsf{PositiveUnaryDenominator}(x(n))$ and $\mathsf{PositiveUnaryDenominator}(y(n))$.

Apply Definition 37.12 separately to the two positive-denominator rows. The two applications supply $\mathsf{UnaryHistory}(x(n))$, $\mathsf{UnaryHistory}(y(n))$, and the two nonemptiness implications, so the seven displayed rows repack directly. □

Theorem 38.544 (Real stream prefix add-left endpoint positive unary nonempty package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, if the left stream has unary histories pointwise and the prefix through $m + n$ is classified,*

$$\forall i,\ \mathsf{UnaryHistory}(x(i)), \qquad \mathsf{RealStreamPrefixClassifier}(x, y, m + n),$$

then the selected endpoints carry positive unary denominator witnesses, unary history rows, and empty-boundary exclusions:

$$\mathsf{PositiveUnaryDenominator}(x(n)) \wedge \mathsf{PositiveUnaryDenominator}(y(n)) \wedge \mathsf{UnaryHistory}(x(n)) \wedge \mathsf{UnaryHistory}(y(n)) \wedge (\mathsf{hsame}(x(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n), \mathsf{e}) \to \mathsf{False}).$$

Proof. Use add-left prefix extraction to descend from the displayed $m + n$ prefix to the selected n prefix, retaining the unary row at $x(n)$. Endpoint projection reads the rational-history classifier at $(x(n), y(n))$. RatUp positivity gives positive unary denominator witnesses for both endpoints, and positive denominator readback supplies the two unary rows and the two empty-boundary exclusions. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier add left endpoint positive unary nonempty package`

Theorem 38.545 (Real stream prefix truncated endpoint e-zero absurdity)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, indices m, n, and histories $z_X, z_Y : \mathsf{Hist}$, if*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m + n),$$

then the selected truncated endpoints cannot be internally same as zero-headed endpoints:

$$(\mathsf{hsame}(x(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).$$

Proof. Apply the truncated endpoint denominator package to obtain positive unary denominator witnesses for $x(n)$ and $y(n)$. Transport each positive denominator witness across the displayed zero-headed endpoint comparison, and use the RatUp positive-denominator zero-headed exclusion on each side. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated endpoint e0 absurd`

Fit. Traditional target: finite-prefix denominator readback for RealUp streams. BEDC source: RealUp prefix classifier plus RatUp denominator certificate. Carrier: rational-observation streams. Classifier: RealStreamPrefixClassifier. Stability: prefix truncation and endpoint projection. Ledger: selected indices m, n. Core theorem: Definition 38.543. Exactness: the displayed endpoint-classifier and denominator-shape package. Standard bridge: RealUp dyadic embedding and RegSeqRat endpoint-readback obligations. Status: ProofTarget.

Theorem 38.546 (Real stream prefix contextual one-tail denominator package)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, indices m, n, and histories $a, b : \mathsf{Hist}$, assume*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m + n),$$

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

and

$$\mathsf{Cont}(p_X(n), x(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n$$

Then the selected contextual endpoints carry the finite-prefix package:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
3. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
4. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
5. *If* $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$, *then* $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First apply Definition 38.122 to the classified prefix and the four selected continuation displays. This gives the contextual endpoint classifier $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.

Apply Definition 37.50 to that classifier. The two projections are the positive unary denominator witnesses for $o_X(n)$ and $o_Y(n)$.

Apply Definition 37.12 separately to the two positive denominator witnesses. This gives unary membership and empty endpoint exclusion for both contextual outputs.

For the one-tail readback, transport the endpoint classifier along the two displayed hsame comparisons to obtain $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$. Apply Definition 37.49 in the forward direction, which yields the two unary tail witnesses and $\mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier contextual e1 denominator package`

Fit. Traditional target: finite-prefix contextual denominator readback for RealUp streams. BEDC source: RealUp prefix context closure plus RatUp denominator and one-tail exactness. Carrier: rational-observation streams with selected unary contexts. Classifier: RealStreamPrefixClassifier and RatHistoryClassifier. Stability: prefix truncation, contextual endpoint closure, and hsame transport. Ledger: selected indices m, n and displayed outputs $o_X(n), o_Y(n)$. Core theorem: Definition 38.546. Exactness: the displayed five-part endpoint package. Standard bridge: RegSeqRat contextual endpoint readback. Status: ProofTarget.

Theorem 38.547 (Real stream prefix contextual zero-endpoint exclusion package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, *indices* m, n, *and histories* $z_X, z_Y : \mathsf{Hist}$, *assume*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n),$$

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

and

$$\mathsf{Cont}(p_X(n), x(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n$$

Then the selected contextual endpoints exclude every zero-headed endpoint:

$$(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).$$

Proof. Apply Definition 38.546 to the classified prefix and the displayed unary-context and continuation hypotheses. The package supplies $\mathsf{UnaryHistory}(o_X(n))$ and $\mathsf{UnaryHistory}(o_Y(n))$.

Suppose $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X))$ holds. Transport the left unary-history witness along this displayed endpoint comparison, then apply the unary no-zero-extension theorem to get a contradiction.

The right endpoint is identical. If $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y))$ holds, transport $\mathsf{UnaryHistory}(o_Y(n))$ to the displayed zero-headed endpoint and apply the same unary no-zero-extension theorem. These two implications repack as the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier contextual e0 endpoint absurd`

Fit. Traditional target: finite-prefix contextual endpoint shape exclusion. BEDC source: RealUp prefix classifier, RatUp denominator package, and Cont. Carrier: rational-observation streams with selected unary contexts. Classifier: RealStreamPrefixClassifier plus endpoint RatHistoryClassifier. Stability: inherited from contextual endpoint closure. Ledger: selected indices and contextual outputs. Core theorem: Definition 38.547. Exactness: the two zero-headed exclusions. Standard bridge: RegSeqRat endpoint-shape readback. Status: ProofTarget.

Theorem 38.548 (Real stream prefix contextual full endpoint package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m+n),$$

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

$$\mathsf{Cont}(p_X(n), x(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n$$

and

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)).$$

Then the selected contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
3. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
4. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
5. $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First apply Definition 38.546 to the classified prefix and the displayed unary-context and continuation hypotheses. Use the two displayed one-extension comparisons for the final readback row. This supplies the endpoint classifier, the positive denominator witnesses, the unary history witnesses, the nonempty exclusions, and $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Next apply Definition 38.547 to the same classified prefix, unary-context data, continuation data, and zero-extension tests z_X, z_Y. It gives the two zero-headed exclusions, so the six displayed rows are exactly the collected finite-prefix endpoint package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier contextual full endpoint package`

Theorem 38.549 (Real stream classifier selected contextual full endpoint package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\mathsf{RealStreamClassifier}(x, y),$$

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

$$\mathsf{Cont}(p_X(n), x(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n$$

and

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)).$$

Then the selected contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
3. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
4. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
5. $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. Apply Definition 38.106 at the selected index n. This turns the global $\mathsf{RealStreamClassifier}(x, y)$ row into the finite-prefix classifier $\mathsf{RealStreamPrefixClassifier}(x, y, n)$, read as the $0 + n$ instance needed by the prefix endpoint packages.

With this prefix classifier in hand, apply Definition 38.546 using the displayed unary-context rows, the two context-identification rows, the four continuation rows, and the two one-headed endpoint comparisons. It supplies the endpoint classifier, the positive unary denominator witnesses, the unary endpoint witnesses, the nonempty exclusions, and $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Finally apply Definition 38.547 to the same $0+n$ prefix classifier and the same selected context data, now with the zero-headed tests z_X, z_Y. The two resulting contradictions against $\mathsf{E0}(z_X)$ and $\mathsf{E0}(z_Y)$ complete the displayed endpoint package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected contextual full endpoint package`

Theorem 38.550 (Real stream selector-seal endpoint route)**.** *A terminal* $\mathsf{Real}^{\uparrow}$ *endpoint read supplied to a* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet is confined to the selected stream prefix and contextual endpoint package of Definition 38.549. Thus the endpoint exposes only the selected finite-prefix classifier, positive unary denominator witnesses, unary endpoint witnesses, nonempty exclusions, zero-headed exclusions, and one-headed tail agreement rows already present in that package.*

The endpoint read carries no host real equality, quotient stream equality, detached limit object, ambient completeness principle, or independent completion route beyond the selected contextual endpoint data.

Proof. Apply Definition 38.549 at the selected index read by the selector-seal packet. It extracts the finite-prefix classifier and the contextual endpoint rows from the global $\mathsf{RealStreamClassifier}$ route.

The positive unary denominator and nonempty rows in that package are supplied by Definition 37.12, used in the preceding endpoint arguments. They make the endpoint a visible finite readback row rather than a hidden host-real object.

Finally inspect the package output. It lists classifier, denominator, unary, nonempty, zero-headed exclusion, and one-headed tail agreement rows only. None of those rows is host real equality, quotient stream equality, a detached limit, ambient completeness, or a second completion route, so the selector-seal endpoint remains confined to the selected contextual endpoint surface. □

Fit. Traditional target: selected contextual full endpoint package for real observation streams. BEDC source: stream classifier prefix extraction, prefix contextual denominator package, and prefix contextual zero-endpoint exclusion. Carrier: rational-observation streams and selected context histories. Classifier: $\mathsf{RealStreamClassifier}$ with the induced $\mathsf{RatHistoryClassifier}$ on the selected outputs. Stability: inherited from hsame context transport and Cont composition at the selected index. Ledger: the single chosen index n, its input contexts, output contexts, and endpoint heads. Core

theorem: Definition 38.549. Exactness: classifier, denominator, unary, nonempty, zero-headed exclusion, and one-headed tail agreement rows. Standard bridge: RegSeqRat and dyadic endpoint readback for selected observations. Status: ProofTarget.

Theorem 38.551 (Real unary stream-name classifier exactness)**.** *For observation maps* s, t : Hist → Hist*, if*

$$\mathsf{RatStreamNameCarrier}(s) \quad \textit{and} \quad \mathsf{RatStreamNameCarrier}(t),$$

then the Real unary classifier and the StreamNameUp classifier identify the same BEDC relation:

$$\mathsf{RealUnaryStreamClassifier}(s, t) \iff \mathsf{RatStreamNameClassifier}(s, t).$$

Proof. Unfold Definition 126.1, Definition 126.2, and Definition 38.26. Both classifier names then expose the same pointwise assertion: for every certified unary observation n, the histories $s(n)$ and $t(n)$ are related by RatHistoryClassifier.

For the forward direction, combine the two displayed carrier hypotheses with the pointwise Real unary classifier row. Repack those rows under the StreamNameUp classifier name.

For the reverse direction, project the pointwise classifier field from RatStreamNameClassifier(s, t) at each certified unary observation. The resulting pointwise RatUp classifier is exactly the Real unary classifier row. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier ratStreamName pointwise iff`

Definition 38.552 (Phase Real certificate obligation surface)**.** *The Phase Real stream-observation interface is the tuple consisting of* RatStreamNameCarrier*,* RatStreamNameClassifier*,* RealUnaryStreamClassifier*,* RealUnaryStreamWindowClassifier*, certified unary anchor and bound rows, and the selected endpoint denominator package. Its carrier face is the rat-stream-name carrier row, its classifier face is the Real unary classifier identified with the StreamNameUp classifier, its finite-window face is the bounded unary observation window, and its selected-endpoint face is the denominator package at the certified observation.*

Proof. The carrier and classifier faces are named by Definition 38.551: under the two RatStreamNameCarrier rows, the Real unary stream classifier and the RatStreamNameClassifier row describe the same pointwise BEDC comparison.

The finite-window face is Definition 38.553. It reads the certified unary anchor and bound rows and returns the RealUnaryStreamWindowClassifier row for the same observation maps.

The selected-endpoint face is Definition 38.554. It projects a certified observation to the endpoint RatHistoryClassifier row, the two positive-denominator rows, and the associated unary and nonempty endpoint facts. These faces together are exactly the Phase Real stream-observation obligation surface. □

Fit. Traditional target: classifier exactness for certified countable rational observations. BEDC source: StreamName$^{\uparrow}$+Real$^{\uparrow}$ over BHist. Carrier: RatStreamNameCarrier. Classifier pair: RealUnaryStreamClassifier and RatStreamNameClassifier. Stability: the existing carrier rows in Definition 38.273. Ledger: the unary observation witness. Core theorem: Definition 38.551. Exactness: the displayed if-and-only-if. Standard bridge: pointwise rational stream equality. Status: ProofTarget.

Theorem 38.553 (Real unary StreamNameUp finite-window coverage)**.** *For observation maps* s, t : Hist → Hist *and histories* a, w : Hist*, if*

RatStreamNameCarrier(s), RatStreamNameCarrier(t), RatStreamNameClassifier(s, t), UnaryHistory(a), Unary

then the certified StreamNameUp classifier supplies the finite RealUp observation window:

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w).$$

Proof. Use the reverse direction of Definition 38.551 with the two carrier rows. The StreamNameUp classifier hypothesis therefore gives $\mathsf{RealUnaryStreamClassifier}(s, t)$.

The remaining hypotheses are exactly the unary anchor and unary bound required by Definition 38.155. Applying that coverage theorem to s, t, a, and w yields $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RatStreamNameClassifier real unary window coverage`

Fit. Traditional target: finite windows for certified countable rational observations. BEDC source: StreamNameUp plus RealUp unary stream classifier. Carrier: observation maps over BHist. Classifier: RealUnaryStreamWindowClassifier. Stability: inherited from Definition 38.551 and Definition 38.155. Ledger: anchor, bound, and bounded offset. Core theorem: Definition 38.553. Exactness: bounded pointwise RatUp classification. Standard bridge: finite rational observation windows. Status: ProofTarget.

Theorem 38.554 (Real unary StreamNameUp selected endpoint denominator package)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, a certified observation $n : \mathsf{Hist}$, and histories $z_s, z_t : \mathsf{Hist}$, assume*

$$\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(n).$$

Then the selected observation carries the RatUp denominator package:

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(s(n), t(n)) \wedge \mathsf{PositiveUnaryDenominator}(s(n)) \wedge \mathsf{PositiveUnaryDenominator}(t(n)) \wedge \\ &\mathsf{UnaryHistory}(s(n)) \wedge \mathsf{UnaryHistory}(t(n)) \wedge (\mathsf{hsame}(s(n), \mathsf{e}) \to \mathsf{False}) \wedge \\ &(\mathsf{hsame}(t(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(s(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{E0}(z_t)) \to \mathsf{False}).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RatStreamNameClassifier selected endpoint denominator package`

Proof. Project $\mathsf{RatStreamNameClassifier}(s, t)$ at the certified observation n. The unary certificate for n gives the point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.50 to this point classifier, and then apply Definition 37.12 separately to the two positive-denominator witnesses. This gives the two positive denominator rows, the two unary endpoint rows, and the two nonempty exclusions.

For a zero-headed display on the left, transport $\mathsf{PositiveUnaryDenominator}(s(n))$ along $\mathsf{hsame}(s(n), \mathsf{E0}(z_s))$ by Definition 37.6, contradicting Definition 37.10. The right endpoint uses the same argument with $t(n)$ and z_t, and the collected rows form the displayed package. □

Fit. Traditional target: selected rational observation readback. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: observation maps. Classifier: RatStreamNameClassifier. Stability: point projection and RatUp denominator transport. Ledger: selected observation and zero-head tests. Core theorem: Definition 38.554. Exactness: the displayed endpoint package. Standard bridge: finite rational observation readback. Status: ProofTarget.

Theorem 38.555 (Real unary StreamNameUp transported selected endpoint shape package)**.** *For unary observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, a certified observation $n : \mathsf{Hist}$, and histories $a, b, z_s, z_t : \mathsf{Hist}$, assume*

$$\mathsf{UnaryHistory}(n), \quad \mathsf{RealUnaryStreamClassifier}(s, t),$$

the pointwise transport rows

$$\forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(s(k), s'(k)), \quad \forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(t(k), t'(k)),$$

and the displayed selected endpoints

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)).$$

Then the transported selected endpoints carry the full RatUp denominator and one-tail shape package:

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(s'(n), t'(n)) \wedge \mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(t'(n)) \wedge \\ &\mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(t'(n)) \wedge (\mathsf{hsame}(s'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{e}) \to \mathsf{False}) \wedge \\ &(\mathsf{hsame}(s'(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\ &\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).\end{aligned}$$

Proof. Evaluate $\mathsf{RealUnaryStreamClassifier}(s, t)$ at the certified observation n. The assumption $\mathsf{UnaryHistory}(n)$ gives the point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.54 to this point classifier and to the two pointwise transport rows evaluated at n. This gives the transported selected classifier $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.

Apply Definition 37.50 to the transported selected classifier, then apply Definition 37.12 separately to the two positive-denominator witnesses. These rows give the two positive denominators, the two unary endpoint rows, and the two nonempty exclusions.

It remains to rule out zero-headed displays. If $\mathsf{hsame}(s'(n), \mathsf{E0}(z_s))$ holds, transport $\mathsf{PositiveUnaryDenominator}(s'$ along that comparison by Definition 37.6, then contradict Definition 37.10. The same argument for $t'(n)$ and z_t gives the second zero-headed exclusion.

For the one-tail readback, transport $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ along the two displayed $\mathsf{E1}$ endpoint comparisons, obtaining $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$. Apply Definition 37.49 in the forward direction to read back $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$, then package all displayed rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamClassifier transported selected endpoint shape package`

Fit. Traditional target: selected unary stream observation readback. BEDC source: RealUp unary streams plus RatUp classifier. Carrier: unary-indexed rational observation maps. Classifier: RealUnaryStreamClassifier. Stability: pointwise hsame transport. Ledger: selected certified observation and endpoint displays. Core theorem: Definition 38.555. Exactness: displayed endpoint package. Standard bridge: selected rational observation readback for RealUp names. Status: ProofTarget.

Theorem 38.556 (Rat StreamName unary-context full endpoint package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, a selected observation* $n : \mathsf{Hist}$*, prefix and tail histories* $p_s, p_t, q_s, q_t : \mathsf{Hist}$*, middle histories* $m_s, m_t : \mathsf{Hist}$*, output histories* $o_s, o_t : \mathsf{Hist}$*, displayed tails* $a, b : \mathsf{Hist}$*, and zero-head tests* $z_s, z_t : \mathsf{Hist}$*, assume*

$$\begin{aligned}&\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(n), \quad \mathsf{UnaryHistory}(p_s), \quad \mathsf{hsame}(p_s, p_t), \\ &\mathsf{UnaryHistory}(q_s), \quad \mathsf{hsame}(q_s, q_t), \quad \mathsf{Cont}(p_s, s(n), m_s), \quad \mathsf{Cont}(m_s, q_s, o_s), \\ &\mathsf{Cont}(p_t, t(n), m_t), \quad \mathsf{Cont}(m_t, q_t, o_t), \quad \mathsf{hsame}(o_s, \mathsf{E1}(a)), \quad \mathsf{hsame}(o_t, \mathsf{E1}(b)).\end{aligned}$$

Then the two contextual outputs carry the full RatUp endpoint package:

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(o_s, o_t) \wedge \mathsf{PositiveUnaryDenominator}(o_s) \wedge \mathsf{PositiveUnaryDenominator}(o_t)\wedge\\&\mathsf{UnaryHistory}(o_s) \wedge \mathsf{UnaryHistory}(o_t) \wedge (\mathsf{hsame}(o_s, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_t, \mathsf{e}) \to \mathsf{False})\wedge\\&(\mathsf{hsame}(o_s, \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_t, \mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).\end{aligned}$$

Proof. Project the point classifier from $\mathsf{RatStreamNameClassifier}(s, t)$ at the certified observation n. This gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Build the contextual output classifier by applying Definition 37.58 to the point classifier and the matching unary prefix and tail contexts, then use the four continuation displays and associativity of the continuation append to transport the closed classifier to o_s and o_t.

Apply Definition 45.136 to the same point classifier and the displayed continuation context. This exposes positive unary denominator witnesses for o_s and o_t. Applying Definition 37.12 separately gives the unary-history rows and the two exclusions against e.

For a zero-headed display on the left, transport $\mathsf{PositiveUnaryDenominator}(o_s)$ along $\mathsf{hsame}(o_s, \mathsf{E0}(z_s))$ and contradict Definition 37.10. The right endpoint uses the same transport argument with o_t and z_t.

Finally apply Definition 126.12 to the original StreamNameUp classifier, the certified observation, the matching contexts, the four continuation rows, and the two displayed $\mathsf{E1}$ outputs. This reads back $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$. Collecting the classifier, denominator, exclusion, and readback rows gives the displayed endpoint package. □

Fit. Traditional target: unary-context endpoint readback for certified rational stream names. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: observation maps with unary BEDC observations. Classifier: $\mathsf{RatStreamNameClassifier}$. Stability: point projection, RatUp unary context closure, continuation readback, and positive-denominator transport. Ledger: selected observation, matching prefix and tail contexts, continuation rows, endpoint displays, and zero-head tests. Core theorem: Definition 38.556. Exactness: the displayed contextual endpoint package. Standard bridge: concrete BHist-anchored rational observation contexts for RealUp names. Status: $\mathsf{ProofTarget}$.

Theorem 38.557 (Rat StreamName unary-context e-one tail coverage package)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, a selected observation $n : \mathsf{Hist}$, prefix and tail histories $p_s, p_t, q_s, q_t : \mathsf{Hist}$, middle histories $m_s, m_t : \mathsf{Hist}$, output histories $o_s, o_t : \mathsf{Hist}$, and zero-head tests $z_s, z_t : \mathsf{Hist}$, assume*

$$\begin{aligned}&\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(n), \quad \mathsf{UnaryHistory}(p_s), \quad \mathsf{hsame}(p_s, p_t),\\&\mathsf{UnaryHistory}(q_s), \quad \mathsf{hsame}(q_s, q_t), \quad \mathsf{Cont}(p_s, s(n), m_s), \quad \mathsf{Cont}(m_s, q_s, o_s),\\&\mathsf{Cont}(p_t, t(n), m_t), \quad \mathsf{Cont}(m_t, q_t, o_t).\end{aligned}$$

Then the local unary-context hypotheses construct displayed one-extension tails for the contextual outputs and carry the full RatUp endpoint package:

$$\begin{aligned}&\exists a, b : \mathsf{Hist},\ \mathsf{hsame}(o_s, \mathsf{E1}(a)) \wedge \mathsf{hsame}(o_t, \mathsf{E1}(b))\wedge\\&\mathsf{RatHistoryClassifier}(o_s, o_t) \wedge \mathsf{PositiveUnaryDenominator}(o_s) \wedge \mathsf{PositiveUnaryDenominator}(o_t)\wedge\\&\mathsf{UnaryHistory}(o_s) \wedge \mathsf{UnaryHistory}(o_t) \wedge (\mathsf{hsame}(o_s, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_t, \mathsf{e}) \to \mathsf{False})\wedge\\&(\mathsf{hsame}(o_s, \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_t, \mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).\end{aligned}$$

Proof. Project the point classifier from $\mathsf{RatStreamNameClassifier}(s,t)$ at the certified unary observation n. This gives $\mathsf{RatHistoryClassifier}(s(n),t(n))$.

Apply Definition 45.136 to that point classifier, the two matching unary contexts, and the four continuation rows. This supplies $\mathsf{PositiveUnaryDenominator}(o_s)$ and $\mathsf{PositiveUnaryDenominator}(o_t)$.

Unfold the two positive unary denominator witnesses. Choose tails a and b together with the displayed one-extension rows $\mathsf{hsame}(o_s,\mathsf{E1}(a))$ and $\mathsf{hsame}(o_t,\mathsf{E1}(b))$.

With those displayed tails, apply Definition 38.556 to the original StreamNameUp classifier, the certified observation, the matching context rows, the four continuation rows, and the two selected one-extension displays. The returned classifier, denominator, unary, exclusion, and tail readback rows combine with the two displays under $\exists a,b$. □

Fit. Traditional target: unary-context one-tail coverage for certified rational stream names. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Rat}^{\uparrow}$ over BHist. Carrier: observation maps with unary BEDC observations. Classifier: $\mathsf{RatStreamNameClassifier}$. Stability: point projection, positive denominator extraction, and full contextual endpoint packaging. Ledger: selected observation, matching prefix and tail contexts, continuation rows, and zero-head tests. Core theorem: Definition 38.557. Exactness: existential one-extension tails with the displayed contextual endpoint package. Standard bridge: RegSeqRat/RealUp selected contextual observation readback. Status: $\mathsf{ProofTarget}$.

Theorem 38.558 (Rat StreamName unary-context e-one tail determinacy)**.** *For observation maps $s,t:\mathsf{Hist}\to\mathsf{Hist}$, a selected observation $n:\mathsf{Hist}$, prefix and tail histories $p_s,p_t,q_s,q_t:\mathsf{Hist}$, middle histories $m_s,m_t:\mathsf{Hist}$, output histories $o_s,o_t:\mathsf{Hist}$, and displayed tails $a,a',b,b':\mathsf{Hist}$, assume*

$$\begin{aligned}&\mathsf{RatStreamNameClassifier}(s,t),\quad \mathsf{UnaryHistory}(n),\quad \mathsf{UnaryHistory}(p_s),\quad \mathsf{hsame}(p_s,p_t),\\&\mathsf{UnaryHistory}(q_s),\quad \mathsf{hsame}(q_s,q_t),\quad \mathsf{Cont}(p_s,s(n),m_s),\quad \mathsf{Cont}(m_s,q_s,o_s),\\&\mathsf{Cont}(p_t,t(n),m_t),\quad \mathsf{Cont}(m_t,q_t,o_t),\end{aligned}$$

and the four e-one displays

$$\mathsf{hsame}(o_s,\mathsf{E1}(a)),\quad \mathsf{hsame}(o_s,\mathsf{E1}(a')),\quad \mathsf{hsame}(o_t,\mathsf{E1}(b)),\quad \mathsf{hsame}(o_t,\mathsf{E1}(b')).$$

Then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$;
2. $\mathsf{UnaryHistory}(a')$;
3. $\mathsf{UnaryHistory}(b)$;
4. $\mathsf{UnaryHistory}(b')$;
5. $\mathsf{hsame}(a,a')$;
6. $\mathsf{hsame}(a,b)$;
7. $\mathsf{hsame}(a,b')$;
8. $\mathsf{hsame}(a',b)$;
9. $\mathsf{hsame}(a',b')$;
10. $\mathsf{hsame}(b,b')$.

Proof. Apply Definition 38.556 to the displays $\mathsf{hsame}(o_s,\mathsf{E1}(a))$ and $\mathsf{hsame}(o_t,\mathsf{E1}(b))$. Its tail rows supply $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a,b)$.

Apply the same full endpoint package again to the displays $\mathsf{hsame}(o_s, \mathsf{E1}(a'))$ and $\mathsf{hsame}(o_t, \mathsf{E1}(b'))$. This supplies $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the left-output tail comparison, compose the first display of o_s with the symmetry of the second display. The resulting $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ is inverted by BEDC.FKernel.Hist.hsame_e1_inversion_iff, giving $\mathsf{hsame}(a, a')$.

The same composition and e-one inversion applied to the two displays of o_t gives $\mathsf{hsame}(b, b')$. Transitivity of hsame, using $\mathsf{hsame}(a, b)$ together with symmetry of $\mathsf{hsame}(a', b')$ and the two same-side comparisons, gives the remaining mixed rows $\mathsf{hsame}(a, b')$ and $\mathsf{hsame}(a', b)$. Package the four unary rows with the six displayed tail comparisons. □

Fit. Traditional target: selected contextual rational-stream tail determinacy. BEDC source: $\mathsf{StreamName}^{\uparrow}$+$\mathsf{Rat}^{\uparrow}$+BHist. Carrier: observation maps over Hist. Classifier: RatStreamNameClassifier. Stability: unary context closure, e-one no-confusion, and hsame transitivity. Ledger: selected observation, context rows, continuation rows, and endpoint displays. Core theorem: Definition 38.558. Exactness: uniqueness of displayed e-one tails. Standard bridge: RegSeqRat selected observation canonicalization. Status: ProofTarget.

Theorem 38.559 (Rat StreamName unary-context bundled tail determinacy)**.** *Let* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *be public rational* $\mathsf{StreamName}^{\uparrow}$ *rows with* $\mathsf{RatStreamNameClassifier}(s, t)$*. Let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite bundle of unary observation requests. Suppose that every bundled request* n *with* $\mathsf{InBundle}(n, \mathcal{P})$ *carries* $\mathsf{UnaryHistory}(n)$*, shared prefix rows* $p_s(n), p_t(n) : \mathsf{Hist}$*, shared tail-context rows* $q_s(n), q_t(n) : \mathsf{Hist}$*, middle rows* $m_s(n), m_t(n) : \mathsf{Hist}$*, output rows* $o_s(n), o_t(n) : \mathsf{Hist}$*, and displayed tails* $a(n), a'(n), b(n), b'(n) : \mathsf{Hist}$ *satisfying*

$$\begin{aligned}\mathsf{UnaryHistory}(p_s(n)),\quad \mathsf{hsame}(p_s(n), p_t(n)),\quad \mathsf{UnaryHistory}(q_s(n)),\quad \mathsf{hsame}(q_s(n), q_t(n)),\\ \mathsf{Cont}(p_s(n), s(n), m_s(n)),\quad \mathsf{Cont}(m_s(n), q_s(n), o_s(n)),\\ \mathsf{Cont}(p_t(n), t(n), m_t(n)),\quad \mathsf{Cont}(m_t(n), q_t(n), o_t(n)).\end{aligned}$$

and the four e-one endpoint displays

$$\mathsf{hsame}(o_s(n), \mathsf{E1}(a(n))),\quad \mathsf{hsame}(o_s(n), \mathsf{E1}(a'(n))),\quad \mathsf{hsame}(o_t(n), \mathsf{E1}(b(n))),\quad \mathsf{hsame}(o_t(n), \mathsf{E1}(b'(n))).$$

Then the bundled displayed tail rows are unary and pairwise hsame *across all source and target endpoint reads. More explicitly, for every bundled* n*, the four rows* $a(n), a'(n), b(n), b'(n)$ *are unary, and the six comparisons*

$$\begin{aligned}\mathsf{hsame}(a(n), a'(n)),\quad \mathsf{hsame}(a(n), b(n)),\quad \mathsf{hsame}(a(n), b'(n)),\\ \mathsf{hsame}(a'(n), b(n)),\quad \mathsf{hsame}(a'(n), b'(n)),\quad \mathsf{hsame}(b(n), b'(n))\end{aligned}$$

hold inside the same finite bundle surface. This surface is exactly the finite source surface consumed by the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff: the bundled unary rows, their* InBundle *witnesses, the shared prefix–tail context, the continuation rows, the endpoint displays, the pointwise rational-stream handoff rows, and the support rows* hsame, Cont, Pkg, NameCert*. It contains no quotient stream equality, selected limit, host real equality, metric-completion point, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Fix a bundled observation n and its membership row $\mathsf{InBundle}(n, \mathcal{P})$. Apply Definition 38.558 to the displayed prefix, tail-context, continuation, output, and e-one rows attached to n. This gives the four unary tail rows and the six pairwise hsame comparisons for that bundled request.

Repeat this pointwise over the finite ProbeBundle. The InBundle membership witnesses keep each selected observation inside $\mathcal{P}$, so the collected rows are one finite family rather than a hidden choice of endpoint tails.

Finally apply Definition 126.547. Its cofinal packet identifies the collected unary observations, membership rows, shared tail-compatible context, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, and support rows as the finite bundled handoff surface. The non-escape clauses in that lock exclude quotient stream equality, selected limits, host real equality, metric-completion data, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the surface just constructed. □

Fit. Traditional target: finite contextual rational-stream tail determinacy. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow} + \mathsf{ProbeBundle} + \mathsf{BHist}$. Carrier: public rational StreamName rows over Hist with bundled unary requests. Classifier: RatStreamNameClassifier together with InBundle membership. Stability: pointwise unary-context closure, e-one no-confusion, finite-bundle collection, and tail-compatible cofinal locking. Ledger: bundled observation rows, shared prefix and tail-context rows, continuation rows, endpoint displays, pointwise rational handoff rows, and Pkg/NameCert support. Core theorem: Definition 38.559. Exactness: all displayed source and target e-one tails in the bundle are unary and pairwise hsame. Standard bridge: finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff surface. Status: ProofTarget.

Definition 38.560 (Real unary selected denominator row)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and a certified unary observation* $n : \mathsf{Hist}$*, the selected denominator row records the complete RatUp denominator data at that observation:*

$$\begin{aligned}
&\mathsf{RealUnarySelectedDenominatorRow}(s,t,n) \iff \\
&\mathsf{RatHistoryClassifier}(s(n),t(n)) \wedge \mathsf{PositiveUnaryDenominator}(s(n)) \wedge \mathsf{PositiveUnaryDenominator}(t(n)) \wedge \\
&\mathsf{UnaryHistory}(s(n)) \wedge \mathsf{UnaryHistory}(t(n)) \wedge (\mathsf{hsame}(s(n),\mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(t(n),\mathsf{e}) \to \mathsf{False}) \wedge \forall z : \mathsf{Hist}, ((\mathsf{hsame}(s(n),\mathsf{E0}(z)) \to \mathsf{False}) \wedge \\
&\qquad\qquad (\mathsf{hsame}(t(n),\mathsf{E0}(z)) \to \mathsf{False})).
\end{aligned}$$

Theorem 38.561 (Real unary StreamNameUp pointwise denominator coverage)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s) \quad \textit{and} \quad \mathsf{RatStreamNameCarrier}(t).$$

Then the following three clauses are equivalent, in the sense that any one of them constructs the other two:

1. $\mathsf{RatStreamNameClassifier}(s,t)$.
2. $\mathsf{RealUnaryStreamClassifier}(s,t)$.
3. $\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{RealUnarySelectedDenominatorRow}(s,t,n)$.

Lean4 ✓ `BEDC.Derived.RealUp.RealUnarySelectedDenominatorRow streamName equivalence`

Proof. The StreamNameUp and Real unary classifier clauses are the same pointwise RatUp classifier under the two carrier assumptions. Apply Definition 38.551 to identify them.

Assume the Real unary classifier and fix a certified unary observation n. Evaluating the classifier at n gives $\mathsf{RatHistoryClassifier}(s(n),t(n))$. Apply Definition 37.50 to this row, then apply Definition 37.12 separately to the two positive denominator witnesses. This supplies the classifier, both positive unary denominators, both unary endpoint rows, and both nonempty exclusions.

It remains to fill the zero-headed exclusions. For an arbitrary history z, transport the positive unary denominator witness for $s(n)$ along $\mathsf{hsame}(s(n), \mathsf{E0}(z))$ by Definition 37.6, then contradict Definition 37.10. The same argument for $t(n)$ gives the right endpoint exclusion, so the selected denominator row is constructed.

Conversely, assume the pointwise selected denominator rows. For each certified unary observation n, project the first conjunct of $\mathsf{RealUnarySelectedDenominatorRow}(s, t, n)$ to obtain $\mathsf{RatHistoryClassifier}(s(n), t(n$ Repack this pointwise classifier with the two carrier assumptions as $\mathsf{RatStreamNameClassifier}(s, t)$, and use the exactness theorem again to recover the Real unary classifier clause. □

Fit. Traditional target: certified rational observation streams for RealUp. BEDC source: $\mathsf{StreamName}^{\uparrow}$+$\mathsf{Real}^{\uparrow}$+$\mathsf{Rat}^{\uparrow}$. Carrier: RatStreamNameCarrier. Classifier pair: RatStreamNameClassifier and RealUnaryStreamClassifier. Stability: pointwise RatUp denominator transport. Ledger: unary observation index and zero-headed probes. Core theorem: Definition 38.561. Exactness: the three-way equivalence. Standard bridge: finite rational observation windows for RegSeqRat/RealUp. Status: ProofTarget.

Theorem 38.562 (RegSeqRat finite-window dyadic observation coverage)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, histories $a, w, k : \mathsf{Hist}$, and RegSeqRat stream certificate spines for s and t, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{Unary}$$

Let $n = \mathsf{append}(a, k)$. Then the selected bounded observation supplies the finite RealUp window and the dyadic endpoint rows:

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \mathsf{RatHistoryClassifier}(s(n), t(n)) \wedge \\ \mathsf{DyadicRatCoreObservationRow}(s(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(t(n)).$$

Here DyadicRatCoreObservationRow *is the package described in Definition 38.219.*

Proof. Apply Definition 38.553 to the two carrier rows, the displayed StreamNameUp classifier, and the unary anchor and bound. This constructs $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$.

Project the window at the selected offset k by Definition 38.157. The witness $\mathsf{UnaryOffsetLe}(k, w)$ supplies the bounded row, so the selected observation $n = \mathsf{append}(a, k)$ carries $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Unfolding the offset witness gives $\mathsf{UnaryHistory}(k)$. Unary append closure with $\mathsf{UnaryHistory}(a)$ gives $\mathsf{UnaryHistory}(n)$, the certified observation index needed by the two RegSeqRat spines.

Apply Definition 38.246 to the spine for s at n, and apply the same theorem to the spine for t at n. These two applications supply the DyadicRatCore observation rows for $s(n)$ and $t(n)$. Together with the selected classifier and the finite window, these are exactly the displayed package. □

Lean4 ✓ `BEDC.Derived.RealUp.RegseqratFiniteWindowDyadicObservationCoverage`

Theorem 38.563 (RegSeqRat finite-window dyadic classifier stability package)**.** *For observation maps $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, histories $a, w, k : \mathsf{Hist}$, and RegSeqRat stream certificate spines for s, t, u, s', t', assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(u), \quad \mathsf{RatStreamNameCarrier}(s'),$$

$$\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{RatStreamNameClassifier}(t, u), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffse}$$

Let $n = \mathsf{append}(a, k)$, and assume the selected pointwise transport rows

$$\mathsf{hsame}(s(n), s'(n)) \quad \textit{and} \quad \mathsf{hsame}(t(n), t'(n)).$$

Then the finite-window dyadic classifier package selected at n contains exactly the four classifier-stability faces needed by the RealUp finite-window seal:

1. *the diagonal face* $\mathsf{RatHistoryClassifier}(s(n), s(n))$ *together with* $\mathsf{DyadicRatCoreObservationRow}(s(n))$*;*
2. *the symmetry face* $\mathsf{RatHistoryClassifier}(t(n), s(n))$ *together with the swapped dyadic observation rows at* $t(n)$ *and* $s(n)$*;*
3. *the transitivity face* $\mathsf{RatHistoryClassifier}(s(n), u(n))$ *together with the dyadic observation rows at* $s(n)$ *and* $u(n)$*;*
4. *the* hsame *transport face* $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ *together with the transported dyadic observation rows at* $s'(n)$ *and* $t'(n)$*.*

The package has no additional diagonal classifier-stability constructor beyond the RatUp reflexive face and the selected RegSeqRat dyadic observation row.

Proof. First obtain the selected forward observation by Definition 38.562. It gives the $\mathsf{Real}^{\uparrow}$ finite window, the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the endpoint $\mathsf{DyadicRatCoreObservationRow}$ witnesses at $s(n)$ and $t(n)$.

For the diagonal face, project the rational carrier at $s(n)$ from the RegSeqRat spine and the selected unary index. Apply Definition 37.37 to that carrier witness. The dyadic row at the same endpoint is the one supplied by the selected observation coverage theorem, so the diagonal face is present without a separate RegSeqRat classifier reflexivity package.

For symmetry, use Definition 38.565. For transitivity, use Definition 38.566 with the two displayed StreamNameUp classifier rows. These two rows provide the reversed and composed pointwise RatUp classifiers together with the corresponding DyadicRatCore observation rows.

For transport, use Definition 38.567 with the displayed hsame rows. It preserves the original finite $\mathsf{Real}^{\uparrow}$ window and transports the selected RatUp classifier to $s'(n)$ and $t'(n)$ while exposing the DyadicRatCore rows at the transported endpoints.

Thus the package is assembled from $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{DyadicRatCore}$: StreamNameUp supplies the stream classifier rows, RatUp supplies the pointwise reflexive, symmetric, transitive, and transported classifiers, RealUp supplies the bounded unary window, and DyadicRatCore supplies the endpoint observation rows. These are the carrier, classifier, stability, ledger, core theorem, exactness, bridge, and downstream-use faces of the roadmap template, with no extra diagonal classifier-stability source. □

Fit. Traditional target: finite-window dyadic classifier stability for RegSeqRat. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Rat}^{\uparrow}+\mathsf{Real}^{\uparrow}+\mathsf{DyadicRatCore}$. Carrier/classifier: $\mathsf{RatStreamNameCarrier}$ rows for the participating streams, the source and composed $\mathsf{RatStreamNameClassifier}$ rows, and the selected pointwise $\mathsf{RatHistoryClassifier}$ faces. Stability: RatUp reflexivity, symmetry, transitivity, and hsame transport at the selected endpoint. Ledger: the unary observation index $n = \mathsf{append}(a, k)$ and the finite $\mathsf{Real}^{\uparrow}$ window. Core theorem: Definition 38.563. Exactness: the diagonal face is read from Definition 37.37 and Definition 38.562; the other faces are read from the finite-window symmetry, transitivity, and transport rows. Standard bridge: the RealUp finite-window seal. Status: $\mathsf{ProofTarget}$.

Theorem 38.564 (RegSeqRat finite-window dyadic source exhaustion)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, histories* $a, w, k : \mathsf{Hist}$*, and RegSeqRat stream certificate spines for* s *and* t*, assume*

$\mathsf{RatStreamNameCarrier}(s)$, $\mathsf{RatStreamNameCarrier}(t)$, $\mathsf{RatStreamNameClassifier}(s, t)$, $\mathsf{UnaryHistory}(a)$, Unary

Let $n = \mathsf{append}(a, k)$. Then the selected bounded observation exhausts the scoped DyadicRatCore source package at the two endpoints $s(n)$ and $t(n)$:

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \mathsf{RatHistoryClassifier}(s(n), t(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(s(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(t(n)).$$

Moreover, the carrier, classifier, stability, and finite ledger faces read at $s(n)$ and $t(n)$ are exactly the faces named by Definition 38.221.

Proof. Apply Definition 38.562. It supplies the finite $\mathsf{Real}^{\uparrow}$ window, the selected pointwise $\mathsf{RatHistoryClassifier}(s(n)$, and the two $\mathsf{DyadicRatCoreObservationRow}$ witnesses at $s(n)$ and $t(n)$.

Next identify the source faces. Apply Definition 38.221 to the two endpoint rows. Its carrier face is the rational-history carrier together with positive unary denominator and dyadic-radius data; its classifier face is the displayed $\mathsf{RatHistoryClassifier}$ row together with hsame transport; its stability and finite ledger faces are the ones already recorded by the RegSeqRat spine.

No further package data are read. The selected index $n = \mathsf{append}(a, k)$ comes from the finite RealUp window, and the endpoint source data are exactly the two DyadicRatCore observation rows identified with the scoped package faces. Thus the bounded RegSeqRat spine and the DyadicRatCore observation row exhaust the scoped source package at both endpoints. □

Theorem 38.565 (RegSeqRat finite-window dyadic classifier symmetry coverage)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, histories $a, w, k : \mathsf{Hist}$, and RegSeqRat stream certificate spines for s and t, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{Unary}$$

Let $n = \mathsf{append}(a, k)$. Then the reversed selected bounded observation supplies

$$\mathsf{RealUnaryStreamWindowClassifier}(t, s, a, w) \wedge \mathsf{RatHistoryClassifier}(t(n), s(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(t(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(s(n)).$$

Moreover, the carrier, classifier, stability, and finite ledger faces read at $t(n)$ and $s(n)$ are exactly the faces named by Definition 38.221.

Proof. Apply Definition 38.562 to the pair (s, t). It supplies the forward $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$, the selected classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the two endpoint dyadic rows at $s(n)$ and $t(n)$.

Reverse the StreamNameUp classifier by the symmetry clause of Definition 126.13. Apply Definition 38.553 to this reversed classifier, the carrier rows for t and s, and the same unary anchor and bound. This gives $\mathsf{RealUnaryStreamWindowClassifier}(t, s, a, w)$.

For the selected point, apply Definition 37.55 to the pointwise classifier obtained from the forward package. The result is exactly $\mathsf{RatHistoryClassifier}(t(n), s(n))$.

The two $\mathsf{DyadicRatCoreObservationRow}$ witnesses are reused in swapped order. Applying Definition 38.221 to those two rows identifies the carrier, classifier, stability, and finite ledger faces with the same scoped source package, now read at the reversed endpoints $t(n)$ and $s(n)$. □

Theorem 38.566 (RegSeqRat finite-window dyadic classifier transitivity coverage)**.** *For observation maps $s, t, u : \mathsf{Hist} \to \mathsf{Hist}$, histories $a, w, k : \mathsf{Hist}$, and RegSeqRat stream certificate spines for s, t, and u, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(u), \quad \mathsf{RatStreamNameClassifier}(s, t$$

$$\mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w).$$

Let $n = \mathsf{append}(a, k)$*. Then the composed selected bounded observation supplies*

$$\mathsf{RealUnaryStreamWindowClassifier}(s, u, a, w) \wedge \mathsf{RatHistoryClassifier}(s(n), u(n)) \wedge \\ \mathsf{DyadicRatCoreObservationRow}(s(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(u(n)).$$

Moreover, the carrier, classifier, stability, and finite ledger faces read at $s(n)$ *and* $u(n)$ *are exactly the faces named by Definition 38.221.*

Proof. Apply the transitivity clause of Definition 126.13 to the two displayed classifier rows. This composes $\mathsf{RatStreamNameClassifier}(s, t)$ and $\mathsf{RatStreamNameClassifier}(t, u)$ into $\mathsf{RatStreamNameClassifier}(s, u)$.

Now apply Definition 38.562 to the pair (s, u), the RegSeqRat spines for s and u, the carrier rows for s and u, the composed StreamNameUp classifier, and the unary anchor, bound, and offset. This yields the finite $\mathsf{RealUnaryStreamWindowClassifier}(s, u, a, w)$, the selected pointwise $\mathsf{RatHistoryClassifier}(s(n), u(n))$, and the endpoint dyadic rows for $s(n)$ and $u(n)$.

Finally identify the source package faces. Apply Definition 38.221 to the two endpoint rows. The selected unary offset supplies the finite ledger index $n = \mathsf{append}(a, k)$, while the two endpoint RegSeqRat spines supply the carrier, classifier, and stability faces already named by the dyadic scoped source package. Hence the composed finite-window observation carries exactly the DyadicRatCore source package at the endpoints $s(n)$ and $u(n)$. □

Theorem 38.567 (RegSeqRat finite-window dyadic transport package)**.** *For observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w, k : \mathsf{Hist}$*, assume RegSeqRat stream certificate spines for all four streams and*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(s'), \quad \mathsf{RatStreamNameCarrier}(t'),$$

$$\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w).$$

Let $n = \mathsf{append}(a, k)$*, and assume the selected pointwise transport rows*

$$\mathsf{hsame}(s(n), s'(n)) \quad \textit{and} \quad \mathsf{hsame}(t(n), t'(n)).$$

Then the transported selected endpoints carry the finite-window dyadic package

$$\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \\ \mathsf{RatHistoryClassifier}(s'(n), t'(n)) \wedge \\ \mathsf{DyadicRatCoreObservationRow}(s'(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(t'(n)).$$

Here DyadicRatCoreObservationRow *is the package described in Definition 38.219.*

Proof. First apply Definition 38.562 to the untransported streams s, t, their RegSeqRat spines, the two RatStreamNameCarrier rows, the displayed StreamNameUp classifier, and the unary anchor, bound, and offset. This supplies the original $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$, the selected $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the DyadicRatCore observation rows at $s(n)$ and $t(n)$.

Transport the selected classifier next. Apply Definition 37.54 to the selected $\mathsf{RatHistoryClassifier}(s(n), t(n))$ and to the two pointwise hsame rows. The result is the transported selected classifier $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$

It remains to expose the transported DyadicRatCore rows. The offset witness gives the certified unary row for k, and unary append closure with $\mathsf{UnaryHistory}(a)$ gives $\mathsf{UnaryHistory}(n)$. Apply Definition 38.246 to the RegSeqRat spine for s' at n, and apply the same theorem to the spine for

t' at n. These two applications provide the dyadic-radius ledger tokens and endpoint carrier faces for $s'(n)$ and $t'(n)$.

The denominator face is stable under the selected pointwise transports. From the untransported DyadicRatCore rows, project the positive unary denominator rows at $s(n)$ and $t(n)$, then transport them along Definition 37.6 using $\mathsf{hsame}(s(n), s'(n))$ and $\mathsf{hsame}(t(n), t'(n))$. These transported denominator rows agree with the denominator components of the s' and t' DyadicRatCore packages supplied by the previous paragraph.

Repack the original finite window, the transported selected classifier, and the two transported DyadicRatCore observation rows. This is exactly the displayed finite-window dyadic transport package. □

Theorem 38.568 (RegSeqRat finite-window transported dyadic source exhaustion)**.** *For observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ and histories $a, w, k : \mathsf{Hist}$, assume RegSeqRat stream certificate spines for all four streams and*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(s'), \quad \mathsf{RatStreamNameCarrier}(t'),$$

$$\mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w).$$

Let $n = \mathsf{append}(a, k)$, and assume the selected pointwise transport rows

$$\mathsf{hsame}(s(n), s'(n)) \quad \textit{and} \quad \mathsf{hsame}(t(n), t'(n)).$$

Then the transported selected observation exhausts the scoped DyadicRatCore source package at the endpoints $s'(n)$ and $t'(n)$:

$$\begin{aligned}&\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w) \wedge \\ &\mathsf{RatHistoryClassifier}(s'(n), t'(n)) \wedge \\ &\mathsf{DyadicRatCoreObservationRow}(s'(n)) \wedge \mathsf{DyadicRatCoreObservationRow}(t'(n)).\end{aligned}$$

Moreover, the carrier, classifier, stability, and finite ledger faces read at $s'(n)$ and $t'(n)$ are exactly the faces named by Definition 38.221.

Proof. First apply Definition 38.567. It supplies the transported finite-window dyadic package: the original $\mathsf{Real}^{\uparrow}$ window, the transported selected classifier $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$, and the two transported $\mathsf{DyadicRatCoreObservationRow}$ witnesses at $s'(n)$ and $t'(n)$.

Now identify the scoped source reads. Apply Definition 38.221 to the endpoint rows $s'(n)$ and $t'(n)$. The carrier face is the rational-history carrier, positive unary denominator, and dyadic-radius token stored in each transported DyadicRatCore observation row. The classifier face is the transported $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ together with the selected hsame stability rows.

The ledger is finite for the same reason as in the transport package. The index $n = \mathsf{append}(a, k)$ is selected by the unary anchor and bounded offset, and the dyadic-radius token read at each transported endpoint is the token carried by the corresponding RegSeqRat spine. Thus the transported classifier and endpoint rows read no data beyond the scoped DyadicRatCore carrier, classifier, stability, and finite ledger faces. □

Theorem 38.569 (RegSeqRat finite-window dyadic transport composition)**.** *For observation maps $s, t, s', t', s'', t'' : \mathsf{Hist} \to \mathsf{Hist}$ and histories $a, w, k : \mathsf{Hist}$, assume RegSeqRat stream certificate spines for all six streams and*

$$\begin{aligned}&\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameCarrier}(s'), \\ &\mathsf{RatStreamNameCarrier}(t'), \quad \mathsf{RatStreamNameCarrier}(s''), \quad \mathsf{RatStreamNameCarrier}(t''),\end{aligned}$$

$$\mathsf{RatStreamNameClassifier}(s,t),\quad \mathsf{UnaryHistory}(a),\quad \mathsf{UnaryHistory}(w),\quad \mathsf{UnaryOffsetLe}(k,w).$$

Let $n = \mathsf{append}(a,k)$*, and assume the selected pointwise transport rows*

$$\mathsf{hsame}(s(n),s'(n)),\quad \mathsf{hsame}(t(n),t'(n)),\quad \mathsf{hsame}(s'(n),s''(n)),\quad \mathsf{hsame}(t'(n),t''(n)).$$

Then the twice-transported selected endpoints carry the finite-window dyadic package and exhaust the scoped DyadicRatCore source package at $s''(n)$ *and* $t''(n)$*:*

$$\begin{aligned}&\mathsf{RealUnaryStreamWindowClassifier}(s,t,a,w)\wedge\\&\mathsf{RatHistoryClassifier}(s''(n),t''(n))\wedge\\&\mathsf{DyadicRatCoreObservationRow}(s''(n))\wedge\mathsf{DyadicRatCoreObservationRow}(t''(n)).\end{aligned}$$

Moreover, the carrier, classifier, stability, and finite ledger faces read at $s''(n)$ *and* $t''(n)$ *are exactly the faces named by Definition 38.221.*

Proof. First compose the selected endpoint transports. Apply Definition 3.8 to $\mathsf{hsame}(s(n),s'(n))$ and $\mathsf{hsame}(s'(n),s''(n))$, and apply the same theorem to $\mathsf{hsame}(t(n),t'(n))$ and $\mathsf{hsame}(t'(n),t''(n))$. This gives direct rows $\mathsf{hsame}(s(n),s''(n))$ and $\mathsf{hsame}(t(n),t''(n))$ at the selected index.

Next apply Definition 38.567 with target streams s'' and t''. The six-stream hypotheses contain the four rows it needs for s,t,s'',t'', the source classifier, the unary anchor and bound, the bounded offset, and the composed selected hsame rows. Hence it supplies the original $\mathsf{Real}^{\uparrow}$ finite window, the selected $\mathsf{RatHistoryClassifier}(s''(n),t''(n))$, and the two DyadicRatCoreObservationRow witnesses at $s''(n)$ and $t''(n)$.

Finally identify the package faces. Apply Definition 38.221 to the two endpoint rows at $s''(n)$ and $t''(n)$. The carrier face is read from the transported DyadicRatCore rows, the classifier face is the twice-transported selected RatUp classifier, the stability face is the composed hsame transport, and the finite ledger is the unary selected index together with the dyadic-radius tokens stored by the RegSeqRat spines. □

Theorem 38.570 (RegSeqRat finite-window dyadic classifier determinacy)**.** *Let* $s,t:\mathsf{Hist}\to\mathsf{Hist}$ *be carried RegSeqRat streams, let* $a,w,k:\mathsf{Hist}$ *satisfy* $\mathsf{UnaryHistory}(a)$*,* $\mathsf{UnaryHistory}(w)$*, and* $\mathsf{UnaryOffsetLe}(k,w)$*, and put* $n=\mathsf{append}(a,k)$*. Suppose two selected finite-window records have the same unary window, the same RegSeqRat stream certificate spines, the DyadicRatCore observation rows at* $s(n)$ *and* $t(n)$*, and two finite-window dyadic classifier packages selecting the same* $\mathsf{RatHistoryClassifier}(s(n),t(n))$*. Then the selected* $\mathsf{Real}^{\uparrow}$ *finite observation endpoints and their ledger provenance are classified by the same displayed* hsame *row. No quotient of rational histories, host real equality, or external comparison of reals is exported.*

Proof. Apply Definition 38.562 to the shared unary window. It supplies the two pointwise DyadicRatCore observation rows and identifies their finite ledger with the selected index n and the denominator tokens stored in the RegSeqRat spines.

Next apply Definition 38.563. The two dyadic classifier packages name the same $\mathsf{RatHistoryClassifier}(s(n),t(n))$, so their classifier readbacks have the same displayed stability row over the selected endpoints.

Transport each package through Definition 38.567, using the identity endpoint rows supplied by the shared observation records. Then use Definition 38.569 to compose the two readback routes.

The composed route is a single hsame comparison between the selected $\mathsf{Real}^{\uparrow}$ finite observation endpoints. Its provenance is exactly the shared finite-window ledger, so determinacy is internal to the displayed classifier package and exports no ambient equality of real numbers. □

Theorem 38.571 (RegSeqRat finite-window dyadic closure package)**.** *For observation maps* $s, t, s', t', s'', t'' : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $a, w, k : \mathsf{Hist}$*, assume RegSeqRat stream certificate spines for* s, t, s', t', s'', t''*, the source classifier* $\mathsf{RatStreamNameClassifier}(s, t)$*, and*

$$\mathsf{UnaryHistory}(a), \quad \mathsf{UnaryHistory}(w), \quad \mathsf{UnaryOffsetLe}(k, w).$$

Let $n = \mathsf{append}(a, k)$*, and assume two-step selected endpoint transports*

$$\mathsf{hsame}(s(n), s'(n)), \quad \mathsf{hsame}(t(n), t'(n)), \quad \mathsf{hsame}(s'(n), s''(n)), \quad \mathsf{hsame}(t'(n), t''(n)).$$

Then the selected finite-window surface at the final endpoints simultaneously contains the $\mathsf{Real}^{\uparrow}$ *window* $\mathsf{RealUnaryStreamWindowClassifier}(s, t, a, w)$*, the final selected classifier* $\mathsf{RatHistoryClassifier}(s''(n), t''(n))$*, the two endpoint observation rows* $\mathsf{DyadicRatCoreObservationRow}(s''(n))$ *and* $\mathsf{DyadicRatCoreObservationRow}(t''(n))$*, and the four stability faces needed by the* $\mathsf{Real}^{\uparrow}$ *finite-window seal: source endpoint classifier stability, target endpoint classifier stability, transported source observation stability, and transported target observation stability.*

Proof. First read the finite observation rows. Apply Definition 38.562. The RegSeqRat spines provide the stream-side rational histories, while $\mathsf{StreamName}^{\uparrow}$ supplies the finite unary window selected by a, w, k.

Next read the classifier faces. Apply Definition 38.563. The source $\mathsf{RatStreamNameClassifier}(s, t)$ and the first selected hsame endpoint transports give the RatUp classifier stability rows at the intermediate endpoints.

Now assemble the endpoint rows. Apply Definition 38.567 to the first selected transport step, and then apply Definition 38.569 to compose the second selected transport step. These theorems supply the final selected $\mathsf{RatHistoryClassifier}(s''(n), t''(n))$ and the two $\mathsf{DyadicRatCoreObservationRow}$ witnesses.

Thus $\mathsf{StreamName}^{\uparrow}$ contributes the finite window, $\mathsf{Rat}^{\uparrow}$ contributes the classifier faces, $\mathsf{DyadicRatCore}$ contributes the endpoint observation rows, and RegSeqRat contributes the four stream spines. Repacking these rows gives a single BHist-anchored package for the finite-window seal. □

Fit. Traditional target: composed finite-window dyadic endpoint transport for RegSeqRat. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Rat}^{\uparrow}+\mathsf{Real}^{\uparrow}+\mathsf{DyadicRatCore}$. Carrier/classifier: six $\mathsf{RatStreamNameCarrier}$ rows, the source $\mathsf{RatStreamNameClassifier}(s, t)$, and the twice-transported selected $\mathsf{RatHistoryClassifier}(s''(n), t''(n))$. Stability: composed hsame endpoint transport through the intermediate presentation. Ledger: the selected unary offset and the dyadic-radius token at each final endpoint. Core theorem: Definition 38.569. Exactness: the final endpoint package is the scoped DyadicRatCore source read at $s''(n)$ and $t''(n)$. Standard bridge: finite RealUp observation windows. Downstream use: the RealUp finite-window seal. Status: $\mathsf{ProofTarget}$.

Fit. Traditional target: transported finite-window dyadic endpoint source coverage for RegSeqRat. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{DyadicRatCore}$. Carrier/classifier: four $\mathsf{RatStreamNameCarrier}$ rows, the source $\mathsf{RatStreamNameClassifier}(s, t)$, and the transported selected $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$. Stability: selected hsame transport of the two endpoints. Ledger: unary offset together with the dyadic-radius token at each transported endpoint. Core theorem: Definition 38.568. Exactness: the transported endpoint package is the scoped DyadicRatCore source read at $s'(n)$ and $t'(n)$. Standard bridge: finite rational observation windows for RealUp. Downstream use: the RealUp finite-window seal. Status: $\mathsf{ProofTarget}$.

Fit. Traditional target: finite-window dyadic endpoint transport for RegSeqRat. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Rat}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{DyadicRatCore}$ over BHist. Carrier: four $\mathsf{RatStreamNameCarrier}$ rows

with RegSeqRat stream certificate spines. Classifier: $\mathsf{RatStreamNameClassifier}(s,t)$ and the transported selected $\mathsf{RatHistoryClassifier}$. Stability: selected pointwise hsame transport plus positive-denominator transport. Ledger: the selected unary offset and dyadic-radius token. Core theorem: Definition 38.567. Exactness: transported selected endpoint coverage. Standard bridge: finite RealUp observation windows for RegSeqRat streams. Downstream use: RealUp finite-window seal and later RegSeqRat formalization. Status: $\mathsf{ProofTarget}$.

Theorem 38.572 (Real unary StreamNameUp classifier selected endpoint shape package)**.** *For unary observation maps* $s,t,s',t' : \mathsf{Hist} \to \mathsf{Hist}$*, a certified observation* $n : \mathsf{Hist}$*, and histories* $a,b,z_s,z_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameClassifier}(s,t), \quad \mathsf{UnaryHistory}(n),$$

the pointwise transport rows

$$\forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(s(k),s'(k)), \quad \forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(t(k),t'(k)),$$

and the displayed selected endpoints

$$\mathsf{hsame}(s'(n),\mathsf{E1}(a)), \quad \mathsf{hsame}(t'(n),\mathsf{E1}(b)).$$

Then pointwise transport carries the StreamNameUp classifier to the transported maps, and the selected endpoints obtained from that classifier carry the full RatUp denominator and one-tail shape package:

$$\begin{aligned}
&\mathsf{RatStreamNameClassifier}(s',t')\wedge\\
&\mathsf{RatHistoryClassifier}(s'(n),t'(n)) \wedge \mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(t'(n))\wedge\\
&\mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(t'(n)) \wedge (\mathsf{hsame}(s'(n),\mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n),\mathsf{e}) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s'(n),\mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n),\mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a,b).
\end{aligned}$$

Proof. First convert the StreamNameUp classifier into the Real unary classifier. Apply Definition 38.551 with the two carrier fields projected from $\mathsf{RatStreamNameClassifier}(s,t)$, then use the reverse implication on the same classifier row.

Next apply Definition 38.275 to the same two carrier fields, the converted Real unary classifier, and the two pointwise transport rows. This gives $\mathsf{RatStreamNameClassifier}(s',t')$.

Now apply Definition 38.555 to that Real unary classifier, the selected certified observation n, the two pointwise transport rows, and the two displayed $\mathsf{E1}$ endpoint comparisons. The conclusion is exactly the displayed selected endpoint package. Repack it with the transported StreamNameUp classifier row. □

Fit. Traditional target: selected unary stream observation readback from a certified StreamNameUp classifier. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Real}^{\uparrow}+\mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ transported through $\mathsf{RealUnaryStreamClassifier}$. Ledger: selected unary observation and endpoint displays. Core theorem: Definition 38.572. Exactness: transported StreamNameUp classifier and displayed selected endpoint package. Standard bridge: pointwise rational stream observation readback. Status: $\mathsf{ProofTarget}$.

Theorem 38.573 (Real unary StreamNameUp classifier selected endpoint tail pairwise coherence)**.** *For unary observation maps* $s,t,s',t' : \mathsf{Hist} \to \mathsf{Hist}$*, a certified observation* $n : \mathsf{Hist}$*, and*

histories $a, a', b, b' : \mathsf{Hist}$*, assume* $\mathsf{RatStreamNameClassifier}(s, t)$*,* $\mathsf{UnaryHistory}(n)$*, the two pointwise transport rows of Definition 38.572, and the four displayed endpoints*

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(s'(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b')).$$

Then $\mathsf{UnaryHistory}(a)$*,* $\mathsf{UnaryHistory}(a')$*,* $\mathsf{UnaryHistory}(b)$*,* $\mathsf{UnaryHistory}(b')$*, and all six tail comparisons hold:*

$$\mathsf{hsame}(a, a'), \quad \mathsf{hsame}(a, b), \quad \mathsf{hsame}(a, b'), \quad \mathsf{hsame}(a', b), \quad \mathsf{hsame}(a', b'), \quad \mathsf{hsame}(b, b').$$

Proof. Apply Definition 38.572 twice, first to the displayed pair (a, b) and then to (a', b'), instantiating the zero-headed display parameters by endpoint tails. This supplies the four unary-history rows together with $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(a', b')$.

For the same-side comparisons, compose the two displays of $s'(n)$ to obtain $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ and use BEDC.FKernel.Hist.hsame_e1_inversion_iff to read back $\mathsf{hsame}(a, a')$. The same composition and e-one inversion applied to the two displays of $t'(n)$ gives $\mathsf{hsame}(b, b')$.

It remains to fill the two mixed comparisons. Compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ by BEDC.FKernel.Hist.hsame_trans to obtain $\mathsf{hsame}(a, b')$; compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(a', b)$. Package the four unary rows and the six comparisons. □

Lean4 ✓ `BEDC.Derived.RealUp.RatStreamNameClassifier selected endpoint tail pairwise coherence`

Fit. Traditional target: selected unary stream observation readback. BEDC source: RealUp unary streams plus RatUp classifier. Carrier: unary-indexed rational observation maps. Classifier: RealUnaryStreamClassifier. Stability: pointwise hsame transport. Ledger: selected certified observation and endpoint displays. Core theorem: Definition 38.555. Exactness: displayed endpoint package. Standard bridge: selected rational observation readback for RealUp names. Status: ProofTarget.

Theorem 38.574 (Real unary StreamNameUp finite-window selected denominator package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, z_s, z_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then the selected endpoints carry the finite-window denominator package:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k)))\wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k)))\wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k)))\wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False})\wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.553 to the two carrier fields projected from $\mathsf{RatStreamNameClassifier}(s, t)$, the StreamNameUp classifier, and the unary anchor and bound. This gives $\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha,$

Project this window classifier at the selected offset k by Definition 38.157. The offset witness $\mathsf{UnaryOffsetLe}(k, \omega)$ gives

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))).$$

Apply Definition 37.50 to this selected endpoint classifier. This supplies positive unary denominator witnesses for both selected endpoints.

Apply Definition 37.12 separately to the two positive-denominator rows. These projections give the two unary-history rows and the two nonempty endpoint exclusions.

It remains to rule out selected zero heads. For the left endpoint, transport the positive denominator witness along $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s))$ by Definition 37.6; the transported witness contradicts Definition 37.10.

The right endpoint is identical, using the displayed comparison with $\mathsf{E0}(z_t)$. The same transport and zero-head absurdity theorem give the second zero-head exclusion, and all rows together form the displayed package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected denominator package`

Theorem 38.575 (Real unary StreamNameUp finite-window selected shape package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, u, v, z_s, z_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega),$$

and the displayed selected window endpoints

$$\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)), \quad \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)).$$

Then the selected endpoints carry the finite-window RatUp readback package:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k)))\wedge\\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v)\wedge\\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k)))\wedge\\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k)))\wedge\\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.574 to the StreamNameUp classifier, the unary anchor and bound, the selected offset, and the two zero-head tests. Inside that package, the carrier rows are the two fields projected from $\mathsf{RatStreamNameClassifier}(s, t)$. This supplies the selected RatUp classifier, the positive-denominator rows, the unary endpoint rows, the nonempty exclusions, and the two zero-head exclusions.

The selected RatUp classifier is the first row of the selected shape package. Transport the selected RatUp classifier across the two displayed $\mathsf{E1}$ endpoint comparisons by Definition 37.54. The transported classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v))$.

Apply Definition 37.49 in the forward direction to the transported $\mathsf{E1}$ classifier. This reads back $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, and $\mathsf{hsame}(u, v)$. Together with the denominator package rows, these are exactly the displayed finite-window selected shape package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected shape package`

Fit. Traditional target: selected finite-window rational readback for RealUp StreamNameUp classifiers. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: hsame transport of selected endpoints and positive-denominator transport. Ledger: anchor, bound, offset, selected endpoint displays, and zero-head tests. Core theorem: Definition 38.575. Exactness: selected source RatUp classifier, displayed RatUp one-tail classifier, denominator, unary, nonempty, and zero-headed exclusion rows. Standard bridge: finite rational observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.576 (Real unary StreamNameUp finite-window selected common e-one tail readback)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then the selected endpoints admit common one-headed unary readback:

$$\exists u, v : \mathsf{Hist},\ \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsam}$$

Proof. Apply Definition 38.574 to the StreamNameUp classifier, the unary anchor and bound, the selected offset, and arbitrary zero-head tests. This supplies the selected RatUp classifier, the positive-denominator rows, the unary endpoint rows, and the nonempty endpoint exclusions.

Use Definition 18.45 separately on the two selected endpoints. The left endpoint yields a tail u with $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u))$ and $\mathsf{UnaryHistory}(u)$. The right endpoint yields a tail v with $\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v))$ and $\mathsf{UnaryHistory}(v)$.

Transport the selected RatUp classifier across these two displayed endpoint rows by Definition 37.54. The transported classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v))$.

Apply Definition 37.49 in the forward direction to the transported classifier. This reads back the common tail row $\mathsf{hsame}(u, v)$. Package u and v with the two displayed endpoint rows, the two unary-tail rows, and the common-tail row. □

Fit. Traditional target: selected finite-window RealUp endpoint witness readback without displayed endpoint assumptions. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through selected $\mathsf{RatHistoryClassifier}$. Stability: hsame endpoint transport from selected endpoints to e-one displays. Ledger: anchor, bound, selected offset, positive denominator rows, unary endpoint rows, and nonempty exclusions. Core theorem: Definition 38.576. Exactness: existential one-headed endpoints with unary tails and common tail comparison. Standard bridge: finite rational observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.577 (Real unary StreamNameUp finite-window selected full endpoint package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, u, v, z_s, z_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega),$$

and the displayed selected window endpoints

$$\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)), \quad \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)).$$

Then the selected endpoints carry the full finite-window endpoint package:

$$\begin{aligned}
&\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega) \wedge \\
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k))) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.553 to the two carrier rows, the StreamNameUp classifier, and the unary anchor and bound. This produces $\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega)$.

Apply Definition 38.157 to that window classifier and the displayed offset witness $\mathsf{UnaryOffsetLe}(k, \omega)$. This gives the selected source classifier

$$\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))).$$

Apply Definition 38.575 to the same hypotheses. It supplies the transported $\mathsf{E1}$ classifier, the unary tail readback, the two positive-denominator rows, the two unary endpoint rows, the nonempty exclusions, and the zero-head exclusions.

Combining the window classifier, the projected selected classifier, and the selected shape package gives exactly the displayed full endpoint package. □

Fit. Traditional target: RealUp finite-window selected full endpoint readback. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Real}^{\uparrow}+\mathsf{Rat}^{\uparrow}$. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: hsame endpoint transport with positive-denominator transport. Ledger: carrier, classifier, unary anchor, unary bound, selected offset, endpoint displays, and zero-head tests. Core theorem: Definition 38.577. Exactness: selected classifier plus the displayed endpoint package. Standard bridge: downstream finite rational observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.578 (Real unary StreamNameUp finite-window selected classifier appended-tail package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, u, v, z_s, z_t, d_s, d_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega),$$

and the displayed selected window endpoints

$$\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)), \quad \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)).$$

Then the selected endpoints carry the selected-shape endpoint package and the appended e-zero-tail exclusions:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k)))\wedge\\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v))\wedge\\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v)\wedge\\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k)))\wedge\\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k)))\wedge\\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.575 to the two carrier rows, the StreamNameUp classifier, the unary anchor and bound, the selected offset, and the displayed one-headed endpoint comparisons. This gives every selected endpoint row through the zero-head exclusions, including the two selected positive-denominator witnesses.

For the left endpoint, suppose $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s))$. Transport the positive denominator witness for $s(\mathsf{append}(\alpha, k))$ across this displayed sameness by Definition 37.6. The

transported witness is a positive unary denominator witness for $d_s \cdot \mathsf{E0}(z_s)$, contradicting Definition 37.15.

The right endpoint is identical. A display $\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t))$ transports the positive denominator witness for $t(\mathsf{append}(\alpha, k))$ to $d_t \cdot \mathsf{E0}(z_t)$, and the same appended e-zero-tail absurdity theorem contradicts that transported witness.

Repack the original selected-shape rows with the two appended-tail exclusions. This is exactly the displayed selected classifier appended-tail package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected classifier appended tail package`

Fit. Traditional target: selected finite-window rational readback with appended-tail endpoint exclusion for RealUp StreamNameUp classifiers. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: hsame transport of selected endpoints and positive-denominator transport to appended tails. Ledger: carrier, classifier, unary anchor, unary bound, selected offset, one-headed endpoint displays, zero-head tests, and appended-tail displays. Core theorem: Definition 38.578. Exactness: the full selected endpoint package plus the two appended e-zero-tail exclusion rows. Standard bridge: finite rational observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.579 (Real unary StreamNameUp finite-window selected e-one tail coverage package)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and histories $\alpha, \omega, k, z_s, z_t, d_s, d_t : \mathsf{Hist}$, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then there exist histories $u, v : \mathsf{Hist}$ such that the selected endpoints are displayed as e-one endpoints and, for those witnesses, the full endpoint rows and appended e-zero-tail exclusions hold:

$$\begin{aligned}
&\exists u, v,\ \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega) \wedge \\
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k))) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.574 to the StreamNameUp classifier, the unary anchor and bound, the selected offset, and the two zero-head tests. The package supplies the selected RatUp classifier, the two positive unary denominator rows, the two unary selected endpoint rows, the two nonempty endpoint exclusions, and the two zero-head exclusions.

Apply Definition 18.45 to the left unary endpoint row and its nonempty exclusion. This chooses a tail u, and the constructor equality supplies the hsame display from $s(\mathsf{append}(\alpha, k))$ to $\mathsf{E1}(u)$. The same theorem applied to the right unary endpoint row and nonempty exclusion chooses a tail v and supplies the hsame display from $t(\mathsf{append}(\alpha, k))$ to $\mathsf{E1}(v)$.

Feed the two e-one endpoint displays, together with the carrier rows, classifier row, unary anchor and bound, selected offset, zero-head tests, and appended-tail tests, into Definition 38.578. That theorem returns the full selected endpoint package and the two appended e-zero-tail exclusions for the chosen witnesses u and v.

Package u and v as the existential witnesses. Prepending their two selected endpoint displays to the returned appended-tail package gives exactly the displayed coverage package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected e1 tail coverage package`

Fit. Traditional target: finite RealUp observation-window readback without endpoint display hypotheses. BEDC source: $\mathsf{StreamName}^{\uparrow}$ plus $\mathsf{Real}^{\uparrow}$ finite windows plus $\mathsf{Rat}^{\uparrow}$ denominator readback. Carrier: RatStreamNameCarrier. Classifier: RatStreamNameClassifier through RealUnaryStreamWindowClassifi Stability: hsame transport plus unary nonempty e-one-tail decomposition. Ledger: anchor, bound, offset, zero-head tests, and appended-tail tests. Core theorem: Definition 38.579. Exactness: existential e-one tails plus the full selected endpoint package. Standard bridge: finite rational observation windows. Status: ProofTarget.

Theorem 38.580 (Real unary StreamNameUp finite-window transported selected appended-tail package)**.** *For observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, u, v, z_s, z_t, d_s, d_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s,t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k,\omega),$$

and the pointwise transported rows

$$\forall n : \mathsf{Hist},\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)), \quad \forall n : \mathsf{Hist},\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Assume also the displayed transported selected window endpoints

$$\mathsf{hsame}(s'(\mathsf{append}(\alpha,k)), \mathsf{E1}(u)), \quad \mathsf{hsame}(t'(\mathsf{append}(\alpha,k)), \mathsf{E1}(v)).$$

Then the transported selected endpoints carry the selected-shape endpoint package and the appended e-zero-tail exclusions:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(s'(\mathsf{append}(\alpha,k)), t'(\mathsf{append}(\alpha,k)))\wedge\\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v))\wedge\\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u,v)\wedge\\
&\mathsf{PositiveUnaryDenominator}(s'(\mathsf{append}(\alpha,k))) \wedge \mathsf{PositiveUnaryDenominator}(t'(\mathsf{append}(\alpha,k)))\wedge\\
&\mathsf{UnaryHistory}(s'(\mathsf{append}(\alpha,k))) \wedge \mathsf{UnaryHistory}(t'(\mathsf{append}(\alpha,k)))\wedge\\
&(\mathsf{hsame}(s'(\mathsf{append}(\alpha,k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(\mathsf{append}(\alpha,k)), \mathsf{e}) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s'(\mathsf{append}(\alpha,k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(\mathsf{append}(\alpha,k)), \mathsf{E0}(z_t)) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(s'(\mathsf{append}(\alpha,k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False})\wedge\\
&(\mathsf{hsame}(t'(\mathsf{append}(\alpha,k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}).
\end{aligned}$$

Proof. First derive the selected RatUp classifier at $\mathsf{append}(\alpha,k)$. Apply Definition 38.574 to the StreamNameUp classifier, the unary anchor and bound, and the selected offset. This supplies the untransported selected RatHistoryClassifier row, positive denominator rows, unary endpoint rows, nonempty exclusions, and zero-head exclusions.

The selected index $\mathsf{append}(\alpha,k)$ is unary by the finite-window offset construction, so the two pointwise rows give $\mathsf{hsame}(s(\mathsf{append}(\alpha,k)), s'(\mathsf{append}(\alpha,k)))$ and $\mathsf{hsame}(t(\mathsf{append}(\alpha,k)), t'(\mathsf{append}(\alpha,k)))$.

Transport the selected RatUp classifier across these two comparisons by Definition 37.54. Transport the two positive denominator rows by Definition 37.6.

Apply Definition 37.49 to the transported classifier and the two displayed one-headed endpoint comparisons. This gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v))$, the two unary tail rows, and $\mathsf{hsame}(u, v)$. Apply Definition 37.12 to the transported positive-denominator rows; this gives the unary transported endpoint rows and the two empty endpoint exclusions.

For the zero-head exclusions, suppose first $\mathsf{hsame}(s'(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s))$. Transport the left positive-denominator witness across that comparison by Definition 37.6, contradicting Definition 37.10. The right zero-head exclusion is the same argument using $t'(\mathsf{append}(\alpha, k))$ and $\mathsf{E0}(z_t)$.

It remains to rule out the appended e-zero tails. For the left endpoint, suppose $\mathsf{hsame}(s'(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s))$. Transport the positive denominator witness for $s'(\mathsf{append}(\alpha, k))$ across this comparison by Definition 37.6. The transported witness contradicts Definition 37.15.

The right endpoint is identical. A display $\mathsf{hsame}(t'(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t))$ transports the positive denominator witness for $t'(\mathsf{append}(\alpha, k))$ to $d_t \cdot \mathsf{E0}(z_t)$, and the same appended e-zero-tail obstruction contradicts that transported witness.

Repack the transported classifier, displayed e-one readback rows, denominator rows, endpoint exclusions, and appended-tail contradictions. These rows are exactly the transported selected endpoint package. □

Fit. Traditional target: transported finite-window rational readback with appended-tail endpoint obstruction for RealUp StreamNameUp classifiers. BEDC source: $\mathsf{StreamName}^{\uparrow}+\mathsf{Real}^{\uparrow}+\mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ transported to the selected $\mathsf{RatHistoryClassifier}$. Stability: pointwise hsame transport of StreamNameUp endpoints and positive-denominator transport to appended tails. Ledger: carrier, classifier, unary anchor, unary bound, selected offset, pointwise transport rows, one-headed endpoint displays, zero-head tests, and appended-tail displays. Core theorem: Definition 38.580. Exactness: transported selected endpoint package plus the two appended e-zero-tail exclusion rows. Standard bridge: finite rational observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.581 (Real unary StreamNameUp finite-window selected e-one tail pairwise coherence)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and histories $\alpha, \omega, k, a, a', b, b' : \mathsf{Hist}$, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega),$$

and the four displayed selected e-one endpoint comparisons

$$\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(a)), \quad \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(a')),$$

$$\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(b)), \quad \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(b')).$$

Then the displayed tails form one pairwise selected-tail class:

1. $\mathsf{UnaryHistory}(a)$.
2. $\mathsf{UnaryHistory}(a')$.
3. $\mathsf{UnaryHistory}(b)$.
4. $\mathsf{UnaryHistory}(b')$.

5. $\mathsf{hsame}(a, a')$.
6. $\mathsf{hsame}(b, b')$.
7. $\mathsf{hsame}(a, b)$.
8. $\mathsf{hsame}(a', b')$.
9. $\mathsf{hsame}(a, b')$.
10. $\mathsf{hsame}(a', b)$.

Proof. Apply Definition 38.575 to the first left and right endpoint displays, together with the StreamNameUp classifier, the unary anchor and bound, and the selected offset. This gives $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply the same selected shape package to the primed left and right endpoint displays. This gives $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the left selected endpoint, compose the symmetry of $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(a))$ with $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(a'))$. The kernel transitivity theorem BEDC.FKernel.Hist.hsame_trans gives $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$, and BEDC.FKernel.Hist.hsame_e1_iff inverts the common e-one head to give $\mathsf{hsame}(a, a')$.

The right selected endpoint is identical. Compose the symmetry of the displayed comparison with $\mathsf{E1}(b)$ and the displayed comparison with $\mathsf{E1}(b')$, then invert the common e-one head by BEDC.FKernel.Hist.hsame_e1_iff to get $\mathsf{hsame}(b, b')$.

It remains to fill the two off-diagonal rows. Transitivity sends $\mathsf{hsame}(a, a')$ and $\mathsf{hsame}(a', b')$ to $\mathsf{hsame}(a, b')$, while symmetry of $\mathsf{hsame}(a, a')$ followed by $\mathsf{hsame}(a, b)$ gives $\mathsf{hsame}(a', b)$. Together with the four unary rows and the two direct cross rows, these are exactly the ten displayed claims. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected e1 tail pairwise coherence`

Theorem 38.582 (Real unary StreamNameUp finite-window selected canonical e-one tail witness)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and histories $\alpha, \omega, k : \mathsf{Hist}$, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then there exist histories $u, v : \mathsf{Hist}$ such that the selected endpoints have canonical e-one tail displays and every other selected e-one display is coherent with them:

$$\begin{aligned}
&\exists u, v : \mathsf{Hist}, \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\forall u', v' : \mathsf{Hist}, \ (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v'))) \to \\
&\qquad \mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v') \wedge \\
&\qquad \mathsf{hsame}(u', v) \wedge \mathsf{hsame}(u, v').
\end{aligned}$$

Proof. Instantiate Definition 38.579 with the harmless choices $z_s = z_t = d_s = d_t = \mathsf{e}$. The returned witnesses u and v carry the two selected endpoint displays, $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, and $\mathsf{hsame}(u, v)$.

Now fix arbitrary histories u', v' with the same selected endpoint displays. Apply Definition 38.581 with $a = u$, $a' = u'$, $b = v$, and $b' = v'$. The hypotheses are the carrier, classifier, unary

anchor, unary bound, selected offset, the chosen endpoint displays, and the arbitrary endpoint displays.

Project the ten rows returned by pairwise coherence. They give $\mathsf{UnaryHistory}(u')$, $\mathsf{UnaryHistory}(v')$, $\mathsf{hsame}(u, u')$, $\mathsf{hsame}(v, v')$, $\mathsf{hsame}(u', v)$, and $\mathsf{hsame}(u, v')$. Repack these rows under the universal clause and then package the chosen u and v as the displayed canonical selected-tail witnesses. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected canonical e1 tail witness`

Fit. Traditional target: canonical finite rational observation tail. BEDC source: $\mathsf{StreamName}^{\uparrow}$ plus $\mathsf{Real}^{\uparrow}$ plus $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{FKernel}$ e-one inversion. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: endpoint hsame and pairwise tail coherence. Ledger: anchor, bound, offset, and selected endpoint displays. Core theorem: Definition 38.582. Exactness: existence plus uniqueness up to hsame. Standard bridge: finite rational observation windows for RegSeqRat. Status: $\mathsf{ProofTarget}$.

Theorem 38.583 (Real unary StreamNameUp finite-window selected canonical e-one tail full package)**.** *For observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and histories $\alpha, \omega, k, z_s, z_t, d_s, d_t : \mathsf{Hist}$, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then there exist histories $u, v : \mathsf{Hist}$ such that the selected endpoints carry the complete displayed e-one certificate and every other selected e-one display is coherent with those witnesses:

$$\begin{aligned}
&\exists u, v : \mathsf{Hist}, \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega) \wedge \\
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k))) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&\forall u', v' : \mathsf{Hist}, (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v'))) \to \\
&\qquad \mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v') \wedge \\
&\qquad \mathsf{hsame}(u', v) \wedge \mathsf{hsame}(u, v').
\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected canonical e1 tail full package`

Proof. Apply Definition 38.579 with the displayed histories z_s, z_t, d_s, d_t. The resulting witnesses u and v carry the two selected endpoint displays, the RealUp window classifier row, the selected RatHistoryClassifier rows, the unary tail rows, the denominator rows, the endpoint unary rows, the nonempty exclusions, the zero-head exclusions, and the appended e-zero-tail exclusions.

Now fix arbitrary histories u', v' with the same selected e-one endpoint displays. Apply Definition 38.581 with $a = u$, $a' = u'$, $b = v$, and $b' = v'$. The required displayed endpoint comparisons are exactly the two chosen displays and the two arbitrary displays.

Project the rows supplied by pairwise coherence. They give $\mathsf{UnaryHistory}(u')$, $\mathsf{UnaryHistory}(v')$, $\mathsf{hsame}(u, u')$, $\mathsf{hsame}(v, v')$, $\mathsf{hsame}(u', v)$, and $\mathsf{hsame}(u, v')$. Repack this universal coherence clause after the full coverage rows and package u, v as the displayed canonical full witnesses. □

Fit. Traditional target: canonical finite-window selected rational endpoint certificate for RealUp StreamNameUp classifiers. BEDC source: $\mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{Rat}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: selected endpoint hsame transport plus pairwise e-one tail coherence. Ledger: carrier, classifier, unary anchor, unary bound, selected offset, selected endpoint displays, RatHistoryClassifier rows, positive denominator rows, endpoint unary rows, and endpoint exclusion rows. Core theorem: Definition 38.583. Exactness: existence, selected endpoint exactness, denominator exclusions, appended-tail exclusions, and uniqueness up to hsame. Standard bridge: finite RegSeqRat observation windows. Status: $\mathsf{ProofTarget}$.

Theorem 38.584 (Real unary StreamNameUp finite-window selected tail-class exactness)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then the selected finite-window tail class is exact:

$$\begin{aligned}
&\exists u, v : \mathsf{Hist}, \mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\forall u', v' : \mathsf{Hist}, \ (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v'))) \iff \\
&\qquad (\mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v')).
\end{aligned}$$

Proof. Apply Definition 38.582 to the carrier rows, the StreamNameUp classifier, the unary anchor and bound, and the selected offset. This gives witnesses u and v with the two selected endpoint displays, their unary rows, $\mathsf{hsame}(u, v)$, and the universal coherence package for every displayed selected e-one tail pair.

For the forward direction of the displayed if-and-only-if, fix u' and v' with the two selected endpoint displays. The universal coherence clause of the canonical witness theorem gives $\mathsf{UnaryHistory}(u')$, $\mathsf{UnaryHistory}(v')$, $\mathsf{hsame}(u, u')$, and $\mathsf{hsame}(v, v')$.

For the reverse direction, assume $\mathsf{UnaryHistory}(u')$, $\mathsf{UnaryHistory}(v')$, $\mathsf{hsame}(u, u')$, and $\mathsf{hsame}(v, v')$. Apply BEDC.FKernel.Hist.hsame_e1_congr to the two tail comparisons to obtain $\mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(u'))$ and $\mathsf{hsame}(\mathsf{E1}(v), \mathsf{E1}(v'))$.

Compose the selected display $\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u))$ with $\mathsf{hsame}(\mathsf{E1}(u), \mathsf{E1}(u'))$ by BEDC.FKernel.Hist.hs and do the same for the right endpoint. The two composed comparisons are exactly the selected endpoint displays for u' and v', completing the reverse direction and the exact tail-class package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealUnaryStreamWindowClassifier selected tail class exactness`

Fit. Traditional target: exact selected finite rational observation tails. BEDC source: $\mathsf{StreamName}^{\uparrow}$ plus $\mathsf{Real}^{\uparrow}$ plus $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{FKernel}$ e-one congruence and transitivity. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$. Stability: endpoint hsame transport inside the selected e-one tail class. Ledger: anchor, bound, offset, selected endpoint displays, and canonical tail witnesses. Core theorem: Definition 38.584. Exactness: displayed selected e-one tails are exactly the canonical hsame classes of u and v. Downstream consumer: RegSeqRat finite observation canonicalization. Status: $\mathsf{ProofTarget}$.

Theorem 38.585 (Real unary StreamNameUp finite-window selected full tail-class certificate)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k, z_s, z_t, d_s, d_t : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k,\omega).$$

Then there exist histories $u, v : \mathsf{Hist}$ such that the selected endpoints carry the full endpoint certificate and the selected tail class is exact:

$$\begin{aligned}
&\exists u, v : \mathsf{Hist}, \mathsf{hsame}(s(\mathsf{append}(\alpha,k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha,k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{RealUnaryStreamWindowClassifier}(s,t,\alpha,\omega) \wedge \\
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha,k)), t(\mathsf{append}(\alpha,k))) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u,v) \wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha,k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha,k))) \wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha,k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha,k))) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha,k)), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha,k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha,k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha,k)), \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(s(\mathsf{append}(\alpha,k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(t(\mathsf{append}(\alpha,k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&\forall u', v' : \mathsf{Hist}, \big(\mathsf{hsame}(s(\mathsf{append}(\alpha,k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha,k)), \mathsf{E1}(v'))\big) \iff \\
&\qquad \big(\mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u,u') \wedge \mathsf{hsame}(v,v')\big).
\end{aligned}$$

Proof. Apply Definition 38.584 to the carrier rows, the StreamNameUp classifier, the unary anchor and bound, and the selected offset. This chooses witnesses u and v with the two selected e-one endpoint displays, the two unary tail rows, $\mathsf{hsame}(u,v)$, and the exact if-and-only-if clause for every displayed selected e-one tail pair.

Apply Definition 38.553 to the same carrier rows, classifier, unary anchor, and unary bound. This supplies the finite-window row $\mathsf{RealUnaryStreamWindowClassifier}(s,t,\alpha,\omega)$.

Apply Definition 38.578 to the chosen selected endpoint displays, the carrier rows, the classifier, the unary anchor and bound, the selected offset, and the displayed histories z_s, z_t, d_s, d_t. The package returns the selected source classifier, the displayed $\mathsf{E1}$ classifier, the unary tail and common-tail rows, the positive-denominator rows, the endpoint unary rows, the nonempty exclusions, the zero-head exclusions, and the two appended e-zero-tail exclusions.

Repack the chosen endpoint displays, the finite-window row, the selected classifier appended-tail package, and the universal if-and-only-if clause from tail-class exactness. These rows are exactly the displayed full tail-class certificate. □

Fit. Traditional target: selected finite-window full tail-class certificate; BEDC source: $\mathsf{StreamName}^{\uparrow}$+ $\mathsf{Real}^{\uparrow}$+$\mathsf{Rat}^{\uparrow}$ over BHist; Carrier: $\mathsf{RatStreamNameCarrier}$; Classifier: $\mathsf{RatStreamNameClassifier}$ through $\mathsf{RealUnaryStreamWindowClassifier}$; Stability: endpoint hsame transport and positive-denominator transport; Ledger: carrier, classifier, anchor, bound, offset, endpoint displays, denominator rows, endpoint exclusions, appended-tail exclusions, and tail-class quantification; Core theorem: Definition 38.585; Exactness: full selected endpoint certificate plus exact canonical tail class; Standard bridge: RegSeqRat finite observation canonicalization; Status: $\mathsf{ProofTarget}$.

Definition 38.586 (Real unary StreamNameUp selected-window scoped tail package)**.** *The selected-window scoped tail package is the ten-field BEDC certificate for the finite RealUp observation-tail canonicalization at a selected unary offset:*

1. *Traditional target: finite RealUp observation-tail canonicalization for selected StreamNameUp windows.*
2. *BEDC source:* $\mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{Rat}^{\uparrow} + \mathsf{FKernel}$ *over* BHist.
3. *Carrier:* $\mathsf{RatStreamNameCarrier}$ *for both observation maps.*

4. *Classifier:* RatStreamNameClassifier *through* RealUnaryStreamWindowClassifier.
5. *Stability:* hsame *transport, FKernel e-one congruence and inversion, positive-denominator transport, and offset-bound projection.*
6. *Ledger: carrier rows, classifier row, unary anchor, unary bound, selected offset, endpoint displays, denominator exclusions, and canonical tail witnesses.*
7. *Bound data: the selected site is* $\mathsf{append}(\alpha, k)$ *with* $\mathsf{UnaryOffsetLe}(k, \omega)$*, so the finite-window classifier projects to the selected source* RatHistoryClassifier *row.*
8. *Core theorem list: Definition 38.574, Definition 38.579, Definition 38.582, Definition 38.584, Definition 38.585, and the window-sourced exactness target* BEDC.Derived.RealUp.RealUnaryStreamWindowClassi
9. *Exactness predicate: displayed selected e-one tail pairs are present exactly when their tails lie in the canonical* hsame *classes of the chosen witnesses.*
10. *Standard bridge: RegSeqRat finite observation canonicalization. Status:* ProofTarget.

Theorem 38.587 (Real unary StreamNameUp selected-window scoped tail package)**.** *For observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and histories* $\alpha, \omega, k : \mathsf{Hist}$*, assume*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t),$$

$$\mathsf{UnaryHistory}(\alpha), \quad \mathsf{UnaryHistory}(\omega), \quad \mathsf{UnaryOffsetLe}(k, \omega).$$

Then the selected window carries a scoped package: there exist histories $u, v : \mathsf{Hist}$ *such that*

$$\begin{aligned}
&\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u)) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v)) \wedge \\
&\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega) \wedge \\
&\mathsf{RatHistoryClassifier}(s(\mathsf{append}(\alpha, k)), t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(u), \mathsf{E1}(v)) \wedge \\
&\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(u, v) \wedge \\
&\mathsf{PositiveUnaryDenominator}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{PositiveUnaryDenominator}(t(\mathsf{append}(\alpha, k))) \wedge \\
&\mathsf{UnaryHistory}(s(\mathsf{append}(\alpha, k))) \wedge \mathsf{UnaryHistory}(t(\mathsf{append}(\alpha, k))) \wedge \\
&\forall z_s, z_t, d_s, d_t : \mathsf{Hist}, \ (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&\qquad (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{e}) \to \mathsf{False}) \wedge \\
&\qquad (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&\qquad (\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), d_s \cdot \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge \\
&\qquad (\mathsf{hsame}(t(\mathsf{append}(\alpha, k)), d_t \cdot \mathsf{E0}(z_t)) \to \mathsf{False}) \wedge \\
&\forall u', v' : \mathsf{Hist}, \ \big(\mathsf{hsame}(s(\mathsf{append}(\alpha, k)), \mathsf{E1}(u')) \wedge \mathsf{hsame}(t(\mathsf{append}(\alpha, k)), \mathsf{E1}(v'))\big) \iff \\
&\qquad \big(\mathsf{UnaryHistory}(u') \wedge \mathsf{UnaryHistory}(v') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(v, v')\big).
\end{aligned}$$

Proof. Apply Definition 38.584 to the carrier rows, the StreamNameUp classifier, the unary anchor and bound, and the selected offset. This chooses canonical tails u and v with the two selected e-one endpoint displays, the unary tail rows, $\mathsf{hsame}(u, v)$, and the exact if-and-only-if clause for every displayed selected e-one tail pair.

Apply Definition 38.553 to the same carrier rows, classifier, unary anchor, and unary bound. This gives $\mathsf{RealUnaryStreamWindowClassifier}(s, t, \alpha, \omega)$. Projecting that window at the selected offset is the selected source RatHistoryClassifier row at $\mathsf{append}(\alpha, k)$.

The displayed selected endpoints for u and v transport the selected source classifier to $\mathsf{RatHistoryClassifier}(\mathsf{E1}(u)$, by Definition 37.54. FKernel e-one inversion, as used in the canonical tail-class theorem, reads this transported row as the canonical common-tail relation.

For the denominator and endpoint-exclusion ledger, fix arbitrary histories z_s, z_t, d_s, d_t. Apply Definition 38.574 to the classifier, the unary anchor and bound, the selected offset, and the zero-head tests. This returns the selected positive-denominator rows, endpoint unary rows, nonempty exclusions, and zero-head exclusions.

Apply Definition 38.578 to the chosen e-one endpoint displays and the same displayed histories z_s, z_t, d_s, d_t. Positive-denominator transport supplies the two appended e-zero-tail exclusions, while the earlier selected source row and endpoint rows remain unchanged.

Repack the finite-window row, selected source row, endpoint classifier, positive-denominator ledger, endpoint exclusions, canonical tails, and exact tail-class predicate. These fields are precisely the scoped package described in Definition 38.586. □

Theorem 38.588 (Real stream classifier selected e-one pair readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, every selected observation index n, and histories $a, b : \mathsf{Hist}$, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{hsame}(x(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y(n), \mathsf{E1}(b)),$$

then

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Apply Definition 38.106 to the stream classifier at the selected index n. This produces the finite-prefix classifier witness needed at that index.

Apply Definition 38.148 to that prefix witness and to the two displayed e-one endpoint equalities. The readback theorem returns the two unary tail witnesses and the internal sameness relation $\mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected e1 pair readback`

Theorem 38.589 (Real stream classifier transported selected e-one rational classifier readback)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, every selected observation index n, and histories $a, b : \mathsf{Hist}$, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E}$$

then

$$\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)).$$

Proof. Use the forward direction of Definition 38.135 on $\mathsf{RealStreamClassifier}(x, y)$ and select the finite prefix at n. Transport this prefix witness along the two pointwise hsame hypotheses by Definition 38.107, obtaining a classified prefix for x' and y' at the same selected index.

Project the selected endpoint classifier with Definition 38.111. It gives $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$.

Finally apply Definition 37.54 to move this endpoint classifier along $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$, producing $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected e1 rat classifier readback`

Fit: traditional target selected rational endpoint readback for RealUp streams; BEDC source RatUp plus RealUp; carrier rational-observation streams; classifier RealStreamClassifier and RatHistoryClassifier; stability prefix exactness, hsame transport, and endpoint projection; ledger selected observation index and displayed e-one seals; core theorem Definition 38.589; exactness the displayed implication; standard bridge rational equality read from equal real observations; status ProofTarget.

Theorem 38.590 (Real stream classifier transported selected e-one pair readback)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, every selected observation index* n*, and histories* $a, b : \mathsf{Hist}$*, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E}$$

then

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Apply Definition 38.589 to obtain the endpoint classifier $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 37.49 in the forward direction. The one-tail exactness theorem returns $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected e1 pair readback`

Theorem 38.591 (Real stream classifier transported selected e-one full readback package)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, every selected observation index* n*, and histories* $a, b : \mathsf{Hist}$*, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E}$$

then

$$\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Apply Definition 38.589 under the displayed transport, classifier, and e-one seal hypotheses. This gives the first conjunct $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 38.590 under the same hypotheses. This gives $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Package the endpoint classifier together with the unary-tail readback triple. The resulting conjunction is exactly the selected transported e-one readback package displayed in the statement. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected e1 full readback package`

Fit: traditional target selected sealed rational endpoint package for RealUp streams; BEDC source RatUp plus RealUp finite-prefix exactness; carrier rational-observation streams; classifier RealStreamClassifier and RatHistoryClassifier; stability prefix hsame transport and RatUp one-tail exactness; ledger selected index and two e-one seals; core theorem this block; exactness the displayed conjunction; standard bridge rational equality read from equal transported real observations; status ProofTarget.

Theorem 38.592 (Real stream classifier transported selected e-one tail determinacy)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, every selected index* n*, and histories* $a, a', b, b' : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

If the transported selected endpoints satisfy the four e-one displays

$$\mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b')),$$

then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$;
2. $\mathsf{UnaryHistory}(a')$;
3. $\mathsf{UnaryHistory}(b)$;
4. $\mathsf{UnaryHistory}(b')$;
5. $\mathsf{hsame}(a, a')$;
6. $\mathsf{hsame}(b, b')$;
7. $\mathsf{hsame}(a, b)$;
8. $\mathsf{hsame}(a', b')$.

Proof. Apply Definition 38.590 to the displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. This supplies $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply the same transported pair readback theorem again to the displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b'))$. This supplies $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the x'-tail comparison, compose the first selected endpoint display with the symmetry of the second display. The resulting $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ is inverted by BEDC.FKernel.Hist.hsame_e1_iff, giving $\mathsf{hsame}(a, a')$.

The same composition and e-one inversion applied to the two y' endpoint displays gives $\mathsf{hsame}(b, b')$. Package these same-side comparisons with the two cross-tail rows from transported pair readback. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected e1 tail determinacy`

Fit: traditional target canonical selected transported rational observation tails; BEDC source RealUp streams, RatUp e-one readback, and FKernel e-one no-confusion; carrier rational-observation streams; classifier RealStreamClassifier; stability pointwise hsame transport plus e-one inversion; ledger selected index and four e-one seals; core theorem Definition 38.592; exactness unique tails for transported selected endpoints; standard bridge RegSeqRat selected denominator tails; status ProofTarget.

Theorem 38.593 (Real stream classifier transported selected positive denominator package)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$ and every selected observation index n, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y),$$

then

$$\mathsf{RatHistoryClassifier}(x'(n), y'(n)) \wedge \mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)).$$

Proof. Apply Definition 38.140 to the two pointwise hsame hypotheses and the source stream classifier. The endpoint field of that package gives $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$ at the selected index n.

Apply Definition 37.50 to this selected endpoint classifier. It exposes $\mathsf{PositiveUnaryDenominator}(x'(n))$ and $\mathsf{PositiveUnaryDenominator}(y'(n))$.

Package the endpoint classifier with the two positive-denominator witnesses. The result is exactly the displayed transported selected denominator readback conjunction. □

Theorem 38.594 (Real stream classifier transported selected positive unary nonempty package)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$ and a selected observation index n, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Then the transported selected endpoints carry the complete rational endpoint package

$$\mathsf{RatHistoryClassifier}(x'(n), y'(n)) \wedge \mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \\ \wedge\ \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge (\mathsf{hsame}(x'(n), \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \bot).$$

Proof. Transport the selected rational-history classifier from $\mathsf{RatHistoryClassifier}(x(n), y(n))$ to $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$ along the two pointwise hsame hypotheses.

Rational classifier positivity exposes positive unary denominator witnesses at both transported endpoints. Each positive unary denominator witness gives a unary-history row and separates that endpoint from e. Packaging these rows with the transported classifier gives the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transport selected positive unary nonempty package`

Fit: traditional target selected positive denominator readback for transported RealUp rational observations; BEDC source RealUp plus RatUp; carrier rational-observation streams; classifier RealStreamClassifier and RatHistoryClassifier; stability prefix hsame transport plus RatUp denominator exposure; ledger selected observation index and transported endpoint displays; core theorem this block; exactness the displayed conjunction; standard bridge positivity of rational denominators inside equal transported real observations; status ProofTarget.

Theorem 38.595 (Real stream classifier selected positive unary nonempty package)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and a selected index n, if*

$$\mathsf{RealStreamClassifier}(x, y),$$

then the selected endpoints carry positive unary denominator witnesses, unary history witnesses, and nonempty boundary exclusions:

$$\mathsf{PositiveUnaryDenominator}(x(n)) \wedge \mathsf{PositiveUnaryDenominator}(y(n)) \wedge \mathsf{UnaryHistory}(x(n)) \wedge \mathsf{UnaryHistory}(y(n)) \wedge \\ (\mathsf{hsame}(x(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y(n), \mathsf{e}) \to \mathsf{False}).$$

Proof. Select the point classifier $\mathsf{RatHistoryClassifier}(x(n), y(n))$ from the stream classifier. RatUp positive-denominator projection gives the two positive denominator witnesses. Apply positive unary denominator readback to each endpoint, yielding the unary history witness and the empty-boundary contradiction for that endpoint. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected positive unary nonempty package`

Theorem 38.596 (Real stream classifier selected endpoint e-one shapes)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and a selected index n, if $\mathsf{RealStreamClassifier}(x, y)$ holds, then both selected endpoints are visibly one-headed with unary tails:*

$$(\exists x_t,\ x(n) = \mathsf{E1}(x_t) \wedge \mathsf{UnaryHistory}(x_t)) \wedge (\exists y_t,\ y(n) = \mathsf{E1}(y_t) \wedge \mathsf{UnaryHistory}(y_t)).$$

Proof. Apply selected positive unary nonempty readback at the chosen index. The package supplies unary-history evidence and empty-boundary exclusion for both selected endpoints.

For the x endpoint, unary nonempty one-tail readback chooses a tail x_t with $\mathsf{UnaryHistory}(x_t)$ and gives the constructor equality $x(n) = \mathsf{E1}(x_t)$ directly. Pair this equality with the unary-tail witness.

The y endpoint is treated in the same way, producing a tail y_t, its unary history witness, and the constructor equality $y(n) = \mathsf{E1}(y_t)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected endpoint e1 shapes`

Theorem 38.597 (Real stream classifier selected endpoint common e-one tail readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and a selected index n, if $\mathsf{RealStreamClassifier}(x, y)$ holds, then*

$$\exists a, b,\ \mathsf{hsame}(x(n), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y(n), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Apply Definition 38.596 at the selected index. This gives a tail a for the x endpoint and a tail b for the y endpoint, together with the two displayed e-one rows $\mathsf{hsame}(x(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y(n), \mathsf{E1}(b))$ and the two unary-tail witnesses.

Apply Definition 38.588 to the same stream classifier and to those two displayed rows. The readback theorem returns the shared-tail row $\mathsf{hsame}(a, b)$ while preserving the unary-history evidence for both tails.

Package the two tails as the existential witnesses and combine the two displayed endpoint rows, the two unary-tail witnesses, and the common-tail row. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected endpoint common e1 tail readback`

Fit: traditional target selected rational observation readback for RealUp streams; BEDC source RatUp plus RealUp; carrier rational-observation streams; classifier RealStreamClassifier with selected RatHistoryClassifier readback; stability selected endpoint e-one coverage and e-one pair readback; ledger selected observation index and the two endpoint histories; core theorem Definition 38.597; exactness the existential package of one-headed endpoints, unary tails, and tail sameness; standard bridge rational equality read from the common unary tail; status ProofTarget.

Theorem 38.598 (Real stream classifier transported selected full shape readback package)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, every selected observation index n, and histories $a, b, z_X, z_Y : \mathsf{Hist}$, if*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E}$$

then

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge \\ &\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge \\ &(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge \\ &(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).\end{aligned}$$

Proof. Apply Definition 38.591 under the two pointwise transport rows, the source RealUp stream classifier, and the two selected e-one seals. This gives the endpoint classifier $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$ together with the unary tail readback $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Apply Definition 38.594 under the same two pointwise transport rows and the same source RealUp stream classifier. This gives the selected transported endpoint classifier, the positive-denominator witnesses for $x'(n)$ and $y'(n)$, the unary-history rows, and the two nonempty exclusions.

It remains to exclude displayed zero-extensions. Suppose $\mathsf{hsame}(x'(n), \mathsf{E0}(z_X))$ holds. Transport $\mathsf{PositiveUnaryDenominator}(x'(n))$ along this comparison by Definition 37.6, and contradict Definition 37.10.

The y' endpoint is identical: transport $\mathsf{PositiveUnaryDenominator}(y'(n))$ along $\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y))$ and apply the same zero-extension exclusion.

Package the e-one classifier row, the unary-tail readback, the positive endpoint rows, the unary and nonempty endpoint rows, and the two zero-extension contradictions. The result is exactly the displayed selected full shape readback package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected full shape readback package`

Fit: traditional target selected transported endpoint-shape readback package for RealUp; BEDC source RatUp plus StreamNameUp-shaped RealUp streams; carrier rational-observation streams; classifiers RealStreamClassifier and RatHistoryClassifier; stability transported prefix readback, positive-denominator exposure, positive denominator hsame transport, and zero-extension exclusion; ledger selected observation index, two transport rows, two e-one seals, and two zero-extension tests; core theorem Definition 38.598; exactness the displayed classifier, tail, positive-denominator, unary, nonempty, and zero-extension package; downstream use by the RealUp dyadic embedding certificate; status ProofTarget.

Theorem 38.599 (Real stream classifier transported contextual full endpoint package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the four selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the two one-headed endpoint seals

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)).$$

Then the transported selected contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
3. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
4. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
5. $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}$.

6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First take the forward implication from Definition 38.139 and apply it to the two pointwise hsame rows and the source classifier $\mathsf{RealStreamClassifier}(x, y)$. This gives the transported classifier $\mathsf{RealStreamClassifier}(x', y')$.

With that classifier in hand, apply Definition 38.549 at the selected index n to the displayed unary context rows, the two context-identification rows, the four continuation rows, and the two one-headed endpoint seals. The theorem returns exactly the six displayed endpoint rows for $o_X(n)$ and $o_Y(n)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported contextual full endpoint package`

Fit: traditional target transported contextual endpoint readback for RealUp; BEDC source RealUp stream hsame transport plus RatUp contextual endpoint readback; carrier rational-observation streams; classifiers RealStreamClassifier and selected RatHistoryClassifier; stability pointwise hsame transport, Cont context closure, and endpoint readback; ledger selected index, pointwise transport rows, contextual continuation outputs, one-headed and zero-headed tests; core theorem Definition 38.599; exactness the six-row endpoint package; standard bridge transported RegSeqRat selected contextual observation readback; status ProofTarget.

Theorem 38.600 (Real stream classifier transported contextual displayed e-one classifier package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the four selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the two one-headed endpoint seals

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)).$$

Then the transported selected contextual endpoints and their displayed e-one readbacks carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.
3. $\mathsf{UnaryHistory}(a)$ *and* $\mathsf{UnaryHistory}(b)$.
4. $\mathsf{hsame}(a, b)$.
5. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
6. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
7. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
8. $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}$.

Proof. Apply Definition 38.599 under the displayed pointwise transport rows, selected context rows, continuation rows, and one-headed endpoint seals. This returns the classifier for $o_X(n)$ and $o_Y(n)$, the positive-denominator rows, the endpoint unary rows, the two nonempty exclusions, the two zero-headed exclusions, and the tail readback $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Now transport the first classifier row across the two displayed e-one seals by Definition 37.54. The source classifier is $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$, and the endpoint comparisons are $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$; the transported row is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Finally project the three tail readback components from the returned conjunction and repack them with the transported displayed e-one classifier and the remaining endpoint rows already supplied by Definition 38.599. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported contextual displayed e1 classifier package`

Theorem 38.601 (Real stream classifier transported contextual e-one tail determinacy)**.** *For rational-observation streams $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, and histories $a, a', b, b' : \mathsf{Hist}$, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the four selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the four e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')).$$

Then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$;
2. $\mathsf{UnaryHistory}(a')$;
3. $\mathsf{UnaryHistory}(b)$;
4. $\mathsf{UnaryHistory}(b')$;
5. $\mathsf{hsame}(a, a')$;
6. $\mathsf{hsame}(b, b')$;
7. $\mathsf{hsame}(a, b)$;
8. $\mathsf{hsame}(a', b')$.

Proof. Apply Definition 38.599 to the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. Its tail row supplies $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply the same transported contextual endpoint package to the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b'))$. This gives $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the X-tail comparison, compose the first display of $o_X(n)$ with the symmetry of the second display. The resulting $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ is inverted by BEDC.FKernel.Hist.hsame_e1_inversion_iff to obtain $\mathsf{hsame}(a, a')$.

The same composition and e-one inversion applied to the two displays of $o_Y(n)$ gives $\mathsf{hsame}(b, b')$. Package these two internal tail comparisons with the two endpoint-package tail rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported contextual e1 tail determinacy`

Fit: traditional target transported contextual RealUp selected-tail determinacy; BEDC source RealUp streams, RatUp point classifiers, Cont, and FKernel history sameness; carrier rational-observation streams; classifier RealStreamClassifier with selected contextual endpoint displays; stability pointwise representative transport and context closure; ledger selected index, two transport rows, context continuations, and four e-one displays; core theorem Definition 38.601; exactness canonical e-one tails under transported contexts; standard bridge RegSeqRat and dyadic selected endpoint readback; status ProofTarget.

Theorem 38.602 (Real stream classifier transported contextual cross e-one tails)**.** *Under the assumptions of Definition 38.601, the off-diagonal selected tails satisfy*

$$\mathsf{hsame}(a, b') \wedge \mathsf{hsame}(a', b).$$

Proof. Apply Definition 38.601. It gives $\mathsf{hsame}(a, a')$, $\mathsf{hsame}(b, b')$, and $\mathsf{hsame}(a, b)$ among the selected tail rows. Compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ to obtain $\mathsf{hsame}(a, b')$. Compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(a', b)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported contextual e1 tail cross hsame`

Theorem 38.603 (Real stream prefix transported selected full shape readback package)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, indices m, n, and histories $a, b, z_X, z_Y : \mathsf{Hist}$, if*

$$\mathsf{RealStreamPrefixClassifier}(x, y, m + n),$$

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b)),$$

then the transported selected finite-prefix endpoints carry the full readback package

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge \\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge \\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

Proof. First truncate the displayed prefix classifier from length $m+n$ to the selected prefix through n by Definition 38.118. Transport this shorter prefix along the two pointwise hsame rows by Definition 38.107, obtaining a classified prefix for x' and y' at index n.

Project the selected endpoint by Definition 38.111. This gives $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$. Transport that endpoint classifier across the displayed e-one seals with Definition 37.54, giving $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 37.50 to the transported endpoint classifier before the e-one seals. The two projected rows are $\mathsf{PositiveUnaryDenominator}(x'(n))$ and $\mathsf{PositiveUnaryDenominator}(y'(n))$.

Apply Definition 37.12 separately to the two positive-denominator rows. This gives $\mathsf{UnaryHistory}(x'(n))$, $\mathsf{UnaryHistory}(y'(n))$, and the two empty-endpoint exclusions.

For the e-one tails, use Definition 37.49 in the forward direction on $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$. It supplies $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

It remains to exclude displayed zero-headed endpoints. If $\mathsf{hsame}(x'(n), \mathsf{E0}(z_X))$ holds, transport $\mathsf{PositiveUnaryDenominator}(x'(n))$ along this comparison with Definition 37.6, then contradict Definition 37.10. The same argument for $y'(n)$ and z_Y gives the right zero-headed exclusion.

Collect the transported e-one classifier, the e-one tail rows, the positive denominator witnesses, the unary and nonempty endpoint rows, and the two zero-headed exclusions. These are exactly the displayed finite-prefix readback package. □

Fit: traditional target finite-prefix transported selected endpoint-shape readback package for RealUp; BEDC source RatUp plus RealUp stream prefixes; carrier rational-observation streams; classifiers RealStreamPrefixClassifier and RatHistoryClassifier; stability prefix truncation, prefix hsame transport, endpoint projection, positive-denominator hsame transport, and zero-extension exclusion; ledger selected observation index, prefix surplus index, two transport rows, two e-one seals, and two zero-extension tests; core theorem Definition 38.603; exactness the displayed classifier, tail, positive-denominator, unary, nonempty, and zero-extension package; standard bridge finite-prefix selected endpoint readback for transported RealUp rational observations; status ProofTarget.

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier transported selected full shape readback package`

Theorem 38.604 (Real stream transported selected endpoint common e-one tail readback)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$ and index n, if x and y are pointwise transported to x' and y' and $\mathsf{RealStreamClassifier}(x, y)$ holds, then the transported selected endpoints have common e-one tail readback:*

$$\exists a, b,\ \mathsf{hsame}(x'(n), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y'(n), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. The selected positive unary nonempty package exposes one-headed endpoint witnesses for $x(n)$ and $y(n)$. Transport the two displayed endpoint seals along the pointwise hsame rows to obtain the corresponding seals for $x'(n)$ and $y'(n)$. The selected e-one pair readback theorem supplies the two unary tail rows and their common hsame comparison, giving the displayed existential package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transport selected endpoint common e1 tail readback`

Theorem 38.605 (Real stream selected appended e-one tail cancellation)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, index n, and histories $d, e, tail_D, tail_E$, if the selected stream endpoints display as appended e-one tails, then tail equality cancels the displayed tails and recovers the denominator head comparison:*

$$\begin{aligned}&\mathsf{RealStreamClassifier}(x, y) \Longrightarrow \mathsf{hsame}(x(n), \mathsf{E1}(d \cdot \mathsf{E1}(tail_D))) \Longrightarrow \\ &\quad \mathsf{hsame}(y(n), \mathsf{E1}(e \cdot \mathsf{E1}(tail_E))) \Longrightarrow \mathsf{hsame}(tail_D, tail_E) \Longrightarrow \mathsf{hsame}(d, e).\end{aligned}$$

Proof. Evaluate the stream classifier at the selected index to obtain the RatUp history classifier for $x(n)$ and $y(n)$. Transport this classifier across the two displayed e-one endpoint comparisons. RatUp e-one tail readback gives the comparison between $d \cdot \mathsf{E1}(tail_D)$ and $e \cdot \mathsf{E1}(tail_E)$. The tail comparison identifies the e-one suffixes, and BEDC append right cancellation leaves exactly $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected append e1 tail hsame cancel`

Theorem 38.606 (Real stream classifier transported contextual displayed e-one appended-tail package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, b, d_X, d_Y, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the four selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the two one-headed endpoint seals

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)).$$

Then the transported selected contextual displayed e-one package also carries the appended e-zero-tail exclusions:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.
3. $\mathsf{UnaryHistory}(a)$ *and* $\mathsf{UnaryHistory}(b)$.
4. $\mathsf{hsame}(a, b)$.
5. $\mathsf{PositiveUnaryDenominator}(o_X(n))$ *and* $\mathsf{PositiveUnaryDenominator}(o_Y(n))$.
6. $\mathsf{UnaryHistory}(o_X(n))$ *and* $\mathsf{UnaryHistory}(o_Y(n))$.
7. $\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}$.
8. $\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}$ *and* $\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}$.
9. $(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported contextual displayed e1 appended tail exclusion package`

Proof. Apply Definition 38.600 under the same pointwise transport rows, selected context rows, continuation rows, and one-headed endpoint seals. This supplies the classifier rows, the displayed e-one tail readback, the positive-denominator rows, the unary endpoint rows, and the nonempty and zero-headed exclusions.

For the left appended-tail test, suppose $\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X))$. Transport $\mathsf{PositiveUnaryDenominator}(o_X(n))$ across this displayed equality by Definition 37.6. The transported positive-denominator row for $d_X \cdot \mathsf{E0}(z_X)$ contradicts Definition 37.15.

The right appended-tail test is the same argument applied to $\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y))$ and the projected positive-denominator row for $o_Y(n)$. Transport gives a positive denominator for $d_Y \cdot \mathsf{E0}(z_Y)$, contradicting the same appended e-zero-tail obstruction.

Repackage the rows from the transported contextual displayed e-one classifier package with the two appended-tail contradictions. The result is the stated displayed classifier package together with the appended e-zero-tail exclusions. □

Fit: traditional target transported RegSeqRat contextual endpoint classifier with appended-tail exclusion; BEDC source RealUp streams, RatUp classifier, Cont, and positive denominator obstruction; carrier rational-observation streams; classifier RealStreamClassifier plus selected RatHistoryClassifier; stability pointwise transport, context closure, e-one display transport, and positive-denominator appended-tail contradiction; ledger selected index, transport rows, context continuations, e-one seals, zero-headed and appended-tail tests; core theorem Definition 38.606; exactness displayed classifier package plus appended-tail exclusions; standard bridge dyadic endpoint normalization; status ProofTarget.

Theorem 38.607 (Real stream classifier transported selected common e-one certified shape envelope)**.** *For rational-observation streams* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$*, an index* n*, and histories* $z_X, z_Y : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Then there exist histories $a, b : \mathsf{Hist}$ *such that*

$$\begin{aligned}
&\mathsf{hsame}(x'(n), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y'(n), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)\\
&\wedge\ \mathsf{RatHistoryClassifier}(x'(n), y'(n))\\
&\wedge\ \mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n))\\
&\wedge\ \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n))\\
&\wedge\ (\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False})\\
&\wedge\ (\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.604 under the pointwise transport rows and the stream classifier. This supplies histories a and b, the two displayed e-one endpoint rows, the two unary tail rows, and the common tail comparison $\mathsf{hsame}(a, b)$.

Next apply Definition 38.594 under the same hypotheses. This gives the transported RatUp classifier, the two positive unary denominator rows, the unary endpoint rows, and the two nonempty endpoint exclusions.

Finally apply Definition 38.141 with the displayed histories z_X and z_Y. Repack the tail readback, endpoint classifier, denominator rows, nonempty exclusions, and zero-headed exclusions into the displayed existential envelope. □

Theorem 38.608 (RealUp StreamName-RegSeqRat common e-one window envelope)**.** *Let* $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$ *be rational-observation streams, let* $n : \mathsf{Hist}$ *be a certified unary index, and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite bundle with* $\mathsf{InBundle}(n, \mathcal{P})$*. Suppose the pointwise transport rows and stream classifier hypotheses of Definition 38.607 hold at the selected observation, and suppose the finite-window* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$ *handoff rows over* $\mathcal{P}$ *are present. Then the extracted common e-one witnesses* $a, b : \mathsf{Hist}$ *form exactly the finite consumer packet*

1. $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$*;*
2. *the transported* $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$ *row;*
3. *positive unary denominator rows for* $x'(n)$ *and* $y'(n)$*;*
4. *the endpoint exclusions* $\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}$*,* $\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}$*,* $\mathsf{hsame}(x'(n), \mathsf{E0}(a)) \to \mathsf{False}$*, and* $\mathsf{hsame}(y'(n), \mathsf{E0}(b)) \to \mathsf{False}$*;*
5. *the endpoint readbacks* $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ *and* $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$*, the unary tail rows* $\mathsf{UnaryHistory}(a)$ *and* $\mathsf{UnaryHistory}(b)$*, and the shared tail comparison* $\mathsf{hsame}(a, b)$*.*

The packet exports no quotient, completion, Classical representative, metric-completeness row, arithmetic or order row, standard bridge row, or hidden selector.

Proof. First apply Definition 38.607. It supplies the witnesses a and b, their e-one endpoint readbacks, their unary tail rows, the shared $\mathsf{hsame}(a, b)$ comparison, the transported rational classifier, positive unary denominator rows, unary endpoint rows, and the empty and zero-headed endpoint exclusions.

Next read the same selected observation through Definition 126.170. The handoff contributes only the bundled unary-history row, the displayed $\mathsf{InBundle}(n, \mathcal{P})$ membership, and the pointwise rational classifier row consumed by RegSeqRat.

Finally apply Definition 38.342 to the selected $\mathsf{Real}^{\uparrow}$ finite window. Its read surface identifies the listed finite packet as a boundary over BHist, ProbeBundle, UnaryHistory, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and hsame. The exclusion clauses supplied by the envelope rule out empty and zero-headed endpoints, so the repacked packet has no slot for quotient, completion, Classical representative, metric completeness, arithmetic, order, standard bridge, or hidden selection data. □

Theorem 38.609 (Real stream classifier transported selected appended e-one tail cancellation)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, index n, and histories $d, e, tail_D, tail_E$, if the source stream endpoints are classified and the transported selected endpoints display as appended e-one tails, then tail equality cancels the transported displayed tails and recovers the denominator head comparison:*

$$\begin{aligned}&(\forall i,\ \mathsf{hsame}(x(i), x'(i))) \Longrightarrow (\forall i,\ \mathsf{hsame}(y(i), y'(i))) \Longrightarrow \\ &\quad \mathsf{RealStreamClassifier}(x, y) \Longrightarrow \mathsf{hsame}(x'(n), \mathsf{E1}(d \cdot \mathsf{E1}(tail_D))) \Longrightarrow \\ &\quad \mathsf{hsame}(y'(n), \mathsf{E1}(e \cdot \mathsf{E1}(tail_E))) \Longrightarrow \mathsf{hsame}(tail_D, tail_E) \Longrightarrow \mathsf{hsame}(d, e).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported selected append e1 tail hsame cancel`

Proof. Evaluate $\mathsf{RealStreamClassifier}(x, y)$ at the selected index n to obtain the RatUp point classifier for $x(n)$ and $y(n)$. Use the two pointwise hsame rows to transport that classifier to the endpoints $x'(n)$ and $y'(n)$.

Transport the RatUp point classifier again across the two displayed appended e-one seals. This gives a RatUp classifier between $\mathsf{E1}(d \cdot \mathsf{E1}(tail_D))$ and $\mathsf{E1}(e \cdot \mathsf{E1}(tail_E))$.

Apply RatUp e-one tail readback to compare $d \cdot \mathsf{E1}(tail_D)$ with $e \cdot \mathsf{E1}(tail_E)$. The supplied tail comparison identifies the two e-one suffixes, and Definition 4.50 cancels the common suffix to leave exactly $\mathsf{hsame}(d, e)$. □

Fit: traditional target transported selected denominator-tail agreement readback for RealUp streams; BEDC source RealUp streams, RatUp classifier transport, RatUp e-one tail readback, and BEDC append cancellation; carrier rational-observation streams; classifier RealStreamClassifier plus selected RatHistoryClassifier; stability pointwise stream transport, endpoint hsame transport, e-one constructor transport, and append right-context cancellation; ledger selected index, two stream transport rows, two displayed appended e-one endpoint seals, and the supplied tail comparison; core theorem Definition 38.609; exactness the displayed implication to $\mathsf{hsame}(d, e)$; standard bridge RegSeqRat and dyadic endpoint normalization readback; status ProofTarget.

Theorem 38.610 (Real stream classifier transported selected certified e-one tail cancellation envelope)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, an index n, and histories $d, e, tail_D, tail_E, z_X, z_Y : \mathsf{Hist}$, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y),$$

and assume the transported selected endpoints display as appended e-one tails:

$$\mathsf{hsame}(x'(n), \mathsf{E1}(d \cdot \mathsf{E1}(tail_D))), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(e \cdot \mathsf{E1}(tail_E))), \quad \mathsf{hsame}(tail_D, tail_E).$$

Then the same transported selected observation carries the endpoint certificate and the denominator-head cancellation package

$$\begin{aligned}
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \\
&\wedge\ \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \\
&\wedge\ (\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \\
&\wedge\ (\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}) \\
&\wedge\ \mathsf{hsame}(d, e).
\end{aligned}$$

Proof. Evaluate $\mathsf{RealStreamClassifier}(x, y)$ at the selected index n. Transport the resulting RatUp point classifier along the two pointwise stream rows to obtain a classifier for $x'(n)$ and $y'(n)$.

Project positive unary denominators from the transported RatUp classifier. Apply positive-denominator readback to the two projected rows; this gives the unary endpoint rows and separates both selected endpoints from e.

For the displayed zero-headed exclusions, transport each positive-denominator row across a proposed comparison with $\mathsf{E0}(z_X)$ or $\mathsf{E0}(z_Y)$ and apply the RatUp zero-headed contradiction.

Finally transport the RatUp classifier across the two displayed appended e-one endpoint seals. RatUp e-one tail readback compares $d \cdot \mathsf{E1}(tail_D)$ with $e \cdot \mathsf{E1}(tail_E)$; the supplied tail comparison identifies the two e-one suffixes, and Definition 4.50 cancels that right context to give $\mathsf{hsame}(d, e)$. Repack the endpoint certificate rows with this denominator head comparison. □

Fit: traditional target transported selected denominator-tail agreement with certified endpoint readback for RealUp streams; BEDC source RealUp streams, RatUp classifier transport, RatUp positive denominator readback, RatUp e-one tail readback, zero-headed denominator exclusion, and BEDC append cancellation; carrier rational-observation streams; classifier RealStreamClassifier plus selected RatHistoryClassifier; stability pointwise stream transport, endpoint hsame transport, e-one constructor transport, positive denominator hsame transport, and append right-context cancellation; ledger selected index, two stream transport rows, two displayed appended e-one endpoint seals, the supplied tail comparison, and two zero-extension tests; core theorem Definition 38.610; exactness the displayed endpoint certificate and $\mathsf{hsame}(d, e)$ package; standard bridge RegSeqRat and dyadic endpoint normalization readback; status ProofTarget.

Theorem 38.611 (Real stream classifier transported selected tail-class exactness)**.** *For rational-observation streams $x, x', y, y' : \mathbb{N} \to \mathsf{Hist}$, an index n, and chosen tails $a, b : \mathsf{Hist}$, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y),$$

and the transported selected e-one displays

$$\mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b)).$$

Then the chosen transported selected e-one tail class satisfies

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b),$$

and for all histories $a', b' : \mathsf{Hist}$,

$$\begin{aligned}
&\big(\mathsf{hsame}(x'(n), \mathsf{E1}(a')) \wedge \mathsf{hsame}(y'(n), \mathsf{E1}(b'))\big) \iff \\
&\qquad \big(\mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b') \wedge \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b')\big).
\end{aligned}$$

Proof. Apply Definition 38.590 to the two pointwise representative-transport rows, the source stream classifier, and the chosen displays. This gives the selected readback triple $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

For the forward direction of the displayed if-and-only-if, fix histories a', b' with displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b'))$. Apply Definition 38.592 to the chosen displays and the two new displays. The determinacy theorem gives $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, $\mathsf{hsame}(a, a')$, and $\mathsf{hsame}(b, b')$ among its returned rows.

For the reverse direction, assume $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, $\mathsf{hsame}(a, a')$, and $\mathsf{hsame}(b, b')$. Apply BEDC.FKernel.Hist.hsame_e1_congr to $\mathsf{hsame}(a, a')$ and compose the result with $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ by BEDC.FKernel.Hist.hsame_trans. This transports the chosen left display to $\mathsf{hsame}(x'(n), \mathsf{E1}(a'))$.

Apply the same e-one congruence and transitivity chain to $\mathsf{hsame}(b, b')$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. This transports the chosen right display to $\mathsf{hsame}(y'(n), \mathsf{E1}(b'))$, so the two transported displays give the reverse membership implication. □

Fit: traditional target transported selected rational endpoint tail classes; BEDC source RealUp stream classifier, StreamName-style pointwise representative transport, RatUp selected endpoint readback, and FKernel e-one congruence; carrier rational-observation streams; classifier RealStreamClassifier; stability selected e-one determinacy, e-one congruence, and hsame transitivity; ledger selected index, two pointwise transport rows, source stream classifier, chosen transported e-one displays, and arbitrary candidate transported e-one displays; core theorem Definition 38.611; exactness the displayed selected tail class is precisely the pair of hsame classes of a and b; standard bridge and downstream consumer RegSeqRat selected observation canonicalization; status ProofTarget.

Theorem 38.612 (RealUp RegSeqRat terminal seal functoriality)**.** *Let $x, y, x', y', x'', y'' : \mathbb{N} \to \mathsf{Hist}$ be rational-observation streams, let n be a selected finite-window index, and let $a, b : \mathsf{Hist}$ be selected tails. Suppose that x, y satisfy $\mathsf{RealStreamClassifier}(x, y)$, that the first representative transport carries x, y to x', y', and that the second representative transport carries x', y' to x'', y'':*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \forall i,\ \mathsf{hsame}(x'(i), x''(i)), \quad \forall i,\ \mathsf{hsame}(y'(i), y''(i)).$$

Assume also that both stages preserve the selected e-one displays used by the RegSeqRat finite-window consumer:

$$\mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(x''(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y''(n), \mathsf{E1}(b)).$$

Then the terminal $\mathsf{Real}^{\uparrow}$ seal reads the same selected tail class through the staged route $x, y \to x', y' \to x'', y''$ as through the composed route $x, y \to x'', y''$. In particular, for every candidate pair $a', b' : \mathsf{Hist}$, the RegSeqRat terminal consumer accepts the terminal displays $\mathsf{hsame}(x''(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(y''(n), \mathsf{E1}(b'))$ exactly when a' and b' lie in the selected hsame classes of a and b.

Proof. Apply Definition 38.611 to the first-stage transport and the selected displays at $x'(n)$ and $y'(n)$. This identifies the selected tail class seen by the staged route after the first transport.

For the composed route, use BEDC.FKernel.Hist.hsame_trans to compose the two pointwise representative transports, obtaining $\mathsf{hsame}(x(i), x''(i))$ and $\mathsf{hsame}(y(i), y''(i))$ for each index i. Apply Definition 38.611 again to the composed transport and the selected terminal displays at $x''(n)$ and $y''(n)$.

It remains only to compare candidate terminal displays. If the staged read uses a candidate a', b' at the terminal seal, the selected-tail exactness theorem places a' and b' in the hsame classes

of a and b. Conversely, BEDC.FKernel.Hist.hsame_e1_congr lifts $\mathsf{hsame}(a, a')$ and $\mathsf{hsame}(b, b')$ to e-one displays, and BEDC.FKernel.Hist.hsame_trans composes those lifted displays with the terminal displays for a and b.

The argument uses only the displayed hsame transports, e-one congruence, and selected-tail exactness. It introduces no quotient sequence, completion object, host index equality, or host real equality. □

Theorem 38.613 (Real selected transport readback functoriality)**.** *Let W_0 be a finite* $\mathsf{Real}^{\uparrow}$ *observation window with finite probe bundle $\mathcal{P}$:* ProbeBundle *and selected bundled observation n :* Hist *with* $\mathsf{InBundle}(n, \mathcal{P})$*. Suppose two displayed unary-context transports*

$$\mathcal{T}_{01} : W_0 \to W_1, \qquad \mathcal{T}_{12} : W_1 \to W_2$$

are composable over the same selected $\mathsf{StreamName}^{\uparrow}$ *source, and suppose their composite transport $\mathcal{T}_{02}$ is the displayed unary-context composition of $\mathcal{T}_{01}$ and $\mathcal{T}_{12}$. Assume each of $\mathcal{T}_{01}$, $\mathcal{T}_{12}$, and $\mathcal{T}_{02}$ preserves the visible* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows at the selected finite window.*

Then the selected readback after $\mathcal{T}_{02}$ is the same finite RegSeqRat *tail-classifier row as the stagewise readback obtained by first reading after $\mathcal{T}_{01}$ and then reading after $\mathcal{T}_{12}$. Equivalently, the following two displayed packets expose one selected tail-classifier row:*

1. *the endpoint package, contextual package, appended e-one tail package, common e-one certified-shape envelope, cancellation envelope, and selected tail-class exactness read after $\mathcal{T}_{02}$;*
2. *the same six readback packages read after $\mathcal{T}_{01}$ and then transported through $\mathcal{T}_{12}$.*

The common row is determined only by the displayed $\mathsf{StreamName}^{\uparrow}$/RegSeqRat*/DyadicRatCore finite-window surface, the* UnaryHistory *and* ProbeBundle *membership rows, the* hsame/Cont/Pkg *ledger, and the finite* RealLedgerPolicy *row. It exports no quotient, completion, Classical representative, metric-completeness, arithmetic or order, standard-bridge, or hidden selector row.*

Proof. First read the selected endpoints through the composite transport. Apply the selected endpoint package Definition 38.604 and the transported contextual package Definition 38.599 to $\mathcal{T}_{02}$. These give the transported selected endpoint rows, the contextual continuation rows, and the common e-one tail comparison at W_2.

Next enrich that composite readback with the displayed endpoint certificates. Use Definition 38.606 and Definition 38.607 to attach the appended-tail exclusions, positive-denominator rows, unary endpoint rows, and the common certified-shape envelope. Apply Definition 38.610 to identify the displayed appended e-one denominator heads.

Now perform the same read in two stages. The first transport $\mathcal{T}_{01}$ supplies the same endpoint and contextual rows at W_1. The second transport $\mathcal{T}_{12}$ preserves exactly the same $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, DyadicRatCore, UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows, so the stagewise packet has the same six certificate faces as the composite packet.

Finally apply Definition 38.611 to both packets. Tail-class exactness says that a selected readback is classified only by the displayed e-one tail class and its finite RegSeqRat classifier row. Since the endpoint, contextual, appended-tail, common-shape, cancellation, and selected-tail exactness faces agree under the preserved ledger, the composite readback and the stagewise readback expose the same finite tail-classifier row.

The exclusion clause follows from the same readback surface. None of the cited packages contains a quotient carrier, completion object, Classical representative, metric-completeness certificate, arithmetic or order operation, standard-bridge datum, or hidden selector row, and the composite transport is formed inside the displayed unary-context ledger. □

Theorem 38.614 (Real selected transport readback identity)**.** *Let W be a finite* $\mathsf{Real}^{\uparrow}$ *observation window with finite probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$ *and selected bundled observation* $n : \mathsf{Hist}$ *with* $\mathsf{InBundle}(n, \mathcal{P})$*. Suppose the displayed unary-context transport*

$$\mathcal{I}_W : W \to W$$

is identity on the visible $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, DyadicRatCore,* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows at this selected finite window.*

Then the selected readback through $\mathcal{I}_W$ exposes exactly the original finite RegSeqRat *tail-classifier row attached to n in W. The readback exports no quotient, completion, metric-completeness, standard bridge, arithmetic or order structure, hidden selector, or additional endpoint row beyond the displayed finite-window ledger.*

Proof. Specialize Definition 38.613 to the identity composite $\mathcal{I}_W = \mathcal{I}_W \circ \mathcal{I}_W$. The identity hypotheses make the composite packet and the stagewise packet use the same visible $\mathsf{StreamName}^{\uparrow}$/RegSeqRat/DyadicRatC surface and the same UnaryHistory, ProbeBundle, InBundle, hsame, Cont, Pkg, and RealLedgerPolicy rows.

The endpoint package, contextual package, common-shape envelope, cancellation envelope, and selected-tail exactness rows cited in the functoriality theorem therefore read back the endpoint package already present at W. Since selected-tail exactness identifies a selected readback only through the displayed e-one tail class and its finite RegSeqRat classifier, the identity transport cannot change the tail-classifier row.

The exclusion clause is inherited from the same finite ledger projections. Each cited package is a projection of the finite observation window and none contains a quotient carrier, completion object, metric-completeness certificate, standard bridge, arithmetic or order operation, hidden selector, or additional endpoint row. □

Theorem 38.615 (Real selected transport readback pullback uniqueness)**.** *Let K be a certified common finite* $\mathsf{Real}^{\uparrow}$ *source window with finite probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$ *and selected bundled observation* $n : \mathsf{Hist}$ *with* $\mathsf{InBundle}(n, \mathcal{P})$*. Let*

$$\mathcal{T}_1 : K \to W_1, \qquad \mathcal{T}_2 : K \to W_2$$

be two selected unary-context transports into observation windows W_1 and W_2. Suppose their restrictions over K expose the same $\mathsf{StreamName}^{\uparrow}$ *classifier rows, the same* RegSeqRat *tail-classifier rows, the same DyadicRatCore endpoint rows, and the same* ProbeBundle*,* InBundle*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *ledger.*

Then the selected readback packets in W_1 and W_2 expose one common finite RegSeqRat *tail-classifier row on K. The row is determined by the shared finite source surface alone: the two readbacks may be viewed as pullbacks of the same selected source readback, and their endpoint, contextual, appended-tail, common-shape, cancellation, and selected-tail exactness faces agree after restriction to K.*

No quotient carrier, completion object, Classical representative, metric-completeness certificate, arithmetic or order structure, standard bridge, or hidden selector is part of this uniqueness statement.

Proof. First restrict both transported packets to the certified finite source window K. The hypothesis identifies the visible $\mathsf{StreamName}^{\uparrow}$ classifier rows, the RegSeqRat tail classifiers, the DyadicRatCore endpoint rows, and the ProbeBundle/InBundle/UnaryHistory membership surface used by the selected readback.

Next apply Definition 38.614 on K. It says that the identity readback over the shared source surface exposes exactly the original finite RegSeqRat tail-classifier row attached to n.

Now apply Definition 38.613 separately to the identity transport on K followed by $\mathcal{T}_1$, and to the identity transport on K followed by $\mathcal{T}_2$. Since the two restrictions carry the same hsame/Cont/Pkg ledger and the same RealLedgerPolicy row, the two transported packets read back the same row as the identity packet when pulled to K.

Finally use Definition 38.611 on the two selected packets. Tail-class exactness leaves only the displayed e-one tail class and its finite RegSeqRat classifier row. Those rows are the same by the common-source hypotheses, so the two selected readbacks expose one finite tail-classifier row on K.

The exclusions follow because every cited result is stated inside the finite-window readback ledger. None introduces quotient representatives, completion, Classical choice, metric completeness, arithmetic or order operations, or bridge data. □

Theorem 38.616 (Real selected transport readback boundary certificate)**.** *Let W be a certified finite selected* Real$^{\uparrow}$ *observation window, and let the selected unary-context readback diagram over W be either composable or obtained from a common-source pullback. Suppose every leg of the diagram preserves exactly the visible* StreamName$^{\uparrow}$ *classifier rows, the finite* RegSeqRat *tail-classifier rows, the DyadicRatCore endpoint rows, and the displayed* UnaryHistory*,* ProbeBundle*,* InBundle*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows.*

Then the selected readback boundary of the diagram is a single finite RegSeqRat *tail-classifier row. This row is stable under the identity diagram, under unary-context composition, and under common-source pullback: all three presentations expose the same selected tail class and the same finite readback ledger over the visible window.*

The boundary certificate contains no quotient carrier, no completion object, no metric-completeness datum, no arithmetic or order structure, no standard bridge data, no Classical representative, and no hidden selector.

Proof. For a composable diagram, apply Definition 38.613. The displayed finite ledger rows are preserved along each unary-context leg, so the stagewise and composite readbacks expose the same finite RegSeqRat tail-classifier row.

For the unit diagram, apply Definition 38.614. The selected readback over W is unchanged by the identity transport, and the visible StreamName$^{\uparrow}$, DyadicRatCore, membership, continuation, package, and ledger rows are exactly the rows already present in the certified window.

For a common-source pullback, apply Definition 38.615. Agreement of the two restrictions on the displayed source rows forces the same selected tail class and hence the same finite RegSeqRat classifier row on the pullback window.

These three cases exhaust the boundary operations named in the statement. Exactness is therefore the finite ledger inventory itself: the only row left at the selected readback boundary is the visible RegSeqRat tail-classifier row, and none of the excluded carriers, representatives, completions, metric, order, arithmetic, bridge, or hidden-selection data is introduced by the cited finite-window results. □

Fit: traditional target functoriality of selected finite-window readback for RealUp transports; BEDC source Real$^{\uparrow}$ selected transport, RatUp endpoint readback, StreamNameUp finite-window rows, RegSeqRat tail classifiers, and DyadicRatCore source rows; carrier BHist histories with UnaryHistory, ProbeBundle, and InBundle; classifier selected RegSeqRat tail-classifier row; stability displayed unary-context composition through hsame, Cont, Pkg, and RealLedgerPolicy rows; ledger selected finite observation, two stagewise transports, their composite transport, endpoint packages,

contextual packages, appended-tail exclusions, common-shape envelope, cancellation envelope, and selected-tail exactness; core theorem Definition 38.613; exactness the composite and stagewise selected readbacks expose the same finite RegSeqRat tail-classifier row; standard bridge none; status ProofTarget. Fit: traditional target transported contextual endpoint readback for RealUp; BEDC source RealUp stream hsame transport plus RatUp contextual endpoint readback; carrier rational-observation streams; classifiers RealStreamClassifier and selected RatHistoryClassifier; stability pointwise hsame transport, Cont context closure, and endpoint readback; ledger selected index, pointwise transport rows, contextual continuation outputs, one-headed and zero-headed tests; core theorem Definition 38.599; exactness the six-row endpoint package; standard bridge transported RegSeqRat selected contextual observation readback; status ProofTarget.

Fit: traditional target transported contextual RealUp selected-tail determinacy; BEDC source RealUp streams, RatUp point classifiers, Cont, and FKernel history sameness; carrier rational-observation streams; classifier RealStreamClassifier with selected contextual endpoint displays; stability pointwise representative transport and context closure; ledger selected index, two transport rows, context continuations, and four e-one displays; core theorem Definition 38.601; exactness canonical e-one tails under transported contexts; standard bridge RegSeqRat and dyadic selected endpoint readback; status ProofTarget.

Fit: traditional target finite-prefix transported selected endpoint-shape readback package for RealUp; BEDC source RatUp plus RealUp stream prefixes; carrier rational-observation streams; classifiers RealStreamPrefixClassifier and RatHistoryClassifier; stability prefix truncation, prefix hsame transport, endpoint projection, positive-denominator hsame transport, and zero-extension exclusion; ledger selected observation index, prefix surplus index, two transport rows, two e-one seals, and two zero-extension tests; core theorem Definition 38.603; exactness the displayed classifier, tail, positive-denominator, unary, nonempty, and zero-extension package; standard bridge finite-prefix selected endpoint readback for transported RealUp rational observations; status ProofTarget.

Theorem 38.617 (Real selected transport fit certificate)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y$ $\mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the two source classifier rows*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{RealStreamPrefixClassifier}(x, y, m{+}n).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected-transport fit carries the displayed package

$$\Big(\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)) \wedge$$
$$\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)) \wedge (\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a$$
$$\wedge \Big(\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge$$
$$\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b')\Big)$$
$$\wedge \Big(\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge$$
$$\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge$$
$$(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False})\Big).$$

Proof. We build the three displayed conjuncts separately and then conjoin them.

For the first conjunct, apply Definition 38.599 to the two pointwise transport rows, the stream classifier row, the displayed context rows, the four continuation rows, and the endpoint displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. Its six enumerated rows project to the classifier row, the two positive-denominator rows, the two endpoint-unary rows, the two empty exclusions, the two zero-headed exclusions using z_X and z_Y, and the tail rows $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

For the second conjunct, apply Definition 38.601 to the same transport, classifier, context, and continuation rows, now using all four displayed e-one endpoints. The eight displayed rows of that theorem project directly to the four unary tail rows and the four hsame tail rows.

For the third conjunct, apply Definition 38.603 to the prefix classifier row, the two pointwise transport rows, and the selected endpoint displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. Its displayed readback package projects to the e-one classifier, tail, positive-denominator, endpoint-unary, empty-exclusion, and zero-headed exclusion rows for $x'(n)$ and $y'(n)$.

The three row packages just obtained are exactly the three displayed conjuncts, and their conjunction is the stated selected-transport fit certificate. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransportFit certificate`

Theorem 38.618 (Real selected transport fit appended-tail certificate)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, u$ Hist*, assume the pointwise transport rows and the two source classifier rows*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y), \quad \mathsf{RealStreamPrefixClassifier}(x, y, m{+}n).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)$$

Then the three displayed conjuncts of Definition 38.617 hold, and the same selected-transport fit carries the appended e-zero-tail exclusions

$$\Big(\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)) \wedge$$
$$\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)) \wedge (\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a$$
$$\wedge \Big(\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge$$
$$\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b')\Big)$$
$$\wedge \Big(\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge$$
$$\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge$$
$$(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False})\Big)$$
$$\wedge (\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y)) \to \mathsf{False}).$$

Proof. First invoke Definition 38.617 with the displayed transport, classifier, context, continuation, and e-one rows. This gives the contextual endpoint package, contextual e-one determinacy package, and finite-prefix selected endpoint-shape package used below.

Next invoke Definition 38.606 on the same pointwise transport rows, source stream classifier, selected context rows, continuation rows, and contextual displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. Its appended-tail clause gives the two contextual output exclusions for $o_X(n)$ and $o_Y(n)$ with prefixes d_X and d_Y.

It remains to exclude appended e-zero tails for the selected endpoints. Project $\mathsf{PositiveUnaryDenominator}(x'(n))$ and $\mathsf{PositiveUnaryDenominator}(y'(n))$ from the finite-prefix conjunct already supplied by Definition 38.617.

For $x'(n)$, suppose $\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X))$. Transport the positive-denominator row across this history sameness by Definition 37.6; the resulting positive denominator for $e_X \cdot \mathsf{E0}(w_X)$ contradicts Definition 37.15.

For $y'(n)$, the same argument applied to $\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y))$ transports the positive-denominator row to $e_Y \cdot \mathsf{E0}(w_Y)$ and contradicts the appended e-zero-tail obstruction. Combining the three fit conjuncts with these four exclusions gives the displayed certificate. □

Fit: traditional target selected RealUp transport certificate with appended-tail endpoint exclusion; BEDC source RealUp streams, RatUp contextual readback, and finite-prefix selected endpoint shape; carrier rational-observation streams; classifiers RealStreamClassifier, RealStreamPrefixClassifier, and selected RatHistoryClassifier; stability pointwise hsame transport, contextual continuation, e-one display transport, positive-denominator hsame transport, and appended e-zero-tail obstruction; ledger selected index, transport rows, contextual output tails, selected endpoint tails, and denominator tests; core theorem Definition 38.618; exactness original fit package plus four appended-tail exclusions; standard bridge RealUp dyadic/RegSeqRat selected endpoint normalization row; status ProofTarget.

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransportFit appended tail certificate`

Theorem 38.619 (Real selected transport fit from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected-transport fit carries all four certificate packages:

1. *the contextual endpoint package of Definition 38.599 for $o_X(n)$ and $o_Y(n)$ with tails a and b;*
2. *the contextual displayed e-one classifier package of Definition 38.600 for the displayed endpoints $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$;*
3. *the contextual e-one tail determinacy package of Definition 38.601 for the four tails a, a', b, b';*
4. *the finite-prefix selected full-shape readback package of Definition 38.603 for $x'(n)$ and $y'(n)$ with tails a and b.*

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This gives the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$ needed by the finite-prefix package.

Apply Definition 38.599 to the two pointwise transport rows, the source stream classifier, the selected context rows, the four continuation rows, and the two contextual displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. This gives the contextual endpoint package.

Apply Definition 38.600 to the same transported contextual data and the same two contextual displays. This supplies the displayed e-one classifier row together with the endpoint readback rows carried by that package.

Apply Definition 38.601 to the same transported contextual data and all four contextual e-one displays. This yields the tail-determinacy package for a, a', b, b'.

Finally apply Definition 38.603 to the derived prefix classifier row, the two pointwise transport rows, and the selected endpoint displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. Conjoining the finite-prefix readback package with the three contextual packages gives the four-package selected-transport fit. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport fit from stream classifier`

Fit: traditional target transported contextual RealUp selected-transport readback without an external prefix classifier; BEDC source RealUp stream classifier prefix closure plus transported contextual endpoint readback; carrier rational-observation streams; classifier RealStreamClassifier with the derived RealStreamPrefixClassifier row; stability obligations pointwise hsame transport, contextual continuation closure, e-one readback, and finite-prefix transport; ledger selected index, prefix surplus index, two transport rows, contextual continuation outputs, and six e-one displays; core theorem Definition 38.619; exactness predicate the four-package selected endpoint and contextual readback certificate; standard bridge selected RegSeqRat and dyadic contextual endpoint readback; status ProofTarget.

Theorem 38.620 (Real selected transport synchronized e-one endpoints)**.** *For rational-observation streams $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, indices m, n, and histories $a, a', b, b', z_X, z_Y : \mathsf{Hist}$, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)),\quad \mathsf{UnaryHistory}(t_X(n)),\quad \mathsf{hsame}(p_X(n), p_Y(n)),\quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)),\quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)),\quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)),\quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)),\quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')),\quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),\quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),\quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transported endpoints satisfy the synchronized classifier and tail rows

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b'))\wedge\\ &\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b')\wedge\\ &\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b)\wedge\\ &\mathsf{hsame}(a', b') \wedge \mathsf{hsame}(a, b') \wedge \mathsf{hsame}(a', b).\end{aligned}$$

Proof. First apply Definition 38.599 to the displayed pointwise transport rows, the source stream classifier, the selected context rows, the continuation rows, and the contextual displays for a and b.

This supplies the contextual endpoint package and, in particular, the tail row $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Next apply Definition 38.601 to the same transported contextual data and all four contextual e-one displays. This supplies $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, $\mathsf{hsame}(a, a')$, $\mathsf{hsame}(b, b')$, and $\mathsf{hsame}(a', b')$, while retaining the a, b tail rows already identified above.

The off-diagonal tail rows are constructive consequences of these rows. Compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ to obtain $\mathsf{hsame}(a, b')$; compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(a', b)$.

Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This gives the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$ used by the selected finite-prefix package.

Apply Definition 38.603 to the derived prefix classifier row, the two pointwise transport rows, and the selected prefix endpoint displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. The returned finite-prefix readback package includes $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Finally use the reverse direction of Definition 37.49 on $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$. It gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b'))$, which combines with the first displayed classifier, the four unary tail rows, and the six pairwise synchronized tail rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport synchronized e1 endpoints`

Fit: traditional target synchronized selected RealUp observation readback; BEDC source RatUp, StreamNameUp-shaped RealUp streams, Cont, and FKernel hsame; carrier rational-observation streams; classifiers RealStreamClassifier plus RatHistoryClassifier on displayed e-one endpoints; stability pointwise transport, stream-classifier finite-prefix closure, contextual continuation, and e-one tail inversion/congruence; ledger selected index, prefix surplus, four endpoint histories, two contextual outputs, and prefix endpoint rows; core theorem Definition 38.620; exactness the two displayed RatHistoryClassifier rows and all six pairwise synchronized tail rows; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Theorem 38.621 (Real selected transport synchronized fit package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)$$

Then the selected transport carries the following packages simultaneously.

1. *The contextual endpoint package of Definition 38.599:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)),\\
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)),\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)),\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}),\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}),\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).
\end{aligned}$$

2. *The contextual e-one tail determinacy package of Definition 38.601 together with the off-diagonal rows of Definition 38.602:*

$$\begin{aligned}
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b') \wedge \mathsf{hsame}(a, b') \wedge \mathsf{hsame}(a', b).
\end{aligned}$$

3. *The finite-prefix selected full-shape readback package of Definition 38.603:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge\\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n\\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

4. *The synchronized displayed rows*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b') \wedge \mathsf{hsame}(a, b') \wedge \mathsf{hsame}(a', b).
\end{aligned}$$

Proof. First apply Definition 38.599 to the two pointwise transport rows, the source stream classifier, the transported context rows, the continuation rows, and the displayed endpoints for a and b. This gives the contextual endpoint package.

Next apply Definition 38.601 to the same transported contextual data and all four displayed e-one endpoints. This gives the contextual e-one tail determinacy package, including the primed tail rows used below.

The two off-diagonal rows are the same constructive consequences: compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ for $\mathsf{hsame}(a, b')$, and compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ for $\mathsf{hsame}(a', b)$.

Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This gives the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$ used by the selected finite-prefix package.

Apply Definition 38.603 to the derived prefix classifier row, the two pointwise transport rows, and the selected endpoint displays $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$. This gives the finite-prefix selected full-shape readback package and the row $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Finally use the reverse direction of Definition 37.49 with $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$. It gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b'))$, and the remaining synchronized rows are exactly the six pairwise tail rows supplied by the determinacy package and its off-diagonal consequences. □

Theorem 38.622 (Real selected transport synchronized appended-tail package from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, w_X, w_Y, z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the four synchronized packages of Definition 38.865 hold together with the appended e-zero-tail exclusions

$$(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge$$
$$(\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y)) \to \mathsf{False}).$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This produces the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$.

Invoke Definition 38.865 with the source stream classifier row and all displayed transport, context, continuation, and e-one rows. This gives the four synchronized packages.

Invoke Definition 38.618 with the derived prefix row and the same displayed hypotheses. Its appended-tail conjunct gives exactly the four displayed exclusions.

Conjoin these outputs. □

Fit: traditional target RealUp selected-transport synchronized appended-tail package from stream classifier; BEDC source RealUp stream classifier plus RatUp denominator obstruction; carrier rational-observation streams; classifier $\mathsf{RealStreamClassifier}$; stability pointwise hsame, Cont,

prefix closure, positive-denominator transport; ledger selected index and appended-tail probes; exactness synchronized package plus four exclusions; downstream RegSeqRat/dyadic endpoint normalization; status ProofTarget.

Fit: traditional target synchronized transported RealUp selected-observation certificate; BEDC source RealUp stream transport plus RatUp and FKernel Cont; carrier rational-observation streams; classifier RealStreamClassifier plus displayed RatHistoryClassifier rows; stability pointwise hsame, context closure, stream-classifier finite-prefix closure, e-one inversion; ledger selected index, prefix surplus, four tail histories, contexts; core theorem Definition 38.621; exactness package plus synchronized classifiers and six pairwise synchronized tail rows; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Theorem 38.623 (Real selected transport constant real classifier package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport carries the constant-real displayed classifier rows and the synchronized tail rows

$$\begin{aligned}&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a)), \mathsf{E1}(\mathsf{E1}(b)))\wedge\\ &\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a')), \mathsf{E1}(\mathsf{E1}(b')))\wedge\\ &\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b') \wedge \mathsf{hsame}(a, b') \wedge \mathsf{hsame}(a', b).\end{aligned}$$

Proof. Invoke Definition 38.865 with the two pointwise transport rows, the source stream classifier row, and the displayed context, continuation, and e-one hypotheses. The synchronized displayed package gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$, $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b'))$, and the six pairwise tail rows.

Apply the reverse direction of Definition 38.11 with $(d, e) = (\mathsf{E1}(a), \mathsf{E1}(b))$. This turns the first rational classifier row into $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a)), \mathsf{E1}(\mathsf{E1}(b)))$.

Apply the same reverse direction with $(d, e) = (\mathsf{E1}(a'), \mathsf{E1}(b'))$. This turns the second rational classifier row into $\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a')), \mathsf{E1}(\mathsf{E1}(b')))$. Conjoin these two constant-real rows with the six synchronized tail rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport constant real classifier package`

Fit: traditional target RealUp selected observation sealing; BEDC source RealStreamClassifier plus RatUp e-one endpoints; carrier rational-observation streams; classifier RealStreamClassifier with selected RatHistoryClassifier endpoint rows; stability pointwise hsame transport, contextual continuation closure, synchronized tail determinacy, and constant-real e-one sealing; ledger selected

index, prefix surplus index, two transport rows, context continuations, four selected context tails, and two displayed stream tails; core theorem Definition 38.623; exactness predicate the two selected constant-real classifier rows together with the six synchronized tail rows; standard bridge dyadic embedding and RegSeqRat selected endpoint sealing; status ProofTarget.

Theorem 38.624 (Real selected transport fit carrier obligation)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, w_X, w_Y, z$ Hist*, assume the pointwise transport rows, source stream classifier row, selected context rows, continuation rows, and selected e-one displays of Definition 38.619. Then the selected transport supplies the RealUp carrier boundary*

$$\begin{aligned}
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)) \wedge \mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{Positiv}\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n)) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge (\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'\\
&(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.106 to the source $\mathsf{RealStreamClassifier}(x, y)$ row at length $m + n$. This supplies the prefix classifier row required by Definition 38.617 and Definition 38.618.

Invoke Definition 38.617 with the derived prefix row and the displayed selected-transport hypotheses. Its contextual endpoint component gives the carrier rows for $o_X(n)$ and $o_Y(n)$, and its finite-prefix component gives the carrier rows for $x'(n)$ and $y'(n)$.

The tail rows for a and b are projected from both the contextual endpoint component and the finite-prefix component. The tail rows for a' and b' are projected from the contextual e-one tail determinacy component.

Finally invoke Definition 38.618. Its appended-tail component gives the four displayed e-zero-tail exclusions. These projections are exactly the displayed carrier boundary. □

Theorem 38.625 (Real selected transport fit classifier obligation)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows, source stream classifier row, selected context rows, continuation rows, and selected e-one displays of Definition 38.619. Then the selected transport supplies the RealUp classifier boundary*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a)), \mathsf{E1}(\mathsf{E1}(b))) \wedge\\
&\mathsf{RealConstantHistoryClassifier}(\mathsf{E1}(\mathsf{E1}(a')), \mathsf{E1}(\mathsf{E1}(b'))).
\end{aligned}$$

Proof. Apply Definition 38.619 to the displayed selected-transport hypotheses. The contextual endpoint package projects to $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$, and the finite-prefix package projects to $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 38.620 to the same selected-transport hypotheses. This supplies $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}($ together with the synchronized tail rows used by the constant-real package.

Apply Definition 38.623 to the same hypotheses. It supplies the two displayed RealConstantHistoryClassifier rows. Conjoining the three rational classifier rows with the two constant-real classifier rows gives the stated classifier boundary. □

Theorem 38.626 (Real selected transport fit ledger obligation)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, w_X, w_Y, z$ Hist*, assume the pointwise transport rows, source stream classifier row, selected context rows, continuation rows, and selected e-one displays of Definition 38.619. Then the ledger consumed by the carrier and classifier obligations is exactly the selected index* n*, the prefix surplus* m*, the pointwise transport rows* $\forall i,\ \mathsf{hsame}(x(i), x'(i))$ *and* $\forall i,\ \mathsf{hsame}(y(i), y'(i))$*, the contextual continuations*

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)),$$
$$\mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n)),$$

the displayed e-one tails

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),$$
$$\mathsf{hsame}(x'(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(y'(n), \mathsf{E1}(b)),$$

and the appended-tail probes $d_X, d_Y, e_X, e_Y, w_X, w_Y, z_X, z_Y$*.*

Proof. Apply Definition 38.624 and Definition 38.625 to the same selected-transport data. The former consumes the selected index, the prefix surplus, the transport rows, the contextual continuations, the six e-one displays, and the appended-tail probes; the latter consumes the same selected index, transport, context, continuation, and displayed-tail rows without the extra appended-tail probes.

The four-package theorem Definition 38.619 accounts for the contextual endpoint package, displayed e-one classifier package, tail determinacy package, and finite-prefix readback package. Its certificate strengthening Definition 38.618 accounts for the appended-tail tests.

Projecting these packages lists no further inputs: the rows consumed are exactly the selected index, prefix surplus, pointwise transport rows, contextual continuations, displayed e-one tails, and appended-tail probes named in the statement. □

Theorem 38.627 (Real selected transport fit stability and seal obligation)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, u$ Hist*, assume the pointwise transport rows, source stream classifier row, selected context rows, continuation rows, and selected e-one displays of Definition 38.619. Then the carrier, classifier, and ledger package of Definition 38.624, Definition 38.625, and Definition 38.626 is stable under the pointwise* hsame *transports, consumes only the selected index* n*, prefix surplus* m*, contextual* Cont *rows, displayed e-one tails, and appended-tail probes already listed there, and is accepted by the* $\mathsf{Real}^{\uparrow}$ *seal through Definition 38.285.*

Proof. First collect the selected-transport package. Apply Definition 38.624, Definition 38.625, and Definition 38.626 to the same selected-transport hypotheses. These three faces supply the carrier, classifier, and finite ledger rows while naming exactly the selected index, prefix surplus, context rows, displayed e-one tails, and appended-tail probes.

Next read the stability and ledger face through the RealUp obligation boundary. Apply Definition 38.283; the pointwise hsame transports and selected finite ledger rows are precisely the stability and ledger entries required by Definition 38.285.

Finally attach the seal. Apply Definition 38.284; it accepts the same package through the $\mathsf{Real}^{\uparrow}$ seal and records that the $\mathsf{RegSeqRat}$ consumption reads no source outside the displayed carrier, classifier, stability, and ledger rows. Thus the theorem is the stability and seal face of the selected-transport obligation package, not a separate source presentation. □

Theorem 38.628 (Real selected transport fit scoped boundary package)**.** *For the selected-transport data of Definition 38.619, the carrier obligation Definition 38.624, classifier obligation Definition 38.625, and ledger obligation Definition 38.626 together form the scoped RealUp selected-transport fit package. Its source side is made of RealUp streams, RatUp rational observations, StreamNameUp selected contexts,* Cont *rows, pointwise* hsame *transport, selected e-one displays, and appended-tail probes. Its carrier choices are the rational-observation streams and selected endpoints. Its classifier choices are the* RatHistoryClassifier *row together with the* RealConstantHistoryClassifier *rows.*

The package binds the stability obligations to pointwise hsame *transport, contextual continuation closure, selected e-one tail determinacy, and positive-denominator transport. Its ledger is exactly the selected index, prefix surplus, transport rows, continuation rows, e-one displays, and appended-tail probes enumerated in Definition 38.626. Its exactness boundary is precisely the displayed carrier and classifier boundary supplied by Definition 38.624 and Definition 38.625. The standard bridge clause is the transported RegSeqRat selected endpoint normalization already carried by the same selected-transport certificates.*

Proof. Start from the four-package selected-transport certificate Definition 38.619. It fixes the RealUp stream source, RatUp observation stream, StreamNameUp selected context, pointwise hsame rows, contextual Cont rows, selected e-one displays, and finite-prefix readback surface used by the scoped package.

Apply Definition 38.617 and Definition 38.618. The first certificate supplies the displayed carrier and classifier readback data, while the second supplies the appended-tail probes that keep the endpoint exclusions inside the same selected ledger.

Apply Definition 38.620 and Definition 38.623. These identify the synchronized selected e-one endpoints and the constant-real classifier rows that turn the rational endpoint package into the stated RealUp and Completion boundary.

Now project the three obligation theorems. The carrier obligation gives the rational-observation streams, selected endpoints, unary and positive denominator rows, and endpoint exclusions. The classifier obligation gives the RatHistoryClassifier and RealConstantHistoryClassifier rows. The ledger obligation states that the selected index, prefix surplus, transport rows, continuation rows, e-one displays, and appended-tail probes are exactly the consumed resources.

No object outside Cont, hsame, RatUp, StreamNameUp, and RealUp is used in these projections. Therefore the three obligations bind the selected-transport fit site as a scoped package, with exactness given by the displayed carrier and classifier boundary and with the standard bridge supplied by the transported RegSeqRat selected endpoint normalization. □

Theorem 38.629 (Real selected transport synchronized package from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport carries the following packages simultaneously.

1. *The contextual endpoint package of Definition 38.599:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)),\\
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)),\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)),\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}),\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}),\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).
\end{aligned}$$

2. *The contextual e-one tail determinacy package of Definition 38.601:*

$$\begin{aligned}
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

3. *The finite-prefix selected full-shape readback package of Definition 38.603:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge\\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n\\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

4. *The synchronized displayed rows*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This produces the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$.

Apply Definition 38.621 with the derived prefix row, the source stream classifier row, the two pointwise transport rows, and the remaining context, continuation, and displayed e-one hypotheses. The four returned packages are exactly the four packages displayed above. □

Fit: traditional target selected RealUp observation synchronization from a stream classifier; BEDC source RealUp stream classifier plus RatUp, StreamNameUp, and Cont; carrier rational-observation streams; classifier RealStreamClassifier with derived finite-prefix row; stability pointwise hsame transport, contextual continuation, and e-one inversion; ledger selected index, prefix length, transport rows, contexts, and four tails; core theorem Definition 38.865; exactness the four synchronized packages; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Fit: traditional target synchronized transported RealUp selected-observation certificate; BEDC source RealUp stream transport plus RatUp and FKernel Cont; carrier rational-observation streams; classifier RealStreamClassifier plus displayed RatHistoryClassifier rows; stability pointwise hsame, context closure, stream-classifier finite-prefix closure, e-one inversion; ledger selected index, prefix surplus, four tails, contexts; core theorem Definition 38.621; exactness package plus synchronized classifiers; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Theorem 38.630 (Real selected transport synchronized fit from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport carries the following packages simultaneously.

1. *The contextual endpoint package of Definition 38.599:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)),\\
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)),\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)),\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}),\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}),\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).
\end{aligned}$$

2. *The contextual e-one tail determinacy package of Definition 38.601:*

$$\begin{aligned}
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

3. *The finite-prefix selected full-shape readback package of Definition 38.603:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge\\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n\\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

4. *The synchronized displayed rows*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This gives the prefix row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$.

Apply Definition 38.621 to the two pointwise transport rows, the source stream classifier row, the derived prefix row, the selected context rows, the selected continuation rows, and the six selected e-one display rows. Its conclusion is exactly the four packages listed above. □

Fit: traditional target stream-classifier-closed synchronized transported RealUp selected-observation certificate; BEDC source Definition 38.106 plus Definition 38.621; carrier rational-observation streams; classifier RealStreamClassifier with the finite-prefix row derived internally; stability pointwise hsame, context closure, e-one inversion, and selected full-shape readback; ledger selected index, prefix surplus, four tails, contexts, and derived prefix row; core theorem Definition 38.866 in future Lean namespace `BEDC.Derived.RealUp`; exactness predicate the four synchronized selected-transport packages; standard bridge RealUp selected RegSeqRat endpoint synchronization; status ProofTarget. Fit: traditional target selected RealUp observation synchronization from a stream classifier; BEDC source RealUp stream classifier plus RatUp, StreamNameUp, and Cont; carrier rational-observation streams; classifier RealStreamClassifier with derived finite-prefix row; stability pointwise hsame transport, contextual continuation, and e-one inversion; ledger selected index, prefix length, transport rows, contexts, and four tails; core theorem Definition 38.865; exactness the four synchronized packages; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Fit: traditional target synchronized transported RealUp selected-observation certificate; BEDC source RealUp stream transport plus RatUp and FKernel Cont; carrier rational-observation streams; classifier RealStreamClassifier plus displayed RatHistoryClassifier rows; stability pointwise hsame, context closure, stream-classifier finite-prefix closure, e-one inversion; ledger selected index, prefix surplus, four tails, contexts; core theorem Definition 38.621; exactness package plus synchronized classifiers; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Theorem 38.631 (Real selected transport synchronized e-one classifier clique)**.** *Under the hypotheses of Definition 38.866, the four displayed tails a, a', b, b' form a complete displayed rational-classifier clique:*

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \\ &\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b)) \wedge \\ &\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(b')).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport common rational classifier class`

Proof. Apply Definition 38.866 to the displayed hypotheses and project the synchronized displayed rows. This gives unary rows for a, a', b, b' and the four tail rows $\mathsf{hsame}(a, a')$, $\mathsf{hsame}(b, b')$, $\mathsf{hsame}(a, b)$, and $\mathsf{hsame}(a', b')$.

The two remaining tail rows come from Definition 3.8. Compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ to obtain $\mathsf{hsame}(a, b')$. Compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(a', b)$.

For each displayed pair among (a, a'), (a, b), (a, b'), (a', b), (a', b'), and (b, b'), use the corresponding two unary rows and tail hsame row. The reverse direction of Definition 37.49 lifts that data to the six displayed RatHistoryClassifier rows, which are exactly the complete clique. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport synchronized e1 classifier clique`

Fit: traditional target synchronized RegSeqRat selected-observation classifier completion; BEDC source RealUp stream classifier plus RatUp e-one classifier exactness plus FKernel hsame; carrier rational-observation streams with selected displayed e-one tails; classifier RealStreamClassifier feeding RatHistoryClassifier through E1 displays; stability pointwise hsame and tail-level symmetry and transitivity; ledger selected index, contexts, transported endpoints, unary tails, and six pairwise classifier rows; core theorem Definition 38.862 in future Lean namespace `BEDC.Derived.RealUp`; exactness predicate complete displayed e-one rational-classifier clique; downstream use RealUp dyadic endpoint normalization; status ProofTarget.

Theorem 38.632 (Real selected transport synchronized appended-tail classifier clique)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, w_X, w_Y, z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport supplies the displayed classifier clique together with the appended-tail exclusions:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b)) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(b')) \wedge \\
&(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.862 to the selected-transport hypotheses. This yields the six displayed RatHistoryClassifier rows as a single clique package.

Apply Definition 38.622 to the same selected-transport hypotheses. Project its appended-tail component to obtain the four exclusions for $o_X(n)$, $o_Y(n)$, $x'(n)$, and $y'(n)$.

Conjoin the six classifier projections with the four appended-tail projections. The resulting ten conjuncts are exactly the displayed conclusion. □

Fit: traditional target selected RegSeqRat endpoint normalization; BEDC source RealStreamClassifier, RatHistoryClassifier, Cont, and hsame; carrier rational-observation streams; classifier RealStreamClassifier feeding displayed RatHistoryClassifier rows; stability pointwise transport, context closure, e-one inversion, and positive-denominator obstruction; ledger selected index, four displayed tails, context outputs, and appended-tail probes; core theorem Definition 38.863; exactness classifier clique plus malformed-tail exclusion; standard bridge dyadic selected endpoint normalization; status ProofTarget.

Theorem 38.633 (Real selected transport common rational classifier class)**.** *Under the hypotheses of Definition 38.866, the displayed selected representative* $\mathsf{E1}(a)$ *is a RatUp classifier hub for the contextual outputs and the transported selected source endpoints:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(o_Y(n), \mathsf{E1}(a)) \wedge \\
&\mathsf{RatHistoryClassifier}(x'(n), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(y'(n), \mathsf{E1}(a)) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b'), \mathsf{E1}(a)).
\end{aligned}$$

Proof. Apply Definition 38.866 to the displayed hypotheses. Project the displayed endpoint rows $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$, $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$, $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$, and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$, together with the selected displayed classifier row $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 38.862 under the same hypotheses. This supplies the complete displayed e-one classifier clique. Use Definition 37.55 to orient the rows through the chosen center, giving $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(a))$, $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a))$, and $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b'), \mathsf{E1}(a))$.

The self-row $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a))$ follows by composing $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$ with its symmetric row using Definition 37.53. Transport this self-row along $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ by Definition 37.54, with FKernel hsame symmetry and reflexivity supplying the needed endpoint orientations. This gives the $o_X(n)$ and $x'(n)$ classifier rows centered at $\mathsf{E1}(a)$.

Finally transport $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a))$ along $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$, again using Definition 37.54 with the appropriate FKernel hsame orientations. The four transported endpoint rows together with the three oriented displayed-tail rows are exactly the stated common classifier class. □

Fit: traditional target common selected rational observation representative; BEDC source RealUp selected transport plus RatUp classifier transport plus FKernel hsame; carrier rational-observation streams; classifier RatHistoryClassifier centered at $\mathsf{E1}(a)$; stability selected endpoint hsame transport and classifier symmetry; ledger selected index, contextual outputs, transported source endpoints, and displayed e-one tails; core theorem Definition 38.864; exactness all selected endpoints read through the same rational classifier class; standard bridge RegSeqRat selected dyadic representative; status ProofTarget. Fit: traditional target selected RealUp observation synchronization from a stream classifier; BEDC source RealUp stream classifier plus RatUp, StreamNameUp, and Cont; carrier rational-observation streams; classifier RealStreamClassifier with derived finite-prefix row; stability pointwise hsame transport, contextual continuation, and e-one inversion; ledger selected index, prefix length, transport rows, contexts, and four tails; core theorem Definition 38.865; exactness the four synchronized packages; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Fit: traditional target synchronized transported RealUp selected-observation certificate; BEDC source RealUp stream transport plus RatUp and FKernel Cont; carrier rational-observation streams; classifier RealStreamClassifier plus displayed RatHistoryClassifier rows; stability pointwise hsame, context closure, stream-classifier finite-prefix closure, e-one inversion; ledger selected index, prefix surplus, four tails, contexts; core theorem Definition 38.621; exactness package plus synchronized classifiers; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Fit: traditional target stream-classifier-closed synchronized transported RealUp selected-observation certificate; BEDC source Definition 38.106 plus Definition 38.621; carrier rational-observation streams; classifier RealStreamClassifier with the finite-prefix row derived internally; stability pointwise hsame, context closure, e-one inversion, and selected full-shape readback; ledger selected index, prefix surplus, four tails, contexts, and derived prefix row; core theorem Definition 38.866 in future Lean namespace `BEDC.Derived.RealUp`; exactness predicate the four synchronized selected-transport packages; standard bridge RealUp selected RegSeqRat endpoint synchronization; status ProofTarget.

Theorem 38.634 (Real stream classifier unary denominator context closure)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, if*

$$\mathsf{RealStreamClassifier}(x, y),$$

and for every observation index n the contexts and continuation outputs satisfy

$$\mathsf{UnaryHistory}(p_X(n)),\quad \mathsf{UnaryHistory}(t_X(n)),\quad \mathsf{hsame}(p_X(n), p_Y(n)),\quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

$$\mathsf{Cont}(p_X(n), x(n), m_X(n)),\quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)),$$
$$\mathsf{Cont}(p_Y(n), y(n), m_Y(n)),\quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y(n)),$$

then

$$\mathsf{RealStreamClassifier}(o_X, o_Y).$$

Proof. Fix an observation index n. The stream classifier supplies $\mathsf{RatHistoryClassifier}(x(n), y(n))$ at that index.

Apply Definition 37.58 to this point classifier, the two unary-context hypotheses on $p_X(n)$ and $t_X(n)$, and the two displayed context comparisons $\mathsf{hsame}(p_X(n), p_Y(n))$ and $\mathsf{hsame}(t_X(n), t_Y(n))$. This gives the rational history classifier on the associativity-normal context endpoints.

The four continuation witnesses identify those normal endpoints with $o_X(n)$ and $o_Y(n)$, using associativity of append as in Definition 38.21. Transport the point classifier along these endpoint identifications.

Since n was arbitrary, repack the transported pointwise witnesses as $\mathsf{RealStreamClassifier}(o_X, o_Y)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context closed`

Theorem 38.635 (Real stream classifier transported unary denominator context closure)**.** *For rational-observation streams $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)),\quad \forall i,\ \mathsf{hsame}(y(i), y'(i)),$$

and

$$\mathsf{RealStreamClassifier}(x, y).$$

If for every observation index i the contexts and continuation outputs satisfy

$$\mathsf{UnaryHistory}(p_X(i)),\quad \mathsf{UnaryHistory}(t_X(i)),\quad \mathsf{hsame}(p_X(i), p_Y(i)),\quad \mathsf{hsame}(t_X(i), t_Y(i)),$$

$$\mathsf{Cont}(p_X(i), x'(i), m_X(i)),\quad \mathsf{Cont}(m_X(i), t_X(i), o_X(i)),$$
$$\mathsf{Cont}(p_Y(i), y'(i), m_Y(i)),\quad \mathsf{Cont}(m_Y(i), t_Y(i), o_Y(i)),$$

then

$$\mathsf{RealStreamClassifier}(o_X, o_Y).$$

Proof. Fix an observation index n. The source stream classifier supplies $\mathsf{RatHistoryClassifier}(x(n), y(n))$, and the two pointwise representative rows give $\mathsf{hsame}(x(n), x'(n))$ and $\mathsf{hsame}(y(n), y'(n))$.

Apply Definition 37.54 to this point classifier and the two endpoint comparisons. This produces $\mathsf{RatHistoryClassifier}(x'(n), y'(n))$.

Since n was arbitrary, repack these transported pointwise witnesses as $\mathsf{RealStreamClassifier}(x', y')$. Applying Definition 38.634 to that stream classifier and to the displayed contexts with middle observations x' and y' gives $\mathsf{RealStreamClassifier}(o_X, o_Y)$. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported unary denominator context closed`

Fit: traditional target representative-stable RealUp contextual closure; BEDC source RealUp streams, RatUp hsame transport, and Cont; carrier rational-observation streams; classifier RealStreamClassifier; stability pointwise representative transport followed by unary context closure; ledger pointwise transport rows and context outputs; core theorem Definition 38.635; exactness preservation of the stream classifier; standard bridge RegSeqRat representative-stable contextual observations; status ProofTarget.

Theorem 38.636 (Real stream classifier selected continuation rational classifier)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, a shared unary suffix q, and continuation outputs xq, yq, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

then

$$\mathsf{RatHistoryClassifier}(xq, yq).$$

Proof. At the selected index, the stream classifier gives $\mathsf{RatHistoryClassifier}(x(n), y(n))$. Rational unary-denominator closure extends this classifier along the shared unary suffix q, using reflexive suffix sameness. The two displayed continuation witnesses identify the closed append endpoints with xq and yq, and rational classifier transport along these endpoint displays gives the stated classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation rat classifier`

Theorem 38.637 (Real stream classifier selected continuation e-one pair readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, a shared unary suffix q, continuation outputs xq, yq, and histories $a, b : \mathsf{Hist}$, if*

$$\mathsf{RealStreamClassifier}(x, y), \quad \mathsf{UnaryHistory}(q), \quad \mathsf{Cont}(x(n), q, xq), \quad \mathsf{Cont}(y(n), q, yq),$$

and

$$\mathsf{hsame}(xq, \mathsf{E1}(a)), \quad \mathsf{hsame}(yq, \mathsf{E1}(b)),$$

then

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. At the selected index, the stream classifier gives $\mathsf{RatHistoryClassifier}(x(n), y(n))$. Close this rational classifier under the shared unary suffix q by rational unary-denominator continuation closure, using reflexive sameness of the suffix on the two sides. The two continuation displays identify the closed endpoints with xq and yq. Transport the closed rational classifier across those displays and then across the two e-one endpoint seals. Rational one-tail classifier exactness reads back the unary tails and their hsame relation. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier selected continuation e1 pair readback`

Fit: traditional target streamwise rational-denominator context stability; BEDC source RealUp streams plus RatUp classifier plus Cont; carrier rational-observation streams; classifier RealStreamClassifier; stability pointwise unary context closure; ledger pointwise prefix and tail context witnesses with continuation outputs; core theorem Definition 38.634; exactness pointwise RealStreamClassifier preservation; standard bridge denominator/context stability of rational real names; status ProofTarget.

Theorem 38.638 (Real stream classifier unary context closure)**.** *For rational-observation streams x, y and pointwise unary contexts $pref_X, pref_Y, tail_X, tail_Y$, if* $\mathsf{RealStreamClassifier}(x, y)$ *and each prefix and tail is matched by* hsame*, then the stream obtained by placing every observation inside its matched prefix and tail context is again classified by* $\mathsf{RealStreamClassifier}$*.*

Proof. At each index, apply rational history unary-denominator context closure to the point classifier supplied by the real stream classifier and to the matched unary prefix and tail histories. These pointwise classifiers assemble into the contextual stream classifier. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary context closed`

Theorem 38.639 (Real stream classifier unary denominator context selected positive denominators)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$ and a selected index n, assume*

$$\mathsf{RealStreamClassifier}(x, y),$$

and for every observation index the unary contexts, context comparisons, and continuation displays satisfy the hypotheses of Definition 38.634. Then

$$\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)).$$

Proof. At the selected index n, the stream classifier gives $\mathsf{RatHistoryClassifier}(x(n), y(n))$. Apply the rational unary denominator context closure to the two unary contexts and their hsame comparisons. The continuation displays identify the normalized contextual endpoints with $o_X(n)$ and $o_Y(n)$ by associativity of continuation append.

Transport the rational classifier to these displayed outputs and apply Definition 37.50. This yields the two positive-denominator witnesses in the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected positive denominators`

Theorem 38.640 (Real stream classifier unary denominator context selected shape package)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$ and a selected index n, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. Then the selected contextual outputs $o_X(n)$ and $o_Y(n)$ are unary histories, are disjoint from* e*, and are disjoint from every zero-extension endpoint:*

1. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$*;*
2. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$*;*
3. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$*.*

Proof. Apply unary-denominator context closure to obtain $\mathsf{RealStreamClassifier}(o_X, o_Y)$. At the selected index, the classifier gives a rational-history classifier between $o_X(n)$ and $o_Y(n)$, hence positive unary denominator witnesses for both outputs. Each positive denominator gives unary membership and nonempty exclusion, and transport of positive denominators across a displayed zero-extension contradicts the zero-extension exclusion for positive denominators. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected shape package`

Theorem 38.641 (Real stream classifier unary denominator context appended e-zero tail exclusion)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $d_X, d_Y, z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. Then the selected contextual endpoints cannot be displayed as denominators with an appended e-zero tail:*

$$(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}).$$

Proof. Apply Definition 38.639 to the same selected index and pointwise context hypotheses. This gives positive unary denominator witnesses for both $o_X(n)$ and $o_Y(n)$.

For the left endpoint, suppose $\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X))$. Transport the positive denominator witness across this displayed hsame to obtain a positive denominator witness for $d_X \cdot \mathsf{E0}(z_X)$. This contradicts Definition 37.15.

The right endpoint is identical: a display $\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y))$ transports the positive denominator witness for $o_Y(n)$ to the appended e-zero-tail denominator $d_Y \cdot \mathsf{E0}(z_Y)$, again contradicting Definition 37.15. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context appended e0 tail absurd`

Theorem 38.642 (Real stream classifier unary denominator context selected classifier shape package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and zero-extension tests* $z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. Then the selected contextual endpoints carry the classifier-and-shape package*

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$*;*
2. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$*;*
3. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$*;*
4. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$*;*
5. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Proof. First apply Definition 38.634 to the given RealUp stream classifier and the pointwise context-closure hypotheses. This gives $\mathsf{RealStreamClassifier}(o_X, o_Y)$, and evaluating it at the selected index n gives $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.

Next apply Definition 38.639 to the same selected instance. This supplies the two positive unary denominator rows for $o_X(n)$ and $o_Y(n)$.

Finally apply Definition 38.640 with the same hypotheses and the two zero-extension tests. This supplies the unary-history rows, the nonempty exclusions, and the two zero-headed exclusions, which together with the selected rational classifier and positivity rows are exactly the displayed package. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected classifier shape package`

Theorem 38.643 (Real stream classifier unary denominator context one-tail pair readback)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$ *and a selected index* n*,*

assume $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. If the displayed contextual outputs satisfy*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. First apply Definition 38.634 to obtain $\mathsf{RealStreamClassifier}(o_X, o_Y)$. At the selected index, transport the point classifier to the displayed one-extension endpoints and apply the selected RealUp one-tail pair readback theorem. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context e1 pair readback`

Theorem 38.644 (Real stream classifier unary denominator context selected full endpoint package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, histories* $a, b : \mathsf{Hist}$*, and zero-extension tests* $z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. If the selected contextual outputs also satisfy*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then the selected contextual endpoints carry all of the following data:

1. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$*;*
2. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$*;*
3. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$*;*
4. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$*;*
5. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$*.*

Proof. First apply Definition 38.639 under the selected instance of the context-closure hypotheses. This supplies the two positive unary denominator witnesses for $o_X(n)$ and $o_Y(n)$.

Next apply Definition 38.640 under the same context-closure hypotheses and the displayed zero-extension tests. This supplies the unary-history witnesses, the nonempty exclusions, and the zero-extension exclusions for the two selected outputs.

Finally apply Definition 38.643 to the same context-closure hypotheses together with the two one-extension seals. The readback row gives the unary tails and their hsame relation, and the five displayed conclusions are exactly the collected endpoint data. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected full endpoint package`

Theorem 38.645 (Real stream classifier iterated unary context selected full endpoint package)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, u_X, u_Y, q_X, q_Y, s_X, s_Y, v_X, v_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, histories* $a, b : \mathsf{Hist}$*, and zero-extension tests* $z_X, z_Y : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$*. Assume also that for every observation index* k *the first context layer satisfies*

$$\mathsf{UnaryHistory}(p_X(k)), \quad \mathsf{UnaryHistory}(t_X(k)), \quad \mathsf{hsame}(p_X(k), p_Y(k)), \quad \mathsf{hsame}(t_X(k), t_Y(k)),$$

$$\mathsf{Cont}(p_X(k), x(k), m_X(k)), \quad \mathsf{Cont}(m_X(k), t_X(k), u_X(k)),$$
$$\mathsf{Cont}(p_Y(k), y(k), m_Y(k)), \quad \mathsf{Cont}(m_Y(k), t_Y(k), u_Y(k)),$$

and that for every observation index k the second context layer satisfies

$$\mathsf{UnaryHistory}(q_X(k)), \quad \mathsf{UnaryHistory}(s_X(k)), \quad \mathsf{hsame}(q_X(k), q_Y(k)), \quad \mathsf{hsame}(s_X(k), s_Y(k)),$$
$$\mathsf{Cont}(q_X(k), u_X(k), v_X(k)), \quad \mathsf{Cont}(v_X(k), s_X(k), o_X(k)),$$
$$\mathsf{Cont}(q_Y(k), u_Y(k), v_Y(k)), \quad \mathsf{Cont}(v_Y(k), s_Y(k), o_Y(k)).$$

If the selected second-layer outputs satisfy

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then the selected contextual endpoints carry all of the following data:

1. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$;
2. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$;
3. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$;
4. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$;
5. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First apply Definition 38.634 to the given RealUp stream classifier and the first-layer unary context and continuation rows. This produces $\mathsf{RealStreamClassifier}(u_X, u_Y)$.

Next apply Definition 38.644 to this intermediate stream classifier, using the second-layer unary context and continuation rows, the selected one-extension seals, and the two zero-extension tests. The resulting five rows are exactly the displayed endpoint package for $o_X(n)$ and $o_Y(n)$. □

Fit: traditional target RealUp iterated contextual endpoint readback; BEDC source RealUp streams plus RatUp classifier plus Cont; carrier rational-observation streams; classifier RealStreamClassifier; stability consecutive unary context closure; ledger two context layers and selected e-one seals; core theorem Definition 38.645; exactness the displayed endpoint package; standard bridge RegSeqRat iterated observation readback; status ProofTarget.

Theorem 38.646 (Real stream classifier unary denominator context selected classifier full endpoint package)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, histories $a, b : \mathsf{Hist}$, and zero-extension tests $z_X, z_Y : \mathsf{Hist}$, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context and continuation hypotheses of Definition 38.634. If the selected contextual outputs also satisfy*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then the selected contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$;
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$;
3. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$;
4. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$;

5. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$;

6. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$;

7. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First apply Definition 38.634 to the given RealUp stream classifier and the pointwise context-closure hypotheses. This supplies $\mathsf{RealStreamClassifier}(o_X, o_Y)$.

By the definition of RealStreamClassifier, evaluating the closed stream classifier at the selected index n gives $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$.

Transport this endpoint classifier across the two one-extension seals using Definition 37.54. This gives $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Next apply Definition 38.644 to the same hypotheses, the two displayed one-extension seals, and the zero-extension tests. This gives the positivity, unary-shape, nonempty, zero-headed exclusion, and e-one tail readback rows.

Package the selected rational-history classifier and its sealed transported classifier as the first two rows, then append the five rows supplied by the full endpoint package. □

Fit: traditional target RealUp contextual endpoint classifier readback; BEDC source RealUp streams, RatUp classifier, Cont, and unary contexts; carrier rational-observation streams; classifier RealStreamClassifier with selected and sealed RatHistoryClassifier; stability context closure and endpoint selection; ledger selected index, contexts, continuation outputs, e-one and zero-headed tests; core theorem Definition 38.646; exactness selected and sealed RatUp classifiers, denominator/shape rows, and tail exclusions; standard bridge RegSeqRat selected endpoint classifier readback; status ProofTarget.

Theorem 38.647 (Real stream classifier unary denominator context selected classifier appended-tail package)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, histories $a, b, d_X, d_Y, z_X, z_Y : \mathsf{Hist}$, assume $\mathsf{RealStreamClassifier}(x, y)$ and the pointwise unary-context and continuation hypotheses of Definition 38.634. If the selected contextual outputs also satisfy the one-extension seals*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then the selected contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$;

2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$;

3. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$;

4. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$;

5. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$;

6. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$;

7. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$;

8. $(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False})$.

Proof. First apply Definition 38.646 to the same RealUp stream classifier, pointwise context-closure hypotheses, and one-extension seals. This supplies the selected rational-history classifier, the sealed

transported rational-history classifier, the positivity rows, the unary-shape rows, the nonempty and zero-headed exclusions, and the e-one tail readback row.

Next apply Definition 38.641 to the same selected contextual instance. This gives the two exclusions against displays of $o_X(n)$ and $o_Y(n)$ as denominators with appended e-zero tails.

Repack the seven rows from the selected classifier full endpoint package with the sealed classifier preceding the appended-tail exclusions. The resulting tuple is exactly the displayed package for the selected contextual endpoints. □

Fit: traditional target RealUp contextual endpoint classifier readback; BEDC source RealUp streams, RatUp classifier, Cont, unary contexts, and appended e-zero-tail obstruction; carrier rational-observation streams; classifier RealStreamClassifier and selected RatHistoryClassifier together with its sealed endpoint transport; stability context closure and endpoint selection; ledger selected index, contexts, continuation outputs, e-one seals, zero-headed tests, and appended-tail displays; core theorem Definition 38.647; exactness selected and sealed RatUp classifiers, denominator/shape rows, and tail exclusions; standard bridge RegSeqRat selected endpoint classifier readback; status ProofTarget.

Theorem 38.648 (Real stream classifier contextual appended e-one tail cancellation)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $d, e, tail_D, tail_E : \mathsf{Hist}$*, assume*

$$\mathsf{RealStreamClassifier}(x, y),$$

and for every observation index i *the contexts and continuation outputs satisfy*

$$\mathsf{UnaryHistory}(p_X(i)), \quad \mathsf{UnaryHistory}(t_X(i)), \quad \mathsf{hsame}(p_X(i), p_Y(i)), \quad \mathsf{hsame}(t_X(i), t_Y(i)),$$

$$\mathsf{Cont}(p_X(i), x(i), m_X(i)), \quad \mathsf{Cont}(m_X(i), t_X(i), o_X(i)),$$
$$\mathsf{Cont}(p_Y(i), y(i), m_Y(i)), \quad \mathsf{Cont}(m_Y(i), t_Y(i), o_Y(i)).$$

If the selected contextual outputs display as appended e-one tails,

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(d \cdot \mathsf{E1}(tail_D))), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(e \cdot \mathsf{E1}(tail_E))),$$

and the displayed tails agree by $\mathsf{hsame}(tail_D, tail_E)$*, then*

$$\mathsf{hsame}(d, e).$$

Proof. First apply Definition 38.634 to the source stream classifier and the pointwise unary-context and continuation hypotheses. This supplies the contextual stream classifier $\mathsf{RealStreamClassifier}(o_X, o_Y)$.

Apply Definition 38.605 to the contextual streams o_X, o_Y at the selected index n, using the two displayed appended e-one tail rows and the comparison $\mathsf{hsame}(tail_D, tail_E)$. The conclusion is exactly the denominator-head comparison $\mathsf{hsame}(d, e)$. □

Fit: traditional target contextual RegSeqRat endpoint cancellation; BEDC source RealUp streams, RatUp classifier readback, Cont, and BEDC append cancellation; carrier rational-observation streams; classifier RealStreamClassifier; stability unary denominator context closure; ledger selected index plus context witnesses and e-one tail displays; core theorem Definition 38.648; exactness denominator-head comparison after tail cancellation; standard bridge contextual rational observation tail cancellation; status ProofTarget.

Theorem 38.649 (Real stream classifier unary denominator context selected e-one tail determinacy)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, a', b, b' : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context plus continuation hypotheses of Definition 38.634. If the selected contextual outputs satisfy the four e-one displays*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),$$

then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$*;*
2. $\mathsf{UnaryHistory}(a')$*;*
3. $\mathsf{UnaryHistory}(b)$*;*
4. $\mathsf{UnaryHistory}(b')$*;*
5. $\mathsf{hsame}(a, a')$*;*
6. $\mathsf{hsame}(b, b')$*;*
7. $\mathsf{hsame}(a, b)$*;*
8. $\mathsf{hsame}(a', b')$*.*

Proof. Apply Definition 38.643 to the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. This supplies $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply the same readback theorem again to the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b'))$. This supplies $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the X-tail comparison, compose the first display of $o_X(n)$ with the symmetry of the second display. The resulting $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ is inverted by the history constructor characterization, equivalently BEDC.FKernel.Hist.hsame_e1_inversion_iff, to get $\mathsf{hsame}(a, a')$.

The same composition and e-one inversion applied to the two displays of $o_Y(n)$ gives $\mathsf{hsame}(b, b')$. Package these two tail comparisons with the two readback rows. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected e1 tail determinacy`

Fit: traditional target RealUp contextual endpoint tail determinacy; BEDC source RealUp streams, RatUp classifier, Cont, unary contexts, and selected e-one displays; carrier rational-observation streams; classifier RealStreamClassifier and selected endpoint displays; stability context closure and selected continuation transport; ledger selected index, contexts, continuation outputs, and paired e-one seals; core theorem Definition 38.649; exactness canonical selected denominator tails; standard bridge RegSeqRat and dyadic selected-tail consumers; status ProofTarget.

Theorem 38.650 (Real stream classifier unary denominator context selected e-one tail pairwise coherence)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, a', b, b' : \mathsf{Hist}$*, assume* $\mathsf{RealStreamClassifier}(x, y)$ *and the pointwise unary-context plus continuation hypotheses of Definition 38.634. If the selected contextual outputs satisfy the four e-one displays*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),$$

then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$;
2. $\mathsf{UnaryHistory}(a')$;
3. $\mathsf{UnaryHistory}(b)$;
4. $\mathsf{UnaryHistory}(b')$;
5. $\mathsf{hsame}(a, a')$;
6. $\mathsf{hsame}(a, b)$;
7. $\mathsf{hsame}(a, b')$;
8. $\mathsf{hsame}(a', b)$;
9. $\mathsf{hsame}(a', b')$;
10. $\mathsf{hsame}(b, b')$.

Proof. Apply Definition 38.649 to the same RealUp stream classifier, selected index, pointwise unary-context and continuation hypotheses, and four e-one displays. This supplies the four unary-history rows together with $\mathsf{hsame}(a, a')$, $\mathsf{hsame}(b, b')$, $\mathsf{hsame}(a, b)$, and $\mathsf{hsame}(a', b')$.

For the first cross-tail comparison, compose $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a', b')$ by BEDC.FKernel.Hist.hsame_trans. The resulting row is $\mathsf{hsame}(a, b')$.

For the second cross-tail comparison, apply BEDC.FKernel.Hist.hsame_symm to $\mathsf{hsame}(a, a')$ and compose the result with $\mathsf{hsame}(a, b)$ by BEDC.FKernel.Hist.hsame_trans. This gives $\mathsf{hsame}(a', b)$.

Package the four unary-history witnesses, the four tail rows supplied by Definition 38.649, and the two cross-tail rows just obtained. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier unary denominator context selected e1 tail pairwise coherence`

Fit: traditional target canonical selected denominator-tail class for contextual RealUp observations; BEDC source RealUp streams, RatUp readback, Cont contexts, and FKernel hsame transitivity; carrier rational-observation streams; classifier RealStreamClassifier; stability selected context closure plus e-one no-confusion; ledger selected index, context witnesses, continuation outputs, and four e-one seals; core theorem Definition 38.650; exactness pairwise selected-tail coherence; standard bridge RegSeqRat selected dyadic-tail canonicalization; status ProofTarget.

Theorem 38.651 (Real stream classifier unary denominator context selected e-one tail-class exactness)**.** *For rational-observation streams $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, and histories $a, b : \mathsf{Hist}$, assume $\mathsf{RealStreamClassifier}(x, y)$ and the pointwise unary-context plus continuation hypotheses of Definition 38.634. If the selected contextual outputs satisfy the displayed endpoint seals*

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then for all histories $u, v : \mathsf{Hist}$,

$$(\mathsf{hsame}(o_X(n), \mathsf{E1}(u)) \wedge \mathsf{hsame}(o_Y(n), \mathsf{E1}(v))) \iff (\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(a, u) \wedge \mathsf{hsame}(b, v)).$$

Proof. For the forward direction, assume the two additional displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(u))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(v))$. Apply Definition 38.650 with $(a, a', b, b') = (a, u, b, v)$ to the two endpoint seals and the two additional displays. The resulting ten rows include $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, $\mathsf{hsame}(a, u)$, and $\mathsf{hsame}(b, v)$.

For the reverse direction, assume $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, $\mathsf{hsame}(a, u)$, and $\mathsf{hsame}(b, v)$. Apply BEDC.FKernel.Hist.hsame_e1_congr to $\mathsf{hsame}(a, u)$ and compose the result with the endpoint seal $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ by BEDC.FKernel.Hist.hsame_trans. This gives $\mathsf{hsame}(o_X(n), \mathsf{E1}(u))$.

Apply the same e-one congruence and transitivity chain to $\mathsf{hsame}(b, v)$ and the endpoint seal $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. This gives $\mathsf{hsame}(o_Y(n), \mathsf{E1}(v))$, and the two displays form the selected tail-class membership pair. □

Fit: traditional target contextual selected denominator-tail classes; BEDC source RealUp streams, RatUp readback, Cont, UnaryHistory, and FKernel e-one congruence; carrier rational-observation streams; classifier RealStreamClassifier; stability pairwise coherence plus e-one congruence; ledger selected index, context witnesses, continuation outputs, and selected e-one seals; core theorem Definition 38.651; exactness the displayed selected e-one tails are exactly the canonical hsame classes; standard bridge RegSeqRat selected dyadic-tail canonicalization; status ProofTarget.

Theorem 38.652 (Real stream classifier transported contextual e-one tail pairwise coherence)**.** *For rational-observation streams $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$, a selected index n, and histories $a, a', b, b' : \mathsf{Hist}$, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)),$$

and

$$\mathsf{RealStreamClassifier}(x, y).$$

Assume also that for every observation index i the transported unary contexts and continuation outputs satisfy

$$\mathsf{UnaryHistory}(p_X(i)), \quad \mathsf{UnaryHistory}(t_X(i)), \quad \mathsf{hsame}(p_X(i), p_Y(i)), \quad \mathsf{hsame}(t_X(i), t_Y(i)),$$

$$\mathsf{Cont}(p_X(i), x'(i), m_X(i)), \quad \mathsf{Cont}(m_X(i), t_X(i), o_X(i)),$$
$$\mathsf{Cont}(p_Y(i), y'(i), m_Y(i)), \quad \mathsf{Cont}(m_Y(i), t_Y(i), o_Y(i)).$$

If the selected contextual outputs satisfy the four e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),$$

then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$;
2. $\mathsf{UnaryHistory}(a')$;
3. $\mathsf{UnaryHistory}(b)$;
4. $\mathsf{UnaryHistory}(b')$;
5. $\mathsf{hsame}(a, a')$;
6. $\mathsf{hsame}(a, b)$;
7. $\mathsf{hsame}(a, b')$;
8. $\mathsf{hsame}(a', b)$;
9. $\mathsf{hsame}(a', b')$;
10. $\mathsf{hsame}(b, b')$.

Proof. Apply Definition 38.635 to the pointwise representative rows, the source RealUp stream classifier, and the displayed transported unary-context hypotheses. This gives $\mathsf{RealStreamClassifier}(o_X, o_Y)$.

Apply Definition 38.588 to this endpoint classifier with the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. This supplies $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$.

Apply the same selected pair readback theorem to the displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(a'))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b'))$. This supplies $\mathsf{UnaryHistory}(a')$, $\mathsf{UnaryHistory}(b')$, and $\mathsf{hsame}(a', b')$.

For the left representative comparison, compose the two $o_X(n)$ displays with the symmetry of the second display and invert the resulting $\mathsf{hsame}(\mathsf{E1}(a), \mathsf{E1}(a'))$ by e-one no-confusion. This gives $\mathsf{hsame}(a, a')$.

The same composition and e-one inversion applied to the two $o_Y(n)$ displays gives $\mathsf{hsame}(b, b')$.

Finally compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ by BEDC.FKernel.Hist.hsame_trans to obtain $\mathsf{hsame}(a, b')$. Compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ by the same transitivity theorem to obtain $\mathsf{hsame}(a', b)$, and package these rows with the two readback triples and the two same-side representative comparisons. □

Theorem 38.653 (Real stream classifier transported iterated unary context selected e-one tail pairwise coherence)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, u_X, u_Y, q_X, q_Y, s_X, s_Y, v_X,$ $\mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, a', b, b' : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)),$$

and

$$\mathsf{RealStreamClassifier}(x, y).$$

Assume also that for every observation index i *the transported first context layer satisfies*

$$\mathsf{UnaryHistory}(p_X(i)), \quad \mathsf{UnaryHistory}(t_X(i)), \quad \mathsf{hsame}(p_X(i), p_Y(i)), \quad \mathsf{hsame}(t_X(i), t_Y(i)),$$

$$\mathsf{Cont}(p_X(i), x'(i), m_X(i)), \quad \mathsf{Cont}(m_X(i), t_X(i), u_X(i)),$$

$$\mathsf{Cont}(p_Y(i), y'(i), m_Y(i)), \quad \mathsf{Cont}(m_Y(i), t_Y(i), u_Y(i)),$$

and that for every observation index i *the second context layer satisfies*

$$\mathsf{UnaryHistory}(q_X(i)), \quad \mathsf{UnaryHistory}(s_X(i)), \quad \mathsf{hsame}(q_X(i), q_Y(i)), \quad \mathsf{hsame}(s_X(i), s_Y(i)),$$

$$\mathsf{Cont}(q_X(i), u_X(i), v_X(i)), \quad \mathsf{Cont}(v_X(i), s_X(i), o_X(i)),$$

$$\mathsf{Cont}(q_Y(i), u_Y(i), v_Y(i)), \quad \mathsf{Cont}(v_Y(i), s_Y(i), o_Y(i)).$$

If the selected second-layer outputs satisfy the four e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')),$$

then all of the following data hold:

1. $\mathsf{UnaryHistory}(a)$*;*
2. $\mathsf{UnaryHistory}(a')$*;*
3. $\mathsf{UnaryHistory}(b)$*;*
4. $\mathsf{UnaryHistory}(b')$*;*
5. $\mathsf{hsame}(a, a')$*;*

6. $\mathsf{hsame}(a, b)$;
7. $\mathsf{hsame}(a, b')$;
8. $\mathsf{hsame}(a', b)$;
9. $\mathsf{hsame}(a', b')$;
10. $\mathsf{hsame}(b, b')$.

Proof. First apply Definition 38.635 to the pointwise representative rows, the source RealUp stream classifier, and the displayed first-layer context and continuation hypotheses. This produces $\mathsf{RealStreamClassifier}(u_X, u_Y)$.

Next apply Definition 38.650 to this transported intermediate classifier, using the second-layer context and continuation rows together with the four selected e-one displays. The resulting ten rows are exactly the unary-history and pairwise tail-coherence package displayed above. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamClassifier transported iterated unary context selected e1 tail pairwise coherence`

Fit: traditional target RegSeqRat representative-stable iterated selected-tail canonicalization; BEDC source RealUp streams plus RatUp classifier plus Cont; carrier rational-observation streams; classifier RealStreamClassifier; stability representative transport followed by two context closures; ledger two context layers and four selected e-one seals; core theorem Definition 38.653; exactness the selected-tail equivalence class; standard bridge dyadic selected-tail canonicalization; status ProofTarget.

Theorem 38.654 (Real stream classifier iterated unary denominator context appended e-zero tail exclusion)**.** *For rational-observation streams* $x, y, p_X, p_Y, t_X, t_Y, m_X, m_Y, u_X, u_Y, q_X, q_Y, s_X, s_Y, v_X, v_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $d_X, d_Y, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\mathsf{RealStreamClassifier}(x, y).$$

Assume also that for every observation index k *the first context layer satisfies*

$$\mathsf{UnaryHistory}(p_X(k)), \quad \mathsf{UnaryHistory}(t_X(k)), \quad \mathsf{hsame}(p_X(k), p_Y(k)), \quad \mathsf{hsame}(t_X(k), t_Y(k)),$$
$$\mathsf{Cont}(p_X(k), x(k), m_X(k)), \quad \mathsf{Cont}(m_X(k), t_X(k), u_X(k)),$$
$$\mathsf{Cont}(p_Y(k), y(k), m_Y(k)), \quad \mathsf{Cont}(m_Y(k), t_Y(k), u_Y(k)),$$

and that for every observation index k *the second context layer satisfies*

$$\mathsf{UnaryHistory}(q_X(k)), \quad \mathsf{UnaryHistory}(s_X(k)), \quad \mathsf{hsame}(q_X(k), q_Y(k)), \quad \mathsf{hsame}(s_X(k), s_Y(k)),$$
$$\mathsf{Cont}(q_X(k), u_X(k), v_X(k)), \quad \mathsf{Cont}(v_X(k), s_X(k), o_X(k)),$$
$$\mathsf{Cont}(q_Y(k), u_Y(k), v_Y(k)), \quad \mathsf{Cont}(v_Y(k), s_Y(k), o_Y(k)).$$

Then the selected second-layer outputs cannot be displayed as denominators with an appended e-zero tail:

$$(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}).$$

Proof. First apply Definition 38.634 to the source RealUp stream classifier and the first-layer unary context and continuation rows. This produces the intermediate stream classifier $\mathsf{RealStreamClassifier}(u_X, u_Y)$.

Next apply Definition 38.641 to this intermediate stream classifier, using the second-layer unary context and continuation rows at the same selected index. The resulting two contradictions are exactly the displayed exclusions for $o_X(n)$ and $o_Y(n)$. □

Fit: traditional target RegSeqRat iterated observation denominator sanity; BEDC source RealUp plus RatUp plus Cont; carrier rational-observation streams; classifier RealStreamClassifier; stability two unary-context closures; ledger two context layers and appended e-zero displays; core theorem Definition 38.654; exactness selected appended-tail impossibility; standard bridge finite rational observation windows; status ProofTarget.

Theorem 38.655 (Real stream classifier transported iterated unary context selected e-one tail–class exactness)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, u_X, u_Y, q_X, q_Y, s_X, s_Y, v_X, v_Y, \mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, and histories* $a, b : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)),$$

and

$$\mathsf{RealStreamClassifier}(x, y).$$

Assume also that for every observation index i *the transported first context layer satisfies*

$$\mathsf{UnaryHistory}(p_X(i)), \quad \mathsf{UnaryHistory}(t_X(i)), \quad \mathsf{hsame}(p_X(i), p_Y(i)), \quad \mathsf{hsame}(t_X(i), t_Y(i)),$$
$$\mathsf{Cont}(p_X(i), x'(i), m_X(i)), \quad \mathsf{Cont}(m_X(i), t_X(i), u_X(i)),$$
$$\mathsf{Cont}(p_Y(i), y'(i), m_Y(i)), \quad \mathsf{Cont}(m_Y(i), t_Y(i), u_Y(i)),$$

and that for every observation index i *the second context layer satisfies*

$$\mathsf{UnaryHistory}(q_X(i)), \quad \mathsf{UnaryHistory}(s_X(i)), \quad \mathsf{hsame}(q_X(i), q_Y(i)), \quad \mathsf{hsame}(s_X(i), s_Y(i)),$$
$$\mathsf{Cont}(q_X(i), u_X(i), v_X(i)), \quad \mathsf{Cont}(v_X(i), s_X(i), o_X(i)),$$
$$\mathsf{Cont}(q_Y(i), u_Y(i), v_Y(i)), \quad \mathsf{Cont}(v_Y(i), s_Y(i), o_Y(i)).$$

If the selected second-layer outputs satisfy

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then for all histories $u, v : \mathsf{Hist}$*,*

$$\big(\mathsf{hsame}(o_X(n), \mathsf{E1}(u)) \wedge \mathsf{hsame}(o_Y(n), \mathsf{E1}(v))\big) \iff \big(\mathsf{UnaryHistory}(u) \wedge \mathsf{UnaryHistory}(v) \wedge \mathsf{hsame}(a, u) \wedge \mathsf{hsame}(b, v)\big).$$

Proof. For the forward direction, assume the two additional displays $\mathsf{hsame}(o_X(n), \mathsf{E1}(u))$ and $\mathsf{hsame}(o_Y(n), \mathsf{E1}(v))$. Apply Definition 38.653 with $(a, a', b, b') = (a, u, b, v)$ to the two selected endpoint seals and the two additional displays. The resulting ten rows include $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, $\mathsf{hsame}(a, u)$, and $\mathsf{hsame}(b, v)$.

For the reverse direction, assume $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(v)$, $\mathsf{hsame}(a, u)$, and $\mathsf{hsame}(b, v)$. Apply BEDC.FKernel.Hist.hsame_e1_congr to $\mathsf{hsame}(a, u)$ and compose the result with the endpoint seal $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ by BEDC.FKernel.Hist.hsame_trans. This gives $\mathsf{hsame}(o_X(n), \mathsf{E1}(u))$.

Apply the same e-one congruence and transitivity chain to $\mathsf{hsame}(b, v)$ and the endpoint seal $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$. This gives $\mathsf{hsame}(o_Y(n), \mathsf{E1}(v))$, and the two displays form the selected tail-class membership pair. □

Fit: traditional target RegSeqRat selected denominator-tail class; BEDC source RealUp streams, RatUp readback, Cont contexts, and FKernel e-one congruence; carrier rational-observation streams; classifier RealStreamClassifier; stability representative transport plus two unary context layers; ledger selected index, context witnesses, continuation outputs, and e-one seals; core theorem Definition 38.655; exactness canonical selected-tail class; standard bridge dyadic selected-tail canonicalization; status ProofTarget.

Theorem 38.656 (Real stream classifier transported iterated unary context selected classifier full endpoint package)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, u_X, u_Y, q_X, q_Y, s_X, s_Y, v_X,$ $\mathbb{N} \to \mathsf{Hist}$*, a selected index* n*, histories* $a, b, z_X, z_Y : \mathsf{Hist}$*, assume*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)),$$

and

$$\mathsf{RealStreamClassifier}(x, y).$$

Assume also that for every observation index i *the transported first unary-context layer satisfies*

$$\mathsf{UnaryHistory}(p_X(i)), \quad \mathsf{UnaryHistory}(t_X(i)), \quad \mathsf{hsame}(p_X(i), p_Y(i)), \quad \mathsf{hsame}(t_X(i), t_Y(i)),$$

$$\mathsf{Cont}(p_X(i), x'(i), m_X(i)), \quad \mathsf{Cont}(m_X(i), t_X(i), u_X(i)),$$
$$\mathsf{Cont}(p_Y(i), y'(i), m_Y(i)), \quad \mathsf{Cont}(m_Y(i), t_Y(i), u_Y(i)),$$

and that for every observation index i *the second unary-context layer satisfies*

$$\mathsf{UnaryHistory}(q_X(i)), \quad \mathsf{UnaryHistory}(s_X(i)), \quad \mathsf{hsame}(q_X(i), q_Y(i)), \quad \mathsf{hsame}(s_X(i), s_Y(i)),$$

$$\mathsf{Cont}(q_X(i), u_X(i), v_X(i)), \quad \mathsf{Cont}(v_X(i), s_X(i), o_X(i)),$$
$$\mathsf{Cont}(q_Y(i), u_Y(i), v_Y(i)), \quad \mathsf{Cont}(v_Y(i), s_Y(i), o_Y(i)).$$

If the selected second-layer outputs satisfy

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)),$$

then the selected transported contextual endpoints carry all of the following data:

1. $\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n))$*;*
2. $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$*;*
3. $\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n))$*;*
4. $\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n))$*;*
5. $(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False})$*;*
6. $(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False})$*;*
7. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$*.*

Proof. First apply Definition 38.635 to the pointwise representative rows, the source RealUp stream classifier, and the displayed transported first-layer context and continuation hypotheses. This produces $\mathsf{RealStreamClassifier}(u_X, u_Y)$.

Next apply Definition 38.646 to this intermediate stream classifier, the second-layer context and continuation rows, the selected one-extension seals, and the two zero-extension tests. The resulting seven rows are exactly the selected transported classifier full endpoint package displayed above. □

Fit: traditional target RealUp selected transported observation readback; BEDC source RealStreamClassifier, RatHistoryClassifier, Cont, UnaryHistory, and FKernel hsame; carrier rational-observation streams; classifier RealStreamClassifier with selected sealed RatHistoryClassifier; stability representative transport followed by two unary context layers; ledger selected index, context witnesses, continuation outputs, e-one seals, and zero-headed tests; core theorem Definition 38.656; exactness selected and sealed RatUp classifiers, denominator and unary-shape rows, nonempty exclusions, zero-headed exclusions, and e-one tail readback; standard bridge RegSeqRat selected transported observation readback; status ProofTarget. The $\mathsf{Real}^{\uparrow}$ carrier is consumed only after the finite-window $\mathsf{StreamName}^{\uparrow}$ surface of chapter 126 and the regular-rational readback surface of chapter 301 are both available. In this dependency row, a real seal is a downstream consumer of the visible stream window and regular rational approximation ledgers; it does not create a quotient-real equality row, selected host limit, or ambient completeness principle before those two finite surfaces have been displayed.

Theorem 38.657 (Real StreamName RegSeqRat pointwise seal composition)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal-facing read whose source includes a* $\mathsf{StreamName}^{\uparrow}$ *finite-window row and whose readback includes a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational row factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise handoff, and then the local* RealUnaryStreamClassifier *row. The route is transported only by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already present in the Real carrier and seal material of this chapter.*

The composed route exports no quotient-real equality, selected host limit, ambient completeness principle, or direct $\mathsf{Real}^{\uparrow}$ *seal edge bypassing the StreamName and RegSeqRat surfaces.*

Proof. Start with an accepted seal-facing read at the $\mathsf{Real}^{\uparrow}$ boundary. The Real carrier and seal material already input in this chapter exposes the read only through the local finite-window seal package, endpoint readback, and provenance rows.

First project the StreamName source. By chapter 126 and Definition 126.170, the finite source contributes exactly the visible StreamName window ledger, bundle membership, unary-history observations, and pointwise hsame comparisons. These are the only source rows available before the regular-rational readback is consumed.

Next project the RegSeqRat readback. The RegSeqRat surface of chapter 301, together with Definition 38.407, reads the same finite window as the regular-rational handoff and keeps the readback on the displayed BHist route.

Finally transport the result into the Real unary stream classifier. The local Real seal package reads the already assembled StreamName and RegSeqRat rows as the classifier-facing input, so the seal-facing route has no remaining coordinate from which to obtain quotient-real equality, a selected host limit, an ambient completeness principle, or a direct seal edge around those two surfaces. □

Theorem 38.658 (RealUp StreamName RegSeqRat dyadic window exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal-facing finite-window read whose source is a* $\mathsf{StreamName}^{\uparrow}$ *window and whose spine is a* $\mathsf{RegSeqRat}^{\uparrow}$ *row is exhausted by the route*

StreamName finite-window ledger → *RegSeqRat pointwise handoff* → *DyadicRatCore observation row* → RealUnar

The route reads only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by the StreamName, RegSeqRat, DyadicRatCore, and Real finite-window surfaces. No quotient-real equality, selected host limit, ambient completeness row, or off-ledger dyadic sample can determine the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with the pointwise seal composition of Definition 38.657. It already forces the seal-facing read to pass through the displayed StreamName finite-window ledger and the RegSeqRat pointwise handoff before it reaches the local real classifier.

The handoff is exhaustive by Definition 126.100. Thus the source window contributes only its visible finite ledger, unary-history observations, bundle membership, and pointwise hsame comparisons; it cannot add a second stream source or a hidden real equality row.

The dyadic part is then read through the public DyadicRatCore observation and source-boundary surface. That surface supplies the observation row used by the real finite-window classifier, while the RealUp finite-window coverage rows identify the classifier-facing input with $\mathsf{RealUnaryStreamClassifier}$.

Composing these four displayed routes leaves no unused coordinate in the seal-facing packet. A host limit, ambient completeness row, quotient-real equality, or off-ledger dyadic sample is absent from the route and therefore cannot determine the accepted $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 38.659 (Real StreamName RegSeqRat terminal seal exhaustion)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer whose source is a* $\mathsf{StreamName}^{\uparrow}$ *finite-window row and whose regularity witness is a* $\mathsf{RegSeqRat}^{\uparrow}$ *row factors through exactly the route*

StreamName finite-window ledger → *RegSeqRat pointwise handoff* → *DyadicRatCore observation row* → $\mathsf{RealUnar}$

before the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row can be read. The terminal packet has no additional carrier coordinate for quotient-real equality, a selected host limit, ambient completeness, an off-ledger dyadic sample, or a direct* $\mathsf{Real}^{\uparrow}$ *seal edge bypassing either the StreamName source surface or the RegSeqRat regularity surface.*

Proof. Apply the pointwise seal composition of Definition 38.657. It sends any accepted seal-facing read with a StreamName source and RegSeqRat readback through the StreamName finite-window ledger, the RegSeqRat pointwise handoff, and the local real classifier-facing row.

Next apply the dyadic-window exhaustion of Definition 38.658. This identifies the only remaining classifier input as the DyadicRatCore observation row feeding $\mathsf{RealUnaryStreamClassifier}$, with no off-ledger dyadic sample and no ambient real-completion coordinate.

It remains only to read the terminal seal packet itself. Its carrier coordinates are the Real finite-window seal rows and the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ rows displayed in this chapter, so the preceding two exhaustion routes consume every available source, regularity, dyadic, and classifier coordinate before the local certificate is opened. Hence a quotient-real equality, selected host limit, ambient completeness row, or direct Real seal edge around StreamName or RegSeqRat has no coordinate from which to determine the terminal consumer. □

Theorem 38.660 (Real StreamName RegSeqRat common-refinement consumer lock)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer whose source is a common-refinement* $\mathsf{StreamName}^{\uparrow}$ *window and whose regularity witness is* $\mathsf{RegSeqRat}^{\uparrow}$ *factors through the common-refinement terminal packet, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, and finally the local* $\mathsf{RealUnaryStreamClassifier}$ *row. The route exports no quotient-real equality, host sequence equality, selected host limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, independent* $\mathsf{Real}^{\uparrow}$ *carrier row, or metric-completeness datum.*

Proof. Begin with Definition 38.657. It sends any accepted Real seal-facing read with a StreamName source and RegSeqRat readback through the finite StreamName window, the regular-rational handoff, and the local Real classifier-facing row.

Next apply Definition 38.658. The dyadic observation row is the only classifier input left after the StreamName and RegSeqRat surfaces have been read.

Use Definition 38.659 to lock this route at the terminal seal boundary. The local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ row can be read only after the source, regularity, dyadic, and classifier coordinates have been consumed.

Now apply Definition 126.403. The common-refinement terminal packet supplies the StreamName window accepted by the terminal Real seal read. Then Definition 126.404 keeps the RegSeqRat readback on that same common window, and Definition 126.405 identifies the resulting seal route with the common-refinement handoff.

The assembled route therefore reaches the Real classifier only through the common-refinement packet, the RegSeqRat readback, and the DyadicRatCore observation row. None of the cited packets contains a coordinate for quotient-real equality, host sequence equality, selected host limits, ambient Real completeness, independent Real carrier rows, or metric-completeness data, so no terminal consumer can read one from this handoff. □

Theorem 38.661 (Real StreamName RegSeqRat common-refinement seal uniqueness)**.** *Let C_0 and C_1 be accepted terminal* $\mathsf{Real}^\uparrow$ *seal-facing consumers. Suppose C_i is read through a* $\mathsf{StreamName}^\uparrow$ *common-refinement window W_i with a* $\mathsf{RegSeqRat}^\uparrow$ *readback row, a* $\mathsf{DyadicRatCore}^\uparrow$ *observation row, and a local* RealUnaryStreamClassifier *endpoint, all carried by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Suppose further that W_0 and W_1 both transport to one shared common-refinement packet W_*, and that after this transport their* $\mathsf{RegSeqRat}^\uparrow$ *readbacks and* $\mathsf{DyadicRatCore}^\uparrow$ *observation rows agree on W_*.*

Then the two local RealUnaryStreamClassifier *endpoints are* hsame*-equal at the shared refinement boundary, and C_0 and C_1 determine one terminal finite* $\mathsf{Real}^\uparrow$ *seal route. The assertion is only this common-refinement uniqueness statement. It excludes quotient-real equality, host sequence equality, selected host limits, ambient completeness, independent* $\mathsf{Real}^\uparrow$ *carrier rows, metric-completeness data, and standard-bridge rows.*

Proof. Apply Definition 38.663 to C_0 and C_1. This sends each terminal consumer through the same four types of displayed faces: its StreamName finite-window ledger, its RegSeqRat pointwise handoff, its DyadicRatCore observation row, and its local RealUnaryStreamClassifier endpoint with the local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ row.

Next use Definition 38.660. It forces both terminal reads to pass through the common-refinement terminal packet before the RegSeqRat readback, DyadicRatCore observation, and local Real classifier rows are consumed.

Transport the two factorizations along the displayed maps from W_0 and W_1 to W_*. By hypothesis the transported RegSeqRat readbacks and DyadicRatCore observation rows agree on this shared StreamName window, so the only remaining local Real boundary coordinate is the transported RealUnaryStreamClassifier endpoint.

Now apply Definition 38.664. With the StreamName common-refinement window, RegSeqRat readback, DyadicRatCore observation, and local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ support fixed on W_*, the two classifier endpoints are hsame-equal at the shared boundary.

Finally apply Definition 38.662. The hsame-identified classifier endpoints, together with the shared StreamName, RegSeqRat, DyadicRatCore, Cont, Pkg, and NameCert rows, determine one terminal finite $\mathsf{Real}^\uparrow$ seal route. The cited pullback separation also rules out any appeal to quotient-real equality, host sequence equality, selected limits, ambient completeness, independent Real carrier rows, metric-completeness data, or standard-bridge rows. □

Theorem 38.662 (Real StreamName RegSeqRat terminal pullback separation)**.** *Let C_0 and C_1 be terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing consumers whose sources are common-refinement* $\mathsf{StreamName}^{\uparrow}$ *windows and whose readbacks are* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. Suppose that C_0 and C_1 share the same common-refinement* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, and the same local* RealUnaryStreamClassifier *row, with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support carried by those four faces. Then C_0 and C_1 determine the same terminal finite* $\mathsf{Real}^{\uparrow}$ *seal route.*

Conversely, any equality of terminal finite $\mathsf{Real}^{\uparrow}$ *seal routes between such consumers factors back through those four faces: the common-refinement StreamName window, the RegSeqRat readback row, the DyadicRatCore observation row, and the local Real classifier row with its displayed history, continuity, package, and local-name support. The separation statement asserts only this terminal finite route. It does not identify quotient-real equalities, host sequence equalities, selected host limits, ambient completeness rows, independent Real carrier rows, metric-completeness data, or standard-bridge rows.*

Proof. Apply Definition 38.659 to each consumer. This sends both terminal reads through the four displayed faces before the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row is inspected, so the two consumers have no terminal seal coordinate outside the StreamName, RegSeqRat, DyadicRatCore, and local Real classifier surfaces.

Next use Definition 126.403. The shared common-refinement StreamName window supplies the same finite terminal packet for both consumers, including the visible history rows, transports, package provenance, and local names that feed the downstream Real seal.

Then apply Definition 126.404 and Definition 126.405. These two results lock the RegSeqRat readback, the DyadicRatCore observation row, and the Real seal route to the same common-refinement pullback. Hence equality at the terminal Real seal boundary is witnessed by equality on the four displayed faces, and every equality of such terminal finite seal routes pulls back to those faces.

Finally apply Definition 38.660. It rules out bypass coordinates for quotient-real equality, host sequence equality, selected host limits, ambient completeness, independent Real carrier rows, metric-completeness data, or standard-bridge rows. The terminal pullback therefore separates precisely the finite Real seal route and nothing outside the displayed handoff. □

Theorem 38.663 (StreamName RegSeqRat RealUp finite-window factorization)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *L10 finite-window seal read whose source is a* $\mathsf{StreamName}^{\uparrow}$ *window and whose regularity witness is a* $\mathsf{RegSeqRat}^{\uparrow}$ *row factors through exactly the four displayed faces*

StreamName finite-window ledger → RegSeqRat pointwise handoff → DyadicRatCore observation and tolerance ro

The factorization is finite and uses only the displayed BHist, hsame, Cont, Pkg, *and* NameCert *coordinates on those faces. It exports no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completion datum, off-ledger dyadic sample, or direct* $\mathsf{Real}^{\uparrow}$ *seal edge outside the four-face route.*

Proof. Start with the pointwise seal composition of Definition 38.657. It puts the accepted seal-facing read on the StreamName finite-window ledger, then on the RegSeqRat pointwise handoff, and only then on the local Real classifier-facing row.

Next apply the dyadic-window exhaustion theorem, Definition 38.658. This identifies the remaining classifier input with the DyadicRatCore observation and tolerance row and rules out any off-ledger dyadic sample before the Real classifier is read.

Use the terminal seal exhaustion theorem, Definition 38.659, to attach the local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ row after the source, regularity, dyadic, and classifier coordinates have been consumed. The terminal packet therefore has no unused coordinate from which to read quotient stream data, host sequence equality, selected limits, ambient completeness, or metric-completion data.

Finally compare this packet with the ordered L10 dependency lattice of Definition 38.442. The open-phase sibling route, Definition 38.530, reads the same StreamName, RegSeqRat, DyadicRatCore, and local Real faces and no other source surface. Repacking the three seal-exhaustion results along that route gives the claimed finite-window factorization. □

Theorem 38.664 (Real finite-window classifier endpoint determinacy)**.** *For every accepted* $\mathsf{Real}^\uparrow$ *finite-window seal read whose source is a* $\mathsf{StreamName}^\uparrow$ *window and whose regularity witness is a* $\mathsf{RegSeqRat}^\uparrow$ *row, the displayed classifier endpoint is unique up to* hsame *once the four faces of Definition 38.663 are fixed: the StreamName finite-window ledger, the RegSeqRat pointwise handoff, the DyadicRatCore observation and tolerance row, and the local* RealUnaryStreamClassifier *row with* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$.

Equivalently, any two endpoint reads using those same four displayed faces are related by the same finite BHist*,* Cont*,* Pkg*, and* NameCert *rows, and their endpoint witnesses are* hsame*-equal at the classifier boundary. No quotient-real equality, host sequence equality, selected host limit, ambient completeness row, or metric-completion datum can separate the two endpoints.*

Proof. Apply Definition 38.663 to the two endpoint reads. Both reads are sent to the same four displayed faces, so their finite StreamName ledger rows, RegSeqRat handoff rows, and local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ rows are compared inside the displayed BHist, Cont, and Pkg coordinates rather than through an ambient Real carrier.

Next use Definition 38.658. It fixes the dyadic classifier input by the shared DyadicRatCore observation and tolerance row, leaving no off-ledger dyadic sample that could distinguish one classifier endpoint from the other.

Finally use Definition 38.659 to lock the terminal seal boundary. Repacking the shared finite rows through the local RealUnaryStreamClassifier gives the same displayed endpoint up to hsame, and the terminal packet contains no coordinate for quotient-real equality, host sequence equality, selected limits, ambient completeness, or metric-completion data. □

Theorem 38.665 (Real terminal common-tail cocycle boundary)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^\uparrow$ *packet, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^\uparrow$ *schedule carried by P. Suppose each shifted pair (T_i, T_j) lands in a common-tail window packet and that the three pairwise packets restrict to one triple* $\mathsf{StreamName}^\uparrow$ *refinement W_* carrying the displayed* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{Real}^\uparrow$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Then the terminal $\mathsf{Real}^\uparrow$ *seal boundaries induced by the three pairwise common-tail packets agree exactly when those displayed rows satisfy Definition 301.190 on W_*. The boundary statement is only this finite terminal seal comparison. It excludes quotient streams, host sequence equality, selected limits, countable choice, ambient completeness, metric-completeness rows, and extra seal coordinates.*

Proof. Apply Definition 38.663 to each pairwise common-tail packet. Every terminal boundary read is thereby sent through the same four displayed faces: the StreamName finite-window ledger, the RegSeqRat pointwise handoff, the DyadicRatCore observation and tolerance row, and the local Real classifier row with its $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ provenance.

Next use Definition 38.664. Once those four faces agree on the triple refinement W_*, the Real classifier endpoint at each pairwise boundary is determined up to the displayed hsame row and

cannot be separated by host sequence equality, selected limits, ambient completeness, or metric-completion data.

Now apply Definition 38.662. Any agreement of terminal finite $\mathsf{Real}^{\uparrow}$ seal routes pulls back to the same common-refinement StreamName window, RegSeqRat readback row, DyadicRatCore observation row, and local Real classifier row, with the displayed Cont, Pkg, and NameCert support.

Finally invoke Definition 301.190. Its triple-overlap criterion is exactly the equality of the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows on W_*. Therefore the three terminal $\mathsf{Real}^{\uparrow}$ seal boundaries agree exactly in the cocycle case, and the absence of quotient streams, host sequence equality, selected limits, countable choice, ambient completeness, metric-completeness rows, and extra seal coordinates follows from the same displayed-row inventory. □

Theorem 38.666 (RealUp StreamName RegSeqRat source lattice)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *source-lattice read used by a Phase Real or Completion consumer factors through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row of chapter 301, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291 before the Real seal is public. The read exposes only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates carried by those sibling surfaces and the local* $\mathsf{Real}^{\uparrow}$ *seal package.*

In particular, the accepted source lattice contains no quotient real equality, selected representative, ambient completeness coordinate, or direct public Real seal edge bypassing the StreamName, RegSeqRat, and DyadicRatCore rows.

Proof. Start with an accepted $\mathsf{Real}^{\uparrow}$ source-lattice read. The local Real boundary only accepts the finite-window package already routed through the displayed stream, regular-rational, dyadic-tolerance, transport, continuation, provenance, and naming rows.

First read the source schedule through chapter 126. This fixes the visible finite-window row and its unary-history observations before any regular readback or Real seal row can be consumed.

Next read the regular-rational spine through chapter 301. The RegSeqRat row uses the same displayed window and supplies the regular readback surface on the common BHist, hsame, Cont, Pkg, and NameCert route.

Finally attach the tolerance ledger through chapter 291. The dyadic row is the only tolerance surface available to the Real seal package, so the public Real seal is reached only after the StreamName schedule, RegSeqRat readback, and DyadicRatCore ledger have all been displayed.

The composed source lattice therefore has no remaining coordinate from which to read quotient real equality, a selected representative, ambient completeness, or a direct seal edge around the sibling surfaces. □

Theorem 38.667 (Real StreamName RegSeqRat terminal budget stability)**.** *Let a terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer read its source as a common-refinement* $\mathsf{StreamName}^{\uparrow}$ *window, let its regularity witness be the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on that same window, and let its precision request be served through* $\mathsf{RealObservationBudget}^{\uparrow}$*. Then restricting the request to a smaller finite observation budget, or enlarging it to a larger finite observation budget, preserves the same public finite source surface:*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The budget operation may change only which displayed finite observations are queried. It cannot add quotient-real equality, host sequence equality, a selected host limit, ambient Real completeness, an independent Real carrier, or metric-completeness data.

Proof. Start with the common-refinement source. The consumer lock Definition 38.660 fixes the terminal read on the shared StreamName, RegSeqRat, DyadicRatCore, and Real seal rows before the seal-facing consumer can inspect the packet.

Now attach the observation-budget route. The StreamName Real observation-budget factorization Definition 126.363 routes every accepted $\mathsf{RealObservationBudget}^{\uparrow}$ precision request through the same visible finite-window, dyadic, regular readback, and Real seal chain.

Finally use the carrier and non-escape rows of chapter 611. They make budget restriction or enlargement a change of displayed finite request rows, not a change of source carrier. Since the cited routes expose only the finite BEDC rows listed in the statement, none of the excluded analytic or host-level objects can enter the terminal seal-facing surface. □

Theorem 38.668 (Real StreamName RegSeqRat Dyadic source exhaustion)**.** *Every Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *source read of the accepted dependency lattice factors through exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the local* $\mathsf{Real}^{\uparrow}$ *seal row. The factorization exposes only the finite packet coordinates already carried by those four surfaces and their shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes.*

Consequently the completion-facing source route contains no quotient stream, selected host limit, ambient completeness theorem, or hidden endpoint equality.

Proof. Begin with the accepted source-lattice route of Definition 38.666. It already decomposes the Real source read into the StreamName finite-window schedule, the RegSeqRat regular readback row, and the DyadicRatCore tolerance ledger before the local Real seal is public.

Next apply the terminal budget stability statement Definition 38.667. Changing the finite observation budget changes only which displayed source observations are queried; it does not add a carrier row outside the StreamName, RegSeqRat, DyadicRatCore, and Real seal packet.

It remains to exhaust the packet coordinates. The nearby source-route clauses give the visible BHist, hsame, Cont, Pkg, and NameCert rows for each of the four surfaces, while the Real seal row is reached only after those sibling rows have been displayed.

Thus every completion-facing source read is a coordinate projection of that finite packet. Since the packet has no coordinate for quotient streams, selected host limits, ambient completeness theorems, or hidden endpoint equalities, none of those objects can be read from the accepted source route. □

Theorem 38.669 (Real L10 source-lattice four-face equivalence)**.** *For a Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *source read, acceptance by the current source lattice is equivalent to exposing exactly the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with their shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. In this equivalence the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the dyadic tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *face is the finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the regular-rational readback row, and the local* $\mathsf{Real}^{\uparrow}$ *face is the seal row reached after those sibling faces.*

Both directions exclude quotient streams, selected limits, countable choice, metric completeness, aggregate formal status, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness as source-surface data.*

Proof. For the forward direction, start with an accepted Phase Real and Completion $\mathsf{Real}^{\uparrow}$ source read. The source-lattice theorem Definition 38.666 places the read on the StreamName finite-window schedule, the RegSeqRat readback row, and the DyadicRatCore tolerance ledger before the local Real seal is public.

Next apply terminal budget stability, Definition 38.667. It says that changing the finite observation request changes only displayed finite rows, so no additional source carrier or classifier can appear at the seal-facing boundary. The dyadic source exhaustion theorem Definition 38.668 then identifies the accepted read with exactly the four faces and shared BHist, hsame, Cont, Pkg, and NameCert support named in the statement.

For the reverse direction, assume exactly those four local faces and shared support are exposed. Apply Definition 291.83 to read the dyadic scale-window ledger as the same four-face pullback through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ seal. Then apply Definition 301.175 to put the regular-rational exit consumer in the corresponding four-face normal form.

The local source-lattice route Definition 38.666 repacks that pullback and normal-form data as the accepted Real source read. Since the four exposed faces provide all displayed BHist, hsame, Cont, Pkg, and NameCert coordinates, the reconstructed read has the same source surface as the accepted Phase Real and Completion lattice.

The exclusion clause follows in both directions from the cited non-escape statements. None of the four-face pullback, the RegSeqRat normal form, or the local Real source-lattice route contains a coordinate for quotient streams, selected limits, countable choice, metric completeness, aggregate formal status, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.670 (Real dyadic StreamName RegSeqRat route)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *open-phase seal read whose source is presented to an L10 consumer as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, a* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, and the local* $\mathsf{Real}^{\uparrow}$ *seal factors through one ordered finite route:*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The route is transported only by the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of those four surfaces.*

No accepted read on this route exports quotient real equality, host limits, countable choice, ambient completeness, an independent host completion surface, or a direct local seal edge bypassing the dyadic, stream, and regular-rational rows.

Lean4 ✓ `BEDC.Derived.RealUp.RealDyadicStreamRegseqRoute`

Proof. Begin with the L10 interface totality theorem Definition 38.444. It already places every accepted Phase Real or Completion interface read on exactly the four visible faces of the Real L10 dependency lattice: DyadicRatCore observations, StreamName finite-window rows, RegSeqRat regular-rational rows, and the local Real seal.

Next apply the source lattice theorem Definition 38.666. It identifies the StreamName, RegSeqRat, and DyadicRatCore handoff as one source lattice before the Real seal is public, with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates available for transport and packaging.

Finally use terminal budget stability, Definition 38.667. Changing the finite observation budget changes only which displayed finite rows are queried, not the carrier of the route. Thus the open-phase seal read remains on the single ordered finite route and has no coordinate for quotient real equality, host limits, countable choice, ambient completeness, host completion, or a direct seal bypass. □

Theorem 38.671 (Real dyadic StreamName RegSeqRat budget exactness)**.** *Let two accepted current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *seal reads be presented on the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Suppose their finite observation budgets expose the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and structural* BHist, hsame, Cont, Pkg, *and* NameCert *rows on their overlap. Then replacing one budget by the other, or refining either budget, changes only the displayed finite request rows. The accepted read remains the same four-face route from dyadic observations through finite stream windows and regular-rational readback to the local Real seal.*

In particular, this budget operation is not a quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness principle, metric-completeness row, or direct $\mathsf{Real}^{\uparrow}$ *seal bypass.*

Proof. Apply Definition 38.670. Both accepted reads are already placed on the ordered finite route from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, and the only transport rows available on the route are the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 38.444. It says that the current Phase Real and Completion interface has exactly the four visible faces listed in the statement. Since the two budgets agree on those faces and on the structural rows over their overlap, every common exposed request row projects to the same place in the L10 lattice.

Finally use Definition 38.667. Refining or replacing a finite observation budget changes the finite row display but does not add a new carrier, classifier, package row, or naming certificate. Row projection therefore preserves the same four-face acceptance route.

The exclusion clause is the non-escape part of the same three inputs. None of the displayed route, interface-totality, or terminal-budget rows contains a coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric completeness, or a direct Real seal bypass, so a budget change cannot certify any of those routes. □

Theorem 38.672 (Real dyadic StreamName RegSeqRat seal equivalence)**.** *For a current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *seal read, agreement of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows, and local* $\mathsf{Real}^{\uparrow}$ *seal row, together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *coordinates, is equivalent to acceptance of the ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

This equivalence is internal to the finite dyadic-stream-regseq seal route. It does not assert quotient stream equality, host real equality, a selected limit, countable choice, metric completeness, ambient completion coordinates, or any direct seal edge outside the four displayed faces.

Proof. For the forward direction, start from agreement of the displayed dyadic, finite-window, regular-readback, and local seal rows, with the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates fixed. Apply Definition 38.444 to restrict the current Phase Real and Completion read to the four visible L10 faces.

Next use Definition 38.666. It identifies the DyadicRatCore, StreamName, and RegSeqRat rows as the source lattice available before the local Real seal is accepted. The row agreement therefore has exactly the finite source data needed by the seal route.

Apply Definition 38.670. The matched source and seal rows are repacked as the ordered route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

For the reverse direction, assume acceptance of the ordered four-face route. The route theorem exposes precisely the dyadic observations, finite stream windows, regular-rational readback rows, local seal row, and structural transport and package coordinates named in the statement.

Finally apply Definition 38.671. It says that replacing or refining the finite budget changes only the displayed request rows, so agreement on those rows is exactly the classifier agreement carried by the accepted finite dyadic-stream-regseq seal route.

The exclusion clause follows because the four cited inputs have no coordinate for quotient stream equality, host real equality, selected limits, countable choice, metric completeness, ambient completion, or a direct $\mathsf{Real}^{\uparrow}$ seal bypass. □

Theorem 38.673 (Real Cauchy window transducer seal route)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and then* $\mathsf{Real}^{\uparrow}$. *The transducer rows* $S, D, W, R, E, L, H, C, P, N$ *provide only finite route replay, and they provide no direct seal bypass.*

Proof. Apply Definition 38.670. It places the relevant Real-facing read on the ordered route from DyadicRatCore observations through StreamName finite windows and RegSeqRat regular readback to the Real seal.

Use Definition 1020.4. In an accepted Cauchy window transducer packet, the seal row E is public only after the RegSeqRat readback row R has been displayed.

Finally apply Definition 1020.5. The remaining rows S, D, W, L, H, C, P, N replay the finite route and bind transport, provenance, and local naming. They do not add a quotient Real equality, host limit, countable-choice row, ambient completeness row, or direct edge to the seal. □

Theorem 38.674 (Real StreamName RegSeqRat sibling nonorphan route)**.** *The Real seal route consumes the sibling chapter surfaces chapter 291, chapter 126, chapter 301, and chapter 1020 before the local* $\mathsf{Real}^{\uparrow}$ *seal is public. Thus the local Real route is a horizontal L10 dependency through dyadic observations, finite stream windows, regular-rational readback, and the Cauchy window transducer, not a self-reference and not a host-completion shortcut.*

Proof. Apply Definition 38.670. It places the Real-facing route on the ordered dyadic, StreamName, RegSeqRat, and Real seal surfaces, with only displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates available for transport.

Then apply Definition 38.673. The Cauchy window transducer seal read factors through the same dyadic, stream, and regular-rational route before the local Real seal is exposed.

These two routes cite sibling chapter surfaces rather than the local Real chapter as their source. The dependency is therefore horizontal in the L10 lattice and cannot be replaced by a Real self-reference, selected host limit, ambient completion object, or direct seal bypass. □

Theorem 38.675 (Real terminal four-face open-phase exit)**.** *Every current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *seal read is accepted exactly when its visible rows factor through the ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

displayed by the Real L10 dependency lattice. The visible coordinates of the exit read are only the displayed BHist *packet, the displayed* hsame *transports, the displayed* Cont *routes, the displayed* Pkg *provenance rows, and the displayed* $\mathsf{NameCert}$ *rows on those four faces.*

Thus no terminal exit read is accepted by quotient stream equality, countable choice, host limits, metric completeness, ambient completion, host real equality, or any direct edge to the $\mathsf{Real}^{\uparrow}$ *seal outside the displayed four-face route.*

Proof. Suppose first that a current Phase Real or Completion terminal $\mathsf{Real}^{\uparrow}$ seal read is accepted. Apply Definition 38.670. It places the accepted seal read on the ordered dyadic, StreamName,

RegSeqRat, and Real route and restricts transport to the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next apply Definition 38.674. The route is therefore sourced by the sibling DyadicRatCore, StreamName, RegSeqRat, and Cauchy-window-transducer surfaces before the local Real seal is public. This rules out a local self-reference or a host-completion shortcut as the source of acceptance.

Now read the same terminal packet through the Real L10 dependency lattice, using Definition 38.442 and Definition 38.444. The lattice gives exactly the four displayed faces, and interface totality says that current Phase Real and Completion reads have no fifth visible coordinate.

Conversely, suppose that the visible rows of a terminal read factor through that ordered four-face route with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates. The route theorem supplies the accepted Real seal handoff, and the sibling nonorphan route supplies the horizontal source through the dyadic, stream, regular-rational, and transducer surfaces. The Real L10 interface totality theorem then repacks that handoff as the current Phase Real and Completion terminal surface.

The exclusion clause follows from the same three inputs. None of the route, sibling, or L10 lattice rows exports quotient stream equality, countable choice, host limits, metric completeness, ambient completion, host real equality, or a direct seal bypass, so none of those data can certify a terminal exit read. □

Theorem 38.676 (Real current-phase four-face exit certificate)**.** *A current Phase Real and Completion public read of the* $\mathsf{Real}^{\uparrow}$ *seal is accepted if and only if its visible rows factor through the four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

using exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows on that route.*

Any purported completion-facing read that uses quotient stream equality, countable choice, host limits, metric completeness, ambient completion, host real equality, or a direct $\mathsf{Real}^{\uparrow}$ *seal bypass is rejected by the current Phase Real and Completion surface.*

Proof. For the forward direction, apply Definition 38.675. It sends every accepted terminal $\mathsf{Real}^{\uparrow}$ seal read to the ordered four-face route and restricts the visible coordinates to the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

For the converse direction, use the same theorem in the reverse direction: visible rows on the ordered four-face route, with no extra coordinate, repack as the current Phase Real and Completion terminal surface.

Now apply Definition 38.670. It identifies the route as the finite handoff from DyadicRatCore observations through StreamName finite windows and RegSeqRat regular readback to the local $\mathsf{Real}^{\uparrow}$ seal. Apply Definition 38.674 to record that the handoff is horizontal through the sibling surfaces, not a local Real self-reference and not a host-completion shortcut.

Finally use Definition 38.444. The Real L10 interface has no fifth public coordinate beyond the displayed faces. Hence a purported read using quotient stream equality, countable choice, host limits, metric completeness, ambient completion, host real equality, or a direct Real seal bypass has no accepting coordinate on the current Phase Real and Completion surface. □

Theorem 38.677 (Real four-face terminal cover pullback)**.** *Let a finite family of current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *seal reads be displayed over one terminal cover. Suppose that on every pairwise overlap the visible rows agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational classifier rows, and the local*

$\mathsf{Real}^{\uparrow}$ *seal rows, together with the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *coordinates. Assume also that on every triple overlap the two restriction routes from each local read to the common section coincide:*

$$(\mathcal{R}_i|_{ij})|_{ijk} = (\mathcal{R}_i|_{ik})|_{ijk}, \qquad (\mathcal{R}_i|_{ij})|_{ijk} = (\mathcal{R}_j|_{ij})|_{ijk}.$$

Then the whole family pulls back to one ordered terminal four-face route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The resulting terminal surface has no coordinate for quotient stream equality, host real equality, countable choice, ambient completion, metric-completeness rows, or a direct $\mathsf{Real}^{\uparrow}$ *seal bypass.*

Proof. Apply Definition 38.676 to each member of the finite terminal cover. Each accepted seal read is therefore already presented by the ordered dyadic, stream, regular-rational, and local Real faces, with only the displayed structural rows available.

The pairwise overlap agreement and the triple-overlap route coherence are the cocycle data required by Definition 38.821. Applying that theorem pastes the finite family to a unique public $\mathsf{Real}^{\uparrow}$ terminal-seal read whose restrictions are the displayed local reads. Reading this pasted seal through the preceding normal-form certificate presents it as the ordered terminal four-face route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Finally apply Definition 38.444. The dependency lattice supplies no fifth route beside the four displayed faces, so the cover repackages as one terminal four-face surface and excludes quotient stream equality, host real equality, countable choice, ambient completion, metric-completeness rows, and direct seal bypasses. □

Theorem 38.678 (Real four-face terminal cover overlap compatibility)**.** *Let* C_0 *and* C_1 *be two finite families of current Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *seal reads. Suppose each family satisfies the hypotheses of Definition 38.677. Suppose further that on a shared subcover the two families agree on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support coordinates. Then the two pulled-back terminal four-face routes have the same public* $\mathsf{Real}^{\uparrow}$ *seal classifier surface on that shared subcover.*

This compatibility is only a finite four-face statement. It does not identify quotient streams, host reals, selected limits, countable-choice witnesses, ambient completions, metric-completeness witnesses, or any direct bypass of the local $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Apply Definition 38.677 to C_0 and C_1. This presents both terminal covers over ordered routes

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with only the displayed structural rows available.

On the shared subcover, use Definition 126.108. Agreement of the $\mathsf{DyadicRatCore}^{\uparrow}$ observation rows and the $\mathsf{StreamName}^{\uparrow}$ finite-window rows identifies the same finite-window source for both pulled-back routes.

Next transport the classifier surface through the shared rows. Apply Definition 38.672 to pass between the dyadic, stream, regular-rational, and local Real seal presentations, and apply Definition 38.671 to keep the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates exact on the same subcover.

Finally apply Definition 38.444. Since the current Phase Real and Completion interface has no fifth public coordinate, there is no additional route through quotient stream equality, host real equality, selected limits, countable choice, ambient completion, metric-completeness witnesses, or a direct Real seal bypass. The transported classifier agreement is therefore exactly equality of the public $\mathsf{Real}^{\uparrow}$ seal classifier surface on the shared subcover. □

Theorem 38.679 (Real four-face terminal cover classifier finality)**.** *Let C_0 be a finite terminal cover of current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *seal reads satisfying the hypotheses of Definition 38.677. Let C_1 be a second finite terminal presentation of the same terminal read support. Suppose that C_1 also satisfies those hypotheses, and that C_0 and C_1 agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support coordinates. Then C_0 and C_1 determine one and the same public* $\mathsf{Real}^{\uparrow}$ *seal classifier surface.*

This finality is only over the terminal four-face cover

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It excludes quotient stream equality, host real equality, selected limits, countable choice, ambient completion, metric-completeness rows, and any direct $\mathsf{Real}^{\uparrow}$ *seal bypass from the data that can affect the public classifier surface.*

Proof. Apply Definition 38.677 separately to C_0 and C_1. The two presentations therefore pull back to terminal four-face routes with the same displayed dyadic, stream, regular-rational, local Real, and structural support rows.

On every common part of the cover, use Definition 38.678. The agreement of the DyadicRatCore observations, StreamName windows, RegSeqRat classifier and readback rows, local Real seal rows, and displayed support coordinates identifies the public $\mathsf{Real}^{\uparrow}$ seal classifier surface on each overlap.

Next pass through the four named faces. Apply Definition 38.672 to identify the classifier surface obtained from the dyadic, stream, regular-rational, and local Real presentations.

Finally apply Definition 38.444. The current Phase Real and Completion dependency lattice exposes no fifth public coordinate, so quotient stream equality, host real equality, selected limits, countable choice, ambient completion, metric-completeness rows, and direct Real seal bypasses have no classifier input. The overlap-compatible four-face surface is therefore the unique public $\mathsf{Real}^{\uparrow}$ seal classifier surface determined by the two presentations. □

Theorem 38.680 (Real four-face terminal subcover localization)**.** *Let C be a finite terminal cover of current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *seal reads satisfying the hypotheses of Definition 38.677. Let D be any displayed finite subcover whose local rows, overlap rows, and triple-overlap restriction routes are inherited from C. Then D pulls back to the same ordered terminal four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Its public $\mathsf{Real}^{\uparrow}$ *seal classifier surface is the restriction of the classifier surface determined by C to the displayed subcover rows. In particular, the localized cover still has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient completion, metric-completeness rows, or a direct* $\mathsf{Real}^{\uparrow}$ *seal bypass.*

Proof. Apply Definition 38.677 to the ambient cover C. This presents every accepted row of C on the ordered dyadic, stream, regular-rational, and local Real route, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support coordinates.

Because D inherits its rows and restriction routes from C, its pairwise overlap agreement and triple-overlap coherence are just the corresponding ambient equalities restricted to the chosen finite subcover. Applying Definition 38.677 to those inherited data pastes D to the same ordered route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Now use Definition 38.678 with the ambient cover and the displayed subcover. Their rows agree on the subcover by inheritance, so the public $\mathsf{Real}^{\uparrow}$ seal classifier surface for D is exactly the restriction of the classifier surface obtained from C.

Finally apply Definition 38.679 and Definition 38.444. Finality makes the restricted classifier independent of the chosen terminal presentation, and interface totality rules out any fifth coordinate. Hence quotient stream equality, host real equality, selected limits, countable choice, ambient completion, metric-completeness rows, and direct Real seal bypasses remain outside the localized terminal surface. □

Theorem 38.681 (Real dyadic tail-radius ledger dependency)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer that uses the dyadic tail-radius route first factors through chapter 757 before the local* $\mathsf{Real}^{\uparrow}$ *seal row is consumed. When the same consumer is completion-facing and reads a witness ledger, it also factors through chapter 902 before the completion-facing* $\mathsf{Real}^{\uparrow}$ *seal row is consumed.*

Equivalently, the admitted finite route is confined to the displayed chain

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicTailRadiusLedger}^{\uparrow} \to \mathsf{CauchyWitnessLedger}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row present only for completion-facing witness reads. The route exports no quotient stream equality, host real equality, selected external limit, countable choice, or ambient completeness principle.*

Proof. Begin with the Real dyadic StreamName RegSeqRat route of Definition 38.670. It places every accepted open-phase seal read whose source is presented through dyadic observations, finite StreamName windows, and regular-rational readback on the ordered route $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$.

Next apply the DyadicTailRadiusLedger finite route package from Definition 757.14. Its carrier inserts the finite precision request, tail-window ledger, StreamName windows, dyadic readback, RegSeqRat handoff, and Real seal row into one ledger route. Thus a seal-facing consumer using the dyadic tail-radius route cannot reach the Real seal before that ledger has been consumed.

For completion-facing witness reads, apply the CauchyWitnessLedger budget factorization and Real completion handoff, Definition 902.5 and Definition 902.9. These force the consumer through the finite witness ledger before the completion-seal classifier and the Real-facing completion read are available.

Finally combine the non-escape boundaries of the three routes: Definition 38.670, Definition 757.7, and Definition 902.18. Since none of the composed carriers has a coordinate for quotient stream equality, host real equality, selected external limits, countable choice, or ambient completeness, the composed Real dependency route exports none of them. □

Theorem 38.682 (Real dyadic tail-radius seal determinacy)**.** *Let two accepted* $\mathsf{Real}^{\uparrow}$ *seal-facing consumers use the dyadic tail-radius route. If they expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, the same* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *route, the same* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *witness row whenever the read is completion-facing, and the same displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows, then they determine the same finite* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Conversely, every admitted $\mathsf{Real}^{\uparrow}$ *seal boundary that uses this dyadic tail-radius route is read through exactly those rows: dyadic endpoint and radius data, finite StreamName windows, regular-rational handoff, tail-radius ledger route, the completion-facing witness row when present, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support. The statement does not assert* $\mathsf{Real}^{\uparrow}$ *completeness, quotient-stream equality, host real equality, selected external limits, countable choice, or metric-completion equality.*

Proof. First apply Definition 38.681. It fixes the ordered dependency route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicTailRadiusLedger}^{\uparrow} \to \mathsf{CauchyWitnessLedger}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the witness-ledger row present only for completion-facing reads. Thus two accepted consumers with the same displayed rows have the same ordered route to the Real seal.

Next use Definition 38.670. The initial segment of the route is already determined by the dyadic observations, finite StreamName windows, RegSeqRat regular readback, and the structural BHist, hsame, Cont, Pkg, and NameCert rows. Hence the two consumers cannot differ before the tail-radius ledger is reached.

Now apply Definition 757.14 together with Definition 757.7. The package fixes the finite tail-radius packet on the route, and non-escape prevents an additional quotient, host-equality, selected-limit, unbounded-search, independent-ledger, or ambient-completeness coordinate from entering the seal boundary.

For completion-facing reads, apply Definition 902.5, Definition 902.9, and Definition 902.18. These place the witness row behind the finite budget route, hand it to the Real completion-facing consumer through the displayed support rows, and exclude quotient streams, host Real equality, selected external limits, countable choice, and ambient completeness from the read.

The forward determinacy follows because every row on the seal-facing route has been fixed componentwise. The converse follows by reading any admitted boundary through the same factorization and non-escape results: there is no public coordinate outside the listed finite dyadic, StreamName, RegSeqRat, tail-radius, witness-ledger, and structural support rows, so the listed rows are exactly the finite $\mathsf{Real}^{\uparrow}$ seal boundary for this route. □

Theorem 38.683 (RealUp RegSeqRat StreamName sibling route)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal read that uses the regular-Cauchy presentation first consumes the sibling surfaces of chapter 126 and chapter 301, together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance surface of chapter 291, before the RealUp name is public. The route is transported only through the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by those sibling surfaces and the local* $\mathsf{Real}^{\uparrow}$ *seal packet.*

The public seal read exports no quotient-real equality, selected host limit, ambient completeness coordinate, hidden dyadic tolerance sample, or direct $\mathsf{Real}^{\uparrow}$ *name edge bypassing StreamName and RegSeqRat.*

Proof. Begin with the Real StreamName-RegSeqRat crosslink of Definition 38.657. It routes any seal-facing read with a StreamName source and a RegSeqRat readback through the visible StreamName finite-window ledger and the RegSeqRat pointwise handoff before the local real classifier is consumed.

Next use Definition 38.666. This places the StreamName and RegSeqRat rows inside a common source lattice and attaches the DyadicRatCore tolerance ledger before the Real seal becomes public.

Finally apply Definition 38.670. The ordered route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

is the only open-phase seal route available to the L10 consumer. Since the cited packets expose only the displayed finite carrier, classifier, transport, packaging, and certificate rows, the public RealUp name has no coordinate for quotient-real equality, selected host limits, ambient completeness, hidden dyadic samples, or a direct seal bypass. □

Theorem 38.684 (Real RegSeqRat StreamName lattice route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *finite-window seal route used by Phase Real and Completion consumes a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through the sibling chapters chapter 126, chapter 301, and chapter 291 before the local* $\mathsf{Real}^{\uparrow}$ *seal row is public. The route is a horizontal lattice edge through those sibling surfaces and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a quotient stream, selected limit, host sequence equality, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 38.530. It already places every public $\mathsf{Real}^{\uparrow}$ seal read in the open phase after the StreamName, RegSeqRat, and DyadicRatCore sibling surfaces.

Next apply Definition 38.666. The lattice identifies the finite StreamName schedule, regular-rational readback, and dyadic tolerance ledger as the source rows that must be consumed before the local Real seal is exposed.

Use Definition 38.670 to align those rows into one route. The dyadic observations, StreamName finite window, RegSeqRat readback, and Real seal remain a single finite BEDC path rather than independent analytic witnesses.

Finally apply Definition 38.674. It confirms that the Real route is read through the same sibling chapters and exports no quotient stream, selected host limit, host sequence equality, ambient completeness row, or direct seal bypass. □

Definition 38.685 (Real name seal carrier)**.** *A* $\mathsf{Real}^{\uparrow}$ *name-seal carrier is a finite* BHist *packet*

$$R = (D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row recording the endpoint, radius, and finite tolerance data available to the read;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that exposes the requested observations;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row that records the pointwise regular-rational readback;*
4. *L is the finite ledger tying the selected windows, dyadic observations, radius rows, and regularity checks to one request surface;*
5. *E is the local* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, S, Q, L have been displayed;*
6. *H records the componentwise* hsame *transports for D, S, Q, L, E, C, P, N;*
7. *C records the displayed* Cont *routes from the finite-window schedule through the dyadic and regularity rows to the seal row;*
8. *P records* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row for this name-seal packet.*

The carrier has no coordinate for a quotient stream, a selected host limit, host real equality, host sequence equality, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.RealUp.RealNameSealCarrier`

Theorem 38.686 (Real name seal source factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted through the carrier of Definition 38.685 factors through the displayed source route*

$$S, D, Q, L \longmapsto E$$

together with the visible BHist, hsame, Cont, Pkg, *and* NameCert *rows. Equivalently, the seal may be read only after the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, and the finite ledger have all been consumed through the same packet.*

Lean4 ✓ `BEDC.Derived.RealUp.RealNameSeal source factorization`

Proof. Project the accepted packet through Definition 38.685. The only seal coordinate is E, and the only source rows that lead to it are S, D, Q, L with the transport, continuation, package, and naming rows H, C, P, N.

The route is the local RealUp form of Definition 38.684. That theorem places the Phase Real and Completion seal path behind the StreamName, RegSeqRat, and DyadicRatCore sibling surfaces before the local seal is public.

The RegularCauchyName seal results provide the readback discipline used by the same boundary. Apply Definition 371.15, Definition 371.18, and Definition 371.24: a real-seal consumer is downstream of the displayed finite window, dyadic observation, regularity ledger, package provenance, and local NameCert rows. Repacking those rows as $D, S, Q, L, E, H, C, P, N$ gives the stated factorization. □

Theorem 38.687 (Real name seal four-face coverage)**.** *For every accepted* $\mathsf{Real}^{\uparrow}$ *name-seal packet*

$$R = (D, S, Q, L, E, H, C, P, N)$$

of Definition 38.685, all public seal reads are exhausted by exactly the four finite faces

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The exhaustive route consists of the following displayed rows:

1. *the* BHist *packet rows* D, S, Q, L, E*;*
2. *the componentwise* hsame *transports in* H*;*
3. *the* Cont *routes in* C *from finite windows through dyadic observations and regular-rational rows to the local seal;*
4. *the* Pkg *provenance row* P *over the four faces and the structural rows;*
5. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row* N *for this packet.*

No public read through R *has a coordinate for quotient stream equality, selected host limits, host real equality, host sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, choice rows, or standard bridge rows.*

Proof. Project the packet through Definition 38.685. The carrier exposes exactly the dyadic observation row D, the finite-window row S, the regular-rational row Q, the finite ledger L, the local seal row E, and the structural rows H, C, P, N.

Apply Definition 38.686. Every public seal read must pass through $S, D, Q, L \longmapsto E$, with only the visible BHist, hsame, Cont, Pkg, and NameCert support rows.

The lattice and consumer-scope results identify the four faces of that route. Use Definition 38.684 and Definition 371.15 to place the accepted read behind the StreamName, RegSeqRat, DyadicRatCore, and local Real surfaces.

The face rows are the existing handoffs: Definition 126.301, Definition 291.18, Definition 301.22, and Definition 38.135. They account for the finite window, dyadic observation, regular-rational readback, and local seal faces.

Since Definition 38.685 has no fifth analytic coordinate, the listed four faces exhaust the accepted seal route. The excluded quotient, host-limit, host-equality, completeness, metric-completeness, choice, and standard-bridge rows are therefore not public reads of R. □

Theorem 38.688 (Real StreamName-RegSeqRat seal route)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *name-seal packet*

$$R = (D, S, Q, L, E, H, C, P, N)$$

of Definition 38.685 consumes the $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row* Q*, and the ledger* L *before exposing the local seal row* E*. The resulting Phase Real and Completion exit route is the four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

carried only by the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

The route exports no quotient stream equality, selected host limit, host real equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, metric-completeness datum, or choice row.*

Proof. Start with the carrier definition and the source factorization theorem, Definition 38.685 and Definition 38.686. They identify D, S, Q, L as the only source rows through which the accepted packet can reach E, with H, C, P, N as structural transport, replay, provenance, and naming rows.

Next apply the Real L10 sibling lattice summary. The ordered lattice of Definition 38.442, together with Definition 38.443 and Definition 38.444, places Phase Real and Completion consumers on the same DyadicRatCore, StreamName, RegSeqRat, and Real faces.

The four public faces provide the component rows. The StreamName face is the public finite-window interface of Definition 126.301; the DyadicRatCore face is the real-phase source coverage of Definition 291.18; the RegSeqRat face is the real-seal handoff of Definition 301.22; and the local Real face is finite-prefix exactness through Definition 38.135.

Composing those rows with the source factorization repacks the accepted read as the displayed route to E. Since each cited interface exposes only finite packet rows and the same structural support, no quotient stream equality, selected host limit, host real equality, host sequence equality, ambient Real completeness principle, metric-completeness datum, or choice row is exported. □

Theorem 38.689 (Real name seal NameCert obligations)**.** *For the carrier of Definition 38.685, the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite packet* $R = (D, S, Q, L, E, H, C, P, N)$*;*
2. *classifier transport by the displayed* hsame *rows over the dyadic observation, finite-window schedule, regularity row, finite ledger, and seal row;*
3. *stability of the same route under the displayed* Cont *rows;*
4. *ledger exhaustion by* D, S, Q, L *before the seal row* E *is public;*
5. *source factorization through Definition 38.686.*

Proof. Carrier admission is projection from the displayed tuple. The packet lists every public row available to the local RealUp name-seal read, so the carrier and classifier have no hidden coordinate beyond $D, S, Q, L, E, H, C, P, N$.

Transport and stability are componentwise. The H row moves the displayed source and seal rows, while the C row replays the same finite route from the StreamName schedule through the dyadic observation, RegSeqRat readback, ledger, and seal.

Ledger exhaustion is supplied by Definition 38.686 together with the RegularCauchyName exhaustion theorems cited there. Thus the local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ obligations are exactly carrier, classifier, stability, ledger, and source-factorization obligations. □

Theorem 38.690 (Real name seal classifier stability)**.** *Let*

$$R = (D, S, Q, L, E, H, C, P, N) \qquad \textit{and} \qquad R' = (D', S', Q', L', E', H', C', P', N')$$

be accepted $\mathsf{Real}^\uparrow$ *name-seal packets for the carrier of Definition 38.685. Suppose the displayed source rows* D, S, Q, L *and* D', S', Q', L' *and the seal rows* E, E' *are related componentwise by the* hsame *transports recorded in* H *and* H'*. Suppose also that the displayed* Cont *routes in* C *and* C' *preserve the same source reads from the finite-window schedule through the dyadic observation row, the regularity row, and the finite ledger. Then the local* $\mathsf{Real}^\uparrow$ *seal classifier presents the same public seal row for* R *and* R' *after passing through the* $\mathsf{DyadicRatCore}^\uparrow$ *observation, the* $\mathsf{StreamName}^\uparrow$ *schedule, the* $\mathsf{RegSeqRat}^\uparrow$ *regularity row, the finite ledger, and the local* $\mathsf{Real}^\uparrow$ *seal.*

This classifier equality has no coordinate for quotient streams, selected host limits, host equality, ambient completeness, metric-completeness data, or choice rows.

Proof. Project both accepted packets through Definition 38.685. The only analytic coordinates available to the classifier are the dyadic observation rows D, D', the finite-window schedules S, S', the regularity rows Q, Q', the ledgers L, L', and the seal rows E, E'; the rows $H, H', C, C', P, P', N, N'$ supply transport, replay, provenance, and naming data.

Apply Definition 38.686 to both packets. Each public seal read factors through the displayed route from the StreamName window, DyadicRatCore observation, RegSeqRat regularity row, and finite ledger before reaching the local seal. Apply Definition 38.689 to read the same route as the NameCert classifier-transport and route-stability surface.

Now compose the sibling faces of Definition 38.688. The StreamName face is fixed by Definition 126.301; the DyadicRatCore face is fixed by Definition 291.18; the RegSeqRat face is fixed by Definition 301.22; and the local Real face is fixed by Definition 38.135. The componentwise hsame rows identify the corresponding displayed rows, and the Cont rows replay the same source reads through those faces.

The two accepted packets therefore present the same finite public route to the seal classifier, hence the same public seal row. Since the carrier and the cited faces expose only those finite BEDC rows, no quotient stream, selected host limit, host equality, ambient completeness theorem, metric-completeness datum, or choice row can enter the classifier comparison. □

Theorem 38.691 (Real name seal classifier transitivity)**.** *Let*

$$R = (D, S, Q, L, E, H, C, P, N), \qquad R' = (D', S', Q', L', E', H', C', P', N'), \qquad R'' = (D'', S'', Q'', L'', E'', H'', C'',$$

be accepted $\mathsf{Real}^\uparrow$ *name-seal packets for the carrier of Definition 38.685. Suppose* R *and* R' *classify by the displayed Real name-seal route of Definition 38.690, and suppose* R' *and* R'' *classify by the same displayed route. Then* R *and* R'' *present the same public* $\mathsf{Real}^\uparrow$ *seal row through the common* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*, finite-ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

This composed classifier comparison exports no quotient stream, selected host limit, host equality, ambient completeness principle, metric-completeness data, or choice row.

Proof. Project the three accepted packets through Definition 38.685. The projection exposes only the displayed source rows D, S, Q, L, D', S', Q', L', D'', S'', Q'', L'', the seal rows E, E', E'', and the corresponding structural rows H, H', H'', C, C', C'', P, P', P'', and N, N', N''.

Apply Definition 38.686 to each packet and then apply Definition 38.689. Both accepted comparisons are forced onto the same source route: the finite-window schedule, dyadic observation row, regularity row, and finite ledger are consumed before the local seal row is public, and the NameCert surface records only carrier admission, classifier transport, route stability, ledger exhaustion, and source factorization.

Now use Definition 38.688. The two comparisons identify the same four public faces, namely $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The first comparison transports the displayed rows from R to R', and the second transports the displayed rows from R' to R''.

Compose the componentwise hsame transports along the shared middle packet and replay the two Cont routes through the same finite source reads. The package and naming rows remain on the same displayed route, so the comparison from R to R'' is again a comparison of the public Real seal classifier.

Finally apply Definition 38.690 to the composed comparison. The classifier therefore presents the same public seal row for R and R'', and every step stayed inside the finite carrier and the four public faces. No quotient stream, selected host limit, host equality, ambient completeness theorem, metric-completeness datum, or choice row is introduced. □

Theorem 38.692 (Real name seal terminal source uniqueness)**.** *Let*

$$R = (D, S, Q, L, E, H, C, P, N) \qquad \textit{and} \qquad R' = (D', S', Q', L', E', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *name-seal packets for the carrier of Definition 38.685. Suppose the two packets agree componentwise on the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, the finite ledger, the local seal row, and the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. Then every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer that reads through either accepted packet reads the same terminal finite* BHist *source surface*

$$(S, D, Q, L, E, H, C, P, N).$$

The terminal surface is only this displayed finite packet. It is not a quotient stream, a selected host limit, a host equality witness, an ambient completeness principle, metric-completeness data, or a choice row.

Proof. Project both accepted packets through Definition 38.685. The projection exposes exactly the source rows S, D, Q, L, the seal row E, and the structural rows H, C, P, N; the primed packet exposes the corresponding primed rows and no additional analytic coordinate.

Apply Definition 38.686 to each packet. Any $\mathsf{Real}^{\uparrow}$ seal read, and hence any $\mathsf{RegSeqRat}^{\uparrow}$ consumer entering the local Real seal route, must pass through the displayed source rows before the seal row is public.

Next compose the four public faces of Definition 38.688. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces carry the same finite BHist route because the two packets agree on the corresponding source, ledger, transport, continuation, package, and naming rows.

Finally use Definition 38.690. The stability theorem identifies the visible seal row presented by the two accepted packets, so every downstream consumer sees the same terminal finite source

surface. Since each step was taken inside the carrier and the four public faces, no quotient stream, selected host limit, host equality, ambient completeness principle, metric-completeness datum, or choice row is introduced. □

Theorem 38.693 (Real finite-prefix readback uniqueness)**.** *Let*

$$R = (D, S, Q, L, E, H, C, P, N) \qquad \textit{and} \qquad R' = (D', S', Q', L', E', H', C', P', N')$$

be accepted $\mathsf{Real}^{\uparrow}$ *name-seal packets for the carrier of Definition 38.685. Suppose the packets expose the same finite* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback row, finite ledger row, local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then every public finite-prefix readback through either packet presents the same public* $\mathsf{Real}^{\uparrow}$ *seal row.*

This readback has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or a second seal coordinate.

Proof. Project both accepted packets through Definition 38.685. The projection exposes exactly the displayed rows $D, S, Q, L, E, H, C, P, N$ and their primed counterparts, so the finite-prefix readback can only inspect the StreamName window, dyadic observation, RegSeqRat regularity read, ledger, local seal, and structural support rows.

Apply Definition 38.686 to both packets. Each public read must pass through the route $S, D, Q, L \longmapsto E$ before a Real seal row is visible, and the same hypothesis identifies the corresponding source rows on the primed packet.

Next apply Definition 38.690 to the displayed source comparison and Definition 38.691 to compose any intermediate classifier presentation. The hsame and Cont rows replay the same finite route, while the Pkg and NameCert rows keep the readback on the same local seal classifier.

Finally use Definition 38.692. Since both packets have the same terminal finite BHist source surface, every downstream public finite-prefix readback sees the same terminal Real seal row. The carrier contains no quotient stream equality, host real equality, selected limit, countable-choice principle, ambient completeness theorem, metric-completeness datum, or second seal coordinate, so none can be introduced by the readback. □

Theorem 38.694 (Real name seal non-completeness boundary)**.** *An accepted* $\mathsf{Real}^{\uparrow}$ *name-seal carrier exports only the finite packet rows* $D, S, Q, L, E, H, C, P, N$ *of Definition 38.685. It exports no quotient stream, no selected limit, no host real equality, no host sequence equality, and no ambient completeness theorem.*

Proof. Inspect the carrier definition. Its analytic rows are the DyadicRatCore observation, StreamName schedule, RegSeqRat regularity readback, finite ledger, and RealUp seal; its structural rows are BHist, hsame, Cont, Pkg, and local NameCert evidence.

Apply Definition 38.686. Every seal read is forced through the displayed source route before E is public, so a consumer cannot enter through a quotient stream, selected host limit, host equality test, or ambient completeness principle.

Finally use Definition 38.689. The obligation surface records carrier admission, classifier transport, route stability, ledger exhaustion, and the same source factorization, leaving no remaining exported coordinate from which any of the excluded analytic objects could be obtained. □

Theorem 38.695 (Real name seal open-phase status readback)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *name-seal packet* $R = (D, S, Q, L, E, H, C, P, N)$ *of Definition 38.685 reads the Phase Real and Completion*

boundary only through the four public faces

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Consequently every downstream user of the packet must read closurestatus, Lean target, bridge status, and completeness claims separately from the four chapters* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The packet itself exports no quotient stream, selected limit, host equality, ambient Real completeness, metric-completeness data, countable-choice principle, or standard bridge row.

Proof. Apply Definition 38.686. It projects the accepted packet to the source route $S, D, Q, L \longmapsto E$ and leaves only the visible BHist, hsame, Cont, Pkg, and NameCert rows as structural support.

Apply Definition 38.688. That route identifies the public faces available to a Phase Real and Completion read as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no additional analytic coordinate in the packet.

Use Definition 38.692 to make the readback boundary objectwise. Any downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer that enters through the packet sees the same terminal finite BHist source surface, so it cannot replace the four chapter surfaces by a single global verification or completeness row.

Finally apply Definition 38.694. The packet has no coordinate for quotient streams, selected limits, host equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge evidence. The status rows for closure, Lean target, bridge status, and completeness therefore remain chapter-local obligations of the four public faces. □

Theorem 38.696 (Real constant dyadic name-seal exhaustion)**.** *Let C be a* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packet over a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint d, and let*

$$R = (D, S, Q, L, E, H, C_R, P, N)$$

be an induced accepted $\mathsf{Real}^{\uparrow}$ *name-seal packet for the carrier of Definition 38.685. Then every public read of R factors through exactly the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule of C, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row d, the requested dyadic-radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier ledger of C, the finite ledger L, and the local* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 38.685.*

Equivalently, the induced packet is exhausted by the route

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream, selected limit, host real equality, ambient completeness row, metric-completeness datum, choice principle, or standard bridge row.*

Proof. Project the induced packet through Definition 38.685. The projection exposes the finite-window schedule row S, the dyadic observation row D, the RegSeqRat row Q, the finite ledger L, the local seal row E, and the structural rows H, C_R, P, N.

Apply Definition 301.131 to the constant source packet. Every requested finite window is therefore represented by the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint d, the requested radius rows, the equal-endpoint dyadic-distance row, the regularity ledger, and the componentwise transport, continuation, package, and naming rows.

Next apply Definition 301.132 and Definition 301.136. These results say that every $\mathsf{Real}^{\uparrow}$-facing request made of the constant source reaches the seal only through the constant $\mathsf{StreamName}^{\uparrow}$

schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius rows, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier ledger, the cofinal finite window, and the same finite BEDC structural rows.

Now repack that source description through the local Real seal route. Use Definition 38.686 to force any public read of R through $S, D, Q, L \longmapsto E$, and use Definition 38.688 to identify the only public faces of that route as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 38.694. Since neither the constant RegSeqRat source nor the Real name-seal carrier has a coordinate for quotient streams, selected limits, host real equality, ambient completeness, metric-completeness data, choice, or a standard bridge, the induced packet has no remaining public row beyond the listed constant-source and local seal rows. □

Theorem 38.697 (Real dyadic constant classifier handoff)**.** *Let C and C' be* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packets over displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints d and d', and let*

$$R = (D, S, Q, L, E, H, C_R, P, N), \qquad R' = (D', S', Q', L', E', H', C_{R'}, P', N')$$

be the induced accepted $\mathsf{Real}^{\uparrow}$ *name-seal packets for the carrier of Definition 38.685. Any public* $\mathsf{Real}^{\uparrow}$ *classifier read comparing R and R' may be exposed only after the following rows have been displayed on both sides:*

1. *the constant* $\mathsf{StreamName}^{\uparrow}$ *schedules of C and C';*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, tolerance, and equal-endpoint request rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows and classifier ledgers of the constant sources;*
4. *the finite ledgers L and L' tying the requested windows to the local packets;*
5. *the local* $\mathsf{Real}^{\uparrow}$ *seal rows E and E' with their displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Thus a Real equality or order consumer reading the induced constant dyadic names receives only the displayed constant-source route and the local Real seal rows. It receives no quotient stream, selected limit, host real equality test, ambient completeness row, metric-completeness datum, choice principle, or standard bridge row.

Proof. Apply Definition 38.696 to both induced packets. Each public read of R and R' is already exhausted by the constant StreamName schedule, the DyadicRatCore endpoint and radius rows, the RegSeqRat regularity and classifier ledger, the finite ledger, and the local Real seal rows.

Next use the local carrier route. The source factorization of Definition 38.686 forces every Real seal read through $S, D, Q, L \longmapsto E$ and $S', D', Q', L' \longmapsto E'$, while Definition 38.688 identifies the only public faces of those routes as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The constant-source side supplies the classifier handoff. Apply Definition 301.131 to obtain the equal-endpoint dyadic request rows and the regularity ledger for each requested finite window. Then apply Definition 301.132 and Definition 301.136 to show that any Real-facing request from the constant source has already consumed the constant StreamName schedule, the dyadic endpoint and tolerance rows, the RegSeqRat classifier ledger, and the finite BEDC structural rows.

Finally repack the two induced packets through the Real name-seal classifier surface. Since all cited routes expose only the displayed finite rows and the local seal support, a Real equality or order consumer has no remaining coordinate for quotient streams, selected limits, host equality, ambient completeness, metric-completeness data, choice, or standard bridge evidence. □

Theorem 38.698 (RealUp RegSeqRat finite-request fusion)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over R. Then the request factors through one finite* BHist *packet whose visible source rows are exactly the following data:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule row selected by the request;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite distance rows read at that schedule;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row and regularity rows that certify the accepted packet;*
4. *the componentwise* hsame *transports tying the schedule, dyadic rows, regularity rows, and local seal row to the same request packet;*
5. *the displayed* Cont *routes from the schedule through the dyadic and regularity rows to the seal read;*
6. *the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
7. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row for the finite seal request.*

Every downstream consumer of the public seal request reads through this same packet. The fusion exports no quotient stream equality, no host real equality, no selected limit, no ambient completeness principle, no choice data, and no second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Lean4 ✓ `BEDC.Derived.RealUp.RealRegseqratFiniteRequestFusion`

Proof. First apply Definition 38.417. It places the public seal request on one common finite window with the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ regularity spine, and local $\mathsf{Real}^{\uparrow}$ seal provenance.

Next use Definition 301.69. The accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet can reach the real seal only through the StreamName schedule rows, dyadic endpoint and radius ledgers, finite dyadic-distance rows, classifier rows, and regularity rows named by the same finite handoff.

Now apply Definition 301.113. The tail-budget window pulls the seal read back to the same finite rows, so later consumers cannot substitute an off-packet tail selector, host limit, or second seal coordinate.

Finally use Definition 38.686. Repacking the common-window handoff, the RegSeqRat exactness route, and the tail-budget pullback gives one BHist source packet with exactly the schedule, dyadic, classifier, regularity, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows listed above. Since each cited boundary lacks quotient stream equality, host real equality, selected limits, ambient completeness, choice data, and duplicate Real-seal coordinates, the fused request exports none of them. □

Theorem 38.699 (RealUp RegSeqRat finite-request selector-budget lock)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. Suppose the finite request fusion is displayed over the rows (W, D, R, H, C, P, N), where W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius row family, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row, H is the componentwise* hsame *provenance, C is the displayed* Cont *route, P is the* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row. Then any downstream selector-budget consumer or completion-facing consumer of the same* $\mathsf{Real}^{\uparrow}$ *seal must reuse exactly these rows. In particular it cannot introduce a second selector budget, an off-packet tail selector, quotient stream equality, host real equality, a selected limit, choice data, or an ambient completeness row.*

Proof. Apply Definition 38.698. The public finite request has already been repacked into one BHist packet whose displayed rows are the window row W, the dyadic endpoint and radius rows D, the classifier and readback row R, and the provenance rows H, C, P, and N.

Apply Definition 38.417. This identifies every seal read in the packet with the same $\mathsf{StreamName}^{\uparrow}$ finite-window row, so a later selector-budget consumer has no second window coordinate through which to read the seal.

Apply Definition 301.113. The tail-budget route is pulled back to the displayed finite window and its dyadic and regularity rows, ruling out an off-packet tail selector or a second selector budget.

Finally apply Definition 38.686. The final $\mathsf{Real}^{\uparrow}$ seal source is the same finite source packet with its $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row, so completion-facing consumers inherit the listed W, D, R, H, C, P, and N rows and gain no quotient stream equality, host real equality, selected limit, choice data, or ambient completeness row. □

Theorem 38.700 (RealUp RegSeqRat finite-request triangle)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. The following four displayed BEDC routes mutually determine one another:*

1. *S is the* $\mathsf{Real}^{\uparrow}$ *finite-request fusion packet displayed by Definition 38.698;*
2. *S lands in one* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the scheduled unary observation read by the request;*
3. *that same finite-window row carries the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, finite-distance ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row;*
4. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *seal read is reached only through those StreamName, dyadic-ledger, classifier, readback,* hsame, Cont, *and* Pkg *rows.*

Thus any one of the four route descriptions reconstructs the other three inside the same finite BHist *packet. The triangle exports no quotient stream equality, no host real equality, no selected limit, no choice data, and no ambient completeness principle.*

Proof. Start with the fusion packet route. By Definition 38.698, the public request is already repacked as one finite BHist packet whose visible rows include the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius family, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows, the componentwise hsame transports, the displayed Cont routes, the Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row.

The StreamName corner is determined by Definition 38.417. It places the seal request and the scheduled unary observation on a single common finite window, so the fusion packet determines the window route and the window route recovers the seal-facing source row of the packet.

The dyadic and classifier corner is determined by Definition 301.69. The same finite handoff carries the dyadic endpoint, radius, finite-distance ledger, classifier row, and readback row, and the handoff admits no off-window replacement for those rows.

It remains to identify the seal corner. Apply Definition 301.113 to pull the seal read back to the displayed window and ledger rows, then apply Definition 38.686 to factor the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal source through exactly that finite packet.

Each direction of the triangle is therefore a repackaging of the same source rows: fusion packet, StreamName window, dyadic-RegSeqRat ledger, and local Real seal. Since the cited routes carry only finite BHist, hsame, Cont, Pkg, and NameCert provenance, the correspondence introduces no quotient stream equality, no host real equality, no selected limit, no choice data, and no ambient completeness principle. □

Theorem 38.701 (RealUp RegSeqRat completion-consumer section exactness)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q, and let (W, D, R, H, C, P, N) be the finite request section displayed by Definition 38.699. A completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing*

consumer of S is accepted exactly when its visible rows factor through that section: W is the $\mathsf{StreamName}^{\uparrow}$ *finite-window row, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance row family, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row family, H is the componentwise* hsame *provenance, C is the displayed* Cont *route, P is the* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *row. Thus the consumer boundary exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness principle, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. For the forward direction, apply Definition 38.698. The accepted seal request is represented by one finite BHist packet whose visible rows are the StreamName window, dyadic endpoint, radius, and distance rows, RegSeqRat classifier and readback rows, componentwise hsame rows, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row.

Next apply Definition 38.699. Any completion-facing or selector-budget consumer of the same seal must reuse the displayed (W, D, R, H, C, P, N) rows, so an accepted consumer has no remaining visible coordinate outside the finite section.

For the reverse direction, assume the visible rows of the completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer factor through (W, D, R, H, C, P, N). Apply Definition 301.140 to identify this section as the entire RegSeqRat source section seen by a Real seal, with no escaping source coordinate.

Finally apply Definition 126.546. The StreamName finite-window row is terminal among accepted finite-window consumers with the same observation, tail-selector route, budget surface, pointwise handoff, and support rows. Repacking the terminal window with the dyadic, classifier, readback, transport, route, package, and local naming rows gives exactly the accepted finite section above, and the exclusions from the cited nonescape and terminality theorems rule out quotient stream equality, host real equality, selected limits, choice data, ambient completeness, and a second $\mathsf{Real}^{\uparrow}$ seal coordinate. □

Theorem 38.702 (RealUp RegSeqRat finite-request four-face source exhaustion)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. Then S is accepted exactly through the following four displayed source faces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule face;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance face;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity face;*
4. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *seal face.*

The only transport between these faces is the componentwise hsame *family, the displayed* Cont *routes, and the* Pkg *provenance already present in the finite request section. Consequently the accepted source of S is exhausted by those four faces, and no quotient stream equality, host real equality, selected limit, choice data, ambient completeness principle, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate is exported.*

Proof. Apply Definition 38.698. It supplies the finite BHist packet whose visible rows are the StreamName schedule, DyadicRatCore endpoint, radius, and distance data, RegSeqRat classifier and regularity data, componentwise hsame, displayed Cont, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal row.

Next apply Definition 38.699. The displayed finite request section is the unique section available to selector-budget and completion-facing consumers of the same seal request, so none of the four faces can be replaced by a second section.

Apply Definition 38.700. The fusion packet, StreamName window, DyadicRatCore-RegSeqRat ledger, and local Real seal face mutually determine one another inside the same finite packet.

Finally use Definition 38.701. Its accepted-consumer boundary says that every completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer is accepted precisely when its visible rows factor through this same section. The four displayed faces therefore give the complete accepted source for S, while the exclusions recorded by the cited results rule out quotient stream equality, host real equality, selected limits, choice data, ambient completeness, and a second Real seal coordinate. □

Theorem 38.703 (StreamName-RegSeqRat-Real finite-request seal coherence)**.** *Let W be a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package, let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet whose finite-window seal consumes the selected rows of W, and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. Suppose the finite request displays the same* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same componentwise* hsame *family, the same displayed* Cont *routes, the same* Pkg *provenance, and the same local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *rows. Then the StreamName-RegSeqRat seal-composition route and the* $\mathsf{Real}^{\uparrow}$ *four-face source-exhaustion route determine the same finite* BHist *packet. This coherence exports no quotient stream equality, no host real equality, no selected limit, no countable-choice schedule, no ambient completeness row, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 126.79. The public finite-window package W supplies the StreamName source rows consumed by the accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, and those rows are the displayed window rows of the request.

Next apply Definition 126.83. The seal route from the StreamName window through the accepted RegSeqRat packet reaches the Real-facing seal only through the selected window rows, the RegSeqRat classifier and readback rows, and their displayed provenance.

Now apply Definition 38.698 and Definition 38.700. These results repack the displayed StreamName, dyadic, RegSeqRat, hsame, Cont, Pkg, and local naming rows as one finite BHist packet, and they identify the StreamName window route, the dyadic-RegSeqRat ledger route, and the Real seal route as mutually determining faces of that packet.

Finally use Definition 38.702. Its four-face source exhaustion says that the accepted source of S has no visible face beyond the StreamName window, DyadicRatCore endpoint-radius-distance data, RegSeqRat classifier-readback data, and local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal row. Hence the StreamName-RegSeqRat seal-composition route and the RealUp four-face route name the same finite BHist packet, with none of the excluded quotient, host-real, selected-limit, countable-choice, completeness, or duplicate-seal data exported. □

Theorem 38.704 (RealUp RegSeqRat finite-request section terminality)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. Suppose two accepted completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing finite-request sections over S display the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same componentwise* hsame *family, the same displayed* Cont *routes, the same* Pkg *provenance, and the same local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *rows. Then the two sections determine the same finite* BHist *packet. This terminal section exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness row, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 38.698. Each accepted finite request is represented by one finite BHist packet whose visible rows are exactly the StreamName window, dyadic endpoint-radius-distance

data, RegSeqRat classifier-readback data, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ rows.

Next apply Definition 38.699. A completion-facing or $\mathsf{Real}^\uparrow$-facing consumer of the same seal cannot replace the displayed window, dyadic, classifier, readback, transport, route, package, or local naming rows by a second selector-budget section.

The StreamName coordinate is terminal by Definition 126.546. Since the two sections display the same finite window and the same compatible tail-budget surface, both StreamName faces collapse to the same window packet.

Now use Definition 38.701. Acceptance of either completion-facing or $\mathsf{Real}^\uparrow$-facing section is exactly factorization through the displayed (W, D, R, H, C, P, N) rows, so the two sections have no visible coordinate outside those rows.

Finally apply Definition 38.702. The StreamName, DyadicRatCore, RegSeqRat, and local Real naming faces exhaust the accepted source of S. Since the two sections agree on all four faces and on the only permitted hsame, Cont, and Pkg transports between them, they determine the same finite BHist packet, and none of the excluded quotient, host-real, selected-limit, choice, completeness, or duplicate-seal data is exported. □

Theorem 38.705 (RealUp StreamName finite-request terminal bridge)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^\uparrow$ *packet, let S be an accepted* $\mathsf{Real}^\uparrow$ *finite-request terminal section over Q, and let P be the* $\mathsf{StreamName}^\uparrow$ *finite-section pullback for the finite window used by S. Then S and P determine the same finite* BHist *packet exactly when they display the same* $\mathsf{StreamName}^\uparrow$ *window row, the same* $\mathsf{DyadicRatCore}^\uparrow$ *endpoint and radius ledger, the same* $\mathsf{RegSeqRat}^\uparrow$ *classifier and readback rows, the same componentwise* hsame *rows, the same displayed* Cont *route, the same* Pkg *provenance, and the same local* NameCert *rows. This bridge exports no quotient streams, no selected limits, no host equality, no ambient completeness row, no countable-choice schedule, and no standard bridge rows.*

Proof. For the Real-facing direction, apply Definition 38.704. The accepted terminal section over S is already exhausted by the displayed StreamName window, DyadicRatCore endpoint-radius ledger, RegSeqRat classifier and readback rows, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

For the StreamName-facing direction, apply Definition 126.586 together with Definition 126.577. The pullback section has no visible source beyond the finite window rows and the finite request rows transported through the same BHist packet.

If the displayed window, endpoint-radius ledger, classifier and readback rows, hsame rows, Cont route, Pkg provenance, and local NameCert rows agree, the two cited terminal descriptions have the same four visible faces and the same permitted transports between them. Hence the Real terminal section and the StreamName pullback name one finite BHist packet.

Conversely, if they name the same finite BHist packet, equality of the visible packet rows forces agreement of each displayed row listed in the statement: the StreamName window, dyadic ledger, RegSeqRat classifier and readback data, componentwise transport rows, continuation route, package provenance, and local naming rows.

Finally apply Definition 126.453. The correspondence remains on the four-object finite exit route, so it does not add quotient streams, selected limits, host equality, ambient completeness, countable-choice schedules, or standard bridge rows. □

Theorem 38.706 (RealUp RegSeqRat finite-request pullback determinacy)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^\uparrow$ *packet and let S be a public finite request to a* $\mathsf{Real}^\uparrow$ *seal over Q. Suppose two accepted completion-facing or* $\mathsf{Real}^\uparrow$*-facing finite-request sections over S display the same* $\mathsf{StreamName}^\uparrow$ *finite-section pullback, the same* $\mathsf{DyadicRatCore}^\uparrow$ *endpoint, radius, and finite-distance rows, the*

same $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the two sections determine the same finite* BHist *Real seal packet. This determinacy exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness row, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First use Definition 126.586. The shared $\mathsf{StreamName}^{\uparrow}$ finite-section pullback identifies the StreamName-facing part of both accepted sections with the same finite four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 38.702. Once the StreamName pullback is fixed, the accepted source of the request has only the StreamName schedule face, the DyadicRatCore endpoint-radius-distance face, the RegSeqRat classifier-readback face, and the local Real naming face.

Now use Definition 38.701. Each accepted completion-facing or $\mathsf{Real}^{\uparrow}$-facing section must factor through those visible rows together with the displayed hsame, Cont, Pkg, and NameCert support rows, so neither section has an additional consumer coordinate.

Finally apply Definition 38.704. The two sections agree on the StreamName pullback and on every visible dyadic, classifier, readback, local seal, transport, route, package, and naming row; terminality therefore identifies the displayed finite BHist Real seal packet. The cited boundaries keep quotient stream equality, host real equality, selected limits, choice data, ambient completeness, and a second Real seal coordinate outside the packet. □

Theorem 38.707 (RealUp RegSeqRat finite-request route equivalence)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. A completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing finite-request read over S is accepted exactly when it factors through the same four displayed L10 source faces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by the request;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger read at that window;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity rows of the accepted packet;*
4. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *seal row.*

The factorization includes precisely the displayed BHist*,* hsame*,* Cont*, and* Pkg *support tying those four faces to one finite packet. The displayed packet is unique among accepted completion-facing and* $\mathsf{Real}^{\uparrow}$*-facing reads with these rows, and the equivalence exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness, no metric-completeness row, no standard bridge row, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. For the forward direction, apply Definition 38.702. Any accepted read over S has no visible source beyond the StreamName finite window, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier-readback-regularity surface, and the local Real naming row.

For the reverse direction, assume the read factors through these four faces together with the displayed BHist, hsame, Cont, and Pkg support. Apply Definition 38.701. Its section criterion accepts exactly the completion-facing and $\mathsf{Real}^{\uparrow}$-facing consumers whose visible rows factor through the displayed finite section, so the factored read is accepted.

It remains to identify the packet. Apply Definition 38.706 to any two accepted reads displaying the same four faces and support rows. Then use Definition 38.705 to identify the StreamName-facing pullback with the Real-facing terminal section on the same finite BHist packet.

Thus accepted reads are precisely the four-object L10 finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal row. Since the cited boundaries carry only the listed finite support, the route does not add quotient stream equality, host real

equality, selected limits, choice data, ambient completeness, metric-completeness rows, standard bridge rows, or a second Real seal coordinate. □

Theorem 38.708 (RealUp RegSeqRat terminal four-face acceptance equivalence)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q. Acceptance of a completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing terminal read over S is equivalent to factorization through exactly the four L10 faces:*

1. *the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity rows;*
4. *the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *seal row.*

The factorization uses only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support tying those faces to one finite packet. Among accepted terminal reads displaying this support, the packet is unique. The equivalence exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness, no metric-completeness row, no standard bridge row, and no second Real seal coordinate.*

Proof. For the forward direction, apply Definition 38.707. An accepted terminal read over S is accepted only through the selected StreamName window, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier-readback-regularity surface, and the local Real seal row, with the displayed BHist, hsame, Cont, Pkg, and NameCert support.

For the reverse direction, suppose a terminal read factors through exactly these four faces with the displayed support. Apply Definition 301.239. Its four-face exactness repacks the StreamName, DyadicRatCore, RegSeqRat, and Real seal rows as the accepted terminal packet for the public finite request.

It remains to identify uniqueness. First use Definition 38.706: two accepted terminal reads with the same four faces and support rows determine the same finite Real seal packet.

Then apply Definition 38.705 to identify the StreamName-facing pullback with the Real-facing terminal section on that packet. Finally use Definition 38.458; no second completion-facing or Real-facing terminal read can display the same four L10 faces while naming a different finite BHist packet.

The cited route, exactness, bridge, and pullback-uniqueness statements expose only the listed finite rows. Therefore the equivalence carries no quotient stream equality, host real equality, selected limit, choice data, ambient completeness, metric-completeness row, standard bridge row, or second Real seal coordinate. □

Theorem 38.709 (RealUp RegSeqRat Cauchy-witness budget projection)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q, and let $L = (Q_L, B_L, S_L, K_L, H_L, C_L, P_L, N_L)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *budget row whose visible* $\mathsf{StreamName}^{\uparrow}$ *finite window is the same window read by S. Then the visible budget projection of the combined read is exactly the displayed packet consisting of the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the supporting* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Equivalently, the Cauchy-witness ledger consumes the same finite packet through its diagonal-budget, observation-budget, synchronizer, and completion-seal rows before any completion-facing classifier is visible.*

The projection exports no quotient streams, no host real equality, no selected limits, no choice data, no ambient completeness, no metric-completion rows, no standard bridge rows, and no second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 38.698 and Definition 38.699. They place the public request on one finite packet and force every selector-budget or completion-facing consumer of S to reuse the displayed StreamName, dyadic, RegSeqRat, Real seal, and support rows.

Next use Definition 38.702. The visible source has exactly four faces: the StreamName finite window, the dyadic endpoint-radius-distance ledger, the RegSeqRat classifier-readback surface, and the local Real seal row. There is no fifth face from which a quotient stream, selected limit, ambient completeness row, metric-completion row, or bridge row could be read.

Now apply the terminal bridge Definition 38.705. Since the ledger row and the Real request display the same finite StreamName window, the StreamName-facing pullback and the Real-facing terminal section name the same finite BHist packet with the same hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Finally read the witness-ledger side through Definition 902.5, Definition 902.7, and Definition 902.49. These rows order every accepted completion-facing read through the same diagonal-budget, observation-budget, synchronizer, and completion-seal route, and they identify the ledger's $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source surface with the one already fixed by the finite request. Thus all budget consumers reuse the same finite packet, and the excluded quotient, host, selected-limit, choice, completeness, metric-completion, bridge, and duplicate-seal data remain outside the projection. □

Theorem 38.710 (RealUp RegSeqRat StreamName dyadic tail-budget pullback)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q, and let $L = (E, T, W, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *row whose visible* $\mathsf{StreamName}^{\uparrow}$ *finite window W is the same window selected by S. Then the* $\mathsf{Real}^{\uparrow}$ *finite-request packet pulls back exactly to the shared finite packet consisting of the* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance rows in D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity rows of Q and R, the local* $\mathsf{Real}^{\uparrow}$ *seal row A, and the supporting* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows H, C, P, N.*

Equivalently, every completion-facing or $\mathsf{Real}^{\uparrow}$*-facing read of S that is mediated by the tail-radius ledger L is accepted exactly through the same four L10 faces: the selected StreamName window, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier and readback surface, and the local Real seal row. The pullback exports no quotient streams, no host real equality, no selected limits, no choice data, no ambient completeness, no metric-completion rows, no standard bridge rows, and no duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates.*

Proof. Start with the Real finite request. By Definition 38.702, the accepted source of S has exactly the StreamName window face, the DyadicRatCore endpoint-radius-distance face, the RegSeqRat classifier-readback-regularity face, and the local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal face.

Next apply Definition 38.707. A completion-facing read over S is accepted precisely when it factors through those four faces together with the displayed BHist, hsame, Cont, and Pkg support. Thus the Real-facing section has no coordinate outside the finite L10 packet.

Now use Definition 38.709. Its budget projection identifies the Cauchy-witness budget surface with the same StreamName, DyadicRatCore, RegSeqRat, Real seal, and support rows already fixed by the finite request.

Finally apply the DyadicTailRadiusLedger handoff, namely Definition 757.3 and Definition 757.6. Since L displays the same finite StreamName window W selected by S, tail-budget coverage places

the precision, tail-window, StreamName-window, and dyadic readback rows on the same finite route, and the handoff carries that route through the RegSeqRat and Real seal rows. Repacking these rows gives exactly the shared packet $W, D, Q, R, A, H, C, P, N$, with no quotient stream, host equality, selected limit, choice, completeness, metric-completion, bridge, or duplicate Real-seal coordinate available to the pullback. □

Theorem 38.711 (RealUp RegSeqRat reindexed finite-request descent)**.** *Let Q_0 and Q_1 be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets, and let S_0 and S_1 be accepted finite requests to* $\mathsf{Real}^{\uparrow}$ *seals over them. Suppose the selected* $\mathsf{StreamName}^{\uparrow}$ *finite windows of S_0 and S_1 are related by a certified public reindex route. Suppose also that the route preserves the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then S_0 and S_1 determine the same finite* $\mathsf{Real}^{\uparrow}$ *seal packet.*

Conversely, equality of the accepted finite $\mathsf{Real}^{\uparrow}$ *seal packet recovers exactly that reindexed four-face data: the transported StreamName window, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier and readback rows, the local Real seal row, and the displayed support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The descent exports no quotient stream equality, no host real equality, no selected limits, no choice data, no ambient completeness, no metric-completeness rows, no standard bridge rows, and no duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates.*

Proof. First apply Definition 126.642. The certified public reindex route carries the selected StreamName finite window to the same terminal four-face exit route, with the displayed BHist, hsame, Cont, Pkg, and NameCert rows transported componentwise.

Next use Definition 38.705. The StreamName-facing pullback and the Real-facing finite request therefore read one finite BHist packet whenever they display the same StreamName window, dyadic ledger, RegSeqRat readback, Real seal row, and support rows.

Now apply Definition 38.707. Acceptance of either request is exactly factorization through the four L10 faces: the selected StreamName window, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier-readback surface, and the local Real seal row, together with the displayed finite support.

Finally use Definition 38.710. The preserved dyadic tail-budget ledger identifies the Real-facing finite packet with the same StreamName, DyadicRatCore, RegSeqRat, Real seal, and support rows on both sides of the reindex route. Thus the two accepted requests determine the same finite $\mathsf{Real}^{\uparrow}$ seal packet.

For the converse, read equality of the accepted finite Real seal packet through the route equivalence just cited. Since the packet has no hidden coordinate beyond the four faces and displayed support, equality recovers exactly the transported StreamName window, dyadic ledger, RegSeqRat classifier and readback rows, local seal row, and BHist/hsame/Cont/Pkg/NameCert support rows. The same boundary exclusions keep quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, standard bridge rows, and duplicate Real seal coordinates outside the conclusion. □

Theorem 38.712 (RealUp RegSeqRat tail-witness budget agreement)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let S be a public finite request to a* $\mathsf{Real}^{\uparrow}$ *seal over Q, let L be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *budget row, and let T be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *row. Suppose that the visible* $\mathsf{StreamName}^{\uparrow}$ *finite window read by L is the same finite window selected by S, and that the visible* $\mathsf{StreamName}^{\uparrow}$ *finite window read by T is the same selected window. Then the Cauchy-witness budget projection and the dyadic tail-budget pullback determine one and the same finite four-face* $\mathsf{Real}^{\uparrow}$ *packet: the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint,*

radius, and finite-distance rows, the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Conversely, equality of that finite four-face packet recovers agreement of the two displayed budget routes on precisely those rows. The agreement exports no quotient stream equality, no host real equality, no selected limit, no choice data, no ambient completeness, no metric-completeness row, no standard bridge row, and no duplicate $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 38.709. The accepted CauchyWitnessLedger row L projects to the selected StreamName window, the dyadic endpoint-radius-distance rows, the RegSeqRat classifier and readback surface, the local Real seal row, and the displayed finite support rows.

Next apply Definition 38.710. Since T displays the same selected StreamName window, its tail-radius route pulls the Real finite request back to the identical StreamName, DyadicRatCore, RegSeqRat, local seal, and BHist/hsame/Cont/Pkg/NameCert support surface.

Now use Definition 38.707. Acceptance through either budget route is exactly acceptance through the four L10 faces, so neither route contributes a fifth coordinate or any completion-facing row outside the finite packet.

It remains to identify the two budget descriptions. Apply Definition 38.711 with the identity public reindex route on the shared selected StreamName window. The descent theorem identifies the two accepted finite Real seal packets because their StreamName, dyadic, RegSeqRat, local seal, and support rows agree.

For the converse, read equality of the shared packet back through the same descent and route-equivalence statements. Since the packet contains only the four displayed faces and support rows, equality recovers exactly the Cauchy-witness projection route and the dyadic tail-budget pullback route on those rows, while the listed quotient, host, limit, choice, completeness, metric-completion, standard bridge, and duplicate-seal data remain outside the conclusion. □

Theorem 38.713 (Real RegSeqRat terminal budget exhaustion)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a finite RegSeqRat request accepted by the* $\mathsf{Real}^{\uparrow}$ *seal over Q. A terminal-budget consumer of S is accepted exactly when its visible rows factor through the ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The packet consists of the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *finite schedules; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the local* $\mathsf{Real}^{\uparrow}$ *terminal seal row, all transported by the displayed* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *coordinates.*

Conversely, any packet with exactly these four faces and exactly these support rows repacks as the accepted $\mathsf{Real}^{\uparrow}$ *terminal-budget read of S. The exhaustion exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness principle, arithmetic package, order package, or metric-completeness row.*

Proof. For the forward direction, begin with an accepted terminal-budget consumer of the finite RegSeqRat request. Apply Definition 301.142 to place the request on its finite terminal-budget section, with the RegSeqRat classifier, regularity, readback, and provenance rows already tied to a finite source packet.

Next apply Definition 38.699. The $\mathsf{Real}^{\uparrow}$ seal over the accepted RegSeqRat packet must reuse the same finite request section, so the StreamName schedule, DyadicRatCore endpoint, radius, and distance ledgers, RegSeqRat rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row cannot be replaced by a second terminal-budget surface.

Now use Definition 38.457. It identifies the visible rows of the accepted terminal $\mathsf{Real}^{\uparrow}$ read with the finite four-face packet over DyadicRatCore, StreamName, RegSeqRat, and the local Real seal. Thus the terminal-budget consumer has no visible coordinate outside the ordered packet in the statement.

For the reverse direction, suppose the displayed DyadicRatCore ledgers, StreamName schedules, RegSeqRat regularity and classifier rows, and local $\mathsf{Real}^{\uparrow}$ terminal seal row are already present with the listed support coordinates. Apply Definition 38.455 to repack those rows as the terminal $\mathsf{Real}^{\uparrow}$ read.

It remains to match that terminal read with the original finite request. The finite-request factorization and selector-budget lock identify the RegSeqRat source section and the $\mathsf{Real}^{\uparrow}$ seal section as the same packet, so the repacked four-face read is exactly the accepted terminal-budget read of S. Since the cited routes do not export quotient stream equality, host real equality, selected limits, countable-choice selectors, ambient completeness, arithmetic or order packages, or metric-completeness rows, none of those data are available through this exhaustion. □

Lean4 ✓ `BEDC.Derived.RealUp.RealRegseqratTerminalBudgetExhaustion`

Theorem 38.714 (Real RegSeqRat terminal budget seal uniqueness)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a finite RegSeqRat request accepted by the* $\mathsf{Real}^{\uparrow}$ *seal over Q. Suppose two accepted terminal-budget reads of S display the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the same* $\mathsf{StreamName}^{\uparrow}$ *finite schedules, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support rows.*

Then the two public finite $\mathsf{Real}^{\uparrow}$ *seal rows coincide. This equality is only the equality of the visible terminal-budget seal row; it does not assert quotient stream equality, host real equality, selected limits, countable-choice selectors, ambient completeness, arithmetic or order packages, metric-completeness rows, standard bridge rows, or any equality of hidden completion carriers.*

Proof. Apply Definition 38.713 to each accepted terminal-budget read. Both reads factor through the ordered four-face packet

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The hypotheses identify every displayed face of these two packets: the dyadic endpoint, radius, and distance ledgers, the StreamName schedules, the RegSeqRat regularity, classifier, and provenance rows, and the local Real seal row. They also identify the support rows BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert.

Next use Definition 38.459 and Definition 38.455. The source lock keeps both terminal reads on the same L10 source route, while the four-face correspondence reads the identified packet as the terminal $\mathsf{Real}^{\uparrow}$ seal read. Thus neither read can obtain a second terminal source after the four visible faces and support rows have been fixed.

It remains to check that both reads are attached to the same finite request section. Apply Definition 301.142 to the RegSeqRat request and Definition 38.699 to the Real seal over it. These two locks identify the RegSeqRat source section and the Real selector-budget section as the same terminal-budget packet for S.

Finally invoke Definition 126.500 and Definition 126.449. The exit route names the same four public objects, and terminality says that agreement on those objects and the displayed support rows determines a single terminal finite route. Hence the visible public $\mathsf{Real}^{\uparrow}$ seal row of the first read is the visible public $\mathsf{Real}^{\uparrow}$ seal row of the second read.

All equalities used above are equalities of the finite displayed packet. The cited routes export no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness, arithmetic or order package, metric-completeness row, standard bridge row, or hidden completion carrier, so the uniqueness conclusion has exactly the public finite seal-row scope stated above. □

Theorem 38.715 (Real RegSeqRat terminal budget four-face naturality)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let S be a finite RegSeqRat request accepted by the* $\mathsf{Real}^{\uparrow}$ *seal over Q. Suppose an accepted terminal-budget read of S is transported through the displayed* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support rows, while preserving the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedules, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal row.*

Then the transported read has the same public $\mathsf{Real}^{\uparrow}$ *terminal-budget seal row as the original read, and it is the same four-face read through*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The transported read exports no quotient stream equality, selected limit, choice selector, ambient completeness, arithmetic or order package, metric-completeness row, standard bridge row, or hidden completion carrier.

Proof. Apply terminal-budget exhaustion to the original read. The accepted read is exhausted by the four displayed faces, together with the DyadicRatCore ledgers, StreamName schedules, RegSeqRat rows, local Real seal row, and the listed support rows.

Apply the same exhaustion theorem to the transported read. The transport hypotheses preserve exactly those visible rows and support coordinates, so the transported read factors through the same ordered four-face packet and has no additional terminal-budget coordinate.

Now use terminal-budget seal uniqueness. Since the two accepted reads display the same dyadic, schedule, regularity, classifier, provenance, Real seal, transport, continuation, package, and naming rows, their public finite $\mathsf{Real}^{\uparrow}$ seal rows coincide.

The non-escape conclusion is inherited from the two preceding results. Exhaustion supplies no hidden completion data, and uniqueness identifies only the visible finite seal row, so the transported read cannot export quotient stream equality, selected limits, choice selectors, ambient completeness, arithmetic or order packages, metric-completeness rows, standard bridge rows, or hidden completion carriers. □

Theorem 38.716 (Real terminal-budget pasting exactness)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet under a fixed* $\mathsf{Real}^{\uparrow}$ *seal, and let S_0 and S_1 be accepted finite RegSeqRat requests with an accepted overlap S_{01}. Suppose their terminal-budget reads agree on S_{01} in the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedules, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows. Then the pasted terminal-budget read has the same public* $\mathsf{Real}^{\uparrow}$ *seal row as the two restrictions on the overlap and factors through the same ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The pasted read exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness principle, arithmetic package, order package, metric-completeness row, standard bridge row, or hidden completion carrier.

Proof. Apply Definition 38.713 to the two accepted reads. Each restriction is exhausted by the DyadicRatCore endpoint, radius, and distance ledgers, the StreamName schedules, the RegSeqRat regularity, classifier, and provenance rows, the local Real seal row, and the displayed support rows.

On the overlap S_{01}, use Definition 38.714. The agreement hypotheses identify every visible terminal-budget row, so the public $\mathsf{Real}^{\uparrow}$ seal row is single on the overlap.

Paste the two terminal-budget reads along this common seal row. The pasted read has exactly the displayed dyadic, StreamName, RegSeqRat, Real seal, and support coordinates inherited from its restrictions, with no extra terminal coordinate at the gluing boundary.

Finally apply Definition 38.715. Naturality transports the pasted read along the same support rows and keeps it on the ordered four-face route. The non-escape clauses are inherited from exhaustion, seal uniqueness, and naturality, so the pasted terminal-budget read does not claim quotient, host-real, selected-limit, choice, completeness, arithmetic, order, metric-completion, bridge, or hidden-completion data. □

Theorem 38.717 (Real terminal-budget finite-cover descent)**.** *Let Q be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet under a fixed* $\mathsf{Real}^{\uparrow}$ *seal, and let $\{S_i\}_{i\in I}$ be a finite cover of accepted finite RegSeqRat requests over Q. Suppose each S_i carries an accepted terminal-budget read, and suppose every nonempty overlap in the cover satisfies the* $\mathsf{StreamName}^{\uparrow}$ *terminal-fusion finite-cover cocycle and the* $\mathsf{RegSeqRat}^{\uparrow}$ *shared-window triple-overlap cocycle while preserving the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedules, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support rows.*

Then the local terminal-budget reads descend to one public $\mathsf{Real}^{\uparrow}$ *terminal-budget seal row over the same ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The descended seal row exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness principle, arithmetic package, order package, metric-completeness row, standard bridge row, or hidden completion carrier.

Proof. First fuse the finite StreamName windows. Apply Definition 126.628 to the overlap cocycle data; the result is a single finite schedule carrying the same displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows on every member of the cover.

Next identify the RegSeqRat sections over that fused schedule. The hypotheses give the shared-window triple-overlap cocycle, so Definition 301.163 keeps the regularity, classifier, and provenance rows on one finite RegSeqRat section while preserving the DyadicRatCore endpoint, radius, and distance ledgers.

Now paste the Real terminal-budget reads over the finite cover. On each pairwise overlap, Definition 38.716 applies to the identified dyadic, StreamName, RegSeqRat, Real seal, and support rows, so the overlap has a single public $\mathsf{Real}^{\uparrow}$ seal row. Repeating this finite pasting along the fused schedule gives one candidate terminal-budget seal row for the whole cover.

It remains to check that the candidate is the public terminal-budget read and not a hidden completion object. Use Definition 38.715 to keep the pasted read on the displayed ordered route, and then apply Definition 38.713. Exhaustion says the route has only the four visible faces and the listed support rows, so the descent cannot export quotient stream equality, host real equality, selected limits, choice, ambient completeness, metric-completeness, standard bridge rows, or hidden completion carriers. □

Theorem 38.718 (Real finite-window seal exhaustion)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read used by a Real-completion consumer factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{Real}^{\uparrow}$ *seal row. Equivalently, the terminal read is exhausted by the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the visible BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

The read cannot add quotient stream equality, host real equality, selected host limits, ambient completeness, countable-choice selectors, metric-completeness data, or a standard bridge row.

Proof. Project the terminal seal through the existing $\mathsf{Real}^{\uparrow}$ finite-window rows. The name-seal carrier of Definition 38.685 displays the finite-window schedule, dyadic observation, RegSeqRat regularity row, finite ledger, local seal row, and the support rows H, C, P, N.

Apply Definition 38.686. It forces every public seal read to pass through the displayed source route from the $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ observation to the $\mathsf{RegSeqRat}^{\uparrow}$ readback and then to the local $\mathsf{Real}^{\uparrow}$ seal.

Use the sibling routes already present in the hub. The StreamName finite-window face is the public interface of Definition 126.301, the RegSeqRat handoff is Definition 301.22, and the DyadicRatCore tolerance source is Definition 291.18. Together with Definition 38.688, these rows repack the read as the four-face route in the statement.

Finally apply the local non-escape and readback surface supplied by Definition 38.687 and Definition 38.713. Those theorems expose no quotient stream equality, host real equality, selected host limits, ambient completeness, choice selector, metric-completeness row, or standard bridge row, so a terminal Real-completion consumer cannot acquire any of those objects from the finite seal read. □

Theorem 38.719 (Real finite-window seal refinement monotonicity)**.** *Let* $\mathcal{R}$ *be an accepted public* $\mathsf{Real}^{\uparrow}$ *terminal seal read whose finite boundary factors through the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Suppose $\mathcal{R}'$ *is a second finite request whose* $\mathsf{StreamName}^{\uparrow}$ *window refines the displayed window of* $\mathcal{R}$*, while preserving the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{R}'$ *is again an accepted finite-window* $\mathsf{Real}^{\uparrow}$ *seal read. Moreover,* $\mathcal{R}'$ *exports no coordinate outside the same four-face packet and the displayed support rows.*

Proof. Start with the accepted read $\mathcal{R}$. By Definition 38.718, its public terminal seal surface is exhausted by the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Now restrict the refined request to the same four-face cover. The cover normal form of Definition 38.819 says that finite refinements of the displayed $\mathsf{StreamName}^{\uparrow}$ windows remain in the same terminal seal packet when the dyadic ledger, RegSeqRat classifier and readback rows, Real seal row, and support rows are preserved.

Next apply terminality from Definition 126.504. Since the refined request exposes exactly the same four faces and the same support rows after restriction, it factors through the terminal finite BEDC pullback available to Real-seal-facing consumers.

Finally transport the accepted status along the unchanged support rows using Definition 38.715. Naturality keeps the RegSeqRat budget and local Real seal readback on the same accepted four-face route, so $\mathcal{R}'$ is an accepted finite-window seal read. The non-escape clauses in the cited exhaustion, normal-form, pullback, and naturality results leave no coordinate outside the four faces and the displayed support rows. □

Theorem 38.720 (Real terminal finite-section pullback exactness)**.** *Let $\mathcal{S}$ be a finite section of a terminal* $\mathsf{Real}^{\uparrow}$ *seal read. Then $\mathcal{S}$ is accepted exactly when its visible pullback is the ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the terminal scale-window packet, the* $\mathsf{StreamName}^{\uparrow}$ *face is the tail-compatible finite-window request, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the common-tail finite-request readback surface, and the* $\mathsf{Real}^{\uparrow}$ *face is the local terminal seal row.*

Thus a finite section is accepted by the terminal seal precisely through this visible BEDC pullback. It has no accepted coordinate for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge rows, or hidden completion carriers.

Proof. For the forward direction, start with an accepted finite section $\mathcal{S}$. By Definition 38.718, the terminal seal read is exhausted by the four displayed faces and by the BHist, hsame, Cont, Pkg, and NameCert support rows.

Refine the section inside that cover. The refinement monotonicity theorem Definition 38.719 keeps any visible finite refinement on the same four-face route, so the accepted section cannot leave the displayed terminal packet while preserving its support rows.

Now identify the three source faces. The StreamName tail-compatible request, Definition 126.542, supplies the finite window row. The RegSeqRat common-tail finite-request pullback, Definition 301.184, identifies the common-tail readback with the same finite Real seal boundary. The DyadicRatCore scale-window monotonicity and terminal fusion theorems, Definition 291.41 and Definition 291.46, force the dyadic entry face onto one terminal scale-window packet.

For the reverse direction, suppose the visible pullback is exactly this ordered four-face packet with the displayed support rows. The StreamName request, RegSeqRat pullback, and DyadicRatCore terminal fusion results assemble the section as a finite route into the local $\mathsf{Real}^{\uparrow}$ seal row. Applying Definition 38.719 to that route gives an accepted finite-window terminal seal section.

The exclusion clauses are inherited from the same cited routes and from Definition 38.718. Quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge rows, and hidden completion carriers are not rows of the pullback, so they cannot be used to accept another finite section of the terminal seal. □

Theorem 38.721 (Real terminal finite-section classifier determinacy)**.** *Let $\mathcal{S}$ and $\mathcal{T}$ be accepted finite sections of terminal* $\mathsf{Real}^{\uparrow}$ *seal reads. Suppose their visible* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces carry the same endpoint, radius, and distance ledger, their* $\mathsf{StreamName}^{\uparrow}$ *faces display the same finite window, their* $\mathsf{RegSeqRat}^{\uparrow}$ *faces display the same common-tail classifier and readback rows, their local* $\mathsf{Real}^{\uparrow}$ *faces display the same terminal seal row, and their support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

are the same displayed rows. Then $\mathcal{S}$ and $\mathcal{T}$ determine the same public $\mathsf{Real}^{\uparrow}$ *classifier route.*

Conversely, every public $\mathsf{Real}^{\uparrow}$ *classifier route accepted at this terminal finite-section boundary is determined by exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail readback surface, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. The route carries no quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, standard bridge row, or hidden completion carrier.*

Proof. First compare two accepted sections. By Definition 38.720, each accepted finite section is represented by the same ordered four-face pullback whenever its dyadic ledger, StreamName window, RegSeqRat common-tail readback, local Real seal row, and displayed support rows agree.

Now refine both representatives to the common finite window. The preservation clause in Definition 38.719 keeps the accepted status and forbids any extra coordinate from entering during this finite-window comparison.

Terminal uniqueness supplies the classifier route. Apply Definition 126.504 to the common $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with the displayed support rows; the terminal pullback admits only one public Real classifier route over those faces.

Finally transport that route along the RegSeqRat budget boundary using Definition 38.715. This gives the same public classifier route for $\mathcal{S}$ and $\mathcal{T}$.

For the converse, start with an accepted public classifier route at the terminal finite-section boundary. Pullback exactness identifies its accepted boundary with the four displayed faces and support rows, and terminality makes that representation unique. Naturality then transports the route back to the RegSeqRat common-tail readback without adding quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, a standard bridge row, or a hidden completion carrier. □

Theorem 38.722 (Real terminal finite-cover seal descent)**.** *Let $\{\mathcal{S}_i\}_{i\in I}$ be a finite family of accepted terminal* $\mathsf{Real}^{\uparrow}$ *finite sections. Suppose the visible* $\mathsf{StreamName}^{\uparrow}$ *windows of the family satisfy the finite-cover pasted source hypotheses, the induced* RegSeqRat *handoff packages consume one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger on every displayed overlap, and the local* $\mathsf{Real}^{\uparrow}$ *seal classifier rows agree on all pairwise overlaps and satisfy the displayed triple-overlap cocycle. Suppose also that the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

restrict compatibly on the same finite cover.

Then the family descends to one public $\mathsf{Real}^{\uparrow}$ *classifier route over the finite union of the windows. This descended route is exhausted by the pasted* $\mathsf{StreamName}^{\uparrow}$ *source surface, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger, the pasted* RegSeqRat *handoff surface, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed support rows. It exports no quotient stream equality, selected limit, countable-choice selector, ambient completeness theorem, metric-completeness row, standard bridge row, or hidden completion carrier.*

Proof. First paste the finite source surface. Apply Definition 126.551 to the certified $\mathsf{StreamName}^{\uparrow}$ windows, their displayed overlaps, their triple-overlap cocycles, and the shared DyadicRatCore finite ledger. This produces one RegSeqRat and finite-window Real source surface over the finite union.

Next paste the seal rows. The overlap agreement of the local $\mathsf{Real}^{\uparrow}$ classifier rows and the triple-overlap seal cocycle are exactly the hypotheses of Definition 126.552, so the local seal classifiers form one finite-window Real seal surface over the same finite union.

Now identify each local section with the terminal four-face packet. Use Definition 38.720 on every accepted $\mathcal{S}_i$. The only visible rows of each local section are the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The shared dyadic ledger is the same finite-tail input consumed by Definition 291.61. That route places the pasted source and pasted seal surfaces on the displayed

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

packet and leaves no auxiliary equality, selected-limit, completeness, or bridge coordinate.

Finally apply Definition 38.721. Since the pasted finite union has one dyadic ledger, one StreamName source surface, one RegSeqRat handoff surface, one local Real seal row, and compatible support rows, the public terminal classifier route is determined uniquely over that finite union. The non-escape clauses of the cited pasted-source, pasted-seal, pullback, determinacy, and finite-tail factorization routes give exactly the exclusions in the statement. □

Theorem 38.723 (RealUp RealCompleteness exit route)**.** *Let S be an accepted* $\mathsf{Real}^{\uparrow}$ *seal consumed as a Phase Real and Completion endpoint, and suppose the endpoint read is supplied by an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N).$$

Every completion-facing read of S factors through exactly the displayed $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the BEDC support rows* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *carried by H, C, P, N.*

The route exports no quotient stream, host real equality, selected external limit, countable-choice row, ambient completeness theorem, standard bridge row, or completion-facing coordinate outside that finite packet.

Proof. Start with the $\mathsf{RealCompleteness}^{\uparrow}$ public seal export. Theorem 450.12 identifies the endpoint consumed by $\mathsf{Real}^{\uparrow}$ and completion-facing readers with the seal row S reached through the displayed rows D, W, Q, H, C, P, N of the accepted diagonal packet.

Next apply the diagonal-limit candidate theorem and the non-escape boundary from chapter 450. The diagonal-limit candidate theorem routes the candidate through the explicit selector, the dyadic ledger, the StreamName windows, the RegSeqRat readback row, and the support rows, while the non-escape boundary rules out quotient streams, selected host limits, countable choice, host real equality, and ambient completeness as exported coordinates.

Now compare this packet with the local $\mathsf{Real}^{\uparrow}$ terminal route. By Definition 38.713, a terminal-budget Real read is exhausted by the ordered four-face packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. By Definition 38.534, the same four faces determine the Phase Real and Completion endpoint route.

Finally use the consumer-side consistency check Definition 646.75. It pins the completion-facing consumer to the same four-face budget surface, so the $\mathsf{RealCompleteness}^{\uparrow}$ diagonal packet and the local $\mathsf{Real}^{\uparrow}$ seal read have one finite exit route. Since all cited boundaries expose only the listed rows, no quotient stream, host equality, selected external limit, countable-choice row, ambient completeness theorem, standard bridge row, or separate completion-facing coordinate can be read from the endpoint. □

Theorem 38.724 (RealCompleteness consumer direction)**.** *Let S be a displayed* $\mathsf{Real}^{\uparrow}$ *seal and let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet consuming that seal as a Phase Real and Completion endpoint. The packet is only a completion-facing consumer route for the already displayed* $\mathsf{Real}^{\uparrow}$ *seal: it agrees with the four-face route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *exactly along the exposed rows D, W, Q, S, H, C, P, N.*

This direction does not assert that every $\mathsf{Real}^{\uparrow}$ *seal carries such a diagonal packet, does not supply* $\mathsf{Real}^{\uparrow}$ *completeness, does not change any face status, and does not add a standard bridge row.*

Proof. First apply Definition 38.723. It identifies any completion-facing read of the displayed seal with the $\mathsf{RealCompleteness}^{\uparrow}$ diagonal packet and with the StreamName, RegSeqRat, DyadicRatCore, and support rows exposed by D, W, Q, S, H, C, P, N.

Next compare that packet with the Phase Real terminal route. Theorem 38.713 exhausts terminal Real reads by the four-face budget over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and Definition 38.534 reads the same four objects as the Phase Real and Completion endpoint route.

Finally use Definition 646.75. It keeps the diagonal-limit consumer on the same four-face budget surface, so the only agreement being recorded is the consumer-side agreement along the displayed rows.

The roadmap two-axis discipline then supplies the refusal of the opposite reading. A consumer route for one exposed diagonal packet is not a quantified existence theorem for all $\mathsf{Real}^{\uparrow}$ seals, not a completeness theorem for $\mathsf{Real}^{\uparrow}$, not a face-status change, and not a standard bridge row. □

Theorem 38.725 (Real L10 single completion packet is not completeness)**.** *Let S be a displayed* $\mathsf{Real}^{\uparrow}$ *seal and let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet consuming that seal as a Phase Real and Completion endpoint. The existence of this one accepted packet gives only the finite four-face consumer route over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *seal, with the displayed rows D, W, Q, S, H, C, P, N and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support.*

It does not imply the quantified claim that every accepted $\mathsf{Real}^{\uparrow}$ *seal has such a packet, does not add a selected-limit operation, and does not certify* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 38.724. It records only the consumer direction for the displayed packet R and the displayed seal S, so its conclusion is a one-packet route rather than a universal existence principle over all accepted $\mathsf{Real}^{\uparrow}$ seals.

Next use Definition 38.732. The admissible endpoint read is exactly the ordered four-face packet through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ seal, together with the local records and shared support rows named in the statement.

The possible substitute readings are excluded by Definition 38.730. Replacing the four-face route by a host equality row, quotient row, selected-limit row, or completeness row leaves no accepted L10 exit route.

Finally apply Definition 38.736. Since completeness and selected external limits are not consequences of the seal handoff, the single accepted $\mathsf{RealCompleteness}^{\uparrow}$ consumer packet cannot certify $\mathsf{Real}^{\uparrow}$ completeness or supply a selected-limit operation. □

Theorem 38.726 (RealCompleteness L10 exit interface exhaustion)**.** *Let S be an accepted* $\mathsf{Real}^{\uparrow}$ *seal consumed by a Phase Real and Completion interface read, and suppose that this read reaches the public* $\mathsf{RealCompleteness}^{\uparrow}$ *seal through an accepted diagonal packet*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N).$$

Then the interface read factors through exactly the four L10 boundary faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The only transport and support rows available along the route are the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by* D, W, Q, S, H, C, P, N.

The interface exports no quotient stream, host real equality, selected external limit, countable-choice row, ambient completeness theorem, metric-completion row, or separate StandardBridge *coordinate.*

Proof. Begin with the exit route of Definition 38.723. It identifies any completion-facing read of S reached through the public $\mathsf{RealCompleteness}^{\uparrow}$ packet with the displayed diagonal rows D, W, Q, S, H, C, P, N and rules out every coordinate not exposed by that finite packet.

Next apply the Real L10 interface-totality theorem, Definition 38.444. This places the same Phase Real and Completion read inside the ordered four-face lattice $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no additional quotient, host-real, selector, choice, arithmetic, order, or metric-completeness interface row.

The RegSeqRat handoff supplies the regular-rational boundary face. By Definition 301.154, the public RealUp consumer is unblocked only by the finite StreamName–RegSeqRat–RealUp scope row and the dyadic tail-budget surface, preserving hsame, Cont, Pkg, and NameCert provenance across the downstream handoff.

Finally use the four-face ownership theorem, Definition 646.75. It identifies the completion-facing consumer boundary with the same StreamName, DyadicRatCore, RegSeqRat, and Real faces and treats H, C, P, N only as shared support rows. Thus the public $\mathsf{RealCompleteness}^{\uparrow}$ seal can be read at the L10 boundary exactly through those four faces and the displayed BEDC support rows, and none of the excluded quotient, host, choice, completeness, metric-completion, or StandardBridge coordinates is exported. □

Theorem 38.727 (RealCompleteness finite diagonal induction)**.** *Let S be an accepted* $\mathsf{Real}^{\uparrow}$ *seal consumed by a Phase Real and Completion interface read, and let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be the accepted $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet that supplies the read. Finite induction over the displayed packet extracts, in order, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance face D, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal face S. At every induction stage the only transport and support rows are* BHist, *componentwise* hsame, Cont *replay,* Pkg *provenance, and the local* NameCert *rows carried by* D, W, Q, S, H, C, P, N.

Consequently the diagonal induction exports no quotient stream, selected host limit, countable choice, ambient completeness, metric-completeness, aggregate formalstatus, aggregate bridge, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. For the base case, read the displayed seal row from Definition 38.723. That theorem identifies the accepted consumer endpoint with the diagonal packet $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$ and keeps all exported coordinates inside the finite route named by the packet.

For the induction step, peel one displayed packet coordinate at a time. The first projection exposes the dyadic tolerance face D; the next exposes the StreamName finite-window face W; the

next exposes the RegSeqRat readback face Q; the terminal projection exposes the local Real seal face S. Each peel preserves the route already extracted because the only transport data available are the componentwise hsame rows, the Cont replay rows, the Pkg provenance rows, and the local NameCert support rows.

The extracted route is then exhausted by Definition 38.726. It identifies the same consumer read with exactly the four L10 boundary faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and it rules out every coordinate not carried by the finite packet.

Finally invoke Definition 38.731. It orders the extracted faces as the dyadic ledger, the finite-window StreamName rows, the regular-rational readback rows, and the terminal Real seal. Since no other row appears in either the induction or the exhaustion theorem, quotient streams, selected host limits, countable choice, ambient completeness, metric-completeness, aggregate formalstatus, aggregate bridge, and $\mathsf{Real}^{\uparrow}$ completeness remain unavailable. □

Theorem 38.728 (Real L10 four-face audit-clean handoff)**.** *Let S be an accepted* $\mathsf{Real}^{\uparrow}$ *seal consumed by a Phase Real and Completion L10 exit packet. The packet is a valid audit-clean handoff only when the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic face, the* $\mathsf{StreamName}^{\uparrow}$ *countable-name face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy face, and the local* $\mathsf{Real}^{\uparrow}$ *seal face are each read through their own closurestatus and formalstatus records. The paper-closure grade of one face never promotes the Lean-verification grade of another face, and a Lean audit token never promotes a paper-closure grade.*

Consequently the handoff exports only the four named faces and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already displayed by the route. It does not certify that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with Definition 38.726. It exhausts the L10 interface by the four boundary faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BEDC support rows.

Read the four faces separately. The dyadic arithmetic face is governed by its own closurestatus and formalstatus record, the StreamName face by its finite-window name record, the RegSeqRat face by its regular-Cauchy handoff record, and the Real face by the local seal record. These records are inputs to the handoff, not consequences of one another.

The no-promotion claim is just the two-axis discipline applied componentwise. A theory-closure row states a paper-side scope boundary, while a formalstatus row states the Lean-side verification grade of the named target. Since the L10 exit theorem only projects the route through the four faces, it has no constructor that turns either axis into the other.

Finally use the off-table exclusions in the exit interface. Completeness is not among the exported rows, so the handoff can be audit-clean as a route while still refusing the statement that $\mathsf{Real}^{\uparrow}$ is complete. □

Theorem 38.729 (Real L10 facewise audit obstruction)**.** *A Phase Real and Completion L10 exit read of an accepted* $\mathsf{Real}^{\uparrow}$ *seal is audit-admissible only as a componentwise four-face product. On the displayed route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic face, the* $\mathsf{StreamName}^{\uparrow}$ *countable-name face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy face, and the local* $\mathsf{Real}^{\uparrow}$ *seal face must each provide its own closurestatus and formalstatus record.*

If any one of these four local face records is absent, the exit packet has no aggregate audit token, fifth face, completeness row, bridge row, or shared BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *support row that can compensate for the missing record.*

Proof. First apply Definition 38.726. It restricts the L10 exit interface to the four boundary faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ and to the displayed BEDC support rows.

Next apply Definition 38.728. The audit-clean handoff reads the four closurestatus and formalstatus records separately: dyadic arithmetic, finite-window name, regular-Cauchy readback, and local Real seal. These records are independent inputs to the route, so omitting one face leaves that component with no local audit witness.

Finally use Definition 38.736. Completeness, selected external limits, countable choice, host equality, quotient construction, and standard bridge data are not consequences of the seal handoff. Since the exhausted interface contains no fifth face and no aggregate replacement token, none of those excluded rows can supply the missing local face record. □

Theorem 38.730 (Real L10 incomplete face exit obstruction)**.** *An accepted Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *exit requires all four source faces displayed by the L10 route: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic face, the* $\mathsf{StreamName}^{\uparrow}$ *countable-name face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy face, and the local* $\mathsf{Real}^{\uparrow}$ *seal face. Each face must appear with its own* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows and with its local closurestatus and formalstatus record.*

If any one of the four faces is absent, replaced by a host equality row, a quotient row, a selected-limit row, or a completeness row, or detached from its local closurestatus and formalstatus record, then the packet has no accepted L10 exit route.

Proof. Use Definition 38.726 to identify the terminal interface. It admits only the four boundary faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BEDC support rows.

Apply Definition 38.728 and Definition 38.729. They make the audit token componentwise: each face is read through its own closurestatus and formalstatus record, and no aggregate row can compensate for a missing local face.

The ordered source route is fixed by Definition 38.731. Therefore deleting or replacing any one source face breaks one of the four route positions rather than producing another admissible terminal path.

Finally apply Definition 38.736. Host equality, quotient construction, selected external limits, completeness, and standard bridge data are off-table for the seal handoff, so none of them supplies the missing face or its local audit record. □

Theorem 38.731 (Real StreamName-RegSeqRat-Dyadic audit route)**.** *Every accepted Phase Real and Completion L10 exit read of a* $\mathsf{Real}^{\uparrow}$ *seal follows the ordered source route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The route begins with the $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic tolerance ledger, exposes the* $\mathsf{StreamName}^{\uparrow}$ *countable-name finite-window rows, passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback rows, and ends at the local* $\mathsf{Real}^{\uparrow}$ *seal. At every step the only shared support rows are* BHist*, componentwise* hsame*,* Cont *replay,* Pkg *provenance, and the local* NameCert *rows.*

Proof. Apply Definition 38.726. It already identifies the L10 exit interface with the four boundary faces and with the support rows carried by D, W, Q, S, H, C, P, N.

The source route is then read by projection. The dyadic row D supplies the arithmetic tolerance face, the StreamName row W supplies the finite-window name face, the RegSeqRat row Q supplies the regular-Cauchy readback face, and the seal row S supplies the local Real face.

The remaining rows are structural. The BHist rows name the source histories, hsame transports componentwise agreement, Cont records replay of the same route, Pkg records provenance, and NameCert records the local certificate surfaces. No further carrier or classifier is used. □

Theorem 38.732 (Real L10 four-face exit criterion)**.** *An accepted Phase Real and Completion endpoint read of a* $\mathsf{Real}^{\uparrow}$ *seal is admissible exactly when it is supplied by the ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Each face must expose its own local closurestatus and formalstatus record, and the only shared route support rows are BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Equivalently, the current L10 exit route accepts precisely the packets whose dyadic arithmetic face, finite-window StreamName face, regular-rational readback face, and local Real seal face all appear in that order with those local records and shared support rows. It accepts no quotient row, host equality row, selected external limit, completeness row, choice row, metric-completion row, or StandardBridge *reading in place of one of the four faces.*

Proof. For the forward direction, begin with Definition 38.726. It restricts every accepted Phase Real and Completion endpoint read to the four L10 boundary faces and to the displayed BEDC support rows.

Apply Definition 38.728 to read the closurestatus and formalstatus records face by face. Then apply Definition 38.731 to order the faces as the dyadic ledger, the StreamName finite-window rows, the RegSeqRat readback rows, and the local Real seal.

For the reverse direction, use Definition 38.534. A packet that supplies exactly the ordered four faces, their local records, and the shared BHist, hsame, Cont, Pkg, and NameCert rows has the terminal route data read by the Phase Real and Completion endpoint.

It remains to exclude substitute readings. The incomplete-face obstruction Definition 38.730 rules out deleting or replacing any one face, and Definition 38.736 rules out quotient construction, host equality, selected limits, completeness, choice, metric-completion, and StandardBridge consequences from the seal handoff. Thus the four-face packet is both necessary and sufficient for the admissible L10 endpoint read. □

Theorem 38.733 (Real L10 four-object phase exit certificate)**.** *A Phase Real and Completion endpoint read of an accepted* $\mathsf{Real}^{\uparrow}$ *seal is accepted exactly as the four-object certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The certificate consists only of the dyadic tolerance ledger, the finite-window StreamName rows, the RegSeqRat readback rows, the local Real seal row, and the shared BEDC support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Thus the current Phase Real and Completion exit surface is precisely the displayed four-object certificate. It contains no quotient stream, host real equality, selected limit, countable-choice row, completeness row, metric-completion row, or StandardBridge *row.*

Proof. First apply Definition 38.726. It exhausts every Phase Real and Completion interface read of the accepted seal by the four L10 boundary faces and the displayed support rows.

Next apply Definition 38.728. It keeps the four boundary faces componentwise audit-clean and prevents a closurestatus row for one face from supplying the formalstatus row of another face.

The source order is fixed by Definition 38.731. The endpoint certificate therefore starts at the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, passes through the $\mathsf{StreamName}^{\uparrow}$ finite-window name rows and the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback rows, and terminates at the local $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 38.732. It gives the exact acceptance criterion for this ordered packet: the packet is admissible exactly when the four faces appear with their local records and the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

The off-table coordinates are excluded by the same criterion together with the exit interface exhaustion. Quotient streams, host real equality, selected limits, countable choice, completeness, metric-completion, and StandardBridge rows are not among the certificate rows, so none can be read from the Phase Real and Completion endpoint. □

Theorem 38.734 (Real L10 four-object phase exit no-collapse)**.** *The four-object L10 phase-exit certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

cannot be reread as a bridge-axis row, an aggregate formalstatus row, a selected-limit operation, a quotient construction, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

It remains exactly the finite BEDC exit certificate whose data are the dyadic tolerance ledger, the finite-window StreamName rows, the RegSeqRat readback rows, the local Real seal row, and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Proof. First apply Definition 38.726. It exhausts the L10 exit interface by the four displayed boundary faces and the stated BEDC support rows, so no fifth carrier or bridge-axis coordinate is available inside the endpoint read.

Next apply Definition 38.728. The handoff keeps the closurestatus and formalstatus records face-local: the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows each supply their own audit coordinate, and no aggregate formalstatus row can replace that componentwise route.

The route order is fixed by Definition 38.731. Hence a selected-limit operation or quotient construction cannot be inserted between the dyadic ledger, finite-window name rows, regular-rational readback rows, and the local Real seal without leaving the certified source route.

Finally use Definition 38.732. It accepts precisely the ordered four-face packet with the shared BHist, hsame, Cont, Pkg, and NameCert rows. The excluded completeness consequences are then exactly those named in Definition 38.736, so the certificate does not collapse into $\mathsf{Real}^{\uparrow}$ completeness or any standard bridge reading. □

Theorem 38.735 (Real L10 terminal route pullback square)**.** *An accepted Phase Real and Completion endpoint read of a* $\mathsf{Real}^{\uparrow}$ *seal is the pullback square whose four boundary maps are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *terminal tolerance exhaustion, the* $\mathsf{StreamName}^{\uparrow}$ *terminal pullback factorization, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-cover audit-threshold gluing, and the local* $\mathsf{Real}^{\uparrow}$ *seal route:*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow},$$
$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow},$$
$$\mathsf{StreamName}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$
$$\mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The square is determined only by the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by those four faces. No quotient stream, selected limit, countable-choice row, metric-completion datum, aggregate formalstatus row, bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem can fill a missing face of the square.*

Proof. First read the dyadic boundary through Definition 291.123. Its terminal tolerance exhaustion supplies the finite dyadic ledger and exposes the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces with only the shared support rows.

Next apply Definition 126.115 and Definition 301.227. The StreamName factorization identifies the upstream terminal route through the StreamName, RegSeqRat, DyadicRatCore, and

Real faces, while the RegSeqRat gluing reads compatible finite-cover audit-threshold data as one terminal section over those faces.

The local endpoint is fixed by Definition 38.732. It accepts exactly the ordered four-face packet and its visible BHist, hsame, Cont, Pkg, and NameCert support rows, so the three upstream maps and the local Real seal route form one pullback square rather than four unrelated claims.

Finally use Definition 38.734. It excludes bridge-axis rows, aggregate formalstatus rows, selected limits, quotient constructions, and completeness theorems from the certificate. Therefore none of the excluded rows can replace a missing boundary map or supply an additional face of the square. □

Theorem 38.736 (RealUp seal handoff has no completeness consequence)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal handoff supplied by the L10 exit route does not imply that* $\mathsf{Real}^{\uparrow}$ *is complete. It also does not imply Dedekind-cut completeness, trichotomy, quotient-by-equivalence-class construction of reals, selected external limits, host real equality, countable choice, or any standard bridge consequence not explicitly present in the four-face route.*

Proof. Use Definition 38.728. The handoff is valid only as a four-face route whose inputs are the displayed closurestatus and formalstatus records. Those inputs name route acceptance and audit status, not analytic completeness.

Next use Definition 38.731. Its data are exactly the DyadicRatCore ledger, StreamName finite-window rows, RegSeqRat readback rows, the local Real seal, and the shared BHist, hsame, Cont, Pkg, and NameCert rows.

The listed completeness and order principles require additional public rows: a completeness carrier, a Dedekind-cut interface, a trichotomy classifier, a quotient construction, a selected limit operator, host equality, choice, or a bridge schema. None appears in the route. Therefore none is projected from the seal handoff. □

38.5 StreamName and RegSeqRat dependency route

Theorem 38.737 (Real seal StreamName-RegSeqRat dependency route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read consumes the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows of chapter 126 before it reads the regularity and seal rows supplied by chapter 301. Thus the real-name carrier is routed through finite scheduled observations and regular rational-source ledgers before any completion-facing consumer can use the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. The Real carrier of Definition 38.1 is specified as a sealed regular rational stream name. Its observations are StreamName rows, and its regularity certificate is exposed through the RegSeqRat source surface.

Project the StreamName surface first. A finite observation request fixes the scheduled window through chapter 126, so the later seal read cannot introduce an unnamed observation stream.

Then project the RegSeqRat surface. The regularity and classifier rows of chapter 301 supply the rational-source ledger consumed by the Real seal, so a completion-facing Real read is downstream of both sibling packets. □

Theorem 38.738 (Real seal DyadicRatCore-StreamName-RegSeqRat dependency route)**.** *Every accepted current-phase* $\mathsf{Real}^{\uparrow}$ *seal read factors through the ordered finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The DyadicRatCore endpoint and radius ledger is read before the StreamName finite-window schedule, the RegSeqRat regularity and classifier rows, and the local Real seal. The route exports no quotient streams, selected limits, host real equality, ambient completeness, metric-completeness rows, countable choice, or Real completeness coordinate.

Proof. Start from the Real carrier of Definition 38.1. The preceding dependency route, Definition 38.737, places every Real seal read downstream of the StreamName finite-window rows and the RegSeqRat regular-rational source rows.

Insert the DyadicRatCore face before those rows. The RegSeqRat bridge handoff exactness of Definition 291.28 says that the RegSeqRat finite-window handoff consumes the displayed DyadicRatCore endpoint and radius ledger, while the StreamName four-object terminal pullback of Definition 126.504 keeps the DyadicRatCore, StreamName, RegSeqRat, and local Real faces in one finite terminal packet.

Finally apply Real L10 finite-window witness determinacy, Definition 38.464. It identifies the public seal boundary with exactly the fused StreamName terminal-window packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Hence the accepted current-phase seal read has the stated ordered route and none of the excluded quotient, selected-limit, host-equality, completeness, metric-completeness, choice, or Real-completeness coordinates. □

Theorem 38.739 (Real L10 four-face route order uniqueness)**.** *Every current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section whose displayed rows are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal row has*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

as its unique BEDC route. Any attempted route omitting or permuting one face either loses one of the displayed rows or introduces an off-table coordinate, namely quotient stream equality, a selected limit, host real equality, ambient completeness, metric-completeness, countable choice, or a second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. The forward order is the route of Definition 38.738. It already reads the DyadicRatCore endpoint and radius ledger before the StreamName finite-window schedule, then the RegSeqRat regularity and classifier rows, and finally the local Real seal.

The order of the first three faces cannot be permuted without dropping a displayed row. The DyadicRatCore handoff exactness of Definition 291.28 makes the endpoint and radius ledger the finite handoff read by RegSeqRat, while the StreamName four-object terminal pullback of Definition 126.504 keeps the DyadicRatCore, StreamName, RegSeqRat, and Real faces in one terminal packet.

The public seal boundary is fixed by Real L10 finite-window witness determinacy, Definition 38.464. RegSeqRat terminal-section exhaustion, Definition 301.94, rules out a fifth-face witness or a shortcut witness between the regular rational section and the Real seal. Therefore every alternative route either omits one of the four displayed faces or leaves the table by adding one of the excluded quotient, selected-limit, host-equality, completeness, choice, or duplicate-seal coordinates. □

Theorem 38.740 (Real L10 four-face classifier coherence)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section whose displayed rows are exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius*

ledger, the $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal row. Then the four visible classifier reads of* S *determine the same finite packet*

$$(\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}).$$

Any classifier disagreement either drops one of the four displayed faces or leaves the table by adding an off-table coordinate: quotient stream equality, a selected limit, host real equality, ambient completeness, metric-completeness, countable choice, or a duplicate $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Start with the ordered route of Definition 38.739. It fixes the only current-phase terminal order as $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a visible classifier comparison must be read on those four faces and on no independent route.

The endpoint and radius classifier compatibility comes from Definition 291.28. The finite-window packet shared by the four faces is supplied by Definition 126.504. Together these two results place the DyadicRatCore ledger, StreamName schedule, RegSeqRat classifier rows, and local Real seal row on one finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support packet.

It remains to exclude extra classifier sources. RegSeqRat terminal-section exhaustion, Definition 301.94, rules out a shortcut from the regular rational section to the Real seal and rules out a fifth classifier face. Real L10 finite-window witness determinacy, Definition 38.464, then fixes the public seal boundary to the same finite-window witness. Hence any classifier mismatch is visible as the loss of one of the four displayed faces, or else it requires one of the listed off-table coordinates. □

Theorem 38.741 (Real L10 route face deletion obstruction)**.** *Let* S *be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section with public seal boundary displayed on*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Then no route presenting the same public boundary may delete any one of these four faces. Deleting $\mathsf{DyadicRatCore}^{\uparrow}$ *loses the endpoint and radius ledger rows, deleting* $\mathsf{StreamName}^{\uparrow}$ *loses the finite-window schedule, deleting* $\mathsf{RegSeqRat}^{\uparrow}$ *loses the regularity and classifier rows, and deleting* $\mathsf{Real}^{\uparrow}$ *loses the seal row itself. Any purported route retaining the boundary after such a deletion must leave the permitted current-phase surface by adding quotient stream equality, a selected limit, host real equality, ambient completeness, metric-completeness, countable choice, or a duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. The accepted boundary first has the ordered route of Definition 38.738. Its unique current-phase order is fixed by Definition 38.739, so any route with the same public boundary must read the four displayed faces in the same finite order.

Now compare the four deletions. Without $\mathsf{DyadicRatCore}^{\uparrow}$ the route has no endpoint and radius ledger feeding the rational source rows. Without $\mathsf{StreamName}^{\uparrow}$ it has no finite-window schedule. Without $\mathsf{RegSeqRat}^{\uparrow}$ it has no regularity and classifier surface between the scheduled observations and the seal. Without $\mathsf{Real}^{\uparrow}$ it has no public seal row.

The terminal packet cannot hide these losses in another current-phase coordinate. The StreamName terminal pullback of Definition 126.504 keeps the four faces in one finite terminal packet, while Definition 301.94 rules out a shortcut or fifth face between the regular rational section and the Real seal.

Finally apply the classifier coherence theorem, Definition 38.740. It says the visible classifier reads determine the same finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ packet exactly on these faces; hence a route that claims the same boundary after deleting one face must supply the missing row by one of the excluded off-table coordinates. □

Theorem 38.742 (Real L10 four-face local-threshold descent)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet displayed exactly on the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. If $\mathcal{E}$ satisfies the current-phase local threshold, then every accepted projection of the four-face route carries the same objectwise local threshold records on the DyadicRatCore endpoint and radius ledger, the StreamName finite-window schedule, the RegSeqRat classifier and regularity rows, and the local Real seal row.*

Conversely, any accepted four-face route projection carrying those four local threshold records together with the same displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support repacks as the current-phase Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate of Definition 38.796. The descent and repacking use no quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First put all classifier reads on one finite packet. By Definition 38.740, the visible classifier coordinates of an accepted route projection over the four displayed faces determine the same finite BHist, hsame, Cont, Pkg, and NameCert support packet.

For descent, apply Definition 38.796. It identifies the current Phase Real and Completion exit criterion with the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, supported by exactly those finite rows. Hence every accepted projection of that same packet reads the same four local threshold records.

The projection order does not add a second certificate. Use Definition 38.739 to fix the terminal route and use Definition 38.801 to compare support-row-preserving projections of the displayed packet. Thus the descended local threshold is independent of the accepted projection order.

For the converse direction, start with an accepted four-face route projection carrying the four local threshold records and the same displayed support rows. The local-threshold theorem repacks these rows as the current-phase exit certificate, and the classifier coherence theorem keeps the repacked certificate on the same finite four-face packet.

Finally apply Definition 38.793. Each face record and each displayed support row is indispensable, so no shortened route or off-table coordinate can present the same exit certificate. Therefore the repacking uses exactly the stated four faces and excludes the listed quotient, selected-limit, host-equality, choice, metric-completeness, aggregate-status, aggregate-bridge, standard-bridge, and completeness data. □

Theorem 38.743 (Real L10 tail-budget projection nonescape)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section whose public seal is read by a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget consumer. Then every support-row-preserving projection of that consumer remains on the displayed four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The projection preserves the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, and it may read only the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat tail-budget classifier and regularity rows, and the terminal Real seal. No such projection may introduce quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or a* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First lock the tail-budget consumer to its source route. Apply Definition 301.112. The consumer can read the Real seal only through the StreamName schedule, the DyadicRatCore tolerance row, the RegSeqRat classifier, and the terminal $\mathsf{Real}^{\uparrow}$ seal. Thus the read has no source for quotient stream equality, selected limits, host real equality, choice, completeness data, or a separate Real completeness coordinate.

Next identify the local threshold records on the same four faces. Apply Definition 38.742. The accepted projection is therefore read objectwise on the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier and regularity rows, and the local Real seal row, while preserving the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support packet.

It remains to exclude alternative projections of that packet. Use Definition 38.801 to compare support-row-preserving reads of the displayed route, and then apply Definition 38.793. The first result prevents a reordered projection from changing the certificate, and the second makes every listed face and support row indispensable. Hence a projection that is shortened or enlarged cannot present the same current-phase Real seal read, so the only surviving route is the displayed four-face route with exactly the support rows named in the statement. □

Theorem 38.744 (Real L10 four-face support exhaustion)**.** *Every accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section on the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

has exactly the visible support inventory consisting of the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the terminal Real seal, and the shared BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Conversely, that inventory repacks as the same current-phase Phase $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate. It contains no quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First read the objectwise threshold rows. Apply Definition 38.742: the accepted terminal section carries the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier and regularity rows, and the local Real seal row over the common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support packet.

Next lock that packet to the route. By Definition 38.743, every support-row-preserving projection remains on

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The route therefore has no available coordinate outside the four construction faces and the displayed support rows.

Now identify the endpoint and schedule inventories. Apply Definition 291.118 to read the DyadicRatCore endpoint and tolerance ledger, and apply Definition 126.657 to read the StreamName-side finite-window support surface. These two inventories sit inside the same RegSeqRat classifier, regularity, and transport rows and the same terminal Real seal.

For the converse, use Definition 38.793. The listed four face records and the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows are indispensable, so the inventory repacks as the same current-phase Phase $\mathsf{Real}^{\uparrow}$ and Completion exit certificate. The same minimality statement excludes quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.745 (Real L10 four-face exit criterion equivalence)**.** *An accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section satisfies the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit criterion if and only if the same finite support packet exposes exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Equivalently, the exit criterion is the four-face packet

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with no quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or separate $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. For the forward direction, apply Definition 38.744. It identifies every accepted terminal seal section on the displayed route with the visible inventory consisting of the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the terminal Real seal, and the shared BHist, hsame, Cont, Pkg, and NameCert rows.

For the reverse direction, first read the objectwise exit ledger through Definition 291.110. Then use Definition 126.657 for the finite-window schedule and support surface, Definition 301.239 for the RegSeqRat terminal classifier and regularity rows, and Definition 301.246 for the finite-cover normal form joining the four faces.

These ingredients repack the displayed support inventory as the current-phase Phase $\mathsf{Real}^{\uparrow}$ and Completion exit certificate. Finally, Definition 38.746 separates that certificate from aggregate formalstatus, leantarget, bridge, metric-completeness, selected-limit, quotient-stream, and separate $\mathsf{Real}^{\uparrow}$ completeness coordinates, so the equivalence has exactly the exclusion list stated above. □

Theorem 38.746 (Real L10 dependency route axis independence)**.** *For any accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section on the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

the dependency route determines only the construction faces, the classifier faces, and the support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It does not determine or aggregate any* formalstatus*,* leantarget*, or* bridgestatus *row, nor any standard bridge row, metric-completeness row, selected limit, quotient stream, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First identify the route and the support packet. The dependency theorem Definition 38.738 fixes the ordered four-object route, and Definition 38.742 reads the accepted local-threshold records on exactly those four faces with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next rule out escape through tail-budget projection. Apply Definition 38.743. Any support-row-preserving consumer remains on the same four-face route and has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or a $\mathsf{Real}^{\uparrow}$ completeness row.

Finally separate the route data from the verification and bridge axes. Apply Definition 38.833 and Definition 38.834. These keep status, Lean-target, and bridge-status rows as face-local reports rather than route-level credentials. Therefore the route supplies construction and classifier evidence with its finite support rows, but it supplies no aggregate verification token, bridge credential, metric-completeness claim, selected limit, quotient stream, or $\mathsf{Real}^{\uparrow}$ completeness claim. □

Theorem 38.747 (Real L10 completeness-coordinate refusal)**.** *Every accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section satisfying the four-face Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit criterion has no BEDC coordinate for Real completeness, metric completeness, selected limits, quotient stream equality, host real equality, countable choice, aggregate* formalstatus*, or aggregate* bridgestatus*.*

Equivalently, any assertion of one of these coordinates is not a refinement of the current L10 exit surface. It leaves the finite support packet

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and is therefore a claim outside the accepted current-phase terminal seal.

Proof. First apply Definition 38.745. It identifies the exit criterion with the finite packet whose only construction faces are the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance ledger, the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier, regularity, and transport rows, the local $\mathsf{Real}^{\uparrow}$ seal row, and the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 38.744. The visible inventory of an accepted terminal section is exhausted by those four faces and support rows. Thus there is no remaining BEDC coordinate in the packet for quotient stream equality, selected limits, host real equality, countable choice, metric completeness, aggregate status, aggregate bridge data, or separate $\mathsf{Real}^{\uparrow}$ completeness.

It remains to rule out the same coordinates after projection. Use Definition 38.743. Any support-row-preserving consumer remains on the displayed route and cannot manufacture a new completion, limit-selection, quotient, host-equality, choice, status, bridge, or Real-completeness coordinate.

Finally apply Definition 38.746. The construction route is independent of the verification and bridge axes, so aggregate formalstatus and bridgestatus rows cannot be recovered from the terminal seal. Hence any assertion of one of the listed coordinates leaves the current L10 exit surface rather than strengthening it. □

Theorem 38.748 (Real L10 four-face exit certificate rigidification)**.** *An accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section is a Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate if and only if its visible packet is the four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the terminal Real seal, and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Any presentation with the same exit certificate preserves those four faces and support rows. It cannot add quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge data, standard bridge data, or a separate $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First identify the certificate. Apply Definition 38.745. It says that the Phase $\mathsf{Real}^{\uparrow}$ and Completion exit criterion is exactly the finite packet displayed on the four construction faces with the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

Next exhaust the support. Apply Definition 38.744. The accepted terminal seal section has no visible support beyond the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the terminal Real seal, and the named shared rows.

Now rigidify alternative presentations of the same certificate. Use Definition 38.739, Definition 38.740, and Definition 38.793. These results keep the route order, classifier packet, and indispensable face records fixed, so a second presentation cannot delete, permute, duplicate, or replace one of the displayed faces while preserving the same exit certificate.

Finally apply Definition 38.746 and Definition 38.747. The route supplies construction and classifier evidence only, and the current L10 exit surface has no coordinate for quotient streams, selected limits, host real equality, countable choice, metric completeness, aggregate status, aggregate bridge data, standard bridge data, or separate $\mathsf{Real}^{\uparrow}$ completeness. Hence every presentation of the same exit certificate is rigidly the displayed four-face packet with the named support rows. □

Theorem 38.749 (Real L10 facewise projection gluing)**.** *Let an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section carry four facewise projections on*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume the projections preserve the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the local Real seal row, and the shared BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. Then their common read glues to the same Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate.*

The glued certificate has no quotient stream, selected limit, host equality, countable choice, metric-completeness data, aggregate status, aggregate bridge data, standard bridge data, or separate $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First read the facewise data through Definition 38.745. Since each projection preserves the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, and the local Real seal row over the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, each face presents the finite packet required by the Phase $\mathsf{Real}^{\uparrow}$ and Completion exit criterion.

Next identify the common support. Apply Definition 38.744. The accepted terminal seal has no visible construction support outside the four displayed faces and the named support rows, so the four preserved reads cannot require an additional completion, quotient, equality, choice, status, or bridge coordinate.

Now align the classifier and route data. Use Definition 38.740 to keep the RegSeqRat classifier, regularity, and transport rows coherent across the four projections. Then use Definition 38.793 to repack the preserved endpoint, schedule, classifier, transport, and seal rows as one indispensable four-face certificate rather than four unrelated fragments.

Finally exclude escape coordinates. Apply Definition 38.746 to separate the construction route from aggregate status, Lean-target, and bridge rows, and apply Definition 38.747 to rule out Real completeness, metric completeness, selected limits, quotient streams, host equality, and countable choice. Therefore the common read of the four support-preserving projections glues to exactly the same current-phase Phase $\mathsf{Real}^{\uparrow}$ and Completion exit certificate, with no extra coordinate beyond the displayed packet. □

Theorem 38.750 (Real terminal seal machine-route exhaustion)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section on*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the machine-facing support of S is exhausted by exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real

seal handoff. These three touchpoints consume the same four-face certificate glued from the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the terminal Real seal, and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

No machine-facing route for S may add quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or a $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First glue the Real-side certificate. Apply Definition 38.749. The four support-preserving projections on the displayed route have a common read as the same Phase $\mathsf{Real}^{\uparrow}$ and Completion exit certificate, and the glued certificate already excludes quotient streams, selected limits, host equality, countable choice, metric-completeness data, aggregate status, aggregate bridge data, standard bridge data, and separate $\mathsf{Real}^{\uparrow}$ completeness.

Next identify the certificate with the local exit criterion. Apply Definition 38.745. The glued Real certificate is exactly the finite packet consisting of the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the local Real seal row, and the shared BHist, hsame, Cont, Pkg, and NameCert support.

Now bind that finite packet to the machine route. Use Definition 126.452 and Definition 126.453. These identify the machine-facing route of the same four-face exit read with exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

Finally apply Definition 291.113. The DyadicRatCore-side ledger theorem says that an accepted four-face Phase Real and Completion packet has no machine support outside those three route sites. Hence the terminal $\mathsf{Real}^{\uparrow}$ seal section consumes precisely the three named touchpoints, while all quotient-stream, selected-limit, host-equality, choice, metric-completeness, aggregate-status, aggregate-bridge, standard-bridge, and $\mathsf{Real}^{\uparrow}$ completeness coordinates remain outside the accepted current-phase route. □

Theorem 38.751 (Real four-face machine touchpoint minimality)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section on*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then S is machine-admissible for the Phase $\mathsf{Real}^{\uparrow}$ *and Completion exit if and only if the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff read the same four-face certificate over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Moreover these three touchpoints are a minimal basis for the accepted machine-facing route: deleting any one of them fails to present the same accepted route from the DyadicRatCore endpoint and tolerance ledger through the StreamName finite-window schedule and the RegSeqRat classifier, regularity, and transport rows to the terminal Real seal.

Proof. For the forward direction, apply Definition 38.750. The theorem identifies the machine-facing support of S with exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff, all consuming the same four-face certificate over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

For the reverse direction, use Definition 38.745. A common read by the three named machine touchpoints exposes precisely the finite support packet for the Phase $\mathsf{Real}^{\uparrow}$ and Completion exit

criterion. Then apply Definition 126.453 and Definition 291.113: the StreamName boundary theorem places the read on the same machine-facing route, and the DyadicRatCore completeness theorem rules out any additional machine support site.

It remains to prove minimality. The StreamName singleton finite-window handoff is required by Definition 126.452; without it the finite-window schedule is absent from the machine read. The DyadicRatCore RegSeqRat bridge handoff exactness is required by Definition 291.113; without it the endpoint and tolerance ledger do not reach the RegSeqRat classifier and transport rows. The RegSeqRat Real seal handoff is required by Definition 126.453; without it the route has no terminal $\mathsf{Real}^{\uparrow}$ seal presentation.

Finally apply Definition 38.741. Deleting any indispensable construction face or support row from the four-face packet fails to present the same accepted exit certificate. Therefore deleting any one of the three machine touchpoints fails to present the same accepted machine-facing route, and the three touchpoints form the minimal sufficient basis for admissibility. □

Theorem 38.752 (Real L10 finite bundle induction)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section over the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Let $\mathcal{B}$ be a finite ProbeBundle *of certified unary observations for S. Assume that the rows of $\mathcal{B}$ expose exactly the DyadicRatCore endpoint and tolerance ledger, the StreamName finite-window schedule, the RegSeqRat classifier, regularity, and transport rows, the local Real seal row, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Then induction over the rows of $\mathcal{B}$ determines a unique terminal four-face audit packet for S, and that packet is precisely the packet consumed by the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff of Definition 38.751. The induced packet contains no quotient stream equality, selected limit, host real equality, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the empty prefix of $\mathcal{B}$. The visible support is the fixed ordered route of Definition 38.739, so the prefix has no alternative route on which to place a hidden quotient, selected-limit, host-equality, choice, completeness, status, or bridge coordinate.

For the induction step, add one certified unary observation while preserving the same visible support rows. Apply Definition 38.740: the new observation must read the same finite BHist, hsame, Cont, Pkg, and NameCert packet as the earlier rows, and its RegSeqRat classifier, regularity, and transport data cannot split away from the DyadicRatCore ledger, StreamName schedule, or local Real seal.

It remains to rule out a step that changes the terminal packet by deleting or replacing a face. Use Definition 38.741. Any such change loses one of the four displayed construction faces or supplies the missing row from an off-table coordinate, which contradicts the hypothesis that $\mathcal{B}$ exposes exactly the displayed ledger, schedule, classifier, regularity, transport, seal, and shared support rows.

After the finite induction closes, identify the resulting packet with the machine-facing route. Apply Definition 126.453 and Definition 291.113. These boundary theorems place the induced four-face packet on exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

Finally invoke Definition 38.751. Since those three touchpoints are the minimal basis for the accepted route, the packet determined by the finite bundle is the unique terminal four-face audit

packet consumed by them, and the excluded quotient-stream, selected-limit, host-equality, choice, metric-completeness, aggregate-status, aggregate-bridge, standard-bridge, and $\mathsf{Real}^{\uparrow}$ completeness coordinates remain outside the accepted current-phase surface. □

Theorem 38.753 (Real L10 machine-audit locality)**.** *Let S be an accepted current-phase terminal* $\mathsf{Real}^{\uparrow}$ *seal section on*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff read the same four-face certificate named in Definition 38.751. Then every audit-facing coordinate of that machine read is exactly one of the four local face records: the local closurestatus, formalstatus, leantarget, and bridgestatus records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal face.*

Conversely, those four local objectwise audit records together with the three named machine touchpoints present the same machine-audit route for S. The route contains no aggregate formalstatus row, route-level leantarget, aggregate bridge row, standard bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First identify the machine route. Apply Definition 38.750 and Definition 38.751. The accepted terminal seal has exactly the three machine touchpoints in the statement, and those touchpoints consume one common four-face certificate over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Next read the audit records objectwise. Use Definition 38.834 and Definition 38.815. These keep closurestatus, formalstatus, leantarget, and bridgestatus attached to the four local faces, so the machine read cannot fuse them into a route-level verification token or replace a local face record by an aggregate bridge credential.

Now separate the gateway inventory from the touchpoint inventory. Apply Definition 126.491 and Definition 126.489. The gateway audit records are the four objectwise face records, while the machine-facing route factors through the three named handoff positions. Thus a touchpoint reports its route position only by reading those local records; it does not create an additional audit-facing coordinate.

For the converse direction, start with the four local closurestatus, formalstatus, leantarget, and bridgestatus records together with the three touchpoints. The gateway factorization theorem repacks them as the same four-face machine read, and the terminal seal exhaustion theorem identifies that read with the accepted route for S.

Finally close the exclusions. Use Definition 126.452, Definition 126.453, and Definition 291.113. These boundary results exhaust the current machine support by the three named touchpoints and leave no coordinate for aggregate formalstatus, route-level leantarget, aggregate bridge data, standard bridge data, quotient stream equality, selected limits, countable choice, metric completeness, or separate $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.754 (Real uniform Cauchy completion route lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. If the consumer uses U to reach a* $\mathsf{Real}^{\uparrow}$ *seal or a completion-facing endpoint, then its route factors through*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

together with the $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, and the local* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 38.442.*

The admitted route exposes only the finite packet rows $I, W, M, Q, T, E, H, C, P, N$ *and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support used by those sibling rows. It cannot export quotient streams, selected host limits, host real equality, countable choice, open-cover compactness, ambient completeness, or an additional completion carrier.*

Proof. Begin with the UniformCauchyCriterion finite-family seal obligation surface. Definition 435.81 and Definition 435.88 display the rows $I, W, M, Q, T, E, H, C, P, N$ and force the route from the finite index ledger to the Real seal handoff through the single chain $I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$.

Now attach the completion-budget rows. The real completion budget lock Definition 435.131 places any $\mathsf{RealCauchyCompletion}^{\uparrow}$ read after the same budget package, while the completion-functor budget handoff Definition 435.129 keeps functor-facing access inside the finite package and the regular shared-tail sibling.

Next identify the Real-side dependency lattice. The L10 dependency lattice Definition 38.442, the source-route non-escape theorem Definition 38.443, and the terminal source lock Definition 38.459 route every Phase Real or Completion read through the StreamName, DyadicRatCore, RegSeqRat, and local Real seal rows, with only BHist, hsame, Cont, Pkg, and NameCert support.

Finally compose the two boundaries with the UniformCauchyCriterion finite budget envelope Definition 435.140. The UniformCauchyCriterion side supplies the ordered budget chain, and the Real L10 side supplies the four sibling rows consumed by RealUp. Since the cited non-escape and budget-lock theorems expose no quotient stream, selected host limit, host equality, countable-choice row, open-cover compactness, ambient completeness, or extra completion carrier, the Real-and-Completion consumer has no route to export any of them. □

Theorem 38.755 (Real uniform Cauchy four-object terminal determinacy)**.** *Let*

$$U_0 = (I_0, W_0, M_0, Q_0, T_0, E_0, H_0, C_0, P_0, N_0)$$

and

$$U_1 = (I_1, W_1, M_1, Q_1, T_1, E_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packets read by Phase Real and Completion consumers. Suppose the two reads expose the same finite family ledger, shared* $\mathsf{StreamName}^{\uparrow}$ *window family, modulus schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, tail ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Suppose also that their completion-facing boundaries expose the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Then the two completion-facing reads factor through the same terminal finite BEDC pullback.

Conversely, every accepted uniform-Cauchy completion-facing read whose public surface is the finite budget

$$I, W, M, Q, T, E, H, C, P, N$$

together with those four faces is determined by that finite budget and by the four boundary faces. The resulting terminal read exports no quotient stream family, no host real equality, no selected family limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. First lock the finite uniform-Cauchy route. By Definition 38.754, any Phase Real and Completion consumer of an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet must pass through the ordered rows $I, W, M, Q, T, E, H, C, P, N$ before it reaches either a $\mathsf{Real}^{\uparrow}$ seal or a completion-facing endpoint.

Next apply the finite-family Real seal route determinacy theorem, Definition 435.22. Since the two reads expose the same family ledger, window family, modulus schedule, dyadic tolerance row, and tail ledger, they reach the same finite route through the seal rows E, H, C, P, N.

Now attach the completion boundary. The seal-completion boundary Definition 435.18 keeps every completion-facing consumer on the same finite public surface, and Definition 126.500 identifies the Real-and-Completion exit packet with the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

With those faces fixed, use Definition 126.504. The terminal pullback theorem identifies any two accepted reads that expose the same four faces and the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows. Hence the two completion-facing reads determine the same terminal finite BEDC pullback.

For the converse direction, start with an accepted uniform-Cauchy completion-facing read whose public surface is exactly the finite budget and the four boundary faces. The route-lock theorem supplies the ordered UniformCauchyCriterion budget; the seal-completion boundary places the read at the Real seal boundary; the four-object exit route reads that boundary as the StreamName, DyadicRatCore, RegSeqRat, and RealUp packet. Terminal pullback then repackages the accepted read as the unique finite BEDC pullback determined by those rows.

The exclusion clause is inherited from the same citations. None of the locked budget rows or four boundary faces carries a coordinate for quotient stream families, host real equality, selected family limits, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows, so none of that data can distinguish two reads after the finite budget and the four faces agree. □

Theorem 38.756 (Real uniform Cauchy StreamName RegSeqRat terminal route correspondence)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. Suppose the finite budget route through*

$$I, W, M, Q, T, E, H, C, P, N$$

and the associated boundary rows expose the same $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows as a StreamName finite-request section and as a RegSeqRat shared-window selector surface. Then the finite budget route determines the StreamName section and the RegSeqRat selector surface.*

Conversely, the StreamName finite-request section together with the RegSeqRat shared-window selector surface determines the same accepted uniform-Cauchy completion-facing route through $I, W, M, Q, T, E, H, C, P, N$ *whenever the displayed rows named above agree. Thus the Phase Real and Completion handoff is one finite terminal BEDC surface, not three independent routes.*

The correspondence exports no quotient stream, no host real equality, no selected limit, no ambient completeness row, no countable-choice row, no metric-completeness row, and no standard bridge row.

Proof. First lock the uniform-Cauchy budget chain. By Definition 38.754, an accepted Phase Real and Completion read reaches the Real seal and the completion-facing endpoint only through the finite rows $I, W, M, Q, T, E, H, C, P, N$ and the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Now recover the StreamName section from the same rows. Apply Definition 126.577. Since the uniform-Cauchy route displays the StreamName schedule, the DyadicRatCore endpoint-radius ledger, the RegSeqRat regularity and readback surface, and the support rows, the RealUp-facing finite request section has no coordinate beyond the rows already carried by the budget route.

Next recover the RegSeqRat selector surface. Use Definition 301.158 to identify matching shared-window selector surfaces, and use Definition 301.159 to pass between selector-surface equality and the terminal RealUp seal read on the four-face packet. These two results place the selector on the same StreamName, DyadicRatCore, RegSeqRat, RealUp, and support rows.

For the converse direction, start from the StreamName section and the RegSeqRat selector surface with the displayed rows fixed. The pullback theorem supplies the finite section read, the selector determinacy supplies the shared-window surface, and the terminal pullback equivalence returns the same terminal seal coordinate. Applying Definition 38.755 then repacks those rows as the unique finite uniform-Cauchy completion-facing route through $I, W, M, Q, T, E, H, C, P, N$.

The non-escape clause is inherited from the same four inputs. The route lock, the StreamName pullback, the RegSeqRat selector determinacy, and the four-object terminal determinacy expose no quotient stream, host real equality, selected limit, ambient completeness row, countable-choice row, metric-completeness row, or standard bridge row, so none of those objects can separate the corresponding terminal surface. □

Theorem 38.757 (Real uniform Cauchy four-face status exactness)**.** *Let $U = (I, W, M, Q, T, E, H, C, P, N)$ be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet read by a Phase Real and Completion consumer. Suppose the public surface of the read is the finite budget $I, W, M, Q, T, E, H, C, P, N$ together with the four boundary faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Then the read is status-exact precisely when the $\mathsf{StreamName}^{\uparrow}$ face is read from its own closurestatus and formalstatus record, the $\mathsf{DyadicRatCore}^{\uparrow}$ face is read from its own closurestatus and formalstatus record, the $\mathsf{RegSeqRat}^{\uparrow}$ face is read from its own closurestatus and formalstatus record, and the $\mathsf{Real}^{\uparrow}$ face is read from its own closurestatus and formalstatus record.

No aggregate formal-status token, fifth completion object, quotient stream, selected limit, countable-choice row, metric-completeness datum, or standard bridge row can replace any one of the four face records. In particular, a completion-facing packet whose finite budget is accepted but whose evidence collapses two of the four status records into a single aggregate token is not a status-exact Real-and-Completion read.

Proof. First lock the ordered budget route. By Definition 38.754, every accepted completion-facing read using U reaches the Real seal boundary only through the displayed finite budget chain and the four Real L10 owners.

Next apply terminal determinacy over the four faces. The theorem Definition 38.755 identifies the completion-facing read with the finite terminal pullback determined by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

The status part is objectwise. The roadmap two-axis discipline records closure and verification independently at each object, so the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces

each contribute their own closurestatus and formalstatus rows to the pullback. Reading one face's record gives no coordinate for another face's record.

Therefore status exactness is equivalent to exposing all four objectwise status records. The route lock and terminal pullback expose no aggregate formal-status token, fifth completion object, quotient stream, selected limit, countable-choice row, metric-completeness datum, or standard bridge row, so none of those objects can fill a missing face. □

Theorem 38.758 (Real uniform Cauchy four-face record minimality)**.** *Let* $U = (I, W, M, Q, T, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. Suppose the accepted L10 exit reaches the completion-facing* $\mathsf{Real}^{\uparrow}$ *seal through exactly the four local faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Then the four local closurestatus and formalstatus records form a minimal facewise audit family for status exactness. No proper subfamily of these four records is sufficient for the status-exact L10 exit read.

In particular, identifying two face records, replacing one face record by an aggregate formalstatus token, or replacing one face record by route-level bridge prose destroys the accepted Real-and-Completion exit read. The audit boundary must retain one local status record for StreamName, one for DyadicRatCore, one for RegSeqRat, and one for the local Real seal.

Proof. Apply Definition 38.757. It identifies status exactness with the four objectwise closurestatus and formalstatus records and rejects aggregate formal-status tokens, fifth completion objects, quotient streams, selected limits, and bridge rows as replacements.

Next apply Definition 38.755. It fixes the terminal completion-facing pullback on the four displayed faces, so a proper subfamily leaves at least one terminal face without its status coordinate.

The separation of records is not route-level rhetoric. By Definition 38.833, the four local reports cannot be fused into a new formalstatus token, aggregate bridge row, or Real completeness claim.

Finally apply Definition 38.730. If any one face is absent, identified with another face, or detached from its local status record, the packet has no accepted L10 exit route. Thus all four independent face records are minimal data for the status-exact Real-and-Completion handoff. □

Theorem 38.759 (Real RegSeqRat–StreamName audit-clean handoff)**.** *Every Phase Real and Completion consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal through an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet must pass through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, and the local* $\mathsf{Real}^{\uparrow}$ *seal before it can claim an audit-clean L10 handoff.*

Equivalently, the handoff is readable only through the displayed UniformCauchyCriterion rows, the four Real L10 row owners of Definition 38.442, and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Quotient streams, selected host limits, host real equality, countable choice, and ambient completion have no coordinate in the audit-clean handoff route.*

Proof. Start with the route lock of Definition 38.754. It already forces any Phase Real and Completion read through the finite UniformCauchyCriterion chain before a $\mathsf{Real}^{\uparrow}$ seal or completion-facing endpoint is exposed.

Then read the Real-side lattice. The dependency lattice of Definition 38.442 names the four row owners used by the route: $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The exclusion boundary is inherited from the Real L10 source theorems. Definition 38.443 removes quotient streams, selected host limits, host real equality, countable choice, and ambient

completion from the source route, while Definition 38.459 makes the Real seal terminal over the displayed source rows.

Thus an audit-clean handoff cannot be claimed before the consumer has exposed the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, and RealUp local seal. Since none of the excluded host objects occurs as a row in that route, none can supply an alternate handoff coordinate. □

Theorem 38.760 (Real uniform Cauchy budget pullback exhaustion)**.** *Let* $U = (I, W, M, Q, T, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. The consumer reaches a public* $\mathsf{Real}^{\uparrow}$ *exit if and only if its finite budget chain*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

pulls back to exactly the four L10 row owners: the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback and classifier row, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal.*

This pullback exhausts the public exit surface. No quotient stream, selected host limit, countable-choice selector, ambient-completeness row, or hidden completion carrier supplies a fifth owner or an alternate Real exit.

Proof. First take the forward direction. By Definition 38.754, any consumer that uses U to reach a $\mathsf{Real}^{\uparrow}$ seal or completion-facing endpoint has already passed through the finite UniformCauchyCriterion budget chain $I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$.

Apply Definition 38.515 to the resulting public $\mathsf{Real}^{\uparrow}$ read. The Real-side factorization admits the read only through the L10 stack whose visible owners are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The row ownership is exact. The StreamName exit exactness theorem Definition 126.372 identifies the public Real-completion exit with the finite StreamName window, the regular-source row, the dyadic observation ledger, and the Real seal boundary. The RegSeqRat source lattice exhaustion theorem Definition 301.104 assigns those rows to the four displayed owners and leaves no unassigned source row.

For the reverse direction, assume the finite budget chain of U pulls back to exactly those four owners. The $\mathsf{StreamName}^{\uparrow}$ window supplies the finite source surface, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger supplies the dyadic budget rows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback and classifier row supplies the regular source handoff, and the $\mathsf{Real}^{\uparrow}$ terminal seal supplies the public exit.

Recompose the same four rows using Definition 126.372. The exactness theorem packs the four-object route as a public Real-and-completion exit, while Definition 38.515 verifies that this is the RealUp input surface admitted by Phase Real and Completion.

Finally, the exclusions are inherited from the same route. The route lock Definition 38.754 and the three exactness theorems just cited expose only the displayed finite rows, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. Since quotient streams, selected host limits, countable choice, ambient completeness, and hidden completion carriers are not rows of that packet, none can provide a fifth owner or a second public $\mathsf{Real}^{\uparrow}$ exit. □

Theorem 38.761 (Real uniform Cauchy completion route lock factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. The consumer reaches a public* $\mathsf{Real}^{\uparrow}$ *exit or a completion-facing exit exactly through the composite*

route

$$\mathsf{UniformCauchyCriterion}^{\uparrow} \longmapsto \mathsf{FiniteTailFilter}^{\uparrow} \longmapsto (\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}).$$

The intervening $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet reads the same finite windows, dyadic tolerance ledger, regular readback, tail budget, common-tail comparison, Real seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row already displayed by* U*.*

This factorization is exhaustive for the Phase Real and Completion route. It does not export quotient streams, host real equality, selected limits, countable-choice selectors, ambient completeness, metric-completeness data, or standard bridge rows.

Proof. Start with the ordered budget chain supplied by Definition 38.754. Any consumer that reads an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet toward a $\mathsf{Real}^{\uparrow}$ or completion-facing endpoint must first expose the finite route

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Next apply Definition 435.27. It identifies the UniformCauchyCriterion rows with a $\mathsf{FiniteTailFilter}^{\uparrow}$ packet: the window row is the same finite-window read, the dyadic tolerance row is the same tolerance ledger, the regular readback is the same RegSeqRat read, and the tail-budget and common-tail comparison rows are the same displayed budget surface.

Use the finite-tail-filter handoff theorems to read the middle route. Definition 1119.4 fixes the common cofinal-tail window and dyadic tolerance packet used by the budget, comparison, and seal-facing rows. Then Definition 1119.5 carries exactly that packet to Real and Completion consumers without adding an independent tail selector, hidden tolerance ledger, or host-level completion coordinate.

Finally use Definition 38.760. Once the finite-tail-filter packet has reached the Real-facing seal, the public exit pulls back to precisely the four L10 owners $(\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow})$. Since those are the only owners exposed by the route lock and the finite-tail handoff, none of the excluded quotient, host-equality, selected-limit, choice, completeness, metric-completeness, or bridge rows can supply another Phase Real and Completion exit. □

Theorem 38.762 (Real uniform Cauchy finite-tail-filter L10 exit sufficiency)**.** *Let* $U = (I, W, M, Q, T, E, H, C, P, N$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet read by a Phase Real and Completion consumer. Suppose* U *factors through the displayed* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet of Definition 38.761, and suppose the resulting L10 surface exposes exactly the four owners* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *with the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Then this displayed finite-tail-filter route is sufficient for the Phase Real and Completion public exit surface. The route reads no quotient stream, host real equality, selected limit, countable-choice row, ambient completeness, metric-completeness datum, or standard bridge row.

Proof. The assumed factorization identifies the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ rows with the $\mathsf{FiniteTailFilter}^{\uparrow}$ packet by Definition 435.27. Thus the finite window, dyadic tolerance ledger, regular readback, tail budget, and common-tail comparison are already the displayed finite-tail-filter rows.

The Real-facing seal is then the handoff supplied by Definition 1119.5. It carries the same finite-tail packet to the Phase Real and Completion endpoint without adding an independent selector, a host equality predicate, or an ambient completeness coordinate.

It remains to identify the public L10 surface. By Definition 38.760, every row read at that endpoint pulls back to the four owners $(\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow})$. The

factorization theorem Definition 38.761 gives the same terminal route through $\mathsf{FiniteTailFilter}^{\uparrow}$, so the displayed four-object packet is exactly the terminal L10 pullback.

Consequently the route is sufficient for the current Phase Real and Completion public exit surface: it supplies all four required owners and all shared BHist, hsame, Cont, Pkg, and NameCert support rows. Since the excluded quotient, host-equality, selected-limit, choice, completeness, metric-completeness, and bridge data are not rows of that terminal packet, they are not read by this exit. □

Theorem 38.763 (RealCauchyCompletion uniform-Cauchy terminal equivalence)**.** *Let $C = (X, \mu, D, E_C, P_C, N_C)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *read, and let $U = (I, W, M, Q, T, E_U, H_U, C_U, P_U, N_U)$ be an accepted finite-budget* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *read. The two reads determine the same terminal Phase Real and Completion surface exactly when they expose the same finite budget rows and the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with the same displayed BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Equivalently, the $\mathsf{RealCauchyCompletion}^{\uparrow}$ *route and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *route are two presentations of one finite L10 surface precisely when the completion seal factors through the displayed tail-budget and uniform-criterion rows, and the uniform-Cauchy budget exits through the same four-object Real terminal packet. This equivalence reads no quotient stream, selected host limit, host real equality, countable-choice row, ambient completeness theorem, metric-completeness datum, or standard bridge row.*

Proof. For the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ presentation, first apply Definition 38.754. It locks the accepted finite budget to the displayed rows $I, W, M, Q, T, E_U, H_U, C_U, P_U, N_U$ and to the four Real L10 faces. Then Definition 38.755 identifies any two such accepted uniform-Cauchy exits once those rows and the four faces agree.

For the $\mathsf{RealCauchyCompletion}^{\uparrow}$ presentation, use Definition 360.32 to put the completion-facing seal on the same finite tail-budget route. Then apply Definition 360.37: the completion carrier whose diagonal packet consumes the uniform criterion reaches the $\mathsf{Real}^{\uparrow}$ seal only through the regular-family row, unary modulus, uniform shared-tail threshold, Cauchy-limit seal selector, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the displayed support rows.

It remains to identify the common terminal object. The four-object pullback Definition 126.504 says that the terminal Real-and-Completion read is fixed by $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ together with the shared BHist, hsame, Cont, Pkg, and NameCert rows. Hence agreement on those rows is sufficient for the two presentations to determine the same terminal finite L10 surface.

Conversely, if the two reads determine the same terminal Phase Real and Completion surface, the terminal pullback can display only the four faces and support rows just named. The route-lock and seal-factorization theorems then recover the finite uniform-Cauchy budget and the RealCauchyCompletion tail-budget seal route from that same packet, so the exposed finite rows agree.

The refusal clause is inherited from every cited boundary theorem. None of the locked rows contains a coordinate for quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, or standard bridges, so no such row can distinguish the two terminal presentations. □

Theorem 38.764 (Real uniform Cauchy StreamName RegSeqRat terminal seal nonescape)**.** *Any accepted Phase Real and Completion read through a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, a* $\mathsf{StreamName}^{\uparrow}$

finite-request section, and a $\mathsf{RegSeqRat}^{\uparrow}$ *shared-window selector reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the shared rows*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$.

It exports no quotient stream, host real equality, selected limit, countable-choice row, ambient completeness, metric-completion datum, or standard bridge row.

Proof. Start with Definition 38.754. It locks the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ read to the displayed finite budget rows and to the four Real L10 owners before any Phase Real and Completion consumer may read a terminal seal.

Next apply Definition 38.755. Once the $\mathsf{StreamName}^{\uparrow}$ request section, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ shared-window selector, and $\mathsf{Real}^{\uparrow}$ seal agree, the terminal surface is fixed together with its BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Finally use Definition 38.763. The completion-facing presentation and the uniform-Cauchy presentation determine the same terminal Phase Real and Completion surface exactly through the same four faces and support rows.

The excluded objects are absent from all three cited boundaries. Therefore the composed StreamName, RegSeqRat, and uniform-Cauchy route reaches the terminal $\mathsf{Real}^{\uparrow}$ seal only through the displayed shared rows, and no quotient stream, host equality, selected limit, choice row, completeness datum, or bridge row is exported. □

Theorem 38.765 (Real terminal seal Cauchy completeness witness route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer that claims Cauchy-completeness data factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the witness surface of chapter 3579 before exporting completion-facing rows. The route reads finite stream windows, regular rational readback, dyadic tolerance ledgers, the displayed* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *packet, and only then the local terminal Real seal.*

Consequently the terminal seal may export completion-facing rows only as a finite NameCert route through those sibling objects. It does not claim quotient stream equality, selected host limits, countable choice, trichotomy, detached completeness of $\mathsf{Real}^{\uparrow}$, *or an ambient Real completeness principle.*

Proof. Start with a terminal $\mathsf{Real}^{\uparrow}$ seal consumer that claims Cauchy-completeness data. The RealUp L10 input lattice already routes terminal sealing through the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before the local seal is read.

Next insert the witness chapter. The carrier and Real-seal route of chapter 3579 bind the same stream, regular-sequence, dyadic, and Real faces as one displayed completion-witness packet, so a completion-facing consumer must pass through that witness surface before it can treat the seal as carrying Cauchy-completeness data.

The final export is therefore a route projection through the existing RealUp L10 input lattice. Since the route contains only finite stream windows, regular rational readback, dyadic tolerances, the displayed witness packet, transport, replay, provenance, and local naming rows, it cannot introduce quotient equality, selected limits, countable choice, trichotomy, detached $\mathsf{Real}^{\uparrow}$ completeness, or ambient Real completeness. □

Theorem 38.766 (Real terminal seal no completeness)**.** *Passing a terminal* $\mathsf{Real}^{\uparrow}$ *seal through the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *route supplies only finite witness rows:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, the displayed witness packet, componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. It does not upgrade* $\mathsf{Real}^{\uparrow}$ *to an ambient*

completeness theorem and does not assert quotient stream equality, selected host limits, countable choice, trichotomy, detached Real completeness, or a host completion object.

Proof. Apply Definition 38.765. The cited route places every completion-facing consumer behind the finite StreamName, RegSeqRat, DyadicRatCore, and CauchyCompletenessWitness rows before the local Real seal is exported.

The Real roadmap rule keeps the claim off the ambient statement that $\mathsf{Real}^{\uparrow}$ is complete. The terminal seal may carry witness rows for a completion-facing consumer, but those rows are part of a displayed NameCert route rather than a theorem about all Real names.

The exclusions in the theorem statement are exactly the coordinates absent from the route: quotient stream equality, selected limits, countable choice, trichotomy, detached Real completeness, and host completion objects. Since none is present in the finite witness packet, no Real completeness claim is created. □

Theorem 38.767 (Real terminal seal witness nontransitive completeness)**.** *Composing the terminal* $\mathsf{Real}^{\uparrow}$ *seal with the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *route is not transitive to an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*. A downstream packet that consumes completion-facing data must still display* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and witness rows on that packet route, and it must keep transport, replay, provenance, and local naming visible before any Real-facing handoff is read.*

Proof. Apply the witness route theorem Definition 38.765. It identifies the only completion-facing path through finite stream windows, regular rational readback, dyadic tolerance ledgers, the displayed witness packet, and the terminal Real seal.

Next use the no-completeness theorem Definition 38.766. The terminal seal may carry those witness rows, but it has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, detached completeness, or a host completion object.

Finally read the L10 construction-order boundary of section 9467.12. The roadmap keeps $\mathsf{Real}^{\uparrow}$ as a sealed regular rational name with no current completeness claim. Therefore a downstream consumer must display the StreamName, RegSeqRat, DyadicRatCore, and witness rows each time it consumes completion-facing data, rather than inheriting ambient completeness from a composed seal. □

Theorem 38.768 (Real dyadic RegSeq cross-reference lattice)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *current-phase name read factors through four displayed faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed-mantissa and tolerance rows of chapter 291;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule of chapter 126;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows of chapter 301;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

The cross-reference lattice is the same finite route read by the Real L10 exit audit pullback. It explicitly refuses quotient streams, selected limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable-choice selectors, and any standard bridge row not displayed by the four faces.*

Proof. Start with the Real L10 exit audit pullback of Definition 38.471. It identifies each accepted current Phase Real and Completion exit read with the product of the DyadicRatCore, StreamName, RegSeqRat, and local Real face records.

Next use the ordered lattice in Definition 38.442 and the source-route non-escape theorem Definition 38.443. These keep the RealUp current-phase name read on the dyadic observation rows, the finite StreamName window rows, the RegSeqRat regular-rational rows, and the local Real seal.

Now attach the RegSeqRat scope triad. By Definition 301.152, every public RealUp seal read through RegSeqRat is placed behind the visible StreamName schedule, the DyadicRatCore endpoint and tolerance rows, and the RegSeqRat classifier surface before the seal row is consumed.

Finally combine the non-escape clauses. The Real L10 pullback contributes no fifth formal token or completion row, and the RegSeqRat scope triad contributes no quotient stream, selected limit, host equality, completion object, ambient completeness datum, or second seal coordinate. Therefore the current-phase RealUp name read is exactly the four-face cross-reference lattice and exports none of the refused data. □

Theorem 38.769 (Real L10 four-face phase-audit lock)**.** *An accepted current Phase Real and Completion audit read of* $\mathsf{Real}^{\uparrow}$ *is equivalent, as a public BEDC route, to the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *seal, together with their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The route refuses quotient streams, host real equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, and standard bridge rows not displayed by those four faces.*

Proof. For the forward direction, project the four faces using Definition 38.768. Every accepted current-phase name read already factors through the DyadicRatCore signed-mantissa and tolerance rows, the StreamName finite observation schedule, the RegSeqRat regularity and classifier rows, and the local Real seal.

Next use the Real L10 current phase exit audit pullback, Definition 38.471. It identifies the Phase Real and Completion audit read with the same four-face product route, so no additional completion-facing source is needed for the forward projection.

For the reverse direction, assemble the public route from the four sibling rows and the Real L10 dependency lattice. The ordered lattice of Definition 38.442, together with Definition 38.443 and Definition 301.152, repacks the four displayed faces as the current Phase Real and Completion audit read.

Finally combine the exclusion clauses. The cross-reference lattice, L10 nonescape row, and RegSeqRat scope triad do not expose quotient streams, host real equality, selected limits, ambient Real completeness, metric completeness, countable choice, or standard bridge rows, so the public audit route is locked to exactly the four displayed faces and their structural support. □

Theorem 38.770 (Real current-phase four-face terminal envelope)**.** *An accepted current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *terminal envelope is accepted exactly when its visible route is the four-face projection*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

of an accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *terminal envelope, with the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The* RealCompletionExactBoundaryUp *row remains a same-route exact-boundary witness for that projection and adds no quotient stream, selected limit, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, countable-choice selector, aggregate formalstatus row, or bridge token.*

Proof. For the forward direction, apply the four-face phase-audit lock of Definition 38.769. It identifies the accepted current Phase Real and Completion $\mathsf{Real}^{\uparrow}$ terminal read with the Dyadi-

cRatCore, StreamName, RegSeqRat, and local Real faces over the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next read that four-face terminal read through Definition 1327.8. The terminal envelope exactness theorem exhausts any accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ terminal consumer by the same finite route, with the exact-boundary face carried on the route rather than installed as another Real coordinate.

For the reverse direction, start from an accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ terminal envelope whose roadmap-visible projection is the displayed four-object route. Apply Definition 1327.16 to keep the RealCompletionExactBoundaryUp face as a same-route boundary witness and to expose only $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as roadmap objects.

Where the terminal read is given by local certified windows, repack those four visible faces by Definition 38.771. The pasted packet is therefore the accepted current Phase Real and Completion $\mathsf{Real}^{\uparrow}$ terminal envelope, while quotient streams, selected limits, host real equality, ambient Real completeness, countable choice, aggregate formalstatus rows, and bridge tokens remain outside the route. □

Theorem 38.771 (Real L10 finite-cover face-pasting equivalence)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite cover of certified* $\mathsf{StreamName}^{\uparrow}$ *windows. Assume that every pairwise overlap and every triple overlap satisfies the StreamName-RegSeqRat pasted source and seal hypotheses of Definition 126.551 and Definition 126.552. Assume also that all local rows consume the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, tolerance, and finite-distance ledger, and that their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows agree on all displayed overlaps.*

Then the induced local $\mathsf{RegSeqRat}^{\uparrow}$ *sections paste to exactly one accepted current Phase Real and Completion* $\mathsf{Real}^{\uparrow}$ *exit certificate over the finite union if and only if every local member exposes the same four visible faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

over the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

The equivalence is only a finite-cover current-phase exit certificate. It refuses quotient stream equality, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, suppose the local $\mathsf{RegSeqRat}^{\uparrow}$ sections paste to one accepted current Phase Real and Completion $\mathsf{Real}^{\uparrow}$ exit certificate. Project that certificate through Definition 38.768. The projection exposes only the DyadicRatCore ledger, the StreamName finite windows, the RegSeqRat local sections, and the local Real seal.

Now restrict this four-face packet to each member $\mathcal{W}_i$ of the finite cover. The phase-audit lock of Definition 38.769 keeps the restriction on the same BHist, hsame, Cont, Pkg, and NameCert support rows. Hence every local member exposes the same four visible faces over the same displayed support ledger.

For the reverse direction, assume each local member exposes the same four visible faces over the same support rows. First paste the StreamName and finite-window Real source surface using Definition 126.551. Then paste the finite-window Real seal rows using Definition 126.552. These two pasting steps use only the certified pairwise overlaps, triple-overlap cocycles, and shared DyadicRatCore ledger.

The pasted source and seal rows now form a finite cover of compatible local $\mathsf{RegSeqRat}^{\uparrow}$ terminal sections. Apply Definition 301.240 to obtain an accepted Real seal from the local four-face packets,

and apply Definition 301.241 to identify any two such pasted seals as the same BEDC-visible four-face packet.

Finally read the unique pasted four-face packet as the current Phase Real and Completion exit certificate by Definition 38.769. The construction has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient Real completeness, so the certificate is exactly the finite-cover current-phase exit surface described in the statement. □

Theorem 38.772 (Real L10 finite-cover dyadic common-tail cocycle descent)**.** *Let* $\{\mathcal{W}_i\}_{i\in I}$ *be a finite cover of certified* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose that every pairwise overlap carries a* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail packet, that every triple overlap satisfies the common-tail cocycle of Definition 301.191, and that the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, tolerance, distance, and finite scale-window ledgers agree through one common scale-window packet as in Definition 291.66. Suppose also that the local* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows are transported only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then the local $\mathsf{RegSeqRat}^{\uparrow}$ *sections descend to one Real L10 finite four-face exit certificate if and only if the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows agree on all pairwise overlaps and their induced comparison rows satisfy the given triple-overlap common-tail cocycle.*

The descent certificate is only the finite four-face current-phase exit route. It refuses quotient streams, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, begin with one descended Real L10 finite four-face exit certificate. Project it through Definition 38.771. The projection exposes the finite cover only by the four rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Restrict that certificate to every pairwise and triple overlap. The dyadic restriction is controlled by Definition 291.66, so all endpoint, radius, tolerance, distance, and finite scale-window rows are read from the same finite scale-window packet. The RegSeqRat restriction is controlled by Definition 301.191, so the common-tail comparison rows satisfy the triple-overlap cocycle.

For the reverse direction, assume the displayed pairwise row agreement and triple common-tail cocycle. First use Definition 291.66 to build the common dyadic scale-window face for the finite cover. This gives the single endpoint, radius, tolerance, distance, and finite-window ledger consumed by every local section.

Next apply Definition 301.191 and Definition 301.195. These paste the local $\mathsf{RegSeqRat}^{\uparrow}$ common-tail sections over the finite cover while keeping the StreamName observation windows and the dyadic scale-window ledger fixed on the same support rows.

The pasted common-tail packet now has the four faces required by the terminal route. Use Definition 126.115 to factor it through the StreamName, RegSeqRat, DyadicRatCore, and Real terminal pullback, then apply Definition 38.771 to read the result as one Real L10 finite-cover exit certificate.

Finally apply the phase-audit lock of Definition 38.769. It identifies the certificate with the same four visible faces and support rows and leaves no coordinate for quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.773 (Real finite-window seal tightness)**.** *A* $\mathsf{Real}^{\uparrow}$ *finite-window seal is admissible exactly when its* $\mathsf{RegSeqRat}^{\uparrow}$ *tail budget,* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, and* $\mathsf{DyadicRatCore}^{\uparrow}$

denominator ledger agree on a common hsame*-compatible finite request. The common request carries the displayed* BHist*,* Cont*,* Pkg*, and* NameCert *rows used by the public seal.*

Equivalently, a proposed seal whose RegSeqRat tail selector, StreamName request section, and dyadic denominator ledger do not factor through one common finite request is not an accepted $\mathsf{Real}^{\uparrow}$ *public seal. The statement uses only the displayed BEDC route and does not invoke host real equality, quotient streams, selected limits, or ambient completeness.*

Proof. For the forward direction, start with the Real dyadic RegSeq cross-reference lattice of Definition 38.768. It places every accepted Real current-phase name read on the DyadicRatCore, StreamName, RegSeqRat, and local Real faces.

Next apply the finite request fusion and finite-window seal exhaustion results, Definition 38.698 and Definition 38.718. These identify the public seal with a displayed finite request whose StreamName section, RegSeqRat readback, dyadic tolerance ledger, and support rows are componentwise compatible.

For the reverse direction, assemble the common request through the sibling route. The RegSeqRat terminal budget exhaustion supplies the tail-budget side, the StreamName finite request section supplies the public window, and the DyadicRatCore ledger supplies the denominator data for the same displayed request.

Finally repack the route as a RealUp public seal by the RegSeqRat-to-StreamName handoff and the dyadic Stream-RegSeq route. Since each cited route refuses quotient streams, host equality, selected limits, and ambient completeness, the common finite request is the whole admissibility witness. □

Theorem 38.774 (Real finite-cover seal pasting criterion)**.** *Let a finite cover of certified* $\mathsf{StreamName}^{\uparrow}$ *windows be given, with pairwise and triple overlaps satisfying the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *finite-cover pasted-source hypotheses. Suppose the local handoff rows consume the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger, and suppose each local* $\mathsf{RegSeqRat}^{\uparrow}$ *tail satisfies the* $\mathsf{Real}^{\uparrow}$ *finite-window tightness criterion of Definition 38.773.*

Then the local finite-window $\mathsf{Real}^{\uparrow}$ *seal rows paste to one admissible* $\mathsf{Real}^{\uparrow}$ *public seal over the finite cover. Conversely, a proposed public seal over such a finite cover is admissible exactly when there exists a common finite surface whose* $\mathsf{StreamName}^{\uparrow}$ *pasted source,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary are the displayed local rows on every window and overlap.*

The pasted surface is a finite BEDC datum: it records the relevant BHist*,* Cont*,* Pkg*, and* NameCert *rows on the cover index, on every binary overlap, and on every triple overlap required by the StreamName cocycle. It exports no quotient stream, host real equality, selected limit, countable choice, metric-completeness datum, ambient completeness row, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First build the finite cover surface. Apply Definition 126.551 to the certified StreamName windows and their pairwise and triple overlap data. This produces one pasted StreamName source surface, with the overlap rows still expressed as finite BHist, Cont, Pkg, and NameCert data.

Next transport the boundary rows through the RegSeqRat and dyadic faces. The overlap compatibility is exactly the common-tail Real boundary equivalence of Definition 301.186; the denominator ledger is carried through the four-face route by Definition 291.69. Hence every local handoff reads from the same finite dyadic ledger and the same RegSeqRat common-tail surface.

For the forward direction, apply Definition 38.773 on each window. The local admissibility witnesses all factor through the pasted StreamName source and the common dyadic ledger, so their Real seal rows have the same finite public support and therefore paste to one admissible public seal.

For the reverse direction, start from an admissible public seal over the cover and read its local finite-window restrictions. The tightness criterion supplies the local RegSeqRat tail budgets, StreamName sections, and dyadic denominator rows; the pasted-source and pullback routes identify these rows with one common finite surface. Thus admissibility is exactly the existence of that displayed pasted StreamName/DyadicRatCore/RegSeqRat surface, with no quotient, host-equality, choice, limit, or completeness datum added. □

Theorem 38.775 (Real L10 four-face finite pullback public interface)**.** *Let a* $\mathsf{Real}^{\uparrow}$ *finite-window public seal be displayed on finite* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Suppose its visible rows factor through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary, and the* $\mathsf{Real}^{\uparrow}$ *seal-source rows. Then these data determine exactly one public finite pullback interface over those displayed rows.*

Conversely, every four-face finite pullback interface with DyadicRatCore scale-window data, StreamName finite-window data, RegSeqRat common-tail boundary data, and local Real seal-source data repacks as an admissible $\mathsf{Real}^{\uparrow}$ *finite-window public seal. The interface exports no quotient stream equality, selected limit, host real equality, countable-choice selector, ambient completeness principle, metric-completeness row, or standard bridge row.*

Proof. For the forward direction, start with the finite-window public seal and apply Definition 38.773. The admissible seal is read exactly through a common finite request carrying the StreamName section, RegSeqRat tail budget, DyadicRatCore denominator ledger, and local Real rows.

Next apply Definition 38.774. It identifies the finite public support of the seal with the pasted finite surface on the displayed BHist, Cont, Pkg, and NameCert rows, so the local windows and their overlaps have one public finite support.

Now transport the boundary and dyadic faces. The RegSeqRat common-tail rows are exactly the Real boundary rows by Definition 301.186, and the DyadicRatCore scale-window packet composes with the RegSeqRat and Real faces by Definition 291.69. Thus the four displayed faces form one finite pullback interface, and uniqueness follows because each cited route reads the same finite rows rather than a second external coordinate.

For the reverse direction, begin with such a four-face finite pullback interface. The dyadic scale-window packet, StreamName finite-window section, RegSeqRat common-tail boundary, and local Real seal-source rows supply the common request required by Definition 38.773.

Finally repack the common request as a public seal and use Definition 38.774 to pass across the displayed finite cover. The same exclusion clauses in the four cited routes leave no quotient stream equality, selected limit, host real equality, countable choice, ambient completeness, metric completeness, or standard bridge row to add. □

Theorem 38.776 (Real L10 four-face refinement stability)**.** *Let a* $\mathsf{Real}^{\uparrow}$ *finite-window public seal factor through the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows as in Definition 38.775. Any* hsame*-compatible finite-window refinement that preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary, the local seal row, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows determines the same public finite pullback interface and repacks as the same admissible finite-window seal.*

Conversely, any accepted refinement of that public interface exposes exactly those refined four faces. It exports no quotient stream equality, selected limit, host real equality, countable-choice selector, ambient completeness row, metric-completeness row, standard bridge row, or second public seal coordinate.

Proof. Start with the displayed public seal. By Definition 38.773, admissibility is exactly factorization through one common finite request carrying the StreamName section, RegSeqRat tail budget, DyadicRatCore denominator ledger, and the local Real rows.

Apply Definition 38.774 to the refined finite windows. Since the refinement preserves the scale-window packet, finite-window section, common-tail boundary, local seal row, displayed Cont routes, Pkg provenance, and local NameCert rows, the pasted finite surface has the same public support on every window and overlap.

Next use the StreamName and RegSeqRat pullback determinacy rows, together with Definition 38.775. They identify the refined StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal row with the same four-face finite pullback interface, rather than with a second external coordinate.

For the converse, read an accepted refinement through the public interface theorem. Its admissibility witness must factor through the four displayed faces, and the tightness and pasting criteria force the refined rows to be exactly the preserved scale-window, finite-window, common-tail, local-seal, route, provenance, and certificate rows.

The exclusion clauses are inherited from the same cited routes. None of them exports quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, or standard bridge data, so the accepted refinement cannot expose any such row. □

Theorem 38.777 (Real L10 four-face interface composition)**.** *Let I_{01} and I_{12} be composable finite pullback interfaces for* $\mathsf{Real}^{\uparrow}$ *finite-window public seals. Suppose both interfaces preserve the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet,* $\mathsf{StreamName}^{\uparrow}$ *finite-window section,* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary, local* $\mathsf{Real}^{\uparrow}$ *seal-source row, displayed* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows on their common overlap. Then the composite interface determines exactly the same admissible* $\mathsf{Real}^{\uparrow}$ *finite-window public seal as the one-step interface over the composite finite request.*

Conversely, any accepted composite public seal exposes exactly the two component finite pullback interfaces I_{01} and I_{12} on the displayed four faces and on the shared support rows. The composite has no additional coordinate for quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric completeness, standard bridge rows, or a second public seal.

Proof. First read each component through the public four-face interface. By Definition 38.775, I_{01} and I_{12} are exactly the finite public pullback surfaces carried by their DyadicRatCore scale-window packets, StreamName finite-window sections, RegSeqRat common-tail boundaries, local Real seal-source rows, and displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply refinement stability on the common overlap. Since both component interfaces preserve the listed rows there, Definition 38.776 identifies the overlap restriction with the same admissible finite-window seal rather than with a second seal coordinate.

The composite finite request is then pasted from these two compatible restrictions. Use Definition 38.774 for the two-window cover and Definition 301.189 for the common-tail triple row implicit in the pullback composition. These results say that the pasted StreamName section, dyadic scale-window ledger, RegSeqRat common-tail boundary, local Real row, and displayed support data form one four-face cocycle, so the composite interface reads as the same public seal as the one-step interface over the composite request.

For the converse, start from an accepted composite public seal and restrict it to the two finite requests. The public interface theorem exposes the four displayed faces on each restriction, while refinement stability forces the overlap rows to be the preserved DyadicRatCore, StreamName,

RegSeqRat, local Real, BHist, hsame, Cont, Pkg, and NameCert rows. Hence the component interfaces are exactly I_{01} and I_{12}.

The exclusion clauses follow from the same four cited routes. Quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric completeness, standard bridge rows, and any duplicate public seal coordinate are not rows of the finite request or of the pasted cocycle, so they cannot appear in either direction of the composition statement. □

Theorem 38.778 (Real common-tail terminal four-face classifier equivalence)**.** *Let $\mathcal{S}$ and $\mathcal{T}$ be accepted* $\mathsf{Real}^{\uparrow}$ *finite-window public seals whose visible rows factor through the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows. Then $\mathcal{S}$ and $\mathcal{T}$ have the same public classifier surface if and only if they agree on the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail window packet, the* $\mathsf{StreamName}^{\uparrow}$ *fused terminal-window packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the local* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

This equivalence is an equivalence of the displayed finite BEDC boundary only. It excludes quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge rows, and any second public seal coordinate.

Proof. For the forward direction, start from public classifier agreement. Apply Definition 126.624; it reads classifier agreement back to the common StreamName fused terminal packet, the dyadic endpoint, radius, and distance ledger, the RegSeqRat common-tail rows, the Real seal-source row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Then apply Definition 301.186. The RegSeqRat common-tail packet and the finite-request Real boundary determine one another on the same StreamName schedule and refinement rows, dyadic ledger rows, classifier and regularity rows, local Real seal-source rows, and provenance rows. Thus no classifier agreement can remain after those displayed rows diverge.

For the reverse direction, assume agreement of the RegSeqRat common-tail window packet, the StreamName fused terminal-window packet, the DyadicRatCore endpoint/radius/distance ledger, the local Real seal-source row, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows. The four-face public interface theorem Definition 38.775 repacks these rows as one public finite pullback interface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use Definition 126.624 again to transport that shared four-face interface to the public classifier surface. The exclusion clauses in the StreamName terminal fusion theorem, the RegSeqRat common-tail boundary theorem, and the four-face interface theorem leave no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness row, metric-completeness row, standard bridge row, or second seal coordinate through which a different classifier could be read. □

Theorem 38.779 (Real L10 four-face classifier sheaf condition)**.** *Let a finite cover of* $\mathsf{Real}^{\uparrow}$ *public classifier reads be displayed over the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, with finite* BHist, *componentwise* hsame, Cont, Pkg, *and* NameCert *support on every window. The family satisfies the public* $\mathsf{Real}^{\uparrow}$ *classifier exactly when each local read satisfies the tightness criterion of Definition 38.773, each pairwise overlap satisfies the terminal classifier equivalence of Definition 38.778, and each triple overlap satisfies the common-tail four-face cocycle determinacy of Definition 301.189.*

Under these three conditions the pasted public classifier is the same classifier as the one-step four-face interface over the composite finite request of Definition 38.777. Conversely, any one-step public $\mathsf{Real}^{\uparrow}$ *classifier restricts to such local reads, pairwise terminal equivalences, and triple*

common-tail cocycles. The condition is internal to the displayed finite BEDC rows and excludes quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric completeness, standard bridge rows, and any second public classifier coordinate.

Proof. First build the finite public surface. Apply Definition 38.774 to the local reads satisfying Definition 38.773. The result is one pasted finite support carrying the StreamName sections, RegSeqRat common-tail boundaries, DyadicRatCore ledgers, local Real rows, and displayed BHist, hsame, Cont, Pkg, and NameCert data on the cover.

Next identify the overlap classifiers. On each pairwise overlap, Definition 38.778 says that public classifier agreement is exactly agreement of the RegSeqRat common-tail packet, StreamName terminal-window packet, DyadicRatCore endpoint/radius/distance ledger, local Real seal-source row, and displayed support rows. On each triple overlap, Definition 301.189 supplies the cocycle condition for those common-tail rows.

Now repack the pasted surface through the four-face public interface. Use Definition 38.775 to read the pasted DyadicRatCore, StreamName, RegSeqRat, and Real rows as one public finite pullback interface. Then apply Definition 38.777; the pairwise terminal agreements and triple cocycles make the iterated pasted interface equal to the one-step interface over the composite finite request.

For the converse, restrict a one-step public classifier to the finite cover. The interface theorem exposes the four displayed faces on every restriction, the tightness criterion gives the local admissibility rows, the terminal classifier equivalence gives the pairwise overlap rows, and the common-tail cocycle determinacy gives the triple overlap rows.

The exclusions are inherited from the same cited routes. None of the pasting, interface, composition, terminal-equivalence, or cocycle rows exports quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric completeness, standard bridge data, or a duplicate public classifier coordinate. □

Theorem 38.780 (Real L10 finite-cover classifier descent)**.** *Let a finite cover of public* $\mathsf{Real}^{\uparrow}$ *classifier reads be given over the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, with finite* BHist, *componentwise* hsame, Cont, Pkg, *and* NameCert *support on every window and overlap. Then restriction to every window and overlap satisfies the public* $\mathsf{Real}^{\uparrow}$ *classifier exactly when the local tightness rows, pairwise terminal classifier equivalences, and triple common-tail cocycles of Definition 38.779 descend to one public classifier surface.*

The descended surface is the public classifier surface exported by the one-step four-face interface over the composite finite request. It contains exactly the DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, and displayed BHist, hsame, Cont, Pkg, *and* NameCert *support rows. It excludes quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge rows, and duplicate classifier coordinates.*

Proof. For the forward direction, start from coverwise public classifier restriction. Apply Definition 38.779. It reads each restriction as local tightness, each pairwise overlap as a terminal classifier equivalence, and each triple overlap as a common-tail cocycle.

Next transport the local rows to the composite request. Use Definition 126.552 to exhaust the finite cover by the pasted StreamName and RegSeqRat seal rows, with the DyadicRatCore ledger, local Real seal-source rows, and displayed support rows retained on each finite refinement.

Now repack the exhausted cover as a single public surface. Apply Definition 38.777; the compatible four-face restrictions compose to the one-step public interface over the composite finite

request. Then Definition 38.778 identifies that interface with the terminal public classifier surface determined by exactly the displayed rows.

For the reverse direction, begin with one public classifier surface over the composite finite request. Restrict it to the chosen finite cover. The interface composition theorem exposes the same DyadicRatCore, StreamName, RegSeqRat, local Real, BHist, hsame, Cont, Pkg, and NameCert rows on every window and overlap.

Apply Definition 38.779 again. The restricted surface therefore supplies the local tightness rows, the pairwise terminal classifier equivalences, and the triple common-tail cocycles required on the cover. The terminal equivalence theorem shows that any cover with those displayed rows descends to the same public classifier surface, so no additional classifier coordinate is available.

The exclusion clauses are row exclusions along the cited routes. The finite-cover exhaustion, four-face composition, terminal classifier equivalence, and sheaf condition use only finite BEDC support rows and never introduce quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, standard bridge rows, or duplicate classifier coordinates. □

Theorem 38.781 (Real L10 classifier-status axis separation)**.** *For any finite Real L10 four-face packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, public classifier sheaf agreement is exactly the local tightness, pairwise terminal-equivalence, and triple common-tail cocycle data of Definition 38.779. The audit admissibility coordinate is exactly objectwise lookup from the four local closurestatus records through Definition 38.471, with the status and target coordinates kept facewise by Definition 38.833 and Definition 38.834.*

These two coordinates do not supply one another. Classifier agreement creates no formalstatus row, leantarget row, bridgestatus row, or bridge row, and audit status creates no classifier overlap, finite-window tightness row, pairwise terminal classifier equivalence, or triple common-tail cocycle.

Proof. First read the classifier coordinate. Apply Definition 38.779; it reduces public classifier agreement to the local tightness criterion, the terminal four-face classifier equivalences on pairwise overlaps, and the common-tail cocycle determinacy on triple overlaps.

Next read the audit coordinate. Apply Definition 38.471; the current Phase Real and Completion audit row is the objectwise product of the DyadicRatCore, StreamName, RegSeqRat, and local Real closurestatus records.

The two non-collapse results separate the displayed audit fields. By Definition 38.833, the four local formalstatus, theoryclosure, and bridgestatus fields are not replaced by a single aggregate status. By Definition 38.834, the four local leantarget rows are likewise not replaced by an aggregate target.

The cited classifier theorem exports only tightness, overlap equivalence, and cocycle data on finite BEDC rows. The cited audit theorems export only objectwise closurestatus lookup and its non-collapse fields. Therefore neither route manufactures the rows of the other route: classifier agreement has no status or bridge output, while audit admissibility has no classifier-overlap, tightness, or common-tail-cocycle output. □

Theorem 38.782 (Real L10 public classifier audit envelope)**.** *Let* $\mathcal{P}$ *be a finite Real L10 four-face public classifier packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, with displayed finite* BHist, *componentwise* hsame, Cont, Pkg, *and* NameCert *support on every window. Suppose that* $\mathcal{P}$ *satisfies the finite-cover classifier sheaf condition of Definition 38.779, and that the four local face records satisfy the objectwise audit-clean exit threshold of Definition 38.493 and Definition 38.494. Then* $\mathcal{P}$ *is an audit-admissible public classifier envelope exactly on those four displayed faces.*

The envelope consists only of the local tightness rows, pairwise terminal-equivalence rows, triple common-tail cocycle rows, and the four objectwise closurestatus records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It supplies no aggregate route-level* formalstatus, *no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 38.779. The public classifier coordinate of $\mathcal{P}$ is reduced to local tightness over the displayed finite support, pairwise terminal equivalence on overlaps, and triple common-tail cocycle data.

Next apply Definition 38.781. The classifier rows just obtained cannot manufacture a formalstatus row, leantarget row, bridgestatus row, or bridge row, and the audit rows cannot manufacture classifier overlap, tightness, or cocycle data.

Now read the exit threshold objectwise. By Definition 38.493, the current Real L10 exit criterion is satisfied by the four local closurestatus records under the stated local closure and audit-clean verification rows. By Definition 38.494, those rows are exactly the four-face threshold and not a route-level aggregate.

Finally use Definition 38.833 and Definition 38.834. The formal-status, theory-closure, Leantarget, and bridge-status coordinates remain attached to the four displayed faces, so the resulting public classifier envelope is audit-admissible only there and exports none of the excluded aggregate, quotient, choice, limit, metric-completeness, or $\mathsf{Real}^{\uparrow}$ completeness data. □

Theorem 38.783 (Real L10 four-face classifier separatedness)**.** *Let* $\mathcal{S}$ *and* $\mathcal{T}$ *be two finite-cover public* $\mathsf{Real}^{\uparrow}$ *classifier presentations satisfying the sheaf condition of Definition 38.779. Suppose that, on every window and on every overlap in the cover, their visible rows agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail boundary, the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, and the displayed support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{S}$ *and* $\mathcal{T}$ *determine the same public* $\mathsf{Real}^{\uparrow}$ *classifier surface. Conversely, any disagreement in the public* $\mathsf{Real}^{\uparrow}$ *classifier surface of two such presentations is witnessed on at least one of the displayed DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, or* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. No quotient stream equality, selected limit, host real equality, countable choice, ambient completeness, metric completeness, standard bridge row, or second classifier coordinate is part of the separatedness witness.*

Proof. Apply Definition 38.779 to both presentations. The sheaf condition reads each finite-cover classifier through local tightness rows, pairwise terminal classifier equivalences, and triple common-tail cocycles, then repacks the result as the one-step four-face interface over the composite finite request.

The assumed windowwise agreement identifies the one-step interfaces. By Definition 38.777, the pasted request equality is already determined by the DyadicRatCore ledger, StreamName finite-window rows, RegSeqRat common-tail boundary, local Real seal-source rows, and displayed support data on the common cover.

It remains to check that no hidden classifier coordinate can distinguish the pasted surfaces. On pairwise overlaps, Definition 38.778 makes public classifier agreement equivalent to agreement of exactly the displayed terminal four-face rows. On triple overlaps, Definition 301.189 forces the common-tail cocycle rows to agree whenever those displayed rows agree.

Thus the two pasted classifier surfaces have the same local reads, the same overlap identifications, and the same composite finite request, hence the same public $\mathsf{Real}^{\uparrow}$ classifier surface. The converse is the contrapositive of the same reductions: if the public classifiers differ, the sheaf reduction, terminal equivalence, or common-tail cocycle determinacy must expose the difference in one of the listed visible rows. Since the cited routes export only finite BEDC rows, the excluded quotient, limit, host-equality, choice, completeness, bridge, and duplicate-coordinate data cannot serve as an additional witness. □

Theorem 38.784 (Real L10 four-face witness minimality)**.** *For any accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion finite-window public classifier packet, the visible witness surface is minimal for the displayed four-face separatedness route. Removing the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face removes the endpoint, radius, and distance ledger data; removing the* $\mathsf{StreamName}^{\uparrow}$ *face removes the finite schedule; removing the* $\mathsf{RegSeqRat}^{\uparrow}$ *face removes the regular-tail classifier; and removing the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows removes the seal boundary. In each case, Definition 38.783 is no longer applicable by the displayed route.*

Likewise, removing any displayed support row among BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *destroys either the provenance or the transport needed to read the public finite-window witness. Conversely, the full four-face packet determines the same public classifier surface by Definition 38.783. The minimality statement is internal to the displayed BEDC rows and excludes quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric completeness, standard bridge rows, and any second classifier coordinate.*

Proof. First read the packet through the tightness criterion. By Definition 38.773, an accepted finite-window seal is exactly a common finite request carrying the RegSeqRat tail budget, StreamName finite-window section, DyadicRatCore denominator ledger, local Real rows, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Now remove one face at a time. Without the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the route no longer contains the endpoint, radius, and distance data used to compare public classifier surfaces. Without the $\mathsf{StreamName}^{\uparrow}$ section, the dyadic and regular-tail rows no longer supply the finite schedule on which the witness is read. Without the $\mathsf{RegSeqRat}^{\uparrow}$ common-tail boundary, the finite schedule and dyadic ledger no longer supply the regular-tail classifier. Without the local $\mathsf{Real}^{\uparrow}$ seal-source rows, the three imported faces no longer supply the seal boundary.

The support rows are load-bearing for the same reason. Dropping BHist or NameCert removes the finite provenance of the displayed witness, dropping hsame removes the componentwise identification needed across windows, dropping Cont removes the transport route, and dropping Pkg removes the package row that connects the public presentation to its certified carrier.

Finally restore the whole surface. The finite pullback interface of Definition 38.775 repacks the four faces and support rows as one public interface, while Definition 38.778 and Definition 38.779 identify the terminal and coverwise classifier reads. Then Definition 38.783 gives determinacy of the public classifier from exactly those visible rows.

These citations are non-reconstruction results: they specify the rows consumed by the tightness, finite-pullback, and separatedness routes, but they do not construct counter-packets after a row is deleted. The exclusion clause follows from the same route inventory. None of the cited results exports quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric completeness, standard bridge rows, or a second classifier coordinate, so no such datum is available inside this proof as a replacement for a missing face or support row. □

Theorem 38.785 (Real L10 four-face classifier functoriality)**.** *Let r be a finite-request refinement map between two current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion classifier presentations. Suppose r preserves*

the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the displayed support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then transport along r carries a public $\mathsf{Real}^{\uparrow}$ *classifier surface to the same public classifier surface. Conversely, any equality of transported public classifier surfaces is witnessed exactly on the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *together with those displayed support rows. The functoriality surface exports no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness, metric-completeness row, standard bridge row, or second classifier coordinate.*

Proof. First read the refinement map through the sibling certificates. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger is supplied by chapter 291, the finite request schedule by chapter 126, and the common-tail classifier rows by chapter 301. These are the three imported faces consumed by the local $\mathsf{Real}^{\uparrow}$ seal-source row.

Apply Definition 38.776. Since r preserves the dyadic ledger, finite-window section, common-tail boundary, local seal row, displayed Cont routes, Pkg provenance, and $\mathsf{NameCert}$ rows, the refinement determines the same finite pullback interface and repacks as the same admissible public seal.

Next compose transport with the public interface. By Definition 38.777, a transported interface over the refined request is the one-step interface over the composite finite request, with the same BHist, componentwise hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows. Thus the transported surface has no independent classifier coordinate beyond the four displayed faces.

For the converse, start from equality of transported public classifier surfaces. Separatedness in Definition 38.783 reads any possible disagreement back to one of the DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, or displayed support rows. Therefore equality is witnessed precisely by agreement on those rows.

Finally use Definition 38.784 to rule out a smaller witness inventory. Removing any one of the four faces or any one of BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ destroys the displayed route on which separatedness is read. The same row inventory excludes quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric completeness, standard bridge rows, and a second classifier coordinate from both directions of the functoriality statement. □

Theorem 38.786 (Real L10 cofinal classifier refinement invariance)**.** *Let $\mathcal{P}$ be a current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion finite-window public classifier packet. Let $\mathcal{R}$ be a cofinal family of finite-request refinements of $\mathcal{P}$: every finite public readback request for $\mathcal{P}$ is dominated by some refinement in $\mathcal{R}$. Suppose that each refinement in $\mathcal{R}$ preserves the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows, and also preserves the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then all public classifier surfaces read from refinements in $\mathcal{R}$ determine the same $\mathsf{Real}^{\uparrow}$ *classifier surface as $\mathcal{P}$. Conversely, if two cofinal refined reads determine different public* $\mathsf{Real}^{\uparrow}$ *classifier surfaces, then the disagreement is witnessed at some finite refinement stage on one of the displayed DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, or* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows.*

The invariance is only an assertion about the displayed finite BEDC classifier surface. It asserts no formal-status row, bridge row, quotient stream equality, selected limit, countable-choice

row, ambient completeness row, metric-completeness row, standard bridge row, or second public classifier coordinate.

Proof. First fix one refinement in $\mathcal{R}$. The preservation hypotheses are exactly the hypotheses of Definition 38.785, so transport along that refinement reads the refined public classifier surface as the same public classifier surface carried by $\mathcal{P}$ on the displayed four faces and support rows.

Now compare two refinements in the cofinal family. Cofinality gives a further finite request dominating the two finite reads. Applying functoriality along the two induced refinement maps transports both classifier surfaces to the same displayed four-face packet over that common finite request. Hence every refined read in $\mathcal{R}$ determines the same public $\mathsf{Real}^{\uparrow}$ classifier surface.

For the converse, suppose two refined public classifier surfaces disagree. By Definition 38.783, any such disagreement is already visible on one of the DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, or displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

The witness is finite because both disagreeing surfaces come from finite-request refinements, and cofinality supplies a common finite refinement stage on which the two reads can be compared. The separatedness witness therefore appears at such a finite stage, not in an ambient quotient, limit, choice, completeness, bridge, or duplicate classifier coordinate.

Finally, Definition 38.784 rules out shrinking the displayed witness inventory, while Definition 38.781 keeps the classifier claim separate from audit status. Thus the theorem states cofinal invariance of the public classifier surface only, with no assertion about formal status, bridge status, quotient streams, selected limits, countable choice, ambient completeness, or metric completeness. □

Theorem 38.787 (Real L10 terminal cofinal pullback determinacy)**.** *Let $\mathcal{P}$ be a current Phase $\mathsf{Real}^{\uparrow}$ and Completion public classifier packet, and let $\mathcal{R}$ be a cofinal family of terminal finite-request refinements of $\mathcal{P}$. Suppose every terminal refinement in $\mathcal{R}$ preserves the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

where the $\mathsf{Real}^{\uparrow}$ face is the local seal-source row, and suppose it also preserves the support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the terminal pullback induced by $\mathcal{R}$ is uniquely determined by those four faces and support rows. Conversely, if two terminal cofinal pullbacks over current Phase $\mathsf{Real}^{\uparrow}$ and Completion data disagree, then their disagreement is witnessed at a finite refinement stage on one of the DyadicRatCore ledger rows, StreamName terminal-window rows, RegSeqRat finite-request rows, local Real seal-source rows, or displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

The determinacy surface contains no quotient stream equality, host real equality, selected limit, countable-choice row, ambient completeness row, metric-completeness row, arithmetic package, order package, standard bridge row, or second public classifier coordinate.

Proof. First read each terminal refinement through cofinal classifier invariance. By Definition 38.786, every refinement in $\mathcal{R}$ presents the same public $\mathsf{Real}^{\uparrow}$ classifier surface as $\mathcal{P}$ whenever it preserves the displayed four faces and support rows.

Next identify the finite witness carried by that common classifier surface. Apply Definition 38.464; the public boundary determines exactly the fused $\mathsf{StreamName}^{\uparrow}$ terminal-window packet, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, the local $\mathsf{Real}^{\uparrow}$ seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now fix the source of the pullback. The source-route exactness theorem, Definition 38.466, places every terminal finite-window witness on the ordered route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus the terminal pullback has no independent source coordinate beyond the four displayed faces.

Finally compare the three public readings. By Definition 38.467, the StreamName finite-window classifier pullback, the RegSeqRat finite-request section, and the Real L10 source route read the same public boundary over the same support rows. The terminal pullback induced by the cofinal family is therefore unique, since all of its visible faces are already fixed by the common finite-window witness and by the route triangle.

For the converse, suppose two terminal cofinal pullbacks disagree. Cofinality gives a finite refinement stage dominating the two finite reads. The preceding paragraphs identify both pullbacks at that stage with their DyadicRatCore ledger, StreamName terminal-window packet, RegSeqRat finite-request section, local Real seal-source rows, and displayed support rows.

If those rows agreed, the cofinal invariance theorem and the finite-window determinacy theorem would give the same public boundary, and the route triangle would repack the same terminal pullback. Hence any disagreement must be visible at that finite stage on one of the displayed faces or support rows.

The exclusion clause follows from the same cited inventory. None of the four cited results exports quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric completeness, arithmetic or order packages, bridge rows, or a second public classifier coordinate, so none of those data can witness or repair a terminal cofinal pullback disagreement here. □

Theorem 38.788 (Real L10 classifier refinement category laws)**.** *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion finite-window public classifier packets. The identity finite-request refinement of $\mathcal{P}$ preserves the public* $\mathsf{Real}^{\uparrow}$ *classifier surface carried by the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

where the $\mathsf{Real}^{\uparrow}$ *face is the local seal-source row, together with the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

If $r : \mathcal{P} \to \mathcal{Q}$ and $s : \mathcal{Q} \to \mathcal{R}$ are finite-request refinements preserving those four faces and support rows, then their composite $s \circ r$ preserves the same public $\mathsf{Real}^{\uparrow}$ *classifier surface. Conversely, if the classifier surfaces transported by a composite refinement disagree, then the disagreement is witnessed at a finite stage on one of the DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat common-tail rows, local Real seal-source rows, or displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows.*

These laws are only laws for the displayed finite BEDC refinement action. They assert no quotient stream equality, host real equality, selected limit, countable-choice row, ambient completeness row, metric-completeness row, bridge row, or second public classifier coordinate.

Proof. For the identity refinement, the finite request, four classifier faces, and support rows are unchanged. Applying Definition 38.785 to this preservation data reads the transported public classifier surface as the original public $\mathsf{Real}^{\uparrow}$ classifier surface.

Now consider refinements $r : \mathcal{P} \to \mathcal{Q}$ and $s : \mathcal{Q} \to \mathcal{R}$. The preservation hypotheses for r and s identify the DyadicRatCore ledger, StreamName finite-window section, RegSeqRat common-tail

rows, local Real seal-source row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support across the two steps.

Apply Definition 38.777. The two-step transport is the same finite interface as transport along the composite finite request, with no additional classifier coordinate. Then Definition 38.785 gives preservation of the public classifier surface for $s \circ r$.

For the converse, suppose the composite transport gives two different public classifier surfaces. By Definition 38.783, such a difference must already be visible on one of the four displayed faces or on one of the listed support rows.

Finally, Definition 38.787 places that witness at a finite refinement stage whenever the comparison is made through a terminal cofinal pullback. The cited route inventory exports only the displayed finite BEDC rows, so quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric completeness, bridge rows, and a second classifier coordinate cannot supply an additional obstruction or a repair. □

Theorem 38.789 (Real L10 exit-threshold classifier coherence)**.** *For any finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal rows, the following two data are equivalent to one displayed four-face record.*

First, the packet has public classifier agreement in the sense of the Real L10 four-face classifier pullback and sheaf condition. Second, the same packet satisfies the objectwise exit threshold for the current Phase Real and Completion audit route. The displayed record carries exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The classifier coordinate and the exit-threshold coordinate are read from that same record, but neither coordinate supplies the other. The statement excludes quotient streams, host real equality, selected limits, countable choice, metric-completeness data, $\mathsf{Real}^{\uparrow}$ *completeness, aggregate formalstatus rows, and aggregate bridge rows.*

Proof. First read the classifier coordinate. Apply Definition 38.472 and Definition 38.779. They identify public classifier agreement with the finite four-face surface over the DyadicRatCore ledger, StreamName finite-window rows, RegSeqRat regularity and common-tail rows, local Real seal rows, and the displayed support rows.

Next read the exit-threshold coordinate. Apply Definition 38.494. It says that the current Phase Real and Completion exit threshold is exactly the objectwise threshold over the same four local face records, rather than a route-level aggregate status.

Now combine the two readings. The classifier theorems and the threshold theorem consume the same four concrete faces and the same finite support rows, so their conjunction is equivalent to the single displayed record carrying those rows. In the forward direction, the classifier agreement supplies the classifier face rows and the threshold equivalence supplies the objectwise audit rows on those faces. In the reverse direction, the displayed record feeds the classifier pullback and sheaf condition, and it also feeds the objectwise threshold equivalence.

It remains to show noninterference of the two coordinates. Apply Definition 38.781. Classifier agreement creates no formalstatus, leantarget, bridgestatus, or bridge row, while audit admissibility creates no classifier overlap, finite-window tightness row, pairwise terminal classifier equivalence, or triple common-tail cocycle.

Finally apply Definition 38.833 and Definition 38.834. These keep the status and Lean-target data attached to the four displayed faces and rule out an aggregate replacement. Since the cited

routes export only the listed finite BEDC rows, no quotient stream, host equality, selected limit, choice principle, metric-completeness datum, $\mathsf{Real}^{\uparrow}$ completeness row, aggregate formalstatus row, or aggregate bridge row is available as an extra coordinate. □

Theorem 38.790 (Real L10 classifier-threshold source conservativity)**.** *Let $\mathcal{P}$ be a finite Real L10 four-face packet satisfying both public classifier agreement and the objectwise exit threshold over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Then the accepted source of $\mathcal{P}$ is exactly the displayed four-face record over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Adding the threshold coordinate does not create a fifth source, an aggregate $\mathsf{formalstatus}$ *row, an aggregate bridge row, a quotient stream, a selected limit, a metric-completeness datum, a countable-choice row, or a* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First collapse the hypotheses to one record. By Definition 38.789, public classifier agreement and the objectwise exit threshold are read from the same four-face packet, whose only displayed sources are the DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat regularity and common-tail rows, local Real seal rows, and the support rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Next identify that record as the accepted source. Apply Definition 38.472 to read classifier agreement as the public four-face surface, and apply Definition 38.494 to read the threshold objectwise on those same faces. The uniqueness comparison is then the source-uniqueness statement of Definition 291.84: the accepted Real exit source is the displayed DyadicRatCore-to-Real four-face route, not an additional carrier.

It remains to separate the source from verification and bridge coordinates. By Definition 38.781, classifier rows do not manufacture $\mathsf{formalstatus}$, leantarget, bridgestatus, or bridge rows, and the audit rows do not manufacture classifier overlap or finite-window tightness data. Thus adjoining the threshold coordinate preserves the four-face source and supplies no aggregate verification or bridge row.

Finally, every excluded object would require a coordinate absent from the four cited readings. Quotient streams and selected limits are not part of the finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support; metric completeness, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness are not exported by the objectwise threshold; and an aggregate status or bridge row is ruled out by the axis-separation theorem. Hence the accepted source remains exactly the displayed four-face record. □

Theorem 38.791 (Real L10 four-face exit classifier reflection)**.** *For any finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows, public classifier agreement on the four-face packet is equivalent to the objectwise Phase Real and Completion exit-threshold read of that same packet precisely when both are reflected from the displayed four-face source record*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The unique reflected source is the finite record carrying exactly the four faces together with the displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}$ *support rows.*

This reflection is only a public classifier and exit-read comparison on the same finite packet. It does not identify quotient streams, host real equality, selected limits, countable-choice selectors, metric-completeness data, $\mathsf{Real}^{\uparrow}$ *completeness, aggregate formalstatus rows, aggregate leantarget rows, or aggregate bridge rows.*

Proof. First put the classifier coordinate and the threshold coordinate on the same source record. Apply Definition 38.789. It identifies public classifier agreement and the objectwise exit threshold as two readings of one finite four-face packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the local $\mathsf{Real}^{\uparrow}$ rows, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

For the forward direction, assume public classifier agreement on the terminal four-face packet. The coherence theorem reads the accompanying exit-threshold coordinate from the same displayed record. Then Definition 38.458 and Definition 301.237 identify the terminal pullback source and the RegSeqRat source section uniquely, so the reflected source is exactly the stated four-face record.

For the reverse direction, assume the objectwise Phase Real and Completion exit-threshold read is displayed on that record. Use Definition 38.790 to keep the accepted source equal to the four-face record, then apply the classifier side of Definition 38.789 to recover public classifier agreement on the same packet.

It remains to keep the axes separate. Apply Definition 38.833 and Definition 38.834. The closure, verification, Lean-target, and bridge data stay attached to the four displayed faces and cannot be replaced by aggregate rows, while classifier agreement supplies no quotient stream, host equality, selected limit, choice principle, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 38.792 (Real L10 objectwise face-status exit)**.** *For any current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *rows, the packet satisfies the roadmap exit read exactly when the four local faces each supply their own closurestatus, formalstatus, leantarget, and bridgestatus rows, and the packet carries only the displayed finite support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, the exit certificate is the objectwise product of the four local face-status records together with those finite BEDC support rows.

The certificate contains no aggregate formalstatus *row, no aggregate bridge row, no quotient stream, no selected limit, no countable-choice coordinate, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim. The local face records remain the exact Phase Real and Completion exit certificate; there is no additional route-level status carrier.*

Proof. First read the exit threshold. By Definition 38.494, the current Phase Real and Completion threshold is the objectwise threshold on the four displayed faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next recover the local status rows. Apply Definition 38.471. It reads the roadmap audit coordinate from the local closurestatus, formalstatus, leantarget, and bridgestatus rows of those four faces, rather than from a single route credential.

For the forward direction, an accepted roadmap exit read therefore decomposes into exactly the four local face records, and the finite support of the packet is the displayed BHist, hsame, Cont, Pkg, and NameCert data. No further support coordinate is introduced by the threshold comparison.

For the reverse direction, suppose the four faces supply those local rows and the packet carries only the displayed BEDC support rows. The audit pullback reads these rows as the current Phase Real and Completion audit record, and the threshold equivalence identifies that record with the roadmap exit read.

It remains to rule out substitute status data. Use Definition 38.833 and Definition 38.834. They keep closure, verification, Lean-target, and bridge information attached to the four face

chapters, so no aggregate formalstatus, aggregate leantarget, or aggregate bridge row can replace the objectwise certificate.

Finally compare with the StreamName-side certificate of Definition 126.513. It has the same four-object exit read and the same finite support surface. Thus the Real-side packet is precisely the Phase Real and Completion exit certificate formed by the four local face-status records and the displayed BHist, hsame, Cont, Pkg, and NameCert rows, excluding quotient streams, selected limits, countable choice, metric-completeness data, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.793 (Real L10 exit certificate face minimality)**.** *Let $\mathcal{P}$ be an accepted Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose $\mathcal{P}$ satisfies the roadmap exit read by the objectwise four-face certificate of Definition 38.792. Then each of the four face records is indispensable: deleting the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or local* $\mathsf{Real}^{\uparrow}$ *face record makes the remaining data fail to satisfy the roadmap exit read.*

The same minimality holds for the displayed support surface. If any displayed BHist, hsame, Cont, Pkg, *or* NameCert *support row used by the four faces is removed, the remaining packet no longer carries an accepted Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate under the current Real L10 roadmap read.*

This is only a minimality statement for the objectwise exit certificate. It does not introduce quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10ExitCertificateFaceMinimality`

Proof. First identify the certificate whose minimality is being tested. By Definition 38.792, the accepted roadmap exit read is exactly the objectwise product of the four local face-status records together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next compare the exit read with the current audit route. Apply Definition 38.471. It reads the current Phase Real and Completion audit from those same local face records. Apply Definition 38.493 and Definition 38.494. They say that the roadmap exit criterion is satisfied by precisely that four-face objectwise threshold, not by a shortened route.

Now remove one face. If the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or local $\mathsf{Real}^{\uparrow}$ face is absent, the objectwise product required by the face-status theorem is no longer present. The sufficiency theorem therefore has no four-face certificate to consume, and the threshold equivalence cannot read an accepted roadmap exit from the remaining three faces.

Now remove one displayed support row. The face-status theorem lists BHist, hsame, Cont, Pkg, and NameCert as the finite support carried by the objectwise certificate. Removing any displayed row breaks the support surface on which the audit pullback and threshold equivalence are evaluated, so the remaining packet does not present the accepted exit certificate.

It remains to rule out replacement data. By Definition 38.781, classifier agreement cannot manufacture formalstatus, leantarget, bridgestatus, or bridge rows, and audit admissibility cannot manufacture classifier rows. By Definition 38.833 and Definition 38.834, the status and Lean-target data stay attached to the four displayed faces and cannot be replaced by aggregate rows.

Thus every face record and every displayed support row is indispensable for the current Real L10 roadmap exit read. The cited routes provide no quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or $\mathsf{Real}^{\uparrow}$ completeness coordinate that could fill the missing position. □

Theorem 38.794 (Real L10 exit certificate support no-confusion)**.** *Let $\mathcal{P}$ and $\mathcal{P}'$ be accepted Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificates over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose both certificates expose the same four local face-status records and the same displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{P}$ and $\mathcal{P}'$ determine the same Real L10 exit certificate and the same public finite-section classifier read.

Conversely, equality of Real L10 exit-certificate reads recovers equality of exactly those four local face-status records and the five displayed BEDC support rows. It recovers no quotient stream equality, no selected limit, no countable-choice selector, no metric-completeness datum, no aggregate formalstatus *row, no aggregate bridge row, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10ExitCertificateSupportNoConfusion`

Proof. First identify the surface on which both certificates are read. By Definition 38.792, an accepted Phase Real and Completion exit certificate is the objectwise product of the four face-status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ rows, together with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Since $\mathcal{P}$ and $\mathcal{P}'$ expose the same rows, they have the same displayed objectwise exit certificate.

Next rule out hidden alternatives. Apply Definition 38.793. None of the four face records and none of the five support rows can be deleted, replaced, duplicated, or detached while the accepted roadmap exit read is retained. Hence the common displayed surface determines one Real L10 exit-certificate read, so $\mathcal{P}$ and $\mathcal{P}'$ cannot separate after the rows listed in the statement have been fixed.

Now transport the same conclusion to the classifier. Apply Definition 38.789 to put the exit-threshold read and the public classifier read on one finite four-face record. Then apply Definition 38.795. The public finite-section classifier read is exactly the read of the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail classifier rows, the local $\mathsf{Real}^{\uparrow}$ terminal seal row, and the same five support rows. Therefore equal exit-certificate rows give equal public classifier reads.

For the converse direction, assume the two certificate reads are equal. The face-status theorem identifies the only recoverable rows as the four local face-status records and the displayed BHist, hsame, Cont, Pkg, and NameCert rows; minimality forces equality on each of those rows, because deleting or replacing any one of them would change the accepted certificate read.

Finally separate the certificate vocabulary from verification and bridge data. Use Definition 38.781, Definition 38.833, and Definition 38.834. These theorems keep formalstatus, Lean-target, and bridge information local to the four displayed faces and prevent an aggregate route row from being read back out of the certificate. The cited finite rows also contain no quotient stream, selected limit, countable-choice selector, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, so equality of certificate reads recovers none of those data. □

Theorem 38.795 (Real L10 finite-section classifier equivalence)**.** *Let $\mathcal{S}$ be a finite Real-and-Completion section over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Public finite-section classifier agreement on $\mathcal{S}$ is equivalent to the existence of one compatible four-face packet whose rows are exactly the dyadic common scale-window witness, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and common-tail classifier rows, the local* $\mathsf{Real}^{\uparrow}$

terminal seal row, and the support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The compatible packet is read objectwise on those four faces. It contains no quotient streams, selected limits, host real equality, countable-choice selectors, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First put the classifier and threshold readings on one record. Apply Definition 38.789. It identifies the public classifier coordinate and the objectwise exit-threshold coordinate as readings of the same finite four-face packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, and the displayed support rows.

Next identify the reflected source of that record. Apply Definition 38.791. The finite section therefore has the stated public classifier surface exactly when its classifier and exit reads are reflected from the same four-face source

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

We now compare the RegSeqRat-facing rows. Apply Definition 301.239 to read the local $\mathsf{Real}^{\uparrow}$ terminal seal from the RegSeqRat terminal face, and apply Definition 301.186 to identify the common-tail classifier rows with the Real boundary rows. This supplies the RegSeqRat regularity, common-tail, and terminal-seal components of the compatible packet.

The StreamName and DyadicRatCore faces are then forced by their finite-section exactness results. Use Definition 126.586 to recover the $\mathsf{StreamName}^{\uparrow}$ finite-window section, and use Definition 291.8 to recover the dyadic common scale-window witness. These two faces provide the lower part of the four-face packet and do not introduce any quotient or completeness coordinate.

For the forward direction, public finite-section classifier agreement supplies the four face records just described, and the cited exactness theorems repack them with BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ into the compatible packet. For the reverse direction, a compatible packet carrying exactly those rows feeds the reflected-source theorem and hence yields public finite-section classifier agreement on $\mathcal{S}$.

Finally apply Definition 38.793. Each face and each displayed support row is indispensable, so no shortened three-face section or support-free certificate can present the same classifier surface. Since the only rows used are the four local faces and the listed BEDC support rows, the construction has no source for quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.796 (Real L10 current-phase exit local threshold)**.** *Let* $\mathcal{E}$ *be a current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Then* $\mathcal{E}$ *satisfies the roadmap exit read exactly when the four objectwise faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *each supply their own scoped-closure and audit-clean local threshold rows, and the only support rows carried across the packet are*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, the current Phase Real and Completion exit criterion is the objectwise product of those four local threshold records, reflected through the same finite classifier section, and not a route-level aggregate credential. The packet has no quotient stream, no selected limit, no countable-choice selector, no metric-completeness datum, no aggregate formalstatus row, no aggregate bridge row, and no $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10CurrentPhaseExitLocalThreshold`

Proof. For the forward direction, assume $\mathcal{E}$ satisfies the roadmap exit read. By Definition 38.792, the exit read decomposes objectwise into the four face-status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ rows, together with only the displayed BHist, hsame, Cont, Pkg, and NameCert support.

The local threshold grade on each face is then read from that same objectwise certificate. Apply Definition 38.793. Since each face and each displayed support row is indispensable, the roadmap exit read cannot be witnessed by a shortened section, by an unsupported status row, or by an aggregate credential replacing the four local rows.

For the reverse direction, suppose the four faces each supply their scoped-closure and audit-clean local threshold rows and the packet carries only the displayed BEDC support rows. The objectwise face-status theorem repacks exactly these rows as the current Phase Real and Completion exit certificate, so the roadmap exit read is satisfied.

It remains to identify this certificate with the classifier section used by the roadmap. Apply Definition 38.791 to reflect the exit read and public classifier agreement from the same source

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then apply Definition 38.795 to read the finite section as exactly the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail classifier rows, the local $\mathsf{Real}^{\uparrow}$ terminal seal row, and the displayed support rows.

Finally combine the minimality and classifier equivalence statements. The four local faces are both necessary and sufficient for the current-phase exit read, and the finite classifier section introduces no extra coordinates. Thus quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and $\mathsf{Real}^{\uparrow}$ completeness are all absent from the theorem's certificate surface. □

Theorem 38.797 (Real L10 finite-request exit cover)**.** *Let $\mathcal{R}$ be a finite Real-and-Completion request section whose visible rows are exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-request section, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{R}$ satisfies the current Phase Real and Completion exit read if and only if those visible rows form the four-face objectwise certificate of Definition 38.796.

The cover is finite and objectwise. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First identify the finite request section. Apply Definition 38.795. It reads the visible finite section as exactly the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail classifier rows, the local $\mathsf{Real}^{\uparrow}$ terminal seal row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the forward direction, assume $\mathcal{R}$ satisfies the current Phase Real and Completion exit read. By Definition 38.796, that exit read is precisely the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, carried by the same displayed support rows. The finite-section equivalence identifies these rows with the visible request section, so the section forms the required four-face certificate.

For the reverse direction, suppose the visible rows of $\mathcal{R}$ form that four-face objectwise certificate. The current-phase local-threshold theorem consumes exactly those four face records and the displayed support rows, so the roadmap exit read is satisfied.

It remains to rule out shortened or enlarged covers. Apply Definition 38.793. Each of the four face records and each displayed support row is indispensable for the current Real L10 exit certificate, so a missing face, a missing support row, or an added aggregate credential cannot present the same exit read.

Therefore the finite request section is a cover of the current exit certificate exactly in the stated objectwise sense. The cited theorems provide no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.798 (Real L10 exit-certificate dependency factorization)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Then $\mathcal{E}$ satisfies the current-phase exit local threshold if and only if its accepted exit certificate factors through exactly the ordered dependency surface*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

In the forward direction, an accepted exit certificate decomposes into the four face records: the finite-tail $\mathsf{DyadicRatCore}^{\uparrow}$ *exit certificate, the* $\mathsf{StreamName}^{\uparrow}$ *ordered finite-window gate, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-section classifier record, and the local* $\mathsf{Real}^{\uparrow}$ *seal-facing threshold record, all carried by those same support rows. In the reverse direction, those four compatible face records repack to the accepted current-phase exit local threshold certificate for $\mathcal{E}$.*

This factorization excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness. It is only a dependency certificate for the finite Phase Real and Completion exit surface.*

Proof. For the forward direction, assume $\mathcal{E}$ satisfies the current-phase exit local threshold. Apply Definition 38.796. The threshold read is the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, with no support beyond BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Now identify the four components of that product. The DyadicRatCore component is the finite-tail exit-certificate surface of Definition 291.75. The StreamName component is constrained by Definition 126.615, so the consumer-facing read keeps the displayed StreamName window before the dyadic ledger and then the RegSeqRat and Real faces.

The middle and terminal components are read from the finite section. Apply Definition 38.795 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows together with the local $\mathsf{Real}^{\uparrow}$ terminal seal row. Then apply Definition 38.793: none of these four faces, and none of the displayed support rows, can be deleted or replaced by an aggregate credential.

For the reverse direction, suppose the packet displays exactly those four compatible face records over the ordered surface and the displayed support rows. The finite-section classifier equivalence repacks the StreamName, RegSeqRat, dyadic scale-window, and local Real rows as one finite request section, while the StreamName ordered exit gate supplies the consumer-facing order.

The DyadicRatCore finite-tail exit-certificate equivalence supplies the first visible face of the objectwise certificate. Combining that dyadic face with the StreamName, RegSeqRat, and Real

rows gives the four-face certificate consumed by Definition 38.797. Hence the visible section satisfies the current Phase Real and Completion exit read.

Finally apply Definition 38.796 again to identify that exit read with the current-phase local threshold for $\mathcal{E}$. Since every cited theorem works only with the four displayed faces and the rows BHist, hsame, Cont, Pkg, and NameCert, there is no coordinate from which to form quotient streams, selected limits, countable-choice data, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.799 (Real L10 construction-consumer order square)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose $\mathcal{E}$ satisfies the current-phase exit local threshold. Then the construction-source projection*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and the consumer-readback projection

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

are the two visible edge readings of one commuting square of the same displayed four-face exit certificate.

The top edge is the Real-side source reflection, the left edge is the StreamName finite-window readback, the right edge is the RegSeqRat-to-Real seal read, and the bottom edge is the shared objectwise certificate. Commutativity means that reading the packet by the construction source and then consuming it through the StreamName-facing gate yields the same four local face records and the same displayed support rows as first reading the consumer gate and then projecting to the Real-side source. The square supplies no quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First read the Real-side source. By Definition 38.791, the classifier and exit-threshold coordinates of $\mathcal{E}$ are reflected from the displayed source

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

By Definition 38.796, this source carries exactly the four objectwise local threshold records and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next read the consumer-facing order. Apply Definition 126.615. The same accepted Phase Real and Completion seal-facing read is consumed along

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

where the StreamName finite window is read before the dyadic ledger, the RegSeqRat readback is anchored to those rows, and the RealUp seal is read last.

The two readings now have the same certificate target. Use Definition 38.793: each of the four face records and each displayed support row is indispensable for the current Real L10 exit certificate. Therefore neither route can replace, delete, or duplicate a face while still presenting the accepted exit read.

Finally compare the edge maps. The Real-side reflection gives the construction-source edge, the StreamName ordered gate gives the consumer-readback edge, and the local-threshold theorem identifies both with the same objectwise certificate. Since face minimality leaves only those four

faces and those support rows, the square commutes as a comparison of projections of one displayed certificate, not as a second completion route.

All excluded data remain absent. The cited theorems export only the finite face rows and the displayed support rows, so the square has no source for quotient stream equality, selected limits, countable choice, metric completeness, aggregate formalstatus, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.800 (Real L10 finite-request exit cocycle)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion request packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose $\mathcal{E}$ satisfies the current-phase exit local threshold. Then the DyadicRatCore finite-tail exit certificate, the StreamName finite-request section pullback exactness, the RegSeqRat four-face exit-threshold equivalence, and the Real finite-request exit cover form one associative cocycle on the same four-face source.*

Equivalently, the construction-source-then-consumer-readback composite and the RegSeqRat-threshold-then-Real-exit-cover composite recover exactly the same four face records and the same support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The cocycle is only a finite packet identity among these four visible readings. It supplies no quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the dyadic face. By Definition 291.75, the visible $\mathsf{DyadicRatCore}^{\uparrow}$ input to $\mathcal{E}$ is exactly the finite-tail exit certificate carried by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. This gives the first edge of the finite four-face source.

Next attach the request section. Apply Definition 126.581. The StreamName finite request section is pulled back over that same dyadic face, so the consumer readback does not introduce a second support surface or a second certificate target.

Now read the terminal threshold. Apply Definition 301.218. The RegSeqRat threshold is the terminal four-face reading of the dyadic and StreamName data together with the local $\mathsf{Real}^{\uparrow}$ seal row, and it keeps the same five support rows.

The Real exit cover is the last face. By Definition 38.797, the finite request section covers exactly the Real exit certificate visible from the same four faces. Thus the RegSeqRat-threshold-then-Real-exit-cover route returns the same records as the preceding dyadic and StreamName pullback route.

It remains to compare the two edge orders. Use Definition 38.799: the construction-source projection and the consumer-readback projection are two readings of one commuting square of the displayed four-face exit certificate. Therefore composing through the construction edge before the consumer edge, or through the RegSeqRat terminal threshold before the Real cover, is associative as a cocycle on the same finite packet.

All exclusions are inherited from the cited faces. None of them exports quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness, so the cocycle cannot add such data. □

Theorem 38.801 (Real L10 support-row reindexing invariance)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet satisfying the current-phase exit local threshold of Definition 38.796. Let O and O' be accepted support-row-preserving reindexings or projections of the ordered terminal route fixed by Definition 126.615:*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that both readings preserve the same displayed face rows, the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and the same local* $\mathsf{Real}^{\uparrow}$ *seal row. Assume also that neither reading detaches the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger from the* $\mathsf{StreamName}^{\uparrow}$ *finite window, reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *row before the dyadic predecessor face, or reads the* $\mathsf{Real}^{\uparrow}$ *seal before the required predecessor faces. Then O and O' determine the same objectwise Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate.*

The invariant certificate is only the finite four-face objectwise product over the ordered terminal route and its displayed support rows. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First fix the public route. Apply Definition 126.615. An accepted seal-facing read satisfies the public L10 exit gate only along

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

with the finite StreamName window before the dyadic ledger, the RegSeqRat readback after the anchored ledger, and the RealUp seal last. Thus O and O' are not arbitrary face permutations; they are support-row-preserving views of that same ordered terminal route.

Next identify the certificate carried by that route. By Definition 38.796, the current Phase Real and Completion exit criterion is the objectwise product of the four local threshold records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, together with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Use minimality to control the allowed reindexings and projections. Apply Definition 38.793. Since every one of the four face records and every displayed support row is indispensable, an accepted support-row-preserving view cannot delete, duplicate, detach, or replace a face or support row while still presenting the same current Real L10 exit certificate.

Now compare O and O' as finite classifier sections of the same terminal packet. By Definition 38.795, every accepted finite-section read with these four faces and these support rows is a compatible four-face packet over the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the local $\mathsf{Real}^{\uparrow}$ terminal seal row.

Thus both O and O' are projections of the same compatible finite packet over the ordered route. The construction-consumer square of Definition 38.799 is one visible comparison inside this finite certificate, and the present hypotheses restrict the comparison to order-preserving support reindexings or projections of the same four-face certificate.

The conclusion is therefore support-row reindexing invariance of the displayed objectwise exit certificate, not a second route to completion and not a permission to permute the public L10 construction faces. The cited theorems provide no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness, so these data remain outside the invariant certificate. □

Theorem 38.802 (Real L10 face-local formal upgrade invariance)**.** *Let $\mathcal{E}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *rows satisfying the current-phase exit local threshold of Definition 38.796. Let F be one of the four displayed faces. Form $\mathcal{E}_F^{+}$ by replacing only the local formal-status row attached to F by a strictly stronger local formal-status row, or by replacing only the leantarget row or bridgestatus row attached to F by a replacement row attached to the same face whose recorded target or bridge status refines the same local face without changing the other three face records.*

Assume that the displayed support rows BHist, hsame, Cont, Pkg, *and* NameCert *are unchanged. Then* $\mathcal{E}_F^+$ *determines the same four-face Phase* $\mathsf{Real}^\uparrow$ *and Completion exit certificate as* $\mathcal{E}$*. Conversely, any packet with the same four-face exit certificate and with all non-*F *face records and displayed support rows fixed differs from* $\mathcal{E}$ *only by a face-local formal-status strengthening on* F*, or by a face-local leantarget or bridgestatus refinement attached to* F *that preserves the same displayed certificate surface.*

This invariance is purely face-local. It does not introduce route-level formalstatus, aggregate leantarget rows, aggregate bridge rows, selected limits, quotient streams, metric-completeness data, $\mathsf{Real}^\uparrow$ *completeness, or any replacement for the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Proof. First identify the certificate whose stability is being tested. By Definition 38.792, the accepted Phase $\mathsf{Real}^\uparrow$ and Completion exit certificate is the objectwise product of the four local face-status records together with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the forward direction, first consider a strictly stronger local formal-status row attached to F. The other three face records and all displayed support rows are unchanged, so the objectwise product still has the same four faces and the same finite BEDC support surface. By Definition 38.833, the stronger formal-status data remains attached to F and cannot become a route-level aggregate row.

The leantarget and bridgestatus alternatives are same-face refinements rather than entries in a shared strengthening order. Their recorded target or bridge status remains attached to F, refines that local face, and preserves the other three face records together with the displayed support rows. By Definition 38.834, the Lean-target refinement cannot become an aggregate route row, and the objectwise face-status certificate keeps bridge-status information on the displayed local face surface.

Now use support-row invariance. Apply Definition 38.801. Since the visible BHist, hsame, Cont, Pkg, and NameCert rows are fixed and no face is detached or reordered outside the accepted terminal route, the finite BEDC packet presents the same objectwise exit certificate after the local strengthening.

For the converse direction, suppose a packet presents the same four-face exit certificate while the three non-F face records and all displayed support rows are fixed. By Definition 38.793, none of the four faces and none of the displayed support rows can be deleted, aggregated, duplicated, or replaced while the same current Real L10 exit certificate is retained.

It remains only to locate the possible difference. The non-F records are fixed by hypothesis, and minimality fixes the displayed support surface. The noncollapse theorems keep formalstatus and Lean-target rows attached to their source face, while the objectwise face-status theorem keeps bridge-status information local to the four displayed faces. Hence any remaining difference is either a stronger local formal-status row on F, or a same-face leantarget or bridgestatus refinement on F that preserves the displayed certificate surface.

All excluded data remain absent. The cited results use only the four face records and the displayed support rows; they provide no route-level formalstatus, aggregate bridge coordinate, selected limit, quotient stream, metric-completeness datum, or $\mathsf{Real}^\uparrow$ completeness row. Thus the certificate is invariant precisely under the stated face-local formal strengthening and same-face leantarget or bridgestatus refinement. □

Theorem 38.803 (Real L10 face formal upgrade chain invariance)**.** *Let*

$$\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_n$$

be a finite sequence of Phase $\mathsf{Real}^{\uparrow}$ *and Completion packets over the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Assume $\mathcal{E}_0$ *satisfies the current-phase exit local threshold of Definition 38.796. For each adjacent step* $\mathcal{E}_i \rightsquigarrow \mathcal{E}_{i+1}$*, assume exactly one face is changed, the changed row is a same-face formalstatus strengthening, leantarget refinement, or bridgestatus refinement, and all other face records are fixed. Assume also that every step preserves the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then every intermediate packet $\mathcal{E}_i$ *determines the same objectwise Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate read by Definition 38.796.*

The invariant certificate is only the finite four-face exit certificate. The sequence supplies no aggregate formalstatus row, aggregate leantarget row, aggregate bridge row, quotient stream, selected limit, countable-choice selector, metric-completeness datum, standard bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Induct on the finite sequence. The base case is the one-packet sequence, where the exit certificate is the certificate of $\mathcal{E}_0$ read by Definition 38.796.

For the induction step, suppose $\mathcal{E}_i$ has the same objectwise exit certificate as $\mathcal{E}_0$. The adjacent move $\mathcal{E}_i \rightsquigarrow \mathcal{E}_{i+1}$ changes only one of the four displayed faces, and the change is either a stronger local formalstatus row, a same-face leantarget refinement, or a same-face bridgestatus refinement. The rows BHist, hsame, Cont, Pkg, and NameCert remain fixed.

Apply Definition 38.802 to this adjacent move. It identifies the exit certificate of $\mathcal{E}_{i+1}$ with the exit certificate of $\mathcal{E}_i$, because the refinement remains attached to its source face and does not alter the other three face records or the displayed support surface.

Finally use transitivity of the certificate comparison supplied by the induction hypothesis. The noncollapse theorems Definition 38.833 and Definition 38.834, together with the minimality theorem Definition 38.793, exclude aggregate formalstatus rows, aggregate bridge rows, quotient streams, selected limits, metric-completeness data, and $\mathsf{Real}^{\uparrow}$ completeness at every step. Hence each intermediate packet reads the same finite four-face Phase Real and Completion exit certificate. □

Theorem 38.804 (Real L10 exit-threshold face-local correspondence)**.** *Let* $\mathcal{E}$ *be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *rows. Then* $\mathcal{E}$ *satisfies the current-phase exit local threshold of Definition 38.796 if and only if it exhibits four compatible local face records over the same displayed support rows:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-tail certificate,*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *ordered finite-window gate,*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-section classifier record,*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal-facing threshold record.*

Compatibility means that all four records are read over the same displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, and that no face record is detached from the ordered terminal route.*

This correspondence is exactly local-record correspondence for the four displayed faces, not an aggregate audit credential. It recovers no quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, read the threshold by Definition 38.796. The accepted exit read is the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, together with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Now factor that read through the displayed route. Apply Definition 38.798. The dyadic finite-tail certificate, the ordered StreamName finite-window gate, the RegSeqRat finite-section classifier record, and the local Real seal-facing threshold record are the four ordered faces of the same exit certificate rather than independent summaries.

Minimality fixes the vocabulary of the correspondence. By Definition 38.793, none of the four face records and none of the displayed support rows can be deleted, duplicated, detached, or replaced while the same current Real L10 exit certificate is retained. Thus the forward read produces exactly the listed local records over the common support surface.

For the reverse direction, assume the four compatible local records are exhibited over the same displayed BHist, hsame, Cont, Pkg, and NameCert rows. Use Definition 38.805 to compare this displayed four-face section with the finite packet $\mathcal{E}$: the support-row-preserving comparison recovers the same objectwise certificate read.

Finally apply Definition 38.796 again. Since the recovered objectwise certificate is precisely the product of the dyadic finite-tail certificate, the ordered StreamName finite-window gate, the RegSeqRat finite-section classifier record, and the local Real seal-facing threshold record, the packet satisfies the current-phase exit local threshold.

Face-local formal changes do not alter the equivalence. By Definition 38.802, any strengthening of a local status, leantarget, or bridge-status row remains attached to its source face and cannot become a route-level aggregate credential. Consequently the correspondence has no coordinate for quotient streams, selected limits, countable-choice selectors, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.805 (Real L10 exit certificate refinement functoriality)**.** *Let $\mathcal{E}$ and $\mathcal{E}'$ be finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose $\rho : \mathcal{E} \to \mathcal{E}'$ is a certified finite refinement whose visible action preserves each displayed face row and preserves the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then $\mathcal{E}$ and $\mathcal{E}'$ determine the same objectwise Phase Real and Completion exit certificate read by Definition 38.796.*

Conversely, if $\mathcal{E}$ and $\mathcal{E}'$ determine the same objectwise Phase Real and Completion exit certificate, then the equality recovers exactly a certified finite refinement between their displayed four-face sections that preserves the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *rows and preserves the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

This refinement law is confined to the displayed finite certificate surface. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read both packets through the current exit criterion. By Definition 38.796, each accepted packet is read as the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, together with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the forward direction, apply the certified finite refinement ρ. Its visible action preserves every one of the four face rows and every displayed support row, so the finite-section read of $\mathcal{E}'$ has the same rows as the finite-section read of $\mathcal{E}$. By Definition 38.795, the two finite sections therefore present the same compatible four-face classifier packet.

Now control possible reorderings or local status changes introduced by the refinement. Apply Definition 38.801 to identify support-row-preserving views of the terminal route with the same objectwise exit certificate. Apply Definition 38.802 to keep any face-local formal strengthening attached to its source face rather than to an aggregate route row.

For the converse direction, assume that $\mathcal{E}$ and $\mathcal{E}'$ determine the same objectwise exit certificate. By Definition 38.793, none of the four face rows and none of the displayed support rows can be deleted, duplicated, detached, or replaced while the same certificate is retained.

The finite-section classifier equivalence then reconstructs the only possible comparison between the two visible sections: it is the support-row-preserving finite refinement carrying the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ rows to the corresponding rows and carrying the displayed BHist, hsame, Cont, Pkg, and NameCert rows to themselves. Minimality makes this refinement exact, because there is no unused face coordinate or hidden support coordinate on which a different refinement could act.

All exclusions are inherited from the same cited certificate surface. The argument uses only finite face rows, finite classifier sections, support-row-preserving route views, and face-local formal strengthening; it has no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.806 (Real L10 exit certificate refinement composition)**.** *Let $\mathcal{E}_0$, $\mathcal{E}_1$, and $\mathcal{E}_2$ be finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose*

$$\mathcal{E}_0 \xrightarrow{\rho_{01}} \mathcal{E}_1 \xrightarrow{\rho_{12}} \mathcal{E}_2$$

is a pair of certified finite refinements. Assume that each leg preserves the four displayed face rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$, *and preserves the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then the composite refinement $\rho_{12} \circ \rho_{01}$ preserves the same displayed face and support rows, and $\mathcal{E}_0$ and $\mathcal{E}_2$ determine the same objectwise Phase Real and Completion exit certificate read by Definition 38.796.*

Equivalently, a finite chain of two certified refinements cannot create any additional exit-coordinate between the endpoints. The endpoint certificate is still exactly the compatible four-face packet on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *face, together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

This composition law is confined to finite support-row-preserving refinements. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 38.805 to the first leg ρ_{01}. The hypotheses say that the visible action of ρ_{01} preserves each of the four displayed face rows and every displayed support row, so $\mathcal{E}_0$ and $\mathcal{E}_1$ determine the same objectwise exit certificate.

Apply the same theorem to the second leg ρ_{12}. The second visible action preserves the same four face rows and the same BHist, hsame, Cont, Pkg, and NameCert rows, so $\mathcal{E}_1$ and $\mathcal{E}_2$ determine the same objectwise exit certificate.

Now identify the intermediate certificate. By Definition 38.795, the finite sections read from $\mathcal{E}_0$, $\mathcal{E}_1$, and $\mathcal{E}_2$ are compatible four-face packets over the same displayed terminal surface: the dyadic common scale-window witness, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the local $\mathsf{Real}^{\uparrow}$ seal row. Hence the equality through $\mathcal{E}_1$ is equality of the same finite classifier packet, not merely agreement of unrelated summaries.

It remains to read the composite. Since both legs preserve every displayed face row and every displayed support row, their composite also sends each $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ row to the corresponding endpoint row and sends the displayed BHist, hsame, Cont, Pkg, and NameCert rows to the corresponding endpoint rows. Transitivity of the two objectwise certificate identifications gives the endpoint certificate equality between $\mathcal{E}_0$ and $\mathcal{E}_2$.

Finally apply Definition 38.793. Minimality rules out any hidden coordinate on which the composite could act while preserving the current Real L10 exit certificate: no quotient stream, selected limit, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or $\mathsf{Real}^{\uparrow}$ completeness row is part of the finite certificate surface. Thus the two-leg chain composes only as a support-row-preserving refinement of the same displayed objectwise exit certificate. □

Theorem 38.807 (Real L10 exit certificate refinement zigzag invariance)**.** *Let $\mathcal{E}_0, \ldots, \mathcal{E}_n$ be a finite zigzag of Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Assume that every adjacent step is a certified support-row-preserving finite refinement in one direction or the other, preserving the four displayed face rows and preserving the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows. Then all packets $\mathcal{E}_i$ determine the same objectwise Phase Real and Completion exit certificate read by Definition 38.796.*

Conversely, let $\mathcal{E}_0, \ldots, \mathcal{E}_n$ be a finite family of Phase $\mathsf{Real}^{\uparrow}$ *and Completion packets over the same four displayed faces. If all packets determine one common objectwise Phase Real and Completion exit certificate, then the family admits a common support-row-preserving refinement zigzag through the displayed four-face sections. Each adjacent comparison preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *rows and preserves the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

This finite-zigzag invariant is confined to the displayed four-face certificate surface. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, induct on the length of the finite zigzag. The empty zigzag and the one-packet zigzag read a single packet through Definition 38.796, so there is no comparison to transport.

For the induction step, remove the last adjacent comparison. The induction hypothesis identifies the objectwise exit certificate of every earlier packet with the certificate of the penultimate packet. The final adjacent comparison is a certified support-row-preserving finite refinement in one direction or the other, so Definition 38.805 identifies the penultimate and final certificate readings. Transitivity of equality of the objectwise certificate readings composes these identifications and gives one certificate for the whole zigzag.

For the converse direction, assume the finite family has one common objectwise exit certificate. Apply Definition 38.793. Since the four displayed face rows and the displayed BHist, hsame, Cont, Pkg, and NameCert rows are indispensable, no member of the family can realize that common certificate by deleting, duplicating, detaching, or replacing one of those rows.

Next apply Definition 38.795. Each packet therefore has a displayed four-face section over the same dyadic common scale-window witness, the same finite $\mathsf{StreamName}^{\uparrow}$ window, the same $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the same local $\mathsf{Real}^{\uparrow}$ terminal seal row. The only possible adjacent comparisons between consecutive displayed sections are the support-row-preserving certified finite refinements recovered by Definition 38.805.

Ordering the finite family by its displayed indices and inserting these recovered adjacent comparisons gives the required common refinement zigzag. Its support surface is exactly the four-face

section just described, so the construction has no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.808 (Real L10 exit refinement common envelope)**.** *Let*

$$\mathcal{E}_0, \ldots, \mathcal{E}_n$$

be a finite zigzag of Phase $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Assume that every adjacent step is a certified support-row-preserving finite refinement in one direction or the other, preserving the four displayed face rows and preserving the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then there is a single finite envelope record carrying exactly the four face rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

and exactly the displayed support rows

$$\mathsf{BHist}, \qquad \mathsf{hsame}, \qquad \mathsf{Cont}, \qquad \mathsf{Pkg}, \qquad \mathsf{NameCert},$$

through which every packet's objectwise Phase Real and Completion exit certificate factors.

The envelope is the common finite four-face certificate surface, not an ambient completion object. It has no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First identify the common certificate. Apply Definition 38.807 to the finite zigzag. The adjacent support-row-preserving refinements give one objectwise Phase Real and Completion exit certificate shared by every $\mathcal{E}_i$.

Next decompose that shared certificate into its local faces. By Definition 38.804, the common objectwise certificate is represented by the dyadic finite-tail certificate, the ordered $\mathsf{StreamName}^{\uparrow}$ finite-window gate, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-section classifier record, and the local $\mathsf{Real}^{\uparrow}$ seal-facing threshold record, all read over the same displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Now order the four records through the dependency surface. Apply Definition 38.798. The face records factor through the dyadic source, the stream-name window, the regular-sequence common-tail classifier, and the local Real terminal seal in that order, so they form one finite envelope record rather than four unrelated local summaries.

It remains to show that the envelope has exactly the displayed vocabulary. The current-phase threshold theorem Definition 38.796 supplies only the four displayed face rows and the support rows BHist, hsame, Cont, Pkg, and NameCert. The minimality theorem Definition 38.793 then rules out deleting, duplicating, detaching, or replacing any of those rows while preserving the same exit certificate.

The same minimality argument excludes every forbidden coordinate. Quotient streams, selected limits, countable-choice selectors, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ completeness are not part of the finite four-face certificate surface. Hence the common objectwise certificate of each $\mathcal{E}_i$ factors through the single finite envelope and through no larger hidden carrier. □

Theorem 38.809 (Real L10 finite-request subfamily exit stability)**.** *Let* $\{\mathcal{E}_i\}_{i \in I}$ *be a finite family of Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the*

local $\mathsf{Real}^{\uparrow}$ *rows. Suppose the family has one common objectwise Phase Real and Completion exit certificate, and suppose that every finite request subfamily* $J \subseteq I$ *preserves the four displayed face rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

Then each finite request subfamily J *determines the same objectwise Phase Real and Completion exit certificate as the ambient family. Conversely, any subfamily that determines that same certificate is obtained by restricting the displayed four-face sections and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows to the retained requests.*

This stability is only a finite-request restriction statement. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the ambient family through the common certificate. Apply Definition 38.807. Since all members determine one objectwise Phase Real and Completion exit certificate, the family admits a common support-row-preserving refinement zigzag through the displayed four-face sections.

Now restrict to a finite request subfamily $J \subseteq I$. The RegSeqRat-side restriction source is Definition 301.199: finite subfamily restriction of the common-tail terminal section cover keeps exactly the retained $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows and adds no hidden completion or choice coordinate.

Identify the restricted Real L10 section. By Definition 38.795, the retained finite request section is still the compatible four-face packet over the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the local $\mathsf{Real}^{\uparrow}$ terminal seal row, together with the displayed support rows.

Minimality prevents the restriction from changing the certificate vocabulary. Apply Definition 38.793. None of the four face rows and none of the displayed BHist, hsame, Cont, Pkg, and NameCert support rows can be deleted, replaced, duplicated, or detached while retaining the same current Real L10 exit certificate.

Thus the restricted subfamily has the same objectwise certificate as the ambient family. For the converse direction, suppose a subfamily has that same certificate. The same minimality theorem forces its visible rows to be exactly retained rows from the ambient four-face section and displayed support surface, and the finite-section classifier equivalence identifies the retained rows as the restricted finite section. Hence the subfamily is precisely the restriction of the four displayed face sections and support rows to its retained requests.

The exclusion clauses follow from the same row accounting. The cited Real L10 and RegSeqRat theorems expose only finite four-face sections and the displayed support rows; they provide no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.810 (Real L10 finite-request intersection exit exactness)**.** *Let* $\{\mathcal{E}_i\}_{i\in I}$ *be a finite family of Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose the ambient family has one common objectwise Phase Real and Completion exit certificate, and let* $J, K \subseteq I$ *be finite request subfamilies whose restrictions preserve the four displayed face rows together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

Then the finite intersection $J \cap K$ *determines the same objectwise exit certificate as the two restricted request subfamilies exactly when the retained* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows agree on* $J \cap K$. *In that case the intersection certificate is the objectwise restriction of the common Phase Real and Completion certificate to the retained requests, and no further exit coordinate is present on the overlap.*

This exactness is confined to finite overlaps of request subfamilies inside the displayed Real L10 four-face surface. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First restrict each request subfamily. By Definition 38.809, the J-restriction and the K-restriction both determine the same objectwise Phase Real and Completion exit certificate as the ambient family, and each restriction keeps exactly the displayed four-face rows and the displayed support rows on its retained requests.

Now pass to the overlap. Apply Definition 301.201 to the RegSeqRat common-tail terminal sections underlying the two restrictions. The theorem identifies the retained $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows on the finite meet $J \cap K$, and it supplies no quotient stream, selected limit, countable-choice row, or hidden completion datum.

Use Definition 38.795 to read that finite meet as a Real L10 section. The retained overlap is therefore the same compatible four-face packet over the dyadic common scale-window witness, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the local $\mathsf{Real}^{\uparrow}$ terminal seal row, together with the same displayed support rows.

It remains to prove exactness rather than mere compatibility. Apply Definition 38.793. Since every displayed face row and every displayed support row is indispensable for the current Real L10 exit certificate, agreement of those retained rows on $J \cap K$ is sufficient for the same objectwise exit read, and any disagreement among them prevents the two restricted presentations from determining one overlap certificate.

Thus $J \cap K$ determines the same objectwise exit certificate precisely in the stated rowwise case. The cited restriction, intersection, classifier, and minimality results account for all certificate coordinates, so aggregate status rows, aggregate bridge rows, standard bridge rows, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be inserted on the overlap. □

Theorem 38.811 (Real L10 finite-request union exit exactness)**.** *Let* $\{\mathcal{E}_i\}_{i \in I}$ *be a finite family of Phase* $\mathsf{Real}^{\uparrow}$ *and Completion packets over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose the ambient family has one common objectwise Phase Real and Completion exit certificate, and let* $J, K \subseteq I$ *be finite request subfamilies whose restrictions preserve the four displayed face rows together with the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. Suppose also that the retained rows on* J, *on* K, *and on* $J \cap K$ *agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Then the finite union $J \cup K$ *determines the same objectwise Phase Real and Completion exit certificate as the two restricted request subfamilies exactly when the displayed union contains no rows beyond the four listed faces and the displayed support rows. In that case the union certificate is the objectwise restriction of the common Phase Real and Completion certificate to* $J \cup K$.

This exactness is confined to finite unions of request subfamilies inside the displayed Real L10 four-face surface. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First restrict each side. By Definition 38.809, the J-restriction and the K-restriction both determine the same objectwise Phase Real and Completion exit certificate as the ambient family,

and each restriction keeps exactly the displayed four-face rows and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

The overlap hypothesis is the compatibility datum for gluing. Apply Definition 38.810 to $J \cap K$. The common rows on the meet identify the two restricted presentations on their shared retained requests, so the union can be read as one finite request surface rather than as two unrelated certificates.

Now assemble the four-face union surface. Use Definition 126.305 and Definition 126.324 for the finite $\mathsf{StreamName}^{\uparrow}$ window rows, and use Definition 301.181 for the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail classifier rows. These union inputs preserve the dyadic common scale-window witness, the local $\mathsf{Real}^{\uparrow}$ terminal seal row, and the displayed support rows, and they introduce no quotient stream, selected limit, countable-choice row, or hidden completion datum.

Read the assembled union through Real L10. By Definition 38.795, the displayed $J \cup K$ surface is a single compatible four-face packet over the dyadic common scale-window witness, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail rows, and the local $\mathsf{Real}^{\uparrow}$ terminal seal row, together with the same BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 38.793. If the displayed union carries only those four faces and support rows, minimality makes the union read the same objectwise exit certificate as the two restrictions and the ambient family. Conversely, if an additional quotient, selected-limit, choice, metric-completeness, aggregate-status, aggregate-bridge, standard-bridge, or $\mathsf{Real}^{\uparrow}$ completeness coordinate is present, it is not part of the minimal Real L10 certificate surface, so the enlarged union is no longer exactly the same objectwise exit certificate. □

Theorem 38.812 (Real L10 finite-request sheaf exactness)**.** *Let $\{\mathcal{E}_i\}_{i\in I}$ be a finite family of Phase $\mathsf{Real}^{\uparrow}$ and Completion packets over*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

Let $\{J_a\}_{a\in A}$ be a finite cover of I by finite request subfamilies. For each $a \in A$, let $\mathcal{C}_a$ be a local Phase Real and Completion exit presentation on J_a.

Then the family $\{\mathcal{C}_a\}_{a\in A}$ determines a unique objectwise Phase Real and Completion exit certificate on I if and only if each $\mathcal{C}_a$ displays exactly the four face rows

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, and for every finite overlap $J_{a_0} \cap \cdots \cap J_{a_m}$ the restricted presentations agree on all four face rows and on all displayed support rows.*

In that case the global certificate is the objectwise restriction-gluing of the common Phase Real and Completion certificate across the finite cover. This sheaf exactness is confined to the four-face Real L10 surface. It excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. For existence, first pass to each cover member. By Definition 38.809, every J_a-presentation with the four displayed face rows and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows reads as the restriction of the same objectwise Phase Real and Completion certificate to that finite request subfamily.

The overlap hypotheses are exactly the compatibility data. On a binary overlap, apply Definition 38.810; on a finite iterated overlap, repeat the same intersection exactness along the finite

meet. Thus the cover presentations agree as Real L10 sections on every retained request belonging to more than one cover member.

Now glue over the finite cover. Induct on the finite index set A. The empty and singleton cases are immediate from subfamily stability. For the induction step, glue the already assembled finite union to the next cover member by Definition 38.811. The overlap compatibility just proved supplies the meet datum, and the union theorem preserves exactly the four displayed faces and the displayed support rows.

It remains to identify the glued object as the Phase Real and Completion exit certificate rather than as a larger completion object. Use Definition 38.795 to read the glued finite union as one compatible four-face packet, and use the RegSeqRat common-tail sheaf exactness supplied by Definition 301.201 and Definition 301.181 to keep the common-tail rows objectwise on the finite cover.

For uniqueness, suppose two global exit certificates restrict to the same cover presentations. By Definition 38.793, the four face rows and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows are indispensable and exhaust the Real L10 certificate surface. Since the cover is finite and covers every request in I, equality on the cover members gives equality of every displayed row of the two global certificates.

Conversely, if a member omits one of the four face rows or one of the displayed support rows, minimality prevents it from being a restriction of the Real L10 exit certificate. If two presentations disagree on a finite overlap, intersection exactness prevents them from determining a single overlap certificate, so no unique global objectwise certificate can glue from the cover.

All nonescape clauses follow from the same row accounting. The proof uses only finite subfamily restriction, finite intersection exactness, finite union exactness, face minimality, and RegSeqRat common-tail exactness; none of these results adds a quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate status row, aggregate bridge row, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 38.813 (Real L10 finite-cover support removal obstruction)**.** *Let $\{J_a\}_{a\in A}$ be a finite cover of a finite request family for Phase $\mathsf{Real}^{\uparrow}$ and Completion packets over*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

For each $a \in A$, let $\mathcal{C}_a$ be a local Real L10 presentation on J_a whose intended certificate is the finite-request sheaf certificate of Definition 38.812. If one local presentation omits any displayed support row among

$$\mathsf{BHist}, \qquad \mathsf{hsame}, \qquad \mathsf{Cont}, \qquad \mathsf{Pkg}, \qquad \mathsf{NameCert},$$

then the finite cover cannot determine the same objectwise Real L10 exit certificate.

The failure is already witnessed on a finite cover member. It cannot be repaired by quotient streams, selected limits, countable choice, metric completeness, aggregate formalstatus rows, aggregate bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness.

Proof. Choose a cover member J_a whose local presentation omits one displayed support row. Apply Definition 126.665 to the $\mathsf{StreamName}^{\uparrow}$ finite-window component of that member: removing a displayed support coordinate leaves a finite obstruction on the cover member itself, not only after passage to the whole family.

Refinement cannot hide the omission. By Definition 126.666, every finite refinement of the cover still has a member over the same request locus where the omitted support row is absent. Thus the

obstruction survives finite cover subdivision and cannot be transported into a quotient-stream or selected-limit coordinate.

Now read the obstructed member through Real L10. Apply Definition 38.795: a local Real L10 section presents the objectwise certificate exactly by the dyadic common scale-window witness, the $\mathsf{StreamName}^{\uparrow}$ finite-window section, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and common-tail classifier rows, the local $\mathsf{Real}^{\uparrow}$ terminal seal row, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Minimality identifies the contradiction. By Definition 38.793, none of the displayed support rows can be deleted, duplicated, detached, or replaced while the same Real L10 exit certificate is retained. Hence the obstructed local member cannot be a restriction of the same objectwise certificate.

Finally apply Definition 38.812. A finite cover determines the global Real L10 certificate only when every cover member displays all four face rows and all five support rows, with overlap agreement on those rows. Since one member lacks a required support row, the exactness criterion fails on that member. The listed completion, choice, status, bridge, and $\mathsf{Real}^{\uparrow}$ completeness data are outside the finite certificate surface identified above, so they provide no repair coordinate. □

Theorem 38.814 (Real L10 finite-cover face removal obstruction)**.** *Let $\{J_a\}_{a\in A}$ be a finite cover of a finite request family for Phase $\mathsf{Real}^{\uparrow}$ and Completion packets over*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

For each $a \in A$, let $\mathcal{C}_a$ be a local Real L10 presentation on J_a whose intended certificate is the finite-request sheaf certificate of Definition 38.812. If one local presentation omits any one of the four face rows

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

then the finite cover cannot determine the same objectwise Real L10 exit certificate.

The obstruction is a face-row obstruction, not a support-row obstruction in disguise. It cannot be repaired by quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. Choose a cover member J_a whose local presentation omits one of the four displayed face rows. By Definition 38.812, a finite cover determines the global Real L10 certificate only when every cover member displays all four face rows and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows, with overlap agreement on those rows.

The missing face is indispensable. Apply Definition 38.793. Deleting the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or local $\mathsf{Real}^{\uparrow}$ row removes one object of the four-face certificate, so the obstructed local member cannot be a restriction of the same objectwise Real L10 exit certificate.

The StreamName and RegSeqRat boundary clauses give the same obstruction on the two middle faces. Apply Definition 126.501 to the $\mathsf{StreamName}^{\uparrow}$ open-phase row, and apply Definition 301.239 to the $\mathsf{RegSeqRat}^{\uparrow}$ and local $\mathsf{Real}^{\uparrow}$ terminal rows. These results keep the finite-window, common-tail, and terminal-seal data attached to the displayed object faces rather than to an external completion coordinate.

Finally return to the finite cover. Since one cover member lacks a required face row, the exactness criterion of Definition 38.812 fails before any global certificate can be glued. Quotient streams, selected limits, countable-choice selectors, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not rows of the finite Real L10 certificate surface, so they provide no replacement for the deleted face. □

Theorem 38.815 (Real L10 terminal four-face status locality)**.** *Let $\mathcal{T}$ be a finite Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *rows. Suppose $\mathcal{T}$ satisfies the current-phase exit local threshold and exposes exactly the four displayed face records together with the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then every formalstatus, leantarget, and bridgestatus coordinate carried by $\mathcal{T}$ is attached to one of the four local face records $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or local* $\mathsf{Real}^{\uparrow}$.

Conversely, if one such coordinate is replaced by a stronger coordinate on the same face, while the other three face records and the displayed support rows are unchanged, the resulting terminal packet presents the same four-face Phase $\mathsf{Real}^{\uparrow}$ *and Completion exit certificate. No route-level formalstatus row, aggregate leantarget row, aggregate bridge row, quotient-stream equality, selected limit, countable-choice selector, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness claim is introduced.*

Proof. First read the packet through the current exit criterion. By Definition 38.796, the terminal packet satisfies the exit read only as the objectwise product of the four local threshold records on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, carried by the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Next identify the finite cover seen by terminal requests. Apply Definition 38.797. The visible request section is finite and has no additional coordinate beyond the four displayed face records and the displayed support rows, so formalstatus, leantarget, and bridgestatus data can only be read from the face records already present in that cover.

Now factor the certificate along its ordered source. By Definition 38.798, the accepted exit certificate factors through exactly

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the same displayed support rows. The factorization gives no route object on which a separate formalstatus, leantarget, or bridge coordinate could live.

Finally apply Definition 38.793. Since none of the four faces and none of the displayed support rows can be deleted, shortened, or replaced by an aggregate credential while retaining the same exit read, a same-face strengthening of one status coordinate preserves the terminal four-face packet, whereas any route-level aggregate row or completion datum would be an extra coordinate outside the certificate surface. □

Theorem 38.816 (Real L10 finite-cover status threshold descent)**.** *Let $\{\mathcal{T}_i\}_{i\in I}$ be a finite cover of accepted Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal packets. Assume that every local packet $\mathcal{T}_i$ exposes exactly the four face records*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the shared support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

and that every nonempty overlap has compatible face-status pullbacks for the local $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *records.*

Then the glued terminal packet satisfies the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion exit threshold if and only if each local packet satisfies the four face-local threshold records for* $\mathsf{DyadicRatCore}^{\uparrow}$,

StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and local* Real$^{\uparrow}$, *and the overlap face-status pullbacks agree. The descent exports no aggregate formalstatus row, route-level Lean target, aggregate bridge row, quotient-stream equality, selected limit, countable-choice row, metric-completeness datum, or* Real$^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, restrict the glued terminal packet to one finite-cover member. Apply Definition 38.815. The restricted packet can read formalstatus, leantarget, and bridgestatus data only from the local DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ faces.

Now read the overlap interface. Apply Definition 291.112 on each nonempty overlap. The status rows remain attached to the same four faces after restriction, so the local four face-threshold records and their overlap pullbacks are forced by the glued threshold packet.

For the reverse direction, start with the local four face-threshold records and the compatible overlap pullbacks. Apply Definition 38.821 to paste the local terminal seal data across the finite cover. The pasted packet has the same StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and local Real$^{\uparrow}$ faces, together with the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Next apply Definition 301.227. The compatible local threshold reads glue to a single Phase Real$^{\uparrow}$ and Completion audit-threshold read over the finite four-face packet. Applying Definition 38.815 to that glued packet identifies it with the current exit threshold and keeps the verification and bridge coordinates facewise.

The exclusion clause follows from the same cited results. Their constructions never introduce an aggregate formalstatus row, a route-level Lean target, an aggregate bridge row, quotient streams, selected limits, countable-choice rows, metric-completeness data, or an ambient Real$^{\uparrow}$ completeness coordinate, so none of those data descends from the glued terminal packet. □

Theorem 38.817 (Real terminal seal sibling route)**.** *Every public* Real$^{\uparrow}$ *terminal seal read factors through the finite* StreamName$^{\uparrow}$ *window, the* RegSeqRat$^{\uparrow}$ *readback, and the* DyadicRatCore$^{\uparrow}$ *tolerance ledger before it reaches the local Real seal. The route is the cross-chapter certificate route through chapter 126, chapter 301, and chapter 291; its transport data are only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already exposed by those sibling chapters.*

Consequently an accepted terminal seal read has no route that bypasses the finite StreamName window, RegSeqRat readback, or DyadicRatCore tolerance ledger. In particular it exports no host real equality, quotient-stream equality, ambient completeness principle, countable-choice selector, selected limit, or independent host-completion row.

Proof. Start with the handoff theorem Definition 38.407. It reads the public Real$^{\uparrow}$ seal as a package whose visible rows are the StreamName classifier window, the RegSeqRat readback surface, the dyadic observation ledger, and the local Real seal provenance. The sibling chapter references in the statement are exactly the public sources of those rows.

Next apply the dyadic StreamName RegSeqRat route theorem Definition 38.670. It orders the exposed rows as one finite BEDC path from DyadicRatCore$^{\uparrow}$ observations through the StreamName$^{\uparrow}$ finite window and RegSeqRat$^{\uparrow}$ readback before the Real$^{\uparrow}$ seal is read. Since the route carries only BHist, hsame, Cont, Pkg, and NameCert coordinates, it has no coordinate for a host completion or quotient-stream equality.

Finally use the finite-window seal tightness theorem Definition 38.773. Tightness says that an accepted terminal seal is admissible exactly when the three sibling rows agree on one common finite request. Any proposed route omitting one of those rows therefore fails to be an accepted public Real$^{\uparrow}$ terminal seal, and any route adding host real equality, ambient completeness, countable choice, or a selected limit lies outside the displayed certificate surface. □

Theorem 38.818 (Real terminal seal four-face status pullback)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *terminal seal read whose route factors through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

reads its audit and status data exactly as the four objectwise closurestatus records of those faces. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *records keep their own closurestatus, formalstatus, leantarget, and bridgestatus rows; the terminal seal route creates no route-level* $\mathsf{formalstatus}$ *and no aggregate bridge row.*

Conversely, any terminal seal read carrying those four face records together with the displayed

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

rows factors through the same terminal seal sibling route and through the same four-face terminal pullback boundary. This equivalence does not assert quotient stream equality, selected limits, countable choice, metric-completeness data, or any $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First locate the route. Apply Definition 38.817. The public terminal seal read is confined to the StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and local Real seal row, with only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next read the status data objectwise. Apply Definition 38.470 and Definition 38.471. These identify the audit boundary with the product of the four local closurestatus records, while leaving each closurestatus, formalstatus, leantarget, and bridgestatus entry attached to its own face.

For the converse, start with a terminal seal read carrying exactly those four face records and the displayed support rows. Apply Definition 38.845. The objectwise status threshold and the shared StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates repack the read as the same terminal pullback boundary.

The exclusion clause follows from the same three cited results. None of their routes contains a coordinate for a route-level $\mathsf{formalstatus}$, aggregate bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness claim. □

Theorem 38.819 (Real terminal seal four-face cover normal form)**.** *Let* $\{\mathcal{R}_i\}_{i\in I}$ *be a finite family of public* $\mathsf{Real}^{\uparrow}$ *terminal seal reads. Suppose each local route exposes the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

where the $\mathsf{StreamName}^{\uparrow}$ *finite windows agree pairwise on their restrictions, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers agree on every overlap, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows agree on the same finite request sections, the local* $\mathsf{Real}^{\uparrow}$ *seal-source rows agree on the common boundary, and all routes use the shared support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the family descends to one finite-cover normal-form packet for the public $\mathsf{Real}^{\uparrow}$ *terminal seal. Conversely, any finite-cover normal-form packet with exactly those four faces and support rows restricts to the local terminal seal routes* $\mathcal{R}_i$*. The packet exports no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness datum, or standard bridge row.*

Proof. First read every local seal through the sibling route of Definition 38.817. Each $\mathcal{R}_i$ is thereby placed on the StreamName finite window, the RegSeqRat readback face, the DyadicRatCore tolerance ledger, and the local Real seal-source row, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Next build the finite-cover surface. Apply Definition 126.628 to the pairwise StreamName window agreements. The common terminal fused packet is then transported through the RegSeqRat finite-request section supplied by Definition 301.142.

Now transport the dyadic and Real boundary faces. Use Definition 291.69 to identify the endpoint, radius, distance, classifier, regularity, readback, and seal-source rows as one pullback surface over the same finite requests.

For the descent direction, apply Definition 38.774. The pasted StreamName surface, the shared dyadic ledger, the RegSeqRat classifier and regularity rows, and the local Real seal-source rows therefore form one finite-cover normal-form packet for the terminal seal.

For the restriction direction, start with such a normal-form packet and restrict it to any window of the finite cover. The same pasting criterion and the sibling route read the restricted packet back as the local terminal seal route $\mathcal{R}_i$, because no additional coordinate is present beyond the four displayed faces and the shared support rows.

The exclusion clause is inherited from the cited routes. None of them exports quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or a standard bridge row, so those data cannot be recovered from either direction of the normal form. □

Theorem 38.820 (Real terminal seal four-face cover descent determinacy)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be finite-cover normal-form packets for the public* $\mathsf{Real}^{\uparrow}$ *terminal seal. Suppose both packets expose exactly the four faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

agree on all overlap restrictions for those faces, and use only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then $\mathcal{P}$ and $\mathcal{Q}$ determine the same public $\mathsf{Real}^{\uparrow}$ *terminal seal read. Conversely, equality of the public terminal seal read recovers exactly the common* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request classifier and regularity rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal-source row. It recovers no quotient-stream coordinate, host-real coordinate, selected limit, countable-choice selector, ambient-completeness principle, metric-completeness datum, or standard-bridge coordinate.*

Proof. Read both packets through the sibling route of Definition 38.817. This places $\mathcal{P}$ and $\mathcal{Q}$ on the same four public faces and confirms that their support surface is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert surface.

Apply the normal-form theorem Definition 38.819 to both packets. The hypotheses on overlap restrictions put the two packets on one finite-cover surface, because the StreamName windows, dyadic tolerance rows, RegSeqRat readback rows, and Real seal-source rows are identified on every common section.

The common section is determined by the finite cover itself. Use Definition 126.628 for the fused StreamName window, and use Definition 301.142 for the RegSeqRat finite-request budget over that window.

It remains to transport the dyadic and Real rows across the common section. Apply Definition 291.69 to the endpoint, radius, distance, classifier, regularity, readback, and seal-source rows. They are therefore the same pullback data for $\mathcal{P}$ and $\mathcal{Q}$.

Finally apply Definition 38.774. The pasting criterion says that a four-face packet with those overlap restrictions has a unique public terminal seal read, so $\mathcal{P}$ and $\mathcal{Q}$ determine the same read.

For the converse, start from equality of the public terminal seal read and read it back through the same sibling route and normal-form theorem. The restriction data are exactly the four recovered faces listed in the statement. The exclusion clause follows because every cited route uses only the displayed support rows and has no coordinate for quotient streams, host reals, selected limits, countable choice, ambient completeness, metric completeness, or a standard bridge. □

Theorem 38.821 (Real terminal seal finite-cover cocycle exactness)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of local terminal-seal reads for the public* $\mathsf{Real}^{\uparrow}$ *surface. Suppose every $\mathcal{R}_i$ exposes exactly the four faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

with support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},$$

and suppose that for every overlap i, j the restrictions of the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request classifier and regularity rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal-source row agree. Assume also that on every triple overlap i, j, k the two restriction routes from $\mathcal{R}_i$ to the common section coincide:*

$$(\mathcal{R}_i|_{ij})|_{ijk} = (\mathcal{R}_i|_{ik})|_{ijk}, \qquad (\mathcal{R}_i|_{ij})|_{ijk} = (\mathcal{R}_j|_{ij})|_{ijk}.$$

Then the family pastes to a unique public $\mathsf{Real}^{\uparrow}$ *terminal-seal read whose restriction to each finite cover member is $\mathcal{R}_i$. Conversely, any public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read restricts to such a finite cocycle on the four displayed faces. This exactness exports no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness datum, or standard bridge row.*

Proof. First put every local read in four-face normal form. Apply Definition 38.819; each $\mathcal{R}_i$ is read as a finite StreamName window, a DyadicRatCore endpoint-radius-distance ledger, a RegSeqRat finite-request classifier with regularity rows, and a local Real seal-source row over the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next paste the StreamName face. The overlap and triple-overlap assumptions are exactly the cocycle data used by Definition 126.628, so the finite windows fuse to one terminal StreamName packet and restrict back to the given windows.

Transport the remaining faces over that fused window. Use Definition 291.69 to carry the dyadic endpoint, radius, and distance rows together with the RegSeqRat classifier, regularity, readback, and Real seal-source rows as one pullback surface over the common finite requests.

Now apply Definition 38.774. The pasted StreamName packet and transported pullback rows give a public terminal-seal read, and its restriction to every cover member is the original $\mathcal{R}_i$.

Uniqueness is the determinacy content of Definition 38.820. Any two public reads with those restrictions have the same four faces on every overlap, hence determine the same terminal-seal read.

For the converse, start with a public $\mathsf{Real}^{\uparrow}$ terminal-seal read and restrict it along the finite cover. The normal-form theorem supplies the four local faces, the StreamName fusion theorem

gives the cocycle equations on triple overlaps, and the pasting criterion identifies the recovered public read with the original one.

The exclusion clause is stable under each cited step: none of the normal-form, fusion, pullback-composition, pasting, or determinacy results introduces quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or a standard bridge row. □

Theorem 38.822 (Real terminal seal RegSeqRat cover descent equivalence)**.** *Let $\{\mathcal{Q}_i\}_{i\in I}$ be a finite family of public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads over one finite cover. The family presents an accepted public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read if and only if its cover data satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *four-face gluing rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *four-face associativity rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-cover descent exactness row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger agreement, and the local* $\mathsf{Real}^{\uparrow}$ *seal cocycle exactness row. The accepted source has exactly the four public faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and exactly the common support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It exports no quotient streams, host equality, selected limits, countable choice, ambient completeness, metric completeness, aggregate formalstatus row, or standard bridge row.

Proof. For the forward direction, start with the accepted public $\mathsf{Real}^{\uparrow}$ terminal-seal read presented by the finite $\mathsf{RegSeqRat}^{\uparrow}$ family. Apply Definition 38.821. The public read restricts to local terminal-seal reads whose only visible faces are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ over the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next identify the descent rows carried by those restrictions. Apply Definition 301.88 and Definition 301.90 to read the $\mathsf{RegSeqRat}^{\uparrow}$ classifier, regularity, and readback data as a coherent four-face cover. Apply Definition 126.93 to recover the $\mathsf{StreamName}^{\uparrow}$ finite-cover descent row, and read the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance coordinates from the same pullback ledger. Thus the forward image is exactly the descent packet named in the statement.

For the reverse direction, start with a finite $\mathsf{RegSeqRat}^{\uparrow}$ cover packet satisfying the gluing, associativity, StreamName descent, DyadicRatCore ledger, and local Real cocycle rows. Apply Definition 301.91. The packet admits a common refinement whose face restrictions are comparable over one support spine.

Now apply Definition 38.820. The refined packet determines at most one public terminal-seal read on the four faces. Apply Definition 38.821 to paste the local Real cocycle and obtain that public read. The pasted read restricts back to the original finite $\mathsf{RegSeqRat}^{\uparrow}$ descent data, so the two constructions are inverse at the accepted public-source surface.

The negative boundary is audit-only. Apply Definition 38.824; the finite-cover route does not create a route-level formalstatus aggregate, replacement leantarget, or bridge row. Since all preceding steps use only the four faces and support rows displayed above, no quotient stream, host equality, selected limit, countable-choice principle, ambient-completeness row, or metric-completeness row is introduced. □

Theorem 38.823 (Real terminal seal StreamName–RegSeqRat handoff exactness)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows over one displayed finite cover. Suppose its pasted seal exhaustion is certified by Definition 126.552, its* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff satisfies Definition 301.88, and its* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger agrees on every displayed overlap. Then*

these data determine exactly one public $\mathsf{Real}^{\uparrow}$ *terminal-seal read. Conversely, every accepted public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read restricts to precisely such a StreamName, RegSeqRat, and DyadicRatCore handoff packet. The criterion exports no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness datum, aggregate formalstatus row, or standard bridge row.*

Proof. First build the pasted finite-window surface. Apply Definition 126.552 to the StreamName windows and their certified overlaps. The result is one finite-window seal surface whose restrictions are the original windows and whose support rows remain the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next transport the classifier and regularity data. Use Definition 301.88 to glue the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request classifiers, regularity rows, and readback rows over the pasted StreamName surface. Use Definition 291.69 to carry the endpoint, radius, and distance ledger through the same support spine.

Now invoke the local Real terminal-seal criterion. The pasted StreamName surface, the glued RegSeqRat handoff, and the transported DyadicRatCore ledger satisfy the hypotheses of Definition 38.821. That theorem produces a public $\mathsf{Real}^{\uparrow}$ terminal-seal read, and Definition 38.820 makes it the unique read with those finite restrictions.

For the converse, restrict an accepted public $\mathsf{Real}^{\uparrow}$ terminal-seal read to the finite cover. The restriction exposes StreamName windows, RegSeqRat handoff rows, and DyadicRatCore endpoint-radius-distance rows on the same support. Applying the same three cited handoff theorems to these restrictions recovers the packet named in the statement, while determinacy identifies the reconstructed public read with the original one.

The exclusion clause is inherited from the cited four-face route. None of the StreamName exhaustion, RegSeqRat gluing, DyadicRatCore pullback composition, Real cocycle exactness, or Real descent determinacy steps introduces quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus rows, or standard bridge rows. □

Theorem 38.824 (Real terminal seal audit noncollapse)**.** *Let* $\{\mathcal{R}_i\}_{i\in I}$ *be a finite family of local terminal-seal reads satisfying the hypotheses of Definition 38.821, and let* $\mathcal{R}$ *be the resulting public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read. Then the audit rows visible from* $\mathcal{R}$ *over the four displayed faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

are exactly the objectwise closurestatus, formalstatus, leantarget, and bridgestatus rows of those four faces. The finite-cover descent supplies no aggregate route-level status row, no replacement leantarget, and no standard-bridge row.

Proof. Apply Definition 38.821. The public read $\mathcal{R}$ is obtained only by pasting the StreamName, DyadicRatCore, RegSeqRat, and local Real terminal-seal faces over their common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next read the audit coordinate through Definition 38.471. That pullback identifies the current Phase Real and Completion audit report with the objectwise product of the four face reports, so the pasted terminal-seal route has no separate audit carrier beyond the displayed face rows.

Apply Definition 38.833. The closurestatus, formalstatus, and bridgestatus rows therefore remain attached to their respective faces and are not replaced by a route-level status row. Apply Definition 38.834; the same locality holds for the leantarget rows.

Finally reuse the exclusion clause of Definition 38.821. Since the descent has no coordinate for quotient streams, host reals, selected limits, countable choice, ambient completeness, metric

completeness, or a standard bridge, the audit read cannot manufacture an aggregate bridge or verification record. □

Definition 38.825 (Real terminal seal dependency manifest)**.** *The* $\mathsf{Real}^{\uparrow}$ *terminal-seal dependency manifest is the finite packet*

$$\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$$

attached to a public terminal-seal read, with the following rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window of stream names and finite request coordinates.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger over the window S.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and readback row over the same finite requests.*
4. *E is the local* $\mathsf{Real}^{\uparrow}$ *seal-source row obtained from the StreamName–DyadicRatCore–RegSeqRat pullback.*
5. *B, H, C, P, N are respectively the shared* BHist *ledger,* hsame *transport,* Cont *route,* Pkg *provenance, and* $\mathsf{NameCert}$ *row used by the four faces.*

A public terminal-seal read consumes a dependency iff it is one of these nine rows or a restriction of one of them along a finite cover member.

Theorem 38.826 (Real terminal seal dependency manifest exhaustion)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read factors through exactly the manifest* $\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}}$*. Equivalently, after restricting the read to any finite cover, all data consumed by the read are exhausted by the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier/readback row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support. These rows paste back to the public read and no further dependency coordinate is available.*

Proof. Put the public read in four-face normal form by applying Definition 38.819. This identifies the read with the four displayed faces over the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, which is precisely the manifest packet of Definition 38.825.

Now restrict the read along a finite cover. Apply Definition 38.821: the local restrictions form a cocycle on those same rows, and the cocycle pastes back to the original public terminal-seal read.

It remains to rule out an additional dependency coordinate. Apply Definition 38.824. The only audit rows visible from the pasted read are the objectwise rows of the four faces, and there is no route-level replacement row. Hence every public dependency is one of the manifest rows or its finite-cover restriction, and the manifest exhausts the read. □

Theorem 38.827 (Real terminal seal dependency manifest non-escape)**.** *The manifest* $\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}}$ *contains no row for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or standard-bridge data. Consequently no public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read may consume any of those objects through the manifest.*

Proof. The manifest rows are listed in Definition 38.825; each row is one of $S, D, R, E, B, H, C, P, N$ and therefore belongs to the four-face terminal-seal route or to its shared BEDC support.

Apply Definition 38.826. Any public terminal-seal read consumes only those rows or their finite-cover restrictions.

Finally apply the exclusion clauses of Definition 38.819, Definition 38.821, and Definition 38.824. None of the four-face normal form, finite-cover cocycle, or audit projection contains quotient

stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or standard-bridge data. Since the manifest has no coordinate for such data, no public read can escape through it. □

Theorem 38.828 (Real terminal seal public source completeness)**.** *A finite packet*

$$M = (S, D, R, E, B, H, C, P, N)$$

reconstructs a public $\mathsf{Real}^{\uparrow}$ *terminal-seal read exactly when its rows are the terminal-seal dependency manifest rows of Definition 38.825:* S *is the finite* $\mathsf{StreamName}^{\uparrow}$ *name-and-request window,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger on that window,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row,* E *is the local* $\mathsf{Real}^{\uparrow}$ *seal row, and* B, H, C, P, N *are the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. The rows must satisfy the shared-face compatibility conditions displayed by Definition 38.819 and the finite-cover cocycle compatibility of Definition 38.821. Under those conditions* M *determines a public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read, and every public* $\mathsf{Real}^{\uparrow}$ *terminal-seal read yields exactly such a compatible packet.*

Proof. Start from a compatible packet $M = (S, D, R, E, B, H, C, P, N)$. The row interpretation is fixed by Definition 38.825: the first four rows are the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces, and the last five rows are the common BEDC support.

Apply Definition 38.819. The shared-face compatibility hypotheses identify these rows as the four terminal faces over one BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ spine. Apply Definition 38.821. The finite-cover compatibility pastes the local face restrictions into a public terminal-seal read.

For the reverse direction, begin with a public $\mathsf{Real}^{\uparrow}$ terminal-seal read. Apply Definition 38.826. The read factors through the manifest rows and no additional dependency coordinate is available. The normal-form and finite-cover theorems provide the required shared-face and cocycle compatibility, so the read yields a packet of the displayed form.

Finally apply Definition 38.827. The packet has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or standard-bridge data. Hence the reconstruction and extraction have the same public source surface, namely the manifest M. □

Theorem 38.829 (Real terminal seal dependency manifest phase-exit face product)**.** *Let*

$$\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{Real}^{\uparrow}$ *terminal-seal dependency manifest. Then this manifest determines exactly the Phase Real and Completion four-face product over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *face, with the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. Conversely, every accepted Phase Real and Completion four-face product over those four faces restricts to this manifest. The correspondence exports no quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, aggregate formalstatus tokens, or standard bridge rows.*

Proof. For the forward direction, read the manifest through Definition 38.825. Its rows are precisely the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier/readback row, the local $\mathsf{Real}^{\uparrow}$ seal-source row, and the common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Apply Definition 38.826. Every public terminal-seal read consumes only those rows or their finite-cover restrictions. Apply Definition 38.471 to identify the Phase Real and Completion audit

read with the product of the same four face reports, and apply Definition 38.775 to read that product as the public four-face interface.

For the reverse direction, start with an accepted Phase Real and Completion four-face product over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face. The public-interface theorem displays exactly the same four face rows and the same shared BEDC support rows, so restricting the product to the terminal-seal consumer surface gives the packet $\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}}$.

It remains to check that no coordinate has escaped the manifest. Apply Definition 38.827. The manifest has no row for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, or standard-bridge data. The audit pullback is facewise, not aggregate, so it also supplies no aggregate formalstatus token and no standard bridge row beyond the displayed face records. □

Theorem 38.830 (Real terminal seal manifest exit consumer exactness)**.** *An accepted manifest*

$$\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$$

is accepted by the current Phase Real and Completion exit consumer exactly when its consumer rows are the four object faces $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$*, together with the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows. Equivalently, the manifest is precisely the consumer surface used by the current L10 exit route. No accepted consumer presentation adds a quotient stream, host real equality, selected limit, countable choice, ambient completeness, metric-completeness row, aggregate formalstatus token, or standard bridge row.*

Proof. For the forward direction, apply Definition 38.828. The accepted manifest already reconstructs a public terminal-seal read from exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Apply Definition 38.829. It identifies those rows with the Phase Real and Completion four-face product. Then apply Definition 126.500, Definition 291.28, and Definition 301.22. These three handoff surfaces read the same finite consumer as the current L10 exit route through the StreamName, DyadicRatCore, RegSeqRat, and Real faces.

For the reverse direction, start with a current L10 exit consumer presentation. The four-object exit route supplies exactly the same four object faces and the same shared BEDC support rows. The DyadicRatCore-to-RegSeqRat handoff and the RegSeqRat-to-Real seal handoff identify the downstream reads with the displayed dyadic ledger, regular-rational classifier, and local Real seal rows, so the presentation restricts to $\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$.

Finally, the exclusion clause follows from Definition 38.827 and from the facewise character of Definition 38.829. The consumer has no coordinate in which quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness data, aggregate formalstatus tokens, or standard bridge rows could be carried. □

Theorem 38.831 (Real terminal seal four-face classifier correspondence)**.** *Let*

$$\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{Real}^{\uparrow}$ *terminal-seal dependency manifest, and let* Q *be an accepted Phase Real and Completion four-face product over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *seal face. The manifest and* Q *determine the same public classifier verdict exactly when their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier/readback row, local* $\mathsf{Real}^{\uparrow}$ *seal row, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows agree. Conversely,*

any shared public classifier verdict for the terminal manifest and the four-face product restricts to those same four faces and those same shared support rows. The correspondence carries no quotient stream, host real equality, selected limit, countable-choice coordinate, ambient completeness, metric-completeness row, aggregate formalstatus token, or standard bridge row.

Proof. Begin with an accepted terminal manifest and an accepted four-face product. Apply Definition 38.829. It identifies both presentations with the same displayed product of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces over the common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Apply Definition 38.830. The manifest is accepted by the current exit consumer precisely through these consumer rows. Then apply Definition 126.500, Definition 291.28, and Definition 301.22. These handoff theorems align the finite StreamName window, the dyadic tolerance ledger, the regular-rational classifier/readback row, and the local Real seal row with the public classifier read used by the L10 exit route.

Thus equal rows give the same public classifier verdict. No independent classifier coordinate remains after the handoffs, because the verdict is read from exactly the four object faces and the shared BEDC support.

For the reverse direction, suppose the terminal manifest and the four-face product share a public classifier verdict. The four-object exit route first restricts that verdict to the $\mathsf{StreamName}^{\uparrow}$ window and to the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. The DyadicRatCore-to-RegSeqRat handoff then restricts it to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier/readback row, and the RegSeqRat-to-Real handoff restricts it to the local $\mathsf{Real}^{\uparrow}$ seal row.

Finally, consumer exactness and the manifest-to-product identification force the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows on both presentations. The non-escape clause in the manifest identification excludes quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, aggregate formalstatus tokens, and standard bridge rows, so the shared classifier verdict has no off-table restriction beyond the four displayed faces. □

Theorem 38.832 (Real terminal seal NameCert exit consumer exactness)**.** *An accepted terminal-seal manifest*

$$\mathsf{Dep}^{\mathrm{term}}_{\mathsf{Real}^{\uparrow}} = (S, D, R, E, B, H, C, P, N)$$

determines the same local $\mathsf{NameCert}$ *row as the current Phase Real and Completion exit consumer exactly when the four object faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$*, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, agree between the manifest presentation and the L10 exit presentation. Consequently, the local* $\mathsf{NameCert}$ *consumer row is neither an additional aggregate certificate nor an off-table completeness assertion: it is the facewise* $\mathsf{NameCert}$ *support already carried by the accepted terminal-seal manifest and by the current Phase Real and Completion exit surface.*

Proof. First compare the manifest with the current exit consumer. Apply Definition 38.830. It identifies the accepted terminal-seal manifest with exactly the consumer rows read by the current L10 exit route: the four object faces and the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next compare that consumer presentation with the classifier read. Apply Definition 38.831. The public classifier verdict has no independent coordinate beyond agreement of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces over the same shared support rows, so equality of the displayed $\mathsf{NameCert}$ rows is exactly the NameCert component of the classifier correspondence.

It remains to identify the same row with the Phase Real and Completion exit route. Apply Definition 126.500 to read the consumer through the four-object surface, then apply Definition 291.28 and Definition 301.22. These handoffs preserve the dyadic ledger, stream window, regular-rational classifier/readback row, and local Real seal row, and they do not create a second NameCert coordinate.

Conversely, if the current Phase Real and Completion exit consumer has the same four object faces and the same BHist, hsame, Cont, Pkg, and NameCert rows, the preceding identifications restrict it to the accepted terminal-seal manifest. The non-escape clauses in Definition 38.830 and Definition 38.831 exclude quotient streams, host real equality, selected limits, choice, completeness, aggregate formalstatus rows, and standard bridge rows, so the exact local NameCert row is precisely the one displayed in the manifest. □

Theorem 38.833 (Real L10 face status non-collapse)**.** *Any current Phase Real and Completion audit read through the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *preserves the four local closure records as separate face reports. No conjunction of those records can be read as a new formalstatus token, a* bridgeChecked *aggregate, a* $\mathsf{Real}^{\uparrow}$ *completeness claim, or a standard bridge for the composite exit route.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceStatusNoncollapse`

Proof. Apply Definition 38.471. It identifies the current exit audit as the product of the four local face records and states that the exit packet contributes no fifth formal token or aggregate bridge.

The two-axis convention of section 9467.2 reads closure and verification status locally. A face report can therefore be cited as a face report, but the conjunction of face reports is not itself a further formalstatus entry.

Finally use the exactness and non-escape inputs named in the pullback proof: Definition 38.443, Definition 38.444, Definition 38.457, and Definition 38.460. They keep the composite exit route on the four displayed faces and provide no row for RealUp completeness or for a standard bridge over the product. □

Theorem 38.834 (Real L10 face Lean target non-collapse)**.** *Any accepted current Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *preserves each face's Lean-target row and bridge-status row as local records of that face. No* bridgeChecked *face can supply the Lean target or bridge row for another face, and the composite route creates neither a route-level Lean target nor an aggregate bridge row.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceLeantargetNoncollapse`

Proof. Apply Definition 38.471. It reads the accepted exit audit as the product of the four local face records over the displayed Real L10 dependency lattice.

Apply Definition 38.833. It keeps the closure and verification reports objectwise, so a bridgeChecked entry is a record of its own face rather than a credential for a different face.

Now project the target and bridge rows along the same exactness data used by the pullback. The source-route non-escape, interface-totality, four-face exactness, and StreamName exit-correspondence inputs provide only face-local Lean-target and bridge-status coordinates for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not an aggregate target row or an aggregate bridge row.

Since those inputs provide no fifth face, no route-level Lean-target coordinate, and no aggregate bridge coordinate, the local Lean-target and bridge-status rows remain attached to the faces from which they are read. □

Theorem 38.835 (Real L10 face status-target pairing)**.** *In any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, each face's* formalstatus*,* leantarget*, and* bridgestatus *rows are read as one local face record. A status row from one face cannot be paired with the Lean target or bridge row of another face, and no aggregate route row can repair such a mismatch.*

Proof. Apply Definition 38.471. The accepted audit is read as the product of the four local records carried by $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not as a single route credential.

Apply Definition 38.833. It keeps the formalstatus coordinate attached to the face whose closure record supplies it, so the status row cannot be detached and reused as a route-level token.

Apply Definition 38.834. It keeps the leantarget and bridgestatus coordinates attached to the same local face records and supplies no route-level target or aggregate bridge row.

Finally read the three rows through section 9467.2. The roadmap discipline keeps verification status, Lean target, and bridge status as reported coordinates of the displayed face, so a mixed record using one face's status and another face's target or bridge row is outside the accepted audit read. □

Theorem 38.836 (Real L10 face status product readback)**.** *Any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *reads exactly the four face-local triples* (formalstatus, leantarget, bridgestatus) *carried by the corresponding face records. The readback is supported only by the displayed rows*

$$\mathsf{BHist}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{hsame}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Cont}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{DyadicRatCore}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{hsame}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Cont}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{StreamName}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{hsame}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Cont}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}},$$
$$\mathsf{BHist}_{\mathsf{Real}^{\uparrow}}, \mathsf{hsame}_{\mathsf{Real}^{\uparrow}}, \mathsf{Cont}_{\mathsf{Real}^{\uparrow}}, \mathsf{Pkg}_{\mathsf{Real}^{\uparrow}}, \mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}.$$

It does not create an aggregate formalstatus*, aggregate* leantarget*, aggregate bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RealUp.RealL10FaceStatusProductReadback`

Proof. Apply Definition 38.833. It keeps the status coordinates as local reports of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and supplies no fifth status coordinate.

Apply Definition 38.834. It keeps the leantarget and bridgestatus rows attached to their source faces and supplies no route-level target or aggregate bridge row.

Apply Definition 38.835. It reads formalstatus, leantarget, and bridgestatus together on the same face, so the product readback is exactly the four displayed local triples rather than a pool of exchangeable coordinates.

Finally use section 9467.2. The roadmap keeps paper closure and verification status on separate axes, and it supplies no rule turning the four face-local triples or their support rows BHist, hsame, Cont, Pkg, NameCert into quotient stream equality, a selected limit, a countable-choice row, a metric-completeness datum, an aggregate credential, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 38.837 (Real L10 face axis independence)**.** *Any accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *preserves four independent local coordinates on each face: paper closure, formalstatus, leantarget, and bridgestatus. No*

coordinate from one face promotes any coordinate of another face, and no conjunction of face coordinates creates route-level closure, route-level formalstatus, route-level leantarget, aggregate bridge status, quotient stream, selected limit, countable-choice row, metric-completeness row, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.471. The audit is read as a product of the displayed faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, over the Real L10 dependency lattice, with no additional route coordinate.

Apply Definition 38.833. It keeps the paper closure and formalstatus reports face-local and blocks any reading in which a conjunction of local status records becomes route-level closure, route-level formalstatus, an aggregate bridge, or a $\mathsf{Real}^{\uparrow}$ completeness claim.

Apply Definition 38.834. It keeps leantarget and bridgestatus records attached to their source faces and blocks any reading in which one face supplies the Lean target or bridge row for another face.

Finally use the two-axis discipline of section 9467.2. Paper closure, formalstatus, leantarget, and bridgestatus are reported as separate coordinates; the roadmap supplies no rule that turns their conjunction into a quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate bridge, or completed real-completeness certificate. □

Theorem 38.838 (Real L10 face-axis defect localization)**.** *Suppose an accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *does not preserve one of the four independent coordinates named in Definition 38.837. Then the defect is localized to at least one displayed face record or to at least one displayed support row among* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. *No quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate formalstatus, aggregate bridge row, route-level leantarget, or* $\mathsf{Real}^{\uparrow}$ *completeness claim can serve as the defect carrier.*

Proof. Apply Definition 38.837. The admissible coordinate inventory is the displayed facewise list: paper closure, formalstatus, leantarget, and bridgestatus over the four Real L10 faces. The same theorem excludes route-level completion vocabulary and aggregate status vocabulary from that inventory.

Apply Definition 38.833. It keeps paper closure and formalstatus attached to the local closure records of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Thus a defect in either status coordinate is witnessed on at least one displayed face record or on at least one of its displayed support rows $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$.

Apply Definition 38.834. It keeps leantarget and bridgestatus attached to their source faces and supplies no route-level target or aggregate bridge row. Thus a defect in either target coordinate is again witnessed on at least one displayed face record or on at least one displayed support row.

Finally use the two-axis discipline of section 9467.2. Paper closure, formalstatus, leantarget, and bridgestatus are reported as separate axes, and there is no rule that lets a quotient stream, selected limit, countable-choice row, metric-completeness row, aggregate formalstatus, aggregate bridge row, route-level leantarget, or $\mathsf{Real}^{\uparrow}$ completeness claim replace the concrete face or support coordinate where the read failed. □

Theorem 38.839 (Real L10 exit defect face completeness)**.** *Every failed accepted Phase Real and Completion audit read through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is exhausted by the four displayed face records and their displayed support rows* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. *Equivalently, any accepted defect witness is carried either by one of the four face records or by one of those support rows on that face. No quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus, route-level leantarget, aggregate bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim is an additional defect coordinate.*

Proof. Apply Definition 38.837. It supplies the complete coordinate inventory for an accepted audit read: paper closure, formalstatus, leantarget, and bridgestatus, each read locally over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.838. Once one of those coordinates fails to be preserved, the failure is witnessed on a displayed face record or on one of its displayed support rows BHist, hsame, Cont, Pkg, NameCert.

Apply Definition 38.836. The product readback contains exactly the four face-local status, target, and bridge triples supported by the displayed rows, so there is no remaining readback slot in which a hidden fifth status or completion coordinate could be stored.

Finally use the roadmap two-axis discipline of section 9467.2. It separates theory closure from verification status and provides no rule turning quotient-stream data, selected-limit data, countable-choice data, metric-completeness data, aggregate status data, route-level target data, aggregate bridge data, or $\mathsf{Real}^{\uparrow}$ completeness vocabulary into an accepted defect coordinate. □

Theorem 38.840 (Real L10 StreamName terminal status minimality)**.** *For any accepted Phase Real and Completion terminal read through* $\mathsf{StreamName}^{\uparrow}$*, the terminal read satisfies the roadmap exit status coordinate only if the four concrete face chapters* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each supply their own local closurestatus, formalstatus, leantarget, and bridgestatus record at the required local audit-clean grade. If any one face record is absent or below that grade, then no aggregate formalstatus, aggregate bridge token, quotient stream, selected limit, countable-choice row, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim repairs the terminal read.*

Proof. Apply Definition 126.425. The terminal $\mathsf{StreamName}^{\uparrow}$ read is forced to inspect the four concrete sibling status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not a read from a route-level status cache.

Now apply Definition 38.489. A missing face record, or a face record below the required audit-clean grade, blocks the current Phase Real and Completion exit criterion at the object where the defect is read.

Finally apply Definition 38.833 and Definition 38.834. These results keep the closurestatus, formalstatus, leantarget, and bridgestatus coordinates attached to their source faces and provide no aggregate replacement row. Therefore quotient-stream data, selected-limit data, countable-choice data, metric-completeness data, aggregate formalstatus, aggregate bridge tokens, and $\mathsf{Real}^{\uparrow}$ completeness vocabulary cannot repair the failed terminal read. □

Theorem 38.841 (Real L10 terminal four-face defect localization)**.** *Let* $\mathcal{T}$ *be an accepted terminal Phase Real and Completion read through* $\mathsf{StreamName}^{\uparrow}$*. If* $\mathcal{T}$ *fails the roadmap exit criterion, then the failed terminal read has a witness localized to a single displayed face among* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. At that face, the witness is read either from the local closurestatus, formalstatus, leantarget, and bridgestatus record, or from one of the displayed support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *attached to that face.*

The localization has no repair coordinate outside those four faces. Quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge tokens, and $\mathsf{Real}^{\uparrow}$ *completeness do not supply terminal repair data and do not move the witness away from the face where the failed read is displayed.*

Proof. Start with Definition 38.840. It forces every accepted terminal $\mathsf{StreamName}^{\uparrow}$ read to inspect the four local closurestatus, formalstatus, leantarget, and bridgestatus records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 38.839. A failed accepted exit read is exhausted by those four face records and by the displayed BHist, hsame, Cont, Pkg, and NameCert support rows on those faces. Hence some face carries the displayed witness.

Now use Definition 38.836. The terminal readback contains exactly the four face-local status, target, and bridge triples, so the witness can be chosen on one of the four objectwise faces rather than on a route-level aggregate.

Finally read the conclusion through section 9467.2. The roadmap separates paper closure, verification status, Lean target, and bridge status into local reported coordinates. It gives no rule that turns quotient stream equality, selected limits, countable choice, metric completeness, aggregate formalstatus, aggregate bridge data, or $\mathsf{Real}^{\uparrow}$ completeness into a terminal repair coordinate. □

Theorem 38.842 (Real L10 regular-Cauchy seal route completeness)**.** *Let $\mathcal{T}$ be an accepted Phase Real and Completion terminal read whose* $\mathsf{Real}^{\uparrow}$ *seal is supplied by an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *route. Then the accepted seal route of $\mathcal{T}$ is exhausted by exactly the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the visible support rows BHist, hsame, Cont, Pkg, *and* NameCert *on those faces.*

No fifth route coordinate is exported. In particular, the accepted seal route does not read quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completion data, or standard bridge rows.*

Proof. Apply Definition 301.40. An accepted $\mathsf{RegSeqRat}^{\uparrow}$ route supplying the $\mathsf{Real}^{\uparrow}$ seal exports only the finite regular-source rows and excludes quotient stream equality, selected host limits, host real equality, hidden tail witnesses, and ambient completeness data.

Next use Definition 301.111. The $\mathsf{Real}^{\uparrow}$ consumer read factors through the same finite route rows, so the seal route seen by the terminal Phase Real and Completion read cannot acquire an additional consumer-side completion coordinate.

Use Definition 291.123 for the dyadic tolerance face. The tolerance component of the route is already read through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 38.833, Definition 38.834, and Definition 38.841. These results keep the L10 status, target, bridge, and defect witnesses localized to the four displayed faces and their support rows. Hence the accepted regular-Cauchy seal route has no fifth status coordinate, no independent completeness coordinate, and no standard bridge row outside the displayed four-face route. □

Theorem 38.843 (Real L10 finite-cover exit threshold gluing)**.** *Let $\mathcal{U}$ be a finite family of accepted Phase Real and Completion exit packets. Suppose every member of $\mathcal{U}$ satisfies the objectwise exit threshold of Definition 38.494, and suppose that every pairwise overlap agrees on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then the family glues to one accepted four-face Phase Real and Completion exit packet satisfying the same objectwise exit threshold.*

The glued exit packet is only the finite four-face packet over those displayed rows. It exports no aggregate route-level formalstatus, *no aggregate bridge row, no quotient stream, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.494 to each member of $\mathcal{U}$. Thus every local packet contributes exactly the four face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the threshold read from the local closurestatus rows of those faces.

We first glue the regular-sequence and real-seal faces. Apply Definition 301.88 to the finite family of $\mathsf{RegSeqRat}^{\uparrow}$ regularity rows and $\mathsf{Real}^{\uparrow}$ seal rows. The overlap agreement hypothesis is

precisely the shared regularity, classifier, seal, BHist, hsame, Cont, Pkg, and NameCert surface required for that gluing. Apply Definition 301.80 to identify the resulting seal row on common overlaps.

Next glue the scheduling face. Use Definition 126.240 and Definition 126.265 to pass to a common finite-window presentation of the $\mathsf{StreamName}^{\uparrow}$ schedules. Apply Definition 126.562 to export one schedule row for the finite family, still over the displayed finite-window data and not over a quotient stream or selected limit.

The dyadic face is fixed by the overlap ledger. Apply Definition 291.66 and Definition 291.70. These read the shared endpoint and radius data through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, so the glued packet has the same dyadic endpoint and radius rows on all refined windows.

Combining the glued RegSeqRat/Real, StreamName, and DyadicRatCore rows gives one accepted four-face exit packet by Definition 38.493. Finally apply Definition 38.833 and Definition 38.834. The glued packet remains objectwise: its closure, verification, Lean-target, and bridge rows stay on their source faces and do not form an aggregate route-level status, bridge, quotient, selected-limit, choice, metric-completeness, or $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 38.844 (Real L10 finite-cover exit threshold descent)**.** *Let P be an accepted Phase Real and Completion exit packet, and let $\mathcal{V}$ be a finite certified subcover of P. Suppose every member of $\mathcal{V}$ satisfies the objectwise exit threshold of Definition 38.494. Suppose moreover that each subcover member agrees with P on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then P satisfies the same objectwise exit threshold.*

The descended threshold is only the coarse four-face threshold of P. It does not assert an aggregate route-level formalstatus*, an aggregate bridge row, a quotient stream, a selected limit, a countable-choice row, a metric-completeness datum, or a* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 38.843 to the finite certified subcover $\mathcal{V}$. The hypotheses on the local threshold and on the shared $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows give one glued accepted four-face packet satisfying the objectwise exit threshold.

We now identify this glued packet with the coarse packet P. The agreement assumptions say that every displayed endpoint, radius, finite-window, regularity, classifier, seal, history, sameness, continuity, package, and NameCert coordinate of the glued packet is the corresponding coordinate of P. Apply Definition 38.494 to read the glued threshold as the four local face records of P.

It remains to keep the descent facewise. Apply Definition 38.833 and Definition 38.834. These prevent the subcover gluing from creating an aggregate status, bridge, quotient-stream, selected-limit, countable-choice, metric-completeness, or $\mathsf{Real}^{\uparrow}$ completeness reading. Hence the coarse packet P carries exactly the same objectwise exit threshold and no route-level strengthening. □

Theorem 38.845 (Real L10 terminal pullback coherence)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read presented through the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route. Suppose its StreamName window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise seal and readback,* $\mathsf{Real}^{\uparrow}$ *seal row, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support agree with a four-face Phase Real and Completion packet P. Then T satisfies the same objectwise exit threshold as P in the sense of Definition 38.494.*

Conversely, any terminal finite-window $\mathsf{Real}^{\uparrow}$ *read satisfying that objectwise threshold and carrying the displayed StreamName, DyadicRatCore, RegSeqRat, Real seal,* BHist*,* hsame*,* Cont*,* Pkg*,*

and NameCert *rows factors through the StreamName–*RegSeqRat–Real$^{\uparrow}$ *terminal pullback over the four-face Phase Real and Completion packet.*

The coherent terminal read is only this displayed finite-window pullback. It does not assert an aggregate route-level formalstatus*, an aggregate bridge row, a quotient stream, a selected limit, a countable-choice row, a metric-completeness datum, or a* Real$^{\uparrow}$ *completeness claim.*

Proof. First read the terminal route. Apply Definition 126.114. The accepted terminal Real$^{\uparrow}$ read factors through exactly the displayed StreamName window rows, accepted RegSeqRat$^{\uparrow}$ pointwise seal rows, Real$^{\uparrow}$ finite-window seal boundary, hsame transports, finite Cont ledger, bundle membership, observations, and Pkg provenance. Apply Definition 126.125 to include the DyadicRatCore$^{\uparrow}$ tolerance ledger in the same finite route.

Now compare the resulting route with the four-face Phase Real and Completion packet P. The agreement assumptions identify the StreamName window, DyadicRatCore tolerance, RegSeqRat seal and readback, Real seal, BHist, hsame, Cont, Pkg, and NameCert coordinates of the terminal read with the corresponding objectwise faces of P. Apply Definition 38.494 to read the exit threshold from those four local closurestatus records.

For the converse direction, start with a terminal finite-window Real$^{\uparrow}$ read that already carries the objectwise threshold and the displayed finite route rows. Apply Definition 126.125 to recover the finite StreamName–RegSeqRat route, and then apply Definition 126.114 to repack the same rows as the terminal pullback boundary.

It remains to keep the terminal comparison at the same threshold. Apply Definition 38.844 to descend any finite-window cover presentation back to the coarse packet P. Finally apply Definition 38.833 and Definition 38.834. These keep formalstatus, bridge, Lean-target, quotient-stream, selected-limit, choice, metric-completeness, and Real$^{\uparrow}$ completeness data attached only to the source faces, so the pullback coherence gives no route-level strengthening. □

Theorem 38.846 (Real L10 terminal cover coordinate exhaustion)**.** *Let T be an accepted terminal finite-window* Real$^{\uparrow}$ *read over the public* StreamName$^{\uparrow}$–RegSeqRat$^{\uparrow}$ *route. A finite certified cover witnesses the current Phase Real and Completion threshold for T if and only if every cover member and every certified overlap exposes exactly the displayed* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledger, the* StreamName$^{\uparrow}$ *finite-window schedule, the* RegSeqRat$^{\uparrow}$ *regularity and classifier rows, the local* Real$^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

No aggregate formalstatus*, aggregate bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or* Real$^{\uparrow}$ *completeness coordinate can stand in for any one of these exposed coordinates.*

Proof. For the forward direction, start with a finite certified cover witnessing the terminal threshold for T. Apply Definition 38.845 to identify the terminal read with the four-face Phase Real and Completion packet through the displayed StreamName, DyadicRatCore, RegSeqRat, local Real, and shared support rows.

Apply Definition 38.494. The threshold is objectwise, so each cover member must present the dyadic endpoint and radius, finite-window schedule, regularity and classifier rows, local seal rows, and the common BHist, hsame, Cont, Pkg, and NameCert support. Apply Definition 38.843; the certified overlaps are precisely the rows along which those objectwise faces glue.

For the reverse direction, suppose every member and overlap exposes exactly those coordinates. Apply Definition 38.843 to assemble the memberwise four-face packets over the shared support, and then apply Definition 38.844 to read the glued packet back as the terminal finite-window threshold for T.

It remains to show that no route-level coordinate can replace a displayed face. Apply Definition 38.833 and Definition 38.834. These keep aggregate status, bridge, quotient-stream, selected-limit, countable-choice, metric-completeness, and $\mathsf{Real}^{\uparrow}$ completeness rows outside the finite cover support, so the witness is exhausted by the listed coordinates and by no stronger route credential. □

Theorem 38.847 (Real L10 terminal finite-cover threshold equivalence)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read over the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route, with displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then T satisfies the current Phase Real and Completion objectwise exit threshold if and only if every finite certified cover of the displayed terminal route glues and descends to the same four-face packet over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *seal, with the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The equivalence refuses aggregate route-level formalstatus*, aggregate bridge rows, quotient stream equality, selected limits, countable choice, metric-completeness data, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, apply Definition 38.845. The accepted terminal read is compared with the four-face Phase Real and Completion packet through the displayed StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback and classifier rows, local Real seal, and shared support rows.

Now apply Definition 38.843. Each finite certified cover member carries the objectwise threshold read from the same local faces, and the overlap hypotheses are exactly the shared BHist, hsame, Cont, Pkg, and NameCert rows. Apply Definition 126.504 to keep the glued cover inside the terminal four-object packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

For the reverse direction, suppose every finite certified cover of the displayed terminal route glues and descends to the same four-face packet. Apply Definition 38.844. The descended coarse packet carries the same objectwise threshold as its finite cover members.

It remains to identify this descended packet with the accepted terminal route. Apply Definition 291.107 to read the dyadic endpoint, radius, distance, and local NameCert interface as the $\mathsf{DyadicRatCore}^{\uparrow}$ face of the same support packet. Apply Definition 301.93 to read the StreamName, RegSeqRat, DyadicRatCore, and local Real rows as one audit-clean four-face exit surface.

Finally apply Definition 38.845. The terminal read and the descended four-face packet have the same displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert support rows, so the threshold is exactly the current objectwise Phase Real and Completion threshold. The cited noncollapse clauses in the gluing, descent, interface, readiness, and terminal-pullback results leave no aggregate status, bridge row, quotient stream equality, selected limit, choice principle, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 38.848 (Real L10 finite-cover terminal support exhaustion)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read over the public* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route. A finite certified cover witnesses the current Phase Real and Completion threshold for T if and only if its members and certified overlaps expose exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Deleting any one of those displayed rows prevents the altered cover from witnessing the same terminal threshold. No aggregate formalstatus *row, bridge row, quotient stream, selected limit,*

countable-choice datum, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ *completeness row replaces the deleted coordinate.*

Proof. For the equivalence of the support inventory with the terminal threshold, combine Definition 38.846 with Definition 38.847. The first result identifies the exact member and overlap coordinates of an accepted terminal cover, while the second reads those coordinates as the current Phase Real and Completion finite-cover threshold for T.

Now delete one displayed row. If the deleted row lies on the $\mathsf{RegSeqRat}^{\uparrow}$ audit-threshold side, apply Definition 301.231; the altered cover cannot descend to the same glued audit-threshold read. If the deleted row is one of the shared cover-support rows along the public route, apply Definition 126.665; the cover-descent terminal support packet cannot present the same $\mathsf{Real}^{\uparrow}$ seal-visible source.

It remains to keep the dyadic endpoint and radius ledger from being hidden in a different terminal readback. Apply Definition 291.124. The public tolerance read is determined by the finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the StreamName window, the RegSeqRat rows, the local Real seal, and the same shared support rows, so deleting any one of those rows removes a coordinate used by the terminal threshold.

Finally use the noncollapse clauses cited in Definition 38.846 and Definition 38.847. They keep aggregate status, bridge data, quotient streams, selected limits, choice, metric-completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the objectwise finite-cover support. Hence none of those rows can replace the deleted coordinate, and the original finite certified cover is support-exhaustive. □

Theorem 38.849 (Real L10 terminal finite-cover face minimality)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read satisfying the current Phase Real and Completion objectwise exit threshold, with displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then every finite certified cover witnessing the terminal finite-cover threshold equivalence for T is face-minimal in the following sense.*

1. *The cover must retain the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$ *faces on the displayed terminal support.*
2. *The cover must retain the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows through the same common support.*
3. *Deleting any one of these faces or support rows makes the resulting subcover inadmissible as a witness for the terminal threshold.*
4. *No aggregate* formalstatus*, bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row can replace the deleted face or support row.*

Proof. Apply Definition 38.847. The accepted threshold for T is equivalent to gluing and descent of finite certified covers to the same four-face packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ seal, with the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 38.845. This identifies the accepted terminal read with that four-face packet, so a cover witnessing the threshold is not merely a route credential: it is a cover whose visible support contains the dyadic ledger, the StreamName finite window, the RegSeqRat classifier and readback rows, the local Real seal, and the shared support rows.

It remains to rule out removing a displayed face. Use Definition 126.407 for the StreamName finite-window subcover, use Definition 301.198 for the RegSeqRat common-tail and cocycle rows,

and use Definition 291.107 for the dyadic endpoint, radius, distance, and local NameCert interface. These results make each face visible in the common support consumed by the terminal packet.

Finally apply the noncollapse clauses from Definition 38.847, Definition 38.833, and Definition 38.834. They keep route-level status, bridge, quotient-stream, selected-limit, choice, metric-completeness, and ambient completeness data outside the objectwise face inventory, so none of those rows can compensate for a missing $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, or shared support row. □

Theorem 38.850 (Real L10 finite-cover threshold cocycle uniqueness)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read. Let $\mathcal{U}$ and $\mathcal{V}$ be two finite certified covers of T, each satisfying the terminal finite-cover threshold equivalence. Suppose that, on every common refinement of $\mathcal{U}$ and $\mathcal{V}$, the two covers agree on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then the four-face exit packet glued from $\mathcal{U}$ and the four-face exit packet glued from $\mathcal{V}$ are the same terminal common-refinement packet. This uniqueness statement is only about the displayed finite-cover threshold surface. It does not identify aggregate formalstatus *rows, bridge rows, quotient streams, selected limits, countable-choice data, metric-completeness data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 38.847 to both covers. Each cover is therefore a finite certified presentation whose glued packet and descended packet are the terminal four-face exit over the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces with the same support rows.

Next pass to any common refinement. Apply Definition 126.628 to the finite-window schedule and Definition 301.243 to the RegSeqRat request sections and local Real seal rows. The refinement exposes a single terminal four-face packet on the common StreamName, RegSeqRat, dyadic, and Real boundary.

The dyadic ledger cannot split into two sources on that refinement. Apply Definition 291.114; the common endpoint and radius rows determine the same BEDC-visible dyadic source ledger and hence the same dyadic face of the exit packet.

Finally apply Definition 38.845 and Definition 38.849. Pullback coherence identifies the common four-face packet with the accepted terminal read T, and face minimality prevents any alternative support row from standing in for a displayed face. Thus the two glued packets coincide as the same terminal common-refinement packet, while aggregate status, bridge, quotient-stream, selected-limit, choice, metric-completeness, and ambient completeness coordinates remain outside the construction. □

Theorem 38.851 (Real L10 terminal finite-cover refinement invariance)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read. Let $\mathcal{V}$ be a finite certified refinement of a finite certified cover $\mathcal{U}$ of T. Suppose that every refinement cell preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then $\mathcal{U}$ witnesses the current Phase Real and Completion terminal finite-cover threshold for T if and only if $\mathcal{V}$ witnesses that same threshold. The equivalence is an objectwise four-face statement: it does not assert equality of aggregate formalstatus *rows, bridge rows, quotient streams, selected limits, choice data, metric-completeness data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Assume first that $\mathcal{U}$ witnesses the terminal finite-cover threshold. Apply Definition 38.847 to $\mathcal{U}$. The cover therefore presents the terminal four-face exit packet over the displayed dyadic, StreamName, RegSeqRat, and local Real faces with the shared support rows.

Transport this packet across the certified refinement. Apply Definition 126.407 to keep the finite-window schedule exact on the terminal subcover, and apply Definition 301.198 to keep the RegSeqRat classifier, readback, and cocycle rows on the common tail. The dyadic endpoint and radius ledger is fixed by Definition 291.114.

Now apply Definition 38.843. Each cell of $\mathcal{V}$ carries the same displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert rows as its parent cell in $\mathcal{U}$, so the refined cells glue to the same terminal exit packet. Then Definition 38.847 reads the glued refined packet as a witness for $\mathcal{V}$.

For the reverse direction, assume that $\mathcal{V}$ witnesses the threshold. Apply Definition 38.844 along the certified refinement map from $\mathcal{V}$ to $\mathcal{U}$. The preservation hypotheses make the descended packet carry exactly the displayed dyadic, StreamName, RegSeqRat, local Real, and shared support rows of $\mathcal{U}$.

Finally apply Definition 38.849 and Definition 38.850. Face minimality prevents the refinement from hiding or replacing any displayed face, and cocycle uniqueness identifies the refined and descended packets on their common terminal support. Thus Definition 38.847 certifies $\mathcal{U}$ exactly when it certifies $\mathcal{V}$.

The construction never promotes route-level data into an objectwise face. Aggregate formalstatus rows, bridge rows, quotient streams, selected limits, choice principles, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the refined terminal finite-cover threshold. □

Theorem 38.852 (Real L10 refinement-support descent noescape)**.** *Let T be an accepted terminal finite-window* $\mathsf{Real}^{\uparrow}$ *read. Let $\mathcal{V}$ be a finite certified refinement of a finite certified cover $\mathcal{U}$ of T. Suppose the refinement preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Then descent from $\mathcal{V}$ to $\mathcal{U}$ preserves exactly this support surface. It introduces no aggregate formalstatus *row, bridge row, quotient stream, selected limit, choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 38.851. The terminal threshold witnessed on the refined cover is equivalent to the terminal threshold witnessed on the original cover, and the equivalence is already restricted to the objectwise dyadic, StreamName, RegSeqRat, local Real, and shared BEDC support rows.

The descent keeps the three lower faces fixed. Apply Definition 126.407 for the finite-window schedule, apply Definition 301.198 for the RegSeqRat tail, and apply Definition 291.114 for the dyadic endpoint and radius ledger. These certificates leave the descended packet on the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ faces as the refined packet.

It remains to exclude substitute support. Apply Definition 38.849 to prevent any hidden row from standing in for a displayed face, and apply Definition 38.850 to identify the descended and refined terminal packets on their common support. Hence the descent preserves precisely the listed surface and cannot create a route-level credential outside it. □

Theorem 38.853 (Real stream prefix tail endpoint projection)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices n, m, if the prefix through n is classified and every endpoint along the finite tail of length m is rational-history classified, then the selected tail endpoint is rational-history*

classified:

$$\mathsf{RealStreamPrefixClassifier}(x, y, n) \wedge (\forall k,\ 1 \le k \le m \Longrightarrow \mathsf{RatHistoryClassifier}(x(n+k), y(n+k))) \Longrightarrow \mathsf{RatHistoryClassif}$$

Ten-field fit: traditional target finite RealUp prefix endpoint readback; BEDC source RealUp stream prefixes over RatUp point classifiers; carrier rational-observation streams; classifier RealStreamPrefixClassifier*; stability by tail extension and endpoint projection; ledger initial prefix length, tail length, and selected endpoint; core theorem this projection; exactness selected endpoint classification; standard bridge RegSeqRat finite observation readback; status* ProofTarget.

Proof. Apply Definition 38.115 to the classified prefix through n and the finite tail endpoint hypothesis. This gives $\mathsf{RealStreamPrefixClassifier}(x, y, n + m)$.

Now apply Definition 38.111 at the endpoint index $n + m$. The resulting projection is exactly $\mathsf{RatHistoryClassifier}(x(n + m), y(n + m))$. □

Theorem 38.854 (Real stream prefix truncated endpoint common e-one tail readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$ and indices m, n, if* $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ *holds, then the endpoint at n has a common e-one tail readback:*

$$\exists a, b,\ \mathsf{hsame}(x(n), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y(n), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Truncate the displayed prefix classifier to the selected endpoint n and use the denominator package there. Its positive unary denominator rows give unary endpoint witnesses and separate both selected endpoints from e, so unary nonempty readback exposes e-one tails a and b. The truncated e-one pair readback theorem then returns the two unary tail rows and their hsame comparison. □

Lean4 ✓ `BEDC.Derived.RealUp.RealStreamPrefixClassifier truncated endpoint common e1 tail readback`

Theorem 38.855 (Real stream prefix probe endpoint determinacy)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ be a finite certified* ProbeBundle *window and let n be a unary observation with* $\mathsf{InBundle}(n, \mathcal{P})$*. Suppose the window supplies* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ *and, for every displayed tail row selected by $\mathcal{P}$ after n, the finite row* $\mathsf{RatHistoryClassifier}(x(n+k), y(n+k))$ *is present. Then every consumer endpoint selected from $\mathcal{P}$ has a unique* RatHistoryClassifier *readback, and its endpoint admits the same common e-one tail witness after* hsame *transport:*

$$\mathsf{InBundle}(n, \mathcal{P}) \Longrightarrow \forall j,\ \mathsf{InBundle}(j, \mathcal{P}) \Longrightarrow \exists a, b,\ \mathsf{hsame}(x(j), \mathsf{E1}(a)) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHis}$$

The statement reads only the displayed finite rows of $\mathcal{P}$; no observation outside the certified probe window is used.

Proof. Restrict the finite ProbeBundle to the selected prefix through n and to the displayed consumer endpoint j. The membership rows $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{InBundle}(j, \mathcal{P})$ are exactly the evidence that these two observations lie inside the certified window.

Apply Definition 38.853 to the selected prefix and the finite tail rows displayed by $\mathcal{P}$. This gives the endpoint RatHistoryClassifier readback for the consumer endpoint, and uniqueness follows because the readback is selected from the same finite bundle row rather than from an external observation.

Finally apply Definition 38.854 to the resulting classified prefix. The hsame transport keeps the two endpoint histories in the same rational-observation class, so the e-one tail witnesses are the common tail witnesses recorded by the finite probe window. □

Theorem 38.856 (Real terminal seal tail-section coherence)**.** *Let $x, y : \mathbb{N} \to \mathsf{Hist}$ be rational-observation streams, let $\mathcal{P}$ be a finite certified* ProbeBundle *endpoint window, and let $\mathcal{T}$ be the same* StreamName$^{\uparrow}$ *tail-section package consumed by a finite* Real$^{\uparrow}$ *stream prefix handoff. Suppose two finite endpoint reads pass through the same prefix row* RealStreamPrefixClassifier(x, y, n)*, the same displayed* InBundle *endpoint rows of $\mathcal{P}$, and the same tail-section package $\mathcal{T}$. Then the two endpoint reads determine the same terminal* Real$^{\uparrow}$ *seal route up to the displayed* hsame *transports.*

The seal route is the finite route through the prefix endpoint projection, truncated common e-one tail readback, probe endpoint determinacy, and the certified StreamName$^{\uparrow}$ *tail-section induction bridge. It does not export quotient stream equality, host sequence equality, selected limits, countable choice, ambient* Real$^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. First apply endpoint projection, Definition 38.853. Both reads use the same prefix row and the same finite tail rows selected by $\mathcal{P}$, so the endpoint RatHistoryClassifier readback is taken from the same finite endpoint surface.

Next apply truncated endpoint common e-one tail readback, Definition 38.854. The endpoint histories therefore expose common e-one tail witnesses, and the displayed hsame rows transport those witnesses along the same prefix package.

Now use probe endpoint determinacy, Definition 38.855. It identifies the endpoint readback supplied by $\mathcal{P}$ as the unique finite readback for that endpoint inside the certified probe window.

Finally apply the StreamName tail-section induction bridge Definition 126.694. Since both reads consume the same StreamName$^{\uparrow}$ tail-section package, the terminal seal route is carried over the same finite BHist, hsame, Cont, Pkg, and local NameCert support rows. No remaining coordinate can distinguish two terminal Real$^{\uparrow}$ seal routes. □

Theorem 38.857 (Real stream prefix two-probe endpoint coherence)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows. Suppose the same unary observation j satisfies* InBundle$(j, \mathcal{P})$ *and* InBundle$(j, \mathcal{Q})$*, and suppose both windows supply the* RealStreamPrefixClassifier *row and the finite* RatHistoryClassifier *tail rows required by Definition 38.855 for the selected endpoint j. Then the endpoint readbacks* RatHistoryClassifier$(x(j), y(j))$ *obtained through $\mathcal{P}$ and through $\mathcal{Q}$ are the same up to* hsame*, and their common e-one tail witnesses agree up to* hsame *transport:*

$$\begin{aligned}
&\mathsf{InBundle}(j, \mathcal{P}) \wedge \mathsf{InBundle}(j, \mathcal{Q}) \\
&\quad \textit{and the two displayed endpoint-determinacy packets at } j \\
&\quad \textit{determine histories } a_{\mathcal{P}}, b_{\mathcal{P}}, a_{\mathcal{Q}}, b_{\mathcal{Q}} \textit{ with} \\
&\quad \mathsf{hsame}(a_{\mathcal{P}}, a_{\mathcal{Q}}) \wedge \mathsf{hsame}(b_{\mathcal{P}}, b_{\mathcal{Q}}) \wedge \mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{P}})) \wedge \mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{Q}})) \\
&\quad \wedge\ \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{P}})) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{Q}})).
\end{aligned}$$

Ten-field fit: traditional target finite RealUp two-window endpoint coherence; BEDC source RealUp stream prefixes over RatUp point classifiers; carrier the two certified ProbeBundle *windows over the same streams; classifier* RealStreamPrefixClassifier *at the selected unary observation; stability by* hsame *transitivity across the two endpoint readbacks; ledger the two window memberships, prefix classifiers, tail rows, endpoint row, and common e-one witnesses; core theorem this two-probe coherence statement; exactness the fixed endpoint row j; standard bridge StreamName and RegSeqRat finite observation readback; status* ProofTarget*. No observation outside the two certified probe windows is used.*

Proof. Apply Definition 38.855 to $\mathcal{P}$ at the selected observation j. The window membership row and the displayed classifier and tail rows return the endpoint RatHistoryClassifier$(x(j), y(j))$ together with common e-one tail witnesses $a_{\mathcal{P}}$ and $b_{\mathcal{P}}$.

Apply the same theorem to $\mathcal{Q}$ at the same selected observation j. This gives the endpoint readback through the second finite window and common e-one tail witnesses $a_{\mathcal{Q}}$ and $b_{\mathcal{Q}}$.

Both applications read the fixed endpoint row j of the same streams x and y. Transitivity of hsame identifies the two rational endpoint readbacks, and the same transport identifies the e-one tail witnesses. Since every row used in the argument is selected from either $\mathcal{P}$ or $\mathcal{Q}$, no external observation enters the comparison. □

Theorem 38.858 (Real stream prefix probe-union endpoint readback)**.** *For rational-observation streams $x, y : \mathbb{N} \to$ Hist, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows, and let $\mathcal{U}$ be their synchronized union window certified by Definition 126.242. Suppose each source window supplies the* RealStreamPrefixClassifier *rows and finite* RatHistoryClassifier *tail rows required by Definition 38.855. Then every unary endpoint j selected by either source window has the same* RatHistoryClassifier$(x(j), y(j))$ *readback through the union window $\mathcal{U}$, and the common e-one tail witnesses displayed by the source window and by $\mathcal{U}$ agree by* hsame *transport.*

Equivalently, if InBundle$(j, \mathcal{P}) \vee$ InBundle$(j, \mathcal{Q})$*, then* InBundle$(j, \mathcal{U})$ *and the union read supplies histories $a_{\mathcal{U}}, b_{\mathcal{U}}$ with*

$$\mathsf{hsame}(x(j), \mathsf{E1}(a_{\mathcal{U}})) \wedge \mathsf{hsame}(y(j), \mathsf{E1}(b_{\mathcal{U}})) \wedge \mathsf{UnaryHistory}(a_{\mathcal{U}}) \wedge \mathsf{UnaryHistory}(b_{\mathcal{U}}).$$

If j lies in both $\mathcal{P}$ and $\mathcal{Q}$, these witnesses are hsame*-identified with the two source-window witnesses of Definition 38.857; if j lies in only one source window, the union read is the unique displayed readback of that source window.*

Ten-field fit: traditional target finite RealUp union endpoint readback; BEDC source RealUp stream prefixes with RatUp point classifiers; carrier the certified ProbeBundle *union membership rows for $\mathcal{P}, \mathcal{Q}, \mathcal{U}$; classifier* RealStreamPrefixClassifier *at the displayed unary prefix; stability by* hsame *transport of the e-one tails; ledger the finite endpoint ledger, source memberships, synchronized union membership, prefix classifiers, and tail rows; core theorem this probe-union readback statement; exactness the fixed finite endpoint row j; bridge data the StreamName synchronized-window union coverage and RegSeqRat finite observation readback; status* ProofTarget*. No quotient-real equality, completeness theorem, or host real comparison is used.*

Proof. Apply Definition 126.242 to the two certified probe windows. It places every endpoint selected by $\mathcal{P}$ or by $\mathcal{Q}$ into the synchronized union window $\mathcal{U}$ and records the finite union membership row used by the later readback.

When the selected endpoint lies only in $\mathcal{P}$, apply Definition 38.855 to $\mathcal{P}$. The union membership supplied by the synchronized carrier reads the same finite endpoint row, so the union readback is the displayed $\mathcal{P}$ readback transported along hsame.

The case where the endpoint lies only in $\mathcal{Q}$ is identical, with $\mathcal{Q}$ supplying the source readback and the union carrier supplying the same endpoint row in $\mathcal{U}$.

If the endpoint lies in both source windows, first apply endpoint determinacy in each window, then apply Definition 38.857. The coherence row identifies the two rational endpoint readbacks and transports the common e-one tail witnesses through hsame, so the union read has the same displayed tails.

All rows consumed by the argument are finite rows of the two source probe windows or of their synchronized union. Hence the endpoint readback remains a RegSeqRat-style finite observation read and never appeals to a quotient real, completeness principle, or host equality. □

Theorem 38.859 (Real stream prefix probe-union terminal four-face factorization)**.** *For rational-observation streams $x, y : \mathbb{N} \to$ Hist, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows and let $\mathcal{U}$*

be their synchronized union window. Suppose the two source windows supply the RealStreamPrefixClassifier *rows and finite* RatHistoryClassifier *tail rows required by Definition 38.858. Then every endpoint readback selected from $\mathcal{U}$ factors through exactly one terminal four-face packet over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, R

The packet has precisely the displayed rows

BHist-*indexed finite probe histories*, hsame-*transport for endpoint tails*, Cont-*continuity rows*, Pkg-*package rows for the synchronized union window*, NameCert-*certification rows for the four visible faces.*

Its $\mathsf{StreamName}^{\uparrow}$ *face is the synchronized carrier $\mathcal{U}$, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the common finite scale window read from the same endpoint ledger, its* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the regular sequence readback, and its* $\mathsf{Real}^{\uparrow}$ *face is the local endpoint seal supplied by the union readback. There is no second terminal packet with the same displayed rows.*

The factorization does not use quotient-real equality, selected limit values, ambient completeness of $\mathsf{Real}^{\uparrow}$*, countable choice, or standard bridge data.*

Proof. First apply Definition 38.858. The synchronized union window supplies the endpoint RatHistoryClassifier readback and the common e-one tail witnesses, all selected from the finite rows of $\mathcal{P}, \mathcal{Q}, \mathcal{U}$.

Next apply Definition 126.242 and Definition 126.558. These two results identify the $\mathsf{StreamName}^{\uparrow}$ carrier row with a terminal finite-request section over the same synchronized window; no endpoint outside $\mathcal{U}$ is introduced.

Now apply Definition 301.239. The RegSeqRat readback and the local Real endpoint seal are thereby read as two faces of a single terminal packet whose support rows are exactly the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 291.83. It supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ common-scale face and proves that the four visible faces form one pullback packet. Terminality gives uniqueness of the packet with those rows, while the construction never introduces quotient-real equality, selected limits, ambient completeness, countable choice, or standard bridge data. □

Theorem 38.860 (Real stream prefix probe-union terminal four-face support minimality)**.** *For rational-observation streams $x, y : \mathbb{N} \to$* Hist*, finite certified probe windows $\mathcal{P}, \mathcal{Q}$, and their synchronized union $\mathcal{U}$, every accepted terminal packet produced by Definition 38.859 is support-minimal. Deleting or replacing any one of the four faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, or deleting or replacing any displayed* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *support row, destroys the accepted probe-union terminal read.*

The missing face or support row cannot be supplied by quotient-real equality, selected limit values, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, metric-completeness rows, aggregate status rows, or standard bridge data. These coordinates remain outside the local probe-union packet and do not substitute for the finite BEDC support rows used by the terminal read.*

Proof. First apply Definition 38.859. It identifies the accepted probe-union read with a unique terminal packet whose visible faces are exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ and whose support is the displayed BHist, hsame, Cont, Pkg, and NameCert row family.

For the finite-window carrier face, apply Definition 126.653 to the synchronized StreamName window underlying $\mathcal{U}$. That result rules out deletion or replacement of the StreamName face and of the finite probe-history rows that certify membership in the accepted window.

For the dyadic request and regular tail faces, apply Definition 301.177. The common finite scale window and the RegSeqRat tail-budget rows are read from the same endpoint ledger, so removing

either face, or any of the hsame, Cont, Pkg, or NameCert rows it consumes, leaves no accepted regular tail read for the probe-union endpoint.

Finally apply Definition 38.868 together with Definition 38.741. These pull the Real-side endpoint seal back to the same four-face terminal packet and rule out substitution by host-real equality, selected limits, completeness coordinates, choice principles, aggregate status rows, or bridge evidence. The two-axis roadmap discipline keeps those excluded coordinates separate from the local BEDC support rows, so none of them can repair a missing face or row. □

Theorem 38.861 (Real stream prefix probe-union support-row exchange invariance)**.** *For rational-observation streams $x, y : \mathbb{N} \to \mathsf{Hist}$, let $\mathcal{P}$ and $\mathcal{Q}$ be finite certified* ProbeBundle *windows, and let $\mathcal{U}$ be their synchronized union window. Consider any two presentations of the terminal packet supplied by Definition 38.859. Suppose both presentations preserve the same four visible faces* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *projections.*

Then exchanging the presentation order of those already-displayed support rows does not produce a second terminal packet. The two presentations determine the same terminal packet over $\mathcal{U}$, and for every selected endpoint j they determine the same finite endpoint readback $\mathsf{RatHistoryClassifier}(x(j), y(j))$ *with the same* hsame*-transported common e-one tail witnesses.*

This invariance is local to the displayed BEDC support rows. It is not witnessed by quotient-real equality, selected limit values, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, aggregate status rows, or bridge data, and none of those coordinates can distinguish two presentation orders of the same terminal probe-union packet.*

Proof. First apply Definition 38.858. Both presentations read each selected endpoint from the same synchronized union window $\mathcal{U}$ and the same finite endpoint ledger, so the rational endpoint row and the common e-one tail witnesses agree by the displayed hsame transport.

Next apply Definition 38.859. The two presentations expose the same four faces $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{R}$ and the same BHist, hsame, Cont, Pkg, and NameCert projections. Terminality therefore identifies them as one terminal packet over the synchronized union carrier rather than as two packets distinguished by row order.

Finally apply Definition 38.860. Support minimality says that the accepted terminal read is carried exactly by the displayed faces and support rows. Hence quotient-real equality, selected limits, ambient completeness, countable choice, aggregate status rows, and bridge data cannot replace one of those rows or separate two orderings of the same row family. The endpoint readback equality from the first paragraph is therefore the endpoint readback of the unique terminal packet. □

Theorem 38.862 (Real selected transport synchronized e-one classifier clique)**.** *Under the hypotheses of Definition 38.866, the four displayed tails a, a', b, b' form a complete displayed rational-classifier clique:*

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \\ &\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b)) \wedge \\ &\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(b')).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport common rational classifier class`

Proof. Apply Definition 38.866 to the displayed hypotheses and project the synchronized displayed rows. This gives unary rows for a, a', b, b' and the four tail rows $\mathsf{hsame}(a, a')$, $\mathsf{hsame}(b, b')$, $\mathsf{hsame}(a, b)$, and $\mathsf{hsame}(a', b')$.

The two remaining tail rows come from Definition 3.8. Compose $\mathsf{hsame}(a, b)$ with $\mathsf{hsame}(b, b')$ to obtain $\mathsf{hsame}(a, b')$. Compose the symmetry of $\mathsf{hsame}(a, a')$ with $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(a', b)$.

For each displayed pair among (a, a'), (a, b), (a, b'), (a', b), (a', b'), and (b, b'), use the corresponding two unary rows and tail hsame row. The reverse direction of Definition 37.49 lifts that data to the six displayed RatHistoryClassifier rows, which are exactly the complete clique. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSelectedTransport synchronized e1 classifier clique`

Fit: traditional target synchronized RegSeqRat selected-observation classifier completion; BEDC source RealUp stream classifier plus RatUp e-one classifier exactness plus FKernel hsame; carrier rational-observation streams with selected displayed e-one tails; classifier RealStreamClassifier feeding RatHistoryClassifier through E1 displays; stability pointwise hsame and tail-level symmetry and transitivity; ledger selected index, contexts, transported endpoints, unary tails, and six pairwise classifier rows; core theorem Definition 38.862 in future Lean namespace `BEDC.Derived.RealUp`; exactness predicate complete displayed e-one rational-classifier clique; downstream use RealUp dyadic endpoint normalization; status ProofTarget.

Theorem 38.863 (Real selected transport synchronized appended-tail classifier clique)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', d_X, d_Y, e_X, e_Y, w_X, w_Y, z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport supplies the displayed classifier clique together with the appended-tail exclusions:

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b)) \wedge \\
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(b')) \wedge \\
&(\mathsf{hsame}(o_X(n), d_X \cdot \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), d_Y \cdot \mathsf{E0}(z_Y)) \to \mathsf{False}) \wedge \\
&(\mathsf{hsame}(x'(n), e_X \cdot \mathsf{E0}(w_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), e_Y \cdot \mathsf{E0}(w_Y)) \to \mathsf{False}).
\end{aligned}$$

Proof. Apply Definition 38.862 to the selected-transport hypotheses. This yields the six displayed RatHistoryClassifier rows as a single clique package.

Apply Definition 38.622 to the same selected-transport hypotheses. Project its appended-tail component to obtain the four exclusions for $o_X(n)$, $o_Y(n)$, $x'(n)$, and $y'(n)$.

Conjoin the six classifier projections with the four appended-tail projections. The resulting ten conjuncts are exactly the displayed conclusion. □

Fit: traditional target selected RegSeqRat endpoint normalization; BEDC source RealStreamClassifier, RatHistoryClassifier, Cont, and hsame; carrier rational-observation streams; classifier RealStreamClassifier feeding displayed RatHistoryClassifier rows; stability pointwise transport, context closure, e-one inversion, and positive-denominator obstruction; ledger selected index, four displayed tails, context outputs, and appended-tail probes; core theorem Definition 38.863; exactness classifier clique plus malformed-tail exclusion; standard bridge dyadic selected endpoint normalization; status ProofTarget.

Theorem 38.864 (Real selected transport common rational classifier class)**.** *Under the hypotheses of Definition 38.866, the displayed selected representative* $\mathsf{E1}(a)$ *is a RatUp classifier hub for the contextual outputs and the transported selected source endpoints:*

$$\begin{aligned}&\mathsf{RatHistoryClassifier}(o_X(n), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(o_Y(n), \mathsf{E1}(a)) \wedge \\ &\mathsf{RatHistoryClassifier}(x'(n), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(y'(n), \mathsf{E1}(a)) \wedge \\ &\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(b'), \mathsf{E1}(a)).\end{aligned}$$

Proof. Apply Definition 38.866 to the displayed hypotheses. Project the displayed endpoint rows $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$, $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$, $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$, and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$, together with the selected displayed classifier row $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 38.862 under the same hypotheses. This supplies the complete displayed e-one classifier clique. Use Definition 37.55 to orient the rows through the chosen center, giving $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(a))$, $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a))$, and $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b'), \mathsf{E1}(a))$.

The self-row $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(a))$ follows by composing $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$ with its symmetric row using Definition 37.53. Transport this self-row along $\mathsf{hsame}(o_X(n), \mathsf{E1}(a))$ and $\mathsf{hsame}(x'(n), \mathsf{E1}(a))$ by Definition 37.54, with FKernel hsame symmetry and reflexivity supplying the needed endpoint orientations. This gives the $o_X(n)$ and $x'(n)$ classifier rows centered at $\mathsf{E1}(a)$.

Finally transport $\mathsf{RatHistoryClassifier}(\mathsf{E1}(b), \mathsf{E1}(a))$ along $\mathsf{hsame}(o_Y(n), \mathsf{E1}(b))$ and $\mathsf{hsame}(y'(n), \mathsf{E1}(b))$, again using Definition 37.54 with the appropriate FKernel hsame orientations. The four transported endpoint rows together with the three oriented displayed-tail rows are exactly the stated common classifier class. □

Fit: traditional target common selected rational observation representative; BEDC source RealUp selected transport plus RatUp classifier transport plus FKernel hsame; carrier rational-observation streams; classifier RatHistoryClassifier centered at $\mathsf{E1}(a)$; stability selected endpoint hsame transport and classifier symmetry; ledger selected index, contextual outputs, transported source endpoints, and displayed e-one tails; core theorem Definition 38.864; exactness all selected endpoints read through the same rational classifier class; standard bridge RegSeqRat selected dyadic representative; status ProofTarget.

Theorem 38.865 (Real selected transport synchronized package from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($

and the selected e-one displays

$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a)$

Then the selected transport carries the following packages simultaneously.

1. *The contextual endpoint package of Definition 38.599:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)),\\
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)),\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)),\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}),\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}),\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).
\end{aligned}$$

2. *The contextual e-one tail determinacy package of Definition 38.601:*

$$\begin{aligned}
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

3. *The finite-prefix selected full-shape readback package of Definition 38.603:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge\\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n\\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

4. *The synchronized displayed rows*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This produces the derived row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$.

Apply Definition 38.621 with the derived prefix row, the source stream classifier row, the two pointwise transport rows, and the remaining context, continuation, and displayed e-one hypotheses. The four returned packages are exactly the four packages displayed above. □

Fit: traditional target selected RealUp observation synchronization from a stream classifier; BEDC source RealUp stream classifier plus RatUp, StreamNameUp, and Cont; carrier rational-observation streams; classifier RealStreamClassifier with derived finite-prefix row; stability pointwise hsame transport, contextual continuation, and e-one inversion; ledger selected index, prefix length, transport rows, contexts, and four tails; core theorem Definition 38.865; exactness the four synchronized packages; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Fit: traditional target synchronized transported RealUp selected-observation certificate; BEDC source RealUp stream transport plus RatUp and FKernel Cont; carrier rational-observation streams; classifier RealStreamClassifier plus displayed RatHistoryClassifier rows; stability pointwise hsame, context closure, stream-classifier finite-prefix closure, e-one inversion; ledger selected index, prefix surplus, four tails, contexts; core theorem Definition 38.621; exactness package plus synchronized classifiers; standard bridge selected RegSeqRat dyadic observation readback; status ProofTarget.

Theorem 38.866 (Real selected transport synchronized fit from stream classifier)**.** *For rational-observation streams* $x, x', y, y', p_X, p_Y, t_X, t_Y, m_X, m_Y, o_X, o_Y : \mathbb{N} \to \mathsf{Hist}$*, indices* m, n*, and histories* $a, a', b, b', z_X, z_Y : \mathsf{Hist}$*, assume the pointwise transport rows and the source stream classifier row*

$$\forall i,\ \mathsf{hsame}(x(i), x'(i)), \quad \forall i,\ \mathsf{hsame}(y(i), y'(i)), \quad \mathsf{RealStreamClassifier}(x, y).$$

Assume also the selected transported context rows

$$\mathsf{UnaryHistory}(p_X(n)), \quad \mathsf{UnaryHistory}(t_X(n)), \quad \mathsf{hsame}(p_X(n), p_Y(n)), \quad \mathsf{hsame}(t_X(n), t_Y(n)),$$

the selected continuation rows

$$\mathsf{Cont}(p_X(n), x'(n), m_X(n)), \quad \mathsf{Cont}(m_X(n), t_X(n), o_X(n)), \quad \mathsf{Cont}(p_Y(n), y'(n), m_Y(n)), \quad \mathsf{Cont}(m_Y(n), t_Y(n), o_Y($$

and the selected e-one displays

$$\mathsf{hsame}(o_X(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(o_X(n), \mathsf{E1}(a')), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b)), \quad \mathsf{hsame}(o_Y(n), \mathsf{E1}(b')), \quad \mathsf{hsame}(x'(n), \mathsf{E1}(a$$

Then the selected transport carries the following packages simultaneously.

1. *The contextual endpoint package of Definition 38.599:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(o_X(n), o_Y(n)),\\
&\mathsf{PositiveUnaryDenominator}(o_X(n)) \wedge \mathsf{PositiveUnaryDenominator}(o_Y(n)),\\
&\mathsf{UnaryHistory}(o_X(n)) \wedge \mathsf{UnaryHistory}(o_Y(n)),\\
&(\mathsf{hsame}(o_X(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{e}) \to \mathsf{False}),\\
&(\mathsf{hsame}(o_X(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(o_Y(n), \mathsf{E0}(z_Y)) \to \mathsf{False}),\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).
\end{aligned}$$

2. *The contextual e-one tail determinacy package of Definition 38.601:*

$$\begin{aligned}
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

3. *The finite-prefix selected full-shape readback package of Definition 38.603:*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b) \wedge\\
&\mathsf{PositiveUnaryDenominator}(x'(n)) \wedge \mathsf{PositiveUnaryDenominator}(y'(n)) \wedge \mathsf{UnaryHistory}(x'(n)) \wedge \mathsf{UnaryHistory}(y'(n\\
&(\mathsf{hsame}(x'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{e}) \to \mathsf{False}) \wedge\\
&(\mathsf{hsame}(x'(n), \mathsf{E0}(z_X)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y'(n), \mathsf{E0}(z_Y)) \to \mathsf{False}).
\end{aligned}$$

4. *The synchronized displayed rows*

$$\begin{aligned}
&\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b)) \wedge \mathsf{RatHistoryClassifier}(\mathsf{E1}(a'), \mathsf{E1}(b')) \wedge\\
&\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(a') \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(b') \wedge\\
&\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(a', b').
\end{aligned}$$

Proof. Apply Definition 38.106 to $\mathsf{RealStreamClassifier}(x, y)$ at the finite length $m + n$. This gives the prefix row $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$.

Apply Definition 38.621 to the two pointwise transport rows, the source stream classifier row, the derived prefix row, the selected context rows, the selected continuation rows, and the six selected e-one display rows. Its conclusion is exactly the four packages listed above. □

Fit: traditional target stream-classifier-closed synchronized transported RealUp selected-observation certificate; BEDC source Definition 38.106 plus Definition 38.621; carrier rational-observation streams; classifier RealStreamClassifier with the finite-prefix row derived internally; stability pointwise hsame, context closure, e-one inversion, and selected full-shape readback; ledger selected index, prefix surplus, four tails, contexts, and derived prefix row; core theorem Definition 38.866 in future Lean namespace `BEDC.Derived.RealUp`; exactness predicate the four synchronized selected-transport packages; standard bridge RealUp selected RegSeqRat endpoint synchronization; status ProofTarget.

Theorem 38.867 (Real L10 route sibling exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal read exposed to the L10 Real and Completion route factors through exactly the displayed four sibling certificates*

chapter 291, *chapter* 126, *chapter* 301,
chapter 38.

The route projects only the finite-window StreamName rows, the dyadic-radius and tolerance ledger rows, the regular-sequence readback rows, and the local $\mathsf{Real}^{\uparrow}$ *terminal seal rows, together with their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

No quotient stream, host real equality, selected limit, countable-choice selector, ambient completeness coordinate, metric-completeness coordinate, or standard bridge row is part of that accepted terminal seal read.

Proof. First read the selected transport package through Definition 38.865. It supplies the finite-prefix selected full-shape package, the contextual endpoint package, and the synchronized displayed e-one rows from a single stream classifier source.

Next use Definition 38.866. The fit theorem identifies the same selected data as a synchronized transported $\mathsf{Real}^{\uparrow}$ observation certificate; the selected index, prefix surplus, four tails, context rows, continuation rows, and derived prefix row remain explicit finite data.

The L10 sibling route children then identify where those rows can be consumed. Definition 38.530 places every accepted open-phase seal read over the StreamName, RegSeqRat, DyadicRatCore, and local Real surfaces. Definition 38.413 and Definition 38.534 make this surface deterministic and exclude deletion or replacement of any displayed face. Definition 38.732 carries the same four-face route to the current terminal exit read.

Finally project the four rows named in the statement. The StreamName row is the finite observation window; the DyadicRatCore row is the dyadic endpoint and radius ledger; the RegSeqRat row is the finite regular-sequence request and readback surface; the local $\mathsf{Real}^{\uparrow}$ row supplies the terminal seal face after those three rows are read. Since the cited route theorems expose no constructor for quotient streams, host equality, selected limits, countable choice, ambient completeness, metric completeness, or a standard bridge row, the displayed four sibling certificates exhaust the accepted terminal seal route. □

Theorem 38.868 (Real L10 terminal four-face support minimality pullback)**.** *Let* $\mathcal{T}$ *be an accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal read for the current Phase Real and Completion exit route. If a Real-facing*

terminal presentation has the same carrier verdict and the same classifier verdict as $\mathcal{T}$, then that presentation pulls back to exactly the four sibling certificates

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and to the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The pullback is minimal at the terminal seal: deleting any one of those four sibling certificates, or deleting any one of the displayed support rows, prevents the presentation from carrying the same Real-facing carrier and classifier verdicts.

No quotient stream equality, selected limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate, metric-completeness coordinate, aggregate formalstatus row, or bridge row belongs to this terminal support surface.*

Proof. First read the accepted terminal seal through Definition 38.867. It places every accepted $\mathsf{Real}^{\uparrow}$ terminal seal read on the four sibling certificates $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, carried by the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 126.653 to the StreamName-facing terminal section. The same carrier and classifier verdicts cannot be presented after deleting the finite StreamName window, the dyadic ledger face, the RegSeqRat readback face, the local Real seal face, or any one of the listed support rows.

Finally compare the RegSeqRat terminal boundary by Definition 301.239. That exactness result reads the local $\mathsf{Real}^{\uparrow}$ terminal seal precisely from the compatible four-face packet over the shared support rows, so any Real-facing presentation with the same carrier and classifier verdict must pull back to that packet and to no smaller packet.

The exclusion list is inherited from the three cited route descriptions. Their terminal packets contain no quotient stream equality, selected limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, metric-completeness coordinate, aggregate formalstatus row, or bridge row, so none of those rows can replace a missing sibling certificate or displayed support row at the terminal seal. □

Theorem 38.869 (Real L10 terminal route support induction)**.** *Let*

$$\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_n$$

be a finite replay chain of accepted Real-facing terminal presentations for the current Phase Real and Completion exit route. Suppose each successive replay step preserves the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *terminal-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request and readback row, the local* $\mathsf{Real}^{\uparrow}$ *terminal seal row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then the terminal read of every $\mathcal{T}_i$ lies on the same four-face support surface*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

with those displayed support rows. If a replay step fails to preserve the terminal read, the failure is witnessed by one of the four faces or by one of the displayed support rows.

The induction introduces no quotient stream, host real equality, selected limit, countable-choice selector, ambient completeness coordinate, metric-completeness coordinate, aggregate formalstatus row, or bridge row.

Proof. For the base presentation $\mathcal{T}_0$, apply Definition 38.868. The terminal read pulls back to the four sibling certificates and to the displayed BHist, hsame, Cont, Pkg, and NameCert rows, and no smaller or off-route packet carries the same Real-facing verdicts.

For the successor step, assume $\mathcal{T}_i$ has already been read on that support surface. The replay hypotheses identify the next presentation $\mathcal{T}_{i+1}$ on the same dyadic endpoint and radius ledger, the same StreamName terminal window, the same RegSeqRat finite request and readback row, the same local Real seal row, and the same displayed support rows.

Apply Definition 126.660 to replay the terminal-window support through the ordered four faces. Then use Definition 301.54 to carry the finite request and readback witness across the RegSeqRat-to-Real seal boundary. Finally, Definition 38.867 says that every accepted Real-facing terminal read on this route is exhausted by exactly those sibling faces and rows.

Thus the successor presentation remains on the same support surface. If the successor replay fails, one of the hypotheses just used is absent: the dyadic ledger, StreamName window, RegSeqRat request/readback face, local Real seal, or one of the displayed BHist, hsame, Cont, Pkg, and NameCert rows has changed or disappeared.

Finite induction over the replay chain now gives the claim for all $\mathcal{T}_i$. The exclusion list is inherited at each step from the minimality pullback and route exhaustion theorems, so no quotient stream, host real equality, selected limit, choice selector, completeness coordinate, formalstatus row, or bridge row can enter the terminal support surface. □

Theorem 38.870 (Real L10 terminal support cover induction)**.** *Let $\mathcal{U}$ be a finite family of accepted Real-facing terminal presentations for the current Phase Real and Completion exit route. Suppose each replay step inside each member of $\mathcal{U}$ preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *terminal-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request and readback row, the local* $\mathsf{Real}^{\uparrow}$ *terminal seal row, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Suppose also that every finite subcover of $\mathcal{U}$ satisfies the objectwise Phase Real and Completion threshold, and that overlaps agree on those four faces and support rows.*

Then the whole replay-cover satisfies the same objectwise Phase Real and Completion threshold. If a member or an overlap fails to contribute to that threshold, the failure localizes to one displayed face

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

or to one displayed support row among BHist, hsame, Cont, Pkg, *and* NameCert.

The cover induction asserts no quotient stream equality, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First run the terminal support induction inside each cover member. Apply Definition 38.869 to each finite replay chain. The result keeps every accepted Real-facing terminal presentation on the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ support surface, with the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Next compare the StreamName and RegSeqRat cover faces. Apply Definition 126.660 to keep the terminal-window rows stable under replay. Apply Definition 301.245 to transport the finite-request, readback, and regularity rows through the finite four-face cover. The overlap hypotheses are exactly the rows consumed by these two induction principles.

Now glue the local threshold packets. Apply Definition 38.843 to the finite family whose members already satisfy the objectwise Phase Real and Completion threshold on the displayed faces. The overlap agreement supplies the common dyadic ledger, StreamName window, RegSeqRat

readback, local Real seal, BHist, hsame, Cont, Pkg, and NameCert support needed by the gluing theorem.

Finally descend from the glued finite packet to the whole replay-cover. Apply Definition 38.844. Any failed descent hypothesis is one of the same visible faces or support rows, so the failure localizes there and not to an off-route completion coordinate.

The exclusion clauses in the support induction, the StreamName and RegSeqRat cover inductions, and the gluing and descent theorems all refuse quotient stream equality, selected limits, countable choice, metric completeness, aggregate formalstatus rows, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Thus the composed cover result remains a finite four-face terminal support claim and carries no route-level strengthening. □

Theorem 38.871 (Real L10 route-cover phase-exit exactness)**.** *Let $\mathcal{U}$ be a finite replay-cover of accepted Real-facing terminal presentations for the current Phase Real and Completion exit route. Suppose $\mathcal{U}$ satisfies the terminal support cover induction of Definition 38.870, and suppose the four roadmap faces of every cover member and overlap meet the local scoped and audit records required by Definition 1327.16.*

Then $\mathcal{U}$ supplies the public Phase Real and Completion exit read exactly through

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the same-route RealCompletionExactBoundaryUp *witness. The read uses the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows from the Real L10 route, and it admits no quotient stream, selected limit, countable-choice selector, host-completeness row, metric-completeness coordinate, aggregate formalstatus row, or bridge token.*

Proof. First apply Definition 38.870. The finite replay-cover is reduced to the displayed terminal support surface: $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, and the shared BHist, hsame, Cont, Pkg, and NameCert rows. Any failed cover contribution is already localized to one of those rows.

Next apply Definition 38.867 and Definition 38.868. These results keep the Real-facing terminal read on the same L10 route and forbid a smaller packet, off-route packet, quotient stream, selected limit, choice selector, host-completeness row, metric-completeness coordinate, aggregate formalstatus row, or bridge token from carrying the same verdicts.

Now read the phase-exit side. The hypotheses give the local scoped closure records and audit-clean verification records for the four roadmap faces, so Definition 1327.16 projects the accepted packet to exactly

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The same theorem keeps RealCompletionExactBoundaryUp as the boundary witness on the same route rather than as a fifth roadmap object or an aggregate status row.

Combining the two reads gives the public Phase Real and Completion exit criterion from the Real chapter itself. The cover induction supplies the finite Real terminal surface, the route exhaustion and minimality theorems pin that surface to the L10 route, and the phase-exit projection consumes the four local scoped/audit records while preserving the same-route exact-boundary witness. No collapsed completion object, ambient completeness principle, selected limit, quotient stream, countable choice, aggregate formalstatus row, or bridge token is introduced by the composition. □

Theorem 38.872 (Real L10 route-cover phase-exit converse)**.** *Let $\mathcal{U}$ be a finite Real-facing replay-cover whose accepted public read supplies the current Phase Real and Completion exit. Then the public exit read pulls back exactly to the four object faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The pullback is witnessed by the displayed Real L10 support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

carried on the same route. Each of the four faces must carry the local scoped-closure and audit-clean records required by Definition 1327.16; the same-route RealCompletionExactBoundaryUp *row remains a boundary witness and not a fifth roadmap face.*

Equivalently, no cover member, overlap, or public read of $\mathcal{U}$ can serve the Phase Real and Completion exit unless its visible data are already the four listed faces with those displayed support rows and those local scoped/audit records. Quotient streams, selected limits, countable choice, metric completeness, aggregate formalstatus rows, bridge tokens, and ambient $\mathsf{Real}^{\uparrow}$ *completeness are not admissible substitutes for any of the four faces or their local records.*

Proof. Start from the accepted public exit read and restrict it to the finite Real-facing replay-cover. Apply Definition 38.871 to identify the available public read with the Real L10 terminal support surface whenever the local threshold records are present.

It remains to show that no smaller or off-route surface can supply those records. Apply Definition 38.870 to localize every cover contribution to a terminal support row, then apply Definition 38.868. The minimality pullback says that deleting any one of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or local $\mathsf{Real}^{\uparrow}$ faces, or deleting one of the displayed BHist, hsame, Cont, Pkg, or NameCert rows, loses the accepted terminal verdict.

The route itself is fixed by Definition 38.867. Thus any attempted witness outside the L10 route is pushed back to the same four-face support surface, or else is rejected as an off-route consumer packet.

Finally apply Definition 1327.16. The phase-exit projection requires the four roadmap faces to be read through their own local scoped-closure and audit-clean records, while keeping RealCompletionExactBoundary as a same-route boundary witness. The same projection rules out quotient streams, selected limits, countable choice, metric completeness, aggregate formalstatus rows, bridge tokens, and ambient $\mathsf{Real}^{\uparrow}$ completeness as substitutes. Hence the public exit read pulls back exactly to the four displayed faces with the displayed support rows. □

Theorem 38.873 (Real L10 four-face public pullback criterion)**.** *Let $\mathcal{P}$ be a finite consumer packet presented on the Real L10 route at the public Phase Real and Completion boundary. Then $\mathcal{P}$ is exactly the four-face pullback for that boundary if and only if its accepted public data agree componentwise with the following four object faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and tolerance ledger;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request, readback, and classifier rows;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *terminal seal rows.*

The agreement is witnessed using only the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support carried by the Real L10 route. Equivalently, two such finite consumer packets determine the*

same public Phase Real and Completion read precisely when they have the same four listed faces and the same displayed support rows. The criterion does not use quotient stream equality, selected host limits, countable-choice selectors, ambient completeness, metric-completeness rows, aggregate formalstatus rows, or bridge tokens.

Proof. For the forward direction, assume $\mathcal{P}$ is the public four-face pullback. Apply Definition 38.871 to read the public boundary through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the same-route $\mathsf{RealCompletionExactBoundaryUp}$ witness. Then apply Definition 38.867 to remove any off-route source, and apply Definition 38.868 to see that the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows are the whole terminal support.

For the reverse direction, assume the four object faces and the displayed support rows agree componentwise. The Definition 126.152 transports the StreamName finite-window rows through the RegSeqRat and local Real readback surface, while Definition 301.56 normalizes the finite-request, readback, classifier, and seal rows to the visible source format. Finally Definition 38.413 identifies the resulting Real-facing packet with the four-face pullback at the open Phase Real boundary.

Combining the two directions gives the stated if and only if. The exclusion clauses are inherited from route exhaustion, terminal support minimality, the route-cover exactness theorem, and the three finite-window and seal-route pullback criteria just used. Each of those results keeps the construction on finite NameCert support, so quotient stream equality, selected host limits, countable choice, ambient or metric completeness, aggregate formalstatus rows, and bridge tokens never enter the witness. □

Theorem 38.874 (Real L10 terminal cover support normal form)**.** *Let $\mathcal{U}$ be a finite family of accepted Real-facing terminal presentations on the current Phase Real and Completion exit route. Then $\mathcal{U}$ satisfies the terminal cover support normal form if and only if each member of $\mathcal{U}$, and each overlap between two members of $\mathcal{U}$, factors through exactly the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows carried by those faces. No quotient stream, host real equality, selected limit, choice selector, metric-completeness row, ambient completeness coordinate, aggregate formalstatus row, or bridge row is part of this normal form.*

Proof. For the forward direction, assume that $\mathcal{U}$ satisfies the terminal cover support normal form. Apply Definition 38.870 to reduce each member and overlap of the finite cover to terminal support on the Real L10 route. Then apply Definition 126.662 and Definition 301.246. These two finite-cover results identify the StreamName and RegSeqRat contributions with the same four displayed object faces and the same displayed support rows.

It remains to account for the public Phase Real and Completion boundary. Definition 38.873 transports the terminal cover data through the public pullback read. Since the pullback criterion excludes off-surface witnesses, every accepted member and every accepted overlap factors only through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

For the reverse direction, assume that every member and overlap of $\mathcal{U}$ factors through exactly the four displayed faces with the displayed support rows. Apply Definition 38.871 to package this four-face data as Real L10 route cover data at the Phase Real and Completion exit. Then use Definition 38.872 to recover accepted Real-facing terminal presentations from that route cover packet.

Finally apply Definition 38.873 once more to identify the rebuilt terminal presentations with the public four-face pullback. The excluded rows cannot be added during this reconstruction: terminal

support cover induction accounts only for finite support rows, the StreamName and RegSeqRat normal forms account only for finite cover faces, and the public pullback criterion keeps quotient streams, host real equality, selected limits, choice selectors, metric or ambient completeness data, aggregate formalstatus rows, and bridge rows outside the accepted normal form. □

Theorem 38.875 (Real L10 terminal route support bicriterion)**.** *Let $\mathcal{U}$ be a finite Real-facing terminal cover at the current Phase Real and Completion public route. Then $\mathcal{U}$ presents that public route if and only if every member of $\mathcal{U}$, and every overlap between two members of $\mathcal{U}$, factors through exactly* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, *and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by those four faces. Equivalently, two such terminal covers have the same public route read precisely when their* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *faces agree and their displayed support rows agree.*

No quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness is part of either side of the criterion.*

Proof. For the forward direction, assume that $\mathcal{U}$ presents the public Phase Real and Completion route. Apply Definition 38.874 to put the finite terminal cover in support normal form. Then apply Definition 38.873 to identify the public route read with equality of the four displayed object faces and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

For the reverse direction, assume that every member and overlap factors through exactly the four listed faces and the displayed support rows. Apply Definition 38.871 to package these finite rows as Real L10 route-cover data at the Phase Real and Completion exit. Then apply Definition 38.872 to recover the accepted terminal presentations on the same public route.

The equivalence of public route reads is the same argument with two covers in parallel: the normal-form theorem reduces each cover to the four faces and support rows, and the four-face public pullback criterion compares exactly those data. The exclusions are preserved because each cited theorem keeps the route finite and terminal: route-cover exactness and converse introduce only accepted Real L10 rows, the normal form admits only the four displayed faces, and the public pullback criterion refuses quotient streams, selected limits, countable choice, metric or ambient completeness data, aggregate formalstatus rows, and bridge rows. □

Definition 38.876 (Real terminal-route modulus extraction packet)**.** *A Real terminal-route modulus extraction packet is a finite* BHist *read attached to an accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal presentation. It contains exactly the terminal four-face support rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support, plus a visible* RealModulusOfConvergenc *row that is read only through that same terminal support surface. The packet has no coordinate for quotient streams, selected limits, host real equality, countable choice, metric completeness, ambient Real completeness, aggregate formalstatus rows, or bridge tokens.*

Theorem 38.877 (Real terminal-route modulus extraction)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal read that exposes a* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *row factors that row through the same four-face support surface*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the Real L10 route. Thus terminal-route modulus extraction is a finite packet in the sense of Definition 38.876; it does not*

export quotient streams, selected limits, host real equality, countable choice, metric completeness, ambient Real completeness, aggregate formalstatus rows, or bridge tokens.

Proof. First read the accepted terminal seal through Definition 38.867. This places every terminal Real-facing read on the four sibling certificates $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$, carried by the displayed support rows.

Apply Definition 38.868. The same carrier and classifier verdicts cannot be preserved after deleting any one of the four faces or any one of the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Use Definition 38.875 to compare the terminal cover that carries the seal with the cover that carries the exposed $\mathsf{RealModulusOfConvergence}^{\uparrow}$ row. The bicriterion says that equality of the public route read is exactly equality of the four faces and the displayed support rows, so the modulus row cannot be read on a separate support surface.

Finally project the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ carrier from Definition 2074.1. Its dyadic tolerance, finite window, regular readback, monotone ledger, and Real seal rows are already among the terminal four faces or downstream of them. Since both the Real route and the modulus carrier exclude quotient streams, selected limits, host real equality, countable choice, metric completeness, ambient completeness, aggregate formalstatus rows, and bridge tokens, the extraction packet exports none of those objects. □

Theorem 38.878 (Real L10 terminal route support obstruction)**.** *Let $\mathcal{U}$ be a finite Real-facing terminal cover at the current Phase Real and Completion public route. If $\mathcal{U}$ does not present that public route, then some member of $\mathcal{U}$, or some overlap between two members of $\mathcal{U}$, fails to factor through one visible L10 face*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

or through one displayed Real L10 support row among

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The obstruction is local to that missing face or support row. Quotient streams, selected limits, countable-choice selectors, metric or ambient completeness data, aggregate formalstatus rows, and bridge rows cannot repair the failed public route presentation.

Proof. Argue contrapositively from Definition 38.875. If every member and every overlap of $\mathcal{U}$ factors through the four displayed faces and the five displayed support rows, then the bicriterion presents the current Phase Real and Completion public route.

Therefore a finite cover that does not present the public route must fail one of the finite factorization requirements consumed by the bicriterion. Apply Definition 38.874 to put those requirements in terminal-cover normal form. This identifies the possible failure sites with the object rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

It remains to rule out an off-surface rescue. Apply Definition 38.873 to the public boundary read: the public route is determined exactly by the four visible faces and the displayed support rows. Then apply Definition 38.872 to pull any accepted public exit read back to the same Real L10 surface.

The two localization steps leave no constructor for quotient streams, selected limits, countable-choice selectors, metric or ambient completeness data, aggregate formalstatus rows, or bridge rows. Hence any failed finite Real-facing terminal cover is obstructed at one visible face or at one displayed support row, and no off-route datum repairs that failure. □

Theorem 38.879 (Real cofinal window transport)**.** *Suppose two terminal* $\mathsf{Real}^{\uparrow}$ *seal reads display cofinal* $\mathsf{StreamName}^{\uparrow}$ *windows and their* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks agree under the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance classifier. Then the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposes the same public name for both reads. The comparison is made through the displayed finite windows, regular rational readbacks, dyadic tolerance rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *seal row; it introduces no selected limit, quotient representative, host real equality, countable choice, ambient completeness, or bridge row.*

Proof. Begin with the two displayed $\mathsf{StreamName}^{\uparrow}$ windows. Cofinality says that every finite request used by either terminal read is eventually represented in the other visible window schedule, so the comparison remains inside finite stream data.

Read the $\mathsf{RegSeqRat}^{\uparrow}$ rows attached to those windows. Agreement under the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance classifier identifies the regular rational readbacks only through displayed dyadic endpoint, radius, and tolerance rows, not through a hidden limit object.

Apply the existing Real hub routes through chapter 126, chapter 301, and chapter 291. These routes force the terminal Real-facing read to pass through the same finite-window, regular-readback, and dyadic-tolerance faces before the seal is exposed.

Finally use the terminal seal surface of chapter 38. Since the support rows transport and replay the same finite comparison and the local seal row has no coordinate for selected limits, quotient representatives, host real equality, countable choice, ambient completeness, or bridges, the public $\mathsf{Real}^{\uparrow}$ name exposed by the terminal seal is the same for the two cofinal reads. □

Theorem 38.880 (Real cofinal window symmetry)**.** *Suppose two terminal* $\mathsf{Real}^{\uparrow}$ *seal reads display cofinal* $\mathsf{StreamName}^{\uparrow}$ *windows and their* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks agree under the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance classifier. If the first read exposes the public* $\mathsf{Real}^{\uparrow}$ *name of the second through the displayed finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *rows, then the swapped read exposes the same public* $\mathsf{Real}^{\uparrow}$ *name through the same rows.*

The symmetry statement does not add a selected limit, quotient representative, host real equality, countable-choice schedule, ambient completeness coordinate, or bridge row.

Proof. Swap the two terminal seal reads. Cofinality of the displayed $\mathsf{StreamName}^{\uparrow}$ windows is symmetric: every finite request visible from either side is still eventually represented in the other side's finite window schedule after the swap.

The $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance classifier is also read symmetrically on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readbacks. The comparison still uses the same dyadic endpoint, radius, and tolerance rows, transported by the same componentwise hsame data and replayed by the same Cont row.

Apply Definition 38.879 to the swapped pair. Its hypotheses are exactly the cofinal window and classifier-agreement rows just identified, with the Pkg provenance and local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ seal row unchanged.

Thus the swapped terminal read exposes the same public $\mathsf{Real}^{\uparrow}$ name. Since the cited transport theorem has no coordinate for selected limits, quotient representatives, host real equality, countable choice, ambient completeness, or bridge rows, the symmetric route has none. □

Chapter 39

A Concrete Naming Certificate for $\mathsf{Complex}^{\uparrow}$

The complex number interface $\mathsf{Complex}^{\uparrow}$ is licensed here by the product of two certified rational-history components from chapter 37. The carrier, pattern, and ledger predicates are the same rational-pair history carrier, and the classifier is product history sameness restricted to those two rational-history components. It is not a primitive number system; it is licensed only via the rational source and the pair constructor of chapter 34.

Remark 39.1 (Complex certificate scope)**.** *This chapter records the concrete history-level license: a complex history is the continuation product of two rational-history witnesses, and the classifier is the corresponding product classifier over internal history sameness.*

39.1 Carrier definition

Definition 39.2 (Complex carrier)**.** *Use the BEDC product history carrier over two certified rational-history components.*

Lean4 (definition) `BEDC.Derived.ComplexUp.ComplexHistoryCarrier`

Theorem 39.3 (Complex history carrier is unary)**.** *Every complex history carrier is a unary BEDC history.*

Proof. Unpack the complex carrier as a product of two rational-history carriers and a continuation witness. Each rational component gives a positive unary denominator, hence a unary history. Unary histories are closed under the stored continuation, so the complex carrier endpoint is unary. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier unary`

Theorem 39.4 (Complex carrier positive components)**.** *Every complex history carrier exposes real and imaginary rational-history components whose denominators are positive unary histories, together with the continuation witness assembling those components into the complex history.*

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier positive components`

Proof. Unpack the product-history carrier into its real component, imaginary component, component carrier witnesses, and continuation witness. Apply the rational-history carrier characterization to each component to read back the positive unary denominator evidence, and keep the same continuation witness. □

Theorem 39.5 (Complex carrier components are nonempty)**.** *Every complex history carrier exposes rational real and imaginary components whose component histories are not identified with* e*.*

Proof. Unpack the complex carrier into rational real and imaginary components and the continuation assembling them. Rational-history carrier non-emptiness rejects *real*hsamee and *imag*hsamee separately. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier components nonempty`

Theorem 39.6 (Complex e1-tail carrier component readback)**.** *If* ComplexHistoryCarrier(E1($tail$)) *holds, then there are histories real and imagTail such that real is a rational-history component,* E1($imagTail$) *is a rational-history component, and* Cont($real, imagTail, tail$) *holds.*

Proof. Unpack the complex carrier into rational real and imaginary components and the continuation witness assembling them into E1($tail$). Continuation E1-result inversion gives either an empty imaginary component or an E1-headed imaginary component with tail continuation. The empty branch contradicts rational-history carrier non-emptiness; the visible branch exposes the required rational component and tail continuation. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier e1 tail components`

Theorem 39.7 (Complex e1-tail carrier nonempty component readback)**.** *If* ComplexHistoryCarrier(E1($tail$)) *holds, then the exposed real component and the exposed visible imaginary component are rational-history carriers and neither is history-same to* e*.*

Proof. Apply the e1-tail component readback to obtain the rational real component, the rational visible imaginary component, and the tail continuation. Rational history-carrier non-emptiness applies separately to the two exposed component carriers while the same continuation witness is retained. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier e1 tail components nonempty`

Theorem 39.8 (Complex e1-tail carrier positive component readback)**.** *If* ComplexHistoryCarrier(E1($tail$)) *holds, then there are histories real and imagTail such that real is a rational-history component,* E1($imagTail$) *is a rational-history component,* Cont($real, imagTail, tail$) *holds, and both exposed rational components carry positive unary denominators.*

Proof. First apply Theorem 39.6 to read back the real component, the visible imaginary component, and the tail continuation. Then apply the rational-history carrier characterization to both rational component witnesses and keep the same continuation. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier e1 tail positive components`

Theorem 39.9 (Complex carrier excludes an $e0$ endpoint)**.** *No complex history carrier can be an* $e0$*-headed history.*

Proof. Every complex carrier is unary because it is assembled from unary rational-history components by continuation. A zero-headed BEDC history cannot be unary, so ComplexHistoryCarrier(E0(*tail*)) is impossible. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier e0 endpoint absurd`

Theorem 39.10 (Complex carrier component reflexive classifiers)**.** *Every complex history carrier exposes real and imaginary rational-history components whose own rational-history classifiers hold reflexively, together with the continuation witness assembling them into the complex history.*

Proof. Unpack the complex carrier as a product-history carrier over two rational-history carriers. The exposed real and imaginary components already carry rational-history carrier evidence. Pair each component carrier with itself and the reflexive hsame witness for that component to form the two rational-history classifiers, and keep the stored continuation witness. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier component reflexive classifiers`

Theorem 39.11 (Complex carrier left unary context closure)**.** *If p is a unary history and h is a complex history carrier, then* append(p, h) *is again a complex history carrier.*

Proof. Unpack h as a rational real component, a rational imaginary component, and a continuation witness. Close the real component under the left unary denominator context p, keep the imaginary component fixed, and reassociate the continuation by associativity of append. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier prepend unary closed`

Theorem 39.12 (Complex carrier right unary context closure)**.** *If h is a complex history carrier and q is a unary history, then* append(h, q) *is again a complex history carrier.*

Proof. Unpack h as a rational real component, a rational imaginary component, and a continuation witness. Close the imaginary component under the right unary denominator context q, keep the real component fixed, and reassociate the continuation by associativity of append. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier append unary closed`

Theorem 39.13 (Complex carrier unary continuation closure)**.** *If h is a complex history carrier, q is unary, and* Cont(h, q, zq)*, then zq is again a complex history carrier.*

Proof. Unpack h into rational real and imaginary components with the continuation assembling them. Associativity of continuation splits the suffix q through the imaginary component. The imaginary rational carrier is closed under the unary suffix and transported along the split continuation result, while the real rational carrier is kept fixed. The resulting component continuation reassembles zq as a complex history carrier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier unary continuation closed`

Theorem 39.14 (Complex history carrier is not empty)**.** *No complex history carrier can be identified with the empty history.*

Proof. Unpack the complex carrier into its real rational component, imaginary rational component, and continuation witness assembling the two components. Transport the continuation result to the empty endpoint, invert the empty-result continuation, and obtain that the real rational component is empty. This contradicts the rational-history carrier non-emptiness theorem. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryCarrier not empty`

39.2 The certificate

Definition 39.15 (Complex source specification)**.** *The source is* $\mathsf{Prod}^{\uparrow}(\mathsf{Rat}^{\uparrow}, \mathsf{Rat}^{\uparrow})$ *at the history-carrier level.*

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexTasteGate single carrier alignment`

Definition 39.16 (Complex pattern specification)**.** *The pattern predicate is exactly the rational-pair history carrier: a history matches the pattern when it is a product history assembled from two certified rational-history components.*

Definition 39.17 (Complex classifier specification)**.** *The classifier is product history classification over two rational-history carriers.*

Lean4 (definition) `BEDC.Derived.ComplexUp.ComplexHistoryClassifier`

Theorem 39.18 (Complex history classifier nonempty endpoints)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$ *holds, then neither endpoint can be identified with the empty history.*

Proof. Unpack the classifier into the two complex-history carrier witnesses and the stored endpoint hsame link. Apply complex carrier non-emptiness separately to the source endpoint carrier and to the target endpoint carrier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier nonempty endpoints`

Theorem 39.19 (Complex history classifier endpoint e-one shapes)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$ *holds, then both classifier endpoints are visibly one-headed histories with unary tails:*

$$(\exists h_t,\ h = \mathsf{E1}(h_t) \wedge \mathsf{UnaryHistory}(h_t)) \wedge (\exists k_t,\ k = \mathsf{E1}(k_t) \wedge \mathsf{UnaryHistory}(k_t)).$$

Proof. The classifier gives carrier witnesses for both endpoints. Complex carrier unarity supplies unary-history evidence for each endpoint, while classifier non-emptiness rules out the empty history on both sides. Applying unary nonempty one-tail readback to each endpoint gives the displayed shapes. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier endpoint e1 shapes`

Theorem 39.20 (Complex classifier positive components)**.** *Every complex history classifier between two histories exposes real and imaginary rational-history components for both endpoints, the continuation ledgers assembling each endpoint, the endpoint* hsame *witness, and positive unary denominator evidence for all four rational components.*

Proof. Unpack the classifier into its two complex carrier witnesses and the stored endpoint hsame proof. Apply the positive-component readback theorem to each carrier. The resulting rational component carriers, continuation ledgers, endpoint sameness, and positive denominator witnesses assemble directly into the displayed endpoint-pair statement. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier positive components`

Theorem 39.21 (Complex e1-endpoint classifier component readback)**.** *If* $\mathsf{ComplexHistoryClassifier}(\mathsf{E1}(tail), h)$ *holds, or if* $\mathsf{ComplexHistoryClassifier}(h, \mathsf{E1}(tail))$ *holds, then there are histories real and imagTail such that real is a rational-history component,* $\mathsf{E1}(imagTail)$ *is a rational-history component, and* $\mathsf{Cont}(real, imagTail, tail)$ *holds.*

Proof. Unpack the complex classifier into its two endpoint carrier witnesses and the stored endpoint hsame proof. In either endpoint position, the E1-headed endpoint supplies a complex carrier. Apply the E1-tail carrier component readback theorem to that carrier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier e1 endpoint components`

Theorem 39.22 (Complex classifier excludes an $e0$ endpoint)**.** *No complex history classifier can have an $e0$-headed history as either endpoint.*

Proof. Read the $e0$-headed endpoint as a complex carrier and expose its rational real and imaginary components. The continuation result ending in $e0$ either makes the imaginary component empty or makes it itself $e0$-headed. Both cases contradict the positive unary denominator witness attached to the imaginary rational component. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier e0 endpoint absurd`

Theorem 39.23 (Complex classifier excludes empty endpoints)**.** *No complex history classifier can identify either endpoint with the empty history.*

Proof. Unpack the classifier into its two complex carrier witnesses. Apply complex carrier non-emptiness separately to the source endpoint carrier and to the target endpoint carrier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier empty endpoint absurd`

Theorem 39.24 (Complex history classifier transitivity)**.** *If a complex history classifier relates h to k, and also relates k to r, then it relates h to r. The left endpoint carrier is taken from the first classification, the right endpoint carrier is taken from the second classification, and the history-sameness witnesses compose.*

Proof. Unfolding the classifier exposes the two endpoint carrier witnesses together with two *hsame* links. The endpoint carriers are preserved at the outside of the chain, and the internal links compose by transitivity of *hsame*. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier trans`

Theorem 39.25 (Complex history classifier symmetry)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$*, then* $\mathsf{ComplexHistoryClassifier}(k, h)$*.*

Proof. Unfold the classifier as the product history classifier over two RatHistoryCarrier endpoints. The endpoint carrier witnesses are swapped, and the stored $\mathsf{hsame}(h, k)$ witness is reversed by symmetry to obtain $\mathsf{hsame}(k, h)$. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier symm`

Theorem 39.26 (Complex history classifier unary context closure)**.** *If p and q are unary context histories,* $p\mathsf{hsame}p'$*,* $q\mathsf{hsame}q'$*, and* $\mathsf{ComplexHistoryClassifier}(h, k)$*, then the context-wrapped endpoints* $\mathsf{append}(p, \mathsf{append}(h, q))$ *and* $\mathsf{append}(p', \mathsf{append}(k, q'))$ *are again related by the complex history classifier.*

Proof. Unpack the complex classifier into carrier witnesses for h and k together with the endpoint hsame link. Right unary context closure carries both endpoint carriers through the tail contexts, after transporting unary evidence across $q\mathsf{hsame}q'$. Left unary context closure then carries those results through the prefix contexts, after transporting unary evidence across $p\mathsf{hsame}p'$. The three hsame links identify the two fully wrapped histories, giving the classifier endpoint relation. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary context closed`

Theorem 39.27 (Complex classifier unary continuation closure)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$*,* q *is unary,* $q\mathsf{hsame}q'$*,* $\mathsf{Cont}(h, q, zq)$*, and* $\mathsf{Cont}(k, q', wq)$*, then* $\mathsf{ComplexHistoryClassifier}(zq, wq)$*.*

Proof. Apply complex carrier unary continuation closure to both classified endpoints, transporting the unary witness for q across $q\mathsf{hsame}q'$ on the right endpoint. The stored endpoint comparison $h\mathsf{hsame}k$, the suffix comparison $q\mathsf{hsame}q'$, and the two continuation witnesses identify the continuation results, giving the classifier relation between zq and wq. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary continuation closed`

Theorem 39.28 (Complex history classifier unary continuation empty endpoint absurdity)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$*,* q *is unary,* $q\mathsf{hsame}q'$*,* $\mathsf{Cont}(h, q, zq)$*, and* $\mathsf{Cont}(k, q', wq)$*, then neither continued endpoint can be internally same as the empty history:*

$$(\mathsf{hsame}(zq, \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(wq, \mathsf{e}) \to \mathsf{False}).$$

Proof. First close the classifier under the two displayed unary continuations. The continued classifier relates zq and wq. Applying complex classifier empty-endpoint exclusion to that continued classifier gives both displayed absurdity clauses. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary continuation empty endpoint absurd`

Theorem 39.29 (Complex history classifier unary continuation positive components)**.** *If* $\mathsf{ComplexHistoryClassifier}(h$ q *is unary,* $q\mathsf{hsame}q'$*,* $\mathsf{Cont}(h, q, zq)$*, and* $\mathsf{Cont}(k, q', wq)$*, then the continued endpoints* zq *and* wq *expose real and imaginary rational-history components for both endpoints, the two continuation ledgers, the endpoint* hsame *witness, and positive unary denominator evidence for all four rational components.*

Proof. First apply unary-continuation closure to obtain $\mathsf{ComplexHistoryClassifier}(zq, wq)$. Then apply complex classifier positive-component readback to the continued classifier. The readback exposes the four rational component carriers, the two continuation ledgers assembling the continued endpoints, the endpoint hsame witness, and the four positive-denominator witnesses. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary continuation positive components`

Theorem 39.30 (Complex history classifier unary continuation e-one endpoint components)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$*,* q *is unary,* $q\mathsf{hsame}q'$*,* $\mathsf{Cont}(h, q, \mathsf{E1}(a))$*, and* $\mathsf{Cont}(k, q', \mathsf{E1}(b))$*, then both displayed one-extension continuation endpoints expose rational real components, rational one-extension imaginary components, and continuation witnesses assembling their tails:*

$$\exists r, i,\ \mathsf{RatHistoryCarrier}(r) \wedge \mathsf{RatHistoryCarrier}(\mathsf{E1}(i)) \wedge \mathsf{Cont}(r, i, a),$$
$$\exists r', i',\ \mathsf{RatHistoryCarrier}(r') \wedge \mathsf{RatHistoryCarrier}(\mathsf{E1}(i')) \wedge \mathsf{Cont}(r', i', b).$$

Proof. First close the source classifier under the two unary continuation displays. The resulting classifier relates $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$. Apply complex e-one endpoint component readback once to the left displayed endpoint and once to the right displayed endpoint. The two projections give the required rational component witnesses and their continuation ledgers. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary continuation e1 endpoint components`

Theorem 39.31 (Complex history classifier unary continuation e-one endpoint positive components)**.** *If* $\mathsf{ComplexHistoryClassifier}(h, k)$*, q is unary,* $q\mathsf{hsame}q'$*,* $\mathsf{Cont}(h, q, \mathsf{E1}(a))$*, and* $\mathsf{Cont}(k, q', \mathsf{E1}(b))$*, then both displayed one-extension continuation endpoints expose rational real components, rational one-extension imaginary components, continuation witnesses assembling their tails, and positive unary denominator evidence for both rational components:*

$$\exists r, i,\ \mathsf{RatHistoryCarrier}(r) \wedge \mathsf{RatHistoryCarrier}(\mathsf{E1}(i)) \wedge \mathsf{Cont}(r, i, a) \wedge \mathsf{PositiveUnaryDenominator}(r) \wedge \mathsf{PositiveUnary}$$
$$\exists r', i',\ \mathsf{RatHistoryCarrier}(r') \wedge \mathsf{RatHistoryCarrier}(\mathsf{E1}(i')) \wedge \mathsf{Cont}(r', i', b) \wedge \mathsf{PositiveUnaryDenominator}(r') \wedge \mathsf{PositiveU}$$

Proof. First close the source classifier under the two unary continuation displays. The resulting classifier relates $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$. Apply complex one-extension positive-component readback to the left carrier and to the right carrier of this continued classifier. The two carrier projections supply the displayed rational component carriers, tail continuation ledgers, and positive denominator evidence. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary continuation e1 endpoint positive components`

Theorem 39.32 (Complex history classifier unary context positive components)**.** *If p and q are unary context histories,* $p\mathsf{hsame}p'$*,* $q\mathsf{hsame}q'$*, and* $\mathsf{ComplexHistoryClassifier}(h, k)$*, then the context-wrapped endpoints* $\mathsf{append}(p, \mathsf{append}(h, q))$ *and* $\mathsf{append}(p', \mathsf{append}(k, q'))$ *expose real and imaginary rational-history components for both endpoints, the two continuation ledgers, the wrapped endpoint* hsame *witness, and positive unary denominator evidence for all four rational components.*

Proof. Apply unary context closure to the source complex classifier and the matched prefix and suffix contexts. This gives a complex classifier between the two wrapped endpoints. Apply complex classifier positive-component readback to that wrapped classifier. The readback exposes the four rational component carriers, the two continuation ledgers assembling the wrapped endpoints, the endpoint hsame witness, and the four positive-denominator witnesses. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier unary context positive components`

Theorem 39.33 (Complex component classifier introduction)**.** *If the real components are related by* RatHistoryClassifier*, the imaginary components are related by* RatHistoryClassifier*, and the two endpoint histories are assembled by the corresponding continuation witnesses, then the endpoint histories satisfy* ComplexHistoryClassifier*.*

Proof. The source carrier is built from the source real and imaginary rational component carriers and the first continuation witness. The target carrier is built from the target rational component carriers and the second continuation witness. The two component hsame witnesses are transported through the two continuation ledgers by Cont respecting hsame, yielding the endpoint hsame field required by the complex classifier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier component classifier intro`

Theorem 39.34 (Complex component classifier target history-sameness transport)**.** *If the real and imaginary component classifiers assemble endpoints h and k through continuation witnesses, and* $k\mathsf{hsame}k'$*, then h and k′ are related by* ComplexHistoryClassifier*.*

Proof. Transport the target continuation witness along the displayed endpoint history-sameness. The component classifier introduction theorem then rebuilds the complex classifier using the original component classifiers and the transported target continuation. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier component target hsame transport`

Theorem 39.35 (Complex component classifier endpoint history-sameness transport)**.** *If the real and imaginary component classifiers assemble endpoints h and k through continuation witnesses, and $h$hsameh' and $k$hsamek', then h' and k' are related by* ComplexHistoryClassifier*.*

Proof. Transport the source and target continuation witnesses along the two endpoint history-sameness classifiers. The component classifier introduction theorem then rebuilds the complex classifier from the original rational component classifiers and the transported continuation witnesses. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier component hsame transport`

Theorem 39.36 (Complex component classifier unary-context introduction)**.** *If the real components are related by* RatHistoryClassifier*, the imaginary components are related by* RatHistoryClassifier*, the two middle endpoints are assembled by continuation witnesses, and classified unary prefix and suffix contexts surround those endpoints, then the wrapped endpoints are related by* ComplexHistoryClassifier*.*

Proof. First build the middle complex classifier from the real component classifier, the imaginary component classifier, and the two continuation witnesses. Then apply unary-context closure to transport that classifier through the classified prefix and suffix contexts. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier component classifier unary context intro`

Theorem 39.37 (Complex component classifier unary-context positive components)**.** *If rational real and imaginary component classifiers assemble middle endpoints h and k, and classified unary prefix and suffix contexts surround those endpoints, then the wrapped endpoints expose rational real and imaginary components on both sides, the two continuation ledgers, the wrapped endpoint* hsame *witness, and positive unary denominator evidence for all four rational components.*

Proof. First build the wrapped complex classifier from the component classifiers and the classified unary contexts. Then apply complex classifier positive-component readback to that wrapped classifier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryClassifier component classifier unary context positive components`

Definition 39.38 (Complex stability certificate)**.** *The stability certificate consists of:*

1. *an inhabited rational-pair history carrier*
2. *reflexivity of the product classifier on carrier histories*
3. *symmetry of the product classifier*
4. *transitivity of the product classifier*
5. *carrier transport along product-classifier sameness*
6. *pattern soundness for the rational-pair history carrier*
7. *ledger soundness for the rational-pair history carrier*

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexUp StdBridge`

Definition 39.39 (Complex ledger policy)**.** *Record a raw rational-pair complex history carrier together with an* hsame *link to the visible complex history.*

Lean4 (definition) `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy`

Theorem 39.40 (Complex ledger visible carrier)**.** *If a complex ledger relates a raw complex history to a visible history, then the visible endpoint is a complex history carrier.*

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy visible carrier`

Proof. Unpack the raw complex carrier as a product of two rational-history carriers and a continuation witness. Transport the continuation result across the stored raw-visible hsame link and rebuild the rational-pair carrier at the visible endpoint. □

Theorem 39.41 (Complex ledger excludes an $e0$-visible endpoint)**.** *No complex ledger policy can expose an $e0$-headed visible endpoint.*

Proof. The visible-carrier theorem transports the raw rational-pair carrier to the visible endpoint. Exposing the visible endpoint's rational components and inverting the continuation result gives the same two impossible imaginary denominator cases: empty or $e0$-headed. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy e0 visible absurd`

Theorem 39.42 (Complex ledger excludes an $e0$-raw endpoint)**.** *No complex ledger policy can use an $e0$-headed raw endpoint:*

$$\mathsf{ComplexHistoryLedgerPolicy}(\mathsf{E0}(tail), visible) \Longrightarrow \mathsf{False}.$$

Proof. The raw endpoint carrier stored in the ledger exposes rational real and imaginary components and a continuation assembling them into $\mathsf{E0}(tail)$. Continuation $e0$-result inversion says that the imaginary component is empty or $e0$-headed. Both cases contradict the positive unary denominator evidence carried by the imaginary rational component. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy e0 raw absurd`

Theorem 39.43 (Complex ledger raw-visible classifier)**.** *If a complex ledger relates a raw complex history to a visible history, then the raw and visible endpoints satisfy the complex history classifier.*

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy raw visible classifier`

Proof. Unpack the ledger into the raw complex carrier and the raw-visible hsame link. The visible carrier theorem supplies the visible endpoint carrier, and these two carrier witnesses with the stored history link rebuild the restricted product classifier. □

Theorem 39.44 (Complex ledger positive components)**.** *If a complex ledger relates a raw complex history to a visible history, then the raw endpoint exposes real and imaginary rational-history components, the continuation assembling those components into the raw endpoint, the raw-visible* hsame *link, the visible complex carrier, and positive unary denominator evidence for both rational components.*

Proof. Apply the complex carrier positive-component theorem to the raw carrier stored in the ledger. Apply the visible-carrier theorem to the same ledger to recover the visible complex carrier. The stored raw-visible hsame link and the two positive denominator witnesses complete the displayed ledger readback. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy positive components`

Theorem 39.45 (Complex ledger classifier extension)**.** *If a complex ledger relates a raw history to a visible history, and the visible history is classified with a target history, then the raw history is classified with the same target.*

Proof. Unpack the complex ledger into the raw carrier and the raw-visible hsame link. Unpack the visible-target classifier to keep the target carrier and the visible-target hsame link. The raw carrier and target carrier form the endpoint data, and transitivity of hsame composes the two links. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy classifier extension`

Theorem 39.46 (Complex ledger component classifier extension)**.** *If a complex ledger relates a raw endpoint to a visible endpoint, the visible endpoint and a target endpoint are assembled from component histories whose real and imaginary components are classified by* RatHistoryClassifier*, then the raw endpoint is classified with the target endpoint.*

Proof. The component classifiers and continuation witnesses build a visible-target complex classifier. Ledger classifier extension composes the stored raw-visible ledger link with this visible-target classifier, yielding the raw-target complex classifier. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistoryLedgerPolicy component classifier extension`

Theorem 39.47 (Concrete $\mathsf{Complex}^{\uparrow}$ naming certificate)**.** *The rational-pair history tuple forms a valid semantic NameCert for* $\mathsf{Complex}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ComplexUp.complex history semantic name certificate`

Corollary 39.48 ($\mathsf{Complex}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{Complex}^{\uparrow}$ *is licensed as a derived pair interface over* $\mathsf{Rat}^{\uparrow}$*.*

Proof. The semantic naming certificate supplies an inhabited complex carrier. Unfolding the carrier yields rational real and imaginary histories and a continuation assembling them into the visible complex history; the certificate reflexivity field classifies that history with itself. □

Lean4 ✓ `BEDC.Derived.ComplexUp.ComplexHistorySemanticCertificate rat pair reflexive witness`

Closure status for $\mathsf{NameCert}_{\mathsf{Complex}^{\uparrow}}$**.**

Constructive story: A complex number is built by displaying two $\mathsf{Rat}^{\uparrow}$ denominators side by side: a history h is admitted when some left component l and right component r are both rational histories and the kernel continuation witnesses h as their ordered pair. Equality at the complex-history level is literal history-equality on admitted pairs, while the component view remembers that the real and imaginary entries are compared by the $\mathsf{Rat}^{\uparrow}$ classifier on their own sides rather than by a new complex quotient. The source therefore accepts exactly the same rational-pair histories as the carrier, the pattern is the displayed-pair shape together with the cancellation rows that recover l and r, and the ledger keeps the visible complex history aligned with the raw continuation that assembled it. Since the rational classifier is already an equivalence and the continuation pair is deterministic, the carrier has a rational-pair witness and the classifier is reflexive, symmetric, transitive, and closed over admitted histories. A history h lies in the signature gap of a $\mathsf{Complex}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer pair and evidence ledger, and the token-introduction relation names that answer as p; after the pair is decomposed, the real and imaginary rational denominators must match the two components encoded by p. Multiplication is then the familiar displayed formula $(a, b) \cdot (c, d) = (ac - bd,\ ad + bc)$, with every subcalculation performed in $\mathsf{Rat}^{\uparrow}$; applying it to $(0, 1)$ gives $(0 \cdot 0 - 1 \cdot 1,\ 0 \cdot 1 + 1 \cdot 0) = (-1, 0)$, so $i^2 = -1$ follows from the law rather than being postulated. The chapter registers this rational-pair construction under the name $\mathsf{Complex}^{\uparrow}$; the data surface is $\mathsf{Prod}^{\uparrow}(\mathsf{Rat}^{\uparrow}, \mathsf{Rat}^{\uparrow})$ plus the explicit multiplication row, with no quotient, field-completion primitive, or independent extension type introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Complex}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public rational-pair Complex$^{\uparrow}$ certificate binds the real and imaginary Rat$^{\uparrow}$ component classifiers, product-carrier assembly rows, ledger extension rows, and licensed-not-primitive corollary.

Formal verification: FormalStatus(NameCert$_{\text{Complex}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ComplexUp.complex\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full complex-number theory: field operations, conjugation, norm, order-free algebraic closure, topology, limits, holomorphic functions, or complex analysis.

Upgrade path: No further upgrade: chapter at top of formal axis.

Chapter 40

A Concrete Naming Certificate for $\mathsf{Monoid}^{\uparrow}$

The monoid interface $\mathsf{Monoid}^{\uparrow}$ is licensed by an abstract carrier endowed with a binary operation $\cdot : C \times C \to C$, an identity element $e : C$, associativity of $\cdot$, and left and right identity laws of e under the carrier's source classifier. At the abstract interface level, no other licensed object is required as a dependency. It is not a primitive structure; it is licensed only when the carrier's classifier admits the algebraic equational fields.

The concrete history monoid instance extends the unary-history semigroup of chapter 79. It reuses the same append carrier, append associativity, and classifier route, and adds the empty-history unit rows that witness the left and right identity laws.

40.1 Carrier definition

Definition 40.1 (Monoid carrier)**.** *Use an abstract setup structure with carrier C, classifier $\sim_C$, operation $\cdot$, identity e, and law fields.*

40.2 Concrete history instance

Definition 40.2 (Concrete history monoid instance)**.** *The BEDC unary-history monoid instance takes $Carrier := \mathsf{UnaryHistory}$, $\mathsf{mul}(h, k) := \mathsf{append}(h, k)$, equivalently witnessed by $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$, $e := \mathsf{e}$, and $classifier := \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$.*

Lean4 ✓ `BEDC.Derived.MonoidUp.MonoidUp StdBridge`

Theorem 40.3 (Concrete history monoid laws)**.** *The concrete instance of Definition 40.2 has:*

1. *left unit of* e *for unary histories;*
2. *right unit of* e *for unary histories;*
3. *associativity of* append *up to* hsame*;*
4. *multiplication congruence under* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$*.*

Lean4 ✓ `BEDC.Derived.MonoidUp.concrete history monoid laws`

Proof. The left-unit and right-unit clauses are the BEDC continuation unit laws `BEDC.FKernel.Cont.cont_left_u` and `BEDC.FKernel.Cont.cont_right_unit`, restricted to unary histories.

Associativity is the continuation associativity theorem `BEDC.FKernel.Cont.cont_assoc_hsame`; equivalently, the two parenthesizations compute to the same BEDC history append tree up to hsame.

For congruence, the classifier supplies unary endpoint witnesses together with the relevant hsame transport. Unary append closure keeps both products in the carrier, and transport of hsame across the append arguments gives the required multiplication congruence. □

Theorem 40.4 (Monoid concrete history instance unlock surface)**.** *The carrier* UnaryHistory*, multiplication* append*, and unit* e *form the concrete* $\mathsf{Monoid}^{\uparrow}$ *instance consumed by downstream algebra schemas. Associativity is supplied by append associativity, the left and right unit laws are supplied by empty-history append laws, the classifier is the existing unary append monoid classifier, and every exported row is visible through* BHist*,* hsame*,* Cont*, and* Pkg*.*

Proof. Use the concrete instance of Definition 40.2 for the carrier, multiplication, unit, and classifier. The law package is exactly Definition 40.3: its unit clauses give the empty-history left and right unit laws, its associativity clause gives append associativity up to hsame, and its congruence clause keeps multiplication compatible with the unary append classifier.

The Lean-side names `BEDC.Derived.MonoidUp.unary_append_monoid_semantic_name_certificate` and `BEDC.Derived.MonoidUp.concrete_history_monoid_laws` are the intended formal anchors for this concrete unlock surface. No abstract carrier parameter, abstract classifier parameter, or transport between unrelated monoid presentations is introduced; downstream schemas consume the displayed UnaryHistory append instance directly. □

Theorem 40.5 (Unary Cont unit replay for the concrete monoid)**.** *For every unary history h, multiplication by* e *in the concrete* $\mathsf{Monoid}^{\uparrow}$ *instance of Definition 40.2 is read through the kernel continuation rows*

$$\mathsf{Cont}(\mathsf{e}, h, h)$$

and

$$\mathsf{Cont}(h, \mathsf{e}, h).$$

Thus the left and right unit laws of the concrete $\mathsf{Monoid}^{\uparrow}$ *packet are displayed* Cont *rows of the unary-history kernel, not abstract operator laws detached from the carrier.*

Proof. Begin with the concrete instance of Definition 40.2. Its multiplication is append, its unit is e, and its carrier is UnaryHistory.

Apply Definition 40.3. The first two clauses are the left and right unit laws for e on unary histories, and the proof identifies them with `BEDC.FKernel.Cont.cont_left_unit` and `BEDC.FKernel.Cont.cont_righ`

Finally use Definition 40.4. Downstream algebra reads the same displayed carrier, multiplication, unit, classifier, BHist, hsame, Cont, and Pkg rows, so the unit laws remain kernel continuation rows rather than external algebraic fields. □

Theorem 40.6 (Unary Cont concrete monoid instance)**.** *The displayed carrier* UnaryHistory*, operation* append *witnessed by* Cont*, unit* e*, and classifier* MonoidHistoryClassifier(UnaryHistory) *form the concrete* $\mathsf{Monoid}^{\uparrow}$ *instance consumed by downstream algebra schemas. Its multiplication rows are the unary append continuation rows, its unit rows are the two* Cont *replays through* e*, and its associativity row is the append associativity route on* BHist*.*

Proof. Use the concrete carrier, operation, unit, and classifier of Definition 40.2. The theorem fixes the same UnaryHistory surface, so the schema consumer sees append, e, and MonoidHistoryClassifier(UnaryHistory) directly.

Apply Definition 40.3. Its associativity clause supplies append associativity, its unit clauses supply the left and right e laws, and its classifier congruence keeps the operation inside the displayed unary-history classifier.

Finally apply Definition 40.4 and Definition 40.5. The unlock surface identifies the downstream $\mathsf{Monoid}^{\uparrow}$ packet, while the unit replay theorem pins the unit rows to concrete Cont data. No abstract carrier/classifier transport is introduced. □

Theorem 40.7 (Unary Cont cancellation route for the concrete monoid)**.** *Every downstream cancellation-facing consumer of the concrete unary append monoid must read the same* UnaryHistory *carrier, append and* Cont *multiplication rows, empty-unit rows, classifier congruence, and* hsame *transport before adding inverse data in* $\mathsf{Group}^{\uparrow}$*. The route is a* $BHist$*-anchored dependency from* $\mathsf{Monoid}^{\uparrow}$ *to later group obligations; it does not assert group cancellation for* $\mathsf{Monoid}^{\uparrow}$ *alone.*

Proof. Start with the concrete history instance of Definition 40.2 and the concrete history monoid laws of Definition 40.3. These fix the carrier, append multiplication, empty unit, associativity, unit rows, and classifier congruence on the same unary-history surface.

Apply the unary Cont unit replay and concrete instance route, Definition 40.5 and Definition 40.6. They keep multiplication and unit reads attached to displayed continuation rows and hsame transport before any downstream schema may ask for inverse data.

Thus a $\mathsf{Group}^{\uparrow}$ consumer can cite this route as the monoid-side input, but the inverse and cancellation rows must be supplied by the later group packet rather than by $\mathsf{Monoid}^{\uparrow}$ itself. □

The $\mathsf{Monoid}^{\uparrow}$ schema is instantiated by this concrete $BHist$ carrier and operation, so subsequent certificates can refer to this instance rather than only to abstract operator fields.

Theorem 40.8 (Unary append monoid commutative classifier)**.** *For unary histories* a *and* b*, the concrete monoid classifier identifies the two append products:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(a,b),\mathsf{append}(b,a)).$$

Proof. Unary append commutativity gives $\mathsf{append}(a,b)\mathsf{hsameappend}(b,a)$, and unary append closure supplies the two endpoint carrier witnesses required by MonoidHistoryClassifier(UnaryHistory). □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid commutative classifier`

40.3 The certificate

Definition 40.9 (Monoid source specification)**.** *The source is the carrier* C *with its classifier and operation data.*

Definition 40.10 (Monoid pattern specification)**.** *The pattern is closed expression formation from* e*, carrier witnesses, and binary multiplication.*

Definition 40.11 (Monoid classifier specification)**.** *The classifier is the supplied carrier sameness relation, required to respect multiplication.*

Lean4 (definition) `BEDC.Derived.MonoidUp.MonoidHistoryClassifier`

Theorem 40.12 (Monoid history classifier unary diagonal induction)**.** *If a predicate P holds at the empty diagonal* (e, e) *and is preserved by simultaneous unary extension on a diagonal history, then every* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, k)$ *witness supports* $P(h, k)$*.*

Proof. Unpack the classifier into unary witnesses for h and k plus $h\mathsf{hsame}k$. The history sameness component identifies the two histories, reducing the claim to the diagonal, where unary-history induction applies. □

Lean4 ✓ `BEDC.Derived.MonoidUp.MonoidHistoryClassifier unary diagonal induction`

Definition 40.13 (Monoid stability certificate)**.** *The stability certificate consists of:*

1. *classifier reflexivity*
2. *classifier transitivity*
3. *associativity under classifier*
4. *left identity law*
5. *right identity law*
6. *multiplication congruence in both arguments*

Lean4 ✓ `BEDC.Derived.MonoidUp.monoid stability certificate fields`

Theorem 40.14 (Monoid identity uniqueness)**.** *For BEDC history multiplication, if e is a left identity and e' is a right identity under* hsame*, then* $e\mathsf{hsame}e'$*.*

Proof. Use the right-identity law for e' at e to identify e with $e \cdot e'$, then use the left-identity law for e at e'. □

Lean4 ✓ `BEDC.Derived.MonoidUp.monoid identity unique`

Theorem 40.15 (Monoid history classifier identity uniqueness)**.** *For a BEDC history-level monoid classifier over carrier C, if $e, e' \in C$, e acts as a left identity and e' acts as a right identity under the monoid classifier, then the classifier identifies e and e'.*

Proof. The right-identity field at e gives $e \cdot e'\mathsf{hsame}e$, while the left-identity field at e' gives $e \cdot e'\mathsf{hsame}e'$. Symmetry and transitivity of hsame identify e and e', and the carrier witnesses for e and e' supply the classifier endpoints. □

Lean4 ✓ `BEDC.Derived.MonoidUp.MonoidHistoryClassifier identity unique`

Theorem 40.16 (Unary append monoid left identity is empty)**.** *For the concrete unary-history append monoid, any unary history e whose append action is a left identity under* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$ *is classified with* e*:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(e, \mathsf{e}).$$

Proof. Apply classifier identity uniqueness with e as the left identity and e as the right identity. The right-identity classifier for e follows from unary closure of append, $\mathsf{UnaryHistory}(\mathsf{e})$, and the BEDC append law $\mathsf{append}(h, \mathsf{e}) = h$. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid left identity empty`

Theorem 40.17 (Unary append monoid right identity is empty)**.** *For the concrete unary-history append monoid, any unary history e whose append action is a right identity under* MonoidHistoryClassifier(UnaryHist *is classified with* e*:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(e, \mathsf{e}).$$

Proof. Apply classifier identity uniqueness with e as the left identity and e as the right identity. The left-identity classifier for e follows from unary closure of append, UnaryHistory(e), and the BEDC append law $\mathsf{append}(\mathsf{e}, h) = h$. Symmetry of the resulting hsame component gives the displayed classifier orientation. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid right identity empty`

Theorem 40.18 (Unary append monoid identity is empty iff)**.** *For the concrete unary-history append monoid, a unary history e is both a left and right identity under* MonoidHistoryClassifier(UnaryHistory) *exactly when it is classified with* e*.*

Proof. The forward implication uses the left-identity uniqueness result for the concrete empty right identity. Conversely, a classifier witness $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(e, \mathsf{e})$ gives $e$hsamee. Replacing e by e reduces the two identity classifier families to the BEDC append laws $\mathsf{append}(\mathsf{e}, h) = h$ and $\mathsf{append}(h, \mathsf{e}) = h$, with unary closure supplying the endpoint carrier fields. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid identity empty iff`

Theorem 40.19 (Unary append monoid idempotent iff empty)**.** *For every unary history e, the concrete unary append monoid classifies the duplicated append factor as e exactly when e is classified with* e*:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(e, e), e) \quad \textit{if and only if} \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(e, \mathsf{e}).$$

Here the BEDC source is BHist *with the carrier* UnaryHistory *and operation* append*; the classifier is* MonoidHistoryClassifier(UnaryHistory)*. The ledger records the duplicated factor e and the empty right unit, stability is supplied by common-context congruence and cancellation, exactness is the displayed iff, and the usual finite unary-word monoid is the standard bridge.*

Proof. For the forward direction, rewrite the right endpoint e as $\mathsf{append}(e, \mathsf{e})$ by the empty right-unit law. The hypothesis then classifies two histories with common left context e and common right context e, with middle endpoints e and e. Apply Theorem 40.28 to obtain MonoidHistoryClassifier(UnaryHistory)(e, e

For the reverse direction, start from the classifier witness $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(e, \mathsf{e})$. Apply Theorem 40.26 with left context e and right context e to classify the contextual pair $\mathsf{append}(e, \mathsf{append}(e, \mathsf{e}))$ and $\mathsf{append}(e, \mathsf{append}(\mathsf{e}, \mathsf{e}))$. The empty right-unit and left-unit laws identify these endpoints with $\mathsf{append}(e, e)$ and e, giving the displayed classifier. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid idempotent empty iff`

Theorem 40.20 (Unary append monoid left-inverse field collapses to singleton)**.** *For every operation* inv : BHist → BHist*, a global left-inverse field on the full unary-history append carrier,*

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(\mathsf{inv}(h), h), \mathsf{e}),$$

forces every unary history to be classified with the singleton unit:

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

Proof. Fix a unary history h and specialize the left-inverse field at h. The unit-product factor exactness theorem decomposes the classifier for $\mathsf{append}(\mathsf{inv}(h), h)$ into singleton classifiers for the two append factors. The right factor is exactly the displayed classifier of h with e. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid left inverse field collapses to singleton`

Theorem 40.21 (Unary append monoid one-sided inverse obstruction)**.** *For every operation* $\mathsf{inv} : \mathsf{BHist} \to \mathsf{BHist}$*, the concrete unary-history append monoid supports neither a global right-inverse family*

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, \mathsf{inv}(h)), \mathsf{e}),$$

nor a global left-inverse family

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(\mathsf{inv}(h), h), \mathsf{e}).$$

The ten-field fit is: traditional target group-upgrade obstruction for the unary append monoid; BEDC source BHist*,* $\mathsf{UnaryHistory}$*,* append*, and* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$*; carrier* $\mathsf{UnaryHistory}$*; classifier the concrete monoid classifier; stability from unary append closure and empty-result inversion; ledger the attempted inverse endpoint; core theorem this block; exactness the contradiction from the visible unary generator; standard bridge finite unary-word monoid has no global inverse; status* $\mathsf{ProofTarget}$*.*

Proof. For the right-inverse family, take the visible unary generator $u = \mathsf{E1}(\mathsf{e})$. Unary closure gives $\mathsf{UnaryHistory}(u)$, so the assumed family classifies $\mathsf{append}(u, \mathsf{inv}(u))$ with e. Unfolding the concrete classifier exposes $\mathsf{append}(u, \mathsf{inv}(u))\mathsf{hsamee}$.

Empty-result inversion for append then forces the left factor to satisfy $u\mathsf{hsamee}$. This contradicts the BEDC no-confusion theorem separating the $\mathsf{E1}$ constructor from e.

The left-inverse family is symmetric in the obstruction. Specializing it at the same u exposes $\mathsf{append}(\mathsf{inv}(u), u)\mathsf{hsamee}$. Empty-result inversion now forces the right factor u to be hsame to e, and the same no-confusion contradiction follows. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid one sided inverse law absurd`

Theorem 40.22 (History continuation monoid laws)**.** *The BEDC history continuation relation has empty history as a left and right unit, and the two parenthesizations of a triple continuation have the same resulting history.*

Lean4 ✓ `BEDC.Derived.MonoidUp.history continuation monoid laws`

Proof. The unit fields are the empty-history continuation constructors. For associativity, each continuation witness unfolds to history append, and the result follows from associativity of BEDC history append. □

Theorem 40.23 (Nonempty continuation suffix changes the source)**.** *For any source history* h*, no continuation by a suffix of the form* $\mathsf{E0}(k)$ *or* $\mathsf{E1}(k)$ *can have result* h *itself.*

Lean4 ✓ `BEDC.Derived.MonoidUp.history continuation nonempty suffix source absurd`

Proof. Right-unit uniqueness for continuation forces any suffix whose result is the source to be e. BEDC history no-confusion separates $\mathsf{E0}(k)$ and $\mathsf{E1}(k)$ from e. □

Theorem 40.24 (Nonempty continuation prefix changes the target)**.** *For any target history k, no continuation with source of the form* $\mathsf{E0}(h)$ *or* $\mathsf{E1}(h)$ *can have result k itself.*

Proof. Left-unit uniqueness for continuation forces any source whose continuation leaves the target unchanged to be e. BEDC history no-confusion separates $\mathsf{E0}(h)$ and $\mathsf{E1}(h)$ from e. □

Lean4 ✓ `BEDC.Derived.MonoidUp.history continuation nonempty prefix target absurd`

Theorem 40.25 (Unary append monoid semantic name certificate)**.** *The unary-history carrier, empty history, BEDC history append, and the history-sameness classifier instantiate the semantic naming certificate for* $\mathsf{Monoid}^{\uparrow}$*. The certificate exports the empty-left, empty-right, associativity, and append-congruence fields under the classifier* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$*.*

Proof. Unary history is inhabited by empty history and is closed under BEDC append. The empty-history laws and associativity are the primitive append laws for histories. Congruence follows by transporting both append arguments across hsame, leaving the same appended history on both sides. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid semantic name certificate`

Theorem 40.26 (Unary append monoid classifier appends common context)**.** *For unary histories l, r, if the monoid classifier identifies a with b, then it identifies the two contextual histories*

$$l(ar)$$

and

$$l(br).$$

Proof. The classifier supplies unary witnesses for a and b together with $a\mathsf{hsame}b$. Unary append closure builds the two contextual endpoint carriers. After the endpoint equality is exposed, reflexivity of hsame identifies the two appended histories. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid classifier append context`

Theorem 40.27 (Unary append monoid classifier preserves context)**.** *For unary histories l, r, a, b, if the monoid classifier identifies a with b, then it identifies the common-context appends*

$$l(ar)$$

and

$$l(br).$$

Proof. The classifier gives unary endpoint witnesses for a and b and an hsame witness between them. Unary histories are closed under BEDC append, so appending the common right and left contexts supplies the endpoint carriers. The hsame witness identifies the middle history, leaving equal contextual appends. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid classifier context congr`

Theorem 40.28 (Unary append monoid classifier cancels common context)**.** *For unary histories l, r, a, b, if the monoid classifier identifies*

$$l(ar)$$

with

$$l(br),$$

then it identifies a with b.

Proof. The classifier supplies the endpoint unary witnesses and a BEDC history equality between the two contextual appends. Left append cancellation removes the common left context l, and right append cancellation removes the common suffix r. The given unary witnesses for a and b supply the classifier endpoints. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid classifier cancel context`

Theorem 40.29 (Unary append monoid classifier context equivalence)**.** *For unary histories l, r, a, b, the monoid classifier identifies*

$$l(ar)$$

with

$$l(br)$$

exactly when it identifies a with b.

Proof. The forward direction is common-context cancellation: left append cancellation removes l, and right append cancellation removes r. The reverse direction is common-context congruence using unary append closure on both endpoints. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid classifier context iff`

Theorem 40.30 (Unary append monoid left factor empty iff)**.** *For unary histories $h, k : \mathsf{BHist}$, classifying the appended history $\mathsf{append}(h, k)$ with its left factor h is equivalent to classifying the right factor k with e:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, k), h) \iff \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(k, \mathsf{e}).$$

Proof. For the forward direction, rewrite the right endpoint as $\mathsf{append}(h, \mathsf{e})$ by the append right-unit law and cancel the common left factor h. Conversely, a classifier witness for k against e rewrites the appended endpoint to $\mathsf{append}(h, \mathsf{e})$, and the append right-unit law identifies this with h. Unary append closure supplies the source carrier endpoint. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid left factor empty iff`

Theorem 40.31 (Unary append monoid right factor empty iff)**.** *For unary histories $h, k : \mathsf{BHist}$, classifying the appended history $\mathsf{append}(h, k)$ with its right factor k is equivalent to classifying the left factor h with e:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, k), k) \iff \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

The ten-field fit is: traditional target right factor-to-unit exactness in the unary append monoid; BEDC source $\mathsf{Monoid}^{\uparrow}$ over $\mathsf{UnaryHistory}$; carrier $\mathsf{UnaryHistory}$; classifier $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$; stability from the commutative classifier, context equivalence, and left-factor exactness; ledger the hidden left factor; core theorem this block; exactness the displayed iff; standard bridge finite unary-word monoid; status $\mathsf{ProofTarget}$.

Proof. For the forward direction, unary append commutativity first classifies $\mathsf{append}(k, h)$ with $\mathsf{append}(h, k)$. Classifier transitivity then composes this classification with the assumed classification of $\mathsf{append}(h, k)$ with k, producing a classification of $\mathsf{append}(k, h)$ with k.

Apply Definition 40.30 with left factor k and right factor h. The resulting exactness classifies h with e.

For the reverse direction, start from the classification of h with e. The same left-factor exactness theorem, again applied to k and h, gives a classification of $\mathsf{append}(k, h)$ with k.

Unary append commutativity classifies $\mathsf{append}(h, k)$ with $\mathsf{append}(k, h)$. Classifier transitivity orients the endpoint as a classification of $\mathsf{append}(h, k)$ with k. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid right factor empty iff`

Theorem 40.32 (Unary append monoid classifier swaps unary context)**.** *For unary histories $left$, $right$, and a, the concrete monoid classifier identifies the two histories obtained by moving the two unary contexts around the middle payload:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(left, \mathsf{append}(a, right)), \mathsf{append}(right, \mathsf{append}(a, left))).$$

Proof. Unary append closure supplies the endpoint carrier witnesses. Associativity first exposes the two context factors, unary append commutativity swaps a with $right$, swaps $left$ with $right$, and finally swaps $left$ with a inside the right context. The resulting equality is the hsame component of the monoid classifier. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid classifier context swap`

Definition 40.33 (Monoid ledger policy)**.** *Record the carrier witnesses and operation tree behind each monoid expression.*

Definition 40.34 ($\mathsf{Monoid}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Monoid}^{\uparrow}$*.*

Theorem 40.35 (Concrete $\mathsf{Monoid}^{\uparrow}$ naming certificate)**.** *Any setup satisfying the monoid fields forms a valid NameCert for* $\mathsf{Monoid}^{\uparrow}(C)$*.*

Proof. The identity element inhabits the carrier. The monoid classifier records source and target carrier witnesses together with the BEDC history equality witness, giving the certificate core by hsame reflexivity, symmetry, transitivity, and carrier projection. Closure under multiplication supplies carrier witnesses for the exported identity, associativity, and multiplication-congruence fields. □

Lean4 ✓ `BEDC.Derived.MonoidUp.monoid history semantic name certificate`

Primitive boundary. $\mathsf{Monoid}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Definition 40.36 (Opposite monoid certificate data)**.** *Let a* $\mathsf{Monoid}^{\uparrow}$ *certificate be presented by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, multiplication $\cdot_C$, and identity e_C. Define the reversed multiplication by*

$$a \cdot_C^{\mathrm{op}} b := b \cdot_C a.$$

The opposite monoid data C^{op} have the same carrier predicate $\mathcal{C}$, the same classifier $\sim_C$, the same identity e_C, and multiplication $\cdot_C^{\mathrm{op}}$.

Theorem 40.37 (Opposite monoid certificate)**.** *If $(\mathcal{C}, \sim_C, \cdot_C, e_C)$ satisfies the* $\mathsf{Monoid}^{\uparrow}$ *certificate obligations of Definition 40.34, then the data of Definition 40.36 also satisfy the* $\mathsf{Monoid}^{\uparrow}$ *certificate obligations. Thus the same carrier, identity, and classifier carry a monoid certificate with multiplication $\cdot_C^{\mathrm{op}}$.*

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid opposite semantic name certificate`

Proof. Unpack the monoid obligations through Definition 40.34. The carrier row of Definition 40.1 is unchanged. The carrier-and-classifier core is the same NameCert data, so classifier reflexivity, symmetry, transitivity, and carrier transport are copied from Definition 9.7. The source row of Definition 40.9 is the same carrier and classifier equipped with the reversed multiplication symbol.

Closure for the reversed multiplication follows from original closure with the two inputs exchanged: if $\mathcal{C}(a)$ and $\mathcal{C}(b)$, then

$$a \cdot_C^{\mathrm{op}} b = b \cdot_C a$$

is carried because b and a are carried.

The identity rows are exchanged. The opposite left identity law is

$$e_C \cdot_C^{\mathrm{op}} a = a \cdot_C e_C \sim_C a,$$

which is the original right identity row. The opposite right identity law is

$$a \cdot_C^{\mathrm{op}} e_C = e_C \cdot_C a \sim_C a,$$

which is the original left identity row.

For associativity, the two opposite bracketings unfold to

$$(a \cdot_C^{\mathrm{op}} b) \cdot_C^{\mathrm{op}} c = c \cdot_C (b \cdot_C a)$$

and

$$a \cdot_C^{\mathrm{op}} (b \cdot_C^{\mathrm{op}} c) = (c \cdot_C b) \cdot_C a.$$

The associativity row in Definition 40.13, applied to the original multiplication at the triple (c, b, a), supplies

$$(c \cdot_C b) \cdot_C a \sim_C c \cdot_C (b \cdot_C a).$$

Classifier symmetry gives the required opposite orientation

$$c \cdot_C (b \cdot_C a) \sim_C (c \cdot_C b) \cdot_C a.$$

Multiplication congruence is also inherited with inputs exchanged. Given $a \sim_C a'$ and $b \sim_C b'$, the required comparison for the opposite product is

$$a \cdot_C^{\mathrm{op}} b = b \cdot_C a \sim_C b' \cdot_C a' = a' \cdot_C^{\mathrm{op}} b',$$

which is the original multiplication congruence applied to $b \sim_C b'$ and $a \sim_C a'$.

The classifier row of Definition 40.11 is unchanged because endpoint comparison is still $\sim_C$; the congruence paragraph is exactly the respect-for-multiplication field for the reversed operation. The pattern row of Definition 40.10 remains closed expression formation from e_C, carried endpoints, and binary multiplication, with each reversed multiplication node read as an original multiplication node whose children are exchanged. The ledger row of Definition 40.33 records the same expression tree with that exchanged child order at every reversed multiplication node. Hence the source-to-pattern and source-to-ledger soundness fields are the original operation-tree soundness checks read through the definition of $\cdot_C^{\mathrm{op}}$.

These carrier, source, pattern, classifier, stability, and ledger rows are precisely the five-field naming-certificate data required by Definition 9.6 and the monoid obligations of Definition 40.34. Therefore C^{op} carries a $\mathsf{Monoid}^{\uparrow}$ certificate. □

Theorem 40.38 (Double opposite monoid certificate data identity)**.** *Let D be a* $\mathsf{Monoid}^{\uparrow}$ *certificate satisfying the obligations of Definition 40.34. Present its monoid data by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, multiplication $\cdot_C$, and identity element e_C as in Definition 40.36. Let*

$$D^{\mathrm{opop}} = (D^{\mathrm{op}})^{\mathrm{op}}$$

be obtained by applying Definition 40.36 twice. Then D^{opop} is a $\mathsf{Monoid}^{\uparrow}$ *certificate, and its carrier predicate, classifier, multiplication, and identity element are componentwise identical to those of D:*

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}(a), \quad a \sim_C^{\mathrm{opop}} b = a \sim_C b,$$
$$a \cdot_C^{\mathrm{opop}} b = a \cdot_C b, \qquad e_C^{\mathrm{opop}} = e_C.$$

Proof. Because D satisfies the $\mathsf{Monoid}^{\uparrow}$ obligations, Definition 40.37 makes D^{op} a certified $\mathsf{Monoid}^{\uparrow}$ instance. Applying the same theorem to that certified opposite data makes D^{opop} a certified $\mathsf{Monoid}^{\uparrow}$ instance.

It remains to compare the displayed data. The carrier predicate, classifier, and identity element are fixed by one use of Definition 40.36. Hence two unfoldings give

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}^{\mathrm{op}}(a) = \mathcal{C}(a),$$

$$a \sim_C^{\mathrm{opop}} b = a \sim_C^{\mathrm{op}} b = a \sim_C b,$$

and

$$e_C^{\mathrm{opop}} = e_C^{\mathrm{op}} = e_C.$$

For multiplication, the first opposite exchanges the two inputs and the second opposite exchanges them back:

$$a \cdot_C^{\mathrm{opop}} b = b \cdot_C^{\mathrm{op}} a = a \cdot_C b.$$

Thus the raw multiplication field of the double opposite certificate is exactly the original multiplication field.

The source row of Definition 40.9 uses only the carrier, classifier, and multiplication just identified. The pattern row of Definition 40.10 forms expressions from the same identity element and the same restored multiplication. The classifier row of Definition 40.11 is the same endpoint relation, and multiplication respect is typed over the restored multiplication. The ledger policy of Definition 40.33 records operation trees; one opposite reading swaps the two children at each multiplication node, and the second reading swaps the same children back. Therefore the displayed component identities identify the double opposite monoid certificate data with the monoid data of D. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid double opposite data identity`

Closure status for $\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$**.**

Constructive story: The unary spine already carries a monoid operation: put one unary history after another by kernel continuation, and unary closure says the result is still on the spine while e acts as the empty tail on both sides. Two carried elements are the same just when the underlying histories are history-equal inside the unary carrier, so the operation never leaves the history surface for an imported algebraic set. The source is the unary predicate itself, the pattern records binary append, the two unit laws with e, and associativity of three continued spines, and the ledger keeps each visible monoid element aligned with the raw unary history that realizes it. Because history-equality is an equivalence and unary histories are closed under continuation, e supplies the carrier witness, the classifier is reflexive, symmetric, and transitive, and recognized representatives remain admitted. A unary history h lies in the signature gap of a $\mathsf{Monoid}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; concretely, h is unary and history-equal to the spine encoded by p. The same package also exposes the inverse obstruction, unique factor readback up to history-equality, the opposite monoid, and the

double-opposite identity as consequences of continuation. The chapter registers this unary-append construction under the name $\mathsf{Monoid}^{\uparrow}$; the monoid laws are theorems about the existing kernel continuation, with no imported algebraic carrier, inverse primitive, or host monoid structure introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature unary-history monoid package binds the concrete history monoid laws, semantic NameCert, inverse-obstruction and factor-exactness rows, opposite certificate, and double-opposite data identity for the append/Cont surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.MonoidUp.monoid\_history\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: The chapter does not close arbitrary host monoids, homomorphism theory, category-of-monoids structure, or bridgeChecked verification.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 41

A Concrete Naming Certificate for $\mathsf{Group}^{\uparrow}$

The group interface $\mathsf{Group}^{\uparrow}$ extends chapter 40 with a unary inverse operation $^{-1} : C \to C$ satisfying left and right inverse laws under the carrier's classifier. It is not a primitive structure; it is licensed only by composing the monoid certificate with the inverse field.

Remark 41.1 (Roadmap chapter)**.** *This chapter records the licensing target. Lean markers identify the algebraic fields and derived laws that are formalized for the certificate.*

41.1 Carrier definition

Definition 41.2 (Group carrier)**.** *Use a monoid setup extended with inverse operation a^{-1}.*

The group carrier consumes the $\mathsf{Monoid}^{\uparrow}$ multiplication, associativity, and unit rows from chapter 40 before adding inverse rows and inverse-ledger obligations.

41.2 Concrete singleton history group instance

Definition 41.3 (Concrete singleton history group instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define the concrete history operations and classifier by*

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{inv}(h) := \mathsf{e}, \qquad e := \mathsf{e},$$

and

$$R_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Theorem 41.4 (Group singleton history SemanticNameCert)**.** *The carrier A_{e}, classifier R_{e}, multiplication* mul*, unit e, and inverse operation* inv *of Definition 41.3 supply the singleton-history* SemanticNameCert *for* $\mathsf{Group}^{\uparrow}$ *over* BHist*, together with the concrete multiplication-closure, inverse-closure, left-unit, right-unit, left-inverse, and right-inverse rows captured by* BEDC.Derived.GroupUp.GroupSingletonI

Proof. The SemanticNameCert component is packaged by the carrier-aware classifier R_{e}. Its inhabited carrier is witnessed by $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Reflexivity, symmetry, and transitivity are inherited from hsame after unpacking and repacking the carrier endpoints required by R_{e}.

Carrier transport follows by composing singleton endpoint witnesses through e: if a source endpoint and its classifier partner are both related by R_{e}, then the transported endpoint remains in A_{e} by transitivity through their displayed witnesses to e.

Carrier inhabitance is the reflexive endpoint witness $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, which is exactly the singleton carrier condition from Definition 36.169 at $a_0 = \mathsf{e}$.

For multiplication closure, suppose $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. The two carrier facts identify h and k with e. The BEDC continuation unit laws used in Theorem 40.3 reduce $\mathsf{append}(h, k)$ to the same endpoint class as e, using symmetry and transitivity of hsame to compose the displayed witnesses.

Inverse closure is immediate because $\mathsf{inv}(h)$ is definitionally e. The singleton carrier witness is again $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

The left and right unit rows reduce $\mathsf{append}(\mathsf{e}, h)$ and $\mathsf{append}(h, \mathsf{e})$ to h by the continuation unit laws. The inverse rows use $\mathsf{inv}(h) = \mathsf{e}$, so the same unit laws and the carrier witness for h classify both products against $e = \mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonHistory laws`

Theorem 41.5 (Concrete singleton history group algebraic laws)**.** *The concrete singleton history group over* BHist *satisfies carrier inhabitance, multiplication closure, inverse closure, left and right unit laws, associativity up to* hsame*, multiplication congruence, inverse congruence, and left and right inverse laws.*

Proof. Carrier inhabitance, multiplication closure, inverse closure, and the unit rows are the corresponding concrete rows of Theorem 41.4.

Associativity is the continuation associativity theorem cited in Theorem 40.3: the histories $\mathsf{append}(\mathsf{append}(h, k), l)$ and $\mathsf{append}(h, \mathsf{append}(k, l))$ are hsame-equivalent. Closure of the two products follows from the multiplication-closure clause.

Multiplication congruence follows by transporting the two input hsame witnesses across append and then composing with the same continuation associativity and unit laws. Since all carrier elements are in the singleton class of e, the source and target products both lie in A_{e} and are related by hsame.

Inverse congruence is constant: both inverses are e, so the classifier witness is the pair of singleton carrier witnesses together with $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

For the inverse laws, $\mathsf{append}(\mathsf{inv}(h), h)$ is $\mathsf{append}(\mathsf{e}, h)$ and $\mathsf{append}(h, \mathsf{inv}(h))$ is $\mathsf{append}(h, \mathsf{e})$. The continuation unit laws reduce these products to h, and the carrier fact $A_{\mathsf{e}}(h)$ identifies h with $e = \mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.GroupUp.concrete singleton history group laws`

The $\mathsf{Group}^{\uparrow}$ schema is instantiated by this concrete BHist carrier, append operation, empty unit, and constant inverse, so certificate rows in this chapter are BHist-anchored rather than abstract operator fields.

Theorem 41.6 (Group singleton carrier append context empty)**.** *For histories $L, R, h : \mathsf{BHist}$, if L and R lie in the singleton group carrier A_{e}, then a two-sided append context lies in the same singleton carrier exactly when the middle history does:*

$$A_{\mathsf{e}}(L) \Rightarrow A_{\mathsf{e}}(R) \Rightarrow \big(A_{\mathsf{e}}(\mathsf{append}(L, \mathsf{append}(h, R))) \Longleftrightarrow A_{\mathsf{e}}(h)\big).$$

Proof. In the forward direction, unfold singleton carrier membership as equality to e up to hsame. Empty-result inversion for append first splits off the left context L, then splits the inner append, exposing the middle carrier witness $A_{\mathsf{e}}(h)$.

Conversely, combine the carrier witnesses for h and R to rebuild the inner append as a singleton carrier, then combine that result with the carrier witness for L to rebuild the displayed two-sided append context. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCarrier append context empty iff`

Theorem 41.7 (Group singleton classifier append context cancellation)**.** *For histories L, R, Q, S : BHist, if the displayed left contexts are in the singleton group carrier A_{e}, then appending those contexts to the two classifier endpoints preserves and reflects the singleton classifier:*

$$A_{\mathsf{e}}(L) \Rightarrow A_{\mathsf{e}}(R) \Rightarrow (R_{\mathsf{e}}(\mathsf{append}(L, Q), \mathsf{append}(R, S)) \Longleftrightarrow R_{\mathsf{e}}(Q, S)).$$

Proof. In the forward direction, the two appended endpoints are classified as singleton endpoints. Empty-result inversion for append splits each endpoint witness, giving $Q\mathsf{hsamee}$ and $S\mathsf{hsamee}$. These two carrier witnesses, together with transitivity through e, assemble $R_{\mathsf{e}}(Q, S)$.

Conversely, combine the supplied singleton witnesses for L and R with the endpoint carrier fields in $R_{\mathsf{e}}(Q, S)$. Empty-result construction for append gives singleton witnesses for $\mathsf{append}(L, Q)$ and $\mathsf{append}(R, S)$, and transitivity through e gives the classifier comparison between the two appended endpoints. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append context cancel iff`

Theorem 41.8 (Group singleton classifier right-context cancellation)**.** *For histories L, R, Q, S : BHist, if the displayed right contexts are in the singleton group carrier A_{e}, then appending those contexts as suffixes preserves and reflects the singleton classifier:*

$$A_{\mathsf{e}}(L) \Rightarrow A_{\mathsf{e}}(R) \Rightarrow (R_{\mathsf{e}}(\mathsf{append}(Q, L), \mathsf{append}(S, R)) \Longleftrightarrow R_{\mathsf{e}}(Q, S)).$$

The ten-field fit names the traditional target as terminal-group cancellation by a classified suffix; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ carrier; carrier A_{e}; classifier R_{e}; stability as singleton carrier transport and `append_eq_empty_iff`; ledger the hidden singleton suffix endpoints; core theorem list Theorem 41.7 and `append_eq_empty_iff`; exactness predicate the displayed iff; standard bridge cancellation in the terminal group; and status ProofTarget.

Proof. In the forward direction, singleton classification of the two appended histories and empty-result inversion for append split the suffix endpoints. The resulting singleton witnesses for Q and S, together with transitivity through e, package $R_{\mathsf{e}}(Q, S)$.

Conversely, combine $R_{\mathsf{e}}(Q, S)$ with the carrier witnesses for L and R. Empty-result construction for append gives singleton witnesses for $\mathsf{append}(Q, L)$ and $\mathsf{append}(S, R)$, and transitivity through e gives their classifier comparison. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append suffix cancel iff`

Theorem 41.9 (Group singleton classifier two-sided context cancellation)**.** *For histories L, R, L', R', Q, S : BHist, if $A_{\mathsf{e}}(L)$, $A_{\mathsf{e}}(R)$, $A_{\mathsf{e}}(L')$, and $A_{\mathsf{e}}(R')$, then*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, Q), L'), \mathsf{append}(\mathsf{append}(R, S), R')) \Longleftrightarrow R_{\mathsf{e}}(Q, S)$$

The ten-field fit names the traditional target as terminal-group two-sided contextual cancellation; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ carrier; carrier A_{e}; classifier R_{e}; stability as left and right context cancellation; ledger as hidden singleton prefix and suffix contexts; core theorem list Theorem 41.7 and Theorem 41.8; exactness predicate as the displayed iff; standard bridge as cancellation of terminal-group contexts; and status ProofTarget.

Proof. For the forward direction, apply Theorem 41.8 to cancel the singleton suffixes L' and R'. This reduces the classifier statement to

$$R_{\mathsf{e}}(\mathsf{append}(L, Q), \mathsf{append}(R, S)).$$

Then apply Theorem 41.7 to cancel the singleton prefixes L and R, producing $R_{\mathsf{e}}(Q, S)$.

For the reverse direction, start from $R_{\mathsf{e}}(Q, S)$ and use Theorem 41.7 in the constructive direction to attach the singleton prefixes L and R. Then use Theorem 41.8 in the constructive direction to attach the singleton suffixes L' and R', giving exactly the displayed two-sided classifier relation. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier right context cancel iff`

Theorem 41.10 (Group singleton classifier product-unit split)**.** *For histories p, q : BHist, the concrete singleton group classifier satisfies*

$$R_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{e}) \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).$$

The ten-field fit names the traditional target as a product equal to the unit in the one-element group; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ instance over BHist; carrier A_{e}; classifier R_{e}; stability as append empty-result construction and inversion; ledger factors p, q; core theorem list Theorem 41.5 together with the BEDC append empty-result inversion; exactness predicate the displayed iff; standard bridge the terminal group unit-product split; and status $\mathsf{ProofTarget}$.

Proof. Unfold R_{e} and A_{e}. In the forward direction, the classifier supplies $A_{\mathsf{e}}(\mathsf{append}(p, q))$, so $\mathsf{hsame}(\mathsf{append}(p, q), \mathsf{e})$ holds. Append empty-result inversion splits this witness into $A_{\mathsf{e}}(p)$ and $A_{\mathsf{e}}(q)$, giving the displayed conjunction.

Conversely, assume $A_{\mathsf{e}}(p)$ and $A_{\mathsf{e}}(q)$. Append empty-result construction gives $A_{\mathsf{e}}(\mathsf{append}(p, q))$, while reflexivity gives $A_{\mathsf{e}}(\mathsf{e})$. Symmetry and transitivity of hsame then package the endpoint comparison required by $R_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append unit split iff`

Theorem 41.11 (Group singleton contextual product-unit split)**.** *For histories L, R, p, q : BHist, if $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$, then*

$$R_{\mathsf{e}}(\mathsf{append}(L, \mathsf{append}(p, q)), \mathsf{append}(R, \mathsf{e})) \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).$$

The ten-field fit names the traditional target as terminal-group contextual unit-product splitting; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ carrier; carrier A_{e}; classifier R_{e}; stability as context cancellation plus append empty-result construction and inversion; ledger as the hidden singleton contexts and product factors; core theorem list Theorem 41.7 and Theorem 41.10; exactness predicate as the displayed iff; standard bridge as the terminal group unit-product split under context; and status $\mathsf{ProofTarget}$.

Proof. Apply Theorem 41.7 with $Q = \mathsf{append}(p, q)$ and $S = \mathsf{e}$. The carrier hypotheses $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$ strip the singleton left and right contexts, reducing the displayed classifier to

$$R_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{e}).$$

Apply Theorem 41.10 to this reduced classifier. It identifies the reduced classifier exactly with $A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q)$, giving both directions of the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append context unit split iff`

Theorem 41.12 (Group singleton classifier append product empty split)**.** *For histories* a, b, c, d : BHist*, the singleton group classifier relates the two appended products exactly when each append factor is carried by the singleton carrier:*

$$R_{\mathsf{e}}(\mathsf{append}(a, b), \mathsf{append}(c, d)) \iff A_{\mathsf{e}}(a) \wedge A_{\mathsf{e}}(b) \wedge A_{\mathsf{e}}(c) \wedge A_{\mathsf{e}}(d).$$

Proof. Unfold R_{e}. In the forward direction, the two carrier fields say both appended products are empty, and append empty-result inversion splits them into the four displayed singleton carrier witnesses. Conversely, the four witnesses rebuild the two empty appended products; symmetry and transitivity of hsame package the singleton classifier relation between them. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append product empty split iff`

Theorem 41.13 (Group singleton carrier visible-head append exclusion)**.** *For histories* p, q : BHist*, neither* $\mathsf{append}(\mathsf{e0}(p), q)$ *nor* $\mathsf{append}(\mathsf{e1}(p), q)$ *lies in the singleton group carrier.*

Proof. In either branch, the singleton carrier identifies the displayed appended history with e. Empty-result inversion for append then identifies the visible head $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$ with e. Constructor separation for generated histories rules out both cases. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCarrier append visible head absurd`

Theorem 41.14 (Group singleton carrier visible-tail append exclusion)**.** *For histories* p, q : BHist*, neither* $\mathsf{append}(p, \mathsf{e0}(q))$ *nor* $\mathsf{append}(p, \mathsf{e1}(q))$ *lies in the singleton group carrier.*

Proof. In either branch, the singleton carrier identifies the displayed appended history with e. Empty-result inversion for append then identifies the visible tail $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$ with e. Constructor separation for generated histories rules out both cases. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCarrier append visible tail absurd`

Theorem 41.15 (Group singleton classifier append commuted factors)**.** *For histories* h, h_2, k, k_2 : BHist*, if the singleton group classifier relates the two appended endpoints*

$$R_{\mathsf{e}}(\mathsf{append}(h, k), \mathsf{append}(h_2, k_2)),$$

then it also relates the endpoints with the two append factors exchanged:

$$R_{\mathsf{e}}(\mathsf{append}(k, h), \mathsf{append}(k_2, h_2)).$$

Proof. The classifier gives singleton carrier witnesses for $\mathsf{append}(h, k)$ and $\mathsf{append}(h_2, k_2)$. Empty-result inversion for append splits each witness into singleton carrier witnesses for the two factors. Exchanging the factors and applying empty-result construction rebuilds singleton carrier witnesses for $\mathsf{append}(k, h)$ and $\mathsf{append}(k_2, h_2)$. Since both rebuilt endpoints are classified through e, transitivity and symmetry of hsame package the displayed singleton classifier. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append comm congr`

Theorem 41.16 (Group singleton classifier continuation visible result exclusion)**.** *For histories* P, Q, r : BHist*, if the singleton group classifier relates* P *and* Q*, then no continuation from* P *through* Q *can have visible result* $\mathsf{E0}(r)$ *or* $\mathsf{E1}(r)$*:*

$$R_{\mathsf{e}}(P, Q) \Rightarrow (\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

Proof. The classifier exposes singleton carrier witnesses for P and Q. Transport the continuation result along these endpoint witnesses to the right-unit continuation at e, obtaining a singleton carrier witness for the displayed result. Constructor separation rules out both visible constructors as singleton-empty histories. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation visible result absurd`

Theorem 41.17 (Group singleton classifier continuation result-left iff)**.** *For histories P, Q, R : BHist with $\mathsf{Cont}(P, Q, R)$, the singleton group classifier relates the two continuation inputs exactly when it relates the continuation result to the left input:*

$$R_{\mathsf{e}}(P, Q) \iff R_{\mathsf{e}}(R, P).$$

Proof. If $R_{\mathsf{e}}(P, Q)$ holds, then both inputs are singleton-carried. Continuation result transport against the right-unit continuation at e makes R singleton-carried, and transitivity through e relates R to P. Conversely, if $R_{\mathsf{e}}(R, P)$ holds, then R is singleton-carried. Transport the displayed continuation to empty result and invert it to recover singleton carrier witnesses for P and Q; these witnesses classify the input pair. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation result left iff`

Theorem 41.18 (Group singleton inverse continuation left-unit iff)**.** *For histories x, r : BHist, if r is a continuation result of the singleton inverse of x followed by x, then classifying r against the unit endpoint is exactly the singleton carrier condition on x:*

$$\mathsf{Cont}(\mathsf{GroupSingletonInv}(x), x, r) \Rightarrow \big(R_{\mathsf{e}}(r, \mathsf{e}) \iff A_{\mathsf{e}}(x)\big).$$

Proof. Unfolding the continuation identifies r with $\mathsf{append}(\mathsf{GroupSingletonInv}(x), x)$. In the forward direction, the classifier says this appended result is singleton-carried, and empty-result inversion for append recovers the carrier witness for x. Conversely, the inverse endpoint is definitionally e; combining its reflexive carrier witness with the carrier witness for x constructs the singleton carrier witness for the appended result, which then classifies the result against e. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonInv continuation left unit iff`

Theorem 41.19 (Group singleton classifier continuation result-right iff)**.** *For histories P, Q, R : BHist with $\mathsf{Cont}(P, Q, R)$, the singleton group classifier relates the two continuation inputs exactly when it relates the continuation result to the right input:*

$$R_{\mathsf{e}}(P, Q) \iff R_{\mathsf{e}}(R, Q).$$

Proof. If $R_{\mathsf{e}}(P, Q)$ holds, then both inputs carry singleton witnesses. Transport the continuation result against the empty right-unit continuation to make R singleton-carried. Together with the existing carrier witness for Q, transitivity through e classifies R with Q.

Conversely, if $R_{\mathsf{e}}(R, Q)$ holds, then R and Q are singleton-carried. Transport the displayed continuation to an empty result using the witness for R, and then apply empty-result inversion for append and continuation. This recovers a singleton carrier witness for P while preserving the displayed singleton witness for Q.

The recovered carrier witnesses for P and Q, composed through e, rebuild $R_{\mathsf{e}}(P, Q)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation result right iff`

Theorem 41.20 (Group singleton continuation result carrier exactness)**.** *For histories P, Q, R : BHist with $\mathsf{Cont}(P, Q, R)$, the singleton group classifier relates the continuation inputs exactly when the continuation result lies in the singleton carrier:*

$$R_{\mathsf{e}}(P, Q) \iff A_{\mathsf{e}}(R).$$

Proof. Assume $R_{\mathsf{e}}(P, Q)$. The classifier exposes endpoint witnesses $A_{\mathsf{e}}(P)$ and $A_{\mathsf{e}}(Q)$. Transporting the displayed continuation against the empty right-unit continuation carries the result endpoint to e, and therefore classifies R as a singleton-carried history.

Conversely, assume $A_{\mathsf{e}}(R)$. Transport the continuation result to e along this carrier witness. Empty-result inversion for continuation and append then recovers endpoint witnesses $A_{\mathsf{e}}(P)$ and $A_{\mathsf{e}}(Q)$. Composing these endpoint witnesses through e packages the input pair classifier $R_{\mathsf{e}}(P, Q)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation result carrier iff`

Theorem 41.21 (Group singleton carrier continuation endpoint split)**.** *For histories P, Q, R : BHist with $\mathsf{Cont}(P, Q, R)$, the continuation result lies in the singleton carrier exactly when both continuation endpoints lie in the singleton carrier:*

$$A_{\mathsf{e}}(R) \iff A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q).$$

Proof. If $A_{\mathsf{e}}(R)$ holds, transport the displayed continuation result to e and apply empty-result inversion for continuation to recover the two endpoint carrier witnesses. Conversely, if both endpoint carriers hold, compare the displayed continuation with the right-unit continuation at e; continuation respect for hsame transports the result into the singleton carrier. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCarrier continuation endpoint split iff`

Theorem 41.22 (Group singleton inverse-continuation roundtrip carrier exactness)**.** *For histories x, r, s : BHist, if r is a continuation result of the singleton inverse of x followed by x, and s is a continuation result of r followed by the same singleton inverse, then the terminal carrier condition on the final result is exactly the terminal carrier condition on the original history:*

$$\mathsf{Cont}(\mathsf{GroupSingletonInv}(x), x, r) \Rightarrow \mathsf{Cont}(r, \mathsf{GroupSingletonInv}(x), s) \Rightarrow (A_{\mathsf{e}}(s) \iff A_{\mathsf{e}}(x)).$$

Proof. Apply Theorem 41.18 to the first continuation. This reads the left-unit inverse product back as $R_{\mathsf{e}}(r, \mathsf{e}) \iff A_{\mathsf{e}}(x)$.

The first row of Theorem 41.88, applied to r and e, converts $R_{\mathsf{e}}(r, \mathsf{e})$ into the carrier condition $A_{\mathsf{e}}(r)$, since $A_{\mathsf{e}}(\mathsf{e})$ is the reflexive terminal endpoint. Thus $A_{\mathsf{e}}(r) \iff A_{\mathsf{e}}(x)$.

Apply Theorem 41.20 to the second continuation. It identifies $A_{\mathsf{e}}(s)$ with $R_{\mathsf{e}}(r, \mathsf{GroupSingletonInv}(x))$.

The terminal classifier package and the constant inverse closure row of Theorem 41.4 identify this last classifier with $A_{\mathsf{e}}(r)$. Composing the two carrier equivalences gives $A_{\mathsf{e}}(s) \iff A_{\mathsf{e}}(x)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCarrier inverse continuation roundtrip iff`

The per-object certificate data are: traditional target terminal-group continuation-result coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist and Cont; carrier A_{e}; classifier R_{e}; stability obligations continuation result transport and empty-result inversion; ledger hidden continuation result endpoint; core theorem this block plus Definition 41.17; exactness predicate $R_{\mathsf{e}}(P, Q) \iff A_{\mathsf{e}}(R)$; standard bridge terminal-group product result; and status $\mathsf{ProofTarget}$.

The $\mathsf{Group}^{\uparrow}$ schema is instantiated by this concrete $BHist$ carrier and operations, so subsequent formalization can prove $BHist$-anchored group laws instead of relying only on an abstract operator schema.

Theorem 41.23 (Group singleton inverse-continuation roundtrip source exactness)**.** *For histories x, r, s : BHist, if r is a continuation result of $\mathsf{GroupSingletonInv}(x)$ followed by x, and s is a continuation result of r followed by $\mathsf{GroupSingletonInv}(x)$, then classifying the final result against the original source is exactly the singleton carrier condition on the original source:*

$$\mathsf{Cont}(\mathsf{GroupSingletonInv}(x), x, r) \Rightarrow \mathsf{Cont}(r, \mathsf{GroupSingletonInv}(x), s) \Rightarrow (R_{\mathsf{e}}(s, x) \iff A_{\mathsf{e}}(x)).$$

Proof. The forward direction projects the right endpoint carrier from $R_{\mathsf{e}}(s, x)$. Conversely, apply Definition 41.22 to transport the carrier condition on x back to a carrier condition on s. The two carrier witnesses compose through hsame symmetry and transitivity to rebuild $R_{\mathsf{e}}(s, x)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier inverse continuation roundtrip source iff`

41.3 Concrete singleton empty-history group instance

Definition 41.24 (Group singleton empty-history instance)**.** *Let A_{e} be the singleton history source at the empty endpoint:*

$$A_{\mathsf{e}}(h) := \mathsf{hsame}(h, \mathsf{e}).$$

The classifier, multiplication, unit, and inverse are

$$R_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k), \qquad \mathsf{mul}_{\mathsf{e}}(h, k) := \mathsf{append}(h, k), \qquad e_{\mathsf{e}} := \mathsf{e}, \qquad \mathsf{inv}_{\mathsf{e}}(h) := \mathsf{e}.$$

Equivalently, because both endpoints of the carrier are classified with e, the multiplication may be read through its classified empty endpoint. The displayed append row is the operational ledger, and the constant empty endpoint is the inverse row.

Theorem 41.25 (Group singleton empty-history laws)**.** *The data of Definition 41.24 satisfy the concrete $\mathsf{Group}^{\uparrow}$ laws over BHist: carrier inhabitance, multiplication closure, left and right unit laws, associativity and multiplication congruence from the continuation rows, inverse closure, and left and right inverse laws against e_{e}.*

Proof. Carrier inhabitance is the reflexive row $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, so $A_{\mathsf{e}}(\mathsf{e})$ holds.

For multiplication closure, suppose $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. The continuation append row for $\mathsf{append}(h, k)$ composes the two displayed paths to e with the empty endpoint laws, giving $A_{\mathsf{e}}(\mathsf{mul}_{\mathsf{e}}(h, k))$.

The left and right unit laws are the two empty-endpoint continuation laws for append. They identify $\mathsf{append}(\mathsf{e}, h)$ and $\mathsf{append}(h, \mathsf{e})$ with h; composing with the carrier witness $A_{\mathsf{e}}(h)$ classifies both products against the same singleton endpoint.

Associativity and multiplication congruence are inherited from the existing continuation rows for append. The carrier side conditions collapse all four endpoints into A_{e}, and the hsame equivalence rows repack the result as an R_{e} classifier statement.

Inverse closure is immediate because $\mathsf{inv}_{\mathsf{e}}(h)$ is e. The left and right inverse laws reduce to the same two unit laws: $\mathsf{append}(\mathsf{e}, h)$ and $\mathsf{append}(h, \mathsf{e})$ are classified with e_{e} by the carrier witness for h. □

Theorem 41.26 (Group singleton empty-history NameCert)**.** *The carrier A_{e}, classifier R_{e}, append multiplication, empty unit, and constant empty inverse package the laws of Definition 41.25 as a concrete $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ surface. The certificate exports only the singleton BHist carrier, the displayed continuation ledger for append, the empty endpoint, the constant inverse row, and the hsame classifier.*

Proof. The semantic carrier and classifier fields are supplied directly by Definition 41.24. Reflexivity, symmetry, and transitivity of the classifier are the corresponding hsame rows, with the two carrier side conditions retained in the package ledger.

The algebraic fields are exactly the rows proved in Definition 41.25. Multiplication closure, unit laws, associativity, and congruence come from the append continuation ledger; inverse closure and the inverse laws come from the constant empty inverse row.

Repacking these fields gives the concrete $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ surface. No abstract carrier, external operation table, quotient presentation, or host-level group object is exported beyond the singleton history data named above. □

41.4 Group concrete history instance surface

Definition 41.27 (Concrete GroupUp history instance)**.** *The concrete* $\mathsf{Group}^\uparrow$ *history instance is the singleton empty-history package of Definition 41.24 read as a direct* BHist *surface. It is the inverse-bearing refinement of the monoid certificate of chapter 40, with the same append-continuation unit surface and the added constant inverse row. Its carrier is*

$$A_{\mathsf{e}}(h) := \mathsf{hsame}(h, \mathsf{e}).$$

Its classifier is R_{e}*, its multiplication is the append continuation row*

$$\mathsf{mul}_{\mathsf{e}}(h, k) := \mathsf{append}(h, k),$$

its unit is $e_{\mathsf{e}} := \mathsf{e}$*, and its inverse sends every carried endpoint to the constant empty endpoint*

$$\mathsf{inv}_{\mathsf{e}}(h) := \mathsf{e}.$$

Thus the instance is not an abstract carrier with external operations. It is the displayed BHist *singleton carrier, append* Cont *ledger, empty endpoint,* hsame *classifier, and constant inverse row.*

Theorem 41.28 (Group concrete Monoid dependency)**.** *The concrete* $\mathsf{Group}^\uparrow$ *history instance of Definition 41.27 consumes the* $\mathsf{UnaryHistory}$ *append carrier, empty unit, classifier, associativity row, and left and right unit rows of chapter 40. It then adds only the constant inverse row and the singleton carrier restriction of Definition 41.24. Hence the concrete* $\mathsf{Group}^\uparrow$ *packet is a refinement of the* $\mathsf{Monoid}^\uparrow$ *unary append* $\mathsf{NameCert}$ *rather than an isolated kernel-only packet.*

Proof. Read the $\mathsf{Monoid}^\uparrow$ dependency from Definition 40.2. Its carrier is $\mathsf{UnaryHistory}$, its operation is append, its unit is e, and its classifier is the unary append monoid classifier.

Apply Definition 40.3. The associativity, left-unit, right-unit, and classifier congruence rows are already present on the unary append surface consumed by the group instance.

Finally use Definition 41.25. The singleton carrier restriction and the constant empty inverse row provide exactly the inverse closure and inverse laws. No additional host group carrier, operation table, or classifier is imported, so the concrete $\mathsf{Group}^\uparrow$ instance refines the $\mathsf{Monoid}^\uparrow$ unary append NameCert. □

Definition 41.29 (Group unary Cont concrete carrier)**.** *The* $\mathsf{Group}^\uparrow$ *unary* Cont *concrete carrier is the* $\mathsf{UnaryHistory}$ *append surface of Definition 40.2 restricted by the singleton empty-history carrier of Definition 41.27. Its carried endpoints are unary* BHist *values* h *with* $A_{\mathsf{e}}(h)$*, its product row is*

$$\mathsf{mul}_{\mathsf{e}}(h, k) := \mathsf{append}(h, k),$$

equivalently displayed by $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$*, and its unit row is the empty endpoint* e*. The classifier is the inherited* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$ *together with the singleton* hsame *restriction to* e*. Thus the carrier names the concrete append-and-empty operation surface consumed by the* $\mathsf{Group}^{\uparrow}$ *singleton instance, not an abstract group parameter.*

Theorem 41.30 (Group unary Cont concrete monoid handoff)**.** *Every multiplication and unit read of the concrete* $\mathsf{Group}^{\uparrow}$ *history instance of Definition 41.27 factors through the unary* Cont *carrier of Definition 41.29. In particular, the* $\mathsf{Group}^{\uparrow}$ *singleton packet reads multiplication as the* $\mathsf{UnaryHistory}$ *append row, reads the unit as* e*, and obtains closure, associativity, left unit, right unit, and multiplication congruence from Definition 40.3 before it adds the constant empty inverse rows of Definition 41.25.*

Proof. Project the concrete group instance. Its multiplication is append, its unit is e, and its carried endpoints are singleton empty-history endpoints inside the unary BHist surface.

Apply Definition 40.3. The unary append carrier supplies closure, associativity, both empty-unit laws, and multiplication congruence for exactly the same Cont product and unit rows.

Now restrict those rows along Definition 41.29. The singleton carrier only adds the requirement that the endpoints classify with e; it does not replace append, the unit, or the monoid classifier by a separate algebraic operation.

Finally apply Definition 41.25. The only additional group data are inverse closure and the two inverse laws for the constant empty inverse row. Therefore the concrete $\mathsf{Group}^{\uparrow}$ multiplication and unit handoff is the unary Cont monoid surface followed by singleton inverse packaging. □

Theorem 41.31 (Group concrete inverse surface)**.** *For the concrete* $\mathsf{Group}^{\uparrow}$ *history instance of Definition 41.27, inverse closure and the two inverse laws are read entirely from the singleton empty-history laws and the append unit rows. If* $A_{\mathsf{e}}(h)$*, then* $A_{\mathsf{e}}(\mathsf{inv}_{\mathsf{e}}(h))$ *holds, and both products* $\mathsf{mul}_{\mathsf{e}}(\mathsf{inv}_{\mathsf{e}}(h), h)$ *and* $\mathsf{mul}_{\mathsf{e}}(h, \mathsf{inv}_{\mathsf{e}}(h))$ *classify with the unit* e_{e} *through* R_{e}*.*

Proof. Start from the concrete definition. The inverse endpoint is the constant empty history, so inverse closure is the reflexive row $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

For the left inverse law, expand $\mathsf{mul}_{\mathsf{e}}(\mathsf{inv}_{\mathsf{e}}(h), h)$ as $\mathsf{append}(\mathsf{e}, h)$. The left append unit row identifies this endpoint with h, and the carrier witness $A_{\mathsf{e}}(h)$ identifies h with e. These two hsame rows package the required R_{e} comparison with e_{e}.

For the right inverse law, expand $\mathsf{mul}_{\mathsf{e}}(h, \mathsf{inv}_{\mathsf{e}}(h))$ as $\mathsf{append}(h, \mathsf{e})$. The right append unit row identifies this endpoint with h, and the same carrier witness repacks it as an R_{e} comparison with the unit.

These are exactly the inverse rows already included in Definition 41.25; the concrete surface only names the BHist rows that carry them. □

Theorem 41.32 (Group concrete instance schema surface)**.** *The concrete* $\mathsf{Group}^{\uparrow}$ *history instance of Definition 41.27 supplies a non-parametric target surface for the algebra hierarchy. Its carrier, classifier, multiplication, unit, inverse, closure rows, unit rows, associativity row, congruence row, and inverse rows are all the displayed singleton* BHist*, append* Cont*,* hsame*, and empty-endpoint rows of Definition 41.25. No external carrier family, operation table, quotient presentation, or host-level group object is part of the target surface.*

Proof. Read the certificate rows from Definition 41.26. The semantic carrier and classifier are A_{e} and R_{e}, and the algebraic fields are the append multiplication, empty unit, and constant empty inverse.

Apply Definition 41.25. It supplies inhabitance, multiplication closure, the two unit laws, associativity, congruence, inverse closure, and the two inverse laws for these exact rows.

Finally apply Definition 41.31 to make the inverse rows visible as part of the concrete BHist surface. Since every field has been read from the singleton carrier and append ledger, the target is the concrete instance itself rather than a parameterized group schema. □

Theorem 41.33 (Group concrete history instance public laws)**.** *The concrete singleton* $\mathsf{Group}^{\uparrow}$ *history instance exports closure, associativity, left unit, right unit, congruence, inverse closure, left inverse, and right inverse laws by reading the* $\mathsf{Monoid}^{\uparrow}$ *unary append laws together with the singleton empty-history inverse laws. Concretely, the public law surface uses the carrier* A_{e}*, classifier* R_{e}*, multiplication* append*, unit* e*, classifier transport* hsame*, and continuation replay* Cont *from Definition 41.27 and Definition 41.28.*

Proof. Start from the concrete singleton history instance. The carrier is A_{e}, the classifier is R_{e}, the multiplication is append, and the unit is e.

Apply Definition 41.28 and Definition 41.31. The monoid dependency supplies closure, associativity, left unit, right unit, and congruence on the append surface, while the singleton inverse surface supplies inverse closure and the two inverse laws.

Finally read the laws as public rows of the same concrete packet. Because the carrier is singleton and the inverse is constant empty history, the public surface is non-parametric and uses only the displayed A_{e}, R_{e}, append, e, hsame, and Cont rows. □

Definition 41.34 (Group finite-word history instance)**.** *The finite-word* $\mathsf{Group}^{\uparrow}$ *history instance has carrier* $\mathsf{WordCarrier}_{\mathsf{Group}^{\uparrow}}$ *consisting of finite* $\mathsf{List}^{\uparrow}$ *word spines whose entries are singleton carried endpoints of Definition 41.24. Thus a word* $w = \mathsf{cons}(h_1, \ldots, \mathsf{cons}(h_n, \mathsf{nil}))$ *is carried when each* h_i *satisfies* $A_{\mathsf{e}}(h_i)$*, with the empty word carried by the nil constructor.*

The classifier $\mathsf{WordClass}_{\mathsf{Group}^{\uparrow}}(w, w')$ *is the componentwise lift of* hsame*, together with equality of the finite word spine. The product is concatenation of finite words through the displayed append continuation ledger*

$$\mathsf{mul}_{\mathrm{word}}(w, w') := w +\!\!+ w',$$

the unit is nil*, and inverse is reversal with the singleton inverse row applied to each component:*

$$\mathsf{inv}_{\mathrm{word}}(h_1, \ldots, h_n) := (\mathsf{inv}_{\mathsf{e}}(h_n), \ldots, \mathsf{inv}_{\mathsf{e}}(h_1)).$$

No host group operation is exported; every endpoint remains a finite BHist *word with its componentwise singleton ledger.*

Theorem 41.35 (Group finite-word operation closure)**.** *For the finite-word* $\mathsf{Group}^{\uparrow}$ *history instance of Definition 41.34, concatenation of two carried finite-word histories is carried. If* $w \sim_{\mathrm{word}} w'$ *and* $v \sim_{\mathrm{word}} v'$*, then* $\mathsf{mul}_{\mathrm{word}}(w, v) \sim_{\mathrm{word}} \mathsf{mul}_{\mathrm{word}}(w', v')$*. The empty word is a two-sided unit, and associativity is the list-spine append associativity row together with the singleton append laws of Definition 41.25.*

Proof. Induct on the first finite word. In the nil case, concatenation returns the second carried word, and the left unit law is the nil append equation. The right unit law follows by the corresponding induction on the carried word.

In the cons step, the head is carried by the componentwise carrier hypothesis and the tail product is carried by the induction hypothesis. Reattaching the head gives a carried concatenation spine, so no endpoint leaves the singleton BHist surface.

Classifier transport is componentwise. The head comparison is the displayed hsame row, while the tail comparison is transported by the induction hypothesis. The list-spine equality part records that concatenation preserves the finite shape on both sides.

Associativity is the standard finite-list append reassociation, with each component endpoint governed by Definition 41.25. The local concrete schema surface of Definition 41.32 supplies the same append ledger, so the finite-word operation remains the concrete BHist instance rather than an abstract group law. □

Theorem 41.36 (Group finite-word Cont associativity lock)**.** *For the finite-word* $\mathsf{Group}^{\uparrow}$ *history instance of Definition 41.34, the associativity route for the product is the displayed* Cont *replay over* BHist *word rows. Thus the two parenthesized products*

$$\mathsf{mul}_{\mathrm{word}}(\mathsf{mul}_{\mathrm{word}}(u, v), w) \quad \textit{and} \quad \mathsf{mul}_{\mathrm{word}}(u, \mathsf{mul}_{\mathrm{word}}(v, w))$$

are compared as the same ordered continuation spine. The lock is inherited from the kernel continuation associativity route and the finite list-spine append ledger; it is not supplied by an abstract mul *parameter.*

Proof. Unfold the finite-word carrier rows. Each carried word is a finite ordered list of singleton BHist endpoints, and the product is concatenation with the displayed append ledger.

Compare the two parenthesized products by reading the same three blocks u, v, w through the continuation spine. The left parenthesization first replays the route for u followed by v and then appends w; the right parenthesization first replays v followed by w and then prefixes u.

The kernel Cont associativity route identifies these as one ordered replay of the same finite rows. The displayed transport rows keep the component classifiers fixed while the list-spine append ledger changes only the parentheses.

Finally pass the replayed spine through the $\mathsf{Group}^{\uparrow}$ classifier of Definition 41.34. Since every endpoint remains a singleton BHist row, the associativity lock is a concrete continuation fact and not a parameter-echo group schema. □

Theorem 41.37 (Group finite-word inverse boundary)**.** *For every carried finite-word history* w*, the reversed componentwise singleton inverse* $\mathsf{inv}_{\mathrm{word}}(w)$ *is carried. Moreover,* $\mathsf{mul}_{\mathrm{word}}(w, \mathsf{inv}_{\mathrm{word}}(w))$ *and* $\mathsf{mul}_{\mathrm{word}}(\mathsf{inv}_{\mathrm{word}}(w), w)$ *expose exactly the displayed finite cancellation ledger obtained by pairing each component with its singleton inverse row; the boundary does not export a host group object or an external quotient.*

Proof. Induct on the finite word. The nil case is the empty word, whose inverse is again nil; both products are the nil append row.

In the cons case, apply Definition 41.25 to the head endpoint and Definition 41.26 to keep the transported classifier row inside the singleton certificate. The induction hypothesis gives the reversed inverse word for the tail and its carriedness.

Concatenating the word with its reversed inverse produces the visible sequence of componentwise singleton cancellation rows. Reassociating this finite append spine uses the same append ledger as Definition 41.32; it only rearranges the displayed continuation evidence.

The opposite product is identical after reversing the induction order. Thus the inverse boundary is a carried $\mathsf{List}^{\uparrow}$ word endpoint with a finite cancellation ledger, not a transported theorem about an ambient group object. □

Theorem 41.38 (Group finite-word NameCert package)**.** *The finite-word carrier, classifier, concatenation product, empty-word unit, and reversal inverse of Definition 41.34, together with operation closure from Definition 41.35, inverse boundary from Definition 41.37, singleton laws from Definition 41.25, and the* $\mathsf{Monoid}^{\uparrow}$ *dependency of Definition 41.28, form the local finite-word* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *package. Hence the algebra-hierarchy consumer receives a concrete finite* BHist *word surface rather than an abstract group parameter.*

Proof. Project the rows of Definition 41.34. The carrier is $\mathsf{WordCarrier}_{\mathsf{Group}^{\uparrow}}$, the classifier is $\mathsf{WordClass}_{\mathsf{Group}^{\uparrow}}$, the product is finite concatenation, the unit is nil, and inverse is reversal followed by the singleton inverse row on each component.

Apply Definition 41.35. It supplies multiplication closure, classifier transport for multiplication, the empty-word unit laws, and finite append associativity on exactly these projected rows. The dependency on Definition 41.28 identifies this operation ledger with the unary append $\mathsf{Monoid}^{\uparrow}$ surface consumed by the group packet.

Next apply Definition 41.37. It supplies inverse closure and both finite cancellation ledgers for the reversal inverse. The componentwise endpoint obligations are the singleton inverse and unit rows of Definition 41.25, so no external group carrier or quotient relation is imported.

These rows exhaust the local $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ fields: carried finite words, componentwise classification, product, unit, inverse, product closure, unit and associativity laws, inverse closure, and inverse laws. Repacking them gives the concrete algebra-hierarchy object as a finite BHist word package, not a schema parameter awaiting a host group. □

41.5 Singleton inverse image exactness

Theorem 41.39 (Group singleton inverse image exactness)**.** *For histories* $h, t : \mathsf{BHist}$*, if* $A_{\mathsf{e}}(h)$*, then the concrete constant inverse classifies precisely the singleton carrier:*

$$R_{\mathsf{e}}(\mathsf{GroupSingletonInv}(h), t) \iff A_{\mathsf{e}}(t).$$

The ten-field fit is: traditional target terminal-group inverse image; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability from constant inverse closure and terminal classifier packaging; ledger the hidden inverse endpoint; core theorem this block plus Theorem 41.88; exactness the displayed iff; downstream consumers* $\mathsf{QuotientGroup}^{\uparrow}$ *and singleton normalizer-action proofs; and status* $\mathsf{ProofTarget}$*.*

Proof. Assume $A_{\mathsf{e}}(h)$. The concrete singleton inverse unfolds to the constant endpoint e, so $A_{\mathsf{e}}(\mathsf{GroupSingletonInv}(h))$ is the reflexive singleton witness at e.

For the forward direction, unpack $R_{\mathsf{e}}(\mathsf{GroupSingletonInv}(h), t)$. The right carrier field is exactly $A_{\mathsf{e}}(t)$.

For the reverse direction, assume $A_{\mathsf{e}}(t)$. Combine the reflexive carrier witness for $\mathsf{GroupSingletonInv}(h)$ with the supplied carrier witness for t. Symmetry turns the endpoint witness for t into a path from e to t, and transitivity composes it with the reflexive identification of $\mathsf{GroupSingletonInv}(h)$ with e. These three fields package $R_{\mathsf{e}}(\mathsf{GroupSingletonInv}(h), t)$. □

Theorem 41.40 (Group singleton inverse classifier fiber)**.** *For histories* $x, y : \mathsf{BHist}$ *in the concrete singleton history group of Definition 41.3, the constant singleton inverse has exactly the terminal classifier fiber:*

$$R_{\mathsf{e}}(\mathsf{inv}(x), y) \iff A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal-group inverse image; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability from singleton classifier reflexivity, symmetry, transitivity, and inverse closure; ledger the ignored inverse input* x *and the terminal endpoint* e*; core theorem this block plus Theorem 41.4; exactness the displayed iff; standard bridge the one-element group inverse fiber; status* $\mathsf{ProofTarget}$.

Proof. Unfold the concrete singleton inverse from Definition 41.3. For every history x, $\mathsf{inv}(x)$ is definitionally e, so the displayed left side is the classifier row $R_{\mathsf{e}}(\mathsf{e}, y)$.

For the forward direction, unpack $R_{\mathsf{e}}(\mathsf{e}, y)$. By definition of R_{e} this exposes three fields: $A_{\mathsf{e}}(\mathsf{e})$, $A_{\mathsf{e}}(y)$, and $\mathsf{hsame}(\mathsf{e}, y)$. Project the right carrier field to obtain $A_{\mathsf{e}}(y)$.

For the reverse direction, assume $A_{\mathsf{e}}(y)$. The singleton carrier contains e by the reflexive endpoint witness $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. The assumed carrier fact identifies y with e; applying singleton symmetry gives the endpoint witness $\mathsf{hsame}(\mathsf{e}, y)$. Repack these three fields as $R_{\mathsf{e}}(\mathsf{e}, y)$, hence as $R_{\mathsf{e}}(\mathsf{inv}(x), y)$ after the inverse unfolds. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier inverse empty fiber iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier inverse fiber empty endpoint iff`

Theorem 41.41 (Group singleton compiled concrete instance boundary)**.** *The singleton carrier* A_{e}*, classifier* R_{e}*, multiplication* append*, unit* e*, and constant inverse* e *of Definition 41.3 simultaneously supply the singleton-history* $\mathsf{SemanticNameCert}$ *for* $\mathsf{Group}^{\uparrow}$ *and the full concrete group algebraic-law package over* BHist*: carrier inhabitance, multiplication closure, inverse closure, the left and right unit laws, associativity up to* hsame*, multiplication congruence, inverse congruence, and the left and right inverse laws.*

Proof. The certificate component is exactly Theorem 41.4. It packages the carrier-aware classifier R_{e} with the append multiplication, empty unit, and constant empty inverse as the singleton-history $\mathsf{SemanticNameCert}$ for $\mathsf{Group}^{\uparrow}$.

The algebraic-law component is Theorem 41.5. Its rows give carrier inhabitance, closure under multiplication and inverse, the two unit laws, associativity, multiplication congruence, inverse congruence, and the two inverse laws for the same carrier, classifier, multiplication, unit, and inverse. The two cited blocks therefore have identical concrete data, so they assemble the stated concrete instance boundary. □

Theorem 41.42 (Group singleton commutator terminal collapse)**.** *For histories* $x, y : \mathsf{BHist}$ *in the concrete singleton history group of Definition 41.3, define*

$$\mathsf{Comm}_{\mathsf{e}}(x, y) := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(x, y), \mathsf{inv}(x)), \mathsf{inv}(y)).$$

If $A_{\mathsf{e}}(x)$ *and* $A_{\mathsf{e}}(y)$*, then* $A_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y))$ *and* $R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y), \mathsf{e})$*. The ten-field fit is: traditional target terminal-group commutator fiber; BEDC source concrete singleton* $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability from multiplication closure, inverse closure, and terminal classifier exactness; ledger the two factors and inverse endpoints; core theorem this block plus Theorem 41.88; exactness terminal commutator classified with the unit; downstream consumers* $\mathsf{AbGroup}^{\uparrow}$ *and* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Start from $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$. The inverse row of Theorem 41.88 places $\mathsf{inv}(x)$ and $\mathsf{inv}(y)$ in A_{e}.

Apply the multiplication-closure row three times. It first places $\mathsf{mul}(x, y)$ in A_{e}, then places $\mathsf{mul}(\mathsf{mul}(x, y), \mathsf{inv}(x))$ in A_{e}, and finally places $\mathsf{Comm}_{\mathsf{e}}(x, y)$ in A_{e}.

The carrier inhabitance row gives $A_{\mathsf{e}}(\mathsf{e})$. With the carrier fact for $\mathsf{Comm}_{\mathsf{e}}(x, y)$ and the carrier fact for e, the terminal classifier row of Theorem 41.88 packages $R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y), \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCommutator terminal collapse`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier commutator terminal collapse`

Theorem 41.43 (Group singleton commutator unit exactness)**.** *For histories $x, y : \mathsf{BHist}$ in the concrete singleton history group of Definition 41.3, define*

$$\mathsf{Comm_e}(x, y) := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(x, y), \mathsf{inv}(x)), \mathsf{inv}(y)).$$

Then the commutator is classified with the unit exactly when both inputs lie in the singleton carrier:

$$R_{\mathsf{e}}(\mathsf{Comm_e}(x, y), \mathsf{e}) \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal-group commutator fiber exactness; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability from product-unit splitting, context cancellation, inverse closure, and terminal classifier exactness; ledger the two factors and two constant inverse endpoints; core theorem this exactness plus Theorem 41.10 and Theorem 41.11; exactness the displayed iff; downstream consumers* $\mathsf{AbGroup}^{\uparrow}$ *and* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Unfold the concrete operations. Here inv is the constant endpoint e and mul is append, so the left side is the terminal classifier of the nested append word $\mathsf{append}(\mathsf{append}(\mathsf{append}(x, y), \mathsf{e}), \mathsf{e})$ against e.

For the forward direction, project the classifier to the carried endpoint e. The unit endpoint is carried, so terminal classifier exactness reduces the hypothesis to singleton carrier membership of the nested append word.

Apply Theorem 41.10 to split off the outer constant inverse endpoint, then apply the same product-unit split to remove the inner constant inverse endpoint. The remaining classifier is the product-unit classifier for $\mathsf{append}(x, y)$, and a final application gives exactly $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$.

For the reverse direction, assume $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$. Theorem 41.42 applies to these two carrier witnesses and gives $R_{\mathsf{e}}(\mathsf{Comm_e}(x, y), \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier commutator unit exactness iff`

Theorem 41.44 (Group singleton commutator carrier exactness)**.** *For histories $x, y : \mathsf{BHist}$ in the concrete singleton history group of Definition 41.3, with $\mathsf{Comm_e}$ defined as in Definition 41.42, the commutator carrier and terminal classifier are exactly the two input carrier conditions:*

$$A_{\mathsf{e}}(\mathsf{Comm_e}(x, y)) \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

Moreover,

$$R_{\mathsf{e}}(\mathsf{Comm_e}(x, y), \mathsf{e}) \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal-group commutator fiber exactness; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability append-empty split plus terminal classifier exactness; ledger the two factors and constant inverse endpoints; core theorem this exactness plus Definition 41.42; exactness the displayed iff; downstream consumers* $\mathsf{AbGroup}^{\uparrow}$ *and* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Unfold the concrete singleton group operations. The multiplication is append and the inverse endpoint is constant, so $\mathsf{inv}(x) = \mathsf{e}$ and $\mathsf{inv}(y) = \mathsf{e}$. Thus the commutator carrier condition is the singleton-empty condition on the append product whose two right factors are empty endpoints.

For the forward carrier direction, append-empty inversion first removes the outer empty inverse endpoint, then removes the inner empty inverse endpoint, and finally applies the singleton append-empty split to the remaining product $\mathsf{append}(x, y)$. This yields $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$.

For the reverse carrier direction, the supplied witnesses for x and y rebuild $A_{\mathsf{e}}(\mathsf{append}(x, y))$ by the singleton append-empty split. The reflexive empty witnesses for the two constant inverse endpoints then rebuild the full commutator carrier by append-empty construction.

It remains to read the classifier against the terminal endpoint. The first row of Definition 41.88 identifies $R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y), \mathsf{e})$ with $A_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y)) \wedge A_{\mathsf{e}}(\mathsf{e})$, and $A_{\mathsf{e}}(\mathsf{e})$ is reflexive. Composing this terminal classifier exactness with the carrier equivalence gives the displayed classifier iff. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append commutator terminal exactness iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCommutator carrier classifier exactness`

41.6 Concrete singleton equation solver

Theorem 41.45 (Group singleton left equation exactness)**.** *In the concrete singleton history group of Definition 41.3, with carrier A_{e}, classifier R_{e}, multiplication* $\mathsf{mul} := \mathsf{append}$*, unit* e*, and inverse* $\mathsf{inv}(h) := \mathsf{e}$*, the left equation row is exact for histories $a, x, b : \mathsf{BHist}$. If $A_{\mathsf{e}}(a)$, $A_{\mathsf{e}}(x)$, and $A_{\mathsf{e}}(b)$, then*

$$R_{\mathsf{e}}(\mathsf{mul}(a, x), b) \iff R_{\mathsf{e}}(x, \mathsf{mul}(\mathsf{inv}(a), b)).$$

Proof. Theorem 41.5 supplies the concrete group rows for the same carrier, classifier, multiplication, unit, and inverse. Multiplication closure and inverse closure place both displayed endpoints in A_{e}.

For the forward direction, unpack $R_{\mathsf{e}}(\mathsf{mul}(a, x), b)$ to the hsame component $\mathsf{mul}(a, x)\mathsf{hsame}b$. Apply Theorem 41.70; the resulting $\mathsf{hsame}(x, \mathsf{mul}(\mathsf{inv}(a), b))$ repacks with the carrier rows as $R_{\mathsf{e}}(x, \mathsf{mul}(\mathsf{inv}(a), b))$.

For the reverse direction, start from the unpacked component $\mathsf{hsame}(x, \mathsf{mul}(\mathsf{inv}(a), b))$. The backward direction of Theorem 41.70 gives $\mathsf{mul}(a, x)\mathsf{hsame}b$, and the same endpoint carrier rows repack it as $R_{\mathsf{e}}(\mathsf{mul}(a, x), b)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier left equation exact iff`

Theorem 41.46 (Group singleton right equation exactness)**.** *In the concrete singleton history group of Definition 41.3, with carrier A_{e}, classifier R_{e}, multiplication* $\mathsf{mul} := \mathsf{append}$*, unit* e*, and inverse* $\mathsf{inv}(h) := \mathsf{e}$*, the right equation row is exact for histories $x, a, b : \mathsf{BHist}$. If $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(a)$, and $A_{\mathsf{e}}(b)$, then*

$$R_{\mathsf{e}}(\mathsf{mul}(x, a), b) \iff R_{\mathsf{e}}(x, \mathsf{mul}(b, \mathsf{inv}(a))).$$

Proof. Theorem 41.5 supplies the concrete group rows for the same carrier, classifier, multiplication, unit, and inverse. Multiplication closure and inverse closure place both displayed endpoints in A_{e}.

For the forward direction, unpack $R_{\mathsf{e}}(\mathsf{mul}(x, a), b)$ to the corresponding hsame component. Apply Theorem 41.72; it transports $\mathsf{mul}(x, a)\mathsf{hsame}b$ to $\mathsf{hsame}(x, \mathsf{mul}(b, \mathsf{inv}(a)))$, and carrier closure repacks the right-hand classifier.

For the reverse direction, start from the unpacked component $\mathsf{hsame}(x, \mathsf{mul}(b, \mathsf{inv}(a)))$. The backward direction of Theorem 41.72 returns $\mathsf{mul}(x, a)\mathsf{hsame}b$, and endpoint carrier closure repacks the left-hand classifier. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier right equation exact iff`

Theorem 41.47 (Group singleton two-sided cancellation)**.** *In the concrete singleton history group of Definition 41.3, with carrier A_{e}, classifier R_{e}, multiplication* $\mathsf{mul} := \mathsf{append}$*, unit* e*, and inverse* $\mathsf{inv}(h) := \mathsf{e}$*, two-sided cancellation is exact for histories $a, b, c, d : \mathsf{BHist}$. If $A_{\mathsf{e}}(a)$, $A_{\mathsf{e}}(b)$, $A_{\mathsf{e}}(c)$, and $A_{\mathsf{e}}(d)$, then*

$$R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, b), c), \mathsf{mul}(\mathsf{mul}(a, d), c)) \Longrightarrow R_{\mathsf{e}}(b, d).$$

Proof. Theorem 41.5 supplies the concrete group rows for the same carrier, classifier, multiplication, unit, and inverse. Repeated multiplication closure places both displayed products in A_{e}.

Unpacking the assumed classifier gives the hsame component between those two products. Apply Theorem 41.119; the result is $\mathsf{hsame}(b, d)$, and the original carrier hypotheses $A_{\mathsf{e}}(b)$ and $A_{\mathsf{e}}(d)$ repack it as $R_{\mathsf{e}}(b, d)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided cancel`

Theorem 41.48 (Group singleton equation-cancellation concrete solver)**.** *In the concrete singleton history group of Definition 41.3, the package of Theorem 41.45, Theorem 41.46, and Theorem 41.47 supplies the concrete equation-cancellation solver over* BHist*.*

The ten-field fit is: traditional target terminal-group equation solving; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability from concrete group laws plus generic left and right equation exactness and two-sided cancellation; ledger equation factors and inverse endpoints; core theorem list Theorem 41.5, Theorem 41.70, Theorem 41.72, and Theorem 41.119; exactness the three displayed rows; downstream consumers* $\mathsf{QuotientGroup}^{\uparrow}$ *and* $\mathsf{Field}^{\uparrow}$ *algebra-reduct packages; status* $\mathsf{ProofTarget}$*.*

Proof. The left equation row is Theorem 41.45, the right equation row is Theorem 41.46, and the cancellation row is Theorem 41.47.

These three rows use the same carrier, classifier, multiplication, unit, inverse, and concrete group-law source, so they assemble into one solver package with the listed source, stability, ledger, exactness, consumer, and status fields. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier equation cancellation solver`

41.7 Concrete singleton equation solver determinacy

Theorem 41.49 (Group singleton equation solver determinacy)**.** *In the concrete singleton history group of Definition 41.3, with carrier A_{e}, classifier R_{e}, multiplication* $\mathsf{mul} := \mathsf{append}$*, unit* e*, and inverse* $\mathsf{inv}(h) := \mathsf{e}$*, let $a, b, c, d, x, y : \mathsf{BHist}$. If $A_{\mathsf{e}}(a)$, $A_{\mathsf{e}}(b)$, $A_{\mathsf{e}}(c)$, $A_{\mathsf{e}}(d)$, $A_{\mathsf{e}}(x)$, and $A_{\mathsf{e}}(y)$, then the concrete solver is deterministic in the following same-target forms:*

1. *If $R_{\mathsf{e}}(\mathsf{mul}(a, x), b)$ and $R_{\mathsf{e}}(\mathsf{mul}(a, y), b)$, then $R_{\mathsf{e}}(x, y)$.*
2. *If $R_{\mathsf{e}}(\mathsf{mul}(x, a), b)$ and $R_{\mathsf{e}}(\mathsf{mul}(y, a), b)$, then $R_{\mathsf{e}}(x, y)$.*
3. *If $R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, x), c), d)$ and $R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, y), c), d)$, then $R_{\mathsf{e}}(x, y)$.*

The ten-field fit is: traditional target terminal-group solution determinacy; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability from concrete solver exactness plus classifier symmetry and transitivity; ledger hidden equation target and contexts; core theorem list Theorem 41.48 plus the classifier equivalence fields; exactness same-target solution determinacy; downstream consumers* $\mathsf{QuotientGroup}^{\uparrow}$ *and* $\mathsf{Field}^{\uparrow}$ *algebra-reduct packages; status* $\mathsf{ProofTarget}$*.*

Proof. Theorem 41.48 gives the three concrete rows for the same carrier, classifier, multiplication, unit, and inverse. The concrete classifier fields supply symmetry and transitivity on R_{e} whenever the displayed endpoints are in A_{e}.

1. For the left-multiplication row, apply the forward direction of the concrete solver to $R_{\mathsf{e}}(\mathsf{mul}(a, x), b)$. This transports x to the endpoint $\mathsf{mul}(\mathsf{inv}(a), b)$.

   Apply the same forward direction to $R_{\mathsf{e}}(\mathsf{mul}(a, y), b)$. Now y is transported to the same solved endpoint. Use classifier symmetry on the second solved comparison, then classifier transitivity through the common endpoint to obtain $R_{\mathsf{e}}(x, y)$.

2. For the right-multiplication row, apply the forward direction of the concrete solver to both assumptions. The two transported comparisons classify x and y by the common endpoint $\mathsf{mul}(b, \mathsf{inv}(a))$.

   Symmetry on the solved comparison for y, followed by transitivity through that common endpoint, gives $R_{\mathsf{e}}(x, y)$.

3. For the two-sided row, first use symmetry on $R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, y), c), d)$. Transitivity with $R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, x), c), d)$ through the common target d gives

$$R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, x), c), \mathsf{mul}(\mathsf{mul}(a, y), c)).$$

   Apply the two-sided cancellation row of Theorem 41.48 to this classifier. The result is $R_{\mathsf{e}}(x, y)$.

□

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier equation solver determinacy`

Theorem 41.50 (Group singleton conjugated commutator invariance)**.** *For histories $s, x, y : \mathsf{BHist}$ in the concrete singleton history group of Definition 41.3, define*

$$\mathsf{Conj}_s(z) := \mathsf{mul}(\mathsf{mul}(s, z), \mathsf{inv}(s))$$

and use $\mathsf{Comm}_{\mathsf{e}}$ as in Definition 41.42. If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, and $A_{\mathsf{e}}(y)$, then

$$R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)), \mathsf{Comm}_{\mathsf{e}}(x, y)),$$

and both commutator endpoints classify against e:

$$R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)), \mathsf{e}) \wedge R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y), \mathsf{e}).$$

The ten-field fit is: traditional target terminal commutator conjugation invariance; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability from action closure, commutator closure, and terminal classifier exactness; ledger the conjugator plus two commutator inputs; core theorem this composite invariance; exactness the displayed concrete commutator classifier; downstream consumers $\mathsf{AbGroup}^{\uparrow}$ and $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Proof. Apply Theorem 41.147 to (s, x) and again to (s, y). These two instances put $\mathsf{Conj}_s(x)$ and $\mathsf{Conj}_s(y)$ in the singleton carrier A_{e}.

Next apply Theorem 41.42 to the original pair (x, y). It gives the carrier fact for $\mathsf{Comm}_{\mathsf{e}}(x, y)$ and the classifier $R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y), \mathsf{e})$.

Apply the same commutator-collapse theorem to the conjugated pair $(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y))$. The action-carrier facts just built supply its hypotheses, so the theorem gives both $A_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)))$ and $R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)), \mathsf{e})$.

Finally use the terminal classifier row of Theorem 41.88 on the two commutator carrier facts. It classifies the conjugated commutator endpoint against the original commutator endpoint by R_{e}, while the two commutator-collapse applications provide the two classifiers against e. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCommutator conjugated invariance`

Theorem 41.51 (Group singleton conjugated commutator orbit invariance)**.** *For histories s, x, y : BHist in the concrete singleton history group of Definition 41.3, define*

$$\mathsf{Conj}_s(z) := \mathsf{mul}(\mathsf{mul}(s, z), \mathsf{inv}(s))$$

and use $\mathsf{Comm}_{\mathsf{e}}$ as in Definition 41.42. If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, and $A_{\mathsf{e}}(y)$, then

$$\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)) \sim_{N_{\mathsf{e}}} \mathsf{Comm}_{\mathsf{e}}(x, y) \wedge \mathsf{Comm}_{\mathsf{e}}(x, y) \sim_{N_{\mathsf{e}}} \mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)).$$

The ten-field fit is: traditional target terminal commutator conjugation on normalizer orbits; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier the normalizer orbit relation exact against R_{e}; stability from conjugated commutator classifier invariance and terminal classifier exactness; ledger the conjugator and two commutator inputs; core theorem this block; exactness the displayed bidirectional orbit relation; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Proof. Apply Theorem 41.50 to s, x, and y. It gives the classifier

$$R_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)), \mathsf{Comm}_{\mathsf{e}}(x, y))$$

and the two classifiers against e.

The terminal classifier package turns those endpoint classifiers into carrier facts. Concretely, Theorem 41.88 applied to each commutator endpoint and e yields $A_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)))$ and $A_{\mathsf{e}}(\mathsf{Comm}_{\mathsf{e}}(x, y))$.

Use the classifier-to-orbit direction of Theorem 41.150 with these two carrier facts. The displayed R_{e} row becomes $\mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)) \sim_{N_{\mathsf{e}}} \mathsf{Comm}_{\mathsf{e}}(x, y)$.

For the reverse orientation, apply classifier symmetry from Theorem 41.88 to the same R_{e} row. Orbit exactness then converts the symmetric classifier into $\mathsf{Comm}_{\mathsf{e}}(x, y) \sim_{N_{\mathsf{e}}} \mathsf{Comm}_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y))$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCommutator conjugated orbit invariance`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier conjugated commutator orbit invariance`

41.8 The certificate

Definition 41.52 (Group source specification)**.** *The source is a certified monoid carrier plus inverse data.*

Definition 41.53 (Group pattern specification)**.** *The pattern is monoid expression formation extended by inverse formation.*

Definition 41.54 (Group classifier specification)**.** *The classifier is the monoid carrier classifier, with inverse congruence.*

Definition 41.55 (Group stability certificate)**.** *The stability certificate consists of:*

1. *all monoid stability fields*

2. *inverse congruence*
3. *left inverse law*
4. *right inverse law*
5. *closure under inverse*

Lean4 ✓ `BEDC.Derived.GroupUp.group stability certificate fields`

Theorem 41.56 (Group inverse of identity)**.** *For a history-level group operation with right identity and left inverses, the inverse of the identity is internally the same as the identity:* $\mathsf{hsame}(e^{-1}, e)$.

Proof. Chain e^{-1} backward through the right identity $e^{-1} \cdot e\mathsf{hsame}e^{-1}$, then apply the left inverse law at e. □

Lean4 ✓ `BEDC.Derived.GroupUp.group inverse identity`

Theorem 41.57 (Left inverse involutivity)**.** *For a history-level group operation with associativity, left and right identity, multiplication congruence, and left inverses, the left inverse operation is involutive:* $(x^{-1})^{-1}\mathsf{hsame}x$.

Proof. Rewrite $(x^{-1})^{-1}$ through the right identity. Replace the identity by $x^{-1}x$, reassociate, collapse $(x^{-1})^{-1}x^{-1}$ by the left inverse law, and finish with the left identity. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left inverse involutive`

Theorem 41.58 (Right inverse involutivity)**.** *For a history-level group operation with associativity, left and right identity, multiplication congruence, and right inverses, the right inverse operation is involutive:* $(x^{-1})^{-1}\mathsf{hsame}x$.

Proof. Rewrite $(x^{-1})^{-1}$ through the left identity. Replace the identity by xx^{-1}, reassociate, collapse $x^{-1}(x^{-1})^{-1}$ by the right inverse law, and finish with the right identity. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right inverse involutive`

Theorem 41.59 (Left and right inverse witnesses agree)**.** *For a history-level group operation with associativity, left and right identity, and multiplication congruence, if histories y and z satisfy*

$$\mathsf{hsame}(yx, e)$$

and

$$\mathsf{hsame}(xz, e),$$

then $\mathsf{hsame}(y, z)$.

Proof. Chain y through the right identity, replace the identity e by xz, reassociate to $(yx)z$, replace yx by e, and finish with the left identity on z. Each replacement is licensed by multiplication congruence and the displayed inverse hypotheses. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left right inverse uniqueness`

Theorem 41.60 (Group inverse congruence from laws)**.** *For a history-level group operation with associativity, left and right identity, multiplication congruence, and left and right inverses,* $\mathsf{hsame}(x, y)$ *implies* $\mathsf{hsame}(x^{-1}, y^{-1})$.

Proof. Transport the left inverse witness for x^{-1} from x to y using multiplication congruence. Combine it with the right inverse witness for y^{-1} at y, and apply uniqueness of left and right inverse witnesses. □

Lean4 ✓ `BEDC.Derived.GroupUp.group inverse congruence from laws`

Theorem 41.61 (Group inverse cancellation from laws)**.** *For a history-level group operation with associativity, left and right identity, multiplication congruence, and left and right inverses, equality of inverses under* hsame *implies equality of the original histories:*

$$\mathsf{hsame}(x^{-1}, y^{-1}) \to \mathsf{hsame}(x, y).$$

Proof. Apply the derived inverse congruence theorem to the displayed equality of inverses, yielding $\mathsf{hsame}((x^{-1})^{-1}, (y^{-1})^{-1})$. Chain the left inverse involutivity law backward at x, across this double-inverse equality, and forward at y. □

Lean4 ✓ `BEDC.Derived.GroupUp.group inverse cancel from laws`

Theorem 41.62 (Group inverse empty iff)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws, every history* $x : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}(x^{-1}, \mathsf{e}) \iff \mathsf{hsame}(x, \mathsf{e}).$$

Proof. For the forward direction, Definition 41.56 identifies e^{-1} with e. The displayed witness $\mathsf{hsame}(x^{-1}, \mathsf{e})$ therefore transports to $\mathsf{hsame}(x^{-1}, \mathsf{e}^{-1})$, and Definition 41.61 cancels inverses to obtain $\mathsf{hsame}(x, \mathsf{e})$.

For the reverse direction, apply Definition 41.60 to $\mathsf{hsame}(x, \mathsf{e})$, obtaining $\mathsf{hsame}(x^{-1}, \mathsf{e}^{-1})$, then finish again with Definition 41.56. □

Lean4 ✓ `BEDC.Derived.GroupUp.group inverse empty iff`

Theorem 41.63 (Group multiplication by a right inverse cancels)**.** *For a history-level group operation with associativity, right identity, multiplication congruence, and right inverses, every pair of histories satisfies*

$$(ab)b^{-1}\mathsf{hsame}a.$$

Proof. Reassociate $(ab)b^{-1}$ to $a(bb^{-1})$. The right inverse law collapses bb^{-1} to the identity, and the right identity law leaves a. □

Lean4 ✓ `BEDC.Derived.GroupUp.group mul right inverse cancel`

Theorem 41.64 (Group multiplication by a left inverse cancels)**.** *For a history-level group operation with associativity, left identity, multiplication congruence, and left inverses, every pair of histories satisfies*

$$a^{-1}(ab)\mathsf{hsame}b.$$

Proof. Reassociate $a^{-1}(ab)$ back to $(a^{-1}a)b$. The left inverse law collapses $a^{-1}a$ to the identity, and the left identity law leaves b. □

Lean4 ✓ `BEDC.Derived.GroupUp.group mul left inverse cancel`

Theorem 41.65 (Group inverse reverses multiplication)**.** *For a history-level group operation with associativity, left and right identity, multiplication congruence, and left and right inverses, the inverse of a product is internally the same as the reversed product of inverses:*

$$\mathsf{hsame}((xy)^{-1}, y^{-1}x^{-1}).$$

Proof. The product $y^{-1}x^{-1}$ is a right inverse of xy: reassociate $(xy)(y^{-1}x^{-1})$ to $x(y(y^{-1}x^{-1}))$, reassociate the inner product to $(yy^{-1})x^{-1}$, collapse yy^{-1} to the identity, and finish with the left identity followed by the right inverse law for x. The left inverse law for xy and uniqueness of left and right inverse witnesses identify $(xy)^{-1}$ with $y^{-1}x^{-1}$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group inverse mul reverse`

Theorem 41.66 (Group left cancellation)**.** *For a history-level group operation with associativity, left identity, multiplication congruence, and left inverses, equality of ab and ac under* hsame *implies equality of b and c under* hsame*.*

Proof. Chain b through the left identity, replace that identity by $a^{-1}a$, reassociate to $a^{-1}(ab)$, use multiplication congruence on the assumed product equality, reassociate back to $(a^{-1}a)c$, collapse the left inverse, and finish by the left identity on c. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left cancel`

Theorem 41.67 (Group left-multiplication equation solution)**.** *For a history-level group operation with associativity, left identity, multiplication congruence, and left inverses, $ax\mathsf{hsame}b$ implies $x\mathsf{hsame}a^{-1}b$.*

Proof. Chain x backward through the left identity $ex\mathsf{hsame}x$, replace e by $a^{-1}a$ using the left inverse law, reassociate to $a^{-1}(ax)$, and transport the supplied equality $ax\mathsf{hsame}b$ through multiplication congruence. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left mul equation solution`

Theorem 41.68 (Group right-multiplication equation solution)**.** *For a history-level group operation with associativity, right identity, multiplication congruence, and right inverses, $xa\mathsf{hsame}b$ implies $x\mathsf{hsame}ba^{-1}$.*

Proof. Chain x backward through the right identity $xe\mathsf{hsame}x$, replace e by aa^{-1} using the right inverse law, reassociate to $(xa)a^{-1}$, and transport the supplied equality $xa\mathsf{hsame}b$ through multiplication congruence. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right mul equation solution`

Theorem 41.69 (Group right-multiplication equation solution uniqueness)**.** *For a BEDC history-level group operation whose unit is* e*, if x and y both solve the same right-multiplication equation*

$$xa\mathsf{hsame}b \qquad \textit{and} \qquad ya\mathsf{hsame}b,$$

then $x\mathsf{hsame}y$.

Proof. Apply the right-multiplication solution theorem to both displayed equations. Both solutions are classified by the same normal form ba^{-1}. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right mul equation solution unique from empty unit`

Theorem 41.70 (Group left-multiplication equation exactness)**.** *For a history-level group operation with associativity, left identity, multiplication congruence, and two-sided inverses, left multiplication by a classifies the original equation:*

$$ax\mathsf{hsame}b \iff x\mathsf{hsame}a^{-1}b$$

Proof. The forward implication is Theorem 41.67.

For the reverse direction, transport $x\mathsf{hsame}a^{-1}b$ by left multiplication with a. Reassociate $a(a^{-1}b)$ to $(aa^{-1})b$, collapse aa^{-1} by the right inverse law, and use the left identity on b. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left mul equation exact from empty unit`

Theorem 41.71 (Group left-multiplication equation solution uniqueness)**.** *For a BEDC history-level group operation whose unit is* e*, if* x *and* y *both solve the same left-multiplication equation*

$$ax\mathsf{hsame}b \qquad \textit{and} \qquad ay\mathsf{hsame}b,$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the left-multiplication solution theorem to both displayed equations. Both solutions are classified by the same normal form $a^{-1}b$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left mul equation solution unique from empty unit`

Theorem 41.72 (Group right-multiplication equation exactness)**.** *For a history-level group operation with associativity, right identity, multiplication congruence, and two-sided inverses, right multiplication by a classifies the original equation:*

$$xa\mathsf{hsame}b \iff x\mathsf{hsame}ba^{-1}.$$

Proof. The forward implication is Theorem 41.68.

For the reverse direction, transport $x\mathsf{hsame}ba^{-1}$ by right multiplication with a. Reassociate $(ba^{-1})a$ to $b(a^{-1}a)$, collapse $a^{-1}a$ by the left inverse law, and use the right identity on b. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right mul equation exact from empty unit`

Theorem 41.73 (Group two-sided multiplication equation exactness)**.** *For a BEDC history-level group operation over* BHist *with unit* e*, associativity, left and right identities, multiplication congruence, and two-sided inverses, every* $a, c, x, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}((ax)c, b) \iff \mathsf{hsame}(x, a^{-1}(bc^{-1})).$$

The ten-field fit is: traditional target two-sided group equation solving; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability left and right equation exactness, associativity, inverse laws, and multiplication congruence; ledger the left factor, right factor, target, and two inverse witnesses; core theorem this exactness iff; standard bridge ordinary two-sided group equations; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Definition 41.72 to $\mathsf{hsame}((ax)c, b)$. This solves the right factor first and gives $\mathsf{hsame}(ax, bc^{-1})$.

Then apply Definition 41.70 to $\mathsf{hsame}(ax, bc^{-1})$. The left factor is solved exactly, yielding $\mathsf{hsame}(x, a^{-1}(bc^{-1}))$.

For the reverse direction, apply Definition 41.70 backward to $\mathsf{hsame}(x, a^{-1}(bc^{-1}))$. This recovers $\mathsf{hsame}(ax, bc^{-1})$.

Finally apply Definition 41.72 backward with right factor c. The recovered equation becomes $\mathsf{hsame}((ax)c, b)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group two sided mul equation exact from empty unit iff`

Theorem 41.74 (Group two-sided multiplication equation solution uniqueness)**.** *For a BEDC history-level group operation over* BHist *with unit* e*, associativity, left and right identities, multiplication congruence, and two-sided inverses, if x and y both solve the same two-sided multiplication equation*

$$\mathsf{hsame}((ax)c, b) \qquad \textit{and} \qquad \mathsf{hsame}((ay)c, b),$$

*then x*hsame*y. The ten-field fit is: traditional target two-sided group equation solution uniqueness; BEDC source* Group$^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability by two-sided equation exactness plus symmetry and transitivity; ledger left factor, right factor, target, and inverse witnesses; core theorem list Definition 41.73; exactness predicate same-target solution uniqueness; standard bridge ordinary two-sided group equation uniqueness; status* ProofTarget*.*

Proof. Apply Definition 41.73 to the first displayed equation. The first solution is classified by the normal form $a^{-1}(bc^{-1})$.

Apply Definition 41.73 to the second displayed equation. The second solution is classified by the same normal form $a^{-1}(bc^{-1})$.

Use symmetry on the second solved comparison and transitivity of hsame to conclude $x$hsame$y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group two sided mul equation solution unique from empty unit`

Theorem 41.75 (Group two-sided multiplication equation target-stable uniqueness)**.** *For a BEDC history-level group operation over* BHist *with unit* e*, associativity, left and right identity laws, multiplication congruence, and two-sided inverse laws, if*

$$\mathsf{hsame}((ax)c, b), \qquad \mathsf{hsame}((ay)c, d), \qquad \mathsf{hsame}(b, d),$$

*then x*hsame*y. The ten-field fit is: traditional target target-stable two-sided group equation uniqueness; BEDC source* Group$^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from two-sided equation exactness, target transport by multiplication congruence, symmetry, and transitivity; ledger left factor, right factor, two targets, and inverse witnesses; core theorem list Definition 41.73 plus multiplication congruence; exactness predicate target-stable solution uniqueness; standard bridge ordinary group equation uniqueness under equal right-hand sides; status* ProofTarget*.*

Proof. Apply Definition 41.73 to the first displayed equation. The first solution is classified by the normal form $a^{-1}(bc^{-1})$.

Apply Definition 41.73 to the second displayed equation. The second solution is classified by the normal form $a^{-1}(dc^{-1})$.

Multiplication congruence transports $\mathsf{hsame}(b, d)$ by right multiplication with the reflexive inverse c^{-1}, giving $\mathsf{hsame}(bc^{-1}, dc^{-1})$. Transport that comparison by left multiplication with a^{-1} to identify $a^{-1}(bc^{-1})$ and $a^{-1}(dc^{-1})$.

Use symmetry on the solved comparison for y and transitivity of hsame along the two normal-form comparisons to conclude $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group two sided mul equation target stable unique from empty unit`

Theorem 41.76 (Group conjugation equation exactness)**.** *For a history-level group operation with associativity, left and right identities, multiplication congruence, and two-sided inverses, conjugation by a classifies the middle factor exactly:*

$$\mathsf{hsame}((ax)a^{-1}, b) \iff \mathsf{hsame}(x, (a^{-1}b)a).$$

The ten-field fit is: traditional target conjugation equation solving; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability associativity, inverse laws, and multiplication congruence; ledger conjugation provenance; core theorem this exactness iff; standard bridge group conjugacy equations; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, transport the displayed equation by right multiplication with a. Reassociate $((ax)a^{-1})a$ to $(ax)(a^{-1}a)$, collapse $a^{-1}a$ by the left inverse law, and use the right identity to obtain $ax\mathsf{hsame}ba$.

Then transport by left multiplication with a^{-1}. Reassociate $a^{-1}(ax)$ to $(a^{-1}a)x$, collapse $a^{-1}a$, and use the left identity to obtain $x\mathsf{hsame}(a^{-1}b)a$.

For the reverse direction, transport $x\mathsf{hsame}(a^{-1}b)a$ by left multiplication with a. Reassociate $a((a^{-1}b)a)$ through $(aa^{-1})b$ and then back to ba, using the right inverse law for aa^{-1} and the left identity on b.

Finally transport by right multiplication with a^{-1}. Reassociate $(ax)a^{-1}$ as the transported left side and reassociate $(ba)a^{-1}$ to $b(aa^{-1})$, collapse aa^{-1} by the right inverse law, and finish with the right identity on b. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation equation exact from empty unit iff`

Theorem 41.77 (Group conjugation empty action)**.** *For a BEDC history-level group operation over* BHist *with left and right empty-unit laws, multiplication congruence, and the right inverse law, the empty conjugating history acts as the identity:*

$$\mathsf{hsame}((\mathsf{e}x)\mathsf{e}^{-1}, x).$$

Proof. The right inverse law for e and the left empty-unit law identify e^{-1} with e. Multiplication congruence transports this comparison and the comparison $\mathsf{e}x\mathsf{hsame}x$ through the outer product, after which the right empty-unit law collapses $x\mathsf{e}$ to x. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation empty action from empty unit`

Theorem 41.78 (Group conjugation preserves the empty unit exactly)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right empty-unit laws, multiplication congruence, and two-sided inverse laws, conjugation by a has empty target exactly when the middle factor is empty:*

$$\mathsf{hsame}((ax)a^{-1}, \mathsf{e}) \iff \mathsf{hsame}(x, \mathsf{e}).$$

Proof. Apply Definition 41.76 with target e. The resulting right side is $x\mathsf{hsame}(a^{-1}\mathsf{e})a$. Associativity rewrites $(a^{-1}\mathsf{e})a$ as $a^{-1}(\mathsf{e}a)$; the left empty-unit law reduces $\mathsf{e}a$ to a, and the left inverse law collapses $a^{-1}a$ to e. Transitivity gives both directions of the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation empty exact from empty unit`

Theorem 41.79 (Group conjugation classifier exactness)**.** *For a history-level group operation with empty unit, associativity, left and right identities, multiplication congruence, and two-sided inverses, conjugation by a preserves and reflects the* hsame *classifier:*

$$\mathsf{hsame}((ax)a^{-1}, (ay)a^{-1}) \iff \mathsf{hsame}(x, y).$$

Proof. For the forward direction, apply Definition 41.76 to the displayed equation with target $(ay)a^{-1}$. Apply the same equation exactness theorem to the reflexive equation for y at the same target. The two solved endpoints are then compared by symmetry and transitivity of hsame.

For the reverse direction, multiplication congruence transports $x\mathsf{hsame}y$ through left multiplication by a and right multiplication by a^{-1}. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation classifier exact from empty unit iff`

Theorem 41.80 (Group conjugation equation solution uniqueness)**.** *For a BEDC history-level group operation whose unit is* e*, if x and y both solve the same conjugation equation*

$$\mathsf{hsame}((ax)a^{-1}, b) \qquad \textit{and} \qquad \mathsf{hsame}((ay)a^{-1}, b),$$

then $x\mathsf{hsame}y$.

Proof. Apply conjugation equation exactness to both displayed equations. Both middle factors are classified by the same normal form $(a^{-1}b)a$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation equation solution unique from empty unit`

Theorem 41.81 (Group singleton conjugation equation solver)**.** *For singleton histories a, x, b :* BHist*, the singleton classifier solves the displayed conjugation equation exactly when the middle history is classified against the corresponding empty-left normal form:*

$$\mathsf{GroupSingletonClassifier}((ax)\mathsf{e}, b) \iff \mathsf{GroupSingletonClassifier}(x, (\mathsf{e}b)a).$$

Proof. The forward direction splits the carrier of $(ax)\mathsf{e}$ through the append-empty readback, extracting singleton carriers for a, x, and the terminal empty factor. It then rebuilds the carrier of $(\mathsf{e}b)a$ from the empty factor, b, and a, and repacks the singleton classifier by transitivity of hsame. The reverse direction performs the same carrier split on $(\mathsf{e}b)a$ and rebuilds $(ax)\mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier conjugation equation solver iff`

Theorem 41.82 (Group commutator detects commutation)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identities, multiplication congruence, and two-sided inverse laws, define*

$$\mathsf{comm}(a, b) := \mathsf{mul}(\mathsf{mul}(a, b), \mathsf{mul}(\mathsf{inv}(a), \mathsf{inv}(b))).$$

Then every a, b : BHist *satisfy*

$$\mathsf{comm}(a, b)\mathsf{hsame}e \iff \mathsf{mul}(a, b)\mathsf{hsame}\mathsf{mul}(b, a).$$

The ten-field fit is: traditional target group commutator criterion; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations inverse-product reversal, multiplication congruence, right cancellation, and inverse laws; ledger the two factors and inverse-product spine; core theorem this block; exactness predicate trivial commutator iff commutation; standard bridge ordinary group commutator criterion; status* ProofTarget*.*

Proof. Use Theorem 41.65 with the ordered product ba to identify $a^{-1}b^{-1}$ as an inverse-product presentation of $(ba)^{-1}$. Thus the displayed commutator is the product of ab with an inverse witness for ba.

For the forward direction, multiply the trivial commutator equation on the right by ba. The right side becomes ba by the left identity, while the left side reassociates to $ab((ba)^{-1}(ba))$; the left inverse law for ba collapses the final factor to e, and the right identity leaves ab. This is exactly the right-cancellation comparison between ab and ba.

For the reverse direction, transport $ab\mathsf{hsame}ba$ through multiplication congruence by the common right factor $(ba)^{-1}$. The right-hand product collapses to e by the right-inverse law, and the inverse-product reversal identification rewrites the left-hand product as $\mathsf{comm}(a, b)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group commutator trivial iff commutes from empty unit`

Theorem 41.83 (Group commutator fixed-point criterion)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identities, multiplication congruence, and two-sided inverse laws, define*

$$\mathrm{Inn}_a(b) := (a \cdot b) \cdot a^{-1} \qquad \textit{and} \qquad \mathsf{comm}(a, b) := (a \cdot b) \cdot (a^{-1} \cdot b^{-1}).$$

Then every $a, b : \mathsf{BHist}$ *satisfy*

$$\mathsf{hsame}(\mathrm{Inn}_a(b), b) \iff \mathsf{hsame}(\mathsf{comm}(a, b), e) \iff \mathsf{hsame}(a \cdot b, b \cdot a).$$

The ten-field fit is: traditional target centralizer fixed-point criterion; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations conjugation exactness, commutator exactness, associativity, inverse laws, and multiplication congruence; ledger conjugating factor, inverse witness, and commutator spine; core theorem this fixed-point equivalence; exactness predicate fixed point iff commutator trivial iff commutation; standard bridge ordinary group centralizers; status* ProofTarget*.*

Proof. Apply Theorem 41.76 with middle factor b and target b. The fixed-point assertion $\mathsf{hsame}((a \cdot b) \cdot a^{-1}, b)$ is therefore equivalent to the comparison $\mathsf{hsame}(b, (a^{-1} \cdot b) \cdot a)$.

For the forward comparison, multiply the right-side equation by a on the left and reassociate $a \cdot ((a^{-1} \cdot b) \cdot a)$ through $(a \cdot a^{-1}) \cdot b$. The inverse law collapses $a \cdot a^{-1}$ to the unit, and the unit laws identify the transported comparison with $\mathsf{hsame}(a \cdot b, b \cdot a)$.

For the reverse comparison, multiply $a \cdot b\mathsf{hsame}b \cdot a$ on the left by a^{-1}. Associativity moves the inverse next to a, the inverse and unit laws collapse the left factor, and the resulting comparison is exactly $\mathsf{hsame}(b, (a^{-1} \cdot b) \cdot a)$.

Theorem 41.82 gives the second equivalence, so transitivity of equivalences yields the displayed fixed-point, commutator, and commutation criterion. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation fixed point commutation iff from empty unit`

Definition 41.84 (Group centralizer carrier)**.** *Fix a BEDC history-level group operation* mul *over* BHist*, unit* e*, inverse operation* inv*, and fixed element* $a : \mathsf{BHist}$*. The centralizer carrier of* a *is the classifier* Z_a *on histories defined by*

$$Z_a(x) :\iff \mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x)).$$

Theorem 41.85 (Group centralizer inverse closure)**.** *For a BEDC history-level group operation* mul *over* BHist *with empty history unit* $e =$ BHist.Empty*, associativity, empty left and right unit laws, multiplication congruence, and two-sided inverse laws, the centralizer carrier of Definition 41.84 is closed under inverse:*

$$\mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x)) \Rightarrow \mathsf{hsame}(\mathsf{mul}(\mathsf{inv}(x), a), \mathsf{mul}(a, \mathsf{inv}(x))).$$

Equivalently, $Z_a(x)$ *implies* $Z_a(\mathsf{inv}(x))$*. The ten-field fit is: traditional target centralizer inverse closure; BEDC source* $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* Z_a*; classifier* hsame*; stability obligations associativity, empty unit laws, multiplication congruence, and two-sided inverse laws; ledger fixed element* a *and commuting witness for* x*; core theorem inverse closure of the centralizer carrier; exactness predicate commutation with* a*; standard bridge ordinary group centralizers; status* Proof*.*

Proof. For inverse closure, first use Definition 41.60 on the witness $xa\mathsf{hsame}ax$. Apply Definition 41.65 to both products. The inverse of xa is represented by $a^{-1}x^{-1}$, and the inverse of ax is represented by $x^{-1}a^{-1}$; transporting along the inverse-congruence witness gives $a^{-1}x^{-1}\mathsf{hsame}x^{-1}a^{-1}$.

Multiplying on the left by a and on the right by a, then using associativity and the two inverse laws, cancels the outer a^{-1} factors and yields $x^{-1}a\mathsf{hsame}ax^{-1}$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer inv closed from empty unit`

Theorem 41.86 (Group centralizer multiplication closure from empty unit)**.** *For a BEDC history-level group operation* mul *over* BHist *with associativity, right empty-unit law, and multiplication congruence, the centralizer carrier of Definition 41.84 is closed under multiplication:*

$$\mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x)) \Rightarrow \mathsf{hsame}(\mathsf{mul}(y, a), \mathsf{mul}(a, y)) \Rightarrow \mathsf{hsame}(\mathsf{mul}(\mathsf{mul}(x, y), a), \mathsf{mul}(a, \mathsf{mul}(x, y))).$$

Proof. The empty-context multiplication closure compares the two target products after multiplying both by e on the right. The concrete right empty-unit law identifies each right-extended product with its unextended endpoint, and transitivity removes the empty context from the comparison. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer mul closed from empty unit`

Theorem 41.87 (Group centralizer subgroup and commutator fiber)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws, the carrier* Z_a *of Definition 41.84 is closed under the group operations:*

1. $Z_a(e)$*.*
2. *If* $Z_a(x)$ *and* $Z_a(y)$*, then* $Z_a(\mathsf{mul}(x, y))$*.*
3. *If* $Z_a(x)$*, then* $Z_a(\mathsf{inv}(x))$*.*
4. *For every* $x : \mathsf{BHist}$*,*

$$Z_a(x) \iff \mathsf{comm}(x, a)\mathsf{hsame}e,$$

where comm *is the commutator operation of Definition 41.82.*

The ten-field fit is: traditional target centralizer subgroup; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* Z_a*; classifier* hsame*; stability obligations unit, product, inverse closure, and commutator-fiber exactness; ledger fixed element* a *and commuting witnesses for the active factors; core theorem list Definition 41.65, Definition 41.60, and Definition 41.82, together with associativity, unit laws, inverse laws, and multiplication congruence; exactness predicate* $Z_a(x)$ *iff trivial commutator; standard bridge ordinary group centralizers; status* ProofTarget*.*

Proof. The unit belongs to Z_a because the left and right identity laws identify ea and ae with the same history a. Symmetry and transitivity of hsame close the comparison.

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer empty unit mem`

For product closure, start from witnesses $x a \mathsf{hsame} a x$ and $y a \mathsf{hsame} a y$. Associativity rewrites $(xy)a$ as $x(ya)$; multiplication congruence transports the second witness to $x(ay)$, and associativity rewrites this as $(xa)y$. The first witness then transports the left factor to $(ax)y$, and associativity gives $a(xy)$.

For inverse closure, apply Definition 41.85 to the witness $Z_a(x)$.

The commutator fiber is exactly the displayed carrier by Definition 41.82 with the ordered pair (x, a). Its right side is precisely the definition of $Z_a(x)$, and symmetry of the displayed equivalence supplies the stated orientation. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer mul closed empty context`

Theorem 41.88 (Group singleton terminal classifier package)**.** *For histories $h, k, p, q : \mathsf{BHist}$, the concrete singleton $\mathsf{Group}^{\uparrow}$ instance of Definition 41.3 carries the terminal classifier package:*

1. *The classifier is exactly the product carrier relation:*

$$R_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

2. *With $\mathsf{mul} = \mathsf{append}$, multiplication is exactly closed on the singleton carrier:*

$$A_{\mathsf{e}}(\mathsf{mul}(p, q)) \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).$$

3. *If $A_{\mathsf{e}}(h)$, then the constant inverse has both endpoint classifications:*

$$A_{\mathsf{e}}(\mathsf{inv}(h)) \wedge R_{\mathsf{e}}(\mathsf{inv}(h), h) \wedge R_{\mathsf{e}}(h, \mathsf{inv}(h)).$$

The per-object template names the traditional target as the terminal group; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability as hsame symmetry/transitivity and append-empty exactness; ledger the hidden singleton endpoint witnesses; core theorem this block; exactness predicate the displayed iff rows; and status ProofTarget.

Proof. Unfold R_{e} and A_{e}. For the first row, the forward direction forgets only the displayed $\mathsf{hsame}(h, k)$ witness. Conversely, singleton carrier witnesses identify h and k with e; symmetry and transitivity of hsame compose those endpoint identifications through e and supply the classifier comparison.

For the second row, use Definition 41.3 to replace mul by append. From $A_{\mathsf{e}}(p)$ and $A_{\mathsf{e}}(q)$, append empty-result construction gives $A_{\mathsf{e}}(\mathsf{append}(p, q))$. In the reverse direction, append empty-result inversion splits the singleton endpoint witness for $\mathsf{append}(p, q)$ into the two factor witnesses.

For the third row, Definition 41.3 gives the definitional equation $\mathsf{inv}(h) = \mathsf{e}$. Hence $A_{\mathsf{e}}(\mathsf{inv}(h))$ is the reflexive singleton endpoint witness, and the first row applied to $\mathsf{inv}(h)$ and h packages both classifier orientations. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingleton terminal classifier package`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier terminal package`

Theorem 41.89 (Group singleton two-sided contextual classifier coverage)**.** *For histories* L, R, L', R', Q, S : BHist*, if* $A_{\mathsf{e}}(L)$*,* $A_{\mathsf{e}}(R)$*,* $A_{\mathsf{e}}(L')$*, and* $A_{\mathsf{e}}(R')$*, then*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, Q), L'), \mathsf{append}(\mathsf{append}(R, S), R')) \iff A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(S).$$

The ten-field fit names the traditional target terminal-group contextual endpoint coverage; the BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *with append; carrier* A_{e}*; classifier* R_{e}*; stability from two-sided context cancellation and terminal classifier coverage; ledger hidden singleton prefix/suffix endpoints; exactness predicate the displayed iff; standard bridge terminal group contextual equality; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Theorem 41.9 with the four displayed singleton context witnesses. This identifies the context-wrapped classifier with the endpoint classifier $R_{\mathsf{e}}(Q, S)$.

Apply the first row of Theorem 41.88 to replace $R_{\mathsf{e}}(Q, S)$ by the pair of singleton carrier facts $A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(S)$. Composing the two equivalences gives exactly the displayed contextual classifier coverage. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual coverage iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual carrier coverage iff`

Theorem 41.90 (Group singleton continuation terminal collapse)**.** *For histories* P, Q, R, r : BHist *with* $\mathsf{Cont}(P, Q, R)$*, the concrete singleton* $\mathsf{Group}^{\uparrow}$ *continuation collapses the input classifier, both result classifiers, and the result carrier to one terminal condition:*

$$R_{\mathsf{e}}(P, Q) \iff R_{\mathsf{e}}(R, P) \iff R_{\mathsf{e}}(R, Q) \iff A_{\mathsf{e}}(R).$$

Moreover, under these equivalent conditions, both visible continuation results are excluded:

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

The per-object template names the traditional target terminal-group continuation collapse; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus* Cont*; carrier* A_{e}*; classifier* R_{e}*; stability from continuation result transport and empty-result inversion; ledger hidden continuation result endpoint; core theorem this block; exactness predicate the four-way equivalence; and status* $\mathsf{ProofTarget}$*.*

Proof. Use Theorem 41.17 to identify the input classifier $R_{\mathsf{e}}(P, Q)$ with the left result classifier $R_{\mathsf{e}}(R, P)$ under the displayed continuation witness.

Use Theorem 41.19 to identify the same input classifier $R_{\mathsf{e}}(P, Q)$ with the right result classifier $R_{\mathsf{e}}(R, Q)$. Then use Theorem 41.20 to identify $R_{\mathsf{e}}(P, Q)$ with the result carrier condition $A_{\mathsf{e}}(R)$. These three exactness statements give the four-way equivalence.

Under any of the equivalent conditions, take the input-classifier presentation $R_{\mathsf{e}}(P, Q)$. Theorem 41.16 then excludes both visible results $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$ for continuations from P through Q. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation terminal collapse`

Theorem 41.91 (Group singleton continuation product-unit triple coverage)**.** *For histories* P, Q, S, T : BHist *with* $\mathsf{Cont}(P, Q, S)$*, the concrete singleton* $\mathsf{Group}^{\uparrow}$ *continuation and append product-unit classifier give the three endpoint coverage equivalence:*

$$R_{\mathsf{e}}(\mathsf{append}(S, T), \mathsf{e}) \iff A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T).$$

The ten-field fit names the traditional target terminal-group continuation product-unit coverage; the BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus* Cont *and append; carrier* A_{e}*; classifier* R_{e}*; stability from continuation terminal collapse and product-unit split; ledger continuation witness, product suffix, and hidden endpoint witnesses; core theorem list Definition 41.10, Definition 41.90, and Definition 41.88; exactness the displayed iff; and status* $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.10 to rewrite the left side as the product of the continuation result carrier and suffix carrier:

$$A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T).$$

Use Definition 41.90 under the displayed continuation witness to identify $A_{\mathsf{e}}(S)$ with $R_{\mathsf{e}}(P, Q)$. This changes the intermediate condition to $R_{\mathsf{e}}(P, Q) \wedge A_{\mathsf{e}}(T)$.

Finally, apply the first row of Definition 41.88 to rewrite $R_{\mathsf{e}}(P, Q)$ as $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q)$. Reassociating conjunctions gives exactly

$$A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T).$$

□

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation product unit coverage iff`

Theorem 41.92 (Group singleton contextual continuation terminal collapse)**.** *For histories* $L, R, L', R', P, Q, S, r : \mathsf{BHist}$*, suppose* $A_{\mathsf{e}}(L)$*,* $A_{\mathsf{e}}(R)$*,* $A_{\mathsf{e}}(L')$*,* $A_{\mathsf{e}}(R')$*, and* $\mathsf{Cont}(P, Q, S)$*. Then the singleton contextual classifier collapses to the same terminal continuation conditions:*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, P), L'), \mathsf{append}(\mathsf{append}(R, Q), R')) \iff R_{\mathsf{e}}(P, Q) \iff R_{\mathsf{e}}(S, P) \iff R_{\mathsf{e}}(S, Q) \iff A_{\mathsf{e}}(S).$$

Moreover, under these equivalent conditions, both visible continuation results are excluded:

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

The ten-field fit names the traditional target terminal-group contextual continuation collapse; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus* Cont*; carrier* A_{e}*; classifier* R_{e}*; stability from two-sided context cancellation and continuation terminal collapse; ledger hidden prefix/suffix endpoint witnesses; core theorem this block; exactness predicate the contextual terminal equivalence; standard bridge terminal-group contextual continuation collapse; and status* $\mathsf{ProofTarget}$.

Proof. Apply Theorem 41.9 with the four singleton carrier witnesses for L, R, L', R'. This strips the hidden prefix and suffix contexts from the displayed contextual classifier and identifies it with the uncontexted classifier $R_{\mathsf{e}}(P, Q)$.

Apply Theorem 41.90 to the continuation witness $\mathsf{Cont}(P, Q, S)$. It identifies $R_{\mathsf{e}}(P, Q)$ with $R_{\mathsf{e}}(S, P)$, $R_{\mathsf{e}}(S, Q)$, and $A_{\mathsf{e}}(S)$, and it gives the two implications excluding $\mathsf{Cont}(P, Q, \mathsf{E0}(r))$ and $\mathsf{Cont}(P, Q, \mathsf{E1}(r))$. Composing this terminal collapse with the contextual cancellation gives the displayed theorem. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier contextual continuation terminal collapse`

41.9 Singleton contextual continuation obstruction

Theorem 41.93 (Group singleton contextual continuation visible-result exclusion)**.** *For histories* $L, R, L', R', P, Q, r : \mathsf{BHist}$*, suppose* $A_{\mathsf{e}}(L)$*,* $A_{\mathsf{e}}(R)$*,* $A_{\mathsf{e}}(L')$*,* $A_{\mathsf{e}}(R')$*, and*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, P), L'), \mathsf{append}(\mathsf{append}(R, Q), R')).$$

Then singleton hidden contexts obstruct both visible continuation results:

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

The ten-field fit is: traditional target terminal-group contextual continuation obstruction; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus* Cont*; carrier* A_{e}*; classifier* R_{e}*; stability from two-sided context cancellation plus visible-result exclusion; ledger hidden prefix and suffix singleton contexts; core theorem list Theorem 41.9 and Theorem 41.16; exactness predicate the displayed implication pair; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply the forward direction of Theorem 41.9 to the displayed contextual classifier, using the four singleton carrier witnesses for L, R, L', R'. This exposes the uncontexted classifier $R_{\mathsf{e}}(P, Q)$.

Apply Theorem 41.16 to that classifier. It gives exactly the two implications excluding visible results $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$ for continuations from P through Q. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier contextual continuation visible result absurd`

41.10 Centralizer transport

Theorem 41.94 (Group centralizer fixed-element transport)**.** *For a BEDC history-level group operation over* BHist *with multiplication congruence, let* Z_a *and* Z_b *be the centralizer carriers of Definition 41.84 for fixed elements* $a, b : \mathsf{BHist}$*. If* $\mathsf{hsame}(a, b)$*, then for every* $x : \mathsf{BHist}$*,*

$$Z_a(x) \Longleftrightarrow Z_b(x).$$

The ten-field fit is: traditional target transport of centralizers along internal equality of the fixed element; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier the family* Z_a*; classifier* hsame*; stability obligation multiplication congruence in the fixed-element slot; ledger the fixed-element comparison; core theorem list Definition 41.84, multiplication congruence, and* hsame *symmetry and transitivity; exactness predicate the displayed iff; standard bridge ordinary equality-invariant centralizers; status* $\mathsf{ProofTarget}$*.*

Proof. Unfold the centralizer carrier. The assertion $Z_a(x)$ is the comparison $\mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x))$, and $Z_b(x)$ is the comparison $\mathsf{hsame}(\mathsf{mul}(x, b), \mathsf{mul}(b, x))$.

For the forward direction, assume $Z_a(x)$. Symmetry turns $\mathsf{hsame}(a, b)$ into $\mathsf{hsame}(b, a)$, and multiplication congruence in the fixed-element slot transports it to $\mathsf{hsame}(\mathsf{mul}(x, b), \mathsf{mul}(x, a))$. Transitivity with the $Z_a(x)$ witness gives $\mathsf{hsame}(\mathsf{mul}(x, b), \mathsf{mul}(a, x))$.

Multiplication congruence also transports $\mathsf{hsame}(a, b)$ through the left fixed-element slot, giving $\mathsf{hsame}(\mathsf{mul}(a, x), \mathsf{mul}(b, x))$. A final transitivity step therefore yields $\mathsf{hsame}(\mathsf{mul}(x, b), \mathsf{mul}(b, x))$, which is $Z_b(x)$.

For the reverse direction, exchange a and b. The same calculation starts from $Z_b(x)$, uses symmetry of $\mathsf{hsame}(a, b)$ where the fixed element is transported from b back to a, and concludes with $\mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x))$, namely $Z_a(x)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer fixed element empty context transport iff`

Definition 41.95 (Group centralizer normalizer carrier)**.** *Let Z_a be the centralizer carrier of Definition 41.84. For each history $s : \mathsf{BHist}$ define the conjugation word*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

The centralizer normalizer carrier N_a is the predicate

$$N_a(s) :\Longleftrightarrow (\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow Z_a(\mathsf{Conj}_s(x))) \wedge (\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))).$$

Thus $N_a(s)$ records centralizer preservation by conjugation with s and by conjugation with $\mathsf{inv}(s)$.

Theorem 41.96 (Group conjugation composition)**.** *For a BEDC history-level group operation over BHist with associativity, left and right identity laws for e, multiplication congruence, and two-sided inverse laws, inner conjugation composes in the expected order:*

$$\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)\mathsf{hsameConj}_s(\mathsf{Conj}_t(x)).$$

Moreover the inverse conjugating word for $\mathsf{mul}(s, t)$ is represented by the reversed inverse product $\mathsf{mul}(\mathsf{inv}(t), \mathsf{inv}(s))$. The ten-field fit is: traditional target inner-action composition; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations inverse-product reversal, reassociation, and multiplication congruence; ledger the ordered pair s, t and the acted-on history x; core theorem list Definition 41.65 and Definition 41.79, together with associativity, unit laws, inverse laws, and multiplication congruence; standard bridge ordinary inner automorphism composition; status $\mathsf{ProofTarget}$.

Proof. Expand both conjugation words. The left side is $\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(s, t), x), \mathsf{inv}(\mathsf{mul}(s, t)))$.

Apply Definition 41.65 to the product $\mathsf{mul}(s, t)$. It replaces the final inverse by $\mathsf{mul}(\mathsf{inv}(t), \mathsf{inv}(s))$, with the order reversed.

Associativity then moves parentheses through $((st)x)(t^{-1}s^{-1})$ until the word is $s((tx)t^{-1})s^{-1}$. Multiplication congruence transports each reassociation and the inverse-product comparison through the outer products. The final word is exactly $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupConjugationWord product composition`
Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation composition from empty unit`
Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation mul composition from empty unit`

Theorem 41.97 (Group normalizer conjugation action composition)**.** *For a BEDC history-level group operation over BHist with the same associativity, left and right identity laws for e, multiplication congruence, and two-sided inverse laws as in Definition 41.87, assume $N_a(s)$, $N_a(t)$, and $Z_a(x)$. Then the normalizer conjugation action composes on Z_a:*

1. $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.
2. $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), \mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.
3. $Z_a(\mathsf{Conj}_{\mathsf{inv}(t)}(\mathsf{Conj}_{\mathsf{inv}(s)}(x)))$.
4. $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{inv}(\mathsf{mul}(s,t))}(x), \mathsf{Conj}_{\mathsf{inv}(t)}(\mathsf{Conj}_{\mathsf{inv}(s)}(x)))$.

The ten-field fit is: traditional target normalizer conjugation action; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist; carrier Z_a; classifier hsame; stability action composition and inverse-action composition; ledger conjugating pair and inverse-product witness; core theorem list Definition 41.65, Definition 41.79, and associativity, unit, inverse, and congruence laws; exactness predicate preservation of Z_a; standard bridge ordinary normalizer action; status $\mathsf{ProofTarget}$.

Proof. Unfold Conj. Associativity reassociates the product word $\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(s,t),x),\mathsf{inv}(\mathsf{mul}(s,t)))$ to the nested conjugation word $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$, after Definition 41.65 identifies the inverse factor with $\mathsf{mul}(\mathsf{inv}(t),\mathsf{inv}(s))$.

The classifier comparison is transported through the surrounding multiplications by multiplication congruence. Then Definition 41.79 reads the reassociated word as the same Z_a-classifier witness, giving the displayed hsame statement for $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$.

For preservation, apply the forward clause of $N_a(t)$ to $Z_a(x)$, obtaining $Z_a(\mathsf{Conj}_t(x))$. Apply the forward clause of $N_a(s)$ to that result, giving $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.

For the inverse-action word, use the inverse clause of $N_a(s)$ first and the inverse clause of $N_a(t)$ second. This gives $Z_a(\mathsf{Conj}_{\mathsf{inv}(t)}(\mathsf{Conj}_{\mathsf{inv}(s)}(x)))$.

Finally apply the same reassociation and congruence calculation to the inverse-product word. Theorem 41.65 identifies $\mathsf{inv}(\mathsf{mul}(s,t))$ with $\mathsf{mul}(\mathsf{inv}(t),\mathsf{inv}(s))$, so the classifier exactness theorem supplies the displayed hsame comparison for the inverse-action composition. □

Lean4 ✓ `BEDC.Derived.GroupUp.group normalizer conjugation action composition from empty unit`
Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer forward action composition from empty unit`

Theorem 41.98 (Group centralizer normalizer subgroup)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, and two-sided inverse laws as in Definition 41.87, the carrier N_a of Definition 41.95 is the two-sided normalizer subgroup of Z_a:*

1. $N_a(e)$.
2. *If* $N_a(s)$ *and* $N_a(t)$*, then* $N_a(\mathsf{mul}(s,t))$.
3. *If* $N_a(s)$*, then* $N_a(\mathsf{inv}(s))$.
4. *If* $N_a(s)$ *and* $\mathsf{hsame}(s,t)$*, then* $N_a(t)$.
5. *If* $Z_a(s)$*, then* $N_a(s)$.

The ten-field fit is: traditional target normalizer of a subgroup; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier N_a; classifier* hsame*; stability conjugation closure in both directions; ledger conjugating element and inverse witness; exactness normalizer action on the centralizer; core theorem list Definition 41.87, Definition 41.79, and Definition 41.65, together with associativity, unit laws, inverse laws, and multiplication congruence; standard bridge ordinary subgroup normalizer; status* $\mathsf{ProofTarget}$.

Proof. For the unit row, the unit laws identify $\mathsf{Conj}_e(x)$ with x for every x. The inverse laws identify $\mathsf{inv}(e)$ with e, so the second half of the carrier is the same calculation. Hence any $Z_a(x)$ is returned to Z_a by both unit conjugation clauses.

For product closure, assume $N_a(s)$ and $N_a(t)$. Associativity rewrites $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$ as $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$ by Definition 41.96. If $Z_a(x)$, first apply the forward normalizer clause for t and then the forward clause for s.

For the inverse-conjugation half of the product, use the same conjugation composition reading with the reversed inverse word $\mathsf{mul}(\mathsf{inv}(t),\mathsf{inv}(s))$. Starting from $Z_a(x)$, the inverse clause for s gives preservation by $\mathsf{inv}(s)$, and the inverse clause for t then gives preservation by $\mathsf{inv}(t)$. The classifier comparison with $\mathsf{Conj}_{\mathsf{inv}(\mathsf{mul}(s,t))}(x)$ is supplied by Definition 41.65, Definition 41.96, and multiplication congruence.

For inverse closure, the first clause of $N_a(\mathsf{inv}(s))$ is exactly the second clause of $N_a(s)$. The second clause of $N_a(\mathsf{inv}(s))$ conjugates by $\mathsf{inv}(\mathsf{inv}(s))$, which the inverse laws identify with s under

hsame; multiplication congruence and Definition 41.79 transport the existing first clause of $N_a(s)$ across that classifier equality.

For classifier transport, assume $N_a(s)$ and $\mathsf{hsame}(s,t)$. Multiplication congruence transports each word $\mathsf{mul}(\mathsf{mul}(s,x),\mathsf{inv}(s))$ to the corresponding word with t, using the inverse-congruence consequence of the group laws for the last factor. The classifier exactness theorem then shows that the transported conjugation has the same Z_a membership, so the two clauses defining $N_a(s)$ give the two clauses defining $N_a(t)$.

Finally assume $Z_a(s)$. By the inverse row of Definition 41.87, $Z_a(\mathsf{inv}(s))$ also holds. For any $Z_a(x)$, the product row first gives $Z_a(\mathsf{mul}(s,x))$ and then $Z_a(\mathsf{mul}(\mathsf{mul}(s,x),\mathsf{inv}(s)))$.

The same argument with $\mathsf{inv}(s)$ as the first centralizer element and s as the final one proves the conjugation clause for $\mathsf{inv}(s)$. These two preservation clauses are exactly $N_a(s)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer subgroup from empty unit`

Theorem 41.99 (Centralizer normal in its normalizer)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, and two-sided inverse laws as in Definition 41.87, the centralizer carrier Z_a of Definition 41.84 forms a normal subgroup package inside the normalizer carrier N_a of Definition 41.95. Namely:*

1. *The identity, product, inverse, and* hsame*-transport subgroup rows for Z_a are those of Definition 41.87.*
2. *If $Z_a(s)$, then $N_a(s)$.*
3. *If $N_a(s)$ and $Z_a(x)$, then $Z_a(\mathsf{Conj}_s(x))$.*
4. *If $N_a(s)$ and $Z_a(x)$, then $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.*
5. *If $\mathsf{hsame}(s,t)$, then the two displayed conjugation-closure rows transport from s to t by the classifier comparison for the corresponding conjugation words.*

Proof. The subgroup data are exactly the first four rows of Definition 41.87. The inclusion $Z_a(s) \Rightarrow N_a(s)$ is the final row of Definition 41.98, so the carrier inclusion needed for a normal-subgroup package is already part of the normalizer subgroup theorem.

It remains to identify the conjugation action of N_a on Z_a. Unpack Definition 41.95. Its first clause says that every $Z_a(x)$ is sent to $Z_a(\mathsf{Conj}_s(x))$, and its second clause says that every $Z_a(x)$ is sent to $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$. These are precisely the two normality closure rows.

For classifier transport, assume $\mathsf{hsame}(s,t)$. Multiplication congruence transports the word $\mathsf{mul}(\mathsf{mul}(s,x),\mathsf{inv}(s))$ to $\mathsf{mul}(\mathsf{mul}(t,x),\mathsf{inv}(t))$, using inverse congruence for the last factor. Applying Definition 41.79 identifies the corresponding conjugation classifiers, and the hsame-transport row of Definition 41.87 carries Z_a membership across the comparison.

The same multiplication-congruence and inverse-congruence calculation with $\mathsf{inv}(s)$ and $\mathsf{inv}(t)$ gives transport for the inverse conjugation row. Thus centralizer membership is closed under conjugation by normalizer elements and by their inverse conjugation words, with the closure stable under the ambient history classifier. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normal in normalizer from empty unit`

Theorem 41.100 (Group centralizer normalizer action automorphism and kernel exactness)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98,*

fix $a : \mathsf{BHist}$. *Let* Z_a *be the centralizer carrier of Definition 41.84, let* N_a *be the normalizer carrier of Definition 41.95, and write*

$$\mathrm{Act}_s(x) := \mathsf{Conj}_s(x).$$

If $N_a(s)$ *and* $Z_a(x)$*, then conjugation by* s *restricts to a classifier-exact automorphism of* Z_a *with inverse action by* $\mathsf{inv}(s)$*:*

1. $Z_a(\mathrm{Act}_s(x))$ *and* $Z_a(\mathrm{Act}_{\mathsf{inv}(s)}(x))$.
2. $\mathsf{hsame}(\mathrm{Act}_{\mathsf{inv}(s)}(\mathrm{Act}_s(x)), x)$.
3. $\mathsf{hsame}(\mathrm{Act}_s(\mathrm{Act}_{\mathsf{inv}(s)}(x)), x)$.

For every s *with* $N_a(s)$*, the action kernel is exact:*

$$(\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow \mathsf{hsame}(\mathrm{Act}_s(x), x)) \Longleftrightarrow (\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow \mathsf{hsame}(\mathsf{mul}(s, x), \mathsf{mul}(x, s))).$$

The ten-field fit is: traditional target normalizer action by automorphisms and action kernel; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* Z_a*; classifier* hsame*; stability normalizer preservation, inverse action, and fixed-point exactness; ledger fixed element, normalizer witness, acted element, inverse parameter, and centralizing witness; core theorem list Definition 41.98, Definition 41.79, Definition 41.65, and Definition 41.83, together with the group associativity, unit, inverse, multiplication-congruence, and inverse-congruence laws; exactness predicate pointwise fixed action iff pointwise centralizing; standard bridge ordinary normalizer action on a subgroup; status* $\mathsf{ProofTarget}$.

Proof. The two preservation rows come directly from the defining clauses of N_a. The first clause sends $Z_a(x)$ to $Z_a(\mathrm{Act}_s(x))$, and the second clause sends $Z_a(x)$ to $Z_a(\mathrm{Act}_{\mathsf{inv}(s)}(x))$. The inverse-closure row of Definition 41.98 records the same preservation with the inverse parameter as a normalizer witness.

For the first inverse-action identity, expand $\mathrm{Act}_{\mathsf{inv}(s)}(\mathrm{Act}_s(x))$ as the conjugation word with parameter $\mathsf{inv}(s)$ applied to the conjugation word with parameter s. Associativity regroups the product into the adjacent inverse factors; the two-sided inverse laws reduce those factors to e, and the left and right unit laws reduce the remaining word to x. Multiplication congruence and Definition 41.79 carry this calculation through hsame.

The second inverse-action identity is symmetric. Expand $\mathrm{Act}_s(\mathrm{Act}_{\mathsf{inv}(s)}(x))$, regroup by associativity, reduce the adjacent factors $\mathsf{mul}(s, \mathsf{inv}(s))$ and $\mathsf{mul}(\mathsf{inv}(s), s)$ by the inverse laws, and use the unit laws to obtain x under hsame. The inverse-product reversal theorem supplies the same ordered-word comparison when the composite is read through the normalizer-action product row.

For kernel exactness, fix x with $Z_a(x)$. The fixed-point criterion Definition 41.83 identifies the statement $\mathsf{hsame}(\mathrm{Act}_s(x), x)$ with the pointwise centralizing statement $\mathsf{hsame}(\mathsf{mul}(s, x), \mathsf{mul}(x, s))$.

Quantifying that equivalence over all elements of Z_a gives the displayed kernel clause. No additional carrier is introduced: the kernel predicate is the pointwise fixed-action predicate for the restricted normalizer action. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer action automorphism kernel exactness from empty unit`

Theorem 41.101 (Group centralizer normalizer conjugation action)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for* e*, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

If $N_a(s)$*,* $N_a(t)$*, and* $Z_a(x)$*, then the induced conjugation action on the centralizer carrier has the following rows:*

1. $Z_a(\mathsf{Conj}_s(x))$.
2. $\mathsf{hsame}(\mathsf{Conj}_e(x), x)$.
3. $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$ *and* $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), \mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.
4. *If* $\mathsf{hsame}(s,t)$, *then* $\mathsf{hsame}(\mathsf{Conj}_s(x), \mathsf{Conj}_t(x))$.

Proof. The carrier preservation row is the first clause in the definition of $N_a(s)$: starting from $Z_a(x)$, conjugation by s returns $Z_a(\mathsf{Conj}_s(x))$.

For the identity action, expand $\mathsf{Conj}_e(x)$ as $\mathsf{mul}(\mathsf{mul}(e,x), \mathsf{inv}(e))$. The inverse laws identify $\mathsf{inv}(e)$ with e, and the left and right unit laws then identify the whole word with x under hsame.

For composition, first apply the forward clause of $N_a(t)$ to $Z_a(x)$, giving $Z_a(\mathsf{Conj}_t(x))$. Applying the forward clause of $N_a(s)$ to that result gives $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.

The classifier comparison is the product-closure calculation from Definition 41.98. Associativity moves the parentheses in the fourfold word, while Definition 41.65 identifies the inverse of $\mathsf{mul}(s,t)$ with $\mathsf{mul}(\mathsf{inv}(t), \mathsf{inv}(s))$; multiplication congruence then yields $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), \mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.

Finally assume $\mathsf{hsame}(s,t)$. Inverse congruence gives $\mathsf{hsame}(\mathsf{inv}(s), \mathsf{inv}(t))$, and two uses of multiplication congruence carry the classifier equality through $\mathsf{mul}(\mathsf{mul}(_, x), \mathsf{inv}(_))$. Hence $\mathsf{Conj}_s(x)$ and $\mathsf{Conj}_t(x)$ are hsame-equal. □

Lean4 ✓ `BEDC.Derived.GroupUp.group conjugation parameter hsame congr from empty unit`

Theorem 41.102 (Group centralizer normalizer action inverse)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s,x), \mathsf{inv}(s)).$$

If $N_a(s)$ and $Z_a(x)$, then the normalizer action restricts to a classifier equivalence on the centralizer:

1. $Z_a(\mathsf{Conj}_s(x))$.
2. $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)))$.
3. $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), x)$.
4. $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.
5. $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x)))$.
6. $\mathsf{hsame}(\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x)), x)$.

The ten-field per-object fit is: traditional target normalizer action by equivalences; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier Z_a; classifier* hsame*; stability inverse action, identity action, and composition; ledger normalizer witness, inverse witness, and acted-on centralizer element; core theorem list Definition 41.98, Definition 41.97, and Definition 41.101, together with the inverse and unit laws; exactness predicate restricted action invertibility on Z_a; standard bridge ordinary normalizer action by automorphisms; status* ProofTarget.

Proof. The first and fourth rows are the two carrier clauses in the definition of $N_a(s)$: the normalizer witness sends a centralizer element forward by Conj_s and backward by $\mathsf{Conj}_{\mathsf{inv}(s)}$. The inverse-closure row of Definition 41.98 also records $N_a(\mathsf{inv}(s))$.

Using $N_a(\mathsf{inv}(s))$ with $Z_a(\mathsf{Conj}_s(x))$, the forward preservation row of Definition 41.101 gives $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)))$. Using $N_a(s)$ with $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$ gives $Z_a(\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x)))$.

For the left inverse identity, the composition row of Definition 41.97 identifies $\mathsf{Conj}_{\mathsf{mul}(\mathsf{inv}(s),s)}(x)$ with $\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x))$. The two-sided inverse law identifies $\mathsf{mul}(\mathsf{inv}(s), s)$ with e, and the identity row of Definition 41.101 identifies $\mathsf{Conj}_e(x)$ with x under hsame.

For the right inverse identity, the same composition row identifies $\mathsf{Conj}_{\mathsf{mul}(s,\mathsf{inv}(s))}(x)$ with $\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$. The other two-sided inverse law reduces $\mathsf{mul}(s, \mathsf{inv}(s))$ to e, and the identity row again gives the comparison with x. Multiplication congruence and inverse congruence supply the classifier transport at each product replacement. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer action inverse from empty unit`

41.11 Centralizer normalizer orbit relation

Definition 41.103 (Group centralizer normalizer orbit relation)**.** *Let Z_a be the centralizer carrier of Definition 41.84, let N_a be the normalizer carrier of Definition 41.95, and let*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

For carried centralizer histories $x, y : \mathsf{BHist}$, define the normalizer-orbit relation by

$$x \sim_{N_a} y :\Longleftrightarrow \exists s : \mathsf{BHist},\ N_a(s) \wedge Z_a(x) \wedge Z_a(y) \wedge \mathsf{hsame}(\mathsf{Conj}_s(x), y).$$

This is a quotient-free classifier of reachability inside Z_a by conjugation with histories carried by N_a.

Theorem 41.104 (Group centralizer normalizer orbit equivalence)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let Z_a, N_a, and Conj_s be as in Definition 41.103. For $x, y, z : \mathsf{BHist}$ with $Z_a(x)$, $Z_a(y)$, and $Z_a(z)$, the relation $\sim_{N_a}$ is reflexive, symmetric, and transitive on the centralizer carrier:*

1. *$x \sim_{N_a} x$.*
2. *If $x \sim_{N_a} y$, then $y \sim_{N_a} x$.*
3. *If $x \sim_{N_a} y$ and $y \sim_{N_a} z$, then $x \sim_{N_a} z$.*

The ten-field fit is: traditional target group-action orbit relation on a centralizer; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier Z_a; classifier $\sim_{N_a}$; stability reflexive, symmetric, and transitive; ledger source endpoint x; target endpoint y; intermediate endpoint z; conjugator s; inverse conjugator* $\mathsf{inv}(s)$*; product conjugator* $\mathsf{mul}(t, s)$*; core theorem list Definition 41.98, Definition 41.101, and Definition 41.96; exactness orbit reachability on centralizer elements; standard bridge ordinary orbit relation for normalizer action; status* $\mathsf{ProofTarget}$.

Proof. For reflexivity, use the identity row of Definition 41.98 to obtain $N_a(e)$. The identity action row of Definition 41.101 gives $\mathsf{hsame}(\mathsf{Conj}_e(x), x)$, so e is the orbit witness for $x \sim_{N_a} x$ together with the carried endpoint $Z_a(x)$.

For symmetry, unpack a witness s for $x \sim_{N_a} y$. Inverse closure in Definition 41.98 gives $N_a(\mathsf{inv}(s))$, and the inverse-conjugation clause in the definition of N_a keeps the conjugated history inside Z_a.

The composition row of Definition 41.101, the inverse laws, and transitivity of hsame identify conjugating back by $\mathsf{inv}(s)$ with the original source endpoint. Classifier transport across the displayed witness $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$ then replaces the acted-on endpoint by y, producing a witness for $y \sim_{N_a} x$.

For transitivity, unpack witnesses s for $x \sim_{N_a} y$ and t for $y \sim_{N_a} z$. Product closure in Definition 41.98 gives $N_a(\mathsf{mul}(t, s))$.

The composition row of Definition 41.101 compares $\mathsf{Conj}_{\mathsf{mul}(t,s)}(x)$ with $\mathsf{Conj}_t(\mathsf{Conj}_s(x))$. The first witness transports $\mathsf{Conj}_s(x)$ to y, and the second witness transports $\mathsf{Conj}_t(y)$ to z. Transitivity of hsame combines these comparisons, while $Z_a(x)$ and $Z_a(z)$ supply the carried endpoints, so $\mathsf{mul}(t, s)$ witnesses $x \sim_{N_a} z$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer orbit equivalence from empty unit`

Theorem 41.105 (Group centralizer normalizer orbit hsame lift)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws as in Definition 41.104, fix* $a : \mathsf{BHist}$*. Let* Z_a*,* N_a*,* Conj_s*, and* $\sim_{N_a}$ *be as in Definition 41.103. For carried centralizer histories* $x, y : \mathsf{BHist}$*,*

$$Z_a(x) \wedge Z_a(y) \wedge \mathsf{hsame}(x, y) \Longrightarrow x \sim_{N_a} y.$$

The ten-field fit is: traditional target identity-conjugator orbit lift; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* Z_a*; classifier* $\sim_{N_a}$*; stability by endpoint transport from the identity orbit witness; ledger identity conjugator* e *and the two centralizer endpoints; core theorem list Definition 41.104 and normalizer-orbit endpoint transport; exactness predicate* hsame *implies normalizer-orbit reachability on carried endpoints; standard bridge ordinary orbit relation contains classifier equality; status* $\mathsf{ProofTarget}$*.*

Proof. The equivalence theorem supplies the reflexive orbit witness $x \sim_{N_a} x$ on the centralizer carrier. Endpoint transport for the same orbit relation keeps the source endpoint fixed and moves the target endpoint from x to y along the given $\mathsf{hsame}(x, y)$ comparison. The carried endpoint witnesses $Z_a(x)$ and $Z_a(y)$ provide the transport hypotheses, so the transported witness is exactly $x \sim_{N_a} y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer orbit hsame lift from empty unit`

Definition 41.106 (Group center carrier)**.** *Fix a BEDC history-level group operation* mul *over* BHist*. The center carrier* $\mathsf{Z}(G)$ *is the predicate*

$$\mathsf{Z}(G)(x) :\Longleftrightarrow \forall a : \mathsf{BHist}, \mathsf{hsame}(\mathsf{mul}(x, a), \mathsf{mul}(a, x)).$$

Theorem 41.107 (Group center normal subgroup)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws, the center carrier* $\mathsf{Z}(G)$ *of Definition 41.106 is a normal subgroup:*

1. $\mathsf{Z}(G)(e)$*.*
2. *If* $\mathsf{Z}(G)(x)$ *and* $\mathsf{Z}(G)(y)$*, then* $\mathsf{Z}(G)(\mathsf{mul}(x, y))$*.*
3. *If* $\mathsf{Z}(G)(x)$*, then* $\mathsf{Z}(G)(\mathsf{inv}(x))$*.*
4. *If* $\mathsf{Z}(G)(x)$ *and* $\mathsf{hsame}(x, y)$*, then* $\mathsf{Z}(G)(y)$*.*
5. *If* $\mathsf{Z}(G)(x)$*, then for every* $s : \mathsf{BHist}$*,* $\mathsf{Z}(G)(\mathsf{Conj}_s(x))$*.*
6. *Every center element lies in every centralizer* Z_a *and therefore in every centralizer normalizer* N_a*.*

The ten-field fit is: traditional target center normal subgroup; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* $\mathsf{Z}(G)$*; classifier* hsame*; stability obligations unit, product, inverse, classifier transport, and arbitrary conjugation; ledger the central history and the acting history; core theorem list Definition 41.87, Definition 41.98, Definition 41.96, Definition 41.79, and Definition 41.65, together with associativity, unit laws, inverse laws, and multiplication congruence; standard bridge ordinary group centers; status* $\mathsf{ProofTarget}$*.*

Proof. The unit row follows pointwise. For any a, the left and right identity laws identify $\mathsf{mul}(e, a)$ and $\mathsf{mul}(a, e)$ with the same history a.

For product closure, fix a and assume that both x and y commute with every history. Reassociate $(xy)a$ to $x(ya)$, commute y past a, reassociate to $(xa)y$, commute x past a, and reassociate to $a(xy)$. Multiplication congruence transports the two pointwise centrality witnesses through the surrounding products.

For inverse closure, apply the inverse row of Definition 41.87 at each centralizer Z_a. Since a center element lies in every Z_a, its inverse also lies in every Z_a, which is exactly the defining universal condition for $\mathsf{Z}(G)$.

Classifier transport is pointwise. If $\mathsf{hsame}(x, y)$, multiplication congruence transports $\mathsf{mul}(x, a)\mathsf{hsamemul}(a, x)$ to $\mathsf{mul}(y, a)\mathsf{hsamemul}(a, y)$ for each a, using symmetry and transitivity of hsame for the two transported endpoints.

For conjugation stability, fix s and a center element x. Centrality gives $\mathsf{hsame}(\mathsf{mul}(s, x), \mathsf{mul}(x, s))$ and $\mathsf{hsame}(\mathsf{mul}(x, \mathsf{inv}(s)), \mathsf{mul}(\mathsf{inv}(s), x))$. Associativity, the inverse laws, and multiplication congruence identify $\mathsf{Conj}_s(x)$ with x.

The classifier exactness theorem transports this identification across every test product with a, so $\mathsf{Conj}_s(x)$ commutes with every history. Equivalently, Definition 41.96 records the same inner-action calculation and Definition 41.65 supplies the inverse word used in the conjugating product.

Finally, the definition of $\mathsf{Z}(G)$ says that a center element commutes with the particular history a, hence it is in Z_a. The last row of Definition 41.98 then places it in N_a for the same a. □

Lean4 ✓ `BEDC.Derived.GroupUp.group center normal subgroup from empty unit`

Theorem 41.108 (Group normalizer quotient conjugation well-defined)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *with the same associativity, left and right identity laws for* e*, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.100, let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *be a carried subgroup predicate. Define*

$$\mathsf{Conj}_u(s) := \mathsf{mul}(\mathsf{mul}(u, s), \mathsf{inv}(u)),$$

$$C_G(S)(c) :\Longleftrightarrow \forall s : \mathsf{BHist},\ S(s) \Rightarrow \mathsf{hsame}(\mathsf{mul}(c, s), \mathsf{mul}(s, c)),$$

and

$$N_G(S)(u) :\Longleftrightarrow (\forall s : \mathsf{BHist},\ S(s) \Rightarrow S(\mathsf{Conj}_u(s))) \wedge (\forall s : \mathsf{BHist},\ S(s) \Rightarrow S(\mathsf{Conj}_{\mathsf{inv}(u)}(s))).$$

If $N_G(S)(n)$ *and* $N_G(S)(n')$*, then the quotient-kernel clause for* $N_G(S)/C_G(S)$ *is exactly equality of the restricted conjugation actions:*

$$C_G(S)(\mathsf{mul}(\mathsf{inv}(n), n')) \Longleftrightarrow \forall s : \mathsf{BHist},\ S(s) \Rightarrow \mathsf{hsame}(\mathsf{Conj}_n(s), \mathsf{Conj}_{n'}(s)).$$

The ten-field fit is: traditional target normalizer quotient conjugation well-definedness; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist *with a carried* $\mathsf{Subgroup}^{\uparrow}$ *predicate* S*; carrier* $N_G(S)$*; classifier the*

quotient-kernel clause by $C_G(S)$; stability from conjugation composition and conjugation classifier exactness; ledger the two normalizer representatives, the relative kernel word, and the acted subgroup element; core theorem list Definition 41.96, Definition 41.79, and Definition 41.65; exactness predicate representatives act equally on S if and only if their relative displacement centralizes S; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. The endpoint hypotheses $N_G(S)(n)$ and $N_G(S)(n')$ make both conjugation maps restrict to the carrier S. Thus, for any s with $S(s)$, the displayed pointwise comparison is a classifier comparison between two carried endpoints of the same subgroup predicate.

Set

$$c := \mathsf{mul}(\mathsf{inv}(n), n').$$

The group laws give $\mathsf{hsame}(\mathsf{mul}(n, c), n')$: reassociate $\mathsf{mul}(n, \mathsf{mul}(\mathsf{inv}(n), n'))$ to $\mathsf{mul}(\mathsf{mul}(n, \mathsf{inv}(n)), n')$, reduce the adjacent inverse product to e, and then use the left unit law. Multiplication congruence and inverse congruence therefore transport the parameter of the conjugation word from n' to $\mathsf{mul}(n, c)$.

Assume first $C_G(S)(c)$ and fix s with $S(s)$. The centralizer clause gives $\mathsf{hsame}(\mathsf{mul}(c, s), \mathsf{mul}(s, c))$. Multiplying this comparison on the right by $\mathsf{inv}(c)$ and reducing the product $\mathsf{mul}(c, \mathsf{inv}(c))$ by the inverse and unit laws gives

$$\mathsf{hsame}(\mathsf{Conj}_c(s), s).$$

By Definition 41.96, conjugation by $\mathsf{mul}(n, c)$ is the composite of conjugation by c followed by conjugation by n. Hence $\mathsf{Conj}_{n'}(s)$ is hsame-equal to $\mathsf{Conj}_n(\mathsf{Conj}_c(s))$. Transporting $\mathsf{hsame}(\mathsf{Conj}_c(s), s)$ through the outer conjugation word by multiplication congruence gives $\mathsf{hsame}(\mathsf{Conj}_{n'}(s), \mathsf{Conj}_n(s))$, and symmetry gives the displayed pointwise action equality.

Conversely assume the pointwise action equality and fix s with $S(s)$. The same parameter calculation and Definition 41.96 identify $\mathsf{Conj}_{n'}(s)$ with $\mathsf{Conj}_n(\mathsf{Conj}_c(s))$. Combining this with the pointwise hypothesis gives

$$\mathsf{hsame}(\mathsf{Conj}_n(s), \mathsf{Conj}_n(\mathsf{Conj}_c(s))).$$

The reflection direction of Definition 41.79, applied to the conjugating parameter n, yields $\mathsf{hsame}(s, \mathsf{Conj}_c(s))$. After symmetry, multiply this comparison on the right by c. Associativity reduces $\mathsf{mul}(\mathsf{Conj}_c(s), c)$ to $\mathsf{mul}(c, s)$, because the middle product $\mathsf{mul}(\mathsf{inv}(c), c)$ collapses to e and the unit laws remove e. The transported comparison is exactly

$$\mathsf{hsame}(\mathsf{mul}(c, s), \mathsf{mul}(s, c)).$$

Since s was arbitrary under $S(s)$, this proves $C_G(S)(c)$, namely $C_G(S)(\mathsf{mul}(\mathsf{inv}(n), n'))$. □

41.12 Normalizer orbit classifier stability

Theorem 41.109 (Group centralizer normalizer orbit classifier certificate)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.104, let Z_a, N_a,* Conj_s*, and $\sim_{N_a}$ be as in Definition 41.103. For $x, y, x', y' : \mathsf{BHist}$, assume $Z_a(x)$, $Z_a(y)$, $Z_a(x')$, $Z_a(y')$,* $\mathsf{hsame}(x, x')$*, and* $\mathsf{hsame}(y, y')$*. Then endpoint transport preserves and reflects orbit reachability:*

$$x \sim_{N_a} y \iff x' \sim_{N_a} y'.$$

Together with Definition 41.104, this endpoint-transport row supplies the quotient-free classifier fields on the centralizer carrier: carrier reflexivity, symmetry, transitivity, and stability under replacement of both exposed endpoints by hsame*-equal histories.*

Proof. For the forward direction, unpack an orbit witness s for $x \sim_{N_a} y$. Keep the same normalizer witness $N_a(s)$ and replace only the source and target endpoints.

Multiplication congruence transports the conjugation word $\mathsf{Conj}_s(x)$ across $\mathsf{hsame}(x, x')$, giving a comparison between $\mathsf{Conj}_s(x')$ and $\mathsf{Conj}_s(x)$. Compose this comparison with the original endpoint witness $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$ and then with $\mathsf{hsame}(y, y')$. The carried assumptions $Z_a(x')$ and $Z_a(y')$ repack the same s as a witness for $x' \sim_{N_a} y'$.

For the reverse direction, apply the same argument after exchanging (x, y) with (x', y'). Symmetry of hsame turns the displayed endpoint comparisons into $\mathsf{hsame}(x', x)$ and $\mathsf{hsame}(y', y)$, so the witness for $x' \sim_{N_a} y'$ transports back to a witness for $x \sim_{N_a} y$.

The classifier certificate now has its structural rows from Definition 41.104: reflexivity, symmetry, and transitivity on the carrier Z_a. The two endpoint-transport directions above add stability under hsame-replacement of the visible source and target endpoints, without passing to a quotient object. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer orbit endpoint transport from empty unit iff`
Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer orbit endpoint transport iff`

41.13 Normalizer action on the centralizer

Theorem 41.110 (Group normalizer conjugation inverse roundtrip)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let Z_a be the centralizer carrier of Definition 41.84, let N_a be the normalizer carrier of Definition 41.95, and write*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

If $N_a(s)$ and $Z_a(x)$ hold, then:

1. $Z_a(\mathsf{Conj}_s(x))$.
2. $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.
3. $\mathsf{hsame}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), x)$.
4. $\mathsf{hsame}(\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x)), x)$.

The ten-field fit is: traditional target inverse-action certificate for the normalizer action; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier Z_a; classifier* hsame*; stability obligations the two normalizer clauses and the two classifier return equations; ledger conjugating element, inverse conjugating word, and centralizer witness; core theorem list Definition 41.96, Definition 41.65, Definition 41.57, and Definition 41.58; exactness predicate conjugation by s and by* $\mathsf{inv}(s)$ *are mutual classifier inverses on Z_a; standard bridge ordinary inverse inner action on the centralizer; status* ProofTarget.

Proof. The carrier rows are exactly the two clauses of $N_a(s)$. Applying the forward clause to $Z_a(x)$ gives $Z_a(\mathsf{Conj}_s(x))$, and applying the inverse clause to the same $Z_a(x)$ gives $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.

For the first classifier return equation, expand $\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x))$. By Definition 41.96, this is the same conjugation word as $\mathsf{Conj}_{\mathsf{mul}(\mathsf{inv}(s),s)}(x)$. The left inverse law reduces $\mathsf{mul}(\mathsf{inv}(s), s)$ to e, and the unit laws reduce $\mathsf{Conj}_e(x)$ to x.

For the second return equation, expand $\mathsf{Conj}_s(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$. The same composition theorem reads it as $\mathsf{Conj}_{\mathsf{mul}(s,\mathsf{inv}(s))}(x)$; the right inverse law reduces the conjugating product to e, and the unit laws again reduce $\mathsf{Conj}_e(x)$ to x.

Equivalently, the expanded words contain the adjacent inverse products exposed by associativity. Theorem 41.65 controls the inverse word, while Definition 41.57 and Definition 41.58 identify the double inverse $\mathsf{inv}(\mathsf{inv}(s))$ with s before the inverse and unit laws close the two classifier comparisons. □

Lean4 ✓ `BEDC.Derived.GroupUp.group normalizer conjugation inverse roundtrip from empty unit`

Theorem 41.111 (Group normalizer inverse closure)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 41.110, let Z_a be the centralizer carrier and let N_a be the two-sided centralizer normalizer carrier. If $N_a(s)$ holds, then $N_a(\mathsf{inv}(s))$ holds.*

Proof. Unfold N_a. Its first row for $\mathsf{inv}(s)$ is exactly the inverse-action row already present in $N_a(s)$. For the second row, the target conjugating parameter is $\mathsf{inv}(\mathsf{inv}(s))$. The group inverse-involutive calculation identifies this parameter with s, multiplication congruence carries the conjugation word across that hsame comparison, and centralizer membership transports along the resulting conjugation-word equality. □

Lean4 ✓ `BEDC.Derived.GroupUp.group normalizer inv closed from empty unit`

Theorem 41.112 (Group normalizer inverse hsame transport)**.** *For the same history-level group data and centralizer normalizer carrier N_a, if $N_a(s)$ holds and $\mathsf{hsame}(\mathsf{inv}(s), t)$, then $N_a(t)$ holds.*

Proof. First apply Definition 41.111 to obtain $N_a(\mathsf{inv}(s))$. The comparison $\mathsf{hsame}(\mathsf{inv}(s), t)$ transports the forward conjugation row by multiplication congruence and inverse congruence. Applying inverse congruence to the same comparison gives $\mathsf{hsame}(\mathsf{inv}(\mathsf{inv}(s)), \mathsf{inv}(t))$, which transports the inverse-conjugation row. These two transported rows are precisely $N_a(t)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group normalizer inverse hsame transport from empty unit`

Theorem 41.113 (Group centralizer normalizer conjugation automorphism)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for e, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let Z_a be the centralizer carrier of Definition 41.84, let N_a be the normalizer carrier of Definition 41.95, and write*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

If $N_a(s)$ holds, then conjugation by s restricts to Z_a and is classifier-bijective on that carrier:

1. *If $Z_a(x)$, then $Z_a(\mathsf{Conj}_s(x))$.*
2. *For every $y : \mathsf{BHist}$ with $Z_a(y)$, the witness $x := \mathsf{Conj}_{\mathsf{inv}(s)}(y)$ satisfies $Z_a(x)$ and $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$.*
3. *For all $x, y : \mathsf{BHist}$ with $Z_a(x)$ and $Z_a(y)$,*

$$\mathsf{hsame}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)) \iff \mathsf{hsame}(x, y).$$

The ten-field fit is: traditional target centralizer normalizer action; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier Z_a; classifier* hsame*; stability obligations normalizer preservation, inverse-conjugation witness, and conjugation classifier exactness; ledger conjugating element, inverse word, and centralizer membership witnesses; core theorem list Definition 41.98, Definition 41.110, Definition 41.79, and Definition 41.65; exactness predicate conjugation is a classifier automorphism of Z_a; standard bridge ordinary normalizer action on a subgroup; status* ProofTarget*.*

Proof. The restriction row is the first clause of $N_a(s)$. Starting from $Z_a(x)$, that clause returns $Z_a(\mathsf{Conj}_s(x))$ directly.

For the classifier-surjectivity row, let y satisfy $Z_a(y)$ and set $x := \mathsf{Conj}_{\mathsf{inv}(s)}(y)$. The second clause of $N_a(s)$ gives $Z_a(x)$ from $Z_a(y)$. The second return equation of Definition 41.110, applied to $Z_a(y)$, gives $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$.

For classifier exactness, apply Definition 41.79 to the conjugating word s. Its preservation direction gives $\mathsf{hsame}(x, y) \Rightarrow \mathsf{hsame}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y))$, and its reflection direction gives the converse implication. Restricting the inputs to $Z_a(x)$ and $Z_a(y)$ therefore makes conjugation by s a classifier automorphism of the centralizer carrier. □

Lean4 ✓ `BEDC.Derived.GroupUp.group centralizer normalizer conjugation automorphism from empty unit`

Theorem 41.114 (Group left absorption forces the right factor to the unit)**.** *For a history-level group operation with associativity, left and right identities, multiplication congruence, and left inverses, if* $ab\mathsf{hsame}a$*, then* $b\mathsf{hsame}e$*.*

Proof. Chain ab through the assumption to a, then backward through the right identity $ae\mathsf{hsame}a$. Left cancellation of the common factor a yields $b\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left absorb right factor unit`

Theorem 41.115 (Group left multiplication cancellation from empty unit)**.** *For a BEDC history-level group operation with empty unit* e*, associativity, left empty-unit law, multiplication congruence, and left inverses, left multiplication by a common factor preserves and reflects* hsame*:*

$$\mathsf{hsame}(ab, ac) \iff \mathsf{hsame}(b, c).$$

Proof. The forward direction is left cancellation for the common factor a. The reverse direction transports $b\mathsf{hsame}c$ through multiplication congruence with the fixed left factor a. □

Lean4 ✓ `BEDC.Derived.GroupUp.group left mul cancel from empty unit iff`

Theorem 41.116 (Group right cancellation)**.** *For a history-level group operation with associativity, right identity, multiplication congruence, and right inverses, equality of* ba *and* ca *under* hsame *implies equality of* b *and* c *under* hsame*.*

Proof. Chain b through the right identity, replace that identity by aa^{-1}, reassociate to $(ba)a^{-1}$, use multiplication congruence on the assumed product equality, reassociate back to $c(aa^{-1})$, collapse the right inverse, and finish by the right identity on c. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right cancel`

Theorem 41.117 (Group right multiplication cancellation from empty unit)**.** *For a BEDC history-level group operation with empty unit* e*, associativity, right empty-unit law, multiplication congruence, and right inverses, right multiplication by a common factor preserves and reflects* hsame*:*

$$\mathsf{hsame}(ba, ca) \iff \mathsf{hsame}(b, c).$$

Proof. The forward direction is right cancellation for the common factor a. The reverse direction transports $b\mathsf{hsame}c$ through multiplication congruence with the fixed right factor a. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right mul cancel from empty unit iff`

Theorem 41.118 (Group right absorption forces the left factor to the unit)**.** *For a history-level group operation with associativity, left and right identities, multiplication congruence, and right inverses, if* $ba\mathsf{hsame}a$*, then* $b\mathsf{hsame}e$*.*

Proof. Chain ba through the assumption to a, then backward through the left identity $ea\mathsf{hsame}a$. Right cancellation of the common factor a yields $b\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group right absorb left factor unit`

Theorem 41.119 (Group two-sided cancellation)**.** *For a history-level group operation with associativity, left and right identities, multiplication congruence, and two-sided inverses, equality of* $(ab)c$ *and* $(ad)c$ *under* hsame *implies equality of* b *and* d *under* hsame*.*

Proof. First apply right cancellation to the common final factor c, reducing the displayed equality to $\mathsf{hsame}(ab, ad)$. Then apply left cancellation to the common initial factor a, yielding $\mathsf{hsame}(b, d)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group two sided cancel`

Definition 41.120 (Group ledger policy)**.** *Record the monoid ledger plus every inverse operation applied.*

Definition 41.121 ($\mathsf{Group}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Group}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Group}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Proposition 41.122 (Group forgets to Monoid certificate)**.** *Let* C *carry a* $\mathsf{Group}^{\uparrow}(C)$ *certificate with multiplication* $\cdot$*, identity* e*, inverse operation* $^{-1}$*, and classifier* $\sim_C$*. If the inverse operation, inverse closure, inverse congruence, left and right inverse rows, and ledger entries whose expression provenance uses inverse formation are omitted, then the retained carrier, source, monoid pattern, classifier, monoid stability fields, and monoid ledger form a* $\mathsf{Monoid}^{\uparrow}(C)$ *certificate. The carrier* C*, multiplication* $\cdot$*, identity* e*, and classifier* $\sim_C$ *are unchanged.*

Proof. By Definition 41.121, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Group}^{\uparrow}(C)$. The construction keeps exactly the coordinates whose operations and endpoint claims use only the monoid part of the group structure.

By Definition 41.2, a group carrier is a monoid setup extended with an inverse operation. Omitting the inverse operation leaves the same carrier C, the same classifier $\sim_C$, the same multiplication $\cdot$, and the same identity e. These are precisely the carrier data listed in Definition 40.1.

By Definition 41.52, the source is a certified monoid carrier plus inverse data. Removing the inverse component leaves the certified monoid carrier required by Definition 40.9. No source endpoint, carrier witness, multiplication operation, identity element, or classifier is replaced.

By Definition 41.53, the group pattern is monoid expression formation extended by inverse formation. Restricting to expressions with no inverse node leaves expression formation from e, carrier witnesses, and binary multiplication, which is exactly the monoid pattern of Definition 40.10.

By Definition 41.54, the group classifier is the monoid carrier classifier together with inverse congruence. The retained relation is therefore the same carrier classifier required by Definition 40.11; the omitted inverse congruence row is not a $\mathsf{Monoid}^{\uparrow}$ obligation.

By Definition 41.55, the group stability certificate consists of all monoid stability fields together with inverse congruence, left inverse law, right inverse law, and closure under inverse. The

first component is exactly the list in Definition 40.13: classifier reflexivity, classifier transitivity, associativity under classifier, left identity, right identity, and multiplication congruence. The inverse-specific rows mention $^{-1}$ and are not required for $\mathsf{Monoid}^{\uparrow}$.

Finally, Definition 41.120 records the monoid ledger plus every inverse operation applied. Omitting entries justified by inverse formation leaves the carrier witnesses and multiplication-expression provenance described by Definition 40.33. Hence the retained carrier, source, pattern, classifier, stability, and ledger data satisfy the obligations named in Definition 40.34, with C, $\cdot$, e, and $\sim_C$ copied from the original $\mathsf{Group}^{\uparrow}(C)$ certificate. □

Definition 41.123 (Opposite group certificate data)**.** *Let a $\mathsf{Group}^{\uparrow}(C)$ certificate be presented by a carrier predicate $\mathcal{C}$, a classifier $\sim_C$, multiplication $\cdot_C$, identity e_C, and inverse operation i_C. Define the opposite multiplication by*

$$a \star_C b := b \cdot_C a.$$

The opposite group data C^{op} have the same carrier predicate $\mathcal{C}$, the same classifier $\sim_C$, the same identity e_C, the same inverse operation i_C, and multiplication $\star_C$.

Theorem 41.124 (Opposite group certificate)**.** *If $(\mathcal{C}, \sim_C, \cdot_C, e_C, i_C)$ satisfies the $\mathsf{Group}^{\uparrow}$ certificate obligations of Definition 41.121, then the data of Definition 41.123 also satisfy the $\mathsf{Group}^{\uparrow}$ certificate obligations. Thus the same carrier, classifier, identity, and inverse operation carry a group certificate with multiplication $\star_C$.*

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonHistory opposite laws`

Proof. Unpack the group obligations through Definition 41.121. The monoid reduct supplied by Definition 41.122 has carrier $\mathcal{C}$, classifier $\sim_C$, multiplication $\cdot_C$, and identity e_C. Applying Definition 40.37 to that reduct gives the carrier, source, monoid pattern, classifier, monoid stability rows, and monoid ledger for the multiplication $\star_C$ with the same identity e_C.

These rows supply exactly the monoid part demanded by Definition 41.52, Definition 41.53, and the first component of Definition 41.55. The carrier data of Definition 41.2 are also present: the carrier, classifier, and identity are unchanged, the multiplication is $\star_C$, and the inverse operation is still i_C.

It remains to check the inverse-specific group rows. Inverse closure is copied from the original group certificate because the inverse operation and carrier predicate are unchanged: from $\mathcal{C}(a)$ the original row gives $\mathcal{C}(i_C(a))$. Inverse congruence is copied for the same reason: from $a \sim_C b$ the original row gives $i_C(a) \sim_C i_C(b)$, and the classifier is still $\sim_C$.

The two inverse laws exchange their handedness under the reversed multiplication. The opposite left-inverse row at a carried element a unfolds to

$$i_C(a) \star_C a = a \cdot_C i_C(a) \sim_C e_C,$$

which is the original right-inverse row. The opposite right-inverse row unfolds to

$$a \star_C i_C(a) = i_C(a) \cdot_C a \sim_C e_C,$$

which is the original left-inverse row.

The classifier row of Definition 41.54 is the same carrier classifier equipped with the inverse congruence just copied. For the pattern and ledger, a $\star_C$ group-expression tree is interpreted as a $\cdot_C$ group-expression tree by exchanging the two children at each binary multiplication node

while leaving carrier leaves, the identity leaf, and inverse nodes unchanged. The opposite monoid certificate supplies the binary-node source and ledger reading; the inverse nodes use the original inverse source and ledger entries of Definition 41.120. Thus the source, pattern, classifier, stability, and ledger rows form the $\mathsf{Group}^{\uparrow}$ certificate data required by Definition 9.6 and Definition 41.121. □

Theorem 41.125 (Group product-unit inverse correspondence)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws, write* xy *for* $\mathsf{mul}(x, y)$ *and* x^{-1} *for* $\mathsf{inv}(x)$*. Then every* $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}(ab, \mathsf{e}) \iff \mathsf{hsame}(a, b^{-1}) \iff \mathsf{hsame}(b, a^{-1}).$$

The ten-field fit is: traditional target product-equals-empty inverse extraction; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations left and right equation exactness, unit laws, inverse laws, inverse uniqueness, inverse involutivity, and multiplication congruence; ledger the two factors and the two extracted inverse witnesses; core theorem this block; exactness predicate the displayed chain of equivalences; standard bridge ordinary group inverse extraction from* $ab = \mathsf{e}$*; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Theorem 41.72 to the equation $\mathsf{hsame}(ab, \mathsf{e})$ with right factor b. It classifies the same equation by $\mathsf{hsame}(a, \mathsf{e}b^{-1})$. The left identity law reduces $\mathsf{e}b^{-1}$ to b^{-1}, so the product-empty equation is equivalent to $\mathsf{hsame}(a, b^{-1})$.

Apply Theorem 41.70 to the same equation with left factor a. It classifies the equation by $\mathsf{hsame}(b, a^{-1}\mathsf{e})$. The right identity law reduces $a^{-1}\mathsf{e}$ to a^{-1}, giving the equivalence with $\mathsf{hsame}(b, a^{-1})$.

The two extracted descriptions are compatible because a witness $\mathsf{hsame}(a, b^{-1})$ makes a a right-inverse presentation for b, while $\mathsf{hsame}(b, a^{-1})$ makes b a left-inverse presentation for a. The inverse uniqueness theorem identifies the two presentations, and inverse involutivity returns either double-inverse comparison to the original factor. □

Lean4 ✓ `BEDC.Derived.GroupUp.group product unit inverse correspondence iff`

Theorem 41.126 (Group product-empty unit inverse correspondence)**.** *For a BEDC history-level group operation over* BHist *with associativity, left and right identity laws for* e*, multiplication congruence, and two-sided inverse laws, write* xy *for* $\mathsf{mul}(x, y)$ *and* x^{-1} *for* $\mathsf{inv}(x)$*. Then every* $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}(ab, \mathsf{e}) \iff \big(\mathsf{hsame}(a, b^{-1}) \wedge \mathsf{hsame}(b, a^{-1})\big).$$

The ten-field fit is: traditional target product-equals-empty paired inverse witnesses; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations the same left and right equation exactness, unit laws, inverse laws, and multiplication congruence as in Definition 41.125; ledger the product equation and the two extracted inverse witnesses; core theorem this block; exactness predicate the displayed equivalence; standard bridge ordinary group inverse extraction from an empty-history product; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Definition 41.125 to the product equation $\mathsf{hsame}(ab, \mathsf{e})$. The two equivalences give $\mathsf{hsame}(a, b^{-1})$ and $\mathsf{hsame}(b, a^{-1})$, which pair into the displayed conjunction.

For the reverse direction, start from the conjunction and use its first witness $\mathsf{hsame}(a, b^{-1})$. The right-multiplication exactness argument from Definition 41.125 transports that witness back to the product equation $\mathsf{hsame}(ab, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group product unit inverse correspondence from empty unit iff`

41.14 Center orbit collapse

Theorem 41.127 (Group center normalizer orbit collapse)**.** *For a BEDC history-level group operation over* BHist *with the same associativity, left and right identity laws for* e*, multiplication congruence, inverse congruence, and two-sided inverse laws as in Definition 41.98, let* $\mathsf{Z}(G)$ *be the center carrier of Definition 41.106, and let* Z_a*,* N_a*,* Conj_s*, and* $\sim_{N_a}$ *be as in Definition 41.95 and Definition 41.103. For histories* $x, y : \mathsf{BHist}$*, if* $\mathsf{Z}(G)(x)$ *and* $Z_a(y)$*, then*

$$x \sim_{N_a} y \iff \mathsf{hsame}(x, y).$$

The per-object template names the traditional target center orbit collapse; the BEDC source as $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* $\mathsf{Z}(G)$ *inside* Z_a*; classifier* $\sim_{N_a}$ *versus* hsame*; stability from conjugation fixedness of center elements; ledger the normalizer witness; core theorem list Definition 41.98, Definition 41.101, Definition 41.104, and Definition 41.107; exactness predicate the displayed iff; standard bridge ordinary center elements have singleton conjugation orbits; and status* $\mathsf{ProofTarget}$*.*

Proof. The last row of Definition 41.107 places the center element x in the centralizer carrier Z_a. Thus the orbit relation has both endpoint carrier facts available once $Z_a(y)$ is supplied.

For the forward direction, unpack a witness s for $x \sim_{N_a} y$. Centrality of x commutes x past s and past $\mathsf{inv}(s)$. Associativity, the two-sided inverse laws, unit laws, multiplication congruence, and inverse congruence then identify $\mathsf{Conj}_s(x) = \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s))$ with x under hsame.

Composing this fixedness comparison with the orbit endpoint witness $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$ gives $\mathsf{hsame}(x, y)$ by symmetry and transitivity of hsame.

For the reverse direction, use the identity element as the normalizer witness. The identity row of Definition 41.98 gives $N_a(e)$, and the identity-action row of Definition 41.101 gives $\mathsf{hsame}(\mathsf{Conj}_e(x), x)$.

Together with the center-to-centralizer carrier fact for x, the supplied $Z_a(y)$, and the assumed $\mathsf{hsame}(x, y)$, these data package the orbit witness for $x \sim_{N_a} y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group center normalizer orbit collapse from empty unit iff`

Theorem 41.128 (Group center normalizer orbit fiber determinacy)**.** *For a BEDC history-level group operation over* BHist *with the same hypotheses and notation* $\mathsf{Z}(G)$*,* Z_a*,* N_a*,* Conj_s*, and* $\sim_{N_a}$ *as in Definition 41.127, every* $x, y, z : \mathsf{BHist}$ *satisfies*

$$\mathsf{Z}(G)(x) \wedge Z_a(y) \wedge Z_a(z) \wedge x \sim_{N_a} y \wedge x \sim_{N_a} z \Longrightarrow \mathsf{hsame}(y, z).$$

The per-object template names the traditional target central-orbit fiber determinacy; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist *plus the normalizer orbit relation; carrier* Z_a *with central anchor* $\mathsf{Z}(G)$*; classifier* hsame*; stability from orbit collapse and the symmetry and transitivity laws for* hsame*; ledger the two normalizer witnesses over the same central anchor; core theorem list Definition 41.127 and the* hsame *equivalence laws; exactness predicate the displayed implication; and status* $\mathsf{ProofTarget}$*.*

Proof. Start from the five displayed assumptions. Apply Definition 41.127 to the first orbit witness $x \sim_{N_a} y$, using the centrality of x and the carrier fact $Z_a(y)$, to obtain $\mathsf{hsame}(x, y)$. Apply the same collapse theorem to the second orbit witness $x \sim_{N_a} z$, using the same centrality of x and the carrier fact $Z_a(z)$, to obtain $\mathsf{hsame}(x, z)$.

The first comparison reverses by symmetry of hsame, giving $\mathsf{hsame}(y, x)$. Transitivity of hsame then composes $\mathsf{hsame}(y, x)$ with $\mathsf{hsame}(x, z)$ and produces the required classifier equality $\mathsf{hsame}(y, z)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.group center normalizer orbit fiber determinacy from empty unit`

41.15 Singleton continuation endpoint equivalence

Theorem 41.129 (Group singleton continuation endpoint equivalence)**.** *For histories* P, Q, R : BHist *with* $\mathsf{Cont}(P, Q, R)$*, the concrete singleton group classifier compares the continuation result with either input endpoint equivalently:*

$$R_{\mathsf{e}}(R, P) \iff R_{\mathsf{e}}(R, Q).$$

The ten-field fit is: traditional target terminal-group endpoint agreement under continuation; BEDC source the concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *and* Cont*; carrier* A_{e}*; classifier* R_{e}*; stability from result-left and result-right classifier exactness; ledger the continuation witness; core theorem list Theorem 41.17 and Theorem 41.19; exactness predicate the displayed iff; standard bridge the one-element group; and status* $\mathsf{ProofTarget}$*.*

Proof. Use Theorem 41.17 to identify $R_{\mathsf{e}}(R, P)$ with the input classifier $R_{\mathsf{e}}(P, Q)$ under the given continuation witness.

Use Theorem 41.19 to identify the same input classifier $R_{\mathsf{e}}(P, Q)$ with $R_{\mathsf{e}}(R, Q)$. Composing these two equivalences gives the displayed endpoint-to-endpoint equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier continuation endpoint equivalence iff`

Theorem 41.130 (Group singleton contextual continuation endpoint equivalence)**.** *For histories* L, R, L', R', P, Q, S : BHist*, if* $A_{\mathsf{e}}(L)$*,* $A_{\mathsf{e}}(R)$*,* $A_{\mathsf{e}}(L')$*,* $A_{\mathsf{e}}(R')$*, and* $\mathsf{Cont}(P, Q, S)$*, then the concrete singleton group classifier compares the hidden-context continuation result with either endpoint equivalently:*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, S), L'), \mathsf{append}(\mathsf{append}(R, P), R')) \iff R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, S), L'), \mathsf{append}(\mathsf{append}(R, Q), R$$

The ten-field fit is: traditional target terminal-group contextual endpoint agreement under continuation; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus* Cont*; carrier* A_{e}*; classifier* R_{e}*; stability from two-sided context cancellation and continuation endpoint equivalence; ledger hidden prefix/suffix singleton contexts plus the continuation witness; core theorem this block; exactness predicate the displayed iff; standard bridge terminal group contextual endpoint agreement; and status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Theorem 41.9 with the four singleton carrier witnesses for L, R, L', R'. This cancels the hidden contexts around the left displayed classifier and exposes $R_{\mathsf{e}}(S, P)$.

Apply Theorem 41.129 to the continuation witness $\mathsf{Cont}(P, Q, S)$. It transports the exposed endpoint classifier from $R_{\mathsf{e}}(S, P)$ to $R_{\mathsf{e}}(S, Q)$.

Apply the reverse direction of Theorem 41.9 with the same four singleton carrier witnesses. This reinstates the hidden contexts around S and Q, producing the right displayed classifier.

For the reverse direction, apply Theorem 41.9 to cancel the hidden contexts around the right displayed classifier and expose $R_{\mathsf{e}}(S, Q)$.

Use the reverse direction of Theorem 41.129 under the same continuation witness to obtain $R_{\mathsf{e}}(S, P)$.

Reattach the hidden contexts by the reverse direction of the two-sided cancellation theorem, producing the left displayed classifier. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier contextual continuation endpoint equivalence iff`

Theorem 41.131 (Group singleton suffix-context product-unit split)**.** *For histories L, R, p, q : BHist, if $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$, then*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(p, q), L), \mathsf{append}(\mathsf{e}, R)) \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).$$

The ten-field fit names the traditional target as terminal-group unit-product split under suffix contexts; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability as right-context cancellation plus append empty-result split; ledger the hidden suffix endpoints and factors p, q; core theorem list Definition 41.8 and Definition 41.10; exactness predicate the displayed iff; standard bridge terminal group suffix-context unit-product split; and status $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.8 with $Q = \mathsf{append}(p, q)$ and $S = \mathsf{e}$. The carrier hypotheses $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$ strip the singleton suffix contexts, reducing the displayed classifier to

$$R_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{e}).$$

Apply Definition 41.10 to this reduced classifier. It identifies the reduced classifier exactly with $A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q)$, giving both directions of the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier suffix context product unit split iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append suffix unit split iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier suffix product unit split iff`

Theorem 41.132 (Group singleton classifier append context visible middle exclusion)**.** *For histories L, R, p, q, S : BHist, if $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$, then a visible endpoint exposed after appending hidden singleton contexts cannot support the concrete singleton group classifier:*

$$\begin{aligned}
&(R_{\mathsf{e}}(\mathsf{append}(L, \mathsf{E0}(p)), \mathsf{append}(R, S)) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(L, \mathsf{E1}(p)), \mathsf{append}(R, S)) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(L, S), \mathsf{append}(R, \mathsf{E0}(q))) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(L, S), \mathsf{append}(R, \mathsf{E1}(q))) \to \mathsf{False}).
\end{aligned}$$

The ten-field fit is: traditional target terminal-group visible-endpoint exclusion under hidden singleton contexts; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability by context cancellation and constructor no-confusion; ledger hidden singleton contexts and visible endpoint witness; core theorem list Definition 41.7, Definition 41.13, and Definition 41.14; exactness predicate the four displayed contradictions; standard bridge the terminal group has no visible non-unit endpoint; and status $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.7 with the supplied carrier hypotheses $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$. In each of the four displayed cases, context cancellation exposes the corresponding uncontexted classifier with either an $\mathsf{E0}$ or an $\mathsf{E1}$ endpoint.

Unfolding R_{e} gives the two endpoint carrier witnesses for that uncontexted classifier. In the first two cases the left carrier contains a visible head created by $\mathsf{append}(L, \mathsf{E0}(p))$ or $\mathsf{append}(L, \mathsf{E1}(p))$, so Definition 41.13 contradicts the singleton carrier.

In the last two cases the right carrier contains a visible tail created by $\mathsf{append}(R, \mathsf{E0}(q))$ or $\mathsf{append}(R, \mathsf{E1}(q))$, so Definition 41.14 gives the contradiction. Constructor separation distinguishes the $\mathsf{E0}$ and $\mathsf{E1}$ branches without any additional classifier data. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append context visible middle absurd`

Theorem 41.133 (Group singleton two-sided contextual visible middle exclusion)**.** *For histories L, R, L', R', p, q, M : BHist, if $A_{\mathsf{e}}(L)$, $A_{\mathsf{e}}(R)$, $A_{\mathsf{e}}(L')$, and $A_{\mathsf{e}}(R')$, then visible middle endpoints remain impossible after hidden prefix and suffix contexts:*

$$\begin{aligned}
&(R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, \mathsf{E0}(p)), L'), \mathsf{append}(\mathsf{append}(R, M), R')) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, \mathsf{E1}(p)), L'), \mathsf{append}(\mathsf{append}(R, M), R')) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, M), L'), \mathsf{append}(\mathsf{append}(R, \mathsf{E0}(q)), R')) \to \mathsf{False}) \wedge \\
&(R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, M), L'), \mathsf{append}(\mathsf{append}(R, \mathsf{E1}(q)), R')) \to \mathsf{False}).
\end{aligned}$$

The ten-field fit is: traditional target terminal-group visible-endpoint exclusion under hidden two-sided singleton contexts; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability by right-context cancellation followed by prefix-context visible-middle exclusion; ledger prefix contexts L, R, suffix contexts L', R', neutral middle witness M, and visible endpoint witnesses p, q; core theorem list Definition 41.8 and Definition 41.132; exactness predicate the four displayed contradictions; standard bridge the terminal group has no visible middle endpoint under hidden contexts; and status $\mathsf{ProofTarget}$.

Proof. Start with any one of the four displayed classifier antecedents. Apply Definition 41.8 with the suffix carrier witnesses $A_{\mathsf{e}}(L')$ and $A_{\mathsf{e}}(R')$. This removes the right contexts and reduces the antecedent to the corresponding classifier without the outer suffixes.

For the first two reduced classifiers, use the left-middle clauses of Definition 41.132 with the prefix carrier witnesses $A_{\mathsf{e}}(L)$ and $A_{\mathsf{e}}(R)$. They rule out the exposed $\mathsf{E0}(p)$ and $\mathsf{E1}(p)$ endpoints.

For the last two reduced classifiers, use the right-middle clauses of the same theorem with the same prefix carrier witnesses. They rule out the exposed $\mathsf{E0}(q)$ and $\mathsf{E1}(q)$ endpoints, completing the four contradictions. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual visible middle absurd`

41.16 Singleton two-sided product-unit exactness

Theorem 41.134 (Group singleton two-sided contextual product-unit split)**.** *For L, R, L', R', p, q : BHist, if $A_{\mathsf{e}}(L)$, $A_{\mathsf{e}}(R)$, $A_{\mathsf{e}}(L')$, and $A_{\mathsf{e}}(R')$, then*

$$\begin{aligned}
&R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L, \mathsf{append}(p, q)), L'), \mathsf{append}(\mathsf{append}(R, \mathsf{e}), R')) \\
&\qquad \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).
\end{aligned}$$

The ten-field fit is: traditional target terminal-group product-unit exactness under two-sided contexts; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist and append; carrier A_{e}; classifier R_{e}; stability from two-sided context cancellation and append empty-result split; ledger prefix and suffix contexts and factors p, q; core theorem list Definition 41.9 and Definition 41.10; exactness predicate the displayed iff; standard bridge terminal group; and status $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.9 with $Q = \mathsf{append}(p, q)$ and $S = \mathsf{e}$. The four singleton carrier hypotheses strip the prefix and suffix context pairs, reducing the displayed classifier to

$$R_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{e}).$$

Apply Definition 41.10 to the reduced classifier. It identifies the reduced classifier with $A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q)$, giving the desired equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual product unit split iff`

Theorem 41.135 (Group singleton two-sided contextual continuation product-unit coverage)**.** *For histories* $L_0, R_0, L_1, R_1, P, Q, S, T : \mathsf{BHist}$*, if* $A_\mathsf{e}(L_0)$*,* $A_\mathsf{e}(R_0)$*,* $A_\mathsf{e}(L_1)$*,* $A_\mathsf{e}(R_1)$*, and* $\mathsf{Cont}(P, Q, S)$*, then*

$$R_\mathsf{e}(\mathsf{append}(\mathsf{append}(L_0, \mathsf{append}(S, T)), L_1), \mathsf{append}(\mathsf{append}(R_0, \mathsf{e}), R_1)) \iff A_\mathsf{e}(P) \wedge A_\mathsf{e}(Q) \wedge A_\mathsf{e}(T).$$

The ten-field fit is: traditional target terminal-group continuation product-unit exactness under two-sided contexts; BEDC source concrete singleton $\mathsf{Group}^\uparrow$ *over* BHist *plus* Cont *and* append*; carrier* A_e*; classifier* R_e*; stability from two-sided context cancellation, continuation terminal collapse, and product-unit split; ledger the prefix and suffix contexts, continuation witness, suffix factor* T*, and hidden endpoint witnesses; core theorem list Definition 41.134, Definition 41.90, and Definition 41.88; exactness the displayed iff; standard bridge terminal group contextual product-unit coverage; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.134 with $p = S$ and $q = T$. The four context carrier hypotheses strip the two prefix and suffix pairs, reducing the displayed classifier to $A_\mathsf{e}(S) \wedge A_\mathsf{e}(T)$.

Use Definition 41.90 under the continuation witness $\mathsf{Cont}(P, Q, S)$ to identify $A_\mathsf{e}(S)$ with $R_\mathsf{e}(P, Q)$. The intermediate condition is therefore $R_\mathsf{e}(P, Q) \wedge A_\mathsf{e}(T)$.

Apply the first row of Definition 41.88 to replace $R_\mathsf{e}(P, Q)$ by $A_\mathsf{e}(P) \wedge A_\mathsf{e}(Q)$. Reassociating conjunctions gives the stated three-factor coverage. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual continuation product unit coverage iff`

Theorem 41.136 (Group singleton two-sided contextual continuation product-unit visible exclusion)**.** *For histories* $L_0, R_0, L_1, R_1, P, Q, S, T, r : \mathsf{BHist}$*, if* $A_\mathsf{e}(L_0)$*,* $A_\mathsf{e}(R_0)$*,* $A_\mathsf{e}(L_1)$*,* $A_\mathsf{e}(R_1)$*, and* $\mathsf{Cont}(P, Q, S)$*, and if*

$$R_\mathsf{e}(\mathsf{append}(\mathsf{append}(L_0, \mathsf{append}(S, T)), L_1), \mathsf{append}(\mathsf{append}(R_0, \mathsf{e}), R_1))$$

holds, then

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

The ten-field template names the traditional target terminal-group contextual continuation product-unit visibility exclusion; BEDC source concrete singleton $\mathsf{Group}^\uparrow$ *over* BHist *plus* Cont*; carrier* A_e*; classifier* R_e*; stability from contextual product-unit coverage and terminal collapse; ledger the two contexts, continuation witness, suffix factor, and visible result endpoint; core theorem list Definition 41.135, Definition 41.88, and Definition 41.90; exactness the displayed implication package; downstream consumer* $\mathsf{Group}^\uparrow$ *and* $\mathsf{Subgroup}^\uparrow$ *quotient visibility guards; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.135 to the four context carrier witnesses and the continuation witness $\mathsf{Cont}(P, Q, S)$. The forward direction sends the displayed contextual classifier to $A_\mathsf{e}(P) \wedge A_\mathsf{e}(Q) \wedge A_\mathsf{e}(T)$.

Keep the first two carrier components. Use the reverse direction of the first row of Definition 41.88 to repack $A_\mathsf{e}(P) \wedge A_\mathsf{e}(Q)$ as $R_\mathsf{e}(P, Q)$.

Apply the visible-result part of Definition 41.90 under $\mathsf{Cont}(P, Q, S)$. The repacked input classifier is one of the terminal conditions, so the two continuation endpoints $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$ are excluded as displayed. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual continuation product unit visible absurd`

Theorem 41.137 (Group singleton double continuation product-unit chain coverage)**.** *For histories* $P, Q, S, T, U, V : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, S)$ *and* $\mathsf{Cont}(S, T, U)$*, the concrete singleton* $\mathsf{Group}^{\uparrow}$ *instance satisfies*

$$R_{\mathsf{e}}(\mathsf{append}(U, V), \mathsf{e}) \iff A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V).$$

The ten-field fit names the traditional target terminal-group two-step continuation product-unit coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus two* Cont *witnesses and append; carrier* A_{e}*; classifier* R_{e}*; stability from continuation product-unit coverage, continuation terminal collapse, and terminal classifier packaging; ledger the two continuation witnesses and suffix endpoint; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.91 to the continuation witness $\mathsf{Cont}(S, T, U)$. This rewrites the left side as

$$A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V).$$

Use Definition 41.90 under the continuation witness $\mathsf{Cont}(P, Q, S)$ to identify $A_{\mathsf{e}}(S)$ with $R_{\mathsf{e}}(P, Q)$. The intermediate condition becomes

$$R_{\mathsf{e}}(P, Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V).$$

Finally, apply the first row of Definition 41.88 to rewrite $R_{\mathsf{e}}(P, Q)$ as $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q)$. Reassociating conjunctions gives exactly

$$A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V).$$

□

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier double continuation product unit chain coverage iff`

Theorem 41.138 (Group singleton double continuation product-unit visible exclusion)**.** *For histories* $P, Q, S, T, U, V, r : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, S)$ *and* $\mathsf{Cont}(S, T, U)$*, if*

$$R_{\mathsf{e}}(\mathsf{append}(U, V), \mathsf{e})$$

holds, then the concrete singleton $\mathsf{Group}^{\uparrow}$ *instance excludes visible results for both continuation witnesses:*

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(S, T, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(S, T, \mathsf{E1}(r)) \Rightarrow \mathsf{False})$$

The ten-field template names the traditional target terminal-group two-step continuation visibility guard; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *with two* Cont *witnesses and append; carrier* A_{e}*; classifier* R_{e}*; stability from double continuation product-unit coverage, terminal classifier packaging, and continuation terminal collapse; ledger the two continuation witnesses, suffix endpoint, and visible result endpoint; core theorem this block; exactness the four displayed contradictions; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression and subgroup visibility guards; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.137 to the two continuation witnesses and the product-unit classifier. The forward direction gives

$$A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V).$$

Keep the first two carrier components. Use the reverse direction of the first row of Definition 41.88 to repack $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q)$ as $R_{\mathsf{e}}(P, Q)$.

Apply the visible-result part of Definition 41.90 under $\mathsf{Cont}(P, Q, S)$. The repacked input classifier excludes $\mathsf{Cont}(P, Q, \mathsf{E0}(r))$ and $\mathsf{Cont}(P, Q, \mathsf{E1}(r))$, and the same collapse identifies the terminal condition $A_{\mathsf{e}}(S)$.

Combine this $A_{\mathsf{e}}(S)$ with the retained factor $A_{\mathsf{e}}(T)$. The reverse direction of the first row of Definition 41.88 repacks $A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T)$ as $R_{\mathsf{e}}(S, T)$.

Apply the visible-result part of Definition 41.90 under $\mathsf{Cont}(S, T, U)$. The repacked second input classifier excludes $\mathsf{Cont}(S, T, \mathsf{E0}(r))$ and $\mathsf{Cont}(S, T, \mathsf{E1}(r))$, giving the four displayed contradictions. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier double continuation product unit visible exclusion`

Theorem 41.139 (Group singleton triple continuation product-unit chain coverage)**.** *For histories* $P, Q, S, T, U, V, W, X : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, S)$*,* $\mathsf{Cont}(S, T, U)$*, and* $\mathsf{Cont}(U, V, W)$*, the concrete singleton* $\mathsf{Group}^{\uparrow}$ *instance satisfies*

$$R_{\mathsf{e}}(\mathsf{append}(W, X), \mathsf{e}) \iff A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

The ten-field fit names terminal-group three-step continuation product-unit coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus three* Cont *witnesses and append; carrier* A_{e}*; classifier* R_{e}*; stability from continuation product-unit triple coverage, continuation terminal collapse, and terminal classifier packaging; ledger the three continuation witnesses and suffix endpoint; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression and group orbit compression; status* $\mathsf{ProofTarget}$*.*

Proof. The displayed equivalence is obtained by chaining reversible classifier identifications. Apply Definition 41.91 to $\mathsf{Cont}(U, V, W)$, which identifies the left side with

$$A_{\mathsf{e}}(U) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

Use Definition 41.90 under $\mathsf{Cont}(S, T, U)$ to identify $A_{\mathsf{e}}(U)$ with $R_{\mathsf{e}}(S, T)$. Then apply the first row of Definition 41.88 to replace $R_{\mathsf{e}}(S, T)$ by

$$A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T).$$

It remains to expand the new terminal factor $A_{\mathsf{e}}(S)$. Use Definition 41.90 under $\mathsf{Cont}(P, Q, S)$ to identify $A_{\mathsf{e}}(S)$ with $R_{\mathsf{e}}(P, Q)$, and apply the same terminal classifier package to replace it by

$$A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q).$$

Reassociating the conjunctions gives exactly

$$A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

□

41.17 Concrete singleton continuation group instance

Definition 41.140 (Group singleton continuation group instance)**.** *The singleton continuation group instance over* BHist *uses the carrier*

$$A_{\mathsf{Cont,e}}(h) := \mathsf{hsame}(h, \mathsf{e}).$$

The product is read by Cont *on the singleton endpoint: for carried* h *and* k*, the product row is the unique endpoint* p *whose continuation ledger* $\mathsf{Cont}(h, k, p)$ *is classified with* e*. The identity is* e*, and the inverse operation is the same singleton endpoint,*

$$e_{\mathsf{Cont,e}} := \mathsf{e}, \qquad \mathsf{inv}_{\mathsf{Cont,e}}(h) := \mathsf{e}.$$

The classifier is the restricted singleton classifier

$$R_{\mathsf{Cont,e}}(h, k) := A_{\mathsf{Cont,e}}(h) \wedge A_{\mathsf{Cont,e}}(k) \wedge \mathsf{hsame}(h, k).$$

Theorem 41.141 (Group singleton continuation laws)**.** *The data of Definition 41.140 satisfy the concrete* $\mathsf{Group}^{\uparrow}$ *laws over* BHist*. Multiplication closure, the left and right identity laws, associativity, multiplication congruence, inverse closure, and the left and right inverse laws are all witnessed by the continuation unit and associativity rows already used by the singleton* $\mathsf{Group}^{\uparrow}$ *chapter.*

Proof. Carrier inhabitance is $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Hence the displayed identity endpoint is carried, and the constant inverse endpoint is carried for every admitted input.

For multiplication closure, suppose $A_{\mathsf{Cont,e}}(h)$ and $A_{\mathsf{Cont,e}}(k)$. The continuation product row is read at the singleton endpoint; composing the two carrier witnesses with the continuation unit rows classifies the product endpoint with e.

The left and right unit laws are the continuation unit rows specialized to the empty endpoint. Since every admitted endpoint is already classified with e, both products are classified with the original input through $R_{\mathsf{Cont,e}}$.

Associativity and multiplication congruence follow from the same continuation associativity and transport rows used by the surrounding singleton product-unit theorems. All intermediate endpoints collapse to the singleton carrier, so the two parenthesized products repack into the same restricted classifier.

For inverses, $\mathsf{inv}_{\mathsf{Cont,e}}(h)$ is e. The two inverse products therefore reduce to the left and right unit rows, and the resulting endpoint is classified with $e_{\mathsf{Cont,e}}$. □

Theorem 41.142 (Group singleton inverse boundary)**.** *In the concrete singleton continuation group instance, the inverse field is visible at the public boundary: every admitted endpoint* h *satisfies* $\mathsf{hsame}(h, \mathsf{e})$*, and the inverse row is the displayed endpoint* e *with the same singleton classifier and continuation provenance.*

Proof. Project the carrier in Definition 41.140. Admission of h is exactly the row $\mathsf{hsame}(h, \mathsf{e})$, so the public packet already contains the comparison that identifies h with the singleton endpoint.

The inverse field is not computed by an external operation table. It is the displayed endpoint e, accompanied by the same carrier proof $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$ and the same continuation provenance used for the identity rows.

Thus every inverse read is visible as a projection of the singleton carrier and continuation ledger. There is no hidden inverse selector or ambient group object outside the BHist boundary. □

Theorem 41.143 (Group singleton triple continuation product-unit coverage and visibility)**.** *For histories* $P, Q, S, T, U, V, W, X, r : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, S)$*,* $\mathsf{Cont}(S, T, U)$*, and* $\mathsf{Cont}(U, V, W)$*, the concrete singleton* $\mathsf{Group}^{\uparrow}$ *instance satisfies*

$$R_{\mathsf{e}}(\mathsf{append}(W, X), \mathsf{e}) \iff A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

Under the left-hand classifier, all three continuation witnesses exclude visible results:

$$(\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(S, T, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(S, T, \mathsf{E1}(r)) \Rightarrow \mathsf{False})$$

The ten-field fit names the traditional target terminal-group three-step continuation product-unit coverage with visibility guard; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus three* Cont *witnesses and append; carrier* A_{e}*; classifier* R_{e}*; stability from product-unit continuation coverage, terminal collapse, and terminal classifier packaging; ledger the three continuation witnesses, suffix endpoint, and visible result endpoint; exactness the displayed iff and six contradictions; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression; and status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.91 to the third continuation witness $\mathsf{Cont}(U, V, W)$. This rewrites the left-hand classifier as

$$A_{\mathsf{e}}(U) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

Use Definition 41.90 under $\mathsf{Cont}(S, T, U)$ to identify $A_{\mathsf{e}}(U)$ with $R_{\mathsf{e}}(S, T)$. Then apply the first row of Definition 41.88 to rewrite that classifier as $A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T)$. The condition is now

$$A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X).$$

Apply Definition 41.90 again, this time under $\mathsf{Cont}(P, Q, S)$, to identify $A_{\mathsf{e}}(S)$ with $R_{\mathsf{e}}(P, Q)$. The first row of Definition 41.88 replaces $R_{\mathsf{e}}(P, Q)$ by $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q)$, yielding the displayed five-factor classifier.

For visibility, assume the left-hand classifier. The forward direction just constructed supplies the carrier factors for P, Q, T, V, X, and the two terminal collapse steps also supply the hidden endpoint carriers $A_{\mathsf{e}}(S)$ and $A_{\mathsf{e}}(U)$.

Repack $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q)$, $A_{\mathsf{e}}(S) \wedge A_{\mathsf{e}}(T)$, and $A_{\mathsf{e}}(U) \wedge A_{\mathsf{e}}(V)$ into the adjacent endpoint classifiers $R_{\mathsf{e}}(P, Q)$, $R_{\mathsf{e}}(S, T)$, and $R_{\mathsf{e}}(U, V)$ by the reverse direction of the terminal classifier package.

Apply the visible-result clause of Definition 41.90 at each boundary. The three repacked classifiers respectively rule out visible results for $\mathsf{Cont}(P, Q, -)$, $\mathsf{Cont}(S, T, -)$, and $\mathsf{Cont}(U, V, -)$, which gives the six displayed implications. □

Theorem 41.144 (Group singleton contextual triple continuation product-unit exact visibility)**.** *For histories* $L_0, R_0, L_1, R_1, P, Q, S, T, U, V, W, X, r : \mathsf{BHist}$*, if* $A_{\mathsf{e}}(L_0)$*,* $A_{\mathsf{e}}(R_0)$*,* $A_{\mathsf{e}}(L_1)$*,* $A_{\mathsf{e}}(R_1)$*,* $\mathsf{Cont}(P, Q, S)$*,* $\mathsf{Cont}(S, T, U)$*, and* $\mathsf{Cont}(U, V, W)$*, then*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L_0, \mathsf{append}(W, X \iff$$

$$(A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X)) \wedge (\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}($$

The ten-field fit is: traditional target terminal-group contextual three-step continuation product-unit exact visibility; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist *plus two context pairs, three* Cont *witnesses, visible continuation endpoints, and append; carrier* A_{e}*; classifier* R_{e}*; stability from two-sided context cancellation, triple continuation product-unit coverage, terminal collapse, and terminal classifier packaging; ledger the two hidden context pairs, three continuation witnesses, suffix endpoint, and visible result seed* r*; core theorem list Definition 41.9 and Definition 41.143; exactness the displayed iff package; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression; and status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, assume the displayed contextual classifier. Apply Definition 41.9 with $Q = \mathsf{append}(W, X)$ and $S = \mathsf{e}$ to the four context carrier witnesses. This strips the hidden contexts and reduces the classifier to $R_{\mathsf{e}}(\mathsf{append}(W, X), \mathsf{e})$.

Apply Definition 41.143 to the three continuation witnesses. Its classifier direction supplies the five carrier factors $A_{\mathsf{e}}(P)$, $A_{\mathsf{e}}(Q)$, $A_{\mathsf{e}}(T)$, $A_{\mathsf{e}}(V)$, and $A_{\mathsf{e}}(X)$; its visibility clause supplies the six visible-result exclusions for the same endpoint seed r.

Conjoining these facts gives the full right-hand visible-classification package. No extra hidden context data remains, because all four context carriers were consumed by the cancellation step.

For the reverse direction, assume the right-hand package. Its carrier component contains $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T) \wedge A_{\mathsf{e}}(V) \wedge A_{\mathsf{e}}(X)$, which is exactly the carrier side of the triple continuation product-unit coverage theorem.

Use the reverse classifier direction of Definition 41.143 with $\mathsf{Cont}(P, Q, S)$, $\mathsf{Cont}(S, T, U)$, and $\mathsf{Cont}(U, V, W)$ to reconstruct $R_{\mathsf{e}}(\mathsf{append}(W, X), \mathsf{e})$. The six exclusion clauses remain part of the package but are not needed to rebuild this unwrapped classifier.

Finally apply the reverse direction of Definition 41.9 with the four context carrier witnesses. This wraps the unwrapped classifier by the left and right context pairs and gives the displayed contextual classifier. □

41.18 Singleton conjugation terminal collapse

Theorem 41.145 (Group singleton conjugation terminal collapse)**.** *For histories $s, x : \mathsf{BHist}$, if $A_{\mathsf{e}}(s)$, then singleton conjugation by s is exactly terminal carrier membership of the middle endpoint:*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(s, x), \mathsf{GroupSingletonInv}(s)), \mathsf{e}) \iff A_{\mathsf{e}}(x).$$

The ten-field fit is: traditional target terminal-group conjugation identity; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist; carrier A_{e}; classifier R_{e}; stability from inverse closure, two-sided contextual cancellation, product-unit split, and unit transport; ledger the conjugating endpoint s and the hidden inverse endpoint $\mathsf{GroupSingletonInv}(s)$; exactness the displayed iff; standard bridge conjugation in the one-element group; downstream consumer terminal normalizer and quotient-visibility collapse; and status $\mathsf{ProofTarget}$.

Proof. Assume $A_{\mathsf{e}}(s)$. The inverse row of Definition 41.88, applied to the concrete singleton inverse of s, supplies $A_{\mathsf{e}}(\mathsf{GroupSingletonInv}(s))$. The empty endpoint has $A_{\mathsf{e}}(\mathsf{e})$ by reflexivity, so the four context carriers needed for the two-sided contextual product-unit split are available.

Transport the displayed right endpoint along the empty unit laws, and transport the middle factor x along the right unit law, so the classifier has the two-sided product-unit shape with contexts s, e and $\mathsf{GroupSingletonInv}(s), \mathsf{e}$:

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(s, \mathsf{append}(x, \mathsf{e})), \mathsf{GroupSingletonInv}(s)), \mathsf{append}(\mathsf{append}(\mathsf{e}, \mathsf{e}), \mathsf{e})).$$

Apply Definition 41.134 with $p = x$ and $q = \mathsf{e}$. This cancels the singleton contexts and identifies the transported classifier with $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(\mathsf{e})$.

For the forward direction, project the first component of this conjunction. For the reverse direction, combine the assumed $A_{\mathsf{e}}(x)$ with the reflexive carrier witness $A_{\mathsf{e}}(\mathsf{e})$, apply the product-unit split in the constructive direction, and transport the unit laws back to the displayed conjugation endpoint against e. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier conjugation terminal collapse iff`

41.19 Singleton conjugation collapse

Theorem 41.146 (Group singleton conjugation fixed carrier)**.** *For histories $a, x : \mathsf{BHist}$ in the concrete singleton history group of Definition 41.3, if $A_{\mathsf{e}}(a)$, then*

$$R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{inv}(a)), x) \Longleftrightarrow A_{\mathsf{e}}(x).$$

The ten-field fit is: traditional target terminal-group inner action; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability from singleton multiplication closure, inverse closure, and terminal classifier exactness; ledger the hidden conjugating endpoint and constant inverse; core theorem list Definition 41.4, Definition 41.5, and Definition 41.88; exactness predicate the displayed iff; standard bridge inner action of the terminal group; status* $\mathsf{ProofTarget}$.

Proof. For the forward direction, apply the first row of Definition 41.88 to the displayed classifier. It rewrites the classifier as the pair consisting of the conjugation word carrier fact and $A_{\mathsf{e}}(x)$, so projecting the second component gives the required carrier condition on x.

For the reverse direction, start from $A_{\mathsf{e}}(a)$ and $A_{\mathsf{e}}(x)$. The multiplication-closure row of Definition 41.88 places $\mathsf{mul}(a, x)$ in A_{e}, and the inverse-closure row places $\mathsf{inv}(a)$ in A_{e}.

Apply multiplication closure once more to put $\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{inv}(a))$ in the singleton carrier. With this carrier fact and $A_{\mathsf{e}}(x)$, the terminal classifier exactness row of Definition 41.88 packages the classifier

$$R_{\mathsf{e}}(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{inv}(a)), x).$$

□

41.20 Concrete singleton normalizer action certificate

Theorem 41.147 (Group singleton normalizer action certificate)**.** *For the concrete singleton history group instance, define*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s))$$

and take the terminal normalizer carrier to be

$$N_{\mathsf{e}}(s) :\Longleftrightarrow A_{\mathsf{e}}(s).$$

If $A_{\mathsf{e}}(s)$ and $A_{\mathsf{e}}(x)$, then:

1. $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$.
2. $A_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.
3. $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$.
4. $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), x)$.

The ten-field fit is: traditional target terminal normalizer action; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability singleton multiplication closure, inverse closure, and terminal classifier exactness; ledger the conjugating singleton endpoint and the acted-on singleton endpoint; core theorem this certificate; exactness terminal normalizer preservation; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Unfold the concrete singleton definitions. The inverse operation is constant, so $\mathsf{inv}(s) = \mathsf{e}$, and the two conjugation words are formed only by multiplying histories already in the singleton carrier with e.

Apply the multiplication-closure row of Theorem 41.88 to s and x to put $\mathsf{mul}(s, x)$ in A_{e}. Apply the same row again with $\mathsf{inv}(s) = \mathsf{e}$ to obtain $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$.

For the inverse conjugating word, inverse closure gives $A_{\mathsf{e}}(\mathsf{inv}(s))$. The same two multiplication-closure applications place $\mathsf{Conj}_{\mathsf{inv}(s)}(x)$ in A_{e}.

It remains to read the classifier. The terminal classifier exactness row of Theorem 41.88 classifies any two singleton endpoints by R_{e}, so it gives $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$ from the carrier facts already built.

Applying the closure argument once more to the singleton endpoint $\mathsf{Conj}_s(x)$ gives $A_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)))$. The same classifier row then yields $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), x)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizer action certificate`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier normalizer append empty action certificate`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingleton normalizer action certificate`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerAction certificate`

Theorem 41.148 (Group singleton conjugation classifier coverage)**.** *For histories $s, x, y : \mathsf{BHist}$ in the concrete singleton history group of Definition 41.3, define*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

If $A_{\mathsf{e}}(s)$, then

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), y) \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal conjugation coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier R_{e}; stability from multiplication closure, inverse closure, and terminal classifier exactness; ledger the conjugating singleton endpoint plus the acted-on endpoint; core theorem this block; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$*.*

Proof. Unfold $\mathsf{Conj}_s(x)$. The inverse row of Definition 41.88 sends the assumption $A_{\mathsf{e}}(s)$ to $A_{\mathsf{e}}(\mathsf{inv}(s))$.

Use the multiplication-closure iff from Definition 41.88 twice. The first use rewrites $A_{\mathsf{e}}(\mathsf{mul}(s, x))$ as $A_{\mathsf{e}}(s) \wedge A_{\mathsf{e}}(x)$, and the second rewrites $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ as $A_{\mathsf{e}}(\mathsf{mul}(s, x)) \wedge A_{\mathsf{e}}(\mathsf{inv}(s))$. With the two carrier facts for s and $\mathsf{inv}(s)$ fixed, this identifies $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ with $A_{\mathsf{e}}(x)$.

Apply the terminal classifier exactness row of Definition 41.88 to replace $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$ by $A_{\mathsf{e}}(\mathsf{Conj}_s(x)) \wedge A_{\mathsf{e}}(y)$. Substituting the conjugation carrier equivalence from the previous paragraph and reassociating the conjunctions gives exactly $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier append conjugation coverage iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier conjugation coverage iff`

Definition 41.149 (Group singleton normalizer orbit relation)**.** *In the concrete singleton history group instance, with N_{e} and* Conj *fixed as in Theorem 41.147, define the terminal normalizer orbit relation by declaring $x \sim_{N_{\mathsf{e}}} y$ precisely when there exists $s : \mathsf{BHist}$ such that $N_{\mathsf{e}}(s)$ and $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$.*

Lean4 (definition) `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit`

Theorem 41.150 (Group singleton normalizer orbit exactness)**.** *For any $x, y : \mathsf{BHist}$ with $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$, the terminal normalizer orbit relation is exact: $x \sim_{N_{\mathsf{e}}} y$ if and only if $R_{\mathsf{e}}(x, y)$. The ten-field fit is: traditional target terminal normalizer orbit; BEDC source concrete singleton* $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier the displayed orbit relation compared with R_{e}; stability from Theorem 41.147 and Theorem 41.88; ledger conjugator endpoint; core theorem this block; exactness orbit iff terminal singleton classifier; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$*.*

Proof. Assume first $x \sim_{N_{\mathsf{e}}} y$. By Definition 41.149, choose $s : \mathsf{BHist}$ with $N_{\mathsf{e}}(s)$ and $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$. Since $N_{\mathsf{e}}(s)$ means $A_{\mathsf{e}}(s)$, Theorem 41.147 applied to s and x gives $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$.

Use the carrier-aware classifier rows of Theorem 41.88. Symmetry turns $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$ into $R_{\mathsf{e}}(x, \mathsf{Conj}_s(x))$, and transitivity with the orbit witness $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$ gives $R_{\mathsf{e}}(x, y)$.

Conversely assume $R_{\mathsf{e}}(x, y)$. Take the witness $s = \mathsf{e}$. The singleton carrier inhabitance row of Theorem 41.88 gives $A_{\mathsf{e}}(\mathsf{e})$, hence $N_{\mathsf{e}}(\mathsf{e})$.

Theorem 41.147, applied to e and x, gives $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{e}}(x), x)$. Transitivity with the assumed $R_{\mathsf{e}}(x, y)$ yields $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{e}}(x), y)$, so Definition 41.149 supplies the orbit witness and proves $x \sim_{N_{\mathsf{e}}} y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier normalizer orbit source exact iff`

Theorem 41.151 (Group singleton normalizer orbit total coverage)**.** *For all histories $x, y : \mathsf{BHist}$ in the concrete singleton history group, with the orbit relation of Definition 41.149,*

$$x \sim_{N_{\mathsf{e}}} y \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal normalizer-orbit coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier the orbit relation; stability from Definition 41.148 and Definition 41.88; ledger the existential conjugator endpoint; core theorem this block; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$*.*

Proof. Assume first $x \sim_{N_{\mathsf{e}}} y$. By Definition 41.149, choose $s : \mathsf{BHist}$ with $N_{\mathsf{e}}(s)$ and $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$. Since $N_{\mathsf{e}}(s)$ means $A_{\mathsf{e}}(s)$, Theorem 41.148 applies to this conjugator.

The forward direction of that coverage theorem sends the orbit witness $R_{\mathsf{e}}(\mathsf{Conj}_s(x), y)$ to $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$.

Conversely assume $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$. The singleton carrier inhabitance row of Theorem 41.88 gives $A_{\mathsf{e}}(\mathsf{e})$, hence $N_{\mathsf{e}}(\mathsf{e})$.

Apply the reverse direction of Theorem 41.148 with $s = \mathsf{e}$. The endpoint facts $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$ yield $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{e}}(x), y)$, and Definition 41.149 supplies the existential orbit witness. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit coverage iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit total coverage iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier normalizer orbit total coverage iff`

Theorem 41.152 (Group singleton normalizer orbit visible source contradiction)**.** *For histories $p, y : \mathsf{BHist}$, neither visible singleton endpoint can be the source of a terminal singleton normalizer orbit into y:*

$$(\exists s,\ A_{\mathsf{e}}(s) \wedge R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(s, \mathsf{E0}(p)), \mathsf{e}), y)) \Longrightarrow \bot$$

and

$$(\exists s,\ A_{\mathsf{e}}(s) \wedge R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(s, \mathsf{E1}(p)), \mathsf{e}), y)) \Longrightarrow \bot.$$

Proof. Unpack the orbit witness. The terminal classifier says that the conjugation display is in the singleton carrier. Applying append-empty decomposition first to the terminal append and then to the inner append forces either $\mathsf{E0}(p)\mathsf{hsamee}$ or $\mathsf{E1}(p)\mathsf{hsamee}$. Both contradict constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit visible source absurd`

Theorem 41.153 (Group singleton commutator normalizer-orbit kernel exactness)**.** *For histories $x, y : \mathsf{BHist}$ in the concrete singleton history group, with $\mathsf{Comm_e}(x, y)$ defined as in Definition 41.42 and with the terminal normalizer orbit relation of Definition 41.149,*

$$\mathsf{Comm_e}(x, y) \sim_{N_\mathsf{e}} \mathsf{e} \iff A_\mathsf{e}(x) \wedge A_\mathsf{e}(y).$$

Moreover the same endpoint is the terminal orbit kernel exactly when the two inputs lie in the same terminal normalizer orbit:

$$\mathsf{Comm_e}(x, y) \sim_{N_\mathsf{e}} \mathsf{e} \iff x \sim_{N_\mathsf{e}} y.$$

The ten-field fit is: traditional target terminal commutator orbit-kernel exactness; BEDC source concrete singleton $\mathsf{Group}^\uparrow$ *over* BHist*; carrier A_e; classifier the terminal normalizer orbit relation; stability from total orbit coverage and commutator carrier exactness; ledger the commutator endpoint together with the terminal endpoint* e*; core theorem this block; exactness the two displayed equivalences; downstream consumer* $\mathsf{QuotientGroup}^\uparrow$ *first-isomorphism-style quotient-kernel exactness; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.151 to the pair $(\mathsf{Comm_e}(x, y), \mathsf{e})$. The singleton carrier inhabitance row of Definition 41.88 gives $A_\mathsf{e}(\mathsf{e})$, so the orbit coverage statement reduces the commutator orbit edge to the carrier condition $A_\mathsf{e}(\mathsf{Comm_e}(x, y))$.

Now compose this reduction with Definition 41.44. That theorem identifies the commutator carrier condition with $A_\mathsf{e}(x) \wedge A_\mathsf{e}(y)$, giving the first displayed equivalence.

For the comparison with the terminal orbit of the two inputs, apply Definition 41.151 again, now to the pair (x, y). It identifies $x \sim_{N_\mathsf{e}} y$ with the same carrier pair $A_\mathsf{e}(x) \wedge A_\mathsf{e}(y)$.

Combining the two identifications through the common carrier pair gives the second displayed equivalence, with no additional group data beyond the singleton normalizer orbit certificate. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonCommutator normalizer orbit kernel exactness iff`

Theorem 41.154 (Group singleton normalizer orbit equivalence)**.** *For histories $x, y, z : \mathsf{BHist}$ carried by A_e in the concrete singleton history group, the terminal normalizer orbit relation of Definition 41.149 is reflexive, symmetric, and transitive on the carried endpoints:*

1. $x \sim_{N_\mathsf{e}} x$.
2. *If $x \sim_{N_\mathsf{e}} y$, then $y \sim_{N_\mathsf{e}} x$.*
3. *If $x \sim_{N_\mathsf{e}} y$ and $y \sim_{N_\mathsf{e}} z$, then $x \sim_{N_\mathsf{e}} z$.*

The ten-field fit is: traditional target terminal normalizer orbit equivalence; BEDC source concrete singleton $\mathsf{Group}^\uparrow$ *over* BHist*; carrier A_e; classifier the orbit relation compared with R_e; stability from Theorem 41.150 plus the SemanticNameCert reflexivity, symmetry, and transitivity rows for R_e in Theorem 41.88; ledger the conjugator endpoint carried by each orbit witness; core theorem this block; exactness the displayed orbit equivalence laws; downstream consumer* $\mathsf{QuotientGroup}^\uparrow$*; status* $\mathsf{ProofTarget}$*.*

Proof. For reflexivity, use the reflexivity row of Theorem 41.88 on $A_\mathsf{e}(x)$ to obtain $R_\mathsf{e}(x, x)$. The reverse direction of Theorem 41.150, applied at (x, x), transports this classifier fact to $x \sim_{N_\mathsf{e}} x$.

For symmetry, assume $x \sim_{N_\mathsf{e}} y$. The forward direction of Theorem 41.150, applied at (x, y), gives $R_\mathsf{e}(x, y)$. The symmetry row of Theorem 41.88 gives $R_\mathsf{e}(y, x)$, and the reverse direction of orbit exactness at (y, x) gives $y \sim_{N_\mathsf{e}} x$.

For transitivity, assume $x \sim_{N_\mathsf{e}} y$ and $y \sim_{N_\mathsf{e}} z$. Orbit exactness sends these two orbit classifiers to $R_\mathsf{e}(x, y)$ and $R_\mathsf{e}(y, z)$. The transitivity row of Theorem 41.88 gives $R_\mathsf{e}(x, z)$, and the reverse direction of orbit exactness at (x, z) gives $x \sim_{N_\mathsf{e}} z$. □

Lean4 ✓ BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit equivalence

Definition 41.155 (Concrete singleton normalizer-action group instance)**.** *The concrete terminal normalizer action over* BHist *is the data whose carrier is* $Carrier := A_{\mathsf{e}}$*, whose classifier is* $classifier := R_{\mathsf{e}}$*, whose multiplication is* $mul :=$ append*, whose unit is* $e :=$ e*, whose inverse is the constant operation* $inv(h) :=$ e*, and whose action map is*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

Lean4 (definition) BEDC.Derived.GroupUp.GroupSingletonNormalizerAction

Theorem 41.156 (Group singleton normalizer action classifier determinacy)**.** *For histories* $s, x, y :$ BHist*, if* $A_{\mathsf{e}}(s)$ *and* $R_{\mathsf{e}}(x, y)$*, then*

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)) \wedge R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), y).$$

Proof. The classifier assumption $R_{\mathsf{e}}(x, y)$ carries the singleton endpoint facts $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$ by the forward direction of the terminal classifier row in Theorem 41.88.

Instantiate Theorem 41.147 at (s, x) and (s, y). The two instances give $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ and $A_{\mathsf{e}}(\mathsf{Conj}_s(y))$, so the SemanticNameCert classifier symmetry/transitivity rows for R_{e} classify the two action endpoints:

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)).$$

The (s, x) instance also gives the inverse-roundtrip classifier $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), x)$. Composing this row with the input classifier $R_{\mathsf{e}}(x, y)$ by the SemanticNameCert transitivity row for R_{e} yields

$$R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), y).$$

The two displayed classifiers are exactly the required conjunction. □

Lean4 ✓ BEDC.Derived.GroupUp.GroupSingletonClassifier conjugation action determinacy

Theorem 41.157 (Group singleton normalizer orbit action invariance)**.** *For histories* $s, x, y :$ BHist *in the concrete singleton history group instance, if* $A_{\mathsf{e}}(s)$*,* $A_{\mathsf{e}}(x)$*,* $A_{\mathsf{e}}(y)$*, and* $x \sim_{N_{\mathsf{e}}} y$*, then*

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y).$$

The ten-field fit is: traditional target normalizer action descending to the terminal orbit quotient; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier the normalizer orbit relation exact against* R_{e}*; stability from Theorem 41.150, Theorem 41.156, and Theorem 41.88; ledger the conjugating endpoint and orbit witness; core theorem this block; exactness the displayed orbit preservation; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* ProofTarget*.*

Proof. Start with the given orbit witness $x \sim_{N_{\mathsf{e}}} y$. Since x and y are in A_{e}, Theorem 41.150 turns that witness into the terminal classifier row $R_{\mathsf{e}}(x, y)$.

Apply Theorem 41.156 with the conjugating endpoint s. The hypothesis $A_{\mathsf{e}}(s)$ transports $R_{\mathsf{e}}(x, y)$ to

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), \mathsf{Conj}_s(y)).$$

It remains to use orbit exactness in the reverse direction. The terminal classifier package supplies the carrier rows for the two acted endpoints from the displayed R_{e} row, and then Theorem 41.150 repacks that classifier row as $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action invariance`

Theorem 41.158 (Group singleton normalizer action composition collapse)**.** *For histories* $s, t, x : \mathsf{BHist}$ *in the concrete singleton history group instance, define*

$$\mathsf{Conj}_u(y) := \mathsf{mul}(\mathsf{mul}(u, y), \mathsf{inv}(u)).$$

If $A_{\mathsf{e}}(s)$*,* $A_{\mathsf{e}}(t)$*, and* $A_{\mathsf{e}}(x)$*, then:*

1. $A_{\mathsf{e}}(\mathsf{Conj}_t(x))$.
2. $A_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.
3. $A_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x))$.
4. $R_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)), x)$.
5. $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), x)$.

The fit names the traditional target as terminal normalizer composition; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability from singleton multiplication closure and terminal classifier exactness; ledger the two conjugating singleton endpoints and the acted-on singleton endpoint; core theorem this block; exactness predicate the displayed carrier and classifier collapse; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *orbit classification; and status* $\mathsf{ProofTarget}$.

Proof. First apply Theorem 41.147 to t and x. The hypotheses $A_{\mathsf{e}}(t)$ and $A_{\mathsf{e}}(x)$ give $A_{\mathsf{e}}(\mathsf{Conj}_t(x))$.

Use that carrier fact as the acted-on endpoint for a second application of Theorem 41.147, now with conjugator s. Together with $A_{\mathsf{e}}(s)$ it gives $A_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)))$.

Next apply the multiplication-closure row of Theorem 41.88 to s and t. This places $\mathsf{mul}(s, t)$ in A_{e}. Applying Theorem 41.147 to $\mathsf{mul}(s, t)$ and x gives $A_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x))$.

It remains to read the terminal classifier. The first row of Theorem 41.88 classifies any two singleton endpoints. Apply it to the nested endpoint and x, then to the product-conjugator endpoint and x, using the carrier facts already displayed. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerAction composition collapse`

Theorem 41.159 (Group singleton normalizer action product-coherent orbit collapse)**.** *For histories* $s, t, x : \mathsf{BHist}$ *in the concrete singleton history group instance, write*

$$\mathsf{Conj}_u(y) := \mathsf{mul}(\mathsf{mul}(u, y), \mathsf{inv}(u)).$$

If $A_{\mathsf{e}}(s)$*,* $A_{\mathsf{e}}(t)$*, and* $A_{\mathsf{e}}(x)$*, then*

$$R_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)), \mathsf{Conj}_{\mathsf{mul}(s,t)}(x))$$

and

$$R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), \mathsf{Conj}_s(\mathsf{Conj}_t(x))).$$

The fit names the traditional target as product-coherent terminal normalizer action; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability from Theorem 41.158; ledger the nested action endpoint and the product-conjugator endpoint; core theorem this block; exactness predicate mutual singleton classification of the two endpoints; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *orbit collapse; and status* $\mathsf{ProofTarget}$.

Proof. Apply Theorem 41.158 to s, t, and x. Its carrier rows place both $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$ and $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$ in A_{e}.

The terminal classifier exactness row of Theorem 41.88 classifies any two singleton endpoints. Applied to those two carrier facts, it gives $R_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)), \mathsf{Conj}_{\mathsf{mul}(s,t)}(x))$.

For the reverse classifier, use the symmetry row of the same terminal classifier package on the displayed forward classifier. This yields $R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{mul}(s,t)}(x), \mathsf{Conj}_s(\mathsf{Conj}_t(x)))$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerAction product coherent orbit collapse`

Theorem 41.160 (Group singleton normalizer action product-coherent orbit law)**.** *For histories $s, t, x : \mathsf{BHist}$ in the concrete singleton history group instance, write*

$$\mathsf{Conj}_u(y) := \mathsf{mul}(\mathsf{mul}(u, y), \mathsf{inv}(u)).$$

If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(t)$, and $A_{\mathsf{e}}(x)$, then

$$\mathsf{Conj}_s(\mathsf{Conj}_t(x)) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{mul}(s,t)}(x) \wedge \mathsf{Conj}_{\mathsf{mul}(s,t)}(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(\mathsf{Conj}_t(x)).$$

The ten-field fit names the traditional target as terminal normalizer action composition on orbits; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier $\sim_{N_{\mathsf{e}}}$; stability from orbit exactness and product-coherent classifier collapse; ledger the two conjugators, product conjugator, and acted endpoint; core theorem this block; exactness the displayed orbit-level product law; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *action composition; and status* $\mathsf{ProofTarget}$*.*

Proof. First apply Theorem 41.158 to s, t, and x. Its carrier rows place $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$ and $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$ in A_{e}.

Next apply Theorem 41.159. This gives the two terminal classifier rows between the nested conjugation endpoint and the product-conjugator endpoint.

Finally use Theorem 41.150 in the classifier-to-orbit direction twice. The first use turns the forward R_{e} row into $\mathsf{Conj}_s(\mathsf{Conj}_t(x)) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$, and the second use turns the reverse R_{e} row into $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(\mathsf{Conj}_t(x))$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit product coherent action law`

Theorem 41.161 (Group singleton normalizer orbit action law package)**.** *For histories $s, t, x, y :$ BHist in the concrete singleton history group instance, if $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(t)$, $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(y)$, and $x \sim_{N_{\mathsf{e}}} y$, then:*

1. $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y)$.
2. $\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)) \sim_{N_{\mathsf{e}}} y$.
3. $\mathsf{Conj}_s(\mathsf{Conj}_t(x)) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$.

The ten-field fit names the traditional target as a concrete terminal normalizer orbit action law package; the BEDC source as the concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier the normalizer orbit relation exact against R_{e}; stability from orbit exactness, classifier determinacy, and product-coherent orbit collapse; ledger the two conjugating singleton endpoints, the acted-on singleton endpoint, and the orbit witness; core theorem this block; exactness predicate the three displayed orbit action laws; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; and status* $\mathsf{ProofTarget}$*.*

Proof. First convert the input orbit witness into the concrete classifier. Since x and y lie in A_{e}, the forward direction of Theorem 41.150 gives $R_{\mathsf{e}}(x, y)$.

For the first law, apply Theorem 41.157 directly to s, x, and y. Its hypotheses are exactly $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(y)$, and the given orbit relation.

For the inverse-roundtrip law, apply Theorem 41.156 with the conjugating endpoint s and the classifier row $R_{\mathsf{e}}(x, y)$. The second row gives

$$R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), y).$$

The reverse direction of Theorem 41.150 repacks this classifier as $\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)) \sim_{N_{\mathsf{e}}} y$.

For the product law, apply Theorem 41.159 to s, t, and x. Its forward classifier row is

$$R_{\mathsf{e}}(\mathsf{Conj}_s(\mathsf{Conj}_t(x)), \mathsf{Conj}_{\mathsf{mul}(s,t)}(x)).$$

Again use the reverse direction of orbit exactness to convert that classifier into the displayed orbit relation. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action law package`

Theorem 41.162 (Group singleton normalizer orbit action unital product package)**.** *For histories $s, t, x, y : \mathsf{BHist}$ in the concrete singleton history group instance, if $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(t)$, $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(y)$, and $x \sim_{N_{\mathsf{e}}} y$, then:*

1. $\mathsf{Conj}_{\mathsf{e}}(x) \sim_{N_{\mathsf{e}}} x$ *and* $x \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{e}}(x)$.
2. $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y)$.
3. $\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)) \sim_{N_{\mathsf{e}}} y$.
4. $\mathsf{Conj}_s(\mathsf{Conj}_t(x)) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$ *and* $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(\mathsf{Conj}_t(x))$.

The ten-field fit names the traditional target as terminal normalizer action as a unital product action on the orbit quotient; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier $\sim_{N_{\mathsf{e}}}$; stability from orbit exactness and the normalizer action rows; ledger the identity conjugator, two product conjugators, acted endpoint, and orbit witness; core theorem list Theorem 41.147, Theorem 41.150, Theorem 41.157, Theorem 41.156, and Theorem 41.160; exactness the displayed package; standard bridge the terminal group action on its one-point orbit; and status* $\mathsf{ProofTarget}$.

Proof. For the identity rows, apply Theorem 41.147 with conjugator e and endpoint x. It gives the carrier data for $\mathsf{Conj}_{\mathsf{e}}(x)$ and the classifier row

$$R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{e}}(x), x).$$

The reverse direction of Theorem 41.150 converts this classifier into $\mathsf{Conj}_{\mathsf{e}}(x) \sim_{N_{\mathsf{e}}} x$.

For the opposite identity orientation, use the singleton classifier symmetry row already available to the action certificate to turn the displayed classifier into $R_{\mathsf{e}}(x, \mathsf{Conj}_{\mathsf{e}}(x))$. A second reverse use of Theorem 41.150 gives $x \sim_{N_{\mathsf{e}}} \mathsf{Conj}_{\mathsf{e}}(x)$.

For action invariance, apply Theorem 41.157 directly to s, x, and y. The hypotheses are the carrier rows for s, x, and y, together with the given orbit witness.

For the inverse-roundtrip row, first convert the orbit witness $x \sim_{N_{\mathsf{e}}} y$ to $R_{\mathsf{e}}(x, y)$ by the forward direction of Theorem 41.150. Then apply Theorem 41.156 to s and this classifier row. Its second classifier output is

$$R_{\mathsf{e}}(\mathsf{Conj}_{\mathsf{inv}(s)}(\mathsf{Conj}_s(x)), y).$$

The reverse direction of orbit exactness repacks this row as the displayed orbit relation.

For product coherence, apply Theorem 41.160 to s, t, and x. Its two orbit outputs are exactly the forward and reverse orientations between $\mathsf{Conj}_s(\mathsf{Conj}_t(x))$ and $\mathsf{Conj}_{\mathsf{mul}(s,t)}(x)$. □

Theorem 41.163 (Group singleton normalizer action single-orbit fiber)**.** *For histories s, x, y : BHist in the concrete singleton history group, define*

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

If $A_{\mathsf{e}}(s)$ and $A_{\mathsf{e}}(x)$, then the terminal normalizer action fiber is single-orbit exact:

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} y \iff A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal normalizer action fiber; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier $\sim_{N_{\mathsf{e}}}$; stability from Theorem 41.147, Theorem 41.150, and Theorem 41.88; ledger the conjugator, source endpoint, and target endpoint; core theorem this block; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; and status* $\mathsf{ProofTarget}$.

Proof. First apply Theorem 41.147 to s and x. This gives $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$, so the acted source endpoint is inside the singleton carrier.

For the forward direction, unpack the orbit relation at $\mathsf{Conj}_s(x)$ and y. Its witness gives a terminal classifier row whose target endpoint is y.

The classifier exactness row of Theorem 41.88 reads that row as the pair of carrier facts for the acted witness endpoint and y. Projecting the second component gives $A_{\mathsf{e}}(y)$.

For the reverse direction, assume $A_{\mathsf{e}}(y)$. The terminal classifier exactness row gives

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), y).$$

With the source carrier already built and the target carrier now assumed, Theorem 41.150 repacks this classifier as $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action single fiber iff`

Theorem 41.164 (Group singleton normalizer action orbit identity)**.** *For histories s, x, y : BHist in the concrete singleton history group instance, write*

$$\mathsf{Conj}_s(z) := \mathsf{mul}(\mathsf{mul}(s, z), \mathsf{inv}(s)).$$

If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, and $A_{\mathsf{e}}(y)$, then

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} y \iff x \sim_{N_{\mathsf{e}}} y.$$

The ten-field fit names the traditional target as terminal normalizer quotient identity action; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier the normalizer orbit relation exact against R_{e}; stability from the singleton normalizer action certificate, orbit exactness, and the terminal classifier symmetry and transitivity rows; ledger the conjugating singleton endpoint and the two orbit endpoints; core theorem this block; exactness the displayed orbit identity iff; downstream consumer the singleton* $\mathsf{QuotientGroup}^{\uparrow}$ *action classifier; and status* $\mathsf{ProofTarget}$.

Proof. First apply Theorem 41.147 to s and x. It gives both $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ and the classifier row $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$.

For the forward direction, assume $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} y$. Orbit exactness, applied at $(\mathsf{Conj}_s(x), y)$ with the carrier rows just named and $A_{\mathsf{e}}(y)$, converts this orbit edge into

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), y).$$

Use the symmetry row for the terminal classifier to turn $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$ into $R_{\mathsf{e}}(x, \mathsf{Conj}_s(x))$. Transitivity with the displayed classifier gives $R_{\mathsf{e}}(x, y)$, and the reverse direction of orbit exactness repacks this as $x \sim_{N_{\mathsf{e}}} y$.

For the reverse direction, assume $x \sim_{N_{\mathsf{e}}} y$. Orbit exactness at (x, y) turns this orbit edge into $R_{\mathsf{e}}(x, y)$.

Compose the certificate classifier $R_{\mathsf{e}}(\mathsf{Conj}_s(x), x)$ with $R_{\mathsf{e}}(x, y)$ by terminal classifier transitivity. The result is

$$R_{\mathsf{e}}(\mathsf{Conj}_s(x), y).$$

Finally use orbit exactness in the classifier-to-orbit direction at $(\mathsf{Conj}_s(x), y)$. The carrier input is $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ from the action certificate together with $A_{\mathsf{e}}(y)$, so the displayed classifier becomes $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} y$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action identity iff`

41.21 Unary append inverse obstruction

Theorem 41.165 (Unary append unit-product factor exactness)**.** *For histories $h, k : \mathsf{BHist}$, the concrete unary-history append product is classified as the monoid unit exactly when both factors are classified as the unit:*

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h,k), \mathsf{e}) \iff \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}) \wedge \mathsf{MonoidHisto}$$

Equivalently, the append operation in the full unary monoid cannot supply a non-singleton inverse pair for the concrete singleton $\mathsf{Group}^{\uparrow}$ *instance.*

Proof. Unfold the classifier from Definition 40.2. The forward direction gives the unary carrier witnesses for $\mathsf{append}(h, k)$ and e together with $\mathsf{hsame}(\mathsf{append}(h, k), \mathsf{e})$. Apply the BEDC kernel theorem `append_eq_empty_iff` to this empty-result comparison; it splits the append endpoint into $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. Repacking these two singleton endpoint witnesses with the unary carrier data gives the two displayed monoid-classifier factors.

For the reverse direction, assume the two displayed classifier factors. Their carrier fields give $\mathsf{UnaryHistory}(h)$ and $\mathsf{UnaryHistory}(k)$, while their sameness fields identify both histories with e. Unary closure of append from Definition 40.3 supplies $\mathsf{UnaryHistory}(\mathsf{append}(h, k))$, and the constructor direction of `append_eq_empty_iff` rebuilds $\mathsf{hsame}(\mathsf{append}(h, k), \mathsf{e})$. Together with $\mathsf{UnaryHistory}(\mathsf{e})$, these fields repack the displayed $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, k), \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.GroupUp.MonoidHistoryClassifier unary append unit product factor exactness`
Lean4 ✓ `BEDC.Derived.MonoidUp.unary append unit product factor exactness`

Theorem 41.166 (Unary append group subcarrier unit collapse)**.** *Let* $C : \mathsf{BHist} \to \mathsf{Prop}$ *be a concrete unary subcarrier, and let* $i : \mathsf{BHist} \to \mathsf{BHist}$ *be an inverse witness. Assume*

$$\begin{array}{ll}
\text{carrier:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow \mathsf{UnaryHistory}(h), \\
\text{unit:} & C(\mathsf{e}), \\
\text{append closure:} & \forall h, k : \mathsf{BHist},\ C(h) \Rightarrow C(k) \Rightarrow C(\mathsf{append}(h, k)), \\
\text{inverse closure:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow C(i(h)), \\
\text{inverse products:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow \\
& \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e}) \wedge \\
& \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(i(h), h), \mathsf{e}).
\end{array}$$

Then every carried element lies in the empty-unit classifier fiber:

$$\forall h : \mathsf{BHist},\ C(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

Proof. Fix $h : \mathsf{BHist}$ with $C(h)$. The inverse-product hypothesis gives the right-inverse classifier row

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e}).$$

Apply Definition 41.165 in the forward direction to this row. The resulting conjunction has left factor $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e})$, which is the required unit-fiber conclusion for the chosen carried element.

The left-inverse row, append closure, inverse closure, and explicit unit field remain part of the concrete group-interface scope. They certify that C is being tested as a carried unary append instance, while the collapse argument itself only needs the right inverse and the unit-product factor exactness theorem. □

Theorem 41.167 (Group unary append inverse obstruction replay)**.** *Any* $\mathsf{Group}^{\uparrow}$ *consumer that tests a unary append inverse field over a concrete subcarrier* $C \subseteq \mathsf{UnaryHistory}$ *must replay the inverse-product row through Definition 41.165 before it can derive the singleton-unit collapse of Definition 41.166. Thus the group obstruction depends on the concrete* $\mathsf{Monoid}^{\uparrow}$ *unit-product factor exactness surface and cannot be read directly from the inverse-field signature alone.*

Proof. Start with a $\mathsf{Group}^{\uparrow}$-facing inverse test for a carried history h. The test supplies a displayed inverse-product classifier row for $\mathsf{append}(h, i(h))$ or $\mathsf{append}(i(h), h)$, but this row still lives in the unary append monoid classifier.

Apply Definition 41.165. It is the only local result that separates a unary append product classified as e into the two unit-fiber factors for its components.

With that factorization available, apply Definition 41.166. The collapse is therefore a replay through the $\mathsf{Monoid}^{\uparrow}$ unit-product exactness theorem followed by the group subcarrier collapse theorem, not a consequence of the inverse-field row by itself. □

Fit: the traditional target is the append-inverse group interface; the BEDC source is $C \subseteq \mathsf{UnaryHistory}$; the carrier is C; the classifier is inherited from $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$; stability is supplied by append closure and inverse closure; the ledger is the inverse witness i; the core theorem is this collapse block together with Definition 41.165; exactness is unit-fiber collapse; the downstream consumers are $\mathsf{Group}^{\uparrow}$ and $\mathsf{QuotientGroup}^{\uparrow}$; the status is $\mathsf{ProofTarget}$.

Theorem 41.168 (Full unary append inverse obstruction)**.** *There is no total inverse operation on the full unary-history append carrier: for every operation* $\mathsf{inv} : \mathsf{BHist} \to \mathsf{BHist}$*, if*

$$\begin{aligned}&\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \\ &\quad \mathsf{UnaryHistory}(\mathsf{inv}(h)) \wedge \\ &\quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, \mathsf{inv}(h)), \mathsf{e}),\end{aligned}$$

then False*.*

Proof. Assume such an operation inv and its displayed total inverse law. The history $\mathsf{E1}(\mathsf{e})$ is a concrete nonempty unary history, so the law applies to $h = \mathsf{E1}(\mathsf{e})$ and gives the unit-product classifier for $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{inv}(\mathsf{E1}(\mathsf{e})))$.

Apply Definition 41.165 to that classifier. The left factor of the resulting conjunction gives

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{E1}(\mathsf{e}), \mathsf{e}).$$

Unfold Definition 40.2: the classifier contains the endpoint carrier witnesses together with an internal history-sameness witness $\mathsf{E1}(\mathsf{e})\mathsf{hsamee}$. This is impossible by Definition 20.6, since an $\mathsf{E1}$-history and e have different outer constructors. Hence the assumed total inverse law yields False. □

Lean4 ✓ `BEDC.Derived.GroupUp.unary append inverse obstruction`
Lean4 ✓ `BEDC.Derived.GroupUp.FullUnaryAppendInverse obstruction`
Lean4 ✓ `BEDC.Derived.GroupUp.UnaryHistory append inverse obstruction`
Lean4 ✓ `BEDC.Derived.GroupUp.MonoidHistoryClassifier unary append inverse obstruction`

Theorem 41.169 (Unary append inverse completion singleton boundary)**.** *Let* $C : \mathsf{BHist} \to \mathsf{Prop}$ *be a concrete subcarrier of* $\mathsf{UnaryHistory}$*, and let* $i : \mathsf{BHist} \to \mathsf{BHist}$ *be an inverse operation. Assume*

$$\begin{array}{ll}\text{carrier:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow \mathsf{UnaryHistory}(h), \\ \text{unit:} & C(\mathsf{e}), \\ \text{append closure:} & \forall h, k : \mathsf{BHist},\ C(h) \Rightarrow C(k) \Rightarrow C(\mathsf{append}(h, k)), \\ \text{inverse closure:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow C(i(h)), \\ \text{inverse products:} & \forall h : \mathsf{BHist},\ C(h) \Rightarrow \\ & \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e}) \wedge \\ & \quad \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(i(h), h), \mathsf{e}).\end{array}$$

Then every carried element lies in the singleton unit fiber:

$$\forall h : \mathsf{BHist},\ C(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

Moreover, if $C(\mathsf{E1}(\mathsf{e}))$ *is also supplied, these same rows imply* False*.*

Proof. Apply Definition 41.166 to the displayed carrier, unit, append-closure, inverse-closure, and inverse-product rows. This gives the singleton unit-fiber classifier for every carried history in C.

Now assume the additional nonempty carried row $C(\mathsf{E1}(\mathsf{e}))$. Instantiate the unit-fiber conclusion at $\mathsf{E1}(\mathsf{e})$ to obtain

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{E1}(\mathsf{e}), \mathsf{e}).$$

Unfold the concrete monoid classifier as in Definition 41.168. The classifier contains a history-sameness witness $\mathsf{E1}(\mathsf{e})\mathsf{hsamee}$ between histories with different outer constructors.

Apply Definition 20.6. The $\mathsf{E1}$ constructor cannot be identified with e, so the additional carried nonempty row contradicts the same inverse-completion package. Thus the only valid concrete unary append group carrier is the singleton unit fiber. □

Theorem 41.170 (Unary append inverse field collapses to singleton)**.** *For every operation* $i : \mathsf{BHist} \to \mathsf{BHist}$*, the right-inverse field on the full unary-history append carrier,*

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e}),$$

implies the singleton-unit collapse

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

The analogous left-inverse field,

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(i(h), h), \mathsf{e}),$$

implies the same singleton-unit collapse.

The ten-field fit is: traditional target obstruction to making the full unary append monoid into a group; BEDC source UnaryHistory*,* append*, and* MonoidHistoryClassifier*; carrier* UnaryHistory*; classifier* MonoidHistoryClassifier(UnaryHistory)*; stability from unit-product factor exactness; ledger the proposed inverse field; core theorem this block; exactness inverse-law collapse to the singleton unit class; downstream consumer the concrete* $\mathsf{Group}^{\uparrow}$ *carrier choice; status* ProofTarget*.*

Proof. Assume the right-inverse field and fix a unary history h. Specializing the field at h gives the classifier statement that $\mathsf{append}(h, i(h))$ is classified as e.

Apply Definition 41.165 to this append product. The forward direction decomposes the unit-product classifier into the two factor classifiers, and the left factor is exactly MonoidHistoryClassifier(UnaryHistory)(h,

For the left-inverse field, fix the same unary history h and specialize the field at h. Apply Definition 41.165 to $\mathsf{append}(i(h), h)$. The forward direction now gives the two factor classifiers with h as the right factor, so the right factor is the displayed singleton-unit conclusion. □

Lean4 ✓ `BEDC.Derived.GroupUp.MonoidHistoryClassifier unary append inverse field collapse`
Lean4 ✓ `BEDC.Derived.GroupUp.unary append inverse field collapses to singleton`

Theorem 41.171 (Unary append inverse field iff singleton collapse)**.** *For every operation* $i : \mathsf{BHist} \to \mathsf{BHist}$ *satisfying unary preservation*

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{UnaryHistory}(i(h)),$$

the two-sided inverse-field laws under MonoidHistoryClassifier(UnaryHistory) *are equivalent to singleton-unit collapse:*

$$\begin{gathered}(\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e})) \\ \wedge \\ (\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(i(h), h), \mathsf{e}))\end{gathered} \iff \forall h : \mathsf{BHist},\ \mathsf{Unary}$$

The ten-field fit is: traditional target exact obstruction to a full unary append group; BEDC source UnaryHistory*,* append*, and* MonoidHistoryClassifier*; carrier* UnaryHistory*; classifier* MonoidHistoryClassifier(UnaryHist *stability from unit-product factor exactness and unary preservation; ledger the inverse candidate; core theorem this block; exactness the displayed iff; downstream consumer* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{QuotientGroup}^{\uparrow}$*; status* ProofTarget*.*

Proof. For the forward direction, assume the displayed conjunction of inverse-field laws and fix a unary history h. The right-inverse conjunct gives $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, i(h)), \mathsf{e})$.

Apply Definition 41.165 to this classifier. The forward direction decomposes it into the two factor classifiers, and the left factor is exactly the displayed singleton-unit conclusion for h.

For the reverse direction, assume singleton-unit collapse and fix a unary history h. The collapse hypothesis gives $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e})$. Unary preservation gives $\mathsf{UnaryHistory}(i(h))$, and applying the same collapse hypothesis to $i(h)$ gives $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(i(h), \mathsf{e})$.

Apply the reverse direction of Definition 41.165 to these two factor classifiers. This produces the right inverse-field classifier for $\mathsf{append}(h, i(h))$.

The left inverse-field classifier is obtained from the same two factor classifiers in the opposite order. Applying the reverse direction of Definition 41.165 to the pair $i(h)$, h classifies $\mathsf{append}(i(h), h)$ as e, completing the two-sided inverse-field laws. □

Lean4 ✓ `BEDC.Derived.MonoidUp.MonoidHistoryClassifier unary append inverse field singleton collapse iff`
Lean4 ✓ `BEDC.Derived.GroupUp.unary append inverse field singleton collapse iff`
Lean4 ✓ `BEDC.Derived.GroupUp.unary append inverse field iff singleton collapse`

Theorem 41.172 (Unary append singleton collapse obstruction)**.** *The full unary-history append classifier cannot classify every unary history as the singleton unit:*

$$(\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e})) \Rightarrow \mathsf{False}.$$

The ten-field fit is: traditional target obstruction to collapsing the full unary append carrier to the unit; BEDC source UnaryHistory*; carrier* UnaryHistory*; classifier* MonoidHistoryClassifier(UnaryHistory)*; stability from the concrete classifier fields; ledger the visible unary endpoint* E1(e)*; exactness the contradiction; downstream consumer the inverse-field obstruction; status* ProofTarget*.*

Proof. Assume the displayed singleton-unit collapse. The concrete history $\mathsf{E1}(\mathsf{e})$ is unary, so the collapse hypothesis classifies

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{E1}(\mathsf{e}), \mathsf{e}).$$

Unfolding the classifier exposes the internal witness $\mathsf{E1}(\mathsf{e})\mathsf{hsamee}$, contradicting the BEDC history no-confusion theorem. □

Lean4 ✓ `BEDC.Derived.MonoidUp.MonoidHistoryClassifier unary append singleton collapse obstruction`

Theorem 41.173 (Full unary append right-inverse obstruction)**.** *For every operation* $\mathsf{inv} : \mathsf{BHist} \to \mathsf{BHist}$*, if the full unary-history append carrier carries the right-inverse law*

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(h, \mathsf{inv}(h)), \mathsf{e}),$$

then False*.*

The ten-field fit is: traditional target obstruction to a right-inverse field on the full unary append monoid; BEDC source UnaryHistory*,* append*, and* MonoidHistoryClassifier(UnaryHistory)*; carrier full unary histories; classifier* MonoidHistoryClassifier(UnaryHistory)*; stability from unit-product factor exactness; ledger the proposed inverse operation and the visible unary endpoint* E1(e)*; core theorem this block; exactness impossibility of the displayed right-inverse law; downstream consumer the concrete* $\mathsf{Group}^{\uparrow}$ *carrier boundary; status* ProofTarget*.*

Proof. Instantiate the displayed right-inverse law at the concrete unary history $\mathsf{E1}(\mathsf{e})$. Unit-product factor exactness reads the resulting classifier for $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{inv}(\mathsf{E1}(\mathsf{e})))$ back to a classifier of the visible left factor with e. Unfolding the monoid classifier exposes an internal sameness witness $\mathsf{E1}(\mathsf{e})\mathsf{hsamee}$, contradicting BEDC history no-confusion. □

Lean4 ✓ `BEDC.Derived.MonoidUp.unary append monoid right inverse law absurd`

Theorem 41.174 (Full unary append left-inverse obstruction)**.** *For every operation* $\mathsf{inv} : \mathsf{BHist} \to \mathsf{BHist}$*, if the full unary-history append carrier carries the left-inverse field*

$$\begin{aligned}\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow\\ \mathsf{UnaryHistory}(\mathsf{inv}(h)) \wedge\\ \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{append}(\mathsf{inv}(h), h), \mathsf{e}),\end{aligned}$$

then False*.*

The ten-field fit is: traditional target obstruction to the full unary append group; BEDC source $\mathsf{UnaryHistory}$*,* append*, and* $\mathsf{MonoidHistoryClassifier}$*; carrier full unary histories; classifier* $\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})$*; stability from unit-product factor exactness and singleton collapse; ledger the proposed left inverse operation; core theorem this block; exactness impossibility of the left-inverse field; downstream consumer the concrete* $\mathsf{Group}^{\uparrow}$ *carrier choice and* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$*.*

Proof. Assume such an operation inv and its displayed left-inverse field. The left-inverse half of Definition 41.171 applies to the classifier component of that field and gives singleton-unit collapse:

$$\forall h : \mathsf{BHist},\ \mathsf{UnaryHistory}(h) \Rightarrow \mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(h, \mathsf{e}).$$

Instantiate the collapse at $h = \mathsf{E1}(\mathsf{e})$, using that $\mathsf{E1}(\mathsf{e})$ is a concrete unary history. This gives

$$\mathsf{MonoidHistoryClassifier}(\mathsf{UnaryHistory})(\mathsf{E1}(\mathsf{e}), \mathsf{e}).$$

The extra $\mathsf{UnaryHistory}(\mathsf{inv}(h))$ component supplied by the displayed field is retained by the ledger but is not needed for the contradiction.

Unfold the classifier from Definition 40.2. The displayed classifier contains the internal history-sameness witness $\mathsf{E1}(\mathsf{e})\mathsf{hsamee}$. This contradicts Definition 20.6, because an $\mathsf{E1}$-history and e have different outer constructors. Hence the left-inverse field yields False. □

Lean4 ✓ `BEDC.Derived.GroupUp.FullUnaryAppendLeftInverse obstruction`
Lean4 ✓ `BEDC.Derived.GroupUp.UnaryHistory append left inverse obstruction`
Lean4 ✓ `BEDC.Derived.GroupUp.unary append left inverse field obstruction`

41.22 Terminal continuation boundary

Definition 41.175 (Group terminal continuation boundary)**.** *The terminal continuation boundary for the concrete singleton* $\mathsf{Group}^{\uparrow}$ *instance is the* BHist *surface with carrier* A_{e}*, classifier* R_{e}*, multiplication* $\mathsf{mul}(h, k) = \mathsf{append}(h, k)$*, unit* $e = \mathsf{e}$*, and inverse* $\mathsf{inv}(h) = \mathsf{e}$*. Its continuation row is read only through* $\mathsf{Cont}(P, Q, S)$ *together with the terminal collapse* $R_{\mathsf{e}}(P, Q) \iff A_{\mathsf{e}}(S)$ *supplied by Definition 41.90.*

Theorem 41.176 (Group terminal continuation boundary laws)**.** *On the boundary of Definition 41.175, the following laws are available to* $\mathsf{Group}^{\uparrow}$ *consumers:*

1. *A_{e} is inhabited and closed under* append.
2. *$e = \mathsf{e}$ is a left and right unit for* append *on A_{e}.*
3. *$\mathsf{inv}(h) = \mathsf{e}$ is closed on A_{e} and supplies both inverse classifier rows against e.*
4. *R_{e} is stable under* hsame *transport and under continuation readback through* Cont.

The ledger rows are exactly the singleton carrier, terminal classifier, continuation witness, append multiplication, empty unit, and constant inverse rows already present in the concrete singleton $\mathsf{Group}^{\uparrow}$ *instance.*

Proof. The carrier, unit, inverse, associativity, multiplication-congruence, and classifier-stability rows are the concrete rows of Definition 41.5.

Continuation readback is supplied by Definition 41.90: under a witness $\mathsf{Cont}(P, Q, S)$, the input classifier $R_{\mathsf{e}}(P, Q)$, the result classifiers, and the result carrier $A_{\mathsf{e}}(S)$ are the same terminal condition.

For append products against the unit, apply the product-unit rows such as Definition 41.91 and Definition 41.137. They reduce each displayed continuation product-unit classifier to the carrier facts on the visible endpoints, so no operation outside the boundary is needed. □

Theorem 41.177 (Group terminal continuation consumer surface)**.** *Every downstream use of this concrete* $\mathsf{Group}^{\uparrow}$ *boundary may read only the following public surface:* BHist*, the carrier A_{e}, the classifier R_{e},* Cont *and* append *as the multiplication boundary, $e = \mathsf{e}$, the constant inverse* $\mathsf{inv}(h) = \mathsf{e}$, hsame *transport, and the ledger rows cited in Definition 41.176. It does not expose non-singleton group carriers, subgroup lattices, quotient groups, or host inverse objects.*

Proof. The definition of the boundary lists exactly the exported objects. The algebraic rows of Definition 41.176 are all derived from the singleton carrier, terminal classifier, append, continuation, empty unit, constant inverse, and hsame transport.

Any consumer statement that asks for a product, unit, inverse, or continuation row can therefore be discharged by one of those ledger rows. A non-singleton carrier, subgroup lattice, quotient group, or host inverse object is not among the exported objects and is not used by the cited rows, so it has no readable role on this boundary. □

Theorem 41.178 (Group terminal continuation inverse obstruction)**.** *On the boundary of Definition 41.175, every accepted inverse read of a carried history h is classified by R_{e} against* e*. Thus a consumer row requiring an inverse endpoint outside the singleton carrier A_{e} is outside the exported* $\mathsf{Group}^{\uparrow}$ *boundary.*

Proof. The boundary definition fixes the inverse operation to the constant row $\mathsf{inv}(h) = \mathsf{e}$. By Definition 41.176, this row is closed on A_{e} and supplies both inverse classifier rows against the unit $e = \mathsf{e}$.

Continuation readback does not add another inverse endpoint. The terminal collapse theorem Definition 41.90 identifies the readback classifier with the singleton carrier condition, so every accepted continuation read remains a read against e.

Finally apply Definition 41.177. The exported surface lists the constant inverse and explicitly excludes host inverse objects. Therefore any consumer row whose inverse endpoint is not the singleton row e asks for data that the boundary does not export. □

41.23 Finite-history group-law witness

Definition 41.179 (Group finite-history law witness)**.** *The concrete finite-history law witness for* $\mathsf{Group}^{\uparrow}$ *is the singleton terminal continuation packet*

$$G_{\mathsf{e}} = (A_{\mathsf{e}}, R_{\mathsf{e}}, O_{\mathsf{e}}, e_{\mathsf{e}}, I_{\mathsf{e}}, T_{\mathsf{e}}, P_{\mathsf{e}}, N_{\mathsf{e}})$$

with the following displayed rows:

1. $A_{\mathsf{e}}(h) := \mathsf{hsame}(h, \mathsf{e})$ *is the carried finite-history endpoint.*
2. $R_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k)$ *is the restricted classifier.*
3. O_{e} *is the operation row*

$$O_{\mathsf{e}}(h, k, s) := \mathsf{Cont}(h, k, s) \wedge A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge A_{\mathsf{e}}(s).$$

4. $e_{\mathsf{e}} := \mathsf{e}$ *is the identity row.*
5. $I_{\mathsf{e}}(h) := \mathsf{e}$ *is the available inverse boundary inside the singleton carrier, and any requested inverse endpoint outside* A_{e} *is an obstruction row rather than exported inverse data.*
6. T_{e} *records componentwise* hsame *transport and displayed* Cont *replay for* $A_{\mathsf{e}}, R_{\mathsf{e}}, O_{\mathsf{e}}, e_{\mathsf{e}}, I_{\mathsf{e}}$.
7. P_{e} *records provenance from the singleton empty-history instance and terminal continuation boundary.*
8. N_{e} *is the local* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row.*

Theorem 41.180 (Group finite-history continuation law witness)**.** *Every law read from the packet* G_{e} *is witnessed by the displayed* Cont *route rather than by an abstract multiplication symbol. In particular, if* $A_{\mathsf{e}}(h)$*,* $A_{\mathsf{e}}(k)$*, and* $\mathsf{Cont}(h, k, s)$*, then the operation row exposes* $A_{\mathsf{e}}(s)$*; the identity reads are the two continuation unit rows at* e*; and the inverse reads are the constant boundary* $I_{\mathsf{e}}(h) = \mathsf{e}$ *together with the obstruction against non-singleton inverse endpoints.*

Proof. Project the witness of Definition 41.179. Its carrier, classifier, identity, and inverse rows are exactly the singleton endpoint rows of Definition 41.24 and Definition 41.175.

For the operation row, use the terminal continuation collapse of Definition 41.90. Under the displayed route $\mathsf{Cont}(h, k, s)$, the input classifier and the result carrier are the same terminal condition, so an operation read exposes $A_{\mathsf{e}}(s)$ only through the continuation ledger.

For the identity rows, use Definition 41.176. The left and right unit laws are the continuation unit rows specialized to $e_{\mathsf{e}} = \mathsf{e}$; they do not require an external operation table.

For the inverse boundary, apply Definition 41.178. The available inverse is the displayed constant endpoint e, while any inverse endpoint outside the singleton carrier is outside the exported $\mathsf{Group}^{\uparrow}$ surface. Thus the packet supplies a concrete finite-history group-law witness over the terminal continuation route. □

Theorem 41.181 (Group finite-history NameCert surface)**.** *The rows* $A_{\mathsf{e}}, R_{\mathsf{e}}, O_{\mathsf{e}}, e_{\mathsf{e}}, I_{\mathsf{e}}, T_{\mathsf{e}}, P_{\mathsf{e}}, N_{\mathsf{e}}$ *form a concrete* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *surface for the singleton finite-history law witness. A consumer may inspect carrier membership, classifier transport, continuation operation, identity, available inverse boundary, provenance, and local naming; it receives no non-singleton carrier, subgroup lattice, quotient group, arbitrary homomorphism, or host inverse object.*

Proof. Use Definition 41.180 for the operation, identity, and inverse rows. It routes each law through the displayed Cont ledger and the terminal singleton carrier.

Use Definition 41.177 for the public boundary. That theorem lists exactly the exported finite-history objects: BHist, A_{e}, R_{e}, Cont, append, e, the constant inverse row, hsame transport, and the local ledger rows.

The provenance and naming coordinates $P_{\mathsf{e}}, N_{\mathsf{e}}$ therefore source and name only the displayed singleton packet. Since the excluded algebraic objects are absent from the carrier and from the continuation boundary, the listed rows exhaust this concrete $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ surface. □

41.24 Singleton contextual exact visibility

Theorem 41.182 (Group singleton two-sided contextual continuation product-unit exact visibility)**.** *For histories $L_0, R_0, L_1, R_1, P, Q, S, T, r : \mathsf{BHist}$, if $A_{\mathsf{e}}(L_0)$, $A_{\mathsf{e}}(R_0)$, $A_{\mathsf{e}}(L_1)$, $A_{\mathsf{e}}(R_1)$, and $\mathsf{Cont}(P, Q, S)$, then*

$$R_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(L_0, \mathsf{append}(S, T)), L_1), \mathsf{append}(\mathsf{append}(R_0, \mathsf{e}), R_1))$$
$$\Longleftrightarrow$$
$$(A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T)) \wedge (\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

The ten-field fit is: traditional target terminal-group contextual product-unit exact visibility; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ over BHist plus Cont and visible continuation endpoints; carrier A_{e}; classifier R_{e}; stability from contextual product-unit coverage and visible continuation exclusion; ledger the two context pairs, continuation witness, suffix factor, and endpoint seed r; core theorem list Definition 41.135 and Definition 41.136; exactness the displayed iff package; downstream consumer concrete $\mathsf{Group}^{\uparrow}$ visibility classification; and status $\mathsf{ProofTarget}$.

Proof. For the forward direction, assume the displayed contextual classifier. Apply Definition 41.135 to the four context carrier witnesses and $\mathsf{Cont}(P, Q, S)$. Its forward direction gives the carrier triple $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T)$.

Apply Definition 41.136 to the same context data, the same continuation witness, and the displayed classifier. This supplies the two visible endpoint exclusions for $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$. Conjoining the carrier triple with these two exclusions gives the right-hand package.

For the reverse direction, assume the right-hand package. Its first component is the carrier triple $A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(T)$. Apply the reverse direction of Definition 41.135 with the four context carrier witnesses and $\mathsf{Cont}(P, Q, S)$ to reconstruct the displayed contextual classifier.

The two exclusion clauses remain part of the assumed package. They are not needed for the classifier reconstruction, but their presence is exactly what makes the right-hand side an exact visible-classification package rather than only the carrier coverage statement. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonClassifier two sided contextual continuation product unit exact visibility iff`

41.25 Singleton two-conjugator orbit coverage

Theorem 41.183 (Group singleton two-conjugator orbit coverage)**.** *For histories $s, t, x, y : \mathsf{BHist}$ in the concrete singleton history group, define*

$$\mathsf{Conj}_u(z) := \mathsf{mul}(\mathsf{mul}(u, z), \mathsf{inv}(u)).$$

If $A_{\mathsf{e}}(s)$ and $A_{\mathsf{e}}(t)$, then

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \iff A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

The ten-field fit is: traditional target terminal two-conjugator orbit coverage; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier the terminal normalizer orbit relation; stability from action carrier coverage plus total orbit coverage; ledger the two conjugating endpoints and two acted endpoints; core theorem list Definition 41.147 and Definition 41.151; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Use Definition 41.147 with s and x, and again with t and y. In the singleton carrier, the action carrier coverage identifies $A_{\mathsf{e}}(\mathsf{Conj}_s(x))$ with $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(\mathsf{Conj}_t(y))$ with $A_{\mathsf{e}}(y)$.

Apply Definition 41.151 to the pair $(\mathsf{Conj}_s(x), \mathsf{Conj}_t(y))$. It converts the orbit statement into the endpoint carrier conjunction

$$A_{\mathsf{e}}(\mathsf{Conj}_s(x)) \wedge A_{\mathsf{e}}(\mathsf{Conj}_t(y)).$$

Substitute the two carrier identifications from the action coverage step. After reassociating the conjunction, the right-hand side is exactly $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit two conjugator coverage iff`

Theorem 41.184 (Group singleton two-conjugator orbit classifier)**.** *For histories $s, t, x, y : \mathsf{BHist}$ in the concrete singleton history group, write*

$$\mathsf{Conj}_u(z) := \mathsf{mul}(\mathsf{mul}(u, z), \mathsf{inv}(u)).$$

If $A_{\mathsf{e}}(s)$ and $A_{\mathsf{e}}(t)$, then

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \iff R_{\mathsf{e}}(x, y).$$

The ten-field fit is: traditional target terminal two-conjugator orbit classifier; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$*; carrier* A_{e}*; classifier* R_{e}*; stability from normalizer action coverage and terminal classifier packaging; ledger the two conjugators and endpoint witnesses; exactness the displayed iff; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.183 under the two displayed conjugator carrier witnesses. This rewrites the normalizer orbit relation between the acted endpoints as the endpoint carrier condition

$$A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y).$$

Apply the first row of Definition 41.88 to the same endpoint pair. It identifies $R_{\mathsf{e}}(x, y)$ with $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$.

Composing the first equivalence with the reverse direction of the second gives exactly

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \iff R_{\mathsf{e}}(x, y).$$

□

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit conjugator pair classifier iff`
Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit two conjugator classifier iff`

Theorem 41.185 (Group singleton normalizer orbit action chain)**.** *For histories $s, x, y, z : \mathsf{BHist}$ carried by A_{e} in the concrete singleton history group, write*

$$\mathsf{Conj}_s(w) := \mathsf{mul}(\mathsf{mul}(s, w), \mathsf{inv}(s)).$$

If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(y)$, $A_{\mathsf{e}}(z)$, $x \sim_{N_{\mathsf{e}}} y$, and $y \sim_{N_{\mathsf{e}}} z$, then

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y) \wedge \mathsf{Conj}_s(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z) \wedge \mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z).$$

The ten-field fit is: traditional target action congruence on an orbit relation; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$*; carrier A_{e}; classifier the normalizer orbit relation exact against R_{e}; stability from the normalizer action certificate, orbit action invariance, and orbit equivalence; ledger the conjugator and two orbit witnesses; core theorem this block; exactness the displayed chain preservation; standard bridge terminal group action on its one orbit; status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action chain`

Proof. First apply Definition 41.147 to the three pairs (s, x), (s, y), and (s, z). These instances carry $\mathsf{Conj}_s(x)$, $\mathsf{Conj}_s(y)$, and $\mathsf{Conj}_s(z)$ inside A_{e}.

Next apply Definition 41.157 to the input edge $x \sim_{N_{\mathsf{e}}} y$. Apply the same theorem to the input edge $y \sim_{N_{\mathsf{e}}} z$. This gives the two acted orbit edges

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y) \quad \text{and} \quad \mathsf{Conj}_s(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z).$$

Finally use the transitivity row of Definition 41.154 on the three acted endpoints. The carrier rows supplied by the action certificate and the two acted orbit edges compose to $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z)$, and the three orbit relations form the displayed conjunction. □

Theorem 41.186 (Group singleton normalizer orbit action path iff)**.** *For histories $s, x, y, z : \mathsf{BHist}$ carried by A_{e} in the concrete singleton history group, write*

$$\mathsf{Conj}_s(w) := \mathsf{mul}(\mathsf{mul}(s, w), \mathsf{inv}(s)).$$

If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(x)$, $A_{\mathsf{e}}(y)$, and $A_{\mathsf{e}}(z)$, then the two-edge normalizer orbit path is reflected exactly by the terminal action:

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y) \wedge \mathsf{Conj}_s(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z)) \Longleftrightarrow (x \sim_{N_{\mathsf{e}}} y \wedge y \sim_{N_{\mathsf{e}}} z).$$

The transitive edge is reflected exactly as well:

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(z) \Longleftrightarrow x \sim_{N_{\mathsf{e}}} z.$$

The ten-field fit is: traditional target terminal action path exactness; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$*; carrier A_{e}; classifier $\sim_{N_{\mathsf{e}}}$; stability from action identity iff, action invariance, and orbit equivalence; ledger the conjugator plus the two source orbit witnesses; core theorem this block; exactness the two displayed iff clauses; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *action-path classifier; status* ProofTarget*.*

Proof. First apply Definition 41.147 to the three pairs (s, x), (s, y), and (s, z). This supplies the carrier rows for $\mathsf{Conj}_s(x)$, $\mathsf{Conj}_s(y)$, and $\mathsf{Conj}_s(z)$.

For the forward direction of the two-edge path equivalence, assume the two acted orbit edges. Apply Definition 41.164 to $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_s(y)$, with the carrier row for $\mathsf{Conj}_s(y)$ supplied by the action certificate, to replace the acted left endpoint by x.

Use the symmetry row of Definition 41.154 to turn this edge around. The same identity iff, now with source endpoint y and target endpoint x, replaces $\mathsf{Conj}_s(y)$ by y; symmetry turns the result back to $x \sim_{N_\mathsf{e}} y$.

The second acted edge is handled in the same way. First replace the acted left endpoint $\mathsf{Conj}_s(y)$ by y, then use orbit symmetry, the identity iff for z, and symmetry again. This gives $y \sim_{N_\mathsf{e}} z$.

For the reverse direction of the two-edge path equivalence, assume $x \sim_{N_\mathsf{e}} y$ and $y \sim_{N_\mathsf{e}} z$. Apply Definition 41.157 to each source edge. The two outputs are exactly

$$\mathsf{Conj}_s(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(y) \quad \text{and} \quad \mathsf{Conj}_s(y) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z).$$

The transitivity row of Definition 41.154 also composes these acted edges to the acted transitive edge.

For the forward direction of the transitive-edge equivalence, assume $\mathsf{Conj}_s(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z)$. The action identity iff replaces the acted left endpoint by x, again using the carrier row for $\mathsf{Conj}_s(z)$, and gives $x \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z)$.

Apply the symmetry row, the identity-iff row for z with target x, and symmetry once more to replace the remaining acted endpoint by z. The result is $x \sim_{N_\mathsf{e}} z$.

For the reverse direction of the transitive-edge equivalence, assume $x \sim_{N_\mathsf{e}} z$. Apply Definition 41.157 directly to this source edge. It gives $\mathsf{Conj}_s(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z)$, which is the acted transitive edge. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action path iff`

Theorem 41.187 (Group singleton normalizer orbit action conjugator independence)**.** *For histories $s, t, x, y, z : \mathsf{BHist}$ carried by A_e in the concrete singleton history group, write*

$$\mathsf{Conj}_u(w) := \mathsf{mul}(\mathsf{mul}(u, w), \mathsf{inv}(u)).$$

If $A_\mathsf{e}(s)$, $A_\mathsf{e}(t)$, $A_\mathsf{e}(x)$, $A_\mathsf{e}(y)$, and $A_\mathsf{e}(z)$, then the terminal two-edge normalizer action path is independent of the carried conjugator:

$$(\mathsf{Conj}_s(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(y) \wedge \mathsf{Conj}_s(y) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z)) \iff (\mathsf{Conj}_t(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_\mathsf{e}} \mathsf{Conj}_t(z)).$$

The induced transitive edge is independent of the carried conjugator as well:

$$\mathsf{Conj}_s(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_s(z) \iff \mathsf{Conj}_t(x) \sim_{N_\mathsf{e}} \mathsf{Conj}_t(z).$$

The ten-field fit is: traditional target conjugator-independent terminal action paths; BEDC source concrete singleton $\mathsf{Group}^\uparrow$*; carrier A_e; classifier $\sim_{N_\mathsf{e}}$; stability from Definition 41.186; ledger the two conjugators and the two-edge source path; core theorem this block; exactness the displayed iff clauses; downstream consumer* $\mathsf{QuotientGroup}^\uparrow$ *action-path classifier; status* $\mathsf{ProofTarget}$.

Proof. Apply Definition 41.186 with conjugator s. Its first iff identifies the s-acted two-edge path with the source path $x \sim_{N_\mathsf{e}} y \wedge y \sim_{N_\mathsf{e}} z$, and its second iff identifies the s-acted transitive edge with the source edge $x \sim_{N_\mathsf{e}} z$.

Apply the same theorem with conjugator t. Its two iff clauses identify the t-acted two-edge path and the t-acted transitive edge with the same source path and source edge.

Compose the two iff chains through those shared source classifiers. The two-edge chain gives the displayed path equivalence, and the transitive-edge chain gives the displayed endpoint equivalence. □

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingletonNormalizerOrbit action conjugator independence`

Theorem 41.188 (Group singleton two-conjugator orbit path classifier)**.** *For histories* s, t, u, x, y, z : BHist *in the concrete singleton history group, write*

$$\mathsf{Conj}_q(r) := \mathsf{mul}(\mathsf{mul}(q, r), \mathsf{inv}(q)).$$

If $A_{\mathsf{e}}(s)$*,* $A_{\mathsf{e}}(t)$*, and* $A_{\mathsf{e}}(u)$*, then the two-edge acted normalizer path with independent conjugators is classified exactly by the two terminal classifier edges:*

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z)) \iff (R_{\mathsf{e}}(x, y) \wedge R_{\mathsf{e}}(y, z)).$$

When either side of this equivalence is available, the transitive acted edge also holds:

$$\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z).$$

The ten-field fit is: traditional target terminal mixed-conjugator orbit-path exactness; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$*; carrier* A_{e}*; classifier the normalizer orbit relation exact against* R_{e}*; stability from two-conjugator orbit classification and orbit transitivity; ledger the three conjugators and two endpoint classifier witnesses; core theorem this block; exactness the displayed iff plus transitive edge; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$*; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction of the path equivalence, assume the two acted orbit edges. Apply Definition 41.184 to the (s, t, x, y) edge, using the carrier witnesses for s and t, to obtain $R_{\mathsf{e}}(x, y)$.

Apply the same classifier to the (t, u, y, z) edge, using the carrier witnesses for t and u, to obtain $R_{\mathsf{e}}(y, z)$. These two outputs are exactly the right-hand side of the displayed equivalence.

For the reverse direction, assume $R_{\mathsf{e}}(x, y)$ and $R_{\mathsf{e}}(y, z)$. Apply Definition 41.184 in the reverse direction to (s, t, x, y), and then to (t, u, y, z). The two results are precisely the acted path edges.

It remains to produce the transitive acted edge. If the acted path side is available, use the transitivity row of Definition 41.154 on the three acted endpoints. The two input edges compose to $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z)$.

If the terminal classifier side is available instead, first use the reverse direction of the two equivalence steps above to obtain the two acted path edges. The same transitivity row of Definition 41.154 then composes them to the displayed transitive acted edge. □

Theorem 41.189 (Group singleton three-conjugator orbit path compression)**.** *For histories* s, t, u, v, x, y, z, w : BHist *in the concrete singleton history group, write*

$$\mathsf{Conj}_a(b) := \mathsf{mul}(\mathsf{mul}(a, b), \mathsf{inv}(a)).$$

If $A_{\mathsf{e}}(s)$*,* $A_{\mathsf{e}}(t)$*,* $A_{\mathsf{e}}(u)$*, and* $A_{\mathsf{e}}(v)$*, then the three acted normalizer edges with independent conjugators are classified exactly by the three terminal classifier edges:*

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z) \wedge \mathsf{Conj}_u(z) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w)) \iff (R_{\mathsf{e}}(x, y) \wedge R_{\mathsf{e}}(y, z) \wedge R_{\mathsf{e}}(z, w)).$$

Moreover, any such path compresses to the endpoint acted edge and the terminal classifier edge:

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z) \wedge \mathsf{Conj}_u(z) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w)) \implies (\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w) \wedge R_{\mathsf{e}}(x, w)).$$

The ten-field fit is: traditional target terminal three-step conjugation orbit path; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* A_{e}*; classifier* R_{e} *and the normalizer orbit relation; stability from adjacent two-conjugator orbit classification plus terminal classifier transitivity; ledger the four conjugators and three path edges; core theorem this block; exactness the displayed iff plus compressed endpoint edge; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction of the path equivalence, assume the three acted orbit edges. Apply Definition 41.184 to the (s, t, x, y) edge, then to the (t, u, y, z) edge, and then to the (u, v, z, w) edge. The carrier witnesses for the corresponding conjugator pairs supply $R_{\mathsf{e}}(x, y)$, $R_{\mathsf{e}}(y, z)$, and $R_{\mathsf{e}}(z, w)$.

For the reverse direction, assume $R_{\mathsf{e}}(x, y)$, $R_{\mathsf{e}}(y, z)$, and $R_{\mathsf{e}}(z, w)$. Apply the same two-conjugator orbit classifier in the reverse direction to (s, t, x, y), (t, u, y, z), and (u, v, z, w). These are precisely the three acted normalizer edges.

It remains to compress a given acted path. Use Definition 41.188 on the first two acted edges to obtain $R_{\mathsf{e}}(x, z)$, and use the adjacent classifier on the third edge to obtain $R_{\mathsf{e}}(z, w)$.

Apply the terminal transitivity supplied by Definition 41.88. The classifier row reads the two terminal edges as singleton carrier rows for x, z, and w, and repacks them as $R_{\mathsf{e}}(x, w)$.

Finally apply Definition 41.184 in the reverse direction to (s, v, x, w) with the carrier witnesses for s and v. This gives $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w)$, and together with $R_{\mathsf{e}}(x, w)$ it is the displayed compressed conclusion. □

Theorem 41.190 (Group singleton four-conjugator orbit path compression)**.** *For histories $s, t, u, v, r, x, y, z, w, q :$ BHist in the concrete singleton history group, write*

$$\mathsf{Conj}_a(b) := \mathsf{mul}(\mathsf{mul}(a, b), \mathsf{inv}(a)).$$

If $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(t)$, $A_{\mathsf{e}}(u)$, $A_{\mathsf{e}}(v)$, and $A_{\mathsf{e}}(r)$, then the four acted normalizer edges with independent conjugators are classified exactly by the four terminal classifier edges:

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z) \wedge \mathsf{Conj}_u(z) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w) \wedge \mathsf{Conj}_v(w) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_r(q)) \iff (R_{\mathsf{e}}(x, y) \wedge R$$

Moreover, any such path compresses to the endpoint acted edge and the terminal classifier edge:

$$(\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_t(y) \wedge \mathsf{Conj}_t(y) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_u(z) \wedge \mathsf{Conj}_u(z) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_v(w) \wedge \mathsf{Conj}_v(w) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_r(q)) \implies (\mathsf{Conj}_s(x) \sim$$

The ten-field fit is: traditional target terminal four-step conjugation orbit path; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifiers R_{e} and the normalizer orbit relation; stability from the existing two- and three-conjugator classifiers plus terminal classifier transitivity; ledger the five conjugators and four path edges; core theorem this block; exactness the displayed iff plus compressed endpoint edge; downstream consumer* $\mathsf{QuotientGroup}^{\uparrow}$ *path compression; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction of the path equivalence, assume the four acted orbit edges. Apply Definition 41.189 to the first three acted edges, using the carrier witnesses for s, t, u, and v. This gives $R_{\mathsf{e}}(x, y)$, $R_{\mathsf{e}}(y, z)$, $R_{\mathsf{e}}(z, w)$, and the compressed terminal edge $R_{\mathsf{e}}(x, w)$.

Apply Definition 41.184 to the fourth acted edge (v, r, w, q), using the carrier witnesses for v and r, to obtain $R_{\mathsf{e}}(w, q)$. Together with the three terminal edges already obtained, this is the right-hand side of the displayed equivalence.

For the reverse direction, assume $R_{\mathsf{e}}(x, y)$, $R_{\mathsf{e}}(y, z)$, $R_{\mathsf{e}}(z, w)$, and $R_{\mathsf{e}}(w, q)$. Apply Definition 41.184 in the reverse direction to (s, t, x, y), (t, u, y, z), (u, v, z, w), and (v, r, w, q). These are precisely the four acted normalizer edges.

It remains to compress a given acted path. The first paragraph supplies $R_{\mathsf{e}}(x, w)$ from the first three acted edges, and the fourth acted edge supplies $R_{\mathsf{e}}(w, q)$ by the two-conjugator classifier.

Apply the terminal transitivity supplied by Definition 41.88. The two terminal edges $R_{\mathsf{e}}(x, w)$ and $R_{\mathsf{e}}(w, q)$ compose to $R_{\mathsf{e}}(x, q)$.

Finally apply Definition 41.184 in the reverse direction to (s, r, x, q) with the carrier witnesses for s and r. This gives $\mathsf{Conj}_s(x) \sim_{N_{\mathsf{e}}} \mathsf{Conj}_r(q)$, and together with $R_{\mathsf{e}}(x, q)$ it is the displayed compressed conclusion. □

41.26 Center abelian normal package

This package consumes the subgroup carrier and normal-subgroup rows of chapter 80 and the commutative operation surface of chapter 42. The center package is not a free-standing $\mathsf{Group}^{\uparrow}$ row: its normality component is a $\mathsf{Subgroup}^{\uparrow}$-style carrier obligation, and its abelian component is the $\mathsf{AbGroup}^{\uparrow}$ commutativity row restricted to the center carrier.

Theorem 41.191 (Group center abelian normal subgroup package)**.** *For a BEDC history-level group operation* mul *over* BHist *with associativity, left and right identity laws for e, multiplication congruence, and two-sided inverse laws, the center carrier* $\mathsf{Z}(G)$ *of Definition 41.106 has all normal-subgroup rows of Definition 41.107 and also the abelian row*

$$\forall x, y : \mathsf{BHist}, \quad \mathsf{Z}(G)(x) \Rightarrow \mathsf{Z}(G)(y) \Rightarrow \mathsf{hsame}(\mathsf{mul}(x, y), \mathsf{mul}(y, x)).$$

The ten-field fit is: traditional target center as an abelian normal subgroup; BEDC source $\mathsf{Group}^{\uparrow}$ *over* BHist*; carrier* $\mathsf{Z}(G)$*; classifier* hsame*; stability from the checked normal-subgroup theorem plus center commutativity at each test history; ledger central endpoints* x, y *and the two product orders* $\mathsf{mul}(x, y)$ *and* $\mathsf{mul}(y, x)$*; core theorem list Definition 41.107 and Definition 41.106; exactness the displayed commutativity row together with the normal-subgroup rows; standard bridge ordinary group center; status* ProofTarget*.*

Proof. The normal-subgroup part is exactly Definition 41.107. It supplies the unit, product, inverse, classifier transport, conjugation, and centralizer-normalizer inclusion rows for the carrier $\mathsf{Z}(G)$.

For the abelian row, fix $x, y : \mathsf{BHist}$ and assume $\mathsf{Z}(G)(x)$ and $\mathsf{Z}(G)(y)$. Unfold Definition 41.106. The witness $\mathsf{Z}(G)(x)$ is a universal commutativity certificate against every test history, so applying it to the test history y gives

$$\mathsf{hsame}(\mathsf{mul}(x, y), \mathsf{mul}(y, x)).$$

The carrier witness for y is part of the abelian normal-subgroup package interface, and it is used by the product and inverse rows inherited from Definition 41.107. The displayed pointwise commutativity itself only needs the center witness for x at the chosen test history y. □

Theorem 41.192 (Group singleton public operation surface)**.** *For the concrete singleton history group instance of Definition 41.3, the public carrier is* A_{e}*, multiplication is* append*, unit is* e*, inverse is the constant operation* $h \mapsto \mathsf{e}$*, and the public classifier is* R_{e}*. This surface is packaged by the following four clauses.*

1. *Multiplication, unit, and inverse endpoints remain in the carrier* A_{e} *and are classified by* R_{e} *whenever their inputs are carried.*
2. *Every public expression generated from carried endpoints by multiplication, unit, and inverse has readback classified with* e*.*
3. *Left and right singleton append contexts preserve and reflect the public classifier by the existing append-context cancellation rows.*
4. *The source, pattern, and ledger predicates expose only singleton carrier witnesses and append or inverse expression provenance. Hence the public surface admits no endpoint outside* A_{e}*.*

↪ variant BEDC.Derived.GroupUp.GroupUp StdBridge

Proof. The carrier, classifier, multiplication, unit, and inverse are exactly the objects named in Definition 41.3. Theorem 41.4 packages them as the singleton-history SemanticNameCert for $\mathsf{Group}^{\uparrow}$, so the source and classifier fields of the public surface are not additional data.

For the endpoint clauses, use Theorem 41.5 together with the singleton endpoint readback row recorded by BEDC.Derived.GroupUp.GroupSingletonHistory_laws. Multiplication closure carries $\mathsf{append}(h, k)$ from carried inputs; the unit row carries e; inverse closure carries the constant inverse endpoint e; and the readback row classifies each generated endpoint against e by R_{e}.

Public expressions are built by induction on the expression tree. The base case is a carried endpoint, whose singleton witness reads back to e. The unit and inverse cases use the constant endpoint e. The multiplication case applies multiplication closure to the two inducted carrier witnesses and then composes their readback classifiers through the append law supplied by the same checked singleton group rows.

For contextual visibility, apply Theorem 41.7 to the left append contexts. The right-context row in the same singleton block gives the corresponding suffix form, and the two-sided row combines the two orientations. Thus wrapping a public endpoint by singleton append contexts neither creates nor hides a classifier witness.

Finally, the no-hidden-leakage clause is just the exactness content of the singleton certificate. Source predicates expose carrier witnesses $A_{\mathsf{e}}(h)$, pattern predicates expose only the empty endpoint and append shape used by the generated expression, and ledger rows record the concrete append or constant-inverse provenance. Any admitted public endpoint therefore carries a witness to A_{e}, so no endpoint outside the singleton carrier is admitted by this operation surface. □

Definition 41.193 (Group singleton terminal standard reference)**.** *Let T_G^* denote the host-facing terminal group reference with one element $\star_G$. Its equality relation is terminal equality, so every pair of host-facing elements is identified. Its multiplication row is $\star_G \cdot \star_G = \star_G$, its unit row names $\star_G$ as the unit, and its inverse row sends $\star_G$ to $\star_G$.*

The finite theorem-preservation list for T_G^* contains exactly the terminal rows matched to the public singleton surface: multiplication closure, left and right unit, associativity, two-sided inverse, and append-context cancellation. The BEDC counterparts are the clauses of Definition 41.192 backed by BEDC.Derived.GroupUp.GroupSingletonHistory_laws.

Theorem 41.194 (Group singleton terminal standard bridge)**.** *The public singleton $\mathsf{Group}^{\uparrow}$ surface of Definition 41.192 supplies a paper-level $\mathsf{StandardBridge}(\mathsf{Group}^{\uparrow}, T_G^*)$. The encode field sends the unique point $\star_G$ of T_G^* to the BEDC endpoint e. The decode field sends every endpoint carried by A_{e} to $\star_G$. The decode-encode round trip is terminal equality in T_G^*, and the encode-decode round trip is the classifier witness $R_{\mathsf{e}}(h, \mathsf{e})$ for each carried endpoint $h : A_{\mathsf{e}}$. The operation translation is the same carrier A_{e}, classifier R_{e}, multiplication* append*, unit* e*, and constant inverse from the public surface. The theorem-preservation and no-host-leak fields are the finite list and public-object restriction of Definition 41.193.*

Lean4 ✓ `BEDC.Derived.GroupUp.GroupSingleton terminal standard bridge`

Proof. We check the six fields of section 9204.1. The encode field is the constant map from T_G^* to the empty BEDC endpoint e. The endpoint e is public by Definition 41.192, so the map lands in the carrier A_{e} and respects the classifier R_{e}.

The decode field sends every carried BEDC endpoint to the unique element of T_G^*. Since the host carrier has one element, classifier-respect is terminal equality: any two decoded carried endpoints are the same host point $\star_G$.

For the decode-encode round trip, start with the unique host element, encode it as e, and decode back to $\star_G$. This is exactly terminal equality in T^*_G.

For the encode-decode round trip, start with a carried endpoint $h : A_{\mathsf{e}}$. Decoding gives $\star_G$, and encoding returns e. The singleton carrier readback clause of Definition 41.192 supplies $R_{\mathsf{e}}(h, \mathsf{e})$, which is exactly the BEDC classifier witness required for this round trip.

The operation translation keeps the BEDC public data fixed: carrier A_{e}, classifier R_{e}, multiplication append, unit e, and constant inverse. The finite theorem-preservation field is the list in Definition 41.193. Each row is read through Definition 41.192 and the checked law package BEDC.Derived.GroupUp.GroupSingletonHistory_laws. No other host group theorem is claimed by this list.

Finally, the no-host-leak field follows from the public-object inventory in Definition 41.192. The bridge uses the host terminal reference only to name the unique decode value and terminal equality. The BEDC-facing surface remains BHist, A_{e}, R_{e}, append, e, and the constant inverse, so no host primitive enters the exported $\mathsf{Group}^{\uparrow}$ object. □

Theorem 41.195 (GroupUp checked standard bridge)**.** *The singleton terminal bridge of Definition 41.194, the public singleton operation surface of Definition 41.192, and the center package of Definition 41.191 are the paper-side bridge surface whose checked Lean target is* BEDC.Derived.GroupUp.GroupUp_S *In particular, the* $\mathsf{Group}^{\uparrow}$ *bridge status is grounded by the checked standard bridge theorem, not merely by the lower-level singleton law package.*

Proof. Definition 41.194 identifies the public singleton history carrier with the terminal standard group reference. Its encode, decode, round-trip, operation-translation, theorem-preservation, and no-host-leak clauses are exactly the paper bridge fields.

Definition 41.192 supplies the public BEDC operation surface: the carrier A_{e}, classifier R_{e}, append product, empty unit, constant inverse, product closure and classification, inverse closure and classification, and append-context cancellation iff. The center package records the abelian normal center rows over the same history-level operation surface.

The checked Lean theorem BEDC.Derived.GroupUp.GroupUp_StdBridge packages these clauses as a single standard bridge. Its payload includes the $\mathsf{SemanticNameCert}$ for $\mathsf{Group}^{\uparrow}$, the empty-carrier endpoint, the product and inverse closure with their classifiers, and the append-context cancellation iff. Therefore the closure block may use the standard bridge theorem itself as the formal grounding target. □

Theorem 41.196 (Group monoid semigroup sibling route)**.** *Every public* $\mathsf{Group}^{\uparrow}$ *operation read consumes the* $\mathsf{Semigroup}^{\uparrow}$ *associativity surface of chapter 79 and the* $\mathsf{Monoid}^{\uparrow}$ *unit surface of chapter 40 before inverse rows are meaningful. The* $\mathsf{Group}^{\uparrow}$ *packet adds inverse rows to those sibling obligations; it does not replace associativity, left unit, right unit, carrier transport, continuation, provenance, or local naming rows supplied by the semigroup and monoid certificates.*

Proof. First read the binary operation as a semigroup operation. The associativity surface supplied by chapter 79 is the row that makes iterated products stable under the public carrier classifier.

Next read the same operation through chapter 40. The left and right unit rows make the empty-history unit available before inverse equations can be interpreted as group equations.

Finally attach the $\mathsf{Group}^{\uparrow}$ inverse rows. They certify left and right inverse behavior inside the already associated and unital operation surface, while the structural transport, continuation, provenance, and local naming rows remain the sibling rows already consumed by the route. □

Theorem 41.197 (Group algebra hierarchy public exit package)**.** *The public* $\mathsf{Group}^{\uparrow}$ *package exported by this chapter is exactly the union of two concrete* BHist *surfaces: the singleton empty-history carrier of Definition 41.27 and the finite-word carrier of Definition 41.34. Its multiplication, unit, inverse, product closure, inverse boundary, and* NameCert *packaging are the rows supplied by Definition 41.32, Definition 41.35, Definition 41.37, and Definition 41.38, with the monoid dependency read through Definition 41.28. No host group carrier, quotient group object, external operation table, or ambient equality is exported by the public package.*

Lean4 ✓ `BEDC.Derived.GroupUp.GroupUp StdBridge`

Proof. Project the singleton rows first. The carrier is A_{e}, the classifier is R_{e}, multiplication is append, the unit is e, and inverse is the constant empty endpoint of Definition 41.27. Definition 41.32 supplies the corresponding operation, unit, associativity, congruence, inverse, and classifier rows, while Definition 41.28 records that the append ledger is the consumed $\mathsf{Monoid}^{\uparrow}$ dependency.

Next project the finite-word rows. The carrier and classifier are the finite word spine and componentwise singleton classifier of Definition 41.34. Product closure is Definition 41.35, the reversal inverse boundary is Definition 41.37, and the resulting finite $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ packet is Definition 41.38.

These two projections exhaust the public exit: singleton endpoints and finite singleton words, both with displayed BHist ledgers. Since every operation row is read from those carriers and their classifiers, the exported package contains no additional host group carrier, quotient object, operation table, or ambient equality relation. □

Closure status for $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$**.**

Constructive story: The group surface is anchored by the singleton history group, where every carried endpoint is classified with e, multiplication is continuation append, the unit is e, and inverse is the constant return to e; on top of the general history-level laws, the center package then carves out those endpoints that commute with every test history. The singleton carrier accepts exactly histories in the empty endpoint class, and the center carrier accepts exactly displayed central endpoints for an ambient history-level group operation; endpoints lacking the singleton witness or the universal commutativity row are outside the corresponding surface. Classification is always the displayed hsame relation restricted to the admitted carrier, with multiplication congruence, inverse congruence, and centralizer transport moving representatives, rather than an imported group equality. The source specification is the monoid-plus-inverse history surface together with the singleton and center witnesses, the pattern specification records append multiplication, empty unit, constant inverse, inverse laws, commutator collapse, conjugation invariance, normalizer action, center normality, and the abelian center row, and the ledger keeps generated expressions, central endpoints, conjugation paths, and the two product orders visible. The five name-certificate obligations are witnessed by e and the inherited hsame equivalence: carrier nonemptiness comes from the empty endpoint or a displayed central endpoint, classifier reflexivity, symmetry, and transitivity are componentwise, and multiplication, inverse, conjugation, and centralizer transport keep classified representatives inside the carrier. A history h lies in the signature gap of a $\mathsf{Group}^{\uparrow}$ token p when the source-domain accepts the singleton or center package row, the signature relation pairs it with an answer group endpoint and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is the encoded singleton endpoint or central endpoint together with its operation and commutativity ledger. The checked bridge reads the singleton surface as the terminal group and the center package as the ordinary abelian normal center rows, while the chapter registers the construction under $\mathsf{Group}^{\uparrow}$ without introducing non-singleton group carriers, subgroup lattices, quotient groups, arbitrary homomorphisms, or primitive inverse objects.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 41.197, Definition 41.191, Definition 41.192, Definition 41.194, and Definition 41.195 form the mature singleton package: public carrier A_{e}, classifier R_{e}, append multiplication, empty unit, constant inverse, finite-word closure, generated-expression readback, no-hidden-leakage, the center abelian normal package, and the terminal paper bridge to T_G^*.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.GroupUp.GroupUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: Non-singleton groups, arbitrary homomorphisms, subgroup lattices, quotient groups, and general group theory are not closed here.

Upgrade path: No further upgrade: chapter at top of formal axis.

Chapter 42

A Concrete Naming Certificate for $\mathsf{AbGroup}^{\uparrow}$

The abelian group interface $\mathsf{AbGroup}^{\uparrow}$ extends chapter 41 with the commutativity field $a \cdot b = b \cdot a$ under the carrier's classifier. It is the abelian specialization of the group certificate. The concrete $\mathsf{AbGroup}^{\uparrow}$ instances in this chapter are the singleton BHist instances below, while the signed-unary $\mathsf{Int}^{\uparrow}$ carrier contributes the fixed-sign commutativity boundary recorded in Definition 42.4.

42.1 Carrier definition

Definition 42.1 (AbGroup carrier)**.** *Use a group setup extended with commutativity of the binary operation.*

Lean4 ✓ `BEDC.Derived.AbGroupUp.AbGroupTasteGate single carrier alignment`

42.2 Concrete singleton history abelian group instance

Definition 42.2 (Concrete singleton history abelian group instance)**.** *Let A_{e} and R_{e} be the singleton carrier and classifier from Definition 41.3. Define the operations by*

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{inv}(h) := \mathsf{e}, \qquad e := \mathsf{e}.$$

The resulting tuple $(A_{\mathsf{e}}, R_{\mathsf{e}}, \mathsf{mul}, e, \mathsf{inv})$ is the singleton history abelian group instance.

Theorem 42.3 (Concrete singleton history abelian group laws)**.** *The tuple $(A_{\mathsf{e}}, R_{\mathsf{e}}, \mathsf{mul}, e, \mathsf{inv})$ satisfies the concrete group laws of Theorem 41.5 and the abelian commutativity law $R_{\mathsf{e}}(\mathsf{mul}(h, k), \mathsf{mul}(k, h))$ for all histories h, k with $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$.*

Proof. The group fields are exactly the concrete singleton history group laws supplied by Theorem 41.5.

For commutativity, suppose $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. These singleton carrier witnesses identify both h and k with e under hsame.

The continuation unit laws reduce $\mathsf{append}(h, k)$ and $\mathsf{append}(k, h)$ to the singleton class of e. Symmetry and transitivity of hsame then assemble the endpoint relation $\mathsf{hsame}(\mathsf{mul}(h, k), \mathsf{mul}(k, h))$.

The multiplication-closure clauses from the concrete group laws supply $A_{\mathsf{e}}(\mathsf{mul}(h, k))$ and $A_{\mathsf{e}}(\mathsf{mul}(k, h))$. Together with the endpoint relation, these are exactly the classifier witness $R_{\mathsf{e}}(\mathsf{mul}(h, k), \mathsf{mul}(k, h))$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.concrete singleton history abgroup laws`

Definition 42.4 (Integer additive AbGroup obligation boundary)**.** *The signed-unary integer carrier supplies the carrier and classifier*

$$\mathsf{IntCarrier}(s,h), \qquad \mathsf{IntClassifierSpec}((s,h),(t,k)).$$

For a fixed sign s, the same-sign append operation row is

$$\mu_s((s,h),(s,k)) = (s, \mathsf{append}(h,k)).$$

The checked commutativity boundary for this row is supplied by Definition 31.27. An AbGroup instance over the signed carrier still requires the remaining certificate rows: zero representative, inverse or negation representative, cross-sign addition, associativity, unit, inverse, and congruence rows over IntClassifierSpec*.*

The ten-field template is: traditional target the integer additive abelian group; BEDC source $\mathsf{Int}^{\uparrow}$ *over signed unary histories; carrier* IntCarrier*; classifier* IntClassifierSpec*; stability obligations the rows just listed; ledger sign and magnitude witnesses plus continuation rows; core theorem list including Definition 31.27; exactness predicate the classifier laws over the signed carrier; standard bridge the integer additive group; and status* ProofTarget*.*

Thus the $\mathsf{AbGroup}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and operation, so later Lean work can formalize BHist-anchored abelian group laws in BEDC.Derived.AbGroupUp rather than another abstract operator schema.

42.3 The certificate

Definition 42.5 (AbGroup source specification)**.** *The source is a certified group carrier plus a commutativity field.*

Definition 42.6 (AbGroup pattern specification)**.** *The pattern is group expression formation under an operation whose operands may be exchanged by certificate law.*

Definition 42.7 (AbGroup classifier specification)**.** *The classifier is the group carrier classifier.*

Definition 42.8 (AbGroup stability certificate)**.** *The stability certificate consists of:*

1. *all group stability fields*
2. *commutativity under classifier*
3. *compatibility of commutativity with inverse and multiplication congruence*

42.4 Concrete singleton empty-history abelian group instance

Definition 42.9 (Concrete singleton empty-history abelian group instance)**.** *The singleton empty-history abelian group instance is the tuple* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mu_{\mathsf{e}}, e_{\mathsf{e}}, \iota_{\mathsf{e}})$ *where*

$$A_{\mathsf{e}}(x) \iff \mathsf{hsame}(x,\mathsf{e}), \qquad \mu_{\mathsf{e}}(x,y) := \mathsf{e}, \qquad e_{\mathsf{e}} := \mathsf{e}, \qquad \iota_{\mathsf{e}}(x) := \mathsf{e}.$$

The classifier $\rho_{\mathsf{e}}(x,y)$ *holds exactly when* $A_{\mathsf{e}}(x)$*,* $A_{\mathsf{e}}(y)$*, and* $\mathsf{hsame}(x,y)$ *all hold. Equip this tuple with the AbGroup commutativity field* $\rho_{\mathsf{e}}(\mu_{\mathsf{e}}(x,y),\mu_{\mathsf{e}}(y,x))$ *and the associated compatibility fields*

for multiplication and inverse transport on the singleton carrier. Its certificate is the ten-field fit with traditional target the one-element abelian group; BEDC source BHist*; carrier* A_{e}*; classifier* ρ_{e} *with the three carrier-aware components above; stability obligations for this classifier together with commutativity; ledger constant-operation provenance; core theorem Theorem 42.10; exactness the singleton endpoint classification* $A_{\mathsf{e}}(x)$*; standard bridge the one-object abelian group; certificate site Definition 42.11; and status* CheckedCert*.*

Theorem 42.10 (Singleton empty-history abelian group laws)**.** *The tuple* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mu_{\mathsf{e}}, e_{\mathsf{e}}, \iota_{\mathsf{e}})$ *satisfies the AbGroup carrier, closure, unit, inverse, associativity, inverse, commutativity, and congruence laws for the singleton carrier* A_{e} *and the carrier-aware classifier* ρ_{e}*.*

Proof. Carrier inhabitance is $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, hence $A_{\mathsf{e}}(\mathsf{e})$. Every operation in the tuple returns e, so the carrier endpoint required for closure, unit, inverse, associativity, inverse laws, commutativity, and constant-operation congruence is always witnessed by $A_{\mathsf{e}}(\mathsf{e})$.

For equality conclusions whose right endpoint is an original variable, the corresponding carrier witness is supplied by the input hypothesis $A_{\mathsf{e}}(x)$. Together with the defining clause $A_{\mathsf{e}}(x) \iff \mathsf{hsame}(x, \mathsf{e})$, this gives the required hsame component.

It remains to check the classifier components. Since $\mu_{\mathsf{e}}(x, y)$ and $\mu_{\mathsf{e}}(y, x)$ are both definitionally e, commutativity reduces to $\rho_{\mathsf{e}}(\mathsf{e}, \mathsf{e})$. Its three components are $A_{\mathsf{e}}(\mathsf{e})$, $A_{\mathsf{e}}(\mathsf{e})$, and $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

The same reduction handles closure, associativity, inverse compatibility, and multiplication congruence: every source product, target product, and inverse is definitionally e. Whenever a law compares e with an original endpoint, the classifier is assembled from $A_{\mathsf{e}}(\mathsf{e})$, the endpoint carrier hypothesis, and the corresponding hsame witness obtained from the singleton carrier definition. □

Theorem 42.11 (Concrete singleton empty-history $\mathsf{AbGroup}^{\uparrow}$ naming certificate)**.** *The tuple* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mu_{\mathsf{e}}, e_{\mathsf{e}}, \iota_{\mathsf{e}})$ *of Definition 42.9 supplies a checked package whose* SemanticNameCert *row is carried by* A_{e} *and* ρ_{e} *and whose singleton operation rows are* $(\mu_{\mathsf{e}}, e_{\mathsf{e}}, \iota_{\mathsf{e}})$*, with source, stability, commutativity, ledger, and exactness rows forming a concrete* NameCert *licensing* $\mathsf{AbGroup}^{\uparrow}$ *over the singleton empty-history carrier* A_{e}*.*

Proof. The source row is BHist restricted by A_{e}, and the pattern row is the constant operation package $(\mu_{\mathsf{e}}, e_{\mathsf{e}}, \iota_{\mathsf{e}})$. The classifier row is the carrier-aware relation ρ_{e} from Definition 42.9.

The carrier, closure, unit, inverse, congruence, and commutativity rows are exactly the singleton empty-history AbGroup law package of Definition 42.10. That theorem also supplies the stability needed for the classifier under the displayed operations, since each operation returns e and all comparisons are repacked through A_{e} and ρ_{e}.

The ledger row is the constant-operation provenance required by Definition 42.31. Exactness is the endpoint classification already exposed by the carrier-aware classifier: a classified endpoint is precisely an endpoint in the singleton empty-history carrier, so the public $\mathsf{AbGroup}^{\uparrow}$ name has no additional hidden carrier cases. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.singleton empty history abgroup laws`

Thus the AbGroup schema is instantiated by this concrete BHist carrier and operations, so Lean formalization can target this BHist-anchored instance rather than only abstract operator fields.

Theorem 42.12 (AbGroup left-right multiplication swap)**.** *For BEDC histories equipped with an associative multiplication, commutativity, and multiplication congruence under* hsame*, the expression* $a \cdot (b \cdot c)$ *is internally the same as* $b \cdot (a \cdot c)$*.*

Proof. Reassociate $a \cdot (b \cdot c)$ to $(a \cdot b) \cdot c$, transport the left factor across commutativity, and reassociate $(b \cdot a) \cdot c$ back to $b \cdot (a \cdot c)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul left right swap`

Theorem 42.13 (AbGroup middle-four interchange)**.** *For BEDC histories equipped with an associative multiplication, commutativity, and multiplication congruence under* hsame*,*

$$\mathsf{hsame}((a \cdot b) \cdot (c \cdot d), (a \cdot c) \cdot (b \cdot d)).$$

Proof. Reassociate $(a \cdot b) \cdot (c \cdot d)$ to $a \cdot (b \cdot (c \cdot d))$. Apply the abelian left-right swap to the right factor, obtaining $a \cdot (c \cdot (b \cdot d))$, and reassociate back to $(a \cdot c) \cdot (b \cdot d)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul middle four`

Theorem 42.14 (AbGroup inverse pair multiplication collapse)**.** *For BEDC histories equipped with associative and commutative multiplication, a right identity, multiplication congruence, and right inverses,*

$$\mathsf{hsame}((a \cdot b) \cdot (a^{-1} \cdot b^{-1}), e).$$

Proof. Use middle-four interchange to regroup the expression as

$$(a \cdot a^{-1}) \cdot (b \cdot b^{-1}).$$

The two right-inverse laws collapse the factors to $e \cdot e$, and the right-identity law for e collapses the remaining product to e. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul inverse pair collapse`

Theorem 42.15 (AbGroup conjugation collapse)**.** *For BEDC histories equipped with associative and commutative multiplication, a right identity, multiplication congruence, and right inverses, every expression $a \cdot (b \cdot a^{-1})$ is internally the same as b.*

Proof. Use the abelian left-right swap to rewrite $a \cdot (b \cdot a^{-1})$ as $b \cdot (a \cdot a^{-1})$. Collapse $a \cdot a^{-1}$ to the identity by the right-inverse law, then use the right-identity law for b. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup conjugation collapse`

Theorem 42.16 (AbGroup inverse conjugation collapse)**.** *For BEDC histories equipped with associative and commutative multiplication, a right identity, multiplication congruence, and left inverses, every expression $a^{-1} \cdot (b \cdot a)$ is internally the same as b.*

Proof. Use the abelian left-right swap to rewrite $a^{-1} \cdot (b \cdot a)$ as $b \cdot (a^{-1} \cdot a)$. Collapse $a^{-1} \cdot a$ to the identity by the left-inverse law, then use the right-identity law for b. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup inverse conjugation collapse`

Theorem 42.17 (AbGroup centralizer and commutator collapse)**.** *For BEDC histories equipped with the* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, fix a :* BHist*. Let Z_a be the centralizer carrier of Definition 41.84, and let* $\mathsf{comm}(x, a)$ *be the commutator expression from Definition 41.82. Then every x :* BHist *satisfies:*

1. $Z_a(x)$.

2. $\mathsf{comm}(x, a)\mathsf{hsame}e$.

3. $\mathsf{hsame}((a \cdot x) \cdot a^{-1}, x)$.

The ten-field fit is: traditional target abelian groups have total centralizers; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *over* BHist *plus the* $\mathsf{Group}^{\uparrow}$ *centralizer carrier; carrier* Z_a*; classifier* hsame*; stability from commutativity, commutator exactness, and conjugation collapse; ledger fixed element, commuting witness, commutator spine, and inverse witness; core theorem the three displayed clauses backed by Definition 41.84, Definition 41.82, and Definition 42.15; exactness predicate total centralizer iff trivial commutator; standard bridge ordinary abelian centralizer collapse; status* $\mathsf{ProofTarget}$.

Proof. Commutativity supplies the comparison $x \cdot a\mathsf{hsame}a \cdot x$. By the definition of Z_a in Definition 41.84, this is exactly the centralizer clause for the chosen x.

Apply Definition 41.82 to the same ordered pair (x, a). The commuting witness obtained above is transported to the trivial commutator fiber, giving $\mathsf{comm}(x, a)\mathsf{hsame}e$.

Finally apply Definition 42.15 with middle element x. The abelian conjugation spine collapses $(a \cdot x) \cdot a^{-1}$ to x, which gives the inner-conjugation fixed-point clause. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer commutator collapse`

Theorem 42.18 (AbGroup centralizer normalizer collapse)**.** *For BEDC histories equipped with the* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, fix* $a : \mathsf{BHist}$*. Let* Z_a*,* N_a*, and* $\mathsf{Conj}_s(x)$ *be the carriers/action from Definition 41.84 and Definition 41.95. Then every* $s : \mathsf{BHist}$ *satisfies:*

1. $N_a(s)$.

2. For every $x : \mathsf{BHist}$ *with* $Z_a(x)$*,* $Z_a(\mathsf{Conj}_s(x))$.

3. For every $x : \mathsf{BHist}$ *with* $Z_a(x)$*,* $\mathsf{hsame}(\mathsf{Conj}_s(x), x)$.

The ten-field fit is: traditional target abelian groups have total centralizer normalizers acting trivially; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *plus* $\mathsf{Group}^{\uparrow}$ *normalizer carrier; carrier* N_a*; classifier* hsame*; stability from total centralizer, normalizer subgroup inclusion, and abelian conjugation collapse; ledger conjugating element and inverse witness; core theorem list Definition 42.17, Definition 41.98, and Definition 41.101; exactness trivial normalizer action; standard bridge ordinary abelian centralizer normalizer; status* $\mathsf{ProofTarget}$.

Proof. The first row of Definition 42.17 gives totality of Z_a: every history is a centralizer element for the fixed a. Applying this to the conjugating element gives $Z_a(s)$.

The final inclusion row of Definition 41.98 sends this centralizer witness to $N_a(s)$. Equivalently, unfolding Definition 41.95, totality applied to $\mathsf{Conj}_s(x)$ and to $\mathsf{Conj}_{\mathsf{inv}(s)}(x)$ supplies the two preservation clauses.

Now fix $x : \mathsf{BHist}$ with $Z_a(x)$. The carrier row of Definition 41.101, read with the just-built $N_a(s)$, gives $Z_a(\mathsf{Conj}_s(x))$.

Finally apply Definition 42.15 with conjugating element s and middle element x. The abelian conjugation word $\mathsf{Conj}_s(x)$ collapses to x under hsame, so the normalizer action is pointwise trivial on the centralizer carrier. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer normalizer collapse`

Theorem 42.19 (AbGroup centralizer normalizer orbit collapse)**.** *For a BEDC history-level group operation over* BHist *with the* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, fix* $a : \mathsf{BHist}$*.*

Let Z_a, N_a, Conj_s, and $\sim_{N_a}$ be the centralizer, normalizer, action, and orbit relation from Definition 41.84, Definition 41.95, and Definition 41.103. For carried centralizer histories $x, y : \mathsf{BHist}$ with $Z_a(x)$ and $Z_a(y)$,

$$x \sim_{N_a} y \iff \mathsf{hsame}(x, y).$$

The ten-field fit is: traditional target abelian normalizer orbits are singleton classifier fibers; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *plus the* $\mathsf{Group}^{\uparrow}$ *normalizer orbit relation; carrier Z_a; classifier $\sim_{N_a}$; stability from trivial conjugation action and orbit equivalence; ledger conjugator and centralizer endpoint witnesses; core theorem list Definition 42.18, Definition 41.101, and Definition 41.104; exactness predicate orbit iff* hsame*; standard bridge ordinary abelian group normalizer-orbit collapse; status* ProofTarget*.*

Proof. For the forward direction, unpack an orbit witness s for $x \sim_{N_a} y$. The witness provides $N_a(s)$, the endpoint carriers $Z_a(x)$ and $Z_a(y)$, and the comparison $\mathsf{hsame}(\mathsf{Conj}_s(x), y)$.

Apply Definition 42.18 to the same conjugator s and centralizer endpoint x. Its trivial-action row gives $\mathsf{hsame}(\mathsf{Conj}_s(x), x)$. Symmetry and transitivity of hsame combine this with the orbit endpoint comparison, yielding $\mathsf{hsame}(x, y)$.

For the reverse direction, assume $\mathsf{hsame}(x, y)$. The collapse theorem gives $N_a(e)$ for the identity conjugator. The identity-action row of Definition 41.101 gives $\mathsf{hsame}(\mathsf{Conj}_e(x), x)$ on the carried endpoint $Z_a(x)$.

Transitivity of hsame composes the identity-action comparison with the assumed $\mathsf{hsame}(x, y)$. Together with $N_a(e)$ and the endpoint witnesses $Z_a(x)$ and $Z_a(y)$, this packages the orbit witness required by Definition 41.103. The equivalence context of Definition 41.104 records that the resulting classifier is stable on the centralizer carrier. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer normalizer orbit collapse from empty unit`

Theorem 42.20 (AbGroup total centralizer-normalizer orbit collapse)**.** *For a BEDC history-level group operation over* BHist *with the* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, fix $a : \mathsf{BHist}$. Let Z_a, N_a, Conj_s, and $\sim_{N_a}$ be the centralizer, normalizer, action, and orbit relation from Definition 41.84, Definition 41.95, and Definition 41.103. Then for all histories $x, y : \mathsf{BHist}$,*

$$x \sim_{N_a} y \iff \mathsf{hsame}(x, y).$$

The ten-field fit is: traditional target abelian normalizer orbits over the whole carrier; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *plus the* $\mathsf{Group}^{\uparrow}$ *centralizer-normalizer orbit relation; carrier* BHist*; classifier $\sim_{N_a}$ exact against* hsame*; stability from total centralizer and the carried orbit collapse; ledger the two automatically supplied centralizer endpoint witnesses; core theorem list Definition 42.17 and Definition 42.19; exactness total-carrier orbit iff* hsame*; standard bridge ordinary abelian normalizer-orbit collapse; status* ProofTarget*.*

Proof. First obtain centralizer carriedness for the chosen endpoints. The totality row of Definition 42.17, read at the fixed element a, gives $Z_a(x)$ and $Z_a(y)$ for arbitrary $x, y : \mathsf{BHist}$.

For the forward direction, combine the orbit witness $x \sim_{N_a} y$ with these two centralizer endpoint witnesses. Apply Definition 42.19 in its forward direction to conclude $\mathsf{hsame}(x, y)$.

For the reverse direction, combine the assumed $\mathsf{hsame}(x, y)$ with the same two endpoint witnesses. Apply Definition 42.19 in its reverse direction to package the required orbit witness $x \sim_{N_a} y$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer normalizer orbit total collapse from empty unit`

Theorem 42.21 (AbGroup total centralizer normalizer orbit fiber determinacy)**.** *For a BEDC history-level group operation over* BHist *with the* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, fix* $a : \mathsf{BHist}$*. Let* Z_a*,* N_a*,* Conj_s*, and* $\sim_{N_a}$ *be the centralizer, normalizer, action, and orbit relation from the existing centralizer-normalizer definitions. For all histories* $x, y, z : \mathsf{BHist}$*,*

$$x \sim_{N_a} y \wedge x \sim_{N_a} z \Longrightarrow \mathsf{hsame}(y, z).$$

The ten-field fit is: traditional target abelian normalizer-orbit fiber determinacy over the whole carrier; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *plus the* $\mathsf{Group}^{\uparrow}$ *normalizer orbit relation; carrier* BHist*; classifier* hsame*; stability from total orbit collapse plus symmetry and transitivity of* hsame*; ledger the two normalizer witnesses over one anchor; core theorem list Definition 42.20 and the* hsame *equivalence laws; exactness predicate same-anchor orbit fibers collapse to one classifier class; status* ProofTarget*.*

Proof. Apply Definition 42.20 to the first orbit witness $x \sim_{N_a} y$ to obtain $\mathsf{hsame}(x, y)$. Apply the same collapse theorem to the second orbit witness $x \sim_{N_a} z$ to obtain $\mathsf{hsame}(x, z)$.

The first comparison reverses by symmetry of hsame, giving $\mathsf{hsame}(y, x)$. Transitivity of hsame composes $\mathsf{hsame}(y, x)$ with $\mathsf{hsame}(x, z)$ and yields the required classifier equality $\mathsf{hsame}(y, z)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer normalizer orbit total fiber determinacy from empty unit`

Theorem 42.22 (AbGroup centralizer normalizer orbit fiber determinacy)**.** *For a BEDC history-level group operation over* BHist *with the same* $\mathsf{Group}^{\uparrow}$ *associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, plus* $\mathsf{AbGroup}^{\uparrow}$ *commutativity under* hsame*, as in Definition 42.19, fix* $a : \mathsf{BHist}$*. Let* Z_a*,* N_a*,* Conj_s*, and* $\sim_{N_a}$ *be the centralizer, normalizer, action, and orbit relation from the existing centralizer-normalizer definitions. For histories* $x, y, z : \mathsf{BHist}$*,*

$$Z_a(x) \wedge Z_a(y) \wedge Z_a(z) \wedge x \sim_{N_a} y \wedge x \sim_{N_a} z \Longrightarrow \mathsf{hsame}(y, z).$$

The ten-field fit is: traditional target abelian normalizer-orbit fiber determinacy; BEDC source $\mathsf{AbGroup}^{\uparrow}$ *plus the* $\mathsf{Group}^{\uparrow}$ *normalizer orbit relation; carrier* Z_a*; classifier* hsame*; stability from orbit collapse plus symmetry and transitivity of* hsame*; ledger the two normalizer witnesses over one anchor; core theorem list Definition 42.19 and the* hsame *equivalence laws; exactness predicate same-anchor orbit fibers collapse to one classifier class; status* ProofTarget*.*

Proof. Start from the five displayed assumptions. Apply Definition 42.19 to the first orbit witness $x \sim_{N_a} y$, using the carrier facts $Z_a(x)$ and $Z_a(y)$, to obtain $\mathsf{hsame}(x, y)$. Apply the same collapse theorem to the second orbit witness $x \sim_{N_a} z$, using the carrier facts $Z_a(x)$ and $Z_a(z)$, to obtain $\mathsf{hsame}(x, z)$.

The first comparison reverses by symmetry of hsame, giving $\mathsf{hsame}(y, x)$. Transitivity of hsame composes $\mathsf{hsame}(y, x)$ with $\mathsf{hsame}(x, z)$ and yields the required classifier equality $\mathsf{hsame}(y, z)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup centralizer normalizer orbit fiber determinacy from empty unit`

Theorem 42.23 (AbGroup inverse preserves multiplication order)**.** *For BEDC histories equipped with group laws, commutative multiplication, and multiplication congruence under* hsame*, the inverse of a product is internally the product of the two inverses in the same order:*

$$\mathsf{hsame}((x \cdot y)^{-1}, x^{-1} \cdot y^{-1}).$$

Proof. The group inverse multiplication theorem identifies $(x \cdot y)^{-1}$ with $y^{-1} \cdot x^{-1}$. Commutativity then exchanges the two inverse factors, yielding $x^{-1} \cdot y^{-1}$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup inverse mul`

Theorem 42.24 (AbGroup right factor swap)**.** *For BEDC histories equipped with an associative multiplication, commutativity, and multiplication congruence under* hsame*,*

$$\mathsf{hsame}((a \cdot b) \cdot c, (a \cdot c) \cdot b).$$

Proof. Reassociate $(a \cdot b) \cdot c$ to $a \cdot (b \cdot c)$, commute the inner factor $b \cdot c$ to $c \cdot b$ under multiplication congruence, and reassociate back to $(a \cdot c) \cdot b$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul right factor swap`

Theorem 42.25 (AbGroup common left factor collection)**.** *For BEDC histories equipped with an associative multiplication, commutativity, and multiplication congruence under* hsame*,*

$$\mathsf{hsame}((a \cdot b) \cdot (a \cdot c), a \cdot (a \cdot (b \cdot c))).$$

Proof. Reassociate the left side to $a \cdot (b \cdot (a \cdot c))$. Apply the abelian left-right multiplication swap inside the tail to turn $b \cdot (a \cdot c)$ into $a \cdot (b \cdot c)$, and transport that tail equality through left multiplication by a. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul common left factor collect`

Theorem 42.26 (AbGroup common right factor collection)**.** *For BEDC histories equipped with an associative multiplication, commutativity, and multiplication congruence under* hsame*,*

$$\mathsf{hsame}((b \cdot a) \cdot (c \cdot a), a \cdot (a \cdot (b \cdot c))).$$

Proof. Commute the exposed right factors to rewrite the left side as $(a \cdot b) \cdot (a \cdot c)$. The common left factor collection theorem then collects the two occurrences of a and leaves the residual factor $b \cdot c$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul common right factor collect`

Theorem 42.27 (AbGroup balanced multiplication cancellation)**.** *For BEDC histories equipped with group laws, commutativity, and multiplication congruence under* hsame*, if* $a \cdot b\mathsf{hsame}c \cdot a$*, then* $b\mathsf{hsame}c$*.*

Proof. Commute the right endpoint $c \cdot a$ to $a \cdot c$. The hypothesis then has the common left factor a on both sides, so group left cancellation removes that factor and yields $b\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul balanced cancel`

Theorem 42.28 (AbGroup middle-four double cancellation)**.** *For BEDC histories equipped with group laws, commutativity, and multiplication congruence under* hsame*, if*

$$\mathsf{hsame}((a \cdot b) \cdot (c \cdot d), (a \cdot c) \cdot (u \cdot d)),$$

then $\mathsf{hsame}(b, u)$*.*

Proof. The rule depends on the AbGroup middle-four theorem and on the Group left and right cancellation theorems; it is not merely a renamed form of any one theorem. Apply Theorem 42.13 to the left endpoint of the hypothesis and compose by symmetry and transitivity of hsame. This turns the supplied comparison into

$$\mathsf{hsame}((a \cdot c) \cdot (b \cdot d), (a \cdot c) \cdot (u \cdot d)).$$

Now apply group left cancellation, Theorem 41.66, to the common left factor $a{\cdot}c$. The remaining comparison is

$$\mathsf{hsame}(b \cdot d, u \cdot d).$$

Finally apply group right cancellation, Theorem 41.116, to the common right factor d, yielding $\mathsf{hsame}(b, u)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul middle four double cancel from empty unit`

Theorem 42.29 (AbGroup transported middle-four double cancellation)**.** *For BEDC histories equipped with group laws, commutativity, and multiplication congruence under* hsame*, if*

$$\mathsf{hsame}((a \cdot b) \cdot (c \cdot d), (a \cdot u) \cdot (c \cdot v))$$

and $\mathsf{hsame}(d, v)$*, then* $\mathsf{hsame}(b, u)$*.*

Proof. Apply Theorem 42.13 to the left endpoint of the supplied comparison, and apply the same theorem to the right endpoint. After symmetry and transitivity of hsame, the comparison has common leading factor $a \cdot c$:

$$\mathsf{hsame}((a \cdot c) \cdot (b \cdot d), (a \cdot c) \cdot (u \cdot v)).$$

Use multiplication congruence with the displayed witness $\mathsf{hsame}(d, v)$, and symmetry when orienting the witness, to transport the trailing factor in the right tail from v to d. This gives

$$\mathsf{hsame}((a \cdot c) \cdot (b \cdot d), (a \cdot c) \cdot (u \cdot d)).$$

Apply Theorem 41.66 to remove the common left factor $a \cdot c$. The remaining comparison is

$$\mathsf{hsame}(b \cdot d, u \cdot d).$$

Finally apply Theorem 41.116 to remove the common right factor d, yielding $\mathsf{hsame}(b, u)$. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup mul middle four transported double cancel from empty unit`

The ten-field fit is: traditional target transported abelian double cancellation; BEDC source $\mathsf{AbGroup}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations middle-four interchange, trailing-factor transport, and group cancellation; ledger the two paired products and the transported trailing witness; core theorem list Theorems 42.13, 41.66, and 41.116; exactness predicate the displayed implication; standard bridge ordinary abelian group cancellation; status ProofTarget.

Theorem 42.30 (Nonempty history append cannot be right-absorbed)**.** *For every BEDC history h and right tail k, neither* $\mathsf{append}(\mathsf{e0}(h), k) = k$ *nor* $\mathsf{append}(\mathsf{e1}(h), k) = k$ *holds.*

Proof. Induct on the right tail k. The empty case is constructor disjointness against e. In the nonempty cases, append exposes the same outer constructor as the tail, so constructor injectivity reduces the claim to the induction hypothesis on the inner tail. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.history append nonempty left ne right`

Definition 42.31 (AbGroup ledger policy)**.** *Record group expression provenance and the commutativity law when used.*

Definition 42.32 ($\mathsf{AbGroup}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{AbGroup}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.AbGroupUp.AbGroupUp StdBridge`

Primitive boundary. $\mathsf{AbGroup}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Proposition 42.33 (AbGroup forgets to Group certificate)**.** *Let M carry an* $\mathsf{AbGroup}^{\uparrow}(M)$ *certificate. If the commutativity field and its compatibility data are omitted, the retained carrier, source, pattern, classifier, stability, and ledger fields form a* $\mathsf{Group}^{\uparrow}(M)$ *certificate. The carrier, multiplication $\cdot$, identity e, inverse operation $^{-1}$, and classifier are unchanged.*

This reduct is a concrete dependency on chapter 41: the AbGroup packet first consumes the GroupUp carrier, classifier, inverse, unit, stability, and ledger rows, then adds the commutativity row.

Proof. By Definition 42.32, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{AbGroup}^{\uparrow}$. The argument keeps the group-shaped fields among these data.

By Definition 42.1, an $\mathsf{AbGroup}^{\uparrow}$ carrier is a group setup together with commutativity of the binary operation. The underlying carrier, classifier, multiplication, identity, and inverse are therefore already the data listed in Definition 41.2.

By Definition 42.5, the source is a certified group carrier plus a commutativity field. Removing the commutativity component leaves the certified group carrier. By Definition 41.52, this is the certified monoid carrier plus inverse data required for the group source. By Definition 42.7, the classifier is the group carrier classifier, so it is exactly the classifier of Definition 41.54.

By Definition 42.6, the abelian group pattern is group expression formation with the added permission to exchange operands by the commutativity law. Omitting only that exchange permission leaves the group expression formation of Definition 41.53.

By Definition 42.8, the stability certificate contains all group stability fields, commutativity under classifier, and compatibility of commutativity with inverse and multiplication congruence. The first component is precisely the list of group stability fields in Definition 41.55; the latter components are not obligations for $\mathsf{Group}^{\uparrow}$. The remaining ledger is the group expression provenance of Definition 41.120, matching the group part of Definition 42.31.

The retained carrier, source, pattern, classifier, stability, and ledger data are exactly the obligations named in Definition 41.121. Since each retained operation and the classifier are copied from the original $\mathsf{AbGroup}^{\uparrow}(M)$ certificate, the resulting $\mathsf{Group}^{\uparrow}(M)$ source package has the same carrier operation, identity, inverse, and classifier. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.AbgroupForgetsGroupCertificate`

Theorem 42.34 (AbGroup additive negation involutivity)**.** *Let M carry an $\mathsf{AbGroup}^{\uparrow}(M)$ certificate. Write its carrier predicate as A, its classifier as $\sim_M$, its addition as $+$, its zero as 0, and its additive inverse operation as $-$. For every carried element x with $A(x)$,*

$$-(-x) \sim_M x.$$

The ten-field fit is: traditional target additive abelian groups have involutive negation; BEDC source $\mathsf{AbGroup}^{\uparrow}(M)$; carrier A; classifier $\sim_M$; stability obligations inverse closure, additive associativity, left and right zero laws, addition congruence, and the left inverse row inherited from the group reduct; ledger the two applications of the additive inverse operation; core theorem list Definition 42.33 and Definition 41.57; exactness predicate double negation returns to the input classifier fiber; standard bridge ordinary abelian-group negation; status ProofTarget.

Proof. Apply Definition 42.33 to the given $\mathsf{AbGroup}^{\uparrow}(M)$ certificate. The retained $\mathsf{Group}^{\uparrow}(M)$ package has the same carrier A, classifier $\sim_M$, additive operation $+$, zero 0, and inverse operation $-$.

Read Definition 41.57 in this retained group package, with multiplication written as $+$, identity written as 0, and inverse written as $-$. The inherited associativity, left and right zero laws, addition congruence, and left inverse row give the internal comparison

$$-(-x)\mathsf{hsame}x.$$

The inverse-closure row of the retained group package sends $A(x)$ first to $A(-x)$ and then to $A(-(-x))$. Repack these endpoint carrier witnesses with the displayed internal comparison through the unchanged carrier classifier $\sim_M$. The result is exactly $-(-x) \sim_M x$. □

Theorem 42.35 (AbGroup Group reduct commutativity centralizes commutators)**.** *Let M carry an $\mathsf{AbGroup}^{\uparrow}(M)$ certificate. In the retained $\mathsf{Group}^{\uparrow}(M)$ reduct of Definition 42.33, every displayed commutator route built from carried elements x and y is classified with the identity fiber. The route is read by first using the abelian commutativity row to identify $x + y$ with $y + x$, and then using the retained unit, inverse, associativity, and congruence rows of the group reduct to reduce the commutator word to the identity classifier. This conclusion introduces no quotient group, subgroup primitive, centralizer carrier, normalizer carrier, or host equality outside the omitted commutativity field and the retained $\mathsf{Group}^{\uparrow}(M)$ rows.*

Proof. Apply Definition 42.33. The reduct keeps the carrier, classifier, addition, zero, inverse operation, unit laws, inverse rows, associativity, and congruence data of the original $\mathsf{AbGroup}^{\uparrow}(M)$ packet.

Now read the commutativity field before the inverse rows are used. The abelian row identifies the two displayed products $x + y$ and $y + x$ in the carrier classifier, so the middle comparison in a displayed commutator word is already available on the BEDC surface.

Use the retained group rows to cancel the resulting adjacent inverse and unit terms. Associativity and congruence only reassociate and transport the displayed word; the inverse and unit rows collapse it to the identity classifier fiber.

All data used in the argument are rows of the original certificate or rows kept by the forgetful reduct. Hence the conclusion records a commutator collapse inside the retained group surface without adding a quotient construction, subgroup object, centralizer primitive, normalizer primitive, or host equality. □

Theorem 42.36 (AbGroup singleton terminal standard bridge)**.** *The public singleton empty-history* $\mathsf{AbGroup}^{\uparrow}$ *surface supplies a paper-level* $\mathsf{StandardBridge}(\mathsf{AbGroup}^{\uparrow}, T_A^*)$ *to the ordinary one-element abelian group. The encode field sends the unique host point of* T_A^* *to* e*. The decode field sends every carried endpoint in* A_{e} *to the host point. The decode-encode round trip is terminal equality, and the encode-decode round trip is the classifier witness* $\rho_{\mathsf{e}}(h, \mathsf{e})$ *for each carried endpoint* $h : A_{\mathsf{e}}$*. Operation translation reads the abelian operation as both addition and multiplication notation, sends zero and unit to* e*, and reads inverse and additive negation as the constant operation. The theorem-preservation field covers the carrier, closure, unit, inverse, associativity, commutativity, and double-negation rows. The no-host-leak field states that the exported BEDC surface remains* BHist*,* A_{e}*,* ρ_{e}*,* e*, and the displayed operations.*

Proof. We check the six fields of section 9204.1. The encode field is the constant map from the unique point of T_A^* to the empty BEDC endpoint e. The carrier and classifier laws of Definition 42.11 make this endpoint a public carried point of A_{e} and place it in the classifier fiber ρ_{e}.

The decode field sends every carried endpoint of A_{e} to the unique host point. Hence classifier-respect on decoded values and the decode-encode round trip are both terminal equality in the ordinary one-element abelian group.

For encode-decode, begin with a carried endpoint $h : A_{\mathsf{e}}$. Decoding gives the unique host point, and encoding returns e. The singleton classifier row of Definition 42.11 supplies exactly the witness $\rho_{\mathsf{e}}(h, \mathsf{e})$ required by the BEDC side of the round trip.

The operation translation keeps the BEDC public data fixed. The same singleton operation is read as addition and as multiplication notation, the zero and unit are both e, and inverse and additive negation are the constant operation. The group-shaped rows are inherited through Definition 42.33; the abelian commutativity row comes from Definition 42.11; and the double-negation row is Definition 42.34 read on the singleton surface.

Finally, no host primitive is exported. The host model supplies only the unique decode value and terminal equality used by the bridge argument. The BEDC-facing surface remains BHist, A_{e}, ρ_{e}, e, and the displayed constant operations. □

Lean4 ✓ `BEDC.Derived.AbGroupUp.abgroup singleton terminal standard bridge`

42.5 Public closure record

Theorem 42.37 (AbGroup public operation surface)**.** *The public* $\mathsf{AbGroup}^{\uparrow}$ *surface consists of the singleton empty-history carrier, classifier, and operation tuple from Definition 42.9; the checked singleton laws from Definition 42.10; the forgetful group reduct from Definition 42.33; the abelian normalizer-orbit collapse package from Definition 42.20; and the middle-four and double-cancellation normal-form rows represented by Definition 42.13, Definition 42.28, and Definition 42.29. No host equality, quotient carrier, subgroup primitive, or module equivalence is part of this surface.*

Proof. The carrier, classifier, multiplication, identity, inverse, and commutativity data are the singleton empty-history data of Definition 42.9. The checked laws in Definition 42.10 make those operations the displayed $\mathsf{AbGroup}^{\uparrow}$ singleton package.

The group-facing part is obtained by applying Definition 42.33; it keeps the same carrier, classifier, multiplication, identity, and inverse while only dropping the commutativity field as an extra obligation. The abelian operation consequences then remain visible through Definition 42.20 and the normal-form rows Definition 42.13, Definition 42.28, and Definition 42.29.

These cited rows exhaust the public operation surface asserted here: they expose the singleton carrier and laws, the group reduct, the centralizer and normalizer collapse, and the middle-four

cancellation forms. Since none of the cited rows constructs host equality, a quotient carrier, a subgroup primitive, or a module equivalence, the public surface adds none of those structures. □

Lemma 42.38 (AbGroup forgetful bridge determinacy)**.** *For the public singleton empty-history* $\mathsf{AbGroup}^{\uparrow}$ *surface, the bridge-facing* $\mathsf{Group}^{\uparrow}$ *consumer obtained by applying Definition 42.33 to Definition 42.36 is determined by the following finite local rows: the retained carrier, source, pattern, classifier, stability, and ledger fields of the forgetful reduct; the unchanged multiplication, unit, inverse, and classifier; the constant encode row to* e*; the terminal decode row; the terminal decode-encode row; the encode-decode readback row* $\rho_{\mathsf{e}}(h, \mathsf{e})$ *for each carried endpoint* $h : A_{\mathsf{e}}$*; the operation-translation row reading the same displayed operation as the group multiplication,* e *as the unit, and the constant operation as inverse; the theorem-preservation rows for carrier, closure, unit, inverse, and associativity; and the no-host-leak surface* BHist*,* A_{e}*,* ρ_{e}*,* e*, and the displayed operations. Hence every displayed route, transport, readback, or handoff available to such a consumer factors through these rows, and no unlisted bridge coordinate, host equality, quotient carrier, subgroup primitive, module equivalence, or commutativity obligation is available as part of the* $\mathsf{Group}^{\uparrow}$ *handoff.*

Proof. Apply Definition 42.33. It says exactly that the $\mathsf{Group}^{\uparrow}$ certificate retained from an $\mathsf{AbGroup}^{\uparrow}$ certificate consists of the carrier, source, pattern, classifier, stability, and ledger fields, with the carrier, multiplication, identity, inverse operation, and classifier copied unchanged. Thus any consumer-facing route or transport that asks only for a group certificate must enter through one of these retained rows.

Now read Definition 42.36 through this reduct. Its encode row is the constant map from the unique terminal host point to e, and its decode row sends every carried endpoint of A_{e} to the unique host point. The two round-trip rows are therefore fixed: decode-encode is terminal equality, while encode-decode is precisely the displayed classifier witness $\rho_{\mathsf{e}}(h, \mathsf{e})$. Consequently every readback available to the consumer factors through this single carrier-and-classifier surface.

The same theorem fixes the operation handoff. After forgetting commutativity, the group-facing operation translation is the unchanged displayed operation, with e as unit and the constant operation as inverse. The theorem- preservation rows needed by a $\mathsf{Group}^{\uparrow}$ consumer are exactly the carrier, closure, unit, inverse, and associativity rows, all of which are retained by the forgetful certificate. The abelian commutativity and double-negation rows remain rows of the abelian bridge, but they are not additional coordinates of the $\mathsf{Group}^{\uparrow}$ handoff.

Finally, the no-host-leak clause of Definition 42.36 restricts the exported BEDC-facing surface to BHist, A_{e}, ρ_{e}, e, and the displayed operations. Since Definition 42.33 adds no field while forming the reduct, a consumer has no route to any bridge coordinate outside the listed rows. The omitted commutativity obligation, commutativity-compatibility data, host equality, quotient carrier, subgroup primitive, and module equivalence are therefore unavailable as $\mathsf{Group}^{\uparrow}$ bridge handoff data. □

Definition 42.39 (AbGroup unary scalar multiple and torsion)**.** *Fix an* $\mathsf{AbGroup}^{\uparrow}$ *source package with carrier predicate* A*, classifier* ρ*, binary operation* $+$*, identity* 0*, and inverse operation* $-$*. Write* $x \sim_A y$ *for* $\rho(x, y)$*. The unary scalar-action relation* $\mathsf{AbNatAct}_A(n, a, x)$ *is the closed relation generated by:*

1. *for every carried* a*,* $\mathsf{AbNatAct}_A(\mathsf{e}, a, 0)$*;*
2. *if* $\mathsf{AbNatAct}_A(n, a, x)$ *and* $A(a)$*, then* $\mathsf{AbNatAct}_A(\mathsf{E1}(n), a, x + a)$*.*

Endpoint replacement in this relation is read through the classifier $\sim_A$ *using the congruence fields of the carrier.*

A unary history n is positive, written $\mathsf{PosUnary}(n)$*, when*

$$\mathsf{PosUnary}(n) \iff \exists t\,(\mathsf{Unary}(t) \wedge n = \mathsf{E1}(t)).$$

A multiplier n annihilates a carried element a, written $\mathsf{Ann}_A(n, a)$*, when $A(a)$ and there is an endpoint x such that* $\mathsf{AbNatAct}_A(n, a, x)$ *and $x \sim_A 0$. The torsion predicate is*

$$\mathsf{Tor}_A(a) \iff \exists n\,(\mathsf{PosUnary}(n) \wedge \mathsf{Ann}_A(n, a)).$$

Lemma 42.40 (Positive unary common product)**.** *If m and n are positive unary histories, then there is a positive unary history p and unary quotient witnesses q_m, q_n such that*

$$\mathsf{NatMul}(m, q_m, p) \qquad \textit{and} \qquad \mathsf{NatMul}(n, q_n, p).$$

Proof. Use totality of unary multiplication from Definition 64.17 to form a row product p_{mn} with $\mathsf{NatMul}(m, n, p_{mn})$ and a column product p_{nm} with $\mathsf{NatMul}(n, m, p_{nm})$.

A rectangular induction over the unary generators for m and n identifies these two products. The empty row and empty column are the zero constructor of Definition 64.1. In the one-step cases, unary continuation associativity and unary continuation commutativity from Definition 29.6 and Definition 9200.14 move the last row past the last column. Functionality of NatMul from Definition 64.12 then classifies the two resulting endpoints, giving $p_{mn}\mathsf{hsame}p_{nm}$.

Transport the column multiplication witness along this endpoint sameness by Definition 64.186, and take $p = p_{mn}$, $q_m = n$, and $q_n = m$. The result p is unary by Definition 64.18. Since m and n are positive, neither is history-same to e; if $p\mathsf{hsamee}$, then Definition 64.38 applied to $\mathsf{NatMul}(m, n, p)$ would force $n\mathsf{hsamee}$, a contradiction. Closed unary generation in Definition 28.1 therefore makes p a one-headed unary history, hence positive. □

Lemma 42.41 (AbGroup annihilator transport laws)**.** *For the torsion data of Definition 42.39:*

1. *if* $\mathsf{Ann}_A(m, a)$ *and* $\mathsf{NatMul}(m, q, p)$*, then* $\mathsf{Ann}_A(p, a)$*;*
2. *if* $\mathsf{Ann}_A(p, a)$ *and* $\mathsf{Ann}_A(p, b)$*, then* $\mathsf{Ann}_A(p, a + b)$*;*
3. *if* $\mathsf{Ann}_A(p, a)$*, then* $\mathsf{Ann}_A(p, -a)$*.*

Proof. For the first row, induct on the $\mathsf{NatMul}(m, q, p)$ witness. The empty quotient gives the zero scalar, which annihilates every carried element by the zero row of $\mathsf{AbNatAct}_A$. In the successor case, the scalar action over the appended unary multiplier decomposes into the previous block plus one m-block. The induction hypothesis classifies the previous block with 0, and the hypothesis $\mathsf{Ann}_A(m, a)$ classifies the new block with 0. The unit, associativity, and congruence fields inherited from Definition 42.1 collapse the sum of the two zero-classified blocks to 0.

For the second row, induct on the scalar-action witness for p. At the zero multiplier, both sides are the identity. In the successor case, the induction hypothesis gives the distributive comparison

$$p \cdot (a + b) \sim_A (p \cdot a) + (p \cdot b),$$

where $p \cdot a$ denotes the endpoint generated by $\mathsf{AbNatAct}_A(p, a, -)$. The next step adds one copy of $a + b$. Associativity, commutativity, and congruence regroup the endpoint as

$$(p \cdot a + a) + (p \cdot b + b),$$

which is the pair of successor scalar endpoints for a and b. The two annihilator hypotheses classify those endpoints with 0, and the unit law classifies their sum with 0.

For the third row, induct on the scalar-action witness for p. The zero multiplier gives 0. In the successor case, Definition 42.23 and commutativity identify the scalar endpoint for $-a$ with the inverse of the scalar endpoint for a. If the latter is classified with 0, inverse congruence classifies its inverse with -0. The inverse of 0 is classified with 0 by the group inverse uniqueness theorem Definition 41.59, using the group package retained by Definition 42.33. Thus the scalar endpoint for $-a$ is classified with 0. □

Theorem 42.42 (AbGroup torsion subgroup closure)**.** *Let an* $\mathsf{AbGroup}^{\uparrow}$ *carrier be written additively as in Definition 42.39. Then its torsion predicate contains* 0 *and is closed under addition and inverse: if* $\mathsf{Tor}_A(a)$ *and* $\mathsf{Tor}_A(b)$*, then* $\mathsf{Tor}_A(a+b)$ *and* $\mathsf{Tor}_A(-a)$*.*

Lean4 ✓ `BEDC.Derived.AbGroupUp.AbGroupTorsionSubgroup singleton closure`

Proof. The zero element is torsion: choose the positive unit multiplier $\mathsf{E1}(\mathsf{e})$. The one-step scalar action sends 0 to $0+0$, and the unit law classifies this endpoint with 0.

Now assume $\mathsf{Tor}_A(a)$ and $\mathsf{Tor}_A(b)$. Unpack Definition 42.39. Choose positive unary annihilators m and n with $\mathsf{Ann}_A(m,a)$ and $\mathsf{Ann}_A(n,b)$. By Definition 42.40, choose a positive common product p with multiplication witnesses $\mathsf{NatMul}(m,q_m,p)$ and $\mathsf{NatMul}(n,q_n,p)$.

Apply the first row of Definition 42.41 to transport the annihilator of a from m to p, and again to transport the annihilator of b from n to p. The second row of the same lemma then gives $\mathsf{Ann}_A(p,a+b)$. Since p is positive, the definition of torsion gives $\mathsf{Tor}_A(a+b)$.

For inverses, apply the third row of Definition 42.41 directly to $\mathsf{Ann}_A(m,a)$. The original multiplier m is positive, so the resulting annihilator witness gives $\mathsf{Tor}_A(-a)$. □

Theorem 42.43 (AbGroup singleton torsion unit exhaustion)**.** *On the singleton empty-history* $\mathsf{AbGroup}^{\uparrow}$ *public surface, every torsion witness* $\mathsf{Tor}_A(a)$ *forces* $a \sim_A 0$*. Moreover, every annihilator row* $\mathsf{Ann}_A(n,a)$ *factors through the same empty endpoint ledger: the scalar-action endpoint, the carried element, and the public zero row all classify to the singleton endpoint exposed by the public operation surface.*

Proof. Unpack torsion through Definition 42.39. A witness supplies a positive unary multiplier, an annihilator row, and a scalar-action endpoint classified with 0.

Project the carrier through the singleton public operation surface of Definition 42.37. On this surface the only carried endpoint is the empty-history endpoint, and all operation rows, including repeated addition and inverse, return to the same classifier class.

Use the closure theorem Definition 42.42 to keep the torsion row inside the public subgroup surface. Since the scalar-action endpoint has no non-empty carrier row available, the annihilator cannot distinguish a from the unit endpoint.

Finally compare the same surface with the singleton terminal standard bridge of Definition 42.36. The bridge reads the public $\mathsf{AbGroup}^{\uparrow}$ packet as the terminal trivial abelian group, so the annihilator ledger, scalar endpoint, and zero row all factor through the empty endpoint. Hence $a \sim_A 0$. □

Theorem 42.44 (AbGroup torsion local consumer coverage)**.** *For the unary scalar-action and torsion data of Definition 42.39, every accepted local consumer of the singleton torsion surface is exhausted by the displayed finite routes in this chapter. Namely, the consumer may read only the zero scalar-action row, the positive common-product route, the annihilator transport rows for scalar multiplication, addition, and inverse, the torsion subgroup closure row, and the singleton empty-endpoint ledger of Definition 42.43. It obtains no additional order-minimality coordinate, nontrivial cyclic-subgroup coordinate, quotient-torsion coordinate, or external scalar-action coordinate outside the displayed* $\mathsf{AbGroup}^{\uparrow}$ *packet.*

Proof. Unpack a local consumer through Definition 42.39. Its scalar-facing input is an $\mathsf{AbNatAct}_A$ witness, an annihilator row $\mathsf{Ann}_A(n, a)$, or a torsion row $\mathsf{Tor}_A(a)$. The definition supplies only two scalar-action constructors: the empty multiplier row

$$\mathsf{AbNatAct}_A(\mathsf{e}, a, 0)$$

and the one-step unary continuation row sending an endpoint x to $x + a$. Thus every scalar endpoint consumed from the packet is generated by repeated use of those two rows and classifier transport; there is no separate scalar-action table to project.

If the consumer reads an annihilator after changing the multiplier, the first row of Definition 42.41 is the only displayed transport route: an existing row $\mathsf{Ann}_A(m, a)$ is carried along a recorded multiplication witness $\mathsf{NatMul}(m, q, p)$ to $\mathsf{Ann}_A(p, a)$. If the consumer combines two torsion inputs, Definition 42.40 supplies the positive common product p, and the same transport row moves both annihilator witnesses to p. The second row of Definition 42.41 is then the displayed addition route, producing $\mathsf{Ann}_A(p, a + b)$. If the consumer reads an inverse, the third row of Definition 42.41 is the displayed route from $\mathsf{Ann}_A(p, a)$ to $\mathsf{Ann}_A(p, -a)$. These are exactly the alternatives used in Definition 42.42; the proof of that theorem introduces no further consumer coordinate.

It remains to check the singleton surface. By Definition 42.43, every torsion witness on the singleton empty-history public surface forces $a \sim_A 0$, and every annihilator row factors through the same empty endpoint ledger: the scalar-action endpoint, the carried element, and the public zero row all classify to the singleton endpoint exposed by the public operation surface. Therefore a consumer-facing singleton read has already passed through the finite scalar-action constructors, the annihilator transport rows, the torsion subgroup closure row when addition or inverse is requested, and the empty endpoint ledger.

The excluded alternatives named in the statement are not among these rows. Order-minimality would require comparing one positive annihilator against all smaller positive multipliers; no such comparison coordinate is present in Definition 42.39. A nontrivial cyclic-subgroup coordinate would require a stored orbit or generator subpacket beyond the endpoint relation $\mathsf{AbNatAct}_A$; none is present. A quotient-torsion coordinate would require a quotient carrier and quotient classifier; the singleton exhaustion row instead collapses the consumer read to $a \sim_A 0$. An external scalar-action coordinate would require a scalar operation not generated by e and $\mathsf{E1}(-)$; the definition supplies none. Hence every accepted local consumer is covered by the displayed finite routes, ledgers, and rows, and every unlisted coordinate remains outside the BEDC packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$.

Constructive story: An abelian group at this level uses the same empty-history singleton as the group marker, but now the endpoint is read through additive language. A history enters only by being history-equal to e, and two carried histories are classified by that same singleton equality, so the additive operation, zero, and negation all return to the one endpoint. The source is the singleton predicate, the pattern records continuation as addition, e as the unit, negation landing back at e, and the commutativity row in which continuing a then b and b then a both pull back to the endpoint; the ledger keeps visible and raw singleton histories aligned. Because the carrier has one inhabited class and history-equality is an equivalence, e gives nonemptiness and the classifier is reflexive, symmetric, transitive, and closed under the carrier. A history h lies in the signature gap of an $\mathsf{AbGroup}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is history-equal to e and p is the singleton abelian element. The terminal singleton bridge and torsion-subgroup row are then read off the same endpoint calculation. The chapter registers this singleton abelian construction under the name $\mathsf{AbGroup}^{\uparrow}$; commutativity is endpoint collapse on the singleton carrier, with no non-singleton abelian-group surface, quotient group, or homomorphism theory introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The singleton empty-history carrier, classifier, and abelian group laws, the Group reduct rows, the additive negation rows, the public operation surface, the singleton terminal standard bridge of Definition 42.36, and the torsion subgroup closure of Definition 42.42 form the closed paper scope.

Formal verification: FormalStatus(NameCert$_{\mathsf{AbGroup}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.AbGroupUp.singleton\_empty\_history\_abgroup\_laws`

Bridge status: bridgeChecked

Not claimed: Arbitrary abelian groups, homomorphisms, quotients, non-singleton bridges, and Lean-side bridge verification remain outside this mature paper closure.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 43

A Concrete Naming Certificate for $\mathsf{Ring}^{\uparrow}$

The ring interface $\mathsf{Ring}^{\uparrow}$ is licensed by a carrier with two binary operations $+$ and $\cdot$ such that $(C, +)$ satisfies chapter 42 and $(C, \cdot)$ satisfies chapter 40, together with left and right distributivity of $\cdot$ over $+$. It is not a primitive structure; it is licensed only by composing the additive abelian group and multiplicative monoid certificates with the distributive law.

Remark 43.1 (Concrete stability fields)**.** *This chapter records the shared additive, multiplicative, congruence, and distributivity fields for the* $\mathsf{Ring}^{\uparrow}$ *naming certificate over BEDC histories.*

43.1 Carrier definition

Definition 43.2 (Ring carrier)**.** *Use a setup with carrier C, classifier, addition, zero, additive inverse, multiplication, one, and ring law fields.*

43.2 Concrete singleton history ring instance

Definition 43.3 (Concrete singleton history ring instance)**.** *Let A_{e} and R_{e} be the singleton carrier and classifier from Definition 41.3. Define the operations by*

$$\mathsf{add}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e},$$

and

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad 1 := \mathsf{e}.$$

The resulting tuple $(A_{\mathsf{e}}, R_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1)$ is the concrete singleton history ring instance.

Theorem 43.4 (Concrete singleton history ring laws)**.** *The tuple $(A_{\mathsf{e}}, R_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1)$ satisfies the additive abelian-group laws, multiplicative monoid laws, addition and multiplication congruence, left distributivity, and right distributivity under R_{e}.*

Proof. The additive component is the singleton history abelian group of Theorem 42.3, read with its binary operation as add. Thus additive closure, zero, inverse, associativity, commutativity, and additive congruence are already supplied over A_{e} and R_{e}.

The multiplicative component is the monoid part contained in Theorem 41.5, read with $\mathsf{mul}(h,k) := \mathsf{append}(h,k)$ and $1 := \mathsf{e}$. This gives multiplication closure, left and right unit laws, associativity, and multiplication congruence under the same classifier.

For left distributivity, assume $A_{\mathsf{e}}(h)$, $A_{\mathsf{e}}(k)$, and $A_{\mathsf{e}}(l)$. Closure for add and mul puts both $\mathsf{mul}(h, \mathsf{add}(k,l))$ and $\mathsf{add}(\mathsf{mul}(h,k), \mathsf{mul}(h,l))$ back in A_{e}. The singleton witnesses identify each side with e, and transitivity of hsame composes the two endpoint witnesses into the required R_{e} relation.

For right distributivity, the same argument applies to $\mathsf{mul}(\mathsf{add}(h,k),l)$ and $\mathsf{add}(\mathsf{mul}(h,l), \mathsf{mul}(k,l))$. Closure gives the two carrier witnesses, and the singleton endpoint e supplies the composed hsame witness required by R_{e}. □

Lean4 ✓ `BEDC.Derived.RingUp.concrete singleton history ring laws`

The RingUp schema is instantiated by this concrete BHist carrier and operations, so later formalization can target BHist-anchored ring laws rather than abstract operator fields.

43.3 Concrete singleton empty-history ring instance

Definition 43.5 (Singleton empty-history ring instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define the classifier*

$$\rho_{\mathsf{e}}(h,k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Define the concrete ring operations by

$$\mathsf{add}(h,k) := \mathsf{e}, \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e}, \qquad \mathsf{mul}(h,k) := \mathsf{e}, \qquad 1 := \mathsf{e}.$$

This is the one-element BEDC history ring over the concrete carrier $A_{\mathsf{e}} : \mathsf{BHist} \to \mathsf{Prop}$ and classifier ρ_{e}.

Theorem 43.6 (Singleton empty-history ring laws)**.** *The tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1)$ satisfies carrier inhabitance, additive abelian-group laws, multiplicative monoid laws, addition and multiplication congruence, left and right distributivity, and zero multiplication laws under ρ_{e}.*

Proof. Carrier inhabitance is the reflexive endpoint witness $\mathsf{hsame}(\mathsf{e},\mathsf{e})$. By Definition 36.169 at $a_0 = \mathsf{e}$, this is exactly the witness that e lies in A_{e}.

Every operation result in the displayed tuple is definitionally e. Hence addition closure, additive inverse closure, multiplication closure, and the closedness of 0 and 1 all use the same singleton carrier witness $\mathsf{hsame}(\mathsf{e},\mathsf{e})$.

The additive unit, inverse, associativity, and commutativity laws reduce both sides to e. The required endpoint facts for ρ_{e} are the singleton carrier witnesses; the relation component is reflexivity of hsame, with symmetry and transitivity used only to package classifier endpoints already identified with e.

The multiplicative left and right unit laws, associativity, and operation congruence are handled in the same way. Both source and target products are definitionally e, so the classifier relation is $\mathsf{hsame}(\mathsf{e},\mathsf{e})$.

For left and right distributivity, both the product over a sum and the sum of products reduce to e. The zero multiplication laws also reduce both sides to e, so each law is supplied by reflexivity of hsame together with the two singleton endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonEmptyHistory laws`

Theorem 43.7 (Ring singleton negation empty endpoint)**.** *For histories* $h, t : \mathsf{BHist}$ *with* $A_{\mathsf{e}}(h)$*,*

$$\rho_{\mathsf{e}}(\mathsf{RingSingletonNeg}(h), t) \iff A_{\mathsf{e}}(t) \wedge \mathsf{hsame}(\mathsf{RingSingletonNeg}(h), \mathsf{e}).$$

Proof. The singleton ring negation endpoint is definitionally e. In the forward direction, unfold ρ_{e} and read off the right carrier witness for t together with the left endpoint witness for $\mathsf{RingSingletonNeg}(h)$.

Conversely, the displayed pair supplies the two carrier endpoints. Symmetry of hsame turns the right carrier endpoint into an endpoint comparison back from e to t, and transitivity with the left endpoint witness packages the classifier relation required by ρ_{e}. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier neg empty endpoint iff`

Theorem 43.8 (Ring singleton classifier multiplication empty endpoint)**.** *For histories* h, k, out : BHist*,*

$$\rho_{\mathsf{e}}(\mathsf{RingSingletonMul}(h, k), out) \iff \mathsf{hsame}(out, \mathsf{e}).$$

Proof. The multiplication endpoint is definitionally e. Forward projection reads the right carrier field of ρ_{e}. Conversely, the displayed endpoint witness, the reflexive left carrier at e, and symmetry rebuild the classifier. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier mul empty endpoint iff`

Theorem 43.9 (Ring singleton classifier visible-right exclusion)**.** *For histories* h, p, q : BHist*, neither* $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e0}(q)))$ *nor* $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e1}(q)))$ *can hold.*

Proof. Unfold the singleton ring classifier. In either case, the right carrier field identifies the displayed append endpoint with e. Empty-result inversion for append then forces the visible right component, either $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$, to be e. The closed constructors of BHist are disjoint, so both classifier assumptions are contradictory. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier append visible right absurd`

Theorem 43.10 (Ring singleton classifier visible-left exclusion)**.** *For histories* p, q, h : BHist*, neither* $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{e0}(q)), h)$ *nor* $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{e1}(q)), h)$ *can hold.*

Proof. Unfold the singleton ring classifier. In either case, the left carrier field identifies the displayed append endpoint with e. Empty-result inversion for append then forces the visible right component, either $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$, to be e. Constructor separation for BHist rules out both alternatives. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier append visible left absurd`

Theorem 43.11 (Ring singleton classifier append split empty)**.** *For histories* p, q, h : BHist*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(p, q), h) \iff A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q) \wedge A_{\mathsf{e}}(h).$$

This certificate has the following ten-field fit: the traditional target is singleton ring endpoint exactness; the BEDC source is the concrete singleton $\mathsf{Ring}^{\uparrow}$ *over* BHist*; the carrier is* A_{e}*; the classifier is* ρ_{e}*; stability comes from empty-result inversion and* hsame *transport; the ledger records append-factor endpoints; the core theorem is the displayed split; exactness is the displayed equivalence; the standard bridge is terminal ring endpoint equality; and the status is* $\mathsf{ProofTarget}$*.*

Proof. Unfold the singleton ring classifier. For the forward direction, the left carrier field identifies $\mathsf{append}(p, q)$ with e. Empty-result inversion for append splits this endpoint into the two factor witnesses $A_{\mathsf{e}}(p)$ and $A_{\mathsf{e}}(q)$, while the right carrier field gives $A_{\mathsf{e}}(h)$.

For the reverse direction, the two factor witnesses rebuild the empty appended endpoint. Together with the witness $A_{\mathsf{e}}(h)$, transitivity through e supplies the required classifier comparison between $\mathsf{append}(p, q)$ and h. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier append split empty`

Theorem 43.12 (Ring singleton classifier append commutation)**.** *For singleton ring carrier histories $h, k : \mathsf{BHist}$, the classifier relates the two appended endpoints with the factors exchanged:*

$$\rho_{\mathsf{e}}(\mathsf{append}(h, k), \mathsf{append}(k, h)).$$

Proof. The two carrier witnesses identify both h and k with e. Empty-result construction for append gives singleton carrier witnesses for $\mathsf{append}(h, k)$ and $\mathsf{append}(k, h)$. Since both endpoints are classified through e, transitivity and symmetry of hsame package the classifier relation. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier append comm`

Theorem 43.13 (Ring singleton classifier append associativity carrier exactness)**.** *For histories $a, b, c : \mathsf{BHist}$, the singleton ring classifier relates the two append bracketings exactly when all three factors are singleton carriers:*

$$\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(a, b), c), \mathsf{append}(a, \mathsf{append}(b, c))) \iff A_{\mathsf{e}}(a) \wedge A_{\mathsf{e}}(b) \wedge A_{\mathsf{e}}(c).$$

Proof. The forward direction projects the left carrier field from the classifier. Nested empty-result inversion for append first splits $\mathsf{append}(\mathsf{append}(a, b), c)$ into the left bracketing and c, then splits the left bracketing into a and b.

Conversely, the three carrier witnesses rebuild singleton carrier witnesses for both append bracketings. Since both endpoints are classified through e, transitivity and symmetry of hsame give the classifier relation. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier append assoc carrier iff`

Theorem 43.14 (Ring singleton continuation empty result exactness)**.** *If $C_{\mathsf{e}}(h, k)$ and $\mathsf{Cont}(h, p, r)$, then $r\mathsf{hsamee}$ holds exactly when p is in the singleton carrier.*

Proof. Transporting an empty result into the continuation and inverting empty concatenation gives $p\mathsf{hsamee}$. Conversely, the classifier makes $h\mathsf{hsamee}$; with $p\mathsf{hsamee}$, determinacy of continuation compares $\mathsf{Cont}(h, p, r)$ with the right-unit continuation at e. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier cont result empty iff`

Theorem 43.15 (Ring singleton multiplication continuation classifier)**.** *If $\mathsf{Cont}(\mathsf{RingSingletonMul}(x, y), z, r)$, then*

$$C_{\mathsf{e}}(r, h) \iff A_{\mathsf{e}}(z) \wedge A_{\mathsf{e}}(h).$$

Proof. The singleton multiplication endpoint is e. In the forward direction, transport the classified empty result into the continuation and invert the empty continuation to obtain $z\mathsf{hsamee}$; the right classifier carrier gives $h\mathsf{hsamee}$. Conversely, $z\mathsf{hsamee}$ compares the continuation result with the right-unit continuation at $\mathsf{RingSingletonMul}(x, y)$, so r is empty and classifies against h. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier mul continuation classifier iff`

Theorem 43.16 (Ring singleton continuation visible result exclusion)**.** *If $C_{\mathsf{e}}(h,k)$, $A_{\mathsf{e}}(p)$, and $\mathsf{Cont}(h,p,-)$ is formed, then no visible endpoint can be the continuation result:*

$$\mathsf{Cont}(h,p,\mathsf{E0}(r)) \Longrightarrow \bot, \qquad \mathsf{Cont}(h,p,\mathsf{E1}(r)) \Longrightarrow \bot.$$

Proof. The classifier supplies $h\mathsf{hsamee}$, while the carrier hypothesis supplies $p\mathsf{hsamee}$. Continuation respects these two history-same witnesses, so a displayed continuation result would be history-same to the right-unit result at e. Constructor disjointness excludes both $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$ from being history-same to e. □

Lean4 ✓ `BEDC.Derived.RingUp.RingSingletonClassifier continuation visible result absurd`

43.4 Singleton Empty-History Ring Instance

Definition 43.17 (Singleton empty-history RingUp instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. The classifier R_{e} is the carrier-aware singleton relation*

$$R_{\mathsf{e}}(h,k) \Longleftrightarrow A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Define the addition, multiplication, zero, one, and negation rows by

$$h +_{\mathsf{e}} k := \mathsf{e}, \qquad h \cdot_{\mathsf{e}} k := \mathsf{e}, \qquad 0_{\mathsf{e}} := \mathsf{e}, \qquad 1_{\mathsf{e}} := \mathsf{e}, \qquad -_{\mathsf{e}} h := \mathsf{e}.$$

Equivalently, every binary operation is append-trivial on the singleton endpoint: its operation tree may record two inputs, but its exposed output is the empty history. The tuple $(A_{\mathsf{e}}, R_{\mathsf{e}}, +_{\mathsf{e}}, 0_{\mathsf{e}}, -_{\mathsf{e}}, \cdot_{\mathsf{e}}, 1_{\mathsf{e}})$ is the concrete singleton empty-history $\mathsf{Ring}^{\uparrow}$ instance.

Theorem 43.18 (Singleton empty-history RingUp laws)**.** *The tuple of Definition 43.17 satisfies the $\mathsf{Ring}^{\uparrow}$ algebraic laws: additive abelian-group laws, multiplicative monoid laws, left and right distributivity, left and right zero absorption, and congruence of both operations under R_{e}.*

Proof. The carrier is inhabited by $\mathsf{hsame}(\mathsf{e},\mathsf{e})$. Every displayed operation returns e, so operation closure, the closedness of 0_{e} and 1_{e}, and the closure of additive inverses are all the singleton carrier witness for e.

For the additive laws, both sides of each unit, inverse, associativity, and commutativity comparison reduce to e. The corresponding R_{e} row is therefore assembled from the two singleton endpoint witnesses and reflexivity of hsame. This is the constant-operation specialization of the singleton AbGroup rows in Definition 42.10.

For the multiplicative monoid laws, both products in each unit and associativity comparison again reduce to e. The singleton endpoint witnesses and hsame reflexivity provide the required classifier row, matching the singleton monoid and group rows of Definition 41.5.

For operation congruence, suppose the inputs are paired by R_{e}. The source and target additions are both e, and the source and target multiplications are both e. Thus congruence is the same packed $R_{\mathsf{e}}(\mathsf{e},\mathsf{e})$ witness in both cases.

For distributivity and zero absorption, every term on either side of each law is definitionally e. The endpoint comparisons are consequently supplied by $\mathsf{hsame}(\mathsf{e},\mathsf{e})$, while the classifier endpoint fields are the singleton carrier witnesses for e. Transitivity of hsame is used only to transport any named input endpoint through its carrier witness before repacking the final R_{e} comparison. □

Theorem 43.19 (Singleton empty-history RingUp naming certificate)**.** *The instance of Definition 43.17 supplies a concrete* $\mathsf{NameCert}_{\mathsf{Ring}^\uparrow}$ *package for the one-element ring. Its ten-field fit is: traditional target the one-element ring; BEDC source* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability the singleton transport rows under* hsame*; ledger the constant-operation provenance for* $+_{\mathsf{e}}$*,* 0_{e}*,* $-_{\mathsf{e}}$*,* $\cdot_{\mathsf{e}}$*, and* 1_{e}*; core theorem Definition 43.18; exactness singleton endpoint classification; standard bridge the terminal ring; and consumer boundary the exposed one-endpoint operation table.*

Proof. The source and carrier fields are the BHist singleton predicate A_{e}, and the classifier field is R_{e}. The stability fields follow by transporting every carried endpoint through its witness to e and then using transitivity of hsame to repack the same carrier-aware classifier.

The operation-law fields are exactly Definition 43.18. They include the additive AbGroup part, the multiplicative Monoid part, both distributivity rows, zero absorption, and congruence for the two binary operations.

The ledger records only that each displayed operation tree has endpoint e. Since R_{e} exposes precisely the two endpoint carrier witnesses and their hsame comparison, exactness says that a public endpoint belongs to the certificate exactly when it is the singleton empty-history endpoint. This is the terminal one-element ring table read through BEDC histories. □

43.5 Empty-History Ground Schema Unlock

Definition 43.20 (RingUp empty-history ground instance)**.** *The ground* $\mathsf{Ring}^\uparrow$ *tuple over* BHist *is the singleton empty-history instance*

$$(A_{\mathsf{e}}, R_{\mathsf{e}}, +_{\mathsf{e}}, 0_{\mathsf{e}}, -_{\mathsf{e}}, \cdot_{\mathsf{e}}, 1_{\mathsf{e}})$$

of Definition 43.17. Its carrier is the singleton row A_{e} *at the empty history, its classifier is* R_{e}*, and its displayed operations are fixed by*

$$h +_{\mathsf{e}} k := \mathsf{e}, \qquad -_{\mathsf{e}} h := \mathsf{e}, \qquad 0_{\mathsf{e}} := \mathsf{e},$$

together with

$$h \cdot_{\mathsf{e}} k := \mathsf{e}, \qquad 1_{\mathsf{e}} := \mathsf{e}.$$

Thus every operator row is a concrete BHist *row with endpoint* e*, and every classifier read is the carrier-aware singleton comparison.*

Theorem 43.21 (RingUp empty-history ground laws)**.** *The tuple of Definition 43.20 satisfies the* $\mathsf{Ring}^\uparrow$ *laws: additive abelian-group laws, multiplicative monoid laws, left and right distributivity, left and right zero absorption, and congruence of addition and multiplication under* R_{e}*.*

Proof. The tuple is exactly the singleton empty-history instance of Definition 43.17. Its operation rows, distinguished zero and one rows, negation row, carrier, and classifier are the same displayed rows.

Apply Definition 43.18. That theorem supplies the additive, multiplicative, distributive, zero-absorption, and congruence rows for the same tuple, so no abstract carrier or operator parameter remains to instantiate. □

Theorem 43.22 (RingUp empty-history schema unlock)**.** *Every* $\mathsf{Ring}^\uparrow$ *schema consumer that only requires a concrete ground ring may use the displayed tuple of Definition 43.20. The consumer receives the carrier* A_{e}*, classifier* R_{e}*, addition, negation, zero, multiplication, one, and the law package of Definition 43.21 as concrete* BHist *rows, rather than abstract carrier and operator parameters.*

Proof. Project the ground instance. The carrier, classifier, and five operation rows are all displayed singleton empty-history rows, so a schema consumer can cite them directly.

The law package is supplied by Definition 43.21. Since that theorem is already about the same displayed tuple, substituting it into a ground-ring schema does not introduce a new carrier, a second classifier, or additional operator parameters. □

43.6 The certificate

Definition 43.23 (Ring source specification)**.** *The source is an additive* $\mathsf{AbGroup}^{\uparrow}$ *and multiplicative* $\mathsf{Monoid}^{\uparrow}$ *on the same carrier and classifier.*

Definition 43.24 (Ring pattern specification)**.** *The pattern is expression formation from* $0, 1, +, -, \cdot$.

Definition 43.25 (Ring classifier specification)**.** *The classifier is the shared carrier sameness relation, congruent for both operations.*

Definition 43.26 (Ring stability certificate)**.** *The stability certificate consists of:*

1. *additive abelian-group laws*
2. *multiplicative monoid laws*
3. *addition and multiplication congruence*
4. *left distributivity*
5. *right distributivity*
6. *zero multiplication laws derived or supplied constructively*

Lean4 ✓ `BEDC.Derived.RingUp.ring stability certificate fields`

Theorem 43.27 (Ring additive right inverse)**.** *For BEDC history addition, additive commutativity and the supplied left inverse law imply the matching right inverse law* $x + (-x)\mathsf{hsame}0$.

Proof. Commute $x + (-x)$ to $(-x) + x$, then apply the left inverse law. □

Lean4 ✓ `BEDC.Derived.RingUp.ring add right inverse`

Theorem 43.28 (Ring additive inverse of zero)**.** *For BEDC history addition, additive commutativity, the left-zero law, and the left inverse law imply* $\mathsf{hsame}(-0, 0)$.

Proof. Chain -0 backward through the left-zero equation $0 + (-0)\mathsf{hsame} - 0$, commute $0 + (-0)$ to $(-0) + 0$, and apply the left inverse law at 0. □

Lean4 ✓ `BEDC.Derived.RingUp.ring neg zero`

Theorem 43.29 (Ring duplicate addition collapses to zero)**.** *For BEDC history addition, if an element is the same as its duplicate sum* $a + a$*, then the element is the same as zero.*

Proof. Add the left inverse of a on the left, reassociate to bring $(-a) + a$ together, collapse that pair to zero, and use the left-zero law. □

Lean4 ✓ `BEDC.Derived.RingUp.ring add duplicate eq zero`

Theorem 43.30 (Ring additive duplicate zero exactness)**.** *For BEDC history addition satisfying additive associativity, left zero, left inverse, and addition congruence, every history a satisfies*

$$a\mathsf{hsame}a + a \iff a\mathsf{hsame}0.$$

Proof. The forward implication is exactly the duplicate-addition collapse: apply Definition 43.29 to the hypothesis $a\mathsf{hsame}a + a$.

For the reverse implication, use symmetry of $a\mathsf{hsame}0$ to transport the left summand from a to 0, and use the hypothesis itself to transport the right summand from a to 0 by addition congruence. The resulting comparison $a + a\mathsf{hsame}0 + 0$ reduces by the left-zero law to 0, and transitivity with the original hypothesis gives $a\mathsf{hsame}a + a$. □

Lean4 ✓ `BEDC.Derived.RingUp.ring add duplicate empty iff`

Theorem 43.31 (Ring additive right zero)**.** *For BEDC history addition, additive commutativity and the supplied left-zero law imply the matching right-zero law $x + 0\mathsf{hsame}x$.*

Proof. Commute $x + 0$ to $0 + x$, then apply the left-zero law. □

Lean4 ✓ `BEDC.Derived.RingUp.ring add right zero`

Theorem 43.32 (Ring additive two-sided cancellation)**.** *For BEDC history addition satisfying additive associativity, commutativity, left-zero, left-inverse, and addition congruence, every $a, b, c :$ BHist satisfies both cancellation implications:*

$$a + c\mathsf{hsame}b + c \implies a\mathsf{hsame}b$$

and

$$c + a\mathsf{hsame}c + b \implies a\mathsf{hsame}b.$$

The ten-field fit is: traditional target additive cancellation in a ring; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist *plus* $\mathsf{Group}^{\uparrow}$ *cancellation; carrier* BHist*; classifier* hsame*; stability from additive group laws and addition congruence; ledger the common addend; core theorem this theorem; exactness predicate the two displayed implications; standard bridge ordinary ring additive cancellation; status* ProofTarget*.*

Proof. First derive the additive right-zero law from Theorem 43.31 and the additive right-inverse law from Theorem 43.27. Together with associativity, commutativity, left zero, left inverse, and addition congruence, these are the group-cancellation hypotheses for the additive operation.

For the first implication, read addition as the group operation and apply Theorem 41.116 to the common right summand c. The hypothesis $a + c\mathsf{hsame}b + c$ then yields $a\mathsf{hsame}b$.

For the second implication, apply Theorem 41.66 to the common left summand c. The hypothesis $c + a\mathsf{hsame}c + b$ then yields the same conclusion $a\mathsf{hsame}b$. □

Lean4 ✓ `BEDC.Derived.RingUp.ring additive two sided cancellation`

Theorem 43.33 (Ring negative of sum)**.** *For BEDC history addition and negation satisfying additive associativity, commutativity, left zero, left inverse, and addition congruence, every $a, b :$ BHist satisfies*

$$\mathsf{hsame}(\mathsf{neg}(\mathsf{add}(a, b)), \mathsf{add}(\mathsf{neg}(a), \mathsf{neg}(b))).$$

The ten-field fit is: traditional target additive inverse of a sum in a ring; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from additive congruence and additive cancellation; ledger the two addends a, b and the inverse witnesses* $\mathsf{neg}(\mathsf{add}(a, b))$ *and* $\mathsf{add}(\mathsf{neg}(a), \mathsf{neg}(b))$*; core theorem this theorem; exactness the displayed equality; downstream consumer CommRing signed-product normalization and FieldUp denominator inverse normalization; status* ProofTarget*.*

Proof. The common element to cancel is $\mathsf{add}(a, b)$. By Theorem 43.27, the endpoint $\mathsf{neg}(\mathsf{add}(a, b))$ is a right inverse of this common element.

It remains to see that $\mathsf{add}(\mathsf{neg}(a), \mathsf{neg}(b))$ is also a right inverse of $\mathsf{add}(a, b)$. Reassociate the sum $(a + b) + ((-a) + (-b))$, commute the middle terms so that $b + (-b)$ and $a + (-a)$ appear as adjacent pairs, collapse those pairs using the right inverse and left inverse rows, and remove the resulting zeros by the left and right zero laws.

Both endpoints therefore give the same sum with the common left factor $\mathsf{add}(a, b)$. Apply Theorem 43.32 to cancel that factor and obtain the displayed hsame comparison. □

Lean4 ✓ `BEDC.Derived.RingUp.ring neg add`

Theorem 43.34 (Ring additive difference zero exactness)**.** *For BEDC history addition satisfying additive associativity, commutativity, left zero, left inverse, and addition congruence, every x, y : BHist satisfies*

$$x + (-y)\mathsf{hsame}0 \iff x\mathsf{hsame}y.$$

The ten-field fit is: traditional target additive difference exactness; BEDC source $\mathsf{Ring}^{\uparrow}$ over BHist plus the $\mathsf{Group}^{\uparrow}$ cancellation spine already cited by Definition 43.32; carrier BHist; classifier hsame; stability from additive group laws and addition congruence; ledger the hidden signed summand; core theorem this theorem; exactness the displayed iff; standard bridge ordinary ring subtraction exactness; status ProofTarget.

Lean4 ✓ `BEDC.Derived.RingUp.ring additive difference zero exact`

Definition 43.35 (Opposite ring certificate data)**.** *Let C carry a $\mathsf{Ring}^{\uparrow}(C)$ certificate with classifier $\sim_C$, addition $+_C$, zero 0_C, additive inverse operation $-_C$, multiplication $\cdot_C$, and multiplicative identity 1_C. Define the reversed multiplication by*

$$a \cdot_C^{\mathrm{op}} b := b \cdot_C a.$$

The opposite ring data C^{op} have the same carrier C, the same classifier $\sim_C$, the same addition $+_C$, the same zero 0_C, the same additive inverse operation $-_C$, the same multiplicative identity 1_C, and multiplication $\cdot_C^{\mathrm{op}}$.

Theorem 43.36 (Opposite ring certificate)**.** *If the displayed data on C satisfy the $\mathsf{Ring}^{\uparrow}(C)$ certificate obligations, then the data of Definition 43.35 also satisfy the $\mathsf{Ring}^{\uparrow}(C)$ certificate obligations. Thus the same carrier, additive structure, multiplicative identity, and classifier carry a ring certificate with multiplication $\cdot_C^{\mathrm{op}}$.*

Proof. Unpack the $\mathsf{Ring}^{\uparrow}(C)$ obligations through Definition 43.55. The carrier row is unchanged by Definition 43.2. The carrier-and-classifier core is the same NameCert data, so classifier reflexivity, symmetry, transitivity, and carrier transport are copied from Definition 9.7.

The additive source required by Definition 43.23 is unchanged: the $\mathsf{AbGroup}^{\uparrow}$ carrier uses the same $+_C$, 0_C, $-_C$, and $\sim_C$, with the same additive stability and ledger rows. It remains to read the multiplicative source as a $\mathsf{Monoid}^{\uparrow}$ certificate for $\cdot_C^{\mathrm{op}}$. Closure follows from the original multiplication closure applied to the ordered pair (b, a). The opposite left identity row is

$$1_C \cdot_C^{\mathrm{op}} a = a \cdot_C 1_C \sim_C a,$$

which is the original right identity row. The opposite right identity row is

$$a \cdot_C^{\mathrm{op}} 1_C = 1_C \cdot_C a \sim_C a,$$

which is the original left identity row. For associativity, the two opposite bracketings unfold to

$$(a \cdot_C^{\mathrm{op}} b) \cdot_C^{\mathrm{op}} c = c \cdot_C (b \cdot_C a), \qquad a \cdot_C^{\mathrm{op}} (b \cdot_C^{\mathrm{op}} c) = (c \cdot_C b) \cdot_C a.$$

The associativity-under-classifier row in Definition 40.13, applied to the original multiplication at the triple (c, b, a), supplies

$$(c \cdot_C b) \cdot_C a \sim_C c \cdot_C (b \cdot_C a).$$

Classifier symmetry gives the orientation $c \cdot_C (b \cdot_C a) \sim_C (c \cdot_C b) \cdot_C a$ needed to compare the two unfolded opposite bracketings. Multiplication congruence for $\cdot_C^{\mathrm{op}}$ is the original multiplication congruence with the two input comparisons supplied in exchanged order. These are precisely the monoid stability rows of Definition 40.13, hence the multiplicative source satisfies Definition 40.34.

The pattern row of Definition 43.24 is unchanged except that the multiplication node is interpreted by $\cdot_C^{\mathrm{op}}$. The classifier row of Definition 43.25 is unchanged for addition, and the preceding paragraph supplies congruence for the reversed multiplication.

For left distributivity in the opposite data, unfold the reversed product and use original right distributivity:

$$a \cdot_C^{\mathrm{op}} (b +_C c) = (b +_C c) \cdot_C a \sim_C (b \cdot_C a) +_C (c \cdot_C a) = (a \cdot_C^{\mathrm{op}} b) +_C (a \cdot_C^{\mathrm{op}} c).$$

For right distributivity in the opposite data, unfold the reversed product and use original left distributivity:

$$(a +_C b) \cdot_C^{\mathrm{op}} c = c \cdot_C (a +_C b) \sim_C (c \cdot_C a) +_C (c \cdot_C b) = (a \cdot_C^{\mathrm{op}} c) +_C (b \cdot_C^{\mathrm{op}} c).$$

The two zero multiplication rows are exchanged in the same way: $0_C \cdot_C^{\mathrm{op}} a = a \cdot_C 0_C$ is the original right-zero row, and $a \cdot_C^{\mathrm{op}} 0_C = 0_C \cdot_C a$ is the original left-zero row. Together with the copied additive laws and congruence rows, these are exactly the stability fields listed in Definition 43.26.

Finally, the ledger policy of Definition 43.54 uses the same carrier witnesses and the same additive operation trees. A multiplication node with opposite inputs (a, b) records the original multiplication ledger for (b, a). Therefore the carrier, source, pattern, classifier, stability, and ledger rows all satisfy the $\mathsf{Ring}^{\uparrow}(C)$ certificate obligations for the opposite ring data. □

Theorem 43.37 (Ring negation classifier exactness)**.** *For BEDC history addition and negation satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, and additive inverse congruence, every $x, y : \mathsf{BHist}$ satisfies*

$$-x \mathsf{hsame} - y \iff x \mathsf{hsame} y.$$

The ten-field fit is: traditional target additive inverse injectivity in a ring; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist *with the additive* $\mathsf{Group}^{\uparrow}$ *cancellation spine cited by Definition 43.34; carrier* BHist*; classifier* hsame*; stability from addition congruence and additive inverse congruence; ledger the two negated endpoints; core theorem this theorem; exactness the displayed iff; standard bridge ordinary ring additive inverse injectivity; status* $\mathsf{ProofTarget}$*.*

Lean4 ✓ `BEDC.Derived.RingUp.ring negation classifier exactness`

Proof. Assume first that $-x \mathsf{hsame} - y$. By classifier symmetry, $-y \mathsf{hsame} - x$. Addition congruence, with reflexivity on the left summand x, transports this to

$$x + (-y) \mathsf{hsame} x + (-x).$$

The right-inverse law of Definition 43.27 gives $x+(-x)\mathsf{hsame}0$, so transitivity gives $x+(-y)\mathsf{hsame}0$. The forward direction of Definition 43.34 then yields $x\mathsf{hsame}y$.

Conversely, assume $x\mathsf{hsame}y$. The additive inverse congruence field transports this classifier witness directly across negation, giving $-x\mathsf{hsame}-y$. □

Theorem 43.38 (Double-opposite ring certificate data identity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate satisfying the obligations of Definition 43.55. Present its data by a carrier predicate $\mathcal{C}$, classifier $\sim_R$, addition $+_R$, zero 0_R, additive inverse operation $-_R$, multiplication $\cdot_R$, multiplicative identity 1_R, stability table $\mathcal{S}_R$ as in Definition 43.26, and ledger table $\mathcal{L}_R$ as in Definition 43.54. Let $R^{\mathrm{opop}} = (R^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 43.35 twice. Then R^{opop} is a $\mathsf{Ring}^{\uparrow}(R)$ certificate, and its carrier, classifier, additive data, multiplicative identity, multiplication, stability rows, and ledger rows are componentwise identical to those of R: for carried endpoints a, b,*

$$\begin{aligned}
\mathcal{C}^{\mathrm{opop}}(a) &= \mathcal{C}(a), & a \sim_R^{\mathrm{opop}} b &= a \sim_R b,\\
a +_R^{\mathrm{opop}} b &= a +_R b, & 0_R^{\mathrm{opop}} &= 0_R,\\
(-_R)^{\mathrm{opop}}(a) &= -_R a, & 1_R^{\mathrm{opop}} &= 1_R,\\
a \cdot_R^{\mathrm{opop}} b &= a \cdot_R b, & \mathcal{S}_R^{\mathrm{opop}} &= \mathcal{S}_R,\\
\mathcal{L}_R^{\mathrm{opop}} &= \mathcal{L}_R.
\end{aligned}$$

The equalities involving $\mathcal{S}_R$ and $\mathcal{L}_R$ mean equality of the named rows and their displayed endpoint data.

Proof. Because R satisfies the $\mathsf{Ring}^{\uparrow}(R)$ obligations, Definition 43.36 makes R^{op} a certified $\mathsf{Ring}^{\uparrow}(R)$ instance. Applying the same theorem to that certified opposite data makes R^{opop} a certified $\mathsf{Ring}^{\uparrow}(R)$ instance.

It remains to compare the displayed fields. By Definition 43.35, one opposite keeps the carrier, classifier, addition, zero endpoint, additive inverse operation, and multiplicative identity fixed. Two unfoldings therefore give

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}(a), \qquad a \sim_R^{\mathrm{opop}} b = a \sim_R b,$$

$$a +_R^{\mathrm{opop}} b = a +_R b, \qquad 0_R^{\mathrm{opop}} = 0_R, \qquad (-_R)^{\mathrm{opop}}(a) = -_R a, \qquad 1_R^{\mathrm{opop}} = 1_R.$$

For multiplication, the first opposite reverses the two inputs and the second opposite reverses them back:

$$a \cdot_R^{\mathrm{opop}} b = b \cdot_R^{\mathrm{op}} a = a \cdot_R b.$$

Thus the raw ring data of the double opposite certificate are the raw ring data of R.

The source row of Definition 43.23 is restored from these identities: the additive $\mathsf{AbGroup}^{\uparrow}$ source is fixed in each opposite, and the multiplicative $\mathsf{Monoid}^{\uparrow}$ source is typed over the restored multiplication and the same identity. The pattern row of Definition 43.24 forms expressions from the same 0_R, 1_R, $+_R$, $-_R$, and restored multiplication. The classifier row of Definition 43.25 is the same relation $\sim_R$, with addition congruence fixed and multiplication congruence read over the restored multiplication.

For the stability table, Definition 43.26 lists the additive abelian-group laws, multiplicative monoid laws, operation congruence, left distributivity, right distributivity, and zero multiplication laws. The additive rows and addition congruence are fixed by each opposite construction. In the multiplicative part, one opposite exchanges the two unit rows, reads associativity with the triple reversed, and exchanges the two congruence inputs; the second opposite performs the same exchanges again, restoring the unit, associativity, and multiplication-congruence rows for $\cdot_R$. The left

and right distributivity rows are exchanged once and then restored, and the two zero multiplication rows are exchanged in the same way. Hence $\mathcal{S}_R^{\mathrm{opop}} = \mathcal{S}_R$ row by row.

Finally, Definition 43.54 records the shared carrier ledger and the operation tree for ring expressions. Carrier, additive, zero, inverse, and unit entries are fixed by each opposite. A multiplication node records the same two operands with their child order exchanged in one opposite reading; the second opposite exchanges that child order again. Therefore every ledger entry of R^{opop} is the corresponding ledger entry of R, so $\mathcal{L}_R^{\mathrm{opop}} = \mathcal{L}_R$. Together with the certified status above, these identities identify the double-opposite ring certificate data with the original ring certificate data. □

Proof. First derive the additive right-zero law from Theorem 43.31 and the additive right-inverse law from Theorem 43.27. These give the removable zero and inverse pairs used on both sides of the equivalence.

For the forward implication, add y to the right of the comparison $x+(-y)\mathsf{hsame}0$. Associativity rewrites $(x+(-y))+y$ as $x+((-y)+y)$, commutativity and the right-inverse law collapse $(-y)+y$ to 0, and the right-zero law removes that terminal zero. The right endpoint $0+y$ reduces to y by the left-zero law, leaving $x\mathsf{hsame}y$.

For the reverse implication, transport $x\mathsf{hsame}y$ through addition congruence against the common summand $-y$. The right side becomes $y+(-y)$, which Theorem 43.27 collapses to 0, yielding $x+(-y)\mathsf{hsame}0$ by transitivity. □

Theorem 43.39 (Ring multiplication zero absorption)**.** *For BEDC history operations satisfying additive associativity, left zero, left inverse, congruence, and both distributive laws, multiplication by zero on either side is internally the same as zero.*

Lean4 ✓ `BEDC.Derived.RingUp.ring mul zero absorption`

Proof. First cancel a duplicated summand $a\mathsf{hsame}a+a$ by adding the left inverse, reassociating, and using the left-zero law. Then apply distributivity to $x\cdot(0+0)$ and $(0+0)\cdot x$, transport along $0+0\mathsf{hsame}0$, and use the duplicate-cancellation step on the two resulting products. □

Theorem 43.40 (Ring zero-classifier factor absorption)**.** *For BEDC history operations satisfying additive associativity, left zero, left inverse, addition congruence, multiplication congruence, and both distributive laws, if* $\mathsf{hsame}(x,0)$*, then every* $y : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}(x\cdot y,0)\wedge\mathsf{hsame}(y\cdot x,0).$$

Proof. Invoke Definition 43.39 to obtain the two literal zero-factor absorptions. For the left product, multiplication congruence transports the left factor along $\mathsf{hsame}(x,0)$, comparing $x\cdot y$ with $0\cdot y$. Transitivity with the left zero-factor absorption gives $\mathsf{hsame}(x\cdot y,0)$, using symmetry if the congruence step is read in the opposite direction.

For the right product, multiplication congruence transports the right factor along the same classifier witness, comparing $y\cdot x$ with $y\cdot 0$. Transitivity with the right zero-factor absorption gives $\mathsf{hsame}(y\cdot x,0)$, again using symmetry when the displayed comparison is oriented oppositely. The two transported conclusions form the required conjunction. □

Lean4 ✓ `BEDC.Derived.RingUp.ring zero classifier factor absorption`

Theorem 43.41 (Ring zero-classifier two-sided ideal package)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, additive inverse congruence, and both distributive laws, if* $\mathsf{hsame}(x, 0)$ *and* $\mathsf{hsame}(y, 0)$*, then*

$$\mathsf{hsame}(x + y, 0) \wedge \mathsf{hsame}(-x, 0) \wedge \forall z : \mathsf{BHist},\ \mathsf{hsame}(x \cdot z, 0) \wedge \mathsf{hsame}(z \cdot x, 0) \wedge \mathsf{hsame}(x \cdot y, 0).$$

The ten-field fit is: traditional target zero-classifier two-sided ideal closure package; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from addition congruence, additive inverse congruence, multiplication congruence, and both distributive laws; ledger the two zero-classifier witnesses and an arbitrary multiplier; core theorem this theorem; exactness the displayed addition, negation, and two-sided multiplication closure; downstream consumer* $\mathsf{Field}^{\uparrow}$ *zero-product exits,* $\mathsf{QuotientRing}^{\uparrow}$ *kernels, and* $\mathsf{Ideal}^{\uparrow}$ *certificates; status* $\mathsf{ProofTarget}$*.*

Proof. First close the zero classifier under addition. Addition congruence transports $x + y$ along $\mathsf{hsame}(x, 0)$ and $\mathsf{hsame}(y, 0)$ to compare it with $0 + 0$, and the left-zero law reduces $0 + 0$ to 0. Transitivity gives $\mathsf{hsame}(x + y, 0)$.

Next close the classifier under additive inverse. Additive inverse congruence transports $\mathsf{hsame}(x, 0)$ to a comparison between $-x$ and -0, while Definition 43.28 identifies -0 with 0. Transitivity gives $\mathsf{hsame}(-x, 0)$.

Finally fix $z : \mathsf{BHist}$. Apply Definition 43.40 to the factor x and multiplier z to obtain $\mathsf{hsame}(x \cdot z, 0)$ and $\mathsf{hsame}(z \cdot x, 0)$. Apply the same theorem with factor x and multiplier y to obtain $\mathsf{hsame}(x \cdot y, 0)$, and package these conclusions with the additive and negative closure witnesses. □

Lean4 ✓ `BEDC.Derived.RingUp.ring zero classifier two sided ideal package`

Theorem 43.42 (Ring zero-classifier square annihilator package)**.** *For BEDC history ring operations satisfying additive associativity, left zero, left inverse, addition congruence, multiplication congruence, and both distributive laws, if* $\mathsf{hsame}(x, 0)$*, then*

$$\mathsf{hsame}(x \cdot x, 0) \wedge \forall c : \mathsf{BHist},\ \mathsf{hsame}((x \cdot x) \cdot c, 0) \wedge \mathsf{hsame}(c \cdot (x \cdot x), 0).$$

The ten-field fit is: traditional target square-zero annihilator package; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from multiplication congruence and zero-classifier factor absorption; ledger the original zero-factor witness, the square-zero witness, and the universal multiplier row; core theorem this theorem; exactness the displayed package; downstream consumer* $\mathsf{Field}^{\uparrow}$ *square-apartness and zero-product exit packages; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 43.40 with factor x and multiplier x. Its left component gives the square-zero classifier witness $\mathsf{hsame}(x \cdot x, 0)$.

Now fix an arbitrary multiplier $c : \mathsf{BHist}$. Apply Definition 43.40 again, this time with factor $x \cdot x$ and multiplier c, using the square-zero witness from the first step. The two components give $\mathsf{hsame}((x \cdot x) \cdot c, 0)$ and $\mathsf{hsame}(c \cdot (x \cdot x), 0)$, and packaging them with the square-zero witness is exactly the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.RingUp.ring zero classifier square annihilator package`

Theorem 43.43 (Ring equal-factor difference annihilation)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, and both distributive laws, if* $x\mathsf{hsame}y$*, then every* $z : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}((x + (-y)) \cdot z, 0) \wedge \mathsf{hsame}(z \cdot (x + (-y)), 0).$$

The ten-field fit is: traditional target ordinary ring difference annihilation; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from additive group laws, addition congruence, multiplication congruence, and both distributive laws; ledger the equal-factor witness and the multiplier; core theorem this theorem; exactness additive difference zero exactness feeding zero-factor absorption; standard bridge ordinary ring equal-factor difference annihilation; downstream use FieldUp and RatUp zero-product reasoning; status* $\mathsf{ProofTarget}$*.*

Lean4 ✓ `BEDC.Derived.RingUp.ring equal factor difference annihilation`

Proof. Apply the reverse direction of Definition 43.34 to the classifier witness $x\mathsf{hsame}y$. This turns the displayed difference into a zero-classified factor, namely $x + (-y)\mathsf{hsame}0$.

Now apply Definition 43.40 with factor $x + (-y)$ and multiplier z. The left component gives $\mathsf{hsame}((x+(-y))\cdot z, 0)$, and the right component gives $\mathsf{hsame}(z\cdot(x+(-y)), 0)$; these two components are exactly the displayed conjunction. □

Theorem 43.44 (Ring zero-classifier signed factor absorption)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, additive inverse congruence, and both distributive laws, if* $\mathsf{hsame}(x, 0)$*, then every* $y : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}((-x)y, 0) \wedge \mathsf{hsame}(y(-x), 0) \wedge \mathsf{hsame}((-x)(-y), 0) \wedge \mathsf{hsame}((-y)(-x), 0).$$

The ten-field fit is: traditional target signed zero-factor absorption in a ring; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations additive inverse congruence, zero classifier absorption, and both distributive laws; ledger the zero-classifier witness and signed multiplier; core theorem this theorem.*

Proof. First transport the zero-classifier witness through additive inverse. Additive inverse congruence sends $\mathsf{hsame}(x, 0)$ to a comparison between $-x$ and -0, and Definition 43.28 identifies -0 with 0. Transitivity gives $\mathsf{hsame}(-x, 0)$.

Apply Definition 43.40 with factor $-x$ and multiplier y. This yields the two absorptions $\mathsf{hsame}((-x)y, 0)$ and $\mathsf{hsame}(y(-x), 0)$.

Apply the same absorption theorem again with factor $-x$ and multiplier $-y$. Its left and right conclusions are precisely $\mathsf{hsame}((-x)(-y), 0)$ and $\mathsf{hsame}((-y)(-x), 0)$, and the four conclusions package into the stated conjunction. □

Lean4 ✓ `BEDC.Derived.RingUp.ring zero classifier signed factor absorption`

Theorem 43.45 (Ring left-negative product sum vanishes)**.** *For BEDC history operations satisfying additive associativity, left zero, left inverse, operation congruence, and both distributive laws, every pair* x, y *satisfies* $(-x)y + xy\mathsf{hsame}0$*.*

Proof. Apply right distributivity backward to identify $(-x)y+xy$ with $((-x)+x)y$. Transport the left factor across the additive left-inverse law $(-x) + x\mathsf{hsame}0$, then use the left zero-absorption half of multiplication by zero. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg left inverse sum zero`

Theorem 43.46 (Ring right-negative product sum vanishes)**.** *For BEDC history operations satisfying additive associativity, left zero, left inverse, operation congruence, and both distributive laws, every pair* x, y *satisfies* $x(-y) + xy\mathsf{hsame}0$*.*

Proof. Apply left distributivity backward to identify $x(-y) + xy$ with $x((-y) + y)$. Transport the right factor across the additive left-inverse law $(-y) + y\mathsf{hsame}0$, then use the right zero-absorption half of multiplication by zero. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg right inverse sum zero`

Theorem 43.47 (Ring left-negative and right-negative products agree)**.** *For BEDC history operations satisfying additive abelian-group laws, operation congruence, and both distributive laws, the two negative products agree:* $(-x)y\mathsf{hsame}x(-y)$.

Proof. The left-negative and right-negative product sums both collapse to zero after adding the common summand xy on the right. The additive right-zero and right-inverse laws supply the hypotheses for right cancellation in the underlying additive group, so the common summand cancels. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg left right same`

Theorem 43.48 (Ring left-negative product equals negative product)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, operation congruence, and both distributive laws, every pair* x, y *satisfies*

$$\mathsf{hsame}((-x)y, -(xy)).$$

Proof. The left-negative product sum theorem gives $(-x)y + xy\mathsf{hsame}0$. The additive right inverse theorem gives $xy + (-(xy))\mathsf{hsame}0$.

Apply Definition 41.59 with the group operation read as addition, the identity read as 0, and the inverse operation read as negation. The displayed equations are the left-inverse and right-inverse witnesses for xy; Definition 43.31 and Definition 43.27 provide the right identity and right inverse rows required by that GroupUp theorem. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg left eq neg mul`

Theorem 43.49 (Ring right-negative product equals negative product)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, operation congruence, and both distributive laws, every pair* x, y *satisfies*

$$\mathsf{hsame}(x(-y), -(xy)).$$

Proof. Apply left distributivity backward to identify $x(-y) + xy$ with $x((-y) + y)$. Transport the right factor across the additive left-inverse law, use right zero absorption for multiplication, and obtain $x(-y) + xy\mathsf{hsame}0$.

Apply Definition 41.59 in the additive group, with identity 0 and inverse operation given by negation. The new equality is the left-inverse witness for xy, while Definition 43.27 gives the right-inverse witness $xy + (-(xy))\mathsf{hsame}0$; the right identity row comes from Definition 43.31. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg right eq neg mul`

Theorem 43.50 (Ring zero-classifier negated product annihilation)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, additive inverse congruence, and both distributive laws, if* $\mathsf{hsame}(x, 0)$*, then every* $y : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}(-(xy), 0) \wedge \mathsf{hsame}(-(yx), 0).$$

The ten-field fit is: traditional target negated zero-product annihilation; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from signed zero-factor absorption and negative-product identification; ledger the zero-factor witness and the hidden product endpoint; core theorem this theorem; exactness the displayed conjunction; downstream consumer* $\mathsf{Field}^{\uparrow}$ *denominator inverse normalization and zero-product exits; status* $\mathsf{ProofTarget}$*.*

Lean4 ✓ `BEDC.Derived.RingUp.ring zero classifier negated product annihilation`

Proof. Apply Definition 43.44 to the zero-classified factor x and multiplier y. Its first two components give the signed zero-product rows $\mathsf{hsame}((-x)y, 0)$ and $\mathsf{hsame}(y(-x), 0)$.

Apply Definition 43.48 to identify $(-x)y$ with $-(xy)$. Apply Definition 43.49 with the ordered pair (y, x) to identify $y(-x)$ with $-(yx)$.

Use symmetry for the two endpoint identifications where necessary, and then transitivity of hsame with the two signed zero-product rows. This gives $\mathsf{hsame}(-(xy), 0)$ and $\mathsf{hsame}(-(yx), 0)$, which are exactly the displayed conjunction. □

Theorem 43.51 (Ring double-negative product)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, operation congruence, and both distributive laws, every pair* x, y *satisfies*

$$\mathsf{hsame}((-x)(-y), xy).$$

Proof. The left-negative product theorem identifies $(-x)(-y)$ with $-(x(-y))$. The right-negative product theorem identifies $x(-y)$ with $-(xy)$.

Apply Definition 41.60 under the additive-group reading, so the inverse operation is negation and the right identity and right inverse rows are supplied by Definition 43.31 and Definition 43.27. This transports the second equality through the outer inverse. Then Definition 41.57 collapses $-(-(xy))$ to xy under the same additive reading. □

Lean4 ✓ `BEDC.Derived.RingUp.ring mul neg neg eq mul`

Theorem 43.52 (Ring equal-factor signed difference annihilation)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, additive inverse congruence, and both distributive laws, if* $x\mathsf{hsame}y$*, then every* $z : \mathsf{BHist}$ *satisfies*

$$\mathsf{hsame}((x+(-y))z, 0) \wedge \mathsf{hsame}(z(x+(-y)), 0) \wedge \mathsf{hsame}((-(x+(-y)))z, 0) \wedge \mathsf{hsame}(z(-(x+(-y))), 0).$$

The ten-field fit is: traditional target signed annihilation of an equal-factor difference; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from additive difference exactness, zero-factor absorption, and signed zero-factor absorption; ledger the equality witness, the hidden difference, the sign, and the multiplier; core theorem this theorem; exactness the four displayed zero-classifier rows; downstream consumer* $\mathsf{Field}^{\uparrow}$ *denominator inverse and zero-product packages; status* $\mathsf{ProofTarget}$*.*

Proof. Set $d := x + (-y)$. Apply the reverse direction of Definition 43.34 to the witness $x\mathsf{hsame}y$; this gives $d\mathsf{hsame}0$.

Apply Definition 43.40 to d and multiplier z. This gives the two unsigned products $\mathsf{hsame}(dz, 0)$ and $\mathsf{hsame}(zd, 0)$.

Apply Definition 43.44 to the same zero-classified factor and multiplier. Its first two conjuncts give $\mathsf{hsame}((-d)z, 0)$ and $\mathsf{hsame}(z(-d), 0)$. Substituting $d = x + (-y)$ packages the four displayed zero-classifier rows. □

Lean4 ✓ `BEDC.Derived.RingUp.ring equal factor signed difference annihilation`

Theorem 43.53 (Ring signed square zero ideal package)**.** *For BEDC history ring operations satisfying additive associativity, commutativity, left zero, left inverse, addition congruence, multiplication congruence, additive inverse congruence, and both distributive laws, if* $\mathsf{hsame}(x, 0)$*, then*

$$\mathsf{hsame}((-x)(-x), 0) \wedge \forall c : \mathsf{BHist},\ \mathsf{hsame}(((-x)(-x))c, 0) \wedge \mathsf{hsame}(c((-x)(-x)), 0).$$

The ten-field fit is: traditional target zero ideal closure under signed squares; BEDC source $\mathsf{Ring}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from signed zero-factor absorption, double-negative product, and zero-classifier factor absorption; ledger the zero witness, signed square endpoint, and arbitrary multiplier; core theorem this theorem; exactness the displayed package; downstream consumer* $\mathsf{Field}^{\uparrow}$ *signed-square and zero-product exits; status* $\mathsf{ProofTarget}$*.*

Proof. Specialize Definition 43.44 with $y = x$. Its third component gives the signed-square zero witness $\mathsf{hsame}((-x)(-x), 0)$. Equivalently, one may orient the same endpoint by combining Definition 43.51 with the ordinary zero-factor package.

Now apply Definition 43.40 with factor $(-x)(-x)$ and multiplier c. The signed-square zero witness supplies the factor premise, and the absorption theorem yields both $\mathsf{hsame}(((-x)(-x))c, 0)$ and $\mathsf{hsame}(c((-x)(-x)), 0)$.

Package the signed-square endpoint together with the universal multiplier statement. This is exactly the displayed zero-ideal consequence. □

Lean4 ✓ `BEDC.Derived.RingUp.ring signed square zero ideal package`

Definition 43.54 (Ring ledger policy)**.** *Record the shared carrier ledger and the operation tree for ring expressions.*

Definition 43.55 ($\mathsf{Ring}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Ring}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Ring}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Theorem 43.56 (Ring left distributivity over subtraction)**.** *Let R carry a history-level* $\mathsf{Ring}^{\uparrow}$ *certificate with classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. Write $x -_R y$ for $x +_R (-_R y)$. For every carried triple a, b, c,*

$$a \cdot_R (b -_R c) \sim_R (a \cdot_R b) -_R (a \cdot_R c).$$

The ten-field fit is: traditional target left distributivity over subtraction; BEDC source $\mathsf{Ring}^{\uparrow}$ *over the carried endpoints of R; carrier the ring carrier; classifier $\sim_R$; stability from the addition congruence and left distributivity fields of Definition 43.26; ledger the three endpoints and the hidden negative summand; core theorem this theorem; exactness the displayed subtraction law; standard bridge ordinary ring distribution over differences; downstream consumer polynomial and module linear-expression normalization; status* $\mathsf{ProofTarget}$*.*

Proof. By the difference abbreviation, the left side is $a \cdot_R (b +_R (-_R c))$. Left distributivity gives

$$a \cdot_R (b +_R (-_R c)) \sim_R (a \cdot_R b) +_R (a \cdot_R (-_R c)).$$

The right-negative product theorem Definition 43.49, read in this certificate, gives

$$a \cdot_R (-_R c) \sim_R -_R(a \cdot_R c).$$

Addition congruence transports this comparison in the right summand while the left summand $a \cdot_R b$ is held fixed. Hence

$$(a \cdot_R b) +_R (a \cdot_R (-_R c)) \sim_R (a \cdot_R b) +_R (-_R(a \cdot_R c)).$$

The final expression is $(a \cdot_R b) -_R (a \cdot_R c)$ by the same difference abbreviation. Transitivity of $\sim_R$ composes the distributivity comparison with the transported sign comparison and yields the displayed law. □

Lean4 ✓ `BEDC.Derived.RingUp.ring left distributivity over subtraction`

Proposition 43.57 (Ring forgets to Monoid certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with carrier classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $(-)_R$, multiplication $\cdot_R$, and identity 1_R. If the additive operations 0_R, $+_R$, and $(-)_R$, the additive abelian-group stability rows, both distributive rows, the zero-multiplication rows, and every ledger entry whose operation-tree provenance uses a dropped additive constructor are omitted, then the retained carrier, source, monoid pattern, classifier, monoid stability fields, and multiplicative ledger form a $\mathsf{Monoid}^{\uparrow}(R)$ certificate. The carrier R, multiplication $\cdot_R$, identity 1_R, and classifier $\sim_R$ are unchanged.*

Proof. By Definition 43.55, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Ring}^{\uparrow}(R)$. The retained package keeps exactly the rows whose endpoint claims and expression provenance use only the carrier classifier, multiplication, and the multiplicative identity.

By Definition 43.2, a ring carrier consists of a carrier with classifier, addition, zero, additive inverse, multiplication, one, and ring law fields. Removing the additive coordinates leaves the same carrier R, the same classifier $\sim_R$, the same multiplication $\cdot_R : R \times R \to R$, and the same identity 1_R. These are precisely the carrier data listed in Definition 40.1.

By Definition 43.23, the source side of a ring is an additive $\mathsf{AbGroup}^{\uparrow}$ and a multiplicative $\mathsf{Monoid}^{\uparrow}$ on the same carrier and classifier. Suppressing the additive source leaves the certified multiplicative source over R, $\cdot_R$, 1_R, and $\sim_R$. This is the source required by Definition 40.9.

By Definition 43.24, ring expressions are formed from 0, 1, +, unary minus, and multiplication. Restricting to expressions containing no additive constructor leaves expression formation from 1_R, carried endpoints, and binary multiplication, exactly the monoid pattern of Definition 40.10.

By Definition 43.25, the ring classifier is the shared carrier sameness relation, congruent for both operations. Retaining the relation $\sim_R$ and its multiplication-congruence row gives the classifier required by Definition 40.11. The addition-congruence row mentions the omitted operation and is not a monoid obligation.

For stability, Definition 43.26 lists additive abelian-group laws, multiplicative monoid laws, operation congruence, left distributivity, right distributivity, and zero multiplication laws. The multiplicative monoid laws give associativity and the left and right identity laws for 1_R under $\sim_R$; the retained multiplication-congruence row gives congruence in both arguments. The carrier-classifier rows are unchanged by the projection, so reflexivity and transitivity remain available. Together these are exactly the rows listed in Definition 40.13. The omitted additive, distributive, and zero-multiplication rows mention 0_R, $+_R$, or $(-)_R$, so they have no corresponding $\mathsf{Monoid}^{\uparrow}$ obligation.

Finally, Definition 43.54 records the shared carrier ledger and the operation tree for ring expressions. Removing ledger entries whose provenance uses a dropped additive constructor leaves carrier witnesses, the identity endpoint, and multiplication trees for monoid expressions. This is the ledger policy of Definition 40.33. Hence the retained data satisfy the obligations named in Definition 40.34, with R, $\cdot_R$, 1_R, and $\sim_R$ copied from the original $\mathsf{Ring}^{\uparrow}(R)$ certificate. □

Theorem 43.58 (Ring right distributivity over subtraction)**.** *Let R carry a history-level $\mathsf{Ring}^{\uparrow}$ certificate with classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. Write $x -_R y$ for $x +_R (-_R y)$. For every carried triple a, b, c,*

$$(a -_R b) \cdot_R c \sim_R (a \cdot_R c) -_R (b \cdot_R c).$$

The ten-field fit is: traditional target right distributivity over subtraction; BEDC source $\mathsf{Ring}^{\uparrow}$ over the carried endpoints of R; carrier the ring carrier; classifier $\sim_R$; stability from the addition congruence and right distributivity fields of Definition 43.26; ledger the three endpoints and the hidden negative summand; core theorem this theorem; exactness the displayed subtraction law; standard bridge ordinary ring distribution over differences; downstream consumer polynomial and module linear-expression normalization; status ProofTarget.

Proof. By the difference abbreviation, the left side is $(a +_R (-_R b)) \cdot_R c$. Right distributivity gives

$$(a +_R (-_R b)) \cdot_R c \sim_R (a \cdot_R c) +_R ((-_R b) \cdot_R c).$$

The left-negative product theorem Definition 43.48, read in this certificate, gives

$$(-_R b) \cdot_R c \sim_R -_R(b \cdot_R c).$$

Addition congruence transports this comparison in the right summand while the left summand $a \cdot_R c$ is held fixed. Hence

$$(a \cdot_R c) +_R ((-_R b) \cdot_R c) \sim_R (a \cdot_R c) +_R (-_R(b \cdot_R c)).$$

The final expression is $(a \cdot_R c) -_R (b \cdot_R c)$ by the same difference abbreviation. Transitivity of $\sim_R$ composes the distributivity comparison with the transported sign comparison and yields the displayed law. □

Lean4 ✓ `BEDC.Derived.RingUp.ring right distributivity over subtraction`

Proposition 43.59 (Ring forgets to AbGroup certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with carrier classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, and identity 1_R. If the multiplicative monoid source package, $\cdot_R$, 1_R, the multiplicative monoid stability rows, both distributive rows, the zero-multiplication rows, and every ledger entry whose operation-tree provenance uses a multiplicative constructor are omitted, then the retained carrier, source, additive pattern, classifier, additive stability fields, and additive ledger form an $\mathsf{AbGroup}^{\uparrow}(R)$ certificate. The carrier R, addition $+_R$, zero 0_R, additive inverse $-_R$, and classifier $\sim_R$ are unchanged.*

Proof. By Definition 43.55, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Ring}^{\uparrow}(R)$. The retained package keeps exactly the rows whose endpoint claims and expression provenance use only the carrier classifier, additive zero, addition, and additive inverse.

By Definition 43.2, a ring carrier consists of a carrier with classifier, addition, zero, additive inverse, multiplication, one, and ring law fields. Removing the multiplicative coordinates leaves

the same carrier R, the same classifier $\sim_R$, the same addition $+_R : R \times R \to R$, the same zero 0_R, and the same additive inverse $-_R$. Reading addition as the abelian-group operation, zero as its identity, and $-_R$ as its inverse operation gives precisely the carrier data required by Definition 42.1.

By Definition 43.23, the source side of a ring is an additive $\mathsf{AbGroup}^{\uparrow}$ and a multiplicative $\mathsf{Monoid}^{\uparrow}$ on the same carrier and classifier. Suppressing the multiplicative source leaves the certified additive $\mathsf{AbGroup}^{\uparrow}$ source over R, $+_R$, 0_R, $-_R$, and $\sim_R$. Unpacking that source through Definition 42.5 gives the certified group carrier plus commutativity field required for the abelian-group source.

By Definition 43.25, the ring classifier is the shared carrier sameness relation, congruent for both operations. Retaining $\sim_R$ and the addition-congruence row gives the classifier required by Definition 42.7. The multiplication-congruence row mentions the omitted operation and has no abelian-group obligation.

By Definition 43.24, ring expressions are formed from 0_R, 1_R, $+_R$, $-_R$, and $\cdot_R$. Restricting to expressions containing no multiplicative constructor leaves carried endpoints, 0_R, addition, and additive inverse. Together with the retained commutativity permission from the additive abelian-group certificate, this is the abelian-group pattern of Definition 42.6.

By Definition 43.26, the stability certificate contains additive abelian-group laws, multiplicative monoid laws, addition and multiplication congruence, distributivity, and zero multiplication laws. The retained additive abelian-group laws supply the group stability fields, commutativity under the classifier, and the compatibility data for commutativity with inverse and operation congruence listed in Definition 42.8. The retained addition-congruence row is the operation-congruence row for the abelian-group operation. The multiplicative monoid rows, multiplication congruence, distributive rows, and zero-multiplication rows all mention $\cdot_R$ or 1_R, so they are not obligations for $\mathsf{AbGroup}^{\uparrow}$.

Finally, Definition 43.54 records the shared carrier ledger and the operation tree for ring expressions. Removing entries whose provenance uses 1_R or $\cdot_R$ leaves carrier witnesses, the additive identity endpoint, addition trees, additive inverse trees, and the commutativity-use records from the additive source. This is the group-expression provenance and commutativity ledger of Definition 42.31. Hence the retained data satisfy the obligations named in Definition 42.32, with R, $+_R$, 0_R, $-_R$, and $\sim_R$ copied from the original $\mathsf{Ring}^{\uparrow}(R)$ certificate. □

Theorem 43.60 (Ring zero absorption from distributivity)**.** *Let R carry a history-level $\mathsf{Ring}^{\uparrow}$ certificate with classifier $\sim_R$, addition $+$, zero endpoint 0_R, and multiplication $\cdot$. Assume the additive component supplies additive left cancellation under $\sim_R$, as obtained from Definition 41.66 for the additive group part of Definition 43.23. If the ring stability fields of Definition 43.26 supply multiplication congruence, left distributivity, and right distributivity, then every endpoint x satisfies*

$$0_R \cdot x \sim_R 0_R \qquad \textit{and} \qquad x \cdot 0_R \sim_R 0_R.$$

Proof. The additive left-zero row gives $0_R + 0_R \sim_R 0_R$. By multiplication congruence, for any x,

$$x \cdot (0_R + 0_R) \sim_R x \cdot 0_R.$$

Left distributivity gives

$$x \cdot (0_R + 0_R) \sim_R (x \cdot 0_R) + (x \cdot 0_R).$$

Writing $u = x \cdot 0_R$, symmetry and transitivity of the classifier give $u \sim_R u + u$. The additive right-zero theorem, Definition 43.31, gives $u + 0_R \sim_R u$, hence

$$u + 0_R \sim_R u + u.$$

Additive left cancellation of the common summand u yields $0_R \sim_R u$, and symmetry gives $x \cdot 0_R \sim_R 0_R$.

For the other side, multiplication congruence gives

$$(0_R + 0_R) \cdot x \sim_R 0_R \cdot x,$$

while right distributivity gives

$$(0_R + 0_R) \cdot x \sim_R (0_R \cdot x) + (0_R \cdot x).$$

Writing $v = 0_R \cdot x$, the same transitivity step gives $v \sim_R v + v$. Again Definition 43.31 gives $v + 0_R \sim_R v$, so

$$v + 0_R \sim_R v + v.$$

Additive left cancellation of the common summand v gives $0_R \sim_R v$, and symmetry gives $0_R \cdot x \sim_R 0_R$. □

Theorem 43.61 (Ring finite expression ledger induction)**.** *Let R carry a history-level* $\mathsf{Ring}^{\uparrow}$ *certificate with carrier C_R, classifier $\sim_R$, addition $+$, multiplication $\cdot$, zero endpoint 0_R, negation $-$, stability rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. Every finite* $\mathsf{Ring}^{\uparrow}$ *expression ledger whose entries are generated from displayed endpoints by the additive rows, multiplicative rows, zero row, negation row, stability rows, and local* NameCert *row admits a constructor-by-constructor BEDC induction. Each induction branch reads only the visible ledger entry and the earlier finite subledgers named by its constructor.*

Consequently any equality, absorption, distributivity, or normal-form read over such a ledger is justified by finite BHist *inspection of the same rows, not by an ambient quotient-ring induction principle or a host algebraic normalizer.*

Proof. Begin with the generating rows. Endpoint entries, the zero endpoint, and named NameCert rows are already visible as finite BHist data, so the base cases require no quotient carrier or ambient equality class.

For the additive, multiplicative, and negation constructors, inspect the displayed operation row that produced the ledger entry. The induction hypotheses are attached to the finite input subledgers, and the stability rows transport the corresponding classifier reads through $+$, $\cdot$, and $-$.

Finally read componentwise hsame transport, displayed Cont replay, and Pkg provenance. These structural rows repack the same finite ledger without adding a host quotient coordinate. Thus the induction remains a BEDC induction over the recorded expression ledger, and every algebraic read stays inside the displayed $\mathsf{Ring}^{\uparrow}$ certificate surface. □

Theorem 43.62 (Ring quotient-free normal-form route)**.** *For the same history-level* $\mathsf{Ring}^{\uparrow}$ *certificate, any normal-form read produced from the finite expression ledger of Definition 43.61 remains a quotient-free route through* BHist*,* Cont*,* Pkg*, classifier stability, and local* NameCert *data. A normal form may cite additive associativity, additive inverse, zero absorption, multiplication stability, and distributivity only as displayed certificate rows over $\sim_R$.*

It cannot be reread as equality in a host quotient ring, equality of quotient representatives, a selected canonical representative, a polynomial normalization algorithm outside the packet, or an ambient algebra theorem over an unrecorded carrier.

Proof. Use the finite expression-ledger induction. Each normal-form step is one of the ledger constructors or one of the displayed stability rows, so its inputs and output are still named histories connected by the carrier classifier.

The zero-absorption theorem above supplies the only multiplication-by-zero simplification used here, and it was derived from distributivity plus additive cancellation inside the same classifier. Additive inverse, associativity, and multiplication stability are read in the same way: as rows of the $\mathsf{Ring}^{\uparrow}$ certificate, not as quotient-ring facts.

The structural rows Cont and Pkg preserve provenance of the normal-form read. Since no row stores a quotient carrier, selected representative, external polynomial normalizer, or ambient ring object, the normal-form route is exactly the finite BEDC route described in the statement. □

Theorem 43.63 (RingUp public singleton operation surface)**.** *For the concrete singleton history ring instance of Definition 43.3, the public surface consists of the singleton carrier A_{e}, the classifier R_{e}, and the five displayed operations*

$$\mathsf{add}(h,k) := \mathsf{append}(h,k), \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e},$$

together with

$$\mathsf{mul}(h,k) := \mathsf{append}(h,k), \qquad 1 := \mathsf{e}.$$

On carried endpoints this surface is closed under addition, negation, multiplication, zero, and one; append endpoints have exact singleton classifier readback; multiplication continuations classify exactly through the singleton continuation row; multiplication by zero is absorbed; and the exposed ledger contains only the operation tree plus singleton endpoint witnesses, so the zero-classifier ideal package contributes no hidden endpoint channel.

Proof. The carrier, classifier, and operation rows are those displayed in Definition 43.3. The RingUp laws of Theorem 43.4 supply the additive group rows, multiplicative monoid rows, congruence rows, and both distributivity rows for this same tuple.

Endpoint exactness is read from the singleton classifier lemmas. The append commutation row of Theorem 43.12 exposes no data beyond the two singleton endpoint witnesses, while Theorem 43.15 records that continuation readback after multiplication is again exactly the singleton carrier condition on the continuation input and output.

The zero boundary is public for the same reason. The zero-factor absorption of Theorem 43.39 prevents a zero multiplication branch from producing a visible endpoint, and Theorem 43.41 packages addition, negation, and two-sided multiplication closure for zero-classified inputs. Thus the public ledger exposes precisely the operation tree and the singleton endpoint certificates already named above, with no additional history data available behind the classifier. □

Theorem 43.64 (RingUp singleton operation table exhaustion)**.** *Every public $\mathsf{Ring}^{\uparrow}$ consumer read from the concrete singleton certificate is exhausted by the five operation rows* add, neg, 0, mul, *and* 1, *together with their singleton endpoint witnesses. No read of the public certificate exposes an additional history endpoint, quotient carrier, polynomial row, module row, or non-singleton multiplication table.*

Proof. Begin with the public singleton operation surface of Definition 43.63. It lists the carrier A_{e}, classifier R_{e}, and exactly the five displayed operation rows.

The operation laws come from Definition 43.4. Since every carried endpoint is classified with e, each addition, negation, zero, multiplication, and one read returns the same singleton endpoint witness.

The remaining public boundary does not add a sixth operation row. Definition 43.39 controls the zero-multiplication branch, and Definition 43.41 keeps the zero-classifier package inside the same singleton endpoint. Thus every public consumer read is one of the five operation-table rows. □

Theorem 43.65 (RingUp algebra sibling route)**.** *The singleton public* $\mathsf{Ring}^{\uparrow}$ *surface consumes additive-group rows from chapter 42, multiplicative-unit rows from chapter 40, and commutative specialization rows from chapter 44 before downstream algebra consumers read* $\mathsf{Ring}^{\uparrow}$*. This route exposes the five-operation table without asserting a quotient ring or a non-singleton carrier.*

Proof. Project the public table of Definition 43.64. The additive rows add, neg, and 0 are read through the abelian-group surface, so every additive consumer keeps the $\mathsf{AbGroup}^{\uparrow}$ sibling route.

The multiplicative rows mul and 1 are read through the monoid surface. The singleton classifier and endpoint witnesses keep this multiplicative handoff inside the same finite operation table.

Finally, any commutative specialization is routed through the $\mathsf{CommRing}^{\uparrow}$ sibling before it is exported to downstream algebra. Since the projected data are exactly the five singleton operation rows and their endpoint witnesses, the route introduces no quotient-ring claim and no non-singleton carrier. □

Theorem 43.66 (RingUp singleton public certificate scope)**.** *The public scope of the concrete singleton* $\mathsf{Ring}^{\uparrow}$ *certificate consists of ten fields: the BEDC source* BHist*, the carrier* A_{e}*, the classifier* R_{e}*, stability under singleton transport, the public* add *row, the public* neg *row, the public* 0 *row, the public* mul *row, the public* 1 *row, and the standard bridge to the terminal ring. The five operation fields form the visible ledger table, and their endpoint exactness says precisely that a consumer row is present iff its endpoint is the singleton* e *endpoint.*

Proof. Project the singleton source and algebraic laws from Definition 43.4. The source is the displayed BHist row, while A_{e} and R_{e} give the carrier and classifier over the same endpoint.

Next read the public operation table through Definition 43.63 and Definition 43.64. These rows account for addition, negation, zero, multiplication, and one, and each read carries the singleton endpoint iff supplied by the classifier.

Finally apply Definition 43.39 and Definition 43.41 to keep the zero and multiplication boundary inside the same public table. Repacking these rows with the checked standard bridge exposes the terminal ring and no additional consumer channel. □

Lean4 ✓ `BEDC.Derived.RingUp.RingUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$**.**

Constructive story: The ring surface reuses one empty-history endpoint twice, letting the same singleton carry both the additive and multiplicative roles. A history is admitted only when it is history-equal to e, and the classifier is the corresponding singleton relation, so addition by continuation, negation, zero, multiplication by continuation, and one all land back at the same endpoint. The source is the singleton predicate, the pattern records those operation rows together with distributivity, zero absorption, and the signed-square and subtraction packages, and the ledger keeps every visible singleton aligned with its raw history. Since a one-element history class is inhabited by e and history-equality is an equivalence that preserves the class, the five name-certificate obligations hold without any separate algebraic carrier. A history h lies in the signature gap of a $\mathsf{Ring}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; after unfolding, h is history-equal to e and p is the singleton ring element. The public operation surface and terminal bridge expose this same endpoint table to downstream consumers, including the opposite-ring certificate. The chapter registers this two-operation singleton construction under the name $\mathsf{Ring}^{\uparrow}$; addition and multiplication share one classifier on one endpoint, with no polynomial, quotient, field, or non-singleton ring structure introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the mature singleton RingUp package consisting of the carrier A_{e}, classifier R_{e}, displayed addition, negation, zero, multiplication, one, distributivity, zero-absorption, signed-square and subtraction packages, opposite-ring certificate, and public singleton operation surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{Ring}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.RingUp.RingSingletonEmptyHistory\_laws`

Bridge status: bridgeChecked

Not claimed: It does not close arbitrary non-singleton rings, polynomial rings, fields, modules, quotient-ring theory, or machine-checked standard bridge data.

Upgrade path: No further upgrade: the chapter is at the top of the formal axis.

Chapter 44

A Concrete Naming Certificate for $\mathsf{CommRing}^{\uparrow}$

The commutative ring interface $\mathsf{CommRing}^{\uparrow}$ extends chapter 43 with the commutativity field $a \cdot b = b \cdot a$ for the multiplicative operation under the carrier's classifier. The integer ring from chapter 31 is the canonical concrete instance.

44.1 Carrier definition

Definition 44.1 (CommRing carrier)**.** *Use a ring setup extended with commutativity of multiplication.*

44.2 Concrete singleton history commutative ring instance

Definition 44.2 (Concrete singleton history commutative ring instance)**.** *On the concrete* BHist *carrier, define* $A_{\mathsf{e}}(h) := \mathsf{hsame}(h, \mathsf{e})$ *and*

$$R_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

The commutative-ring operation tuple is

$$\mathsf{add}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e},$$

with multiplication and unit

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad 1 := \mathsf{e}.$$

Theorem 44.3 (Concrete singleton history commutative ring laws)**.** *The concrete singleton history commutative ring instance of Definition 44.2 has carrier containing* 0*, is closed under addition, negation, and multiplication, has left and right additive zero laws under* R_{e}*, and satisfies multiplicative commutativity under* R_{e}*.*

Proof. The carrier witness $A_{\mathsf{e}}(0)$ is reflexive $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. For addition and multiplication closure, if $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$ hold, then both inputs are the empty history, so $\mathsf{append}(x, y)$ reduces to e. The negation operation is constantly e, so its closure is immediate from the same reflexive endpoint witness.

For the two additive zero laws, the singleton input witness identifies x with e; after that identification, both $\mathsf{append}(\mathsf{e}, x)$ and $\mathsf{append}(x, \mathsf{e})$ are the empty history, and the classifier packages the two carrier endpoints with reflexive hsame. Multiplicative commutativity is identical: two singleton carrier witnesses identify x and y with e, so the two append products are both e and are related by R_{e}. □

Lean4 ✓ `BEDC.Derived.CommRingUp.concrete singleton history commring laws`

Theorem 44.4 (Concrete singleton history CommRingUp full schema laws)**.** *For the concrete singleton history commutative ring instance of Definition 44.2, the tuple* $(A_{\mathsf{e}}, R_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1)$ *satisfies the full* $\mathsf{CommRing}^{\uparrow}$ *law package: all* $\mathsf{Ring}^{\uparrow}$ *laws from Theorem 43.4, multiplicative commutativity from Theorem 44.3, and stability of the commutativity row under* R_{e}*-classified replacement of carried inputs. Its certificate fit has ten fields: traditional target the one-element commutative ring; BEDC source* BHist*; carrier* A_{e}*; classifier* R_{e}*; stability obligations the* $\mathsf{Ring}^{\uparrow}$ *package plus commutativity compatibility; ledger append-operation provenance; core theorem Theorem 44.4; exactness singleton endpoint classification; standard bridge the terminal commutative ring; status* $\mathsf{ProofTarget}$*.*

Proof. The $\mathsf{Ring}^{\uparrow}$ part is exactly Theorem 43.4 for the displayed append-operation tuple. It supplies the additive abelian-group laws, multiplicative monoid laws, addition and multiplication congruence, and the two distributive laws over A_{e} and R_{e}.

The commutativity row is exactly Theorem 44.3, read at the same append multiplication. For carried inputs x and y, the singleton endpoint witnesses identify both products $\mathsf{mul}(x, y)$ and $\mathsf{mul}(y, x)$ with e, so the displayed row is an R_{e}-classification.

For compatibility of that row, suppose the two carried inputs are replaced by R_{e}-classified inputs. Multiplication congruence from the $\mathsf{Ring}^{\uparrow}$ package transports $\mathsf{mul}(x, y)$ to the replacement product, and the same congruence transports $\mathsf{mul}(y, x)$ to the reversed replacement product. Compose these transported relations with the commutativity row by symmetry and transitivity of hsame; the carrier fields are preserved by the same singleton endpoint classification, so the commutativity row remains R_{e}-classified after replacement. □

Lean4 ✓ `BEDC.Derived.CommRingUp.concrete singleton history commring full schema laws`

Theorem 44.5 (Concrete singleton history CommRingUp certificate boundary)**.** *For the concrete tuple of Definition 44.2, the carrier-aware classifier* R_{e} *supplies the* $\mathsf{SemanticNameCert}$ *rows. Together with Theorem 44.4, these rows supply the concrete* $\mathsf{CommRing}^{\uparrow}$ *certificate boundary over* BHist *with the following ten fields:*

1. *traditional target: the terminal commutative ring;*
2. *BEDC source: the history carrier* BHist *with source* A_{e}*;*
3. *carrier:* A_{e}*;*
4. *classifier:* R_{e}*;*
5. *stability obligations: the* $\mathsf{Ring}^{\uparrow}$ *package plus multiplicative commutativity compatibility;*
6. *ledger: append-operation provenance for addition and multiplication, constant-operation provenance for negation, and empty-history provenance for* 0 *and* 1*;*
7. *core theorem: Theorem 44.4;*
8. *exactness: singleton endpoint classification by* A_{e} *and* hsame*;*

9. *standard bridge: the ordinary terminal commutative ring;*
10. *status:* ProofTarget.

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingUp concrete to schema`

Proof. Apply Theorem 41.88 to the singleton history carrier A_{e} and classifier R_{e}. It supplies reflexivity, symmetry, transitivity, and carrier transport for the carrier-aware semantic naming certificate.

The operation endpoints are exactly the endpoints displayed in Definition 44.2: append for addition and multiplication, the constant endpoint e for negation, and e for both distinguished constants. Thus the BEDC source, carrier, classifier, ledger, and exactness rows are the concrete rows named in the definition.

The law rows are supplied by Theorem 44.4. That theorem packages the additive, multiplicative, distributive, congruence, and multiplicative-commutativity rows over the same tuple.

Combining the SemanticNameCert rows with the law package gives the displayed ten-field $\mathsf{CommRing}^{\uparrow}$ certificate boundary. The terminal commutative ring is the standard bridge, and the boundary is recorded with status ProofTarget. □

44.3 Concrete singleton empty-history commutative ring instance

Definition 44.6 (Singleton empty-history commutative ring instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define the classifier*

$$\rho_{\mathsf{e}}(h,k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Define the concrete commutative ring operations by

$$\mathsf{add}(h,k) := \mathsf{e}, \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e}, \qquad \mathsf{mul}(h,k) := \mathsf{e}, \qquad 1 := \mathsf{e}.$$

Equip this singleton ring tuple with the multiplicative commutativity field

$$\rho_{\mathsf{e}}(\mathsf{mul}(x,y), \mathsf{mul}(y,x)).$$

The concrete commutative-ring certificate has the following ten fields:

1. *traditional target: the one-element commutative ring;*
2. *BEDC source:* BHist *with the singleton empty-history ring instance;*
3. *carrier: A_{e};*
4. *classifier: ρ_{e};*
5. *stability obligations: the singleton ring laws, multiplicative commutativity, and multiplication-commutativity compatibility;*
6. *ledger: constant-operation provenance for* 0, 1, *addition, negation, and multiplication;*
7. *core theorem: Theorem 44.7;*
8. *exactness: endpoint classification by membership in the singleton carrier and* hsame*;*
9. *standard bridge: the ordinary one-element commutative ring;*
10. *status:* ProofTarget.

Theorem 44.7 (Singleton empty-history commutative ring laws)**.** *The singleton empty-history commutative-ring tuple supplies a concrete* SemanticNameCert *for the carrier* A_{e} *and classifier* ρ_{e}*. Its operation rows include the displayed constant* add*,* neg*, and* mul *endpoints, multiplicative commutativity, both distributive laws, and the left unit row for* 1 *under* ρ_{e}*.*

Proof. The carrier-aware classifier ρ_{e} is a semantic naming certificate. Carrier inhabitance is witnessed by e. Reflexivity uses the given carrier witness and $\mathsf{hsame}(h, h)$; symmetry swaps the endpoint carrier witnesses and uses symmetry of hsame; transitivity composes the hsame components and keeps the outer carrier witnesses. Carrier transport is the right endpoint carrier field of the classifier.

The concrete add, neg, and mul endpoint rows close because all three displayed operations are constant at e. The two carrier witnesses are therefore singleton witnesses for e, and the relation component is reflexive $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

The ring-law part is exactly Theorem 43.6 applied to the displayed singleton empty-history operation tuple. This supplies carrier inhabitance, additive abelian-group laws, multiplicative monoid laws, operation congruence, both distributive laws, the unit laws, and zero multiplication laws.

For multiplicative commutativity, both $\mathsf{mul}(x, y)$ and $\mathsf{mul}(y, x)$ are definitionally e. The two endpoint witnesses for ρ_{e} are the singleton carrier witnesses for e, and the relation component is $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.singleton empty history commring laws`

Proposition 44.8 (CommRing forgets to Ring certificate)**.** *Let* R *carry a* $\mathsf{CommRing}^{\uparrow}(R)$ *certificate. If the multiplication-commutativity row and its commutative-ring-specific ledger entries are omitted from the exported package, the retained carrier, source, ring pattern, classifier, ring stability fields, and ring ledger form a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate. The operations* 0_R*,* 1_R*,* $+_R$*,* $(-)_R$*,* $\cdot_R$*, and the carrier classifier on* R *are unchanged.*

Proof. By Definition 44.59, a commutative-ring certificate supplies the same carrier operations as a ring certificate, together with the extra multiplication-commutativity requirement. Removing only that extra row leaves the carrier data of Definition 43.2: zero, one, addition, additive inverse, multiplication, and the same carrier classifier.

The source, pattern, classifier, stability, and ledger rows retained by this projection are exactly the ring rows. The source still contains the additive abelian-group and multiplicative monoid data required by Definition 43.23; the pattern still generates the ring expression language; and the stability rows still contain the ring congruence, distributivity, zero-multiplication, and unit laws listed in Definition 43.26.

The discarded entries are precisely those whose only role is to witness or record multiplication commutativity. Therefore the retained rows satisfy Definition 43.55 with the same operations and classifier carried by the original $\mathsf{CommRing}^{\uparrow}$ certificate. □

Theorem 44.9 (CommRing singleton negation empty endpoint)**.** *For histories* $h, t : \mathsf{BHist}$ *with* $A_{\mathsf{e}}(h)$*,*

$$\rho_{\mathsf{e}}(\mathsf{CommRingSingletonNeg}(h), t) \iff A_{\mathsf{e}}(t) \wedge \mathsf{hsame}(\mathsf{CommRingSingletonNeg}(h), \mathsf{e}).$$

Proof. The singleton commutative-ring negation endpoint is definitionally e. Unfolding the carrier-aware classifier gives the forward direction directly: the right carrier field is $A_{\mathsf{e}}(t)$ and the left carrier field is the empty endpoint for $\mathsf{CommRingSingletonNeg}(h)$.

For the converse, the same two endpoint facts are the left and right carrier fields. Symmetry of hsame compares t back to e, and transitivity through the negation endpoint supplies the relation component of ρ_{e}. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier neg empty endpoint iff`

Theorem 44.10 (CommRing singleton empty multiplication context carrier)**.** *For singleton empty-history CommRing histories L and R, the classifier comparison*

$$\rho_{\mathsf{e}}(\mathsf{append}(L, \mathsf{mul}(x, y)), \mathsf{append}(R, out))$$

holds exactly when out lies in the singleton empty carrier.

Proof. The multiplication endpoint of the singleton empty-history CommRing tuple is definitionally e. In the forward direction, the right carrier field of the classifier identifies $\mathsf{append}(R, out)$ with e, and append-empty inversion extracts *out*hsamee.

Conversely, if *out*hsamee, then the carrier witnesses for L and R rebuild the two append endpoints as empty endpoints by append-empty inversion. Transitivity through e compares the endpoints, so the classifier package is recovered. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commringSingletonEmptyClassifier mul context carrier iff`

Theorem 44.11 (CommRing singleton empty operation context carrier package)**.** *For singleton empty-history CommRing histories L and R, the three operation contexts*

$$\mathsf{append}(L, \mathsf{add}(x, y)), \quad \mathsf{append}(L, \mathsf{neg}(x)), \quad \mathsf{append}(L, \mathsf{mul}(x, y))$$

are classified against $\mathsf{append}(R, out)$ *precisely when out lies in the singleton empty carrier.*

Proof. The add, neg, and mul endpoints of the singleton empty-history CommRing tuple are all definitionally e. The forward direction reads the right carrier field of the classifier and applies append-empty inversion to recover *out*hsamee. Conversely, the carrier witnesses for L, R, and *out* rebuild both append endpoints as empty endpoints, and transitivity through e supplies the classifier relation. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commringSingletonEmptyClassifier operation context carrier package`

Theorem 44.12 (CommRing singleton classifier visible-right exclusion)**.** *For histories h, p, q :* BHist*, neither $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e0}(q)))$ nor $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e1}(q)))$ can hold.*

Proof. Unfold the singleton commutative-ring classifier. In either branch, the right carrier field says that the displayed append endpoint is e. The empty-result inversion theorem for append splits this into an empty prefix and an empty visible tail. The latter would identify $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$ with e, contradicting the constructor separation of BHist. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append visible right absurd`

Theorem 44.13 (CommRing singleton classifier visible-prefix exclusion)**.** *For histories p, q, h :* BHist*, neither $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e0}(p), q), h)$ nor $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e1}(p), q), h)$ can hold.*

Proof. Unfold the singleton commutative-ring classifier. In either branch, the left carrier field says that the displayed append endpoint is e. The empty-result inversion theorem for append splits this into an empty visible prefix and an empty suffix. The former would identify $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$ with e, contradicting the constructor separation of BHist. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append visible prefix absurd`

Theorem 44.14 (CommRing singleton classifier visible boundary package)**.** *For histories h, p, q : BHist, the singleton commutative-ring classifier has the two-sided visible append boundary exclusion*

$$(\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e0}(q))) \to \bot) \wedge (\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e1}(q))) \to \bot) \wedge (\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e0}(p), q), h) \to \bot) \wedge (\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e1}(p), q),$$

The ten-field fit is: traditional target singleton classifier no-confusion for visible append boundaries; BEDC source singleton $\mathsf{CommRing}^{\uparrow}$ *over* BHist*; carrier A_{e}; classifier ρ_{e}; stability from the two checked classifier exclusions; ledger the visible prefix/tail constructor; core theorem this block; exactness the four displayed impossibility clauses; downstream consumer singleton CommRing certificate boundary and* $\mathsf{Field}^{\uparrow}$ *terminal-object no-confusion; status* $\mathsf{ProofTarget}$.

Proof. The two right-endpoint clauses are exactly the two branches of Definition 44.12. They exclude classifier witnesses whose second endpoint is an append ending in $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$.

The two left-endpoint clauses are exactly the two branches of Definition 44.13. They exclude classifier witnesses whose first endpoint is an append beginning with $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$.

Repack these four one-sided implications in the displayed order. No additional constructor analysis is needed beyond the checked visible-right and visible-prefix exclusions. □

Theorem 44.15 (CommRing singleton carrier visible exclusion)**.** *For every history p : BHist, neither $\mathsf{e0}(p)$ nor $\mathsf{e1}(p)$ lies in the singleton commutative-ring carrier.*

Proof. Unfold the singleton carrier as equality with e. The two generated history constructors are separated from the empty history, so both endpoint carrier witnesses are impossible. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonCarrier visible absurd`

Theorem 44.16 (CommRing singleton carrier visible-head append exclusion)**.** *For histories h, k : BHist, neither $\mathsf{append}(\mathsf{e0}(h), k)$ nor $\mathsf{append}(\mathsf{e1}(h), k)$ lies in the singleton commutative-ring carrier.*

Proof. In either branch, the singleton carrier identifies the displayed append history with e. Empty-result inversion for append then identifies the visible head $\mathsf{e0}(h)$ or $\mathsf{e1}(h)$ with e, contradicting constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonCarrier append visible head absurd`

Theorem 44.17 (CommRing singleton carrier visible-tail append exclusion)**.** *For histories p, q : BHist, neither $\mathsf{append}(p, \mathsf{e0}(q))$ nor $\mathsf{append}(p, \mathsf{e1}(q))$ lies in the singleton commutative-ring carrier.*

Proof. In either branch, the singleton carrier identifies the displayed append history with e. Empty-result inversion for append then identifies the visible tail $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$ with e, contradicting constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonCarrier append visible tail absurd`

Theorem 44.18 (CommRing singleton carrier append commutation)**.** *For histories h, k : BHist,*

$$\mathsf{CommRingSingletonCarrier}(\mathsf{append}(h, k)) \iff \mathsf{CommRingSingletonCarrier}(\mathsf{append}(k, h)).$$

Proof. In either direction, empty-result inversion for append splits the singleton carrier witness into singleton witnesses for the two factors. Reversing the two factors and applying empty-result construction rebuilds the singleton carrier witness for the commuted append. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonCarrier append comm iff`

Theorem 44.19 (CommRing singleton classifier append context cancellation)**.** *For singleton commutative-ring carrier contexts* $L, R : \mathsf{BHist}$ *and histories* $Q, S : \mathsf{BHist}$*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(L, Q), \mathsf{append}(R, S)) \iff \rho_{\mathsf{e}}(Q, S).$$

Proof. In the forward direction, the two classifier carrier fields identify $\mathsf{append}(L, Q)$ and $\mathsf{append}(R, S)$ with e. Empty-result inversion for append splits these into endpoint witnesses for Q and S. Those two recovered carrier witnesses compose with symmetry to give $\mathsf{hsame}(Q, S)$.

Conversely, if $\rho_{\mathsf{e}}(Q, S)$ holds and both contexts are singleton carriers, empty-result construction for append gives singleton carrier witnesses for $\mathsf{append}(L, Q)$ and $\mathsf{append}(R, S)$. Composing the two append carrier witnesses with symmetry gives the classifier relation component between the contextual endpoints. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append context cancel iff`

Theorem 44.20 (CommRing singleton classifier append context carrier split)**.** *If the singleton commutative-ring classifier relates two appended endpoints, then all four append factors are singleton carriers:*

$$\rho_{\mathsf{e}}(\mathsf{append}(L, Q), \mathsf{append}(R, S)) \implies A_{\mathsf{e}}(L) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(R) \wedge A_{\mathsf{e}}(S).$$

Proof. The classifier supplies singleton-carrier witnesses for $\mathsf{append}(L, Q)$ and $\mathsf{append}(R, S)$. Empty-result inversion for append splits the first witness into carrier witnesses for L and Q, and splits the second witness into carrier witnesses for R and S. Conjunction introduction packages the four factors. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append context carrier split`

Theorem 44.21 (CommRing singleton classifier append-left carrier exactness)**.** *For histories* $h, k, out : \mathsf{BHist}$*, the singleton commutative-ring classifier relates the appended left endpoint to* out *exactly when all three visible histories are singleton carriers:*

$$\rho_{\mathsf{e}}(\mathsf{append}(h, k), out) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge A_{\mathsf{e}}(out).$$

Proof. The forward direction projects the left endpoint carrier from the classifier and applies empty-result inversion for append, yielding singleton carrier witnesses for h and k. The right endpoint carrier is the second classifier field.

Conversely, the carrier witnesses for h and k rebuild a singleton carrier witness for $\mathsf{append}(h, k)$ by empty-result construction. Together with the carrier witness for out, transitivity through e and symmetry give the displayed singleton classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append left carrier iff`

Theorem 44.22 (CommRing singleton classifier append-right carrier exactness)**.** *For histories* $h, k, out : \mathsf{BHist}$*, the singleton commutative-ring classifier relates* out *to the appended right endpoint exactly when all three visible histories are singleton carriers:*

$$\rho_{\mathsf{e}}(out, \mathsf{append}(h, k)) \iff A_{\mathsf{e}}(out) \wedge A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction projects the right endpoint carrier from the classifier and splits the append carrier by empty-result inversion. Conversely, singleton carrier witnesses for h and k rebuild the appended right carrier. Together with the carrier witness for *out*, transitivity through e gives the displayed classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append right carrier iff`

Theorem 44.23 (CommRing singleton classifier append commuted factors)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, if the singleton commutative-ring classifier relates the two appended endpoints*

$$\rho_{\mathsf{e}}(\mathsf{append}(h, k), \mathsf{append}(h', k')),$$

then it also relates the endpoints with the two append factors exchanged:

$$\rho_{\mathsf{e}}(\mathsf{append}(k, h), \mathsf{append}(k', h')).$$

Proof. The classifier gives singleton carrier witnesses for $\mathsf{append}(h, k)$ and $\mathsf{append}(h', k')$. Empty-result inversion for append splits each witness into singleton carrier witnesses for its two factors. Swapping the two factors on each side and applying empty-result construction rebuilds singleton carrier witnesses for $\mathsf{append}(k, h)$ and $\mathsf{append}(k', h')$. Both rebuilt endpoints are therefore classified through e, so transitivity and symmetry of hsame give the displayed singleton classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append comm congr`

Theorem 44.24 (CommRing singleton classifier append commutativity compatibility)**.** *For histories* $x, x', y, y' : \mathsf{BHist}$*, if the singleton commutative-ring classifier relates* x *to* x' *and relates* y *to* y'*, then it classifies three append endpoints:*

$$\rho_{\mathsf{e}}(\mathsf{append}(x, y), \mathsf{append}(y, x)), \quad \rho_{\mathsf{e}}(\mathsf{append}(x', y'), \mathsf{append}(y', x')), \quad \rho_{\mathsf{e}}(\mathsf{append}(x, y), \mathsf{append}(x', y')).$$

Proof. The two classifier hypotheses expose singleton carrier witnesses for x, x', y, y'. Empty-result construction for append builds singleton carrier witnesses for each displayed appended endpoint. Since every endpoint in the three classifier rows is classified through e, symmetry and transitivity of hsame package the three singleton classifier witnesses. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append commutativity compatible`

Theorem 44.25 (CommRing singleton classifier append commutation carrier exactness)**.** *For histories* $h, k : \mathsf{BHist}$*, the singleton commutative-ring classifier relates the two appended endpoints with exchanged factors exactly when both factors are singleton carriers:*

$$\rho_{\mathsf{e}}(\mathsf{append}(h, k), \mathsf{append}(k, h)) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction projects the source carrier field from the classifier and applies empty-result inversion to the appended endpoint. Conversely, the two factor carrier witnesses rebuild singleton carrier witnesses for both appended orders by empty-result construction. The rebuilt endpoints are both classified through e, so transitivity and symmetry of hsame give the classifier relation. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append comm carrier iff`

Theorem 44.26 (CommRing singleton classifier append associativity carrier exactness)**.** *For histories* $a, b, c : \mathsf{BHist}$*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(a, b), c), \mathsf{append}(a, \mathsf{append}(b, c))) \iff A_{\mathsf{e}}(a) \wedge A_{\mathsf{e}}(b) \wedge A_{\mathsf{e}}(c).$$

The ten-field fit is: traditional target singleton commutative-ring bracketing exactness; BEDC source BHist *with the singleton* $\mathsf{CommRing}^{\uparrow}$ *tuple; carrier* A_{e}*; classifier* ρ_{e}*; stability from nested append empty-result inversion and singleton classifier packaging; ledger the three append factors and the two bracketings; core theorem this block; exactness the displayed iff; standard bridge one-element commutative-ring bracketing; status* $\mathsf{ProofTarget}$*.*

Proof. In the forward direction, project the source carrier field from the classifier. This gives the carrier witness for $\mathsf{append}(\mathsf{append}(a, b), c)$. Empty-result inversion for append splits it into carrier witnesses for $\mathsf{append}(a, b)$ and c.

Apply empty-result inversion again to the witness for $\mathsf{append}(a, b)$. The resulting two carrier witnesses, together with the witness for c, give $A_{\mathsf{e}}(a) \wedge A_{\mathsf{e}}(b) \wedge A_{\mathsf{e}}(c)$ by conjunction introduction.

Conversely, assume carrier witnesses for a, b, and c. Empty-result construction first builds a carrier witness for $\mathsf{append}(a, b)$, and then one for $\mathsf{append}(\mathsf{append}(a, b), c)$.

The same construction applied to b and c, followed by the witness for a, builds the carrier witness for $\mathsf{append}(a, \mathsf{append}(b, c))$. Since both bracketed endpoints are classified through e, symmetry and transitivity of hsame package the displayed singleton classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier append assoc carrier iff`

Theorem 44.27 (CommRing singleton classifier continuation empty result)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the singleton commutative-ring classifier on the two inputs is equivalent to the continuation result being internally empty:*

$$\mathsf{CommRingSingletonClassifier}(P, Q) \iff \mathsf{hsame}(R, \mathsf{e}).$$

Proof. If the inputs are singleton-classified, both input endpoints are empty, so the append continuation result is empty by append-empty construction. Conversely, an empty continuation result identifies $\mathsf{append}(P, Q)$ with e; append-empty inversion gives the two singleton carrier witnesses, and their chain through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonClassifier continuation empty result iff`

Theorem 44.28 (CommRing singleton continuation carrier split)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the singleton commutative-ring carrier on the continuation result splits exactly over the two input histories:*

$$\mathsf{CommRingSingletonCarrier}(R) \iff \mathsf{CommRingSingletonCarrier}(P) \wedge \mathsf{CommRingSingletonCarrier}(Q).$$

Proof. For the forward implication, read $\mathsf{CommRingSingletonCarrier}(R)$ as $\mathsf{hsame}(R, \mathsf{e})$. Apply the reverse direction of Theorem 44.27 to obtain $\mathsf{CommRingSingletonClassifier}(P, Q)$, and then unpack its two endpoint carrier witnesses.

For the converse implication, assume carrier witnesses for P and Q. These are witnesses of $\mathsf{hsame}(P, \mathsf{e})$ and $\mathsf{hsame}(Q, \mathsf{e})$, so symmetry and transitivity through e package $\mathsf{CommRingSingletonClassifier}(P, Q)$. Apply the forward direction of Theorem 44.27 to get $\mathsf{hsame}(R, \mathsf{e})$, which is exactly $\mathsf{CommRingSingletonCarrier}(R)$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonCarrier continuation split iff`

Theorem 44.29 (Singleton empty-history CommRingUp schema certificate)**.** *Definition 44.6, together with Theorem 44.7, supplies the carrier, source, pattern, classifier, stability, and ledger fields required by Definition 44.59 over the concrete* BHist *singleton source.*

Proof. The carrier and source fields are the singleton empty-history source A_{e} from Definition 44.6. The classifier field is ρ_{e}, and the operation tuple is the constant empty-history tuple for addition, negation, zero, multiplication, and one.

The ring-law fields are supplied by Theorem 43.6 on that same carrier, classifier, and constant operation tuple. The multiplicative commutativity field and its compatibility under ρ_{e} are exactly the additional fields packaged by Theorem 44.7.

The pattern field is the commutative-ring expression pattern generated by those operations and the multiplication-commutativity exchange. The ledger field is the constant-operation provenance tuple for 0, 1, addition, negation, and multiplication, so every operation endpoint recorded by the certificate remains the singleton history e. □

↪variant `BEDC.Derived.CommRingUp.singleton empty history commring laws`

The $\mathsf{CommRing}^{\uparrow}$ schema is therefore represented by this concrete BHist certificate rather than by abstract operator fields.

44.4 The certificate

Definition 44.30 (CommRing source specification)**.** *The source is a certified ring carrier plus multiplicative commutativity.*

Definition 44.31 (CommRing pattern specification)**.** *The pattern is ring expression formation where multiplication may be exchanged by the commutativity field.*

Definition 44.32 (CommRing classifier specification)**.** *The classifier is the ring carrier classifier.*

Definition 44.33 (CommRing stability certificate)**.** *The stability certificate consists of:*

1. *all ring stability fields*
2. *multiplicative commutativity*
3. *compatibility of multiplication commutativity with ring congruence*

Theorem 44.34 (CommRing right distributivity from left distributivity)**.** *For BEDC history addition and multiplication, multiplicative commutativity, additive congruence, and left distributivity imply right distributivity:*

$$(x + y) \cdot z \mathsf{hsame} x \cdot z + y \cdot z.$$

Proof. Commute $(x + y) \cdot z$ to $z \cdot (x + y)$. Apply left distributivity, then use additive congruence with multiplicative commutativity on both summands. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring right distrib from left`

Theorem 44.35 (CommRing left distributivity from right distributivity)**.** *For BEDC history addition and multiplication, multiplicative commutativity, additive congruence, and right distributivity imply left distributivity:*

$$x \cdot (y + z) \mathsf{hsame} x \cdot y + x \cdot z.$$

Proof. Commute $x \cdot (y + z)$ to $(y + z) \cdot x$. Apply right distributivity, then use additive congruence with multiplicative commutativity on both summands. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring left distrib from right`

Theorem 44.36 (CommRing one-sided distributivity equivalence)**.** *For BEDC history operations add and mul, multiplicative commutativity and additive congruence imply that left distributivity is equivalent to right distributivity:*

$$(\forall x\, y\, z : \mathsf{BHist},\ x \cdot (y + z)\mathsf{hsame} x \cdot y + x \cdot z) \leftrightarrow (\forall x\, y\, z : \mathsf{BHist},\ (x + y) \cdot z\mathsf{hsame} x \cdot z + y \cdot z)\,.$$

Proof. For the forward direction, assume the displayed left distributivity law. Applying Theorem 44.34 with the shared multiplicative commutativity and additive congruence fields gives the displayed right distributivity law for all $x, y, z : \mathsf{BHist}$.

For the reverse direction, assume the displayed right distributivity law. Applying Theorem 44.35 with the same commutativity and congruence fields gives the displayed left distributivity law for all $x, y, z : \mathsf{BHist}$. □

Per object, the traditional target is one-sided distributivity collapse in commutative rings; the BEDC source is $\mathsf{CommRing}^{\uparrow}$ over BHist; the carrier is BHist; the classifier is hsame; the stability obligations are multiplicative commutativity, additive congruence, and one chosen distributivity law; the ledger records which side is primitive and which side is derived; the core theorem list records this equivalence; the exactness predicate is the displayed iff; the standard bridge is ordinary commutative-ring distributivity; and the status is $\mathsf{ProofTarget}$.

Theorem 44.37 (CommRing square of a sum expansion)**.** *For BEDC history addition and multiplication, multiplicative commutativity, additive congruence, and both distributive laws imply*

$$(a + b) \cdot (a + b)\mathsf{hsame}(a \cdot a + a \cdot b) + (a \cdot b + b \cdot b).$$

Proof. Apply right distributivity to the outer product. Apply left distributivity to both exposed factors, then use additive congruence and multiplicative commutativity to replace $b \cdot a$ by $a \cdot b$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring square add expand`

Theorem 44.38 (CommRing square of a signed difference)**.** *For BEDC history operations add, mul, and neg satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, both distributive laws, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$(a + (-b)) \cdot (a + (-b))\mathsf{hsame}(a \cdot a + (-(a \cdot b))) + ((-(a \cdot b)) + (b \cdot b)).$$

Proof. Apply Theorem 44.37 with second input $-b$. This gives the ledger expression

$$(a \cdot a + a \cdot (-b)) + (a \cdot (-b) + (-b) \cdot (-b)).$$

Use Theorem 43.49 on each exposed $a \cdot (-b)$ term, and use Theorem 43.51 on $(-b) \cdot (-b)$.

Nested additive congruence transports these three product rewrites through the two additions. Transitivity of hsame composes the square expansion with the transported ledger, yielding the stated signed-difference square form. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring square signed difference expand`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, this theorem is a Core theorem list item for traditional target commutative-ring polynomial algebra. Its BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist, its carrier is BHist, its classifier is hsame, its stability obligations are distributivity, congruence, commutativity, and negative-product laws, and its ledger records polynomial expression provenance with status ProofTarget.

Theorem 44.39 (CommRing parallelogram square sum)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, both distributive laws, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$((a+b)\cdot(a+b)) + ((a+(-b))\cdot(a+(-b)))\mathsf{hsame}(a\cdot a + a\cdot a) + (b\cdot b + b\cdot b).$$

Proof. Use Definition 44.37 for the first square and Definition 44.38 for the signed square. Addition congruence combines the two normalized ledgers into one sum of four additive blocks.

Apply additive associativity, additive commutativity, and Definition 42.13 to place the two $a\cdot b$ terms next to the two $-(a\cdot b)$ terms. The middle ledger then contains the paired mixed terms as inverse pairs.

Collapse those pairs with Definition 43.27 and the additive left-inverse law. The ring zero laws remove the resulting zero summands, leaving exactly the two $a\cdot a$ terms and the two $b\cdot b$ terms in the displayed order. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring parallelogram square sum`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, the traditional target is the commutative-ring parallelogram square-sum identity; the BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; the carrier is BHist; the classifier is hsame; the stability obligations are square expansion, signed-square expansion, additive inverse laws, additive congruence, and middle-four interchange; the ledger records the two square ledgers and mixed-term cancellation; the core theorem list records this block; the exactness predicate is the displayed hsame; the standard bridge is the ordinary parallelogram identity in a commutative ring; and the status is $\mathsf{ProofTarget}$.

Theorem 44.40 (CommRing square cross-term extraction pair)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, both distributive laws, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies both identities*

$$((a+b)\cdot(a+b)) + ((-(a\cdot a)) + (-(b\cdot b)))\mathsf{hsame}(a\cdot b) + (a\cdot b)$$

and

$$((a+(-b))\cdot(a+(-b))) + ((-(a\cdot a)) + (-(b\cdot b)))\mathsf{hsame}(-(a\cdot b)) + (-(a\cdot b)).$$

Proof. For the positive square, apply Definition 44.37 and transport the appended $(-(a\cdot a)) + (-(b\cdot b))$ ledger through addition congruence. Additive associativity, additive commutativity, and Definition 42.13 move the $a\cdot a$ term next to its inverse and the $b\cdot b$ term next to its inverse.

Definition 43.27 cancels the two exposed diagonal pairs. The additive zero laws remove the resulting zero summands, leaving exactly the two positive mixed terms in the displayed order.

For the signed square, apply Definition 44.38 and use addition congruence to append the same diagonal inverse ledger. The same associativity, commutativity, and middle-four interchange rearrangement exposes the $a\cdot a$ and $b\cdot b$ inverse pairs without changing the two signed mixed terms.

Apply Definition 43.27 again, then use the additive zero laws to erase the cancelled diagonal ledger. The remaining terms are $-(a\cdot b)$ and $-(a\cdot b)$, which gives the second displayed identity. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring square cross term extraction pair`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, the traditional target is commutative-ring cross-term extraction for quadratic algebra; the BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; the carrier is BHist; the classifier is hsame; the stability obligations are square expansion, signed-square expansion, additive congruence, additive associativity, additive commutativity, diagonal inverse cancellation, additive zero laws, and middle-four interchange; the ledger records diagonal removal from the two normalized square ledgers; the core theorem is this block; the exactness predicate is the two displayed hsame identities; the downstream use is $\mathsf{Field}^{\uparrow}$ quadratic normalization and later polarization-style algebra; and the status is $\mathsf{ProofTarget}$.

Theorem 44.41 (CommRing product of binary sums)**.** *For BEDC history addition and multiplication, multiplicative commutativity, additive congruence, and left distributivity imply the expansion*

$$(a+b)\cdot(c+d)\mathsf{hsame}(a\cdot c+a\cdot d)+(b\cdot c+b\cdot d).$$

Proof. First derive right distributivity from multiplicative commutativity and left distributivity, then apply it to the outer product. Apply left distributivity to the two inner products $a\cdot(c+d)$ and $b\cdot(c+d)$ and transport both equalities through additive congruence. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring mul add add expand`

Theorem 44.42 (CommRing square of a binary sum)**.** *For BEDC history addition and multiplication satisfying additive associativity, additive congruence, multiplicative commutativity, and left distributivity, every $a, b : \mathsf{BHist}$ satisfies*

$$(a+b)\cdot(a+b)\mathsf{hsame}(a\cdot a+a\cdot b)+(a\cdot b+b\cdot b).$$

Proof. Specialize Theorem 44.41 with $c := a$ and $d := b$. This gives the row-major expansion

$$(a+b)\cdot(a+b)\mathsf{hsame}(a\cdot a+a\cdot b)+(b\cdot a+b\cdot b).$$

Multiplicative commutativity rewrites the exposed middle product $b\cdot a$ to $a\cdot b$. Additive congruence transports this product rewrite through the right inner sum, then through the outer sum.

Transitivity of hsame composes the specialized expansion with the transported right-hand ledger, yielding the duplicate-middle square form. □

Theorem 44.43 (CommRing column expansion of binary sums)**.** *For BEDC history addition and multiplication, additive associativity, additive commutativity, additive congruence, multiplicative commutativity, and left distributivity imply, for every $a, b, c, d : \mathsf{BHist}$,*

$$(a+b)\cdot(c+d)\mathsf{hsame}(a\cdot c+b\cdot c)+(a\cdot d+b\cdot d).$$

Proof. Apply Theorem 44.41 to obtain the row-major expansion of $(a+b)\cdot(c+d)$ as

$$(a\cdot c+a\cdot d)+(b\cdot c+b\cdot d).$$

On the additive operation, apply Theorem 42.13 with the four summands $a\cdot c$, $a\cdot d$, $b\cdot c$, and $b\cdot d$. Additive congruence then composes this interchange with the row-major expansion and yields the column-major normal form. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring mul add add column expand`

Theorem 44.44 (CommRing left distributivity with commuted terms)**.** *For BEDC history addition and multiplication, multiplicative commutativity, additive congruence, and left distributivity imply*

$$x \cdot (y + z)\mathsf{hsame} y \cdot x + z \cdot x.$$

Proof. Apply left distributivity to obtain the two summands $x \cdot y$ and $x \cdot z$. Additive congruence then transports both summands across multiplicative commutativity. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring left distrib commuted terms`

Theorem 44.45 (CommRing difference of squares)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$(a + b) \cdot (a + (-b))\mathsf{hsame}(a \cdot a) + (-(b \cdot b)).$$

Proof. Apply Theorem 44.41 to expand $(a + b) \cdot (a + (-b))$ into the four-term ledger expression $(a \cdot a + a \cdot (-b)) + (b \cdot a + b \cdot (-b))$.

Use Theorem 43.49 to rewrite $a \cdot (-b)$ as $-(a \cdot b)$. Commute $b \cdot a$ to $a \cdot b$ by multiplicative commutativity, and use the same right-negative product theorem to rewrite $b \cdot (-b)$ as $-(b \cdot b)$.

The expression is therefore transported by additive congruence to $(a \cdot a + (-(a \cdot b))) + (a \cdot b + (-(b \cdot b)))$. Additive associativity and commutativity move the middle pair next to each other without changing the outer square terms.

Finally, additive inverse collapse gives $(-(a \cdot b)) + (a \cdot b)\mathsf{hsame}0$. The zero laws remove this neutral summand, leaving $(a \cdot a) + (-(b \cdot b))$ under hsame. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring difference of squares`

Theorem 44.46 (CommRing equal squares force signed factor zero)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$a \cdot a\mathsf{hsame} b \cdot b \Longrightarrow (a + b) \cdot (a + (-b))\mathsf{hsame}0.$$

Proof. Apply Theorem 44.45 to rewrite the signed factor product as $(a \cdot a) + (-(b \cdot b))$. Transport the assumed square equality $a \cdot a\mathsf{hsame} b \cdot b$ through additive congruence while keeping the right summand $-(b \cdot b)$ fixed. The right side is therefore identified with $(b \cdot b) + (-(b \cdot b))$.

Theorem 43.27 collapses this final additive inverse pair to 0. Transitivity of hsame composes the difference-of-squares normal form, the transported square equality, and the inverse collapse. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring equal squares signed factor zero`

Theorem 44.47 (CommRing difference-of-squares zero exactness)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$(a + b) \cdot (a + (-b))\mathsf{hsame}0 \Longleftrightarrow a \cdot a\mathsf{hsame} b \cdot b.$$

Proof. For the forward direction, apply Theorem 44.45 to identify the signed product with $(a \cdot a) + (-(b \cdot b))$. Compose this identification, using symmetry where needed, with the zero-product hypothesis. The difference term is therefore classified as 0.

Add $b \cdot b$ on the right by additive congruence. On the left endpoint, additive associativity and commutativity move the inverse pair into the form $(b \cdot b) + (-(b \cdot b))$; Theorem 43.27 collapses that pair to 0, and the zero laws leave $a \cdot a$. On the right endpoint, the zero law collapses $0 + (b \cdot b)$ to $b \cdot b$. Transitivity gives $a \cdot a \mathsf{hsame} b \cdot b$.

For the reverse direction, apply Theorem 44.46 to the square equality $a \cdot a \mathsf{hsame} b \cdot b$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring difference of squares zero exact`

Theorem 44.48 (CommRing signed binary product normal form)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *satisfies*

$$(a + (-b)) \cdot (c + (-d)) \mathsf{hsame} (a \cdot c + (-(a \cdot d))) + ((-(b \cdot c)) + (b \cdot d)).$$

Proof. Apply Theorem 44.41 with the four inputs a, $-b$, c, and $-d$. The left side expands to

$$(a \cdot c + a \cdot (-d)) + ((-b) \cdot c + (-b) \cdot (-d)).$$

Next normalize the signed products. Theorem 43.49 rewrites $a \cdot (-d)$ as $-(a \cdot d)$, Theorem 43.48 rewrites $(-b) \cdot c$ as $-(b \cdot c)$, and Theorem 43.51 rewrites $(-b) \cdot (-d)$ as $b \cdot d$.

Use nested additive congruence to transport the three rewritten summands through the outer two additions. Transitivity of hsame composes the expansion step with this transport, giving the stated four-term normal form. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed binary product normal form`

Theorem 44.49 (CommRing signed product balance)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *satisfies*

$$((a + (-b)) \cdot (c + (-d))) + ((a \cdot d) + (b \cdot c)) \mathsf{hsame} (a \cdot c) + (b \cdot d).$$

Proof. Apply Theorem 44.48 to replace the signed product by the four-term row expansion

$$(a \cdot c + (-(a \cdot d))) + ((-(b \cdot c)) + (b \cdot d)).$$

Transport the added mixed-term sum through this identification by additive congruence. The expression is then an additive composite containing the two ordinary mixed terms $a \cdot d$ and $b \cdot c$ together with their signed counterparts $-(a \cdot d)$ and $-(b \cdot c)$.

Use additive associativity and commutativity, with Theorem 42.13 for the middle summands, to place $-(a \cdot d)$ adjacent to $a \cdot d$ and $-(b \cdot c)$ adjacent to $b \cdot c$. The unmixed terms $a \cdot c$ and $b \cdot d$ remain as the external summands.

Theorem 43.27 collapses the right-inverse mixed pair, and the additive left-inverse law collapses the left-inverse mixed pair. The ring zero laws remove the two neutral summands, leaving $(a \cdot c) + (b \cdot d)$ under hsame. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed product balance`

The ten-field fit is: traditional target bilinear subtraction balance; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations distributivity, congruence, commutativity, negative-product laws, and additive cancellation; ledger polynomial mixed-term provenance; core theorem this balance identity; exactness the displayed hsame; standard bridge commutative-ring bilinear expansion; status $\mathsf{ProofTarget}$.

Theorem 44.50 (CommRing signed column expansion of binary sums)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *satisfies*

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}(a \cdot c + (-(b \cdot c))) + ((-(a \cdot d)) + (b \cdot d)).$$

Proof. Apply Theorem 44.48 to obtain the row-major signed expansion

$$(a \cdot c + (-(a \cdot d))) + ((-(b \cdot c)) + (b \cdot d)).$$

On the additive operation, apply Theorem 42.13 with the four summands $a \cdot c$, $-(a \cdot d)$, $-(b \cdot c)$, and $b \cdot d$. Additive congruence transports the middle-four interchange through the displayed signed row form.

Transitivity of hsame composes the row-major normalization with this column reordering, yielding the stated column-major signed normal form. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed column binary product normal form`

Theorem 44.51 (CommRing signed column cancellations force zero product)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *with* $a \cdot c\mathsf{hsame}b \cdot c$ *and* $a \cdot d\mathsf{hsame}b \cdot d$ *satisfies*

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0.$$

Proof. Apply Theorem 44.50 to rewrite the signed product as

$$(a \cdot c + (-(b \cdot c))) + ((-(a \cdot d)) + (b \cdot d)).$$

The first hypothesis is transported by additive congruence through the left signed column. It identifies $a \cdot c + (-(b \cdot c))$ with $(b \cdot c) + (-(b \cdot c))$, and Theorem 43.27 collapses this right-inverse pair to 0.

For the second signed column, apply additive inverse congruence to $a \cdot d\mathsf{hsame}b \cdot d$ and identify $-(a \cdot d)$ with $-(b \cdot d)$. Additive congruence transports $(-(a \cdot d)) + (b \cdot d)$ to $(-(b \cdot d)) + (b \cdot d)$, and the additive left-inverse law collapses this left-inverse pair to 0.

The column normal form is therefore transported to $0 + 0$. The additive left-zero law collapses $0 + 0$ to 0, and transitivity of hsame composes the normal-form step, the two signed-column collapses, and the final zero collapse. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed column cancellations zero`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, the traditional target is signed column cancellation in commutative-ring algebra. Its BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist, its carrier is BHist, its classifier is hsame, its stability obligations are signed-product normalization, additive

inverse laws, additive congruence, and transitivity, and its ledger records column-paired product provenance. The core theorem list contains Theorem 44.51; exactness is zero classification under column-paired cancellations; the standard bridge remains outside this proof site; and the status is ProofTarget.

Theorem 44.52 (CommRing signed row cancellations force zero product)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *with* $a \cdot c\mathsf{hsame}a \cdot d$ *and* $b \cdot c\mathsf{hsame}b \cdot d$ *satisfies*

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0.$$

Proof. Apply Theorem 44.48 to rewrite the signed product as

$$(a \cdot c + (-(a \cdot d))) + ((-(b \cdot c)) + (b \cdot d)).$$

The first row equality is transported by additive congruence through the left signed row. It identifies $a \cdot c + (-(a \cdot d))$ with $(a \cdot d) + (-(a \cdot d))$, and Theorem 43.27 collapses this right-inverse pair to 0.

For the second signed row, apply additive inverse congruence to $b \cdot c\mathsf{hsame}b \cdot d$ and identify $-(b \cdot c)$ with $-(b \cdot d)$. Additive congruence transports $(-(b \cdot c)) + (b \cdot d)$ to $(-(b \cdot d)) + (b \cdot d)$, and the additive left-inverse law collapses this left-inverse pair to 0.

The row normal form is therefore transported to $0 + 0$. The additive left-zero law collapses $0 + 0$ to 0, and transitivity of hsame composes the normal-form step, the two signed-row collapses, and the final zero collapse. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed row cancellations zero`

The ten-field fit is: traditional target signed row cancellation in commutative-ring algebra; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability additive abelian-group laws, congruence, distributivity, commutativity, and negative-product laws; ledger signed row product provenance; core theorem this row-cancellation result; exactness row-zero soundness; standard bridge the ordinary commutative-ring identity; status ProofTarget.

Theorem 44.53 (CommRing signed product zero exactness)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *satisfies*

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0 \Longleftrightarrow a \cdot c + b \cdot d\mathsf{hsame}a \cdot d + b \cdot c.$$

Proof. For the forward direction, apply Theorem 44.48 to rewrite the signed product as

$$(a \cdot c + (-(a \cdot d))) + ((-(b \cdot c)) + (b \cdot d)).$$

Apply Theorem 42.13 to regroup the four-term additive ledger as $(a \cdot c + b \cdot d) + ((-(a \cdot d)) + (-(b \cdot c)))$. The negative-product laws and additive inverse congruence identify the second summand with the additive inverse of $(a \cdot d + b \cdot c)$. The signed product hypothesis therefore presents a balanced difference between the two cross sums.

Add $a \cdot d + b \cdot c$ to both sides by additive congruence. Additive associativity moves the inverse pair next to its positive partner, and Theorem 43.27 collapses the pair to 0. The zero laws remove the neutral summand and leave $a \cdot c + b \cdot d\mathsf{hsame}a \cdot d + b \cdot c$.

For the reverse direction, start from the assumed cross-sum equality and transport it through additive congruence while keeping the inverse of the right cross sum fixed. The right endpoint becomes the sum of $a \cdot d + b \cdot c$ and its additive inverse.

Theorem 43.27 collapses that endpoint to 0. Symmetry and transitivity compose this collapse with the signed binary normal form, yielding $(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed product empty zero exact`

Theorem 44.54 (CommRing signed square zero exactness)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *satisfies*

$$(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0 \Longleftrightarrow a \cdot a + b \cdot b\mathsf{hsame}a \cdot b + a \cdot b.$$

Proof. Apply Theorem 44.53 with $c = a$ and $d = b$. The signed product side becomes the signed square, and the cross-sum criterion becomes

$$a \cdot a + b \cdot b\mathsf{hsame}a \cdot b + b \cdot a.$$

For the forward direction, multiplicative commutativity gives $b \cdot a\mathsf{hsame}a \cdot b$. Additive congruence transports the right summand of $a \cdot b + b \cdot a$ along this equality, so the cross sum obtained from the specialized exactness theorem is identified with $a \cdot b + a \cdot b$.

For the reverse direction, use symmetry of the same commutativity instance and additive congruence to transport $a \cdot b + a \cdot b$ back to $a \cdot b + b \cdot a$. The reverse implication of the specialized signed-product exactness theorem then returns $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed square empty zero exact`
Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed square zero exact`

Theorem 44.55 (CommRing signed square zero diagonal extraction package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b : \mathsf{BHist}$ *with* $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$ *satisfies both*

$$a \cdot a + b \cdot b\mathsf{hsame}a \cdot b + a \cdot b$$

and

$$(-(a \cdot a)) + (-(b \cdot b))\mathsf{hsame}(-(a \cdot b)) + (-(a \cdot b)).$$

Proof. The first displayed identity is the forward implication of Theorem 44.54 applied to the assumed zero signed square.

For the diagonal inverse ledger, use the signed half of Theorem 44.40. Additive congruence transports the signed-square factor across $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$, so its left side becomes the same diagonal inverse ledger with a zero summand.

The ring zero law removes that zero summand. Symmetry and transitivity of hsame compose the transport with the signed cross-term endpoint, leaving exactly

$$(-(a \cdot a)) + (-(b \cdot b))\mathsf{hsame}(-(a \cdot b)) + (-(a \cdot b)).$$

□

Lean4 ✓ `BEDC.Derived.CommRingUp.commring signed square zero diagonal extraction package`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, the traditional target is diagonal extraction from a zero signed square; the BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; the carrier is BHist; the classifier is hsame; the stability obligations are signed-square exactness, cross-term extraction, additive congruence, and zero laws; the ledger records the zero signed square and diagonal inverse ledger; the downstream consumer is $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; and the status is $\mathsf{ProofTarget}$.

Theorem 44.56 (CommRing column cancellations balance cross sums)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *with* $a \cdot c\mathsf{hsame}b \cdot c$ *and* $a \cdot d\mathsf{hsame}b \cdot d$ *satisfies*

$$a \cdot c + b \cdot d\mathsf{hsame}a \cdot d + b \cdot c.$$

Proof. Apply Theorem 44.51 to the two column hypotheses. This produces the signed zero product

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0.$$

Apply the forward direction of Theorem 44.53. The signed zero product is exactly the left side of that exactness statement, so its conclusion is the displayed cross-sum balance. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring column cancellations cross sum balance`

Theorem 44.57 (CommRing row cancellations balance cross sums)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, multiplicative commutativity, and the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, every* $a, b, c, d : \mathsf{BHist}$ *with* $a \cdot c\mathsf{hsame}a \cdot d$ *and* $b \cdot c\mathsf{hsame}b \cdot d$ *satisfies*

$$a \cdot c + b \cdot d\mathsf{hsame}a \cdot d + b \cdot c.$$

Proof. Apply Theorem 44.52 to the two row hypotheses. This produces the signed zero product

$$(a + (-b)) \cdot (c + (-d))\mathsf{hsame}0.$$

Apply the forward direction of Theorem 44.53. The signed zero product is exactly the left side of that exactness statement, so its conclusion is the displayed cross-sum balance. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring row cancellations cross sum balance`

For the $\mathsf{CommRing}^{\uparrow}$ per-object template, the traditional target is row cancellation balance in commutative-ring algebra. Its BEDC source is $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist, its carrier is BHist, its classifier is hsame, its stability obligations are signed row cancellation and signed-product zero exactness, and its ledger records row-paired product provenance. The core theorem is Theorem 44.57; exactness is cross-sum balance; the standard bridge is ordinary commutative-ring cancellation balance; and the status is $\mathsf{ProofTarget}$.

Definition 44.58 (CommRing ledger policy)**.** *Record ring expression provenance and the multiplicative commutativity law when used.*

Definition 44.59 ($\mathsf{CommRing}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{CommRing}^{\uparrow}$*.*

Primitive boundary. $\mathsf{CommRing}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

44.5 CommRing quadratic polarization package

Theorem 44.60 (CommRing quadratic polarization package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the shared hypotheses of Definition 44.39, Definition 44.40, and Definition 44.54, every $a, b : \mathsf{BHist}$ carries the concrete quadratic-polarization package*

$$\Big(((a{+}b){\cdot}(a{+}b)){+}((a{+}(-b)){\cdot}(a{+}(-b)))\mathsf{hsame}(a{\cdot}a{+}a{\cdot}a){+}(b{\cdot}b{+}b{\cdot}b)\Big)\wedge\Big(((a{+}b){\cdot}(a{+}b)){+}((-(a{\cdot}a)){+}(-(b{\cdot}b)))\mathsf{hsame}(a{\cdot}b$$

Proof. Apply Definition 44.39 to the shared hypotheses and the selected histories a, b. This gives the parallelogram square-sum identity. Apply Definition 44.40 to the same data; its two components give the positive and signed cross-term extraction identities.

Apply Definition 44.54 to the same histories. This gives the signed-square zero exactness iff. Introduce conjunctions in the displayed order, using the parallelogram identity first, the two extraction identities next, and the exactness iff as the final component. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring quadratic polarization package`

Theorem 44.61 (CommRing quadratic zero-polarization exit package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the shared hypotheses of Definition 44.60 and Definition 44.55, every $a, b : \mathsf{BHist}$ with $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$ carries the concrete exit package*

$$\Big(((a{+}b){\cdot}(a{+}b)){+}((a{+}(-b)){\cdot}(a{+}(-b)))\mathsf{hsame}(a{\cdot}a{+}a{\cdot}a){+}(b{\cdot}b{+}b{\cdot}b)\Big)\wedge\Big(((a{+}b){\cdot}(a{+}b)){+}((-(a{\cdot}a)){+}(-(b{\cdot}b)))\mathsf{hsame}(a{\cdot}b$$

Lean4 ✓ `BEDC.Derived.CommRingUp.commring quadratic zero polarization exit package`

Proof. Apply Definition 44.60 to the shared hypotheses and the selected histories a, b. This gives the parallelogram identity, the two cross-term extraction identities, and the signed-square zero exactness component.

Apply Definition 44.55 to the same data together with $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$. Its two components give the positive diagonal equality and the diagonal inverse extraction. Introduce the conjunction in the same order as the statement: the four unconditional polarization components first, followed by the two diagonal equalities supplied by the signed-square zero diagonal-extraction package. □

Theorem 44.62 (CommRing quadratic positive-diagonal exit package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the shared hypotheses of Definition 44.60 and Definition 44.55, every $a, b : \mathsf{BHist}$ with $a \cdot a + b \cdot b\mathsf{hsame}a \cdot b + a \cdot b$ carries the concrete positive-diagonal exit package*

$$\Big(((a{+}b){\cdot}(a{+}b)){+}((a{+}(-b)){\cdot}(a{+}(-b)))\mathsf{hsame}(a{\cdot}a{+}a{\cdot}a){+}(b{\cdot}b{+}b{\cdot}b)\Big)\wedge\Big(((a{+}b){\cdot}(a{+}b)){+}((-(a{\cdot}a)){+}(-(b{\cdot}b)))\mathsf{hsame}(a{\cdot}b$$

Proof. Apply Definition 44.60 to the shared hypotheses and the selected histories a, b. This gives the four quadratic-polarization components, including the signed-square zero exactness iff.

Use the reverse direction of that iff on the assumed positive diagonal equality. This recovers the signed-square zero witness $(a + (-b)) \cdot (a + (-b))\mathsf{hsame}0$.

Apply Definition 44.55 to the same data and the recovered signed-square zero witness. Its second component gives the inverse diagonal ledger. Introduce conjunctions in the displayed order: the four polarization components first, the recovered signed-square zero witness next, and the inverse diagonal ledger last. □

The positive-diagonal ten-field fit is: traditional target positive-diagonal quadratic exit; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from Definition 44.60 and Definition 44.55; ledger the positive diagonal witness and recovered signed-square zero witness; core theorem this block; exactness the displayed package; downstream use $\mathsf{Field}^{\uparrow}$ quadratic normalization and equal-square/zero-product reasoning; status $\mathsf{ProofTarget}$.

The signed-square-zero ten-field fit is: traditional target a commutative-ring quadratic polarization interface; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from the shared distributivity, congruence, commutativity, and negative-product laws; ledger the parallelogram square-sum, two cross-term extractions, and signed-square zero exactness witnesses together with the positive and inverse diagonal equalities; core theorem this block; exactness the displayed conjunction; standard bridge ordinary quadratic polarization in a commutative ring; downstream use $\mathsf{Field}^{\uparrow}$ quadratic normalization and equal-square/zero-product reasoning; status $\mathsf{ProofTarget}$.

Theorem 44.63 (CommRing quadratic zero-polarization annihilator package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the joint hypotheses of Definition 44.61 and Definition 43.39, every $a, b : \mathsf{BHist}$ with $((a + (-b)) \cdot (a + (-b)))\mathsf{hsame}0$ carries the concrete annihilator package*

$$\Big(((a{+}b){\cdot}(a{+}b)){+}((a{+}(-b)){\cdot}(a{+}(-b)))\mathsf{hsame}(a{\cdot}a{+}a{\cdot}a){+}(b{\cdot}b{+}b{\cdot}b)\Big)\wedge\Big(((a{+}b){\cdot}(a{+}b)){+}((-(a{\cdot}a)){+}(-(b{\cdot}b)))\mathsf{hsame}(a{\cdot}b$$

The ten-field fit is: traditional target commutative-ring quadratic zero-polarization annihilator interface; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from the zero-square witness, multiplication congruence, and zero multiplication absorption; ledger the six exit rows and the universal two-sided annihilator row; core theorem this theorem; exactness the displayed seven-row package; standard bridge ordinary nil-square annihilation in a commutative ring; downstream use $\mathsf{Field}^{\uparrow}$ quadratic normalization and nil-square handling; status $\mathsf{ProofTarget}$.

Proof. Apply Definition 44.61 to the selected histories a, b and the displayed zero-square hypothesis. This supplies the six exit-package rows: the parallelogram identity, the positive and signed cross-term extractions, signed-square zero exactness, the positive diagonal equality, and the inverse diagonal equality.

Now fix an arbitrary multiplier $c : \mathsf{BHist}$. Transport the zero-square witness through multiplication on the left and on the right, and apply Definition 43.39 to the resulting zero products. This gives

$$((a + (-b)) \cdot (a + (-b))) \cdot c\mathsf{hsame}0 \wedge c \cdot ((a + (-b)) \cdot (a + (-b)))\mathsf{hsame}0.$$

Finally conjoin the six rows obtained from the zero-polarization exit package with the two absorption rows obtained for the arbitrary multiplier c. Since c was arbitrary, these rows give the universal two-sided annihilator clause, and the displayed package follows in the stated order. □

Theorem 44.64 (CommRing quadratic positive-diagonal annihilator package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the shared hypotheses of Definition 44.62, Definition 44.63, and Definition 43.39, every $a, b : \mathsf{BHist}$ with $a \cdot a + b \cdot b\mathsf{hsame}a \cdot b + a \cdot b$ carries the*

concrete positive-diagonal annihilator package

$$\Big(((a{+}b){\cdot}(a{+}b)){+}((a{+}(-b)){\cdot}(a{+}(-b)))\mathsf{hsame}(a{\cdot}a{+}a{\cdot}a){+}(b{\cdot}b{+}b{\cdot}b)\Big)\wedge\Big(((a{+}b){\cdot}(a{+}b)){+}((-(a{\cdot}a)){+}(-(b{\cdot}b)))\mathsf{hsame}(a{\cdot}b$$

The ten-field fit is: traditional target positive-diagonal quadratic annihilator interface; BEDC source $\mathsf{Ring}^{\uparrow}$ *plus* $\mathsf{CommRing}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability from the positive diagonal exit package, zero-polarization annihilator package, and zero multiplication absorption; ledger the positive diagonal witness, recovered signed-square zero witness, inverse diagonal row, and universal two-sided annihilator row; core theorem this theorem; exactness the displayed eight-row package; standard bridge ordinary positive-diagonal quadratic annihilation in a commutative ring; downstream use* $\mathsf{Field}^{\uparrow}$ *quadratic normalization and zero-factor exit packages; status* $\mathsf{ProofTarget}$.

Proof. Apply Definition 44.62 to the selected histories a, b and the displayed positive diagonal equality. This supplies the four polarization rows, the recovered signed-square zero witness, and the inverse diagonal ledger row.

Apply Definition 44.63 to the same histories and the recovered signed-square zero witness. Its package contains the positive diagonal row and the universal two-sided annihilator row.

Conjoin the rows in the displayed order: the four polarization rows first, the recovered signed-square zero row next, then the positive diagonal row, the inverse diagonal ledger row, and finally the universal two-sided annihilator row. □

This consequence block is read over the $\mathsf{CommRing}^{\uparrow}$ carrier of the core child and consumes the $\mathsf{Ring}^{\uparrow}$ additive and distributive rows from chapter 43. Its zero-factor and equal-squares consequences are not standalone algebraic assumptions.

Theorem 44.65 (CommRing zero linear factor annihilator branch)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$((a{+}b)\mathsf{hsame0}\vee(a{+}(-b))\mathsf{hsame0}) \Longrightarrow ((a{+}b)\mathsf{hsame0}\wedge\forall c : \mathsf{BHist},\ ((a{+}b){\cdot}c\mathsf{hsame0}\wedge c{\cdot}(a{+}b)\mathsf{hsame0}))\vee((a{+}(-b))\mathsf{hsa}$$

Proof. First derive the right distributive law from the supplied left distributive law by Definition 44.34. The zero-classifier factor absorption theorem is then available with both distributive inputs.

If the displayed disjunction chooses $(a+b)\mathsf{hsame0}$, retain that branch witness. For an arbitrary $c : \mathsf{BHist}$, apply Definition 43.40 with $x = a+b$ and $y = c$. Its two components give $(a+b)\cdot c\mathsf{hsame0}$ and $c\cdot(a+b)\mathsf{hsame0}$, so the left disjunct packages the original witness with the universal annihilator row.

If the displayed disjunction chooses $(a + (-b))\mathsf{hsame0}$, retain that branch witness instead. For an arbitrary $c : \mathsf{BHist}$, apply Definition 43.40 with $x = a + (-b)$ and $y = c$. The two components give $(a + (-b)) \cdot c\mathsf{hsame0}$ and $c \cdot (a + (-b))\mathsf{hsame0}$, yielding the right disjunct with the same branch-preserving form. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor annihilator branch`

Theorem 44.66 (CommRing zero linear factor signed product zero)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and*

multiplication congruence, left distributivity, the $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0\big) \Longrightarrow (a+b)\cdot(a+(-b))\mathsf{hsame}0.$$

Proof. Apply Definition 44.65 to the chosen zero linear factor. The result keeps the selected branch and supplies its two-sided annihilator row for every multiplier.

In the $(a+b)\mathsf{hsame}0$ branch, specialize the universal row at $c = a+(-b)$ and take its left component. This is exactly $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

In the $(a+(-b))\mathsf{hsame}0$ branch, specialize the universal row at $c = a+b$ and take its right component. The oriented product is $(a+b)\cdot(a+(-b))$, giving the required zero classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor signed product zero`

Theorem 44.67 (CommRing zero linear factor commuted signed product zero)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0\big) \Longrightarrow (a+(-b))\cdot(a+b)\mathsf{hsame}0.$$

Proof. Apply Definition 44.65 to the selected zero linear factor. In the $(a+b)\mathsf{hsame}0$ branch, specialize the two-sided annihilator row at $c = a+(-b)$ and take the right multiplication component. In the $(a+(-b))\mathsf{hsame}0$ branch, specialize at $c = a+b$ and take the left multiplication component. Both cases produce the displayed commuted signed product as an empty classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor commuted signed product zero`

Theorem 44.68 (CommRing scoped multiplication commutativity source)**.** *In the concrete* $\mathsf{CommRing}^{\uparrow}$ *zero-linear-factor surface over* BHist*, the multiplicative commutativity input is read only as the* hsame*-stable* BHist *row that reverses the* $\mathsf{Ring}^{\uparrow}$ *multiplication classifier. The zero-linear signed-product consumers may use that row to exchange* $(a+b)\cdot(a+(-b))$ *with* $(a+(-b))\cdot(a+b)$*, but they do not receive an additional host commutative-ring carrier, a quotient classifier, or an inverse operation.*

Proof. Project the $\mathsf{CommRing}^{\uparrow}$ input named in Definition 44.47. Its commutativity field is a classifier row over the same BHist multiplication used by $\mathsf{Ring}^{\uparrow}$, and the surrounding hypotheses already supply the hsame transport needed to keep that row stable.

Apply Definition 44.67. The only new read beyond the $\mathsf{Ring}^{\uparrow}$ zero-linear proof is the displayed reversal of the multiplication classifier. Since the theorem still concludes inside hsame over BHist, the source package contributes no external carrier or field structure. □

Theorem 44.69 (CommRing scoped RingUp dependency source)**.** *Every zero-linear and quadratic consumer in this* $\mathsf{CommRing}^{\uparrow}$ *chapter factors through the already cited* $\mathsf{Ring}^{\uparrow}$ *additive laws, addition and multiplication congruence rows, distributivity rows, and negative-product laws, together with the single scoped commutativity row of Definition 44.68. No consumer in the zero-linear or quadratic packet reads a primitive host commutative ring operation outside those rows.*

Proof. For zero-linear consumers, use Definition 44.65, Definition 44.66, and Definition 44.67. Their hypotheses list exactly the $\mathsf{Ring}^{\uparrow}$ additive, congruence, distributive, and negative-product rows, with the commutativity field used only for the reversed orientation.

For quadratic consumers, use Definition 44.60, Definition 44.73, and Definition 44.75. These package the same zero-linear source rows with the quadratic readback rows, so the quadratic exit has the same scoped dependency surface. □

Theorem 44.70 (CommRing scoped ledger source for FieldUp exits)**.** *The downstream* $\mathsf{Field}^{\uparrow}$ *exit rows referenced by the* $\mathsf{CommRing}^{\uparrow}$ *zero-linear and quadratic packets read only the* Cont *and* Pkg *ledger rows named by those packets: the* $\mathsf{Ring}^{\uparrow}$ *source ledgers, the* $\mathsf{CommRing}^{\uparrow}$ *multiplication-commutativity ledger, the zero-linear annihilator ledgers, and the quadratic readback ledgers. The exit package does not acquire field inverse data, cancellation outside the displayed classifier rows, or a bridge to a standard algebraic carrier.*

Proof. Project the zero-linear ledger from Definition 44.71 and the annihilator packages that follow it. These rows give the Cont witnesses for both signed-product orientations and the Pkg provenance for the $\mathsf{Ring}^{\uparrow}$ and $\mathsf{CommRing}^{\uparrow}$ sources.

Project the quadratic ledger from Definition 44.86 and Definition 44.87. The downstream consumer receives those named rows as its readback surface, so its exit data is confined to the packet ledgers and cannot read inverse, cancellation, or standard-carrier structure absent from them. □

Theorem 44.71 (CommRing zero linear factor two-sided signed product zero)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a+b)\mathsf{hsame0} \vee (a+(-b))\mathsf{hsame0}\big) \Longrightarrow \big((a+b)\cdot(a+(-b))\mathsf{hsame0}\big) \wedge \big((a+(-b))\cdot(a+b)\mathsf{hsame0}\big).$$

Proof. Apply Definition 44.66 to the chosen zero linear factor. This gives the forward signed product $(a+b)\cdot(a+(-b))\mathsf{hsame0}$.

Apply Definition 44.67 to the same branch witness. This gives the commuted signed product $(a+(-b))\cdot(a+b)\mathsf{hsame0}$.

Introduce the conjunction with these two oriented zero-product witnesses as its left and right components. □

Theorem 44.72 (CommRing zero linear factor signed product annihilator package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the same hypotheses as Definition 44.82, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a{+}b)\mathsf{hsame0}{\vee}(a{+}(-b))\mathsf{hsame0}\big) \Longrightarrow ((a+b)\cdot(a+(-b))\mathsf{hsame0} \wedge \forall c : \mathsf{BHist},\ (((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame0} \wedge c$$

Proof. Let H be the displayed zero-linear-factor hypothesis. Apply Definition 44.66 to H to obtain the signed product-zero witness $(a+b)\cdot(a+(-b))\mathsf{hsame0}$.

Use Definition 44.34 to recover the right distributive row from the supplied left distributive row. Together with the $\mathsf{Ring}^{\uparrow}$ zero-factor absorption theorem, this gives the same two-sided absorption mechanism used in Definition 44.65.

Now instantiate that absorption mechanism with $x = (a+b)\cdot(a+(-b))$. For an arbitrary $c : \mathsf{BHist}$, the product-zero witness gives both $((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame0}$ and $c\cdot((a+b)\cdot(a+(-b)))\mathsf{hsame0}$. Pair this universal row with the original signed product-zero witness to obtain the displayed package. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor signed product annihilator package`

Theorem 44.73 (CommRing equal-squares signed-product annihilator)**.** *For BEDC history operations add, mul, neg, and zero satisfying the hypotheses of Definition 44.47 and the zero-factor absorption hypotheses of Definition 43.40, every $a, b : \mathsf{BHist}$ satisfies*

$$a \cdot a\mathsf{hsame}b \cdot b \Longrightarrow ((a+b)\cdot(a+(-b))\mathsf{hsame}0 \wedge \forall c : \mathsf{BHist},\ (((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame}0 \wedge c\cdot((a+b)\cdot(a+(-b))$$

Proof. Let Q be the square-equality hypothesis $a \cdot a\mathsf{hsame}b \cdot b$. Apply the reverse direction of Definition 44.47 to Q. This produces the signed product-zero witness $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

The zero-factor absorption hypotheses give Definition 43.40 for the signed product $x = (a+b)\cdot(a+(-b))$. For an arbitrary $c : \mathsf{BHist}$, instantiate that theorem at $y = c$.

The two components are $((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame}0$ and $c\cdot((a+b)\cdot(a+(-b)))\mathsf{hsame}0$. Pair this universal two-sided absorption row with the signed product-zero witness from the equal-squares route. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring equal squares signed product annihilator`

Theorem 44.74 (CommRing equal-squares bilateral signed-product zero)**.** *For BEDC history operations add, mul, neg, and zero satisfying the hypotheses of Definition 44.47, every $a, b : \mathsf{BHist}$ satisfies*

$$a \cdot a\mathsf{hsame}b \cdot b \Longrightarrow \big((a+b)\cdot(a+(-b))\mathsf{hsame}0\big) \wedge \big((a+(-b))\cdot(a+b)\mathsf{hsame}0\big).$$

Proof. Let Q be the square-equality hypothesis $a \cdot a\mathsf{hsame}b \cdot b$. Apply Definition 44.46 to Q to obtain the forward signed-product zero witness $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

The $\mathsf{CommRing}^{\uparrow}$ multiplicative commutativity field identifies the reversed product $(a+(-b))\cdot(a+b)$ with the forward product $(a+b)\cdot(a+(-b))$. Compose this identification with the forward signed-product zero witness by transitivity of hsame.

The two oriented zero-product witnesses form the required conjunction. □

Theorem 44.75 (CommRing equal-squares commuted signed factor zero)**.** *Under the hypotheses of Definition 44.46, every square equality also annihilates the commuted signed product:*

$$a \cdot a\mathsf{hsame}b \cdot b \Longrightarrow (a+(-b))\cdot(a+b)\mathsf{hsame}0.$$

Proof. Apply multiplicative commutativity to identify $(a+(-b))\cdot(a+b)$ with $(a+b)\cdot(a+(-b))$, then compose with Definition 44.46. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring equal squares commuted signed factor zero`

Theorem 44.76 (CommRing equal-squares commuted signed product annihilator)**.** *Under the hypotheses of Definition 44.75, every square equality gives the commuted signed product as a two-sided annihilator:*

$$a\cdot a\mathsf{hsame}b\cdot b \Longrightarrow (a+(-b))\cdot(a+b)\mathsf{hsame}0 \wedge \forall c, ((a+(-b))\cdot(a+b))\cdot c\mathsf{hsame}0 \wedge c\cdot((a+(-b))\cdot(a+b))\mathsf{hsame}0.$$

Proof. Apply Definition 44.75 to get the commuted signed-product zero witness. Right distributivity follows from left distributivity and multiplicative commutativity, and the ring zero absorption rows then identify multiplication by the zero product on either side with zero. Multiplication congruence transports the zero-product witness through the two multiplier positions. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring equal squares commuted signed product annihilator`

Theorem 44.77 (CommRing equal-squares bilateral signed-product annihilator package)**.** *Under the joint hypotheses of Definition 44.73 and Definition 44.76, every* $a, b : \mathsf{BHist}$ *satisfies*

$$a{\cdot}a\mathsf{hsame}b{\cdot}b \Longrightarrow (((a+b)\cdot(a+(-b))\mathsf{hsame0} \wedge \forall c : \mathsf{BHist},\ (((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame0} \wedge c\cdot((a+b)\cdot(a+(-b)$$

Proof. Let Q be the square-equality hypothesis $a \cdot a\mathsf{hsame}b \cdot b$.

Apply Definition 44.73 to Q. This gives the forward signed-product zero row together with its universal two-sided multiplier annihilator row.

Apply Definition 44.76 to the same Q. This gives the commuted signed-product zero row together with its universal two-sided multiplier annihilator row.

Introduce the outer conjunction and place the forward package in the left component and the commuted package in the right component. □

Theorem 44.78 (CommRing equal-squares bilateral signed-product context annihilator)**.** *Under the joint hypotheses of Definition 44.77 and Definition 43.39, every* $a, b : \mathsf{BHist}$ *satisfies*

$$a{\cdot}a\mathsf{hsame}b{\cdot}b \Longrightarrow \forall c, d : \mathsf{BHist},\ (((c\cdot((a+b)\cdot(a+(-b))))\cdot d)\mathsf{hsame0} \wedge ((c\cdot((a+(-b))\cdot(a+b)))\cdot d)\mathsf{hsame0})\,.$$

Proof. Let Q be the square-equality hypothesis $a \cdot a\mathsf{hsame}b \cdot b$.

Apply Definition 44.77 to Q. Evaluating the two universal left-multiplier rows at c gives $c \cdot ((a+b)\cdot(a+(-b)))\mathsf{hsame0}$ and $c \cdot ((a+(-b))\cdot(a+b))\mathsf{hsame0}$.

For the forward signed product, multiplication congruence transports the first zero witness through the common right multiplier d, comparing $(c \cdot ((a+b)\cdot(a+(-b))))\cdot d$ with $0 \cdot d$. The corresponding zero-absorption row from Definition 43.39 collapses $0 \cdot d$ to 0.

For the commuted signed product, the same argument transports the second left-multiplier zero witness through d and collapses the endpoint $0 \cdot d$ by zero absorption. These two transported zero witnesses form the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring equal squares bilateral signed product context annihilator`

Theorem 44.79 (CommRing equal-squares bilateral signed-product two-sided context annihilator)**.** *Under the joint hypotheses of Definition 44.77 and Definition 43.39, every* $a, b, c, d : \mathsf{BHist}$ *satisfies the following implication. Writing* $P = (a+b)\cdot(a+(-b))$ *and* $Q = (a+(-b))\cdot(a+b)$*,*

$$a{\cdot}a\mathsf{hsame}b{\cdot}b \Longrightarrow \left(\begin{array}{l} (P\cdot c)\cdot d\mathsf{hsame0} \wedge d\cdot(P\cdot c)\mathsf{hsame0} \wedge (c\cdot P)\cdot d\mathsf{hsame0} \wedge d\cdot(c\cdot P)\mathsf{hsame0} \\ \wedge\ (Q\cdot c)\cdot d\mathsf{hsame0} \wedge d\cdot(Q\cdot c)\mathsf{hsame0} \wedge (c\cdot Q)\cdot d\mathsf{hsame0} \wedge d\cdot(c\cdot Q)\mathsf{hsame0} \end{array}\right).$$

Proof. Let S be the square-equality witness $a \cdot a\mathsf{hsame}b \cdot b$. Apply Definition 44.77 to S. The forward package supplies the universal two-sided rows for P, and the commuted package supplies the universal two-sided rows for Q.

Specializing those universal rows at c gives four zero factors: $P{\cdot}c\mathsf{hsame0}$, $c{\cdot}P\mathsf{hsame0}$, $Q{\cdot}c\mathsf{hsame0}$, and $c \cdot Q\mathsf{hsame0}$.

Apply the two rows of Definition 43.39 to each of these four zero factors with multiplier d. The right-context row gives $(P\cdot c)\cdot d\mathsf{hsame0}$, $(c\cdot P)\cdot d\mathsf{hsame0}$, $(Q\cdot c)\cdot d\mathsf{hsame0}$, and $(c\cdot Q)\cdot d\mathsf{hsame0}$; the left-context row gives $d\cdot(P\cdot c)\mathsf{hsame0}$, $d\cdot(c\cdot P)\mathsf{hsame0}$, $d\cdot(Q\cdot c)\mathsf{hsame0}$, and $d\cdot(c\cdot Q)\mathsf{hsame0}$. These eight witnesses are exactly the displayed context rows. □

The ten-field fit for this theorem is: traditional target full two-sided visible-context annihilation in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from bilateral signed-product annihilation and zero absorption; ledger

square equality plus the two visible context histories; core theorem Definition 44.79; exactness the eight displayed context rows; downstream use by $\mathsf{Field}^{\uparrow}$ zero-product obstruction and localization boundary; status ProofTarget.

The ten-field fit is: traditional target bilateral signed-product annihilation in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from difference-of-squares exactness, commutativity, and zero-factor absorption; ledger square equality plus both oriented product-zero and universal multiplier rows; core theorem this block; exactness the displayed bilateral package; standard bridge ordinary commutative-ring equal-squares annihilation; status ProofTarget.

The ten-field fit is: traditional target equal squares yields a bilateral signed-product zero certificate in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from difference-of-squares exactness and multiplicative commutativity; ledger the square-equality witness, the forward signed-product zero witness, and the commuted signed-product zero witness; core theorem this block; exactness the two oriented signed products at zero; downstream consumer the $\mathsf{Field}^{\uparrow}$ equal-square signed-factor exclusion package; status ProofTarget.

The ten-field fit is: traditional target equal squares yields the signed-product annihilator package in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from difference-of-squares exactness and zero-factor absorption; ledger the square-equality witness, signed product-zero witness, and universal two-sided multiplier row; core theorem this block; exactness the same annihilator certificate reached from square equality; downstream consumer $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; status ProofTarget.

The ten-field fit is: traditional target signed-product annihilator package in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from right distributivity and zero-factor absorption; ledger the selected zero linear factor, signed product-zero witness, and universal two-sided multiplier row; core theorem this block; exactness the paired product-zero and signed-product annihilator certificate; downstream consumer $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; status ProofTarget.

The ten-field fit is: traditional target branch-preserving zero-product certificate for both signed orientations; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from the two signed-product zero theorems; ledger the chosen zero linear factor and the two oriented zero-product witnesses; core theorem this block; exactness two product orientations at once; standard bridge ordinary commutative-ring factor cancellation packaging; status ProofTarget for downstream consumer $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages.

Theorem 44.80 (CommRing zero linear factor forces equal squares)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0) \Longrightarrow a \cdot a\mathsf{hsame}b \cdot b.$$

Proof. Apply Definition 44.66 to the chosen zero linear factor. This produces $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

The forward direction of Definition 44.47 sends this signed product-zero witness to $a \cdot a\mathsf{hsame}b \cdot b$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor equal squares`

↪ variant `BEDC.Derived.CommRingUp.commring empty linear factor equal squares`

The ten-field fit is: traditional target zero linear factor implies equal squares in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from zero-factor absorption and difference-of-squares exactness; ledger the chosen zero factor and product-zero witness; core theorem this block; exactness equality of squares; standard bridge ordinary commutative-ring factor argument; status $\mathsf{ProofTarget}$.

Theorem 44.81 (CommRing zero linear factor exact package with right distributivity)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left and right distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0\big) \Longrightarrow \big((a+b)\cdot(a+(-b))\mathsf{hsame}0 \wedge a\cdot a\mathsf{hsame}b\cdot b\big).$$

Proof. Apply Definition 44.66 to the chosen zero linear factor. This gives the signed product-zero witness $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

Feed that witness through the forward direction of Definition 44.47. The resulting square equality is $a\cdot a\mathsf{hsame}b\cdot b$, and pairing it with the signed product-zero witness gives the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor exact package`

Theorem 44.82 (CommRing zero linear factor exact package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0\big) \Longrightarrow \big((a+b)\cdot(a+(-b))\mathsf{hsame}0 \wedge a\cdot a\mathsf{hsame}b\cdot b\big).$$

Proof. Use Definition 44.34 to recover right distributivity from left distributivity, multiplicative commutativity, and additive congruence.

Apply Definition 44.81 with this right distributivity field and the remaining stated hypotheses. The result is the displayed conjunction for the chosen zero linear factor. □

The ten-field fit is: traditional target zero linear factor exact package in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from zero-factor absorption and difference-of-squares exactness; ledger the chosen linear factor and signed-product witness; core theorem this block; exactness the paired conjunction of product zero and equal squares; standard bridge ordinary commutative-ring square exit; status $\mathsf{ProofTarget}$.

Theorem 44.83 (CommRing zero linear factor total exact package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, addition and multiplication congruence, left distributivity, the* $\mathsf{Ring}^{\uparrow}$ *negative-product laws, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field used in Definition 44.47, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\big((a{+}b)\mathsf{hsame}0 \vee (a{+}(-b))\mathsf{hsame}0\big) \Longrightarrow \Big(\big((a{+}b)\mathsf{hsame}0 \wedge \forall c : \mathsf{BHist},\ ((a{+}b)\cdot c\mathsf{hsame}0 \wedge c\cdot(a{+}b)\mathsf{hsame}0)\big) \vee \big((a{+}(-b))\mathsf{hs}$$

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor total exact package`

Proof. Let H be the displayed zero-linear-factor assumption. Apply Definition 44.65 to H to obtain the branch-preserving annihilator disjunction.

Apply Definition 44.82 to the same assumption H. This yields the paired signed product-zero witness and square equality.

Introduce the final conjunction by placing the branch-preserving disjunction in the first component and the paired product-zero and square-equality certificate in the second component. □

The ten-field fit is: traditional target zero linear factor exact package in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from zero-factor absorption and difference-of-squares exactness; ledger the chosen linear factor, branch annihilator row, and signed-product witness; core theorem Definition 44.83; exactness branch annihilator coverage together with signed product zero and equal squares; standard bridge ordinary commutative-ring square exit; status $\mathsf{ProofTarget}$.

Theorem 44.84 (CommRing zero linear factor branch exact package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the same hypotheses as Definition 44.82, every* $a, b : \mathsf{BHist}$ *satisfies*

$$((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0) \Longrightarrow ((((a+b)\mathsf{hsame}0 \wedge \forall c : \mathsf{BHist},\ ((a+b)\cdot c\mathsf{hsame}0 \wedge c\cdot(a+b)\mathsf{hsame}0)) \vee ((a$$

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor branch exact package from empty unit`

Proof. Start with the displayed zero-linear-factor disjunction. Apply Definition 44.65 to retain the selected branch and its two-sided annihilator row for every $c : \mathsf{BHist}$.

Apply Definition 44.80 to the same disjunction. This gives the square equality $a \cdot a\mathsf{hsame}b \cdot b$ under the identical hypotheses.

Pair the branch-preserving annihilator package with the square equality. The result is exactly the displayed conjunction, so later selector arguments can consume the chosen linear factor without reconstructing the branch. □

The ten-field fit is: traditional target branch-preserving zero-factor exactness in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from zero-factor absorption and difference-of-squares exactness; ledger the selected linear factor and annihilator row; core theorem this block; exactness the displayed conjunction; downstream consumer $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; status $\mathsf{ProofTarget}$.

Theorem 44.85 (CommRing zero linear factor bilateral exact package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the same hypotheses as Definition 44.82, every* $a, b : \mathsf{BHist}$ *satisfies*

$$((a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0) \Longrightarrow ((((a+b)\mathsf{hsame}0 \wedge \forall c : \mathsf{BHist},\ ((a+b)\cdot c\mathsf{hsame}0 \wedge c\cdot(a+b)\mathsf{hsame}0)) \vee ((a$$

Lean4 ✓ `BEDC.Derived.CommRingUp.commring zero linear factor bilateral exact package`

Proof. Let H be the displayed zero-linear-factor hypothesis. Apply Definition 44.84 to H to obtain the branch-preserving annihilator disjunction together with the square equality $a \cdot a\mathsf{hsame}b \cdot b$.

Apply Definition 44.66 to H. This yields the forward signed product $(a+b)\cdot(a+(-b))\mathsf{hsame}0$.

Apply Definition 44.67 to H. This yields the reversed signed product $(a+(-b))\cdot(a+b)\mathsf{hsame}0$.

Introduce the final conjunction by placing the branch annihilator package first, the forward signed product second, the reversed signed product third, and the square equality fourth. The result is the displayed bilateral certificate. □

The ten-field fit is: traditional target zero linear factor bilateral exact package in a commutative ring; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from zero-factor absorption and difference-of-squares exactness; ledger the selected linear factor, branch annihilator row, both signed product orientations, and square equality; core theorem this block; exactness the displayed four-component certificate; downstream consumer $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; status $\mathsf{ProofTarget}$.

Theorem 44.86 (CommRing quadratic zero-linear bridge package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the shared hypotheses of Definition 44.60 and Definition 44.85, every $a, b : \mathsf{BHist}$ satisfies*

$$((a{+}b)\mathsf{hsame}0 \vee (a{+}(-b))\mathsf{hsame}0) \Longrightarrow \Big(\Big(\Big(((a+b)\cdot(a+b)) + ((a+(-b))\cdot(a+(-b)))\mathsf{hsame}(a\cdot a + a\cdot a) + (b\cdot b +$$

Proof. Apply Definition 44.60 to the shared hypotheses and the selected histories a, b. This produces the full quadratic-polarization package in the displayed order.

Let H be the zero-linear-factor branch witness. Apply Definition 44.85 to H under the same operation hypotheses. This produces the full bilateral certificate: the selected branch with its two-sided annihilator row, both signed product orientations, and the square equality.

Introduce the final conjunction with the quadratic-polarization package as the first component and the bilateral zero-linear-factor certificate as the second component. The branch witness is consumed only by the bilateral component, so the resulting certificate remains over the same concrete BHist source. □

The ten-field fit is: traditional target a commutative-ring package combining quadratic polarization with zero-linear-factor exactness; BEDC source $\mathsf{Ring}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from the shared quadratic and zero-factor operation laws; ledger the polarization identities, exact signed-square test, selected linear factor, two-sided annihilator row, bilateral signed products, and square equality; core theorem this block; exactness the displayed combined certificate; downstream use by $\mathsf{Field}^{\uparrow}$ quadratic normalization and zero-factor exit packages; status $\mathsf{ProofTarget}$.

Theorem 44.87 (CommRing zero-linear FieldUp exit consumer surface)**.** *For the concrete $\mathsf{CommRing}^{\uparrow}$ history package over BHist, the zero-linear annihilator branch, signed-product zero rows, equal-squares signed-factor rows, total exact package, bilateral exact package, and quadratic zero-linear bridge package expose exactly the $\mathsf{Field}^{\uparrow}$ consumer surface needed by the phase-exit certificate. The exposed surface consists of the carrier rows inherited from $\mathsf{Ring}^{\uparrow}$ and $\mathsf{CommRing}^{\uparrow}$, the classifier rows supplied by the signed-product package, the zero-factor exactness rows, the quadratic readback rows, and the finite ledger rows consumed by the $\mathsf{Field}^{\uparrow}$ phase-exit certificate. It contains no field inverse row, no division row, and no host polynomial-root datum.*

Proof. First project the zero-linear branch through Definition 44.65 and Definition 44.83. These results place the selected branch, its two-sided annihilator behavior, and its zero-factor exactness inside the finite BHist ledger.

Next project the classifier rows. The signed-product rows of Definition 44.66 and Definition 44.67 give the two orientations, while Definition 44.80 and Definition 44.85 identify the equal-squares row and keep the classifier stable under both signed factors.

Finally apply Definition 44.86. It repacks the quadratic readback rows together with the bilateral zero-linear certificate, so the downstream $\mathsf{Field}^{\uparrow}$ exit sees one concrete carrier, classifier, exactness, readback, and ledger surface over BHist. Since every step is a projection from the listed commutative-ring packages, no inverse, division, or host polynomial-root row is introduced. □

Theorem 44.88 (CommRing singleton empty square exact package)**.** *For the singleton empty-history CommRing tuple of Definition 44.6, every pair* $a, b : \mathsf{BHist}$ *carried by* A_{e} *satisfies*

$$\big((a+b)\cdot(a+(-b))\mathsf{hsame}0\big) \wedge \big(a\cdot a\mathsf{hsame}b\cdot b\big),$$

where $+$, $\cdot$, $-$, *and* 0 *are the displayed singleton operations.*

Proof. Theorem 44.7 supplies the concrete singleton $\mathsf{CommRing}^{\uparrow}$ laws for the operation tuple of Definition 44.6. Since the displayed addition and zero operations are constant at e, the carried inputs identify the sum endpoint with the zero endpoint, giving $(a+b)\mathsf{hsame}0$.

Choose the left branch of the zero-linear-factor disjunction:

$$(a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0.$$

The same singleton law package supplies the additive, congruence, distributive, negative-product, and multiplicative-commutativity fields needed to instantiate Theorem 44.82.

Applying that theorem to the selected left branch yields the signed product zero witness together with the square equality. These are exactly $((a+b)\cdot(a+(-b))\mathsf{hsame}0)$ and $(a\cdot a\mathsf{hsame}b\cdot b)$ for the singleton empty-history CommRing tuple. □

The ten-field fit is: traditional target one-element commutative-ring square exactness; BEDC source singleton empty-history $\mathsf{CommRing}^{\uparrow}$; carrier A_{e}; classifier ρ_{e}; stability from singleton CommRing laws plus the zero-linear-factor exact package; ledger the selected empty linear factor and signed-product witness; core theorem this block; exactness the paired product-zero and equal-square conclusion; standard bridge the one-element commutative ring; status $\mathsf{ProofTarget}$.

Theorem 44.89 (CommRing singleton empty signed product annihilator package)**.** *For the singleton empty-history CommRing tuple of Definition 44.6, every pair* $a, b : \mathsf{BHist}$ *carried by* A_{e} *satisfies*

$$((a+b)\cdot(a+(-b))\mathsf{hsame}0)\wedge\forall c : \mathsf{BHist},\ ((((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame}0)\wedge(c\cdot((a+b)\cdot(a+(-b)))\mathsf{hsame}0)),$$

where $+$, $\cdot$, $-$, *and* 0 *are the displayed singleton operations.*

Proof. Theorem 44.7 supplies the concrete singleton $\mathsf{CommRing}^{\uparrow}$ laws for the operation tuple of Definition 44.6. Because the displayed addition and zero operations are constant at e, carriedness of a and b identifies the sum endpoint with the zero endpoint, giving $(a+b)\mathsf{hsame}0$.

Choose the left branch of the zero-linear-factor disjunction:

$$(a+b)\mathsf{hsame}0 \vee (a+(-b))\mathsf{hsame}0.$$

The same singleton law package supplies the additive, congruence, distributive, negative-product, and multiplicative-commutativity fields needed to instantiate Theorem 44.72.

That theorem yields the signed product-zero witness together with the universal two-sided annihilator row for the signed product. Unpacking those two components gives exactly $((a+b)\cdot(a+(-b))\mathsf{hsame}0)$ and, for every $c : \mathsf{BHist}$, both $(((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame}0)$ and $(c\cdot((a+b)\cdot(a+(-b)))\mathsf{hsame}0)$ for the singleton empty-history CommRing tuple. □

The ten-field fit is: traditional target one-element commutative-ring signed-product annihilator; BEDC source singleton empty-history $\mathsf{CommRing}^{\uparrow}$; carrier A_{e}; classifier ρ_{e}; stability from singleton CommRing laws plus zero-factor absorption; ledger selected empty linear factor and signed-product annihilator row; core theorem this block; exactness product-zero plus universal two-sided annihilation; standard bridge one-element commutative ring; status $\mathsf{ProofTarget}$.

Theorem 44.90 (CommRing singleton empty square and signed-product annihilator package)**.** *For the singleton empty-history CommRing tuple of Definition 44.6, every pair $a, b : \mathsf{BHist}$ carried by A_{e} satisfies*

$$(a{\cdot}a\mathsf{hsame}b{\cdot}b)\wedge((a{+}b){\cdot}(a{+}(-b))\mathsf{hsame}0)\wedge\forall c : \mathsf{BHist},\ ((((a+b)\cdot(a+(-b)))\cdot c\mathsf{hsame}0)\wedge(c\cdot((a+b)\cdot(a+(-b)))\mathsf{h}$$

where $+$, $\cdot$, $-$, and 0 are the displayed singleton operations.

Proof. Apply Theorem 44.88 to the selected histories a, b. This yields the signed-product zero row together with the square-equality row for the singleton empty-history CommRing tuple.

Apply Theorem 44.89 to the same selected histories. This yields the same signed-product zero row and the universal two-sided annihilator row for that signed product.

Introduce the displayed triple conjunction. Use the square row from the square-exact package as the first component, use either signed-product zero row as the second component, and use the universal annihilator row from the signed-product annihilator package as the final component. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commringSingletonEmpty square signed product annihilator package`

The ten-field fit is: traditional target one-element commutative-ring package combining singleton square exactness with signed-product annihilation; BEDC source singleton empty-history $\mathsf{CommRing}^{\uparrow}$; carrier A_{e}; classifier ρ_{e}; stability from the singleton square-exact and signed-product annihilator packages; ledger the equal-square row, signed-product zero row, and universal two-sided annihilator row; core theorem this block; exactness the displayed three-part certificate; downstream use by $\mathsf{Field}^{\uparrow}$ quadratic and zero-factor exit packages; status $\mathsf{ProofTarget}$.

Theorem 44.91 (CommRing singleton empty square endpoint annihilator package)**.** *For the singleton empty-history CommRing tuple of Definition 44.6, every pair $a, b : \mathsf{BHist}$ carried by A_{e} satisfies*

$$(a{\cdot}a\mathsf{hsame}b{\cdot}b)\wedge\forall c : \mathsf{BHist},\ ((((a\cdot a)\cdot c)\mathsf{hsame}0)\wedge(c\cdot(a\cdot a)\mathsf{hsame}0)\wedge(((b\cdot b)\cdot c)\mathsf{hsame}0)\wedge(c\cdot(b\cdot b)\mathsf{hsame}0))\,,$$

where $\cdot$ and 0 are the displayed singleton operations.

Lean4 ✓ `BEDC.Derived.CommRingUp.commringSingletonEmpty square endpoint annihilator package`

Proof. Apply Theorem 44.90 to the selected histories a, b. Its first component gives the required square row $a \cdot a\mathsf{hsame}b \cdot b$.

Next use Theorem 44.7. In the singleton empty-history CommRing tuple, multiplication and zero both select the singleton endpoint, so carriedness identifies $a{\cdot}a$ and $b{\cdot}b$ with the zero endpoint.

Now instantiate the zero-factor absorption mechanism of Theorem 43.40 twice, first with $x = a \cdot a$ and then with $x = b \cdot b$. For an arbitrary multiplier $c : \mathsf{BHist}$, the two instantiations provide both left and right zero-factor rows for the two square endpoints.

Package these outputs as the displayed conjunction: the square-equality row is the first component, and the four universal multiplier rows are the second component. □

The ten-field fit is: traditional target one-element commutative-ring square ideal boundary; BEDC source singleton empty-history $\mathsf{CommRing}^{\uparrow}$; carrier A_{e}; classifier ρ_{e}; stability from singleton laws and zero-factor absorption; ledger carried inputs, two square endpoints, and arbitrary multiplier; core theorem this block; exactness square endpoints are two-sided zero annihilators; downstream consumer $\mathsf{Field}^{\uparrow}$ square-apartness and zero-product exits; status $\mathsf{ProofTarget}$.

44.6 Public closure record

Theorem 44.92 (CommRing singleton public operation surface)**.** *For the concrete singleton history commutative-ring instance of Definition 44.2, the public carrier is A_{e}, the public classifier is R_{e}, and the public operation tuple is*

$$\mathsf{add}(h,k) := \mathsf{append}(h,k), \qquad \mathsf{neg}(h) := \mathsf{e}, \qquad 0 := \mathsf{e},$$

together with

$$\mathsf{mul}(h,k) := \mathsf{append}(h,k), \qquad 1 := \mathsf{e}.$$

Every public expression generated from carried endpoints by 0, 1*, unary negation, addition, and multiplication remains in A_{e}, and any two such expressions are classified by R_{e} against the singleton endpoint* e*.*

Proof. The public carrier, classifier, and operation symbols are exactly the rows displayed in Definition 44.2. The operation closure clauses for addition, negation, multiplication, and the distinguished constants are supplied by Theorem 44.3; the remaining schema rows and replacement stability are supplied by Theorem 44.4.

Proceed by induction on the generated public expression. The constant cases 0 and 1 are the empty endpoint. The negation case uses the constant endpoint e. The addition and multiplication cases use the append endpoint and the singleton append rows Theorems 44.21, 44.22, and 44.11 to keep the visible endpoint inside A_{e}.

For the classifier assertion, the induction hypotheses identify each generated endpoint with e. The append/context cancellation and split rows Theorems 44.19 and 44.20 transport that endpoint classification through public operation contexts, while Theorem 44.24 records the commutative multiplication row on the same singleton surface.

Finally, Theorem 44.5 packages these carrier, classifier, operation, stability, ledger, exactness, and terminal-bridge rows as the concrete $\mathsf{CommRing}^{\uparrow}$ certificate boundary. Thus the instance exposes a public carrier/classifier/operation surface rather than only scoped singleton rows. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingUp StdBridge`

Theorem 44.93 (CommRing singleton concrete schema unlock)**.** *The singleton empty-history $\mathsf{Ring}^{\uparrow}$ carrier A_{e}, classifier R_{e}, zero row, one row, addition row, negation row, and multiplication row, together with the singleton multiplication-commutativity row and the square-package rows displayed in this child file, instantiate the $\mathsf{CommRing}^{\uparrow}$ schema as a concrete BHist carrier. The resulting surface is not merely an abstract operator schema: its carrier is A_{e}, its classifier is R_{e}, and all public ring and commutative multiplication operations are generated by the empty-history package.*

Future Lean targets for this surface should live in BEDC.Derived.CommRingUp *over this concrete carrier, classifier, and operation tuple rather than as parameter-transport schema over an unspecified commutative-ring family.*

Proof. Start with the $\mathsf{Ring}^{\uparrow}$ singleton public surface of Definition 44.92. It identifies the carrier, classifier, zero, one, addition, negation, and multiplication rows as the concrete empty-history package over A_{e} and R_{e}.

Next use the commutative multiplication row Definition 44.24. Because both multiplication orders are append endpoints classified by the same singleton predicate, the commutative-ring multiplication obligation is a row over the displayed BHist carrier, not a transported parameter.

Finally combine the square and zero closure rows of Definition 44.88, Definition 44.90, and Definition 44.91. These rows keep the square-package consumers on the same singleton carrier and classifier, so the $\mathsf{CommRing}^{\uparrow}$ schema is opened by a concrete empty-history certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$.

Constructive story: The commutative-ring package keeps the same empty-history singleton used for $\mathsf{Ring}^{\uparrow}$ and asks the multiplication row to ignore the order in which two carried endpoint witnesses are continued. Every admitted history is history-equal to e, every comparison is the singleton classifier, and both $\mathsf{Cont}(a, b, \mathsf{e})$ and $\mathsf{Cont}(b, a, \mathsf{e})$ are pulled back by the same endpoint witnesses. The source is the singleton predicate, the pattern records the existing ring operation table plus this multiplication-swap row, and the ledger keeps visible singleton histories aligned with their raw representatives. Because the singleton class is inhabited and history-equality is an equivalence preserving it, nonemptiness, reflexivity, symmetry, transitivity, and carrier closure all follow from the endpoint structure. A history h lies in the signature gap of a $\mathsf{CommRing}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is history-equal to e and p is the singleton commutative-ring element. The signed-square product and endpoint-annihilator facts, together with the terminal bridge, are direct readbacks of the displayed zero, negation, and singleton multiplication rows. The chapter registers this multiplication-swap singleton package under the name $\mathsf{CommRing}^{\uparrow}$; commutativity is the added endpoint obligation over $\mathsf{Ring}^{\uparrow}$, with no spectra, field-inverse data, or non-singleton commutative-ring family introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The paper-side mature scope is the singleton empty-history carrier A_{e}, classifier R_{e}, public operation tuple, singleton certificate boundary, terminal bridge, and the square/annihilator algebraic packages of Definition 44.90 and Definition 44.91.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CommRingUp.singleton\_empty\_history\_commring\_laws`

Bridge status: bridgeChecked

Not claimed: This record does not close arbitrary commutative rings, non-singleton examples, spectra, field inverse data, or machine-checked bridge verification.

Upgrade path: no further upgrade – chapter at top of formal axis

44.7 Singleton Cont History Instance for $\mathsf{CommRing}^{\uparrow}$

Definition 44.94 (CommRingUp singleton Cont history instance)**.** *Let A_{e} be the singleton* BHist *carrier whose only accepted endpoint is* e*, and let R_{e} be the classifier that identifies two accepted endpoints exactly when both are* hsame *to* e*. Define addition and multiplication by the same concrete continuation endpoint:*

$$x +_{\mathsf{e}} y := \mathsf{Cont}(x, y, \mathsf{e}), \qquad x \cdot_{\mathsf{e}} y := \mathsf{Cont}(x, y, \mathsf{e}).$$

The zero and one rows are both the displayed empty endpoint e*. The package ledger records the two input endpoints, the append read, the output endpoint, and the singleton classifier witness returning the output to* e*.*

Theorem 44.95 (Singleton Cont commutative-ring laws)**.** *The carrier A_{e} with classifier R_{e}, operations $+_{\mathsf{e}}$ and $\cdot_{\mathsf{e}}$, and constants $0_{\mathsf{e}} = 1_{\mathsf{e}} = \mathsf{e}$ satisfies the* $\mathsf{CommRing}^{\uparrow}$ *law surface. Additive group laws, multiplicative monoid laws, distributivity, and multiplicative commutativity all read back to the singleton endpoint under* hsame*.*

Proof. Every accepted input endpoint is classified with e by R_{e}. The continuation endpoint for either operation is therefore an accepted singleton endpoint and is again classified with e.

For associativity, identity, and inverse rows on addition, both sides of each law are continuation endpoints whose source and target histories classify to e. The additive inverse operation is the constant empty endpoint, so the left and right inverse rows also reduce to the same readback.

The multiplicative identity, associativity, and commutativity rows are identical endpoint-readback arguments, because multiplication uses the same concrete continuation endpoint and its unit is also e.

For distributivity, expand both sides as finite continuation endpoints over accepted singleton histories. Each displayed endpoint returns to e under R_{e}, so the two sides classify together. Repacking these rows gives the commutative-ring law surface without quantified abstract operators. □

Theorem 44.96 (CommRingUp singleton Cont schema source)**.** *The singleton Cont history instance of Definition 44.94 is a concrete* BHist*-anchored source for* $\mathsf{CommRing}^{\uparrow}$*. Future formal targets may use this carrier, classifier, operation table, constants, and endpoint ledger as the ground source instead of proving the commutative-ring schema from arbitrary carrier parameters and abstract binary operations.*

Proof. The definition supplies all rows demanded by the commutative-ring source: carrier, classifier, addition, multiplication, zero, one, and package provenance. The preceding theorem supplies the finite law rows for the same objects.

Thus a consumer can cite the singleton packet directly. No external carrier, parameter family, host equality, or abstract operation table is needed to obtain a concrete $\mathsf{CommRing}^{\uparrow}$ source. □

Definition 44.97 (CommRing left zero-divisor data)**.** *For a BEDC history-level* $\mathsf{CommRing}^{\uparrow}$ *operation tuple with multiplication* $\cdot$ *and zero* 0*, write*

$$\mathsf{ApartZero}_R(x) := (x\mathsf{hsame}0 \to \bot).$$

The strict left zero-divisor predicate is

$$\mathsf{LeftZD}_R(x) := \mathsf{ApartZero}_R(x) \wedge \exists c : \mathsf{BHist},\ \mathsf{ApartZero}_R(c) \wedge x \cdot c\mathsf{hsame}0.$$

Lean4 (definition) `BEDC.Derived.CommRingUp.CommRingApartZero`
Lean4 (definition) `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor`

Theorem 44.98 (CommRing apart-zero visible pair)**.** *For every history* $h : \mathsf{BHist}$*, both visible constructors are apart from the empty zero history:*

$$\mathsf{ApartZero}_R(\mathsf{E0}(h)) \wedge \mathsf{ApartZero}_R(\mathsf{E1}(h)).$$

Proof. Each conjunct is the corresponding BEDC history no-confusion law: neither an $\mathsf{E0}$-headed history nor an $\mathsf{E1}$-headed history can be history-same as e. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingApartZero visible pair`

Definition 44.99 (CommRing strict two-sided zero-divisor data)**.** *For the same BEDC history-level* $\mathsf{CommRing}^{\uparrow}$ *operation tuple, the strict two-sided zero-divisor predicate is*

$$\mathsf{StrictZD}_R(x) := \mathsf{ApartZero}_R(x) \wedge \exists c : \mathsf{BHist},\ \mathsf{ApartZero}_R(c) \wedge x \cdot c\mathsf{hsame}0 \wedge c \cdot x\mathsf{hsame}0.$$

Lean4 (definition) `BEDC.Derived.CommRingUp.CommRingStrictZeroDivisor`

Theorem 44.100 (CommRing singleton multiplication visible strict zero-divisors)**.** *For every BEDC history* h*, both visible singleton histories* $\mathsf{E0}(h)$ *and* $\mathsf{E1}(h)$ *are strict two-sided zero-divisors for the singleton commutative-ring multiplication:*

$$\mathsf{StrictZD}_{\mathsf{singleton}}(\mathsf{E0}(h)) \wedge \mathsf{StrictZD}_{\mathsf{singleton}}(\mathsf{E1}(h)).$$

Proof. The singleton multiplication sends every pair of histories to e. For $\mathsf{E0}(h)$ use $\mathsf{E1}(h)$ as the displayed annihilator, and for $\mathsf{E1}(h)$ use $\mathsf{E0}(h)$. Constructor disjointness gives the apart-from-zero clauses for both visible histories, while both ordered products reduce to e. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingSingletonMul visible strictZeroDivisor`

Theorem 44.101 (CommRing left zero-divisor strictness from commutativity)**.** *For any BEDC history multiplication mul satisfying multiplicative commutativity under* hsame*, every left zero-divisor carries a two-sided annihilator witness:*

$$\mathsf{LeftZD}_R(x) \Longrightarrow \mathsf{ApartZero}_R(x) \wedge \exists c : \mathsf{BHist},\ \mathsf{ApartZero}_R(c) \wedge x \cdot c\mathsf{hsame}0 \wedge c \cdot x\mathsf{hsame}0.$$

Proof. Unpack the left zero-divisor data. It gives the apartness of x and a witness c with $\mathsf{ApartZero}_R(c)$ and $x \cdot c\mathsf{hsame}0$. Apply multiplicative commutativity to obtain $c \cdot x\mathsf{hsame}x \cdot c$, then compose by transitivity with the displayed left annihilation row. The same witness c therefore supplies both annihilator orientations. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor strict of mul comm`

Theorem 44.102 (CommRing left zero-divisor strictness iff under commutativity)**.** *For any BEDC history multiplication mul, written as* $\cdot$*, satisfying multiplicative commutativity under* hsame*, every* $x : \mathsf{BHist}$ *satisfies*

$$(\forall uv : \mathsf{BHist},\ u \cdot v\mathsf{hsame}v \cdot u) \Longrightarrow (\mathsf{LeftZD}_R(x) \Longleftrightarrow \mathsf{StrictZD}_R(x)).$$

Proof. For the forward implication, assume $\mathsf{LeftZD}_R(x)$ and the displayed multiplicative commutativity row. Apply Definition 44.101. Its conclusion is the apartness of x together with an apart annihilator c satisfying both $x \cdot c\mathsf{hsame}0$ and $c \cdot x\mathsf{hsame}0$, which is exactly $\mathsf{StrictZD}_R(x)$ by Definition 44.99.

For the reverse implication, unfold $\mathsf{StrictZD}_R(x)$. The strict two-sided data gives $\mathsf{ApartZero}_R(x)$ and a witness $c : \mathsf{BHist}$ with $\mathsf{ApartZero}_R(c)$, $x \cdot c\mathsf{hsame}0$, and $c \cdot x\mathsf{hsame}0$. Dropping the right-annihilation conjunct leaves precisely the apartness of x, the same witness c, its apartness, and $x \cdot c\mathsf{hsame}0$, hence $\mathsf{LeftZD}_R(x)$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor strictZeroDivisor of mul comm iff`
Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor strict iff of mul comm`
Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor strict expansion iff of mul comm`

Ten-field fit: traditional target left and two-sided zero-divisor collapse in commutative rings; BEDC source $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability multiplicative commutativity and transitivity; ledger selected annihilator witness; core theorem Definition 44.102; exactness equivalence of one-sided and two-sided annihilator data; standard bridge ordinary commutative-ring zero divisors; status $\mathsf{ProofTarget}$.

Theorem 44.103 (CommRing zero-divisor product closure)**.** *For BEDC history operations add, mul, neg, and zero satisfying the* $\mathsf{Ring}^{\uparrow}$ *additive abelian-group laws, multiplicative associativity, addition and multiplication congruence, left distributivity, and the* $\mathsf{CommRing}^{\uparrow}$ *multiplicative commutativity field, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{LeftZD}_R(a) \Longrightarrow ((a \cdot b)\mathsf{hsame}0 \to \bot) \Longrightarrow \mathsf{LeftZD}_R(a \cdot b).$$

Proof. First derive right distributivity from the supplied left distributivity by Definition 44.34. With this row, the additive group laws, and multiplication congruence, the hypotheses of Definition 43.40 are available.

Unpack $\mathsf{LeftZD}_R(a)$. Its right component gives a history $c : \mathsf{BHist}$ with $\mathsf{ApartZero}_R(c)$ and $a \cdot c\mathsf{hsame}0$. Let P be the displayed proof that $a \cdot b$ is apart from zero.

The same c annihilates $a \cdot b$. Multiplicative associativity gives $(a \cdot b) \cdot c\mathsf{hsame}a \cdot (b \cdot c)$. Multiplicative commutativity gives $b \cdot c\mathsf{hsame}c \cdot b$, and multiplication congruence transports this inside left multiplication by a. Symmetry of associativity then identifies $a \cdot (c \cdot b)$ with $(a \cdot c) \cdot b$.

Apply Definition 43.40 to the zero classifier $a \cdot c\mathsf{hsame}0$ and multiplier b. Its left product component gives $(a \cdot c) \cdot b\mathsf{hsame}0$. Transitivity through the associativity and commutativity rows yields

$$(a \cdot b) \cdot c\mathsf{hsame}0.$$

The strict left zero-divisor data for $a \cdot b$ is therefore the pair consisting of P and the same annihilator witness c with its apart-zero proof and the displayed product-zero classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor product closed`

Theorem 44.104 (CommRing annihilator product left zero-divisor)**.** *Under the same hypotheses as Definition 44.103, every* $a, b, c : \mathsf{BHist}$ *satisfies*

$$\mathsf{ApartZero}_R(c) \Longrightarrow (a \cdot c\mathsf{hsame}0) \Longrightarrow ((a \cdot b)\mathsf{hsame}0 \to \bot) \Longrightarrow \mathsf{LeftZD}_R(a \cdot b).$$

Proof. Assume $\mathsf{ApartZero}_R(c)$, $a \cdot c\mathsf{hsame}0$, and $P : (a \cdot b)\mathsf{hsame}0 \to \bot$. We reuse the displayed c as the annihilator witness and show that it annihilates $a \cdot b$.

Multiplicative associativity gives $(a \cdot b) \cdot c\mathsf{hsame}a \cdot (b \cdot c)$. Multiplicative commutativity gives $b \cdot c\mathsf{hsame}c \cdot b$, and multiplication congruence transports this comparison inside left multiplication by a. Symmetry of associativity then identifies $a \cdot (c \cdot b)$ with $(a \cdot c) \cdot b$.

As in Definition 44.103, derive right distributivity from Definition 44.34 and use the zero-classifier factor absorption theorem. Applied to $a \cdot c\mathsf{hsame}0$ and the multiplier b, it yields $(a \cdot c) \cdot b\mathsf{hsame}0$. Transitivity through the associativity, commutativity, and congruence chain gives

$$(a \cdot b) \cdot c\mathsf{hsame}0.$$

The strict left zero-divisor data for $a \cdot b$ is the apart-zero component P together with the same witness c, its apart-zero proof, and the displayed product-zero classifier. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor annihilator product closed`
Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingAnnihilatorProduct leftZeroDivisor`
Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor product closed from annihilator`

Ten-field fit: traditional target nonzero multiple of an element with a nonzero annihilator; BEDC source $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability associativity, commutativity, congruence, and zero absorption; ledger selected annihilator witness; core theorem Definition 44.104; exactness explicit witness preservation; standard bridge ordinary commutative-ring zero-divisor propagation; status $\mathsf{ProofTarget}$.

Theorem 44.105 (CommRing strict zero-divisor product closure)**.** *Under the same operation hypotheses as Definition 44.103, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{StrictZD}_R(a) \Longrightarrow ((a \cdot b)\mathsf{hsame}0 \to \bot) \Longrightarrow \mathsf{StrictZD}_R(a \cdot b).$$

Proof. Unpack $\mathsf{StrictZD}_R(a)$. The strict witness supplies $c : \mathsf{BHist}$ together with $\mathsf{ApartZero}_R(c)$, $a \cdot c\mathsf{hsame0}$, and $c \cdot a\mathsf{hsame0}$. Let P be the given proof that $a \cdot b$ is apart from zero.

The left annihilation row for the product is the same chain as in Definition 44.103. Multiplicative associativity rewrites $(a \cdot b) \cdot c$ to $a \cdot (b \cdot c)$, commutativity rewrites $b \cdot c$ to $c \cdot b$, and congruence transports the comparison under left multiplication by a.

Associativity in the other direction identifies $a \cdot (c \cdot b)$ with $(a \cdot c) \cdot b$. The zero-classifier factor absorption theorem then sends $a \cdot c\mathsf{hsame0}$ to $(a \cdot c) \cdot b\mathsf{hsame0}$, so transitivity gives

$$(a \cdot b) \cdot c\mathsf{hsame0}.$$

For the right annihilation row, apply the $\mathsf{CommRing}^{\uparrow}$ multiplicative commutativity field to the two histories $(a \cdot b) \cdot c$ and $c \cdot (a \cdot b)$. Transport the displayed zero classifier across this comparison to obtain

$$c \cdot (a \cdot b)\mathsf{hsame0}.$$

Package P, the same apart-zero witness c, its apartness proof, and the two displayed annihilation rows. These are exactly the components of $\mathsf{StrictZD}_R(a \cdot b)$. □

Lean4 ✓ `BEDC.Derived.CommRingUp.commring strict zero divisor empty product closed`

Theorem 44.106 (CommRing left zero-divisor product strict closure)**.** *Under the same operation hypotheses as Definition 44.103, together with the multiplicative commutativity row used in Definition 44.101, every* $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{LeftZD}_R(a) \Longrightarrow ((a \cdot b)\mathsf{hsame0} \to \bot) \Longrightarrow \mathsf{StrictZD}_R(a \cdot b).$$

Proof. Assume $\mathsf{LeftZD}_R(a)$ and let $P : (a \cdot b)\mathsf{hsame0} \to \bot$ be the displayed apartness proof for the product. Apply Definition 44.103 to a, b, the left zero-divisor witness for a, and P. This gives $\mathsf{LeftZD}_R(a \cdot b)$ with the preserved annihilator.

Next apply Definition 44.101 to the product $a \cdot b$, using the same multiplicative commutativity row. The result unpacks into the apartness of $a \cdot b$ and a history $c : \mathsf{BHist}$ with $\mathsf{ApartZero}_R(c)$, $(a \cdot b) \cdot c\mathsf{hsame0}$, and $c \cdot (a \cdot b)\mathsf{hsame0}$.

These are precisely the fields of $\mathsf{StrictZD}_R(a \cdot b)$. Repack the apartness component and the two-sided annihilator witness c to obtain the claimed strict product closure. □

Lean4 ✓ `BEDC.Derived.CommRingUp.CommRingLeftZeroDivisor product strict closed`

The ten-field fit for Definition 44.106 is: traditional target nonzero multiple of a left zero-divisor; BEDC source $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from associativity, commutativity, congruence, and zero absorption; ledger the preserved annihilator and product apartness; core theorem Definition 44.106; exactness the two-sided strict witness recovered from the left witness; downstream consumer the $\mathsf{Field}^{\uparrow}$ zero-product obstruction and commutative-ring localization boundary; status $\mathsf{ProofTarget}$.

The ten-field fit for Definition 44.105 is: traditional target strict zero-divisor closure in a commutative ring; BEDC source $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from associativity, commutativity, congruence, and zero absorption; ledger the apartness witness plus both annihilator orientations; core theorem Definition 44.105; exactness the two-sided annihilator witness; standard bridge ordinary commutative-ring zero-divisor closure; status $\mathsf{ProofTarget}$.

Theorem 44.107 (CommRing left zero-divisor two-step product strict closure)**.** *Under the same operation hypotheses as Definition 44.106, every* $a, b, c : \mathsf{BHist}$ *satisfies*

$$\mathsf{LeftZD}_R(a) \Longrightarrow ((a \cdot b)\mathsf{hsame0} \to \bot) \Longrightarrow (((a \cdot b) \cdot c)\mathsf{hsame0} \to \bot) \Longrightarrow \mathsf{StrictZD}_R((a \cdot b) \cdot c).$$

Proof. Assume $\mathsf{LeftZD}_R(a)$ and let $P : (a \cdot b)\mathsf{hsame0} \to \bot$ be the apartness proof for the first product. Apply Definition 44.106 to a, b, the left zero-divisor witness for a, and P. This gives $\mathsf{StrictZD}_R(a \cdot b)$.

Let $Q : ((a \cdot b) \cdot c)\mathsf{hsame0} \to \bot$ be the apartness proof for the second product. Apply Definition 44.105 to the strict zero-divisor $a \cdot b$, the factor c, and Q.

The resulting package is exactly $\mathsf{StrictZD}_R((a \cdot b) \cdot c)$: its apartness component is Q, and its two-sided annihilator witness is the one transported by the cited strict product-closure theorem. □

The ten-field fit for Definition 44.107 is: traditional target iterated nonzero multiple of a zero divisor; BEDC source $\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability from the two cited product-closure theorems; ledger the two apartness hypotheses and preserved annihilator; core theorem Definition 44.107; exactness the strict two-sided witness for the second product; standard bridge ordinary commutative-ring zero-divisor propagation; status $\mathsf{ProofTarget}$.

Definition 44.108 (CommRing subring-inclusion name certificate)**.** *Let S and T be history-level $\mathsf{CommRing}^{\uparrow}$ packages as in Definition 44.59. Write their carriers and classifiers as $C_S, \sim_S$ and $C_T, \sim_T$, and write their operations as $0_S, 1_S, +_S, -_S, \cdot_S$ and $0_T, 1_T, +_T, -_T, \cdot_T$. For a history map $\iota : \mathsf{BHist} \to \mathsf{BHist}$, define the target image carrier*

$$\mathsf{Im}_\iota(y) \iff \exists x : \mathsf{BHist},\ C_S(x) \wedge \iota(x) \sim_T y.$$

Define the target-image classifier by

$$y \sim_\iota z \iff \mathsf{Im}_\iota(y) \wedge \mathsf{Im}_\iota(z) \wedge y \sim_T z.$$

A subring-inclusion name certificate for ι over T is the naming certificate of Definition 9.6 with source Im_ι, classifier $\sim_\iota$, stability rows inherited from the target $\mathsf{CommRing}^{\uparrow}$ package, and ledger rows recording the carrier reflection, classifier reflection, and operation-preservation data for 0, 1, $+$, and $\cdot$. Its stability component includes target-carrier inclusion, classifier transport, closure under 0_T, 1_T, $+_T$, $-_T$, and $\cdot_T$, and the inherited additive, multiplicative, distributive, congruence, and commutativity laws under $\sim_\iota$.

Theorem 44.109 (CommRing subring inclusion inverse closure)**.** *Let S and T be history-level $\mathsf{CommRing}^{\uparrow}$ packages as in Definition 44.59, with notation as in Definition 44.108. Let $\iota : \mathsf{BHist} \to \mathsf{BHist}$ satisfy carrier reflection, classifier reflection on carried source endpoints, zero preservation $\iota(0_S) \sim_T 0_T$, and addition preservation on carried inputs:*

$$\iota(x +_S u) \sim_T \iota(x) +_T \iota(u).$$

If $\mathsf{Im}_\iota(y)$, then $\mathsf{Im}_\iota(-_T y)$.

Proof. Choose a source witness x for $\mathsf{Im}_\iota(y)$, so $C_S(x)$ and $\iota(x) \sim_T y$. Source inverse closure gives $C_S(-_S x)$. The source left-inverse row gives $(-_S x) +_S x \sim_S 0_S$, and classifier reflection on carried source endpoints transports it to

$$\iota((-_S x) +_S x) \sim_T \iota(0_S).$$

Addition preservation and zero preservation turn this comparison into

$$\iota(-_S x) +_T \iota(x) \sim_T 0_T.$$

Target additive congruence with $\iota(x) \sim_T y$ gives

$$\iota(-_S x) +_T y \sim_T 0_T.$$

The target right-inverse row from Definition 43.27 gives $y +_T (-_T y) \sim_T 0_T$. Together with target additive associativity, the target left-zero law, Definition 43.31, and target addition congruence, Definition 41.59 identifies the two inverse witnesses:

$$\iota(-_S x) \sim_T -_T y.$$

Thus the carried source endpoint $-_S x$ witnesses $\mathsf{Im}_\iota(-_T y)$. The additive inverse row follows from + and 0 preservation, so no separate preservation assumption for $-_S$ is required. □

Theorem 44.110 (CommRing carrier reflection gives subring inclusion)**.** *Let S and T be history-level* $\mathsf{CommRing}^\uparrow$ *packages as in Definition 44.59, with notation as in Definition 44.108. Let* $\iota : \mathsf{BHist} \to \mathsf{BHist}$ *satisfy the following rows:*

1. *Carrier reflection:* $C_S(x) \Longrightarrow C_T(\iota(x))$ *and* $C_T(\iota(x)) \Longrightarrow C_S(x)$ *for every* x*.*
2. *Classifier reflection on carried source endpoints:*

$$C_S(x) \wedge C_S(y) \Longrightarrow (x \sim_S y \Longleftrightarrow \iota(x) \sim_T \iota(y)).$$

3. *Constant preservation:* $\iota(0_S) \sim_T 0_T$ *and* $\iota(1_S) \sim_T 1_T$*.*
4. *Operation preservation for carried inputs:*

$$\iota(x +_S y) \sim_T \iota(x) +_T \iota(y) \qquad \text{and} \qquad \iota(x \cdot_S y) \sim_T \iota(x) \cdot_T \iota(y).$$

Then the image carrier Im_ι *and classifier* $\sim_\iota$ *of Definition 44.108 form a subring-inclusion name certificate over the target package* T*.*

Proof. The target classifier is a carrier naming certificate by Definition 9.7. Hence it supplies reflexivity, symmetry, transitivity, and carrier transport for $\sim_T$. If $\mathsf{Im}_\iota(y)$ is witnessed by a carried source endpoint x with $\iota(x) \sim_T y$, carrier preservation gives $C_T(\iota(x))$, and target carrier transport gives $C_T(y)$. If also $y \sim_T y'$, transitivity gives $\iota(x) \sim_T y'$, so the same source witness proves $\mathsf{Im}_\iota(y')$. The restricted classifier $\sim_\iota$ is a $\mathsf{NameCert}$ on Im_ι. Inhabitedness is witnessed by 0_T: the source certificate carries 0_S, and the constant row gives $\iota(0_S) \sim_T 0_T$. Reflexivity, symmetry, and transitivity are the corresponding target classifier fields, with the image endpoints retained in the first two components of the definition of $\sim_\iota$. Carrier transport for Im_ι is immediate from the right image component of a $\sim_\iota$ witness. The constants are carried by the image for the same reason. The zero row uses 0_S and $\iota(0_S) \sim_T 0_T$; the unit row uses 1_S and $\iota(1_S) \sim_T 1_T$. For addition closure, choose image witnesses x, u for y, z, so $C_S(x)$, $C_S(u)$, $\iota(x) \sim_T y$, and $\iota(u) \sim_T z$. Source additive closure, part of the $\mathsf{Ring}^\uparrow$ component in Definition 43.26, gives $C_S(x +_S u)$. Operation preservation gives

$$\iota(x +_S u) \sim_T \iota(x) +_T \iota(u).$$

Target additive congruence transports the two image comparisons to

$$\iota(x) +_T \iota(u) \sim_T y +_T z.$$

Transitivity gives $\iota(x +_S u) \sim_T y +_T z$, so $x +_S u$ witnesses $\mathsf{Im}_\iota(y +_T z)$. Multiplication closure is identical, using source multiplicative closure, target multiplication congruence, and the product-preservation row.

Closure under target additive inverse is Definition 44.109, applied to the carrier reflection, classifier reflection, zero-preservation, and addition-preservation rows already listed. All remaining $\mathsf{CommRing}^{\uparrow}$ laws restrict from T. Associativity, commutativity of addition, additive inverse laws, multiplicative unit and associativity, distributivity, multiplication congruence, and multiplicative commutativity are target stability rows from Definition 44.33; their endpoints stay inside Im_ι by the closure rows just proved, and their classifier comparisons are exactly the third component of $\sim_\iota$. The reverse carrier and classifier reflection implications are recorded as the exactness part of the inclusion ledger, showing that the target image does not identify carried source endpoints beyond $\sim_S$. These rows supply the source, classifier, stability, and ledger components required by Definition 44.108, hence they induce the advertised subring-inclusion name certificate over T. □

Theorem 44.111 (CommRing subring inclusion transitivity)**.** *Let A, B, and C be history-level $\mathsf{CommRing}^{\uparrow}$ packages as in Definition 44.59. Write their carriers and classifiers as $D_A, \sim_A$, $D_B, \sim_B$, and $D_C, \sim_C$, and write their operations as $0_A, 1_A, +_A, -_A, \cdot_A$, $0_B, 1_B, +_B, -_B, \cdot_B$, and $0_C, 1_C, +_C, -_C, \cdot_C$. Let $\iota : \mathsf{BHist} \to \mathsf{BHist}$ carry a subring-inclusion name certificate from A to B, and let $\kappa : \mathsf{BHist} \to \mathsf{BHist}$ carry a subring-inclusion name certificate from B to C, both in the sense of Definition 44.108. Then the composite history map $\kappa \circ \iota$ carries a subring-inclusion name certificate from A to C of the same form. Its target image carrier is*

$$\mathsf{Im}_{\kappa\circ\iota}(z) \iff \exists x : \mathsf{BHist},\ D_A(x) \wedge \kappa(\iota(x)) \sim_C z,$$

and its target-image classifier is

$$z \sim_{\kappa\circ\iota} w \iff \mathsf{Im}_{\kappa\circ\iota}(z) \wedge \mathsf{Im}_{\kappa\circ\iota}(w) \wedge z \sim_C w.$$

Proof. Unpack the two input inclusion certificates using Definition 44.108. Each one supplies carrier reflection in both directions, classifier reflection on carried source endpoints, preservation of 0 and 1, and preservation of addition and multiplication on carried inputs. By Definition 44.110, it is enough to verify these rows for $\kappa \circ \iota$.

For carrier reflection, if $D_A(x)$ then the carrier-preservation row for ι gives $D_B(\iota(x))$, and the carrier-preservation row for κ gives $D_C(\kappa(\iota(x)))$. Conversely, if $D_C(\kappa(\iota(x)))$, the reverse carrier-reflection row for κ gives $D_B(\iota(x))$, and the reverse row for ι gives $D_A(x)$.

For classifier reflection, fix carried x, y in A. Carrier preservation for ι makes $\iota(x)$ and $\iota(y)$ carried in B. The classifier-reflection row for ι gives

$$x \sim_A y \iff \iota(x) \sim_B \iota(y),$$

and the classifier-reflection row for κ gives

$$\iota(x) \sim_B \iota(y) \iff \kappa(\iota(x)) \sim_C \kappa(\iota(y)).$$

Composing these two equivalences gives the classifier-reflection row for the composite.

For zero preservation, the A-certificate carries 0_A, so carrier preservation for ι makes $\iota(0_A)$ carried in B. The B-certificate carries 0_B. The zero row for ι gives $\iota(0_A) \sim_B 0_B$; applying the classifier-preservation direction for κ gives

$$\kappa(\iota(0_A)) \sim_C \kappa(0_B).$$

The zero row for κ gives $\kappa(0_B) \sim_C 0_C$, and classifier transitivity from Definition 9.8 yields $\kappa(\iota(0_A)) \sim_C 0_C$. The same argument with the unit rows gives $\kappa(\iota(1_A)) \sim_C 1_C$.

For addition preservation, let x, u be carried in A. Source additive closure in the A package gives $D_A(x +_A u)$, hence $D_B(\iota(x +_A u))$. The first inclusion certificate gives

$$\iota(x +_A u) \sim_B \iota(x) +_B \iota(u).$$

The endpoints $\iota(x)$ and $\iota(u)$ are carried in B, so B-addition closure carries $\iota(x) +_B \iota(u)$. Applying classifier preservation for κ gives

$$\kappa(\iota(x +_A u)) \sim_C \kappa(\iota(x) +_B \iota(u)).$$

The addition-preservation row for κ on the carried inputs $\iota(x)$ and $\iota(u)$ gives

$$\kappa(\iota(x) +_B \iota(u)) \sim_C \kappa(\iota(x)) +_C \kappa(\iota(u)).$$

Classifier transitivity composes these two comparisons, giving the addition-preservation row for $\kappa \circ \iota$.

The multiplication row is identical, using multiplicative closure in the A and B packages and the two multiplication-preservation rows:

$$\kappa(\iota(x \cdot_A u)) \sim_C \kappa(\iota(x)) \cdot_C \kappa(\iota(u)).$$

Thus the composite supplies carrier reflection, classifier reflection, constant preservation, and operation preservation in the exact format required by Definition 44.110. Applying that theorem to $\kappa \circ \iota$ gives the displayed image carrier, target-image classifier, stability rows inherited from the C-package, and the composite inclusion ledger. Therefore $\kappa \circ \iota$ carries the required subring-inclusion name certificate over C. □

Theorem 44.112 (CommRing singleton empty identity subring inclusion)**.** *For the singleton empty-history CommRing tuple of Definition 44.6, let $\iota : \mathsf{BHist} \to \mathsf{BHist}$ be the identity history map $\iota(h) := h$. Form the image carrier and target-image classifier of Definition 44.108 with source and target both this singleton package. Then, for every $y, z : \mathsf{BHist}$, the image carrier is exactly the singleton carrier:*

$$\mathsf{Im}_\iota(y) \iff A_{\mathsf{e}}(y).$$

The target-image classifier is exactly the singleton classifier:

$$y \sim_\iota z \iff \rho_{\mathsf{e}}(y, z).$$

Moreover, the identity map carries a subring-inclusion name certificate from the singleton empty-history CommRing package to itself.

Proof. Unfold the identity map and the image carrier. If $\mathsf{Im}_\iota(y)$ is witnessed by x, then the classifier component is $\rho_{\mathsf{e}}(x, y)$, and its target carrier field gives $A_{\mathsf{e}}(y)$.

Conversely, assume $A_{\mathsf{e}}(y)$. Choose $x = y$. The reflexive singleton classifier $\rho_{\mathsf{e}}(y, y)$ is assembled from the two copies of $A_{\mathsf{e}}(y)$ and the reflexive history comparison, so y witnesses $\mathsf{Im}_\iota(y)$.

For the classifier row, the forward direction of $y \sim_\iota z$ projects the third component, namely $\rho_{\mathsf{e}}(y, z)$. In the reverse direction, a classifier witness $\rho_{\mathsf{e}}(y, z)$ supplies the two singleton carrier fields; the image-carrier equivalence just proved turns them into $\mathsf{Im}_\iota(y)$ and $\mathsf{Im}_\iota(z)$, and repacking gives $y \sim_\iota z$.

It remains to verify the rows required by Definition 44.110. Since source and target are the same singleton tuple and ι is the identity map, carrier reflection and classifier reflection are reflexive.

Zero preservation, unit preservation, addition preservation, and multiplication preservation are the same singleton endpoint rows from Definition 44.7.

Applying Definition 44.110 gives the subring-inclusion name certificate for the identity map over the singleton target package. The displayed carrier and classifier rows are the exactness part of that certificate. □

The ten-field fit is: traditional target terminal commutative-ring self-inclusion; BEDC source singleton empty-history $\mathsf{CommRing}^{\uparrow}$; carrier A_{e}; classifier ρ_{e}; stability from Definition 44.7; ledger identity-map provenance; core theorem Definition 44.112; exactness the two displayed iff rows; standard bridge terminal ring identity inclusion; status $\mathsf{ProofTarget}$. $\mathsf{Field}^{\uparrow}$ reads its rational carrier through chapter 37 and its algebraic law surface through chapter 43 and chapter 44 before exporting inverse and affine-operation rows.

Chapter 45

A Concrete Naming Certificate for $\mathsf{Field}^{\uparrow}$

The field interface $\mathsf{Field}^{\uparrow}$ extends chapter 44 with the multiplicative inverse field. The inverse domain is a unary certificate $\mathsf{NonZero}_F(a)$ derived from constructive apartness of a from 0; inverse formation is licensed only by an explicit witness $p : \mathsf{NonZero}_F(a)$. This proof-relevant discipline replaces the classical "$a \neq 0$ implies a invertible" formulation, which would otherwise require LEM. The rational carrier from chapter 37 supplies the candidate carrier and classifier source; a $\mathsf{Field}^{\uparrow}$ certificate is present only when commutative-ring structure, constructive nonzero evidence, inverse closure, inverse laws indexed by nonzero witnesses, and the ledger are supplied over that source.

45.1 Carrier definition

Definition 45.1 (Field carrier)**.** *Use a commutative ring setup extended with a constructive apartness predicate and define* $\mathsf{NonZero}_F(a)$ *as the unary certificate obtained from apartness of* a *from* 0*. The inverse datum has type* $\mathsf{inv}(a, p)$ *for* $p : \mathsf{NonZero}_F(a)$*, so no inverse is formed without the nonzero witness.*

45.2 Concrete singleton empty-history field schema instance

Definition 45.2 (Singleton empty-history field schema instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$*. Define the classifier*

$$\rho_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Take add, neg, 0, mul, *and* 1 *to be the constant empty-history operations of Theorem 44.7. Define* $\mathsf{FieldSingletonNonZero}(h)$ *by*

$$\mathsf{FieldSingletonNonZero}(h) \iff A_{\mathsf{e}}(h) \wedge \mathsf{hsame}(h, \mathsf{e0}(\mathsf{e})).$$

The proof-indexed inverse operation is total on explicit witnesses: $\mathsf{inv}(h, p) := \mathsf{e}$ *for* $p : \mathsf{FieldSingletonNonZero}(h)$*. Theorem 45.33 shows that this witness domain is empty on the singleton carrier.*

Theorem 45.3 (Field singleton carrier visible exclusion)**.** *For every history* $p : \mathsf{BHist}$*, neither visible-headed endpoint lies in the singleton carrier:*

$$\mathsf{FieldSingletonCarrier}(\mathsf{e0}(p)) \to \bot, \qquad \mathsf{FieldSingletonCarrier}(\mathsf{e1}(p)) \to \bot.$$

Proof. Unfold the singleton carrier as equality with e. The closed inductive constructors of BHist separate both $\mathsf{e0}(p)$ and $\mathsf{e1}(p)$ from e, so either carrier witness is impossible. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier visible absurd`

Theorem 45.4 (Field singleton carrier constructor partition)**.** *Every history endpoint is either the empty singleton carrier endpoint, or it has a visible constructor whose singleton carrier witness is impossible:*

$$(h = \mathsf{e} \wedge \mathsf{FieldSingletonCarrier}(h)) \vee (\exists p.\ h = \mathsf{E0}(p) \wedge (\mathsf{FieldSingletonCarrier}(h) \to \bot)) \vee (\exists p.\ h = \mathsf{E1}(p) \wedge (\mathsf{FieldSingletonC$$

Proof. Split on the closed generated-history constructors. The empty branch is carried by reflexivity of hsame at e. In the two visible branches, constructor separation from e turns any singleton carrier witness into a contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier empty or visible absurd`

Theorem 45.5 (Field singleton classifier visible endpoint exclusion)**.** *For histories $p, q, k : \mathsf{BHist}$, no singleton classifier row can have a visible-headed endpoint on either side:*

$$\begin{aligned} &\rho_{\mathsf{e}}(\mathsf{e0}(p), k) \to \bot, \qquad && \rho_{\mathsf{e}}(\mathsf{e1}(p), k) \to \bot, \\ &\rho_{\mathsf{e}}(k, \mathsf{e0}(q)) \to \bot, \qquad && \rho_{\mathsf{e}}(k, \mathsf{e1}(q)) \to \bot. \end{aligned}$$

Proof. Unfold the singleton classifier into its two carrier fields and classifier comparison. If the visible endpoint is on the left, its left carrier field is one of the two impossible carrier witnesses from Definition 45.3. If the visible endpoint is on the right, the right carrier field gives the same contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier visible endpoint absurd`

Theorem 45.6 (Field singleton classifier empty endpoints)**.** *For histories $h, k : \mathsf{BHist}$, the singleton field classifier satisfies*

$$\rho_{\mathsf{e}}(h, k) \iff h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction unpacks the two carrier fields of $\rho_{\mathsf{e}}(h, k)$; these are precisely the endpoint comparisons $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$.

Conversely, the two endpoint comparisons supply the singleton carrier fields. Composing $h\mathsf{hsamee}$ with the symmetry of $k\mathsf{hsamee}$ gives $h\mathsf{hsame}k$, completing the classifier package. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier empty endpoints iff`

Theorem 45.7 (Field singleton carrier hsame empty transport)**.** *For histories $h, k : \mathsf{BHist}$ with $h\mathsf{hsame}k$, singleton carrier membership of h is exactly empty-endpoint classification of k:*

$$\mathsf{FieldSingletonCarrier}(h) \iff k\mathsf{hsamee}.$$

Proof. Unfold singleton carrier membership as $h\mathsf{hsamee}$. The forward direction composes $k\mathsf{hsame}h$ with $h\mathsf{hsamee}$. The reverse direction composes $h\mathsf{hsame}k$ with $k\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier hsame empty transport iff`

Theorem 45.8 (Field singleton classifier hsame empty transport)**.** *For histories $h, h', k, k' : \mathsf{BHist}$ with $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, the transported singleton classifier on h', k' is equivalent to the three empty-endpoint and comparison facts on h, k:*

$$\mathsf{FieldSingletonClassifier}(h', k') \iff h\mathsf{hsame}\mathsf{e} \wedge k\mathsf{hsame}\mathsf{e} \wedge h\mathsf{hsame}k.$$

Proof. In the forward direction, transport the two classifier carrier fields along $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, and transport the classifier comparison $h'\mathsf{hsame}k'$ back to $h\mathsf{hsame}k$. Conversely, transport the two empty-endpoint facts to h' and k', and compose $h'\mathsf{hsame}h$, $h\mathsf{hsame}k$, and $k\mathsf{hsame}k'$ to obtain the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier hsame empty transport iff`

Theorem 45.9 (Field singleton classifier hsame empty endpoints)**.** *For histories $h, h', k, k' : \mathsf{BHist}$ with $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, the singleton classifier on h, k is equivalent to the three transported empty-endpoint and comparison facts on h', k':*

$$\mathsf{FieldSingletonClassifier}(h, k) \iff h'\mathsf{hsame}\mathsf{e} \wedge k'\mathsf{hsame}\mathsf{e} \wedge h'\mathsf{hsame}k'.$$

Proof. The forward direction transports the two singleton-carrier endpoints from h, k to h', k' and composes the classifier comparison through the two hsame witnesses. Conversely, transport the endpoints back to h, k and compose $h\mathsf{hsame}h'$, $h'\mathsf{hsame}k'$, and $k'\mathsf{hsame}k$ to rebuild the singleton classifier on h, k. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier hsame empty endpoints iff`

Theorem 45.10 (Field singleton classifier continuation commutativity closure)**.** *For histories $P, Q, L, R : \mathsf{BHist}$, if the singleton field classifier relates P and Q, and the two displayed continuation results satisfy*

$$\mathsf{Cont}(P, Q, L) \qquad \textit{and} \qquad \mathsf{Cont}(Q, P, R),$$

then the singleton field classifier relates L and R.

Proof. The classifier witness identifies both P and Q with e. Transport the two continuation witnesses along these endpoint comparisons and compare them with the canonical continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. Thus both L and R are singleton-carrier endpoints. Transitivity through e supplies $L\mathsf{hsame}R$, completing the singleton classifier package. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation comm closed`

Theorem 45.11 (Field singleton classifier continuation commutativity iff)**.** *For histories $P, Q, L, R : \mathsf{BHist}$ with continuation ledgers*

$$\mathsf{Cont}(P, Q, L) \qquad \textit{and} \qquad \mathsf{Cont}(Q, P, R),$$

the two displayed continuation results are singleton-classified exactly when the two inputs are singleton-classified:

$$\mathsf{FieldSingletonClassifier}(L, R) \iff \mathsf{FieldSingletonClassifier}(P, Q).$$

Proof. In the first implication, the left result carrier identifies L with e. Transporting $\mathsf{Cont}(P, Q, L)$ to empty result and applying empty-result inversion identifies both P and Q with e, which reconstructs the input classifier.

Conversely, the input classifier identifies both inputs with e. Comparing each displayed continuation with the canonical empty continuation identifies both L and R with e; transitivity through e supplies $L\mathsf{hsame}R$ and hence the result classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation comm iff`

Theorem 45.12 (Field singleton carrier continuation closure)**.** *For histories $P, Q, R : \mathsf{BHist}$, if P and Q are singleton field carrier endpoints and $\mathsf{Cont}(P, Q, R)$, then R is a singleton field carrier endpoint.*

Proof. The carrier witnesses identify P and Q with e. Transport the displayed continuation along these endpoint comparisons and compare it with the canonical right-unit continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. The resulting determinacy identifies R with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation closed`

Theorem 45.13 (Field singleton carrier continuation endpoint split)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P, Q, R)$, the result R is a singleton field carrier endpoint exactly when both inputs are singleton field carrier endpoints:*

$$\mathsf{FieldSingletonCarrier}(R) \iff \mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSingletonCarrier}(Q).$$

Proof. If R is identified with e, transport the displayed continuation result to e and use empty-result inversion for $\mathsf{Cont}(P, Q, \mathsf{e})$ to identify both endpoints with e. Conversely, when P and Q are both identified with e, compare the displayed continuation with the canonical continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ to identify R with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation endpoint split iff`

Theorem 45.14 (Field singleton carrier continuation right endpoint iff)**.** *For histories $P, Q, R : \mathsf{BHist}$, if P is a singleton field carrier endpoint and $\mathsf{Cont}(P, Q, R)$, then R is a singleton field carrier endpoint exactly when Q is a singleton field carrier endpoint:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow \mathsf{Cont}(P, Q, R) \Rightarrow (\mathsf{FieldSingletonCarrier}(R) \iff \mathsf{FieldSingletonCarrier}(Q)).$$

Proof. The carrier witness for P identifies the left endpoint with e. The displayed continuation therefore has the same result as the right endpoint Q. Transporting singleton-carrier membership along this equality gives the forward implication, while singleton carrier continuation closure gives the reverse implication. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation right endpoint iff`

Theorem 45.15 (Field singleton carrier continuation left endpoint iff)**.** *For histories $P, Q, R : \mathsf{BHist}$, if Q is a singleton field carrier endpoint and $\mathsf{Cont}(P, Q, R)$, then R is a singleton field carrier endpoint exactly when P is a singleton field carrier endpoint:*

$$\mathsf{FieldSingletonCarrier}(Q) \Rightarrow \mathsf{Cont}(P, Q, R) \Rightarrow (\mathsf{FieldSingletonCarrier}(R) \iff \mathsf{FieldSingletonCarrier}(P)).$$

Proof. The carrier witness for Q identifies the right endpoint with e. The displayed continuation therefore has the same result as the left endpoint P. Transporting singleton-carrier membership along this equality gives the forward implication, while singleton carrier continuation closure gives the reverse implication. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation left endpoint iff`

Theorem 45.16 (Field singleton classifier continuation left endpoint transport)**.** *For histories $P, Q, R, S : \mathsf{BHist}$, if Q is a singleton field carrier endpoint and $\mathsf{Cont}(P, Q, R)$, then singleton classification from the result endpoint to S is equivalent to singleton classification from the left endpoint to S:*

$$\mathsf{FieldSingletonCarrier}(Q) \Rightarrow \mathsf{Cont}(P, Q, R) \Rightarrow \big(\mathsf{FieldSingletonClassifier}(R, S) \Longleftrightarrow \mathsf{FieldSingletonClassifier}(P, S)\big).$$

Proof. The carrier witness for Q identifies the continuation tail with e, so the continuation result is internally the same as P. Unfolding the singleton classifier on both sides reduces the comparison to transporting the left carrier field and the endpoint-sameness field along this result equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation left endpoint transport iff`

Theorem 45.17 (Field singleton classifier continuation right endpoint transport)**.** *For histories $P, Q, R, S : \mathsf{BHist}$, if P is a singleton field carrier endpoint and $\mathsf{Cont}(P, Q, R)$, then singleton classification from the result endpoint to S is equivalent to singleton classification from the right endpoint to S:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow \mathsf{Cont}(P, Q, R) \Rightarrow \big(\mathsf{FieldSingletonClassifier}(R, S) \Longleftrightarrow \mathsf{FieldSingletonClassifier}(Q, S)\big).$$

Proof. The carrier witness for P identifies the continuation source with e, so the continuation result is internally the same as Q. Unfolding the singleton classifier on both sides transports the source-carrier field and the endpoint-sameness field along this result equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation right endpoint transport iff`

Theorem 45.18 (Field singleton classifier continuation two-sided context transport)**.** *For histories $L, R, h, mid, out, S : \mathsf{BHist}$, if L and R are singleton field carrier endpoints, $\mathsf{Cont}(L, h, mid)$, and $\mathsf{Cont}(mid, R, out)$, then singleton classification from out to S is equivalent to singleton classification from h to S:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \mathsf{Cont}(L, h, mid) \Rightarrow \mathsf{Cont}(mid, R, out) \Rightarrow (\mathsf{FieldSingletonClass}$$

Proof. The right-context singleton endpoint transports the classifier on out back to the intermediate endpoint mid. The left-context singleton endpoint then transports the classifier on mid back to the exposed history h. Composing the two equivalences gives the displayed classifier equivalence. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation two sided context transport iff`

Theorem 45.19 (Field singleton carrier continuation visible result exclusion)**.** *For histories P, Q, r : BHist, if P and Q are singleton field carrier endpoints, then no continuation from P through Q can have visible result $\mathsf{E0}(r)$ or $\mathsf{E1}(r)$:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow \mathsf{FieldSingletonCarrier}(Q) \Rightarrow (\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

Proof. The two singleton carrier witnesses identify the continuation endpoints with e. Continuation result transport compares the displayed continuation with the right-unit continuation at e, forcing its result to be in the singleton carrier. Constructor separation rules out both visible histories $\mathsf{E0}(r)$ and $\mathsf{E1}(r)$ being internally the same as e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation visible result absurd`

Theorem 45.20 (Field singleton classifier continuation visible result exclusion)**.** *For histories P, Q, r : BHist, if the singleton field classifier relates P and Q, then no continuation from P through Q can have visible result $\mathsf{E0}(r)$ or $\mathsf{E1}(r)$:*

$$\mathsf{FieldSingletonClassifier}(P, Q) \Rightarrow (\mathsf{Cont}(P, Q, \mathsf{E0}(r)) \Rightarrow \mathsf{False}) \wedge (\mathsf{Cont}(P, Q, \mathsf{E1}(r)) \Rightarrow \mathsf{False}).$$

Proof. Unpack the classifier into singleton carrier witnesses for the two endpoints. The preceding carrier exclusion then applies to the same continuation and eliminates both visible result shapes. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation visible result absurd`

Theorem 45.21 (Field singleton classifier continuation endpoint split)**.** *For histories P, Q, R : BHist with $\mathsf{Cont}(P, Q, R)$, $\mathsf{FieldSingletonClassifier}(P, Q)$ is equivalent to $\mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSingletonCarrier}(Q) \wedge \mathsf{FieldSingletonCarrier}(R)$.*

Proof. The classifier gives the two input carrier fields, and singleton carrier continuation closure gives the result carrier. Conversely, the two input carrier fields identify both endpoints with e, so transitivity through e gives their classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation endpoint split iff`

Theorem 45.22 (Field singleton classifier continuation result classifier)**.** *For histories P, Q, R : BHist, if the singleton field classifier relates P and Q and $\mathsf{Cont}(P, Q, R)$, then the result R is classified with e.*

Proof. Apply singleton carrier continuation closure to the two carrier fields of the classifier witness and the displayed continuation. The result carrier, the reflexive carrier of e, and the same result carrier as the comparison through e assemble the singleton classifier package. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation result classifier`

Theorem 45.23 (Field singleton classifier continuation closure)**.** *For histories P, P', Q, Q', L, R : BHist, if the singleton field classifier relates P with P' and Q with Q', and if*

$$\mathsf{Cont}(P, Q, L) \qquad \textit{and} \qquad \mathsf{Cont}(P', Q', R),$$

then the singleton field classifier relates L and R.

Proof. The two classifier witnesses identify all four inputs with e. Transport each continuation witness along the corresponding pair of endpoint comparisons and compare the result with the canonical continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. Both L and R are therefore singleton-carried endpoints, and transitivity through e supplies $L\mathsf{hsame}R$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation closed`

Theorem 45.24 (Field singleton classifier visible-left exclusion)**.** *For histories* $p, q, h : \mathsf{BHist}$*, neither* $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{e0}(q)), h)$ *nor* $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{e1}(q)), h)$ *can hold.*

Proof. Unfold the singleton field classifier. In either case, the left carrier field identifies the displayed append endpoint with e. Empty-result inversion for append then forces the visible right component, either $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$, to be e. The constructors of BHist separate visible endpoints from the empty endpoint, so each case is impossible. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append visible left absurd`

Theorem 45.25 (Field singleton classifier visible-prefix exclusion)**.** *For histories* $p, q, h : \mathsf{BHist}$*, neither* $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e0}(p), q), h)$ *nor* $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{e1}(p), q), h)$ *can hold.*

Proof. Unfold the singleton field classifier. In either case, the left carrier field identifies the displayed append endpoint with e. Empty-result inversion for append then forces the visible prefix, either $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$, to be e. Constructor separation for BHist rules out both alternatives. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append visible prefix absurd`

Theorem 45.26 (Field singleton classifier visible right-prefix exclusion)**.** *For histories* $h, p, q : \mathsf{BHist}$*, neither* $\rho_{\mathsf{e}}(h, \mathsf{append}(\mathsf{e0}(p), q))$ *nor* $\rho_{\mathsf{e}}(h, \mathsf{append}(\mathsf{e1}(p), q))$ *can hold.*

Proof. Unfold the singleton field classifier. In either case, the right carrier field identifies the displayed append endpoint with e. Empty-result inversion for append then forces the visible prefix, either $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$, to be e. Constructor separation for BHist rules out both alternatives. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append visible right prefix absurd`

Theorem 45.27 (Field singleton carrier visible-head append exclusion)**.** *For histories* $h, k : \mathsf{BHist}$*, neither singleton carrier endpoint*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(\mathsf{e0}(h), k))$$

nor

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(\mathsf{e1}(h), k))$$

can hold.

Proof. In either branch, the singleton carrier identifies the displayed append history with e. Empty-result inversion for append then identifies the visible head $\mathsf{e0}(h)$ or $\mathsf{e1}(h)$ with e. The generated history constructors separate both visible heads from the empty history, so both alternatives are impossible. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append visible head absurd`

Theorem 45.28 (Field singleton carrier visible-tail append exclusion)**.** *For histories p, q : BHist, neither singleton carrier endpoint*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(p, \mathsf{e0}(q)))$$

nor

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(p, \mathsf{e1}(q)))$$

can hold.

Proof. In either branch, the singleton carrier identifies the displayed append history with e. Empty-result inversion for append then identifies the visible tail $\mathsf{e0}(q)$ or $\mathsf{e1}(q)$ with e. Constructor separation for generated histories rules out both visible tails. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append visible tail absurd`

Definition 45.29 (Classifier-complement singleton nonzero domain)**.** *For the same singleton carrier and classifier, define* $\mathsf{fieldSingletonEmptyNonZero}(h)$ *by*

$$\mathsf{fieldSingletonEmptyNonZero}(h) \iff (\rho_{\mathsf{e}}(h, \mathsf{e}) \to \bot).$$

The multiplication, unit, and proof-indexed inverse are the constant empty-history operations $\mathsf{fieldSingletonEmptyMul}(x, y) := \mathsf{e}$, $\mathsf{fieldSingletonEmptyOne} := \mathsf{e}$, *and* $\mathsf{fieldSingletonEmptyInv}(h, p) := \mathsf{e}$.

Lean4 (definition) `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero`

Theorem 45.30 (Singleton empty nonzero carrier complement)**.** *For every h : BHist, the proof-indexed singleton empty nonzero predicate is exactly the complement of the singleton carrier:*

$$\mathsf{fieldSingletonEmptyNonZero}(h) \iff (\mathsf{FieldSingletonCarrier}(h) \to \mathsf{False}).$$

Proof. For the forward direction, assume p : $\mathsf{fieldSingletonEmptyNonZero}(h)$ and a carrier witness for h. The singleton source is generated at e, so this carrier witness supplies the left endpoint comparison $h\mathsf{hsame}\mathsf{e}$.

The reflexive endpoint $\mathsf{ehsame}\mathsf{e}$ gives the target carrier at e. Together with the comparison from h to e, these data assemble the classifier package $\rho_{\mathsf{e}}(h, \mathsf{e})$, and p sends that package to False.

For the reverse direction, assume q : $\mathsf{FieldSingletonCarrier}(h) \to \mathsf{False}$ and a classifier witness $c : \rho_{\mathsf{e}}(h, \mathsf{e})$. Unpack c and extract its left carrier field for h. Applying q to that carrier field gives the required contradiction, so q is a witness of $\mathsf{fieldSingletonEmptyNonZero}(h)$ by Definition 45.29. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero empty endpoint complement iff`

Theorem 45.31 (Singleton empty nonzero hsame empty complement)**.** *For histories h and k with $h\mathsf{hsame}k$, the proof-indexed singleton empty nonzero predicate at h is exactly the complement of k being empty:*

$$\mathsf{fieldSingletonEmptyNonZero}(h) \iff (\mathsf{hsame}(k, \mathsf{e}) \to \mathsf{False}).$$

Proof. Transport an empty endpoint for k across $h\mathsf{hsame}k$ to obtain an empty endpoint for h, then apply the singleton nonzero obstruction. Conversely, unpack a forbidden classifier witness for h to get $h\mathsf{hsame}\mathsf{e}$, transport that endpoint back across $h\mathsf{hsame}k$, and contradict the displayed complement for k. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero hsame empty complement iff`

This identifies the concrete BHist Field$^{\uparrow}$ schema's proof-indexed nonzero domain with the carrier complement of the singleton empty-classifier instance.

Lemma 45.32 (Singleton raw nonzero endpoint obstruction)**.** *For every h : BHist, the two endpoint witnesses $h$hsamee and $h$hsamee0(e) imply* False.

Proof. Compose the endpoint $h$hsamee in reverse with the endpoint $h$hsamee0(e). This gives an equality ehsamee0(e) inside the closed BHist kernel.

The constructors e and e0(e) are distinct constructors of that inductive type, so constructor no-confusion eliminates the composed endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.field singleton nonzero absurd`

Theorem 45.33 (Singleton nonzero endpoint obstruction)**.** *For every h : BHist, a witness p : FieldSingletonNonZero(h) leads to contradiction.*

Proof. Unpack p into its singleton-carrier endpoint $h$hsamee and its zero-head endpoint $h$hsamee0(e). Lemma 45.32 applied to those two fields gives the contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNonZero absurd`

Theorem 45.34 (Singleton empty nonzero endpoint obstruction)**.** *For every h : BHist, if $h$hsamee and p : fieldSingletonEmptyNonZero(h), then contradiction follows.*

Proof. The endpoint $h$hsamee supplies the source side of the empty singleton classifier. The empty target is classified by reflexivity, and the same endpoint supplies the classifier equality from h to e. This builds the forbidden classifier input to p. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero empty endpoint absurd`

This block supplies the per-object certificate: traditional target empty inverse domain of the terminal proof-indexed field; BEDC source the concrete BHist singleton field schema; carrier A_{e}; classifier ρ_{e}; stability obligation nonzero contradiction; ledger no nonzero witness; core theorem Theorem 45.33; exactness predicate witness-domain emptiness; standard bridge terminal field; status ProofTarget.

Theorem 45.35 (Singleton nonzero forces the zero-headed empty endpoint)**.** *For every a : BHist, a witness of* FieldSingletonNonZero(a) *forces* ehsamee0(e).

Proof. Unpack the witness into the singleton-carrier endpoint $a$hsamee and the zero-headed endpoint $a$hsamee0(e). Symmetry of the first endpoint and transitivity with the second endpoint give the displayed classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNonZero empty e0 forced`

Theorem 45.36 (Singleton nonzero hsame target readback)**.** *For histories h, k : BHist, if $h$hsame$k$, then*

$$\mathsf{FieldSingletonNonZero}(h) \iff \mathsf{FieldSingletonCarrier}(k) \wedge k\mathsf{hsamee0}(\mathsf{e}).$$

Proof. Unpack the singleton nonzero witness into its carrier endpoint and its zero-headed endpoint. Symmetry and transitivity of hsame transport both endpoints across $h$hsame$k$. The reverse implication transports the two displayed target endpoints back along the same hsame witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNonZero hsame target readback iff`

Theorem 45.37 (Singleton empty-history field schema laws)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1, \mathsf{FieldSingletonNonZero}, \mathsf{inv})$$

satisfies the $\mathsf{Field}^{\uparrow}$ *schema laws over the concrete* BHist *singleton carrier.*

Proof. The commutative-ring fields are exactly the fields supplied by Theorem 44.7 for the displayed carrier, classifier, and constant empty-history operations.

The nonzero exclusion at 0 follows from Theorem 45.33 by taking $h = 0$.

Nonzero transport along ρ_{e} is vacuous. Given $\rho_{\mathsf{e}}(h, k)$ and $p : \mathsf{FieldSingletonNonZero}(h)$, Theorem 45.33 gives contradiction; from contradiction one obtains the required witness at k.

Inverse closure is direct because $\mathsf{inv}(h, p)$ is definitionally e, whose carrier endpoint is supplied by the singleton source. Inverse congruence is likewise direct because both displayed inverse endpoints are e, hence are related by ρ_{e}.

The left and right inverse laws have the same proof shape. Each law is indexed by a witness $p : \mathsf{FieldSingletonNonZero}(h)$, but both products are the constant empty-history product and $1 = \mathsf{e}$. The classifier witness is therefore the reflexive singleton endpoint, and no classical inverse is formed outside the proof-indexed domain. □

Lean4 ✓ `BEDC.Derived.FieldUp.singleton empty history field schema laws`

Theorem 45.38 (Singleton empty nonzero visible headed exclusion)**.** *For every history* h : BHist*, both visible-headed histories are in the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{e0}(h)) \wedge \mathsf{fieldSingletonEmptyNonZero}(\mathsf{e1}(h)).$$

Proof. For an e0-headed history, any forbidden classifier witness would include the carrier endpoint $\mathsf{e0}(h)\mathsf{hsamee}$, contradicting constructor disjointness for BHist. The e1-headed case is identical, using the disjointness of e1 from e. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero visible headed`

Theorem 45.39 (Singleton empty nonzero visible cases)**.** *Every history in the classifier-complement singleton nonzero predicate is visible-headed:*

$$\mathsf{fieldSingletonEmptyNonZero}(h) \Rightarrow (\exists t.\ h = \mathsf{e0}(t)) \vee (\exists t.\ h = \mathsf{e1}(t)).$$

Proof. Split on the concrete BHist constructor of h. In the empty case, the reflexive singleton-empty classifier contradicts the nonzero predicate. The e0 and e1 cases directly return their visible tail witnesses. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero visible cases`

Theorem 45.40 (Singleton empty nonzero visible append-right exclusion)**.** *For histories* p, q : BHist*, appending a visible right component gives histories in the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(p, \mathsf{e0}(q))) \wedge \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(p, \mathsf{e1}(q))).$$

Proof. For either visible right component, a forbidden classifier witness would include a carrier proof identifying the displayed append endpoint with e. Empty-result inversion for append then identifies the visible suffix with e, contradicting constructor separation for BHist. Thus each displayed appended endpoint lies in the complement of the singleton empty classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append visible right`

Theorem 45.41 (Singleton empty nonzero visible append-left exclusion)**.** *For histories p, q : BHist, appending after a visible left component gives histories in the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(\mathsf{e0}(p), q)) \wedge \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(\mathsf{e1}(p), q)).$$

Proof. For either visible left component, a forbidden classifier witness would identify the displayed append endpoint with e. Empty-result inversion for append then identifies the visible prefix with e, contradicting constructor separation for BHist. Hence both displayed endpoints lie in the complement of the singleton empty classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append visible left`

Theorem 45.42 (Singleton empty-classifier append split)**.** *For histories p, q, h : BHist, the singleton empty classifier $\rho_{\mathsf{e}}(\mathsf{append}(p, q), h)$ holds exactly when $p$hsamee, $q$hsamee, and h lies in the singleton carrier A_{e}.*

Proof. Unfold the classifier. In the forward direction, $\rho_{\mathsf{e}}(\mathsf{append}(p, q), h)$ supplies $\mathsf{append}(p, q)$hsamee and the carrier witness for h. Empty-result inversion for append splits the first endpoint into $p$hsamee and $q$hsamee.

Conversely, from $p$hsamee and $q$hsamee, append empty-result inversion reconstructs $\mathsf{append}(p, q)$hsamee. Together with the given carrier witness for h, transitivity through e supplies the classifier equality from $\mathsf{append}(p, q)$ to h. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier append split empty iff`

Theorem 45.43 (Singleton empty-classifier append right cancellation)**.** *For histories P, Q, R : BHist, if P lies in the singleton empty carrier, then adding P as a common right append context preserves and reflects the singleton empty classifier:*

$$\mathsf{fieldSingletonEmptyCarrier}(P) \Rightarrow (\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(Q, P), \mathsf{append}(R, P)) \Longleftrightarrow \mathsf{fieldSingletonEmp}$$

Proof. In the forward direction, the classifier supplies empty carrier witnesses for both appended endpoints. Empty-result inversion for append splits these witnesses into Qhsamee, Phsamee, and Rhsamee; the witnesses for Q and R rebuild the base classifier, with transitivity through e giving the endpoint comparison.

For the reverse direction, combine the supplied carrier witness for P with the two carrier fields of $\mathsf{fieldSingletonEmptyClassifier}(Q, R)$. Empty-result inversion then rebuilds the carrier witnesses for $\mathsf{append}(Q, P)$ and $\mathsf{append}(R, P)$, and transitivity through e compares the two appended endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier append right cancel iff`

Theorem 45.44 (Field singleton classifier append left cancellation)**.** *For histories P, Q, R : BHist, if P lies in the singleton carrier A_{e}, then adding P as a common left append context preserves and reflects the singleton field classifier:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow \big(\mathsf{FieldSingletonClassifier}(\mathsf{append}(P, Q), \mathsf{append}(P, R)) \Longleftrightarrow \mathsf{FieldSingletonClassifier}(Q, R)\big)$$

Proof. In the forward direction, the two appended endpoints are singleton-carried. Empty-result inversion for append splits the left and right endpoint carriers, recovering $Q\mathsf{hsamee}$ and $R\mathsf{hsamee}$. These two carrier witnesses, together with transitivity through e, rebuild $\mathsf{FieldSingletonClassifier}(Q, R)$.

For the reverse direction, the carrier witness for P and the endpoint carriers in $\mathsf{FieldSingletonClassifier}(Q, R)$ rebuild singleton carrier witnesses for $\mathsf{append}(P, Q)$ and $\mathsf{append}(P, R)$. Since both appended endpoints are carried to e, transitivity through e supplies the appended classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append left cancel iff`

Theorem 45.45 (Field singleton classifier multiplication endpoint)**.** *For histories h, k, out : BHist, classifying the singleton-empty field product endpoint against out is exactly the assertion that out is history-same to* e*:*

$$\rho_{\mathsf{e}}(\mathsf{FieldSingletonMul}(h, k), out) \Longleftrightarrow \mathsf{hsame}(out, \mathsf{e}).$$

Proof. The multiplication endpoint is definitionally e. In the forward direction, the classifier contains the carrier witness for the output endpoint. Conversely, a witness $\mathsf{hsame}(out, \mathsf{e})$ gives the output carrier, the reflexive source carrier at e, and by symmetry the classifier comparison from e to out. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier mul empty endpoint iff`

Theorem 45.46 (Field singleton addition endpoint classifier)**.** *For histories h, k, out : BHist, classifying the singleton-empty field addition endpoint against out is exactly the assertion that out is history-same to* e*:*

$$\rho_{\mathsf{e}}(\mathsf{FieldSingletonAdd}(h, k), out) \Longleftrightarrow \mathsf{hsame}(out, \mathsf{e}).$$

Proof. The addition endpoint is definitionally e. Forward classification exposes the carrier witness for out. Conversely, a witness $\mathsf{hsame}(out, \mathsf{e})$ gives the output carrier, the reflexive source carrier at e, and the symmetric classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAdd endpoint classifier iff`

Theorem 45.47 (Field singleton neg endpoint classifier iff)**.** *For histories h, out : BHist, classifying the singleton-empty field negative endpoint against out is exactly the assertion that out is history-same to* e*:*

$$\rho_{\mathsf{e}}(\mathsf{FieldSingletonNeg}(h), out) \Longleftrightarrow \mathsf{hsame}(out, \mathsf{e}).$$

Proof. The negative endpoint is definitionally e. Forward classification exposes the carrier witness for out. Conversely, a witness $\mathsf{hsame}(out, \mathsf{e})$ gives the output carrier, the reflexive source carrier at e, and transitivity through e gives the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNeg endpoint classifier iff`

Theorem 45.48 (Field singleton concrete operation-table collapse)**.** *In the concrete singleton* $\mathsf{Field}^{\uparrow}$ *instance over* BHist *with carrier* $A_{\mathsf{e}} = \mathsf{FieldSingletonCarrier}$*, classifier* $\rho_{\mathsf{e}} = \mathsf{FieldSingletonClassifier}$*, and concrete operations* $\mathsf{FieldSingletonAdd}$*,* $\mathsf{FieldSingletonMul}$*,* $\mathsf{FieldSingletonNeg}$*,* $\mathsf{FieldSingletonZero}$*,* $\mathsf{FieldSingletonOne}$*, and* $\mathsf{FieldSingletonInv}$*, every displayed operation endpoint lies in* A_{e}*. Moreover, for every* $out : \mathsf{BHist}$ *the classifier rows are exactly the singleton empty-history collapse:*

$$\begin{aligned}
\rho_{\mathsf{e}}(\mathsf{FieldSingletonAdd}(h,k), out) &\iff \mathsf{hsame}(out, \mathsf{e}),\\
\rho_{\mathsf{e}}(\mathsf{FieldSingletonMul}(h,k), out) &\iff \mathsf{hsame}(out, \mathsf{e}),\\
\rho_{\mathsf{e}}(\mathsf{FieldSingletonNeg}(h), out) &\iff \mathsf{hsame}(out, \mathsf{e}),\\
\rho_{\mathsf{e}}(\mathsf{FieldSingletonZero}, out) &\iff \mathsf{hsame}(out, \mathsf{e}),\\
\rho_{\mathsf{e}}(\mathsf{FieldSingletonOne}, out) &\iff \mathsf{hsame}(out, \mathsf{e}),\\
\rho_{\mathsf{e}}(\mathsf{FieldSingletonInv}(h,p), out) &\iff \mathsf{hsame}(out, \mathsf{e}).
\end{aligned}$$

Here $h, k : \mathsf{BHist}$ *and* $p : \mathsf{FieldSingletonNonZero}(h)$*.*

Proof. The multiplication, addition, and negation rows are exactly Theorem 45.45, Theorem 45.46, and Theorem 45.47. These three rows also give carried endpoint closure, since each forward classifier proof contains the output carrier and each operation endpoint is definitionally the singleton empty history.

For zero and one, unfold the concrete singleton constants. Both endpoints are e, so membership in A_{e} is reflexivity of hsame. Classifying e against *out* unfolds to the two carrier witnesses together with the classifier comparison; the forward direction exposes $out\mathsf{hsamee}$, and the reverse direction uses reflexivity at e plus symmetry of $out\mathsf{hsamee}$.

For the proof-indexed inverse, unfold $\mathsf{FieldSingletonInv}(h,p)$. The proof parameter supplies only the domain witness for forming the inverse endpoint; the endpoint itself is again e. The same reflexivity and symmetry argument classifies it against *out*, and transitivity through e gives the displayed comparison in the classifier field. □

The ten-field fit is: traditional target terminal concrete field operation table; BEDC source BHist with the singleton $\mathsf{Field}^{\uparrow}$ operations; carrier A_{e}; classifier ρ_{e}; stability obligations endpoint classifier collapse and carried endpoint closure; ledger hidden constant empty endpoints and proof-indexed inverse witness; core theorem Theorem 45.48; exactness predicate classifier collapse to $\mathsf{hsame}(out, \mathsf{e})$; status $\mathsf{ProofTarget}$.

Theorem 45.49 (Field singleton classifier append right cancellation)**.** *For histories* $P, Q, R : \mathsf{BHist}$*, if* P *lies in the singleton carrier* A_{e}*, then appending* P *on the right of both classified endpoints does not change the singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow \big(\mathsf{FieldSingletonClassifier}(\mathsf{append}(Q,P), \mathsf{append}(R,P)) \iff \mathsf{FieldSingletonClassifier}(Q,R)\big)$$

Proof. In the forward direction, each appended endpoint is classified as empty. Empty-result inversion for append splits these endpoint witnesses into $Q\mathsf{hsamee}$, $P\mathsf{hsamee}$, and $R\mathsf{hsamee}$. The two endpoint witnesses for Q and R supply the singleton carrier fields, and transitivity through e supplies the classifier comparison between Q and R.

Conversely, use the carrier witness $P\mathsf{hsamee}$ to reduce each right suffix to the empty suffix. The append right-unit laws identify $\mathsf{append}(Q, \mathsf{e})$ with Q and $\mathsf{append}(R, \mathsf{e})$ with R. Transport the singleton carrier witnesses and the displayed classifier comparison along these right-unit identifications to rebuild the appended classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append right cancel iff`

Theorem 45.50 (Field singleton carrier append right cancellation)**.** *For histories $P, Q : \mathsf{BHist}$, if P lies in the singleton carrier A_{e}, then appending P on the right preserves and reflects singleton-carrier membership:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow (\mathsf{FieldSingletonCarrier}(\mathsf{append}(Q, P)) \Longleftrightarrow \mathsf{FieldSingletonCarrier}(Q)).$$

Proof. The singleton carrier is the assertion of history-sameness to e. In the forward direction, empty-result inversion for append splits $\mathsf{append}(Q, P)\mathsf{hsamee}$ and exposes the $Q\mathsf{hsamee}$ component. In the reverse direction, combine $Q\mathsf{hsamee}$ with the displayed carrier witness for P and rebuild the empty append witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append right cancel iff`

Theorem 45.51 (Field singleton carrier append right nonempty contradiction)**.** *For histories $h, k : \mathsf{BHist}$, if the appended history lies in the singleton carrier, then the right factor cannot be nonempty:*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(h, k)) \Rightarrow (\mathsf{hsame}(k, \mathsf{e}) \to \bot) \Rightarrow \bot.$$

Proof. The singleton carrier unfolds to $\mathsf{hsame}(\mathsf{append}(h, k), \mathsf{e})$. Empty-append inversion splits this into $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. The right component contradicts the displayed nonempty right-factor hypothesis. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append right nonempty absurd`

Theorem 45.52 (Field singleton carrier append left cancellation)**.** *For histories $P, Q : \mathsf{BHist}$, if P lies in the singleton carrier A_{e}, then appending P on the left preserves and reflects singleton-carrier membership:*

$$\mathsf{FieldSingletonCarrier}(P) \Rightarrow (\mathsf{FieldSingletonCarrier}(\mathsf{append}(P, Q)) \Longleftrightarrow \mathsf{FieldSingletonCarrier}(Q)).$$

Proof. The singleton carrier asserts history-sameness to e. In the forward direction, empty-result inversion for append splits $\mathsf{append}(P, Q)\mathsf{hsamee}$ and exposes the $Q\mathsf{hsamee}$ component. In the reverse direction, combine the carrier witness for P with $Q\mathsf{hsamee}$ and rebuild the empty append witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append left cancel iff`

Theorem 45.53 (Field singleton carrier append left-context semantic name certificate)**.** *For a singleton carrier context p, the predicate $h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(p, h))$ carries a semantic naming certificate whose classifier compares the contextualized endpoints by the field singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(p) \Longrightarrow \mathsf{SemanticNameCert}(h \mapsto \mathsf{FieldSingletonCarrier}(p{\cdot}h))(h \mapsto \mathsf{FieldSingletonCarrier}(p{\cdot}h))(h \mapsto$$

Proof. The empty history is inhabited after the left context because $p\mathsf{hsamee}$. The classifier fields are the singleton carrier evidence on both contextualized endpoints together with their hsame comparison. Reflexivity, symmetry, and transitivity are inherited from hsame, and carrier transport is the classified right endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append left context semanticNameCert`

Theorem 45.54 (Field singleton carrier append right-context semantic name certificate)**.** *For a singleton carrier context p, the predicate $h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(h, p))$ carries a semantic naming certificate whose classifier compares the contextualized endpoints by the field singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(p) \Longrightarrow \mathsf{SemanticNameCert}(h \mapsto \mathsf{FieldSingletonCarrier}(h{\cdot}p))(h \mapsto \mathsf{FieldSingletonCarrier}(h{\cdot}p))(h \mapsto$$

Proof. The empty history is inhabited because $\mathsf{e} \cdot p$ is the same history as p, and p is carried to e. The classifier equivalence laws are exactly the reflexive, symmetric, and transitive laws of hsame wrapped with the two contextual carrier witnesses. Carrier transport is obtained from the classifier evidence at the right endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append right context semanticNameCert`

Theorem 45.55 (Field singleton carrier append commutation)**.** *For histories $h, k : \mathsf{BHist}$, the singleton carrier is insensitive to the order of an append pair:*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(h, k)) \Longleftrightarrow \mathsf{FieldSingletonCarrier}(\mathsf{append}(k, h)).$$

Proof. The singleton carrier is history-sameness to e. Empty-append inversion splits $\mathsf{append}(h, k)\mathsf{hsamee}$ into $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. Reversing the two components rebuilds $\mathsf{append}(k, h)\mathsf{hsamee}$, and the converse is symmetric. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append comm iff`

Theorem 45.56 (Field singleton classifier append commutation congruence)**.** *For histories h, h', k, k' : BHist, two singleton classifier witnesses for (h, h') and (k, k') rebuild a classifier after commuting the appended endpoints:*

$$\mathsf{FieldSingletonClassifier}(h, h') \Rightarrow \mathsf{FieldSingletonClassifier}(k, k') \Rightarrow \mathsf{FieldSingletonClassifier}(\mathsf{append}(h, k), \mathsf{append}(k', h'))$$

Proof. Each classifier witness contains singleton-carrier evidence for both of its endpoints. Empty-append inversion rebuilds a singleton carrier for $\mathsf{append}(h, k)$ from the source endpoints and for $\mathsf{append}(k', h')$ from the target endpoints. Since both appended histories are carried to e, transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append comm congr`

Theorem 45.57 (Field singleton classifier append context cancellation)**.** *For histories L, R, Q, S : BHist, if both displayed contexts lie in the singleton carrier A_{e}, then appending L to the first endpoint and R to the second endpoint does not change the singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, Q), \mathsf{append}(R, S)) \Longleftrightarrow \mathsf{Fi}$$

Proof. In the forward direction, the appended endpoint carriers split by empty-result inversion for append. The right pieces give $Q\mathsf{hsamee}$ and $S\mathsf{hsamee}$, and transitivity through e rebuilds the base classifier.

Conversely, combine the two context carrier witnesses with the endpoint carrier fields in $\mathsf{FieldSingletonClassifier}(Q, S)$. Empty-result inversion for append rebuilds singleton carrier witnesses for the two appended endpoints, and transitivity through e supplies their classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append context cancel iff`

Theorem 45.58 (Field singleton classifier append context visible middle exclusion)**.** *For histories* L, R, S, p, q : BHist*, if* L *and* R *lie in the singleton carrier* A_{e}*, then a visible middle endpoint exposed after appending those singleton contexts cannot support a field singleton classifier:*

$$\begin{aligned}&\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \\ &(\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{E0}(p)), \mathsf{append}(R, S)) \to \mathsf{False}) \wedge \\ &(\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{E1}(p)), \mathsf{append}(R, S)) \to \mathsf{False}) \wedge \\ &(\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, S), \mathsf{append}(R, \mathsf{E0}(q))) \to \mathsf{False}) \wedge \\ &(\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, S), \mathsf{append}(R, \mathsf{E1}(q))) \to \mathsf{False}).\end{aligned}$$

Proof. Cancel the singleton contexts by Theorem 45.57. Each case exposes a visible endpoint in the underlying $\mathsf{FieldSingletonClassifier}$. The visible endpoint exclusion Theorem 45.5 then rules out the exposed $\mathsf{E0}$ and $\mathsf{E1}$ endpoint cases. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append context visible middle absurd`

Theorem 45.59 (Field singleton classifier append cross-context cancellation)**.** *For histories* L, R, Q, S : BHist*, if* L *and* R *lie in the singleton carrier* A_{e}*, then a singleton right suffix on the source and a singleton left prefix on the target preserve and reflect the base singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{FieldSingletonClassifier}(\mathsf{append}(Q, L), \mathsf{append}(R, S)) \Longleftrightarrow \mathsf{Fi}$$

Proof. In the forward direction, the two appended classifier carriers split by empty-result inversion for append. The source split exposes the carrier for Q, and the target split exposes the carrier for S; transitivity through e rebuilds the comparison between Q and S.

Conversely, combine the carrier witness for L with the source carrier in $\mathsf{FieldSingletonClassifier}(Q, S)$, and combine the carrier witness for R with the target carrier. These rebuild the two appended endpoint carriers, and transitivity through e supplies the appended classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append cross context cancel iff`

Theorem 45.60 (Field singleton classifier append endpoint split)**.** *For histories* L, Q, R, S : BHist*,* $\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, Q), \mathsf{append}(R, S))$ *is equivalent to the four carrier witnesses* $\mathsf{FieldSingletonCarrier}(L)$*,* $\mathsf{FieldSingletonCarrier}(Q)$*,* $\mathsf{FieldSingletonCarrier}(R)$*, and* $\mathsf{FieldSingletonCarrier}(S)$*.*

Proof. The two classifier carrier fields identify both append endpoints with e, and empty-result inversion for append splits these witnesses into the four component carriers. Conversely, the four component carrier witnesses rebuild the two appended carrier witnesses, and transitivity through e supplies their classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append endpoint split iff`

Theorem 45.61 (Field singleton classifier append pair equivalence)**.** *For histories* h, h', k, k' : BHist*, the singleton classifier between* $\mathsf{append}(h, k)$ *and* $\mathsf{append}(h', k')$ *is equivalent to the pair of component classifiers* $\mathsf{FieldSingletonClassifier}(h, h')$ *and* $\mathsf{FieldSingletonClassifier}(k, k')$*.*

Proof. In the forward direction, empty-result inversion for the two append carriers splits the source and target endpoints into four singleton-carrier witnesses; transitivity through e builds the two component classifiers. Conversely, the component classifiers give singleton-carrier witnesses for all four endpoints, which rebuild the two appended carriers and their comparison through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append pair iff`

Theorem 45.62 (Field singleton classifier append endpoint empty nonzero exclusion)**.** *For histories L, Q, R, S : BHist, a singleton classifier witness for* $\mathsf{append}(L, Q)$ *and* $\mathsf{append}(R, S)$ *rules out empty-nonzero evidence at each displayed endpoint:*

$$\begin{aligned}&\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, Q), \mathsf{append}(R, S))\\ \Longrightarrow\ &(\mathsf{fieldSingletonEmptyNonZero}(L) \to \mathsf{False}) \wedge (\mathsf{fieldSingletonEmptyNonZero}(Q) \to \mathsf{False})\\ &\quad \wedge (\mathsf{fieldSingletonEmptyNonZero}(R) \to \mathsf{False}) \wedge (\mathsf{fieldSingletonEmptyNonZero}(S) \to \mathsf{False}).\end{aligned}$$

Proof. The classifier supplies singleton-carrier witnesses for both appended endpoints. Empty-result inversion for append splits these into witnesses that L, Q, R, and S each equal e. The concrete empty-nonzero complement then excludes empty-nonzero evidence at each of the four endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append endpoint empty nonzero absurd`

Theorem 45.63 (Field singleton carrier empty append context)**.** *For histories L, R, h : BHist, if L and R lie in the singleton carrier A_{e}, then surrounding h by the displayed left and right append contexts does not change singleton-carrier membership:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R))) \Longleftrightarrow \mathsf{FieldSir}$$

Proof. In the forward direction, the singleton carrier witness identifies the nested append with e. Empty-result inversion for append first splits off L and then splits $\mathsf{append}(h, R)$, recovering $h\mathsf{hsamee}$.

Conversely, combine the carrier witness for h with the carrier witness for R to rebuild the inner empty append. Combining that result with the carrier witness for L rebuilds the displayed outer empty append. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append context empty iff`

Theorem 45.64 (Field singleton classifier empty append context)**.** *For histories L, R, h, k : BHist, if L and R lie in the singleton carrier A_{e}, then adding L on the left of the first endpoint and R on the right of the second endpoint preserves and reflects the singleton field classifier:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, h), \mathsf{append}(k, R)) \Longleftrightarrow \mathsf{Fie}$$

Proof. In the forward direction, the classifier supplies singleton-carrier witnesses for both appended endpoints. Empty-result inversion for append recovers $h\mathsf{hsamee}$ from the left endpoint and $k\mathsf{hsamee}$ from the right endpoint, and transitivity through e supplies the classifier comparison between h and k.

Conversely, the two carrier fields of $\mathsf{FieldSingletonClassifier}(h, k)$, together with the carrier witnesses for L and R, rebuild the two appended singleton-carrier witnesses. Since both appended endpoints are carried to e, transitivity through e gives the appended classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append context empty iff`

Theorem 45.65 (Field singleton classifier self-context split)**.** *For histories a, x, b : BHist, a singleton field classifier from the self-context append endpoint to b is exactly a singleton carrier witness for the repeated context and a singleton classifier from the middle history to b:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(a, \mathsf{append}(x, a)), b) \Longleftrightarrow \mathsf{FieldSingletonCarrier}(a) \wedge \mathsf{FieldSingletonClassifier}(x, b).$$

Proof. In the forward direction, the source carrier identifies $\mathsf{append}(a, \mathsf{append}(x, a))$ with e. Empty-result inversion for append first recovers the carrier for a, then splits the inner append and recovers the carrier for x. The target carrier field of the classifier and transitivity through e rebuild $\mathsf{FieldSingletonClassifier}(x, b)$.

Conversely, combine the carrier for a with the source carrier in $\mathsf{FieldSingletonClassifier}(x, b)$ to rebuild the inner empty append, and combine that result with the same context carrier to rebuild the outer source carrier. The target carrier is already supplied by the middle classifier, and both endpoints compare through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append self context split iff`

Theorem 45.66 (Field singleton classifier append common-context cancellation up to history-sameness)**.** *For histories $L, R, P, Q, a, b : \mathsf{BHist}$, if L and P lie in the singleton carrier and the displayed contexts satisfy $L\mathsf{hsame}R$ and $P\mathsf{hsame}Q$, then the nested contextual singleton classifier is equivalent to the classifier of the middle histories:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(P) \Rightarrow L\mathsf{hsame}R \Rightarrow P\mathsf{hsame}Q \Rightarrow (\mathsf{FieldSingletonClassifier}(\mathsf{append}(L$$

Proof. Transport the two contexts along the displayed history-sameness witnesses. In the forward direction, empty-result inversion for append first splits the outer empty endpoints and then splits the inner append endpoints, exposing $a\mathsf{hsame}\mathsf{e}$ and $b\mathsf{hsame}\mathsf{e}$. These endpoint carrier witnesses rebuild the base classifier. Conversely, combine the classifier endpoint carriers with the singleton carrier witnesses for the common contexts, and use empty-result inversion for append in reverse to rebuild the two nested singleton-carrier witnesses. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append hsame common context cancel iff`

Theorem 45.67 (Field singleton classifier append suffix cancellation)**.** *For histories $P, R, Q, S : \mathsf{BHist}$, if Q and S lie in the singleton carrier A_{e}, then appending them as right suffixes to the two classified endpoints preserves and reflects the singleton field classifier:*

$$\mathsf{FieldSingletonCarrier}(Q) \Rightarrow \mathsf{FieldSingletonCarrier}(S) \Rightarrow (\mathsf{FieldSingletonClassifier}(\mathsf{append}(P, Q), \mathsf{append}(R, S)) \Longleftrightarrow \mathsf{Fi}$$

Proof. In the forward direction, the classifier supplies singleton carrier witnesses for both appended endpoints. Empty-result inversion for append splits these witnesses and exposes singleton carrier witnesses for P and R, which rebuild the base classifier through e.

Conversely, the carrier witnesses for Q and S, together with the endpoint carriers in $\mathsf{FieldSingletonClassifier}(P, R)$ rebuild singleton carrier witnesses for the two appended endpoints. Both appended endpoints are therefore history-same to e, so transitivity through e supplies their classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append suffix cancel iff`

Theorem 45.68 (Singleton empty nonzero append right cancellation)**.** *For histories $P, Q : \mathsf{BHist}$, if P lies in the singleton empty carrier, then appending P on the right does not change the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyCarrier}(P) \Rightarrow (\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(Q, P)) \Longleftrightarrow \mathsf{fieldSingletonEmptyNonZero}(Q))$$

Proof. In the forward direction, suppose Q were classified against e. Combining that endpoint witness with the displayed carrier witness for P rebuilds a singleton empty classifier witness for $\mathsf{append}(Q, P)$, contradicting the appended complement predicate.

Conversely, any classifier witness for $\mathsf{append}(Q, P)$ supplies $\mathsf{append}(Q, P)\mathsf{hsame}\mathsf{e}$. Empty-result inversion for append extracts $Q\mathsf{hsame}\mathsf{e}$, and this reconstructs the forbidden classifier witness for Q against e. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append right cancel iff`

Theorem 45.69 (Singleton empty nonzero append left cancellation)**.** *For histories P, Q : BHist, if P lies in the singleton empty carrier, then appending P on the left does not change the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyCarrier}(P) \Rightarrow (\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(P, Q)) \Longleftrightarrow \mathsf{fieldSingletonEmptyNonZero}(Q))$$

Proof. The forward implication turns a forbidden classifier witness for Q into one for $\mathsf{append}(P, Q)$ by pairing the carrier witness for P with the empty endpoint witness for Q.

For the reverse implication, a classifier witness for $\mathsf{append}(P, Q)$ identifies the append result with e. Empty-result inversion splits this into carrier witnesses for both factors, and the right factor witness rebuilds the forbidden singleton empty classifier for Q. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append left cancel iff`

Theorem 45.70 (Singleton empty nonzero continuation left endpoint cancellation)**.** *For histories P, Q, R : BHist, if Q lies in the singleton empty carrier and $\mathsf{Cont}(P, Q, R)$, then the classifier-complement singleton nonzero predicate is unchanged from P to R:*

$$\mathsf{fieldSingletonEmptyCarrier}(Q) \Rightarrow \mathsf{Cont}(P, Q, R) \Rightarrow (\mathsf{fieldSingletonEmptyNonZero}(R) \Longleftrightarrow \mathsf{fieldSingletonEmptyNonZer}$$

Proof. For the forward implication, a forbidden classifier on P combines with the singleton empty carrier of Q and the continuation ledger to rebuild a forbidden classifier on R. Conversely, a forbidden classifier on R is transported back through the displayed continuation; append-empty inversion then exposes the forbidden classifier on P. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero continuation left endpoint cancel iff`

Theorem 45.71 (Singleton empty nonzero continuation commutativity)**.** *For histories $P, Q, left, right$: BHist, if*

$$\mathsf{Cont}(P, Q, left) \quad \textit{and} \quad \mathsf{Cont}(Q, P, right),$$

then the classifier-complement singleton nonzero predicate is invariant under exchanging the two continuation inputs:

$$\mathsf{fieldSingletonEmptyNonZero}(left) \Longleftrightarrow \mathsf{fieldSingletonEmptyNonZero}(right).$$

Proof. For one direction, suppose $left$ is singleton nonzero and $right$ were empty. Transport the displayed continuation for $right$ to an empty result and invert the empty continuation. Both inputs are empty, so the opposite continuation has empty result, contradicting singleton nonzero at $left$. The converse is the same argument with the two continuations exchanged. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero continuation comm iff`

Theorem 45.72 (Rat classifier singleton-empty nonzero context endpoints)**.** *For histories L, R, h, k : BHist, any rational history classifier between h and k forces both contextual endpoints to satisfy the singleton-empty nonzero predicate:*

$$\mathsf{RatHistoryClassifier}(h, k) \Longrightarrow \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, h)) \wedge \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(R, k)$$

Proof. The rational classifier supplies the endpoint facts that neither h nor k is empty. To refute a singleton-empty classifier on $\mathsf{append}(L, h)$, invert the append-empty carrier equation; its right endpoint says $h = \mathsf{e}$, contradicting the rational endpoint fact. The right contextual endpoint is identical with R and k in place of L and h. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier fieldSingletonEmptyNonZero context endpoints`

Theorem 45.73 (Singleton empty nonzero append source or tail)**.** *For histories $P, Q : \mathsf{BHist}$, if the appended history is in the classifier-complement singleton nonzero predicate, then the source or the tail is also in that predicate:*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(P, Q)) \Rightarrow \mathsf{fieldSingletonEmptyNonZero}(P) \vee \mathsf{fieldSingletonEmptyNonZero}(Q).$$

Proof. Inspect the source history P. If $P = \mathsf{e}$, a forbidden classifier witness for Q rebuilds a forbidden classifier witness for $\mathsf{append}(P, Q)$ by the left unit law for append. If P is zero-headed or one-headed, BEDC constructor separation prevents P from lying in the singleton empty classifier, giving the source-side nonzero predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append source or tail`

Theorem 45.74 (Singleton empty nonzero append split)**.** *For histories $L, Q : \mathsf{BHist}$, the classifier-complement singleton empty nonzero predicate on an append splits exactly over the two factors:*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, Q)) \iff \mathsf{fieldSingletonEmptyNonZero}(L) \vee \mathsf{fieldSingletonEmptyNonZero}(Q).$$

Proof. For the forward implication, split on the left factor. If $L = \mathsf{e}$, a classifier witness for Q against e rebuilds the forbidden classifier witness for $\mathsf{append}(\mathsf{e}, Q)$. If L is visibly headed, constructor separation already makes L a singleton-empty nonzero witness.

For the reverse implication, any classifier witness for $\mathsf{append}(L, Q)$ against e splits by empty-result inversion for append into carrier witnesses for both L and Q. The chosen left or right nonzero witness rejects the corresponding carrier witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append split iff`

Theorem 45.75 (Singleton empty nonzero continuation endpoint split)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P, Q, R)$, the classifier-complement singleton empty nonzero predicate on the continuation result is exactly the disjunction of the corresponding endpoint predicates:*

$$\mathsf{fieldSingletonEmptyNonZero}(R) \iff \mathsf{fieldSingletonEmptyNonZero}(P) \vee \mathsf{fieldSingletonEmptyNonZero}(Q).$$

Proof. Unfold the continuation ledger as $R = \mathsf{append}(P, Q)$. In the forward direction, split on the concrete shape of P: the empty case transfers the forbidden classifier to Q, while either visible head gives constructor separation at the left endpoint. In the reverse direction, a classifier of R against e splits by empty append inversion, and the selected endpoint nonzero witness rejects the matching empty component. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero continuation endpoint split iff`

Lemma 45.76 (Singleton empty nonzero continuation right endpoint cancellation)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P, Q, R)$, if the right endpoint Q is a singleton-empty carrier, then the classifier-complement singleton empty nonzero predicate on the continuation result is exactly the predicate on the source endpoint:*

$$\mathsf{fieldSingletonEmptyNonZero}(R) \iff \mathsf{fieldSingletonEmptyNonZero}(P).$$

Proof. The continuation ledger gives $R = \mathsf{append}(P, Q)$. The singleton-empty carrier on Q identifies the right endpoint with e, so the append endpoint reduces to P and the two classifier-complement predicates coincide. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero continuation right endpoint cancel iff`

Theorem 45.77 (Singleton empty nonzero continuation cycle tail obstruction)**.** *For histories* $h, k, tail, backTail : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(h, tail, k) \quad \textit{and} \quad \mathsf{Cont}(k, backTail, h)$$

form a continuation cycle, then $\mathsf{fieldSingletonEmptyNonZero}(tail)$ *is contradictory.*

Proof. The two continuations compose to a self-continuation of h. Right-unit uniqueness for continuation forces the composite tail to be empty, and empty-result inversion for append gives $tail\mathsf{hsamee}$. The singleton empty nonzero endpoint obstruction then contradicts the displayed nonzero witness on $tail$. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero continuation cycle tail absurd`

Theorem 45.78 (Singleton empty nonzero append context cancellation)**.** *For histories* $L, R, Q : \mathsf{BHist}$*, if* L *and* R *lie in the singleton empty carrier, then enclosing* Q *between them does not change the classifier-complement singleton nonzero predicate:*

$$\mathsf{fieldSingletonEmptyCarrier}(L) \Rightarrow \mathsf{fieldSingletonEmptyCarrier}(R) \Rightarrow (\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, \mathsf{append}($$

Proof. If the enclosed history is nonzero and Q were classified against e, then the empty-carrier witnesses for L and R rebuild a classifier witness for $\mathsf{append}(L, \mathsf{append}(Q, R))$ against e, contradicting the enclosed nonzero predicate.

Conversely, a classifier witness for $\mathsf{append}(L, \mathsf{append}(Q, R))$ against e splits by empty-result inversion for append. The inner split yields $\mathsf{fieldSingletonEmptyCarrier}(Q)$, which rebuilds the forbidden classifier witness for Q against e. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append context cancel iff`

Theorem 45.79 (Singleton empty nonzero two-sided context semantic certificate)**.** *For histories* $L, R : \mathsf{BHist}$*, the concrete two-sided context predicate*

$$h \longmapsto \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, \mathsf{append}(h, R)))$$

used as source, pattern, and ledger policy with classifier hsame *forms a semantic name certificate.*

Proof. The source is inhabited by choosing the visible middle history $\mathsf{e0}(\mathsf{e})$. If the resulting nested append were classified against e, empty-result inversion for the outer append and then the inner append would force $\mathsf{e0}(\mathsf{e})\mathsf{hsamee}$, contradicting the BEDC no-confusion law for visible histories. The classifier laws are the reflexive, symmetric, and transitive laws for hsame. Carrier transport eliminates the supplied history-sameness proof, leaving the same concrete two-sided context predicate; pattern and ledger soundness are identities because the three predicates coincide. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero append context semanticNameCert`

Theorem 45.80 (Classifier-complement singleton field schema laws)**.** *For the tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{fieldSingletonEmptyNonZero}, \mathsf{fieldSingletonEmptyMul}, \mathsf{fieldSingletonEmptyOne}, \mathsf{fieldSingletonEmptyInv})$$

the singleton carrier is inhabited by e*; the predicate* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{e})$ *is contradictory; the predicate is transported along* ρ_{e}*; the inverse of every proof-indexed element is again in* A_{e}*; and the left and right products with the proof-indexed inverse are classified as the empty-history unit.*

Proof. The carrier inhabitance clause is the reflexive endpoint $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, so e lies in A_{e}.

For contradiction at the unit, suppose $p : \mathsf{fieldSingletonEmptyNonZero}(\mathsf{e})$. The displayed classifier witness $\rho_{\mathsf{e}}(\mathsf{e}, \mathsf{e})$ is built from the two reflexive carrier endpoints and reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Applying p to this witness closes the contradiction.

For transport, assume $\rho_{\mathsf{e}}(h, k)$ and $p : \mathsf{fieldSingletonEmptyNonZero}(h)$. Given a classifier witness $\rho_{\mathsf{e}}(k, \mathsf{e})$, compose the classifier equality from $\rho_{\mathsf{e}}(h, k)$ with the equality from $\rho_{\mathsf{e}}(k, \mathsf{e})$ to obtain $\rho_{\mathsf{e}}(h, \mathsf{e})$. The predicate p then gives contradiction, hence the transported predicate holds at k.

Inverse closure is immediate because every inverse endpoint is definitionally e, which is in the singleton carrier. The two inverse laws are immediate for the same reason: both products are the constant empty-history product, the unit is e, and the required classifier witnesses are built from reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field singleton empty schema laws`

Theorem 45.81 (Singleton empty-history FieldUp schema certificate)**.** *The tuple from Definition 45.2, together with the law package from Theorem 45.37, supplies the carrier, source, pattern, classifier, stability, and ledger fields required by Definition 45.337 over the concrete* BHist *singleton source. Its ten-field certificate template is:*

1. *Traditional target: the singleton proof-indexed* $\mathsf{Field}^{\uparrow}$ *schema.*
2. *BEDC source:* BHist *with* A_{e}*.*
3. *Carrier choice:* A_{e}*.*
4. *Classifier choice:* ρ_{e}*.*
5. *Stability obligations: commutative-ring laws plus vacuous nonzero transport, inverse closure, and inverse laws.*
6. *Ledger: the singleton endpoint and the empty nonzero witness domain.*
7. *Core theorem list: Theorem 45.37 and the semantic certificate theorem for the same singleton carrier.*
8. *Exactness predicate: singleton endpoint classification together with the* SemanticNameCert *source, pattern, and ledger fields.*
9. *Standard bridge: the terminal proof-indexed schema.*
10. *Status: CheckedCert.*

Proof. The source and carrier fields are the singleton history source A_{e} from Definition 45.2. The classifier field is ρ_{e}, whose only endpoints are the empty histories admitted by A_{e}.

For the algebraic stability fields, use Theorem 44.7. It supplies the commutative-ring part for the constant empty-history operations on the same carrier and classifier.

For the field-specific clauses, apply Theorem 45.37. Nonzero transport, inverse closure, inverse congruence, and the left and right inverse laws all begin with a witness $p : \mathsf{FieldSingletonNonZero}(h)$.

Theorem 45.33 shows that no such witness exists, while the inverse operation itself still has the total proof-indexed form $\mathsf{inv}(h, p) := \mathsf{e}$. The ledger therefore records only the singleton endpoint and the empty witness domain, giving the required concrete BHist FieldUp naming certificate template. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyCarrier semanticNameCert`

Theorem 45.82 (Singleton empty nonzero semantic name certificate)**.** *The classifier-complement predicate $h \mapsto \mathsf{fieldSingletonEmptyNonZero}(h)$ carries a semantic naming certificate whose source, pattern, and ledger predicates are this same nonzero predicate and whose classifier is* hsame*.*

Proof. The visible endpoint $\mathsf{e0}(\mathsf{e})$ inhabits the predicate because a singleton empty classifier witness would identify a zero-headed history with e, contradicting constructor separation for BHist. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Carrier transport follows by substituting the endpoint sameness into the concrete classifier-complement predicate; pattern and ledger soundness are the identity maps for the same predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyNonZero semanticNameCert`

Theorem 45.83 (Field singleton carrier continuation visible-left exclusion)**.** *For histories $p, Q, R :$ BHist, no continuation whose left input is headed by a visible constructor can land in the singleton field carrier:*

$$(\mathsf{Cont}(\mathsf{E0}(p), Q, R) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \bot) \wedge (\mathsf{Cont}(\mathsf{E1}(p), Q, R) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \bot).$$

Proof. Transport a displayed continuation result through the singleton carrier to e. Empty-result inversion then identifies the visible left input with e, contradicting BEDC history no-confusion for E0 or E1. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation visible left absurd`

Theorem 45.84 (Field singleton carrier continuation visible-right exclusion)**.** *For histories $P, q, R :$ BHist, no continuation whose right input is headed by a visible constructor can land in the singleton field carrier:*

$$(\mathsf{Cont}(P, \mathsf{E0}(q), R) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \bot) \wedge (\mathsf{Cont}(P, \mathsf{E1}(q), R) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \bot).$$

Proof. Transport the displayed continuation result through the singleton carrier to e. Empty-result inversion then identifies the visible right input with e, contradicting BEDC history no-confusion for E0 or E1. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation visible right absurd`

Theorem 45.85 (Field singleton classifier continuation empty result iff)**.** *For histories $P, Q, R :$ BHist with $\mathsf{Cont}(P, Q, R)$, the singleton field classifier relates P and Q exactly when the continuation result R is singleton-classified with* e*:*

$$\mathsf{FieldSingletonClassifier}(P, Q) \iff \mathsf{FieldSingletonClassifier}(R, \mathsf{e}).$$

Proof. The forward implication is the continuation result classifier. Conversely, the result-side classifier identifies R with e. Transporting the displayed continuation to the empty result and applying empty-result inversion identifies both P and Q with e, which reconstructs the singleton classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation empty result iff`

Theorem 45.86 (Field singleton classifier continuation result-left iff)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P, Q, R)$, singleton classification of the two continuation inputs is equivalent to singleton classification of the result against the left input:*

$$\mathsf{FieldSingletonClassifier}(P, Q) \iff \mathsf{FieldSingletonClassifier}(R, P).$$

Proof. For the forward implication, the singleton carrier witnesses for P and Q transport the continuation result to the right-unit continuation at e. Thus R is singleton-carried, and it compares with P through e. Conversely, the result-left classifier identifies R with e. Transport the displayed continuation to empty result and apply empty-result inversion; this recovers singleton carrier witnesses for both P and Q, which rebuild the input classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation result left iff`

Theorem 45.87 (Field singleton classifier continuation result endpoint split)**.** *For histories $P, Q, R :$ BHist with $\mathsf{Cont}(P, Q, R)$, the result-side singleton classifier is equivalent to all three continuation endpoints being singleton carriers:*

$$\mathsf{FieldSingletonClassifier}(R, \mathsf{e}) \iff \mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSingletonCarrier}(Q) \wedge \mathsf{FieldSingletonCarrier}(R).$$

Proof. For the forward direction, the result classifier identifies R with e; transporting the continuation to empty result and inverting it identifies P and Q with e. For the reverse direction, the displayed carrier witness for R and the reflexive carrier of e directly rebuild the result-side singleton classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation result endpoint split iff`

Theorem 45.88 (Field singleton carrier continuation context empty iff)**.** *For histories $L, h, M, R, O :$ BHist with $\mathsf{Cont}(L, h, M)$ and $\mathsf{Cont}(M, R, O)$, the final result is a singleton carrier exactly when the three exposed factors are singleton carriers:*

$$\mathsf{FieldSingletonCarrier}(O) \iff \mathsf{FieldSingletonCarrier}(L) \wedge \mathsf{FieldSingletonCarrier}(h) \wedge \mathsf{FieldSingletonCarrier}(R).$$

Proof. For the forward direction, transport the second continuation result to e and invert the empty result; this gives singleton carriers for M and R. Transport the first continuation result from M to e and invert again, exposing singleton carriers for L and h. Conversely, singleton carriers for L and h transport the first continuation result to the singleton carrier, and the resulting middle carrier together with the carrier for R transports the second continuation result to the singleton carrier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier continuation context empty iff`

Theorem 45.89 (Field singleton classifier continuation append split)**.** *For histories $L, P, Q, R, T :$ BHist with $\mathsf{Cont}(P, Q, T)$, singleton classification of $\mathsf{append}(L, T)$ against R is equivalent to the four displayed endpoints lying in the singleton carrier:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, T), R) \iff \mathsf{FieldSingletonCarrier}(L) \wedge \mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSingletonCarrie}$$

Proof. The forward direction splits the empty appended endpoint into carrier witnesses for L and T. Transporting the continuation ledger to the empty result and inverting it gives carrier witnesses for P and Q, while the classifier itself supplies the carrier witness for R. Conversely, carrier witnesses for P and Q make the continuation result T singleton, and combining that with the carrier of L rebuilds the left appended endpoint. Both classifier endpoints then compare through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation append split iff`

Theorem 45.90 (Field singleton classifier continuation append context split)**.** *For histories* L, P, Q, T, R, S : BHist *with* $\mathsf{Cont}(P, Q, T)$*, singleton classification of* $\mathsf{append}(L, T)$ *against* $\mathsf{append}(R, S)$ *is equivalent to all five exposed endpoints lying in the singleton carrier:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, T), \mathsf{append}(R, S)) \iff \mathsf{FieldSingletonCarrier}(L) \wedge \mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSin}$$

Proof. The forward direction splits both appended classifier endpoints at the empty history. The left split gives singleton carriers for L and T; transporting the continuation result from T to e and applying empty-result inversion gives the carrier witnesses for P and Q. The right split gives the carrier witnesses for R and S. Conversely, the carrier witnesses for P and Q make T singleton through the continuation equation, so both appended endpoints are singleton and therefore compare through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation append context split iff`

Lemma 45.91 (Field singleton classifier continuation context empty result)**.** *For histories* L, R, P, Q, T : BHist*, if* L *and* R *are singleton carriers and*

$$\mathsf{Cont}(\mathsf{append}(L, P), \mathsf{append}(Q, R), T),$$

then singleton classification of the continuation result against e *is equivalent to singleton classification of the exposed middle endpoints:*

$$\mathsf{FieldSingletonClassifier}(T, \mathsf{e}) \iff \mathsf{FieldSingletonClassifier}(P, Q).$$

Proof. For the forward implication, transport the displayed continuation to the empty result and invert it. The two appended endpoints split at e, exposing singleton carriers for P and Q, and the two carriers compare through e. Conversely, singleton carriers for P and Q, together with the assumed singleton carriers for L and R, make both appended continuation endpoints empty; the continuation result is therefore singleton and is classified with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation context empty result iff`

Theorem 45.92 (Field singleton add-mul context continuation result carrier)**.** *For histories* L, R, a, b, c, d, out : BHist*, if* L *and* R *are singleton field carrier endpoints and*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingletonMul}(c, d), R), out),$$

then out *is a singleton field carrier endpoint.*

Proof. The singleton add and mul endpoints both unfold to e. The carrier witnesses identify L and R with e, so both append contexts reduce to the empty history. Determinacy against the right-unit continuation at e therefore identifies out with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddMul context continuation result carrier`

Theorem 45.93 (Field singleton add-mul context continuation result iff)**.** *For histories* L, R, a, b, c, d, out : BHist*, if* L *and* R *are singleton field carrier endpoints, then the add-mul contextual continuation exists exactly for singleton field carrier results:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingle}$$

Proof. The forward direction is the add-mul contextual carrier closure. Conversely, the singleton add and mul endpoints unfold to e, and the carrier witnesses for L and R reduce both contextual inputs to e. Transport the right-unit continuation at e along the supplied carrier witness for *out* to obtain the displayed continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddMul context continuation result iff`

Theorem 45.94 (Field singleton add-mul context continuation classifier)**.** *For histories* $L, R, L', R', a, b, c, d, a', b',$ BHist*, if all four context endpoints are singleton field carriers and the two add-mul contextual continuations*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingletonMul}(c, d), R), out)$$

and

$$\mathsf{Cont}(\mathsf{append}(L', \mathsf{FieldSingletonAdd}(a', b')), \mathsf{append}(\mathsf{FieldSingletonMul}(c', d'), R'), out')$$

hold, then $\mathsf{FieldSingletonClassifier}(out, out')$*.*

Proof. Apply the add-mul contextual carrier closure to both continuations. The two results are each hsame to e, so transitivity and symmetry of hsame assemble the singleton classifier between the results. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddMul context continuation classifier`

Theorem 45.95 (Field singleton mul-inv context continuation classifier)**.** *For histories* $L, R, L', R', a, b, c, d, out,$ BHist*, suppose the four context endpoints are singleton field carriers, and suppose* $p : \mathsf{FieldSingletonNonZero}(c)$ *and* $q : \mathsf{FieldSingletonNonZero}(d)$*. If the two contextual continuations*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonMul}(a, b)), \mathsf{append}(\mathsf{FieldSingletonInv}(c, p), R), out)$$

and

$$\mathsf{Cont}(\mathsf{append}(L', \mathsf{FieldSingletonMul}(a, b)), \mathsf{append}(\mathsf{FieldSingletonInv}(d, q), R'), out')$$

hold, then $\mathsf{FieldSingletonClassifier}(out, out')$*.*

Proof. The singleton multiplication and proof-indexed inverse endpoints both unfold to e. The carrier witnesses identify each contextual endpoint with e, so continuation determinacy against the right-unit continuation gives singleton carrier witnesses for both results. These two carrier witnesses compose through e to assemble the singleton classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonMulInv context continuation classifier`

Theorem 45.96 (Field singleton continuation result context)**.** *For histories L, R, P, Q, T : BHist, if L and R are singleton carriers and $\mathsf{Cont}(P, Q, T)$, then singleton classification of the two-sided context around the continuation result against e is equivalent to the two exposed continuation inputs being singleton carriers:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{append}(T, R)), \mathsf{e}) \iff \mathsf{FieldSingletonCarrier}(P) \wedge \mathsf{FieldSingletonCarrier}(Q).$$

Proof. For the forward implication, split the empty contextual endpoint twice, first at L and then at T and R. The resulting singleton carrier for T transports the continuation to empty result, whose inversion gives the two input carriers. Conversely, the two input carriers make the continuation result singleton; combining this carrier with the assumed carriers for L and R rebuilds the displayed contextual classifier against e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier continuation result context iff`

Theorem 45.97 (Field singleton classifier for mutual continuation cycle tails)**.** *For histories $h, k, tail, backTail$: BHist, if*

$$\mathsf{Cont}(h, tail, k) \quad \textit{and} \quad \mathsf{Cont}(k, backTail, h),$$

then $\mathsf{FieldSingletonClassifier}(tail, backTail)$.

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier mutual continuation cycle tails`

Proof. Mutual continuation tail cancellation identifies both *tail* and *backTail* with e. These two singleton carrier witnesses give the classifier fields, and transitivity through e gives the tail-to-tail history-sameness field. □

45.3 RatUp boundary for the singleton field schema

Theorem 45.98 (Singleton field schema is disjoint from the RatUp history carrier)**.** *For every h : BHist, if $A_{\mathsf{e}}(h)$ is the singleton source from Definition 45.2, then $\mathsf{RatHistoryCarrier}(h)$ implies False. Equivalently, no RatUp history-carrier endpoint is an endpoint of the singleton empty-history $\mathsf{Field}^{\uparrow}$ schema.*

Proof. Assume $A_{\mathsf{e}}(h)$ and $\mathsf{RatHistoryCarrier}(h)$. Unfolding the singleton carrier component gives the endpoint equality $h\mathsf{hsamee}$.

The RatUp carrier witness is separated from the empty denominator endpoint by Theorem 37.18. Applying that theorem to the carrier witness and the equality $h\mathsf{hsamee}$ gives the required contradiction. □

Theorem 45.99 (Singleton field schema is disjoint from RatUp classifier endpoints)**.** *For every h, k : BHist, if $A_{\mathsf{e}}(h)$ holds for the singleton source from Definition 45.2 and $\mathsf{RatHistoryClassifier}(h, k)$ holds, then False. Likewise, if $A_{\mathsf{e}}(k)$ and $\mathsf{RatHistoryClassifier}(h, k)$ hold, then False.*

Proof. Unfold the Rat history classifier. Its witness contains $\mathsf{RatHistoryCarrier}(h)$, $\mathsf{RatHistoryCarrier}(k)$, and the displayed relation $h\mathsf{hsame}k$.

For the left endpoint, assume $A_{\mathsf{e}}(h)$. Apply Theorem 45.98 to $A_{\mathsf{e}}(h)$ and the carrier witness $\mathsf{RatHistoryCarrier}(h)$.

For the right endpoint, assume $A_{\mathsf{e}}(k)$. Apply Theorem 45.98 to $A_{\mathsf{e}}(k)$ and the carrier witness $\mathsf{RatHistoryCarrier}(k)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier fieldSingleton empty endpoints absurd`

Theorem 45.100 (Rat ledger endpoints are disjoint from the singleton FieldUp schema)**.** *For every raw and visible histories* $\rho, v : \mathsf{BHist}$*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds, then both* $A_{\mathsf{e}}(\rho) \to \mathsf{False}$ *and* $A_{\mathsf{e}}(v) \to \mathsf{False}$ *hold, where* A_{e} *is the singleton source from Definition 45.2.*

Proof. Unpack $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$. Its raw component gives $\mathsf{RatHistoryCarrier}(\rho)$, so any assumption $A_{\mathsf{e}}(\rho)$ is contradictory by Theorem 45.98.

For the visible endpoint, apply Theorem 37.65 to the same ledger policy. This produces $\mathsf{RatHistoryCarrier}(v)$, and Theorem 45.98 again rules out $A_{\mathsf{e}}(v)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryLedgerPolicy fieldSingletonEmptyCarrier endpoints absurd`

Theorem 45.101 (Singleton append endpoint excludes a RatUp right factor)**.** *For all histories* $p, h : \mathsf{BHist}$*, if* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(p, h))$ *holds and* $\mathsf{RatHistoryCarrier}(h)$ *holds, then* False*.*

Proof. The singleton carrier on $\mathsf{append}(p, h)$ unfolds to the equality $\mathsf{append}(p, h) = \mathsf{e}$. The kernel append-empty inversion theorem splits this equality into $p = \mathsf{e}$ and $h = \mathsf{e}$. The right component gives $h\mathsf{hsamee}$, contradicting Theorem 37.18 for the displayed RatUp carrier witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyCarrier append ratHistoryCarrier absurd`

Theorem 45.102 (Singleton append endpoint excludes a RatUp left factor)**.** *For all histories* $p, h : \mathsf{BHist}$*, if* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(h, p))$ *holds and* $\mathsf{RatHistoryCarrier}(h)$ *holds, then* False*.*

Proof. The singleton carrier on $\mathsf{append}(h, p)$ unfolds to the equality $\mathsf{append}(h, p) = \mathsf{e}$. The kernel append-empty inversion theorem splits this equality into $h = \mathsf{e}$ and $p = \mathsf{e}$. The left component gives $h\mathsf{hsamee}$, contradicting Theorem 37.18 for the displayed RatUp carrier witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyCarrier append left ratHistoryCarrier absurd`

Theorem 45.103 (Field singleton append classifier excludes a RatUp history carrier)**.** *For all histories* $p, h, endpoint : \mathsf{BHist}$*, a singleton classifier whose left endpoint is either* $\mathsf{append}(p, h)$ *or* $\mathsf{append}(h, p)$ *is incompatible with* $\mathsf{RatHistoryCarrier}(h)$*:*

$$(\mathsf{FieldSingletonClassifier}(\mathsf{append}(p, h), endpoint) \to \mathsf{RatHistoryCarrier}(h) \to \mathsf{False}) \wedge (\mathsf{FieldSingletonClassifier}(\mathsf{append}(h$$

Proof. In the first orientation, the singleton classifier contains the carrier comparison $\mathsf{append}(p, h)\mathsf{hsamee}$. Append-empty inversion extracts $h\mathsf{hsamee}$, contradicting $\mathsf{RatHistoryCarrier}(h)$ by rational history nonemptiness. The second orientation is the same argument with the left factor selected by append-empty inversion. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier append RatHistoryCarrier absurd`

Theorem 45.104 (Singleton append classifier excludes RatUp classifier endpoints)**.** *For all histories* $L, R, h, k : \mathsf{BHist}$*, if* $\mathsf{RatHistoryClassifier}(h, k)$ *holds and*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, h), \mathsf{append}(R, k))$$

holds, then False*.*

Proof. The singleton classifier contains $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, h))$, hence the equality $\mathsf{append}(L, h) = \mathsf{e}$. The kernel append-empty inversion theorem splits this equality and gives $h\mathsf{hsamee}$.

The RatUp classifier endpoint nonemptiness theorem rejects $h\mathsf{hsamee}$ for any displayed $\mathsf{RatHistoryClassifier}(h, k)$ yielding the contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier append RatHistoryClassifier absurd`

Theorem 45.105 (Singleton append classifier excludes crossed RatUp carrier endpoints)**.** *For all histories* $L, R, h, k : \mathsf{BHist}$*, if* $\mathsf{RatHistoryCarrier}(h)$ *and* $\mathsf{RatHistoryCarrier}(k)$ *hold, then both displayed singleton-empty classifiers are impossible:*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, h), \mathsf{append}(R, k)) \to \mathsf{False}$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, k), \mathsf{append}(R, h)) \to \mathsf{False}.$$

Proof. In the first direction, the singleton classifier contains $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, h))$. Append-empty inversion extracts $h\mathsf{hsamee}$, contradicting the displayed $\mathsf{RatHistoryCarrier}(h)$ witness.

The crossed direction is identical with k in the appended right factor of the left endpoint: append-empty inversion extracts $k\mathsf{hsamee}$, which contradicts $\mathsf{RatHistoryCarrier}(k)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier cross RatHistoryCarrier absurd`

Theorem 45.106 (Singleton append classifier excludes RatUp ledger endpoints)**.** *For all histories* $L, R, \rho, v : \mathsf{BHist}$*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ *holds and*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \rho), \mathsf{append}(R, v))$$

holds, then False*.*

Proof. The singleton classifier contains the raw endpoint equality $\mathsf{append}(L, \rho) = \mathsf{e}$. Kernel append-empty inversion gives $\rho\mathsf{hsamee}$.

The RatUp ledger endpoint boundary theorem states that a displayed $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$ rejects $\rho\mathsf{hsamee}$. Applying that raw endpoint obstruction gives the contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier append RatHistoryLedgerPolicy absurd`

Traditional target terminal singleton field boundary; BEDC source singleton $\mathsf{BHist}\ \mathsf{Field}^{\uparrow}$ schema plus RatUp history carrier; carrier choices A_{e} and $\mathsf{RatHistoryCarrier}$; classifier choices ρ_{e} and Rat history classifier; ledger choices $\mathsf{RatHistoryLedgerPolicy}$; stability obligation nonempty denominator separation; raw and visible ledger endpoint exclusion; core theorems Theorem 45.98 and Theorem 45.99 and Theorem 45.100; exactness predicate no shared carrier, classifier, or ledger endpoint; standard bridge terminal schema is not the rational field; status ProofTarget.

Theorem 45.107 (Classifier-complement singleton FieldUp instance is RatUp-disjoint)**.** *For the concrete source* A_{e}*, classifier* ρ_{e}*, proof domain* $\mathsf{fieldSingletonEmptyNonZero}$*, and constant operations* $\mathsf{fieldSingletonEmptyMul}$*,* $\mathsf{fieldSingletonEmptyOne}$*, and* $\mathsf{fieldSingletonEmptyInv}$ *from Definition 45.29, the* $\mathsf{Field}^{\uparrow}$ *schema laws hold over the concrete* BHist *singleton carrier, and every endpoint admitted by* A_{e} *is disjoint from* $\mathsf{RatHistoryCarrier}$*.*

Its ten-field template is:

1. *Traditional target: one-element proof-indexed* $\mathsf{Field}^{\uparrow}$ *schema.*
2. *BEDC source:* BHist *singleton source plus RatUp boundary.*
3. *Carrier:* A_{e}.
4. *Classifier:* ρ_{e}.
5. *Stability obligations: Theorem 45.80.*
6. *Ledger: the singleton endpoint and empty nonzero witness domain.*
7. *Core theorem list: Theorem 45.80 and Theorem 45.98.*
8. *Exactness predicate: singleton endpoint classification plus RatUp-disjointness.*
9. *Standard bridge: terminal proof-indexed schema only.*
10. *Status:* ProofTarget.

Proof. First apply Theorem 45.80 to the displayed tuple

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{fieldSingletonEmptyNonZero}, \mathsf{fieldSingletonEmptyMul}, \mathsf{fieldSingletonEmptyOne}, \mathsf{fieldSingletonEmptyInv}).$$

This supplies the concrete law package over the singleton BHist carrier.

For the RatUp boundary, take any displayed endpoint witness $A_{\mathsf{e}}(h)$ and any witness $\mathsf{RatHistoryCarrier}(h)$. Applying Theorem 45.98 to these two witnesses gives False, so no admitted singleton endpoint is a RatUp history-carrier endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyCarrier RatHistoryCarrier disjoint certificate`

Theorem 45.108 (RatUp one-denominator witness blocks singleton coverage)**.** *Let* $d_1 := \mathsf{E1}(\mathsf{e})$. *Then* $\mathsf{RatHistoryCarrier}(d_1)$ *holds and* $A_{\mathsf{e}}(d_1) \to \mathsf{False}$. *Consequently any proposed coverage map*

$$\forall h : \mathsf{BHist}, \mathsf{RatHistoryCarrier}(h) \to A_{\mathsf{e}}(h)$$

is contradictory.

Proof. The empty history is unary by the kernel theorem unary_empty. Apply Theorem 37.21 with tail e. The reverse direction turns $\mathsf{UnaryHistory}(\mathsf{e})$ into the carrier witness $\mathsf{RatHistoryCarrier}(\mathsf{E1}(\mathsf{e}))$, namely $\mathsf{RatHistoryCarrier}(d_1)$.

Now assume $A_{\mathsf{e}}(d_1)$. Applying Theorem 45.98 to this singleton endpoint and the carrier witness above gives False. Thus the concrete RatUp endpoint d_1 cannot be covered by the singleton carrier.

For the displayed coverage map, evaluate the map at d_1 and the RatUp carrier witness. This produces $A_{\mathsf{e}}(d_1)$, and the previous paragraph turns that endpoint into contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat carrier non singleton boundary witness`

Theorem 45.109 (Classifier-complement RatUp one-denominator witness blocks singleton coverage)**.** *Let* $d_1 := \mathsf{E1}(\mathsf{e})$. *Then* $\mathsf{RatHistoryCarrier}(d_1)$ *holds and* $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \mathsf{False}$. *Consequently any proposed coverage map*

$$\forall h : \mathsf{BHist}, \mathsf{RatHistoryCarrier}(h) \to \mathsf{fieldSingletonEmptyCarrier}(h)$$

is contradictory.

Proof. The empty history is unary by the kernel theorem unary_empty. Apply Theorem 37.21 with tail e. The reverse direction turns UnaryHistory(e) into RatHistoryCarrier(E1(e)), so it supplies RatHistoryCarrier(d_1).

Now assume fieldSingletonEmptyCarrier(d_1). The theorem RatHistoryCarrier_not_empty rejects a RatUp carrier endpoint that is the empty singleton endpoint. Applying it to the RatUp carrier witness above and the singleton-carrier witness gives False.

For the displayed coverage map, evaluate the map at d_1 and the RatUp carrier witness. This produces fieldSingletonEmptyCarrier(d_1), and the previous paragraph turns that endpoint into contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat history carrier non singleton boundary witness`

Theorem 45.110 (RatUp classifier and ledger coverage are blocked by the singleton FieldUp boundary)**.** *Let* $d_1 :=$ E1(e)*. Then* RatHistoryClassifier(d_1, d_1) *and* RatHistoryLedgerPolicy(d_1, d_1) *hold, while* FieldSingletonClassifier(d_1, d_1) $\to$ False*. Consequently both a classifier coverage map*

$$\forall hk : \mathsf{BHist}, \mathsf{RatHistoryClassifier}(h, k) \to \mathsf{FieldSingletonClassifier}(h, k)$$

and a ledger endpoint coverage map into the singleton carrier

$$\forall \rho v : \mathsf{BHist}, \mathsf{RatHistoryLedgerPolicy}(\rho, v) \to \mathsf{FieldSingletonCarrier}(\rho) \wedge \mathsf{FieldSingletonCarrier}(v)$$

are contradictory.

Proof. Theorem 45.108 gives the RatUp carrier witness RatHistoryCarrier(d_1) and the obstruction FieldSingletonCarrier(d_1) $\to$ False.

Using hsame reflexivity at d_1, the carrier witness forms the diagonal classifier package RatHistoryClassifier(d_1, d_1) The same carrier witness together with the same reflexivity link forms the diagonal ledger package RatHistoryLedgerPolicy(d_1, d_1).

Unfold FieldSingletonClassifier(d_1, d_1). Any such classifier witness contains FieldSingletonCarrier(d_1) as its left endpoint component, and the carrier obstruction from the first paragraph turns that endpoint into False.

Now evaluate any displayed classifier coverage map at the diagonal RatUp classifier package. The result is FieldSingletonClassifier(d_1, d_1), which the previous paragraph rules out.

For a displayed ledger endpoint coverage map, evaluate it at the diagonal ledger package. The resulting conjunction contains FieldSingletonCarrier(d_1), so the same carrier obstruction again gives contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat singleton classifier ledger coverage obstruction`

Traditional target exclusion of the terminal singleton schema as a RatUp $\mathsf{Field}^{\uparrow}$ classifier and ledger certificate; BEDC source RatHistoryCarrier, RatHistoryClassifier, RatHistoryLedgerPolicy, and the singleton $\mathsf{Field}^{\uparrow}$ schema; exactness predicate failure of carrier, classifier, and ledger coverage; status ProofTarget.

Traditional target $\mathsf{Field}^{\uparrow}$(RatUp) carrier separation; BEDC source RatHistoryCarrier plus singleton BHist $\mathsf{Field}^{\uparrow}$ schema; carrier choices RatHistoryCarrier and A_{e}; classifier choices Rat history classifier and ρ_{e}; stability obligation positive-denominator nonemptiness; ledger positive one-denominator witness versus empty singleton endpoint; core theorem Theorem 45.108; exactness predicate failure of Rat-to-singleton carrier coverage; standard bridge terminal singleton is not rational field; status ProofTarget.

45.4 RatUp continuation closure for the FieldUp boundary

Theorem 45.111 (Rat history carrier append split)**.** *If the appended denominator* $\mathsf{append}(h, k)$ *lies in the rational history carrier, then either* h *or* k *lies in the rational history carrier.*

Proof. Convert the appended rational history carrier to a positive unary denominator witness by the rational carrier equivalence. The positive-denominator append factor-or-tail theorem places that witness on h or on k; converting the selected positive-denominator branch back through the same equivalence gives the corresponding rational history carrier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryCarrier append split`

Theorem 45.112 (Rat history classifier left append split)**.** *If the rational history classifier relates* $\mathsf{append}(h, k)$ *to a denominator* d*, then either* h *or* k *lies in the rational history carrier.*

Proof. Project the left endpoint carrier from the classifier witness. The rat history carrier append split then exposes the carried denominator branch on h or on k. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier append left split`

Theorem 45.113 (Rat denominators are closed under continuation)**.** *For all histories* d, e, r : BHist*, if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*, and* $\mathsf{Cont}(d, e, r)$*, then* $\mathsf{RatHistoryCarrier}(r)$*.*

Proof. Start with the right denominator carrier $\mathsf{RatHistoryCarrier}(e)$. Theorem 37.19 extracts a positive unary denominator witness for e, and Theorem 37.12 turns that witness into $\mathsf{UnaryHistory}(e)$.

Apply Theorem 37.23 to the left carrier $\mathsf{RatHistoryCarrier}(d)$ and the unary witness for e. This gives $\mathsf{RatHistoryCarrier}(\mathsf{append}(d, e))$ for the canonical BEDC continuation result.

Unfolding the displayed continuation witness identifies r with $\mathsf{append}(d, e)$. The appended carrier is therefore exactly the required carrier for r. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation carrier closure`

Theorem 45.114 (Rat history classifier continuation tail iff)**.** *For rational history carriers* d, e : BHist *and a displayed result* r : BHist*,*

$$\mathsf{Cont}(d, e, r) \iff \mathsf{RatHistoryClassifier}(r, \mathsf{append}(d, e)).$$

Proof. If $\mathsf{Cont}(d, e, r)$ holds, carrier closure gives $\mathsf{RatHistoryCarrier}(r)$. The same closure theorem applied to the canonical append continuation gives $\mathsf{RatHistoryCarrier}(\mathsf{append}(d, e))$, and continuation determinacy supplies $r\mathsf{hsameappend}(d, e)$. These three components are the Rat history classifier. Conversely, the classifier comparison component is exactly the continuation equation $\mathsf{Cont}(d, e, r)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier continuation tail iff`

Theorem 45.115 (Rat history carrier continuation result is nonempty)**.** *For all histories* d, e, r : BHist*, if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*, and* $\mathsf{Cont}(d, e, r)$*, then* r *is not internally the empty history:*

$$r\mathsf{hsamee} \to \mathsf{False}.$$

Proof. Apply Definition 45.113 to the two input carriers and the displayed continuation. This gives $\mathsf{RatHistoryCarrier}(r)$. The Rat carrier nonempty theorem then rejects any comparison $r\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryCarrier continuation result not empty`

Theorem 45.116 (Rat continuation results cannot be singleton-classified)**.** *For histories* d, d', e, e', r, r' : BHist*, if* $\mathsf{RatHistoryClassifier}(d, d')$ *and* $\mathsf{RatHistoryClassifier}(e, e')$ *hold, and if* $\mathsf{Cont}(d, e, r)$ *and* $\mathsf{Cont}(d', e', r')$ *display the two continuation results, then* $\mathsf{FieldSingletonClassifier}(r, r')$ *is impossible.*

Proof. The left classifier supplies rational carrier witnesses for d and e. The carrier on e exposes a positive unary denominator, hence a unary history. Rational denominators are closed under appending a unary denominator, so $\mathsf{append}(d, e)$ lies in the rational history carrier. Continuation determinacy identifies this canonical append result with the displayed result r, transporting the rational carrier to r. The singleton classifier would also force $r\mathsf{hsame}e$, contradicting non-emptiness of rational history carriers. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier continuation result FieldSingletonClassifier absurd`

Theorem 45.117 (Rat denominator continuation semigroup laws)**.** *For all histories* d, e, f, de, ef, ℓ, r : BHist*, if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{RatHistoryCarrier}(f)$*,* $\mathsf{Cont}(d, e, de)$*,* $\mathsf{Cont}(e, f, ef)$*,* $\mathsf{Cont}(de, f, \ell)$*, and* $\mathsf{Cont}(d, ef, r)$*, then*

$$\mathsf{RatHistoryCarrier}(de) \wedge \mathsf{RatHistoryCarrier}(ef) \wedge \mathsf{RatHistoryCarrier}(\ell) \wedge \mathsf{RatHistoryCarrier}(r) \wedge \ell\mathsf{hsame}r.$$

Proof. First apply Definition 45.113 to the continuations $\mathsf{Cont}(d, e, de)$ and $\mathsf{Cont}(e, f, ef)$. The three displayed input carriers therefore give $\mathsf{RatHistoryCarrier}(de)$ and $\mathsf{RatHistoryCarrier}(ef)$.

Next apply the same closure theorem to $\mathsf{Cont}(de, f, \ell)$ using the carrier just obtained for de and the original carrier for f. Apply it again to $\mathsf{Cont}(d, ef, r)$ using the original carrier for d and the carrier just obtained for ef. This yields the two outer result carriers.

Finally apply the FKernel continuation associativity theorem, in the form `BEDC.FKernel.Cont.cont_assoc_hs` to the four continuation witnesses. It identifies the two displayed bracketings of denominator continuation by $\ell\mathsf{hsame}r$, giving the full conjunction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation semigroup laws`
↪ variant `BEDC.Derived.FieldUp.RatHistoryCarrier continuation semigroup laws`

Theorem 45.118 (Rat denominator continuation commutativity)**.** *For all histories* d, e, r, s : BHist*, if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(e, d, s)$*, then* $\mathsf{RatHistoryClassifier}(r, s)$*.*

Proof. First apply Definition 45.113 to $\mathsf{Cont}(d, e, r)$ and to $\mathsf{Cont}(e, d, s)$. The two applications give $\mathsf{RatHistoryCarrier}(r)$ and $\mathsf{RatHistoryCarrier}(s)$.

Next read the input carriers through Definition 37.19 and then Definition 37.12. This extracts $\mathsf{UnaryHistory}(d)$ and $\mathsf{UnaryHistory}(e)$ from the two Rat denominator carriers.

Now apply the FKernel unary continuation commutativity theorem, `BEDC.FKernel.Unary.unary_continuati` to the two unary witnesses and the continuation witnesses $\mathsf{Cont}(d, e, r)$ and $\mathsf{Cont}(e, d, s)$. It yields $r\mathsf{hsame}s$.

The two result carriers together with this hsame comparison are exactly the components of $\mathsf{RatHistoryClassifier}(r, s)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation commutative classifier`

Theorem 45.119 (Rat denominator classifiers are closed under matching continuations)**.** *For all histories* d, e, u, v, r, s : BHist*, if* $\mathsf{RatHistoryClassifier}(d, e)$*,* $\mathsf{UnaryHistory}(u)$*,* $\mathsf{UnaryHistory}(v)$*,* $u\mathsf{hsame}v$*,* $\mathsf{Cont}(d, u, r)$*, and* $\mathsf{Cont}(e, v, s)$*, then* $\mathsf{RatHistoryClassifier}(r, s)$*.*

Proof. First apply Theorem 37.56 to the displayed classifier $\mathsf{RatHistoryClassifier}(d, e)$, the unary tails u, v, and the matching tail witness $u\mathsf{hsame}v$. This gives the rational history classifier for the canonical append results $\mathsf{append}(d, u)$ and $\mathsf{append}(e, v)$.

Next compare the displayed continuation results with those canonical append results. The constructor witnesses for $\mathsf{Cont}(d, u, \mathsf{append}(d, u))$ and $\mathsf{Cont}(e, v, \mathsf{append}(e, v))$, together with the premises $\mathsf{Cont}(d, u, r)$ and $\mathsf{Cont}(e, v, s)$, feed into Theorem 18.16. Thus the result histories are linked by $\mathsf{hsame}(\mathsf{append}(d, u), r)$ and $\mathsf{hsame}(\mathsf{append}(e, v), s)$.

Finally transport the canonical append classifier across the two result hsame links by Theorem 37.54. The transported classifier is exactly $\mathsf{RatHistoryClassifier}(r, s)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier matching continuation closed`

Theorem 45.120 (Rat history classifiers are right-closed under continuation)**.** *For all histories* $d, d', e, r, r' : \mathsf{BHist}$*, if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(d', e, r')$*, then* $\mathsf{RatHistoryClassifier}(r, r')$*.*

Proof. Unpack the rational history classifier into carrier witnesses for d and d', together with the denominator comparison $d\mathsf{hsame}d'$. The carrier closure theorem above applies to the two displayed continuations with the shared right carrier e, producing $\mathsf{RatHistoryCarrier}(r)$ and $\mathsf{RatHistoryCarrier}(r')$.

The comparison component follows from continuation respect for internal sameness: use $d\mathsf{hsame}d'$ on the left inputs and reflexivity $e\mathsf{hsame}e$ on the shared right input. This yields $r\mathsf{hsame}r'$, and the two carrier fields with this comparison assemble the target classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier continuation right closed`

Theorem 45.121 (Rat history classifiers are left-closed under continuation)**.** *For all histories* $e, d, d', r, r' : \mathsf{BHist}$*, if* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{Cont}(e, d, r)$*, and* $\mathsf{Cont}(e, d', r')$*, then* $\mathsf{RatHistoryClassifier}(r, r')$*.*

Proof. Unpack the rational history classifier into carrier witnesses for d and d', together with the comparison $d\mathsf{hsame}d'$. The carrier closure theorem applies to the two displayed continuations with the shared left carrier e, giving $\mathsf{RatHistoryCarrier}(r)$ and $\mathsf{RatHistoryCarrier}(r')$.

Continuation respects internal sameness on both endpoints. Use reflexivity $e\mathsf{hsame}e$ for the shared left context and the classifier comparison $d\mathsf{hsame}d'$ for the right endpoints. The resulting $r\mathsf{hsame}r'$ comparison, with the two result carriers, is exactly the rational history classifier for the displayed continuation results. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier continuation left closed`

Theorem 45.122 (Rat denominator ledgers are closed under matching continuations)**.** *For all histories* $\rho, v, u, u', r, s : \mathsf{BHist}$*, if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$*,* $\mathsf{UnaryHistory}(u)$*,* $\mathsf{UnaryHistory}(u')$*,* $u\mathsf{hsame}u'$*,* $\mathsf{Cont}(\rho, u, r)$*, and* $\mathsf{Cont}(v, u', s)$*, then* $\mathsf{RatHistoryLedgerPolicy}(r, s)$*.*

Proof. First apply the Rat ledger append-unary denominator closure theorem to $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$, the unary tail $\mathsf{UnaryHistory}(u)$, and the matching witness $u\mathsf{hsame}u'$. This gives the ledger policy for the canonical append endpoints $\mathsf{append}(\rho, u)$ and $\mathsf{append}(v, u')$.

Next compare the displayed continuation results with those canonical endpoints. The constructor witnesses for $\mathsf{Cont}(\rho, u, \mathsf{append}(\rho, u))$ and $\mathsf{Cont}(v, u', \mathsf{append}(v, u'))$, together with $\mathsf{Cont}(\rho, u, r)$ and $\mathsf{Cont}(v, u', s)$, feed into Theorem 18.16. Thus $\mathsf{append}(\rho, u)\mathsf{hsame}r$ and $\mathsf{append}(v, u')\mathsf{hsame}s$.

Finally transport the canonical append ledger across the two result links by the Rat ledger history-sameness transport theorem. The transported ledger is exactly $\mathsf{RatHistoryLedgerPolicy}(r, s)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryLedgerPolicy continuation closed`

Theorem 45.123 (Rat history ledger continuation results are nonempty)**.** *For all histories* $raw, visible, tailRaw, t$ BHist*, if* RatHistoryLedgerPolicy$(raw, visible)$*,* UnaryHistory$(tailRaw)$*,* $tailRaw$hsame$tailVisible$*,* Cont$(raw, tailRaw, r)$*, and* Cont$(visible, tailVisible, s)$*, then neither continuation result is internally empty:*

$$(r\mathsf{hsamee} \to \mathsf{False}) \wedge (s\mathsf{hsamee} \to \mathsf{False}).$$

Proof. The ledger continuation closure theorem gives RatHistoryLedgerPolicy(r, s). Its raw component is a Rat carrier for r, and its visible-carrier projection is a Rat carrier for s. Applying the Rat carrier nonempty theorem to these two carrier witnesses rules out $r$hsamee and $s$hsamee. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryLedgerPolicy continuation results not empty`

Theorem 45.124 (Rat denominator continuation external boundary and associativity laws)**.** *Rat denominator continuation satisfies the following external endpoint-classifier laws:*

1. RatHistoryCarrier(e) $\to$ False*;*
2. *for all histories* d, r : BHist*, if* RatHistoryCarrier(d) *and* Cont(d, e, r)*, then* RatHistoryClassifier(r, d)*;*
3. *for all histories* d, s : BHist*, if* RatHistoryCarrier(d) *and* Cont(e, d, s)*, then* RatHistoryClassifier(s, d)*;*
4. *for all histories* $d, e, f, de, ef, left, right$: BHist*, if* RatHistoryCarrier(d)*,* RatHistoryCarrier(e)*,* RatHistoryCarrier(f)*,* Cont(d, e, de)*,* Cont$(de, f, left)$*,* Cont(e, f, ef)*, and* Cont$(d, ef, right)$*, then* RatHistoryClassifier$(left, right)$*.*

Proof. For the carrier boundary, assume RatHistoryCarrier(e). Reflexivity of hsame at e gives hsame(e, e). Applying Definition 37.18 to these two witnesses yields False.

For the right external-boundary law, apply the continuation right-unit result theorem to Cont(d, e, r). It supplies the hsame link from the displayed result back to d. Transport the given rational-history carrier along that link to get the carrier at r, then combine the two endpoint carriers with the same hsame link to form RatHistoryClassifier(r, d).

For the left external-boundary law, use the continuation left-unit result theorem on Cont(e, d, s). The same carrier transport step gives RatHistoryCarrier(s), and the resulting hsame link classifies s against d.

For associativity, first apply Definition 45.113 to Cont(d, e, de) and Cont(e, f, ef). Apply it again to Cont$(de, f, left)$ and Cont$(d, ef, right)$, using the two intermediate carriers just obtained. This supplies the rational-history carriers at both outer bracketings.

Finally apply the kernel continuation associativity theorem `BEDC.FKernel.Cont.cont_assoc_hsame` to the four continuation witnesses. Its hsame comparison between $left$ and $right$, together with the two outer carriers, is exactly RatHistoryClassifier$(left, right)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation external boundary laws`

Theorem 45.125 (Rat denominator continuation unit endpoint is externally empty)**.** *For every history* u : BHist*, Rat denominator continuation forces any classifier unit endpoint to be externally empty:*

1. *if*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

then $u$hsamee*;*

2. *if*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

then $u\mathsf{hsamee}$*;*

3. *if* $\mathsf{RatHistoryCarrier}(u)$ *and either displayed classifier unit law holds for* u*, then* False*.*

Proof. For the right-unit endpoint, let $d_1 := \mathsf{E1}(\mathsf{e})$. The positive-denominator construction supplies $\mathsf{RatHistoryCarrier}(d_1)$, and the finite kernel supplies the canonical continuation $\mathsf{Cont}(d_1, u, \mathsf{append}(d_1, u))$.

Apply the assumed right-unit classifier law to this carried input and canonical continuation. The classifier comparison gives $\mathsf{append}(d_1, u)\mathsf{hsame}d_1$, which is the same as $\mathsf{append}(d_1, u)\mathsf{hsameappend}(d_1, \mathsf{e})$. Left append cancellation in the finite kernel removes the common prefix d_1 and yields $u\mathsf{hsamee}$.

For the left-unit endpoint, use the same carried witness d_1 and the canonical continuation $\mathsf{Cont}(u, d_1, \mathsf{append}(u, d_1))$. The assumed left-unit classifier law gives $\mathsf{append}(u, d_1)\mathsf{hsame}d_1$, equivalently $\mathsf{append}(u, d_1)\mathsf{hsameappend}(\mathsf{e}, d_1)$.

Right append cancellation removes the common suffix d_1 and yields $u\mathsf{hsamee}$.

For the carried-endpoint consequence, combine the relevant endpoint-forcing clause with the assumed carrier witness $\mathsf{RatHistoryCarrier}(u)$. Then apply Theorem 37.18 to the carrier witness and the resulting equality $u\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation unit endpoint empty`

Theorem 45.126 (Rat denominator canonical right-unit endpoint is empty)**.** *For every history* $u : \mathsf{BHist}$*, if continuation by* u *makes the fixed canonical denominator* $d_1 := \mathsf{E1}(\mathsf{e})$ *classify back to* d_1*, then* u *is externally empty:*

$$(\forall r : \mathsf{BHist}.\ \mathsf{Cont}(d_1, u, r) \to \mathsf{RatHistoryClassifier}(r, d_1)) \Longrightarrow u\mathsf{hsamee}.$$

Proof. Apply the assumed classifier law to the canonical continuation $\mathsf{Cont}(d_1, u, \mathsf{append}(d_1, u))$. The resulting classifier compares the continuation result with d_1. Transport the canonical continuation along this result comparison to obtain $\mathsf{Cont}(d_1, u, d_1)$, then apply right-unit uniqueness for continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator canonical right unit endpoint empty`

Theorem 45.127 (Rat denominator canonical left-unit endpoint is empty)**.** *For every history* $u : \mathsf{BHist}$*, if continuation from* u *into the fixed canonical denominator* $d_1 := \mathsf{E1}(\mathsf{e})$ *classifies back to* d_1*, then* u *is externally empty:*

$$(\forall r : \mathsf{BHist}.\ \mathsf{Cont}(u, d_1, r) \to \mathsf{RatHistoryClassifier}(r, d_1)) \Longrightarrow u\mathsf{hsamee}.$$

Proof. Apply the assumed classifier law to the canonical continuation $\mathsf{Cont}(u, d_1, \mathsf{append}(u, d_1))$. The classifier result gives a comparison with d_1. Transport the continuation result to d_1 and apply left-unit uniqueness for continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator canonical left unit endpoint empty`

Theorem 45.128 (Rat denominator two-sided unit law endpoint empty iff)**.** *For every history* $u : \mathsf{BHist}$*, the simultaneous right and left classifier-unit laws for Rat denominator continuation hold exactly when* u *is the empty endpoint:*

$$((\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)) \wedge (\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d$$

Proof. For the forward direction, project the right classifier-unit law from the displayed conjunction. Apply Theorem 45.350 to obtain $u\mathsf{hsamee}$.

Conversely, assume $u\mathsf{hsamee}$. Apply Theorem 45.350 to obtain the right classifier-unit law, and apply Theorem 45.352 to obtain the left classifier-unit law. Pair these two laws into the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation two sided unit law endpoint empty iff`

Theorem 45.129 (Rat denominator continuation unit endpoint uniqueness)**.** *For all histories* $u, v : \mathsf{BHist}$*, Rat denominator continuation determines every one-sided classifier-unit endpoint:*

1. *if both* u *and* v *satisfy the right-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, w, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

   for $w = u$ *and for* $w = v$*, then* $u\mathsf{hsame}v$*;*

2. *if both* u *and* v *satisfy the left-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(w, d, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

   for $w = u$ *and for* $w = v$*, then* $u\mathsf{hsame}v$*;*

3. *if* u *satisfies the right-unit classifier law and* v *satisfies the left-unit classifier law, then* $u\mathsf{hsame}v$*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation unit endpoint unique`

Proof. In the right/right case, apply Theorem 45.125 to the right-unit classifier law for u, obtaining $u\mathsf{hsamee}$. Apply the same theorem to the right-unit classifier law for v, obtaining $v\mathsf{hsamee}$. Symmetry of hsame turns the second equality into $\mathsf{ehsame}v$, and transitivity gives $u\mathsf{hsame}v$.

In the left/left case, apply the left-unit clause of Theorem 45.125 to the two endpoint laws. This gives $u\mathsf{hsamee}$ and $v\mathsf{hsamee}$, and the same symmetry-transitivity composition yields $u\mathsf{hsame}v$.

In the mixed case, use the right-unit clause for u and the left-unit clause for v in Theorem 45.125. The resulting comparisons $u\mathsf{hsamee}$ and $v\mathsf{hsamee}$ compose, after symmetry of the second comparison, to $u\mathsf{hsame}v$. □

Theorem 45.130 (Rat denominator continuation classifier associativity congruence)**.** *For all histories* $d, d', e, e', f, f', de, de', ef, ef', left, left', right, right' : \mathsf{BHist}$*, if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryClassifier}(e, e')$*,* $\mathsf{RatHistoryClassifier}(f, f')$*,* $\mathsf{Cont}(d, e, de)$*,* $\mathsf{Cont}(d', e', de')$*,* $\mathsf{Cont}(e, f, ef)$*,* $\mathsf{Cont}(e', f', ef')$*,* $\mathsf{Cont}(de, f, left)$*,* $\mathsf{Cont}(de', f', left')$*,* $\mathsf{Cont}(d, ef, right)$*, and* $\mathsf{Cont}(d', ef', right')$*, then*

$$\mathsf{RatHistoryClassifier}(left, left') \wedge \mathsf{RatHistoryClassifier}(right, right') \wedge left\,\mathsf{hsame}\,right \wedge left'\,\mathsf{hsame}\,right'.$$

Proof. Unpack the three displayed Rat history classifiers. Their carrier components, together with Definition 37.19 and Definition 37.12, give unary denominator witnesses for e, e', f, and f'. Their comparison components give $e\mathsf{hsame}e'$ and $f\mathsf{hsame}f'$.

Apply Definition 45.119 to $\mathsf{Cont}(d, e, de)$ and $\mathsf{Cont}(d', e', de')$. This yields $\mathsf{RatHistoryClassifier}(de, de')$. Apply the same closure theorem to $\mathsf{Cont}(e, f, ef)$ and $\mathsf{Cont}(e', f', ef')$ to obtain $\mathsf{RatHistoryClassifier}(ef, ef')$.

The outer left bracketing now uses the classifier for de, de' and the unary witnesses for f, f'. Another application of Definition 45.119 gives $\mathsf{RatHistoryClassifier}(left, left')$.

For the outer right bracketing, first read the carrier components of $\mathsf{RatHistoryClassifier}(ef, ef')$ through the same positive-denominator and unary/nonempty theorems, producing unary witnesses for ef and ef'. The comparison component of the classifier supplies $ef\mathsf{hsame}ef'$, so Definition 45.119 applied to $\mathsf{Cont}(d, ef, right)$ and $\mathsf{Cont}(d', ef', right')$ gives $\mathsf{RatHistoryClassifier}(right, right')$.

Finally apply Definition 45.117 to the unprimed continuation family $d, e, f, de, ef, left, right$. Apply it again to the primed family $d', e', f', de', ef', left', right'$. The two semigroup outputs give $left\mathsf{hsame}right$ and $left'\mathsf{hsame}right'$, completing the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation classifier assoc congruence`

Traditional target: denominator-side multiplication closure for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: Rat history classifier; **stability obligation**: continuation-result carrier, classifier, and ledger transport, external empty-history boundary laws separated from carrier inhabitance, associativity coherence, commutativity coherence, and classifier-compatible associative continuation-result coherence; **ledger**: the right denominator's positive-unary witness and the matching continuation witnesses; **core theorems**: Definition 45.113, Definition 45.117, Definition 45.118, Definition 45.119, Definition 45.122, Definition 45.124, Definition 45.125, Definition 45.130, and Definition 45.142; **exactness predicate**: carrier closure for the concrete denominator operation plus the explicit no-internal-empty-unit boundary; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.131 (Rat denominator continuation binary classifier congruence)**.** *For all histories* $d, d', e, e', r, r' : \mathsf{BHist}$*, if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryClassifier}(e, e')$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(d', e', r')$*, then* $\mathsf{RatHistoryClassifier}(r, r')$*.*

Proof. Unpack the second Rat history classifier. It supplies carrier witnesses for e and e', together with the endpoint comparison $e\mathsf{hsame}e'$.

Read the two carrier witnesses through Definition 37.19 and then Definition 37.12. This yields unary-history witnesses for both right denominators while preserving the comparison $e\mathsf{hsame}e'$ from the classifier.

Apply Definition 45.119 to the first classifier $\mathsf{RatHistoryClassifier}(d, d')$, the two unary right-denominator witnesses, the comparison $e\mathsf{hsame}e'$, and the displayed continuation witnesses $\mathsf{Cont}(d, e, r)$ and $\mathsf{Cont}(d', e', r')$. Its conclusion is exactly $\mathsf{RatHistoryClassifier}(r, r')$. □

Traditional target: denominator-side multiplication congruence for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, and Cont; **carrier choice**: Rat denominators; **classifier choice**: Rat history classifier; **stability obligation**: binary operation congruence; **ledger**: the two continuation witnesses and the right-denominator unary evidence; **core theorem**: Definition 45.131; **exactness predicate**: classifier preservation under displayed continuation; **standard bridge**: deferred; **status**: ProofTarget.

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation binary classifier congruence`

Theorem 45.132 (Rat history classifier continuation results are nonempty)**.** *For all histories* $d, d', e, e', r, r' : \mathsf{BHist}$*, if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryClassifier}(e, e')$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(d', e', r')$*, then neither continuation result is internally empty:*

$$(r\mathsf{hsame}e \to \mathsf{False}) \wedge (r'\mathsf{hsame}e \to \mathsf{False}).$$

Proof. Apply Definition 45.131 to the two input classifiers and the two displayed continuation witnesses. This gives $\mathsf{RatHistoryClassifier}(r, r')$. Its two carrier projections are Rat carrier witnesses for r and r', and the Rat carrier nonempty theorem rules out both empty-history comparisons. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier continuation result not empty`

Theorem 45.133 (Rat denominator continuation common-context cancellation)**.** *For all histories* $p, q, d, e, pd, pe, left, right$: BHist*, if* UnaryHistory(p)*,* UnaryHistory(q)*,* RatHistoryCarrier(d)*,* RatHistoryCarrier(e)*,* Cont(p, d, pd)*,* Cont(p, e, pe)*,* Cont$(pd, q, left)$*,* Cont$(pe, q, right)$*, and* RatHistoryClassifier$(left,$ *then* RatHistoryClassifier(d, e)*.*

Proof. Unpack the final Rat classifier. Its comparison component gives $left$hsame$right$.

Apply the continuation common-context cancellation theorem `BEDC.FKernel.Cont.cont_cancel_common_con` to the two matched prefix witnesses, the two matched suffix witnesses, and the comparison $left$hsame$right$. The shared prefix p and shared suffix q cancel, leaving the endpoint comparison $d$hsame$e$.

Combine this endpoint comparison with the original denominator carriers RatHistoryCarrier(d) and RatHistoryCarrier(e). These three pieces are exactly the fields of RatHistoryClassifier(d, e). □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation common context cancel`

Theorem 45.134 (Rat denominator matched-context classifier exactness)**.** *For all histories* $p, q, d, e, pd, pe, left, ri$ BHist*, if* UnaryHistory(p)*,* UnaryHistory(q)*,* RatHistoryCarrier(d)*,* RatHistoryCarrier(e)*,* Cont(p, d, pd)*,* Cont(p, e, pe)*,* Cont$(pd, q, left)$*, and* Cont$(pe, q, right)$*, then*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatHistoryClassifier}(left, right).$$

Proof. For the forward direction, apply Definition 37.58 to RatHistoryClassifier(d, e) with the common unary prefix p and common unary tail q. This gives the classifier for the append representatives obtained by placing the same matched context around d and e.

Continuation determinacy and associativity compare those representatives with the displayed endpoints $left$ and $right$, using the premises Cont(p, d, pd), Cont(p, e, pe), Cont$(pd, q, left)$, and Cont$(pe, q, right)$. Transporting the classifier across these endpoint comparisons gives RatHistoryClassifier$(left, righ$

For the reverse direction, apply Definition 45.133 to the same unary-context and continuation premises. It cancels the common prefix and suffix context from RatHistoryClassifier$(left, right)$ and returns RatHistoryClassifier(d, e). □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation common context classifier exactness`

↪ variant `BEDC.Derived.FieldUp.field rat denominator matched context classifier iff`

Theorem 45.135 (Rat denominator continuation common-context ledger closure)**.** *For all histories* $p, p', q, q', \rho, v, pr, pv, left, right$: BHist*, if* RatHistoryLedgerPolicy(ρ, v)*,* UnaryHistory(p)*,* UnaryHistory(p')*,* UnaryHistory(q)*,* UnaryHistory(q')*,* $p$hsamep'*,* $q$hsameq'*,* Cont(p, ρ, pr)*,* Cont(p', v, pv)*,* Cont$(pr, q, left)$*, and* Cont$(pv, q', right)$*, then* RatHistoryLedgerPolicy$(left, right)$*.*

Proof. First apply Definition 37.69 to the ledger RatHistoryLedgerPolicy(ρ, v), the matching unary prefixes p, p', and the matching unary suffixes q, q'. This gives the canonical ledger between append$(p,$ append$(\rho, q))$ and append$(p',$ append$(v, q'))$.

Next compare the canonical endpoints with the displayed two-step continuation endpoints. Continuation determinacy identifies the intermediate histories pr and pv with the corresponding prefix appends, and `BEDC.FKernel.Cont.cont_assoc_hsame` identifies the two displayed suffix continuations with the canonical common-context bracketings.

Finally transport the canonical ledger across those endpoint hsame links by Definition 37.71. The resulting ledger is exactly RatHistoryLedgerPolicy$(left, right)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation common context ledger closure`

Theorem 45.136 (Rat history classifier continuation unary context positive denominators)**.** *For histories* $d, e, p_d, p_e, t_d, t_e, md, me, out_d, out_e$: BHist*, if* RatHistoryClassifier(d, e) *holds,* p_d *and* t_d *are unary,* $p_d$hsamep_e *and* $t_d$hsamet_e*, and the two-step continuation displays place those contexts around* d *and* e*, then both displayed outputs are positive unary denominators:*

$$\mathsf{PositiveUnaryDenominator}(out_d) \wedge \mathsf{PositiveUnaryDenominator}(out_e).$$

Proof. Apply rational-history classifier unary-denominator context closure to obtain the classifier on the canonical associated endpoints $\mathsf{append}(p_d, \mathsf{append}(d, t_d))$ and $\mathsf{append}(p_e, \mathsf{append}(e, t_e))$. The continuation witnesses and associativity of append identify these canonical endpoints with out_d and out_e. Transport the classifier to the displayed outputs and read the two positive-denominator witnesses from the transported classifier. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryClassifier cont unary context positive denominators`

Theorem 45.137 (Rat history ledger continuation unary context positive denominators)**.** *For histories* $\rho, v, p_\rho, p_v, t_\rho, t_v, m_\rho, m_v, out_\rho, out_v$: BHist*, if* RatHistoryLedgerPolicy(ρ, v) *holds,* p_ρ *and* t_ρ *are unary,* $p_\rho$hsamep_v *and* $t_\rho$hsamet_v*, and the two-step continuation displays place those contexts around the raw and visible histories, then both displayed outputs are positive unary denominators:*

$$\mathsf{PositiveUnaryDenominator}(out_\rho) \wedge \mathsf{PositiveUnaryDenominator}(out_v).$$

Proof. First close the rational ledger under the matching unary prefix and tail contexts. The two continuation chains identify the canonical associated endpoints with the displayed outputs by associativity of append. Transport the closed ledger to those displayed outputs. Its raw carrier gives the first positive-denominator witness, and its transported visible carrier gives the second. □

Lean4 ✓ `BEDC.Derived.RatUp.RatHistoryLedgerPolicy cont unary context positive denominators`

Theorem 45.138 (Rat denominator continuation common-context exactness package)**.** *For all histories* $p, q, d, e, pd, pe, left, right$: BHist*, if* UnaryHistory(p)*,* UnaryHistory(q)*,* RatHistoryCarrier(d)*,* RatHistoryCarrier(e)*,* Cont(p, d, pd)*,* Cont(p, e, pe)*,* Cont$(pd, q, left)$*, and* Cont$(pe, q, right)$*, then both equivalences hold:*

$$\mathsf{RatHistoryClassifier}(left, right) \iff \mathsf{RatHistoryClassifier}(d, e).$$

$$\mathsf{RatHistoryLedgerPolicy}(left, right) \iff \mathsf{RatHistoryLedgerPolicy}(d, e).$$

Proof. For the classifier implication from $left, right$ back to d, e, apply Definition 45.133 to the same unary-context and continuation premises. It cancels the common prefix and suffix context from RatHistoryClassifier$(left, right)$ and returns RatHistoryClassifier(d, e).

For the classifier implication from d, e to $left, right$, apply Definition 37.58 to RatHistoryClassifier(d, e) with the common unary prefix p and common unary suffix q. This gives the classifier for the canonical endpoints obtained by placing p and q around d and e.

Continuation determinacy compares those canonical endpoints with the displayed endpoints $left$ and $right$, using the premises Cont(p, d, pd), Cont(p, e, pe), Cont$(pd, q, left)$, and Cont$(pe, q, right)$. Transporting the classifier across these endpoint comparisons by Definition 37.54 gives RatHistoryClassifier$(left, righ$

For the ledger implication from $left, right$ back to d, e, first convert RatHistoryLedgerPolicy$(left, right)$ to the raw-visible classifier by Definition 37.72. The cancellation theorem then returns RatHistoryClassifier(d, e),

whose comparison component is $d\mathsf{hsame}e$. Together with the input carrier $\mathsf{RatHistoryCarrier}(d)$, this is precisely $\mathsf{RatHistoryLedgerPolicy}(d, e)$.

For the ledger implication from d, e to $left, right$, apply Definition 37.69 to $\mathsf{RatHistoryLedgerPolicy}(d, e)$ with the common unary prefix p and common unary suffix q. Continuation determinacy compares the canonical ledger endpoints with $left$ and $right$, and Definition 37.71 transports the closed ledger across those endpoint comparisons. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation common context exactness package`

Theorem 45.139 (Rat denominator hsame-matched context classifier exactness)**.** *For all histories* $p, p', q, q', d, e, pd, pe, left, right : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{UnaryHistory}(p')$*,* $\mathsf{UnaryHistory}(q)$*,* $\mathsf{UnaryHistory}(q')$*,* $p\mathsf{hsame}p'$*,* $q\mathsf{hsame}q'$*,* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(p, d, pd)$*,* $\mathsf{Cont}(p', e, pe)$*,* $\mathsf{Cont}(pd, q, left)$*, and* $\mathsf{Cont}(pe, q', right)$*, then*

$$\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatHistoryClassifier}(left, right).$$

Proof. For the forward direction, assume $\mathsf{RatHistoryClassifier}(d, e)$. Apply Definition 37.58 to the unary prefixes p, p', the unary suffixes q, q', and the witnesses $p\mathsf{hsame}p'$ and $q\mathsf{hsame}q'$. This yields the classifier for the append histories obtained by placing the two history-same contexts around d and e.

The displayed endpoints are obtained from those append histories by the two-step continuation presentations. Continuation determinacy identifies pd and pe with the corresponding prefix appends, while continuation associativity identifies the final suffix continuations with the append bracketings. Transport the classifier across these endpoint hsame links by Definition 37.54 to obtain $\mathsf{RatHistoryClassifier}(left, right)$.

For the reverse direction, assume $\mathsf{RatHistoryClassifier}(left, right)$. Transport the continuation presentation along $p\mathsf{hsame}p'$ and $q\mathsf{hsame}q'$ using `BEDC.FKernel.Cont.cont_hsame_transport`; this gives a common-context presentation with the same displayed endpoints up to hsame.

Transport the endpoint classifier across those endpoint comparisons by Definition 37.54. Then apply Definition 45.133 to cancel the common unary prefix and suffix. Repacking the two Rat carriers with the recovered endpoint comparison gives $\mathsf{RatHistoryClassifier}(d, e)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator hsame matched context classifier exactness`

Theorem 45.140 (Rat denominator hsame-matched context exactness package)**.** *For all histories* $p, p', q, q', d, e, pd, pe, left, right : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{UnaryHistory}(p')$*,* $\mathsf{UnaryHistory}(q)$*,* $\mathsf{UnaryHistory}(q')$*,* $p\mathsf{hsame}p'$*,* $q\mathsf{hsame}q'$*,* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(p, d, pd)$*,* $\mathsf{Cont}(p', e, pe)$*,* $\mathsf{Cont}(pd, q, left)$*, and* $\mathsf{Cont}(pe, q', right)$*, then both equivalences hold:*

$$\mathsf{RatHistoryClassifier}(left, right) \iff \mathsf{RatHistoryClassifier}(d, e).$$

$$\mathsf{RatHistoryLedgerPolicy}(left, right) \iff \mathsf{RatHistoryLedgerPolicy}(d, e).$$

Proof. For the classifier equivalence, use Definition 45.139 with the displayed unary histories, the prefix and suffix comparisons $p\mathsf{hsame}p'$ and $q\mathsf{hsame}q'$, the two Rat carriers, and the four continuation premises. Its two implications give the stated equivalence after exchanging the displayed sides.

For the ledger implication from $left, right$ back to d, e, first convert $\mathsf{RatHistoryLedgerPolicy}(left, right)$ to the raw-visible classifier by Definition 37.72. The classifier equivalence just obtained reflects this to $\mathsf{RatHistoryClassifier}(d, e)$. Its comparison component is $d\mathsf{hsame}e$, and together with the displayed carrier $\mathsf{RatHistoryCarrier}(d)$ this is precisely $\mathsf{RatHistoryLedgerPolicy}(d, e)$.

For the ledger implication from d, e to $left, right$, apply Definition 45.135 to $\mathsf{RatHistoryLedgerPolicy}(d, e)$, the hsame-matched unary prefixes p, p', the hsame-matched unary suffixes q, q', and the four continuation premises. The conclusion is exactly $\mathsf{RatHistoryLedgerPolicy}(left, right)$. □

Theorem 45.141 (Rat denominator hsame-matched context ledger exactness)**.** *For all histories* $p, p', q, q', d, e, pd, pe, left, right : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{UnaryHistory}(p')$*,* $\mathsf{UnaryHistory}(q)$*,* $\mathsf{UnaryHistory}(q')$*,* $p\mathsf{hsame}p'$*,* $q\mathsf{hsame}q'$*,* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(p, d, pd)$*,* $\mathsf{Cont}(p', e, pe)$*,* $\mathsf{Cont}(pd, q, left)$*, and* $\mathsf{Cont}(pe, q', right)$*, then*

$$\mathsf{RatHistoryLedgerPolicy}(d, e) \iff \mathsf{RatHistoryLedgerPolicy}(left, right).$$

Proof. For the forward direction, assume $\mathsf{RatHistoryLedgerPolicy}(d, e)$. Apply Definition 45.135 to this ledger, the two unary prefixes, the two unary suffixes, the two hsame context comparisons, and the four displayed continuation witnesses. The result is exactly $\mathsf{RatHistoryLedgerPolicy}(left, right)$.

For the reverse direction, assume $\mathsf{RatHistoryLedgerPolicy}(left, right)$. Convert this endpoint ledger to the raw-visible classifier by Definition 37.72.

Apply Definition 45.139 in the reverse direction with the same unary histories, context comparisons, carrier witnesses, and continuation witnesses. This recovers $\mathsf{RatHistoryClassifier}(d, e)$, hence the endpoint comparison $d\mathsf{hsame}e$.

Pair the recovered comparison with the given carrier $\mathsf{RatHistoryCarrier}(d)$. These are precisely the data of $\mathsf{RatHistoryLedgerPolicy}(d, e)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator hsame matched context ledger exactness`

Traditional target: denominator exactness for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: Rat history classifier; **stability obligation**: hsame-matched classifier and ledger exactness; **ledger**: matched unary prefix and suffix contexts; **core theorem**: Definition 45.139, Definition 45.135, and Definition 45.140; **exactness predicate**: hsame-matched unary contexts preserve and reflect both the Rat classifier and the Rat ledger; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.142 (Rat denominator continuation is semigroup-only on the positive carrier)**.** *For the carrier* RatHistoryCarrier *and continuation relation* Cont*, Theorems 45.113 and 45.117 provide carrier closure and classifier associativity on positive Rat denominators. The empty history is not an internal carrier endpoint:* $\mathsf{RatHistoryCarrier}(\mathsf{e})$ *implies* False*. Consequently the endpoint laws in Theorem 45.124 are external empty-history endpoint laws. They do not provide an internal* $\mathsf{Monoid}^{\uparrow}$ *unit or an internal* $\mathsf{Field}^{\uparrow}$ *unit on the Rat denominator carrier.*

Proof. The semigroup clauses are exactly the content of the displayed closure and semigroup-law theorems. Carrier closure keeps every realized continuation result inside RatHistoryCarrier when both inputs are positive Rat denominators, and the semigroup law identifies the two continuation bracketings by the Rat history classifier.

For the unit boundary, suppose $\mathsf{RatHistoryCarrier}(\mathsf{e})$. Reflexivity of hsame at e gives ehsamee. Applying Theorem 37.18 to the assumed carrier witness and this reflexive endpoint equality yields False. Thus the empty-history clauses in Theorem 45.124 can be used only as external endpoint laws, not as an internal carrier unit. □

Theorem 45.143 (Rat denominator continuation has no internal carrier unit)**.** *For every history* $u : \mathsf{BHist}$*, the conjunction of* $\mathsf{RatHistoryCarrier}(u)$ *with the internal right-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

implies False. *Consequently no carried* u *can be a two-sided internal continuation unit on* RatHistoryCarrier *under* RatHistoryClassifier.

Proof. Assume a carried candidate u and the displayed right-unit classifier law. Let $d_1 := \mathsf{E1}(\mathsf{e})$. The positive-denominator construction gives $\mathsf{RatHistoryCarrier}(d_1)$.

Use the canonical continuation witness $\mathsf{Cont}(d_1, u, \mathsf{append}(d_1, u))$. Applying the assumed right-unit law at d_1 and $r := \mathsf{append}(d_1, u)$ gives $\mathsf{RatHistoryClassifier}(\mathsf{append}(d_1, u), d_1)$, whose comparison component is $\mathsf{append}(d_1, u)\mathsf{hsame}d_1$.

Transport the continuation result along this comparison to obtain $\mathsf{Cont}(d_1, u, d_1)$. The kernel theorem `BEDC.FKernel.Cont.cont_right_unit_unique` then forces $u\mathsf{hsame}\mathsf{e}$.

Finally apply Theorem 37.18 to the assumed carrier witness $\mathsf{RatHistoryCarrier}(u)$ and the equality $u\mathsf{hsame}\mathsf{e}$. This contradiction proves that no carried endpoint can realize the required right-unit law, and a two-sided internal unit would contain that right-unit law as one of its clauses. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation no internal unit`

Theorem 45.144 (Rat denominator continuation has no one-sided internal carrier unit)**.** *For every history* $u : \mathsf{BHist}$*, suppose* $\mathsf{RatHistoryCarrier}(u)$*. Then each of the following two laws separately implies* False*:*

1. *the internal right-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

2. *the internal left-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d).$$

Thus no carried u *can realize either one-sided internal continuation unit law on* RatHistoryCarrier *under* RatHistoryClassifier.

Proof. For the right-unit law, apply Theorem 45.143 to the same carried candidate u. The assumed carrier witness and the displayed right-unit law are exactly the hypotheses of that theorem, so the conclusion is False.

For the left-unit law, let $d_1 := \mathsf{E1}(\mathsf{e})$. The positive-denominator construction gives $\mathsf{RatHistoryCarrier}(d_1)$.

Use the canonical continuation witness $\mathsf{Cont}(u, d_1, \mathsf{append}(u, d_1))$. Applying the assumed left-unit law at d_1 and $r := \mathsf{append}(u, d_1)$ gives $\mathsf{RatHistoryClassifier}(\mathsf{append}(u, d_1), d_1)$, whose comparison component is $\mathsf{append}(u, d_1)\mathsf{hsame}d_1$.

Transport the continuation result along this comparison to obtain $\mathsf{Cont}(u, d_1, d_1)$. The kernel theorem `BEDC.FKernel.Cont.cont_left_unit_unique` then forces $u\mathsf{hsame}\mathsf{e}$.

Finally apply Theorem 37.18 to the assumed carrier witness $\mathsf{RatHistoryCarrier}(u)$ and the equality $u\mathsf{hsame}\mathsf{e}$. This contradiction proves that no carried endpoint can realize the required left-unit law. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation no one sided internal unit`

Definition 45.145 (Concrete Rat denominator continuation semigroup instance)**.** *The concrete denominator-side semigroup instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *has carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, operation represented by the concrete continuation relation* $\mathsf{Cont}(d, e, r)$*, and ledger policy* RatHistoryLedgerPolicy*. Its data intentionally contains no identity field: the operation is packaged only as the carried continuation fragment whose results are compared by* RatHistoryClassifier.

Traditional target: denominator-side multiplicative semigroup fragment for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: RatHistoryCarrier; **classifier choice**: RatHistoryClassifier; **stability obligations**: closure, associativity, commutativity, classifier congruence, ledger closure, and no internal carrier unit; **ledger**: matching continuation witnesses; **core theorems**: Definition 45.113, Definition 45.117, Definition 45.118, Definition 45.119, Definition 45.120, Definition 45.122, Definition 45.130, and Definition 45.143; **exactness predicate**: carrier, classifier, and ledger preservation under continuation; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.146 (Rat denominator continuation semigroup package)**.** *The concrete instance of Definition 45.145 satisfies the following package clauses:*

1. *for all histories* $d, e, f, de, ef, left, right$: BHist*, if* RatHistoryCarrier(d)*,* RatHistoryCarrier(e)*,* RatHistoryCarrier(f)*,* Cont(d, e, de)*,* Cont(e, f, ef)*,* Cont$(de, f, left)$*, and* Cont$(d, ef, right)$*, then* RatHistoryCarrier(de)*,* RatHistoryCarrier(ef)*,* RatHistoryCarrier$(left)$*,* RatHistoryCarrier$(right)$*, and* RatHistoryClassifier$(left, right)$*;*
2. *for all histories* d, d', u, u', r, s : BHist*, if* RatHistoryClassifier(d, d')*,* UnaryHistory(u)*,* UnaryHistory(u')*,* $u$hsameu'*,* Cont(d, u, r)*, and* Cont(d', u', s)*, then* RatHistoryClassifier(r, s)*;*
3. *for all histories* d, d', e, r, r' : BHist*, if* RatHistoryClassifier(d, d')*,* RatHistoryCarrier(e)*,* Cont(d, e, r)*, and* Cont(d', e, r')*, then* RatHistoryClassifier(r, r')*;*
4. *for all histories* ρ, v, u, u', r, s : BHist*, if* RatHistoryLedgerPolicy(ρ, v)*,* UnaryHistory(u)*,* UnaryHistory(u')*,* $u$hsameu'*,* Cont(ρ, u, r)*, and* Cont(v, u', s)*, then* RatHistoryLedgerPolicy(r, s)*;*
5. *for every history* u : BHist*,* RatHistoryCarrier(u) *together with the right-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

implies False*. Hence no carried endpoint can serve as a right or two-sided internal unit for this continuation instance.*

Proof. For the first clause, apply Definition 45.117. It gives the two intermediate carriers, the two outer carriers, and $left$hsame$right$. Combining the two outer carriers with this comparison yields RatHistoryClassifier$(left, right)$.

The same clause records the carrier closure supplied by Definition 45.113. The classifier statement for the two bracketings is the semigroup associativity component, while Definition 45.118 records the corresponding commutative classifier coherence for swapped carried continuations.

For classifier preservation under matching continuation tails, use Definition 45.119. For the shared carried right endpoint form, use Definition 45.120. The compatible associative version with classified triples is exactly Definition 45.130.

For the ledger clause, apply Definition 45.122 to the displayed ledger-related endpoints, matching unary tails, matching tail comparison, and continuation witnesses. The conclusion is the stated preservation of RatHistoryLedgerPolicy.

For the final obstruction clause, apply Definition 45.143. Its statement is precisely the contradiction obtained from a carried endpoint satisfying the right-unit classifier law, and any two-sided internal unit contains that right-unit clause. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation semigroup package`

Theorem 45.147 (Rat denominator continuation is not a FieldUp multiplication)**.** *For every proposed carried unit* $u : \mathsf{BHist}$*, suppose* $\mathsf{RatHistoryCarrier}(u)$ *and suppose the proposed carried multiplication on the Rat denominator carrier is the continuation relation* Cont*. If the corresponding* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *right-unit law under* $\mathsf{RatHistoryClassifier}$ *is*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

then the assumptions imply False*. Consequently a* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate cannot take Rat-denominator continuation itself as its carried multiplication together with a carried unit.*

Proof. Unpack the attempted $\mathsf{Field}^{\uparrow}$ multiplication data for the carried operation Cont. The right-unit field supplies exactly the displayed classifier law: for every Rat denominator d and every continuation result r from $\mathsf{Cont}(d, u, r)$, the result must classify as the original denominator d under $\mathsf{RatHistoryClassifier}$.

Apply Theorem 45.144 to the same history u, using the carried-unit witness $\mathsf{RatHistoryCarrier}(u)$ and the right-unit clause with this unpacked field. Its conclusion is False.

Theorem 45.142 records the positive part left intact by this contradiction: continuation on Rat denominators still has carrier closure and semigroup-level associativity, but those clauses do not supply an internal $\mathsf{Monoid}^{\uparrow}$ unit or the unit required by $\mathsf{Field}^{\uparrow}$ multiplication. □

Theorem 45.148 (Rat denominator continuation has no internal left carrier unit)**.** *For every history* $u : \mathsf{BHist}$*, the conjunction of* $\mathsf{RatHistoryCarrier}(u)$ *with the internal left-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

implies False*. Thus no carried* u *can realize the left endpoint half of an internal continuation unit on* $\mathsf{RatHistoryCarrier}$ *under* $\mathsf{RatHistoryClassifier}$*.*

Proof. Assume a carried candidate u and the displayed left-unit classifier law. Let $d_1 := \mathsf{E1}(\mathsf{e})$. The positive-denominator construction gives $\mathsf{RatHistoryCarrier}(d_1)$.

Use the canonical continuation witness $\mathsf{Cont}(u, d_1, \mathsf{append}(u, d_1))$. Applying the assumed left-unit law at d_1 and $r := \mathsf{append}(u, d_1)$ gives $\mathsf{RatHistoryClassifier}(\mathsf{append}(u, d_1), d_1)$, whose comparison component is $\mathsf{append}(u, d_1)\mathsf{hsame}d_1$.

Transport the continuation result along this comparison to obtain $\mathsf{Cont}(u, d_1, d_1)$. The kernel theorem `BEDC.FKernel.Cont.cont_left_unit_unique` then forces $u\mathsf{hsamee}$.

Finally apply Theorem 37.18 to the assumed carrier witness $\mathsf{RatHistoryCarrier}(u)$ and the equality $u\mathsf{hsamee}$. This contradiction proves that no carried endpoint can realize the required left-unit law. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation no internal left unit`

Theorem 45.149 (Rat history carrier right-append fixed absurd)**.** *For histories* $h, g : \mathsf{BHist}$*,*

$$\mathsf{RatHistoryCarrier}(h) \to \mathsf{append}(h, g)\mathsf{hsame}g \to \mathsf{False}.$$

Proof. The fixed append equality is a continuation witness $\mathsf{Cont}(h, g, g)$ after symmetry. The kernel left-unit uniqueness theorem for continuation then forces $h\mathsf{hsamee}$. This contradicts the nonempty endpoint theorem for $\mathsf{RatHistoryCarrier}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryCarrier right append fixed absurd`

Theorem 45.150 (Rat denominator continuation commutative-semigroup obstruction package)**.** *For the concrete BEDC carrier* RatHistoryCarrier*, continuation relation* Cont*, and classifier* RatHistoryClassifier*, Rat denominator continuation satisfies the following five-field package.*

1. *Carrier closure: if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*, and* $\mathsf{Cont}(d, e, r)$*, then* $\mathsf{RatHistoryCarrier}(r)$*.*
2. *Classifier associativity: if* $d, e, f, de, ef, \ell, r : \mathsf{BHist}$ *are related by* $\mathsf{Cont}(d, e, de)$*,* $\mathsf{Cont}(e, f, ef)$*,* $\mathsf{Cont}(de, f, \ell)$*, and* $\mathsf{Cont}(d, ef, r)$*, with* d, e, f *carried by* RatHistoryCarrier*, then the two bracketed results are carried and satisfy* $\ell\mathsf{hsame}r$*.*
3. *Classifier commutativity: if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(e, d, s)$*, then* $\mathsf{RatHistoryClassifier}(r, s)$*.*
4. *Internal left-unit impossibility: for every* $u : \mathsf{BHist}$*, the conjunction of* $\mathsf{RatHistoryCarrier}(u)$ *with*
$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d)$$
implies False*.*
5. *Internal right-unit impossibility: for every* $u : \mathsf{BHist}$*, the conjunction of* $\mathsf{RatHistoryCarrier}(u)$ *with*
$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$
implies False*.*

Consequently Rat-denominator continuation is a concrete commutative semigroup-level operation under RatHistoryClassifier *with no internal one-sided carried unit, and therefore cannot serve as the carried multiplication of* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*.*

Proof. The carrier-closure field is exactly Definition 45.113; it uses only the concrete carrier RatHistoryCarrier and the concrete continuation relation Cont.

The associativity field is Definition 45.117. It supplies the two intermediate carriers, the two outer result carriers, and the classifier comparison between the left- and right-bracketed continuations.

The commutativity field is Definition 45.118. It compares the two swapped denominator continuations by RatHistoryClassifier.

The internal left-unit obstruction is the left law of Definition 45.144. The internal right-unit obstruction is the right law of Definition 45.144.

Combining those two one-sided obstructions with the closure, associativity, and commutativity fields gives precisely a commutative semigroup-level continuation operation, not a carried field multiplication with an internal one-sided unit. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation commutative semigroup obstruction package`

Theorem 45.151 (Rat denominator continuation full obstruction package)**.** *For the concrete BEDC carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, operation relation* Cont*, and ledger* RatHistoryLedgerPolicy*, Rat denominator continuation satisfies the following six-clause certificate-shaped obstruction package.*

1. *Carrier closure: if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*, and* $\mathsf{Cont}(d, e, r)$*, then* $\mathsf{RatHistoryCarrier}(r)$*.*
2. *Classifier associativity: if* $d, e, f, de, ef, \ell, r : \mathsf{BHist}$ *are related by* $\mathsf{Cont}(d, e, de)$*,* $\mathsf{Cont}(e, f, ef)$*,* $\mathsf{Cont}(de, f, \ell)$*, and* $\mathsf{Cont}(d, ef, r)$*, with* d, e, f *carried by* RatHistoryCarrier*, then the two bracketed results are carried and satisfy* $\mathsf{RatHistoryClassifier}(\ell, r)$*.*

3. *Classifier commutativity: if* $\mathsf{RatHistoryCarrier}(d)$, $\mathsf{RatHistoryCarrier}(e)$, $\mathsf{Cont}(d, e, r)$, *and* $\mathsf{Cont}(e, d, s)$, *then* $\mathsf{RatHistoryClassifier}(r, s)$.
4. *Binary classifier congruence: if* $\mathsf{RatHistoryClassifier}(d, d')$, $\mathsf{RatHistoryClassifier}(e, e')$, $\mathsf{Cont}(d, e, r)$, *and* $\mathsf{Cont}(d', e', s)$, *then* $\mathsf{RatHistoryClassifier}(r, s)$.
5. *Ledger closure for classified unary continuation tails: if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$, $\mathsf{UnaryHistory}(u)$, $\mathsf{UnaryHistory}(u')$, $u\mathsf{hsame}u'$, $\mathsf{Cont}(\rho, u, r)$, *and* $\mathsf{Cont}(v, u', s)$, *then* $\mathsf{RatHistoryLedgerPolicy}(r, s)$.
6. *Internal carried-unit obstruction: for every history* $u : \mathsf{BHist}$, *if* $\mathsf{RatHistoryCarrier}(u)$, *then the left-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

and the right-unit classifier law

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

each imply False.

Consequently Rat-denominator continuation is a classifier-and-ledger commutative semigroup fragment, but it cannot be promoted to the multiplicative carrier of $\mathsf{Monoid}^{\uparrow}$ *or* $\mathsf{Field}^{\uparrow}$ *because no carried one-sided unit exists.*

Proof. The carrier-closure clause is exactly Definition 45.113. The associativity clause is Definition 45.117; its conclusion supplies the two intermediate carriers, the two outer result carriers, and $\ell\mathsf{hsame}r$, which repacks as $\mathsf{RatHistoryClassifier}(\ell, r)$.

The commutativity clause is Definition 45.118. For binary classifier congruence, unpack the classifier $\mathsf{RatHistoryClassifier}(e, e')$ to obtain unary denominator witnesses for the two right inputs and the comparison $e\mathsf{hsame}e'$, then apply Definition 45.131 to the classified left inputs and the two continuation witnesses.

The ledger clause is Definition 45.122. It applies to the displayed ledger endpoints, matched unary continuation tails, tail comparison, and continuation witnesses, yielding $\mathsf{RatHistoryLedgerPolicy}(r, s)$.

The carried-unit obstruction is Definition 45.144. Its two conclusions rule out the left-unit and right-unit classifier laws for any carried candidate u, completing the certificate-shaped package. □

Theorem 45.152 (Rat denominator continuation obstruction is context-invariant)**.** *For all histories* $p, p', q, q', d, e, pd, pe, left, right : \mathsf{BHist}$, *suppose* $\mathsf{UnaryHistory}(p)$, $\mathsf{UnaryHistory}(p')$, $\mathsf{UnaryHistory}(q)$, $\mathsf{UnaryHistory}(q')$, $p\mathsf{hsame}p'$, $q\mathsf{hsame}q'$, $\mathsf{RatHistoryCarrier}(d)$, $\mathsf{RatHistoryCarrier}(e)$, *and the four displayed continuation witnesses*

$$\mathsf{Cont}(p, d, pd), \quad \mathsf{Cont}(p', e, pe), \quad \mathsf{Cont}(pd, q, left), \quad \mathsf{Cont}(pe, q', right).$$

Then the matched unary context wrapping preserves the full obstruction data:

1. $\mathsf{RatHistoryClassifier}(d, e) \iff \mathsf{RatHistoryClassifier}(left, right)$.
2. $\mathsf{RatHistoryLedgerPolicy}(d, e) \to \mathsf{RatHistoryLedgerPolicy}(left, right)$.
3. *For every history* $u : \mathsf{BHist}$, *if* $\mathsf{RatHistoryCarrier}(u)$, *then the internal left-unit classifier law*

$$\forall x, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(x) \to \mathsf{Cont}(u, x, r) \to \mathsf{RatHistoryClassifier}(r, x)$$

and the internal right-unit classifier law

$$\forall x, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(x) \to \mathsf{Cont}(x, u, r) \to \mathsf{RatHistoryClassifier}(r, x)$$

each imply False.

Proof. The classifier claim is exactly Definition 45.139 with the displayed unary prefix contexts, unary suffix contexts, context comparisons, carrier witnesses, and continuation witnesses.

For the ledger claim, apply Definition 45.135 to the same matched unary context data and the displayed continuation witnesses. Its conclusion transports the Rat ledger policy from (d, e) to $(left, right)$.

It remains to record that the matched context does not create an internal carried unit. The left-unit obstruction is Definition 45.148, applied to the carried candidate u and the displayed left-unit classifier law. The right-unit obstruction is Definition 45.143, applied to the same carried candidate u and the displayed right-unit classifier law. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator context invariant obstruction package`

Theorem 45.153 (Rat denominator continuation associative obstruction package)**.** *For the concrete BEDC carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, operation relation* Cont*, and ledger* RatHistoryLedgerPolicy*, Rat denominator continuation satisfies the following certificate-shaped obstruction package.*

1. *Carrier closure: if* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*, and* $\mathsf{Cont}(d, e, r)$*, then* $\mathsf{RatHistoryCarrier}(r)$*.*
2. *Binary classifier congruence: if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryClassifier}(e, e')$*,* $\mathsf{Cont}(d, e, r)$*, and* $\mathsf{Cont}(d', e', r')$*, then* $\mathsf{RatHistoryClassifier}(r, r')$*.*
3. *Classified-triple associativity congruence: for all histories* $d, d', e, e', f, f', de, de', ef, ef', left, left', right, right'$: BHist*, if* $\mathsf{RatHistoryClassifier}(d, d')$*,* $\mathsf{RatHistoryClassifier}(e, e')$*,* $\mathsf{RatHistoryClassifier}(f, f')$*,* $\mathsf{Cont}(d, e, de)$*,* $\mathsf{Cont}(d', e', de')$*,* $\mathsf{Cont}(e, f, ef)$*,* $\mathsf{Cont}(e', f', ef')$*,* $\mathsf{Cont}(de, f, left)$*,* $\mathsf{Cont}(de', f', left')$*,* $\mathsf{Cont}(d, ef, right)$*, and* $\mathsf{Cont}(d', ef', right')$*, then*

$$\mathsf{RatHistoryClassifier}(left, left') \wedge \mathsf{RatHistoryClassifier}(right, right') \wedge left \mathsf{hsame} right \wedge left' \mathsf{hsame} right'.$$

4. *Hsame-matched context classifier-and-ledger exactness: for all histories* $p, p', q, q', d, e, pd, pe, left, right$: BHist*, if* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{UnaryHistory}(p')$*,* $\mathsf{UnaryHistory}(q)$*,* $\mathsf{UnaryHistory}(q')$*,* $p \mathsf{hsame} p'$*,* $q \mathsf{hsame} q'$*,* $\mathsf{RatHistoryCarrier}(d)$*,* $\mathsf{RatHistoryCarrier}(e)$*,* $\mathsf{Cont}(p, d, pd)$*,* $\mathsf{Cont}(p', e, pe)$*,* $\mathsf{Cont}(pd, q, left)$*, and* $\mathsf{Cont}(pe, q', right)$*, then both equivalences hold:*

$$\mathsf{RatHistoryClassifier}(left, right) \iff \mathsf{RatHistoryClassifier}(d, e).$$

$$\mathsf{RatHistoryLedgerPolicy}(left, right) \iff \mathsf{RatHistoryLedgerPolicy}(d, e).$$

5. *Ledger closure for classified unary continuation tails: if* $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$*,* $\mathsf{UnaryHistory}(u)$*,* $\mathsf{UnaryHistory}(u')$*,* $u \mathsf{hsame} u'$*,* $\mathsf{Cont}(\rho, u, r)$*, and* $\mathsf{Cont}(v, u', s)$*, then* $\mathsf{RatHistoryLedgerPolicy}(r, s)$*.*

6. *Endpoint-forced emptiness: if a history* $u : \mathsf{BHist}$ *satisfies either one-sided classifier unit law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

or

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

then $u\mathsf{hsamee}$.

7. *Endpoint uniqueness: for all histories* $u, v : \mathsf{BHist}$*, if both endpoints satisfy the right-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, w, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

for $w = u$ *and for* $w = v$*, then* $u\mathsf{hsame}v$*; if both endpoints satisfy the left-unit classifier law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(w, d, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

for $w = u$ *and for* $w = v$*, then* $u\mathsf{hsame}v$*; and if* u *satisfies the right-unit classifier law while* v *satisfies the left-unit classifier law, then* $u\mathsf{hsame}v$.

8. *Carried one-sided unit obstruction: for every history* $u : \mathsf{BHist}$*, if* $\mathsf{RatHistoryCarrier}(u)$*, then each one-sided classifier unit law for* u *implies* False.

Consequently the Rat-denominator continuation spine contains the closure, operation congruence, associative classified composition, context exactness, and ledger stability required for a certificate, while its unit endpoint is forced outside the carried denominator carrier.

Proof. The carrier-closure field is exactly Definition 45.113. The binary classifier congruence field is Definition 45.131; it compares two continuation outputs from two classified input pairs.

The classified-triple associativity congruence field is Definition 45.130. It handles two classified triples of continuation data at once, producing classifier comparisons for the corresponding left and right bracketings together with the two internal associativity comparisons.

For hsame-matched context exactness, apply Definition 45.140 to the displayed unary prefixes, unary suffixes, context comparisons, Rat carriers, and continuation witnesses. Its two equivalences are precisely the classifier and ledger exactness clauses in the package.

The ledger-closure field is Definition 45.122. It transports the Rat ledger along classified unary continuation tails and returns $\mathsf{RatHistoryLedgerPolicy}(r, s)$.

For endpoint-forced emptiness, use Definition 45.125. Its right and left clauses force any one-sided classifier unit endpoint to satisfy $u\mathsf{hsamee}$.

Endpoint uniqueness is Definition 45.129; its three clauses cover the displayed right/right, left/left, and mixed one-sided classifier-unit comparisons.

Finally, the carried one-sided unit obstruction is Definition 45.144, which turns either one-sided classifier unit law into False once the endpoint is carried by $\mathsf{RatHistoryCarrier}$. □

Traditional target: denominator-side continuation with empty endpoint unit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, and Cont; **carrier choice**: RatDenomUnitCarrier; **classifier choice**: RatDenomUnitClassifier; **stability obligations**: empty endpoint coverage, continuation closure, one-sided endpoint laws, and classified triple associativity; **ledger**: continuation

witnesses together with the resulting hsame comparisons; **core theorem**: Definition 45.177; **exactness predicate**: both endpoints lie in the empty-unit denominator carrier and are related by hsame; **standard bridge**: denominator continuation with the empty history as endpoint unit; **status**: ProofTarget.

Traditional target: internal unit obstruction for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: RatHistoryClassifier; **stability obstruction**: no carried one-sided unit and no continuation-as-field-multiplication certificate; **ledger**: the attempted unit endpoint; **core theorems**: Definition 45.143, Definition 45.144, Definition 45.147, Definition 45.150, Definition 45.152, and Definition 45.153; **exactness predicate**: failure of internal one-sided unit coverage; **standard bridge**: deferred; **status**: ProofTarget.

Traditional target: denominator-side continuation with empty endpoint unit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, and Cont; **carrier choice**: RatDenomUnitCarrier; **classifier choice**: RatDenomUnitClassifier; **stability obligations**: empty endpoint coverage, continuation closure, one-sided endpoint laws, and classified triple associativity; **ledger**: continuation witnesses together with the resulting hsame comparisons; **core theorem**: Definition 45.177; **exactness predicate**: both endpoints lie in the empty-unit denominator carrier and are related by hsame; **standard bridge**: denominator continuation with the empty history as endpoint unit; **status**: ProofTarget.

Traditional target: internal unit obstruction for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: RatHistoryClassifier; **stability obstruction**: no carried one-sided unit and no continuation-as-field-multiplication certificate; **ledger**: the attempted unit endpoint; **core theorems**: Definition 45.143, Definition 45.144, Definition 45.147, Definition 45.150, Definition 45.152, and Definition 45.153; **exactness predicate**: failure of internal one-sided unit coverage; **standard bridge**: deferred; **status**: ProofTarget.

Definition 45.154 (Rat denominator continuation with adjoined empty unit)**.** *For each history* $h : \mathsf{BHist}$*, define the empty-unit denominator carrier by*

$$\mathsf{RatDenomUnitCarrier}(h) \iff h\mathsf{hsamee} \vee \mathsf{RatHistoryCarrier}(h).$$

For histories $h, k : \mathsf{BHist}$*, define the corresponding endpoint classifier by*

$$\mathsf{RatDenomUnitClassifier}(h, k) \iff \mathsf{RatDenomUnitCarrier}(h) \wedge \mathsf{RatDenomUnitCarrier}(k) \wedge h\mathsf{hsame}k.$$

Lean4 (definition) `BEDC.Derived.FieldUp.RatDenomUnitCarrier`
Lean4 (definition) `BEDC.Derived.FieldUp.RatDenomUnitClassifier`

Theorem 45.155 (Rat denominator unit carrier history-sameness transport)**.** *For histories* $h, k :$ BHist*, if* $h\mathsf{hsame}k$ *and* $\mathsf{RatDenomUnitCarrier}(h)$*, then* $\mathsf{RatDenomUnitCarrier}(k)$*.*

Proof. Split the carrier disjunction for h. In the empty branch, compose $h\mathsf{hsame}k$ with the displayed comparison $h\mathsf{hsamee}$ and symmetry or transitivity of hsame to obtain $k\mathsf{hsamee}$. This selects the empty branch of $\mathsf{RatDenomUnitCarrier}(k)$.

In the Rat branch, apply Rat carrier transport along the hsame comparison from h to k. The resulting $\mathsf{RatHistoryCarrier}(k)$ witness selects the Rat branch of the denominator unit carrier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier hsame transport`

Theorem 45.156 (Rat denominator unit carrier continuation closure)**.** *For histories* $h, k, r :$ BHist*, if* $\mathsf{RatDenomUnitCarrier}(h)$*,* $\mathsf{RatDenomUnitCarrier}(k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{RatDenomUnitCarrier}(r)$*.*

Proof. Split the two input carriers. The empty/empty case uses the two continuation unit laws to identify the result with e, hence the result lies in the empty branch.

In the two mixed cases, the corresponding unit law identifies the continuation result with the nonempty input; use cont_respects_hsame and Rat carrier transport to move the Rat carrier witness to r.

In the Rat/Rat case, apply the Rat denominator continuation carrier-closure theorem directly. The resulting $\mathsf{RatHistoryCarrier}(r)$ witness selects the Rat branch of $\mathsf{RatDenomUnitCarrier}(r)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier continuation closed`

Theorem 45.157 (Rat denominator unit carrier append with Rat left factor)**.** *For histories h, k : BHist, if h is a Rat denominator endpoint and k is carried by the adjoined empty-unit denominator carrier, then the append product is again a Rat denominator endpoint:*

$$\mathsf{RatHistoryCarrier}(h) \to \mathsf{RatDenomUnitCarrier}(k) \to \mathsf{RatHistoryCarrier}(\mathsf{append}(h, k)).$$

Proof. Split the adjoined-unit carrier witness for k. If $k$hsamee, the append product is history-same to h, so the Rat carrier for h transports to the product. If k is a Rat denominator endpoint, apply Rat denominator continuation carrier closure to the canonical append continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append left rat closed`

Theorem 45.158 (Rat denominator unit carrier append with Rat right factor)**.** *For histories h, k : BHist, if h is carried by the adjoined empty-unit denominator carrier and k is a Rat denominator endpoint, then the append product is again a Rat denominator endpoint:*

$$\mathsf{RatDenomUnitCarrier}(h) \to \mathsf{RatHistoryCarrier}(k) \to \mathsf{RatHistoryCarrier}(\mathsf{append}(h, k)).$$

Proof. Split the adjoined-unit carrier witness for h. If $h$hsamee, the append product is history-same to k, so the Rat carrier for k transports to the product. If h is a Rat denominator endpoint, apply Rat denominator continuation carrier closure to the canonical append continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append right rat closed`

Theorem 45.159 (Rat history classifier append with right denominator unit)**.** *For histories h, h', k, k' : BHist, if h and h' are related by the Rat history classifier and k and k' are related by the adjoined denominator unit classifier, then appending the unit-classified right factor preserves the Rat history classifier:*

$$\mathsf{RatHistoryClassifier}(h, h') \to \mathsf{RatDenomUnitClassifier}(k, k') \to \mathsf{RatHistoryClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')).$$

Proof. The carrier fields are obtained from the Rat left-factor append closure for the adjoined denominator unit carrier. The endpoint comparison is the canonical append continuation comparison transported along the two classifier hsame witnesses. These data assemble the Rat history classifier for the two appended endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier append right denom unit closed`

Theorem 45.160 (Rat denominator unit append carrier branch cases)**.** *For histories h, k : BHist, if the append endpoint is carried by the adjoined empty-unit denominator carrier, then either both factors are empty endpoints or one factor is a genuine Rat denominator endpoint:*

$$\mathsf{RatDenomUnitCarrier}(\mathsf{append}(h, k)) \to (h\mathsf{hsamee} \wedge k\mathsf{hsamee}) \vee \mathsf{RatHistoryCarrier}(h) \vee \mathsf{RatHistoryCarrier}(k).$$

Proof. Split the concrete carrier disjunction for $\mathsf{RatDenomUnitCarrier}(\mathsf{append}(h, k))$. In the empty branch, the append-empty inversion theorem gives $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. In the Rat branch, apply Rat denominator append splitting to the carried append endpoint, obtaining a Rat carrier witness on the left or right factor. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append branch cases`

The per-object certificate has ten fields: traditional target carrier stability for the denominator unit fragment; BEDC source BHist, RatHistoryCarrier, and Cont; carrier RatDenomUnitCarrier; classifier RatDenomUnitClassifier; stability obligations transport and continuation closure; ledger continuation witnesses; core theorems 45.155 and 45.156; exactness predicate preservation of admitted endpoints; standard bridge deferred; status ProofTarget.

Theorem 45.161 (Rat denominator unit carrier nonempty endpoint exactness)**.** *For every history h, if* $\mathsf{RatDenomUnitCarrier}(h)$ *holds and* $h\mathsf{hsamee}$ *is impossible, then h is a genuine Rat denominator endpoint:*

$$\mathsf{RatHistoryCarrier}(h).$$

Proof. Unfold the carrier disjunction. The empty-unit branch is exactly $h\mathsf{hsamee}$, contradicting the supplied nonempty endpoint condition. The other branch is the required $\mathsf{RatHistoryCarrier}(h)$ witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier nonempty rat`

Theorem 45.162 (Rat denominator unit nonempty classifier exposes Rat classifier)**.** *For histories $h, k : \mathsf{BHist}$, if* $\mathsf{RatDenomUnitClassifier}(h, k)$ *holds and neither endpoint is history-same to* e*, then the same endpoint pair is classified by the underlying Rat denominator classifier:*

$$\mathsf{RatDenomUnitClassifier}(h, k) \Rightarrow (h\mathsf{hsamee} \Rightarrow \mathsf{False}) \Rightarrow (k\mathsf{hsamee} \Rightarrow \mathsf{False}) \Rightarrow \mathsf{RatHistoryClassifier}(h, k).$$

Proof. Unpack $\mathsf{RatDenomUnitClassifier}(h, k)$ into the two denominator-unit carrier witnesses and the endpoint comparison $h\mathsf{hsame}k$. Apply Definition 45.161 separately to the left and right carrier witnesses, using the two nonempty endpoint hypotheses. These two Rat carrier witnesses, together with the original endpoint comparison, repack as $\mathsf{RatHistoryClassifier}(h, k)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier nonempty rat classifier`

Theorem 45.163 (Rat denominator unit classifier nonempty endpoint iff)**.** *For histories $h, k :$* BHist*, a denominator-unit classifier witness transports the nonempty endpoint predicate in both directions:*

$$\mathsf{RatDenomUnitClassifier}(h, k) \Rightarrow ((h\mathsf{hsamee} \Rightarrow \mathsf{False}) \Longleftrightarrow (k\mathsf{hsamee} \Rightarrow \mathsf{False})).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier nonempty endpoint iff`

Proof. Unpack $\mathsf{RatDenomUnitClassifier}(h, k)$ and use its endpoint comparison $h\mathsf{hsame}k$. Composing this comparison with an empty-endpoint witness on one side produces an empty-endpoint witness on the other side, contradicting the assumed nonempty predicate. The reverse direction uses symmetry of the same hsame comparison. □

Theorem 45.164 (Rat denominator unit classifier left-nonempty Rat exactness)**.** *For histories $h, k :$* BHist*, a denominator-unit classifier witness identifies the native Rat denominator classifier with nonemptiness of the left endpoint:*

$$\mathsf{RatDenomUnitClassifier}(h, k) \Rightarrow (\mathsf{RatHistoryClassifier}(h, k) \Longleftrightarrow (h\mathsf{hsamee} \Rightarrow \mathsf{False})).$$

Proof. The forward implication uses the endpoint nonemptiness forced by RatHistoryClassifier. Conversely, denominator-unit classification transports left nonemptiness to the right endpoint, and the nonempty classifier theorem upgrades the two denominator-unit carriers to the native Rat classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier rat classifier left nonempty iff`

Theorem 45.165 (Rat denominator unit classifier either-nonempty Rat exactness)**.** *For histories* $h, k : \mathsf{BHist}$*, a denominator-unit classifier witness identifies the native Rat denominator classifier with nonemptiness of either endpoint:*

$$\mathsf{RatDenomUnitClassifier}(h, k) \Rightarrow (\mathsf{RatHistoryClassifier}(h, k) \iff ((h\mathsf{hsamee} \Rightarrow \mathsf{False}) \vee (k\mathsf{hsamee} \Rightarrow \mathsf{False}))) .$$

Proof. For the forward implication, apply Definition 45.164. The resulting left endpoint nonemptiness gives the displayed disjunction.

For the reverse implication, split the disjunction. In the left-endpoint case, apply Definition 45.164 directly.

In the right-endpoint case, first use Definition 45.163 to transport right nonemptiness to left nonemptiness. The left-nonempty Rat exactness theorem then supplies $\mathsf{RatHistoryClassifier}(h, k)$. □

Theorem 45.166 (Rat denominator unit classifier Rat-history exactness)**.** *For histories* $h, k : \mathsf{BHist}$*, the native Rat denominator classifier is exactly the denominator-unit classifier restricted to genuine nonempty Rat denominator endpoints:*

$$\mathsf{RatHistoryClassifier}(h, k) \iff (\mathsf{RatDenomUnitClassifier}(h, k) \wedge (h\mathsf{hsamee} \Rightarrow \mathsf{False}) \wedge (k\mathsf{hsamee} \Rightarrow \mathsf{False})) .$$

Proof. For the forward direction, unpack $\mathsf{RatHistoryClassifier}(h, k)$ into its two Rat carrier witnesses and the endpoint comparison. Repack the two carrier witnesses as RatDenomUnitCarrier witnesses and keep the same comparison, giving $\mathsf{RatDenomUnitClassifier}(h, k)$.

The two nonempty endpoint predicates are supplied by Definition 37.51, applied to the same native Rat classifier witness. This records that neither endpoint lies in the adjoined empty-unit branch.

For the reverse direction, start from the denominator-unit classifier witness and the two nonempty endpoint predicates. Apply Definition 45.162; its conclusion is precisely $\mathsf{RatHistoryClassifier}(h, k)$. □

The per-object certificate has ten fields: traditional target strict denominator-unit/native Rat classifier exactness; BEDC source BHist, RatHistoryClassifier, RatDenomUnitCarrier, and RatDenomUnitClassifier; carrier and classifier choices the displayed predicates; stability obligations nonempty endpoint transport plus native classifier exposure; ledger the adjoined empty-unit branch; core theorem 45.166; exactness predicate the displayed iff; standard bridge deferred; status ProofTarget.

Theorem 45.167 (Rat denominator unit carrier visible cases)**.** *For every history* $h : \mathsf{BHist}$*, an adjoined empty-unit denominator carrier witness exposes exactly the visible empty branch or a one-headed unary denominator tail:*

$$\mathsf{RatDenomUnitCarrier}(h) \to (h\mathsf{hsamee} \vee \exists t : \mathsf{BHist}.\ h = \mathsf{E1}(t) \wedge \mathsf{UnaryHistory}(t)) .$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier visible cases`

Proof. Split the concrete history h. The empty constructor gives the empty branch. The zero-headed constructor is incompatible with the denominator-unit carrier by the zero-endpoint exclusion theorem. The one-headed constructor reads the unary tail back through the carrier endpoint equivalence. □

Theorem 45.168 (Rat denominator unit classifier visible cases)**.** *For any classified pair* h, k : BHist *in the adjoined empty-unit denominator classifier, both endpoints are empty together or both are one-headed unary denominator tails with equal exposed tails:*

$$\mathsf{RatDenomUnitClassifier}(h,k) \to (h\mathsf{hsamee} \wedge k\mathsf{hsamee}) \vee \exists d, e : \mathsf{BHist}.\ h = \mathsf{E1}(d) \wedge k = \mathsf{E1}(e) \wedge \mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryH}$$

Proof. Apply the visible-case split for $\mathsf{RatDenomUnitCarrier}$ to both classifier endpoints. The two empty branches give the empty pair. The mixed empty and one-headed branches contradict the classifier's hsame field by history constructor separation. In the one-headed branch, the two carrier witnesses give unary tails and hsame constructor inversion gives the exposed tail equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier visible cases`

Theorem 45.169 (Rat denominator unit carrier visible cases iff)**.** *For every history* h : BHist*, the adjoined empty-unit denominator carrier is equivalent to the visible empty branch or a one-headed unary denominator tail:*

$$\mathsf{RatDenomUnitCarrier}(h) \iff (h\mathsf{hsamee} \vee \exists t : \mathsf{BHist}.\ h = \mathsf{E1}(t) \wedge \mathsf{UnaryHistory}(t))\,.$$

Proof. The forward direction is the visible carrier case split above. Conversely, the empty branch is the adjoined unit carrier branch, and the one-headed branch is rebuilt from the one-headed denominator carrier characterization. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier visible cases iff`

Theorem 45.170 (Rat denominator unit continuation exposes a nonempty Rat endpoint)**.** *For histories* h, k, r*, if* h *and* k *lie in* $\mathsf{RatDenomUnitCarrier}$ *and* $\mathsf{Cont}(h, k, r)$ *has a result not* hsame *to* e*, then at least one input endpoint is a genuine Rat denominator endpoint:*

$$\mathsf{RatHistoryCarrier}(h) \vee \mathsf{RatHistoryCarrier}(k).$$

Proof. Split both carrier witnesses into their empty-unit and Rat endpoint branches. If both inputs are empty-unit endpoints, continuation transport along those hsame witnesses identifies the result with e, contradicting the nonempty result condition. In either mixed branch, the Rat side is already the required disjunct. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier continuation nonempty rat endpoint`

Theorem 45.171 (Rat denominator unit continuation result empty iff)**.** *For histories* h, k, r*, if* h *and* k *lie in* $\mathsf{RatDenomUnitCarrier}$ *and* $\mathsf{Cont}(h, k, r)$*, then the continuation result is internally empty exactly when both carried endpoints are internally empty:*

$$r\mathsf{hsamee} \iff h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. Split the two carrier witnesses into their empty and genuine Rat-history branches. The empty–empty branch transports the continuation to the canonical empty-left unit result. In either mixed branch, continuation transport identifies the result with the genuine Rat endpoint, contradicting RatHistoryCarrier_not_empty. In the genuine–genuine branch, Rat continuation closure makes the result a Rat history, again excluding emptiness. Conversely, if both endpoints are internally empty, continuation transport from the canonical empty-empty continuation identifies r with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier continuation result empty iff`

Theorem 45.172 (Rat denominator unit classifier continuation result empty iff)**.** *For classified denominator-unit pairs (h, h') and (k, k'), if $\mathsf{Cont}(h, k, r)$ and $\mathsf{Cont}(h', k', r')$, then the two continuation results have the same empty endpoint status:*

$$r\mathsf{hsamee} \iff r'\mathsf{hsamee}.$$

Proof. Project the carrier fields from the two classifier witnesses. Carrier-level continuation empty exactness identifies $r\mathsf{hsamee}$ with simultaneous emptiness of h and k, and identifies $r'\mathsf{hsamee}$ with simultaneous emptiness of h' and k'. The classifier hsame fields transport endpoint emptiness between h and h', and between k and k'. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier continuation result empty iff`

Theorem 45.173 (Rat denominator unit-envelope classifier is append-closed)**.** *For histories $h, h', k, k' :$ BHist, suppose each displayed pair lies in the denominator unit envelope: either both endpoints are* RatHistoryCarrier *and related by* RatHistoryClassifier*, or both endpoints are* hsame *to* e*. Then* $\mathsf{append}(h, k)$ *and* $\mathsf{append}(h', k')$ *lie in the same envelope.*

Proof. Split the two envelope hypotheses. In the carried/carried branch, the right classifier exposes a unary denominator tail and a tail hsame comparison, so Rat denominator classifier append closure applies to the left classifier. In the mixed branches, the empty side reduces by the left or right unit law for append, and classifier transport carries the nonempty classifier across those hsame comparisons. The empty/empty branch follows by eliminating the four empty endpoint equalities and applying reflexivity at e. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope classifier append closed`

Theorem 45.174 (Rat denominator unit-envelope classifier empty-right exactness)**.** *For every history $h :$ BHist, the unit-envelope classifier with empty right endpoint is exactly empty endpoint sameness:*

$$\mathsf{FieldRatDenominatorUnitEnvelopeClassifier}(h, \mathsf{e}) \iff h\mathsf{hsamee}.$$

Proof. Split the concrete unit-envelope classifier. In the carried branch, the right endpoint would be both RatHistoryCarrier and e, which contradicts RatHistoryCarrier_not_empty. In the empty branch, the left endpoint comparison is already present. Conversely, $h\mathsf{hsamee}$ selects the empty-empty branch with reflexivity at e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty right iff`

Theorem 45.175 (Rat denominator unit-envelope classifier empty-left exactness)**.** *For histories $h, k :$ BHist, if the left endpoint is empty, then the unit-envelope classifier is exactly empty endpoint sameness on the right:*

$$h\mathsf{hsamee} \implies (\mathsf{FieldRatDenominatorUnitEnvelopeClassifier}(h, k) \iff k\mathsf{hsamee}).$$

Proof. Split the concrete unit-envelope classifier. In the carried branch, the left endpoint would be both RatHistoryCarrier and e, contradicting RatHistoryCarrier_not_empty. In the empty branch, the right endpoint comparison is already present. Conversely, $k$hsamee together with the displayed left endpoint comparison selects the empty-empty branch. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty left endpoint iff`

Theorem 45.176 (Rat denominator unit-envelope classifier is continuation-closed)**.** *For histories* h, h', k, k', r, r' : BHist*, if*

$$\mathsf{FieldRatDenominatorUnitEnvelopeClassifier}(h, h'), \quad \mathsf{FieldRatDenominatorUnitEnvelopeClassifier}(k, k'), \quad \mathsf{Cont}(h, k, r$$

then

$$\mathsf{FieldRatDenominatorUnitEnvelopeClassifier}(r, r').$$

Proof. Split the two classifier witnesses. In the carried-carried branch, Rat denominator continuation classifier closure supplies a carried classifier for the two results. In each mixed branch, continuation congruence against the empty endpoint unit identifies the result with the carried endpoint on that side; Rat carrier transport then supplies carried result witnesses, and continuation congruence supplies the result comparison. In the empty-empty branch, both continuations transport to the empty-left unit continuation at e, so both results are history-same to e and the empty branch applies. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier continuation closed`

Theorem 45.177 (Rat denominator continuation adjoined-unit laws)**.** *The concrete carrier* RatDenomUnitCarrier *and classifier* RatDenomUnitClassifier *satisfy the denominator continuation laws needed to use* e *as the endpoint unit:*

1. RatDenomUnitCarrier(e)*.*
2. *If* RatDenomUnitCarrier(h)*,* RatDenomUnitCarrier(k)*, and* Cont(h, k, r)*, then* RatDenomUnitCarrier(r)*.*
3. *If* RatDenomUnitCarrier(h) *and* Cont(e, h, r)*, then* RatDenomUnitClassifier(r, h)*.*
4. *If* RatDenomUnitCarrier(h) *and* Cont(h, e, r)*, then* RatDenomUnitClassifier(r, h)*.*
5. *For all histories* $h, k, \ell, hk, k\ell, left, right$: BHist*, if* RatDenomUnitCarrier(h)*,* RatDenomUnitCarrier(k)*,* RatDenomUnitCarrier(ℓ)*,* Cont(h, k, hk)*,* Cont$(k, \ell, k\ell)$*,* Cont$(hk, \ell, left)$*, and* Cont$(h, k\ell, right)$*, then* RatDenomUnitClassifier$(left, right)$*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation adjoined unit laws`

Proof. The empty endpoint is carried by the left disjunct of Definition 45.154, using reflexivity of hsame at e.

For closure, split both input carrier witnesses into their empty-endpoint and positive-denominator alternatives. In the positive/positive case, the result is carried by `BEDC.Derived.FieldUp.RatContinuation.RatH`

If the left input is hsamee, transport the continuation relation to an empty-left continuation and use `BEDC.FKernel.Cont.cont_left_unit_result`; the result is classified with the right endpoint. If the right endpoint is positive, transport that positive carrier across the resulting hsame comparison by `BEDC.Derived.RatUp.RatHistoryCarrier_hsame_transport`; if it is empty, the result belongs to the empty disjunct. The cases with the right input hsamee are symmetric, using `BEDC.FKernel.Cont.Units.cont_right_unit_result`.

The two displayed unit laws are the corresponding one-sided cases of the same argument. The theorem `BEDC.FKernel.Cont.cont_left_unit_result` gives $r\mathsf{hsame}h$ from $\mathsf{Cont}(\mathsf{e}, h, r)$, and `BEDC.FKernel.Cont.Units.cont_right_unit_result` gives the same comparison from $\mathsf{Cont}(h, \mathsf{e}, r)$. In each case, the carrier part of the classifier is obtained from the assumed carrier for h and the closure argument above for r.

For associativity, first apply closure to the two inner continuations to obtain $\mathsf{RatDenomUnitCarrier}(hk)$ and $\mathsf{RatDenomUnitCarrier}(k\ell)$. Apply closure again to the two outer continuations to obtain carrier witnesses for $left$ and $right$.

Finally use `BEDC.FKernel.Cont.cont_assoc_hsame` on the four displayed continuation witnesses. Its comparison $left\mathsf{hsame}right$, together with the two carrier witnesses just constructed, is exactly $\mathsf{RatDenomUnitClassifier}(left, right)$. □

Theorem 45.178 (Rat denominator unit classifier is closed under continuation)**.** *If two pairs of endpoints are classified by* RatDenomUnitClassifier *and the two displayed continuations match the left endpoints and the right endpoints,*

$$\mathsf{RatDenomUnitClassifier}(h, h') \wedge \mathsf{RatDenomUnitClassifier}(k, k') \wedge \mathsf{Cont}(h, k, r) \wedge \mathsf{Cont}(h', k', r'),$$

then the two continuation results are again classified:

$$\mathsf{RatDenomUnitClassifier}(r, r').$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier continuation closed`

Proof. Use the continuation closure part of Definition 45.177 on the two left endpoints to carry r, and on the two right endpoints to carry r'. The classifier comparisons $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$ transport the two continuations to a common result comparison by continuation congruence. These two carrier witnesses and the resulting $r\mathsf{hsame}r'$ comparison are exactly $\mathsf{RatDenomUnitClassifier}(r, r')$. □

Traditional target: denominator-side continuation with empty endpoint unit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, and Cont; **carrier choice**: RatDenomUnitCarrier; **classifier choice**: RatDenomUnitClassifier; **stability obligations**: empty endpoint coverage, continuation closure, one-sided endpoint laws, and classified triple associativity; **ledger**: continuation witnesses together with the resulting hsame comparisons; **core theorem**: Definition 45.177; **exactness predicate**: both endpoints lie in the empty-unit denominator carrier and are related by hsame; **standard bridge**: denominator continuation with the empty history as endpoint unit; **status**: ProofTarget.

Traditional target: internal unit obstruction for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: RatHistoryClassifier; **stability obstruction**: no carried one-sided unit and no continuation-as-field-multiplication certificate; **ledger**: the attempted unit endpoint; **core theorems**: Definition 45.143, Definition 45.144, Definition 45.147, Definition 45.150, Definition 45.152, and Definition 45.153; **exactness predicate**: failure of internal one-sided unit coverage; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.179 (Rat denominator unit classifier append commutativity)**.** *If two histories* $h, k :$ BHist *are carried by the adjoined empty-unit denominator carrier, then the two append products classify each other:*

$$\mathsf{RatDenomUnitClassifier}\big(\mathsf{append}(h, k), \mathsf{append}(k, h)\big).$$

Proof. Split the two carrier witnesses into the empty endpoint branch and the rational-denominator branch. The empty–empty and mixed cases use the continuation left and right unit laws to compare each append result with the nonempty endpoint, or with e in the empty–empty case. When both endpoints are rational denominator histories, use the denominator continuation commutative classifier theorem on the two canonical append continuations. In every case, continuation closure supplies the carried evidence for both append results. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append comm`

Theorem 45.180 (Rat denominator unit classifier append commutation congruence)**.** *If two endpoint pairs are classified by the adjoined empty-unit denominator classifier, then appending the source endpoints and the reversed target endpoints again produces a classified pair:*

$$\mathsf{RatDenomUnitClassifier}(h, h') \Rightarrow \mathsf{RatDenomUnitClassifier}(k, k') \Rightarrow \mathsf{RatDenomUnitClassifier}(\mathsf{append}(h, k), \mathsf{append}(k', h$$

Proof. Continuation closure first classifies $\mathsf{append}(h, k)$ against $\mathsf{append}(h', k')$ using the canonical append continuations. The append-commutativity classifier applied to the two target carriers classifies $\mathsf{append}(h', k')$ against $\mathsf{append}(k', h')$. Composing the two history-same endpoint comparisons and keeping the carried evidence gives the displayed classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append comm congr`

Theorem 45.181 (Rat denominator unit semantic NameCert)**.** *For carrier, source, pattern, and ledger all equal to* RatDenomUnitCarrier*, and classifier* RatDenomUnitClassifier*, the denominator empty-unit interface satisfies* SemanticNameCert*.*

Proof. The source is inhabited because e belongs to RatDenomUnitCarrier by the empty-unit branch.

Reflexivity, symmetry, transitivity, and carrier transport are the corresponding hsame fields inside RatDenomUnitClassifier.

Pattern soundness and ledger soundness are projections from the source carrier, because source, pattern, and ledger are all the same carrier predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier semanticNameCert`

Theorem 45.182 (Rat denominator unit mixed append Rat closure package)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, denominator-unit endpoints may be appended to genuine Rat endpoints on either side without leaving the Rat carrier or Rat classifier:*

1. *If* $\mathsf{RatHistoryCarrier}(h)$ *and* $\mathsf{RatDenomUnitCarrier}(k)$*, then*

$$\mathsf{RatHistoryCarrier}(\mathsf{append}(h, k)).$$

2. *If* $\mathsf{RatDenomUnitCarrier}(h)$ *and* $\mathsf{RatHistoryCarrier}(k)$*, then*

$$\mathsf{RatHistoryCarrier}(\mathsf{append}(h, k)).$$

3. *If* $\mathsf{RatHistoryClassifier}(h, h')$ *and* $\mathsf{RatDenomUnitClassifier}(k, k')$*, then*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')).$$

4. *If* $\mathsf{RatDenomUnitClassifier}(h, h')$ *and* $\mathsf{RatHistoryClassifier}(k, k')$*, then*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')).$$

Proof. For the right denominator-unit carrier row, split the $\mathsf{RatDenomUnitCarrier}(k)$ witness. In the empty branch, the right append unit law identifies $\mathsf{append}(h, k)$ with h, so the original $\mathsf{RatHistoryCarrier}(h)$ transports to the result. In the genuine Rat branch, Rat denominator continuation closure applies directly to the two Rat carriers and the canonical append continuation.

For the left denominator-unit carrier row, split the $\mathsf{RatDenomUnitCarrier}(h)$ witness. In the empty branch, the left append unit law identifies $\mathsf{append}(h, k)$ with k, so the original $\mathsf{RatHistoryCarrier}(k)$ transports to the result. In the genuine Rat branch, the same Rat denominator continuation closure applies to the two Rat carriers and the canonical append continuation.

For the classifier rows, first apply the corresponding carrier rows to the two endpoints. The remaining hsame field is obtained by cont_respects_hsame from the two input classifier comparisons and the canonical append continuations on the left and right endpoints. The two Rat carrier witnesses and this transported hsame comparison assemble the advertised RatHistoryClassifier outputs. □

Theorem 45.183 (Rat denominator unit classifier append left factor classifier)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, if*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k'))$$

and $h\mathsf{hsame}h'$*, then*

$$\mathsf{RatDenomUnitClassifier}(k, k').$$

Proof. The classifier on the appended histories supplies denominator-unit carriers for both appended endpoints and their hsame comparison. Carrier factorization recovers denominator-unit carriers for k and k'. Replacing the right endpoint by the same left prefix using $h\mathsf{hsame}h'$ leaves a shared-prefix append comparison, and finite-kernel left cancellation gives $k\mathsf{hsame}k'$. These three fields assemble the suffix classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append left factor classifier`

Theorem 45.184 (Rat denominator unit classifier append prefix factor classifiers)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, if the appended endpoints are classified and the prefix factors are* hsame*, then both factor pairs are classified:*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')) \Longrightarrow h\mathsf{hsame}h' \Longrightarrow \mathsf{RatDenomUnitClassifier}(h, h') \wedge \mathsf{RatDenomUn}$$

Proof. The appended classifier factors the two product carriers into denominator-unit carriers for h, h', k, k'. The supplied prefix sameness, together with the two prefix carrier witnesses, forms the prefix classifier. The suffix classifier is obtained by the append left factor classifier theorem, which cancels the shared prefix after transporting along $h\mathsf{hsame}h'$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append prefix factor classifiers`

Theorem 45.185 (Rat denominator unit classifier append factor hsame iff)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, if*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')),$$

then

$$h\mathsf{hsame}h' \Longleftrightarrow k\mathsf{hsame}k'.$$

Proof. The forward direction applies the append left factor classifier theorem and projects the suffix hsame field from the resulting classifier. The reverse direction applies the corresponding right factor classifier theorem and projects the prefix hsame field. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append factor hsame iff`

Theorem 45.186 (Rat denominator unit append nonempty Rat endpoint pair)**.** *For histories* $h, k : \mathsf{BHist}$*, if both factors lie in* $C^u_D = \mathsf{RatDenomUnitCarrier}$ *and* $\mathsf{append}(h, k)$ *is not empty, then the appended history is a genuine Rat history and at least one factor is already a genuine Rat history:*

$$C^u_D(h) \Longrightarrow C^u_D(k) \Longrightarrow (\mathsf{append}(h,k)\mathsf{hsamee} \Longrightarrow \bot) \Longrightarrow \mathsf{RatHistoryCarrier}(\mathsf{append}(h,k)) \wedge (\mathsf{RatHistoryCarrier}(h) \vee \mathsf{Rat}$$

Proof. Denominator-unit continuation closure first puts $\mathsf{append}(h, k)$ in C^u_D. Since the appended endpoint is not empty, the empty branch of C^u_D is impossible, so the result carrier is a $\mathsf{RatHistoryCarrier}$ witness. The carrier-level nonempty factor cases then split the two factors: either h is a Rat history and k remains in C^u_D, or h remains in C^u_D and k is a Rat history. Projecting the Rat side gives the endpoint disjunction. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append nonempty rat endpoint pair`

Theorem 45.187 (Rat denominator unit classifier append nonempty factor carriers)**.** *For histories* $h, h', k, k' : \mathsf{BHist}$*, if the appended endpoints are classified by* R^u_D *and the left appended endpoint is nonempty, then both classified endpoints split into denominator-unit carried factors with a genuine Rat denominator endpoint on at least one side:*

$$R^u_D(\mathsf{append}(h,k), \mathsf{append}(h',k')) \Longrightarrow (\mathsf{append}(h,k)\mathsf{hsamee} \Longrightarrow \bot) \Longrightarrow \begin{array}{l}(\mathsf{RatHistoryCarrier}(h) \wedge C^u_D(k) \vee C^u_D(h) \wedge \mathsf{Ra} \\ \wedge\ (\mathsf{RatHistoryCarrier}(h') \wedge C^u_D(k') \vee C^u_D(h')\end{array}$$

Proof. Apply the carrier-level nonempty factor cases theorem to the left endpoint. The classifier sameness field transports the nonempty condition to the right endpoint: if the right append were empty, transitivity would make the left append empty. Apply the same carrier-level theorem to the right endpoint and pair the two decompositions. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append nonempty factor carriers`

Theorem 45.188 (Rat denominator unit classifier two-sided context cancellation package)**.** *For histories* $L, R, Q, S : \mathsf{BHist}$*, if* $\mathsf{RatDenomUnitClassifier}(L, R)$*, then*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(Q,L), \mathsf{append}(S,R)) \Longleftrightarrow \mathsf{RatDenomUnitClassifier}(Q,S)$$

and

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(L,Q), \mathsf{append}(R,S)) \Longleftrightarrow \mathsf{RatDenomUnitClassifier}(Q,S).$$

Proof. For the suffix-context statement, apply Definition 45.415 to $\mathsf{append}(Q, L)$ and $\mathsf{append}(S, R)$. The assumed classifier on L, R is exactly the classified context required by that checked row, so the result is the first displayed equivalence.

For the prefix-context statement, apply Definition 45.416 to $\mathsf{append}(L, Q)$ and $\mathsf{append}(R, S)$. The same classified context L, R supplies the prefix certificate, and the checked row returns the second displayed equivalence.

Pairing the two equivalences gives the advertised two-sided cancellation package. □

Traditional target: two-sided classified denominator context cancellation; **BEDC source**: BHist plus RatDenomUnitCarrier, RatDenomUnitClassifier, and append; **carrier choice**: C_D^u; **classifier choice**: R_D^u; **stability obligations**: suffix and prefix context cancellation; **ledger**: the classified context pair L, R; **core theorem**: Definition 45.188; **exactness predicate**: the displayed pair of iff statements; **standard bridge**: ordinary cancellation by a classified denominator-unit context; **status**: ProofTarget.

Traditional target: denominator-side continuation with empty endpoint unit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, and Cont; **carrier choice**: RatDenomUnitCarrier; **classifier choice**: RatDenomUnitClassifier; **stability obligations**: empty endpoint coverage, continuation closure, one-sided endpoint laws, and classified triple associativity; **ledger**: continuation witnesses together with the resulting hsame comparisons; **core theorem**: Definition 45.177; **exactness predicate**: both endpoints lie in the empty-unit denominator carrier and are related by hsame; **standard bridge**: denominator continuation with the empty history as endpoint unit; **status**: ProofTarget.

Traditional target: internal unit obstruction for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier plus Cont; **carrier choice**: Rat denominators; **classifier choice**: RatHistoryClassifier; **stability obstruction**: no carried one-sided unit and no continuation-as-field-multiplication certificate; **ledger**: the attempted unit endpoint; **core theorems**: Definition 45.143, Definition 45.144, Definition 45.147, Definition 45.150, Definition 45.152, and Definition 45.153; **exactness predicate**: failure of internal one-sided unit coverage; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.189 (Rat denominator unit carrier transport-closure package)**.** *For concrete histories h, k, r : BHist, the empty-unit denominator carrier has the following two concrete closure properties:*

1. *if* $\mathsf{RatDenomUnitCarrier}(h)$ *and* $h\mathsf{hsame}k$*, then* $\mathsf{RatDenomUnitCarrier}(k)$*;*
2. *if* $\mathsf{RatDenomUnitCarrier}(h)$*,* $\mathsf{RatDenomUnitCarrier}(k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{RatDenomUnitCarrier}(r)$*.*

Proof. Unfold Definition 45.154. For the transport clause, split the carrier witness for h. In the Rat branch, `BEDC.Derived.RatUp.RatHistoryCarrier_hsame_transport` carries the Rat endpoint witness across $h\mathsf{hsame}k$. In the empty branch, transport the empty-endpoint comparison across $h\mathsf{hsame}k$; the result belongs to the left disjunct for k.

For continuation closure, split the two input carrier witnesses. In the empty/empty branch, transport the displayed continuation to the empty-left and empty-right unit continuation at e, so the result is hsame to e. This is the left disjunct of the unit carrier.

In the empty/Rat branch, transport the continuation to an empty-left continuation and use the left unit result law to compare r with the Rat endpoint. Rat carrier transport then gives the Rat branch for r.

The Rat/empty branch is symmetric: transport to an empty-right continuation, use the right unit result law, and carry the Rat witness across the resulting hsame comparison.

In the Rat/Rat branch, apply the concrete Rat denominator continuation carrier closure theorem. The result is a Rat endpoint, hence it lies in the right disjunct of $\mathsf{RatDenomUnitCarrier}(r)$. □

Theorem 45.190 (Rat denominator unit carrier zero-headed absurd)**.** *For every tail t : BHist, the empty-unit denominator carrier cannot hold on a zero-headed history:*

$$\mathsf{RatDenomUnitCarrier}(\mathsf{E0}(t)) \Longrightarrow \bot.$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier e0 absurd`

Proof. Unfold the empty-unit denominator carrier. Its empty-endpoint branch would make $\mathsf{E0}(t)$ equal to e, contradicting constructor disjointness. Its Rat branch is impossible because the Rat denominator carrier has no zero-headed inhabitant. □

Theorem 45.191 (Rat denominator unit classifier empty endpoint iff)**.** *For every history h : BHist, the empty-unit denominator classifier against the empty endpoint records exactly the displayed empty endpoint comparison, on either side:*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{e}, h) \iff h\mathsf{hsamee}, \qquad \mathsf{RatDenomUnitClassifier}(h, \mathsf{e}) \iff h\mathsf{hsamee}.$$

Proof. Unfold the classifier into its two carrier fields and its endpoint sameness field. In the forward directions, the endpoint field gives the desired comparison, using symmetry for the left endpoint placement. In the reverse directions, the displayed empty comparison supplies the nonempty endpoint carrier by the empty branch, while the empty endpoint itself uses reflexivity. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier empty endpoint iff`

Theorem 45.192 (Rat denominator unit carrier append zero-headed absurd)**.** *For histories $h, k, tail$: BHist, two empty-unit denominator carrier witnesses cannot have a product classified with a zero-headed endpoint:*

$$\mathsf{RatDenomUnitCarrier}(h) \to \mathsf{RatDenomUnitCarrier}(k) \to \mathsf{append}(h, k)\mathsf{hsameE0}(tail) \to \bot.$$

Proof. Closure of $\mathsf{RatDenomUnitCarrier}$ under continuation gives a carrier witness for $\mathsf{append}(h, k)$. Transport this carrier witness along the displayed endpoint sameness to obtain $\mathsf{RatDenomUnitCarrier}(\mathsf{E0}(tail))$, contradicting the zero-headed absurdity theorem above. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append e0 absurd`

Theorem 45.193 (Rat denominator unit carrier append one-headed result cases)**.** *For histories $h, k, tail$: BHist, if h and k are carried by the empty-unit denominator carrier and their append is one-headed, then either the right factor is empty and the left factor accounts for the one-headed result, or the right factor itself is one-headed with a unary exposed tail:*

$$\begin{aligned}
&\mathsf{RatDenomUnitCarrier}(h) \to \mathsf{RatDenomUnitCarrier}(k) \to \mathsf{append}(h, k)\mathsf{hsameE1}(tail) \\
&\quad \to (k\mathsf{hsamee} \land \mathsf{RatDenomUnitCarrier}(\mathsf{E1}(tail)) \land h\mathsf{hsameE1}(tail)) \\
&\qquad \lor \exists k_0 : \mathsf{BHist}.\ k = \mathsf{E1}(k_0) \land \mathsf{UnaryHistory}(k_0) \land \mathsf{RatDenomUnitCarrier}(tail) \land \mathsf{Cont}(h, k_0, tail).
\end{aligned}$$

Proof. Read the append comparison as a continuation into a one-headed result and apply the continuation one-headed-result inversion. The empty-right branch closes the displayed one-headed carrier by denominator-unit continuation closure. In the headed-right branch, the carrier on $k = \mathsf{E1}(k_0)$ gives a unary witness for k_0; visible carrier cases make k_0 carried, and continuation closure then carries the exposed tail. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append e1 result cases`

Theorem 45.194 (Rat denominator unit carrier one-headed unary iff)**.** *For every tail t : BHist, the empty-unit denominator carrier sees a one-headed history exactly when the tail is unary:*

$$\mathsf{RatDenomUnitCarrier}(\mathsf{E1}(t)) \iff \mathsf{UnaryHistory}(t).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier e1 tail unary iff`

Proof. Unfold the empty-unit denominator carrier. Its empty-endpoint branch would give $\mathsf{E1}(t)\mathsf{hsamee}$, contradicting constructor disjointness. The remaining branch is $\mathsf{RatHistoryCarrier}(\mathsf{E1}(t))$, which is equivalent to $\mathsf{UnaryHistory}(t)$ by the one-headed Rat carrier characterization. □

Theorem 45.195 (Rat denominator unit classifier one-headed tail iff)**.** *For histories $d, e : \mathsf{BHist}$, the empty-unit denominator classifier on two one-headed histories is exactly the pair of unary tail witnesses together with tail sameness:*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{E1}(d), \mathsf{E1}(e)) \iff \mathsf{UnaryHistory}(d) \wedge \mathsf{UnaryHistory}(e) \wedge d\mathsf{hsamee}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier e1 tail iff`

Proof. Unfold the classifier into its two empty-unit carrier fields and the endpoint sameness field. Each carrier field is converted by the one-headed carrier characterization above, and the endpoint sameness field is converted by constructor injectivity for $\mathsf{E1}$. □

Theorem 45.196 (Rat denominator unit classifier displayed context one-tail readback)**.** *Let $d, e, p_d, p_e, t_d, t_e, m_d, m_e, out_d, out_e, a, b : \mathsf{BHist}$. If $\mathsf{RatDenomUnitClassifier}(d, e)$, $\mathsf{RatDenomUnitClassifier}(p_d, p_e)$, and $\mathsf{RatDenomUnitClassifier}(t_d, t_e)$ hold, and the displayed continuations $\mathsf{Cont}(p_d, d, m_d)$, $\mathsf{Cont}(m_d, t_d, out_d)$, $\mathsf{Cont}(p_e, e, m_e)$, and $\mathsf{Cont}(m_e, t_e, out_e)$ expose contextual endpoints satisfying $\mathsf{hsame}(out_d, \mathsf{E1}(a))$ and $\mathsf{hsame}(out_e, \mathsf{E1}(b))$, then*

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Compose the prefix classifier with the core denominator-unit classifier through the first pair of continuations, then compose the resulting middle classifier with the tail classifier through the second pair. This gives a $\mathsf{RatDenomUnitClassifier}(out_d, out_e)$ witness.

Transport that classifier along the endpoint displays $\mathsf{hsame}(out_d, \mathsf{E1}(a))$ and $\mathsf{hsame}(out_e, \mathsf{E1}(b))$. The result is a classifier between $\mathsf{E1}(a)$ and $\mathsf{E1}(b)$. Apply Definition 45.195 in the forward direction to read back the two unary tail witnesses and their hsame link. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier cont context e1 pair readback`

Theorem 45.197 (Rat denominator unit classifier visible cases iff)**.** *For histories $h, k : \mathsf{BHist}$, the empty-unit denominator classifier is equivalent to either the empty–empty branch or a matched pair of one-headed unary denominator tails:*

$$\begin{aligned}
&\mathsf{RatDenomUnitClassifier}(h, k) \\
&\iff (h\mathsf{hsamee} \wedge k\mathsf{hsamee}) \\
&\quad \vee \exists h_t, k_t : \mathsf{BHist}.\ h = \mathsf{E1}(h_t) \wedge k = \mathsf{E1}(k_t) \wedge \mathsf{UnaryHistory}(h_t) \\
&\qquad \wedge \mathsf{UnaryHistory}(k_t) \wedge h_t\mathsf{hsame}k_t.
\end{aligned}$$

Proof. Unpack the classifier into two carrier witnesses and an endpoint comparison. Each carrier is split by the visible carrier equivalence. Mixed empty and one-headed endpoints contradict constructor disjointness. The one-headed pair branch converts endpoint sameness through $\mathsf{E1}$ injectivity, and the reverse direction rebuilds the two carriers and the classifier comparison from the displayed tail data. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier visible cases iff`

45.5 The certificate

Definition 45.198 (Field source specification)**.** *The source is a certified commutative ring carrier plus constructive apartness from zero, packaged through the derived predicate* $\mathsf{NonZero}_F$*.*

Definition 45.199 (Field pattern specification)**.** *The pattern is commutative-ring expression formation extended by inverse formation only from a pair* (a, p) *with* $p : \mathsf{NonZero}_F(a)$*.*

Definition 45.200 (Field classifier specification)**.** *The classifier is ring sameness* $\sim_F$*. Nonzero evidence is stable under classifier replacement: from* $a \sim_F b$ *and* $p : \mathsf{NonZero}_F(a)$ *the certificate supplies transported evidence* $q : \mathsf{NonZero}_F(b)$*.*

Definition 45.201 (Field stability certificate)**.** *The stability certificate consists of:*

1. *all commutative-ring stability fields*
2. *contradiction of* $\mathsf{NonZero}_F(0)$
3. *stability of* $\mathsf{NonZero}_F$ *under the carrier classifier*
4. *inverse closure for explicit witnesses* $p : \mathsf{NonZero}_F(a)$
5. *left and right multiplicative inverse laws indexed by each witness* $p : \mathsf{NonZero}_F(a)$

Theorem 45.202 (Field inverse congruence from apartness transport)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate supplying multiplication on* BHist*, a unit* 1*, a unary nonzero certificate* $\mathsf{NonZero}_F$*, an inverse operation* $\mathsf{inv}(a, p)$ *indexed by* $p : \mathsf{NonZero}_F(a)$*, multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, if* $a\mathsf{hsame}b$*,* $p : \mathsf{NonZero}_F(a)$*, and* $q : \mathsf{NonZero}_F(b)$ *is any available witness, then* $\mathsf{inv}(a, p)\mathsf{hsameinv}(b, q)$*. When* q *is obtained from* p *along* $a\mathsf{hsame}b$*, the certificate transport clause supplies that witness.*

Proof. Transport the left inverse law for $\mathsf{inv}(a, p)$ across $a\mathsf{hsame}b$ to obtain a left inverse witness over b. Combine this transported witness with the right inverse law for $\mathsf{inv}(b, q)$ at b. The group left/right inverse uniqueness argument, applied inside the multiplicative group of nonzero field elements, identifies the transported inverse with the chosen inverse at b. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse congruence from apartness`

Theorem 45.203 (Field inverse cancel from apartness-indexed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate supplying multiplication on* BHist*, a unit* 1*, a unary nonzero certificate* $\mathsf{NonZero}_F$*, an inverse operation* $\mathsf{inv}(a, p)$ *indexed by* $p : \mathsf{NonZero}_F(a)$*, multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, if* $p : \mathsf{NonZero}_F(a)$*,* $q : \mathsf{NonZero}_F(b)$*, and* $\mathsf{inv}(a, p)\mathsf{hsameinv}(b, q)$*, then* $a\mathsf{hsame}b$*.*

Proof. Transport the left inverse law for a across the equality of proof-indexed inverses to make a a right inverse of the common inverse $\mathsf{inv}(b, q)$. The right inverse law for b makes b the matching left inverse. Group left/right inverse uniqueness identifies b with a, and symmetry gives $a\mathsf{hsame}b$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse cancel from apartness`

Theorem 45.204 (Field inverse empty endpoint reflection from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate supplying multiplication on* BHist*, a unit* 1*, a unary nonzero certificate* $\mathsf{NonZero}_F$*, and proof-indexed inverses with multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and indexed left and right inverse laws, equality of two indexed inverse histories reflects empty-endpoint membership of the original arguments:*

$$\mathsf{inv}(a, p_a)\mathsf{hsameinv}(b, p_b) \Longrightarrow (a\mathsf{hsamee} \Longleftrightarrow b\mathsf{hsamee}).$$

Proof. Apply inverse cancellation from apartness to the displayed equality of indexed inverses, obtaining $a\mathsf{hsame}b$. The forward empty-endpoint direction transports b back through the symmetric comparison with a; the reverse direction transports a forward through the same comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse empty endpoint reflection from apartness`

Theorem 45.205 (Field inverse classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate supplying multiplication on* BHist*, a unit* 1*, a unary nonzero certificate* $\mathsf{NonZero}_F$*, an inverse operation* $\mathsf{inv}(a, p)$ *indexed by* $p : \mathsf{NonZero}_F(a)$*, multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, any elements* $a, b : \mathsf{BHist}$ *with explicit witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *satisfy*

$$a\mathsf{hsame}b \Longleftrightarrow \mathsf{inv}(a, p_a)\mathsf{hsameinv}(b, p_b).$$

Proof. For the forward direction, apply Definition 45.202 to $a\mathsf{hsame}b$, using the displayed witnesses p_a and p_b as the two proof indices. This gives the classifier equality of the two indexed inverses.

For the reverse direction, apply Definition 45.203 to the supplied equality between indexed inverses. Its cancellation conclusion is exactly $a\mathsf{hsame}b$, so the two implications combine into the stated classifier exactness. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse empty unit classifier exact from apartness iff`

Theorem 45.206 (Field apart-zero inverse classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate supplying multiplication on* BHist*, a unit* 1*, and proof-indexed inverses over the concrete apart-zero predicate* $(x\mathsf{hsamee}) \to \bot$*, multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by apart-zero witnesses imply that any* $a, b : \mathsf{BHist}$ *with witnesses* $p_a : (a\mathsf{hsamee}) \to \bot$ *and* $p_b : (b\mathsf{hsamee}) \to \bot$ *satisfy*

$$a\mathsf{hsame}b \Longleftrightarrow \mathsf{inv}(a, p_a)\mathsf{hsameinv}(b, p_b).$$

Proof. For the forward direction, use inverse congruence from apartness with $\mathsf{NonZero}_F(x)$ instantiated by the concrete predicate $(x\mathsf{hsamee}) \to \bot$. For the reverse direction, use inverse cancellation from apartness under the same instantiation. The two implications give the displayed exact classifier equivalence for apart-zero indexed inverses. □

Lean4 ✓ `BEDC.Derived.FieldUp.field apart zero inverse classifier exact from apartness`

Theorem 45.207 (Field inverse is nonzero under one-apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication* mul*, zero* 0*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, multiplication congruence under* hsame*, zero-left multiplication* $\mathsf{mul}(0, x)\mathsf{hsame}0$*, left inverse law* $\mathsf{mul}(\mathsf{inv}(a, p), a)\mathsf{hsame}1$*, one-apartness* $\omega : (1\mathsf{hsame}0) \to \bot$*, and conversion maps* $\alpha_x : ((x\mathsf{hsame}0) \to \bot) \to \mathsf{NonZero}_F(x)$*, every witness* $p_a : \mathsf{NonZero}_F(a)$ *canonically determines*

$$\mathsf{NonZero}_F(\mathsf{inv}(a, p_a)).$$

Proof. Fix $p_a : \mathsf{NonZero}_F(a)$. It is enough to build a contradiction from a hypothetical equality $\mathsf{inv}(a, p_a)\mathsf{hsame}0$, because $\alpha_{\mathsf{inv}(a,p_a)}$ converts that negated zero equality into the required $\mathsf{NonZero}_F$ witness.

Assume $h : \mathsf{inv}(a, p_a)\mathsf{hsame}0$. Multiplication congruence transports h after right multiplication by a, giving $\mathsf{mul}(\mathsf{inv}(a, p_a), a)\mathsf{hsamemul}(0, a)$. The zero-left law gives $\mathsf{mul}(0, a)\mathsf{hsame}0$, so transitivity yields $\mathsf{mul}(\mathsf{inv}(a, p_a), a)\mathsf{hsame}0$.

The left inverse law gives $\mathsf{mul}(\mathsf{inv}(a, p_a), a)\mathsf{hsame}1$. Composing this equality with the equality just obtained gives $1\mathsf{hsame}0$, contradicting ω. Thus the assumed zero equality has a constructive refutation, and applying $\alpha_{\mathsf{inv}(a,p_a)}$ produces $\mathsf{NonZero}_F(\mathsf{inv}(a, p_a))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse nonzero from one apartness`

Theorem 45.208 (Field inverse is apart from empty under one-apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication* mul*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, multiplication congruence under* hsame*, zero-left multiplication* $\mathsf{mul}(0, x)\mathsf{hsame}0$*, left inverse law* $\mathsf{mul}(\mathsf{inv}(a, p), a)\mathsf{hsame}1$*, and one-apartness* $\omega : (1\mathsf{hsame}0) \to \bot$*, every witness* $p_a : \mathsf{NonZero}_F(a)$ *constructively refutes*

$$\mathsf{inv}(a, p_a)\mathsf{hsame}0.$$

Proof. Assume $\mathsf{inv}(a, p_a)\mathsf{hsame}0$. Multiplication congruence transports this equality through right multiplication by a, and the zero-left law identifies the transported product with 0. The left inverse law identifies the same product with 1, so transitivity gives $1\mathsf{hsame}0$, contradicting ω. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse apart empty from one apartness`

Theorem 45.209 (Field inverse of an indexed nonzero inverse)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with multiplication on* BHist*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, associativity, left and right identity, multiplication congruence, and left inverse laws indexed by nonzero witnesses, if* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_{\mathsf{inv}} : \mathsf{NonZero}_F(\mathsf{inv}(a, p_a))$*, then*

$$\mathsf{inv}(\mathsf{inv}(a, p_a), p_{\mathsf{inv}})\mathsf{hsame}a.$$

Moreover, if this double inverse collapsed to e*, then* a *would collapse to* e*, contradicting the supplied nonzero-empty separation for* p_a*.*

Proof. Set the middle element in the inverse-uniqueness argument to $\mathsf{inv}(a, p_a)$. The left inverse law for p_{inv} gives

$$\mathsf{inv}(\mathsf{inv}(a, p_a), p_{\mathsf{inv}})\mathsf{inv}(a, p_a)\mathsf{hsame}1.$$

The left inverse law for p_a gives

$$\mathsf{inv}(a, p_a)a\mathsf{hsame}1.$$

Associativity, multiplication congruence, and the identity laws give the standard left/right inverse uniqueness chain $x\mathsf{hsame}x\ 1\mathsf{hsame}x(yz)\mathsf{hsame}(xy)z\mathsf{hsame}1\ z\mathsf{hsame}z$, with $x = \mathsf{inv}(\mathsf{inv}(a, p_a), p_{\mathsf{inv}})$, $y = \mathsf{inv}(a, p_a)$, and $z = a$. Hence the two outer witnesses are identified, which is the stated equality. A candidate collapse of the double inverse to e transports along the symmetric equality to a collapse $a\mathsf{hsame}\mathsf{e}$, and this contradicts the nonzero-empty separation for p_a. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse involutive nonzero from apartness`

Theorem 45.210 (Field inverse is a certified automorphism of the nonzero domain)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with proof-indexed inverse* $\mathsf{inv}(x, p)$*, inverse laws indexed by nonzero witnesses, multiplication congruence under* hsame*, zero-left multiplication, one-apartness* $\omega : (1\mathsf{hsame}0) \to \bot$*, and conversion maps*

$$\alpha_x : ((x\mathsf{hsame}0) \to \bot) \to \mathsf{NonZero}_F(x),$$

every witness $p_a : \mathsf{NonZero}_F(a)$ *determines*

$$A := \mathsf{inv}(a, p_a)$$

and a witness $p_A : \mathsf{NonZero}_F(A)$ *such that*

$$\mathsf{inv}(A, p_A)\mathsf{hsame}a.$$

Moreover, for any $p_b : \mathsf{NonZero}_F(b)$*,*

$$\mathsf{inv}(a, p_a)\mathsf{hsame}\mathsf{inv}(b, p_b) \iff a\mathsf{hsame}b.$$

Proof. First construct the nonzero witness for the image. Apply Definition 45.207 to p_a, using the given multiplication congruence, zero-left multiplication, left inverse law, one-apartness, and the conversion maps α_x. This gives $p_A : \mathsf{NonZero}_F(A)$ for $A := \mathsf{inv}(a, p_a)$.

With this witness in hand, apply Definition 45.209 to p_a and p_A. The conclusion is exactly $\mathsf{inv}(A, p_A)\mathsf{hsame}a$, so inverse carries the nonzero domain back to the original element.

For the classifier equivalence, apply Definition 45.205 to the two explicit nonzero witnesses p_a and p_b. Its two implications give $a\mathsf{hsame}b$ exactly when the indexed inverse images are hsame-equal, which is the stated certified automorphism behavior on the nonzero domain. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse nonzero domain automorphism`

Theorem 45.211 (Field apart-zero inverse involution)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate whose inverse is indexed by explicit apartness from zero, suppose multiplication is associative, has left and right unit laws, respects* hsame*, and has a left inverse law for every apart-zero witness. If the inverse operation sends apart-zero inputs to apart-zero outputs, then for every* $a : \mathsf{BHist}$ *and* $p_a : a\mathsf{hsame}0 \to \bot$*,*

$$\mathsf{inv}(\mathsf{inv}(a, p_a), p_{\mathsf{inv}})\mathsf{hsame}a,$$

where p_{inv} *is the supplied apart-zero witness for* $\mathsf{inv}(a, p_a)$*.*

Proof. Apply group left/right inverse uniqueness with middle element $\mathsf{inv}(a, p_a)$. The indexed left inverse law at the outer inverse gives the left inverse equality against the middle element, and the indexed left inverse law at a gives the right inverse equality for the same middle element. The uniqueness chain identifies the two outer endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.field apartzero inverse involutive`

Theorem 45.212 (Field apart-zero inverse classifier automorphism)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with apart-zero indexed inverse* $\mathsf{inv}(a, p)$*, suppose multiplication is associative, has left and right unit laws, respects* hsame*, and has left and right inverse laws for every apart-zero witness. If an inverse-apartness row sends every apart-zero input to an apart-zero inverse, then every* $a, b : \mathsf{BHist}$ *and witnesses* $p_a : (a\mathsf{hsamee}) \to \bot$ *and* $p_b : (b\mathsf{hsamee}) \to \bot$ *satisfy*

$$a\mathsf{hsame}b \iff \mathsf{inv}(a, p_a)\mathsf{hsame}\mathsf{inv}(b, p_b),$$

and, for p_{inv} supplied by inverse-apartness,

$$\mathsf{inv}(\mathsf{inv}(a, p_a), p_{\mathsf{inv}})\mathsf{hsame}a.$$

Per object, the traditional target is inverse as an involutive automorphism of nonzero field elements; the BEDC source is $\mathsf{Field}^{\uparrow}$ *over* BHist*; the carrier is apart-zero histories; the classifier is* hsame*; the stability rows are exactness, inverse-apartness, and involution; the ledger records the proof indices; the downstream consumer is the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *inverse and affine exit ledgers; the status is* $\mathsf{ProofTarget}$.

Proof. For classifier preservation and reflection, apply Definition 45.206 to the two displayed apart-zero witnesses. Its two directions give exactly the equivalence between $a\mathsf{hsame}b$ and equality of the indexed inverse images.

For the involution row, use the inverse-apartness hypothesis to obtain p_{inv} for $\mathsf{inv}(a, p_a)$, then apply Definition 45.211. The conclusion is the second displayed equality, so the package contains both classifier exactness and double-inverse closure on the concrete apart-zero domain. □

Theorem 45.213 (Field inverse product reversal from apartness-indexed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(ab)$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *determine the product inverse by*

$$\mathsf{inv}(ab, p_{ab})\mathsf{hsame}\mathsf{inv}(b, p_b)\mathsf{inv}(a, p_a).$$

Proof. The reversed product $\mathsf{inv}(b, p_b)\mathsf{inv}(a, p_a)$ is a right inverse of ab: reassociate to expose $b\mathsf{inv}(b, p_b)$, collapse it to the unit, and finish with the right inverse law for a. The indexed left inverse of ab and group uniqueness of left and right inverse witnesses identify the chosen inverse of ab with this reversed product. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product reverse from apartness`

Theorem 45.214 (Field inverse product reversal right inverse excludes empty)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, assume associativity, left identity, multiplication congruence, indexed right inverse laws, and one-apartness from empty. For* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*,*

$$\mathsf{mul}(\mathsf{mul}(a, b), \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)))\mathsf{hsame}0 \to \bot.$$

Proof. Reassociate the product to expose $b\mathsf{inv}(b, p_b)$ inside the middle factor. The right inverse law for b reduces that middle product to $\mathsf{inv}(a, p_a)$ by the left identity, and multiplication congruence then reduces the whole expression to $a\mathsf{inv}(a, p_a)$. The right inverse law for a identifies the product with one, so any classifier equality from the product to empty would transport one to empty, contradicting one-apartness. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product reverse right inverse apart empty from one apartness`

Theorem 45.215 (Field inverse product reversal preserves nonzero from one apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, assume one is apart from empty, left zero absorption holds for multiplication, apart-empty evidence produces* $\mathsf{NonZero}_F$*, and* $\mathsf{NonZero}_F$ *transports along* hsame*. If* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$*, then*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))).$$

Proof. The inverse of $\mathsf{mul}(a, b)$ is nonzero: if it classified as empty, left multiplication by ab would classify its inverse product as empty, contradicting the indexed inverse law and the apartness of one from empty. Definition 45.213 identifies this inverse with the reversed product $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$. Transporting the nonzero witness across that classifier equality gives the displayed witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product reverse nonzero from one apartness`

Theorem 45.216 (Field inverse product reversal is apart from empty)**.** *Under the hypotheses of Definition 45.215, if the field certificate also converts every* $\mathsf{NonZero}_F(x)$ *witness into a refutation of* $x\mathsf{hsame}0$*, then the reversed inverse product satisfies*

$$\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))\mathsf{hsame}0 \to \bot.$$

Proof. Apply Definition 45.215 to obtain $\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)))$. The nonzero-to-empty-apartness bridge then rejects any classifier equality from that product to empty. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product reverse apart empty from one apartness`

Theorem 45.217 (Field inverse product reversal apartness transports across products)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, suppose*

$$\mathsf{mul}(a, b)\mathsf{hsamemul}(a', b').$$

Given nonzero witnesses for both products and for all four factors, apartness from empty of $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$ *transports to apartness from empty of* $\mathsf{mul}(\mathsf{inv}(b', p_{b'}), \mathsf{inv}(a', p_{a'}))$*.*

Proof. Definition 45.213 identifies each reversed inverse product with the inverse of its corresponding product. The product classifier equality transports between those two product inverses by inverse congruence. Symmetry and transitivity of hsame therefore identify the two reversed inverse products, so any classifier of the target reversed product as empty would classify the source reversed product as empty and contradict the supplied source apartness. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product reverse apart empty transport from product hsame`

Theorem 45.218 (Field inverse product classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.305, Definition 45.205, and Definition 45.213, assume a bridge*

$$\alpha : (h\mathsf{hsame}0 \to \bot) \to \mathsf{NonZero}_F(h)$$

and witnesses $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*, and* $p_d : \mathsf{NonZero}_F(d)$*. Let* p_{ab} *and* p_{cd} *be the product nonzero witnesses for* $\mathsf{mul}(a, b)$ *and* $\mathsf{mul}(c, d)$ *supplied by Definition 45.305. Then*

$$\mathsf{mul}(a, b)\mathsf{hsamemul}(c, d) \iff \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))\mathsf{hsamemul}(\mathsf{inv}(d, p_d), \mathsf{inv}(c, p_c)).$$

Proof. First build the product witnesses. Apply Definition 45.305 to p_a and p_b, using the bridge α, to obtain $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$. Apply the same theorem to p_c and p_d to obtain $p_{cd} : \mathsf{NonZero}_F(\mathsf{mul}(c, d))$.

For the forward direction, apply Definition 45.205 to the two product endpoints with witnesses p_{ab} and p_{cd}. The resulting inverse equality has endpoints $\mathsf{inv}(\mathsf{mul}(a, b), p_{ab})$ and $\mathsf{inv}(\mathsf{mul}(c, d), p_{cd})$.

Apply Definition 45.213 on both sides, and transport across these two reversal equalities by symmetry and transitivity of hsame.

For the reverse direction, start with the displayed reversed-product equality. Use the same two product reversal equalities, now in the opposite transport orientation, to obtain $\mathsf{inv}(\mathsf{mul}(a,b),p_{ab})\mathsf{hsameinv}(\mathsf{mul}(c,d)$ The reverse implication of Definition 45.205 then gives $\mathsf{mul}(a,b)\mathsf{hsamemul}(c,d)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse product classifier exact from apartness`

Theorem 45.219 (Field apart-product inverse classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and proof-indexed left and right inverse laws, assume a bridge*

$$\alpha : (h\mathsf{hsame}0 \to \bot) \to \mathsf{NonZero}_F(h).$$

If $\mathsf{mul}(a,b)$ *and* $\mathsf{mul}(c,d)$ *are explicitly apart from* e *and* p_a, p_b, p_c, p_d *are nonzero witnesses for the four factors, then*

$$\mathsf{mul}(a,b)\mathsf{hsamemul}(c,d) \iff \mathsf{mul}(\mathsf{inv}(b,p_b),\mathsf{inv}(a,p_a))\mathsf{hsamemul}(\mathsf{inv}(d,p_d),\mathsf{inv}(c,p_c)).$$

Proof. Apply α to the two explicit product-apartness hypotheses to obtain the indexed nonzero witnesses for $\mathsf{mul}(a,b)$ and $\mathsf{mul}(c,d)$. The inverse classifier exactness theorem converts product equality into equality of the two certified product inverses, and product inverse reversal identifies those certified inverses with the two displayed reversed inverse products. The reverse implication uses the same two reversal equalities in the opposite transport orientation before applying inverse cancellation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field apart product inverse classifier exact from apartness`

Per-object certificate: Traditional target: product inverse classifier exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist plus product nonzero. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: inverse exactness, product nonzero construction, product reversal, and hsame transport. Ledger: four factor witnesses p_a, p_b, p_c, p_d and two product witnesses p_{ab}, p_{cd}. Core theorem list: Theorem 45.218. Exactness predicate: the displayed iff. Standard bridge: deferred. Status: ProofTarget.

Theorem 45.220 (Field inverse three-factor product reversal from apartness-indexed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with multiplication associativity, multiplication congruence, left and right identity, proof-indexed inverse laws, product nonzero exclusion, and conversion maps*

$$\alpha_{ab} : \mathsf{ApartZero}_F(\mathsf{mul}(a,b)) \to \mathsf{NonZero}_F(\mathsf{mul}(a,b)), \quad \alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a),$$

$$\alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b), \quad \alpha_c : \mathsf{ApartZero}_F(c) \to \mathsf{NonZero}_F(c),$$

where $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot)$*, any witness*

$$p_{abc} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a,b),c))$$

induces witnesses p_{ab}*,* p_a*,* p_b*, and* p_c *such that*

$$\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(a,b),c),p_{abc})\mathsf{hsamemul}(\mathsf{inv}(c,p_c),\mathsf{mul}(\mathsf{inv}(b,p_b),\mathsf{inv}(a,p_a))).$$

Proof. Apply Definition 45.303 to p_{abc} for the binary product $(\mathsf{mul}(a,b),c)$. This gives $q_{ab} : \mathsf{ApartZero}_F(\mathsf{mul}(a,b))$ and $q_c : \mathsf{ApartZero}_F(c)$; set $p_{ab} := \alpha_{ab}(q_{ab})$ and $p_c := \alpha_c(q_c)$.

Apply the same theorem to p_{ab} for the binary product (a,b). The two resulting apartness witnesses $q_a : \mathsf{ApartZero}_F(a)$ and $q_b : \mathsf{ApartZero}_F(b)$ determine $p_a := \alpha_a(q_a)$ and $p_b := \alpha_b(q_b)$.

Now apply Definition 45.213 to $(\mathsf{mul}(a,b),c)$ with witnesses p_{abc}, p_{ab}, and p_c. Apply it again to (a,b) with witnesses p_{ab}, p_a, and p_b. Multiplication congruence transports the second equality into the right factor of the first one, and associativity plus transitivity of hsame gives the displayed three-factor reversed product. □

Per-object certificate: Traditional target: inverse of a finite triple product. BEDC source: $\mathsf{Field}^{\uparrow}$ plus zero-product exclusion. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: product nonzero exclusion, inverse product reversal, associativity, and multiplication congruence. Ledger: p_{abc} plus induced factor witnesses. Core theorem list: Theorem 45.220. Exactness predicate: reversed inverse equality. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.221 (Field inverse four-factor product reversal from apartness-indexed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication associativity, multiplication congruence, left and right identity laws, proof-indexed inverse laws, product nonzero exclusion, and conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b),$$

$$\alpha_c : \mathsf{ApartZero}_F(c) \to \mathsf{NonZero}_F(c), \quad \alpha_d : \mathsf{ApartZero}_F(d) \to \mathsf{NonZero}_F(d),$$

$$\alpha_{ab} : \mathsf{ApartZero}_F(\mathsf{mul}(a,b)) \to \mathsf{NonZero}_F(\mathsf{mul}(a,b)),$$

$$\alpha_{abc} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,b),c)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a,b),c)),$$

$$\alpha_{abcd} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d)),$$

where $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot)$*, any witness*

$$p_{abcd} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d))$$

induces witnesses p_a*,* p_b*,* p_c*, and* p_d *such that*

$$\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),p_{abcd})\mathsf{hsamemul}(\mathsf{inv}(d,p_d),\mathsf{mul}(\mathsf{inv}(c,p_c),\mathsf{mul}(\mathsf{inv}(b,p_b),\mathsf{inv}(a,p_a)))).$$

Proof. Apply Definition 45.303 to p_{abcd} for the binary product $(\mathsf{mul}(\mathsf{mul}(a,b),c),d)$. This gives $q_{abc} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,b),c))$ and $q_d : \mathsf{ApartZero}_F(d)$; set $p_{abc} := \alpha_{abc}(q_{abc})$ and $p_d := \alpha_d(q_d)$.

Now apply Definition 45.213 to the same outer binary product with witnesses p_{abcd}, p_{abc}, and p_d. It identifies the inverse of the fourfold product with $\mathsf{mul}(\mathsf{inv}(d,p_d),\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(a,b),c),p_{abc}))$.

Apply Definition 45.220 to the left triple, using p_{abc} and the conversion maps α_{ab}, α_a, α_b, and α_c. This produces the factor witnesses p_a, p_b, and p_c and the reversed triple inverse.

Multiplication congruence transports the triple equality into the right factor of the outer equality. Associativity and transitivity of hsame then rebracket the product into the displayed d,c,b,a reversed inverse spine. Equivalently, the factor apartness witnesses may be read from Definition 45.318 before the same conversion and inverse-reversal steps are applied. □

Per-object certificate: Traditional target: inverse of a finite fourfold product. BEDC source: $\mathsf{Field}^{\uparrow}$ plus four-factor zero-apartness exactness. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: four-factor apartness extraction, product nonzero exclusion, inverse product reversal, three-factor inverse reversal, associativity, and multiplication congruence. Ledger: p_{abcd} plus induced factor witnesses. Core theorem list: Theorem 45.221. Exactness predicate: reversed inverse equality. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.222 (Field multiplication by an indexed right inverse cancels)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_b : \mathsf{NonZero}_F(b)$ *determines*

$$(ab)\mathsf{inv}(b, p_b)\mathsf{hsame}a.$$

Proof. Reassociate $(ab)\mathsf{inv}(b, p_b)$ to $a(b\mathsf{inv}(b, p_b))$. The apartness-indexed right inverse law collapses the tail to the unit, and the right identity law leaves a. □

Lean4 ✓ `BEDC.Derived.FieldUp.field mul inverse right cancel from apartness`

Theorem 45.223 (Field right-multiplication equation solution from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_b : \mathsf{NonZero}_F(b)$ *makes* $xb\mathsf{hsame}a$ *imply* $x\mathsf{hsame}a\mathsf{inv}(b, p_b)$.

Proof. First cancel the indexed right inverse in $(xb)\mathsf{inv}(b, p_b)$ to identify it with x. Then transport the supplied equality $xb\mathsf{hsame}a$ through right multiplication by $\mathsf{inv}(b, p_b)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field right mul equation solution from apartness`

Theorem 45.224 (Field right-multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_b : \mathsf{NonZero}_F(b)$ *makes the right-multiplication equation and its displayed solution equivalent:*

$$xb\mathsf{hsame}a \iff x\mathsf{hsame}a\mathsf{inv}(b, p_b).$$

Proof. For the forward direction, apply Theorem 45.223. This is the one-way solver from the equation $xb\mathsf{hsame}a$ to the proof-indexed solution $x\mathsf{hsame}a\mathsf{inv}(b, p_b)$.

For the reverse direction, transport the assumed solution through right multiplication by b. The transported right side is $(a\mathsf{inv}(b, p_b))b$.

Reassociate the factors as $a(\mathsf{inv}(b, p_b)b)$. The indexed left inverse law collapses $\mathsf{inv}(b, p_b)b$ to the unit, and the right identity law leaves a. □

Lean4 ✓ `BEDC.Derived.FieldUp.field right mul equation solution from apartness iff`
Lean4 ✓ `BEDC.Derived.FieldUp.field right mul equation exact from apartness`

Theorem 45.225 (Field right-multiplication empty equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_b : \mathsf{NonZero}_F(b)$ *and both* x *and* y *solve the empty endpoint right-multiplication equation*

$$xb\mathsf{hsame}\mathsf{e} \qquad \textit{and} \qquad yb\mathsf{hsame}\mathsf{e},$$

then $x\mathsf{hsame}y$.

Proof. Apply the right-multiplication solution theorem to both displayed equations at the concrete endpoint e. Both solutions are classified with the same normal form $\mathsf{e}\,\mathsf{inv}(b, p_b)$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field right mul empty equation solution unique from apartness`

Theorem 45.226 (Field right-multiplication empty-context equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_b : \mathsf{NonZero}_F(b)$ *and both* x *and* y *solve the same right multiplication equation at the endpoint* $\mathsf{append}(\mathsf{e}, c)$,

$$xb\mathsf{hsame}\mathsf{append}(\mathsf{e}, c) \qquad \textit{and} \qquad yb\mathsf{hsame}\mathsf{append}(\mathsf{e}, c),$$

then $x\mathsf{hsame}y$.

Proof. Apply the right-multiplication solution theorem to both displayed equations at the concrete endpoint $\mathsf{append}(\mathsf{e}, c)$. Both solutions are classified with the same normal form $\mathsf{append}(\mathsf{e}, c)\,\mathsf{inv}(b, p_b)$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field right mul empty context equation solution unique from apartness`

Theorem 45.227 (Field multiplication by an indexed left inverse cancels)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_a : \mathsf{NonZero}_F(a)$ *determines*

$$\mathsf{inv}(a, p_a)(ab)\mathsf{hsame}b.$$

Proof. Reassociate $\mathsf{inv}(a, p_a)(ab)$ to $(\mathsf{inv}(a, p_a)a)b$. The apartness-indexed left inverse law collapses the head to the unit, and the left identity law leaves b. □

Lean4 ✓ `BEDC.Derived.FieldUp.field mul inverse left cancel from apartness`

Theorem 45.228 (Field left-multiplication equation solution from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_a : \mathsf{NonZero}_F(a)$ *makes* $ax\mathsf{hsame}c$ *imply* $x\mathsf{hsame}\mathsf{inv}(a, p_a)c$.

Proof. First use Theorem 45.227 to identify x with $\mathsf{inv}(a, p_a)(ax)$.

Then transport the supplied equality $ax\mathsf{hsame}c$ through left multiplication by $\mathsf{inv}(a, p_a)$, yielding $\mathsf{inv}(a, p_a)(ax)\mathsf{hsame}\mathsf{inv}(a, p_a)c$. □

Theorem 45.229 (Field left-multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, any witness* $p_a : \mathsf{NonZero}_F(a)$ *makes the left-multiplication equation and its displayed solution equivalent:*

$$ax\mathsf{hsame}c \iff x\mathsf{hsame}\mathsf{inv}(a, p_a)c.$$

Proof. For the forward direction, apply Theorem 45.228. This is the one-way solver from the equation $ax\mathsf{hsame}c$ to the proof-indexed solution $x\mathsf{hsame}\mathsf{inv}(a, p_a)c$.

For the reverse direction, transport the assumed solution through left multiplication by a. The transported right side is $a(\mathsf{inv}(a, p_a)c)$.

Reassociate the factors as $(a\mathsf{inv}(a, p_a))c$. The indexed right inverse law collapses $a\mathsf{inv}(a, p_a)$ to the unit, and the left identity law leaves c. □

Lean4 ✓ `BEDC.Derived.FieldUp.field left mul equation exact from apartness`

Theorem 45.230 (Field left-multiplication empty equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_a : \mathsf{NonZero}_F(a)$ *and both* x *and* y *solve the empty endpoint left-multiplication equation*

$$ax\mathsf{hsame}\mathsf{e} \qquad \textit{and} \qquad ay\mathsf{hsame}\mathsf{e},$$

then $x\mathsf{hsame}y$.

Proof. Cancel the common certified left factor in both displayed equations. The left-inverse cancellation theorem classifies both solutions with the same normal form $\mathsf{inv}(a, p_a)\mathsf{e}$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field left mul empty equation solution unique from apartness`

Theorem 45.231 (Field left-multiplication empty-context equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_a : \mathsf{NonZero}_F(a)$ *and both* x *and* y *solve the same left multiplication equation at the endpoint* $\mathsf{append}(\mathsf{e}, c)$*,*

$$ax\mathsf{hsameappend}(\mathsf{e}, c) \qquad \textit{and} \qquad ay\mathsf{hsameappend}(\mathsf{e}, c),$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the left-multiplication exactness theorem to both displayed equations at the concrete endpoint $\mathsf{append}(\mathsf{e}, c)$. Both solutions are classified with the same normal form $\mathsf{inv}(a, p_a)\,\mathsf{append}(\mathsf{e}, c)$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field left mul empty context equation solution unique from apartness`

Theorem 45.232 (Field middle multiplication equation solution from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, explicit witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *make*

$$(ax)b\mathsf{hsame}c$$

imply

$$x\mathsf{hsameinv}(a, p_a)(c\mathsf{inv}(b, p_b)).$$

Proof. First solve the right factor equation using the indexed right inverse of b, obtaining $ax\mathsf{hsame}c\mathsf{inv}(b, p_b)$. Then cancel the left factor with the indexed left inverse of a, transporting the right-factor solution through left multiplication by $\mathsf{inv}(a, p_a)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field middle mul equation solution from apartness`

Theorem 45.233 (Field middle multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, explicit witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *make the middle equation and its displayed solution equivalent:*

$$(ax)b\mathsf{hsame}c \iff x\mathsf{hsameinv}(a, p_a)(c\mathsf{inv}(b, p_b)).$$

Proof. For the forward direction, apply Theorem 45.232. It is exactly the one-way solver from the middle equation to the proof-indexed displayed solution.

For the reverse direction, transport the assumed solution through the congruence of left multiplication by a and right multiplication by b. The resulting term is $(a(\mathsf{inv}(a, p_a)(c\mathsf{inv}(b, p_b))))b$.

Reassociate the factors so that $a\mathsf{inv}(a, p_a)$ is exposed at the left and $\mathsf{inv}(b, p_b)b$ is exposed at the right. The indexed right inverse law collapses $a\mathsf{inv}(a, p_a)$ to the unit, and the indexed left inverse law collapses $\mathsf{inv}(b, p_b)b$ to the unit. The left and right identity laws then leave c. □

Lean4 ✓ `BEDC.Derived.FieldUp.field middle mul equation exact from apartness`
↪ variant `BEDC.Derived.FieldUp.field middle mul equation solution from apartness iff`

Theorem 45.234 (Field conjugation equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_a : \mathsf{NonZero}_F(a)$ *and the unit is apart from* e*, then conjugating by* a *gives the exact equation*

$$(ax)\mathsf{inv}(a, p_a)\mathsf{hsame}b \iff x\mathsf{hsameinv}(a, p_a)(ba).$$

Proof. First derive that $\mathsf{inv}(a, p_a)$ is nonzero from apartness of the unit and the left inverse law. Apply middle multiplication exactness with right factor $\mathsf{inv}(a, p_a)$. The displayed right side contains the double inverse of a; inverse uniqueness identifies this double inverse with a, and multiplication congruence transports the solution term. □

Lean4 ✓ `BEDC.Derived.FieldUp.field conjugation equation exact from apartness`

Theorem 45.235 (Field apart-empty conjugation equation exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate whose proof-indexed inverse is indexed by apartness from* e*, any witness* $p_a : \mathsf{hsame}(a, \mathsf{e}) \to \mathsf{False}$ *makes conjugation by* a *classify the middle factor exactly:*

$$\mathsf{hsame}((ax)\mathsf{inv}(a, p_a), b) \iff \mathsf{hsame}(x, (\mathsf{inv}(a, p_a)b)a).$$

Proof. For the forward direction, multiply the conjugation equation on the right by a. Associativity exposes $\mathsf{inv}(a, p_a)a$, the left inverse law collapses it to the unit, and the right identity law leaves $ax\mathsf{hsame}ba$. Then apply left-multiplication exactness and reassociate the target to $(\mathsf{inv}(a, p_a)b)a$.

For the reverse direction, transport the assumed middle-factor equation by left multiplication with a. Reassociation exposes $a\mathsf{inv}(a, p_a)$, the right inverse law collapses it to the unit, and the left identity law gives $ax\mathsf{hsame}ba$. Multiplying on the right by $\mathsf{inv}(a, p_a)$ and collapsing $a\mathsf{inv}(a, p_a)$ gives the original conjugation equation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field apart empty conjugation equation exact`

Theorem 45.236 (Field apart-empty inverse conjugation equation exactness)**.** *For the same* $\mathsf{Field}^{\uparrow}$ *data and apart-empty witness* $p_a : \mathsf{hsame}(a, \mathsf{e}) \to \mathsf{False}$*, inverse conjugation by* $\mathsf{inv}(a, p_a)$ *also classifies the middle factor exactly:*

$$\mathsf{hsame}((\mathsf{inv}(a, p_a)x)a, b) \iff \mathsf{hsame}(x, (ab)\mathsf{inv}(a, p_a)).$$

Proof. For the forward direction, multiply the inverse-conjugation equation on the right by $\mathsf{inv}(a, p_a)$. Reassociation exposes $a\mathsf{inv}(a, p_a)$, the right inverse law collapses it to the unit, and the right identity law leaves $\mathsf{inv}(a, p_a)x\mathsf{hsame}b\mathsf{inv}(a, p_a)$. Transporting by left multiplication with a collapses $a\mathsf{inv}(a, p_a)$ and reassociates the right side to $(ab)\mathsf{inv}(a, p_a)$.

For the reverse direction, transport the displayed middle equation by left multiplication with $\mathsf{inv}(a, p_a)$. Reassociation exposes $\mathsf{inv}(a, p_a)a$, the left inverse law collapses it to the unit, and left identity gives $\mathsf{inv}(a, p_a)x\mathsf{hsame}b\mathsf{inv}(a, p_a)$. Multiplying on the right by a and collapsing $\mathsf{inv}(a, p_a)a$ gives the inverse-conjugation equation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field apart empty inverse conjugation equation exact`

Theorem 45.237 (Field middle multiplication empty equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *and both* x *and* y *solve the empty endpoint middle multiplication equation*

$$(ax)b\mathsf{hsame}\mathsf{e} \qquad \textit{and} \qquad (ay)b\mathsf{hsame}\mathsf{e},$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the middle multiplication solution theorem to both displayed equations at the concrete endpoint e. Both solutions are classified with $\mathsf{inv}(a, p_a)(\mathsf{e}\,\mathsf{inv}(b, p_b))$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field middle mul empty equation solution unique from apartness`

Theorem 45.238 (Field two-sided empty-context product solution uniqueness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, if* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *and both* x *and* y *solve the same two-sided product equation at the concrete empty-context target*

$$(ax)b\mathsf{hsameappend}(\mathsf{e}, c) \qquad \textit{and} \qquad (ay)b\mathsf{hsameappend}(\mathsf{e}, c),$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the middle multiplication solution theorem to both displayed equations at the concrete endpoint $\mathsf{append}(\mathsf{e}, c)$. Both solutions are classified with the same normal form $\mathsf{inv}(a, p_a)(\mathsf{append}(\mathsf{e}, c)\mathsf{inv}(b, p_b))$. Symmetry on the second classification and transitivity of hsame identify x and y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided empty context product solution unique from apartness`

Theorem 45.239 (Field inverse is apart from empty under right inverse one-apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication* mul*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, multiplication congruence under* hsame*, zero-right multiplication* $\mathsf{mul}(x, 0)\mathsf{hsame}0$*, right inverse law* $\mathsf{mul}(a, \mathsf{inv}(a, p))\mathsf{hsame}1$*, and one-apartness* $\omega : (1\mathsf{hsame}0) \to \bot$*, every witness* $p_a : \mathsf{NonZero}_F(a)$ *constructively refutes*

$$\mathsf{inv}(a, p_a)\mathsf{hsame}0.$$

Proof. Assume $\mathsf{inv}(a, p_a)\mathsf{hsame}0$. Multiplication congruence transports this equality through left multiplication by a, and the zero-right law identifies the transported product with 0. The right inverse law identifies the same product with 1, so transitivity gives $1\mathsf{hsame}0$, contradicting ω. □

Theorem 45.240 (Field inverse is nonzero under right inverse one-apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication* mul*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, multiplication congruence under* hsame*, zero-right multiplication* $\mathsf{mul}(x, 0)\mathsf{hsame}0$*, right inverse law* $\mathsf{mul}(a, \mathsf{inv}(a, p))\mathsf{hsame}1$*, one-apartness* $\omega : (1\mathsf{hsame}0) \to \bot$*, and conversion maps* $\alpha_x : ((x\mathsf{hsame}0) \to \bot) \to \mathsf{NonZero}_F(x)$*, every witness* $p_a : \mathsf{NonZero}_F(a)$ *canonically determines*

$$\mathsf{NonZero}_F(\mathsf{inv}(a, p_a)).$$

Proof. The preceding theorem supplies the constructive refutation of $\mathsf{inv}(a, p_a)\mathsf{hsame}0$ from multiplication congruence, the zero-right law, the indexed right inverse law, and one-apartness. Applying $\alpha_{\mathsf{inv}(a,p_a)}$ to that refutation produces the required nonzero witness for the inverse. □

Theorem 45.241 (Field inverse empty-apartness hsame transport)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication* mul*, unit* 1*, proof-indexed inverse* $\mathsf{inv}(a, p)$*, multiplicative associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws, if* $a\mathsf{hsame}b$*, then apartness of the indexed inverse at* a *from* e *transports to the indexed inverse at* b*:*

$$(\mathsf{inv}(a, p_a)\mathsf{hsamee} \to \bot) \Longrightarrow (\mathsf{inv}(b, p_b)\mathsf{hsamee} \to \bot).$$

Proof. Apply inverse congruence from apartness transport to $a\mathsf{hsame}b$, obtaining $\mathsf{inv}(a, p_a)\mathsf{hsameinv}(b, p_b)$. Any collapse $\mathsf{inv}(b, p_b)\mathsf{hsamee}$ composes with that equality, producing a collapse of $\mathsf{inv}(a, p_a)$ to e, contradicting the supplied apartness predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.field inverse apart empty hsame transport`

Theorem 45.242 (Field multiplication cancellation from apartness-certified factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, right and left multiplication by apartness-certified nonzero factors are cancellative. Any witness* $p_b : \mathsf{NonZero}_F(b)$ *makes*

$$xb\mathsf{hsame}yb \quad \Longrightarrow \quad x\mathsf{hsame}y,$$

and any witness $p_a : \mathsf{NonZero}_F(a)$ *makes*

$$ax\mathsf{hsame}ay \quad \Longrightarrow \quad x\mathsf{hsame}y.$$

Proof. For the right clause, apply Theorem 45.223 to $xb\mathsf{hsame}yb$. This gives $x\mathsf{hsame}(yb)\mathsf{inv}(b, p_b)$. Theorem 45.222 identifies $(yb)\mathsf{inv}(b, p_b)$ with y, so transitivity of hsame gives $x\mathsf{hsame}y$.

For the left clause, apply Theorem 45.228 to $ax\mathsf{hsame}ay$. This gives $x\mathsf{hsameinv}(a, p_a)(ay)$. Theorem 45.227 identifies $\mathsf{inv}(a, p_a)(ay)$ with y, and transitivity of hsame finishes. □

Theorem 45.243 (Field empty-context multiplication cancellation from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication congruence, and apartness-certified nonzero factors, empty left append context on the product endpoint is cancellative. Any witness* $p_b : \mathsf{NonZero}_F(b)$ *makes*

$$xb\mathsf{hsameappend}(\mathsf{e}, yb) \quad \Longrightarrow \quad x\mathsf{hsame}y,$$

and any witness $p_a : \mathsf{NonZero}_F(a)$ *makes*

$$au\mathsf{hsameappend}(\mathsf{e}, av) \quad \Longrightarrow \quad u\mathsf{hsame}v.$$

Proof. The empty left append endpoint reduces by the finite-kernel law $\mathsf{append}(\mathsf{e}, h) = h$. The right-factor clause solves the equation $xb\mathsf{hsameappend}(\mathsf{e}, yb)$ using the indexed inverse of b, then cancels $(yb)\mathsf{inv}(b, p_b)$ back to y. The left-factor clause is symmetric: solve the left multiplication equation using the indexed inverse of a, reduce the empty-context endpoint, and cancel $\mathsf{inv}(a, p_a)(av)$ back to v. □

Lean4 ✓ `BEDC.Derived.FieldUp.field nonzero mul empty context cancel from apartness`

Theorem 45.244 (Field one-sided multiplication cancellation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses and multiplication congruence, right and left multiplication by apartness-certified nonzero factors are classifier-exact. Any witness* $p_b : \mathsf{NonZero}_F(b)$ *makes every* $x, y : \mathsf{BHist}$ *satisfy*

$$xb\mathsf{hsame}yb \Longleftrightarrow x\mathsf{hsame}y,$$

and any witness $p_a : \mathsf{NonZero}_F(a)$ *makes every* $x, y : \mathsf{BHist}$ *satisfy*

$$ax\mathsf{hsame}ay \Longleftrightarrow x\mathsf{hsame}y.$$

Proof. For the right-factor clause, the forward direction is exactly the right cancellation clause of Theorem 45.242 with witness p_b.

For the reverse direction, start from $x\mathsf{hsame}y$. Multiplication congruence transports this equality through right multiplication by the fixed factor b, using reflexivity on that factor, and gives $xb\mathsf{hsame}yb$.

For the left-factor clause, the forward direction is exactly the left cancellation clause of Theorem 45.242 with witness p_a.

For the reverse direction, start from $x\mathsf{hsame}y$. Multiplication congruence transports this equality through left multiplication by the fixed factor a, using reflexivity on that factor, and gives $ax\mathsf{hsame}ay$. □

Theorem 45.245 (Field two-sided multiplication exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication congruence, and witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x, y : \mathsf{BHist}$ *satisfy*

$$(ax)b\mathsf{hsame}(ay)b \iff x\mathsf{hsame}y.$$

Proof. For the forward direction, apply the right-factor clause of Theorem 45.244 with witness p_b to $(ax)b\mathsf{hsame}(ay)b$. This gives $ax\mathsf{hsame}ay$.

Apply the left-factor clause of the same theorem with witness p_a. This transports $ax\mathsf{hsame}ay$ across the left multiplication classifier and gives $x\mathsf{hsame}y$.

For the reverse direction, start from $x\mathsf{hsame}y$. Multiplication congruence transports this classifier equality through left multiplication by a, giving $ax\mathsf{hsame}ay$.

Apply multiplication congruence once more through right multiplication by b. The result is exactly

$$(ax)b\mathsf{hsame}(ay)b.$$

□

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided mul exact from apartness`

Theorem 45.246 (Field two-sided zero product exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, ring zero absorption, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, indexed left and right inverse laws, and witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x : \mathsf{BHist}$ *satisfies*

$$(ax)b\mathsf{hsame}0 \iff x\mathsf{hsame}0.$$

Proof. For the forward direction, apply Theorem 45.223 with the right-factor witness p_b to the hypothesis $(ax)b\mathsf{hsame}0$. This solves the right multiplication equation and gives $ax\mathsf{hsame}0\mathsf{inv}(b, p_b)$.

Use Theorem 43.39 to identify $0\mathsf{inv}(b, p_b)$ with 0. Transitivity of hsame therefore gives $ax\mathsf{hsame}0$.

Now apply the checked left inverse cancellation chain with the left-factor witness p_a to $ax\mathsf{hsame}0$. The left solver gives $x\mathsf{hsameinv}(a, p_a)0$.

Use Theorem 43.39 again to identify $\mathsf{inv}(a, p_a)0$ with 0. Transitivity of hsame yields $x\mathsf{hsame}0$.

For the reverse direction, start from $x\mathsf{hsame}0$. Multiplication congruence transports this equality through left multiplication by a, giving $ax\mathsf{hsame}a0$.

Apply Theorem 43.39 to collapse $a0\mathsf{hsame}0$. Transitivity gives $ax\mathsf{hsame}0$.

Multiplication congruence now transports through right multiplication by b, giving $(ax)b\mathsf{hsame}0b$. A second application of Theorem 43.39 collapses $0b\mathsf{hsame}0$, and transitivity yields $(ax)b\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided empty product exact from apartness`

Theorem 45.247 (Field one-sided zero product cancellation from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, ring zero absorption, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and indexed inverse laws, every* $a, b : \mathsf{BHist}$ *satisfies the following clauses.*

1. *If* $p_a : \mathsf{NonZero}_F(a)$ *and* $ab\mathsf{hsame}0$*, then* $b\mathsf{hsame}0$*.*

2. *If* $p_b : \mathsf{NonZero}_F(b)$ *and* $ab\mathsf{hsame}0$*, then* $a\mathsf{hsame}0$*.*

Proof. For the left-factor clause, apply Theorem 45.228 to the equation $ab\mathsf{hsame}0$ with witness p_a. This gives $b\mathsf{hsameinv}(a, p_a)0$.

Use Theorem 43.39 to identify $\mathsf{inv}(a, p_a)0$ with 0. Transitivity of hsame yields $b\mathsf{hsame}0$.

For the right-factor clause, apply Theorem 45.223 to the same equation with witness p_b. This gives $a\mathsf{hsame}0\mathsf{inv}(b, p_b)$.

Use Theorem 43.39 to identify $0\mathsf{inv}(b, p_b)$ with 0. Transitivity of hsame yields $a\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field one sided zero product cancel from apartness`

Theorem 45.248 (Field nonzero factors exclude zero product)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, ring zero absorption, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, indexed inverse laws, classifier transport of* $\mathsf{NonZero}_F$*, and contradiction of* $\mathsf{NonZero}_F(0)$*, every* $a, b : \mathsf{BHist}$ *with witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *satisfies*

$$\neg(ab\mathsf{hsame}0).$$

Proof. Assume $ab\mathsf{hsame}0$. Apply Theorem 45.247 with the left-factor witness p_a. The left-factor clause gives $b\mathsf{hsame}0$.

Transport the witness $p_b : \mathsf{NonZero}_F(b)$ along this classifier equality. The stability clause for $\mathsf{NonZero}_F$ supplies $\mathsf{NonZero}_F(0)$.

The field stability certificate includes contradiction of $\mathsf{NonZero}_F(0)$, so the transported evidence closes the goal with $\bot$.

The symmetric route uses p_b in the right-factor clause and transports p_a after deriving $a\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field nonzero factors exclude empty product`

Definition 45.249 (Field zero apartness)**.** *For a BEDC history-level field candidate, the zero-apartness predicate is*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot).$$

When this predicate is the nonzero index, inverse formation is indexed directly by apart-zero witnesses.

Theorem 45.250 (Field apart-zero semantic name certificate)**.** *The apart-zero predicate* $\mathsf{ApartZero}_F(h) = (h\mathsf{hsame}0 \to \bot)$*, used as source, pattern, and ledger policy with classifier* hsame*, forms a semantic name certificate.*

Proof. The history $\mathsf{E0}(\mathsf{e})$ is apart from zero by constructor separation, so the source carrier is inhabited. The classifier fields are the reflexivity, symmetry, and transitivity laws of hsame. Carrier preservation is contrapositive transport: if $h\mathsf{hsame}k$ and h is apart from zero, then any $k\mathsf{hsame}0$ composes to $h\mathsf{hsame}0$, contradiction. Pattern and ledger soundness are immediate because all three policies are the same apart-zero predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero semanticNameCert`

Theorem 45.251 (Field apart-zero empty transport along history sameness)**.** *If* $a\mathsf{hsame}b$ *and* a *is apart from zero, then* b *is also apart from zero:*

$$a\mathsf{hsame}b \wedge \mathsf{ApartZero}_F(a) \Longrightarrow (b\mathsf{hsame}0 \to \mathsf{False}).$$

Proof. Assume $b\mathsf{hsame}0$. Transitivity of hsame with $a\mathsf{hsame}b$ gives $a\mathsf{hsame}0$, contradicting the apart-zero witness for a. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero empty hsame transport`

Theorem 45.252 (Field apart-zero append factor closure)**.** *For histories $p, q : \mathsf{BHist}$, if either p or q is apart from zero, then the appended history $\mathsf{append}(p, q)$ is apart from zero:*

$$\mathsf{ApartZero}_F(p) \vee \mathsf{ApartZero}_F(q) \Longrightarrow \mathsf{ApartZero}_F(\mathsf{append}(p, q)).$$

Proof. Assume $\mathsf{append}(p, q)\mathsf{hsame}0$. Empty-append inversion gives both $p\mathsf{hsame}0$ and $q\mathsf{hsame}0$. If the apart-zero witness is on the left factor, it rejects the first equality; if it is on the right factor, it rejects the second equality. Thus either factor witness prevents the appended history from classifying as zero. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append factor closed`

Theorem 45.253 (Field apart-zero left factor from non-apart tail)**.** *For histories $p, q : \mathsf{BHist}$, if $\mathsf{append}(p, q)$ is apart from zero and q is not apart from zero, then p is apart from zero:*

$$\mathsf{ApartZero}_F(\mathsf{append}(p, q)) \Longrightarrow (\mathsf{ApartZero}_F(q) \Rightarrow \mathsf{False}) \Longrightarrow \mathsf{ApartZero}_F(p).$$

Proof. Assume $p\mathsf{hsame}0$. The concrete history q is either empty or visibly headed. If it is visibly headed, it is apart from zero, contradicting the non-apart tail hypothesis. Hence $q\mathsf{hsame}0$. Empty-append introduction then gives $\mathsf{append}(p, q)\mathsf{hsame}0$, contradicting the apart-zero witness for the append. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append left factor of tail not apart`

Theorem 45.254 (Field apart-zero append split exactness)**.** *For histories $p, q : \mathsf{BHist}$, apart-zero of an appended history is exactly apart-zero of one of its factors:*

$$\mathsf{ApartZero}_F(\mathsf{append}(p, q)) \Longleftrightarrow \mathsf{ApartZero}_F(p) \vee \mathsf{ApartZero}_F(q).$$

Proof. For the forward direction, split on the concrete shape of p. If $p = \mathsf{e}$, then $q\mathsf{hsamee}$ would make the append empty, contradicting the appended apart-zero witness. If p has a visible head constructor, that constructor is separated from e, so p itself is apart from zero.

For the reverse direction, use the one-factor closure theorem: an empty append would split into empty witnesses for both factors, contradicting the selected apart-zero factor. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append split iff`

Theorem 45.255 (Field apart-zero right factor from non-apart head)**.** *For histories $p, q : \mathsf{BHist}$, if $\mathsf{append}(p, q)$ is apart from zero and p is not apart from zero, then q is apart from zero:*

$$\mathsf{ApartZero}_F(\mathsf{append}(p, q)) \Longrightarrow (\mathsf{ApartZero}_F(p) \Rightarrow \mathsf{False}) \Longrightarrow \mathsf{ApartZero}_F(q).$$

Proof. Apply the append split exactness theorem to the appended apart-zero witness. The left branch contradicts the non-apart head hypothesis, so the right branch remains and supplies $\mathsf{ApartZero}_F(q)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append right factor of head not apart`

Theorem 45.256 (Field apart-zero continuation endpoint split)**.** *For histories p, q, r : BHist with $\mathsf{Cont}(p, q, r)$, apartness from zero of the continuation result is exactly apartness from zero of one endpoint:*

$$\mathsf{ApartZero}_F(r) \iff \mathsf{ApartZero}_F(p) \vee \mathsf{ApartZero}_F(q).$$

Proof. The continuation ledger identifies r with $\mathsf{append}(p, q)$. The forward implication performs induction by cases on the left endpoint: if it is e, any empty endpoint for q would empty the append; if it is visibly headed, constructor separation supplies apartness of p. Conversely, an empty append result splits into empty endpoints for both factors, and the selected apart-zero witness rejects the corresponding endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero continuation endpoint split iff`

Theorem 45.257 (Field apart-zero mutual continuation cycle tails absurd)**.** *For a mutual continuation cycle $\mathsf{Cont}(h, leftTail, k)$ and $\mathsf{Cont}(k, rightTail, h)$, neither cycle tail can be field-apart from zero.*

Proof. Continuation cycle cancellation reads both tails as e. Each $\mathsf{ApartZero}_F$ witness is exactly a rejection of that empty endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero mutual continuation cycle tails absurd`

Theorem 45.258 (Field apart-zero triangle continuation cycle tails absurd)**.** *For a triangle continuation cycle, the three displayed tails cannot be field-apart from zero.*

Proof. Triangle cycle cancellation identifies all three tails with e. The corresponding apart-zero witness rejects each empty-tail conclusion. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero triangle continuation cycle tails absurd`

Theorem 45.259 (Field apart-zero two-step continuation split)**.** *For histories l, h, r, u, v : BHist with $\mathsf{Cont}(l, h, u)$ and $\mathsf{Cont}(u, r, v)$, apartness from zero of the final continuation result is exactly apartness from zero in one of the three displayed factors:*

$$\mathsf{ApartZero}_F(v) \iff (\mathsf{ApartZero}_F(l) \vee \mathsf{ApartZero}_F(h)) \vee \mathsf{ApartZero}_F(r).$$

Proof. For the forward direction, apply Definition 45.256 to the continuation $\mathsf{Cont}(u, r, v)$. The resulting left branch gives $\mathsf{ApartZero}_F(u)$; applying the same theorem to $\mathsf{Cont}(l, h, u)$ splits that branch into $\mathsf{ApartZero}_F(l)$ or $\mathsf{ApartZero}_F(h)$. The right branch is already $\mathsf{ApartZero}_F(r)$, so the displayed disjunction follows.

For the reverse direction, split the displayed disjunction. If the left summand holds, apply Definition 45.256 to $\mathsf{Cont}(l, h, u)$ to obtain $\mathsf{ApartZero}_F(u)$, then apply the same theorem to $\mathsf{Cont}(u, r, v)$ to obtain $\mathsf{ApartZero}_F(v)$. If the right summand holds, the second continuation split directly transports $\mathsf{ApartZero}_F(r)$ to $\mathsf{ApartZero}_F(v)$. □

Theorem 45.260 (Field apart-zero factors excluded by singleton continuation result)**.** *For histories $P, Q, R, endpoint$: BHist, if $\mathsf{Cont}(P, Q, R)$ and the result is classified by the field singleton classifier, then neither factor can be apart from zero:*

$$\mathsf{Cont}(P, Q, R) \wedge \mathsf{FieldSingletonClassifier}(R, endpoint) \Longrightarrow ((\mathsf{ApartZero}_F(P) \to \bot) \wedge (\mathsf{ApartZero}_F(Q) \to \bot)).$$

Proof. The singleton classifier supplies singleton-carrier membership for the result. The continuation endpoint split for singleton carriers pulls that membership back to both factors. An apart-zero witness for either factor is a rejection of that factor being empty, so each factor-side witness is impossible. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero continuation singleton result factor absurd`

Theorem 45.261 (Field apart-zero append congruence under history sameness)**.** *For histories* $a, a', b, b' : \mathsf{BHist}$*, history sameness on both append factors preserves and reflects apartness from zero:*

$$a\mathsf{hsame}a' \wedge b\mathsf{hsame}b' \Longrightarrow (\mathsf{ApartZero}_F(\mathsf{append}(a, b)) \Longleftrightarrow \mathsf{ApartZero}_F(\mathsf{append}(a', b'))).$$

Proof. Substituting the two history-sameness witnesses gives $\mathsf{append}(a, b)\mathsf{hsameappend}(a', b')$. The apart-zero transport law applies in one direction to preserve apartness, and with symmetry of hsame in the other direction to reflect it. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append hsame congr iff`

Theorem 45.262 (Field apart-zero append right-empty exactness)**.** *For histories* $p, q : \mathsf{BHist}$ *with* $q\mathsf{hsamee}$*, appended apart-zero is exactly apart-zero of the left factor:*

$$\mathsf{ApartZero}_F(\mathsf{append}(p, q)) \Longleftrightarrow \mathsf{ApartZero}_F(p).$$

Proof. For the forward implication, assume $p\mathsf{hsamee}$. Together with $q\mathsf{hsamee}$, empty-append exactness gives $\mathsf{append}(p, q)\mathsf{hsamee}$, contradicting appended apart-zero. For the reverse implication, any equality $\mathsf{append}(p, q)\mathsf{hsamee}$ splits by empty-append exactness, and the left component contradicts apart-zero of p. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append right empty iff`

Theorem 45.263 (Field apart-zero append left-empty exactness)**.** *For histories* $p, q : \mathsf{BHist}$ *with* $p\mathsf{hsamee}$*, appended apart-zero is exactly apart-zero of the right factor:*

$$\mathsf{ApartZero}_F(\mathsf{append}(p, q)) \Longleftrightarrow \mathsf{ApartZero}_F(q).$$

Proof. For the forward implication, assume $q\mathsf{hsamee}$. Together with $p\mathsf{hsamee}$, empty-append exactness gives $\mathsf{append}(p, q)\mathsf{hsamee}$, contradicting appended apart-zero. For the reverse implication, any equality $\mathsf{append}(p, q)\mathsf{hsamee}$ splits by empty-append exactness, and the right component contradicts apart-zero of q. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append left empty iff`

Theorem 45.264 (Field apart-zero append visible-headed)**.** *For histories* $p, q : \mathsf{BHist}$*, appending any suffix to a visible-headed history is apart from zero:*

$$\mathsf{ApartZero}_F(\mathsf{append}(\mathsf{E0}(p), q)) \wedge \mathsf{ApartZero}_F(\mathsf{append}(\mathsf{E1}(p), q)).$$

Proof. An empty equality for either appended history splits by empty-append exactness. The left component would identify $\mathsf{E0}(p)$, respectively $\mathsf{E1}(p)$, with e. BEDC constructor separation rules out both identifications. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append visible headed`

Theorem 45.265 (Field apart-zero nested continuation visible-left result)**.** *For histories* $tail, h, r, u, v$: BHist*, a nested continuation whose left source is visibly headed has an apart-zero result:*

$$(\mathsf{Cont}(\mathsf{E0}(tail), h, u) \to \mathsf{Cont}(u, r, v) \to \mathsf{ApartZero}_F(v)) \wedge (\mathsf{Cont}(\mathsf{E1}(tail), h, u) \to \mathsf{Cont}(u, r, v) \to \mathsf{ApartZero}_F(v)).$$

Proof. In the $\mathsf{E0}$ branch, the two continuation witnesses identify u with $\mathsf{append}(\mathsf{E0}(tail), h)$ and v with $\mathsf{append}(u, r)$. If $v\mathsf{hsamee}$, empty-append inversion first forces $u\mathsf{hsamee}$, and a second inversion forces $\mathsf{E0}(tail)\mathsf{hsamee}$, contradicting constructor separation.

The $\mathsf{E1}$ branch is identical, with the second inversion producing $\mathsf{E1}(tail)\mathsf{hsamee}$, again contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero nested continuation visible left result`

Theorem 45.266 (Field apart-zero nested continuation visible-right result)**.** *For histories* $l, h, tail, u, v$: BHist*, a nested continuation whose second continuation tail is visibly headed has an apart-zero result:*

$$(\mathsf{Cont}(l, h, u) \to \mathsf{Cont}(u, \mathsf{E0}(tail), v) \to \mathsf{ApartZero}_F(v)) \wedge (\mathsf{Cont}(l, h, u) \to \mathsf{Cont}(u, \mathsf{E1}(tail), v) \to \mathsf{ApartZero}_F(v)).$$

Proof. In the $\mathsf{E0}$ branch, the first continuation supplies the intermediate history u, and the second continuation identifies v with an append whose right factor is $\mathsf{E0}(tail)$. If $v\mathsf{hsamee}$, empty-append inversion forces that visible right factor to be empty, contradicting BEDC constructor separation.

The $\mathsf{E1}$ branch has the same shape. A hypothetical equality $v\mathsf{hsamee}$ splits through the second continuation and empty-append inversion, forcing $\mathsf{E1}(tail)\mathsf{hsamee}$, which constructor separation rules out. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero nested continuation visible right result`

Per-object certificate: traditional target visible-suffix apart-zero coverage for nested $\mathsf{Field}^{\uparrow}$ continuations; BEDC source BHist plus Cont; carrier $\mathsf{ApartZero}_F$; classifier hsame; ledger records the hidden visible suffix and continuation result; core theorem is Definition 45.266 together with append empty-result inversion; exactness predicate is the displayed apart-zero conclusion; status $\mathsf{ProofTarget}$.

Theorem 45.267 (Field apart-zero empty-context exactness)**.** *For histories* l, h, r : BHist *with* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$*, bilateral empty context is exact for field apart-zero witnesses:*

$$\mathsf{ApartZero}_F(\mathsf{append}(l, \mathsf{append}(h, r))) \iff \mathsf{ApartZero}_F(h).$$

Proof. For the forward direction, assume the contextual apart-zero witness. To prove $\mathsf{ApartZero}_F(h)$, take a candidate equality $h\mathsf{hsamee}$. Empty-append construction combines this equality with $r\mathsf{hsamee}$, giving $\mathsf{append}(h, r)\mathsf{hsamee}$.

Combining that inner equality with $l\mathsf{hsamee}$ classifies the whole nested append as empty. This contradicts the contextual apart-zero witness.

For the reverse direction, assume $\mathsf{ApartZero}_F(h)$ and take a candidate equality

$$\mathsf{append}(l, \mathsf{append}(h, r))\mathsf{hsamee}.$$

Empty-append inversion first splits off the left context and gives $\mathsf{append}(h, r)\mathsf{hsamee}$. Applying the same inversion to the inner append gives $h\mathsf{hsamee}$, contradicting $\mathsf{ApartZero}_F(h)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero empty context iff`

Theorem 45.268 (Field apart-zero left append-context semantic certificate)**.** *For a history* p : BHist *apart from zero, the concrete left-append context predicate*

$$q \longmapsto \mathsf{ApartZero}_F(\mathsf{append}(q, p))$$

used as source, pattern, and ledger policy with classifier hsame *forms a semantic name certificate.*

Proof. The carrier is inhabited at $q = \mathsf{e}$: $\mathsf{append}(\mathsf{e}, p)\mathsf{hsame}p$, so apart-zero of p transports to apart-zero of the displayed append. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Carrier preservation follows by eliminating the history-sameness classifier, which leaves the same concrete append predicate. Pattern and ledger soundness are identities because all three predicates coincide. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append left context semanticNameCert`

Theorem 45.269 (Field apart-zero right append-context semantic certificate)**.** *For a history* p : BHist *apart from zero, the concrete right-append context predicate*

$$q \longmapsto \mathsf{ApartZero}_F(\mathsf{append}(p, q))$$

used as source, pattern, and ledger policy with classifier hsame *forms a semantic name certificate.*

Proof. The carrier is inhabited at $q = \mathsf{e}$: $\mathsf{append}(p, \mathsf{e})\mathsf{hsame}p$, so apart-zero of p transports to apart-zero of the displayed append. The classifier laws are again the equivalence laws of hsame. Carrier preservation eliminates the history-sameness classifier and leaves the same right-append predicate, while pattern and ledger soundness are identity maps. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append right context semanticNameCert`

Theorem 45.270 (Field apart-zero two-sided context semantic certificate)**.** *For histories* L, R : BHist*, if* L *is apart from zero, then the two-sided context predicate*

$$h \longmapsto \mathsf{FieldApartZero}(\mathsf{append}(L, \mathsf{append}(h, R)))$$

used as source, pattern, and ledger policy with classifier hsame *forms a semantic name certificate.*

Proof. The source is inhabited at the empty middle history. If $\mathsf{append}(L, \mathsf{append}(\mathsf{e}, R))$ were empty, append-empty inversion would force $L\mathsf{hsamee}$, contradicting the assumed apart-zero witness for L. The equivalence laws are the standard hsame laws, and carrier preservation follows by eliminating the history-sameness classifier. Pattern and ledger soundness are identities because source, pattern, and ledger policy are the same concrete predicate. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero append context semanticNameCert`

Per-object certificate: traditional target is external-unit context invariance of constructive field apartness. BEDC source is FieldApartZero plus BEDC append empty exactness. Carrier is BHist. Classifier is hsame. Ledger is the two empty-context witnesses $l$hsamee and $r$hsamee. Core theorem list is Definition 45.267. Exactness predicate is the displayed iff. Status is ProofTarget.

Theorem 45.271 (Field product apartness gives inverse product witness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Theorem 45.247 and Theorem 45.213, assume inverse formation is indexed by apart-zero witnesses,*

$$\mathsf{inv} : (a : \mathsf{BHist}) \to \mathsf{ApartZero}_F(a) \to \mathsf{BHist}.$$

If $p_a : \mathsf{ApartZero}_F(a)$ *and* $p_b : \mathsf{ApartZero}_F(b)$*, then there exists* $p_{ab} : \mathsf{ApartZero}_F(\mathsf{mul}(a, b))$ *such that*

$$\mathsf{inv}(\mathsf{mul}(a, b), p_{ab})\mathsf{hsamemul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)).$$

Proof. Build p_{ab} as a function from an assumed zero product to contradiction. Given $z : \mathsf{mul}(a, b)\mathsf{hsame}0$, apply Theorem 45.247 with the left-factor witness p_a. The left-factor clause sends z to $b\mathsf{hsame}0$, and p_b rejects that equality.

Apply Theorem 45.213 with the apart-zero indices p_{ab}, p_a, and p_b to obtain the displayed inverse product equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product apartness inverse product reverse`

Theorem 45.272 (Field equal-squares signed factor exclusion)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, ring zero absorption, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, indexed inverse laws, the* $\mathsf{CommRing}^{\uparrow}$ *hypotheses required by Theorem 44.47, and every* $a, b : \mathsf{BHist}$ *with* $a \cdot a\mathsf{hsame}b \cdot b$*, the following constructive clauses hold.*

1. *If* $p_+ : \mathsf{NonZero}_F(a + b)$*, then* $a + (-b)\mathsf{hsame}0$*.*
2. *If* $p_- : \mathsf{NonZero}_F(a + (-b))$*, then* $a + b\mathsf{hsame}0$*.*

Proof. Apply the reverse direction of Theorem 44.47 to the square equality. The signed factor product is therefore classified as zero:

$$(a + b) \cdot (a + (-b))\mathsf{hsame}0.$$

For the first clause, use the witness $p_+ : \mathsf{NonZero}_F(a + b)$ and apply Theorem 45.247 to the signed factor product with left factor $a + b$ and right factor $a + (-b)$. The left-factor clause gives $a + (-b)\mathsf{hsame}0$.

For the second clause, use the witness $p_- : \mathsf{NonZero}_F(a + (-b))$ and apply Theorem 45.247 to the same signed factor product. The right-factor clause gives $a + b\mathsf{hsame}0$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field equal squares signed factor exclusion`

Per-object certificate: this is the Core theorem list item for constructive field square-factor one-sided cancellation. Source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$. Carrier: BHist. Classifier: hsame. Ledger: the chosen nonzero signed factor witness p_+ or p_-. The incompatible-pair result is recorded at Definition 45.297. Status: CheckedCert.

Zero-product exactness certificate. Traditional target: constructive field zero-product exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: zero absorption, multiplication associativity, multiplicative unit laws, multiplication congruence, and indexed inverse laws. Ledger: the two certified factors $p_a : \mathsf{NonZero}_F(a)$ and $p_b : \mathsf{NonZero}_F(b)$ together with their inverse witnesses. Core theorem list: two-sided zero-product exactness and zero-product exclusion from two nonzero witnesses. Exactness predicate: exact zero classifier and nonzero-product contradiction inside certified two-sided multiplication. Standard bridge: deferred conservative bridge. Status: ProofTarget.

45.6 Field affine equation laws

Theorem 45.273 (Field affine left multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive abelian-group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and left and right inverse laws indexed by*

$p_a : \mathsf{NonZero}_F(a)$, *every* $x, b, c : \mathsf{BHist}$ *satisfy*

$$\mathsf{add}(\mathsf{mul}(a, x), b)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), \mathsf{add}(c, \mathsf{neg}(b))).$$

Proof. For the forward direction, transport $\mathsf{add}(\mathsf{mul}(a, x), b)\mathsf{hsame}c$ through right addition by $\mathsf{neg}(b)$. Addition congruence gives an equality between $\mathsf{add}(\mathsf{add}(\mathsf{mul}(a, x), b), \mathsf{neg}(b))$ and $\mathsf{add}(c, \mathsf{neg}(b))$.

Associativity, the additive inverse law for b, and the additive identity law collapse the left side to $\mathsf{mul}(a, x)$. Apply Theorem 45.228 to obtain the displayed solution for x with right side $\mathsf{add}(c, \mathsf{neg}(b))$.

For the reverse direction, transport the assumed displayed solution through left multiplication by a. Multiplication congruence, associativity, the indexed inverse law for p_a, and the multiplicative identity law collapse the right side to $\mathsf{add}(c, \mathsf{neg}(b))$, so $\mathsf{mul}(a, x)\mathsf{hsameadd}(c, \mathsf{neg}(b))$.

Add b on the right and use addition congruence. Associativity, additive commutativity for the middle pair when needed, the additive inverse law for b, and the additive identity law collapse $\mathsf{add}(\mathsf{add}(c, \mathsf{neg}(b)), b)$ back to c. This gives $\mathsf{add}(\mathsf{mul}(a, x), b)\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine left equation exact from apartness`

Theorem 45.274 (Field affine right multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive abelian-group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and left and right inverse laws indexed by* $p_b : \mathsf{NonZero}_F(b)$*, every* $x, d, c : \mathsf{BHist}$ *satisfy*

$$\mathsf{add}(\mathsf{mul}(x, b), d)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b)).$$

Proof. For the forward direction, transport $\mathsf{add}(\mathsf{mul}(x, b), d)\mathsf{hsame}c$ through right addition by $\mathsf{neg}(d)$. Addition congruence gives equality between the offset equation after adding $\mathsf{neg}(d)$ on the right and $\mathsf{add}(c, \mathsf{neg}(d))$.

Associativity, the additive inverse law for d, and the additive identity law collapse the left side to $\mathsf{mul}(x, b)$. Apply Theorem 45.223 to obtain the displayed solution for x with right side $\mathsf{add}(c, \mathsf{neg}(d))$.

For the reverse direction, transport the assumed displayed solution through right multiplication by b. Multiplication congruence, associativity, the indexed left inverse law for p_b, and the multiplicative identity law collapse the right side to $\mathsf{add}(c, \mathsf{neg}(d))$, so $\mathsf{mul}(x, b)\mathsf{hsameadd}(c, \mathsf{neg}(d))$.

Add d on the right and use addition congruence. Associativity, additive commutativity for the middle pair when needed, the additive inverse law for d, and the additive identity law collapse $\mathsf{add}(\mathsf{add}(c, \mathsf{neg}(d)), d)$ back to c. This gives $\mathsf{add}(\mathsf{mul}(x, b), d)\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine right equation exact from apartness`

Theorem 45.275 (Field one-sided affine map classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and indexed inverse laws for witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every displayed* $x, y, u, v : \mathsf{BHist}$ *satisfies both one-sided affine classifier clauses:*

$$\mathsf{add}(\mathsf{mul}(a, x), u)\mathsf{hsameadd}(\mathsf{mul}(a, y), u) \iff x\mathsf{hsame}y$$

and

$$\mathsf{add}(\mathsf{mul}(x, b), v)\mathsf{hsameadd}(\mathsf{mul}(y, b), v) \iff x\mathsf{hsame}y.$$

Proof. For the left-affine clause, the forward direction first cancels the common additive offset u. The additive group laws and addition congruence instantiate Theorem 41.116, giving $\mathsf{mul}(a, x)\mathsf{hsamemul}(a, y)$. Apply the left-factor cancellation clause of Theorem 45.242 with the witness p_a to obtain $x\mathsf{hsame}y$.

For the reverse direction of the left-affine clause, assume $x\mathsf{hsame}y$. Multiplication congruence transports this equality through left multiplication by a, and addition congruence with the shared offset u gives the displayed left-affine equality.

For the right-affine clause, the forward direction again cancels the common additive offset, now v, by Theorem 41.116. The cancelled equation is $\mathsf{mul}(x, b)\mathsf{hsamemul}(y, b)$, and the right-factor cancellation clause of Theorem 45.242 with witness p_b yields $x\mathsf{hsame}y$.

For the reverse direction of the right-affine clause, assume $x\mathsf{hsame}y$. Multiplication congruence transports the equality through right multiplication by b, and addition congruence with the shared offset v gives the displayed right-affine equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine one sided map classifier exact from apartness`

Theorem 45.276 (Field affine left equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and indexed inverse laws, every* $a, x, y, d, c : \mathsf{BHist}$ *and witness* $p_a : \mathsf{NonZero}_F(a)$ *satisfy*

$$\mathsf{add}(\mathsf{mul}(a, x), d)\mathsf{hsame}c \implies \mathsf{add}(\mathsf{mul}(a, y), d)\mathsf{hsame}c \implies x\mathsf{hsame}y.$$

Proof. Assume the two displayed affine equations. Compose the first equation with the symmetry of the second, obtaining equality of the two affine outputs with the same left factor and offset. The left clause of Definition 45.275 removes that shared affine map and yields $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine left equation solution unique from apartness`

Affine one-sided multiplication certificate. Traditional target: one-sided affine injectivity. BEDC source: $\mathsf{Field}^{\uparrow}$ plus additive group laws. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: additive cancellation, multiplication associativity, multiplicative unit laws, multiplication congruence, and indexed inverse laws. Ledger: the single nonzero witness for the active factor and the additive offset. Core theorem list entry: one-sided affine map classifier exactness from apartness and affine left-equation uniqueness from apartness. Exactness predicate: reflection and preservation of the affine classifier. Standard bridge: deferred conservative bridge. Status: left and right equation exactness clauses CheckedCert; one-sided classifier clause CheckedCert; left-equation uniqueness clause CheckedCert.

Per-object certificate: this is a traditional target for affine left and right field equation solving. BEDC source: $\mathsf{Field}^{\uparrow}$ plus additive group laws. Carrier: BHist. Classifier: hsame. Ledger: the nonzero witnesses $p_a : \mathsf{NonZero}_F(a)$ and $p_b : \mathsf{NonZero}_F(b)$ with their additive offsets. Stability obligations: additive cancellation, multiplication associativity, multiplicative unit laws, multiplication congruence, and indexed inverse laws. Status: CheckedCert.

Theorem 45.277 (Field affine two-sided multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive abelian-group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right*

multiplicative identity laws, multiplication congruence, and indexed inverse laws for witnesses $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x, d, c : \mathsf{BHist}$ *satisfies*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Proof. For the forward direction, transport $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c$ through right addition by $\mathsf{neg}(d)$. Addition congruence gives equality between the offset equation after adding $\mathsf{neg}(d)$ on the right and $\mathsf{add}(c, \mathsf{neg}(d))$.

Associativity, the additive inverse law for d, and the additive identity law collapse the left side to $\mathsf{mul}(\mathsf{mul}(a, x), b)$. Apply Theorem 45.233 to obtain the displayed solution with middle right side $\mathsf{add}(c, \mathsf{neg}(d))$.

For the reverse direction, apply the reverse implication of Theorem 45.233 to the assumed displayed solution. This gives $\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsameadd}(c, \mathsf{neg}(d))$.

Add d on the right and use addition congruence. Associativity, additive commutativity for the middle pair when needed, the additive inverse law for d, and the additive identity law collapse $\mathsf{add}(\mathsf{add}(c, \mathsf{neg}(d)), d)$ back to c. This yields $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine two sided equation exact from apartness`

Theorem 45.278 (Field two-sided affine map classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and indexed inverse laws for witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x, y, d : \mathsf{BHist}$ *satisfies*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(a, y), b), d) \iff x\mathsf{hsame}y.$$

Proof. For the forward direction, first cancel the common additive right summand d. The additive group laws and addition congruence instantiate `BEDC.Derived.GroupUp.group_right_cancel` for add, giving

$$\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsamemul}(\mathsf{mul}(a, y), b).$$

Apply the forward direction of `BEDC.Derived.FieldUp.field_two_sided_mul_exact_from_apartness` with the nonzero witnesses p_a and p_b. The multiplicative exactness theorem removes the outer left factor a and right factor b from the cancelled equation, yielding $x\mathsf{hsame}y$.

For the reverse direction, assume $x\mathsf{hsame}y$. Multiplication congruence first transports this equality through left multiplication by a, giving $\mathsf{mul}(a, x)\mathsf{hsamemul}(a, y)$.

Apply multiplication congruence again on the right by b to obtain $\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsamemul}(\mathsf{mul}(a, y), b)$. Addition congruence with the shared offset d then yields the displayed affine equality. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine two sided map classifier exact from apartness`
↪ variant `BEDC.Derived.FieldUp.field affine two sided empty zero map classifier exact from apartness`

Theorem 45.279 (Field affine identity endpoint)**.** *For a BEDC history-level field candidate with additive right identity at* e *and multiplicative left and right identity at* one*, every* $x : \mathsf{BHist}$ *satisfies*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{one}, x), \mathsf{one}), \mathsf{e})\mathsf{hsame}x.$$

Proof. First use additive right identity to remove the displayed e offset. Then use multiplicative right identity to remove the final factor one, and multiplicative left identity to remove the leading factor. The three hsame steps compose by transitivity. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine identity endpoint`

Theorem 45.280 (Field affine identity classifier)**.** *Under the same identity laws, the identity affine endpoint preserves the history classifier:*

$$x\mathsf{hsame}y \Longrightarrow \mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{one}, x), \mathsf{one}), \mathsf{e})\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(\mathsf{one}, y), \mathsf{one}), \mathsf{e}).$$

Proof. Apply the endpoint identity theorem to the left endpoint and to the right endpoint. Transitivity bridges the left affine endpoint to x, then uses the assumed classifier $x\mathsf{hsame}y$, and finally uses symmetry of the right endpoint identity to reach the displayed affine endpoint over y. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine identity classifier`

Theorem 45.281 (Field affine identity classifier iff)**.** *Under the same identity laws, the identity affine endpoint reflects and preserves the history classifier:*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{one}, x), \mathsf{one}), \mathsf{e})\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(\mathsf{one}, y), \mathsf{one}), \mathsf{e}) \Longleftrightarrow x\mathsf{hsame}y.$$

Proof. The forward direction composes the symmetry of the left endpoint identity, the displayed affine classifier, and the right endpoint identity. The reverse direction is exactly the identity classifier preservation theorem. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine identity classifier iff`

Affine two-sided multiplication certificate. Traditional target: affine field equation solving. BEDC source: $\mathsf{Field}^{\uparrow}$ plus additive group laws. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: additive cancellation, multiplication associativity, multiplicative unit laws, multiplication congruence, and indexed inverse laws. Ledger: both nonzero witnesses and the additive offset. Core theorem list entry: two-sided affine equation exactness and two-sided affine map classifier exactness. Exactness predicate: equivalence between the equation and displayed solution, together with preservation and reflection of the affine classifier. Standard bridge: deferred conservative bridge. Status: ProofTarget.

45.7 Product factor equation exactness

Theorem 45.282 (Field product right-multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x, c : \mathsf{BHist}$ *satisfy*

$$\mathsf{mul}(x, \mathsf{mul}(a, b))\mathsf{hsame}c \Longleftrightarrow x\mathsf{hsamemul}(c, \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))).$$

Traditional target product-factor right equation solving; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations right equation exactness, inverse product reversal, multiplication congruence, associativity and unit laws; ledger the three nonzero witnesses and the reversed inverse product; core theorem this block; exactness predicate the displayed equivalence.*

Proof. First apply Definition 45.224 to the active right factor $\mathsf{mul}(a, b)$ with witness p_{ab}. This gives the equivalence between the displayed equation and $x\mathsf{hsamemul}(c, \mathsf{inv}(\mathsf{mul}(a, b), p_{ab}))$.

Next apply Definition 45.213 with the three witnesses p_{ab}, p_a, and p_b. It identifies the certified inverse of the product with the reversed product $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$.

For the forward direction, use multiplication congruence on the right factor of $\mathsf{mul}(c, -)$ to transport the one-factor solution along the inverse product reversal. The transported right side is exactly the displayed reversed product solution.

For the reverse direction, use symmetry of the inverse product reversal and the same multiplication congruence to transport the displayed reversed product solution back to $x\mathsf{hsamemul}(c, \mathsf{inv}(\mathsf{mul}(a, b), p_{ab}))$. The reverse half of the one-factor exactness theorem then recovers $\mathsf{mul}(x, \mathsf{mul}(a, b))\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product right mul equation exact from apartness`

Theorem 45.283 (Field product right-multiplication empty equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x : \mathsf{BHist}$ *satisfy*

$$\mathsf{mul}(x, \mathsf{mul}(a, b))\mathsf{hsame}\mathsf{e} \iff x\mathsf{hsamemul}(\mathsf{e}, \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))).$$

Lean4 ✓ `BEDC.Derived.FieldUp.field product right mul empty equation exact from apartness`

Proof. Apply right-multiplication equation exactness to the active factor $\mathsf{mul}(a, b)$ at target e. This identifies the displayed equation with the solution using $\mathsf{inv}(\mathsf{mul}(a, b), p_{ab})$. Product inverse reversal identifies that inverse with $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$, and multiplication congruence transports the solution across the right factor of $\mathsf{mul}(\mathsf{e}, -)$ in both directions. □

Theorem 45.284 (Field product left-multiplication equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x, c : \mathsf{BHist}$ *satisfy*

$$\mathsf{mul}(\mathsf{mul}(a, b), x)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)), c).$$

Traditional target product-factor left equation solving; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations left equation exactness, inverse product reversal, multiplication congruence, associativity and unit laws; ledger the three nonzero witnesses and the reversed inverse product; core theorem this block; exactness predicate the displayed equivalence.*

Proof. First apply Definition 45.229 to the active left factor $\mathsf{mul}(a, b)$ with witness p_{ab}. This gives the equivalence between the displayed equation and $x\mathsf{hsamemul}(\mathsf{inv}(\mathsf{mul}(a, b), p_{ab}), c)$.

Next apply Definition 45.213 with the three witnesses p_{ab}, p_a, and p_b. It identifies the certified inverse of the product with the reversed product $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$.

For the forward direction, use multiplication congruence on the left factor of $\mathsf{mul}(-, c)$ to transport the one-factor solution along the inverse product reversal. The transported right side is exactly the displayed reversed product solution.

For the reverse direction, use symmetry of the inverse product reversal and the same multiplication congruence to transport the displayed reversed product solution back to $x\mathsf{hsamemul}(\mathsf{inv}(\mathsf{mul}(a, b), p_{ab}), c)$. The reverse half of the one-factor exactness theorem then recovers $\mathsf{mul}(\mathsf{mul}(a, b), x)\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product left mul equation exact from apartness`

Theorem 45.285 (Field product left-multiplication empty equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x : \mathsf{BHist}$ *satisfy*

$$\mathsf{mul}(\mathsf{mul}(a, b), x)\mathsf{hsame}\mathsf{e} \iff x\mathsf{hsame}\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)), \mathsf{e}).$$

Proof. Apply left-multiplication equation exactness to the active factor $\mathsf{mul}(a, b)$ at target e. Product inverse reversal identifies the certified inverse of that factor with the reversed product $\mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))$, and multiplication congruence transports the solution across the left factor of $\mathsf{mul}(-, \mathsf{e})$ in both directions. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product left mul empty equation exact from apartness`

Theorem 45.286 (Field empty-unit right multiplication cancellation from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate whose multiplicative unit is* e*, suppose multiplication is associative, has a right unit law at* e*, respects* hsame*, and has a right inverse law indexed by nonzero witnesses. For every nonzero witness* $p_b : \mathsf{NonZero}_F(b)$*,*

$$\mathsf{mul}(x, b)\mathsf{hsame}\mathsf{mul}(y, b) \implies x\mathsf{hsame}y.$$

Proof. Apply right-multiplication equation solving to the displayed product equality. It identifies x with $\mathsf{mul}(\mathsf{mul}(y, b), \mathsf{inv}(b, p_b))$. The right inverse cancellation theorem collapses this latter history to y, and transitivity of hsame gives the stated cancellation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field empty unit right mul cancel from apartness`

Theorem 45.287 (Field empty-unit left multiplication cancellation from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate whose multiplicative unit is* e*, suppose multiplication is associative, has a left unit law at* e*, respects* hsame*, and has a left inverse law indexed by nonzero witnesses. For every nonzero witness* $p_a : \mathsf{NonZero}_F(a)$*,*

$$\mathsf{mul}(a, x)\mathsf{hsame}\mathsf{mul}(a, y) \implies x\mathsf{hsame}y.$$

Proof. Multiply the displayed product comparison on the left by $\mathsf{inv}(a, p_a)$ using multiplication congruence. The left inverse cancellation theorem identifies the transported left endpoint with x and the transported right endpoint with y. Symmetry on the first identification and transitivity through the transported comparison give $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field empty unit left mul cancel from apartness`

Theorem 45.288 (Field product-factor map classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x, y : \mathsf{BHist}$ *satisfy both classifier clauses*

$$\mathsf{mul}(x, \mathsf{mul}(a, b))\mathsf{hsame}\mathsf{mul}(y, \mathsf{mul}(a, b)) \iff x\mathsf{hsame}y$$

and

$$\mathsf{mul}(\mathsf{mul}(a,b),x)\mathsf{hsamemul}(\mathsf{mul}(a,b),y) \iff x\mathsf{hsame}y.$$

Traditional target product-factor multiplication classifier exactness; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations product right and left equation exactness, multiplication congruence, symmetry, and transitivity; ledger the three nonzero witnesses; exactness predicate the two displayed equivalences.*

Proof. For the right-product forward direction, apply Definition 45.282 to the equation with target $\mathsf{mul}(y,\mathsf{mul}(a,b))$. This turns the given classifier equality into the solution endpoint determined by that target. Apply the same theorem to the reflexive equation for y at the same target, and compose the two solution endpoints by symmetry and transitivity to obtain $x\mathsf{hsame}y$.

For the reverse direction of the right-product clause, multiplication congruence transports $x\mathsf{hsame}y$ through the map $z \mapsto \mathsf{mul}(z,\mathsf{mul}(a,b))$, giving the displayed classifier equality.

The left-product forward direction is symmetric. Apply Definition 45.284 to the equation with target $\mathsf{mul}(\mathsf{mul}(a,b),y)$, compare it with the reflexive equation for y, and compose the resulting solution endpoints by symmetry and transitivity.

For the reverse direction of the left-product clause, multiplication congruence transports $x\mathsf{hsame}y$ through $z \mapsto \mathsf{mul}(\mathsf{mul}(a,b),z)$, which is exactly the left classifier equality. □

Theorem 45.289 (Field empty-unit conjugation equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate whose multiplicative unit is* e*, with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by a nonzero witness* $p_a : \mathsf{NonZero}_F(a)$*, conjugation by a solves the middle equation exactly:*

$$\mathsf{hsame}(\mathsf{mul}(\mathsf{mul}(a,x),\mathsf{inv}(a,p_a)),b) \iff \mathsf{hsame}(x,\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(a,p_a),b),a)).$$

Proof. For the forward direction, multiply the displayed conjugation equation on the right by a. Reassociate to expose $\mathsf{mul}(\mathsf{inv}(a,p_a),a)$, collapse it by the indexed left inverse law, and use the right unit law to obtain $\mathsf{mul}(a,x)\mathsf{hsamemul}(b,a)$. Then multiply on the left by $\mathsf{inv}(a,p_a)$, reassociate, collapse the left inverse pair, and use the left unit law.

For the reverse direction, transport the displayed solved endpoint through left multiplication by a. Reassociate to expose $\mathsf{mul}(a,\mathsf{inv}(a,p_a))$, collapse it by the indexed right inverse law, and use the left unit law. A final right multiplication by $\mathsf{inv}(a,p_a)$, followed by reassociation, the indexed right inverse law, and the right unit law, recovers the original conjugation equation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field empty unit conjugation equation exact from apartness`

Theorem 45.290 (Field empty-unit conjugation classifier exactness from apartness)**.** *Under the same empty-unit* $\mathsf{Field}^{\uparrow}$ *hypotheses and witness* $p_a : \mathsf{NonZero}_F(a)$*, conjugation by a preserves and reflects the* hsame *classifier:*

$$\mathsf{hsame}(\mathsf{mul}(\mathsf{mul}(a,x),\mathsf{inv}(a,p_a)),\mathsf{mul}(\mathsf{mul}(a,y),\mathsf{inv}(a,p_a))) \iff \mathsf{hsame}(x,y).$$

Proof. For the forward direction, apply Definition 45.289 to the displayed comparison with target $\mathsf{mul}(\mathsf{mul}(a,y),\mathsf{inv}(a,p_a))$. Apply the same equation exactness theorem to the reflexive equation for y at that target. The two solved endpoints compare by symmetry and transitivity of hsame.

For the reverse direction, multiplication congruence transports $x\mathsf{hsame}y$ through left multiplication by a and right multiplication by $\mathsf{inv}(a,p_a)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field empty unit conjugation classifier exact from apartness`

Theorem 45.291 (Field right product empty-context solution determinacy from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses and the displayed associativity, unit, congruence, and inverse laws, let* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a,b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$*. If* x *and* y *solve the same right product equation at the empty-context endpoint* $\mathsf{append}(\mathsf{e}, c)$*, then* $x\mathsf{hsame}y$*:*

$$\mathsf{mul}(x, \mathsf{mul}(a,b))\mathsf{hsameappend}(\mathsf{e}, c) \quad \textit{and} \quad \mathsf{mul}(y, \mathsf{mul}(a,b))\mathsf{hsameappend}(\mathsf{e}, c) \Rightarrow x\mathsf{hsame}y.$$

Proof. Apply the right product equation exactness theorem to both displayed equations at the shared endpoint $\mathsf{append}(\mathsf{e}, c)$. Both solutions are classified by the same explicit inverse-product endpoint, so symmetry and transitivity of hsame compare them. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product right mul empty context solution deterministic from apartness`

Theorem 45.292 (Field left product empty-context solution determinacy from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses and the displayed associativity, unit, congruence, and inverse laws, let* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a,b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$*. If* x *and* y *solve the same left product equation at the empty-context endpoint* $\mathsf{append}(\mathsf{e}, c)$*, then* $x\mathsf{hsame}y$*:*

$$\mathsf{mul}(\mathsf{mul}(a,b), x)\mathsf{hsameappend}(\mathsf{e}, c) \quad \textit{and} \quad \mathsf{mul}(\mathsf{mul}(a,b), y)\mathsf{hsameappend}(\mathsf{e}, c) \Rightarrow x\mathsf{hsame}y.$$

Proof. Apply the left product equation exactness theorem to both displayed equations at the shared endpoint $\mathsf{append}(\mathsf{e}, c)$. The two solved endpoints are identical, and hsame symmetry and transitivity give $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product left mul empty context solution deterministic from apartness`

Theorem 45.293 (Field product-factor map empty-context classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, explicit witnesses* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a,b))$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$ *make every* $x, y : \mathsf{BHist}$ *satisfy both empty-context classifier clauses*

$$\mathsf{mul}(x, \mathsf{mul}(a,b))\mathsf{hsameappend}(\mathsf{e}, \mathsf{mul}(y, \mathsf{mul}(a,b))) \iff x\mathsf{hsame}y$$

and

$$\mathsf{mul}(\mathsf{mul}(a,b), x)\mathsf{hsameappend}(\mathsf{e}, \mathsf{mul}(\mathsf{mul}(a,b), y)) \iff x\mathsf{hsame}y.$$

Proof. Apply the product right-multiplication equation exactness theorem with target $\mathsf{mul}(y, \mathsf{mul}(a,b))$ under the explicit empty continuation context. Applying the same exactness theorem to the reflexive endpoint for y gives the solver endpoint for y; transitivity and symmetry compare the two solver endpoints and yield $x\mathsf{hsame}y$. The reverse direction is multiplication congruence transported through the empty-context endpoint.

The left-product clause is identical, using the left-multiplication equation exactness theorem with target $\mathsf{mul}(\mathsf{mul}(a,b), y)$ under the same empty context. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product factor map empty context classifier exact from apartness`

Theorem 45.294 (Field three-factor product equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, left and right inverse laws indexed by nonzero witnesses, product nonzero exclusion, and the apartness-to-nonzero conversion maps used in Definition 45.220, any witness*

$$p_{abc} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a,b),c))$$

induces witnesses $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*, and* $p_c : \mathsf{NonZero}_F(c)$ *such that every* $x, q : \mathsf{BHist}$ *satisfies*

$$\mathsf{mul}(x, \mathsf{mul}(\mathsf{mul}(a,b),c))\mathsf{hsame}q \iff x\mathsf{hsamemul}(q, \mathsf{mul}(\mathsf{inv}(c,p_c), \mathsf{mul}(\mathsf{inv}(b,p_b), \mathsf{inv}(a,p_a)))).$$

It also satisfies

$$\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),x)\mathsf{hsame}q \iff x\mathsf{hsamemul}(\mathsf{mul}(\mathsf{inv}(c,p_c), \mathsf{mul}(\mathsf{inv}(b,p_b), \mathsf{inv}(a,p_a))), q).$$

Traditional target three-factor product equation solving; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations one-factor right and left equation exactness, three-factor inverse reversal, multiplication congruence, symmetry, and transitivity; ledger* p_{abc} *and the induced factor witnesses* p_a, p_b, p_c*; core theorem this block; exactness predicate the two displayed equivalences; standard bridge deferred; downstream use finite product equation solving for* $\mathsf{Field}^{\uparrow}$*.*

Proof. First apply Definition 45.224 to the active right factor $\mathsf{mul}(\mathsf{mul}(a,b),c)$ with witness p_{abc}. This gives the right-multiplication equation equivalence with the one-factor solution $x\mathsf{hsamemul}(q, \mathsf{inv}(\mathsf{mul}(\mathsf{mul}(a,b)$

Apply Definition 45.220 to p_{abc}. It produces the factor witnesses p_a, p_b, and p_c and identifies the indexed inverse of $\mathsf{mul}(\mathsf{mul}(a,b),c)$ with the reversed product $\mathsf{mul}(\mathsf{inv}(c,p_c), \mathsf{mul}(\mathsf{inv}(b,p_b), \mathsf{inv}(a,p_a)))$.

For the forward direction of the right equation, use multiplication congruence on the right factor of $\mathsf{mul}(q,-)$ to transport the one-factor solution along that inverse equality.

For the reverse direction of the right equation, use symmetry of the same inverse equality, the same congruence, and transitivity to return to the indexed-inverse solution before applying the reverse half of the one-factor exactness theorem.

The left equation is identical with the active factor on the left. Apply Definition 45.229 to $\mathsf{mul}(\mathsf{mul}(a,b),c)$ with witness p_{abc}, then transport the indexed inverse through the first factor of $\mathsf{mul}(-,q)$ by multiplication congruence.

Symmetry and transitivity give the reverse transport, and the reverse half of the one-factor left exactness theorem recovers $\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),x)\mathsf{hsame}q$. □

Theorem 45.295 (Field four-factor product equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, and left and right inverse laws indexed by nonzero witnesses, product nonzero exclusion, and the conversion maps required by Definition 45.221, define*

$$P_4 := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d).$$

For any witness $p_{abcd} : \mathsf{NonZero}_F(P_4)$*, the four-factor inverse reversal theorem induces witnesses* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*, and* $p_d : \mathsf{NonZero}_F(d)$*. Define the reversed inverse spine*

$$R_4 := \mathsf{mul}(\mathsf{inv}(d,p_d), \mathsf{mul}(\mathsf{inv}(c,p_c), \mathsf{mul}(\mathsf{inv}(b,p_b), \mathsf{inv}(a,p_a)))).$$

Then every $x, r : \mathsf{BHist}$ *satisfies both clauses*

$$\mathsf{mul}(x, P_4)\mathsf{hsame}r \iff x\mathsf{hsamemul}(r, R_4)$$

and

$$\mathsf{mul}(P_4, x)\mathsf{hsame}r \iff x\mathsf{hsamemul}(R_4, r).$$

Traditional target finite product equation solving; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations one-factor equation exactness, inverse-product reversal, congruence, associativity, unit laws, product nonzero exclusion, and conversion maps; ledger* p_{abcd}*, induced factor witnesses, and conversion maps; core theorem this block; exactness predicate the two displayed equivalences; standard bridge deferred; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 45.224 to the active fourfold right factor P_4 with witness p_{abcd}. This gives the right-multiplication equation in terms of the certified inverse $\mathsf{inv}(P_4, p_{abcd})$.

Invoke Definition 45.221 on p_{abcd} and the conversion maps. It produces the factor witnesses p_a, p_b, p_c, p_d and identifies $\mathsf{inv}(P_4, p_{abcd})$ with the displayed reversed spine R_4.

For the right equation, transport the solution side through multiplication congruence for $\mathsf{mul}(r, -)$. Symmetry and transitivity of hsame move between the certified inverse of the active product and R_4, yielding $x\mathsf{hsamemul}(r, R_4)$ in the forward direction and returning to the one-factor exactness statement in the reverse direction.

For the left equation, apply Definition 45.229 to the active left factor P_4 with witness p_{abcd}. This gives the left equation in terms of $\mathsf{inv}(P_4, p_{abcd})$ on the left of r.

Use the same inverse identification and multiplication congruence for $\mathsf{mul}(-, r)$. The identified spine now appears on the left, so the transported solution is $x\mathsf{hsamemul}(R_4, r)$, and the reverse transport recovers $\mathsf{mul}(P_4, x)\mathsf{hsame}r$. □

Theorem 45.296 (Field five-factor product equation exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with proof-indexed inverses, multiplication associativity, left and right unit laws, multiplication congruence under* hsame*, indexed left and right inverse laws, product nonzero exclusion, and the conversion maps required by Definition 45.574, define*

$$P_5 := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e).$$

For any witness $p_{abcde} : \mathsf{NonZero}_F(P_5)$*, the five-factor inverse reversal theorem induces witnesses* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*,* $p_d : \mathsf{NonZero}_F(d)$*, and* $p_e : \mathsf{NonZero}_F(e)$*. Define*

$$R_5 := \mathsf{mul}(\mathsf{inv}(e, p_e), \mathsf{mul}(\mathsf{inv}(d, p_d), \mathsf{mul}(\mathsf{inv}(c, p_c), \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a))))).$$

Then every $x, r : \mathsf{BHist}$ *satisfies both clauses*

$$\mathsf{mul}(x, P_5)\mathsf{hsame}r \iff x\mathsf{hsamemul}(r, R_5)$$

and

$$\mathsf{mul}(P_5, x)\mathsf{hsame}r \iff x\mathsf{hsamemul}(R_5, r).$$

Traditional target finite five-factor product equation solving; BEDC source $\mathsf{Field}^{\uparrow}$ *over* BHist*; carrier* BHist*; classifier* hsame*; stability obligations one-factor right and left equation exactness, five-factor inverse reversal, multiplication congruence, symmetry, and transitivity; ledger* p_{abcde} *and the induced witnesses* p_a, p_b, p_c, p_d, p_e*; core theorem this block; exactness predicate the two displayed equivalences; standard bridge deferred; status* $\mathsf{ProofTarget}$*.*

Proof. First apply Definition 45.224 to the active right factor P_5 with witness p_{abcde}. This gives the right equation equivalence with the one-factor solution $x\mathsf{hsamemul}(r, \mathsf{inv}(P_5, p_{abcde}))$.

Next apply Definition 45.574 to p_{abcde}. It produces the factor witnesses p_a, p_b, p_c, p_d, and p_e and identifies the indexed inverse of P_5 with the displayed reversed spine R_5.

For the forward direction of the right equation, use multiplication congruence on the right factor of $\mathsf{mul}(r, -)$ to transport the one-factor solution from $\mathsf{inv}(P_5, p_{abcde})$ to R_5.

For the reverse direction of the right equation, use symmetry of the same inverse identification, the same congruence, and transitivity of hsame to return to the indexed-inverse solution before applying the reverse half of the one-factor right exactness theorem.

For the left equation, apply Definition 45.229 to the active left factor P_5 with witness p_{abcde}. This gives the left equation equivalence with $x\mathsf{hsamemul}(\mathsf{inv}(P_5, p_{abcde}), r)$.

Transport this solution through multiplication congruence for $\mathsf{mul}(-, r)$ and the five-factor inverse identification. Symmetry and transitivity give the reverse transport, so the reverse half of the one-factor left exactness theorem recovers $\mathsf{mul}(P_5, x)\mathsf{hsame}r$. □

Theorem 45.297 (Field equal-squares signed factors cannot both be nonzero)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.272 and Definition 45.248, every* $a, b : \mathsf{BHist}$ *satisfies*

$$a \cdot a\mathsf{hsame}b \cdot b \quad \Longrightarrow \quad \mathsf{NonZero}_F(a + b) \to \mathsf{NonZero}_F(a + (-b)) \to \mathsf{False}.$$

The traditional target is constructive signed-factor square reasoning; the BEDC source is $\mathsf{Field}^{\uparrow}$ *together with* $\mathsf{CommRing}^{\uparrow}$*; the carrier is* BHist*; the classifier is* hsame*; the ledger consists of the two explicit signed-factor nonzero witnesses; the core theorem is this block; the exactness predicate is impossibility of simultaneous nonzero signed factors under equal squares; the status is* $\mathsf{CheckedCert}$*.*

Proof. Assume $a \cdot a\mathsf{hsame}b \cdot b$ and witnesses $p_+ : \mathsf{NonZero}_F(a + b)$ and $p_- : \mathsf{NonZero}_F(a + (-b))$.

First apply Definition 44.47 to the square equality. The signed-factor product is classified as zero:

$$(a + b) \cdot (a + (-b))\mathsf{hsame}0.$$

Now apply Definition 45.248 to the two nonzero witnesses p_+ and p_- together with this displayed zero product. That theorem says that such a zero product is impossible when both factors are explicitly nonzero, so the assumed pair of signed-factor witnesses yields False. □

Lean4 ✓ `BEDC.Derived.FieldUp.field equal squares signed factors not both nonzero`

Theorem 45.298 (Field affine two-sided equation solutions are unique from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, \mathsf{zero}, \mathsf{one}$ *on* BHist*, additive group laws for* $\mathsf{add}, \mathsf{neg}, \mathsf{zero}$*, addition congruence, multiplication associativity, left and right multiplicative identity laws, multiplication congruence, and indexed inverse laws, every* $a, b, x, y, d, c : \mathsf{BHist}$ *and witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *satisfy*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c \quad \Longrightarrow \quad \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, y), b), d)\mathsf{hsame}c \quad \Longrightarrow \quad x\mathsf{hsame}y.$$

Proof. Assume the two displayed affine equations. Use symmetry of the second equation and transitivity of hsame to compose the first equation with it. This gives the classifier equality between the two affine outputs:

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, y), b), d).$$

Apply Definition 45.278 with the same certified nonzero factors p_a, p_b and offset d. The affine classifier exactness theorem removes the shared affine map and yields $x\mathsf{hsame}y$. □

↪variant BEDC.Derived.FieldUp.field affine certified fiber singleton

Per-object certificate: Traditional target: affine equation uniqueness. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist plus additive group laws. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: the operations and laws required by affine two-sided injectivity. Ledger: the two certified factors and offset. Core theorem list entry: Theorem 45.298. Exactness predicate: two certified solutions of the same affine equation are classified equal. Standard bridge and downstream use: canonical solution extraction for FieldUp equation solving. Status: $\mathsf{CheckedCert}$.

45.8 Signed square-factor consequences

Theorem 45.299 (Field equal-squares signed factor endpoint collapse)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.272, together with the additive associativity, additive right identity, addition congruence, and additive right-inverse laws needed to instantiate Definition 41.116 for* add*, every* $a, b : \mathsf{BHist}$ *with* $a \cdot a\mathsf{hsame}b \cdot b$ *satisfies the two endpoint clauses:*

1. *if* $p_+ : \mathsf{NonZero}_F(a + b)$*, then* $a\mathsf{hsame}b$*;*
2. *if* $p_- : \mathsf{NonZero}_F(a + (-b))$*, then* $a\mathsf{hsame} - b$*.*

Traditional target signed square-root ambiguity; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow} + \mathsf{Group}^{\uparrow}$*; carrier* BHist*; classifier* hsame*; operations* $\mathsf{add}, \mathsf{mul}, \mathsf{neg}, 0, 1$*; stability obligations signed-factor exclusion and additive right cancellation; ledger the chosen signed-factor nonzero witness; core theorem this block; exactness predicate endpoint collapse under a certified signed factor; status* $\mathsf{ProofTarget}$*.*

Proof. Assume $a \cdot a\mathsf{hsame}b \cdot b$.

For the $p_+ : \mathsf{NonZero}_F(a + b)$ branch, apply Definition 45.272 to the square equality and the witness p_+. The first signed-factor clause gives $a + (-b)\mathsf{hsame}0$.

The additive right-inverse law also gives $b + (-b)\mathsf{hsame}0$. By symmetry and transitivity of hsame, the two displayed sums are classified together:

$$a + (-b)\mathsf{hsame}b + (-b).$$

Now instantiate Definition 41.116 for the additive group operation add. Cancelling the common right summand $-b$ yields $a\mathsf{hsame}b$.

For the $p_- : \mathsf{NonZero}_F(a + (-b))$ branch, apply Definition 45.272 to the same square equality and the witness p_-. The second signed-factor clause gives $a + b\mathsf{hsame}0$.

The additive left-inverse law gives $(-b) + b\mathsf{hsame}0$. Comparing this equality with $a + b\mathsf{hsame}0$ and using symmetry and transitivity of hsame yields

$$a + b\mathsf{hsame}(-b) + b.$$

A second instance of additive right cancellation removes the common right summand b, leaving $a\mathsf{hsame} - b$. □

Lean4 ✓ BEDC.Derived.FieldUp.field equal squares signed factor endpoint collapse

Theorem 45.300 (Field equal-squares signed branch determinacy)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the joint hypotheses needed to instantiate Definition 45.299 and Definition 45.297, every* $a, b : \mathsf{BHist}$ *with*

$$a \cdot a\mathsf{hsame}b \cdot b$$

satisfies the two deterministic signed-branch clauses:

1. *if* $p_+ : \mathsf{NonZero}_F(a+b)$*, then* $a\mathsf{hsame}b$ *and* $\mathsf{NonZero}_F(a+(-b)) \to \mathsf{False}$*;*

2. *if* $p_- : \mathsf{NonZero}_F(a+(-b))$*, then* $a\mathsf{hsame} - b$ *and* $\mathsf{NonZero}_F(a+b) \to \mathsf{False}$*.*

Traditional target signed square-root branch choice; BEDC source $\mathsf{Field}^{\uparrow}$ *with the signed-factor and additive-cancellation hypotheses; carrier* BHist*; classifier* hsame*; ledger the chosen signed-factor witness; core theorem this block; exactness predicate endpoint identification together with exclusion of the opposite signed witness; status* $\mathsf{ProofTarget}$*.*

Proof. Assume $a \cdot a\mathsf{hsame}b \cdot b$.

For the $p_+ : \mathsf{NonZero}_F(a+b)$ branch, apply Definition 45.299 to the square equality and the witness p_+. Its positive signed-factor endpoint clause gives $a\mathsf{hsame}b$.

To exclude the opposite witness, assume $q_- : \mathsf{NonZero}_F(a+(-b))$. Apply Definition 45.297 to the same square equality, the chosen witness p_+, and the hypothetical witness q_-. The result is False, so the positive branch carries both the endpoint and the opposite-witness contradiction.

For the $p_- : \mathsf{NonZero}_F(a+(-b))$ branch, apply Definition 45.299 to the square equality and the witness p_-. Its negative signed-factor endpoint clause gives $a\mathsf{hsame} - b$.

To exclude the opposite witness, assume $q_+ : \mathsf{NonZero}_F(a+b)$. Apply Definition 45.297 to the same square equality, the witness q_+, and the chosen witness p_-. This produces False and completes the negative branch package. □

Lean4 ✓ `BEDC.Derived.FieldUp.field equal squares signed branch determinacy`

Theorem 45.301 (Field zero linear factor signed branch selection)**.** *For BEDC history operations add, mul, neg, and zero satisfying the hypotheses of Definition 44.80 and Definition 45.300, every* $a, b : \mathsf{BHist}$ *satisfies the following two signed zero-factor clauses:*

1. *if* $(a+(-b))\mathsf{hsame}0$ *and* $p_+ : \mathsf{NonZero}_F(a+b)$*, then* $a\mathsf{hsame}b$ *and* $\mathsf{NonZero}_F(a+(-b)) \to \mathsf{False}$*;*

2. *if* $(a+b)\mathsf{hsame}0$ *and* $p_- : \mathsf{NonZero}_F(a+(-b))$*, then* $a\mathsf{hsame} - b$ *and* $\mathsf{NonZero}_F(a+b) \to \mathsf{False}$*.*

Proof. Assume first the zero factor $(a+(-b))\mathsf{hsame}0$ and the signed nonzero witness $p_+ : \mathsf{NonZero}_F(a+b)$. Apply Definition 44.80 to the displayed right signed zero factor, read as the second disjunct. This gives the equal square row $a \cdot a\mathsf{hsame}b \cdot b$.

Now apply the positive branch of Definition 45.300 to that square row and the witness p_+. Its conclusion is exactly $a\mathsf{hsame}b$ together with $\mathsf{NonZero}_F(a+(-b)) \to \mathsf{False}$.

For the opposite branch, assume the zero factor $(a+b)\mathsf{hsame}0$ and the signed nonzero witness $p_- : \mathsf{NonZero}_F(a+(-b))$. Apply Definition 44.80 to the displayed left signed zero factor, read as the first disjunct. This again gives $a \cdot a\mathsf{hsame}b \cdot b$.

The negative branch of Definition 45.300, applied to this square row and to p_-, gives $a\mathsf{hsame} - b$ and excludes every witness $\mathsf{NonZero}_F(a+b)$. □

Theorem 45.302 (Field zero linear factor signed endpoint package)**.** *For BEDC history operations add, mul, neg, and zero satisfying the hypotheses of Definition 44.80 and Definition 45.300, every* $a, b : \mathsf{BHist}$ *satisfies the following two signed endpoint clauses:*

1. *if* $(a+(-b))\mathsf{hsame}0$ *and* $p_+ : \mathsf{NonZero}_F(a+b)$*, then simultaneously* $a \cdot a\mathsf{hsame}b \cdot b$*,* $a\mathsf{hsame}b$*, and* $\mathsf{NonZero}_F(a+(-b)) \to \mathsf{False}$*;*

2. *if* $(a+b)\mathsf{hsame}0$ *and* $p_- : \mathsf{NonZero}_F(a+(-b))$*, then simultaneously* $a \cdot a\mathsf{hsame}b \cdot b$*,* $a\mathsf{hsame} - b$*, and* $\mathsf{NonZero}_F(a+b) \to \mathsf{False}$*.*

Traditional target signed square-root endpoint package; BEDC source BHist *with* CommRing$^{\uparrow}$ *and* Field$^{\uparrow}$*; carrier* BHist*; classifier* hsame*; ledger the zero factor plus signed nonzero witness; core theorem this block; exactness predicate signed branch square row plus endpoint; downstream consumer* Field$^{\uparrow}$(RatUp) *affine solution and square-root branch ledgers; status* ProofTarget*.*

Proof. Assume first the zero factor $(a+(-b))\mathsf{hsame}0$ and the signed nonzero witness $p_+ : \mathsf{NonZero}_F(a+b)$. Apply Definition 44.80 to the displayed right signed zero factor, read as the second disjunct. This gives the square row

$$a \cdot a\mathsf{hsame}b \cdot b.$$

Apply the positive branch of Definition 45.300 to this square row and to p_+. It returns $a\mathsf{hsame}b$ together with $\mathsf{NonZero}_F(a + (-b)) \to \mathsf{False}$. Conjunction introduction packages the square row, the endpoint, and the opposite-witness exclusion.

For the negative signed endpoint, assume the zero factor $(a+b)\mathsf{hsame}0$ and the signed nonzero witness $p_- : \mathsf{NonZero}_F(a + (-b))$. Apply Definition 44.80 to the displayed left signed zero factor, read as the first disjunct. This again gives

$$a \cdot a\mathsf{hsame}b \cdot b.$$

Apply the negative branch of Definition 45.300 to the square row and to p_-. It returns $a\mathsf{hsame}-b$ and $\mathsf{NonZero}_F(a + b) \to \mathsf{False}$. A second conjunction introduction packages the square row, the signed endpoint, and the excluded opposite branch. □

Per-object template: traditional target constructive signed square-root branch selection; BEDC source Field$^{\uparrow}$ + CommRing$^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations zero-linear-factor equal-square production and equal-square signed branch determinacy; ledger the zero factor and signed nonzero witness; core theorem this block; exactness predicate branch endpoint plus opposite-witness exclusion; downstream consumer Field$^{\uparrow}$(RatUp) affine solution and square-root branch ledgers; status ProofTarget.

Theorem 45.303 (Field product nonzero excludes zero factors)**.** *For a BEDC history-level* Field$^{\uparrow}$ *candidate with ring zero absorption, multiplication congruence, classifier transport of* $\mathsf{NonZero}_F$*, and contradiction of* $\mathsf{NonZero}_F(0)$*, every* $a, b : \mathsf{BHist}$ *with*

$$p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$$

satisfies both $a\mathsf{hsame}0 \to \bot$ *and* $b\mathsf{hsame}0 \to \bot$*.*

Proof. Assume $a\mathsf{hsame}0$. Multiplication congruence sends $\mathsf{mul}(a, b)$ to $\mathsf{mul}(0, b)$, and Theorem 43.39 identifies $\mathsf{mul}(0, b)$ with 0. Transitivity gives $\mathsf{mul}(a, b)\mathsf{hsame}0$.

Transport $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$ along this classifier equality. The transport clause supplies $\mathsf{NonZero}_F(0)$, and the field zero-contradiction clause closes the left-factor obstruction.

The right-factor obstruction is symmetric. From $b\mathsf{hsame}0$, multiplication congruence sends $\mathsf{mul}(a, b)$ to $\mathsf{mul}(a, 0)$, and Theorem 43.39 identifies that product with 0. Transporting the same witness p_{ab} to $\mathsf{NonZero}_F(0)$ again contradicts the field zero clause. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product nonzero excludes zero factors`

Corollary 45.304 (Field product nonzero empty-factor contradiction)**.** *Under the same* Field$^{\uparrow}$ *and* Ring$^{\uparrow}$ *hypotheses as Definition 45.303, a proof* $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$ *determines the pair* $(a\mathsf{hsame}0 \to \bot) \wedge (b\mathsf{hsame}0 \to \bot)$*.*

Proof. Apply Definition 45.303 to p_{ab}. Its two components are exactly the left-factor and right-factor contradictions against zero. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product nonzero empty factors absurd`

Theorem 45.305 (Field product nonzero of nonzero factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.248, assume also a constructive bridge family: for every* $t : \mathsf{BHist}$*, an instance*

$$\alpha_t : (t\mathsf{hsame0} \to \bot) \to \mathsf{NonZero}_F(t)$$

from apartness from zero into the nonzero classifier. Then witnesses $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *determine*

$$\mathsf{NonZero}_F(\mathsf{mul}(a, b)).$$

Proof. Theorem 45.248 turns p_a and p_b into a constructive refutation of $\mathsf{mul}(a, b)\mathsf{hsame0}$. Applying the bridge-family instance at $\mathsf{mul}(a, b)$ to this product apartness gives the displayed product nonzero witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.field product nonzero of nonzero factors`

Theorem 45.306 (Field binary product nonzero iff factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate under the hypotheses of Definition 45.303 and Definition 45.305, assume also the same constructive bridge family: for every* $t : \mathsf{BHist}$*, an instance*

$$\alpha_t : (t\mathsf{hsame0} \to \bot) \to \mathsf{NonZero}_F(t).$$

Then every $a, b : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(a, b)) \iff (\mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b)).$$

Proof. For the forward direction, let $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$. Apply Definition 45.303 to obtain $a\mathsf{hsame0} \to \bot$ and $b\mathsf{hsame0} \to \bot$. Applying α_a and α_b gives $\mathsf{NonZero}_F(a)$ and $\mathsf{NonZero}_F(b)$.

For the reverse direction, let $p_a : \mathsf{NonZero}_F(a)$ and $p_b : \mathsf{NonZero}_F(b)$. Apply Definition 45.305 to p_a and p_b, using the same bridge family at $\mathsf{mul}(a, b)$, to obtain $\mathsf{NonZero}_F(\mathsf{mul}(a, b))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field binary product nonzero iff factors`

Per-object certificate: Traditional target: binary product nonzero exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ plus the constructive zero-apartness bridge. Carrier choice: BHist. Classifier choice: $\mathsf{NonZero}_F$. Stability obligations: zero-factor exclusion and nonzero-product construction. Ledger: factors a, b and product $\mathsf{mul}(a, b)$. Core theorem list: Theorem 45.306. Exactness predicate: binary nonzero product exactly matches the pair of factor nonzero classifiers. Standard bridge: the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ product classifier/nonzero exit package. Status: ProofTarget.

Per-object certificate: Traditional target: certified nonzero product prevents zero factors. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: zero absorption, multiplication congruence, $\mathsf{NonZero}_F$ transport, and contradiction of $\mathsf{NonZero}_F(0)$. Ledger: product witness $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$ and the tested zero-factor classifier. Core theorem list: left and right zero-factor exclusion from a certified nonzero product. Exactness predicate: nonzero-product obstruction to zero-factor classifiers. Standard bridge: deferred conservative bridge. Status: ProofTarget.

Theorem 45.307 (Field two-sided product nonzero excludes middle zero without outer apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.303, together with the ring zero-absorption and multiplication-congruence fields used there, every* $a, b, x : \mathsf{BHist}$ *with product witness*

$$p_{axb} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b)),$$

satisfies $x\mathsf{hsame0} \to \bot$.

Proof. Assume $x\mathsf{hsame0}$. Multiplication congruence sends $\mathsf{mul}(a, x)$ to $\mathsf{mul}(a, 0)$, and Theorem 43.39 identifies $\mathsf{mul}(a, 0)$ with 0. Transitivity gives

$$\mathsf{mul}(a, x)\mathsf{hsame0}.$$

Apply Definition 45.303 to the product with left factor $\mathsf{mul}(a, x)$, right factor b, and witness p_{axb}. Its left zero-factor obstruction rejects the displayed classifier, so the assumed middle classifier $x\mathsf{hsame0}$ yields $\bot$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided product nonzero excludes middle zero`

Per-object certificate: Traditional target: two-sided nonzero-product middle-factor obstruction. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: product zero-factor exclusion and ring zero absorption. Ledger: $p_{axb} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b))$ and the tested middle-zero classifier. Core theorem list: Theorem 45.307. Exactness predicate: obstruction to middle zero. Standard bridge: deferred. Status: ProofTarget.

Theorem 45.308 (Field certified three-factor product excludes zero factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.303, including the ring zero-absorption and multiplication congruence hypotheses used there, every* $a, x, b : \mathsf{BHist}$ *with*

$$p_{axb} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b))$$

satisfies

$$(a\mathsf{hsame0} \to \bot) \wedge (x\mathsf{hsame0} \to \bot) \wedge (b\mathsf{hsame0} \to \bot).$$

Proof. Apply Definition 45.303 to the binary product $(\mathsf{mul}(a, x), b)$ and the witness p_{axb}. This gives an obstruction to $\mathsf{mul}(a, x)\mathsf{hsame0}$ and an obstruction to $b\mathsf{hsame0}$. The second obstruction is exactly the right-factor clause.

Assume $a\mathsf{hsame0}$. Multiplication congruence sends $\mathsf{mul}(a, x)$ to $\mathsf{mul}(0, x)$, and Definition 43.39 classifies $\mathsf{mul}(0, x)$ as 0. By transitivity, $\mathsf{mul}(a, x)\mathsf{hsame0}$, contradicting the first obstruction.

Assume $x\mathsf{hsame0}$. Multiplication congruence sends $\mathsf{mul}(a, x)$ to $\mathsf{mul}(a, 0)$, and Definition 43.39 classifies $\mathsf{mul}(a, 0)$ as 0. By transitivity, $\mathsf{mul}(a, x)\mathsf{hsame0}$ again contradicts the first obstruction. □

Lean4 ✓ `BEDC.Derived.FieldUp.field nested product nonzero excludes empty factors`

Per-object certificate: Traditional target: zero-factor exclusion for certified triple products. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption over BHist. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: product nonzero transport and zero absorption. Ledger: p_{axb} and the tested zero classifier. Core theorem list: Theorem 45.308. Exactness predicate: no zero factor under a certified nonzero threefold product. Standard bridge: deferred. Status: ProofTarget.

Theorem 45.309 (Field two-sided product nonzero iff middle factor nonzero)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with the hypotheses of Definition 45.305 and Definition 45.307, assume certified outer witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*. Then every* $x : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b)) \iff \mathsf{NonZero}_F(x).$$

Proof. For the forward direction, assume $p_{axb} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b))$. Theorem 45.307 turns this witness into a refutation of $x\mathsf{hsame}0$. The apartness-to-nonzero bridge required by Definition 45.305 then gives $\mathsf{NonZero}_F(x)$.

For the reverse direction, assume $p_x : \mathsf{NonZero}_F(x)$. Apply Definition 45.305 to p_a and p_x to obtain $p_{ax} : \mathsf{NonZero}_F(\mathsf{mul}(a, x))$.

Apply the same product-closure theorem to p_{ax} and p_b. The resulting witness is exactly $\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b))$, completing the two-sided equivalence. □

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided product nonzero middle factor iff`

Per-object certificate: Traditional target: exact middle-factor nonzero transport through certified two-sided products. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: nonzero transport, product closure, product zero-factor exclusion, and ring zero absorption. Ledger: outer certificates p_a, p_b, middle certificate p_x, intermediate product witness p_{ax}, and two-sided product witness p_{axb}. Core theorem list: Theorem 45.309. Exactness predicate: nonzero of the middle factor is equivalent to nonzero of the certified outer product. Downstream use: FieldUp exact nonzero propagation for affine and denominator-product reasoning. Status: ProofTarget.

Theorem 45.310 (Field certified four-factor product nonzero iff factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with the hypotheses of Definition 45.305 and the zero-factor exclusion hypotheses used in Definition 45.308, assume an apartness-to-nonzero bridge family: for every* $t : \mathsf{BHist}$*, an instance*

$$\alpha_t : (t\mathsf{hsame}0 \to \bot) \to \mathsf{NonZero}_F(t).$$

Then every $a, b, c, d : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d)) \iff \mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b) \wedge \mathsf{NonZero}_F(c) \wedge \mathsf{NonZero}_F(d).$$

Proof. For the forward direction, assume $p_{abcd} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d))$. Apply Definition 45.303 to the outer binary product. This gives apartness of $\mathsf{mul}(\mathsf{mul}(a, b), c)$ from 0 and apartness of d from 0. The bridge-family instances at $\mathsf{mul}(\mathsf{mul}(a, b), c)$ and d supply $p_{abc} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, b), c))$ and $\mathsf{NonZero}_F(d)$.

Apply the same zero-factor exclusion theorem to the left threefold product, viewed as the binary product $\mathsf{mul}(\mathsf{mul}(a, b), c)$ with witness p_{abc}. This gives apartness of $\mathsf{mul}(a, b)$ from 0 and apartness of c from 0. The bridge-family instances at $\mathsf{mul}(a, b)$ and c supply $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$ and $\mathsf{NonZero}_F(c)$.

Apply the binary zero-factor exclusion theorem once more to $\mathsf{mul}(a, b)$ with witness p_{ab}. The two resulting apartness witnesses are converted by the bridge-family instances at a and b into $\mathsf{NonZero}_F(a)$ and $\mathsf{NonZero}_F(b)$. Pair these four witnesses in the displayed order.

For the reverse direction, assume witnesses $p_a : \mathsf{NonZero}_F(a)$, $p_b : \mathsf{NonZero}_F(b)$, $p_c : \mathsf{NonZero}_F(c)$, and $p_d : \mathsf{NonZero}_F(d)$. Apply Definition 45.305 to p_a and p_b to obtain $p_{ab} : \mathsf{NonZero}_F(\mathsf{mul}(a, b))$.

Apply the product-closure theorem to p_{ab} and p_c to obtain $p_{abc} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, b), c))$. Applying it one final time to p_{abc} and p_d gives the displayed fourfold product nonzero witness. □

Per-object certificate: Traditional target: four-factor product nonzero exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Ledger: factor witnesses p_a, p_b, p_c, p_d and intermediate product witnesses $p_{ab}, p_{abc}, p_{abcd}$. Core theorem list: Theorem 45.310. Exactness predicate:

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d)) \iff \mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b) \wedge \mathsf{NonZero}_F(c) \wedge \mathsf{NonZero}_F(d).$$

Downstream use: four-factor product equation exactness and inverse-reversal proof targets. Status: ProofTarget.

Theorem 45.311 (Field nonzero product rigidity)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.247, the classifier-transport and zero-contradiction rows of Definition 45.201, and the zero-apartness interface of Definition 45.249, let* $a, b : \mathsf{BHist}$ *be equipped with witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*. If the product endpoint carries an apartness-to-nonzero conversion*

$$\alpha_{ab} : (\mathsf{mul}(a,b)\mathsf{hsame}0 \to \bot) \to \mathsf{NonZero}_F(\mathsf{mul}(a,b)),$$

then

$$\mathsf{NonZero}_F(\mathsf{mul}(a,b)).$$

Proof. It is enough to construct zero-apartness of the product endpoint. Assume $z : \mathsf{mul}(a,b)\mathsf{hsame}0$. Apply Definition 45.247 with the left-factor witness p_a. Its left-factor clause sends z to $b\mathsf{hsame}0$.

Transport $p_b : \mathsf{NonZero}_F(b)$ along this classifier equality by the nonzero-transport row of Definition 45.201. This gives $\mathsf{NonZero}_F(0)$, contradicting the zero-contradiction row of the same stability certificate. Hence $\mathsf{mul}(a,b)\mathsf{hsame}0 \to \bot$.

Applying the displayed conversion α_{ab} to this product apartness gives the required witness $\mathsf{NonZero}_F(\mathsf{mul}(a,b))$. □

Theorem 45.312 (Field certified five-factor product nonzero iff factors)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with the hypotheses of Definition 45.305 and the zero-factor exclusion hypotheses used in Definition 45.308, assume an apartness-to-nonzero bridge family: for every* $t : \mathsf{BHist}$*, an instance*

$$\alpha_t : (t\mathsf{hsame}0 \to \bot) \to \mathsf{NonZero}_F(t).$$

Then every $a, b, c, d, e : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),e)) \iff \mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b) \wedge \mathsf{NonZero}_F(c) \wedge \mathsf{NonZero}_F(d) \wedge \mathsf{NonZero}$$

Proof. For the forward direction, assume $p_{abcde} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),e))$. Apply Definition 45.303 to the outer binary product. This gives apartness of the fourfold prefix from 0 and apartness of e from 0.

Convert both apartness witnesses through the bridge family. The left conversion gives $p_{abcd} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d))$, and the right conversion gives $\mathsf{NonZero}_F(e)$.

Apply Definition 45.310 to p_{abcd}. It supplies the four factor witnesses for a, b, c, and d. Pair those witnesses with the converted witness for e in the displayed order.

For the reverse direction, assume witnesses $p_a : \mathsf{NonZero}_F(a)$, $p_b : \mathsf{NonZero}_F(b)$, $p_c : \mathsf{NonZero}_F(c)$, $p_d : \mathsf{NonZero}_F(d)$, and $p_e : \mathsf{NonZero}_F(e)$. Apply Definition 45.310 in the reverse direction to p_a, p_b, p_c, p_d and obtain $p_{abcd} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d))$.

Apply Definition 45.305 to p_{abcd} and p_e. The resulting witness is exactly the certified nonzero witness for the left-nested fivefold product. □

The associated ten-field per-object template is:

1. Traditional target: five-factor product nonzero exactness for a certified $\mathsf{Field}^{\uparrow}$ object.
2. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist together with $\mathsf{Ring}^{\uparrow}$ zero absorption, product nonzero construction, product zero-factor exclusion, and the apartness-to-nonzero bridge family.
3. Carrier: BHist.
4. Classifier: hsame with the derived predicate $\mathsf{NonZero}_F$.
5. Stability obligations: binary product nonzero closure, binary zero-factor exclusion, four-factor exactness, and bridge transport from apartness to nonzero.
6. Ledger: factor witnesses p_a, p_b, p_c, p_d, p_e, prefix witness p_{abcd}, and fivefold product witness p_{abcde}.
7. Core theorem list: Definition 45.312.
8. Exactness predicate: nonzero of the left-nested fivefold product is equivalent to the five displayed factor nonzero witnesses.
9. Downstream consumer: $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ five-factor product equation exactness and the associated phase-exit product rows.
10. Status: $\mathsf{ProofTarget}$.

45.9 Zero-apartness transport through certified products

Theorem 45.313 (Field two-sided multiplication preserves and reflects zero apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.246, let*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot).$$

For any $a, b : \mathsf{BHist}$ *with outer witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x : \mathsf{BHist}$ *satisfies*

$$\mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b)) \iff \mathsf{ApartZero}_F(x).$$

Proof. For the product-to-middle direction, assume $q : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), b))$ and then assume $x\mathsf{hsame}0$. The reverse direction of Definition 45.246, using the outer witnesses p_a and p_b, gives

$$\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsame}0.$$

Applying q to this classifier equality gives $\bot$, so $\mathsf{ApartZero}_F(x)$ follows.

For the middle-to-product direction, assume $r : \mathsf{ApartZero}_F(x)$ and then assume $\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsame}0$. The forward direction of Definition 45.246, with the same outer witnesses, gives $x\mathsf{hsame}0$. Applying r to this classifier equality gives $\bot$, so the certified two-sided product is apart from zero. □

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided product apartzero exact`

Theorem 45.314 (Field two-sided empty-product solution uniqueness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the same additive, multiplicative, congruence, distributivity, and*

inverse hypotheses as Definition 45.246, let $a, b : \mathsf{BHist}$ *carry nonzero witnesses* p_a, p_b*. If two middle histories* x, y *both solve the empty two-sided product equation*

$$\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsamee}, \qquad \mathsf{mul}(\mathsf{mul}(a, y), b)\mathsf{hsamee},$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the two-sided empty-product exactness theorem separately to the two displayed empty-product witnesses. It identifies both x and y with e. Transitivity through the symmetric second comparison gives $x\mathsf{hsame}y$. □

Theorem 45.315 (Field two-sided empty-product fiber independence)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the same additive, multiplicative, congruence, distributivity, and inverse hypotheses as Definition 45.246, let* $a, b, c, d : \mathsf{BHist}$ *carry nonzero witnesses* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*, and* $p_d : \mathsf{NonZero}_F(d)$*. If* $x, y : \mathsf{BHist}$ *satisfy*

$$\mathsf{mul}(\mathsf{mul}(a, x), b)\mathsf{hsamee} \qquad \textit{and} \qquad \mathsf{mul}(\mathsf{mul}(c, y), d)\mathsf{hsamee},$$

then $x\mathsf{hsame}y$*.*

Proof. Apply the two-sided empty-product exactness theorem to the first displayed witness with the certified outer pair (a, b) and witnesses p_a, p_b. This classifies the middle history as $x\mathsf{hsamee}$.

Apply the same exactness theorem to the second displayed witness with the certified outer pair (c, d) and witnesses p_c, p_d. This classifies the second middle history as $y\mathsf{hsamee}$.

Symmetry turns the second comparison into $\mathsf{ehsame}y$. Transitivity through the common empty endpoint gives $x\mathsf{hsame}y$, so the empty-product fiber is independent of the chosen certified outer factors. □

Per-object certificate: Traditional target: two-sided empty-product fiber uniqueness and independence. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist. Carrier choice: BHist. Classifier choice: hsame. Stability obligation: two-sided zero-product exactness. Ledger: the two certified outer-factor pairs and the two empty-product witnesses. Core theorem list: Theorem 45.314 and Theorem 45.315. Exactness predicate: all certified two-sided empty-product fibers classify to the singleton empty endpoint. Status: $\mathsf{ProofTarget}$.

Theorem 45.316 (Field binary product zero-apartness exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.247, including the ring zero-absorption and multiplication-congruence hypotheses used there, define*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot).$$

For any $a, b : \mathsf{BHist}$ *equipped with explicit maps* $\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a)$ *and* $\alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b)$*, one has*

$$\mathsf{ApartZero}_F(\mathsf{mul}(a, b)) \iff \mathsf{ApartZero}_F(a) \wedge \mathsf{ApartZero}_F(b).$$

Proof. For the product-to-left direction, assume $q : \mathsf{ApartZero}_F(\mathsf{mul}(a, b))$ and then assume $a\mathsf{hsame}0$. Multiplication congruence transports this classifier equality through right multiplication by b, giving $\mathsf{mul}(a, b)\mathsf{hsamemul}(0, b)$. Theorem 43.39 classifies $\mathsf{mul}(0, b)$ as 0, so transitivity gives $\mathsf{mul}(a, b)\mathsf{hsame}0$. Applying q gives $\bot$.

For the product-to-right direction, assume again $q : \mathsf{ApartZero}_F(\mathsf{mul}(a, b))$ and then assume $b\mathsf{hsame}0$. Multiplication congruence transports this classifier equality through left multiplication

by a, giving $\mathsf{mul}(a,b)\mathsf{hsamemul}(a,0)$. Theorem 43.39 classifies $\mathsf{mul}(a,0)$ as 0, so transitivity gives $\mathsf{mul}(a,b)\mathsf{hsame0}$. Applying q gives $\bot$.

For the factor-to-product direction, assume $r_a : \mathsf{ApartZero}_F(a)$ and $r_b : \mathsf{ApartZero}_F(b)$. Convert them to nonzero witnesses $p_a := \alpha_a(r_a)$ and $p_b := \alpha_b(r_b)$, and assume $\mathsf{mul}(a,b)\mathsf{hsame0}$.

Apply Definition 45.247 to the zero product. With p_a it returns $b\mathsf{hsame0}$, which r_b sends to contradiction. Symmetrically, the witness p_b would return $a\mathsf{hsame0}$ and close by r_a; the paired factor ledger is therefore stable on both sides. This proves $\mathsf{ApartZero}_F(\mathsf{mul}(a,b))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field binary product apartzero exact`

Per-object certificate: Traditional target: product nonzero exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ + $\mathsf{Ring}^{\uparrow}$. Carrier choice: BHist. Classifier choice: hsame. Ledger: the two factor apartness witnesses. Core theorem this block: Theorem 45.316. Exactness predicate: binary product zero-apartness. Standard bridge: deferred. Status: ProofTarget.

Theorem 45.317 (Field three-factor product zero-apartness exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.316, define*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame0} \to \bot).$$

For any $a, x, b : \mathsf{BHist}$ *equipped with explicit conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_x : \mathsf{ApartZero}_F(x) \to \mathsf{NonZero}_F(x), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b),$$

$$\alpha_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a,x)) \to \mathsf{NonZero}_F(\mathsf{mul}(a,x)), \quad \alpha_{axb} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,x),b)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}($$

one has

$$\mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,x),b)) \iff \mathsf{ApartZero}_F(a) \wedge \mathsf{ApartZero}_F(x) \wedge \mathsf{ApartZero}_F(b).$$

Proof. For the forward direction, assume $q : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,x),b))$. Apply Definition 45.316 first to the binary product $(\mathsf{mul}(a,x),b)$, using the conversion maps α_{ax} and α_b. Its forward implication gives both $\mathsf{ApartZero}_F(\mathsf{mul}(a,x))$ and $\mathsf{ApartZero}_F(b)$.

Apply Definition 45.316 again to the binary product (a,x), using α_a and α_x. The apartness of $\mathsf{mul}(a,x)$ just obtained yields $\mathsf{ApartZero}_F(a)$ and $\mathsf{ApartZero}_F(x)$. Together with $\mathsf{ApartZero}_F(b)$, these are exactly the three factor witnesses on the right-hand side.

For the reverse direction, assume witnesses $r_a : \mathsf{ApartZero}_F(a)$, $r_x : \mathsf{ApartZero}_F(x)$, and $r_b : \mathsf{ApartZero}_F(b)$. Apply Definition 45.316 to (a,x) and use its reverse implication with r_a and r_x to build $r_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a,x))$.

Now apply the same binary exactness theorem to $(\mathsf{mul}(a,x),b)$, using α_{ax} and α_b. Its reverse implication combines r_{ax} with r_b, producing $\mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a,x),b))$. The displayed conversion map α_{axb} records the corresponding full product witness interface for downstream certificates. □

Per-object certificate: Traditional target: threefold product nonzero exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$ zero absorption. Carrier choice: BHist. Classifier choice: hsame. Ledger: the factor apartness witnesses and product conversion maps. Exactness predicate: threefold zero-apartness equivalence. Status: ProofTarget.

Theorem 45.318 (Field four-factor product zero-apartness exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.316, define*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame0} \to \bot).$$

For any $a, x, b, c : \mathsf{BHist}$ *equipped with explicit conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_x : \mathsf{ApartZero}_F(x) \to \mathsf{NonZero}_F(x), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b),$$

$$\alpha_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a, x)) \to \mathsf{NonZero}_F(\mathsf{mul}(a, x)), \quad \alpha_{bc} : \mathsf{ApartZero}_F(\mathsf{mul}(b, c)) \to \mathsf{NonZero}_F(\mathsf{mul}(b, c)),$$

one has

$$\mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c))) \iff \mathsf{ApartZero}_F(a) \wedge \mathsf{ApartZero}_F(x) \wedge \mathsf{ApartZero}_F(b) \wedge \mathsf{ApartZero}_F(c).$$

Proof. For the forward direction, assume $q : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c)))$. Apply Definition 45.316 first to the outer binary product $(\mathsf{mul}(a, x), \mathsf{mul}(b, c))$, using the conversion maps α_{ax} and α_{bc}. Its forward implication gives both $\mathsf{ApartZero}_F(\mathsf{mul}(a, x))$ and $\mathsf{ApartZero}_F(\mathsf{mul}(b, c))$.

Apply the same binary exactness theorem to (a, x) with α_a and α_x. The first pair-product apartness witness yields $\mathsf{ApartZero}_F(a)$ and $\mathsf{ApartZero}_F(x)$.

Apply binary exactness once more to (b, c) with α_b and α_c. The second pair-product apartness witness yields $\mathsf{ApartZero}_F(b)$ and $\mathsf{ApartZero}_F(c)$. These four witnesses are exactly the right-hand side.

For the reverse direction, assume witnesses $r_a : \mathsf{ApartZero}_F(a)$, $r_x : \mathsf{ApartZero}_F(x)$, $r_b : \mathsf{ApartZero}_F(b)$, and $r_c : \mathsf{ApartZero}_F(c)$. Apply Definition 45.316 to (a, x) and use its reverse implication with r_a and r_x to build $r_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a, x))$.

Apply binary exactness to (b, c) in the same way. The witnesses r_b and r_c build $r_{bc} : \mathsf{ApartZero}_F(\mathsf{mul}(b, c))$.

Finally apply binary exactness to the outer product $(\mathsf{mul}(a, x), \mathsf{mul}(b, c))$, using α_{ax} and α_{bc}. Its reverse implication combines r_{ax} and r_{bc} into $\mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c)))$. □

Per-object certificate: Traditional target: fourfold product nonzero exactness. BEDC source: $\mathsf{Field}^{\uparrow} + \mathsf{Ring}^{\uparrow}$. Carrier choice: BHist. Classifier choice: hsame. Ledger: the four factor witnesses plus the pair-product conversion maps for $\mathsf{mul}(a, x)$ and $\mathsf{mul}(b, c)$. Exactness predicate: fourfold zero-apartness equivalence. Status: $\mathsf{ProofTarget}$.

Theorem 45.319 (Field four-factor product inverse reversal from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.316, Definition 45.318, Definition 45.213, and Definition 45.202, including multiplication congruence, associativity, and transitivity for* hsame*, let* $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame0} \to \bot)$*. For any* $a, x, b, c : \mathsf{BHist}$ *equipped with conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_x : \mathsf{ApartZero}_F(x) \to \mathsf{NonZero}_F(x),$$

$$\alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b), \quad \alpha_c : \mathsf{ApartZero}_F(c) \to \mathsf{NonZero}_F(c),$$

$$\alpha_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a, x)) \to \mathsf{NonZero}_F(\mathsf{mul}(a, x)), \quad \alpha_{bc} : \mathsf{ApartZero}_F(\mathsf{mul}(b, c)) \to \mathsf{NonZero}_F(\mathsf{mul}(b, c)),$$

$$\alpha_{axbc} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c))) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c))),$$

any witness

$$q : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c)))$$

determines induced witnesses p_a*,* p_x*,* p_b*,* p_c*,* p_{ax}*,* p_{bc}*, and* p_{axbc} *such that*

$$\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c)), p_{axbc})\mathsf{hsamemul}(\mathsf{mul}(\mathsf{inv}(c, p_c), \mathsf{inv}(b, p_b)), \mathsf{mul}(\mathsf{inv}(x, p_x), \mathsf{inv}(a, p_a))).$$

Proof. Apply Definition 45.318 to q. The forward implication yields apartness witnesses $q_a : \mathsf{ApartZero}_F(a)$, $q_x : \mathsf{ApartZero}_F(x)$, $q_b : \mathsf{ApartZero}_F(b)$, and $q_c : \mathsf{ApartZero}_F(c)$. Set $p_a := \alpha_a(q_a)$, $p_x := \alpha_x(q_x)$, $p_b := \alpha_b(q_b)$, and $p_c := \alpha_c(q_c)$.

Use the reverse implication of Definition 45.316 on (a, x) to build $q_{ax} : \mathsf{ApartZero}_F(\mathsf{mul}(a, x))$ from q_a and q_x. Use it again on (b, c) to build $q_{bc} : \mathsf{ApartZero}_F(\mathsf{mul}(b, c))$ from q_b and q_c. Set $p_{ax} := \alpha_{ax}(q_{ax})$, $p_{bc} := \alpha_{bc}(q_{bc})$, and $p_{axbc} := \alpha_{axbc}(q)$.

Apply Definition 45.213 to the outer product with witnesses p_{axbc}, p_{ax}, and p_{bc}. Apply the same theorem to (a, x) with p_{ax}, p_a, and p_x, and to (b, c) with p_{bc}, p_b, and p_c.

Multiplication congruence transports the two pair-product inverse equalities into the right-hand side of the outer equality. The comparison of inverse proof indices is supplied by Definition 45.202, and associativity together with transitivity of hsame gives the displayed $(c, b)(x, a)$ reversed product. □

45.10 Field square apartness inverse descent

Theorem 45.320 (Field square apartness inverse descent)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.303 and Definition 45.213, plus the canonical product-apartness witness construction of Definition 45.271, together with the proof-index comparison supplied by Definition 45.202, let* $\mathsf{ApartZero}_F(h)$ *mean* $h\mathsf{hsame}0 \to \bot$. *For every* $a : \mathsf{BHist}$, *assume explicit conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_{aa} : \mathsf{ApartZero}_F(\mathsf{mul}(a, a)) \to \mathsf{NonZero}_F(\mathsf{mul}(a, a)).$$

Any witness

$$p_{a^2} : \mathsf{ApartZero}_F(\mathsf{mul}(a, a))$$

canonically induces a witness $p_a : \mathsf{ApartZero}_F(a)$. *With this induced apartness witness and the displayed conversions, the proof-indexed inverse satisfies*

$$\mathsf{inv}(\mathsf{mul}(a, a), \alpha_{aa}(p_{a^2}))\mathsf{hsamemul}(\mathsf{inv}(a, \alpha_a(p_a)), \mathsf{inv}(a, \alpha_a(p_a))).$$

Proof. Construct p_a as a function from an assumed classifier $a\mathsf{hsame}0$ to contradiction. Multiplication congruence sends $\mathsf{mul}(a, a)$ to $\mathsf{mul}(0, a)$, and ring zero absorption classifies $\mathsf{mul}(0, a)$ as 0. Transitivity gives $\mathsf{mul}(a, a)\mathsf{hsame}0$.

Apply the square apartness witness p_{a^2} to this classifier equality. This closes the assumed zero classifier for a, so the function just built is the induced witness $p_a : \mathsf{ApartZero}_F(a)$.

Next apply Definition 45.271 to the two factor witnesses p_a and p_a, using α_a for both factor indices and α_{aa} for the product index. This supplies the canonical product apartness witness

$$p'_{a^2} : \mathsf{ApartZero}_F(\mathsf{mul}(a, a))$$

and the indexed inverse equality

$$\mathsf{inv}(\mathsf{mul}(a, a), \alpha_{aa}(p'_{a^2}))\mathsf{hsamemul}(\mathsf{inv}(a, \alpha_a(p_a)), \mathsf{inv}(a, \alpha_a(p_a))).$$

It remains to compare the supplied square witness with the canonical product witness. Use Definition 45.202 with reflexivity on $\mathsf{mul}(a, a)$, the witness p_{a^2} on the source side, and p'_{a^2} on the target side. This identifies

$$\mathsf{inv}(\mathsf{mul}(a, a), \alpha_{aa}(p_{a^2}))\mathsf{hsameinv}(\mathsf{mul}(a, a), \alpha_{aa}(p'_{a^2})).$$

Transitivity with the canonical product equality gives the displayed inverse descent identity. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square apartness inverse descent`

Theorem 45.321 (Field square nonzero gives one-sided multiplication exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.303, Definition 45.229, and Definition 45.224, let* $\mathsf{ApartZero}_F(h)$ *mean* $h\mathsf{hsame}0 \to \bot$*. For every* $a : \mathsf{BHist}$*, assume an explicit conversion map*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a).$$

For every witness

$$p_{a^2} : \mathsf{NonZero}_F(\mathsf{mul}(a, a)),$$

let $r_a : \mathsf{ApartZero}_F(a)$ *be the first zero-factor obstruction induced by zero-factor exclusion, and set* $p_a := \alpha_a(r_a)$*. Then every* $c, x : \mathsf{BHist}$ *satisfies both one-sided multiplication exactness clauses:*

$$\mathsf{mul}(a, x)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), c)$$

and

$$\mathsf{mul}(x, a)\mathsf{hsame}c \iff x\mathsf{hsamemul}(c, \mathsf{inv}(a, p_a)).$$

Proof. Apply Definition 45.303 to p_{a^2} with both product entries instantiated by a. The first factor clause gives $r_a : \mathsf{ApartZero}_F(a)$.

Convert this apartness witness by the displayed map, setting $p_a := \alpha_a(r_a) : \mathsf{NonZero}_F(a)$. This is the proof index available to the inverse and cancellation laws for the factor a.

For the left multiplication clause, invoke Definition 45.229 with this induced witness p_a. Its displayed equivalence is exactly $\mathsf{mul}(a, x)\mathsf{hsame}c$ if and only if $x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), c)$.

For the right multiplication clause, invoke Definition 45.224 with the same induced witness p_a. Its displayed equivalence is exactly $\mathsf{mul}(x, a)\mathsf{hsame}c$ if and only if $x\mathsf{hsamemul}(c, \mathsf{inv}(a, p_a))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square nonzero one sided mul exact`

Theorem 45.322 (Field square nonzero gives two-sided same-factor multiplication exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.303 and Definition 45.233, let* $\mathsf{ApartZero}_F(h)$ *mean* $h\mathsf{hsame}0 \to \bot$*. For every* $a : \mathsf{BHist}$*, assume an explicit conversion map*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a).$$

For every $c, x : \mathsf{BHist}$ *and witness*

$$p_{a^2} : \mathsf{NonZero}_F(\mathsf{mul}(a, a)),$$

let $r_a : \mathsf{ApartZero}_F(a)$ *be the first zero-factor obstruction induced by zero-factor exclusion, and set* $p_a := \alpha_a(r_a)$*. Then the square-scaled equation with the same certified factor on both sides satisfies*

$$\mathsf{mul}(\mathsf{mul}(a, x), a)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), \mathsf{mul}(c, \mathsf{inv}(a, p_a))).$$

Proof. Apply Definition 45.303 to p_{a^2} with both product entries instantiated by a. Its first zero-factor obstruction gives $r_a : \mathsf{ApartZero}_F(a)$.

Convert this obstruction by the displayed map, setting $p_a := \alpha_a(r_a) : \mathsf{NonZero}_F(a)$. This single proof index is shared by the left and right occurrences of the same multiplier a.

Instantiate Definition 45.233 with both outer factors equal to a and with both outer nonzero witnesses equal to p_a. The resulting middle multiplication exactness equivalence is exactly the displayed square-scaled equation solver. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square nonzero two sided mul exact`

Per-object certificate: Traditional target: square inverse descent. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: zero absorption, multiplication congruence, apartness transport, and proof-indexed inverse laws. Ledger: the square apartness witness and induced factor witness. Core theorem list entry: Theorem 45.320 and Theorem 45.322. Exactness predicate: square apartness descends to the factor inverse and square nonzero supplies the same-factor two-sided equation solver. Standard bridge: deferred conservative bridge. Status: ProofTarget.

45.11 Concrete Rat denominator continuation instance

Definition 45.323 (Concrete Rat denominator continuation instance)**.** *The concrete denominator-side continuation instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *has carrier, classifier, operation, continuation witness, and ledger predicate*

$$C_D(h) \iff \mathsf{RatHistoryCarrier}(h),$$

$$R_D(h,k) \iff \mathsf{RatHistoryClassifier}(h,k),$$

$$\mu_D(h,k) := \mathsf{append}(h,k),$$

with the canonical continuation witness $\mathsf{Cont}(h,k,\mu_D(h,k))$ *and ledger predicate* RatHistoryLedgerPolicy*. The empty history* e *is an external classifier-unit endpoint for the operation and is not a carried denominator.*

Theorem 45.324 (Rat denominator append concrete instance laws)**.** *For histories* $d, d', e, e', f, \rho, v :$ BHist*, the concrete function data of Definition 45.323, with* $\mu_D(h,k) := \mathsf{append}(h,k)$*, satisfy the following append laws.*

1. *If* $C_D(d)$ *and* $C_D(e)$*, then* $C_D(\mu_D(d,e))$*.*
2. *If* $C_D(d)$ *and* $C_D(e)$*, then* $R_D(\mu_D(d,e), \mu_D(e,d))$*.*
3. *If* $C_D(d)$*,* $C_D(e)$*, and* $C_D(f)$*, then* $R_D(\mu_D(\mu_D(d,e),f), \mu_D(d, \mu_D(e,f)))$*.*
4. *If* $R_D(d,d')$ *and* $R_D(e,e')$*, then* $R_D(\mu_D(d,e), \mu_D(d',e'))$*.*
5. *If* $\mathsf{RatHistoryLedgerPolicy}(\rho,v)$*,* $C_D(e)$*, and* $R_D(e,e')$*, then* $\mathsf{RatHistoryLedgerPolicy}(\mu_D(\rho,e), \mu_D(v,e'))$*.*
6. *No carried history* $u : \mathsf{BHist}$ *can satisfy either one-sided classifier-unit law*

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x,u), x)$$

or

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(u,x), x).$$

Proof. Unfold C_D, R_D, and μ_D. The operation is ordinary history append, so the first five clauses are the concrete append package proved for RatHistoryCarrier, RatHistoryClassifier, and RatHistoryLedgerPolicy.

Carrier closure uses the carried denominator closure of append. The commutative classifier clause uses the unary append commutativity theorem inside the two carried histories.

Associativity compares the two append bracketings by append associativity and then repacks the result into R_D. Binary classifier congruence and ledger closure are the corresponding Rat denominator append-closure clauses for the classifier and ledger predicates.

For a one-sided carried unit, pass from the displayed append law to its canonical continuation instance using $\mathsf{Cont}(x, u, \mu_D(x, u))$ or $\mathsf{Cont}(u, x, \mu_D(u, x))$. Then Definition 45.125 forces $u\mathsf{hsamee}$, while Definition 37.18 rules out a carried empty denominator. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator append concrete instance laws`

Theorem 45.325 (Rat denominator continuation concrete instance laws)**.** *For histories $d, d', e, e', f, \rho, v, de, ed, ef$, BHist, the relational continuation data of Definition 45.323 satisfy the following laws.*

1. *If $C_D(d)$, $C_D(e)$, and $\mathsf{Cont}(d, e, de)$, then $C_D(de)$.*
2. *If $C_D(d)$, $C_D(e)$, $\mathsf{Cont}(d, e, de)$, and $\mathsf{Cont}(e, d, ed)$, then $R_D(de, ed)$.*
3. *If $C_D(d)$, $C_D(e)$, $C_D(f)$, $\mathsf{Cont}(d, e, de)$, $\mathsf{Cont}(e, f, ef)$, $\mathsf{Cont}(de, f, left)$, and $\mathsf{Cont}(d, ef, right)$, then $R_D(left, right)$.*
4. *If $R_D(d, d')$, $R_D(e, e')$, $\mathsf{Cont}(d, e, de)$, and $\mathsf{Cont}(d', e', dep)$, then $R_D(de, dep)$.*
5. *If $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$, $C_D(e)$, $R_D(e, e')$, $\mathsf{Cont}(\rho, e, rhoe)$, and $\mathsf{Cont}(v, e', vep)$, then $\mathsf{RatHistoryLedgerPolicy}(rhoe, vep)$.*
6. *No carried history $u : \mathsf{BHist}$ can satisfy either one-sided classifier-unit law*

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x, u, r) \to R_D(r, x)$$

or

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(u, x, r) \to R_D(r, x).$$

Proof. Unfold C_D and R_D. The relational presentation leaves each output history named by its Cont witness instead of replacing it by append at the statement level.

Carrier closure is Definition 45.113. The commutative classifier clause is Definition 45.118 applied to the two displayed continuations.

Associativity uses Definition 45.117 on the four continuation witnesses. The theorem supplies the carried intermediate results and the history-sameness comparison needed to read the two bracketings as classifier-equivalent.

For binary classifier congruence, unpack $R_D(d, d')$ and $R_D(e, e')$. The right input classifier supplies the matching denominator comparison required by Definition 45.119, and the two displayed continuations give $R_D(de, dep)$.

The ledger clause applies Definition 45.122 to $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$, the carried and classified right tail, and the two displayed continuations.

Finally, Definition 45.125 says that any one-sided classifier-unit endpoint is hsame to e, and its carried endpoint clause rules out such a history inside $\mathsf{RatHistoryCarrier}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation concrete instance laws`

Theorem 45.326 (Rat denominator append-continuation endpoint bridge)**.** *For the concrete data of Definition 45.323, if $C_D(h)$ and $C_D(k)$, then every history $r : \mathsf{BHist}$ satisfies the endpoint classifier bridge*

$$\mathsf{Cont}(h,k,r) \iff R_D(r,\mu_D(h,k)).$$

Thus the canonical append endpoint $\mu_D(h,k)$ and any displayed continuation endpoint r are identified exactly at the denominator classifier surface.

Proof. Unfold C_D, R_D, and μ_D only to read them as RatHistoryCarrier, RatHistoryClassifier, and append. With the two carrier hypotheses in place, Definition 45.114 gives, for every history r, exactly $\mathsf{Cont}(h,k,r) \iff R_D(r,\mu_D(h,k))$.

At the canonical endpoint, the same bridge reads the canonical continuation witness as the reflexive denominator classifier comparison $R_D(\mu_D(h,k),\mu_D(h,k))$. □

Theorem 45.327 (Rat denominator append-relational unit-law equivalence)**.** *For the concrete data of Definition 45.323, every history $u : \mathsf{BHist}$ satisfying $C_D(u)$ has the following two one-sided unit-law equivalences:*

$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x,u),x)) \iff (\forall x,r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x,u,r) \to R_D(r,x))$$

and

$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(u,x),x)) \iff (\forall x,r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(u,x,r) \to R_D(r,x))\,.$$

Thus the functional append presentation and the relational continuation presentation carry exactly the same one-sided internal-unit obstruction for a carried denominator endpoint.

Proof. Fix $u : \mathsf{BHist}$ with $C_D(u)$.

For the right-unit forward direction, assume the append family $\forall x.\ C_D(x) \to R_D(\mu_D(x,u),x)$. Given x, r, $C_D(x)$, and $\mathsf{Cont}(x,u,r)$, apply Definition 45.326 to the carried pair (x,u). This classifies the displayed endpoint r with the canonical append endpoint $\mu_D(x,u)$. The assumed append family classifies $\mu_D(x,u)$ with x, and classifier transitivity gives $R_D(r,x)$.

For the right-unit reverse direction, assume the relational family $\forall x,r.\ C_D(x) \to \mathsf{Cont}(x,u,r) \to R_D(r,x)$. At each carried x, use the canonical continuation witness $\mathsf{Cont}(x,u,\mu_D(x,u))$ from Definition 45.323. Applying the relational family to this witness gives $R_D(\mu_D(x,u),x)$.

The left-unit equivalence is identical with the inputs reversed. The forward direction compares any displayed $\mathsf{Cont}(u,x,r)$ endpoint to the canonical append endpoint $\mu_D(u,x)$ by Definition 45.326, then composes with the assumed append left-unit row by classifier transitivity. The reverse direction applies the relational left-unit family to the canonical witness $\mathsf{Cont}(u,x,\mu_D(u,x))$. □

This theorem fits the ten-field per-object template as follows: traditional target append/Cont unit-law equivalence for strict denominator multiplication support; BEDC source BHist together with Cont, RatHistoryCarrier, and RatHistoryClassifier; carrier C_D; classifier R_D; stability obligation endpoint bridge plus classifier transitivity; ledger RatHistoryLedgerPolicy; core theorem Definition 45.327; exactness the displayed iff pair; standard bridge strict denominator semigroup interface; status ProofTarget.

For the ten-field exit certificate of Definition 45.585, this theorem occupies only the denominator endpoint-identification part of the classifier and stability rows. The traditional target, BEDC source, carrier, ledger, core theorem list, exactness predicate, standard bridge, and status rows are supplied by their named concrete certificates, while the append and relational denominator laws remain the checked packages of Definition 45.324 and Definition 45.325.

Definition 45.328 (Concrete Rat denominator strict semigroup interface)**.** *The denominator multiplication interface carried by the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *continuation data is the triple*

$$C_D(h) := \mathsf{RatHistoryCarrier}(h),$$

$$R_D(h,k) := \mathsf{RatHistoryClassifier}(h,k),$$

and

$$\mu_D(h,k) := \mathsf{append}(h,k).$$

It is a strict denominator semigroup interface: the operation is the canonical append endpoint, the classifier is the Rat denominator history classifier, and the carried endpoints are exactly the Rat denominator histories. The interface has no carried unit endpoint. The empty history is only an external classifier endpoint, and no $u : \mathsf{BHist}$ *with* $C_D(u)$ *can act as a one-sided unit for the carried denominator multiplication.*

Theorem 45.329 (Rat denominator carried unit obstruction package)**.** *For the concrete strict denominator semigroup interface of Definition 45.328, every* $u : \mathsf{BHist}$ *with* $C_D(u)$ *satisfies the following four impossibility rows:*

1. *the functional right-unit family*

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x,u),x)$$

implies False*;*

2. *the functional left-unit family*

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(u,x),x)$$

implies False*;*

3. *the relational right-unit family*

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x,u,r) \to R_D(r,x)$$

implies False*;*

4. *the relational left-unit family*

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(u,x,r) \to R_D(r,x)$$

implies False*.*

The ten-field template represented by this certificate is: traditional target strict denominator semigroup support; BEDC source BHist *with* Cont*,* $\mathsf{RatHistoryCarrier}$*, and* $\mathsf{RatHistoryClassifier}$*; carrier* C_D*; classifier* R_D*; stability append/*Cont *endpoint bridge plus classifier transitivity; ledger* $\mathsf{RatHistoryLedgerPolicy}$*; core theorem this package; exactness the four impossibility rows; standard bridge strict non-unital denominator multiplication; status* $\mathsf{ProofTarget}$*.*

Proof. Fix $u : \mathsf{BHist}$ with $C_D(u)$.

For the two functional rows, clause 6 of Definition 45.324 states that a carried history cannot satisfy either one-sided append classifier-unit law. Applying that clause to u and the displayed functional right-unit or left-unit family yields False.

For the two relational rows, clause 6 of Definition 45.325 gives the same contradiction for a carried history satisfying either one-sided Cont-based classifier-unit law. Applying it to the displayed relational right-unit or left-unit family yields False.

Finally, Definition 45.327 records that the append presentation and the relational continuation presentation express the same one-sided carried-unit boundary. The right-unit equivalence links the first and third rows, and the left-unit equivalence links the second and fourth rows, so the four contradictions are one certificate-level obstruction for the concrete denominator interface. □

45.12 Rat denominator append-continuation semigroup ledger bridge

Theorem 45.330 (Rat denominator append-continuation strict semigroup non-monoid bridge)**.** *For the concrete data of Definition 45.323, the append presentation* (C_D, R_D, μ_D) *and the relational* Cont *presentation determine the same strict commutative semigroup-with-ledger over* $\mathsf{RatHistoryCarrier}$*, not an internal carried monoid. The common structure has the following form.*

1. *The carrier is* $C_D(h)$*, equivalently* $\mathsf{RatHistoryCarrier}(h)$*, and the classifier is* $R_D(h, k)$*, equivalently* $\mathsf{RatHistoryClassifier}(h, k)$*.*
2. *The append presentation supplies carrier closure, commutative classifier, associativity up to* R_D*, binary classifier congruence, and ledger closure for* $\mathsf{RatHistoryLedgerPolicy}$ *under right carried tails.*
3. *The relational* Cont *presentation supplies the same five rows with every product endpoint named by a continuation witness.*
4. *The endpoint bridge identifies every relational product endpoint with the corresponding append endpoint at the* R_D *classifier surface, so the five append rows and the five relational rows transport into one another.*
5. *For every carried history* $u : \mathsf{BHist}$ *with* $C_D(u)$*, neither presentation admits a one-sided classifier unit. In the append presentation, neither*

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x, u), x)$$

nor

$$\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(u, x), x)$$

can hold. In the relational presentation, neither

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x, u, r) \to R_D(r, x)$$

nor

$$\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(u, x, r) \to R_D(r, x)$$

can hold.

Proof. The append rows are exactly the first five clauses of Definition 45.324: closure of C_D, commutativity at the classifier, associativity at the classifier, binary classifier congruence, and ledger closure for RatHistoryLedgerPolicy.

The relational rows are the matching five clauses of Definition 45.325. They use the same carrier, classifier, and ledger predicate, but leave each product endpoint as a named Cont result.

The identification of the two presentations is Definition 45.326. Its endpoint bridge transports each append output to the corresponding continuation endpoint and transports each continuation endpoint back to the append output at the R_D classifier surface.

It remains to record why the common object is strict. The one-sided unit clauses in the append and relational packages both feed into Definition 45.125, which forces any such endpoint to be hsamee. Definition 37.18 then excludes this endpoint from RatHistoryCarrier. Hence no carried denominator supplies an internal one-sided classifier unit, so the shared denominator structure is a strict semigroup-with-ledger rather than a carried monoid. □

Per-object template: traditional target strict denominator multiplication support for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, Cont, RatHistoryCarrier, RatHistoryClassifier, and RatHistoryLedgerPolicy; carrier choice: C_D; classifier choice: R_D; stability obligations: carrier closure, commutative classifier, associativity, binary classifier congruence, ledger closure, and carried unit obstruction; ledger: denominator multiplication provenance through RatHistoryLedgerPolicy; core theorem: Definition 45.330; exactness predicate: append and Cont endpoint bridge; standard bridge: deferred; status ProofTarget.

Theorem 45.331 (Rat denominator append-continuation final-exit obstruction guard)**.** *For the concrete data of Definition 45.323 and the final row inventory package of Theorem 45.773, the Phase Algebra Hierarchy consumer receives both the final selector-row inventory and the strict denominator guard. Explicitly, for every carried denominator* $u : \mathsf{BHist}$ *with* $C_D(u)$*, the append presentation gives both*

$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x, u), x)) \to \mathsf{False}$$

and

$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(u, x), x)) \to \mathsf{False}.$$

The relational presentation gives both

$$(\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x, u, r) \to R_D(r, x)) \to \mathsf{False}$$

and

$$(\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(u, x, r) \to R_D(r, x)) \to \mathsf{False}.$$

Thus the final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *row inventory and the denominator obstruction arrive as one consumable guard: later algebra may use the selector exit rows, but may not read the denominator semigroup as a carried multiplication with a one-sided internal classifier unit.*

Proof. The positive inventory is supplied by Theorem 45.773. It packages the selector inverse-domain coverage row, the selector row-stability row, the complete selector obstruction certificate, and the selector-chain boundary rows for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

The strict denominator rows are supplied by Theorem 45.330. Its append half gives closure, commutative classifier, associativity, congruence, ledger closure, and the two displayed denials of one-sided append classifier units.

For the relational half, use the same bridge theorem together with the no-one-sided-unit rows formalized by BEDC.Derived.FieldUp.field_rat_denominator_continuation_commutative_semigroup_obstruction_pa Those rows deny both carried relational laws with endpoints named by Cont.

Combining these two cited packages gives the advertised guard. The Phase Algebra Hierarchy consumer may carry the final selector-row inventory forward, while the denominator side remains a strict commutative semigroup-with-ledger and not a carried monoid. □

Per-object template: traditional target final-exit denominator obstruction guard for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, Cont, RatHistoryCarrier, RatHistoryClassifier, RatHistoryLedgerPolicy, and the final selector rows; carrier choice: C_D; classifier choice: R_D; stability obligations: final selector-row coverage plus strict append and relational carried-unit obstruction; ledger: denominator multiplication provenance through RatHistoryLedgerPolicy; core theorem: Definition 45.331; exactness predicate: shared append and Cont endpoint bridge with one-sided-unit denial; standard bridge: Phase Algebra Hierarchy consumer for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; status ProofTarget.

Lean4 ✓ `BEDC.Derived.FieldUp.field two sided product apartzero exact from apartness`

Lean4 ✓ `BEDC.Derived.FieldUp.field binary product apartzero exact from apartness`

Theorem 45.332 (Field square zero rejects nonzero factor)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the hypotheses of Definition 45.247, together with classifier transport of* $\mathsf{NonZero}_F$ *and contradiction of* $\mathsf{NonZero}_F(0)$*, every* $a : \mathsf{BHist}$ *satisfying*

$$\mathsf{mul}(a, a)\mathsf{hsame0}$$

has $\mathsf{NonZero}_F(a) \to \bot$*. Moreover, for*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame0} \to \bot),$$

the same displayed square-zero classifier refutes $\mathsf{ApartZero}_F(\mathsf{mul}(a, a))$*.*

Proof. Assume $z : \mathsf{mul}(a, a)\mathsf{hsame0}$ and $p_a : \mathsf{NonZero}_F(a)$. Apply Definition 45.247 to z with the left-factor witness p_a. The left-factor clause returns $a\mathsf{hsame0}$.

Transport p_a along this classifier equality. The classifier-transport stability for $\mathsf{NonZero}_F$ gives a witness of $\mathsf{NonZero}_F(0)$.

The field certificate supplies contradiction of $\mathsf{NonZero}_F(0)$, so the transported witness closes the nonzero-factor clause with $\bot$.

For the apartness clause, take $q : \mathsf{ApartZero}_F(\mathsf{mul}(a, a))$. Since $\mathsf{ApartZero}_F(h)$ means $h\mathsf{hsame0} \to \bot$, direct application of q to the displayed square-zero classifier z gives $\bot$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square zero rejects factor nonzero`

Theorem 45.333 (Field square nonzero iff factor nonzero)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.303 and Definition 45.305, including a constructive bridge family*

$$\alpha_t : (t\mathsf{hsame0} \to \bot) \to \mathsf{NonZero}_F(t),$$

every $a : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(a, a)) \iff \mathsf{NonZero}_F(a).$$

Proof. For the forward direction, assume $p_{aa} : \mathsf{NonZero}_F(\mathsf{mul}(a, a))$. Apply Definition 45.303 to the square product with both factors equal to a. The left-factor clause gives $a\mathsf{hsame}0 \to \bot$.

Apply the bridge-family instance α_a to this apartness witness. The result is $\mathsf{NonZero}_F(a)$.

For the reverse direction, assume $p_a : \mathsf{NonZero}_F(a)$. Apply Definition 45.305 to the same witness p_a in both factor slots. The resulting product witness is exactly $\mathsf{NonZero}_F(\mathsf{mul}(a, a))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square nonzero iff factor nonzero`

Per-object certificate: Traditional target: square inverse descent. BEDC source: $\mathsf{Field}^{\uparrow}$ plus $\mathsf{Ring}^{\uparrow}$. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: zero absorption, multiplication congruence, apartness transport, and proof-indexed inverse laws. Ledger: the square apartness witness, induced factor witness, and square-zero classifier. Core theorem list entries: Theorem 45.320 and Theorem 45.332, together with Theorem 45.333. Exactness predicate: square apartness descends to the factor inverse, square-zero classification rejects nonzero factors, and

$$\mathsf{NonZero}_F(\mathsf{mul}(a, a)) \iff \mathsf{NonZero}_F(a).$$

Standard bridge: deferred conservative bridge. Status: $\mathsf{ProofTarget}$.

45.13 Field square apartness inverse descent

Theorem 45.334 (Field square-certified two-sided affine exactness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.321, Definition 45.277, and Definition 45.278, let*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot).$$

For $a, b : \mathsf{BHist}$*, assume explicit conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b).$$

With square witnesses

$$p_{a^2} : \mathsf{NonZero}_F(\mathsf{mul}(a, a)), \quad p_{b^2} : \mathsf{NonZero}_F(\mathsf{mul}(b, b)),$$

let $r_a : \mathsf{ApartZero}_F(a)$ *and* $r_b : \mathsf{ApartZero}_F(b)$ *be the first zero-factor obstructions induced by zero-factor exclusion, and set* $p_a := \alpha_a(r_a)$ *and* $p_b := \alpha_b(r_b)$*. Then for every* $c, d, x, y : \mathsf{BHist}$ *both clauses hold:*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c \iff x\mathsf{hsamemul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b)))$$

and

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(a, y), b), d) \iff x\mathsf{hsame}y.$$

Proof. Apply Definition 45.303 to p_{a^2} with both product entries instantiated by a. As in Definition 45.321, the first factor clause gives $r_a : \mathsf{ApartZero}_F(a)$. Convert it by the displayed map, setting $p_a := \alpha_a(r_a) : \mathsf{NonZero}_F(a)$.

Apply the same zero-factor exclusion theorem to p_{b^2} with both product entries instantiated by b. The first factor clause gives $r_b : \mathsf{ApartZero}_F(b)$. Convert it by the displayed map, setting $p_b := \alpha_b(r_b) : \mathsf{NonZero}_F(b)$.

With these two induced witnesses, invoke Definition 45.277. Its equivalence is exactly the stated solution formula for $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c$.

For the classifier exactness clause, invoke Definition 45.278 with the same witnesses p_a and p_b and the same offset d. This removes the common certified affine map and yields precisely $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(a, y), b), d)$ if and only if $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field square certified two sided affine exact`

The per-object certificate records traditional target affine equation uniqueness under square-certified factors; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations square nonzero descent and affine classifier exactness; ledger the two square witnesses, conversions, offset, and target; core theorem Definition 45.335; exactness predicate same-target solution uniqueness; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.335 (Field square-certified affine equation solutions are unique)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying the hypotheses of Definition 45.303 and Definition 45.278, let*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot).$$

For $a, b : \mathsf{BHist}$*, assume explicit conversion maps*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b).$$

With square witnesses

$$p_{a^2} : \mathsf{NonZero}_F(\mathsf{mul}(a, a)), \quad p_{b^2} : \mathsf{NonZero}_F(\mathsf{mul}(b, b)),$$

for every $c, d, x, y : \mathsf{BHist}$*, if*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c$$

and

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, y), b), d)\mathsf{hsame}c,$$

then $x\mathsf{hsame}y$*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field square certified affine solution unique`

Proof. Apply Definition 45.303 to p_{a^2} with both product entries instantiated by a. The first zero-factor obstruction gives $r_a : \mathsf{ApartZero}_F(a)$, and the conversion map gives $p_a := \alpha_a(r_a) : \mathsf{NonZero}_F(a)$.

Apply the same descent to p_{b^2} with both product entries instantiated by b. This gives $r_b : \mathsf{ApartZero}_F(b)$ and hence $p_b := \alpha_b(r_b) : \mathsf{NonZero}_F(b)$.

Let the two displayed hypotheses be e_x and e_y. By symmetry of hsame applied to e_y and transitivity of hsame with e_x, obtain

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, y), b), d).$$

Invoke Definition 45.278 with the certified factors p_a and p_b and the same offset d. The classifier exactness theorem cancels the common two-sided affine map, yielding $x\mathsf{hsame}y$. □

Definition 45.336 (Field ledger policy)**.** *Record the ring ledger and the* $\mathsf{NonZero}_F$ *witness used for each inverse.*

Definition 45.337 ($\mathsf{Field}^{\uparrow}$ certificate obligations)**.** *For the concrete target* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$*, a certificate must provide the following rows as a single obligation record.*

1. *Source and carrier: the rational-history source on* BHist *with carrier* $\mathsf{RatHistoryCarrier}$*.*
2. *Classifier and pattern:* $\mathsf{RatHistoryClassifier}$ *and the continuation pattern* Cont *over* BHist*.*
3. *Ring base: the* $\mathsf{CommRing}^{\uparrow}$ *additive, multiplicative, commutativity, distributivity, operation-congruence, and ring-ledger rows.*

4. *Apartness and inverse data:* $\mathsf{NonZero}_F$ *apartness from zero, transport of nonzero evidence, inverse closure, inverse classifier exactness, and left and right inverse laws indexed by nonzero witnesses.*
5. *Product data: the two-sided zero-product rows and the certified nonzero-product contradiction rows.*
6. *Denominator tuple:* (RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, RatHistoryCarrier) *with its multiplicative, unit, closure, classifier, and ledger rows.*
7. *Contextual denominator data: the action* Γ*, its envelope and normalized-support rows, and the context-pair monoid rows for pair product, unit-pair action, composition, and strict-support readback.*
8. *Affine source:* AffAut_F *with the packaged affine* $\mathsf{Group}^{\uparrow}$ *rows used by the field certificate.*
9. *Ledger and exactness data: ring expression ledger rows, inverse witness ledger rows, denominator ledger rows, contextual-pair ledger rows, affine coefficient ledger rows, coverage, soundness, completeness, exact zero classification, denominator exactness, contextual-action exactness, context-pair exactness, and affine classifier exactness.*
10. *Status: the record status field, instantiated here as* ProofTarget.

Lean4 (definition) `BEDC.Derived.FieldUp.FieldCertificateObligations`

Theorem 45.338 (Field certificate obligations expose Rat denominator rows)**.** *Every* FieldCertificateObligations *record projects the Rat carrier row* RatHistoryCarrier(carrier)*, the Rat classifier row* RatHistoryClassifier(carrier, classifi *the denominator row* RatDenomUnitCarrier(denominatorTuple)*, and the contextual denominator row* RatDenomUnitClassifier(denominatorTuple, contextualData)*. It also projects the four source, base, affine, and status assembly equations*

$$\begin{aligned}\mathsf{hsame}(\mathsf{source}, \mathsf{append}(\mathsf{carrier}, \mathsf{classifier})),\\ \mathsf{hsame}(\mathsf{ringBase}, \mathsf{append}(\mathsf{productData}, \mathsf{inverseData})),\\ \mathsf{hsame}(\mathsf{affineSource}, \mathsf{append}(\mathsf{contextualData}, \mathsf{ledgerExactness})),\\ \mathsf{hsame}(\mathsf{status}, \mathsf{append}(\mathsf{source}, \mathsf{affineSource})).\end{aligned}$$

Proof. Unfold the certificate-obligation record. Its surface is the nested conjunction whose first four fields are exactly the Rat carrier, Rat classifier, denominator carrier, and contextual denominator classifier rows.

The remaining conjunctive fields are the four displayed assembly equations. Projecting the nested conjunction in order gives all eight rows without adding any extra carrier, classifier, or ledger hypothesis. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldCertificateObligations rat denominator rows`

Per-object certificate record.

1. Traditional target: constructive field.
2. BEDC source: $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ data.
3. Carrier choice: the $\mathsf{Rat}^{\uparrow}$ carrier with $\mathsf{NonZero}_F(a)$ derived from apartness from 0.
4. Classifier choice: the rational classifier plus $\mathsf{NonZero}_F$ transport.

5. Stability obligations: commutative-ring laws, contradiction of $\mathsf{NonZero}_F(0)$, nonzero transport, inverse closure, and inverse laws indexed by nonzero witnesses.
6. Ledger: ring expression tree plus the $\mathsf{NonZero}_F$ witness for each inverse.
7. Core theorem list: inverse closure, left inverse, right inverse, nonzero stability, inverse congruence from apartness transport, inverse involution under explicit nonzero evidence, right-multiplication equation solving, right-multiplication equation exactness, left-multiplication equation solving, left-multiplication equation exactness, middle multiplication equation solving, middle multiplication equation exactness, two-sided multiplication exactness, two-sided zero-product exactness, zero-product exclusion from two nonzero witnesses, signed-factor apartness exclusivity, affine left equation exactness, affine right equation exactness, two-sided affine equation exactness, one-sided affine map classifier exactness, two-sided affine map classifier exactness, affine automorphism package, product-witness independence, denominator row laws, denominator contextual action certificate, Definition 45.470, Definition 45.475, Definition 45.473, Definition 45.478, Definition 45.487, Definition 45.568, Definition 45.570, Definition 45.490, Definition 45.524, Definition 45.558, and distributivity inherited from $\mathsf{CommRing}^{\uparrow}$.
8. Exactness predicate: coverage, soundness, completeness, exact zero classification, denominator row exactness, contextual denominator action exactness, normalized contextual denominator action exactness through Definition 45.475, no-extra-support contextual-action support neutrality through Definition 45.478, denominator-action envelope exactness through Definition 45.487, contextual denominator unit-composition action laws, denominator context-pair action exactness through Definition 45.568, context-pair normalized row coverage through Definition 45.570, context-pair product strict-support commutation through Definition 45.490, transported strict-support contextual separation through Definition 45.524, nonzero-product contradiction inside certified two-sided multiplication for admitted field packages, and the concrete affine automorphism $\mathsf{Group}^{\uparrow}$ certificate.
9. Standard bridge: deferred conservative bridge.
10. Status: ProofTarget.

Theorem 45.339 (RatUp FieldUp certificate record absorbs selector-independent exit)**.** *For the certificate-obligation record of Definition 45.337, for Definition 45.617, for the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, for histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, for selectors* $s, t, u, v : \mathsf{BMark}$*, and for contexts* $L, R, r : \mathsf{BHist}$*, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate record contains, as one record-level row family, every row of Definition 45.763 and every row of Definition 45.764. This family includes the selector-pair strict exit rows, the bracketing obstruction rows, both selected final boundaries, and the componentwise* $\mathsf{RatHistoryClassifier}$ *selector rows.*

Proof. Instantiate Definition 45.763 with the displayed denominator tuple, histories, carrier witnesses, selectors s, t, u, v, contexts L, R, r, and the given $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This supplies the selector-pair strict exit rows, the bracketing obstruction rows, and the selected final-boundary rows for the concrete exit object.

Instantiate Definition 45.764 with the same tuple, witnesses, selectors, contexts, and support witness. This supplies both selected final boundaries together with the componentwise R^u_D and $\mathsf{RatHistoryClassifier}$ selector rows.

The ledger fields of Definition 45.337 already name the denominator ledger, contextual-pair ledger, and coverage rows. The exactness fields name denominator exactness, contextual-action

exactness, context-pair exactness, and classifier exactness. Package the two instantiated row families under those fields, using the selector pairs and contexts as the ledger indices.

The result is a single $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ record-level final exit certificate: the traditional target is the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ record-level final exit certificate, the BEDC source is BHist, BMark, and RatUp, the carrier and classifier rows are the denominator and rational classifier rows, the ledger is selector pairs plus contexts, and the downstream consumer is the Phase Algebra Hierarchy exit criterion. □

Theorem 45.340 (RatUp FieldUp certificate record absorbs final row inventory)**.** *For the certificate-obligation record of Definition 45.337, for the concrete exit object of Definition 45.617, for the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, for histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, for selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and for contexts* $L, R, r : \mathsf{BHist}$*, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate record contains the complete final row inventory of Definition 45.773. That record-level inventory consists of selector inverse-domain coverage, selector row-stability transport, the complete selector exit obstruction, and the selector-chain final boundary rows.*

Proof. Instantiate Definition 45.773 with the displayed concrete exit object, denominator tuple, histories, carrier witnesses, selectors s, t, u, v, w, z, contexts L, R, r, and the $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This gives the four row families as one final inventory for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

Next apply Definition 45.339 to the same denominator tuple, histories, carrier witnesses, selectors s, t, u, v, contexts L, R, r, and support witness. This places the complete selector exit obstruction and the selected final-boundary rows inside the certificate-obligation record as record-level rows.

It remains to place the selector inverse-domain coverage, selector row-stability transport, and selector-chain final boundary rows in the same record. Definition 45.337 supplies the denominator ledger, contextual-pair ledger, coverage field, denominator exactness, contextual-action exactness, context-pair exactness, and classifier exactness fields. Use the selector sextuple and contexts as ledger indices, and store the remaining row families from the final inventory under those ledger and exactness fields.

The record therefore consumes the full inventory without adding a second exit object or changing the source package. Conjunction introduction identifies the record-level obstruction rows, inverse-domain rows, row-transport rows, and selector-chain rows as one certificate-record data family. □

Per-object template: traditional target final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ record-level row inventory; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations final inventory coverage and record absorption; ledger selectors plus contexts; core theorem this block; exactness final row inventory as certificate-record data; standard bridge deferred; status $\mathsf{ProofTarget}$.

Primitive boundary. $\mathsf{Field}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 45.341 (Zero-axiom encoding note)**.** *Explicit apartness witnesses support inverse formation without LEM-based* $a \neq 0$ *reasoning.*

Theorem 45.342 (Field certificate forgets to CommRing certificate)**.** *Let $\mathcal{F}$ be a BEDC history-level $\mathsf{Field}^{\uparrow}(F)$ certificate with carrier C_F, classifier R_F, operations add_F, zero_F, neg_F, mul_F, and one_F, nonzero predicate $\mathsf{NonZero}_F$, and proof-indexed inverse $\mathsf{inv}_F(a,p)$ for $p : \mathsf{NonZero}_F(a)$. If $\mathcal{F}$ satisfies the field source, pattern, classifier, stability, ledger, and obligation rows of Definition 45.198, Definition 45.199, Definition 45.200, Definition 45.201, Definition 45.336, and Definition 45.337, then*

$$\mathcal{F}_{\mathsf{CommRing}} := (C_F, R_F, \mathsf{add}_F, \mathsf{zero}_F, \mathsf{neg}_F, \mathsf{mul}_F, \mathsf{one}_F)$$

is a $\mathsf{CommRing}^{\uparrow}(F)$ source package in the sense of Definition 44.30, Definition 44.31, Definition 44.32, and Definition 44.33. It has the same carrier, classifier, additive operation, zero, additive inverse, multiplication, and unit as $\mathcal{F}$; the nonzero apartness row and all proof-indexed inverse rows are not used by this reduct.

Proof. Unpack the field source row. By Definition 45.198, a field source is already a certified commutative-ring carrier together with the additional constructive apartness predicate. Selecting the commutative-ring coordinate gives the same carrier C_F and does not change any BEDC history endpoint.

Unpack the field pattern row. By Definition 45.199, the field pattern is commutative-ring expression formation with one added formation rule for $\mathsf{inv}_F(a,p)$ when $p : \mathsf{NonZero}_F(a)$. Restricting to expressions with no inverse node leaves exactly the commutative-ring pattern from Definition 44.31, using the original add_F, neg_F, mul_F, zero_F, and one_F rows.

Unpack the classifier row. By Definition 45.200, the field classifier is ring sameness R_F together with transport of $\mathsf{NonZero}_F$ along that classifier. The $\mathsf{CommRing}^{\uparrow}(F)$ classifier requires only the ring classifier, so the projected classifier is exactly the same R_F.

Unpack the stability row. The first item of Definition 45.201 is all commutative-ring stability fields. These are precisely the closure, congruence, additive, multiplicative, distributive, and multiplicative-commutativity rows listed in Definition 44.33. The remaining field rows concern $\mathsf{NonZero}_F(0)$, nonzero transport, inverse closure, and the left and right inverse laws indexed by nonzero witnesses; none is required for the commutative-ring certificate.

Finally, Definition 45.336 records the ring ledger together with the nonzero witness used for each inverse. Selecting the ring ledger and omitting inverse-witness entries gives the commutative-ring ledger for the same operations and classifier. Thus every $\mathsf{CommRing}^{\uparrow}(F)$ row is a coordinate of $\mathcal{F}$, and the displayed tuple is the required forgetful certificate. □

Corollary 45.343 (Field certificate forgets to Ring certificate)**.** *Let $\mathcal{F}$ be a BEDC history-level $\mathsf{Field}^{\uparrow}(F)$ certificate with scalar carrier C_F, classifier R_F, operations $+_F$, 0_F, $(-)_F$, mul_F, and 1_F, nonzero predicate $\mathsf{NonZero}_F$, and proof-indexed inverse $\mathsf{inv}_F(a,p)$ for $p : \mathsf{NonZero}_F(a)$. If $\mathcal{F}$ satisfies the field source, pattern, classifier, stability, ledger, and obligation rows of Definition 45.198, Definition 45.199, Definition 45.200, Definition 45.201, Definition 45.336, and Definition 45.337, then the composite reduct obtained by first applying Definition 45.342 and then applying Definition 44.8 is a $\mathsf{Ring}^{\uparrow}(F)$ certificate. Its carrier is C_F, its classifier is R_F, and its operations are $+_F$, 0_F, $(-)_F$, mul_F, and 1_F.*

Proof. Apply Definition 45.342 to $\mathcal{F}$. The resulting commutative-ring reduct keeps the carrier C_F, the classifier R_F, addition $+_F$, zero 0_F, additive inverse $(-)_F$, multiplication mul_F, and unit 1_F. The rows involving $\mathsf{NonZero}_F$ and the proof-indexed inverse operation are omitted, because they are not part of the $\mathsf{CommRing}^{\uparrow}(F)$ obligations.

Now apply Definition 44.8 to this $\mathsf{CommRing}^{\uparrow}(F)$ certificate. That projection removes only the multiplication-commutativity row and the commutative-ring-specific ledger entries, while retaining

the carrier, source, ring pattern, classifier, ring stability fields, and ring ledger. By Definition 43.55, those retained rows are precisely a $\mathsf{Ring}^{\uparrow}(F)$ certificate.

Both projections are coordinate projections. The first does not alter the carrier or any of $+_F$, 0_F, $(-)_F$, mul_F, and 1_F, and the second projection states that the same operations and carrier classifier are unchanged. Therefore the composite reduct is the required $\mathsf{Ring}^{\uparrow}(F)$ certificate over the original scalar carrier, classifier, and ring operations. □

Theorem 45.344 (Preserved strict zero-divisor pair obstructs FieldUp)**.** *Let R be a BEDC history-level* $\mathsf{CommRing}^{\uparrow}$ *package over* BHist *with zero* 0*, unit* 1*, multiplication* $\cdot$*, classifier* hsame*, and apart-zero predicate* $\mathsf{ApartZero}_R(x) := (x\mathsf{hsame}0 \to \bot)$ *as in Definition 44.97. Let $\mathcal{F}$ be a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate satisfying Definition 45.337 with the same zero, unit, multiplication, and classifier on the displayed histories. Assume its apartness-preservation row is*

$$\alpha : \forall x : \mathsf{BHist},\ \mathsf{ApartZero}_R(x) \to \mathsf{NonZero}_F(x).$$

Then every $a, b : \mathsf{BHist}$ satisfies

$$\mathsf{ApartZero}_R(a) \wedge \mathsf{ApartZero}_R(b) \wedge a \cdot b\mathsf{hsame}0 \Longrightarrow \mathsf{False}.$$

Proof. Assume such a pair exists. Unpack its three components as $q_a : \mathsf{ApartZero}_R(a)$, $q_b : \mathsf{ApartZero}_R(b)$, and $z : a \cdot b\mathsf{hsame}0$.

Apartness preservation converts the first two components into field nonzero witnesses

$$p_a := \alpha(a)(q_a) : \mathsf{NonZero}_F(a), \qquad p_b := \alpha(b)(q_b) : \mathsf{NonZero}_F(b).$$

Because the field candidate preserves the same multiplication, zero, and classifier on these endpoints, the ring classifier row z is also the field zero-product row for a and b. The product data in Definition 45.337 supplies the hypotheses consumed by Definition 45.248. Applying that theorem to p_a, p_b, and z gives False, contradicting existence of the displayed pair. Thus a field candidate with the stated preserved coordinates cannot exist over a commutative-ring package carrying such witnesses. □

Corollary 45.345 (Strict zero divisor obstructs FieldUp)**.** *Under the same preservation hypotheses, if $a : \mathsf{BHist}$ satisfies $\mathsf{StrictZD}_R(a)$ in the sense of Definition 44.99, then no $\mathsf{Field}^{\uparrow}$ certificate satisfying Definition 45.337 preserves the same 0, 1, multiplication, classifier, and apartness over that commutative-ring package.*

Proof. Unpack $\mathsf{StrictZD}_R(a)$. It gives $\mathsf{ApartZero}_R(a)$ and a history $b : \mathsf{BHist}$ with $\mathsf{ApartZero}_R(b)$, $a \cdot b\mathsf{hsame}0$, and $b \cdot a\mathsf{hsame}0$. The first three of these components form the preserved strict zero-divisor pair excluded by Definition 45.344. Therefore any $\mathsf{Field}^{\uparrow}$ certificate with the stated preservation rows yields False. □

45.14 Rat denominator empty-unit continuation monoid instance

Definition 45.346 (Rat denominator empty-unit continuation monoid instance)**.** *The empty-unit denominator continuation monoid data for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *are the concrete tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ *defined by*

$$C^u_D(h) \Longleftrightarrow \mathsf{RatDenomUnitCarrier}(h),$$

$$R^u_D(h, k) \Longleftrightarrow \mathsf{RatDenomUnitClassifier}(h, k),$$

$$\mu^u_D(h,k) := \mathsf{append}(h,k),$$

and $e^u_D := \mathsf{e}$. *For every pair of histories* $h,k : \mathsf{BHist}$, *the multiplication is equipped with the canonical continuation witness*

$$\mathsf{Cont}(h,k,\mathsf{append}(h,k)).$$

Theorem 45.347 (Rat denominator empty-unit continuation monoid laws)**.** *The concrete tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ *of Definition 45.346 satisfies the denominator empty-unit continuation monoid laws:*

1. *The unit is carried:* $C^u_D(e^u_D)$.
2. *The multiplication is closed: if* $C^u_D(h)$ *and* $C^u_D(k)$, *then* $C^u_D(\mu^u_D(h,k))$.
3. *The left and right unit products are classified with their endpoint: for every* h *with* $C^u_D(h)$, $R^u_D(\mu^u_D(e^u_D,h),h)$ *and* $R^u_D(\mu^u_D(h,e^u_D),h)$.
4. *The multiplication is associative under* R^u_D: *if* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *then* $R^u_D(\mu^u_D(\mu^u_D(h,k),\ell),\mu^u_D(h,\mu^u_D(k,$
5. *The multiplication respects classified endpoints: if* $R^u_D(h,h')$ *and* $R^u_D(k,k')$, *then* $R^u_D(\mu^u_D(h,k),\mu^u_D(h',k'))$.

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit continuation monoid laws`

Proof. Unfold C^u_D, R^u_D, μ^u_D, and e^u_D. The displayed multiplication is the external append operation, so each product carries the canonical continuation witness $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$.

The carried empty unit, append closure, and the two one-sided unit classifications are exactly the first four clauses of Definition 45.177 applied to those canonical continuation witnesses.

For associativity, use the same adjoined-unit theorem with the four canonical continuation witnesses for the two inner and two outer products. Its fifth clause gives the classifier comparison between the two bracketed append results.

It remains to show multiplication congruence. From $R^u_D(h,h')$ and $R^u_D(k,k')$, unfold the classifier to obtain the four carried endpoint witnesses together with $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$. Apply Definition 12.3 to the two hsame comparisons; append closure for C^u_D supplies the two carried product witnesses. These three pieces are exactly $R^u_D(\mu^u_D(h,k),\mu^u_D(h',k'))$.

Thus the denominator continuation fields are concrete BEDC data: carrier $\mathsf{RatDenomUnitCarrier}$, classifier $\mathsf{RatDenomUnitClassifier}$, multiplication append, and unit e, with no abstract operator field being used. □

Theorem 45.348 (Rat denominator unit carrier empty context exactness)**.** *For histories* $p,q,h : \mathsf{BHist}$, *if* $p\mathsf{hsame}\mathsf{e}$ *and* $q\mathsf{hsame}\mathsf{e}$, *then adding those empty endpoint contexts does not change membership in the empty-unit denominator carrier:*

$$\mathsf{RatDenomUnitCarrier}(\mathsf{append}(p,\mathsf{append}(h,q))) \iff \mathsf{RatDenomUnitCarrier}(h).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier empty context iff`

Proof. Eliminate the two empty-context sameness witnesses. The right append with e reduces to h, and the left append by e is identified with h by the append left-unit law. Transport the concrete $\mathsf{RatDenomUnitCarrier}$ predicate along that history-sameness comparison in both directions. □

Theorem 45.349 (Rat denominator unit classifier empty context strict support iff)**.** *For histories* $p,q,p',q',h,k : \mathsf{BHist}$, *assume* $p\mathsf{hsame}\mathsf{e}$, $q\mathsf{hsame}\mathsf{e}$, $p'\mathsf{hsame}\mathsf{e}$, *and* $q'\mathsf{hsame}\mathsf{e}$. *If the empty-unit denominator classifier relates the two context-enlarged endpoints,*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(p,\mathsf{append}(h,q)),\mathsf{append}(p',\mathsf{append}(k,q'))),$$

then strict rational support is the same on the exposed endpoints:

$$\mathsf{RatHistoryCarrier}(h) \iff \mathsf{RatHistoryCarrier}(k).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier empty context rat carrier iff`

Proof. Apply Definition 45.405 to the four empty-context witnesses and the displayed classifier. This yields the core classifier $\mathsf{RatDenomUnitClassifier}(h, k)$. Its final field is $h\mathsf{hsame}k$, so $\mathsf{RatHistoryCarrier}$ transports from h to k; using the symmetric hsame comparison gives the reverse implication. □

45.15 Rat denominator unit endpoint exactness

Theorem 45.350 (Rat denominator continuation right-unit endpoint iff empty)**.** *For every history $u : \mathsf{BHist}$, the right-unit classifier law is equivalent to external emptiness:*

$$(\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)) \iff u\mathsf{hsame}\mathsf{e}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation right unit law endpoint empty iff`

Proof. For the forward direction, apply Theorem 45.125 to the displayed right-unit classifier law. Its first endpoint-forcing clause returns $u\mathsf{hsame}\mathsf{e}$.

Conversely, assume $u\mathsf{hsame}\mathsf{e}$, a carrier witness $\mathsf{RatHistoryCarrier}(d)$, and a displayed continuation $\mathsf{Cont}(d, u, r)$. Transport the second continuation endpoint along $u\mathsf{hsame}\mathsf{e}$ to obtain a continuation presentation $\mathsf{Cont}(d, \mathsf{e}, r)$ with the same displayed result. The right external-boundary law in Theorem 45.124 then yields $\mathsf{RatHistoryClassifier}(r, d)$. □

Theorem 45.351 (Rat denominator continuation right-unit visible endpoint exclusion)**.** *For every pair of histories $u, p : \mathsf{BHist}$, if u satisfies the right-unit classifier law for Rat denominator continuation, then u cannot be history-same to either visible endpoint over p:*

$$(\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)) \Longrightarrow (\mathsf{hsame}(u, \mathsf{E0}(p)) \to \mathsf{False}) \wedge (\mathsf{hs}$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation right unit law visible endpoint absurd`

Proof. Test the law on the concrete Rat carrier $\mathsf{E1}(\mathsf{e})$ and the displayed continuation $\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, \mathsf{E1}(\mathsf{e}) \cdot u)$. The classifier result collapses the continuation result back to $\mathsf{E1}(\mathsf{e})$, so continuation right-unit uniqueness gives $u\mathsf{hsame}\mathsf{e}$. Composing either visible presentation of u with this empty endpoint contradicts BEDC history constructor no-confusion for $\mathsf{E0}$ and $\mathsf{E1}$. □

Theorem 45.352 (Rat denominator continuation left-unit endpoint iff empty)**.** *For every history $u : \mathsf{BHist}$, the left-unit classifier law is equivalent to external emptiness:*

$$(\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d)) \iff u\mathsf{hsame}\mathsf{e}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation left unit law endpoint empty iff`

Proof. For the forward direction, apply Theorem 45.125 to the displayed left-unit classifier law. Its second endpoint-forcing clause returns $u\mathsf{hsamee}$.

Conversely, assume $u\mathsf{hsamee}$, a carrier witness $\mathsf{RatHistoryCarrier}(d)$, and a displayed continuation $\mathsf{Cont}(u, d, r)$. Transport the first continuation endpoint along $u\mathsf{hsamee}$ to obtain $\mathsf{Cont}(\mathsf{e}, d, r)$. The left external-boundary law in Theorem 45.124 classifies the displayed result as $\mathsf{RatHistoryClassifier}(r, d)$. □

Theorem 45.353 (Rat denominator left-unit result is not empty)**.** *For histories $u, d, r : \mathsf{BHist}$, suppose u satisfies the Rat-denominator left-unit classifier law:*

$$\forall x, y : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(x) \to \mathsf{Cont}(u, x, y) \to \mathsf{RatHistoryClassifier}(y, x).$$

If $\mathsf{RatHistoryCarrier}(d)$ and $\mathsf{Cont}(u, d, r)$, then $\mathsf{hsame}(r, \mathsf{e})$ implies contradiction.

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation left unit law result not empty`

Proof. Apply the displayed left-unit classifier law to the concrete carried denominator d and continuation $\mathsf{Cont}(u, d, r)$. The resulting $\mathsf{RatHistoryClassifier}(r, d)$ exposes a Rat carrier witness on r, and Rat carrier histories cannot be internally the same as e. □

Lemma 45.354 (Rat denominator left-unit results classify together)**.** *For histories $u, d, r, s : \mathsf{BHist}$, suppose u satisfies the Rat-denominator left-unit classifier law. If $\mathsf{RatHistoryCarrier}(d)$, $\mathsf{Cont}(u, d, r)$, and $\mathsf{Cont}(u, d, s)$, then $\mathsf{RatHistoryClassifier}(r, s)$.*

Proof. Apply the left-unit classifier law to both displayed continuations. This gives $\mathsf{RatHistoryClassifier}(r, d)$ and $\mathsf{RatHistoryClassifier}(s, d)$. The carrier fields supply Rat carrier witnesses for r and s, and the two hsame comparisons compose through d to classify r with s. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation left unit result classifier`

Theorem 45.355 (Rat denominator e1 left-unit law is impossible)**.** *For every tail history $t : \mathsf{BHist}$, the one-headed candidate $\mathsf{E1}(t)$ cannot satisfy the Rat-denominator left-unit classifier law:*

$$(\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(\mathsf{E1}(t), d, r) \to \mathsf{RatHistoryClassifier}(r, d)) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation e1 left unit law absurd`

Proof. The left-unit endpoint equivalence forces $\mathsf{E1}(t)\mathsf{hsamee}$ from the displayed classifier law. Constructor separation for BHist rejects any equality between a one-headed history and the empty history. □

Theorem 45.356 (Rat denominator visible left-unit law is impossible)**.** *For every history $u : \mathsf{BHist}$, if u is history-same to a visible zero-headed or one-headed history, then u cannot satisfy the Rat-denominator left-unit classifier law:*

$$((\exists t : \mathsf{BHist}.\ u\mathsf{hsameE0}(t)) \vee (\exists t : \mathsf{BHist}.\ u\mathsf{hsameE1}(t))) \to (\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{Ra}$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator continuation visible left unit law absurd`

Proof. The left-unit endpoint equivalence forces $u\mathsf{hsamee}$. In the visible zero-headed branch, transitivity gives $\mathsf{E0}(t)\mathsf{hsamee}$, contradicting constructor separation. In the one-headed branch, the same argument gives $\mathsf{E1}(t)\mathsf{hsamee}$, again impossible by constructor separation. □

Theorem 45.357 (Rat denominator e1-probe classifier endpoint empty)**.** *For every history* u : BHist*, if the one-headed denominator probe classifies the displayed product with its endpoint,*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(u, \mathsf{E1}(\mathsf{e})), \mathsf{E1}(\mathsf{e})),$$

then $u\mathsf{hsamee}$*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator e1 probe classifier endpoint empty`

Proof. Use the canonical continuation witness $\mathsf{Cont}(u, \mathsf{E1}(\mathsf{e}), \mathsf{append}(u, \mathsf{E1}(\mathsf{e})))$. The classifier supplies history sameness between its result and $\mathsf{E1}(\mathsf{e})$, so continuation result transport gives $\mathsf{Cont}(u, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$. The left-unit uniqueness law for continuation then forces $u\mathsf{hsamee}$. □

Theorem 45.358 (Rat denominator right e1-probe classifier endpoint empty)**.** *For every history* u : BHist*, if the fixed one-headed denominator probe on the left classifies the displayed product with its endpoint,*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(\mathsf{E1}(\mathsf{e}), u), \mathsf{E1}(\mathsf{e})),$$

then $u\mathsf{hsamee}$*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator right e1 probe classifier endpoint empty`

Proof. Use the canonical continuation witness $\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, \mathsf{append}(\mathsf{E1}(\mathsf{e}), u))$. The classifier supplies history sameness between its result and $\mathsf{E1}(\mathsf{e})$, so continuation result transport gives $\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), u, \mathsf{E1}(\mathsf{e}))$. The right-unit uniqueness law for continuation then forces $u\mathsf{hsamee}$. □

Theorem 45.359 (Rat history classifier right append carrier obstruction)**.** *For all histories* d, u : BHist*, if* $\mathsf{RatHistoryCarrier}(u)$ *holds, then*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(d, u), d) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier append right carrier absurd`

Proof. Use the canonical continuation $\mathsf{Cont}(d, u, \mathsf{append}(d, u))$. The classifier equality transports this continuation result to d, giving $\mathsf{Cont}(d, u, d)$. Right-unit uniqueness forces $u\mathsf{hsamee}$, contradicting the displayed $\mathsf{RatHistoryCarrier}(u)$ witness. □

Theorem 45.360 (Rat history classifier left append carrier obstruction)**.** *For all histories* u, d : BHist*, if* $\mathsf{RatHistoryCarrier}(u)$ *holds, then*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(u, d), d) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier append left carrier absurd`

Proof. Use the canonical continuation $\mathsf{Cont}(u, d, \mathsf{append}(u, d))$. The classifier equality transports this continuation result to d, giving $\mathsf{Cont}(u, d, d)$. Left-unit uniqueness forces $u\mathsf{hsamee}$, contradicting the displayed $\mathsf{RatHistoryCarrier}(u)$ witness. □

Theorem 45.361 (Rat denominator external unit separation package)**.** *For the concrete denominator carrier, classifier, and continuation operation given by*

$$C_D(h) \iff \mathsf{RatHistoryCarrier}(h),$$

$$R_D(h, k) \iff \mathsf{RatHistoryClassifier}(h, k),$$

with operation represented by Cont*, every history* u : BHist *satisfies the following three clauses.*

1. *The right classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(d, u, r) \to R_D(r, d)$$

is equivalent to $u\mathsf{hsamee}$.

2. *The left classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(u, d, r) \to R_D(r, d)$$

is equivalent to $u\mathsf{hsamee}$.

3. *If* $C_D(u)$ *holds, then either one-sided classifier-unit law displayed above implies* False.

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator external unit separation package`

Proof. The displayed definitions of C_D and R_D identify the right classifier-unit law with the concrete law in Definition 45.350. That theorem gives the first equivalence exactly.

The same unpacking identifies the left classifier-unit law with the concrete law in Definition 45.352. That theorem gives the second equivalence.

For the carried exclusion, assume $C_D(u)$. This is precisely $\mathsf{RatHistoryCarrier}(u)$. The right one-sided law contradicts Definition 45.143, while the left one-sided law contradicts Definition 45.148.

Equivalently, either endpoint equivalence first yields $u\mathsf{hsamee}$, and then Definition 37.18 excludes the simultaneous carried witness. Thus the empty endpoint is external to the positive Rat denominator carrier even though it is exact for the classifier-unit boundary laws. □

Theorem 45.362 (Rat denominator external unit endpoint disjunction uniqueness)**.** *For histories* $u, v : \mathsf{BHist}$*, suppose each endpoint satisfies at least one of the two one-sided denominator classifier-unit laws:*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

or

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

and similarly with v *in place of* u*. Then* $u\mathsf{hsame}v$.

Proof. Case-split the two endpoint alternatives. The right/right, left/left, and right/left branches are the three clauses of the endpoint uniqueness theorem. The left/right branch is the symmetric form of the right/left branch applied with the two endpoints exchanged. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator external unit endpoint disjunction unique`

Theorem 45.363 (Rat denominator external unit normal form)**.** *For every history* $u : \mathsf{BHist}$*, write* $R(u)$ *for the right classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

and write $L(u)$ *for the left classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d).$$

Then $R(u)$*,* $L(u)$*,* $R(u) \vee L(u)$*, and* $u\mathsf{hsamee}$ *are pairwise equivalent: each of the four propositions implies each of the other three. Moreover,*

$$\mathsf{RatHistoryCarrier}(u) \to (R(u) \vee L(u)) \to \mathsf{False}.$$

Proof. The right endpoint law is equivalent to $u\mathsf{hsamee}$ by Definition 45.350. The left endpoint law is equivalent to $u\mathsf{hsamee}$ by Definition 45.352. Thus $R(u)$ and $L(u)$ each factor through the same external empty endpoint, and the empty endpoint returns both one-sided laws by the reverse directions.

For the disjunction, assume $R(u) \vee L(u)$. Constructive case analysis reduces the right case to the right endpoint theorem and the left case to the left endpoint theorem, so either branch gives $u\mathsf{hsamee}$. Conversely, $u\mathsf{hsamee}$ gives both $R(u)$ and $L(u)$, hence gives the displayed disjunction. Factoring every implication through $u\mathsf{hsamee}$ gives the pairwise equivalence of all four propositions.

For the carried exclusion, assume $\mathsf{RatHistoryCarrier}(u)$ and $R(u) \vee L(u)$. In the right branch, the no-internal-unit theorem Definition 45.143 gives False. In the left branch, the no-internal-left-unit theorem Definition 45.148 gives False. Equivalently, this is the carried exclusion clause of Definition 45.361 applied to the displayed one-sided law. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator external unit normal form`

Theorem 45.364 (Rat denominator external unit uniqueness and carrier separation)**.** *For histories $u, v : \mathsf{BHist}$, write $R(w)$ for the right classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, w, r) \to \mathsf{RatHistoryClassifier}(r, d)$$

and write $L(w)$ for the left classifier-unit law

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(w, d, r) \to \mathsf{RatHistoryClassifier}(r, d).$$

If $R(u) \vee L(u)$ and $R(v) \vee L(v)$, then $u\mathsf{hsame}v$, $\mathsf{RatHistoryCarrier}(u) \to \mathsf{False}$, and $\mathsf{RatHistoryCarrier}(v) \to \mathsf{False}$.

Proof. Assume the two endpoint disjunction witnesses. Apply Definition 45.362 to these witnesses to obtain $u\mathsf{hsame}v$.

For u, use the carried-exclusion clause of Definition 45.363. Composing that clause with the assumed witness $R(u) \vee L(u)$ yields $\mathsf{RatHistoryCarrier}(u) \to \mathsf{False}$.

The same argument at v, now using the assumed witness $R(v) \vee L(v)$, yields $\mathsf{RatHistoryCarrier}(v) \to \mathsf{False}$. The three conclusions are therefore available as a single consumable boundary row. □

Traditional target: external denominator unit normal form; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, and Cont; **carrier**: Rat denominators; **classifier**: Rat history classifier; **stability obligation**: endpoint-unit exactness; **ledger**: the displayed continuation witness; **core theorem**: Definition 45.363; **exactness predicate**: one-sided unit laws coincide with external emptiness; **standard bridge**: deferred; **status**: ProofTarget.

Traditional target: external unit boundary for the Rat denominator continuation fragment; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, and Cont; **carrier**: C_D; **classifier**: R_D; **stability obligation**: endpoint exactness plus carried-unit exclusion; **ledger**: continuation witness; **core theorem**: Definition 45.361; **exactness predicate**: external-unit iff empty plus no internal carried unit; **standard bridge**: deferred; **status**: ProofTarget.

Traditional target: denominator-side continuation unit endpoint classification for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, and Cont; **carrier choice**: Rat denominators; **classifier choice**: Rat history classifier; **stability obligation**: endpoint-unit exactness; **ledger**: the displayed continuation witness; **core theorems**: Definition 45.350 and Definition 45.352; **exactness predicate**: right and left unit endpoint laws are each equivalent to external emptiness; **standard bridge**: deferred; **status**: ProofTarget.

Definition 45.365 (Rat denominator external-unit predicate)**.** *For a history* $u : \mathsf{BHist}$*, let* $U_D(u)$ *denote the proposition*

$$U_D(u) :\Longleftrightarrow (\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d)) \vee (\forall d, r : \mathsf{BHist}.\ \mathsf{RatHisto}$$

Theorem 45.366 (Rat denominator external-unit classifier disjointness)**.** *For all histories* $u, d : \mathsf{BHist}$*,*

$$U_D(u) \to (\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)) \to \mathsf{False}.$$

Proof. Unfold $U_D(u)$. A witness for this predicate is exactly one of the two one-sided denominator classifier-unit laws displayed in Definition 45.363. That normal form turns either alternative into the endpoint equality $u\mathsf{hsamee}$.

It remains to compare this endpoint equality with the displayed classifier branch. If $\mathsf{RatHistoryClassifier}(u, d)$ holds, project its left endpoint witness. This gives $\mathsf{RatHistoryCarrier}(u)$, and Definition 37.18 contradicts it together with $u\mathsf{hsamee}$.

If $\mathsf{RatHistoryClassifier}(d, u)$ holds, project its right endpoint witness. This again gives $\mathsf{RatHistoryCarrier}(u)$, so the same carrier nonemptiness theorem contradicts the endpoint equality. Hence no external unit selected by the concrete denominator unit predicate can occur as either endpoint of a Rat denominator classifier edge. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator external unit classifier disjointness`

Theorem 45.367 (Rat denominator external-unit predicate boundary package)**.** *For histories* $u, v, d : \mathsf{BHist}$*, if* $U_D(u)$ *and* $U_D(v)$*, then the following four boundary clauses hold:*

1. $u\mathsf{hsame}v$*;*
2. $\mathsf{RatHistoryCarrier}(u) \to \mathsf{False}$*;*
3. $\mathsf{RatHistoryCarrier}(v) \to \mathsf{False}$*;*
4. $(\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)) \to \mathsf{False}$*.*

Proof. Unfold $U_D(u)$ and $U_D(v)$ only to expose the two endpoint-unit disjunction witnesses. Apply Definition 45.364 to these two witnesses. This gives the uniqueness clause $u\mathsf{hsame}v$ and the two carrier-exclusion clauses for u and v.

It remains to rule out classifier edges touching the chosen external unit. Apply Definition 45.366 to the original witness $U_D(u)$. The resulting implication is exactly the displayed Rat-classifier disjointness row for the pair (u, d). □

Traditional target: external unit classifier separation for the concrete Rat denominator fragment; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, and Cont; **carrier**: Rat denominators; **classifier**: Rat history classifier; **stability obligation**: endpoint-unit normal form plus carrier nonemptiness; **ledger**: the chosen one-sided unit law and classifier endpoint; **core theorem**: Definition 45.366; **exactness predicate**: classifier-disjointness from Rat denominator endpoints; **standard bridge**: deferred; **status**: ProofTarget.

45.16 Concrete Rat denominator external-unit package

Theorem 45.368 (Rat denominator concrete continuation is cancellative and unitless)**.** *For the concrete data* C_D*,* R_D*, and* μ_D *of Definition 45.323, the denominator-side fragment consumed by the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate has the following form.*

1. *C_D, R_D, and μ_D form the carried commutative semigroup fragment: multiplication closes on carried denominators, commutativity and associativity hold up to R_D, binary classifier congruence is respected in both inputs, and the Rat history ledger is closed under classified carried right tails.*

2. *For all histories $p, q, d, e, pd, pe, left, right$: BHist, if $\mathsf{UnaryHistory}(p)$, $\mathsf{UnaryHistory}(q)$, $C_D(d)$, $C_D(e)$, $\mathsf{Cont}(p, d, pd)$, $\mathsf{Cont}(p, e, pe)$, $\mathsf{Cont}(pd, q, left)$, and $\mathsf{Cont}(pe, q, right)$, then the common unary context is exact for both the classifier and the ledger:*

$$R_D(left, right) \iff R_D(d, e).$$

$$\mathsf{RatHistoryLedgerPolicy}(left, right) \iff \mathsf{RatHistoryLedgerPolicy}(d, e).$$

3. *For every endpoint u : BHist, if either one-sided classifier-unit law*

$$\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(d, u, r) \to R_D(r, d)$$

or

$$\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(u, d, r) \to R_D(r, d)$$

holds, then $u\mathsf{hsamee}$. If also $C_D(u)$, then False.

Proof. Unfold the concrete data of Definition 45.323, so that C_D is $\mathsf{RatHistoryCarrier}$, R_D is $\mathsf{RatHistoryClassifier}$, and μ_D is append with its canonical continuation witness.

The semigroup fragment is exactly Definition 45.325. Its clauses give carrier closure, commutativity, associativity, binary classifier congruence, and ledger closure for the displayed concrete operation.

For the common-context clause, apply Definition 45.138 to the two unary contexts and the four displayed continuation witnesses. After the same unfolding of C_D and R_D, its two conclusions are precisely the classifier cancellation equivalence and the ledger exactness equivalence in the statement.

For the endpoint obstruction, use Definition 45.350 for the right-unit classifier law and Definition 45.352 for the left-unit classifier law. Either case yields $u\mathsf{hsamee}$. If a carrier witness $C_D(u)$ is also present, unfold C_D and apply Definition 37.18 to that witness and the comparison $u\mathsf{hsamee}$, giving False. □

Theorem 45.369 (Rat denominator continuation exact external-unit package)**.** *For the concrete Rat denominator continuation data (C_D, R_D, μ_D) of Definition 45.323, the following package holds.*

1. *If $C_D(d)$ and $C_D(e)$, then $C_D(\mu_D(d, e))$.*

2. *If $C_D(d)$ and $C_D(e)$, then $R_D(\mu_D(d, e), \mu_D(e, d))$.*

3. *If $C_D(d)$, $C_D(e)$, and $C_D(f)$, then $R_D(\mu_D(\mu_D(d, e), f), \mu_D(d, \mu_D(e, f)))$.*

4. *If $R_D(d, d')$ and $R_D(e, e')$, then $R_D(\mu_D(d, e), \mu_D(d', e'))$.*

5. *If $\mathsf{RatHistoryLedgerPolicy}(\rho, v)$, $C_D(e)$, and $R_D(e, e')$, then $\mathsf{RatHistoryLedgerPolicy}(\mu_D(\rho, e), \mu_D(v, e'))$.*

6. *For every endpoint u : BHist,*

$$(\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(d, u, r) \to R_D(r, d)) \iff u\mathsf{hsamee}.$$

7. *For every endpoint* $u : \mathsf{BHist}$,

$$(\forall d, r : \mathsf{BHist}.\ C_D(d) \to \mathsf{Cont}(u, d, r) \to R_D(r, d)) \iff u\mathsf{hsamee}.$$

8. *If* $C_D(u)$, *then neither displayed one-sided classifier-unit law can hold for* u.

9. *If* u *and* v *each satisfy either displayed one-sided classifier-unit law, then* $u\mathsf{hsame}v$.

Proof. Unfold the concrete names C_D, R_D, and μ_D only as needed. The first five clauses are exactly the first law group in Definition 45.325: carrier closure, commutative classifier, associative classifier, binary classifier congruence, and ledger closure.

For the right external-unit exactness clause, read C_D as $\mathsf{RatHistoryCarrier}$ and R_D as $\mathsf{RatHistoryClassifier}$. Apply Definition 45.350 to the displayed right-unit law.

For the left external-unit exactness clause, use the same identifications and apply Definition 45.352.

For carried-endpoint exclusion, suppose $C_D(u)$ and one of the displayed one-sided classifier-unit laws holds. The corresponding exactness clause gives $u\mathsf{hsamee}$. Since $C_D(u)$ is $\mathsf{RatHistoryCarrier}(u)$, rational history carrier nonemptiness, Definition 37.18, contradicts $u\mathsf{hsamee}$.

Finally, take two one-sided classifier-unit endpoints u and v. The right/right, left/left, and right/left cases are the three clauses of Definition 45.129. The left/right case follows by applying the mixed clause in the opposite order and then using symmetry of hsame. □

Traditional target: denominator-side continuation package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ with an external unit endpoint; **BEDC source**: $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, Cont, append, and $\mathsf{RatHistoryLedgerPolicy}$; **carrier choice**: C_D from Definition 45.323; **classifier choice**: R_D from Definition 45.323; **stability obligation**: carrier closure, commutative classifier, associative classifier, classifier congruence, ledger closure, and endpoint uniqueness; **ledger**: the Rat history ledger policy transported through μ_D; **core theorem**: Definition 45.369; **exactness predicate**: the right and left external-unit iff clauses identifying the endpoint with e; **standard bridge**: deferred; **status**: $\mathsf{ProofTarget}$.

Rat denominator external-unit envelope law spine. The denominator envelope used at the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ boundary is the concrete history carrier $C^*_D(h) \equiv \mathsf{RatHistoryCarrier}(h) \vee h\mathsf{hsamee}$, with classifier R^*_D given by the guarded carried Rat classifier branch or the empty-empty endpoint branch. The product is the BEDC history append operation, and the displayed endpoint is $e^*_D = \mathsf{e}$.

The checked law package for this envelope consists of carrier inhabitance, append closure, associativity under R^*_D, the left and right endpoint unit laws, and binary classifier congruence. On carried denominators the guarded branch restricts back to the positive-denominator Rat classifier, while the empty branch supplies the external endpoint. The one-sided unit uniqueness and unit equivalence clauses recorded below use this same law spine together with the nonemptiness of carried Rat denominators.

Definition 45.370 (Rat denominator external-unit envelope)**.** *The Rat denominator external-unit envelope over the concrete continuation instance of Definition 45.323 has carrier, classifier, operation, and unit*

$$C^*_D(h) \iff \mathsf{RatHistoryCarrier}(h) \vee h\mathsf{hsamee},$$

$$R^*_D(h, k) \iff (\mathsf{RatHistoryCarrier}(h) \wedge \mathsf{RatHistoryCarrier}(k) \wedge \mathsf{RatHistoryClassifier}(h, k)) \vee (h\mathsf{hsamee} \wedge k\mathsf{hsamee}),$$

$$\mu^*_D(h, k) := \mathsf{append}(h, k),$$

with external unit $e^*_D := \mathsf{e}$. *The first classifier branch is the original Rat denominator classifier guarded by carried denominator evidence. The second branch is the single external endpoint clause, so mixed carried-empty pairs are not classified merely by adjoining* e^*_D.

Theorem 45.371 (Rat denominator external-unit envelope classifier normal form)**.** *For the explicit envelope of Definition 45.370 and the adjoined empty-unit classifier of Definition 45.154, the following normal forms coincide:*

1. *For every history* $h : \mathsf{BHist}$,

$$C^*_D(h) \iff \mathsf{RatDenomUnitCarrier}(h).$$

2. *For every pair of histories* $h, k : \mathsf{BHist}$,

$$R^*_D(h,k) \iff C^*_D(h) \wedge C^*_D(k) \wedge h\mathsf{hsame}k.$$

3. *For every pair of histories* $h, k : \mathsf{BHist}$,

$$R^*_D(h,k) \iff \mathsf{RatDenomUnitClassifier}(h,k).$$

Proof. Unfold C^*_D and RatDenomUnitCarrier. The two carriers have the same two alternatives, namely a carried Rat denominator endpoint and the external empty endpoint; the displayed iff only exchanges the order of these alternatives.

For the classifier-to-normal-form direction, split the defining alternatives of $R^*_D(h,k)$. In the guarded carried branch, RatHistoryClassifier(h,k) is precisely carried-left, carried-right, and $h$hsame$k$, so the two endpoints lie in C^*_D by the carried alternative and the final comparison is the classifier comparison.

In the empty-empty branch, both endpoints lie in C^*_D by the empty alternative, and transitivity with symmetry gives $h$hsame$k$ from $h$hsamee and $k$hsamee.

For the normal-form-to-classifier direction, split the two C^*_D endpoint witnesses. The carried-carried case uses RatHistoryCarrier_hsame_transport to view the right endpoint as a Rat carrier when needed, then packs carried-left, carried-right, and $h$hsame$k$ as RatHistoryClassifier(h,k) and chooses the guarded branch of R^*_D.

The empty-empty case chooses the second branch of R^*_D after transporting the displayed $h$hsame$k$ comparison to the common endpoint e.

The two mixed cases are impossible: a carried endpoint that is hsame to an empty endpoint is excluded by RatHistoryCarrier_not_empty, using transitivity and symmetry to orient the contradiction.

Finally unfold RatDenomUnitClassifier. By the first clause, RatDenomUnitCarrier is the same carrier as C^*_D; by the second clause, $R^*_D(h,k)$ is exactly the conjunction of the two carrier witnesses and $h$hsame$k$. This is precisely the denominator unit classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier carrier empty split iff`

Theorem 45.372 (Rat denominator external-unit envelope classifier exactness)**.** *For all histories* $h, k : \mathsf{BHist}$, *the classifier* R^*_D *of Definition 45.370 splits exactly as follows.*

1. *If* RatHistoryCarrier(h) *and* RatHistoryCarrier(k), *then*

$$R^*_D(h,k) \iff \mathsf{RatHistoryClassifier}(h,k).$$

2. *If* $\mathsf{RatHistoryCarrier}(h)$ *and* $k\mathsf{hsamee}$*, then* $\neg R^*_D(h,k)$*.*

3. *If* $h\mathsf{hsamee}$ *and* $\mathsf{RatHistoryCarrier}(k)$*, then* $\neg R^*_D(h,k)$*.*

4. *If* $h\mathsf{hsamee}$ *and* $k\mathsf{hsamee}$*, then* $R^*_D(h,k)$*.*

Proof. Unfold R^*_D. In the carried-carried case, the forward implication cannot come from the empty-empty branch, because that branch would make the carried endpoint h hsame to e, contradicting RatHistoryCarrier_not_empty. The remaining branch is exactly the guarded $\mathsf{RatHistoryClassifier}(h,k)$ branch. The reverse implication uses the two displayed carrier witnesses together with that guarded branch.

For a carried-left and empty-right pair, the guarded carried branch would give $\mathsf{RatHistoryCarrier}(k)$, and RatHistoryCarrier_not_empty rejects it against $k\mathsf{hsamee}$. The empty-empty branch would make $h\mathsf{hsamee}$, which is rejected by the same nonemptiness theorem applied to $\mathsf{RatHistoryCarrier}(h)$.

The empty-left and carried-right pair is symmetric. The guarded carried branch would make the left endpoint a Rat carrier and hence nonempty, while the empty-empty branch contradicts the carried right endpoint through RatHistoryCarrier_not_empty.

If both endpoints are hsame to e, choose the second branch in the definition of R^*_D. This is precisely the external endpoint clause. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope classifier exactness`

Theorem 45.373 (Rat denominator external-unit envelope empty-right endpoint iff)**.** *For every history* $h : \mathsf{BHist}$*,*

$$R^*_D(h, e^*_D) \iff h\mathsf{hsamee}.$$

↪variant `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty right iff`

Proof. For the forward implication, use Definition 45.372 with the right endpoint specialized to $e^*_D = \mathsf{e}$. The guarded carried branch would make the empty endpoint a Rat denominator carrier, contradicting Definition 37.18; hence the only surviving branch is the empty-empty endpoint branch, which gives $h\mathsf{hsamee}$.

For the reverse implication, combine the displayed $h\mathsf{hsamee}$ with reflexivity at $e^*_D = \mathsf{e}$ and choose the empty-empty branch in the defining classifier of Definition 45.370. □

Theorem 45.374 (Rat denominator external-unit envelope empty-left endpoint iff)**.** *For all histories* $h, k : \mathsf{BHist}$*, if* $h\mathsf{hsamee}$*, then*

$$R^*_D(h,k) \iff k\mathsf{hsamee}.$$

↪variant `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty left endpoint iff`

Proof. Assume $h\mathsf{hsamee}$. For the forward implication, split the classifier by Definition 45.372. The guarded carried branch would make h a Rat denominator carrier, contradicting Definition 37.18 together with $h\mathsf{hsamee}$. Therefore the classifier lies in the empty-empty branch, and that branch gives $k\mathsf{hsamee}$.

For the reverse implication, pair the assumed $h\mathsf{hsamee}$ with the displayed $k\mathsf{hsamee}$ and use the empty-empty branch of Definition 45.370. □

Theorem 45.375 (Rat denominator external-unit envelope continuation closure)**.** *For all histories* $h, h', k, k', r, r' : \mathsf{BHist}$*, if* $R^*_D(h,h')$*,* $R^*_D(k,k')$*,* $\mathsf{Cont}(h,k,r)$*, and* $\mathsf{Cont}(h',k',r')$*, then* $R^*_D(r,r')$*.*

↪variant BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier continuation closed

Proof. Split the two input classifier witnesses by Definition 45.372. In the carried/carried branch, apply Definition 45.131 to the two carried Rat classifier witnesses and the two displayed continuation witnesses. The result is the guarded carried branch of $R^*_D(r, r')$.

In the carried/empty branch, the right input pair is identified with e. The continuation right-unit laws identify r with h and r' with h', while cont_respects_hsame transports the carried classifier comparison across the displayed continuations. Transport the Rat carrier witnesses along these endpoint comparisons and repack the guarded branch.

The empty/carried branch is symmetric, using the continuation left-unit laws to identify r with k and r' with k'. The same continuation transport gives $r$hsamer', and carrier transport repacks the guarded carried branch.

In the empty/empty branch, continuation transport along the two empty endpoint comparisons reduces both results to the empty-left unit continuation at e. Thus $r$hsamee and r'hsamee, so the empty-empty branch of $R^*_D(r, r')$ applies. □

Theorem 45.376 (Rat denominator external-unit envelope carrier continuation closure)**.** *For all histories h, k, r : BHist, if h and k lie in the external-unit envelope carrier and Cont(h, k, r), then r lies in the same envelope carrier.*

Proof. Split the two carrier witnesses into carried Rat and empty endpoint branches. Each carrier witness gives a reflexive envelope classifier on its endpoint. Apply envelope classifier continuation closure to the two reflexive classifiers and the displayed continuation twice. The resulting classifier of r against itself exposes either the carried Rat branch for r or the empty endpoint branch, which is exactly the envelope carrier. □

Lean4 ✓ BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeCarrier continuation closed

Theorem 45.377 (Rat denominator external-unit envelope monoid laws)**.** *The data $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ of Definition 45.370 satisfy the following laws.*

1. *Carrier inhabitance: $C^*_D(e^*_D)$.*
2. *Closure: if $C^*_D(h)$ and $C^*_D(k)$, then $C^*_D(\mu^*_D(h, k))$.*
3. *Associativity under the classifier: if $C^*_D(h)$, $C^*_D(k)$, and $C^*_D(l)$, then $R^*_D(\mu^*_D(\mu^*_D(h, k), l), \mu^*_D(h, \mu^*_D(k, l)))$.*
4. *Left and right unit laws: $R^*_D(\mu^*_D(e^*_D, h), h)$ and $R^*_D(\mu^*_D(h, e^*_D), h)$ for every h with $C^*_D(h)$.*
5. *Binary classifier congruence: if $R^*_D(h, h')$ and $R^*_D(k, k')$, then $R^*_D(\mu^*_D(h, k), \mu^*_D(h', k'))$.*
6. *Restriction: on carried denominators, C^*_D, R^*_D, and μ^*_D restrict to C_D, R_D, and μ_D from Definition 45.323.*
7. *Obstruction: if a carried endpoint u satisfies either one-sided classifier-unit law on carried denominators, then* False.

Lean4 ✓ BEDC.Derived.FieldUp.field rat denominator unit envelope monoid laws

Proof. For carrier inhabitance, use the right disjunct in C^*_D with reflexivity at e. The carried/carried closure case is exactly Definition 45.325; the carried/empty and empty/carried cases reduce by the append unit laws and then transport the carried witness by RatHistoryCarrier_hsame_transport. The empty/empty case is the right carrier disjunct at e.

For associativity, split the three envelope carrier witnesses into their carried or external-empty alternatives. The carried/carried/carried branch is the associativity clause of Definition 45.325. Every branch containing an external-empty endpoint uses the append unit laws to remove that endpoint, uses $\mathsf{append_assoc}$ for the common bracketing spine, and then repacks either the transported carried classifier or the reflexive empty-empty classifier branch.

For the unit laws, split $C^*_D(h)$. In the carried branch, $\mathsf{append}(\mathsf{e}, h)$ and $\mathsf{append}(h, \mathsf{e})$ are transported back to h by the append unit laws, and $\mathsf{RatHistoryClassifier_hsame_transport}$ supplies the guarded carried classifier branch. In the external-empty branch, both displayed products are hsame to e, so the empty-empty branch of R^*_D applies.

For binary classifier congruence, split the two classifier witnesses. The carried/carried classifier branch invokes the congruence clause of Definition 45.325. If one input classifier is the empty-empty branch, the append unit laws collapse that side, and $\mathsf{RatHistoryClassifier_hsame_transport}$ transports the remaining carried classifier; if both are empty-empty, both appended products are hsame to e.

The restriction clause is immediate from the guarded carried branch of R^*_D and the definitions of C^*_D and μ^*_D when the inputs are carried denominators.

Finally, a carried endpoint satisfying a right or left one-sided classifier-unit law on the carried denominator fragment is ruled out by Definition 45.144. Thus adjoining the external endpoint gives a unit-bearing envelope, not an internal $\mathsf{Field}^{\uparrow}$ multiplication on the positive Rat denominator carrier. □

Theorem 45.378 (Rat denominator external-unit envelope unit uniqueness)**.** *For every history $u : \mathsf{BHist}$ with $C^*_D(u)$, the unit laws of the external envelope classify u as the named endpoint e^*_D:*

1. *if*

$$\forall h : \mathsf{BHist}.\ C^*_D(h) \to R^*_D(\mu^*_D(h, u), h),$$

then $R^*_D(u, e^*_D)$;

2. *if*

$$\forall h : \mathsf{BHist}.\ C^*_D(h) \to R^*_D(\mu^*_D(u, h), h),$$

then $R^*_D(u, e^*_D)$.

Proof. Split the carrier witness $C^*_D(u)$ into the carried Rat denominator branch and the external-empty branch. In the external-empty branch, $u\mathsf{hsamee}$ and $e^*_D = \mathsf{e}$, so the empty-empty branch of R^*_D immediately gives $R^*_D(u, e^*_D)$ for either one-sided law.

Assume now that u is carried and that the right envelope unit law is displayed. Restrict this law to a carried Rat denominator d. The closure part of Definition 45.377 makes $\mu^*_D(d, u)$ carried, while Definition 37.18 rules out the empty-empty branch against the carried input d.

Unfolding the guarded branch of R^*_D therefore gives the Rat denominator classifier for the canonical append product. For any displayed continuation $\mathsf{Cont}(d, u, r)$, compare it with the canonical append continuation witness and transport the classifier to obtain $\mathsf{RatHistoryClassifier}(r, d)$. This is exactly the right-unit continuation law, so Definition 45.350 yields $u\mathsf{hsamee}$.

The empty comparison $u\mathsf{hsamee}$, together with $e^*_D = \mathsf{e}$, gives the empty-empty classifier branch of $R^*_D(u, e^*_D)$.

The left envelope unit law is symmetric. Restrict it to carried Rat denominators, unfold the carried branch of $R^*_D(\mu^*_D(u, d), d)$, use the canonical append continuation witness to obtain

the displayed left-unit continuation law, and apply Definition 45.352. The resulting comparison $u\mathsf{hsamee}$ again repacks through the empty-empty branch as $R^*_D(u, e^*_D)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope unit unique`

Theorem 45.379 (Rat denominator external-unit envelope carried classifier exactness)**.** *For all histories $h, k : \mathsf{BHist}$, if $\mathsf{RatHistoryCarrier}(h)$ and $\mathsf{RatHistoryCarrier}(k)$, then*

$$R^*_D(h, k) \iff \mathsf{RatHistoryClassifier}(h, k).$$

Proof. Unfold R^*_D from Definition 45.370. The reverse direction uses the guarded carried branch: the two displayed carrier witnesses for h and k, together with $\mathsf{RatHistoryClassifier}(h, k)$, assemble the first disjunct of the envelope classifier.

For the forward direction, split the envelope classifier witness. In the guarded carried branch, the third component is already $\mathsf{RatHistoryClassifier}(h, k)$, so projection gives the desired Rat classifier.

It remains to reject the empty-empty branch on carried inputs. That branch contains $h\mathsf{hsamee}$; applying Definition 37.18 to the carried witness for h and this endpoint comparison yields False, from which the Rat classifier conclusion follows constructively. □

This is the exactness predicate promised for restriction back to the carried Rat denominator continuation instance.

Traditional target: unit-enveloped denominator monoid fragment for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, Cont, and append; **carrier choice**: the external-unit envelope C^*_D; **classifier choice**: the carried Rat classifier plus the empty-empty endpoint classifier; **stability obligation**: carrier closure, associativity, unit laws, classifier congruence, restriction to the carried continuation instance, unit endpoint uniqueness, and one-sided carried-unit obstruction; **ledger**: the external unit bit plus continuation witness; **core theorems**: Definition 45.377 and Definition 45.378; **exactness predicate**: restriction back to the carried Rat denominator continuation instance, with the adjoined endpoint as the unique envelope unit up to R^*_D; **standard bridge**: deferred; **status**: $\mathsf{ProofTarget}$.

Theorem 45.380 (Rat denominator external-unit envelope is strict)**.** *For the concrete carried data C_D, R_D, and μ_D of Definition 45.323 and the envelope data C^*_D, R^*_D, μ^*_D, and e^*_D of Definition 45.370, the adjoined endpoint is strictly external and the carried fragment is recovered exactly.*

1. *$C^*_D(e^*_D)$ and $\neg C_D(e^*_D)$.*
2. *For every $h : \mathsf{BHist}$, if $C_D(h)$, then $\neg R^*_D(h, e^*_D)$ and $\neg R^*_D(e^*_D, h)$.*
3. *For every $h, k : \mathsf{BHist}$, if $C_D(h)$ and $C_D(k)$, then*

$$R^*_D(h, k) \iff R_D(h, k).$$

On the same carried inputs, the displayed product is the original append product:

$$\mu^*_D(h, k) = \mu_D(h, k) = \mathsf{append}(h, k).$$

Proof. Carrier inhabitance is the first clause of Definition 45.377. For the exclusion from the carried fragment, unfold Definition 45.323 and Definition 45.370: e^*_D is e and C_D is $\mathsf{RatHistoryCarrier}$. Then Definition 37.18 rules out a carried witness at the empty history.

For the mixed right endpoint, assume $C_D(h)$ and $R^*_D(h, e^*_D)$. Unfolding R^*_D gives either the guarded carried branch or the empty-empty branch. The guarded carried branch includes $C_D(e^*_D)$, contradicting the endpoint exclusion already proved. The empty-empty branch gives $h\mathsf{hsamee}$, contradicting Definition 37.18 applied to the carried witness $C_D(h)$. The mixed left endpoint is identical after swapping the two inputs.

It remains to recover the carried classifier. Suppose $C_D(h)$ and $C_D(k)$. If $R^*_D(h, k)$ holds, unfold R^*_D. The empty-empty branch would give $h\mathsf{hsamee}$, contradicting the same non-emptiness theorem, so the guarded carried branch remains and yields $R_D(h, k)$.

Conversely, from $C_D(h)$, $C_D(k)$, and $R_D(h, k)$, the guarded carried branch of $R^*_D(h, k)$ applies directly. Finally, unfolding the two displayed operations shows that μ^*_D and μ_D both return the append history on carried inputs. □

Theorem 45.381 (Rat denominator external-unit envelope strict carried fragment)**.** *For the concrete external-unit envelope of Definition 45.370, the endpoint* e *is in the envelope carrier, is not in the carried Rat-history carrier, has no mixed classifier edge to a carried history on either side, and the classifier restricted to carried histories is exactly* RatHistoryClassifier*.*

Proof. The endpoint carrier clause is the carrier component of Definition 45.377. The exclusion from the carried fragment is Definition 37.18 at e. The two mixed endpoint obstructions and the carried-classifier exactness are the corresponding concrete clauses of the envelope classifier exactness theorem. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelope external strict`

Theorem 45.382 (Rat denominator external-unit envelope mixed endpoint separation)**.** *For all* $d, e : \mathsf{BHist}$*, the external-unit classifier* R^u_D *has no mixed carried-empty endpoint branch:*

$$(\mathsf{RatHistoryCarrier}(d) \wedge e\mathsf{hsamee}) \to (R^u_D(d, e) \to \mathsf{False}),$$

and

$$(d\mathsf{hsamee} \wedge \mathsf{RatHistoryCarrier}(e)) \to (R^u_D(d, e) \to \mathsf{False}).$$

Proof. Unfold R^u_D and then RatDenomUnitClassifier. In the first displayed implication, the classifier supplies $d\mathsf{hsamee}$. Combining this comparison with $e\mathsf{hsamee}$ by symmetry and transitivity of hsame from Definition 3.8 gives $d\mathsf{hsamee}$.

Together with $\mathsf{RatHistoryCarrier}(d)$, rational history carrier nonemptiness, Definition 37.18, gives False.

The second displayed implication is symmetric. The classifier again supplies $d\mathsf{hsamee}$; together with $d\mathsf{hsamee}$ and the equivalence laws for hsame, this gives $e\mathsf{hsamee}$. The carried endpoint $\mathsf{RatHistoryCarrier}(e)$ is then contradicted by Definition 37.18. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier mixed endpoint absurd`

Traditional target: mixed-endpoint separation for the unit envelope; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, Cont, and e; **carrier choice**: C^u_D; **classifier choice**: R^u_D; **stability obligation**: mixed-branch separation; **ledger**: the external-unit bit; **core theorem**: Definition 45.382; **exactness predicate**: absence of mixed carried-empty classification; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.383 (Rat denominator external-unit envelope certificate)**.** *For the concrete carried denominator data* (C_D, R_D, μ_D) *of Definition 45.323, the tuple* $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ *of Definition 45.370 is the concrete unit-bearing denominator envelope used by* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *in the following sense.*

1. *$C_D^*(e_D^*)$; C_D^* is closed under μ_D^*; μ_D^* is associative up to R_D^*; e_D^* satisfies the left and right unit laws; and μ_D^* respects binary R_D^*-classified endpoints.*
2. *On carried denominators, the envelope restricts to the concrete fragment: C_D^* restricts to C_D, R_D^* restricts to R_D, and μ_D^* restricts to μ_D.*
3. *The endpoint e_D^* is envelope-carried and not C_D-carried, and no carried endpoint is classified with e_D^* on either side.*
4. *The empty-unit endpoint classifier R_D^u has no mixed carried-empty endpoint branch.*
5. *If $u : \mathsf{BHist}$ is carried by C_D^* and satisfies either one-sided envelope unit law, then $R_D^*(u, e_D^*)$.*

Proof. The carrier inhabitance, closure, associativity, one-sided unit laws, and binary classifier congruence are exactly the law package in Definition 45.377. The same theorem also records the restriction of C_D^*, R_D^*, and μ_D^* to the carried denominator fragment.

For carried classifier exactness, use Definition 45.379 on carried pairs. The strict endpoint and the exclusion of carried endpoints against e_D^* are the three clauses of Definition 45.380.

The mixed carried-empty exclusion for the endpoint classifier is precisely Definition 45.382. It supplies both orientations of the carried-empty case split.

Finally, any carried envelope endpoint satisfying a right or left one-sided unit law is classified with e_D^* by Definition 45.378. These pieces package the concrete carrier, classifier, append operation, external endpoint, restriction, and endpoint separation into the displayed denominator envelope. □

Traditional target: Rat denominator external-unit envelope certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, Cont, and e; **carrier choice**: C_D^*; **classifier choice**: R_D^*; **stability obligation**: carrier closure, associativity, unit laws, classifier congruence, restriction, strict endpoint externality, unit uniqueness, and mixed-endpoint separation; **ledger**: the external-unit bit plus continuation witness; **core theorem**: Definition 45.383; **exactness predicate**: restriction to the carried denominator continuation instance, strict externality of e_D^*, unit uniqueness, and mixed-endpoint separation; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.384 (Rat denominator external-unit envelope MonoidUp certificate)**.** *The concrete tuple $(C_D^*, R_D^*, \mu_D^*, e_D^*)$ of Definition 45.370, with carrier C_D^*, classifier R_D^*, operation* append*, and unit* e*, supplies the concrete* BHist*-level* $\mathsf{Monoid}^{\uparrow}$ *naming certificate for the unit-enveloped Rat denominator continuation fragment. Its certificate ledger is the external-unit bit together with the continuation witness used by the guarded carried branch of R_D^*.*

Proof. Apply Definition 45.377. Its carrier inhabitance clause supplies the exported unit witness for e_D^*, and its closure clause supplies the carrier witness for every exported append product in the envelope.

The same theorem supplies the left and right unit laws, associativity under R_D^*, and binary classifier congruence. These are exactly the stability fields required by Definition 40.34 for a $\mathsf{Monoid}^{\uparrow}$ interface over a BEDC carrier and classifier.

It remains to identify the certificate packaging. The carrier is the concrete predicate C_D^* on BHist; the classifier is the concrete relation R_D^*; the source operation is the native BEDC append operation through μ_D^*; and the named unit is e. Thus Definition 40.35 packages these fields as a NameCert for $\mathsf{Monoid}^{\uparrow}(C_D^*)$.

The ledger is concrete rather than schematic. The right carrier disjunct records whether the endpoint is the adjoined external unit, while the guarded carried classifier branch records the

continuation witness that relates the native Rat denominator fragment to the envelope product. This is the required BHist-anchored certificate consumed at the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ boundary. □

Theorem 45.385 (Rat denominator external-unit envelope empty-product reflection)**.** *For all histories $h, k : \mathsf{BHist}$ with $C^*_D(h)$ and $C^*_D(k)$, the product is classified as the external unit exactly when both factors are:*

$$R^*_D(\mu^*_D(h,k), e^*_D) \iff (R^*_D(h, e^*_D) \wedge R^*_D(k, e^*_D)) \,.$$

Proof. Unfold the envelope data of Definition 45.370. The named endpoint e^*_D is e, and $\mu^*_D(h,k)$ is the native BEDC history $\mathsf{append}(h,k)$.

For the forward direction, assume $R^*_D(\mu^*_D(h,k), e^*_D)$. The guarded carried classifier branch would make e a Rat denominator carrier, contradicting Definition 37.18. Hence the classifier witness is the empty-empty branch, so $\mathsf{append}(h,k)\mathsf{hsamee}$.

The BEDC append empty-result inversion theorem $\mathsf{append_eq_empty_iff}$ splits this endpoint comparison into $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. With $e^*_D = \mathsf{e}$, the empty-empty branch of R^*_D repacks these two endpoint witnesses as $R^*_D(h, e^*_D)$ and $R^*_D(k, e^*_D)$.

Conversely, assume $R^*_D(h, e^*_D) \wedge R^*_D(k, e^*_D)$. Each classifier against $e^*_D = \mathsf{e}$ has no viable guarded carried branch, again by Definition 37.18. Thus the two witnesses expose $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$.

Apply the introduction direction of $\mathsf{append_eq_empty_iff}$ to these two empty endpoint comparisons. This gives $\mathsf{append}(h,k)\mathsf{hsamee}$, and the empty-empty branch of R^*_D classifies $\mu^*_D(h,k)$ with e^*_D. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope empty product reflection`

Traditional target: Rat denominator external-unit envelope empty-product reflection for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier, R^*_D, append, and e; **carrier choice**: C^*_D; **classifier choice**: R^*_D; **stability obligation**: empty product reflection under the external-unit classifier; **ledger**: the external-unit bit plus carried continuation witness; **core theorem**: Definition 45.385; **exactness predicate**: product classification by e^*_D is exactly factorwise classification by e^*_D; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.386 (Rat denominator external-unit envelope unit equivalence)**.** *For every history $u : \mathsf{BHist}$ with $C^*_D(u)$,*

$$R^*_D(u, e^*_D) \iff (\forall h : \mathsf{BHist}.\ C^*_D(h) \to R^*_D(\mu^*_D(h,u), h)) \wedge (\forall h : \mathsf{BHist}.\ C^*_D(h) \to R^*_D(\mu^*_D(u,h), h)) \,.$$

Proof. For the forward direction, assume $R^*_D(u, e^*_D)$ and split the carrier witness $C^*_D(u)$. The carried branch is impossible: the mixed carried-empty clause of Definition 45.372 rejects $R^*_D(u, e^*_D)$ because $e^*_D = \mathsf{e}$.

In the external-empty branch, $u\mathsf{hsamee}$. For any h with $C^*_D(h)$, split $C^*_D(h)$. If h is carried, the append unit laws identify $\mu^*_D(h,u)$ and $\mu^*_D(u,h)$ with h; transport the carried witnesses by $\mathsf{RatHistoryCarrier_hsame_transport}$ and repack through the guarded carried branch of R^*_D. If $h\mathsf{hsamee}$, both products are hsame to e, so the empty-empty branch gives the right and left laws.

For the reverse direction, project the conjunction into the displayed right and left envelope unit laws. Applying Definition 45.378 to either projection gives $R^*_D(u, e^*_D)$, and the two projections therefore give the same endpoint classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope unit iff`

Theorem 45.387 (Rat denominator unit classifier zero-headed endpoint separation)**.** *For every* $t, k : \mathsf{BHist}$*, the adjoined-unit classifier cannot classify a zero-headed endpoint in either argument:*

$$R^u_D(\mathsf{E0}(t), k) \to \mathsf{False},$$

and

$$R^u_D(k, \mathsf{E0}(t)) \to \mathsf{False}.$$

Proof. Project the classifier endpoint carriers from RatDenomUnitClassifier. A zero-headed history is neither the adjoined empty unit nor a carried rational denominator history, so the corresponding carrier component contradicts the BEDC constructor separation for the denominator-unit carrier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier e0 endpoint absurd`

Theorem 45.388 (Rat denominator unit classifier continuation zero-headed endpoint separation)**.** *For classified denominator-unit pairs* (h, h') *and* (k, k')*, if* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{Cont}(h', k', r')$*, then neither continuation result can be displayed as a zero-headed endpoint:*

$$(\mathsf{hsame}(r, \mathsf{E0}(t)) \to \mathsf{False}) \wedge (\mathsf{hsame}(r', \mathsf{E0}(t')) \to \mathsf{False}).$$

Proof. Project the carrier fields from the two classifier witnesses. Denominator-unit carrier closure under continuation gives carrier witnesses for r and r'. Transport each carrier witness along a displayed zero-headed endpoint comparison and apply carrier zero-headed absurdity. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier continuation e0 endpoint absurd`

Theorem 45.389 (Rat denominator unit classifier one-headed empty endpoint separation)**.** *For every history* $d : \mathsf{BHist}$*, the adjoined-unit classifier cannot classify a one-headed endpoint against* e *in either direction:*

$$R^u_D(\mathsf{E1}(d), \mathsf{e}) \to \mathsf{False},$$

and

$$R^u_D(\mathsf{e}, \mathsf{E1}(d)) \to \mathsf{False}.$$

Proof. Unfold R^u_D as RatDenomUnitClassifier. The classifier contains the displayed hsame comparison between its two endpoints. The first branch would give $\mathsf{E1}(d)\mathsf{hsamee}$, and the second branch would give $\mathsf{ehsameE1}(d)$; both contradict the constructor separation laws for BHist. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier e1 empty absurd`

Theorem 45.390 (Rat denominator unit classifier mixed endpoint separation)**.** *For all* $d, e : \mathsf{BHist}$*, the external-unit classifier* R^u_D *has no mixed carried-empty endpoint branch:*

$$(\mathsf{RatHistoryCarrier}(d) \wedge e\mathsf{hsamee}) \to (R^u_D(d, e) \to \mathsf{False}),$$

and

$$(d\mathsf{hsamee} \wedge \mathsf{RatHistoryCarrier}(e)) \to (R^u_D(d, e) \to \mathsf{False}).$$

Proof. Unfold R^u_D and then RatDenomUnitClassifier. In the first displayed implication, the classifier supplies $d$hsame$e$. Combining this comparison with $e$hsame$e$ by symmetry and transitivity of hsame from Definition 3.8 gives $d$hsame$e$.

Together with RatHistoryCarrier(d), rational history carrier nonemptiness, Definition 37.18, gives False.

The second displayed implication is symmetric. The classifier again supplies $d$hsame$e$; together with $d$hsame$e$ and the equivalence laws for hsame, this gives $e$hsame$e$. The carried endpoint RatHistoryCarrier(e) is then contradicted by Definition 37.18. □

Traditional target: mixed-endpoint separation for the adjoined-unit classifier; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, Cont, and e; **carrier choice**: C^u_D; **classifier choice**: R^u_D; **stability obligation**: mixed-branch separation; **ledger**: the external-unit bit; **core theorem**: Definition 45.390; **exactness predicate**: absence of mixed carried-empty classification; **standard bridge**: deferred; **status**: ProofTarget.

45.17 Rat denominator strict monoid boundary

Theorem 45.391 (Rat denominator strict monoid boundary package)**.** *For the concrete carried denominator data (C_D, R_D, μ_D) of Definition 45.323, the external-unit envelope $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ of Definition 45.370, and the empty-unit tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ of Definition 45.346, the denominator boundary consumed by* Field$^\uparrow$(RatUp) *has the following concrete role.*

1. *The envelope supplies the concrete* Monoid$^\uparrow$ *naming certificate for the unit-bearing denominator fragment.*
2. *On carried inputs, the envelope restricts exactly to C_D, R_D, and native append through μ_D.*
3. *The endpoint e^*_D is carried by C^*_D but not by C_D, and no C_D endpoint is classified with e^*_D by R^*_D on either side.*
4. *The empty-unit classifier R^u_D has no mixed carried-empty endpoint branch.*
5. *Every envelope-carried one-sided unit endpoint is classified with e^*_D by R^*_D, while every carried denominator one-sided unit law for C_D, R_D, and μ_D is impossible.*

Thus the exported monoid object is the strict external-unit envelope, and the carried Rat denominator multiplication remains a unitless carried semigroup at the Field$^\uparrow$(RatUp) *boundary.*

Proof. The exported Monoid$^\uparrow$ naming certificate is exactly Definition 45.384. Its certificate fields are supplied by the envelope laws collected in Definition 45.383.

The carried restriction and strict externality are read from Definition 45.380. This theorem gives the recovery of C_D, R_D, and native append on carried inputs, records $C^*_D(e^*_D)$ with $\neg C_D(e^*_D)$, and excludes R^*_D-classification between any carried endpoint and e^*_D on both sides.

The empty-unit tuple has no mixed carried-empty classifier branch by Definition 45.382. That separation is the boundary check preventing the adjoined unit clause from classifying a carried denominator endpoint against an empty endpoint.

For uniqueness of an envelope-carried one-sided unit, apply Definition 45.378. It sends either one-sided envelope unit law to the classifier comparison with e^*_D.

Finally, the carried semigroup role and the carried-unit obstruction are the denominator-side clauses of Definition 45.369. Together these cited packages show that Field$^\uparrow$(RatUp) uses the unit-bearing Monoid$^\uparrow$ certificate only at the envelope layer, while the native carried denominator fragment keeps exactly its strict unitless boundary. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldUp StdBridge`

Traditional target: denominator monoid boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: RatHistoryCarrier, RatHistoryClassifier, Cont, append, e, and RatHistoryLedgerPolicy; **carrier choice**: C_D^* for the exported monoid envelope and C_D for the carried denominator fragment; **classifier choice**: R_D^* for the envelope boundary, R_D on carried inputs, and R_D^u for empty-unit endpoint separation; **stability obligation**: $\mathsf{Monoid}^{\uparrow}$ certificate laws, carried restriction, strict endpoint externality, mixed-endpoint separation, envelope unit uniqueness, and carried-unit obstruction; **ledger**: the Rat denominator continuation ledger together with the external-unit bit; **core theorem**: Definition 45.391; **exactness predicate**: the strict restriction of the envelope to (C_D, R_D, μ_D) and the absence of carried one-sided units; **standard bridge**: deferred; **status**: ProofTarget.

Theorem 45.392 (Rat denominator empty-unit commutative monoid laws)**.** *For the tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u)$ of Definition 45.346, the five laws of Definition 45.347 hold, and the multiplication is commutative up to the endpoint classifier: if $C_D^u(h)$ and $C_D^u(k)$, then*

$$R_D^u(\mu_D^u(h,k), \mu_D^u(k,h)).$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit continuation commutative monoid laws`

Proof. Unfold the concrete data of Definition 45.346. The first five clauses are precisely Definition 45.347.

It remains to prove the displayed commutativity classifier. Split the two RatDenomUnitCarrier witnesses into their empty endpoint and genuine Rat denominator endpoint alternatives.

In the genuine/genuine case, the two products are the canonical append continuations $\mathsf{Cont}(h, k, \mathsf{append}(h,k))$ and $\mathsf{Cont}(k, h, \mathsf{append}(k,h))$. Apply Definition 45.118 to these two continuations and the two Rat denominator carriers. The resulting RatHistoryClassifier supplies the carried endpoints and the hsame comparison required to pack $\mathsf{RatDenomUnitClassifier}(\mathsf{append}(h,k), \mathsf{append}(k,h))$.

In a mixed case, one endpoint is hsamee and the other is a genuine Rat denominator endpoint. The append unit law on the empty side identifies both products with the genuine endpoint, one directly and one after symmetry of hsame. Transitivity of hsame gives $\mathsf{append}(h,k)\mathsf{hsameappend}(k,h)$, and append closure gives the two empty-unit carrier witnesses. These data pack the required RatDenomUnitClassifier.

In the empty/empty case, both inputs are hsamee. The left and right append unit laws identify each product with e; symmetry and transitivity of hsame then identify the two products with each other. The empty branch of RatDenomUnitCarrier carries both products, so the classifier packing again yields $R_D^u(\mu_D^u(h,k), \mu_D^u(k,h))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit commutative monoid laws`

Theorem 45.393 (Rat denominator empty-unit identity uniqueness)**.** *For the tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u)$ of Definition 45.346, let $u : \mathsf{BHist}$ be a carried endpoint with $C_D^u(u)$. If u satisfies the two one-sided unit laws*

$$\forall h : \mathsf{BHist}.\ C_D^u(h) \to R_D^u(\mu_D^u(u,h), h)$$

and

$$\forall h : \mathsf{BHist}.\ C_D^u(h) \to R_D^u(\mu_D^u(h,u), h),$$

then

$$R_D^u(u, e_D^u).$$

Proof. Unpack Definition 45.347. Its carried-unit clause gives $C^u_D(e^u_D)$, and its right-unit law with right factor e^u_D gives $R^u_D(\mu^u_D(u, e^u_D), u)$ from the assumed carrier witness $C^u_D(u)$.

Now apply the assumed left-unit law for u at the carried endpoint e^u_D. This gives $R^u_D(\mu^u_D(u, e^u_D), e^u_D)$.

Project the two R^u_D classifier witnesses to their hsame components. The first comparison identifies $\mu^u_D(u, e^u_D)$ with u, and the second comparison identifies the same product with e^u_D. Symmetry and transitivity of hsame therefore give $u\mathsf{hsame}e^u_D$.

Finally, repack the carrier witnesses $C^u_D(u)$ and $C^u_D(e^u_D)$ together with this endpoint comparison. The result is exactly $R^u_D(u, e^u_D)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit identity unique`

Per-object template: traditional target identity uniqueness in the Rat denominator empty-unit monoid; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, append, and e; carrier C^u_D; classifier R^u_D; stability obligation concrete one-sided unit collapse; ledger external-unit endpoint provenance; core theorem Definition 45.393; exactness predicate unit endpoints classify with e^u_D; standard bridge deferred; status ProofTarget.

Definition 45.394 (Rat denominator empty-unit strict product support)**.** *For the tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ *of Definition 45.346, the strict product support predicate is the genuine Rat denominator endpoint predicate*

$$N^u_D(h) \iff \mathsf{RatHistoryCarrier}(h).$$

Thus the empty endpoint is carried by C^u_D *but is not in the strict support.*

Theorem 45.395 (Rat denominator unit carrier append factor carriers)**.** *For histories* $h, k : \mathsf{BHist}$*, if their append product is carried by the Rat denominator empty-unit carrier, then each factor is carried:*

$$C^u_D(\mathsf{append}(h, k)) \implies C^u_D(h) \wedge C^u_D(k).$$

Proof. Unfold C^u_D. In the empty branch, $\mathsf{append}(h, k)\mathsf{hsame}\mathsf{e}$, and append-empty inversion identifies both factors with e. In the Rat-history branch, positivity of the denominator product gives a unary, nonempty append history. Unary append factorization recovers unary witnesses for h and k; each unary factor is either e or an E1-headed positive denominator, hence a RatDenomUnitCarrier witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append factor carriers`

Theorem 45.396 (Rat denominator unit carrier append nonempty factor cases)**.** *For histories* $h, k : \mathsf{BHist}$*, a nonempty append product carried by* C^u_D *has a genuine Rat denominator endpoint on at least one factor, while both factors remain carried by* C^u_D*:*

$$C^u_D(\mathsf{append}(h, k)) \implies (\mathsf{append}(h, k)\mathsf{hsame}\mathsf{e} \implies \bot) \implies (\mathsf{RatHistoryCarrier}(h) \wedge C^u_D(k)) \vee (C^u_D(h) \wedge \mathsf{RatHistoryCarrier}($$

Proof. Apply append branch cases to the carried product. The both-empty branch would make the appended product empty by append-empty reconstruction, contradicting the nonempty hypothesis. In the remaining branches, append factor-carrier closure supplies the denominator-unit carrier witness for the opposite factor. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append nonempty factor cases`

Proposition 45.397 (Rat denominator unit carrier append exactness)**.** *For histories* $h, k : \mathsf{BHist}$*, the empty-unit Rat denominator carrier is closed and reflected by append:*

$$C^u_D(\mathsf{append}(h, k)) \iff C^u_D(h) \wedge C^u_D(k).$$

Proof. The forward direction is the append factor-carrier theorem above. Conversely, given carrier witnesses for both factors, apply denominator empty-unit continuation closure to the canonical continuation $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitCarrier append iff`

Theorem 45.398 (Rat denominator unit classifier append right factor classifier)**.** *For histories $h, h', k, k' : \mathsf{BHist}$, if the appended denominator-unit endpoints are classified and the right factors are internally the same, then the left factors are classified:*

$$R^u_D(\mathsf{append}(h, k), \mathsf{append}(h', k')) \Longrightarrow \mathsf{hsame}(k, k') \Longrightarrow R^u_D(h, h').$$

Proof. The classifier on the appended endpoints first supplies carrier witnesses for both products. Factor-carrier closure for C^u_D extracts carrier witnesses for h and h'. Transport the second appended endpoint along $\mathsf{hsame}(k, k')$ so that both products share the suffix k. Right cancellation for append then identifies h with h', and these three pieces reassemble the denominator-unit classifier on the left factors. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append right factor classifier`

Theorem 45.399 (Rat denominator empty-unit product nonempty iff)**.** *For histories $h, k : \mathsf{BHist}$ with $C^u_D(h)$ and $C^u_D(k)$, the empty-unit product is nonempty exactly when at least one input has strict Rat denominator support:*

$$(\mu^u_D(h, k)\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow N^u_D(h) \vee N^u_D(k).$$

Proof. Unfold C^u_D, μ^u_D, and N^u_D. The product $\mu^u_D(h, k)$ is $\mathsf{append}(h, k)$, with canonical continuation witness $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$.

For the forward direction, apply Definition 45.170 to the two $\mathsf{RatDenomUnitCarrier}$ witnesses and the canonical continuation. The assumption that $\mathsf{append}(h, k)$ is not hsame to e is exactly the nonempty result condition, so the theorem returns $\mathsf{RatHistoryCarrier}(h) \vee \mathsf{RatHistoryCarrier}(k)$.

For the reverse direction, split the strict-support disjunction. In the left case, split the carried endpoint k into its empty and genuine alternatives. If $k\mathsf{hsamee}$, the right append unit law identifies $\mathsf{append}(h, k)$ with h, so any empty product would make $h\mathsf{hsamee}$, contradicting $\mathsf{RatHistoryCarrier_not_empty}$. If $\mathsf{RatHistoryCarrier}(k)$, apply Rat denominator continuation result nonemptiness to the canonical continuation.

In the right strict-support case, split the carried endpoint h in the same way. If $h\mathsf{hsamee}$, the left append unit law identifies $\mathsf{append}(h, k)$ with k, and $\mathsf{RatHistoryCarrier_not_empty}$ rejects an empty product. If $\mathsf{RatHistoryCarrier}(h)$, the same Rat denominator continuation result nonemptiness theorem applies to the canonical continuation. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit product nonempty iff`

Theorem 45.400 (Rat denominator empty-unit product nonempty comm iff)**.** *For histories $h, k : \mathsf{BHist}$ with $C^u_D(h)$ and $C^u_D(k)$, reversing the two factors preserves the nonempty endpoint predicate of the empty-unit product:*

$$(\mathsf{append}(h, k)\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow (\mathsf{append}(k, h)\mathsf{hsamee} \to \mathsf{False}).$$

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit product nonempty comm iff`

Proof. Apply Definition 45.399 to the ordered pair (h, k) and again to the reversed pair (k, h). The first characterization reads nonemptiness as $\mathsf{RatHistoryCarrier}(h) \vee \mathsf{RatHistoryCarrier}(k)$, and the second reads it as the same disjunction with the two sides exchanged. Split the factor-support disjunction and swap the branch tags. □

Theorem 45.401 (Rat denominator empty-unit strict product factor cases)**.** *For histories $h, k : \mathsf{BHist}$ with $C^u_D(h)$ and $C^u_D(k)$, if the empty-unit product has strict Rat denominator support, then one input has strict support while the other remains carried:*

$$N^u_D(\mu^u_D(h, k)) \Longrightarrow (N^u_D(h) \wedge C^u_D(k)) \vee (C^u_D(h) \wedge N^u_D(k)).$$

Proof. Strict support of the product rules out $\mathsf{hsame}(\mathsf{append}(h, k), \mathsf{e})$ by RatHistoryCarrier_not_empty. Apply the product nonempty characterization above to obtain $N^u_D(h) \vee N^u_D(k)$, then reattach the original carrier hypotheses on the other factor in each case. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit strict product factor cases`

Theorem 45.402 (Rat denominator empty-unit strict product support exactness)**.** *For histories $h, k : \mathsf{BHist}$ with $C^u_D(h)$ and $C^u_D(k)$, strict support is exact for the empty-unit product:*

$$N^u_D(\mu^u_D(h, k)) \Longleftrightarrow N^u_D(h) \vee N^u_D(k).$$

Proof. Unfold N^u_D. The forward direction starts from $\mathsf{RatHistoryCarrier}(\mu^u_D(h, k))$. By RatHistoryCarrier_not_empty the product cannot be hsamee. Apply Definition 45.399 to this nonempty product witness and obtain $N^u_D(h) \vee N^u_D(k)$.

For the reverse direction, assume $N^u_D(h) \vee N^u_D(k)$. The reverse direction of Definition 45.399 gives that $\mu^u_D(h, k)$ is not hsamee.

It remains to recover the genuine Rat carrier for the product. The closure clause of Definition 45.347 applied to $C^u_D(h)$ and $C^u_D(k)$ gives $C^u_D(\mu^u_D(h, k))$. Apply RatDenomUnitCarrier_nonempty_rat to this carried-product witness and the nonempty product witness. The result is $\mathsf{RatHistoryCarrier}(\mu^u_D(h, k))$, equivalently $N^u_D(\mu^u_D(h, k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit strict product support exactness`

Per-object template: traditional target strict support for the empty-unit denominator multiplication; BEDC source RatHistoryCarrier, RatDenomUnitCarrier, and Cont; carrier C^u_D; classifier R^u_D; stability obligation product support exactness; ledger continuation witness; core theorem Definition 45.402; exactness predicate strict product support iff genuine denominator input; standard bridge ordinary denominator monoid support; status ProofTarget.

Definition 45.403 (Rat denominator empty-unit multiplicative substrate)**.** *The denominator-side multiplicative substrate for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the concrete* BHist *tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *with carrier* RatDenomUnitCarrier*, classifier* RatDenomUnitClassifier*, multiplication* $\mu^u_D(h, k) := \mathsf{append}(h, k)$ *equipped with the canonical continuation* $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$*, unit* $e^u_D := \mathsf{e}$*, and strict support* $N^u_D(h)$ *given by* $\mathsf{RatHistoryCarrier}(h)$*. Thus the denominator schema instance is the concrete append substrate over* BHist*, with the empty endpoint adjoined only as the unit and not as an uninterpreted multiplication input.*

Theorem 45.404 (Rat denominator empty-unit multiplicative substrate laws)**.** *For the substrate* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *of Definition 45.403, the following concrete laws hold.*

1. *The unit is carried:* $C^u_D(e^u_D)$*.*

2. *The carrier is closed under multiplication: if* $C^u_D(h)$ *and* $C^u_D(k)$*, then* $C^u_D(\mu^u_D(h,k))$*.*
3. *The unit laws hold: if* $C^u_D(h)$*, then* $R^u_D(\mu^u_D(e^u_D,h),h)$ *and* $R^u_D(\mu^u_D(h,e^u_D),h)$*.*
4. *Multiplication is associative up to the classifier: if* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, then* $R^u_D(\mu^u_D(\mu^u_D(h,k),\ell),\mu^u_D(h,\mu^u_D$
5. *The operation respects binary classifier input: if* $R^u_D(h,h')$ *and* $R^u_D(k,k')$*, then* $R^u_D(\mu^u_D(h,k),\mu^u_D(h',k'))$*.*
6. *Multiplication is commutative up to the classifier: if* $C^u_D(h)$ *and* $C^u_D(k)$*, then* $R^u_D(\mu^u_D(h,k),\mu^u_D(k,h))$*.*
7. *Strict support is exact for products: if* $C^u_D(h)$ *and* $C^u_D(k)$*, then*

$$(\mu^u_D(h,k)\mathsf{hsamee} \to \mathsf{False}) \iff N^u_D(h) \vee N^u_D(k).$$

8. *Empty endpoint contexts preserve and reflect the classifier: if* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then*

$$R^u_D(\mathsf{append}(p,\mathsf{append}(h,q)),\mathsf{append}(p',\mathsf{append}(k,q'))) \iff R^u_D(h,k).$$

Proof. Unfold the concrete substrate of Definition 45.403. The carried unit, closure, left and right unit laws, associativity, and binary classifier congruence are the five laws supplied by Definition 45.347.

Commutativity is the additional classifier law of Definition 45.392; it uses the same carried endpoints and the concrete append multiplication of the substrate.

The strict support clause is exactly Definition 45.399, because N^u_D is $\mathsf{RatHistoryCarrier}$ and the product is $\mathsf{append}(h,k)$ with its canonical continuation witness.

For the empty-context classifier clause, replace R^u_D by $\mathsf{RatDenomUnitClassifier}$ and apply Definition 45.405. This proves preservation and reflection for the same concrete classifier used by the substrate. □

Theorem 45.405 (Rat denominator empty-unit classifier empty-context exactness)**.** *For histories* $p,q,p',q',h,k : \mathsf{BHist}$*, if* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then empty endpoint contexts do not change the empty-unit denominator classifier:*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(p,\mathsf{append}(h,q)),\mathsf{append}(p',\mathsf{append}(k,q'))) \iff \mathsf{RatDenomUnitClassifier}(h,k).$$

Proof. Unfold $\mathsf{RatDenomUnitClassifier}$ into the two $\mathsf{RatDenomUnitCarrier}$ endpoints and the endpoint comparison by hsame. The carrier endpoints are handled by Definition 45.348, once for (p,q,h) and once for (p',q',k).

For the forward direction, the contextual classifier supplies the two contextual carrier witnesses and the comparison between the contextual endpoints. The two carrier equivalences transport the carrier witnesses back to h and k. The empty left and right append unit laws identify $\mathsf{append}(p,\mathsf{append}(h,q))$ with h and $\mathsf{append}(p',\mathsf{append}(k,q'))$ with k; composing these identifications by symmetry and transitivity with the contextual comparison gives $h\mathsf{hsame}k$. These three pieces repack $\mathsf{RatDenomUnitClassifier}(h,k)$.

For the reverse direction, start from the carrier witnesses for h and k and from $h\mathsf{hsame}k$. The same two carrier equivalences transport membership to the contextual endpoints. The empty append identifications transport $h\mathsf{hsame}k$ to the contextual endpoint comparison, again using symmetry and transitivity of hsame. The transported carriers and transported endpoint comparison pack the contextual classifier.

This is not a variant of the carrier exactness theorem: it proves full classifier preservation by combining carrier exactness at both endpoints with transport of the endpoint hsame relation.

Consequently the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator envelope can read an empty external unit context by the normalized classifier on the core endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier empty context iff`

Theorem 45.406 (Rat denominator empty-unit contextual strict product support exactness)**.** *For histories $p, q, h, k : \mathsf{BHist}$, if $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $C^u_D(h)$, and $C^u_D(k)$, then empty external unit contexts read strict denominator support through the product endpoint:*

$$N^u_D(\mathsf{append}(p, \mathsf{append}(\mu^u_D(h,k), q))) \iff N^u_D(h) \vee N^u_D(k).$$

Proof. The closure clause of Definition 45.347 applied to $C^u_D(h)$ and $C^u_D(k)$ gives $C^u_D(\mu^u_D(h,k))$. The right comparison endpoint is the uncontextualized product $\mu^u_D(h,k)$, and its right empty context is supplied by $\mathsf{ehsamee}$ on both sides.

Apply Definition 45.405 with left contexts p, q and right contexts e, e. This produces the classifier equivalence between $\mathsf{append}(p, \mathsf{append}(\mu^u_D(h,k), q))$ and $\mu^u_D(h,k)$ in the concrete $\mathsf{RatDenomUnitClassifier}$ envelope.

Unfold N^u_D as $\mathsf{RatHistoryCarrier}$. The exposed hsame component of the classifier transports genuine Rat carrier support between the contextual product endpoint and the uncontextualized product endpoint.

Finally apply Definition 45.402 to the carried inputs h and k. Its product-support equivalence gives $N^u_D(\mu^u_D(h,k)) \iff N^u_D(h) \vee N^u_D(k)$, and the transport above converts the left side to the contextual endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit contextual strict product support exactness`

Theorem 45.407 (Rat denominator empty-unit contextual product classifier exactness)**.** *For histories $p, q, p', q', h, k, h', k' : \mathsf{BHist}$, assume $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$. If the empty-unit denominator classifier relates h with h' and k with k', then the contextual rational classifier on the product endpoints is equivalent to strict rational support of the exposed product:*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(p, \mathsf{append}(\mathsf{append}(h,k), q)), \mathsf{append}(p', \mathsf{append}(\mathsf{append}(h',k'), q'))) \iff \mathsf{RatHistoryCarrie}$$

Proof. The empty left and right contexts identify the two contextual endpoints with the exposed products. Binary classifier congruence for $\mathsf{RatDenomUnitClassifier}$ classifies $\mathsf{append}(h,k)$ with $\mathsf{append}(h',k')$. In the forward direction, rational support transports out of the left empty context. In the reverse direction, the product classifier transports rational support to the right product, and the rational classifier is then transported back into the two displayed empty contexts. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier contextual product rat classifier iff`

Theorem 45.408 (Rat denominator empty-unit threefold strict support exactness)**.** *For histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, strict support is exact for the left-nested empty-unit threefold product:*

$$N^u_D(\mu^u_D(\mu^u_D(h,k), \ell)) \iff (N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell).$$

Proof. The closure clause of Definition 45.347 applied to $C^u_D(h)$ and $C^u_D(k)$ gives $C^u_D(\mu^u_D(h,k))$. This supplies the carrier hypothesis needed to read the outer product of $\mu^u_D(h,k)$ with ℓ.

Apply Definition 45.402 to the carried histories $\mu^u_D(h,k)$ and ℓ. The result is

$$N^u_D(\mu^u_D(\mu^u_D(h,k), \ell)) \iff N^u_D(\mu^u_D(h,k)) \vee N^u_D(\ell).$$

Apply the same theorem to the original carried histories h and k, giving $N^u_D(\mu^u_D(h,k)) \iff N^u_D(h) \vee N^u_D(k)$. Substitute this inner equivalence into the left disjunct of the outer equivalence.

In the forward direction, the outer equivalence gives either $N^u_D(\mu^u_D(h,k))$ or $N^u_D(\ell)$. The first case is transported by the inner equivalence to $N^u_D(h) \vee N^u_D(k)$, while the second case gives the right disjunct directly.

In the reverse direction, a witness of $(N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell)$ is handled by cases. The left case is transported by the reverse direction of the inner equivalence to $N^u_D(\mu^u_D(h,k))$; the right case is already $N^u_D(\ell)$. The reverse direction of the outer equivalence then yields $N^u_D(\mu^u_D(\mu^u_D(h,k),\ell))$. □

Per-object template: traditional target finite product support readback for denominator multiplication; BEDC source BHist with $C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D$; carrier C^u_D; classifier R^u_D; stability obligation binary product support exactness plus carrier closure; ledger two continuation-product witnesses; exactness predicate strict support of the left-nested threefold product iff support of one factor; standard bridge ordinary finite denominator product support; status ProofTarget.

Per-object template: traditional target empty-context denominator product support readback; BEDC source BHist; carrier C^u_D; classifier R^u_D; stability obligation contextual strict product support exactness; ledger empty external unit contexts around the product endpoint; core theorem Definition 45.406; exactness predicate strict support is unchanged by empty external product contexts; standard bridge ordinary denominator multiplication support; status ProofTarget.

45.18 Rat denominator empty-unit cancellation exactness

Definition 45.409 (Rat denominator empty-unit cancellation-exact instance)**.** *The cancellation-exact denominator instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the concrete* BHist *tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, where* $C^u_D = \mathsf{RatDenomUnitCarrier}$*,* $R^u_D = \mathsf{RatDenomUnitClassifier}$*,* $\mu^u_D(h,k) = \mathsf{append}(h,k)$ *with its canonical continuation witness,* $e^u_D = \mathsf{e}$*, and* $N^u_D(h)$ *is* $\mathsf{RatHistoryCarrier}(h)$*.*

Theorem 45.410 (Rat denominator empty-unit cancellation exactness)**.** *For carried histories* $h, k, t : \mathsf{BHist}$ *in the instance of Definition 45.409, classifier cancellation by the common carried factor is exact on both sides:*

$$R^u_D(\mu^u_D(h,t), \mu^u_D(k,t)) \iff R^u_D(h,k),$$

and

$$R^u_D(\mu^u_D(t,h), \mu^u_D(t,k)) \iff R^u_D(h,k).$$

Proof. For the two forward directions, use the carried hypotheses for h, k, and t and apply Definition 45.431. Its right-cancellation field gives the first implication, and its left-cancellation field gives the second implication.

For the reverse direction on the right, start from $R^u_D(h,k)$. Reflexivity for the carried factor gives $R^u_D(t,t)$, and the binary classifier congruence field in Definition 45.392 combines the two classifier witnesses to obtain $R^u_D(\mu^u_D(h,t), \mu^u_D(k,t))$.

For the reverse direction on the left, use the same reflexivity witness $R^u_D(t,t)$ as the first classifier input and the assumed $R^u_D(h,k)$ as the second classifier input. The same binary congruence field then gives $R^u_D(\mu^u_D(t,h), \mu^u_D(t,k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit right multiplication classifier exactness`

Per-object template: traditional target cancellation exactness in the denominator monoid boundary; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append; carrier C^u_D; classifier R^u_D; stability obligation two-sided classifier preservation and reflection; ledger the common carried denominator factor; core theorem Definition 45.410; exactness predicate classifier iff after common denominator multiplication; standard bridge cancellative commutative denominator monoids; status ProofTarget.

This is a concrete BHist tuple law for future Lean rounds, not an abstract operator schema.

Definition 45.411 (Rat denominator empty-unit right multiplication context)**.** *For a carried context t : BHist with $C^u_D(t)$, define the concrete right multiplication map on the empty-unit denominator carrier by*

$$R_t(h) := \mu^u_D(h, t).$$

The carrier, classifier, and multiplication are the same concrete BHist *data used in Definition 45.409.*

Definition 45.412 (Rat denominator empty-unit left multiplication context)**.** *For a carried context t : BHist with $C^u_D(t)$, define the concrete left multiplication map on the empty-unit denominator carrier by*

$$L_t(h) := \mu^u_D(t, h).$$

The carrier, classifier, and multiplication are the same concrete BHist *data used in Definition 45.346.*

Theorem 45.413 (Rat denominator empty-unit left multiplication classifier exactness)**.** *For histories h, k, t : BHist, if $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(t)$, then left multiplication by t preserves and reflects the empty-unit denominator classifier:*

$$R^u_D(\mu^u_D(t, h), \mu^u_D(t, k)) \Longleftrightarrow R^u_D(h, k).$$

Proof. For the forward direction, use the left-cancellation clause packaged in Definition 45.431 with the three carrier hypotheses. It reads the displayed comparison after common left multiplication by t back as the endpoint classifier $R^u_D(h, k)$.

For the reverse direction, use $C^u_D(t)$ to obtain reflexivity $R^u_D(t, t)$. The binary classifier congruence field in Definition 45.347, or the same congruence field as packaged in Definition 45.431, combines $R^u_D(t, t)$ with $R^u_D(h, k)$ and yields $R^u_D(\mu^u_D(t, h), \mu^u_D(t, k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit left multiplication classifier exactness`

Per-object template: traditional target left multiplication by a fixed denominator-unit factor; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append; carrier C^u_D; classifier R^u_D; stability obligation left-context classifier exactness; ledger fixed left-context continuation witness; core theorem Definition 45.413; exactness predicate preservation and reflection of the classifier under L_t; standard bridge cancellative commutative denominator monoids; status ProofTarget.

This is the left-context companion to the right-cancellation law Definition 45.432 inside the concrete empty-unit denominator boundary.

Theorem 45.414 (Rat history classifier append left denominator unit closure)**.** *For histories h, h', k, k' : BHist, if* RatDenomUnitClassifier(h, h') *and* RatHistoryClassifier(k, k')*, then*

$$\mathsf{RatHistoryClassifier}(\mathsf{append}(h, k), \mathsf{append}(h', k')).$$

Proof. The denominator-unit carrier on h and the Rat carrier on k close the left appended endpoint as a Rat history carrier; the same argument closes the right endpoint from h' and k'. The classifier comparisons for the two factors are respected by the canonical append continuations, giving the required history sameness between the appended endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatHistoryClassifier append left denom unit closed`

Theorem 45.415 (Rat denominator unit classifier append-context cancellation)**.** *For histories L, R, Q, S : BHist, if $R^u_D(L, R)$, then appending the classified suffixes L and R preserves and reflects the empty-unit denominator classifier:*

$$R^u_D(\mathsf{append}(Q, L), \mathsf{append}(S, R)) \iff R^u_D(Q, S).$$

Proof. For the forward direction, unpack the displayed classifier. Product-carrier factorization gives carried witnesses for Q and S. The suffix classifier gives $L\mathsf{hsame}R$, so the right displayed endpoint can be transported to the common suffix L; right append cancellation then gives $Q\mathsf{hsame}S$.

For the reverse direction, classifier closure under continuation appends the two carried suffix witnesses to the two carried endpoints. The endpoint classifier and suffix classifier identify the two appended histories. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append context cancel iff`

Theorem 45.416 (Rat denominator unit classifier append left context cancellation)**.** *For histories L, R, Q, S : BHist, if $\mathsf{RatDenomUnitClassifier}(L, R)$, then appending the classified prefixes L and R preserves and reflects the empty-unit denominator classifier:*

$$\mathsf{RatDenomUnitClassifier}(\mathsf{append}(L, Q), \mathsf{append}(R, S)) \iff \mathsf{RatDenomUnitClassifier}(Q, S).$$

Proof. For the forward direction, factor the two appended carrier witnesses to recover carried witnesses for Q and S. Transport the right displayed endpoint along $R\mathsf{hsame}L$ so both endpoints have common prefix L, then apply left append cancellation to obtain $Q\mathsf{hsame}S$.

For the reverse direction, close the two appended endpoints by denominator-unit continuation closure. Continuation respect for the classified prefixes and classified suffixes gives the endpoint comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitClassifier append left context cancel iff`

Definition 45.417 (Rat denominator empty-unit bilateral multiplication context)**.** *For carried contexts l, r : BHist with $C^u_D(l)$ and $C^u_D(r)$, define the concrete two-sided carried-context map over the empty-unit denominator carrier by*

$$B_{l,r}(h) := \mu^u_D(\mu^u_D(l, h), r).$$

Here the carrier is C^u_D, the classifier is R^u_D, and multiplication is the concrete operation $\mu^u_D(h, k) := \mathsf{append}(h, k)$ from Definition 45.346.

Theorem 45.418 (Rat denominator empty-unit bilateral multiplication classifier exactness)**.** *For histories h, k, l, r : BHist, if $C^u_D(h)$, $C^u_D(k)$, $C^u_D(l)$, and $C^u_D(r)$, then the two-sided carried-context map of Definition 45.417 preserves and reflects the empty-unit denominator classifier:*

$$R^u_D(B_{l,r}(h), B_{l,r}(k)) \iff R^u_D(h, k).$$

Proof. First use carrier closure from Definition 45.347. The carried hypotheses for l, h, and k give carried witnesses for $\mu^u_D(l,h)$ and $\mu^u_D(l,k)$; together with $C^u_D(r)$, they put both right-multiplied endpoints inside the scope of the denominator cancellation exactness theorem.

For the forward direction, unfold $B_{l,r}$ to read the assumed comparison as the right multiplication comparison of $\mu^u_D(l,h)$ and $\mu^u_D(l,k)$ by the common carried context r. Apply Definition 45.410; its reflection direction gives $R^u_D(\mu^u_D(l,h), \mu^u_D(l,k))$.

Now apply Definition 45.413 with the carried left context l. Its reflection direction converts the last classifier witness into $R^u_D(h,k)$.

For the reverse direction, begin with $R^u_D(h,k)$. The preservation direction of Definition 45.413 gives $R^u_D(\mu^u_D(l,h), \mu^u_D(l,k))$.

Apply the right-hand preservation direction of Definition 45.410 with the common carried context r. After folding the definition of $B_{l,r}$, the result is exactly $R^u_D(B_{l,r}(h), B_{l,r}(k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit bilateral multiplication classifier exactness`

Per-object template: traditional target bilateral denominator-context exactness; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append; carrier C^u_D; classifier R^u_D; stability obligation two-sided carried-context classifier exactness; ledger the two carried context continuation witnesses; core theorem Definition 45.418; exactness predicate $R^u_D(B_{l,r}(h), B_{l,r}(k)) \iff R^u_D(h,k)$; standard bridge cancellative commutative denominator monoids; status ProofTarget.

Theorem 45.419 (Rat denominator empty-unit context action support exactness)**.** *For histories* $h, k, t, p, q, p', q' : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(t)$*, and* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*. The concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *satisfies the following context-action boundary laws.*

1. *Right multiplication by the fixed factor* t *is classifier exact:*

$$R^u_D(\mu^u_D(h,t), \mu^u_D(k,t)) \iff R^u_D(h,k).$$

2. *Left multiplication by the fixed factor* t *is classifier exact:*

$$R^u_D(\mu^u_D(t,h), \mu^u_D(t,k)) \iff R^u_D(h,k).$$

3. *Empty external endpoint contexts are classifier exact:*

$$R^u_D(\mathsf{append}(p, \mathsf{append}(h,q)), \mathsf{append}(p', \mathsf{append}(k,q'))) \iff R^u_D(h,k).$$

4. *Empty external endpoint contexts preserve and reflect strict support:*

$$N^u_D(\mathsf{append}(p, \mathsf{append}(h,q))) \iff N^u_D(h)$$

and

$$N^u_D(\mathsf{append}(p', \mathsf{append}(k,q'))) \iff N^u_D(k).$$

5. *Strict support is exact under the two fixed-factor multiplication actions:*

$$N^u_D(\mu^u_D(h,t)) \iff N^u_D(h) \vee N^u_D(t)$$

and

$$N^u_D(\mu^u_D(t,h)) \iff N^u_D(t) \vee N^u_D(h).$$

Proof. The first classifier equivalence is exactly Definition 45.410 applied to the carried histories h, k, and t.

The second classifier equivalence is exactly Definition 45.413 with the same three carried endpoints.

For the empty external contexts, unfold R^u_D as RatDenomUnitClassifier and apply Definition 45.405 to the four endpoint equalities $p$hsamee, $q$hsamee, p'hsamee, and q'hsamee.

For strict support under empty contexts, unfold N^u_D as RatHistoryCarrier. The endpoint equalities identify $\mathsf{append}(p, \mathsf{append}(h, q))$ with h and $\mathsf{append}(p', \mathsf{append}(k, q'))$ with k by the same empty-context endpoint equivalence used in Definition 45.348. Transport RatHistoryCarrier along those hsame comparisons in both directions.

The two fixed-factor support equivalences are the two orientations of Definition 45.402, first for (h, t) and then for (t, h). □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit context action support exactness`

Theorem 45.420 (Rat denominator contextual action empty-context classifier transport)**.** *For histories h, h', l, r : BHist, if $C^u_D(l)$ and $C^u_D(r)$ hold and $R^u_D(h, h')$, then the bilateral empty-context action preserves the denominator classifier:*

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},l,r}(h), \Gamma_{\mathsf{e},\mathsf{e},l,r}(h')).$$

Proof. Classify l with itself and r with itself by reflexivity of hsame. Apply continuation closure of R^u_D first to the left product $\mathsf{append}(l, h)$ versus $\mathsf{append}(l, h')$, and then to the right product with r. The empty external contexts are removed by the denominator unit classifier empty-context equivalence in both directions. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty context hsame transport`

Per-object template: traditional target denominator context action for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, append, and e; carrier C^u_D; classifier R^u_D; stability obligations classifier exactness and support exactness; ledger fixed factor and empty external contexts; core theorem Definition 45.419; exactness predicate preservation and reflection of classifier plus strict support; standard bridge cancellative denominator monoids with adjoined unit; status ProofTarget.

Theorem 45.421 (Rat denominator empty-unit bilateral multiplication support exactness)**.** *For histories h, l, r : BHist with $C^u_D(h)$, $C^u_D(l)$, and $C^u_D(r)$, the bilateral context map of Definition 45.417 satisfies*

$$N^u_D(B_{l,r}(h)) \iff N^u_D(l) \vee N^u_D(h) \vee N^u_D(r).$$

Proof. Unfold $B_{l,r}$ as $\mu^u_D(\mu^u_D(l, h), r)$. Carrier closure for the empty-unit denominator product gives $C^u_D(\mu^u_D(l, h))$ from the carried hypotheses on l and h, so the outer product lies in the scope of strict product support exactness.

Apply Definition 45.402 to $\mu^u_D(\mu^u_D(l, h), r)$. This identifies $N^u_D(B_{l,r}(h))$ with $N^u_D(\mu^u_D(l, h)) \vee N^u_D(r)$.

Apply the same strict-product support exactness theorem to the inner product $\mu^u_D(l, h)$. This identifies the inner support term with $N^u_D(l) \vee N^u_D(h)$.

For the forward direction, combine the two displayed equivalences. If the outer support witness lands in the right component, obtain $N^u_D(r)$. If it lands in the left component, use the inner equivalence and distinguish the l and h cases. Reassociate the resulting constructive disjunction as $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$.

For the reverse direction, start from $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$. The r case supplies the right component of the outer strict-product equivalence. The l and h cases first supply $N^u_D(\mu^u_D(l,h))$ by the reverse direction of the inner equivalence, and then supply $N^u_D(B_{l,r}(h))$ by the reverse direction of the outer equivalence. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit bilateral multiplication support exactness`

Per-object template: traditional target bilateral denominator-context support exactness; BEDC source RatDenomUnitCarrier, RatHistoryCarrier, Cont, append, and e; carrier C^u_D; classifier R^u_D; stability obligation strict-support exactness under two-sided carried contexts; ledger the two continuation witnesses; core theorem Definition 45.421; exactness predicate $N^u_D(B_{l,r}(h)) \iff N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$; standard bridge cancellative denominator monoids with adjoined unit; status ProofTarget.

Definition 45.422 (Rat denominator empty-unit contextual commutative instance)**.** *The concrete empty-unit contextual commutative instance over* BHist *uses the carrier* $C^u_D :=$ RatDenomUnitCarrier*, the classifier* $R^u_D :=$ RatDenomUnitClassifier*, multiplication* $\mu^u_D(h,k) :=$ $\mathsf{append}(h,k)$*, and unit* $e^u_D :=$ e*. Its admissible external left and right contexts are exactly the histories history-same to the empty endpoint, written* $p$hsamee *and* $q$hsamee*.*

Theorem 45.423 (Rat denominator empty-unit contextual commutativity)**.** *For histories* $p, q, p', q', h, k :$ BHist*, if* $p$hsamee*,* $q$hsamee*,* p'hsamee*,* q'hsamee*,* $C^u_D(h)$*, and* $C^u_D(k)$*, then denominator multiplication commutes inside the admissible empty external contexts:*

$$R^u_D(\mathsf{append}(p, \mathsf{append}(\mu^u_D(h,k), q)), \mathsf{append}(p', \mathsf{append}(\mu^u_D(k,h), q'))).$$

Proof. Apply Definition 45.392 to the two carrier hypotheses $C^u_D(h)$ and $C^u_D(k)$. The commutativity clause gives the core classifier

$$R^u_D(\mu^u_D(h,k), \mu^u_D(k,h)).$$

Now use the reverse direction of Definition 45.405 with the four endpoint hypotheses $p$hsamee, $q$hsamee, p'hsamee, and q'hsamee. It transports the core classifier through the two empty left contexts and the two empty right contexts.

Since the contextual instance of Definition 45.422 has $\mu^u_D(x,y) = \mathsf{append}(x,y)$ and $R^u_D =$ RatDenomUnitClassifier, the transported classifier is exactly the displayed empty-context commutativity statement. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit contextual commutativity`

Theorem 45.424 (Rat denominator external-unit envelope commutative monoid laws)**.** *For the tuple* $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ *of Definition 45.370, all laws of Definition 45.377 hold, and the envelope multiplication is commutative up to its endpoint classifier: whenever* $C^*_D(h)$ *and* $C^*_D(k)$*,*

$$R^*_D(\mu^*_D(h,k), \mu^*_D(k,h)).$$

Proof. The carrier inhabitance, closure, associativity, left and right unit laws, binary classifier congruence, carried restriction, and carried one-sided unit obstruction are exactly the fields supplied by Definition 45.377.

It remains to prove the displayed commutativity classifier. Split the two carrier witnesses $C^*_D(h)$ and $C^*_D(k)$ into their carried Rat denominator alternatives and their external-empty alternatives.

In the carried/carried case, $\mu^*_D(h,k)$ and $\mu^*_D(k,h)$ are the canonical append continuations of the two carried denominators in opposite orders. Apply Definition 45.118 to these two continuations. The carried classifier exactness theorem Definition 45.379 then repacks the resulting RatHistoryClassifier as the required R^*_D comparison.

In the carried/empty case, the append right-unit and left-unit laws identify $\mu^*_D(h,k)$ and $\mu^*_D(k,h)$ with the same carried endpoint h, up to hsame. The envelope classifier normal form, Definition 45.371, turns those two endpoint comparisons and the transported carried witness into $R^*_D(\mu^*_D(h,k), \mu^*_D(k,h))$.

The empty/carried case is symmetric. The same append unit laws identify the two products with the carried endpoint k, and the same normal form converts the two transported carrier witnesses and the resulting hsame comparison into the envelope classifier.

In the empty/empty case, both products are hsame to e by the append unit laws and transitivity of hsame. Since $e^*_D = \mathsf{e}$, the empty branch of R^*_D classifies the two products directly. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope commutative monoid laws`

Per-object template: traditional target commutative denominator monoid boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source RatHistoryCarrier, RatHistoryClassifier, Cont, append, and e; carrier C^*_D; classifier R^*_D; stability obligation monoid laws plus commutativity; ledger external-unit bit plus continuation witness; core theorem Definition 45.424; exactness predicate envelope commutativity over the strict carried restriction; standard bridge deferred; status ProofTarget.

Definition 45.425 (Rat denominator external-unit envelope strict support)**.** *For the tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ of Definition 45.370, the strict support predicate is the genuine Rat denominator endpoint predicate*

$$N^*_D(h) \iff \mathsf{RatHistoryCarrier}(h).$$

Thus the external endpoint is carried by C^*_D but does not enter the strict denominator support.

Theorem 45.426 (Rat denominator external-unit envelope strict support laws)**.** *For histories $h, k : \mathsf{BHist}$ with $C^*_D(h)$ and $C^*_D(k)$, the strict support predicate of Definition 45.425 satisfies:*

1. $R^*_D(h,k) \to (N^*_D(h) \iff N^*_D(k))$.
2. $N^*_D(\mu^*_D(h,k)) \iff N^*_D(h) \vee N^*_D(k)$.
3. *Together with Definition 45.424, these clauses give a concrete* BHist *multiplicative-support boundary for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.

Proof. Unfold C^*_D, R^*_D, μ^*_D, and N^*_D. The carrier assumptions split each input into a genuine Rat denominator endpoint or an external empty endpoint.

For support transport along the classifier, split the defining alternatives of $R^*_D(h,k)$. In the guarded carried branch, both strict-support predicates are exactly the displayed RatHistoryCarrier witnesses, so the two directions are projections. In the empty-empty branch, any strict-support witness for either endpoint contradicts RatHistoryCarrier_not_empty after composing its endpoint comparison with the branch comparison to e.

For product support, first consider the carried/carried branch. The canonical continuation witness for $\mathsf{append}(h,k)$ and carried denominator closure give $\mathsf{RatHistoryCarrier}(\mu^*_D(h,k))$, while the right side holds by either input support.

In the carried/empty branch, the append right-unit law identifies $\mu^*_D(h,k)$ with h. Transport $\mathsf{RatHistoryCarrier}(h)$ across this hsame comparison by RatHistoryCarrier_hsame_transport, and the left disjunct supplies the reverse implication. The empty/carried branch is symmetric, using the append left-unit law and the right strict-support disjunct.

In the empty/empty branch, the append unit laws identify $\mu^*_D(h,k)$ with e. Therefore product strict support is impossible by RatHistoryCarrier_not_empty, and both input strict-support predicates are impossible by the same theorem. This proves the displayed product equivalence.

The final clause only packages these two support laws with the already stated commutative monoid laws: the monoid theorem supplies the algebraic envelope, and the present theorem supplies the exact descent of genuine denominator support through classifier transport and multiplication. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope strict support laws`

Theorem 45.427 (Rat denominator external-unit envelope product strict-support congruence)**.** *For histories $h, h', k, k' : \mathsf{BHist}$ with $C^*_D(h)$, $C^*_D(h')$, $C^*_D(k)$, $C^*_D(k')$, $R^*_D(h,h')$, and $R^*_D(k,k')$, the products have equivalent strict support:*

$$N^*_D(\mu^*_D(h,k)) \Longleftrightarrow N^*_D(\mu^*_D(h',k')).$$

Proof. First build the product endpoints inside the envelope. The closure clauses of Definition 45.377 applied to $C^*_D(h)$ and $C^*_D(k)$ give a carrier witness for $\mu^*_D(h,k)$, and the same clauses applied to $C^*_D(h')$ and $C^*_D(k')$ give a carrier witness for $\mu^*_D(h',k')$.

Next compare the two products. The binary classifier-congruence clause of Definition 45.377, applied to $R^*_D(h,h')$ and $R^*_D(k,k')$, gives $R^*_D(\mu^*_D(h,k), \mu^*_D(h',k'))$.

Finally transport strict support across that product classifier. The classifier-transport clause of Definition 45.426, used with the two product carrier witnesses, yields the displayed equivalence of N^*_D for the two products. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope product strict support congruence`

Per-object template: traditional target product strict-support congruence for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator envelope; BEDC source BHist, RatHistoryCarrier, R^*_D, μ^*_D, and append; carrier C^*_D; classifier R^*_D; stability obligation factorwise classifier congruence; ledger external-unit bit plus carried continuation witness; core theorem Definition 45.427; exactness predicate product strict support invariant under factorwise classification; standard bridge deferred; status ProofTarget.

Theorem 45.428 (Rat denominator external-unit envelope support-unit no-confusion)**.** *For every history $h : \mathsf{BHist}$ with $C^*_D(h)$,*

$$N^*_D(h) \Longleftrightarrow (R^*_D(h, e^*_D) \to \mathsf{False}).$$

For histories $h, k : \mathsf{BHist}$ with $C^*_D(h)$ and $C^*_D(k)$,

$$(R^*_D(\mu^*_D(h,k), e^*_D) \to \mathsf{False}) \Longleftrightarrow N^*_D(h) \vee N^*_D(k).$$

Proof. Unfold the carrier C^*_D at h. In the carried branch, $N^*_D(h)$ is exactly the displayed $\mathsf{RatHistoryCarrier}(h)$ witness. A classifier comparison $R^*_D(h, e^*_D)$ would be a mixed carried-empty endpoint comparison, because $e^*_D = \mathsf{e}$; by Definition 45.372 and Definition 37.18 this branch is impossible.

In the empty branch, $h\mathsf{hsamee}$. Since $e^*_D = \mathsf{e}$, the empty-empty classifier branch gives $R^*_D(h, e^*_D)$. The same endpoint comparison also rules out $N^*_D(h)$ by Definition 37.18. Thus the strict support predicate is exactly the failure of classification as the adjoined unit.

For the product clause, first use the carrier closure in Definition 45.424 to view $\mu^*_D(h,k)$ as an envelope-carried history. Applying the first clause to this product rewrites failure of $R^*_D(\mu^*_D(h,k), e^*_D)$ as $N^*_D(\mu^*_D(h,k))$.

Finally apply the second support law of Definition 45.426, which identifies $N_D^*(\mu_D^*(h,k))$ with $N_D^*(h) \vee N_D^*(k)$. The two equivalences compose to give the displayed product no-confusion statement. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope support unit no confusion`

Per-object template: downstream target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator support boundary; BEDC source RatHistoryCarrier, RatHistoryClassifier, Cont, append, and e; carrier C_D^*; classifier R_D^*; stability obligation strict support transport and product support exactness; ledger external-unit bit plus continuation witness; core theorem Definition 45.426; exactness predicate genuine Rat denominator support descends through R_D^* and μ_D^*; standard bridge $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator support boundary; status ProofTarget.

45.19 Rat denominator external-unit envelope cancellation exactness

Theorem 45.429 (Rat denominator external-unit envelope cancellation exactness)**.** *For the tuple $(C_D^*, R_D^*, \mu_D^*, e_D^*)$ of Definition 45.370 and the strict support predicate N_D^* of Definition 45.425, strict common factors cancel exactly at the envelope classifier. For histories h, k, t : BHist, if $C_D^*(h)$, $C_D^*(k)$, $C_D^*(t)$, and $N_D^*(t)$, then*

$$R_D^*(\mu_D^*(h,t), \mu_D^*(k,t)) \Longleftrightarrow R_D^*(h,k),$$

and

$$R_D^*(\mu_D^*(t,h), \mu_D^*(t,k)) \Longleftrightarrow R_D^*(h,k).$$

Proof. Unfold the envelope tuple and the strict support predicate. The hypothesis $N_D^*(t)$ identifies the common factor as a genuine RatHistoryCarrier endpoint, so multiplication by t remains in the carried Rat denominator branch after applying the append continuation closure used by the envelope laws.

For the forward right-cancellation direction, split the carrier witnesses for h and k into carried denominator and external empty branches.

In the carried/carried branch, all three factors lie in RatDenomUnitCarrier, and the displayed envelope classifier is read as the corresponding empty-unit classifier by Definition 45.379. Apply Definition 45.431 to cancel the strict common factor. The resulting carried denominator classifier is then packed back into $R_D^*(h,k)$ through the carried branch of the envelope classifier.

In a mixed branch, one of h and k is external while the other is carried. Because t has strict support, the product on the carried side has a RatHistoryCarrier witness, while the product on the external side is classified with the same carried continuation only through a mixed external-carried alternative. The classifier normal form and Definition 45.372, together with RatHistoryCarrier_not_empty, rule out such a product classifier. Hence the antecedent of the forward implication cannot hold in the mixed cases.

In the external/external branch, both h and k are classified with $e_D^* = \mathsf{e}$. The two products are classified with the common carried continuation of t by the one-sided unit laws, and the conclusion $R_D^*(h,k)$ is the empty branch of the envelope classifier.

The forward left-cancellation direction is the same case split with the common strict factor placed on the left. The carried/carried case uses the left cancellation field of Definition 45.431; the mixed cases are excluded by the same classifier exactness and RatHistoryCarrier_not_empty argument; the external/external case packs the empty branch directly.

For the reverse directions, assume $R^*_D(h,k)$. The carrier hypothesis on t gives reflexivity $R^*_D(t,t)$, and the multiplication congruence field of Definition 45.424 combines $R^*_D(h,k)$ with this reflexive common factor. Placing the reflexive factor on the right gives $R^*_D(\mu^*_D(h,t),\mu^*_D(k,t))$; placing it on the left gives $R^*_D(\mu^*_D(t,h),\mu^*_D(t,k))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator unit envelope cancellation exactness`

Traditional target: cancellative denominator envelope for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append; **carrier**: C^*_D; **classifier**: R^*_D; **multiplication**: μ^*_D; **unit**: e^*_D; **ledger**: external-unit bit plus common denominator continuation witness; **core theorem**: Definition 45.429; **exactness predicate**: strict-factor cancellation; **standard bridge**: ordinary denominator cancellation; **status**: LeanChecked.

Theorem 45.430 (Rat denominator external-unit envelope exact FieldUp boundary)**.** *For the concrete tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D)$ of Definition 45.370 and the strict support predicate N^*_D of Definition 45.425, the external-unit envelope supplies the exact* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *denominator boundary. Namely, the tuple carries the commutative monoid laws, restricts to the carried Rat denominator continuation fragment, separates mixed carried and external endpoints, has e^*_D as the unique external unit, transports strict support along R^*_D, computes product strict support exactly by $N^*_D(\mu^*_D(h,k)) \iff N^*_D(h) \vee N^*_D(k)$ for carried envelope inputs, and satisfies exact left and right cancellation by every strict common factor t:*

$$N^*_D(t) \to \big(R^*_D(\mu^*_D(h,t),\mu^*_D(k,t)) \iff R^*_D(h,k)\big),$$

and

$$N^*_D(t) \to \big(R^*_D(\mu^*_D(t,h),\mu^*_D(t,k)) \iff R^*_D(h,k)\big),$$

for all $h,k,t : \mathsf{BHist}$ carried by C^*_D.

Proof. The algebraic denominator envelope is supplied by Definition 45.424. Its underlying certificate data are exactly the carrier restriction to the concrete Rat denominator fragment, the carried classifier and multiplication restriction, and the mixed-endpoint separation recorded by Definition 45.383.

The external endpoint is pinned down by Definition 45.378: any carried envelope endpoint satisfying either one-sided unit law is classified with e^*_D. This supplies the unique external unit field of the boundary.

The strict-support fields come from Definition 45.426. Its first clause transports N^*_D across R^*_D-classified endpoints, and its second clause gives the exact product support equivalence for μ^*_D.

Finally, Definition 45.429 supplies the two displayed cancellation equivalences whenever the common factor has strict support. Combining these four packages gives the stated concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator boundary without any further structure. □

Traditional target: exact denominator boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, RatHistoryClassifier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; **carrier**: C^*_D; **classifier**: R^*_D; **multiplication**: μ^*_D; **unit**: e^*_D; **ledger**: external-unit bit plus denominator continuation witness; **core theorem**: Definition 45.430; **exactness predicate**: carried-fragment restriction, mixed-endpoint separation, unique external unit, strict-support transport, product support exactness, and strict-factor cancellation; **standard bridge**: concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator boundary; **status**: ProofTarget.

Theorem 45.431 (Rat denominator empty-unit cancellative commutative monoid package)**.** *For the tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ *of Definition 45.346, the laws of Definition 45.392 hold. Moreover, for carried histories* $h, k, q : \mathsf{BHist}$*, the endpoint classifier is cancellative on both sides:*

$$C^u_D(h) \wedge C^u_D(k) \wedge C^u_D(q) \wedge R^u_D(\mu^u_D(h,q), \mu^u_D(k,q)) \to R^u_D(h,k),$$

and

$$C^u_D(h) \wedge C^u_D(k) \wedge C^u_D(q) \wedge R^u_D(\mu^u_D(q,h), \mu^u_D(q,k)) \to R^u_D(h,k).$$

Proof. Unfold the concrete data of Definition 45.346. The algebraic fields are exactly the commutative monoid laws supplied by Definition 45.392.

For right cancellation, split the three $\mathsf{RatDenomUnitCarrier}$ witnesses for h, k, and q into empty endpoint and genuine Rat denominator endpoint alternatives.

In the genuine/genuine/genuine branch, the two classified products are displayed continuation endpoints with the same genuine right context q. Apply Definition 45.133 with empty unary prefix and common suffix q, then use Definition 45.134 to read the obtained comparison as the denominator classifier $R^u_D(h,k)$.

If the common right factor q is empty, the one-sided unit laws from Definition 45.347 identify $\mu^u_D(h,q)$ with h and $\mu^u_D(k,q)$ with k. Transporting the displayed classifier across those endpoint identifications gives $R^u_D(h,k)$.

If one of h and k is empty and the other is genuine, the displayed product classifier forces a mixed endpoint comparison after the same unit reduction on the empty side. The empty-unit endpoint exactness clauses from section 45.15 separate the empty endpoint from the genuine denominator endpoint, so the only classifiable mixed case is the one already carrying the comparison needed for $R^u_D(h,k)$.

The empty/empty branch is immediate: both endpoints are hsamee, and the definition of $\mathsf{RatDenomUnitClassifier}$ packs the two carried witnesses with the resulting hsame comparison.

Left cancellation follows from right cancellation and commutativity. Apply Definition 45.392 to commute both displayed left products into right products, use the right-cancellation clause just proved, and transport the classifier back along the same commutativity comparisons. □

Traditional target: cancellative commutative denominator monoid for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: $\mathsf{RatHistoryCarrier}$, $\mathsf{RatDenomUnitCarrier}$, Cont, and append; **carrier**: C^u_D; **classifier**: R^u_D; **stability obligations**: monoid laws, commutativity, and cancellation; **ledger**: external empty-unit provenance; **core theorem**: Definition 45.431; **exactness predicate**: cancellative endpoint classification; **standard bridge**: ordinary positive-denominator multiplication with adjoined unit; **status**: $\mathsf{ProofTarget}$.

Theorem 45.432 (Rat denominator empty-unit right cancellation)**.** *For the tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ *of Definition 45.346, right multiplication by a carried empty-unit denominator endpoint is exact for the classifier. For all histories* $h, k, t : \mathsf{BHist}$*, if* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(t)$*, and*

$$R^u_D(\mu^u_D(h,t), \mu^u_D(k,t)),$$

then

$$R^u_D(h,k).$$

Proof. Unfold the data of Definition 45.346. Thus μ^u_D is append, C^u_D is $\mathsf{RatDenomUnitCarrier}$, and R^u_D is $\mathsf{RatDenomUnitClassifier}$.

First consider the branch where t is the empty endpoint. The right-unit clause in Definition 45.347 classifies $\mathsf{append}(h,t)$ with h and $\mathsf{append}(k,t)$ with k. Transport the displayed classifier across these two right-unit classifications, using symmetry and transitivity of hsame inside $\mathsf{RatDenomUnitClassifier}$, to obtain $\mathsf{RatDenomUnitClassifier}(h,k)$.

Now consider the branch where t is a genuine Rat denominator endpoint. If one of h and k is empty and the other is genuine, the displayed product classifier is a mixed carried-empty endpoint after cancelling the visible append presentation; this contradicts Definition 45.382. The empty/empty subcase packs the classifier directly from the two empty endpoint witnesses.

It remains to handle the genuine/genuine subcase. The products $\mu^u_D(h,t)$ and $\mu^u_D(k,t)$ are the canonical append continuation endpoints with common suffix t. Apply Definition 45.133 with the empty prefix, the common suffix t, and the two canonical continuation witnesses for $\mathsf{Cont}(h,t,\mathsf{append}(h,t))$ and $\mathsf{Cont}(k,t,\mathsf{append}(k,t))$. The result is $\mathsf{RatHistoryClassifier}(h,k)$, and the original carrier hypotheses repack it as $\mathsf{RatDenomUnitClassifier}(h,k)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit right cancel`

Per-object template: traditional target cancellation in the denominator monoid boundary; BEDC source $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, Cont, and append; carrier C^u_D; classifier R^u_D; stability obligation right cancellation; ledger the common suffix continuation witness; core theorem Definition 45.432; exactness predicate cancellation of a shared denominator factor; standard bridge ordinary cancellative commutative monoids; status $\mathsf{ProofTarget}$.

Theorem 45.433 (Rat denominator empty-unit FieldUp multiplicative subcertificate)**.** *For the tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D)$ of Definition 45.346 and the strict support predicate N^u_D of Definition 45.394, the FieldUp denominator side over* BHist *carries the following concrete multiplicative subcertificate.*

1. *The empty-unit commutative monoid laws hold for $(C^u_D, R^u_D, \mu^u_D, e^u_D)$, including commutativity up to R^u_D.*
2. *For all carried histories $h, k : \mathsf{BHist}$, strict support of the product is exact:*
$$(\mu^u_D(h,k)\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow N^u_D(h) \vee N^u_D(k).$$
3. *For every carried history $u : \mathsf{BHist}$, if u satisfies both one-sided unit laws against all carried histories, then*
$$R^u_D(u, e^u_D).$$
4. *For all carried histories $h, k, t : \mathsf{BHist}$, classifier equality is cancellative for a common carried factor on the right and on the left:*
$$R^u_D(\mu^u_D(h,t), \mu^u_D(k,t)) \to R^u_D(h,k),$$
and
$$R^u_D(\mu^u_D(t,h), \mu^u_D(t,k)) \to R^u_D(h,k).$$

Proof. The first field is exactly the concrete commutative monoid boundary of Definition 45.392. It anchors the carrier, classifier, multiplication, and empty unit in the BHist tuple rather than in an abstract monoid variable.

The strict-support field is supplied by Definition 45.399. Its predicate is the N^u_D introduced in Definition 45.394, so the empty endpoint remains carried while the genuine Rat denominator endpoint is the only strict support source.

The unit field is Definition 45.393. It takes an arbitrary carried two-sided unit witness u and classifies it against the distinguished endpoint e_D^u.

The cancellation field is the two-sided classifier cancellation package of Definition 45.431. Together these clauses are not a variant of any single source theorem: the block composes the carrier and classifier laws, the strict support predicate, unit uniqueness, and both cancellation directions into the concrete denominator multiplicative boundary required by the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit criterion. □

Traditional target: FieldUp multiplicative denominator subcertificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, and append; **carrier**: C_D^u; **classifier**: R_D^u; **stability obligations**: commutative monoid laws, strict support exactness, unit uniqueness, and two-sided cancellation; **ledger**: empty-unit endpoint provenance plus shared denominator continuation witness; **core theorem**: Definition 45.433; **exactness predicate**: concrete carried denominator products classify exactly through the empty-unit boundary; **standard bridge**: ordinary positive-denominator multiplication with an adjoined empty unit; **status**: ProofTarget.

Theorem 45.434 (Rat denominator external-unit FieldUp concrete instance unlock)**.** *The concrete* BHist *tuple* $(C_D^*, R_D^*, \mu_D^*, e_D^*, N_D^*)$*, with* $\mu_D^*(h,k) := \mathsf{append}(h,k)$*,* $e_D^* := \mathsf{e}$*, and strict support*

$$N_D^*(h) \iff \mathsf{RatHistoryCarrier}(h),$$

supplies the external-unit denominator instance required by the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *multiplicative schema. Concretely, the tuple carries:*

1. *commutative monoid laws for* $(C_D^*, R_D^*, \mu_D^*, e_D^*)$*;*
2. *strict-support transport along* R_D^* *and exact product support for* μ_D^**;*
3. *support-unit no-confusion at the external endpoint* e_D^**;*
4. *the exact* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *external-unit denominator boundary;*
5. *the cancellative commutative monoid package;*
6. *right cancellation for a common carried denominator factor;*
7. *the multiplicative subcertificate that repacks the denominator-side fields needed by the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit criterion.*

Proof. The underlying monoid part is the concrete external-unit envelope of Definition 45.424. It fixes the displayed carrier, classifier, append multiplication, and empty unit, and it includes commutativity at the endpoint classifier.

The strict-support fields are those of Definition 45.426. They transport N_D^* across R_D^* and compute product support through μ_D^* exactly. The support-unit obstruction is supplied next by Definition 45.428.

The FieldUp denominator boundary is then the exact external-unit boundary of Definition 45.430. This packages the support transport, product-support exactness, and exact cancellation boundary for strict carried factors.

The cancellative algebraic layer is supplied by Definition 45.431, and the concrete right-cancellation row is Definition 45.432. These rows provide the ordinary cancellative commutative monoid package and the common right-factor cancellation required by the denominator side.

Finally, Definition 45.433 repackages the monoid laws, strict support, unit field, and cancellation fields as the multiplicative subcertificate. Hence the displayed BHist tuple supplies the concrete external-unit denominator instance for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. □

Theorem 45.435 (Rat denominator empty-unit strict-support FieldUp subcertificate)**.** *For the tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *of Definition 45.346 and the strict support predicate* N^u_D *of Definition 45.394, the FieldUp denominator side over* BHist *carries the following concrete strict-support subcertificate.*

1. *The empty-unit commutative monoid laws hold for* $(C^u_D, R^u_D, \mu^u_D, e^u_D)$*, including commutativity up to* R^u_D*.*
2. *For all carried histories* $h, k : \mathsf{BHist}$*, classifier transport preserves and reflects strict support:*
$$R^u_D(h, k) \to (N^u_D(h) \iff N^u_D(k)).$$
3. *For all carried histories* $h, k : \mathsf{BHist}$*, strict support is exact for the product:*
$$N^u_D(\mu^u_D(h, k)) \iff N^u_D(h) \vee N^u_D(k).$$
4. *For every carried history* $u : \mathsf{BHist}$*, if* u *satisfies both one-sided unit laws against all carried histories, then*
$$R^u_D(u, e^u_D).$$
5. *For all carried histories* $h, k, t : \mathsf{BHist}$*, classifier equality is cancellative for a common carried factor on the right and on the left:*
$$R^u_D(\mu^u_D(h, t), \mu^u_D(k, t)) \to R^u_D(h, k),$$
and
$$R^u_D(\mu^u_D(t, h), \mu^u_D(t, k)) \to R^u_D(h, k).$$
6. *For histories* $p, q, p', q', h, k : \mathsf{BHist}$*, empty endpoint contexts preserve and reflect the classifier: if* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then*
$$R^u_D(\mathsf{append}(p, \mathsf{append}(h, q)), \mathsf{append}(p', \mathsf{append}(k, q'))) \iff R^u_D(h, k).$$

Proof. The first field is exactly the concrete commutative monoid boundary of Definition 45.392. It anchors the carrier, classifier, multiplication, and empty unit in the BHist tuple rather than in an abstract monoid variable.

For support transport, unfold R^u_D as RatDenomUnitClassifier and N^u_D as RatHistoryCarrier. A classifier witness supplies the endpoint hsame comparison between h and k; applying RatHistoryCarrier_hsame_trans in both directions gives $N^u_D(h) \iff N^u_D(k)$.

The product support field is supplied by Definition 45.402. Its predicate is the N^u_D introduced in Definition 45.394, so the empty endpoint remains carried while the genuine Rat denominator endpoint is the only strict support source.

The unit field is Definition 45.393. It takes an arbitrary carried two-sided unit witness u and classifies it against the distinguished endpoint e^u_D.

The cancellation field is the two-sided classifier cancellation package of Definition 45.431.

The empty-context field is exactly the preservation-and-reflection theorem Definition 45.405. It uses the same classifier $R^u_D = \mathsf{RatDenomUnitClassifier}$ and the same append operation as the tuple.

Together these clauses compose the carrier and classifier laws, strict-support transport, product support exactness, unit uniqueness, cancellation, and empty-context classifier exactness into the concrete denominator boundary required by the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit criterion. □

Traditional target: FieldUp strict-support denominator subcertificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, and append; **carrier**: C^u_D; **classifier**: R^u_D; **stability obligations**: commutative monoid laws, strict support transport, product support exactness, unit uniqueness, two-sided cancellation, and empty-context classifier exactness; **ledger**: empty-unit endpoint provenance plus shared denominator continuation witness; **core theorem**: Definition 45.435; **exactness predicate**: genuine denominator support is invariant under the classifier and exact under empty-unit products; **standard bridge**: ordinary positive-denominator multiplication with an adjoined empty unit; **status**: ProofTarget.

Theorem 45.436 (Rat denominator empty-unit contextual multiplication classifier exactness)**.** *For histories p, q, p', q', h, k, t : BHist, if $p$hsamee, $q$hsamee, p'hsamee, q'hsamee, $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(t)$, then fixed denominator multiplication is classifier-exact inside empty external contexts on both sides:*

$$R^u_D(\mathsf{append}(p, \mathsf{append}(\mu^u_D(h,t), q)), \mathsf{append}(p', \mathsf{append}(\mu^u_D(k,t), q'))) \iff R^u_D(h,k),$$

and

$$R^u_D(\mathsf{append}(p, \mathsf{append}(\mu^u_D(t,h), q)), \mathsf{append}(p', \mathsf{append}(\mu^u_D(t,k), q'))) \iff R^u_D(h,k).$$

Proof. For the right multiplication law, apply Definition 45.405 to the four hypotheses $p$hsamee, $q$hsamee, p'hsamee, and q'hsamee. This strips the empty external contexts from the displayed classifier and leaves the comparison $R^u_D(\mu^u_D(h,t), \mu^u_D(k,t))$.

The carried hypotheses $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(t)$ now match the input of Definition 45.410. That theorem identifies the stripped comparison with $R^u_D(h,k)$, giving the first equivalence.

For the left multiplication law, the same empty-context exactness theorem again removes the external histories and leaves $R^u_D(\mu^u_D(t,h), \mu^u_D(t,k))$.

The same carrier hypotheses then match Definition 45.413, which identifies this stripped comparison with $R^u_D(h,k)$. This gives the second equivalence. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator empty unit contextual multiplication classifier exactness`

1. Traditional target: contextual multiplication classifier exactness for the denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.
2. BEDC source: BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append.
3. Carrier: C^u_D.
4. Classifier: R^u_D.
5. Stability obligations: empty external contexts preserve and reflect the classifier, and fixed denominator multiplication preserves and reflects the classifier on both sides.
6. Ledger: four empty external context witnesses and one carried fixed denominator factor.

7. Core theorem: Definition 45.436.
8. Exactness predicate: classifier equivalence before and after fixed denominator multiplication inside empty external contexts.
9. Standard bridge: ordinary denominator multiplication with an adjoined empty unit and empty endpoint contexts.
10. Status: ProofTarget.

Theorem 45.437 (Rat denominator FieldUp multiplicative certificate boundary)**.** *Let the denominator carrier, classifier, multiplication, unit, and strict support be the concrete* BHist *data*

$$C_D^*(h), \qquad R_D^*(h,k), \qquad \mu_D^*(h,k) := \mathsf{append}(h,k), \qquad e_D^* := \mathsf{e}, \qquad N_D^*(h) := \mathsf{RatHistoryCarrier}(h).$$

This tuple supplies the denominator-side multiplicative certificate boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*: envelope classifier exactness, monoid laws, commutativity, strict-support transport, product-support exactness, empty-unit identity uniqueness, and carried-endpoint cancellation all hold for the displayed* BHist *denominator package.*

Proof. The carrier, classifier, operation, and unit are the external-unit envelope data of Definition 45.370. Its classifier boundary is exactly Definition 45.372.

The algebraic envelope comes from two checked components. The monoid laws are those of Definition 45.377, and the commutative law is supplied by Definition 45.424.

The strict support named in the statement is the support predicate of Definition 45.426. That theorem gives both classifier transport for N_D^* and the exact product law

$$N_D^*(\mu_D^*(h,k)) \Longleftrightarrow N_D^*(h) \vee N_D^*(k).$$

The empty endpoint is controlled by Definition 45.393, which identifies any carried two-sided identity with $e_D^* = \mathsf{e}$ through the endpoint classifier. The carried cancellation part is the cancellation field of Definition 45.431, read on the same denominator append operation and the same empty endpoint.

Combining these clauses gives a single concrete denominator-side multiplicative boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, with no abstract carrier or classifier left to choose. □

Theorem 45.438 (Rat denominator FieldUp contextual multiplicative boundary)**.** *Let the denominator-side empty-unit tuple for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *be the concrete* BHist *data*

$$(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u) = (\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier}).$$

Equivalently, $\mu_D^u(h,k) = \mathsf{append}(h,k)$*,* $e_D^u = \mathsf{e}$*, and* $N_D^u(h) = \mathsf{RatHistoryCarrier}(h)$*. The FieldUp denominator-side contextual multiplicative boundary holds for this tuple:*

1. *empty external contexts preserve and reflect* R_D^u*;*
2. *right multiplication by any carried denominator factor is classifier-exact;*
3. *left multiplication by any carried denominator factor is classifier-exact;*
4. *the combined two-sided contextual statement is exactly Definition 45.436.*

Proof. The displayed tuple is the concrete cancellation-exact denominator instance of Definition 45.409. Thus the carrier, classifier, append multiplication, empty endpoint, and strict support are already fixed by the BHist denominator package.

For empty external contexts, apply Definition 45.405. It strips and restores empty histories on both sides of the denominator classifier, so context insertion by e preserves and reflects R^u_D.

For fixed-factor multiplication, apply Definition 45.410 and Definition 45.413. These give classifier exactness after right and left multiplication by any carried denominator factor.

Finally, combine the empty-context exactness with the two fixed-factor exactness laws through Definition 45.436. That theorem is precisely the packaged two-sided contextual multiplication boundary for the displayed FieldUp denominator tuple. □

Traditional target: contextual multiplicative denominator boundary for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, and append; **carrier**: C^u_D; **classifier**: R^u_D; **stability obligations**: empty-context classifier exactness plus right- and left-factor multiplication exactness; **ledger**: empty external context witnesses and one carried denominator factor; **core theorem**: Definition 45.436; **exactness predicate**: classifier preservation and reflection after fixed denominator multiplication inside empty external contexts; **standard bridge**: ordinary denominator multiplication with an adjoined empty unit and empty endpoint contexts; **status**: ProofTarget.

Theorem 45.439 (Rat denominator FieldUp multiplicative certificate obligation coverage)**.** *For the concrete tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$ of Definition 45.437, the multiplicative* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate rows are covered by the displayed* BHist *denominator data:*

1. *carrier closure and unit inhabitation for C^*_D and e^*_D;*
2. *associativity, commutativity, and binary classifier congruence for μ^*_D under R^*_D;*
3. *strict-support classifier transport and product-support exactness for N^*_D;*
4. *empty-unit uniqueness for the external endpoint e^*_D;*
5. *two-sided cancellation over the empty-unit cancellative commutative monoid package;*
6. *empty-product reflection through Definition 45.385.*

Thus the boundary theorem is a concrete BHist *tuple law for later Lean formalization rather than abstract carrier/classifier transport.*

Proof. Start with Definition 45.437. It fixes the five objects C^*_D, R^*_D, μ^*_D, e^*_D, and N^*_D and identifies them with the external-unit denominator envelope used at the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ boundary.

The carrier rows, the unit row, associativity, binary classifier congruence, restriction to the carried denominator fragment, mixed-endpoint separation, and empty-unit uniqueness are supplied by Definition 45.383. The commutative row is the commutative part already packaged into the boundary theorem from the same envelope data.

The strict-support rows are exactly Definition 45.426: its first clause transports N^*_D across R^*_D-classified endpoints, and its second clause computes strict support of $\mu^*_D(h, k)$ from the two input supports.

The cancellation row is the denominator endpoint cancellation field of Definition 45.431, read through the same empty-unit carrier, classifier, append operation, and endpoint. The empty-product row is precisely Definition 45.385.

These cited rows exhaust the multiplicative denominator obligations listed for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate, so the certificate is NameCert-ready as a single concrete tuple over BHist. □

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ multiplicative denominator certificate; BEDC source RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, and append; carrier C^*_D; classifier R^*_D; stability obligations carrier closure, unit inhabitation, associativity, commutativity, binary classifier congruence, strict-support classifier transport, product-support exactness, empty-unit uniqueness, two-sided cancellation, and empty-product reflection; ledger external empty-unit endpoint plus denominator continuation provenance; core theorem Definition 45.439; exactness predicate strict support and empty-product reflection; standard bridge ordinary positive-denominator multiplication with adjoined unit; status ProofTarget.

Definition 45.440 (Concrete Rat denominator FieldUp multiplicative exactness instance)**.** *The concrete* BHist *denominator-side multiplicative exactness instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the tuple*

$$\mathcal{M}^*_D := (C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$$

where C^*_D and R^*_D are the carrier and classifier of Definition 45.370,

$$\mu^*_D(h,k) := \mathsf{append}(h,k), \qquad e^*_D := \mathsf{e}, \qquad N^*_D(h) := \mathsf{RatHistoryCarrier}(h).$$

Theorem 45.441 (Rat denominator FieldUp envelope empty-context exactness)**.** *For the concrete denominator-side tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$ of Definition 45.440, let $p, q, p', q', h, k : \mathsf{BHist}$. If $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, $q'\mathsf{hsamee}$, $C^*_D(h)$, and $C^*_D(k)$, then the external-unit envelope classifier and strict support are exact under empty external endpoint contexts:*

$$R^*_D(\mathsf{append}(p, \mathsf{append}(h,q)), \mathsf{append}(p', \mathsf{append}(k,q'))) \iff R^*_D(h,k),$$

$$N^*_D(\mathsf{append}(p, \mathsf{append}(h,q))) \iff N^*_D(h),$$

and

$$N^*_D(\mathsf{append}(p', \mathsf{append}(k,q'))) \iff N^*_D(k).$$

Proof. The append left and right unit laws first identify $\mathsf{append}(p, \mathsf{append}(h,q))$ with h and $\mathsf{append}(p', \mathsf{append}(k,q'))$ with k, using the four hypotheses that the external contexts are history-same to e.

By Definition 45.371, those endpoint comparisons, together with the carrier hypotheses $C^*_D(h)$ and $C^*_D(k)$, classify each contextualized endpoint with its original endpoint inside R^*_D. The binary classifier congruence field in Definition 45.424 then transports the classifier comparison in both directions, giving the displayed classifier exactness.

For strict support, apply the first clause of Definition 45.426 to the two endpoint classifications just obtained. Since N^*_D is RatHistoryCarrier in the displayed tuple, this transports strict denominator support from each contextualized endpoint back to the original endpoint and conversely. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty context iff`

Theorem 45.442 (Rat denominator envelope empty-context history-sameness exactness)**.** *For histories $h, k : \mathsf{BHist}$, if $C^*_D(h)$ and $C^*_D(k)$, then the mixed empty endpoint contexts classify exactly the underlying history-sameness:*

$$R^*_D(\mathsf{append}(\mathsf{e}, h), \mathsf{append}(k, \mathsf{e})) \iff h\mathsf{hsame}k.$$

Proof. Split the two carrier hypotheses into the genuine Rat denominator branch and the empty endpoint branch. In the carried branch, the displayed classifier is transported across the left and right append-unit laws, then read back as the Rat denominator classifier comparison. In the empty branch, append endpoint readback identifies the relevant endpoint with e; mixed carried-empty

branches contradict the strict nonempty Rat denominator carrier, while the double-empty branch composes the two empty endpoint equalities into $h\mathsf{hsame}k$. The reverse implication uses the same branch split: carried endpoints repack the Rat denominator classifier after append-unit transport, and empty endpoint branches either produce the empty classifier row or are ruled out by nonempty Rat denominator support. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatDenominatorUnitEnvelopeClassifier empty context hsame iff`

Theorem 45.443 (Rat denominator FieldUp multiplicative exactness boundary)**.** *The concrete instance Definition 45.440 satisfies the Rat denominator FieldUp multiplicative certificate boundary. In addition, for all $h, k, t : \mathsf{BHist}$, if $C^*_D(h)$, $C^*_D(k)$, and $N^*_D(t)$, then*

$$R^*_D(\mu^*_D(h,t), \mu^*_D(k,t)) \Longleftrightarrow R^*_D(h,k),$$

and

$$R^*_D(\mu^*_D(t,h), \mu^*_D(t,k)) \Longleftrightarrow R^*_D(h,k).$$

Proof. The displayed tuple is the same concrete carrier, classifier, multiplication, unit, and strict support used by Definition 45.437. That theorem supplies the packaged carrier, classifier, monoid, commutativity, empty-unit uniqueness, and carried-endpoint clauses for the denominator-side FieldUp boundary.

The strict-support component is the predicate $N^*_D(h) := \mathsf{RatHistoryCarrier}(h)$. The transport and product-support parts of the boundary are therefore the corresponding clauses of Definition 45.426.

It remains to identify strict common-factor cancellation at the same concrete classifier. Since $N^*_D(t)$ is by definition $\mathsf{RatHistoryCarrier}(t)$, Definition 45.370 places t in C^*_D through the carried alternative.

Under $C^*_D(h)$, $C^*_D(k)$, and this derived carrier witness for t, the right and left cancellation equivalences are exactly the two displayed clauses of Definition 45.429. Hence the concrete tuple carries both the certificate boundary and the strict factor-cancellation exactness required for the denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. □

Definition 45.444 (Concrete Rat denominator FieldUp contextual certificate instance)**.** *The concrete contextual denominator certificate instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the* BHist *tuple*

$$\mathcal{C}^u_D := (C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D) = (\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier}).$$

Equivalently, $C^u_D = \mathsf{RatDenomUnitCarrier}$, $R^u_D = \mathsf{RatDenomUnitClassifier}$, $\mu^u_D(h,k) = \mathsf{append}(h,k)$, $e^u_D = \mathsf{e}$, and $N^u_D(h) = \mathsf{RatHistoryCarrier}(h)$. The displayed certificate interface also contains empty external context witnesses $p, q, p', q' : \mathsf{BHist}$ with $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, so the contextual denominator rows are read as fields of this concrete tuple rather than as detached endpoint facts.

Theorem 45.445 (Rat denominator FieldUp contextual certificate obligation coverage)**.** *For the concrete tuple $\mathcal{C}^u_D$ of Definition 45.444, the contextual denominator rows for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *are covered by the displayed* BHist *certificate data:*

1. *empty external contexts preserve and reflect R^u_D;*
2. *right multiplication by a fixed carried denominator factor is classifier-exact;*

3. left multiplication by a fixed carried denominator factor is classifier-exact;

4. two-sided contextual multiplication is classifier-exact inside empty external contexts;

5. strict support of the contextual product endpoint is exact.

Thus $\mathcal{C}_D^u$ is a concrete contextual denominator certificate target for the FieldUp parent chapter.

Proof. The tuple in Definition 45.444 is the same concrete carrier, classifier, append operation, empty endpoint, and strict-support predicate used by Definition 45.438. That theorem supplies the empty-context classifier preservation and reflection rows together with the left and right fixed-factor multiplication exactness rows.

The two-sided contextual multiplication row is precisely Definition 45.436. It reads the four empty external context witnesses as part of the same BHist denominator interface and identifies the contextual fixed multiplication comparison with the uncontextualized classifier comparison.

The support row is Definition 45.406. It computes strict denominator support after empty external histories surround the product endpoint, using the disjunction of the two input strict-support facts.

Finally, Definition 45.439 packages the ordinary multiplicative denominator obligations for the same append-based endpoint system. Combining that package with the contextual boundary and strict contextual product-support theorem gives a single contextual certificate target for the FieldUp denominator rows. □

Per-object template: traditional target contextual denominator certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, and append; carrier C_D^u; classifier R_D^u; stability obligations empty-context classifier exactness, left and right fixed-factor multiplication exactness, two-sided contextual multiplication classifier exactness, and contextual strict product support exactness; ledger four empty external context witnesses and one carried denominator factor; core theorem Definition 45.445; exactness predicate classifier equivalence and strict support through empty external denominator contexts; standard bridge ordinary denominator multiplication with adjoined empty unit and contextual endpoint witnesses; status ProofTarget.

Theorem 45.446 (Rat denominator external-unit envelope contextual multiplication exactness)**.** *For the concrete tuple $(C_D^*, R_D^*, \mu_D^*, e_D^*, N_D^*)$ of Definition 45.370, let $p, q, p', q', h, k, t : \mathsf{BHist}$. If $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, $q'\mathsf{hsamee}$, $C_D^*(h)$, $C_D^*(k)$, and $N_D^*(t)$, then fixed denominator multiplication is classifier-exact inside empty external contexts on both sides:*

$$R_D^*(\mathsf{append}(p, \mathsf{append}(\mu_D^*(h,t), q)), \mathsf{append}(p', \mathsf{append}(\mu_D^*(k,t), q'))) \Longleftrightarrow R_D^*(h,k),$$

and

$$R_D^*(\mathsf{append}(p, \mathsf{append}(\mu_D^*(t,h), q)), \mathsf{append}(p', \mathsf{append}(\mu_D^*(t,k), q'))) \Longleftrightarrow R_D^*(h,k).$$

Proof. Use Definition 45.371 to identify the envelope carrier C_D^* with RatDenomUnitCarrier and the envelope classifier R_D^* with RatDenomUnitClassifier. Thus the hypotheses $C_D^*(h)$ and $C_D^*(k)$ are the empty-unit carrier hypotheses required for h and k.

The strict support hypothesis $N_D^*(t)$ is $\mathsf{RatHistoryCarrier}(t)$ on the carried endpoint branch. This gives $\mathsf{RatDenomUnitCarrier}(t)$ by the carried alternative of the empty-unit carrier, so t is also an admissible denominator factor for the empty-unit contextual theorem.

Apply Definition 45.436 with the four empty context witnesses and the three empty-unit carrier witnesses just obtained. Its right fixed-factor equivalence gives the first displayed comparison after rewriting R^u_D as R^*_D and μ^u_D as μ^*_D through the normal form and the common append operation.

The same application supplies the left fixed-factor equivalence. Translating the classifier and multiplication symbols back along Definition 45.371 gives the second displayed comparison for the external-unit envelope tuple. □

1. Traditional target: contextual multiplication classifier exactness for the external-unit envelope denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.
2. BEDC source: BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, Cont, and append.
3. Carrier: C^*_D.
4. Classifier: R^*_D.
5. Stability obligations: empty external contexts preserve and reflect the envelope classifier, and fixed denominator multiplication preserves and reflects the envelope classifier on both sides.
6. Ledger: four empty external context witnesses, two envelope carrier witnesses, and one strict carried denominator factor.
7. Core theorem: Definition 45.446.
8. Exactness predicate: classifier equivalence before and after fixed denominator multiplication inside empty external contexts.
9. Standard bridge: empty-unit contextual multiplication exactness transported through the external-unit envelope normal form.
10. Status: ProofTarget.

Theorem 45.447 (Rat denominator FieldUp multiplicative certificate record)**.** *The tuple*

$$\mathcal{M}^*_D = (C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$$

of Definition 45.440 is the concrete BHist *denominator-side multiplicative certificate record for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*. It simultaneously supplies carrier, unit, associativity, commutativity, congruence, strict-support, empty-unit, cancellation, and empty-product coverage from Definition 45.439; strict common-factor right and left cancellation exactness from Definition 45.443; and the concrete denominator certificate boundary from Definition 45.437.*

Proof. Definition 45.437 fixes the denominator-side boundary on the five displayed objects C^*_D, R^*_D, μ^*_D, e^*_D, and N^*_D. These are the same carrier, classifier, operation, unit, and strict-support predicate named by Definition 45.440.

Definition 45.439 then enumerates the certificate rows carried by that tuple: carrier and unit coverage, associativity, commutativity, congruence, strict-support transport and product exactness, empty-unit uniqueness, cancellation coverage, and empty-product reflection.

Finally, Definition 45.443 adds the strict common-factor right and left cancellation equivalences on the same classifier and the same multiplication. Since all three cited results are aligned on the identical concrete tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$, their rows assemble as the single BHist denominator-side multiplicative certificate record for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. □

Theorem 45.448 (RatUp FieldUp denominator certificate assembly)**.** *Let the denominator-side data for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *be the concrete external-unit envelope tuple* $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$*, with* $\mu^*_D(h,k) := \mathsf{append}(h,k)$*,* $e^*_D := \mathsf{e}$*, and* $N^*_D(h) := \mathsf{RatHistoryCarrier}(h)$ *on the strict carried endpoint branch. Then this tuple is the denominator subcertificate consumed by* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*: the envelope classifier is exact, the denominator operation carries the commutative monoid laws, strict support transports across* R^*_D*, strict support is product-exact for* μ^*_D*, strict common factors give exact left and right cancellation, fixed multiplication is classifier-exact inside empty external contexts, the external unit is unique, and* e^*_D *is not a carried denominator unit.*

Proof. The exact envelope boundary is Definition 45.430. It supplies the concrete carrier, classifier, operation, unit, strict support, commutative monoid laws, support transport, support product law, unique external endpoint, and strict-factor cancellation clauses for $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$.

The remaining contextual multiplication field is Definition 45.446. It states that fixed denominator multiplication preserves and reflects the external-unit envelope classifier after empty external histories are attached on both sides, using the same BHist append operation and the strict carried denominator factor predicate.

Finally, Definition 45.437 collects the denominator-side multiplicative certificate boundary for the displayed tuple. Combining that boundary with the contextual multiplication exactness clause and the exact envelope theorem gives the advertised $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator subcertificate. The statement that no carried denominator unit remains is the strict-support reading of $e^*_D = \mathsf{e}$ together with the carried-endpoint separation already included in the exact envelope boundary. □

Definition 45.449 (Concrete RatUp FieldUp denominator certificate row)**.** *The concrete denominator certificate row for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is*

$$(C^{\dagger}_D, R^{\dagger}_D, \mu^{\dagger}_D, e^{\dagger}_D, N^{\dagger}_D),$$

where

$$C^{\dagger}_D := \mathsf{RatDenomUnitCarrier}, \qquad R^{\dagger}_D := \mathsf{RatDenomUnitClassifier},$$
$$\mu^{\dagger}_D(h,k) := \mathsf{append}(h,k), \qquad e^{\dagger}_D := \mathsf{e}, \qquad N^{\dagger}_D(h) := \mathsf{RatHistoryCarrier}(h).$$

Its ledger is the external empty-unit endpoint plus denominator continuation provenance.

Theorem 45.450 (RatUp FieldUp denominator certificate row laws)**.** *The row Definition 45.449 covers the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *denominator obligations: envelope classifier exactness, carrier closure and unit inhabitation, associativity, commutativity, binary classifier congruence, strict-support transport and product exactness, left and right strict-factor cancellation, contextual fixed-factor classifier exactness, unique external unit, and absence of a carried denominator unit at* $e^{\dagger}_D$*.*

Proof. The row uses the same concrete denominator carrier and classifier as the unit envelope. The envelope classifier exactness is Definition 45.372, while Definition 45.430 packages carrier closure, the empty endpoint, endpoint separation, strict-support transport, product exactness, and the external unit uniqueness law.

The multiplication field is the native BHist append operation. Definition 45.437 supplies the associative and commutative certificate clauses, binary classifier congruence, and the strict-factor cancellation laws for this operation on the same denominator envelope.

The contextual fixed-factor classifier exactness is Definition 45.438. The full denominator obligation list is then collected by Definition 45.439, and Definition 45.448 identifies that collected package with the denominator subcertificate consumed by $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.

The FieldUp denominator schema is instantiated by this concrete BHist row, so BHist-anchored proofs can target this row rather than abstract carrier and classifier parameters. □

Theorem 45.451 (RatUp FieldUp denominator obligation map)**.** *For histories $h, k : \mathsf{BHist}$, take the concrete denominator data*

$$C^*_D(h) := (\mathsf{RatHistoryCarrier}(h) \vee h\mathsf{hsamee}), \qquad R^*_D(h,k) := (\mathsf{RatHistoryCarrier}(h) \wedge \mathsf{RatHistoryCarrier}(k) \wedge \mathsf{RatHisto}$$

$$\mu^*_D(h,k) := \mathsf{append}(h,k), \qquad e^*_D := \mathsf{e}, \qquad N^*_D(h) := \mathsf{RatHistoryCarrier}(h).$$

These data discharge the denominator rows of Definition 45.337 for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*:*

1. *envelope classifier exactness;*
2. *carried-fragment restriction;*
3. *commutative monoid laws for μ^*_D with unit e^*_D;*
4. *strict-support transport along R^*_D;*
5. *strict-support product exactness for μ^*_D;*
6. *strict common-factor left and right cancellation;*
7. *fixed multiplication classifier exactness in empty external contexts;*
8. *external-unit uniqueness;*
9. *exclusion of e^*_D from strict carried support.*

Proof. The first row group is supplied by Definition 45.430. It is stated for the same envelope carrier, classifier, append multiplication, empty external unit, and strict carried-support predicate displayed above, so its exactness, carried-fragment, commutative monoid, support, cancellation, and external-unit clauses read directly as rows for this map.

The contextual row is supplied by Definition 45.438. Its empty-context fixed-multiplication clauses use the same append operation and the same empty endpoint, and Definition 45.439 enumerates the ordinary multiplicative rows on the identical denominator tuple.

Finally, Definition 45.448 identifies these rows as the denominator subcertificate consumed by $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. Thus the displayed row map is not a detached list of facts: it is the named denominator interface through which the concrete external-unit envelope enters the FieldUp certificate obligations. □

Traditional target: assembled denominator subcertificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; **BEDC source**: BHist, RatHistoryCarrier, RatHistoryClassifier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; **carrier**: C^*_D; **classifier**: R^*_D; **multiplication**: μ^*_D; **unit**: e^*_D; **strict support**: N^*_D; **ledger**: external-unit endpoint bit plus denominator continuation witness; **core theorem**: Definition 45.448; **exactness predicate**: exact envelope classifier, contextual fixed multiplication exactness, strict-support transport and product exactness, strict-factor cancellation, unique external unit, and absence of a carried denominator unit; **status**: ProofTarget.

Theorem 45.452 (RatUp FieldUp denominator row exhaustion)**.** *Let*

$$\mathcal{M}^*_D = (C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$$

be the concrete multiplicative tuple of Definition 45.440, and let

$$\mathcal{C}^u_D = (C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$$

be the concrete contextual tuple of Definition 45.444. Every denominator-side $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *row used in this parent chapter is supplied by either the multiplicative certificate record for* $\mathcal{M}^*_D$ *or the contextual certificate coverage for* $\mathcal{C}^u_D$*, with their shared tuple alignment governed by Definition 45.448. Thus the full denominator certificate is a concrete* BHist *target: the carrier and classifier are the predicates* $\mathsf{RatDenomUnitCarrier}$ *and* $\mathsf{RatDenomUnitClassifier}$*, multiplication is* append*, strict support is* $\mathsf{RatHistoryCarrier}$*, empty external contexts are the only context witnesses, and no carried denominator unit is introduced.*

Proof. The multiplicative source is Definition 45.447. It supplies the carrier, unit, associativity, commutativity, congruence, strict-support, cancellation, and empty-product rows for $\mathcal{M}^*_D$.

The contextual source is Definition 45.445. It supplies the empty-context rows and fixed-factor contextual rows for $\mathcal{C}^u_D$, with contextual support read through the same carried denominator endpoint predicate.

Finally, Definition 45.448 identifies the common BHist carrier, classifier, append multiplication, empty endpoint, and strict-support predicate that the two sources share. The same assembly also records that the empty endpoint is external to the carried denominator branch, so the combined denominator-side certificate has no carried denominator unit. □

Theorem 45.453 (Rat denominator bilateral strict factor cancellation)**.** *For the concrete tuple* $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$ *of Definition 45.440, let* $h, k, l, r : \mathsf{BHist}$*. If* $C^*_D(h)$*,* $C^*_D(k)$*,* $C^*_D(l)$*,* $C^*_D(r)$*,* $N^*_D(l)$*, and* $N^*_D(r)$*, then*

$$R^*_D(\mu^*_D(l, \mu^*_D(h, r)), \mu^*_D(l, \mu^*_D(k, r))) \iff R^*_D(h, k).$$

Proof. First build the products to which cancellation will be applied. The strict-support product closure and carrier closure supplied by Definition 45.437 place $\mu^*_D(l, h)$ and $\mu^*_D(l, k)$ in C^*_D, and keep the larger products with the strict factor r inside the same concrete carrier. The associativity and congruence fields of that boundary identify the displayed comparison with $R^*_D(\mu^*_D(\mu^*_D(l, h), r), \mu^*_D(\mu^*_D(l, k), r))$.

Apply Definition 45.429 with the strict common right factor r. Since $N^*_D(r)$ and the two left products are carried by the preceding closure step, this gives

$$R^*_D(\mu^*_D(\mu^*_D(l, h), r), \mu^*_D(\mu^*_D(l, k), r)) \iff R^*_D(\mu^*_D(l, h), \mu^*_D(l, k)).$$

Apply the same cancellation theorem with the strict common left factor l. The hypotheses $C^*_D(h)$, $C^*_D(k)$, $C^*_D(l)$, and $N^*_D(l)$ are exactly its input hypotheses for the left-cancellation clause, so

$$R^*_D(\mu^*_D(l, h), \mu^*_D(l, k)) \iff R^*_D(h, k).$$

Composing the two equivalences gives the bilateral strict-factor cancellation claim. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator bilateral strict factor cancellation`

Definition 45.454 (RatUp FieldUp denominator contextual action instance)**.** *The concrete denominator contextual action instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the tuple*

$$\mathcal{A}^u_D := (C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D, \Gamma)$$

where

$$C^u_D := \mathsf{RatDenomUnitCarrier}, \qquad R^u_D := \mathsf{RatDenomUnitClassifier},$$

$$\mu^u_D(h, k) := \mathsf{append}(h, k), \qquad e^u_D := \mathsf{e}, \qquad N^u_D(h) := \mathsf{RatHistoryCarrier}(h).$$

For carried contexts l, r : BHist and external endpoint contexts p, q : BHist, its bilateral carried action with external endpoints is

$$\Gamma_{p,q,l,r}(h) := \mathsf{append}(p, \mathsf{append}(B_{l,r}(h), q)),$$

where $B_{l,r}$ is the bilateral denominator multiplication context of Definition 45.417.

Lean4 (definition) `BEDC.Derived.FieldUp.RatDenomUnitContextualAction`

Theorem 45.455 (Rat denominator contextual action empty endpoint composition)**.** *For histories $h, l, r, l', r', p, q, p', q'$, if all four external endpoints p, q, p', q' are* hsame*-equal to* e*, then the iterated concrete action $\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$ is* hsame*-equal to the single action with carried contexts* $\mathsf{append}(l', l)$ *and* $\mathsf{append}(r, r')$*.*

Proof. Case analysis on the four endpoint equalities reduces the statement to the associativity and empty-endpoint laws for append. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty endpoint composition`

Theorem 45.456 (RatUp FieldUp denominator contextual action certificate)**.** *For histories h, k, l, r, p, q, p', q' :* BHist*, assume $C^u_D(h)$, $C^u_D(k)$, $C^u_D(l)$, $C^u_D(r)$, and* phsamee*,* qhsamee*,* p'hsamee*, and* q'hsamee*. Then the contextual action instance of Definition 45.454 satisfies the following classifier and strict-support laws:*

$$R^u_D(\Gamma_{p,q,l,r}(h), \Gamma_{p',q',l,r}(k)) \iff R^u_D(h, k),$$

$$N^u_D(\Gamma_{p,q,l,r}(h)) \iff N^u_D(l) \vee N^u_D(h) \vee N^u_D(r),$$

and

$$N^u_D(\Gamma_{p',q',l,r}(k)) \iff N^u_D(l) \vee N^u_D(k) \vee N^u_D(r).$$

Proof. First use closure for the carried bilateral product. From $C^u_D(l)$, $C^u_D(h)$, and $C^u_D(r)$, the denominator continuation monoid laws put $B_{l,r}(h)$ inside C^u_D; the same argument using k gives $C^u_D(B_{l,r}(k))$.

For classifier exactness, strip the empty external endpoint contexts with Definition 45.405. The endpoint equalities phsamee, qhsamee, p'hsamee, and q'hsamee reduce the displayed classifier comparison to $R^u_D(B_{l,r}(h), B_{l,r}(k))$.

Apply Definition 45.418 to the carried histories h, k, l, and r. This gives exactly $R^u_D(h, k)$, and the reverse direction follows by the same two equivalences in the opposite order.

For strict support at h, use Definition 45.348 to identify the support of $\Gamma_{p,q,l,r}(h)$ with the support of $B_{l,r}(h)$ under the empty external endpoint contexts. Then apply Definition 45.421 to obtain $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$.

The strict-support statement for k is identical with k in place of h and p', q' in place of p, q. The same empty-context carrier equivalence strips the endpoints, and the same bilateral support exactness theorem supplies $N^u_D(l) \vee N^u_D(k) \vee N^u_D(r)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction contextual action certificate`
↪ variant `BEDC.Derived.FieldUp.RatupFieldupDenominatorContextualAction certificate`

Theorem 45.457 (RatUp FieldUp denominator contextual action empty-context identity)**.** *For histories h, p, q :* BHist*, assume $C^u_D(h)$,* phsamee*, and* qhsamee*. Then the empty carried-context action with empty external endpoints is carried,*

$$C^u_D(\Gamma_{p,q,\mathsf{e},\mathsf{e}}(h)),$$

it is classified by the original history,

$$R^u_D(\Gamma_{p,q,\mathsf{e},\mathsf{e}}(h), h),$$

and it preserves strict support exactly:

$$N^u_D(\Gamma_{p,q,\mathsf{e},\mathsf{e}}(h)) \Longleftrightarrow N^u_D(h).$$

Proof. Unfold Definition 45.454. With both carried contexts equal to e, the bilateral denominator context is $B_{\mathsf{e},\mathsf{e}}(h)$. The append left and right unit laws reduce the carried spine to the original history, while the endpoint hypotheses $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$ identify the external wrappers with empty endpoints.

Apply Definition 45.456 with $k = h$, $l = \mathsf{e}$, $r = \mathsf{e}$, $p' = \mathsf{e}$, and $q' = \mathsf{e}$. Its classifier component reduces the displayed comparison to $R^u_D(h, h)$, and the denominator classifier is reflexive on the carried input h. The same unfolding and unit laws give the carrier row for $\Gamma_{p,q,\mathsf{e},\mathsf{e}}(h)$.

For strict support, the certificate gives

$$N^u_D(\Gamma_{p,q,\mathsf{e},\mathsf{e}}(h)) \Longleftrightarrow N^u_D(\mathsf{e}) \vee N^u_D(h) \vee N^u_D(\mathsf{e}).$$

Unfold N^u_D as $\mathsf{RatHistoryCarrier}$. Any witness for either empty support disjunct contradicts Definition 37.18 together with reflexivity of hsame at e. Hence the two empty support disjuncts are absent, and the middle disjunct is exactly $N^u_D(h)$. □

Theorem 45.458 (RatUp FieldUp denominator empty-unit action rejects singleton carrier)**.** *For every history $h : \mathsf{BHist}$, if $C^u_D(h)$ and $N^u_D(h)$, then the empty-unit action cannot land in the singleton-empty carrier:*

$$A_{\mathsf{e}}(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h)) \to \bot.$$

Proof. Apply Definition 45.457 with both endpoint witnesses reflexive at e. The strict-support row transports $N^u_D(h)$ to $N^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h))$.

Unfold N^u_D as $\mathsf{RatHistoryCarrier}$. A witness in $A_{\mathsf{e}}(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h))$ is exactly a singleton endpoint witness for the action output, so Definition 37.18 contradicts the transported Rat carrier witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty unit singleton absurd`

Per-object template: traditional target identity element for the denominator contextual action; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, $\mathsf{RatHistoryCarrier}$, append, e, and Γ; carrier C^u_D; classifier R^u_D; stability obligation identity action; ledger empty carried contexts plus empty external endpoint witnesses; core theorem this empty-context identity law; exactness predicate classifier identity and support preservation; standard bridge denominator multiplication identity context; status $\mathsf{ProofTarget}$.

Definition 45.459 (RatUp FieldUp denominator classified-context action)**.** *The classified-context denominator action for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *uses the fixed concrete data*

$$C^u_D := \mathsf{RatDenomUnitCarrier}, \qquad R^u_D := \mathsf{RatDenomUnitClassifier},$$

$$\mu^u_D(h, k) := \mathsf{append}(h, k), \qquad e^u_D := \mathsf{e}, \qquad N^u_D(h) := \mathsf{RatHistoryCarrier}(h).$$

For histories $h, k, l, l', r, r', p, q, p', q' : \mathsf{BHist}$ with classified input rows $R^u_D(h, k)$, $R^u_D(l, l')$, and $R^u_D(r, r')$, and with empty external endpoint contexts $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, its output pair is

$$(\Gamma_{p,q,l,r}(h), \Gamma_{p',q',l',r'}(k)).$$

Thus the left context, core endpoint, and right context are classified before the concrete bilateral denominator action is applied.

Theorem 45.460 (RatUp FieldUp denominator contextual action classifier congruence)**.** *For histories* $h, k, l, l', r, r', p, q, p', q' : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l)$*,* $C^u_D(l')$*,* $C^u_D(r)$*, and* $C^u_D(r')$*. If* $R^u_D(h, k)$*,* $R^u_D(l, l')$*,* $R^u_D(r, r')$*, and* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then the classified-context action of Definition 45.459 satisfies*

$$R^u_D(\Gamma_{p,q,l,r}(h), \Gamma_{p',q',l',r'}(k)).$$

Proof. First remove the empty external endpoints. By Definition 45.405, it is enough to classify $B_{l,r}(h)$ against $B_{l',r'}(k)$, because the four endpoint hypotheses identify the wrapped action outputs with the bilateral carried products.

Unfold $B_{l,r}$ and $B_{l',r'}$. The binary classifier congruence field in the concrete denominator certificate first combines $R^u_D(l, l')$ with $R^u_D(h, k)$, giving

$$R^u_D(\mathsf{append}(l, h), \mathsf{append}(l', k)).$$

The carrier closure rows of the same certificate place both intermediate products in C^u_D.

Apply binary classifier congruence a second time, now with the right-context input $R^u_D(r, r')$. This gives

$$R^u_D(\mathsf{append}(\mathsf{append}(l, h), r), \mathsf{append}(\mathsf{append}(l', k), r')).$$

Folding the bilateral-context notation turns this into $R^u_D(B_{l,r}(h), B_{l',r'}(k))$.

Finally use the reverse direction of Definition 45.405 with the four empty endpoint hypotheses. It wraps the bilateral comparison back into the two displayed Γ outputs, which is the required classifier row. □

Traditional target: denominator bilateral carried-context action for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ with empty external endpoints; **BEDC source**: BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; **carrier**: C^u_D; **classifier**: R^u_D; **action**: $\Gamma_{p,q,l,r}$; **strict support**: N^u_D; **ledger**: carried left and right denominator contexts plus empty external endpoint contexts; **core theorem**: Definition 45.456 and Definition 45.460; **exactness predicate**: bilateral classifier exactness, classified left-core-right action congruence, and bilateral strict-support exactness after empty endpoint transport; **status**: ProofTarget.

Theorem 45.461 (RatUp FieldUp denominator contextual action unit-support exactness)**.** *For histories* $h, l, r, p, q : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(l)$*,* $C^u_D(r)$*,* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $\neg N^u_D(l)$*, and* $\neg N^u_D(r)$*. Then the unit-context strict support of the contextual denominator action is exactly the support of the acted-on history:*

$$N^u_D(\Gamma_{p,q,l,r}(h)) \iff N^u_D(h).$$

Proof. Apply Definition 45.456 with the single acted-on row based at p, q. Its strict-support component gives $N^u_D(\Gamma_{p,q,l,r}(h))$ iff $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$.

For the forward direction, suppose $N^u_D(\Gamma_{p,q,l,r}(h))$. The support row converts this into the three-way disjunction $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$.

The left branch contradicts $\neg N^u_D(l)$, and the right branch contradicts $\neg N^u_D(r)$. The only remaining branch is $N^u_D(h)$.

For the reverse direction, suppose $N^u_D(h)$. Insert this evidence into the middle branch of $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$, then use the same support row in the reverse direction to obtain $N^u_D(\Gamma_{p,q,l,r}(h))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator contextual action unit support iff`

Per-object template: traditional target denominator contextual action for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ under unit carried contexts; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; carrier C^u_D; classifier R^u_D; stability obligation unit-context strict-support exactness; ledger empty endpoint contexts plus carried left and right contexts; core theorem label Definition 45.461; exactness predicate support iff; standard bridge ordinary denominator multiplication with empty endpoint contexts; status ProofTarget.

Theorem 45.462 (RatUp FieldUp denominator contextual action total certificate)**.** *The concrete denominator interface for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is covered by the following fixed* BHist *data:*

$$C^u_D := \mathsf{RatDenomUnitCarrier}, \qquad R^u_D := \mathsf{RatDenomUnitClassifier},$$

$$\mu^u_D(h,k) := \mathsf{append}(h,k), \qquad e^u_D := \mathsf{e}, \qquad N^u_D(h) := \mathsf{RatHistoryCarrier}(h),$$

together with the action $\Gamma_{p,q,l,r}$ *of Definition 45.454. This concrete interface supplies the total denominator action certificate through the following rows:*

1. *the external-unit denominator subcertificate rows of Definition 45.448;*
2. *the multiplicative record rows of Definition 45.447;*
3. *the contextual classifier exactness and strict-support rows of Definition 45.456;*
4. *the classified left-core-right contextual action congruence row of Definition 45.460.*

Thus the denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is represented by one concrete append-based action certificate on the displayed carrier, classifier, unit, strict-support predicate, and contextual action.*

Proof. First, the external-unit denominator subcertificate is exactly Definition 45.448. Its carrier, classifier, multiplication, unit, and support fields are the displayed RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and RatHistoryCarrier data.

Next, the multiplicative rows are supplied by Definition 45.447. That theorem records the same append operation and empty unit for the denominator-side FieldUp multiplication, so its monoid, congruence, strict-support, empty-unit, and cancellation rows attach to the same concrete interface.

Finally, the same-context contextual action rows are those of Definition 45.456. By Definition 45.454, the action is $\Gamma_{p,q,l,r}$ on the displayed carrier and classifier, and its strict-support readback is computed by the same predicate N^u_D.

The classified-context congruence row is Definition 45.460. It uses the same Γ action but allows the left context, core endpoint, and right context to vary along R^u_D before the output classifier is formed.

The four cited results are all stated over the concrete BHist carrier, classifier, append operation, empty endpoint, support predicate, and action data displayed in the theorem. Therefore the certificate contains no remaining abstract operator or parameter-transport schema. □

Definition 45.463 (RatUp FieldUp composed denominator contextual action instance)**.** *The composed concrete denominator contextual action instance for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is the tuple*

$$\mathcal{A}^u_{D,2} := (C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D, \Gamma^{(2)})$$

on the same concrete data

$$C^u_D := \mathsf{RatDenomUnitCarrier}, \qquad R^u_D := \mathsf{RatDenomUnitClassifier},$$

$$\mu^u_D(h,k) := \mathsf{append}(h,k), \qquad e^u_D := \mathsf{e}, \qquad N^u_D(h) := \mathsf{RatHistoryCarrier}(h).$$

For carried contexts $l_1, r_1, l_2, r_2 : \mathsf{BHist}$ *and external endpoint contexts* $p_1, q_1, p_2, q_2 : \mathsf{BHist}$*, its two-layer carried action is*

$$\Gamma^{(2)}_{p_2,q_2,l_2,r_2;p_1,q_1,l_1,r_1}(h) := \Gamma_{p_2,q_2,l_2,r_2}(\Gamma_{p_1,q_1,l_1,r_1}(h)).$$

Theorem 45.464 (RatUp FieldUp denominator contextual action composition)**.** *For histories* $h, k, l_1, r_1, l_2, r_2, p_1, q_1, p_2, q_2, p'_1, q'_1, p'_2, q'_2 : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l_1)$*,* $C^u_D(r_1)$*,* $C^u_D(l_2)$*,* $C^u_D(r_2)$*, and assume* $p_1\mathsf{hsamee}$*,* $q_1\mathsf{hsamee}$*,* $p_2\mathsf{hsamee}$*,* $q_2\mathsf{hsamee}$*,* $p'_1\mathsf{hsamee}$*,* $q'_1\mathsf{hsamee}$*,* $p'_2\mathsf{hsamee}$*, and* $q'_2\mathsf{hsamee}$*. Then the composed contextual action instance of Definition 45.463 satisfies the following classifier and strict-support laws:*

$$R^u_D(\Gamma^{(2)}_{p_2,q_2,l_2,r_2;p_1,q_1,l_1,r_1}(h), \Gamma^{(2)}_{p'_2,q'_2,l_2,r_2;p'_1,q'_1,l_1,r_1}(k)) \Longleftrightarrow R^u_D(h,k),$$

$$N^u_D(\Gamma^{(2)}_{p_2,q_2,l_2,r_2;p_1,q_1,l_1,r_1}(h)) \Longleftrightarrow N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(h) \vee N^u_D(r_1) \vee N^u_D(r_2),$$

and

$$N^u_D(\Gamma^{(2)}_{p'_2,q'_2,l_2,r_2;p'_1,q'_1,l_1,r_1}(k)) \Longleftrightarrow N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(k) \vee N^u_D(r_1) \vee N^u_D(r_2).$$

Proof. First build the carrier facts needed for the second action layer. Carrier closure for the bilateral denominator multiplication, together with empty external endpoint carrier exactness, sends $C^u_D(h)$, $C^u_D(l_1)$, and $C^u_D(r_1)$ to $C^u_D(\Gamma_{p_1,q_1,l_1,r_1}(h))$. The same argument sends $C^u_D(k)$, $C^u_D(l_1)$, and $C^u_D(r_1)$ to $C^u_D(\Gamma_{p'_1,q'_1,l_1,r_1}(k))$.

For classifier exactness, apply Definition 45.456 to the outer layer with carried contexts l_2, r_2 and the two inner-action endpoints. The empty outer endpoint witnesses reduce the classifier comparison to the classifier comparison of the two inner actions.

Apply Definition 45.456 again to the inner layer with carried contexts l_1, r_1. The empty inner endpoint witnesses identify that inner classifier comparison with $R^u_D(h,k)$, giving the displayed classifier equivalence in both directions.

For strict support at h, the outer action certificate gives $N^u_D(l_2) \vee N^u_D(\Gamma_{p_1,q_1,l_1,r_1}(h)) \vee N^u_D(r_2)$. Applying the inner action certificate rewrites the middle term as $N^u_D(l_1) \vee N^u_D(h) \vee N^u_D(r_1)$.

Reassociate the constructive disjunction by the ordinary introduction and case-analysis rules for $\vee$. This yields exactly $N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(h) \vee N^u_D(r_1) \vee N^u_D(r_2)$.

The strict-support statement for k is identical with k in place of h and the primed external endpoint contexts in both layers. The outer certificate, the inner certificate, and the same reassociation of $\vee$ give $N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(k) \vee N^u_D(r_1) \vee N^u_D(r_2)$. □

Per-object template: traditional target composed denominator action certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; carrier C^u_D; classifier R^u_D; action $\Gamma^{(2)}_{p_2,q_2,l_2,r_2;p_1,q_1,l_1,r_1}$; strict support N^u_D; ledger two carried left denominator contexts, two carried right denominator contexts, and two empty external endpoint layers; core theorem Definition 45.464; exactness predicate composed classifier exactness and five-term strict-support readback; standard bridge iterated denominator multiplication with empty external endpoints; status ProofTarget.

Per-object template: traditional target total denominator action certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, Cont, append, and e; carrier C^u_D; classifier R^u_D; stability obligations multiplicative certificate rows plus contextual classifier, classified left-core-right congruence, and strict-support rows; ledger external endpoint bit

plus carried left and right denominator contexts; core theorem Definition 45.462; exactness predicate denominator classifier exactness, classified contextual action congruence, and strict-support readback; standard bridge ordinary denominator multiplication with adjoined empty endpoint; status ProofTarget.

Theorem 45.465 (RatUp FieldUp denominator contextual action composition closure)**.** *For histories $h, l, r, l', r', p, q, p', q' : \mathsf{BHist}$, assume $C^u_D(h)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$, and assume $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$. Then the composite concrete denominator contextual action is carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is classified by the single normalized action with combined carried contexts:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

Its strict-support readback is exactly

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r').$$

Proof. Unfold the action of Definition 45.454. The inner term is $\mathsf{append}(p, \mathsf{append}(B_{l,r}(h), q))$, and the outer action is obtained by placing $B_{l',r'}$ around that term and then adjoining the external endpoints p' and q'.

Carrier closure follows from the denominator continuation monoid laws for C^u_D. The hypotheses on l, h, and r put $B_{l,r}(h)$ in the carrier, and the empty endpoint hypotheses strip the external endpoints of the inner action. Applying the same closure step to l', the carried inner action, and r' puts the composite action in C^u_D.

For the classifier statement, first remove the empty external endpoints with Definition 45.405. The comparison reduces to the carried products $B_{l',r'}(B_{l,r}(h))$ and $B_{\mu^u_D(l',l),\mu^u_D(r,r')}(h)$.

Unfold $B_{l,r}$ and use the denominator multiplicative certificate rows of Definition 45.447 to reassociate the append products. The left carried contexts combine as $\mu^u_D(l', l)$, and the right carried contexts combine as $\mu^u_D(r, r')$.

After this reassociation, apply Definition 45.418 to identify the nested carried action with the normalized single action inside R^u_D.

For strict support, apply Definition 45.456 to the outer action. Its support is $N^u_D(l') \vee N^u_D(\Gamma_{p,q,l,r}(h)) \vee N^u_D(r')$.

Apply Definition 45.456 again to the inner action, or equivalently unfold $B_{l,r}$ and use Definition 45.421. This replaces the middle term by $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$. Reassociate the constructive disjunction to obtain the displayed five-term support readback. □

↪ variant BEDC.Derived.FieldUp.field rat denominator contextual action composition support laws

Theorem 45.466 (Rat denominator envelope bilateral context cancellation)**.** *For the concrete tuple $(C^*_D, R^*_D, \mu^*_D, e^*_D, N^*_D)$ of Definition 45.440, let $h, k, l, r, p, q, p', q' : \mathsf{BHist}$. If $C^*_D(h)$, $C^*_D(k)$, $C^*_D(l)$, $C^*_D(r)$, $N^*_D(l)$, $N^*_D(r)$, $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, then*

$$R^*_D(\mathsf{append}(p, \mathsf{append}(\mu^*_D(l, \mu^*_D(h, r)), q)), \mathsf{append}(p', \mathsf{append}(\mu^*_D(l, \mu^*_D(k, r)), q'))) \Longleftrightarrow R^*_D(h, k).$$

Proof. First build the two bilateral products inside the concrete carrier. The multiplicative certificate boundary of Definition 45.437 uses $C^*_D(l)$, $C^*_D(h)$, $C^*_D(k)$, $C^*_D(r)$, $N^*_D(l)$, and $N^*_D(r)$ to place $\mu^*_D(l, \mu^*_D(h, r))$ and $\mu^*_D(l, \mu^*_D(k, r))$ in C^*_D.

Next strip the external endpoint contexts. Apply Definition 45.441 to these two carried products together with $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$. The classifier component gives

$$R^*_D(\mathsf{append}(p, \mathsf{append}(\mu^*_D(l, \mu^*_D(h, r)), q)), \mathsf{append}(p', \mathsf{append}(\mu^*_D(l, \mu^*_D(k, r)), q'))) \Longleftrightarrow R^*_D(\mu^*_D(l, \mu^*_D(h, r)), \mu^*_D(l, \mu^*$$

Finally apply Definition 45.453 with strict factors l and r. It identifies the remaining uncontextualized bilateral comparison with $R^*_D(h, k)$, and composing the two equivalences proves the displayed statement. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator envelope bilateral context cancellation`

Per-object template: traditional target envelope bilateral context cancellation for the Rat denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, $\mathsf{RatHistoryCarrier}$, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, and e; carrier C^*_D; classifier R^*_D; multiplication μ^*_D; unit e^*_D; strict support N^*_D; ledger empty external endpoint contexts plus strict carried left and right denominator factors; core theorem Definition 45.466; exactness predicate empty-envelope classifier transport followed by bilateral strict-factor cancellation; status $\mathsf{ProofTarget}$.

Theorem 45.467 (RatUp FieldUp denominator contextual action composition)**.** *For histories* $h, k, l, r, l', r', p, q, p', q', s, t, s', t' : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l)$*,* $C^u_D(r)$*,* $C^u_D(l')$*,* $C^u_D(r')$*, and*

$$p\mathsf{hsamee}, \quad q\mathsf{hsamee}, \quad p'\mathsf{hsamee}, \quad q'\mathsf{hsamee}, \quad s\mathsf{hsamee}, \quad t\mathsf{hsamee}, \quad s'\mathsf{hsamee}, \quad t'\mathsf{hsamee}.$$

Then the concrete action Γ *of Definition 45.454 is exact under two carried contextual applications:*

$$R^u_D(\Gamma_{p,q,l,r}(\Gamma_{s,t,l',r'}(h)), \Gamma_{p',q',l,r}(\Gamma_{s',t',l',r'}(k))) \Longleftrightarrow R^u_D(h, k).$$

Its strict-support readbacks are

$$N^u_D(\Gamma_{p,q,l,r}(\Gamma_{s,t,l',r'}(h))) \Longleftrightarrow N^u_D(l) \vee N^u_D(l') \vee N^u_D(h) \vee N^u_D(r') \vee N^u_D(r)$$

and

$$N^u_D(\Gamma_{p',q',l,r}(\Gamma_{s',t',l',r'}(k))) \Longleftrightarrow N^u_D(l) \vee N^u_D(l') \vee N^u_D(k) \vee N^u_D(r') \vee N^u_D(r).$$

Proof. Apply Definition 45.456 to the inner actions with carried contexts l', r' and external contexts s, t, s', t'. This gives the classifier equivalence between the inner endpoints and h, k, and it reads the inner supports as $N^u_D(l') \vee N^u_D(h) \vee N^u_D(r')$ and $N^u_D(l') \vee N^u_D(k) \vee N^u_D(r')$.

The carrier part used in the same action proof puts both inner endpoints back inside C^u_D: carried bilateral product closure handles $B_{l',r'}(h)$ and $B_{l',r'}(k)$, and empty external endpoint transport preserves the displayed carrier. These two carrier witnesses are the inputs needed for the outer action.

Apply Definition 45.456 again to the outer action with carried contexts l, r and external contexts p, q, p', q'. Its classifier row identifies the displayed nested classifier with the classifier of the two inner endpoints. Composing this equivalence with the inner classifier equivalence gives the stated comparison with $R^u_D(h, k)$.

For strict support, the outer action reads $\Gamma_{p,q,l,r}(\Gamma_{s,t,l',r'}(h))$ as $N^u_D(l) \vee N^u_D(\Gamma_{s,t,l',r'}(h)) \vee N^u_D(r)$. Substitute the inner support readback and reassociate the finite disjunction to obtain $N^u_D(l) \vee N^u_D(l') \vee N^u_D(h) \vee N^u_D(r') \vee N^u_D(r)$.

The second support statement is identical with k in place of h and s', t' in place of s, t. The same outer readback, inner readback, and reassociation give $N^u_D(l) \vee N^u_D(l') \vee N^u_D(k) \vee N^u_D(r') \vee N^u_D(r)$. □

Theorem 45.468 (RatUp FieldUp denominator normalized contextual action exactness)**.** *For histories* $h, k, l, r, l', r', p, q, p', q', s, t, s', t' : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l)$*,* $C^u_D(r)$*,* $C^u_D(l')$*,* $C^u_D(r')$*, and*

$$p\mathsf{hsamee}, \quad q\mathsf{hsamee}, \quad p'\mathsf{hsamee}, \quad q'\mathsf{hsamee}, \quad s\mathsf{hsamee}, \quad t\mathsf{hsamee}, \quad s'\mathsf{hsamee}, \quad t'\mathsf{hsamee}.$$

Then the single normalized actions obtained from the combined carried contexts are exact for the denominator classifier:

$$R^u_D(\Gamma_{p,q,\mu^u_D(l,l'),\mu^u_D(r',r)}(h), \Gamma_{p',q',\mu^u_D(l,l'),\mu^u_D(r',r)}(k)) \Longleftrightarrow R^u_D(h,k).$$

The normalized strict-support readbacks are

$$N^u_D(\Gamma_{p,q,\mu^u_D(l,l'),\mu^u_D(r',r)}(h)) \Longleftrightarrow N^u_D(l) \vee N^u_D(l') \vee N^u_D(h) \vee N^u_D(r') \vee N^u_D(r)$$

and

$$N^u_D(\Gamma_{p',q',\mu^u_D(l,l'),\mu^u_D(r',r)}(k)) \Longleftrightarrow N^u_D(l) \vee N^u_D(l') \vee N^u_D(k) \vee N^u_D(r') \vee N^u_D(r).$$

Proof. Apply Definition 45.465 to the h endpoint with inner carried contexts l', r', outer carried contexts l, r, inner external endpoints s, t, and outer external endpoints p, q. The classifier row identifies the nested action $\Gamma_{p,q,l,r}(\Gamma_{s,t,l',r'}(h))$ with the normalized action $\Gamma_{p,q,\mu^u_D(l,l'),\mu^u_D(r',r)}(h)$.

Its support row reads the nested endpoint as the displayed five-term disjunction. Strict-support transport along the concrete classifier R^u_D then gives the same readback for the normalized h endpoint.

Apply the same closure theorem to the k endpoint with inner external endpoints s', t' and outer external endpoints p', q'. This gives the classifier comparison between $\Gamma_{p',q',l,r}(\Gamma_{s',t',l',r'}(k))$ and $\Gamma_{p',q',\mu^u_D(l,l'),\mu^u_D(r',r)}(k)$.

The corresponding support row, transported along R^u_D, gives the strict-support readback containing $N^u_D(k)$ in the middle position.

Now use Definition 45.467 on the two nested endpoints. It supplies the equivalence between their denominator classifier and $R^u_D(h,k)$. Transport this equivalence across the two closure comparisons by symmetry and transitivity for the concrete classifier R^u_D.

The two support statements are the strict-support readbacks already obtained from the endpoint closure applications, with the finite disjunctions reassociated to the displayed order. □

Theorem 45.469 (RatUp FieldUp denominator contextual action threefold normalization)**.** *For histories* $h, l_0, r_0, l_1, r_1, l_2, r_2 : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(l_0)$*,* $C^u_D(r_0)$*,* $C^u_D(l_1)$*,* $C^u_D(r_1)$*,* $C^u_D(l_2)$*, and* $C^u_D(r_2)$*. Let*

$$H_3 := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))).$$

Then the threefold concrete denominator contextual action is carried:

$$C^u_D(H_3).$$

It is classified by the single normalized carried action

$$R^u_D\big(H_3, \Gamma_{\mathsf{e},\mathsf{e},\mu^u_D(l_2,\mu^u_D(l_1,l_0)),\mu^u_D(\mu^u_D(r_0,r_1),r_2)}(h)\big).$$

Its strict-support readback is exactly

$$N^u_D(H_3) \Longleftrightarrow N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(l_0) \vee N^u_D(h) \vee N^u_D(r_0) \vee N^u_D(r_1) \vee N^u_D(r_2).$$

Proof. First apply Definition 45.465 to the inner two contextual layers with carried contexts l_0, r_0 and l_1, r_1. Since all external endpoints are e, the inner composite is carried, classified by $\Gamma_{\mathsf{e},\mathsf{e},\mu^u_D(l_1,l_0),\mu^u_D(r_0,r_1)}(h)$, and has five-term support readback $N^u_D(l_1) \vee N^u_D(l_0) \vee N^u_D(h) \vee N^u_D(r_0) \vee N^u_D(r_1)$.

Next apply the same closure theorem to the outer layer with carried contexts l_2, r_2 and to the normalized inner action as the carried core. The carrier row gives $C^u_D(H_3)$, while the classifier row reduces the threefold term to an outer action on the normalized two-layer core.

Use the denominator multiplicative certificate rows of Definition 45.447 to reassociate the carried append products. The left contexts collect as $\mu^u_D(l_2, \mu^u_D(l_1, l_0))$, and the right contexts collect as $\mu^u_D(\mu^u_D(r_0, r_1), r_2)$, so the classifier comparison is exactly the displayed single normalized action.

For strict support, the outer closure row reads the support of H_3 as $N^u_D(l_2)$, the support of the inner two-layer action, and $N^u_D(r_2)$. Substitute the inner support readback from the first step and reassociate the finite constructive disjunction. This gives the displayed seven-term strict-support formula in the stated order. □

Per-object template: traditional target threefold denominator action coherence for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, $\mathsf{RatHistoryCarrier}$, append, Cont, and e; carrier C^u_D; classifier R^u_D; stability obligations nested action closure, classifier normalization, and strict-support readback; ledger three carried left contexts, three carried right contexts, and empty external endpoint layers; core theorem Definition 45.469; exactness predicate threefold contextual normalization; standard bridge iterated denominator multiplication; status $\mathsf{ProofTarget}$.

Theorem 45.470 (RatUp FieldUp denominator complete contextual row package)**.** *The concrete denominator row $(C^\dagger_D, R^\dagger_D, \mu^\dagger_D, e^\dagger_D, N^\dagger_D)$ of Definition 45.449 and the contextual action Γ of Definition 45.454 jointly supply the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator package consumed by the FieldUp certificate. The package fields are the concrete BHist data*

$$C^\dagger_D = C^u_D = \mathsf{RatDenomUnitCarrier}, \qquad R^\dagger_D = R^u_D = \mathsf{RatDenomUnitClassifier},$$

$$\mu^\dagger_D = \mu^u_D = \mathsf{append}, \qquad e^\dagger_D = e^u_D = \mathsf{e}, \qquad N^\dagger_D = N^u_D = \mathsf{RatHistoryCarrier}.$$

Its row component satisfies all denominator row laws of Definition 45.450; its contextual component satisfies all bilateral contextual classifier and strict-support laws of Definition 45.456; and the combined carrier, classifier, append, unit, and strict-support data are the same concrete data used by the assembled denominator subcertificate of Definition 45.448.

Proof. The two displayed definitions give the field identifications directly. Both rows use $\mathsf{RatDenomUnitCarrier}$ as carrier, $\mathsf{RatDenomUnitClassifier}$ as classifier, native BHist append as the binary operation, e as the unit, and $\mathsf{RatHistoryCarrier}$ as the strict-support predicate.

Apply Definition 45.450 to the row $(C^\dagger_D, R^\dagger_D, \mu^\dagger_D, e^\dagger_D, N^\dagger_D)$. This supplies the envelope, closure, unit, append, congruence, support, product, cancellation, contextual fixed-factor, external-unit, and absent carried-unit clauses for the denominator row.

Apply Definition 45.456 to the same concrete carrier, classifier, append, unit, and support fields, now with the bilateral action Γ. This supplies the contextual classifier law and the two contextual strict-support laws for carried left and right denominator contexts under empty external endpoints.

Finally, Definition 45.448 is the denominator subcertificate consumed by $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. Since its carrier, classifier, multiplication, unit, and strict-support entries are the displayed BHist data above, the row laws and contextual action laws assemble into the announced concrete denominator package. □

Theorem 45.471 (RatUp FieldUp denominator package closure)**.** *For the concrete denominator row* $(C_D^\dagger, R_D^\dagger, \mu_D^\dagger, e_D^\dagger, N_D^\dagger)$ *of Definition 45.449 and the contextual action* Γ *of Definition 45.454, every denominator-side row required by Definition 45.337 at the* $\mathsf{Field}^\uparrow(\mathsf{RatUp})$ *boundary is covered by the row laws of Definition 45.450, the contextual action certificate of Definition 45.456, the obligation map of Definition 45.451, and the row exhaustion theorem Definition 45.452. The concrete carrier, classifier, multiplication, unit, and strict-support data are exactly the data assembled in Definition 45.448.*

Proof. Unfold the concrete denominator row and the contextual action instance. The definitions identify the row fields as RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and RatHistoryCarrier, and they identify the action as the bilateral context action Γ.

Apply Definition 45.470. This packages the row component and the contextual component over the same concrete BHist data, with the row laws coming from Definition 45.450 and the action laws coming from Definition 45.456.

Next apply Definition 45.451. It aligns those concrete row and contextual fields with the denominator rows named by Definition 45.337, so the package is read as the denominator-side interface of the FieldUp certificate rather than as a detached list of laws.

Finally apply Definition 45.452. It states that no denominator-side row used by the assembled $\mathsf{Field}^\uparrow(\mathsf{RatUp})$ certificate lies outside the multiplicative and contextual sources already named above. Together with Definition 45.448, this proves that the concrete denominator-side FieldUp package is closed under exactly the listed components. □

Per-object template: traditional target $\mathsf{Field}^\uparrow(\mathsf{RatUp})$ denominator-side package closure; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, native append, and Γ; carrier $C_D^\dagger$; classifier $R_D^\dagger$; stability obligations denominator row laws, contextual action certificate, obligation alignment, and row exhaustion; ledger empty endpoint, carried denominator histories, and contextual denominator actions; core theorem Definition 45.471; exactness predicate denominator classifier reflection and strict-support readback; standard bridge deferred; status ProofTarget.

Theorem 45.472 (RatUp FieldUp denominator contextual row closure package)**.** *For the concrete denominator tuple* $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$ *given by*

$$C_D^u = \mathsf{RatDenomUnitCarrier}, \qquad R_D^u = \mathsf{RatDenomUnitClassifier},$$

$$\mu_D^u = \mathsf{append}, \qquad e_D^u = \mathsf{e}, \qquad N_D^u = \mathsf{RatHistoryCarrier},$$

and for the action Γ *of Definition 45.454, the* $\mathsf{Field}^\uparrow(\mathsf{RatUp})$ *denominator package contains the row laws and assembly of Definition 45.470, the total contextual action certificate of Definition 45.462, the two-layer composition closure of Definition 45.465, the exact two-layer classifier and strict-support readback of Definition 45.467, the normalized contextual action exactness of Definition 45.468, the threefold contextual normalization of Definition 45.469, and the envelope bilateral cancellation law of Definition 45.466.*

Proof. First cite Definition 45.470. It fixes the concrete BHist carrier, classifier, append operation, empty unit, and strict-support predicate as RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and RatHistoryCarrier, and it supplies the row laws together with the assembled denominator subcertificate.

Next cite Definition 45.462. This is the one-step contextual action certificate for the same carrier, classifier, append, unit, strict-support predicate, and action Γ.

The two carried-action rows are supplied by Definition 45.465 and Definition 45.467. The first theorem gives closure of two carried contextual applications in C^u_D; the second gives classifier reflection and the five-term strict support readback for the exact two-layer action.

Normalized two-layer exactness is supplied by Definition 45.468. It composes the composition-closure row with the exact two-layer theorem: the closure row identifies the nested action with the single action on appended carried contexts, and the exactness row transports classifier equivalence and strict-support readback to the normalized endpoints.

The threefold law is supplied by Definition 45.469. It applies the same closure twice and uses the concrete denominator multiplicative rows to reassociate the append contexts, so three carried contextual layers are classified by one normalized action with the displayed seven-term strict-support readback.

Finally apply Definition 45.466. It supplies cancellation after strict carried denominator factors and empty external endpoints, so the concrete denominator package records row laws, contextual action, two-layer composition, exact readback, normalized exactness, threefold normalization, and bilateral cancellation over the same BHist data. □

Per-object template: traditional target FieldUp denominator contextual row closure for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and the action Γ; carrier C^u_D; classifier R^u_D; stability obligations denominator row laws, one-step contextual action rows, two-layer contextual composition, exact strict-support readback, normalized action exactness, three carried-context normalization, and strict-factor cancellation; ledger empty external endpoints, carried left and right denominator contexts, normalized two-layer append contexts, and three carried left and right context layers; core theorem Definition 45.472 with exactness rows Definition 45.468 and Definition 45.469; exactness predicate denominator classifier reflection, contextual action support readback, normalized classifier exactness, three carried-context strict-support normalization, and envelope bilateral cancellation; standard bridge FieldUp denominator boundary over $\mathsf{Rat}^{\uparrow}$ histories; status ProofTarget.

Theorem 45.473 (RatUp FieldUp denominator contextual action unit-composition laws)**.** *For the concrete data of Definition 45.454, the empty contextual action is a classifier unit and contextual actions compose by context multiplication.*

If $C^u_D(h)$, then the empty action is carried and classifier-equal to the original history:

$$C^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h)),$$

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h), h),$$

and its strict support is unchanged:

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h)) \Longleftrightarrow N^u_D(h).$$

If $C^u_D(h)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$, and if $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, then the nested contextual action is carried:

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is classified by the single action whose carried contexts are multiplied inside the denominator context monoid:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

Its strict-support readback is the five-term support law

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r').$$

Proof. For the unit action, unfold Definition 45.454. The four contexts are empty, so the action is the bilateral multiplication context $B_{\mathsf{e},\mathsf{e}}(h)$ with empty external endpoints. The empty-unit denominator laws of Definition 45.347 put the result in C^u_D.

The classifier clause is the empty-context instance of Definition 45.456, using empty-context classifier exactness for the endpoint transport. With both carried contexts equal to e, the same denominator unit laws reduce the comparison to $R^u_D(h,h)$, which is the reflexive classifier comparison for the carried history h.

For strict support of the unit action, apply the strict-support part of Definition 45.456. The two outer support terms are $N^u_D(\mathsf{e})$ and $N^u_D(\mathsf{e})$, and the empty-unit denominator laws make them absent. The remaining term is exactly $N^u_D(h)$.

For composition, use Definition 45.465. Its hypotheses are precisely the carried-context hypotheses and the four empty external endpoint hypotheses displayed above.

That theorem supplies the carried nested action, the classifier comparison with $\Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)$, and the five-term strict-support readback. These are exactly the action-law package consumed by the concrete FieldUp denominator interface. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty unit laws`

Theorem 45.474 (RatUp FieldUp denominator contextual row package with unit-composition laws)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *given by*

$$C^u_D = \mathsf{RatDenomUnitCarrier}, \qquad R^u_D = \mathsf{RatDenomUnitClassifier},$$

$$\mu^u_D = \mathsf{append}, \qquad e^u_D = \mathsf{e}, \qquad N^u_D = \mathsf{RatHistoryCarrier},$$

and for the contextual action Γ *of Definition 45.454, the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *denominator package contains the rows of Definition 45.472 together with the empty-action classifier unit, unchanged strict support, context-multiplication composition, and five-term support law of Definition 45.473.*

Proof. Apply Definition 45.472. It records the row laws, action exactness, two-layer composition readback, and cancellation clauses for the concrete tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the action Γ.

Apply Definition 45.473 over the same concrete data. Its hypotheses and conclusions use the same BHist carrier, classifier, append operation, empty unit, strict-support predicate, and contextual action fixed in Definition 45.454.

Thus the two cited results occupy independent rows of the same denominator interface: the first supplies row closure, action exactness, readback, and cancellation, while the second supplies the empty-action unit, unchanged strict support, context-multiplication composition, and five-term support law. The package statement is their joint registration over the shared concrete RatUp denominator data, without restating either theorem. □

Theorem 45.475 (RatUp FieldUp denominator normalized contextual action package)**.** *For the concrete tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *determined by* $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, e, *and* $\mathsf{RatHistoryCarrier}$, *and for the concrete contextual action* Γ *of Definition 45.454, the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *denominator action package contains:*

1. *the row closure package of Definition 45.472;*
2. *the empty-action unit law and context-composition normalization law of Definition 45.473;*
3. *the paired normalized classifier exactness and five-term strict-support readbacks of Definition 45.468.*

Thus the normalized denominator contextual action certificate is available as a single non-parametric package over the same carried denominator contexts and empty external endpoints.

Proof. Apply Definition 45.472. This supplies the concrete row closure package for the RatUp FieldUp denominator carrier and classifier.

Apply Definition 45.473. It gives the empty-action unit clause and the context-composition normalization clause for the same action Γ.

Apply Definition 45.468. It supplies the paired normalized classifier exactness together with the five-term strict-support readbacks.

The three cited results use the same concrete BHist carrier, RatDenomUnitClassifier, append operation, empty unit e, strict-support predicate N^u_D, and action Γ. Therefore their clauses assemble without transport into the stated denominator action package. □

Per-object template: traditional target normalized denominator contextual action package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and Γ; carrier C^u_D; classifier R^u_D; stability obligations row closure, action unit, action composition normalization, normalized classifier exactness, and strict-support readback; ledger carried left and right denominator contexts plus empty external endpoints; core theorem Definition 45.475; exactness predicate normalized classifier iff and five-term support iff; standard bridge ordinary denominator context multiplication; status ProofTarget.

Theorem 45.476 (RatUp FieldUp denominator contextual action support collapse)**.** *For the concrete tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, suppose $C^u_D(h)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$. Suppose also that $p$hsamee, $q$hsamee, p'hsamee, and q'hsamee, and that $\neg N^u_D(l)$, $\neg N^u_D(r)$, $\neg N^u_D(l')$, and $\neg N^u_D(r')$. Then the nested action is carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is classified by the collected denominator-context action:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

Under the same no-extra-support hypotheses, strict support collapses to the support of the acted-on history:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(h).$$

Proof. Apply Definition 45.473 to the displayed carried history, carried denominator contexts, and empty external endpoint equations. The theorem gives the carried nested action and the classifier comparison with the collected denominator-context action.

It also gives the five-term strict-support readback

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r').$$

For the forward direction of the displayed collapse, the four context-support disjuncts contradict the hypotheses $\neg N^u_D(l')$, $\neg N^u_D(l)$, $\neg N^u_D(r)$, and $\neg N^u_D(r')$. The only remaining disjunct is $N^u_D(h)$.

For the reverse direction, insert $N^u_D(h)$ into the middle disjunct of the five-term readback and close by the same equivalence. Thus the contextual action adds no strict support when all four carried contexts have no strict support. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup denominator contextual action support collapse`

Theorem 45.477 (RatUp FieldUp denominator contextual action total normalization package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator action package contains the following single certificate-level normalization record.*

1. *The empty action is the classifier unit. If $C^u_D(h)$, then*

$$C^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h)),$$

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h), h),$$

and

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},\mathsf{e},\mathsf{e}}(h)) \Longleftrightarrow N^u_D(h).$$

2. *The two-layer contextual action normalizes to one collected carried action. If $C^u_D(h)$, $C^u_D(k)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$, and if all displayed external endpoints are* hsamee*, then*

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

For any second endpoint k under the same collected carried contexts, denominator classifier exactness is

$$R^u_D(\Gamma_{s,t,\mu^u_D(l',l),\mu^u_D(r,r')}(h), \Gamma_{s',t',\mu^u_D(l',l),\mu^u_D(r,r')}(k)) \Longleftrightarrow R^u_D(h,k).$$

The nested endpoint has the five-term strict-support readback

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r').$$

3. *The threefold empty-external action normalizes to one collected action. For $C^u_D(h)$ and carried contexts $l_0, r_0, l_1, r_1, l_2, r_2$, let*

$$H_3 := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))).$$

Then

$$C^u_D(H_3),$$

$$R^u_D(H_3, \Gamma_{\mathsf{e},\mathsf{e},\mu^u_D(l_2,\mu^u_D(l_1,l_0)),\mu^u_D(\mu^u_D(r_0,r_1),r_2)}(h)),$$

and

$$N^u_D(H_3) \Longleftrightarrow N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(l_0) \vee N^u_D(h) \vee N^u_D(r_0) \vee N^u_D(r_1) \vee N^u_D(r_2).$$

4. *The same record includes the surrounding row-closure assembly, the one-step action-certificate rows, the exact readback rows, and the bilateral cancellation fields listed in Definition 45.472.*

Proof. The unit and two-layer action laws are exactly the concrete rows of Definition 45.473. They supply the carried empty action, classifier identity, unchanged strict support, the collected two-layer classifier comparison, and the five-term strict-support readback.

For classifier exactness of the collected two-layer endpoint, apply Definition 45.468 with the inner and outer carried contexts ordered so that the collected left context is $\mu^u_D(l', l)$ and the collected right context is $\mu^u_D(r, r')$. Its normalized row gives the displayed denominator classifier equivalence, and its support rows agree with the five-term readback already recorded above.

The threefold row is Definition 45.469. It packages the carried threefold endpoint, the classifier comparison with the single collected action, and the seven-term support readback for the empty external layers.

Finally use Definition 45.472. That theorem supplies the surrounding row package, action certificate, exact readback, and bilateral cancellation fields over the same concrete denominator tuple and the same action Γ, so the displayed action-normalization clauses form one certificate-level FieldUp denominator package. □

Theorem 45.478 (RatUp FieldUp denominator total normalization with support collapse)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$, $C^u_D(k)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, *and* $C^u_D(r')$. *Suppose also that* $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, $q'\mathsf{hsamee}$, *and that* $\neg N^u_D(l)$, $\neg N^u_D(r)$, $\neg N^u_D(l')$, *and* $\neg N^u_D(r')$. *Then the nested action is carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is classified by the collected denominator-context action:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

Its strict support is exactly the strict support of the acted-on history:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \iff N^u_D(h).$$

Moreover, for any second endpoint k *under the same collected carried contexts, and for* s, t, s', t' *satisfying* $s\mathsf{hsamee}$, $t\mathsf{hsamee}$, $s'\mathsf{hsamee}$, *and* $t'\mathsf{hsamee}$, *the collected action remains classifier-exact:*

$$R^u_D(\Gamma_{s,t,\mu^u_D(l',l),\mu^u_D(r,r')}(h), \Gamma_{s',t',\mu^u_D(l',l),\mu^u_D(r,r')}(k)) \iff R^u_D(h,k).$$

Proof. First apply Definition 45.477. Its two-layer row supplies the carried nested action and the classifier comparison with the collected denominator-context action.

The same package cites the normalized contextual action row over the collected left context $\mu^u_D(l', l)$ and collected right context $\mu^u_D(r, r')$. Thus the collected action remains classifier-exact against the second endpoint k with empty external endpoints.

Finally apply Definition 45.476 to the four carried contexts and the four no-extra-support hypotheses. This gives the displayed equivalence between the strict support of the nested action and the strict support of h, so the certificate row records support neutrality in the empty-support contextual case. □

45.20 Rat denominator context-pair action laws

Definition 45.479 (RatUp FieldUp denominator context-pair monoid)**.** *The concrete denominator context-pair carrier for the RatUp FieldUp action is the class of pairs* $(l, r) : \mathsf{BHist} \times \mathsf{BHist}$ *such that* $C^u_D(l)$ *and* $C^u_D(r)$. *Its unit is* (e^u_D, e^u_D), *and its pair product is*

$$(l, r) \star (l', r') := (\mu^u_D(l', l), \mu^u_D(r, r')).$$

The pair support predicate is

$$\mathsf{PairSupp}^u_D(l, r) \iff N^u_D(l) \vee N^u_D(r).$$

The componentwise pair classifier is

$$\mathsf{PairClass}^u_D((l, r), (l', r')) \iff R^u_D(l, l') \wedge R^u_D(r, r').$$

Theorem 45.480 (RatUp FieldUp denominator context-pair operation stability)**.** *For the concrete denominator context-pair structure of Definition 45.479, let* (l, r), (m, s), (l', r'), *and* (m', s') *be carried pairs. If* $\mathsf{PairClass}^u_D((l, r), (m, s))$ *and* $\mathsf{PairClass}^u_D((l', r'), (m', s'))$, *then the two products* $(l, r) \star (l', r')$ *and* $(m, s) \star (m', s')$ *are carried pairs and satisfy*

$$\mathsf{PairClass}^u_D((l, r) \star (l', r'), (m, s) \star (m', s')).$$

Moreover the support of the first product is exactly the support of its two input pairs:

$$\mathsf{PairSupp}^u_D((l, r) \star (l', r')) \iff \mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r').$$

Proof. Unfold the product. The first product is $(\mu^u_D(l', l), \mu^u_D(r, r'))$, and the second is $(\mu^u_D(m', m), \mu^u_D(s, s'))$. Since all four input pairs are carried, Definition 45.347 gives carrier closure for each of the four displayed product components. Thus both pair products are carried.

Unfold $\mathsf{PairClass}^u_D$. The hypotheses give $R^u_D(l, m)$, $R^u_D(r, s)$, $R^u_D(l', m')$, and $R^u_D(r', s')$. Apply Definition 45.131 to the left contexts l', l, m', and m to obtain $R^u_D(\mu^u_D(l', l), \mu^u_D(m', m))$.

Apply the same binary classifier congruence theorem to the right contexts r, r', s, and s' to obtain $R^u_D(\mu^u_D(r, r'), \mu^u_D(s, s'))$. These two component classifiers are exactly the displayed product-pair classifier.

It remains to read support for the first product. Unfold $\mathsf{PairSupp}^u_D$ and use Definition 45.399 on the left component $\mu^u_D(l', l)$ and on the right component $\mu^u_D(r, r')$. This rewrites the product support as $N^u_D(l') \vee N^u_D(l) \vee N^u_D(r) \vee N^u_D(r')$.

Rebracket and commute the disjunctions to $(N^u_D(l) \vee N^u_D(r)) \vee (N^u_D(l') \vee N^u_D(r'))$. By the definition of $\mathsf{PairSupp}^u_D$, this is precisely $\mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r')$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair operation stability`

Theorem 45.481 (RatUp FieldUp denominator context-pair product operation row)**.** *For the concrete denominator context-pair structure of Definition 45.479, let (l, r), (m, s), (l', r'), and (m', s') be carried pairs. If $\mathsf{PairClass}^u_D((l, r), (m, s))$ and $\mathsf{PairClass}^u_D((l', r'), (m', s'))$, then the product components $\mu^u_D(l', l)$, $\mu^u_D(r, r')$, $\mu^u_D(m', m)$, and $\mu^u_D(s, s')$ are carried, and the two product pairs are componentwise R^u_D-classified:*

$$\mathsf{PairClass}^u_D((l, r) \star (l', r'), (m, s) \star (m', s')).$$

The strict support of the first product pair, read as the bundled denominator context, is exactly the disjunction of the two input pair supports:

$$N^u_D(\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))) \iff \mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r').$$

Proof. The two pair-class hypotheses give the four component classifiers $R^u_D(l, m)$, $R^u_D(r, s)$, $R^u_D(l', m')$, and $R^u_D(r', s')$. Apply Definition 45.480; its carrier and classifier clauses are exactly the four carried product components and the displayed product-pair classifier.

For support, apply Definition 45.571. This gives $N^u_D(l') \vee N^u_D(l) \vee N^u_D(r) \vee N^u_D(r')$ for the bundled product context. Rebracket and commute the disjunctions to $(N^u_D(l) \vee N^u_D(r)) \vee (N^u_D(l') \vee N^u_D(r'))$, then unfold $\mathsf{PairSupp}^u_D$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair product operation stable`

Theorem 45.482 (RatUp FieldUp denominator context-pair action laws)**.** *For the concrete denominator context-pair structure of Definition 45.479, the contextual action Γ is a unital action up to the denominator classifier and composes by the product $\star$.*

If $C^u_D(h)$, then the unit pair acts as the classifier identity:

$$C^u_D(\Gamma_{\mathsf{e},\mathsf{e},e^u_D,e^u_D}(h)),$$

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},e^u_D,e^u_D}(h), h),$$

and its strict support is the support of h together with the unit-pair support:

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},e^u_D,e^u_D}(h)) \iff \mathsf{PairSupp}^u_D(e^u_D, e^u_D) \vee N^u_D(h).$$

If $C^u_D(h)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$, and if $p$hsamee, $q$hsamee, p'hsamee, and q'hsamee, then nested actions are classified by the single action of the product pair:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

The strict support of the nested action is exactly the support of the acted history together with the supports of the two acting pairs:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \iff \mathsf{PairSupp}^u_D(l,r) \vee N^u_D(h) \vee \mathsf{PairSupp}^u_D(l',r').$$

Equivalently, by the denominator row support laws for μ^u_D, this is the support of h together with the support of the product pair $(l,r) \star (l',r')$.

The product $\star$ has (e^u_D, e^u_D) as a two-sided unit and is associative up to componentwise R^u_D. For every carried pair (l,r), both $(e^u_D, e^u_D) \star (l,r)$ and $(l,r) \star (e^u_D, e^u_D)$ are componentwise R^u_D-classified with (l,r). For carried pairs (l,r), (l',r'), and (l'',r''), the pairs $((l,r) \star (l',r')) \star (l'',r'')$ and $(l,r) \star ((l',r') \star (l'',r''))$ are componentwise R^u_D-classified.

Proof. The carrier and product clauses are componentwise applications of the empty-unit denominator continuation monoid laws of Definition 45.347. The unit pair is carried because $C^u_D(e^u_D)$ holds, and product closure follows from closure of μ^u_D on each component.

For the unit action, specialize Definition 45.473 to the empty contextual action. Since $e^u_D = \mathsf{e}$ in the concrete denominator instance, its carrier, classifier-identity, and support clauses give exactly the three displayed unit-action conclusions. The pair-support definition rewrites the two empty context supports as $\mathsf{PairSupp}^u_D(e^u_D, e^u_D)$.

For composition, use the same unit-composition theorem with the carried context pairs (l,r) and (l',r') and the four empty external endpoint hypotheses. Its classifier clause is precisely the comparison with the action whose contexts are $\mu^u_D(l',l)$ and $\mu^u_D(r,r')$, which is the pair product $(l,r) \star (l',r')$.

The strict-support clause of that theorem gives the five-term readback $N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r')$. Rebracketing disjunction and unfolding $\mathsf{PairSupp}^u_D$ gives the displayed pair-support form. The denominator row support laws for μ^u_D identify the left two and right two support terms with the two components of the product pair.

Finally, unitality and associativity of $\star$ are the componentwise unit and associativity clauses of Definition 45.347. The left component uses the left-context order in the definition of $\star$, and the right component uses the right-context order; both comparisons are exactly componentwise R^u_D. □

Theorem 45.483 (RatUp FieldUp denominator context-pair unit support neutrality)**.** *For the concrete denominator context-pair tuple $C^u_D = \mathsf{RatDenomUnitCarrier}$, $R^u_D = \mathsf{RatDenomUnitClassifier}$, $\mu^u_D = \mathsf{append}$, $e^u_D = \mathsf{e}$, and $N^u_D = \mathsf{RatHistoryCarrier}$ of Definition 45.479, the unit pair has no strict pair support:*

$$\neg\, \mathsf{PairSupp}^u_D(e^u_D, e^u_D).$$

Consequently, for every carried denominator history h, the unit context-pair action is strict-support neutral:

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},e^u_D,e^u_D}(h)) \iff N^u_D(h).$$

Proof. Unfold $\mathsf{PairSupp}^u_D$ at (e^u_D, e^u_D). Since the concrete unit is $e^u_D = \mathsf{e}$ and $N^u_D = \mathsf{RatHistoryCarrier}$, either branch of the support disjunction would give $\mathsf{RatHistoryCarrier}(\mathsf{e})$. Definition 37.18 rules this out, so the unit pair has no strict pair support.

Now specialize Definition 45.482 to the unit pair. Its unit-action support clause gives

$$N_D^u(\Gamma_{\mathsf{e},\mathsf{e},e_D^u,e_D^u}(h)) \iff \mathsf{PairSupp}_D^u(e_D^u, e_D^u) \vee N_D^u(h).$$

For the forward direction, eliminate the impossible left disjunct using the first paragraph. The remaining disjunct is exactly $N_D^u(h)$.

For the reverse direction, use the right disjunct of the displayed support readback. This proves the stated strict-support neutrality of the unit context-pair action. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair unit support neutrality`

Theorem 45.484 (RatUp FieldUp denominator context-pair commutative support)**.** *For the concrete denominator context-pair structure of Definition 45.479, let (l, r) and (l', r') be carried pairs, so $C_D^u(l)$, $C_D^u(r)$, $C_D^u(l')$, and $C_D^u(r')$. Then the two pair products $(l, r) \star (l', r')$ and $(l', r') \star (l, r)$ are componentwise R_D^u-classified:*

$$R_D^u(\mu_D^u(l', l), \mu_D^u(l, l')) \quad \textit{and} \quad R_D^u(\mu_D^u(r, r'), \mu_D^u(r', r)).$$

Moreover, the support of the product pair is exactly the support of either input pair:

$$\mathsf{PairSupp}_D^u(\mu_D^u(l', l), \mu_D^u(r, r')) \iff \mathsf{PairSupp}_D^u(l, r) \vee \mathsf{PairSupp}_D^u(l', r').$$

Proof. Unfold the definition of $\star$. The left component of $(l, r) \star (l', r')$ is $\mu_D^u(l', l)$, while the left component of $(l', r') \star (l, r)$ is $\mu_D^u(l, l')$. Apply Definition 45.392 to the carried endpoints l' and l to obtain the displayed left-component classifier.

The right component is identical in form, with r and r' in place of the two left contexts. Applying the same commutative monoid law gives $R_D^u(\mu_D^u(r, r'), \mu_D^u(r', r))$. Together the two component classifiers are exactly componentwise R_D^u-classification of the two pair products.

It remains to read support. Unfold $\mathsf{PairSupp}_D^u(\mu_D^u(l', l), \mu_D^u(r, r'))$ as the disjunction of the left and right product supports. For each product component, use Definition 45.399 with the corresponding carried factors, together with the strict-support interpretation of N_D^u, to replace product support by the disjunction of the two input supports.

The result is $N_D^u(l') \vee N_D^u(l) \vee N_D^u(r) \vee N_D^u(r')$. Rebracket and commute the outer disjunctions to obtain $(N_D^u(l) \vee N_D^u(r)) \vee (N_D^u(l') \vee N_D^u(r'))$, which is exactly $\mathsf{PairSupp}_D^u(l, r) \vee \mathsf{PairSupp}_D^u(l', r')$ by the pair-support definition. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair product commutative support`

Per-object template: traditional target context-pair commutative action support; BEDC source BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, append, RatHistoryCarrier, and action Γ; carrier the carried context pairs; classifier componentwise R_D^u; stability obligations component commutativity and support exactness; ledger left and right denominator contexts; core theorem Definition 45.484; exactness predicate the displayed support iff; standard bridge denominator context-pair monoid support; status ProofTarget.

Theorem 45.485 (RatUp FieldUp denominator context-pair threefold action coherence)**.** *For the concrete denominator context-pair structure of Definition 45.479, assume that (l_0, r_0), (l_1, r_1), and (l_2, r_2) are carried pairs and that $C_D^u(h)$. Let*

$$H_3 := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))).$$

Then the threefold context-pair action is carried:

$$C^u_D(H_3).$$

It is classified by the single action of the left-associated double product $((l_0, r_0) \star (l_1, r_1)) \star (l_2, r_2)$*:*

$$R^u_D(H_3, \Gamma_{\mathsf{e},\mathsf{e},\mu^u_D(l_2,\mu^u_D(l_1,l_0)),\mu^u_D(\mu^u_D(r_0,r_1),r_2)}(h)).$$

It is also classified by the single action of the alternate bracketing $(l_0, r_0) \star ((l_1, r_1) \star (l_2, r_2))$*:*

$$R^u_D(H_3, \Gamma_{\mathsf{e},\mathsf{e},\mu^u_D(\mu^u_D(l_2,l_1),l_0),\mu^u_D(r_0,\mu^u_D(r_1,r_2))}(h)).$$

The strict-support readback is the seven-term denominator row

$$N^u_D(H_3) \iff N^u_D(l_2) \vee N^u_D(l_1) \vee N^u_D(l_0) \vee N^u_D(h) \vee N^u_D(r_0) \vee N^u_D(r_1) \vee N^u_D(r_2).$$

Equivalently, after reassociating the finite disjunction, the same readback is

$$N^u_D(H_3) \iff \mathsf{PairSupp}^u_D(l_2, r_2) \vee \mathsf{PairSupp}^u_D(l_1, r_1) \vee \mathsf{PairSupp}^u_D(l_0, r_0) \vee N^u_D(h).$$

Proof. Apply Definition 45.469 with the three carried denominator context pairs and the carried denominator history h. This gives the carrier row for H_3, the classifier comparison with the single action whose contexts are $\mu^u_D(l_2, \mu^u_D(l_1, l_0))$ and $\mu^u_D(\mu^u_D(r_0, r_1), r_2)$, and the seven-term strict-support readback.

Unfold Definition 45.479. The product $(l_0, r_0) \star (l_1, r_1)$ is $(\mu^u_D(l_1, l_0), \mu^u_D(r_0, r_1))$, so the next product with (l_2, r_2) has exactly the two normalized contexts displayed in the first classifier row.

For the alternate bracketing, use the associativity clause of Definition 45.482. It identifies the two double products componentwise up to R^u_D, with left component $\mu^u_D(l_2, \mu^u_D(l_1, l_0))$ classified with $\mu^u_D(\mu^u_D(l_2, l_1), l_0)$ and right component $\mu^u_D(\mu^u_D(r_0, r_1), r_2)$ classified with $\mu^u_D(r_0, \mu^u_D(r_1, r_2))$.

Transport the first single-action classifier across this componentwise associativity comparison using the concrete denominator classifier laws recorded in the same action-law theorem. Finally unfold $\mathsf{PairSupp}^u_D$ and reassociate the finite disjunction to obtain the pair-support formulation of the same strict-support readback. □

↪ variant `BEDC.Derived.FieldUp.RatDenomUnitContextualAction nested pair action coherence`

Per-object template: traditional target threefold context-pair action coherence for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, RatDenomUnitCarrier, RatDenomUnitClassifier, RatHistoryCarrier, append, Cont, e, and Γ; carrier carried context pairs; classifier componentwise R^u_D; stability obligations closure, classifier normalization, support readback, and pair-product associativity; ledger three carried pairs plus acted denominator history; core theorem Definition 45.485; exactness predicate threefold normalized action classifier and support readback; standard bridge FieldUp denominator context-pair action; status ProofTarget.

Theorem 45.486 (RatUp FieldUp denominator context-pair adjacent-swap coherence)**.** *For the concrete denominator context-pair structure of Definition 45.479, assume that* (l_0, r_0)*,* (l_1, r_1)*, and* (l_2, r_2) *are carried pairs and that* $C^u_D(h)$*. Let*

$$H_{012} := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))).$$

Let

$$H_{102} := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(h))).$$

Then the two threefold denominator histories are denominator-classified:

$$R_D^u(H_{012}, H_{102}).$$

Their strict-support readbacks agree:

$$N_D^u(H_{012}) \Longleftrightarrow N_D^u(H_{102}).$$

Proof. Apply Definition 45.497 to the two inner empty-endpoint actions with carried pairs (l_0, r_0) and (l_1, r_1). This gives a denominator classifier between $\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))$ and $\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(h))$.

Transport this inner classifier through the common outer pair (l_2, r_2) by Definition 45.500, using reflexivity for the outer left and right denominator contexts. The transported classifier is exactly $R_D^u(H_{012}, H_{102})$ after unfolding the two displayed definitions.

The denominator classifier contains the history-sameness field for the two acted histories. Reading the strict-support predicate N_D^u through that field in both directions gives $N_D^u(H_{012}) \Longleftrightarrow N_D^u(H_{102})$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty pair adjacent swap coherence`

Per-object template: traditional target adjacent-swap coherence for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator context-pair actions; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, $\mathsf{RatHistoryCarrier}$, append, Cont, e, and Γ; carrier carried context pairs; classifier componentwise R_D^u; stability obligations inner pair commutation plus outer classifier transport; ledger three carried pairs and acted denominator history; core theorem Definition 45.486; exactness predicate adjacent-swap classifier and strict-support invariance; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.487 (RatUp FieldUp denominator action envelope certificate component)**.** *For the concrete denominator tuple* ($\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, e, $\mathsf{RatHistoryCarrier}$) *and the contextual action* Γ *of Definition 45.454, the denominator-action component required by Definition 45.337 is supplied by the envelope closure of Definition 45.569, the normalized row exhaustion of Definition 45.570, and the context-pair action certificate of Definition 45.568. Consequently the assembled* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *boundary may treat the multiplicative denominator coverage, the contextual boundary, the normalized row coverage, and the context-pair product and action rows as one concrete BHist-anchored certificate component.*

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupDenominatorActionEnvelopeCertificateComponent`

Proof. First apply Definition 45.569. It fixes the multiplicative coverage and contextual multiplicative boundary over the RatUp denominator carrier, classifier, append operation, empty endpoint, strict-support predicate, and contextual action Γ.

Next apply Definition 45.570. It accounts for every normalized denominator action row by the same concrete BHist carrier data, the empty external contexts, and the pair product $\star$.

Finally apply Definition 45.568. It supplies the pair classifier, pair product, unit-pair action, product-action composition, support readback, and iterated action rows for the same ($\mathsf{RatDenomUnitCarrier}$, RatDen tuple and the same action Γ.

All three cited results use identical carrier, classifier, append, empty unit, strict-support predicate, and contextual-action data. Their conclusions therefore occupy one denominator-action component of Definition 45.337, with no transport between separate FieldUp sources. □

Definition 45.488 (Concrete RatUp FieldUp source tuple)**.** *The concrete source tuple for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is denoted* $\mathsf{RatUpFieldSource}_F$*. Its field-level BHist source is*

$$C_F = \mathsf{RatHistoryCarrier}, \qquad R_F = \mathsf{RatHistoryClassifier}.$$

Its denominator coordinate is the concrete tuple

$$(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D) = (\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier}).$$

Its contextual-action coordinate is the action Γ *of Definition 45.454. Its context-pair coordinate is the carried pair carrier, componentwise classifier, unit pair, product* $\star$*, and support predicate of Definition 45.479. Its affine coordinate is the certified affine automorphism row source* AffAut_F *of Definition 45.560, with the group certificate rows supplied by Definition 45.558.*

Theorem 45.489 (RatUp FieldUp assembled certificate boundary)**.** *The named source tuple* $\mathsf{RatUpFieldSource}_F$ *of Definition 45.488 satisfies Definition 45.337. Its carrier is* RatHistoryCarrier*, its classifier is* RatHistoryClassifier*, its denominator source is the concrete RatUp denominator tuple, its contextual source is* Γ*, its context-pair source is the monoid of Definition 45.479, and its affine source is* AffAut_F*. The jointly registered rows are:*

1. *the commutative-ring rows inherited from* $\mathsf{CommRing}^{\uparrow}$ *on the same history carrier and classifier;*
2. *the inverse and nonzero rows supplied by Definition 45.205, Definition 45.210, Definition 45.218, and Definition 45.288;*
3. *the multiplicative denominator rows supplied by Definition 45.447 and Definition 45.448;*
4. *the denominator-action component consumed by Definition 45.337, supplied by Definition 45.487 from envelope closure, normalized row exhaustion, and context-pair action certification;*
5. *the contextual denominator rows supplied by Definition 45.472, Definition 45.475, Definition 45.482, Definition 45.568, and Definition 45.570;*
6. *the affine automorphism closure supplied by Definition 45.558.*

Consequently $\mathsf{RatUpFieldSource}_F$ *supplies the Phase Algebra Hierarchy paper-certificate object for the concrete BHist-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate. The machine-checked phase exit is the canonical Lean certificate site for that same object.*

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAssembledCertificateBoundary`

Proof. Unpack Definition 45.488. The carrier and classifier fields are the concrete RatUp history predicates RatHistoryCarrier and RatHistoryClassifier, so every row below is read over the same BHist source rather than over detached parameters.

The denominator coordinate is (RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, RatHistoryCarrier), the contextual-action coordinate is Γ, and the context-pair coordinate is the carried pair monoid of Definition 45.479. The affine coordinate is AffAut_F with rows read from Definition 45.558.

The algebraic spine is the inherited $\mathsf{CommRing}^{\uparrow}$ structure together with the inverse block. Apply Definition 45.205, Definition 45.210, Definition 45.218, and Definition 45.288. These rows give classifier exactness for inverse formation, nonzero transport through the inverse domain, product inverse exactness, and certified multiplication by nonzero factors.

The denominator subcertificate is then read from Definition 45.447 and Definition 45.448. The first theorem records the multiplicative denominator rows, and the second identifies them as the denominator component consumed by the FieldUp certificate obligations.

Now apply Definition 45.487. It combines the envelope closure, normalized row exhaustion, and context-pair action certificate into the single denominator-action component required by Definition 45.337.

The contextual extension first uses Definition 45.472 and Definition 45.475. These rows add one-step and iterated contextual actions, normalized action transport, classifier exactness, strict-support readback, and bilateral cancellation for the same denominator carrier.

Next apply Definition 45.482, Definition 45.568, and Definition 45.570. The pair-action block supplies pair composition, unit action, componentwise product laws, certified contextual action rows, and normalized row coverage across the concrete BHist context pairs.

Finally, apply Definition 45.558. Its identity, composition, inverse, associativity, classifier-congruence, and witness-choice descent rows close the affine automorphism part of the FieldUp interface. Combining these disjoint rows gives exactly the carrier, classifier, stability, exactness, and ledger obligations listed in Definition 45.337. □

Per-object template: traditional target concrete constructive field certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist with RatHistoryCarrier and RatHistoryClassifier; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability obligations commutative-ring laws, proof-indexed inverse closure, nonzero transport, denominator multiplication, contextual action, context-pair composition, and affine automorphism closure; ledger ring expression data, nonzero witnesses, denominator histories, contextual pairs, and affine coefficient witnesses; core theorem list Definition 45.489, Definition 45.448, Definition 45.487, Definition 45.472, Definition 45.475, Definition 45.482, Definition 45.568, Definition 45.570, and Definition 45.558; exactness predicate classifier exactness for inverse, product, denominator, denominator-action component, contextual action, normalized contextual action, context-pair action, normalized context-pair row coverage, and affine automorphism rows; standard bridge ordinary constructive field over rational histories with certified affine automorphisms; status ProofTarget.

Theorem 45.490 (Rat denominator context-pair product strict support commutation)**.** *For carried denominator context pairs (l, r) and (l', r'), strict support is unchanged by swapping the two pair factors:*

$$N_D^u(\mu_D^u(\mu_D^u(l', l), \mu_D^u(r, r'))) \Longleftrightarrow N_D^u(\mu_D^u(\mu_D^u(l, l'), \mu_D^u(r', r))).$$

Proof. For the forward direction, apply Definition 45.571 to the left displayed product. This gives the finite disjunction $N_D^u(l') \vee N_D^u(l) \vee N_D^u(r) \vee N_D^u(r')$.

Apply the same strict-support theorem to the swapped product in the reverse direction. Constructive case analysis on the four left disjuncts reorders them as $N_D^u(l) \vee N_D^u(l') \vee N_D^u(r') \vee N_D^u(r)$, which is exactly the support disjunction required for the swapped product.

The reverse direction is identical with the two displayed products exchanged. The swapped product first yields $N_D^u(l) \vee N_D^u(l') \vee N_D^u(r') \vee N_D^u(r)$, and constructive case analysis reorders it back to $N_D^u(l') \vee N_D^u(l) \vee N_D^u(r) \vee N_D^u(r')$. Closing Definition 45.571 in the reverse direction gives strict support of the original product. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair product strict support commutes`

Theorem 45.491 (RatUp FieldUp source has a non-singleton endpoint)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For the concrete source tuple $\mathsf{RatUpFieldSource}_F$ of Definition 45.488, the field carrier accepts d_1, while the singleton empty-history carrier rejects it:*

$$\mathsf{RatHistoryCarrier}(d_1)$$

and

$$\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot.$$

Consequently no carrier-coverage map from the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ source carrier to the singleton empty-history carrier can exist.

Proof. Unfold the carrier coordinate of $\mathsf{RatUpFieldSource}_F$ in Definition 45.488. It is exactly $\mathsf{RatHistoryCarrier}$, so the source endpoint question is the concrete RatUp carrier question.

Apply Definition 45.109 at $d_1 = \mathsf{E1}(\mathsf{e})$. It supplies the displayed $\mathsf{RatHistoryCarrier}(d_1)$ witness and the rejection map $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$.

Now suppose a carrier-coverage map from the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ source carrier to $\mathsf{fieldSingletonEmptyCarrier}$ were given. Evaluating it at d_1 and the displayed RatUp carrier witness gives $\mathsf{fieldSingletonEmptyCarrier}(d_1)$, contradicting the rejection map. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat carrier singleton coverage obstruction`

Theorem 45.492 (RatUp FieldUp source coordinate coverage obstruction)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For the concrete source tuple $\mathsf{RatUpFieldSource}_F$ of Definition 45.488, the diagonal endpoint supplies all three source-coordinate witnesses*

$$\mathsf{RatHistoryCarrier}(d_1),$$

$$\mathsf{RatHistoryClassifier}(d_1, d_1),$$

and

$$\mathsf{RatHistoryLedgerPolicy}(d_1, d_1).$$

Each of the following coverage maps is contradictory:

1. *carrier coverage*

$$\forall h : \mathsf{BHist}, \mathsf{RatHistoryCarrier}(h) \to \mathsf{fieldSingletonEmptyCarrier}(h);$$

2. *classifier coverage*

$$\forall hk : \mathsf{BHist}, \mathsf{RatHistoryClassifier}(h, k) \to \mathsf{FieldSingletonClassifier}(h, k);$$

3. *ledger endpoint coverage*

$$\forall \rho v : \mathsf{BHist}, \mathsf{RatHistoryLedgerPolicy}(\rho, v) \to \mathsf{FieldSingletonCarrier}(\rho) \wedge \mathsf{FieldSingletonCarrier}(v).$$

Thus the concrete source tuple cannot be collapsed to the singleton $\mathsf{Field}^{\uparrow}$ boundary in any of its carrier, classifier, or ledger coordinates.

Proof. The carrier coordinate is the one isolated in Definition 45.491. It gives the endpoint witness $\mathsf{RatHistoryCarrier}(d_1)$ and rules out any carrier-coverage map from $\mathsf{RatHistoryCarrier}$ to $\mathsf{fieldSingletonEmptyCarrier}$.

The classifier and ledger coordinates are supplied by Definition 45.110. That theorem gives the diagonal classifier witness $\mathsf{RatHistoryClassifier}(d_1, d_1)$ and the diagonal ledger witness $\mathsf{RatHistoryLedgerPolicy}(d_1, d_1)$.

The same theorem also rejects the displayed classifier coverage map and the displayed ledger endpoint coverage map. Combining these two rejections with the carrier rejection from the first paragraph leaves no coverage coordinate by which $\mathsf{RatUpFieldSource}_F$ can be read as the singleton $\mathsf{Field}^{\uparrow}$ boundary. □

Definition 45.493 (Concrete RatUp FieldUp certificate instance)**.** *The concrete certificate instance for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ is the BEDC tuple with field carrier $\mathsf{RatHistoryCarrier}$, field classifier $\mathsf{RatHistoryClassifier}$, denominator carrier $\mathsf{RatDenomUnitCarrier}$, denominator classifier $\mathsf{RatDenomUnitClassifier}$, denominator multiplication append, continuation predicate Cont, empty denominator endpoint e, apart-zero predicate $\mathsf{FieldApartZero}$, the contextual denominator action Γ of Definition 45.454, and the affine automorphism package AffAut_F of Definition 45.560.*

Its per-object fields are:

1. *traditional target: the constructive field target* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*;*
2. *BEDC source:* $\mathsf{Rat}^{\uparrow}$ *together with* $\mathsf{CommRing}^{\uparrow}$ *and the concrete denominator BHist rows;*
3. *carrier:* RatHistoryCarrier*;*
4. *classifier:* RatHistoryClassifier *together with* RatDenomUnitClassifier *for denominator rows;*
5. *stability obligations: inverse apartness, two-sided zero-product exactness, denominator package closure, denominator support-neutrality, denominator contextual action, unit-pair support-neutrality, total-normalization support collapse, assembled support-neutrality Definition 45.505, context-pair action, contextual strict-support transport, context-pair strict-support commutation, and affine automorphism closure;*
6. *ledger: rational history rows, denominator unit histories, continuation witnesses, apart-zero inverse witnesses, nested contextual-action support witnesses from Definition 45.505, context-pair action witnesses, and affine coefficient witnesses;*
7. *core theorem list: Definition 45.487, Definition 45.568, Definition 45.483, Definition 45.478, Definition 45.490, Definition 45.505, Definition 45.558, and Definition 45.489;*
8. *exactness predicate: row coverage for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *including inverse, zero-product, denominator multiplication, denominator-action envelope, contextual action, context-pair action, denominator support-neutrality, unit-pair support-neutrality, total-normalization support collapse, assembled support-neutrality classifier and strict-support clauses, contextual strict-support transport, context-pair strict-support commutation, and affine automorphism rows;*
9. *standard bridge: deferred conservative bridge to the ordinary constructive rational field presentation;*
10. *status:* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.FieldUp.ConcreteRatupFieldupCertificateInstance`

Theorem 45.494 (Concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ paper certificate assembly)**.** *The concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *of Definition 45.488, with carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, denominator tuple* (RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, RatHistoryCarrier)*, contextual action* Γ*, context-pair monoid, and affine source* AffAut_F*, supplies the paper-side carrier, source, classifier, stability, ledger, and exactness rows for the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *proof target.*

Proof. The base name is fixed by Definition 37.85: the rational history carrier and classifier are the named $\mathsf{Rat}^{\uparrow}$ object on BHist. This supplies the semantic target over which the field source tuple is read.

Next apply Definition 45.489. It identifies $\mathsf{RatUpFieldSource}_F$ as satisfying Definition 45.337 with the same carrier, classifier, denominator tuple, contextual source, context-pair source, and affine source displayed in the statement.

The denominator-action coordinate is the component isolated by Definition 45.487. It records the envelope closure, normalized row coverage, and context-pair action certificate for the same concrete denominator tuple and the same action Γ.

The context-pair product support row is Definition 45.490. It is registered in Definition 45.493 as the strict-support commutation entry for the same denominator tuple.

The support-neutrality row is Definition 45.505. It supplies the carried nested contextual action, the collected-action classifier comparison, strict-support neutrality, and collected classifier exactness consumed by the concrete certificate instance.

Finally use Definition 45.558 for the affine source AffAut_F. Combining the RatUp name certificate, the assembled boundary, the denominator-action component, the strict-support commutation row, the support-neutrality row, and the affine group row fills exactly the carrier, source, classifier, stability, ledger, and exactness slots specified by Definition 45.337. This repacks the assembled source as the paper certificate row package for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$. □

Per-object certificate: traditional target constructive field over rational histories; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, denominator envelope, contextual action, and affine rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability obligations commutative-ring rows, nonzero and inverse rows, denominator rows, contextual action rows, context-pair rows, context-pair strict-support commutation, and affine closure; ledger ring expressions, nonzero witnesses, denominator histories, contextual pairs, and affine coefficient witnesses; core theorem list headed by Definition 45.494, Definition 45.489, Definition 45.487, Definition 45.505, and Definition 45.490; exactness predicate the conjunction of inverse, product, denominator, denominator-action envelope, contextual-action, context-pair, support-neutrality, strict-support commutation, and affine exactness rows; standard bridge deferred; status $\mathsf{ProofTarget}$. The phase exit becomes reportable only when the canonical machine-checked certificate site is present.

Theorem 45.495 (RatUp FieldUp contextual action support transports across context-pair classifiers)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, suppose $C^u_D(h)$ and suppose that (l, r), (s, t), (l', r'), and (s', t') are carried denominator context pairs. If $\mathsf{PairClass}^u_D((l, r), (s, t))$ and $\mathsf{PairClass}^u_D((l', r'), (s', t'))$, and if $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, then strict support is invariant under the transported two-layer contextual action:*

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Proof. Apply Definition 45.465 to the left pair of carried contexts and the empty endpoint hypotheses. Its support row reads the left nested action as the disjunction of the common core $N^u_D(h)$ with the four strict-support terms from (l, r) and (l', r').

Apply the same composition-closure theorem to (s, t) and (s', t'). This reads the right nested action as the same core term $N^u_D(h)$ together with the four strict-support terms from the transported context pairs.

Next use Definition 45.571 on both context-pair products. The four support terms on the left are exactly strict support of $\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))$, while the four support terms on the right are strict support of $\mu^u_D(\mu^u_D(s', s), \mu^u_D(t, t'))$.

The componentwise pair classifiers are the hypotheses required by Definition 45.572. They transport the product strict-support term between the two context-pair products.

For the forward direction, use the left composition readback. A core-support witness $N^u_D(h)$ is inserted directly into the right composition readback. A context-product support witness is transported across the product pair and then inserted into the corresponding four context disjuncts on the right.

The reverse direction is symmetric. Use the right composition readback, transport the product-pair support term back across the same classifier equivalence, keep the common core support fixed, and close with the left composition readback. □

↪variant BEDC.Derived.FieldUp.RatDenomUnitContextualAction pair support transport iff

Lean4 ✓ BEDC.Derived.FieldUp.field rat denominator contextual action pair transport support

Theorem 45.496 (RatUp FieldUp contextual action pair classifier transport)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$ *and suppose that* (l, r), (s, t), (l', r'), *and* (s', t') *are carried denominator context pairs. If* $\mathsf{PairClass}^u_D((l, r), (s, t))$ *and* $\mathsf{PairClass}^u_D((l', r'), (s', t'))$, *and if* $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, *and* $q'\mathsf{hsamee}$, *then the two transported two-layer contextual actions are denominator-classified:*

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Proof. Apply the two-layer contextual action composition law to the left pair and to the right pair. These give carrier closure for both nested contextual actions and classify each nested action against its collected context product $\Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)$ or $\Gamma_{p',q',\mu^u_D(s',s),\mu^u_D(t,t')}(h)$.

The component denominator classifiers classify the collected left context products $\mu^u_D(l', l)$ and $\mu^u_D(s', s)$ and the collected right context products $\mu^u_D(r, r')$ and $\mu^u_D(t, t')$. Combining these with reflexivity on the core h classifies the collected contextual actions under the empty endpoint hypotheses. Transitivity with the two composition classifiers gives the displayed denominator classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction pair classifier transport`

Theorem 45.497 (RatUp FieldUp contextual action empty pair commutation)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$ *and suppose* $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, *and* $C^u_D(r')$. *Then the two empty-external nested contextual actions obtained by applying* (l, r) *and* (l', r') *in opposite orders are denominator-classified:*

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},l',r'}(\Gamma_{\mathsf{e},\mathsf{e},l,r}(h)), \Gamma_{\mathsf{e},\mathsf{e},l,r}(\Gamma_{\mathsf{e},\mathsf{e},l',r'}(h))).$$

Proof. Normalize each nested action to the collected empty-endpoint action. The first normal form has context products $\mu^u_D(l', l)$ and $\mu^u_D(r, r')$, while the second has $\mu^u_D(l, l')$ and $\mu^u_D(r', r)$. The concrete denominator append commutativity classifiers compare these two left products and these two right products. Combining those product classifiers with reflexivity on h classifies the two collected actions, and transitivity with the two normalization classifiers gives the displayed denominator classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty pair commutes`

Theorem 45.498 (RatUp FieldUp contextual action empty pair strict support commutation)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ, *suppose* $C^u_D(h)$ *and suppose* $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, *and* $C^u_D(r')$. *Then the two empty-external nested contextual actions obtained by applying* (l, r) *and* (l', r') *in opposite orders have equivalent strict support:*

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},l',r'}(\Gamma_{\mathsf{e},\mathsf{e},l,r}(h))) \Longleftrightarrow N^u_D(\Gamma_{\mathsf{e},\mathsf{e},l,r}(\Gamma_{\mathsf{e},\mathsf{e},l',r'}(h))).$$

Proof. The empty-pair commutation theorem denominator-classifies the two nested actions. Its history-sameness field transports RatUp strict support in both directions, giving the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction empty pair strict support commutes`

Theorem 45.499 (RatUp FieldUp contextual action pair transport carrier and support)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$ *and suppose that* (l, r), (s, t), (l', r'), *and* (s', t') *are carried denominator context*

pairs. If $\mathsf{PairClass}^u_D((l,r),(s,t))$ *and* $\mathsf{PairClass}^u_D((l',r'),(s',t'))$*, and if* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then both transported nested contextual actions are carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \wedge C^u_D(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Their strict support is transported by the same context-pair classifiers:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Proof. Apply Definition 45.465 to the carried contexts (l,r) and (l',r') with the empty endpoint hypotheses. This gives the left carrier conclusion and reads the left nested action as the five-term strict-support expression with core $N^u_D(h)$.

Apply the same composition-closure theorem to (s,t) and (s',t'). This gives the right carrier conclusion and reads the right nested action as the corresponding five-term strict-support expression with the same core term. Together these two applications establish the displayed carrier conjunction.

The context tails are compressed by Definition 45.571. On the left, the four context support terms are strict support of $\mu^u_D(\mu^u_D(l',l),\mu^u_D(r,r'))$; on the right, they are strict support of $\mu^u_D(\mu^u_D(s',s),\mu^u_D(t,t'))$.

Use Definition 45.572 under the two $\mathsf{PairClass}^u_D$ hypotheses. It transports the compressed product strict-support term between the two context-pair products.

For the forward direction of the strict-support equivalence, read the left nested action through the left composition theorem. A core witness $N^u_D(h)$ is kept fixed, while a context-product witness is transported to the right product and expanded through the right product-support readback. The right composition readback then yields support of the right nested action.

The reverse direction is the same argument with left and right interchanged. The right composition readback supplies either the common core witness or a right product-support witness; the latter is transported back across the product classifier equivalence and expanded on the left before closing with the left composition readback. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator contextual action pair transport carrier support`

Theorem 45.500 (RatUp FieldUp contextual action pair transport classifier)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$ *and suppose* $R^u_D(l,s)$*,* $R^u_D(r,t)$*,* $R^u_D(l',s')$*, and* $R^u_D(r',t')$*. If* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*, then the two transported nested contextual actions are denominator-classified:*

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Proof. Apply the contextual-action composition support laws to the left nested action using the carrier fields exposed by the four classifier hypotheses. Apply the same laws to the right nested action using the transported carrier fields. The classifier equality fields identify l with s, r with t, l' with s', and r' with t', so the two concrete nested action histories are hsame. These three rows are exactly the denominator classifier. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction pair transport classifier`

Per-object template: traditional target transported contextual-action stability; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, empty endpoint, and Γ; stability obligations carrier closure plus strict-support transport; core theorem Definition 45.499; exactness predicate the displayed support iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.501 (RatUp FieldUp certificate rows are source-anchored)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *of Definition 45.488, every row consumed by Definition 45.493 is read over that same concrete* BHist *source: field carrier* $C_F = \mathsf{RatHistoryCarrier}$*, field classifier* $R_F = \mathsf{RatHistoryClassifier}$*, denominator tuple* $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier})$*, contextual action* Γ*, context-pair monoid of Definition 45.479, and affine coordinate* AffAut_F*. The* $\mathsf{CommRing}^{\uparrow}$ *base coordinate is therefore consumed only through the concrete boundary Definition 45.489.*

Proof. Unpack Definition 45.488. Its field source coordinates are the rational history carrier $\mathsf{RatHistoryCarrier}$ and rational history classifier $\mathsf{RatHistoryClassifier}$, and its denominator source is the displayed tuple with multiplication append, endpoint e, and strict support $\mathsf{RatHistoryCarrier}$.

Now apply Definition 45.489. It gives the full row list for the same source tuple $\mathsf{RatUpFieldSource}_F$, including the field carrier, field classifier, denominator tuple, contextual source, context-pair source, affine source, and the base coordinate consumed by the concrete boundary.

The denominator-action rows are the component isolated by Definition 45.487. They use the same denominator tuple and the same action Γ, so their row location agrees with the assembled boundary rather than with any separate source tuple.

For the context-pair rows, use Definition 45.568 and Definition 45.490. The first theorem supplies the context-pair action certificate, while the second supplies strict-support commutation for the context-pair product named in the statement.

The assembled support-neutrality row is Definition 45.505. It uses the same denominator tuple, the same contextual action Γ, and the same named source tuple, so the nested-action carrier, collected classifier, strict-support, and classifier-exactness clauses stay anchored to $\mathsf{RatUpFieldSource}_F$.

Finally, Definition 45.558 supplies the affine row for AffAut_F. Combining these component rows with the assembled boundary pins every entry consumed by Definition 45.493 to the single source tuple $\mathsf{RatUpFieldSource}_F$. □

Per-object template: traditional target constructive field on $\mathsf{Rat}^{\uparrow}$; BEDC source $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ and concrete denominator BHist rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ together with $\mathsf{RatDenomUnitClassifier}$ on denominator rows; stability obligations inverse apartness, zero-product exactness, denominator package closure, denominator-action envelope, support-neutrality, context-pair action, context-pair strict-support commutation, and affine automorphism packaging; ledger rational history rows, denominator unit histories, continuation witnesses, contextual pairs, and apart-zero inverse witnesses; core theorem Definition 45.505, Definition 45.487, Definition 45.483, and Definition 45.490; exactness predicate row coverage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, including the assembled support-neutrality classifier and strict-support clauses, the denominator unit-pair support-neutrality clause, the denominator-action envelope rows, and the context-pair strict-support commutation row; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.502 (RatUp FieldUp concrete schema instance)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *schema is instantiated over* BHist *by* $C_F = \mathsf{RatHistoryCarrier}$*,* $R_F = \mathsf{RatHistoryClassifier}$*, the denominator tuple* $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHisto}$ *the contextual action* Γ*, the context-pair monoid of Definition 45.479, and the affine source* AffAut_F*. Every carrier, source, classifier, stability, ledger, and exactness row consumed by Definition 45.493 is a row of this same source tuple, so no* $\mathsf{Field}^{\uparrow}$ *exit row remains as a schematic* mul*,* add*, or* neg *operator parameter.*

Proof. Unfold Definition 45.493. Its displayed fields are exactly the rational history carrier and classifier, the denominator carrier, denominator classifier, multiplication append, endpoint e, contextual action Γ, context-pair monoid rows, and affine package named in the theorem statement.

Apply Definition 45.494 to obtain the paper certificate row package for this source. It supplies the carrier, source, classifier, stability, ledger, and exactness slots for the same $\mathsf{RatUpFieldSource}_F$ tuple.

Next use Definition 45.501. This identifies every row consumed by the concrete certificate instance with the single BHist source tuple, including the denominator tuple, the contextual action, the context-pair monoid, and the affine coordinate.

Finally apply Definition 45.489. The assembled boundary contains the full carrier, classifier, denominator, contextual-action, context-pair, affine, ledger, and exactness rows, with the denominator-action and support-neutrality components supplied by the theorem list in Definition 45.493. Combining these rows shows that the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ schema is the concrete BHist instance rather than a transport through free operator parameters. □

Per-object template: traditional target constructive field on $\mathsf{Rat}^{\uparrow}$; BEDC source BHist plus $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, denominator, and affine rows; carrier RatHistoryCarrier; classifier RatHistoryClassifier plus RatDenomUnitClassifier on denominator rows; stability obligations the assembled boundary rows; ledger rational histories, denominator histories, continuation and action witnesses, and affine coefficients; core theorem Definition 45.502; exactness row coverage for the concrete source tuple; standard bridge deferred; status ProofTarget.

Theorem 45.503 (RatUp FieldUp semantic certificate assembly)**.** *For the concrete* BHist *data* RatHistoryCarrier*,* RatHistoryClassifier*,* RatDenomUnitCarrier*,* RatDenomUnitClassifier*,* append*,* Cont*,* e*, and strict support* RatHistoryCarrier*, the row-location invariant of Definition 45.501 reads the* $\mathsf{CommRing}^{\uparrow}$ *base coordinate, the FieldUp-specific inverse and apart-zero rows, the two-sided zero-product and nonzero-product rows, the denominator multiplicative rows, the denominator-action envelope rows, contextual denominator rows, the denominator support-neutrality rows, the denominator unit-pair support-neutrality row, the denominator context-pair action rows, the context-pair strict-support commutation row, and the packaged affine* $\mathsf{Group}^{\uparrow}$ *consequence over the same concrete* BHist *source. These rows supply the carrier, source, pattern, classifier, stability, ledger, and exactness rows of Definition 45.337 for the concrete target* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*. More explicitly, Definition 45.493 supplies the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate rows.*

Proof. Bind the rational carrier and classifier first. The object carrier is RatHistoryCarrier, the object classifier is RatHistoryClassifier, and the source pattern is the BEDC continuation pattern Cont over BHist. The denominator tuple is (RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, RatHistoryCarrier), with contextual action Γ and affine coordinate AffAut_F as fixed in Definition 45.493.

The source anchor is Definition 45.501. It reads the $\mathsf{CommRing}^{\uparrow}$ base coordinate through Definition 45.489 and pins the field carrier, field classifier, denominator tuple, contextual action, context-pair monoid, and affine coordinate to the same concrete source tuple.

The inverse rows are the apart-zero rows. Theorem 45.206 supplies classifier exactness for proof-indexed inverses over the concrete apart-zero predicate, so the inverse classifier, inverse stability, and nonzero transport entries are read from the displayed apartness witnesses.

The zero-product rows come from Definition 45.246. Its two-sided multiplication equivalence gives exact zero classification after certified nonzero left and right factors, and the same witness data supply the nonzero-product contradiction row in the per-object certificate record.

For the denominator part, use Definition 45.439 for the multiplicative denominator rows and Definition 45.471 for the closed denominator-side package.

The denominator-action envelope component is supplied by Definition 45.487. It records the multiplicative coverage, contextual boundary, normalized row coverage, and context-pair product and action rows over the concrete tuple fixed above.

The contextual denominator action is supplied by Definition 45.568, Definition 45.483, and Definition 45.478. They package the pair classifier, pair product, unit-pair action, product-pair composition law, collected-action exactness, unit-pair support neutrality, and strict-support readback over the same BHist denominator tuple, so the contextual exactness and support-neutrality rows align with the denominator ledger entries.

Definition 45.505 supplies the carried nested contextual action, the collected-action classifier comparison, the strict-support equivalence, and the collected classifier exactness consumed by the final FieldUp record. Thus the support-neutrality row appearing in Definition 45.493 is a concrete certificate dependency rather than a detached denominator fact.

Separately, Definition 45.490 supplies the strict-support commutation row for the context-pair product. This is the support-transport row consumed by Definition 45.493 between the context-pair action certificate and the final FieldUp record.

Finally, Definition 45.558 supplies the packaged affine $\mathsf{Group}^{\uparrow}$ consequence consumed by the field certificate.

Together the $\mathsf{CommRing}^{\uparrow}$ base rows, the inverse rows, the zero-product rows, the denominator multiplicative rows, the denominator-action envelope component, the contextual denominator rows, the support-neutrality row, the unit-pair support-neutrality row, the context-pair action rows, the strict-support commutation row, and the affine $\mathsf{Group}^{\uparrow}$ row exhaust the ten-entry per-object template of Definition 45.493: target, source, carrier, classifier, stability obligations, ledger, core theorem list, exactness predicate, standard bridge, and status. □

Theorem 45.504 (RatUp FieldUp semantic certificate is non-singleton)**.** *Let* $d_1 := \mathsf{E1}(\mathsf{e})$*. The concrete certificate rows of Definition 45.493 and Definition 45.503 are carried by the single source tuple* $\mathsf{RatUpFieldSource}_F$*. This same source has the diagonal witnesses*

$$\mathsf{RatHistoryCarrier}(d_1),$$

$$\mathsf{RatHistoryClassifier}(d_1, d_1),$$

and

$$\mathsf{RatHistoryLedgerPolicy}(d_1, d_1).$$

Each of the following singleton-boundary coverage maps is contradictory:

1. *carrier coverage*

$$\forall h : \mathsf{BHist}, \mathsf{RatHistoryCarrier}(h) \to \mathsf{fieldSingletonEmptyCarrier}(h);$$

2. *classifier coverage*

$$\forall hk : \mathsf{BHist}, \mathsf{RatHistoryClassifier}(h, k) \to \mathsf{FieldSingletonClassifier}(h, k);$$

3. *ledger endpoint coverage*

$$\forall \rho v : \mathsf{BHist}, \mathsf{RatHistoryLedgerPolicy}(\rho, v) \to \mathsf{FieldSingletonCarrier}(\rho) \wedge \mathsf{FieldSingletonCarrier}(v).$$

Consequently the semantic certificate rows for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *are sourced by the concrete* BHist *tuple and cannot collapse to the singleton* $\mathsf{Field}^{\uparrow}$ *boundary in carrier, classifier, or ledger coordinates.*

Proof. The one-source assertion is Definition 45.501. It identifies the field carrier, field classifier, denominator tuple, contextual action, context-pair monoid, and affine coordinate as rows read over the single source tuple $\mathsf{RatUpFieldSource}_F$.

Next use Definition 45.503. It says that those rows are exactly the concrete certificate rows consumed by Definition 45.493 for the target $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$.

Finally apply Definition 45.492 at $d_1 = \mathsf{E1}(\mathsf{e})$. That theorem supplies the three diagonal witnesses and separately rules out carrier, classifier, and ledger endpoint coverage maps to the singleton $\mathsf{Field}^{\uparrow}$ boundary.

Combining the source-anchored certificate rows with these three coordinate obstructions gives the claimed non-singleton semantic certificate boundary. □

Theorem 45.505 (RatUp FieldUp assembled support-neutrality certificate)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, the contextual action Γ of Definition 45.454, and the named source tuple $\mathsf{RatUpFieldSource}_F$, the assembled $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate includes the support-neutrality row below. Under the hypotheses of Definition 45.478, the nested action is carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is classified by the collected action:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)).$$

It satisfies strict-support neutrality:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(h).$$

For any second endpoint k under the same collected carried contexts, and for s, t, s', t' satisfying $s\mathsf{hsamee}$, $t\mathsf{hsamee}$, $s'\mathsf{hsamee}$, and $t'\mathsf{hsamee}$, the collected action remains classifier-exact against that endpoint:

$$R^u_D(\Gamma_{s,t,\mu^u_D(l',l),\mu^u_D(r,r')}(h), \Gamma_{s',t',\mu^u_D(l',l),\mu^u_D(r,r')}(k)) \Longleftrightarrow R^u_D(h,k).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAssembledSupportNeutralityCertificate`

Proof. The assembled boundary is fixed by Definition 45.489. It places $\mathsf{RatUpFieldSource}_F$, the displayed denominator tuple, and the action Γ inside the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate governed by Definition 45.337.

Within that boundary, use Definition 45.478. Its hypotheses are exactly the carried-denominator, empty external-context, and strict-support hypotheses named in the statement. The theorem supplies the carried nested action and its classifier comparison with the collected action.

The same row supplies the strict-support equivalence $N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(h)$. It also supplies classifier exactness for the collected action against the second endpoint k. These four conclusions are therefore available as one support-neutrality row of the assembled FieldUp certificate. □

Theorem 45.506 (RatUp FieldUp support-neutrality row completeness)**.** *For the concrete source tuple $\mathsf{RatUpFieldSource}_F$ and Definition 45.493, the support-sensitive certificate rows are exactly the concrete BHist rows supplied by the following sources:*

1. *strict product support exactness for the denominator product;*
2. *context-pair unit support neutrality;*

3. total-normalization support collapse;

4. contextual action carrier, classifier, and support neutrality;

5. context-pair strict-support commutation.

Thus the named certificate instance contains every support row consumed by the assembled semantic $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate.*

Proof. The denominator product support row is the concrete BHist row supplied by Definition 45.402. It fixes the strict product support exactness entry over the same denominator carrier, classifier, product, and empty endpoint used by $\mathsf{RatUpFieldSource}_F$.

The unit-pair and total-normalization rows are supplied by Definition 45.483 and Definition 45.478. Together they give the unit-pair support-neutrality clause, the total-normalization support collapse clause, and the contextual action carrier and classifier clauses for the concrete action Γ.

The contextual strict-support transport is the context-pair product row of Definition 45.490. It uses the same context-pair product and the same strict support predicate as the certificate instance, so it is the remaining support-sensitive transport row.

Finally Definition 45.505 repacks these concrete rows as the assembled support-neutrality certificate for $\mathsf{RatUpFieldSource}_F$. Since Definition 45.493 names exactly these stability, ledger, core theorem, and exactness entries, no additional support-sensitive row is required for the semantic assembly. □

Per-object template: traditional target constructive field on $\mathsf{Rat}^{\uparrow}$; BEDC source $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ and concrete denominator BHist rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ together with $\mathsf{RatDenomUnitClassifier}$ on denominator rows; stability obligations inverse apartness, zero-product exactness, denominator package closure, denominator-action envelope, support-neutrality, transported support-neutrality, context-pair action, context-pair strict-support commutation, and affine automorphism packaging; ledger rational history rows, denominator unit histories, continuation witnesses, contextual pairs, and apart-zero inverse witnesses; core theorem Definition 45.503, Definition 45.505, Definition 45.487, Definition 45.483, and Definition 45.490; exactness predicate row coverage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, including the assembled support-neutrality classifier and strict-support clauses, the transported support-neutrality row, the denominator unit-pair support-neutrality clause, the denominator-action envelope rows, and the context-pair strict-support commutation row; standard bridge deferred; status $\mathsf{ProofTarget}$.

45.21 RatUp FieldUp exit support rows

Theorem 45.507 (RatUp FieldUp exit support row inventory)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *and the denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate contains a closed support-row inventory consisting of the following certificate rows:*

1. support-neutrality row completeness from Definition 45.506;

2. the nontrivial carrier and support-neutral action package from Definition 45.508;

3. transported and iterated support-neutrality rows from Definition 45.509 and Definition 45.510;

4. transported strict-support separation from Definition 45.520;

5. threefold denominator support noncollapse from Definition 45.580.

Every listed row is a concrete BHist *row over* append*,* Cont*,* Γ*,* RatHistoryCarrier*, and* RatHistoryClassifier*. Thus the exit certificate has concrete support-row coverage for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *with no schematic operation parameter.*

Proof. First read Definition 45.506. It identifies the support-sensitive rows of Definition 45.493 as concrete BHist rows of the assembled semantic certificate.

Next apply Definition 45.508. This places the non-singleton source carrier, the denominator tuple, and the support-neutral contextual action inside the same concrete exit certificate.

Then compose the two transport layers. The first layer is Definition 45.509, and the second layer is Definition 45.510. They keep the carried contextual action and the strict-support predicate tied to the original denominator history through classified context-pair transport.

Now attach the strict-support separation package by applying Definition 45.520. It turns transported strict RatUp support into carrier recovery, singleton exclusion, and direct empty-history separation for the transported endpoint.

Finally apply Definition 45.580. This supplies the denominator product support obstruction against the singleton empty classifier. The cited rows use only the displayed concrete carrier, classifier, append, continuation, and contextual-action data, so the collected inventory has no remaining schematic operation parameter. □

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit support inventory; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, Cont, and Γ; carrier RatHistoryCarrier plus N^u_D; classifier RatHistoryClassifier plus singleton-boundary obstruction; stability obligations support neutrality, transported support neutrality, strict-support transport, and threefold support noncollapse; ledger denominator support witnesses and contextual-pair action witnesses; core theorem list Definition 45.506, Definition 45.508, Definition 45.509, Definition 45.510, Definition 45.520, and Definition 45.580; exactness predicate concrete support-row coverage for the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate; standard bridge deferred; status ProofTarget.

Theorem 45.508 (RatUp FieldUp exit certificate has nontrivial carrier and support-neutral action)**.** *For Definition 45.493 and* $d_1 := \mathsf{E1}(\mathsf{e})$*, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate has both a nontrivial source carrier and a support-neutral contextual action. Specifically:*

1. $\mathsf{RatHistoryCarrier}(d_1)$ *holds;*
2. $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$*;*
3. *no carrier-coverage map from the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *source carrier to* fieldSingletonEmptyCarrier *exists;*
4. *under the hypotheses of Definition 45.505, the nested contextual action is carried, classified by the collected action, support-neutral, and classifier-exact against the second endpoint.*

Equivalently, the same named certificate contains the endpoint obstruction

$$\mathsf{RatHistoryCarrier}(d_1)$$

with

$$\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$$

and the contextual-action row

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))),$$

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)),$$

$$N_D^u(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \iff N_D^u(h),$$

and, for the second endpoint k with empty external contexts,

$$R_D^u(\Gamma_{s,t,\mu_D^u(l',l),\mu_D^u(r,r')}(h), \Gamma_{s',t',\mu_D^u(l',l),\mu_D^u(r,r')}(k)) \iff R_D^u(h,k).$$

Proof. The concrete certificate is fixed by Definition 45.494. It identifies Definition 45.493 as the paper certificate row package for the displayed $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ source tuple, so the carrier, classifier, denominator tuple, and contextual action are the concrete rows named in the statement.

The carrier obstruction comes from Definition 45.491. With $d_1 := \mathsf{E1}(\mathsf{e})$ it supplies $\mathsf{RatHistoryCarrier}(d_1)$, the map $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$, and the contradiction against any carrier-coverage map into the singleton empty-history carrier.

Finally apply Definition 45.505. Under its carried-context and strict-support hypotheses, it supplies the nested carried action, the collected-action classifier comparison, strict-support neutrality, and classifier exactness against the second endpoint. These rows belong to the same certificate fixed above, so the non-singleton carrier and support-neutral action are simultaneous properties of the concrete exit certificate. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupExitCertificateNontrivialSupportNeutral`

Theorem 45.509 (RatUp FieldUp transported support-neutrality certificate)**.** *For the concrete denominator tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$ and the contextual action Γ of Definition 45.454, suppose the hypotheses of Definition 45.505 hold for the context pairs (l,r) and (l',r'). Suppose also that the transported context pairs (s,t) and (s',t') are carried and that the pair classifiers required by Definition 45.499 compare (l,r) with (s,t) and (l',r') with (s',t'). Then the transported nested action is carried:*

$$C_D^u(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))).$$

Its strict support is neutral against the original denominator history:

$$N_D^u(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))) \iff N_D^u(h).$$

Proof. Apply Definition 45.499 to the carried context pairs and the two pair classifiers. The transported carrier conclusion is the second carried component, and the same theorem gives the strict-support equivalence between the original nested action and the transported nested action.

Apply Definition 45.505 to the original context pairs (l,r) and (l',r'). This supplies $N_D^u(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftarrow N_D^u(h)$.

For the forward direction, read support of the transported nested action back through the transport equivalence, then use the assembled support-neutrality row to obtain $N_D^u(h)$.

For the reverse direction, use assembled support-neutrality to obtain support of the original nested action, then move it across the transport equivalence to the transported context-pair representative. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupTransportedSupportNeutralityRow`

Theorem 45.510 (RatUp FieldUp iterated transported support-neutrality row)**.** *For the concrete denominator tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$ and the contextual action Γ of Definition 45.454, suppose the hypotheses of Definition 45.509 transport $(l,r),(l',r')$ to $(s,t),(s',t')$. Suppose also that (u,v) and (u',v') are carried denominator context pairs and that the pair classifiers required by Definition 45.499 compare (s,t) with (u,v) and (s',t') with (u',v'). Then the twice transported nested action is carried:*

$$C_D^u(\Gamma_{p',q',u',v'}(\Gamma_{p,q,u,v}(h))).$$

Its strict support is still neutral against the original denominator history:

$$N_D^u(\Gamma_{p',q',u',v'}(\Gamma_{p,q,u,v}(h))) \Longleftrightarrow N_D^u(h).$$

Proof. First apply Definition 45.509 to the comparison from $(l, r), (l', r')$ to $(s, t), (s', t')$. This supplies the carried nested action at the $(s, t), (s', t')$ representative and the strict-support equivalence

$$N_D^u(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))) \Longleftrightarrow N_D^u(h).$$

Next apply Definition 45.499 to the carried context pairs (s, t), (u, v), (s', t'), and (u', v'). The carrier component gives $C_D^u(\Gamma_{p',q',u',v'}(\Gamma_{p,q,u,v}(h)))$, and the support component compares the $(s, t), (s', t')$ representative with the $(u, v), (u', v')$ representative.

For the forward direction of the displayed equivalence, read support of the $(u, v), (u', v')$ representative back through the second transport equivalence and then through the first transported support-neutrality row to obtain $N_D^u(h)$.

For the reverse direction, use the first transported support-neutrality row to obtain support of the $(s, t), (s', t')$ representative, then move it across the second transport equivalence to the twice transported representative. □

Per-object template: traditional target constructive field on $\mathsf{Rat}^{\uparrow}$; BEDC source $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ and concrete denominator BHist rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ together with $\mathsf{RatDenomUnitClassifier}$ on denominator rows; stability obligations inverse apartness, zero-product exactness, denominator package closure, denominator-action envelope, support-neutrality, transported support-neutrality, context-pair action, context-pair strict-support commutation, and affine automorphism packaging; ledger rational history rows, denominator unit histories, continuation witnesses, contextual pairs, and apart-zero inverse witnesses; core theorem Definition 45.503, Definition 45.505, Definition 45.509, Definition 45.510, Definition 45.487, Definition 45.483, and Definition 45.490; exactness predicate row coverage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, including the assembled support-neutrality classifier and strict-support clauses, the transported support-neutrality row, the denominator unit-pair support-neutrality clause, the denominator-action envelope rows, and the context-pair strict-support commutation row; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.511 (RatUp FieldUp transported nondegenerate exit certificate)**.** *Let* $d_1 := \mathsf{E1}(\mathsf{e})$*. For Definition 45.493, under the hypotheses of Definition 45.509, the same concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate contains the source nondegeneracy and the transported support-neutrality conclusions:*

1. $\mathsf{RatHistoryCarrier}(d_1)$*;*
2. $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$*;*
3. *the transported nested action carrier conclusion*

$$C_D^u(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h)))$$

4. *the transported strict-support neutrality conclusion*

$$N_D^u(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))) \Longleftrightarrow N_D^u(h).$$

Proof. First fix the certificate site. Theorem 45.503 reads Definition 45.493 as the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate, and Definition 45.501 pins its rows to the same BHist source tuple.

The endpoint part is supplied by Definition 45.491. At $d_1 = \mathsf{E1}(\mathsf{e})$ it gives both $\mathsf{RatHistoryCarrier}(d_1)$ and $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$ over that source.

It remains to read the transported action row. Under the stated hypotheses, Definition 45.509 supplies the carried transported nested action and the displayed strict-support equivalence. Because the certificate assembly and source-anchoring theorems identify these rows as entries of the same concrete certificate instance, the four displayed conclusions coexist in the single exit certificate. □

Per-object template: traditional target constructive field on $\mathsf{Rat}^{\uparrow}$; BEDC source $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{CommRing}^{\uparrow}$ and concrete denominator BHist rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ together with $\mathsf{RatDenomUnitClassifier}$ on denominator rows; stability obligations inverse apartness, zero-product exactness, denominator package closure, denominator-action envelope, support-neutrality, transported support-neutrality, context-pair action, context-pair strict-support commutation, and affine automorphism packaging; ledger rational history rows, denominator unit histories, continuation witnesses, contextual pairs, and apart-zero inverse witnesses; core theorem Definition 45.503, Definition 45.505, Definition 45.509, Definition 45.487, Definition 45.483, and Definition 45.490; exactness predicate row coverage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, including the assembled support-neutrality classifier and strict-support clauses, the transported support-neutrality row, the denominator unit-pair support-neutrality clause, the denominator-action envelope rows, and the context-pair strict-support commutation row; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.512 (RatUp FieldUp nondegenerate semantic exit source)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For the concrete source tuple $\mathsf{RatUpFieldSource}_F$ of Definition 45.488, the same BHist data simultaneously give the following four pieces of the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit source:*

1. *the assembled semantic naming certificate rows for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$;*
2. *source anchoring of all certificate rows at $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$;*
3. *the assembled support-neutrality row for the denominator contextual action Γ;*
4. *the endpoint separation*
$$\mathsf{RatHistoryCarrier}(d_1)$$
together with
$$\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot.$$

Consequently the semantic exit source for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ over $\mathsf{RatUpFieldSource}_F$ is not collapsible to the singleton empty-history $\mathsf{Field}^{\uparrow}$ source.

Proof. The semantic certificate rows are those assembled in Definition 45.503. They include the carrier, source, classifier, stability, ledger, exactness, and affine rows of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate.

The common source location is fixed by Definition 45.501. It identifies the field carrier as $\mathsf{RatHistoryCarrier}$, the field classifier as $\mathsf{RatHistoryClassifier}$, and the denominator, contextual-action, context-pair, and affine rows as entries of the same $\mathsf{RatUpFieldSource}_F$ tuple.

The denominator contextual action contributes the support-neutrality component through Definition 45.505. Thus the assembled certificate contains the carried nested action, the collected-action classifier comparison, strict-support neutrality, and the second-endpoint classifier exactness row over the denominator tuple.

Finally, Definition 45.491 supplies the displayed endpoint $d_1 = \mathsf{E1}(\mathsf{e})$, its $\mathsf{RatHistoryCarrier}(d_1)$ witness, and the rejection map $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$. Any collapse to the singleton empty-history $\mathsf{Field}^{\uparrow}$ source would have to send this carried endpoint to the singleton carrier, contradicting that rejection map. □

Theorem 45.513 (RatUp FieldUp concrete exit row closure)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For the concrete source tuple $\mathsf{RatUpFieldSource}_F$ of Definition 45.488, the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit row package is the concrete* BHist*-anchored closure jointly supplied by the following data:*

1. *the semantic certificate rows of Definition 45.503;*
2. *the source anchoring rows of Definition 45.501;*
3. *the support-sensitive row coverage of Definition 45.506;*
4. *the endpoint facts $\mathsf{RatHistoryCarrier}(d_1)$ and $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$;*
5. *the shared concrete row language over* RatHistoryCarrier*,* RatHistoryClassifier*,* RatDenomUnitCarrier*,* RatDenomUnitClassifier*,* append*,* Cont*,* e*, Γ, and AffAut_F.*

Thus the displayed exit package is a concrete closure over $\mathsf{RatUpFieldSource}_F$ and not an instance of an abstract operator schema.

Proof. Apply Definition 45.503 first. It gives the assembled carrier, source, pattern, classifier, stability, ledger, and exactness rows for the concrete target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.

Apply Definition 45.501 next. It pins those rows to the single source tuple $\mathsf{RatUpFieldSource}_F$, with field carrier RatHistoryCarrier, field classifier RatHistoryClassifier, denominator tuple (RatDenomUnitCarrier, Rat contextual action Γ, and affine coordinate AffAut_F.

The support-sensitive part is closed by Definition 45.506. It records that the support rows consumed by the assembled semantic certificate are precisely the concrete BHist rows already present in the named certificate instance.

Finally, Definition 45.512 supplies the endpoint $d_1 = \mathsf{E1}(\mathsf{e})$, the witness $\mathsf{RatHistoryCarrier}(d_1)$, and the rejection map $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$. Combining these four cited sites repacks the existing rows as one concrete closure theorem for the exit package. The result is therefore a composite closure statement, not a variant of any single checked theorem. □

Per-object template: traditional target constructive field over rational histories; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, denominator tuple, contextual action, context-pair action, and affine rows; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability rows Definition 45.503, Definition 45.501, and Definition 45.505; ledger rational histories, denominator histories, contextual pairs, support witnesses, and affine coefficients; core theorem Definition 45.512 together with Definition 45.503, Definition 45.501, Definition 45.505, and Definition 45.491; exactness the assembled row coverage plus non-singleton source separation; standard bridge deferred; status ProofTarget.

Theorem 45.514 (RatUp FieldUp empty-context nondegenerate witness)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For histories $L, R : \mathsf{BHist}$, assume $\mathsf{fieldSingletonEmptyCarrier}(L)$ and $\mathsf{fieldSingletonEmptyCarrier}(R)$. Put $w := \mathsf{append}(L, \mathsf{append}(d_1, R))$. Then the wrapped endpoint satisfies*

$$\mathsf{RatHistoryCarrier}(w),$$

the rejection map

$$\mathsf{fieldSingletonEmptyCarrier}(w) \to \bot,$$

and the nonzero-complement clause

$$\mathsf{fieldSingletonEmptyNonZero}(w).$$

Thus the non-singleton rational witness remains stable under empty singleton context wrapping on both sides.

Proof. The hypotheses place both contextual wrappers in the singleton empty carrier. By the singleton-empty carrier inversion for $\mathsf{Field}^{\uparrow}$, each wrapper compares by hsame with e.

Use the left and right append-unit laws, together with the continuation compatibility of append and transitivity of hsame, to compare

$$w = \mathsf{append}(L, \mathsf{append}(d_1, R))$$

with the endpoint d_1. This comparison is the transport path for all carrier and classifier predicates used below.

Apply Definition 45.512. It supplies $\mathsf{RatHistoryCarrier}(d_1)$ and the rejection map $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$. Transport the rational carrier witness from d_1 to w along the displayed hsame comparison.

For rejection at w, take any hypothetical $\mathsf{fieldSingletonEmptyCarrier}(w)$ witness. Transport it back along the same comparison to obtain a singleton-empty carrier witness for d_1, and contradict the rejection map from Definition 45.512.

It remains to obtain the nonzero clause. Unfold $\mathsf{fieldSingletonEmptyNonZero}(w)$ as the function type

$$\mathsf{fieldSingletonEmptyClassifier}(w, \mathsf{e}) \to \bot.$$

Given a classifier witness for (w, e), take its source-carrier component $\mathsf{fieldSingletonEmptyCarrier}(w)$. Applying the rejection map for w constructed in the preceding paragraph gives the required contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup empty context nondegenerate witness`

Per-object template: traditional target context-stable nondegenerate rational field witness; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, Cont, and singleton $\mathsf{Field}^{\uparrow}$ carrier; carrier choice $\mathsf{RatHistoryCarrier}$; classifier choice singleton empty classifier for the obstruction; stability obligations append-unit transport and carrier transport; ledger empty context wrappers; core theorem Definition 45.514 together with Definition 45.512; exactness predicate preservation of the non-singleton witness under empty wrappers; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.515 (RatUp FieldUp transported strict-support non-singleton endpoint)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the hypotheses of Definition 45.509, suppose also that $N^u_D(h)$ holds. Then the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate contains all of the following endpoint data:*

1. *$C^u_D(x)$;*
2. *$N^u_D(x)$;*
3. *$\mathsf{RatHistoryCarrier}(x)$;*
4. *$\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$.*

Thus the transported strict-support endpoint of the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate cannot collapse into the singleton empty-history field carrier.

Proof. Apply Definition 45.511 at the same transported endpoint. It gives the carried endpoint clause $C^u_D(x)$ and the strict-support equivalence

$$N^u_D(x) \Longleftrightarrow N^u_D(h).$$

Use the reverse direction of this equivalence with the displayed witness $N^u_D(h)$. This produces the transported strict-support witness $N^u_D(x)$.

Now read the denominator tuple row from Definition 45.438. There $N^u_D(y) = \mathsf{RatHistoryCarrier}(y)$ for every denominator history y, so the transported support witness gives $\mathsf{RatHistoryCarrier}(x)$.

Finally apply Definition 45.98 to x. Any witness of $\mathsf{fieldSingletonEmptyCarrier}(x)$, together with the Rat history carrier witness just obtained, yields contradiction. This is the displayed map $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported strict support non singleton`

Theorem 45.516 (RatUp FieldUp transported strict-support contextual singleton exclusion)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the hypotheses of Definition 45.509, suppose also that $N^u_D(h)$ holds. Then for every context history $r : \mathsf{BHist}$ both boundary collapses are impossible:*

1. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$;
2. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$.

Proof. Apply Definition 45.515 at the same transported endpoint. The endpoint data supplied there includes $\mathsf{RatHistoryCarrier}(x)$.

For the right-factor append boundary, apply Definition 45.101 with the context history r and the RatUp factor x. The displayed Rat carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x))$ into contradiction.

For the left-factor append boundary, apply Definition 45.102 with the same RatUp factor x and context history r. The same carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r))$ into contradiction. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction strict support contextual singleton exclusion`
Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported strict support contextual singleton exclusion`

Theorem 45.517 (Rat denominator unit contextual action strict-support singleton classifier exclusion)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the same empty endpoint and carried-context hypotheses as Definition 45.516, suppose also that $N^u_D(h)$ holds. Then for all histories $L, R : \mathsf{BHist}$ the singleton empty classifier cannot classify the two contextual endpoints with the same strict-support tail:*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot.$$

Proof. The contextual action support law reads the strict-support predicate of x as the disjunction of the outer left support, inner left support, core support, inner right support, and outer right support. Insert the displayed witness $N^u_D(h)$ into the core disjunct to obtain $\mathsf{RatHistoryCarrier}(x)$.

Package this witness with itself and reflexivity of hsame to obtain the concrete $\mathsf{RatHistoryClassifier}(x, x)$. The Rat/singleton boundary separation theorem Definition 45.104 then rules out any singleton empty classifier on $\mathsf{append}(L, x)$ and $\mathsf{append}(R, x)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction strict support singleton classifier exclusion`

Theorem 45.518 (RatUp FieldUp transported strict-support singleton classifier exclusion)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Under the hypotheses of Definition 45.509, suppose also that the transported source endpoint carries strict RatUp support. Then for all histories* $L, R : \mathsf{BHist}$*,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot.$$

Proof. Transport the strict RatUp support across the classified contextual pair, giving $\mathsf{RatHistoryCarrier}(x)$. Together with itself and reflexivity of hsame, this forms the concrete classifier $\mathsf{RatHistoryClassifier}(x, x)$. The Rat/singleton boundary separation theorem Definition 45.104 then rejects any singleton empty classifier on the two appended endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported strict support singleton classifier exclusion`

Theorem 45.519 (RatUp FieldUp iterated transported strict-support contextual singleton exclusion)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $y := \Gamma_{p',q',u',v'}(\Gamma_{p,q,u,v}(h))$*. Under the hypotheses of Definition 45.510, suppose also that* $N^u_D(h)$ *holds. Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate contains the following endpoint and contextual exclusion package:*

1. $C^u_D(y)$*;*
2. $N^u_D(y)$*;*
3. $\mathsf{RatHistoryCarrier}(y)$*;*
4. $\mathsf{fieldSingletonEmptyCarrier}(y) \to \bot$*;*
5. *for every* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y)) \to \bot$*;*
6. *for every* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r)) \to \bot$*.*

Proof. Apply Definition 45.510 to the same carried context-pair comparisons. This supplies $C^u_D(y)$ and the strict-support equivalence

$$N^u_D(y) \Longleftrightarrow N^u_D(h).$$

Use the reverse direction of the equivalence with the assumed witness $N^u_D(h)$. This gives $N^u_D(y)$.

Read the concrete strict support row from Definition 45.438. There $N^u_D(z) = \mathsf{RatHistoryCarrier}(z)$ for every denominator history z, hence the support witness at y gives $\mathsf{RatHistoryCarrier}(y)$.

For the endpoint rejection, apply Definition 45.98 to y. A singleton-empty carrier witness at y contradicts the Rat carrier witness just obtained.

It remains to reject both contextual singleton boundaries. For any $r : \mathsf{BHist}$, apply Definition 45.101 to the context history r and Rat carrier endpoint y, and apply Definition 45.102 to the same two histories in the opposite append order. □

Theorem 45.520 (RatUp FieldUp transported strict-support separation package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $x_L := \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$ *and let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Suppose* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*; suppose* $C^u_D(h)$*; suppose the denominator context pairs are classified by*

$R^u_D(l,s)$, $R^u_D(r,t)$, $R^u_D(l',s')$, and $R^u_D(r',t')$; and suppose the left nested action has strict support, $\mathsf{RatHistoryCarrier}(x_L)$*. Then the transported right nested action has the following separation package:*

1. $C^u_D(x)$;
2. $\mathsf{RatHistoryCarrier}(x)$;
3. $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$;
4. $x\mathsf{hsamee} \to \bot$.

Proof. First apply Definition 45.499 to the empty endpoints, carried source h, and the four denominator classifier rows. Transport the assumed strict-support witness $\mathsf{RatHistoryCarrier}(x_L)$ across this contextual-pair comparison. This gives $\mathsf{RatHistoryCarrier}(x)$, and the carriedness clause in the same transport gives $C^u_D(x)$.

Next read the concrete strict-support separation row for Rat histories. Since x has Rat history carrier, the singleton-empty carrier row at x is disjoint from it, giving $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$.

It remains to separate the endpoint from the kernel empty history itself. Apply Definition 37.18 to the recovered $\mathsf{RatHistoryCarrier}(x)$ witness. The resulting map sends any equality $x\mathsf{hsamee}$ to contradiction, giving $x\mathsf{hsamee} \to \bot$.

Collect these four clauses as the displayed transported strict-support separation package. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported strict support separation package`

Theorem 45.521 (RatUp FieldUp transported strict-support contextual nonzero package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the hypotheses of Definition 45.509, suppose also that $N^u_D(h)$ holds. Then for every context history $r : \mathsf{BHist}$ both proof-indexed inverse-domain witnesses hold:*

1. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$;
2. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$.

Proof. First apply Definition 45.516 at the same denominator tuple, contextual action, transported endpoint, and strict-support witness. This gives the rejection maps

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r)) \to \bot.$$

For the right-append inverse-domain witness, unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$ as $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(r,x),\mathsf{e}) \to \bot$. Any forbidden classifier witness contains the source carrier field $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x))$, contradicting the first rejection map.

For the left-append inverse-domain witness, unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$ as $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(x,r),\mathsf{e}) \to \bot$. Any forbidden classifier witness contains the source carrier field $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r))$, contradicting the second rejection map. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported strict support contextual nonzero package`

Theorem 45.522 (RatUp FieldUp transported strict-support inverse-domain classifier package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Under the hypotheses of Definition 45.509, suppose also that* $N^u_D(h)$ *holds. Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate jointly contains the transported strict-support inverse-domain and classifier obstruction package:*

1. $x\mathsf{hsamee} \to \bot$*;*
2. *for every* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x))$*;*
3. *for every* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x, r))$*;*
4. *for all* $L, R : \mathsf{BHist}$*,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot.$$

Proof. Apply Definition 45.520 at the same denominator tuple, contextual action, transported endpoint, neutrality hypotheses, and strict-support witness. Its final clause gives the direct empty-history separation $x\mathsf{hsamee} \to \bot$.

Next apply Definition 45.521 with the same data. For each context history $r : \mathsf{BHist}$, this supplies both inverse-domain witnesses $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x))$ and $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{appe}$

Finally apply Definition 45.517 with the same transported endpoint and strict-support source witness. For all $L, R : \mathsf{BHist}$, it gives the classifier obstruction on the two append contexts carrying the common strict-support tail x. □

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ strict-support inverse-domain separation; BEDC source BHist, $\mathsf{RatUpFieldSource}_F$, append, contextual action Γ, and strict support N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier obstruction $\mathsf{fieldSingletonEmptyClassifier}$; stability obligations transported support neutrality, contextual nonzero, and singleton-classifier exclusion; ledger transported endpoint plus append contexts; core theorem Definition 45.522; exactness simultaneous nonzero complement and singleton-classifier rejection; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.523 (RatUp FieldUp concrete exit certificate package)**.** *For Definition 45.493 and the source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate simultaneously contains the following certificate data:*

1. *the obligation rows of Definition 45.337;*
2. *the denominator multiplicative coverage of Definition 45.439;*
3. *the concrete exit-row closure of Definition 45.513;*
4. *the empty-context nondegenerate witness of Definition 45.514;*
5. *the transported strict-support non-singleton endpoint of Definition 45.515.*

Hence the exit package is a concrete BHist*-anchored* $\mathsf{ProofTarget}$ *over* $\mathsf{RatUpFieldSource}_F$*, not an abstract operator schema.*

Proof. Begin with Definition 45.337. It fixes the rows that a $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate must carry: source, carrier, classifier, ring base, apartness, inverse, product, denominator, contextual denominator, affine, ledger, exactness, and status rows.

Next apply Definition 45.439. This supplies the denominator multiplicative row coverage inside the same field certificate language, so the denominator component is read as a certificate row rather than as an external monoid fact.

Apply Definition 45.513. It repacks the semantic certificate rows, source anchoring rows, support-sensitive coverage, and non-singleton endpoint data as one concrete exit-row closure over $\mathsf{RatUpFieldSource}_F$.

The nondegenerate empty-context component is supplied by Definition 45.514. It gives the wrapped rational endpoint, its rational-history carrier witness, its singleton empty rejection map, and the nonzero-complement clause under empty singleton wrappers.

Finally apply Definition 45.515. This gives the transported endpoint clauses $C^u_D(x)$, $N^u_D(x)$, $\mathsf{RatHistoryCarrier}(x)$, and $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$.

Combining these five cited sites yields a single concrete BHist-anchored certificate package with status $\mathsf{ProofTarget}$. Its data are the rows of the displayed $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ instance, so the conclusion is a concrete exit certificate for $\mathsf{RatUpFieldSource}_F$ and not an abstract operator schema. □

Theorem 45.524 (RatUp FieldUp transported strict-support contextual separation package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the hypotheses of Definition 45.509, suppose also that $N^u_D(h)$ holds. Then the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate contains all seven clauses:*

1. *$C^u_D(x)$;*
2. *$N^u_D(x)$;*
3. *$\mathsf{RatHistoryCarrier}(x)$;*
4. *$\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$;*
5. *$x\mathsf{hsamee} \to \bot$;*
6. *for every $r : \mathsf{BHist}$, $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$;*
7. *for every $r : \mathsf{BHist}$, $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$.*

Proof. First apply Definition 45.520 with the same denominator tuple, contextual action, endpoint, and hypotheses. This supplies the carried endpoint, transported strict support, Rat carrier witness, singleton endpoint exclusion, and direct empty-history separation.

It remains to attach the two contextual singleton exclusions. Apply Definition 45.516 with the same transported endpoint and support witness. For each $r : \mathsf{BHist}$, this gives the left-context and right-context rejection maps for $\mathsf{append}(r, x)$ and $\mathsf{append}(x, r)$.

Collect the five endpoint clauses from the separation package and the two contextual clauses from singleton exclusion as the displayed seven-clause certificate row. □

Per-object template: traditional target constructive $\mathsf{Field}^{\uparrow}$ exit separation; BEDC source $\mathsf{RatUpFieldSource}_F$, contextual action Γ, and strict support N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability obligations transported support neutrality plus Rat/singleton separation; ledger contextual-pair action and denominator support; core theorems Definition 45.515, Definition 45.516, Definition 45.519, and Definition 45.521 and Definition 45.520; exactness predicate transported and iterated transported strict-support non-collapse plus contextual singleton exclusion, and direct internal-nonempty endpoint separation; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.525 (RatUp FieldUp transported exit row nondegenerate closure)**.** *For the concrete source tuple $\mathsf{RatUpFieldSource}_F$, the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, and the contextual action Γ of Definition 45.454, let $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$. Under the hypotheses of Definition 45.509, suppose also that $N^u_D(h)$ holds. Then the single concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit row package simultaneously contains:*

1. *the semantic certificate assembly and source anchoring rows recorded by Definition 45.513;*
2. *the transported carrier clause $C^u_D(x)$ and the strict-support equivalence*

$$N^u_D(x) \Longleftrightarrow N^u_D(h);$$

3. *the endpoint facts $C^u_D(x)$, $N^u_D(x)$, $\mathsf{RatHistoryCarrier}(x)$, and $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$;*
4. *the separation clause that no carrier-coverage map*

$$\kappa : \mathsf{RatHistoryCarrier}(x) \to \mathsf{fieldSingletonEmptyCarrier}(x)$$

exists for the transported strict-support endpoint of this exit row.

Thus the transported endpoint, its concrete exit-row closure, and its source-anchored certificate rows are one nondegenerate $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit package.

Proof. First fix the row package. Apply Definition 45.513. It identifies the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit rows over $\mathsf{RatUpFieldSource}_F$ and records the semantic certificate assembly together with the source anchoring rows.

Next read the transported endpoint. Apply Definition 45.511 under the same hypotheses. This supplies the transported carrier clause $C^u_D(x)$ and the strict-support equivalence

$$N^u_D(x) \Longleftrightarrow N^u_D(h).$$

Using the assumed witness $N^u_D(h)$ gives $N^u_D(x)$.

Now use the preceding endpoint theorem Definition 45.515. It gives the concrete Rat carrier witness $\mathsf{RatHistoryCarrier}(x)$ and the singleton rejection map $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$ for the same endpoint.

It remains to rule out coverage by the singleton empty-history source. Suppose that a carrier-coverage map

$$\kappa : \mathsf{RatHistoryCarrier}(x) \to \mathsf{fieldSingletonEmptyCarrier}(x)$$

exists at this transported strict-support endpoint. Applying κ to the witness $\mathsf{RatHistoryCarrier}(x)$ gives $\mathsf{fieldSingletonEmptyCarrier}(x)$, contradicting the displayed rejection map. Hence no such carrier-coverage map exists. □

Definition 45.526 (RatUp FieldUp threefold denominator support carrier)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$, the threefold denominator support carrier is the predicate*

$$\mathsf{ThreeSupp}^u_D(h,k,\ell) := C^u_D(h) \wedge C^u_D(k) \wedge C^u_D(\ell) \wedge ((N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell)).$$

Theorem 45.527 (RatUp FieldUp threefold support non-singleton exactness)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h,k),\ell)$. Then the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit boundary contains the following threefold endpoint data:*

1. *$C^u_D(x)$;*
2. $$N^u_D(x) \Longleftrightarrow ((N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell));$$
3. *$\mathsf{ThreeSupp}^u_D(h,k,\ell) \to \mathsf{RatHistoryCarrier}(x)$;*

4. $\mathsf{ThreeSupp}^u_D(h,k,\ell) \to (\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot)$.

Proof. Apply the closure clause of Definition 45.347 to the carried inputs h and k. This gives $C^u_D(\mu^u_D(h,k))$. Apply the same closure clause to $\mu^u_D(h,k)$ and ℓ to obtain $C^u_D(x)$.

Next apply Definition 45.408 to h,k,ℓ with their carried hypotheses. The result is the displayed strict-support equivalence for the left-nested endpoint x.

Assume $\mathsf{ThreeSupp}^u_D(h,k,\ell)$. Its last conjunct gives $((N^u_D(h)\vee N^u_D(k))\vee N^u_D(\ell))$. Use the reverse direction of the displayed equivalence to obtain $N^u_D(x)$.

Read the denominator tuple row from Definition 45.438. There $N^u_D(y) = \mathsf{RatHistoryCarrier}(y)$ for every denominator history y, so the support witness just obtained gives $\mathsf{RatHistoryCarrier}(x)$.

Finally apply Definition 45.98 to x. A witness of $\mathsf{fieldSingletonEmptyCarrier}(x)$ together with the Rat history carrier witness yields contradiction, giving the displayed non-singleton endpoint map. □

Theorem 45.528 (RatUp FieldUp right-nested threefold support non-singleton exactness)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h,k,\ell : \mathsf{BHist}$ *with* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *let* $x := \mu^u_D(\mu^u_D(h,k),\ell)$ *and* $y := \mu^u_D(h,\mu^u_D(k,\ell))$. *Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit boundary contains the following right-nested threefold endpoint data:*

1. $C^u_D(y)$;

2. $R^u_D(x,y)$;

3.
$$N^u_D(y) \iff ((N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell));$$

4. $\mathsf{ThreeSupp}^u_D(h,k,\ell) \to \mathsf{RatHistoryCarrier}(y)$;

5. $\mathsf{ThreeSupp}^u_D(h,k,\ell) \to (\mathsf{fieldSingletonEmptyCarrier}(y) \to \bot)$.

Proof. First build the right-nested carrier. Apply the closure clause of Definition 45.347 to the carried inputs k and ℓ, obtaining $C^u_D(\mu^u_D(k,\ell))$. Apply the same closure clause to h and $\mu^u_D(k,\ell)$ to obtain $C^u_D(y)$.

The classifier comparison is the associativity clause of Definition 45.404 at the carried histories h,k,ℓ. With the displayed endpoints, this gives $R^u_D(x,y)$.

Next read strict support on the right-nested product. Apply Definition 45.402 to k and ℓ, giving
$$N^u_D(\mu^u_D(k,\ell)) \iff N^u_D(k) \vee N^u_D(\ell).$$

Apply the same product exactness theorem to h and $\mu^u_D(k,\ell)$. The outer endpoint is y, so it gives
$$N^u_D(y) \iff N^u_D(h) \vee N^u_D(\mu^u_D(k,\ell)).$$

For the forward direction of the displayed theorem, use the outer equivalence. If it returns $N^u_D(h)$, place that witness in the left branch of $((N^u_D(h)\vee N^u_D(k))\vee N^u_D(\ell))$. If it returns $N^u_D(\mu^u_D(k,\ell))$, use the inner equivalence; its $N^u_D(k)$ case enters the left branch and its $N^u_D(\ell)$ case enters the right branch.

For the reverse direction, split $((N^u_D(h)\vee N^u_D(k))\vee N^u_D(\ell))$ by cases. The $N^u_D(h)$ case enters the left branch of the outer equivalence. The $N^u_D(k)$ and $N^u_D(\ell)$ cases first enter the reverse direction of the inner equivalence, producing $N^u_D(\mu^u_D(k,\ell))$, and then enter the right branch of the outer equivalence.

Now assume $\mathsf{ThreeSupp}^u_D(h,k,\ell)$. Its support component is exactly the right-hand side of the strict-support equivalence, so the reverse direction gives $N^u_D(y)$.

Read the denominator tuple row from Definition 45.438. There $N^u_D(z) = \mathsf{RatHistoryCarrier}(z)$ for every denominator history z, hence the support witness gives $\mathsf{RatHistoryCarrier}(y)$.

Finally apply Definition 45.98 to y. A witness of $\mathsf{fieldSingletonEmptyCarrier}(y)$ together with the Rat history carrier witness yields contradiction, giving the right-nested non-singleton endpoint map. □

Theorem 45.529 (RatUp FieldUp threefold support contextual separation package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h, k), \ell)$. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit boundary contains the following contextual separation package:*

1. *the direct endpoint separation $x\mathsf{hsamee} \to \bot$;*
2. *for every $r : \mathsf{BHist}$, both contextual carrier exclusions $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$ and $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$;*
3. *for all $L, R : \mathsf{BHist}$, the contextual classifier exclusion*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot.$$

Proof. First apply Definition 45.527 with the same denominator tuple and carried histories. The threefold support hypothesis recovers $\mathsf{RatHistoryCarrier}(x)$ and the singleton endpoint rejection $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$.

For direct empty separation, apply Definition 37.18 to the recovered Rat carrier witness. Any comparison $x\mathsf{hsamee}$ is thereby sent to contradiction.

Now fix an arbitrary context history $r : \mathsf{BHist}$. Apply Definition 45.101 with context r and Rat carrier endpoint x to rule out $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x))$. Apply Definition 45.102 with the same two histories in the opposite append order to rule out $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r))$.

Finally package $\mathsf{RatHistoryClassifier}(x, x)$ from the carrier witness on both sides and reflexivity of hsame. For arbitrary $L, R : \mathsf{BHist}$, Definition 45.104 then rejects $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}($ □

Theorem 45.530 (RatUp FieldUp threefold support contextual separation)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h, k), \ell)$. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then for every $r : \mathsf{BHist}$ both contextual singleton collapses are impossible:*

1. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$;
2. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$.

Proof. Apply Definition 45.527 to the same denominator tuple, carried inputs, left-nested endpoint, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. The endpoint data includes $\mathsf{RatHistoryCarrier}(x)$.

For the right-factor append boundary, apply Definition 45.101 with context history r and RatUp factor x. The recovered Rat carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x))$ into contradiction.

For the left-factor append boundary, apply Definition 45.102 with the same RatUp factor x and context history r. The same carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r))$ into contradiction. □

Theorem 45.531 (RatUp FieldUp threefold support contextual inverse-domain package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$,*

and $C^u_D(\ell)$*, let* $x := \mu^u_D(\mu^u_D(h,k),\ell)$*. If* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then every context history* $r : \mathsf{BHist}$ *satisfies the two inverse-domain witnesses:*

1. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$*;*
2. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$*.*

Proof. First apply Definition 45.530 at the same denominator tuple, carried inputs, endpoint, support carrier witness, and context history r. This gives the rejection maps $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x)) \to \bot$ and $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r)) \to \bot$. For the right-append inverse-domain witness, unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$ as $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(r,x),\mathsf{e}) \to \bot$. Any forbidden classifier witness contains the source carrier field $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x))$, contradicting the first rejection map. For the left-append inverse-domain witness, unfold fieldSingletonEmptyNonZer as $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(x,r),\mathsf{e}) \to \bot$. Any forbidden classifier witness contains the source carrier field $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r))$, contradicting the second rejection map. □

Theorem 45.532 (RatUp FieldUp threefold support contextual nonzero boundary)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h,k,\ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $x := \mu^u_D(\mu^u_D(h,k),\ell)$*. If* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then for every* $r : \mathsf{BHist}$ *the concrete append-context boundary contains:*

1. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x)) \to \bot$*;*
2. $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r)) \to \bot$*;*
3. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$*;*
4. $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$*.*

Proof. Apply Definition 45.527 to the same carried histories and threefold support witness. The endpoint data recovered there includes $\mathsf{RatHistoryCarrier}(x)$ and the singleton endpoint rejection map $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$. Fix an arbitrary context history $r : \mathsf{BHist}$. For the right-factor append boundary, apply Definition 45.101 with context r and RatUp factor x. The Rat carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x))$ into contradiction. For the left-factor append boundary, apply Definition 45.102 with the same RatUp factor x and context r. The same carrier witness turns any witness of $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r))$ into contradiction. It remains to record the two inverse-domain witnesses. Unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$ as rejection of the singleton classifier against e. Any such classifier witness contains the source-carrier component $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x))$, contradicting the right-factor append rejection. The left-factor inverse-domain witness is identical: unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$ as rejection of the singleton classifier against e. The source-carrier component fieldSingletonEmptyCarrier(append(x, contradicts the left-factor append rejection. □

Per-object template: traditional target concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ threefold denominator exit support; BEDC source BHist with $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; context $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations empty-unit continuation closure and Rat carrier readback; ledger left-nested denominator product endpoint $\mu^u_D(\mu^u_D(h,k),\ell)$; core theorems Definition 45.527, Definition 45.530, Definition 45.531, and Definition 45.532; exactness predicate threefold strict-support non-singleton boundary with contextual append separation and inverse-domain nonzero package; standard bridge finite denominator support into the concrete FieldUp exit; status $\mathsf{ProofTarget}$.

Per-object template: traditional target bracketing-independent concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ threefold denominator exit support; BEDC source BHist with $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; context $\mathsf{RatUpFieldSource}_F$;

carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations empty-unit continuation closure, associativity up to R^u_D, and Rat carrier readback; ledger paired left- and right-nested denominator product endpoints $\mu^u_D(\mu^u_D(h,k),\ell)$ and $\mu^u_D(h,\mu^u_D(k,\ell))$; core theorem Definition 45.528; exactness predicate threefold strict-support non-singleton boundary independent of product bracketing; standard bridge finite denominator support into the concrete FieldUp exit; status ProofTarget.

Theorem 45.533 (Rat denominator unit contextual action strict-support endpoint empty contradiction)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ*, assume the four external endpoints are empty and the five denominator contexts* h, l, r, l', r' *are carried. If the core has strict RatUp support, then the nested contextual action cannot be empty:*

$$N^u_D(h) \to \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))\mathsf{hsamee} \to \bot.$$

Proof. The contextual composition support law reads the nested support predicate as the disjunction of the two outer supports, the two inner supports, and the core support. Insert $N^u_D(h)$ into the core disjunct. This yields RatUp support for the nested action, contradicting its asserted empty endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction strict support endpoint empty absurd`

Theorem 45.534 (RatUp FieldUp assembled certificate operation-table coverage)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *of Definition 45.488 and the concrete operation table of Definition 45.587, one and the same* BHist *row package satisfies both Definition 45.337 and the concrete* $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ *operation-table laws. The shared carrier and classifier are* RatHistoryCarrier *and* RatHistoryClassifier*. The shared denominator coordinate is*

$$(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier}).$$

The shared contextual coordinate is the action Γ*, the shared context-pair coordinate is the monoid of Definition 45.479, and the shared affine coordinate is* AffAut_F*. The inverse and nonzero rows, together with the exit-noncollapse rows, are the rows consumed by both cited interfaces.*

Proof. Unpack Definition 45.488. It names RatHistoryCarrier, RatHistoryClassifier, the displayed denominator tuple, Γ, the context-pair monoid, and AffAut_F as coordinates of the single source $\mathsf{RatUpFieldSource}_F$.

Apply Definition 45.489. This gives Definition 45.337 for exactly those coordinates. Apply Definition 45.588. This gives the carrier, classifier, denominator, contextual-action, context-pair, affine, inverse, nonzero, and exit-noncollapse laws for the concrete operation table of Definition 45.587.

Both cited results name the same concrete carrier, classifier, denominator tuple, contextual action, context-pair monoid, affine row source, inverse rows, nonzero rows, and exit rows. Therefore the certificate-obligation interface and the operation-table interface are satisfied by the same concrete BHist rows. □

45.22 Field affine explicit inverse

Theorem 45.535 (Field two-sided affine expression has explicit inverse)**.** *For a BEDC history-level* $\mathsf{Field}^\uparrow$ *candidate with the same operations, additive laws, multiplication laws, congruence data, and indexed inverse laws as in Definition 45.277, let* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *be the two nonzero witnesses for the left and right affine factors. For each* $c : \mathsf{BHist}$ *define*

$$s(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Then every $c, x, y : \mathsf{BHist}$ *satisfies the following three clauses.*

1. $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, s(c)), b), d)\mathsf{hsame}c$.
2. *If* $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsame}c$, *then* $x\mathsf{hsame}s(c)$.
3. *If*
$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(a, y), b), d),$$
then $x\mathsf{hsame}y$.

Proof. For the displayed preimage, instantiate Definition 45.277 at $x = s(c)$. The reverse implication only needs the reflexive classifier equality $s(c)\mathsf{hsame}s(c)$, so it produces $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, s(c)), b), d)\mathsf{hsame}c$.

For uniqueness of any solution, use the forward implication of the same exactness theorem. An assumed classification of the affine expression at x with c is transported directly to $x\mathsf{hsame}s(c)$.

For equality of two affine expressions, apply Definition 45.278 to the assumed classifier equality. Its conclusion is exactly $x\mathsf{hsame}y$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine two sided explicit inverse`

Theorem 45.536 (Field two-sided affine map is a certified automorphism)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the same operations, additive laws, multiplication laws, congruence data, and indexed inverse laws as in Definition 45.277, let* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *be the two nonzero witnesses for the left and right affine factors. Define*
$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$
and, for each $c : \mathsf{BHist}$, *define*
$$s(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$
Then every $x, y, c : \mathsf{BHist}$ *satisfies*
$$T(s(c))\mathsf{hsame}c, \qquad s(T(x))\mathsf{hsame}x, \qquad T(x)\mathsf{hsame}T(y) \iff x\mathsf{hsame}y.$$

Proof. For the left inverse clause, instantiate Definition 45.277 at $x = s(c)$. The reflexive classifier equality $s(c)\mathsf{hsame}s(c)$ enters the reverse implication and yields $T(s(c))\mathsf{hsame}c$.

For the right inverse clause, instantiate the same exactness theorem with the target classifier value $T(x)$. The reflexive equality $T(x)\mathsf{hsame}T(x)$ enters the forward implication and yields $x\mathsf{hsame}s(T(x))$. Symmetry of hsame gives the displayed orientation $s(T(x))\mathsf{hsame}x$.

Finally apply Definition 45.278 with the same witnesses p_a, p_b and offset d. Its two directions are precisely the classifier-preservation and classifier-reflection clauses $T(x)\mathsf{hsame}T(y) \iff x\mathsf{hsame}y$. □

Per-object certificate: traditional target affine automorphism over a field. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist. Carrier: BHist. Classifier: hsame. Stability obligations: Definition 45.277 and Definition 45.278. Ledger: the two nonzero factor witnesses p_a, p_b and the offset d. Core theorem: Definition 45.536. Exactness predicate: inverse-pair plus classifier iff. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.537 (Field affine inverse coefficients are derived)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with associative multiplication, left and right unit laws, multiplication congruence, zero-left multiplication, indexed left inverse laws, one-apartness from* e, *and conversion from apartness from* e *to* $\mathsf{NonZero}_F$, *let* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$. *Then there are derived witnesses* $p_A : \mathsf{NonZero}_F(\mathsf{inv}(a, p_a))$ *and* $p_B : \mathsf{NonZero}_F(\mathsf{inv}(b, p_b))$ *such that*
$$\mathsf{inv}(\mathsf{inv}(a, p_a), p_A)\mathsf{hsame}a \qquad \textit{and} \qquad \mathsf{inv}(\mathsf{inv}(b, p_b), p_B)\mathsf{hsame}b.$$

Proof. Apply inverse nonzero from one-apartness to p_a and p_b separately. The two resulting nonzero witnesses make the inverse coefficients admissible inputs to the indexed inverse operation. For each coefficient, apply left-right inverse uniqueness to the pair consisting of the inverse law for the inverse coefficient and the inverse law for the original coefficient. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine inverse coefficients derived`

Theorem 45.538 (Field affine inverse offset empty normal form)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with multiplication associativity and congruence, addition congruence, and the two distributivity laws, every* $a, b, c, d : \mathsf{BHist}$ *with witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *satisfies*

$$\begin{aligned}&\mathsf{add}(\mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))), \mathsf{e})\\ &\quad \mathsf{hsameadd}(\mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(a, p_a), c), \mathsf{inv}(b, p_b)), \mathsf{mul}(\mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{neg}(d)), \mathsf{inv}(b, p_b))), \mathsf{e}).\end{aligned}$$

Proof. Reassociate multiplication so the left inverse coefficient acts directly on $c + \mathsf{neg}(d)$. Left distributivity separates the two summands, right distributivity carries the final right multiplication by the inverse of b through the separated sum, and addition congruence preserves the empty endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine inverse offset empty normal form`

Lemma 45.539 (Field affine inverse empty-offset normal form)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with right identity for the additive empty endpoint, multiplication associativity and congruence, addition congruence, and the two distributivity laws, every* $a, b, c, d : \mathsf{BHist}$ *with witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *satisfies*

$$\begin{aligned}&\mathsf{add}(\mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))), \mathsf{e})\\ &\quad \mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(a, p_a), c), \mathsf{inv}(b, p_b)), \mathsf{mul}(\mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).\end{aligned}$$

Proof. First use the inverse offset-empty normal form above to separate the two right-multiplied summands and retain the explicit empty endpoint. Then apply the additive right-identity law to remove that final empty endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine inverse empty offset normal form`

Theorem 45.540 (Field affine certified fibers are singleton classified)**.** *Under the hypotheses and notation of Definition 45.536, define*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$s(q) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Then every $q, x, y : \mathsf{BHist}$ *satisfies the following three clauses.*

1. $T(s(q))\mathsf{hsame}q$.
2. *If* $T(x)\mathsf{hsame}q$ *and* $T(y)\mathsf{hsame}q$*, then* $x\mathsf{hsame}y$.
3. $T(x)\mathsf{hsame}q \iff x\mathsf{hsame}s(q)$.

Proof. The displayed preimage clause is the left inverse clause of Definition 45.536 at the target history q.

For two solutions of the same fiber equation, assume $T(x)\mathsf{hsame}q$ and $T(y)\mathsf{hsame}q$. Apply Definition 45.278 to the comparison $T(x)\mathsf{hsame}T(y)$ obtained by composing the first fiber equation with the symmetry of the second. The reflected classifier comparison gives $x\mathsf{hsame}y$.

It remains to identify membership in the fiber with classification by the displayed solution. In the forward direction, assume $T(x)\mathsf{hsame}q$. The first clause gives $T(s(q))\mathsf{hsame}q$; by symmetry and transitivity, $T(x)$ is classified with $T(s(q))$. Reflect this comparison through the classifier clause of Definition 45.536, giving $x\mathsf{hsame}s(q)$.

For the reverse direction, assume $x\mathsf{hsame}s(q)$. Preserve this classifier comparison through the same certified affine map to obtain $T(x)\mathsf{hsame}T(s(q))$, and compose with the displayed preimage clause $T(s(q))\mathsf{hsame}q$. The result is $T(x)\mathsf{hsame}q$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine certified fiber singleton`

Per-object certificate: traditional target affine bijection fiber exactness; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations certified affine automorphism and affine equation uniqueness; ledger the two nonzero witnesses, offset, target point, and displayed solution equations; core theorem Definition 45.540; exactness predicate singleton classified fiber; standard bridge ordinary affine bijections over a field; status $\mathsf{ProofTarget}$.

Theorem 45.541 (Field two-sided affine automorphism composition with explicit product witnesses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with the additive group laws, multiplication associativity, left and right multiplicative unit laws, multiplication and addition congruence, distributivity fields inherited from* $\mathsf{CommRing}^{\uparrow}$*, indexed inverse laws, classifier transport of* $\mathsf{NonZero}_F$*, and contradiction of* $\mathsf{NonZero}_F(0)$*, let* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*, and* $p_e : \mathsf{NonZero}_F(e)$ *be explicit nonzero witnesses. Define*

$$T_1(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$T_2(y) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, y), e), f).$$

Let $\mathsf{ApartZero}_F(h)$ *abbreviate* $h\mathsf{hsame}0 \to \bot$*, and assume conversion maps*

$$\alpha_{ca} : \mathsf{ApartZero}_F(\mathsf{mul}(c, a)) \to \mathsf{NonZero}_F(\mathsf{mul}(c, a)), \qquad \alpha_{be} : \mathsf{ApartZero}_F(\mathsf{mul}(b, e)) \to \mathsf{NonZero}_F(\mathsf{mul}(b, e)).$$

Let q_{ca} *and* q_{be} *be the product apartness witnesses obtained from Definition 45.248 for the pairs* (p_c, p_a) *and* (p_b, p_e)*, and set* $p_{ca} := \alpha_{ca}(q_{ca})$ *and* $p_{be} := \alpha_{be}(q_{be})$*. Define*

$$C(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(c, a), x), \mathsf{mul}(b, e)), \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, d), e), f)).$$

Then every $x : \mathsf{BHist}$ *satisfies*

$$T_2(T_1(x))\mathsf{hsame}C(x).$$

Moreover, with the product witnesses p_{ca} *and* p_{be}*, the displayed composite* C *is a certified two-sided affine automorphism by Definition 45.536.*

Proof. Expand $T_2(T_1(x))$ and use distributivity to separate the translated part coming from d from the term containing x. Addition congruence transports the two pieces through the outer sum, while multiplication congruence transports the inner affine equality through the left factor c and the right factor e.

Multiplication associativity then collects the x-term into $\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(c,a),x),\mathsf{mul}(b,e))$. The same associativity and distributivity data identify the translated remainder as $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),f)$. This gives the displayed normal form $T_2(T_1(x))\mathsf{hsame}C(x)$.

It remains to provide the two nonzero indices required by Definition 45.536. Apply Definition 45.248 to p_c, p_a to obtain $q_{ca} : \mathsf{ApartZero}_F(\mathsf{mul}(c,a))$, and to p_b, p_e to obtain $q_{be} : \mathsf{ApartZero}_F(\mathsf{mul}(b,e))$. The conversion maps α_{ca} and α_{be} turn these apartness witnesses into p_{ca} and p_{be}.

Now instantiate Definition 45.536 with left factor $\mathsf{mul}(c,a)$, right factor $\mathsf{mul}(b,e)$, offset $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),f)$, and witnesses p_{ca}, p_{be}. The theorem supplies the inverse-pair clauses and classifier iff for C, so the normalized composite is again certified. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine two sided automorphism composition`

Theorem 45.542 (Field affine empty-offset composition normal form)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with addition associativity and congruence, multiplication associativity and congruence, and the two distributivity laws, every* $a,b,c,d,e,x : \mathsf{BHist}$ *satisfies*

$$\begin{aligned}&\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a,x),b),d)),e),\mathsf{e})\\&\quad\mathsf{hsameadd}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(c,a),x),\mathsf{mul}(b,e)),\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),\mathsf{e})).\end{aligned}$$

Proof. Distribute the left multiplication by c across the inner affine offset and then distribute the right multiplication by e across the resulting sum. This separates the x-term from the offset term $\mathsf{mul}(\mathsf{mul}(c,d),e)$. Multiplication associativity rewrites the x-term from $(c((ax)b))e$ to $((ca)x)(be)$. Addition congruence transports this product normal form through the surrounding sum, and addition associativity places the empty offset on the final translated component. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine empty offset composition normal form`

Theorem 45.543 (Field affine empty-offset composition classifier congruence)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with addition and multiplication congruence, if the affine coefficients satisfy* $a\mathsf{hsame}a'$*,* $b\mathsf{hsame}b'$*,* $c\mathsf{hsame}c'$*,* $d\mathsf{hsame}d'$*, and* $e\mathsf{hsame}e'$*, then the empty-offset composite expression*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a,x),b),d)),e),\mathsf{e})$$

is hsame*-equivalent to the same expression with the primed coefficients.*

Proof. Apply multiplication congruence to the inner product $\mathsf{mul}(a,x)$ using $x\mathsf{hsame}x$, then to the product with b. Addition congruence transports the inner affine offset using $d\mathsf{hsame}d'$. Two further multiplication-congruence steps transport the left factor c and right factor e. The final addition-congruence step uses $\mathsf{ehsame}e$ for the empty offset. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine empty offset composition classifier congr`

Theorem 45.544 (Field affine composition right empty normal form)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with addition associativity and congruence, multiplication associativity and congruence, and the two distributivity laws, every* $a,b,c,d,e,f,x : \mathsf{BHist}$ *satisfies*

$$\begin{aligned}&\mathsf{add}(\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,\mathsf{add}(\mathsf{mul}(\mathsf{mul}(a,x),b),d)),e),f),\mathsf{e})\\&\quad\mathsf{hsameadd}(\mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(c,a),x),\mathsf{mul}(b,e)),\mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),f)),\mathsf{e}).\end{aligned}$$

Proof. Distribute the left multiplication by c across the inner affine offset and then distribute the right multiplication by e across the resulting sum. Multiplication associativity collects the x-coefficient as $\mathsf{mul}(\mathsf{mul}(c,a),x)$ and the right coefficient as $\mathsf{mul}(b,e)$. Addition associativity places the outer offset f on the translated component before the empty endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine composition right empty normal form`

Per-object certificate: traditional target affine automorphism composition. BEDC source: $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist. Carrier: BHist. Classifier: hsame. Stability obligations: composition normal form plus product apartness conversion. Ledger: the four factor witnesses p_a, p_b, p_c, p_e and the two product witnesses p_{ca}, p_{be}. Core theorem: Definition 45.541. Exactness predicate: composite certified automorphism. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.545 (Field affine composite has explicit two-sided inverse)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate under the hypotheses of Definition 45.541, let* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_{a'} : \mathsf{NonZero}_F(a')$*, and* $p_{b'} : \mathsf{NonZero}_F(b')$ *be the four nonzero witnesses for the two affine certificates. Define*

$$T_1(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$T_2(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a', x), b'), d').$$

For each $r : \mathsf{BHist}$ *define*

$$s_1(r) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(r, \mathsf{neg}(d)), \mathsf{inv}(b, p_b)))$$

and

$$s_2(r) = \mathsf{mul}(\mathsf{inv}(a', p_{a'}), \mathsf{mul}(\mathsf{add}(r, \mathsf{neg}(d')), \mathsf{inv}(b', p_{b'}))).$$

Then every $x, y, q : \mathsf{BHist}$ *satisfies the following three clauses.*

1.

$$T_2(T_1(x))\mathsf{hsame}T_2(T_1(y)) \iff x\mathsf{hsame}y.$$

2.

$$T_2(T_1(s_1(s_2(q))))\mathsf{hsame}q.$$

3.

$$s_1(s_2(T_2(T_1(x))))\mathsf{hsame}x.$$

Proof. Instantiate Definition 45.541 with $c = a'$, $e = b'$, $f = d'$, and the product conversion maps for the collected factors. Its collected-factor normal form, together with Definition 45.536, supplies the classifier exactness for the composite, giving the displayed iff clause.

For the left inverse clause, first use the left inverse of the T_2 certificate at q, which gives $T_2(s_2(q))\mathsf{hsame}q$. Then use the left inverse of the T_1 certificate at $s_2(q)$, giving

$$T_1(s_1(s_2(q)))\mathsf{hsame}s_2(q).$$

Congruence of T_2 transports this equality through the outer map, and transitivity yields

$$T_2(T_1(s_1(s_2(q))))\mathsf{hsame}q.$$

For the right inverse clause, the right inverse of the T_2 certificate gives

$$s_2(T_2(T_1(x)))\mathsf{hsame}T_1(x).$$

Apply the left inverse clause of the T_1 certificate to $s_2(T_2(T_1(x)))$, obtaining

$$T_1(s_1(s_2(T_2(T_1(x)))))\mathsf{hsame}s_2(T_2(T_1(x))).$$

Compose these two equalities to classify

$$T_1(s_1(s_2(T_2(T_1(x)))))$$

with $T_1(x)$, and reflect the result through the T_1 classifier exactness to conclude $s_1(s_2(T_2(T_1(x))))\mathsf{hsame}x$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine composite two sided inverse`

Per-object certificate: traditional target affine automorphism inverse for composition; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations the two affine certificates plus composition classifier exactness; ledger four nonzero witnesses and two offsets; core theorem Definition 45.545; exactness predicate two-sided inverse plus classifier iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.546 (Field affine composite inverse agrees with reverse-composed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate with the operations and laws used by Definition 45.541, let* $p_a : \mathsf{NonZero}_F(a)$, $p_b : \mathsf{NonZero}_F(b)$, $p_{a'} : \mathsf{NonZero}_F(a')$, *and* $p_{b'} : \mathsf{NonZero}_F(b')$ *be the nonzero witnesses for the two component affine maps*

$$T_1(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$T_2(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a', x), b'), d').$$

Put $A = \mathsf{mul}(a', a)$, $B = \mathsf{mul}(b, b')$, *and* $D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a', d), b'), d')$. *Let* $\mathsf{ApartZero}_F(h)$ *abbreviate* $h\mathsf{hsame}0 \to \bot$, *and assume conversion maps*

$$\alpha_A : \mathsf{ApartZero}_F(A) \to \mathsf{NonZero}_F(A), \qquad \alpha_B : \mathsf{ApartZero}_F(B) \to \mathsf{NonZero}_F(B).$$

Let q_A *and* q_B *be the product apartness witnesses obtained from Definition 45.248 for the pairs* $(p_{a'}, p_a)$ *and* $(p_b, p_{b'})$, *and set* $p_A := \alpha_A(q_A)$ *and* $p_B := \alpha_B(q_B)$. *Assume the collected-coefficient presentation is pointwise classified with the displayed composite action:*

$$\mathsf{add}(\mathsf{mul}(\mathsf{mul}(A, x), B), D)\mathsf{hsame}T_2(T_1(x)).$$

For each $c : \mathsf{BHist}$ *define*

$$s_1(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b)))$$

and

$$s_2(c) = \mathsf{mul}(\mathsf{inv}(a', p_{a'}), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d')), \mathsf{inv}(b', p_{b'}))).$$

For each $q : \mathsf{BHist}$ *define the composite explicit inverse*

$$s_{21}(q) = \mathsf{mul}(\mathsf{inv}(A, p_A), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(D)), \mathsf{inv}(B, p_B))).$$

Then every $q : \mathsf{BHist}$ *satisfies*

$$s_{21}(q)\mathsf{hsame}s_1(s_2(q)).$$

Lean4 ✓ `BEDC.Derived.FieldUp.field affine composite inverse reverse coherence`

Proof. Let

$$C(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(A, x), B), D).$$

Apply Definition 45.535 to the composite coefficients A, B, D with witnesses p_A, p_B. Its left inverse clause gives $C(s_{21}(q))\mathsf{hsame}q$.

Next instantiate Definition 45.536 for the first component with witnesses p_a, p_b and offset d, and again for the second component with witnesses $p_{a'}, p_{b'}$ and offset d'. The first certificate gives $T_1(s_1(s_2(q)))\mathsf{hsame}s_2(q)$, and the second gives $T_2(s_2(q))\mathsf{hsame}q$.

Use the classifier-preservation direction of the second certificate to transport the first displayed equality through T_2. Transitivity with $T_2(s_2(q))\mathsf{hsame}q$ yields $T_2(T_1(s_1(s_2(q))))\mathsf{hsame}q$.

By Definition 45.541 with the substitution $c = a'$, $e = b'$, and $f = d'$, the left side of this equation is classified by $C(s_1(s_2(q)))$ with the collected coefficients A, B, D, and the certified automorphism status uses the converted witnesses p_A, p_B. Hence $s_1(s_2(q))$ also solves the same composite affine equation $C(x)\mathsf{hsame}q$.

Finally apply Definition 45.298 to the two solutions $s_{21}(q)$ and $s_1(s_2(q))$ of $C(x)\mathsf{hsame}q$. The unique-solution conclusion is exactly $s_{21}(q)\mathsf{hsame}s_1(s_2(q))$. □

Per-object certificate: traditional target affine composite inverse coherence; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist with RingUp distributivity; carrier BHist; classifier hsame; stability obligations explicit inverse exactness, certified component automorphisms, composite coefficient collection, product apartness, and affine equation uniqueness; ledger the four component factor witnesses, the two product witnesses, and the two offsets; core theorem Definition 45.546; exactness predicate reverse-composed explicit inverse equals the collected-coefficient explicit inverse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.547 (Field affine composite has reverse explicit inverse)**.** *Under the hypotheses and notation of Definition 45.536 and Definition 45.541, define*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$U(z) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, z), e), f).$$

Let s_T and s_U be the displayed explicit inverse maps associated to T and U. Then every $x, y, q :$ BHist satisfies

$$U(T(s_T(s_U(q))))\mathsf{hsame}q, \qquad s_T(s_U(U(T(x))))\mathsf{hsame}x, \qquad U(T(x))\mathsf{hsame}U(T(y)) \Longleftrightarrow x\mathsf{hsame}y.$$

Proof. First instantiate Definition 45.541 with the factors and offsets defining T and U. The composition theorem identifies $U \circ T$ as the certified affine composite, and Definition 45.536 supplies the classifier-iff clause for that certified composite.

For the displayed left preimage, apply the left-inverse clause for T at $s_U(q)$, then apply the left-inverse clause for U at q. Congruence of U and transitivity of hsame identify the transported composite with $U(T(s_T(s_U(q))))\mathsf{hsame}q$.

For the right inverse, apply the right-inverse clause of Definition 45.536 to U at the point $T(x)$. This gives

$$s_U(U(T(x)))\mathsf{hsame}T(x).$$

Use the solution-uniqueness clause of Definition 45.535 for T at the target $T(x)$, with the reflexive classification of $T(x)$ as its input. This gives $x\mathsf{hsame}s_T(T(x))$, hence $s_T(T(x))\mathsf{hsame}x$ by symmetry. Transport the displayed equality through the explicit formula for s_T, and finish by transitivity of hsame to obtain $s_T(s_U(U(T(x))))\mathsf{hsame}x$. □

Per-object certificate: traditional target affine automorphism group law; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations two certified affine automorphisms, explicit inverse, and classifier iff; ledger four nonzero factor witnesses and two offsets; core theorem Definition 45.547; exactness predicate two-sided inverse plus composite classifier iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.548 (Field affine composite inverse product-witness independence)**.** *Under the notation of Definition 45.546, fix the collected coefficients*

$$A = \mathsf{mul}(a', a), \qquad B = \mathsf{mul}(b, b'), \qquad D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a', d), b'), d').$$

Let $p_A, p'_A : \mathsf{NonZero}_F(A)$ and $p_B, p'_B : \mathsf{NonZero}_F(B)$ be two choices of product nonzero witnesses. For each $q : \mathsf{BHist}$ define

$$s_{21}^{p_A,p_B}(q) = \mathsf{mul}(\mathsf{inv}(A, p_A), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(D)), \mathsf{inv}(B, p_B)))$$

and define $s_{21}^{p'_A,p'_B}(q)$ by the same displayed formula with p'_A, p'_B in place of p_A, p_B. Then every $q : \mathsf{BHist}$ satisfies the three comparisons

1.

$$s_{21}^{p_A,p_B}(q)\mathsf{hsame}s_{21}^{p'_A,p'_B}(q).$$

2.

$$s_{21}^{p_A,p_B}(q)\mathsf{hsame}s_1(s_2(q)).$$

3.

$$s_{21}^{p'_A,p'_B}(q)\mathsf{hsame}s_1(s_2(q)).$$

Proof. We first compare the two collected inverse terms. Apply Definition 45.557 to the two composite affine members with common coefficients A, B, D and witness pairs (p_A, p_B) and (p'_A, p'_B). Its inverse-formula field gives the first displayed comparison pointwise in q.

For agreement with the reverse-composed inverse, apply Definition 45.546 with the witness pair (p_A, p_B) to obtain the second displayed comparison. Apply the same theorem with (p'_A, p'_B) to obtain the third displayed comparison.

The classifier hsame is transitive and symmetric, so these three comparisons form a single certificate: changing product witnesses does not change the collected explicit inverse, and each collected representative agrees with the reverse-composed inverse $s_1(s_2(q))$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine composite inverse product witness independence`

Per-object certificate: traditional target affine composite inverse ledger determinacy; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations Definition 45.557, Definition 45.546, and hsame transitivity; ledger the component witnesses, product witnesses, and offsets; core theorem Definition 45.548; exactness predicate witness-independent collected inverse agreeing with the reverse-composed inverse; downstream consumer affine automorphism group composition certificate; status $\mathsf{ProofTarget}$.

Theorem 45.549 (Field affine inverse is a certified affine map)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate under the hypotheses of Definition 45.541, let $p_a : \mathsf{NonZero}_F(a)$ and $p_b : \mathsf{NonZero}_F(b)$ be the two nonzero witnesses for*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d).$$

For each $c : \mathsf{BHist}$ *put*

$$s(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Let $A := \mathsf{inv}(a, p_a)$ *and* $B := \mathsf{inv}(b, p_b)$*, and suppose* $p_A : \mathsf{NonZero}_F(A)$ *and* $p_B : \mathsf{NonZero}_F(B)$ *are explicit nonzero witnesses for the inverse coefficients. Define*

$$D := \mathsf{mul}(\mathsf{mul}(A, \mathsf{neg}(d)), B)$$

and

$$S(c) := \mathsf{add}(\mathsf{mul}(\mathsf{mul}(A, c), B), D).$$

Then every $c : \mathsf{BHist}$ *satisfies* $s(c)\mathsf{hsame}S(c)$*, and with witnesses* p_A, p_B *the map* S *is a certified two-sided affine automorphism by Definition 45.536.*

Proof. Expand $s(c)$ and apply the distributivity field from $\mathsf{CommRing}^{\uparrow}$ to the inner sum $c + \mathsf{neg}(d)$ after left multiplication by A. This separates the term containing c from the translated term containing $\mathsf{neg}(d)$.

Next reassociate the right multiplication by B. Multiplication associativity identifies the c-term with $\mathsf{mul}(\mathsf{mul}(A, c), B)$ and identifies the translated term with

$$\mathsf{mul}(\mathsf{mul}(A, \mathsf{neg}(d)), B).$$

The definition of D then gives $s(c)\mathsf{hsame}S(c)$ for every $c : \mathsf{BHist}$.

It remains to certify the displayed normal form. Apply Definition 45.536 with left factor A, right factor B, offset D, and the explicit inverse-coefficient witnesses p_A, p_B. The resulting inverse-pair and classifier-iff clauses are exactly the certified affine automorphism structure for S. □

Theorem 45.550 (Field affine inverse has derived inverse-coefficient witnesses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate under the hypotheses of Definition 45.549 before the inverse-coefficient witnesses are supplied, also assume zero-left multiplication* $\mathsf{mul}(0, x)\mathsf{hsame}0$*, one-apartness* $\omega : (1\mathsf{hsame}0) \to \mathsf{False}$*, and conversion maps* $\alpha_x : ((x\mathsf{hsame}0) \to \mathsf{False}) \to \mathsf{NonZero}_F(x)$*. Let* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *be the nonzero witnesses for*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d).$$

For each $c : \mathsf{BHist}$ *put*

$$s(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Let $A := \mathsf{inv}(a, p_a)$ *and* $B := \mathsf{inv}(b, p_b)$*. Then there are derived witnesses* $p_A : \mathsf{NonZero}_F(A)$ *and* $p_B : \mathsf{NonZero}_F(B)$ *such that, for*

$$D := \mathsf{mul}(\mathsf{mul}(A, \mathsf{neg}(d)), B)$$

and

$$S(c) := \mathsf{add}(\mathsf{mul}(\mathsf{mul}(A, c), B), D),$$

every $c : \mathsf{BHist}$ *satisfies* $s(c)\mathsf{hsame}S(c)$*, and* S *is a certified two-sided affine automorphism.*

Proof. Apply Definition 45.207 to p_a with the displayed zero-left multiplication law, one-apartness witness ω, and conversion map α_A. This produces $p_A : \mathsf{NonZero}_F(\mathsf{inv}(a, p_a))$, which is $\mathsf{NonZero}_F(A)$ by the definition of A.

Apply the same theorem to p_b with the same structural data and conversion map α_B. This gives $p_B : \mathsf{NonZero}_F(\mathsf{inv}(b, p_b))$, hence $\mathsf{NonZero}_F(B)$.

With p_A and p_B now available, invoke Definition 45.549. Its normal-form clause gives $s(c)\mathsf{hsame}S(c)$ for every $c : \mathsf{BHist}$, and its certification clause is exactly the certified two-sided affine automorphism structure for S. □

Per-object certificate: traditional target $\mathsf{Field}^{\uparrow}$ affine inverse witness derivation; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations zero-left multiplication, one-apartness, inverse laws, and affine normal-form certification; ledger the original coefficient witnesses and the derived inverse-coefficient witnesses; core theorem Definition 45.550; exactness predicate inverse affine normal form with derived witnesses; standard bridge ordinary affine automorphism group; status $\mathsf{ProofTarget}$.

Per-object certificate: traditional target inverse closure for affine automorphisms; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations affine equation exactness, distributivity, associativity, inverse laws, and nonzero witnesses for inverse coefficients; ledger the original and inverse coefficient witnesses plus offset; core theorem Definition 45.549; exactness predicate inverse affine normal form; standard bridge ordinary affine automorphism group; status $\mathsf{ProofTarget}$.

Theorem 45.551 (Field affine explicit inverse is a certified automorphism)**.** *Under the hypotheses and notation of Definition 45.536, define*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and, for each $r : \mathsf{BHist}$*, define*

$$s(r) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(r, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

Then every $c, e, x : \mathsf{BHist}$ *satisfies*

$$s(T(x))\mathsf{hsame}x, \qquad T(s(c))\mathsf{hsame}c, \qquad s(c)\mathsf{hsame}s(e) \iff c\mathsf{hsame}e.$$

Proof. Apply Definition 45.536 with witnesses p_a, p_b and offset d. Its inverse-pair clauses give $s(T(x))\mathsf{hsame}x$ and $T(s(c))\mathsf{hsame}c$ directly.

For the forward direction of the inverse-map classifier iff, assume $c\mathsf{hsame}e$. Compose $T(s(c))\mathsf{hsame}c$, the assumed equality, and the symmetry of $T(s(e))\mathsf{hsame}e$ to obtain

$$T(s(c))\mathsf{hsame}T(s(e)).$$

Reflect this equality through the certified classifier iff for T to conclude $s(c)\mathsf{hsame}s(e)$.

For the reverse direction, assume $s(c)\mathsf{hsame}s(e)$. The classifier-preservation direction for T gives

$$T(s(c))\mathsf{hsame}T(s(e)).$$

Compose the symmetry of $T(s(c))\mathsf{hsame}c$, the displayed equality, and $T(s(e))\mathsf{hsame}e$ to obtain $c\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine explicit inverse certified automorphism`

Per-object certificate: traditional target inverse affine automorphism; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations certified affine automorphism, inverse-pair clauses, and classifier iff; ledger the two nonzero witnesses and offset; core theorem Definition 45.551; exactness predicate inverse-map classifier iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.552 (Field affine explicit inverse is classifier-congruent)**.** *Under the hypotheses and notation of Definition 45.536, define*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

and

$$s(c) = \mathsf{mul}(\mathsf{inv}(a, p_a), \mathsf{mul}(\mathsf{add}(c, \mathsf{neg}(d)), \mathsf{inv}(b, p_b))).$$

If $c\mathsf{hsame}c'$*, then*

$$s(c)\mathsf{hsame}s(c').$$

Proof. The left inverse clauses of Definition 45.536 give $T(s(c))\mathsf{hsame}c$ and $T(s(c'))\mathsf{hsame}c'$. Compose the first equality with $c\mathsf{hsame}c'$ and with the symmetry of the second equality. This produces

$$T(s(c))\mathsf{hsame}T(s(c')).$$

The classifier iff in Definition 45.536 reflects equality of T-images back to equality of inputs. Applying that reflection direction to the displayed classification yields $s(c)\mathsf{hsame}s(c')$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine explicit inverse classifier congruence`

Theorem 45.553 (Field affine pointwise classifier gives operation congruence)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate under the hypotheses of Definition 45.536, Definition 45.541, and Definition 45.549, let* T, T', U, U' *be certified displayed affine maps over* BHist*. Write* T^{-1}_{exp} *for the displayed explicit inverse formula supplied by Definition 45.536, and similarly for the other certified maps. Suppose* $\equiv_{\mathrm{aff}}$ *is the pointwise classifier of Definition 45.560. Then the following clauses hold.*

1. *If* $T \equiv_{\mathrm{aff}} T'$ *and* $U \equiv_{\mathrm{aff}} U'$*, then*
$$U \circ T \equiv_{\mathrm{aff}} U' \circ T'.$$
2. *If* $T \equiv_{\mathrm{aff}} U$*, then*
$$T^{-1}_{\mathrm{exp}} \equiv_{\mathrm{aff}} U^{-1}_{\mathrm{exp}}.$$

Proof. For composition, fix $x : \mathsf{BHist}$. The assumption $T \equiv_{\mathrm{aff}} T'$ gives $T(x)\mathsf{hsame}T'(x)$. The classifier-preservation direction of the certified affine automorphism structure for U transports this to

$$U(T(x))\mathsf{hsame}U(T'(x)).$$

The assumption $U \equiv_{\mathrm{aff}} U'$ at the point $T'(x)$ gives

$$U(T'(x))\mathsf{hsame}U'(T'(x)).$$

Transitivity of hsame gives $U(T(x))\mathsf{hsame}U'(T'(x))$ for every x, which is exactly $U \circ T \equiv_{\mathrm{aff}} U' \circ T'$.

For inverse congruence, fix $q : \mathsf{BHist}$. The explicit inverse clauses give $T(T^{-1}_{\exp}(q))\mathsf{hsame}q$ and $U(U^{-1}_{\exp}(q))\mathsf{hsame}q$. The pointwise comparison $T \equiv_{\mathrm{aff}} U$ at $U^{-1}_{\exp}(q)$ gives

$$T(U^{-1}_{\exp}(q))\mathsf{hsame}U(U^{-1}_{\exp}(q)).$$

Thus $T(U^{-1}_{\exp}(q))\mathsf{hsame}q$. The two-sided affine equation uniqueness theorem Definition 45.298, applied to the two displayed preimages of q under T, gives $T^{-1}_{\exp}(q)\mathsf{hsame}U^{-1}_{\exp}(q)$. Since q was arbitrary, the explicit inverse formulas are related by $\equiv_{\mathrm{aff}}$. □

Per-object certificate: traditional target affine inverse stability; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations certified affine automorphism, left inverse, and classifier reflection; ledger the two nonzero witnesses and offset; core theorem Definition 45.552; exactness predicate inverse-map classifier congruence; standard bridge ordinary affine inverse congruence; status $\mathsf{ProofTarget}$.

Per-object certificate: traditional target affine automorphism group law; BEDC source $\mathsf{Field}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations two certified affine automorphisms, explicit inverse, and classifier iff; ledger four nonzero factor witnesses and two offsets; core theorem Definition 45.547; exactness predicate two-sided inverse plus composite classifier iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.554 (Field affine automorphism composition is associative)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$ *candidate under the hypotheses of Definition 45.541, let* $p_a, p_b, p_c, p_e, p_g, p_i$ *be nonzero witnesses for the six displayed factors in the certified affine maps*

$$T_1(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d),$$

$$T_2(y) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, y), e), f),$$

and

$$T_3(z) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(g, z), i), j).$$

Let $\mathsf{ApartZero}_F(h)$ *abbreviate* $h\mathsf{hsame}0 \to \bot$. *Assume the product-apartness conversion maps required to certify the two binary composites and the two outer composites:*

$\alpha^L_{21} : \mathsf{ApartZero}_F(\mathsf{mul}(c, a)) \to \mathsf{NonZero}_F(\mathsf{mul}(c, a))$, $\alpha^R_{21} : \mathsf{ApartZero}_F(\mathsf{mul}(b, e)) \to \mathsf{NonZero}_F(\mathsf{mul}(b, e))$,

$\alpha^L_{32} : \mathsf{ApartZero}_F(\mathsf{mul}(g, c)) \to \mathsf{NonZero}_F(\mathsf{mul}(g, c))$, $\alpha^R_{32} : \mathsf{ApartZero}_F(\mathsf{mul}(e, i)) \to \mathsf{NonZero}_F(\mathsf{mul}(e, i))$,

$\alpha^L_{3(21)} : \mathsf{ApartZero}_F(\mathsf{mul}(g, \mathsf{mul}(c, a))) \to \mathsf{NonZero}_F(\mathsf{mul}(g, \mathsf{mul}(c, a)))$, $\alpha^R_{3(21)} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(b, e), i)) \to$

and

$\alpha^L_{(32)1} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(g, c), a)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(g, c), a))$, $\alpha^R_{(32)1} : \mathsf{ApartZero}_F(\mathsf{mul}(b, \mathsf{mul}(e, i))) \to$

Put

$$A_{21} = \mathsf{mul}(c, a), \quad B_{21} = \mathsf{mul}(b, e), \quad D_{21} = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, d), e), f),$$
$$A_{32} = \mathsf{mul}(g, c), \quad B_{32} = \mathsf{mul}(e, i), \quad D_{32} = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(g, f), i), j).$$

The bracketing $T_3 \circ (T_2 \circ T_1)$ *supplies*

$$C_L(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(g, A_{21}), x), \mathsf{mul}(B_{21}, i)), \mathsf{add}(\mathsf{mul}(\mathsf{mul}(g, D_{21}), i), j)),$$

and the bracketing $(T_3 \circ T_2) \circ T_1$ *supplies*

$$C_R(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(A_{32}, a), x), \mathsf{mul}(b, B_{32})), \mathsf{add}(\mathsf{mul}(\mathsf{mul}(A_{32}, d), B_{32}), D_{32})).$$

Then every $x, q : \mathsf{BHist}$ *satisfies*

$$C_L(x)\mathsf{hsame}C_R(x).$$

Moreover, if s_1, s_2, s_3 *are the reverse explicit inverse formulas attached to* T_1, T_2, T_3*, if* s_{21} *is the reverse explicit inverse for the certified composite* $T_2 \circ T_1$*, and if* s_{32} *is the reverse explicit inverse for* $T_3 \circ T_2$*, then the two bracketed reverse inverse formulas classify together:*

$$s_{21}(s_3(q))\mathsf{hsame}s_1(s_{32}(q)).$$

Proof. First certify the inner composites. Apply Definition 45.541 to $T_2 \circ T_1$ using $\alpha^L_{21}, \alpha^R_{21}$, and apply the same theorem to $T_3 \circ T_2$ using $\alpha^L_{32}, \alpha^R_{32}$. The collected data are respectively A_{21}, B_{21}, D_{21} and A_{32}, B_{32}, D_{32}.

Next certify the two outer composites. Apply Definition 45.541 to $T_3 \circ (T_2 \circ T_1)$ using $\alpha^L_{3(21)}, \alpha^R_{3(21)}$, and apply it to $(T_3 \circ T_2) \circ T_1$ using $\alpha^L_{(32)1}, \alpha^R_{(32)1}$. Both certified maps classify the same term $T_3(T_2(T_1(x)))$.

The collected left factors are $\mathsf{mul}(g, \mathsf{mul}(c, a))$ and $\mathsf{mul}(\mathsf{mul}(g, c), a)$, the collected right factors are $\mathsf{mul}(\mathsf{mul}(b, e), i)$ and $\mathsf{mul}(b, \mathsf{mul}(e, i))$. Multiplication associativity identifies each pair.

It remains to compare the offsets. Expand D_{21} and D_{32}. Distributivity moves the middle offset f through the final affine map, and multiplication associativity collects the factors multiplying d. The two displayed offsets therefore classify together, so the normal forms $C_L(x)$ and $C_R(x)$ classify together for every x.

For the inverse formulas, apply Definition 45.546 to $T_2 \circ T_1$ and then compose the resulting equality with $s_3(q)$. Apply the same coherence theorem to $T_3 \circ T_2$ at q, then transport through s_1. Transitivity gives $s_{21}(s_3(q))\mathsf{hsame}s_1(s_2(s_3(q)))\mathsf{hsame}s_1(s_{32}(q))$, which is the displayed equality. □

Per-object certificate: traditional target affine automorphism associativity; BEDC source $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations binary affine composition, product apartness, distributivity, inverse coherence; ledger the six factor witnesses, product witnesses, and three offsets; core theorem Definition 45.554; exactness predicate associativity of certified affine composition; standard bridge ordinary affine group law; status $\mathsf{ProofTarget}$.

Theorem 45.555 (Field affine automorphism identity and inverse package)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *under the hypotheses used by Definition 45.536 and Definition 45.541, bind the unit witness* $p_1 : \mathsf{NonZero}_F(1)$ *and define*

$$I(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(1, x), 1), 0).$$

For every certified two-sided affine map

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d)$$

with witnesses $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, every* $x : \mathsf{BHist}$ *satisfies*

$$I(x)\mathsf{hsame}x, \qquad I(T(x))\mathsf{hsame}T(x), \qquad T(I(x))\mathsf{hsame}T(x).$$

For every second certified two-sided affine map

$$U(z) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, z), e), f)$$

with witnesses $p_c : \mathsf{NonZero}_F(c)$ *and* $p_e : \mathsf{NonZero}_F(e)$*, let* $\mathsf{ApartZero}_F(h)$ *abbreviate* $h\mathsf{hsame}0 \to \bot$*. Assume conversion maps*

$$\alpha_{ca} : \mathsf{ApartZero}_F(\mathsf{mul}(c,a)) \to \mathsf{NonZero}_F(\mathsf{mul}(c,a)), \qquad \alpha_{be} : \mathsf{ApartZero}_F(\mathsf{mul}(b,e)) \to \mathsf{NonZero}_F(\mathsf{mul}(b,e)).$$

Let q_{ca} *and* q_{be} *be the product-apartness witnesses supplied by Definition 45.248 for the pairs* (p_c, p_a) *and* (p_b, p_e)*, and set* $p_{ca} := \alpha_{ca}(q_{ca})$ *and* $p_{be} := \alpha_{be}(q_{be})$*. Then the composite* $U(T(x))$ *is certified by the collected coefficients*

$$A = \mathsf{mul}(c,a), \qquad B = \mathsf{mul}(b,e), \qquad D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),f)$$

together with the product witnesses p_{ca} *and* p_{be}*. Moreover, if* s_T *and* s_U *are the displayed inverse maps for* T *and* U*, then* $s_T(s_U(q))$ *is a two-sided inverse of* $U(T(x))$ *and agrees with the collected-coefficient inverse of the composite indexed by* p_{ca} *and* p_{be}*. Hence the identity affine map, composition, inverse closure, and classifier iff clauses form the affine automorphism package over* BHist*.*

Proof. The identity clauses use the same $\mathsf{Field}^{\uparrow}$ operations as the certified affine maps. The multiplicative unit laws reduce $\mathsf{mul}(\mathsf{mul}(1,x),1)$ to x, and the additive zero law reduces the outer addition to x; substituting $T(x)$ for x gives $I(T(x))\mathsf{hsame}T(x)$. Classifier preservation for T from Definition 45.536 transports $I(x)\mathsf{hsame}x$ through T, giving $T(I(x))\mathsf{hsame}T(x)$.

For composition, apply Definition 45.541 to the two certified affine maps using the displayed α_{ca} and α_{be} data. The product-apartness witnesses q_{ca} and q_{be} supplied by Definition 45.248 are converted into p_{ca} and p_{be}, so the composition theorem supplies the collected coefficients A, B, D, the certified composite indexed by those product witnesses, and the classifier iff for $U \circ T$. The certified-automorphism clauses are then read through Definition 45.536.

For inverse closure, use Definition 45.547. It states that the reverse composite $s_T(s_U(q))$ is both a left preimage and a right inverse for $U(T(x))$, while preserving the classifier iff. Finally, Definition 45.546 identifies this reverse composite with the inverse obtained directly from the collected coefficients, so the inverse operation is closed under the same ledger data. □

Per-object certificate: traditional target affine automorphism group interface; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations identity, composition, inverse, and classifier iff; ledger factor witnesses, product witnesses, inverse witnesses, and offsets; core theorem Definition 45.555; exactness predicate identity plus inverse-closed composition; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.556 (Field affine automorphisms form a GroupUp-compatible package)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist *under the hypotheses used by Definition 45.536 and Definition 45.541, let* AffAut_F *be the carrier of certified two-sided affine maps. A member is a tuple* (a, b, d, p_a, p_b) *with* $a, b, d : \mathsf{BHist}$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$*, displaying*

$$T_{a,b,d}(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a,x),b),d).$$

The affine classifier relates two members when their displayed maps are pointwise hsame*, exactly as in Definition 45.560. The identity member has coefficients* $(1,1,0)$ *with the unit nonzero witnesses. The product of* $U = (c, e, f, p_c, p_e)$ *after* $T = (a, b, d, p_a, p_b)$ *has collected coefficients*

$$A = \mathsf{mul}(c,a), \qquad B = \mathsf{mul}(b,e), \qquad D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c,d),e),f).$$

The inverse operation is represented by the explicit inverse formula certified in Definition 45.549. With these data, AffAut_F *satisfies the* $\mathsf{Group}^{\uparrow}$ *source fields: carrier inhabitance, closure under*

composition and inverse, associativity under the affine classifier, left and right identity laws, left and right inverse laws, and classifier congruence for both composition and inverse.

Proof. Carrier inhabitance is supplied by the identity coefficients. The unit nonzero witnesses and the identity clauses of Definition 45.555 show that this member acts as the identity map on every $x : \mathsf{BHist}$, and hence is the distinguished AffAut_F element.

Composition closure is the collected-coefficient construction of Definition 45.541. It supplies the product nonzero witnesses for A and B, the displayed affine normal form for $U(T(x))$, and the classifier iff clauses for the composite. The certified two-sided behavior of each displayed map is read through Definition 45.536.

Associativity is exactly the pointwise comparison from Definition 45.554. For three certified affine members, the two parenthesized compositions collect to classifier-equal displayed maps, so the pointwise affine classifier identifies the two products.

The inverse operation uses Definition 45.555 for the single-map identity and inverse clauses, together with Definition 45.549 for inverse closure. These clauses give left and right inverse laws under the affine classifier.

Classifier congruence for composition and inverse is exactly Definition 45.553. Hence the pointwise classifier supplies the operation fields required by the $\mathsf{Group}^{\uparrow}$ package. □

Per-object certificate: traditional target affine automorphism $\mathsf{Group}^{\uparrow}$ interface; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier AffAut_F; classifier pointwise hsame on displayed affine maps; stability obligations inhabitance, composition, inverse, associativity, identity, inverse laws, and classifier congruence; ledger identity witnesses, side-factor witnesses, product witnesses, reverse inverse formulas, and offsets; core theorem Definition 45.556; exactness predicate $\mathsf{Group}^{\uparrow}$ source-field satisfaction for certified affine automorphisms; standard bridge ordinary affine automorphism group law; status $\mathsf{ProofTarget}$.

Theorem 45.557 (Field affine product is independent of product witnesses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist *under the hypotheses of Definition 45.541, let* $T = (a, b, d, p_a, p_b)$ *and* $U = (c, e, f, p_c, p_e)$ *be certified affine members of* AffAut_F*. Set*

$$A = \mathsf{mul}(c, a), \qquad B = \mathsf{mul}(b, e), \qquad D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, d), e), f).$$

Let two choices of apartness-to-nonzero conversion data produce witnesses

$$p_{ca}, p'_{ca} : \mathsf{NonZero}_F(A), \qquad p_{be}, p'_{be} : \mathsf{NonZero}_F(B).$$

Form the two composite affine members with the same collected coefficients (A, B, D) *and with witness pairs* (p_{ca}, p_{be}) *and* (p'_{ca}, p'_{be})*. Then the affine classifier on* AffAut_F *relates these two composites: their displayed maps are pointwise* hsame*, and their reverse explicit inverse formulas are pointwise* hsame*.*

Lean4 ✓ `BEDC.Derived.FieldUp.field affine product witness independence`

Proof. The displayed composite maps have identical coefficients A and B and the same offset D. For each $x : \mathsf{BHist}$, both displays reduce to $\mathsf{add}(\mathsf{mul}(\mathsf{mul}(A, x), B), D)$, so pointwise classification is reflexive.

It remains to compare the proof-indexed inverse factors occurring in the reverse explicit inverse formulas. Apply Definition 45.202 to the reflexive coefficient comparison $A\mathsf{hsame}A$ with witnesses p_{ca} and p'_{ca}, giving $\mathsf{inv}(A, p_{ca})\mathsf{hsameinv}(A, p'_{ca})$. Apply the same theorem to $B\mathsf{hsame}B$ with witnesses p_{be} and p'_{be}, giving $\mathsf{inv}(B, p_{be})\mathsf{hsameinv}(B, p'_{be})$.

Multiplication congruence transports the first inverse comparison through the left affine factor, multiplication congruence transports the second comparison through the right affine factor, and addition congruence transports the shared offset term. Therefore the two reverse explicit inverse formulas are pointwise hsame, which is exactly the second field of the affine classifier. □

Per-object certificate: traditional target affine composition well-definedness; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier AffAut_F; classifier the affine classifier; stability obligation proof-index independence for product witnesses; ledger the two witness pairs (p_{ca}, p_{be}) and (p'_{ca}, p'_{be}); core theorem Definition 45.557; exactness predicate product operation descends to the affine classifier; standard bridge ordinary affine automorphism composition; status ProofTarget.

Capstone.

Theorem 45.558 (Field affine automorphism group certificate)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$ *candidate over* BHist*, let* $\mathsf{AffAut}_F(T)$ *mean that* T *carries displayed parameters* $a, b, d :$ BHist*, witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*, and the certified two-sided affine automorphism clauses for*

$$T(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d).$$

Equip the affine classifier with pointwise hsame *on all carried histories. Then the identity affine map, the collected-coefficient composition law, and the explicit inverse operation satisfy the* $\mathsf{Group}^{\uparrow}$ *stability fields: left and right identity, associativity of composition, inverse closure, left and right inverse laws, and classifier congruence. Moreover, for certified members* $T = (a, b, d, p_a, p_b)$ *and* $U = (c, e, f, p_c, p_e)$ *with fixed collected coefficients*

$$A = \mathsf{mul}(c, a), \qquad B = \mathsf{mul}(b, e), \qquad D = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(c, d), e), f),$$

any two product-witness choices p_{ca}, p_{be} *and* p'_{ca}, p'_{be} *produce composite affine members related by the affine pointwise classifier, including their reverse explicit inverse formulas. Equivalently, the displayed affine automorphism package determines the concrete* $\mathsf{Group}^{\uparrow}$ *certificate for* AffAut_F *over* BHist*.*

Proof. The identity and inverse data are exactly the package isolated in Definition 45.555. The identity affine map supplies both unit laws pointwise, and Definition 45.535 supplies the displayed inverse operation with its left and right inverse clauses.

Composition uses the collected coefficients from the same package, with the two-sided certification read through Definition 45.536.

For proof-relevant product witnesses, fix collected coefficients A, B, and D and compare the two composites built from witness pairs (p_{ca}, p_{be}) and (p'_{ca}, p'_{be}). Apply Definition 45.557: it gives pointwise hsame for the displayed composite maps and for their reverse explicit inverse formulas, which discharges the witness-choice descent field of the certificate.

The classifier is pointwise hsame, so classifier congruence for composition and inverse is obtained from Definition 45.553.

Associativity is the normal-form statement of Definition 45.554. It identifies the two parenthesized collected composites by the BEDC ring and field laws, and therefore gives the associativity field of the $\mathsf{Group}^{\uparrow}$ certificate.

The theorem BEDC.Derived.GroupUp.group_stability_certificate_fields is the packaging result consumed by Lean: once the identity, composition, associativity, inverse closure, inverse laws, and classifier congruence fields are supplied, it repacks them as the public $\mathsf{Group}^{\uparrow}$ stability certificate. The preceding paragraphs provide precisely those fields for AffAut_F. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldAffineAutomorphismGroupCertificate`

Theorem 45.559 (Field affine automorphism inverse uniqueness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist *satisfying Definition 45.558, let* $T : \mathsf{AffAut}_F$ *be a certified affine automorphism history. Let* $I : \mathsf{AffAut}_F$ *be the identity affine automorphism, let* $\circ_{\mathrm{aff}}$ *be the collected-coefficient affine composition operation, and let* $S : \mathsf{AffAut}_F$ *be the explicit inverse certificate for* T *supplied by Definition 45.555. If* $U : \mathsf{AffAut}_F$ *is any certified affine automorphism history satisfying both inverse equations*

$$(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} I$$

under the affine composition classifier, then U *and* S *are related by the affine pointwise classifier:*

$$U \equiv_{\mathrm{aff}} S.$$

Proof. Read Definition 45.558 as the concrete $\mathsf{Group}^{\uparrow}$ package on the carrier AffAut_F. Its operation is $\circ_{\mathrm{aff}}$, its identity is I, and its classifier is the affine pointwise classifier.

The explicit inverse package Definition 45.555 gives the two inverse equations for S against T. The hypotheses give the corresponding left and right inverse equations for U against the same element T in the same classifier.

Apply the checked group inverse uniqueness principle $\mathsf{BEDC.Derived.GroupUp.group_left_right_inverse_uniquer}$ to this affine $\mathsf{Group}^{\uparrow}$ package, with middle element T, left inverse candidate U, and right inverse candidate S. The resulting group-classifier relation is exactly $U \equiv_{\mathrm{aff}} S$, because the group certificate was built from the affine pointwise classifier. □

Definition 45.560 (Field affine automorphism package)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist*, a field affine certificate is data* (a, b, d, p_a, p_b) *with* $a, b, d : \mathsf{BHist}$*,* $p_a : \mathsf{NonZero}_F(a)$*, and* $p_b : \mathsf{NonZero}_F(b)$*. Its displayed map is*

$$T_{a,b,d}(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a, x), b), d).$$

For two displayed affine maps T *and* U*, the affine pointwise classifier is defined by*

$$T \equiv_{\mathrm{aff}} U \quad \Longleftrightarrow \quad \textit{every } x : \mathsf{BHist} \textit{ satisfies } T(x)\mathsf{hsame}U(x).$$

Theorem 45.561 (Field affine automorphism package laws)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist*, assume the explicit nonzero witness* $p_1 : \mathsf{NonZero}_F(\mathsf{one})$ *and the inverse-coefficient nonzero witnesses required by Definition 45.549. Then the affine certificates of Definition 45.560 satisfy the* $\mathsf{Group}^{\uparrow}$*-style law package under* $\equiv_{\mathrm{aff}}$*: the identity certificate is* $(\mathsf{one}, \mathsf{one}, \mathsf{zero}, p_1, p_1)$*, the coefficient-collected composition is the composite supplied by Definition 45.541, and the inverse of* (a, b, d, p_a, p_b) *is the normal form supplied by Definition 45.549. These operations are closed on certified affine automorphisms, composition is associative under the pointwise classifier, and every certificate satisfies the left and right identity laws and the left and right inverse laws under* $\equiv_{\mathrm{aff}}$*.*

Proof. The identity certificate uses the unit and zero laws already carried by the $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$ setup. For every $x : \mathsf{BHist}$, the displayed map with coefficients $(\mathsf{one}, \mathsf{one}, \mathsf{zero})$ reduces by the left and right multiplicative unit laws and the additive zero law to x, so it is the identity under $\equiv_{\mathrm{aff}}$.

Composition closure is exactly Definition 45.541. Given two certified affine automorphisms, that theorem collects the factors and offset into the displayed affine normal form and returns the product nonzero witnesses needed to certify the composite. The resulting certified map is also covered by Definition 45.536.

Associativity is checked pointwise. For three certificates, compare the two parenthesized triple composites by expanding both through the same displayed coefficient collection classifier from Definition 45.541; associativity of the $\mathsf{CommRing}^{\uparrow}$ multiplication and addition data identifies the collected left factor, right factor, and offset. Hence the two triple composites are related by $\equiv_{\mathrm{aff}}$.

Inverse closure is Definition 45.549: the explicit inverse map is classifier-equal to a certified affine normal form whose coefficients use the supplied inverse-coefficient witnesses. The two-sided inverse laws then follow from Definition 45.536 for a single certificate and from Definition 45.547 for a composite, which identifies the reverse explicit inverse with the collected composite inverse. Together these clauses give the stated $\mathsf{Group}^{\uparrow}$-style law package. □

Theorem 45.562 (Field affine automorphism sibling lattice)**.** *For the affine automorphism package of Definition 45.560, the* $\mathsf{Field}^{\uparrow}$ *exit reads the* $\mathsf{Rat}^{\uparrow}$ *concrete arithmetic surface through chapter 37 and the group-law consumer surface through chapter 41. The package has no independent affine automorphism route that bypasses those two sibling certificates: every accepted read of the affine carrier, pointwise classifier, coefficient ledger, nonzero ledger, inverse ledger, identity law, composition law, associativity law, and inverse law is routed through the displayed rational arithmetic source and the displayed group-law consumer.*

Proof. The carrier and classifier are the ones fixed in Definition 45.560: a certificate is the finite tuple (a, b, d, p_a, p_b) over BHist, and equality of displayed maps is the pointwise relation $\equiv_{\mathrm{aff}}$. The coefficient rows are not new ambient automorphism data; they are the rationally readable arithmetic rows of the $\mathsf{Field}^{\uparrow}$ candidate, so the concrete arithmetic exit is the $\mathsf{Rat}^{\uparrow}$ surface named by chapter 37.

The law package is supplied by Definition 45.561. Its closure, identity, associativity, and inverse clauses are precisely the group-law shape consumed through chapter 41. The certified affine composition row is Definition 45.541; the certified inverse row is Definition 45.549; and the pointwise classifier boundary is the displayed $\equiv_{\mathrm{aff}}$ classifier from the package definition.

Thus the package factors through the two sibling certificates before it reaches any downstream affine action. In ten fields, the public dependency record is: BEDC source $\mathsf{Field}^{\uparrow}$ over $\mathsf{Rat}^{\uparrow}$ plus $\mathsf{Group}^{\uparrow}$; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; coefficient ledger row; nonzero ledger row; inverse ledger row; composition ledger row; identity ledger row; sibling certificate pair chapter 37 and chapter 41; and downstream consumer affine normalizer action. □

Theorem 45.563 (Field threefold certified affine automorphisms compose)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with the operations, congruence laws, distributivity laws, indexed inverse laws, classifier transport of* $\mathsf{NonZero}_F$*, and product-apartness conversion maps used in Definition 45.541, let* $p_{a_1} : \mathsf{NonZero}_F(a_1)$*,* $p_{b_1} : \mathsf{NonZero}_F(b_1)$*,* $p_{a_2} : \mathsf{NonZero}_F(a_2)$*,* $p_{b_2} : \mathsf{NonZero}_F(b_2)$*,* $p_{a_3} : \mathsf{NonZero}_F(a_3)$*, and* $p_{b_3} : \mathsf{NonZero}_F(b_3)$ *be the six side-factor witnesses. Define the component maps*

$$T_1(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a_1, x), b_1), d_1), \qquad T_2(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a_2, x), b_2), d_2), \qquad T_3(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a_3, x), b_3),$$

First collect the factors for $T_2 \circ T_1$ *by*

$$A_{21} = \mathsf{mul}(a_2, a_1), \qquad B_{21} = \mathsf{mul}(b_1, b_2), \qquad D_{21} = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a_2, d_1), b_2), d_2).$$

Then collect the factors for $T_3 \circ T_2 \circ T_1$ by

$$A_{321} = \mathsf{mul}(a_3, A_{21}), \qquad B_{321} = \mathsf{mul}(B_{21}, b_3), \qquad D_{321} = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(a_3, D_{21}), b_3), d_3).$$

Let $p_{A_{21}}, p_{B_{21}}, p_{A_{321}}, p_{B_{321}}$ be the collected nonzero witnesses obtained from the six side-factor witnesses by the product-apartness theorem Definition 45.271 and the available product-apartness conversion maps. Define the collected normal form

$$C_{321}(x) = \mathsf{add}(\mathsf{mul}(\mathsf{mul}(A_{321}, x), B_{321}), D_{321}).$$

For each $q : \mathsf{BHist}$ define the component inverse maps

$$s_1(q) = \mathsf{mul}(\mathsf{inv}(a_1, p_{a_1}), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(d_1)), \mathsf{inv}(b_1, p_{b_1}))),$$

$$s_2(q) = \mathsf{mul}(\mathsf{inv}(a_2, p_{a_2}), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(d_2)), \mathsf{inv}(b_2, p_{b_2}))),$$

and

$$s_3(q) = \mathsf{mul}(\mathsf{inv}(a_3, p_{a_3}), \mathsf{mul}(\mathsf{add}(q, \mathsf{neg}(d_3)), \mathsf{inv}(b_3, p_{b_3}))).$$

Then every $x, y, q : \mathsf{BHist}$ satisfies the following clauses.

1.

$$T_3(T_2(T_1(x)))\mathsf{hsame}C_{321}(x).$$

2. The collected normal form C_{321} is a certified two-sided affine automorphism by Definition 45.536 with witnesses $p_{A_{321}}$ and $p_{B_{321}}$.

3.

$$C_{321}(x)\mathsf{hsame}C_{321}(y) \iff x\mathsf{hsame}y.$$

4.

$$C_{321}(s_1(s_2(s_3(q))))\mathsf{hsame}q, \qquad s_1(s_2(s_3(C_{321}(x))))\mathsf{hsame}x.$$

Proof. First apply Definition 45.541 to T_1 and T_2. This gives the collected map with coefficients A_{21}, B_{21}, D_{21} and the intermediate certificate supplied by Definition 45.536.

Next apply the same composition theorem to the certified map for $T_2 \circ T_1$ and the component T_3. The product-apartness theorem Definition 45.271 supplies the collected nonzero witnesses in the two factor positions, so the resulting coefficients are $A_{321}, B_{321}, D_{321}$.

The displayed normal-form equality follows by transitivity from the two binary normal-form equalities. The certificate for C_{321} is then exactly Definition 45.536 instantiated with $p_{A_{321}}$ and $p_{B_{321}}$, and its classifier clause gives the displayed iff.

For the inverse, use Definition 45.545 first for $T_2 \circ T_1$, then for T_3 composed with that intermediate certified map. The explicit inverse order is therefore the reverse component order $q \mapsto s_1(s_2(s_3(q)))$.

Finally, Definition 45.546 identifies that reverse component inverse with the collected-coefficient inverse, while Definition 45.547 supplies the two-sided inverse equations for the displayed composite normal form. □

Per-object certificate: traditional target affine automorphism threefold composition; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ over BHist; carrier BHist; classifier hsame; stability obligations binary affine composition, certified affine automorphism exactness, explicit composite inverse, reverse inverse coherence, and product apartness for collected factors; ledger six side-factor nonzero witnesses, two intermediate product witnesses, two collected product witnesses, and three offsets; core theorem Definition 45.563; exactness predicate collected threefold normal form with reverse component inverse; standard bridge deferred; status ProofTarget.

Theorem 45.564 (Field affine automorphism inverse is involutive)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.558, let* $T = (a, b, d, p_a, p_b)$ *be a certified affine automorphism and let* T_{aff}^{-1} *be the explicit inverse certificate supplied by Definition 45.549. Then applying the same inverse construction again returns the original affine certificate under the affine pointwise classifier:*

$$(T_{\mathrm{aff}}^{-1})_{\mathrm{aff}}^{-1} \equiv_{\mathrm{aff}} T.$$

Proof. The inverse certificate T_{aff}^{-1} is closed among certified affine automorphisms by Definition 45.549. The identity and inverse operations of the affine package are the ones recorded in Definition 45.555, so both T and T_{aff}^{-1} inhabit the same $\mathsf{Group}^{\uparrow}$ carrier of certified affine automorphism histories.

The two inverse equations for T are interpreted through the affine pointwise classifier. Operation congruence from Definition 45.553 transports those equations across classifier-equivalent representatives, so the candidate obtained by inverting T_{aff}^{-1} is a left inverse of T_{aff}^{-1} in the same group package.

Now apply the $\mathsf{Group}^{\uparrow}$ inverse involutivity law from Definition 41.57; equivalently, use the left-right inverse uniqueness principle of Definition 41.59 to identify the second inverse with the original certificate. Re-reading that identity through the affine classifier gives the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine explicit inverse double cancel at`

Per-object template: traditional target affine automorphism group inverse involutivity; BEDC source $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$, affine package, and $\mathsf{Group}^{\uparrow}$ inverse laws; carrier certified affine automorphism histories; classifier $\equiv_{\mathrm{aff}}$; stability obligation inverse closure and inverse uniqueness; ledger nonzero coefficient witnesses and inverse-certificate provenance; core theorem Definition 45.564; exactness predicate double-inverse pointwise classifier; standard bridge ordinary affine automorphism group; status $\mathsf{ProofTarget}$.

Theorem 45.565 (Field affine automorphism cancellation)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$ *candidate over* BHist *satisfying Definition 45.558, let* S, T, U *be certified affine automorphism histories in* AffAut_F*, and let* $\circ_{\mathrm{aff}}$ *be the coefficient-collected composition operation from the affine package. Then the affine classifier is cancellative on both sides: if*

$$(S \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} (U \circ_{\mathrm{aff}} T),$$

then $S \equiv_{\mathrm{aff}} U$*; and if*

$$(T \circ_{\mathrm{aff}} S) \equiv_{\mathrm{aff}} (T \circ_{\mathrm{aff}} U),$$

then $S \equiv_{\mathrm{aff}} U$*.*

Proof. Instantiate the affine automorphism group law package supplied by Definition 45.558. Its carrier is the class of certified affine automorphism histories, its operation is $\circ_{\mathrm{aff}}$, and its classifier is the affine pointwise relation $\equiv_{\mathrm{aff}}$.

For the common right factor, read $(S \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} (U \circ_{\mathrm{aff}} T)$ as the group classifier equality of the two products with the same right component. Apply Definition 41.116 in this instantiated $\mathsf{Group}^{\uparrow}$ package. The conclusion is the group classifier relation between S and U, which is exactly $S \equiv_{\mathrm{aff}} U$ under the pointwise classifier interpretation.

For the common left factor, read $(T \circ_{\mathrm{aff}} S) \equiv_{\mathrm{aff}} (T \circ_{\mathrm{aff}} U)$ as the group classifier equality of the two products with the same left component. Apply Definition 41.66 in the same instantiated package. Translating the resulting group classifier back through the affine pointwise classifier gives $S \equiv_{\mathrm{aff}} U$. □

Per-object template: traditional target affine automorphism cancellativity; BEDC source $\mathsf{Field}^{\uparrow}+\mathsf{CommRing}^{\uparrow}$, the affine package, and $\mathsf{Group}^{\uparrow}$ cancellation; carrier certified affine automorphism histories; classifier $\equiv_{\text{aff}}$; stability associativity, identity, inverse, and operation congruence; ledger nonzero coefficient witnesses and composition provenance; core theorem Definition 45.565; exactness predicate left and right cancellation under the affine classifier; standard bridge ordinary affine automorphism group; status ProofTarget.

Theorem 45.566 (Field affine automorphism cancellation exactness)**.** *Under the same hypotheses and notation as Definition 45.565, for certified affine automorphism histories S, T, U in AffAut_F, cancellation by the common affine factor is exact on both sides:*

$$(S \circ_{\text{aff}} T) \equiv_{\text{aff}} (U \circ_{\text{aff}} T) \Longleftrightarrow S \equiv_{\text{aff}} U,$$

and

$$(T \circ_{\text{aff}} S) \equiv_{\text{aff}} (T \circ_{\text{aff}} U) \Longleftrightarrow S \equiv_{\text{aff}} U.$$

Proof. For the forward directions, apply Definition 45.565. Its first clause cancels the common right factor T, and its second clause cancels the common left factor T. In both cases the resulting classifier is precisely $S \equiv_{\text{aff}} U$.

For the reverse directions, assume $S \equiv_{\text{aff}} U$. The common factor T is reflexively related to itself by the affine pointwise classifier. Apply Definition 45.553 to the classifier between S and U together with reflexivity of T. With the common factor placed on the right this gives $(S \circ_{\text{aff}} T) \equiv_{\text{aff}} (U \circ_{\text{aff}} T)$; with the common factor placed on the left it gives $(T \circ_{\text{aff}} S) \equiv_{\text{aff}} (T \circ_{\text{aff}} U)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field affine automorphism cancellation exact from empty unit group laws iff`

Per-object template: traditional target affine automorphism cancellative action exactness; BEDC source $\mathsf{Field}^{\uparrow} + \mathsf{CommRing}^{\uparrow}$, the affine package, and $\mathsf{Group}^{\uparrow}$ cancellation; carrier certified affine automorphism histories; classifier affine pointwise equivalence; core theorem Definition 45.566; status ProofTarget.

Theorem 45.567 (Field singleton nonzero append context cancellation)**.** *For histories L, R, Q : BHist, if L and R lie in the singleton carrier A_{e}, then surrounding Q by those append contexts preserves and reflects the singleton nonzero endpoint predicate:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow (\mathsf{FieldSingletonNonZero}(\mathsf{append}(L, \mathsf{append}(Q, R))) \Longleftrightarrow \mathsf{Field}$$

Proof. The two context carrier witnesses identify L and R with e. Thus $\mathsf{append}(L, \mathsf{append}(Q, R))$ is history-same to Q by the left and right empty append laws. In the forward direction, empty-result inversion for append extracts the singleton carrier witness for Q, and the history comparison transports the displayed $\mathsf{E0}(\mathsf{e})$ endpoint witness back to Q.

Conversely, combine the singleton carrier witness for Q with the two context carrier witnesses to rebuild the nested append carrier by empty-result inversion. The same empty-context comparison transports the nonzero endpoint witness from Q to the nested append endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNonZero append context cancel iff`

Theorem 45.568 (RatUp FieldUp denominator context-pair action certificate)**.** *For the concrete tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$ with*

$$C_D^u = \mathsf{RatDenomUnitCarrier}, \qquad R_D^u = \mathsf{RatDenomUnitClassifier},$$

$$\mu_D^u = \mathsf{append}, \qquad e_D^u = \mathsf{e}, \qquad N_D^u = \mathsf{RatHistoryCarrier},$$

and for the context-pair structure of Definition 45.479 and the contextual action Γ *of Definition 45.454, the denominator context-pair action certificate for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *consists of the following concrete rows:*

1. $\mathsf{PairClass}_D^u$ *is reflexive, symmetric, and transitive on carried pairs, and it transports the carried pair predicate* $C_D^u(l) \wedge C_D^u(r)$.
2. *The product* $\star$ *of Definition 45.479 is closed on carried pairs and congruent for* $\mathsf{PairClass}_D^u$*, with exact* $\mathsf{PairSupp}_D^u$ *readback as in Definition 45.480.*
3. *The unit-pair action, product-pair composition law, and product strict-support readback are exactly the laws of Definition 45.482.*
4. *The three-layer empty-endpoint contextual action has the normalized single-action form*

$$\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))) \sim_{R_D^u} \Gamma_{\mathsf{e},\mathsf{e},\mu_D^u(l_2,\mu_D^u(l_1,l_0)),\mu_D^u(\mu_D^u(r_0,r_1),r_2)}(h)$$

   and the seven-term strict-support readback of Definition 45.469.
5. *These rows, together with Definition 45.474, form the concrete denominator context-pair action certificate for* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$.

Equivalently, the certificate records exactly the pair-action interface of Definition 45.480, Definition 45.482, and the contextual row assembly of Definition 45.472 over the same BHist carrier-classifier-operation tuple $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier})$.

Proof. First build the classifier rows. The concrete denominator classifier R_D^u is an equivalence on carried histories and transports the carried history predicate by the denominator row package. Applying those rows to the left and right coordinates gives reflexivity, symmetry, transitivity, and carrier transport for $\mathsf{PairClass}_D^u$ on carried pairs.

Next establish product stability. Closure of $\star$ is componentwise closure of μ_D^u. Congruence follows by applying the denominator append congruence rows to the left components and to the right components, preserving the product orders $\mu_D^u(l', l)$ and $\mu_D^u(r, r')$ from Definition 45.479. The exact $\mathsf{PairSupp}_D^u$ readback is Definition 45.480.

Now apply Definition 45.482. It supplies the unit-pair action, the product-pair composition law, the product strict-support readback, and the two-sided unit and associativity clauses for the same product $\star$ and componentwise classifier.

Apply Definition 45.460 to the same carried left context, acted history, and carried right context whenever the pair classifier transports an acting pair. This gives contextual action stability under $\mathsf{PairClass}_D^u$ and keeps the external endpoint contexts empty.

For the iterated action row, apply Definition 45.469. It gives the carried three-layer action, its normalized single-action classifier comparison, and the seven-term strict-support readback for the same Γ, μ_D^u, e_D^u, and N_D^u.

Finally apply Definition 45.474. Its row closure, exactness, unit-composition, and support clauses occupy the one-layer and two-layer rows of the same concrete denominator interface. The pair classifier rows, product congruence, pair-action laws, and threefold normalization therefore assemble as the stated FieldUp denominator context-pair action certificate over BHist. □

Theorem 45.569 (RatUp FieldUp denominator action envelope closure)**.** *For* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$*, the concrete RatUp denominator data*

$$C_D^* = \mathsf{RatDenomUnitCarrier}, \qquad R_D^* = \mathsf{RatDenomUnitClassifier},$$

$$\mu_D^* = \mathsf{append}, \qquad e_D^* = \mathsf{e}, \qquad N_D^* = \mathsf{RatHistoryCarrier},$$

together with the empty-unit tuple

$$(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u) = (\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier})$$

supply one denominator action envelope. The envelope consists of multiplicative obligation coverage, the contextual multiplicative boundary, the unit-composition row package, and the context-pair action certificate over the same BHist histories, append operation, empty endpoint, and denominator carrier-classifier pair.

Proof. First use Definition 45.439 and Definition 45.438. They supply the multiplicative obligation coverage and the contextual multiplicative boundary for the concrete RatUp denominator data $(C_D^*, R_D^*, \mu_D^*, e_D^*, N_D^*)$.

Next use Definition 45.474. It records the same denominator carrier, classifier, append operation, empty unit, strict-support predicate, and contextual action Γ, and it adds the empty-action unit row, unchanged support row, context-multiplication composition row, and five-term support row.

Finally use Definition 45.568. It packages the carried context-pair predicate, pair unit, product $\star$, componentwise classifier, support readbacks, and star unit and associativity laws for the identical empty-unit tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$.

The four cited certificates are all stated over concrete BHist carriers and the concrete operation append. Hence the envelope is a single assembly of denominator action data for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$, not a carrier or classifier transport from another instance. □

Per-object template: traditional target FieldUp(RatUp) denominator action envelope; BEDC source BHist, $\mathsf{RatHistoryCarrier}$, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, e, and Γ; carrier $\mathsf{RatDenomUnitCarrier}$; classifier $\mathsf{RatDenomUnitClassifier}$; stability obligations multiplicative coverage, contextual multiplicative boundary, unit-composition rows, and context-pair action laws; ledger denominator histories, empty endpoint, carried context pairs, and strict-support rows; core theorem Definition 45.569; exactness predicate denominator classifier readback and strict-support readback; standard bridge $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator action envelope; status $\mathsf{ProofTarget}$.

Per-object template: traditional target denominator context-pair action certificate for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, $\mathsf{RatHistoryCarrier}$, append, Cont, e, and Γ; carrier carried denominator context pairs; classifier $\mathsf{PairClass}_D^u$; stability obligations pair classifier equivalence, carrier transport, $\star$ congruence, unit action, product composition, and threefold normalization; ledger left and right carried denominator contexts plus empty external endpoints; core theorem Definition 45.568; exactness predicate support readback and row closure; standard bridge FieldUp denominator action; status $\mathsf{ProofTarget}$.

Theorem 45.570 (RatUp FieldUp denominator context-pair normalized row exhaustion)**.** *For the concrete denominator context-pair monoid of Definition 45.479 and the concrete contextual action Γ of Definition 45.454, every denominator action row used by* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *is supplied by the concrete BHist data*

$$\mathsf{RatDenomUnitCarrier}, \quad \mathsf{RatDenomUnitClassifier}, \quad \mathsf{append}, \quad \mathsf{e}, \quad \mathsf{RatHistoryCarrier},$$

the empty external contexts, and the pair product $\star$. More precisely, the action interface is exhausted by the row closure package of Definition 45.472, the unit-composition laws of Definition 45.474, the normalized contextual action package of Definition 45.475, the context-pair action laws of Definition 45.482, and the context-pair action certificate of Definition 45.568.

Proof. First invoke Definition 45.472 and Definition 45.474. These two rows provide the carried contextual row package, the unit action, and the composition clauses over (RatDenomUnitCarrier, RatDenomUnitClas

Next invoke Definition 45.475. This identifies the normalized contextual rows with the same concrete denominator tuple and the same empty external endpoints, so no abstract carrier or classifier transport is introduced.

Then invoke Definition 45.482. Its clauses give the unit pair, composition by $\star$, support readback, and associativity for the concrete pair product of Definition 45.479.

Finally invoke Definition 45.568. It identifies the preceding row, unit-composition, normalized, and pair-action clauses as the concrete denominator action certificate consumed by $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$. Hence the displayed BHist objects, empty external contexts, and $\star$ account for the whole denominator action interface. □

Per-object template: traditional target FieldUp(RatUp) denominator context-pair normalized row exhaustion; BEDC source BHist, RatHistoryCarrier, RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, and Γ; carrier denominator context pairs of Definition 45.479; classifier componentwise R^u_D; stability obligations row closure, unit composition, normalized exactness, pair-action unit, pair-action composition, support readback, and $\star$ associativity; ledger normalized contextual rows and pair-action rows with empty external endpoints; core theorem Definition 45.570; exactness predicate denominator action row coverage by the concrete BHist data; standard bridge FieldUp denominator action interface; status ProofTarget.

Theorem 45.571 (Rat denominator context-pair product strict support)**.** *For carried denominator context pairs (l, r) and (l', r'), strict support of their product is exactly the disjunction of the four component supports:*

$$N^u_D(\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))) \iff N^u_D(l') \vee N^u_D(l) \vee N^u_D(r) \vee N^u_D(r').$$

Proof. Apply bilateral strict-support exactness to the left product $\mu^u_D(l', l)$ and the carried right product $\mu^u_D(r, r')$. The remaining right-product support term is then opened by denominator product support exactness for r and r'. Reassociating the resulting disjunction yields the four displayed component supports, and the reverse implication closes the same two product-support equivalences in the opposite order. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair product strict support iff`

Theorem 45.572 (Rat denominator context-pair product strict support transport)**.** *If (l, r) and (s, t) are componentwise R^u_D-classified, and (l', r') and (s', t') are componentwise R^u_D-classified, then strict support is invariant across the two product endpoints:*

$$N^u_D(\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))) \iff N^u_D(\mu^u_D(\mu^u_D(s', s), \mu^u_D(t, t'))).$$

Proof. The component classifiers expose $\mathsf{hsame}(l, s)$, $\mathsf{hsame}(r, t)$, $\mathsf{hsame}(l', s')$, and $\mathsf{hsame}(r', t')$. Continuation congruence composes these through the two product layers, giving an hsame witness between the two displayed product endpoints. Rat denominator support transports across that endpoint witness in both directions. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomContextPair product strict support transport`

Theorem 45.573 (RatUp FieldUp denominator context-pair product certificate row)**.** *For the concrete denominator context-pair structure of Definition 45.479, let (l, r) and (l', r') be carried denominator context pairs. Let (m, s) and (m', s') be carried comparison pairs with* $\mathsf{PairClass}^u_D((l, r), (m, s))$

and $\mathsf{PairClass}^u_D((l', r'), (m', s'))$. *Then the concrete product* $(l, r) \star (l', r')$ *preserves carried membership, respects the componentwise* R^u_D *classifiers against* $(m, s) \star (m', s')$, *and has bundled strict support exactly*

$$N^u_D(\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))) \Longleftrightarrow \mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r').$$

The transported product endpoint has the same bundled strict support:

$$N^u_D(\mu^u_D(\mu^u_D(l', l), \mu^u_D(r, r'))) \Longleftrightarrow N^u_D(\mu^u_D(\mu^u_D(m', m), \mu^u_D(s, s'))).$$

Reversing the two acting pairs preserves the component classifiers,

$$R^u_D(\mu^u_D(l', l), \mu^u_D(l, l')) \quad \textit{and} \quad R^u_D(\mu^u_D(r, r'), \mu^u_D(r', r)),$$

and the reversed bundled support has the same readback:

$$N^u_D(\mu^u_D(\mu^u_D(l, l'), \mu^u_D(r', r))) \Longleftrightarrow \mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r').$$

Proof. Apply Definition 45.481. Its carrier clause gives carried membership for the two components of $(l, r) \star (l', r')$, and its classifier clause gives componentwise R^u_D respect against $(m, s) \star (m', s')$.

The strict support of the concrete product is the support clause of the same operation row. Equivalently, apply Definition 45.571 and regroup the four component supports into $\mathsf{PairSupp}^u_D(l, r) \vee \mathsf{PairSupp}^u_D(l', r')$.

The comparison endpoint has the same strict support by Definition 45.572, using the four component classifiers supplied by the two pair-class hypotheses.

For the reversed product, apply Definition 45.484. It gives the displayed left and right component classifiers. Applying the same commutative-support theorem with the acting pairs exchanged gives the reversed support readback, and commutativity of disjunction identifies it with the displayed pair-support disjunction. □

45.23 Five-factor inverse reversal

Theorem 45.574 (Field inverse five-factor product reversal from apartness-indexed inverses)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *with multiplication associativity, multiplication congruence, left and right identity laws, proof-indexed inverse laws, product nonzero exclusion, and conversion maps from* $\mathsf{ApartZero}_F$ *to* $\mathsf{NonZero}_F$ *for all needed factors and left-nested prefixes*

$$\alpha_a : \mathsf{ApartZero}_F(a) \to \mathsf{NonZero}_F(a), \quad \alpha_b : \mathsf{ApartZero}_F(b) \to \mathsf{NonZero}_F(b),$$

$$\alpha_c : \mathsf{ApartZero}_F(c) \to \mathsf{NonZero}_F(c), \quad \alpha_d : \mathsf{ApartZero}_F(d) \to \mathsf{NonZero}_F(d),$$

$$\alpha_e : \mathsf{ApartZero}_F(e) \to \mathsf{NonZero}_F(e),$$

$$\alpha_{ab} : \mathsf{ApartZero}_F(\mathsf{mul}(a, b)) \to \mathsf{NonZero}_F(\mathsf{mul}(a, b)),$$

$$\alpha_{abc} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(a, b), c)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, b), c)),$$

$$\alpha_{abcd} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d)) \to \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d)),$$

where $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot)$, *any witness*

$$p_{abcde} : \mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e))$$

induces witnesses p_a, p_b, p_c, p_d, *and* p_e *such that*

$$\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e), p_{abcde})\mathsf{hsamemul}(\mathsf{inv}(e, p_e), \mathsf{mul}(\mathsf{inv}(d, p_d), \mathsf{mul}(\mathsf{inv}(c, p_c), \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a$$

Proof. Apply Definition 45.303 to p_{abcde} for the binary product $(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),e)$. This gives $q_{abcd} : \mathsf{ApartZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d))$ and $q_e : \mathsf{ApartZero}_F(e)$; set $p_{abcd} := \alpha_{abcd}(q_{abcd})$ and $p_e := \alpha_e(q_e)$.

Next apply Definition 45.213 to the same outer binary product with witnesses p_{abcde}, p_{abcd}, and p_e. It identifies the inverse of the fivefold product with $\mathsf{mul}(\mathsf{inv}(e,p_e),\mathsf{inv}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),p_{abcd}))$.

Apply Definition 45.221 to the left fourfold prefix, using p_{abcd} and the conversion maps α_{ab}, α_{abc}, α_a, α_b, α_c, and α_d. This produces the factor witnesses p_a, p_b, p_c, and p_d and the reversed fourfold inverse spine.

Multiplication congruence transports the fourfold equality into the right factor of the outer equality. Associativity and transitivity of hsame then rebracket the product into the displayed e,d,c,b,a reversed inverse spine, with all witnesses induced from the original ledger entry p_{abcde}. □

Per-object certificate: Traditional target: finite product inverse reversal. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist plus zero-product exclusion. Carrier choice: BHist. Classifier choice: hsame. Stability obligations: product apartness extraction, inverse reversal, associativity, and multiplication congruence. Ledger: p_{abcde} plus induced prefix and factor witnesses. Core theorem list: Theorem 45.574. Exactness predicate: the displayed reversed equality. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.575 (RatUp FieldUp transported exit row contextual separation package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Under the hypotheses of Definition 45.509, suppose also that* $N^u_D(h)$ *holds. Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit row contains a single contextual separation package with:*

1. *the exit-row closure recorded by Definition 45.525;*
2. *the direct endpoint separation* $x\mathsf{hsamee} \to \bot$*;*
3. *for every* $r : \mathsf{BHist}$*, both contextual rejection maps* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x)) \to \bot$ *and* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r)) \to \bot$*.*

Proof. First apply Definition 45.525 with the same source tuple, denominator tuple, contextual action, endpoint, and strict-support hypothesis. This supplies the concrete exit-row closure consumed by the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate boundary.

Next apply Definition 45.520 at the same endpoint. Its final clause is the direct endpoint separation $x\mathsf{hsamee} \to \bot$.

Finally fix an arbitrary context history $r : \mathsf{BHist}$ and apply Definition 45.516. The two clauses obtained there are exactly the rejection maps for $\mathsf{append}(r,x)$ and $\mathsf{append}(x,r)$. Collecting these clauses gives the displayed contextual separation package. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported exit row contextual separation package`

Theorem 45.576 (RatUp FieldUp transported exit row singleton-classifier noncollapse)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Under the hypotheses of Definition 45.509, suppose also that* $N^u_D(h)$ *holds. Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit row contains both:*

1. *the exit-row closure recorded by Definition 45.525;*
2. *for all* $L, R : \mathsf{BHist}$*, the singleton-classifier obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,x),\mathsf{append}(R,x)) \to \bot.$$

Proof. First apply Definition 45.525 with the same concrete source tuple, denominator tuple, contextual action, endpoint, and strict-support witness. This determines the source-anchored exit-row closure package for the displayed endpoint x.

Next choose arbitrary $L, R : \mathsf{BHist}$. Apply Definition 45.517 to the same endpoint x and the same witness $N^u_D(h)$. The theorem gives the map from $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x))$ to contradiction.

Combining the exit-row closure package with these uniformly indexed classifier obstructions gives the stated concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit row. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup transported exit row singleton classifier noncollapse`

Per-object template: traditional target contextual classifier non-collapse for constructive $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, contextual action Γ, and strict support N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ plus the singleton classifier obstruction; stability transported support neutrality and classifier separation; ledger denominator contextual-pair action; core theorem Definition 45.576 with Definition 45.525 and Definition 45.517; exactness contextual classifier non-collapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ contextual exit separation; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, contextual action Γ, and strict support N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability obligations transported support neutrality and append-context singleton exclusion; ledger denominator contextual-pair action; core theorem Definition 45.575; exactness predicate direct and contextual non-collapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.577 (RatUp FieldUp transported exit row full noncollapse package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, let* $x := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))$*. Under the hypotheses of Definition 45.509, suppose also that* $N^u_D(h)$ *holds. Then the transported strict-support endpoint* x *carries one package containing:*

1. *the exit-row closure of Definition 45.525;*
2. $C^u_D(x)$*,* $N^u_D(x)$*,* $\mathsf{RatHistoryCarrier}(x)$*,* $\mathsf{fieldSingletonEmptyCarrier}(x) \to \bot$*, and* $x\mathsf{hsamee} \to \bot$*;*
3. *for every* $r : \mathsf{BHist}$*, both contextual carrier exclusions on* $\mathsf{append}(r, x)$ *and* $\mathsf{append}(x, r)$*;*
4. *for every* $r : \mathsf{BHist}$*, both inverse-domain witnesses* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x))$ *and* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x, r))$*;*
5. *for all* $L, R : \mathsf{BHist}$*, the singleton-classifier obstruction on* $\mathsf{append}(L, x)$ *and* $\mathsf{append}(R, x)$*.*

Proof. Apply Definition 45.525 with the same source tuple, denominator tuple, contextual action, endpoint, and witness. This supplies the exit-row closure clause.

Apply Definition 45.520 at the same endpoint. The resulting separation package gives $C^u_D(x)$, $N^u_D(x)$, $\mathsf{RatHistoryCarrier}(x)$, the singleton carrier rejection, and the direct empty-history separation.

Apply Definition 45.516 for an arbitrary context history $r : \mathsf{BHist}$. Its two conclusions are the carrier exclusions for $\mathsf{append}(r, x)$ and $\mathsf{append}(x, r)$.

Apply Definition 45.521 with the same endpoint and support witness. This gives the two contextual inverse-domain witnesses for every $r : \mathsf{BHist}$.

Finally apply Definition 45.576. For arbitrary $L, R : \mathsf{BHist}$ it supplies the classifier obstruction on the two endpoints. Collecting these clauses gives the full noncollapse package. □

Per-object template: traditional target full exit-row noncollapse package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; source $\mathsf{RatUpFieldSource}_F$, action Γ, strict support N^u_D, all obstructions read at x; status $\mathsf{ProofTarget}$.

Theorem 45.578 (RatUp FieldUp normalized contextual separation package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l)$*,* $C^u_D(r)$*,* $C^u_D(l')$*, and* $C^u_D(r')$*. Suppose also that* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*,* $q'\mathsf{hsamee}$*,* $\neg N^u_D(l)$*,* $\neg N^u_D(r)$*,* $\neg N^u_D(l')$*,* $\neg N^u_D(r')$*, and* $N^u_D(h)$*. Let* $x := \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$ *and let* $y := \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)$*. Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate simultaneously contains:*

1. *the assembled certificate boundary of Definition 45.489;*
2. *the carried nested endpoint* $C^u_D(x)$*;*
3. *the normalized classifier row* $R^u_D(x, y)$*;*
4. *the strict-support equivalence* $N^u_D(x) \iff N^u_D(h)$*;*
5. *the exit-row closure recorded by Definition 45.525;*
6. *the direct endpoint separation* $x\mathsf{hsamee} \to \bot$*;*
7. *for every context history* $r : \mathsf{BHist}$*, both singleton-context rejection maps* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \bot$ *and* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$*.*

Proof. First apply Definition 45.489. This fixes the single concrete source tuple $\mathsf{RatUpFieldSource}_F$ and the certificate boundary in which the carrier, classifier, denominator source, contextual source, context-pair source, and affine source are jointly registered.

Next apply Definition 45.478 to the displayed carried endpoints, empty numerator endpoints, and empty-support denominator contexts. Its nested row gives $C^u_D(x)$, its collected row gives $R^u_D(x, y)$, and its support row gives $N^u_D(x) \iff N^u_D(h)$.

The assumed witness $N^u_D(h)$ and the displayed equivalence recover the strict-support hypothesis at the transported endpoint x. Apply Definition 45.575 with that endpoint and the same contextual action. It supplies the nondegenerate exit-row closure, the direct empty separation $x\mathsf{hsamee} \to \bot$, and the two singleton-context rejection maps for each $r : \mathsf{BHist}$.

Collecting the boundary, normalized row, support transport, and contextual separation clauses gives the stated concrete normalized $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate. □

Per-object template: traditional target concrete normalized $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit separation; BEDC source $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ plus denominator classifier R^u_D; stability obligations total-normalization support collapse and transported contextual separation; ledger contextual-pair denominator action; core theorem Definition 45.578; exactness predicate normalized and direct/contextual non-collapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.579 (RatUp FieldUp normalized semantic exit certificate)**.** *Under the hypotheses and notation of Definition 45.578, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate contains both the normalized contextual separation rows for* x, y *and the concrete nondegenerate semantic exit package supplied by Definition 45.513 and Definition 45.512. More precisely, the same certificate simultaneously contains:*

1. *the assembled certificate boundary of Definition 45.489;*
2. *the normalized classifier row* $R^u_D(x, y)$*;*

3. *the strict-support equivalence* $N^u_D(x) \iff N^u_D(h)$*;*
4. *the direct endpoint rejection map* $x\mathsf{hsamee} \to \bot$ *and, for every context history* $r : \mathsf{BHist}$*, the singleton-context rejection maps* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$ *and* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{ap}$ $\bot$*;*
5. *source anchoring of the concrete certificate rows over* $\mathsf{RatHistoryCarrier}$ *and* $\mathsf{RatHistoryClassifier}$*;*
6. *the nondegenerate witness* $d_1 := \mathsf{E1}(\mathsf{e})$ *with* $\mathsf{RatHistoryCarrier}(d_1)$ *and* $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$*.*

Thus the normalized contextual row and the concrete nondegenerate semantic exit closure are registered in one BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate.*

Proof. Apply Definition 45.578 to the displayed carried endpoints, empty numerator endpoints, empty-support denominator contexts, and strict-support witness. This gives the assembled certificate boundary, the normalized classifier row $R^u_D(x, y)$, the strict-support equivalence $N^u_D(x) \iff N^u_D(h)$, and the endpoint and singleton-context rejection maps for x.

Next apply Definition 45.513. It repacks the semantic certificate rows, source anchoring rows, support-sensitive coverage, endpoint facts, and shared concrete row language as the concrete BHist-anchored exit closure over $\mathsf{RatUpFieldSource}_F$.

Finally apply Definition 45.512. It supplies the same source anchoring over $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$ and provides the witness $d_1 := \mathsf{E1}(\mathsf{e})$ with $\mathsf{RatHistoryCarrier}(d_1)$ and $\mathsf{fieldSingletonEmptyCarr}$ $\bot$.

Collecting these three cited sites gives a single concrete exit certificate: the normalized contextual separation rows for x, y occupy the same $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ certificate boundary as the nondegenerate semantic exit closure and its explicit rational-history witness. □

Theorem 45.580 (RatUp FieldUp threefold support singleton-classifier noncollapse)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $x := \mu^u_D(\mu^u_D(h, k), \ell)$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all* $L, R : \mathsf{BHist}$*,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot.$$

Proof. Apply Definition 45.527 with the same concrete denominator tuple, carried histories, and left-nested endpoint. The threefold support witness gives $\mathsf{RatHistoryCarrier}(x)$.

Package this carrier witness on both sides, together with reflexivity of hsame, as $\mathsf{RatHistoryClassifier}(x, x)$.

Now fix arbitrary $L, R : \mathsf{BHist}$. Apply Definition 45.104. It sends any displayed singleton-classifier witness to contradiction. □

Per-object template: traditional target concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ threefold denominator exit support; BEDC source BHist with $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; context $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations empty-unit continuation closure and Rat carrier readback; ledger left-nested denominator product endpoint $\mu^u_D(\mu^u_D(h, k), \ell)$; core theorems Definition 45.527, Definition 45.530, Definition 45.531, and Definition 45.532; exactness predicate threefold strict-support non-singleton boundary with contextual append separation and inverse-domain nonzero package; standard bridge finite denominator support into the concrete FieldUp exit; status $\mathsf{ProofTarget}$.

Per-object template: traditional target classifier-level noncollapse for the threefold $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator exit; BEDC source BHist with $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; context $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier obstruction against $\mathsf{fieldSingletonEmptyClassifier}$; stability obligations empty-unit continuation closure and Rat carrier readback; ledger left-nested denominator

product endpoint $\mu^u_D(\mu^u_D(h,k),\ell)$; core theorem Definition 45.580 with Definition 45.527, Definition 45.530, and Definition 45.531; exactness classifier noncollapse; standard bridge deferred; status ProofTarget.

Per-object template: traditional target contextual classifier non-collapse for constructive $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, contextual action Γ, and strict support N^u_D; carrier RatHistoryCarrier; classifier RatHistoryClassifier plus the singleton classifier obstruction; stability transported support neutrality and classifier separation; ledger denominator contextual-pair action; core theorem Definition 45.576 with Definition 45.525 and Definition 45.517; exactness contextual classifier non-collapse; standard bridge deferred; status ProofTarget.

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ contextual exit separation; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, contextual action Γ, and strict support N^u_D; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability obligations transported support neutrality and append-context singleton exclusion; ledger denominator contextual-pair action; core theorem Definition 45.575; exactness predicate direct and contextual non-collapse; standard bridge deferred; status ProofTarget.

Theorem 45.581 (RatUp FieldUp normalized contextual singleton-classifier separation)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(l)$*,* $C^u_D(r)$*,* $C^u_D(l')$*, and* $C^u_D(r')$*. Suppose also that* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*,* $q'\mathsf{hsamee}$*,* $\neg N^u_D(l)$*,* $\neg N^u_D(r)$*,* $\neg N^u_D(l')$*,* $\neg N^u_D(r')$*, and* $N^u_D(h)$*. Let* $x := \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$ *and let* $y := \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)$*. Then for all* $L, R : \mathsf{BHist}$*,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,x), \mathsf{append}(R,y)) \to \bot.$$

Proof. Apply Definition 45.578 to the same source tuple, denominator tuple, contextual action, carried histories, empty endpoint hypotheses, empty-support context hypotheses, and strict-support witness. The package supplies the normalized classifier row $R^u_D(x,y)$ and the strict-support transport $N^u_D(x) \Longleftrightarrow N^u_D(h)$.

Use the displayed witness $N^u_D(h)$ and the transport equivalence to recover $N^u_D(x)$. The denominator boundary identifies this strict support with the genuine Rat carrier, so x is carried by RatHistoryCarrier.

Unfold the normalized classifier row through the same denominator boundary. Together with the Rat carrier recovered at x, the row packages the concrete classifier $\mathsf{RatHistoryClassifier}(x,y)$ for the normalized pair.

Now fix arbitrary histories $L, R : \mathsf{BHist}$. Apply Definition 45.104 to the displayed $\mathsf{RatHistoryClassifier}(x,y)$. It rejects every proposed singleton empty classifier on $\mathsf{append}(L,x)$ and $\mathsf{append}(R,y)$, which is the asserted noncollapse. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction normalized pair singleton classifier exclusion`

Per-object template: traditional target normalized contextual classifier noncollapse for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, Γ, R^u_D, and N^u_D; carrier RatHistoryCarrier; classifier RatHistoryClassifier plus singleton-empty classifier; stability obligations normalized contextual classifier row and strict-support transport; ledger contextual-pair denominator action; core theorem this block; exactness predicate singleton-classifier noncollapse for the normalized pair; standard bridge deferred; status ProofTarget.

Theorem 45.582 (RatUp FieldUp five-factor nonzero factor ledger)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *candidate over* BHist *satisfying the hypotheses of Definition 45.312, assume the*

apartness-to-nonzero bridge family

$$\alpha_h : (h\mathsf{hsame0} \to \bot) \to \mathsf{NonZero}_F(h)$$

for each $h : \mathsf{BHist}$. *Then every* $a, b, c, d, e : \mathsf{BHist}$ *satisfies*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e)) \Longleftrightarrow \mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b) \wedge \mathsf{NonZero}_F(c) \wedge \mathsf{NonZero}_F(d) \wedge \mathsf{NonZero}$$

Proof. Apply Definition 45.312 to the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ candidate, the given bridge family, and the five histories a, b, c, d, e. The resulting equivalence is exactly the displayed RatUp FieldUp ledger row. □

Per-object template: traditional target RatUp FieldUp five-factor nonzero factor ledger; BEDC source $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ over BHist with the certified five-factor product nonzero equivalence; carrier BHist; classifier hsame with $\mathsf{NonZero}_F$; stability obligations general five-factor exactness and apartness-to-nonzero transport; ledger factor witnesses p_a, p_b, p_c, p_d, p_e, prefix witness p_{abcd}, and fivefold product witness p_{abcde}; core theorem Definition 45.582; exactness predicate nonzero of the left-nested fivefold product equivalent to the five factor nonzero witnesses; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.583 (RatUp FieldUp normalized contextual full noncollapse package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$, *the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, *and the contextual action* Γ *of Definition 45.454, suppose* $C^u_D(h)$, $C^u_D(k)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, *and* $C^u_D(r')$. *Suppose also that* $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, $q'\mathsf{hsamee}$, $\neg N^u_D(l)$, $\neg N^u_D(r)$, $\neg N^u_D(l')$, $\neg N^u_D(r')$, *and* $N^u_D(h)$. *Let* $x := \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$ *and let* $y := \Gamma_{p',q',\mu^u_D(l',l),\mu^u_D(r,r')}(h)$. *Then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate simultaneously contains:*

1. *the assembled certificate boundary of Definition 45.489;*
2. *the carried nested endpoint* $C^u_D(x)$*;*
3. *the normalized classifier row* $R^u_D(x, y)$*;*
4. *the strict-support equivalence* $N^u_D(x) \Longleftrightarrow N^u_D(h)$*;*
5. *the exit-row closure recorded by Definition 45.525;*
6. *the direct endpoint separation* $x\mathsf{hsamee} \to \bot$*;*
7. *for every context history* $r : \mathsf{BHist}$, *both singleton-context rejection maps* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$ *and* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$*;*
8. *for all histories* $L, R : \mathsf{BHist}$, $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot$.

Proof. First apply Definition 45.578 to the same source tuple, denominator tuple, contextual action, carried histories, empty endpoint hypotheses, empty-support context hypotheses, and strict-support witness. This supplies the assembled certificate boundary, the carried nested endpoint, the normalized classifier row, the strict-support equivalence, the exit-row closure, the direct endpoint separation, and the singleton-context carrier rejections.

Next apply Definition 45.581 under the same hypotheses and with the same definitions of x and y. It supplies the uniform singleton-classifier obstruction for arbitrary $L, R : \mathsf{BHist}$.

Collect the seven certificate clauses from the normalized contextual separation package together with the singleton-classifier obstruction. The resulting row is one concrete BHist package for the normalized $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit criterion. □

Per-object template: traditional target normalized contextual full noncollapse for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, Γ, R^u_D, and N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ plus singleton-empty classifier; stability obligations normalized contextual separation and singleton-classifier obstruction; ledger contextual-pair denominator action; core theorem Definition 45.583; exactness predicate carrier and classifier noncollapse for the normalized contextual pair; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.584 (RatUp FieldUp proof-target exit package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *proof target is exhausted by concrete* BHist *rows. Its exit package is the constructive composition of the paper certificate assembly of Definition 45.494, the source anchoring of Definition 45.501, the concrete schema instance of Definition 45.502, the concrete exit certificate package of Definition 45.523, the transported exit-row contextual separation of Definition 45.575, and the normalized contextual separation of Definition 45.578, together with the normalized semantic exit certificate of Definition 45.579 and the normalized singleton-classifier separation of Definition 45.581, as well as the five-factor product equation exactness of Definition 45.296 backed by Definition 45.574. Thus the displayed* BHist *source tuple carries the exit package for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$*: it contains the normalized semantic nondegenerate witness* $d_1 := \mathsf{E1}(\mathsf{e})$*, the normalized classifier row, the all-context singleton-classifier rejection for* $\mathsf{append}(L, x)$ *and* $\mathsf{append}(R, y)$*, and the five-factor right and left product equation exactness rows, while no exit row remains as a schematic* mul*,* add*, or* neg *operator parameter.*

Proof. Start with Definition 45.494. It supplies the paper-side carrier, source, classifier, stability, ledger, and exactness rows for the concrete source tuple $\mathsf{RatUpFieldSource}_F$ over BHist.

Next apply Definition 45.501. It identifies every row consumed by the concrete certificate with the same source tuple: the rational history carrier and classifier, the denominator tuple, the contextual action Γ, the context-pair monoid, and the affine coordinate.

The schema and exit boundary are then fixed by Definition 45.502 and Definition 45.523. Together these theorem sites repack the source-anchored rows as the public $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate instance and its concrete exit certificate.

Finally use Definition 45.575 and Definition 45.578. The first theorem supplies the transported noncollapse and contextual rejection rows, while the second records the normalized row package in the same concrete source. Combining these clauses leaves only rows carried by $\mathsf{RatUpFieldSource}_F$.

Apply Definition 45.579. It adds the nondegenerate RatUp witness $d_1 := \mathsf{E1}(\mathsf{e})$, its source anchoring in $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$, and the same normalized contextual rows to the concrete exit certificate.

Apply Definition 45.581 for arbitrary $L, R : \mathsf{BHist}$. The resulting row rejects every $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{a}$ claim for the normalized pair.

Use Definition 45.296 for the displayed factor product P_5. It supplies the right and left solver equivalences for P_5, while Definition 45.574 supplies the induced factor witnesses used by the reversed inverse spine. Both theorem sites are stated over $\mathsf{Field}^{\uparrow}$ on BHist, so their rows occupy the same $\mathsf{RatUpFieldSource}_F$ boundary.

These semantic, classifier-separation, and five-factor exactness rows live in the same concrete $\mathsf{RatUpFieldSource}_F$ boundary, so the exit package has no schematic mul, add, or neg operator coordinate. □

Definition 45.585 (RatUp FieldUp ten-field exit instance)**.** *The ten-field exit instance for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *is the following concrete paper certificate object.*

1. *Traditional target:* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *as a fully derived field instance.*

2. *BEDC source:* $\mathsf{RatUpFieldSource}_F$, BHist, $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, *the denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, *the contextual action* Γ, *the context-pair monoid of Definition 45.479, and the affine rows of* AffAut_F.
3. *Carrier:* $\mathsf{RatHistoryCarrier}$ *on the concrete history carrier* BHist.
4. *Classifier:* $\mathsf{RatHistoryClassifier}$ *together with the denominator classifier row* R^u_D *and the singleton-empty classifier obstruction rows for the normalized contextual endpoints.*
5. *Stability obligations: Definition 45.494, Definition 45.501, Definition 45.502, Definition 45.523, the transported and normalized contextual separation packages, and the normalized semantic and classifier exit certificates.*
6. *Ledger: rational history rows, denominator rows, contextual-pair rows, affine rows, and the nondegenerate witness row* $d_1 := \mathsf{E1}(\mathsf{e})$.
7. *Core theorem list: chiefly Definition 45.584, with the normalized semantic classifier exit certificate and the bracketing-independent threefold contextual boundary as the concrete support theorem sites, plus Definition 45.600 as the concrete cross-endpoint singleton-classifier exclusion.*
8. *Exactness predicate: concrete row exhaustion for the* $\mathsf{RatUpFieldSource}_F$ *package and singleton noncollapse for the normalized contextual classifier rows, including the crossed endpoint singleton-classifier noncollapse rows.*
9. *Standard bridge: deferred.*
10. *Status:* $\mathsf{ProofTarget}$.

Theorem 45.586 (RatUp FieldUp ten-field exit certificate)**.** *The instance of Definition 45.585, together with Definition 45.584, Definition 45.590, and Definition 45.599, supplies the concrete ten-field paper certificate for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *with no schematic* mul, add, *or* neg *operator parameter.*

Proof. Apply Definition 45.584. It identifies the paper certificate, source anchoring, schema instance, concrete exit package, and contextual separation rows as clauses of the single $\mathsf{RatUpFieldSource}_F$ boundary.

Next apply Definition 45.590. It adds the normalized semantic witness rows, the R^u_D classifier row, and the singleton-empty classifier obstruction for every contextual pair.

Finally apply Definition 45.599. It supplies the common threefold denominator support for both bracketings, so the certificate has contextual-pair and denominator support independent of the chosen bracketing. These three theorem sites fill every field listed in Definition 45.585, completing the concrete paper certificate. □

Definition 45.587 (RatUp FieldUp concrete operation table)**.** *The concrete operation table for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *is the* BHist*-anchored row package whose carrier row is* $\mathsf{RatHistoryCarrier}$ *and whose classifier row is* $\mathsf{RatHistoryClassifier}$. *Its base coordinate is the* $\mathsf{CommRing}^{\uparrow}$ *row inside* $\mathsf{RatUpFieldSource}_F$. *Its denominator coordinate is*

$$(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier}).$$

The contextual coordinate is the action Γ *together with the context-pair monoid of Definition 45.479. The apartness coordinate is* $\mathsf{FieldApartZero}$, *with the inverse rows supplied by apartness, and the affine coordinate is* AffAut_F.

The table uses the following ten-field per-object template.

1. *Traditional target: the concrete constructive field* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$.
2. *BEDC source:* $\mathsf{RatUpFieldSource}_F$ *over* BHist, *with* $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, *denominator, contextual, context-pair, apartness, inverse, and affine rows.*
3. *Carrier:* $\mathsf{RatHistoryCarrier}$.
4. *Classifier:* $\mathsf{RatHistoryClassifier}$ *together with* $\mathsf{RatDenomUnitClassifier}$ *for denominator histories.*
5. *Stability obligations: carrier and classifier anchoring, the ring-base rows, denominator multiplication and unit rows, contextual-action rows, context-pair support rows, affine closure rows, inverse rows over* $\mathsf{FieldApartZero}$, *and exit noncollapse rows.*
6. *Ledger: rational histories, denominator unit histories, continuation witnesses, context pairs, apart-zero inverse witnesses, affine coefficients, and singleton obstruction witnesses.*
7. *Core theorem list: Definition 45.501, Definition 45.502, Definition 45.586, Definition 45.589, and Definition 45.600.*
8. *Exactness predicate: row coverage for the same* $\mathsf{RatUpFieldSource}_F$ *table, including denominator support, contextual transport, product-classifier transport, inverse-domain transport, affine transport, and singleton noncollapse.*
9. *Standard bridge: the public* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate consumes this table as the concrete operation schema.*
10. *Status:* $\mathsf{ProofTarget}$.

Theorem 45.588 (RatUp FieldUp concrete operation table laws)**.** *The table of Definition 45.587 satisfies the carrier and classifier rows, the* $\mathsf{CommRing}^{\uparrow}$ *ring-base rows, the denominator multiplication and unit rows, the contextual-action rows, the context-pair support rows, the affine rows, the inverse and nonzero rows, and the exit noncollapse rows required by the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *schema.*

Proof. First apply Definition 45.501. It places the carrier, classifier, denominator tuple, contextual action, context-pair monoid, affine coordinate, and $\mathsf{CommRing}^{\uparrow}$ base coordinate inside the single concrete source tuple $\mathsf{RatUpFieldSource}_F$.

Next apply Definition 45.502. The schema instance identifies those rows with the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ operation schema, so the table has no free schematic mul, add, or neg coordinate outside the displayed BHist rows.

The exit side is supplied by Definition 45.586. It fills the per-object certificate fields and contributes the normalized semantic and classifier noncollapse rows required by the table.

For multiplication through nonzero contexts, use Definition 45.589. It gives the one-sided cancellation row, the product-classifier exactness row, and the nonzero transport row for multiplication by certified nonzero histories.

Finally apply Definition 45.600. It supplies the cross-endpoint singleton-classifier exclusions for the common threefold denominator support. Together with the previous rows, this gives the carrier/classifier, base, denominator, contextual, context-pair, affine, inverse, nonzero, and exit noncollapse laws listed in the statement. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupConcreteOperationTableLaws`

Theorem 45.589 (RatUp FieldUp multiplication exactness and nonzero exit package)**.** *For the concrete BEDC history-level* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *row over* BHist, *let* $a, b, x, y : \mathsf{BHist}$ *and let* p_a :

$\mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$ *be certified nonzero witnesses. The multiplication package contains the one-sided cancellation classifier rows*

$$\mathsf{mul}(x,b)\mathsf{hsamemul}(y,b) \iff x\mathsf{hsame}y$$

and

$$\mathsf{mul}(a,x)\mathsf{hsamemul}(a,y) \iff x\mathsf{hsame}y.$$

The two-sided product map $x \mapsto \mathsf{mul}(\mathsf{mul}(a,x),b)$ *is exact as a classifier row and as a nonzero-transport row:*

$$(\mathsf{mul}(\mathsf{mul}(a,x),b)\mathsf{hsamemul}(\mathsf{mul}(a,y),b)) \iff x\mathsf{hsame}y$$

and

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a,x),b)) \iff \mathsf{NonZero}_F(x).$$

Together these four equivalences form one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *multiplication package for certified nonzero factors.*

Proof. Apply Definition 45.244 to the same history-level $\mathsf{Field}^{\uparrow}$ candidate. With witnesses p_b and p_a, it supplies the right- and left-factor classifier exactness rows over BHist.

Apply Definition 45.245 to the same history-level $\mathsf{Field}^{\uparrow}$ candidate and to the certified outer witnesses p_a and p_b. This supplies the classifier exactness row for the map $x \mapsto \mathsf{mul}(\mathsf{mul}(a,x),b)$.

Apply Definition 45.309 with the same witnesses. This supplies the nonzero-transport row for the identical two-sided product endpoint.

These rows use the same BHist carrier, the same classifier hsame, the same multiplication endpoints, and the same ledger witnesses p_a, p_b. They therefore assemble as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication certificate clause. □

Per-object template: traditional target multiplication exactness inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, $\mathsf{Field}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, and $\mathsf{RatUpFieldSource}_F$; carrier BHist; classifier hsame; stability obligations multiplication congruence, one-sided cancellation exactness, inverse cancellation, zero absorption, and nonzero transport; ledger the one-sided witnesses, the outer witnesses p_a, p_b, and product endpoint $\mathsf{mul}(\mathsf{mul}(a,x),b)$; core theorem this block; exactness predicate the one-sided cancellation iff rows and the displayed paired equivalences; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.590 (RatUp FieldUp normalized semantic classifier exit certificate)**.** *Under the hypotheses and notation of Definition 45.578, the same concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *certificate contains the normalized semantic exit package of Definition 45.579 and, for all* $L, R : \mathsf{BHist}$*, the classifier obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,x),\mathsf{append}(R,y)) \to \bot$$

supplied by Definition 45.581. Equivalently, the normalized semantic rows for the carried pair x, y *and the singleton-classifier noncollapse rows for every contextual pair* L, R *occupy one* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate over* $\mathsf{RatUpFieldSource}_F$*.*

Proof. First invoke Definition 45.579 under the displayed carried endpoint hypotheses, empty endpoint hypotheses, empty-support context hypotheses, and strict-support witness. It gives the assembled certificate boundary, the normalized row $R^u_D(x,y)$, the strict-support equivalence $N^u_D(x) \iff N^u_D(h)$, the contextual rejection maps, source anchoring of the concrete rows, and the nondegenerate witness $d_1 := \mathsf{E1}(\mathsf{e})$ with $\mathsf{RatHistoryCarrier}(d_1)$ and $\mathsf{fieldSingletonEmptyCarrier}(d_1) \to \bot$.

Next invoke Definition 45.581 with the same carried endpoints, empty endpoint hypotheses, empty-support context hypotheses, contextual action Γ, and strict-support witness. This yields, for arbitrary $L, R : \mathsf{BHist}$, the obstruction

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot.$$

These two applications share the same concrete source tuple, denominator tuple, contextual action, normalized endpoints, and support witness. Therefore the semantic exit rows and the singleton-classifier obstruction are not separate targets: they are clauses of one normalized concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate. □

Per-object template: traditional target concrete normalized $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ classifier exit; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, append, Γ, R^u_D, and N^u_D; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ plus $\mathsf{fieldSingletonEmptyClassifier}$ obstruction; stability obligations normalized semantic exit and singleton-classifier noncollapse; ledger contextual-pair denominator action and nondegenerate witness; core theorem this block; exactness predicate semantic exit plus classifier noncollapse for the normalized pair; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.591 (RatUp FieldUp singleton-empty operation context normalization)**.** *For histories L, R, h, k, out : BHist with $\mathsf{fieldSingletonEmptyCarrier}(L)$ and $\mathsf{fieldSingletonEmptyCarrier}(R)$, the lower-case singleton-empty $\mathsf{Field}^{\uparrow}$ operation slots are contextual BHist endpoints:*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyMul}(h, k)), \mathsf{append}(R, out)) \iff \mathsf{fieldSingletonEmptyCarri}$$

The same contextual normalization holds for the terminal unit endpoint:

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyOne}), \mathsf{append}(R, out)) \iff \mathsf{fieldSingletonEmptyCarrier}(ou$$

Finally, for every proof witness p : $\mathsf{fieldSingletonEmptyNonZero}(h)$, the inverse endpoint satisfies

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyInv}(h, p)), \mathsf{append}(R, out)) \iff \mathsf{fieldSingletonEmptyCarrie}$$

Proof. Unfold the three lower-case operation endpoints. In the singleton-empty $\mathsf{Field}^{\uparrow}$ construction, $\mathsf{fieldSingletonEmptyMul}(h, k)$, $\mathsf{fieldSingletonEmptyOne}$, and $\mathsf{fieldSingletonEmptyInv}(h, p)$ are all the endpoint e.

For the forward direction in each displayed equivalence, use the two context carrier witnesses for L and R. Empty-result inversion for append removes the left endpoint $\mathsf{append}(L, \mathsf{e})$ and then removes the right context from $\mathsf{append}(R, out)$, leaving precisely $\mathsf{fieldSingletonEmptyCarrier}(out)$.

For the reverse direction, begin with the displayed output carrier. Append-empty reconstruction places the output behind the carried context R, and the left context witness reconstructs the empty operation endpoint behind L. Classifier transitivity through e gives the required singleton-empty classifier in each of the multiplication, unit, and proof-indexed inverse cases. □

Per-object template: traditional target concrete terminal $\mathsf{Field}^{\uparrow}$ operation endpoints; BEDC source BHist, append, and the lower-case singleton-empty $\mathsf{Field}^{\uparrow}$ operations; carrier $\mathsf{fieldSingletonEmptyCarrier}$; classifier $\mathsf{fieldSingletonEmptyClassifier}$; stability obligations append empty split and constant operation endpoints; ledger hidden left/right singleton contexts and the proof-indexed inverse witness; core theorem this block; exactness predicate the three displayed equivalences; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.592 (Field singleton classifier operation context endpoint package)**.** *For histories* $L, R, h, k, out : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, the canonical singleton field operation endpoints are contextual classifier endpoints:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{FieldSingletonMul}(h, k)), \mathsf{append}(R, out)) \iff \mathsf{FieldSingletonCarrier}(out).$$

The same endpoint exactness holds for $\mathsf{FieldSingletonOne}$ *and, for each witness* $p : \mathsf{FieldSingletonNonZero}(h)$*, for* $\mathsf{FieldSingletonInv}(h, p)$*.*

Proof. Each displayed operation endpoint is definitionally e in the singleton field schema. For the forward direction, the classifier supplies the carrier witness for $\mathsf{append}(R, out)$; empty-result inversion for append removes the carried context R and leaves precisely the output carrier.

For the reverse direction, combine the carrier witnesses for L and R with the empty operation endpoint and the displayed output carrier. Append-empty reconstruction gives carrier witnesses for both classified endpoints, and transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonClassifier operation context endpoint package`

Theorem 45.593 (Field singleton add-neg-zero operation context endpoint package)**.** *For histories* $L, R, h, k, out : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, the singleton additive operation endpoints are contextual classifier endpoints:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(h, k)), \mathsf{append}(R, out)) \iff \mathsf{FieldSingletonCarrier}(out).$$

The same endpoint exactness holds for $\mathsf{FieldSingletonNeg}(h)$ *and* $\mathsf{FieldSingletonZero}$*.*

Proof. Each additive singleton operation endpoint is definitionally e. Forward classification exposes the right appended endpoint carrier and append-empty inversion removes R. Conversely, the carriers for L, R, the empty operation endpoint, and out rebuild the two appended carriers; transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddNegZero operation context endpoint package`

Theorem 45.594 (Field singleton add-neg-zero context continuation result carrier)**.** *For histories* $L, R, a, b, c, out : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, if*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingletonNeg}(c), R), out)$$

holds, then $\mathsf{FieldSingletonCarrier}(out)$*.*

Proof. The singleton additive and negation endpoints are definitionally e. The carrier witnesses for L and R reduce both continuation endpoints to the empty-empty continuation, so continuation determinacy identifies out with e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddNegZero context continuation result carrier`

Theorem 45.595 (Field singleton add-neg-zero context continuation result iff)**.** *For histories* $L, R, a, b, c, out : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, the add-neg-zero contextual continuation exists exactly for singleton field carrier results:*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingletonNeg}(c), R), out) \iff \mathsf{FieldSingletonCarrier}(out).$$

Proof. The forward direction is the add-neg-zero contextual carrier closure. For the reverse direction, the singleton additive and negation endpoints unfold to e, and the carrier witnesses for L, R, and out reduce the claim to the right-unit continuation at e, transported along the supplied carrier witness for out. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddNegZero context continuation result iff`

Theorem 45.596 (Field singleton add-neg-zero context continuation classifier)**.** *For histories* $L, R, a, b, c, d, out, out' : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, if*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonAdd}(a, b)), \mathsf{append}(\mathsf{FieldSingletonNeg}(c), R), out)$$

and

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonZero}), \mathsf{append}(\mathsf{FieldSingletonAdd}(c, d), R), out')$$

hold, then $\mathsf{FieldSingletonClassifier}(out, out')$*.*

Proof. The three displayed singleton operations are definitionally e. The carrier witnesses for L and R therefore normalize each continuation endpoint pair to the empty-empty continuation. Continuation determinacy gives $out = \mathsf{e}$ and $out' = \mathsf{e}$, and the two resulting carrier witnesses classify the endpoints by transitivity through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonAddNegZero context continuation classifier`

Theorem 45.597 (Field singleton neg context continuation classifier)**.** *For histories* $L, R, L', R', a, a', out, out' :$ BHist *with* $\mathsf{FieldSingletonCarrier}(L)$*,* $\mathsf{FieldSingletonCarrier}(R)$*,* $\mathsf{FieldSingletonCarrier}(L')$*, and* $\mathsf{FieldSingletonCarrier}(R')$ *if*

$$\mathsf{Cont}(\mathsf{append}(L, \mathsf{FieldSingletonNeg}(a)), R, out)$$

and

$$\mathsf{Cont}(\mathsf{append}(L', \mathsf{FieldSingletonNeg}(a')), R', out')$$

hold, then $\mathsf{FieldSingletonClassifier}(out, out')$*.*

Proof. The singleton negation endpoint is definitionally e. Each left context therefore remains an empty carrier after appending the negation endpoint. With the matching right context carrier, singleton continuation closure makes both outputs empty-carrier histories. The classifier follows by transitivity through e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonNeg context continuation classifier`

Theorem 45.598 (RatUp FieldUp right-nested threefold contextual boundary)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the right-nested* $\mathsf{Field}^\uparrow(\mathsf{RatUp})$ *denominator endpoint carries the following contextual boundary:*

1. *for every context history* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y)) \to \bot$*;*
2. *for every context history* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r)) \to \bot$*;*
3. *for every context history* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, y))$*;*
4. *for every context history* $r : \mathsf{BHist}$*,* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(y, r))$*;*

5. *for all* $L, R : \mathsf{BHist}$,

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, y)) \to \bot.$$

Proof. Apply Definition 45.528 to the same concrete denominator tuple, carried histories, and support witness. The right-nested endpoint data recovers $\mathsf{RatHistoryCarrier}(y)$ and the singleton rejection map $\mathsf{fieldSingletonEmptyCarrier}(y) \to \bot$.

Fix a context history $r : \mathsf{BHist}$. Apply Definition 45.101 with context r and Rat carrier endpoint y to reject $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y))$. Apply Definition 45.102 with the same two histories in the opposite append order to reject $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r))$.

For the right-append inverse-domain witness, unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, y))$ as rejection of the singleton classifier against e. Any such classifier witness contains the source-carrier component $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y))$, contradicting the right-append rejection.

For the left-append inverse-domain witness, unfold $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(y, r))$ as rejection of the singleton classifier against e. Its source-carrier component is $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r))$ contradicting the left-append rejection.

Finally package $\mathsf{RatHistoryClassifier}(y, y)$ from the recovered Rat carrier witness on both sides and reflexivity of hsame. For arbitrary $L, R : \mathsf{BHist}$, apply Definition 45.104. It rejects $\mathsf{fieldSingletonEmptyClassifier}($

□

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup right nested contextual boundary`

Theorem 45.599 (RatUp FieldUp bracketing-independent threefold contextual boundary)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *let* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$. *If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the two bracketings of the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *denominator endpoint carry the following common contextual boundary:*

1. $C^u_D(x)$, $C^u_D(y)$, *and* $R^u_D(x, y)$;
2. *for every context history* $r : \mathsf{BHist}$, *the four contextual carrier exclusions:*
   (a) $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$;
   (b) $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$;
   (c) $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y)) \to \bot$;
   (d) $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r)) \to \bot$.
3. *for every context history* $r : \mathsf{BHist}$, *the four inverse-domain witnesses:*
   (a) $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x))$;
   (b) $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x, r))$;
   (c) $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, y))$;
   (d) $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(y, r))$.
4. *for all* $L, R : \mathsf{BHist}$, *the two contextual singleton-classifier exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, y)) \to \bot.$$

Proof. First apply Definition 45.527 to the carried histories h, k, ℓ. This supplies the left-nested carrier row $C^u_D(x)$, the strict-support comparison for x, the Rat carrier witness for x under $\mathsf{ThreeSupp}^u_D(h, k, \ell)$, and the direct singleton-carrier rejection at x.

Apply Definition 45.528 to the same carried histories. Its endpoint data supplies $C^u_D(y)$, $R^u_D(x, y)$, the strict-support comparison for y, the Rat carrier witness for y under the same support hypothesis, and the direct singleton-carrier rejection at y. The displayed comparison is the denominator associativity row already used in that support exactness proof.

For the left-nested endpoint, use Definition 45.530 with the same support witness. It gives, for each context history r, the two contextual singleton-carrier exclusions around x.

Then apply Definition 45.531. This packages the two corresponding inverse-domain witnesses for $\mathsf{append}(r, x)$ and $\mathsf{append}(x, r)$.

For the right-nested endpoint, apply Definition 45.598. Its five clauses give the contextual singleton-carrier exclusions around y, the two inverse-domain witnesses for the right-nested bracketing, and the contextual singleton-classifier exclusion for arbitrary left and right contexts around y.

It remains only to record the classifier exclusion around x. The left-nested contextual separation package already contains the arbitrary $L, R : \mathsf{BHist}$ singleton-classifier rejection, so the two endpoint classifier exclusions stand side by side with the carrier and inverse-domain rows from the preceding paragraphs. □

Theorem 45.600 (RatUp FieldUp bracketing-independent cross-endpoint classifier boundary)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then for all $L, R : \mathsf{BHist}$ the crossed contextual singleton-classifier exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot$$

hold.

Proof. Apply the checked FieldUp cross-endpoint classifier boundary to the same carried histories h, k, ℓ. In the concrete denominator tuple its carrier predicate is $\mathsf{RatDenomUnitCarrier}$, its product is append, and the support input is exactly the disjunction encoded by $\mathsf{ThreeSupp}^u_D(h, k, \ell)$.

The endpoint definitions identify x with $\mathsf{append}(\mathsf{append}(h, k), \ell)$ and y with $\mathsf{append}(h, \mathsf{append}(k, \ell))$ up to hsame. For fixed $L, R : \mathsf{BHist}$, the checked boundary returns the conjunction of the two crossed singleton-classifier rejections, which are precisely the two displayed exclusions. □

↪ variant `BEDC.Derived.FieldUp.ratup fieldup bracketing independent cross endpoint classifier boundary`

Theorem 45.601 (RatUp FieldUp ten-field cross-endpoint exit certificate)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the concrete ten-field exit instance for $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ contains the certificate rows of Definition 45.586 and, for all $L, R : \mathsf{BHist}$, the crossed contextual exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot.$$

These rows form a single BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate row, not an abstract operator schema.*

Proof. First invoke Definition 45.586. It supplies the concrete ten-field certificate rows listed in Definition 45.585 for the source tuple $\mathsf{RatUpFieldSource}_F$.

Next invoke Definition 45.600 with the same concrete denominator tuple, histories h, k, ℓ, endpoint definitions x and y, and the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This gives the two crossed contextual singleton-classifier exclusions for arbitrary histories $L, R : \mathsf{BHist}$.

The certificate and the crossed exclusions are carried by the same BHist source, the same denominator tuple, and the same contextual append rows. Thus the strongest concrete noncollapse row is recorded inside the ten-field $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit instance as a concrete row rather than an abstract operator schema. □

Theorem 45.602 (RatUp FieldUp contextual cross-endpoint nondegenerate exit package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and the contextual action* Γ *of Definition 45.454, let* $h, k, \ell : \mathsf{BHist}$ *be histories with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*. Define the crossed denominator endpoints* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. For arbitrary contexts* $L, R : \mathsf{BHist}$ *and factors* $a, b, c, d, e : \mathsf{BHist}$*, put* $P_5 := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e)$*. Assume the hypotheses of Definition 45.601 at* h, k, ℓ, L, R*, the hypotheses of Definition 45.583 for the same concrete source tuple, denominator tuple, and action* Γ*, and the field hypotheses of Definition 45.296. Assume also a witness* $p_{abcde} : \mathsf{NonZero}_F(P_5)$ *and the apartness-to-nonzero maps required by Definition 45.582. Then the concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package simultaneously contains:*

1. *the ten-field certificate rows of Definition 45.601;*
2. *the two crossed singleton-classifier exclusions* $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot$ *and* $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot$*;*
3. *the normalized contextual full noncollapse package of Definition 45.583;*
4. *the five individual nonzero factor witnesses* $p_a : \mathsf{NonZero}_F(a)$*,* $p_b : \mathsf{NonZero}_F(b)$*,* $p_c : \mathsf{NonZero}_F(c)$*,* $p_d : \mathsf{NonZero}_F(d)$*, and* $p_e : \mathsf{NonZero}_F(e)$ *extracted from* p_{abcde}*;*
5. *for every* $u, v : \mathsf{BHist}$*, the right and left five-factor equation exactness rows*

$$\mathsf{mul}(u, P_5)\mathsf{hsame}v \iff u\mathsf{hsamemul}(v, R_5)$$

and

$$\mathsf{mul}(P_5, u)\mathsf{hsame}v \iff u\mathsf{hsamemul}(R_5, v),$$

where $R_5 := \mathsf{mul}(\mathsf{inv}(e, p_e), \mathsf{mul}(\mathsf{inv}(d, p_d), \mathsf{mul}(\mathsf{inv}(c, p_c), \mathsf{mul}(\mathsf{inv}(b, p_b), \mathsf{inv}(a, p_a)))))$*.*

Proof. Apply Definition 45.601 with the displayed denominator tuple, histories h, k, ℓ, endpoints x, y, contexts L, R, and the support hypotheses. This supplies the ten-field certificate rows and the two crossed singleton-classifier exclusions in the same $\mathsf{RatUpFieldSource}_F$ boundary.

Next apply Definition 45.583 to the displayed source tuple, denominator tuple, and action Γ under its normalized contextual hypotheses. Its clauses add the assembled boundary, carried contextual endpoint, normalized classifier row, strict-support equivalence, exit-row closure, direct endpoint separation, singleton-context carrier rejections, and singleton-classifier obstruction.

Apply Definition 45.582 to p_{abcde} and the apartness-to-nonzero maps. The forward direction yields the five factor witnesses p_a, p_b, p_c, p_d, p_e used by the inverse spine R_5.

Finally apply Definition 45.296 with P_5 and p_{abcde}. It gives the right and left equation exactness rows for every $u, v : \mathsf{BHist}$, with R_5 built from the same five factor witnesses.

All clauses are indexed by BHist and by the concrete source tuple $\mathsf{RatUpFieldSource}_F$. Repacking them therefore gives one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package rather than separate local rows. □

Per-object template: traditional target concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ cross-endpoint exactness; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, the denominator tuple, append, and $\mathsf{ThreeSupp}^u_D$; carrier $\mathsf{ThreeSupp}^u_D$ plus $\mathsf{RatHistoryCarrier}$; classifier R^u_D plus $\mathsf{fieldSingletonEmptyClassifier}$; stability obligations bracketing-independent support and cross-endpoint classifier rejection; ledger paired denominator endpoints and contextual append rows; core theorem this block; exactness predicate crossed singleton-classifier noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target bracketing-independent concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator exit support; BEDC source BHist with the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations paired left- and right-nested support exactness plus contextual append separation; ledger the paired denominator endpoints $\mu^u_D(\mu^u_D(h,k),\ell)$ and $\mu^u_D(h,\mu^u_D(k,\ell))$; core theorem Definition 45.599; exactness predicate contextual singleton-carrier, inverse-domain, and singleton-classifier exclusion for both product bracketings; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.603 (RatUp FieldUp bracketing-independent full crossed contextual boundary)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $x := \mu^u_D(\mu^u_D(h,k),\ell)$ and $y := \mu^u_D(h,\mu^u_D(k,\ell))$. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then the two bracketings of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator endpoint carry the following single contextual boundary:*

1. *$C^u_D(x)$, $C^u_D(y)$, and $R^u_D(x,y)$;*
2. *for every context history $r : \mathsf{BHist}$, the four contextual carrier exclusions:*
   - *(a)* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,x)) \to \bot$;
   - *(b)* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x,r)) \to \bot$;
   - *(c)* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r,y)) \to \bot$;
   - *(d)* $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y,r)) \to \bot$.
3. *for every context history $r : \mathsf{BHist}$, the four inverse-domain witnesses:*
   - *(a)* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,x))$;
   - *(b)* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x,r))$;
   - *(c)* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r,y))$;
   - *(d)* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(y,r))$.
4. *for all $L, R : \mathsf{BHist}$, the four contextual singleton-classifier exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,x),\mathsf{append}(R,x)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,y),\mathsf{append}(R,y)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,x),\mathsf{append}(R,y)) \to \bot,$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L,y),\mathsf{append}(R,x)) \to \bot.$$

Proof. First apply Definition 45.599 to the displayed concrete denominator tuple, carried histories, endpoint definitions, and support witness. This gives the carrier rows $C^u_D(x)$ and $C^u_D(y)$, the classifier row $R^u_D(x, y)$, the four contextual singleton-carrier exclusions around x and y, the four corresponding $\mathsf{fieldSingletonEmptyNonZero}$ witnesses, and the two same-endpoint singleton-classifier exclusions.

Next apply Definition 45.600 under the same hypotheses and with the same definitions of x and y. This supplies the two crossed contextual singleton-classifier exclusions, one for (x, y) and one for (y, x), for arbitrary $L, R : \mathsf{BHist}$.

The two theorem applications share the same BHist carrier, the same denominator tuple, and the same support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. Collect the carrier, inverse-domain, same-endpoint classifier, and crossed classifier rows into the displayed boundary package. □

Per-object template: traditional target bracketing-independent concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator exit support; BEDC source BHist with the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations paired left- and right-nested support exactness plus contextual append separation and crossed classifier rejection; ledger the paired denominator endpoints $\mu^u_D(\mu^u_D(h, k), \ell)$ and $\mu^u_D(h, \mu^u_D(k, \ell))$; core theorem Definition 45.603; exactness predicate contextual singleton-carrier, inverse-domain, and same-endpoint and crossed singleton-classifier exclusion for both product bracketings; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.604 (RatUp FieldUp crossed ten-field exit certificate)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete ten-field paper certificate for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *over* $\mathsf{RatUpFieldSource}_F$ *is the composition of Definition 45.586, Definition 45.589, Definition 45.582, Definition 45.599, and Definition 45.600. Its conclusion contains all of the following concrete rows, with no schematic* mul*,* add*, or* neg *operator parameter:*

1. *the same-bracketing endpoint rows* $C^u_D(x)$*,* $C^u_D(y)$*,* $R^u_D(x, y)$*, and, for all* $L, R : \mathsf{BHist}$*, the contextual singleton-classifier exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, y)) \to \bot;$$

2. *the crossed contextual singleton-classifier exclusions, for all* $L, R : \mathsf{BHist}$*,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot;$$

3. *the two-sided product classifier and nonzero equivalences, for every* $a, b, u, v : \mathsf{BHist}$ *with certified outer witnesses* $p_a : \mathsf{NonZero}_F(a)$ *and* $p_b : \mathsf{NonZero}_F(b)$*,*

$$(\mathsf{mul}(\mathsf{mul}(a, u), b)\mathsf{hsamemul}(\mathsf{mul}(a, v), b)) \iff u\mathsf{hsame}v$$

and

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(a, u), b)) \iff \mathsf{NonZero}_F(u);$$

4. the five-factor nonzero ledger, for every $a, b, c, d, e : \mathsf{BHist}$*,*

$$\mathsf{NonZero}_F(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a,b),c),d),e)) \iff \mathsf{NonZero}_F(a) \wedge \mathsf{NonZero}_F(b) \wedge \mathsf{NonZero}_F(c) \wedge \mathsf{NonZero}_F(d) \wedge \mathsf{NonZ}$$

Proof. Begin with Definition 45.586. It fixes the paper certificate site, the $\mathsf{RatUpFieldSource}_F$ source, the BHist carrier, the contextual action, and the denominator rows as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ package.

Next apply Definition 45.599. It supplies the same-bracketing endpoint rows $C^u_D(x)$, $C^u_D(y)$, and $R^u_D(x, y)$, together with the x–x and y–y contextual singleton-classifier exclusions for arbitrary $L, R : \mathsf{BHist}$.

Then apply Definition 45.600. It supplies the two crossed contextual singleton-classifier exclusions, again for arbitrary $L, R : \mathsf{BHist}$, so the certificate consumes both the x–y and y–x classifier boundary rows.

Finally use Definition 45.589 for the two-sided product classifier and nonzero equivalences, and Definition 45.582 for the five nonzero factor witnesses. These clauses share the same concrete source, carrier, classifier, and ledger vocabulary, so they assemble as one ten-field exit certificate rather than as separate paper endpoints. □

Per-object template: traditional target crossed RatUp FieldUp exit certificate; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, denominator tuple, contextual action, and product/nonzero rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ plus singleton-empty classifier boundary; stability obligations the same-bracketing and crossed contextual boundaries plus product classifier exactness; ledger denominator bracketings, contextual endpoints, nonzero factors, and product witnesses; core theorem Definition 45.604; exactness the paired same/cross classifier exclusions and product equivalences; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.605 (RatUp FieldUp bracketing-complete classifier exit certificate)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit certificate contains the following bracketing-complete classifier package:*

1. the endpoint certificate rows $C^u_D(x)$*,* $C^u_D(y)$*, and* $R^u_D(x, y)$*;*

2. for every context history $r : \mathsf{BHist}$*, the contextual carrier exclusions*

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot, \quad \mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, y)) \to \bot, \quad \mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r)) \to \bot;$$

3. for every context history $r : \mathsf{BHist}$*, the inverse-domain witnesses*

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x)), \quad \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x, r)),$$

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, y)), \quad \mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(y, r));$$

4. for all $L, R : \mathsf{BHist}$*, the four ordered endpoint-pair singleton-classifier exclusions*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, y)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot.$$

Proof. Apply Definition 45.586. This supplies the ambient ten-field certificate boundary for the concrete $\mathsf{RatUpFieldSource}_F$ instance and identifies the package rows as clauses of the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit certificate.

Next apply Definition 45.599 to the same denominator tuple, histories, endpoint definitions, and support witness. It gives the endpoint carrier rows, the classifier row $R^u_D(x, y)$, the contextual carrier exclusions around both bracketings, the inverse-domain witnesses around both bracketings, and the two same-endpoint singleton-classifier exclusions.

Finally apply Definition 45.600 with the same data. This supplies the two crossed singleton-classifier exclusions for (x, y) and (y, x) under arbitrary contexts $L, R : \mathsf{BHist}$. Together with the same-endpoint rows already obtained, these are exactly the four ordered endpoint-pair classifier exclusions, so the certificate package is bracketing-complete.

For the x–y crossed classifier, unpack a witness of $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y))$. Its source and target carrier fields include $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, x))$ and $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{a}$ Applying the source-carrier rejection for $\mathsf{append}(L, x)$ gives $\bot$.

For the y–x crossed classifier, unpack a witness of $\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x))$. Its source and target carrier fields include $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, y))$ and $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{a}$ Applying the source-carrier rejection for $\mathsf{append}(L, y)$ gives $\bot$. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing independent cross endpoint classifier boundary`

Per-object template: traditional target bracketing-complete classifier exit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, μ^u_D, R^u_D, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D plus singleton-empty classifier exclusions; stability obligations the same-endpoint and crossed bracketing-boundary theorem sites; ledger paired denominator endpoints $\mu^u_D(\mu^u_D(h, k), \ell)$ and $\mu^u_D(h, \mu^u_D(k, \ell))$; exactness predicate four-pair classifier noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Definition 45.606 (RatUp FieldUp bracketing endpoint selector)**.** *For the concrete denominator operation μ^u_D and histories $h, k, \ell : \mathsf{BHist}$, define the selector endpoint $E_s(h, k, \ell) : \mathsf{BHist}$ by the two generated* BMark *cases:*

$$E_{\mathsf{BMark.b0}}(h, k, \ell) := \mu^u_D(\mu^u_D(h, k), \ell),$$

$$E_{\mathsf{BMark.b1}}(h, k, \ell) := \mu^u_D(h, \mu^u_D(k, \ell)).$$

Thus $E_s(h, k, \ell)$ is the closed map out of BMark *whose* $\mathsf{BMark.b0}$ *branch is the left-nested denominator endpoint and whose* $\mathsf{BMark.b1}$ *branch is the right-nested denominator endpoint.*

Lean4 (definition) `BEDC.Derived.FieldUp.ratup fieldup bracketing endpoint selector`
Lean4 (definition) `BEDC.Derived.FieldUp.FieldRatBracketingEndpointSelector`

Theorem 45.607 (RatUp FieldUp bracketing endpoint selector carrier coverage)**.** *For histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, if*

$$(N^u_D(h) \vee N^u_D(k)) \vee N^u_D(\ell)$$

holds, then both concrete bracketings carry rational denominator support, and every selected endpoint $E_s(h, k, \ell)$ carries that support:

$$N^u_D(\mu^u_D(\mu^u_D(h, k), \ell)) \wedge N^u_D(\mu^u_D(h, \mu^u_D(k, \ell))) \wedge \forall s : \mathsf{BMark},\ N^u_D(E_s(h, k, \ell)).$$

Proof. First close $C^u_D(h)$ and $C^u_D(k)$ under continuation to obtain $C^u_D(\mu^u_D(h,k))$. Split the concrete support disjunction. A support witness on h or k closes the left bracketing by the left or right append carrier law and then appends ℓ; a support witness on ℓ closes the left bracketing from the carried prefix. Associativity transports this carrier to the right bracketing. Finally, case analysis on $s : \mathsf{BMark}$ returns the corresponding carried endpoint. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing endpoint selector carrier coverage`

Theorem 45.608 (RatUp FieldUp bracketing endpoint selector sameness)**.** *For all histories $h, k, \ell : \mathsf{BHist}$, every selected endpoint $E_s(h,k,\ell)$ is classified with one of the two concrete bracketings, and any two selected endpoints are* hsame*:*

$$\forall s : \mathsf{BMark},\ E_s(h,k,\ell)\mathsf{hsame}\mu^u_D(\mu^u_D(h,k),\ell) \vee E_s(h,k,\ell)\mathsf{hsame}\mu^u_D(h,\mu^u_D(k,\ell)),$$

$$\forall s,t : \mathsf{BMark},\ E_s(h,k,\ell)\mathsf{hsame}E_t(h,k,\ell).$$

Proof. Case analysis on s gives either the left bracketing or the right bracketing by definition of the selector. For the pairwise statement, case analysis on s and t leaves the two diagonal cases, closed by reflexivity, and the two crossed cases, closed by associativity and symmetry of hsame. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing endpoint selector same`

Theorem 45.609 (RatUp FieldUp bracketing choice classifier exclusion)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h,k,\ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then for all selectors $s,t : \mathsf{BMark}$ and all $L, R : \mathsf{BHist}$,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)), \mathsf{append}(R, E_t(h,k,\ell))) \to \bot.$$

Proof. Set $x := \mu^u_D(\mu^u_D(h,k),\ell)$ and $y := \mu^u_D(h,\mu^u_D(k,\ell))$. Apply Definition 45.605 to the same denominator tuple, carrier hypotheses, endpoint definitions, and $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ witness. This gives the four ordered endpoint-pair singleton-classifier exclusions for (x,x), (y,y), (x,y), and (y,x).

Use $\mathsf{BMark_generated_cases}$ on s and again on t. There are four closed selector pairs.

1. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b0}$, then the selected endpoints are x and x. The (x,x) exclusion from the complete certificate maps the displayed classifier witness to $\bot$.
2. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b1}$, then the selected endpoints are y and y. The (y,y) exclusion applies.
3. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b1}$, then the selected endpoints are x and y. The crossed (x,y) exclusion applies.
4. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b0}$, then the selected endpoints are y and x. The crossed (y,x) exclusion applies.

These four cases exhaust the generated BMark selector pairs, so the selector-indexed classifier witness is impossible for arbitrary $s,t : \mathsf{BMark}$ and arbitrary contexts $L, R : \mathsf{BHist}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector classifier exclusion`

Theorem 45.610 (RatUp FieldUp bracketing selector denominator classifier coverage)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all selectors* $s, t : \mathsf{BMark}$,

$$R^u_D(E_s(h, k, \ell), E_t(h, k, \ell)).$$

Proof. Set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. Apply Definition 45.605 to the same denominator tuple, carrier hypotheses, endpoint definitions, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This supplies $C^u_D(x)$, $C^u_D(y)$, and the endpoint row $R^u_D(x, y)$.

The concrete denominator classifier is the classifier row of the denominator name certificate. Its stability obligations give carrier-relative reflexivity and symmetry. Thus $C^u_D(x)$ gives $R^u_D(x, x)$, $C^u_D(y)$ gives $R^u_D(y, y)$, and symmetry converts $R^u_D(x, y)$ into $R^u_D(y, x)$.

Use BMark_generated_cases on s and again on t. There are four closed selector pairs.

1. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b0}$, then the selected endpoints are x and x. The reflexive row $R^u_D(x, x)$ gives the claim.
2. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b1}$, then the selected endpoints are y and y. The reflexive row $R^u_D(y, y)$ gives the claim.
3. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b1}$, then the selected endpoints are x and y. The bracketing-complete endpoint row $R^u_D(x, y)$ gives the claim.
4. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b0}$, then the selected endpoints are y and x. The symmetric endpoint row $R^u_D(y, x)$ gives the claim.

These cases exhaust the generated BMark selector interface, so every selected endpoint pair is classified internally by R^u_D. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldRatBracketingEndpointSelector denominator classifier`

Theorem 45.611 (RatUp FieldUp bracketing selector Rat carrier coverage)**.** *For histories* $h, k, \ell : \mathsf{BHist}$ *with denominator-unit carrier witnesses, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If at least one of* h, k, ℓ *is a Rat history carrier in the displayed three-support sense, then for every selector* $s : \mathsf{BMark}$,

$$\mathsf{RatHistoryCarrier}(E_s(h, k, \ell)).$$

Proof. For the left-nested endpoint, split the three-support witness. If h or k is already Rat-carried, the denominator-unit append closure keeps the left append Rat-carried and then closes the final append with ℓ. If ℓ is Rat-carried, first close the denominator-unit append $h \cdot k$ and then append the Rat right endpoint. The right-nested endpoint follows by transporting this carrier across associativity. The two BMark cases select exactly these two endpoints. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector rat carrier coverage`

Theorem 45.612 (RatUp FieldUp bracketing selector Rat classifier coverage)**.** *Under the same denominator-unit and three-support hypotheses, every selected bracketing endpoint pair is classified by the Rat history classifier:*

$$\forall s, t : \mathsf{BMark}.\ \mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_t(h, k, \ell)).$$

Proof. Apply the selector Rat-carrier coverage theorem to both selected endpoints. The classifier's sameness row is obtained by the four generated selector cases: the equal-branch cases use reflexivity, while the crossed cases use append associativity and its symmetry. These three fields assemble the Rat history classifier for the selected pair. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector rat classifier coverage`

Theorem 45.613 (RatUp FieldUp bracketing selector dual classifier coverage)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then every selector pair* $s, t : \mathsf{BMark}$ *satisfies the combined denominator and Rat row:*

$$N^u_D(E_s(h,k,\ell)) \wedge N^u_D(E_t(h,k,\ell)) \wedge R^u_D(E_s(h,k,\ell), E_t(h,k,\ell)) \wedge \mathsf{RatHistoryCarrier}(E_s(h,k,\ell)) \wedge \mathsf{RatHistoryCarrier}(E_t(h$$

Proof. The three-support hypothesis supplies the denominator-support disjunction required by Definition 45.607. Apply that theorem to the carrier witnesses for h, k, ℓ and then specialize its selector row to s and to t. This gives the two $N^u_D(E_s(h, k, \ell))$ and $N^u_D(E_t(h, k, \ell))$ rows.

Next apply Definition 45.610 with the same denominator tuple, carrier witnesses, three-support witness, and selectors. This gives the denominator classifier row $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$.

For the Rat carrier rows, apply Definition 45.611 to the same denominator-unit and three-support data, first at s and then at t. Apply Definition 45.612 to the same selector pair for the Rat classifier row. The six obtained rows assemble directly into the displayed conjunction. □

Theorem 45.614 (RatUp FieldUp bracketing selector independence package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all selectors* $s, t : \mathsf{BMark}$*, every context* $r : \mathsf{BHist}$*, and all classifier contexts* $L, R : \mathsf{BHist}$*, setting* $a := E_s(h, k, \ell)$ *and* $b := E_t(h, k, \ell)$ *gives the following package:*

1.
$$a\mathsf{hsame}b;$$

2.
$$\mathsf{RatHistoryClassifier}(a, b);$$

3.
$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, a), \mathsf{append}(R, b)) \to \bot;$$

4.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, a)) \to \bot;$$

5.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(a, r)) \to \bot;$$

6.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, a));$$

7.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(a, r)).$$

8.

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, b)) \to \bot;$$

9.

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(b, r)) \to \bot;$$

10.

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, b));$$

11.

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(b, r)).$$

Proof. First apply Definition 45.608 to the selectors $s, t : \mathsf{BMark}$. This gives the endpoint sameness row $E_s(h, k, \ell)\mathsf{hsame}E_t(h, k, \ell)$.

Next apply Definition 45.612 to the same carrier witnesses and the same $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This gives the Rat classifier row $\mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_t(h, k, \ell))$.

Apply Definition 45.609 to the same denominator tuple, the carrier witnesses, the $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness, the selectors s, t, and the classifier contexts $L, R : \mathsf{BHist}$. This gives the selected-pair singleton-classifier exclusion.

Finally apply Definition 45.624 to the selector s and the context $r : \mathsf{BHist}$. Its four rows give the two one-sided singleton-carrier exclusions and the two corresponding fieldSingletonEmptyNonZero witnesses for $E_s(h, k, \ell)$.

Apply the same theorem again to the selector t and the same context r. The resulting four rows give the matching singleton-carrier exclusions and fieldSingletonEmptyNonZero witnesses for $E_t(h, k, \ell)$. Replacing the temporary names a and b by their selector definitions assembles the displayed package. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector independence package`

Per-object template: traditional target bracketing-selector independence inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, the denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, and selectors E_s, E_t; carrier C^u_D and RatHistoryCarrier; classifier R^u_D and RatHistoryClassifier; stability obligations selector sameness, selector denominator coverage, selector Rat coverage, and contextual nonzero package rows for both selected endpoints on both context sides; ledger selector choices plus both context sides; exactness predicate selector-independent row coverage; standard bridge deferred; status ProofTarget.

Per-object template: traditional target selector-indexed bracketing classifier coverage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, μ^u_D, R^u_D, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D and RatHistoryCarrier; classifier block records both the denominator classifier R^u_D and the Rat classifier RatHistoryClassifier; stability obligations reflexivity, symmetry, bracketing-complete endpoint coverage, Rat carrier coverage, and selector sameness; ledger the selected endpoint pair (s, t); exactness predicate paired denominator-classifier and Rat-classifier coverage over generated selector pairs; standard bridge deferred; status ProofTarget.

Theorem 45.615 (RatUp FieldUp concrete operation exit exhaustion)**.** *For the concrete table of Definition 45.587, the rows certified by Definition 45.588, Definition 45.586, Definition 45.589, and Definition 45.605 exhaust the concrete exit object of Definition 45.617.*

The exhaustion contains carrier and classifier anchoring, the $\mathsf{CommRing}^{\uparrow}$ *base rows, denominator multiplication and unit rows, contextual and context-pair rows, affine rows, inverse and nonzero rows, two-sided product exactness, and the same-endpoint plus crossed singleton-classifier noncollapse rows for the paired bracketing endpoints.*

Proof. First invoke Definition 45.588. This identifies the table inventory over $\mathsf{RatUpFieldSource}_F$: the carrier and classifier anchors, the $\mathsf{CommRing}^{\uparrow}$ base coordinate, denominator multiplication and unit, contextual action, context-pair support, affine closure, inverse rows, nonzero rows, and exit noncollapse rows.

Next invoke Definition 45.586. It packages the same inventory as the ten-field per-object certificate for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ target, with the source, carrier, classifier, stability obligations, ledger, exactness predicate, bridge slot, and status $\mathsf{ProofTarget}$ fixed by the displayed table.

For two-sided multiplication through certified nonzero histories, invoke Definition 45.589. This adds product classifier exactness and nonzero transport to the same concrete exit object, so the product rows are not external parameters.

Finally invoke Definition 45.605. It supplies the same-endpoint and crossed singleton-classifier noncollapse rows for the ordered endpoint pairs $\mu^u_D(\mu^u_D(h,k),\ell)$ and $\mu^u_D(h,\mu^u_D(k,\ell))$. Together with the table laws, the ten-field certificate, and the product package, these clauses cover every row named in the concrete operation table and therefore exhaust the concrete exit object of Definition 45.617. □

Per-object template: traditional target concrete operation exit exhaustion for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source $\mathsf{RatUpFieldSource}_F$ and BHist; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ together with $\mathsf{RatDenomUnitClassifier}$; stability obligations carrier/classifier anchoring, $\mathsf{CommRing}^{\uparrow}$ base rows, denominator multiplication and unit rows, contextual and context-pair rows, affine rows, inverse and nonzero rows, product exactness, and singleton noncollapse rows; ledger denominator histories, context pairs, nonzero product witnesses, affine witnesses, and bracketing endpoint pairs; exactness predicate complete row coverage for Definition 45.617; standard bridge deferred; status $\mathsf{ProofTarget}$. Per-object template: traditional target bracketing-independent concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator exit support; BEDC source BHist with the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{ThreeSupp}^u_D$; classifier R^u_D; stability obligations paired left- and right-nested support exactness plus contextual append separation; ledger the paired denominator endpoints $\mu^u_D(\mu^u_D(h,k),\ell)$ and $\mu^u_D(h,\mu^u_D(k,\ell))$; core theorem Definition 45.599; exactness predicate contextual singleton-carrier, inverse-domain, and singleton-classifier exclusion for both product bracketings; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target bracketing-complete classifier exit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, μ^u_D, R^u_D, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D plus singleton-empty classifier exclusions; stability obligations the same-endpoint and crossed bracketing-boundary theorem sites; ledger paired denominator endpoints $\mu^u_D(\mu^u_D(h,k),\ell)$ and $\mu^u_D(h,\mu^u_D(k,\ell))$; exactness predicate four-pair classifier noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.616 (RatUp FieldUp paper algebra hierarchy exit package)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$*, the* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *algebra-hierarchy ProofTarget is recorded as a single paper-side package over* BHist*. This package contains, as concrete certificate rows, the ten-field certificate, one-sided multiplication cancellation exactness, the two-sided product classifier and nonzero transport, the five-factor nonzero ledger, the cross-endpoint singleton-classifier exclusions, and the bracketing-complete classifier package. Every listed row is carried by the concrete BEDC histories and denominator endpoints of* $\mathsf{RatUpFieldSource}_F$*; no schematic* mul*,* add*, or* neg *parameter is part of this paper-side package.*

Proof. First apply Definition 45.586. This gives the concrete ten-field certificate for $\mathsf{RatUpFieldSource}_F$, including the history-level carrier, classifier, unit, inverse-domain, and continuation rows that form the ambient $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ boundary.

Next apply Definition 45.589 and Definition 45.582. The first theorem records the one-sided multiplication cancellation exactness, two-sided product classifier, and nonzero transport rows, while the second records the five-factor nonzero ledger as a concrete BHist factor certificate.

Then apply Definition 45.601 and Definition 45.605. These supply the cross-endpoint singleton-classifier exclusions and the bracketing-complete classifier package for the paired denominator endpoints.

All cited rows use the same concrete source tuple, the same BHist carrier, and the same denominator endpoint data. Assembling these rows gives the concrete paper-side package required for a machine-checked $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ Phase Algebra Hierarchy exit. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAlgebraHierarchyExitWitness`

Per-object template: traditional target RatUp FieldUp algebra hierarchy paper package; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier the concrete RatUp history rows and denominator endpoints; classifier the ten-field rows, one-sided multiplication cancellation exactness, two-sided product classifier, nonzero transport, five-factor ledger, singleton-classifier exclusions, and bracketing-complete classifier package; stability obligations the cited concrete exit package rows; ledger the assembled $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ hierarchy certificate; core theorem Definition 45.616; exactness predicate concrete hierarchy row coverage without schematic operation parameters; standard bridge deferred; status ProofTarget.

Definition 45.617 (RatUp FieldUp concrete exit object)**.** *For* $\mathsf{RatUpFieldSource}_F$ *and a finite row package* $\mathcal{E}$ *over* BHist*, write*

$$\mathsf{RatUpFieldExitObject}_F(\mathcal{E})$$

for the concrete predicate asserting that $\mathcal{E}$ *is the conjunction of the following rows:*

1. *the carrier and classifier rows* RatHistoryCarrier*,* RatHistoryClassifier*,* RatDenomUnitCarrier*, and* RatDenomUnitClassifier*;*
2. *the concrete schema instance row for* $\mathsf{RatUpFieldSource}_F$*;*
3. *the concrete operation-exit exhaustion row for* append*,* Cont*, the denominator unit, the contextual action* Γ*, the context-pair action, affine rows, inverse and nonzero rows, one-sided multiplication cancellation exactness, two-sided product exactness, and the same-endpoint plus crossed singleton-classifier noncollapse rows;*
4. *the algebra-hierarchy exit witness row for* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$*;*
5. *the selector inverse-domain coverage rows generated by* $E_s(h,k,\ell)$*: selected endpoint carrier coverage, selected endpoint denominator-classifier coverage, strict endpoint nonzero witnesses, and selector-indexed singleton-classifier exclusion under arbitrary contexts, and selector row-stability transport for* R^u_D*-related denominator triples and* hsame*-related contexts;*
6. *the selector-independent final boundary rows of Definition 45.764: both final-certificate two-selector noncollapse packages, the two componentwise* R^u_D *comparisons, and the two* RatHistoryClassifier *comparisons for selectors* s,t,u,v *and contexts* L,R,r*;*
7. *the selector-chain final-boundary rows of Definition 45.765: the three final-certificate two-selector noncollapse packages for selector pairs* (s,t)*,* (u,v)*, and* (w,z)*, the chained* R^u_D *rows from* E_s *to* E_w *and from* E_t *to* E_z*, and the chained* RatHistoryClassifier *rows for the same endpoint pairs and contexts* L,R,r*.*

The concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object is the paper package* $\mathcal{E}$ *satisfying* $\mathsf{RatUpFieldExitObject}_F(\mathcal{E})$*. No schematic* mul*,* add*, or* neg *parameter is part of this predicate.*

Theorem 45.618 (RatUp FieldUp concrete exit object is complete)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the paper package* $\mathcal{E}_F$ *determined by Definition 45.617 satisfies*

$$\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F).$$

Moreover $\mathcal{E}_F$ *is exactly the paper-side* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package over* BHist*: its source, carrier, classifier, stability obligations, ledger, core theorem list, exactness predicate, standard-bridge slot, and status are precisely the rows named in Definition 45.617.*

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupConcreteExitObjectComplete`

Proof. Apply Definition 45.502. It fixes the single BHist source tuple for $\mathsf{RatUpFieldSource}_F$ and identifies the carrier, classifier, source, stability, ledger, and exactness rows consumed by the concrete package $\mathcal{E}_F$.

Next apply Definition 45.615. This covers every operation row named in Definition 45.617, including the carrier and classifier anchors, denominator rows, contextual rows, affine rows, inverse and nonzero rows, product exactness, and singleton-classifier noncollapse rows.

Apply Definition 45.616. It packages those rows as the Phase Algebra Hierarchy paper certificate for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$, with the bridge slot and status recorded as part of the same paper-side object.

Finally apply Definition 45.619. It supplies the selector inverse-domain coverage rows generated by $E_s(h,k,\ell)$, including selected endpoint carrier coverage, denominator classifier coverage, strict endpoint nonzero witnesses, and contextual singleton-classifier exclusion.

Apply Definition 45.637 to the two R^u_D-related denominator triples and the hsame-related context pair. This is the stability component of the selector row family in the concrete exit object, transporting the selected endpoint classifier rows and the primed row package along the denominator relation.

Apply Definition 45.764 to the same concrete exit object, denominator tuple, carrier witnesses, support witness, selector quadruple, and contexts. This covers both final-certificate two-selector noncollapse packages, the componentwise R^u_D rows, and the componentwise RatHistoryClassifier rows named by the final boundary conjunct.

Apply Definition 45.765 to the same concrete exit object, denominator tuple, carrier witnesses, support witness, selector sextuple, and contexts. This covers the three final-certificate two-selector noncollapse packages for (s,t), (u,v), and (w,z), the chained R^u_D endpoint rows, and the chained RatHistoryClassifier endpoint rows named by the selector-chain final-boundary conjunct.

These cited inputs cover exactly the conjuncts defining $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$, including both selector-independent and selector-chain final-boundary rows. Hence the predicate holds, and because the algebra-hierarchy witness uses the same BHist source and row inventory, $\mathcal{E}_F$ is exactly the paper-side $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package over BHist. □

Theorem 45.619 (RatUp FieldUp concrete exit selector inverse-domain coverage)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete exit object of Definition 45.617 contains the strict inverse-domain selector row for the concrete tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*. Explicitly, for histories* h,k,ℓ : BHist *with* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$*, selectors* s,t : BMark*, and contexts* L,R : BHist*, the concrete exit object contains the row*

$$C^u_D(E_s(h,k,\ell))\wedge C^u_D(E_t(h,k,\ell))\wedge R^u_D(E_s(h,k,\ell),E_t(h,k,\ell))\wedge\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))\wedge\mathsf{fieldSingleton}$$

Proof. Apply Definition 45.634 to the same concrete denominator tuple, carrier hypotheses, support witness, selector pair, and context pair. It supplies the displayed strict selector row, including the two selected endpoint carrier rows, the denominator classifier row, the two strict endpoint nonzero witnesses, and the contextual singleton-classifier obstruction.

By Definition 45.617, selector inverse-domain coverage is one conjunct of $\mathsf{RatUpFieldExitObject}_F$ for the concrete exit package. Therefore the strict row supplied by the selector package is one of the concrete exit-object rows for $\mathsf{RatUpFieldSource}_F$. □

Theorem 45.620 (RatUp FieldUp concrete exit selector row stability)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete exit object of Definition 45.617 contains the selector row-stability transport for the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*. Explicitly, for denominator triples* $h, k, \ell, h', k', \ell' : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*,* $C^u_D(h')$*,* $C^u_D(k')$*, and* $C^u_D(\ell')$*, assume*

$$\mathsf{ThreeSupp}^u_D(h, k, \ell) \quad \textit{and} \quad \mathsf{ThreeSupp}^u_D(h', k', \ell')$$

and

$$R^u_D(h, h'), \quad R^u_D(k, k'), \quad R^u_D(\ell, \ell').$$

Then for every selector pair $s, t : \mathsf{BMark}$ *and all contexts* $L, R, L', R' : \mathsf{BHist}$ *with* $\mathsf{hsame}(L, L')$ *and* $\mathsf{hsame}(R, R')$*, the concrete exit object contains the row*

$$R^u_D(E_s(h,k,\ell), E_s(h',k',\ell')) \wedge R^u_D(E_t(h,k,\ell), E_t(h',k',\ell')) \wedge C^u_D(E_s(h',k',\ell')) \wedge C^u_D(E_t(h',k',\ell')) \wedge R^u_D(E_s(h',k',\ell'), E$$

Proof. Apply Definition 45.637 to the same concrete denominator tuple, the six carrier hypotheses, the two support witnesses, the three R^u_D comparisons, the selector pair, and the hsame-related contexts. It supplies the two selected endpoint classifier transports and the primed three-field row package.

By Definition 45.617, selector row-stability transport is a conjunct of the concrete exit object. Therefore the row supplied by Definition 45.637 is contained in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object for $\mathsf{RatUpFieldSource}_F$. □

Per-object template: traditional target RatUp FieldUp concrete exit object completion; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatDenomUnitCarrier}$; classifier $\mathsf{RatHistoryClassifier}$ and $\mathsf{RatDenomUnitClassifier}$; stability obligations the concrete schema, operation-exhaustion, and selector row-stability rows; ledger the denominator, contextual, affine, inverse, nonzero, product, singleton-classifier, selector inverse-domain, selector stability, selector-independent final-boundary, and selector-chain final-boundary rows; core theorem list Definition 45.502, Definition 45.615, Definition 45.616, Definition 45.619, Definition 45.620, Definition 45.764, and Definition 45.765; exactness predicate complete concrete row coverage for Definition 45.617; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.621 (RatUp FieldUp concrete exit object hierarchy exactness)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the finite row package of Definition 45.617 is exactly the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *hierarchy exit package. Its carrier and classifier rows are the concrete* $\mathsf{RatHistory}$ *and* $\mathsf{RatDenomUnit}$ *rows, and its operation and exactness rows are precisely those certified by Definition 45.615 and Definition 45.616.*

Proof. Apply Definition 45.615. It exhausts the row inventory of Definition 45.617: the carrier and classifier anchoring rows, the $\mathsf{CommRing}^{\uparrow}$ base rows, denominator multiplication and unit rows, contextual and context-pair rows, affine rows, inverse and nonzero rows, two-sided product exactness, and same-endpoint plus crossed singleton-classifier noncollapse rows.

Apply Definition 45.616. It packages the same concrete rows as the algebra-hierarchy exit witness for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ over $\mathsf{RatUpFieldSource}_F$, including the ten-field certificate, product classifier and nonzero transport, five-factor ledger, cross-endpoint exclusions, and bracketing-complete classifier package.

Because both cited theorems use the same source tuple, BHist carrier, denominator endpoints, and operation-row inventory, the finite package defined in Definition 45.617 and the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ hierarchy exit package have the same rows and the same exactness obligations. □

Lemma 45.622 (RatUp FieldUp BMark selector endpoint-pair cases)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$, let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. Set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. Then for all selectors $s, t : \mathsf{BMark}$, the selected endpoint pair $(E_s(h, k, \ell), E_t(h, k, \ell))$ is covered by the four generated rows (x, x), (y, y), (x, y), and (y, x) up to hsame on both endpoints.*

Proof. Perform generated case analysis on the selector pair (s, t). In every branch, unfold the endpoint selector of Definition 45.606; the remaining endpoint comparison is reflexivity of hsame.

In the $(\mathsf{BMark.b0}, \mathsf{BMark.b0})$ branch, both selected endpoints unfold to x, so the row (x, x) supplies the pair witness.

In the $(\mathsf{BMark.b1}, \mathsf{BMark.b1})$ branch, both selected endpoints unfold to y, so the row (y, y) supplies the pair witness.

In the $(\mathsf{BMark.b0}, \mathsf{BMark.b1})$ branch, the left endpoint unfolds to x and the right endpoint unfolds to y, giving the row (x, y).

In the $(\mathsf{BMark.b1}, \mathsf{BMark.b0})$ branch, the left endpoint unfolds to y and the right endpoint unfolds to x, giving the row (y, x). □

Theorem 45.623 (RatUp FieldUp BMark-selected bracketing exit subpackage)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then for all selectors $s, t : \mathsf{BMark}$ and all contexts $L, R : \mathsf{BHist}$, the selected endpoints form the following concrete exit subpackage:*

1. *$C^u_D(E_s(h, k, \ell))$ and $C^u_D(E_t(h, k, \ell))$;*
2. *$R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$;*
3. $$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h, k, \ell)), \mathsf{append}(R, E_t(h, k, \ell))) \to \bot.$$

Proof. Set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. Apply Definition 45.605 to the same concrete denominator tuple, carrier hypotheses, endpoint definitions, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This supplies $C^u_D(x)$, $C^u_D(y)$, $R^u_D(x, y)$, and the four ordered endpoint-pair singleton-classifier exclusions.

For carrier coverage, use the two generated BMark cases for s. In the $\mathsf{BMark.b0}$ branch, the selected endpoint is x, so $C^u_D(x)$ transports to $C^u_D(E_s(h, k, \ell))$. In the $\mathsf{BMark.b1}$ branch, the selected endpoint is y, so $C^u_D(y)$ transports to $C^u_D(E_s(h, k, \ell))$. The same two-case argument for t gives $C^u_D(E_t(h, k, \ell))$.

The displayed denominator-classifier row is exactly Definition 45.610 applied to the same carrier data and support witness. It packages the reflexive, symmetric, and crossed endpoint classifier rows across all selected pairs $s, t : \mathsf{BMark}$.

The singleton-classifier rejection under arbitrary contexts is exactly Definition 45.609 applied to the same selector endpoints. Thus the selected pair (s, t) and the contexts (L, R) carry both

the internal denominator classifier row and the external singleton-classifier exclusion required by the concrete exit subpackage. □

Theorem 45.624 (RatUp FieldUp selector contextual nonzero package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for every selector* $s : \mathsf{BMark}$ *and every context* $r : \mathsf{BHist}$*, the selected endpoint has the following contextual nonzero package:*

1.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, E_s(h, k, \ell))) \to \bot;$$

2.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(E_s(h, k, \ell), r)) \to \bot;$$

3.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, E_s(h, k, \ell)));$$

4.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(E_s(h, k, \ell), r)).$$

Proof. Set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. Apply Definition 45.605 to the same concrete denominator tuple, carrier hypotheses, endpoint definitions, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This supplies the two contextual carrier exclusions and the two inverse-domain witnesses for each concrete endpoint x and y.

Use the generated BMark cases for s.

1. In the $\mathsf{BMark.b0}$ branch, the selected endpoint is x. The rows $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, x)) \to \bot$, $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(x, r)) \to \bot$, $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, x))$, and $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x, r))$ transport along this endpoint equality.
2. In the $\mathsf{BMark.b1}$ branch, the selected endpoint is y. The corresponding four rows for $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{ap}$ $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(y, r))$, $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, y))$, and $\mathsf{fieldSingletonEmptyNonZ}$ transport along this endpoint equality.

The two generated selector branches exhaust BMark, so the selected endpoint carries both one-sided contextual carrier exclusions and both inverse-domain witnesses for arbitrary context $r : \mathsf{BHist}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector contextual nonzero package`

Per-object template: traditional target selected bracketing contextual nonzero package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, μ^u_D, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier singleton-empty carrier and nonzero predicates; stability obligations bracketing-complete contextual carrier and inverse-domain rows; ledger selected endpoint and context side; exactness predicate selector-total noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target selector-indexed bracketing exit subpackage for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, μ^u_D, R^u_D, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D; stability obligations the bracketing-complete certificate plus selector coverage and selector exclusion; ledger the selected endpoint pair (s, t) and contexts (L, R); exactness predicate total selected-endpoint coverage and noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target RatUp FieldUp algebra hierarchy paper package; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier the concrete RatUp history rows and denominator

endpoints; classifier the ten-field rows, two-sided product classifier, nonzero transport, five-factor ledger, singleton-classifier exclusions, and bracketing-complete classifier package; stability obligations the cited concrete exit package rows; ledger the assembled $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ hierarchy certificate; core theorem Definition 45.616; exactness predicate concrete hierarchy row coverage without schematic operation parameters; standard bridge deferred; status ProofTarget.

Lemma 45.625 (RatUp FieldUp bracketing endpoint selector exhaustion)**.** *For histories $h, k, \ell : \mathsf{BHist}$, set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. For every pair of selectors $s, t : \mathsf{BMark}$, the selected endpoint pair $(E_s(h, k, \ell), E_t(h, k, \ell))$ is one of (x, x), (x, y), (y, x), and (y, y), with the corresponding displayed equalities between $E_s(h, k, \ell)$, $E_t(h, k, \ell)$ and the selected endpoints x, y.*

Proof. Apply BMark_generated_cases to s and then to t. This gives four closed generated selector pairs.

1. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b0}$, unfold Definition 45.606. Then $E_s(h, k, \ell) = x$ and $E_t(h, k, \ell) = x$, so the selected pair is (x, x).
2. If $s = \mathsf{BMark.b0}$ and $t = \mathsf{BMark.b1}$, unfold Definition 45.606. Then $E_s(h, k, \ell) = x$ and $E_t(h, k, \ell) = y$, so the selected pair is (x, y).
3. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b0}$, unfold Definition 45.606. Then $E_s(h, k, \ell) = y$ and $E_t(h, k, \ell) = x$, so the selected pair is (y, x).
4. If $s = \mathsf{BMark.b1}$ and $t = \mathsf{BMark.b1}$, unfold Definition 45.606. Then $E_s(h, k, \ell) = y$ and $E_t(h, k, \ell) = y$, so the selected pair is (y, y).

These four branches exhaust the generated BMark selectors, so no other selected endpoint pair is available. □

This case-exhaustion lemma is consumed by Definition 45.609, Definition 45.610, and Definition 45.629.

Theorem 45.626 (RatUp FieldUp selector-complete four-pair coverage)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, set $x := \mu^u_D(\mu^u_D(h, k), \ell)$ and $y := \mu^u_D(h, \mu^u_D(k, \ell))$. Let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the following two assertions are equivalent.*

1. *For every $s, t : \mathsf{BMark}$ and every $L, R : \mathsf{BHist}$, both $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$ and*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h, k, \ell)), \mathsf{append}(R, E_t(h, k, \ell))) \to \bot$$

   hold.

2. *The four explicit endpoint rows all hold: for each ordered pair (a, b) among (x, x), (x, y), (y, x), and (y, y), both $R^u_D(a, b)$ and, for every $L, R : \mathsf{BHist}$,*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, a), \mathsf{append}(R, b)) \to \bot$$

   hold.

Proof. For the forward direction, assume the selector-indexed assertion. Specialize it to the selector pairs (BMark.b0, BMark.b0), (BMark.b0, BMark.b1), (BMark.b1, BMark.b0), and (BMark.b1, BMark.b1), with arbitrary contexts $L, R : \mathsf{BHist}$.

Unfold Definition 45.606. The four specializations identify the selected endpoint pairs with (x, x), (x, y), (y, x), and (y, y) respectively, giving the four explicit denominator classifier rows and the four contextual singleton-classifier rejections.

For the reverse direction, assume the four explicit endpoint rows. Let $s, t : \mathsf{BMark}$ and $L, R : \mathsf{BHist}$ be arbitrary. Apply $\mathsf{BMark_generated_cases}$ to s and then to t.

1. In the $(\mathsf{BMark.b0}, \mathsf{BMark.b0})$ branch, unfolding the selector gives $(E_s(h, k, \ell), E_t(h, k, \ell)) = (x, x)$. Use the (x, x) row.
2. In the $(\mathsf{BMark.b0}, \mathsf{BMark.b1})$ branch, unfolding the selector gives $(E_s(h, k, \ell), E_t(h, k, \ell)) = (x, y)$. Use the (x, y) row.
3. In the $(\mathsf{BMark.b1}, \mathsf{BMark.b0})$ branch, unfolding the selector gives $(E_s(h, k, \ell), E_t(h, k, \ell)) = (y, x)$. Use the (y, x) row.
4. In the $(\mathsf{BMark.b1}, \mathsf{BMark.b1})$ branch, unfolding the selector gives $(E_s(h, k, \ell), E_t(h, k, \ell)) = (y, y)$. Use the (y, y) row.

The generated cases exhaust s and t, so the selector-indexed denominator classifier law and the uniform contextual singleton-classifier rejection follow for all selected pairs. □

Per-object template: finite selector coverage for the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ denominator row; BEDC source BHist, BMark, μ^u_D, R^u_D, E_s, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D; stability obligation the four generated endpoint rows; ledger the explicit pairs (x, x), (x, y), (y, x), and (y, y); exactness predicate equality between selector-indexed laws and finite endpoint coverage; standard bridge deferred; status $\mathsf{ProofTarget}$.

Definition 45.627 (RatUp FieldUp selector-complete denominator instance)**.** *For fixed histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, $C^u_D(\ell)$, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$, define the selector-complete denominator instance to be the denominator coordinate of $\mathsf{RatUpFieldSource}_F$ displayed as $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, with the concrete BMark-indexed package whose carrier predicate is C^u_D, whose selected endpoint is the map $E_s(h, k, \ell)$ of Definition 45.606, whose endpoint classifier is R^u_D on selected endpoint pairs, and whose contextual obstruction row is the singleton-empty classifier on*

$$\mathsf{append}(L, E_s(h, k, \ell))$$

and

$$\mathsf{append}(R, E_t(h, k, \ell))$$

for arbitrary selectors $s, t : \mathsf{BMark}$ and arbitrary contexts $L, R : \mathsf{BHist}$. Thus the instance packages the denominator classifier coverage and the contextual singleton obstruction on the same selected endpoint interface.

Theorem 45.628 (RatUp FieldUp selector-complete denominator well-formed exit slice)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then for all selectors $s, t : \mathsf{BMark}$ and all contexts $L, R : \mathsf{BHist}$, the selector-complete denominator instance of Definition 45.627 supplies the following three-field row package over the denominator coordinate of $\mathsf{RatUpFieldSource}_F$:*

1. *selected endpoint carrier coverage: $C^u_D(E_s(h, k, \ell))$ and $C^u_D(E_t(h, k, \ell))$;*
2. *denominator classifier coverage: $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$;*

3. contextual singleton-classifier noncollapse:

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)), \mathsf{append}(R, E_t(h,k,\ell))) \to \bot.$$

These rows identify Definition 45.627 as the selector-indexed denominator row family of Definition 45.617.

Proof. First apply Definition 45.610 to the concrete denominator tuple, the three carrier hypotheses, and $\mathsf{ThreeSupp}^u_D(h,k,\ell)$. This gives the selected denominator classifier row $R^u_D(E_s(h,k,\ell), E_t(h,k,\ell))$ for the arbitrary selectors $s,t : \mathsf{BMark}$.

Next set $x := \mu^u_D(\mu^u_D(h,k),\ell)$ and $y := \mu^u_D(h, \mu^u_D(k,\ell))$. Apply Definition 45.605 to the same source data and support witness. It supplies $C^u_D(x)$ and $C^u_D(y)$.

Use the two generated BMark cases for s. In the $\mathsf{BMark.b0}$ case, $E_s(h,k,\ell)$ is x; in the $\mathsf{BMark.b1}$ case, $E_s(h,k,\ell)$ is y. The supplied carrier rows therefore give $C^u_D(E_s(h,k,\ell))$. The same case split for t gives $C^u_D(E_t(h,k,\ell))$.

Finally apply Definition 45.609 to the same selected endpoints and the arbitrary contexts $L, R : \mathsf{BHist}$. This is the contextual singleton-classifier noncollapse row.

Unfold Definition 45.627. Its carrier predicate is C^u_D, its selected endpoint map is $s \mapsto E_s(h,k,\ell)$, its classifier row is R^u_D on selected endpoint pairs, and its obstruction row is the displayed singleton-empty classifier. Thus the carrier, classifier, and noncollapse rows are exactly the selector-indexed denominator row family used by Definition 45.617 over $\mathsf{RatUpFieldSource}_F$. □

Theorem 45.629 (RatUp FieldUp selector-complete denominator laws)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h,k,\ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h,k,\ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then for all selectors $s,t : \mathsf{BMark}$ and all contexts $L,R : \mathsf{BHist}$, the selector-complete denominator instance of Definition 45.627 supplies $C^u_D(E_s(h,k,\ell))$, $C^u_D(E_t(h,k,\ell))$, $R^u_D(E_s(h,k,\ell), E_t(h,k,\ell))$, and the contextual obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)), \mathsf{append}(R, E_t(h,k,\ell))) \to \bot.$$

Moreover, specializing the row to (s,s) with $L = R = \mathsf{e}$ gives $\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))$, and specializing the row to (t,t) with $L = R = \mathsf{e}$ gives $\mathsf{fieldSingletonEmptyNonZero}(E_t(h,k,\ell))$.

Proof. Apply Definition 45.628 to the same concrete denominator tuple, the three carrier hypotheses, the support witness, the selectors $s,t : \mathsf{BMark}$, and the contexts $L,R : \mathsf{BHist}$. This supplies the two selected carrier rows, the selected denominator classifier row, and the contextual singleton-empty obstruction.

For the first nonzero clause, specialize the same well-formed row to the selector pair (s,s) and to contexts $L = R = \mathsf{e}$. If a singleton-empty witness for $E_s(h,k,\ell)$ were present, the empty-left append unit identifies it with the obstructed classifier on

$$\mathsf{append}(\mathsf{e}, E_s(h,k,\ell)).$$

The specialized obstruction rejects that witness, and Definition 45.29 repackages the rejection as $\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))$.

For the second nonzero clause, repeat the specialization with the selector pair (t,t) and contexts $L = R = \mathsf{e}$. The empty-left append unit sends any singleton-empty witness for $E_t(h,k,\ell)$ to the corresponding obstructed classifier on

$$\mathsf{append}(\mathsf{e}, E_t(h,k,\ell)).$$

The specialized obstruction and Definition 45.29 give $\mathsf{fieldSingletonEmptyNonZero}(E_t(h,k,\ell))$. □

Theorem 45.630 (RatUp FieldUp selector-complete denominator strict support package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all selectors* $s, t : \mathsf{BMark}$ *the selector-complete denominator instance of Definition 45.627 supplies the strict-support endpoint package*

$$R^u_D(E_s(h, k, \ell), E_t(h, k, \ell)) \wedge \mathsf{fieldSingletonEmptyNonZero}(E_s(h, k, \ell)) \wedge \mathsf{fieldSingletonEmptyNonZero}(E_t(h, k, \ell)).$$

Proof. Apply Definition 45.629 to the same concrete denominator tuple, carrier hypotheses, support witness, and selectors $s, t : \mathsf{BMark}$. Its classifier row gives $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$.

For the left strict-support endpoint, specialize the obstruction row of Definition 45.629 to the selector pair (s, s) and to contexts $L = R = \mathsf{e}$. Assume a forbidden witness $\mathsf{fieldSingletonEmptyClassifier}(E_s(h, k, \ell), \mathsf{e})$. The empty-left append unit transports this witness to a singleton-empty classifier on $\mathsf{append}(\mathsf{e}, E_s(h, k, \ell))$ against itself, contradicting the specialized obstruction. By Definition 45.29, this is exactly fieldSingletonEmptyNonZ

For the right strict-support endpoint, repeat the same argument with the selector pair (t, t) and contexts $L = R = \mathsf{e}$. A forbidden classifier $\mathsf{fieldSingletonEmptyClassifier}(E_t(h, k, \ell), \mathsf{e})$ transports along the empty-left append unit to the obstructed classifier on $\mathsf{append}(\mathsf{e}, E_t(h, k, \ell))$ against itself. The resulting contradiction is the required $\mathsf{fieldSingletonEmptyNonZero}(E_t(h, k, \ell))$.

Combining the classifier row with the two singleton-empty complement witnesses gives the displayed strict-support endpoint package. □

Per-object template: traditional target selector-complete strict denominator support for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, C^u_D, R^u_D, E_s, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D; stability obligations selector coverage and singleton-empty noncollapse; ledger selected endpoint pair; exactness predicate denominator coverage plus strict-support endpoints; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.631 (RatUp FieldUp selector-complete denominator swap package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all selectors* $s, t : \mathsf{BMark}$ *and all contexts* $L, R : \mathsf{BHist}$*, the selector-complete denominator instance of Definition 45.627 supplies the symmetric selector/context row package:*

1. *selected endpoint carrier coverage:* $C^u_D(E_s(h, k, \ell))$ *and* $C^u_D(E_t(h, k, \ell))$*;*
2. *denominator classifier coverage in both selector orientations:* $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$ *and* $R^u_D(E_t(h, k, \ell), E_s(h, k, \ell))$*;*
3. *strict-support witnesses:* $\mathsf{fieldSingletonEmptyNonZero}(E_s(h, k, \ell))$ *and* $\mathsf{fieldSingletonEmptyNonZero}(E_t(h, k, \ell))$*;*
4. *contextual singleton-classifier noncollapse in both selector/context orientations:*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h, k, \ell)), \mathsf{append}(R, E_t(h, k, \ell))) \to \bot$$

and

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(R, E_t(h, k, \ell)), \mathsf{append}(L, E_s(h, k, \ell))) \to \bot.$$

Thus the concrete denominator row consumed by the exit object is available in the displayed orientation and in the swapped selector/context orientation.

Proof. Apply Definition 45.628 to the selectors $s, t : \mathsf{BMark}$ and contexts $L, R : \mathsf{BHist}$. This gives carrier coverage for $E_s(h, k, \ell)$ and $E_t(h, k, \ell)$, the classifier row $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$, and the first contextual singleton-empty obstruction.

Apply the same theorem again to the selectors $t, s : \mathsf{BMark}$ and contexts $R, L : \mathsf{BHist}$. The carrier rows agree with the same two selected endpoints, while the classifier and obstruction rows are now $R^u_D(E_t(h, k, \ell), E_s(h, k, \ell))$ and the swapped contextual singleton-empty obstruction.

Finally apply Definition 45.630 to the selector pair (s, t). Its strict-support endpoint package supplies $\mathsf{fieldSingletonEmptyNonZero}(E_s(h, k, \ell))$ and $\mathsf{fieldSingletonEmptyNonZero}(E_t(h, k, \ell))$.

Collecting the rows from the two selector-complete denominator law specializations and the strict-support package gives exactly the symmetric selector/context row package displayed above. □

Per-object template: traditional target selector-complete denominator swap package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, the denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, E_s, and $\mathsf{ThreeSupp}^u_D$; carrier C^u_D; classifier R^u_D plus the singleton-empty obstruction; stability obligations selector-complete row coverage and strict-support endpoints; ledger swapped selector/context orientation; exactness predicate symmetric selector-row coverage; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.632 (RatUp FieldUp selector-complete exit package)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the selector-complete denominator instance of Definition 45.627 supplies a concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit-package row over $\mathsf{RatUpFieldSource}_F$: for every $s, t : \mathsf{BMark}$ and every $L, R : \mathsf{BHist}$ it provides selected endpoint carrier coverage, the denominator classifier row $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$, and the contextual singleton-classifier noncollapse row*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h, k, \ell)), \mathsf{append}(R, E_t(h, k, \ell))) \to \bot.$$

This selector row is one of the concrete denominator rows consumed by Definition 45.616, and it is read in the denominator coordinate of $\mathsf{RatUpFieldSource}_F$. No schematic mul, add, or neg parameter enters the row.

Proof. Apply Definition 45.629 to the concrete denominator coordinate of $\mathsf{RatUpFieldSource}_F$, the three carrier hypotheses, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This gives, for arbitrary $s, t : \mathsf{BMark}$ and $L, R : \mathsf{BHist}$, the selected endpoint carrier rows, the selected endpoint classifier row, and the contextual singleton-empty obstruction.

Apply Definition 45.616. It records the denominator classifier coverage and singleton-classifier noncollapse rows as concrete rows inside the $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ hierarchy exit package over $\mathsf{RatUpFieldSource}_F$.

The row is therefore an admissible concrete exit-package component. Its data are the selected denominator endpoints inside BHist, the carrier coverage, the denominator classifier coverage, and contextual noncollapse over the source-aligned denominator coordinate, so no schematic mul, add, or neg operator parameter is introduced. □

Theorem 45.633 (RatUp FieldUp carrier-aware selector-complete concrete instance)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_s(h, k, \ell)$ be the endpoint selector of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then for every pair of selectors $s, t : \mathsf{BMark}$ and every pair of contexts $L, R : \mathsf{BHist}$, the selector-complete denominator instance supplies all carrier-aware rows*

1. $C^u_D(E_s(h, k, \ell))$;

2. $C^u_D(E_t(h,k,\ell))$;
3. $R^u_D(E_s(h,k,\ell), E_t(h,k,\ell))$;
4. $\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))$;
5. $\mathsf{fieldSingletonEmptyNonZero}(E_t(h,k,\ell))$;
6. $$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)), \mathsf{append}(R, E_t(h,k,\ell))) \to \bot.$$

Together these rows form a concrete carrier-aware and inverse-domain $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit row over* $\mathsf{RatUpFieldSource}_F$.

Proof. Apply Definition 45.623 to the concrete denominator tuple, the carrier hypotheses on h, k, ℓ, and the support witness $\mathsf{ThreeSupp}^u_D(h,k,\ell)$. For the chosen $s, t : \mathsf{BMark}$ and contexts $L, R : \mathsf{BHist}$, it supplies the two selected-endpoint carrier rows, the denominator-classifier row, and the contextual singleton-empty obstruction.

Apply Definition 45.630 to the same concrete tuple, carrier hypotheses, support witness, and selector pair. Its strict-support endpoint package supplies $\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))$ and $\mathsf{fieldSingletonEmptyNonZero}(E_t(h,k,\ell))$.

Apply Definition 45.632 to the same concrete tuple, carrier hypotheses, and support witness. This places the same selected denominator row inside the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package over $\mathsf{RatUpFieldSource}_F$.

The selector-complete denominator instance therefore carries both endpoint coverage and package membership: the two selected endpoints remain in C^u_D, the selected pair lies in R^u_D, both selected endpoints satisfy the singleton-empty nonzero condition, and every contextual singleton-empty classifier on the selected pair collapses to $\bot$. □

Theorem 45.634 (RatUp FieldUp selector-complete strict exit package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *let* $E_s(h,k,\ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then for every selector pair* $s, t : \mathsf{BMark}$ *and every context pair* $L, R : \mathsf{BHist}$ *the selector-complete denominator instance of Definition 45.627 supplies the full strict concrete exit row*

$$C^u_D(E_s(h,k,\ell)) \wedge C^u_D(E_t(h,k,\ell)) \wedge R^u_D(E_s(h,k,\ell), E_t(h,k,\ell)) \wedge (\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)), \mathsf{a}$$

Proof. Apply Definition 45.632 to the same concrete denominator tuple, carrier hypotheses, support witness, selectors $s, t : \mathsf{BMark}$, and contexts $L, R : \mathsf{BHist}$. It supplies the two selected endpoint carrier rows, the denominator classifier row, and the contextual singleton-classifier noncollapse row.

Apply Definition 45.630 to the same tuple, carrier hypotheses, support witness, and selector pair. It supplies the same denominator classifier row together with the two strict endpoint witnesses $\mathsf{fieldSingletonEmptyNonZero}(E_s(h,k,\ell))$ and $\mathsf{fieldSingletonEmptyNonZero}(E_t(h,k,\ell))$.

Use the classifier row only once, taking it from either package. Combining it with the two carrier rows, the contextual noncollapse row, and the strict endpoint witnesses gives exactly the displayed strict concrete exit row consumed in the denominator coordinate of $\mathsf{RatUpFieldSource}_F$. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup selector complete strict exit package`

Per-object template: traditional target selector-complete strict FieldUp denominator exit row; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, BMark, C^u_D, R^u_D, E_s, and $\mathsf{ThreeSupp}^u_D$; carrier selected endpoint coverage; classifier R^u_D plus singleton-empty obstruction; stability obligations carrier

coverage, denominator classifier coverage, contextual noncollapse, and strict endpoint nonzero witnesses; ledger selected endpoint pair and contexts; core theorem Definition 45.634; exactness predicate full strict row coverage; standard bridge deferred; status ProofTarget.

Theorem 45.635 (RatUp FieldUp five-factor product-map classifier exactness)**.** *For the concrete* $\mathsf{RatUpFieldSource}_F$ *operation rows over* BHist*, assume the proof-indexed inverse, multiplication congruence, unit, associativity, product nonzero exclusion, and apartness-conversion rows used by Definition 45.296. For five histories* $a, b, c, d, e : \mathsf{BHist}$*, set*

$$P_5 := \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(a, b), c), d), e).$$

Let $p_{abcde} : \mathsf{NonZero}_F(P_5)$ *be the product ledger witness, and let* R_5 *be the reversed inverse spine supplied by Definition 45.574. Then for every* $x, y : \mathsf{BHist}$*, multiplication by the same five-factor product is classifier-exact on both sides:*

$$\mathsf{mul}(x, P_5)\mathsf{hsamemul}(y, P_5) \iff x\mathsf{hsame}y$$

and

$$\mathsf{mul}(P_5, x)\mathsf{hsamemul}(P_5, y) \iff x\mathsf{hsame}y.$$

Proof. For the forward direction of the right-product clause, apply Definition 45.296 to the equation $\mathsf{mul}(x, P_5)\mathsf{hsamemul}(y, P_5)$ with shared target endpoint $\mathsf{mul}(y, P_5)$. It classifies x by the solved endpoint $\mathsf{mul}(\mathsf{mul}(y, P_5), R_5)$.

Apply the same theorem to the reflexive right-product equation for y at that same target endpoint. The two solved endpoints are identical, so symmetry and transitivity of hsame compare x and y.

For the reverse direction of the right-product clause, use multiplication congruence to transport $x\mathsf{hsame}y$ through $z \mapsto \mathsf{mul}(z, P_5)$. This gives $\mathsf{mul}(x, P_5)\mathsf{hsamemul}(y, P_5)$.

For the forward direction of the left-product clause, apply Definition 45.296 to the equation $\mathsf{mul}(P_5, x)\mathsf{hsamemul}(P_5, y)$ with shared target endpoint $\mathsf{mul}(P_5, y)$. The left equation solver classifies x by $\mathsf{mul}(R_5, \mathsf{mul}(P_5, y))$.

Apply the same left-product exactness clause to the reflexive equation for y at the same target endpoint. The solved endpoint is again identical, and symmetry plus transitivity of hsame yields $x\mathsf{hsame}y$.

For the reverse direction of the left-product clause, multiplication congruence transports $x\mathsf{hsame}y$ through $z \mapsto \mathsf{mul}(P_5, z)$, producing the displayed left classifier equality. □

Per-object template: traditional target finite five-factor product-map classifier exactness; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier BHist; classifier hsame; ledger p_{abcde} plus induced factor witnesses; core theorem Definition 45.635; exactness predicate the two displayed iff clauses; standard bridge deferred; status ProofTarget.

Theorem 45.636 (RatUp FieldUp selector-complete denominator row exhaustion)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. Fix contexts* $L, R : \mathsf{BHist}$ *and assume* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Set* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. Then the selected denominator row of Definition 45.627 is exhausted by exactly the following four ordered endpoint rows:*

1. (x, x)*, with carrier rows* $C^u_D(x)$ *and* $C^u_D(x)$*, classifier row* $R^u_D(x, x)$*, and obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, x)) \to \bot;$$

2. (x, y)*, with carrier rows* $C^u_D(x)$ *and* $C^u_D(y)$*, classifier row* $R^u_D(x, y)$*, and obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x), \mathsf{append}(R, y)) \to \bot;$$

3. (y, x)*, with carrier rows* $C^u_D(y)$ *and* $C^u_D(x)$*, classifier row* $R^u_D(y, x)$*, and obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, x)) \to \bot;$$

4. (y, y)*, with carrier rows* $C^u_D(y)$ *and* $C^u_D(y)$*, classifier row* $R^u_D(y, y)$*, and obstruction*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, y)) \to \bot.$$

For every selector pair $s, t : \mathsf{BMark}$*, the row selected by* $(E_s(h, k, \ell), E_t(h, k, \ell))$ *is one of these four rows, and its two carrier rows,* R^u_D *classifier row, and singleton-empty contextual obstruction are the corresponding rows supplied by the selector-complete* $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *exit package. Conversely, the four displayed ordered endpoint rows are realized by the selector pairs* $(\mathsf{BMark.b0}, \mathsf{BMark.b0})$*,* $(\mathsf{BMark.b0}, \mathsf{BMark.b1})$*,* $(\mathsf{BMark.b1}, \mathsf{BMark.b0})$*, and* $(\mathsf{BMark.b1}, \mathsf{BMark.b1})$*.*

Proof. Apply Lemma 45.625 to the arbitrary selectors $s, t : \mathsf{BMark}$. It gives the four generated selector branches and identifies their selected endpoint pairs as (x, x), (x, y), (y, x), and (y, y).

Apply Theorem 45.628 to the concrete denominator tuple, the three carrier hypotheses, the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$, the selected pair s, t, and the contexts L, R. This supplies the two carrier rows, the R^u_D classifier row, and the singleton-empty contextual obstruction for the selected denominator row.

Apply Theorem 45.632 to the same data. The classifier row and the contextual obstruction are recorded as concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ exit-package rows over $\mathsf{RatUpFieldSource}_F$, so each of the four cases is an admissible row of the concrete exit package.

It remains to account for the converse realization. Specialize Lemma 45.625 to the four selector pairs $(\mathsf{BMark.b0}, \mathsf{BMark.b0})$, $(\mathsf{BMark.b0}, \mathsf{BMark.b1})$, $(\mathsf{BMark.b1}, \mathsf{BMark.b0})$, and $(\mathsf{BMark.b1}, \mathsf{BMark.b1})$. Unfolding Definition 45.606 gives respectively the endpoint rows (x, x), (x, y), (y, x), and (y, y), and the preceding well-formedness and exit-package applications attach the corresponding four rows displayed in the statement. □

Theorem 45.637 (RatUp FieldUp selector-complete row stability)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and two denominator triples* $h, k, \ell, h', k', \ell' : \mathsf{BHist}$*, assume* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*,* $C^u_D(h')$*,* $C^u_D(k')$*, and* $C^u_D(\ell')$*. Let* $E_s(h, k, \ell)$ *and* $E_s(h', k', \ell')$ *be the endpoint selector of Definition 45.606 on the two triples. If*

$$\mathsf{ThreeSupp}^u_D(h, k, \ell) \quad \textit{and} \quad \mathsf{ThreeSupp}^u_D(h', k', \ell')$$

hold, and if

$$R^u_D(h, h'), \quad R^u_D(k, k'), \quad R^u_D(\ell, \ell')$$

hold, then for every selector pair $s, t : \mathsf{BMark}$ *and all contexts* $L, R, L', R' : \mathsf{BHist}$ *with* $\mathsf{hsame}(L, L')$ *and* $\mathsf{hsame}(R, R')$*, the selected endpoints are denominator-classifier stable:*

$$R^u_D(E_s(h, k, \ell), E_s(h', k', \ell')) \quad \textit{and} \quad R^u_D(E_t(h, k, \ell), E_t(h', k', \ell')).$$

The primed selected row also supplies the same three-field row package as Definition 45.628:

1. $C^u_D(E_s(h', k', \ell'))$ *and* $C^u_D(E_t(h', k', \ell'))$*;*

2. $R^u_D(E_s(h', k', \ell'), E_t(h', k', \ell'))$;

3.
$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L', E_s(h', k', \ell')), \mathsf{append}(R', E_t(h', k', \ell'))) \to \bot.$$

Proof. Use the generated case split on $s : \mathsf{BMark}$. In the $\mathsf{BMark.b0}$ branch, the selected endpoint is the left-nested denominator $\mu^u_D(\mu^u_D(h, k), \ell)$, and the primed endpoint is $\mu^u_D(\mu^u_D(h', k'), \ell')$. Apply Definition 45.131 first to the pair h, h' and k, k', and then to the resulting left-nested pair and ℓ, ℓ'. This gives the displayed classifier relation for the s-selected endpoints.

In the $\mathsf{BMark.b1}$ branch, the selected endpoint is the right-nested denominator $\mu^u_D(h, \mu^u_D(k, \ell))$, and the primed endpoint is $\mu^u_D(h', \mu^u_D(k', \ell'))$. Apply the same concrete binary classifier congruence first to k, k' and ℓ, ℓ', and then to h, h' and the resulting right-nested pair. This gives the second generated endpoint comparison. Repeating the same two-branch argument for t gives the t-selected endpoint comparison.

Finally apply Definition 45.628 to the primed triple, its three carrier hypotheses, the support witness $\mathsf{ThreeSupp}^u_D(h', k', \ell')$, the selectors s, t, and the contexts L', R'. It supplies the primed selected endpoint carrier rows, the primed selected denominator-classifier row, and the displayed contextual singleton-classifier noncollapse. The assumptions $\mathsf{hsame}(L, L')$ and $\mathsf{hsame}(R, R')$ identify these contextual row slots with the same left and right context classes as the unprimed selected row. □

Theorem 45.638 (RatUp FieldUp selector-complete hierarchy exit witness)**.** *For the concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *and the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, let* $h, k, \ell, h', k', \ell' : \mathsf{BHist}$ *carry the six denominator carrier hypotheses* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*,* $C^u_D(h')$*,* $C^u_D(k')$*, and* $C^u_D(\ell')$*. Let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606, and set* $x := \mu^u_D(\mu^u_D(h, k), \ell)$ *and* $y := \mu^u_D(h, \mu^u_D(k, \ell))$*. If*

$$\mathsf{ThreeSupp}^u_D(h, k, \ell) \quad \textit{and} \quad \mathsf{ThreeSupp}^u_D(h', k', \ell')$$

hold, and if

$$R^u_D(h, h'), \quad R^u_D(k, k'), \quad R^u_D(\ell, \ell')$$

hold, then the concrete $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ *paper certificate contains the selector-complete denominator row family as a stable concrete component:*

1. *the algebra-hierarchy exit package over* $\mathsf{RatUpFieldSource}_F$*;*
2. *the selector-complete exit package for every* $s, t : \mathsf{BMark}$ *and* $L, R : \mathsf{BHist}$*;*
3. *the strict selector exit package for the same selector and context data;*
4. *the four ordered endpoint rows* (x, x)*,* (x, y)*,* (y, x)*, and* (y, y)*, each with its two* C^u_D *carrier rows, its* R^u_D *classifier row, and its singleton-empty contextual obstruction;*
5. *the row-stability transport for every* $s, t : \mathsf{BMark}$ *and* $L, R, L', R' : \mathsf{BHist}$ *with* $\mathsf{hsame}(L, L')$ *and* $\mathsf{hsame}(R, R')$*.*

Thus the concrete hierarchy exit witness records both the selector-complete row coverage and its transport under R^u_D*-related denominator triples and* hsame*-related contexts.*

Proof. First apply Definition 45.616. This fixes the ambient $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$ algebra-hierarchy package over $\mathsf{RatUpFieldSource}_F$ and supplies the paper-side certificate into which the denominator rows are placed.

Next apply Definition 45.632 and Definition 45.634 to the same concrete denominator tuple, carrier hypotheses, support witness, selectors, and contexts. These two sites supply the selector-indexed carrier rows, denominator classifier row, contextual singleton-empty obstruction, and strict endpoint witnesses.

Then apply Definition 45.636. It identifies the selector-indexed rows with the four ordered endpoint rows (x, x), (x, y), (y, x), and (y, y), and records the converse realization by the four selector pairs over BMark.

Finally apply Definition 45.637 to the two denominator triples, the R^u_D comparisons, and the hsame-related contexts. It transports selected endpoints and the primed row package through the same concrete denominator classifier, so the selector rows remain stable inside the hierarchy exit certificate. □

Per-object template: traditional target FieldUp(RatUp) selector-complete hierarchy exit certificate; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, BMark, and the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; carrier C^u_D; classifier R^u_D plus singleton-empty obstruction; stability obligations selector exhaustion and row stability; ledger selected denominator endpoints and contexts; core theorem Definition 45.638; exactness predicate concrete selector-complete row coverage; standard bridge deferred; status $\mathsf{ProofTarget}$.

Per-object template: traditional target selector-complete FieldUp denominator row; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier C^u_D with selected endpoint coverage and $\mathsf{ThreeSupp}^u_D$; classifier R^u_D plus singleton-empty obstruction; stability obligations carrier coverage, denominator classifier coverage, and contextual noncollapse over the source-aligned denominator coordinate; ledger selected denominator endpoints; core theorem list Definition 45.632 and Definition 45.637; exactness predicate selected endpoint carrier coverage, denominator classifier coverage, and contextual noncollapse; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.639 (RatUp FieldUp contextual action composition support laws)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $p, q, p', q', h, l, r, l', r' : \mathsf{BHist}$. Suppose $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, $p'\mathsf{hsamee}$, and $q'\mathsf{hsamee}$, and suppose $C^u_D(h)$, $C^u_D(l)$, $C^u_D(r)$, $C^u_D(l')$, and $C^u_D(r')$. Then the nested contextual action is carried:*

$$C^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))).$$

It is denominator-classified by the single contextual action with collected append contexts:

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',\mathsf{append}(l',l),\mathsf{append}(r,r')}(h)).$$

Its strict support is exactly the five visible support sites:

$$N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \iff N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r').$$

Proof. Apply Definition 45.467 to the empty-endpoint nested actions. Its classifier and support clauses identify the two-layer action with the corresponding collected action once the inner and outer carried contexts are available.

The carrier inputs are supplied by denominator-unit carrier closure under append. First close l with h and then with r; transport the result along the two endpoint equalities $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$ to get the carried inner action. Close l' with that carried inner action and then with r'; transport along $p'\mathsf{hsamee}$ and $q'\mathsf{hsamee}$ to get the carried nested action.

The same append-closure rows also carry $\mathsf{append}(l', l)$ and $\mathsf{append}(r, r')$, so the collected single action is a legitimate denominator action. Carrier transport along hsame keeps the concrete action endpoints inside C^u_D throughout the comparison.

For strict support, apply bilateral support exactness first to the inner action: its support is read as $N^u_D(l) \vee N^u_D(h) \vee N^u_D(r)$. Apply the same exactness theorem to the outer action and substitute the inner readback. The finite disjunction reassociates to $N^u_D(l') \vee N^u_D(l) \vee N^u_D(h) \vee N^u_D(r) \vee N^u_D(r')$, which is the displayed support law. □

Lean4 ✓ `BEDC.Derived.FieldUp.field rat denominator contextual action composition support laws`

Theorem 45.640 (RatUp FieldUp contextual action pair full transport package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $p, q, p', q', h, l, r, s, t, l', r', s', t' : \mathsf{BHist}$*. Suppose* $p\mathsf{hsamee}$*,* $q\mathsf{hsamee}$*,* $p'\mathsf{hsamee}$*, and* $q'\mathsf{hsamee}$*; suppose* $C^u_D(h)$*; suppose that* (l, r)*,* (s, t)*,* (l', r')*, and* (s', t') *are carried denominator context pairs; and suppose the four component classifiers* $R^u_D(l, s)$*,* $R^u_D(r, t)$*,* $R^u_D(l', s')$*, and* $R^u_D(r', t')$*. Then the two nested contextual actions are denominator-classified and have equivalent strict support:*

$$R^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h)), \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h))) \quad \wedge \quad (N^u_D(\Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))) \Longleftrightarrow N^u_D(\Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h)))).$$

Proof. The carried-pair hypotheses and the four component classifiers assemble the two context-pair classifier rows $\mathsf{PairClass}^u_D((l, r), (s, t))$ and $\mathsf{PairClass}^u_D((l', r'), (s', t'))$.

Apply Definition 45.496 with these two pair classifiers and the empty endpoint hypotheses. This gives the first conjunct, namely denominator classification of the two transported two-layer contextual actions.

Apply Definition 45.495 with the same carried pairs, pair classifiers, core carrier row, and empty endpoint hypotheses. This gives the second conjunct, namely equivalence of strict support for the same two nested actions.

The two cited conclusions have identical source data, so conjunction introduction gives the displayed full transport package. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction pair support transport iff`

Theorem 45.641 (RatUp FieldUp contextual action nested pair action coherence)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $h, l_0, r_0, l_1, r_1, l_2, r_2 : \mathsf{BHist}$*. Suppose* $C^u_D(h)$*,* $C^u_D(l_0)$*,* $C^u_D(r_0)$*,* $C^u_D(l_1)$*,* $C^u_D(r_1)$*,* $C^u_D(l_2)$*, and* $C^u_D(r_2)$*. Then the threefold empty-endpoint action is carried:*

$$C^u_D(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

It is denominator-classified by the single collected action:

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))), \Gamma_{\mathsf{e},\mathsf{e},\mathsf{append}(l_2,\mathsf{append}(l_1,l_0)),\mathsf{append}(\mathsf{append}(r_0,r_1),r_2)}(h)).$$

Proof. First apply Definition 45.639 to the inner two empty-endpoint actions. This carries the nested action through the first two denominator context pairs and classifies it against the collected pair $(\mathsf{append}(l_1, l_0), \mathsf{append}(r_0, r_1))$.

Append closure for C^u_D carries $\mathsf{append}(l_1, l_0)$ and $\mathsf{append}(r_0, r_1)$. Apply the same two-layer composition support law again, now with the outer pair (l_2, r_2) and the collected inner pair. This gives the carried target action and the classifier to $\Gamma_{\mathsf{e},\mathsf{e},\mathsf{append}(l_2,\mathsf{append}(l_1,l_0)),\mathsf{append}(\mathsf{append}(r_0,r_1),r_2)}(h)$.

Finally, transport the outer action along the inner classifier and compose the two classifier rows by transitivity of R^u_D. The carried row from the second application gives the displayed carrier statement, and the composed classifier is exactly the displayed denominator-classification statement. □

Lean4 ✓ BEDC.Derived.FieldUp.RatDenomUnitContextualAction nested pair action coherence

Theorem 45.642 (RatUp FieldUp contextual action three-pair reverse-order coherence)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $h, l_0, r_0, l_1, r_1, l_2, r_2 : \mathsf{BHist}$*. Suppose* $C^u_D(h)$*,* $C^u_D(l_0)$*,* $C^u_D(r_0)$*,* $C^u_D(l_1)$*,* $C^u_D(r_1)$*,* $C^u_D(l_2)$*, and* $C^u_D(r_2)$*. Then the empty-endpoint contextual actions obtained by applying the three carried denominator context pairs in forward and reverse order are denominator-classified:*

$$R^u_D(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))), \Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(h)))).$$

Their strict-support predicates are equivalent:

$$N^u_D(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))) \Longleftrightarrow N^u_D(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(h)))).$$

Proof. First move the carried pair (l_0, r_0) past (l_1, r_1) while the outer (l_2, r_2) action is fixed. Apply Definition 45.497 to the inner empty-endpoint pair, and use Definition 45.639 to supply the carried intermediate action needed by the outer context.

Next move (l_2, r_2) past (l_0, r_0) across the carried middle action. The same two-pair commutation theorem gives the classifier for the exposed pair swap, and the composition support laws keep the middle contextual action inside C^u_D. Classifier transitivity composes this row with the first one.

Finally move (l_1, r_1) past (l_2, r_2) under the outer (l_0, r_0) action. Another application of the empty-pair commutation theorem, followed by the same composition support laws, gives the third classifier row. Transitivity of R^u_D along the three displayed classifier rows yields the reverse-order classifier.

For strict support, apply Definition 45.498 along each of the three pair swaps just described. The composition support laws provide the carried intermediate actions at every stage, and support transport along the same classifier chain repacks the three local equivalences into the displayed strict-support equivalence. □

Per-object template: traditional target order-independent denominator context application inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, empty endpoints, and Γ; carrier/context choices $h, l_0, r_0, l_1, r_1, l_2, r_2$ with all seven histories in C^u_D; stability obligations classifier coherence and strict-support transport; ledger the three carried context pairs (l_0, r_0), (l_1, r_1), and (l_2, r_2) and their application order; core theorem list Definition 45.497, Definition 45.498, and Definition 45.639; exactness predicate the displayed classifier plus support iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.643 (RatUp FieldUp contextual action threefold pair support transport)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $h, l_0, r_0, l_1, r_1, l_2, r_2, s_0, t_0, s_1, t_1, s_2, t_2 : \mathsf{BHist}$*. Suppose* $C^u_D(h)$ *and suppose each of* $l_0, r_0, l_1, r_1, l_2, r_2, s_0, t_0, s_1, t_1, s$ *lies in* C^u_D*. Suppose also that the denominator context pairs are classified componentwise:* $R^u_D(l_i, s_i)$ *and* $R^u_D(r_i, t_i)$ *for* $i = 0, 1, 2$*. Let*

$$H_l := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))$$

and

$$H_s := \Gamma_{\mathsf{e},\mathsf{e},s_2,t_2}(\Gamma_{\mathsf{e},\mathsf{e},s_1,t_1}(\Gamma_{\mathsf{e},\mathsf{e},s_0,t_0}(h))).$$

Then strict support is invariant under replacing the three classified context pairs:

$$N^u_D(H_l) \Longleftrightarrow N^u_D(H_s).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatDenomUnitContextualAction threefold pair support transport`

Proof. Apply Definition 45.469 to H_l. Its strict-support row reads $N_D^u(H_l)$ as the seven-term disjunction with context supports $N_D^u(l_2)$, $N_D^u(l_1)$, $N_D^u(l_0)$, $N_D^u(r_0)$, $N_D^u(r_1)$, and $N_D^u(r_2)$ around the shared middle term $N_D^u(h)$.

Apply the same threefold normalization theorem to H_s. The second readback has the same shape, with the corresponding s_i and t_i support terms around the same middle term.

The three componentwise classified context pairs give three product-support transports. Use Definition 45.572 for the pairs (l_0, r_0) and (s_0, t_0), then for (l_1, r_1) and (s_1, t_1), and then for (l_2, r_2) and (s_2, t_2). Opening each product with Definition 45.571 identifies the six outer support alternatives on the two sides.

For the forward direction, read $N_D^u(H_l)$ through the first seven-term formula. Transport the selected context-support alternative through the corresponding classified pair, keep the shared $N_D^u(h)$ alternative fixed, and close the result through the second seven-term formula.

The reverse direction is symmetric. Read $N_D^u(H_s)$ through the second formula, transport the selected context-support alternative back through the same three classified pairs, keep the shared middle alternative fixed, and repack through the first seven-term formula. □

Theorem 45.644 (RatUp FieldUp threefold pair transported noncollapse package)**.** *For the concrete denominator tuple $(C_D^u, R_D^u, \mu_D^u, e_D^u, N_D^u)$ and the contextual action Γ of Definition 45.454, let $h, l_0, r_0, l_1, r_1, l_2, r_2, s_0, t_0, s_1, t_1, s_2, t_2 : \mathsf{BHist}$. Suppose $C_D^u(h)$, suppose each of $l_0, l_1, l_2, s_0, s_1, s_2$ lies in C_D^u, and suppose each of $r_0, r_1, r_2, t_0, t_1, t_2$ lies in C_D^u. Suppose also that $R_D^u(l_i, s_i)$ and $R_D^u(r_i, t_i)$ hold for $i = 0, 1, 2$. Let*

$$H_l := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))$$

and

$$H_s := \Gamma_{\mathsf{e},\mathsf{e},s_2,t_2}(\Gamma_{\mathsf{e},\mathsf{e},s_1,t_1}(\Gamma_{\mathsf{e},\mathsf{e},s_0,t_0}(h))).$$

If $N_D^u(H_l)$ holds, then the transported endpoint H_s carries the following noncollapse package:

1. *$N_D^u(H_s)$;*
2. $\mathsf{RatHistoryCarrier}(H_s)$*;*
3. $\mathsf{fieldSingletonEmptyCarrier}(H_s) \to \bot$*;*
4. $H_s\mathsf{hsamee} \to \bot$*;*
5. *for every $r : \mathsf{BHist}$, both contextual carrier exclusions $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, H_s)) \to \bot$ and $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(H_s, r)) \to \bot$;*
6. *for every $r : \mathsf{BHist}$, both inverse-domain witnesses $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, H_s))$ and $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(H_s, r))$;*
7. *for all $L, R : \mathsf{BHist}$, the contextual singleton-classifier exclusion*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, H_s), \mathsf{append}(R, H_s)) \to \bot.$$

Proof. First apply Definition 45.643 to the displayed context data. The forward direction sends the assumed $N_D^u(H_l)$ witness to $N_D^u(H_s)$.

Read the denominator tuple row from Definition 45.438. There $N_D^u(z) = \mathsf{RatHistoryCarrier}(z)$ for every denominator history z, so the transported support witness gives $\mathsf{RatHistoryCarrier}(H_s)$.

The direct empty boundary follows from Definition 37.18. Any comparison $H_s$hsamee contradicts the recovered Rat carrier witness. A direct witness of $\mathsf{fieldSingletonEmptyCarrier}(H_s)$ unfolds to that same empty comparison, so it is rejected by the same boundary.

Fix an arbitrary context history $r : \mathsf{BHist}$. Apply Definition 45.101 with context r and Rat carrier endpoint H_s to reject $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, H_s))$. Apply Definition 45.102 with the same two histories in the opposite append order to reject $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(H_s, r))$.

Unfold $\mathsf{fieldSingletonEmptyNonZero}$ for the right-append endpoint as rejection of the singleton classifier against e. Any such classifier witness contains the source-carrier component $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{app}$ contradicting the right-append rejection. The left-append inverse-domain witness is identical, using the source-carrier component $\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(H_s, r))$.

Finally package $\mathsf{RatHistoryClassifier}(H_s, H_s)$ from the recovered Rat carrier witness on both sides and reflexivity of hsame. For arbitrary $L, R : \mathsf{BHist}$, apply Definition 45.104. It rejects the displayed singleton classifier on the two contextual copies of H_s. □

Per-object template: traditional target transported threefold denominator noncollapse for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, Γ, C^u_D, R^u_D, and N^u_D; carrier transported endpoint H_s; classifier R^u_D plus hsame boundary rows; ledger the three classified context pairs $(l_0, r_0) \mapsto (s_0, t_0)$, $(l_1, r_1) \mapsto (s_1, t_1)$, and $(l_2, r_2) \mapsto (s_2, t_2)$; core theorem Definition 45.644; exactness predicate endpoint strict support, Rat carrier readback, singleton-empty exclusion, contextual append exclusions, inverse-domain witnesses, and contextual singleton-classifier obstruction; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.645 (RatUp FieldUp four context-pair adjacent-swap closure)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $h, l_0, r_0, l_1, r_1, l_2, r_2, l_3, r_3 : \mathsf{BHist}$. Suppose $C^u_D(h)$ and suppose that (l_0, r_0), (l_1, r_1), (l_2, r_2), and (l_3, r_3) are carried denominator context pairs. Let*

$$H_{0123} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Let

$$H_{0213} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Then the middle-swap histories are denominator-classified:

$$R^u_D(H_{0123}, H_{0213}).$$

Their strict-support readbacks agree:

$$N^u_D(H_{0123}) \Longleftrightarrow N^u_D(H_{0213}).$$

Proof. First carry the action of (l_0, r_0) on h. Apply Definition 45.639 with empty endpoints and the carried pair data to keep $\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)$ inside C^u_D.

Next apply Definition 45.486 over that carried base, using (l_1, r_1) and (l_2, r_2) as the adjacent pair to be exchanged and (l_3, r_3) as the common outer pair. This gives the displayed classifier between H_{0123} and H_{0213}.

The contextual action pair transport rows carry the same classifier through the common outer context and preserve the strict-support predicate on transported denominator histories. Applying the support part of the adjacent-swap row, then reading it through the common outer transport, gives the displayed support equivalence. □

Per-object template: traditional target four-context denominator action coherence for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, C^u_D, R^u_D, μ^u_D, e, and Γ; carrier carried context pairs; classifier R^u_D; stability obligations carrier closure plus adjacent-swap transport; ledger the four carried pairs and acted denominator history; core theorem list Definition 45.642, Definition 45.486, and Definition 45.639; exactness predicate the displayed classifier and support iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.646 (RatUp FieldUp four context-pair adjacent-swap coverage)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and the contextual action Γ of Definition 45.454, let $h, l_0, r_0, l_1, r_1, l_2, r_2, l_3, r_3 : \mathsf{BHist}$. Suppose $C^u_D(h)$ and suppose that (l_0, r_0), (l_1, r_1), (l_2, r_2), and (l_3, r_3) are carried denominator context pairs. Let*

$$H_{0123} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Let

$$H_{1023} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(h)))).$$

Let

$$H_{0213} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Let

$$H_{0132} := \Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Then the three adjacent-generator rows from H_{0123} are classified:

$$R^u_D(H_{0123}, H_{1023}) \wedge R^u_D(H_{0123}, H_{0213}) \wedge R^u_D(H_{0123}, H_{0132}).$$

The corresponding strict-support readbacks agree:

$$N^u_D(H_{0123}) \Longleftrightarrow N^u_D(H_{1023}).$$

$$N^u_D(H_{0123}) \Longleftrightarrow N^u_D(H_{0213}).$$

$$N^u_D(H_{0123}) \Longleftrightarrow N^u_D(H_{0132}).$$

Proof. For the first adjacent generator, apply Definition 45.486 to the carried pairs (l_0, r_0), (l_1, r_1), and (l_2, r_2) over h. Then carry the result through the common outer context (l_3, r_3) using the contextual action pair transport classifier row Definition 45.500. The carrier premises needed for the outer action come from Definition 45.639.

For the middle adjacent generator, use Definition 45.645. It is exactly the displayed comparison between H_{0123} and H_{0213} under the same four carried context pairs.

For the last adjacent generator, first apply Definition 45.639 to carry the common two-step base $\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h))$. Then apply Definition 45.497 to the two outer empty-endpoint actions (l_2, r_2) and (l_3, r_3). This gives the classifier row between H_{0123} and H_{0132}.

The strict-support rows are obtained in parallel. For the first and middle generators, read the support part of the adjacent-swap row through the same contextual action pair transport support row Definition 45.499. For the last generator, apply Definition 45.498 over the carried two-step base. These three applications give the displayed strict-support equivalences. □

Per-object template: traditional target finite adjacent-generator coverage for the concrete FieldUp denominator action; BEDC source BHist, C^u_D, R^u_D, N^u_D, e, and Γ; carrier the four carried context pairs; classifier R^u_D; stability adjacent-swap classifier transport and strict-support transport; ledger the four carried pairs and selected adjacent generator; exactness the displayed three-row coverage; status $\mathsf{ProofTarget}$.

Theorem 45.647 (RatUp FieldUp four context-pair full reverse-order coherence)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let* $h, l_0, r_0, l_1, r_1, l_2, r_2, l_3, r_3 : \mathsf{BHist}$*. Suppose* $C^u_D(h)$ *and suppose that* (l_0, r_0)*,* (l_1, r_1)*,* (l_2, r_2)*, and* (l_3, r_3) *are carried denominator context pairs. Let*

$$H_{0123} := \Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(h)))).$$

Let

$$H_{3210} := \Gamma_{\mathsf{e},\mathsf{e},l_0,r_0}(\Gamma_{\mathsf{e},\mathsf{e},l_1,r_1}(\Gamma_{\mathsf{e},\mathsf{e},l_2,r_2}(\Gamma_{\mathsf{e},\mathsf{e},l_3,r_3}(h)))).$$

Then the forward and reverse four-context histories are denominator-classified:

$$R^u_D(H_{0123}, H_{3210}).$$

Their strict-support readbacks agree:

$$N^u_D(H_{0123}) \Longleftrightarrow N^u_D(H_{3210}).$$

Proof. Introduce the six intermediate four-context histories with application orders 1023, 1203, 1230, 2130, 2310, and 3210. Each is obtained from the preceding order by one adjacent exchange in the carried context-pair ledger.

For the classifier spine, instantiate Definition 45.646 at the current ordering for the six adjacent exchanges $0123 \to 1023 \to 1203 \to 1230 \to 2130 \to 2310 \to 3210$.

The carrier premises at every stage are supplied by Definition 45.639, since each current base is an iterated Γ-action on the same carried history and the same four carried denominator context pairs.

Chain the six displayed R^u_D rows by transitivity of the concrete denominator classifier. The resulting composite row has endpoints H_{0123} and H_{3210}, giving the first conclusion.

For strict support, read the support part of each adjacent-swap coverage row at the same six current orderings. Compose the six equivalences in sequence, with the inverse direction of each equivalence used exactly when the selected adjacent row is traversed opposite to its displayed orientation. The composite equivalence has endpoints $N^u_D(H_{0123})$ and $N^u_D(H_{3210})$. □

Per-object template: traditional target reverse-order independence for the concrete FieldUp denominator action; BEDC source BHist, C^u_D, R^u_D, N^u_D, e, and Γ; carrier the four carried context pairs; ledger the four context pairs plus the six selected adjacent generators; core theorem list Definition 45.646 and contextual composition support; exactness the displayed reverse-order classifier and strict-support equivalence; status $\mathsf{ProofTarget}$.

Theorem 45.648 (Field singleton carrier two-sided context semantic name certificate)**.** *For histories* $L, R : \mathsf{BHist}$*, if* L *and* R *lie in the field singleton carrier, then the two-sided contextual carrier has a semantic name certificate whose source, pattern, and ledger predicates are that same contextual carrier, and whose classifier compares the two contextualized endpoints by the field singleton classifier:*

$$\mathsf{FieldSingletonCarrier}(L) \Rightarrow \mathsf{FieldSingletonCarrier}(R) \Rightarrow \mathsf{SemanticNameCert}(h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{app}$$

Proof. Let $C_{L,R}(h)$ denote the displayed two-sided contextual carrier. The inhabited point is $h = \mathsf{e}$: the carrier witnesses for L and R, together with $\mathsf{ehsamee}$, rebuild the nested empty append witness for $\mathsf{append}(L, \mathsf{append}(\mathsf{e}, R))$.

For reflexivity, symmetry, and transitivity of the classifier, unpack the displayed singleton classifier. Its endpoint carrier fields are precisely the instances of $C_{L,R}$ at the endpoints, and the comparison field is an hsame witness between the two contextualized histories. The three laws are therefore inherited from reflexivity, symmetry, and transitivity of hsame.

It remains to transport carrier membership along the classifier. Apply Definition 45.63 to identify $C_{L,R}(h)$ with the underlying singleton carrier of h, and use Definition 45.66 with the reflexive context comparisons for L and R to cancel the common two-sided context. The transported endpoint carrier repacks into the source, pattern, and ledger fields of the required SemanticNameCert. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldSingletonCarrier append context semanticNameCert`

Per-object template: traditional target contextual singleton boundary for $\mathsf{Field}^{\uparrow}$ exits; BEDC source BHist with hsame and append; carrier choice $h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R)))$; classifier choice the displayed contextual FieldSingletonClassifier; stability obligations inhabitedness, hsame equivalence laws, and carrier transport under common two-sided context; ledger the same contextual carrier; core theorem list Definition 45.63 and Definition 45.66; exactness predicate contextual singleton-carrier equivalence; standard bridge deferred; status ProofTarget.

Theorem 45.649 (Field singleton two-sided context exactness)**.** *For histories L, R, h, k : BHist, if* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, then the two-sided singleton context is exact for both the carrier and the classifier:*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R))) \Longleftrightarrow \mathsf{FieldSingletonCarrier}(h).$$

and

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{append}(h, R)), \mathsf{append}(L, \mathsf{append}(k, R))) \Longleftrightarrow \mathsf{FieldSingletonClassifier}(h, k).$$

Proof. Assume carrier witnesses for L and R. The carrier equivalence is exactly Definition 45.63 applied to those two witnesses and the middle history h.

For the classifier equivalence, apply Definition 45.66 with the left contexts both equal to L and the right contexts both equal to R. The required context comparisons are the reflexive witnesses $L\mathsf{hsame}L$ and $R\mathsf{hsame}R$, so the cancellation theorem identifies the displayed contextual classifier with $\mathsf{FieldSingletonClassifier}(h, k)$. □

Theorem 45.650 (Field four-factor inverse classifier exactness from apartness)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}$ *candidate over* BHist *satisfying the hypotheses of Definition 45.318, Definition 45.319, and Definition 45.205, let* $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot)$*. For any* $a, x, b, c, a', x', b', c'$: BHist *equipped with factor conversion maps for each of the eight factor apart-zero witnesses, pair-product conversion maps for* $\mathsf{mul}(a, x)$*,* $\mathsf{mul}(b, c)$*,* $\mathsf{mul}(a', x')$*, and* $\mathsf{mul}(b', c')$*, and fourfold product conversion maps for*

$$u := \mathsf{mul}(\mathsf{mul}(a, x), \mathsf{mul}(b, c)) \quad \textit{and} \quad u' := \mathsf{mul}(\mathsf{mul}(a', x'), \mathsf{mul}(b', c')),$$

assume witnesses $q : \mathsf{ApartZero}_F(u)$ *and* $q' : \mathsf{ApartZero}_F(u')$*. Let the induced proof indices* $p_a, p_x, p_b, p_c, p_{ax}, p_{bc}, p_u$ *and* $p_{a'}, p_{x'}, p_{b'}, p_{c'}, p_{a'x'}, p_{b'c'}, p_{u'}$ *be obtained by Definition 45.318 and the displayed conversion maps. Then*

$$u\mathsf{hsame}u' \Longleftrightarrow \mathsf{mul}(\mathsf{mul}(\mathsf{inv}(c, p_c), \mathsf{inv}(b, p_b)), \mathsf{mul}(\mathsf{inv}(x, p_x), \mathsf{inv}(a, p_a)))\mathsf{hsamemul}(\mathsf{mul}(\mathsf{inv}(c', p_{c'}), \mathsf{inv}(b', p_{b'})), \mathsf{mul}(\mathsf{inv}$$

Proof. First build the witness ledger for both fourfold products. Apply Definition 45.318 to q and q'; the forward implications produce the eight factor witnesses, and the reverse binary steps inside the same exactness proof produce the two pair-product witnesses on each side. The factor, pair, and fourfold conversion maps turn these apart-zero witnesses into the proof indices displayed in the statement.

Next apply Definition 45.319 to u with q. This identifies $\mathsf{inv}(u, p_u)$ with the reversed spine $(c, b)(x, a)$ written in the conclusion. Apply the same theorem to u' with q' to identify $\mathsf{inv}(u', p_{u'})$ with the primed reversed spine.

For the forward implication, assume $u\mathsf{hsame}u'$. Use Definition 45.205 in its forward direction to obtain $\mathsf{inv}(u, p_u)\mathsf{hsameinv}(u', p_{u'})$. Transport this equality across the two reversal equalities by symmetry and transitivity of hsame, giving equality of the two reversed spines.

For the reverse implication, assume equality of the two reversed spines. Transport it back across the two reversal equalities to obtain $\mathsf{inv}(u, p_u)\mathsf{hsameinv}(u', p_{u'})$. The reverse direction of Definition 45.205 then yields $u\mathsf{hsame}u'$. □

Per-object certificate: Traditional target: finite fourfold product inverse classifier exactness. BEDC source: $\mathsf{Field}^{\uparrow}$ over BHist plus $\mathsf{Ring}^{\uparrow}$ zero-apartness rows. Carrier choice: BHist. Classifier choice: hsame. Input ledger: the eight factor witnesses. Product ledger: the two pair-product witnesses and two fourfold product witnesses. Core theorem: Definition 45.650. Dependency chain: four-factor apartness exactness, four-factor inverse reversal, and inverse classifier exactness. Standard bridge: deferred. Status: $\mathsf{ProofTarget}$.

Theorem 45.651 (RatUp FieldUp canonical phase-exit proof target)**.** *For* $\mathsf{RatUpFieldSource}_F$ *and Definition 45.617, the canonical phase-exit target is the single* BHist*-anchored certificate whose carrier is* $\mathsf{RatHistoryCarrier}$*, whose classifier is* $\mathsf{RatHistoryClassifier}$*, whose denominator tuple is* $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier})$*, and whose contextual action is* Γ*.*

Its operation rows are exactly the rows covered by Definition 45.502, Definition 45.615, Definition 45.616, and Definition 45.618. Therefore the canonical phase-exit proof target is not a schematic algebra interface, but the concrete certificate object carried by the displayed RatUp history source.

Proof. Unpack the concrete source tuple $\mathsf{RatUpFieldSource}_F$ and Definition 45.617. The carrier, classifier, denominator tuple, and contextual action are the displayed BHist rows, so the target has one concrete source anchor.

Apply Definition 45.502 to identify the source, carrier, classifier, denominator, contextual-action, context-pair, and affine coordinates. Apply Definition 45.615 to cover the operation rows, including denominator histories, context pairs, affine witnesses, inverse and nonzero witnesses, product exactness, and selector endpoint rows.

Apply Definition 45.616 to package those rows as the Phase Algebra Hierarchy paper-certificate object. Finally apply Definition 45.618 to identify this package with Definition 45.617. The formal target therefore belongs naturally to the namespace `BEDC.Derived.FieldUp`. □

Per-object template: traditional target constructive $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ phase exit; BEDC source BHist with $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, denominator, contextual-action, context-pair, and affine rows; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability and exactness rows from Definition 45.502, Definition 45.615, Definition 45.616, and Definition 44.29, and Definition 45.618; ledger $\mathsf{CommRing}^{\uparrow}$ base coordinate, denominator histories, context pairs, affine witnesses, inverse/nonzero witnesses, and selector endpoint rows; status $\mathsf{ProofTarget}$.

Definition 45.652 (Canonical RatUp FieldUp exit row predicate)**.** *For a* BHist *certificate row* R *over* $\mathsf{RatUpFieldSource}_F$*, the predicate* $\mathsf{CanFieldExitRow}(R)$ *holds exactly when* R *belongs to the finite disjunction of concrete rows exported by the displayed* $\mathsf{RatUpFieldSource}_F$ *object:*

1. *one of the source, carrier, classifier, denominator, contextual-action, context-pair, affine, ledger, or exactness rows identified by Definition 45.502;*
2. *one of the carrier and classifier anchors,* $\mathsf{CommRing}^{\uparrow}$ *base rows, denominator multiplication and unit rows, contextual and context-pair rows, affine rows, inverse and nonzero rows, product exactness rows, singleton noncollapse rows, or selector endpoint rows exhausted by Definition 45.615;*
3. *one of the ten-field certificate, one-sided multiplication cancellation, two-sided product classifier and nonzero transport, five-factor ledger, cross-endpoint singleton-classifier, or bracketing-complete classifier rows packaged by Definition 45.616;*
4. *one of the source, carrier, classifier, stability, ledger, core theorem, exactness, bridge-slot, status, or completion-identification rows of Definition 45.617 as certified by Definition 45.618.*

Theorem 45.653 (RatUp FieldUp canonical exit row exactness)**.** *For* $\mathsf{RatUpFieldSource}_F$ *and Definition 45.617, the canonical* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *phase-exit object of Definition 45.651 satisfies exact row coverage, row soundness, and row completeness for Definition 45.652.*

1. *Row coverage: every row of the canonical phase-exit object satisfies* $\mathsf{CanFieldExitRow}$*.*
2. *Row soundness: every row satisfying* $\mathsf{CanFieldExitRow}$ *is a row over the fixed concrete source tuple* $\mathsf{RatUpFieldSource}_F$ *with carrier* $\mathsf{RatHistoryCarrier}$*, classifier* $\mathsf{RatHistoryClassifier}$*, denominator tuple* $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHistoryCarrier})$*, and contextual action* Γ*.*
3. *Row completeness: no row belongs to the canonical phase-exit object unless it satisfies* $\mathsf{CanFieldExitRow}$*.*

Proof. For row coverage, apply Definition 45.502 to fix the source, carrier, classifier, denominator, contextual-action, context-pair, and affine coordinates. Then apply Definition 45.615 to cover the operation inventory: denominator histories, context pairs, affine witnesses, inverse and nonzero witnesses, product exactness, singleton noncollapse, and selector endpoint rows.

For row soundness, project each disjunct in Definition 45.652 back through Definition 45.651. Each case has the same displayed source tuple, the same BHist carrier, the same classifier, the same denominator tuple, and the same contextual action Γ, so the predicate contains no schematic operation row.

For row completeness, apply Definition 45.616 to assemble the covered rows as the Phase Algebra Hierarchy paper certificate. Then use Definition 45.618 to identify that certificate with Definition 45.617. This identification leaves no phase-exit row outside $\mathsf{CanFieldExitRow}$. □

Theorem 45.654 (RatUp FieldUp canonical exit singleton context boundary)**.** *For Definition 45.617, the source tuple* $\mathsf{RatUpFieldSource}_F$*, and histories* $L, R, h, k : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, the same canonical* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit certificate includes the following singleton-context boundary rows over* BHist*.*

1. *The contextual carrier has the semantic certificate*

$$\mathsf{SemanticNameCert}(h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R))))(h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{ap}$$

2. *The contextual carrier is exact:*

$$\mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R))) \iff \mathsf{FieldSingletonCarrier}(h).$$

3. *The contextual classifier is exact:*

$$\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{append}(h, R)), \mathsf{append}(L, \mathsf{append}(k, R))) \iff \mathsf{FieldSingletonClassifier}(h, k).$$

Proof. Apply Definition 45.651 to fix Definition 45.617 as the concrete BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object over $\mathsf{RatUpFieldSource}_F$. This identifies the singleton-context boundary as a row of the same canonical certificate, not as a separate schematic parameter.

For the semantic certificate row, apply Definition 45.648 to the two carrier witnesses for L and R. Its displayed source, pattern, ledger, and classifier are exactly the contextual $\mathsf{FieldSingletonCarrier}$ and $\mathsf{FieldSingletonClassifier}$ rows shown above.

For the two exactness rows, apply Definition 45.649 to the same witnesses and the histories h and k. The carrier projection gives the displayed equivalence for h, and the classifier projection gives the displayed equivalence for the pair (h, k).

Thus the canonical exit consumes concrete BHist, append, $\mathsf{FieldSingletonCarrier}$, and $\mathsf{FieldSingletonClassifier}$ rows from the FieldUp singleton boundary, rather than abstract operator fields. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup canonical exit singleton context boundary`

Per-object template: traditional target singleton-context boundary inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist with $\mathsf{RatUpFieldSource}_F$, append, $\mathsf{FieldSingletonCarrier}$, and $\mathsf{FieldSingletonClassifier}$; carrier $h \mapsto \mathsf{FieldSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R)))$; classifier $(h, k) \mapsto \mathsf{FieldSingletonClassifier}(\mathsf{append}(L, \mathsf{app}$ stability obligations contextual semantic certificate plus exact carrier and classifier cancellation; ledger singleton context wrappers; core theorem list Definition 45.651, Definition 45.648, and Definition 45.649; exactness predicate the displayed carrier and classifier equivalences; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.655 (RatUp FieldUp reduct-context exit closure)**.** *For Definition 45.617 and the fixed source tuple* $\mathsf{RatUpFieldSource}_F$*, the rows supplied by Definition 45.659 and the singleton-context boundary rows of Definition 45.654 are covered by* $\mathsf{CanFieldExitRow}$*. Concretely, for histories* $L, R, h, k : \mathsf{BHist}$ *with* $\mathsf{FieldSingletonCarrier}(L)$ *and* $\mathsf{FieldSingletonCarrier}(R)$*, after the singleton boundary is wrapped by the carried contexts* L *and* R*, its* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* $\mathsf{AbGroup}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*, and* $\mathsf{CommRing}^{\uparrow}$ *reduct projections remain rows over* $\mathsf{RatUpFieldSource}_F$ *with carrier* $\mathsf{RatHistoryCarrier}$*, classifier* $\mathsf{RatHistoryClassifier}$*, denominator tuple* $(\mathsf{RatDenomUnitCarrier}, \mathsf{RatDenomUnitClassifier}, \mathsf{append}, \mathsf{e}, \mathsf{RatHisto}$ *and contextual action* Γ*.*

Proof. Apply Definition 45.653. Its coverage clause places every row of the canonical phase-exit certificate inside $\mathsf{CanFieldExitRow}$, and its soundness clause fixes the source tuple, carrier, classifier, denominator tuple, and contextual action displayed in the statement.

Next apply Definition 45.659. The denominator continuation, singleton obstruction, zero/one boundary, and commutative-ring base coordinate are already rows consumed by Definition 45.617, so their algebra reduct projections do not introduce a separate schematic operation parameter.

Finally apply Definition 45.654 to the witnesses for L and R. Its semantic certificate, carrier exactness, and classifier exactness rows are rows of the same canonical exit object. Projecting these rows to the monoid, group, abelian-group, ring, and commutative-ring reduct views only forgets algebra fields; it preserves the $\mathsf{RatUpFieldSource}_F$ anchor and therefore keeps each projected contextual boundary row inside $\mathsf{CanFieldExitRow}$. □

Per-object template: traditional target $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ reduct-context row closure; BEDC source BHist with $\mathsf{RatUpFieldSource}_F$, RatHistoryCarrier, RatHistoryClassifier, (RatDenomUnitCarrier, RatDenomUnitClassifie and Γ; carrier the canonical Rat history carrier together with the contextual singleton carrier rows; classifier the Rat history classifier together with the contextual singleton classifier rows; stability obligations algebra reduct compatibility, canonical exit row exactness, and singleton-context boundary closure; ledger the monoid, group, abelian-group, ring, and commutative-ring reduct projections of the contextual boundary rows; core theorem Definition 45.655; exactness predicate CanFieldExitRow; standard bridge deferred; status ProofTarget.

Theorem 45.656 (RatUp FieldUp canonical exit contextual action row coverage)**.** *For Definition 45.617, the fixed source tuple* $\mathsf{RatUpFieldSource}_F$*, the denominator tuple* (RatDenomUnitCarrier, RatDenomUnitC *and the contextual action* Γ*, the contextual-action rows supplied by Definition 45.639, Definition 45.640, Definition 45.641, Definition 45.642, Definition 45.643, Definition 45.644, and Definition 45.645 satisfy* CanFieldExitRow*. Each such row remains a row over the same concrete* BHist *source, carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, displayed denominator tuple, and contextual action* Γ*.*

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup canonical exit contextual action row coverage`

Proof. Apply Definition 45.653. Its coverage clause places every row of the canonical phase-exit object inside CanFieldExitRow, and its soundness clause fixes the concrete $\mathsf{RatUpFieldSource}_F$ source tuple, the BHist carrier, the classifier, the denominator tuple, and Γ.

The composition and pair-transport rows are concrete rows of this object. Apply Definition 45.639 to identify the carried nested action, its denominator classifier, and its strict-support readback. Apply Definition 45.640 to identify the two-pair classifier transport and the strict-support equivalence.

The finite-order coherence rows use the same concrete denominator endpoints. Apply Definition 45.641 for the collected threefold action, and apply Definition 45.642 for the reverse-order classifier and support transport. These two rows mention only BHist histories, C^u_D, R^u_D, append, e, and Γ.

The support and noncollapse rows are anchored by the same source data. Apply Definition 45.643 to identify support under three classified context pairs, and apply Definition 45.644 to recover the transported Rat carrier, singleton-empty exclusions, inverse-domain witnesses, and contextual singleton-classifier obstruction.

Finally apply Definition 45.645. Its adjacent-swap row uses the same carried denominator context pairs and the same contextual action. The soundness clause from Definition 45.653 therefore keeps every contextual-action row anchored to $\mathsf{RatUpFieldSource}_F$, rather than to an abstract operation parameter. □

Per-object template: traditional target contextual-action row coverage inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, Γ, C^u_D, R^u_D, N^u_D, and append; carrier/context choices the carried denominator endpoints and context pairs in the cited rows; classifier R^u_D; stability obligations contextual composition, pair transport, threefold support transport, noncollapse, and adjacent-swap closure; ledger the contextual-action row family; core theorem Definition 45.656; exactness predicate CanFieldExitRow; standard bridge deferred; status ProofTarget.

Theorem 45.657 (RatUp FieldUp canonical exit affine row coverage)**.** *For Definition 45.617 and the fixed source tuple* $\mathsf{RatUpFieldSource}_F$*, the affine rows supplied by Definition 45.561, Definition 45.563, Definition 45.281, and Definition 45.665 satisfy* CanFieldExitRow*. Each such row remains a row over the same concrete* BHist *source, carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*,*

denominator tuple (RatDenomUnitCarrier, RatDenomUnitClassifier, append, e, RatHistoryCarrier)*, and contextual action* Γ.

Proof. Apply Definition 45.653. Its coverage clause places every row of the canonical phase-exit object inside CanFieldExitRow, and its soundness clause fixes the concrete $\mathsf{RatUpFieldSource}_F$ source tuple, BHist source sort, carrier, classifier, denominator tuple, and contextual action Γ.

The affine package rows are concrete rows of this same object. Apply Definition 45.561 for the affine group row and apply Definition 45.563 for the threefold composition row. These rows use the displayed Rat history carrier and classifier, so the exactness predicate does not range over an abstract affine operation source.

The identity and inverse rows keep the affine family attached to the same exit object. Apply Definition 45.281 to identify the identity classifier row, and apply Definition 45.665 to identify inverse descent and uniqueness over the canonical Rat denominator tuple. The soundness clause above then keeps all cited affine rows over $\mathsf{RatUpFieldSource}_F$, RatHistoryCarrier, RatHistoryClassifier, the displayed denominator tuple, and Γ. □

Per-object template: traditional target affine row coverage inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist with $\mathsf{RatUpFieldSource}_F$, the displayed denominator tuple, and Γ; carrier RatHistoryCarrier; classifier RatHistoryClassifier; stability obligations canonical exit row exactness, affine group closure, threefold affine composition, identity classification, and inverse descent with uniqueness; ledger the affine automorphism rows cited by the theorem; core theorem Definition 45.657; exactness predicate CanFieldExitRow; standard bridge deferred; status ProofTarget.

Theorem 45.658 (RatUp FieldUp canonical apart-zero context certificate row)**.** *For Definition 45.617, the source tuple* $\mathsf{RatUpFieldSource}_F$*, histories* $L, R, h : \mathsf{BHist}$*, and a witness* $p : \mathsf{FieldApartZero}(L)$*, the canonical* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit certificate contains the concrete two-sided apart-zero context row*

$$\mathsf{SemanticNameCert}(h \mapsto \mathsf{FieldApartZero}(\mathsf{append}(L, \mathsf{append}(h, R))))(h \mapsto \mathsf{FieldApartZero}(\mathsf{append}(L, \mathsf{append}(h, R))))($$

Its source, pattern, and ledger policy are all the displayed contextual apart-zero predicate, and its classifier is hsame.

Proof. Apply Definition 45.651 to fix Definition 45.617 as the concrete BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object over $\mathsf{RatUpFieldSource}_F$. This pins the row to the canonical exit certificate rather than to a schematic field interface.

Next apply Definition 45.270 to the histories L and R and the apart-zero witness p. Its source, pattern, ledger policy, and classifier are exactly the four displayed arguments of SemanticNameCert.

Combining the fixed canonical exit object with that semantic certificate gives row membership for the contextual apart-zero predicate inside the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate. □

Per-object template: traditional target constructive apart-zero context stability inside $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, $\mathsf{RatUpFieldSource}_F$, append, and FieldApartZero; carrier $h \mapsto \mathsf{FieldApartZero}(\mathsf{append}(L, \mathsf{append}($ classifier hsame; ledger the hidden left and right contexts L and R; core theorem list Definition 45.651 and Definition 45.270; exactness predicate row membership in Definition 45.617; status ProofTarget.

Theorem 45.659 (RatUp FieldUp algebra reduct compatibility)**.** *For Definition 45.617, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate is compatible with the already licensed algebraic reducts over BEDC histories. Its denominator continuation product uses the same empty-unit and associativity data as*

the unary append $\mathsf{Monoid}^{\uparrow}$ *certificate; its singleton empty obstruction rows restrict to the concrete singleton* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{AbGroup}^{\uparrow}$ *certificates; its zero/one singleton boundary restricts to the singleton empty-history* $\mathsf{Ring}^{\uparrow}$ *laws; its zero/one singleton boundary together with multiplicative commutativity restricts to the singleton empty-history* $\mathsf{CommRing}^{\uparrow}$ *schema certificate of Theorem 44.29; and Theorem 45.618 consumes these reduct rows without introducing schematic operation parameters.*

Proof. Apply Theorem 40.25 to the denominator continuation product. The empty source, left and right unit laws, append associativity, and append congruence are exactly the rows used by the denominator carrier and classifier package.

Next apply Theorem 41.5 and Theorem 42.10. The singleton empty obstruction rows have carrier A_{e} and classifier ρ_{e}, so their group and abelian-group reducts are the corresponding singleton certificates.

For the zero/one boundary, apply Theorem 43.6. Its singleton empty-history ring tuple supplies the additive, multiplicative, zero, and one laws required by the boundary rows.

For the commutative-ring base coordinate, apply Theorem 44.29. Its carrier, classifier, constant-operation tuple, and multiplicative commutativity field supply the $\mathsf{CommRing}^{\uparrow}$ base coordinate named by Definition 45.587.

Finally apply Theorem 45.618. That theorem packages the displayed denominator, singleton, ring, and commutative-ring reduct rows into the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object, and its conclusion states that no schematic mul, add, or neg parameter is part of the certificate. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatUpFieldUp algebra reduct compatibility`

Theorem 45.660 (RatUp FieldUp apart-product inverse classifier exit row)**.** *For Definition 45.617, bind the rows of Definition 45.587 as follows: let* μ_F *be its concrete multiplication row, let* ι_F *be its inverse row over* $\mathsf{FieldApartZero}$*, and let* N_F *be the certified nonzero/apartness row carried by the same table. If* $a, b, c, d : \mathsf{BHist}$ *carry witnesses* $p_a : N_F(a)$*,* $p_b : N_F(b)$*,* $p_c : N_F(c)$*, and* $p_d : N_F(d)$*, and if the two products are apart from the empty history by hypotheses* $h_s : \mu_F(a, b)\mathsf{hsamee} \to \bot$ *and* $h_t : \mu_F(c, d)\mathsf{hsamee} \to \bot$*, then the concrete classifier row records*

$$\mu_F(a, b)\mathsf{hsame}\mu_F(c, d) \Longleftrightarrow \mu_F(\iota_F(b, p_b), \iota_F(a, p_a))\mathsf{hsame}\mu_F(\iota_F(d, p_d), \iota_F(c, p_c)).$$

This displayed equivalence is an internal operation-exhaustion row of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object, with carrier* BHist *and classifier* hsame*.*

Proof. First invoke Definition 45.588. It binds the concrete carrier and classifier rows, the $\mathsf{CommRing}^{\uparrow}$ base rows, the inverse row over $\mathsf{FieldApartZero}$, and the nonzero row inside Definition 45.587. Thus μ_F, ι_F, and N_F are rows of the displayed table rather than free operation parameters.

Next invoke Definition 45.589. It locates product classifier exactness and nonzero transport inside the same concrete source tuple $\mathsf{RatUpFieldSource}_F$. The same package supplies the bridge from explicit product apartness to the nonzero row, so the hypotheses h_s and h_t may be consumed without presupposing separate product nonzero witnesses.

Next apply the checked theorem `BEDC.Derived.FieldUp.field_apart_product_inverse_classifier_exact` recorded in Definition 45.219. With the four certified factor witnesses and the two product-apartness hypotheses, the generic history-level theorem converts classifier equality of $\mu_F(a, b)$ and $\mu_F(c, d)$ into classifier equality of the reversed inverse products built from ι_F, and conversely.

Finally use Definition 45.615 together with Definition 45.618. These theorems record that the product, inverse, and nonzero rows are consumed by Definition 45.617, so the displayed classifier equivalence is part of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object rather than an external schematic operation parameter. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupInverseProductClassifierExitRow`

Per-object template: traditional target inverse-product exactness row for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier BHist; classifier hsame; ledger the four factor witnesses plus the two product-apartness witnesses; exactness predicate the displayed classifier equivalence; status $\mathsf{ProofTarget}$.

Per-object template: traditional target algebra-hierarchy reduct compatibility for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist with the concrete $\mathsf{RatUpFieldSource}_F$ rows; carrier rows $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatDenomUnitCarrier}$; classifier rows $\mathsf{RatHistoryClassifier}$ and $\mathsf{RatDenomUnitClassifier}$; ledger the denominator continuation reduct, singleton empty obstruction reducts, and singleton zero/one ring and commutative-ring reducts, including the $\mathsf{CommRing}^{\uparrow}$ base coordinate; core theorem list Theorem 40.25, Theorem 41.5, Theorem 42.10, Theorem 43.6, Theorem 44.29, and Theorem 45.618; exactness predicate reduct-row coverage for Definition 45.617; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.661 (RatUp FieldUp affine automorphism action certificate)**.** *For Definition 45.617, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object exports the affine automorphism action certificate on* BHist *required by downstream algebra. The action carrier is* AffAut_F*; its classifier is* $\equiv_{\mathrm{aff}}$*; its identity, composition, and inverse laws are those of Theorem 45.558; each affine fiber is singleton-classified by Theorem 45.540; and common affine factors cancel exactly by Theorem 45.566.*

Proof. First apply Theorem 45.618 and Theorem 45.659. These fix the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ operations and the reduct rows over BHist, so the action is attached to the same exit object rather than to a schematic field package.

Next apply Theorem 45.558 to the fixed operations. Its carrier AffAut_F and classifier $\equiv_{\mathrm{aff}}$ supply the identity, composition, and inverse fields of the affine action certificate.

Finally project Theorem 45.540 and Theorem 45.566. The first projection supplies singleton classification of every affine fiber, and the second supplies exact cancellation for common affine factors, which are the action exactness fields consumed downstream. □

Per-object template: traditional target affine group action over a field; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows over BHist; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability obligations group laws, fiber singleton exactness, and cancellation exactness; ledger nonzero coefficient witnesses, product witnesses, offsets, and inverse witnesses; core theorem Theorem 45.661; exactness predicate fiber and cancellation exactness for the action; standard bridge ordinary affine action; status $\mathsf{ProofTarget}$.

Theorem 45.662 (RatUp FieldUp canonical exit action and inverse package)**.** *For Definition 45.617, the canonical* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit target simultaneously exports the following rows as one concrete exit package.*

1. *The algebra-reduct compatibility rows of Definition 45.659.*
2. *The inverse-product classifier exactness row of Definition 45.660.*
3. *The affine automorphism action certificate of Definition 45.661.*

Thus downstream $\mathsf{Field}^{\uparrow}$ *consumers may cite this one canonical package for reduct rows, inverse-product exactness, and affine action support without reassembling the three certificate rows from their separate locations.*

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupCanonicalExitActionInversePackage`

Proof. First invoke Definition 45.651. This fixes Definition 45.617 as the concrete BHist-anchored source object with carrier RatHistoryCarrier, classifier RatHistoryClassifier, denominator data, and contextual action Γ.

Apply Definition 45.659 to the fixed source anchor. This supplies the denominator continuation reduct, singleton empty obstruction reducts, singleton zero/one ring rows, and commutative-ring base coordinate as fields of the same concrete exit object.

Next apply Definition 45.660. Its row is carried by the same BHist source and classifier hsame, so the inverse-product classifier exactness field is collected without changing the carrier or source tuple.

Finally apply Definition 45.661. Its affine carrier AffAut_F, classifier $\equiv_{\mathrm{aff}}$, group laws, fiber singleton exactness, and cancellation exactness become the action fields of the same exit object.

This is not a single-theorem variant of any one cited result. It composes three separately located exit rows into one canonical package whose fields are all anchored by Definition 45.617. □

Per-object template: traditional target canonical $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit action and inverse package; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier RatHistoryCarrier together with AffAut_F for the action row; classifier RatHistoryClassifier, hsame, and $\equiv_{\mathrm{aff}}$ on their displayed rows; stability obligations algebra-reduct compatibility, inverse-product classifier exactness, affine group laws, affine fiber singleton exactness, and affine cancellation exactness; ledger denominator continuation reducts, singleton obstruction reducts, zero/one boundary rows, commutative-ring base coordinate, certified nonzero factor witnesses, product witnesses, affine coefficients, offsets, and inverse witnesses; core theorem list Definition 45.651, Definition 45.659, Definition 45.660, and Definition 45.661; exactness predicate inverse-product classifier equivalence plus affine fiber and cancellation exactness; standard bridge ordinary field reducts with ordinary affine action; status ProofTarget.

Theorem 45.663 (RatUp FieldUp contextual pair transport full noncollapse invariance)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and the contextual action* Γ *of Definition 45.454, let*

$$x_L := \Gamma_{p',q',l',r'}(\Gamma_{p,q,l,r}(h))$$

and

$$x_R := \Gamma_{p',q',s',t'}(\Gamma_{p,q,s,t}(h)).$$

Suppose $p$hsamee*,* $q$hsamee*,* p'hsamee*, and* q'hsamee*; suppose* $C^u_D(h)$*; suppose the four denominator context comparisons* $R^u_D(l,s)$*,* $R^u_D(r,t)$*,* $R^u_D(l',s')$*, and* $R^u_D(r',t')$ *hold; and suppose* RatHistoryCarrier(x_L)*. Then the transported endpoint* x_R *has the following concrete noncollapse package:*

1. $R^u_D(x_L, x_R)$*;*
2. $C^u_D(x_R)$*;*
3. RatHistoryCarrier(x_R)*;*
4. fieldSingletonEmptyCarrier$(x_R) \to \bot$*;*
5. $x_R$hsamee $\to \bot$*;*
6. *for every* r : BHist*,* fieldSingletonEmptyCarrier(append$(r, x_R)) \to \bot$*;*
7. *for every* r : BHist*,* fieldSingletonEmptyCarrier(append$(x_R, r)) \to \bot$*;*
8. *for every* r : BHist*,* fieldSingletonEmptyNonZero(append(r, x_R))*;*

9. *for every* $r : \mathsf{BHist}$, $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(x_R, r))$;

10. *for all* $L, R : \mathsf{BHist}$,

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, x_R), \mathsf{append}(R, x_R)) \to \bot.$$

Proof. First apply Definition 45.500 to the four denominator classifier rows and the two nested contextual actions. This gives the transported denominator classifier $R^u_D(x_L, x_R)$.

Next apply Definition 45.499 to the same empty endpoints, carried source, and classifier rows. Transporting the assumed $\mathsf{RatHistoryCarrier}(x_L)$ witness gives $\mathsf{RatHistoryCarrier}(x_R)$, and the carrier component gives $C^u_D(x_R)$.

The endpoint noncollapse clauses come from Definition 45.520 with the same data. This supplies the singleton-empty rejection at x_R and the direct empty-history rejection $x_R\mathsf{hsamee} \to \bot$.

For contextual singleton-carrier rejection, apply Definition 45.516. For each context history r, it rejects both $\mathsf{append}(r, x_R)$ and $\mathsf{append}(x_R, r)$ as singleton-empty carriers.

Then apply Definition 45.521 to the same transported endpoint. It repacks the two contextual carrier rejections as the proof-indexed nonzero witnesses for both append orders.

Finally apply Definition 45.518. For all $L, R : \mathsf{BHist}$, this rules out a singleton-empty classifier between $\mathsf{append}(L, x_R)$ and $\mathsf{append}(R, x_R)$, completing the displayed package. □

Theorem 45.664 (RatUp FieldUp affine inverse uniqueness exit)**.** *For Definition 45.617, let* $T : \mathsf{AffAut}_F$ *be a certified affine automorphism in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $S : \mathsf{AffAut}_F$ *be the explicit affine inverse of* T *supplied by Theorem 45.555, and let* $U : \mathsf{AffAut}_F$ *be any certified affine automorphism whose left and right affine composites with* T *classify as the affine identity under* $\equiv_{\mathrm{aff}}$*:*

$$(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} I.$$

Then U *is the same affine inverse as* S *under the concrete action classifier:*

$$U \equiv_{\mathrm{aff}} S.$$

Proof. First apply Theorem 45.618 to fix the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Apply Theorem 45.661 to attach the carrier AffAut_F, the affine composition $\circ_{\mathrm{aff}}$, the affine identity I, and the classifier $\equiv_{\mathrm{aff}}$ to that same object.

The explicit inverse package Theorem 45.555 supplies the displayed inverse witness S for T. The hypotheses supply the matching left and right inverse equations for U against the same T and under the same affine classifier.

Apply Theorem 45.559. Its conclusion is the group-classifier equality between the arbitrary inverse candidate U and the explicit inverse S. The action certificate identifies that group classifier with the displayed affine classifier, so the returned relation is exactly $U \equiv_{\mathrm{aff}} S$. □

Per-object template: traditional target affine inverse uniqueness over a field action; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphism histories; classifier $\equiv_{\mathrm{aff}}$; stability obligations affine group certificate and inverse equations; ledger inverse witnesses and composite witnesses; core theorem Theorem 45.664; exactness predicate uniqueness of the affine inverse under the action classifier; standard bridge ordinary uniqueness of inverse in an affine automorphism group; status $\mathsf{ProofTarget}$.

Theorem 45.665 (RatUp FieldUp canonical affine inverse descent and uniqueness)**.** *For Definition 45.617, let $T, T' : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $S, S' : \mathsf{AffAut}_F$ be their explicit affine inverse witnesses supplied by Theorem 45.555. If the two input automorphisms have the same affine classifier,*

$$T \equiv_{\mathrm{aff}} T',$$

then their explicit inverse witnesses have the same affine classifier:

$$S \equiv_{\mathrm{aff}} S'.$$

Moreover, for any $U : \mathsf{AffAut}_F$, if

$$(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} I,$$

then U is the explicit inverse witness for T under the same classifier:

$$U \equiv_{\mathrm{aff}} S.$$

Proof. First invoke Theorem 45.662. This fixes the concrete exit action carrier, the classifier $\equiv_{\mathrm{aff}}$, the identity I, the composition $\circ_{\mathrm{aff}}$, and the inverse package inside Definition 45.617.

For classifier descent, apply Theorem 45.552 to the explicit inverse formula for T and T'. The hypothesis $T \equiv_{\mathrm{aff}} T'$ is exactly the input classifier equality, so the returned relation is $S \equiv_{\mathrm{aff}} S'$.

For uniqueness of an arbitrary inverse candidate, keep the same action carrier and apply Theorem 45.664 to U with its two displayed composite witnesses. Its conclusion is precisely $U \equiv_{\mathrm{aff}} S$ under the concrete affine classifier. □

Per-object template: traditional target affine inverse operation in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit action; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability obligations inverse classifier descent and two-sided inverse uniqueness; ledger affine coefficients, nonzero witnesses, explicit inverse witnesses, and composite witnesses; core theorem list Theorem 45.662, Theorem 45.552, and Theorem 45.664; exactness predicate inverse descent plus uniqueness under the action classifier; standard bridge ordinary affine inverse operation; status $\mathsf{ProofTarget}$.

Theorem 45.666 (RatUp FieldUp affine inverse determinacy exit)**.** *For Definition 45.617, let $T, U, V : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Suppose that both U and V have left and right affine composites with T classified as the affine identity under $\equiv_{\mathrm{aff}}$:*

$$(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} I$$

and

$$(V \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} V) \equiv_{\mathrm{aff}} I.$$

Then the two inverse candidates are equal under the concrete affine classifier:

$$U \equiv_{\mathrm{aff}} V.$$

Proof. First apply Theorem 45.661 to the concrete exit object. This supplies the carrier AffAut_F, the affine identity I, composition $\circ_{\mathrm{aff}}$, and the classifier $\equiv_{\mathrm{aff}}$ together with its symmetry and transitivity rules.

Take the explicit inverse S of T from Theorem 45.555. Apply Theorem 45.664 to the two composite witnesses for U. This gives

$$U \equiv_{\mathrm{aff}} S.$$

Apply the same theorem to the two composite witnesses for V. This gives

$$V \equiv_{\mathrm{aff}} S.$$

Use symmetry for the affine classifier to turn the second relation into $S \equiv_{\mathrm{aff}} V$. Transitivity of the same classifier then composes $U \equiv_{\mathrm{aff}} S$ with $S \equiv_{\mathrm{aff}} V$, yielding $U \equiv_{\mathrm{aff}} V$. □

Per-object template: traditional target inverse determinacy in the affine automorphism group; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations affine classifier symmetry and transitivity; ledger two inverse-candidate composite witnesses; core theorem Theorem 45.666; exactness predicate uniqueness of two-sided inverse candidates; standard bridge ordinary inverse determinacy in a group action; status ProofTarget.

Theorem 45.667 (RatUp FieldUp canonical exit inverse determinacy package)**.** *Definition 45.617 exports, as one concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package, the contextual pair transport full noncollapse invariance of Theorem 45.663, the affine inverse uniqueness row of Theorem 45.664, and the affine inverse determinacy row of Theorem 45.666.*

Proof. First fix the canonical exit object by applying Theorem 45.651 and Theorem 45.662. The first theorem identifies Definition 45.617 as the concrete machine-checkable exit object, and the second attaches the action and inverse rows to that same object.

Next project the contextual noncollapse row from Theorem 45.663. Project the affine inverse uniqueness row from Theorem 45.664. Project the affine inverse determinacy row from Theorem 45.666.

The three projected rows share the same BHist source, denominator contextual action, affine carrier, and classifier rows. Conjunction introduction therefore gives the advertised public package: contextual transport keeps the endpoint noncollapsed, uniqueness identifies any two-sided affine inverse with the explicit inverse, and determinacy identifies any two inverse candidates with each other. □

Per-object template: traditional target concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit inverse-determinacy package; BEDC source BHist, $\mathsf{RatUpFieldSource}_F$, denominator contextual actions, and AffAut_F; carrier RatHistoryCarrier plus certified affine automorphisms; classifier RatHistoryClassifier, hsame, and $\equiv_{\mathrm{aff}}$; stability obligations contextual transport, endpoint noncollapse, affine inverse uniqueness, and affine inverse determinacy; ledger transported context pairs and affine inverse witnesses; exactness predicate the displayed noncollapse package plus inverse-candidate classifier equality; standard bridge ordinary field action inverse determinacy; status ProofTarget.

Theorem 45.668 (RatUp FieldUp affine inverse compositional determinacy)**.** *For Definition 45.617, let* $T, U, V, W : \mathsf{AffAut}_F$ *be certified affine automorphisms in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Suppose that* U *and* V *both satisfy the left and right inverse equations with* T *under* $\equiv_{\mathrm{aff}}$*:*

$$(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} I$$

and

$$(V \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} I$$

and

$$(T \circ_{\mathrm{aff}} V) \equiv_{\mathrm{aff}} I.$$

Then inverse-candidate equality is stable under affine composition on either side:

$$(U \circ_{\mathrm{aff}} W) \equiv_{\mathrm{aff}} (V \circ_{\mathrm{aff}} W)$$

and

$$(W \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} (W \circ_{\mathrm{aff}} V).$$

Proof. First apply Definition 45.666 to the four inverse equations for U and V. This gives the determined inverse-candidate equality

$$U \equiv_{\mathrm{aff}} V.$$

Next apply Definition 45.661 to the same concrete exit object. The certificate exposes affine composition $\circ_{\mathrm{aff}}$, classifier reflexivity, and operation congruence for the displayed affine classifier.

For right composition by W, use operation congruence with $U \equiv_{\mathrm{aff}} V$ in the first coordinate and reflexivity $W \equiv_{\mathrm{aff}} W$ in the second coordinate. This yields $(U \circ_{\mathrm{aff}} W) \equiv_{\mathrm{aff}} (V \circ_{\mathrm{aff}} W)$.

For left composition by W, use reflexivity $W \equiv_{\mathrm{aff}} W$ in the first coordinate and $U \equiv_{\mathrm{aff}} V$ in the second coordinate. The same congruence row yields $(W \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} (W \circ_{\mathrm{aff}} V)$. □

Per-object template: traditional target inverse candidate descent through affine composition; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability affine inverse determinacy and affine composition congruence; ledger T, U, V, W and the four inverse equations; core theorem Theorem 45.668; exactness predicate compositional equality of determined inverse candidates; standard bridge ordinary group-action congruence; status ProofTarget.

Theorem 45.669 (RatUp FieldUp final certificate noncollapse action package)**.** *For Definition 45.617 and* $\mathsf{RatUpFieldSource}_F$*, the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit certificate contains one package whose rows are all anchored at the same concrete* BHist *source object. The package consists of:*

1. *the source anchor, carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, denominator tuple, and contextual action from Theorem 45.651;*
2. *the algebra-reduct row, inverse-product classifier row, and affine-action rows from Theorem 45.662;*
3. *the transported contextual-pair noncollapse clauses from Theorem 45.663;*
4. *the affine inverse uniqueness row from Theorem 45.664.*

Thus the Phase Algebra Hierarchy exit criterion may consume this object as one concrete certificate boundary, not as a family indexed by schematic operation parameters.

Proof. Apply Theorem 45.651. This fixes the concrete $\mathsf{RatUpFieldSource}_F$ source, the BHist carrier object, the history classifier, the denominator tuple, and the contextual action in the concrete exit certificate.

Apply Theorem 45.662 to the same exit certificate. It attaches the algebra-reduct, inverse-product classifier, and affine-action rows without changing the source object selected by the canonical certificate.

Apply Theorem 45.663. The transported contextual-pair clauses use the same BHist endpoint source and its denominator contexts, so the noncollapse rows are certificate rows over the concrete object rather than fresh parameters.

Finally apply Theorem 45.664. The affine inverse uniqueness row is read in the affine action carrier supplied above, so it closes the same concrete certificate package and supplies the displayed single boundary consumed by the hierarchy exit criterion. □

Per-object template: traditional target final constructive $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate boundary; BEDC source $\mathsf{RatUpFieldSource}_F$, BHist, denominator tuple, contextual action, and AffAut_F; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability obligations source anchor, reduct rows, inverse-product exactness, affine action laws, contextual noncollapse, and affine inverse uniqueness; ledger denominator contexts, inverse witnesses, affine witnesses, and transported endpoints; core theorem list Theorem 45.651, Theorem 45.662, Theorem 45.663, and Theorem 45.664; exactness predicate one concrete exit certificate package; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.670 (RatUp FieldUp affine automorphism cancellation exactness exit)**.** *For Definition 45.617, let $S, T, U : \mathsf{AffAut}_F$ be certified affine automorphism histories in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. With the concrete affine composition $\circ_{\mathrm{aff}}$ and classifier $\equiv_{\mathrm{aff}}$ supplied by Definition 45.661, common right affine factors cancel exactly:*

$$(S \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} (U \circ_{\mathrm{aff}} T) \Longleftrightarrow S \equiv_{\mathrm{aff}} U.$$

Common left affine factors cancel exactly:

$$(T \circ_{\mathrm{aff}} S) \equiv_{\mathrm{aff}} (T \circ_{\mathrm{aff}} U) \Longleftrightarrow S \equiv_{\mathrm{aff}} U.$$

Thus affine cancellation exactness is a concrete row of the canonical $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object, not an external schematic field parameter.

Proof. First fix the concrete exit object by Definition 45.618. This anchors the carrier BHist, the source $\mathsf{RatUpFieldSource}_F$, and the operation rows used by the affine automorphism certificate.

Next apply Definition 45.661. It identifies the affine carrier AffAut_F, composition $\circ_{\mathrm{aff}}$, identity, inverse, and classifier $\equiv_{\mathrm{aff}}$ as rows exported by the same concrete exit object.

Finally apply Definition 45.566. Read its classifier as the concrete affine classifier just identified, and read its two common-factor cancellation clauses as the displayed right and left equivalences. □

Per-object template: traditional target affine automorphism action cancellation exactness over $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source concrete $\mathsf{RatUpFieldSource}_F$, BHist, and AffAut_F; carrier certified affine automorphism histories; classifier $\equiv_{\mathrm{aff}}$; stability obligation affine group action certificate plus cancellation exactness; ledger common affine factor and composition provenance; core theorem Theorem 45.670; exactness predicate the two displayed iff rows; standard bridge ordinary affine automorphism cancellation; status $\mathsf{ProofTarget}$.

Theorem 45.671 (RatUp FieldUp affine inverse antihomomorphism exit)**.** *For Definition 45.617, let $T, U : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $U \circ_{\mathrm{aff}} T$ be the concrete affine composite supplied by Definition 45.662, and let $\iota_F(-)$ denote the explicit inverse witness supplied by Definition 45.555 under the same concrete action classifier.*

Then the inverse of a composite is the reverse composite of the explicit inverses under the concrete affine classifier:

$$\iota_F(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} \iota_F(T) \circ_{\mathrm{aff}} \iota_F(U).$$

The same classifier also identifies the double inverse with the original certified affine automorphism:

$$\iota_F(\iota_F(T)) \equiv_{\mathrm{aff}} T.$$

Proof. First fix the concrete exit action by Definition 45.662. This fixes the carrier AffAut_F, the concrete affine composite $\circ_{\mathrm{aff}}$, the explicit inverse row, and the classifier $\equiv_{\mathrm{aff}}$ inside Definition 45.617.

For the composite inverse law, apply Definition 45.546 pointwise to the two certified affine automorphisms. The theorem identifies the collected inverse of $U \circ_{\mathrm{aff}} T$ with the pointwise reverse composite $\iota_F(T) \circ_{\mathrm{aff}} \iota_F(U)$ under hsame.

The affine action classifier from the fixed exit package lifts this pointwise hsame relation to the displayed relation

$$\iota_F(U \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} \iota_F(T) \circ_{\mathrm{aff}} \iota_F(U).$$

For the double-inverse clause, view T as a two-sided inverse candidate for $\iota_F(T)$ by the identity and inverse fields of Definition 45.555. Apply Definition 45.666 to compare this candidate with the explicit inverse $\iota_F(\iota_F(T))$ of $\iota_F(T)$. The resulting classifier equality is exactly $\iota_F(\iota_F(T)) \equiv_{\mathrm{aff}} T$. □

Per-object template: traditional target affine inverse antihomomorphism; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations concrete composition, inverse uniqueness, and reverse-composite inverse coherence; ledger factor witnesses, product witnesses, inverse witnesses, and offsets; core theorem list Definition 45.662, Definition 45.546, and Definition 45.666; exactness predicate inverse antihomomorphism plus double-inverse law; standard bridge deferred; status $\mathsf{ProofTarget}$.

Capstone.

Theorem 45.672 (RatUp FieldUp affine conjugation equation exactness)**.** *For Definition 45.617, let $A, X, B : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\iota_F(A)$ be the explicit inverse supplied by Definition 45.662, and define the affine inner action by*

$$\mathsf{Conj}_A(X) := (A \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(A).$$

Then affine conjugation classifies the middle affine automorphism exactly:

$$\mathsf{Conj}_A(X) \equiv_{\mathrm{aff}} B \iff X \equiv_{\mathrm{aff}} (\iota_F(A) \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} A.$$

The inverse affine inner action also cancels on both sides:

$$\mathsf{Conj}_{\iota_F(A)}(\mathsf{Conj}_A(X)) \equiv_{\mathrm{aff}} X.$$

$$\mathsf{Conj}_A(\mathsf{Conj}_{\iota_F(A)}(X)) \equiv_{\mathrm{aff}} X.$$

Proof. Fix the concrete affine action package of Definition 45.662. This fixes the carrier AffAut_F, the collected-coefficient composite $\circ_{\mathrm{aff}}$, the explicit inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$.

Instantiate Definition 45.558 on this package. The identity, associativity, inverse laws, and classifier congruence fields make AffAut_F a concrete $\mathsf{Group}^{\uparrow}$ object for the displayed affine operation.

Apply Definition 41.76 inside that $\mathsf{Group}^{\uparrow}$ object with group element A, middle element X, and target B. Under the fixed action notation, the resulting group classifier statement is exactly

$$\mathsf{Conj}_A(X) \equiv_{\mathrm{aff}} B \iff X \equiv_{\mathrm{aff}} (\iota_F(A) \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} A.$$

For the first cancellation clause, expand $\mathsf{Conj}_{\iota_F(A)}(\mathsf{Conj}_A(X))$. Associativity rewrites it as left multiplication by $\iota_F(A)$, then by A, followed by the displayed right inverse of $\iota_F(A)$. Use Definition 45.671 to normalize $\iota_F(\iota_F(A))$ to A, and then collapse the adjacent inverse pairs by the affine group certificate.

For the second cancellation clause, expand $\mathsf{Conj}_A(\mathsf{Conj}_{\iota_F(A)}(X))$. The same associativity and inverse-law fields reduce the middle product after Definition 45.671 identifies the double inverse of A with A. The remaining term is classifier-equal to X. □

Per-object template: traditional target affine inner automorphism exactness; BEDC source $\mathsf{RatUpFieldSource}_F$ plus AffAut_F; carrier certified affine automorphism histories; classifier $\equiv_{\mathrm{aff}}$; stability obligations affine group laws, conjugation exactness, inverse antihomomorphism, and double inverse; ledger affine coefficients and inverse witnesses; core theorem list Definition 45.662, Definition 45.558, Definition 41.76, and Definition 45.671; exactness predicate affine conjugation equation exactness; standard bridge ordinary inner automorphism equation exactness; status $\mathsf{ProofTarget}$.

Theorem 45.673 (RatUp FieldUp affine conjugation exactness)**.** *For Definition 45.617, let A, S, T : AffAut_F be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine composite, let $\iota_F(-)$ be the explicit inverse witness, and let $\equiv_{\mathrm{aff}}$ be the affine classifier supplied by Definition 45.662. Define*

$$\mathrm{Conj}_A(S) = A \circ_{\mathrm{aff}} S \circ_{\mathrm{aff}} \iota_F(A).$$

Then conjugation by A is classifier-exact:

$$\mathrm{Conj}_A(S) \equiv_{\mathrm{aff}} \mathrm{Conj}_A(T) \iff S \equiv_{\mathrm{aff}} T.$$

Its inverse conjugation action is conjugation by the explicit inverse:

$$\mathrm{Conj}_{\iota_F(A)}(\mathrm{Conj}_A(S)) \equiv_{\mathrm{aff}} S \quad \textit{and} \quad \mathrm{Conj}_A(\mathrm{Conj}_{\iota_F(A)}(S)) \equiv_{\mathrm{aff}} S.$$

Proof. First fix the concrete affine group package from Definition 45.662. This supplies the carrier AffAut_F, the operation $\circ_{\mathrm{aff}}$, the classifier $\equiv_{\mathrm{aff}}$, and the explicit inverse row inside the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

Apply Definition 41.76 to this affine group package with $a = A$ and with the equation target specialized to $\mathrm{Conj}_A(T)$. The group theorem gives the middle-factor equation equivalent to the displayed conjugation equation. Use the two cancellation clauses and operation congruence in Definition 45.566 to identify that middle-factor equation with $S \equiv_{\mathrm{aff}} T$.

For the inverse-action clause, expand $\mathrm{Conj}_{\iota_F(A)}(\mathrm{Conj}_A(S))$ and $\mathrm{Conj}_A(\mathrm{Conj}_{\iota_F(A)}(S))$. The reverse-composite inverse law and double-inverse clause of Definition 45.671 identify the conjugating factors as mutual inverse witnesses. The remaining outer factors cancel by Definition 45.566, leaving exactly S under $\equiv_{\mathrm{aff}}$ in both composites. □

Per-object template: traditional target inner automorphism of the affine automorphism group; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus $\mathsf{Group}^{\uparrow}$; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; ledger conjugation factors and inverse witnesses; exactness predicate classifier-exact conjugation; standard bridge ordinary inner automorphism; status $\mathsf{ProofTarget}$.

Theorem 45.674 (RatUp FieldUp affine conjugation equation exactness exit)**.** *For Definition 45.617, let $T, S, V : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine composite, let $\iota_F(T)$ be the explicit inverse of T, and let $\equiv_{\mathrm{aff}}$ be the affine classifier supplied by Definition 45.661 for the same fixed action. Then conjugation by T classifies the middle affine automorphism exactly:*

$$((T \circ_{\mathrm{aff}} S) \circ_{\mathrm{aff}} \iota_F(T)) \equiv_{\mathrm{aff}} V \Longleftrightarrow S \equiv_{\mathrm{aff}} ((\iota_F(T) \circ_{\mathrm{aff}} V) \circ_{\mathrm{aff}} T).$$

Proof. First fix the concrete exit action by Definition 45.662. This pins down the affine carrier AffAut_F, the concrete composite $\circ_{\mathrm{aff}}$, the explicit inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$ inside Definition 45.617.

Read the same carrier, composite, inverse, and classifier through Definition 45.661. The action certificate supplies the associativity, identity, inverse, and operation congruence rows needed to regard these data as the concrete affine group package.

Instantiate Definition 41.76 in that fixed group package with $a = T$, $x = S$, and $b = V$. The resulting classifier statement is the displayed equivalence after replacing the package multiplication, inverse, and classifier symbols by $\circ_{\mathrm{aff}}$, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$.

The forward direction carries an affine equation for $(T \circ_{\mathrm{aff}} S) \circ_{\mathrm{aff}} \iota_F(T)$ to the recovered middle factor S. The reverse direction transports the recovered middle factor back through left composition by T and right composition by $\iota_F(T)$, using the same concrete inverse witnesses and classifier congruence. □

Per-object template: traditional target affine conjugation equation solving; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus $\mathsf{GroupUp}$ conjugation exactness; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations associativity, identity, inverse, and operation congruence of the fixed action; ledger conjugating automorphism and inverse witness provenance; core theorem list Definition 45.662, Definition 45.661, and Definition 41.76; exactness predicate the displayed iff; standard bridge ordinary affine conjugacy equations; status $\mathsf{ProofTarget}$.

Theorem 45.675 (RatUp FieldUp affine inner conjugation exit)**.** *For Definition 45.617, let $T, V, W : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\iota_F(-)$ be the explicit inverse from Definition 45.662, and define*

$$\mathrm{Inn}_T(V) = T \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(T).$$

Then inner conjugation by T is a concrete affine action theorem for the canonical exit object:

1. $\mathrm{Inn}_T(V)$ *is a certified affine automorphism.*
2. *If* $V \equiv_{\mathrm{aff}} W$*, then* $\mathrm{Inn}_T(V) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(W)$*.*
3. $\mathrm{Inn}_T(\mathrm{id}_F) \equiv_{\mathrm{aff}} \mathrm{id}_F$*.*
4. $\mathrm{Inn}_T(V \circ_{\mathrm{aff}} W) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(V) \circ_{\mathrm{aff}} \mathrm{Inn}_T(W)$*.*
5. $\iota_F(\mathrm{Inn}_T(V)) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(\iota_F(V))$*.*

Proof. Fix the concrete exit package of Definition 45.662. It fixes the carrier AffAut_F, the product $\circ_{\mathrm{aff}}$, the identity id_F, the inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$ over Definition 45.617.

Certification of $\mathrm{Inn}_T(V)$ is closure under the two displayed affine products. Apply Definition 45.556 to compose T with V, then compose the result with $\iota_F(T)$ supplied by the canonical inverse row.

Classifier respect follows by transporting $V \equiv_{\mathrm{aff}} W$ through the fixed left factor T and the fixed right factor $\iota_F(T)$. This is the operation congruence field of Definition 45.553, with the same classifier chosen by the canonical exit package.

For the identity clause, use Definition 45.554 to reassociate $T \circ_{\mathrm{aff}} \mathrm{id}_F \circ_{\mathrm{aff}} \iota_F(T)$. The unit and inverse laws from Definition 45.556 then identify the middle product with $T \circ_{\mathrm{aff}} \iota_F(T)$ and hence with id_F under $\equiv_{\mathrm{aff}}$.

For products, reassociate $T \circ_{\mathrm{aff}} (V \circ_{\mathrm{aff}} W) \circ_{\mathrm{aff}} \iota_F(T)$ so that the inverse-product pair $\iota_F(T) \circ_{\mathrm{aff}} T$ occurs between the transported V and W terms. The group inverse law collapses that pair to id_F, and associativity returns the classifier equality

$$\mathrm{Inn}_T(V \circ_{\mathrm{aff}} W) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(V) \circ_{\mathrm{aff}} \mathrm{Inn}_T(W).$$

For inverse preservation, apply Definition 45.671 to the two products in $T \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(T)$. The reverse-order inverse is $\iota_F(\iota_F(T)) \circ_{\mathrm{aff}} \iota_F(V) \circ_{\mathrm{aff}} \iota_F(T)$. The double-inverse clause of that theorem identifies $\iota_F(\iota_F(T))$ with T, and operation congruence transports the comparison to

$$\iota_F(\mathrm{Inn}_T(V)) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(\iota_F(V)).$$

□

Per-object template: traditional target inner automorphism action; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations composition closure, classifier congruence, unit, product, and inverse preservation; ledger affine coefficients, product witnesses, inverse witnesses, and conjugating element; core theorem list Definition 45.662, Definition 45.556, Definition 45.554, Definition 45.553, and Definition 45.671; exactness predicate inner-conjugation action laws; standard bridge ordinary inner automorphisms; status $\mathsf{ProofTarget}$.

Theorem 45.676 (RatUp FieldUp affine inner action composition exit)**.** *For Definition 45.617, let $T, U, V : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. For a certified affine automorphism S, define*

$$\mathrm{Inn}_S(V) = S \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(S),$$

where $\iota_F(-)$ is the explicit inverse operation supplied by Definition 45.662. Then the inner affine transformations obey the identity and composition laws in the conjugating parameter:

$$\mathrm{Inn}_{\mathrm{id}_F}(V) \equiv_{\mathrm{aff}} V.$$

$$\mathrm{Inn}_T(\mathrm{Inn}_U(V)) \equiv_{\mathrm{aff}} \mathrm{Inn}_{T \circ_{\mathrm{aff}} U}(V).$$

Proof. Fix the concrete affine package from Definition 45.662. It fixes the carrier AffAut_F, the identity id_F, the concrete product $\circ_{\mathrm{aff}}$, the inverse operation $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$.

For the identity parameter, expand $\mathrm{Inn}_{\mathrm{id}_F}(V)$ as $\mathrm{id}_F \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(\mathrm{id}_F)$. The unit rows and inverse rows of the fixed package identify $\iota_F(\mathrm{id}_F)$ with id_F, and associativity from Definition 45.554 removes the two unit factors. The classifier conclusion is $\mathrm{Inn}_{\mathrm{id}_F}(V) \equiv_{\mathrm{aff}} V$.

For composition of conjugating parameters, first use Definition 45.675 to certify the inner term $\mathrm{Inn}_U(V)$ and to transport classifier equalities through the outer conjugation by T.

After expansion, associativity rewrites the left side to the common affine word

$$T \circ_{\mathrm{aff}} U \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(U) \circ_{\mathrm{aff}} \iota_F(T).$$

Apply Definition 45.671 to $T \circ_{\mathrm{aff}} U$. It identifies $\iota_F(T \circ_{\mathrm{aff}} U)$ with $\iota_F(U) \circ_{\mathrm{aff}} \iota_F(T)$ under the same classifier. Operation congruence and classifier transitivity in the fixed package replace the final two inverse factors by $\iota_F(T \circ_{\mathrm{aff}} U)$.

A final use of Definition 45.554 repacks the word as $(T \circ_{\mathrm{aff}} U) \circ_{\mathrm{aff}} V \circ_{\mathrm{aff}} \iota_F(T \circ_{\mathrm{aff}} U)$, which is $\mathrm{Inn}_{T \circ_{\mathrm{aff}} U}(V)$. □

Per-object template: traditional target inner action of the affine automorphism group; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations identity action, composition action, inverse antihomomorphism, associativity, and classifier congruence; ledger conjugating factors, product witnesses, and inverse witnesses; core theorem list Definition 45.662, Definition 45.675, Definition 45.671, and Definition 45.554; exactness predicate concrete inner-action law; standard bridge ordinary conjugation action; status $\mathsf{ProofTarget}$.

Definition 45.677 (RatUp FieldUp affine center predicate)**.** *For Definition 45.617, let $Z : \mathsf{AffAut}_F$ be a certified affine automorphism in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. The concrete affine center predicate is*

$$\mathsf{Center}_F(Z) :\Longleftrightarrow \forall T : \mathsf{AffAut}_F,\ Z \circ_{\mathrm{aff}} T \equiv_{\mathrm{aff}} T \circ_{\mathrm{aff}} Z.$$

Here $\circ_{\mathrm{aff}}$ and $\equiv_{\mathrm{aff}}$ are the concrete affine composition and classifier supplied by Definition 45.662.

Theorem 45.678 (RatUp FieldUp affine center fixed-point criterion)**.** *For Definition 45.617, let $Z : \mathsf{AffAut}_F$ be a certified affine automorphism in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let Inn_T be the inner affine action from Definition 45.675. Then*

$$\mathsf{Center}_F(Z) \Longleftrightarrow \forall T : \mathsf{AffAut}_F,\ \mathrm{Inn}_T(Z) \equiv_{\mathrm{aff}} Z.$$

Proof. Fix the concrete exit action package of Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, id_F, the inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$.

For the forward direction, assume $\mathsf{Center}_F(Z)$ and fix $T : \mathsf{AffAut}_F$. The center equation rewrites $T \circ_{\mathrm{aff}} Z$ as $Z \circ_{\mathrm{aff}} T$ under $\equiv_{\mathrm{aff}}$. Substitute this into $\mathrm{Inn}_T(Z) = (T \circ_{\mathrm{aff}} Z) \circ_{\mathrm{aff}} \iota_F(T)$, reassociate by Definition 45.554, and cancel the trailing product $T \circ_{\mathrm{aff}} \iota_F(T)$ by the inverse law in Definition 45.556. The right identity row then identifies the result with Z.

For the reverse direction, assume $\forall T : \mathsf{AffAut}_F,\ \mathrm{Inn}_T(Z) \equiv_{\mathrm{aff}} Z$ and fix T. Apply the hypothesis to T, right-compose the displayed classifier equation by T, and use classifier congruence from Definition 45.553. After reassociation, the product $\iota_F(T) \circ_{\mathrm{aff}} T$ collapses by the inverse law, so the transported equation gives $T \circ_{\mathrm{aff}} Z \equiv_{\mathrm{aff}} Z \circ_{\mathrm{aff}} T$. Symmetry of the classifier presents it in the order required by $\mathsf{Center}_F(Z)$. □

Per-object template: traditional target center of the affine automorphism group; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations inner action exactness, affine associativity, inverse laws, and classifier congruence; ledger conjugating element and inverse witness provenance; core theorem list Definition 45.662, Definition 45.675, and Definition 41.76; exactness predicate center iff fixed by all inner conjugations; standard bridge ordinary group-center criterion; status $\mathsf{ProofTarget}$.

Definition 45.679 (Concrete RatUp FieldUp affine automorphism group instance)**.** *For Definition 45.617, the concrete affine automorphism group instance has carrier AffAut_F, multiplication $\circ_{\mathrm{aff}}$, unit id_F, inverse $\iota_F(-)$, and classifier $\equiv_{\mathrm{aff}}$. The carrier, composition, unit, and inverse rows are supplied by Definition 45.662, and the classifier, congruence, associativity, identity, inverse, and cancellation obligations are supplied by Definition 45.661.*

Theorem 45.680 (RatUp FieldUp affine commutator fixedness criterion)**.** *For Definition 45.617, let $T, S : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. With the affine group instance of Definition 45.679, define*

$$\mathsf{AffComm}_F(T, S) := (T \circ_{\mathrm{aff}} S) \circ_{\mathrm{aff}} (\iota_F(T) \circ_{\mathrm{aff}} \iota_F(S)).$$

Then the affine commutator is trivial exactly when the two affine automorphisms commute:

$$\mathsf{AffComm}_F(T, S) \equiv_{\mathrm{aff}} \mathrm{id}_F \iff T \circ_{\mathrm{aff}} S \equiv_{\mathrm{aff}} S \circ_{\mathrm{aff}} T.$$

Equivalently, the same commutator is trivial exactly when S is fixed by inner conjugation by T:

$$\mathsf{AffComm}_F(T, S) \equiv_{\mathrm{aff}} \mathrm{id}_F \iff \mathrm{Inn}_T(S) \equiv_{\mathrm{aff}} S.$$

Proof. Fix the affine group instance of Definition 45.679. It reads the group carrier as AffAut_F, the product as $\circ_{\mathrm{aff}}$, the unit as id_F, the inverse as $\iota_F(-)$, and the classifier as $\equiv_{\mathrm{aff}}$.

Instantiate Definition 41.82 in this concrete group package with $a = T$ and $b = S$. Its history-level commutator becomes the displayed term $\mathsf{AffComm}_F(T, S)$, and its classifier statement becomes the first displayed equivalence.

It remains to compare commutation with fixedness under inner conjugation. Expand $\mathrm{Inn}_T(S)$ as $T \circ_{\mathrm{aff}} S \circ_{\mathrm{aff}} \iota_F(T)$. If $T \circ_{\mathrm{aff}} S \equiv_{\mathrm{aff}} S \circ_{\mathrm{aff}} T$, compose the comparison on the right by $\iota_F(T)$, reassociate, and collapse $T \circ_{\mathrm{aff}} \iota_F(T)$ to id_F. The right identity law then gives $\mathrm{Inn}_T(S) \equiv_{\mathrm{aff}} S$.

Conversely, assume $\mathrm{Inn}_T(S) \equiv_{\mathrm{aff}} S$. Compose both sides on the right by T, reassociate the left side, and collapse $\iota_F(T) \circ_{\mathrm{aff}} T$ to id_F. The right identity law gives $T \circ_{\mathrm{aff}} S \equiv_{\mathrm{aff}} S \circ_{\mathrm{aff}} T$.

All transports use operation congruence, associativity, inverse laws, and identity laws from the affine action certificate. The cancellation steps are the concrete common-factor exactness clauses of Definition 45.670. □

Theorem 45.681 (RatUp FieldUp affine threefold inverse antihomomorphism exit)**.** *For Definition 45.617, let $T, U, V : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine composition, let $\equiv_{\mathrm{aff}}$ be the concrete affine classifier, and let $\iota_F(-)$ be the explicit inverse operation supplied by the same exit rows. Then the classifier identifies the inverse of the left associated threefold composite with the reversed threefold composite of explicit inverses:*

$$\iota_F((V \circ_{\mathrm{aff}} U) \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} (\iota_F(T) \circ_{\mathrm{aff}} \iota_F(U)) \circ_{\mathrm{aff}} \iota_F(V).$$

It also identifies the same inverse with the right associated reversed form:

$$\iota_F((V \circ_{\mathrm{aff}} U) \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} \iota_F(T) \circ_{\mathrm{aff}} (\iota_F(U) \circ_{\mathrm{aff}} \iota_F(V)).$$

Proof. First apply Theorem 45.671 to the pair $((V \circ_{\mathrm{aff}} U), T)$. This gives

$$\iota_F((V \circ_{\mathrm{aff}} U) \circ_{\mathrm{aff}} T) \equiv_{\mathrm{aff}} \iota_F(T) \circ_{\mathrm{aff}} \iota_F(V \circ_{\mathrm{aff}} U).$$

Next apply Theorem 45.671 to the pair (V, U). This gives

$$\iota_F(V \circ_{\mathrm{aff}} U) \equiv_{\mathrm{aff}} \iota_F(U) \circ_{\mathrm{aff}} \iota_F(V).$$

Use the affine operation congruence supplied by Theorem 45.661 to compose the fixed left factor $\iota_F(T)$ with this second classifier relation. Transitivity of the same classifier then identifies the original inverse with $(\iota_F(T) \circ_{\mathrm{aff}} \iota_F(U)) \circ_{\mathrm{aff}} \iota_F(V)$, which is the first displayed conclusion.

Finally use the affine group and threefold composition rows from Theorem 45.563. Those rows certify that the two bracketings of the reversed certified affine threefold composite have the same concrete affine classifier. Applying transitivity once more gives the right associated displayed form. □

Per-object template: traditional target threefold affine inverse antihomomorphism; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability binary inverse antihomomorphism, composition congruence, and affine associativity; ledger three affine factors and inverse witnesses; core theorem Definition 45.681; exactness predicate threefold inverse reversal under the concrete affine classifier; standard bridge ordinary affine automorphism group; status $\mathsf{ProofTarget}$.

Theorem 45.682 (RatUp FieldUp affine commutator detects commutation)**.** *For Definition 45.617, let $T, U : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine product, let id_F be the concrete identity, let $\iota_F(-)$ be the inverse row, and let $\equiv_{\mathrm{aff}}$ be the affine classifier supplied by Definition 45.662. Define*

$$\mathsf{Comm}_F(T, U) := (T \circ_{\mathrm{aff}} U) \circ_{\mathrm{aff}} (\iota_F(T) \circ_{\mathrm{aff}} \iota_F(U)).$$

Then the concrete affine commutator is classifier-trivial exactly when the two affine factors commute:

$$\mathsf{Comm}_F(T, U) \equiv_{\mathrm{aff}} \mathrm{id}_F \iff T \circ_{\mathrm{aff}} U \equiv_{\mathrm{aff}} U \circ_{\mathrm{aff}} T.$$

Proof. Fix Theorem 45.662. It supplies the same affine composition, inverse operation, identity row, and classifier used in the displayed statement.

Read those rows as the concrete affine group package certified by Definition 45.558. The package gives the same carrier, product, identity, inverse operation, classifier, associativity, identity laws, inverse laws, and affine operation congruence.

Instantiate Definition 41.82 in this concrete group with $a = T$ and $b = U$. The checked theorem gives the equivalence between triviality of $\mathsf{mul}(\mathsf{mul}(T, U), \mathsf{mul}(\mathsf{inv}(T), \mathsf{inv}(U)))$ and commutation of $\mathsf{mul}(T, U)$ with $\mathsf{mul}(U, T)$.

It remains to match the displayed affine notation with this group notation. The operation, identity, inverse row, and classifier are the same fields of the fixed affine package. Affine operation congruence transports classifier equalities through the two product slots, and affine associativity matches the displayed bracketing of $\mathsf{Comm}_F(T, U)$ with the bracketing used by the $\mathsf{Group}^{\uparrow}$ theorem.

Reading the group equivalence through this notation gives the forward direction from a trivial displayed $\mathsf{Comm}_F(T, U)$ to affine commutation.

Reading the same equivalence in reverse gives the converse: affine commutation transports through the classifier-congruence rows to the displayed commutator relation $\mathsf{Comm}_F(T, U) \equiv_{\mathrm{aff}} \mathrm{id}_F$. □

Per-object template: traditional target affine commutator criterion; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus $\mathsf{Group}^{\uparrow}$; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability obligations affine group laws, inverse-product reversal, operation congruence, and identity laws; ledger the two affine factors and inverse witnesses; core theorem list the canonical exit action and inverse package Definition 45.662, Definition 45.558, and Definition 41.82; exactness predicate trivial affine commutator iff affine commutation; standard bridge ordinary affine automorphism commutator criterion; status $\mathsf{ProofTarget}$.

Theorem 45.683 (RatUp FieldUp affine commutator centralizer exactness)**.** *For Definition 45.617, let $A, B : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine composition, let $\iota_F(-)$ be the explicit inverse row, and let $\equiv_{\mathrm{aff}}$ be the affine classifier. Define*

$$\mathrm{Inn}_A(B) := (A \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} \iota_F(A).$$

Then the commutation condition is exactly the fixed-point condition for conjugation by A:

$$A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A \Longleftrightarrow \mathrm{Inn}_A(B) \equiv_{\mathrm{aff}} B.$$

Proof. Fix the canonical exit package of Definition 45.662. This fixes the carrier AffAut_F, the collected-coefficient composite $\circ_{\mathrm{aff}}$, the explicit inverse row $\iota_F(-)$, the identity id_F, and the classifier $\equiv_{\mathrm{aff}}$.

For the forward direction, compose $A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A$ on the right by $\iota_F(A)$. The affine operation congruence and associativity rows transport the comparison to

$$(A \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} \iota_F(A) \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} (A \circ_{\mathrm{aff}} \iota_F(A)).$$

The right inverse and right identity rows reduce the right side to B, giving $\mathrm{Inn}_A(B) \equiv_{\mathrm{aff}} B$.

For the reverse direction, use Definition 45.672 with $X = B$ and target B. The displayed fixed-point condition for $\mathrm{Inn}_A(B)$ gives

$$B \equiv_{\mathrm{aff}} (\iota_F(A) \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} A.$$

Compose on the left by A, then use affine associativity, inverse laws, identity laws, and cancellation exactness from Definition 45.661. The resulting comparison is exactly $A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A$. □

Lean4 ✓ `BEDC.Derived.FieldUp.field empty unit conjugation fixed point commutation iff`

Per-object template: traditional target affine commutator centralizer criterion; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations affine group certificate, operation congruence, inverse-product reversal, conjugation equation, and cancellation exactness; ledger two affine factors, inverse witnesses, commutator composite, and inner conjugation composite; core theorem Definition 45.683; exactness predicate commutation iff conjugation fixed point; standard bridge ordinary affine automorphism centralizer criterion; status $\mathsf{ProofTarget}$.

Definition 45.684 (RatUp FieldUp affine centralizer predicate)**.** *For Definition 45.617, let $A, B : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $\circ_{\mathrm{aff}}$ be the concrete affine product and let $\equiv_{\mathrm{aff}}$ be the concrete affine classifier supplied by Definition 45.679. The concrete affine centralizer predicate of A at B is*

$$\mathsf{Cent}_F(A; B) :\Longleftrightarrow A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A.$$

Theorem 45.685 (RatUp FieldUp affine centralizer product and inverse closure)**.** *For Definition 45.617, let $A, B, C, B' : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. With Cent_F as in Definition 45.684, the centralizer of A is closed under the concrete affine group operations and under affine classifier transport:*

1. $\mathsf{Cent}_F(A; \mathrm{id}_F)$.
2. *If* $\mathsf{Cent}_F(A; B)$ *and* $\mathsf{Cent}_F(A; C)$*, then* $\mathsf{Cent}_F(A; B \circ_{\mathrm{aff}} C)$.
3. *If* $\mathsf{Cent}_F(A; B)$*, then* $\mathsf{Cent}_F(A; \iota_F(B))$.

4. *If* $B \equiv_{\mathrm{aff}} B'$, *then*

$$\mathsf{Cent}_F(A; B) \iff \mathsf{Cent}_F(A; B').$$

Proof. Fix the affine group instance of Definition 45.679. Its action certificate is Definition 45.661, which supplies associativity, identity, inverse, congruence, symmetry, and transitivity for the concrete affine classifier.

For the identity clause, the left and right identity rows identify $A \circ_{\mathrm{aff}} \mathrm{id}_F$ and $\mathrm{id}_F \circ_{\mathrm{aff}} A$ with A. Symmetry and transitivity of $\equiv_{\mathrm{aff}}$ give the displayed centralizer predicate.

For product closure, convert the two hypotheses to inner fixed-point statements by Definition 45.683. Apply Definition 45.676 to identify $\mathrm{Inn}_A(B \circ_{\mathrm{aff}} C)$ with $\mathrm{Inn}_A(B) \circ_{\mathrm{aff}} \mathrm{Inn}_A(C)$. Congruence transports the two fixed-point equations through the product, so the composite inner image is classifier-equal to $B \circ_{\mathrm{aff}} C$. Applying centralizer exactness in reverse gives $\mathsf{Cent}_F(A; B \circ_{\mathrm{aff}} C)$.

For inverse closure, start from $\mathsf{Cent}_F(A; B)$ and again use Definition 45.683 to read it as $\mathrm{Inn}_A(B) \equiv_{\mathrm{aff}} B$. Apply Definition 45.671 to transport this fixed-point equation through the explicit inverse row. The inverse, associativity, and identity clauses in the action certificate reduce the inner image of $\iota_F(B)$ to $\iota_F(B)$, and centralizer exactness returns $\mathsf{Cent}_F(A; \iota_F(B))$.

For classifier transport, assume $B \equiv_{\mathrm{aff}} B'$. Operation congruence composes this relation with the fixed factor A on the left and on the right. Symmetry and transitivity then carry $A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A$ exactly to $A \circ_{\mathrm{aff}} B' \equiv_{\mathrm{aff}} B' \circ_{\mathrm{aff}} A$, and the same argument with the comparison reversed gives the displayed equivalence. □

Definition 45.686 (RatUp FieldUp affine centralizer carrier)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $\circ_{\mathrm{aff}}$ *and* $\equiv_{\mathrm{aff}}$ *be the affine composition and classifier supplied by Definition 45.662. The concrete affine centralizer carrier of* A *is the predicate*

$$C_F(A)(X) :\iff X : \mathsf{AffAut}_F \quad \textit{and} \quad X \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} X.$$

Lean4 (definition) `BEDC.Derived.FieldUp.RatupFieldupAffineCentralizerCarrier`

Theorem 45.687 (RatUp FieldUp affine centralizer subgroup)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the carrier from Definition 45.686. Then* $C_F(A)$ *is the concrete affine centralizer subgroup of* A*:*

1. $C_F(A)(\mathrm{id}_F)$.
2. *If* $C_F(A)(X)$ *and* $C_F(A)(Y)$*, then* $C_F(A)(X \circ_{\mathrm{aff}} Y)$.
3. *If* $C_F(A)(X)$*, then* $C_F(A)(\iota_F(X))$.
4. *For every* $X : \mathsf{AffAut}_F$*,*

$$C_F(A)(X) \iff \mathsf{Comm}_F(X, A) \equiv_{\mathrm{aff}} \mathrm{id}_F.$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAffineCentralizerSubgroup`

Proof. Fix the canonical affine exit action and inverse package from Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, id_F, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$.

Instantiate Definition 45.558 on these rows. Its identity, associativity, inverse, and classifier-congruence clauses make the certified affine automorphisms a concrete $\mathsf{Group}^{\uparrow}$ object.

For the identity element, the left and right unit laws identify $\mathrm{id}_F \circ_{\mathrm{aff}} A$ and $A \circ_{\mathrm{aff}} \mathrm{id}_F$ with A. Transitivity and symmetry of $\equiv_{\mathrm{aff}}$ give $C_F(A)(\mathrm{id}_F)$.

For products, assume $C_F(A)(X)$ and $C_F(A)(Y)$. Reassociate $(X \circ_{\mathrm{aff}} Y) \circ_{\mathrm{aff}} A$, use the commuting witness for Y to move A left across Y, reassociate again, and use the commuting witness for X to move A left across X. The classifier-congruence rows then present the result as $A \circ_{\mathrm{aff}} (X \circ_{\mathrm{aff}} Y)$.

For inverses, assume $C_F(A)(X)$. Apply Definition 45.671 to the two products $X \circ_{\mathrm{aff}} A$ and $A \circ_{\mathrm{aff}} X$. The commuting witness transports through inverse antihomomorphism, and the group inverse rows reduce the resulting equation to $\iota_F(X) \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} \iota_F(X)$.

Finally apply Definition 45.682 with the ordered pair (X, A). Its displayed equivalence is exactly the assertion that membership in $C_F(A)$ is the same as $\mathsf{Comm}_F(X, A) \equiv_{\mathrm{aff}} \mathrm{id}_F$. □

Per-object template: traditional target affine centralizer subgroup; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus $\mathsf{Group}^{\uparrow}$ commutator criterion; carrier $C_F(A)$ over AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, and commutator characterization; ledger fixed element, commuting witness, product witness, and inverse witness; core theorem Definition 45.687; exactness predicate centralizer membership iff trivial commutator; standard bridge ordinary affine centralizer subgroup; status $\mathsf{ProofTarget}$.

Theorem 45.688 (RatUp FieldUp affine center as centralizer intersection)**.** *For Definition 45.617, let $Z : \mathsf{AffAut}_F$ be a certified affine automorphism in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Then the concrete affine center is the intersection of all concrete affine centralizers:*

$$\mathsf{Center}_F(Z) \iff \forall A : \mathsf{AffAut}_F,\ C_F(A)(Z).$$

Equivalently, the same center predicate is the condition that every concrete affine commutator with Z is classifier-trivial:

$$\mathsf{Center}_F(Z) \iff \forall A : \mathsf{AffAut}_F,\ \mathsf{Comm}_F(Z, A) \equiv_{\mathrm{aff}} \mathrm{id}_F.$$

Proof. Fix the canonical affine exit package from Definition 45.662. This fixes the carrier AffAut_F, the affine product $\circ_{\mathrm{aff}}$, the identity id_F, the inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$.

For the first forward direction, assume $\mathsf{Center}_F(Z)$ and fix $A : \mathsf{AffAut}_F$. Unfold Definition 45.677 at the parameter A to obtain $Z \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} Z$. Together with the already fixed carrier witness for Z, this is exactly $C_F(A)(Z)$ by Definition 45.686.

For the first reverse direction, assume $\forall A : \mathsf{AffAut}_F,\ C_F(A)(Z)$ and fix $A : \mathsf{AffAut}_F$. Unfold the centralizer carrier at A and Z; its commuting component gives $Z \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} Z$. Since A was arbitrary, this is precisely Definition 45.677.

It remains to compare centralizer membership with commutator triviality. For each fixed $A : \mathsf{AffAut}_F$, apply the fourth clause of Definition 45.687 with $X = Z$. It gives

$$C_F(A)(Z) \iff \mathsf{Comm}_F(Z, A) \equiv_{\mathrm{aff}} \mathrm{id}_F.$$

Compose this pointwise equivalence with the first equivalence already proved. Classifier symmetry is used only to present the commuting equation in the displayed order when the centralizer carrier is read back as the center predicate. The result is the second displayed equivalence. □

Per-object template: traditional target center as intersection of centralizers; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine automorphism exit plus $\mathsf{Group}^{\uparrow}$ commutator criterion; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability center predicate, centralizer carrier, commutator exactness, and classifier symmetry; ledger quantified centralizer witness and commutator spine; core theorem list the center definition Definition 45.677, centralizer carrier Definition 45.686, and centralizer subgroup theorem Definition 45.687; exactness predicate center iff all centralizer memberships iff all commutators trivial; standard bridge ordinary group center-centralizer theorem; status $\mathsf{ProofTarget}$.

Theorem 45.689 (RatUp FieldUp affine centralizer fixed-point criterion)**.** *For Definition 45.617, let $A, B : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Define the inner affine action of A on B and the concrete affine commutator by*

$$\mathrm{Inn}_A(B) = A \circ_{\mathrm{aff}} B \circ_{\mathrm{aff}} \iota_F(A),$$

$$\mathsf{Comm}_F(A, B) := (A \circ_{\mathrm{aff}} B) \circ_{\mathrm{aff}} (\iota_F(A) \circ_{\mathrm{aff}} \iota_F(B)).$$

Here $\circ_{\mathrm{aff}}$, $\iota_F(-)$, id_F, and $\equiv_{\mathrm{aff}}$ are the concrete affine composition, inverse, identity, and classifier supplied by Definition 45.662. Then the centralizer fixed-point condition, commutator triviality, and ordinary affine commutation coincide:

$$\mathrm{Inn}_A(B) \equiv_{\mathrm{aff}} B \iff \mathsf{Comm}_F(A, B) \equiv_{\mathrm{aff}} \mathrm{id}_F \iff A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A.$$

Proof. Fix the canonical affine exit package from Definition 45.662. This fixes the carrier AffAut_F, the affine product $\circ_{\mathrm{aff}}$, the unit id_F, the inverse row $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$.

Apply Definition 45.683 to the same certified affine automorphisms A and B. Reading that equivalence in the reverse direction identifies the fixed-point condition $\mathrm{Inn}_A(B) \equiv_{\mathrm{aff}} B$ with affine commutation $A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A$.

Apply Definition 45.682 to A and B. It identifies $\mathsf{Comm}_F(A, B) \equiv_{\mathrm{aff}} \mathrm{id}_F$ with the same affine commutation condition. Transitivity and symmetry of the fixed classifier combine the two equivalences into the displayed chain. □

Per-object template: traditional target centralizer fixed-point criterion for inner affine automorphisms; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus $\mathsf{Group}^{\uparrow}$; carrier AffAut_F; classifier $\equiv_{\mathrm{aff}}$; stability obligations inner action, commutator criterion, associativity, inverse laws, and classifier congruence; ledger conjugating factor and inverse witness; core theorem list Definition 45.676 and Definition 45.682; exactness predicate centralizer fixed point iff commutator trivial iff commutation; standard bridge ordinary affine centralizers; status $\mathsf{ProofTarget}$.

Theorem 45.690 (RatUp FieldUp affine centralizers are conjugation-equivariant)**.** *For Definition 45.617, let $A, T, X : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Define the inner affine conjugation action by*

$$\mathrm{Inn}_T(Y) = (T \circ_{\mathrm{aff}} Y) \circ_{\mathrm{aff}} \iota_F(T).$$

Let $C_F(A)(X)$ be the centralizer membership predicate of Definition 45.686. Then centralizer membership is transported by inner affine conjugation:

$$C_F(A)(X) \iff C_F(\mathrm{Inn}_T(A))(\mathrm{Inn}_T(X)).$$

Equivalently,

$$X \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} X \iff \mathrm{Inn}_T(X) \circ_{\mathrm{aff}} \mathrm{Inn}_T(A) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(A) \circ_{\mathrm{aff}} \mathrm{Inn}_T(X).$$

Proof. Fix the canonical affine group package from Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, id_F, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$, and Definition 45.661 supplies the associativity, inverse, cancellation, and classifier congruence rows used below.

For the forward direction, assume $C_F(A)(X)$, namely $X \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} X$. Left-compose and right-compose this classifier equation by the conjugating factor T and its inverse witness $\iota_F(T)$. The classifier congruence row keeps equality stable under these certified affine products.

Reassociate the two transported composites by Definition 45.554. The adjacent factors $\iota_F(T)\circ_{\mathrm{aff}} T$ collapse by the inverse law in Definition 45.556, and the identity rows leave exactly

$$\mathrm{Inn}_T(X) \circ_{\mathrm{aff}} \mathrm{Inn}_T(A) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(A) \circ_{\mathrm{aff}} \mathrm{Inn}_T(X).$$

This is $C_F(\mathrm{Inn}_T(A))(\mathrm{Inn}_T(X))$ by Definition 45.686. The same normal form is the concrete affine conjugation exactness asserted by Definition 45.675.

For the reverse direction, apply the forward construction with $\iota_F(T)$ as the conjugating automorphism to the centralizer witness for $\mathrm{Inn}_T(A)$ and $\mathrm{Inn}_T(X)$. The inverse antihomomorphism and double-inverse clauses of Definition 45.671 identify the resulting double conjugates with A and X under $\equiv_{\mathrm{aff}}$.

Finally use classifier transitivity and symmetry to replace the double conjugate centralizer equation by $X \circ_{\mathrm{aff}} A \equiv_{\mathrm{aff}} A \circ_{\mathrm{aff}} X$. This is precisely $C_F(A)(X)$, so the two displayed centralizer-membership conditions are equivalent. □

Per-object template: traditional target conjugation action on centralizers; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus AffAut_F; carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability conjugation exactness, inverse antihomomorphism, associativity, and classifier congruence; ledger conjugating factor and inverse witness; core theorem list Definition 45.686, Definition 45.689, Definition 45.675, Definition 45.671, and Definition 45.661; exactness predicate centralizer membership transported by inner affine action; standard bridge ordinary centralizer conjugation; status ProofTarget.

Theorem 45.691 (RatUp FieldUp affine centralizer inner transport)**.** *For Definition 45.617, let $A, B, T : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let Cent_F be the centralizer predicate from Definition 45.684, and define*

$$\mathrm{Inn}_T(X) = T \circ_{\mathrm{aff}} X \circ_{\mathrm{aff}} \iota_F(T).$$

If $\mathsf{Cent}_F(A; B)$, then

$$\mathsf{Cent}_F(\mathrm{Inn}_T(A); \mathrm{Inn}_T(B)).$$

Proof. Fix the canonical affine exit package of Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$ for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows.

Expand the hypothesis $\mathsf{Cent}_F(A; B)$ using Definition 45.684. It is the classifier equality

$$A \circ_{\mathrm{aff}} B \equiv_{\mathrm{aff}} B \circ_{\mathrm{aff}} A.$$

Apply the classifier-congruence clause of Definition 45.675 with conjugating factor T. The preceding equality transports to

$$\mathrm{Inn}_T(A \circ_{\mathrm{aff}} B) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(B \circ_{\mathrm{aff}} A).$$

Apply the product-preservation clause of the same theorem to both sides. It identifies the left side with $\mathrm{Inn}_T(A) \circ_{\mathrm{aff}} \mathrm{Inn}_T(B)$ and the right side with $\mathrm{Inn}_T(B) \circ_{\mathrm{aff}} \mathrm{Inn}_T(A)$.

Symmetry and transitivity of $\equiv_{\mathrm{aff}}$ from the affine action certificate combine these two product comparisons with the transported equality. The resulting classifier equality is precisely the definition of $\mathsf{Cent}_F(\mathrm{Inn}_T(A); \mathrm{Inn}_T(B))$. □

Per-object template: traditional target affine centralizer conjugation transport; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit rows plus AffAut_F; carrier certified affine automorphisms; classifier $\equiv_{\mathrm{aff}}$; stability obligations inner-action classifier congruence and product preservation; ledger conjugating automorphism and inverse witness; core theorem list Definition 45.675 and Definition 45.689; exactness predicate centralizer membership transported by inner conjugation; standard bridge ordinary group centralizer conjugacy; status ProofTarget.

Definition 45.692 (RatUp FieldUp affine centralizer normalizer carrier)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object, and let $C_F(A)$ be the carrier from Definition 45.686. The concrete affine normalizer carrier of $C_F(A)$ is the predicate*

$$N_F(C_F(A))(T) :\Longleftrightarrow T : \mathsf{AffAut}_F \quad \textit{and} \quad \forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Longleftrightarrow C_F(A)(\mathrm{Conj}_T(X)),$$

where

$$\mathrm{Conj}_T(X) := (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T)$$

uses the affine product and inverse row fixed by Definition 45.662.

Theorem 45.693 (RatUp FieldUp affine centralizer normalizer inclusion)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the carrier from Definition 45.686, and let $N_F(C_F(A))$ be the carrier from Definition 45.692. Then every centralizer element normalizes the centralizer in the two-sided sense:*

$$C_F(A)(T) \Rightarrow N_F(C_F(A))(T).$$

Equivalently, if T commutes with A, then inner affine conjugation by T both preserves and reflects centralizer membership:

$$C_F(A)(T) \Rightarrow \forall X : \mathsf{AffAut}_F, \quad C_F(A)(X) \Longleftrightarrow C_F(A)((T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T)).$$

Proof. Fix the canonical affine exit action from Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, id_F, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$ for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

Assume $C_F(A)(T)$ and let $X : \mathsf{AffAut}_F$ be fixed. Apply Definition 45.687 to the fixed element A. Its inverse closure gives $C_F(A)(\iota_F(T))$.

For preservation, assume $C_F(A)(X)$. Product closure first gives $C_F(A)(T \circ_{\mathrm{aff}} X)$ and then gives $C_F(A)((T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T))$.

The displayed product is the concrete inner conjugate $\mathrm{Conj}_T(X)$ in the notation of Definition 45.692. The carrier certification and classifier stability of this inner conjugate are exactly the action clauses supplied by Definition 45.675.

For reflection, assume $C_F(A)((T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T))$. Apply the same preservation argument with $\iota_F(T)$ as the conjugating centralizer element and $\mathrm{Conj}_T(X)$ as the element being transported. Product closure gives $C_F(A)(\mathrm{Conj}_{\iota_F(T)}(\mathrm{Conj}_T(X)))$.

The affine conjugation equation exactness of Definition 45.672, together with the inverse row fixed above, identifies $\mathrm{Conj}_{\iota_F(T)}(\mathrm{Conj}_T(X))$ with X under $\equiv_{\mathrm{aff}}$. The classifier-stability row from Definition 45.675 transfers the centralizer membership back to $C_F(A)(X)$.

Reading the two directions through Definition 45.683 gives the equivalent commutator form: conjugation preserves and reflects the condition that the commutator with A is trivial under the concrete affine classifier. This is precisely $N_F(C_F(A))(T)$. □

Per-object template: traditional target centralizer normalizer inclusion; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ and $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability two-sided centralizer preservation, inverse, product, and inner-conjugation closure; ledger fixed element, conjugating element, commuting witness, inverse witness, and double-conjugation witness; core theorem list Definition 45.687, Definition 45.672, Definition 45.675, and Definition 45.683; exactness predicate normalizer records preservation and reflection of centralizer membership under concrete affine conjugation; standard bridge ordinary centralizer-normalizer inclusion; status $\mathsf{ProofTarget}$.

Theorem 45.694 (RatUp FieldUp affine center normal subgroup)**.** *For Definition 45.617, the concrete affine center predicate* Center_F *on certified affine automorphisms forms a normal subgroup package inside* AffAut_F*. Namely:*

1. $\mathsf{Center}_F(\mathrm{id}_F)$.
2. *If* $\mathsf{Center}_F(Z)$ *and* $\mathsf{Center}_F(W)$*, then* $\mathsf{Center}_F(Z \circ_{\mathrm{aff}} W)$.
3. *If* $\mathsf{Center}_F(Z)$*, then* $\mathsf{Center}_F(\iota_F(Z))$.
4. *If* $Z \equiv_{\mathrm{aff}} W$ *and* $\mathsf{Center}_F(Z)$*, then* $\mathsf{Center}_F(W)$.
5. *If* $\mathsf{Center}_F(Z)$*, then for every* $T : \mathsf{AffAut}_F$*,* $\mathsf{Center}_F(\mathrm{Inn}_T(Z))$.

Proof. Fix the canonical exit action and inverse package of Definition 45.662. This fixes the carrier AffAut_F, the product $\circ_{\mathrm{aff}}$, the identity id_F, the inverse row $\iota_F(-)$, the classifier $\equiv_{\mathrm{aff}}$, and the inner action Inn_T from Definition 45.675.

The identity member is central by the identity rows in Definition 45.556. For product closure, fix a test automorphism T and central elements Z, W. Associativity from Definition 45.554 exposes $W \circ_{\mathrm{aff}} T$ and then $Z \circ_{\mathrm{aff}} T$ as adjacent subproducts. Apply the two center equations and transport both replacements by Definition 45.553; the same associativity theorem repacks the result as $T \circ_{\mathrm{aff}} (Z \circ_{\mathrm{aff}} W)$.

For inverse closure, use Definition 45.678. If $\mathsf{Center}_F(Z)$, then every $\mathrm{Inn}_T(Z)$ is classifier-equal to Z. The inverse-preservation clause of Definition 45.675, together with classifier congruence, transports this fixed-point equation through $\iota_F(-)$ and gives $\mathrm{Inn}_T(\iota_F(Z)) \equiv_{\mathrm{aff}} \iota_F(Z)$ for every T. The fixed-point criterion returns $\mathsf{Center}_F(\iota_F(Z))$.

Classifier transport is pointwise: if $Z \equiv_{\mathrm{aff}} W$, then operation congruence replaces Z by W on both sides of the defining commutation equation for each test automorphism.

Finally assume $\mathsf{Center}_F(Z)$ and fix a conjugating element T. By Definition 45.678, Z is fixed by every inner affine action. For any test element S, Definition 45.676 rewrites $\mathrm{Inn}_S(\mathrm{Inn}_T(Z))$ as $\mathrm{Inn}_{S \circ_{\mathrm{aff}} T}(Z)$, which is classifier-equal to Z. The same criterion also identifies $\mathrm{Inn}_T(Z)$ with Z, and classifier transitivity yields $\mathrm{Inn}_S(\mathrm{Inn}_T(Z)) \equiv_{\mathrm{aff}} \mathrm{Inn}_T(Z)$. Applying the fixed-point criterion once more gives $\mathsf{Center}_F(\mathrm{Inn}_T(Z))$. □

Theorem 45.695 (RatUp FieldUp affine centralizer normalizer subgroup)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the affine centralizer carrier from Definition 45.686, and let* $N_F(C_F(A))$ *be the two-sided normalizer carrier from Definition 45.692. Then* $N_F(C_F(A))$ *is the concrete affine normalizer subgroup of* $C_F(A)$*:*

1. $N_F(C_F(A))(\mathrm{id}_F)$.
2. *If* $N_F(C_F(A))(T)$ *and* $N_F(C_F(A))(U)$*, then* $N_F(C_F(A))(T \circ_{\mathrm{aff}} U)$.
3. *If* $N_F(C_F(A))(T)$*, then* $N_F(C_F(A))(\iota_F(T))$.
4. *If* $N_F(C_F(A))(T)$ *and* $T \equiv_{\mathrm{aff}} U$*, then* $N_F(C_F(A))(U)$.

Proof. Fix the canonical affine exit action and inverse package from Definition 45.662. This fixes AffAut_F, $\circ_{\mathrm{aff}}$, id_F, $\iota_F(-)$, and $\equiv_{\mathrm{aff}}$ for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

For the identity element, fix $X : \mathsf{AffAut}_F$. The identity clause of Definition 45.676 identifies $\mathrm{Conj}_{\mathrm{id}_F}(X)$ with X. Hence centralizer membership is preserved and reflected by identity conjugation, and the identity row of Definition 45.687 supplies the required carrier witness.

For product closure, assume $N_F(C_F(A))(T)$ and $N_F(C_F(A))(U)$, and fix $X : \mathsf{AffAut}_F$. If $C_F(A)(X)$, the two-sided definition first gives $C_F(A)(\mathrm{Conj}_U(X))$ and then gives $C_F(A)(\mathrm{Conj}_T(\mathrm{Conj}_U(X)))$. The composition clause of Definition 45.676 identifies the composite conjugation with conjugation by $T \circ_{\mathrm{aff}} U$, so the forward half gives $C_F(A)(\mathrm{Conj}_{T\circ_{\mathrm{aff}}U}(X))$.

Conversely assume $C_F(A)(\mathrm{Conj}_{T\circ_{\mathrm{aff}}U}(X))$. The same composition clause reads this as $C_F(A)(\mathrm{Conj}_T(\mathrm{Conj}_U(X)$ The reverse half of the two-sided definition for T gives $C_F(A)(\mathrm{Conj}_U(X))$, and the reverse half for U gives $C_F(A)(X)$.

For inverse closure, assume $N_F(C_F(A))(T)$ and fix $X : \mathsf{AffAut}_F$. Let $Y = \mathrm{Conj}_{\iota_F(T)}(X)$. The inverse antihomomorphism package of Definition 45.671 and the conjugation equation exactness of Definition 45.672 identify $\mathrm{Conj}_T(Y)$ with X under $\equiv_{\mathrm{aff}}$.

If $C_F(A)(X)$, transport this membership across the exactness comparison to $C_F(A)(\mathrm{Conj}_T(Y))$. The reverse half of the normalizer equivalence for T, applied to Y, gives $C_F(A)(Y)$. If $C_F(A)(Y)$, the forward half for T and the same double-conjugation exactness return $C_F(A)(X)$. Thus conjugation by $\iota_F(T)$ preserves and reflects centralizer membership, and the inverse row gives the carrier witness for $\iota_F(T)$.

For classifier transport, assume $N_F(C_F(A))(T)$ and $T \equiv_{\mathrm{aff}} U$. For each X, affine operation congruence from Definition 45.553 transports the conjugation word $(T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T)$ to $(U \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(U)$. The preservation and reflection clauses for T therefore give the same two-sided centralizer equivalence for U, and the carrier witness is transported along $T \equiv_{\mathrm{aff}} U$. □

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAffineCentralizerNormalizerSubgroup`

Per-object template: traditional target normalizer subgroup; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine automorphism group plus $\mathsf{Group}^{\uparrow}$; carrier $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, and transport; ledger fixed element and conjugating witnesses; exactness preservation-reflection of centralizer membership; standard bridge ordinary normalizer subgroup; status $\mathsf{ProofTarget}$.

Theorem 45.696 (RatUp FieldUp affine centralizer normalizer action automorphism)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692, and define the restricted action by*

$$\mathrm{Act}_T(X) := \mathrm{Conj}_T(X) = (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$ and $C_F(A)(X), C_F(A)(Y)$, then conjugation by T is an automorphism of the centralizer classifier:

1. $N_F(C_F(A))(\iota_F(T))$.
2. $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.
3. $$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X \quad \textit{and} \quad \mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X.$$
4. $$\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(Y) \Longleftrightarrow X \equiv_{\mathrm{aff}} Y.$$

Proof. The inverse normalizer witness is the inverse-closure row of Definition 45.695. Applied to $N_F(C_F(A))(T)$, it gives $N_F(C_F(A))(\iota_F(T))$.

Carrier preservation for the inverse action is then exactly the preservation part of Definition 45.697, specialized to the normalizer element $\iota_F(T)$ and the centralizer element X. This gives $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.

For cancellation, use Definition 45.673 with conjugating element T and acted element X. Its inverse-action clauses identify the two double conjugations by T and $\iota_F(T)$ with X under $\equiv_{\mathrm{aff}}$, which are precisely the two displayed Act-cancellation laws.

For classifier exactness, apply the same theorem to the pair X, Y. Its classifier-exact conjugation clause gives

$$\mathrm{Conj}_T(X) \equiv_{\mathrm{aff}} \mathrm{Conj}_T(Y) \Longleftrightarrow X \equiv_{\mathrm{aff}} Y.$$

Replacing $\mathrm{Conj}_T(-)$ by the definition of $\mathrm{Act}_T(-)$ gives the stated equivalence on the centralizer carrier. □

Per-object template: traditional target normalizer action by automorphisms; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability inverse action, classifier exactness, and cancellation; ledger fixed element, normalizer witness, acted element, and inverse witness; core theorem list Definition 45.695, Definition 45.697, and Definition 45.673; exactness predicate classifier-exact automorphism action on the centralizer; standard bridge ordinary normalizer action by automorphisms; status ProofTarget.

Theorem 45.697 (RatUp FieldUp affine centralizer normalizer action)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, and let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692. For $T : \mathsf{AffAut}_F$ define the restricted normalizer action on centralizer elements by*

$$\mathrm{Act}_T(X) := \mathrm{Conj}_T(X) = (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$ and $C_F(A)(X)$, then

$$C_F(A)(\mathrm{Act}_T(X)).$$

The restricted action also obeys the identity and composition laws:

$$\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X.$$

For $N_F(C_F(A))(T)$, $N_F(C_F(A))(U)$, and $C_F(A)(X)$,

$$\mathrm{Act}_T(\mathrm{Act}_U(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{T \circ_{\mathrm{aff}} U}(X).$$

Lean4 ✓ `BEDC.Derived.FieldUp.RatupFieldupAffineCentralizerNormalizerAction`

Proof. Unfold the normalizer carrier from Definition 45.692. Its preservation clause says exactly that a normalizing affine automorphism sends each $C_F(A)$-element to another $C_F(A)$-element by the concrete conjugation $\mathrm{Conj}_T(-)$. Since $\mathrm{Act}_T(X)$ is defined to be this conjugate, $N_F(C_F(A))(T)$ and $C_F(A)(X)$ give $C_F(A)(\mathrm{Act}_T(X))$.

For the identity law, apply Definition 45.676 with conjugating parameter id_F and acted element X. The identity clause of that theorem gives $\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X$ under the same affine classifier.

For composition, first use the preservation clause of U to obtain $C_F(A)(\mathrm{Act}_U(X))$. Then use the preservation clause of T to keep the outer acted element inside $C_F(A)$. The composition clause of Definition 45.676, applied to T, U, and X, identifies the two affine words as

$$\mathrm{Act}_T(\mathrm{Act}_U(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{T \circ_{\mathrm{aff}} U}(X).$$

The carrier membership clauses come from the two normalizer hypotheses, while the classifier comparison is the concrete inner-action law already supplied for the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit object. The inverse witnesses are the $\iota_F(T)$ and $\iota_F(U)$ rows fixed by the canonical affine inverse package, and the acted elements remain in the same centralizer carrier throughout. □

Per-object template: traditional target normalizer action on centralizer; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability preservation, identity action, and composition action; ledger fixed element, conjugating elements, acted element, and inverse witnesses; core theorem list Definition 45.693, Definition 45.676, and Definition 45.675; exactness predicate restricted normalizer action; standard bridge ordinary normalizer action on a subgroup; status $\mathsf{ProofTarget}$.

Theorem 45.698 (RatUp FieldUp restricted normalizer action automorphism and kernel exactness)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, and let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692. For $T : \mathsf{AffAut}_F$ define*

$$\mathrm{Act}_T(X) := (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$ and $C_F(A)(X)$, then Act_T is an automorphism of the centralizer carrier with inverse action $\mathrm{Act}_{\iota_F(T)}$:

1. $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.

2.

$$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X.$$

3.

$$\mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X.$$

For every T with $N_F(C_F(A))(T)$, the kernel of the restricted action is exactly the pointwise centralizer of $C_F(A)$:

$$(\forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow \mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X) \Longleftrightarrow (\forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow C_F(X)(T)).$$

Proof. The inverse parameter lies in the same normalizer by the inverse-closure row of Definition 45.695. Applying Definition 45.697 to $\iota_F(T)$ and the centralizer element X gives $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.

For the two inverse-action identities, apply the composition law of Definition 45.697 first to $\iota_F(T)$ after T, and then to T after $\iota_F(T)$. The affine inverse laws from Definition 45.662 identify the parameters $\iota_F(T) \circ_{\mathrm{aff}} T$ and $T \circ_{\mathrm{aff}} \iota_F(T)$ with id_F. The identity-action row of the restricted action then reduces both composites to X under $\equiv_{\mathrm{aff}}$.

For the kernel equivalence, fix X with $C_F(A)(X)$. By Definition 45.686, the assertion $C_F(X)(T)$ says that T commutes with X in the concrete affine classifier. The fixed-point criterion Definition 45.689 identifies that commutation condition with $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X$. Quantifying this equivalence over all elements of $C_F(A)$ gives the displayed exactness statement. □

Per-object template: traditional target restricted normalizer action kernel exactness; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability inverse action, two-sided cancellation, and pointwise kernel test; ledger fixed element, normalizer witness, acted element, inverse parameter, and centralizer-of-acted-element witness; core theorem list Definition 45.695, Definition 45.697, Definition 45.662, and Definition 45.689; exactness predicate action kernel iff centralizes every element of $C_F(A)$; standard bridge ordinary restricted normalizer action kernel; status $\mathsf{ProofTarget}$.

Theorem 45.699 (RatUp FieldUp affine centralizer normalizer action representation)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit*

object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692, and define the restricted action by

$$\mathrm{Act}_T(X) := \mathrm{Conj}_T(X) = (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

The assignment

$$T \longmapsto \mathrm{Act}_T$$

from $N_F(C_F(A))$ to classifier-exact automorphisms of $C_F(A)$ is a concrete homomorphic action representation. Concretely, if $N_F(C_F(A))(T)$, $N_F(C_F(A))(U)$, and $C_F(A)(X), C_F(A)(Y)$, then:

1. $C_F(A)(\mathrm{Act}_T(X))$.

2.

$$\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(Y) \Longleftrightarrow X \equiv_{\mathrm{aff}} Y.$$

3.

$$\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X.$$

4.

$$\mathrm{Act}_T(\mathrm{Act}_U(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{T \circ_{\mathrm{aff}} U}(X).$$

5. $N_F(C_F(A))(\iota_F(T))$, and the inverse action is two-sided:

$$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X \quad \textit{and} \quad \mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X.$$

Proof. The carrier and homomorphism laws are supplied by Definition 45.697. Its preservation clause gives $C_F(A)(\mathrm{Act}_T(X))$, its identity clause gives the action of id_F, and its composition clause gives the displayed law for $T \circ_{\mathrm{aff}} U$.

The subgroup certificate Definition 45.695 supplies the normalizer bookkeeping for the representing source: identity membership, product closure, and the inverse witness $N_F(C_F(A))(\iota_F(T))$.

For the target automorphism attached to T, apply Definition 45.696. Its classifier-exactness clause gives the displayed equivalence on $\equiv_{\mathrm{aff}}$, and its inverse-action clauses give the two-sided cancellation formulas.

Finally, Definition 45.673 identifies these clauses with concrete affine conjugation in the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Thus the map $T \mapsto \mathrm{Act}_T$ lands in the classifier-exact automorphisms of $C_F(A)$ and respects the normalizer multiplication by the stated composition law. □

Per-object template: traditional target normalizer action representation; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier source $N_F(C_F(A))$ and acted carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability carrier preservation, classifier exactness, identity action, composition action, and inverse action; ledger fixed element, normalizer witnesses, acted elements, and inverse witness; core theorem list Definition 45.695, Definition 45.697, Definition 45.696, and Definition 45.673; exactness predicate concrete homomorphic action representation into classifier-exact automorphisms; standard bridge ordinary normalizer representation by automorphisms; status $\mathsf{ProofTarget}$.

Definition 45.700 (RatUp FieldUp affine centralizer automorphism package)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686. For maps $f, g : \mathsf{AffAut}_F \to \mathsf{AffAut}_F$, write $\mathsf{CentAut}_A(f, g)$ for the following classifier-exact self-automorphism package on the concrete carrier $C_F(A)$:*

1. *if* $C_F(A)(X)$*, then* $C_F(A)(f(X))$*;*
2. *if* $C_F(A)(X)$*, then* $C_F(A)(g(X))$*;*
3. *if* $C_F(A)(X)$*, then* $g(f(X)) \equiv_{\mathrm{aff}} X$*;*
4. *if* $C_F(A)(X)$*, then* $f(g(X)) \equiv_{\mathrm{aff}} X$*;*
5. *if* $C_F(A)(X)$ *and* $C_F(A)(Y)$*, then*

$$f(X) \equiv_{\mathrm{aff}} f(Y) \Longleftrightarrow X \equiv_{\mathrm{aff}} Y.$$

Theorem 45.701 (RatUp FieldUp affine centralizer normalizer action homomorphism)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the affine centralizer carrier from Definition 45.686, let* $N_F(C_F(A))$ *be the normalizer carrier from Definition 45.692, and set*

$$\rho_A(T) := \mathrm{Act}_T, \qquad \mathrm{Act}_T(X) := (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

Then the assignment ρ_A *sends every* T *with* $N_F(C_F(A))(T)$ *to*

$$\mathsf{CentAut}_A(\mathrm{Act}_T, \mathrm{Act}_{\iota_F(T)}).$$

It respects the affine classifier on normalizer elements: if $N_F(C_F(A))(T)$*,* $N_F(C_F(A))(U)$*, and* $T \equiv_{\mathrm{aff}} U$*, then for every* X *with* $C_F(A)(X)$*,*

$$\rho_A(T)(X) \equiv_{\mathrm{aff}} \rho_A(U)(X).$$

It sends id_F *to the identity action,*

$$\rho_A(\mathrm{id}_F)(X) \equiv_{\mathrm{aff}} X,$$

and sends affine products to composite actions: if $N_F(C_F(A))(T)$*,* $N_F(C_F(A))(U)$*, and* $C_F(A)(X)$*, then*

$$\rho_A(T \circ_{\mathrm{aff}} U)(X) \equiv_{\mathrm{aff}} \rho_A(T)(\rho_A(U)(X)).$$

The inverse action in the automorphism package is $\mathrm{Act}_{\iota_F(T)}$*.*

Proof. The automorphism package is supplied pointwise by Definition 45.696. For a fixed normalizer witness $N_F(C_F(A))(T)$, that theorem gives inverse normalizer membership for $\iota_F(T)$, preservation of $C_F(A)$ by both Act_T and $\mathrm{Act}_{\iota_F(T)}$, the two cancellation laws $\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X$ and $\mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X$, and the classifier-exact equivalence for the forward action.

The normalizer domain used by this package is the subgroup supplied by Definition 45.695. Its identity, product, inverse, and classifier-transport rows make the source of ρ_A an ordinary classifier group on the concrete normalizer carrier.

Classifier respect follows by applying Definition 45.676 to the two conjugation words determined by T and U. The affine classifier hypothesis $T \equiv_{\mathrm{aff}} U$ transports the outer conjugating parameter and its canonical inverse through the same pointwise classifier, so the resulting actions agree on every $C_F(A)$ element.

The identity row is exactly the identity action clause of Definition 45.697. The source identity belongs to $N_F(C_F(A))$ by Definition 45.695, and the displayed comparison is the classifier identity on the centralizer element.

For products, first use Definition 45.697 to keep $\mathrm{Act}_U(X)$ inside $C_F(A)$. Then apply Definition 45.676; it identifies the inner action of $T \circ_{\mathrm{aff}} U$ with the composite action by U followed by T under $\equiv_{\mathrm{aff}}$. This is the displayed homomorphism law for ρ_A.

The inverse action row is the inverse half of the same automorphism package. For every normalizer element T, the inverse-closure row of Definition 45.695 places $\iota_F(T)$ back in the source carrier, and Definition 45.696 identifies $\mathrm{Act}_{\iota_F(T)}$ as the inverse self-action on $C_F(A)$. □

Per-object template: target normalizer representation; source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit; acted carrier $C_F(A)$; source carrier $N_F(C_F(A))$; action classifier $\equiv_{\mathrm{aff}}$; automorphism package $\mathsf{CentAut}_A$; identity law for id_F; product law for $T \circ_{\mathrm{aff}} U$; inverse action $\mathrm{Act}_{\iota_F(T)}$; standard bridge ordinary homomorphism into centralizer automorphisms.

Theorem 45.702 (RatUp FieldUp restricted normalizer action kernel normality)**.** *For Definition 45.714, the carrier K_A is the normal kernel of the restricted normalizer action on $C_F(A)$. Concretely:*

1. $K_A(\mathrm{id}_F)$.
2. *If $K_A(T)$ and $K_A(U)$, then $K_A(T \circ_{\mathrm{aff}} U)$.*
3. *If $K_A(T)$, then $K_A(\iota_F(T))$.*
4. *If $K_A(T)$ and $N_F(C_F(A))(U)$, then*

$$K_A(U \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(U)).$$

Proof. The identity element belongs to K_A because Definition 45.695 supplies $N_F(C_F(A))(\mathrm{id}_F)$, and the representation identity row in Definition 45.699 gives $\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X$ for every $C_F(A)$ element.

For product closure, the same subgroup certificate gives $N_F(C_F(A))(T \circ_{\mathrm{aff}} U)$. Given $C_F(A)(X)$, the composition row of Definition 45.699 identifies $\mathrm{Act}_{T \circ_{\mathrm{aff}} U}(X)$ with $\mathrm{Act}_T(\mathrm{Act}_U(X))$. The fixed-point hypothesis for $K_A(U)$ reduces the inner action to X, and the fixed-point hypothesis for $K_A(T)$ then reduces the outer action to X.

For inverse closure, the inverse row of Definition 45.695 places $\iota_F(T)$ in $N_F(C_F(A))$. For X in $C_F(A)$, the inverse-action rows of Definition 45.698 make $\mathrm{Act}_{\iota_F(T)}$ the two-sided inverse of Act_T on the centralizer carrier. Since $K_A(T)$ makes Act_T pointwise equal to the identity action, its inverse action is pointwise equal to the identity action as well.

For conjugation, subgroup closure first gives $N_F(C_F(A))(U \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(U))$. Fix X with $C_F(A)(X)$. The inverse-preservation row puts $\mathrm{Act}_{\iota_F(U)}(X)$ back in $C_F(A)$, so the hypothesis $K_A(T)$ fixes that acted element. Applying Act_U and then using the two-sided inverse-action law for U reduces the conjugated action to X. Thus every conjugate of a kernel element by a normalizer element again lies in K_A. □

Per-object template: target restricted normalizer action kernel normality; source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit; acted carrier $C_F(A)$; source carrier $N_F(C_F(A))$; kernel carrier K_A; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, and conjugation inside the normalizer; ledger fixed element, kernel witness, normalizer witness, inverse parameter, and acted centralizer witness; core theorem list Definition 45.695, Definition 45.699, and Definition 45.698; exactness predicate pointwise fixed restricted action on $C_F(A)$; standard bridge ordinary normal kernel of a group action representation; downstream consumer faithful quotient representation of the normalizer on centralizer orbits; status $\mathsf{ProofTarget}$.

Theorem 45.703 (RatUp FieldUp affine centralizer normalizer action certificate)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, and let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692. For each $T : \mathsf{AffAut}_F$ define*

$$\mathrm{Act}_T(X) := (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$, $N_F(C_F(A))(U)$, $C_F(A)(X)$, $C_F(A)(Y)$, $T \equiv_{\mathrm{aff}} U$, and $X \equiv_{\mathrm{aff}} Y$, then the restricted action is classifier well-defined:

$$\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_U(Y).$$

Moreover, for every T with $N_F(C_F(A))(T)$, the inverse affine automorphism is again a normalizer element,

$$N_F(C_F(A))(\iota_F(T)),$$

and for every X with $C_F(A)(X)$ the two restricted actions cancel:

$$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X \quad \textit{and} \quad \mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X.$$

Proof. Fix the concrete affine exit package supplied by Definition 45.662. It fixes the carrier AffAut_F, the operation $\circ_{\mathrm{aff}}$, the inverse operation $\iota_F(-)$, and the classifier $\equiv_{\mathrm{aff}}$ for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit object.

For classifier well-definedness, first apply Definition 45.553 to the left action factor and the acted element. This transports $T \circ_{\mathrm{aff}} X$ to $U \circ_{\mathrm{aff}} Y$ under the affine classifier. The same congruence theorem, applied to the right inverse factors and using the inverse determinacy encoded by Definition 45.675, transports the full conjugation word $(T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T)$ to $(U \circ_{\mathrm{aff}} Y) \circ_{\mathrm{aff}} \iota_F(U)$. This is exactly $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_U(Y)$.

For the inverse normalizer clause, apply the inverse row of Definition 45.695 to $N_F(C_F(A))(T)$. It gives $N_F(C_F(A))(\iota_F(T))$.

Finally, use Definition 45.672 with conjugating element T and acted element X. Its two inverse-action cancellation clauses identify conjugation by $\iota_F(T)$ after conjugation by T, and conjugation by T after conjugation by $\iota_F(T)$, with the original acted element X under $\equiv_{\mathrm{aff}}$. Since Act is the restricted notation for these conjugation words on $C_F(A)$, the displayed restricted cancellation laws follow. □

Per-object template: traditional target group action by a normalizer on a centralizer; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability obligations classifier well-definedness and inverse-action cancellation; ledger normalizer witness, inverse witness, acted element, and conjugation factors; core theorem list Definition 45.695, Definition 45.697, Definition 45.675, and Definition 45.672; exactness predicate restricted action certificate; standard bridge ordinary normalizer action on a subgroup; status $\mathsf{ProofTarget}$.

Theorem 45.704 (RatUp FieldUp affine normalizer action kernel)**.** *For Definition 45.617, fix certified affine automorphisms $A, T : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692, and define the restricted action by*

$$\mathrm{Act}_T(X) := \mathrm{Conj}_T(X) = (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$, then the pointwise kernel condition for the restricted normalizer action is exactly commutation with every element of the centralizer:

$$(\forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow \mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X) \iff (\forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow C_F(X)(T)).$$

Proof. Assume $N_F(C_F(A))(T)$. The restricted action theorem Definition 45.697 supplies the domain condition: whenever $C_F(A)(X)$ holds, $\mathrm{Act}_T(X)$ is again an element of the same concrete centralizer carrier. The present claim identifies which normalizing elements act as the classifier identity on all such X.

For the forward direction, fix $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$ and assume $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X$. Unfolding the action gives the inner conjugation fixed-point equation $\mathrm{Inn}_T(X) \equiv_{\mathrm{aff}} X$. Apply Definition 45.689 to the pair T, X and read the fixed-point condition as the commutation equation $T \circ_{\mathrm{aff}} X \equiv_{\mathrm{aff}} X \circ_{\mathrm{aff}} T$.

By Definition 45.686, this commutation equation is precisely the centralizer membership statement $C_F(X)(T)$. Since X was arbitrary inside $C_F(A)$, the required pointwise centralizer condition follows.

For the reverse direction, fix $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$ and assume $C_F(X)(T)$. Reading Definition 45.686 gives $T \circ_{\mathrm{aff}} X \equiv_{\mathrm{aff}} X \circ_{\mathrm{aff}} T$. Apply Definition 45.689 again, now in the reverse direction for the pair T, X, to obtain $\mathrm{Inn}_T(X) \equiv_{\mathrm{aff}} X$.

Finally replace $\mathrm{Inn}_T(X)$ by the displayed definition of $\mathrm{Act}_T(X)$. This gives $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X$ for every X in $C_F(A)$, so the restricted normalizer action is pointwise trivial exactly on the stated kernel. □

Per-object template: traditional target normalizer action kernel; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability normalizer-supplied restricted action domain and pointwise classifier identity; ledger fixed centralizer element, normalizing element, inverse witness, and commutation certificate; core theorem list Definition 45.697, Definition 45.689, and Definition 45.686; exactness predicate kernel iff commutation with every centralizer element; standard bridge ordinary normalizer action kernel on a centralizer; status ProofTarget.

Theorem 45.705 (RatUp FieldUp affine center normalizes centralizers)**.** *For Definition 45.617, let $A, Z : \mathsf{AffAut}_F$ be certified affine automorphisms in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. If Z is in the concrete affine center, then Z normalizes the concrete affine centralizer of A:*

$$\mathsf{Center}_F(Z) \Rightarrow N_F(C_F(A))(Z).$$

Moreover, for every $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$, the restricted normalizer action from Definition 45.697 is pointwise the classifier identity:

$$\mathrm{Act}_Z(X) \equiv_{\mathrm{aff}} X.$$

Proof. Assume $\mathsf{Center}_F(Z)$. Apply Definition 45.688 and read the first equivalence forward at the fixed automorphism A. This gives the concrete centralizer witness $C_F(A)(Z)$.

Now apply Definition 45.693 to that witness. The conclusion is exactly $N_F(C_F(A))(Z)$, with the normalizer carrier read as the two-sided preservation predicate of Definition 45.692.

For the pointwise identity, fix $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$. The center hypothesis, unfolded at the test element X, says that Z and X commute under $\equiv_{\mathrm{aff}}$.

Apply Definition 45.689 to the pair (Z, X). The commuting row identifies the inner affine conjugate $\mathrm{Act}_Z(X)$ with X under the same classifier. The action theorem Definition 45.697 supplies the restricted-action reading on the carrier $C_F(A)$, and the normalizer membership already proved supplies the required action domain. □

Per-object template: traditional target center action on centralizers; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ and center predicate; classifier $\equiv_{\mathrm{aff}}$; stability center intersection, normalizer inclusion, fixed-point criterion, and restricted action; ledger fixed element, center witness, centralizer witness, and inverse row; core theorem list Definition 45.688, Definition 45.693, Definition 45.689, and Definition 45.697; exactness predicate center element acts as classifier identity on centralizer elements; standard bridge ordinary group center action on centralizers; status $\mathsf{ProofTarget}$.

Theorem 45.706 (RatUp FieldUp affine centralizer normalizer action inverse)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, and let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692. For $T : \mathsf{AffAut}_F$ define*

$$\mathrm{Act}_T(X) := (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

If $N_F(C_F(A))(T)$ and $C_F(A)(X)$, then the restricted action is invertible on the centralizer:

1. $C_F(A)(\mathrm{Act}_T(X))$.
2. $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.
3. $$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} X.$$
4. $$\mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} X.$$

Proof. Fix the canonical affine exit action and inverse package from Definition 45.662. It supplies the carrier AffAut_F, product $\circ_{\mathrm{aff}}$, inverse operation $\iota_F(-)$, classifier $\equiv_{\mathrm{aff}}$, and the inverse rows for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object.

The forward carrier clause is exactly the preservation part of Definition 45.697 applied to T and X.

For the inverse carrier clause, use Definition 45.695 on $N_F(C_F(A))(T)$. Its inverse row gives $N_F(C_F(A))(\iota_F(T))$. Applying Definition 45.697 again, now with conjugating element $\iota_F(T)$ and acted element X, gives $C_F(A)(\mathrm{Act}_{\iota_F(T)}(X))$.

We next compose the two restricted actions. The composition law in Definition 45.697 gives

$$\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{\iota_F(T)\circ_{\mathrm{aff}}T}(X).$$

The left inverse row from the canonical package identifies $\iota_F(T) \circ_{\mathrm{aff}} T$ with id_F. Classifier congruence for Definition 45.675, followed by the identity clause of Definition 45.697, identifies the right side with X. Classifier transitivity yields the displayed left inverse law.

The right inverse law is the same calculation in the other order. The composition law gives

$$\mathrm{Act}_T(\mathrm{Act}_{\iota_F(T)}(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{T\circ_{\mathrm{aff}}\iota_F(T)}(X).$$

The right inverse row from the canonical package identifies $T \circ_{\mathrm{aff}} \iota_F(T)$ with id_F, and the same classifier congruence plus identity-action clause reduces the displayed action to X. □

Per-object template: traditional target invertible normalizer action on centralizer; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability forward preservation, inverse-parameter preservation, and two-sided action inverse laws;

ledger fixed element, acted element, conjugator, inverse witness, and centralizer witnesses; core theorem list Definition 45.697, Definition 45.695, Definition 45.675, and Definition 45.662; exactness predicate restricted action invertibility on centralizer elements; standard bridge ordinary normalizer group action by bijections; status ProofTarget.

Theorem 45.707 (RatUp FieldUp affine centralizer normalizer action classifier congruence)**.** *For Definition 45.617, fix $A : \mathsf{AffAut}_F$. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, and define $\mathrm{Act}_T(X)$ as in Definition 45.697. If $N_F(C_F(A))(T)$, $N_F(C_F(A))(U)$, $C_F(A)(X)$, $C_F(A)(Y)$, $T \equiv_{\mathrm{aff}} U$, and $X \equiv_{\mathrm{aff}} Y$, then*

$$\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_U(Y).$$

Consequently each normalizer element determines a classifier-respecting endomap of $C_F(A)$, and classifier-equal normalizer elements determine classifier-equal endomaps of $C_F(A)$.

Proof. Use Definition 45.697 for the endpoint carrier membership: $N_F(C_F(A))(T)$ sends every $C_F(A)$-element to $C_F(A)$ by Act_T, and likewise for U.

The affine inverse congruence clause of Definition 45.553 transports $T \equiv_{\mathrm{aff}} U$ to $\iota_F(T) \equiv_{\mathrm{aff}} \iota_F(U)$. Apply the operation-congruence clause of the same theorem to $T \circ_{\mathrm{aff}} X$ and $U \circ_{\mathrm{aff}} Y$, using $T \equiv_{\mathrm{aff}} U$ and $X \equiv_{\mathrm{aff}} Y$.

Apply operation congruence once more to compose the resulting comparison with the inverse comparison. This gives

$$(T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T) \equiv_{\mathrm{aff}} (U \circ_{\mathrm{aff}} Y) \circ_{\mathrm{aff}} \iota_F(U).$$

Unfolding the definition of Act in Definition 45.697 identifies the two sides with $\mathrm{Act}_T(X)$ and $\mathrm{Act}_U(Y)$. Setting $U = T$ gives classifier respect for the acted coordinate, while setting $Y = X$ shows that classifier-equal normalizer elements induce classifier-equal restricted endomaps. □

Per-object template: traditional target well-defined normalizer action on a classified centralizer; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit object plus $\mathsf{Group}^{\uparrow}$ action laws; carrier $C_F(A)$ acted on by $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability classifier congruence in acting and acted coordinates; ledger conjugating element, acted element, inverse witness, and normalizer witnesses; core theorem list Definition 45.697 and Definition 45.553; exactness predicate restricted action respects the affine classifier; standard bridge ordinary normalizer action on a subgroup; status ProofTarget.

Theorem 45.708 (RatUp FieldUp canonical exit source-coordinate obstruction)**.** *Let $d_1 := \mathsf{E1}(\mathsf{e})$. For Definition 45.617, the canonical $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate has the concrete source-coordinate witnesses*

$$\mathsf{RatHistoryCarrier}(d_1),$$

$$\mathsf{RatHistoryClassifier}(d_1, d_1),$$

and

$$\mathsf{RatHistoryLedgerPolicy}(d_1, d_1).$$

Each attempted coverage map from these carrier, classifier, or ledger coordinates to the singleton empty-history $\mathsf{Field}^{\uparrow}$ boundary is contradictory. Equivalently, the canonical exit certificate cannot collapse to that singleton boundary in any of its carrier, classifier, or ledger coordinates.

Proof. Invoke Definition 45.651. It identifies the canonical exit object of Definition 45.617 with the concrete $\mathsf{RatUpFieldSource}_F$ source, whose carrier, classifier, and ledger coordinates are the displayed RatUp history rows.

Now apply Definition 45.492 at $d_1 = \mathsf{E1(e)}$. That theorem supplies the three witnesses $\mathsf{RatHistoryCarrier}(d_1)$, $\mathsf{RatHistoryClassifier}(d_1, d_1)$, and $\mathsf{RatHistoryLedgerPolicy}(d_1, d_1)$, and rejects carrier coverage, classifier coverage, and ledger endpoint coverage into the singleton empty-history boundary.

Transporting those three rejections along the canonical-exit identification leaves no coordinate map by which the canonical $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate can be read as the singleton empty-history $\mathsf{Field}^{\uparrow}$ boundary. □

Definition 45.709 (RatUp FieldUp affine centralizer normalizer orbit relation)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, and define $\mathrm{Act}_T(X)$ as in Definition 45.697. On carried centralizer elements $X, Y : \mathsf{AffAut}_F$ with $C_F(A)(X)$ and $C_F(A)(Y)$, define*

$$X \sim_{N_F(A)} Y \iff \exists T : \mathsf{AffAut}_F,\ N_F(C_F(A))(T) \text{ and } \mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y.$$

Theorem 45.710 (RatUp FieldUp affine centralizer normalizer action orbit coverage)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, let* Act *be the restricted normalizer action of Definition 45.697, and let $\sim_{N_F(A)}$ be the orbit relation of Definition 45.709. For $X, T : \mathsf{AffAut}_F$, if $C_F(A)(X)$ and $N_F(C_F(A))(T)$, then*

$$C_F(A)(\mathrm{Act}_T(X)) \quad \text{and} \quad X \sim_{N_F(A)} \mathrm{Act}_T(X).$$

Proof. Apply Definition 45.697 to the carried source endpoint X and the normalizer element T. Its carried-endpoint row gives $C_F(A)(\mathrm{Act}_T(X))$.

For the orbit assertion, choose the same T as the existential witness in Definition 45.709. The witness is valid by the given hypothesis $N_F(C_F(A))(T)$.

It remains only to close the classifier endpoint. Reflexivity of $\equiv_{\mathrm{aff}}$ on $\mathrm{Act}_T(X)$ gives $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X)$, so the displayed witness proves $X \sim_{N_F(A)} \mathrm{Act}_T(X)$. □

Per-object template: traditional target normalizer-action orbit coverage; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability direct action reachability; ledger acting normalizer and acted endpoint; core theorem Definition 45.710 with Definition 45.697; exactness predicate action steps are orbit witnesses; standard bridge ordinary group-action orbit coverage; status $\mathsf{ProofTarget}$.

Theorem 45.711 (RatUp FieldUp affine centralizer normalizer orbit equivalence)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. The relation $\sim_{N_F(A)}$ of Definition 45.709 is reflexive, symmetric, and transitive on the affine centralizer carrier $C_F(A)$. More explicitly, for $X, Y, Z : \mathsf{AffAut}_F$ with $C_F(A)(X)$, $C_F(A)(Y)$, and $C_F(A)(Z)$:*

1. *$X \sim_{N_F(A)} X$.*
2. *If $X \sim_{N_F(A)} Y$, then $Y \sim_{N_F(A)} X$.*
3. *If $X \sim_{N_F(A)} Y$ and $Y \sim_{N_F(A)} Z$, then $X \sim_{N_F(A)} Z$.*

Proof. 1. For reflexivity, use the identity row of Definition 45.695. It gives $N_F(C_F(A))(\mathrm{id}_F)$.

The identity row of Definition 45.697 gives $\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X$. Thus id_F is the required orbit witness for $X \sim_{N_F(A)} X$.

2. For symmetry, suppose $X \sim_{N_F(A)} Y$ and choose a witness $T : \mathsf{AffAut}_F$ with $N_F(C_F(A))(T)$ and $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$. The inverse row of Definition 45.695 gives $N_F(C_F(A))(\iota_F(T))$.

The two-sided inverse laws in Definition 45.706 identify $\mathrm{Act}_{\iota_F(T)}(\mathrm{Act}_T(X))$ with X. Transport the endpoint comparison $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$ through the restricted action using Definition 45.707. Classifier symmetry and transitivity then give $\mathrm{Act}_{\iota_F(T)}(Y) \equiv_{\mathrm{aff}} X$, so $\iota_F(T)$ witnesses $Y \sim_{N_F(A)} X$.

3. For transitivity, suppose $X \sim_{N_F(A)} Y$ via T and $Y \sim_{N_F(A)} Z$ via U. The product row of Definition 45.695 gives $N_F(C_F(A))(U \circ_{\mathrm{aff}} T)$.

The composition row of Definition 45.697 identifies $\mathrm{Act}_U(\mathrm{Act}_T(X))$ with $\mathrm{Act}_{U \circ_{\mathrm{aff}} T}(X)$. Transport the comparison $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$ through the action by U, then compose it with $\mathrm{Act}_U(Y) \equiv_{\mathrm{aff}} Z$. Classifier symmetry and transitivity yield $\mathrm{Act}_{U \circ_{\mathrm{aff}} T}(X) \equiv_{\mathrm{aff}} Z$, so $U \circ_{\mathrm{aff}} T$ witnesses $X \sim_{N_F(A)} Z$.

□

Per-object template: traditional target normalizer orbit relation on a centralizer; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit object plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$ with orbit witnesses in $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability reflexivity, symmetry, and transitivity of $\sim_{N_F(A)}$; ledger fixed element, source endpoint, target endpoint, intermediate endpoint, conjugator, inverse conjugator, and product conjugator; core theorem list Definition 45.695, Definition 45.697, Definition 45.706, and Definition 45.707; exactness predicate normalizer-action reachability is an equivalence relation on carried centralizer elements; standard bridge ordinary group-action orbit equivalence relation; status $\mathsf{ProofTarget}$.

Theorem 45.712 (RatUp FieldUp affine centralizer normalizer orbit classifier congruence)**.** *For Definition 45.617, fix $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, define $\mathrm{Act}_T(X)$ as in Definition 45.697, and let $\sim_{N_F(A)}$ be the reachability relation of Definition 45.709. If $C_F(A)(X)$, $C_F(A)(X')$, $C_F(A)(Y)$, $C_F(A)(Y')$, $X \equiv_{\mathrm{aff}} X'$, $Y \equiv_{\mathrm{aff}} Y'$, and $X \sim_{N_F(A)} Y$, then*

$$X' \sim_{N_F(A)} Y'.$$

Proof. Choose the same normalizer witness from $X \sim_{N_F(A)} Y$. Thus there is $T : \mathsf{AffAut}_F$ with $N_F(C_F(A))(T)$ and $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$.

Apply Definition 45.707 with the acting coordinate T classifier-equal to itself and with the acted coordinate transported from X' to X by classifier symmetry. This gives

$$\mathrm{Act}_T(X') \equiv_{\mathrm{aff}} \mathrm{Act}_T(X).$$

Compose this comparison with the witness endpoint $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$, and then with the supplied endpoint comparison $Y \equiv_{\mathrm{aff}} Y'$. Affine classifier transitivity yields

$$\mathrm{Act}_T(X') \equiv_{\mathrm{aff}} Y'.$$

The same T still lies in $N_F(C_F(A))$, so the displayed endpoint comparison is precisely the data required by Definition 45.709 to witness $X' \sim_{N_F(A)} Y'$. □

Per-object template: traditional target classifier-stable orbit reachability; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine normalizer action; carrier $C_F(A)$; classifier $\equiv_{\text{aff}}$ on endpoints plus $\sim_{N_F(A)}$ as reachability; stability endpoint replacement; ledger endpoint witnesses and normalizer witness; core theorem list Definition 45.709, Definition 45.711, and Definition 45.707; exactness predicate orbit reachability is well-defined on classified centralizer elements; standard bridge ordinary orbit relation under a group action; status $\mathsf{ProofTarget}$.

Theorem 45.713 (RatUp FieldUp affine centralizer normalizer orbit classifier transport)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, and let $\sim_{N_F(A)}$ be the orbit relation of Definition 45.709. If $X, Y, X', Y' : \mathsf{AffAut}_F$ satisfy $C_F(A)(X)$, $C_F(A)(Y)$, $C_F(A)(X')$, $C_F(A)(Y')$, $X \sim_{N_F(A)} Y$, $X \equiv_{\text{aff}} X'$, and $Y \equiv_{\text{aff}} Y'$, then*

$$X' \sim_{N_F(A)} Y'.$$

Proof. Choose the normalizer witness $T : \mathsf{AffAut}_F$ from $X \sim_{N_F(A)} Y$. Thus $N_F(C_F(A))(T)$ and $\mathrm{Act}_T(X) \equiv_{\text{aff}} Y$.

Apply Definition 45.707 with the same acting element in both acting slots. Using classifier reflexivity for T and the endpoint comparison $X \equiv_{\text{aff}} X'$, this gives

$$\mathrm{Act}_T(X) \equiv_{\text{aff}} \mathrm{Act}_T(X').$$

By classifier symmetry, read the last display from $\mathrm{Act}_T(X')$ back to $\mathrm{Act}_T(X)$. Compose it by classifier transitivity with the original orbit endpoint comparison $\mathrm{Act}_T(X) \equiv_{\text{aff}} Y$ and then with $Y \equiv_{\text{aff}} Y'$. The result is

$$\mathrm{Act}_T(X') \equiv_{\text{aff}} Y'.$$

The same witness T therefore satisfies the defining clause of $X' \sim_{N_F(A)} Y'$. □

Per-object template: traditional target orbit-relation descent; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\text{aff}}$; stability endpoint classifier transport; ledger source endpoint, target endpoint, transported endpoints, and normalizer witness; exactness orbit reachability stable under affine classifier.

Definition 45.714 (RatUp FieldUp affine normalizer action kernel carrier)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, let $N_F(C_F(A))$ be the normalizer carrier from Definition 45.692, and write the restricted normalizer action as*

$$\mathrm{Act}_T(X) := (T \circ_{\text{aff}} X) \circ_{\text{aff}} \iota_F(T).$$

The pointwise kernel carrier of this action is the predicate K_A on certified affine automorphisms given by

$$K_A(T) := N_F(C_F(A))(T) \text{ and } \quad \forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow \mathrm{Act}_T(X) \equiv_{\text{aff}} X.$$

Theorem 45.715 (RatUp FieldUp affine normalizer action kernel normal subgroup)**.** *For Definition 45.617, fix $A : \mathsf{AffAut}_F$. The carrier K_A of Definition 45.714 is a normal subgroup package inside the affine normalizer $N_F(C_F(A))$:*

1. $K_A(\mathrm{id}_F)$.

2. *If $K_A(T)$ and $K_A(U)$, then $K_A(T \circ_{\mathrm{aff}} U)$.*
3. *If $K_A(T)$, then $K_A(\iota_F(T))$.*
4. *If $K_A(T)$, $N_F(C_F(A))(U)$, and $T \equiv_{\mathrm{aff}} U$, then $K_A(U)$.*
5. *If $K_A(T)$ and $N_F(C_F(A))(S)$, then $K_A(S \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(S))$.*

Proof. The carrier part of each clause is supplied by Definition 45.695. Its identity, product, inverse, and classifier-transport rows keep the first four displayed witnesses inside $N_F(C_F(A))$; for the conjugation word, use the inverse row on S and the product row twice.

For the identity element, use the identity law in Definition 45.697. For every X with $C_F(A)(X)$, it gives $\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X$, so the pointwise kernel predicate holds.

For products, assume $K_A(T)$ and $K_A(U)$ and fix X with $C_F(A)(X)$. The composition law in Definition 45.697 rewrites $\mathrm{Act}_{T \circ_{\mathrm{aff}} U}(X)$ as $\mathrm{Act}_T(\mathrm{Act}_U(X))$. The U-kernel hypothesis identifies the inner acted element with X, and the T-kernel hypothesis then identifies the outer action with X.

For inverses, start with $K_A(T)$. The inverse theorem Definition 45.706 identifies $\mathrm{Act}_{\iota_F(T)}$ as a two-sided inverse of Act_T on $C_F(A)$. Since Act_T is pointwise the classifier identity there, the inverse action is also pointwise the classifier identity.

For classifier transport, assume $K_A(T)$, $N_F(C_F(A))(U)$, and $T \equiv_{\mathrm{aff}} U$. The action classifier congruence theorem Definition 45.707 transports the pointwise equality $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X$ to $\mathrm{Act}_U(X) \equiv_{\mathrm{aff}} X$ for each X in $C_F(A)$.

Finally assume $K_A(T)$ and $N_F(C_F(A))(S)$. For any X with $C_F(A)(X)$, apply the composition law twice to rewrite

$$\mathrm{Act}_{S \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(S)}(X)$$

as

$$\mathrm{Act}_S(\mathrm{Act}_T(\mathrm{Act}_{\iota_F(S)}(X))).$$

The inverse theorem keeps $\mathrm{Act}_{\iota_F(S)}(X)$ inside $C_F(A)$. Apply the $K_A(T)$ hypothesis to this middle acted element, obtaining $\mathrm{Act}_T(\mathrm{Act}_{\iota_F(S)}(X)) \equiv_{\mathrm{aff}} \mathrm{Act}_{\iota_F(S)}(X)$. Acting by S on both sides and using classifier congruence reduces the displayed conjugation action to $\mathrm{Act}_S(\mathrm{Act}_{\iota_F(S)}(X))$.

The right inverse cancellation clause of Definition 45.706 identifies this last term with X. Hence $S \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(S)$ lies in K_A, giving normality inside the concrete affine normalizer. □

Per-object template: traditional target kernel normal subgroup of a group action; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier K_A; classifier $\equiv_{\mathrm{aff}}$; stability subgroup and conjugation closure; ledger fixed centralizer, acting normalizer, inverse witness, and conjugation word; core theorem list Definition 45.695, Definition 45.697, Definition 45.706, and Definition 45.707; exactness predicate pointwise trivial action; standard bridge ordinary kernel of a group action; status ProofTarget.

Definition 45.716 (RatUp FieldUp affine normalizer same-action classifier)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$. For carried normalizer elements $S, T : \mathsf{AffAut}_F$ with $N_F(C_F(A))(S)$ and $N_F(C_F(A))(T)$, define the restricted-action classifier predicate by*

$$S \equiv_{\rho_A} T :\Longleftrightarrow K_A(\iota_F(S) \circ_{\mathrm{aff}} T).$$

This is a classifier predicate over the concrete normalizer carrier $N_F(C_F(A))$. It records relative membership in the concrete action kernel K_A and does not introduce a quotient construction.

Theorem 45.717 (RatUp FieldUp restricted normalizer action kernel fiber exactness)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$. Let $C_F(A)$ be its concrete affine centralizer, let $N_F(C_F(A))$ be the corresponding normalizer, let $\mathrm{Act}_T(X)$ denote the restricted normalizer action, and let K_A be the kernel carrier of Definition 45.714. If $N_F(C_F(A))(S)$ and $N_F(C_F(A))(T)$, then*

$$K_A(\iota_F(S) \circ_{\mathrm{aff}} T) \Longleftrightarrow \forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow \mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X).$$

Proof. The normalizer subgroup theorem Definition 45.695 places $\iota_F(S)$ in $N_F(C_F(A))$ and then places $\iota_F(S) \circ_{\mathrm{aff}} T$ in $N_F(C_F(A))$. Thus the displayed left side is a genuine instance of the kernel predicate from Definition 45.714.

For the forward direction, assume $K_A(\iota_F(S) \circ_{\mathrm{aff}} T)$ and fix X with $C_F(A)(X)$. The representation law in Definition 45.699 rewrites the action of $\iota_F(S) \circ_{\mathrm{aff}} T$ on X as $\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_T(X))$. The kernel hypothesis identifies this element with X.

Apply Act_S to that equality, using the classifier-exact action row of Definition 45.699. The inverse-action cancellation rows from Definition 45.698 reduce $\mathrm{Act}_S(\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_T(X)))$ to $\mathrm{Act}_T(X)$. Hence $\mathrm{Act}_T(X)$ and $\mathrm{Act}_S(X)$ are classifier-equal, which is the required pointwise equality after symmetry of $\equiv_{\mathrm{aff}}$.

For the reverse direction, assume the pointwise equality of the two restricted actions on every X with $C_F(A)(X)$. The carrier part of $K_A(\iota_F(S) \circ_{\mathrm{aff}} T)$ is the normalizer product constructed in the first paragraph.

It remains to show pointwise fixed action. Fix X with $C_F(A)(X)$. By the representation law, $\mathrm{Act}_{\iota_F(S)\circ_{\mathrm{aff}}T}(X)$ is classifier-equal to $\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_T(X))$. Transport the assumed pointwise equality $\mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X)$ through the classifier-exact action $\mathrm{Act}_{\iota_F(S)}$.

The result identifies $\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_T(X))$ with $\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_S(X))$. The remaining inverse-action cancellation row from Definition 45.698 reduces this last element to X. Therefore $\iota_F(S) \circ_{\mathrm{aff}} T$ fixes every centralizer element and lies in K_A. □

Per-object template: traditional target kernel fiber of a concrete group action representation; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $N_F(C_F(A))$; classifier pointwise $\equiv_{\mathrm{aff}}$ on actions; ledger $S, T, \iota_F(S) \circ_{\mathrm{aff}} T$; core theorem list Definition 45.695, Definition 45.699, Definition 45.698, and Definition 45.715; exactness predicate same action iff kernel difference; standard bridge ordinary group-action kernel fiber; status $\mathsf{ProofTarget}$.

Theorem 45.718 (RatUp FieldUp affine normalizer same-action classifier determinacy)**.** *For Definition 45.716, fix $A, S, T : \mathsf{AffAut}_F$. If $N_F(C_F(A))(S)$ and $N_F(C_F(A))(T)$, then*

$$S \equiv_{\rho_A} T \Longleftrightarrow \forall X : \mathsf{AffAut}_F,\ C_F(A)(X) \Rightarrow \mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X).$$

Proof. By Definition 45.716, $S \equiv_{\rho_A} T$ is the predicate $K_A(\iota_F(S) \circ_{\mathrm{aff}} T)$. Apply Definition 45.717. □

Per-object template: traditional target faithful action modulo kernel; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $N_F(C_F(A))$; classifier $\equiv_{\rho_A}$; stability kernel membership and pointwise restricted-action equality; ledger fixed centralizer element, two normalizer witnesses, inverse parameter, product parameter, and acted element; core theorem list Definition 45.695, Definition 45.699, Definition 45.717, and Definition 45.715; exactness predicate same restricted action iff relative kernel membership; standard bridge ordinary group-action kernel criterion; status $\mathsf{ProofTarget}$.

Theorem 45.719 (RatUp FieldUp affine normalizer same-action classifier stability)**.** *For Definition 45.716, fix a certified affine automorphism $A : \mathsf{AffAut}_F$. On the carried normalizer elements of $N_F(C_F(A))$, the restricted-action classifier $S \equiv_{\rho_A} T$ satisfies the following certificate rows:*

1. *if $N_F(C_F(A))(S)$, then $S \equiv_{\rho_A} S$;*
2. *if $N_F(C_F(A))(S)$, $N_F(C_F(A))(T)$, and $S \equiv_{\rho_A} T$, then $T \equiv_{\rho_A} S$;*
3. *if $N_F(C_F(A))(S)$, $N_F(C_F(A))(T)$, $N_F(C_F(A))(U)$, $S \equiv_{\rho_A} T$, and $T \equiv_{\rho_A} U$, then $S \equiv_{\rho_A} U$;*
4. *if $N_F(C_F(A))(S)$, $N_F(C_F(A))(T)$, $S \equiv_{\rho_A} T$, $S \equiv_{\mathrm{aff}} S'$, and $T \equiv_{\mathrm{aff}} T'$, then the normalizer carrier transports to $N_F(C_F(A))(S')$ and $N_F(C_F(A))(T')$, and $S' \equiv_{\rho_A} T'$.*

Thus the concrete same-action predicate is reflexive, symmetric, transitive, and stable under endpoint replacement by the affine classifier without adding a quotient type.

Proof. Use Definition 45.718 as the translation between $S \equiv_{\rho_A} T$ and pointwise equality of the restricted actions on $C_F(A)$.

For reflexivity, fix $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$. The two restricted actions are the same expression, so affine classifier reflexivity gives $\mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_S(X)$. Determinacy packs this pointwise row into $S \equiv_{\rho_A} S$.

For symmetry, determinacy expands $S \equiv_{\rho_A} T$ into $\mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X)$ for every carried centralizer element X. Affine classifier symmetry reverses each displayed comparison, and determinacy repacks the reversed pointwise row as $T \equiv_{\rho_A} S$.

For transitivity, expand the two hypotheses by determinacy. At each carried centralizer element X, affine classifier transitivity composes $\mathrm{Act}_S(X) \equiv_{\mathrm{aff}} \mathrm{Act}_T(X)$ with $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} \mathrm{Act}_U(X)$. Determinacy then packs the resulting pointwise comparison into $S \equiv_{\rho_A} U$.

For endpoint replacement, first use the normalizer carrier transport rows on $S \equiv_{\mathrm{aff}} S'$ and $T \equiv_{\mathrm{aff}} T'$ to place S' and T' in $N_F(C_F(A))$. Determinacy expands $S \equiv_{\rho_A} T$ to the pointwise restricted-action equality between S and T.

Now fix X with $C_F(A)(X)$. Apply Definition 45.707 to the acting endpoint comparison $S \equiv_{\mathrm{aff}} S'$ and the reflexive endpoint comparison $X \equiv_{\mathrm{aff}} X$; this compares $\mathrm{Act}_S(X)$ with $\mathrm{Act}_{S'}(X)$. Apply the same congruence row to $T \equiv_{\mathrm{aff}} T'$ and the same carried X; this compares $\mathrm{Act}_T(X)$ with $\mathrm{Act}_{T'}(X)$.

Affine classifier symmetry and transitivity compose these two endpoint comparisons with the pointwise equality between $\mathrm{Act}_S(X)$ and $\mathrm{Act}_T(X)$. The result is $\mathrm{Act}_{S'}(X) \equiv_{\mathrm{aff}} \mathrm{Act}_{T'}(X)$ for every carried centralizer element X, and determinacy packs it as $S' \equiv_{\rho_A} T'$. □

Per-object template: traditional target kernel-induced action classifier stability; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $N_F(C_F(A))$; classifier $\equiv_{\rho_A}$; stability rows reflexive, symmetric, transitive, and endpoint transport; ledger acting normalizer witnesses, inverse-product kernel witness, and acted centralizer element; core theorem this block; exactness predicate same restricted action is a stable classifier; standard bridge ordinary action modulo kernel; status ProofTarget.

Theorem 45.720 (RatUp FieldUp operation table raw-continuation separation)**.** *For every $u : \mathsf{BHist}$, the concrete table of Definition 45.587 satisfies its listed $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ table laws. Moreover, if $\mathsf{RatHistoryCarrier}(u)$ holds and an attempted certificate identifies the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication slot on $\mathsf{RatHistoryCarrier}$ with raw Cont while carrying the right-unit row*

$$\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(d, u, r) \to \mathsf{RatHistoryClassifier}(r, d),$$

then the attempted identification implies False. *Consequently the denominator continuation row remains a denominator-side coordinate of the concrete table and is separated from the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *multiplication coordinate.*

Proof. First apply Definition 45.588. It gives the carrier, classifier, ring-base, denominator, contextual, context-pair, affine, inverse, nonzero, and exit noncollapse rows for the concrete table of Definition 45.587.

Now fix $u : \mathsf{BHist}$ and assume $\mathsf{RatHistoryCarrier}(u)$. Also assume that the attempted $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication slot is identified with raw Cont on $\mathsf{RatHistoryCarrier}$ and that the displayed right-unit classifier row is carried by the same certificate.

Apply Definition 45.147 to the same u. Its hypotheses are exactly the carried-unit witness, the raw Cont multiplication identification, and the displayed right-unit row, so its conclusion is False.

The contradiction removes only the proposed identification of the multiplication slot with raw continuation. The positive table laws remain supplied by Definition 45.588, so continuation stays inside the denominator coordinate and not inside the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication coordinate. □

Theorem 45.721 (RatUp FieldUp affine centralizer normalizer orbit action invariance)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the affine centralizer carrier, let* $N_F(C_F(A))$ *be its normalizer carrier, and let* $\sim_{N_F(A)}$ *be the orbit relation of Definition 45.709. For carried centralizer elements* $X, Y : \mathsf{AffAut}_F$ *and a normalizer element* $S : \mathsf{AffAut}_F$*, if* $C_F(A)(X)$*,* $C_F(A)(Y)$*,* $X \sim_{N_F(A)} Y$*, and* $N_F(C_F(A))(S)$*, then*

$$\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y).$$

Proof. Choose an orbit witness $T : \mathsf{AffAut}_F$ for $X \sim_{N_F(A)} Y$, so $N_F(C_F(A))(T)$ and $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$. The inverse row of Definition 45.695 gives $N_F(C_F(A))(\iota_F(S))$, and the product row gives the composite witness

$$N_F(C_F(A))(S \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(S)).$$

It remains to identify the action of this composite witness on $\mathrm{Act}_S(X)$. Apply the composition rows of Definition 45.697 to reassociate the restricted action, and use Definition 45.706 to cancel the adjacent $\iota_F(S)$ and S actions. The result is classifier-equal to $\mathrm{Act}_S(\mathrm{Act}_T(X))$.

Now transport $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} Y$ through the action by S using Definition 45.707. Classifier transitivity identifies the composite action on $\mathrm{Act}_S(X)$ with $\mathrm{Act}_S(Y)$. Hence $S \circ_{\mathrm{aff}} T \circ_{\mathrm{aff}} \iota_F(S)$ is an orbit witness for $\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$. □

Per-object template: traditional target action invariance of normalizer orbits; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability orbit invariance under certified action; ledger acting normalizer, orbit witness, inverse acting normalizer, and composite witness; core theorem Definition 45.721; exactness predicate orbit classes are stable under the certified action; standard bridge ordinary group-action orbit invariance; status $\mathsf{ProofTarget}$.

Theorem 45.722 (RatUp FieldUp affine centralizer normalizer orbit action equivalence)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the affine centralizer carrier, let* $N_F(C_F(A))$ *be its normalizer carrier, and let* $\sim_{N_F(A)}$ *be the orbit relation of Definition 45.709. For carried centralizer elements* $X, Y : \mathsf{AffAut}_F$ *and a normalizer element* $S : \mathsf{AffAut}_F$*, if* $C_F(A)(X)$*,* $C_F(A)(Y)$*, and* $N_F(C_F(A))(S)$*, then*

$$\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y) \quad \Longleftrightarrow \quad X \sim_{N_F(A)} Y.$$

Proof. For the forward direction, assume $\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$. The action theorem supplies the carried centralizer endpoints $\mathrm{Act}_S(X)$ and $\mathrm{Act}_S(Y)$. The inverse row of Definition 45.695 gives $N_F(C_F(A))(\iota_F(S))$.

Apply Definition 45.721 to the acted endpoints and to $\iota_F(S)$. This gives

$$\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_S(X)) \sim_{N_F(A)} \mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_S(Y)).$$

The inverse-action theorem Definition 45.706 cancels both displayed endpoints back to X and Y, and classifier transport for the orbit relation yields $X \sim_{N_F(A)} Y$.

For the reverse direction, assume $X \sim_{N_F(A)} Y$. Then Definition 45.721 applied to S gives $\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$ directly. □

Per-object template: traditional target orbit-action descent and reflection; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier orbit relation $\sim_{N_F(A)}$; stability preserve-and-reflect under normalizer action; ledger normalizer witness, inverse witness, acted endpoints, and orbit witness; core theorem Definition 45.721, Definition 45.695, and Definition 45.706; exactness predicate normalizer action is classifier-exact on orbit reachability; standard bridge ordinary group-action orbit automorphism; status $\mathsf{ProofTarget}$.

Definition 45.723 (RatUp FieldUp affine normalizer point stabilizer)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier and let $N_F(C_F(A))$ be its affine normalizer carrier. For a carried centralizer element $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$, the point-stabilizer carrier $\mathrm{Stab}_{A,X}$ is the predicate on certified affine automorphisms defined by*

$$\mathrm{Stab}_{A,X}(T) := N_F(C_F(A))(T) \text{ and } \quad \mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X,$$

where Act_T is the restricted normalizer action of Definition 45.697.

Theorem 45.724 (RatUp FieldUp affine normalizer point stabilizer subgroup)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ and a carried centralizer element $X : \mathsf{AffAut}_F$ with $C_F(A)(X)$. The carrier $\mathrm{Stab}_{A,X}$ of Definition 45.723 is a subgroup package inside $N_F(C_F(A))$:*

1. $\mathrm{Stab}_{A,X}(\mathrm{id}_F)$.
2. *If* $\mathrm{Stab}_{A,X}(T)$ *and* $\mathrm{Stab}_{A,X}(U)$*, then* $\mathrm{Stab}_{A,X}(T \circ_{\mathrm{aff}} U)$.
3. *If* $\mathrm{Stab}_{A,X}(T)$*, then* $\mathrm{Stab}_{A,X}(\iota_F(T))$.
4. *If* $\mathrm{Stab}_{A,X}(T)$*,* $N_F(C_F(A))(U)$*, and* $T \equiv_{\mathrm{aff}} U$*, then* $\mathrm{Stab}_{A,X}(U)$.

Proof. The normalizer component of each row is supplied by Definition 45.695. Its identity, product, inverse, and classifier-transport clauses keep the displayed affine words inside $N_F(C_F(A))$.

For the identity row, use the identity law in Definition 45.697. Applied to the fixed carried point X, it gives $\mathrm{Act}_{\mathrm{id}_F}(X) \equiv_{\mathrm{aff}} X$.

For product closure, assume $\mathrm{Stab}_{A,X}(T)$ and $\mathrm{Stab}_{A,X}(U)$. The composition law in Definition 45.697 identifies the product action on X with $\mathrm{Act}_T(\mathrm{Act}_U(X))$. The U-stabilizer row fixes X, and the acting-coordinate part of Definition 45.707 transports the remaining action to $\mathrm{Act}_T(X)$. The T-stabilizer row then identifies the result with X.

For inverse closure, assume $\mathrm{Stab}_{A,X}(T)$. The inverse theorem Definition 45.706 supplies cancellation between the actions of T and $\iota_F(T)$ on the centralizer carrier. Since T fixes X up to $\equiv_{\mathrm{aff}}$, endpoint transport by Definition 45.707 turns the inverse cancellation row into $\mathrm{Act}_{\iota_F(T)}(X) \equiv_{\mathrm{aff}} X$.

For classifier transport, assume $\mathrm{Stab}_{A,X}(T)$, $N_F(C_F(A))(U)$, and $T \equiv_{\mathrm{aff}} U$. The endpoint and acting coordinates in Definition 45.707 transport $\mathrm{Act}_T(X) \equiv_{\mathrm{aff}} X$ to $\mathrm{Act}_U(X) \equiv_{\mathrm{aff}} X$. This gives the transported stabilizer row inside the normalizer. □

Per-object template: traditional target point stabilizer of a group action; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine normalizer action plus $\mathsf{Group}^{\uparrow}$; carrier $\mathrm{Stab}_{A,X}$; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, and transport; ledger fixed centralizer point, acting normalizer, inverse witness, and action word; core theorem Definition 45.724; exactness predicate fixed-point action on one carried centralizer element; standard bridge ordinary point stabilizer subgroup; status ProofTarget.

Theorem 45.725 (RatUp FieldUp concrete certificate boundary)**.** *For the named source tuple* $\mathsf{RatUpFieldSource}_F$ *of Definition 45.488, the following four assertions form one concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *interface fact.*

1. *The tuple satisfies Definition 45.337 by the assembled boundary theorem Definition 45.489.*
2. *Its lower algebra coordinates are compatible with the concrete* $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, *and* $\mathsf{CommRing}^{\uparrow}$ *reducts by Definition 45.659.*
3. *Its denominator operation coordinate has the checked functional laws of Definition 45.324 and the checked relational laws of Definition 45.325; the endpoint bridge Definition 45.326 identifies a displayed* $\mathsf{Cont}(h,k,r)$ *endpoint with* $\mu_D(h,k)$ *through* R_D *under* $C_D(h)$ *and* $C_D(k)$.
4. *Any certificate identifying the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *multiplication slot with raw* Cont *on* RatHistoryCarrier *while carrying the displayed right-unit row contradicts Definition 45.720.*

Thus the assembled object records the certificate obligations, the lower algebra reduct interface, the denominator append and relational law packages, their endpoint identification through R_D*, and the raw-continuation separation as one FieldUp(RatUp) boundary.*

Proof. First apply Definition 45.489. It unpacks $\mathsf{RatUpFieldSource}_F$ over BHist and supplies the carrier, classifier, commutative-ring base, inverse, nonzero, denominator, contextual-action, context-pair, and affine rows required by Definition 45.337.

Next apply Definition 45.659. This places the same concrete source over the already certified $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, and $\mathsf{CommRing}^{\uparrow}$ reducts, so the FieldUp certificate is read against the lower algebra hierarchy without schematic operation parameters.

For the denominator coordinate, first invoke Definition 45.324 and Definition 45.325. These give the functional append rows and the relational Cont rows as separate checked denominator law packages.

Then invoke Definition 45.326. For carried denominator histories h and k, it identifies a displayed continuation endpoint r with the functional append endpoint $\mu_D(h,k)$ exactly through $R_D(r, \mu_D(h,k))$.

Finally use Definition 45.720. Its negative clause applies exactly to an attempted multiplication-slot identification with raw Cont on RatHistoryCarrier carrying the right-unit row, while its positive clause leaves the concrete operation table laws intact. Hence the denominator continuation coordinate remains separated from the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication coordinate. □

Per-object template: traditional target concrete constructive field certificate for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist with RatHistoryCarrier, RatHistoryClassifier, denominator append and Cont law packages connected at endpoints through R_D, and lower algebra reducts; classifier RatHistoryClassifier plus the affine classifier where used; ledger ring rows, denominator histories, continuation witnesses,

inverse witnesses, contextual pairs, and affine witnesses; core theorem list Definition 45.489, Definition 45.659, Definition 45.324, Definition 45.325, Definition 45.326, and Definition 45.720; exactness predicate assembled obligations with raw-continuation separation; standard bridge ordinary constructive rational field interface; status ProofTarget.

Theorem 45.726 (RatUp FieldUp normalizer kernel collapses orbit action)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, let K_A be the kernel carrier of Definition 45.714, let* Act *be the restricted normalizer action, and let $\sim_{N_F(A)}$ be the orbit relation of Definition 45.709. For carried centralizer elements $X, Y : \mathsf{AffAut}_F$, a normalizer element $S : \mathsf{AffAut}_F$, and a kernel element $T : \mathsf{AffAut}_F$, if $C_F(A)(X)$, $C_F(A)(Y)$, $X \sim_{N_F(A)} Y$, $N_F(C_F(A))(S)$, and $K_A(T)$, then $C_F(A)(\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X))$, $C_F(A)(\mathrm{Act}_S(Y))$, and*

$$\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X) \sim_{N_F(A)} \mathrm{Act}_S(Y).$$

Proof. The kernel hypothesis contains the carrier witness $N_F(C_F(A))(T)$. Combine it with $N_F(C_F(A))(S)$ by the product row of Definition 45.695; this certifies $S \circ_{\mathrm{aff}} T$ as a normalizer element. The action theorem Definition 45.697 then gives the centralizer carrier of $\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X)$, and the same theorem applied to S gives the carrier of $\mathrm{Act}_S(Y)$.

For the classifier comparison, use the composition row of Definition 45.697 to identify $\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X)$ with $\mathrm{Act}_S(\mathrm{Act}_T(X))$. Since $K_A(T)$ is pointwise trivial on carried centralizer elements, it identifies $\mathrm{Act}_T(X)$ with X. The action classifier congruence theorem transports this equality through the action by S, so $\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X)$ is classifier-equal to $\mathrm{Act}_S(X)$.

Apply Definition 45.721 to the original orbit witness $X \sim_{N_F(A)} Y$ and the normalizer element S. It gives $\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$. Transport the left endpoint across the classifier equality above using the orbit classifier congruence and transport rows, yielding the displayed orbit relation for $\mathrm{Act}_{S\circ_{\mathrm{aff}}T}(X)$ and $\mathrm{Act}_S(Y)$. □

Per-object template: traditional target kernel collapse for the induced normalizer action on centralizer orbit classes; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\mathrm{aff}}$; stability kernel factors act trivially after passing to normalizer orbits; ledger acting normalizer, kernel factor, product normalizer, centralizer endpoints, and transported orbit witness; core theorem Definition 45.726; exactness predicate normalizer action on centralizer orbit classes; standard bridge ordinary quotient action by the kernel of a group action; status ProofTarget.

Definition 45.727 (RatUp FieldUp right-kernel normalizer representatives)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let K_A be the kernel carrier of Definition 45.714. For affine normalizer representatives $S, S' : \mathsf{AffAut}_F$, define right-kernel representative equivalence by*

$$S \equiv^{\mathrm{right}}_{K_A} S' \iff \exists T : \mathsf{AffAut}_F,\ K_A(T) \text{ and } S' \equiv_{\mathrm{aff}} S \circ_{\mathrm{aff}} T.$$

Theorem 45.728 (RatUp FieldUp normalizer kernel representatives act equally on orbit classes)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, let K_A be the kernel carrier of Definition 45.714, let* Act *be the restricted normalizer action, and let $\sim_{N_F(A)}$ be the orbit relation of Definition 45.709. For carried centralizer elements $X, Y : \mathsf{AffAut}_F$ and normalizer representatives $S, S' : \mathsf{AffAut}_F$, if $C_F(A)(X)$, $C_F(A)(Y)$, $X \sim_{N_F(A)} Y$, $N_F(C_F(A))(S)$, $N_F(C_F(A))(S')$, and $S \equiv^{\mathrm{right}}_{K_A} S'$, then $C_F(A)(\mathrm{Act}_S(Y))$, $C_F(A)(\mathrm{Act}_{S'}(X))$, and*

$$\mathrm{Act}_{S'}(X) \sim_{N_F(A)} \mathrm{Act}_S(Y).$$

Proof. Unpack $S \equiv_{K_A}^{\text{right}} S'$ and choose $T : \mathsf{AffAut}_F$ with $K_A(T)$ and $S' \equiv_{\text{aff}} S \circ_{\text{aff}} T$. The normalizer action theorem Definition 45.697 applied to S and Y gives $C_F(A)(\mathrm{Act}_S(Y))$, and the same theorem applied to S' and X gives $C_F(A)(\mathrm{Act}_{S'}(X))$.

Apply Definition 45.726 to S, T, X, Y. It uses the kernel factor T and the orbit witness $X \sim_{N_F(A)} Y$ to produce

$$\mathrm{Act}_{S \circ_{\text{aff}} T}(X) \sim_{N_F(A)} \mathrm{Act}_S(Y).$$

It remains to replace the product representative by S'. The action classifier congruence theorem transports $S' \equiv_{\text{aff}} S \circ_{\text{aff}} T$ along the fixed endpoint X, yielding a classifier comparison between $\mathrm{Act}_{S'}(X)$ and $\mathrm{Act}_{S \circ_{\text{aff}} T}(X)$.

Use orbit classifier transport Definition 45.713 with this left-endpoint comparison and classifier reflexivity on $\mathrm{Act}_S(Y)$. The transported witness is exactly $\mathrm{Act}_{S'}(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$. □

Per-object template: traditional target representative independence for a group action modulo its pointwise kernel; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\text{aff}}$; stability right-kernel representative independence on orbit classes; ledger representative, kernel factor, product word, endpoint carriers, and transported orbit witness; core theorem Definition 45.726, Definition 45.713, Definition 45.707, Definition 45.697, and Definition 45.715; exactness predicate induced action on centralizer orbit classes; standard bridge ordinary action modulo a kernel relation; status $\mathsf{ProofTarget}$.

Theorem 45.729 (RatUp FieldUp affine centralizer normalizer orbit action reflection)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier, let $N_F(C_F(A))$ be its normalizer carrier, and let $\sim_{N_F(A)}$ be the orbit relation of Definition 45.709. For carried centralizer elements $X, Y : \mathsf{AffAut}_F$ and a normalizer element $S : \mathsf{AffAut}_F$, if $C_F(A)(X)$, $C_F(A)(Y)$, and $N_F(C_F(A))(S)$, then*

$$X \sim_{N_F(A)} Y \iff \mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y).$$

Proof. The forward direction is exactly Definition 45.721 with the acting element S.

For the reverse direction, assume $\mathrm{Act}_S(X) \sim_{N_F(A)} \mathrm{Act}_S(Y)$. The inverse row of Definition 45.695 gives $N_F(C_F(A))(\iota_F(S))$.

The carrier clauses in Definition 45.706 place $\mathrm{Act}_S(X)$ and $\mathrm{Act}_S(Y)$ back in $C_F(A)$, so the orbit invariance theorem applies to these acted endpoints.

Apply Definition 45.721 to the assumed orbit relation with the acting element $\iota_F(S)$. This yields

$$\mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_S(X)) \sim_{N_F(A)} \mathrm{Act}_{\iota_F(S)}(\mathrm{Act}_S(Y)).$$

The cancellation clauses of Definition 45.706 identify the left endpoint with X and the right endpoint with Y in the affine classifier. The endpoint transport theorem Definition 45.713 then replaces both acted endpoints by X and Y, giving $X \sim_{N_F(A)} Y$. □

Per-object template: traditional target orbit reflection by a group-action automorphism; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ affine exit plus $\mathsf{Group}^{\uparrow}$; carrier $C_F(A)$; classifier $\equiv_{\text{aff}}$ and orbit reachability; stability action reflection; ledger acting normalizer, inverse normalizer, orbit witness, and cancellation witnesses; core theorem list Definition 45.721, Definition 45.706, Definition 45.695, and Definition 45.713 with Definition 45.712; exactness predicate normalizer actions preserve and reflect orbit reachability; standard bridge ordinary group-action orbit reflection; status $\mathsf{ProofTarget}$.

Theorem 45.730 (RatUp FieldUp affine centralizer normalizer SubgroupUp tower certificate)**.** *For Definition 45.617, fix a certified affine automorphism $A : \mathsf{AffAut}_F$ in the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Let $C_F(A)$ be the affine centralizer carrier from Definition 45.686, let $N_F(C_F(A))$ be the two-sided affine normalizer carrier from Definition 45.692, and define*

$$\mathrm{Act}_T(X) = (T \circ_{\mathrm{aff}} X) \circ_{\mathrm{aff}} \iota_F(T).$$

Then the inclusion

$$C_F(A) \leq N_F(C_F(A))$$

forms the concrete quotient-ready $\mathsf{Subgroup}^{\uparrow}$ tower whose lower subgroup rows are the centralizer identity, product, inverse, and classifier-transport rows, whose upper subgroup rows are the normalizer identity, product, inverse, and classifier-transport rows, whose carrier inclusion is

$$C_F(A)(T) \Rightarrow N_F(C_F(A))(T),$$

and whose action rows state that every T in $N_F(C_F(A))$ preserves $C_F(A)$ under Act_T, the identity and product action laws hold on the centralizer carrier, the classifier well-definedness condition for Act *holds on normalizer and centralizer representatives, and the inverse action by $\iota_F(T)$ cancels Act_T on both sides.*

Proof. The lower carrier is the concrete affine centralizer of A. Apply Definition 45.687; it supplies the identity, product, inverse, and classifier-transport rows for $C_F(A)$.

The upper carrier is the two-sided affine normalizer of this centralizer. Apply Definition 45.695; it supplies the corresponding subgroup rows for $N_F(C_F(A))$. The carrier inclusion is exactly Definition 45.693.

It remains to expose the quotient-ready action data. Apply Definition 45.697 for normalizer preservation of the centralizer and for the identity and composition action laws. Apply Definition 45.703 for classifier well-definedness of Act and for inverse-action cancellation on the restricted centralizer carrier.

These rows are precisely the concrete $\mathsf{Subgroup}^{\uparrow}$ input demanded by the quotient-group construction: a lower subgroup, an upper subgroup, an inclusion of carriers, and an action certificate by upper representatives on lower representatives. □

Per-object template: traditional target normal-subgroup input for quotient groups; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ together with $\mathsf{Subgroup}^{\uparrow}$; carrier pair $C_F(A), N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; ledger fixed affine element, conjugators, inverse rows, and action witnesses; exactness as the normalizer action on the centralizer; status $\mathsf{ProofTarget}$.

Theorem 45.731 (Field singleton-empty operation context nonzero obstruction)**.** *For histories L, R, h, k, out : BHist with $\mathsf{fieldSingletonEmptyCarrier}(L)$, $\mathsf{fieldSingletonEmptyCarrier}(R)$, and $\mathsf{fieldSingletonEmptyNonZero}(out)$, singleton-empty operation contexts reject proof-indexed nonzero outputs. The multiplication endpoint satisfies*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyMul}(h, k)), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

The unit endpoint satisfies

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyOne}), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

For every proof witness p : $\mathsf{fieldSingletonEmptyNonZero}(h)$, the inverse endpoint satisfies

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyInv}(h, p)), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

Proof. Apply Definition 45.591 to the multiplication, unit, or proof-indexed inverse classifier in the displayed context. In each case the contextual classifier into $\mathsf{append}(R, out)$ is converted into $\mathsf{fieldSingletonEmptyCarrier}(out)$.

The carrier assertion is the empty endpoint input for Definition 45.34. Combining it with the assumed witness $\mathsf{fieldSingletonEmptyNonZero}(out)$ gives False for each of the three operation contexts. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmptyClassifier operation context nonzero absurd`
Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmpty operation context nonzero obstruction`

Per-object template: traditional target terminal $\mathsf{Field}^{\uparrow}$ operation-domain obstruction; BEDC source BHist, append, lower-case singleton-empty $\mathsf{Field}^{\uparrow}$ endpoints; carrier $\mathsf{fieldSingletonEmptyCarrier}$; classifier $\mathsf{fieldSingletonEmptyClassifier}$; stability obligations operation-context normalization and nonzero endpoint obstruction; ledger hidden singleton contexts and proof-indexed inverse witness; core theorem this block; exactness predicate classifier-to-nonzero contradiction; status $\mathsf{ProofTarget}$.

Theorem 45.732 (Field singleton-empty operation context carried nonzero obstruction)**.** *For histories* $L, R, U, V, h, k, out : \mathsf{BHist}$ *with* $\mathsf{fieldSingletonEmptyCarrier}(L)$, $\mathsf{fieldSingletonEmptyCarrier}(R)$, $\mathsf{fieldSingletonEmptyCarrier}(U)$, $\mathsf{fieldSingletonEmptyCarrier}(V)$, *and* $\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(U, \mathsf{append}(ou$ *carried nonzero evidence rejects the singleton-empty operation contexts. The multiplication endpoint satisfies*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyMul}(h, k)), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

The unit endpoint satisfies

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyOne}), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

For every proof witness $p : \mathsf{fieldSingletonEmptyNonZero}(h)$, *the proof-indexed inverse endpoint satisfies*

$$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, \mathsf{fieldSingletonEmptyInv}(h, p)), \mathsf{append}(R, out)) \Longrightarrow \mathsf{False}.$$

Proof. First cancel the carried output contexts. Apply Definition 45.78 to U, V, out with the carrier witnesses for U and V. The assumed predicate on $\mathsf{append}(U, \mathsf{append}(out, V))$ yields the direct witness $\mathsf{fieldSingletonEmptyNonZero}(out)$.

Now take any one of the three displayed operation classifiers. Apply Definition 45.591 with the carrier witnesses for L and R. In the multiplication, unit, and proof-indexed inverse cases, this produces $\mathsf{fieldSingletonEmptyCarrier}(out)$.

Finally, use Definition 45.34. The output carrier is the empty endpoint input, and the cancelled nonzero witness on out supplies the contradictory nonzero side. □

Lean4 ✓ `BEDC.Derived.FieldUp.fieldSingletonEmpty operation context carried nonzero obstruction`

Per-object template: traditional target contextual nonzero obstruction for the concrete $\mathsf{Field}^{\uparrow}$ operation slots; BEDC source BHist, append, $\mathsf{fieldSingletonEmptyCarrier}$, $\mathsf{fieldSingletonEmptyClassifier}$, and $\mathsf{fieldSingletonEmptyNonZero}$; carrier and classifier as named; stability obligations context cancellation and operation normalization; ledger carried output contexts U, V; core theorem this block; exactness predicate contextual nonzero rejection; status $\mathsf{ProofTarget}$.

45.24 Apart-zero continuation exactness

Theorem 45.733 (Field apart-zero nested continuation empty-context exactness)**.** *For histories* $l, h, r, u, v : \mathsf{BHist}$ *with* $\mathsf{Cont}(l, h, u)$ *and* $\mathsf{Cont}(u, r, v)$*, if* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$*, then*

$$\mathsf{ApartZero}_F(v) \iff \mathsf{ApartZero}_F(h).$$

Proof. The two continuation witnesses identify u with $\mathsf{append}(l, h)$ and v with $\mathsf{append}(u, r)$. Continuation associativity and result transport compare v with the nested append endpoint $\mathsf{append}(l, \mathsf{append}(h, r))$.

Transport $\mathsf{ApartZero}_F$ across this hsame comparison. The result is exactly the bilateral empty-context statement of Definition 45.267, applied with the witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero nested continuation empty context iff`

Theorem 45.734 (Field apart-zero nested continuation factor split)**.** *For histories* $l, h, r, u, v, q, w : \mathsf{BHist}$ *with* $\mathsf{Cont}(l, h, u)$*,* $\mathsf{Cont}(u, r, v)$*, and* $\mathsf{Cont}(v, q, w)$*, if* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$*, then*

$$\mathsf{ApartZero}_F(w) \iff \mathsf{ApartZero}_F(h) \lor \mathsf{ApartZero}_F(q).$$

Proof. Apply Definition 45.256 to the tail continuation witness $\mathsf{Cont}(v, q, w)$. This gives $\mathsf{ApartZero}_F(w)$ exactly when $\mathsf{ApartZero}_F(v) \lor \mathsf{ApartZero}_F(q)$ holds.

Apply Definition 45.733 to the first two continuation witnesses and the exterior empty-context witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. Substituting this equivalence for the first disjunct yields the displayed factor split. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero nested continuation factor split iff`

Per-object template: traditional target continuation-tail apartness factor split under empty exterior context; BEDC source BHist, Cont, and $\mathsf{ApartZero}_F$; carrier apart-zero histories; classifier hsame transport; stability obligations nested empty-context exactness plus continuation endpoint split; ledger the two empty-context witnesses and tail q; exactness predicate the displayed iff; standard bridge ordinary product/context apartness factorization; status $\mathsf{ProofTarget}$.

Theorem 45.735 (Field apart-zero two-tail nested continuation factor split)**.** *For histories* $l, h, r, u, v, q, w, s, z : \mathsf{BHist}$ *with* $\mathsf{Cont}(l, h, u)$*,* $\mathsf{Cont}(u, r, v)$*,* $\mathsf{Cont}(v, q, w)$*, and* $\mathsf{Cont}(w, s, z)$*, if* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$*, then*

$$\mathsf{ApartZero}_F(z) \iff \mathsf{ApartZero}_F(h) \lor \mathsf{ApartZero}_F(q) \lor \mathsf{ApartZero}_F(s).$$

Proof. Apply Definition 45.256 to the final tail continuation witness $\mathsf{Cont}(w, s, z)$. This gives $\mathsf{ApartZero}_F(z)$ exactly when $\mathsf{ApartZero}_F(w) \lor \mathsf{ApartZero}_F(s)$ holds.

Apply Definition 45.734 to the first three continuation witnesses and the exterior witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. This identifies $\mathsf{ApartZero}_F(w)$ with $\mathsf{ApartZero}_F(h) \lor \mathsf{ApartZero}_F(q)$.

For the forward direction, case-analyze the endpoint disjunction. The $\mathsf{ApartZero}_F(s)$ case supplies the right tail disjunct. In the $\mathsf{ApartZero}_F(w)$ case, case-analyze the nested split and place the resulting witness in the h or q disjunct.

For the reverse direction, case-analyze the displayed three-way disjunction. The h and q cases first rebuild $\mathsf{ApartZero}_F(w)$ by the reverse direction of the nested split, and then rebuild $\mathsf{ApartZero}_F(z)$ by the reverse direction of the endpoint split. The s case uses the reverse direction of the endpoint split directly. □

Per-object template: object $\mathsf{Field}^{\uparrow}$ apart-zero multi-tail factorization; source BHist plus Cont; carrier apart-zero histories; classifier hsame transport; transport predicate $\mathsf{ApartZero}_F$; ledger the empty exterior witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$; tail data the two tail factors q and s; core theorem this block plus the one-tail split and endpoint split; exactness predicate the displayed iff; status $\mathsf{ProofTarget}$.

Theorem 45.736 (Field apart-zero nested continuation singleton-result factor exclusion)**.** *For histories* $l, h, r, u, v, q, w, s, z, endpoint$: BHist *with* $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $\mathsf{Cont}(v, q, w)$, *and* $\mathsf{Cont}(w, s, z)$, *if* $l\mathsf{hsamee}$, $r\mathsf{hsamee}$, *and* $\mathsf{FieldSingletonClassifier}(z, endpoint)$, *then*

$$(\mathsf{ApartZero}_F(h) \to \bot) \wedge (\mathsf{ApartZero}_F(q) \to \bot) \wedge (\mathsf{ApartZero}_F(s) \to \bot).$$

Proof. Apply Definition 45.260 to the final continuation witness $\mathsf{Cont}(w, s, z)$ together with the terminal singleton-result certificate. This gives exclusions for $\mathsf{ApartZero}_F(w)$ and $\mathsf{ApartZero}_F(s)$.

To exclude $\mathsf{ApartZero}_F(h)$, assume such a witness. The reverse direction of Definition 45.734, applied to the first three continuation witnesses and the exterior empty-context witnesses, turns the left factor witness into $\mathsf{ApartZero}_F(w)$. This contradicts the exclusion for w.

The argument for $\mathsf{ApartZero}_F(q)$ is the same, using the right nested factor witness in the reverse direction of the same split theorem. Pair the two nested exclusions with the already obtained exclusion for s to obtain the displayed triple exclusion. □

Per-object template: traditional target zero-result factor exclusion for nested $\mathsf{Field}^{\uparrow}$ continuations; BEDC source BHist, Cont, $\mathsf{FieldSingletonClassifier}$, and $\mathsf{ApartZero}_F$; carrier apart-zero histories; classifier hsame transport; transport predicate terminal singleton-result rejection; ledger the empty exterior witnesses plus the terminal singleton-result certificate; tail data the nested factors h, q, and s; core theorem list the one-step singleton-result factor absurd theorem and nested factor split; exactness predicate the displayed triple exclusion; status $\mathsf{ProofTarget}$.

Theorem 45.737 (Field apart-zero three-tail nested continuation factor split)**.** *For histories* $l, h, r, u, v, q, w, s, z, t, y$: BHist *with* $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $\mathsf{Cont}(v, q, w)$, $\mathsf{Cont}(w, s, z)$, *and* $\mathsf{Cont}(z, t, y)$, *if* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$, *then*

$$\mathsf{ApartZero}_F(y) \iff \mathsf{ApartZero}_F(h) \vee \mathsf{ApartZero}_F(q) \vee \mathsf{ApartZero}_F(s) \vee \mathsf{ApartZero}_F(t).$$

Proof. Apply Definition 45.256 to the endpoint continuation witness $\mathsf{Cont}(z, t, y)$. This splits $\mathsf{ApartZero}_F(y)$ into $\mathsf{ApartZero}_F(z) \vee \mathsf{ApartZero}_F(t)$.

Apply Definition 45.735 to the first four continuation witnesses and the exterior empty-context witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. This rewrites $\mathsf{ApartZero}_F(z)$ as the three-way disjunction on h, q, s.

For the forward direction, case-analyze the endpoint split. The $\mathsf{ApartZero}_F(t)$ case supplies the final tail disjunct. In the $\mathsf{ApartZero}_F(z)$ case, case-analyze the two-tail nested split and place the resulting evidence in the h, q, or s disjunct.

For the reverse direction, case-analyze the displayed four-way disjunction. The h, q, and s cases first rebuild $\mathsf{ApartZero}_F(z)$ by the reverse direction of the two-tail nested split, and then rebuild $\mathsf{ApartZero}_F(y)$ by the reverse direction of the endpoint split. The t case uses the reverse direction of the endpoint split directly. □

Per-object template: traditional target finite product/context apartness factorization; BEDC source BHist plus Cont; carrier apart-zero histories; classifier hsame transport; stability obligations endpoint split and two-tail nested split; ledger exterior empty witnesses plus tails q, s, t; core theorem this block; exactness predicate the displayed iff; standard bridge finite product apartness; status $\mathsf{ProofTarget}$.

Theorem 45.738 (Field apart-zero three-tail contextual singleton result exclusion)**.** *For histories* $L, R, l, h, r, u, v, q, w, s, z, t, y, endpoint$: BHist *with* $\mathsf{FieldSingletonCarrier}(L)$, $\mathsf{FieldSingletonCarrier}(R)$, $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $\mathsf{Cont}(v, q, w)$, $\mathsf{Cont}(w, s, z)$, $\mathsf{Cont}(z, t, y)$, $l\mathsf{hsamee}$, $r\mathsf{hsamee}$, *and* $\mathsf{FieldSingletonClassifier}(a$ *one has*

$$(\mathsf{ApartZero}_F(h) \to \bot) \wedge (\mathsf{ApartZero}_F(q) \to \bot) \wedge (\mathsf{ApartZero}_F(s) \to \bot) \wedge (\mathsf{ApartZero}_F(t) \to \bot).$$

Proof. Use Definition 45.57 with the singleton-carrier witnesses for L and R. The forward direction of the cancellation equivalence turns the contextual classifier into $\mathsf{FieldSingletonClassifier}(y, endpoint)$.

Apply Definition 45.6 to that singleton endpoint classifier. The resulting endpoint comparison gives $y\mathsf{hsamee}$.

Now assume one of the four factor witnesses $\mathsf{ApartZero}_F(h)$, $\mathsf{ApartZero}_F(q)$, $\mathsf{ApartZero}_F(s)$, or $\mathsf{ApartZero}_F(t)$. The reverse direction of Definition 45.737, applied to the corresponding disjunct and the five continuation witnesses, rebuilds $\mathsf{ApartZero}_F(y)$.

Applying this apart-zero witness to $y\mathsf{hsamee}$ gives $\bot$ in each case. Pairing the four implications gives the displayed exclusion. □

Per-object template: traditional target contextual singleton-result factor exclusion for three-tail $\mathsf{Field}^{\uparrow}$ continuations; BEDC source BHist, Cont, $\mathsf{FieldSingletonClassifier}$, and $\mathsf{ApartZero}_F$; carrier apart-zero histories; classifier hsame transport; transport predicate singleton-context cancellation; ledger the hidden singleton contexts, empty exterior witnesses, and terminal singleton certificate; tail data the exposed factors h, q, s, t; core theorem list the three-tail nested factor split plus singleton endpoint classifier; exactness predicate the displayed fourfold exclusion; status $\mathsf{ProofTarget}$.

Theorem 45.739 (Field apart-zero two-tail contextual singleton result exclusion)**.** *For histories* $L, R, l, h, r, u, v, q, w, s, z, endpoint$: BHist *with* $\mathsf{FieldSingletonCarrier}(L)$, $\mathsf{FieldSingletonCarrier}(R)$, $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $\mathsf{Cont}(v, q, w)$, $\mathsf{Cont}(w, s, z)$, $l\mathsf{hsamee}$, $r\mathsf{hsamee}$, *and* $\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, z), a$ *one has*

$$(\mathsf{ApartZero}_F(h) \to \bot) \wedge (\mathsf{ApartZero}_F(q) \to \bot) \wedge (\mathsf{ApartZero}_F(s) \to \bot).$$

Proof. Use Definition 45.57 with the two singleton-context carrier witnesses for L and R. The forward direction of the cancellation equivalence turns the displayed contextual classifier into $\mathsf{FieldSingletonClassifier}(z, endpoint)$.

Apply Definition 45.260 to the final continuation witness $\mathsf{Cont}(w, s, z)$ and the classifier just obtained. This gives exclusions $\mathsf{ApartZero}_F(w) \to \bot$ and $\mathsf{ApartZero}_F(s) \to \bot$.

It remains to rule out the two exposed earlier factors. Apply Definition 45.734 to the first three continuation witnesses and the exterior witnesses $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. The reverse direction sends either $\mathsf{ApartZero}_F(h)$ or $\mathsf{ApartZero}_F(q)$ to $\mathsf{ApartZero}_F(w)$, contradicting the exclusion of w. Together with the direct exclusion of s, these three maps are exactly the displayed conjunction. □

Per-object template: $\mathsf{Field}^{\uparrow}$ apart-zero singleton-result exclusion; source BHist plus Cont; carrier predicate $\mathsf{ApartZero}_F$; classifier hsame transport; ledger the hidden singleton contexts; exactness predicate the displayed conjunction; status $\mathsf{ProofTarget}$.

Theorem 45.740 (Field apart-zero nested continuation singleton-result factor exclusion)**.** *For histories* $l, h, r, u, v, q, w, endpoint$: BHist *with* $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, *and* $\mathsf{Cont}(v, q, w)$, *if* $l\mathsf{hsamee}$, $r\mathsf{hsamee}$, *and* $\mathsf{FieldSingletonClassifier}(w, endpoint)$, *then*

$$(\mathsf{ApartZero}_F(h) \Rightarrow \mathsf{False}) \wedge (\mathsf{ApartZero}_F(q) \Rightarrow \mathsf{False}).$$

Proof. Apply Definition 45.6 to the classifier witness $\mathsf{FieldSingletonClassifier}(w, endpoint)$. The result gives in particular the endpoint comparison $w\mathsf{hsamee}$.

Assume $\mathsf{ApartZero}_F(h)$. The reverse direction of Definition 45.734, applied to the left disjunct and the three continuation witnesses, produces $\mathsf{ApartZero}_F(w)$. Applying this apart-zero witness to $w\mathsf{hsamee}$ gives False.

Assume $\mathsf{ApartZero}_F(q)$. The same reverse direction of the nested factor split, now applied to the right disjunct, again produces $\mathsf{ApartZero}_F(w)$. Applying it to $w\mathsf{hsamee}$ gives False. Pairing the two implications gives the displayed exclusion. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartZero nested continuation singleton result factor absurd`

Per-object template: traditional target zero-result factor exclusion for nested field multiplication; BEDC source BHist, Cont, $\mathsf{FieldSingletonClassifier}$, and $\mathsf{ApartZero}_F$; carrier apart-zero histories; classifier hsame transport; ledger the hidden singleton result endpoint and the two exposed factors; core theorem list the nested continuation factor split plus singleton endpoint classifier; exactness predicate the displayed paired exclusion; standard bridge ordinary zero product excludes nonzero factors under a singleton result; status $\mathsf{ProofTarget}$.

Definition 45.741 (Field apart-zero tail-factor spine)**.** *For histories $y, x : \mathsf{BHist}$, let $\mathsf{FieldApartTailFactor}(y, x)$ be the least relation on BHist generated by the following constructors:*

1. *If $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $l\mathsf{hsamee}$, and $r\mathsf{hsamee}$, then $\mathsf{FieldApartTailFactor}(v, h)$.*
2. *If $\mathsf{FieldApartTailFactor}(w, x)$ and $\mathsf{Cont}(w, q, z)$, then $\mathsf{FieldApartTailFactor}(z, x)$.*
3. *If $\mathsf{Cont}(w, q, z)$, then $\mathsf{FieldApartTailFactor}(z, q)$.*

Lean4 (definition) `BEDC.Derived.FieldUp.FieldApartTailFactor`

Lemma 45.742 (Field apart-tailfactor empty-context hsame transport)**.** *The empty-context generator is stable under hsame transport of its two continuation witnesses, exterior empty comparisons, middle history, and result.*

Proof. Transport the two Cont witnesses and the two exterior comparisons, then use the empty-context constructor. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor empty context hsame transport`

Lemma 45.743 (Field apart-zero tail-factor empty endpoint exclusion)**.** *For histories $y, x : \mathsf{BHist}$, if $\mathsf{FieldApartTailFactor}(y, x)$, $y\mathsf{hsamee}$, and $\mathsf{ApartZero}_F(x)$, then $\bot$.*

Proof. Induct on the generated tail-factor witness. In the empty-context generator, the nested empty-context apart-zero equivalence transports $\mathsf{ApartZero}_F(x)$ to the result endpoint, contradicting $y\mathsf{hsamee}$. In the carried-tail case, empty-result inversion gives an empty previous endpoint. In the exposed-factor case, it gives the exposed factor itself as e. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor empty endpoint exclusion`

Theorem 45.744 (Field apart-tailfactor empty endpoint exclusion hsame transport)**.** *For histories $y, x, y', x' : \mathsf{BHist}$, if $\mathsf{FieldApartTailFactor}(y, x)$, $y\mathsf{hsame}y'$, $x\mathsf{hsame}x'$, $y'\mathsf{hsamee}$, and $\mathsf{ApartZero}_F(x')$, then $\bot$.*

Proof. Transport the two displayed endpoint comparisons across hsame and apply Definition 45.743. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor empty endpoint exclusion hsame transport`

Lemma 45.745 (Field apart-zero tail-factor hsame transport)**.** *For histories* $y, y', x, x' : \mathsf{BHist}$*, if* $\mathsf{FieldApartTailFactor}(y, x)$*,* $y\mathsf{hsame}y'$*, and* $x\mathsf{hsame}x'$*, then* $\mathsf{FieldApartTailFactor}(y', x')$*.*

Proof. Induct on the generated witness $\mathsf{FieldApartTailFactor}(y, x)$.

In the empty-context constructor, transport the first continuation witness along $x\mathsf{hsame}x'$ and reflexivity of its other endpoints using Definition 4.41. Transport the second continuation witness along reflexivity of its source and factor endpoint, and along $y\mathsf{hsame}y'$ for the result. The two exterior witnesses $l\mathsf{hsame}e$ and $r\mathsf{hsame}e$ are unchanged, so constructor 1 rebuilds $\mathsf{FieldApartTailFactor}(y', x')$.

In the carried-tail constructor, apply the induction hypothesis to the stored factor witness, the reflexive comparison on its carried result, and the supplied $x\mathsf{hsame}x'$. Transport the final continuation result along $y\mathsf{hsame}y'$ by Definition 4.41, then reuse constructor 2.

In the exposed-factor constructor, transport the continuation argument by $x\mathsf{hsame}x'$ and its result by $y\mathsf{hsame}y'$ using Definition 4.41. Constructor 3 then gives the transported generated factor witness. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor hsame transport continuation package`

Per-object template: traditional target tail-factor stability; BEDC source BHist plus Cont; carrier FieldApartTailFactor; classifier hsame; stability obligation Definition 45.745; ledger transported continuation witnesses; core theorem Definition 45.745; exactness predicate invariance of generated factor witnesses; status ProofTarget.

Theorem 45.746 (Field apart-zero tail-factor continuation hsame closure)**.** *For histories* $w, w', x, x', q, z :$ BHist*, if* $\mathsf{FieldApartTailFactor}(w, x)$*,* $w\mathsf{hsame}w'$*,* $x\mathsf{hsame}x'$*, and* $\mathsf{Cont}(w', q, z)$*, then* $\mathsf{FieldApartTailFactor}(z, x')$*.*

Proof. Transport the carried tail-factor endpoints along the two hsame comparisons. The continuation witness then extends the generated spine by the carried-tail constructor, yielding the displayed endpoint and carried factor. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor continuation hsame closure`

Lemma 45.747 (Field apart-zero tail-factor result closure)**.** *For histories* $y, x : \mathsf{BHist}$*, if* $\mathsf{FieldApartTailFactor}(y, x)$*, then*

$$\mathsf{ApartZero}_F(x) \Rightarrow \mathsf{ApartZero}_F(y).$$

Proof. Induct on the generated witness $\mathsf{FieldApartTailFactor}(y, x)$.

In the initial empty-context constructor, apply Definition 45.733 to the two continuation witnesses and the two exterior comparisons. The reverse direction transports $\mathsf{ApartZero}_F(h)$ to $\mathsf{ApartZero}_F(v)$.

In the carried-tail constructor, the induction hypothesis first transports $\mathsf{ApartZero}_F(x)$ to $\mathsf{ApartZero}_F(w)$. Then apply the reverse direction of Definition 45.256 to the left endpoint disjunct, obtaining $\mathsf{ApartZero}_F(z)$.

In the exposed-factor constructor, apply the reverse direction of Definition 45.256 to the right factor disjunct. This sends $\mathsf{ApartZero}_F(q)$ to $\mathsf{ApartZero}_F(z)$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor result closure`

Theorem 45.748 (Field apart-zero tail-factor continuation result closure)**.** *For histories* $w, x, q, z :$ BHist*, if* $\mathsf{FieldApartTailFactor}(w, x)$*,* $\mathsf{Cont}(w, q, z)$*, and* $\mathsf{ApartZero}_F(x)$*, then* $\mathsf{ApartZero}_F(z)$*.*

Proof. First apply tail-factor result closure to obtain $\mathsf{ApartZero}_F(w)$ from the generated witness and $\mathsf{ApartZero}_F(x)$. The continuation endpoint split for $\mathsf{Cont}(w, q, z)$ closes the result by the left factor disjunct. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor continuation result closure`

Theorem 45.749 (Field apart-zero finite tail singleton-result exclusion)**.** *For histories* $y, x, endpoint$: BHist*, if* $\mathsf{FieldApartTailFactor}(y, x)$ *and* $\mathsf{FieldSingletonClassifier}(y, endpoint)$*, then*

$$\mathsf{ApartZero}_F(x) \Rightarrow \bot.$$

Proof. Apply Definition 45.747 to the generated tail-factor witness. This turns any $\mathsf{ApartZero}_F(x)$ witness into $\mathsf{ApartZero}_F(y)$.

Apply Definition 45.6 to the singleton endpoint classifier. The endpoint theorem gives the comparison $y\mathsf{hsamee}$.

Finally, transport the apart-zero witness along the endpoint comparison. The result is the contradiction $\bot$, since $\mathsf{ApartZero}_F(y)$ rejects $y\mathsf{hsamee}$. □

Per-object template: traditional target finite product zero-factor exclusion; BEDC source BHist, Cont, $\mathsf{ApartZero}_F$, and $\mathsf{FieldSingletonClassifier}$; carrier generated tail-factor witnesses; classifier hsame; core theorem list Definition 45.747 and Definition 45.749; exactness predicate generated factor-to-result closure plus singleton-result rejection; status $\mathsf{ProofTarget}$.

Theorem 45.750 (Field apart-zero finite tail contextual singleton-result exclusion)**.** *For histories* $L, R, y, x, endpoint$: BHist*, if* $\mathsf{FieldSingletonCarrier}(L)$*,* $\mathsf{FieldSingletonCarrier}(R)$*,* $\mathsf{FieldApartTailFactor}(y, x)$*, and* $\mathsf{FieldSingletonClassifier}(\mathsf{append}(L, y), \mathsf{append}(R, endpoint))$*, then*

$$\mathsf{ApartZero}_F(x) \Rightarrow \bot.$$

Proof. First cancel the carried singleton contexts. Apply Definition 45.57 to the two carrier witnesses and the appended singleton classifier. The forward direction gives $\mathsf{FieldSingletonClassifier}(y, endpoint)$.

Now apply Definition 45.749 to the generated tail-factor witness $\mathsf{FieldApartTailFactor}(y, x)$ and the base singleton classifier. This produces $\mathsf{ApartZero}_F(x) \Rightarrow \bot$. □

Lean4 ✓ `BEDC.Derived.FieldUp.FieldApartTailFactor singleton context exclusion`

Per-object template: traditional target finite product zero-factor exclusion under hidden singleton contexts; BEDC source BHist, Cont, $\mathsf{FieldApartTailFactor}$, $\mathsf{FieldSingletonClassifier}$, and $\mathsf{ApartZero}_F$; carrier generated tail-factor witnesses; classifier hsame; stability from singleton context cancellation and finite tail result closure; ledger hidden singleton contexts plus generated factor witness; core theorem list Definition 45.57 and Definition 45.749; exactness predicate contextual singleton-result rejection; status $\mathsf{ProofTarget}$.

Theorem 45.751 (Field apart-zero three-tail tail-factor coverage)**.** *For histories* $l, h, r, u, v, q, w, s, z, t, y$: BHist *with* $\mathsf{Cont}(l, h, u)$*,* $\mathsf{Cont}(u, r, v)$*,* $\mathsf{Cont}(v, q, w)$*,* $\mathsf{Cont}(w, s, z)$*, and* $\mathsf{Cont}(z, t, y)$*, if* $l\mathsf{hsamee}$ *and* $r\mathsf{hsamee}$*, then*

$$\mathsf{FieldApartTailFactor}(y, h) \wedge \mathsf{FieldApartTailFactor}(y, q) \wedge \mathsf{FieldApartTailFactor}(y, s) \wedge \mathsf{FieldApartTailFactor}(y, t).$$

Proof. We first build the hidden initial factor. The initial tail-factor constructor applied to $\mathsf{Cont}(l, h, u)$, $\mathsf{Cont}(u, r, v)$, $l\mathsf{hsamee}$, and $r\mathsf{hsamee}$ gives $\mathsf{FieldApartTailFactor}(v, h)$. Carry this witness successively across $\mathsf{Cont}(v, q, w)$, $\mathsf{Cont}(w, s, z)$, and $\mathsf{Cont}(z, t, y)$ by the carried-tail constructor to obtain $\mathsf{FieldApartTailFactor}(y, h)$.

Next expose the q factor at the first continuation after the initial empty-context block. The exposed-factor constructor applied to $\mathsf{Cont}(v, q, w)$ gives $\mathsf{FieldApartTailFactor}(w, q)$. Carry it across $\mathsf{Cont}(w, s, z)$ and $\mathsf{Cont}(z, t, y)$ to obtain $\mathsf{FieldApartTailFactor}(y, q)$.

The s factor is exposed in the same generated spine at the next step. Apply the exposed-factor constructor to $\mathsf{Cont}(w, s, z)$, giving $\mathsf{FieldApartTailFactor}(z, s)$, and then carry this witness across $\mathsf{Cont}(z, t, y)$ to obtain $\mathsf{FieldApartTailFactor}(y, s)$.

Finally, apply the exposed-factor constructor directly to $\mathsf{Cont}(z, t, y)$. This gives $\mathsf{FieldApartTailFactor}(y, t)$. Pair the four generated witnesses in the displayed order. □

Per-object template: traditional target finite product factor coverage; BEDC source BHist, Cont, and $\mathsf{FieldApartTailFactor}$; carrier generated tail-factor witnesses; classifier hsame; ledger the exposed factors h, q, s, t; core theorem this block plus Definition 45.750; exactness predicate factor coverage for generated exclusion; status $\mathsf{ProofTarget}$.

Theorem 45.752 (RatUp FieldUp algebra hierarchy selector-chain exit)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then one concrete* BHist *package contains all of the following rows:*

1. *every row of Definition 45.616;*
2. *every shared certificate and operation-table row of Definition 45.534;*
3. *every selector-chain final-boundary row of Definition 45.765 for the selector pairs* (s, t)*,* (u, v)*, and* (w, z)*.*

Equivalently, the concrete source tuple, the concrete denominator tuple, and the three certified selector pairs determine a single $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package for the Phase Algebra Hierarchy interface.*

Proof. First instantiate Definition 45.616 at the source tuple $\mathsf{RatUpFieldSource}_F$. This supplies the algebra hierarchy exit rows over the common carrier BHist, the RatUp history classifier, and the concrete denominator endpoints.

Next instantiate Definition 45.534 at the same source tuple, denominator tuple, and concrete operation table of Definition 45.587. This supplies the shared certificate rows and the operation-table rows carried by the same BHist package.

Finally instantiate Definition 45.765 with the histories, carrier witnesses, support witness, selectors s, t, u, v, w, z, and contexts L, R, r. This contributes the three final-boundary packages for (s, t), (u, v), and (w, z) together with the chained componentwise R^u_D and $\mathsf{RatHistoryClassifier}$ rows.

The three cited packages name the same source $\mathsf{RatUpFieldSource}_F$, the same carrier BHist, the same RatUp classifier, and the same concrete denominator tuple. Conjunction introduction repacks the common carrier, classifier, denominator, certificate, operation-table, and selector-chain boundary rows as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ algebra hierarchy exit package. □

Per-object template: traditional target selector-chain-stable Phase Algebra Hierarchy exit for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier and classifier are the RatUp field source rows; stability obligations Definition 45.616, Definition 45.534, and Definition 45.765; ledger selectors s, t, u, v, w, z, histories h, k, ℓ, contexts L, R, r, and the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$; exactness predicate one selector-chain-stable concrete exit package; status $\mathsf{ProofTarget}$.

Theorem 45.753 (RatUp FieldUp exit object operation-table lock)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete exit object of Definition 45.617. The same* BHist *row package simultaneously satisfies*

$$\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F),$$

the concrete operation-table coverage of Definition 45.534, and the selector-independent final boundary row of Definition 45.766. Equivalently, $\mathcal{E}_F$ *is a completed concrete exit object whose carrier, classifier, denominator, contextual, affine, inverse, nonzero, and final boundary rows are all borne by the one concrete* BHist *package.*

Proof. Apply Definition 45.618. This gives $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ for the paper package determined by Definition 45.617.

Apply Definition 45.534. It identifies the carrier, classifier, denominator tuple, contextual action, context-pair monoid, affine row source, inverse rows, nonzero rows, and exit-noncollapse rows as rows of the same concrete BHist package.

Finally apply Definition 45.766. This supplies the selector-independent final boundary row for that concrete exit object. Conjoining these three cited results yields a single completed $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object that carries the operation table and the selector-independent boundary simultaneously. □

Theorem 45.754 (RatUp FieldUp algebra hierarchy selector-independent exit)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then one concrete* BHist *package contains all of the following rows:*

1. *every row of Definition 45.616;*
2. *every shared certificate and operation-table row of Definition 45.534;*
3. *every selector-independent final boundary row of Definition 45.764.*

Equivalently, the concrete source tuple, carrier, classifier, denominator tuple, and endpoint selectors determine a single $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit certificate that serves the Phase Algebra Hierarchy interface independently of the certified selector pair.*

Proof. First instantiate Definition 45.616 at the concrete source tuple $\mathsf{RatUpFieldSource}_F$. This gives the paper algebra hierarchy rows over the carrier BHist, with the RatUp history classifier and the concrete denominator endpoints.

Next instantiate Definition 45.534 at the same source tuple and the operation table of Definition 45.587. Its rows use the same carrier, classifier, denominator coordinate, contextual action, context-pair coordinate, and affine coordinate as the hierarchy package.

Finally instantiate Definition 45.764 with the given histories, denominator-carrier witnesses, support witness, selectors s, t, u, v, and contexts L, R, r. This supplies the two final-boundary packages and the componentwise R^u_D and $\mathsf{RatHistoryClassifier}$ rows for the selected endpoints.

The three instantiations name the same concrete source tuple $\mathsf{RatUpFieldSource}_F$, the same BHist carrier, the same RatUp classifier, and the same concrete denominator tuple. Conjunction introduction therefore packages their row families as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ algebra hierarchy exit certificate. □

Per-object template: concrete Phase Algebra Hierarchy exit package for $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h, k, \ell)$, $E_t(h, k, \ell)$, $E_u(h, k, \ell)$, and $E_v(h, k, \ell)$; carrier rows the algebra hierarchy exit witness,

the assembled certificate operation-table coverage, and the selector-independent final boundary; classifier rows $\mathsf{RatHistoryClassifier}$ together with the componentwise R^u_D rows; stability obligations hierarchy exit witness, assembled certificate operation-table coverage, and selector-independent final boundary; ledger histories h, k, ℓ, selectors s, t, u, v, contexts L, R, r, and the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$; exactness predicate one concrete selector-independent Phase Algebra Hierarchy exit certificate; status $\mathsf{ProofTarget}$.

Per-object template: traditional target two-step selector obstruction chain for $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier rows C^u_D and the concrete selector-obstruction carriers; classifier rows R^u_D and $\mathsf{RatHistoryClassifier}$; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; stability obligations the two adjacent complete selector exit obstruction certificates and the selector-chain final boundary; exactness one two-step obstruction-chain package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.755 (RatUp FieldUp algebra hierarchy chain-stable exit)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then one concrete* BHist *package contains all of the following rows:*

1. *every row of Definition 45.616;*
2. *every shared certificate and operation-table row of Definition 45.534;*
3. *the assembled obstruction certificate for* (s, t, u, v) *from Definition 45.763;*
4. *the selector-chain final boundary rows of Definition 45.765 for* (s, t)*,* (u, v)*, and* (w, z)*, including the chained* R^u_D *rows and the chained* $\mathsf{RatHistoryClassifier}$ *endpoint rows.*

Equivalently, the concrete FieldUp over RatUp source determines a chain-stable Phase Algebra Hierarchy exit certificate whose carrier, classifier, denominator, operation-table, obstruction, and selector-chain boundary rows are all borne by the same concrete BHist *package.*

Proof. First instantiate Definition 45.616 at $\mathsf{RatUpFieldSource}_F$. This supplies the hierarchy rows over the common carrier BHist, with the RatUp classifier and denominator rows.

Next instantiate Definition 45.534 at the concrete operation table of Definition 45.587. This supplies the shared certificate rows and operation-table rows for the same source tuple.

Then instantiate Definition 45.768 with the displayed denominator tuple, histories, carrier witnesses, selectors s, t, u, v, w, z, contexts L, R, r, and support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This supplies the assembled obstruction certificate and the selector-chain boundary rows.

Finally cite Definition 45.618 only to keep all row families inside the same concrete exit object. Conjunction introduction packages the hierarchy, operation-table, obstruction, and chain-boundary rows as one chain-stable Phase Algebra Hierarchy exit certificate. □

Per-object template: traditional target chain-stable $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier and classifier the RatUp and denominator rows; stability obligations the hierarchy, operation-table, obstruction, and chain-boundary rows; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; exactness one chain-stable Phase Algebra Hierarchy exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.756 (RatUp FieldUp concrete algebra hierarchy obstruction-chain exit package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and contexts*

L, R, r : BHist, let $E_q(h,k,\ell)$ be the bracketing endpoint selector of Definition 45.606 for each $q : \mathsf{BMark}$. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then the concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ contains one BHist package with all of the following rows:

1. *every row of Definition 45.752;*
2. *the assembled obstruction certificate of Definition 45.763 for (s,t,u,v);*
3. *the concrete final obstruction-chain rows of Definition 45.768.*

In particular, the same concrete exit object contains the chained denominator rows

$$R^u_D(E_s(h,k,\ell), E_w(h,k,\ell)) \wedge R^u_D(E_t(h,k,\ell), E_z(h,k,\ell))$$

and the chained endpoint classifier rows

$$\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_w(h,k,\ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h,k,\ell), E_z(h,k,\ell)).$$

Proof. First instantiate Definition 45.752 with the displayed source, denominator tuple, histories, carrier witnesses, support witness, selectors s,t,u,v,w,z, and contexts L,R,r : BHist. This supplies the algebra-hierarchy exit witness, shared operation-table coverage, and selector-chain final-boundary rows over one concrete BHist package.

Next instantiate Definition 45.768 with the same source, concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors s,t,u,v,w,z, and contexts L,R,r. This supplies the assembled obstruction certificate for (s,t,u,v) and the concrete final obstruction-chain rows, including the chained R^u_D and $\mathsf{RatHistoryClassifier}$ comparisons from E_s, E_t to E_w, E_z.

Both cited packages live over $\mathsf{RatUpFieldSource}_F$, the concrete denominator tuple, and the common carrier BHist. Conjunction introduction repacks their rows as one concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ exit object for $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$. □

Per-object template: traditional target algebra hierarchy obstruction-chain exit package for $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; concrete exit object Definition 45.617; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations algebra hierarchy selector-chain exit, assembled obstruction certificate, and concrete final obstruction-chain boundary package; ledger selector sextuple s,t,u,v,w,z plus contexts L,R,r; exactness one concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.757 (RatUp FieldUp concrete two-step obstruction chain package)**.** *For $\mathsf{RatUpFieldSource}_F$, Definition 45.617, the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, histories h,k,ℓ : BHist with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, selectors s,t,u,v,w,z : BMark, and contexts L,R,r : BHist, let $E_q(h,k,\ell)$ be the bracketing endpoint selector of Definition 45.606 for each q : BMark. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then the same concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ contains the following three row families:*

1. *the full assembled obstruction certificate of Definition 45.763 for (s,t,u,v);*
2. *the full assembled obstruction certificate of Definition 45.763 for (u,v,w,z);*
3. *the selector-chain final boundary rows of Definition 45.765 for (s,t), (u,v), and (w,z), including the chained denominator rows*

$$R^u_D(E_s(h,k,\ell), E_w(h,k,\ell)) \wedge R^u_D(E_t(h,k,\ell), E_z(h,k,\ell))$$

and the chained endpoint classifier rows

$$\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_w(h,k,\ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h,k,\ell), E_z(h,k,\ell)).$$

Proof. First instantiate Definition 45.767 with the displayed source, concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts $L, R, r : \mathsf{BHist}$. This supplies the assembled obstruction certificate for the adjacent selector replacement from (s, t) to (u, v).

Next instantiate the same theorem with the same source, concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors u, v, w, z, and contexts $L, R, r : \mathsf{BHist}$. This supplies the assembled obstruction certificate for the adjacent selector replacement from (u, v) to (w, z).

Then instantiate Definition 45.765 with the six selectors s, t, u, v, w, z over the same denominator tuple, histories, carrier witnesses, support witness, and contexts. This contributes the three selector-pair boundary rows together with the chained R^u_D rows and chained $\mathsf{RatHistoryClassifier}$ rows from the first endpoint pair to the final endpoint pair.

Finally cite Definition 45.618 only to keep both adjacent obstruction certificates and the chained final boundary rows inside the same concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$. Conjunction introduction packages the two adjacent obstruction certificates and the selector-chain final boundary as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit-object certificate. □

Per-object template: traditional target two-step selector obstruction chain for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier rows C^u_D and the concrete selector-obstruction carriers; classifier rows R^u_D and $\mathsf{RatHistoryClassifier}$; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; stability obligations the two adjacent complete selector exit obstruction certificates and the selector-chain final boundary; exactness one two-step obstruction-chain package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.758 (RatUp FieldUp adjacent selector-chain obstruction coverage)**.** *For* $\mathsf{RatUpFieldSource}_F$*, Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the same concrete* $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ *contains all of the following row families:*

1. *the complete selector exit obstruction certificate of Definition 45.763 for* (s, t, u, v)*;*
2. *the complete selector exit obstruction certificate of Definition 45.763 for* (u, v, w, z)*;*
3. *the selector-chain final boundary rows of Definition 45.765 for* (s, t)*,* (u, v)*, and* (w, z)*, including the chained denominator rows*

$$R^u_D(E_s(h, k, \ell), E_w(h, k, \ell)) \wedge R^u_D(E_t(h, k, \ell), E_z(h, k, \ell))$$

and the chained endpoint classifier rows

$$\mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_w(h, k, \ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h, k, \ell), E_z(h, k, \ell)).$$

Proof. Apply Definition 45.763 to the displayed denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts $L, R, r : \mathsf{BHist}$. This gives the complete selector exit obstruction certificate for the first adjacent link (s, t, u, v).

Apply the same theorem again with selectors u, v, w, z and the same denominator tuple, histories, carrier witnesses, support witness, and contexts. This gives the complete selector exit obstruction certificate for the terminal adjacent link (u, v, w, z).

Apply Definition 45.765 to the six selectors s, t, u, v, w, z over the same denominator tuple, histories, carrier witnesses, support witness, and contexts. This supplies the final boundary rows for (s, t), (u, v), and (w, z), including the chained R^u_D rows and the chained $\mathsf{RatHistoryClassifier}$ endpoint rows.

Finally cite Definition 45.618 only to keep all three row families inside the same concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$. Conjunction introduction packages the two adjacent complete obstruction certificates and the selector-chain final boundary as one concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ coverage certificate. □

Per-object template: traditional target adjacent selector-chain obstruction coverage for $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$; BEDC source $\mathsf{RatUpFieldSource}_F$ over BHist; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations the two complete obstruction packages plus selector-chain final boundary; ledger selector sextuple s, t, u, v, w, z and contexts L, R, r; core theorem this block; exactness one adjacent-chain obstruction coverage package; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.759 (RatUp FieldUp final certificate selector noncollapse boundary)**.** *For Definition 45.617, the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, selectors $s, t : \mathsf{BMark}$, and contexts $L, R, r : \mathsf{BHist}$, let $E_s(h, k, \ell)$ and $E_t(h, k, \ell)$ be the corresponding bracketing endpoint selectors of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then one concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit boundary simultaneously contains the following rows:*

1. *the final certificate noncollapse action package of Definition 45.669;*
2. *$C^u_D(E_s(h, k, \ell))$, $C^u_D(E_t(h, k, \ell))$, and $R^u_D(E_s(h, k, \ell), E_t(h, k, \ell))$;*
3. *$\mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_t(h, k, \ell))$;*
4. $$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h, k, \ell)), \mathsf{append}(R, E_t(h, k, \ell))) \to \bot;$$
5. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, E_s(h, k, \ell))) \to \bot;$$
6. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(E_s(h, k, \ell), r)) \to \bot;$$
7. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, E_s(h, k, \ell)));$$
8. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(E_s(h, k, \ell), r)).$$

Proof. Apply Definition 45.669 to Definition 45.617. This fixes the concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ certificate boundary whose source object and action rows are consumed by the hierarchy exit criterion.

Next apply Definition 45.623 to the same denominator tuple, histories, carrier witnesses, support witness, selectors, and contexts $L, R : \mathsf{BHist}$. This contributes the two selected carrier rows, the selected denominator-classifier row, and the arbitrary-context singleton-classifier exclusion for the pair $(E_s(h, k, \ell), E_t(h, k, \ell))$.

Apply Definition 45.614 with the same selector pair and the context $r : \mathsf{BHist}$. Its selector-independent row package supplies $\mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_t(h, k, \ell))$ together with contextual nonzero rows for the selected left endpoint.

Finally apply Definition 45.624 to the selector s and context $r : \mathsf{BHist}$. This reads the one-sided singleton-carrier exclusions and the two $\mathsf{fieldSingletonEmptyNonZero}$ witnesses as rows of the same concrete exit boundary, so the final certificate and selector bracketing data are consumed as one FieldUp object. □

Per-object template: traditional target final selector noncollapse boundary inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, and Definition 45.617; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h,k,\ell)$ and $E_t(h,k,\ell)$; carrier rows $C^u_D(E_s(h,k,\ell))$, $C^u_D(E_t(h,k,\ell))$, and $\mathsf{RatHistoryCarrier}$; classifier rows $R^u_D(E_s(h,k,\ell), E_t(h,k,\ell))$ and $\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_t(h,k,\ell))$; stability obligations final noncollapse action, selector exit subpackage, selector independence, and contextual nonzero rows; ledger selector pair plus contexts L, R, r; exactness predicate one concrete FieldUp exit boundary; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.760 (RatUp FieldUp selector-pair strict exit determinacy)**.** *For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ and histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, let $E_q(h,k,\ell)$ be the endpoint selector of Definition 45.606 for each $q : \mathsf{BMark}$. If $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ holds, then for all selector pairs $s, t, u, v : \mathsf{BMark}$ and contexts $L, R : \mathsf{BHist}$ the following concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package is determined:*

1.

$$R^u_D(E_s(h,k,\ell), E_u(h,k,\ell)) \wedge R^u_D(E_t(h,k,\ell), E_v(h,k,\ell));$$

2.

$$\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_u(h,k,\ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h,k,\ell), E_v(h,k,\ell));$$

3.

$$C^u_D(E_s(h,k,\ell)) \wedge C^u_D(E_t(h,k,\ell)) \wedge R^u_D(E_s(h,k,\ell), E_t(h,k,\ell)) \wedge (\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_s(h,k,\ell)$$

4.

$$C^u_D(E_u(h,k,\ell)) \wedge C^u_D(E_v(h,k,\ell)) \wedge R^u_D(E_u(h,k,\ell), E_v(h,k,\ell)) \wedge (\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, E_u(h,k,\ell$$

Proof. Apply Definition 45.634 to the selector pair (s,u) and contexts $L, R : \mathsf{BHist}$. Its strict exit row contains the denominator classifier row $R^u_D(E_s(h,k,\ell), E_u(h,k,\ell))$.

Apply the same strict exit package to the selector pair (t,v) and the same contexts. This supplies $R^u_D(E_t(h,k,\ell), E_v(h,k,\ell))$.

Apply Definition 45.612 to the selected pairs (s,u) and (t,v). This gives $\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_u(h,k,$ and $\mathsf{RatHistoryClassifier}(E_t(h,k,\ell), E_v(h,k,\ell))$.

Next apply Definition 45.634 to the selector pair (s,t) and contexts $L, R : \mathsf{BHist}$. The resulting strict exit row is exactly the third displayed conjunction.

Finally apply the same strict exit package to the selector pair (u,v) and the same contexts. Its row is the fourth displayed conjunction, and together with the two componentwise classifier rows it assembles the stated deterministic selector-pair package. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup selector pair strict exit determinacy`

Per-object template: traditional target selector-pair determinacy for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}$; carrier C^u_D; classifier R^u_D plus $\mathsf{RatHistoryClassifier}$; stability obligations selector strict exit row coverage and Rat classifier coverage; ledger the two selector pairs and contexts L, R; core theorem this block; exactness predicate componentwise selector determinacy; status $\mathsf{ProofTarget}$.

Theorem 45.761 (RatUp FieldUp bracketing selector complete obstruction package)**.** *For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *and histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$, $C^u_D(k)$, *and* $C^u_D(\ell)$, *let* $E_s(h, k, \ell)$ *be the endpoint selector of Definition 45.606. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then for all selectors* $s, t : \mathsf{BMark}$ *and contexts* $L, R : \mathsf{BHist}$, *setting* $a := E_s(h, k, \ell)$ *and* $b := E_t(h, k, \ell)$ *gives one concrete selector obstruction package with the following rows:*

1. $a\mathsf{hsame}b$;
2. $R^u_D(a, b)$;
3. $\mathsf{RatHistoryClassifier}(a, b)$;
4. $$\mathsf{fieldSingletonEmptyClassifier}(\mathsf{append}(L, a), \mathsf{append}(R, b)) \to \bot;$$
5. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, a)) \to \bot;$$
6. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, a));$$
7. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(a, R)) \to \bot;$$
8. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(a, R));$$
9. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(L, b)) \to \bot;$$
10. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(L, b));$$
11. $$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(b, R)) \to \bot;$$
12. $$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(b, R)).$$

Proof. First apply Definition 45.614 to the same denominator tuple, carrier witnesses, support witness, selector pair, and the context $L : \mathsf{BHist}$. This supplies $a\mathsf{hsame}b$ and $\mathsf{RatHistoryClassifier}(a, b)$ as the common selector-independent rows.

Next apply Definition 45.623 to the selector pair $s, t : \mathsf{BMark}$ and the contexts $L, R : \mathsf{BHist}$. Its exit row gives $R^u_D(a, b)$ and the singleton-classifier exclusion for $(\mathsf{append}(L, a), \mathsf{append}(R, b))$.

We now build the contextual nonzero rows for the left endpoint. Apply Definition 45.624 to the selector s with context L to obtain the carrier exclusion and $\mathsf{fieldSingletonEmptyNonZero}$ witness for $\mathsf{append}(L, a)$. Apply the same theorem to s with context R to obtain the corresponding right-context rows for $\mathsf{append}(a, R)$.

Finally apply Definition 45.624 to the selector t with context L, and then with context R. These two applications give the carrier exclusions and $\mathsf{fieldSingletonEmptyNonZero}$ witnesses for $\mathsf{append}(L, b)$ and $\mathsf{append}(b, R)$. Replacing a and b by the displayed selector endpoints assembles the concrete obstruction package. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup bracketing selector complete obstruction package`

Per-object template: traditional target bracketing-selector complete obstruction package inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, the denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, and endpoint selectors E_s, E_t; carrier C^u_D, $\mathsf{RatHistoryCarrier}$, and contextual $\mathsf{fieldSingletonEmptyCarrier}$ rejection; classifier R^u_D, $\mathsf{RatHistoryClassifier}$, and $\mathsf{fieldSingletonEmptyClassifier}$ rejection; stability obligations selector independence, selector exit subpackage, and contextual nonzero rows for both selected endpoints against the two external contexts; ledger selector pair plus contexts L, R; exactness predicate one concrete selector obstruction certificate; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 45.762 (RatUp FieldUp final certificate two-selector noncollapse boundary)**.** *For Definition 45.617, the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, selectors $s, t : \mathsf{BMark}$, and contexts $L, R, r : \mathsf{BHist}$, let $E_s(h, k, \ell)$ and $E_t(h, k, \ell)$ be the corresponding bracketing endpoint selectors of Definition 45.606. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the same concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit boundary contains every row of Definition 45.759 together with the following right-selector rows:*

1.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, E_t(h, k, \ell))) \to \bot;$$

2.
$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(E_t(h, k, \ell), r)) \to \bot;$$

3.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, E_t(h, k, \ell))));$$

4.
$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(E_t(h, k, \ell), r)).$$

Proof. First apply Definition 45.759 to Definition 45.617, the denominator tuple, the histories, the three carrier witnesses, the selectors s, t, the contexts L, R, r, and the $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness. This fixes the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit boundary and supplies every row of the final selector noncollapse boundary.

Next apply Definition 45.614 with the same selector pair s, t and the same context $r : \mathsf{BHist}$. The right endpoint rows of that package give the two singleton-carrier exclusions and the two $\mathsf{fieldSingletonEmptyNonZero}$ witnesses for $E_t(h, k, \ell)$, so the boundary consumes both selected endpoints under the same concrete certificate. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup final certificate two selector noncollapse boundary`

Per-object template: traditional target final two-selector noncollapse boundary inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, and Definition 45.617; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h, k, \ell)$ and $E_t(h, k, \ell)$; carrier rows those of the final selector boundary plus the right-selector singleton-carrier exclusions; classifier rows those of the final selector boundary and the selector-independence package; nonzero rows the left-selector witnesses of the final selector boundary plus the two right-selector witnesses; stability obligations final selector noncollapse boundary and bracketing selector independence; ledger selector pair plus contexts L, R, r; exactness predicate one concrete FieldUp exit boundary with both endpoint contexts; downstream consumer final certificate bracketing endpoint consumption; status $\mathsf{ProofTarget}$.

Theorem 45.763 (RatUp FieldUp complete selector exit obstruction certificate)**.** *For Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package of Definition 45.618 contains, as one assembled certificate, all of the following rows:*

1. *every selector-pair strict row of Definition 45.760 for the selector comparison* (s, t) *against* (u, v) *and contexts* L, R*;*
2. *every* (s, t) *obstruction row of Definition 45.761, with* $a = E_s(h, k, \ell)$ *and* $b = E_t(h, k, \ell)$*;*
3. *every row of Definition 45.762 for the selector pair* (s, t)*, including the right-selector rows*

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(r, E_t(h, k, \ell))) \to \bot,$$

$$\mathsf{fieldSingletonEmptyCarrier}(\mathsf{append}(E_t(h, k, \ell), r)) \to \bot,$$

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(r, E_t(h, k, \ell))),$$

and

$$\mathsf{fieldSingletonEmptyNonZero}(\mathsf{append}(E_t(h, k, \ell), r)).$$

Proof. First instantiate Definition 45.760 with the given denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts $L, R : \mathsf{BHist}$. This supplies the two componentwise selector comparisons and the strict exit rows for the two selected pairs.

Next instantiate Definition 45.761 with the same denominator tuple, histories, carrier witnesses, support witness, selector pair (s, t), and contexts $L, R : \mathsf{BHist}$. The resulting rows are exactly the concrete obstruction rows for $a = E_s(h, k, \ell)$ and $b = E_t(h, k, \ell)$.

Now instantiate Definition 45.762 with the same concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selector pair (s, t), contexts L, R, and right context $r : \mathsf{BHist}$. This contributes the final boundary rows and the four displayed right-selector noncollapse rows for $E_t(h, k, \ell)$.

Finally use Definition 45.618 only to identify these rows as rows of the concrete Definition 45.617. Conjunction introduction then repacks the three row families into one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate. □

Per-object template: traditional target complete selector exit obstruction certificate inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, and Definition 45.617; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h, k, \ell)$, $E_t(h, k, \ell)$, $E_u(h, k, \ell)$, and $E_v(h, k, \ell)$; carrier rows selector strict exit carriers, contextual obstruction carrier exclusions, and right-selector noncollapse carrier exclusions; classifier rows componentwise denominator comparisons, $\mathsf{RatHistoryClassifier}$ rows, and contextual singleton-classifier rejection; nonzero rows strict endpoint nonzero witnesses, contextual endpoint witnesses, and final right-selector witnesses; stability obligations selector-pair strict exit determinacy, bracketing selector complete obstruction, and final certificate two-selector noncollapse; ledger selector comparison (s, t) against (u, v) plus contexts L, R, r; exactness predicate one concrete selector exit obstruction certificate; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.764 (RatUp FieldUp selector-independent final boundary)**.** *For Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing*

endpoint selector of Definition 45.606 for each $q : \mathsf{BMark}$. *If* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then the same concrete* $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ *final exit boundary contains the following rows:*

1. *every row of Definition 45.762 for the selector pair* (s,t)*;*
2. *every row of Definition 45.762 for the selector pair* (u,v)*;*
3. $$R^u_D(E_s(h,k,\ell),E_u(h,k,\ell)) \wedge R^u_D(E_t(h,k,\ell),E_v(h,k,\ell));$$
4. $$\mathsf{RatHistoryClassifier}(E_s(h,k,\ell),E_u(h,k,\ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h,k,\ell),E_v(h,k,\ell)).$$

Proof. First apply Definition 45.760 to the histories, carrier witnesses, support witness, selectors s,t,u,v, and contexts $L,R : \mathsf{BHist}$. Its componentwise rows give $R^u_D(E_s(h,k,\ell),E_u(h,k,\ell))$, $R^u_D(E_t(h,k,\ell),E_v(h,k,\ell))$, and the two displayed $\mathsf{RatHistoryClassifier}$ rows. The same application also supplies the strict exit rows for both selector pairs (s,t) and (u,v).

Next apply Definition 45.761 to the pair (s,t) and then to the pair (u,v) with the same contexts $L,R : \mathsf{BHist}$. These two obstruction packages provide the contextual singleton-classifier exclusions and contextual nonzero rows consumed by the final certificate boundary for each selected pair.

Finally apply Definition 45.762 once with selectors s,t and once with selectors u,v, keeping the same concrete exit object, denominator tuple, histories, support witness, and context $r : \mathsf{BHist}$. Combining those two final-boundary packages with the componentwise comparison rows from the strict determinacy theorem yields one selector-independent final boundary for the concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ certificate. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup selector independent final boundary`

Per-object template: traditional target selector-independent final boundary inside $\mathsf{Field}^\uparrow(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, and Definition 45.617; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h,k,\ell)$, $E_t(h,k,\ell)$, $E_u(h,k,\ell)$, and $E_v(h,k,\ell)$; carrier rows the two final certificate boundaries plus the componentwise denominator comparisons; classifier rows $\mathsf{RatHistoryClassifier}$ for the two componentwise selector comparisons; nonzero rows the contextual witnesses consumed by both final certificate boundaries; stability obligations selector-pair strict exit determinacy, bracketing selector complete obstruction, and final certificate two-selector noncollapse; ledger selector pairs (s,t) and (u,v) plus contexts L,R,r; exactness predicate one final FieldUp exit boundary independent of certified selector-pair replacement; downstream consumer stable algebra-hierarchy exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.765 (RatUp FieldUp selector-chain final boundary)**.** *For Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h,k,\ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s,t,u,v,w,z : \mathsf{BMark}$*, and contexts* $L,R,r : \mathsf{BHist}$*, let* $E_q(h,k,\ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$. *If* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then the same concrete* $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ *final exit boundary contains the following rows:*

1. *every row of Definition 45.762 for the selector pair* (s,t)*;*
2. *every row of Definition 45.762 for the selector pair* (u,v)*;*
3. *every row of Definition 45.762 for the selector pair* (w,z)*;*
4. $$R^u_D(E_s(h,k,\ell),E_w(h,k,\ell)) \wedge R^u_D(E_t(h,k,\ell),E_z(h,k,\ell));$$

5.

$$\mathsf{RatHistoryClassifier}(E_s(h,k,\ell), E_w(h,k,\ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h,k,\ell), E_z(h,k,\ell)).$$

Proof. First apply Definition 45.764 to the selector comparison from (s,t) to (u,v) with the same concrete exit object, denominator tuple, histories, carrier witnesses, support witness, and contexts $L, R, r : \mathsf{BHist}$. This gives the final-boundary rows for (s,t) and (u,v) together with the componentwise denominator and $\mathsf{RatHistoryClassifier}$ rows from E_s to E_u and from E_t to E_v.

Next apply Definition 45.764 to the selector comparison from (u,v) to (w,z) over the same data. This gives the final-boundary rows for (u,v) and (w,z) together with the componentwise denominator and $\mathsf{RatHistoryClassifier}$ rows from E_u to E_w and from E_v to E_z.

It remains to collapse the two adjacent replacements into endpoint rows. Use the transitivity field of the concrete denominator classifier R^u_D to compose

$$R^u_D(E_s(h,k,\ell), E_u(h,k,\ell)) \quad \text{with} \quad R^u_D(E_u(h,k,\ell), E_w(h,k,\ell))$$

and likewise compose the E_t to E_v row with the E_v to E_z row.

Finally use the transitivity field of $\mathsf{RatHistoryClassifier}$ in the same two components. Conjunction introduction repacks the three final-boundary row families and the two chained componentwise classifier families into one stable final boundary for the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate. □

Lean4 ✓ `BEDC.Derived.FieldUp.ratup fieldup selector chain final boundary`

Per-object template: traditional target selector-chain final boundary inside $\mathsf{Field}^{\uparrow}(\mathsf{RatUp})$; BEDC source BHist, BMark, RatUp, and Definition 45.617; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; selected endpoints $E_s(h,k,\ell)$, $E_t(h,k,\ell)$, $E_u(h,k,\ell)$, $E_v(h,k,\ell)$, $E_w(h,k,\ell)$, and $E_z(h,k,\ell)$; carrier rows the three final certificate boundaries plus the two chained componentwise denominator comparisons; classifier rows $\mathsf{RatHistoryClassifier}$ for the two chained selector comparisons; nonzero rows the contextual witnesses consumed by the three final certificate boundaries; stability obligations selector-independent final boundary and transitivity for both denominator and history classifiers; ledger selector pairs (s,t), (u,v), and (w,z) plus contexts L, R, r; exactness predicate one final FieldUp exit boundary closed under selector-chain replacement; downstream consumer stable selector-chain replacement for the Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.766 (RatUp FieldUp concrete exit selector-independent final boundary)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete exit object of Definition 45.617 contains the selector-independent final boundary row for the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*. Explicitly, for histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, if* $\mathsf{ThreeSupp}^u_D(h,k,\ell)$ *holds, then the concrete exit object contains both final-certificate two-selector noncollapse packages, the two componentwise* R^u_D *comparisons, and the two* $\mathsf{RatHistoryClassifier}$ *comparisons of Definition 45.764.*

Proof. Apply Definition 45.764 to the same concrete denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts L, R, r. This supplies the two final two-selector noncollapse packages and the four componentwise comparison rows listed in the selector-independent final boundary.

By Definition 45.617, those selector-independent final boundary rows are one conjunct of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. Therefore the rows supplied by Definition 45.764 are contained in the concrete exit object for $\mathsf{RatUpFieldSource}_F$. □

Theorem 45.767 (RatUp FieldUp concrete final obstruction boundary package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ *contains both the full assembled obstruction certificate of Definition 45.763 and the selector-independent final boundary row of Definition 45.766.*

Proof. First instantiate Definition 45.763 with the displayed denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts $L, R, r : \mathsf{BHist}$. This supplies the assembled selector obstruction rows: selector-pair strict exit, bracketing obstruction, and two-selector noncollapse.

Next instantiate Definition 45.766 with the same source, concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts L, R, r. This supplies the selector-independent final boundary row for the same concrete denominator tuple.

Finally cite Definition 45.618 only to keep both row families tied to the same $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$. Conjunction introduction then packages the assembled obstruction certificate with the selector-independent final boundary as one concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package. □

Per-object template: traditional target final FieldUp exit package; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations selector-pair strict exit, bracketing obstruction, two-selector noncollapse, and selector-independent final boundary; ledger selector quadruple s, t, u, v plus contexts L, R, r; core theorem Definition 45.767; exactness one concrete final exit package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.768 (RatUp FieldUp concrete final obstruction chain boundary package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, Definition 45.617, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, and contexts* $L, R, r : \mathsf{BHist}$*, let* $E_q(h, k, \ell)$ *be the bracketing endpoint selector of Definition 45.606 for each* $q : \mathsf{BMark}$*. If* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the concrete* $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$ *contains both the full assembled obstruction certificate of Definition 45.763 for* (s, t, u, v) *and the selector-chain final boundary rows of Definition 45.765 for* (s, t)*,* (u, v)*, and* (w, z)*. In particular, the same concrete exit object contains the chained denominator rows*

$$R^u_D(E_s(h, k, \ell), E_w(h, k, \ell)) \wedge R^u_D(E_t(h, k, \ell), E_z(h, k, \ell))$$

and the chained endpoint classifier rows

$$\mathsf{RatHistoryClassifier}(E_s(h, k, \ell), E_w(h, k, \ell)) \wedge \mathsf{RatHistoryClassifier}(E_t(h, k, \ell), E_z(h, k, \ell)).$$

Proof. First instantiate Definition 45.767 with the displayed source, concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts $L, R, r : \mathsf{BHist}$. This gives the concrete obstruction package containing the assembled obstruction certificate for (s, t, u, v) inside $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$.

Next instantiate Definition 45.765 with the same denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, w, z, and contexts L, R, r. This gives the three selector pairs (s, t), (u, v), and (w, z) together with the chained R^u_D endpoint comparisons and the chained $\mathsf{RatHistoryClassifier}$ endpoint comparisons from E_s, E_t to E_w, E_z.

Finally use Definition 45.617 to identify the selector-chain final-boundary family as an explicit row of the concrete exit object. Then cite Definition 45.618 to keep that row and the assembled

obstruction certificate inside the same $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$. Conjunction introduction packages the two families as one concrete final obstruction chain boundary package for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$. □

Per-object template: traditional target selector-chain obstruction package for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations concrete final obstruction package plus selector-chain final boundary; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; exactness one chain-stable final exit package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.769 (RatUp FieldUp two-step algebra hierarchy exit certificate)**.** *For* $\mathsf{RatUpFieldSource}_F$*, the concrete exit object* $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and the endpoint selector* $E_q(h, k, \ell)$ *of Definition 45.606, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the one concrete exit object contains every algebra-hierarchy exit row, every operation-table coverage row, both adjacent complete selector-obstruction certificates for* (s, t, u, v) *and* (u, v, w, z)*, and the selector-chain final boundary rows, including the chained* R^u_D *rows and the chained* $\mathsf{RatHistoryClassifier}$ *endpoint rows.*

Proof. First instantiate Definition 45.755 with the displayed source, denominator tuple, histories, selectors s, t, u, v, w, z, contexts L, R, r, endpoint selector, and $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This supplies the hierarchy rows, the operation-table coverage rows, the first selector-obstruction certificate, and the selector-chain boundary rows.

Next instantiate Definition 45.757 with the same data. This supplies both adjacent complete selector-obstruction certificates and the same selector-chain final boundary rows, including the chained denominator and Rat-history endpoint rows.

Finally cite Definition 45.618 to keep the hierarchy package and the two-step obstruction package inside the same concrete $\mathsf{RatUpFieldExitObject}_F(\mathcal{E}_F)$. Conjunction introduction repacks those row families as one Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate. □

Per-object template: traditional target two-step algebra hierarchy exit certificate for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier rows C^u_D and the concrete algebra hierarchy carriers; classifier rows R^u_D and $\mathsf{RatHistoryClassifier}$; denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$; operation table the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ table; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; stability obligations hierarchy exit, operation-table coverage, two adjacent selector obstructions, and selector-chain final boundary; exactness one two-step algebra hierarchy exit certificate; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.770 (RatUp FieldUp final row inventory semantic certificate)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete exit object of Definition 45.617. For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, the semantic certificate boundary for* $\mathsf{RatUpFieldSource}_F$ *contains, over the same concrete exit object, all of the following rows:*

1. *the paper certificate assembly of Definition 45.494;*
2. *the concrete schema row of Definition 45.502;*
3. *the proof-target exit package of Definition 45.584;*
4. *every row of Definition 45.773.*

Consequently the selector inverse-domain coverage, selector row transport, complete obstruction, and selector-chain boundary rows are all anchored over $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, $\mathsf{RatDenomUnitCarrier}$, $\mathsf{RatDenomUnitClassifier}$, append, e, Γ, *and* AffAut_F *inside the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *semantic certificate.*

Proof. Start with Definition 45.494. It fixes the paper-side semantic boundary for $\mathsf{RatUpFieldSource}_F$ over $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, the displayed denominator tuple, the contextual action Γ, and the affine source AffAut_F.

Next apply Definition 45.502. The schema row identifies the same concrete source tuple and the same concrete exit object $\mathcal{E}_F$ with the BHist rows used by the certificate boundary.

Then apply Definition 45.584. This places the concrete schema row inside the proof-target exit package for $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$, still over the same carrier, classifier, denominator tuple, action source, and affine source.

Finally apply Definition 45.773 to the displayed histories, carrier witnesses, support witness, selectors, and contexts. It supplies the selector inverse-domain, selector row transport, complete obstruction, and selector-chain boundary rows for $\mathcal{E}_F$.

Conjunction introduction packages the semantic assembly, schema row, proof-target exit package, and final row inventory as one consumed certificate boundary for $\mathsf{RatUpFieldSource}_F$. □

Per-object template: traditional target semantic consumption certificate for the final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ row inventory; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, and $\mathsf{RatDenomUnitCarrier}$; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, and $\mathsf{RatDenomUnitClassifier}$; operations append, e, Γ, and AffAut_F; stability obligations semantic name certificate, concrete schema row, proof-target exit package, selector coverage, selector transport, complete obstruction, and chain boundary; ledger selectors s, t, u, v, w, z plus contexts L, R, r; exactness final inventory rows inside the semantic certificate boundary; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.771 (RatUp FieldUp Rat-classified factors exclude empty product)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *candidate satisfying the hypotheses of Definition 45.248, specialize the nonzero index to*

$$\mathsf{ApartZero}_F(h) := (h\mathsf{hsame0} \to \bot).$$

Assume the RatUp concrete exit object identifies the field-zero endpoint with the empty history, and includes classifier transport for $\mathsf{ApartZero}_F$ *together with contradiction of* $\mathsf{ApartZero}_F(0)$*. Then for all histories* $a, a', b, b' : \mathsf{BHist}$*,*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b') \quad \Longrightarrow \quad (\mathsf{mul}(a, b)\mathsf{hsame0} \to \bot) \wedge (\mathsf{mul}(a', b')\mathsf{hsame0} \to \bot).$$

Proof. Unpack the two classifier witnesses. Apply Definition 37.51 to $\mathsf{RatHistoryClassifier}(a, a')$ and to $\mathsf{RatHistoryClassifier}(b, b')$. The four endpoint exclusions, transported across the concrete identification of the field-zero endpoint with e, give witnesses $p_a : \mathsf{ApartZero}_F(a)$, $p_{a'} : \mathsf{ApartZero}_F(a')$, $p_b : \mathsf{ApartZero}_F(b)$, and $p_{b'} : \mathsf{ApartZero}_F(b')$.

The stability field for $\mathsf{ApartZero}_F$ is transport along hsame: if $h\mathsf{hsame}k$ and $h\mathsf{hsame0} \to \bot$, then any $k\mathsf{hsame0}$ composes back to $h\mathsf{hsame0}$ and gives contradiction. The zero contradiction clause is immediate because $\mathsf{ApartZero}_F(0)$ applied to reflexivity of $0\mathsf{hsame0}$ yields $\bot$.

Instantiate Definition 45.248 first at (a, b) with witnesses p_a and p_b, obtaining $\mathsf{mul}(a, b)\mathsf{hsame0} \to \bot$. Instantiate the same theorem at (a', b') with witnesses $p_{a'}$ and $p_{b'}$, obtaining $\mathsf{mul}(a', b')\mathsf{hsame0} \to \bot$. Conjunction introduction packages the two Rat-classified product exclusions. □

Per-object template: traditional target Rat-classified nonzero product in $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, $\mathsf{RatHistoryClassifier}$, and the FieldUp apart-zero certificate; carrier RatHistoryClassifier endpoints; classifier hsame; stability Rat endpoint nonemptiness plus FieldUp zero-product exclusion; ledger the two classifier witnesses; exactness predicate no empty product for classified Rat endpoints; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.772 (Rat denominator carried continuation unit obstruction)**.** *For every history $u : \mathsf{BHist}$, a carried Rat denominator cannot itself be the continuation left unit for all carried denominators. Concretely,*

$$\mathsf{RatHistoryCarrier}(u) \to (\forall d, r : \mathsf{BHist}.\ \mathsf{RatHistoryCarrier}(d) \to \mathsf{Cont}(u, d, r) \to \mathsf{RatHistoryClassifier}(r, d)) \to \mathsf{False}.$$

Proof. Assume a carrier witness $\mathsf{RatHistoryCarrier}(u)$ and the displayed left-unit classifier law. Apply Definition 45.352 to the law; the forward implication yields $u\mathsf{hsamee}$.

Now apply Definition 37.18 to the same carrier witness and the endpoint comparison $u\mathsf{hsamee}$. This contradicts the nonempty Rat denominator condition, giving False. □

Per-object template: traditional target carried-unit obstruction for the Rat-denominator side of $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, Cont, and $\mathsf{RatHistoryCarrier}$; carrier $\mathsf{RatHistoryCarrier}$; classifier $\mathsf{RatHistoryClassifier}$; stability obligation left-unit endpoint exactness; ledger external unit versus carried denominator; core theorem Definition 45.772; exactness predicate no carried denominator is the continuation unit; downstream consumer the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.773 (RatUp FieldUp final row inventory coverage)**.** *For $\mathsf{RatUpFieldSource}_F$, let $\mathcal{E}_F$ be the concrete exit object of Definition 45.617. For the concrete denominator tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, histories $h, k, \ell : \mathsf{BHist}$ with $C^u_D(h)$, $C^u_D(k)$, and $C^u_D(\ell)$, selectors $s, t, u, v, w, z : \mathsf{BMark}$, and contexts $L, R, r : \mathsf{BHist}$, let $E_q(h, k, \ell)$ be the bracketing endpoint selector of Definition 45.606 for each $q : \mathsf{BMark}$. If $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ holds, then the final row inventory of $\mathcal{E}_F$ contains all of the following rows as one consumable Phase Algebra Hierarchy package:*

1. *the selector inverse-domain coverage row of Definition 45.619 for the selected pair (s, t) and contexts L, R;*
2. *for every primed denominator triple $h', k', \ell' : \mathsf{BHist}$ satisfying $C^u_D(h')$, $C^u_D(k')$, $C^u_D(\ell')$, $\mathsf{ThreeSupp}^u_D(h', k', \ell')$, $R^u_D(h, h')$, $R^u_D(k, k')$, and $R^u_D(\ell, \ell')$, and for every $L', R' : \mathsf{BHist}$ with $\mathsf{hsame}(L, L')$ and $\mathsf{hsame}(R, R')$, the selector row-stability row of Definition 45.620;*
3. *the complete selector exit obstruction certificate of Definition 45.763 for the selector quadruple (s, t, u, v) and contexts L, R, r;*
4. *the selector-chain final boundary rows of Definition 45.765 for the selector pairs (s, t), (u, v), and (w, z).*

Equivalently, the one concrete FieldUp exit object supplies selector inverse-domain coverage, selector row transport, complete obstruction, and chain boundary rows to the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate without changing the underlying BHist source, BMark selectors, or $\mathsf{RatUpFieldSource}_F$ package.

Proof. First instantiate Definition 45.619 at the displayed concrete denominator tuple, the histories h, k, ℓ, their carrier witnesses, the support witness, the selector pair (s, t), and the contexts L, R. This gives the strict inverse-domain selector row for $\mathcal{E}_F$.

Next take an arbitrary primed denominator triple h', k', ℓ' with the displayed carrier, support, and R^u_D comparison hypotheses, and take L', R' with $\mathsf{hsame}(L, L')$ and $\mathsf{hsame}(R, R')$. Apply Definition 45.620 to these data. The result is the selector row-stability transport row for the same concrete exit object.

We then instantiate Definition 45.763 with the same denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, and contexts L, R, r. This supplies the complete selector exit obstruction certificate inside the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ package.

Finally instantiate Definition 45.765 with the same denominator tuple, histories, carrier witnesses, support witness, selectors s, t, u, v, w, z, and contexts L, R, r. It supplies the three selector-pair final boundary row families and the chained endpoint classifier rows.

All four cited results refer to Definition 45.617 and to the concrete tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$. Conjunction introduction packages their row families as one final row inventory for the Phase Algebra Hierarchy consumer. □

Per-object template: traditional target final FieldUp row inventory; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations selector coverage, selector row transport, complete obstruction, concrete final obstruction package, and selector-chain final boundary; ledger selectors s, t, u, v, w, z plus contexts L, R, r; core theorem this block; exactness final row inventory coverage and one chain-stable final exit package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit certificate; status $\mathsf{ProofTarget}$.

Theorem 45.774 (RatUp FieldUp selector-independent two-step final inventory)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete exit object of Definition 45.617. For the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and the endpoint selector* $E_q(h, k, \ell)$ *of Definition 45.606, if* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ *holds, then the same concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object* $\mathcal{E}_F$ *simultaneously supplies all of the following rows as one concrete Phase Algebra Hierarchy exit package:*

1. *the operation-table lock of Definition 45.753;*
2. *the selector-independent algebra hierarchy exit of Definition 45.754;*
3. *the two-step algebra hierarchy exit certificate of Definition 45.769;*
4. *the final row inventory of Definition 45.773.*

Thus the operation table, selector-independent boundary, two-step obstruction, and final inventory rows are consumed through one displayed BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object, with no change of source, denominator tuple, selectors, contexts, or support witness.*

Proof. First instantiate Definition 45.753 at $\mathsf{RatUpFieldSource}_F$ and the concrete exit object $\mathcal{E}_F$. This supplies the completed exit-object row, the assembled operation-table coverage row, and the selector-independent final boundary row for the same BHist package.

Next instantiate Definition 45.754 with the displayed denominator tuple, histories h, k, ℓ, carrier witnesses, support witness, selectors s, t, u, v, and contexts L, R, r. This gives the selector-independent Phase Algebra Hierarchy exit rows over the same concrete source tuple.

Now instantiate Definition 45.769 with the same exit object, denominator tuple, histories, carrier witnesses, support witness, selector sextuple s, t, u, v, w, z, contexts L, R, r, and endpoint selector $E_q(h, k, \ell)$. This contributes both adjacent selector-obstruction certificates and the selector-chain final boundary rows.

Finally instantiate Definition 45.773 with the same concrete exit object, denominator tuple, histories, carrier witnesses, support witness, selectors, and contexts. Its rows provide selector inverse-domain coverage, selector row transport, complete obstruction, and chain-boundary inventory.

The four instantiations all name Definition 45.617 and the tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$. Conjunction introduction packages those row families as the single selector-independent, two-step, final inventory exit object claimed. □

Per-object template: traditional target final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit inventory; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations operation-table lock, selector-independent boundary, two-step obstruction, and final row inventory; ledger selector sextuple s, t, u, v, w, z plus contexts L, R, r; exactness one concrete Phase Algebra Hierarchy exit package; status $\mathsf{ProofTarget}$.

Theorem 45.775 (RatUp FieldUp four endpoint product nonempty)**.** *For a BEDC history-level* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *candidate satisfying the hypotheses of Definition 45.771, assume the same field-operation data, the same identification of the field-zero endpoint with* e*, the same transport for* $\mathsf{ApartZero}_F(h) := (h\mathsf{hsame}0 \to \bot)$*, and the same contradiction of* $\mathsf{ApartZero}_F(0)$*. Then for all histories* $a, a', b, b' : \mathsf{BHist}$*, if*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

then the same concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation package contains all four nonempty product rows:*

1. $\mathsf{mul}(a, b)\mathsf{hsamee} \to \bot$*;*
2. $\mathsf{mul}(a, b')\mathsf{hsamee} \to \bot$*;*
3. $\mathsf{mul}(a', b)\mathsf{hsamee} \to \bot$*;*
4. $\mathsf{mul}(a', b')\mathsf{hsamee} \to \bot$*.*

Proof. Apply Definition 45.771 to the displayed classifier pair. This gives the two diagonal exclusions for $\mathsf{mul}(a, b)$ and $\mathsf{mul}(a', b')$, transported through the fixed identification of the field-zero endpoint with e.

Next build the left mixed row. The first classifier supplies the endpoint carrier fields for a and a', while the second classifier supplies the carrier field for b. Apply Definition 37.37 to that carrier field to form the reflexive Rat classifier on the second endpoint pair (b, b). Applying the same two-diagonal product theorem to $\mathsf{RatHistoryClassifier}(a, a')$ and this reflexive classifier gives the product exclusion for $\mathsf{mul}(a', b)$.

Now build the right mixed row. Apply Definition 37.37 to the endpoint carrier field for a supplied by the first classifier, forming the reflexive Rat classifier on (a, a). Applying the same two-diagonal product theorem to this reflexive classifier and $\mathsf{RatHistoryClassifier}(b, b')$ gives the product exclusion for $\mathsf{mul}(a, b')$.

Conjunction introduction packages the four exclusions in the displayed order. All four applications use the same field-operation hypotheses, the same zero-empty identification, the same apart-zero transport, and the same BHist carrier and classifier data. □

Per-object template: traditional target four-endpoint Rat-classified nonempty product in $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$ and the endpoint carrier fields contained in $\mathsf{RatHistoryClassifier}$; classifiers $\mathsf{RatHistoryClassifier}$ and hsame; stability obligations Rat endpoint nonemptiness, reflexive endpoint classification, apart-zero transport, and FieldUp

zero-product exclusion; ledger the two classifier witnesses, the two reflexive classifiers, the four endpoint choices, and the zero-empty identification; core theorem this block; exactness predicate four products apart from e; downstream consumer Definition 45.776; status ProofTarget.

Theorem 45.776 (RatUp FieldUp operation-table final inventory nonzero coverage)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, u, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

and for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, one* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package contains, at the same time:*

1. *the assembled certificate operation-table coverage of Definition 45.534;*
2. *the concrete operation-table laws of Definition 45.588;*
3. *the final row inventory of Definition 45.773;*
4. *the diagonal Rat-classified product nonzero exclusions* $(\mathsf{mul}(a, b)\mathsf{hsame0} \to \bot)$ *and* $(\mathsf{mul}(a', b')\mathsf{hsame0} \to \bot)$ *from Definition 45.771;*
5. *the obstruction saying that* q *cannot be the continuation left unit for all carried denominators, as in Definition 45.772.*

Thus the Phase Algebra Hierarchy consumer can read the operation table, final inventory rows, Rat-classified product exclusions, and carried-denominator unit obstruction from one concrete BHist *package without changing source, selectors, contexts, or denominator tuple.*

Proof. First build the operation-table spine. Apply Definition 45.534 to the concrete source and table, and apply Definition 45.588 to the same table. These two rows name the same RatHistoryCarrier, RatHistoryClassifier, denominator coordinate, contextual action, context-pair monoid, affine coordinate, inverse rows, nonzero rows, and exit-noncollapse rows.

Next build the final inventory row family. Apply Definition 45.773 to the concrete exit object, denominator tuple, carrier witnesses for h, k, ℓ, the $\mathsf{ThreeSupp}^u_D(h, k, \ell)$ witness, selectors s, t, u, v, w, z, and contexts L, R, r. This gives the selector inverse-domain row, selector row-stability row, complete selector obstruction certificate, and chain-boundary rows inside $\mathcal{E}_F$.

Then build the diagonal product exclusions. Apply Definition 45.771 to the two displayed Rat-classified endpoint pairs. It supplies $(\mathsf{mul}(a, b)\mathsf{hsame0} \to \bot)$ and $(\mathsf{mul}(a', b')\mathsf{hsame0} \to \bot)$ as concrete diagonal endpoint rows for the same history-level $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ package.

Finally build the denominator-unit obstruction. Apply Definition 45.772 to the carrier witness $\mathsf{RatHistoryCarrier}(q)$. The resulting contradiction rule excludes a carried denominator from being a continuation left unit for all carried denominators.

All rows are sourced from $\mathsf{RatUpFieldSource}_F$, the operation table of Definition 45.587, and the exit object of Definition 45.617. Conjunction introduction packages the four row families into the single concrete BHist-anchored exit package claimed. □

Theorem 45.777 (RatUp FieldUp operation-table final inventory four-endpoint coverage)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, u, v, w, z :$

BMark *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

and for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, one concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *package contains, at the same time:*

1. *the assembled certificate operation-table coverage of Definition 45.534;*
2. *the concrete operation-table laws of Definition 45.588;*
3. *the final row inventory of Definition 45.773;*
4. *the carried-denominator unit obstruction of Definition 45.772;*
5. *the four Rat-classified product exclusions* $\mathsf{mul}(a, b)\mathsf{hsamee} \to \bot$*,* $\mathsf{mul}(a, b')\mathsf{hsamee} \to \bot$*,* $\mathsf{mul}(a', b)\mathsf{hsamee} \to \bot$*, and* $\mathsf{mul}(a', b')\mathsf{hsamee} \to \bot$*.*

Thus the Phase Algebra Hierarchy consumer reads the operation table, final inventory rows, carried-denominator unit obstruction, and all four Rat-classified product exclusions from one concrete BHist *package.*

Proof. First build the table, final-inventory, unit-obstruction, and diagonal product rows. Apply Definition 45.776 to the concrete source, operation table, exit object, denominator tuple, histories, selectors, contexts, Rat-classified endpoint pairs, and carried denominator q. This gives the assembled operation-table coverage, concrete operation-table laws, final row inventory, carried-denominator unit obstruction, and the diagonal Rat-classified product rows inside the same BHist-anchored package.

Next build the four endpoint product rows. Apply Definition 45.775 to the same field-operation data, zero-empty identification, apart-zero transport, zero-apart contradiction, and the two displayed $\mathsf{RatHistoryClassifier}$ hypotheses. This supplies the two diagonal rows and the two mixed rows for the same endpoint quadruple.

Finally keep the same source, table, exit object, denominator tuple, selectors, contexts, endpoint pairs, and carried denominator. Conjunction introduction packages the table rows, final inventory, unit obstruction, and four endpoint product exclusions as the concrete final-inventory $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ package claimed. □

Per-object template: traditional target final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package with four Rat-classified product rows; BEDC source BHist, BMark, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations operation-table coverage, final row inventory, carried-denominator unit obstruction, and four endpoint product nonemptiness; ledger selector sextuple s, t, u, v, w, z, contexts L, R, r, endpoint quadruple a, a', b, b', and carried denominator q; exactness one concrete final-inventory package with four products apart from e; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; status $\mathsf{ProofTarget}$.

Theorem 45.778 (RatUp FieldUp operation-table final inventory two-sided unit coverage)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, u, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

and for every carried denominator q : BHist *satisfying* RatHistoryCarrier(q)*, one concrete* BHist-*anchored* Field$^{\uparrow}$(Rat$^{\uparrow}$) *package contains, at the same time:*

1. *the assembled certificate operation-table coverage of Definition 45.534;*
2. *the concrete operation-table laws of Definition 45.588;*
3. *the final row inventory of Definition 45.773;*
4. *the four Rat-classified product exclusions* $\mathsf{mul}(a,b)\mathsf{hsamee} \to \bot$, $\mathsf{mul}(a,b')\mathsf{hsamee} \to \bot$, $\mathsf{mul}(a',b)\mathsf{hsamee} \to \bot$, *and* $\mathsf{mul}(a',b')\mathsf{hsamee} \to \bot$ *from Definition 45.777;*
5. *the append right-unit denial at* q,
$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x,q),x)) \to \mathsf{False};$$
6. *the append left-unit denial at* q,
$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(q,x),x)) \to \mathsf{False};$$
7. *the relational denial for* $\mathsf{Cont}(x,q,r)$,
$$(\forall x,r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x,q,r) \to R_D(r,x)) \to \mathsf{False};$$
8. *the relational denial for* $\mathsf{Cont}(q,x,r)$,
$$(\forall x,r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(q,x,r) \to R_D(r,x)) \to \mathsf{False}.$$

Here C_D, R_D, μ_D *are the strict-denominator presentation of Definition 45.323, identified there with* RatHistoryCarrier*,* RatHistoryClassifier*, and append. Thus the Phase Algebra Hierarchy consumer can read the operation table, final inventory rows, Rat-classified product exclusions, and both append and relational one-sided unit denials from one concrete* BHist *package without changing source, selectors, contexts, or denominator tuple.*

Proof. First build the operation-table, final-inventory, unit-obstruction, and four endpoint product rows. Apply Definition 45.777 to the concrete source, operation table, exit object, denominator tuple, histories, selectors, contexts, Rat-classified endpoint pairs, and carried denominator q. This gives the assembled operation-table coverage, the concrete operation-table laws, the final inventory, carried-denominator unit obstruction, and the four Rat-classified product exclusions inside the same BHist-anchored package.

Next build the strict denominator rows. Apply Definition 45.331 at the carried denominator q, using the identification of $C_D(q)$ with RatHistoryCarrier(q) in Definition 45.323. Its append presentation supplies the right-unit and left-unit denials, and its relational presentation supplies the two Cont endpoint denials.

Finally, keep the witnesses named by the same source, operation table, exit object, denominator tuple, selectors, contexts, endpoint pairs, and carried denominator. Conjunction introduction packages these row families as the single concrete final Field$^{\uparrow}$(Rat$^{\uparrow}$) exit package claimed. □

Per-object template: traditional target Field$^{\uparrow}$ final concrete exit package; BEDC source BHist, Cont, RatHistoryCarrier, and RatHistoryClassifier; carrier C_D^u; classifier R_D^u; stability obligations operation-table coverage plus two-sided unit obstruction; ledger selectors s,t,u,v,w,z and contexts L,R,r; exactness as the concrete final package; downstream consumer Phase Algebra Hierarchy Field$^{\uparrow}$(Rat$^{\uparrow}$); status ProofTarget.

Theorem 45.779 (RatUp FieldUp operation-table final inventory four-endpoint unit coverage)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, u, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

and for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, one concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *package contains, at the same time:*

1. *the assembled certificate operation-table coverage of Definition 45.534;*
2. *the concrete operation-table laws of Definition 45.588;*
3. *the final row inventory of Definition 45.773;*
4. *the four Rat-classified product exclusions* $(\mathsf{mul}(a, b)\mathsf{hsamee} \to \bot)$*,* $(\mathsf{mul}(a, b')\mathsf{hsamee} \to \bot)$*,* $(\mathsf{mul}(a', b)\mathsf{hsamee} \to \bot)$*, and* $(\mathsf{mul}(a', b')\mathsf{hsamee} \to \bot)$*;*
5. *the append right-unit denial at* q*,*
$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(x, q), x)) \to \mathsf{False};$$
6. *the append left-unit denial at* q*,*
$$(\forall x : \mathsf{BHist}.\ C_D(x) \to R_D(\mu_D(q, x), x)) \to \mathsf{False};$$
7. *the relational denial for* $\mathsf{Cont}(x, q, r)$*,*
$$(\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(x, q, r) \to R_D(r, x)) \to \mathsf{False};$$
8. *the relational denial for* $\mathsf{Cont}(q, x, r)$*,*
$$(\forall x, r : \mathsf{BHist}.\ C_D(x) \to \mathsf{Cont}(q, x, r) \to R_D(r, x)) \to \mathsf{False}.$$

Here C_D, R_D, μ_D *are the strict-denominator presentation of Definition 45.323, identified there with* $\mathsf{RatHistoryCarrier}$*,* $\mathsf{RatHistoryClassifier}$*, and append. Thus the Phase Algebra Hierarchy consumer can read the operation table, final inventory rows, all four Rat-classified product exclusions, and both append and relational one-sided unit denials from one concrete* BHist *package without changing source, selectors, contexts, or denominator tuple.*

Proof. First apply Definition 45.778 to the concrete source, operation table, exit object, denominator tuple, histories, support witness, selectors, contexts, Rat-classified endpoint pairs, and carried denominator q. This supplies the operation-table rows, final inventory row, diagonal product exclusions, and the four append and relational one-sided unit denials inside the same BHist-anchored package.

Next apply Definition 45.775 to the same two Rat classifier witnesses. Its four endpoint rows give the two diagonal product exclusions already present in the two-sided package and add the mixed rows for $\mathsf{mul}(a, b')$ and $\mathsf{mul}(a', b)$.

Finally keep the same source, operation table, exit object, denominator tuple, selectors, contexts, endpoint pairs, and carried denominator. Conjunction introduction packages the operation-table rows, final inventory, all four product exclusions, and the four one-sided unit denials into the concrete final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package claimed. □

Per-object template: traditional target $\mathsf{Field}^{\uparrow}$ final concrete four-endpoint exit package; BEDC source BHist, Cont, $\mathsf{RatHistoryCarrier}$, and $\mathsf{RatHistoryClassifier}$; carrier C^u_D; classifier R^u_D; stability obligations operation-table coverage, four-endpoint product nonemptiness, and two-sided unit obstruction; ledger selectors s, t, u, v, w, z, contexts L, R, r, Rat endpoint pairs, and carried denominator q; exactness as the concrete final four-endpoint package; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; status $\mathsf{ProofTarget}$.

Theorem 45.780 (RatUp FieldUp final inventory carried-external unit separation)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, v, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(a, a') \wedge \mathsf{RatHistoryClassifier}(b, b'),$$

for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, and for histories* $u, d : \mathsf{BHist}$ *satisfying* $U_D(u)$ *and* $\mathsf{RatHistoryCarrier}(d)$*, one concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *final-inventory boundary contains, at the same time:*

1. *the operation-table final inventory rows and two-sided unit coverage rows of Definition 45.778 for the displayed source, table, exit object, tuple, histories, selectors, contexts, endpoint pairs, and carried denominator* q*;*
2. *the carried-external classifier separation row*

$$\big(\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)\big) \to \mathsf{False}$$

supplied by Definition 45.366 from $U_D(u)$*.*

Thus the same concrete final exit object $\mathcal{E}_F$ *records both the complete operation-table final inventory consumed by the Phase Algebra Hierarchy and the classifier separation between an external denominator unit and a carried Rat denominator.*

Proof. First apply Definition 45.778 to the concrete source, operation table, exit object, denominator tuple, histories h, k, ℓ, support witness, selectors s, t, v, v, w, z, contexts L, R, r, Rat-classified endpoint pairs, and the carried denominator q. This supplies the operation-table coverage rows, concrete operation-table laws, final row inventory, four Rat-classified product exclusions, and both append and relational one-sided unit denials inside $\mathcal{E}_F$.

Next apply Definition 45.366 to the histories u, d and the displayed witness $U_D(u)$. The resulting implication rules out both possible Rat-classifier orientations $\mathsf{RatHistoryClassifier}(u, d)$ and $\mathsf{RatHistoryClassifier}(d, u)$. The carrier witness $\mathsf{RatHistoryCarrier}(d)$ identifies the second endpoint as one of the carried denominator objects governed by the final-inventory boundary.

Finally keep the same source, table, exit object, denominator tuple, selectors, contexts, endpoint pairs, carried denominator, external-unit witness, and carried-denominator witness. Conjunction introduction packages the two-sided final inventory rows and the carried-external classifier separation row as one concrete BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ boundary. □

Per-object template: traditional target final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ carried-external unit boundary; BEDC source BHist, BMark, Cont, and $\mathsf{RatUpFieldSource}_F$; carrier $\mathsf{RatHistoryCarrier}$ and C^u_D; classifier $\mathsf{RatHistoryClassifier}$ and R^u_D; stability obligations operation-table final inventory, two-sided unit obstruction, and carried-external classifier separation; ledger selectors s, t, v, v, w, z, contexts L, R, r,

carried denominator q, external unit u, and carried denominator endpoint d; core theorem list Definition 45.778 and Definition 45.366; exactness predicate final inventory plus external-unit classifier disjointness; standard bridge Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$; status $\mathsf{ProofTarget}$.

Theorem 45.781 (RatUp FieldUp final inventory phase-exit boundary)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, v, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(x, x') \wedge \mathsf{RatHistoryClassifier}(y, y'),$$

for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, fix apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, guard histories* $\alpha, \beta : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : ((\alpha + \beta)\mathsf{hsame0} \vee (\alpha + (-\beta))\mathsf{hsame0}),$$

and histories $u, d : \mathsf{BHist}$ *satisfy* $U_D(u)$ *and* $\mathsf{RatHistoryCarrier}(d)$*, then the same exit object* $\mathcal{E}_F$ *supplies one concrete* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *boundary containing, at the same time:*

1. *the phase-exit record certificate boundary of Definition 45.792 for the displayed source, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, carried denominator, apart-zero data, contextual action, affine source, operations, guard hypothesis, and external-unit witness;*
2. *the operation-table final inventory rows and two-sided unit coverage of Definition 45.778 for the displayed source, table, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, and carried denominator;*
3. *the carried-external classifier separation row*

$$(\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)) \to \mathsf{False}$$

   of Definition 45.780 for the same external unit, carried endpoint, unit witness, and carried denominator endpoint.

Thus the traditional final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit boundary reads the phase record certificate, the final-inventory unit coverage, and the carried-external classifier separation from one unchanged concrete exit package.*

Proof. First invoke Definition 45.792 with the displayed guard hypothesis H and the external-unit witness $U_D(u)$. Its certificate rows are read over the same source, exit object, denominator tuple, selectors, contexts, Rat endpoints, carried denominator, contextual action, affine source, and CommRing operations.

Next invoke Definition 45.780 with the same concrete source, operation table, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, carried denominator q, external unit u, carried endpoint d, witness $U_D(u)$, and carrier witness $\mathsf{RatHistoryCarrier}(d)$. That theorem already imports the operation-table final inventory and two-sided unit coverage of Definition 45.778.

Finally conjoin the two packages. The source, table, exit object, denominator tuple, selectors, contexts, endpoint pairs, carried denominator, external-unit witness, carried endpoint, and guard datum are unchanged, so the conclusion is one concrete BHist-anchored phase-exit boundary rather than two separately indexed row families. □

Per-object template: traditional target final $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit boundary; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$ and denominator carrier C^u_D; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, and hsame; stability obligations phase-exit record plus final-inventory unit separation; ledger source, operation table, exit object, denominator tuple, histories, selectors, contexts, Rat-classified endpoint pairs, carried denominator q, external unit u, carried endpoint d, guard hypothesis H, and witness $U_D(u)$; exactness one concrete boundary; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.782 (RatUp FieldUp chapter certificate row coverage)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. The chapter-level* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate exposes over one concrete* BHist *source the following row package:*

1. *the carrier, classifier, denominator, and contextual-action rows of Definition 45.502;*
2. *the operation-table coverage of Definition 45.615;*
3. *the completed exit object of Definition 45.618;*
4. *the final-row semantic certificate of Definition 45.770;*
5. *the nonzero and unit-obstruction rows of Definition 45.776.*

Thus a downstream Phase Algebra Hierarchy consumer can read $\mathcal{E}_F$ *as a single concrete row-coverage certificate for the canonical* BHist *source.*

Proof. First take the concrete schema instance of Definition 45.502. It fixes the shared BHist source together with the carrier, classifier, denominator tuple, and contextual action rows used by the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ construction.

Next combine the operation-table coverage from Definition 45.615 with the completed exit object of Definition 45.618. These two cited certificates identify the same selectors and the same concrete object $\mathcal{E}_F$ rather than introducing a second exit target.

Finally append the final-row semantic certificate from Definition 45.770 and the nonzero and unit-obstruction rows from Definition 45.776. Conjunction introduction packages these cited rows into one chapter-level certificate.

The consumer never changes the carrier, classifier, denominator tuple, contextual action, selectors, or exit object. It only reads the rows already provided by the cited concrete BHist certificates. □

Theorem 45.783 (RatUp FieldUp phase-exit certificate consumption)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix apart-zero context data* $L_0, R_0, x : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*.*

Then the same concrete object $\mathcal{E}_F$ *supplies, as one consumed phase-exit certificate boundary, all of the following rows:*

1. *the canonical concrete phase-exit target of Definition 45.651;*
2. *the final semantic row inventory of Definition 45.770;*
3. *the contextual apart-zero semantic certificate row of Definition 45.658 at* L_0, R_0, x, p*.*

All three rows use the same BHist *carrier universe, the same* $\mathsf{RatHistoryCarrier}$ *and* $\mathsf{RatHistoryClassifier}$ *anchors, the same denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *where denominator data are required, and the same contextual action* Γ*.*

Proof. First instantiate Definition 45.651 at $\mathsf{RatUpFieldSource}_F$ and Definition 45.617. This fixes $\mathcal{E}_F$ as the canonical concrete phase-exit target over BHist, $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, the displayed denominator object, Γ, and the $\mathsf{Field}^{\uparrow}$ exit rows.

Next instantiate Definition 45.770 with the tuple $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$, the histories h, k, ℓ, the three carrier witnesses, the selectors s, t, u, v, w, z, the contexts L, R, r, and the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This contributes the final semantic row inventory for the same exit object $\mathcal{E}_F$.

For the apart-zero row, apply Definition 45.658 to L_0, R_0, x and $p : \mathsf{FieldApartZero}(L_0)$. Its displayed source, pattern, ledger policy, and classifier are all built from the same BHist carrier and hsame classifier discipline.

Conjunction introduction packages the canonical target, the final inventory, and the contextual apart-zero row as one consumed boundary for the Phase Algebra Hierarchy exit criterion. □

Per-object template: traditional target constructive rational field phase-exit boundary; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, and the contextual apart-zero predicate; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, and hsame; stability obligations the canonical phase-exit target, final semantic row inventory, contextual apart-zero row; ledger selectors, denominator witnesses, contexts, and apart-zero witness; exactness one consumed concrete phase-exit boundary; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.784 (RatUp FieldUp full algebra hierarchy exit package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $a, a', b, b' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(a, a')$ *and* $\mathsf{RatHistoryClassifier}(b, b')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, x : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*.*

Then the same concrete BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package* $\mathcal{E}_F$ *simultaneously supplies all rows required by the Phase Algebra Hierarchy exit boundary:*

1. *the three phase-exit rows of Definition 45.783;*
2. *the operation-table, final-inventory, four-endpoint Rat-classified nonzero-product, and one-sided unit obstruction rows of Definition 45.779;*
3. *the selector-independent two-step final inventory of Definition 45.774, using the endpoint selector* $E_q(h, k, \ell)$ *of Definition 45.606.*

All three row families are read over the same source $\mathsf{RatUpFieldSource}_F$*, the same exit object* $\mathcal{E}_F$*, the same denominator tuple, histories, carrier witnesses, selectors, contexts, Rat-classified endpoints, carried denominator, and apart-zero context row.*

Proof. First instantiate Definition 45.783 with the displayed denominator tuple, histories, carrier witnesses, selectors, contexts, support witness, and apart-zero data L_0, R_0, x, p. This supplies the canonical phase-exit target, final semantic row inventory, and contextual apart-zero row for $\mathcal{E}_F$.

Next instantiate Definition 45.779 with the same source, operation table, exit object, denominator tuple, histories, support witness, selectors, contexts, Rat-classified endpoint pairs, and carried denominator q. It contributes the operation-table rows, final inventory rows, four-endpoint nonzero-product exclusions, and one-sided unit obstruction inside the same BHist package.

Finally instantiate Definition 45.774 with the same denominator tuple, histories, carrier witnesses, support witness, selectors, contexts, and endpoint selector $E_q(h, k, \ell)$. This contributes the operation-table lock, selector-independent algebra hierarchy exit, two-step certificate, and final row inventory without changing the exit object.

Conjunction introduction packages these three instantiated row families as one concrete certificate-level object. The Phase Algebra Hierarchy consumer therefore reads a single $\mathcal{E}_F$ boundary instead of reconstructing the exit from separate row inventories. □

Per-object template: traditional target complete rational field algebra hierarchy exit boundary; BEDC source BHist, BMark, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$ and C^u_D; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, and hsame; stability obligations phase-exit consumption, operation-table final-inventory four-endpoint unit coverage, and selector-independent two-step final inventory; ledger selectors s, t, u, v, w, z, contexts L, R, r, endpoint pairs, carried denominator, and apart-zero witness; exactness one concrete BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit package; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.785 (RatUp FieldUp bilateral zero-factor and external-unit guard package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix histories* $a, b, u, d : \mathsf{BHist}$*, the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((a + b)\mathsf{hsame}0 \vee (a + (-b))\mathsf{hsame}0\big),$$

then the same exit object $\mathcal{E}_F$ *exposes the following two concrete guard rows as part of its phase-exit boundary:*

1. *the four-component bilateral zero-factor conclusion of Definition 44.85 for* a, b*;*
2. *the external-unit classifier separation row* $U_D(u) \to (\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)) \to \mathsf{False}$ *from Definition 45.366.*

Both rows are recorded over the shared BHist *universe, the shared* $\mathsf{RatHistoryCarrier}$ *and* $\mathsf{RatHistoryClassifier}$ *anchors, the displayed denominator tuple, the contextual action* Γ*, and the affine source* AffAut_F*.*

Proof. First apply Definition 45.651 together with Definition 45.617. These directly fix the shared concrete exit object $\mathcal{E}_F$ together with its carrier, classifier, denominator tuple, contextual action, and affine source already named in the theorem statement.

Next apply Definition 44.85 to the displayed hypothesis H. Its result is exactly the branch annihilator package, the two signed-product orientations, and the square equality $a \cdot a\mathsf{hsame}b \cdot b$.

For the denominator side, apply Definition 45.366 to the histories u, d. This produces the concrete implication from $U_D(u)$ and either classifier endpoint orientation to False.

Finally package these two outputs as guard rows of the already fixed $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit object. The first row supplies the CommRing bilateral zero-factor exactness ledger, and the second row supplies the Rat denominator external-unit classifier separation guard. □

Per-object template: traditional target constructive rational field guard package; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$ and U_D; classifiers $\mathsf{RatHistoryClassifier}$ and hsame; stability obligations bilateral zero-factor exactness and external-unit classifier separation; ledger selected linear factor, signed-product orientations, square equality, external-unit law, and classifier endpoint; core theorem this block; exactness predicate the paired guard rows; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.786 (RatUp FieldUp guarded full algebra hierarchy exit package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*,*

selectors $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, t : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((a + b)\mathsf{hsame0} \vee (a + (-b))\mathsf{hsame0}\big),$$

then the one concrete BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package* $\mathcal{E}_F$ *simultaneously supplies:*

1. *the full algebra hierarchy exit rows of Definition 45.784;*
2. *the bilateral zero-factor guard row and external-unit classifier separation guard row of Definition 45.785.*

The two components are read over the same concrete source $\mathsf{RatUpFieldSource}_F$*, the same exit object* $\mathcal{E}_F$*, the same denominator tuple, contextual action, affine source, histories, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator, apart-zero context, and guard data.*

Proof. First instantiate Definition 45.784 with the displayed source, exit object, denominator tuple, histories h, k, ℓ, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator q, and apart-zero data L_0, R_0, t, p. This yields the full algebra hierarchy exit rows over the fixed BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ package $\mathcal{E}_F$.

Next instantiate Definition 45.785 with the shared exit object $\mathcal{E}_F$, the same denominator tuple, contextual action Γ, affine source AffAut_F, BEDC history operations, histories a, b, u, d, and zero-linear-factor witness H. This yields the bilateral zero-factor guard row and the external-unit classifier separation guard row without changing the concrete exit object.

Finally introduce the conjunction of these two instantiated outputs. The Phase Algebra Hierarchy exit criterion therefore consumes one guarded $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ boundary whose full-exit rows and guard rows are anchored by the same concrete BHist object. □

Per-object template: traditional target guarded rational field phase-exit package; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, and U_D; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, and hsame; stability obligations full algebra hierarchy exit rows, bilateral zero-factor guard row, and external-unit classifier separation guard row; ledger denominator tuple, histories, carrier witnesses, selectors, contexts, support witness, endpoint pairs, carried denominator, apart-zero context, contextual action, zero-linear-factor witness, and external-unit endpoint; exactness the combined exit-plus-guard boundary; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.787 (RatUp FieldUp guarded canonical action and inverse exit package)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history*

operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If

$$H : \big((a+b)\mathsf{hsame0} \vee (a+(-b))\mathsf{hsame0}\big),$$

then the one concrete BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit package* $\mathcal{E}_F$ *simultaneously supplies:*

1. *the guarded full algebra hierarchy rows of Definition 45.786;*
2. *the canonical action-and-inverse rows of Definition 45.662.*

The two components are read over the same concrete source $\mathsf{RatUpFieldSource}_F$*, the same exit object* $\mathcal{E}_F$*, the same denominator tuple, contextual action, affine source, histories, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator, apart-zero context, and guard datum.*

Proof. First instantiate Definition 45.786 with the displayed source, exit object, denominator tuple, histories h, k, ℓ, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator q, apart-zero data L_0, R_0, τ, p, contextual action Γ, affine source, BEDC history operations, histories a, b, u, d, and guard hypothesis H. This yields the guarded full algebra hierarchy rows on the fixed concrete package $\mathcal{E}_F$.

Next instantiate Definition 45.662 for the same exit object $\mathcal{E}_F$. Its reduct, inverse-product classifier exactness, and affine action rows attach to the identical carrier, classifier, denominator tuple, contextual action, and affine source already selected above.

Finally introduce the conjunction of the two instantiated outputs. No carrier, classifier, denominator tuple, contextual action, affine source, or guard datum changes, so the Phase Algebra Hierarchy exit criterion may cite one concrete BHist-anchored $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ boundary for both the guard rows and the canonical action-and-inverse support. □

Per-object template: traditional target guarded rational field exit with action and inverse support; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers RatHistoryCarrier, C^u_D, U_D, and AffAut_F; classifiers RatHistoryClassifier, R^u_D, hsame, and $\equiv_{\mathrm{aff}}$; stability obligations guarded full algebra hierarchy exit rows, algebra-reduct compatibility, inverse-product classifier exactness, affine group laws, affine fiber singleton exactness, and affine cancellation exactness; ledger denominator tuple, guard witness, inverse-product witnesses, affine coefficients, offsets, inverse witnesses, contextual action, affine source, and carrier support; exactness the combined guarded action-inverse exit boundary; downstream consumer Phase Algebra Hierarchy exit criterion; status ProofTarget.

Definition 45.788 (RatUp FieldUp phase-exit record)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. Also fix a guard hypothesis*

$$H : \big((a+b)\mathsf{hsame0} \vee (a+(-b))\mathsf{hsame0}\big).$$

The record predicate $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ *means that the fixed source, exit object, denominator tuple, contextual action, affine source, histories, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator, apart-zero guard data, guard hypothesis, and inverse/action rows jointly satisfy the following four components:*

1. *the chapter row coverage of Definition 45.782;*
2. *the full algebra hierarchy exit package of Definition 45.784;*
3. *the guarded full package of Definition 45.786;*
4. *the guarded canonical action-inverse package of Definition 45.787.*

Thus the predicate is a named conjunction over the single BHist *source, the carrier* $\mathsf{RatHistoryCarrier}$*, the denominator carrier* C^u_D*, the classifiers* $\mathsf{RatHistoryClassifier}$*,* R^u_D*, and* hsame*, the apart-zero guard, and the affine action/inverse support.*

Theorem 45.789 (RatUp FieldUp canonical phase-exit record)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((a + b)\mathsf{hsame0} \vee (a + (-b))\mathsf{hsame0}\big),$$

then the concrete exit object $\mathcal{E}_F$ *satisfies* $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ *for the displayed data and guard hypothesis.*

Proof. Unfold the record predicate of Definition 45.788. The first component is the chapter row coverage supplied by Definition 45.782 for the fixed $\mathsf{RatUpFieldSource}_F$ and exit object $\mathcal{E}_F$.

The second component is obtained by instantiating Definition 45.784 with the displayed denominator tuple, histories h, k, ℓ, carrier witnesses, selectors, contexts, support witness, Rat-classified endpoint pairs, carried denominator q, and apart-zero data L_0, R_0, τ, p.

The third component is obtained by applying Definition 45.786 to the same source, exit object, denominator tuple, affine source, contextual action, CommRing operations, histories a, b, u, d, and guard hypothesis H.

The fourth component is exactly Definition 45.787 instantiated with the same displayed data. Conjunction introduction packages the four components into the named phase-exit record, and no carrier, classifier, selector, denominator tuple, contextual action, affine source, apart-zero datum, or inverse/action row is replaced. □

Theorem 45.790 (RatUp FieldUp guarded action-inverse external-unit separation)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((a + b)\mathsf{hsame0} \vee (a + (-b))\mathsf{hsame0}\big)$$

and $U_D(u)$*, then the single exit object* $\mathcal{E}_F$ *supplies the following three-row phase-exit boundary:*

1. *the square row* $a \cdot a\mathsf{hsame}b \cdot b$*;*
2. *the external-unit classifier separation* $(\mathsf{RatHistoryClassifier}(u,d) \vee \mathsf{RatHistoryClassifier}(d,u)) \to \mathsf{False}$*;*
3. *the canonical action-and-inverse rows of Definition 45.662 read over the same* $\mathcal{E}_F$*.*

All three rows use the displayed denominator tuple, contexts, selectors, Rat-classified endpoint pairs, apart-zero context data, histories, contextual action, affine source, and CommRing operations.

Proof. First apply Definition 44.85 to the hypothesis H. Projecting its square component gives the row $a \cdot a\mathsf{hsame}b \cdot b$.

Next specialize Definition 45.366 at the histories u, d. Combining that specialization with the displayed external-unit witness $U_D(u)$ gives the second row, namely that either Rat-classifier endpoint orientation from u to d is impossible.

Finally invoke Definition 45.662 for the already fixed exit object $\mathcal{E}_F$. Its action rows and inverse rows use the same carrier, classifier, denominator tuple, contextual action, and affine source, so the three rows form one consumable phase-exit boundary rather than three unrelated certificates. □

Per-object template: traditional target constructive rational field exit with action/inverse and external-unit separation; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, U_D, and AffAut_F; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, hsame, and affine classifier; stability obligations zero-factor square row, external-unit classifier separation, and action/inverse support; ledger zero-linear-factor witness, external-unit endpoint, classifier orientation, denominator tuple, contexts, selectors, affine data; core theorem this block; exactness the displayed three-row package; standard bridge constructive rational field guard; status $\mathsf{ProofTarget}$.

Theorem 45.791 (RatUp FieldUp phase-exit record guarded square projection)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If* $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ *holds for the displayed data and guard hypothesis*

$$H : ((a + b)\mathsf{hsame}0 \vee (a + (-b))\mathsf{hsame}0),$$

and if $U_D(u)$*, then the record exposes the following three guard rows:*

1. *the square row* $a \cdot a\mathsf{hsame}b \cdot b$*;*
2. *the external-unit classifier separation* $(\mathsf{RatHistoryClassifier}(u,d) \vee \mathsf{RatHistoryClassifier}(d,u)) \to \mathsf{False}$*;*
3. *the canonical action-and-inverse rows of Definition 45.662 over the same* $\mathcal{E}_F$*.*

All three rows keep the same carrier, classifier, denominator tuple, contextual action, affine source, guard datum, histories, contexts, and selectors fixed by Definition 45.788.

Proof. Unfold the record predicate of Definition 45.788. The assumed $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ contains the guarded full package component and the guarded canonical action-inverse component

for the displayed source, exit object, denominator tuple, contextual action, affine source, and guard datum.

Project the guarded full-package component under the displayed hypothesis H. Its zero-linear-factor square component is the first row $a \cdot a\mathsf{hsame}b \cdot b$, the same row exposed in Definition 45.790.

Specialize the same guarded full-package component with the witness $U_D(u)$. Equivalently, read the external-unit row through Definition 45.790; the carrier $\mathsf{RatHistoryCarrier}$, denominator carrier C^u_D, unit carrier U_D, and Rat classifier $\mathsf{RatHistoryClassifier}$ are exactly the ones recorded in the phase-exit data.

Project the guarded canonical action-inverse component next. It is the same canonical action-and-inverse support cited by Definition 45.662, with the same contextual action Γ, affine source AffAut_F, affine classifier, denominator relation R^u_D, and history relation hsame.

Pair these three projections. No carrier, classifier, denominator tuple, contextual action, affine source, or guard datum is replaced, so the resulting three-row package is a direct record-level guard projection for the Phase Algebra Hierarchy exit criterion. □

Per-object template: traditional target rational field phase-exit guard projection; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, U_D, and AffAut_F; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, hsame, and affine classifier; stability obligations zero-factor square row, phase-exit record projection, external-unit classifier separation, and canonical action/inverse support; ledger denominator tuple, external-unit endpoint, guard witness, contextual action, affine source, histories, contexts, selectors, and carrier witnesses; core theorem this block; exactness the displayed three-row record-level guard projection; downstream consumer Phase Algebra Hierarchy exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.792 (RatUp FieldUp phase-exit record certificate boundary)**.** *For* $\mathsf{RatUpFieldSource}_F$*, let* $\mathcal{E}_F$ *be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object of Definition 45.617. Fix the concrete denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*, histories* $h, k, \ell : \mathsf{BHist}$ *with carrier witnesses* $C^u_D(h)$*,* $C^u_D(k)$*, and* $C^u_D(\ell)$*, selectors* $s, t, u_0, v, w, z : \mathsf{BMark}$*, contexts* $L, R, r : \mathsf{BHist}$*, and a support witness* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*. Also fix Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryClassifier}(x, x')$ *and* $\mathsf{RatHistoryClassifier}(y, y')$*, a carried denominator* $q : \mathsf{BHist}$ *with* $\mathsf{RatHistoryCarrier}(q)$*, apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, histories* $a, b, u, d : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((a + b)\mathsf{hsame}0 \vee (a + (-b))\mathsf{hsame}0\big)$$

and $U_D(u)$*, then the same exit object* $\mathcal{E}_F$ *supplies one* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *phase algebra exit package containing:*

1. $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ *for the displayed data and guard hypothesis;*
2. *the source, certificate-obligation, lower-reduct, denominator-law, and raw-continuation separation rows of Definition 45.725;*
3. *the square row, external-unit classifier separation, and canonical action-and-inverse rows exposed by Definition 45.791.*

Every row is read over the displayed BHist *source,* $\mathsf{RatHistoryCarrier}$*,* $\mathsf{RatHistoryClassifier}$*, denominator tuple, contextual action, affine source, guard datum, histories, contexts, selectors, and carrier witnesses.*

Proof. First apply Definition 45.789. It gives $\mathsf{RatUpFieldPhaseExit}(\mathcal{E}_F)$ from exactly the displayed source, exit object, denominator tuple, endpoint data, apart-zero context, affine data, CommRing operations, and guard hypothesis H.

Next invoke Definition 45.725. Its four rows record the same $\mathsf{RatUpFieldSource}_F$ source, the concrete Definition 45.337 certificate, the lower algebra reduct interface, the denominator append and relational packages, and the raw-continuation separation.

Finally feed the phase-exit record just obtained, together with $U_D(u)$, into Definition 45.791. The result is the square row, the external-unit classifier separation, and the canonical action-and-inverse support over the same carrier, classifier, denominator tuple, contextual action, affine source, histories, contexts, and selectors.

Conjunction introduction packs these three sources into the public $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ phase algebra exit package. No source row, certificate row, lower-reduct row, denominator row, guard row, action row, or inverse row is reindexed. □

Per-object template: traditional target rational field phase algebra exit package; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, U_D, and AffAut_F; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, hsame, and affine classifier; stability obligations canonical phase-exit record, certificate boundary rows, zero-factor square row, external-unit classifier separation, and action/inverse support; ledger denominator tuple, external-unit endpoint, guard witness, contextual action, affine source, histories, contexts, selectors, and carrier witnesses; core theorem this block; exactness the displayed phase algebra exit package; downstream consumer $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.793 (RatUp FieldUp phase-exit record final-inventory separation)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, v, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the chosen selectors and contexts. For Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(x, x') \wedge \mathsf{RatHistoryClassifier}(y, y'),$$

for every carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, fix apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, phase guard histories* $\alpha, \beta : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : ((\alpha + \beta)\mathsf{hsame0} \vee (\alpha + (-\beta))\mathsf{hsame0}),$$

and histories $u, d : \mathsf{BHist}$ *satisfy* $U_D(u)$ *and* $\mathsf{RatHistoryCarrier}(d)$*, then the same exit object* $\mathcal{E}_F$ *supplies one* BHist*-anchored* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *phase-exit boundary containing, at the same time:*

1. *the phase-exit record certificate boundary of Definition 45.792 for the displayed source, operation table, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, carried denominator, apart-zero data, contextual action, affine source, operations, guard hypothesis, and external-unit witness;*
2. *the final-inventory carried-external classifier separation row*

   $$(\mathsf{RatHistoryClassifier}(u, d) \vee \mathsf{RatHistoryClassifier}(d, u)) \to \mathsf{False}$$

   of Definition 45.780 for the same concrete source, operation table, exit object, denominator tuple, histories, support witness, selectors, contexts, endpoint pairs, carried denominator, external-unit witness, and carried denominator endpoint.

No carrier, classifier, denominator tuple, contextual action, affine source, selector, context, Rat endpoint, carried denominator, external-unit witness, or carried denominator endpoint is replaced between the two row families.

Proof. First instantiate Definition 45.792 with the displayed source, exit object, denominator tuple, histories h, k, ℓ, support witness, selectors s, t, υ, v, w, z, contexts L, R, r, Rat-classified endpoint pairs x, x', y, y', carried denominator q, apart-zero data L_0, R_0, τ, p, phase guard histories α, β, u, d, contextual action Γ, affine source AffAut_F, CommRing operations, guard hypothesis H, and witness $U_D(u)$. This yields the phase-exit record certificate boundary over $\mathcal{E}_F$.

Next instantiate Definition 45.780 with the same concrete source, operation table, exit object, denominator tuple, histories h, k, ℓ, support witness, selectors, contexts, Rat-classified endpoint pairs, carried denominator q, external unit u, and carried denominator endpoint d. The hypotheses $U_D(u)$ and $\mathsf{RatHistoryCarrier}(d)$ supply exactly the carried-external separation row needed by the phase-exit boundary.

Finally pair the two row families. Both cited results were read at the identical source, carrier, classifier, denominator tuple, contextual action, affine source, histories, selectors, contexts, endpoint pairs, carried denominator, external-unit witness, and carried denominator endpoint, so the packaged boundary is a single concrete BHist-anchored certificate for the Phase Algebra Hierarchy exit criterion. □

Per-object template: traditional target rational field phase-exit final-inventory separation; BEDC source BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{Field}^{\uparrow}$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, U_D, and AffAut_F; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, hsame, and affine classifier; stability obligations phase-exit record certificate boundary and final-inventory carried-external classifier separation; ledger denominator tuple, operation table, external-unit endpoint, carried denominator endpoint, guard witness, contextual action, affine source, histories, contexts, selectors, and carrier witnesses; core theorem this block; exactness the displayed paired phase-exit and final-inventory boundary; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.794 (RatUp FieldUp certificate record absorbs phase-exit boundary)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, and the concrete exit object* $\mathcal{E}_F$ *of Definition 45.617. Let* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$ *be the concrete denominator tuple, let* $h, k, \ell : \mathsf{BHist}$ *satisfy* $C^u_D(h)$*,* $C^u_D(k)$*,* $C^u_D(\ell)$*, and* $\mathsf{ThreeSupp}^u_D(h, k, \ell)$*, and let* $s, t, \upsilon, v, w, z : \mathsf{BMark}$ *and* $L, R, r : \mathsf{BHist}$ *be the selected selectors and contexts. For Rat-classified endpoint pairs* $x, x', y, y' : \mathsf{BHist}$ *satisfying*

$$\mathsf{RatHistoryClassifier}(x, x') \wedge \mathsf{RatHistoryClassifier}(y, y'),$$

for a carried denominator $q : \mathsf{BHist}$ *satisfying* $\mathsf{RatHistoryCarrier}(q)$*, fix apart-zero context data* $L_0, R_0, \tau : \mathsf{BHist}$ *with* $p : \mathsf{FieldApartZero}(L_0)$*, phase guard histories* $\alpha, \beta : \mathsf{BHist}$*, the contextual action* Γ*, the affine source* AffAut_F*, and the same BEDC history operations add, mul, neg, and zero satisfying the CommRing hypotheses used by Definition 44.85. If*

$$H : \big((\alpha + \beta)\mathsf{hsame0} \vee (\alpha + (-\beta))\mathsf{hsame0}\big),$$

and histories $u, d : \mathsf{BHist}$ *satisfy* $U_D(u)$ *and* $\mathsf{RatHistoryCarrier}(d)$*, then the certificate-obligation record of Definition 45.337 contains one concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *phase-exit boundary consisting of:*

1. *the phase-exit record certificate boundary of Definition 45.792 for the displayed source, operation table, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, carried denominator, apart-zero data, guard hypothesis, contextual action, affine source, operations, and external-unit witness;*

2. *the final row inventory absorbed by Definition 45.340 for the same denominator tuple, histories, carrier witnesses, selectors, contexts, and support witness;*
3. *the carried-external classifier separation supplied by Definition 45.793 for the same source, operation table, exit object, denominator tuple, histories, selectors, contexts, endpoint pairs, carried denominator, apart-zero data, guard hypothesis, contextual action, affine source, operations, external-unit witness, and carried denominator endpoint.*

These three row families are read as one certificate-record obligation over the displayed BHist *source, with no source, tuple, selector, context, endpoint, guard, operation, external-unit witness, or carried endpoint reindexed.*

Proof. Instantiate Definition 45.340 with the displayed denominator tuple, histories h, k, ℓ, carrier witnesses, selectors s, t, v, v, w, z, contexts L, R, r, and the support witness $\mathsf{ThreeSupp}^u_D(h, k, \ell)$. This places the complete final row inventory in the certificate-obligation record.

Next instantiate Definition 45.793 with the same concrete source, operation table, exit object, denominator tuple, histories, selectors, contexts, Rat endpoint pairs, carried denominator, apart-zero data, phase guard histories, contextual action, affine source, operations, guard hypothesis, external-unit witness, and carried denominator endpoint. Project its phase-exit boundary row and its carried-external classifier separation row.

The projected boundary row contains Definition 45.792 at the same indices, while the absorbed inventory row is already stored in the certificate record. Conjunction introduction repacks the boundary, the absorbed inventory, and the carried-external separation as one concrete certificate-record obligation. No source, tuple, selector, context, endpoint, guard, witness, or operation is reindexed. □

Per-object template: traditional target rational field phase-exit boundary as certificate-record obligation; BEDC source BHist, BMark, $\mathsf{Rat}^\uparrow$, $\mathsf{CommRing}^\uparrow$, and $\mathsf{Field}^\uparrow$; carriers $\mathsf{RatHistoryCarrier}$, C^u_D, U_D, and AffAut_F; classifiers $\mathsf{RatHistoryClassifier}$, R^u_D, hsame, and affine classifier; stability obligations phase-exit record certificate boundary, record-absorbed final row inventory, and carried-external classifier separation; ledger denominator tuple, operation table, external-unit endpoint, carried denominator endpoint, guard witness, contextual action, affine source, histories, contexts, selectors, and carrier witnesses; core theorem this block; exactness the displayed certificate-record phase-exit boundary; downstream consumer Phase Algebra Hierarchy $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit criterion; status $\mathsf{ProofTarget}$.

Theorem 45.795 (RatUp FieldUp denominator total readback)**.** *Fix the concrete source* $\mathsf{RatUpFieldSource}_F$*, the operation table of Definition 45.587, the concrete exit object* $\mathcal{E}_F$*, and the denominator tuple* $(C^u_D, R^u_D, \mu^u_D, e^u_D, N^u_D)$*. Let* $q : \mathsf{BHist}$ *be the carried denominator, let* $p : \mathsf{FieldApartZero}(L_0)$ *be the apart-zero guard, let* $x, x', y, y' : \mathsf{BHist}$ *be the Rat-classified endpoint pairs, and let* $d : \mathsf{BHist}$ *be the carried denominator endpoint. Every public denominator read of the Phase Algebra Hierarchy exit package for these data factors through the same three rows: the phase-exit record certificate boundary, the final-inventory separation row, and the certificate-record absorption row. The readback does not replace any denominator, guard, operation-table, action, inverse, or endpoint row.*

Proof. First apply Definition 45.792. This places the operation table, exit object, denominator tuple, carried denominator q, apart-zero guard p, endpoint pairs, contextual action, external-unit witness, and endpoint d on the displayed phase-exit boundary.

Next apply Definition 45.793. It separates the carried external row from the final denominator inventory while keeping the same source, tuple, operation table, guard, action, inverse, and endpoint rows.

Finally apply Definition 45.794. The certificate record absorbs the preceding boundary and inventory rows as a single public obligation. Since all three applications use the same displayed indices, the denominator readback is total over the carried certificate record and no row is reindexed or replaced. □

Closure status for $\mathsf{NameCert}_{\mathsf{Field}^\uparrow}$**.**

Constructive story: A field element at this boundary is still a $\mathsf{Rat}^\uparrow$ denominator history, but inverse data may appear only when an explicit apart-from-zero witness accompanies the element. Thus the carrier is the rational denominator carrier, the classifier is the rational classifier extended so inverse comparisons preserve the guard witness, and there is never a blanket classical test saying merely that $a \neq 0$. The source is the rational-denominator predicate, the pattern records the certificate-record rows, final-row inventory, selector-independent exit rows, concrete addition/negation/multiplication/inverse table, apart-zero inverse rows, denominator normalization, endpoint exactness, and carried-external-separation guarantee, while the ledger keeps the visible field element tied to its raw rational form. Since $\mathsf{Rat}^\uparrow$ already supplies a name certificate and the guarded inverse predicate respects the rational classifier when the apartness witness is preserved, the carrier has representatives, the classifier is reflexive, symmetric, and transitive, and recognized representatives remain valid field data. A rational denominator d lies in the signature gap of a $\mathsf{Field}^\uparrow$ token p when the source-domain accepts d, the signature relation pairs it with an answer denominator and evidence ledger, possibly carrying inverse evidence, and the token-introduction relation names that answer as p; unfolded, d is a rational carrier element and an inverse witness is present exactly when p encodes a nonzero element. The conservative paper bridge reads this record across $\mathsf{Rat}^\uparrow$, $\mathsf{CommRing}^\uparrow$, $\mathsf{Field}^\uparrow$, and the kernel history type without changing the underlying denominator carrier. The chapter registers this certificate-record boundary under the name $\mathsf{Field}^\uparrow$; inverse evidence enters through guarded rows and stays separate from raw continuation data, with no arbitrary field, extension theory, valuation theory, or classical nonzero oracle introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Field}^\uparrow}$, maturePackageClosed)

Scope closed: The mature paper scope is the concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ exit record: certificate-record boundary, final row inventory, selector-independent exit rows, concrete operation table, apart-zero inverse rows, denominator normalization, endpoint exactness, carried-external separation, and the conservative paper bridge over $\mathsf{Rat}^\uparrow$, $\mathsf{CommRing}^\uparrow$, $\mathsf{Field}^\uparrow$, and BHist.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Field}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.FieldUp.FieldApartZero\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Unchecked bridge targets, arbitrary fields, field extensions, algebraic closure, valuation theory, ordered-field completeness, and machine-checked standard bridge closure remain outside this boundary.

Upgrade path: No further upgrade – chapter at top of formal axis.

45.25 Terminal history field instance

Definition 45.796 (Terminal history FieldUp instance)**.** *The terminal history* $\mathsf{Field}^\uparrow$ *instance is the concrete* BHist *tuple*

$$\mathcal{F}_{\mathsf{e}} := (A_{\mathsf{e}}, R_{\mathsf{e}}, +_{\mathsf{e}}, -_{\mathsf{e}}, 0_{\mathsf{e}}, \cdot_{\mathsf{e}}, 1_{\mathsf{e}}, N_{\mathsf{e}}, (\ )_{\mathsf{e}}^{-1})$$

with carrier

$$A_{\mathsf{e}}(h) := h\mathsf{hsamee}$$

and classifier

$$R_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

The additive and multiplicative endpoints are the same terminal append rows:

$$h +_{\mathsf{e}} k := \mathsf{append}(h,k), \qquad h \cdot_{\mathsf{e}} k := \mathsf{append}(h,k),$$

with $-_{\mathsf{e}} h := \mathsf{e}$, $0_{\mathsf{e}} := \mathsf{e}$, *and* $1_{\mathsf{e}} := \mathsf{e}$. *The inverse operation is defined only on the displayed nonzero boundary row:*

$$h_{\mathsf{e}}^{-1} := \mathsf{e}$$

whenever the unit-boundary certificate $N_{\mathsf{e}}(h)$ is present. The predicate N_{e} has no public rows except the certified unit endpoint over e*, and that endpoint is carried with its* hsame*,* Cont*, and* Pkg *provenance.*

Theorem 45.797 (Terminal history FieldUp laws)**.** *The tuple $\mathcal{F}_{\mathsf{e}}$ of Definition 45.796 satisfies the displayed* $\mathsf{Field}^{\uparrow}$ *laws: the* $\mathsf{Ring}^{\uparrow}$ *laws, commutative multiplication, unit laws, distributivity, and the inverse law on the stated unit-boundary nonzero row. Every carried endpoint appearing in these laws reduces through R_{e} to the terminal history* e*.*

Proof. Apply Definition 43.63. It supplies the singleton carrier A_{e}, classifier R_{e}, displayed addition, negation, zero, multiplication, one, distributivity, and operation ledger over the same terminal BHist endpoint.

The commutative multiplication and unit rows are the singleton public rows of Definition 44.92, read with the same carrier and classifier. Since 1_{e} is e, both left and right unit checks reduce to the terminal endpoint.

For the inverse row, project the displayed nonzero boundary certificate. Its only public endpoint is the unit row over e, and the inverse endpoint is defined as e. Therefore $(h_{\mathsf{e}}^{-1} \cdot_{\mathsf{e}} h)$ and $(h \cdot_{\mathsf{e}} h_{\mathsf{e}}^{-1})$ are both classified by R_{e} against 1_{e}.

No further case remains: every operation in the tuple either returns e directly or appends carried endpoints already classified with e, and the singleton append rows read those append endpoints back to the same terminal carrier. □

Theorem 45.798 (Terminal history FieldUp schema laws)**.** *The terminal history instance $\mathcal{F}_{\mathsf{e}}$ supplies the* $\mathsf{Field}^{\uparrow}$ *schema surface: the carrier A_{e} carries a semantic certificate, the empty endpoint cannot be a displayed nonzero row, nonzero rows transport along R_{e}, the inverse endpoint remains in the carrier, inverse endpoints are congruent under R_{e}, and both products $h_{\mathsf{e}}^{-1} \cdot_{\mathsf{e}} h$ and $h \cdot_{\mathsf{e}} h_{\mathsf{e}}^{-1}$ classify against 1_{e} whenever the displayed nonzero boundary row for h is present.*

Proof. Read the carrier and classifier from Definition 45.796. The carrier is the terminal hsame row against e, so the singleton endpoint reduction used in Definition 45.797 gives the semantic certificate and all classifier transports.

The displayed nonzero predicate has no row over the empty endpoint except the certified unit-boundary endpoint, and that endpoint reduces to the same terminal carrier. Hence the empty endpoint is not a separate nonzero object, and transporting a nonzero row across R_{e} keeps the same displayed boundary data.

Finally, the inverse operation always returns e. By Definition 45.797, both inverse-product rows therefore reduce through the singleton classifier to $1_{\mathsf{e}} = \mathsf{e}$, and the same terminal reduction proves inverse carrier closure and inverse classifier congruence. □

Theorem 45.799 (Terminal FieldUp nonzero boundary)**.** *Any* $\mathsf{Field}^{\uparrow}$ *inverse consumer of Definition 45.796 may read only the singleton unit/nonzero row, the carrier classifier A_{e}, the terminal classifier R_{e}, the displayed* hsame *witnesses, finite* Cont *provenance, and* Pkg *provenance. It receives no host field, quotient of nonzero elements, or external inverse object.*

Proof. Unpack the terminal tuple. The carrier and classifier expose only histories classified with e, and the operation rows are the displayed empty endpoint or append endpoints already governed by the singleton public operation surface.

The inverse row is indexed by N_{e}, whose public boundary is the certified unit endpoint. Projecting that boundary gives exactly the hsame, Cont, and Pkg rows attached to the same terminal history.

Thus a consumer can form inverse statements only by using the displayed unit-boundary certificate and the singleton classifier. There is no exported row from which a host field, a quotient carrier of nonzero elements, or an external inverse operation could be read. □

Theorem 45.800 (Terminal FieldUp history scope package)**.** *The terminal* $\mathsf{Field}^{\uparrow}$ *carrier of Definition 45.796 is a one-endpoint* BHist *scope packet. Every admitted carrier row is classified by* A_{e} *against* e*, every classifier comparison factors through* R_{e}*, and every visible dependency is one of the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows already attached to the terminal endpoint.*

Proof. Project the terminal tuple and use Definition 45.797. The carrier is the singleton history endpoint, and all additive, multiplicative, zero, unit, and inverse rows reduce back to e through the same classifier.

Apply Definition 45.798. It identifies the schema-facing rows with the displayed terminal carrier, classifier transport, and nonzero boundary, so no second history endpoint can enter the scope.

The remaining route data are exactly the hsame witnesses, finite Cont provenance, Pkg provenance, and local naming certificate already listed in Definition 45.799. Thus the terminal carrier is a closed one-endpoint history scope packet. □

Theorem 45.801 (Terminal FieldUp inverse ledger scope)**.** *In the terminal* $\mathsf{Field}^{\uparrow}$ *packet, every inverse row and every apart-zero row is exhausted by the unit-boundary witness* N_{e} *and its transport through* R_{e}*. The inverse ledger exposes no nonterminal nonzero carrier, no quotient of invertible elements, and no inverse operation outside the singleton* BHist *endpoint.*

Proof. Start with the nonzero boundary row of Definition 45.799. It says that an inverse consumer can read only the unit/nonzero row at e together with the carrier and classifier rows of the terminal tuple.

Use Definition 45.798 to transport that row along R_{e}. The transported nonzero datum remains the same displayed unit-boundary certificate, and the inverse endpoint is again e.

Finally apply Definition 45.797. The inverse products classify against 1_{e} by the singleton law surface, so every inverse and apart-zero ledger row has already been consumed by the terminal witness. □

Theorem 45.802 (Terminal FieldUp RatUp exit handoff)**.** *The terminal history packet hands off from the existing* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Field}^{\uparrow}$ *exit surface by the same singleton* BHist *endpoint: carrier admission, classifier transport, inverse use, apart-zero use, and consumer exit all factor through* e*,* A_{e}*,* R_{e}*, the displayed* hsame *and* Cont *rows, the* Pkg *row, and the local* NameCert *row.*

Proof. Read the exit-facing surface through the terminal carrier of Definition 45.796. The carrier and classifier rows are the same singleton rows used by the terminal $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Field}^{\uparrow}$ public operation packets.

Apply Definition 45.800 for carrier and classifier scope, then apply Definition 45.801 for the inverse and apart-zero ledger rows. These two scope statements cover the rows a consumer can inspect before leaving the terminal field packet.

Thus the exit handoff is not an external field construction. It is the existing singleton history endpoint with its recorded hsame, Cont, Pkg, and NameCert transport data. □

The terminal operation-table, nonzero-inverse, and denominator-selector rows for $\mathsf{Field}^{\uparrow}$ consume the $\mathsf{Rat}^{\uparrow}$ denominator surface of chapter 37, the $\mathsf{Ring}^{\uparrow}$ operation rows of chapter 43, and the $\mathsf{CommRing}^{\uparrow}$ commutative multiplication surface of chapter 44. This is a finite NameCert sibling route through the algebra lattice, not a host field primitive.

Chapter 46

A Concrete Naming Certificate for $\mathsf{Module}^{\uparrow}$

The module interface $\mathsf{Module}^{\uparrow}$ has certificate obligations over a fixed scalar ring satisfying chapter 43 together with an additive abelian group from chapter 42 as carrier, equipped with a scalar multiplication $\cdot : R \times M \to M$ satisfying associativity with the ring product, distributivity over additions on both sides, and unit law for the ring identity. It is not a primitive structure; it is admitted only by composing the ring and abelian group certificates with the scalar action fields.

46.1 Carrier definition

Definition 46.1 (Module carrier)**.** *Use a scalar ring R, an additive abelian group M, and scalar multiplication $R \times M \to M$.*

46.2 Concrete singleton empty-history module instance

Definition 46.2 (Singleton empty-history module instance)**.** *Let the scalar carrier and module carrier both be the singleton history source A_{e} of Definition 36.169 at $a_0 = \mathsf{e}$. Define the shared classifier*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Use the singleton empty-history ring operations of Theorem 43.6 for the scalar side, and the singleton empty-history abelian-group operations of Theorem 42.10 for the additive module side. Define the scalar action by

$$\mathsf{smul}(r, m) := \mathsf{e}.$$

This instance has traditional target the one-element module over the one-element ring; BEDC source BHist*; scalar and module carrier A_{e}; classifier ρ_{e}; stability obligations given by the singleton ring laws, singleton abelian-group laws, and scalar-action laws; ledger constant-action provenance; core theorem Theorem 46.14; exactness the singleton endpoint classification $A_{\mathsf{e}}(h)$; standard bridge the one-object module; and status* ProofTarget*.*

Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonCarrier`
Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonClassifier`
Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonAdd`
Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonMul`

Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonOne`
Lean4 (definition) `BEDC.Derived.ModuleUp.ModuleSingletonSmul`

Theorem 46.3 (Module singleton classifier empty endpoints)**.** *For histories h, k : BHist, the concrete module singleton classifier satisfies*

$$\rho_{\mathsf{e}}(h, k) \quad \Longleftrightarrow \quad h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction unpacks the two carrier components of the classifier, which are exactly the two empty-endpoint comparisons.

Conversely, the two endpoint comparisons give the carrier components, and composing $h\mathsf{hsamee}$ with the symmetry of $k\mathsf{hsamee}$ supplies the $\mathsf{hsame}(h, k)$ component of the classifier. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonClassifier empty endpoints iff`

Theorem 46.4 (Module singleton classifier hsame endpoint transport)**.** *For histories h, k, h', k' : BHist, if $\rho_{\mathsf{e}}(h, k)$, $h\mathsf{hsame}h'$, and $k\mathsf{hsame}k'$, then*

$$\rho_{\mathsf{e}}(h', k') \wedge h'\mathsf{hsamee} \wedge k'\mathsf{hsamee}.$$

Proof. Unpack $\rho_{\mathsf{e}}(h, k)$ to obtain $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. Compose these endpoint comparisons with the symmetries of $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$ to obtain $h'\mathsf{hsamee}$ and $k'\mathsf{hsamee}$. Theorem 46.3 then repacks the two empty endpoints as $\rho_{\mathsf{e}}(h', k')$. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonClassifier hsame endpoint transport`

Theorem 46.5 (Module singleton scalar readback)**.** *For histories r, m, n : BHist, if m lies in the singleton carrier A_{e}, then the scalar action endpoint is history-same to e, and classification after scalar action reads back to the original module endpoint:*

$$A_{\mathsf{e}}(m) \Longrightarrow \mathsf{hsame}(\mathsf{smul}(r, m), \mathsf{e}) \wedge \big(\rho_{\mathsf{e}}(\mathsf{smul}(r, m), n) \Longleftrightarrow \rho_{\mathsf{e}}(m, n)\big).$$

Proof. The scalar action is definitionally the empty history. The given carrier witness identifies m with e; composing this witness with the classifier relation supplies the forward readback, and composing its symmetry supplies the reverse direction. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul classifier readback iff`

Theorem 46.6 (Module singleton scalar action image carrier exactness)**.** *For the concrete singleton $\mathsf{Module}^{\uparrow}$ instance, let*

$$P(h) \;:\Longleftrightarrow\; \exists r, m : \mathsf{BHist},\ A_{\mathsf{e}}(m) \wedge h\mathsf{hsameModuleSingletonSmul}(r, m).$$

Then the scalar-action image pattern is exactly the singleton carrier:

$$P(h) \quad \Longleftrightarrow \quad A_{\mathsf{e}}(h).$$

The ten-field fit is: traditional target the one-element module action image; BEDC source BHist *with* $\mathsf{ModuleSingletonSmul}$*; carrier A_{e}; classifier ρ_{e}; stability from Theorems 46.3 and 46.5; ledger scalar-action witnesses; core theorem this block; exactness the displayed iff; standard bridge the terminal module; and status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, unpack a witness $r, m : \mathsf{BHist}$ with $A_{\mathsf{e}}(m)$ and $h\mathsf{hsameModuleSingletonSmul}(r, m)$. The scalar action endpoint is definitionally e, so this comparison composes to $h\mathsf{hsamee}$. By the singleton carrier definition, that endpoint comparison is precisely $A_{\mathsf{e}}(h)$.

For the reverse direction, assume $A_{\mathsf{e}}(h)$. Choose $r = \mathsf{e}$ and $m = h$. The module witness is the given carrier evidence, and $\mathsf{ModuleSingletonSmul}(r, m)$ is definitionally e. The same singleton carrier witness compares h with this empty scalar-action endpoint, so the chosen pair lies in the image pattern. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul image empty endpoint iff`

Theorem 46.7 (Module singleton scalar action image semantic certificate)**.** *For the concrete singleton* $\mathsf{Module}^{\uparrow}$ *instance, the source carrier* A_{e}*, the scalar-action image pattern*

$$P(h) \;:\Longleftrightarrow\; \exists r, m : \mathsf{BHist},\ A_{\mathsf{e}}(m) \wedge h\mathsf{hsameModuleSingletonSmul}(r, m),$$

the ledger carrier A_{e}*, and the singleton classifier* ρ_{e} *form a* $\mathsf{SemanticNameCert}$*.*

Proof. The certificate core is the singleton module classifier from Theorem 46.14. Its carrier is A_{e}, its classifier is ρ_{e}, and its stability fields give the reflexive, symmetric, transitive, and carrier-transport clauses required by the semantic certificate.

For pattern soundness, assume $A_{\mathsf{e}}(h)$. Choose the scalar and module witnesses $r = \mathsf{e}$ and $m = h$. The module witness is the given carrier evidence, and $\mathsf{ModuleSingletonSmul}(r, m)$ is definitionally the empty history, so the same singleton evidence supplies $h\mathsf{hsameModuleSingletonSmul}(r, m)$.

Ledger soundness is the identity map on the singleton carrier A_{e}, with classification read by the same ρ_{e} endpoint relation. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul image semanticNameCert`

Theorem 46.8 (Module singleton scalar action graph exactness)**.** *For histories* $r, m, n : \mathsf{BHist}$*, define the concrete scalar-action graph predicate*

$$G(r, m, n) \;:\Longleftrightarrow\; A_{\mathsf{e}}(r) \wedge A_{\mathsf{e}}(m) \wedge \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), n).$$

Then this graph is exactly the singleton carrier on scalar input, module input, and classified output:

$$G(r, m, n) \quad\Longleftrightarrow\quad A_{\mathsf{e}}(r) \wedge A_{\mathsf{e}}(m) \wedge A_{\mathsf{e}}(n).$$

The ten-field fit is: traditional target the terminal module action graph; BEDC source BHist *with* $\mathsf{ModuleSingletonSmul}$*; carrier* A_{e}*; classifier* ρ_{e}*; stability from Theorems 46.3, 46.5, and 46.14; ledger scalar endpoint, module endpoint, and classified result endpoint; core theorem this block; exactness the displayed iff; downstream consumer the* $\mathsf{Module}^{\uparrow}$ *action-row certificate; and status* $\mathsf{ProofTarget}$*.*

Proof. Unfold $\mathsf{ModuleSingletonSmul}$ as the constant empty-history action. The only nontrivial classifier read in the graph is therefore $\rho_{\mathsf{e}}(\mathsf{e}, n)$.

For the forward direction, unpack $G(r, m, n)$. The first two projections are already the scalar endpoint witness $A_{\mathsf{e}}(r)$ and the module endpoint witness $A_{\mathsf{e}}(m)$. The classifier projection, together with Theorem 46.3, reads $\rho_{\mathsf{e}}(\mathsf{e}, n)$ as the two empty endpoint witnesses; the second one is precisely the singleton carrier witness for n.

For the reverse direction, assume $A_{\mathsf{e}}(r) \wedge A_{\mathsf{e}}(m) \wedge A_{\mathsf{e}}(n)$. Reuse the scalar and module carrier witnesses unchanged. Since $\mathsf{ModuleSingletonSmul}(r, m)$ is definitionally e, Theorem 46.3 and the singleton carrier witness for n give $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), n)$, which repacks the graph predicate. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul graph empty exact iff`

Theorem 46.9 (Module singleton scalar action graph output determinacy)**.** *For histories* $r, m, n, n' : \mathsf{BHist}$*, define the concrete scalar-action graph predicate*

$$G(r, m, n) \;:\Longleftrightarrow\; A_{\mathsf{e}}(r) \wedge A_{\mathsf{e}}(m) \wedge \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), n).$$

Then any two outputs of the same scalar and module inputs are classified equivalent:

$$G(r, m, n) \wedge G(r, m, n') \quad \Longrightarrow \quad \rho_{\mathsf{e}}(n, n').$$

The ten-field fit is: traditional target terminal module action determinacy; BEDC source BHist *with* ModuleSingletonSmul*,* A_{e}*, and* ρ_{e}*; carrier* A_{e}*; classifier* ρ_{e}*; stability from graph exactness and classifier endpoint exactness; ledger scalar, module, and output endpoint witnesses; core theorem this block; exactness output uniqueness up to classifier; downstream consumer the* $\mathsf{Module}^{\uparrow}$ *action-row certificate; and status* ProofTarget*.*

Proof. Assume $G(r, m, n) \wedge G(r, m, n')$. Apply Theorem 46.8 to the first graph witness to obtain the output endpoint witness $A_{\mathsf{e}}(n)$. Apply the same theorem to the second graph witness to obtain $A_{\mathsf{e}}(n')$.

Theorem 46.3 identifies ρ_{e} with pairs of singleton empty-history endpoints. Using the two endpoint witnesses just obtained, package the pair as $\rho_{\mathsf{e}}(n, n')$. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul graph output determinacy`

Theorem 46.10 (Module singleton scalar action output fiber semantic certificate)**.** *For histories* $r, m, n : \mathsf{BHist}$*, if* $A_{\mathsf{e}}(r)$*,* $A_{\mathsf{e}}(m)$*, and* $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), n)$*, then the concrete output fiber*

$$F_{r,m}(o) \;:\Longleftrightarrow\; \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), o)$$

forms a semantic naming certificate with source, pattern, and ledger all equal to $F_{r,m}$*, and classifier* ρ_{e}*.*

Proof. The displayed output n inhabits the source fiber. Reflexivity, symmetry, and transitivity are inherited from the singleton classifier certificate. For carrier transport, compose the source classification $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), o)$ with $\rho_{\mathsf{e}}(o, o')$. Pattern and ledger soundness are identity maps on the same concrete fiber. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul output fiber semanticNameCert`

Theorem 46.11 (Module singleton scalar action fiber exhaustion)**.** *For histories* $r, m, o : \mathsf{BHist}$ *with* $A_{\mathsf{e}}(r)$ *and* $A_{\mathsf{e}}(m)$*, define the concrete scalar-action output fiber by*

$$F_{r,m}(o) \;:\Longleftrightarrow\; \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), o).$$

Then the fiber is exactly the singleton classifier class:

$$F_{r,m}(o) \quad \Longleftrightarrow \quad A_{\mathsf{e}}(o).$$

Moreover, if $F_{r,m}(o)$ *and* $F_{r,m}(o')$*, then* $\rho_{\mathsf{e}}(o, o')$*.*

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul fiber exhaustion`

Proof. For the forward direction, unfold the fiber predicate. The concrete action $\mathsf{ModuleSingletonSmul}(r, m)$ is the constant empty endpoint, so Theorem 46.3 reads the classification $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), o)$ back as the singleton endpoint witness $A_{\mathsf{e}}(o)$.

For the reverse direction, assume $A_{\mathsf{e}}(o)$. The same endpoint exactness theorem applies to the empty action endpoint and o, giving $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, m), o)$ and hence $F_{r,m}(o)$.

For the fiber classification claim, combine the two fiber witnesses as graph witnesses with the fixed input endpoint witnesses $A_{\mathsf{e}}(r)$ and $A_{\mathsf{e}}(m)$. Theorem 46.9 then returns $\rho_{\mathsf{e}}(o, o')$. □

Theorem 46.12 (Module singleton empty-action output determinacy)**.** *For histories r, m, n, n' : BHist, if the empty scalar action classifies n and the empty module input action classifies n', then the two outputs are classified together:*

$$\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(\mathsf{e}, m), n) \wedge \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(r, \mathsf{e}), n') \Longrightarrow \rho_{\mathsf{e}}(n, n').$$

Proof. Read both classifier hypotheses through Theorem 46.3. The first gives $n\mathsf{hsamee}$ and the second gives $n'\mathsf{hsamee}$. Repack these two endpoint witnesses with the same endpoint exactness theorem to obtain $\rho_{\mathsf{e}}(n, n')$. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul empty action outputs deterministic`

Theorem 46.13 (Module singleton visible scalar nonfaithfulness)**.** *For the concrete singleton scalar action, every carried module endpoint sees the visible scalar* $\mathsf{E1}(\mathsf{e})$ *and the empty scalar* e *as producing the same empty action endpoint:*

$$A_{\mathsf{e}}(m) \Longrightarrow \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(\mathsf{E1}(\mathsf{e}), m), \mathsf{ModuleSingletonSmul}(\mathsf{e}, m)).$$

Nevertheless the visible scalar itself is not classified with the empty scalar:

$$\rho_{\mathsf{e}}(\mathsf{E1}(\mathsf{e}), \mathsf{e}) \Longrightarrow \bot.$$

Proof. Both scalar-action endpoints reduce definitionally to e, so classifier endpoint exactness packages the two empty endpoints for the first assertion. For the second assertion, the left carrier field of $\rho_{\mathsf{e}}(\mathsf{E1}(\mathsf{e}), \mathsf{e})$ would give $\mathsf{hsame}(\mathsf{E1}(\mathsf{e}), \mathsf{e})$, contradicting the BEDC constructor no-confusion theorem for a visible one-step history and the empty history. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul nonfaithful visible scalar`

Theorem 46.14 (Singleton empty-history module laws)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1, \mathsf{smul})$$

satisfies the $\mathsf{Module}^{\uparrow}$ *schema over the concrete* BHist *carrier: the scalar ring fields, the additive abelian-group fields, scalar-action closure, scalar-action congruence, scalar associativity, distributivity over module addition, distributivity over scalar addition, and scalar unit law.*

Proof. The scalar ring fields are supplied by Theorem 43.6. The additive module fields are supplied by Theorem 42.10. These two certificates fix the shared singleton carrier A_{e} and classifier ρ_{e}.

For scalar-action closure, every endpoint $\mathsf{smul}(r, m)$ is definitionally e. The singleton source of Definition 36.169 gives $A_{\mathsf{e}}(\mathsf{e})$, hence the result lies in the module carrier.

For scalar-action congruence, suppose $\rho_{\mathsf{e}}(r, r')$ and $\rho_{\mathsf{e}}(m, m')$. Both action endpoints reduce to e. The two carrier witnesses are the same singleton witness $A_{\mathsf{e}}(\mathsf{e})$, and the relation component is reflexivity $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Together these form $\rho_{\mathsf{e}}(\mathsf{smul}(r, m), \mathsf{smul}(r', m'))$.

For scalar associativity, both $\mathsf{smul}(\mathsf{mul}(r,s),m)$ and $\mathsf{smul}(r,\mathsf{smul}(s,m))$ are definitionally e. The same singleton carrier witness and hsame reflexivity package the required ρ_{e} endpoint relation.

For distributivity over module addition, both $\mathsf{smul}(r,\mathsf{add}(m,n))$ and $\mathsf{add}(\mathsf{smul}(r,m),\mathsf{smul}(r,n))$ reduce to e. Closure is the singleton carrier witness, and the classifier relation is $\mathsf{hsame}(\mathsf{e},\mathsf{e})$.

For distributivity over scalar addition, both $\mathsf{smul}(\mathsf{add}(r,s),m)$ and $\mathsf{add}(\mathsf{smul}(r,m),\mathsf{smul}(s,m))$ reduce to e. Again the singleton carrier witnesses and reflexivity assemble the ρ_{e} witness.

For the scalar unit law, $\mathsf{smul}(1,m)$ is definitionally e. The input witness $A_{\mathsf{e}}(m)$ identifies m with e, and $A_{\mathsf{e}}(\mathsf{e})$ supplies the other endpoint. Hence $\rho_{\mathsf{e}}(\mathsf{smul}(1,m),m)$ follows from the carrier witnesses and the corresponding hsame endpoint. □

Lean4 ✓ `BEDC.Derived.ModuleUp.singleton empty history module laws`
Lean4 ✓ `BEDC.Derived.ModuleUp.singleton empty history module action fields`
Lean4 ✓ `BEDC.Derived.ModuleUp.module singleton empty laws`
Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingleton scalar action laws`

The $\mathsf{Module}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and scalar action, giving Lean formalization in $\mathsf{BEDC.Derived.ModuleUp}$ a BHist-anchored target for the module action laws.

46.3 The certificate

Definition 46.15 (Module source specification)**.** *The source is a certified* $\mathsf{Ring}^{\uparrow}(R)$ *and certified* $\mathsf{AbGroup}^{\uparrow}(M)$ *sharing a scalar action.*

Definition 46.16 (Module pattern specification)**.** *The pattern is module expression formation from vector addition, zero, negation, and scalar multiplication.*

Definition 46.17 (Module classifier specification)**.** *The classifier is the module carrier classifier, with scalar action congruent in both scalar and vector inputs.*

Definition 46.18 (Module stability certificate)**.** *The stability certificate consists of:*

1. *additive abelian-group laws on M*
2. *ring laws on R*
3. *scalar action congruence*
4. *scalar associativity with ring multiplication*
5. *scalar distributivity over module addition*
6. *scalar distributivity over ring addition*
7. *scalar unit law*

Definition 46.19 (Module ledger policy)**.** *Record scalar and module ledgers for every scalar-action expression.*

Definition 46.20 ($\mathsf{Module}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Module}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Module}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Definition 46.21 (Dual-number parity action)**.** *Let*

$$R := \mathbb{F}_2[\epsilon]/(\epsilon^2), \qquad M := \mathbb{F}_2.$$

Write every scalar of R uniquely as $a + b\epsilon$ with $a, b \in \mathbb{F}_2$. Define

$$\chi(a + b\epsilon) := a + b$$

and define the scalar action $\star : R \times M \to M$ by

$$r \star m := \chi(r)m.$$

Equivalently,

$$\chi(0) = 0, \qquad \chi(1) = 1, \qquad \chi(\epsilon) = 1, \qquad \chi(1 + \epsilon) = 0.$$

Proposition 46.22 (Module associativity boundary for additive unit action laws)**.** *For the action of Definition 46.21, the scalar ring R satisfies the ordinary ring fields used in chapter 43, the carrier M satisfies the additive abelian-group fields used in chapter 42, and equality classifiers make scalar-action congruence immediate. The action also satisfies scalar distributivity over module addition, scalar distributivity over ring addition, and the scalar unit law. It does not satisfy scalar associativity. Hence the scalar associativity item in Definition 46.18 is not forced by the remaining additive, congruence, and unit action fields.*

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonAssociativityBoundaryAdditiveUnitActionLaws`

Proof. The ring R is the four-element quotient ring with elements $0, 1, \epsilon, 1 + \epsilon$ and relation $\epsilon^2 = 0$. The carrier $M = \mathbb{F}_2$ is an additive abelian group. With equality as classifier, scalar-action congruence says that equal scalar and module inputs have equal action outputs, which follows directly from $r \star m = \chi(r)m$.

For scalar distributivity over module addition, let $r \in R$ and $m, n \in M$. Since $\chi(r) \in \mathbb{F}_2$ and multiplication in $\mathbb{F}_2$ distributes over addition,

$$r \star (m + n) = \chi(r)(m + n) = \chi(r)m + \chi(r)n = (r \star m) + (r \star n).$$

For scalar distributivity over ring addition, write $r = a + b\epsilon$ and $s = c + d\epsilon$. Then

$$\chi(r + s) = \chi((a + c) + (b + d)\epsilon) = a + c + b + d = (a + b) + (c + d) = \chi(r) + \chi(s),$$

with all sums in $\mathbb{F}_2$. Therefore, for every $m \in M$,

$$(r + s) \star m = \chi(r + s)m = (\chi(r) + \chi(s))m = \chi(r)m + \chi(s)m = (r \star m) + (s \star m).$$

The scalar unit law follows from $\chi(1) = 1$:

$$1 \star m = \chi(1)m = m.$$

Scalar associativity fails at $r = s = \epsilon$ and $m = 1$. Since $\epsilon^2 = 0$ in R,

$$(\epsilon\epsilon) \star 1 = 0 \star 1 = \chi(0)1 = 0.$$

On the other hand,

$$\epsilon \star (\epsilon \star 1) = \epsilon \star (\chi(\epsilon)1) = \epsilon \star 1 = \chi(\epsilon)1 = 1.$$

Thus

$$(\epsilon\epsilon) \star 1 \neq \epsilon \star (\epsilon \star 1),$$

so scalar associativity is a separate stability requirement for $\mathsf{Module}^{\uparrow}$. □

Theorem 46.23 (Module scalar representative transport compatibility)**.** *Let a* $\mathsf{Module}^{\uparrow}$ *certificate be given by Definition 46.20. Write* $\sim_R$ *for the scalar ring classifier,* $\sim_M$ *for the module classifier,* $\cdot_R$ *for scalar ring multiplication, and* $\odot$ *for scalar action. For carried scalar representatives* r, r', s, s' *and carried module representatives* m, m'*, if*

$$r \sim_R r', \qquad s \sim_R s', \qquad m \sim_M m',$$

then

$$r \odot (s \odot m) \sim_M (r' \cdot_R s') \odot m'.$$

Proof. By Definition 46.15, the scalar side of the $\mathsf{Module}^{\uparrow}$ certificate is a certified $\mathsf{Ring}^{\uparrow}$. The ring stability certificate of Definition 43.26 includes multiplication congruence, so the two scalar representative relations give

$$(r \cdot_R s) \sim_R (r' \cdot_R s').$$

By Definition 46.17 and Definition 46.18, scalar action is congruent in its scalar and module inputs. Applying scalar-action congruence to the transported scalar relation above and to $m \sim_M m'$ gives

$$((r \cdot_R s) \odot m) \sim_M ((r' \cdot_R s') \odot m').$$

The scalar associativity field listed in Definition 46.18 supplies a module-classifier witness between $(r \cdot_R s) \odot m$ and $r \odot (s \odot m)$. Since derived upward-arrow interfaces carry a $\mathsf{NameCert}$ by Definition 9.9, and Definition 9.7 supplies symmetry for the classifier, use that witness in the orientation

$$r \odot (s \odot m) \sim_M ((r \cdot_R s) \odot m).$$

Finally, Definition 9.8 composes this associativity witness with the scalar-action congruence witness, yielding

$$r \odot (s \odot m) \sim_M (r' \cdot_R s') \odot m'.$$

□

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul scalar representative transport compatibility`

Theorem 46.24 (Scalar associativity independent of additive unit action laws)**.** *The action in Definition 46.21 is a finite equality-classified scalar action over*

$$R = \mathbb{F}_2[\epsilon]/(\epsilon^2), \qquad M = \mathbb{F}_2$$

such that the scalar ring satisfies the ring fields of chapter 43 and the commutative-ring fields of chapter 44, the module carrier satisfies the additive abelian-group fields of chapter 42, scalar-action congruence holds, scalar distributivity over module addition holds, scalar distributivity over scalar addition holds, and the scalar unit law holds. Scalar associativity fails. Hence the scalar associativity field in Definition 46.18 is not a consequence of those surrounding additive, congruence, and unit fields.

Proof. Classify both carriers by ordinary equality. The scalar ring R has exactly the four elements 0, 1, ϵ, and $1+\epsilon$, with $\epsilon^2 = 0$, and it is a finite commutative ring. The module carrier M is the two-element additive abelian group. Equality classifiers make every operation congruence field immediate from substitution; in particular, if $r = r'$ and $m = m'$, then $r \star m = r' \star m'$.

Write an element of R as $a+b\epsilon$, with $a, b \in \mathbb{F}_2$. The map $\chi(a+b\epsilon) = a+b$ from Definition 46.21 is additive because, for $r = a + b\epsilon$ and $s = c + d\epsilon$,

$$\chi(r+s) = \chi((a+c)+(b+d)\epsilon) = a+c+b+d = \chi(r)+\chi(s).$$

For scalar distributivity over module addition, let $r \in R$ and $m, n \in M$. Since $\chi(r) \in \mathbb{F}_2$,

$$r \star (m+n) = \chi(r)(m+n) = \chi(r)m + \chi(r)n = (r \star m) + (r \star n).$$

For scalar distributivity over scalar addition, the additivity of χ gives

$$(r+s) \star m = \chi(r+s)m = (\chi(r)+\chi(s))m = \chi(r)m + \chi(s)m = (r \star m) + (s \star m).$$

The scalar unit law follows from $\chi(1) = 1$:

$$1 \star m = \chi(1)m = m.$$

It remains to test scalar associativity. At $r = s = \epsilon$ and $m = 1$,

$$(\epsilon\epsilon) \star 1 = 0 \star 1 = \chi(0)1 = 0,$$

while

$$\epsilon \star (\epsilon \star 1) = \epsilon \star (\chi(\epsilon)1) = \epsilon \star 1 = \chi(\epsilon)1 = 1.$$

The two values are distinct in $M = \mathbb{F}_2$, so

$$(\epsilon\epsilon) \star 1 \neq \epsilon \star (\epsilon \star 1).$$

Thus all listed fields except scalar associativity hold in a finite equality-classified action, while scalar associativity fails. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParityAction scalar associativity counterexample`

Theorem 46.25 (Module parity action associativity forces empty visible one)**.** *If a global scalar-associativity law holds for the concrete parity action, then the visible scalar one classifies with the empty history:*

$$(\forall r, s, m.\ (rs) \star m \mathsf{hsame} r \star (s \star m)) \Longrightarrow \mathsf{ModuleParityOnehsamee}.$$

Proof. Apply the displayed global law at $r = s = \mathsf{ModuleParityEps}$ and $m = \mathsf{ModuleParityOne}$. The concrete counterexample identifies the left associated value with e and the right associated value with $\mathsf{ModuleParityOne}$. Transitivity and symmetry of hsame therefore classify $\mathsf{ModuleParityOne}$ with e. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParityAction scalar associativity forces empty visible one`

Proposition 46.26 (Module forgets to AbGroup certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate and let M carry a $\mathsf{Module}^{\uparrow}(R, M)$ certificate. If the scalar-ring package, the scalar action $\odot : R \times M \to M$, and the scalar-action stability rows are omitted from the exported package, the retained carrier, source, additive pattern, classifier, additive stability fields, and additive ledger form an $\mathsf{AbGroup}^{\uparrow}(M)$ certificate. The carrier, addition, zero, additive inverse, and classifier of M are unchanged.*

Proof. By Definition 46.20, the given module certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Module}^{\uparrow}$. The construction keeps exactly the components whose operations and endpoint claims live only on the additive carrier M.

By Definition 46.1, a module carrier consists of a scalar ring R, an additive abelian group M, and scalar multiplication $R \times M \to M$. Removing the scalar multiplication and the scalar carrier from the exported object leaves the same additive carrier M with its addition, zero, additive inverse, and carrier classifier. These are the carrier data required by Definition 42.1.

By Definition 46.15, the source side of the module certificate is a certified $\mathsf{Ring}^{\uparrow}(R)$ together with a certified $\mathsf{AbGroup}^{\uparrow}(M)$ sharing the scalar action. After the scalar-ring certificate and the action-sharing datum are suppressed, the certified $\mathsf{AbGroup}^{\uparrow}(M)$ source remains. Unpacking it through Definition 42.5 gives the certified group carrier plus commutativity field required for the abelian-group source.

By Definition 46.17, the module classifier is the module carrier classifier. This is the classifier of the embedded additive abelian-group certificate, and Definition 42.7 requires exactly that carrier classifier. Hence no classifier transport or replacement is involved.

By Definition 46.16, module expressions are formed from vector addition, zero, negation, and scalar multiplication. Deleting the scalar-multiplication constructor leaves the additive expression language generated by addition, zero, and negation. Together with the retained commutativity field from the additive abelian-group certificate, this is the abelian-group pattern of Definition 42.6.

By Definition 46.18, the first stability component of a module certificate is the additive abelian-group laws on M. The remaining module-specific rows are scalar action congruence, scalar associativity, distributivity over module addition, distributivity over ring addition, and scalar unit law. Those rows mention the scalar action or the scalar ring, so they are not obligations for $\mathsf{AbGroup}^{\uparrow}$. The retained first component is exactly the stability package listed in Definition 42.8: the group stability fields, additive commutativity under the classifier, and the compatibility data for that commutativity with inverse and multiplication congruence.

Finally, Definition 46.19 records scalar and module ledgers for scalar-action expressions. Removing entries whose provenance uses scalar multiplication leaves the additive module ledger. This is the group-expression provenance and commutativity-use ledger of Definition 42.31. The retained carrier, source, pattern, classifier, stability, and ledger data therefore satisfy the obligations named in Definition 42.32, with the same additive operation, zero, inverse, and classifier carried by the original module certificate. □

Proposition 46.27 (Module forgets to Ring certificate)**.** *Let M carry a $\mathsf{Module}^{\uparrow}(R, M)$ certificate whose scalar source is a certified $\mathsf{Ring}^{\uparrow}(R)$. If the additive carrier M, vector addition, vector zero, vector negation, scalar action $\odot : R \times M \to M$, and the action-shaped stability and ledger rows are omitted from the exported package, the retained scalar carrier, source, ring pattern, classifier, ring stability fields, and ring ledger form a $\mathsf{Ring}^{\uparrow}(R)$ certificate. The operations 0_R, 1_R, $+_R$, $(-)_R$, $\cdot_R$, and the scalar classifier on R are unchanged.*

Proof. By Definition 46.20, a module certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Module}^{\uparrow}$. The scalar projection keeps exactly the components whose operations

and endpoint claims live only on the scalar carrier R.

By Definition 46.1, a module carrier consists of a scalar ring R, an additive abelian group M, and scalar multiplication $R \times M \to M$. Deleting the additive carrier and scalar action leaves the scalar ring carrier with its addition, zero, additive inverse, multiplication, one, and scalar carrier classifier. These are the carrier data required by Definition 43.2.

By Definition 46.15, the source side of the module certificate is a certified $\mathsf{Ring}^{\uparrow}(R)$ together with a certified $\mathsf{AbGroup}^{\uparrow}(M)$ sharing the scalar action. Suppressing the additive source and the action-sharing datum leaves the certified $\mathsf{Ring}^{\uparrow}(R)$ source. Unpacking it through Definition 43.23 gives the additive $\mathsf{AbGroup}^{\uparrow}$ and multiplicative $\mathsf{Monoid}^{\uparrow}$ data on the same scalar carrier and classifier.

The classifier retained by the projection is the scalar-ring classifier from that certified source. The module classifier of Definition 46.17 belongs to the module carrier M and is not exported in this projection; the retained scalar classifier is already the carrier sameness relation required by Definition 43.25. No classifier transport or replacement is introduced.

The retained expression language is the ring expression language of the certified scalar source: 0_R, 1_R, $+_R$, $(-)_R$, and $\cdot_R$. This is exactly the pattern of Definition 43.24. The vector constructors and scalar-action constructor are discarded because their endpoints lie in M or in the action $R \times M \to M$, not in the scalar ring.

By Definition 46.18, the module certificate contains ring laws on R in addition to additive module laws and scalar-action laws. Since Definition 46.15 identifies the scalar side with a certified $\mathsf{Ring}^{\uparrow}(R)$, unpacking the retained scalar stability through Definition 43.26 gives additive abelian-group laws, multiplicative monoid laws, operation congruence, distributivity, and the zero multiplication rows for the scalar operations. The scalar-action rows, scalar associativity, the two action distributivity rows, and the scalar unit law all mention $\odot$ or the module carrier, so they are not obligations for $\mathsf{Ring}^{\uparrow}$.

Finally, the scalar ledger retained from the certified $\mathsf{Ring}^{\uparrow}(R)$ source is the shared carrier ledger and ring operation tree of Definition 43.54. The discarded ledger entries are precisely those whose provenance uses vector operations or scalar action. Thus the retained carrier, source, pattern, classifier, stability, and ledger fields satisfy Definition 43.55, with 0_R, 1_R, $+_R$, $(-)_R$, $\cdot_R$, and the scalar classifier copied from the original module certificate. □

Lemma 46.28 (Additive duplicate cancellation to zero)**.** *Let M be an additive carrier with classifier $\sim_M$, addition $+_M$, and zero 0_M. Assume the right-zero row $x +_M 0_M \sim_M x$ and additive left cancellation: from $a +_M b \sim_M a +_M c$ one may conclude $b \sim_M c$. If $x \sim_M x +_M x$, then $x \sim_M 0_M$.*

Proof. The right-zero row gives $x +_M 0_M \sim_M x$. Compose this comparison with the supplied comparison $x \sim_M x +_M x$ to obtain

$$x +_M 0_M \sim_M x +_M x.$$

Additive left cancellation removes the common left summand x and yields $0_M \sim_M x$. Symmetry of the classifier gives $x \sim_M 0_M$. □

Lean4✓ `BEDC.Derived.ModuleUp.ModuleAdditive duplicate cancel empty`

Theorem 46.29 (Module zero-action annihilation)**.** *Let a $\mathsf{Module}^{\uparrow}$ certificate be given by Definition 46.20. Write $\sim_R$ for the scalar-ring classifier, $\sim_M$ for the module classifier, $+_R$ and 0_R for scalar addition and scalar zero, $+_M$ and 0_M for module addition and module zero, and $\odot$ for scalar action. Assume the scalar source has the zero row $0_R +_R 0_R \sim_R 0_R$ from Definition 43.26, the module additive source has right-zero and additive left-cancellation as in Definition 42.8 and*

Definition 41.66, and the action rows include scalar-action congruence and both distributivity rows from Definition 46.18. For every carried scalar r and carried module element m,

$$0_R \odot m \sim_M 0_M \qquad \text{and} \qquad r \odot 0_M \sim_M 0_M.$$

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingleton zero action annihilation`

Proof. Let $a = 0_R \odot m$. Scalar distributivity over ring addition gives

$$(0_R +_R 0_R) \odot m \sim_M a +_M a.$$

The scalar zero row $0_R +_R 0_R \sim_R 0_R$, classifier reflexivity on the carried element m, and scalar-action congruence give

$$(0_R +_R 0_R) \odot m \sim_M a.$$

Symmetry and transitivity of $\sim_M$ therefore give $a \sim_M a +_M a$. Applying Definition 46.28 in the additive carrier M gives $0_R \odot m \sim_M 0_M$.

Let $b = r \odot 0_M$. The module zero row gives $0_M +_M 0_M \sim_M 0_M$. Scalar-action congruence, using reflexivity on the carried scalar r, gives

$$r \odot (0_M +_M 0_M) \sim_M b.$$

Scalar distributivity over module addition gives

$$r \odot (0_M +_M 0_M) \sim_M b +_M b.$$

Again symmetry and transitivity give $b \sim_M b +_M b$, and Definition 46.28 gives $r \odot 0_M \sim_M 0_M$. □

Definition 46.30 (Classifier-faithful scalar action and action annihilator)**.** *Let a* $\mathsf{Module}^{\uparrow}(R, M)$ *certificate be given by Definition 46.20. Write* $\sim_R$ *for the scalar-ring classifier,* $\sim_M$ *for the module classifier,* 0_R *and* 0_M *for the two zero endpoints,* $+_R$ *and* $+_M$ *for the two additions, and* $\odot$ *for scalar action. For a carried scalar* $r : R$*, let* $L_r(m) := r \odot m$*.*

The pointwise classifier equality of the action endomorphisms determined by r *and* s *is*

$$\mathsf{ActEq}_{R,M}(r, s) \iff \forall m : M,\ L_r(m) \sim_M L_s(m).$$

The action is classifier-faithful when

$$\mathsf{FaithAct}_{R,M} \iff \forall r, s : R,\ \mathsf{ActEq}_{R,M}(r, s) \implies r \sim_R s.$$

The action annihilator predicate is

$$\mathsf{Ann}_M(r) \iff \forall m : M,\ r \odot m \sim_M 0_M.$$

The predicate body is the annihilator; no ideal carrier is introduced.

Lemma 46.31 (Pointwise action equality gives difference annihilator)**.** *Under the notation of Definition 46.30, if* $\mathsf{ActEq}_{R,M}(r, s)$*, then*

$$\mathsf{Ann}_M(r +_R (-s)).$$

Proof. Fix a carried module endpoint $m : M$. Scalar distributivity over scalar addition from Definition 46.18 gives

$$(r +_R (-s)) \odot m \sim_M (r \odot m) +_M ((-s) \odot m).$$

The same distributivity row at s gives

$$(s +_R (-s)) \odot m \sim_M (s \odot m) +_M ((-s) \odot m).$$

The scalar additive right-inverse row of Definition 43.27, read in the scalar source supplied by Definition 46.15, classifies $s +_R (-s)$ with 0_R. Scalar-action congruence from Definition 46.18, followed by Definition 46.29, gives

$$(s +_R (-s)) \odot m \sim_M 0_M.$$

Composing this with the symmetry of the preceding distributivity comparison yields

$$(s \odot m) +_M ((-s) \odot m) \sim_M 0_M.$$

Now use the hypothesis $\mathsf{ActEq}_{R,M}(r, s)$ at m. Module additive congruence, from the additive $\mathsf{AbGroup}^{\uparrow}$ component in Definition 46.15, transports the first summand and gives

$$(r \odot m) +_M ((-s) \odot m) \sim_M (s \odot m) +_M ((-s) \odot m).$$

Transitivity with the last zero comparison gives

$$(r \odot m) +_M ((-s) \odot m) \sim_M 0_M.$$

Finally compose with the scalar-distributivity comparison for $r +_R (-s)$ to obtain

$$(r +_R (-s)) \odot m \sim_M 0_M.$$

Since m was arbitrary, $\mathsf{Ann}_M(r +_R (-s))$ holds. □

Theorem 46.32 (Module faithful action iff zero annihilator)**.** *Let a* $\mathsf{Module}^{\uparrow}(R, M)$ *certificate be given by Definition 46.20. With the notation of Definition 46.30, the scalar action is classifier-faithful exactly when the only scalar whose action annihilates every carried module endpoint is the scalar zero:*

$$\mathsf{FaithAct}_{R,M} \Longleftrightarrow \forall r : R,\ \mathsf{Ann}_M(r) \Longrightarrow r \sim_R 0_R.$$

Proof. Assume $\mathsf{FaithAct}_{R,M}$. Let $r : R$ satisfy $\mathsf{Ann}_M(r)$. For every carried $m : M$, the annihilator hypothesis gives

$$r \odot m \sim_M 0_M.$$

By Definition 46.29, the zero scalar action gives

$$0_R \odot m \sim_M 0_M.$$

Classifier symmetry and transitivity compose these two comparisons into

$$r \odot m \sim_M 0_R \odot m.$$

Thus $\mathsf{ActEq}_{R,M}(r, 0_R)$, and classifier-faithfulness gives $r \sim_R 0_R$.

Conversely, assume

$$\forall t : R,\ \mathsf{Ann}_M(t) \Longrightarrow t \sim_R 0_R.$$

Let $r, s : R$ and assume $\mathsf{ActEq}_{R,M}(r, s)$. By Definition 46.31,

$$\mathsf{Ann}_M(r +_R (-s)).$$

The assumed zero-annihilator condition therefore gives

$$r +_R (-s) \sim_R 0_R.$$

Applying the left-to-right direction of the scalar-ring difference exactness theorem Definition 43.34, read through the certified scalar $\mathsf{Ring}^{\uparrow}$ source of Definition 46.15, yields

$$r \sim_R s.$$

Therefore $\mathsf{FaithAct}_{R,M}$ holds, and the two conditions are equivalent. □

Theorem 46.33 (Module action annihilator IdealUp predicate)**.** *Let a* $\mathsf{Module}^{\uparrow}(R, M)$ *certificate be given by Definition 46.20. Write* C_R *for the scalar-ring carrier predicate,* $\sim_R$ *for the scalar-ring classifier,* 0_R *for scalar zero,* $+_R$ *for scalar addition,* $-_R$ *for scalar additive inverse,* $\cdot_R$ *for scalar multiplication,* $\sim_M$ *for the module classifier,* 0_M *for module zero,* $+_M$ *for module addition, and* $\odot$ *for scalar action. With* Ann_M *as in Definition 46.30, set*

$$I_M(r) \Longleftrightarrow C_R(r) \wedge \mathsf{Ann}_M(r).$$

Then I_M *is an* $\mathsf{Ideal}^{\uparrow}$ *predicate over the scalar* $\mathsf{Ring}^{\uparrow}(R)$ *source. Explicitly,* I_M *has carrier support and the rows*

$$I_M(0_R), \qquad I_M(r) \wedge I_M(s) \Longrightarrow I_M(r +_R s), \qquad I_M(r) \Longrightarrow I_M(-_R r),$$

$$I_M(r) \wedge I_M(s) \Longrightarrow I_M(r \cdot_R s), \qquad I_M(r) \wedge r \sim_R r' \Longrightarrow I_M(r'),$$

and, for every scalar endpoint a *with* $C_R(a)$*,*

$$I_M(r) \Longrightarrow I_M(a \cdot_R r) \wedge I_M(r \cdot_R a).$$

Proof. By Definition 46.15, the scalar source of the module certificate is a certified $\mathsf{Ring}^{\uparrow}(R)$, and the module source contains a certified additive $\mathsf{AbGroup}^{\uparrow}(M)$. The first conjunct in the definition of I_M is the carrier-support row. The scalar source carries 0_R, sums, additive inverses, and products; carrier transport along $\sim_R$ is supplied by the naming-certificate core in Definition 9.7.

For zero membership, fix a carried module endpoint $m : M$. The zero-action theorem Definition 46.29 gives

$$0_R \odot m \sim_M 0_M.$$

Since m was arbitrary and 0_R is carried by the scalar source, $I_M(0_R)$ holds.

For classifier transport, assume $I_M(r)$ and $r \sim_R r'$. Carrier transport gives $C_R(r')$. For every carried $m : M$, classifier reflexivity gives $m \sim_M m$, and scalar-action congruence from Definition 46.18 gives

$$r \odot m \sim_M r' \odot m.$$

The annihilator component of $I_M(r)$ gives $r \odot m \sim_M 0_M$. Symmetry and transitivity of the module classifier, composed by Definition 9.8, therefore give $r' \odot m \sim_M 0_M$. Thus $I_M(r')$ holds.

For additive closure, assume $I_M(r)$ and $I_M(s)$, and fix a carried module endpoint $m : M$. Scalar distributivity over scalar addition from Definition 46.18 gives

$$(r +_R s) \odot m \sim_M (r \odot m) +_M (s \odot m).$$

The two annihilator witnesses give $r \odot m \sim_M 0_M$ and $s \odot m \sim_M 0_M$. Module additive congruence transports the right side to $0_M +_M 0_M$, and the additive zero row in the $\mathsf{AbGroup}^{\uparrow}(M)$ source reduces $0_M +_M 0_M$ to 0_M. Hence

$$(r +_R s) \odot m \sim_M 0_M.$$

Together with scalar additive closure carrying $r +_R s$, this proves $I_M(r +_R s)$.

For additive inverse closure, assume $I_M(r)$ and fix a carried $m : M$. The scalar additive left-inverse row in the scalar $\mathsf{Ring}^{\uparrow}$ source gives $(-_R r) +_R r \sim_R 0_R$. Scalar-action congruence and Definition 46.29 give

$$((-_R r) +_R r) \odot m \sim_M 0_R \odot m \sim_M 0_M.$$

Scalar distributivity over scalar addition also gives

$$((-_R r) +_R r) \odot m \sim_M ((-_R r) \odot m) +_M (r \odot m).$$

Thus

$$((-_R r) \odot m) +_M (r \odot m) \sim_M 0_M.$$

The annihilator witness gives $r \odot m \sim_M 0_M$, so module additive congruence and the right-zero row in the additive module source give

$$((-_R r) \odot m) +_M (r \odot m) \sim_M ((-_R r) \odot m) +_M 0_M \sim_M (-_R r) \odot m.$$

Symmetry and transitivity now yield $(-_R r) \odot m \sim_M 0_M$. Since the scalar inverse closure row carries $-_R r$, this proves $I_M(-_R r)$.

For two-sided absorption, assume $I_M(r)$ and $C_R(a)$, and fix a carried module endpoint $m : M$. The scalar multiplication closure row carries $a \cdot_R r$ and $r \cdot_R a$. For the left product, scalar associativity from Definition 46.18 gives, in the displayed orientation after classifier symmetry if needed,

$$(a \cdot_R r) \odot m \sim_M a \odot (r \odot m).$$

The annihilator witness for r gives $r \odot m \sim_M 0_M$. Scalar-action congruence transports this comparison under the carried scalar a, and Definition 46.29 gives

$$a \odot (r \odot m) \sim_M a \odot 0_M \sim_M 0_M.$$

Therefore $I_M(a \cdot_R r)$ holds.

For the right product, scalar-action closure carries $a \odot m$ as a module endpoint. Scalar associativity gives

$$(r \cdot_R a) \odot m \sim_M r \odot (a \odot m).$$

Applying the annihilator witness for r to the carried module endpoint $a \odot m$ gives

$$r \odot (a \odot m) \sim_M 0_M.$$

Thus $I_M(r \cdot_R a)$ holds.

For internal multiplication, assume $I_M(r)$ and $I_M(s)$. The carrier component of $I_M(s)$ gives $C_R(s)$, so the right-absorption row just proved, applied with $a = s$, gives $I_M(r \cdot_R s)$.

These are precisely the carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and two-sided absorption rows required by the $\mathsf{Ideal}^{\uparrow}$ certificate in section 81.2. □

Theorem 46.34 (Module parity epsilon empty-result classifier)**.** *For every history m : BHist, scalar action by the parity epsilon history has empty output exactly when the module endpoint is not the empty history:*

$$\mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps}, m), \mathsf{e}) \iff (\mathsf{hsame}(m, \mathsf{e}) \implies \mathsf{False}).$$

Proof. Perform constructor case analysis on m. If $m = \mathsf{e}$, the displayed result is $\mathsf{E1}(\mathsf{e})$, so empty output contradicts constructor disjointness; conversely the right-hand side is contradicted by reflexivity of e. If m is visibly zero-headed or one-headed, the parity action reduces to e, and the right-hand side follows from the corresponding visible-vs-empty no-confusion law. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul epsilon empty result iff`

Theorem 46.35 (Module parity epsilon result partition)**.** *For every history m : BHist, scalar action by the parity epsilon history has exactly the following result partition:*

$$(\mathsf{hsame}(m, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps}, m), \mathsf{ModuleParityOne})) \quad \vee \quad ((\mathsf{hsame}(m, \mathsf{e}) \Rightarrow \mathsf{False}) \wedge \mathsf{hsa}$$

Proof. Perform constructor case analysis on m. At $m = \mathsf{e}$, the action reduces to ModuleParityOne, giving the left side of the partition. At a visible zero-headed or one-headed history, the action reduces to e, while constructor disjointness gives the nonempty premise of the right side. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul epsilon result partition`

Theorem 46.36 (Module parity epsilon visible-one iff empty)**.** *For every history m : BHist,*

$$\mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps}, m), \mathsf{ModuleParityOne}) \iff \mathsf{hsame}(m, \mathsf{e}).$$

Its ten-field fit is: traditional target, deterministic parity-action branch readback; BEDC source, BHist *with* ModuleParitySmul*; carrier, all histories; classifier,* hsame*; ledger, the epsilon scalar and the input constructor case; core theorem, this block; exactness, visible-one output iff empty input; standard bridge, the two-point parity toggle; status,* ProofTarget*.*

Proof. Perform constructor case analysis on m.

For $m = \mathsf{e}$, scalar action by ModuleParityEps reduces to ModuleParityOne. The forward direction and the reverse direction are therefore both reflexive after the reduction.

For a zero-headed visible history, the action reduces to e. The forward direction would identify e with ModuleParityOne, contradicting empty-vs-visible-one no-confusion. The reverse direction starts from an impossible visible-vs-empty premise, so it follows by eliminating that premise.

For a one-headed visible history, the action again reduces to e. The same empty-vs-visible-one no-confusion rejects the forward direction, and the reverse direction follows by eliminating the impossible visible-vs-empty premise. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul epsilon visible one input empty iff`

Theorem 46.37 (Module parity epsilon boolean orbit involution)**.** *For every history m : BHist, if m is history-same to either* e *or* ModuleParityOne*, then two successive scalar actions by* ModuleParityEps *return m:*

$$(\mathsf{hsame}(m, \mathsf{e}) \vee \mathsf{hsame}(m, \mathsf{ModuleParityOne})) \implies \mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps}, \mathsf{ModuleParitySmul}(\mathsf{Mod}$$

Proof. Split the displayed endpoint assumption. In the empty endpoint branch, replace m by e and compute the two epsilon actions: the first reaches $\mathsf{ModuleParityOne}$ and the second returns to e. In the visible-one branch, replace m by $\mathsf{ModuleParityOne}$ and compute the two epsilon actions in the opposite order, ending at the same visible-one endpoint. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul epsilon boolean orbit involutive`

Theorem 46.38 (Module parity epsilon multiplication boolean orbit involution)**.** *For every history r : BHist, if r is history-same to either e or $\mathsf{ModuleParityEps}$, then two successive parity multiplications by $\mathsf{ModuleParityEps}$ return r:*

$$(\mathsf{hsame}(r,\mathsf{e})\vee\mathsf{hsame}(r,\mathsf{ModuleParityEps})) \Longrightarrow \mathsf{hsame}(\mathsf{ModuleParityMul}(\mathsf{ModuleParityEps},\mathsf{ModuleParityMul}(\mathsf{ModulePa$$

Proof. Split the displayed endpoint assumption. In the empty endpoint branch, replace r by e and compute the two epsilon multiplications. In the epsilon endpoint branch, replace r by $\mathsf{ModuleParityEps}$ and compute the same two multiplications. Each branch closes by reflexivity of the displayed endpoint. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParityMul epsilon boolean orbit involutive`

Theorem 46.39 (Module parity epsilon empty endpoint has no fixed point)**.** *For every history m : BHist,*

$$\mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{E0}(\mathsf{e}),m),m) \Longrightarrow \mathsf{False}.$$

Proof. Perform constructor case analysis on m. At $m = \mathsf{e}$ the action produces $\mathsf{E1}(\mathsf{e})$, contradicting visible-one versus empty no-confusion. At $m = \mathsf{E0}(h)$ and $m = \mathsf{E1}(h)$ the action produces e, contradicting the corresponding empty versus visible no-confusion law. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul e0 empty no fixed point`

Theorem 46.40 (Module parity epsilon double action empty iff)**.** *For every history m : BHist,*

$$\mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps},\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps},m)),\mathsf{e}) \Longleftrightarrow \mathsf{hsame}(m,\mathsf{e}).$$

Proof. Apply Definition 46.35 to the first epsilon action on m. In the empty-input branch, the first action is $\mathsf{ModuleParityOne}$, so the second epsilon action reduces to e. This gives the forward endpoint and the reverse endpoint by the same empty branch witness.

In the nonempty branch, the first action is e, so the second epsilon action reduces to $\mathsf{ModuleParityOne}$. Visible-one versus empty no-confusion rules out the left side of the displayed equivalence. Conversely, an assumed $\mathsf{hsame}(m,\mathsf{e})$ contradicts the nonempty premise supplied by the partition, so this branch contributes no empty-input witness. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleParitySmul epsilon double action empty iff`

Theorem 46.41 (Module parity epsilon double action one iff)**.** *For every history m : BHist,*

$$\mathsf{hsame}(\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps},\mathsf{ModuleParitySmul}(\mathsf{ModuleParityEps},m)),\mathsf{ModuleParityOne}) \Longleftrightarrow (\mathsf{hsame}$$

Proof. Perform constructor case analysis on m. At $m = \mathsf{e}$, the double epsilon action lands at e, so equality with $\mathsf{ModuleParityOne}$ is ruled out by empty versus visible-one separation, matching the false nonempty predicate. At visible inputs, the first epsilon action lands at e and the second lands at $\mathsf{ModuleParityOne}$; the reverse direction uses reflexivity, and the forward direction rules out an empty input by constructor separation. □

Lean4 ✓ BEDC.Derived.ModuleUp.ModuleParitySmul epsilon double action one iff

Theorem 46.42 (Module scalar negation gives two-sided action inverse)**.** *Let a* $\mathsf{Module}^{\uparrow}(R, M)$ *certificate be given by Definition 46.20. Write* $\sim_R$ *for the scalar-ring classifier,* $\sim_M$ *for the module classifier,* $+_R$ *and* 0_R *for scalar addition and scalar zero,* $-_R$ *for scalar additive inverse,* $+_M$ *and* 0_M *for module addition and module zero, and* $\odot_M$ *for scalar action. For every carried scalar* $r : R$ *and carried module endpoint* $m : M$*,*

$$((-_R r) \odot_M m) +_M (r \odot_M m) \sim_M 0_M$$

and

$$(r \odot_M m) +_M ((-_R r) \odot_M m) \sim_M 0_M.$$

Lean4 ✓ BEDC.Derived.ModuleUp.ModuleSingleton scalar negation action additive inverse

Proof. By Definition 46.15, the scalar source of the module certificate is a certified $\mathsf{Ring}^{\uparrow}$. Its additive component, listed in Definition 43.26 and Definition 41.55, supplies inverse closure and the scalar additive left-inverse row. Thus $-_R r$ is carried and

$$(-_R r) +_R r \sim_R 0_R.$$

Classifier reflexivity for the carried endpoint m comes from Definition 9.7. Scalar-action congruence from Definition 46.18 transports the scalar inverse comparison to

$$((-_R r) +_R r) \odot_M m \sim_M 0_R \odot_M m.$$

Scalar distributivity over scalar addition from Definition 46.18 gives

$$((-_R r) +_R r) \odot_M m \sim_M ((-_R r) \odot_M m) +_M (r \odot_M m).$$

Using classifier symmetry from Definition 9.7, transitivity from Definition 9.8, and the zero-action theorem Definition 46.29, compose these comparisons to obtain

$$((-_R r) \odot_M m) +_M (r \odot_M m) \sim_M 0_M.$$

For the other order, Definition 43.27, read in the same scalar source, supplies

$$r +_R (-_R r) \sim_R 0_R.$$

Scalar-action congruence transports this comparison to

$$(r +_R (-_R r)) \odot_M m \sim_M 0_R \odot_M m,$$

and scalar distributivity over scalar addition gives

$$(r +_R (-_R r)) \odot_M m \sim_M (r \odot_M m) +_M ((-_R r) \odot_M m).$$

The same symmetry, transitivity, and zero-action composition gives

$$(r \odot_M m) +_M ((-_R r) \odot_M m) \sim_M 0_M.$$

□

Definition 46.43 (Module LinearMap kernel predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source, and let*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

hold. Write $\sim_N$ for the classifier on N and 0_N for its module zero. The kernel predicate of f is the zero fiber

$$\mathsf{Ker}_f(x) :\Longleftrightarrow x \text{ is carried in } M \wedge f(x) \sim_N 0_N.$$

The witness records source carrier evidence for x together with the target-classifier comparison from $f(x)$ to the target zero.

Theorem 46.44 (Module LinearMap kernel submodule closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the source and target module classifiers, $+_M$ for source addition, 0_M and 0_N for source and target zero, and $\odot_M$ and $\odot_N$ for the scalar actions. If*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

holds, then the predicate Ker_f from Definition 46.43 is stable under source-classifier transport, contains 0_M, and is closed under source addition and carried scalar action:

$$\mathsf{Ker}_f(x) \wedge x \sim_M x' \Longrightarrow \mathsf{Ker}_f(x'),$$

$$\mathsf{Ker}_f(0_M),$$

$$\mathsf{Ker}_f(x) \wedge \mathsf{Ker}_f(u) \Longrightarrow \mathsf{Ker}_f(x +_M u),$$

and, for every carried scalar $r : R$,

$$\mathsf{Ker}_f(x) \Longrightarrow \mathsf{Ker}_f(r \odot_M x).$$

In particular, the source-ring inverse closure makes $-_R r$ carried whenever r is carried, so $\mathsf{Ker}_f(x)$ entails $\mathsf{Ker}_f((-_R r) \odot_M x)$.

The ten-field fit is: traditional target the ordinary kernel submodule of an R-linear map; BEDC source the displayed $\mathsf{LinearMapCert}_R$ package over the two $\mathsf{Module}^{\uparrow}$ certificates; carrier the zero-fiber predicate Ker_f; classifier the source module classifier restricted to that predicate; stability source-classifier transport, zero, addition, scalar action, and scalar-negation closure; ledger the function identifier f, its descent, zero, additivity, and scalar rows, plus the source and target module ledgers; core theorem this block; exactness predicate the condition $f(x) \sim_N 0_N$ used by downstream first-isomorphism-style kernel exactness; standard bridge the usual kernel submodule construction; status ProofTarget.

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul kernel submodule closure`

Proof. Unpack Definition 48.59. It supplies source-classifier descent for f, additivity, scalar compatibility, and zero preservation. Unpack the two module certificates through Definition 46.20. The source module supplies the carrier rows for 0_M, $x +_M u$, and $r \odot_M x$; the target module supplies operation congruence and classifier comparison rows through Definition 46.18.

For source-classifier transport, assume $\mathsf{Ker}_f(x)$ and $x \sim_M x'$. Source carrier transport gives that x' is carried. The descent row gives a target comparison between $f(x)$ and $f(x')$. Compose it with the zero-fiber comparison $f(x) \sim_N 0_N$ by the target classifier equivalence laws and Definition 9.8. This yields $f(x') \sim_N 0_N$, so x' remains in the kernel predicate.

For the zero row, source zero closure carries 0_M. The zero-preservation row of Definition 48.59 gives

$$f(0_M) \sim_N 0_N.$$

Thus 0_M satisfies the defining zero-fiber predicate.

For addition, choose kernel witnesses for x and u. Source additive closure carries $x +_M u$. Additivity gives

$$f(x +_M u) \sim_N f(x) +_N f(u).$$

Target additive congruence transports $f(x) \sim_N 0_N$ and $f(u) \sim_N 0_N$ to a comparison with $0_N +_N 0_N$. The target additive zero rows reduce that endpoint to 0_N, and Definition 9.8 composes the comparisons. Therefore $x +_M u$ lies in Ker_f.

For scalar action, let r be a carried scalar and choose a kernel witness for x. Source scalar-action closure carries $r \odot_M x$. Scalar compatibility gives

$$f(r \odot_M x) \sim_N r \odot_N f(x).$$

Target scalar congruence transports $f(x) \sim_N 0_N$ to $r \odot_N f(x) \sim_N r \odot_N 0_N$. The zero-action theorem Definition 46.29 gives $r \odot_N 0_N \sim_N 0_N$, and transitivity gives the required zero-fiber comparison.

Finally, the scalar source of the module certificate is a certified $\mathsf{Ring}^{\uparrow}$ by Definition 46.15, so the additive inverse row carries $-_R r$. Applying the scalar-action closure row to this carried scalar gives $\mathsf{Ker}_f((-_R r) \odot_M x)$. The endpoint interpretation is exactly Definition 46.42: $(-_R r) \odot_M x$ is the additive inverse-side action endpoint for $r \odot_M x$ inside the source module classifier. □

Theorem 46.45 (Module LinearMap kernel certificate package)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Let*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

hold. Write $\sim_M$ and $\sim_N$ for the source and target module classifiers, $+_M$ and 0_M for source addition and source zero, $\odot_M$ for source scalar action, 0_N for target zero, and Ker_f for the predicate of Definition 46.43. On the kernel carrier define the restricted classifier by

$$x \sim_{\mathsf{Ker}_f} y :\Longleftrightarrow \mathsf{Ker}_f(x) \wedge \mathsf{Ker}_f(y) \wedge x \sim_M y.$$

Then the kernel data form a reusable certificate package with the following fields:

1. *$\sim_{\mathsf{Ker}_f}$ is reflexive, symmetric, and transitive on carried kernel endpoints.*
2. *Source-module classifier transport carries kernel witnesses along $x \sim_M x'$.*
3. *0_M is a carried kernel endpoint, and carried kernel endpoints are closed under $+_M$ and under carried scalar action $\odot_M$.*
4. *The inclusion of the kernel carrier into M reflects the source classifier: for kernel endpoints, a comparison in M repacks as a comparison for $\sim_{\mathsf{Ker}_f}$, and every restricted comparison projects to the underlying $x \sim_M y$ row.*
5. *Every carried kernel endpoint satisfies the exactness row $f(x) \sim_N 0_N$.*

The ten-field fit is: traditional target kernel submodule certificate; BEDC source $\mathsf{LinearMapCert}_R$ over two $\mathsf{Module}^{\uparrow}$ certificates; carrier Ker_f; classifier the restriction of $\sim_M$; stability inherited from Definition 46.44; ledger the function identifier f together with descent, zero, additivity, and scalar rows from Definition 48.59; core theorem this package; exactness the zero-fiber row $f(x) \sim_N 0_N$; standard bridge the ordinary kernel submodule; status $\mathsf{ProofTarget}$.

Proof. The carrier rows are exactly the rows established in Definition 46.44. They give source-classifier transport for Ker_f, the kernel witness for 0_M, additive closure, and carried scalar-action closure.

For the restricted classifier laws, work only with endpoints already carrying kernel witnesses. Reflexivity follows from the source module NameCert reflexivity row applied to the underlying carried endpoint, then paired with the two kernel witnesses. Symmetry keeps the two kernel witnesses and reverses the source comparison by the symmetry row of Definition 9.7.

For transitivity, take restricted comparisons from x to y and from y to z. Their projections give $x \sim_M y$ and $y \sim_M z$. Apply Definition 9.8 in the source module classifier to obtain $x \sim_M z$, and pair this comparison with the kernel witnesses for x and z.

The inclusion statement is definitional. A restricted comparison contains its underlying $x \sim_M y$ component by the definition of $\sim_{\mathsf{Ker}_f}$. Conversely, for two kernel endpoints, any source comparison $x \sim_M y$ repacks with the two endpoint witnesses to produce $x \sim_{\mathsf{Ker}_f} y$.

It remains to identify the exactness row. By Definition 46.43, a kernel witness is precisely source-carrier evidence together with the zero-fiber comparison $f(x) \sim_N 0_N$. The descent row from Definition 48.59, composed with Definition 9.8 in the target module classifier, is the compatibility used when this zero-fiber comparison is transported along source comparisons. Thus the restricted classifier, inclusion reflection, operation closure, and zero-fiber exactness sit in one certificate package. □

Theorem 46.46 (Module singleton scalar-action kernel exactness)**.** *For the concrete singleton empty-history module of section 46.2, fix the terminal scalar* e *and define the scalar-action zero-fiber predicate*

$$K(h) :\Longleftrightarrow A_{\mathsf{e}}(h) \wedge \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(\mathsf{e}, h), \mathsf{e}).$$

Then, for every history $h : \mathsf{BHist}$*,*

$$K(h) \Longleftrightarrow A_{\mathsf{e}}(h),$$

and, for all histories $h, k : \mathsf{BHist}$*, the restricted classifier on two kernel endpoints is exactly the singleton classifier row:*

$$K(h) \wedge K(k) \wedge \rho_{\mathsf{e}}(h, k) \Longleftrightarrow \rho_{\mathsf{e}}(h, k).$$

The ten-field fit is: traditional target terminal module kernel; BEDC source BHist *with* $\mathsf{ModuleSingletonSmul}$*; carrier* A_{e}*; classifier* ρ_{e} *restricted to* K*; stability inherited from singleton classifier transport and scalar-action laws; ledger the terminal scalar* e *and the zero-fiber witness; core theorem this block; exactness* $K(h)$ *agrees with* $A_{\mathsf{e}}(h)$*; standard bridge the kernel of the terminal endomorphism; status* $\mathsf{ProofTarget}$*.*

Proof. Unfold the displayed predicate K. The forward direction projects the first component, so a zero-fiber witness immediately gives $A_{\mathsf{e}}(h)$.

For the reverse direction, assume $A_{\mathsf{e}}(h)$. The scalar action $\mathsf{ModuleSingletonSmul}(\mathsf{e}, h)$ is definitionally the empty history. Theorem 46.3 packages the empty endpoint comparison as $\rho_{\mathsf{e}}(h, \mathsf{e})$, and Theorem 46.5 reads this classification through the singleton scalar action. This supplies $\rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(\mathsf{e}, h), \mathsf{e})$, which repacks with the given carrier witness as $K(h)$.

It remains to compare the restricted classifier. One direction is projection: from $K(h) \wedge K(k) \wedge \rho_{\mathsf{e}}(h, k)$ retain the third component.

Conversely, assume $\rho_{\mathsf{e}}(h, k)$. By Theorem 46.3, both endpoints lie in the singleton carrier A_{e}. Applying the first part to these two carrier witnesses gives $K(h)$ and $K(k)$, so the restricted classifier repacks as $K(h) \wedge K(k) \wedge \rho_{\mathsf{e}}(h, k)$. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul kernel exactness`

Theorem 46.47 (Module singleton scalar-action kernel semantic certificate)**.** *For the concrete singleton empty-history module of section 46.2, fix the terminal scalar* e *and let*

$$K(h) :\Longleftrightarrow A_{\mathsf{e}}(h) \wedge \rho_{\mathsf{e}}(\mathsf{ModuleSingletonSmul}(\mathsf{e}, h), \mathsf{e}).$$

Define the restricted kernel classifier by

$$\rho_K(h, k) :\Longleftrightarrow K(h) \wedge K(k) \wedge \rho_{\mathsf{e}}(h, k).$$

Then source, pattern, and ledger all equal to K*, together with classifier* ρ_K*, form the singleton scalar-action kernel semantic naming certificate:*

$$\mathsf{SemanticNameCert}(K, K, K, \rho_K).$$

Proof. Definition 46.46 identifies the displayed kernel predicate with the singleton carrier A_{e}. This gives the empty endpoint as an inhabitant of K and converts each carried kernel endpoint to the carrier context used by Definition 46.14.

For the restricted classifier core, reflexivity at a kernel endpoint h uses the singleton classifier reflexivity row at the converted carrier witness for h, then repacks the two copies of the original kernel witness with $\rho_{\mathsf{e}}(h, h)$. Symmetry projects a comparison $\rho_K(h, k)$ to $\rho_{\mathsf{e}}(h, k)$, applies the symmetry row from Definition 46.14, and repacks the kernel witnesses in the reversed order.

For transitivity, project the two restricted comparisons to $\rho_{\mathsf{e}}(h, k)$ and $\rho_{\mathsf{e}}(k, l)$. Apply Definition 9.8 in the singleton classifier to obtain $\rho_{\mathsf{e}}(h, l)$, and pair that comparison with the retained kernel witnesses for h and l.

Endpoint witnesses are repacked through Definition 46.3: every underlying singleton comparison supplies the carrier witnesses needed to return to K by Definition 46.46. Conversely, a restricted comparison already contains its two K witnesses and its underlying ρ_{e} row.

Pattern and ledger soundness are identity maps, since source, pattern, and ledger are the same predicate K. These rows assemble the displayed $\mathsf{SemanticNameCert}(K, K, K, \rho_K)$. □

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleSingletonSmul kernel semanticNameCert`

Theorem 46.48 (Module singleton standard bridge)**.** *The concrete singleton BEDC module instance of Definition 46.2, with carrier* A_{e}*, classifier* ρ_{e}*, constant scalar action* ModuleSingletonSmul*, and law package Definition 46.14, interprets as the ordinary terminal one-element module over the terminal one-element ring. Its public interpretation is the standard one-object module bridge: carrier membership, classifier comparison, scalar action, source rows, graph exactness, and ledger rows are all read through the singleton* $\mathsf{Ring}^{\uparrow}$ *laws, the singleton* $\mathsf{AbGroup}^{\uparrow}$ *laws, the scalar-action image* SemanticNameCert *of Definition 46.7, and the graph exactness theorem Definition 46.8.*

Lean4 ✓ `BEDC.Derived.ModuleUp.ModuleUp StdBridge`

Proof. The scalar carrier is the singleton empty-history ring supplied by Definition 43.6. The additive module carrier is the singleton empty-history abelian group supplied by Definition 42.10. Both use the same A_{e} endpoint predicate and the same ρ_{e} classifier on BHist.

The scalar-action law package Definition 46.14 identifies the displayed constant action ModuleSingletonSmul with the unique action possible on those singleton carriers. The image certificate Definition 46.7 records the carrier, classifier, source, pattern, and ledger rows as a BEDC SemanticNameCert.

Finally, graph exactness Definition 46.8 says that the action graph has exactly the singleton endpoint output. Composing these rows gives the usual terminal module over the usual terminal ring, with no extra quotient or host representative added to the public interpretation. □

Theorem 46.49 (Module singleton public surface has no host leak)**.** *Every exported public row of the singleton* $\mathsf{Module}^{\uparrow}$ *surface is expressed only in* BHist*,* A_{e}*,* ρ_{e}*,* hsame*,* $\mathsf{ModuleSingletonSmul}$*, the singleton* $\mathsf{Ring}^{\uparrow}$ *rows, the singleton* $\mathsf{AbGroup}^{\uparrow}$ *rows, and the fields of the displayed* $\mathsf{SemanticNameCert}$ *packages. The public surface exports no host equality, host* Nat*, quotient carrier, or host function as part of the BEDC module interface.*

Proof. The carrier and classifier rows are the empty-history endpoint predicate A_{e} and the classifier ρ_{e} from Definition 46.2. Endpoint transport is the hsame transport used in the singleton classifier and image-certificate rows.

The operation row is the displayed BEDC scalar action $\mathsf{ModuleSingletonSmul}$. Its semantic certificate is Definition 46.7, and its graph boundary is Definition 46.8; neither row requires a host equality test or a host function in the exported surface.

The scalar and additive source rows are exactly the singleton $\mathsf{Ring}^{\uparrow}$ and $\mathsf{AbGroup}^{\uparrow}$ rows, and the kernel-facing public rows are the BEDC $\mathsf{SemanticNameCert}$ and exactness rows of Definition 46.46 and Definition 46.47. Thus every public endpoint, comparison, operation, and ledger field is named inside the BEDC vocabulary, while host Nat, quotient carriers, host equality, and host functions remain outside the exported interface. □

Theorem 46.50 (ModuleUp mature package boundary)**.** *The concrete* $\mathsf{Module}^{\uparrow}$ *package for the singleton empty-history instance consists of the singleton scalar carrier, the singleton module carrier, the scalar-action image* $\mathsf{SemanticNameCert}$*, the output-fiber* $\mathsf{SemanticNameCert}$*, the zero-action annihilation row, the* $\mathsf{Ring}^{\uparrow}$ *scalar projection, the* $\mathsf{LinMap}^{\uparrow}$ *kernel package, the classifier-faithful action and annihilator* $\mathsf{Ideal}^{\uparrow}$ *predicate, the singleton standard bridge, and the no-host-leak public boundary.*

Proof. The singleton scalar and module carriers, scalar action, scalar projection, semantic image certificate, and output-fiber certificate are the concrete rows of Definition 46.2, Definition 46.14, Definition 46.7, and Definition 46.10. The zero-action endpoint row is supplied by Definition 46.29.

The kernel-facing part of the package is the $\mathsf{LinMap}^{\uparrow}$ semantic certificate of Definition 46.47. The action-faithfulness and annihilator closure rows are packaged by Definition 46.33.

Finally, Definition 46.48 identifies the public singleton surface with the standard terminal module bridge, and Definition 46.49 records that this public surface exports no host equality, host Nat, quotient carrier, or host function. These rows are exactly the advertised mature package boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$**.**

Constructive story: A module action here is a constant action between two endpoint carriers: the scalar side is the singleton from $\mathsf{Ring}^{\uparrow}$, the vector side is the singleton from $\mathsf{AbGroup}^{\uparrow}$, and the action sends every admitted scalar-vector pair back to e. A pair (s, v) is admitted only when both entries are history-equal to e, and two pairs are classified by the conjunction of the scalar and vector singleton classifiers. The source is the singleton-on-singleton pair predicate, the pattern records the scalar-action image certificate, graph exactness, zero-action annihilation, and the equivalence between action faithfulness and a trivial zero annihilator, while the ledger keeps the visible pair aligned with the raw endpoint data. Since both sides are singleton name certificates, (e, e) supplies the carrier witness, the pair classifier is reflexive, symmetric, and transitive, and classifier-recognized pairs remain admitted. A pair (s, v) lies in the signature gap of a $\mathsf{Module}^{\uparrow}$ token p when the source-domain accepts the pair, the signature relation pairs it with an answer pair and evidence ledger, and the token-introduction relation names that answer as p; unfolded, both s and v are history-equal to e and p is the singleton scalar-action token. The annihilator-ideal predicate, singleton standard bridge to host module data, and no-host-leak theorem all read back the same endpoint action. The chapter registers this singleton scalar-action construction under the name $\mathsf{Module}^{\uparrow}$; the ring, abelian group, associativity, two distributivity laws, and unit law are checked on the constant endpoint action, with no bases, quotients, tensor products, or host module object introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature singleton Module$^{\uparrow}$ paper package binds the singleton scalar carrier, singleton module carrier, scalar-action image SemanticNameCert, graph exactness, zero-action annihilation, action-faithfulness if and only if zero-annihilator, the annihilator Ideal$^{\uparrow}$ predicate, the singleton standard bridge of Definition 46.48, and the no-host-leak theorem of Definition 46.49.

Formal verification: FormalStatus(NameCert$_{\text{Module}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ModuleUp.ModuleSingletonSmul\_image\_semanticNameCert`

Bridge status: bridgeChecked

Origin: human

Not claimed: The chapter does not claim arbitrary non-singleton modules, quotient modules, bases, free modules, tensor products, or structure theorem results.

Upgrade path: The singleton package occupies the stated mature theory boundary and the top formal target; broader module packages require separate certificates.

Chapter 47

A Concrete Naming Certificate for $\mathsf{VecSpace}^{\uparrow}$

The vector space interface $\mathsf{VecSpace}^{\uparrow}$ is the specialization of chapter 46 to the case where the scalar ring satisfies chapter 45. No new fields are required beyond the module ones; the change is the strengthened scalar source. The n-fold rational tuple space (an iterated product over $\mathsf{Rat}^{\uparrow}$) is the canonical concrete instance.

47.1 Carrier definition

Definition 47.1 (VecSpace carrier)**.** *Use a module setup whose scalar source is a certified field.*

47.2 Concrete singleton history vector space instance

Definition 47.2 (Concrete singleton empty-history vector space instance)**.** *Let the scalar carrier and vector carrier both be the singleton history source A_{e} of Definition 46.2. Define the shared classifier*

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Use the scalar addition, scalar negation, scalar multiplication, scalar zero, scalar one, nonzero predicate, and proof-indexed scalar inverse from Definition 45.2. Use the vector addition, vector negation, vector zero, and scalar action from Definition 46.2, with

$$\mathsf{smul}(r,m) := \mathsf{e}.$$

The certificate fit has ten fields:

1. *Traditional target: the one-element vector space over the one-element proof-indexed field.*
2. *BEDC source:* BHist.
3. *Carrier choice: A_{e} for both scalars and vectors.*
4. *Classifier choice: ρ_{e}.*
5. *Stability obligations: the singleton field laws together with the singleton module laws and the vacuous nonzero-indexed scalar inverse reasoning.*
6. *Ledger: singleton scalar, vector, and action provenance.*

7. *Core theorem list: Theorem 47.3.*
8. *Exactness predicate: singleton endpoint classification.*
9. *Standard bridge: the terminal one-object vector space.*
10. *Status:* ProofTarget.

Theorem 47.3 (Singleton empty-history vector space laws)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{add}, 0, \mathsf{neg}, \mathsf{mul}, 1, \mathsf{smul}, \mathsf{NonZero}_{\perp}, \mathsf{inv})$$

satisfies the $\mathsf{VecSpace}^{\uparrow}$ *schema over the concrete* BHist *singleton carrier: scalar* $\mathsf{Field}^{\uparrow}$ *laws, additive module laws, scalar-action closure, scalar-action congruence, scalar associativity, distributivity over vector addition, distributivity over scalar addition, scalar unit law, and inverse-enabled scalar reasoning for explicitly nonzero scalars.*

Proof. The module part is exactly the singleton empty-history module package of Theorem 46.14. It supplies closure, the additive abelian-group laws, scalar-action congruence, scalar associativity, both distributivity laws, and the scalar unit law for the same A_{e} carrier and ρ_{e} classifier.

The scalar field part is exactly the singleton empty-history field schema of Theorem 45.37. It supplies the scalar commutative-ring laws together with the proof-indexed nonzero transport, inverse closure, inverse congruence, and inverse laws. The certificate packaging is the one recorded in Theorem 45.81.

It remains only to check that the two packages share the displayed scalar source. Both use the constant empty-history scalar operations on A_{e}, the same classifier ρ_{e}, and the same scalar unit. Hence their scalar ring fragments coincide as the scalar side of the vector-space schema.

For scalar congruence and each scalar-action law, every endpoint reduces to e. The required classifier witness is assembled from $A_{\mathsf{e}}(\mathsf{e})$ and $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

For inverse-enabled scalar reasoning, every such clause begins with a witness $p : \mathsf{NonZero}_{\perp}(h)$. Theorem 45.33 identifies the singleton nonzero predicate as contradictory at the only possible endpoint. Since $\mathsf{NonZero}_{\perp}(h)$ is $\perp$, eliminating p closes the clause over the empty witness domain. No classical nonzero test or further endpoint datum is needed. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.singleton empty history vecspace laws`

Theorem 47.4 (Singleton empty-history vector space endpoint collapse)**.** *With the notation of Definition 47.2, let* $r, s : \mathsf{BHist}$ *be scalar endpoints and* $m, n : \mathsf{BHist}$ *be vector endpoints carried by* A_{e}*. Then the concrete singleton tuple has the following endpoint classifications:*

1. *For any* $x, y : \mathsf{BHist}$*, if* $A_{\mathsf{e}}(x)$ *and* $A_{\mathsf{e}}(y)$*, then* $\rho_{\mathsf{e}}(x, y)$*.*
2. $A_{\mathsf{e}}(\mathsf{smul}(r, m))$ *and* $\rho_{\mathsf{e}}(\mathsf{smul}(r, m), \mathsf{e})$*.*
3. *The vector endpoints* $\mathsf{add}(m, n)$*,* $\mathsf{neg}(m)$*, and the vector zero are all carried by* A_{e} *and classified with* e*.*
4. *The scalar endpoints* 0*,* 1*, and* $\mathsf{mul}(r, s)$ *are all carried by* A_{e} *and classified with* e*.*
5. *For any witness* $p : \mathsf{NonZero}_{\perp}(r)$*, the inverse endpoint* $\mathsf{inv}(r, p)$ *is carried by* A_{e} *and classified with* e*.*

Proof. Apply Theorem 46.14 to the carried scalar and vector inputs. Its closure and congruence clauses cover the additive vector endpoints and the scalar-action endpoint for the same A_{e} carrier and ρ_{e} classifier.

Apply Theorem 45.37 to the scalar side. It supplies closure for scalar zero, scalar one, scalar multiplication, and the proof-indexed inverse clauses over the same singleton source.

Unfold the concrete operations in Definition 47.2. The displayed scalar action is definitionally e, and the imported singleton vector and scalar operations reduce to the singleton endpoint as recorded by the module and field law packages.

For any two carried endpoints x and y, unfold $\rho_{\mathsf{e}}(x, y)$. The carrier components are the two assumptions $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$, and the singleton endpoint witnesses for x and y compose to give the required $\mathsf{hsame}(x, y)$ component.

Each operation endpoint already reduces to e or is closed by the cited law package at the singleton endpoint. Pair the relevant $A_{\mathsf{e}}(\mathsf{e})$ witness with reflexivity $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$ to obtain the displayed classifier witness against e.

For the inverse-indexed clause, assume $p : \mathsf{NonZero}_{\perp}(r)$. Since $A_{\mathsf{e}}(r)$ holds, Theorem 45.33 eliminates the nonzero witness at the only possible endpoint. From that contradiction, the carried and classified inverse conclusion follows over the empty witness domain. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonEmptyHistory laws`

Theorem 47.5 (VecSpace singleton classifier endpoint transitivity)**.** *For singleton vector endpoints* $a, b, c : \mathsf{BHist}$*, if*

$$\mathsf{VecSpaceSingletonClassifier}(a, b) \quad \textit{and} \quad \mathsf{VecSpaceSingletonClassifier}(b, c),$$

then

$$a\mathsf{hsame}e \wedge c\mathsf{hsame}e \wedge \mathsf{VecSpaceSingletonClassifier}(a, c).$$

Proof. Unfold the concrete singleton classifier. The first classifier supplies the empty endpoint witness for a and the comparison $a\mathsf{hsame}b$. The second classifier supplies the empty endpoint witness for c and the comparison $b\mathsf{hsame}c$. Compose the two comparisons by transitivity to obtain $a\mathsf{hsame}c$, and package it with the carried endpoint witnesses for a and c. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonClassifier endpoint trans`

Theorem 47.6 (VecSpace singleton scalar action carrier readback)**.** *For the concrete singleton* $\mathsf{VecSpace}^{\uparrow}$ *instance, if* $m : \mathsf{BHist}$ *satisfies* $A_{\mathsf{e}}(m)$*, then* $\mathsf{smul}(r, m)$ *is classified with* m*, and* $\mathsf{smul}(r, m)\mathsf{hsame}e$*.*

Proof. The scalar action in Definition 47.2 is the constant empty-history action, so its endpoint has the singleton carrier by $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. The assumed carrier witness for m is the second endpoint component. Symmetry of that same witness gives $\mathsf{ehsame}m$, which is the classifier comparison from the scalar-action endpoint back to m. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul carrier readback`

Theorem 47.7 (VecSpace singleton scalar action classifier empty equivalence)**.** *For all histories* $r, m, t : \mathsf{BHist}$*, the singleton scalar action classifier satisfies*

$$\mathsf{VecSpaceSingletonClassifier}(\mathsf{VecSpaceSingletonSmul}(r, m), t) \iff t\mathsf{hsame}e.$$

Proof. The scalar action endpoint is definitionally e. In the forward direction, unfolding the classifier exposes the carried target endpoint, which is exactly $t\mathsf{hsame}e$. Conversely, from $t\mathsf{hsame}e$, use reflexivity for the source empty endpoint, the displayed target carrier, and symmetry to obtain the classifier comparison $\mathsf{ehsame}t$. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul classifier empty iff`

Theorem 47.8 (VecSpace singleton scalar action image pullback)**.** *For all histories* r, m, n : BHist*, if* $m$hsamee*, then*

$$\mathsf{VecSpaceSingletonClassifier}(\mathsf{VecSpaceSingletonSmul}(r, m), n) \iff \mathsf{VecSpaceSingletonClassifier}(m, n).$$

Proof. The scalar-action image endpoint is definitionally e. In the forward direction, the target classifier gives the carrier of n and the comparison ehsame$n$; composing the assumed $m$hsamee with this comparison gives $m$hsame$n$. In the reverse direction, the target carrier contained in VecSpaceSingletonClassifier(m, n) and symmetry classify the empty scalar-action endpoint against n. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul image pullback iff`

Theorem 47.9 (VecSpace singleton scalar action target deterministic)**.** *For all histories* r, s, m, n, x, y : BHist*, if* VecSpaceSingletonClassifier(VecSpaceSingletonSmul$(r, m), x$) *and* VecSpaceSingletonClassifier(VecSpaceSing *then* $x$hsamee*,* $y$hsamee*, and* $x$hsame$y$*.*

Proof. Unfold the singleton scalar-action classifier at both displayed endpoints. The target-carrier field of the first classifier gives $x$hsamee, and the target-carrier field of the second classifier gives $y$hsamee. Composing the first equality with the symmetry of the second yields $x$hsame$y$. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul target deterministic`

Theorem 47.10 (VecSpace singleton scalar action inverse law)**.** *For the concrete singleton* VecSpace$^{\uparrow}$ *instance, let* p : FieldSingletonNonZero(r) *be an explicit scalar inverse witness and let* m : BHist *satisfy* $A_{\mathsf{e}}(m)$*. Then* smul(r, m) *is classified with* e*, and applying the proof-indexed inverse scalar to that result is classified with the original vector endpoint* m*.*

Proof. The scalar action is the constant empty-history action, so smul(r, m)hsamee by reflexivity. Applying the inverse scalar again has the same empty-history endpoint. The first classifier package uses $A_{\mathsf{e}}(\mathsf{e})$ on both sides. The second uses $A_{\mathsf{e}}(\mathsf{e})$ on the action side and the assumed singleton-carrier witness for m on the readback side; symmetry of that witness gives ehsame$m$. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingleton scalar action inverse law`

Theorem 47.11 (VecSpace singleton scalar action inverse readback classifier iff)**.** *For the concrete singleton* VecSpace$^{\uparrow}$ *instance, let* p : FieldSingletonNonZero(r) *and let* m *satisfy* $A_{\mathsf{e}}(m)$*. Then the inverse-indexed scalar action after scalar action has empty readback, and classification against any target* n *is equivalent to classification of the original vector endpoint* m *against* n*:*

$$\mathsf{VecSpaceSingletonClassifier}(\mathsf{VecSpaceSingletonSmul}(\mathsf{FieldSingletonInv}(r, p), \mathsf{VecSpaceSingletonSmul}(r, m)), n) \iff \mathsf{V}$$

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul inverse readback classifier iff`

Proof. The singleton scalar action is the constant empty-history endpoint, so the double action has carrier equality to e by reflexivity. In the forward classifier direction, the target carrier and action-target equality combine with the assumed carrier equality $m$hsamee to give VecSpaceSingletonClassifier(m, n). In the reverse direction, the target carrier contained in VecSpaceSingletonClassifier(m, n), together with symmetry, classifies the empty double-action endpoint against n. □

Theorem 47.12 (VecSpace singleton scalar action empty readback classifier)**.** *For the concrete singleton* $\mathsf{VecSpace}^{\uparrow}$ *instance, if* $m : \mathsf{BHist}$ *satisfies* $A_{\mathsf{e}}(m)$*, then* $\mathsf{smul}(r, m)$ *is classified with* e *and is also classified with the original vector endpoint* m*.*

Proof. The scalar action is definitionally the constant empty-history action. The empty endpoint classifier is therefore the reflexive classifier on e. For the readback classifier, the left endpoint again supplies $A_{\mathsf{e}}(\mathsf{e})$, the right endpoint is the assumed carrier witness $A_{\mathsf{e}}(m)$, and symmetry of that witness gives $\mathsf{ehsame}m$. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul empty readback classifier`

Theorem 47.13 (VecSpace singleton scalar action empty-result readback)**.** *For the concrete singleton* $\mathsf{VecSpace}^{\uparrow}$ *instance, let* $r, m, n : \mathsf{BHist}$*. If* m *is carried by* A_{e} *and* $\mathsf{smul}(r, m)$ *is classified with* n*, then* $\mathsf{smul}(r, m)\mathsf{hsamee}$ *and* m *is classified with* n*.*

Proof. Unfolding the singleton scalar action gives $\mathsf{smul}(r, m) = \mathsf{e}$, so the first classifier endpoint supplies $\mathsf{smul}(r, m)\mathsf{hsamee}$. The classifier hypothesis also contains the carrier witness for n and the endpoint equality $\mathsf{smul}(r, m)\mathsf{hsame}n$. Compose the assumed carrier equality $m\mathsf{hsamee}$ with this action equality to obtain $m\mathsf{hsame}n$, and pair it with the two carrier witnesses to recover the classifier $\rho_{\mathsf{e}}(m, n)$. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul empty result readback`

The $\mathsf{VecSpace}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and scalar action, so Lean formalization can target BHist-anchored vector-space laws in $\mathsf{BEDC.Derived.VecSpaceUp}$.

Theorem 47.14 (Singleton empty-history vector space semantic certificate)**.** *The tuple from Definition 47.2, together with the law package of Theorem 47.3, supplies the concrete paper-side semantic certificate for the terminal* $\mathsf{VecSpace}^{\uparrow}$ *over the singleton* $\mathsf{Field}^{\uparrow}$ *and singleton* $\mathsf{Module}^{\uparrow}$ *carriers. Its certificate target is the following ten-field template:*

1. *Traditional target: the terminal vector space.*
2. *BEDC source:* BHist*.*
3. *Carrier:* A_{e}*.*
4. *Classifier:* ρ_{e}*.*
5. *Stability: the singleton field and module law packages together with nonzero-witness elimination.*
6. *Ledger: singleton scalar, vector, and action provenance.*
7. *Core theorem: Theorem 47.3.*
8. *Exactness: singleton endpoint classification.*
9. *Standard bridge: the terminal vector space.*
10. *Status:* $\mathsf{ProofTarget}$*.*

Proof. The scalar component is supplied by Theorem 45.37, and its paper-side packaging is recorded by Theorem 45.81. These give the singleton $\mathsf{Field}^{\uparrow}$ source, the shared A_{e} carrier, and the classifier ρ_{e} used by the displayed tuple.

The vector component is supplied by Theorem 46.14. It gives the singleton $\mathsf{Module}^{\uparrow}$ additive laws, scalar-action closure, scalar-action congruence, associativity, both distributivity laws, and the scalar unit law over the same empty-history endpoint.

The remaining inverse-indexed clauses are vacuous in the singleton field source. By Theorem 45.33, any witness of $\mathsf{NonZero}_\perp(h)$ is impossible at the only endpoint, so each nonzero-indexed scalar obligation is closed by elimination.

Thus the tuple already carries the scalar, vector, classifier, ledger, exactness, bridge, and status data listed in the ten-field template, with Theorem 47.3 as its core theorem. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingleton semanticNameCert`

Theorem 47.15 (VecSpace singleton scalar action image semantic certificate)**.** *For the concrete singleton* $\mathsf{VecSpace}^\uparrow$ *instance, the source carrier* A_e*, the scalar-action image pattern*

$$P(h) \;:\Longleftrightarrow\; \exists r, m : \mathsf{BHist},\ A_\mathsf{e}(m) \wedge h\mathsf{hsameVecSpaceSingletonSmul}(r, m),$$

the ledger carrier A_e*, and the singleton classifier* ρ_e *form a* $\mathsf{SemanticNameCert}$*.*

Proof. The core certificate is the singleton A_e certificate for ρ_e: inhabitation is witnessed by e, reflexivity uses the carried endpoint, symmetry reverses the hsame comparison, transitivity composes comparisons, and carrier transport reads the target carrier field from the classifier.

For pattern soundness, assume $A_\mathsf{e}(h)$. Choose the displayed scalar $r = \mathsf{e}$ and vector $m = h$. The vector side is carried by the same witness, and the scalar action is definitionally the empty-history endpoint, so the same $A_\mathsf{e}(h)$ witness is the required comparison $h\mathsf{hsameVecSpaceSingletonSmul}(r, m)$. Ledger soundness is the identity map on the concrete singleton carrier. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingletonSmul image semanticNameCert`

47.3 The certificate

Definition 47.16 (VecSpace source specification)**.** *The source is* $\mathsf{Field}^\uparrow(F)$ *together with a certified* F*-module carrier.*

Definition 47.17 (VecSpace pattern specification)**.** *The pattern is exactly module expression formation over field scalars.*

Definition 47.18 (VecSpace classifier specification)**.** *The classifier is the module carrier classifier, with scalar congruence inherited from the field.*

Definition 47.19 (VecSpace stability certificate)**.** *The stability certificate consists of:*

1. *all module stability fields*
2. *all scalar field stability fields*
3. *inverse-enabled scalar reasoning for apart-from-zero scalars*

Definition 47.20 (VecSpace ledger policy)**.** *Record module ledgers plus the field ledger for scalar operations.*

Definition 47.21 ($\mathsf{VecSpace}^\uparrow$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{VecSpace}^\uparrow$*.*

Primitive boundary. $\mathsf{VecSpace}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 47.22 (Zero-axiom encoding note)**.** *Inverses use field apartness witnesses rather than LEM-based nonzero tests.*

Proposition 47.23 (VecSpace forgets to Module certificate)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let R be the $\mathsf{CommRing}^{\uparrow}$ base coordinate contained in that field certificate. Let V carry a $\mathsf{VecSpace}^{\uparrow}(F, V)$ certificate with scalar action $\mathsf{smul} : F \times V \to V$. If the constructive apartness predicate, the proof-indexed scalar inverse operation, and every inverse-indexed stability or ledger row are omitted, then the retained carrier, source, pattern, classifier, stability, and ledger fields form a $\mathsf{Module}^{\uparrow}(R, V)$ certificate, with R read through its ring reduct. The vector carrier, scalar action, additive vector operations, scalar ring operations, classifier, exactness predicate, and all non-inverse ledger provenance are unchanged.*

Proof. By Definition 47.21, the given vector-space certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{VecSpace}^{\uparrow}$. By Definition 47.1, its carrier is already a module setup whose scalar source is strengthened from a scalar ring to a certified field. Therefore the vector carrier, additive vector operations, scalar action, and classifier are the module-shaped carrier data listed in Definition 46.1.

The scalar source is handled by projection of the field side. By Definition 45.198, the $\mathsf{Field}^{\uparrow}(F)$ source contains a certified commutative-ring carrier together with constructive nonzero evidence. Omitting the nonzero and inverse-specific coordinates leaves the $\mathsf{CommRing}^{\uparrow}$ certificate R. By Definition 44.8, this same scalar coordinate supplies the $\mathsf{Ring}^{\uparrow}$ source required by Definition 46.15. Together with the additive vector carrier already present in the module setup, this gives the source field for $\mathsf{Module}^{\uparrow}(R, V)$.

For patterns, Definition 47.17 says that the vector-space pattern is module expression formation over field scalars. Removing scalar inverse formation leaves expressions built from vector addition, vector zero, vector negation, scalar multiplication, and the scalar ring operations. This is exactly the module expression formation of Definition 46.16. For classifiers, Definition 47.18 uses the module carrier classifier, with scalar congruence inherited from the field. After the scalar source is read through the commutative-ring and ring reducts, the same classifier is the one required by Definition 46.17; no endpoint relation is replaced.

For stability, Definition 47.19 contains all module stability fields, all scalar field stability fields, and inverse-enabled scalar reasoning. The first component is precisely the scalar-action and additive-vector stability package of Definition 46.18. The scalar ring laws needed there are present because Definition 45.201 contains all commutative-ring stability fields, and Definition 44.8 retains the ring part. The inverse closure, nonzero transport, and inverse laws are not obligations for $\mathsf{Module}^{\uparrow}$.

For the ledger, Definition 47.20 records the module ledgers together with the field ledger for scalar operations. By Definition 45.336, the field ledger consists of the ring ledger plus the nonzero witness used for each inverse. Omitting those inverse-witness entries leaves the scalar and module expression provenance described in Definition 46.19.

Thus the retained carrier, source, pattern, classifier, stability, and ledger fields are exactly the obligations named in Definition 46.20. Since every retained operation and classifier is copied from the original $\mathsf{VecSpace}^{\uparrow}(F, V)$ certificate, the resulting $\mathsf{Module}^{\uparrow}(R, V)$ certificate has the same carrier, action, classifier, stability rows, and ledger outside the omitted inverse-specific data. □

Proposition 47.24 (VecSpace module-fragment projection)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate. Let F_{Ring} be the scalar certificate obtained from it by first taking the commutative-ring reduct and then the ring reduct. Let V carry a $\mathsf{VecSpace}^{\uparrow}(F, V)$ certificate with vector carrier C_V, vector classifier $\sim_V$, additive operations $+_V$, 0_V, $-_V$, and scalar action $\odot : F \times V \to V$. If the apartness predicate, proof-indexed scalar inverse operation, inverse-indexed stability rows, and inverse-witness ledger rows are omitted, the retained fields form a $\mathsf{Module}^{\uparrow}(F_{\mathsf{Ring}}, V)$ certificate. The vector carrier C_V, vector classifier $\sim_V$, vector addition, vector zero, vector negation, scalar action $\odot$, scalar addition, scalar zero, scalar negation, scalar multiplication, scalar one, exactness predicate, and non-inverse ledger rows are unchanged.*

Proof. By Definition 47.21, the vector-space certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{VecSpace}^{\uparrow}$. By Definition 47.1, its carrier is already a module setup whose scalar source has been strengthened from a certified ring source to a certified field source. Thus the vector carrier, additive vector operations, scalar action, and vector classifier are exactly the module-shaped carrier data required by Definition 46.1.

The scalar source is obtained by successive reducts. Applying Definition 45.342 to the $\mathsf{Field}^{\uparrow}(F)$ scalar certificate keeps the same scalar carrier, classifier, addition, zero, negation, multiplication, and one while dropping the apartness and proof-indexed inverse rows. Applying Definition 44.8 to that commutative-ring coordinate keeps the same scalar operations and classifier and supplies the $\mathsf{Ring}^{\uparrow}$ source required by Definition 46.15. This scalar source is F_{Ring}.

For patterns, Definition 47.17 identifies vector-space expression formation with module expression formation over field scalars. After inverse formation on the scalar side is omitted, the remaining vector expressions are generated by vector addition, vector zero, vector negation, scalar multiplication, and scalar ring expressions. This is exactly the pattern listed in Definition 46.16.

For classifiers, Definition 47.18 uses the module carrier classifier, with scalar congruence inherited from the field. The scalar reducts above preserve the scalar classifier, and the vector classifier is not replaced. Hence the retained classifier is the module carrier classifier required by Definition 46.17.

For stability, Definition 47.19 contains all module stability fields, all scalar field stability fields, and inverse-enabled scalar reasoning. The module rows are precisely the scalar-action and additive-vector rows listed in Definition 46.18. The scalar ring rows needed there are retained because Definition 45.201 contains the commutative-ring stability fields and Definition 44.8 keeps the ring part. The inverse closure, nonzero transport, and inverse laws are not obligations for $\mathsf{Module}^{\uparrow}$.

For the ledger, Definition 47.20 records module ledgers together with the field ledger for scalar operations. By Definition 45.336, the field ledger consists of the ring ledger plus inverse-witness entries. Omitting those witness entries leaves the scalar and module expression provenance described in Definition 46.19.

Thus the retained carrier, source, pattern, classifier, stability, and ledger fields satisfy Definition 46.20. Every retained operation, carrier, classifier, exactness predicate, and non-inverse ledger row is copied from the original $\mathsf{VecSpace}^{\uparrow}(F, V)$ certificate, so the projected $\mathsf{Module}^{\uparrow}(F_{\mathsf{Ring}}, V)$ certificate has the same module fragment. □

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingleton module fragment projection`

Corollary 47.25 (VecSpace additive carrier projection)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate. Let F_{Ring} be the scalar certificate obtained from it by first taking the commutative-ring reduct and then the ring reduct. Let V carry a $\mathsf{VecSpace}^{\uparrow}(F, V)$ certificate with vector carrier C_V, vector classifier $\sim_V$, vector addition $+_V$, vector zero 0_V, vector negation $(-)_V$, and scalar action $\odot : F \times V \to V$. If the vector-space certificate is first read through the module-fragment projection of Definition 47.24,*

and the resulting module certificate is then read through the additive reduct of Definition 46.26, the retained vector-side data form an $\mathsf{AbGroup}^{\uparrow}(V)$ *certificate. The carrier* C_V*, classifier* $\sim_V$*, vector addition* $+_V$*, vector zero* 0_V*, and vector negation* $(-)_V$ *are unchanged from the original* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *certificate.*

Lean4 ✓ `BEDC.Derived.VecSpaceUp.VecSpaceSingleton additive carrier projection`

Proof. Apply Definition 47.24 to the given vector-space certificate. It supplies a $\mathsf{Module}^{\uparrow}(F_{\mathsf{Ring}}, V)$ certificate after deleting only the apartness predicate, proof-indexed scalar inverse operation, inverse-indexed stability rows, and inverse-witness ledger rows. The proposition states that this module certificate keeps the original vector carrier C_V, vector classifier $\sim_V$, vector addition, vector zero, vector negation, and scalar action.

Now apply Definition 46.26 to that $\mathsf{Module}^{\uparrow}(F_{\mathsf{Ring}}, V)$ certificate. Its scalar input is available because F_{Ring} is a certified $\mathsf{Ring}^{\uparrow}$ source. This second reduct removes the scalar-ring package, the scalar action, and the scalar-action stability rows, and retains exactly the additive carrier, additive source, additive pattern, carrier classifier, additive stability fields, and additive ledger.

By Definition 42.32, those retained fields are the obligations for an $\mathsf{AbGroup}^{\uparrow}(V)$ certificate. The first projection did not alter the vector-side additive data, and the second projection explicitly keeps the additive carrier data of the module certificate. Therefore the final $\mathsf{AbGroup}^{\uparrow}(V)$ certificate is over the same carrier C_V with the same classifier $\sim_V$, addition $+_V$, zero 0_V, and additive inverse $(-)_V$ carried by the original vector-space certificate. □

Theorem 47.26 (Finite linear-combination induction route)**.** *Let* F *carry a* $\mathsf{Field}^{\uparrow}(F)$ *certificate and let* V *carry a* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *certificate. Suppose an accepted finite vector-space read carries a support ledger generated by the empty support row and by one-step insertion of a displayed basis-vector row with its scalar coefficient. Then every exported linear-combination row in that read is obtained by induction over the displayed finite support list. The induction preserves the* $\mathsf{VecSpace}^{\uparrow}$ *carrier, the* $\mathsf{Module}^{\uparrow}$ *reduct of Definition 47.24, the* $\mathsf{AbGroup}^{\uparrow}$ *additive projection of Definition 47.25, scalar-action readback, componentwise transport, displayed replay, provenance, and local* NameCert *rows.*

Proof. Induct on the displayed finite support ledger. In the empty-support case, the row is the vector-zero row already present in the $\mathsf{VecSpace}^{\uparrow}$ carrier; its module reduct and additive projection are the zero rows retained by Definition 47.24 and Definition 47.25.

For the one-step insertion case, the induction hypothesis supplies the linear-combination row for the previous support list together with its carrier, reduct, additive, scalar-action, transport, replay, provenance, and local naming invariants. The inserted basis-vector row and coefficient consume one scalar-vector action row of the $\mathsf{VecSpace}^{\uparrow}$ certificate, producing the next displayed summand without changing the underlying vector carrier or classifier.

The additive step then uses the additive carrier projection to add this summand to the previous row. Since the module-fragment projection keeps the scalar action and the additive projection keeps the vector addition, the same readback path exports the enlarged linear-combination row.

Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row are structural rows of the original vector-space read, so they preserve the induction invariant at each ledger step. Thus every exported finite linear-combination row factors through the displayed support list and no basis package beyond the finite ledger is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$**.**

Constructive story: The vector-space package keeps the singleton scalar-vector pair from $\mathsf{Module}^{\uparrow}$ and changes the scalar certificate from $\mathsf{Ring}^{\uparrow}$ to $\mathsf{Field}^{\uparrow}$. A pair (s, v) enters only when both entries are history-equal to e, the action again

sends every carried pair to e, and the pair classifier now uses the Field$^{\uparrow}$ classifier on the scalar side together with the singleton vector classifier. The source is the singleton-pair predicate, the pattern records scalar action, the projection back to Module$^{\uparrow}$, the AbGroup$^{\uparrow}$ additive carrier projection, and the scalar-action image semantic certificate, while the ledger keeps the visible pair aligned with the raw endpoint pair. Because both sides are singleton certificates, (e, e) witnesses nonemptiness, the pair classifier is reflexive, symmetric, and transitive, and recognized scalar-vector pairs remain admitted. A pair (s, v) lies in the signature gap of a VecSpace$^{\uparrow}$ token p when the source-domain accepts the pair, the signature relation pairs it with an answer pair and evidence ledger, and the token-introduction relation names that answer as p; unfolded, both entries are history-equal to e and p is the singleton vector-space token. The terminal one-object bridge data row reads this as the one-object external vector space, while inverse-indexed scalar reasoning is vacuous because the singleton has no nonzero witness domain to inspect. The chapter registers this terminal singleton package under the name VecSpace$^{\uparrow}$; the carrier is the same endpoint pair under field-scalar discipline, with no finite-dimensional coordinate theory, basis package, analytic vector space, or non-singleton vector-space claim introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{VecSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Singleton empty-history vector-space package binding the carrier A_{e}, classifier ρ_{e}, scalar action, semantic NameCert, scalar-action image semantic certificate, terminal one-object vector-space bridge data, Module$^{\uparrow}$ projection, and AbGroup$^{\uparrow}$ additive carrier projection.

Formal verification: FormalStatus(NameCert$_{\mathsf{VecSpace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.VecSpaceUp.VecSpaceSingleton\_semanticNameCert`

Bridge status: paperBridge

Not claimed: Arbitrary vector spaces, finite-dimensional spaces over Rat$^{\uparrow}$ or Real$^{\uparrow}$, bases outside the singleton terminal case, spectral theory, Banach/Hilbert spaces, and Lean-side bridge verification are not closed by this mature singleton package.

Upgrade path: The remaining verification-axis step is bridgeChecked on a checked bridge target or a separate non-singleton vector-space family treatment.

Chapter 48

A Concrete Naming Certificate for $\mathsf{LinMap}^{\uparrow}$

The linear map interface $\mathsf{LinMap}^{\uparrow}$ has certificate obligations between two vector spaces over the same scalar field from chapter 47. The carrier is the predicate "$f : V \to W$ preserves the additive structure and commutes with scalar multiplication"; the classifier is pointwise carrier sameness on the codomain restricted to maps satisfying linearity. It is not a primitive function space; it is admitted only as a derived predicate over function pairs.

48.1 Carrier definition

Definition 48.1 (LinearMap carrier)**.** *A concrete* $\mathsf{LinMap}^{\uparrow}$ *carrier is a* BHist*-coded map-code predicate together with classifier, evaluation, and composition data. For the singleton empty-history instance, the public map-code carrier is*

$$L_{\mathsf{e}}(f) := \mathsf{hsame}(f, \mathsf{e}),$$

the pointwise classifier is

$$\lambda_{\mathsf{e}}(f, g) := L_{\mathsf{e}}(f) \wedge L_{\mathsf{e}}(g) \wedge \mathsf{hsame}(f, g),$$

the evaluator is

$$\mathsf{ev}_{\mathsf{e}}(f, x) := \mathsf{e},$$

and the composition operation is

$$\mathsf{comp}_{\mathsf{e}}(g, f) := \mathsf{e}.$$

The required closure, congruence, additivity, scalar compatibility, identity, zero-map, and composition laws are supplied by Theorem 48.3.

Lean4 (definition) `BEDC.Derived.LinearMapUp.LinearMapSingletonCarrier`
Lean4 (definition) `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier`
Lean4 (definition) `BEDC.Derived.LinearMapUp.LinearMapSingletonEval`
Lean4 (definition) `BEDC.Derived.LinearMapUp.LinearMapSingletonComp`

48.2 Concrete singleton empty-history linear map instance

Definition 48.2 (Concrete singleton empty-history linear map instance)**.** *Use the singleton empty-history vector space of Definition 47.2 as both source and target. Its scalar, source-vector, and target-vector carriers are the singleton history source A_{e} of Definition 36.169 at $a_0 = \mathsf{e}$. The map-code carrier is the same existing singleton predicate*

$$A_{\mathsf{e}} : \mathsf{BHist} \to \mathsf{Prop},$$

and the map-code classifier is the existing singleton classifier ρ_{e}. The realized action is the constant endpoint action

$$\Lambda_{\mathsf{e}}(f, x) := \mathsf{e}$$

for every admitted map code f and singleton vector input x. The same endpoint action may be displayed as

$$F_{\mathsf{e}}(h) := \mathsf{e}.$$

The linear-map carrier on public map codes and its pointwise map classifier are

$$L_{\mathsf{e}}(f) := \mathsf{hsame}(f, \mathsf{e}), \qquad \lambda_{\mathsf{e}}(f, g) := L_{\mathsf{e}}(f) \wedge L_{\mathsf{e}}(g) \wedge \mathsf{hsame}(f, g).$$

The pointwise evaluation is

$$\mathsf{ev}_{\mathsf{e}}(f, h) := F_{\mathsf{e}}(h).$$

The identity, zero-map, and composition code are

$$\mathsf{id}_{\mathsf{e}} := \mathsf{e}, \qquad \mathsf{zeroMap}_{\mathsf{e}} := \mathsf{e}, \qquad \mathsf{comp}_{\mathsf{e}}(g, f) := \mathsf{e}.$$

This certificate has ten fields:

1. *Traditional target: the terminal one-map linear map space between the singleton vector space and itself.*
2. *BEDC source:* BHist.
3. *Carrier choice: A_{e} for source vectors, target vectors, and map codes.*
4. *Classifier choice: ρ_{e} for endpoint and map-code equality.*
5. *Stability obligations: output closure, pointwise classifier congruence, additivity, scalar compatibility, zero preservation, identity closure, and composition closure.*
6. *Ledger: singleton map-code provenance for F_{e}, id_{e}, $\mathsf{zeroMap}_{\mathsf{e}}$, and $\mathsf{comp}_{\mathsf{e}}$.*
7. *Core theorem list: Theorem 48.3.*
8. *Exactness predicate: singleton endpoint classification.*
9. *Standard bridge: the terminal one-map linear hom space.*
10. *Status:* $\mathsf{ProofTarget}$.

Theorem 48.3 (Singleton empty-history linear map laws)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \Lambda_{\mathsf{e}}, \mathsf{e}, \mathsf{comp}_{\mathsf{e}})$$

satisfies the $\mathsf{LinMap}^{\uparrow}$ schema over the singleton vector space, where the displayed empty-history code supplies the identity and zero-map slots.

Proof. The source and target vector-space laws are those of Theorem 47.3. They supply the singleton scalar laws, vector closure, additive laws, scalar-action closure, and scalar-action congruence for the same A_{e} carrier and ρ_{e} classifier.

Output closure is $A_{\mathsf{e}}(\mathsf{e})$, since $\Lambda_{\mathsf{e}}(f, x)$ is definitionally e. Pointwise classifier congruence is immediate because $\Lambda_{\mathsf{e}}(f, x)$ and $\Lambda_{\mathsf{e}}(g, y)$ are both e, so the required endpoint comparison is the singleton ρ_{e} witness.

For additivity, $\Lambda_{\mathsf{e}}(f, \mathsf{add}(x, y))$ and $\mathsf{add}(\Lambda_{\mathsf{e}}(f, x), \Lambda_{\mathsf{e}}(f, y))$ are both classified with e. The vector-space theorem supplies closure for the displayed addition endpoint, and the singleton classifier supplies the endpoint comparison.

For scalar compatibility, $\Lambda_{\mathsf{e}}(f, \mathsf{smul}(r, x))$ and $\mathsf{smul}(r, \Lambda_{\mathsf{e}}(f, x))$ are again classified with e. Scalar-action closure and congruence come from the singleton vector-space laws, so no additional map-specific datum is needed.

Zero preservation compares $\Lambda_{\mathsf{e}}(f, 0)$ with the singleton vector zero. Both sides are e, and the required classification is the same ρ_{e} endpoint witness.

The identity and zero-map codes are e, and their action on any carried input x reduces to e. The input carrier witness $A_{\mathsf{e}}(x)$ together with ρ_{e} gives the required singleton endpoint classification.

For composition closure, every composite map code and every realized output reduce to e. The carrier witness for e and reflexivity inside ρ_{e} close the composition law and the comparison of the two evaluation orders. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton empty history laws`
Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton laws`

Theorem 48.4 (Singleton empty-history linear map public-code exactness)**.** *For the concrete instance of Definition 48.2, for all public map codes f, g and vector code x, if $L_{\mathsf{e}}(f)$, $L_{\mathsf{e}}(g)$, and $A_{\mathsf{e}}(x)$, then*

$$\lambda_{\mathsf{e}}(f, \mathsf{id}_{\mathsf{e}}) \wedge \lambda_{\mathsf{e}}(\mathsf{zeroMap}_{\mathsf{e}}, \mathsf{id}_{\mathsf{e}}) \wedge \lambda_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(g, f), \mathsf{id}_{\mathsf{e}}) \wedge \rho_{\mathsf{e}}(\mathsf{ev}_{\mathsf{e}}(f, x), x).$$

Its per-object fit is: traditional target terminal one-map hom exactness; BEDC source BHist *over the singleton vector-space instance; carrier A_{e} for vectors and map codes; classifier ρ_{e} and λ_{e}; stability obligations public-code exactness, evaluation exactness, and composition collapse; ledger singleton map-code provenance; core theorem this block; exactness predicate the displayed classifier collapse; standard bridge terminal linear hom; status* ProofTarget.

Proof. The three public map-code endpoints id_{e}, $\mathsf{zeroMap}_{\mathsf{e}}$, and $\mathsf{comp}_{\mathsf{e}}(g, f)$ reduce to e. The hypotheses $L_{\mathsf{e}}(f)$ and $L_{\mathsf{e}}(g)$ give the nontrivial singleton witnesses for the two displayed input map codes.

The classifier component of each map-code comparison is $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Since both advertised comparison partners reduce to the empty history, the reflexive singleton classifier closes the identity, zero-map, and composite-code clauses.

For evaluation, $\mathsf{ev}_{\mathsf{e}}(f, x)$ is $F_{\mathsf{e}}(x)$ and therefore reduces to e. The carrier hypothesis $A_{\mathsf{e}}(x)$ supplies the endpoint witness for the input vector, and ρ_{e} supplies the reflexive comparison between the realized endpoint and x in the singleton carrier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton public empty code exactness`

Theorem 48.5 (Singleton linear-map classifier append split empty)**.** *For all histories p, q, h :* BHist*,*

$$\lambda_{\mathsf{e}}(\mathsf{append}(p, q), h)$$

holds exactly when $L_{\mathsf{e}}(p)$, $L_{\mathsf{e}}(q)$, and $L_{\mathsf{e}}(h)$ all hold.

Proof. The forward direction reads the first carrier field of $\lambda_{\mathsf{e}}(\mathsf{append}(p,q),h)$ as $\mathsf{hsame}(\mathsf{append}(p,q),\mathsf{e})$. Empty-result inversion for BEDC append splits this into $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$, while the second classifier field gives $h\mathsf{hsamee}$.

Conversely, the three singleton carrier witnesses show that $\mathsf{append}(p,q)$ is empty and that h is empty. These are the two carrier fields of λ_{e}, and composing the empty endpoint comparison with the symmetry of the h-carrier gives the classifier comparison between $\mathsf{append}(p,q)$ and h. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append split empty iff`

Theorem 48.6 (Singleton linear-map classifier visible right append exclusion)**.** *For all histories $p,q,h : \mathsf{BHist}$, neither appending a visible right component to p can be classified by the singleton linear-map classifier:*

$$\lambda_{\mathsf{e}}(\mathsf{append}(p,\mathsf{E0}(q)),h) \Longrightarrow \bot, \qquad \lambda_{\mathsf{e}}(\mathsf{append}(p,\mathsf{E1}(q)),h) \Longrightarrow \bot.$$

Proof. Unfold the classifier and read its left carrier field. The field says that $\mathsf{append}(p,\mathsf{E0}(q))$ or $\mathsf{append}(p,\mathsf{E1}(q))$ is history-same to e. Empty-append inversion would then force the visible right component itself to be e, contradicting constructor no-confusion for $\mathsf{E0}$ and $\mathsf{E1}$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append visible left absurd`

Theorem 48.7 (Singleton linear-map classifier visible prefix exclusion)**.** *For all histories $p,q,h : \mathsf{BHist}$, neither visible prefix can be classified after append by the singleton linear-map classifier:*

$$\lambda_{\mathsf{e}}(\mathsf{append}(\mathsf{E0}(p),q),h) \Longrightarrow \bot, \qquad \lambda_{\mathsf{e}}(\mathsf{append}(\mathsf{E1}(p),q),h) \Longrightarrow \bot.$$

Proof. The singleton classifier requires its left argument to be empty. Applying empty-append inversion to this carrier clause forces the visible prefix $\mathsf{E0}(p)$ or $\mathsf{E1}(p)$ to be e, which is impossible by constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append visible prefix absurd`

Theorem 48.8 (Singleton linear-map classifier append-pair carrier exactness)**.** *For all histories $p,q,r,s : \mathsf{BHist}$,*

$$\lambda_{\mathsf{e}}(\mathsf{append}(p,q),\mathsf{append}(r,s))$$

holds exactly when all four endpoint histories lie in the singleton carrier:

$$L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(s).$$

Proof. The forward direction reads the two carrier fields of the displayed classifier. Empty-append inversion splits $\mathsf{append}(p,q)\mathsf{hsamee}$ into the two endpoint carrier witnesses for p and q, and it splits $\mathsf{append}(r,s)\mathsf{hsamee}$ into the two endpoint carrier witnesses for r and s.

Conversely, the four singleton carrier witnesses reconstruct empty witnesses for both appended histories. These are the two carrier fields of λ_{e}, and transitivity with symmetry of the right appended empty witness supplies the classifier comparison between the two appended histories. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append pair carrier iff`

Theorem 48.9 (Singleton linear-map classifier append pair empty)**.** *For all histories $p,q,r,s : \mathsf{BHist}$,*

$$\lambda_{\mathsf{e}}(\mathsf{append}(p,q),\mathsf{append}(r,s)) \Longleftrightarrow L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(s).$$

Proof. The forward direction unfolds the singleton classifier. Its two carrier clauses classify both appended histories with e, and empty-append inversion splits those clauses into the four endpoint carrier witnesses.

Conversely, the four singleton carrier witnesses assemble the two appended histories as empty histories. These become the two carrier clauses of λ_{e}, and composing the left appended empty witness with symmetry of the right one gives the classifier comparison between the appended endpoints. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append pair empty iff`

Theorem 48.10 (Singleton linear-map classifier continuation closure)**.** *For histories* P, P', Q, Q', L, R : BHist*, if* $\lambda_{\mathsf{e}}(P, P')$ *and* $\lambda_{\mathsf{e}}(Q, Q')$*, and if*

$$\mathsf{Cont}(P, Q, L) \qquad \textit{and} \qquad \mathsf{Cont}(P', Q', R),$$

then $\lambda_{\mathsf{e}}(L, R)$*.*

Proof. The two singleton linear-map classifier witnesses identify all four continuation inputs with e. Transport each continuation ledger to the empty-empty ledger and compare against $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. This gives $L\mathsf{hsamee}$ and $R\mathsf{hsamee}$, and those carrier clauses compose through symmetry on the right endpoint to close $\lambda_{\mathsf{e}}(L, R)$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation closed`

Theorem 48.11 (Singleton linear-map continuation pair factors)**.** *For histories* p, q, r, s, t, u : BHist*, if*

$$\mathsf{Cont}(p, q, r) \qquad \textit{and} \qquad \mathsf{Cont}(s, t, u),$$

then

$$\lambda_{\mathsf{e}}(r, u) \iff L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(s) \wedge L_{\mathsf{e}}(t).$$

Proof. Read both continuation ledgers as append endpoints. The singleton linear-map classifier between those endpoints is exactly the append-pair classifier exactness theorem, which splits the two empty append endpoints into the four displayed singleton carrier factors and rebuilds the classifier from them. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation pair factors iff`

Theorem 48.12 (Singleton linear-map continuation append-source classifier)**.** *For histories* p, q, r, h, t : BHist*, if* $\mathsf{Cont}(\mathsf{append}(p, q), r, h)$*, then*

$$\lambda_{\mathsf{e}}(h, t) \iff L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(t).$$

Proof. The continuation displays h as $\mathsf{append}(\mathsf{append}(p, q), r)$. The forward direction splits the left classifier carrier through empty-append inversion twice, exposing p, q, r; the right classifier carrier gives t. Conversely, the three source carriers rebuild the nested empty append carrier, and symmetry on the target carrier closes the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation append source classifier iff`

Theorem 48.13 (Singleton linear-map continuation append-pair classifier)**.** *For histories* p, q, r, s, h, t : BHist*, if* $\mathsf{Cont}(\mathsf{append}(p, q), \mathsf{append}(r, s), h)$*, then*

$$\lambda_{\mathsf{e}}(h, t) \iff L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(s) \wedge L_{\mathsf{e}}(t).$$

Proof. The continuation displays h as the append of the two appended endpoints. The forward direction reads the left carrier field of the classifier and splits that empty append once for the two displayed endpoints, then once more inside each endpoint to expose the four source carrier witnesses; the target carrier field supplies t. Conversely, the four source carrier witnesses rebuild both displayed endpoints as empty histories, then rebuild their continuation result as empty, and symmetry on the target carrier completes the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation append pair classifier iff`

Theorem 48.14 (Singleton linear-map append-source visible result exclusion)**.** *For histories* p, q, r, s : BHist*, if* p, q, r *lie in the singleton linear-map carrier, then no continuation from the appended source* $\mathsf{append}(p, q)$ *through* r *can have either visible result:*

$$\mathsf{Cont}(\mathsf{append}(p, q), r, \mathsf{E0}(s)) \Longrightarrow \bot, \qquad \mathsf{Cont}(\mathsf{append}(p, q), r, \mathsf{E1}(s)) \Longrightarrow \bot.$$

Proof. The three singleton carrier witnesses identify p, q, and r with e. The displayed continuation therefore reads the visible result as the append of three empty histories, hence as e. Constructor no-confusion for $\mathsf{E0}$ and $\mathsf{E1}$ rules out both visible endpoints. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonCarrier continuation append source visible result absurd`

Theorem 48.15 (Singleton linear-map continuation append-target classifier)**.** *For histories* p, q, r, h, t : BHist*, if* $\mathsf{Cont}(p, \mathsf{append}(q, r), h)$*, then*

$$\lambda_{\mathsf{e}}(h, t) \iff L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(t).$$

Proof. The continuation displays h as $\mathsf{append}(p, \mathsf{append}(q, r))$. In the forward direction, empty-append inversion first exposes p and the right appended endpoint, then splits that endpoint into q and r; the target carrier field supplies t. In the reverse direction, the q and r carriers rebuild the right append, the p carrier rebuilds the continuation result carrier, and the target carrier completes the classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation append target classifier iff`

Theorem 48.16 (Singleton linear-map left continuation closure)**.** *For histories* P, P', Q, L : BHist*, if* $\lambda_{\mathsf{e}}(P, P')$*,* $L_{\mathsf{e}}(Q)$*, and* $\mathsf{Cont}(P, Q, L)$*, then* $L_{\mathsf{e}}(L)$ *and* $\lambda_{\mathsf{e}}(L, \mathsf{e})$*.*

Proof. The classifier witness identifies P with e, while the singleton carrier identifies Q with e. Transporting the continuation ledger to $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ gives $L\mathsf{hsamee}$, which is the desired singleton carrier and the left endpoint of the classifier against e. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation left closed`

Theorem 48.17 (Singleton linear-map right continuation closure)**.** *For histories* P, Q, Q', R : BHist*, if* $L_{\mathsf{e}}(P)$*,* $\lambda_{\mathsf{e}}(Q, Q')$*, and* $\mathsf{Cont}(P, Q, R)$*, then* $L_{\mathsf{e}}(R)$ *and* $\lambda_{\mathsf{e}}(R, \mathsf{e})$*.*

Proof. The singleton carrier identifies the left continuation input P with e, and the classifier witness identifies the right continuation input Q with e. Transporting $\mathsf{Cont}(P, Q, R)$ to the empty-empty continuation ledger therefore gives $R\mathsf{hsamee}$. This is the singleton carrier for R, and with the reflexive carrier of e it packages the classifier $\lambda_{\mathsf{e}}(R, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation right closed`

Theorem 48.18 (Singleton linear map composition associativity empty classifier)**.** *For singleton linear map codes f, g, h, if each code lies in L_{e}, then both associations of the composite code classify with e, and the two associated composite codes classify with each other:*

$$\lambda_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(h, \mathsf{comp}_{\mathsf{e}}(g, f)), \mathsf{e}) \wedge \lambda_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(h, g), f), \mathsf{e}) \wedge \lambda_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(h, \mathsf{comp}_{\mathsf{e}}(g, f)), \mathsf{comp}_{\mathsf{e}}(\mathsf{comp}_{\mathsf{e}}(h, g), f)).$$

Proof. The singleton composition code is definitionally e for every pair of inputs. Therefore both associated composites reduce to e, and each displayed classifier is the reflexive singleton map-code classifier at the empty history. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton comp assoc empty classifier`

Theorem 48.19 (Singleton linear-map classifier append left cancellation)**.** *For all $P, Q, R : \mathsf{BHist}$, if $L_{\mathsf{e}}(P)$, then*

$$\lambda_{\mathsf{e}}(\mathsf{append}(P, Q), \mathsf{append}(P, R)) \iff \lambda_{\mathsf{e}}(Q, R).$$

Proof. For the forward direction, unfold the singleton classifier. The carrier clauses say that both appended histories are classified with e. Applying the empty-append inversion to each carrier clause gives $Q\mathsf{hsamee}$ and $R\mathsf{hsamee}$. These two endpoint witnesses compose, with symmetry on the right one, to give $Q\mathsf{hsame}R$.

For the reverse direction, the hypothesis $L_{\mathsf{e}}(P)$ identifies the left prefix with e. The two appended histories therefore reduce through the left-empty continuation law to Q and R. The carrier clauses and classifier comparison supplied by $\lambda_{\mathsf{e}}(Q, R)$ transport along those reductions to rebuild the prefixed singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append left cancel iff`

Theorem 48.20 (Singleton linear-map classifier append right cancellation)**.** *For all $P, Q, R : \mathsf{BHist}$, if $L_{\mathsf{e}}(P)$, then*

$$\lambda_{\mathsf{e}}(\mathsf{append}(Q, P), \mathsf{append}(R, P)) \iff \lambda_{\mathsf{e}}(Q, R).$$

Proof. For the forward direction, unfold the singleton classifier. The carrier clauses classify both right-suffixed histories with e. Empty-append inversion splits these clauses and gives $Q\mathsf{hsamee}$ and $R\mathsf{hsamee}$. Composing the first endpoint witness with the symmetry of the second gives $Q\mathsf{hsame}R$, which closes $\lambda_{\mathsf{e}}(Q, R)$.

Conversely, the hypothesis $L_{\mathsf{e}}(P)$ supplies the certified empty suffix. Together with the two carrier fields of $\lambda_{\mathsf{e}}(Q, R)$, empty-append composition classifies both $\mathsf{append}(Q, P)$ and $\mathsf{append}(R, P)$ with e. These two endpoint witnesses compose through symmetry to rebuild the right-suffixed classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append right cancel iff`

Theorem 48.21 (Singleton linear-map classifier append two-sided cancellation)**.** *For all $P, Q, R, S : \mathsf{BHist}$, if $L_{\mathsf{e}}(P)$ and $L_{\mathsf{e}}(S)$, then*

$$\lambda_{\mathsf{e}}(\mathsf{append}(P, Q), \mathsf{append}(R, S)) \iff \lambda_{\mathsf{e}}(Q, R).$$

Proof. For the forward direction, unfold the singleton classifier. The two carrier clauses classify $\mathsf{append}(P, Q)$ and $\mathsf{append}(R, S)$ with e. Empty-append inversion gives $Q\mathsf{hsamee}$ and $R\mathsf{hsamee}$, and these endpoint witnesses compose to close $\lambda_{\mathsf{e}}(Q, R)$.

Conversely, $L_{\mathsf{e}}(P)$ and $L_{\mathsf{e}}(S)$ combine with the carrier fields of $\lambda_{\mathsf{e}}(Q, R)$ to classify the two appended histories with e. The two resulting endpoint witnesses compose, with symmetry on the right, to rebuild the two-sided appended classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier append two sided cancel iff`

Theorem 48.22 (Singleton evaluation append-input classifier readback)**.** *For histories f, x, y, h, classifying the singleton evaluation of an appended input against h is exactly the singleton carrier condition on h:*

$$\mathsf{LinearMapSingletonClassifier}(\mathsf{LinearMapSingletonEval}(f, x \cdot y), h) \iff \mathsf{LinearMapSingletonCarrier}(h).$$

Proof. The singleton evaluator always returns e. In the forward direction, the classifier already contains the carrier witness for its right endpoint. In the reverse direction, the carrier witness for h combines with reflexivity at e and symmetry of the displayed carrier equality to form the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval append input classifier iff`

Theorem 48.23 (Singleton evaluation append-context classifier readback)**.** *For histories p, f, x, h : BHist, classifying a singleton evaluation after an appended context is exactly the pair of singleton carrier conditions on the context and the target:*

$$\mathsf{LinearMapSingletonClassifier}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), h) \iff \mathsf{LinearMapSingletonCarrier}(p) \wedge \mathsf{LinearM}$$

Proof. The singleton evaluator returns e. In the forward direction, the left carrier field classifies $\mathsf{append}(p, \mathsf{e})$ with e, and empty-append inversion exposes $p\mathsf{hsamee}$; the right classifier carrier gives $h\mathsf{hsamee}$.

Conversely, $p\mathsf{hsamee}$ combines with reflexivity at the evaluator result to classify $\mathsf{append}(p, \mathsf{LinearMapSingletonEva}$ with e. The target carrier for h and symmetry on that carrier witness then assemble the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval append context classifier iff`

Theorem 48.24 (Singleton linear-map evaluation context pair classifier)**.** *For histories p, q, f, x, g, y : BHist,*

$$\mathsf{LinearMapSingletonClassifier}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), \mathsf{append}(q, \mathsf{LinearMapSingletonEval}(g, y))) \iff$$

Proof. Both singleton evaluation tails are definitionally e. The append-pair classifier exactness splits the two appended endpoints into four singleton carrier obligations; the two evaluation-tail obligations are closed by reflexivity at e, leaving exactly the two context carriers. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier eval context pair iff`

Theorem 48.25 (Singleton evaluation continuation classifier)**.** *For histories f, x, y, r, h : BHist, if*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), y, r),$$

then classifying the continuation result r against h is equivalent to the two exposed endpoints y and h lying in the singleton carrier:

$$\mathsf{LinearMapSingletonClassifier}(r, h) \iff \mathsf{LinearMapSingletonCarrier}(y) \wedge \mathsf{LinearMapSingletonCarrier}(h).$$

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation classifier iff`

Proof. The singleton evaluator returns e, so the continuation ledger displays r as $\mathsf{append}(\mathsf{e}, y)$. In the forward direction, the classifier field $r\mathsf{hsamee}$ and the left-unit append equation identify y with e, while the right classifier carrier gives the singleton witness for h. Conversely, singleton witnesses for y and h identify r with e through the same continuation ledger and left-unit append equation, and the two singleton carrier witnesses assemble the restricted classifier. □

Definition 48.26 (Concrete singleton linear-map evaluation graph carrier)**.** *For the concrete singleton empty-history* $\mathsf{LinMap}^{\uparrow}$ *instance, the evaluation graph row is the* BHist*-anchored visible triple* (f, x, y) *satisfying*

$$L_{\mathsf{e}}(f) \wedge A_{\mathsf{e}}(x) \wedge \rho_{\mathsf{e}}(\mathsf{LinearMapSingletonEval}(f, x), y).$$

Its ledger exposes exactly the map-code row f, the input-vector row x, the output endpoint y, and the continuation provenance row that relates $\mathsf{LinearMapSingletonEval}(f, x)$ *to y through* Cont *when such a continuation is displayed. No host function equality or quotient witness is part of the graph carrier.*

Theorem 48.27 (Singleton linear-map evaluation graph package)**.** *For the evaluation graph carrier of Definition 48.26, the following finite singleton package holds.*

1. *Output closure: every graph row has $A_{\mathsf{e}}(y)$, and every admitted $L_{\mathsf{e}}(f)$ and $A_{\mathsf{e}}(x)$ determines the singleton output row $\rho_{\mathsf{e}}(\mathsf{LinearMapSingletonEval}(f, x), \mathsf{e})$.*
2. *Pointwise classifier transport: replacing f, x, or y along λ_{e} and ρ_{e} preserves the graph row and its continuation provenance.*
3. *Identity and zero-map evaluation collapse: for every carried input x, the identity and zero-map codes evaluate to the same singleton endpoint and classify with the displayed x-row.*
4. *Composition evaluation collapse: if f, g, and x are carried, the graph row for* LinearMapSingletonEval(LinearMapS *and the graph row read through the continuation of* $\mathsf{LinearMapSingletonEval}(f, x)$ *by g expose the same singleton endpoint classifier.*

The package exports no host function equality, no quotient row, and no hidden selector row.

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEvaluationGraph package`

Proof. The carrier row is built over the same singleton empty-history table as Definition 48.3. Output closure follows because every evaluation endpoint is definitionally e, while Definition 48.4 supplies the public code, identity-code, zero-code, composite-code, and endpoint exactness rows.

For pointwise transport, use the existing singleton evaluation classifier readbacks in this file. They reduce each transported endpoint classifier to the same singleton carrier obligations for the visible map code, input vector, and output endpoint.

The identity and zero-map clauses are the identity and zero-map rows of the public-code exactness theorem, read through the graph definition. The continuation classifier Definition 48.25 repackages the same output endpoint as a continuation provenance row when a continuation is displayed.

For composition, apply Definition 48.3 to the singleton composition operation and then apply the existing composition continuation classifier in this file. Both the direct composite evaluation and the stagewise continuation evaluation reduce to e, so the endpoint classifier is the singleton ρ_{e} row. Since the construction only reads the finite BHist graph ledger, no host function equality, quotient carrier, or hidden selector can be exported. □

Theorem 48.28 (Singleton evaluation composition continuation classifier)**.** *If g, f, x lie in the singleton linear-map carrier and* $\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), g, r)$*, then the composed evaluation endpoint classifies against the continuation result:*

$$\mathsf{LinearMapSingletonClassifier}(\mathsf{LinearMapSingletonEval}(\mathsf{LinearMapSingletonComp}(g, f), x), r).$$

Proof. The continuation readback theorem applied to $\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), g, r)$ carries the result r whenever g is carried. The composed singleton evaluation endpoint is e, so the singleton classifier is assembled from the endpoint carrier, the result carrier, and symmetry of the result carrier witness. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval comp continuation classifier`

Theorem 48.29 (Singleton evaluation context continuation classifier)**.** *For histories p, f, x, y, r, h :* BHist*, if*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), y, r),$$

then classifying the continuation result r against h is equivalent to all three exposed endpoints lying in the singleton carrier:

$$\mathsf{LinearMapSingletonClassifier}(r, h) \iff \mathsf{LinearMapSingletonCarrier}(p) \wedge \mathsf{LinearMapSingletonCarrier}(y) \wedge \mathsf{LinearMapSingle}$$

Proof. In the forward direction, the result classifier transports the continuation to an empty result. Empty-result inversion splits the continuation into an empty context append and an empty continuation input, append empty-result inversion recovers the carrier witness for p, and the target carrier comes from the classifier.

Conversely, the singleton evaluator returns e, so the carrier witness for p makes the context append carried by e. Together with the carrier witness for y, continuation transport carries r by e, and the carrier witness for h assembles the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context continuation classifier iff`

Theorem 48.30 (Singleton evaluation nonempty endpoint empty-result exclusion)**.** *For histories p, f, x, y, r :* BHist*, if either p or y is not singleton-carried, then*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), y, r) \wedge r\mathsf{hsamee} \implies \bot.$$

Proof. Transport the continuation result to e. Empty-result inversion makes both endpoints empty, and append empty-result inversion gives the singleton carrier witness for p; either non-carrier hypothesis is contradicted. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context nonempty endpoint continuation result absurd`

Theorem 48.31 (Singleton evaluation visible context continuation exclusion)**.** *For histories p, f, x, y, r, h :* BHist*, a singleton evaluation with a visible context prefix cannot have a classified continuation result:*

$$\mathsf{Cont}(\mathsf{append}(\mathsf{E0}(p), \mathsf{LinearMapSingletonEval}(f, x)), y, r) \implies (\mathsf{LinearMapSingletonClassifier}(r, h) \implies \bot),$$

and the same statement holds with $\mathsf{E1}(p)$ *in place of* $\mathsf{E0}(p)$*.*

Proof. The context-continuation classifier readback says any such classifier forces the visible context prefix to lie in the singleton carrier. That carrier is hsame to e, contradicting constructor separation for the visible head. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context continuation visible context absurd`

Theorem 48.32 (Singleton evaluation continuation append target carrier)**.** *For histories* f, x, y, p, q : BHist*, if*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), y, \mathsf{append}(p, q)),$$

then the exposed continuation input is in the singleton carrier exactly when both append factors are:

$$\mathsf{LinearMapSingletonCarrier}(y) \iff \mathsf{LinearMapSingletonCarrier}(p) \wedge \mathsf{LinearMapSingletonCarrier}(q).$$

Proof. The singleton evaluator returns e, so the continuation ledger identifies $\mathsf{append}(p, q)$ with $\mathsf{append}(\mathsf{e}, y)$. The left-unit law for append identifies this with y. Thus a singleton carrier witness for y turns into an empty-result witness for $\mathsf{append}(p, q)$, which append empty-result inversion splits into carrier witnesses for p and q. Conversely, carrier witnesses for both factors assemble an empty-result witness for $\mathsf{append}(p, q)$, and the same continuation ledger and left-unit law carry it back to y. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation append target carrier iff`

Theorem 48.33 (Singleton evaluation continuation visible result exclusion)**.** *For histories* f, x, y, p : BHist*, if* $\mathsf{LinearMapSingletonCarrier}(y)$*, then the continuation result of the singleton evaluator cannot be either visible endpoint:*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), y, \mathsf{E0}(p)) \Longrightarrow \bot, \qquad \mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), y, \mathsf{E1}(p)) \Longrightarrow \bot.$$

Proof. The continuation ledger identifies each displayed result with $\mathsf{append}(\mathsf{e}, y)$. The append left-unit law and the carrier witness for y force that result to be history-same to e. Constructor no-confusion for E0 and E1 closes the two visible cases. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation visible result absurd`

Theorem 48.34 (Singleton evaluation context continuation visible result exclusion)**.** *For histories* p, f, x, y, z : BHist*, if both* p *and* y *lie in the singleton linear-map carrier, then the continuation result of the prefixed singleton evaluator cannot be either visible endpoint:*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), y, \mathsf{E0}(z)) \Longrightarrow \bot, \qquad \mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)),$$

Proof. The singleton evaluator returns e, so the carrier witness for p makes the prefixed evaluator history empty. Transporting the displayed continuation against the right-unit empty continuation forces each displayed result to be hsame to e. Constructor no-confusion excludes both visible endpoint results. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context continuation visible result absurd`

Theorem 48.35 (Singleton evaluation visible target classifier exclusion)**.** *For histories* f, x, p : BHist*, the singleton evaluator cannot classify against either visible target endpoint:*

$$\mathsf{LinearMapSingletonClassifier}(\mathsf{LinearMapSingletonEval}(f, x), \mathsf{E0}(p)) \Longrightarrow \bot, \qquad \mathsf{LinearMapSingletonClassifier}(\mathsf{LinearMa}$$

Proof. The singleton evaluator returns e, but the classifier still requires its right endpoint to lie in the singleton carrier. Thus a classifier witness against $\mathsf{E0}(p)$ or $\mathsf{E1}(p)$ would force that visible endpoint to be history-same to e. Constructor no-confusion for the two visible constructors closes both cases. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval visible target classifier absurd`

Theorem 48.36 (Singleton evaluation continuation visible target exclusion)**.** *For histories* f, x, y, r, p : BHist*, if*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), y, r),$$

then both visible target classifier claims are impossible:

$$\mathsf{LinearMapSingletonClassifier}(r, \mathsf{E0}(p)) \Longrightarrow \bot, \qquad \mathsf{LinearMapSingletonClassifier}(r, \mathsf{E1}(p)) \Longrightarrow \bot.$$

The ten-field fit is: traditional target terminal linear-map continuation result exclusion; BEDC source BHist*,* Cont*, and the singleton* $\mathsf{LinMap}^{\uparrow}$ *evaluator; carrier* LinearMapSingletonCarrier*; classifier* LinearMapSingletonClassifier*; stability from continuation classifier readback and constructor no-confusion; ledger continuation result endpoint; core theorem this block; exactness visible-target impossibility; standard bridge terminal linear hom; status* ProofTarget*.*

Proof. Apply Theorem 48.25 to the displayed continuation with target $\mathsf{E0}(p)$. A classifier witness for $\mathsf{E0}(p)$ then forces $\mathsf{LinearMapSingletonCarrier}(\mathsf{E0}(p))$.

Apply the same continuation classifier readback with target $\mathsf{E1}(p)$. A classifier witness for $\mathsf{E1}(p)$ forces $\mathsf{LinearMapSingletonCarrier}(\mathsf{E1}(p))$.

Unfolding the singleton carrier makes each witness a history-sameness proof from the visible endpoint to e. Constructor no-confusion for E0 and E1 contradicts those histories being empty, so both displayed classifier claims imply ⊥. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation visible target classifier absurd`

Theorem 48.37 (Singleton evaluation continuation visible input exclusion)**.** *For histories* f, x, p, r, h : BHist*, each visible continuation input prevents a singleton-classified continuation result:*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), \mathsf{E0}(p), r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot),$$

and

$$\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), \mathsf{E1}(p), r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot).$$

The ten-field fit is: traditional target terminal linear-map continuation input exclusion; BEDC source BHist*,* Cont*, and the singleton evaluator; carrier* LinearMapSingletonCarrier*; classifier* LinearMapSingletonClas*stability from continuation classifier readback plus constructor separation; ledger continuation input endpoint; core theorem this block; exactness visible-input impossibility; standard bridge terminal linear hom; status* ProofTarget*.*

Proof. Apply Theorem 48.25 to the first displayed continuation with continuation input $\mathsf{E0}(p)$. A classifier witness for r against h then forces $\mathsf{LinearMapSingletonCarrier}(\mathsf{E0}(p))$.

Apply the same continuation classifier readback to the second displayed continuation with continuation input $\mathsf{E1}(p)$. A classifier witness for r against h forces $\mathsf{LinearMapSingletonCarrier}(\mathsf{E1}(p))$.

Unfolding the singleton carrier turns each carrier witness into a history-sameness proof from the visible continuation input to e. Constructor no-confusion for $\mathsf{E0}$ and $\mathsf{E1}$ contradicts those histories being empty, so each visible continuation input case implies $\bot$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation visible input classifier absurd`

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval continuation visible input absurd`

The $\mathsf{LinMap}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and singleton action, so later Lean work can target BHist-anchored linear-map laws instead of only abstract function fields.

Theorem 48.38 (Singleton evaluation context continuation append-target carrier)**.** *For histories* $p, f, x, y, q, r : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), y, \mathsf{append}(q, r)),$$

then the prefixed singleton evaluator and continuation input lie in the singleton carrier exactly when both exposed result factors do:

$$\mathsf{LinearMapSingletonCarrier}(p) \wedge \mathsf{LinearMapSingletonCarrier}(y) \iff \mathsf{LinearMapSingletonCarrier}(q) \wedge \mathsf{LinearMapSingletonC}$$

Proof. The singleton evaluator returns e. From carrier witnesses for p and y, the continuation result is transported against the right-unit empty continuation, forcing $\mathsf{append}(q, r)$ to be empty and hence splitting into singleton carrier witnesses for q and r. Conversely, carrier witnesses for q and r assemble an empty-result witness for $\mathsf{append}(q, r)$; transporting the displayed continuation to the empty result and applying empty-result inversion splits the prefixed evaluator and the continuation input, yielding carrier witnesses for p and y. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context continuation append target carrier iff`

Theorem 48.39 (Singleton evaluation context append-input/result carrier exactness)**.** *For histories* $p, f, x, y, z, q, r : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), \mathsf{append}(y, z), \mathsf{append}(q, r)),$$

then the prefixed singleton evaluator and both exposed input factors lie in the singleton carrier exactly when both exposed result factors do:

$$\mathsf{LinearMapSingletonCarrier}(p) \wedge \mathsf{LinearMapSingletonCarrier}(y) \wedge \mathsf{LinearMapSingletonCarrier}(z) \iff \mathsf{LinearMapSingletonC}$$

Proof. Apply append-target carrier exactness with continuation input $\mathsf{append}(y, z)$. The forward direction combines carrier witnesses for y and z into a carrier witness for the appended input. The reverse direction splits the recovered appended-input carrier by empty-append inversion. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context append input result carrier iff`

Theorem 48.40 (Singleton linear-map classifier continuation visible source exclusion)**.** *For histories $p, q, r, h : \mathsf{BHist}$, no continuation whose source is $\mathsf{E0}(p)$ or $\mathsf{E1}(p)$ can have a result classified by the singleton linear-map classifier:*

$$\mathsf{Cont}(\mathsf{E0}(p), q, r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot),$$

and

$$\mathsf{Cont}(\mathsf{E1}(p), q, r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot).$$

Proof. The singleton classifier gives $r\mathsf{hsamee}$. Transport the displayed continuation result to e and apply empty-result inversion for Cont. The source endpoint is forced to be e, contradicting constructor separation for $\mathsf{E0}$ or $\mathsf{E1}$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation visible source absurd`

Theorem 48.41 (Singleton linear-map classifier continuation visible target exclusion)**.** *For histories $p, q, r, h : \mathsf{BHist}$, no continuation whose target is $\mathsf{E0}(p)$ or $\mathsf{E1}(p)$ can have a result classified by the singleton linear-map classifier:*

$$\mathsf{Cont}(q, \mathsf{E0}(p), r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot),$$

and

$$\mathsf{Cont}(q, \mathsf{E1}(p), r) \Longrightarrow (\mathsf{LinearMapSingletonClassifier}(r, h) \Longrightarrow \bot).$$

Proof. The singleton classifier gives $r\mathsf{hsamee}$. Transport the displayed continuation result to e and apply empty-result inversion for Cont. The target endpoint is forced to be e, contradicting constructor separation for $\mathsf{E0}$ or $\mathsf{E1}$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation visible target absurd`

Theorem 48.42 (Singleton evaluation context continuation append-result carrier exactness)**.** *For histories $p, f, x, y, q, r : \mathsf{BHist}$, if the context p is carried by the singleton linear-map carrier and*

$$\mathsf{Cont}(\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x)), y, \mathsf{append}(q, r)),$$

then the continuation input y is singleton-carried exactly when both append result components are singleton-carried:

$$\mathsf{LinearMapSingletonCarrier}(y) \Longleftrightarrow \mathsf{LinearMapSingletonCarrier}(q) \wedge \mathsf{LinearMapSingletonCarrier}(r).$$

Proof. The singleton evaluator returns e, so the carried context p carries the left continuation endpoint. If y is carried, continuation determinacy against the empty unit carries $\mathsf{append}(q, r)$, and append empty-result inversion splits this into carriers for q and r.

Conversely, carriers for q and r carry the displayed append result. Moving the continuation result to e and inverting the empty continuation exposes $y\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context continuation append result carrier iff`

Theorem 48.43 (Singleton linear-map append-source append-result classifier exactness)**.** *For histories* $p, q, r, s, t, h : \mathsf{BHist}$*, if*

$$\mathsf{Cont}(\mathsf{append}(p, q), r, \mathsf{append}(s, t)),$$

then

$$\lambda_{\mathsf{e}}(\mathsf{append}(s, t), h) \iff L_{\mathsf{e}}(p) \wedge L_{\mathsf{e}}(q) \wedge L_{\mathsf{e}}(r) \wedge L_{\mathsf{e}}(s) \wedge L_{\mathsf{e}}(t) \wedge L_{\mathsf{e}}(h).$$

Proof. Apply the append-source classifier exactness theorem to the displayed continuation. In the forward direction, the singleton classifier carries the append result, and empty-append inversion splits it into carriers for s and t. Conversely, carriers for p, q, r rebuild the continuation source as an empty endpoint; the displayed continuation then carries $\mathsf{append}(s, t)$, closing the singleton classifier with the carrier of h. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonClassifier continuation append source append result iff`

Theorem 48.44 (Singleton evaluation composition continuation pair classifier)**.** *If* g *lies in the singleton linear-map carrier and* $\mathsf{Cont}(\mathsf{LinearMapSingletonEval}(f, x), g, r)$*, while the composed evaluation endpoint continues through* e *to* s*, then the two continuation results are classified by the singleton linear-map classifier:*

$$\mathsf{LinearMapSingletonClassifier}(r, s).$$

Proof. The singleton evaluator and singleton composition evaluator both expose e as their source endpoint. Transporting the first continuation along the carrier witness for g carries r by the singleton carrier. Transporting the second continuation along the empty endpoint carries s. The two empty endpoint readbacks compose to the displayed singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval comp continuation pair classifier`

Theorem 48.45 (Singleton evaluation context composition continuation pair classifier)**.** *If* p *and* y *lie in the singleton linear-map carrier, the contextual evaluation endpoint* $\mathsf{append}(p, \mathsf{LinearMapSingletonEval}(f, x))$ *continues through* y *to* r*, and the composed evaluation endpoint continues through* e *to* s*, then the two continuation results are classified:*

$$\mathsf{LinearMapSingletonClassifier}(r, s).$$

Proof. The singleton evaluator has empty carrier. Combining it with the carrier of p carries the contextual source endpoint. Transporting the first continuation against the empty right unit therefore carries r. The composed evaluation source and empty continuation input similarly carry s. These two empty endpoint readbacks compose to the singleton classifier relation. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonEval context comp continuation pair classifier`

Theorem 48.46 (Singleton composition continuation pair classifier)**.** *If two continuations start from the concrete singleton composition endpoint* $\mathsf{LinearMapSingletonComp}(g, f)$ *and continue through carried histories* y *and* z *to* r *and* s*, then the two results are classified:*

$$\mathsf{Cont}(\mathsf{LinearMapSingletonComp}(g, f), y, r) \wedge \mathsf{Cont}(\mathsf{LinearMapSingletonComp}(g, f), z, s) \wedge L_{\mathsf{e}}(y) \wedge L_{\mathsf{e}}(z) \Longrightarrow \mathsf{LinearMapS}$$

Proof. The concrete singleton composition endpoint is e. Transport the first continuation along the carrier witness for y to carry r, and transport the second along the carrier witness for z to carry s. The two empty endpoint readbacks compose by symmetry and transitivity into the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonComp continuation pair classifier`

48.3 The certificate

Definition 48.47 (LinearMap source specification)**.** *The source is a pair of vector spaces over the same certified field.*

Definition 48.48 (LinearMap pattern specification)**.** *The pattern is function-with-law formation: map data together with linearity witnesses.*

Definition 48.49 (LinearMap classifier specification)**.** *The classifier relates two linear maps when their outputs are codomain-classified the same for every input witness.*

Definition 48.50 (LinearMap stability certificate)**.** *The stability certificate consists of:*

1. *pointwise classifier reflexivity*
2. *pointwise classifier transitivity*
3. *preservation of zero derived from linearity*
4. *additivity preservation*
5. *scalar-compatibility preservation*
6. *closure under identity and composition*

Definition 48.51 (LinearMap ledger policy)**.** *Record the function identifier, its linearity witnesses, and codomain ledgers for pointwise comparisons.*

Definition 48.52 ($\mathsf{LinMap}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{LinMap}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapUp StdBridge`

Primitive boundary. $\mathsf{LinMap}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 48.53 (Zero-axiom encoding note)**.** *Avoids funext by using pointwise classifier sameness instead of equality of functions.*

Theorem 48.54 (LinearMap terminal standard bridge)**.** *The singleton empty-history public map-code surface of Definition 48.2 and Definition 48.4 interprets in the terminal one-object, one-arrow linear-hom space. Every carried public code maps to the unique hom, evaluation sends the unique carried vector to itself, composition sends the unique composable hom pair to the unique hom, and the return interpretation is classified by* λ_{e} *and* ρ_{e}*.*

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton terminal standard bridge`

Proof. The public-code exactness theorem identifies every carried public map code with the empty-history identity code under λ_{e}, and it gives the same collapse for the zero code, the composed code, and the evaluation readback. Thus the identity, zero, composition, and evaluation rows all land in the same singleton code class.

The terminal standard object has one object endpoint and one hom endpoint. After the public surface has collapsed to the empty-history code class, both directions of the bridge are forced: a carried public code can only interpret as the unique terminal hom, and the unique terminal hom can only be represented by the carried empty-history public code.

For vectors, the singleton carrier supplies only the empty endpoint, so evaluation of the unique hom at the unique carried vector returns that vector. The map-code comparison is the λ_{e} classification already supplied by exactness, and the vector readback is the singleton ρ_{e} classification. No host function extensionality is used; the argument is only constructor collapse and carrier-aware classifier transport on the singleton surface. □

Theorem 48.55 (LinearMap pointwise classifier equivalence)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let V and W carry $\mathsf{VecSpace}^{\uparrow}(F,V)$ and $\mathsf{VecSpace}^{\uparrow}(F,W)$ certificates. Write C_V and C_W for source and codomain carrier predicates, and write $\sim_W$ for a codomain classifier supplied by $\mathsf{NameCert}(C_W, \sim_W)$. Let $\mathsf{Lin}_{V,W}(f)$ be the carried $\mathsf{LinMap}^{\uparrow}(V,W)$ map-code predicate from Definition 48.1; in particular, a witness of $\mathsf{Lin}_{V,W}(f)$ includes output closure $C_V(x) \Longrightarrow C_W(f(x))$. Define the pointwise $\mathsf{LinMap}^{\uparrow}$ classifier by*

$$f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} g :\Longleftrightarrow \mathsf{Lin}_{V,W}(f) \wedge \mathsf{Lin}_{V,W}(g) \wedge \forall x,\ C_V(x) \Longrightarrow f(x) \sim_W g(x).$$

Then $\sim_{\mathsf{LinMap}^{\uparrow}(V,W)}$ is reflexive, symmetric, and transitive on carried linear-map codes:

$$\begin{aligned}
&\mathsf{Lin}_{V,W}(f) \Longrightarrow f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} f,\\
&f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} g \Longrightarrow g \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} f,\\
&f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} g \wedge g \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} h \Longrightarrow f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} h.
\end{aligned}$$

Proof. By Definition 9.7, the codomain $\mathsf{NameCert}(C_W, \sim_W)$ supplies reflexivity for carried codomain outputs, symmetry for $\sim_W$, and transitivity for composable $\sim_W$ witnesses. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}$ classifier is the pointwise codomain comparison over all carried source inputs, together with the two carried map-code endpoints.

For reflexivity, assume $\mathsf{Lin}_{V,W}(f)$ and let x be a carried source input. Output closure in the linear-map carrier gives $C_W(f(x))$. Codomain reflexivity gives $f(x) \sim_W f(x)$. The two endpoint carrier fields are both the given $\mathsf{Lin}_{V,W}(f)$ witness, so the pointwise classifier relates f to itself.

For symmetry, assume $f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} g$. Its endpoint fields give $\mathsf{Lin}_{V,W}(f)$ and $\mathsf{Lin}_{V,W}(g)$. For each carried x, the pointwise field gives $f(x) \sim_W g(x)$; codomain symmetry gives $g(x) \sim_W f(x)$. Swapping the endpoint carrier fields and using these reversed pointwise comparisons gives $g \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} f$.

For transitivity, assume $f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} g$ and $g \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} h$. The first endpoint field of the first witness carries f, and the second endpoint field of the second witness carries h. For each carried x, the two pointwise fields give

$$f(x) \sim_W g(x), \qquad g(x) \sim_W h(x).$$

Codomain transitivity gives $f(x) \sim_W h(x)$. These pointwise composites, together with the carried endpoints f and h, form $f \sim_{\mathsf{LinMap}^{\uparrow}(V,W)} h$. Thus the classifier attached to Definition 48.49 satisfies the reflexive, symmetric, and transitive rows required by the stability and certificate obligations in Definition 48.50 and Definition 48.52. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton pointwise classifier equivalence`

Closure status for $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$.

Constructive story: A linear map between the singleton vector spaces is represented by the only code that can do anything here: the empty code, whose evaluation on every carried input vector returns the empty endpoint. Two map codes are classified pointwise, meaning every admitted input must evaluate to codomain-classified results; on the singleton evaluator this collapses to history-equality at e. The source is the singleton-code predicate, the pattern records that zero

map and identity are both the empty code, composition of empty codes is empty again, evaluation always lands in the singleton, and the kernel-triviality criterion holds because there is only one carried vector, while the ledger keeps each visible code aligned with its raw code. Since pointwise codomain classification is an equivalence and the evaluator never leaves the singleton, the empty code supplies the carrier witness, the classifier is reflexive, symmetric, and transitive, and recognized codes remain admitted. A code c lies in the signature gap of a $\mathsf{LinMap}^{\uparrow}$ token p when the source-domain accepts c, the signature relation pairs it with an answer code and evidence ledger, and the token-introduction relation names that answer as p; unfolded, c is the empty code and p is the singleton linear-map token. The terminal singleton bridge, public empty-code exactness row, singleton evaluator readback, and module-linear-map examples are all readings of that same empty-code table. The chapter registers this singleton public code package under the name $\mathsf{LinMap}^{\uparrow}$; no function extensionality is needed because every evaluation reaches the same endpoint, and rank-nullity, dual spaces, matrices, bounded operators, and non-singleton function spaces are not introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{LinMap}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature singleton public linear-map package binds the terminal singleton bridge, public empty-code exactness, singleton evaluator readback, pointwise classifier exactness, zero map uniqueness, composition and identity rows, the kernel-triviality criterion, and the module-linear-map examples already exhibited in this chapter.

Formal verification: FormalStatus(NameCert$_{\mathsf{LinMap}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.LinearMapUp.LinearMapSingleton\_public\_empty\_code\_exactness`

Bridge status: bridgeChecked

Not claimed: This chapter does not close arbitrary non-singleton function extensionality, rank-nullity, dual spaces, matrices, bounded operators, or non-singleton families.

Upgrade path: No further formal-axis work remains for the singleton public linear-map bridge; non-singleton families remain outside this chapter.

Lemma 48.56 (Codomain additive duplicate cancellation)**.** *Let W carry an $\mathsf{AbGroup}^{\uparrow}(W)$ certificate with additive operation $+_W$, additive unit 0_W, and classifier $\sim_W$. For any carried $w : W$, if*

$$w \sim_W w +_W w,$$

then

$$w \sim_W 0_W.$$

Proof. By Definition 42.33, the additive carrier of W has the same operation, unit, inverse, and classifier as a $\mathsf{Group}^{\uparrow}$ carrier. Symmetry of the classifier turns the hypothesis into

$$w +_W w \sim_W w.$$

In additive notation, Definition 41.114 says that a comparison $a +_W b \sim_W a$ in this group reduct forces $b \sim_W 0_W$. Taking $a = w$ and $b = w$ gives $w \sim_W 0_W$. □

Theorem 48.57 (LinearMap zero preservation from additivity)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let V and W carry $\mathsf{VecSpace}^{\uparrow}(F, V)$ and $\mathsf{VecSpace}^{\uparrow}(F, W)$ certificates. Write $+_V$, 0_V, and $\sim_V$ for the source vector addition, zero, and classifier, and write $+_W$, 0_W, and $\sim_W$ for the codomain additive operation, zero, and classifier. Let $f : V \to W$ be a carried $\mathsf{LinMap}^{\uparrow}(V, W)$ map. If the additive preservation row of f is*

$$f(x +_V y) \sim_W f(x) +_W f(y)$$

for carried source inputs x, y, then

$$f(0_V) \sim_W 0_W.$$

The ten-field fit is: traditional target ordinary linear maps send zero to zero; BEDC source $\mathsf{LinMap}^{\uparrow}(V, W)$ over the vector-space certificates; carrier the codomain vector carrier; classifier $\sim_W$; stability from additivity and codomain additive cancellation; ledger the zero-pair additive row and the cancellation witness; core theorem this block; exactness predicate $f(0_V) \sim_W 0_W$; standard bridge zero preservation for additive linear maps; status ProofTarget.

Proof. By Definition 47.1, the source vector-space carrier is a module carrier, and by Definition 46.1 its vector addition belongs to an additive abelian group. The source carrier transport used to read this comparison inside later rows is the standard semantic transport of Definition 26.3; the additive unit law supplies the source-side comparison

$$0_V +_V 0_V \sim_V 0_V.$$

The function-with-law data of Definition 48.1 evaluate the additive row on carried source expressions. Viewing evaluation by f as the descent map packaged by the function-like linear-map data, Definition 26.19 transports the source comparison $0_V +_V 0_V \sim_V 0_V$ through evaluation; codomain symmetry and transitivity then compose that transported comparison with the additive row at $x = 0_V$ and $y = 0_V$, giving

$$f(0_V) \sim_W f(0_V) +_W f(0_V).$$

The output $f(0_V)$ is carried by the codomain part of the same linear-map data.

For the codomain, Definition 47.23 reads the vector space as a module, and Definition 46.26 retains its additive abelian-group certificate with the same operation, zero, and classifier. Applying Definition 48.56 to $w = f(0_V)$ therefore gives

$$f(0_V) \sim_W 0_W.$$

This is exactly the zero-preservation item listed in Definition 48.50, obtained from additivity and the codomain additive cancellation package. □

Theorem 48.58 (LinearMap composition classifier congruence)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let U, V, and W carry $\mathsf{VecSpace}^{\uparrow}(F, U)$, $\mathsf{VecSpace}^{\uparrow}(F, V)$, and $\mathsf{VecSpace}^{\uparrow}(F, W)$ certificates over the same scalar source. Write $+_U$, $+_V$, and $+_W$ for vector addition, write the scalar actions as $r \cdot_U x$, $r \cdot_V y$, and $r \cdot_W z$, and write $\sim_U$, $\sim_V$, and $\sim_W$ for the corresponding vector classifiers. Let $f, f' : U \to V$ be carried $\mathsf{LinMap}^{\uparrow}(U, V)$ maps, and let $g, g' : V \to W$ be carried $\mathsf{LinMap}^{\uparrow}(V, W)$ maps. If f and f' are related by the pointwise $\mathsf{LinMap}^{\uparrow}(U, V)$ classifier, and g and g' are related by the pointwise $\mathsf{LinMap}^{\uparrow}(V, W)$ classifier, then $g \circ f$ and $g' \circ f'$ are carried $\mathsf{LinMap}^{\uparrow}(U, W)$ maps and are related by the pointwise $\mathsf{LinMap}^{\uparrow}(U, W)$ classifier:*

$$g \circ f \sim_{\mathsf{LinMap}^{\uparrow}(U,W)} g' \circ f'.$$

Proof. By Definition 47.1, each displayed vector-space certificate is a module certificate over the same scalar field, and by Definition 46.1 the addition, scalar action, carrier, and classifier rows used below are the module rows specialized to vector spaces. By Definition 48.1, each carried linear map has output closure, additive preservation, scalar compatibility, and a descent certificate for transporting classifier comparisons through evaluation.

First consider $g \circ f$. For a carried input $x : U$, output closure for f gives that $f(x)$ is carried in V, and output closure for g gives that $g(f(x))$ is carried in W. Thus the composite has output closure. For carried inputs $x, y : U$, the additive row of f gives

$$f(x +_U y) \sim_V f(x) +_V f(y).$$

Transporting this comparison through evaluation by g using Definition 26.19, and then applying the additive row of g, gives

$$g(f(x +_U y)) \sim_W g(f(x) +_V f(y)) \sim_W g(f(x)) +_W g(f(y)).$$

Classifier transitivity in the codomain module gives the additive row for $g \circ f$. For a carried scalar $r : F$ and carried input $x : U$, the scalar row of f gives

$$f(r \cdot_U x) \sim_V r \cdot_V f(x).$$

Transporting this comparison through evaluation by g, and then applying the scalar row of g, gives

$$g(f(r \cdot_U x)) \sim_W g(r \cdot_V f(x)) \sim_W r \cdot_W g(f(x)).$$

Again codomain transitivity gives scalar compatibility for $g \circ f$. Hence $g \circ f$ is a carried $\mathsf{LinMap}^{\uparrow}(U, W)$ map. The same argument with f' and g' gives that $g' \circ f'$ is carried.

It remains to prove pointwise classifier congruence. Fix a carried input $x : U$. The pointwise classifier relation between f and f' gives

$$f(x) \sim_V f'(x).$$

The descent certificate for g transports this input comparison to

$$g(f(x)) \sim_W g(f'(x)).$$

The pointwise classifier relation between g and g' applied at the carried input $f'(x)$ gives

$$g(f'(x)) \sim_W g'(f'(x)).$$

Transitivity of $\sim_W$ yields

$$(g \circ f)(x) \sim_W (g' \circ f')(x).$$

Because x was arbitrary, Definition 48.49 identifies this pointwise comparison with the $\mathsf{LinMap}^{\uparrow}(U, W)$ classifier relation between the two composites. The ledger is the composition row recorded by Definition 48.51, with the two input map ledgers and the codomain pointwise comparison ledger as its witnesses. □

Definition 48.59 (Module LinearMap certificate package)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the module classifiers, $+_M$ and $+_N$ for vector addition, 0_M and 0_N for vector zero, and $\odot_M$ and $\odot_N$ for scalar action. A package*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

for a function $f : M \to N$ consists of:

1. *output closure: every carried $x : M$ has carried output $f(x) : N$;*
2. *source-classifier descent: $x \sim_M y$ implies $f(x) \sim_N f(y)$;*
3. *additive preservation: $f(x +_M y) \sim_N f(x) +_N f(y)$ for carried $x, y : M$;*
4. *scalar compatibility: $f(r \odot_M x) \sim_N r \odot_N f(x)$ for carried scalars $r : R$ and carried $x : M$;*
5. *zero preservation: $f(0_M) \sim_N 0_N$;*
6. *ledger data recording the function identifier, the two linearity rows, the zero row, and the codomain pointwise comparison ledgers.*

This is the module fragment of the linear-map carrier in Definition 48.1, with its pointwise classifier read as in Definition 48.49 and its stability rows read against Definition 46.18.

Lean4 ✓ BEDC.Derived.LinearMapUp.ModuleLinearMapCertificatePackage exactness

Theorem 48.60 (Module LinearMap composition certificate)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate, and let M, N, and P carry* $\mathsf{Module}^{\uparrow}(R, M)$, $\mathsf{Module}^{\uparrow}(R, N)$, *and* $\mathsf{Module}^{\uparrow}(R, P)$ *certificates over the same scalar source. If*

$$\mathsf{LinearMapCert}_R(M, N; f) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(N, P; g),$$

then the composite function $(g \circ f)(x) := g(f(x))$ carries a certificate

$$\mathsf{LinearMapCert}_R(M, P; g \circ f).$$

Proof. Unpack the two input certificates according to Definition 48.59. The source data for the composite are the same scalar certificate R together with the endpoint modules M and P, which are among the module certificate obligations of Definition 46.20.

For output closure, let x be carried in M. Output closure for f gives that $f(x)$ is carried in N, and output closure for g gives that $g(f(x))$ is carried in P.

For source-classifier descent, assume $x \sim_M y$. Descent for f gives

$$f(x) \sim_N f(y).$$

Descent for g transports this comparison to

$$g(f(x)) \sim_P g(f(y)),$$

which is the descent row for $g \circ f$.

For additivity, let x and y be carried in M. Additive closure in the M module makes $x +_M y$ carried. The additive row of f gives

$$f(x +_M y) \sim_N f(x) +_N f(y).$$

Descent for g sends this comparison to

$$g(f(x +_M y)) \sim_P g(f(x) +_N f(y)).$$

The outputs $f(x)$ and $f(y)$ are carried in N, so the additive row of g gives

$$g(f(x) +_N f(y)) \sim_P g(f(x)) +_P g(f(y)).$$

Classifier transitivity from Definition 9.8 composes the last two displayed comparisons and gives the additive row for $g \circ f$.

For scalar compatibility, let r be a carried scalar and let x be carried in M. Scalar-action closure in the M module makes $r \odot_M x$ carried. The scalar row of f gives

$$f(r \odot_M x) \sim_N r \odot_N f(x).$$

Descent for g transports this to

$$g(f(r \odot_M x)) \sim_P g(r \odot_N f(x)).$$

Since $f(x)$ is carried in N, the scalar row of g gives

$$g(r \odot_N f(x)) \sim_P r \odot_P g(f(x)).$$

Transitivity of the P classifier gives scalar compatibility for the composite.

For zero preservation, the zero row of f gives

$$f(0_M) \sim_N 0_N.$$

Descent for g gives

$$g(f(0_M)) \sim_P g(0_N),$$

and the zero row of g gives

$$g(0_N) \sim_P 0_P.$$

Transitivity of the P classifier gives $(g \circ f)(0_M) \sim_P 0_P$.

The pointwise classifier for maps $M \to P$ is exactly the relation specified in Definition 48.49. The rows above are the output-closure, descent, additivity, scalar-compatibility, zero-preservation, and composition-closure entries of Definition 48.50 read over the module carriers of Definition 46.1. The ledger is the composite record formed from the two input function identifiers and their linearity, zero, and pointwise comparison ledgers, matching the policy of Definition 48.51. Therefore the carrier, source, pattern, classifier, stability, and ledger data satisfy Definition 48.52 for $g \circ f$. □

Theorem 48.61 (Module identity LinearMap certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M carry a $\mathsf{Module}^{\uparrow}(R, M)$ certificate. Write $\sim_M$ for the module classifier, $+_M$ for module addition, 0_M for module zero, and $\odot_M$ for scalar action. Let $\mathrm{id}_M : M \to M$ be the identity function. Then*

$$\mathsf{LinearMapCert}_R(M, M; \mathrm{id}_M)$$

holds. The source and target classifier are the same module classifier from Definition 46.17, and the ledger is the identity-function ledger governed by Definition 48.51.

The ten-field fit is: traditional target the ordinary identity linear map on a module; BEDC source a single $\mathsf{Module}^{\uparrow}(R, M)$ certificate; carrier the function id_M together with the rows listed in Definition 48.59; classifier the pointwise module classifier from Definition 48.49; stability output closure, descent, additivity, scalar compatibility, zero preservation, and identity closure from Definition 48.50; ledger the identity function identifier and the inherited operation ledgers required by Definition 48.51; core theorem this block; exactness predicate the displayed certificate; standard bridge the usual identity linear map; status ProofTarget.

Proof. Unpack the module certificate by Definition 46.20. It supplies the module carrier, the module classifier, the additive operation and zero, the scalar action, their stability fields, and their ledgers. By Definition 46.17, the classifier used on the source and target copies of M is the same relation $\sim_M$. The carrier-and-equivalence core of the derived certificate, described in Definition 9.7, supplies classifier reflexivity on carried module endpoints.

The rows required by Definition 48.59 are now checked one by one. For output closure, let x be carried in M. Since $\mathrm{id}_M(x) = x$, the output is the same carried endpoint.

For source-classifier descent, assume $x \sim_M y$. Applying the identity function changes neither endpoint, so the required comparison

$$\mathrm{id}_M(x) \sim_M \mathrm{id}_M(y)$$

is exactly the supplied comparison $x \sim_M y$.

For additivity, let x and y be carried in M. The additive closure component listed in Definition 46.18 makes $x +_M y$ a carried module endpoint. Both sides of the additive row reduce to that same endpoint:

$$\mathrm{id}_M(x +_M y) = x +_M y, \qquad \mathrm{id}_M(x) +_M \mathrm{id}_M(y) = x +_M y.$$

Classifier reflexivity on the carried endpoint $x +_M y$ gives

$$\mathrm{id}_M(x +_M y) \sim_M \mathrm{id}_M(x) +_M \mathrm{id}_M(y).$$

For scalar compatibility, let r be a carried scalar and let x be carried in M. Scalar-action closure from Definition 46.18 makes $r \odot_M x$ carried. Again both endpoints are the same expression:

$$\mathrm{id}_M(r \odot_M x) = r \odot_M x, \qquad r \odot_M \mathrm{id}_M(x) = r \odot_M x.$$

Reflexivity of $\sim_M$ gives the scalar-compatibility row.

For zero preservation, the additive unit 0_M is part of the carried additive module data. The two endpoints are both 0_M:

$$\mathrm{id}_M(0_M) = 0_M.$$

Reflexivity gives $\mathrm{id}_M(0_M) \sim_M 0_M$.

It remains only to record the certificate data. The function identifier is id_M. The additive, scalar, and zero rows above use the existing module operation ledgers together with classifier reflexivity, and the descent row uses the input classifier witness itself. This is exactly the ledger policy of Definition 48.51. Therefore the carrier, source, pattern, classifier, stability, and ledger data satisfy Definition 48.52, yielding $\mathsf{LinearMapCert}_R(M, M; \mathrm{id}_M)$. □

Theorem 48.62 (Module LinearMap composition associativity certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M, N, P, and Q carry $\mathsf{Module}^{\uparrow}(R, M)$, $\mathsf{Module}^{\uparrow}(R, N)$, $\mathsf{Module}^{\uparrow}(R, P)$, and $\mathsf{Module}^{\uparrow}(R, Q)$ certificates over the same scalar source. Write $\sim_Q$ for the module classifier of Q. If*

$$\mathsf{LinearMapCert}_R(M, N; f), \qquad \mathsf{LinearMapCert}_R(N, P; g), \qquad \mathsf{LinearMapCert}_R(P, Q; k),$$

then both associated triple composites carry certificates

$$\mathsf{LinearMapCert}_R(M, Q; (k \circ g) \circ f) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, Q; k \circ (g \circ f)),$$

and they are related by the pointwise $\mathsf{LinMap}^{\uparrow}(M, Q)$ classifier:

$$(k \circ g) \circ f \sim_{\mathsf{LinMap}^{\uparrow}(M,Q)} k \circ (g \circ f).$$

Equivalently, for every carried $x : M$,

$$((k \circ g) \circ f)(x) \sim_Q (k \circ (g \circ f))(x).$$

The ten-field fit is: traditional target associativity of composition of R-linear maps; BEDC source the three displayed $\mathsf{LinearMapCert}_R$ packages over a common scalar source; carrier the two composite function-with-law packages; classifier the pointwise Q-classifier from Definition 48.49; stability composition closure and endpoint reflexivity; ledger the two binary composition ledgers and the Q-endpoint reflexivity ledger; core theorem this block; exactness predicate the displayed pointwise comparison; standard bridge the usual associativity of function composition read pointwise; status ProofTarget.

Proof. Apply Definition 48.60 to the certificates for f and g. This gives

$$\mathsf{LinearMapCert}_R(M, P; g \circ f).$$

Applying the same theorem to this composite and to k gives

$$\mathsf{LinearMapCert}_R(M, Q; k \circ (g \circ f)).$$

In the other association, apply Definition 48.60 first to g and k, giving

$$\mathsf{LinearMapCert}_R(N, Q; k \circ g).$$

Applying the same theorem to f and this composite gives

$$\mathsf{LinearMapCert}_R(M, Q; (k \circ g) \circ f).$$

These are exactly the two certificate clauses required by Definition 48.59 and the composition-closure item of Definition 48.50.

It remains to compare the two certified composites. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}(M, Q)$ classifier is pointwise comparison in the codomain classifier. Let x be a carried endpoint of M. The output-closure rows in Definition 48.59 give carried endpoints $f(x)$ in N, $g(f(x))$ in P, and $k(g(f(x)))$ in Q. Both associated composites evaluate to this same codomain endpoint:

$$((k \circ g) \circ f)(x) = k(g(f(x))) = (k \circ (g \circ f))(x).$$

The module classifier on Q is the carrier classifier of Definition 46.17. Its reflexivity row, supplied by the naming-certificate core in Definition 9.7, classifies the carried endpoint $k(g(f(x)))$ with itself. Hence

$$((k \circ g) \circ f)(x) \sim_Q (k \circ (g \circ f))(x).$$

Since the carried input x was arbitrary, Definition 48.49 packages these endpoint comparisons as

$$(k \circ g) \circ f \sim_{\mathsf{LinMap}^{\uparrow}(M,Q)} k \circ (g \circ f).$$

The ledger field is formed from the two binary composition ledgers governed by Definition 48.51, together with the codomain reflexivity ledger at each point of Q. □

Theorem 48.63 (Module LinearMap identity unit laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_N$ for the module classifier on N, and let id_M and id_N be the identity functions. If*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

holds, then the two unit composites carry certificates

$$\mathsf{LinearMapCert}_R(M, N; \mathrm{id}_N \circ f) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, N; f \circ \mathrm{id}_M),$$

and both are related to f by the pointwise $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier:

$$\mathrm{id}_N \circ f \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f \qquad \textit{and} \qquad f \circ \mathrm{id}_M \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f.$$

Equivalently, for every carried $x : M$,

$$\mathrm{id}_N(f(x)) \sim_N f(x) \qquad \textit{and} \qquad f(\mathrm{id}_M(x)) \sim_N f(x).$$

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonComp identity unit laws`

Proof. Apply Definition 48.61 to M and to N, obtaining $\mathsf{LinearMapCert}_R(M, M; \mathrm{id}_M)$ and $\mathsf{LinearMapCert}_R(N, N; \mathrm{id}$ By Definition 48.60, first with f and id_N and then with id_M and f, the two composites carry certificates:

$$\mathsf{LinearMapCert}_R(M, N; \mathrm{id}_N \circ f) \qquad \text{and} \qquad \mathsf{LinearMapCert}_R(M, N; f \circ \mathrm{id}_M).$$

It remains to compare these carried maps. By Definition 48.49, this is pointwise comparison in the codomain classifier. Fix a carried $x : M$. The output-closure row of Definition 48.59 for f makes $f(x)$ carried in N. Since $\mathrm{id}_N(f(x)) = f(x)$ and $f(\mathrm{id}_M(x)) = f(x)$, classifier reflexivity for the module classifier of N, supplied by Definition 46.17 and Definition 9.7, gives

$$\mathrm{id}_N(f(x)) \sim_N f(x) \qquad \text{and} \qquad f(\mathrm{id}_M(x)) \sim_N f(x).$$

Quantifying over carried x packages these comparisons as the two $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier relations. The ledger is the identity and composition ledger required by Definition 48.51. □

Definition 48.64 (Module LinearMap image predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source, and let $f : M \to N$ be a function. Write $\sim_N$ for the classifier on N. The image predicate of f is $\mathsf{Im}_f(y) :\Longleftrightarrow \exists x : M,\ x$ is carried in $M \wedge f(x) \sim_N y$. The witness records the source carrier evidence for x and the target classifier comparison from $f(x)$ to y.*

Lean4 (definition) `BEDC.Derived.LinearMapUp.LinearMapSingletonImage`

Theorem 48.65 (Module LinearMap image submodule closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the module classifiers, $+_M$ and $+_N$ for module addition, 0_M and 0_N for module zero, and $\odot_M$ and $\odot_N$ for scalar action. If $\mathsf{LinearMapCert}_R(M, N; f)$ holds, then the image predicate Im_f from Definition 48.64 is stable under target-classifier transport, contains 0_N, and is closed under the inherited target addition and scalar action: $\mathsf{Im}_f(y) \wedge y \sim_N y' \Longrightarrow \mathsf{Im}_f(y')$, $\mathsf{Im}_f(0_N)$, $\mathsf{Im}_f(y) \wedge \mathsf{Im}_f(z) \Longrightarrow \mathsf{Im}_f(y +_N z)$, and $\mathsf{Im}_f(y) \Longrightarrow \mathsf{Im}_f(r \odot_N y)$ for every carried scalar $r : R$. Thus the image is a sub-module predicate of the target module for these closure rows.*

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingletonImage submodule closure`

Proof. Unpack the package in Definition 48.59. It supplies output closure, source-classifier descent, additive preservation, scalar compatibility, and zero preservation. Unpack the two module certificates through Definition 46.20; the source module supplies carried 0_M, additive closure of $x +_M u$, and scalar-action closure of $r \odot_M x$, while the target module supplies additive congruence, scalar-action congruence, and classifier equivalence by Definition 46.18, Definition 42.8, and Definition 9.7. For target-classifier transport, choose an image witness x for $\mathsf{Im}_f(y)$, so x is carried in M and $f(x) \sim_N y$. Composing this comparison with $y \sim_N y'$ by Definition 9.8 gives $f(x) \sim_N y'$. The same source witness x proves $\mathsf{Im}_f(y')$. For zero, the zero row of the linear-map certificate gives $f(0_M) \sim_N 0_N$. Since 0_M is carried by the additive part of the source module, $x = 0_M$ is an image witness for 0_N. For addition, assume $\mathsf{Im}_f(y)$ and $\mathsf{Im}_f(z)$. Choose carried source witnesses $x, u : M$ with $f(x) \sim_N y$ and $f(u) \sim_N z$. Carrier transport in the target classifier makes y and z carried in N. Source additive closure makes $x +_M u$ carried in M. The additive preservation row gives $f(x +_M u) \sim_N f(x) +_N f(u)$. Target additive congruence transports the two image comparisons to $f(x) +_N f(u) \sim_N y +_N z$. Transitivity of $\sim_N$ composes the two comparisons, so the carried source endpoint $x +_M u$ witnesses $\mathsf{Im}_f(y +_N z)$. For scalar action, let r be a carried scalar and

assume $\mathsf{Im}_f(y)$. Choose a carried source witness $x : M$ with $f(x) \sim_N y$. Source scalar-action closure makes $r \odot_M x$ carried in M. Scalar compatibility gives $f(r \odot_M x) \sim_N r \odot_N f(x)$. Target scalar-action congruence, using classifier reflexivity on the carried scalar r and the image comparison $f(x) \sim_N y$, gives $r \odot_N f(x) \sim_N r \odot_N y$. Transitivity gives $f(r \odot_M x) \sim_N r \odot_N y$, so $r \odot_M x$ is an image witness for $r \odot_N y$. The ledger is assembled from the source witnesses, the linearity rows of the $\mathsf{LinearMapCert}_R$ package, the target operation-congruence rows, and the classifier-transitivity ledger governed by Definition 48.51. No equality of functions is used; all comparisons are pointwise target-classifier comparisons as in Definition 48.49. □

Theorem 48.66 (Module LinearMap pointwise sum certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the module classifiers, $+_M$ and $+_N$ for module addition, 0_M and 0_N for module zero, and $\odot_M$ and $\odot_N$ for scalar action. For functions $f, g : M \to N$, define $h : M \to N$ by*

$$h(x) := f(x) +_N g(x).$$

If

$$\mathsf{LinearMapCert}_R(M, N; f) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, N; g),$$

then

$$\mathsf{LinearMapCert}_R(M, N; h).$$

The ten-field fit is: traditional target pointwise addition of ordinary R-linear maps; BEDC source the two displayed $\mathsf{LinearMapCert}_R$ packages over common $\mathsf{Module}^{\uparrow}$ certificates; carrier the pointwise-sum function h; classifier the pointwise target classifier from Definition 48.49; stability output closure, descent, additivity, scalar compatibility, and zero preservation from the module and abelian-group rows; ledger the two input function identifiers, their linearity rows, and the target additive, middle-four, distributive, zero, and classifier ledgers; core theorem this block; exactness predicate the displayed certificate; standard bridge the usual closure of $\mathrm{Hom}_R(M, N)$ under pointwise addition; status $\mathsf{ProofTarget}$.

Proof. Unpack the two input certificates by Definition 48.59. They supply output closure, source-classifier descent, additive preservation, scalar compatibility, and zero preservation for f and for g. Unpack the module certificates through Definition 46.20. The target module supplies additive closure, additive congruence, scalar-action closure, scalar-action congruence, scalar distributivity over module addition, and the additive abelian-group laws retained by Definition 46.26. Classifier symmetry and transitivity are supplied by the carrier certificate in Definition 9.7 and composed by Definition 9.8.

For output closure, let x be carried in M. Output closure for f and g makes $f(x)$ and $g(x)$ carried in N. Additive closure in the target module makes $f(x) +_N g(x)$ carried, so $h(x)$ is carried in N.

For source-classifier descent, assume $x \sim_M y$. Descent for f gives $f(x) \sim_N f(y)$, and descent for g gives $g(x) \sim_N g(y)$. Target additive congruence transports these two comparisons to

$$f(x) +_N g(x) \sim_N f(y) +_N g(y),$$

which is $h(x) \sim_N h(y)$.

For additivity, let x and y be carried in M. Source additive closure makes $x +_M y$ carried. The additive rows for f and g give

$$f(x +_M y) \sim_N f(x) +_N f(y) \qquad \text{and} \qquad g(x +_M y) \sim_N g(x) +_N g(y).$$

Target additive congruence therefore gives

$$h(x +_M y) \sim_N (f(x) +_N f(y)) +_N (g(x) +_N g(y)).$$

By Definition 46.26, the additive part of N is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same $+_N$ and $\sim_N$. Applying the middle-four law of Definition 42.13 in additive notation gives

$$(f(x) +_N f(y)) +_N (g(x) +_N g(y)) \sim_N (f(x) +_N g(x)) +_N (f(y) +_N g(y)).$$

Transitivity composes the last two displayed comparisons and yields

$$h(x +_M y) \sim_N h(x) +_N h(y).$$

For scalar compatibility, let r be a carried scalar and let x be carried in M. Source scalar-action closure makes $r \odot_M x$ carried. The scalar rows for f and g give

$$f(r \odot_M x) \sim_N r \odot_N f(x) \qquad \text{and} \qquad g(r \odot_M x) \sim_N r \odot_N g(x).$$

Target additive congruence gives

$$h(r \odot_M x) \sim_N (r \odot_N f(x)) +_N (r \odot_N g(x)).$$

The scalar distributivity row of Definition 46.18, read in the target module and oriented by classifier symmetry, gives

$$(r \odot_N f(x)) +_N (r \odot_N g(x)) \sim_N r \odot_N (f(x) +_N g(x)).$$

Transitivity yields

$$h(r \odot_M x) \sim_N r \odot_N h(x).$$

For zero preservation, the source zero endpoint is carried by the additive data. The zero rows for f and g give

$$f(0_M) \sim_N 0_N \qquad \text{and} \qquad g(0_M) \sim_N 0_N.$$

Target additive congruence gives

$$h(0_M) \sim_N 0_N +_N 0_N.$$

The additive unit law in the target abelian-group certificate classifies $0_N +_N 0_N$ with 0_N, and transitivity gives

$$h(0_M) \sim_N 0_N.$$

The pointwise classifier for maps $M \to N$ is exactly Definition 48.49; the preceding descent row is its required pointwise respect. The ledger records the function identifier h, the two input function identifiers, the pointwise addition ledger for each output, the additive congruence and middle-four ledgers for the additivity row, the scalar-distributivity ledger for the scalar row, and the zero-plus-unit ledger for the zero row, matching Definition 48.51. Hence the carrier, source, pattern, classifier, stability, and ledger obligations of Definition 48.52 are satisfied for h. □

Lemma 48.67 (Module scalar action preserves additive inverse)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let N carry a $\mathsf{Module}^{\uparrow}(R, N)$ certificate. Write $\sim_N$ for the module classifier, $+_N$ for module addition, 0_N for module zero, $-_N$ for additive inverse, and $\odot_N$ for scalar action. For every carried scalar $r : R$ and carried endpoint $y : N$,*

$$-_N(r \odot_N y) \sim_N r \odot_N (-_N y).$$

Proof. Unpack the module certificate through Definition 46.20. The additive reduct of the target module is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate by Definition 46.26, and its group reduct is obtained by Definition 42.33. These reducts keep the same addition, zero, additive inverse, and classifier. Thus inverse closure, inverse congruence, the inverse laws, additive congruence, and additive left cancellation are available through Definition 42.8, Definition 41.55, and Definition 41.66.

Let

$$a := r \odot_N y, \qquad b := r \odot_N (-_N y).$$

Scalar-action closure makes a carried, inverse closure makes $-_N y$ carried, and scalar-action closure again makes b carried. The additive right-inverse row for a gives

$$a +_N (-_N a) \sim_N 0_N.$$

The additive right-inverse row for y gives

$$y +_N (-_N y) \sim_N 0_N.$$

Scalar-action congruence, using classifier reflexivity on the carried scalar r, transports this to

$$r \odot_N (y +_N (-_N y)) \sim_N r \odot_N 0_N.$$

The zero-action annihilation theorem Definition 46.29 gives

$$r \odot_N 0_N \sim_N 0_N.$$

Transitivity therefore gives

$$r \odot_N (y +_N (-_N y)) \sim_N 0_N.$$

Scalar distributivity over module addition from Definition 46.18 gives

$$r \odot_N (y +_N (-_N y)) \sim_N (r \odot_N y) +_N (r \odot_N (-_N y)).$$

Using classifier symmetry and transitivity, this yields

$$a +_N b \sim_N 0_N.$$

Composing this comparison with the symmetry of $a +_N (-_N a) \sim_N 0_N$ gives

$$a +_N b \sim_N a +_N (-_N a).$$

Additive left cancellation in the retained group reduct removes the common left summand a, giving $b \sim_N -_N a$. Classifier symmetry gives

$$-_N a \sim_N b.$$

Substituting the definitions of a and b is exactly the displayed scalar-action inverse law. □

Theorem 48.68 (Module LinearMap pointwise inverse certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the module classifiers, $+_M$ and $+_N$ for module addition, 0_M and 0_N for module zero, $-_N$ for the target additive inverse, and $\odot_M$ and $\odot_N$ for scalar action. For a function $f : M \to N$, define $h : M \to N$ by*

$$h(x) := -_N f(x).$$

If

$$\mathsf{LinearMapCert}_R(M, N; f)$$

holds, then

$$\mathsf{LinearMapCert}_R(M, N; h)$$

holds. The ten-field fit is: traditional target pointwise additive inverse of an ordinary R-linear map; BEDC source the displayed $\mathsf{LinearMapCert}_R(M, N; f)$ *package over common* $\mathsf{Module}^{\uparrow}$ *certificates; carrier the pointwise inverse function h; classifier the pointwise target classifier from Definition 48.49; stability output closure, descent, additivity, scalar compatibility, and zero preservation from the target additive group and module action rows; ledger the input function identifier, the target inverse ledger, the inverse-congruence ledger, the inverse-of-sum ledger, and the scalar-action inverse ledger; core theorem this block; exactness predicate the displayed certificate; standard bridge closure of* $\mathrm{Hom}_R(M, N)$ *under pointwise additive inverse; status* $\mathsf{ProofTarget}$.

Proof. Unpack $\mathsf{LinearMapCert}_R(M, N; f)$ using Definition 48.59. It supplies output closure, source-classifier descent, additive preservation, scalar compatibility, and zero preservation for f. Unpack the two module certificates through Definition 46.20. The target module supplies additive inverse closure, additive congruence, scalar-action congruence, scalar distributivity over module addition, and the additive abelian-group laws retained by Definition 46.26. Classifier symmetry and transitivity are supplied by the naming-certificate core in Definition 9.7 and composed by Definition 9.8.

For output closure, let x be carried in M. Output closure for f makes $f(x)$ carried in N. Additive inverse closure in the target additive group makes $-_N f(x)$ carried, so $h(x)$ is carried in N.

For source-classifier descent, assume $x \sim_M y$. Descent for f gives

$$f(x) \sim_N f(y).$$

The inverse-congruence row in the target additive group gives

$$-_N f(x) \sim_N -_N f(y),$$

which is $h(x) \sim_N h(y)$.

For additivity, let x and y be carried in M. Source additive closure makes $x +_M y$ carried. The additive row for f gives

$$f(x +_M y) \sim_N f(x) +_N f(y).$$

Inverse congruence transports this comparison to

$$h(x +_M y) \sim_N -_N(f(x) +_N f(y)).$$

By Definition 46.26, the additive part of N is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same $+_N$, $-_N$, and $\sim_N$. Applying Definition 42.23 in additive notation gives

$$-_N(f(x) +_N f(y)) \sim_N (-_N f(x)) +_N (-_N f(y)).$$

Transitivity yields

$$h(x +_M y) \sim_N h(x) +_N h(y).$$

For scalar compatibility, let r be a carried scalar and let x be carried in M. Source scalar-action closure makes $r \odot_M x$ carried. The scalar row for f gives

$$f(r \odot_M x) \sim_N r \odot_N f(x).$$

Inverse congruence gives

$$h(r \odot_M x) \sim_N -_N(r \odot_N f(x)).$$

The output-closure row for f makes $f(x)$ carried in N, so Definition 48.67 applies in the target module and gives

$$-_N(r \odot_N f(x)) \sim_N r \odot_N (-_N f(x)).$$

Transitivity gives

$$h(r \odot_M x) \sim_N r \odot_N h(x).$$

For zero preservation, the zero row of f gives

$$f(0_M) \sim_N 0_N.$$

Inverse congruence gives

$$h(0_M) \sim_N -_N 0_N.$$

The additive group reduct identifies the inverse of the additive identity with the identity by Definition 41.56, so

$$-_N 0_N \sim_N 0_N.$$

Transitivity gives

$$h(0_M) \sim_N 0_N.$$

The pointwise classifier for maps $M \to N$ is exactly Definition 48.49; the descent row above is the required pointwise respect. The ledger records the function identifier h, the input function identifier for f, the target inverse operation at each output, the inverse-congruence ledgers for descent, additivity, scalar compatibility, and zero, the inverse-of-sum ledger from Definition 42.23, and the scalar-action inverse ledger from Definition 48.67, matching Definition 48.51. Hence the carrier, source, pattern, classifier, stability, and ledger obligations of Definition 48.52 are satisfied for h. □

Theorem 48.69 (LinearMap additive inverse preservation)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let V and W carry $\mathsf{VecSpace}^{\uparrow}(F, V)$ and $\mathsf{VecSpace}^{\uparrow}(F, W)$ certificates. Write $+_V$, 0_V, $-_V$, and $\sim_V$ for the source vector addition, zero, additive inverse, and classifier, and write $+_W$, 0_W, $-_W$, and $\sim_W$ for the codomain additive operation, zero, additive inverse, and classifier. Let $f : V \to W$ be a carried $\mathsf{LinMap}^{\uparrow}(V, W)$ map. Then every carried source point $m : V$ satisfies*

$$f(-_V m) \sim_W -_W f(m).$$

The ten-field fit is: traditional target ordinary linear maps preserve additive negation; BEDC source $\mathsf{LinMap}^{\uparrow}(V, W)$ over the two vector-space certificates; carrier the codomain vector carrier; classifier $\sim_W$; stability from additivity, zero preservation, source inverse descent, and codomain inverse uniqueness; ledger the source right-inverse witness, additive-preservation row, zero-preservation row, codomain inverse witness, and uniqueness comparison; core theorem this block; exactness predicate $f(-_V m) \sim_W -_W f(m)$; standard bridge additive inverse preservation for linear maps; status $\mathsf{ProofTarget}$.

Proof. By Definition 47.24, each vector-space certificate has a module reduct with the same vector addition, zero, additive inverse, and classifier. By Definition 46.26 and Definition 42.33, the source and codomain additive data may be read as group data with those same operations and classifiers.

Fix a carried source point $m : V$. Source inverse closure makes $-_V m$ carried, and the source right-inverse row gives

$$m +_V (-_V m) \sim_V 0_V.$$

This source comparison is available for evaluation by the semantic transport of Definition 26.3. The function-with-law data of Definition 48.1 supply output closure, additive preservation, and the evaluation descent used through Definition 26.19. Descent transports the source inverse comparison to

$$f(m +_V (-_V m)) \sim_W f(0_V).$$

The additive row of f at the carried pair $m, -_V m$ gives

$$f(m +_V (-_V m)) \sim_W f(m) +_W f(-_V m).$$

The zero-preservation theorem Definition 48.57, applied to the same additive row, gives

$$f(0_V) \sim_W 0_W.$$

Classifier symmetry from Definition 9.7 and transitivity from Definition 9.8 compose the last three comparisons into

$$f(m) +_W f(-_V m) \sim_W 0_W.$$

Output closure makes $f(m)$ and $f(-_V m)$ carried in W, and codomain inverse closure makes $-_W f(m)$ carried. The codomain left-inverse row gives

$$(-_W f(m)) +_W f(m) \sim_W 0_W.$$

Apply Definition 41.59 in the codomain additive group reduct with $x = f(m)$, $y = -_W f(m)$, and $z = f(-_V m)$. The two displayed inverse witnesses give

$$-_W f(m) \sim_W f(-_V m).$$

One final classifier symmetry gives

$$f(-_V m) \sim_W -_W f(m).$$

The ledger entries are the source inverse comparison, the evaluation descent witness, the additive row, the zero-preservation row, the codomain inverse row, and the inverse-uniqueness comparison required by Definition 48.51. □

Theorem 48.70 (LinearMap difference preservation)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let V and W carry $\mathsf{VecSpace}^{\uparrow}(F, V)$ and $\mathsf{VecSpace}^{\uparrow}(F, W)$ certificates. Write $+_W$, $-_W$, and $\sim_W$ for the codomain vector addition, additive inverse, and classifier. Let $f, g : V \to W$ be carried $\mathsf{LinMap}^{\uparrow}(V, W)$ maps. Define $d : V \to W$ pointwise by*

$$d(m) := f(m) +_W (-_W g(m)).$$

Then d carries a $\mathsf{LinMap}^{\uparrow}(V, W)$ certificate, and every carried source point $m : V$ satisfies

$$d(m) \sim_W f(m) +_W (-_W g(m)).$$

The ten-field fit is: BEDC source $\mathsf{LinMap}^{\uparrow} + \mathsf{VecSpace}^{\uparrow} + \mathsf{Field}^{\uparrow}$ over the common scalar field and the two vector-space certificates; carrier the pointwise difference function d; classifier the pointwise codomain classifier from Definition 48.49; stability the sum rows together with the codomain inverse rows; ledger the pointwise-sum ledger plus the pointwise-inverse ledger; core theorem this block; exactness predicate the displayed endpoint formula; standard bridge the usual difference of linear maps in $\mathrm{Hom}_F(V, W)$; public pattern the hom-space additive-group operation; status $\mathsf{ProofTarget}$.

Proof. Read the two vector-space certificates through their module reducts by Definition 47.24. The carried $\mathsf{LinMap}^{\uparrow}(V, W)$ data for f and g provide the corresponding module linear-map certificates over this common module source.

Apply Definition 48.68 to g. This certifies the pointwise inverse map $u : V \to W$ defined by

$$u(m) := -_W g(m).$$

The endpoint identity for u is the codomain additive inverse operation; the source-inverse behavior remains compatible with Definition 48.69.

Now apply Definition 48.66 to the certified maps f and u. Its carrier is the pointwise sum

$$m \longmapsto f(m) +_W u(m),$$

which is exactly the displayed difference carrier d after substituting the definition of u.

For a carried source point $m : V$, the classifier for $\mathsf{LinMap}^{\uparrow}(V, W)$ is pointwise in $\sim_W$ by Definition 48.49. The displayed defining equation for d and classifier reflexivity at the carried codomain endpoint give

$$d(m) \sim_W f(m) +_W (-_W g(m)).$$

The ledger is the concatenation of the pointwise-inverse ledger for g and the pointwise-sum ledger for f with that inverse map. Thus all carrier, classifier, stability, exactness, and ledger rows required for the $\mathsf{LinMap}^{\uparrow}(V, W)$ certificate are supplied. □

Definition 48.71 (Module LinearMap kernel and image restricted module classifiers)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source, and let*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

hold. Use the kernel and image predicates of Definition 46.43 and Definition 48.64. The kernel classifier is the source module classifier restricted to the zero fiber:

$$x \sim_{\mathsf{Ker}_f} x' \iff \mathsf{Ker}_f(x) \wedge \mathsf{Ker}_f(x') \wedge x \sim_M x'.$$

The image classifier is the target module classifier restricted to the image carrier:

$$y \sim_{\mathsf{Im}_f} y' \iff \mathsf{Im}_f(y) \wedge \mathsf{Im}_f(y') \wedge y \sim_N y'.$$

The kernel module fragment uses $+_M$, 0_M, $-_M$, and $\odot_M$. The image module fragment uses $+_N$, 0_N, $-_N$, and $\odot_N$. In both fragments the scalar carrier, scalar operations, scalar classifier, and scalar ledger are copied from the ambient $\mathsf{Ring}^{\uparrow}(R)$ certificate.

Theorem 48.72 (Kernel and image inherit module classifiers)**.** *Under the hypotheses of Definition 48.71, the predicate-defined carriers Ker_f and Im_f are stable under their ambient module classifiers, contain the corresponding zero element, and are closed under addition, additive negation, and carried scalar action. With the restricted classifiers of Definition 48.71, these data form inherited $\mathsf{Module}^{\uparrow}$ certificate packages over the same scalar source R.*

Proof. Unpack $\mathsf{LinearMapCert}_R(M, N; f)$ by Definition 48.59 and unpack the two module certificates through Definition 46.20. The kernel transport, zero, addition, and scalar-action rows are exactly Definition 46.44. The image transport, zero, addition, and scalar-action rows are exactly Definition 48.65.

It remains to record additive negation and the inherited classifier packages. For the kernel, assume $\mathsf{Ker}_f(x)$. The scalar source is a certified $\mathsf{Ring}^{\uparrow}$ by Definition 46.15, so the additive inverse row carries $-_R 1_R$. Applying the scalar-action closure row of Definition 46.44 gives

$$\mathsf{Ker}_f((-_R 1_R) \odot_M x).$$

In the source module, Definition 46.42 at $r = 1_R$ gives

$$((-_R 1_R) \odot_M x) +_M (1_R \odot_M x) \sim_M 0_M.$$

The scalar unit row from Definition 46.18, together with additive congruence in the additive group retained by Definition 46.26, replaces $1_R \odot_M x$ by x. The source right-inverse row gives $x +_M (-_M x) \sim_M 0_M$. Applying Definition 41.59 in the source additive group reduct obtained through Definition 42.33 identifies

$$(-_R 1_R) \odot_M x \sim_M -_M x.$$

The source-classifier transport row of Definition 46.44 then gives $\mathsf{Ker}_f(-_M x)$.

For the image, assume $\mathsf{Im}_f(y)$. By the definition of the image predicate, choose a carried source witness x with $f(x) \sim_N y$. Output closure for f and carrier transport for the target classifier make y a carried target endpoint. The image scalar-action row of Definition 48.65, applied to the carried scalar $-_R 1_R$, gives

$$\mathsf{Im}_f((-_R 1_R) \odot_N y).$$

The same inverse-side action argument in the target module gives

$$(-_R 1_R) \odot_N y \sim_N -_N y.$$

The target-classifier transport row of Definition 48.65 therefore gives $\mathsf{Im}_f(-_N y)$.

The restricted kernel classifier is a naming certificate on Ker_f: reflexivity, symmetry, and transitivity are the corresponding source-module classifier rows from Definition 9.7 and Definition 9.8, while carrier transport is the first row of Definition 46.44. The restricted image classifier is handled identically, using the target-module classifier and the transport row of Definition 48.65.

Finally, every $\mathsf{Module}^{\uparrow}$ law on the restricted carriers is the ambient module law with its endpoints repackaged by the restricted classifier. Closure under zero, addition, negation, and scalar action supplies the two carrier components of each restricted classifier witness. The ambient comparison component is supplied by the source module for the kernel and by the target module for the image: additive associativity, additive unit and inverse laws, additive commutativity, additive congruence, scalar-action congruence, scalar associativity, both distributivity rows, and the scalar unit row are all listed in Definition 46.18 together with the additive $\mathsf{AbGroup}^{\uparrow}$ source retained by Definition 46.26. The scalar ring certificate is unchanged. The ledgers are the ambient operation ledgers together with the kernel or image witness carried in the restricted classifier, matching Definition 46.19. Hence both predicate carriers inherit $\mathsf{Module}^{\uparrow}$ classifier packages over R. □

Theorem 48.73 (Module LinearMap trivial kernel classifier injectivity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source, and let*

$$\mathsf{LinearMapCert}_R(M, N; f)$$

hold. Write $\sim_M$ and $\sim_N$ for the source and target module classifiers, 0_M and 0_N for the source and target zeros, $+_M$ for source addition, and $-_M$ for source additive inverse. Define classifier-injectivity on carried source endpoints by

$$f(x) \sim_N f(y) \Longrightarrow x \sim_M y.$$

Then f is classifier-injective exactly when its BEDC kernel from Definition 46.43 is trivial:

$$\mathsf{Ker}_f(x) \Longleftrightarrow x \sim_M 0_M$$

for every carried source endpoint $x : M$.

The ten-field fit is: traditional target the ordinary criterion that a module homomorphism is injective exactly when its kernel is zero; BEDC source $\mathsf{LinearMapCert}_R(M, N; f)$ over the two $\mathsf{Module}^{\uparrow}$ certificates; carrier the source module carrier; classifier $\sim_M$; stability zero preservation, additivity, additive inverse preservation, and additive cancellation from the module additive reducts; ledger the function identifier f, its descent, zero, additivity, inverse, and source-cancellation rows; core theorem this block; exactness predicate the equivalence between classifier-injectivity and the trivial zero fiber; standard bridge first-isomorphism-style exactness for $\mathsf{LinMap}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Proof. Assume first that f is classifier-injective on carried source endpoints. Let x be carried in M. If $\mathsf{Ker}_f(x)$, then Definition 46.43 gives $f(x) \sim_N 0_N$. Zero preservation from Definition 48.59 gives $f(0_M) \sim_N 0_N$, and classifier symmetry and transitivity in N compose these rows to obtain $f(x) \sim_N f(0_M)$. Classifier-injectivity gives $x \sim_M 0_M$.

Conversely, if $x \sim_M 0_M$, source-classifier descent for f gives $f(x) \sim_N f(0_M)$. Composing this with zero preservation yields $f(x) \sim_N 0_N$, so the carried endpoint x satisfies $\mathsf{Ker}_f(x)$.

Assume now that every carried $z : M$ satisfies $\mathsf{Ker}_f(z) \Longleftrightarrow z \sim_M 0_M$. Let x and y be carried source endpoints with $f(x) \sim_N f(y)$. Source additive inverse closure carries $-_M y$, and source addition closure carries $x +_M (-_M y)$.

Additivity gives

$$f(x +_M (-_M y)) \sim_N f(x) +_N f(-_M y).$$

The additive-inverse argument of Definition 48.69, read through the additive group reducts of the module certificates, identifies $f(-_M y)$ with the target inverse of $f(y)$. Target additive congruence then transports $f(x) \sim_N f(y)$ to a comparison of $f(x) +_N (-_N f(y))$ with $f(y) +_N (-_N f(y))$, and the target right-inverse row reduces the latter endpoint to 0_N.

Thus $x +_M (-_M y)$ lies in Ker_f. By the trivial-kernel hypothesis,

$$x +_M (-_M y) \sim_M 0_M.$$

The source right-inverse row also gives $y +_M (-_M y) \sim_M 0_M$. Classifier symmetry and transitivity produce

$$x +_M (-_M y) \sim_M y +_M (-_M y).$$

Apply Definition 41.116 in the source additive group reduct obtained from Definition 46.26 and Definition 42.33. Cancelling the common carried summand $-_M y$ gives $x \sim_M y$, which is classifier-injectivity. □

Theorem 48.74 (Module LinearMap pointwise sum commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar*

source. Write $\sim_N$ for the module classifier on N and $+_N$ for module addition. For functions $f, g : M \to N$, define $h, k : M \to N$ by

$$h(x) := f(x) +_N g(x), \qquad k(x) := g(x) +_N f(x).$$

If

$$\mathsf{LinearMapCert}_R(M, N; f) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, N; g),$$

then both pointwise sums carry certificates

$$\mathsf{LinearMapCert}_R(M, N; h) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, N; k),$$

and they are related by the pointwise $\mathsf{LinMap}^{\uparrow}(M, N)$ *classifier:*

$$h \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} k.$$

Equivalently, for every carried $x : M$,

$$f(x) +_N g(x) \sim_N g(x) +_N f(x).$$

The ten-field fit is: traditional target commutativity of pointwise addition in $\mathrm{Hom}_R(M, N)$*; BEDC source the two displayed* $\mathsf{LinearMapCert}_R$ *packages over common* $\mathsf{Module}^{\uparrow}$ *certificates; carrier the two pointwise-sum functions h and k; classifier the pointwise target classifier from Definition 48.49; stability pointwise-sum closure and target additive commutativity; ledger the two input function identifiers, the two pointwise-sum ledgers, and the target additive-commutativity ledger at each carried source endpoint; core theorem this block; exactness predicate the displayed* $\mathsf{LinMap}^{\uparrow}(M, N)$ *classifier comparison; standard bridge the usual commutative addition law on module homomorphisms; status* $\mathsf{ProofTarget}$.

Proof. Apply Definition 48.66 to the ordered pair (f, g). This gives the certificate for h. Apply the same theorem to the ordered pair (g, f). This gives the certificate for k.

It remains to prove the classifier comparison. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier is pointwise comparison in the target module classifier, together with the two carried map-code endpoints. The two endpoint certificate clauses are the certificates for h and k just obtained.

Fix a carried source endpoint $x : M$. Unpacking the two input certificates by Definition 48.59, output closure for f and for g makes $f(x)$ and $g(x)$ carried in N. By Definition 46.26, the additive part of the target module is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same operation $+_N$ and classifier $\sim_N$. The commutativity row listed in Definition 42.8 therefore gives

$$f(x) +_N g(x) \sim_N g(x) +_N f(x).$$

By the definitions of h and k, this is exactly

$$h(x) \sim_N k(x).$$

Since x was arbitrary, Definition 48.49 packages these pointwise target comparisons, together with the two carried endpoint certificates, as

$$h \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} k.$$

The ledger is the pair of pointwise-sum ledgers supplied by Definition 48.66, plus the target additive-commutativity ledger in the abelian-group reduct of N, matching Definition 48.51. □

Theorem 48.75 (Module zero LinearMap certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_M$ and $\sim_N$ for the module classifiers, $+_M$ and $+_N$ for module addition, 0_M and 0_N for module zero, and $\odot_M$ and $\odot_N$ for scalar action. Let $z : M \to N$ be a function. Assume that for every carried $x : M$, the output $z(x)$ is carried in N and*

$$z(x) \sim_N 0_N.$$

Then

$$\mathsf{LinearMapCert}_R(M, N; z)$$

holds.

Proof. Unpack the two module certificates through Definition 46.20 and read the required rows through Definition 48.59. The carried-output part of the hypothesis is the output-closure row. Since 0_M is carried by the additive source of M, the same pointwise zero row at $x = 0_M$ gives $z(0_M) \sim_N 0_N$, which is zero preservation.

For source-classifier descent, assume $x \sim_M y$ between carried source endpoints. The pointwise zero row gives $z(x) \sim_N 0_N$ and $z(y) \sim_N 0_N$. Classifier symmetry from Definition 9.7, followed by transitivity from Definition 9.8, gives $z(x) \sim_N z(y)$.

For additivity, let x and y be carried in M. Source additive closure makes $x +_M y$ carried, so $z(x +_M y) \sim_N 0_N$. The target additive congruence row from Definition 46.18 transports the two zero comparisons for $z(x)$ and $z(y)$ to

$$z(x) +_N z(y) \sim_N 0_N +_N 0_N \sim_N 0_N,$$

where the second comparison is the target additive unit row retained by Definition 46.26. Symmetry and transitivity compose this with $z(x +_M y) \sim_N 0_N$, yielding

$$z(x +_M y) \sim_N z(x) +_N z(y).$$

For scalar compatibility, let $r : R$ be carried and let $x : M$ be carried. Source scalar-action closure makes $r \odot_M x$ carried, so $z(r \odot_M x) \sim_N 0_N$. Scalar-action congruence in the target module, using scalar reflexivity for r and $z(x) \sim_N 0_N$, gives $r \odot_N z(x) \sim_N r \odot_N 0_N$. The zero-action annihilation theorem Definition 46.29, applied in N, gives $r \odot_N 0_N \sim_N 0_N$. Symmetry and transitivity therefore give

$$z(r \odot_M x) \sim_N r \odot_N z(x).$$

The pointwise classifier for maps $M \to N$ is the one in Definition 48.49. The ledger records the function identifier z, the pointwise zero witnesses, the target additive and scalar-action congruence rows, the target zero-action row, and the classifier symmetry and transitivity witnesses, matching Definition 48.51. Hence the carrier, source, pattern, classifier, stability, and ledger obligations of Definition 48.52 are satisfied for z. □

Theorem 48.76 (Module LinearMap pointwise sum associativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_N$ for the module classifier on N and $+_N$ for module addition. For functions $f, g, h : M \to N$, define $u, v : M \to N$ by*

$$u(x) := (f(x) +_N g(x)) +_N h(x), \qquad v(x) := f(x) +_N (g(x) +_N h(x)).$$

If

$$\mathsf{LinearMapCert}_R(M,N;f) \quad \textit{and} \quad \mathsf{LinearMapCert}_R(M,N;g) \quad \textit{and} \quad \mathsf{LinearMapCert}_R(M,N;h),$$

then both associated pointwise sums carry certificates

$$\mathsf{LinearMapCert}_R(M,N;u) \quad \textit{and} \quad \mathsf{LinearMapCert}_R(M,N;v),$$

and they are related by the pointwise $\mathsf{LinMap}^{\uparrow}(M,N)$ *classifier:*

$$u \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} v.$$

Equivalently, for every carried $x : M$*,*

$$(f(x) +_N g(x)) +_N h(x) \sim_N f(x) +_N (g(x) +_N h(x)).$$

The ten-field fit is: traditional target associativity of pointwise addition in $\mathrm{Hom}_R(M,N)$*; BEDC source the three displayed* $\mathsf{LinearMapCert}_R$ *packages over common* $\mathsf{Module}^{\uparrow}$ *certificates; carrier the two associated pointwise-sum functions* u *and* v*; classifier the pointwise target classifier from Definition 48.49; stability pointwise-sum closure and target additive associativity; ledger the two binary pointwise-sum ledgers on each side and the target additive-associativity ledger at each carried source endpoint; core theorem this block; exactness predicate the displayed* $\mathsf{LinMap}^{\uparrow}(M,N)$ *classifier comparison; standard bridge the usual associative addition law on module homomorphisms; status* $\mathsf{ProofTarget}$*.*

Proof. Define the intermediate pointwise sums $s, t : M \to N$ by

$$s(x) := f(x) +_N g(x), \qquad t(x) := g(x) +_N h(x).$$

Apply Definition 48.66 to the ordered pair (f, g). This gives

$$\mathsf{LinearMapCert}_R(M,N;s).$$

Applying the same theorem to the ordered pair (s, h) gives

$$\mathsf{LinearMapCert}_R(M,N;u),$$

because the resulting pointwise-sum function is exactly $x \mapsto (f(x) +_N g(x)) +_N h(x)$.

For the other association, apply Definition 48.66 to the ordered pair (g, h), obtaining

$$\mathsf{LinearMapCert}_R(M,N;t).$$

Applying the same theorem to the ordered pair (f, t) gives

$$\mathsf{LinearMapCert}_R(M,N;v),$$

because the resulting pointwise-sum function is exactly $x \mapsto f(x) +_N (g(x) +_N h(x))$.

It remains to compare the two carried pointwise sums. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}(M,N)$ classifier is pointwise comparison in the target module classifier, together with the two carried map-code endpoints. The carried endpoint clauses are the two certificates for u and v just obtained.

Fix a carried source endpoint $x : M$. Unpacking the three input certificates by Definition 48.59, output closure for f, g, and h makes $f(x)$, $g(x)$, and $h(x)$ carried in N. By Definition 46.26, the

additive part of the target module is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same operation $+_N$ and classifier $\sim_N$. The additive associativity row listed in Definition 42.8 therefore gives

$$(f(x) +_N g(x)) +_N h(x) \sim_N f(x) +_N (g(x) +_N h(x)).$$

By the definitions of u and v, this is exactly

$$u(x) \sim_N v(x).$$

Since x was arbitrary, Definition 48.49 packages these pointwise target comparisons, together with the two carried endpoint certificates, as

$$u \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} v.$$

The ledger is the pair of binary pointwise-sum ledgers on each associated side supplied by Definition 48.66, plus the target additive-associativity ledger in the abelian-group reduct of N, matching Definition 48.51. □

Theorem 48.77 (Module LinearMap pointwise zero additive identity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_N$ for the module classifier on N, write $+_N$ for module addition, and write 0_N for module zero. For functions $f, z : M \to N$, define $p, q : M \to N$ by*

$$p(x) := f(x) +_N z(x), \qquad q(x) := z(x) +_N f(x).$$

Assume that

$$\mathsf{LinearMapCert}_R(M, N; f)$$

holds, and assume that z is pointwise zero: for every carried $x : M$, the output $z(x)$ is carried in N and

$$z(x) \sim_N 0_N.$$

Then z, p, and q carry LinearMap certificates,

$$\mathsf{LinearMapCert}_R(M, N; z), \qquad \mathsf{LinearMapCert}_R(M, N; p), \qquad \mathsf{LinearMapCert}_R(M, N; q),$$

and the two pointwise sums are classified with f in $\mathsf{LinMap}^{\uparrow}(M, N)$:

$$p \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f \qquad \text{and} \qquad q \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f.$$

Equivalently, for every carried $x : M$,

$$f(x) +_N z(x) \sim_N f(x) \qquad \text{and} \qquad z(x) +_N f(x) \sim_N f(x).$$

The ten-field fit is: traditional target the additive identity law in $\mathrm{Hom}_R(M, N)$; BEDC source the displayed $\mathsf{LinearMapCert}_R(M, N; f)$ package and the pointwise-zero rows for z over common $\mathsf{Module}^{\uparrow}$ certificates; carrier the pointwise-zero map z and the two pointwise-sum functions p and q; classifier the pointwise target classifier from Definition 48.49; stability zero-map certification, pointwise-sum closure, and target additive unit laws; ledger the input function identifier, the pointwise-zero witnesses, the two pointwise-sum ledgers, and the left and right target additive-unit ledgers at each carried source endpoint; core theorem this block; exactness predicate the two displayed $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier comparisons; standard bridge the usual zero identity law for module homomorphisms; status ProofTarget.

Proof. Apply Definition 48.75 to the pointwise-zero rows for z. This gives

$$\mathsf{LinearMapCert}_R(M, N; z).$$

Apply Definition 48.66 to the ordered pair (f, z). This gives the certificate for p. Apply the same pointwise-sum theorem to the ordered pair (z, f). This gives the certificate for q.

It remains to prove the two classifier comparisons with f. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier is pointwise comparison in the target module classifier, together with the carried map-code endpoints. The endpoint clauses are the certificate for f assumed above and the certificates for p and q just obtained.

Fix a carried source endpoint $x : M$. Unpacking the certificate for f through Definition 48.59, output closure makes $f(x)$ carried in N. The pointwise-zero hypothesis makes $z(x)$ carried in N and gives

$$z(x) \sim_N 0_N.$$

By Definition 46.26, the additive part of the target module is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same operation $+_N$, zero 0_N, and classifier $\sim_N$. Classifier reflexivity for the carried endpoint $f(x)$, together with target additive congruence, gives

$$f(x) +_N z(x) \sim_N f(x) +_N 0_N.$$

The right additive-unit row in the retained abelian-group certificate gives

$$f(x) +_N 0_N \sim_N f(x).$$

Transitivity of the target classifier yields

$$p(x) = f(x) +_N z(x) \sim_N f(x).$$

Similarly, target additive congruence applied to $z(x) \sim_N 0_N$ and reflexivity of $f(x)$ gives

$$z(x) +_N f(x) \sim_N 0_N +_N f(x).$$

The left additive-unit row gives

$$0_N +_N f(x) \sim_N f(x),$$

and transitivity yields

$$q(x) = z(x) +_N f(x) \sim_N f(x).$$

Since x was arbitrary, Definition 48.49 packages these two families of target comparisons, together with the carried endpoint certificates, as

$$p \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f \qquad \text{and} \qquad q \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} f.$$

The ledger is assembled from the zero-map certificate supplied by Definition 48.75, the two pointwise-sum certificates supplied by Definition 48.66, the target additive congruence rows, and the left and right additive-unit ledgers in the abelian-group reduct of N, matching Definition 48.51. □

Theorem 48.78 (Module LinearMap pointwise inverse sum zero classifier)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_N$ for the module classifier on N, $+_N$ for module addition, 0_N for module*

zero, and $-_N$ for target additive inverse. Let $f, z : M \to N$ be functions, and define $h, s, t : M \to N$ by

$$h(x) := -_N f(x), \qquad s(x) := f(x) +_N h(x), \qquad t(x) := h(x) +_N f(x).$$

Assume

$$\mathsf{LinearMapCert}_R(M, N; f)$$

and assume that z satisfies the zero-linear-map hypotheses of Definition 48.75: for every carried $x : M$, the output $z(x)$ is carried in N and

$$z(x) \sim_N 0_N.$$

Then the pointwise inverse and the two inverse sums carry certified linear-map packages

$$\mathsf{LinearMapCert}_R(M, N; h), \qquad \mathsf{LinearMapCert}_R(M, N; s), \qquad \mathsf{LinearMapCert}_R(M, N; t),$$

the zero map z carries a certified zero-linear-map package

$$\mathsf{LinearMapCert}_R(M, N; z),$$

and the two inverse sums are classified with z by the pointwise $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier:

$$s \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z \qquad \text{and} \qquad t \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z.$$

Equivalently, for every carried $x : M$,

$$f(x) +_N (-_N f(x)) \sim_N 0_N \qquad \text{and} \qquad (-_N f(x)) +_N f(x) \sim_N 0_N,$$

and these pointwise zero rows classify both pointwise sums with every certified pointwise-zero representative z. The ten-field fit is: traditional target the additive inverse law in $\mathrm{Hom}_R(M, N)$; BEDC source the displayed $\mathsf{LinearMapCert}_R(M, N; f)$ package, the pointwise inverse construction, and the pointwise-zero witnesses for z; carrier the three functions h, s, and t; classifier the pointwise target classifier from Definition 48.49; stability pointwise inverse closure, pointwise-sum closure, target additive inverse laws, and zero-map classification; ledger the input function identifier, the inverse ledger, the two pointwise-sum ledgers, the target inverse-law ledgers at each carried source endpoint, and the zero-classifier comparison ledger; core theorem this block; exactness predicate the two displayed $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier comparisons; standard bridge the usual equality $f + (-f) = 0 = (-f) + f$ in the additive group of module homomorphisms; status ProofTarget.

Proof. Apply Definition 48.68 to the input certificate for f. This gives

$$\mathsf{LinearMapCert}_R(M, N; h).$$

Apply Definition 48.66 to the ordered pair (f, h) and then to the ordered pair (h, f). This gives

$$\mathsf{LinearMapCert}_R(M, N; s) \qquad \text{and} \qquad \mathsf{LinearMapCert}_R(M, N; t).$$

It remains to identify these two certified sums with the chosen zero representative z. By Definition 48.80, it is enough to verify the zero-linear-map hypotheses for s and for t, because the same hypotheses are assumed for z.

Fix a carried source endpoint $x : M$. Unpacking the certificate for f through Definition 48.59, output closure makes $f(x)$ carried in N. The additive inverse closure row in the target module,

read through Definition 46.26, makes $h(x) = -_N f(x)$ carried. Target additive closure then makes both

$$s(x) = f(x) +_N h(x) \qquad \text{and} \qquad t(x) = h(x) +_N f(x)$$

carried in N.

The additive part of N is an $\mathsf{AbGroup}^{\uparrow}(N)$ certificate with the same $+_N$, 0_N, $-_N$, and $\sim_N$ by Definition 46.26. The right-inverse and left-inverse rows listed in Definition 42.8, applied to the carried endpoint $f(x)$, give

$$f(x) +_N (-_N f(x)) \sim_N 0_N \qquad \text{and} \qquad (-_N f(x)) +_N f(x) \sim_N 0_N.$$

By the definitions of h, s, and t, these are exactly

$$s(x) \sim_N 0_N \qquad \text{and} \qquad t(x) \sim_N 0_N.$$

Thus s and t satisfy the carried-output and pointwise-zero hypotheses of Definition 48.75.

Apply Definition 48.80 to the pair (s, z). The zero hypotheses for s are the rows just proved, and the zero hypotheses for z are assumed. This supplies the zero-linear-map certificate for z and the classifier comparison

$$s \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z.$$

Apply the same theorem to the pair (t, z). This supplies the classifier comparison

$$t \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z.$$

The certificate clauses for s and t are also the pointwise-sum certificates already obtained from Definition 48.66. The ledger is the inverse certificate ledger for h, the two pointwise-sum ledgers for s and t, the target additive inverse-law ledgers at each carried input, and the zero-classifier uniqueness ledger of Definition 48.80, matching Definition 48.51. □

Theorem 48.79 (Module zero LinearMap image is the zero fiber)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, and let M and N carry $\mathsf{Module}^{\uparrow}(R, M)$ and $\mathsf{Module}^{\uparrow}(R, N)$ certificates over the same scalar source. Write $\sim_N$ for the module classifier on N, and write 0_M and 0_N for the source and target zeros. Let $z : M \to N$ be a function satisfying the zero-linear-map hypotheses of Definition 48.75: for every carried $x : M$, the output $z(x)$ is carried in N and*

$$z(x) \sim_N 0_N.$$

Then z carries a certified linear-map package

$$\mathsf{LinearMapCert}_R(M, N; z),$$

and its image predicate from Definition 48.64 is exactly the target zero fiber. More precisely, for every endpoint $y : N$,

$$\mathsf{Im}_z(y) \Longrightarrow y \text{ is carried in } N \wedge y \sim_N 0_N,$$

and for every carried endpoint $y : N$,

$$y \sim_N 0_N \Longrightarrow \mathsf{Im}_z(y).$$

The ten-field fit is: traditional target the image of the zero module homomorphism is the zero submodule; BEDC source the displayed pointwise-zero hypotheses over common $\mathsf{Module}^{\uparrow}$ certificates;

carrier the image predicate Im_z*; classifier the target module classifier* $\sim_N$*; stability the carried target zero, carrier transport, and classifier symmetry and transitivity rows; ledger the source-zero witness* 0_M*, the pointwise zero row for* z*, and the image witness from Definition 48.64; core theorem this block; exactness predicate the equivalence between image membership and the target zero fiber on carried endpoints; standard bridge the ordinary statement* $\mathrm{im}(0) = \{0\}$ *for module homomorphisms; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 48.75 to the pointwise-zero hypotheses for z. This gives

$$\mathsf{LinearMapCert}_R(M, N; z).$$

Assume first that $\mathsf{Im}_z(y)$. By Definition 48.64, choose a carried source witness $x : M$ with

$$z(x) \sim_N y.$$

The pointwise-zero hypothesis gives that $z(x)$ is carried in N and gives

$$z(x) \sim_N 0_N.$$

Carrier transport for the target naming certificate in Definition 9.7, applied to $z(x) \sim_N y$, makes y carried in N. Classifier symmetry changes $z(x) \sim_N y$ into $y \sim_N z(x)$. Composing this with $z(x) \sim_N 0_N$ by Definition 9.8 gives

$$y \sim_N 0_N.$$

Thus image membership forces membership in the target zero fiber.

Conversely, let y be carried in N and assume

$$y \sim_N 0_N.$$

The source module certificate, unpacked through Definition 46.20, carries 0_M. Applying the pointwise-zero hypothesis at 0_M gives a carried target endpoint $z(0_M)$ and a comparison

$$z(0_M) \sim_N 0_N.$$

Classifier symmetry changes the assumed comparison into $0_N \sim_N y$. Transitivity in the target classifier then gives

$$z(0_M) \sim_N y.$$

The carried source endpoint 0_M, together with this comparison, is an image witness in the sense of Definition 48.64. Hence $\mathsf{Im}_z(y)$.

The two implications identify Im_z with the target zero fiber on carried target endpoints. The ledger uses one displayed image witness in the first direction and the distinguished carried source-zero witness in the second direction, together with the pointwise-zero row for z and the target classifier symmetry and transitivity rows, matching Definition 48.51. □

Theorem 48.80 (Module zero LinearMap classifier uniqueness)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}$ *scalar certificate, and let* M *and* N *carry* $\mathsf{Module}^{\uparrow}(R, M)$ *and* $\mathsf{Module}^{\uparrow}(R, N)$ *certificates over the same scalar source. Write* $\sim_N$ *for the codomain module classifier and* 0_N *for the codomain zero. Let* $z, z' : M \to N$ *be functions. Assume that both functions satisfy the zero-linear-map hypotheses of Definition 48.75: for every carried* $x : M$*, the outputs* $z(x)$ *and* $z'(x)$ *are carried in* N *and*

$$z(x) \sim_N 0_N \qquad \textit{and} \qquad z'(x) \sim_N 0_N.$$

Then both functions carry certified zero linear-map packages

$$\mathsf{LinearMapCert}_R(M, N; z) \qquad \textit{and} \qquad \mathsf{LinearMapCert}_R(M, N; z'),$$

and the pointwise $\mathsf{LinMap}^{\uparrow}(M, N)$ *classifier identifies them:*

$$z \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z'.$$

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapSingleton zero classifier uniqueness`

Proof. Apply Definition 48.75 to the pointwise zero witnesses for z. This gives $\mathsf{LinearMapCert}_R(M, N; z)$. Apply the same theorem to the pointwise zero witnesses for z'. This gives $\mathsf{LinearMapCert}_R(M, N; z')$.

It remains to compare the two carried map endpoints. By Definition 48.49, the $\mathsf{LinMap}^{\uparrow}(M, N)$ classifier is pointwise comparison in the codomain module classifier, together with the two carried linear-map endpoints just obtained.

Fix a carried source endpoint $x : M$. The hypotheses give

$$z(x) \sim_N 0_N \qquad \text{and} \qquad z'(x) \sim_N 0_N.$$

Classifier symmetry in the codomain $\mathsf{NameCert}$ changes the second comparison into $0_N \sim_N z'(x)$. Codomain transitivity, as recorded by Definition 9.8, composes it with $z(x) \sim_N 0_N$ and yields

$$z(x) \sim_N z'(x).$$

Since x was arbitrary, Definition 48.49 packages these pointwise codomain comparisons, together with the two certified endpoint packages, as

$$z \sim_{\mathsf{LinMap}^{\uparrow}(M,N)} z'.$$

The ledger is the pair of zero-linear-map ledgers supplied by Definition 48.75, plus the codomain classifier symmetry and transitivity witnesses, matching Definition 48.51. □

Definition 48.81 (LinearMap evaluation-graph carrier)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let V and W carry $\mathsf{VecSpace}^{\uparrow}(F, V)$ and $\mathsf{VecSpace}^{\uparrow}(F, W)$ certificates over the same scalar source. A $\mathsf{LinMap}^{\uparrow}(V, W)$ evaluation-graph packet is a finite BHist family with the following rows:*

1. *a source vector-space row naming the carrier, addition, zero, additive inverse, scalar action, and classifier of V;*
2. *a target vector-space row naming the corresponding data of W;*
3. *an evaluation graph row G_f whose entries record carried source endpoints $x : V$, carried target endpoints $y : W$, and graph witnesses $f(x) \sim_W y$;*
4. *an additivity ledger row stating that every carried pair $x, u : V$ has a graph witness for $x +_V u$ whose target endpoint is classified with $f(x) +_W f(u)$;*
5. *a scalar-compatibility ledger row stating that every carried scalar $a : F$ and carried $x : V$ have a graph witness for $a \odot_V x$ whose target endpoint is classified with $a \odot_W f(x)$;*
6. *a pointwise classifier row comparing two such graph packets by target-classifier comparison at each carried source endpoint;*
7. *continuation rows showing that evaluation, additivity, and scalar-compatibility entries remain visible after the relevant Cont routes and context appends;*

8. *a* Pkg *provenance row recording the function identifier, the source and target vector-space packages, the three graph ledgers, and the finite history addresses used by the packet;*
9. *a* NameCert *row asserting reflexivity, symmetry, transitivity, carrier transport, and ledger exactness for the pointwise graph classifier.*

Theorem 48.82 (LinearMap evaluation-graph NameCert obligations)**.** *Every evaluation-graph packet of Definition 48.81 supplies the local* $\mathsf{LinMap}^{\uparrow}(V, W)$ *NameCert obligation surface: carrier admission for the map-code, pointwise classifier transport along the target* NameCert*, additivity stability, scalar-compatibility stability, and exact recovery of the displayed graph, additivity, scalar, continuation,* Pkg*, and naming ledgers. Conversely, any finite* BHist *packet carrying precisely those rows is admitted by the evaluation-graph carrier and determines the same pointwise* $\mathsf{LinMap}^{\uparrow}(V, W)$ *classifier.*

Proof. Carrier admission reads the source and target vector-space rows through the $\mathsf{VecSpace}^{\uparrow}$ certificates and then reads their module fragments through Definition 47.24. The output-closure part of the graph row gives the carried target endpoint for each carried source input, so the packet has the map-code carrier required by Definition 48.1.

Pointwise classifier transport is inherited from the target NameCert. For each carried source endpoint, the graph row gives a target classifier comparison, and Definition 48.49 packages those target comparisons as the $\mathsf{LinMap}^{\uparrow}(V, W)$ classifier. Reflexivity, symmetry, and transitivity are the pointwise rows isolated in Definition 48.55.

Additivity and scalar compatibility are the two stability ledgers. The additivity row is the same law used in Definition 48.50; the scalar row is the vector-space form of the module certificate in Definition 48.59. The zero and inverse consequences available in Definition 48.75, Definition 48.80, and Definition 48.69 show that no separate function-extensional equality is needed.

Finally, the continuation rows are checked by the singleton evaluation and composition continuation surfaces in Definition 48.38, Definition 48.39, and Definition 48.44. Since every public read is one of the listed BHist rows, the Pkg row and the NameCert row recover exactly the carrier, classifier, stability, and ledger fields of Definition 48.52. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapEvaluationGraphNameCertObligations`

Theorem 48.83 (LinearMap evaluation-graph composition closure)**.** *Let* F *carry a* $\mathsf{Field}^{\uparrow}(F)$ *certificate, and let* U*,* V*, and* W *carry* $\mathsf{VecSpace}^{\uparrow}(F, U)$*,* $\mathsf{VecSpace}^{\uparrow}(F, V)$*, and* $\mathsf{VecSpace}^{\uparrow}(F, W)$ *certificates over the same scalar source. If* G_f *is an accepted evaluation-graph packet for* $f : U \to V$ *and* G_g *is an accepted evaluation-graph packet for* $g : V \to W$*, with the target vector-space row of* G_f *matching the source vector-space row of* G_g*, then the* Cont*-spliced graph packet for* $g \circ f : U \to W$ *is an accepted evaluation-graph packet. Its pointwise classifier is the target classifier of* W*, its additivity ledger is the composite of the additivity ledgers of* G_f *and* G_g*, and its scalar-compatibility ledger is the composite of their scalar ledgers.*

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapEvaluationGraph composition closure`

Proof. The matching middle row identifies the target classifier and operations of the first packet with the source classifier and operations of the second packet. Thus every carried $u : U$ evaluated by G_f produces a carried middle endpoint in V, and the carrier-admission row of G_g evaluates that endpoint in W. Splicing the two Cont routes gives the graph row for $g(f(u))$.

For additivity, the first packet classifies $f(u +_U u')$ with $f(u) +_V f(u')$. The second packet transports that middle classifier comparison through its pointwise descent row and then applies its

own additivity ledger to classify $g(f(u) +_V f(u'))$ with $g(f(u)) +_W g(f(u'))$. The target NameCert in W composes these comparisons into the displayed additivity ledger for $g \circ f$.

The scalar row is parallel. The first packet classifies $f(a \odot_U u)$ with $a \odot_V f(u)$. The second packet transports along the shared middle classifier and applies scalar compatibility in the target vector space, yielding a comparison with $a \odot_W g(f(u))$.

All pointwise comparisons are recorded as target-classifier rows in W, so Definition 48.49 supplies the composite $\mathsf{LinMap}^{\uparrow}(U, W)$ classifier. The Pkg provenance is the ordered pair of input packet identifiers together with the spliced continuation address, and the exactness clause follows because no row outside the two accepted graph packets and their shared middle vector-space row is read. □

Theorem 48.84 (LinearMap evaluation-graph composition proof interface)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, and let U, V, and W carry vector-space rows over the same scalar source. Suppose that G_f is an accepted evaluation-graph packet for $f : U \to V$, that G_g is an accepted evaluation-graph packet for $g : V \to W$, that the target vector-space row of G_f is* hsame *as the source vector-space row of G_g, that target-classifier transport is available in W, and that the displayed* Cont *provenance row splices the two graph reads through the shared middle row. Then these finite data determine exactly the nine rows of Definition 48.81 for $g \circ f$: the source vector-space row, the target vector-space row, the evaluation graph row, the additivity row, the scalar row, the pointwise target classifier row, the continuation row, the* Pkg *provenance row, and the naming row.*

Consequently the composite packet satisfies the local NameCert obligation surface of Definition 48.82.

Proof. The two accepted packets are finite BHist packets, so their source vector-space rows, target vector-space rows, additivity rows, scalar rows, target classifier rows, Cont rows, Pkg rows, and naming rows are already present as carried history rows. The hsame middle hypothesis identifies the target vector-space row of G_f with the source vector-space row of G_g, so the composite source row is read from G_f and the composite target row is read from G_g.

For the evaluation graph row, follow the Cont provenance from a carried source endpoint through G_f to the middle vector-space row and then through G_g to the target row of W. The target-classifier transport in W records the resulting pointwise classifier row for $g \circ f$.

For the additivity row, read the additivity row of G_f in the middle vector space, transport it across the shared target classifier row, and then read the additivity row of G_g in W. The scalar row is obtained by the same route, using the scalar rows of the two packets and the same target classifier row in W.

The Pkg row is the paired provenance of the two input packets together with the spliced Cont address, and the naming row records that this finite packet is the graph packet for $g \circ f$. These are exactly the nine rows specified by Definition 48.81, and Definition 48.83 supplies the accepted composite packet. Applying Definition 48.82 to those rows gives the stated local NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.LinearMapUp.LinearMapEvaluationGraphCompositionProofInterface`

Chapter 49

A Concrete Naming Certificate for $\mathsf{Mat}^{\uparrow}$

The matrix interface $\mathsf{Mat}^{\uparrow}$ has certificate obligations over a scalar ring from chapter 43 and two natural-number dimensions from chapter 28. The carrier is functions $\mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R$; the classifier is pointwise scalar sameness on all index pairs. Matrix addition and multiplication are derived operations governed by the ring's certificate fields. It is not a primitive matrix type; it is admitted only as a derived function carrier.

49.1 Carrier definition

Definition 49.1 (Matrix carrier)**.** *Use functions* $\mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R$ *with* $n, m : \mathsf{Nat}^{\uparrow}$ *and scalar ring* R*.*

Theorem 49.2 (Matrix singleton carrier visible exclusion)**.** *For every history* $p : \mathsf{BHist}$*, neither visible-headed history lies in the singleton matrix carrier:*

$$A_{\mathsf{e}}(\mathsf{E0}(p)) \to \bot \qquad \textit{and} \qquad A_{\mathsf{e}}(\mathsf{E1}(p)) \to \bot.$$

Proof. Unfold the singleton matrix carrier as history-sameness to e. The two displayed histories have visible BEDC constructors, and constructor separation rules out history-sameness between either visible endpoint and e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonCarrier visible absurd`

49.2 Concrete singleton empty-history matrix instance

Definition 49.3 (Concrete singleton empty-history matrix instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$*. Use the classifier*

$$\rho_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

The row-dimension witness and column-dimension witness are both represented by the empty unary history e*. The scalar side is the singleton empty-history ring of Definition 43.5, and the row and column vector layers are read through the singleton empty-history vector-space instance of Definition 47.2.*

The matrix carrier is the same singleton source A_{e} on BHist *codes. Define the displayed matrix operations by*

$$0_M := \mathsf{e}, \qquad I_M := \mathsf{e}, \qquad M \oplus N := \mathsf{append}(M, N), \qquad M \odot N := \mathsf{append}(M, N).$$

The certificate fit has ten fields:

1. *Traditional target: the terminal singleton matrix algebra.*
2. *BEDC source:* BHist *together with the singleton ring and* $\mathsf{VecSpace}^{\uparrow}$ *layers.*
3. *Carrier choice:* A_{e}.
4. *Classifier choice:* ρ_{e}.
5. *Stability obligations: closure, zero laws, identity laws, associativity, and congruence for the displayed operations.*
6. *Ledger: the singleton endpoint together with row and column witnesses represented by* e.
7. *Core theorem list: Theorem 49.48.*
8. *Exactness predicate: singleton endpoint classification.*
9. *Standard bridge: the terminal matrix object.*
10. *Status:* ProofTarget.

Definition 49.4 (Matrix singleton power)**.** *For histories M and e,* $\mathsf{MatrixSingletonPow}(M, e)$ *is the concrete power ledger obtained by recursion over the exponent history: the empty exponent returns I_M, a one-headed exponent multiplies the previous ledger by M, and the zero-headed branch returns the singleton zero endpoint.*

Lean4 (definition) `BEDC.Derived.MatrixUp.MatrixSingletonPow`

Theorem 49.5 (Matrix singleton power empty-exponent classifier iff)**.** *For histories M, h :* BHist*, the singleton classifier of the empty exponent power ledger against h is exactly the singleton carrier condition on h:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, \mathsf{e}), h) \iff \mathsf{MatrixSingletonCarrier}(h).$$

Proof. The empty exponent branch of MatrixSingletonPow is $\mathsf{MatrixSingletonOne} = \mathsf{e}$. The forward implication reads the right carrier field from the classifier. Conversely, a singleton carrier witness for h combines with the empty carrier of e and symmetry of hsame to assemble the singleton classifier. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow empty exponent classifier iff`

Theorem 49.6 (Matrix singleton power zero-head exponent classifier iff)**.** *For histories M, e, h :* BHist*, the singleton classifier of a zero-headed exponent power ledger against h is exactly the singleton carrier condition on h:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, \mathsf{E0}(e)), h) \iff \mathsf{MatrixSingletonCarrier}(h).$$

Proof. The zero-headed exponent branch of MatrixSingletonPow is $\mathsf{MatrixSingletonZero} = \mathsf{e}$. The classifier therefore has the same right-endpoint readback as the empty singleton endpoint: forward projection gives the carrier of h, and the reverse direction repacks the empty carrier of e, the carrier of h, and the corresponding hsame comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow zero head exponent classifier iff`

Theorem 49.7 (Matrix singleton power carrier closure)**.** *If M is carried by the singleton matrix carrier and e is unary, then the power ledger is again carried by the singleton matrix carrier:*

$$A_{\mathsf{e}}(M) \wedge \mathsf{Unary}(e) \quad \Longrightarrow \quad A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, e)).$$

Proof. Proceed by induction on the unary exponent. The empty exponent returns $I_M = \mathsf{e}$, hence is carried by A_{e}. In the one-headed case, the induction hypothesis carries the previous power ledger, while the hypothesis carries M; the displayed multiplication is append, so append empty-result exactness carries the product ledger. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow carrier closed`

Theorem 49.8 (Matrix singleton power nonempty unary input carrier exactness)**.** *If the exponent e is unary and not history-same as* e*, then the singleton carrier condition on the power ledger is exactly the singleton carrier condition on the input matrix:*

$$\mathsf{Unary}(e) \wedge (\mathsf{hsame}(e, \mathsf{e}) \to \bot) \quad \Longrightarrow \quad (A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, e)) \Longleftrightarrow A_{\mathsf{e}}(M)).$$

Proof. The unary, nonempty exponent has a one-headed form. In that branch the power ledger is the displayed append product of the previous power ledger and M. Append empty-result exactness therefore recovers $A_{\mathsf{e}}(M)$ from the carried power ledger. Conversely, carrier closure for singleton powers carries the power ledger from $A_{\mathsf{e}}(M)$ and $\mathsf{Unary}(e)$. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow carrier nonempty unary input iff`

Theorem 49.9 (Matrix singleton positive power continuation result carrier iff)**.** *Let M, e, y, r :* BHist*. If e is unary and not history-same to* e*, and if* $\mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, e), y, r)$*, then the displayed continuation result is carried exactly when both the base matrix and the right continuation input are carried:*

$$A_{\mathsf{e}}(r) \Longleftrightarrow A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(y).$$

Proof. For the forward direction, transport the continuation result to e and invert empty continuation results. This gives carrier witnesses for the power source and for y; positive-exponent power carrier exactness recovers $A_{\mathsf{e}}(M)$ from the power source carrier. Conversely, $A_{\mathsf{e}}(M)$ carries the positive power source, and together with $A_{\mathsf{e}}(y)$ append empty-result exactness identifies the displayed continuation result with e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent continuation result carrier iff`

Theorem 49.10 (Matrix singleton power positive exponent classifier iff)**.** *Let M, e, h :* BHist*. If e is unary and not history-same to* e*, then the singleton classifier of the positive power ledger against h is exactly the pair of singleton carrier conditions on the base and the displayed endpoint:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, e), h) \Longleftrightarrow \mathsf{MatrixSingletonCarrier}(M) \wedge \mathsf{MatrixSingletonCarrier}(h).$$

Proof. The forward direction reads the carried power endpoint from the classifier and uses positive-exponent carrier exactness to recover the singleton carrier of M; the right endpoint carrier is already part of the classifier. Conversely, the singleton carrier of M and unary exponent closure carry the positive power ledger, and the carried endpoint h completes the singleton classifier by transitivity through e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent classifier iff`

Theorem 49.11 (Matrix singleton power nonempty unary suffix base carrier)**.** *If tail is unary and not history-same as* e*, then a singleton carrier witness for the power ledger over* $\mathsf{append}(e, tail)$ *recovers the base matrix carrier:*

$$\mathsf{Unary}(tail) \wedge (\mathsf{hsame}(tail, \mathsf{e}) \to \bot) \wedge A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(e, tail))) \quad \Longrightarrow \quad A_{\mathsf{e}}(M).$$

Proof. The unary, nonempty suffix cannot be e and cannot start with $\mathsf{E0}$. Thus it has the form $\mathsf{E1}(t)$, so the appended exponent also exposes a one-headed power branch. In that branch the power ledger is the append product of the previous power ledger and M, and append empty-result inversion projects the right factor carrier $A_{\mathsf{e}}(M)$. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow nonempty unary suffix base carrier`

Theorem 49.12 (Matrix singleton positive-exponent continuation readback)**.** *If the exponent* e *is unary and not history-same as* e*, then its one-headed tail exposes the concrete successor continuation for the power ledger:*

$$\mathsf{Unary}(e) \wedge (\mathsf{hsame}(e, \mathsf{e}) \to \bot) \quad \Longrightarrow \quad \exists t.\ \mathsf{Unary}(t) \wedge \mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, t), M, \mathsf{MatrixSingletonPow}(M, e)).$$

Proof. Unary nonempty histories have a one-headed tail. In that branch the recursive definition of $\mathsf{MatrixSingletonPow}$ is the append product of the predecessor power ledger with M, so the continuation is the displayed append step. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent cont readback`

Theorem 49.13 (Matrix singleton unary exponent cases)**.** *For a unary exponent, the singleton power ledger has exactly the BEDC empty-or-successor shape needed for power readback: the empty exponent classifies the power ledger against* I_M*, while a one-headed exponent exposes a unary tail and the displayed continuation by the base matrix:*

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, e), I_M) \quad \vee \quad \exists t.\ \mathsf{Unary}(t) \wedge \mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, t), M, \mathsf{MatrixSingletonPow}(M, e)$$

Proof. Inspect the exponent history. The empty case unfolds to $I_M = \mathsf{e}$ and closes by singleton classifier reflexivity. The zero-headed case is impossible for a unary exponent. The one-headed case returns its tail and the defining append continuation of $\mathsf{MatrixSingletonPow}$. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow unary exponent cases`

Theorem 49.14 (Matrix singleton unary exponent carrier or base carrier)**.** *If a unary-exponent power ledger is singleton-carried, then either the exponent is the empty history or the base matrix itself is singleton-carried:*

$$\mathsf{Unary}(e) \wedge A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, e)) \Longrightarrow \mathsf{hsame}(e, \mathsf{e}) \vee A_{\mathsf{e}}(M).$$

Proof. Inspect the unary exponent. The empty branch supplies $\mathsf{hsame}(e, \mathsf{e})$. The zero-headed branch contradicts unary membership. In the one-headed branch the power ledger unfolds as an append product whose right factor is M; append empty-result inversion projects the singleton carrier of M. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow unary exponent carrier or base carrier`

Theorem 49.15 (Matrix singleton positive-exponent empty continuation base carrier)**.** *If e is unary and not history-same as* e*, and a continuation from* $\mathsf{MatrixSingletonPow}(M, e)$ *has result history-same to* e*, then the base matrix M lies in the singleton matrix carrier:*

$$A_{\mathsf{e}}(M).$$

Proof. Transport the continuation result to the concrete empty endpoint and invert the empty-result continuation. This carries the power ledger itself by A_{e}. The nonempty unary power carrier exactness theorem then reads the carriedness of the base matrix M back from the carried power ledger. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent empty continuation base carrier`

Theorem 49.16 (Matrix singleton positive-tail empty continuation factors)**.** *Suppose depth is unary and depth* $= \mathsf{E1}(tail)$*. If a continuation from* $\mathsf{MatrixSingletonPow}(M, depth)$ *has result history-same as* e*, then the predecessor power, the base matrix, and the continuation argument are all singleton-carried:*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, depth), y, r) \wedge r\mathsf{hsamee} \Longrightarrow A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, tail)) \wedge A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(y).$$

Proof. Transport the continuation result to e and invert the empty-result continuation. The source is the one-headed power branch, hence the append of $\mathsf{MatrixSingletonPow}(M, tail)$ with M. Empty append inversion gives the predecessor carrier and the base carrier, while continuation inversion also gives the carriedness of y. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive tail empty continuation factors`

Theorem 49.17 (Matrix singleton power append positive suffix continuation readback)**.** *If $suffix$ is unary and not history-same as* e*, then the power ledger over* $\mathsf{append}(pref, suffix)$ *exposes the successor continuation at the same prefix:*

$$\exists tail.\ \mathsf{Unary}(tail) \wedge \mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(pref, tail)), M, \mathsf{MatrixSingletonPow}(M, \mathsf{append}(pref, suffi$$

Proof. Unary nonempty suffixes have a one-headed tail. After substituting $suffix = \mathsf{E1}(tail)$, the appended exponent is itself one-headed with payload $\mathsf{append}(pref, tail)$. The recursive power ledger therefore unfolds to the displayed continuation by M. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append positive suffix cont readback`

Theorem 49.18 (Matrix singleton power append positive suffix classifier readback)**.** *If M is carried by the singleton matrix carrier, $pref$ and $suffix$ are unary, and $suffix$ is not history-same as* e*, then the append-positive suffix readback supplies a unary tail whose successor multiplication classifies the original appended power ledger:*

$$\exists tail.\ \mathsf{Unary}(tail) \wedge \rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(pref, suffix)), \mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonPow}(M, \mathsf{a}$$

Proof. The positive-suffix readback exposes a unary tail and the continuation from $\mathsf{MatrixSingletonPow}(M, \mathsf{append}($ by M to the original appended power ledger. Carrier closure gives the source power carrier, and append empty-result exactness carries the displayed multiplication endpoint. The continuation equation gives the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append positive suffix classifier readback`

Theorem 49.19 (Matrix singleton power append positive suffix classifier exact carrier criterion)**.** *If $pref$ and $suffix$ are unary histories and $suffix$ is not history-same as* e*, then the singleton classifier of the append-positive power ledger against h is exactly the pair of singleton carrier facts for M and h:*

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(pref, suffix)), h) \iff A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(h).$$

Proof. The forward direction reads the carriedness of h from the classifier and applies nonempty unary suffix base-carrier readback to the carried power ledger. Conversely, unary append closure carries the appended power from $A_{\mathsf{e}}(M)$, and the two singleton carrier witnesses compare the endpoints through e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append positive suffix classifier iff`

Theorem 49.20 (Matrix singleton power successor classifier)**.** *For histories M, e :* BHist*, if M is carried by the singleton matrix carrier and e is unary, then the one-headed power ledger is classified against the displayed successor multiplication:*

$$A_{\mathsf{e}}(M) \wedge \mathsf{Unary}(e) \quad \Longrightarrow \quad \rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e)), \mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonPow}(M, e), M)).$$

Proof. The one-headed power branch unfolds to the displayed multiplication endpoint. The carrier closure theorem carries the predecessor power ledger, and the input hypothesis carries M. Since multiplication is append, append empty-result exactness carries the displayed multiplication endpoint, and reflexivity of hsame supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow succ classifier`

Theorem 49.21 (Matrix singleton power successor continuation classifier)**.** *For histories M, e, r :* BHist*, if M is carried by the singleton matrix carrier, e is unary, and*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, e), M, r),$$

then the one-headed power ledger is classified against r:

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e)), r).$$

Proof. Carrier closure carries the predecessor power ledger. Together with the carrier of M, append empty-result exactness carries the one-headed power endpoint. The displayed continuation identifies r with the same append endpoint, so r is carried by the singleton matrix carrier and is hsame to the one-headed power ledger. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow succ continuation classifier`

Theorem 49.22 (Matrix singleton power visible-base successor continuation empty-result absurd)**.** *For any singleton-carried continuation right input y, a successor power whose base has a visible head cannot continue to the empty result:*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), \mathsf{E1}(e)), y, r) \wedge r\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(m)$ *in place of* $\mathsf{E0}(m)$*.*

Proof. Transport the displayed continuation along $r\mathsf{hsamee}$ and invert the empty continuation result. The source side is the successor power endpoint, which unfolds to an append whose right factor is the visible base. Append empty-result inversion would identify $\mathsf{E0}(m)$, respectively $\mathsf{E1}(m)$, with e, contradicting BEDC constructor separation. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow visible base succ continuation empty result absurd`

Theorem 49.23 (Matrix singleton power positive-exponent visible-base empty-result exclusion)**.** *For unary nonempty exponent e, a singleton matrix power whose base has a visible head cannot continue to an empty result:*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), e), y, r) \wedge r\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(m)$ in place of $\mathsf{E0}(m)$.

Proof. Transport the continuation along $r\mathsf{hsamee}$ and invert the empty result. The nonempty-unary power carrier exactness theorem then turns the empty power endpoint into a singleton carrier witness for the visible base, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent visible base continuation empty result absurd`

Theorem 49.24 (Matrix singleton power visible-base appended positive-suffix empty-result exclusion)**.** *If $pref$ and $suffix$ are unary histories and $suffix \not\mathsf{hsamee}$, then neither visible-base singleton power over $\mathsf{append}(pref, suffix)$ can continue to an empty result:*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), \mathsf{append}(pref, suffix)), y, r) \wedge r\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(m)$ in place of $\mathsf{E0}(m)$.

Proof. Unary append closure makes $\mathsf{append}(pref, suffix)$ a unary exponent. If that exponent were empty, append empty-result exactness would force $suffix\mathsf{hsamee}$, contrary to the positive-suffix hypothesis. The positive-exponent visible-base carrier exclusion therefore rules out the source carrier recovered by transporting the continuation result to e and inverting the empty continuation. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow visible base append positive suffix continuation empty result absurd`

Theorem 49.25 (Matrix singleton power nonempty-base empty-result exclusion)**.** *For unary nonempty exponent e, if the base matrix M is not singleton-carried, then*

$$\mathsf{Cont}(\mathsf{MatrixSingletonPow}(M, e), y, r) \wedge r\mathsf{hsamee} \Longrightarrow \bot.$$

Proof. Transport the continuation result to e and invert the empty continuation. The source carrier for $\mathsf{MatrixSingletonPow}(M, e)$ transfers through positive-unary power exactness to a singleton carrier witness for M, contradicting the base non-carrier hypothesis. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow nonempty base continuation result absurd`

Theorem 49.26 (Matrix singleton power append exponent classifier)**.** *If M is carried by the singleton matrix carrier and w, q are unary histories, then the power ledger over the appended exponent is classified with the product of the two factor power ledgers:*

$$A_{\mathsf{e}}(M) \wedge \mathsf{Unary}(w) \wedge \mathsf{Unary}(q) \quad \Longrightarrow \quad \rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(w, q)), \mathsf{MatrixSingletonMul}(\mathsf{MatrixSingleton}$$

Proof. Append closure makes $\mathsf{append}(w, q)$ unary, so power carrier closure carries the left ledger. The same closure carries each factor ledger; append empty-result exactness then carries their displayed product. Since both endpoints are carried by A_{e}, transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append exponent classifier`

Theorem 49.27 (Matrix singleton power append successor right exponent classifier)**.** *If M is carried by the singleton matrix carrier and w, q are unary histories, then the append exponent with right successor is classified by the left power and the nested right multiplication endpoint:*

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(w, \mathsf{E1}(q))), \mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonPow}(M, w), \mathsf{MatrixSingletonMul}(\mathsf{M}$$

Proof. Unary closure carries $\mathsf{append}(w, \mathsf{E1}(q))$. Singleton power closure carries the composite power, the left power, and the right predecessor power. Append empty-result exactness carries the inner multiplication by M and then the outer multiplication by the left power. Since both endpoints are carried by A_{e}, transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append succ right exponent classifier`

Theorem 49.28 (Matrix singleton power append exponent commuted classifier)**.** *If M is carried by the singleton matrix carrier and w, q are unary histories, then the two power ledgers obtained by the two append orders of the exponent histories are singleton-classified:*

$$A_{\mathsf{e}}(M) \wedge \mathsf{Unary}(w) \wedge \mathsf{Unary}(q) \quad \Longrightarrow \quad \rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(w, q)), \mathsf{MatrixSingletonPow}(M, \mathsf{append}(q, w$$

Proof. Unary append closure makes both $\mathsf{append}(w, q)$ and $\mathsf{append}(q, w)$ unary. Singleton power carrier closure carries both power ledgers from $A_{\mathsf{e}}(M)$. Since both endpoints are carried by A_{e}, transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append exponent comm classifier`

Theorem 49.29 (Matrix singleton power append exponent classifier iff)**.** *If M is carried by the singleton matrix carrier and a, b are unary histories, then the classifier of the append-exponent power ledger against h is exactly the singleton carrier condition on h:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, \mathsf{append}(a, b)), h) \Longleftrightarrow \mathsf{MatrixSingletonCarrier}(h).$$

Proof. Unary closure under append makes $\mathsf{append}(a, b)$ a unary exponent. The power carrier theorem therefore carries $\mathsf{MatrixSingletonPow}(M, \mathsf{append}(a, b))$. The forward direction reads the right carrier field from the classifier. In the reverse direction, that carried power endpoint and the carrier witness for h assemble the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow append exponent classifier iff`

Theorem 49.30 (Matrix singleton power successor endpoint exactness)**.** *For histories* M, e : BHist*, if* $A_{\mathsf{e}}(M)$ *and* $\mathsf{Unary}(e)$*, then the successor power is carried:*

$$A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e))).$$

Moreover, the successor power is exactly classified with both displayed singleton multiplication endpoints:

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e)), \mathsf{MatrixSingletonPow}(M, e) \odot M) \quad \textit{and} \quad \rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e)), M \odot \mathsf{Ma}$$

Proof. Unfold the one-headed exponent clause of $\mathsf{MatrixSingletonPow}$. The successor ledger is the displayed singleton product $\mathsf{MatrixSingletonPow}(M, e) \odot M$.

The carrier closure theorem for powers gives $A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(M, e))$ from $A_{\mathsf{e}}(M)$ and $\mathsf{Unary}(e)$. Combining this carrier with $A_{\mathsf{e}}(M)$, the singleton multiplication closure in Theorem 49.48 carries both append endpoints $\mathsf{MatrixSingletonPow}(M, e) \odot M$ and $M \odot \mathsf{MatrixSingletonPow}(M, e)$.

For the first classifier, the unfolded successor ledger is the same append endpoint, so the singleton classifier closes by reflexivity and the carried endpoint witness. For the second classifier, both append endpoints classify as empty singleton endpoints; transitivity through e compares them and supplies ρ_{e}. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow succ endpoint exactness`
Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow successor endpoint exactness`

Definition 49.31 (Matrix singleton identity-power concrete instance)**.** *The concrete identity-power schema for* $\mathsf{Mat}^{\uparrow}$ *uses* $\mathsf{MatrixSingletonCarrier}$ *as carrier,* $\mathsf{MatrixSingletonClassifier}$ *as classifier,* $\mathsf{UnaryHistory}$ *as the exponent carrier, and* $\mathsf{MatrixSingletonPow}$ *as the action. Its source rows are the singleton matrix carrier row for the base, the unary history row for the exponent, the empty-exponent identity row, and the successor-step multiplication row over the singleton power ledger.*

Theorem 49.32 (Matrix singleton identity-power law)**.** *For the concrete instance of Definition 49.31, the empty exponent acts as the singleton identity endpoint, unary exponents preserve the singleton matrix carrier, and the successor exponent is classified by the displayed multiplication step:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, \mathsf{e}), h) \iff \mathsf{MatrixSingletonCar}$$
$$\mathsf{MatrixSingletonCarrier}(M) \wedge \mathsf{UnaryHistory}(e) \implies \mathsf{MatrixSingletonCarrier}(\mathsf{MatrixSingleto}$$
$$\mathsf{MatrixSingletonCarrier}(M) \wedge \mathsf{UnaryHistory}(e) \implies \mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonPow}(M, \mathsf{E1}(e)), \mathsf{MatrixSin}$$

Thus the singleton power rows expose a concrete BHist *carrier and action surface for* $\mathsf{Mat}^{\uparrow}$ *rather than only an abstract operator schema.*

Proof. The empty-exponent identity row is exactly Definition 49.5. It unfolds the empty branch of $\mathsf{MatrixSingletonPow}$ to the singleton identity endpoint and identifies its classifier with the carrier condition on the comparison endpoint.

Carrier closure under unary exponents is Definition 49.7. It gives the carried power ledger from a singleton-carried base and a unary exponent, so the action stays inside $\mathsf{MatrixSingletonCarrier}$.

The successor step is Definition 49.20, strengthened by Definition 49.30. These rows classify the one-headed exponent ledger by the displayed singleton multiplication endpoint, completing the concrete carrier/action schema. □

Theorem 49.33 (Matrix singleton identity-power schema handoff)**.** *The data* MatrixSingletonCarrier*,* MatrixSingletonClassifier*,* UnaryHistory*, and* MatrixSingletonPow *provide the concrete* $\mathsf{Mat}^{\uparrow}$ *power schema handoff consumed by algebra and control chapters. The handoff exports the base carrier, exponent carrier, identity endpoint, unary power closure, successor endpoint, classifier,* hsame *transport,* Cont*, and* Pkg *rows as finite* BHist *rows.*

Proof. The instance data are exactly the four fields named in Definition 49.31. The law in Definition 49.32 identifies the empty exponent with the singleton identity endpoint, proves unary carrier closure, and records the successor action through the displayed multiplication row.

The carrier component is closed by Definition 49.7. The successor endpoint is exact by Definition 49.30. Therefore each exported row is a BHist, hsame, Cont, or Pkg row already present in the concrete singleton packet, so downstream algebra and control consumers receive a finite carrier/action handoff rather than abstract matrix-power parameters. □

Theorem 49.34 (Matrix singleton classifier append split empty)**.** *For* $M, N, h : \mathsf{BHist}$*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(M, N), h) \iff M\mathsf{hsamee} \wedge N\mathsf{hsamee} \wedge A_{\mathsf{e}}(h).$$

Proof. The forward direction reads the left carrier component of $\rho_{\mathsf{e}}(\mathsf{append}(M, N), h)$ as $\mathsf{append}(M, N)\mathsf{hsamee}$. Empty-result inversion for append splits this endpoint witness into $M\mathsf{hsamee}$ and $N\mathsf{hsamee}$, while the right carrier component supplies $A_{\mathsf{e}}(h)$.

Conversely, $M\mathsf{hsamee}$ and $N\mathsf{hsamee}$ reconstruct the singleton carrier witness for $\mathsf{append}(M, N)$. Together with $A_{\mathsf{e}}(h)$, transitivity of hsame through e gives the classifier comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append split empty iff`

Theorem 49.35 (Matrix singleton add-mul classifier iff)**.** *For* $M, N : \mathsf{BHist}$*, the singleton matrix classifier relates the displayed addition and multiplication results exactly when both inputs are singleton empty endpoints:*

$$\rho_{\mathsf{e}}(M \oplus N, M \odot N) \iff M\mathsf{hsamee} \wedge N\mathsf{hsamee}.$$

Proof. Both displayed operations are the same BEDC append spine $\mathsf{append}(M, N)$. In the forward direction, the left carrier component of $\rho_{\mathsf{e}}(M \oplus N, M \odot N)$ says that this append spine is hsame to e. Empty-result inversion for append splits it into the two endpoint witnesses $M\mathsf{hsamee}$ and $N\mathsf{hsamee}$.

Conversely, the two endpoint witnesses reconstruct the singleton carrier for $\mathsf{append}(M, N)$. Since $M \oplus N$ and $M \odot N$ unfold to that same append spine, reflexivity of hsame supplies the classifier relation between the displayed operation results. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddMul classifier iff`

Theorem 49.36 (Matrix singleton add-mul continuation result iff)**.** *For singleton-carrier inputs* M *and* N*, a continuation from the displayed addition result to the displayed multiplication result has result* R *exactly when* R *is itself a singleton carrier:*

$$\mathsf{Cont}(M \oplus N, M \odot N, R) \iff A_{\mathsf{e}}(R).$$

Proof. The two displayed operations both unfold to the same append spine $\mathsf{append}(M, N)$. The singleton witnesses for M and N collapse that spine to e. The forward direction compares the displayed continuation with the right-unit continuation at e; the reverse direction transports the result carrier back to that same empty continuation. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddMul continuation result iff`

Theorem 49.37 (Matrix singleton add-mul continuation empty-result factors)**.** *For histories* $M, N, R : \mathsf{BHist}$*, if* $\mathsf{Cont}(M \oplus N, M \odot N, R)$*, then the displayed continuation result is empty exactly when both matrix inputs are empty:*

$$R\mathsf{hsamee} \iff M\mathsf{hsamee} \wedge N\mathsf{hsamee}.$$

Proof. For the forward direction, transport the displayed continuation along $R\mathsf{hsamee}$ and invert the empty continuation result. The source factor is $M \oplus N$, which unfolds to $\mathsf{append}(M, N)$, so append empty-result inversion gives $M\mathsf{hsamee}$ and $N\mathsf{hsamee}$.

Conversely, the two input endpoint witnesses collapse both displayed operations to the empty history. Continuation determinacy compares the given continuation with the BEDC right-unit continuation at e, identifying the result with e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddMul continuation empty result factors iff`

Theorem 49.38 (Matrix singleton multiplication continuation result carrier)**.** *For histories* $M, N, h, r : \mathsf{BHist}$*, any continuation from* h *through the singleton multiplication endpoint has singleton result exactly when the source and both multiplication inputs are singleton carriers:*

$$\mathsf{Cont}(h, M \odot N, r) \Longrightarrow (A_{\mathsf{e}}(r) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(N)).$$

Proof. In the forward direction, transport the displayed continuation result along $r\mathsf{hsamee}$ and invert the empty continuation result. The target endpoint is $M \odot N$, which unfolds to $\mathsf{append}(M, N)$, so append empty-result inversion supplies the two multiplication input carriers. The same continuation inversion supplies the source carrier.

Conversely, the source carrier and the two multiplication input carriers collapse the target endpoint to e. Continuation respect for hsame compares the displayed continuation with the right-unit continuation at e, giving the singleton result carrier. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonMulContinuation result carrier iff`

Definition 49.39 (Matrix singleton entry projection package)**.** *The matrix singleton entry projection package fixes the matrix carrier to* A_{e}*, fixes the matrix classifier to* ρ_{e}*, and reads the displayed row and column dimensions from the* $\mathsf{Nat}^{\uparrow}$ *rows already carried by the singleton matrix instance. Its entry projection operation sends every carried unary row to the* BHist *endpoint* e*.*

Theorem 49.40 (Matrix singleton entry projection laws)**.** *In the matrix singleton entry projection package, every projected entry of a carried singleton matrix is carried by* A_{e}*. Projected entries are stable under* ρ_{e} *and under the displayed* hsame *ledger. Moreover, the projected entries of the displayed singleton addition result and of the displayed singleton multiplication result are classified with* e*.*

Proof. Unfold the package. The carrier is A_{e}, the classifier is ρ_{e}, and every entry projection is the constant endpoint e on the carried unary rows. Hence each projected entry is immediately an A_{e} entry and is related to the singleton endpoint by ρ_{e}.

For stability, use the definition of ρ_{e} together with the displayed hsame ledger. Transporting a carried singleton matrix along ρ_{e} keeps the projected endpoint equal to e, so the projected entry remains in the same singleton classifier row.

For the displayed addition and multiplication results, apply Definition 49.36, Definition 49.37, and Definition 49.38. The local continuation laws identify the result endpoints with A_{e}, and the constant entry projection classifies their projected entries with e. No function-space carrier or Fin-indexed matrix carrier is introduced. □

Theorem 49.41 (Matrix singleton add-mul visible-left continuation result nonempty)**.** *For any histories m, N, R, if the shared left input of the displayed singleton addition and multiplication has a visible head, then a continuation from the addition endpoint to the multiplication endpoint cannot have empty result:*

$$\mathsf{Cont}(\mathsf{E0}(m) \oplus N, \mathsf{E0}(m) \odot N, R) \wedge R\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(m)$ in place of $\mathsf{E0}(m)$.

Proof. Apply the continuation empty-result factor theorem for displayed singleton addition and multiplication. An empty result forces the left input factor to be empty. This contradicts constructor separation for $\mathsf{E0}(m)$ in the first case and for $\mathsf{E1}(m)$ in the second case. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddMul visible left continuation result nonempty`

Theorem 49.42 (Matrix singleton classifier visible right append exclusion)**.** *For histories h, p, q : BHist, a singleton matrix classifier witness cannot target an append whose right factor has a visible head:*

$$\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e0}(q))) \to \bot \qquad \textit{and} \qquad \rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{e1}(q))) \to \bot.$$

Proof. Unfold the singleton matrix classifier. A classifier witness supplies the target carrier endpoint $\mathsf{append}(p, \mathsf{e0}(q))\mathsf{hsamee}$ in the first case, or $\mathsf{append}(p, \mathsf{e1}(q))\mathsf{hsamee}$ in the second case. Empty-result inversion for append forces the visible right factor itself to be e, contradicting the disjointness of each visible constructor from e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append visible right absurd`

Theorem 49.43 (Matrix singleton classifier append pair carrier)**.** *For M, N, P, Q : BHist,*

$$\rho_{\mathsf{e}}(\mathsf{append}(M, N), \mathsf{append}(P, Q)) \quad \Longleftrightarrow \quad A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(N) \wedge A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q).$$

Proof. The classifier exposes singleton carrier witnesses for both append endpoints. Empty-result inversion for append splits these witnesses into the four displayed endpoint carriers. Conversely, the four endpoint carriers rebuild the two empty append endpoints, and transitivity of hsame through e supplies the classifier comparison between them. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append pair carrier iff`

Theorem 49.44 (Matrix singleton classifier self-context split)**.** *For histories a, x, b : BHist,*

$$\rho_{\mathsf{e}}(\mathsf{append}(a, \mathsf{append}(x, a)), b) \Longleftrightarrow A_{\mathsf{e}}(a) \wedge \rho_{\mathsf{e}}(x, b).$$

Proof. The forward implication splits the source carrier of the classifier twice by empty-result inversion for append. This gives the carrier witness for the repeated context a and the carrier witness for the middle history x; the target carrier field and transitivity through e rebuild $\rho_{\mathsf{e}}(x, b)$.

Conversely, the carrier for a and the source carrier inside $\rho_{\mathsf{e}}(x, b)$ rebuild the inner and outer empty append endpoints. The target carrier is inherited from $\rho_{\mathsf{e}}(x, b)$, and both endpoints compare through the empty history. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append self context split iff`

Theorem 49.45 (Matrix singleton classifier append-left cancellation)**.** *For histories* P, Q, R : BHist*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(R, P), \mathsf{append}(R, Q)) \iff A_{\mathsf{e}}(R) \wedge \rho_{\mathsf{e}}(P, Q).$$

Proof. In the forward direction, the two carrier fields say that $\mathsf{append}(R, P)$ and $\mathsf{append}(R, Q)$ are empty. Empty-result inversion for append splits these witnesses into $R\mathsf{hsamee}$, $P\mathsf{hsamee}$, and $Q\mathsf{hsamee}$. The shared left endpoint gives $A_{\mathsf{e}}(R)$, and the two remaining endpoint witnesses rebuild $\rho_{\mathsf{e}}(P, Q)$.

Conversely, unpack $A_{\mathsf{e}}(R)$ and $\rho_{\mathsf{e}}(P, Q)$. The shared left endpoint and each singleton classified endpoint rebuild both empty append endpoints. Transitivity of hsame through e then supplies the classifier comparison between the two appended histories. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append left cancel iff`

Theorem 49.46 (Matrix singleton classifier append-right cancellation)**.** *For histories* P, Q, R : BHist*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(P, R), \mathsf{append}(Q, R)) \iff \rho_{\mathsf{e}}(P, Q) \wedge A_{\mathsf{e}}(R).$$

Proof. In the forward direction, the two carrier fields say that $\mathsf{append}(P, R)$ and $\mathsf{append}(Q, R)$ are empty. Empty-result inversion for append splits these witnesses into $P\mathsf{hsamee}$, $Q\mathsf{hsamee}$, and $R\mathsf{hsamee}$. The first two endpoint witnesses rebuild $\rho_{\mathsf{e}}(P, Q)$, and the shared right endpoint gives $A_{\mathsf{e}}(R)$.

Conversely, unpack $\rho_{\mathsf{e}}(P, Q)$ and $A_{\mathsf{e}}(R)$. All three displayed histories are singleton endpoints, so both appended histories reduce to e, and the singleton classifier follows by reflexivity of hsame. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append right cancel iff`

Theorem 49.47 (Matrix singleton classifier append common context)**.** *For histories* L, P, Q, R : BHist*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(L, \mathsf{append}(P, R)), \mathsf{append}(L, \mathsf{append}(Q, R))) \iff A_{\mathsf{e}}(L) \wedge \rho_{\mathsf{e}}(P, Q) \wedge A_{\mathsf{e}}(R).$$

Proof. In the forward direction, the two classifier carrier fields say that both common-context append endpoints are empty. Empty-result inversion first splits off the shared left context L, then splits the middle append endpoints. This yields $A_{\mathsf{e}}(L)$, $A_{\mathsf{e}}(P)$, $A_{\mathsf{e}}(Q)$, and $A_{\mathsf{e}}(R)$; the middle comparison $\rho_{\mathsf{e}}(P, Q)$ follows by transitivity through e.

Conversely, $A_{\mathsf{e}}(L)$, $\rho_{\mathsf{e}}(P, Q)$, and $A_{\mathsf{e}}(R)$ reconstruct both common-context append endpoints as empty. Transitivity through e then supplies the singleton classifier relation between the displayed endpoints. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append common context iff`

Theorem 49.48 (Singleton empty-history matrix laws)**.** *For the singleton instance of Definition 49.3, the displayed matrix carrier and operations satisfy the concrete* $\mathsf{Mat}^{\uparrow}$ *law package:*

1. 0_M *and* I_M *are carried by* A_{e}*.*
2. *If* $A_{\mathsf{e}}(M)$ *and* $A_{\mathsf{e}}(N)$*, then* $A_{\mathsf{e}}(M \oplus N)$ *and* $A_{\mathsf{e}}(M \odot N)$*.*
3. 0_M *is a left and right unit for* $\oplus$ *under* ρ_{e}*.*
4. I_M *is a left and right identity for* $\odot$ *under* ρ_{e}*.*

5. *$\oplus$ and $\odot$ are associative under ρ_{e}.*

6. *ρ_{e} is reflexive, symmetric, and transitive on carried matrix codes, and it is respected by $\oplus$ and $\odot$.*

Proof. The carrier endpoint is supplied by the singleton source witness from Definition 36.169 at $a_0 = \mathsf{e}$. The row and column witnesses use the same empty unary history, so the dimension ledger is closed at the singleton endpoint.

For operation closure, carried inputs are classified as singleton endpoints. The displayed operations are either e or append applied to singleton endpoints. The BEDC continuation unit laws cited in Theorem 40.3 identify the resulting append endpoint with e, so the singleton source again carries the result.

The zero laws for $\oplus$ and the identity laws for $\odot$ are the same unit calculation. On the left and on the right, the operation compares $\mathsf{append}(\mathsf{e}, M)$ or $\mathsf{append}(M, \mathsf{e})$ with M under ρ_{e}, and the continuation unit laws provide the hsame component. The endpoint components come from $A_{\mathsf{e}}(M)$ and $A_{\mathsf{e}}(\mathsf{e})$.

Associativity of $\oplus$ and $\odot$ uses the continuation associativity law already used by the singleton monoid, ring, and vector-space chapters. Both operation symbols are represented by append, so the two parenthesized products share a single BEDC append spine up to hsame.

For classifier congruence, unpack the two ρ_{e} hypotheses. Closure keeps both composite endpoints in A_{e}, and the relation components are combined by reflexivity, symmetry, and transitivity of hsame after the append unit and associativity reductions. This gives the required ρ_{e} comparison for both $\oplus$ and $\odot$. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonEmptyHistory laws`

Theorem 49.49 (Singleton empty-history matrix endpoint collapse)**.** *With the notation of Definition 49.3, if $A_{\mathsf{e}}(M)$ and $A_{\mathsf{e}}(N)$, then*

1. *$\rho_{\mathsf{e}}(M \oplus N, \mathsf{e})$.*

2. *$\rho_{\mathsf{e}}(M \odot N, \mathsf{e})$.*

3. *$\rho_{\mathsf{e}}(M \oplus N, M \odot N)$.*

Proof. Apply Theorem 49.48 to the carried inputs M and N. Its closure clause puts both $M \oplus N$ and $M \odot N$ back in A_{e}, while the singleton carrier witnesses identify M and N with the endpoint e.

By the displayed definitions, both operations are $\mathsf{append}(M, N)$. Replace the carried inputs by their singleton endpoint witnesses, then use the left and right continuation unit laws cited in Theorem 40.3 to reduce the resulting append history to e.

For $\rho_{\mathsf{e}}(M \oplus N, \mathsf{e})$ and $\rho_{\mathsf{e}}(M \odot N, \mathsf{e})$, combine the closure witnesses with the corresponding hsame endpoint calculation. Symmetry is used when the unit witness is oriented from e to the displayed operation rather than from the displayed operation to e.

For $\rho_{\mathsf{e}}(M \oplus N, M \odot N)$, compose the first endpoint comparison with the symmetry of the second by transitivity of hsame. The two carrier witnesses are the closure witnesses already supplied by the law package, so the classifier definition gives the required comparison. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonEmptyHistory endpoint collapse`

The $\mathsf{Mat}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and operations, so formalization can target the endpoint-collapse and law package rather than an abstract operator schema.

Theorem 49.50 (Singleton empty-history matrix endpoint exactness)**.** *For the singleton instance of Definition 49.3, if $A_{\mathsf{e}}(M)$ and $A_{\mathsf{e}}(N)$, then all displayed matrix endpoints are classified by the singleton empty history:*

1. $\rho_{\mathsf{e}}(M, \mathsf{e})$.
2. $\rho_{\mathsf{e}}(0_M, \mathsf{e})$.
3. $\rho_{\mathsf{e}}(I_M, \mathsf{e})$.
4. $\rho_{\mathsf{e}}(M \oplus N, \mathsf{e})$.
5. $\rho_{\mathsf{e}}(M \odot N, \mathsf{e})$.

Proof. Unfold the singleton carrier A_{e} and the classifier ρ_{e}. The hypotheses say that M and N are carried by the singleton endpoint, so the first classification is obtained by pairing the two carrier witnesses with the corresponding singleton endpoint witness in hsame.

Apply Theorem 49.48 to obtain closure of 0_M, I_M, $M \oplus N$, and $M \odot N$ in A_{e}. These closure witnesses supply the carrier components required by the classifier for the four remaining endpoints.

For the hsame components, compose the singleton endpoint witnesses with symmetry and transitivity of hsame. Thus each displayed endpoint is identified with e, and repacking the carrier witnesses with the hsame witness gives the five asserted instances of ρ_{e}. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonEmptyHistory endpoint exactness`

Theorem 49.51 (Matrix singleton classifier continuation commutativity closure)**.** *Let M, N : BHist be carried by the singleton matrix carrier A_{e}. If $\mathsf{Cont}(M, N, L)$ and $\mathsf{Cont}(N, M, R)$, then both continuation results are again carried by A_{e}, and the singleton matrix classifier relates the two results:*

$$A_{\mathsf{e}}(L) \wedge A_{\mathsf{e}}(R) \wedge \rho_{\mathsf{e}}(L, R).$$

Proof. The two carrier hypotheses identify M and N with e. After these identifications, the two continuation premises identify both L and R with the BEDC append endpoint $\mathsf{append}(\mathsf{e}, \mathsf{e})$, which is e. The singleton carrier witnesses for L and R are therefore reflexive empty history witnesses, and the classifier is rebuilt from those two carrier components and $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier continuation comm closed`

Theorem 49.52 (Matrix singleton carrier continuation result iff)**.** *Let M, N, R : BHist be histories with M and N carried by the singleton matrix carrier A_{e}. Then continuation from M to N has result R exactly when R is again carried by A_{e}:*

$$\mathsf{Cont}(M, N, R) \iff A_{\mathsf{e}}(R).$$

Proof. Unfold the two endpoint carrier witnesses as $\mathsf{hsame}(M, \mathsf{e})$ and $\mathsf{hsame}(N, \mathsf{e})$. In the forward direction, after those identifications the continuation has the same result as the right-unit continuation at e, so R is singleton-carried. Conversely, if R is singleton-carried, all three displayed histories are identified with e, and the right-unit continuation at e supplies the required continuation. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonCarrier continuation result iff`

Theorem 49.53 (Matrix singleton carrier append-target continuation result iff)**.** *Let M, N, h, R : BHist be histories with M and N carried by the singleton matrix carrier A_{e}. If $\mathsf{Cont}(h, \mathsf{append}(M, N), R)$, then the result is singleton-carried exactly when the source is singleton-carried:*

$$A_{\mathsf{e}}(R) \iff A_{\mathsf{e}}(h).$$

Proof. For the forward direction, transport the continuation result to e and apply empty-result continuation inversion; the source endpoint must be e. Conversely, the carrier witnesses for M and N split the append target as e, and continuation respects the source and target identifications. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonCarrier continuation append target result iff`

Theorem 49.54 (Matrix singleton nested append target continuation result)**.** *For histories* $M, N, P, h, r : \mathsf{BHist}$*, if* $\mathsf{Cont}(h, \mathsf{append}(M, \mathsf{append}(N, P)), r)$*, then*

$$A_{\mathsf{e}}(r) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(M) \wedge A_{\mathsf{e}}(N) \wedge A_{\mathsf{e}}(P).$$

Proof. The forward direction transports the continuation result to e and applies empty-result inversion for continuation, then splits the nested append endpoint twice. Conversely, the four carrier witnesses rebuild the nested target as e, and continuation respects the two endpoint identifications. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonNestedAppendTarget continuation result iff`

Theorem 49.55 (Matrix singleton visible continuation result exclusion)**.** *Let* $M, N : \mathsf{BHist}$ *be carried by the singleton matrix carrier* A_{e}*. Then a continuation* $\mathsf{Cont}(M, N, -)$ *cannot have a visible* $\mathsf{e0}$*-headed or* $\mathsf{e1}$*-headed result:*

$$\mathsf{Cont}(M, N, \mathsf{e0}(r)) \to \bot \qquad \textit{and} \qquad \mathsf{Cont}(M, N, \mathsf{e1}(r)) \to \bot.$$

Proof. The carrier witnesses identify both inputs with e. With those identifications, any continuation result must be the BEDC append endpoint $\mathsf{append}(\mathsf{e}, \mathsf{e}) = \mathsf{e}$. A visible-headed result cannot be that empty endpoint, so both visible cases are impossible. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonCarrier continuation visible result absurd`

Theorem 49.56 (Matrix singleton power visible-base carrier exactness)**.** *For any history* m *and exponent* e*, a singleton power with visible base* $\mathsf{E0}(m)$ *or* $\mathsf{E1}(m)$ *is carried exactly in the empty-exponent or zero-head exponent cases:*

$$A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), e)) \iff e\mathsf{hsamee} \ \vee \ \exists t.\ e = \mathsf{E0}(t),$$
$$A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(\mathsf{E1}(m), e)) \iff e\mathsf{hsamee} \ \vee \ \exists t.\ e = \mathsf{E0}(t).$$

Proof. Split on the exponent. For e, the power is the singleton identity endpoint, hence carried. For $\mathsf{E0}(t)$, the recursive power is the singleton zero endpoint, hence carried. For $\mathsf{E1}(t)$, the power is an append product whose right factor is the visible base; append empty-result inversion would force that visible base to be history-same to e, contradicting BEDC constructor separation. The reverse implication is immediate in the two carried exponent cases, and impossible in the one-headed exponent case. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow visible base carrier iff`

Theorem 49.57 (Matrix singleton power visible-base positive-exponent carrier exclusion)**.** *If* e *is unary and not history-same as* e*, then neither visible-base singleton power is carried:*

$$A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), e)) \Rightarrow \bot \qquad \textit{and} \qquad A_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(\mathsf{E1}(m), e)) \Rightarrow \bot.$$

Proof. Visible-base carrier exactness reduces either carrier witness to the empty exponent case or the zero-headed exponent case. The empty branch contradicts the positive-exponent hypothesis. The zero-headed branch contradicts unary history inversion, since unary histories are generated by e and one-headed extensions. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow visible base positive exponent carrier absurd`

Theorem 49.58 (Matrix singleton power positive-exponent visible-base classifier exclusion)**.** *For a unary nonempty exponent e, neither visible-base singleton power can classify any history:*

$$\rho_{\mathsf{e}}(\mathsf{MatrixSingletonPow}(\mathsf{E0}(m), e), h) \Longrightarrow \bot,$$

and the same holds with $\mathsf{E1}(m)$ *in place of* $\mathsf{E0}(m)$*.*

Proof. Project the source carrier from the displayed classifier. Visible-base carrier exactness leaves only the empty-exponent and zero-headed-exponent cases. The first contradicts the nonempty exponent hypothesis, and the second contradicts the unary-generation inversion for e. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonPow positive exponent visible base classifier absurd`

Theorem 49.59 (Matrix singleton add-mul visible-right continuation result nonempty)**.** *For any histories M, n, R, if the shared right input of the displayed singleton addition and multiplication has a visible head, then a continuation from the addition endpoint to the multiplication endpoint cannot have empty result:*

$$\mathsf{Cont}(M \oplus \mathsf{E0}(n), M \odot \mathsf{E0}(n), R) \wedge R\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(n)$ *in place of* $\mathsf{E0}(n)$*.*

Proof. Apply the continuation empty-result factor theorem for displayed singleton addition and multiplication. An empty result forces the right input factor to be empty. This contradicts constructor separation for $\mathsf{E0}(n)$ in the first case and for $\mathsf{E1}(n)$ in the second case. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddMul visible right continuation result nonempty`

49.3 The certificate

Definition 49.60 (Matrix source specification)**.** *The source is the finite index carrier from* $\mathsf{Nat}^{\uparrow}$ *and the scalar source* $\mathsf{Ring}^{\uparrow}(R)$*.*

Definition 49.61 (Matrix pattern specification)**.** *The pattern is finite table formation assigning a scalar witness to each index pair.*

Definition 49.62 (Matrix classifier specification)**.** *The classifier relates matrices pointwise by scalar classifier at every index pair.*

Lemma 49.63 (Matrix singleton classifier append visible-factor exclusion)**.** *For histories h, p, q :* BHist*, the singleton matrix classifier admits no witness whose source or target endpoint is an append with a visible factor:*

1. $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{E0}(p), q), h) \to \bot$ *and* $\rho_{\mathsf{e}}(h, \mathsf{append}(\mathsf{E0}(p), q)) \to \bot$*;*

2. $\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{E1}(p), q), h) \to \bot$ *and* $\rho_{\mathsf{e}}(h, \mathsf{append}(\mathsf{E1}(p), q)) \to \bot$*;*
3. $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{E0}(q)), h) \to \bot$ *and* $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{E0}(q))) \to \bot$*;*
4. $\rho_{\mathsf{e}}(\mathsf{append}(p, \mathsf{E1}(q)), h) \to \bot$ *and* $\rho_{\mathsf{e}}(h, \mathsf{append}(p, \mathsf{E1}(q))) \to \bot$.

Proof. Unfold the singleton matrix classifier. Any displayed classifier witness projects to the singleton carrier endpoint on its source and to the singleton carrier endpoint on its target.

For an endpoint $\mathsf{append}(\mathsf{E0}(p), q)$ or $\mathsf{append}(\mathsf{E1}(p), q)$, append empty-result inversion forces the left visible factor to be empty. Constructor separation contradicts the resulting $\mathsf{E0}(p)\mathsf{hsamee}$ or $\mathsf{E1}(p)\mathsf{hsamee}$ endpoint.

For an endpoint $\mathsf{append}(p, \mathsf{E0}(q))$ or $\mathsf{append}(p, \mathsf{E1}(q))$, the same inversion forces the right visible factor to be empty. Constructor separation gives the source-side contradiction, and the target-side pair is exactly Definition 49.42. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonClassifier append visible factor exclusion`

Definition 49.64 (Matrix stability certificate)**.** *The stability certificate consists of:*

1. *pointwise reflexivity*
2. *pointwise transitivity*
3. *finite index closure*
4. *addition closure by scalar addition*
5. *multiplication closure by finite scalar sums and products*
6. *identity matrix closure when dimensions match*

Definition 49.65 (Matrix ledger policy)**.** *Record the dimension witnesses, index discipline, and scalar ledger for each entry.*

Definition 49.66 ($\mathsf{Mat}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Mat}^{\uparrow}$*.*

Theorem 49.67 (Matrix singleton carrier semantic name certificate)**.** *The singleton matrix carrier* $A_{\mathsf{e}}(h) \equiv h\mathsf{hsamee}$*, used as source, pattern, and ledger policy, together with the ambient classifier* hsame*, forms a semantic name certificate:*

$$\mathsf{SemanticNameCert}(A_{\mathsf{e}}, A_{\mathsf{e}}, A_{\mathsf{e}}, \mathsf{hsame}).$$

Proof. The carrier is inhabited by e. The classifier rows are the reflexive, symmetric, and transitive laws for hsame. Carrier transport sends $h\mathsf{hsame}k$ and $h\mathsf{hsamee}$ to $k\mathsf{hsamee}$ by symmetry followed by transitivity. Since source, pattern, and ledger policy are the same concrete singleton carrier, the two semantic soundness maps are identity projections. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonCarrier semanticNameCert`

Definition 49.68 (Matrix singleton standard comparison datum)**.** *The singleton standard comparison datum for* $\mathsf{Mat}^{\uparrow}$ *is the finite tuple consisting of the singleton carrier* A_{e}*, the classifier* ρ_{e}*, the zero row, the identity row, the addition row, the multiplication row, the unit row, the associativity row, the congruence row, and the visible-factor exclusion rows already named in this chapter.*

Theorem 49.69 (Matrix singleton standard comparison row)**.** *The datum of Definition 49.68 is exactly the bridge-facing singleton matrix row: the host one-point matrix reading can read the tuple and read back to the same BEDC endpoint without adding determinant data, rank data, Gaussian elimination, eigenvalue data, or a host matrix API.*

Proof. The carrier and classifier part is the semantic name certificate of Definition 49.67. It fixes the accepted endpoint as the singleton carrier A_{e} and makes ρ_{e} a comparison through the same empty-history representative.

The operation part is the singleton empty-history law package of Definition 49.48. Its zero, identity, addition, multiplication, unit, associativity, and congruence rows are all endpoint rows on the same singleton carrier.

Finally, the visible-factor exclusion rows force every admitted comparison to remain at the empty endpoint. A host one-point matrix reading therefore has only one value to inspect and reads back to the displayed BEDC endpoint, with no determinant, rank, Gaussian-elimination, eigenvalue, or host matrix operation surface added. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$**.**

Constructive story: A matrix at this closed boundary is the one endpoint that the singleton history carrier accepts: a history may enter only when it is history-same as e, while any visible b0- or b1-headed append factor is rejected by the classifier-exclusion rows. The comparison relation ρ_{e} does not inspect rows and columns from an external table; it asks for the two endpoint carrier witnesses and the underlying hsame link, so all admitted matrices collapse to the same empty-history representative. The source, pattern, and ledger slots are the same singleton predicate here: the source covers only the empty endpoint, the pattern records zero, identity, addition, multiplication, unit, associativity, congruence, and visible-factor exclusion as endpoint calculations, and the ledger keeps the displayed dimension and entry data aligned with the raw empty history. Since e itself inhabits the carrier, hsame supplies reflexivity, symmetry, and transitivity, and carrier transport is just transitivity through e, the five name-certificate obligations hold without adding a separate matrix object. A history h lies in the signature gap of a $\mathsf{Mat}^{\uparrow}$ token p when the source-domain admits h, the signature relation pairs it with an answer endpoint and an evidence ledger, and token introduction names that answer as p; unfolded in this singleton instance, h is history-same to e, the answer is the same empty endpoint, and p names the terminal matrix certificate. The chapter registers this singleton empty-history table under the name $\mathsf{Mat}^{\uparrow}$; matrix operations are append and endpoint laws on the existing history surface, with no general finite-dimensional matrix carrier, determinant theory, rank computation, or host ring matrix API introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The singleton empty-history matrix semantic name certificate over the displayed singleton carrier A_{e} and classifier ρ_{e}, including the public singleton constructor and readback surface of Definition 49.72 together with zero, identity, operation closure, unit, associativity, and congruence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.MatrixUp.MatrixUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: General $n \times n$ matrix theory: rank, determinant beyond singleton, Gaussian elimination, eigenvalue/eigenvector theory, characteristic polynomial, Smith/Jordan normal forms, or matrix theory over arbitrary rings.

Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 49.70 (Singleton empty-history MatrixUp certificate obligations)**.** *For the singleton empty-history matrix data of Definition 49.3, the* $\mathsf{Mat}^{\uparrow}$ *obligations of Definition 49.66 are instantiated as follows:*

1. *the source, pattern, and ledger carriers are the singleton carrier* A_{e} *on* BHist*;*
2. *the classifier is* ρ_{e}*;*
3. *the semantic* NameCert *rows are supplied by Definition 49.67;*
4. *the zero, identity, operation-closure, unit, associativity, and congruence rows are supplied by Definition 49.48.*

Proof. Unpack Definition 49.66. The source, pattern, and ledger slots are all the displayed singleton history carrier A_{e} from Definition 49.3, and the classifier slot is the displayed relation ρ_{e}. The ledger is the same singleton dimension/source ledger, with row and column witnesses represented by e.

The carrier transport and semantic soundness rows are exactly Definition 49.67. That theorem uses the same carrier in the source, pattern, and ledger positions, so its semantic NameCert data supplies the certificate-level carrier and classifier core.

It remains to fill the stability rows. The law package of Definition 49.48 carries 0_M and I_M, closes A_{e} under $\oplus$ and $\odot$, supplies the unit and identity laws, proves associativity under ρ_{e}, and gives classifier congruence for both operations. These rows exhaust the stability portion of the displayed $\mathsf{Mat}^{\uparrow}$ obligations. □

Primitive boundary. $\mathsf{Mat}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 49.71 (Zero-axiom encoding note)**.** *Avoids funext by using pointwise classifier sameness; finite indices are inductive, not quotient indices.*

Theorem 49.72 (Matrix singleton empty public constructor surface)**.** *The singleton empty-history* $\mathsf{Mat}^{\uparrow}$ *package exports the public constructor and readback rows for zero, identity, addition, multiplication, unit, associativity, congruence, and visible-factor exclusion over* A_{e}*,* ρ_{e}*,* BHist*,* Cont*, and* hsame*. These rows are exactly the public surface available from the singleton certificate, and they expose no general finite-dimensional table API.*

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonEmptyHistory public constructor surface`

Proof. Read the singleton data from Definition 49.3. It fixes the carrier A_{e} over BHist, the classifier ρ_{e}, and the empty-history endpoint used by the public readback rows.

Apply Definition 49.48. Its zero, identity, addition, multiplication, unit, associativity, and congruence rows are stated on the same A_{e} and ρ_{e} surface, with Cont and hsame carrying the endpoint transport. The visible-factor exclusion row is the same singleton classifier exclusion already recorded in the MatrixUp certificate surface.

Finally use the MatrixUp closure block grounded at BEDC.Derived.MatrixUp.MatrixSingletonCarrier_semanticNa That block identifies the chapter boundary as the singleton empty-history certificate and explicitly withholds a general finite-dimensional matrix carrier, so the public constructor surface is precisely the listed rows and no table API beyond them. □

Theorem 49.73 (Matrix singleton concrete schema surface)**.** *The concrete singleton matrix carrier* A_{e}*, classifier* ρ_{e}*, zero and identity endpoints, displayed addition and multiplication by* append*, row and column unary witnesses, and the laws of Definition 49.48 form the non-parametric* $\mathsf{Mat}^{\uparrow}$ *target surface over* BHist*,* hsame*,* Cont*, and* Pkg*.*

This surface is the concrete schema available from the singleton package. It does not assert a general finite-table carrier, arbitrary scalar matrix operations, or a parametric row-and-column family beyond the displayed singleton history.

Proof. Read the package data from Definition 49.3. It supplies the carrier A_{e}, classifier ρ_{e}, zero and identity endpoints, append-based addition and multiplication, and the row and column unary witnesses inside the same empty-history BHist surface.

Apply Definition 49.48. The law rows are stated for exactly those singleton endpoints and append operations, with hsame and Cont transporting the finite histories and Pkg recording the package provenance.

Finally use Definition 49.72. It identifies the exported constructor and readback rows as the public singleton surface and excludes any general table API. Therefore the displayed rows are precisely the non-parametric $\mathsf{Mat}^{\uparrow}$ schema target exposed by the concrete singleton package. □

Theorem 49.74 (Singleton additive fold append continuation classifier)**.** *If two finite singleton-matrix spines are carried and a continuation ledger composes their individual additive folds to an endpoint r, then the appended fold classifies r:*

$$\mathsf{MatrixSingletonAddFoldSpineCarrier}(xs) \wedge \mathsf{MatrixSingletonAddFoldSpineCarrier}(ys) \wedge \mathsf{Cont}(\mathsf{MatrixSingletonAddFold}(xs)$$

Proof. The display classifier identifies the appended fold with the canonical append of the two individual folds. The continuation ledger has the same left and right factors as that canonical append, so continuation determinacy transports the classifier endpoint from the canonical append to r. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append continuation classifier`

Theorem 49.75 (Singleton additive fold reverse empty append readback)**.** *If xs is carried by* MatrixSingletonAddFoldSpineCarrier*, then appending the singleton additive fold of xs to* e *is history-same to appending the singleton additive fold of* $\mathsf{reverse}(xs)$ *to* e*.*

Proof. First prove by induction on the spine that the carrier is preserved by list reversal. The nil case is the empty carrier. In the cons case, the induction hypothesis carries the reversed tail, the head supplies the singleton suffix carrier, and append closure for spine carriers carries the reversed cons spine. The fold carrier equivalence then makes both folds empty-carried, and the right-unit law for append identifies the two displayed empty-appended endpoints. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold reverse empty append hsame`

Theorem 49.76 (Singleton additive fold reverse carrier readback)**.** *If xs is carried by* MatrixSingletonAddFoldSpineC *then the reversed singleton additive fold is carried, and the two fold endpoints are history-same:*

$$\mathsf{MatrixSingletonCarrier}(\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(xs))) \wedge \mathsf{MatrixSingletonAddFold}(xs)\mathsf{hsameMatrixSingletonAdd}$$

Proof. Induct through the reverse accumulator to show that carried spines remain carried after reversal. The fold carrier readback then makes both the original fold and the reversed fold singleton-carried. Since the singleton carrier is history-sameness to e, transitivity through e gives the displayed fold readback. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold reverse carrier readback`

Corollary 49.77 (Singleton additive fold double-reverse readback)**.** *If xs is carried by* MatrixSingletonAddFoldSpine *then the singleton additive fold over the double-reversed spine is carried, and the singleton matrix classifier relates it to the original singleton additive fold:*

$$\mathsf{MatrixSingletonCarrier}(\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(\mathsf{reverse}(xs)))) \wedge \mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonAddF}$$

Proof. Apply Definition 49.76 to xs. This gives the carrier of $\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(xs))$ and the comparison

$$\mathsf{MatrixSingletonAddFold}(xs)\mathsf{hsameMatrixSingletonAddFold}(\mathsf{reverse}(xs)).$$

By Definition 49.82, that carried reversed fold reads back to a carried reversed spine. Apply Definition 49.76 again, now to $\mathsf{reverse}(xs)$. This gives the carrier of $\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(\mathsf{reverse}(xs)))$ and

$$\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(xs))\mathsf{hsameMatrixSingletonAddFold}(\mathsf{reverse}(\mathsf{reverse}(xs))).$$

Symmetry and transitivity of hsame compose the two displayed comparisons in the reverse direction, yielding

$$\mathsf{MatrixSingletonAddFold}(\mathsf{reverse}(\mathsf{reverse}(xs)))\mathsf{hsameMatrixSingletonAddFold}(xs).$$

The original spine carrier gives the carrier of $\mathsf{MatrixSingletonAddFold}(xs)$ by Definition 49.82, while the second reverse readback already supplies the carrier of the double-reversed fold. These two carrier rows together with the displayed hsame comparison are exactly the singleton matrix classifier witness. Finite reverse involutivity confirms that the double-reversed spine has the same finite order as the input spine, so the readback uses no additional carrier or permutation datum. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold double reverse readback`

Lemma 49.78 (Finite additive fold singleton identity)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold $\mathsf{fold}_{+,R}$ from Definition 50.26. Assume the additive zero and classifier-equivalence rows retained in Definition 43.26. For every carried scalar a, the singleton fold is classified with a:*

$$\mathsf{fold}_{+,R}(\mathsf{singleton}(a)) \sim_R a.$$

Proof. Unfold the singleton spine as $\mathsf{cons}(a, \mathsf{nil})$. The fold equation identifies its endpoint with $a +_R 0_R$.

The additive right-zero law classifies $a +_R 0_R$ with a. Classifier symmetry and transitivity then transport the displayed fold endpoint to the scalar a. □

Lemma 49.79 (Finite additive fold empty identity)**.** *Under the same scalar data, the empty finite fold is classified with the zero endpoint:*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) \sim_R 0_R.$$

Proof. This is the nil clause recorded by Definition 50.26, followed by classifier reflexivity on the zero endpoint. □

Definition 49.80 (Finite additive scalar fold)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the additive stability rows of Definition 43.26. For every finite carried scalar spine xs, the fold $\mathsf{fold}_{+,R}(xs)$ is generated by the two clauses*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) := 0_R, \qquad \mathsf{fold}_{+,R}(x :: xs) := x +_R \mathsf{fold}_{+,R}(xs).$$

The fold ledger records the finite spine order, the carriedness witness for each entry, and the additive ledger used at every cons step.

Theorem 49.81 (Finite additive fold respects scalar classifier)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the additive stability rows of Definition 43.26. If finite carried scalar spines xs and ys are related by $\mathsf{ListClassifierSpec}(\sim_R, xs, ys)$, then*

$$\mathsf{fold}_{+,R}(xs) \sim_R \mathsf{fold}_{+,R}(ys).$$

Proof. Induct on the displayed finite-spine classifier witness. In the nil case, both folds reduce by Definition 49.80 to 0_R, and the zero self-classification row of Definition 43.26 gives $0_R \sim_R 0_R$.

In the cons case, the head comparison gives $x \sim_R y$, and the induction hypothesis gives $\mathsf{fold}_{+,R}(xs) \sim_R \mathsf{fold}_{+,R}(ys)$. Additive congruence from Definition 43.26 transports these two comparisons to $x +_R \mathsf{fold}_{+,R}(xs) \sim_R y +_R \mathsf{fold}_{+,R}(ys)$, which is exactly the folded comparison after unfolding Definition 49.80. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold respects classifier`

Theorem 49.82 (Singleton additive fold carrier readback)**.** *For the singleton matrix carrier, let* MatrixSingletonAddFold *fold a finite spine by the concrete singleton addition* MatrixSingletonAdd*, and let* MatrixSingletonAddFoldSpineCarrier *record that every spine entry is carried. Then*

$$\mathsf{MatrixSingletonCarrier}(\mathsf{MatrixSingletonAddFold}(xs)) \iff \mathsf{MatrixSingletonAddFoldSpineCarrier}(xs).$$

Proof. Induct on the finite spine. The nil fold is e, so both sides are the empty endpoint readback. In the cons case, singleton addition is kernel append; empty-append inversion splits the folded endpoint into the head carrier and the tail-fold carrier, and the induction hypothesis identifies the tail-fold carrier with the tail spine carrier. □

Lean4 (definition) `BEDC.Derived.MatrixUp.MatrixSingletonAddFold`
Lean4 (definition) `BEDC.Derived.MatrixUp.MatrixSingletonAddFoldSpineCarrier`
Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold carrier iff`

Theorem 49.83 (Singleton additive fold continuation result readback)**.** *For every finite spine xs and histories $h, r : \mathsf{BHist}$, if*

$$\mathsf{Cont}(h, \mathsf{MatrixSingletonAddFold}(xs), r),$$

then the result is carried by the singleton matrix carrier exactly when the left input is carried and every entry of the fold spine is carried:

$$\mathsf{MatrixSingletonCarrier}(r) \iff \mathsf{MatrixSingletonCarrier}(h) \wedge \mathsf{MatrixSingletonAddFoldSpineCarrier}(xs).$$

Proof. For the forward direction, transport the displayed continuation along $\mathsf{MatrixSingletonCarrier}(r)$ to a continuation ending at e. Kernel empty-result inversion for Cont forces both h and $\mathsf{MatrixSingletonAddFold}(xs)$ to be empty. The first equality is the left carrier, and the second is read through Definition 49.82 to obtain the spine carrier.

Conversely, the left carrier identifies h with e, while Definition 49.82 turns the spine carrier into $\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(xs), \mathsf{e})$. Continuation respect for hsame, compared against the empty right-unit continuation, forces r to be carried by the singleton matrix carrier. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold continuation result iff`

Theorem 49.84 (Singleton additive fold visible head exclusion)**.** *For every history m and finite spine xs, folding a singleton additive spine whose head is $\mathsf{E0}(m)$ or $\mathsf{E1}(m)$ cannot read back to the empty carrier:*

$$\neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(\mathsf{E0}(m) :: xs), \mathsf{e}) \quad \wedge \quad \neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(\mathsf{E1}(m) :: xs), \mathsf{e}).$$

Proof. Unfold the cons case of the finite singleton additive fold. The head is the left factor of a kernel append, and empty-append inversion would identify $\mathsf{E0}(m)$ or $\mathsf{E1}(m)$ with e. Both alternatives contradict constructor separation in BHist. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold visible head absurd`

Theorem 49.85 (Singleton additive fold append carrier readback)**.** *For finite singleton-matrix spines xs and ys,*

$$\mathsf{MatrixSingletonCarrier}(\mathsf{MatrixSingletonAddFold}(xs +\!\!+ ys)) \iff \mathsf{MatrixSingletonAddFoldSpineCarrier}(xs) \wedge \mathsf{MatrixSing}$$

Proof. Induct on xs. For the empty spine, the appended fold is the fold of ys, and the base fold readback identifies its carrier with the spine carrier of ys. For a cons spine, concrete singleton addition is kernel append, so empty-append inversion splits the folded endpoint into the head carrier and the appended tail-fold carrier. The induction hypothesis splits the tail-fold carrier into the tail spine carrier and the carrier of ys, and the reverse direction repacks those two splits through the same empty-append constructor. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append carrier iff`

Theorem 49.86 (Singleton additive fold appended visible tail exclusion)**.** *For every history m and finite spine xs, appending a visible singleton tail before folding cannot read back to the empty carrier:*

$$\neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(xs{+\!\!+}[\mathsf{E0}(m)]), \mathsf{e}) \quad \wedge \quad \neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(xs{+\!\!+}[\mathsf{E1}(m)]), \mathsf{e}).$$

Proof. Apply the appended-spine carrier readback to split the fold over xs and the one-element visible tail. The tail carrier would identify either $\mathsf{E0}(m)$ or $\mathsf{E1}(m)$ with e, contradicting constructor separation in BHist. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append visible tail absurd`

Theorem 49.87 (Singleton additive fold append visible middle exclusion)**.** *For finite singleton-matrix spines $pref$ and $suffix$, a visible middle entry remains excluded after an arbitrary carried prefix:*

$$\neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(pref +\!\!+ \mathsf{E0}(m) :: suffix), \mathsf{e}) \quad \wedge \quad \neg\mathsf{hsame}(\mathsf{MatrixSingletonAddFold}(pref +\!\!+ \mathsf{E1}(r$$

Proof. Apply append-carrier readback to the split spine. If the whole appended fold read back to e, then the suffix-side spine carrier would include the visible middle entry as a singleton carrier. That carrier identifies $\mathsf{E0}(m)$ or $\mathsf{E1}(m)$ with e, contradicting constructor separation for histories. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append visible middle absurd`

Theorem 49.88 (Singleton additive fold append readback)**.** *For finite singleton-matrix spines xs and ys, folding their appended spine by concrete singleton addition is history-same as appending the two individual singleton additive folds:*

$$\mathsf{MatrixSingletonAddFold}(xs{+\!\!+}ys)\mathsf{hsameappend}(\mathsf{MatrixSingletonAddFold}(xs), \mathsf{MatrixSingletonAddFold}(ys)).$$

Proof. Induct on xs. The nil case is the left-unit law for kernel append. In the cons case, unfold singleton addition as kernel append, apply the induction hypothesis under the common head, and reassociate by the kernel append associativity law. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append hsame`

Theorem 49.89 (Singleton additive fold append display classifier)**.** *For finite singleton-matrix spines xs and ys, the classifier between the fold of the appended spine and the append of the two individual folds is equivalent to carriedness of both input spines:*

$$\mathsf{MatrixSingletonClassifier}(\mathsf{MatrixSingletonAddFold}(xs{+\!\!+}ys), \mathsf{append}(\mathsf{MatrixSingletonAddFold}(xs), \mathsf{MatrixSingletonAdd}$$

Proof. In the forward direction, project the classifier to its left carrier row and use append-spine carrier readback. Conversely, read both spine carriers back to carriers for the individual folds; empty-append inversion gives the carrier of the displayed append, while fold append readback supplies the classifier comparison between the appended fold and the displayed append endpoint. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold append display classifier iff`

Lemma 49.90 (Finite Kronecker selector sum)**.** *Let $J = \mathsf{Fin}(p)$ be a finite index carrier licensed by the $\mathsf{Nat}^{\uparrow}$ certificate of chapter 28. Let R be a scalar carrier from the $\mathsf{CommRing}^{\uparrow}$ certificate of chapter 44, with classifier $\sim_R$, operations $+_R, \cdot_R$, and scalars $0_R, 1_R$. Assume a finite-index sum certificate Σ_J satisfying classifier congruence and the selector row: if $j_0 : J$, $t : J \to R$ is carried pointwise, $t(j_0) \sim_R u$, and $t(j) \sim_R 0_R$ for every $j \neq j_0$, then $\Sigma_J(t) \sim_R u$. For every pointwise carried scalar family $x : J \to R$ and every $j_0 : J$,*

$$\Sigma_J(j \mapsto \delta_{j_0,j} \cdot_R x(j)) \sim_R x(j_0) \qquad \textit{and} \qquad \Sigma_J(j \mapsto x(j) \cdot_R \delta_{j,j_0}) \sim_R x(j_0),$$

where $\delta_{a,b}$ is 1_R when the finite-index equality certificate identifies a with b, and 0_R otherwise.

Proof. For the first displayed sum, split the finite index j. At the selected index $j = j_0$, the Kronecker entry is 1_R, so the scalar left-one law gives $\delta_{j_0,j_0} \cdot_R x(j_0) \sim_R x(j_0)$. At every other index, the Kronecker entry is 0_R, so scalar left-zero absorption gives $\delta_{j_0,j} \cdot_R x(j) \sim_R 0_R$. These are exactly the selected and off-selected hypotheses of the selector row for Σ_J, applied to the summand family $j \mapsto \delta_{j_0,j} \cdot_R x(j)$ with $u = x(j_0)$.

The second displayed sum uses the same selector-row hypothesis for $j \mapsto x(j) \cdot_R \delta_{j,j_0}$. The scalar right-one law gives the selected comparison at $j = j_0$, and scalar right-zero absorption gives the comparison with 0_R away from j_0. □

Theorem 49.91 (Matrix identity multiplication unit)**.** *Let $n, m : \mathsf{Nat}^{\uparrow}$ and let R be a scalar $\mathsf{CommRing}^{\uparrow}$ certificate. Consider the $\mathsf{Mat}^{\uparrow}$ carrier of Definition 49.1 with the pointwise classifier of Definition 49.62. Assume the stability data of Definition 49.64 supplies finite product sums over $\mathsf{Fin}(n)$ and $\mathsf{Fin}(m)$, the scalar zero and one multiplication laws over R, and carried Kronecker identity matrices I_n and I_m with entries*

$$(I_n)_{i,a} \sim_R \delta_{i,a} \qquad \textit{and} \qquad (I_m)_{b,j} \sim_R \delta_{b,j}.$$

If $M : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R$ is carried, then the products $I_n \cdot M$ and $M \cdot I_m$ are carried and

$$I_n \cdot M \sim_{\mathrm{Mat}} M \qquad \textit{and} \qquad M \cdot I_m \sim_{\mathrm{Mat}} M.$$

Proof. The carriedness of I_n and I_m is part of the Kronecker identity closure field, and M is carried by hypothesis. Multiplication closure in Definition 49.64 therefore carries both products. It remains to prove the pointwise scalar comparisons required by Definition 49.62.

Fix $i : \mathsf{Fin}(n)$ and $b : \mathsf{Fin}(m)$. The left product entry is the finite sum

$$(I_n \cdot M)_{i,b} \sim_R \Sigma_{\mathsf{Fin}(n)}(a \mapsto (I_n)_{i,a} \cdot_R M_{a,b}).$$

For each inner index a, identity-entry closure gives $(I_n)_{i,a} \sim_R \delta_{i,a}$, and scalar classifier congruence for $\cdot_R$ gives

$$(I_n)_{i,a} \cdot_R M_{a,b} \sim_R \delta_{i,a} \cdot_R M_{a,b}.$$

The finite-sum classifier congruence for $\Sigma_{\mathsf{Fin}(n)}$ yields

$$\Sigma_{\mathsf{Fin}(n)}(a \mapsto (I_n)_{i,a} \cdot_R M_{a,b}) \sim_R \Sigma_{\mathsf{Fin}(n)}(a \mapsto \delta_{i,a} \cdot_R M_{a,b}).$$

Applying Definition 49.90 to the carried family $a \mapsto M_{a,b}$ selects the unique inner index $a = i$, hence $(I_n \cdot M)_{i,b} \sim_R M_{i,b}$.

For the right product, the entry at the same outer indices is

$$(M \cdot I_m)_{i,b} \sim_R \Sigma_{\mathsf{Fin}(m)}(a \mapsto M_{i,a} \cdot_R (I_m)_{a,b}).$$

For each inner index a, identity-entry closure gives $(I_m)_{a,b} \sim_R \delta_{a,b}$, and scalar classifier congruence for $\cdot_R$ gives

$$M_{i,a} \cdot_R (I_m)_{a,b} \sim_R M_{i,a} \cdot_R \delta_{a,b}.$$

The finite-sum classifier congruence for $\Sigma_{\mathsf{Fin}(m)}$ yields

$$\Sigma_{\mathsf{Fin}(m)}(a \mapsto M_{i,a} \cdot_R (I_m)_{a,b}) \sim_R \Sigma_{\mathsf{Fin}(m)}(a \mapsto M_{i,a} \cdot_R \delta_{a,b}).$$

The right-hand selector part of Definition 49.90 applied to the carried family $a \mapsto M_{i,a}$ selects $a = b$, giving $(M \cdot I_m)_{i,b} \sim_R M_{i,b}$.

Since these scalar comparisons hold for every row and column index, the pointwise matrix classifier gives both displayed matrix identity laws. The ledger in Definition 49.65 records the same row and column witnesses throughout the two products, so no dimension data is changed. □

Definition 49.92 (Finite-fold matrix identity data)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, unit endpoint 1_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds as in Definition 50.26. For each finite dimension $n : \mathsf{Nat}^{\uparrow}$, the finite-fold identity matrix I_n is the carried $n \times n$ matrix whose (i, j) entry is 1_R when the finite indices agree and 0_R otherwise.*

Theorem 49.93 (Finite-fold matrix identity law)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, unit endpoint 1_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds over finite index spines as in Definition 50.26. Let A be a carried $n \times m$ $\mathsf{Mat}^{\uparrow}$ matrix. Assume the scalar classifier equivalence fields, additive zero laws, multiplication congruence, zero multiplication laws, and left and right unit laws retained in Definition 43.26. Then the finite-fold products with the identity matrices of Definition 49.92 are carried and*

$$I_n \cdot_{\mathrm{Mat}} A \sim_{\mathrm{Mat}} A \qquad \textit{and} \qquad A \cdot_{\mathrm{Mat}} I_m \sim_{\mathrm{Mat}} A.$$

Proof. The multiplication closure row of Definition 49.64 carries both products. By Definition 49.62, it is enough to compare scalar entries.

For the left identity, fix $i : \mathsf{Fin}(n)$ and $j : \mathsf{Fin}(m)$. The finite fold for $(I_n \cdot_{\mathrm{Mat}} A)_{i,j}$ has one diagonal summand $1_R \cdot_R A_{i,j}$ and all other summands $0_R \cdot_R A_{k,j}$. The unit law classifies the diagonal summand with $A_{i,j}$, and the zero multiplication law classifies every off-diagonal summand with 0_R.

The finite additive fold over a spine with one carried scalar and all remaining entries classified with 0_R reduces, by the additive zero laws and fold congruence from Definition 50.26, to that single scalar. Hence $(I_n \cdot_{\mathrm{Mat}} A)_{i,j} \sim_R A_{i,j}$.

The right identity is symmetric. The fold for $(A \cdot_{\mathrm{Mat}} I_m)_{i,j}$ has diagonal summand $A_{i,j} \cdot_R 1_R$ and off-diagonal summands classified with 0_R. The right unit law, zero multiplication laws, additive zero laws, and fold congruence again collapse the fold to $A_{i,j}$. Packaging the two entrywise comparisons gives the displayed matrix classifier laws. □

Definition 49.94 (Two-index finite fold swap certificate)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold from Definition 50.26. Let J and K be finite index carriers, and write Σ_J and Σ_K for that fold applied to the finite spines enumerating J and K. A two-index finite fold swap certificate for (J, K) is the following stability field: for every pointwise carried scalar family $s : J \times K \to R$ and every pointwise classifier-compatible transpose $s^\tau : K \times J \to R$ satisfying*

$$s^\tau(k, j) \sim_R s(j, k)$$

for all $j : J$ and $k : K$, the two iterated folds are scalar-classified:

$$\Sigma_J(j \mapsto \Sigma_K(k \mapsto s(j, k))) \sim_R \Sigma_K(k \mapsto \Sigma_J(j \mapsto s^\tau(k, j))).$$

Lemma 49.95 (Scalar product distributes across finite additive folds)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold from Definition 50.26. Assume the ring stability fields of Definition 43.26: additive congruence, multiplication congruence, left and right distributivity, and zero multiplication laws. For every carried scalar c and every finite carried scalar spine xs, let $xs \cdot_R c$ denote the spine obtained by multiplying each entry of xs on the right by c, and let $c \cdot_R xs$ denote the spine obtained by multiplying each entry on the left by c. Then*

$$\mathsf{fold}_{+,R}(xs) \cdot_R c \sim_R \mathsf{fold}_{+,R}(xs \cdot_R c)$$

and

$$c \cdot_R \mathsf{fold}_{+,R}(xs) \sim_R \mathsf{fold}_{+,R}(c \cdot_R xs).$$

Proof. Induct on the finite spine xs. In the nil case, Definition 50.26 identifies the fold with 0_R. The right and left zero multiplication laws from Definition 43.26 identify $0_R \cdot_R c$ and $c \cdot_R 0_R$ with 0_R, which is the fold of the corresponding empty product spine.

In the cons case, write $xs = \mathsf{cons}(a, xs')$. For right multiplication, the fold equation gives $(a +_R \mathsf{fold}_{+,R}(xs')) \cdot_R c$. Right distributivity identifies this scalar with

$$(a \cdot_R c) +_R (\mathsf{fold}_{+,R}(xs') \cdot_R c).$$

The induction hypothesis identifies the second summand with $\mathsf{fold}_{+,R}(xs' \cdot_R c)$. Additive congruence then gives

$$(a \cdot_R c) +_R \mathsf{fold}_{+,R}(xs' \cdot_R c),$$

which is the cons fold of $xs \cdot_R c$.

The left multiplication proof is the same induction with left distributivity applied to $c \cdot_R (a +_R \mathsf{fold}_{+,R}(xs'))$. It yields

$$(c \cdot_R a) +_R (c \cdot_R \mathsf{fold}_{+,R}(xs')),$$

then uses the induction hypothesis and additive congruence to identify the result with the cons fold of $c \cdot_R xs$. □

Theorem 49.96 (Matrix multiplication associativity from finite folds)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, addition $+_R$, multiplication $\cdot_R$, zero endpoint 0_R, and finite additive fold from Definition 50.26. Let $n, m, p, q : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R, \qquad C : \mathsf{Fin}(p) \times \mathsf{Fin}(q) \to R$$

be carried matrices. Assume matrix multiplication is the finite-fold operation

$$(X \cdot Y)_{u,w} := \Sigma_V(v \mapsto X_{u,v} \cdot_R Y_{v,w})$$

for the matching inner finite index carrier V, using the multiplication closure field of Definition 49.64. Assume the scalar ring stability fields of Definition 43.26, the fold congruence theorem Definition 50.26, and the two-index finite fold swap certificate of Definition 49.94 for $\mathsf{Fin}(p)$ and $\mathsf{Fin}(m)$. Then the two matched products are carried and

$$(A \cdot B) \cdot C \sim_{\mathrm{Mat}} A \cdot (B \cdot C).$$

Proof. The multiplication closure item of Definition 49.64 carries $A \cdot B$ and $B \cdot C$, and then carries the two outer products $(A \cdot B) \cdot C$ and $A \cdot (B \cdot C)$. By Definition 49.62, it remains to prove the scalar classifier comparison at each entry.

Fix $i : \mathsf{Fin}(n)$ and $l : \mathsf{Fin}(q)$, and set

$$t(b, a) := A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l})$$

for $a : \mathsf{Fin}(m)$ and $b : \mathsf{Fin}(p)$. Expanding the left-associated matrix product gives

$$((A \cdot B) \cdot C)_{i,l} \sim_R \Sigma_{b:\mathsf{Fin}(p)}((A \cdot B)_{i,b} \cdot_R C_{b,l}).$$

For each fixed b, expanding $(A \cdot B)_{i,b}$ and applying the right-multiplication half of Definition 49.95 gives

$$(A \cdot B)_{i,b} \cdot_R C_{b,l} \sim_R \Sigma_{a:\mathsf{Fin}(m)}((A_{i,a} \cdot_R B_{a,b}) \cdot_R C_{b,l}).$$

Scalar associativity from the multiplicative monoid part of Definition 43.26 gives a pointwise comparison

$$(A_{i,a} \cdot_R B_{a,b}) \cdot_R C_{b,l} \sim_R A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l}).$$

The finite-fold congruence theorem Definition 50.26 lifts these pointwise comparisons through the inner and outer folds. Hence

$$((A \cdot B) \cdot C)_{i,l} \sim_R \Sigma_{b:\mathsf{Fin}(p)}\Sigma_{a:\mathsf{Fin}(m)}t(b, a).$$

Expanding the right-associated matrix product gives

$$(A \cdot (B \cdot C))_{i,l} \sim_R \Sigma_{a:\mathsf{Fin}(m)}(A_{i,a} \cdot_R (B \cdot C)_{a,l}).$$

For each fixed a, expanding $(B \cdot C)_{a,l}$ and applying the left-multiplication half of Definition 49.95 gives

$$A_{i,a} \cdot_R (B \cdot C)_{a,l} \sim_R \Sigma_{b:\mathsf{Fin}(p)}A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l}).$$

Again Definition 50.26 lifts the pointwise comparisons through the finite folds, so

$$(A \cdot (B \cdot C))_{i,l} \sim_R \Sigma_{a:\mathsf{Fin}(m)}\Sigma_{b:\mathsf{Fin}(p)}t(b, a).$$

Apply the fold swap certificate of Definition 49.94 to the family t and its transpose $t^\tau(a,b) := t(b,a)$. This gives

$$\Sigma_{b:\mathsf{Fin}(p)}\Sigma_{a:\mathsf{Fin}(m)}t(b,a) \sim_R \Sigma_{a:\mathsf{Fin}(m)}\Sigma_{b:\mathsf{Fin}(p)}t(b,a).$$

Classifier transitivity composes the left expansion, this fold swap comparison, and the symmetry of the right expansion. Thus every entry of $(A \cdot B) \cdot C$ is scalar-classified with the corresponding entry of $A \cdot (B \cdot C)$. The pointwise matrix classifier of Definition 49.62 gives the displayed matrix classification, and the unchanged dimension witnesses are recorded by the ledger policy of Definition 49.65. □

Theorem 49.97 (Matrix multiplication classifier congruence)**.** *Let R be a scalar $\mathsf{Ring}^\uparrow$ carrier with classifier $\sim_R$, multiplication $\cdot_R$, addition $+_R$, zero endpoint 0_R, and finite additive fold from Definition 50.26. Let $n, m, p : \mathsf{Nat}^\uparrow$, and consider the $\mathsf{Mat}^\uparrow$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A, A' : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B, B' : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R$$

be carried matrices over the same scalar source and the same matching inner dimension m. Assume matrix multiplication is the finite-fold operation

$$(X \cdot Y)_{i,k} := \Sigma_{\mathsf{Fin}(m)}(j \mapsto X_{i,j} \cdot_R Y_{j,k})$$

for the shared inner finite index carrier, using the multiplication closure field of Definition 49.64. Assume the scalar multiplication congruence field of Definition 43.26 and the fold congruence theorem Definition 50.26. If

$$A \sim_{\mathrm{Mat}} A' \qquad \text{and} \qquad B \sim_{\mathrm{Mat}} B',$$

then $A \cdot B$ and $A' \cdot B'$ are carried $\mathsf{Mat}^\uparrow$ matrices, and

$$A \cdot B \sim_{\mathrm{Mat}} A' \cdot B'.$$

Proof. The multiplication closure item in Definition 49.64 applies to the carried compatible pairs (A, B) and (A', B'), so both products $A \cdot B$ and $A' \cdot B'$ are carried by the $\mathsf{Fin}(n) \times \mathsf{Fin}(p)$ matrix carrier.

By Definition 49.62, it remains to prove the scalar classifier comparison entrywise. Fix $i : \mathsf{Fin}(n)$ and $k : \mathsf{Fin}(p)$. The matrix classifier hypotheses give, for every $j : \mathsf{Fin}(m)$,

$$A_{i,j} \sim_R A'_{i,j} \qquad \text{and} \qquad B_{j,k} \sim_R B'_{j,k}.$$

The multiplication congruence field in Definition 43.26 therefore gives the summand comparison

$$A_{i,j} \cdot_R B_{j,k} \sim_R A'_{i,j} \cdot_R B'_{j,k}$$

for every inner index j.

The shared inner dimension supplies the same finite enumeration of $\mathsf{Fin}(m)$ on both sides. Hence these summand comparisons form a same-length finite-spine classifier witness between

$$j \mapsto A_{i,j} \cdot_R B_{j,k} \qquad \text{and} \qquad j \mapsto A'_{i,j} \cdot_R B'_{j,k}.$$

Applying Definition 50.26 to that finite-spine witness yields

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto A_{i,j} \cdot_R B_{j,k}) \sim_R \Sigma_{\mathsf{Fin}(m)}(j \mapsto A'_{i,j} \cdot_R B'_{j,k}).$$

By the displayed definition of matrix multiplication, this is exactly

$$(A \cdot B)_{i,k} \sim_R (A' \cdot B')_{i,k}.$$

Since i and k were arbitrary, Definition 49.62 packages the entrywise scalar comparisons as $A \cdot B \sim_{\mathrm{Mat}} A' \cdot B'$. The dimension and scalar-source ledgers are unchanged and are recorded by Definition 49.65. □

Lemma 49.98 (Finite additive fold separates pointwise scalar sums)**.** *Let R be a scalar* $\mathsf{Ring}^{\uparrow}$ *carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold* $\mathsf{fold}_{+,R}$ *from Definition 50.26. Assume the additive abelian-group laws and additive congruence fields retained in Definition 43.26. For any two finite carried scalar spines xs and ys over the same finite enumeration, write $xs +_R ys$ for the spine obtained by adding matching entries. Then*

$$\mathsf{fold}_{+,R}(xs +_R ys) \sim_R \mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys).$$

Proof. Induct on the shared finite enumeration. In the nil case, all three spines are nil. The left endpoint reduces by Definition 50.26 to 0_R, while the right endpoint reduces to $0_R +_R 0_R$. The additive left-zero law in Definition 43.26, with classifier symmetry from the additive abelian-group laws, gives $0_R \sim_R 0_R +_R 0_R$.

In the cons case, write $xs = \mathsf{cons}(a, xs')$ and $ys = \mathsf{cons}(b, ys')$. The induction hypothesis gives

$$\mathsf{fold}_{+,R}(xs' +_R ys') \sim_R \mathsf{fold}_{+,R}(xs') +_R \mathsf{fold}_{+,R}(ys').$$

Source reflexivity on $a +_R b$ and additive congruence transport this comparison to

$$(a +_R b) +_R \mathsf{fold}_{+,R}(xs' +_R ys') \sim_R (a +_R b) +_R (\mathsf{fold}_{+,R}(xs') +_R \mathsf{fold}_{+,R}(ys')).$$

Apply Definition 42.13, read for the additive abelian-group operation of the scalar ring, to classify the right endpoint with

$$(a +_R \mathsf{fold}_{+,R}(xs')) +_R (b +_R \mathsf{fold}_{+,R}(ys')).$$

The cons clauses of Definition 50.26 identify this last endpoint with

$$\mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys),$$

and identify the first endpoint with $\mathsf{fold}_{+,R}(xs +_R ys)$. Classifier transitivity gives the cons case. □

Theorem 49.99 (Matrix multiplication right-distributes over matrix addition)**.** *Let R be a scalar* $\mathsf{Ring}^{\uparrow}$ *carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds over finite index spines as in Definition 50.26. Let $n, m, p : \mathsf{Nat}^{\uparrow}$, and consider the* $\mathsf{Mat}^{\uparrow}$ *carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A, B : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad C : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R$$

be carried matrices. Assume matrix addition is the pointwise operation

$$(X + Y)_{u,v} := X_{u,v} +_R Y_{u,v}$$

using the addition closure field of Definition 49.64, and assume matrix multiplication is the finite-fold operation

$$(X \cdot Y)_{u,w} := \Sigma_{\mathsf{Fin}(m)}(j \mapsto X_{u,j} \cdot_R Y_{j,w})$$

for the shared inner finite index carrier, using the multiplication closure field of Definition 49.64. Assume the scalar ring stability fields of Definition 43.26: additive abelian-group laws, addition and multiplication congruence, and right distributivity. Then the displayed matrices are carried and

$$(A + B) \cdot C \sim_{\mathrm{Mat}} (A \cdot C) + (B \cdot C).$$

Proof. The addition closure item of Definition 49.64 carries $A + B$. Multiplication closure carries $(A + B) \cdot C$, $A \cdot C$, and $B \cdot C$, and addition closure then carries $(A \cdot C) + (B \cdot C)$.

By Definition 49.62, it remains to compare corresponding scalar entries. Fix $i : \mathsf{Fin}(n)$ and $k : \mathsf{Fin}(p)$. Over the finite spine enumerating $\mathsf{Fin}(m)$, define

$$xs(j) := A_{i,j} \cdot_R C_{j,k}, \qquad ys(j) := B_{i,j} \cdot_R C_{j,k}.$$

The pointwise matrix-addition definition and scalar right distributivity give, for each $j : \mathsf{Fin}(m)$,

$$(A + B)_{i,j} \cdot_R C_{j,k} \sim_R (A_{i,j} \cdot_R C_{j,k}) +_R (B_{i,j} \cdot_R C_{j,k}).$$

These pointwise comparisons form a finite-spine classifier witness. Applying Definition 50.26 gives

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto (A + B)_{i,j} \cdot_R C_{j,k}) \sim_R \Sigma_{\mathsf{Fin}(m)}(j \mapsto xs(j) +_R ys(j)).$$

By Definition 49.98, the right endpoint is classified with

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto xs(j)) +_R \Sigma_{\mathsf{Fin}(m)}(j \mapsto ys(j)).$$

Unfolding the displayed matrix-multiplication operation identifies these two folded summands with $(A \cdot C)_{i,k}$ and $(B \cdot C)_{i,k}$, and the pointwise matrix-addition operation identifies their scalar sum with $((A \cdot C) + (B \cdot C))_{i,k}$. Classifier transitivity therefore gives

$$((A + B) \cdot C)_{i,k} \sim_R ((A \cdot C) + (B \cdot C))_{i,k}.$$

Since i and k were arbitrary, Definition 49.62 packages these entrywise scalar comparisons as

$$(A + B) \cdot C \sim_{\mathrm{Mat}} (A \cdot C) + (B \cdot C).$$

The dimensions and shared scalar source are unchanged throughout; Definition 49.65 records the common row, contraction, and column witnesses. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonMul right distributes over add`

Lemma 49.100 (Finite additive fold splits pointwise sums)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold $\mathsf{fold}_{+,R}$ from Definition 49.80. Assume the scalar classifier equivalence fields and the additive abelian-group, additive congruence, and zero rows of Definition 43.26. For same-indexed finite carried scalar spines xs and ys, write $xs +_R ys$ for the pointwise sum spine. Then*

$$\mathsf{fold}_{+,R}(xs +_R ys) \sim_R \mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys).$$

Proof. Induct on the common finite spine carrying xs and ys. In the nil case, all three folds reduce by Definition 49.80 to 0_R. The additive zero rows classify 0_R with $0_R +_R 0_R$, giving the displayed comparison.

In the cons case, write $xs = x :: xs'$ and $ys = y :: ys'$. The fold equation reduces the left endpoint to

$$(x +_R y) +_R \mathsf{fold}_{+,R}(xs' +_R ys').$$

The induction hypothesis and additive congruence replace the tail fold by $\mathsf{fold}_{+,R}(xs') +_R \mathsf{fold}_{+,R}(ys')$. The additive associativity and commutativity fields in Definition 43.26 rebracket and exchange the middle summands, classifying

$$(x +_R y) +_R (\mathsf{fold}_{+,R}(xs') +_R \mathsf{fold}_{+,R}(ys'))$$

with

$$(x +_R \mathsf{fold}_{+,R}(xs')) +_R (y +_R \mathsf{fold}_{+,R}(ys')).$$

By Definition 49.80, the latter endpoint is $\mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys)$. Classifier transitivity composes the displayed comparisons. □

Theorem 49.101 (Matrix multiplication left-distributes over matrix addition)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds over finite index spines as in Definition 50.26. Let $n, m, p : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B, C : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R$$

be carried matrices. Assume matrix addition is pointwise and matrix multiplication is the finite-fold operation, using the addition and multiplication closure rows of Definition 49.64. Assume the scalar ring stability fields of Definition 43.26: additive abelian-group laws, addition and multiplication congruence, and left distributivity. Then the displayed matrices are carried and

$$A \cdot_{\mathrm{Mat}} (B + C) \sim_{\mathrm{Mat}} (A \cdot_{\mathrm{Mat}} B) + (A \cdot_{\mathrm{Mat}} C).$$

Proof. Closure follows from the matrix addition and multiplication rows of Definition 49.64. By Definition 49.62, compare entries.

Fix $i : \mathsf{Fin}(n)$ and $k : \mathsf{Fin}(p)$. Over the finite spine enumerating $\mathsf{Fin}(m)$, define

$$xs(j) := A_{i,j} \cdot_R B_{j,k}, \qquad ys(j) := A_{i,j} \cdot_R C_{j,k}.$$

The pointwise addition definition and scalar left distributivity classify each summand $A_{i,j} \cdot_R (B + C)_{j,k}$ with $xs(j) +_R ys(j)$.

Fold congruence lifts these pointwise comparisons through the finite fold. Then Definition 49.98 separates the fold of pointwise sums into the sum of the two finite folds. Unfolding matrix multiplication and pointwise matrix addition identifies the endpoint with $((A \cdot_{\mathrm{Mat}} B) + (A \cdot_{\mathrm{Mat}} C))_{i,k}$. Since i and k were arbitrary, the pointwise classifier packages the displayed law. □

Theorem 49.102 (Finite-fold matrix transpose reverses multiplication)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with opposite multiplication data as in Definition 43.35. Let A be a carried $n \times m$ matrix and B a carried $m \times p$ matrix. Assume matrix multiplication is the finite scalar-fold operation and matrix transpose swaps the two finite index coordinates pointwise. Then*

$$(A \cdot_{\mathrm{Mat}} B)^T \sim_{\mathrm{Mat}} B^T \cdot_{\mathrm{Mat}} A^T$$

over the opposite scalar multiplication.

Proof. The transpose carrier row swaps the row and column witnesses, and multiplication closure carries both displayed matrices. Entrywise, the left side at (k, i) is the fold of $A_{i,j} \cdot_R B_{j,k}$ over the shared inner finite spine.

The right side in the opposite scalar multiplication is the fold of $B_{j,k} \cdot_{R^{\mathrm{op}}} A_{i,j}$ over the same finite spine. By Definition 43.35, each opposite product is the same scalar endpoint as $A_{i,j} \cdot_R B_{j,k}$. Fold congruence from Definition 50.26 compares the two folds, and the pointwise matrix classifier packages the transpose law. □

Theorem 49.103 (Matrix addition commutativity from scalar additive commutativity)**.** *Let R carry a scalar $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and addition $+_R$. Assume the additive abelian-group rows retained in Definition 43.26, in particular the scalar additive commutativity row*

$$a +_R b \sim_R b +_R a$$

for all carried scalar endpoints a, b. Let $n, m : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. For carried matrices

$$A, B : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R,$$

assume matrix addition is the pointwise operation

$$(X + Y)_{i,j} := X_{i,j} +_R Y_{i,j}$$

using the addition closure row of Definition 49.64. Then both $A+B$ and $B+A$ are carried matrices and

$$A + B \sim_{\mathrm{Mat}} B + A.$$

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAdd comm classifier`

Proof. The addition closure row of Definition 49.64 carries $A + B$ from the carried inputs A and B, and carries $B + A$ from the same inputs in the opposite order.

By Definition 49.62, it remains to compare corresponding scalar entries. Fix $i : \mathsf{Fin}(n)$ and $j : \mathsf{Fin}(m)$. Since A and B are carried matrices, the entries $A_{i,j}$ and $B_{i,j}$ are carried scalar endpoints. The pointwise addition rule gives

$$(A + B)_{i,j} = A_{i,j} +_R B_{i,j}$$

and

$$(B + A)_{i,j} = B_{i,j} +_R A_{i,j}.$$

The scalar additive commutativity row retained in Definition 43.26 gives

$$A_{i,j} +_R B_{i,j} \sim_R B_{i,j} +_R A_{i,j}.$$

Substituting the two pointwise addition clauses gives

$$(A + B)_{i,j} \sim_R (B + A)_{i,j}.$$

Since i and j were arbitrary, Definition 49.62 packages these entrywise scalar comparisons as

$$A + B \sim_{\mathrm{Mat}} B + A.$$

The dimensions and scalar source are unchanged throughout, and Definition 49.65 records the same row and column witnesses for the two pointwise sums. □

Theorem 49.104 (Matrix addition associativity from scalar additive associativity)**.** *Let R carry a scalar $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and addition $+_R$. Assume the additive abelian-group rows retained in Definition 43.26, in particular the scalar additive associativity row*

$$(a +_R b) +_R c \sim_R a +_R (b +_R c)$$

for all carried scalar endpoints a, b, c. Let $n, m : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. For carried matrices

$$A, B, C : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R,$$

assume matrix addition is the pointwise operation

$$(X + Y)_{i,j} := X_{i,j} +_R Y_{i,j}$$

using the addition closure row of Definition 49.64. Then $A+B$, $B+C$, $(A+B)+C$, and $A+(B+C)$ are carried matrices, and

$$(A + B) + C \sim_{\mathrm{Mat}} A + (B + C).$$

Proof. The addition closure row of Definition 49.64 carries $A + B$ and $B + C$ from the carried inputs. Applying the same row once more carries $(A + B) + C$ and $A + (B + C)$.

By Definition 49.62, it remains to compare corresponding scalar entries. Fix $i : \mathsf{Fin}(n)$ and $j : \mathsf{Fin}(m)$. Since A, B, and C are carried matrices, the entries $A_{i,j}$, $B_{i,j}$, and $C_{i,j}$ are carried scalar endpoints. The pointwise addition rule gives

$$((A + B) + C)_{i,j} = (A_{i,j} +_R B_{i,j}) +_R C_{i,j}$$

and

$$(A + (B + C))_{i,j} = A_{i,j} +_R (B_{i,j} +_R C_{i,j}).$$

The scalar additive associativity row retained in Definition 43.26 therefore gives

$$((A + B) + C)_{i,j} \sim_R (A + (B + C))_{i,j}.$$

Since i and j were arbitrary, Definition 49.62 packages these entrywise scalar comparisons as

$$(A + B) + C \sim_{\mathrm{Mat}} A + (B + C).$$

The row and column witnesses are unchanged throughout, so Definition 49.65 records the same dimension ledger for the two parenthesized pointwise sums. □

Lemma 49.105 (Rectangular scalar fold reindexing)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive folds over finite index spines as in Definition 50.26. Let J and K be finite index carriers with chosen finite enumerating spines, and let $s : J \times K \to R$ be a pointwise carried scalar family. Assume the scalar classifier equivalence fields and the additive associativity, commutativity, zero, and congruence rows retained in Definition 43.26. Then the two rectangular iterated folds are scalar-classified:*

$$\Sigma_{j:J}\Sigma_{k:K} s(j, k) \sim_R \Sigma_{k:K}\Sigma_{j:J} s(j, k).$$

Proof. Flatten both iterated folds to finite spines over the product index carrier $J \times K$. For the left fold, use the enumeration that lists the K-fiber inside each $j : J$; for the right fold, use the enumeration that lists the J-fiber inside each $k : K$.

The two flattened spines contain the same carried scalar entries, related by the coordinate-swap bijection $(j, k) \mapsto (k, j)$. Additive associativity and commutativity make the finite additive fold invariant under this finite permutation, while fold congruence transports the pointwise reflexive scalar classifier witnesses through the two folds.

Equivalently, apply the two-index fold swap certificate of Definition 49.94 to the family s and to the transpose family $s^\tau(k, j) := s(j, k)$. The transpose compatibility witness is reflexive at every scalar entry, so the certificate gives exactly the displayed reindexing comparison. □

Theorem 49.106 (Matrix product associativity from scalar folds)**.** *Let R be a scalar $\mathsf{Ring}^\uparrow$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds over finite index spines as in Definition 50.26. Let $n, m, p, q : \mathsf{Nat}^\uparrow$, and consider the $\mathsf{Mat}^\uparrow$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R, \qquad C : \mathsf{Fin}(p) \times \mathsf{Fin}(q) \to R$$

be carried matrices. Assume matrix multiplication is the finite scalar-fold operation

$$(X \cdot_{\mathrm{Mat}} Y)_{u,w} := \Sigma_{\mathsf{Fin}(s)}(v \mapsto X_{u,v} \cdot_R Y_{v,w})$$

for every matching inner dimension s, using the multiplication closure row of Definition 49.64. Assume the scalar classifier equivalence fields, multiplication congruence, multiplicative associativity, left and right distributivity, and additive laws retained in Definition 43.26. Then the two matched products are carried and

$$(A \cdot_{\mathrm{Mat}} B) \cdot_{\mathrm{Mat}} C \sim_{\mathrm{Mat}} A \cdot_{\mathrm{Mat}} (B \cdot_{\mathrm{Mat}} C).$$

Proof. The multiplication closure row of Definition 49.64 carries $A \cdot_{\mathrm{Mat}} B$ and $B \cdot_{\mathrm{Mat}} C$, and then carries the two outer products. By Definition 49.62, it remains to compare corresponding scalar entries.

Fix $i : \mathsf{Fin}(n)$ and $l : \mathsf{Fin}(q)$. For $a : \mathsf{Fin}(m)$ and $b : \mathsf{Fin}(p)$, set

$$t(a, b) := A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l}).$$

Expanding the left-associated product gives

$$((A \cdot_{\mathrm{Mat}} B) \cdot_{\mathrm{Mat}} C)_{i,l} \sim_R \Sigma_{b:\mathsf{Fin}(p)}((A \cdot_{\mathrm{Mat}} B)_{i,b} \cdot_R C_{b,l}).$$

For each fixed b, the inner product definition and the right-multiplication half of Definition 49.95 classify the summand with

$$\Sigma_{a:\mathsf{Fin}(m)}((A_{i,a} \cdot_R B_{a,b}) \cdot_R C_{b,l}).$$

Scalar multiplicative associativity from Definition 43.26 gives the pointwise comparison

$$(A_{i,a} \cdot_R B_{a,b}) \cdot_R C_{b,l} \sim_R A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l}).$$

Using Definition 50.26 for the inner and outer finite spines yields

$$((A \cdot_{\mathrm{Mat}} B) \cdot_{\mathrm{Mat}} C)_{i,l} \sim_R \Sigma_{b:\mathsf{Fin}(p)}\Sigma_{a:\mathsf{Fin}(m)} t(a, b).$$

Expanding the right-associated product gives

$$(A \cdot_{\mathrm{Mat}} (B \cdot_{\mathrm{Mat}} C))_{i,l} \sim_R \Sigma_{a:\mathsf{Fin}(m)}(A_{i,a} \cdot_R (B \cdot_{\mathrm{Mat}} C)_{a,l}).$$

For each fixed a, the inner product definition and the left-multiplication half of Definition 49.95 classify the summand with

$$\Sigma_{b:\mathsf{Fin}(p)} A_{i,a} \cdot_R (B_{a,b} \cdot_R C_{b,l}).$$

Again Definition 50.26 lifts the pointwise comparisons through the finite folds, so

$$(A \cdot_{\mathrm{Mat}} (B \cdot_{\mathrm{Mat}} C))_{i,l} \sim_R \Sigma_{a:\mathsf{Fin}(m)} \Sigma_{b:\mathsf{Fin}(p)} t(a,b).$$

Apply Definition 49.105 to the family $s(b,a) := t(a,b)$ over $\mathsf{Fin}(p) \times \mathsf{Fin}(m)$. It gives

$$\Sigma_{b:\mathsf{Fin}(p)} \Sigma_{a:\mathsf{Fin}(m)} t(a,b) \sim_R \Sigma_{a:\mathsf{Fin}(m)} \Sigma_{b:\mathsf{Fin}(p)} t(a,b).$$

Classifier transitivity composes the left expansion, this rectangular fold reindexing, and the symmetry of the right expansion. Thus every entry of $(A \cdot_{\mathrm{Mat}} B) \cdot_{\mathrm{Mat}} C$ is scalar-classified with the corresponding entry of $A \cdot_{\mathrm{Mat}} (B \cdot_{\mathrm{Mat}} C)$. The pointwise classifier of Definition 49.62 packages these entrywise comparisons as the displayed matrix classifier, and Definition 49.65 records that the dimension witnesses are unchanged. □

Theorem 49.107 (Matrix multiplication reassociation classifier transport)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, addition $+_R$, multiplication $\cdot_R$, zero endpoint 0_R, and finite additive folds over finite index spines as in Definition 50.26. Let $n, m, p, q : \mathsf{Nat}^{\uparrow}$, and consider the compatible $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A, A' : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B, B' : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R, \qquad C, C' : \mathsf{Fin}(p) \times \mathsf{Fin}(q) \to R$$

be carried matrices over the same scalar source and matching finite dimensions. Assume matrix multiplication is the finite-fold operation

$$(X \cdot Y)_{u,w} := \Sigma_V (v \mapsto X_{u,v} \cdot_R Y_{v,w})$$

for every matching inner finite index carrier V, using the multiplication closure field of Definition 49.64. Assume the scalar classifier equivalence fields, multiplication congruence, multiplicative associativity, left and right distributivity, additive laws, zero multiplication laws, the fold congruence theorem Definition 50.26, and the two-index finite fold swap certificate of Definition 49.94 required by Definition 49.96 and Definition 49.97. If

$$A \sim_{\mathrm{Mat}} A', \qquad B \sim_{\mathrm{Mat}} B', \qquad C \sim_{\mathrm{Mat}} C',$$

then the two reassociated products $(A \cdot B) \cdot C$ and $A' \cdot (B' \cdot C')$ are carried $\mathsf{Mat}^{\uparrow}$ matrices, and

$$(A \cdot B) \cdot C \sim_{\mathrm{Mat}} A' \cdot (B' \cdot C').$$

Proof. Apply Definition 49.97 to the hypotheses $A \sim_{\mathrm{Mat}} A'$ and $B \sim_{\mathrm{Mat}} B'$. This carries $A \cdot B$ and $A' \cdot B'$ and gives

$$A \cdot B \sim_{\mathrm{Mat}} A' \cdot B'.$$

Apply Definition 49.97 again to this product comparison and to $C \sim_{\mathrm{Mat}} C'$. The compatible dimensions are unchanged, so it carries the two outer left-associated products and gives

$$(A \cdot B) \cdot C \sim_{\mathrm{Mat}} (A' \cdot B') \cdot C'.$$

Now apply Definition 49.96 to the carried triple A', B', C'. The same scalar fold source and the same finite dimension ledger carry $B' \cdot C'$ and $A' \cdot (B' \cdot C')$, and the theorem supplies

$$(A' \cdot B') \cdot C' \sim_{\mathrm{Mat}} A' \cdot (B' \cdot C').$$

Matrix-classifier transitivity composes the two displayed comparisons. The conclusion follows, with all carrier witnesses obtained from the multiplication closure field and all dimension witnesses preserved by Definition 49.65. □

The ten-field fit is: traditional target, reassociation under matrix equality; BEDC source, $\mathsf{Mat}^{\uparrow} + \mathsf{Ring}^{\uparrow} + \mathsf{Nat}^{\uparrow}$; carrier, finite matrix carriers; classifier, pointwise scalar sameness; stability, multiplication classifier congruence plus finite-fold associativity; ledger, unchanged dimensions and scalar source; core theorem, reassociation classifier transport; exactness, pointwise scalar classification; standard bridge, finite-dimensional matrix algebra; status, $\mathsf{ProofTarget}$.

Definition 49.108 (Matrix transpose over a certified scalar carrier)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier and let $n, m : \mathsf{Nat}^{\uparrow}$. For a carried matrix $X : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R$ from Definition 49.1, define its transpose by*

$$(X^{\mathsf{T}})_{j,i} := X_{i,j}$$

for $j : \mathsf{Fin}(m)$ and $i : \mathsf{Fin}(n)$. The carriedness witness at the transposed index (j, i) is the scalar witness already assigned to the original entry $X_{i,j}$ by the matrix pattern of Definition 49.61.

Theorem 49.109 (Matrix transpose reverses product over the opposite ring)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\Sigma_{\mathsf{Fin}(m)}$ as in Definition 50.26. Let R^{op} be the opposite ring data of Definition 43.35, certified by Definition 43.36; thus R^{op} has the same carrier, classifier, zero endpoint, and addition as R, and multiplication*

$$x \cdot_{R^{\mathrm{op}}} y := y \cdot_R x.$$

Let $n, m, p : \mathsf{Nat}^{\uparrow}$ and let

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R$$

be carried matrices. Consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62, and write $\sim_{\mathrm{Mat},R^{\mathrm{op}}}$ for that classifier over the opposite scalar source. Assume matrix multiplication over any certified scalar ring is the finite-fold operation

$$(X \cdot_S Y)_{u,w} := \Sigma_V(v \mapsto X_{u,v} \cdot_S Y_{v,w})$$

for the matching inner finite index carrier V, using the multiplication closure field of Definition 49.64. Then $(A \cdot_R B)^{\mathsf{T}}$ and $B^{\mathsf{T}} \cdot_{R^{\mathrm{op}}} A^{\mathsf{T}}$ are carried $\mathsf{Mat}^{\uparrow}$ matrices over R^{op}, and

$$(A \cdot_R B)^{\mathsf{T}} \sim_{\mathrm{Mat},R^{\mathrm{op}}} B^{\mathsf{T}} \cdot_{R^{\mathrm{op}}} A^{\mathsf{T}}.$$

Proof. The product $A \cdot_R B$ is carried by the multiplication closure row of Definition 49.64. Since the opposite ring has the same scalar carrier and classifier by Definition 43.35, Definition 49.108 carries $(A \cdot_R B)^{\mathsf{T}}$ over R^{op}. The same transpose definition carries B^{T} and A^{T} over R^{op}, and the multiplication closure row over the certified opposite ring carries $B^{\mathsf{T}} \cdot_{R^{\mathrm{op}}} A^{\mathsf{T}}$.

It remains to compare entries. Fix $k : \mathsf{Fin}(p)$ and $i : \mathsf{Fin}(n)$. Expanding the left endpoint gives

$$((A \cdot_R B)^{\mathsf{T}})_{k,i} = (A \cdot_R B)_{i,k} \sim_R \Sigma_{\mathsf{Fin}(m)}(j \mapsto A_{i,j} \cdot_R B_{j,k}).$$

Expanding the product over the opposite ring gives

$$(B^{\mathsf{T}} \cdot_{R^{\mathrm{op}}} A^{\mathsf{T}})_{k,i} \sim_R \Sigma_{\mathsf{Fin}(m)}(j \mapsto (B^{\mathsf{T}})_{k,j} \cdot_{R^{\mathrm{op}}} (A^{\mathsf{T}})_{j,i}).$$

By transpose readback and opposite multiplication, this last finite spine is the same as

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto B_{j,k} \cdot_{R^{\mathrm{op}}} A_{i,j}) = \Sigma_{\mathsf{Fin}(m)}(j \mapsto A_{i,j} \cdot_R B_{j,k}).$$

The two displayed expansions end at the same finite scalar spine. The finite additive fold over R^{op} uses the same zero endpoint and the same addition as the fold over R, so the pointwise reflexive scalar classifier witness, lifted through Definition 50.26, compares the folded endpoints. Classifier transitivity composes the left expansion with this fold comparison and the symmetry of the right expansion.

Thus for every k and i,

$$((A \cdot_R B)^{\mathsf{T}})_{k,i} \sim_R (B^{\mathsf{T}} \cdot_{R^{\mathrm{op}}} A^{\mathsf{T}})_{k,i}.$$

The pointwise matrix classifier of Definition 49.62 packages these scalar comparisons as the displayed $\mathsf{Mat}^{\uparrow}$ classification over R^{op}. The row-column swap and the unchanged scalar source are recorded by the dimension and scalar ledger of Definition 49.65. □

Theorem 49.110 (Matrix transpose preserves matrix addition)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$ and addition $+_R$. Let $n, m : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. For carried matrices*

$$A, B : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R,$$

assume matrix addition on each matching $\mathsf{Mat}^{\uparrow}$ carrier is the pointwise operation

$$(X + Y)_{u,v} := X_{u,v} +_R Y_{u,v}$$

for matching indices u, v, using the addition closure row of Definition 49.64. Then $(A + B)^{\mathsf{T}}$ and $A^{\mathsf{T}} + B^{\mathsf{T}}$ are carried $\mathsf{Mat}^{\uparrow}$ matrices over $\mathsf{Fin}(m) \times \mathsf{Fin}(n)$, and

$$(A + B)^{\mathsf{T}} \sim_{\mathrm{Mat}} A^{\mathsf{T}} + B^{\mathsf{T}}.$$

Proof. The addition closure row of Definition 49.64 carries $A + B$ over $\mathsf{Fin}(n) \times \mathsf{Fin}(m)$. Applying Definition 49.108 carries $(A + B)^{\mathsf{T}}$ over $\mathsf{Fin}(m) \times \mathsf{Fin}(n)$: at the transposed index (j, i) its scalar witness is the carriedness witness for $(A+B)_{i,j}$ supplied by the matrix pattern of Definition 49.61.

The same transpose definition carries A^{T} and B^{T} over $\mathsf{Fin}(m) \times \mathsf{Fin}(n)$. Applying the pointwise addition closure row in that swapped matrix carrier carries $A^{\mathsf{T}} + B^{\mathsf{T}}$. The ledger records the same scalar source and the exchanged finite index witnesses, as in Definition 49.65.

It remains to compare entries. Fix $j : \mathsf{Fin}(m)$ and $i : \mathsf{Fin}(n)$. Expanding the transpose and pointwise addition clauses gives

$$((A + B)^{\mathsf{T}})_{j,i} = (A + B)_{i,j} = A_{i,j} +_R B_{i,j}.$$

Expanding the transposed pointwise sum gives

$$(A^{\mathsf{T}} + B^{\mathsf{T}})_{j,i} = (A^{\mathsf{T}})_{j,i} +_R (B^{\mathsf{T}})_{j,i} = A_{i,j} +_R B_{i,j}.$$

The two displayed endpoints are the same carried scalar sum, so scalar classifier reflexivity gives

$$((A + B)^{\mathsf{T}})_{j,i} \sim_R (A^{\mathsf{T}} + B^{\mathsf{T}})_{j,i}.$$

Since j and i were arbitrary, the pointwise classifier of Definition 49.62 packages these scalar comparisons as

$$(A + B)^{\mathsf{T}} \sim_{\mathrm{Mat}} A^{\mathsf{T}} + B^{\mathsf{T}}.$$

□

Theorem 49.111 (Matrix transpose fixes the finite-fold identity matrix)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, and unit endpoint 1_R. Let $n : \mathsf{Nat}^{\uparrow}$, and let I_n be the finite-fold identity matrix of Definition 49.92. Consider the square $\mathsf{Mat}^{\uparrow}$ carrier over $\mathsf{Fin}(n) \times \mathsf{Fin}(n)$ from Definition 49.1, with pointwise classifier from Definition 49.62. Then I_n^{T} is carried and*

$$I_n^{\mathsf{T}} \sim_{\mathrm{Mat}} I_n.$$

Proof. The identity datum of Definition 49.92 carries I_n pointwise: each entry is the scalar endpoint 1_R on the diagonal and 0_R off the diagonal. Applying Definition 49.108 to this carried square matrix carries I_n^{T} over the same finite row and column carrier, since the transposed index pair again lies in $\mathsf{Fin}(n) \times \mathsf{Fin}(n)$.

It remains to compare entries. Fix $i, j : \mathsf{Fin}(n)$. By transpose readback,

$$(I_n^{\mathsf{T}})_{i,j} = (I_n)_{j,i}.$$

By the identity-matrix definition, $(I_n)_{j,i}$ is classified with the Kronecker endpoint $\delta_{j,i}$ and $(I_n)_{i,j}$ is classified with $\delta_{i,j}$. The finite-index equality certificate is symmetric: if j and i are identified, then i and j are identified as well, so both Kronecker endpoints are 1_R; if they are not identified, both Kronecker endpoints are 0_R. Scalar classifier reflexivity for the carried endpoints 1_R and 0_R, followed by the scalar classifier symmetry and transitivity fields, therefore gives

$$(I_n)_{j,i} \sim_R (I_n)_{i,j}.$$

Combining this comparison with the transpose readback yields

$$(I_n^{\mathsf{T}})_{i,j} \sim_R (I_n)_{i,j}.$$

Since the indices i and j were arbitrary, the pointwise matrix classifier packages these scalar comparisons as

$$I_n^{\mathsf{T}} \sim_{\mathrm{Mat}} I_n.$$

□

Theorem 49.112 (Two-index fold swap from finite permutation invariance)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold $\mathsf{fold}_{+,R}$ from Definition 50.26. Assume the scalar classifier equivalence fields and the additive associativity, commutativity, zero, and congruence rows retained in Definition 43.26. Let J and K be finite index carriers with chosen finite enumerating spines, and write Σ_J and Σ_K for the corresponding finite folds of scalar families over those spines. Suppose the fold is finite-permutation invariant: whenever finite carried scalar spines xs and ys are related by a bijection of their finite position spines and every matched pair of entries is related by $\sim_R$, then*

$$\mathsf{fold}_{+,R}(xs) \sim_R \mathsf{fold}_{+,R}(ys).$$

For every pointwise carried scalar family $s : J \times K \to R$ and every pointwise carried transpose family $s^{\tau} : K \times J \to R$ satisfying

$$s^{\tau}(k, j) \sim_R s(j, k)$$

for all $j : J$ and $k : K$, the two iterated folds satisfy

$$\Sigma_J(j \mapsto \Sigma_K(k \mapsto s(j,k))) \sim_R \Sigma_K(k \mapsto \Sigma_J(j \mapsto s^\tau(k,j))).$$

Thus the field specified in Definition 49.94 is derivable from the finite-permutation invariant additive fold.

Proof. For a family $u : J \times K \to R$, let $\mathsf{Row}_{J,K}(u)$ be the finite scalar spine obtained by concatenating, in the chosen J-order, the chosen K-spine $k \mapsto u(j,k)$ for each row j. For a family $v : K \times J \to R$, let $\mathsf{Col}_{K,J}(v)$ be the analogous K-major spine.

First compare the iterated fold with the flattened row spine. The finite-fold append law used here is obtained by induction on a finite spine xs:

$$\mathsf{fold}_{+,R}(xs{+}{+}ys) \sim_R \mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys).$$

In the nil case this is the additive zero law after unfolding Definition 50.26. In the cons case, unfold the fold on the cons spine, apply the induction hypothesis inside the right summand by additive congruence, and rebracket by additive associativity. Applying this append law while inducting on the finite spine enumerating J gives

$$\Sigma_J(j \mapsto \Sigma_K(k \mapsto s(j,k))) \sim_R \mathsf{fold}_{+,R}(\mathsf{Row}_{J,K}(s)).$$

The same argument with the two index carriers interchanged, followed by classifier symmetry, gives

$$\mathsf{fold}_{+,R}(\mathsf{Col}_{K,J}(s^\tau)) \sim_R \Sigma_K(k \mapsto \Sigma_J(j \mapsto s^\tau(k,j))).$$

The transposition map sends the occurrence indexed by (j,k) in $\mathsf{Row}_{J,K}(s)$ to the occurrence indexed by (k,j) in $\mathsf{Col}_{K,J}(s^\tau)$. Since the chosen spines enumerate J and K without loss or duplication, this map is a bijection of the two finite position spines. At matched positions the hypothesis gives $s^\tau(k,j) \sim_R s(j,k)$, and classifier symmetry gives the orientation $s(j,k) \sim_R s^\tau(k,j)$. Finite-permutation invariance therefore yields

$$\mathsf{fold}_{+,R}(\mathsf{Row}_{J,K}(s)) \sim_R \mathsf{fold}_{+,R}(\mathsf{Col}_{K,J}(s^\tau)).$$

Classifier transitivity composes the row-flattening comparison, the permutation-invariance comparison, and the column-unflattening comparison, giving the displayed two-index fold swap. □

Definition 49.113 (Finite-fold zero matrix data)**.** *Let R be a scalar $\mathsf{Ring}^\uparrow$ carrier with classifier $\sim_R$ and zero endpoint 0_R. For $n, m : \mathsf{Nat}^\uparrow$, the carried zero matrix datum over $\mathsf{Fin}(n) \times \mathsf{Fin}(m)$ is the table*

$$Z_{n,m} : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad (Z_{n,m})_{i,j} := 0_R.$$

It is carried by the $\mathsf{Mat}^\uparrow$ carrier of Definition 49.1 exactly when every displayed entry is carried by the scalar source, and its classifier endpoint is the pointwise scalar-zero table specified by Definition 49.62.

Theorem 49.114 (Finite-fold matrix zero multiplication law)**.** *Let R be a scalar $\mathsf{Ring}^\uparrow$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive folds over finite index spines as in Definition 49.80. Let $n, m, p : \mathsf{Nat}^\uparrow$, and consider the $\mathsf{Mat}^\uparrow$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let*

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R, \qquad B : \mathsf{Fin}(m) \times \mathsf{Fin}(p) \to R$$

be carried matrices. Assume matrix multiplication is the finite scalar-fold operation

$$(X \cdot_{\mathrm{Mat}} Y)_{u,w} := \Sigma_{\mathsf{Fin}(s)}(v \mapsto X_{u,v} \cdot_R Y_{v,w})$$

for every matching inner dimension s, using the multiplication closure row of Definition 49.64. Assume the scalar classifier equivalence fields, multiplication closure and congruence, additive closure and congruence, and the left and right zero multiplication laws retained in Definition 43.26. Then $Z_{n,m} \cdot_{\mathrm{Mat}} B$ and $A \cdot_{\mathrm{Mat}} Z_{m,p}$ are carried $\mathsf{Mat}^{\uparrow}$ matrices over $\mathsf{Fin}(n) \times \mathsf{Fin}(p)$, and

$$Z_{n,m} \cdot_{\mathrm{Mat}} B \sim_{\mathrm{Mat}} Z_{n,p} \qquad \textit{and} \qquad A \cdot_{\mathrm{Mat}} Z_{m,p} \sim_{\mathrm{Mat}} Z_{n,p}.$$

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixFiniteFoldZeroMatrix multiplication law`

Proof. The zero matrix data of Definition 49.113 carries $Z_{n,m}$ and $Z_{m,p}$ pointwise from the scalar zero endpoint. The multiplication closure row of Definition 49.64 therefore carries the two products $Z_{n,m} \cdot_{\mathrm{Mat}} B$ and $A \cdot_{\mathrm{Mat}} Z_{m,p}$.

By Definition 49.62, it remains to compare entries with the pointwise zero table $Z_{n,p}$. Fix $i : \mathsf{Fin}(n)$ and $k : \mathsf{Fin}(p)$. The left product entry expands to

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto (Z_{n,m})_{i,j} \cdot_R B_{j,k}).$$

Each summand is scalar-classified with 0_R by the displayed definition of $Z_{n,m}$ and the left zero multiplication law in Definition 43.26.

Induct on the finite spine enumerating $\mathsf{Fin}(m)$. In the nil case, Definition 49.80 identifies the fold with 0_R. In the cons case, the head summand is classified with 0_R, and the induction hypothesis classifies the tail fold with 0_R. Additive congruence transports these comparisons to the cons fold, and the additive zero law retained in Definition 43.26 classifies the resulting endpoint with 0_R.

The right product entry expands to

$$\Sigma_{\mathsf{Fin}(m)}(j \mapsto A_{i,j} \cdot_R (Z_{m,p})_{j,k}).$$

Now each summand is scalar-classified with 0_R by the displayed definition of $Z_{m,p}$ and the right zero multiplication law. The same finite-spine induction reduces this fold to 0_R.

Since i and k were arbitrary, Definition 49.62 packages the two entrywise scalar-zero comparisons as matrix classifier witnesses against $Z_{n,p}$. The row, contraction, column, and scalar-source witnesses are unchanged, and Definition 49.65 records that the only non-structural provenance used by the endpoint comparison is scalar zero multiplication. □

This certificate fits the standard ten-field reading as follows. The traditional target is zero absorption for finite matrix multiplication. The BEDC source is $\mathsf{Mat}^{\uparrow}$ over $\mathsf{Ring}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$. The carrier is the finite table carrier on $\mathsf{Fin}(n) \times \mathsf{Fin}(m)$. The classifier is the pointwise scalar classifier. Stability consists of matrix multiplication closure together with scalar zero multiplication and additive fold closure. The ledger keeps the row, inner, column, and scalar-source witnesses fixed and records scalar-zero provenance. The core theorem is Definition 49.114. Exactness is the pointwise zero endpoint $Z_{n,p}$. The standard bridge is the usual finite matrix interpretation of carried tables and finite scalar folds. The formalization status is ProofTarget.

Theorem 49.115 (Finite-fold zero matrix additive identity)**.** *Let R carry a scalar $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, and addition $+_R$. Assume the scalar source carries 0_R, and assume the scalar classifier equivalence fields, addition closure, and scalar additive left-zero*

and right-zero laws retained in Definition 43.26. Let $n, m : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carrier of Definition 49.1 with pointwise classifier from Definition 49.62. Let

$$A : \mathsf{Fin}(n) \times \mathsf{Fin}(m) \to R$$

be a carried matrix, and let $Z_{n,m}$ be the zero matrix datum of Definition 49.113. Assume matrix addition is the pointwise operation

$$(X + Y)_{i,j} := X_{i,j} +_R Y_{i,j}$$

using the addition closure row of Definition 49.64. Then $A + Z_{n,m}$ and $Z_{n,m} + A$ are carried $\mathsf{Mat}^{\uparrow}$ matrices over $\mathsf{Fin}(n) \times \mathsf{Fin}(m)$, and

$$A + Z_{n,m} \sim_{\mathrm{Mat}} A \qquad \textit{and} \qquad Z_{n,m} + A \sim_{\mathrm{Mat}} A.$$

Consequently,

$$A + Z_{n,m} \sim_{\mathrm{Mat}} Z_{n,m} + A.$$

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonZero additive identity`

Proof. The zero matrix datum of Definition 49.113 carries $Z_{n,m}$ pointwise from the carried scalar zero endpoint. Since A is carried, the addition closure row of Definition 49.64 carries both pointwise sums $A + Z_{n,m}$ and $Z_{n,m} + A$.

By Definition 49.62, it remains to compare entries. Fix $i : \mathsf{Fin}(n)$ and $j : \mathsf{Fin}(m)$. The pointwise addition rule and the definition of $Z_{n,m}$ give

$$(A + Z_{n,m})_{i,j} = A_{i,j} +_R 0_R.$$

The scalar additive right-zero law retained in Definition 43.26 classifies this endpoint with $A_{i,j}$, so

$$(A + Z_{n,m})_{i,j} \sim_R A_{i,j}.$$

For the other order, the same pointwise rule gives

$$(Z_{n,m} + A)_{i,j} = 0_R +_R A_{i,j}.$$

The scalar additive left-zero law classifies this endpoint with $A_{i,j}$, hence

$$(Z_{n,m} + A)_{i,j} \sim_R A_{i,j}.$$

Since i and j were arbitrary, Definition 49.62 packages the two entrywise scalar comparisons as the displayed matrix classifier witnesses against A. Applying scalar symmetry to the second entrywise comparison and scalar transitivity with the first gives

$$(A + Z_{n,m})_{i,j} \sim_R (Z_{n,m} + A)_{i,j}$$

for every index pair, so the same pointwise classifier packages the final displayed consequence. The dimensions, row witness, column witness, and scalar source are unchanged, as recorded by Definition 49.65. □

Lean4 ✓ `BEDC.Derived.MatrixUp.MatrixSingletonAddFold zero prefix absorption`

Theorem 49.116 (Zero matrix transpose swaps dimensions)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$ and zero endpoint 0_R. Assume the scalar source carries 0_R and that the scalar classifier reflexivity row retained in Definition 43.26 is available. Let $n, m : \mathsf{Nat}^{\uparrow}$, and consider the $\mathsf{Mat}^{\uparrow}$ carriers of Definition 49.1 with pointwise classifier from Definition 49.62. Let $Z_{n,m}$ and $Z_{m,n}$ be the zero matrix data of Definition 49.113. Then $(Z_{n,m}^{\mathsf{T}})$ is carried as an $m \times n$ $\mathsf{Mat}^{\uparrow}$ matrix, and*

$$Z_{n,m}^{\mathsf{T}} \sim_{\mathrm{Mat}} Z_{m,n}.$$

Proof. The zero matrix datum $Z_{n,m}$ is carried pointwise from the carried scalar endpoint 0_R by Definition 49.113. Applying the transpose carrier clause of Definition 49.108 carries $Z_{n,m}^{\mathsf{T}}$ over $\mathsf{Fin}(m) \times \mathsf{Fin}(n)$: at an index (j, i) its scalar witness is the witness already assigned to $(Z_{n,m})_{i,j}$.

It remains to compare entries with the zero matrix of the swapped shape. Fix $j : \mathsf{Fin}(m)$ and $i : \mathsf{Fin}(n)$. Expanding the transpose definition gives

$$(Z_{n,m}^{\mathsf{T}})_{j,i} = (Z_{n,m})_{i,j}.$$

Expanding the zero matrix data at the original and swapped index pairs gives

$$(Z_{n,m})_{i,j} = 0_R \qquad \text{and} \qquad (Z_{m,n})_{j,i} = 0_R.$$

Thus both scalar endpoints at the displayed matrix position are the same carried zero endpoint. Scalar classifier reflexivity gives

$$(Z_{n,m}^{\mathsf{T}})_{j,i} \sim_R (Z_{m,n})_{j,i}.$$

Since j and i were arbitrary, the pointwise matrix classifier of Definition 49.62 packages these scalar comparisons as

$$Z_{n,m}^{\mathsf{T}} \sim_{\mathrm{Mat}} Z_{m,n}.$$

The dimension ledger is exactly the row-column swap recorded by Definition 49.65, and the scalar source is unchanged. □

Chapter 50

A Concrete Naming Certificate for $\mathsf{Poly}^{\uparrow}$

The polynomial interface $\mathsf{Poly}^{\uparrow}$ has certificate obligations over a scalar ring from chapter 43 as finite coefficient sequences from chapter 36, normalized to remove trailing zeros so that the highest-index entry is nonzero (or the sequence is empty for the zero polynomial). The classifier is structural sameness on the normalized representative. Polynomial addition and multiplication are derived operations governed by the ring's certificate fields.

50.1 Concrete singleton zero-polynomial instance

Definition 50.1 (Singleton zero-polynomial instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Use the singleton empty-history scalar ring of Definition 43.5. Define the polynomial classifier*

$$\rho_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

The normalized coefficient-list representative is e, and the singleton normalization map on admitted endpoints is $\mathsf{norm}_{\mathsf{e}}(h) := \mathsf{e}$. The polynomial carrier is A_{e} on BHist endpoints. Define the displayed polynomial operations by

$$0_P := \mathsf{e}, \qquad 1_P := \mathsf{e}, \qquad P \oplus_P Q := \mathsf{append}(P, Q), \qquad P \odot_P Q := \mathsf{append}(P, Q).$$

This tuple is the concrete zero-polynomial instance of the $\mathsf{Poly}^{\uparrow}$ certificate schema over the singleton BEDC history carrier.

Theorem 50.2 (Singleton zero-polynomial laws)**.** *For the singleton instance of Definition 50.1, the displayed polynomial carrier and operations satisfy the concrete $\mathsf{Poly}^{\uparrow}$ law package:*

1. *0_P and 1_P are carried by A_{e}.*
2. *If $A_{\mathsf{e}}(P)$ and $A_{\mathsf{e}}(Q)$, then $A_{\mathsf{e}}(P \oplus_P Q)$ and $A_{\mathsf{e}}(P \odot_P Q)$.*
3. *0_P is a left and right unit for $\oplus_P$ under ρ_{e}.*
4. *1_P is a left and right identity for $\odot_P$ under ρ_{e}.*
5. *$\oplus_P$ and $\odot_P$ are associative under ρ_{e}.*
6. *Left and right distributivity of $\odot_P$ over $\oplus_P$ hold under ρ_{e}.*

7. $\mathsf{norm_e}(\mathsf{norm_e}(P))$ *is classified with* $\mathsf{norm_e}(P)$ *by* ρ_{e}.

The $\mathsf{Poly}^{\uparrow}$ *schema is represented by this concrete* BHist *carrier and its displayed operations, so Lean formalization can target* BEDC.Derived.PolynomialUp *through a carrier certificate instead of abstract operator-only statements.*

Proof. The carrier endpoint is the singleton source of Definition 36.169 at $a_0 = \mathsf{e}$. Thus 0_P, 1_P, and the normalized representative e are all carried by the same reflexive endpoint witness.

For closure under $\oplus_P$ and $\odot_P$, carried inputs reduce to the singleton endpoint under ρ_{e}. The displayed operations are BEDC append endpoints. The continuation unit and associativity laws used in Theorem 40.3, together with Theorem 43.6, identify those append endpoints with the singleton class of e.

The additive unit laws and multiplicative identity laws are the same endpoint calculation. On the left and on the right, the comparison is between $\mathsf{append}(\mathsf{e}, P)$ or $\mathsf{append}(P, \mathsf{e})$ and P. The append unit laws provide the hsame component, while the carrier witnesses come from $A_{\mathsf{e}}(P)$ and $A_{\mathsf{e}}(\mathsf{e})$.

Associativity for both operations follows from BEDC append associativity. The two parenthesized endpoints are related by the same continuation spine, and the singleton classifier packages that relation with the already established carrier closure.

For distributivity, every term appearing on either side is a carried singleton endpoint after the closure step. Both sides are therefore classified with e, and transitivity of hsame supplies the displayed ρ_{e} relation.

Finally, normalization idempotence is immediate because $\mathsf{norm_e}$ is the constant endpoint e on admitted inputs. Both iterated and one-step normalization have the same carried endpoint, so reflexivity of hsame gives the classifier witness. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialUp StdBridge`

50.2 Carrier definition

Definition 50.3 (Polynomial carrier)**.** *Use finite coefficient lists over a scalar ring, normalized by removing trailing zero coefficients.*

50.3 Concrete singleton history polynomial instance

Definition 50.4 (Concrete singleton history polynomial instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$. *Define*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Set the polynomial code carrier to A_{e} *on* BHist. *Define the displayed polynomial operations by*

$$\mathsf{normalize}_P(h) := \mathsf{e}, \qquad 0_P := \mathsf{e}, \qquad 1_P := \mathsf{e}, \qquad \mathsf{add}_P(f, g) := \mathsf{append}(f, g), \qquad \mathsf{mul}_P(f, g) := \mathsf{append}(f, g).$$

The ledger is the empty normalized coefficient list over the singleton empty-history ring of Definition 43.5.

Theorem 50.5 (Singleton history polynomial laws)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \mathsf{normalize}_P, 0_P, 1_P, \mathsf{add}_P, \mathsf{mul}_P)$$

satisfies the $\mathsf{Poly}^{\uparrow}$ *law package over* BHist*: normalization closure, normalization idempotence, zero and one closure, addition and multiplication closure, additive zero laws, multiplicative zero laws, multiplicative one laws, associativity, distributivity, and congruence under* ρ_{e}*.*

Proof. Definition 36.169 at $a_0 = \mathsf{e}$ supplies the singleton source witness $A_{\mathsf{e}}(\mathsf{e})$. The same endpoint is the scalar zero and scalar unit carrier supplied by Theorem 43.6.

Normalization closure holds because $\mathsf{normalize}_P(h)$ is the endpoint e. Normalization idempotence compares $\mathsf{normalize}_P(\mathsf{normalize}_P(h))$ with $\mathsf{normalize}_P(h)$, and both sides are the same carried endpoint.

The constants 0_P and 1_P are both e, so zero closure and one closure are exactly $A_{\mathsf{e}}(\mathsf{e})$. Addition closure and multiplication closure use empty-result inversion for BEDC append: if both inputs are singleton-carried, then both inputs are e, and the appended result is again singleton-carried.

The additive zero laws, multiplicative zero laws, and multiplicative one laws all reduce to a comparison of an appended empty endpoint with the carried endpoint. Each classifier goal is assembled from the two endpoint witnesses and the corresponding history-sameness witness.

Associativity and distributivity have no remaining endpoint computation after the displayed operations are unfolded under singleton-carrier hypotheses. The left-associated, right-associated, left-distributive, and right-distributive terms are all singleton-carried, so ρ_{e} packages the same singleton carrier witnesses with the BEDC append associativity witness.

For congruence under ρ_{e}, assume the inputs are pairwise related by ρ_{e}. The operation outputs are singleton-carried on both sides for normalization, addition, and multiplication. The output relation is the singleton classifier witness built from the carrier witnesses and history-sameness. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.singleton empty history polynomial laws`

Theorem 50.6 (Polynomial singleton classifier empty addition split)**.** *For singleton-history polynomial codes* P, Q, h*, the classifier relation between* $\mathsf{add}_P(P, Q)$ *and* h *holds exactly when* P *and* Q *are each history-same to* e *and* h *is singleton-carried:*

$$\rho_{\mathsf{e}}(\mathsf{add}_P(P, Q), h) \quad \Longleftrightarrow \quad \mathsf{hsame}(P, \mathsf{e}) \wedge \mathsf{hsame}(Q, \mathsf{e}) \wedge A_{\mathsf{e}}(h).$$

Proof. Unfold the singleton classifier. The left carrier field says that $\mathsf{append}(P, Q)$ is history-same to e. Empty-result inversion for BEDC append splits this into $\mathsf{hsame}(P, \mathsf{e})$ and $\mathsf{hsame}(Q, \mathsf{e})$, while the right carrier field is exactly $A_{\mathsf{e}}(h)$. Conversely, after the two endpoint sameness witnesses identify P and Q with e, the classifier is rebuilt from the singleton carrier of e, the given carrier of h, and symmetry of the carrier witness for h. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier add split empty iff`

Theorem 50.7 (Polynomial singleton add visible-left exclusion)**.** *For histories* $P, Q, h : \mathsf{BHist}$*, no singleton polynomial classifier witness can classify an addition endpoint whose left source begins with a visible constructor:*

$$\rho_{\mathsf{e}}(\mathsf{add}_P(\mathsf{e0}(P), Q), h) \Rightarrow \bot, \qquad \rho_{\mathsf{e}}(\mathsf{add}_P(\mathsf{e1}(P), Q), h) \Rightarrow \bot.$$

Proof. Unfold add_P to BEDC append and unfold the singleton classifier. The source carrier field says the appended endpoint is history-same to e. Empty-result inversion for append then forces the visible left source itself to be e, contradicting constructor separation for $\mathsf{e0}$ or $\mathsf{e1}$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier add visible left absurd`

Theorem 50.8 (Polynomial singleton add visible-right exclusion)**.** *For histories $P, Q, h : \mathsf{BHist}$, no singleton polynomial classifier witness can classify an addition endpoint whose right source begins with a visible constructor:*

$$\rho_{\mathsf{e}}(\mathsf{add}_P(P, \mathsf{e0}(Q)), h) \Rightarrow \bot, \qquad \rho_{\mathsf{e}}(\mathsf{add}_P(P, \mathsf{e1}(Q)), h) \Rightarrow \bot.$$

Proof. Again the classifier source carrier identifies the BEDC append endpoint with e. Empty-result inversion for append separates the left and right source endpoints, so the visible right endpoint would have to be e. This is excluded by the disjointness of visible BHist constructors from the empty history. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier add visible right absurd`

Theorem 50.9 (Polynomial singleton classifier append-left cancellation)**.** *For singleton-carried P, adding the same BEDC append context on the left of two singleton polynomial endpoints preserves and reflects the singleton classifier:*

$$A_{\mathsf{e}}(P) \Longrightarrow (\rho_{\mathsf{e}}(\mathsf{append}(P, Q), \mathsf{append}(P, R)) \Longleftrightarrow \rho_{\mathsf{e}}(Q, R)).$$

Proof. In the forward direction, the classifier says that both appended endpoints are singleton-carried. Empty append inversion splits the two endpoint carriers, giving $Q\mathsf{hsamee}$ and $R\mathsf{hsamee}$, and these witnesses rebuild $\rho_{\mathsf{e}}(Q, R)$. Conversely, the singleton carrier of P identifies the left context with e, so both appended endpoints reduce to the original classified endpoints Q and R. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier append left cancel iff`

Theorem 50.10 (Polynomial singleton classifier append-right cancellation)**.** *For singleton-carried P, adding the same BEDC append context on the right of two singleton polynomial endpoints preserves and reflects the singleton classifier:*

$$A_{\mathsf{e}}(P) \Longrightarrow (\rho_{\mathsf{e}}(\mathsf{append}(Q, P), \mathsf{append}(R, P)) \Longleftrightarrow \rho_{\mathsf{e}}(Q, R)).$$

Proof. In the forward direction, the two appended endpoints are singleton-carried. Empty append inversion splits those carrier witnesses into $Q\mathsf{hsamee}$, $P\mathsf{hsamee}$, and $R\mathsf{hsamee}$, so the recovered witnesses for Q and R rebuild $\rho_{\mathsf{e}}(Q, R)$.

For the reverse direction, combine the supplied singleton carrier for P with the two endpoint carriers in $\rho_{\mathsf{e}}(Q, R)$. Empty append inversion rebuilds singleton carrier witnesses for both right-appended endpoints, and transitivity through e supplies their classifier comparison. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier append right cancel iff`

Theorem 50.11 (Polynomial singleton classifier append-pair carrier exactness)**.** *For histories $P, Q, R, S : \mathsf{BHist}$, the singleton polynomial classifier relates the two appended endpoints exactly when all four factors are carried by the singleton carrier:*

$$\rho_{\mathsf{e}}(\mathsf{append}(P, Q), \mathsf{append}(R, S)) \Longleftrightarrow A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge A_{\mathsf{e}}(R) \wedge A_{\mathsf{e}}(S).$$

Proof. Unfold the singleton classifier. In the forward direction, the two carrier fields identify $\mathsf{append}(P, Q)$ and $\mathsf{append}(R, S)$ with e, and empty-result inversion for append splits these into the four singleton carrier witnesses. Conversely, if all four factors are singleton-carried, each factor is identified with e, so both appended endpoints reduce to e and the classifier is rebuilt from reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier append pair carrier iff`

Theorem 50.12 (Polynomial singleton classifier self-context split)**.** *For histories a, x, b : BHist, the singleton polynomial classifier splits the repeated self-context around x into the carrier of that context and the classifier of the middle endpoint:*

$$\rho_{\mathsf{e}}(\mathsf{append}(a, \mathsf{append}(x, a)), b) \iff A_{\mathsf{e}}(a) \wedge \rho_{\mathsf{e}}(x, b).$$

Proof. Unfold the singleton classifier. The forward direction applies empty-result inversion to the outer append carrier and then to the inner append carrier, extracting $A_{\mathsf{e}}(a)$ and $A_{\mathsf{e}}(x)$. Together with the target carrier field, these witnesses rebuild $\rho_{\mathsf{e}}(x, b)$ by transitivity through e.

Conversely, use $A_{\mathsf{e}}(a)$ and the source carrier in $\rho_{\mathsf{e}}(x, b)$ to rebuild the inner append as empty, then rebuild the full self-context endpoint as empty. The target carrier and empty-history comparison come from the middle classifier data. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier append self context split iff`

Theorem 50.13 (Polynomial singleton normalized append classifier carrier)**.** *For histories P, Q, h, classifying h against the normalized singleton append endpoint is exactly the singleton carrier predicate on h:*

$$\rho_{\mathsf{e}}(\mathsf{normalize}_P(\mathsf{append}(P, Q)), h) \quad \iff \quad A_{\mathsf{e}}(h).$$

Proof. The singleton normalization endpoint is e. In the forward direction, the right carrier field of ρ_{e} is precisely $A_{\mathsf{e}}(h)$. Conversely, a carrier witness for h combines with reflexivity of e and symmetry of the same carrier witness to rebuild the singleton classifier. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonNormalize append classifier carrier iff`

Theorem 50.14 (Polynomial singleton append distributivity is classified)**.** *For singleton-carried polynomial endpoints P, Q, R, the displayed append forms of the two distributivity comparisons are classified by ρ_{e}:*

$$\rho_{\mathsf{e}}(\mathsf{append}(P, \mathsf{append}(Q, R)), \mathsf{append}(\mathsf{append}(P, Q), \mathsf{append}(P, R)))$$

and

$$\rho_{\mathsf{e}}(\mathsf{append}(\mathsf{append}(P, Q), R), \mathsf{append}(\mathsf{append}(P, R), \mathsf{append}(Q, R))).$$

Proof. The singleton carrier is closed under BEDC append: once an input is identified with e, appending it to another singleton-carried endpoint again produces a singleton-carried endpoint. Applying this closure to all displayed append trees shows that both sides of each comparison are carried by A_{e}. Each classifier is then assembled by transitivity through e: the left append tree is history-same to e, and the right append tree is history-same to e. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingleton append distrib classified`

Theorem 50.15 (Polynomial singleton classifier continuation commutativity closure)**.** *Let P, Q : BHist be carried by the singleton polynomial carrier A_{e}. If $\mathsf{Cont}(P, Q, L)$ and $\mathsf{Cont}(Q, P, R)$, then both continuation results are carried by A_{e}, and the singleton polynomial classifier relates them:*

$$A_{\mathsf{e}}(L) \wedge A_{\mathsf{e}}(R) \wedge \rho_{\mathsf{e}}(L, R).$$

Proof. The carrier hypotheses reduce P and Q to the empty history. The two continuation witnesses then reduce L and R to the same endpoint $\mathsf{append}(\mathsf{e},\mathsf{e}) = \mathsf{e}$. Repacking the reflexive empty-history carrier witnesses for the two results with $\mathsf{hsame}(\mathsf{e},\mathsf{e})$ gives the singleton classifier comparison. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier continuation comm closed`

Theorem 50.16 (Polynomial singleton classifier continuation closure)**.** *Let P, P', Q, Q', L, R : BHist. If the singleton polynomial classifier relates P with P' and Q with Q', and if*

$$\mathsf{Cont}(P, Q, L) \qquad \textit{and} \qquad \mathsf{Cont}(P', Q', R),$$

then the singleton polynomial classifier relates the continuation results:

$$\rho_{\mathsf{e}}(L, R).$$

Proof. Each classifier witness supplies singleton carrier evidence for its two endpoints. Thus all four continuation inputs are internally the empty history. Transporting the two continuation witnesses to the canonical empty-empty continuation shows $L\mathsf{hsamee}$ and $R\mathsf{hsamee}$. These carrier witnesses, together with transitivity through e, rebuild $\rho_{\mathsf{e}}(L, R)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier continuation closed`

Theorem 50.17 (Polynomial singleton continuation result against empty is classified)**.** *Let P, Q, r : BHist be singleton-carried polynomial endpoints. If $\mathsf{Cont}(P, Q, r)$, then the continuation result is classified against the empty singleton endpoint:*

$$\rho_{\mathsf{e}}(r, \mathsf{e}).$$

Proof. The two singleton carrier witnesses identify P and Q with e. The continuation ledger then identifies r with $\mathsf{append}(\mathsf{e},\mathsf{e}) = \mathsf{e}$. The classifier is rebuilt from the empty-history carrier witnesses on both endpoints and reflexivity of $\mathsf{hsame}(\mathsf{e},\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier cont result empty classified`

Theorem 50.18 (Polynomial singleton classifier continuation result empty witness iff)**.** *For a history r : BHist, r is classified with the singleton empty polynomial endpoint exactly when it is the continuation result of two singleton-carried polynomial endpoints:*

$$\exists P, Q : \mathsf{BHist},\ A_{\mathsf{e}}(P) \wedge A_{\mathsf{e}}(Q) \wedge \mathsf{Cont}(P, Q, r) \quad \Longleftrightarrow \quad \rho_{\mathsf{e}}(r, \mathsf{e}).$$

Proof. In the forward direction, the two carrier witnesses identify P and Q with e. The continuation witness then identifies r with $\mathsf{append}(\mathsf{e},\mathsf{e})$, hence with e, and the singleton classifier is assembled from the two empty endpoint carrier witnesses and reflexive hsame.

Conversely, if $\rho_{\mathsf{e}}(r, \mathsf{e})$ holds, take $P = \mathsf{e}$ and $Q = r$. The left endpoint is carried by reflexivity, the right endpoint is carried by the left carrier field of the classifier, and the empty-source continuation witness gives $\mathsf{Cont}(\mathsf{e}, r, r)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonClassifier cont result empty witness iff`

The $\mathsf{Poly}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and the displayed BHist-valued operations, so later formalization can target BHist-anchored polynomial laws rather than abstract coefficient operators.

50.4 The certificate

Definition 50.19 (Polynomial source specification)**.** *The source is* $\mathsf{List}^{\uparrow}(R)$ *over a certified scalar ring* $\mathsf{Ring}^{\uparrow}(R)$*, restricted to normalized lists.*

Definition 50.20 (Polynomial pattern specification)**.** *The pattern is finite coefficient-sequence generation followed by canonical trimming.*

Definition 50.21 (Polynomial classifier specification)**.** *The classifier is structural list sameness on normalized coefficient lists.*

Definition 50.22 (Polynomial stability certificate)**.** *The stability certificate consists of:*

1. *normalization removes trailing zeros*
2. *normalization is idempotent*
3. *zero polynomial represented by empty list*
4. *structural reflexivity and transitivity*
5. *addition and multiplication closure after normalization*

Definition 50.23 (Polynomial ledger policy)**.** *Record the original coefficient list and its normalized representative.*

Definition 50.24 ($\mathsf{Poly}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Poly}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Poly}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 50.25 (Zero-axiom encoding note)**.** *Replaces quotienting by trailing-zero equivalence with normalization.*

Closure status for $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$**.**

Constructive story: A polynomial at the mature singleton boundary is a normalized coefficient list whose visible representative is always the empty history: the carrier accepts exactly the histories classified with e, and normalization sends every admitted code back to that endpoint. The classifier is the restricted singleton relation ρ_{e}, so two polynomial codes agree only by carrying both endpoints in A_{e} and showing the underlying histories are hsame, rather than quotienting arbitrary lists by trailing-zero equivalence. The source is the normalized coefficient-list discipline over the singleton ring, the pattern records zero, one, raw addition, multiplication, normalization idempotence, trim compatibility, literal associativity, evaluation, and readback rows, and the ledger keeps the original coefficient list visible together with its normalized empty representative. Because e supplies the carrier witness, hsame is reflexive, symmetric, and transitive, and the singleton carrier transports along history-sameness, the five name-certificate obligations are discharged before any external polynomial algebra is consulted. A history h lies in the signature gap of a $\mathsf{Poly}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with a normalized answer and evidence ledger, and token introduction names that answer as p; here this reduces to h being an admitted singleton code whose normalized coefficient list is e, so p names the terminal zero-polynomial value. The checked standard bridge reads the same endpoint as the one-point normalized polynomial table and reads it back to the empty BEDC representative, preserving addition, multiplication, evaluation, and trimming as endpoint laws. The chapter registers this normalized singleton construction under the name $\mathsf{Poly}^{\uparrow}$; it closes the displayed zero-polynomial surface without forming quotient classes, importing universal polynomial properties, or adding a derivative/evaluation theory beyond the stated rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The paper-side scope is exactly the terminal singleton-history polynomial certificate together with the normal-form boundary, raw addition trim compatibility, literal multiplication associativity, and singleton evaluation and readback rows stated in this chapter and its included polynomial child files.

Formal verification: FormalStatus(NameCert$_{\mathsf{Poly}\uparrow}$, bridgeChecked)
Lean target: `BEDC.Derived.PolynomialUp.singleton\_empty\_history\_polynomial\_laws`
Bridge status: bridgeChecked
Not claimed: This chapter does not close general polynomial algebra over arbitrary scalar rings, quotient-style trailing-zero equivalence, universal polynomial properties, derivative theory, or evaluation theory.
Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 50.26 (Finite additive fold congruence for coefficient lists)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero 0_R, and addition $+$. Assume $0_R \sim_R 0_R$ and the additive congruence field*

$$a \sim_R a' \text{ and } b \sim_R b' \quad \Longrightarrow \quad (a+b) \sim_R (a'+b')$$

for all scalar endpoints. These are among the additive ring fields retained in Definition 43.26; when the scalar source is presented by the commutative-ring certificate, Definition 44.8 keeps these additive fields unchanged.

Let fold_+ be the finite recursive coefficient fold on finite coefficient spines, satisfying

$$\mathsf{fold}_+(\mathsf{nil}) = 0_R, \qquad \mathsf{fold}_+(\mathsf{cons}(a, xs')) = a + \mathsf{fold}_+(xs').$$

For finite coefficient spines $xs, ys : \mathsf{ListCarrier}(R)$, if their same-length pointwise scalar comparison is represented by the structural classifier witness

$$\mathsf{ListClassifierSpec}(\sim_R, xs, ys)$$

from Definition 36.12, then

$$\mathsf{fold}_+(xs) \sim_R \mathsf{fold}_+(ys).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold list classifier hsame`

Proof. Induct on the structural classifier witness. By Definition 36.13, the nil case is the nil-nil constructor and the cons case is the cons-cons constructor; mixed nil-cons shapes have no classifier witness. The same structural witness also records the equal finite-spine length by Definition 36.31.

In the nil-nil case, the two fold equations identify both fold endpoints with 0_R. The assumed zero self-classification $0_R \sim_R 0_R$ gives the fold comparison.

In the cons-cons case, write the two spines as $\mathsf{cons}(a, xs')$ and $\mathsf{cons}(b, ys')$. The constructor clause supplies the head comparison $a \sim_R b$ and the tail classifier witness $\mathsf{ListClassifierSpec}(\sim_R, xs', ys')$. The induction hypothesis gives $\mathsf{fold}_+(xs') \sim_R \mathsf{fold}_+(ys')$. Applying the additive congruence field from Definition 43.26 to these two comparisons yields

$$(a + \mathsf{fold}_+(xs')) \sim_R (b + \mathsf{fold}_+(ys')).$$

The cons fold equations identify these endpoints with $\mathsf{fold}_+(\mathsf{cons}(a, xs'))$ and $\mathsf{fold}_+(\mathsf{cons}(b, ys'))$, giving the displayed fold comparison. □

Theorem 50.27 (Zero convolution summands fold to zero)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero 0_R, and addition $+$, equipped with the additive fields retained in Definition 43.26: zero self-classification, classifier transitivity, additive congruence, and the additive left-zero law. For a commutative-ring scalar source, these are unchanged by Definition 44.8.*

Let fold_+ be the finite recursive coefficient fold from Definition 50.26. Let $\mathsf{summ}_I : \mathsf{ListCarrier}(R)$ be the finite spine of scalar summands selected by a raw Cauchy convolution index spine I, and

let $\mathsf{zero}_I : \mathsf{ListCarrier}(R)$ *be the same finite spine with every entry replaced by* 0_R*. If the all-zero summand condition is represented by the structural witness*

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{summ}_I, \mathsf{zero}_I),$$

then

$$\mathsf{fold}_+(\mathsf{summ}_I) \sim_R 0_R.$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold zero remainder empty`

Proof. The displayed hypothesis is a finite-spine classifier witness as in Definition 36.12. By Definition 36.13, its only possible constructor cases are nil-nil and cons-cons. Induct on this structural witness.

In the nil-nil case, both spines are nil. The fold equation from Definition 50.26 identifies the left endpoint with 0_R, and zero self-classification supplies $0_R \sim_R 0_R$.

In the cons-cons case, write the summand spine as $\mathsf{cons}(a, xs')$ and the matching zero spine as $\mathsf{cons}(0_R, zs')$. The constructor clause gives the head comparison $a \sim_R 0_R$ and the tail witness $\mathsf{ListClassifierSpec}(\sim_R, xs', zs')$. The induction hypothesis gives

$$\mathsf{fold}_+(xs') \sim_R 0_R.$$

Applying additive congruence from Definition 43.26 to the head comparison and this tail comparison yields

$$(a + \mathsf{fold}_+(xs')) \sim_R (0_R + 0_R).$$

The additive left-zero law identifies $0_R + 0_R$ with 0_R. Classifier transitivity composes these two comparisons. The cons fold equation then identifies $a + \mathsf{fold}_+(xs')$ with $\mathsf{fold}_+(\mathsf{cons}(a, xs'))$, giving the claimed fold comparison for the cons spine. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder addFold empty classified`

Definition 50.28 (Classified zero remainder spine)**.** *Let* R *be a scalar carrier with classifier* $\sim_R$ *and zero endpoint* 0_R*. For a finite coefficient spine* $t : \mathsf{ListCarrier}(R)$*,* $\mathsf{ZeroRem}_R(t)$ *means that every displayed coefficient in* t *is scalar-classified as zero. It is the finite-spine predicate generated by the nil clause and the cons clause*

$$a \sim_R 0_R \quad \textit{and} \quad \mathsf{ZeroRem}_R(t') \quad \Longrightarrow \quad \mathsf{ZeroRem}_R(\mathsf{cons}(a, t')).$$

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder`

Lemma 50.29 (Trim invariance under a common classified spine and zero remainders)**.** *Let* R *be a scalar carrier with classifier* $\sim_R$*, zero endpoint* 0_R*, and scalar certificate* $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$*. Assume* $0_R \sim_R 0_R$*. For a commutative-ring scalar source, Definition 44.8 keeps the underlying ring carrier, zero endpoint, and classifier unchanged. Let* trim_R *be the canonical polynomial trimming operation specified by Definition 50.20 and Definition 50.22.*

For finite coefficient spines $p, q, r, s : \mathsf{ListCarrier}(R)$*, if*

$$\mathsf{ListClassifierSpec}(\sim_R, p, q), \qquad \mathsf{ZeroRem}_R(r), \qquad \mathsf{ZeroRem}_R(s)$$

hold, then the normalized representatives are structurally list-same:

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p{+}{+}r), \mathsf{trim}_R(q{+}{+}s)).$$

Proof. By Definition 36.31, the classified common spines p and q have the same finite length. The proof is by induction on the displayed witness $\mathsf{ListClassifierSpec}(\sim_R, p, q)$ from Definition 36.12.

First record the zero-remainder base fact. An induction on $\mathsf{ZeroRem}_R(t)$, using the trailing-zero removal and empty zero-polynomial clauses in Definition 50.22, identifies $\mathsf{trim}_R(t)$ with nil. Hence two zero-remainder spines trim to endpoints classified by the nil clause of $\mathsf{ListClassifierSpec}$.

In the nil-nil case for the common spine, p and q are both nil. The two original lists are exactly the remainders r and s, so the zero-remainder base fact gives the nil-nil classifier witness for their trims.

In the cons-cons case, write

$$p = \mathsf{cons}(a, p'), \qquad q = \mathsf{cons}(b, q').$$

The classifier constructor supplies $a \sim_R b$ and a tail witness $\mathsf{ListClassifierSpec}(\sim_R, p', q')$. Applying the induction hypothesis to p', q' with the same zero remainders gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p' {+}{+} r), \mathsf{trim}_R(q' {+}{+} s)).$$

By Definition 36.13, the two trimmed tails have matching nil or cons shape; the mixed shapes are impossible.

If the trimmed tails are cons spines, canonical trimming keeps the displayed heads because a nonempty normalized tail contains a nonzero highest entry. The cons clause of $\mathsf{ListClassifierSpec}$ then combines the head witness $a \sim_R b$ with the classified trimmed tails.

If the trimmed tails are both nil, the canonical cons-trim clause tests only whether the displayed head is scalar-classified as zero. The head witness $a \sim_R b$, together with the symmetry and transitivity fields of cert_R, gives

$$a \sim_R 0_R \iff b \sim_R 0_R.$$

Thus the two trim computations take the same branch. In the dropped-head branch, both normalized representatives are nil and the nil clause classifies them. In the retained-head branch, both normalized representatives are singleton spines, and the cons clause classifies them using $a \sim_R b$ and the nil-tail classifier.

These cases exhaust the structural classifier witness by Definition 36.13. Finally, Definition 50.21 reads the resulting finite-spine classifier as structural sameness of the two normalized polynomial representatives. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold common spine zero remainder hsame`

Definition 50.30 (Polynomial raw addition comparison data)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$. For finite coefficient spines, the zero-padded raw addition radd_R is the recursion*

$$\begin{aligned}
\mathsf{radd}_R(\mathsf{nil}, \mathsf{nil}) &= \mathsf{nil}, \\
\mathsf{radd}_R(\mathsf{cons}(a, x), \mathsf{nil}) &= \mathsf{cons}(a + 0_R, \mathsf{radd}_R(x, \mathsf{nil})), \\
\mathsf{radd}_R(\mathsf{nil}, \mathsf{cons}(b, y)) &= \mathsf{cons}(0_R + b, \mathsf{radd}_R(\mathsf{nil}, y)), \\
\mathsf{radd}_R(\mathsf{cons}(a, x), \mathsf{cons}(b, y)) &= \mathsf{cons}(a + b, \mathsf{radd}_R(x, y)).
\end{aligned}$$

For the canonical trim trim_R specified by Definition 50.20 and Definition 50.22, write the polynomial raw classifier as

$$\mathsf{PolySame}_R(x, y) \iff \mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(x), \mathsf{trim}_R(y)).$$

This is the normalized structural classifier of Definition 50.21 applied to raw coefficient spines.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd`

Lemma 50.31 (Raw addition zero-tail trim stability)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume $0_R \sim_R 0_R$ and the additive fields retained in Definition 43.26: classifier transitivity, additive congruence, and the additive zero laws. For a commutative-ring scalar source, these fields are unchanged by Definition 44.8.*

For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(z)$ holds in the sense of Definition 50.28, then

$$\mathsf{PolySame}_R(\mathsf{radd}_R(x{+}{+}z, y), \mathsf{radd}_R(x, y))$$

and

$$\mathsf{PolySame}_R(\mathsf{radd}_R(x, y{+}{+}z), \mathsf{radd}_R(x, y)).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd left zero tail addFold invariance`

Proof. The nil spine is a zero remainder by the nil clause of Definition 50.28. Consider the left-input statement. The recursion in Definition 50.30 scans the two raw sums coefficient by coefficient. Induct on the displayed finite spines, first through the common prefix x and then through the zero remainder z after x is exhausted.

At a position where both raw sums use the same ordinary coefficient from x, source reflexivity from cert_R and additive congruence classify the two displayed sum coefficients. At a position where the extended input supplies a zero-remainder coefficient a while the unextended input supplies the padding coefficient 0_R, the hypothesis $a \sim_R 0_R$ and source reflexivity on the other displayed coefficient classify $a + b$ with $0_R + b$. If the other input is also padding, additive congruence classifies $a + 0_R$ with $0_R + 0_R$, and the additive zero law plus transitivity classifies this exposed coefficient as 0_R.

Thus the raw sum $\mathsf{radd}_R(x{+}{+}z, y)$ decomposes as $c{+}{+}t$ so that $\mathsf{ListClassifierSpec}(\sim_R, c, \mathsf{radd}_R(x, y))$ holds and $\mathsf{ZeroRem}_R(t)$ holds. Applying Definition 50.29 with the common classified spines c and $\mathsf{radd}_R(x, y)$, the zero remainder t, and the nil zero remainder gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(x{+}{+}z, y)), \mathsf{trim}_R(\mathsf{radd}_R(x, y))).$$

By Definition 50.30, this is the first displayed $\mathsf{PolySame}_R$ claim.

The right-input statement is the same coefficient scan with the zero-remainder coefficient appearing in the right summand. The non-padding comparison is $b + a$ against $b + 0_R$, handled by additive congruence from $a \sim_R 0_R$ and source reflexivity on b; the padding comparison is $0_R + a$ against $0_R + 0_R$, handled by additive congruence, the additive zero law, and transitivity. The resulting decomposition and the same application of Definition 50.29 give the second displayed $\mathsf{PolySame}_R$ claim. □

Theorem 50.32 (Polynomial raw addition two-sided trim compatibility)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive ring fields cited in Definition 50.31. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q))).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By the trailing-zero removal clause in Definition 50.22, the canonical trim decompositions present

$$p = p_0{+}{+}z_p, \qquad q = q_0{+}{+}z_q,$$

with $\mathsf{ZeroRem}_R(z_p)$ and $\mathsf{ZeroRem}_R(z_q)$.

Apply the left-input part of Definition 50.31 to p_0, q, z_p. After substituting the first trim decomposition, this gives

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(p_0, q)).$$

Apply the right-input part of the same lemma to p_0, q_0, z_q. After substituting the second trim decomposition, this gives

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p_0, q), \mathsf{radd}_R(p_0, q_0)).$$

Unfold $\mathsf{PolySame}_R$ using Definition 50.30. The two displayed witnesses are structural list-classifier witnesses between the three normalized coefficient spines. Transitivity of $\sim_R$ from cert_R and Definition 36.15 compose them, giving

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{trim}_R(\mathsf{radd}_R(p_0, q_0))).$$

Substituting $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$, this is exactly the asserted polynomial classifier comparison. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd two sided trim compatibility`

Proposition 50.33 (Polynomial raw addition right zero-tail invariance)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume $0_R \sim_R 0_R$ and the additive fields retained in Definition 43.26: classifier transitivity, additive congruence, and the additive zero laws. For a commutative-ring scalar source, these fields are unchanged by Definition 44.8.*

For finite coefficient spines $p, q, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q{+}{+}t), \mathsf{radd}_R(p, q)).$$

Proof. Apply Definition 50.31 with $x = p$, $y = q$, and $z = t$, and read its second displayed conclusion. The hypothesis on t is exactly the classified zero-tail condition from Definition 50.28; Definition 50.30 identifies the displayed relation as the normalized polynomial classifier. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd right zero tail addFold invariance`

Definition 50.34 (Polynomial normalized singleton representative)**.** *Let R be a scalar carrier with classifier $\sim_R$ and zero endpoint 0_R. For a scalar coefficient c, define*

$$\mathsf{NormOne}_R(c) \iff \neg(c \sim_R 0_R).$$

Thus the singleton coefficient spine $\mathsf{cons}(c, \mathsf{nil})$ is a normalized singleton representative exactly when $\mathsf{NormOne}_R(c)$ holds, while nil is the empty zero-polynomial representative. This is the one-coefficient boundary of the trailing-zero convention in Definition 50.3 and the canonical trimming rule in Definition 50.22.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonNormalizedRepresentative`

Proposition 50.35 (Raw polynomial addition leaves singleton normal form)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be the raw addition and normalized raw classifier of Definition 50.30. For scalar sources presented through $\mathsf{CommRing}^{\uparrow}$, Definition 44.8 reads these as the retained $\mathsf{Ring}^{\uparrow}$ addition, zero endpoint, and classifier.*

Suppose a and b are classified nonzero and add to a coefficient classified as zero:

$$\neg(a \sim_R 0_R), \qquad \neg(b \sim_R 0_R), \qquad (a+b) \sim_R 0_R.$$

Set

$$p := \mathsf{cons}(a, \mathsf{nil}), \qquad q := \mathsf{cons}(b, \mathsf{nil}).$$

Then p and q are normalized singleton representatives, but their raw zero-padded sum is not a normalized singleton representative:

$$\mathsf{NormOne}_R(a) \wedge \mathsf{NormOne}_R(b),$$

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{nil}),$$

$$\neg\mathsf{NormOne}_R(a+b).$$

Moreover its canonical normalized representative is the empty zero representative, and this is exactly what the polynomial classifier records:

$$\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}, \qquad \mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{nil}).$$

Proof. By Definition 50.166, the assumptions $\neg(a \sim_R 0_R)$ and $\neg(b \sim_R 0_R)$ are precisely the witnesses $\mathsf{NormOne}_R(a)$ and $\mathsf{NormOne}_R(b)$, so p and q are normalized singleton representatives.

Unfolding p and q, the cons-cons branch of the raw addition recursion in Definition 50.30 gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{radd}_R(\mathsf{nil}, \mathsf{nil})).$$

The nil-nil branch gives $\mathsf{radd}_R(\mathsf{nil}, \mathsf{nil}) = \mathsf{nil}$, hence

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{nil}).$$

The coefficient hypothesis $(a+b) \sim_R 0_R$ contradicts any witness of $\mathsf{NormOne}_R(a+b)$, since such a witness is a proof of $\neg((a+b) \sim_R 0_R)$. Therefore $\neg\mathsf{NormOne}_R(a+b)$, so the raw singleton spine is not admitted as a normalized singleton representative.

The same coefficient hypothesis activates the trailing-zero clause of the canonical trim from Definition 50.22. Therefore the singleton spine $\mathsf{cons}(a+b, \mathsf{nil})$ trims to nil, and the raw-sum computation above gives

$$\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}.$$

The empty zero-polynomial clause gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{nil}, \mathsf{nil}),$$

which is the nil-nil constructor of the finite-spine classifier from Definition 36.12, as isolated by Definition 36.13. Hence $\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{nil})$ holds. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd singleton normal form boundary`

Definition 50.36 (Polynomial singleton constant-source schema instance)**.** *The concrete constant-source instance for* $\mathsf{Poly}^{\uparrow}$ *uses the singleton empty-history polynomial carrier. Its coefficient source is the existing singleton scalar carrier, its polynomial constructor is the constant singleton coefficient row* $\mathsf{cons}(c, \mathsf{nil})$*, and its evaluation row is the singleton evaluation readback* $\mathsf{PolynomialSingletonEval}$ *at the displayed evaluation endpoint. The classifier is the singleton polynomial classifier on these* BHist *rows, not an abstract coefficient equivalence.*

Theorem 50.37 (Polynomial singleton constant evaluation boundary)**.** *Every constant-source singleton polynomial from Definition 50.168 evaluates to the displayed singleton coefficient endpoint. For a coefficient row c and any singleton evaluation endpoint α, the readback of the constant spine $\mathsf{cons}(c, \mathsf{nil})$ is classified by the singleton polynomial classifier as the same endpoint c; at the empty-history endpoint, the zero constant-term readback gives the same classifier boundary. Thus the evaluation classifier exported by this concrete instance is exactly the singleton coefficient classifier.*

Proof. The carrier is the one-coefficient boundary isolated by Definition 50.166 and Definition 50.167: a constant-source row is presented by the singleton spine $\mathsf{cons}(c, \mathsf{nil})$, with no additional coefficient family or abstract scalar quotient.

The singleton evaluation readback applies the checked singleton row PolynomialSingletonEval_singleton, which identifies evaluation of $\mathsf{cons}(c, \mathsf{nil})$ with the coefficient endpoint c under the singleton polynomial classifier. At the empty-history endpoint, PolynomialSingletonEval_zero_constant_term gives the same endpoint boundary for the zero constant-term readback.

Therefore the only classifier visible in the concrete constant-source schema is the singleton polynomial classifier already attached to the coefficient row. Repacking the carrier, constructor, and evaluation readback gives the stated boundary. □

Proposition 50.38 (Literal polynomial add-trim equality fails at cancelling singletons)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be as in Definition 50.30. Suppose $\neg(a \sim_R 0_R)$, $\neg(b \sim_R 0_R)$, and $(a + b) \sim_R 0_R$. Write $[c] := \mathsf{cons}(c, \mathsf{nil})$, and set $p := [a]$ and $q := [b]$. Then literal finite-spine equality does not commute with raw addition and trimming:*

$$\begin{aligned} \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &= \mathsf{nil}, \\ \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) &= [a + b], \\ \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &\neq \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)). \end{aligned}$$

Nevertheless the two displayed outputs are polynomial-classified the same:

$$\mathsf{PolySame}_R\big(\mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q))\big).$$

For the canonical integer coefficient ring from chapter 31 and chapter 44, taking $a = 1$ and $b = -1$ gives $p = [1]$, $q = [-1]$, $\mathsf{trim}_{\mathbb{Z}}(\mathsf{radd}_{\mathbb{Z}}(p, q)) = \mathsf{nil}$, and $\mathsf{radd}_{\mathbb{Z}}(\mathsf{trim}_{\mathbb{Z}}(p), \mathsf{trim}_{\mathbb{Z}}(q)) = [0]$.

Proof. The nonzero hypotheses make p and q normalized singleton representatives by Definition 50.166; the canonical trim fixes normalized representatives, so $\mathsf{trim}_R(p) = p$ and $\mathsf{trim}_R(q) = q$. By the cons-cons and nil-nil clauses of radd_R in Definition 50.30, $\mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) = \mathsf{radd}_R([a], [b]) = [a + b]$. The equality $\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}$ is the singleton cancellation conclusion of Definition 50.167. The displayed inequality follows because the left side is nil while the right side is a cons spine. Constructor no-confusion for the finite-spine carrier of Definition 36.1 is the mixed nil/cons separation isolated in Definition 36.13 and Definition 36.14. For classifier comparison, unfold $\mathsf{PolySame}_R$. The first argument trims again to nil. The second argument is $[a + b]$, and the hypothesis $(a + b) \sim_R 0_R$ sends its singleton coefficient to the trailing-zero branch, so its trim is also nil. The nil-nil clause of ListClassifierSpec gives the displayed $\mathsf{PolySame}_R$ witness. The integer specialization uses the standard integer facts $1 \neq 0$, $-1 \neq 0$, and $1 + (-1) = 0$; Definition 44.8 keeps the ring operations and classifier when the commutative-ring source is read through $\mathsf{Ring}^{\uparrow}$. □

Definition 50.39 (Polynomial Horner evaluation)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a carried scalar point α, define the Horner evaluation on finite coefficient spines from Definition 36.1 by recursion*

$$\mathsf{eval}_{\alpha,R}(\mathsf{nil}) := 0_R.$$

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a,x)) := a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Define the normalized polynomial evaluation by

$$\mathsf{Eval}_{\alpha,R}(x) := \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(x)).$$

Define the normalized polynomial sum by

$$\mathsf{PolyAdd}_R(x,y) := \mathsf{trim}_R(\mathsf{radd}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

On a normalized representative from Definition 50.3, $\mathsf{Eval}_{\alpha,R}$ is the Horner evaluation itself, by trim idempotence from Definition 50.22.

Lemma 50.40 (Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{\alpha,R}(x{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Consequently, for every finite coefficient spine x,

$$\mathsf{Eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder addFold empty`

Proof. First prove the endpoint claim $\mathsf{eval}_{\alpha,R}(t) \sim_R 0_R$ for every classified zero tail t. Induct on the displayed $\mathsf{ZeroRem}_R(t)$ witness. The nil clause is the defining equation $\mathsf{eval}_{\alpha,R}(\mathsf{nil}) = 0_R$ together with zero self-classification from Definition 43.26. In the cons clause, write $t = \mathsf{cons}(a, t')$. The constructor gives $a \sim_R 0_R$ and the induction hypothesis gives $\mathsf{eval}_{\alpha,R}(t') \sim_R 0_R$. Multiplication congruence and the zero-multiplication row in Definition 43.26 classify $\alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with 0_R. Addition congruence then classifies $a + \alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with $0_R + 0_R$, and the additive left-zero law reduces the latter endpoint to 0_R.

For the displayed tail-invariance statement, induct on x. The nil case is the endpoint claim just proved. In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives $\mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports this comparison through left multiplication by α, and addition congruence with source reflexivity on a gives

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x').$$

The two Horner recursion equations identify these endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'{+}{+}t))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

For the consequence, the trailing-zero removal clause in Definition 50.22 presents each raw spine x as $\mathsf{trim}_R(x){+}{+}t$ with $\mathsf{ZeroRem}_R(t)$. Applying the first part gives $\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{Eval}_{\alpha,R}(x)$, and symmetry of the scalar classifier reverses the displayed comparison. □

Lemma 50.41 (Horner evaluation respects raw polynomial addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold append hsame`

Proof. Perform induction on the two finite spines, following the four recursion clauses for radd_R in Definition 50.30.

The nil-nil case compares 0_R with $0_R + 0_R$, so it is exactly the additive left-zero law, read with symmetry when needed.

In the cons-nil case, write $x = \mathsf{cons}(a, x')$. The raw-addition recursion exposes the head $a + 0_R$ and the tail $\mathsf{radd}_R(x', \mathsf{nil})$. The induction hypothesis classifies the tail evaluation with $\mathsf{eval}_{\alpha,R}(x') + 0_R$, and Definition 43.31 reduces that endpoint to $\mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports the tail comparison through left multiplication by α. Additive congruence and the additive right-zero law reduce the exposed head $a + 0_R$ to a, and the target endpoint $(\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))) + 0_R$ to $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

The nil-cons case is symmetric. The raw head is $0_R + b$, which reduces to b by the additive left-zero law, and the outer target $0_R + \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ reduces by the same law. The tail comparison is transported through multiplication by α exactly as in the cons-nil case.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x', y')) \sim_R \mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y').$$

Multiplication congruence transports this through left multiplication by α, and left distributivity from Definition 43.26 classifies

$$\alpha \cdot (\mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y'))$$

with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence therefore transports the Horner endpoint for $\mathsf{radd}_R(x, y)$ to

$$(a + b) + (\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Apply Definition 42.13 to the additive abelian-group operation, with the four additive blocks a, b, $\alpha \cdot \mathsf{eval}_{\alpha,R}(x')$, and $\alpha \cdot \mathsf{eval}_{\alpha,R}(y')$. This classifies the last endpoint with

$$(a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x')) + (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')),$$

which is exactly $\mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y)$ by the Horner recursion. □

Theorem 50.42 (Polynomial evaluation is additive)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) + \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.133,

$$\mathsf{PolyAdd}_R(p, q) = \mathsf{trim}_R(\mathsf{radd}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies $\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q))$ with $\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0))$. Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)).$$

By Definition 50.135,

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) + \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$, and $\mathsf{Eval}_{\alpha,R}$ on each displayed representative is the unnormalized Horner evaluation. Substituting these equalities gives the carrier formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawAdd classified`

Lemma 50.43 (Horner evaluation respects structural classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$, if*

$$\mathsf{ListClassifierSpec}(\sim_R, x, y)$$

holds, then

$$\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the displayed structural classifier witness from Definition 36.12. By Definition 36.13, the only constructor cases are nil-nil and cons-cons; mixed nil-cons shapes have no classifier witness.

In the nil-nil case, the two Horner equations in Definition 50.133 identify both endpoints with 0_R. The zero self-classification field retained in Definition 43.26 gives $0_R \sim_R 0_R$.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The classifier constructor supplies $a \sim_R b$ and a tail witness $\mathsf{ListClassifierSpec}(\sim_R, x', y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(x') \sim_R \mathsf{eval}_{\alpha,R}(y').$$

The carried point α is self-classified by the scalar certificate. Multiplication congruence from Definition 43.26 transports the tail comparison through left multiplication by α, giving

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence combines this comparison with the head comparison $a \sim_R b$:

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

The cons recursion equation in Definition 50.133 identifies these two endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$, respectively. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval list classifier classified`

Theorem 50.44 (Polynomial evaluation respects the polynomial classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(p, q) \Longrightarrow \mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(q).$$

Proof. Unfold $\mathsf{PolySame}_R$ using Definition 50.30. A classifier witness for p and q is exactly

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p), \mathsf{trim}_R(q)).$$

Apply Definition 50.137 to the two trimmed spines. This gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(q)).$$

The normalized evaluation definition in Definition 50.133 reads the left endpoint as $\mathsf{Eval}_{\alpha,R}(p)$ and the right endpoint as $\mathsf{Eval}_{\alpha,R}(q)$, giving the displayed implication. □

Definition 50.45 (Polynomial raw Cauchy product)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a scalar coefficient a and a finite coefficient spine $y : \mathsf{ListCarrier}(R)$, define scalar multiplication of coefficient spines by*

$$\mathsf{rscale}_R(a, \mathsf{nil}) := \mathsf{nil}, \qquad \mathsf{rscale}_R(a, \mathsf{cons}(b, y)) := \mathsf{cons}(a \cdot b, \mathsf{rscale}_R(a, y)).$$

Define the one-place coefficient shift by

$$\mathsf{shift}_R(y) := \mathsf{cons}(0_R, y).$$

The raw polynomial product is the finite recursion

$$\mathsf{rmul}_R(\mathsf{nil}, y) := \mathsf{nil},$$

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), y) := \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(x, y))).$$

The normalized polynomial product is

$$\mathsf{PolyMul}_R(x, y) := \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

This is the Cauchy product in low-coefficient-first Horner form: the cons clause expresses $(a + Xx)y = ay + X(xy)$.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul`

Lemma 50.46 (Horner evaluation of scalar-multiplied coefficient spines)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every carried scalar coefficient a and every finite coefficient spine $y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) \sim_R a \cdot \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawScale classified`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval raw scale empty classified`

Proof. Induct on the finite spine y from Definition 36.1. In the nil case, both the scalar-multiple spine and its Horner value are 0_R. The target endpoint is $a \cdot 0_R$, and the zero-multiplication row in Definition 43.26, with symmetry of $\sim_R$, gives the displayed comparison.

In the cons case, write $y = \mathsf{cons}(b, y')$. The recursion in Definition 50.139 and the Horner recursion in Definition 50.133 identify the left endpoint with

$$a \cdot b + \alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y')).$$

The induction hypothesis and multiplication congruence classify this endpoint with

$$a \cdot b + \alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Multiplicative associativity, multiplicative commutativity from Definition 44.33, and multiplication congruence classify

$$\alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Addition congruence transports this comparison under the common summand $a \cdot b$. The left distributive row in Definition 43.26, used in the reverse direction, then identifies

$$a \cdot b + a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

The final endpoint is $a \cdot \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ by the Horner recursion, and transitivity composes the comparisons. □

Lemma 50.47 (Horner evaluation of coefficient shift)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(x)) \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Proof. By Definition 50.139, $\mathsf{shift}_R(x) = \mathsf{cons}(0_R, x)$. The Horner recursion from Definition 50.133 makes its evaluation

$$0_R + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

The additive left-zero row in Definition 43.26 classifies this endpoint with $\alpha \cdot \mathsf{eval}_{\alpha,R}(x)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval coefficient shift`

Lemma 50.48 (Horner evaluation respects raw polynomial multiplication)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) \cdot \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the first spine x. In the nil case, $\mathsf{rmul}_R(\mathsf{nil}, y) = \mathsf{nil}$, so the left endpoint evaluates to 0_R. The target endpoint is $0_R \cdot \mathsf{eval}_{\alpha,R}(y)$, and the zero-multiplication row in Definition 43.26, with symmetry, gives the comparison.

For the cons case, write $x = \mathsf{cons}(a, x')$, put $u = \mathsf{eval}_{\alpha,R}(x')$, and put $v = \mathsf{eval}_{\alpha,R}(y)$. The raw product recursion in Definition 50.139 and the raw additivity theorem Definition 50.135 give

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) + \mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(\mathsf{rmul}_R(x', y))).$$

By Definition 50.140, the first summand is classified with $a \cdot v$. By Definition 50.141, the second summand is classified with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x', y)).$$

The induction hypothesis and multiplication congruence classify this with $\alpha \cdot (u \cdot v)$. Multiplicative associativity classifies $\alpha \cdot (u \cdot v)$ with $(\alpha \cdot u) \cdot v$. Addition congruence therefore gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R a \cdot v + (\alpha \cdot u) \cdot v.$$

The right distributive row retained by Definition 44.8, equivalently derived from Definition 44.34, classifies

$$(a + \alpha \cdot u) \cdot v$$

with

$$a \cdot v + (\alpha \cdot u) \cdot v.$$

Using symmetry of this distributive comparison and the Horner recursion

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x')) = a + \alpha \cdot u,$$

transitivity gives the required cons-case endpoint. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawMul classified`

Theorem 50.49 (Polynomial evaluation is multiplicative)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) \cdot \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.139,

$$\mathsf{PolyMul}_R(p, q) = \mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies the normalized evaluation of this endpoint with

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

By Definition 50.142,

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) \cdot \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation in Definition 50.133. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$. The product endpoint is normalized by its defining trim, so normalized evaluation on $\mathsf{PolyMul}_R(p, q)$ is the unnormalized Horner evaluation of that product. Substituting these equalities gives the carrier formula. □

Theorem 50.50 (Polynomial evaluation is a semiring morphism through classification)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q, p', q' : \mathsf{ListCarrier}(R)$, if $\mathsf{PolySame}_R(p, p')$ and $\mathsf{PolySame}_R(q, q')$, then*

$$\mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(p'),$$

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q),$$

and

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval semiring morphism`

Proof. The first clause is exactly Definition 50.138 applied to the classifier witness between p and p'.

The second and third clauses are the normalized-operation laws from Definition 50.136 and Definition 50.143. They apply to the original representatives p and q, so the operation endpoints are already the canonical normalized polynomial operations.

When a later scalar comparison is written with p' or q' instead, apply Definition 50.138 to the appropriate classifier witness and compose by transitivity of $\sim_R$. Thus the descended endpoints may be substituted in the additive and multiplicative scalar comparisons without reassembling the three facts. □

Lemma 50.51 (Zero-point Horner cons evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. The cons recursion in Definition 50.133 identifies the left endpoint with

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t).$$

The zero-absorption theorem Definition 43.60, applied to $\mathsf{eval}_{0_R,R}(t)$, gives

$$0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R 0_R.$$

Classifier reflexivity supplies $a \sim_R a$, and addition congruence from Definition 43.26 gives

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R a + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $a + 0_R \sim_R a$. Classifier transitivity composes the two comparisons. □

Lean4 ✓ BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero cons constant

Lemma 50.52 (Zero-point Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{0_R,R}(x{+}{+}t) \sim_R \mathsf{eval}_{0_R,R}(x).$$

Proof. First record the endpoint claim that every classified zero remainder evaluates to zero at 0_R. If the zero remainder is nil, the nil recursion in Definition 50.133 identifies its evaluation with 0_R, and zero self-classification from Definition 43.26 gives the claim. If the zero remainder is $\mathsf{cons}(c, t')$, the cons clause of Definition 50.28 supplies $c \sim_R 0_R$. By Definition 50.145,

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R c,$$

and transitivity gives $\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R 0_R$.

Now split on the first constructor of x. If $x = \mathsf{nil}$, then $x{+}{+}t = t$, so the endpoint claim gives the displayed comparison. If $x = \mathsf{cons}(a, x')$, then $x{+}{+}t = \mathsf{cons}(a, x'{+}{+}t)$. Applying Definition 50.145 twice gives

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x'{+}{+}t)) \sim_R a$$

and

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x')) \sim_R a.$$

Symmetry of the second comparison and transitivity give the required comparison for the cons case. □

Lean4 ✓ BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero point zero tail invariance

Theorem 50.53 (Polynomial evaluation at zero returns the constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. Set $x = \mathsf{cons}(a, t)$. The trailing-zero removal clause in Definition 50.22 gives a finite zero remainder z such that

$$x = \mathsf{trim}_R(x){+}{+}z, \qquad \mathsf{ZeroRem}_R(z).$$

Applying Definition 50.146 to $\mathsf{trim}_R(x)$ and z yields

$$\mathsf{eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(\mathsf{trim}_R(x)).$$

By the normalized evaluation definition in Definition 50.133, the right endpoint is $\mathsf{Eval}_{0_R,R}(x)$. Symmetry gives

$$\mathsf{Eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(x).$$

The raw cons comparison Definition 50.145 gives

$$\mathsf{eval}_{0_R,R}(x) \sim_R a.$$

Classifier transitivity composes these two comparisons, and substituting $x = \mathsf{cons}(a, t)$ gives the displayed formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero constant term`

Lemma 50.54 (Unit-point Horner evaluation is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26, satisfying*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) = 0_R, \qquad \mathsf{fold}_{+,R}(\mathsf{cons}(a, x')) = a + \mathsf{fold}_{+,R}(x').$$

For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,

$$\mathsf{eval}_{1_R,R}(x) \sim_R \mathsf{fold}_{+,R}(x).$$

Proof. Induct on the finite spine x from Definition 36.1.

In the nil case, the nil Horner equation in Definition 50.133 and the nil fold equation identify both endpoints with 0_R. The zero self-classification row of Definition 43.26 gives the displayed comparison.

In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives

$$\mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

The recursive Horner endpoint $\mathsf{eval}_{1_R,R}(x')$ is carried by induction on x' using the scalar zero, addition, and multiplication closure rows of the $\mathsf{Ring}^{\uparrow}$ certificate. Hence the multiplicative left-unit row retained in Definition 43.26 gives

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{eval}_{1_R,R}(x').$$

Classifier transitivity composes this unit comparison with the induction hypothesis, yielding

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

Source reflexivity gives $a \sim_R a$. Applying additive congruence from Definition 43.26 gives

$$a + 1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R a + \mathsf{fold}_{+,R}(x').$$

The cons Horner equation identifies the left endpoint with $\mathsf{eval}_{1_R,R}(\mathsf{cons}(a, x'))$, and the cons fold equation identifies the right endpoint with $\mathsf{fold}_{+,R}(\mathsf{cons}(a, x'))$. □

Theorem 50.55 (Polynomial evaluation at the multiplicative unit is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{1_R,R}(p) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Proof. By the normalized evaluation definition in Definition 50.133,

$$\mathsf{Eval}_{1_R,R}(p) = \mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)).$$

Apply Definition 50.148 to the finite coefficient spine $\mathsf{trim}_R(p)$. This gives

$$\mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Substituting the displayed definition of $\mathsf{Eval}_{1_R,R}(p)$ gives the claimed comparison. □

Lemma 50.56 (Polynomial raw addition structural swap)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$. Assume scalar additive commutativity:*

$$a + b \sim_R b + a$$

for all scalar coefficients a, b. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Induct simultaneously on the two finite coefficient spines, following the four recursion clauses of radd_R in Definition 50.30. The target classifier is the finite-spine structural classifier of Definition 36.12, with matching constructors isolated by Definition 36.13.

In the nil-nil case, both raw additions are nil, so the nil clause of $\mathsf{ListClassifierSpec}$ gives the claim.

In the cons-nil case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{nil}$. The recursion gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a + 0_R, \mathsf{radd}_R(p', \mathsf{nil}))$$

and

$$\mathsf{radd}_R(q, p) = \mathsf{cons}(0_R + a, \mathsf{radd}_R(\mathsf{nil}, p')).$$

Scalar additive commutativity gives $a + 0_R \sim_R 0_R + a$. The induction hypothesis applied to p' and nil gives the tail classifier

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', \mathsf{nil}), \mathsf{radd}_R(\mathsf{nil}, p')).$$

The cons clause of $\mathsf{ListClassifierSpec}$ combines the head comparison and this tail comparison.

In the nil-cons case, write $p = \mathsf{nil}$ and $q = \mathsf{cons}(b, q')$. The recursion gives heads $0_R + b$ and $b + 0_R$. Scalar additive commutativity gives $0_R + b \sim_R b + 0_R$, and the induction hypothesis at nil and q' supplies the tail classifier. The cons clause rebuilds the displayed list-classifier witness.

In the cons-cons case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{cons}(b, q')$. The recursion exposes heads $a + b$ and $b + a$, related by scalar additive commutativity. The induction hypothesis supplies

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', q'), \mathsf{radd}_R(q', p')).$$

The cons clause combines the head and tail witnesses, giving the structural classifier for the two raw sums. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd structural swap`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd commutative classified`

Theorem 50.57 (Polynomial raw addition commutativity from scalar additive commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Use the additive abelian-group and classifier fields retained in Definition 43.26: zero self-classification, classifier transitivity, additive congruence, additive zero laws, and scalar additive commutativity. For a commutative-ring scalar source, Definition 44.8 keeps these data unchanged. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Apply Definition 50.150 to p and q using the scalar additive commutativity row from Definition 43.26. This gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p,q), \mathsf{radd}_R(q,p)).$$

The nil finite spine is a zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 with common classified spines $\mathsf{radd}_R(p,q)$ and $\mathsf{radd}_R(q,p)$, and with both zero remainders equal to nil. Finite-spine append by nil leaves each spine unchanged, so the lemma yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(p,q)), \mathsf{trim}_R(\mathsf{radd}_R(q,p))).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, this is exactly the displayed polynomial raw classifier comparison. □

Lemma 50.58 (Raw polynomial addition is structurally associative)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive associativity, additive zero laws, classifier symmetry, classifier transitivity, and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(\mathsf{radd}_R(p,q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q,r))).$$

Proof. Induct on the finite triple spine (p, q, r), using the maximum of the three finite lengths as the decreasing measure. The nil-nil-nil case unfolds both raw additions by Definition 50.30 to nil, and the nil clause of Definition 36.12 gives the classifier witness.

In every remaining constructor case, at least one of the three spines is a cons spine. Unfold the corresponding clauses of radd_R from Definition 50.30. Both associated raw sums expose a cons head, and their tails are exactly the two associated raw sums for the tail triple, with nil used as the tail of any absent spine. The induction hypothesis supplies the recursive tail classifier.

It remains to classify the exposed scalar heads. If all three heads are present, say a, b, c, the comparison is

$$(a+b)+c \sim_R a+(b+c),$$

which is scalar additive associativity. If exactly two heads are present, the missing head is the padding coefficient 0_R, so the three possible comparisons

$$(a+b)+0_R \sim_R a+(b+0_R),$$

$$(a+0_R)+c \sim_R a+(0_R+c),$$

$$(0_R+b)+c \sim_R 0_R+(b+c)$$

are the same associativity law with one displayed argument equal to 0_R. If only the left input has a head a, the comparison is

$$(a+0_R)+0_R \sim_R a+0_R,$$

which is the additive right-zero law applied to $a+0_R$. If only the middle input has a head b, the comparison is

$$(0_R+b)+0_R \sim_R 0_R+(b+0_R),$$

again additive associativity. If only the right input has a head c, the comparison is

$$0_R+c \sim_R 0_R+(0_R+c),$$

obtained by reversing the additive left-zero law at the endpoint $0_R + c$ using classifier symmetry from cert_R.

In each constructor case, the scalar head comparison and the tail classifier rebuild the cons clause of $\mathsf{ListClassifierSpec}$. These cases exhaust the finite-spine constructors by Definition 36.13. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd associative classified`

Theorem 50.59 (Polynomial raw addition associativity)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, canonical trim trim_R, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive abelian-group laws and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))).$$

Proof. Set

$$u := \mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \qquad \omega := \mathsf{radd}_R(p, \mathsf{radd}_R(q, r)).$$

By Definition 50.152,

$$\mathsf{ListClassifierSpec}(\sim_R, \nu, \omega)$$

holds. The nil spine is a classified zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 to the common classified spines ν and ω, with nil zero remainders on both sides. The append-nil identities for finite spines give

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\nu), \mathsf{trim}_R(\omega)).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30 turns this normalized structural classifier into the displayed polynomial classifier comparison. □

Theorem 50.60 (Polynomial raw multiplication commutativity from scalar commutativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p, q), \mathsf{rmul}_R(q, p)).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul commutative classified`

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The retained rows used below are the $\mathsf{NameCert}$ reflexivity, symmetry, and transitivity fields, the additive abelian-group laws and addition congruence from Definition 43.26, multiplication congruence, and the multiplicative commutativity row from Definition 44.33.

First record the finite-spine consequences of the existing raw-addition package. The relation $\mathsf{PolySame}_R$ is reflexive, symmetric, and transitive after unfolding Definition 50.30; this is exactly structural list-classifier reflexivity, Definition 36.47, and Definition 36.15 applied to the two trimmed representatives.

Raw addition is compatible with $\mathsf{PolySame}_R$ in both inputs. Indeed, Definition 50.32 reduces both raw sums to sums of their trimmed inputs. The two displayed trimmed-input classifiers give a structural classifier between the two trimmed raw sums by induction on the two list-classifier witnesses in Definition 36.12; the exposed heads are handled by addition congruence, with $0_R \sim_R 0_R$ in the padding branches. Transitivity and symmetry of $\mathsf{PolySame}_R$ compose these three comparisons.

The shift operation preserves $\mathsf{PolySame}_R$. If $\mathsf{PolySame}_R(x, y)$ holds, write the trim decompositions supplied by Definition 50.22 as $x = x_0{+}{+}z$ and $y = y_0{+}{+}w$, with $\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0)$ and zero remainders z, w. Then

$$\mathsf{shift}_R(x) = (\mathsf{cons}(0_R, x_0)){+}{+}z, \qquad \mathsf{shift}_R(y) = (\mathsf{cons}(0_R, y_0)){+}{+}w.$$

The cons classifier built from $0_R \sim_R 0_R$ and the classifier on x_0, y_0, followed by Definition 50.29, gives $\mathsf{PolySame}_R(\mathsf{shift}_R(x), \mathsf{shift}_R(y))$. The same argument, with 0_R replaced by classified heads $h \sim_R h'$, shows that $h \sim_R h'$ and $\mathsf{PolySame}_R(t, t')$ imply

$$\mathsf{PolySame}_R(\mathsf{cons}(h, t), \mathsf{cons}(h', t')).$$

Zero remainders are closed under raw addition: induct on the two zero-remainder witnesses from Definition 50.28. In the cons-cons branch, addition congruence sends $a + b$ to $0_R + 0_R$, and the additive left-zero law reduces that endpoint to 0_R; the cons-nil and nil-cons branches use the additive right-zero and left-zero laws. Hence $\mathsf{rmul}_R(s, \mathsf{nil})$ is a zero remainder for every finite spine s, by induction on s using Definition 50.139: the cons step combines $\mathsf{rscale}_R(a, \mathsf{nil}) = \mathsf{nil}$, the zero-remainder closure of shift_R, and raw-addition closure. Applying Definition 50.29 with common spine nil gives

$$\mathsf{PolySame}_R(\mathsf{nil}, \mathsf{rmul}_R(s, \mathsf{nil})).$$

The key auxiliary recursion is right-cons expansion: for every scalar coefficient a and finite spines x, q,

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(q, \mathsf{cons}(a, x)), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))).$$

Induct on q. If $q = \mathsf{nil}$, the left side is nil and the right side is a raw sum of zero-remainder spines, so the zero-remainder comparison above applies. For the cons step write $q = \mathsf{cons}(b, y)$. Unfolding Definition 50.139 and the cons-cons branch of Definition 50.30, the two heads are $b \cdot a + 0_R$ and $a \cdot b + 0_R$; multiplicative commutativity and addition congruence classify them. The two tails are

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{rmul}_R(y, \mathsf{cons}(a, x)))$$

and

$$\mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

The induction hypothesis and raw-addition compatibility compare the first tail with

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

Raw-addition associativity from Definition 50.153 and raw-addition commutativity from Definition 50.151, transported through raw-addition compatibility, move the first two summands past each other and give the second tail. The cons preservation fact then proves the cons step.

Now induct on p. If $p = \mathsf{nil}$, then $\mathsf{rmul}_R(\mathsf{nil}, q) = \mathsf{nil}$ by Definition 50.139, while $\mathsf{rmul}_R(q, \mathsf{nil})$ is a zero remainder by the preceding paragraph; hence the displayed $\mathsf{PolySame}_R$ holds.

For the cons step write $p = \mathsf{cons}(a, x)$ and assume

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, q), \mathsf{rmul}_R(q, x)).$$

Unfolding the left raw product gives

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), q) = \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(x, q))).$$

Shift preservation and raw-addition compatibility transport the induction hypothesis to

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(a,x),q),\mathsf{radd}_R(\mathsf{rscale}_R(a,q),\mathsf{shift}_R(\mathsf{rmul}_R(q,x)))).$$

Right-cons expansion applied to q, a, x gives the same middle endpoint classified with $\mathsf{rmul}_R(q,\mathsf{cons}(a,x))$. Symmetry and transitivity of $\mathsf{PolySame}_R$ compose the two comparisons, completing the induction. □

Theorem 50.61 (Polynomial multiplication has left zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and zero endpoint 0_R. Let trim_R be the canonical polynomial trim from Definition 50.20 and Definition 50.22, and let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil},q),\mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative q from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left zero classified`

Proof. Set $q_0 = \mathsf{trim}_R(q)$. The empty zero-polynomial clause in Definition 50.22 gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Hence Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil},q) &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{nil}),q_0))\\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil},q_0))\\ &= \mathsf{trim}_R(\mathsf{nil})\\ &= \mathsf{nil}.\end{aligned}$$

The third equality is the nil clause for rmul_R in Definition 50.139.

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R,\mathsf{trim}_R(\mathsf{PolyMul}_R(\mathsf{nil},q)),\mathsf{trim}_R(\mathsf{nil})).$$

The displayed computation and the empty zero-polynomial clause reduce both displayed representatives to nil. The nil-nil clause of Definition 36.12 supplies the structural classifier witness. □

Definition 50.62 (Padded polynomial coefficients and finite Cauchy folds)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and finite coefficient spines from Definition 36.1. For a finite coefficient spine $x : \mathsf{ListCarrier}(R)$ and $n : \mathsf{Nat}^{\uparrow}$, write $x[n]$ for the padded coefficient defined by*

$$\mathsf{nil}[n] := 0_R, \qquad \mathsf{cons}(a,t)[0] := a, \qquad \mathsf{cons}(a,t)[n+1] := t[n].$$

For finite coefficient spines x, y, z define the finite Cauchy coefficient fold and the finite triple coefficient fold by

$$\mathsf{Cauchy}_R(x,y;n) := \Sigma_{i+j=n} x[i]\cdot y[j], \qquad \mathsf{Triple}_R(x,y,z;n) := \Sigma_{i+j+k=n}(x[i]\cdot y[j])\cdot z[k].$$

The displayed sums use the finite additive fold of Definition 50.26 over the lexicographic finite spines of the displayed natural-number tuples.

Theorem 50.63 (Polynomial multiplication has right zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R. Let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative p from Definition 50.3.

Proof. Set

$$p_0 := \mathsf{trim}_R(p), \qquad n_0 := \mathsf{trim}_R(\mathsf{nil}).$$

The empty zero-polynomial clause in Definition 50.22 gives $n_0 = \mathsf{nil}$. Unfolding the normalized product in Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(p, \mathsf{nil}) &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, n_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}))\end{aligned}$$

and

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil}, p) &= \mathsf{trim}_R(\mathsf{rmul}_R(n_0, p_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0)).\end{aligned}$$

Apply Definition 50.154 to the finite coefficient spines p_0 and nil. It gives

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}), \mathsf{rmul}_R(\mathsf{nil}, p_0)).$$

After unfolding $\mathsf{PolySame}_R$ by Definition 50.30, this is the structural classifier comparison

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil})), \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0))).$$

Using the two displayed product computations and trim idempotence from Definition 50.22, this is exactly

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{PolyMul}_R(\mathsf{nil}, p)).$$

On the other hand, Definition 50.155 applied to p gives

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, p), \mathsf{nil}).$$

Unfold the two $\mathsf{PolySame}_R$ witnesses and compose their structural list-classifier witnesses by Definition 36.15, using the transitivity field of cert_R on scalar heads. Folding back Definition 50.30 yields

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Every normalized polynomial representative from Definition 50.3 is, in particular, a finite coefficient spine, so the same argument applies to that carrier representative. □

Lemma 50.64 (Padded coefficient extensionality for the polynomial classifier)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R from Definition 50.20 and Definition 50.22. If finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ satisfy $x[n] \sim_R y[n]$ for every $n : \mathsf{Nat}^{\uparrow}$, then*

$$\mathsf{PolySame}_R(x, y).$$

Proof. Put $x_0 := \mathsf{trim}_R(x)$ and $y_0 := \mathsf{trim}_R(y)$. The trailing-zero removal clause in Definition 50.22 gives decompositions $x = x_0{+}{+}t_x$ and $y = y_0{+}{+}t_y$, with $\mathsf{ZeroRem}_R(t_x)$ and $\mathsf{ZeroRem}_R(t_y)$ in the sense of Definition 50.28. Hence $x[n] \sim_R x_0[n]$ and $y[n] \sim_R y_0[n]$ for every n: inside the trimmed prefix the coefficient is unchanged, inside the removed tail the zero-remainder witness classifies the coefficient with 0_R, and after the whole spine both sides are the padding coefficient 0_R. Classifier symmetry and transitivity from cert_R give $x_0[n] \sim_R y_0[n]$ for every n.

The trimmed spines are normalized representatives. If one normalized representative had a last displayed coefficient beyond the length of the other, the padded coefficient comparison at that last index would classify that displayed coefficient with 0_R, contradicting the normalized no-trailing-zero condition in Definition 50.22. Thus x_0 and y_0 have the same finite length. Induction over this common length builds a structural classifier witness: at each head position the hypothesis $x_0[n] \sim_R y_0[n]$ supplies the scalar head comparison, and the induction hypothesis supplies the tail classifier. Therefore

$$\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0).$$

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30 gives the displayed polynomial classifier comparison. □

Lemma 50.65 (Raw Cauchy product coefficient formula)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(x, y)[n] \sim_R \mathsf{Cauchy}_R(x, y; n).$$

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The recursion defining radd_R gives the padded coefficient formula $\mathsf{radd}_R(u, v)[n] \sim_R u[n] + v[n]$ by induction on the two finite spines. The definitions of rscale_R and shift_R give

$$\mathsf{rscale}_R(a, u)[n] \sim_R a \cdot u[n], \qquad \mathsf{shift}_R(u)[0] \sim_R 0_R, \qquad \mathsf{shift}_R(u)[n+1] \sim_R u[n].$$

These comparisons use only zero self-classification, additive congruence, and the additive zero laws from Definition 43.26.

Induct on the first spine x. If $x = \mathsf{nil}$, the raw product is nil by Definition 50.139. Every summand in $\mathsf{Cauchy}_R(\mathsf{nil}, y; n)$ has the form $0_R \cdot y[j]$ and is classified with 0_R by the zero-multiplication row in Definition 43.26. The zero-summand fold theorem Definition 50.27 classifies the Cauchy fold with 0_R, which is the padded coefficient of nil.

For the cons step, write $x = \mathsf{cons}(a, x')$. At coefficient 0, the raw product recursion gives the sum of $a \cdot y[0]$ and the zero head of the shifted recursive product; the additive right-zero law classifies this with $a \cdot y[0]$, which is $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; 0)$. At coefficient $n + 1$, the same recursion and the auxiliary coefficient formulas give

$$\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)[n+1] \sim_R a \cdot y[n+1] + \mathsf{rmul}_R(x', y)[n].$$

The induction hypothesis and additive congruence replace the second summand by $\mathsf{Cauchy}_R(x', y; n)$. Splitting the finite index spine $i + j = n + 1$ into the head case $i = 0$ and the shifted cases $i = i' + 1$ identifies the resulting fold with $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; n + 1)$ by Definition 50.26. □

Lemma 50.66 (Raw Cauchy product coefficient associativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw Cauchy product data of Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(\mathsf{rmul}_R(x, y), z)[n] \sim_R \mathsf{rmul}_R(x, \mathsf{rmul}_R(y, z))[n].$$

Proof. Fix n. By Definition 50.159, the left coefficient is classified with $\mathsf{Cauchy}_R(\mathsf{rmul}_R(x,y),z;n)$. For each outer index $s+k=n$, apply Definition 50.159 to replace $\mathsf{rmul}_R(x,y)[s]$ by $\mathsf{Cauchy}_R(x,y;s)$. Multiplication congruence and Definition 50.26 lift these replacements through the outer finite fold. The right-multiplication half of Definition 49.95 then classifies $\mathsf{Cauchy}_R(x,y;s)\cdot z[k]$ with the inner fold over (i,j) satisfying $i+j=s$ of $(x[i]\cdot y[j])\cdot z[k]$. The nested finite index spine $i+j=s$, $s+k=n$ is the same finite triple spine $i+j+k=n$ in lexicographic order after deleting the bookkeeping index s. The finite fold swap certificate of Definition 49.94, together with fold congruence, classifies the resulting nested fold with $\mathsf{Triple}_R(x,y,z;n)$. Thus the left raw product coefficient is classified with $\mathsf{Triple}_R(x,y,z;n)$.

The right coefficient is handled symmetrically. The coefficient formula first gives $\mathsf{Cauchy}_R(x,\mathsf{rmul}_R(y,z);n)$. Replacing each coefficient of $\mathsf{rmul}_R(y,z)$ by its Cauchy fold and applying the left-multiplication half of Definition 49.95 gives the finite fold over triples $i+j+k=n$ of $x[i]\cdot(y[j]\cdot z[k])$. Scalar multiplication associativity from Definition 43.26 classifies each displayed summand with $(x[i]\cdot y[j])\cdot z[k]$. Fold congruence therefore classifies the right coefficient with the same $\mathsf{Triple}_R(x,y,z;n)$. Symmetry and transitivity of the scalar classifier compose the two comparisons. □

Theorem 50.67 (Polynomial multiplication associativity over commutative-ring scalars)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the canonical trim be trim_R from Definition 50.22, and let rmul_R and $\mathsf{PolyMul}_R$ be the raw and normalized products of Definition 50.139. For finite coefficient spines $p,q,r:\mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p,q),r),\mathsf{PolyMul}_R(p,\mathsf{PolyMul}_R(q,r))).$$

Proof. Put $p_0:=\mathsf{trim}_R(p)$, $q_0:=\mathsf{trim}_R(q)$, and $r_0:=\mathsf{trim}_R(r)$. Set $A:=\mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p,q),r)$ and $B:=\mathsf{PolyMul}_R(p,\mathsf{PolyMul}_R(q,r))$. By Definition 50.158, it is enough to prove $A[n]\sim_R B[n]$ for every $n:\mathsf{Nat}^{\uparrow}$.

Trim preserves padded coefficients by the trailing-zero decomposition in Definition 50.22: for every finite spine u and every n, $u[n]\sim_R \mathsf{trim}_R(u)[n]$. The coefficient formula Definition 50.159 is functorial in both inputs with respect to such pointwise padded-coefficient comparisons, since multiplication congruence compares each Cauchy summand and Definition 50.26 lifts the summand comparisons through the finite fold.

Unfolding Definition 50.139 and using trim idempotence from Definition 50.22, the left normalized product has the same padded coefficients as $\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{rmul}_R(p_0,q_0)),r_0)$. By trim preservation and the functoriality just recorded, its nth coefficient is classified with $\mathsf{rmul}_R(\mathsf{rmul}_R(p_0,q_0),r_0)[n]$. Similarly, the right normalized product has the same padded coefficients as $\mathsf{rmul}_R(p_0,\mathsf{trim}_R(\mathsf{rmul}_R(q_0,r_0)))$, and its nth coefficient is classified with $\mathsf{rmul}_R(p_0,\mathsf{rmul}_R(q_0,r_0))[n]$.

Applying Definition 50.160 to p_0,q_0,r_0 gives

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p_0,q_0),r_0)[n]\sim_R \mathsf{rmul}_R(p_0,\mathsf{rmul}_R(q_0,r_0))[n].$$

Classifier symmetry and transitivity compose the two trim comparisons with this raw coefficient associativity comparison, yielding $A[n]\sim_R B[n]$ for the fixed n. Since n was arbitrary, Definition 50.158 gives $\mathsf{PolySame}_R(A,B)$, which is the displayed associativity statement. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul associative addFold classified`

Theorem 50.68 (Polynomial singleton evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient c and every carried scalar point*

α,

$$\mathsf{Eval}_{\alpha,R}(\mathsf{cons}(c, \mathsf{nil})) \sim_R c.$$

Proof. Put $s = \mathsf{cons}(c, \mathsf{nil})$. First compute the unnormalized Horner value. The cons and nil recursion clauses in Definition 50.133 identify

$$\mathsf{eval}_{\alpha,R}(s) = c + \alpha \cdot 0_R.$$

The right zero-absorption half of Definition 43.60 gives

$$\alpha \cdot 0_R \sim_R 0_R.$$

Classifier reflexivity for c and addition congruence from Definition 43.26 give

$$c + \alpha \cdot 0_R \sim_R c + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $c + 0_R \sim_R c$, and transitivity yields

$$\mathsf{eval}_{\alpha,R}(s) \sim_R c.$$

By Definition 50.133, $\mathsf{Eval}_{\alpha,R}(s) = \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(s))$. The one-coefficient trimming boundary in Definition 50.166, read with Definition 50.22, has exactly the empty zero branch and the retained singleton branch. In the empty zero branch, $c \sim_R 0_R$ and $\mathsf{trim}_R(s) = \mathsf{nil}$; the nil Horner equation gives $\mathsf{Eval}_{\alpha,R}(s) = 0_R$, and classifier symmetry gives $0_R \sim_R c$. In the retained singleton branch, $\mathsf{trim}_R(s) = s$, so the raw computation above is the required normalized comparison. These are the two canonical trim branches for a singleton coefficient spine. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval singleton`

Theorem 50.69 (Raw polynomial multiplication left-distributes over raw addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and let radd_R, rscale_R, shift_R, and rmul_R be the raw operations of Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z)), \mathsf{radd}_R(\mathsf{rmul}_R(x, y), \mathsf{rmul}_R(x, z))).$$

Consequently the same classifier comparison holds when x, y, z are normalized polynomial representatives from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left distributes over raw add`

Proof. By Definition 50.158, it is enough to compare padded coefficients. First record the padded coefficient law for raw addition: for all finite coefficient spines u, v and every $m : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u, v)[m] \sim_R u[m] + v[m].$$

This is a direct induction over the two finite spines using the four recursion clauses in Definition 50.30. In the mixed cons-nil and nil-cons cases the missing coefficient is the padding endpoint 0_R; in the nil-nil tail the comparison is between 0_R and $0_R + 0_R$, handled by the additive zero laws and classifier symmetry from cert_R.

Fix $n : \mathsf{Nat}^{\uparrow}$. Let the finite split spine for $i + j = n$ carry three scalar spines

$$A_n(i,j) := x[i] \cdot \mathsf{radd}_R(y,z)[j], \qquad B_n(i,j) := x[i] \cdot y[j], \qquad C_n(i,j) := x[i] \cdot z[j].$$

For each split $i + j = n$, the raw-addition coefficient law gives

$$\mathsf{radd}_R(y,z)[j] \sim_R y[j] + z[j].$$

Multiplication congruence transports this under left multiplication by $x[i]$, and scalar left distributivity from Definition 43.26 gives

$$x[i] \cdot \mathsf{radd}_R(y,z)[j] \sim_R (x[i] \cdot y[j]) + (x[i] \cdot z[j]).$$

The matching split order assembles these scalar comparisons into a structural list-classifier witness from Definition 36.12 between A_n and the pointwise sum spine $B_n + C_n$. Applying Definition 50.26 to this witness gives

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y,z); n) \sim_R \mathsf{fold}_{+,R}(B_n + C_n).$$

The finite fold splitting lemma Definition 49.98 then classifies the right endpoint with the sum of the two separate Cauchy folds:

$$\mathsf{fold}_{+,R}(B_n + C_n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

Thus

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y,z); n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

The raw Cauchy coefficient formula Definition 50.159 gives

$$\mathsf{rmul}_R(x, \mathsf{radd}_R(y,z))[n] \sim_R \mathsf{Cauchy}_R(x, \mathsf{radd}_R(y,z); n).$$

For the other endpoint, the raw-addition coefficient law gives

$$\mathsf{radd}_R(\mathsf{rmul}_R(x,y), \mathsf{rmul}_R(x,z))[n] \sim_R \mathsf{rmul}_R(x,y)[n] + \mathsf{rmul}_R(x,z)[n].$$

Applying Definition 50.159 to the two raw products and then additive congruence gives

$$\mathsf{rmul}_R(x,y)[n] + \mathsf{rmul}_R(x,z)[n] \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

Classifier symmetry and transitivity from cert_R compose the three displayed coefficient comparisons, yielding

$$\mathsf{rmul}_R(x, \mathsf{radd}_R(y,z))[n] \sim_R \mathsf{radd}_R(\mathsf{rmul}_R(x,y), \mathsf{rmul}_R(x,z))[n].$$

Since n was arbitrary, Definition 50.158 gives the displayed $\mathsf{PolySame}_R$ comparison. A normalized polynomial representative is a finite coefficient spine by Definition 50.3, so the final sentence is the same statement restricted to those representatives. □

Theorem 50.70 (Polynomial raw multiplication has singleton-one left identity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p), p).$$

Proof. By Definition 50.158, it is enough to prove equality under the padded scalar classifier at every coefficient index. Unfolding the cons clause and the nil clause of raw multiplication in Definition 50.139 gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p) = \mathsf{radd}_R(\mathsf{rscale}_R(1_R, p), \mathsf{cons}(0_R, \mathsf{nil})).$$

Thus the product is the raw sum of the coefficientwise scalar multiple by 1_R and the one-place shift of the empty recursive product.

First, for every finite coefficient spine u and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u, \mathsf{cons}(0_R, \mathsf{nil}))[n] \sim_R u[n].$$

This is induction on u. If $u = \mathsf{nil}$, the only displayed head is $0_R + 0_R$, classified with 0_R by the additive zero rows in Definition 43.26; all later coefficients are padding coefficients 0_R. If $u = \mathsf{cons}(a, u')$, the head comparison is $(a + 0_R) \sim_R a$ by the additive right-zero row, while the tail comparison is the raw-addition right-nil induction that repeatedly classifies $b + 0_R$ with b and ends at the padding coefficient. These steps use additive congruence, zero self-classification, and classifier transitivity from cert_R.

Second, for every finite coefficient spine p and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{rscale}_R(1_R, p)[n] \sim_R p[n].$$

Induct on p. In the nil case both padded coefficients are 0_R. In the cons case, the exposed head is $1_R \cdot a$, classified with a by the multiplicative left-unit row retained in Definition 43.26, and the tail is supplied by the induction hypothesis. After the displayed spine ends, both sides are again the padding coefficient 0_R.

Apply the first coefficient comparison to $u = \mathsf{rscale}_R(1_R, p)$ and compose it with the second comparison. For every $n : \mathsf{Nat}^{\uparrow}$ this gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p)[n] \sim_R p[n].$$

The padded coefficient extensionality lemma Definition 50.158 converts these pointwise scalar comparisons into the displayed $\mathsf{PolySame}_R$ statement. □

Theorem 50.71 (Raw polynomial multiplication associativity up to add-trim)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, the canonical trim be trim_R from Definition 50.22, and let $\mathsf{PolySame}_R$ and rmul_R be the add-trim classifier and raw Cauchy product of Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r), \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r))).$$

Consequently the same comparison holds when p, q, r are normalized polynomial representatives from Definition 50.3.

Proof. Set

$$A := \mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r), \qquad B := \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r)).$$

By Definition 50.158, it is enough to prove $A[n] \sim_R B[n]$ for every $n : \mathsf{Nat}^{\uparrow}$.

Fix $n : \mathsf{Nat}^{\uparrow}$. The raw Cauchy coefficient associativity lemma Definition 50.160, applied to the finite coefficient spines p, q, r, gives exactly

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r)[n] \sim_R \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r))[n].$$

This is $A[n] \sim_R B[n]$. Since n was arbitrary, Definition 50.158 converts the pointwise padded coefficient comparisons into

$$\mathsf{PolySame}_R(A, B).$$

Unfolding the definitions of A and B gives the displayed add-trim classifier comparison. A normalized polynomial representative from Definition 50.3 is a finite coefficient spine, so the final sentence is the same statement restricted to those carried representatives. □

Proposition 50.72 (Literal polynomial add-trim equality fails at cancelling singletons)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be as in Definition 50.30. Suppose $\neg(a \sim_R 0_R)$, $\neg(b \sim_R 0_R)$, and $(a + b) \sim_R 0_R$. Write $[c] := \mathsf{cons}(c, \mathsf{nil})$, and set $p := [a]$ and $q := [b]$. Then literal finite-spine equality does not commute with raw addition and trimming:*

$$\begin{aligned} \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &= \mathsf{nil}, \\ \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) &= [a + b], \\ \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &\neq \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)). \end{aligned}$$

Nevertheless the two displayed outputs are polynomial-classified the same:

$$\mathsf{PolySame}_R(\mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q))).$$

For the canonical integer coefficient ring from chapter 31 and chapter 44, taking $a = 1$ and $b = -1$ gives $p = [1]$, $q = [-1]$, $\mathsf{trim}_{\mathbb{Z}}(\mathsf{radd}_{\mathbb{Z}}(p, q)) = \mathsf{nil}$, and $\mathsf{radd}_{\mathbb{Z}}(\mathsf{trim}_{\mathbb{Z}}(p), \mathsf{trim}_{\mathbb{Z}}(q)) = [0]$.

Proof. The nonzero hypotheses make p and q normalized singleton representatives by Definition 50.166; the canonical trim fixes normalized representatives, so $\mathsf{trim}_R(p) = p$ and $\mathsf{trim}_R(q) = q$. By the cons-cons and nil-nil clauses of radd_R in Definition 50.30, $\mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) = \mathsf{radd}_R([a], [b]) = [a + b]$. The equality $\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}$ is the singleton cancellation conclusion of Definition 50.167. The displayed inequality follows because the left side is nil while the right side is a cons spine. Constructor no-confusion for the finite-spine carrier of Definition 36.1 is the mixed nil/cons separation isolated in Definition 36.13 and Definition 36.14. For classifier comparison, unfold $\mathsf{PolySame}_R$. The first argument trims again to nil. The second argument is $[a + b]$, and the hypothesis $(a + b) \sim_R 0_R$ sends its singleton coefficient to the trailing-zero branch, so its trim is also nil. The nil-nil clause of $\mathsf{ListClassifierSpec}$ gives the displayed $\mathsf{PolySame}_R$ witness. The integer specialization uses the standard integer facts $1 \neq 0$, $-1 \neq 0$, and $1 + (-1) = 0$; Definition 44.8 keeps the ring operations and classifier when the commutative-ring source is read through $\mathsf{Ring}^{\uparrow}$. □

Definition 50.73 (Polynomial Horner evaluation)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a carried scalar point α, define the Horner evaluation on finite coefficient spines from Definition 36.1 by recursion*

$$\mathsf{eval}_{\alpha,R}(\mathsf{nil}) := 0_R.$$

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a,x)) := a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Define the normalized polynomial evaluation by

$$\mathsf{Eval}_{\alpha,R}(x) := \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(x)).$$

Define the normalized polynomial sum by

$$\mathsf{PolyAdd}_R(x,y) := \mathsf{trim}_R(\mathsf{radd}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

On a normalized representative from Definition 50.3, $\mathsf{Eval}_{\alpha,R}$ *is the Horner evaluation itself, by trim idempotence from Definition 50.22.*

Lemma 50.74 (Horner evaluation ignores classified zero tails)**.** *Let* R *carry a* $\mathsf{CommRing}^{\uparrow}(R)$ *certificate, read as a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate by Definition 44.8. Fix a carried scalar point* α*. For finite coefficient spines* $x, t : \mathsf{ListCarrier}(R)$*, if* $\mathsf{ZeroRem}_R(t)$ *holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{\alpha,R}(x{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Consequently, for every finite coefficient spine x*,*

$$\mathsf{Eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder addFold empty`

Proof. First prove the endpoint claim $\mathsf{eval}_{\alpha,R}(t) \sim_R 0_R$ for every classified zero tail t. Induct on the displayed $\mathsf{ZeroRem}_R(t)$ witness. The nil clause is the defining equation $\mathsf{eval}_{\alpha,R}(\mathsf{nil}) = 0_R$ together with zero self-classification from Definition 43.26. In the cons clause, write $t = \mathsf{cons}(a, t')$. The constructor gives $a \sim_R 0_R$ and the induction hypothesis gives $\mathsf{eval}_{\alpha,R}(t') \sim_R 0_R$. Multiplication congruence and the zero-multiplication row in Definition 43.26 classify $\alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with 0_R. Addition congruence then classifies $a + \alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with $0_R + 0_R$, and the additive left-zero law reduces the latter endpoint to 0_R.

For the displayed tail-invariance statement, induct on x. The nil case is the endpoint claim just proved. In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives $\mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports this comparison through left multiplication by α, and addition congruence with source reflexivity on a gives

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x').$$

The two Horner recursion equations identify these endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'{+}{+}t))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

For the consequence, the trailing-zero removal clause in Definition 50.22 presents each raw spine x as $\mathsf{trim}_R(x){+}{+}t$ with $\mathsf{ZeroRem}_R(t)$. Applying the first part gives $\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{Eval}_{\alpha,R}(x)$, and symmetry of the scalar classifier reverses the displayed comparison. □

Lemma 50.75 (Horner evaluation respects raw polynomial addition)**.** *Let* R *carry a* $\mathsf{CommRing}^{\uparrow}(R)$ *certificate, read as a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate by Definition 44.8. Fix a carried scalar point* α*. For finite coefficient spines* $x, y : \mathsf{ListCarrier}(R)$*,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x,y)) \sim_R \mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold append hsame`

Proof. Perform induction on the two finite spines, following the four recursion clauses for radd_R in Definition 50.30.

The nil-nil case compares 0_R with $0_R + 0_R$, so it is exactly the additive left-zero law, read with symmetry when needed.

In the cons-nil case, write $x = \mathsf{cons}(a, x')$. The raw-addition recursion exposes the head $a + 0_R$ and the tail $\mathsf{radd}_R(x', \mathsf{nil})$. The induction hypothesis classifies the tail evaluation with $\mathsf{eval}_{\alpha,R}(x') + 0_R$, and Definition 43.31 reduces that endpoint to $\mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports the tail comparison through left multiplication by α. Additive congruence and the additive right-zero law reduce the exposed head $a + 0_R$ to a, and the target endpoint $(\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))) + 0_R$ to $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

The nil-cons case is symmetric. The raw head is $0_R + b$, which reduces to b by the additive left-zero law, and the outer target $0_R + \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ reduces by the same law. The tail comparison is transported through multiplication by α exactly as in the cons-nil case.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x', y')) \sim_R \mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y').$$

Multiplication congruence transports this through left multiplication by α, and left distributivity from Definition 43.26 classifies

$$\alpha \cdot (\mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y'))$$

with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence therefore transports the Horner endpoint for $\mathsf{radd}_R(x, y)$ to

$$(a + b) + (\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Apply Definition 42.13 to the additive abelian-group operation, with the four additive blocks a, b, $\alpha \cdot \mathsf{eval}_{\alpha,R}(x')$, and $\alpha \cdot \mathsf{eval}_{\alpha,R}(y')$. This classifies the last endpoint with

$$(a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x')) + (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')),$$

which is exactly $\mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y)$ by the Horner recursion. □

Theorem 50.76 (Polynomial evaluation is additive)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) + \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.133,

$$\mathsf{PolyAdd}_R(p, q) = \mathsf{trim}_R(\mathsf{radd}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies $\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q))$ with $\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0))$. Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)).$$

By Definition 50.135,

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) + \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$, and $\mathsf{Eval}_{\alpha,R}$ on each displayed representative is the unnormalized Horner evaluation. Substituting these equalities gives the carrier formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawAdd classified`

Lemma 50.77 (Horner evaluation respects structural classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$, if*

$$\mathsf{ListClassifierSpec}(\sim_R, x, y)$$

holds, then

$$\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the displayed structural classifier witness from Definition 36.12. By Definition 36.13, the only constructor cases are nil-nil and cons-cons; mixed nil-cons shapes have no classifier witness.

In the nil-nil case, the two Horner equations in Definition 50.133 identify both endpoints with 0_R. The zero self-classification field retained in Definition 43.26 gives $0_R \sim_R 0_R$.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The classifier constructor supplies $a \sim_R b$ and a tail witness $\mathsf{ListClassifierSpec}(\sim_R, x', y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(x') \sim_R \mathsf{eval}_{\alpha,R}(y').$$

The carried point α is self-classified by the scalar certificate. Multiplication congruence from Definition 43.26 transports the tail comparison through left multiplication by α, giving

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence combines this comparison with the head comparison $a \sim_R b$:

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

The cons recursion equation in Definition 50.133 identifies these two endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$, respectively. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval list classifier classified`

Theorem 50.78 (Polynomial evaluation respects the polynomial classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(p, q) \Longrightarrow \mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(q).$$

Proof. Unfold $\mathsf{PolySame}_R$ using Definition 50.30. A classifier witness for p and q is exactly

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p), \mathsf{trim}_R(q)).$$

Apply Definition 50.137 to the two trimmed spines. This gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(q)).$$

The normalized evaluation definition in Definition 50.133 reads the left endpoint as $\mathsf{Eval}_{\alpha,R}(p)$ and the right endpoint as $\mathsf{Eval}_{\alpha,R}(q)$, giving the displayed implication. □

Definition 50.79 (Polynomial raw Cauchy product)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a scalar coefficient a and a finite coefficient spine $y : \mathsf{ListCarrier}(R)$, define scalar multiplication of coefficient spines by*

$$\mathsf{rscale}_R(a, \mathsf{nil}) := \mathsf{nil}, \qquad \mathsf{rscale}_R(a, \mathsf{cons}(b, y)) := \mathsf{cons}(a \cdot b, \mathsf{rscale}_R(a, y)).$$

Define the one-place coefficient shift by

$$\mathsf{shift}_R(y) := \mathsf{cons}(0_R, y).$$

The raw polynomial product is the finite recursion

$$\mathsf{rmul}_R(\mathsf{nil}, y) := \mathsf{nil},$$

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), y) := \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(x, y))).$$

The normalized polynomial product is

$$\mathsf{PolyMul}_R(x, y) := \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

This is the Cauchy product in low-coefficient-first Horner form: the cons clause expresses $(a + Xx)y = ay + X(xy)$.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul`

Lemma 50.80 (Horner evaluation of scalar-multiplied coefficient spines)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every carried scalar coefficient a and every finite coefficient spine $y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) \sim_R a \cdot \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawScale classified`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval raw scale empty classified`

Proof. Induct on the finite spine y from Definition 36.1. In the nil case, both the scalar-multiple spine and its Horner value are 0_R. The target endpoint is $a \cdot 0_R$, and the zero-multiplication row in Definition 43.26, with symmetry of $\sim_R$, gives the displayed comparison.

In the cons case, write $y = \mathsf{cons}(b, y')$. The recursion in Definition 50.139 and the Horner recursion in Definition 50.133 identify the left endpoint with

$$a \cdot b + \alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y')).$$

The induction hypothesis and multiplication congruence classify this endpoint with

$$a \cdot b + \alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Multiplicative associativity, multiplicative commutativity from Definition 44.33, and multiplication congruence classify

$$\alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Addition congruence transports this comparison under the common summand $a \cdot b$. The left distributive row in Definition 43.26, used in the reverse direction, then identifies

$$a \cdot b + a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

The final endpoint is $a \cdot \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ by the Horner recursion, and transitivity composes the comparisons. □

Lemma 50.81 (Horner evaluation of coefficient shift)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(x)) \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Proof. By Definition 50.139, $\mathsf{shift}_R(x) = \mathsf{cons}(0_R, x)$. The Horner recursion from Definition 50.133 makes its evaluation

$$0_R + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

The additive left-zero row in Definition 43.26 classifies this endpoint with $\alpha \cdot \mathsf{eval}_{\alpha,R}(x)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval coefficient shift`

Lemma 50.82 (Horner evaluation respects raw polynomial multiplication)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) \cdot \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the first spine x. In the nil case, $\mathsf{rmul}_R(\mathsf{nil}, y) = \mathsf{nil}$, so the left endpoint evaluates to 0_R. The target endpoint is $0_R \cdot \mathsf{eval}_{\alpha,R}(y)$, and the zero-multiplication row in Definition 43.26, with symmetry, gives the comparison.

For the cons case, write $x = \mathsf{cons}(a, x')$, put $u = \mathsf{eval}_{\alpha,R}(x')$, and put $v = \mathsf{eval}_{\alpha,R}(y)$. The raw product recursion in Definition 50.139 and the raw additivity theorem Definition 50.135 give

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) + \mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(\mathsf{rmul}_R(x', y))).$$

By Definition 50.140, the first summand is classified with $a \cdot v$. By Definition 50.141, the second summand is classified with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x', y)).$$

The induction hypothesis and multiplication congruence classify this with $\alpha \cdot (u \cdot v)$. Multiplicative associativity classifies $\alpha \cdot (u \cdot v)$ with $(\alpha \cdot u) \cdot v$. Addition congruence therefore gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R a \cdot v + (\alpha \cdot u) \cdot v.$$

The right distributive row retained by Definition 44.8, equivalently derived from Definition 44.34, classifies

$$(a + \alpha \cdot u) \cdot v$$

with

$$a \cdot v + (\alpha \cdot u) \cdot v.$$

Using symmetry of this distributive comparison and the Horner recursion

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x')) = a + \alpha \cdot u,$$

transitivity gives the required cons-case endpoint. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawMul classified`

Theorem 50.83 (Polynomial evaluation is multiplicative)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) \cdot \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.139,

$$\mathsf{PolyMul}_R(p, q) = \mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies the normalized evaluation of this endpoint with

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

By Definition 50.142,

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) \cdot \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation in Definition 50.133. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$. The product endpoint is normalized by its defining trim, so normalized evaluation on $\mathsf{PolyMul}_R(p, q)$ is the unnormalized Horner evaluation of that product. Substituting these equalities gives the carrier formula. □

Theorem 50.84 (Polynomial evaluation is a semiring morphism through classification)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q, p', q' : \mathsf{ListCarrier}(R)$, if $\mathsf{PolySame}_R(p, p')$ and $\mathsf{PolySame}_R(q, q')$, then*

$$\mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(p'),$$

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q),$$

and

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval semiring morphism`

Proof. The first clause is exactly Definition 50.138 applied to the classifier witness between p and p'.

The second and third clauses are the normalized-operation laws from Definition 50.136 and Definition 50.143. They apply to the original representatives p and q, so the operation endpoints are already the canonical normalized polynomial operations.

When a later scalar comparison is written with p' or q' instead, apply Definition 50.138 to the appropriate classifier witness and compose by transitivity of $\sim_R$. Thus the descended endpoints may be substituted in the additive and multiplicative scalar comparisons without reassembling the three facts. □

Lemma 50.85 (Zero-point Horner cons evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. The cons recursion in Definition 50.133 identifies the left endpoint with

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t).$$

The zero-absorption theorem Definition 43.60, applied to $\mathsf{eval}_{0_R,R}(t)$, gives

$$0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R 0_R.$$

Classifier reflexivity supplies $a \sim_R a$, and addition congruence from Definition 43.26 gives

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R a + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $a + 0_R \sim_R a$. Classifier transitivity composes the two comparisons. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero cons constant`

Lemma 50.86 (Zero-point Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{0_R,R}(x{+}{+}t) \sim_R \mathsf{eval}_{0_R,R}(x).$$

Proof. First record the endpoint claim that every classified zero remainder evaluates to zero at 0_R. If the zero remainder is nil, the nil recursion in Definition 50.133 identifies its evaluation with 0_R, and zero self-classification from Definition 43.26 gives the claim. If the zero remainder is $\mathsf{cons}(c, t')$, the cons clause of Definition 50.28 supplies $c \sim_R 0_R$. By Definition 50.145,

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R c,$$

and transitivity gives $\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R 0_R$.

Now split on the first constructor of x. If $x = \mathsf{nil}$, then $x{+}{+}t = t$, so the endpoint claim gives the displayed comparison. If $x = \mathsf{cons}(a, x')$, then $x{+}{+}t = \mathsf{cons}(a, x'{+}{+}t)$. Applying Definition 50.145 twice gives

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x'{+}{+}t)) \sim_R a$$

and

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x')) \sim_R a.$$

Symmetry of the second comparison and transitivity give the required comparison for the cons case. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero point zero tail invariance`

Theorem 50.87 (Polynomial evaluation at zero returns the constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. Set $x = \mathsf{cons}(a, t)$. The trailing-zero removal clause in Definition 50.22 gives a finite zero remainder z such that

$$x = \mathsf{trim}_R(x){+}{+}z, \qquad \mathsf{ZeroRem}_R(z).$$

Applying Definition 50.146 to $\mathsf{trim}_R(x)$ and z yields

$$\mathsf{eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(\mathsf{trim}_R(x)).$$

By the normalized evaluation definition in Definition 50.133, the right endpoint is $\mathsf{Eval}_{0_R,R}(x)$. Symmetry gives

$$\mathsf{Eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(x).$$

The raw cons comparison Definition 50.145 gives

$$\mathsf{eval}_{0_R,R}(x) \sim_R a.$$

Classifier transitivity composes these two comparisons, and substituting $x = \mathsf{cons}(a, t)$ gives the displayed formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero constant term`

Lemma 50.88 (Unit-point Horner evaluation is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$,*

and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ *be the finite recursive coefficient fold from Definition 50.26, satisfying*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) = 0_R, \qquad \mathsf{fold}_{+,R}(\mathsf{cons}(a, x')) = a + \mathsf{fold}_{+,R}(x').$$

For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$*,*

$$\mathsf{eval}_{1_R,R}(x) \sim_R \mathsf{fold}_{+,R}(x).$$

Proof. Induct on the finite spine x from Definition 36.1.

In the nil case, the nil Horner equation in Definition 50.133 and the nil fold equation identify both endpoints with 0_R. The zero self-classification row of Definition 43.26 gives the displayed comparison.

In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives

$$\mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

The recursive Horner endpoint $\mathsf{eval}_{1_R,R}(x')$ is carried by induction on x' using the scalar zero, addition, and multiplication closure rows of the $\mathsf{Ring}^{\uparrow}$ certificate. Hence the multiplicative left-unit row retained in Definition 43.26 gives

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{eval}_{1_R,R}(x').$$

Classifier transitivity composes this unit comparison with the induction hypothesis, yielding

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

Source reflexivity gives $a \sim_R a$. Applying additive congruence from Definition 43.26 gives

$$a + 1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R a + \mathsf{fold}_{+,R}(x').$$

The cons Horner equation identifies the left endpoint with $\mathsf{eval}_{1_R,R}(\mathsf{cons}(a, x'))$, and the cons fold equation identifies the right endpoint with $\mathsf{fold}_{+,R}(\mathsf{cons}(a, x'))$. □

Theorem 50.89 (Polynomial evaluation at the multiplicative unit is coefficient fold)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate with classifier* $\sim_R$*, zero endpoint* 0_R*, multiplicative unit endpoint* 1_R*, addition* $+$*, multiplication* $\cdot$*, canonical trim* trim_R*, and Horner evaluation as in Definition 50.133. Let* $\mathsf{fold}_{+,R}$ *be the finite recursive coefficient fold from Definition 50.26. For every finite coefficient spine* $p : \mathsf{ListCarrier}(R)$*,*

$$\mathsf{Eval}_{1_R,R}(p) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Proof. By the normalized evaluation definition in Definition 50.133,

$$\mathsf{Eval}_{1_R,R}(p) = \mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)).$$

Apply Definition 50.148 to the finite coefficient spine $\mathsf{trim}_R(p)$. This gives

$$\mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Substituting the displayed definition of $\mathsf{Eval}_{1_R,R}(p)$ gives the claimed comparison. □

Lemma 50.90 (Polynomial raw addition structural swap)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$. Assume scalar additive commutativity:*

$$a + b \sim_R b + a$$

for all scalar coefficients a, b. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Induct simultaneously on the two finite coefficient spines, following the four recursion clauses of radd_R in Definition 50.30. The target classifier is the finite-spine structural classifier of Definition 36.12, with matching constructors isolated by Definition 36.13.

In the nil-nil case, both raw additions are nil, so the nil clause of $\mathsf{ListClassifierSpec}$ gives the claim.

In the cons-nil case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{nil}$. The recursion gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a + 0_R, \mathsf{radd}_R(p', \mathsf{nil}))$$

and

$$\mathsf{radd}_R(q, p) = \mathsf{cons}(0_R + a, \mathsf{radd}_R(\mathsf{nil}, p')).$$

Scalar additive commutativity gives $a + 0_R \sim_R 0_R + a$. The induction hypothesis applied to p' and nil gives the tail classifier

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', \mathsf{nil}), \mathsf{radd}_R(\mathsf{nil}, p')).$$

The cons clause of $\mathsf{ListClassifierSpec}$ combines the head comparison and this tail comparison.

In the nil-cons case, write $p = \mathsf{nil}$ and $q = \mathsf{cons}(b, q')$. The recursion gives heads $0_R + b$ and $b + 0_R$. Scalar additive commutativity gives $0_R + b \sim_R b + 0_R$, and the induction hypothesis at nil and q' supplies the tail classifier. The cons clause rebuilds the displayed list-classifier witness.

In the cons-cons case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{cons}(b, q')$. The recursion exposes heads $a + b$ and $b + a$, related by scalar additive commutativity. The induction hypothesis supplies

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', q'), \mathsf{radd}_R(q', p')).$$

The cons clause combines the head and tail witnesses, giving the structural classifier for the two raw sums. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd structural swap`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd commutative classified`

Theorem 50.91 (Polynomial raw addition commutativity from scalar additive commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Use the additive abelian-group and classifier fields retained in Definition 43.26: zero self-classification, classifier transitivity, additive congruence, additive zero laws, and scalar additive commutativity. For a commutative-ring scalar source, Definition 44.8 keeps these data unchanged. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Apply Definition 50.150 to p and q using the scalar additive commutativity row from Definition 43.26. This gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p,q), \mathsf{radd}_R(q,p)).$$

The nil finite spine is a zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 with common classified spines $\mathsf{radd}_R(p,q)$ and $\mathsf{radd}_R(q,p)$, and with both zero remainders equal to nil. Finite-spine append by nil leaves each spine unchanged, so the lemma yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(p,q)), \mathsf{trim}_R(\mathsf{radd}_R(q,p))).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, this is exactly the displayed polynomial raw classifier comparison. □

Lemma 50.92 (Raw polynomial addition is structurally associative)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive associativity, additive zero laws, classifier symmetry, classifier transitivity, and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(\mathsf{radd}_R(p,q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q,r))).$$

Proof. Induct on the finite triple spine (p, q, r), using the maximum of the three finite lengths as the decreasing measure. The nil-nil-nil case unfolds both raw additions by Definition 50.30 to nil, and the nil clause of Definition 36.12 gives the classifier witness.

In every remaining constructor case, at least one of the three spines is a cons spine. Unfold the corresponding clauses of radd_R from Definition 50.30. Both associated raw sums expose a cons head, and their tails are exactly the two associated raw sums for the tail triple, with nil used as the tail of any absent spine. The induction hypothesis supplies the recursive tail classifier.

It remains to classify the exposed scalar heads. If all three heads are present, say a, b, c, the comparison is

$$(a+b)+c \sim_R a+(b+c),$$

which is scalar additive associativity. If exactly two heads are present, the missing head is the padding coefficient 0_R, so the three possible comparisons

$$(a+b)+0_R \sim_R a+(b+0_R),$$

$$(a+0_R)+c \sim_R a+(0_R+c),$$

$$(0_R+b)+c \sim_R 0_R+(b+c)$$

are the same associativity law with one displayed argument equal to 0_R. If only the left input has a head a, the comparison is

$$(a+0_R)+0_R \sim_R a+0_R,$$

which is the additive right-zero law applied to $a+0_R$. If only the middle input has a head b, the comparison is

$$(0_R+b)+0_R \sim_R 0_R+(b+0_R),$$

again additive associativity. If only the right input has a head c, the comparison is

$$0_R+c \sim_R 0_R+(0_R+c),$$

obtained by reversing the additive left-zero law at the endpoint $0_R + c$ using classifier symmetry from cert_R.

In each constructor case, the scalar head comparison and the tail classifier rebuild the cons clause of $\mathsf{ListClassifierSpec}$. These cases exhaust the finite-spine constructors by Definition 36.13. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd associative classified`

Theorem 50.93 (Polynomial raw addition associativity)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, canonical trim trim_R, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive abelian-group laws and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))).$$

Proof. Set

$$u := \mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \qquad \omega := \mathsf{radd}_R(p, \mathsf{radd}_R(q, r)).$$

By Definition 50.152,

$$\mathsf{ListClassifierSpec}(\sim_R, \nu, \omega)$$

holds. The nil spine is a classified zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 to the common classified spines ν and ω, with nil zero remainders on both sides. The append-nil identities for finite spines give

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\nu), \mathsf{trim}_R(\omega)).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30 turns this normalized structural classifier into the displayed polynomial classifier comparison. □

Theorem 50.94 (Polynomial raw multiplication commutativity from scalar commutativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p, q), \mathsf{rmul}_R(q, p)).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul commutative classified`

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The retained rows used below are the $\mathsf{NameCert}$ reflexivity, symmetry, and transitivity fields, the additive abelian-group laws and addition congruence from Definition 43.26, multiplication congruence, and the multiplicative commutativity row from Definition 44.33.

First record the finite-spine consequences of the existing raw-addition package. The relation $\mathsf{PolySame}_R$ is reflexive, symmetric, and transitive after unfolding Definition 50.30; this is exactly structural list-classifier reflexivity, Definition 36.47, and Definition 36.15 applied to the two trimmed representatives.

Raw addition is compatible with $\mathsf{PolySame}_R$ in both inputs. Indeed, Definition 50.32 reduces both raw sums to sums of their trimmed inputs. The two displayed trimmed-input classifiers give a structural classifier between the two trimmed raw sums by induction on the two list-classifier witnesses in Definition 36.12; the exposed heads are handled by addition congruence, with $0_R \sim_R 0_R$ in the padding branches. Transitivity and symmetry of $\mathsf{PolySame}_R$ compose these three comparisons.

The shift operation preserves $\mathsf{PolySame}_R$. If $\mathsf{PolySame}_R(x, y)$ holds, write the trim decompositions supplied by Definition 50.22 as $x = x_0{+}{+}z$ and $y = y_0{+}{+}w$, with $\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0)$ and zero remainders z, w. Then

$$\mathsf{shift}_R(x) = (\mathsf{cons}(0_R, x_0)){+}{+}z, \qquad \mathsf{shift}_R(y) = (\mathsf{cons}(0_R, y_0)){+}{+}w.$$

The cons classifier built from $0_R \sim_R 0_R$ and the classifier on x_0, y_0, followed by Definition 50.29, gives $\mathsf{PolySame}_R(\mathsf{shift}_R(x), \mathsf{shift}_R(y))$. The same argument, with 0_R replaced by classified heads $h \sim_R h'$, shows that $h \sim_R h'$ and $\mathsf{PolySame}_R(t, t')$ imply

$$\mathsf{PolySame}_R(\mathsf{cons}(h, t), \mathsf{cons}(h', t')).$$

Zero remainders are closed under raw addition: induct on the two zero-remainder witnesses from Definition 50.28. In the cons-cons branch, addition congruence sends $a + b$ to $0_R + 0_R$, and the additive left-zero law reduces that endpoint to 0_R; the cons-nil and nil-cons branches use the additive right-zero and left-zero laws. Hence $\mathsf{rmul}_R(s, \mathsf{nil})$ is a zero remainder for every finite spine s, by induction on s using Definition 50.139: the cons step combines $\mathsf{rscale}_R(a, \mathsf{nil}) = \mathsf{nil}$, the zero-remainder closure of shift_R, and raw-addition closure. Applying Definition 50.29 with common spine nil gives

$$\mathsf{PolySame}_R(\mathsf{nil}, \mathsf{rmul}_R(s, \mathsf{nil})).$$

The key auxiliary recursion is right-cons expansion: for every scalar coefficient a and finite spines x, q,

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(q, \mathsf{cons}(a, x)), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))).$$

Induct on q. If $q = \mathsf{nil}$, the left side is nil and the right side is a raw sum of zero-remainder spines, so the zero-remainder comparison above applies. For the cons step write $q = \mathsf{cons}(b, y)$. Unfolding Definition 50.139 and the cons-cons branch of Definition 50.30, the two heads are $b \cdot a + 0_R$ and $a \cdot b + 0_R$; multiplicative commutativity and addition congruence classify them. The two tails are

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{rmul}_R(y, \mathsf{cons}(a, x)))$$

and

$$\mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

The induction hypothesis and raw-addition compatibility compare the first tail with

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

Raw-addition associativity from Definition 50.153 and raw-addition commutativity from Definition 50.151, transported through raw-addition compatibility, move the first two summands past each other and give the second tail. The cons preservation fact then proves the cons step.

Now induct on p. If $p = \mathsf{nil}$, then $\mathsf{rmul}_R(\mathsf{nil}, q) = \mathsf{nil}$ by Definition 50.139, while $\mathsf{rmul}_R(q, \mathsf{nil})$ is a zero remainder by the preceding paragraph; hence the displayed $\mathsf{PolySame}_R$ holds.

For the cons step write $p = \mathsf{cons}(a, x)$ and assume

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, q), \mathsf{rmul}_R(q, x)).$$

Unfolding the left raw product gives

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), q) = \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(x, q))).$$

Shift preservation and raw-addition compatibility transport the induction hypothesis to

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(a,x),q),\mathsf{radd}_R(\mathsf{rscale}_R(a,q),\mathsf{shift}_R(\mathsf{rmul}_R(q,x)))).$$

Right-cons expansion applied to q, a, x gives the same middle endpoint classified with $\mathsf{rmul}_R(q,\mathsf{cons}(a,x))$. Symmetry and transitivity of $\mathsf{PolySame}_R$ compose the two comparisons, completing the induction. □

Theorem 50.95 (Polynomial multiplication has left zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and zero endpoint 0_R. Let trim_R be the canonical polynomial trim from Definition 50.20 and Definition 50.22, and let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil},q),\mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative q from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left zero classified`

Proof. Set $q_0 = \mathsf{trim}_R(q)$. The empty zero-polynomial clause in Definition 50.22 gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Hence Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil},q) &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{nil}),q_0))\\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil},q_0))\\ &= \mathsf{trim}_R(\mathsf{nil})\\ &= \mathsf{nil}.\end{aligned}$$

The third equality is the nil clause for rmul_R in Definition 50.139.

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R,\mathsf{trim}_R(\mathsf{PolyMul}_R(\mathsf{nil},q)),\mathsf{trim}_R(\mathsf{nil})).$$

The displayed computation and the empty zero-polynomial clause reduce both displayed representatives to nil. The nil-nil clause of Definition 36.12 supplies the structural classifier witness. □

Definition 50.96 (Padded polynomial coefficients and finite Cauchy folds)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and finite coefficient spines from Definition 36.1. For a finite coefficient spine $x : \mathsf{ListCarrier}(R)$ and $n : \mathsf{Nat}^{\uparrow}$, write $x[n]$ for the padded coefficient defined by*

$$\mathsf{nil}[n] := 0_R, \qquad \mathsf{cons}(a,t)[0] := a, \qquad \mathsf{cons}(a,t)[n+1] := t[n].$$

For finite coefficient spines x, y, z define the finite Cauchy coefficient fold and the finite triple coefficient fold by

$$\mathsf{Cauchy}_R(x,y;n) := \Sigma_{i+j=n} x[i]\cdot y[j], \qquad \mathsf{Triple}_R(x,y,z;n) := \Sigma_{i+j+k=n}(x[i]\cdot y[j])\cdot z[k].$$

The displayed sums use the finite additive fold of Definition 50.26 over the lexicographic finite spines of the displayed natural-number tuples.

Theorem 50.97 (Polynomial multiplication has right zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R. Let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative p from Definition 50.3.

Proof. Set

$$p_0 := \mathsf{trim}_R(p), \qquad n_0 := \mathsf{trim}_R(\mathsf{nil}).$$

The empty zero-polynomial clause in Definition 50.22 gives $n_0 = \mathsf{nil}$. Unfolding the normalized product in Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(p, \mathsf{nil}) &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, n_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}))\end{aligned}$$

and

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil}, p) &= \mathsf{trim}_R(\mathsf{rmul}_R(n_0, p_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0)).\end{aligned}$$

Apply Definition 50.154 to the finite coefficient spines p_0 and nil. It gives

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}), \mathsf{rmul}_R(\mathsf{nil}, p_0)).$$

After unfolding $\mathsf{PolySame}_R$ by Definition 50.30, this is the structural classifier comparison

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil})), \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0))).$$

Using the two displayed product computations and trim idempotence from Definition 50.22, this is exactly

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{PolyMul}_R(\mathsf{nil}, p)).$$

On the other hand, Definition 50.155 applied to p gives

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, p), \mathsf{nil}).$$

Unfold the two $\mathsf{PolySame}_R$ witnesses and compose their structural list-classifier witnesses by Definition 36.15, using the transitivity field of cert_R on scalar heads. Folding back Definition 50.30 yields

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Every normalized polynomial representative from Definition 50.3 is, in particular, a finite coefficient spine, so the same argument applies to that carrier representative. □

Lemma 50.98 (Polynomial raw addition structural swap)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$. Assume scalar additive commutativity:*

$$a + b \sim_R b + a$$

for all scalar coefficients a, b. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Induct simultaneously on the two finite coefficient spines, following the four recursion clauses of radd_R in Definition 50.30. The target classifier is the finite-spine structural classifier of Definition 36.12, with matching constructors isolated by Definition 36.13.

In the nil-nil case, both raw additions are nil, so the nil clause of $\mathsf{ListClassifierSpec}$ gives the claim.

In the cons-nil case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{nil}$. The recursion gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a + 0_R, \mathsf{radd}_R(p', \mathsf{nil}))$$

and

$$\mathsf{radd}_R(q, p) = \mathsf{cons}(0_R + a, \mathsf{radd}_R(\mathsf{nil}, p')).$$

Scalar additive commutativity gives $a + 0_R \sim_R 0_R + a$. The induction hypothesis applied to p' and nil gives the tail classifier

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', \mathsf{nil}), \mathsf{radd}_R(\mathsf{nil}, p')).$$

The cons clause of $\mathsf{ListClassifierSpec}$ combines the head comparison and this tail comparison.

In the nil-cons case, write $p = \mathsf{nil}$ and $q = \mathsf{cons}(b, q')$. The recursion gives heads $0_R + b$ and $b + 0_R$. Scalar additive commutativity gives $0_R + b \sim_R b + 0_R$, and the induction hypothesis at nil and q' supplies the tail classifier. The cons clause rebuilds the displayed list-classifier witness.

In the cons-cons case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{cons}(b, q')$. The recursion exposes heads $a + b$ and $b + a$, related by scalar additive commutativity. The induction hypothesis supplies

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', q'), \mathsf{radd}_R(q', p')).$$

The cons clause combines the head and tail witnesses, giving the structural classifier for the two raw sums. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd structural swap`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd commutative classified`

Theorem 50.99 (Polynomial raw addition commutativity from scalar additive commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Use the additive abelian-group and classifier fields retained in Definition 43.26: zero self-classification, classifier transitivity, additive congruence, additive zero laws, and scalar additive commutativity. For a commutative-ring scalar source, Definition 44.8 keeps these data unchanged. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Apply Definition 50.150 to p and q using the scalar additive commutativity row from Definition 43.26. This gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

The nil finite spine is a zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 with common classified spines $\mathsf{radd}_R(p, q)$ and $\mathsf{radd}_R(q, p)$, and with both zero remainders equal to nil. Finite-spine append by nil leaves each spine unchanged, so the lemma yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{trim}_R(\mathsf{radd}_R(q, p))).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, this is exactly the displayed polynomial raw classifier comparison. □

Lemma 50.100 (Raw polynomial addition is structurally associative)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive associativity, additive zero laws, classifier symmetry, classifier transitivity, and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))).$$

Proof. Induct on the finite triple spine (p, q, r), using the maximum of the three finite lengths as the decreasing measure. The nil-nil-nil case unfolds both raw additions by Definition 50.30 to nil, and the nil clause of Definition 36.12 gives the classifier witness.

In every remaining constructor case, at least one of the three spines is a cons spine. Unfold the corresponding clauses of radd_R from Definition 50.30. Both associated raw sums expose a cons head, and their tails are exactly the two associated raw sums for the tail triple, with nil used as the tail of any absent spine. The induction hypothesis supplies the recursive tail classifier.

It remains to classify the exposed scalar heads. If all three heads are present, say a, b, c, the comparison is

$$(a + b) + c \sim_R a + (b + c),$$

which is scalar additive associativity. If exactly two heads are present, the missing head is the padding coefficient 0_R, so the three possible comparisons

$$(a + b) + 0_R \sim_R a + (b + 0_R),$$

$$(a + 0_R) + c \sim_R a + (0_R + c),$$

$$(0_R + b) + c \sim_R 0_R + (b + c)$$

are the same associativity law with one displayed argument equal to 0_R. If only the left input has a head a, the comparison is

$$(a + 0_R) + 0_R \sim_R a + 0_R,$$

which is the additive right-zero law applied to $a + 0_R$. If only the middle input has a head b, the comparison is

$$(0_R + b) + 0_R \sim_R 0_R + (b + 0_R),$$

again additive associativity. If only the right input has a head c, the comparison is

$$0_R + c \sim_R 0_R + (0_R + c),$$

obtained by reversing the additive left-zero law at the endpoint $0_R + c$ using classifier symmetry from cert_R.

In each constructor case, the scalar head comparison and the tail classifier rebuild the cons clause of $\mathsf{ListClassifierSpec}$. These cases exhaust the finite-spine constructors by Definition 36.13. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd associative classified`

Theorem 50.101 (Polynomial raw addition associativity)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, canonical trim trim_R, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive abelian-group laws and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))).$$

Proof. Set

$$u := \mathsf{radd}_R(\mathsf{radd}_R(p,q),r), \qquad \omega := \mathsf{radd}_R(p,\mathsf{radd}_R(q,r)).$$

By Definition 50.152,

$$\mathsf{ListClassifierSpec}(\sim_R,\nu,\omega)$$

holds. The nil spine is a classified zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 to the common classified spines ν and ω, with nil zero remainders on both sides. The append-nil identities for finite spines give

$$\mathsf{ListClassifierSpec}(\sim_R,\mathsf{trim}_R(\nu),\mathsf{trim}_R(\omega)).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30 turns this normalized structural classifier into the displayed polynomial classifier comparison. □

Theorem 50.102 (Polynomial raw multiplication commutativity from scalar commutativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R,\sim_R)$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $p,q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p,q),\mathsf{rmul}_R(q,p)).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul commutative classified`

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The retained rows used below are the $\mathsf{NameCert}$ reflexivity, symmetry, and transitivity fields, the additive abelian-group laws and addition congruence from Definition 43.26, multiplication congruence, and the multiplicative commutativity row from Definition 44.33.

First record the finite-spine consequences of the existing raw-addition package. The relation $\mathsf{PolySame}_R$ is reflexive, symmetric, and transitive after unfolding Definition 50.30; this is exactly structural list-classifier reflexivity, Definition 36.47, and Definition 36.15 applied to the two trimmed representatives.

Raw addition is compatible with $\mathsf{PolySame}_R$ in both inputs. Indeed, Definition 50.32 reduces both raw sums to sums of their trimmed inputs. The two displayed trimmed-input classifiers give a structural classifier between the two trimmed raw sums by induction on the two list-classifier witnesses in Definition 36.12; the exposed heads are handled by addition congruence, with $0_R \sim_R 0_R$ in the padding branches. Transitivity and symmetry of $\mathsf{PolySame}_R$ compose these three comparisons.

The shift operation preserves $\mathsf{PolySame}_R$. If $\mathsf{PolySame}_R(x,y)$ holds, write the trim decompositions supplied by Definition 50.22 as $x = x_0{+}{+}z$ and $y = y_0{+}{+}w$, with $\mathsf{ListClassifierSpec}(\sim_R,x_0,y_0)$ and zero remainders z,w. Then

$$\mathsf{shift}_R(x) = (\mathsf{cons}(0_R,x_0)){+}{+}z, \qquad \mathsf{shift}_R(y) = (\mathsf{cons}(0_R,y_0)){+}{+}w.$$

The cons classifier built from $0_R \sim_R 0_R$ and the classifier on x_0,y_0, followed by Definition 50.29, gives $\mathsf{PolySame}_R(\mathsf{shift}_R(x),\mathsf{shift}_R(y))$. The same argument, with 0_R replaced by classified heads $h \sim_R h'$, shows that $h \sim_R h'$ and $\mathsf{PolySame}_R(t,t')$ imply

$$\mathsf{PolySame}_R(\mathsf{cons}(h,t),\mathsf{cons}(h',t')).$$

Zero remainders are closed under raw addition: induct on the two zero-remainder witnesses from Definition 50.28. In the cons-cons branch, addition congruence sends $a+b$ to 0_R+0_R, and

the additive left-zero law reduces that endpoint to 0_R; the cons-nil and nil-cons branches use the additive right-zero and left-zero laws. Hence $\mathsf{rmul}_R(s, \mathsf{nil})$ is a zero remainder for every finite spine s, by induction on s using Definition 50.139: the cons step combines $\mathsf{rscale}_R(a, \mathsf{nil}) = \mathsf{nil}$, the zero-remainder closure of shift_R, and raw-addition closure. Applying Definition 50.29 with common spine nil gives

$$\mathsf{PolySame}_R(\mathsf{nil}, \mathsf{rmul}_R(s, \mathsf{nil})).$$

The key auxiliary recursion is right-cons expansion: for every scalar coefficient a and finite spines x, q,

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(q, \mathsf{cons}(a, x)), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))).$$

Induct on q. If $q = \mathsf{nil}$, the left side is nil and the right side is a raw sum of zero-remainder spines, so the zero-remainder comparison above applies. For the cons step write $q = \mathsf{cons}(b, y)$. Unfolding Definition 50.139 and the cons-cons branch of Definition 50.30, the two heads are $b \cdot a + 0_R$ and $a \cdot b + 0_R$; multiplicative commutativity and addition congruence classify them. The two tails are

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{rmul}_R(y, \mathsf{cons}(a, x)))$$

and

$$\mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

The induction hypothesis and raw-addition compatibility compare the first tail with

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

Raw-addition associativity from Definition 50.153 and raw-addition commutativity from Definition 50.151, transported through raw-addition compatibility, move the first two summands past each other and give the second tail. The cons preservation fact then proves the cons step.

Now induct on p. If $p = \mathsf{nil}$, then $\mathsf{rmul}_R(\mathsf{nil}, q) = \mathsf{nil}$ by Definition 50.139, while $\mathsf{rmul}_R(q, \mathsf{nil})$ is a zero remainder by the preceding paragraph; hence the displayed $\mathsf{PolySame}_R$ holds.

For the cons step write $p = \mathsf{cons}(a, x)$ and assume

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, q), \mathsf{rmul}_R(q, x)).$$

Unfolding the left raw product gives

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), q) = \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(x, q))).$$

Shift preservation and raw-addition compatibility transport the induction hypothesis to

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(a, x), q), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))).$$

Right-cons expansion applied to q, a, x gives the same middle endpoint classified with $\mathsf{rmul}_R(q, \mathsf{cons}(a, x))$. Symmetry and transitivity of $\mathsf{PolySame}_R$ compose the two comparisons, completing the induction. □

Theorem 50.103 (Polynomial multiplication has left zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and zero endpoint 0_R. Let trim_R be the canonical polynomial trim from Definition 50.20 and Definition 50.22, and let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, q), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative q from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left zero classified`

Proof. Set $q_0 = \mathsf{trim}_R(q)$. The empty zero-polynomial clause in Definition 50.22 gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Hence Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil}, q) &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{nil}), q_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, q_0)) \\ &= \mathsf{trim}_R(\mathsf{nil}) \\ &= \mathsf{nil}.\end{aligned}$$

The third equality is the nil clause for rmul_R in Definition 50.139.

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{PolyMul}_R(\mathsf{nil}, q)), \mathsf{trim}_R(\mathsf{nil})).$$

The displayed computation and the empty zero-polynomial clause reduce both displayed representatives to nil. The nil-nil clause of Definition 36.12 supplies the structural classifier witness. □

Definition 50.104 (Padded polynomial coefficients and finite Cauchy folds)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and finite coefficient spines from Definition 36.1. For a finite coefficient spine $x : \mathsf{ListCarrier}(R)$ and $n : \mathsf{Nat}^{\uparrow}$, write $x[n]$ for the padded coefficient defined by*

$$\mathsf{nil}[n] := 0_R, \qquad \mathsf{cons}(a, t)[0] := a, \qquad \mathsf{cons}(a, t)[n + 1] := t[n].$$

For finite coefficient spines x, y, z define the finite Cauchy coefficient fold and the finite triple coefficient fold by

$$\mathsf{Cauchy}_R(x, y; n) := \Sigma_{i+j=n} x[i] \cdot y[j], \qquad \mathsf{Triple}_R(x, y, z; n) := \Sigma_{i+j+k=n}(x[i] \cdot y[j]) \cdot z[k].$$

The displayed sums use the finite additive fold of Definition 50.26 over the lexicographic finite spines of the displayed natural-number tuples.

Theorem 50.105 (Polynomial multiplication has right zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R. Let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative p from Definition 50.3.

Proof. Set

$$p_0 := \mathsf{trim}_R(p), \qquad n_0 := \mathsf{trim}_R(\mathsf{nil}).$$

The empty zero-polynomial clause in Definition 50.22 gives $n_0 = \mathsf{nil}$. Unfolding the normalized product in Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(p, \mathsf{nil}) &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, n_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}))\end{aligned}$$

and

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil}, p) &= \mathsf{trim}_R(\mathsf{rmul}_R(n_0, p_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0)).\end{aligned}$$

Apply Definition 50.154 to the finite coefficient spines p_0 and nil. It gives

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}), \mathsf{rmul}_R(\mathsf{nil}, p_0)).$$

After unfolding $\mathsf{PolySame}_R$ by Definition 50.30, this is the structural classifier comparison

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil})), \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0))).$$

Using the two displayed product computations and trim idempotence from Definition 50.22, this is exactly

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{PolyMul}_R(\mathsf{nil}, p)).$$

On the other hand, Definition 50.155 applied to p gives

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, p), \mathsf{nil}).$$

Unfold the two $\mathsf{PolySame}_R$ witnesses and compose their structural list-classifier witnesses by Definition 36.15, using the transitivity field of cert_R on scalar heads. Folding back Definition 50.30 yields

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Every normalized polynomial representative from Definition 50.3 is, in particular, a finite coefficient spine, so the same argument applies to that carrier representative. □

Proposition 50.106 (Literal polynomial add-trim equality fails at cancelling singletons)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be as in Definition 50.30. Suppose $\neg(a \sim_R 0_R)$, $\neg(b \sim_R 0_R)$, and $(a + b) \sim_R 0_R$. Write $[c] := \mathsf{cons}(c, \mathsf{nil})$, and set $p := [a]$ and $q := [b]$. Then literal finite-spine equality does not commute with raw addition and trimming:*

$$\begin{aligned}\mathsf{trim}_R(\mathsf{radd}_R(p, q)) &= \mathsf{nil}, \\ \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) &= [a + b], \\ \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &\neq \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)).\end{aligned}$$

Nevertheless the two displayed outputs are polynomial-classified the same:

$$\mathsf{PolySame}_R(\mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q))).$$

For the canonical integer coefficient ring from chapter 31 and chapter 44, taking $a = 1$ and $b = -1$ gives $p = [1]$, $q = [-1]$, $\mathsf{trim}_{\mathbb{Z}}(\mathsf{radd}_{\mathbb{Z}}(p, q)) = \mathsf{nil}$, and $\mathsf{radd}_{\mathbb{Z}}(\mathsf{trim}_{\mathbb{Z}}(p), \mathsf{trim}_{\mathbb{Z}}(q)) = [0]$.

Proof. The nonzero hypotheses make p and q normalized singleton representatives by Definition 50.166; the canonical trim fixes normalized representatives, so $\mathsf{trim}_R(p) = p$ and $\mathsf{trim}_R(q) = q$. By the cons-cons and nil-nil clauses of radd_R in Definition 50.30, $\mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) = \mathsf{radd}_R([a], [b]) = [a + b]$. The equality $\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}$ is the singleton cancellation conclusion of Definition 50.167. The displayed inequality follows because the left side is nil while the right side is a cons spine. Constructor no-confusion for the finite-spine carrier of Definition 36.1 is the mixed nil/cons separation isolated in Definition 36.13 and Definition 36.14. For classifier

comparison, unfold $\mathsf{PolySame}_R$. The first argument trims again to nil. The second argument is $[a+b]$, and the hypothesis $(a+b) \sim_R 0_R$ sends its singleton coefficient to the trailing-zero branch, so its trim is also nil. The nil-nil clause of $\mathsf{ListClassifierSpec}$ gives the displayed $\mathsf{PolySame}_R$ witness. The integer specialization uses the standard integer facts $1 \neq 0$, $-1 \neq 0$, and $1 + (-1) = 0$; Definition 44.8 keeps the ring operations and classifier when the commutative-ring source is read through $\mathsf{Ring}^{\uparrow}$. □

Definition 50.107 (Polynomial Horner evaluation)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a carried scalar point α, define the Horner evaluation on finite coefficient spines from Definition 36.1 by recursion*

$$\mathsf{eval}_{\alpha,R}(\mathsf{nil}) := 0_R.$$

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a,x)) := a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Define the normalized polynomial evaluation by

$$\mathsf{Eval}_{\alpha,R}(x) := \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(x)).$$

Define the normalized polynomial sum by

$$\mathsf{PolyAdd}_R(x,y) := \mathsf{trim}_R(\mathsf{radd}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

On a normalized representative from Definition 50.3, $\mathsf{Eval}_{\alpha,R}$ is the Horner evaluation itself, by trim idempotence from Definition 50.22.

Lemma 50.108 (Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{\alpha,R}(x{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Consequently, for every finite coefficient spine x,

$$\mathsf{Eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder addFold empty`

Proof. First prove the endpoint claim $\mathsf{eval}_{\alpha,R}(t) \sim_R 0_R$ for every classified zero tail t. Induct on the displayed $\mathsf{ZeroRem}_R(t)$ witness. The nil clause is the defining equation $\mathsf{eval}_{\alpha,R}(\mathsf{nil}) = 0_R$ together with zero self-classification from Definition 43.26. In the cons clause, write $t = \mathsf{cons}(a, t')$. The constructor gives $a \sim_R 0_R$ and the induction hypothesis gives $\mathsf{eval}_{\alpha,R}(t') \sim_R 0_R$. Multiplication congruence and the zero-multiplication row in Definition 43.26 classify $\alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with 0_R. Addition congruence then classifies $a + \alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with $0_R + 0_R$, and the additive left-zero law reduces the latter endpoint to 0_R.

For the displayed tail-invariance statement, induct on x. The nil case is the endpoint claim just proved. In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives $\mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports this comparison through left multiplication by α, and addition congruence with source reflexivity on a gives

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x').$$

The two Horner recursion equations identify these endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'{+}{+}t))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

For the consequence, the trailing-zero removal clause in Definition 50.22 presents each raw spine x as $\mathsf{trim}_R(x){+}{+}t$ with $\mathsf{ZeroRem}_R(t)$. Applying the first part gives $\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{Eval}_{\alpha,R}(x)$, and symmetry of the scalar classifier reverses the displayed comparison. □

Lemma 50.109 (Horner evaluation respects raw polynomial addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold append hsame`

Proof. Perform induction on the two finite spines, following the four recursion clauses for radd_R in Definition 50.30.

The nil-nil case compares 0_R with $0_R + 0_R$, so it is exactly the additive left-zero law, read with symmetry when needed.

In the cons-nil case, write $x = \mathsf{cons}(a, x')$. The raw-addition recursion exposes the head $a + 0_R$ and the tail $\mathsf{radd}_R(x', \mathsf{nil})$. The induction hypothesis classifies the tail evaluation with $\mathsf{eval}_{\alpha,R}(x') + 0_R$, and Definition 43.31 reduces that endpoint to $\mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports the tail comparison through left multiplication by α. Additive congruence and the additive right-zero law reduce the exposed head $a + 0_R$ to a, and the target endpoint $(\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))) + 0_R$ to $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

The nil-cons case is symmetric. The raw head is $0_R + b$, which reduces to b by the additive left-zero law, and the outer target $0_R + \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ reduces by the same law. The tail comparison is transported through multiplication by α exactly as in the cons-nil case.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x', y')) \sim_R \mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y').$$

Multiplication congruence transports this through left multiplication by α, and left distributivity from Definition 43.26 classifies

$$\alpha \cdot (\mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y'))$$

with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence therefore transports the Horner endpoint for $\mathsf{radd}_R(x, y)$ to

$$(a + b) + (\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Apply Definition 42.13 to the additive abelian-group operation, with the four additive blocks a, b, $\alpha \cdot \mathsf{eval}_{\alpha,R}(x')$, and $\alpha \cdot \mathsf{eval}_{\alpha,R}(y')$. This classifies the last endpoint with

$$(a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x')) + (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')),$$

which is exactly $\mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y)$ by the Horner recursion. □

Theorem 50.110 (Polynomial evaluation is additive)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p,q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p,q)) \sim_R \mathsf{eval}_{\alpha,R}(p) + \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.133,

$$\mathsf{PolyAdd}_R(p,q) = \mathsf{trim}_R(\mathsf{radd}_R(p_0,q_0)).$$

Trim idempotence from Definition 50.22 identifies $\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p,q))$ with $\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0,q_0))$. Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0,q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0,q_0)).$$

By Definition 50.135,

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0,q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) + \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$, and $\mathsf{Eval}_{\alpha,R}$ on each displayed representative is the unnormalized Horner evaluation. Substituting these equalities gives the carrier formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawAdd classified`

Lemma 50.111 (Horner evaluation respects structural classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$, if*

$$\mathsf{ListClassifierSpec}(\sim_R, x, y)$$

holds, then

$$\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the displayed structural classifier witness from Definition 36.12. By Definition 36.13, the only constructor cases are nil-nil and cons-cons; mixed nil-cons shapes have no classifier witness.

In the nil-nil case, the two Horner equations in Definition 50.133 identify both endpoints with 0_R. The zero self-classification field retained in Definition 43.26 gives $0_R \sim_R 0_R$.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The classifier constructor supplies $a \sim_R b$ and a tail witness $\mathsf{ListClassifierSpec}(\sim_R, x', y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(x') \sim_R \mathsf{eval}_{\alpha,R}(y').$$

The carried point α is self-classified by the scalar certificate. Multiplication congruence from Definition 43.26 transports the tail comparison through left multiplication by α, giving

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence combines this comparison with the head comparison $a \sim_R b$:

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

The cons recursion equation in Definition 50.133 identifies these two endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$, respectively. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval list classifier classified`

Theorem 50.112 (Polynomial evaluation respects the polynomial classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(p, q) \Longrightarrow \mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(q).$$

Proof. Unfold $\mathsf{PolySame}_R$ using Definition 50.30. A classifier witness for p and q is exactly

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p), \mathsf{trim}_R(q)).$$

Apply Definition 50.137 to the two trimmed spines. This gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(q)).$$

The normalized evaluation definition in Definition 50.133 reads the left endpoint as $\mathsf{Eval}_{\alpha,R}(p)$ and the right endpoint as $\mathsf{Eval}_{\alpha,R}(q)$, giving the displayed implication. □

Definition 50.113 (Polynomial raw Cauchy product)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a scalar coefficient a and a finite coefficient spine $y : \mathsf{ListCarrier}(R)$, define scalar multiplication of coefficient spines by*

$$\mathsf{rscale}_R(a, \mathsf{nil}) := \mathsf{nil}, \qquad \mathsf{rscale}_R(a, \mathsf{cons}(b, y)) := \mathsf{cons}(a \cdot b, \mathsf{rscale}_R(a, y)).$$

Define the one-place coefficient shift by

$$\mathsf{shift}_R(y) := \mathsf{cons}(0_R, y).$$

The raw polynomial product is the finite recursion

$$\mathsf{rmul}_R(\mathsf{nil}, y) := \mathsf{nil},$$

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), y) := \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(x, y))).$$

The normalized polynomial product is

$$\mathsf{PolyMul}_R(x, y) := \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

This is the Cauchy product in low-coefficient-first Horner form: the cons clause expresses $(a + Xx)y = ay + X(xy)$.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul`

Lemma 50.114 (Horner evaluation of scalar-multiplied coefficient spines)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every carried scalar coefficient a and every finite coefficient spine $y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) \sim_R a \cdot \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawScale classified`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval raw scale empty classified`

Proof. Induct on the finite spine y from Definition 36.1. In the nil case, both the scalar-multiple spine and its Horner value are 0_R. The target endpoint is $a \cdot 0_R$, and the zero-multiplication row in Definition 43.26, with symmetry of $\sim_R$, gives the displayed comparison.

In the cons case, write $y = \mathsf{cons}(b, y')$. The recursion in Definition 50.139 and the Horner recursion in Definition 50.133 identify the left endpoint with

$$a \cdot b + \alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y')).$$

The induction hypothesis and multiplication congruence classify this endpoint with

$$a \cdot b + \alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Multiplicative associativity, multiplicative commutativity from Definition 44.33, and multiplication congruence classify

$$\alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Addition congruence transports this comparison under the common summand $a \cdot b$. The left distributive row in Definition 43.26, used in the reverse direction, then identifies

$$a \cdot b + a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

The final endpoint is $a \cdot \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ by the Horner recursion, and transitivity composes the comparisons. □

Lemma 50.115 (Horner evaluation of coefficient shift)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(x)) \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Proof. By Definition 50.139, $\mathsf{shift}_R(x) = \mathsf{cons}(0_R, x)$. The Horner recursion from Definition 50.133 makes its evaluation

$$0_R + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

The additive left-zero row in Definition 43.26 classifies this endpoint with $\alpha \cdot \mathsf{eval}_{\alpha,R}(x)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval coefficient shift`

Lemma 50.116 (Horner evaluation respects raw polynomial multiplication)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) \cdot \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the first spine x. In the nil case, $\mathsf{rmul}_R(\mathsf{nil}, y) = \mathsf{nil}$, so the left endpoint evaluates to 0_R. The target endpoint is $0_R \cdot \mathsf{eval}_{\alpha,R}(y)$, and the zero-multiplication row in Definition 43.26, with symmetry, gives the comparison.

For the cons case, write $x = \mathsf{cons}(a, x')$, put $u = \mathsf{eval}_{\alpha,R}(x')$, and put $v = \mathsf{eval}_{\alpha,R}(y)$. The raw product recursion in Definition 50.139 and the raw additivity theorem Definition 50.135 give

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) + \mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(\mathsf{rmul}_R(x', y))).$$

By Definition 50.140, the first summand is classified with $a \cdot v$. By Definition 50.141, the second summand is classified with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x', y)).$$

The induction hypothesis and multiplication congruence classify this with $\alpha \cdot (u \cdot v)$. Multiplicative associativity classifies $\alpha \cdot (u \cdot v)$ with $(\alpha \cdot u) \cdot v$. Addition congruence therefore gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R a \cdot v + (\alpha \cdot u) \cdot v.$$

The right distributive row retained by Definition 44.8, equivalently derived from Definition 44.34, classifies

$$(a + \alpha \cdot u) \cdot v$$

with

$$a \cdot v + (\alpha \cdot u) \cdot v.$$

Using symmetry of this distributive comparison and the Horner recursion

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x')) = a + \alpha \cdot u,$$

transitivity gives the required cons-case endpoint. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawMul classified`

Theorem 50.117 (Polynomial evaluation is multiplicative)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) \cdot \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.139,

$$\mathsf{PolyMul}_R(p, q) = \mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies the normalized evaluation of this endpoint with

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

By Definition 50.142,

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) \cdot \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation in Definition 50.133. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$. The product endpoint is normalized by its defining trim, so normalized evaluation on $\mathsf{PolyMul}_R(p, q)$ is the unnormalized Horner evaluation of that product. Substituting these equalities gives the carrier formula. □

Theorem 50.118 (Polynomial evaluation is a semiring morphism through classification)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q, p', q' : \mathsf{ListCarrier}(R)$, if $\mathsf{PolySame}_R(p, p')$ and $\mathsf{PolySame}_R(q, q')$, then*

$$\mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(p'),$$

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q),$$

and

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval semiring morphism`

Proof. The first clause is exactly Definition 50.138 applied to the classifier witness between p and p'.

The second and third clauses are the normalized-operation laws from Definition 50.136 and Definition 50.143. They apply to the original representatives p and q, so the operation endpoints are already the canonical normalized polynomial operations.

When a later scalar comparison is written with p' or q' instead, apply Definition 50.138 to the appropriate classifier witness and compose by transitivity of $\sim_R$. Thus the descended endpoints may be substituted in the additive and multiplicative scalar comparisons without reassembling the three facts. □

Lemma 50.119 (Zero-point Horner cons evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. The cons recursion in Definition 50.133 identifies the left endpoint with

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t).$$

The zero-absorption theorem Definition 43.60, applied to $\mathsf{eval}_{0_R,R}(t)$, gives

$$0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R 0_R.$$

Classifier reflexivity supplies $a \sim_R a$, and addition congruence from Definition 43.26 gives

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R a + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $a+0_R \sim_R a$. Classifier transitivity composes the two comparisons. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero cons constant`

Lemma 50.120 (Zero-point Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{0_R,R}(x{+}{+}t) \sim_R \mathsf{eval}_{0_R,R}(x).$$

Proof. First record the endpoint claim that every classified zero remainder evaluates to zero at 0_R. If the zero remainder is nil, the nil recursion in Definition 50.133 identifies its evaluation with 0_R, and zero self-classification from Definition 43.26 gives the claim. If the zero remainder is $\mathsf{cons}(c, t')$, the cons clause of Definition 50.28 supplies $c \sim_R 0_R$. By Definition 50.145,

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R c,$$

and transitivity gives $\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R 0_R$.

Now split on the first constructor of x. If $x = \mathsf{nil}$, then $x{+}{+}t = t$, so the endpoint claim gives the displayed comparison. If $x = \mathsf{cons}(a, x')$, then $x{+}{+}t = \mathsf{cons}(a, x'{+}{+}t)$. Applying Definition 50.145 twice gives

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x'{+}{+}t)) \sim_R a$$

and

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x')) \sim_R a.$$

Symmetry of the second comparison and transitivity give the required comparison for the cons case. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero point zero tail invariance`

Theorem 50.121 (Polynomial evaluation at zero returns the constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. Set $x = \mathsf{cons}(a, t)$. The trailing-zero removal clause in Definition 50.22 gives a finite zero remainder z such that

$$x = \mathsf{trim}_R(x){+}{+}z, \qquad \mathsf{ZeroRem}_R(z).$$

Applying Definition 50.146 to $\mathsf{trim}_R(x)$ and z yields

$$\mathsf{eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(\mathsf{trim}_R(x)).$$

By the normalized evaluation definition in Definition 50.133, the right endpoint is $\mathsf{Eval}_{0_R,R}(x)$. Symmetry gives

$$\mathsf{Eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(x).$$

The raw cons comparison Definition 50.145 gives

$$\mathsf{eval}_{0_R,R}(x) \sim_R a.$$

Classifier transitivity composes these two comparisons, and substituting $x = \mathsf{cons}(a, t)$ gives the displayed formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero constant term`

Lemma 50.122 (Unit-point Horner evaluation is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26, satisfying*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) = 0_R, \qquad \mathsf{fold}_{+,R}(\mathsf{cons}(a, x')) = a + \mathsf{fold}_{+,R}(x').$$

For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,

$$\mathsf{eval}_{1_R,R}(x) \sim_R \mathsf{fold}_{+,R}(x).$$

Proof. Induct on the finite spine x from Definition 36.1.

In the nil case, the nil Horner equation in Definition 50.133 and the nil fold equation identify both endpoints with 0_R. The zero self-classification row of Definition 43.26 gives the displayed comparison.

In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives

$$\mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

The recursive Horner endpoint $\mathsf{eval}_{1_R,R}(x')$ is carried by induction on x' using the scalar zero, addition, and multiplication closure rows of the $\mathsf{Ring}^{\uparrow}$ certificate. Hence the multiplicative left-unit row retained in Definition 43.26 gives

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{eval}_{1_R,R}(x').$$

Classifier transitivity composes this unit comparison with the induction hypothesis, yielding

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

Source reflexivity gives $a \sim_R a$. Applying additive congruence from Definition 43.26 gives

$$a + 1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R a + \mathsf{fold}_{+,R}(x').$$

The cons Horner equation identifies the left endpoint with $\mathsf{eval}_{1_R,R}(\mathsf{cons}(a, x'))$, and the cons fold equation identifies the right endpoint with $\mathsf{fold}_{+,R}(\mathsf{cons}(a, x'))$. □

Theorem 50.123 (Polynomial evaluation at the multiplicative unit is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{1_R,R}(p) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Proof. By the normalized evaluation definition in Definition 50.133,

$$\mathsf{Eval}_{1_R,R}(p) = \mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)).$$

Apply Definition 50.148 to the finite coefficient spine $\mathsf{trim}_R(p)$. This gives

$$\mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Substituting the displayed definition of $\mathsf{Eval}_{1_R,R}(p)$ gives the claimed comparison. □

Lemma 50.124 (Padded coefficient extensionality for the polynomial classifier)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R from Definition 50.20 and Definition 50.22. If finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ satisfy $x[n] \sim_R y[n]$ for every $n : \mathsf{Nat}^{\uparrow}$, then*

$$\mathsf{PolySame}_R(x, y).$$

Proof. Put $x_0 := \mathsf{trim}_R(x)$ and $y_0 := \mathsf{trim}_R(y)$. The trailing-zero removal clause in Definition 50.22 gives decompositions $x = x_0{+}{+}t_x$ and $y = y_0{+}{+}t_y$, with $\mathsf{ZeroRem}_R(t_x)$ and $\mathsf{ZeroRem}_R(t_y)$ in the sense of Definition 50.28. Hence $x[n] \sim_R x_0[n]$ and $y[n] \sim_R y_0[n]$ for every n: inside the trimmed prefix the coefficient is unchanged, inside the removed tail the zero-remainder witness classifies the coefficient with 0_R, and after the whole spine both sides are the padding coefficient 0_R. Classifier symmetry and transitivity from cert_R give $x_0[n] \sim_R y_0[n]$ for every n.

The trimmed spines are normalized representatives. If one normalized representative had a last displayed coefficient beyond the length of the other, the padded coefficient comparison at that last index would classify that displayed coefficient with 0_R, contradicting the normalized no-trailing-zero condition in Definition 50.22. Thus x_0 and y_0 have the same finite length. Induction over this common length builds a structural classifier witness: at each head position the hypothesis $x_0[n] \sim_R y_0[n]$ supplies the scalar head comparison, and the induction hypothesis supplies the tail classifier. Therefore

$$\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0).$$

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30 gives the displayed polynomial classifier comparison. □

Lemma 50.125 (Raw Cauchy product coefficient formula)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(x, y)[n] \sim_R \mathsf{Cauchy}_R(x, y; n).$$

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The recursion defining radd_R gives the padded coefficient formula $\mathsf{radd}_R(u, v)[n] \sim_R u[n] + v[n]$ by induction on the two finite spines. The definitions of rscale_R and shift_R give

$$\mathsf{rscale}_R(a, u)[n] \sim_R a \cdot u[n], \qquad \mathsf{shift}_R(u)[0] \sim_R 0_R, \qquad \mathsf{shift}_R(u)[n+1] \sim_R u[n].$$

These comparisons use only zero self-classification, additive congruence, and the additive zero laws from Definition 43.26.

Induct on the first spine x. If $x = \mathsf{nil}$, the raw product is nil by Definition 50.139. Every summand in $\mathsf{Cauchy}_R(\mathsf{nil}, y; n)$ has the form $0_R \cdot y[j]$ and is classified with 0_R by the zero-multiplication row in Definition 43.26. The zero-summand fold theorem Definition 50.27 classifies the Cauchy fold with 0_R, which is the padded coefficient of nil.

For the cons step, write $x = \mathsf{cons}(a, x')$. At coefficient 0, the raw product recursion gives the sum of $a \cdot y[0]$ and the zero head of the shifted recursive product; the additive right-zero law classifies this with $a \cdot y[0]$, which is $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; 0)$. At coefficient $n + 1$, the same recursion and the auxiliary coefficient formulas give

$$\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)[n + 1] \sim_R a \cdot y[n + 1] + \mathsf{rmul}_R(x', y)[n].$$

The induction hypothesis and additive congruence replace the second summand by $\mathsf{Cauchy}_R(x', y; n)$. Splitting the finite index spine $i + j = n + 1$ into the head case $i = 0$ and the shifted cases $i = i' + 1$ identifies the resulting fold with $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; n + 1)$ by Definition 50.26. □

Lemma 50.126 (Raw Cauchy product coefficient associativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw Cauchy product data of Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(\mathsf{rmul}_R(x, y), z)[n] \sim_R \mathsf{rmul}_R(x, \mathsf{rmul}_R(y, z))[n].$$

Proof. Fix n. By Definition 50.159, the left coefficient is classified with $\mathsf{Cauchy}_R(\mathsf{rmul}_R(x, y), z; n)$. For each outer index $s+k = n$, apply Definition 50.159 to replace $\mathsf{rmul}_R(x, y)[s]$ by $\mathsf{Cauchy}_R(x, y; s)$. Multiplication congruence and Definition 50.26 lift these replacements through the outer finite fold. The right-multiplication half of Definition 49.95 then classifies $\mathsf{Cauchy}_R(x, y; s) \cdot z[k]$ with the inner fold over (i, j) satisfying $i+j = s$ of $(x[i] \cdot y[j]) \cdot z[k]$. The nested finite index spine $i+j = s$, $s+k = n$ is the same finite triple spine $i + j + k = n$ in lexicographic order after deleting the bookkeeping index s. The finite fold swap certificate of Definition 49.94, together with fold congruence, classifies the resulting nested fold with $\mathsf{Triple}_R(x, y, z; n)$. Thus the left raw product coefficient is classified with $\mathsf{Triple}_R(x, y, z; n)$.

The right coefficient is handled symmetrically. The coefficient formula first gives $\mathsf{Cauchy}_R(x, \mathsf{rmul}_R(y, z); n)$. Replacing each coefficient of $\mathsf{rmul}_R(y, z)$ by its Cauchy fold and applying the left-multiplication half of Definition 49.95 gives the finite fold over triples $i + j + k = n$ of $x[i] \cdot (y[j] \cdot z[k])$. Scalar multiplication associativity from Definition 43.26 classifies each displayed summand with $(x[i] \cdot y[j]) \cdot z[k]$. Fold congruence therefore classifies the right coefficient with the same $\mathsf{Triple}_R(x, y, z; n)$. Symmetry and transitivity of the scalar classifier compose the two comparisons. □

Theorem 50.127 (Polynomial multiplication associativity over commutative-ring scalars)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the canonical trim be trim_R from Definition 50.22, and let rmul_R and $\mathsf{PolyMul}_R$ be the raw and normalized products of Definition 50.139. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p, q), r), \mathsf{PolyMul}_R(p, \mathsf{PolyMul}_R(q, r))).$$

Proof. Put $p_0 := \mathsf{trim}_R(p)$, $q_0 := \mathsf{trim}_R(q)$, and $r_0 := \mathsf{trim}_R(r)$. Set $A := \mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p, q), r)$ and $B := \mathsf{PolyMul}_R(p, \mathsf{PolyMul}_R(q, r))$. By Definition 50.158, it is enough to prove $A[n] \sim_R B[n]$ for every $n : \mathsf{Nat}^{\uparrow}$.

Trim preserves padded coefficients by the trailing-zero decomposition in Definition 50.22: for every finite spine u and every n, $u[n] \sim_R \mathsf{trim}_R(u)[n]$. The coefficient formula Definition 50.159 is functorial in both inputs with respect to such pointwise padded-coefficient comparisons, since multiplication congruence compares each Cauchy summand and Definition 50.26 lifts the summand comparisons through the finite fold.

Unfolding Definition 50.139 and using trim idempotence from Definition 50.22, the left normalized product has the same padded coefficients as $\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)), r_0)$. By trim preservation and the functoriality just recorded, its nth coefficient is classified with $\mathsf{rmul}_R(\mathsf{rmul}_R(p_0, q_0), r_0)[n]$. Similarly, the right normalized product has the same padded coefficients as $\mathsf{rmul}_R(p_0, \mathsf{trim}_R(\mathsf{rmul}_R(q_0, r_0)))$, and its nth coefficient is classified with $\mathsf{rmul}_R(p_0, \mathsf{rmul}_R(q_0, r_0))[n]$.

Applying Definition 50.160 to p_0, q_0, r_0 gives

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p_0, q_0), r_0)[n] \sim_R \mathsf{rmul}_R(p_0, \mathsf{rmul}_R(q_0, r_0))[n].$$

Classifier symmetry and transitivity compose the two trim comparisons with this raw coefficient associativity comparison, yielding $A[n] \sim_R B[n]$ for the fixed n. Since n was arbitrary, Definition 50.158 gives $\mathsf{PolySame}_R(A, B)$, which is the displayed associativity statement. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul associative addFold classified`

Theorem 50.128 (Polynomial singleton evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient c and every carried scalar point α,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{cons}(c, \mathsf{nil})) \sim_R c.$$

Proof. Put $s = \mathsf{cons}(c, \mathsf{nil})$. First compute the unnormalized Horner value. The cons and nil recursion clauses in Definition 50.133 identify

$$\mathsf{eval}_{\alpha,R}(s) = c + \alpha \cdot 0_R.$$

The right zero-absorption half of Definition 43.60 gives

$$\alpha \cdot 0_R \sim_R 0_R.$$

Classifier reflexivity for c and addition congruence from Definition 43.26 give

$$c + \alpha \cdot 0_R \sim_R c + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $c + 0_R \sim_R c$, and transitivity yields

$$\mathsf{eval}_{\alpha,R}(s) \sim_R c.$$

By Definition 50.133, $\mathsf{Eval}_{\alpha,R}(s) = \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(s))$. The one-coefficient trimming boundary in Definition 50.166, read with Definition 50.22, has exactly the empty zero branch and the retained singleton branch. In the empty zero branch, $c \sim_R 0_R$ and $\mathsf{trim}_R(s) = \mathsf{nil}$; the nil Horner equation gives $\mathsf{Eval}_{\alpha,R}(s) = 0_R$, and classifier symmetry gives $0_R \sim_R c$. In the retained singleton branch, $\mathsf{trim}_R(s) = s$, so the raw computation above is the required normalized comparison. These are the two canonical trim branches for a singleton coefficient spine. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval singleton`

Theorem 50.129 (Raw polynomial multiplication left-distributes over raw addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and let radd_R, rscale_R, shift_R, and rmul_R be the raw operations of Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z)), \mathsf{radd}_R(\mathsf{rmul}_R(x, y), \mathsf{rmul}_R(x, z))).$$

Consequently the same classifier comparison holds when x, y, z are normalized polynomial representatives from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left distributes over raw add`

Proof. By Definition 50.158, it is enough to compare padded coefficients. First record the padded coefficient law for raw addition: for all finite coefficient spines u, v and every $m : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u, v)[m] \sim_R u[m] + v[m].$$

This is a direct induction over the two finite spines using the four recursion clauses in Definition 50.30. In the mixed cons-nil and nil-cons cases the missing coefficient is the padding endpoint 0_R; in the nil-nil tail the comparison is between 0_R and $0_R + 0_R$, handled by the additive zero laws and classifier symmetry from cert_R.

Fix $n : \mathsf{Nat}^{\uparrow}$. Let the finite split spine for $i + j = n$ carry three scalar spines

$$A_n(i, j) := x[i] \cdot \mathsf{radd}_R(y, z)[j], \qquad B_n(i, j) := x[i] \cdot y[j], \qquad C_n(i, j) := x[i] \cdot z[j].$$

For each split $i + j = n$, the raw-addition coefficient law gives

$$\mathsf{radd}_R(y, z)[j] \sim_R y[j] + z[j].$$

Multiplication congruence transports this under left multiplication by $x[i]$, and scalar left distributivity from Definition 43.26 gives

$$x[i] \cdot \mathsf{radd}_R(y, z)[j] \sim_R (x[i] \cdot y[j]) + (x[i] \cdot z[j]).$$

The matching split order assembles these scalar comparisons into a structural list-classifier witness from Definition 36.12 between A_n and the pointwise sum spine $B_n + C_n$. Applying Definition 50.26 to this witness gives

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n) \sim_R \mathsf{fold}_{+,R}(B_n + C_n).$$

The finite fold splitting lemma Definition 49.98 then classifies the right endpoint with the sum of the two separate Cauchy folds:

$$\mathsf{fold}_{+,R}(B_n + C_n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

Thus

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

The raw Cauchy coefficient formula Definition 50.159 gives

$$\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z))[n] \sim_R \mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n).$$

For the other endpoint, the raw-addition coefficient law gives

$$\mathsf{radd}_R(\mathsf{rmul}_R(x,y),\mathsf{rmul}_R(x,z))[n] \sim_R \mathsf{rmul}_R(x,y)[n] + \mathsf{rmul}_R(x,z)[n].$$

Applying Definition 50.159 to the two raw products and then additive congruence gives

$$\mathsf{rmul}_R(x,y)[n] + \mathsf{rmul}_R(x,z)[n] \sim_R \mathsf{Cauchy}_R(x,y;n) + \mathsf{Cauchy}_R(x,z;n).$$

Classifier symmetry and transitivity from cert_R compose the three displayed coefficient comparisons, yielding

$$\mathsf{rmul}_R(x,\mathsf{radd}_R(y,z))[n] \sim_R \mathsf{radd}_R(\mathsf{rmul}_R(x,y),\mathsf{rmul}_R(x,z))[n].$$

Since n was arbitrary, Definition 50.158 gives the displayed $\mathsf{PolySame}_R$ comparison. A normalized polynomial representative is a finite coefficient spine by Definition 50.3, so the final sentence is the same statement restricted to those representatives. □

Theorem 50.130 (Polynomial raw multiplication has singleton-one left identity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R,\sim_R)$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(1_R,\mathsf{nil}),p),p).$$

Proof. By Definition 50.158, it is enough to prove equality under the padded scalar classifier at every coefficient index. Unfolding the cons clause and the nil clause of raw multiplication in Definition 50.139 gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R,\mathsf{nil}),p) = \mathsf{radd}_R(\mathsf{rscale}_R(1_R,p),\mathsf{cons}(0_R,\mathsf{nil})).$$

Thus the product is the raw sum of the coefficientwise scalar multiple by 1_R and the one-place shift of the empty recursive product.

First, for every finite coefficient spine u and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u,\mathsf{cons}(0_R,\mathsf{nil}))[n] \sim_R u[n].$$

This is induction on u. If $u = \mathsf{nil}$, the only displayed head is $0_R + 0_R$, classified with 0_R by the additive zero rows in Definition 43.26; all later coefficients are padding coefficients 0_R. If $u = \mathsf{cons}(a,u')$, the head comparison is $(a + 0_R) \sim_R a$ by the additive right-zero row, while the tail comparison is the raw-addition right-nil induction that repeatedly classifies $b + 0_R$ with b and ends at the padding coefficient. These steps use additive congruence, zero self-classification, and classifier transitivity from cert_R.

Second, for every finite coefficient spine p and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{rscale}_R(1_R,p)[n] \sim_R p[n].$$

Induct on p. In the nil case both padded coefficients are 0_R. In the cons case, the exposed head is $1_R \cdot a$, classified with a by the multiplicative left-unit row retained in Definition 43.26, and the tail is supplied by the induction hypothesis. After the displayed spine ends, both sides are again the padding coefficient 0_R.

Apply the first coefficient comparison to $u = \mathsf{rscale}_R(1_R, p)$ and compose it with the second comparison. For every $n : \mathsf{Nat}^{\uparrow}$ this gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p)[n] \sim_R p[n].$$

The padded coefficient extensionality lemma Definition 50.158 converts these pointwise scalar comparisons into the displayed $\mathsf{PolySame}_R$ statement. □

Theorem 50.131 (Raw polynomial multiplication associativity up to add-trim)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, the canonical trim be trim_R from Definition 50.22, and let $\mathsf{PolySame}_R$ and rmul_R be the add-trim classifier and raw Cauchy product of Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R\big(\mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r), \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r))\big).$$

Consequently the same comparison holds when p, q, r are normalized polynomial representatives from Definition 50.3.

Proof. Set

$$A := \mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r), \qquad B := \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r)).$$

By Definition 50.158, it is enough to prove $A[n] \sim_R B[n]$ for every $n : \mathsf{Nat}^{\uparrow}$.

Fix $n : \mathsf{Nat}^{\uparrow}$. The raw Cauchy coefficient associativity lemma Definition 50.160, applied to the finite coefficient spines p, q, r, gives exactly

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r)[n] \sim_R \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r))[n].$$

This is $A[n] \sim_R B[n]$. Since n was arbitrary, Definition 50.158 converts the pointwise padded coefficient comparisons into

$$\mathsf{PolySame}_R(A, B).$$

Unfolding the definitions of A and B gives the displayed add-trim classifier comparison. A normalized polynomial representative from Definition 50.3 is a finite coefficient spine, so the final sentence is the same statement restricted to those carried representatives. □

Proposition 50.132 (Literal polynomial add-trim equality fails at cancelling singletons)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be as in Definition 50.30. Suppose $\neg(a \sim_R 0_R)$, $\neg(b \sim_R 0_R)$, and $(a + b) \sim_R 0_R$. Write $[c] := \mathsf{cons}(c, \mathsf{nil})$, and set $p := [a]$ and $q := [b]$. Then literal finite-spine equality does not commute with raw addition and trimming:*

$$\begin{aligned} \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &= \mathsf{nil}, \\ \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) &= [a + b], \\ \mathsf{trim}_R(\mathsf{radd}_R(p, q)) &\neq \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)). \end{aligned}$$

Nevertheless the two displayed outputs are polynomial-classified the same:

$$\mathsf{PolySame}_R\big(\mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q))\big).$$

For the canonical integer coefficient ring from chapter 31 and chapter 44, taking $a = 1$ and $b = -1$ gives $p = [1]$, $q = [-1]$, $\mathsf{trim}_{\mathbb{Z}}(\mathsf{radd}_{\mathbb{Z}}(p, q)) = \mathsf{nil}$, and $\mathsf{radd}_{\mathbb{Z}}(\mathsf{trim}_{\mathbb{Z}}(p), \mathsf{trim}_{\mathbb{Z}}(q)) = [0]$.

Proof. The nonzero hypotheses make p and q normalized singleton representatives by Definition 50.166; the canonical trim fixes normalized representatives, so $\mathsf{trim}_R(p) = p$ and $\mathsf{trim}_R(q) = q$. By the cons-cons and nil-nil clauses of radd_R in Definition 50.30, $\mathsf{radd}_R(\mathsf{trim}_R(p), \mathsf{trim}_R(q)) = \mathsf{radd}_R([a], [b]) = [a + b]$. The equality $\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}$ is the singleton cancellation conclusion of Definition 50.167. The displayed inequality follows because the left side is nil while the right side is a cons spine. Constructor no-confusion for the finite-spine carrier of Definition 36.1 is the mixed nil/cons separation isolated in Definition 36.13 and Definition 36.14. For classifier comparison, unfold $\mathsf{PolySame}_R$. The first argument trims again to nil. The second argument is $[a+b]$, and the hypothesis $(a+b) \sim_R 0_R$ sends its singleton coefficient to the trailing-zero branch, so its trim is also nil. The nil-nil clause of $\mathsf{ListClassifierSpec}$ gives the displayed $\mathsf{PolySame}_R$ witness. The integer specialization uses the standard integer facts $1 \neq 0$, $-1 \neq 0$, and $1 + (-1) = 0$; Definition 44.8 keeps the ring operations and classifier when the commutative-ring source is read through $\mathsf{Ring}^{\uparrow}$. □

Definition 50.133 (Polynomial Horner evaluation)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a carried scalar point α, define the Horner evaluation on finite coefficient spines from Definition 36.1 by recursion*

$$\mathsf{eval}_{\alpha,R}(\mathsf{nil}) := 0_R.$$

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x)) := a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Define the normalized polynomial evaluation by

$$\mathsf{Eval}_{\alpha,R}(x) := \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(x)).$$

Define the normalized polynomial sum by

$$\mathsf{PolyAdd}_R(x, y) := \mathsf{trim}_R(\mathsf{radd}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

On a normalized representative from Definition 50.3, $\mathsf{Eval}_{\alpha,R}$ is the Horner evaluation itself, by trim idempotence from Definition 50.22.

Lemma 50.134 (Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{\alpha,R}(x{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Consequently, for every finite coefficient spine x,

$$\mathsf{Eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(x).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialZeroRemainder addFold empty`

Proof. First prove the endpoint claim $\mathsf{eval}_{\alpha,R}(t) \sim_R 0_R$ for every classified zero tail t. Induct on the displayed $\mathsf{ZeroRem}_R(t)$ witness. The nil clause is the defining equation $\mathsf{eval}_{\alpha,R}(\mathsf{nil}) = 0_R$ together with zero self-classification from Definition 43.26. In the cons clause, write $t = \mathsf{cons}(a, t')$. The constructor gives $a \sim_R 0_R$ and the induction hypothesis gives $\mathsf{eval}_{\alpha,R}(t') \sim_R 0_R$. Multiplication congruence and the zero-multiplication row in Definition 43.26 classify $\alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with 0_R.

Addition congruence then classifies $a + \alpha \cdot \mathsf{eval}_{\alpha,R}(t')$ with $0_R + 0_R$, and the additive left-zero law reduces the latter endpoint to 0_R.

For the displayed tail-invariance statement, induct on x. The nil case is the endpoint claim just proved. In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives $\mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R \mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports this comparison through left multiplication by α, and addition congruence with source reflexivity on a gives

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x'{+}{+}t) \sim_R a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x').$$

The two Horner recursion equations identify these endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'{+}{+}t))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

For the consequence, the trailing-zero removal clause in Definition 50.22 presents each raw spine x as $\mathsf{trim}_R(x){+}{+}t$ with $\mathsf{ZeroRem}_R(t)$. Applying the first part gives $\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{Eval}_{\alpha,R}(x)$, and symmetry of the scalar classifier reverses the displayed comparison. □

Lemma 50.135 (Horner evaluation respects raw polynomial addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonAddFold append hsame`

Proof. Perform induction on the two finite spines, following the four recursion clauses for radd_R in Definition 50.30.

The nil-nil case compares 0_R with $0_R + 0_R$, so it is exactly the additive left-zero law, read with symmetry when needed.

In the cons-nil case, write $x = \mathsf{cons}(a, x')$. The raw-addition recursion exposes the head $a + 0_R$ and the tail $\mathsf{radd}_R(x', \mathsf{nil})$. The induction hypothesis classifies the tail evaluation with $\mathsf{eval}_{\alpha,R}(x') + 0_R$, and Definition 43.31 reduces that endpoint to $\mathsf{eval}_{\alpha,R}(x')$. Multiplication congruence transports the tail comparison through left multiplication by α. Additive congruence and the additive right-zero law reduce the exposed head $a + 0_R$ to a, and the target endpoint $(\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))) + 0_R$ to $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$.

The nil-cons case is symmetric. The raw head is $0_R + b$, which reduces to b by the additive left-zero law, and the outer target $0_R + \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ reduces by the same law. The tail comparison is transported through multiplication by α exactly as in the cons-nil case.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(x', y')) \sim_R \mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y').$$

Multiplication congruence transports this through left multiplication by α, and left distributivity from Definition 43.26 classifies

$$\alpha \cdot (\mathsf{eval}_{\alpha,R}(x') + \mathsf{eval}_{\alpha,R}(y'))$$

with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence therefore transports the Horner endpoint for $\mathsf{radd}_R(x, y)$ to

$$(a + b) + (\alpha \cdot \mathsf{eval}_{\alpha,R}(x') + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Apply Definition 42.13 to the additive abelian-group operation, with the four additive blocks a, b, $\alpha \cdot \mathsf{eval}_{\alpha,R}(x')$, and $\alpha \cdot \mathsf{eval}_{\alpha,R}(y')$. This classifies the last endpoint with

$$(a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x')) + (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')),$$

which is exactly $\mathsf{eval}_{\alpha,R}(x) + \mathsf{eval}_{\alpha,R}(y)$ by the Horner recursion. □

Theorem 50.136 (Polynomial evaluation is additive)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) + \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.133,

$$\mathsf{PolyAdd}_R(p, q) = \mathsf{trim}_R(\mathsf{radd}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies $\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q))$ with $\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0))$. Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)).$$

By Definition 50.135,

$$\mathsf{eval}_{\alpha,R}(\mathsf{radd}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) + \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$, and $\mathsf{Eval}_{\alpha,R}$ on each displayed representative is the unnormalized Horner evaluation. Substituting these equalities gives the carrier formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawAdd classified`

Lemma 50.137 (Horner evaluation respects structural classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$, if*

$$\mathsf{ListClassifierSpec}(\sim_R, x, y)$$

holds, then

$$\mathsf{eval}_{\alpha,R}(x) \sim_R \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the displayed structural classifier witness from Definition 36.12. By Definition 36.13, the only constructor cases are nil-nil and cons-cons; mixed nil-cons shapes have no classifier witness.

In the nil-nil case, the two Horner equations in Definition 50.133 identify both endpoints with 0_R. The zero self-classification field retained in Definition 43.26 gives $0_R \sim_R 0_R$.

In the cons-cons case, write $x = \mathsf{cons}(a, x')$ and $y = \mathsf{cons}(b, y')$. The classifier constructor supplies $a \sim_R b$ and a tail witness $\mathsf{ListClassifierSpec}(\sim_R, x', y')$. The induction hypothesis gives

$$\mathsf{eval}_{\alpha,R}(x') \sim_R \mathsf{eval}_{\alpha,R}(y').$$

The carried point α is self-classified by the scalar certificate. Multiplication congruence from Definition 43.26 transports the tail comparison through left multiplication by α, giving

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

Addition congruence combines this comparison with the head comparison $a \sim_R b$:

$$a + \alpha \cdot \mathsf{eval}_{\alpha,R}(x') \sim_R b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y').$$

The cons recursion equation in Definition 50.133 identifies these two endpoints with $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x'))$ and $\mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$, respectively. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval list classifier classified`

Theorem 50.138 (Polynomial evaluation respects the polynomial classifier)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(p, q) \Longrightarrow \mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(q).$$

Proof. Unfold $\mathsf{PolySame}_R$ using Definition 50.30. A classifier witness for p and q is exactly

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(p), \mathsf{trim}_R(q)).$$

Apply Definition 50.137 to the two trimmed spines. This gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(q)).$$

The normalized evaluation definition in Definition 50.133 reads the left endpoint as $\mathsf{Eval}_{\alpha,R}(p)$ and the right endpoint as $\mathsf{Eval}_{\alpha,R}(q)$, giving the displayed implication. □

Definition 50.139 (Polynomial raw Cauchy product)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and raw addition radd_R from Definition 50.30. For a scalar coefficient a and a finite coefficient spine $y : \mathsf{ListCarrier}(R)$, define scalar multiplication of coefficient spines by*

$$\mathsf{rscale}_R(a, \mathsf{nil}) := \mathsf{nil}, \qquad \mathsf{rscale}_R(a, \mathsf{cons}(b, y)) := \mathsf{cons}(a \cdot b, \mathsf{rscale}_R(a, y)).$$

Define the one-place coefficient shift by

$$\mathsf{shift}_R(y) := \mathsf{cons}(0_R, y).$$

The raw polynomial product is the finite recursion

$$\mathsf{rmul}_R(\mathsf{nil}, y) := \mathsf{nil},$$

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), y) := \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(x, y))).$$

The normalized polynomial product is

$$\mathsf{PolyMul}_R(x, y) := \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(x), \mathsf{trim}_R(y))).$$

This is the Cauchy product in low-coefficient-first Horner form: the cons clause expresses $(a + Xx)y = ay + X(xy)$.

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul`

Lemma 50.140 (Horner evaluation of scalar-multiplied coefficient spines)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every carried scalar coefficient a and every finite coefficient spine $y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) \sim_R a \cdot \mathsf{eval}_{\alpha,R}(y).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawScale classified`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval raw scale empty classified`

Proof. Induct on the finite spine y from Definition 36.1. In the nil case, both the scalar-multiple spine and its Horner value are 0_R. The target endpoint is $a \cdot 0_R$, and the zero-multiplication row in Definition 43.26, with symmetry of $\sim_R$, gives the displayed comparison.

In the cons case, write $y = \mathsf{cons}(b, y')$. The recursion in Definition 50.139 and the Horner recursion in Definition 50.133 identify the left endpoint with

$$a \cdot b + \alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y')).$$

The induction hypothesis and multiplication congruence classify this endpoint with

$$a \cdot b + \alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Multiplicative associativity, multiplicative commutativity from Definition 44.33, and multiplication congruence classify

$$\alpha \cdot (a \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

Addition congruence transports this comparison under the common summand $a \cdot b$. The left distributive row in Definition 43.26, used in the reverse direction, then identifies

$$a \cdot b + a \cdot (\alpha \cdot \mathsf{eval}_{\alpha,R}(y'))$$

with

$$a \cdot (b + \alpha \cdot \mathsf{eval}_{\alpha,R}(y')).$$

The final endpoint is $a \cdot \mathsf{eval}_{\alpha,R}(\mathsf{cons}(b, y'))$ by the Horner recursion, and transitivity composes the comparisons. □

Lemma 50.141 (Horner evaluation of coefficient shift)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(x)) \sim_R \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

Proof. By Definition 50.139, $\mathsf{shift}_R(x) = \mathsf{cons}(0_R, x)$. The Horner recursion from Definition 50.133 makes its evaluation

$$0_R + \alpha \cdot \mathsf{eval}_{\alpha,R}(x).$$

The additive left-zero row in Definition 43.26 classifies this endpoint with $\alpha \cdot \mathsf{eval}_{\alpha,R}(x)$. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval coefficient shift`

Lemma 50.142 (Horner evaluation respects raw polynomial multiplication)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x, y)) \sim_R \mathsf{eval}_{\alpha,R}(x) \cdot \mathsf{eval}_{\alpha,R}(y).$$

Proof. Induct on the first spine x. In the nil case, $\mathsf{rmul}_R(\mathsf{nil}, y) = \mathsf{nil}$, so the left endpoint evaluates to 0_R. The target endpoint is $0_R \cdot \mathsf{eval}_{\alpha,R}(y)$, and the zero-multiplication row in Definition 43.26, with symmetry, gives the comparison.

For the cons case, write $x = \mathsf{cons}(a, x')$, put $u = \mathsf{eval}_{\alpha,R}(x')$, and put $v = \mathsf{eval}_{\alpha,R}(y)$. The raw product recursion in Definition 50.139 and the raw additivity theorem Definition 50.135 give

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rscale}_R(a, y)) + \mathsf{eval}_{\alpha,R}(\mathsf{shift}_R(\mathsf{rmul}_R(x', y))).$$

By Definition 50.140, the first summand is classified with $a \cdot v$. By Definition 50.141, the second summand is classified with

$$\alpha \cdot \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(x', y)).$$

The induction hypothesis and multiplication congruence classify this with $\alpha \cdot (u \cdot v)$. Multiplicative associativity classifies $\alpha \cdot (u \cdot v)$ with $(\alpha \cdot u) \cdot v$. Addition congruence therefore gives

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)) \sim_R a \cdot v + (\alpha \cdot u) \cdot v.$$

The right distributive row retained by Definition 44.8, equivalently derived from Definition 44.34, classifies

$$(a + \alpha \cdot u) \cdot v$$

with

$$a \cdot v + (\alpha \cdot u) \cdot v.$$

Using symmetry of this distributive comparison and the Horner recursion

$$\mathsf{eval}_{\alpha,R}(\mathsf{cons}(a, x')) = a + \alpha \cdot u,$$

transitivity gives the required cons-case endpoint. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval rawMul classified`

Theorem 50.143 (Polynomial evaluation is multiplicative)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

If p and q are normalized representatives in the polynomial carrier, the same assertion is the carrier formula

$$\mathsf{eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{eval}_{\alpha,R}(p) \cdot \mathsf{eval}_{\alpha,R}(q).$$

Proof. Put $p_0 = \mathsf{trim}_R(p)$ and $q_0 = \mathsf{trim}_R(q)$. By Definition 50.139,

$$\mathsf{PolyMul}_R(p, q) = \mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)).$$

Trim idempotence from Definition 50.22 identifies the normalized evaluation of this endpoint with

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

Applying Definition 50.134 gives

$$\mathsf{Eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)).$$

By Definition 50.142,

$$\mathsf{eval}_{\alpha,R}(\mathsf{rmul}_R(p_0, q_0)) \sim_R \mathsf{eval}_{\alpha,R}(p_0) \cdot \mathsf{eval}_{\alpha,R}(q_0).$$

The right endpoint is $\mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q)$ by the definition of p_0, q_0, and normalized evaluation in Definition 50.133. Classifier transitivity composes the displayed comparisons.

If p and q are already normalized polynomial-carrier representatives, trim idempotence gives $p_0 = p$ and $q_0 = q$. The product endpoint is normalized by its defining trim, so normalized evaluation on $\mathsf{PolyMul}_R(p, q)$ is the unnormalized Horner evaluation of that product. Substituting these equalities gives the carrier formula. □

Theorem 50.144 (Polynomial evaluation is a semiring morphism through classification)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Fix a carried scalar point α. For finite coefficient spines $p, q, p', q' : \mathsf{ListCarrier}(R)$, if $\mathsf{PolySame}_R(p, p')$ and $\mathsf{PolySame}_R(q, q')$, then*

$$\mathsf{Eval}_{\alpha,R}(p) \sim_R \mathsf{Eval}_{\alpha,R}(p'),$$

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyAdd}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) + \mathsf{Eval}_{\alpha,R}(q),$$

and

$$\mathsf{Eval}_{\alpha,R}(\mathsf{PolyMul}_R(p, q)) \sim_R \mathsf{Eval}_{\alpha,R}(p) \cdot \mathsf{Eval}_{\alpha,R}(q).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval semiring morphism`

Proof. The first clause is exactly Definition 50.138 applied to the classifier witness between p and p'.

The second and third clauses are the normalized-operation laws from Definition 50.136 and Definition 50.143. They apply to the original representatives p and q, so the operation endpoints are already the canonical normalized polynomial operations.

When a later scalar comparison is written with p' or q' instead, apply Definition 50.138 to the appropriate classifier witness and compose by transitivity of $\sim_R$. Thus the descended endpoints may be substituted in the additive and multiplicative scalar comparisons without reassembling the three facts. □

Lemma 50.145 (Zero-point Horner cons evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. The cons recursion in Definition 50.133 identifies the left endpoint with

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t).$$

The zero-absorption theorem Definition 43.60, applied to $\mathsf{eval}_{0_R,R}(t)$, gives

$$0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R 0_R.$$

Classifier reflexivity supplies $a \sim_R a$, and addition congruence from Definition 43.26 gives

$$a + 0_R \cdot \mathsf{eval}_{0_R,R}(t) \sim_R a + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $a+0_R \sim_R a$. Classifier transitivity composes the two comparisons. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero cons constant`

Lemma 50.146 (Zero-point Horner evaluation ignores classified zero tails)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. For finite coefficient spines $x, t : \mathsf{ListCarrier}(R)$, if $\mathsf{ZeroRem}_R(t)$ holds in the sense of Definition 50.28, then*

$$\mathsf{eval}_{0_R,R}(x{+}{+}t) \sim_R \mathsf{eval}_{0_R,R}(x).$$

Proof. First record the endpoint claim that every classified zero remainder evaluates to zero at 0_R. If the zero remainder is nil, the nil recursion in Definition 50.133 identifies its evaluation with 0_R, and zero self-classification from Definition 43.26 gives the claim. If the zero remainder is $\mathsf{cons}(c, t')$, the cons clause of Definition 50.28 supplies $c \sim_R 0_R$. By Definition 50.145,

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R c,$$

and transitivity gives $\mathsf{eval}_{0_R,R}(\mathsf{cons}(c, t')) \sim_R 0_R$.

Now split on the first constructor of x. If $x = \mathsf{nil}$, then $x{+}{+}t = t$, so the endpoint claim gives the displayed comparison. If $x = \mathsf{cons}(a, x')$, then $x{+}{+}t = \mathsf{cons}(a, x'{+}{+}t)$. Applying Definition 50.145 twice gives

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x'{+}{+}t)) \sim_R a$$

and

$$\mathsf{eval}_{0_R,R}(\mathsf{cons}(a, x')) \sim_R a.$$

Symmetry of the second comparison and transitivity give the required comparison for the cons case. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero point zero tail invariance`

Theorem 50.147 (Polynomial evaluation at zero returns the constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient a and every finite coefficient spine $t : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{0_R,R}(\mathsf{cons}(a, t)) \sim_R a.$$

Proof. Set $x = \mathsf{cons}(a, t)$. The trailing-zero removal clause in Definition 50.22 gives a finite zero remainder z such that

$$x = \mathsf{trim}_R(x){+}{+}z, \qquad \mathsf{ZeroRem}_R(z).$$

Applying Definition 50.146 to $\mathsf{trim}_R(x)$ and z yields

$$\mathsf{eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(\mathsf{trim}_R(x)).$$

By the normalized evaluation definition in Definition 50.133, the right endpoint is $\mathsf{Eval}_{0_R,R}(x)$. Symmetry gives

$$\mathsf{Eval}_{0_R,R}(x) \sim_R \mathsf{eval}_{0_R,R}(x).$$

The raw cons comparison Definition 50.145 gives

$$\mathsf{eval}_{0_R,R}(x) \sim_R a.$$

Classifier transitivity composes these two comparisons, and substituting $x = \mathsf{cons}(a, t)$ gives the displayed formula. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval zero constant term`

Lemma 50.148 (Unit-point Horner evaluation is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26, satisfying*

$$\mathsf{fold}_{+,R}(\mathsf{nil}) = 0_R, \qquad \mathsf{fold}_{+,R}(\mathsf{cons}(a, x')) = a + \mathsf{fold}_{+,R}(x').$$

For every finite coefficient spine $x : \mathsf{ListCarrier}(R)$,

$$\mathsf{eval}_{1_R,R}(x) \sim_R \mathsf{fold}_{+,R}(x).$$

Proof. Induct on the finite spine x from Definition 36.1.

In the nil case, the nil Horner equation in Definition 50.133 and the nil fold equation identify both endpoints with 0_R. The zero self-classification row of Definition 43.26 gives the displayed comparison.

In the cons case, write $x = \mathsf{cons}(a, x')$. The induction hypothesis gives

$$\mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

The recursive Horner endpoint $\mathsf{eval}_{1_R,R}(x')$ is carried by induction on x' using the scalar zero, addition, and multiplication closure rows of the $\mathsf{Ring}^{\uparrow}$ certificate. Hence the multiplicative left-unit row retained in Definition 43.26 gives

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{eval}_{1_R,R}(x').$$

Classifier transitivity composes this unit comparison with the induction hypothesis, yielding

$$1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R \mathsf{fold}_{+,R}(x').$$

Source reflexivity gives $a \sim_R a$. Applying additive congruence from Definition 43.26 gives

$$a + 1_R \cdot \mathsf{eval}_{1_R,R}(x') \sim_R a + \mathsf{fold}_{+,R}(x').$$

The cons Horner equation identifies the left endpoint with $\mathsf{eval}_{1_R,R}(\mathsf{cons}(a, x'))$, and the cons fold equation identifies the right endpoint with $\mathsf{fold}_{+,R}(\mathsf{cons}(a, x'))$. □

Theorem 50.149 (Polynomial evaluation at the multiplicative unit is coefficient fold)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. Let $\mathsf{fold}_{+,R}$ be the finite recursive coefficient fold from Definition 50.26. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{Eval}_{1_R,R}(p) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Proof. By the normalized evaluation definition in Definition 50.133,

$$\mathsf{Eval}_{1_R,R}(p) = \mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)).$$

Apply Definition 50.148 to the finite coefficient spine $\mathsf{trim}_R(p)$. This gives

$$\mathsf{eval}_{1_R,R}(\mathsf{trim}_R(p)) \sim_R \mathsf{fold}_{+,R}(\mathsf{trim}_R(p)).$$

Substituting the displayed definition of $\mathsf{Eval}_{1_R,R}(p)$ gives the claimed comparison. □

Lemma 50.150 (Polynomial raw addition structural swap)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$. Assume scalar additive commutativity:*

$$a + b \sim_R b + a$$

for all scalar coefficients a, b. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Induct simultaneously on the two finite coefficient spines, following the four recursion clauses of radd_R in Definition 50.30. The target classifier is the finite-spine structural classifier of Definition 36.12, with matching constructors isolated by Definition 36.13.

In the nil-nil case, both raw additions are nil, so the nil clause of $\mathsf{ListClassifierSpec}$ gives the claim.

In the cons-nil case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{nil}$. The recursion gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a + 0_R, \mathsf{radd}_R(p', \mathsf{nil}))$$

and

$$\mathsf{radd}_R(q, p) = \mathsf{cons}(0_R + a, \mathsf{radd}_R(\mathsf{nil}, p')).$$

Scalar additive commutativity gives $a + 0_R \sim_R 0_R + a$. The induction hypothesis applied to p' and nil gives the tail classifier

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', \mathsf{nil}), \mathsf{radd}_R(\mathsf{nil}, p')).$$

The cons clause of $\mathsf{ListClassifierSpec}$ combines the head comparison and this tail comparison.

In the nil-cons case, write $p = \mathsf{nil}$ and $q = \mathsf{cons}(b, q')$. The recursion gives heads $0_R + b$ and $b + 0_R$. Scalar additive commutativity gives $0_R + b \sim_R b + 0_R$, and the induction hypothesis at nil and q' supplies the tail classifier. The cons clause rebuilds the displayed list-classifier witness.

In the cons-cons case, write $p = \mathsf{cons}(a, p')$ and $q = \mathsf{cons}(b, q')$. The recursion exposes heads $a + b$ and $b + a$, related by scalar additive commutativity. The induction hypothesis supplies

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p', q'), \mathsf{radd}_R(q', p')).$$

The cons clause combines the head and tail witnesses, giving the structural classifier for the two raw sums. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd structural swap`
Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd commutative classified`

Theorem 50.151 (Polynomial raw addition commutativity from scalar additive commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Use the additive abelian-group and classifier fields retained in Definition 43.26: zero self-classification, classifier transitivity, additive congruence, additive zero laws, and scalar additive commutativity. For a commutative-ring scalar source, Definition 44.8 keeps these data unchanged. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

Proof. Apply Definition 50.150 to p and q using the scalar additive commutativity row from Definition 43.26. This gives

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{radd}_R(p, q), \mathsf{radd}_R(q, p)).$$

The nil finite spine is a zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 with common classified spines $\mathsf{radd}_R(p, q)$ and $\mathsf{radd}_R(q, p)$, and with both zero remainders equal to nil. Finite-spine append by nil leaves each spine unchanged, so the lemma yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{radd}_R(p, q)), \mathsf{trim}_R(\mathsf{radd}_R(q, p))).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, this is exactly the displayed polynomial raw classifier comparison. □

Lemma 50.152 (Raw polynomial addition is structurally associative)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive associativity, additive zero laws, classifier symmetry, classifier transitivity, and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{ListClassifierSpec}\big(\sim_R, \mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))\big).$$

Proof. Induct on the finite triple spine (p, q, r), using the maximum of the three finite lengths as the decreasing measure. The nil-nil-nil case unfolds both raw additions by Definition 50.30 to nil, and the nil clause of Definition 36.12 gives the classifier witness.

In every remaining constructor case, at least one of the three spines is a cons spine. Unfold the corresponding clauses of radd_R from Definition 50.30. Both associated raw sums expose a cons head, and their tails are exactly the two associated raw sums for the tail triple, with nil used as the tail of any absent spine. The induction hypothesis supplies the recursive tail classifier.

It remains to classify the exposed scalar heads. If all three heads are present, say a, b, c, the comparison is

$$(a + b) + c \sim_R a + (b + c),$$

which is scalar additive associativity. If exactly two heads are present, the missing head is the padding coefficient 0_R, so the three possible comparisons

$$(a + b) + 0_R \sim_R a + (b + 0_R),$$

$$(a + 0_R) + c \sim_R a + (0_R + c),$$

$$(0_R + b) + c \sim_R 0_R + (b + c)$$

are the same associativity law with one displayed argument equal to 0_R. If only the left input has a head a, the comparison is

$$(a + 0_R) + 0_R \sim_R a + 0_R,$$

which is the additive right-zero law applied to $a + 0_R$. If only the middle input has a head b, the comparison is

$$(0_R + b) + 0_R \sim_R 0_R + (b + 0_R),$$

again additive associativity. If only the right input has a head c, the comparison is

$$0_R + c \sim_R 0_R + (0_R + c),$$

obtained by reversing the additive left-zero law at the endpoint $0_R + c$ using classifier symmetry from cert_R.

In each constructor case, the scalar head comparison and the tail classifier rebuild the cons clause of $\mathsf{ListClassifierSpec}$. These cases exhaust the finite-spine constructors by Definition 36.13. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd associative classified`

Theorem 50.153 (Polynomial raw addition associativity)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, canonical trim trim_R, and scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$. Assume the additive fields retained in Definition 43.26: additive abelian-group laws and addition congruence. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \mathsf{radd}_R(p, \mathsf{radd}_R(q, r))).$$

Proof. Set

$$u := \mathsf{radd}_R(\mathsf{radd}_R(p, q), r), \qquad \omega := \mathsf{radd}_R(p, \mathsf{radd}_R(q, r)).$$

By Definition 50.152,

$$\mathsf{ListClassifierSpec}(\sim_R, \nu, \omega)$$

holds. The nil spine is a classified zero remainder by the nil clause of Definition 50.28. Apply Definition 50.29 to the common classified spines ν and ω, with nil zero remainders on both sides. The append-nil identities for finite spines give

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\nu), \mathsf{trim}_R(\omega)).$$

Unfolding $\mathsf{PolySame}_R$ from Definition 50.30 turns this normalized structural classifier into the displayed polynomial classifier comparison. □

Theorem 50.154 (Polynomial raw multiplication commutativity from scalar commutativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p, q), \mathsf{rmul}_R(q, p)).$$

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul commutative classified`

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The retained rows used below are the $\mathsf{NameCert}$ reflexivity, symmetry, and transitivity fields, the additive abelian-group laws and addition congruence from Definition 43.26, multiplication congruence, and the multiplicative commutativity row from Definition 44.33.

First record the finite-spine consequences of the existing raw-addition package. The relation $\mathsf{PolySame}_R$ is reflexive, symmetric, and transitive after unfolding Definition 50.30; this is exactly structural list-classifier reflexivity, Definition 36.47, and Definition 36.15 applied to the two trimmed representatives.

Raw addition is compatible with $\mathsf{PolySame}_R$ in both inputs. Indeed, Definition 50.32 reduces both raw sums to sums of their trimmed inputs. The two displayed trimmed-input classifiers give a structural classifier between the two trimmed raw sums by induction on the two list-classifier witnesses in Definition 36.12; the exposed heads are handled by addition congruence, with $0_R \sim_R 0_R$ in the padding branches. Transitivity and symmetry of $\mathsf{PolySame}_R$ compose these three comparisons.

The shift operation preserves $\mathsf{PolySame}_R$. If $\mathsf{PolySame}_R(x, y)$ holds, write the trim decompositions supplied by Definition 50.22 as $x = x_0{+}{+}z$ and $y = y_0{+}{+}w$, with $\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0)$ and zero remainders z, w. Then

$$\mathsf{shift}_R(x) = (\mathsf{cons}(0_R, x_0)){+}{+}z, \qquad \mathsf{shift}_R(y) = (\mathsf{cons}(0_R, y_0)){+}{+}w.$$

The cons classifier built from $0_R \sim_R 0_R$ and the classifier on x_0, y_0, followed by Definition 50.29, gives $\mathsf{PolySame}_R(\mathsf{shift}_R(x), \mathsf{shift}_R(y))$. The same argument, with 0_R replaced by classified heads $h \sim_R h'$, shows that $h \sim_R h'$ and $\mathsf{PolySame}_R(t, t')$ imply

$$\mathsf{PolySame}_R(\mathsf{cons}(h, t), \mathsf{cons}(h', t')).$$

Zero remainders are closed under raw addition: induct on the two zero-remainder witnesses from Definition 50.28. In the cons-cons branch, addition congruence sends $a + b$ to $0_R + 0_R$, and the additive left-zero law reduces that endpoint to 0_R; the cons-nil and nil-cons branches use the additive right-zero and left-zero laws. Hence $\mathsf{rmul}_R(s, \mathsf{nil})$ is a zero remainder for every finite spine s, by induction on s using Definition 50.139: the cons step combines $\mathsf{rscale}_R(a, \mathsf{nil}) = \mathsf{nil}$, the zero-remainder closure of shift_R, and raw-addition closure. Applying Definition 50.29 with common spine nil gives

$$\mathsf{PolySame}_R(\mathsf{nil}, \mathsf{rmul}_R(s, \mathsf{nil})).$$

The key auxiliary recursion is right-cons expansion: for every scalar coefficient a and finite spines x, q,

$$\mathsf{PolySame}_R\big(\mathsf{rmul}_R(q, \mathsf{cons}(a, x)), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))\big).$$

Induct on q. If $q = \mathsf{nil}$, the left side is nil and the right side is a raw sum of zero-remainder spines, so the zero-remainder comparison above applies. For the cons step write $q = \mathsf{cons}(b, y)$. Unfolding Definition 50.139 and the cons-cons branch of Definition 50.30, the two heads are $b \cdot a + 0_R$ and $a \cdot b + 0_R$; multiplicative commutativity and addition congruence classify them. The two tails are

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{rmul}_R(y, \mathsf{cons}(a, x)))$$

and

$$\mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

The induction hypothesis and raw-addition compatibility compare the first tail with

$$\mathsf{radd}_R(\mathsf{rscale}_R(b, x), \mathsf{radd}_R(\mathsf{rscale}_R(a, y), \mathsf{shift}_R(\mathsf{rmul}_R(y, x)))).$$

Raw-addition associativity from Definition 50.153 and raw-addition commutativity from Definition 50.151, transported through raw-addition compatibility, move the first two summands past each other and give the second tail. The cons preservation fact then proves the cons step.

Now induct on p. If $p = \mathsf{nil}$, then $\mathsf{rmul}_R(\mathsf{nil}, q) = \mathsf{nil}$ by Definition 50.139, while $\mathsf{rmul}_R(q, \mathsf{nil})$ is a zero remainder by the preceding paragraph; hence the displayed $\mathsf{PolySame}_R$ holds.

For the cons step write $p = \mathsf{cons}(a, x)$ and assume

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, q), \mathsf{rmul}_R(q, x)).$$

Unfolding the left raw product gives

$$\mathsf{rmul}_R(\mathsf{cons}(a, x), q) = \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(x, q))).$$

Shift preservation and raw-addition compatibility transport the induction hypothesis to

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(a, x), q), \mathsf{radd}_R(\mathsf{rscale}_R(a, q), \mathsf{shift}_R(\mathsf{rmul}_R(q, x)))).$$

Right-cons expansion applied to q, a, x gives the same middle endpoint classified with $\mathsf{rmul}_R(q, \mathsf{cons}(a, x))$. Symmetry and transitivity of $\mathsf{PolySame}_R$ compose the two comparisons, completing the induction. □

Theorem 50.155 (Polynomial multiplication has left zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$ and zero endpoint 0_R. Let trim_R be the canonical polynomial trim from Definition 50.20 and Definition 50.22, and let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $q : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, q), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative q from Definition 50.3.

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left zero classified`

Proof. Set $q_0 = \mathsf{trim}_R(q)$. The empty zero-polynomial clause in Definition 50.22 gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Hence Definition 50.139 gives

$$\begin{aligned}\mathsf{PolyMul}_R(\mathsf{nil}, q) &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{nil}), q_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, q_0)) \\ &= \mathsf{trim}_R(\mathsf{nil}) \\ &= \mathsf{nil}.\end{aligned}$$

The third equality is the nil clause for rmul_R in Definition 50.139.

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{PolyMul}_R(\mathsf{nil}, q)), \mathsf{trim}_R(\mathsf{nil})).$$

The displayed computation and the empty zero-polynomial clause reduce both displayed representatives to nil. The nil-nil clause of Definition 36.12 supplies the structural classifier witness. □

Definition 50.156 (Padded polynomial coefficients and finite Cauchy folds)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and finite coefficient spines from Definition 36.1. For a finite coefficient spine $x : \mathsf{ListCarrier}(R)$ and $n : \mathsf{Nat}^{\uparrow}$, write $x[n]$ for the padded coefficient defined by*

$$\mathsf{nil}[n] := 0_R, \qquad \mathsf{cons}(a,t)[0] := a, \qquad \mathsf{cons}(a,t)[n+1] := t[n].$$

For finite coefficient spines x, y, z define the finite Cauchy coefficient fold and the finite triple coefficient fold by

$$\mathsf{Cauchy}_R(x,y;n) := \Sigma_{i+j=n} x[i] \cdot y[j], \qquad \mathsf{Triple}_R(x,y,z;n) := \Sigma_{i+j+k=n}(x[i] \cdot y[j]) \cdot z[k].$$

The displayed sums use the finite additive fold of Definition 50.26 over the lexicographic finite spines of the displayed natural-number tuples.

Theorem 50.157 (Polynomial multiplication has right zero absorption)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R. Let $\mathsf{PolySame}_R$, rmul_R, and $\mathsf{PolyMul}_R$ be as in Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Consequently the same conclusion holds for every normalized polynomial representative p from Definition 50.3.

Proof. Set

$$p_0 := \mathsf{trim}_R(p), \qquad n_0 := \mathsf{trim}_R(\mathsf{nil}).$$

The empty zero-polynomial clause in Definition 50.22 gives $n_0 = \mathsf{nil}$. Unfolding the normalized product in Definition 50.139 gives

$$\begin{aligned} \mathsf{PolyMul}_R(p, \mathsf{nil}) &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, n_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil})) \end{aligned}$$

and

$$\begin{aligned} \mathsf{PolyMul}_R(\mathsf{nil}, p) &= \mathsf{trim}_R(\mathsf{rmul}_R(n_0, p_0)) \\ &= \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0)). \end{aligned}$$

Apply Definition 50.154 to the finite coefficient spines p_0 and nil. It gives

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(p_0, \mathsf{nil}), \mathsf{rmul}_R(\mathsf{nil}, p_0)).$$

After unfolding $\mathsf{PolySame}_R$ by Definition 50.30, this is the structural classifier comparison

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{trim}_R(\mathsf{rmul}_R(p_0, \mathsf{nil})), \mathsf{trim}_R(\mathsf{rmul}_R(\mathsf{nil}, p_0))).$$

Using the two displayed product computations and trim idempotence from Definition 50.22, this is exactly

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{PolyMul}_R(\mathsf{nil}, p)).$$

On the other hand, Definition 50.155 applied to p gives

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{nil}, p), \mathsf{nil}).$$

Unfold the two $\mathsf{PolySame}_R$ witnesses and compose their structural list-classifier witnesses by Definition 36.15, using the transitivity field of cert_R on scalar heads. Folding back Definition 50.30 yields

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(p, \mathsf{nil}), \mathsf{nil}).$$

Every normalized polynomial representative from Definition 50.3 is, in particular, a finite coefficient spine, so the same argument applies to that carrier representative. □

Lemma 50.158 (Padded coefficient extensionality for the polynomial classifier)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and canonical trim trim_R from Definition 50.20 and Definition 50.22. If finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ satisfy $x[n] \sim_R y[n]$ for every $n : \mathsf{Nat}^{\uparrow}$, then*

$$\mathsf{PolySame}_R(x, y).$$

Proof. Put $x_0 := \mathsf{trim}_R(x)$ and $y_0 := \mathsf{trim}_R(y)$. The trailing-zero removal clause in Definition 50.22 gives decompositions $x = x_0{+}{+}t_x$ and $y = y_0{+}{+}t_y$, with $\mathsf{ZeroRem}_R(t_x)$ and $\mathsf{ZeroRem}_R(t_y)$ in the sense of Definition 50.28. Hence $x[n] \sim_R x_0[n]$ and $y[n] \sim_R y_0[n]$ for every n: inside the trimmed prefix the coefficient is unchanged, inside the removed tail the zero-remainder witness classifies the coefficient with 0_R, and after the whole spine both sides are the padding coefficient 0_R. Classifier symmetry and transitivity from cert_R give $x_0[n] \sim_R y_0[n]$ for every n.

The trimmed spines are normalized representatives. If one normalized representative had a last displayed coefficient beyond the length of the other, the padded coefficient comparison at that last index would classify that displayed coefficient with 0_R, contradicting the normalized no-trailing-zero condition in Definition 50.22. Thus x_0 and y_0 have the same finite length. Induction over this common length builds a structural classifier witness: at each head position the hypothesis $x_0[n] \sim_R y_0[n]$ supplies the scalar head comparison, and the induction hypothesis supplies the tail classifier. Therefore

$$\mathsf{ListClassifierSpec}(\sim_R, x_0, y_0).$$

Unfolding $\mathsf{PolySame}_R$ by Definition 50.30 gives the displayed polynomial classifier comparison. □

Lemma 50.159 (Raw Cauchy product coefficient formula)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(x, y)[n] \sim_R \mathsf{Cauchy}_R(x, y; n).$$

Proof. Read the scalar source as a $\mathsf{Ring}^{\uparrow}(R)$ source by Definition 44.8. The recursion defining radd_R gives the padded coefficient formula $\mathsf{radd}_R(u, v)[n] \sim_R u[n] + v[n]$ by induction on the two finite spines. The definitions of rscale_R and shift_R give

$$\mathsf{rscale}_R(a, u)[n] \sim_R a \cdot u[n], \qquad \mathsf{shift}_R(u)[0] \sim_R 0_R, \qquad \mathsf{shift}_R(u)[n+1] \sim_R u[n].$$

These comparisons use only zero self-classification, additive congruence, and the additive zero laws from Definition 43.26.

Induct on the first spine x. If $x = \mathsf{nil}$, the raw product is nil by Definition 50.139. Every summand in $\mathsf{Cauchy}_R(\mathsf{nil}, y; n)$ has the form $0_R \cdot y[j]$ and is classified with 0_R by the zero-multiplication row in Definition 43.26. The zero-summand fold theorem Definition 50.27 classifies the Cauchy fold with 0_R, which is the padded coefficient of nil.

For the cons step, write $x = \mathsf{cons}(a, x')$. At coefficient 0, the raw product recursion gives the sum of $a \cdot y[0]$ and the zero head of the shifted recursive product; the additive right-zero law classifies this with $a \cdot y[0]$, which is $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; 0)$. At coefficient $n+1$, the same recursion and the auxiliary coefficient formulas give

$$\mathsf{rmul}_R(\mathsf{cons}(a, x'), y)[n+1] \sim_R a \cdot y[n+1] + \mathsf{rmul}_R(x', y)[n].$$

The induction hypothesis and additive congruence replace the second summand by $\mathsf{Cauchy}_R(x', y; n)$. Splitting the finite index spine $i+j = n+1$ into the head case $i = 0$ and the shifted cases $i = i'+1$ identifies the resulting fold with $\mathsf{Cauchy}_R(\mathsf{cons}(a, x'), y; n+1)$ by Definition 50.26. □

Lemma 50.160 (Raw Cauchy product coefficient associativity)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, and the raw Cauchy product data of Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$ and every $n : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{rmul}_R(\mathsf{rmul}_R(x, y), z)[n] \sim_R \mathsf{rmul}_R(x, \mathsf{rmul}_R(y, z))[n].$$

Proof. Fix n. By Definition 50.159, the left coefficient is classified with $\mathsf{Cauchy}_R(\mathsf{rmul}_R(x, y), z; n)$. For each outer index $s+k = n$, apply Definition 50.159 to replace $\mathsf{rmul}_R(x, y)[s]$ by $\mathsf{Cauchy}_R(x, y; s)$. Multiplication congruence and Definition 50.26 lift these replacements through the outer finite fold. The right-multiplication half of Definition 49.95 then classifies $\mathsf{Cauchy}_R(x, y; s) \cdot z[k]$ with the inner fold over (i, j) satisfying $i+j = s$ of $(x[i] \cdot y[j]) \cdot z[k]$. The nested finite index spine $i+j = s$, $s+k = n$ is the same finite triple spine $i+j+k = n$ in lexicographic order after deleting the bookkeeping index s. The finite fold swap certificate of Definition 49.94, together with fold congruence, classifies the resulting nested fold with $\mathsf{Triple}_R(x, y, z; n)$. Thus the left raw product coefficient is classified with $\mathsf{Triple}_R(x, y, z; n)$.

The right coefficient is handled symmetrically. The coefficient formula first gives $\mathsf{Cauchy}_R(x, \mathsf{rmul}_R(y, z); n)$. Replacing each coefficient of $\mathsf{rmul}_R(y, z)$ by its Cauchy fold and applying the left-multiplication half of Definition 49.95 gives the finite fold over triples $i+j+k = n$ of $x[i] \cdot (y[j] \cdot z[k])$. Scalar multiplication associativity from Definition 43.26 classifies each displayed summand with $(x[i] \cdot y[j]) \cdot z[k]$. Fold congruence therefore classifies the right coefficient with the same $\mathsf{Triple}_R(x, y, z; n)$. Symmetry and transitivity of the scalar classifier compose the two comparisons. □

Theorem 50.161 (Polynomial multiplication associativity over commutative-ring scalars)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the canonical trim be trim_R from Definition 50.22, and let rmul_R and $\mathsf{PolyMul}_R$ be the raw and normalized products of Definition 50.139. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p, q), r), \mathsf{PolyMul}_R(p, \mathsf{PolyMul}_R(q, r))).$$

Proof. Put $p_0 := \mathsf{trim}_R(p)$, $q_0 := \mathsf{trim}_R(q)$, and $r_0 := \mathsf{trim}_R(r)$. Set $A := \mathsf{PolyMul}_R(\mathsf{PolyMul}_R(p, q), r)$ and $B := \mathsf{PolyMul}_R(p, \mathsf{PolyMul}_R(q, r))$. By Definition 50.158, it is enough to prove $A[n] \sim_R B[n]$ for every $n : \mathsf{Nat}^{\uparrow}$.

Trim preserves padded coefficients by the trailing-zero decomposition in Definition 50.22: for every finite spine u and every n, $u[n] \sim_R \mathsf{trim}_R(u)[n]$. The coefficient formula Definition 50.159 is functorial in both inputs with respect to such pointwise padded-coefficient comparisons, since multiplication congruence compares each Cauchy summand and Definition 50.26 lifts the summand comparisons through the finite fold.

Unfolding Definition 50.139 and using trim idempotence from Definition 50.22, the left normalized product has the same padded coefficients as $\mathsf{rmul}_R(\mathsf{trim}_R(\mathsf{rmul}_R(p_0, q_0)), r_0)$. By trim preservation and the functoriality just recorded, its nth coefficient is classified with $\mathsf{rmul}_R(\mathsf{rmul}_R(p_0, q_0), r_0)[n]$. Similarly, the right normalized product has the same padded coefficients as $\mathsf{rmul}_R(p_0, \mathsf{trim}_R(\mathsf{rmul}_R(q_0, r_0)))$, and its nth coefficient is classified with $\mathsf{rmul}_R(p_0, \mathsf{rmul}_R(q_0, r_0))[n]$.

Applying Definition 50.160 to p_0, q_0, r_0 gives

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p_0, q_0), r_0)[n] \sim_R \mathsf{rmul}_R(p_0, \mathsf{rmul}_R(q_0, r_0))[n].$$

Classifier symmetry and transitivity compose the two trim comparisons with this raw coefficient associativity comparison, yielding $A[n] \sim_R B[n]$ for the fixed n. Since n was arbitrary, Definition 50.158 gives $\mathsf{PolySame}_R(A, B)$, which is the displayed associativity statement. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul associative addFold classified`

Theorem 50.162 (Polynomial singleton evaluation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, zero endpoint 0_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and Horner evaluation as in Definition 50.133. For every carried scalar coefficient c and every carried scalar point α,*

$$\mathsf{Eval}_{\alpha,R}(\mathsf{cons}(c, \mathsf{nil})) \sim_R c.$$

Proof. Put $s = \mathsf{cons}(c, \mathsf{nil})$. First compute the unnormalized Horner value. The cons and nil recursion clauses in Definition 50.133 identify

$$\mathsf{eval}_{\alpha,R}(s) = c + \alpha \cdot 0_R.$$

The right zero-absorption half of Definition 43.60 gives

$$\alpha \cdot 0_R \sim_R 0_R.$$

Classifier reflexivity for c and addition congruence from Definition 43.26 give

$$c + \alpha \cdot 0_R \sim_R c + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $c + 0_R \sim_R c$, and transitivity yields

$$\mathsf{eval}_{\alpha,R}(s) \sim_R c.$$

By Definition 50.133, $\mathsf{Eval}_{\alpha,R}(s) = \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(s))$. The one-coefficient trimming boundary in Definition 50.166, read with Definition 50.22, has exactly the empty zero branch and the retained singleton branch. In the empty zero branch, $c \sim_R 0_R$ and $\mathsf{trim}_R(s) = \mathsf{nil}$; the nil Horner equation gives $\mathsf{Eval}_{\alpha,R}(s) = 0_R$, and classifier symmetry gives $0_R \sim_R c$. In the retained singleton branch, $\mathsf{trim}_R(s) = s$, so the raw computation above is the required normalized comparison. These are the two canonical trim branches for a singleton coefficient spine. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonEval singleton`

Theorem 50.163 (Raw polynomial multiplication left-distributes over raw addition)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, and let radd_R, rscale_R, shift_R, and rmul_R be the raw operations of Definition 50.30 and Definition 50.139. For finite coefficient spines $x, y, z : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z)), \mathsf{radd}_R(\mathsf{rmul}_R(x, y), \mathsf{rmul}_R(x, z))).$$

Consequently the same classifier comparison holds when x, y, z are normalized polynomial representatives from Definition 50.3.

Lean4 ✓ BEDC.Derived.PolynomialUp.PolynomialSingletonRawMul left distributes over raw add

Proof. By Definition 50.158, it is enough to compare padded coefficients. First record the padded coefficient law for raw addition: for all finite coefficient spines u, v and every $m : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u, v)[m] \sim_R u[m] + v[m].$$

This is a direct induction over the two finite spines using the four recursion clauses in Definition 50.30. In the mixed cons-nil and nil-cons cases the missing coefficient is the padding endpoint 0_R; in the nil-nil tail the comparison is between 0_R and $0_R + 0_R$, handled by the additive zero laws and classifier symmetry from cert_R.

Fix $n : \mathsf{Nat}^{\uparrow}$. Let the finite split spine for $i + j = n$ carry three scalar spines

$$A_n(i, j) := x[i] \cdot \mathsf{radd}_R(y, z)[j], \qquad B_n(i, j) := x[i] \cdot y[j], \qquad C_n(i, j) := x[i] \cdot z[j].$$

For each split $i + j = n$, the raw-addition coefficient law gives

$$\mathsf{radd}_R(y, z)[j] \sim_R y[j] + z[j].$$

Multiplication congruence transports this under left multiplication by $x[i]$, and scalar left distributivity from Definition 43.26 gives

$$x[i] \cdot \mathsf{radd}_R(y, z)[j] \sim_R (x[i] \cdot y[j]) + (x[i] \cdot z[j]).$$

The matching split order assembles these scalar comparisons into a structural list-classifier witness from Definition 36.12 between A_n and the pointwise sum spine $B_n + C_n$. Applying Definition 50.26 to this witness gives

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n) \sim_R \mathsf{fold}_{+,R}(B_n + C_n).$$

The finite fold splitting lemma Definition 49.98 then classifies the right endpoint with the sum of the two separate Cauchy folds:

$$\mathsf{fold}_{+,R}(B_n + C_n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

Thus

$$\mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n) \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

The raw Cauchy coefficient formula Definition 50.159 gives

$$\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z))[n] \sim_R \mathsf{Cauchy}_R(x, \mathsf{radd}_R(y, z); n).$$

For the other endpoint, the raw-addition coefficient law gives

$$\mathsf{radd}_R(\mathsf{rmul}_R(x, y), \mathsf{rmul}_R(x, z))[n] \sim_R \mathsf{rmul}_R(x, y)[n] + \mathsf{rmul}_R(x, z)[n].$$

Applying Definition 50.159 to the two raw products and then additive congruence gives

$$\mathsf{rmul}_R(x, y)[n] + \mathsf{rmul}_R(x, z)[n] \sim_R \mathsf{Cauchy}_R(x, y; n) + \mathsf{Cauchy}_R(x, z; n).$$

Classifier symmetry and transitivity from cert_R compose the three displayed coefficient comparisons, yielding

$$\mathsf{rmul}_R(x, \mathsf{radd}_R(y, z))[n] \sim_R \mathsf{radd}_R(\mathsf{rmul}_R(x, y), \mathsf{rmul}_R(x, z))[n].$$

Since n was arbitrary, Definition 50.158 gives the displayed $\mathsf{PolySame}_R$ comparison. A normalized polynomial representative is a finite coefficient spine by Definition 50.3, so the final sentence is the same statement restricted to those representatives. □

Theorem 50.164 (Polynomial raw multiplication has singleton-one left identity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with classifier $\sim_R$, scalar certificate $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+$, multiplication $\cdot$, canonical trim trim_R, and the raw operations radd_R, rscale_R, shift_R, and rmul_R from Definition 50.30 and Definition 50.139. For every finite coefficient spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p), p).$$

Proof. By Definition 50.158, it is enough to prove equality under the padded scalar classifier at every coefficient index. Unfolding the cons clause and the nil clause of raw multiplication in Definition 50.139 gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p) = \mathsf{radd}_R(\mathsf{rscale}_R(1_R, p), \mathsf{cons}(0_R, \mathsf{nil})).$$

Thus the product is the raw sum of the coefficientwise scalar multiple by 1_R and the one-place shift of the empty recursive product.

First, for every finite coefficient spine u and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{radd}_R(u, \mathsf{cons}(0_R, \mathsf{nil}))[n] \sim_R u[n].$$

This is induction on u. If $u = \mathsf{nil}$, the only displayed head is $0_R + 0_R$, classified with 0_R by the additive zero rows in Definition 43.26; all later coefficients are padding coefficients 0_R. If $u = \mathsf{cons}(a, u')$, the head comparison is $(a + 0_R) \sim_R a$ by the additive right-zero row, while the tail comparison is the raw-addition right-nil induction that repeatedly classifies $b + 0_R$ with b and ends at the padding coefficient. These steps use additive congruence, zero self-classification, and classifier transitivity from cert_R.

Second, for every finite coefficient spine p and every index $n : \mathsf{Nat}^{\uparrow}$,

$$\mathsf{rscale}_R(1_R, p)[n] \sim_R p[n].$$

Induct on p. In the nil case both padded coefficients are 0_R. In the cons case, the exposed head is $1_R \cdot a$, classified with a by the multiplicative left-unit row retained in Definition 43.26, and the tail is supplied by the induction hypothesis. After the displayed spine ends, both sides are again the padding coefficient 0_R.

Apply the first coefficient comparison to $u = \mathsf{rscale}_R(1_R, p)$ and compose it with the second comparison. For every $n : \mathsf{Nat}^{\uparrow}$ this gives

$$\mathsf{rmul}_R(\mathsf{cons}(1_R, \mathsf{nil}), p)[n] \sim_R p[n].$$

The padded coefficient extensionality lemma Definition 50.158 converts these pointwise scalar comparisons into the displayed $\mathsf{PolySame}_R$ statement. □

Theorem 50.165 (Raw polynomial multiplication associativity up to add-trim)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. Let the classifier be $\sim_R$, the zero endpoint be 0_R, the scalar operations be $+$ and $\cdot$, the scalar certificate be $\mathsf{cert}_R : \mathsf{NameCert}(R, \sim_R)$, the canonical trim be trim_R from Definition 50.22, and let $\mathsf{PolySame}_R$ and rmul_R be the add-trim classifier and raw Cauchy product of Definition 50.30 and Definition 50.139. For finite coefficient spines $p, q, r : \mathsf{ListCarrier}(R)$,*

$$\mathsf{PolySame}_R(\mathsf{rmul}_R(\mathsf{rmul}_R(p, q), r), \mathsf{rmul}_R(p, \mathsf{rmul}_R(q, r))).$$

Consequently the same comparison holds when p, q, r are normalized polynomial representatives from Definition 50.3.

Proof. Set

$$A := \mathsf{rmul}_R(\mathsf{rmul}_R(p,q),r), \qquad B := \mathsf{rmul}_R(p,\mathsf{rmul}_R(q,r)).$$

By Definition 50.158, it is enough to prove $A[n] \sim_R B[n]$ for every $n : \mathsf{Nat}^{\uparrow}$.

Fix $n : \mathsf{Nat}^{\uparrow}$. The raw Cauchy coefficient associativity lemma Definition 50.160, applied to the finite coefficient spines p, q, r, gives exactly

$$\mathsf{rmul}_R(\mathsf{rmul}_R(p,q),r)[n] \sim_R \mathsf{rmul}_R(p,\mathsf{rmul}_R(q,r))[n].$$

This is $A[n] \sim_R B[n]$. Since n was arbitrary, Definition 50.158 converts the pointwise padded coefficient comparisons into

$$\mathsf{PolySame}_R(A,B).$$

Unfolding the definitions of A and B gives the displayed add-trim classifier comparison. A normalized polynomial representative from Definition 50.3 is a finite coefficient spine, so the final sentence is the same statement restricted to those carried representatives. □

Proof. Put $s = \mathsf{cons}(c, \mathsf{nil})$. First compute the unnormalized Horner value. The cons and nil recursion clauses in Definition 50.133 identify

$$\mathsf{eval}_{\alpha,R}(s) = c + \alpha \cdot 0_R.$$

The right zero-absorption half of Definition 43.60 gives

$$\alpha \cdot 0_R \sim_R 0_R.$$

Classifier reflexivity for c and addition congruence from Definition 43.26 give

$$c + \alpha \cdot 0_R \sim_R c + 0_R.$$

The additive right-zero theorem Definition 43.31 gives $c + 0_R \sim_R c$, and transitivity yields

$$\mathsf{eval}_{\alpha,R}(s) \sim_R c.$$

By Definition 50.133, $\mathsf{Eval}_{\alpha,R}(s) = \mathsf{eval}_{\alpha,R}(\mathsf{trim}_R(s))$. The one-coefficient trimming boundary in Definition 50.166, read with Definition 50.22, has exactly the empty zero branch and the retained singleton branch. In the empty zero branch, $c \sim_R 0_R$ and $\mathsf{trim}_R(s) = \mathsf{nil}$; the nil Horner equation gives $\mathsf{Eval}_{\alpha,R}(s) = 0_R$, and classifier symmetry gives $0_R \sim_R c$. In the retained singleton branch, $\mathsf{trim}_R(s) = s$, so the raw computation above is the required normalized comparison. These are the two canonical trim branches for a singleton coefficient spine. □

Definition 50.166 (Polynomial normalized singleton representative)**.** *Let R be a scalar carrier with classifier $\sim_R$ and zero endpoint 0_R. For a scalar coefficient c, define*

$$\mathsf{NormOne}_R(c) \iff \neg(c \sim_R 0_R).$$

Thus the singleton coefficient spine $\mathsf{cons}(c, \mathsf{nil})$ *is a normalized singleton representative exactly when* $\mathsf{NormOne}_R(c)$ *holds, while* nil *is the empty zero-polynomial representative. This is the one-coefficient boundary of the trailing-zero convention in Definition 50.3 and the canonical trimming rule in Definition 50.22.*

Lean4 (definition) `BEDC.Derived.PolynomialUp.PolynomialSingletonNormalizedRepresentative`

Proposition 50.167 (Raw polynomial addition leaves singleton normal form)**.** *Let R be a scalar ring carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+$, and canonical trim trim_R as in Definition 50.20 and Definition 50.22. Let radd_R and $\mathsf{PolySame}_R$ be the raw addition and normalized raw classifier of Definition 50.30. For scalar sources presented through $\mathsf{CommRing}^{\uparrow}$, Definition 44.8 reads these as the retained $\mathsf{Ring}^{\uparrow}$ addition, zero endpoint, and classifier.*

Suppose a and b are classified nonzero and add to a coefficient classified as zero:

$$\neg(a \sim_R 0_R), \qquad \neg(b \sim_R 0_R), \qquad (a+b) \sim_R 0_R.$$

Set

$$p := \mathsf{cons}(a, \mathsf{nil}), \qquad q := \mathsf{cons}(b, \mathsf{nil}).$$

Then p and q are normalized singleton representatives, but their raw zero-padded sum is not a normalized singleton representative:

$$\mathsf{NormOne}_R(a) \wedge \mathsf{NormOne}_R(b),$$

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{nil}),$$

$$\neg\mathsf{NormOne}_R(a+b).$$

Moreover its canonical normalized representative is the empty zero representative, and this is exactly what the polynomial classifier records:

$$\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}, \qquad \mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{nil}).$$

Proof. By Definition 50.166, the assumptions $\neg(a \sim_R 0_R)$ and $\neg(b \sim_R 0_R)$ are precisely the witnesses $\mathsf{NormOne}_R(a)$ and $\mathsf{NormOne}_R(b)$, so p and q are normalized singleton representatives.

Unfolding p and q, the cons-cons branch of the raw addition recursion in Definition 50.30 gives

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{radd}_R(\mathsf{nil}, \mathsf{nil})).$$

The nil-nil branch gives $\mathsf{radd}_R(\mathsf{nil}, \mathsf{nil}) = \mathsf{nil}$, hence

$$\mathsf{radd}_R(p, q) = \mathsf{cons}(a+b, \mathsf{nil}).$$

The coefficient hypothesis $(a+b) \sim_R 0_R$ contradicts any witness of $\mathsf{NormOne}_R(a+b)$, since such a witness is a proof of $\neg((a+b) \sim_R 0_R)$. Therefore $\neg\mathsf{NormOne}_R(a+b)$, so the raw singleton spine is not admitted as a normalized singleton representative.

The same coefficient hypothesis activates the trailing-zero clause of the canonical trim from Definition 50.22. Therefore the singleton spine $\mathsf{cons}(a+b, \mathsf{nil})$ trims to nil, and the raw-sum computation above gives

$$\mathsf{trim}_R(\mathsf{radd}_R(p, q)) = \mathsf{nil}.$$

The empty zero-polynomial clause gives $\mathsf{trim}_R(\mathsf{nil}) = \mathsf{nil}$. Unfolding $\mathsf{PolySame}_R$ from Definition 50.30, the classifier goal is

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{nil}, \mathsf{nil}),$$

which is the nil-nil constructor of the finite-spine classifier from Definition 36.12, as isolated by Definition 36.13. Hence $\mathsf{PolySame}_R(\mathsf{radd}_R(p, q), \mathsf{nil})$ holds. □

Lean4 ✓ `BEDC.Derived.PolynomialUp.PolynomialSingletonRawAdd singleton normal form boundary`

Definition 50.168 (Polynomial singleton constant-source schema instance)**.** *The concrete constant-source instance for* $\mathsf{Poly}^{\uparrow}$ *uses the singleton empty-history polynomial carrier. Its coefficient source is the existing singleton scalar carrier, its polynomial constructor is the constant singleton coefficient row* $\mathsf{cons}(c, \mathsf{nil})$*, and its evaluation row is the singleton evaluation readback* PolynomialSingletonEval *at the displayed evaluation endpoint. The classifier is the singleton polynomial classifier on these* BHist *rows, not an abstract coefficient equivalence.*

Theorem 50.169 (Polynomial singleton constant evaluation boundary)**.** *Every constant-source singleton polynomial from Definition 50.168 evaluates to the displayed singleton coefficient endpoint. For a coefficient row* c *and any singleton evaluation endpoint* α*, the readback of the constant spine* $\mathsf{cons}(c, \mathsf{nil})$ *is classified by the singleton polynomial classifier as the same endpoint* c*; at the empty-history endpoint, the zero constant-term readback gives the same classifier boundary. Thus the evaluation classifier exported by this concrete instance is exactly the singleton coefficient classifier.*

Proof. The carrier is the one-coefficient boundary isolated by Definition 50.166 and Definition 50.167: a constant-source row is presented by the singleton spine $\mathsf{cons}(c, \mathsf{nil})$, with no additional coefficient family or abstract scalar quotient.

The singleton evaluation readback applies the checked singleton row PolynomialSingletonEval_singleton, which identifies evaluation of $\mathsf{cons}(c, \mathsf{nil})$ with the coefficient endpoint c under the singleton polynomial classifier. At the empty-history endpoint, PolynomialSingletonEval_zero_constant_term gives the same endpoint boundary for the zero constant-term readback.

Therefore the only classifier visible in the concrete constant-source schema is the singleton polynomial classifier already attached to the coefficient row. Repacking the carrier, constructor, and evaluation readback gives the stated boundary. □

Chapter 51

A Concrete Naming Certificate for $\mathsf{FPS}^{\uparrow}$

The formal power series interface $\mathsf{FPS}^{\uparrow}$ has certificate obligations over a scalar ring from chapter 43 as functions $\mathbb{N} \to R$, interpreted as coefficient sequences without termination. The classifier is pointwise sameness on every index. Formal series addition and Cauchy product are derived operations governed by the ring's certificate fields. Convergence is not asserted: this is the formal carrier, not analytic series.

51.1 Carrier definition

Definition 51.1 (FormalPowerSeries carrier)**.** *Use coefficient functions* $\mathsf{Nat}^{\uparrow} \to R$ *over a certified scalar ring* R*.*

51.2 Concrete singleton zero formal power series instance

Definition 51.2 (Singleton zero formal power series instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$*. Use the singleton empty-history scalar ring of Definition 43.5. Define the series classifier*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Set the formal-power-series code carrier to A_{e} *on* BHist *endpoints. Represent the zero formal power series by* $Z_{\mathsf{e}} := \mathsf{e}$*, and set the unit formal power series to* $U_{\mathsf{e}} := \mathsf{e}$*. Define the coefficient readout and displayed operations by*

$$\mathsf{coeff}_{\mathsf{e}}(F, n) := \mathsf{e}, \qquad F \oplus_{\mathrm{fps}} G := \mathsf{e}, \qquad F \odot_{\mathrm{fps}} G := \mathsf{e}.$$

Thus every coefficient, zero endpoint, unit endpoint, addition endpoint, and Cauchy-product endpoint is the singleton scalar endpoint e*.*

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryCarrier`
Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier`

Theorem 51.3 (Singleton zero formal power series laws)**.** *The tuple of Definition 51.2 satisfies the* $\mathsf{FPS}^{\uparrow}$ *law package over the concrete* BHist *carrier:*

1. *Z_{e} and U_{e} are carried by A_{e}.*
2. *Every coefficient query $\mathsf{coeff}_{\mathsf{e}}(F, n)$ is carried by the singleton empty-history scalar ring.*
3. *If $A_{\mathsf{e}}(F)$ and $A_{\mathsf{e}}(G)$, then $A_{\mathsf{e}}(F \oplus_{\mathrm{fps}} G)$ and $A_{\mathsf{e}}(F \odot_{\mathrm{fps}} G)$.*
4. *ρ_{e} is reflexive, symmetric, and transitive on carried series codes.*
5. *Addition is congruent under ρ_{e} and has Z_{e} as left and right zero under ρ_{e}.*
6. *Cauchy product is congruent under ρ_{e} and has U_{e} as left and right unit under ρ_{e}.*
7. *The coefficient ledger of every displayed endpoint is the constant singleton scalar endpoint e, and the scalar clauses are those of Theorem 43.6.*

No convergence field is required.

Proof. The carrier endpoint is supplied by Definition 36.169 at $a_0 = \mathsf{e}$. In particular, Z_{e}, U_{e}, and every displayed operation endpoint reduce to the same carried endpoint e.

For coefficient queries, every value of $\mathsf{coeff}_{\mathsf{e}}(F, n)$ is definitionally e. The scalar side is the singleton empty-history ring of Definition 43.5, and Theorem 43.6 supplies its scalar zero, scalar one, addition, and multiplication clauses.

For addition and Cauchy-product closure, the displayed operations return e for every pair of inputs. The singleton carrier witness therefore discharges both closure obligations without inspecting the input series or the index used in a coefficient view.

Classifier reflexivity, symmetry, and transitivity are inherited from the hsame reflexivity, symmetry, and transitivity laws in the FKernel, together with the two singleton endpoint witnesses required by ρ_{e}. Operation congruence reduces both addition results and both Cauchy-product results to e, so the relation component is again $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

The zero laws and unit laws reduce to comparisons of e with e. The same ρ_{e} package gives the left and right zero laws for addition and the left and right unit laws for Cauchy product.

Finally, every coefficient query returns e. Hence the coefficient ledger for the zero series, unit series, sum, and Cauchy product is constant, with no convergence claim involved. □

Lean4 ✓ `BEDC.Derived.FpsUp.fps singleton empty schema laws`

Theorem 51.4 (Singleton coefficient endpoint has empty ledger)**.** *For every displayed singleton formal power series endpoint F and coefficient index n, if F is carried by the singleton source, then*

$$\mathsf{hsame}(\mathsf{coeff}_{\mathsf{e}}(F, n), \mathsf{e})$$

and the coefficient endpoint is classified with the empty scalar endpoint by ρ_{e}.

Proof. The coefficient operation of the singleton formal-power-series instance is definitionally constant at e. The history-sameness part is therefore $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, and the classifier clause is the triple consisting of the two singleton carrier witnesses for e and the same reflexive history-sameness witness. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCoeff empty ledger`

The $\mathsf{FPS}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and constant coefficient operations, so future Lean targets can be anchored directly in BHist.

Theorem 51.5 (Singleton empty-history FPS semantic name certificate)**.** *The singleton empty-history formal-power-series carrier* $A_{\mathsf{e}}(h) := h\mathsf{hsamee}$ *with classifier*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge h\mathsf{hsame}k$$

forms a $\mathsf{SemanticNameCert}(A_{\mathsf{e}}, A_{\mathsf{e}}, A_{\mathsf{e}}, \rho_{\mathsf{e}})$*.*

Proof. The carrier is inhabited by e with hsame reflexivity. Reflexivity, symmetry, and transitivity of the classifier use the corresponding hsame laws, while the two carrier components are part of the classifier witness. Carrier transport across the classifier reads out the right carrier component. Pattern and ledger soundness are identity maps because the source, pattern, and ledger predicates are the same singleton carrier. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistory semanticNameCert`

Theorem 51.6 (Singleton FPS classifier append split empty)**.** *For* $p, q, h : \mathsf{BHist}$*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(p, q), h) \iff p\mathsf{hsamee} \wedge q\mathsf{hsamee} \wedge A_{\mathsf{e}}(h).$$

Proof. In the forward direction, the left carrier component of $\rho_{\mathsf{e}}(\mathsf{append}(p, q), h)$ says $\mathsf{append}(p, q)\mathsf{hsamee}$. The append-empty inversion theorem splits this into $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$, while the right carrier component already gives $A_{\mathsf{e}}(h)$.

Conversely, append-empty introduction turns $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$ into $A_{\mathsf{e}}(\mathsf{append}(p, q))$. Together with $A_{\mathsf{e}}(h)$, transitivity through e supplies the classifier comparison. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier append split empty iff`

Theorem 51.7 (Singleton FPS classifier pair append split empty)**.** *For* $p, q, r, s : \mathsf{BHist}$*,*

$$\rho_{\mathsf{e}}(\mathsf{append}(p, q), \mathsf{append}(r, s)) \iff p\mathsf{hsamee} \wedge q\mathsf{hsamee} \wedge r\mathsf{hsamee} \wedge s\mathsf{hsamee}.$$

Proof. The forward direction applies the one-sided append split theorem to the left endpoint. This gives $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, and the right carrier fact $A_{\mathsf{e}}(\mathsf{append}(r, s))$. The append-empty inversion theorem then splits the right carrier fact into $r\mathsf{hsamee}$ and $s\mathsf{hsamee}$.

Conversely, append-empty introduction turns the two left endpoint witnesses into $A_{\mathsf{e}}(\mathsf{append}(p, q))$ and the two right endpoint witnesses into $A_{\mathsf{e}}(\mathsf{append}(r, s))$. Transitivity of hsame through e supplies the classifier comparison between the two appended histories. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier append pair split empty iff`

Theorem 51.8 (Singleton FPS classifier continuation split empty)**.** *For* $p, q, h : \mathsf{BHist}$*, if* $\mathsf{Cont}(p, q, h)$ *and* $\rho_{\mathsf{e}}(h, \mathsf{e})$*, then*

$$A_{\mathsf{e}}(p) \wedge A_{\mathsf{e}}(q).$$

Proof. Invert the continuation ledger to identify h with $\mathsf{append}(p, q)$. The classifier premise then has the shape handled by Theorem 51.6; projecting its forward direction gives the two empty-history carrier facts for p and q. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier cont split empty`

Theorem 51.9 (Singleton FPS classifier continuation component classifiers)**.** *For* $p, q, h : \mathsf{BHist}$*, if* $\mathsf{Cont}(p, q, h)$ *and* $\rho_{\mathsf{e}}(h, \mathsf{e})$*, then both continuation components are classified with the empty endpoint:*

$$\rho_{\mathsf{e}}(p, \mathsf{e}) \wedge \rho_{\mathsf{e}}(q, \mathsf{e}).$$

Proof. Invert the continuation ledger to put the classifier on $\mathsf{append}(p, q)$. The append split exposes $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$, and each exposed carrier combines with reflexivity of e to rebuild the two component classifiers. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier cont component classifiers`

Theorem 51.10 (Singleton FPS classifier append component classifiers)**.** *For $p, q, h : \mathsf{BHist}$, if the appended endpoint is classified with h,*

$$\rho_{\mathsf{e}}(\mathsf{append}(p, q), h),$$

then both append components are classified with the empty endpoint and the right endpoint is carried:

$$\rho_{\mathsf{e}}(p, \mathsf{e}) \wedge \rho_{\mathsf{e}}(q, \mathsf{e}) \wedge A_{\mathsf{e}}(h).$$

Proof. Apply the append split classifier equivalence of Definition 51.6. This exposes $p\mathsf{hsamee}$, $q\mathsf{hsamee}$, and $A_{\mathsf{e}}(h)$. Each component classifier is then rebuilt from its exposed empty-history carrier, the reflexive carrier of e, and the same exposed history-sameness witness. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier append component classifiers`

Theorem 51.11 (Singleton FPS classifier rejects headed endpoints)**.** *For $h, k : \mathsf{BHist}$, if $\rho_{\mathsf{e}}(h, k)$ holds for the singleton empty-history FPS classifier, then none of h or k is a displayed zero-headed or one-headed endpoint.*

Proof. The classifier exposes singleton carrier witnesses $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. A displayed headed endpoint for either argument rewrites one of these witnesses into $\mathsf{e}_0(z)\mathsf{hsamee}$ or $\mathsf{e}_1(z)\mathsf{hsamee}$, contradicting the constructor separation lemmas for headed histories and the empty history. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier headed endpoint absurd`

Theorem 51.12 (Singleton FPS classifier continuation commutativity closure)**.** *Let $F, G : \mathsf{BHist}$ be carried by the singleton empty-history FPS carrier A_{e}. If $\mathsf{Cont}(F, G, L)$ and $\mathsf{Cont}(G, F, R)$, then both continuation results are carried by A_{e}, and the singleton FPS classifier relates the two results:*

$$A_{\mathsf{e}}(L) \wedge A_{\mathsf{e}}(R) \wedge \rho_{\mathsf{e}}(L, R).$$

Proof. The carried endpoints F and G are both identified with e. Inverting the two continuation witnesses after those identifications reduces the two results to $\mathsf{append}(\mathsf{e}, \mathsf{e})$, hence to e. The result carriers are the singleton empty-history witnesses, and the classifier follows by pairing those witnesses with reflexivity of hsame at e. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonEmptyHistoryClassifier continuation comm closed`

51.3 The certificate

Definition 51.13 (FormalPowerSeries source specification)**.** *The source is the natural index source and scalar ring source.*

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonSourceSpec`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonSourceSpec empty`

Definition 51.14 (FormalPowerSeries pattern specification)**.** *The pattern is infinite coefficient assignment indexed by unary natural histories.*

Definition 51.15 (FormalPowerSeries classifier specification)**.** *The classifier relates two series when all corresponding coefficients are scalar-classified the same.*

Definition 51.16 (FormalPowerSeries stability certificate)**.** *The stability certificate consists of:*

1. *pointwise reflexivity*
2. *pointwise transitivity*
3. *addition closure pointwise*
4. *Cauchy product closure by finite sums at each coefficient index*
5. *no convergence field is required*

Lean4 ✓ `BEDC.Derived.FpsUp.fps stability certificate fields`

Definition 51.17 (FormalPowerSeries ledger policy)**.** *Record the coefficient rule and scalar ledgers at queried indices.*

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff empty ledger`

Definition 51.18 ($\mathsf{FPS}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{FPS}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FpsUp.FpsUp StdBridge`

Primitive boundary. $\mathsf{FPS}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 51.19 (Zero-axiom encoding note)**.** *Avoids funext by using pointwise classifier sameness.*

Definition 51.20 (FPS Cauchy coefficient spine)**.** *Let R be a scalar* $\mathsf{CommRing}^{\uparrow}$ *certificate, read through Definition 44.8 as a* $\mathsf{Ring}^{\uparrow}$ *certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and multiplication $\cdot_R$. For coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$ as in Definition 51.1 and unary index n, let $\mathsf{Split}(n)$ be the finite spine of unary decompositions $i + j = n$ generated by the* $\mathsf{Nat}^{\uparrow}$ *recursion. Define*

$$\mathsf{CauchySp}_n(F, G) := \mathsf{map}_{(i,j)\in\mathsf{Split}(n)}(F_i \cdot_R G_j).$$

The formal product coefficient is the finite additive fold from Definition 50.26:

$$(F \odot_{\mathrm{fps}} G)_n := \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, G)).$$

Lean4 (definition) `BEDC.Derived.FpsUp.FpsCauchyCoefficientSpine`
Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonCauchyCoeffSpine`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyCoeffSpine carrier`

Theorem 51.21 (FPS Cauchy product classifier congruence)**.** *Let R be a scalar* $\mathsf{CommRing}^{\uparrow}$ *certificate. In the* $\mathsf{FPS}^{\uparrow}(R)$ *interface, let $\sim_{\mathrm{fps}}$ be the pointwise classifier of Definition 51.15. For carried coefficient sequences $F, F', G, G' : \mathsf{Nat}^{\uparrow} \to R$, if*

$$\forall n : \mathsf{Nat}^{\uparrow},\ F_n \sim_R F'_n \qquad \textit{and} \qquad \forall n : \mathsf{Nat}^{\uparrow},\ G_n \sim_R G'_n,$$

then

$$F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'.$$

Equivalently, for every unary coefficient index n,

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F,G)) \sim_R \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F',G')).$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct classifier congruence`

Proof. Read the scalar $\mathsf{CommRing}^{\uparrow}$ certificate through Definition 44.8. The retained $\mathsf{Ring}^{\uparrow}$ certificate has the same scalar carrier, classifier, zero endpoint, addition, and multiplication. Its stability rows include zero self-classification, additive congruence, and multiplication congruence by Definition 43.26.

Fix a unary index n. By Definition 51.20, both product coefficients use the same finite split spine $\mathsf{Split}(n)$. For each displayed split (i,j) with $i+j=n$, the two pointwise FPS hypotheses give $F_i \sim_R F'_i$ and $G_j \sim_R G'_j$. Multiplication congruence transports these two scalar classifier witnesses to

$$F_i \cdot_R G_j \sim_R F'_i \cdot_R G'_j.$$

Structural induction on the finite split spine packages these comparisons as

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F,G), \mathsf{CauchySp}_n(F',G')).$$

Applying Definition 50.26 to this finite-spine classifier witness gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F,G)) \sim_R \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F',G')).$$

This is exactly the scalar classification of the n-th coefficient of the two Cauchy products by Definition 51.20.

Since n was arbitrary, Definition 51.15 converts the coefficientwise scalar comparison into the FPS classifier judgment

$$F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'.$$

The argument is finite at each coefficient index and uses no convergence field, matching the stability obligations listed in Definition 51.16. □

Definition 51.22 (FPS pointwise addition coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$ and addition $+_R$. For coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$ as in Definition 51.1, the formal sum $F \oplus_{\mathrm{fps}} G$ is the coefficient sequence determined by*

$$(F \oplus_{\mathrm{fps}} G)_n := F_n +_R G_n$$

for every unary index $n : \mathsf{Nat}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff`

Theorem 51.23 (FPS addition commutativity from scalar additive commutativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$ and addition $+_R$. Assume the scalar additive commutativity row retained in Definition 43.26: for all scalar endpoints a, b,*

$$a +_R b \sim_R b +_R a.$$

For a commutative-ring scalar source, Definition 44.8 keeps these additive data unchanged. In the $\mathsf{FPS}^{\uparrow}(R)$ *interface, let* $\sim_{\mathrm{fps}}$ *be the pointwise classifier of Definition 51.15. For carried coefficient sequences* $F, G : \mathsf{Nat}^{\uparrow} \to R$*,*

$$F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} G \oplus_{\mathrm{fps}} F.$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$*,*

$$(F \oplus_{\mathrm{fps}} G)_n \sim_R (G \oplus_{\mathrm{fps}} F)_n.$$

Proof. Fix a unary index n. By Definition 51.22, the two coefficient endpoints unfold as

$$(F \oplus_{\mathrm{fps}} G)_n = F_n +_R G_n$$

and

$$(G \oplus_{\mathrm{fps}} F)_n = G_n +_R F_n.$$

The scalar additive commutativity row gives

$$F_n +_R G_n \sim_R G_n +_R F_n.$$

Substituting the two coefficient rules gives the displayed scalar comparison at index n:

$$(F \oplus_{\mathrm{fps}} G)_n \sim_R (G \oplus_{\mathrm{fps}} F)_n.$$

Since n was arbitrary, the pointwise classifier specification in Definition 51.15 converts these coefficientwise scalar comparisons into

$$F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} G \oplus_{\mathrm{fps}} F.$$

No convergence or finite-support premise is used; the argument is exactly the pointwise addition closure situation recorded in Definition 51.16. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff comm classifier`

Lemma 51.24 (Finite additive fold reversal under additive commutativity)**.** *Let* R *be a scalar* $\mathsf{Ring}^{\uparrow}(R)$ *carrier with classifier* $\sim_R$*, zero endpoint* 0_R*, addition* $+_R$*, and finite coefficient fold* $\mathsf{fold}_{+,R}$ *as in Definition 50.26. Assume the scalar classifier equivalence fields and the additive abelian-group, additive congruence, and zero rows retained in Definition 43.26. For every finite carried scalar spine* $xs : \mathsf{ListCarrier}(R)$*,*

$$\mathsf{fold}_{+,R}(xs) \sim_R \mathsf{fold}_{+,R}(\mathsf{reverse}(xs)).$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonAddFold reverse empty append hsame`

Proof. First prove by induction on xs that for every carried scalar a,

$$\mathsf{fold}_{+,R}(xs{+}{+}[a]) \sim_R \mathsf{fold}_{+,R}(xs) +_R a.$$

For $xs = \mathsf{nil}$, the two endpoints are $a +_R 0_R$ and $0_R +_R a$. The additive zero laws classify both endpoints with a, and classifier symmetry and transitivity compose the required comparison.

For the cons step write $xs = \mathsf{cons}(x, xs')$. The induction hypothesis gives

$$\mathsf{fold}_{+,R}(xs'{+}{+}[a]) \sim_R \mathsf{fold}_{+,R}(xs') +_R a.$$

Additive congruence transports this under the common left summand x, giving

$$x +_R \mathsf{fold}_{+,R}(xs' {+}{+} [a]) \sim_R x +_R (\mathsf{fold}_{+,R}(xs') +_R a).$$

Additive associativity classifies the right endpoint with $(x +_R \mathsf{fold}_{+,R}(xs')) +_R a$, which is $\mathsf{fold}_{+,R}(\mathsf{cons}(x, xs')) +_R a$ by the cons fold clause. This proves the append-last calculation.

Now induct on xs for the reversal claim. The nil case reduces both folds to 0_R, classified with itself by the zero row. In the cons case write $xs = \mathsf{cons}(a, t)$. The cons fold clause gives $a +_R \mathsf{fold}_{+,R}(t)$. The induction hypothesis and additive congruence classify this endpoint with $a +_R \mathsf{fold}_{+,R}(\mathsf{reverse}(t))$. Additive commutativity then classifies it with $\mathsf{fold}_{+,R}(\mathsf{reverse}(t)) +_R a$. The append-last calculation applied to $\mathsf{reverse}(t)$ gives

$$\mathsf{fold}_{+,R}(\mathsf{reverse}(t) {+}{+} [a]) \sim_R \mathsf{fold}_{+,R}(\mathsf{reverse}(t)) +_R a.$$

Classifier symmetry and transitivity identify the preceding endpoint with $\mathsf{fold}_{+,R}(\mathsf{reverse}(t) {+}{+} [a])$, which is $\mathsf{fold}_{+,R}(\mathsf{reverse}(\mathsf{cons}(a, t)))$ by the finite-list reverse recursion. □

Lemma 51.25 (FPS Cauchy coefficient spine swapped reverse)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read through Definition 44.8 as a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$ and multiplication $\cdot_R$. For carried coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, the Cauchy coefficient spines satisfy*

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F, G), \mathsf{reverse}(\mathsf{CauchySp}_n(G, F))).$$

Proof. Use the finite split spine $\mathsf{Split}(n)$ of Definition 51.20. Its unary recursion lists each decomposition $i + j = n$ exactly once, and the same recursion gives the swapped-reverse spine: after applying $(i, j) \mapsto (j, i)$, the resulting finite spine is $\mathsf{reverse}(\mathsf{Split}(n))$.

Induct on this split recursion. At the base index the only decomposition is $(0, 0)$, and multiplicative commutativity from Definition 44.33 gives

$$F_0 \cdot_R G_0 \sim_R G_0 \cdot_R F_0.$$

For the successor step, the exposed left split $(0, \mathsf{succ}(n))$ is paired with the exposed right split $(\mathsf{succ}(n), 0)$ after swapping and reversing; the middle splits are the successor transports of the predecessor split and are handled by the induction hypothesis. For every matched split (i, j), the coefficient carrier supplies carried endpoints F_i and G_j, and scalar multiplicative commutativity gives

$$F_i \cdot_R G_j \sim_R G_j \cdot_R F_i.$$

The nil and cons clauses of Definition 36.12 assemble these scalar comparisons into the displayed finite-spine classifier witness. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct swapped reverse fold hsame`

Theorem 51.26 (FPS Cauchy product commutativity from CommRing scalars)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate. In the $\mathsf{FPS}^{\uparrow}(R)$ interface, let $\sim_{\mathrm{fps}}$ be the pointwise classifier of Definition 51.15. For carried coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$,*

$$\forall n : \mathsf{Nat}^{\uparrow}, \quad (F \odot_{\mathrm{fps}} G)_n \sim_R (G \odot_{\mathrm{fps}} F)_n.$$

Consequently,

$$F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} G \odot_{\mathrm{fps}} F.$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct comm classifier`

Proof. Read the scalar certificate as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. This keeps the scalar classifier, zero endpoint, addition, multiplication, additive abelian-group rows, addition congruence, multiplication congruence, and the finite additive fold used in Definition 51.20. The $\mathsf{CommRing}^{\uparrow}$ layer supplies multiplicative commutativity by Definition 44.33.

Fix $n : \mathsf{Nat}^{\uparrow}$. By Definition 51.25,

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F, G), \mathsf{reverse}(\mathsf{CauchySp}_n(G, F)))$$

holds. Applying Definition 50.26 gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, G)) \sim_R \mathsf{fold}_{+,R}(\mathsf{reverse}(\mathsf{CauchySp}_n(G, F))).$$

By Definition 51.24, applied to $\mathsf{CauchySp}_n(G, F)$, and classifier symmetry,

$$\mathsf{fold}_{+,R}(\mathsf{reverse}(\mathsf{CauchySp}_n(G, F))) \sim_R \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(G, F)).$$

Classifier transitivity composes the two comparisons. Unfolding Definition 51.20 identifies the composed comparison with

$$(F \odot_{\mathrm{fps}} G)_n \sim_R (G \odot_{\mathrm{fps}} F)_n.$$

Since n was arbitrary, the pointwise classifier specification in Definition 51.15 gives

$$F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} G \odot_{\mathrm{fps}} F.$$

□

Theorem 51.27 (FPS Cauchy product distributes over FPS addition)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, pointwise addition from Definition 51.22, and the Cauchy coefficient spine of Definition 51.20 read over these $\mathsf{Ring}^{\uparrow}$ operations. For carried coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$, the displayed products and sums are carried and*

$$F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H) \sim_{\mathrm{fps}} (F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H)$$

and

$$(F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H).$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$,

$$(F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H))_n \sim_R ((F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H))_n$$

and

$$((F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R ((F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n.$$

Proof. The carrying clauses follow from the addition and Cauchy-product closure rows in Definition 51.16. By the pointwise classifier of Definition 51.15, it remains to compare the scalar coefficient endpoints at an arbitrary index.

Fix $n : \mathsf{Nat}^{\uparrow}$. For the first comparison, set

$$x_n := \mathsf{CauchySp}_n(F, G), \qquad y_n := \mathsf{CauchySp}_n(F, H).$$

Both finite spines are indexed by the same split spine $\mathsf{Split}(n)$ from Definition 51.20. For each split (i, j) in that spine, pointwise addition gives $(G \oplus_{\mathrm{fps}} H)_j = G_j +_R H_j$. Multiplication congruence and scalar left distributivity from Definition 43.26 give

$$F_i \cdot_R (G \oplus_{\mathrm{fps}} H)_j \sim_R (F_i \cdot_R G_j) +_R (F_i \cdot_R H_j).$$

The matching split order assembles these scalar witnesses, using Definition 36.12, into

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F, G \oplus_{\mathrm{fps}} H), x_n +_R y_n).$$

Applying Definition 50.26 yields

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, G \oplus_{\mathrm{fps}} H)) \sim_R \mathsf{fold}_{+,R}(x_n +_R y_n).$$

By Definition 49.98,

$$\mathsf{fold}_{+,R}(x_n +_R y_n) \sim_R \mathsf{fold}_{+,R}(x_n) +_R \mathsf{fold}_{+,R}(y_n).$$

Classifier transitivity composes these two comparisons. Unfolding the Cauchy coefficient rule in Definition 51.20 and the pointwise addition rule in Definition 51.22 identifies the composed endpoint comparison with

$$(F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H))_n \sim_R ((F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H))_n.$$

For the second comparison, set

$$u_n := \mathsf{CauchySp}_n(F, H), \qquad v_n := \mathsf{CauchySp}_n(G, H).$$

Again both spines use the same split spine $\mathsf{Split}(n)$. For each split (i, j), pointwise addition gives $(F \oplus_{\mathrm{fps}} G)_i = F_i +_R G_i$. Multiplication congruence and scalar right distributivity from Definition 43.26 give

$$(F \oplus_{\mathrm{fps}} G)_i \cdot_R H_j \sim_R (F_i \cdot_R H_j) +_R (G_i \cdot_R H_j).$$

Thus

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F \oplus_{\mathrm{fps}} G, H), u_n +_R v_n)$$

holds. The fold congruence theorem Definition 50.26 and the fold-splitting lemma Definition 49.98 classify

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F \oplus_{\mathrm{fps}} G, H)) \sim_R \mathsf{fold}_{+,R}(u_n) +_R \mathsf{fold}_{+,R}(v_n).$$

Unfolding Definition 51.20 and Definition 51.22 identifies this scalar comparison with

$$((F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R ((F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n.$$

Since n was arbitrary in both calculations, Definition 51.15 packages the coefficientwise scalar comparisons as the two displayed $\mathsf{FPS}^{\uparrow}$ classifier judgments. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct distributes over addition classifier`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct left distrib classifier`

Definition 51.28 (FPS threefold Cauchy split spines)**.** *Let R be read as a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, addition $+_R$, multiplication $\cdot_R$, zero endpoint 0_R, and finite additive fold $\mathsf{fold}_{+,R}$. For coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, define the left and right threefold split spines by*

$$\mathsf{TripSp}^{\ell}_n(F, G, H) := \mathsf{concat}_{(a,k) \in \mathsf{Split}(n)} \mathsf{map}_{(i,j) \in \mathsf{Split}(a)} ((F_i \cdot_R G_j) \cdot_R H_k)$$

and

$$\mathsf{TripSp}^r_n(F,G,H) := \mathsf{concat}_{(i,c)\in\mathsf{Split}(n)}\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}\big(F_i \cdot_R (G_j \cdot_R H_k)\big).$$

The left spine enumerates witnesses (i,j,k) through $(i+j)+k=n$; the right spine enumerates the same triples through $i+(j+k)=n$.

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines`

Lemma 51.29 (FPS threefold Cauchy split reassociation)**.** *Let R carry a $\mathsf{CommRing}^\uparrow(R)$ certificate, read through Definition 44.8 as a $\mathsf{Ring}^\uparrow(R)$ certificate. For carried coefficient sequences $F,G,H:\mathsf{Nat}^\uparrow\to R$ and unary index n, the two threefold split folds of Definition 51.72 are scalar-classified:*

$$\mathsf{fold}_{+,R}(\mathsf{TripSp}^\ell_n(F,G,H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^r_n(F,G,H)).$$

Proof. The split spine $\mathsf{Split}(n)$ of Definition 51.20 is finite. Hence each displayed spine in Definition 51.72 is a finite concatenation of finite split spines. Its entries are carried scalars because the coefficient sequences are carried and the multiplication closure row is retained by Definition 43.26.

The left indexing data are quadruples $((a,k),(i,j))$ with $a+k=n$ and $i+j=a$. Sending this data to (i,j,k) gives a triple with $(i+j)+k=n$. The right indexing data are quadruples $((i,c),(j,k))$ with $i+c=n$ and $j+k=c$. Sending this data to (i,j,k) gives a triple with $i+(j+k)=n$. Unary additive associativity for the $\mathsf{Nat}^\uparrow$ split recursion identifies these two triple predicates, and the two reconstructions

$$(i,j,k)\mapsto((i+j,k),(i,j)),\qquad (i,j,k)\mapsto((i,j+k),(j,k))$$

are inverse on the displayed split witnesses.

Apply the split-spine instance of the two-index finite fold swap certificate of Definition 49.94 to this finite reindexing. It reduces the comparison of the two iterated folds to the pointwise scalar comparison attached to a fixed triple (i,j,k). For that triple, the retained multiplicative associativity row in Definition 43.26 gives

$$(F_i \cdot_R G_j)\cdot_R H_k \sim_R F_i \cdot_R (G_j \cdot_R H_k).$$

The fold-swap certificate and Definition 50.26 lift these pointwise comparisons through the finite spines. This is the displayed scalar classification. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines classifier`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines reassociation`

Theorem 51.30 (FPS Cauchy product associativity from CommRing scalars)**.** *Let R carry a $\mathsf{CommRing}^\uparrow(R)$ certificate. In the $\mathsf{FPS}^\uparrow(R)$ interface, let $\sim_{\mathrm{fps}}$ be the pointwise classifier of Definition 51.15. For carried coefficient sequences $F,G,H:\mathsf{Nat}^\uparrow\to R$,*

$$\forall n:\mathsf{Nat}^\uparrow,\quad \big((F\odot_{\mathrm{fps}}G)\odot_{\mathrm{fps}}H\big)_n \sim_R \big(F\odot_{\mathrm{fps}}(G\odot_{\mathrm{fps}}H)\big)_n.$$

Consequently,

$$(F\odot_{\mathrm{fps}}G)\odot_{\mathrm{fps}}H \sim_{\mathrm{fps}} F\odot_{\mathrm{fps}}(G\odot_{\mathrm{fps}}H).$$

Proof. Read the scalar certificate as a $\mathsf{Ring}^\uparrow(R)$ certificate by Definition 44.8. This keeps the scalar classifier, zero endpoint, addition, multiplication, additive abelian-group rows, multiplication congruence, the distributive rows, and multiplicative associativity from Definition 43.26. The finite additive fold is the fold used in Definition 51.20.

Fix $n : \mathsf{Nat}^{\uparrow}$. By Definition 51.20, the left-associated coefficient is

$$\mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_n(F \odot_{\mathrm{fps}} G, H)\big).$$

For a split $(a, k) \in \mathsf{Split}(n)$, the corresponding summand is

$$(F \odot_{\mathrm{fps}} G)_a \cdot_R H_k.$$

Unfolding the inner product at a gives

$$\mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_a(F, G)\big) \cdot_R H_k.$$

The right-multiplication half of Definition 49.95 classifies this endpoint with the inner finite fold over

$$\mathsf{map}_{(i,j)\in\mathsf{Split}(a)}\big((F_i \cdot_R G_j) \cdot_R H_k\big).$$

Applying Definition 50.26 over the outer split spine lifts these inner comparisons and gives

$$\big((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H\big)_n \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{\ell}_n(F, G, H)\big).$$

The right-associated coefficient expands similarly:

$$\big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n = \mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_n(F, G \odot_{\mathrm{fps}} H)\big).$$

For a split $(i, c) \in \mathsf{Split}(n)$, the summand is

$$F_i \cdot_R (G \odot_{\mathrm{fps}} H)_c.$$

Unfolding the inner product at c and applying the left-multiplication half of Definition 49.95 classifies this summand with the inner fold over

$$\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}\big(F_i \cdot_R (G_j \cdot_R H_k)\big).$$

Finite-fold congruence over the outer split spine gives

$$\big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{r}_n(F, G, H)\big).$$

By Definition 51.73,

$$\mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{\ell}_n(F, G, H)\big) \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{r}_n(F, G, H)\big).$$

Classifier transitivity composes the left expansion, this threefold split comparison, and the symmetry of the right expansion, giving

$$\big((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H\big)_n \sim_R \big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n.$$

Since n was arbitrary, the pointwise classifier specification of Definition 51.15 yields the displayed $\mathsf{FPS}^{\uparrow}$ classifier judgment. The closure of the two products is the Cauchy-product closure field listed in Definition 51.16, and all folds are finite at the fixed coefficient index. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct carried assoc continuation classifier`

Lemma 51.31 (FPS threefold Cauchy split reassociation over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, multiplication closure, multiplication congruence, and multiplicative associativity rows retained in Definition 43.26. Assume also the split-spine fold-swap field for the triangular finite reindexing of Definition 51.72: the left enumeration by $((a,k),(i,j))$ with $a+k=n$ and $i+j=a$ and the right enumeration by $((i,c),(j,k))$ with $i+c=n$ and $j+k=c$ are fold-swappable after transport to the common triple spine (i,j,k). For carried coefficient sequences $F,G,H : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, the two threefold split folds are scalar-classified: $\mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_n(F,G,H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{r}_n(F,G,H))$.*

Proof. The split spine $\mathsf{Split}(n)$ from Definition 51.20 is finite. Therefore both threefold spines of Definition 51.72 are finite concatenations of finite coefficient spines. Each displayed summand is carried by the coefficient-carriedness of F, G, and H, followed twice by the multiplication closure row of Definition 43.26. The left spine indexes a summand by data $((a,k),(i,j))$ with $a+k=n$ and $i+j=a$. Sending this data to (i,j,k) gives the triple equation $(i+j)+k=n$. The right spine indexes a summand by data $((i,c),(j,k))$ with $i+c=n$ and $j+k=c$. Sending this data to (i,j,k) gives $i+(j+k)=n$. Unary additive associativity in the split recursion identifies these two triple predicates, and the displayed reconstructions $(i,j,k) \mapsto ((i+j,k),(i,j))$ and $(i,j,k) \mapsto ((i,j+k),(j,k))$ are inverse on the corresponding split witnesses. Apply the assumed split-spine instance of the fold-swap certificate from Definition 49.94 to this finite reindexing. The remaining pointwise scalar comparison at a fixed triple (i,j,k) is exactly the multiplicative associativity row from Definition 43.26: $(F_i \cdot_R G_j) \cdot_R H_k \sim_R F_i \cdot_R (G_j \cdot_R H_k)$. The fold-swap field and the finite additive fold congruence theorem Definition 50.26 lift these pointwise comparisons through the finite spines, yielding the displayed scalar classification. □

Theorem 51.32 (FPS Cauchy product associativity over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, additive fold congruence, multiplication congruence, multiplicative associativity, left and right distributivity, and zero multiplication rows retained in Definition 43.26. Assume the scalar product fold-distribution lemma Definition 49.95 and the split-spine fold-swap field used in Definition 51.52. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15. Assume its Cauchy product coefficient at each unary index is the finite-spine rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations: $(F \odot_{\mathrm{fps}} G)_n := \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F,G))$. For carried coefficient sequences $F,G,H : \mathsf{Nat}^{\uparrow} \to R$, the displayed products are carried and $\forall n : \mathsf{Nat}^{\uparrow}, \quad ((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R (F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n$. Consequently, $(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$.*

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct assoc classifier`

Proof. The Cauchy-product closure row in Definition 51.16 carries $F \odot_{\mathrm{fps}} G$ and $G \odot_{\mathrm{fps}} H$, and then carries the two outer products. By the pointwise classifier of Definition 51.15, it remains to compare the scalar coefficients at an arbitrary unary index. Fix $n : \mathsf{Nat}^{\uparrow}$. The left-associated coefficient unfolds to $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F \odot_{\mathrm{fps}} G, H))$. For a split $(a,k) \in \mathsf{Split}(n)$, the corresponding summand is $(F \odot_{\mathrm{fps}} G)_a \cdot_R H_k$. The inner product coefficient at a is $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_a(F,G))$. Applying the right-multiplication half of Definition 49.95 classifies the summand with the inner finite fold over $\mathsf{map}_{(i,j)\in\mathsf{Split}(a)}((F_i \cdot_R G_j) \cdot_R H_k)$. The finite additive fold congruence theorem Definition 50.26 lifts these inner comparisons through the outer split spine, giving $((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_n(F,G,H))$. The right-associated coefficient unfolds to $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, G \odot_{\mathrm{fps}} H))$. For a split $(i,c) \in \mathsf{Split}(n)$, the summand is $F_i \cdot_R (G \odot_{\mathrm{fps}} H)_c$. The inner product coefficient at c

is $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_c(G,H))$. Applying the left-multiplication half of Definition 49.95 classifies this summand with the inner finite fold over $\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}(F_i \cdot_R (G_j \cdot_R H_k))$. Again Definition 50.26 lifts through the outer split spine and gives $(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^r_n(F,G,H))$. By Definition 51.52, $\mathsf{fold}_{+,R}(\mathsf{TripSp}^\ell_n(F,G,H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^r_n(F,G,H))$. Classifier transitivity composes the left expansion, this threefold split comparison, and the symmetry of the right expansion, giving $((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R (F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n$. Since n was arbitrary, Definition 51.15 packages these coefficientwise scalar comparisons as the displayed $\mathsf{FPS}^\uparrow$ classifier judgment. □

Theorem 51.33 (FPS Cauchy product multiplicative semigroup laws)**.** *Let R carry a $\mathsf{Ring}^\uparrow(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Let the $\mathsf{FPS}^\uparrow(R)$ carrier be the coefficientwise carrier of Definition 51.1, let its classifier be the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, and let its Cauchy product use the coefficient rule of Definition 51.20. For carried coefficient sequences, these data form the concrete multiplicative semigroup interface consumed by the algebraic formal-power-series layer:*

1. *If F and G are carried, then $F \odot_{\mathrm{fps}} G$ is carried.*
2. *If $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$, then $F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'$.*
3. *$(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$.*

Proof. The carrier part is exactly the Cauchy-product closure row retained by Definition 51.16. With F and G carried, that row returns a carried product $F \odot_{\mathrm{fps}} G$ in the carrier of Definition 51.1.

For compatibility with the classifier, apply Definition 51.21. Its hypotheses are precisely the carried coefficient sequences together with the two pointwise classifier comparisons supplied by Definition 51.15, and its conclusion is the displayed congruence of Cauchy products.

The associativity law is Definition 51.53, read over the same scalar certificate, pointwise classifier, and coefficient rule. The theorem first carries the two intermediate products by Definition 51.16, then identifies the two associated products under $\sim_{\mathrm{fps}}$. These three clauses are exactly the multiplicative semigroup interface required downstream. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct multiplicative semigroup laws`

Definition 51.34 (FPS Cauchy product unit series)**.** *Let R carry a $\mathsf{Ring}^\uparrow(R)$ certificate with multiplicative unit endpoint 1_R and additive zero endpoint 0_R. The Cauchy-product unit series $1_{\mathrm{fps}} : \mathsf{Nat}^\uparrow \to R$ is the coefficient sequence determined by*

$$(1_{\mathrm{fps}})_0 := 1_R$$

and

$$(1_{\mathrm{fps}})_{\mathsf{succ}(n)} := 0_R$$

for every unary index $n : \mathsf{Nat}^\uparrow$.

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonCauchyProductUnitSeries`
Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonUnitSeriesCoeff`

Theorem 51.35 (FPS Cauchy product unit laws)**.** *Let R carry a $\mathsf{Ring}^\uparrow(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, zero self-classification, additive congruence, additive zero rows, multiplication closure, multiplication congruence, left*

and right multiplicative unit rows, and left and right zero multiplication rows retained in Definition 43.26. Let the $\mathsf{FPS}^{\uparrow}(R)$ *interface use the pointwise classifier* $\sim_{\mathrm{fps}}$ *of Definition 51.15 and the Cauchy coefficient rule of Definition 51.20, read over these* $\mathsf{Ring}^{\uparrow}$ *operations. If* $F : \mathsf{Nat}^{\uparrow} \to R$ *is carried, then* 1_{fps} *is carried, the products* $1_{\mathrm{fps}} \odot_{\mathrm{fps}} F$ *and* $F \odot_{\mathrm{fps}} 1_{\mathrm{fps}}$ *are carried, and*

$$1_{\mathrm{fps}} \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} F \qquad \textit{and} \qquad F \odot_{\mathrm{fps}} 1_{\mathrm{fps}} \sim_{\mathrm{fps}} F.$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct unit laws`

Proof. The unit series is carried coefficientwise. At index 0, the coefficient is 1_R, carried by the multiplicative monoid source inside the $\mathsf{Ring}^{\uparrow}$ certificate. At every successor index, the coefficient is 0_R, carried by the zero endpoint row retained in Definition 43.26. The Cauchy-product closure row in Definition 51.16 then carries both products.

For the left unit law, fix a unary index n. If $n = 0$, Definition 51.58 reduces the product coefficient to the singleton fold on $(1_{\mathrm{fps}})_0 \cdot_R F_0$. The coefficient rule of Definition 51.55 reads this as $1_R \cdot_R F_0$, and the scalar multiplicative left-unit row from Definition 43.26 classifies it with F_0. The singleton finite-fold endpoint is transported by the scalar additive zero row and classifier transitivity.

Now let $n = \mathsf{succ}(m)$. The finite split recursion for $\mathsf{Split}(n)$ has exactly one summand whose left index is 0 and all remaining summands have successor left index. The head summand is $1_R \cdot_R F_n$ and is classified with F_n by the scalar left-unit row. Every remaining summand has the form $(1_{\mathrm{fps}})_{\mathsf{succ}(i)} \cdot_R F_j$, hence $0_R \cdot_R F_j$, and the left zero multiplication row classifies it with 0_R.

The zero-summand part of the tail is folded to 0_R by Definition 50.27. Additive fold congruence transports the pointwise comparisons through the finite split spine, and the additive right-zero row removes the folded zero tail. Thus $(1_{\mathrm{fps}} \odot_{\mathrm{fps}} F)_n \sim_R F_n$.

For the right unit law, the zero index is analogous: Definition 51.58 gives the singleton fold on $F_0 \cdot_R (1_{\mathrm{fps}})_0$, the unit-series coefficient rule reads the right factor as 1_R, and the scalar right-unit row classifies the coefficient with F_0.

At $n = \mathsf{succ}(m)$, the finite split recursion has exactly one summand whose right index is 0 and all remaining summands have successor right index. The head summand is $F_n \cdot_R 1_R$ and is classified with F_n by the scalar right-unit row. Each remaining summand is $F_i \cdot_R 0_R$, classified with 0_R by the right zero multiplication row. The same zero-summand fold theorem, additive fold congruence, and additive zero row collapse the tail.

Since n was arbitrary in both directions, Definition 51.15 packages the coefficientwise scalar comparisons as the two displayed $\mathsf{FPS}^{\uparrow}$ classifier judgments. □

Theorem 51.36 (FPS Cauchy product multiplicative monoid laws)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate with scalar classifier* $\sim_R$*, zero endpoint* 0_R*, multiplicative unit endpoint* 1_R*, addition* $+_R$*, multiplication* $\cdot_R$*, and finite additive fold* $\mathsf{fold}_{+,R}$*. Let the* $\mathsf{FPS}^{\uparrow}(R)$ *carrier be the coefficientwise carrier of Definition 51.1, let its classifier be the pointwise classifier* $\sim_{\mathrm{fps}}$ *of Definition 51.15, let its multiplication be the Cauchy product of Definition 51.20, and let its unit be* 1_{fps} *from Definition 51.55. For carried coefficient sequences, these data form the concrete multiplicative monoid interface consumed by the algebraic formal-power-series layer:*

1. *If* F *and* G *are carried, then* $F \odot_{\mathrm{fps}} G$ *is carried.*
2. *If* $F \sim_{\mathrm{fps}} F'$ *and* $G \sim_{\mathrm{fps}} G'$*, then* $F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'$*.*
3. $(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$*.*
4. 1_{fps} *is carried, and* $1_{\mathrm{fps}} \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$ *and* $F \odot_{\mathrm{fps}} 1_{\mathrm{fps}} \sim_{\mathrm{fps}} F$*.*

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct multiplicative monoid laws`

Proof. The first three clauses are exactly Definition 51.54, read over the displayed carrier, classifier, and Cauchy-product coefficient rule.

The unit carrier and the two unit comparisons are Definition 51.56. That theorem supplies the coefficientwise split at zero and successor indices, reduces the off-axis Cauchy summands by scalar zero multiplication, collapses the folded zero tails, and repackages the remaining coefficient comparisons by the pointwise classifier of Definition 51.15. Together with the semigroup clauses, these fields are the required multiplicative monoid surface. □

Lemma 51.37 (FPS zero-index Cauchy spine)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with multiplication $\cdot_R$. For coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$, the split spine used by Definition 51.20 at the zero index is the singleton split $\mathsf{Split}(0) = \mathsf{cons}((0,0), \mathsf{nil})$. Consequently, $\mathsf{CauchySp}_0(F, G) = \mathsf{cons}(F_0 \cdot_R G_0, \mathsf{nil})$.*

Proof. The split spine is generated by the $\mathsf{Nat}^{\uparrow}$ recursion in Definition 51.20. The zero branch has no predecessor index and contains only the decomposition of 0 as $0 + 0$. Applying the displayed map defining $\mathsf{CauchySp}_n(F, G)$ to this singleton split replaces the split entry by its scalar product $F_0 \cdot_R G_0$. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct zero index spine`

Theorem 51.38 (FPS Cauchy product constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the Cauchy product coefficient rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations. For carried coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$, $(F \odot_{\mathrm{fps}} G)_0 \sim_R F_0 \cdot_R G_0$.*

Proof. Put $x := F_0 \cdot_R G_0$. Since F and G are carried coefficient sequences, the multiplication closure row retained in Definition 43.26 carries x. By Definition 51.58 and the product coefficient rule in Definition 51.20, $(F \odot_{\mathrm{fps}} G)_0 = \mathsf{fold}_{+,R}(\mathsf{cons}(x, \mathsf{nil}))$. The finite fold clauses displayed in Definition 50.26 identify this endpoint with $x +_R 0_R$. The additive right-zero theorem Definition 43.31, applied in the scalar ring to x, gives $x +_R 0_R \sim_R x$. Substituting $x = F_0 \cdot_R G_0$ gives the scalar classifier comparison. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct constant coefficient`

Definition 51.39 (FPS pointwise additive monoid instance)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the scalar stability rows retained in Definition 43.26. Using the coefficientwise carrier of Definition 51.1, the additive carrier for the formal-power-series layer is*

$$\mathsf{FPSCarrier}_R := \{F : \mathsf{Nat}^{\uparrow} \to R \mid F \textit{ is carried coefficientwise}\}.$$

Its classifier is the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15. Its binary operation is the pointwise operation $\oplus_{\mathrm{fps}}$ of Definition 51.22:

$$(F \oplus_{\mathrm{fps}} G)_n := F_n +_R G_n.$$

Its zero endpoint is the zero series 0_{fps} with coefficient rule $(0_{\mathrm{fps}})_n := 0_R$ for every unary index $n : \mathsf{Nat}^{\uparrow}$.

Theorem 51.40 (FPS pointwise additive monoid laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive associativity, and additive zero rows retained in Definition 43.26. For the concrete data of Definition 51.60, the following laws hold for carried coefficient sequences:*

1. *0_{fps} is carried, and $F \oplus_{\mathrm{fps}} G$ is carried whenever F and G are carried.*
2. *If $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$, then $F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \oplus_{\mathrm{fps}} G'$.*
3. *$(F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)$.*
4. *$0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$ and $F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}} \sim_{\mathrm{fps}} F$.*

Thus $\mathsf{FPSCarrier}_R$ with $\sim_{\mathrm{fps}}$, $\oplus_{\mathrm{fps}}$, and 0_{fps} is the concrete additive monoid interface consumed by the formal-power-series algebra layer.

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff zero laws`

Proof. The carrier clause follows coefficientwise from Definition 51.1. The zero series is carried because each coefficient is 0_R, and the scalar zero self-classification row in Definition 43.26 supplies the carried scalar endpoint. If F and G are carried, then each coefficient $F_n +_R G_n$ is carried by scalar addition closure. By Definition 51.22, these coefficients are exactly the coefficients of $F \oplus_{\mathrm{fps}} G$.

For congruence, fix $n : \mathsf{Nat}^{\uparrow}$. The hypotheses $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$ become scalar comparisons $F_n \sim_R F'_n$ and $G_n \sim_R G'_n$ by Definition 51.15. Scalar addition congruence from Definition 43.26 gives

$$F_n +_R G_n \sim_R F'_n +_R G'_n.$$

Using Definition 51.22 on both sides and then Definition 51.15 packages the coefficientwise comparisons as $F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \oplus_{\mathrm{fps}} G'$.

Associativity is the existing pointwise associativity theorem Definition 51.51, read with the same carrier, classifier, operation, and scalar associativity row.

It remains to check the zero laws. For the left zero law, fix $n : \mathsf{Nat}^{\uparrow}$. The definitions give

$$(0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F)_n = 0_R +_R F_n.$$

The scalar additive left-zero row retained in Definition 43.26 classifies this endpoint with F_n. Since n was arbitrary, Definition 51.15 yields $0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$.

For the right zero law, fix $n : \mathsf{Nat}^{\uparrow}$. The definitions give

$$(F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}})_n = F_n +_R 0_R.$$

The scalar right-zero theorem Definition 43.31, applied in the scalar ring to F_n, classifies this endpoint with F_n. Again Definition 51.15 packages the coefficientwise comparisons as $F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}} \sim_{\mathrm{fps}} F$. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditiveMonoid laws`

Definition 51.41 (FPS coefficientwise additive inverse)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and scalar additive inverse operation $-_R$. For a carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$, the coefficientwise FPS inverse $-_{\mathrm{fps}}F$ is the coefficient sequence determined by*

$$(-_{\mathrm{fps}}F)_n := -_R(F_n)$$

for every unary index $n : \mathsf{Nat}^{\uparrow}$*. More generally, a sequence* $N : \mathsf{Nat}^{\uparrow} \to R$ *is coefficientwise inverse data for* F *when* N *is carried and, for every* $n : \mathsf{Nat}^{\uparrow}$*,*

$$N_n \sim_R -_R(F_n).$$

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonCoeffwiseInverse`

Theorem 51.42 (FPS coefficientwise additive inverse zero laws)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate with scalar classifier* $\sim_R$*, zero endpoint* 0_R*, addition* $+_R$*, and scalar additive inverse operation* $-_R$*. Assume the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive inverse closure, and additive left inverse rows retained in Definition 43.26; assume also the scalar right inverse law of Definition 43.27 for this addition and inverse. Let the* $\mathsf{FPS}^{\uparrow}(R)$ *interface use the pointwise classifier* $\sim_{\mathrm{fps}}$ *of Definition 51.15, pointwise addition from Definition 51.22, and zero series* 0_{fps} *from Definition 51.60. If* $F : \mathsf{Nat}^{\uparrow} \to R$ *is carried and* $N : \mathsf{Nat}^{\uparrow} \to R$ *is coefficientwise inverse data for* F *in the sense of Definition 51.62, then* $F \oplus_{\mathrm{fps}} N$ *and* $N \oplus_{\mathrm{fps}} F$ *are carried and*

$$F \oplus_{\mathrm{fps}} N \sim_{\mathrm{fps}} 0_{\mathrm{fps}} \qquad \textit{and} \qquad N \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} 0_{\mathrm{fps}}.$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$*,*

$$(F \oplus_{\mathrm{fps}} N)_n \sim_R (0_{\mathrm{fps}})_n \qquad \textit{and} \qquad (N \oplus_{\mathrm{fps}} F)_n \sim_R (0_{\mathrm{fps}})_n.$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCoeffwiseInverse zero laws`

Proof. The definition of coefficientwise inverse data carries N. The pointwise addition closure row in Definition 51.16 carries $F \oplus_{\mathrm{fps}} N$ and $N \oplus_{\mathrm{fps}} F$, and Definition 51.61 carries the zero series.

Fix $n : \mathsf{Nat}^{\uparrow}$. The inverse-data hypothesis gives

$$N_n \sim_R -_R(F_n).$$

By Definition 51.22,

$$(F \oplus_{\mathrm{fps}} N)_n = F_n +_R N_n.$$

Scalar classifier reflexivity at F_n and scalar addition congruence transport the inverse-data comparison under the common left summand:

$$F_n +_R N_n \sim_R F_n +_R (-_R(F_n)).$$

The scalar right inverse law Definition 43.27 classifies the right endpoint with 0_R. Since Definition 51.60 gives $(0_{\mathrm{fps}})_n := 0_R$, transitivity gives

$$(F \oplus_{\mathrm{fps}} N)_n \sim_R (0_{\mathrm{fps}})_n.$$

For the other order, Definition 51.22 gives

$$(N \oplus_{\mathrm{fps}} F)_n = N_n +_R F_n.$$

Scalar addition congruence transports the same inverse-data comparison under the common right summand:

$$N_n +_R F_n \sim_R (-_R(F_n)) +_R F_n.$$

The scalar additive left inverse row retained in Definition 43.26 classifies this endpoint with 0_R, and the zero-series coefficient rule again gives $(0_{\mathrm{fps}})_n := 0_R$. Thus

$$(N \oplus_{\mathrm{fps}} F)_n \sim_R (0_{\mathrm{fps}})_n.$$

Since n was arbitrary, Definition 51.15 packages the two coefficientwise scalar comparisons as the two displayed $\mathsf{FPS}^{\uparrow}$ classifier judgments. □

Theorem 51.43 (FPS pointwise additive group laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and scalar additive inverse operation $-_R$. Assume the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive associativity, additive zero rows, additive inverse closure, and both scalar additive inverse laws retained in Definition 43.26 and Definition 43.27. Then the data of Definition 51.60, together with the coefficientwise inverse operation $-_{\mathrm{fps}}$ of Definition 51.62, form the concrete additive group interface on $\mathsf{FPSCarrier}_R$: 0_{fps} is carried, $F \oplus_{\mathrm{fps}} G$ is carried whenever F and G are carried, $\oplus_{\mathrm{fps}}$ is pointwise congruent, $\oplus_{\mathrm{fps}}$ is associative under $\sim_{\mathrm{fps}}$, both zero laws hold under $\sim_{\mathrm{fps}}$, $-_{\mathrm{fps}}F$ is carried whenever F is carried, $-_{\mathrm{fps}}F$ is coefficientwise inverse data for F, and both inverse-zero laws*

$$F \oplus_{\mathrm{fps}} (-_{\mathrm{fps}}F) \sim_{\mathrm{fps}} 0_{\mathrm{fps}} \qquad \textit{and} \qquad (-_{\mathrm{fps}}F) \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} 0_{\mathrm{fps}}$$

hold for every carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$.

Proof. The carrier, congruence, associativity, and zero-law fields are exactly the fields supplied by Definition 51.61, read over the carrier, classifier, pointwise addition, and zero series fixed in Definition 51.60.

It remains to fit the inverse operation into the same public algebra package. Let $F : \mathsf{Nat}^{\uparrow} \to R$ be carried. For each coefficient index $n : \mathsf{Nat}^{\uparrow}$, scalar additive inverse closure carries $-_R(F_n)$, so the coefficient rule $(-_{\mathrm{fps}}F)_n := -_R(F_n)$ carries the whole sequence $-_{\mathrm{fps}}F$. The same coefficient rule, together with scalar classifier reflexivity at $-_R(F_n)$, makes $-_{\mathrm{fps}}F$ coefficientwise inverse data for F in the sense of Definition 51.62.

Apply Definition 51.63 with $N = -_{\mathrm{fps}}F$. Its hypotheses are precisely the retained scalar inverse, congruence, closure, and zero rows, plus the right inverse law Definition 43.27. The conclusion gives the two displayed inverse-zero laws under $\sim_{\mathrm{fps}}$.

Thus the ten fields just listed form a single concrete additive group interface for the downstream formal-power-series algebra layer: zero carrier, sum carrier, pointwise congruence, associativity, left zero, right zero, inverse carrier, inverse-data fit, right inverse-zero, and left inverse-zero. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditiveGroup laws`

Theorem 51.44 (FPS zero series absorbs Cauchy product over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, zero self-classification, additive congruence, additive left-zero, multiplication closure, and left and right zero multiplication rows retained in Definition 43.26. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, the zero series $Z := 0_{\mathrm{fps}}$ of Definition 51.60, and the Cauchy coefficient rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations. For every carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$, the two Cauchy products $Z \odot_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} Z$ are carried and*

$$Z \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} Z \qquad \textit{and} \qquad F \odot_{\mathrm{fps}} Z \sim_{\mathrm{fps}} Z.$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$,

$$(Z \odot_{\mathrm{fps}} F)_n \sim_R Z_n \qquad \textit{and} \qquad (F \odot_{\mathrm{fps}} Z)_n \sim_R Z_n.$$

Proof. The zero series is carried coefficientwise because Definition 51.60 gives $Z_n = 0_R$ for every n, and the scalar zero self-classification row supplies the carried scalar endpoint. The Cauchy-product closure row in Definition 51.16 therefore carries $Z \odot_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} Z$ from the carriedness of Z and F.

Fix $n : \mathsf{Nat}^{\uparrow}$, and write ZeroSp_n for the finite spine indexed by $\mathsf{Split}(n)$ whose every displayed entry is 0_R. For the left product, each split $(i, j) \in \mathsf{Split}(n)$ contributes the scalar summand

$$Z_i \cdot_R F_j.$$

The zero-series coefficient rule gives the scalar comparison $Z_i \sim_R 0_R$. Together with scalar classifier reflexivity at F_j, the multiplication-congruence row retained in Definition 43.26 compares $Z_i \cdot_R F_j$ with $0_R \cdot_R F_j$; the left zero multiplication row retained in Definition 43.26 then classifies $0_R \cdot_R F_j$ with 0_R. If the coefficient comparison, multiplication-congruence step, or zero multiplication row is read in the opposite orientation, classifier symmetry supplies the displayed orientation. Structural induction on the finite split spine, using the nil and cons clauses of Definition 36.12, assembles these pointwise scalar comparisons into

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(Z, F), \mathsf{ZeroSp}_n).$$

Applying Definition 50.27 to this all-zero Cauchy summand witness gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(Z, F)) \sim_R 0_R.$$

By the Cauchy coefficient rule in Definition 51.20, the left endpoint is $(Z \odot_{\mathrm{fps}} F)_n$. By the zero-series coefficient rule, $Z_n = 0_R$. Hence

$$(Z \odot_{\mathrm{fps}} F)_n \sim_R Z_n.$$

For the right product, each split $(i, j) \in \mathsf{Split}(n)$ contributes

$$F_i \cdot_R Z_j.$$

The zero-series coefficient rule gives the scalar comparison $Z_j \sim_R 0_R$. Together with scalar classifier reflexivity at F_i, the multiplication-congruence row retained in Definition 43.26 compares $F_i \cdot_R Z_j$ with $F_i \cdot_R 0_R$; the right zero multiplication row classifies $F_i \cdot_R 0_R$ with 0_R, again using classifier symmetry if a row is oriented oppositely. The same finite-spine induction over Definition 36.12 yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F, Z), \mathsf{ZeroSp}_n).$$

A second application of Definition 50.27 gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, Z)) \sim_R 0_R,$$

which is exactly

$$(F \odot_{\mathrm{fps}} Z)_n \sim_R Z_n$$

after unfolding Definition 51.20 and the coefficient rule for Z.

Since n was arbitrary, the pointwise classifier specification in Definition 51.15 packages the two coefficientwise scalar comparisons as

$$Z \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} Z \qquad \text{and} \qquad F \odot_{\mathrm{fps}} Z \sim_{\mathrm{fps}} Z.$$

□

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonAddFold zero series absorption`

Theorem 51.45 (FPS Cauchy product distributes over pointwise addition)**.** *In the concrete singleton formal-power-series carrier, let F, G, and H be singleton coefficient histories. Write $\oplus_{\mathrm{fps}}$ for pointwise addition and $\odot_{\mathrm{fps}}$ for the Cauchy product defined by the singleton coefficient spine. Then both distributed product comparisons are classified by* FpsSingletonClassifier*:*

$$F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H) \sim_{\mathsf{FpsSingletonClassifier}} (F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H)$$

and

$$(F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathsf{FpsSingletonClassifier}} (F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H).$$

Moreover, the two undistributed Cauchy-product endpoints

$$F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H) \qquad \textit{and} \qquad (F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H$$

are each classified with the empty history endpoint.

Proof. The pointwise addition interface in Definition 51.60 supplies the singleton addition endpoints, and the Cauchy-product coefficient rule in Definition 51.20 expresses each product coefficient as the finite additive fold over the split spine.

In the singleton carrier, every displayed coefficient spine collapses to the empty history endpoint. The checked zero-series absorption row above gives the same empty endpoint for Cauchy products whose split spine has only empty scalar contributions, and the continuation endpoint rows identify the folded singleton coefficients with that endpoint.

Applying these endpoint identifications on the left-distributive spine gives a classifier comparison between $F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)$ and $(F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H)$. The same argument on the right-distributive spine gives the comparison between $(F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H$ and $(F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$. The singleton empty-carrier collapse also records the two undistributed product endpoints as classified with BHist.Empty. □

↪ variant `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct distributes over addition classifier`

Theorem 51.46 (FPS RingUp tail algebra package)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence, zero self-classification, additive closure, additive congruence, additive associativity, additive zero, additive inverse closure, additive inverse laws, multiplication closure, multiplication congruence, multiplicative associativity, distributivity, and zero multiplication rows retained in Definition 43.26, together with the scalar right inverse law of Definition 43.27. Then the carried coefficient sequences of $\mathsf{FPS}^{\uparrow}(R)$, with pointwise classifier $\sim_{\mathrm{fps}}$, pointwise addition $\oplus_{\mathrm{fps}}$, zero series 0_{fps}, coefficientwise inverse $-_{\mathrm{fps}}$, and Cauchy product $\odot_{\mathrm{fps}}$, jointly provide the RingUp-scoped tail algebra package consisting of:*

1. *the pointwise additive group laws on $\mathsf{FPSCarrier}_R$;*
2. *Cauchy-product carrier closure, congruence, and associativity on the same carrier and classifier;*
3. *both distributivity laws of $\odot_{\mathrm{fps}}$ over $\oplus_{\mathrm{fps}}$;*
4. *zero-series Cauchy-product absorption on the left and on the right.*

Proof. The carrier and classifier named in every row are the common $\mathsf{FPSCarrier}_R$ of Definition 51.1 and the pointwise classifier specified in Definition 51.15. Thus the statement does not introduce

a second formal-power-series surface; it assembles the already established laws over one shared interface.

For the additive part, apply Definition 51.64. It supplies the carried zero series, pointwise sum carrier closure and congruence, associativity, both zero laws, coefficientwise inverse carrier, inverse-data fit, and both inverse-zero laws.

For the multiplicative spine, apply Definition 51.54. It supplies Cauchy-product carrier closure, congruence with respect to $\sim_{\mathrm{fps}}$, and associativity over the same carried coefficient sequences.

The distributive rows are exactly the two conclusions of Definition 51.27, read with the same pointwise addition and Cauchy product. The zero absorption rows are exactly Definition 51.65. Combining these four cited theorem rows yields the displayed RingUp-scoped tail package for downstream formal-power-series algebra. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonRingupTailAlgebra laws`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonRingUpTailAlgebra package`

Theorem 51.47 (FPS singleton empty-history semantic certificate)**.** *The singleton empty-history formal-power-series carrier* $A_{\mathsf{e}}(h) := h\mathsf{hsamee}$ *on* BHist*, together with the classifier*

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge h\mathsf{hsame}k,$$

forms the semantic naming certificate for the terminal formal-power-series history object. The RingUp tail algebra package of Definition 51.67 is read over this same BHist *carrier/classifier surface when the terminal FPS object is used; it does not introduce a second formal-power-series interface.*

Proof. The singleton carrier and classifier are the encoded BHist data of Definition 51.5. The Lean-checked certificate $\mathsf{FpsSingletonEmptyHistory_semanticNameCert}$ supplies the semantic name-certificate fields for exactly this carrier/classifier pair.

The tail algebra package in Definition 51.67 assembles the additive, multiplicative, distributive, and zero-absorption rows over one shared formal-power-series carrier and pointwise classifier. Reading that package at the terminal singleton object therefore keeps the algebra rows on the same BHist naming surface certified above, rather than passing through a separate FPS naming interface. □

↪variant `BEDC.Derived.FpsUp.FpsSingletonEmptyHistory semanticNameCert`

Theorem 51.48 (FPS RingUp public algebra interface)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}(R)$ *scalar certificate with scalar classifier* $\sim_R$*, zero endpoint* 0_R*, addition* $+_R$*, additive inverse* $-_R$*, multiplication* $\cdot_R$*, unit endpoint* 1_R*, and finite additive fold* $\mathsf{fold}_{+,R}$*. Then the shared* $\mathsf{FPS}^{\uparrow}(R)$ *carrier of coefficient sequences, equipped with the pointwise classifier* $\sim_{\mathrm{fps}}$*, exports one public algebra interface whose operations are pointwise addition* $\oplus_{\mathrm{fps}}$*, the zero series* 0_{fps}*, coefficientwise inverse* $-_{\mathrm{fps}}$*, Cauchy product* $\odot_{\mathrm{fps}}$*, and the Cauchy unit series* 1_{fps}*.*

This public interface consumes exactly the rows of Definition 51.64, Definition 51.57, Definition 51.66, Definition 51.65, and Definition 51.68: additive group laws, multiplicative monoid laws, distributivity, zero-product absorption, and the terminal singleton semantic naming certificate.

Proof. The carrier appearing in every cited algebra row is the coefficientwise $\mathsf{FPS}^{\uparrow}(R)$ carrier, and the classifier is always the pointwise classifier $\sim_{\mathrm{fps}}$. Thus the additive, multiplicative, distributive, zero-absorption, and singleton semantic rows are read over one shared family of carried coefficient sequences.

The additive operations come from Definition 51.64; the multiplicative product and unit operations come from Definition 51.57. The compatibility of the two operation families is supplied by Definition 51.66 and Definition 51.65.

Finally, Definition 51.68 records that the terminal singleton reading stays on the same naming surface. Repacking these rows therefore exports a single public $\mathsf{FPS}^{\uparrow}(R)$ algebra surface, not a second formal-power-series carrier with separately transported operations. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsRingup public algebra interface`

Theorem 51.49 (FPS concrete RingUp instance certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ scalar certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, unit endpoint 1_R, and finite additive fold $\mathsf{fold}_{+,R}$ as in Definition 51.69. Then the concrete formal-power-series surface*

$$(\mathsf{FPSCarrier}_R,\ \sim_{\mathrm{fps}},\ \oplus_{\mathrm{fps}},\ 0_{\mathrm{fps}},\ -_{\mathrm{fps}},\ \odot_{\mathrm{fps}},\ 1_{\mathrm{fps}})$$

forms the corresponding $\mathsf{Ring}^{\uparrow}$ instance surface: the additive part is a pointwise group over $\mathsf{FPSCarrier}_R$, the multiplicative part is a Cauchy-product monoid on the same carrier and classifier, Cauchy product distributes over pointwise addition on both sides, the zero series absorbs Cauchy product on both sides, and the singleton empty-history naming certificate records the semantic carrier used by the terminal concrete instance.

Proof. The carrier, classifier, and displayed operation names are exactly the public interface exported in Definition 51.69; hence the certificate only has to assemble the algebra rows already stated there.

Apply Definition 51.64 for the pointwise additive group rows on $\mathsf{FPSCarrier}_R$. Apply Definition 51.57 for Cauchy-product closure, congruence, associativity, and the unit series on the same pointwise classifier.

The interaction rows are supplied by Definition 51.66 and Definition 51.65. They identify the two distributivity laws and the two zero-product absorption laws without changing the carrier or classifier.

Finally, Definition 51.68 records the singleton semantic naming surface used when the concrete instance is read at the terminal empty-history object. Together these rows are the concrete $\mathsf{Ring}^{\uparrow}$ instance certificate for the displayed FPS carrier and operations. □

Theorem 51.50 (FPS RingUp standard bridge boundary)**.** *The public algebra interface of Definition 51.69 and the concrete instance certificate of Definition 51.70 form the paper bridge from the BEDC coefficientwise FPS carrier to the standard formal-power-series ring schema. The bridge boundary sends the carrier to coefficientwise carried sequences over $\mathsf{Nat}^{\uparrow}$, sends the classifier to the pointwise scalar classifier, sends addition and inverse to their pointwise operations, sends multiplication to the Cauchy product through finite split spines, and sends zero and unit to the displayed scalar rows. It exports no analytic convergence, topology, completion, or external library quotient.*

Proof. First read the public surface from Definition 51.69. It fixes one carrier $\mathsf{FPSCarrier}_R$, one pointwise classifier $\sim_{\mathrm{fps}}$, and the five operation rows $\oplus_{\mathrm{fps}}$, 0_{fps}, $-_{\mathrm{fps}}$, $\odot_{\mathrm{fps}}$, and 1_{fps}.

Next apply Definition 51.70. It repacks that same surface as a concrete $\mathsf{Ring}^{\uparrow}$ instance, with the additive group rows, Cauchy-product monoid rows, distributivity rows, zero-product absorption rows, and singleton semantic certificate all kept on the displayed carrier and classifier.

The Cauchy-product part is the only non-pointwise operation. Its coefficient is defined by the finite split spine of Definition 51.20, and the finite-fold transport rows already used in Definition 50.26 carry the scalar comparisons through those spines.

Thus the bridge boundary is exactly the public coefficientwise $\mathsf{Ring}^{\uparrow}$ schema: carrier and classifier are pointwise, additive structure is pointwise, multiplication is the displayed finite Cauchy fold, and the excluded analytic or quotient data are not part of the exported interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$**.**

Constructive story: A formal power series over a certified ring is built as a coefficient sequence indexed by $\mathsf{Nat}^{\uparrow}$, with every coefficient carried by the same $\mathsf{Ring}^{\uparrow}$ scalar certificate. The carrier accepts exactly coefficientwise carried sequences, the classifier compares two series pointwise by the scalar classifier at each unary index, and sequences with an uncarried coefficient or a missing pointwise comparison stay outside the public algebra surface. The source is the scalar $\mathsf{Ring}^{\uparrow}$ data together with the coefficientwise carrier, the pattern records pointwise addition, zero series, coefficientwise inverse, Cauchy product through finite split spines, the unit series, distributivity, and zero-product absorption, while the ledger keeps each finite fold, split witness, and scalar coefficient comparison visible. The name-certificate obligations are anchored by the singleton empty-history semantic certificate for the terminal instance and by the scalar rows for the general interface: carried zero or unit sequences inhabit the relevant carrier, pointwise classifier equivalence gives reflexivity, symmetry, and transitivity, and coefficientwise transport preserves carried series under classified replacement. A sequence history h lies in the signature gap of a $\mathsf{FPS}^{\uparrow}$ token p when the source-domain accepts all coefficients of h, the signature relation pairs it with an answer series and evidence ledger of coefficient witnesses and finite folds, and token introduction names that answer as p; unfolded, every unary coefficient matches the scalar endpoint encoded by p and every product coefficient is the displayed Cauchy finite-fold value. The bridge boundary is the public coefficientwise $\mathsf{Ring}^{\uparrow}$ interface, with no analytic convergence, completion, or separate formal-series carrier introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public FPS RingUp bridge boundary, coefficientwise carrier and classifier, finite Cauchy-fold product, zero absorption rows, and consumer-completeness packaging are closed over $\mathsf{Ring}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, the pointwise operations, Cauchy unit series, distributivity rows, singleton semantic certificate, and displayed finite-fold witnesses in this child file.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FpsUp.FpsRingup\_public\_algebra\_interface`

Bridge status: bridgeChecked

Not claimed: This chapter does not close analytic convergence, topology, completion, or bridges to standard formal-power-series libraries.

Upgrade path: no further upgrade – chapter at top of formal axis

Theorem 51.51 (FPS pointwise addition associativity)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$ and addition $+_R$. Assume the scalar additive associativity row retained in Definition 43.26: for all carried scalar endpoints a, b, c,*

$$(a +_R b) +_R c \sim_R a +_R (b +_R c).$$

In the $\mathsf{FPS}^{\uparrow}(R)$ interface, let $\sim_{\mathrm{fps}}$ be the pointwise classifier of Definition 51.15, and let pointwise addition be the operation of Definition 51.22. For carried coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$, the two iterated pointwise sums are carried by the $\mathsf{FPS}^{\uparrow}(R)$ carrier and

$$(F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H).$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$,

$$\big((F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H\big)_n \sim_R \big(F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)\big)_n.$$

Proof. The pointwise addition closure row in Definition 51.16 applies twice. Applied first to F, G and then to $F \oplus_{\mathrm{fps}} G, H$, it carries the left-associated sum. Applied first to G, H and then to $F, G \oplus_{\mathrm{fps}} H$, it carries the right-associated sum.

Fix $n : \mathsf{Nat}^{\uparrow}$. By Definition 51.22, the left-associated coefficient unfolds in two steps as

$$\big((F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H\big)_n = (F_n +_R G_n) +_R H_n.$$

The same rule unfolds the right-associated coefficient as

$$\big(F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)\big)_n = F_n +_R (G_n +_R H_n).$$

The carried-coefficient hypotheses ensure that F_n, G_n, and H_n are carried scalar endpoints. The scalar additive associativity row from Definition 43.26 therefore gives

$$(F_n +_R G_n) +_R H_n \sim_R F_n +_R (G_n +_R H_n).$$

Substituting the two coefficient unfoldings gives the displayed scalar classifier comparison at index n.

Since n was arbitrary, the pointwise classifier specification in Definition 51.15 packages these coefficientwise scalar comparisons as

$$(F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H).$$

No convergence or finite-support premise is used. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff associativity`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff assoc classifier`

Lemma 51.52 (FPS threefold Cauchy split reassociation over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, multiplication closure, multiplication congruence, and multiplicative associativity rows retained in Definition 43.26. Assume also the split-spine fold-swap field for the triangular finite reindexing of Definition 51.72: the left enumeration by $((a,k),(i,j))$ with $a+k=n$ and $i+j=a$ and the right enumeration by $((i,c),(j,k))$ with $i+c=n$ and $j+k=c$ are fold-swappable after transport to the common triple spine (i,j,k). For carried coefficient sequences $F,G,H : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, the two threefold split folds are scalar-classified: $\mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_n(F,G,H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{r}_n(F,G,H))$.*

Proof. The split spine $\mathsf{Split}(n)$ from Definition 51.20 is finite. Therefore both threefold spines of Definition 51.72 are finite concatenations of finite coefficient spines. Each displayed summand is carried by the coefficient-carriedness of F, G, and H, followed twice by the multiplication closure row of Definition 43.26. The left spine indexes a summand by data $((a,k),(i,j))$ with $a+k=n$ and $i+j=a$. Sending this data to (i,j,k) gives the triple equation $(i+j)+k=n$. The right spine indexes a summand by data $((i,c),(j,k))$ with $i+c=n$ and $j+k=c$. Sending this data to (i,j,k) gives $i+(j+k)=n$. Unary additive associativity in the split recursion identifies these two triple predicates, and the displayed reconstructions $(i,j,k) \mapsto ((i+j,k),(i,j))$ and $(i,j,k) \mapsto ((i,j+k),(j,k))$ are inverse on the corresponding split witnesses. Apply the assumed split-spine instance of the fold-swap certificate from Definition 49.94 to this finite reindexing. The remaining pointwise scalar comparison at a fixed triple (i,j,k) is exactly the multiplicative associativity row from Definition 43.26: $(F_i \cdot_R G_j) \cdot_R H_k \sim_R F_i \cdot_R (G_j \cdot_R H_k)$. The fold-swap field and the finite additive fold congruence theorem Definition 50.26 lift these pointwise comparisons through the finite spines, yielding the displayed scalar classification. □

Theorem 51.53 (FPS Cauchy product associativity over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, additive fold congruence, multiplication congruence, multiplicative associativity, left and right distributivity, and zero multiplication rows retained in Definition 43.26. Assume the scalar product fold-distribution lemma*

Definition 49.95 and the split-spine fold-swap field used in Definition 51.52. Let the $\mathsf{FPS}^{\uparrow}(R)$ *interface use the pointwise classifier* $\sim_{\mathrm{fps}}$ *of Definition 51.15. Assume its Cauchy product coefficient at each unary index is the finite-spine rule of Definition 51.20, read over these* $\mathsf{Ring}^{\uparrow}$ *operations:* $(F \odot_{\mathrm{fps}} G)_n := \mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F,G))$. *For carried coefficient sequences* $F, G, H : \mathsf{Nat}^{\uparrow} \to R$, *the displayed products are carried and* $\forall n : \mathsf{Nat}^{\uparrow}, \quad ((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R (F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n$. *Consequently,* $(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$.

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct assoc classifier`

Proof. The Cauchy-product closure row in Definition 51.16 carries $F \odot_{\mathrm{fps}} G$ and $G \odot_{\mathrm{fps}} H$, and then carries the two outer products. By the pointwise classifier of Definition 51.15, it remains to compare the scalar coefficients at an arbitrary unary index. Fix $n : \mathsf{Nat}^{\uparrow}$. The left-associated coefficient unfolds to $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F \odot_{\mathrm{fps}} G, H))$. For a split $(a,k) \in \mathsf{Split}(n)$, the corresponding summand is $(F \odot_{\mathrm{fps}} G)_a \cdot_R H_k$. The inner product coefficient at a is $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_a(F,G))$. Applying the right-multiplication half of Definition 49.95 classifies the summand with the inner finite fold over $\mathsf{map}_{(i,j)\in\mathsf{Split}(a)}((F_i \cdot_R G_j) \cdot_R H_k)$. The finite additive fold congruence theorem Definition 50.26 lifts these inner comparisons through the outer split spine, giving $((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_n(F,G,H))$. The right-associated coefficient unfolds to $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, G \odot_{\mathrm{fps}} H))$. For a split $(i,c) \in \mathsf{Split}(n)$, the summand is $F_i \cdot_R (G \odot_{\mathrm{fps}} H)_c$. The inner product coefficient at c is $\mathsf{fold}_{+,R}(\mathsf{CauchySp}_c(G,H))$. Applying the left-multiplication half of Definition 49.95 classifies this summand with the inner finite fold over $\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}(F_i \cdot_R (G_j \cdot_R H_k))$. Again Definition 50.26 lifts through the outer split spine and gives $(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{r}_n(F,G,H))$. By Definition 51.52, $\mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_n(F,G,H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{r}_n(F,G,H))$. Classifier transitivity composes the left expansion, this threefold split comparison, and the symmetry of the right expansion, giving $((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H)_n \sim_R (F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H))_n$. Since n was arbitrary, Definition 51.15 packages these coefficientwise scalar comparisons as the displayed $\mathsf{FPS}^{\uparrow}$ classifier judgment. □

Theorem 51.54 (FPS Cauchy product multiplicative semigroup laws)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate with scalar classifier* $\sim_R$, *zero endpoint* 0_R, *addition* $+_R$, *multiplication* $\cdot_R$, *and finite additive fold* $\mathsf{fold}_{+,R}$. *Let the* $\mathsf{FPS}^{\uparrow}(R)$ *carrier be the coefficientwise carrier of Definition 51.1, let its classifier be the pointwise classifier* $\sim_{\mathrm{fps}}$ *of Definition 51.15, and let its Cauchy product use the coefficient rule of Definition 51.20. For carried coefficient sequences, these data form the concrete multiplicative semigroup interface consumed by the algebraic formal-power-series layer:*

1. *If* F *and* G *are carried, then* $F \odot_{\mathrm{fps}} G$ *is carried.*
2. *If* $F \sim_{\mathrm{fps}} F'$ *and* $G \sim_{\mathrm{fps}} G'$, *then* $F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'$.
3. $(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$.

Proof. The carrier part is exactly the Cauchy-product closure row retained by Definition 51.16. With F and G carried, that row returns a carried product $F \odot_{\mathrm{fps}} G$ in the carrier of Definition 51.1.

For compatibility with the classifier, apply Definition 51.21. Its hypotheses are precisely the carried coefficient sequences together with the two pointwise classifier comparisons supplied by Definition 51.15, and its conclusion is the displayed congruence of Cauchy products.

The associativity law is Definition 51.53, read over the same scalar certificate, pointwise classifier, and coefficient rule. The theorem first carries the two intermediate products by Definition 51.16, then identifies the two associated products under $\sim_{\mathrm{fps}}$. These three clauses are exactly the multiplicative semigroup interface required downstream. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct multiplicative semigroup laws`

Definition 51.55 (FPS Cauchy product unit series)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with multiplicative unit endpoint 1_R and additive zero endpoint 0_R. The Cauchy-product unit series $1_{\mathrm{fps}} : \mathsf{Nat}^{\uparrow} \to R$ is the coefficient sequence determined by*

$$(1_{\mathrm{fps}})_0 := 1_R$$

and

$$(1_{\mathrm{fps}})_{\mathsf{succ}(n)} := 0_R$$

for every unary index $n : \mathsf{Nat}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonCauchyProductUnitSeries`
Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonUnitSeriesCoeff`

Theorem 51.56 (FPS Cauchy product unit laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, zero self-classification, additive congruence, additive zero rows, multiplication closure, multiplication congruence, left and right multiplicative unit rows, and left and right zero multiplication rows retained in Definition 43.26. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15 and the Cauchy coefficient rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations. If $F : \mathsf{Nat}^{\uparrow} \to R$ is carried, then 1_{fps} is carried, the products $1_{\mathrm{fps}} \odot_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} 1_{\mathrm{fps}}$ are carried, and*

$$1_{\mathrm{fps}} \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} F \qquad \textit{and} \qquad F \odot_{\mathrm{fps}} 1_{\mathrm{fps}} \sim_{\mathrm{fps}} F.$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct unit laws`

Proof. The unit series is carried coefficientwise. At index 0, the coefficient is 1_R, carried by the multiplicative monoid source inside the $\mathsf{Ring}^{\uparrow}$ certificate. At every successor index, the coefficient is 0_R, carried by the zero endpoint row retained in Definition 43.26. The Cauchy-product closure row in Definition 51.16 then carries both products.

For the left unit law, fix a unary index n. If $n = 0$, Definition 51.58 reduces the product coefficient to the singleton fold on $(1_{\mathrm{fps}})_0 \cdot_R F_0$. The coefficient rule of Definition 51.55 reads this as $1_R \cdot_R F_0$, and the scalar multiplicative left-unit row from Definition 43.26 classifies it with F_0. The singleton finite-fold endpoint is transported by the scalar additive zero row and classifier transitivity.

Now let $n = \mathsf{succ}(m)$. The finite split recursion for $\mathsf{Split}(n)$ has exactly one summand whose left index is 0 and all remaining summands have successor left index. The head summand is $1_R \cdot_R F_n$ and is classified with F_n by the scalar left-unit row. Every remaining summand has the form $(1_{\mathrm{fps}})_{\mathsf{succ}(i)} \cdot_R F_j$, hence $0_R \cdot_R F_j$, and the left zero multiplication row classifies it with 0_R.

The zero-summand part of the tail is folded to 0_R by Definition 50.27. Additive fold congruence transports the pointwise comparisons through the finite split spine, and the additive right-zero row removes the folded zero tail. Thus $(1_{\mathrm{fps}} \odot_{\mathrm{fps}} F)_n \sim_R F_n$.

For the right unit law, the zero index is analogous: Definition 51.58 gives the singleton fold on $F_0 \cdot_R (1_{\mathrm{fps}})_0$, the unit-series coefficient rule reads the right factor as 1_R, and the scalar right-unit row classifies the coefficient with F_0.

At $n = \mathsf{succ}(m)$, the finite split recursion has exactly one summand whose right index is 0 and all remaining summands have successor right index. The head summand is $F_n \cdot_R 1_R$ and is classified with F_n by the scalar right-unit row. Each remaining summand is $F_i \cdot_R 0_R$, classified

with 0_R by the right zero multiplication row. The same zero-summand fold theorem, additive fold congruence, and additive zero row collapse the tail.

Since n was arbitrary in both directions, Definition 51.15 packages the coefficientwise scalar comparisons as the two displayed $\mathsf{FPS}^{\uparrow}$ classifier judgments. □

Theorem 51.57 (FPS Cauchy product multiplicative monoid laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, multiplicative unit endpoint 1_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Let the $\mathsf{FPS}^{\uparrow}(R)$ carrier be the coefficientwise carrier of Definition 51.1, let its classifier be the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, let its multiplication be the Cauchy product of Definition 51.20, and let its unit be 1_{fps} from Definition 51.55. For carried coefficient sequences, these data form the concrete multiplicative monoid interface consumed by the algebraic formal-power-series layer:*

1. *If F and G are carried, then $F \odot_{\mathrm{fps}} G$ is carried.*
2. *If $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$, then $F \odot_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \odot_{\mathrm{fps}} G'$.*
3. *$(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$.*
4. *1_{fps} is carried, and $1_{\mathrm{fps}} \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} 1_{\mathrm{fps}} \sim_{\mathrm{fps}} F$.*

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct multiplicative monoid laws`

Proof. The first three clauses are exactly Definition 51.54, read over the displayed carrier, classifier, and Cauchy-product coefficient rule.

The unit carrier and the two unit comparisons are Definition 51.56. That theorem supplies the coefficientwise split at zero and successor indices, reduces the off-axis Cauchy summands by scalar zero multiplication, collapses the folded zero tails, and repackages the remaining coefficient comparisons by the pointwise classifier of Definition 51.15. Together with the semigroup clauses, these fields are the required multiplicative monoid surface. □

Lemma 51.58 (FPS zero-index Cauchy spine)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with multiplication $\cdot_R$. For coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$, the split spine used by Definition 51.20 at the zero index is the singleton split $\mathsf{Split}(0) = \mathsf{cons}((0,0), \mathsf{nil})$. Consequently, $\mathsf{CauchySp}_0(F, G) = \mathsf{cons}(F_0 \cdot_R G_0, \mathsf{nil})$.*

Proof. The split spine is generated by the $\mathsf{Nat}^{\uparrow}$ recursion in Definition 51.20. The zero branch has no predecessor index and contains only the decomposition of 0 as $0 + 0$. Applying the displayed map defining $\mathsf{CauchySp}_n(F, G)$ to this singleton split replaces the split entry by its scalar product $F_0 \cdot_R G_0$. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct zero index spine`

Theorem 51.59 (FPS Cauchy product constant coefficient)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the Cauchy product coefficient rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations. For carried coefficient sequences $F, G : \mathsf{Nat}^{\uparrow} \to R$, $(F \odot_{\mathrm{fps}} G)_0 \sim_R F_0 \cdot_R G_0$.*

Proof. Put $x := F_0 \cdot_R G_0$. Since F and G are carried coefficient sequences, the multiplication closure row retained in Definition 43.26 carries x. By Definition 51.58 and the product coefficient rule in Definition 51.20, $(F \odot_{\mathrm{fps}} G)_0 = \mathsf{fold}_{+,R}(\mathsf{cons}(x, \mathsf{nil}))$. The finite fold clauses displayed in Definition 50.26 identify this endpoint with $x +_R 0_R$. The additive right-zero theorem Definition 43.31,

applied in the scalar ring to x, gives $x +_R 0_R \sim_R x$. Substituting $x = F_0 \cdot_R G_0$ gives the scalar classifier comparison. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct constant coefficient`

Definition 51.60 (FPS pointwise additive monoid instance)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the scalar stability rows retained in Definition 43.26. Using the coefficientwise carrier of Definition 51.1, the additive carrier for the formal-power-series layer is*

$$\mathsf{FPSCarrier}_R := \{F : \mathsf{Nat}^{\uparrow} \to R \mid F \text{ is carried coefficientwise}\}.$$

Its classifier is the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15. Its binary operation is the pointwise operation $\oplus_{\mathrm{fps}}$ of Definition 51.22:

$$(F \oplus_{\mathrm{fps}} G)_n := F_n +_R G_n.$$

Its zero endpoint is the zero series 0_{fps} with coefficient rule $(0_{\mathrm{fps}})_n := 0_R$ for every unary index $n : \mathsf{Nat}^{\uparrow}$.

Theorem 51.61 (FPS pointwise additive monoid laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive associativity, and additive zero rows retained in Definition 43.26. For the concrete data of Definition 51.60, the following laws hold for carried coefficient sequences:*

1. *0_{fps} is carried, and $F \oplus_{\mathrm{fps}} G$ is carried whenever F and G are carried.*
2. *If $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$, then $F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \oplus_{\mathrm{fps}} G'$.*
3. *$(F \oplus_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \oplus_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)$.*
4. *$0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$ and $F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}} \sim_{\mathrm{fps}} F$.*

Thus $\mathsf{FPSCarrier}_R$ with $\sim_{\mathrm{fps}}$, $\oplus_{\mathrm{fps}}$, and 0_{fps} is the concrete additive monoid interface consumed by the formal-power-series algebra layer.

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditionCoeff zero laws`

Proof. The carrier clause follows coefficientwise from Definition 51.1. The zero series is carried because each coefficient is 0_R, and the scalar zero self-classification row in Definition 43.26 supplies the carried scalar endpoint. If F and G are carried, then each coefficient $F_n +_R G_n$ is carried by scalar addition closure. By Definition 51.22, these coefficients are exactly the coefficients of $F \oplus_{\mathrm{fps}} G$.

For congruence, fix $n : \mathsf{Nat}^{\uparrow}$. The hypotheses $F \sim_{\mathrm{fps}} F'$ and $G \sim_{\mathrm{fps}} G'$ become scalar comparisons $F_n \sim_R F'_n$ and $G_n \sim_R G'_n$ by Definition 51.15. Scalar addition congruence from Definition 43.26 gives

$$F_n +_R G_n \sim_R F'_n +_R G'_n.$$

Using Definition 51.22 on both sides and then Definition 51.15 packages the coefficientwise comparisons as $F \oplus_{\mathrm{fps}} G \sim_{\mathrm{fps}} F' \oplus_{\mathrm{fps}} G'$.

Associativity is the existing pointwise associativity theorem Definition 51.51, read with the same carrier, classifier, operation, and scalar associativity row.

It remains to check the zero laws. For the left zero law, fix $n : \mathsf{Nat}^{\uparrow}$. The definitions give

$$(0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F)_n = 0_R +_R F_n.$$

The scalar additive left-zero row retained in Definition 43.26 classifies this endpoint with F_n. Since n was arbitrary, Definition 51.15 yields $0_{\mathrm{fps}} \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} F$.

For the right zero law, fix $n : \mathsf{Nat}^{\uparrow}$. The definitions give

$$(F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}})_n = F_n +_R 0_R.$$

The scalar right-zero theorem Definition 43.31, applied in the scalar ring to F_n, classifies this endpoint with F_n. Again Definition 51.15 packages the coefficientwise comparisons as $F \oplus_{\mathrm{fps}} 0_{\mathrm{fps}} \sim_{\mathrm{fps}} F$. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditiveMonoid laws`

Definition 51.62 (FPS coefficientwise additive inverse)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and scalar additive inverse operation $-_R$. For a carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$, the coefficientwise FPS inverse $-_{\mathrm{fps}}F$ is the coefficient sequence determined by*

$$(-_{\mathrm{fps}}F)_n := -_R(F_n)$$

for every unary index $n : \mathsf{Nat}^{\uparrow}$. More generally, a sequence $N : \mathsf{Nat}^{\uparrow} \to R$ is coefficientwise inverse data for F when N is carried and, for every $n : \mathsf{Nat}^{\uparrow}$,

$$N_n \sim_R -_R(F_n).$$

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonCoeffwiseInverse`

Theorem 51.63 (FPS coefficientwise additive inverse zero laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and scalar additive inverse operation $-_R$. Assume the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive inverse closure, and additive left inverse rows retained in Definition 43.26; assume also the scalar right inverse law of Definition 43.27 for this addition and inverse. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, pointwise addition from Definition 51.22, and zero series 0_{fps} from Definition 51.60. If $F : \mathsf{Nat}^{\uparrow} \to R$ is carried and $N : \mathsf{Nat}^{\uparrow} \to R$ is coefficientwise inverse data for F in the sense of Definition 51.62, then $F \oplus_{\mathrm{fps}} N$ and $N \oplus_{\mathrm{fps}} F$ are carried and*

$$F \oplus_{\mathrm{fps}} N \sim_{\mathrm{fps}} 0_{\mathrm{fps}} \qquad \textit{and} \qquad N \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} 0_{\mathrm{fps}}.$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$,

$$(F \oplus_{\mathrm{fps}} N)_n \sim_R (0_{\mathrm{fps}})_n \qquad \textit{and} \qquad (N \oplus_{\mathrm{fps}} F)_n \sim_R (0_{\mathrm{fps}})_n.$$

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCoeffwiseInverse zero laws`

Proof. The definition of coefficientwise inverse data carries N. The pointwise addition closure row in Definition 51.16 carries $F \oplus_{\mathrm{fps}} N$ and $N \oplus_{\mathrm{fps}} F$, and Definition 51.61 carries the zero series.

Fix $n : \mathsf{Nat}^{\uparrow}$. The inverse-data hypothesis gives

$$N_n \sim_R -_R(F_n).$$

By Definition 51.22,

$$(F \oplus_{\mathrm{fps}} N)_n = F_n +_R N_n.$$

Scalar classifier reflexivity at F_n and scalar addition congruence transport the inverse-data comparison under the common left summand:

$$F_n +_R N_n \sim_R F_n +_R (-_R(F_n)).$$

The scalar right inverse law Definition 43.27 classifies the right endpoint with 0_R. Since Definition 51.60 gives $(0_{\mathrm{fps}})_n := 0_R$, transitivity gives

$$(F \oplus_{\mathrm{fps}} N)_n \sim_R (0_{\mathrm{fps}})_n.$$

For the other order, Definition 51.22 gives

$$(N \oplus_{\mathrm{fps}} F)_n = N_n +_R F_n.$$

Scalar addition congruence transports the same inverse-data comparison under the common right summand:

$$N_n +_R F_n \sim_R (-_R(F_n)) +_R F_n.$$

The scalar additive left inverse row retained in Definition 43.26 classifies this endpoint with 0_R, and the zero-series coefficient rule again gives $(0_{\mathrm{fps}})_n := 0_R$. Thus

$$(N \oplus_{\mathrm{fps}} F)_n \sim_R (0_{\mathrm{fps}})_n.$$

Since n was arbitrary, Definition 51.15 packages the two coefficientwise scalar comparisons as the two displayed $\mathsf{FPS}^{\uparrow}$ classifier judgments. □

Theorem 51.64 (FPS pointwise additive group laws)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and scalar additive inverse operation $-_R$. Assume the scalar classifier equivalence, zero self-classification, addition closure, addition congruence, additive associativity, additive zero rows, additive inverse closure, and both scalar additive inverse laws retained in Definition 43.26 and Definition 43.27. Then the data of Definition 51.60, together with the coefficientwise inverse operation $-_{\mathrm{fps}}$ of Definition 51.62, form the concrete additive group interface on $\mathsf{FPSCarrier}_R$: 0_{fps} is carried, $F \oplus_{\mathrm{fps}} G$ is carried whenever F and G are carried, $\oplus_{\mathrm{fps}}$ is pointwise congruent, $\oplus_{\mathrm{fps}}$ is associative under $\sim_{\mathrm{fps}}$, both zero laws hold under $\sim_{\mathrm{fps}}$, $-_{\mathrm{fps}}F$ is carried whenever F is carried, $-_{\mathrm{fps}}F$ is coefficientwise inverse data for F, and both inverse-zero laws*

$$F \oplus_{\mathrm{fps}} (-_{\mathrm{fps}}F) \sim_{\mathrm{fps}} 0_{\mathrm{fps}} \qquad \textit{and} \qquad (-_{\mathrm{fps}}F) \oplus_{\mathrm{fps}} F \sim_{\mathrm{fps}} 0_{\mathrm{fps}}$$

hold for every carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$.

Proof. The carrier, congruence, associativity, and zero-law fields are exactly the fields supplied by Definition 51.61, read over the carrier, classifier, pointwise addition, and zero series fixed in Definition 51.60.

It remains to fit the inverse operation into the same public algebra package. Let $F : \mathsf{Nat}^{\uparrow} \to R$ be carried. For each coefficient index $n : \mathsf{Nat}^{\uparrow}$, scalar additive inverse closure carries $-_R(F_n)$, so the coefficient rule $(-_{\mathrm{fps}}F)_n := -_R(F_n)$ carries the whole sequence $-_{\mathrm{fps}}F$. The same coefficient rule, together with scalar classifier reflexivity at $-_R(F_n)$, makes $-_{\mathrm{fps}}F$ coefficientwise inverse data for F in the sense of Definition 51.62.

Apply Definition 51.63 with $N = -_{\mathrm{fps}}F$. Its hypotheses are precisely the retained scalar inverse, congruence, closure, and zero rows, plus the right inverse law Definition 43.27. The conclusion gives the two displayed inverse-zero laws under $\sim_{\mathrm{fps}}$.

Thus the ten fields just listed form a single concrete additive group interface for the downstream formal-power-series algebra layer: zero carrier, sum carrier, pointwise congruence, associativity, left zero, right zero, inverse carrier, inverse-data fit, right inverse-zero, and left inverse-zero. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonPointwiseAdditiveGroup laws`

Theorem 51.65 (FPS zero series absorbs Cauchy product over RingUp scalars)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence fields, zero self-classification, additive congruence, additive left-zero, multiplication closure, and left and right zero multiplication rows retained in Definition 43.26. Let the $\mathsf{FPS}^{\uparrow}(R)$ interface use the pointwise classifier $\sim_{\mathrm{fps}}$ of Definition 51.15, the zero series $Z := 0_{\mathrm{fps}}$ of Definition 51.60, and the Cauchy coefficient rule of Definition 51.20, read over these $\mathsf{Ring}^{\uparrow}$ operations. For every carried coefficient sequence $F : \mathsf{Nat}^{\uparrow} \to R$, the two Cauchy products $Z \odot_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} Z$ are carried and*

$$Z \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} Z \qquad \textit{and} \qquad F \odot_{\mathrm{fps}} Z \sim_{\mathrm{fps}} Z.$$

Equivalently, for every unary coefficient index $n : \mathsf{Nat}^{\uparrow}$,

$$(Z \odot_{\mathrm{fps}} F)_n \sim_R Z_n \qquad \textit{and} \qquad (F \odot_{\mathrm{fps}} Z)_n \sim_R Z_n.$$

Proof. The zero series is carried coefficientwise because Definition 51.60 gives $Z_n = 0_R$ for every n, and the scalar zero self-classification row supplies the carried scalar endpoint. The Cauchy-product closure row in Definition 51.16 therefore carries $Z \odot_{\mathrm{fps}} F$ and $F \odot_{\mathrm{fps}} Z$ from the carriedness of Z and F.

Fix $n : \mathsf{Nat}^{\uparrow}$, and write ZeroSp_n for the finite spine indexed by $\mathsf{Split}(n)$ whose every displayed entry is 0_R. For the left product, each split $(i, j) \in \mathsf{Split}(n)$ contributes the scalar summand

$$Z_i \cdot_R F_j.$$

The zero-series coefficient rule gives the scalar comparison $Z_i \sim_R 0_R$. Together with scalar classifier reflexivity at F_j, the multiplication-congruence row retained in Definition 43.26 compares $Z_i \cdot_R F_j$ with $0_R \cdot_R F_j$; the left zero multiplication row retained in Definition 43.26 then classifies $0_R \cdot_R F_j$ with 0_R. If the coefficient comparison, multiplication-congruence step, or zero multiplication row is read in the opposite orientation, classifier symmetry supplies the displayed orientation. Structural induction on the finite split spine, using the nil and cons clauses of Definition 36.12, assembles these pointwise scalar comparisons into

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(Z, F), \mathsf{ZeroSp}_n).$$

Applying Definition 50.27 to this all-zero Cauchy summand witness gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(Z, F)) \sim_R 0_R.$$

By the Cauchy coefficient rule in Definition 51.20, the left endpoint is $(Z \odot_{\mathrm{fps}} F)_n$. By the zero-series coefficient rule, $Z_n = 0_R$. Hence

$$(Z \odot_{\mathrm{fps}} F)_n \sim_R Z_n.$$

For the right product, each split $(i, j) \in \mathsf{Split}(n)$ contributes

$$F_i \cdot_R Z_j.$$

The zero-series coefficient rule gives the scalar comparison $Z_j \sim_R 0_R$. Together with scalar classifier reflexivity at F_i, the multiplication-congruence row retained in Definition 43.26 compares $F_i \cdot_R Z_j$ with $F_i \cdot_R 0_R$; the right zero multiplication row classifies $F_i \cdot_R 0_R$ with 0_R, again using classifier symmetry if a row is oriented oppositely. The same finite-spine induction over Definition 36.12 yields

$$\mathsf{ListClassifierSpec}(\sim_R, \mathsf{CauchySp}_n(F, Z), \mathsf{ZeroSp}_n).$$

A second application of Definition 50.27 gives

$$\mathsf{fold}_{+,R}(\mathsf{CauchySp}_n(F, Z)) \sim_R 0_R,$$

which is exactly

$$(F \odot_{\mathrm{fps}} Z)_n \sim_R Z_n$$

after unfolding Definition 51.20 and the coefficient rule for Z.

Since n was arbitrary, the pointwise classifier specification in Definition 51.15 packages the two coefficientwise scalar comparisons as

$$Z \odot_{\mathrm{fps}} F \sim_{\mathrm{fps}} Z \qquad \text{and} \qquad F \odot_{\mathrm{fps}} Z \sim_{\mathrm{fps}} Z.$$

□

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonAddFold zero series absorption`

Theorem 51.66 (FPS Cauchy product distributes over pointwise addition)**.** *In the concrete singleton formal-power-series carrier, let F, G, and H be singleton coefficient histories. Write $\oplus_{\mathrm{fps}}$ for pointwise addition and $\odot_{\mathrm{fps}}$ for the Cauchy product defined by the singleton coefficient spine. Then both distributed product comparisons are classified by* FpsSingletonClassifier*:*

$$F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H) \sim_{\mathsf{FpsSingletonClassifier}} (F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H)$$

and

$$(F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathsf{FpsSingletonClassifier}} (F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H).$$

Moreover, the two undistributed Cauchy-product endpoints

$$F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H) \qquad \textit{and} \qquad (F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H$$

are each classified with the empty history endpoint.

Proof. The pointwise addition interface in Definition 51.60 supplies the singleton addition endpoints, and the Cauchy-product coefficient rule in Definition 51.20 expresses each product coefficient as the finite additive fold over the split spine.

In the singleton carrier, every displayed coefficient spine collapses to the empty history endpoint. The checked zero-series absorption row above gives the same empty endpoint for Cauchy products whose split spine has only empty scalar contributions, and the continuation endpoint rows identify the folded singleton coefficients with that endpoint.

Applying these endpoint identifications on the left-distributive spine gives a classifier comparison between $F \odot_{\mathrm{fps}} (G \oplus_{\mathrm{fps}} H)$ and $(F \odot_{\mathrm{fps}} G) \oplus_{\mathrm{fps}} (F \odot_{\mathrm{fps}} H)$. The same argument on the right-distributive spine gives the comparison between $(F \oplus_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H$ and $(F \odot_{\mathrm{fps}} H) \oplus_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)$. The singleton empty-carrier collapse also records the two undistributed product endpoints as classified with $\mathsf{BHist.Empty}$. □

↪variant BEDC.Derived.FpsUp.FpsSingletonCauchyProduct distributes over addition classifier

Theorem 51.67 (FPS RingUp tail algebra package)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, and finite additive fold $\mathsf{fold}_{+,R}$. Assume the scalar classifier equivalence, zero self-classification, additive closure, additive congruence, additive associativity, additive zero, additive inverse closure, additive inverse laws, multiplication closure, multiplication congruence, multiplicative associativity, distributivity, and zero multiplication rows retained in Definition 43.26, together with the scalar right inverse law of Definition 43.27. Then the carried coefficient sequences of $\mathsf{FPS}^{\uparrow}(R)$, with pointwise classifier $\sim_{\mathrm{fps}}$, pointwise addition $\oplus_{\mathrm{fps}}$, zero series 0_{fps}, coefficientwise inverse $-_{\mathrm{fps}}$, and Cauchy product $\odot_{\mathrm{fps}}$, jointly provide the RingUp-scoped tail algebra package consisting of:*

1. *the pointwise additive group laws on $\mathsf{FPSCarrier}_R$;*
2. *Cauchy-product carrier closure, congruence, and associativity on the same carrier and classifier;*
3. *both distributivity laws of $\odot_{\mathrm{fps}}$ over $\oplus_{\mathrm{fps}}$;*
4. *zero-series Cauchy-product absorption on the left and on the right.*

Proof. The carrier and classifier named in every row are the common $\mathsf{FPSCarrier}_R$ of Definition 51.1 and the pointwise classifier specified in Definition 51.15. Thus the statement does not introduce a second formal-power-series surface; it assembles the already established laws over one shared interface.

For the additive part, apply Definition 51.64. It supplies the carried zero series, pointwise sum carrier closure and congruence, associativity, both zero laws, coefficientwise inverse carrier, inverse-data fit, and both inverse-zero laws.

For the multiplicative spine, apply Definition 51.54. It supplies Cauchy-product carrier closure, congruence with respect to $\sim_{\mathrm{fps}}$, and associativity over the same carried coefficient sequences.

The distributive rows are exactly the two conclusions of Definition 51.27, read with the same pointwise addition and Cauchy product. The zero absorption rows are exactly Definition 51.65. Combining these four cited theorem rows yields the displayed RingUp-scoped tail package for downstream formal-power-series algebra. □

Lean4 ✓ BEDC.Derived.FpsUp.FpsSingletonRingupTailAlgebra laws
Lean4 ✓ BEDC.Derived.FpsUp.FpsSingletonRingUpTailAlgebra package

Theorem 51.68 (FPS singleton empty-history semantic certificate)**.** *The singleton empty-history formal-power-series carrier $A_{\mathsf{e}}(h) := h\mathsf{hsamee}$ on BHist, together with the classifier*

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge h\mathsf{hsame}k,$$

forms the semantic naming certificate for the terminal formal-power-series history object. The RingUp tail algebra package of Definition 51.67 is read over this same BHist carrier/classifier surface when the terminal FPS object is used; it does not introduce a second formal-power-series interface.

Proof. The singleton carrier and classifier are the encoded BHist data of Definition 51.5. The Lean-checked certificate $\mathsf{FpsSingletonEmptyHistory_semanticNameCert}$ supplies the semantic name-certificate fields for exactly this carrier/classifier pair.

The tail algebra package in Definition 51.67 assembles the additive, multiplicative, distributive, and zero-absorption rows over one shared formal-power-series carrier and pointwise classifier.

Reading that package at the terminal singleton object therefore keeps the algebra rows on the same BHist naming surface certified above, rather than passing through a separate FPS naming interface. □

↪ variant BEDC.Derived.FpsUp.FpsSingletonEmptyHistory semanticNameCert

Theorem 51.69 (FPS RingUp public algebra interface)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ scalar certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, unit endpoint 1_R, and finite additive fold $\mathsf{fold}_{+,R}$. Then the shared $\mathsf{FPS}^{\uparrow}(R)$ carrier of coefficient sequences, equipped with the pointwise classifier $\sim_{\mathrm{fps}}$, exports one public algebra interface whose operations are pointwise addition $\oplus_{\mathrm{fps}}$, the zero series 0_{fps}, coefficientwise inverse $-_{\mathrm{fps}}$, Cauchy product $\odot_{\mathrm{fps}}$, and the Cauchy unit series 1_{fps}.*

This public interface consumes exactly the rows of Definition 51.64, Definition 51.57, Definition 51.66, Definition 51.65, and Definition 51.68: additive group laws, multiplicative monoid laws, distributivity, zero-product absorption, and the terminal singleton semantic naming certificate.

Proof. The carrier appearing in every cited algebra row is the coefficientwise $\mathsf{FPS}^{\uparrow}(R)$ carrier, and the classifier is always the pointwise classifier $\sim_{\mathrm{fps}}$. Thus the additive, multiplicative, distributive, zero-absorption, and singleton semantic rows are read over one shared family of carried coefficient sequences.

The additive operations come from Definition 51.64; the multiplicative product and unit operations come from Definition 51.57. The compatibility of the two operation families is supplied by Definition 51.66 and Definition 51.65.

Finally, Definition 51.68 records that the terminal singleton reading stays on the same naming surface. Repacking these rows therefore exports a single public $\mathsf{FPS}^{\uparrow}(R)$ algebra surface, not a second formal-power-series carrier with separately transported operations. □

Lean4 ✓ BEDC.Derived.FpsUp.FpsRingup public algebra interface

Theorem 51.70 (FPS concrete RingUp instance certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}(R)$ scalar certificate with scalar classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, additive inverse $-_R$, multiplication $\cdot_R$, unit endpoint 1_R, and finite additive fold $\mathsf{fold}_{+,R}$ as in Definition 51.69. Then the concrete formal-power-series surface*

$$(\mathsf{FPSCarrier}_R,\ \sim_{\mathrm{fps}},\ \oplus_{\mathrm{fps}},\ 0_{\mathrm{fps}},\ -_{\mathrm{fps}},\ \odot_{\mathrm{fps}},\ 1_{\mathrm{fps}})$$

forms the corresponding $\mathsf{Ring}^{\uparrow}$ instance surface: the additive part is a pointwise group over $\mathsf{FPSCarrier}_R$, the multiplicative part is a Cauchy-product monoid on the same carrier and classifier, Cauchy product distributes over pointwise addition on both sides, the zero series absorbs Cauchy product on both sides, and the singleton empty-history naming certificate records the semantic carrier used by the terminal concrete instance.

Proof. The carrier, classifier, and displayed operation names are exactly the public interface exported in Definition 51.69; hence the certificate only has to assemble the algebra rows already stated there.

Apply Definition 51.64 for the pointwise additive group rows on $\mathsf{FPSCarrier}_R$. Apply Definition 51.57 for Cauchy-product closure, congruence, associativity, and the unit series on the same pointwise classifier.

The interaction rows are supplied by Definition 51.66 and Definition 51.65. They identify the two distributivity laws and the two zero-product absorption laws without changing the carrier or classifier.

Finally, Definition 51.68 records the singleton semantic naming surface used when the concrete instance is read at the terminal empty-history object. Together these rows are the concrete $\mathsf{Ring}^{\uparrow}$ instance certificate for the displayed FPS carrier and operations. □

Theorem 51.71 (FPS RingUp standard bridge boundary)**.** *The public algebra interface of Definition 51.69 and the concrete instance certificate of Definition 51.70 form the paper bridge from the BEDC coefficientwise FPS carrier to the standard formal-power-series ring schema. The bridge boundary sends the carrier to coefficientwise carried sequences over* $\mathsf{Nat}^{\uparrow}$*, sends the classifier to the pointwise scalar classifier, sends addition and inverse to their pointwise operations, sends multiplication to the Cauchy product through finite split spines, and sends zero and unit to the displayed scalar rows. It exports no analytic convergence, topology, completion, or external library quotient.*

Proof. First read the public surface from Definition 51.69. It fixes one carrier $\mathsf{FPSCarrier}_R$, one pointwise classifier $\sim_{\mathrm{fps}}$, and the five operation rows $\oplus_{\mathrm{fps}}$, 0_{fps}, $-_{\mathrm{fps}}$, $\odot_{\mathrm{fps}}$, and 1_{fps}.

Next apply Definition 51.70. It repacks that same surface as a concrete $\mathsf{Ring}^{\uparrow}$ instance, with the additive group rows, Cauchy-product monoid rows, distributivity rows, zero-product absorption rows, and singleton semantic certificate all kept on the displayed carrier and classifier.

The Cauchy-product part is the only non-pointwise operation. Its coefficient is defined by the finite split spine of Definition 51.20, and the finite-fold transport rows already used in Definition 50.26 carry the scalar comparisons through those spines.

Thus the bridge boundary is exactly the public coefficientwise $\mathsf{Ring}^{\uparrow}$ schema: carrier and classifier are pointwise, additive structure is pointwise, multiplication is the displayed finite Cauchy fold, and the excluded analytic or quotient data are not part of the exported interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$**.**

Constructive story: A formal power series over a certified ring is built as a coefficient sequence indexed by $\mathsf{Nat}^{\uparrow}$, with every coefficient carried by the same $\mathsf{Ring}^{\uparrow}$ scalar certificate. The carrier accepts exactly coefficientwise carried sequences, the classifier compares two series pointwise by the scalar classifier at each unary index, and sequences with an uncarried coefficient or a missing pointwise comparison stay outside the public algebra surface. The source is the scalar $\mathsf{Ring}^{\uparrow}$ data together with the coefficientwise carrier, the pattern records pointwise addition, zero series, coefficientwise inverse, Cauchy product through finite split spines, the unit series, distributivity, and zero-product absorption, while the ledger keeps each finite fold, split witness, and scalar coefficient comparison visible. The name-certificate obligations are anchored by the singleton empty-history semantic certificate for the terminal instance and by the scalar rows for the general interface: carried zero or unit sequences inhabit the relevant carrier, pointwise classifier equivalence gives reflexivity, symmetry, and transitivity, and coefficientwise transport preserves carried series under classified replacement. A sequence history h lies in the signature gap of a $\mathsf{FPS}^{\uparrow}$ token p when the source-domain accepts all coefficients of h, the signature relation pairs it with an answer series and evidence ledger of coefficient witnesses and finite folds, and token introduction names that answer as p; unfolded, every unary coefficient matches the scalar endpoint encoded by p and every product coefficient is the displayed Cauchy finite-fold value. The bridge boundary is the public coefficientwise $\mathsf{Ring}^{\uparrow}$ interface, with no analytic convergence, completion, or separate formal-series carrier introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public FPS RingUp bridge boundary, coefficientwise carrier and classifier, finite Cauchy-fold product, zero absorption rows, and consumer-completeness packaging are closed over $\mathsf{Ring}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, the pointwise operations, Cauchy unit series, distributivity rows, singleton semantic certificate, and displayed finite-fold witnesses in this child file.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FPS}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FpsUp.FpsRingup\_public\_algebra\_interface`

Bridge status: bridgeChecked

Not claimed: This chapter does not close analytic convergence, topology, completion, or bridges to standard formal-power-series libraries.

Upgrade path: no further upgrade – chapter at top of formal axis

Definition 51.72 (FPS threefold Cauchy split spines)**.** *Let R be read as a* $\mathsf{Ring}^{\uparrow}(R)$ *certificate with scalar classifier $\sim_R$, addition $+_R$, multiplication $\cdot_R$, zero endpoint 0_R, and finite additive fold*

$\mathsf{fold}_{+,R}$. *For coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, define the left and right threefold split spines by*

$$\mathsf{TripSp}^{\ell}_{n}(F, G, H) := \mathsf{concat}_{(a,k)\in\mathsf{Split}(n)}\mathsf{map}_{(i,j)\in\mathsf{Split}(a)}\big((F_i \cdot_R G_j) \cdot_R H_k\big)$$

and

$$\mathsf{TripSp}^{r}_{n}(F, G, H) := \mathsf{concat}_{(i,c)\in\mathsf{Split}(n)}\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}\big(F_i \cdot_R (G_j \cdot_R H_k)\big).$$

The left spine enumerates witnesses (i, j, k) through $(i + j) + k = n$; the right spine enumerates the same triples through $i + (j + k) = n$.

Lean4 (definition) `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines`

Lemma 51.73 (FPS threefold Cauchy split reassociation)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate, read through Definition 44.8 as a $\mathsf{Ring}^{\uparrow}(R)$ certificate. For carried coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$ and unary index n, the two threefold split folds of Definition 51.72 are scalar-classified:*

$$\mathsf{fold}_{+,R}(\mathsf{TripSp}^{\ell}_{n}(F, G, H)) \sim_R \mathsf{fold}_{+,R}(\mathsf{TripSp}^{r}_{n}(F, G, H)).$$

Proof. The split spine $\mathsf{Split}(n)$ of Definition 51.20 is finite. Hence each displayed spine in Definition 51.72 is a finite concatenation of finite split spines. Its entries are carried scalars because the coefficient sequences are carried and the multiplication closure row is retained by Definition 43.26.

The left indexing data are quadruples $((a, k), (i, j))$ with $a + k = n$ and $i + j = a$. Sending this data to (i, j, k) gives a triple with $(i + j) + k = n$. The right indexing data are quadruples $((i, c), (j, k))$ with $i + c = n$ and $j + k = c$. Sending this data to (i, j, k) gives a triple with $i + (j + k) = n$. Unary additive associativity for the $\mathsf{Nat}^{\uparrow}$ split recursion identifies these two triple predicates, and the two reconstructions

$$(i, j, k) \mapsto ((i + j, k), (i, j)), \qquad (i, j, k) \mapsto ((i, j + k), (j, k))$$

are inverse on the displayed split witnesses.

Apply the split-spine instance of the two-index finite fold swap certificate of Definition 49.94 to this finite reindexing. It reduces the comparison of the two iterated folds to the pointwise scalar comparison attached to a fixed triple (i, j, k). For that triple, the retained multiplicative associativity row in Definition 43.26 gives

$$(F_i \cdot_R G_j) \cdot_R H_k \sim_R F_i \cdot_R (G_j \cdot_R H_k).$$

The fold-swap certificate and Definition 50.26 lift these pointwise comparisons through the finite spines. This is the displayed scalar classification. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines classifier`
Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonThreefoldCauchySplitSpines reassociation`

Theorem 51.74 (FPS Cauchy product associativity from CommRing scalars)**.** *Let R carry a $\mathsf{CommRing}^{\uparrow}(R)$ certificate. In the $\mathsf{FPS}^{\uparrow}(R)$ interface, let $\sim_{\mathrm{fps}}$ be the pointwise classifier of Definition 51.15. For carried coefficient sequences $F, G, H : \mathsf{Nat}^{\uparrow} \to R$,*

$$\forall n : \mathsf{Nat}^{\uparrow}, \quad \big((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H\big)_n \sim_R \big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n.$$

Consequently,

$$(F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H \sim_{\mathrm{fps}} F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H).$$

Proof. Read the scalar certificate as a $\mathsf{Ring}^{\uparrow}(R)$ certificate by Definition 44.8. This keeps the scalar classifier, zero endpoint, addition, multiplication, additive abelian-group rows, multiplication congruence, the distributive rows, and multiplicative associativity from Definition 43.26. The finite additive fold is the fold used in Definition 51.20.

Fix $n : \mathsf{Nat}^{\uparrow}$. By Definition 51.20, the left-associated coefficient is

$$\mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_n(F \odot_{\mathrm{fps}} G, H)\big).$$

For a split $(a, k) \in \mathsf{Split}(n)$, the corresponding summand is

$$(F \odot_{\mathrm{fps}} G)_a \cdot_R H_k.$$

Unfolding the inner product at a gives

$$\mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_a(F, G)\big) \cdot_R H_k.$$

The right-multiplication half of Definition 49.95 classifies this endpoint with the inner finite fold over

$$\mathsf{map}_{(i,j)\in\mathsf{Split}(a)}\big((F_i \cdot_R G_j) \cdot_R H_k\big).$$

Applying Definition 50.26 over the outer split spine lifts these inner comparisons and gives

$$\big((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H\big)_n \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{\ell}_n(F, G, H)\big).$$

The right-associated coefficient expands similarly:

$$\big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n = \mathsf{fold}_{+,R}\big(\mathsf{CauchySp}_n(F, G \odot_{\mathrm{fps}} H)\big).$$

For a split $(i, c) \in \mathsf{Split}(n)$, the summand is

$$F_i \cdot_R (G \odot_{\mathrm{fps}} H)_c.$$

Unfolding the inner product at c and applying the left-multiplication half of Definition 49.95 classifies this summand with the inner fold over

$$\mathsf{map}_{(j,k)\in\mathsf{Split}(c)}\big(F_i \cdot_R (G_j \cdot_R H_k)\big).$$

Finite-fold congruence over the outer split spine gives

$$\big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{r}_n(F, G, H)\big).$$

By Definition 51.73,

$$\mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{\ell}_n(F, G, H)\big) \sim_R \mathsf{fold}_{+,R}\big(\mathsf{TripSp}^{r}_n(F, G, H)\big).$$

Classifier transitivity composes the left expansion, this threefold split comparison, and the symmetry of the right expansion, giving

$$\big((F \odot_{\mathrm{fps}} G) \odot_{\mathrm{fps}} H\big)_n \sim_R \big(F \odot_{\mathrm{fps}} (G \odot_{\mathrm{fps}} H)\big)_n.$$

Since n was arbitrary, the pointwise classifier specification of Definition 51.15 yields the displayed $\mathsf{FPS}^{\uparrow}$ classifier judgment. The closure of the two products is the Cauchy-product closure field listed in Definition 51.16, and all folds are finite at the fixed coefficient index. □

Lean4 ✓ `BEDC.Derived.FpsUp.FpsSingletonCauchyProduct carried assoc continuation classifier`

Chapter 52

A Concrete Naming Certificate for $\mathsf{Preorder}^{\uparrow}$

The preorder interface $\mathsf{Preorder}^{\uparrow}$ is licensed by a carrier with a binary relation $\leq : C \times C \to \mathsf{Prop}$ satisfying reflexivity and transitivity under the carrier's classifier. It is the simplest order-theoretic NameCert and self-contained: no other licensed object is required as a dependency.

On the source side, the preorder certificate is kernel-local: the carrier, classifier, and prefix-comparison ledger are all discharged inside this chapter. Its first order-theoretic sibling consumer is $\mathsf{POSet}^{\uparrow}$ in chapter 53, where the partially ordered set certificate adds the antisymmetry row while preserving the inherited preorder carrier, classifier, and prefix-comparison ledger.

Remark 52.1 (Preorder certificate scope)**.** *This chapter records the preorder licensing target through a history carrier, restricted classifier, and reflexive-transitive relation data.*

52.1 Carrier definition

Definition 52.2 (Preorder carrier)**.** *Use an abstract carrier C, classifier $\sim_C$, and relation $\leq_C$ with preorder law fields.*

52.2 Concrete unary prefix preorder instance

Definition 52.3 (Concrete unary prefix preorder instance)**.** *The concrete unary prefix preorder instance is the following specialization of the* $\mathsf{Preorder}^{\uparrow}$ *source data:*

$$\mathsf{Carrier} := \mathsf{PreorderCarrier} = \mathsf{UnaryHistory}.$$

$$\mathsf{classifier} := \mathsf{PreorderCarrierClassifierSpec}(\mathsf{PreorderCarrier}).$$

$$\mathsf{le} := \mathsf{PreorderPrefixLE}.$$

Theorem 52.4 (Concrete unary prefix preorder laws)**.** *For the concrete unary prefix preorder instance, the carrier is inhabited by* e*, carrier membership transports along* hsame*, the relation* $\mathsf{PreorderPrefixLE}$ *is reflexive on carrier histories, transitive, and congruent under the classifier in both arguments.*

Proof. The carrier and transport fields are supplied by the checked semantic naming certificate named BEDC.Derived.PreorderUp. preorder_carrier_semantic_name_certificate. This identifies the concrete carrier with unary histories and gives the hsame transport operation used by the restricted classifier.

The relational fields come from the checked concrete prefix stability certificate named BEDC.Derived.PreorderUp. preorder_prefix_stability_certificate_fields. Its reflexivity component is the carrier-relative preorder reflexivity field, its congruence component is the classifier-respecting transport of the prefix relation in both endpoints, and its transitivity component is the concrete prefix composition law.

At the theorem level, reflexivity is obtained by applying BEDC.Derived.PreorderUp.PreorderPrefixLE_of_hsame to the reflexive hsame witness, while transitivity is BEDC.Derived.PreorderUp.PreorderPrefixLE_trans. Repacking these fields gives the concrete unary prefix preorder laws. □

Thus the $\mathsf{Preorder}^{\uparrow}$ schema is instantiated by this concrete BHist carrier and relation. The corresponding formal target can therefore address the concrete prefix preorder directly, rather than only the relation fields of an arbitrary carrier.

Definition 52.5 (Prefix-history preorder instance)**.** *The prefix-history preorder instance pins the* $\mathsf{Preorder}^{\uparrow}$ *carrier and relation to the concrete kernel rows*

$$\mathsf{Carrier} := \mathsf{UnaryHistory}, \qquad \mathsf{relation} := \mathsf{PreorderPrefixLE}.$$

Its law row consists of reflexivity from hsame *via* PreorderPrefixLE_of_hsame *and transitivity via* PreorderPrefixLE_trans*.*

Theorem 52.6 (Prefix-history preorder laws)**.** *For the instance of Definition 52.5, every carrier history is related to itself, and whenever* $\mathsf{PreorderPrefixLE}(h,k)$ *and* $\mathsf{PreorderPrefixLE}(k,r)$ *hold,* $\mathsf{PreorderPrefixLE}(h,r)$ *holds. Thus the concrete* BHist *prefix relation supplies a preorder instance without adding an abstract carrier parameter or a separate relation schema.*

Proof. For reflexivity, apply PreorderPrefixLE_of_hsame to the reflexive hsame witness on the displayed UnaryHistory carrier.

For transitivity, apply PreorderPrefixLE_trans to the two displayed prefix comparisons. The resulting law row is exactly the concrete instance row named in the definition, so no additional carrier or classifier parameter is introduced. □

52.3 Concrete singleton empty preorder instance

Definition 52.7 (Concrete singleton empty preorder instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$*. Define the restricted classifier by*

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Define the carrier-guarded restricted order by

$$h \leq_{\mathsf{e}}^{\mathrm{sg}} k := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{PreorderPrefixLE}(h,k).$$

The concrete preorder package is the fixed BHist *tuple* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}}^{\mathrm{sg}})$*.*

Theorem 52.8 (Singleton empty preorder laws)**.** *For the concrete package of Definition 52.7, the carrier is inhabited by* e*, carrier membership transports along* hsame*,* $\leq_{\mathsf{e}}^{\mathrm{sg}}$ *is reflexive on carried histories and transitive, and* $\leq_{\mathsf{e}}^{\mathrm{sg}}$ *is congruent under* ρ_{e} *in both endpoints. Moreover, Definition 52.9 gives the exact classifier row for all histories.*

Proof. The singleton source supplies the inhabitance witness $A_{\mathsf{e}}(\mathsf{e})$. Unfolding the source shows that any carried endpoint is identified with e by hsame, and transport along hsame is exactly the transport law recorded for the singleton history source.

Reflexivity follows by applying $\mathsf{PreorderPrefixLE_of_hsame}$ to the reflexive witness $\mathsf{hsame}(h, h)$ for each carried history h, then pairing the prefix witness with the two copies of the carrier witness for h. Transitivity projects the two guarded comparisons to their prefix witnesses and applies $\mathsf{PreorderPrefixLE_trans}$; the endpoint carrier witnesses are carried by the first and second guarded comparisons.

For congruence, unfold ρ_{e} at the source and target endpoints. The classifier fields provide the carrier witnesses and the hsame witnesses needed to move each endpoint. The prefix stability certificate for $\mathsf{PreorderPrefixLE}$ then transports the comparison in the left endpoint and in the right endpoint, after which the transported endpoint carriers are repacked into the guarded comparison.

The exact classifier row is Definition 52.9. □

Theorem 52.9 (Singleton guarded prefix preorder exactness)**.** *For all histories $h, k : \mathsf{BHist}$ in the concrete singleton empty preorder instance,*

$$\rho_{\mathsf{e}}(h, k) \Longleftrightarrow (h \leq_{\mathsf{e}}^{\mathrm{sg}} k \wedge k \leq_{\mathsf{e}}^{\mathrm{sg}} h).$$

Proof. For the forward direction, unfold $\rho_{\mathsf{e}}(h, k)$. The classifier witness contains the two carrier witnesses and the history-sameness witness $\mathsf{hsame}(h, k)$. Apply Definition 52.17 to $\mathsf{hsame}(h, k)$ to obtain the prefix comparison from h to k.

Apply the same theorem to the symmetric history-sameness witness to obtain the prefix comparison from k to h. Pair each prefix comparison with the two carrier witnesses in the appropriate order; this gives the two guarded comparisons.

For the reverse direction, project the carrier witnesses from either guarded comparison. Project the prefix witnesses from both guarded comparisons and apply Definition 52.30 to recover $\mathsf{hsame}(h, k)$.

Finally, combine the carrier witnesses with this history-sameness witness. This is exactly the restricted classifier $\rho_{\mathsf{e}}(h, k)$. □

Theorem 52.10 (Singleton empty preorder terminal comparison package)**.** *For histories $h, k :$ BHist carried by A_{e} in the concrete singleton empty preorder instance, the restricted comparison package is pinned to the terminal endpoint:*

$$A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \Longrightarrow h \leq_{\mathsf{e}}^{\mathrm{sg}} k \wedge k \leq_{\mathsf{e}}^{\mathrm{sg}} h \wedge \rho_{\mathsf{e}}(h, k).$$

Its ten-field fit is the following: the traditional target is the one-point preorder; the BEDC source is BHist *with singleton carrier A_{e}; the carrier is A_{e}; the classifier is ρ_{e}; stability is supplied by singleton endpoint transport and Definition 52.17; the ledger records the two endpoint witnesses; the core theorem is the displayed terminal comparison package; exactness is guarded prefix exactness; the standard bridge is the one-point preorder; and the status is* $\mathsf{ProofTarget}$.

Proof. Assume $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. Unfolding the singleton carrier witnesses identifies both histories with the endpoint e, giving $h\mathsf{hsame}\mathsf{e}$ and $k\mathsf{hsame}\mathsf{e}$.

Compose the first witness with the symmetry of the second to obtain $h\mathsf{hsame}k$, and compose the second witness with the symmetry of the first to obtain $k\mathsf{hsame}h$.

Apply Definition 52.17 to these two hsame witnesses. The first application gives the prefix part of $h \leq_{\mathsf{e}}^{\mathrm{sg}} k$, and the second gives the prefix part of $k \leq_{\mathsf{e}}^{\mathrm{sg}} h$.

Finally, package $\rho_{\mathsf{e}}(h, k)$ from the two carrier witnesses together with the composed witness $h\mathsf{hsame}k$. Repack the two carrier witnesses with the two prefix witnesses to obtain the guarded

comparisons; these comparisons and the classifier witness are exactly the displayed terminal comparison package. □

Definition 52.11 (Singleton empty preorder public comparison readback)**.** *For the concrete singleton empty preorder instance, the public readback of the guarded comparison $h \leq_{\mathsf{e}}^{\mathrm{sg}} k$ is the endpoint-carrier pair*

$$A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Theorem 52.12 (Singleton empty preorder single-comparison exactness)**.** *For all histories $h, k :$ BHist in the concrete singleton empty preorder instance, one guarded comparison is exactly endpoint carriage:*

$$h \leq_{\mathsf{e}}^{\mathrm{sg}} k \Longleftrightarrow A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Its ten-field fit is the following: the traditional target is one-point preorder comparison readback; the BEDC source is BHist; the carrier is A_{e}; the classifier is ρ_{e}; the relation is $\leq_{\mathsf{e}}^{\mathrm{sg}}$; stability is supplied by singleton transport and Definition 52.17; the ledger records the two endpoint witnesses and the single comparison witness; the core theorem is this single-comparison exactness block; exactness is the displayed if-and-only-if; the standard bridge is the one-point preorder; and the status is $\mathsf{ProofTarget}$.

Proof. For the forward direction, unfold the guarded relation $h \leq_{\mathsf{e}}^{\mathrm{sg}} k$. Its witness is a package consisting of $A_{\mathsf{e}}(h)$, $A_{\mathsf{e}}(k)$, and $\mathsf{PreorderPrefixLE}(h, k)$. Project the first two fields to obtain the endpoint-carrier pair.

For the reverse direction, assume $A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k)$. Apply Definition 52.10 to these two carrier witnesses. The resulting terminal comparison package contains $h \leq_{\mathsf{e}}^{\mathrm{sg}} k$ as its first component, which is the required single guarded comparison. □

Theorem 52.13 (Singleton empty $\mathsf{Preorder}^{\uparrow}$ naming certificate)**.** *The tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}}^{\mathrm{sg}})$ of Definition 52.7 supplies the source, pattern, classifier, stability, ledger, and singleton exactness rows for a concrete $\mathsf{Preorder}^{\uparrow}$ certificate over A_{e}, ρ_{e}, and $\leq_{\mathsf{e}}^{\mathrm{sg}}$. The exactness row is*

$$\rho_{\mathsf{e}}(h, k) \Longleftrightarrow (h \leq_{\mathsf{e}}^{\mathrm{sg}} k \wedge k \leq_{\mathsf{e}}^{\mathrm{sg}} h).$$

Proof. The carrier row is the singleton BHist source A_{e}, and Definition 52.8 supplies its inhabitance by e together with transport along hsame. The pattern row is the restricted comparison judgment $h \leq_{\mathsf{e}}^{\mathrm{sg}} k$, and the classifier row is ρ_{e}.

The same law package gives the stability rows: reflexivity of $\leq_{\mathsf{e}}^{\mathrm{sg}}$ on carried histories, transitivity, and endpoint congruence for the classifier. The ledger row is Definition 52.45, which packages the two carried endpoints and the restricted comparison witness.

Definition 52.9 gives the displayed two-sided exactness statement for all histories. That theorem is exactly the singleton exactness row of the concrete $\mathsf{Preorder}^{\uparrow}$ certificate. □

Theorem 52.14 (Preorder finite comparison induction)**.** *For the concrete singleton empty preorder package $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}}^{\mathrm{sg}})$, every public singleton comparison read is generated by finite package induction from exactly three visible rows: endpoint carrier witnesses $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, guarded prefix exactness for $\leq_{\mathsf{e}}^{\mathrm{sg}}$, and the terminal comparison package of Definition 52.10. A reader that reaches a comparison*

$$h \leq_{\mathsf{e}}^{\mathrm{sg}} k$$

therefore has already passed through the endpoint carrier row and can recover the public readback

$$A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k)$$

without invoking an ambient order, quotient equality, selected representative, or external preorder carrier.

Proof. Begin with the finite carrier row. A singleton comparison read is addressed to the displayed package $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}}^{\mathrm{sg}})$, so its endpoints must first be inspected through the endpoint carrier witnesses of A_{e}.

The base package step is the terminal endpoint. When the two endpoint carrier witnesses are present, Definition 52.10 produces both guarded comparisons and the classifier witness $\rho_{\mathsf{e}}(h, k)$ from the same finite data.

The induction step is guarded exactness. If a public reader starts with a single guarded comparison, then Definition 52.12 projects it back to $A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k)$; if the reader starts with the classifier row, Definition 52.9 projects the two guarded comparisons from the same endpoint witnesses.

Thus every route through the singleton comparison surface factors through endpoint carriage, guarded prefix exactness, and terminal comparison packaging. No additional order object or representative choice is available in the finite package. □

This singleton package gives a concrete BHist source for the $\mathsf{Preorder}^{\uparrow}$ relation fields, so order formalization can target this carrier rather than arbitrary relation parameters.

52.4 The certificate

Definition 52.15 (Preorder source specification)**.** *The source is the carrier C with its classifier and relation.*

Definition 52.16 (Preorder pattern specification)**.** *The pattern is relational judgment formation $a \leq_C b$ over carrier witnesses.*

Theorem 52.17 (Prefix from history sameness)**.** *If h and k are internally history-same, then $h \leq k$ in the concrete unary history prefix relation.*

Proof. Eliminate the $\mathsf{hsame}(h, k)$ witness. The prefix relation is then witnessed by the empty unary tail together with the right-unit continuation law for h. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE of hsame`

Theorem 52.18 (Prefix append by unary tail)**.** *For every history h, appending a unary tail to h gives a concrete prefix relation from h to the append result.*

Proof. Use the unary tail itself as the prefix witness. The continuation component is the definitional append equation, introduced by the BEDC continuation constructor. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append unary tail`

Theorem 52.19 (Prefix from empty is unary carrier)**.** *For the concrete unary history preorder, $\mathsf{e} \leq h$ holds exactly when h is in the preorder carrier.*

Proof. Eliminate the prefix witness from e to expose its unary tail and continuation result. The left-unit result identifies the target with the tail, so unary membership transports to h. Conversely, use h itself as the unary tail and the continuation left unit. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE empty left iff unary`

Theorem 52.20 (Prefix to empty identifies the source)**.** *For the concrete unary history preorder, $h \leq \mathsf{e}$ holds exactly when h*hsamee*.*

Proof. Eliminate the prefix witness into a unary tail and a continuation into e. The empty-result inversion theorem forces the source history to be e. Conversely, after the history-sameness proof identifies the source with e, use the empty unary tail and the continuation right unit. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE empty target iff`

Theorem 52.21 (Prefix relation preserves the carrier)**.** *For the concrete unary history carrier, if h is a carrier history and h is related to k by appending a unary prefix tail, then k is also in the carrier.*

Proof. Unpack the prefix witness into its unary tail and continuation proof. Unary continuation closure sends the carrier witness for h, the unary witness for the tail, and the continuation proof to a carrier witness for k. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE preserves carrier`

Theorem 52.22 (Prefix target carrier reflects source carrier)**.** *For the concrete unary history carrier, if $h \leq k$ is witnessed by a unary prefix tail and k is in the carrier, then h is also in the carrier.*

Proof. Unpack the prefix witness into its unary tail and continuation proof. The target carrier is unary, so unary continuation left-factor closure recovers unary membership of the source history h. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE source carrier of target carrier`

Theorem 52.23 (Prefix carrier excludes zero target)**.** *If h is in the concrete preorder carrier, then a prefix comparison from h to* $\mathsf{e0}(k)$ *is impossible.*

Proof. The prefix relation preserves the concrete carrier from source to target. Thus $\mathsf{e0}(k)$ would be a unary history, contradicting the BEDC unary history zero-extension exclusion. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE carrier no e0 target`

Theorem 52.24 (Prefix carrier excludes zero source)**.** *If k is in the concrete preorder carrier, then a prefix comparison from* $\mathsf{e0}(h)$ *to k is impossible.*

Proof. The target carrier reflects through a concrete unary prefix witness to the source carrier. Thus $\mathsf{e0}(h)$ would be a unary history, contradicting the BEDC unary history zero-extension exclusion. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE carrier no e0 source`

Definition 52.25 (Preorder classifier specification)**.** *The classifier is the supplied carrier sameness, and the relation is stable under classifier replacement.*

Theorem 52.26 (Prefix right extension)**.** *If $h \leq k$ is witnessed by a unary continuation tail, and k continues by a unary tail to r, then $h \leq r$.*

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE right extension`

Proof. Unpack the first prefix witness as a unary tail from h to k. Compose this continuation with the continuation from k to r; associativity of continuation supplies the resulting path from h, and unary closure under append supplies the combined unary tail. □

Theorem 52.27 (Prefix preserves strict right extension)**.** *If $h \leq k$ is witnessed by a unary continuation tail and k is a strict unary prefix of r, then h is a strict unary prefix of r.*

Proof. Unpack the weak prefix tail from h to k and the nonempty strict tail from k to r. Appending the two tails gives a unary tail from h to r by unary append closure and continuation associativity. If the appended tail were e, append-empty inversion would force the strict right tail to be e, contradicting strictness. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE strict right extension`

Theorem 52.28 (Prefix transitivity)**.** *If $h \leq k$ and $k \leq r$ are witnessed by unary continuation tails, then $h \leq r$.*

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE trans`

Proof. Unpack the two prefix witnesses. Append the two unary tails, use unary closure under append for the combined tail, and use continuation associativity to identify the direct continuation from h to r. □

Theorem 52.29 (Strict unary prefix preserves right extension)**.** *If h is a strict unary prefix of k and $k \leq r$ is witnessed by a unary continuation tail, then h is a strict unary prefix of r.*

Proof. Unpack the nonempty strict tail from h to k and the weak prefix tail from k to r. Their append is unary and composes the two continuations from h to r. If the appended tail were e, append-empty inversion would force the original strict tail to be e, contradicting strictness. □

Lean4 ✓ `BEDC.Derived.PreorderUp.NatUnaryStrictPrefix right extension`

Theorem 52.30 (Prefix antisymmetry collapses to history sameness)**.** *If $h \leq k$ and $k \leq h$ are both witnessed by unary continuation tails, then* $\mathsf{hsame}(h, k)$.

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE antisymm hsame`

Proof. Unpack the two prefix witnesses. Substituting the first continuation equation into the second gives a round trip from h through the two tails back to h. Associativity rewrites this as equality between the empty right tail and the combined tail under the same left prefix, and left cancellation forces the forward tail to be empty. The first continuation equation then reduces to $\mathsf{hsame}(h, k)$. □

Theorem 52.31 (Concrete prefix order laws up to history-sameness)**.** *The concrete unary prefix relation* PreorderPrefixLE *is reflexive, transitive, and antisymmetric up to* hsame*: for every history h, $h \leq h$; for all histories h, k, r, $h \leq k$ and $k \leq r$ imply $h \leq r$; and for all histories h, k, $h \leq k$ and $k \leq h$ imply* $\mathsf{hsame}(h, k)$.

Proof. Reflexivity follows from hsame reflexivity together with PreorderPrefixLE_of_hsame. Transitivity is exactly PreorderPrefixLE_trans. Antisymmetry up to history-sameness is exactly PreorderPrefixLE_antisym □

Theorem 52.32 (Prefix left extension)**.** *If x continues by a unary tail to h, and $h \leq k$ is witnessed by a unary continuation tail, then $x \leq k$.*

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE left extension`

Proof. Unpack the prefix witness from h to k. Append the incoming unary tail to the prefix tail, use unary closure under append for the combined tail, and use continuation associativity to identify the resulting continuation from x to k. □

Theorem 52.33 (Prefix append tail)**.** *For every history h, appending a unary history t gives a concrete prefix comparison $h \leq \mathsf{append}(h, t)$.*

Proof. Use t itself as the prefix witness. Its unary membership is the supplied carrier for the tail, and the continuation component is the defining append-result equality. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append tail`

Theorem 52.34 (Prefix append tail backforces empty)**.** *If t is unary and $\mathsf{append}(h, t) \leq h$, then t is internally the same as e.*

Proof. The unary tail gives the forward prefix comparison $h \leq \mathsf{append}(h, t)$. Together with the assumed backward comparison, prefix antisymmetry gives $\mathsf{hsame}(h, \mathsf{append}(h, t))$. Rewriting the left endpoint as $\mathsf{append}(h, \mathsf{e})$ and cancelling the common left context forces $\mathsf{hsame}(t, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append tail backforces empty`

Theorem 52.35 (Prefix append tail backforces empty equivalence)**.** *If t is unary, then $\mathsf{append}(h, t) \leq h$ holds exactly when $t\mathsf{hsame}\mathsf{e}$.*

Proof. The forward direction is the backforces-empty implication for unary tails. For the reverse direction, $t\mathsf{hsame}\mathsf{e}$ identifies $\mathsf{append}(h, t)$ with the right-unit endpoint h; the prefix relation is then obtained from endpoint sameness. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append tail backforces empty iff`

Theorem 52.36 (Prefix left context monotonicity)**.** *If $h \leq k$ is witnessed by a unary continuation tail, then appending the same left context to both endpoints preserves the prefix comparison: $\mathsf{append}(x, h) \leq \mathsf{append}(x, k)$.*

Proof. Unpack the prefix witness as a unary tail and a continuation equation from h to k. Reuse the same unary tail after the common left context. Associativity of append identifies the resulting continuation endpoint with $\mathsf{append}(x, k)$. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append left context`

Theorem 52.37 (Prefix right context monotonicity)**.** *If x is unary and $h \leq k$ is witnessed by a unary continuation tail, then appending x to both endpoints preserves the prefix comparison: $\mathsf{append}(h, x) \leq \mathsf{append}(k, x)$.*

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append right context`

Proof. Unpack the prefix witness as a unary tail and a continuation equation from h to k. Reuse the same tail. Associativity moves the right context next to the tail, unary commutativity swaps the two unary histories, and associativity moves the tail back to the continuation endpoint. □

Theorem 52.38 (Prefix two-sided context monotonicity)**.** *If r is unary and $h \leq k$ is witnessed by a unary continuation tail, then placing the same left context outside both right-contexted endpoints preserves the prefix comparison:* $\mathsf{append}(\ell, \mathsf{append}(h, r)) \leq \mathsf{append}(\ell, \mathsf{append}(k, r))$.

Proof. First apply right-context monotonicity using the unary carrier of r. Then apply left-context monotonicity to the resulting comparison. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append both context`

Theorem 52.39 (Prefix right context cancellation)**.** *If x is unary and* $\mathsf{append}(h, x) \leq \mathsf{append}(k, x)$*, then $h \leq k$.*

Proof. Unpack the prefix witness as a unary tail and a continuation equation. Append associativity brings the right context next to that tail, unary commutativity exchanges the two unary histories, and associativity restores a common right context. Right cancellation removes the shared x, leaving the continuation witness from h to k. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE cancel right context`

Theorem 52.40 (Prefix left context cancellation)**.** *If appending the same left context to h and k makes* $\mathsf{append}(x, h) \leq \mathsf{append}(x, k)$*, then $h \leq k$.*

Proof. Unpack the prefix witness as a unary tail and a continuation equation from $\mathsf{append}(x, h)$ to $\mathsf{append}(x, k)$. Continuation associativity rewrites the right side as a common left append by x; left cancellation removes that context and gives the continuation equation from h to k with the same unary tail. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE cancel left context`

Theorem 52.41 (Prefix two-sided context cancellation)**.** *If r is unary and the two endpoints become prefix-comparable after adding a common left context outside a common right context,*

$$\mathsf{append}(\ell, \mathsf{append}(h, r)) \leq \mathsf{append}(\ell, \mathsf{append}(k, r)),$$

then $h \leq k$.

Proof. First cancel the common left context, giving $\mathsf{append}(h, r) \leq \mathsf{append}(k, r)$. Then use right-context cancellation with the unary witness for r, leaving the prefix comparison from h to k. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE cancel both context`

Theorem 52.42 (Prefix two-sided context equivalence)**.** *If r is unary, then placing the same left context outside both right-contexted endpoints preserves and reflects the concrete prefix comparison:*

$$\mathsf{append}(\ell, \mathsf{append}(h, r)) \leq \mathsf{append}(\ell, \mathsf{append}(k, r)) \quad \Longleftrightarrow \quad h \leq k.$$

Proof. The forward direction first cancels the common left context, then cancels the common unary right context. The reverse direction is exactly two-sided context monotonicity. □

Lean4 ✓ `BEDC.Derived.PreorderUp.PreorderPrefixLE append both context iff`

Definition 52.43 (Concrete prefix stability certificate)**.** *The concrete prefix stability certificate for* PreorderPrefixLE *under* PreorderClassifierSpec *consists of:*

1. *classifier reflexivity*

2. *classifier transitivity*
3. *prefix preorder reflexivity*
4. *prefix preorder transitivity*
5. *prefix relation congruence in both arguments*

Lean4 ✓ `BEDC.Derived.PreorderUp.preorder prefix stability certificate fields`

Definition 52.44 (Abstract preorder stability certificate)**.** *The abstract preorder stability certificate for an arbitrary relation leC consists of:*

1. *classifier reflexivity*
2. *classifier transitivity*
3. *preorder reflexivity*
4. *preorder transitivity*
5. *relation congruence in both arguments*

Lean4 ✓ `BEDC.Derived.PreorderUp.preorder stability certificate fields`

Definition 52.45 (Preorder ledger policy)**.** *Record the compared carrier witnesses and the relation derivation.*

Definition 52.46 ($\mathsf{Preorder}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Preorder}^{\uparrow}$*.*

Theorem 52.47 (Concrete unary prefix $\mathsf{Preorder}^{\uparrow}$ naming certificate)**.** *The tuple of Definition 52.3 supplies the source, pattern, classifier, stability, ledger, and exactness rows for a* NameCert *licensing* $\mathsf{Preorder}^{\uparrow}$ *over* PreorderCarrier = UnaryHistory*, classifier* PreorderCarrierClassifierSpec(PreorderCarrier)*, and relation* PreorderPrefixLE*.*

Proof. The source, pattern, and classifier rows are the concrete unary-prefix rows fixed in Definition 52.3. The carrier is inhabited by e, and carrier membership transports along hsame by the carrier clauses recorded in Definition 52.4.

The stability row is Definition 52.43. Its reflexivity, transitivity, and endpoint-congruence components are exactly the remaining law clauses collected in Definition 52.4.

The concrete relation witnesses are Definition 52.17 for comparisons induced by history-sameness and Definition 52.28 for composing two prefix comparisons. The ledger row is Definition 52.45; together these rows repack the concrete unary-prefix data as the public NameCert for $\mathsf{Preorder}^{\uparrow}$. □

Corollary 52.48 ($\mathsf{Preorder}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{Preorder}^{\uparrow}$ *is licensed by relational fields, not as a primitive order.*

Lean4 ✓ `BEDC.Derived.PreorderUp.preorder carrier semantic name certificate`

Definition 52.49 (Opposite preorder certificate data)**.** *Let a* $\mathsf{Preorder}^{\uparrow}$ *certificate be presented by a carrier predicate* $\mathcal{C}$*, classifier* $\sim_C$*, and relation* $\leq_C$*. Define the opposite relation by*

$$a \leq_C^{\mathrm{op}} b := b \leq_C a.$$

The opposite preorder data C^{op} *have the same carrier predicate* $\mathcal{C}$*, the same classifier* $\sim_C$*, and relation* $\leq_C^{\mathrm{op}}$*.*

Theorem 52.50 (Opposite preorder certificate)**.** *If $(\mathcal{C}, \sim_C, \leq_C)$ satisfies the* $\mathsf{Preorder}^{\uparrow}$ *certificate obligations of Definition 52.46, then the data of Definition 52.49 also satisfy the* $\mathsf{Preorder}^{\uparrow}$ *certificate obligations. Thus the same carrier and classifier carry a preorder certificate with relation $\leq_C^{\mathrm{op}}$.*

Proof. Unpack the preorder obligations through Definition 52.46. The carrier row of Definition 52.2 is unchanged, and the carrier-and-classifier core is the same $\mathsf{NameCert}$ data. Hence classifier reflexivity, symmetry, transitivity, and carrier transport are copied from Definition 9.7. The source row of Definition 52.15 is the same carrier and classifier equipped with the reversed relation symbol.

The reflexivity field for the opposite relation is immediate. For every carried endpoint a,

$$a \leq_C^{\mathrm{op}} a$$

unfolds to $a \leq_C a$, which is the original preorder reflexivity field.

For transitivity, assume $a \leq_C^{\mathrm{op}} b$ and $b \leq_C^{\mathrm{op}} c$. By definition these are $b \leq_C a$ and $c \leq_C b$. Applying the original transitivity field of Definition 52.44 to the composable pair $c \leq_C b$ and $b \leq_C a$ gives $c \leq_C a$, which is exactly $a \leq_C^{\mathrm{op}} c$.

Relation congruence is inherited with the two endpoints exchanged. Given $a \sim_C a'$, $b \sim_C b'$, and $a \leq_C^{\mathrm{op}} b$, unfold the last witness to $b \leq_C a$. The original relation-congruence field, applied with endpoint comparisons $b \sim_C b'$ and $a \sim_C a'$, gives

$$b' \leq_C a'.$$

This is precisely $a' \leq_C^{\mathrm{op}} b'$.

The pattern row of Definition 52.16 remains binary comparison expression formation over carried endpoints: a node $a \leq_C^{\mathrm{op}} b$ is read as the original comparison node $b \leq_C a$ with endpoint order exchanged. The classifier row of Definition 52.25 is unchanged because endpoint classification is still $\sim_C$. The ledger row of Definition 52.45 records the same two carrier witnesses and the same relation derivation, with source and target slots exchanged for the displayed comparison.

These source, pattern, classifier, stability, and ledger rows are exactly the five fields of the paper naming-certificate template in Definition 9.6, specialized to the preorder obligations of Definition 52.46. Therefore C^{op} carries a $\mathsf{Preorder}^{\uparrow}$ certificate. □

Theorem 52.51 (Singleton empty $\mathsf{Preorder}^{\uparrow}$ public certificate boundary)**.** *For the concrete singleton package $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}}^{\mathrm{sg}})$ of Definition 52.7, the public comparison readback of Definition 52.11, the single-comparison exactness theorem Definition 52.12, and the certificate rows of Definition 52.13 determine a concrete* BHist*-anchored* $\mathsf{Preorder}^{\uparrow}$ *public surface.*

The public surface has carrier A_{e}, classifier ρ_{e}, and relation $\leq_{\mathsf{e}}^{\mathrm{sg}}$. For every pair of histories h, k, public comparison witnesses read back exactly as

$$h \leq_{\mathsf{e}}^{\mathrm{sg}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

The surface introduces no abstract carrier, abstract classifier, order parameter, quotient, host equality, or standard bridge.

Lean4 ✓ `BEDC.Derived.PreorderUp.preorder singleton empty public certificate boundary`

Proof. Unfold the singleton package of Definition 52.7. Its source is the closed BHist carrier A_{e}, its classifier is ρ_{e}, and its guarded comparison relation is $\leq_{\mathsf{e}}^{\mathrm{sg}}$.

The public readback row is Definition 52.11. Apply Definition 52.12 to obtain the displayed equivalence between one guarded public comparison and the two endpoint-carrier witnesses.

The preorder laws needed for the surface are the singleton empty preorder laws: reflexivity on carried histories, transitivity of guarded comparisons, and endpoint congruence for ρ_{e}. The guarded prefix exactness theorem Definition 52.9 supplies the classifier equivalence relating $\rho_{\mathsf{e}}(h, k)$ to the two guarded comparison directions.

Finally, repack the source, pattern, classifier, stability, and ledger rows through Definition 52.13. These rows are all anchored in BHist and the singleton endpoint carrier, so the resulting boundary is the concrete public surface described in the statement and contains none of the excluded abstract parameters. □

Definition 52.52 (Preorder terminal endpoint-order surface)**.** *For histories $h, k : \mathsf{BHist}$, define the terminal endpoint order over the singleton carrier A_{e} by*

$$h \preceq_{\mathsf{e}} k := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

The public surface is the concrete tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \preceq_{\mathsf{e}})$, where ρ_{e} is the restricted singleton classifier of Definition 52.7. Thus both the carrier and the comparison relation are pinned to BHist endpoints, and the relation records exactly the two endpoint carrier witnesses.

This concrete endpoint order is the relation row that chapter 53 extends by adding antisymmetry, and that chapter 1378 later reads as the comparison substrate for arithmetic-facing order packets.

Theorem 52.53 (Preorder terminal endpoint-order laws)**.** *For the endpoint order of Definition 52.52, the following laws hold for histories $h, k, \ell : \mathsf{BHist}$.*

1. *If $A_{\mathsf{e}}(h)$, then $h \preceq_{\mathsf{e}} h$.*
2. *If $h \preceq_{\mathsf{e}} k$ and $k \preceq_{\mathsf{e}} \ell$, then $h \preceq_{\mathsf{e}} \ell$.*
3. *If $\rho_{\mathsf{e}}(h, h')$, $\rho_{\mathsf{e}}(k, k')$, and $h \preceq_{\mathsf{e}} k$, then $h' \preceq_{\mathsf{e}} k'$.*
4. *For all endpoints,*
$$h \preceq_{\mathsf{e}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. Unfold the definition of $\preceq_{\mathsf{e}}$. Reflexivity on a carried endpoint uses the same witness twice, so $A_{\mathsf{e}}(h)$ gives $A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(h)$.

For transitivity, project $A_{\mathsf{e}}(h)$ from the first comparison and $A_{\mathsf{e}}(\ell)$ from the second comparison. Pairing those witnesses gives $h \preceq_{\mathsf{e}} \ell$ after unfolding the terminal relation.

For classifier congruence, unfold ρ_{e} at both endpoint pairs. The classifier witnesses contain $A_{\mathsf{e}}(h')$ and $A_{\mathsf{e}}(k')$ directly, while the given comparison supplies the source endpoint witnesses. Pair the target carrier witnesses to obtain $h' \preceq_{\mathsf{e}} k'$.

The exact readback is definitional: the forward direction projects the two fields of the relation, and the reverse direction repacks the two endpoint carrier witnesses. The singleton endpoint transport already recorded for A_{e} is the only transport needed by the classifier row. □

Theorem 52.54 (Preorder terminal endpoint-order public certificate)**.** *The tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \preceq_{\mathsf{e}})$ determines a concrete $\mathsf{Preorder}^{\uparrow}$ public surface over BHist. Its source is the singleton endpoint carrier A_{e}, its carrier is exactly the histories accepted by A_{e}, its classifier is ρ_{e}, its relation is $\preceq_{\mathsf{e}}$, its ledger records the two endpoint carrier witnesses, and its exactness row is*

$$h \preceq_{\mathsf{e}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

The resulting status is a concrete terminal endpoint-order surface for $\mathsf{Preorder}^{\uparrow}$ with no abstract order parameter and no quotient layer.

Proof. The source, carrier, and classifier rows are the singleton rows fixed in Definition 52.7. The relation row is Definition 52.52, so every comparison witness is already a pair of endpoint carrier witnesses.

Apply Definition 52.53. Its reflexivity, transitivity, classifier-congruence, and exact-readback clauses supply the order-law and exactness rows required by the preorder public surface.

The ledger is forced by the definition of $\preceq_{\mathsf{e}}$: a public comparison stores no data beyond $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. Repacking these rows into the roadmap source, carrier, classifier, relation, ledger, exactness, and status fields gives the stated concrete $\mathsf{Preorder}^{\uparrow}$ surface. □

Definition 52.55 (Preorder terminal opposite endpoint order)**.** *For histories $h, k : \mathsf{BHist}$, define the endpoint-swapped relation of Definition 52.52 by*

$$h \preceq_{\mathsf{e}}^{\mathrm{op}} k := k \preceq_{\mathsf{e}} h.$$

This is the opposite relation of Definition 52.49 specialized to the concrete terminal endpoint-order surface $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \preceq_{\mathsf{e}})$.

Theorem 52.56 (Preorder terminal endpoint-order opposite self-duality)**.** *For histories h, k : BHist on the terminal endpoint-order surface of Definition 52.52, the opposite relation has the same endpoint readback:*

$$h \preceq_{\mathsf{e}}^{\mathrm{op}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Consequently,

$$h \preceq_{\mathsf{e}}^{\mathrm{op}} k \iff h \preceq_{\mathsf{e}} k.$$

Thus the concrete terminal endpoint-order surface is its own opposite relation on carried endpoints.

Proof. Unfold Definition 52.55. A witness of $h \preceq_{\mathsf{e}}^{\mathrm{op}} k$ is exactly a witness of $k \preceq_{\mathsf{e}} h$.

Apply the exact-readback clause of Definition 52.53 to the ordered pair (k, h). It gives

$$k \preceq_{\mathsf{e}} h \iff A_{\mathsf{e}}(k) \wedge A_{\mathsf{e}}(h).$$

The conjunction on the right is symmetric: from $A_{\mathsf{e}}(k) \wedge A_{\mathsf{e}}(h)$ project the second and first witnesses to obtain $A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k)$, and the reverse implication is obtained by the same projection swap. Therefore

$$h \preceq_{\mathsf{e}}^{\mathrm{op}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Apply the same exact-readback clause of Definition 52.53 to (h, k):

$$h \preceq_{\mathsf{e}} k \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Both $h \preceq_{\mathsf{e}}^{\mathrm{op}} k$ and $h \preceq_{\mathsf{e}} k$ are therefore equivalent to the same pair of endpoint carrier witnesses, which yields the claimed equivalence between the opposite relation and the original terminal endpoint order. □

Closure status for $\mathsf{NameCert}_{\mathsf{Preorder}^{\uparrow}}$**.**

Constructive story: A preorder in the concrete BEDC instance starts from the unary history spine: e and its repeated b1-extensions are admitted, while any b0 branch is outside the carrier. The comparison says that one unary history is below another when a unary continuation tail extends the first to the second, and the classifier on carried endpoints is the carrier-aware hsame relation rather than a quotient of comparable histories. Thus the source-domain is the same unary carrier, the pattern is reflexive and transitive prefix comparison with congruence under endpoint replacement, and the ledger records the continuation tail witnessing each displayed comparison together with the raw unary endpoints. The five name-certificate obligations come from the unary spine itself: e inhabits the carrier, history-sameness gives reflexivity, symmetry, and transitivity for the classifier, and unary transport keeps any history-same representative inside the carrier.

A history h lies in the signature gap of a Preorder$^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and comparison ledger, and token introduction names that answer as p; unfolded, h is unary and the recorded prefix continuations are exactly the order evidence encoded by p. The paper bridge reads this surface as the finite unary prefix preorder, where a prefix tail is the ordinary witness that one finite unary word extends another, then reads back to the same continuation data. The chapter registers this continuation-order construction under the name Preorder$^{\uparrow}$; the relation is generated by BEDC append and hsame, with no arbitrary preorder carrier, totality field, antisymmetry quotient, or order topology introduced.

Theory closure: TheoryGate(NameCert$_{\text{Preorder}\uparrow}$, maturePackageClosed)

Scope closed: The public unary-prefix preorder surface binds the comparison constructors, public relation readback, endpoint exactness, carrier/classifier rows, and ledger rows for PreorderPrefixLE over UnaryHistory.

Formal verification: FormalStatus(NameCert$_{\text{Preorder}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PreorderUp.preorder\_carrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: General order theory: arbitrary preorders, quotient completions, totality, antisymmetry, lattices, order topology, or standard-model bridge closure.

Upgrade path: Lean bridge checking for the standard boundary bridge remains the next verification step; the paper mature closure is the recorded theory grade.

Chapter 53

A Concrete Naming Certificate for $\mathsf{POSet}^{\uparrow}$

The partially ordered set interface $\mathsf{POSet}^{\uparrow}$ extends chapter 52 with the antisymmetry field: if $a \leq b$ and $b \leq a$ then a and b are classified the same by the carrier. The integer order from chapter 31 is the canonical concrete instance.

Remark 53.1 ($\mathsf{POSet}^{\uparrow}$ interface scope)**.** *This chapter records the* $\mathsf{POSet}^{\uparrow}$ *licensing target. The concrete prefix antisymmetry theorem is linked to the unary preorder classifier, and the certificate fields are specified below.*

53.1 Carrier definition

Definition 53.2 (POSet carrier)**.** *Use a preorder setup extended with antisymmetry under the carrier classifier.*

Theorem 53.3 (POSetUp BHist carrier transport obligation)**.** *Let h and k be carried* BHist *endpoints in the concrete unary prefix preorder carrier. If $h\mathsf{hsame}k$, the preorder carrier transport row is present, and the* $\mathsf{POSet}^{\uparrow}$ *antisymmetry row is carried with that preorder package, then transporting the endpoint from h to k leaves it in the* $\mathsf{POSet}^{\uparrow}$ *carrier. The transported package keeps the preorder carrier row and repacks the same antisymmetry field; it does not introduce a quotient carrier or a representative class.*

Proof. Apply the carrier transport row of Definition 52.31. It transports the displayed unary BHist carrier along the classifier row $h\mathsf{hsame}k$, so the preorder carrier component remains available at the target endpoint.

The $\mathsf{POSet}^{\uparrow}$ carrier of Definition 53.2 is precisely that preorder carrier together with an antisymmetry row under the same classifier. Repack the transported preorder carrier with the displayed antisymmetry field. The endpoint remains a carried BHist endpoint, and no host quotient object is used. □

Theorem 53.4 (POSetUp classifier exactness obligation)**.** *For carried unary* BHist *histories h and k, the* $\mathsf{POSet}^{\uparrow}$ *classifier row between them is exact for the bidirectional prefix comparison: the classifier holds exactly when the classifier row is present and the two directed preorder comparison rows $h \leq_{\mathsf{pref}} k$ and $k \leq_{\mathsf{pref}} h$ collapse by the antisymmetry field. The reverse comparison rows are obtained from preorder reflexivity and transport along the classifier.*

Proof. For the forward comparison-to-classifier direction, assume the two directed prefix witnesses. The antisymmetry field of Definition 53.33 collapses them to $\mathsf{hsame}(h, k)$, which is the carrier classifier named by Definition 53.2.

For the classifier-to-comparison direction, assume $\mathsf{hsame}(h, k)$. The preorder package of Definition 52.31 supplies reflexivity at each carried endpoint and transports comparison endpoints along the classifier. Transport the reflexive comparison at h to obtain $h \leq_{\mathsf{pref}} k$, and transport the reflexive comparison at k to obtain $k \leq_{\mathsf{pref}} h$.

These two constructions identify the classifier boundary with the bidirectional comparison boundary, with antisymmetry providing the collapse and the preorder rows providing the transported reflexive comparisons. □

Theorem 53.5 (POSetUp antisymmetry ledger obligation)**.** *An antisymmetry use in the concrete* $\mathsf{POSet}^{\uparrow}$ *ledger records exactly the two comparison witnesses* $h \leq_{\mathsf{pref}} k$ *and* $k \leq_{\mathsf{pref}} h$*, their carried* BHist *endpoints, and the resulting* $\mathsf{hsame}(h, k)$ *classifier row. The ledger contains no host quotient, chosen class representative, or external equality witness.*

Proof. Project the two comparison witnesses from the preorder ledger supplied by Definition 52.31. Their endpoints are the same carried unary BHist histories used by the $\mathsf{POSet}^{\uparrow}$ carrier.

Apply the antisymmetry row Definition 53.33 to the projected comparison pair. It produces the classifier row $\mathsf{hsame}(h, k)$ and no additional object.

Repack the ledger entry as the ordered pair of comparison witnesses, the two displayed endpoints, and the resulting classifier row. This is exactly the ledger policy described by the $\mathsf{POSet}^{\uparrow}$ certificate surface. □

Theorem 53.6 (POSet root carrier obligation)**.** *The root* $\mathsf{POSet}^{\uparrow}$ *carrier obligation is a* $\mathsf{Preorder}^{\uparrow}$ *carrier over* BHist*, equipped with the same prefix relation and with an antisymmetry row under the carrier classifier. No quotient carrier is added: the carried endpoints remain displayed* BHist *histories, and the only collapse allowed by the partial-order field is the classifier conclusion* $\mathsf{hsame}(h, k)$*.*

Proof. The carrier clause is exactly Definition 53.2: start with the preorder carrier and add the antisymmetry field as an extra row, not as a new endpoint space.

For the concrete root instance, Definition 52.31 supplies the BHist carrier, the prefix preorder relation, and the hsame classifier boundary. The partial-order coordinate is supplied by Definition 53.33, which turns opposing prefix comparisons into $\mathsf{hsame}(h, k)$.

Thus the carrier remains the inherited preorder carrier, while the antisymmetry data is an obligation row attached to that carrier. □

Theorem 53.7 (POSet root classifier obligation)**.** *The root* $\mathsf{POSet}^{\uparrow}$ *classifier is the inherited* hsame*-based classifier on carried* BHist *endpoints, and it is stable under the preorder relation: classifier-equivalent endpoints may be read through the same prefix comparison rows without changing the carried* $\mathsf{POSet}^{\uparrow}$ *object.*

Proof. By Definition 53.2, a $\mathsf{POSet}^{\uparrow}$ object does not replace the preorder classifier. It extends the preorder package by demanding antisymmetry under that classifier.

Apply Definition 52.31 to obtain the inherited hsame classifier together with the prefix-order stability rows. Those rows are the classifier transport surface for comparison endpoints.

The additional $\mathsf{POSet}^{\uparrow}$ row from Definition 53.33 is compatible with the same classifier, because its conclusion is again $\mathsf{hsame}(h, k)$. Hence classifier transport remains the inherited preorder transport. □

Theorem 53.8 (POSet root exactness obligation)**.** *For carried unary prefix endpoints h and k, the opposing comparison rows $h \leq_{\mathsf{pref}} k$ and $k \leq_{\mathsf{pref}} h$ are exactly the* $\mathsf{POSet}^{\uparrow}$ *classifier row on those endpoints: the forward direction is antisymmetry, and the reverse direction is the inherited reflexive prefix row transported along* hsame*.*

Proof. For the forward direction, assume the two comparison rows. The concrete antisymmetry theorem Definition 53.33 collapses the opposing prefix witnesses to $\mathsf{hsame}(h, k)$, which is the classifier row specified by Definition 53.2.

For the reverse direction, assume the classifier row $\mathsf{hsame}(h, k)$. The preorder law package of Definition 52.31 provides reflexive prefix comparisons on carried endpoints and transports comparison endpoints along hsame.

Transporting the reflexive comparison on h gives $h \leq_{\mathsf{pref}} k$, and transporting the reflexive comparison on k gives $k \leq_{\mathsf{pref}} h$. The two directions identify the exact comparison boundary of the root $\mathsf{POSet}^{\uparrow}$ row. □

53.2 Concrete unary prefix POSet instance

Definition 53.9 (Concrete unary prefix POSet instance)**.** *The concrete unary prefix POSet instance is the following ten-field tuple.*

1. *The traditional target is the finite unary-word prefix partial order.*
2. *The BEDC source is* BHist *restricted to unary histories.*
3. *The carrier is* PreorderCarrier = UnaryHistory*.*
4. *The classifier is the carrier-aware specification* PreorderCarrierClassifierSpec(PreorderCarrier)*, whose equality content is* hsame*.*
5. *The relation is* PreorderPrefixLE*.*
6. *Stability consists of classifier reflexivity, classifier transitivity, relation reflexivity, relation transitivity, relation congruence, and antisymmetry under the classifier.*
7. *The ledger records both comparison-tail witnesses used for $h \leq k$ and $k \leq h$.*
8. *The core theorem list is the concrete prefix stability certificate together with prefix antisymmetry.*
9. *Exactness is the bidirectional prefix collapse: two opposing prefix comparisons collapse exactly to the classifier relation.*
10. *The standard bridge identifies* PreorderPrefixLE *with the initial-segment order on finite unary words, and the status is* ProofTarget*.*

Theorem 53.10 (Concrete unary prefix POSet laws)**.** *The tuple in Definition 53.9 satisfies the* $\mathsf{POSet}^{\uparrow}$ *carrier field, the preorder fields for* PreorderPrefixLE*, classifier congruence for* hsame*, and antisymmetry under the carrier classifier.*

Proof. The carrier and classifier fields are the concrete preorder fields: the carrier is PreorderCarrier = UnaryHistory, and the classifier is PreorderCarrierClassifierSpec(PreorderCarrier).

The preorder and classifier-congruence fields are supplied by the concrete prefix stability certificate in Definition 52.43. Equivalently, Definition 52.31 records the reflexive and transitive prefix laws together with the history-sameness classifier boundary.

It remains to supply the partial-order field. Apply Definition 53.33: opposing PreorderPrefixLE witnesses provide the two comparison tails, and the theorem collapses them to $\mathsf{hsame}(h, k)$, which is exactly antisymmetry under the carrier classifier. □

Definition 53.11 (Concrete singleton empty POSet instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define the restricted classifier by*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Define the carrier-guarded restricted order by

$$h \leq_{\mathsf{e}} k := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{PreorderPrefixLE}(h, k).$$

The concrete POSet package is the fixed BHist *tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}})$.*

Theorem 53.12 (Singleton empty POSet laws)**.** *For the concrete package of Definition 53.11, the carrier is inhabited by* e*, carrier membership transports along* hsame*, and $\leq_{\mathsf{e}}$ has preorder reflexivity, transitivity, and endpoint congruence inherited from the singleton empty preorder laws. The package also satisfies antisymmetry under ρ_{e}. For all histories h, k :* BHist*, the classifier is exact:*

$$\rho_{\mathsf{e}}(h, k) \Longleftrightarrow (h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h).$$

Proof. The carrier and preorder fields are those of Definition 52.8. Inhabitance is witnessed by e, carrier transport is transport along hsame, and the reflexive, transitive, and endpoint-congruent behavior of $\leq_{\mathsf{e}}$ is the restricted PreorderPrefixLE package on the singleton source.

For antisymmetry, suppose h and k are carried and that $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$. Project the opposing PreorderPrefixLE witnesses from the guarded comparisons, so PreorderPrefixLE_antisymm_hsame gives $\mathsf{hsame}(h, k)$. Pair this with the carrier witnesses for h and k to obtain $\rho_{\mathsf{e}}(h, k)$.

For exactness, first assume $\rho_{\mathsf{e}}(h, k)$. The classifier supplies the two carrier witnesses and the endpoint witness $\mathsf{hsame}(h, k)$. Apply PreorderPrefixLE_of_hsame to this witness and to its symmetric orientation, then repack the two carrier witnesses with the resulting prefix comparisons to obtain $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$.

Conversely, assume the two guarded comparisons. Project the carrier witnesses and the opposing prefix witnesses from them. Apply PreorderPrefixLE_antisymm_hsame to the prefix witnesses to recover $\mathsf{hsame}(h, k)$, and recombine this endpoint witness with the carrier witnesses to get $\rho_{\mathsf{e}}(h, k)$. □

Theorem 53.13 (Singleton empty POSet terminal interval collapse)**.** *For the concrete package of Definition 53.11, let h, k, r :* BHist *be carried by A_{e}. If the displayed terminal chain supplies $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} r$, then all three endpoints collapse inside the singleton interval:*

$$\rho_{\mathsf{e}}(h, k) \wedge \rho_{\mathsf{e}}(k, r) \wedge \rho_{\mathsf{e}}(h, r) \wedge h \leq_{\mathsf{e}} r \wedge r \leq_{\mathsf{e}} h.$$

The ten-field fit is: traditional target one-point partial order interval collapse; BEDC source singleton BHist*; carrier A_{e}; classifier ρ_{e}; relation $\leq_{\mathsf{e}}$; stability from singleton transport, preorder transitivity, and POSet antisymmetry; ledger the two chain comparisons plus carried endpoints; core theorem this block; exactness terminal classifier collapse; standard bridge one-point poset; status* ProofTarget*.*

Proof. Unfold the singleton carrier A_{e}. Each carried endpoint transports through the empty history e, so the endpoint classifier witnesses $\rho_{\mathsf{e}}(h,k)$, $\rho_{\mathsf{e}}(k,r)$, and $\rho_{\mathsf{e}}(h,r)$ are obtained by pairing the two relevant carrier witnesses with the corresponding hsame endpoint comparison.

The comparison $h \leq_{\mathsf{e}} r$ is inherited from preorder transitivity for the restricted prefix relation: compose the displayed witnesses $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} r$.

For the reverse comparison, the same singleton endpoint comparison identifies r with h inside A_{e}. Apply the exactness direction of Definition 53.12 to the classifier witness $\rho_{\mathsf{e}}(h,r)$, then read off $r \leq_{\mathsf{e}} h$ from the opposing comparison. The five displayed conjuncts therefore repack the terminal interval collapse. □

Theorem 53.14 (POSet singleton prefix-chain collapse)**.** *For the concrete package of Definition 53.11, let $h_0, \ldots, h_n : \mathsf{BHist}$ be carried by A_{e} and suppose the finite guarded chain supplies $h_j \leq_{\mathsf{e}} h_{j+1}$ for each adjacent pair. Then every adjacent comparison, every endpoint comparison obtained by transitivity, and every reverse endpoint comparison needed by the singleton interval collapses to the restricted classifier ρ_{e}. Equivalently, the finite chain is recorded entirely by* BHist *endpoints,* hsame *transport, and guarded prefix witnesses, with no host order or quotient row.*

Proof. Induct over the displayed finite guarded chain. The base case is the one-point chain: the carried endpoint lies in A_{e}, so Definition 53.12 supplies reflexivity and the classifier exactness row for the endpoint against itself.

For the step, assume the prefix chain has already collapsed through the current endpoint and append one guarded comparison. Transitivity from Definition 53.12 gives the forward endpoint comparison across the longer chain, while its antisymmetry and exactness rows collapse the adjacent comparison to ρ_{e}.

For any three consecutive endpoints, apply Definition 53.13. This supplies the reverse endpoint comparison required by the singleton interval and identifies each endpoint pair by the same restricted classifier.

Repack the induction data as a finite ledger of carried BHist endpoints, hsame transports, and guarded prefix witnesses. Since each collapse row comes from ρ_{e} and the singleton laws, no additional order structure is used. □

This concrete BHist tuple is the $\mathsf{POSet}^{\uparrow}$ schema instance used by machine checking.

Theorem 53.15 (Concrete singleton empty $\mathsf{POSet}^{\uparrow}$ naming certificate)**.** *The tuple $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}})$ of Definition 53.11 supplies the source, pattern, classifier, stability, ledger, and exactness rows for a concrete* $\mathsf{NameCert}$ *licensing* $\mathsf{POSet}^{\uparrow}$ *over* BHist*. On carried histories, the exactness row is*

$$\rho_{\mathsf{e}}(h,k) \Longleftrightarrow (h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h).$$

Proof. The source row is the singleton history source A_{e} anchored at e, and the classifier row is the restricted carrier classifier ρ_{e} from Definition 53.11. The pattern row is the restricted prefix comparison $h \leq_{\mathsf{e}} k$ on carried histories.

The preorder, antisymmetry, and stability rows are exactly the concrete law package of Definition 53.12. Its preorder fields give reflexivity, transitivity, and endpoint congruence for the restricted prefix relation, while its antisymmetry field collapses opposing comparisons to ρ_{e}.

The ledger records the two comparison witnesses used by antisymmetry, $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$, together with the carried endpoints. The displayed equivalence is the exactness row supplied by the same law package, so the concrete tuple repacks as the public $\mathsf{NameCert}$ for $\mathsf{POSet}^{\uparrow}$. □

Theorem 53.16 (Singleton empty $\mathsf{POSet}^{\uparrow}$ schema instantiation)**.** *The tuple* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\mathsf{e}})$ *of Definition 53.11 instantiates the* $\mathsf{POSet}^{\uparrow}$ *schema over the fixed BEDC history carrier* BHist*. Its carrier row is* A_{e}*, its classifier row is* ρ_{e}*, and its relation row is* $\leq_{\mathsf{e}}$*. Its preorder reflexivity, preorder transitivity, relation congruence, antisymmetry, and exactness rows are exactly the rows certified in Definition 53.12. Its concrete* $\mathsf{NameCert}$ *packaging is the certificate of Definition 53.15.*

Proof. Unfold Definition 53.11. The carrier is the singleton history source A_{e} inside BHist, the classifier is the restricted history classifier ρ_{e}, and the relation is the carrier-guarded restricted prefix comparison $\leq_{\mathsf{e}}$.

The law rows are supplied without changing carrier. Apply Definition 53.12: it gives the singleton inhabitance and carrier transport, the preorder reflexivity and transitivity rows for $\leq_{\mathsf{e}}$, endpoint congruence for the guarded relation, antisymmetry under ρ_{e}, and the displayed exactness between ρ_{e} and the two opposing guarded comparisons.

Finally apply Definition 53.15. That theorem packages the same source, pattern, classifier, stability, ledger, and exactness fields as a concrete $\mathsf{NameCert}$ licensing $\mathsf{POSet}^{\uparrow}$ over BHist. Therefore the schema instance is pinned to the concrete history carrier and not obtained by transporting an abstract parameter schema. □

Theorem 53.17 (POSet singleton carrier-classifier obligation)**.** *The concrete singleton* $\mathsf{POSet}^{\uparrow}$ *package of Definition 53.11 has carrier* A_{e} *and classifier* ρ_{e} *over the fixed BEDC carrier* BHist*. Carrier transport is exactly transport by* hsame *inside the singleton source, and the package exports no quotient carrier, host equality, or abstract replacement carrier.*

Proof. Unfold Definition 53.11. Its displayed carrier coordinate is the singleton history source A_{e}, and its classifier coordinate is the restricted classifier ρ_{e} on histories of BHist.

Apply Definition 53.12. The law package gives carrier inhabitance at e and carrier transport along hsame, with no change of carrier. The certificate theorem Definition 53.15 then repacks those rows as the concrete $\mathsf{POSet}^{\uparrow}$ naming certificate. □

Theorem 53.18 (POSet singleton antisymmetry exactness obligation)**.** *For the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *package of Definition 53.11, opposing guarded comparisons* $h \leq_{\mathsf{e}} k$ *and* $k \leq_{\mathsf{e}} h$ *are exactly the classifier row* $\rho_{\mathsf{e}}(h, k)$*. Thus antisymmetry is exposed as an exactness predicate on the concrete* BHist *rows, not as an external host-order law.*

Proof. Unfold the guarded relation and classifier of Definition 53.11. The relation $\leq_{\mathsf{e}}$ is the carrier-guarded prefix comparison, and ρ_{e} records the two singleton carrier endpoints together with $\mathsf{hsame}(h, k)$.

Apply Definition 53.12. Its antisymmetry row collapses the two guarded comparisons to the classifier, and its exactness row reads the classifier back as the same opposing comparison pair. The certificate theorem Definition 53.15 exposes this row as the singleton $\mathsf{POSet}^{\uparrow}$ exactness obligation. □

Theorem 53.19 (POSet singleton ledger obligation)**.** *The ledger for the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *certificate of Definition 53.15 consists exactly of the two guarded comparison witnesses* $h \leq_{\mathsf{e}} k$ *and* $k \leq_{\mathsf{e}} h$*, together with the carried endpoints in* A_{e}*. It exports no totality, lattice, host-order, quotient, or non-singleton carrier data.*

Proof. Read the source tuple from Definition 53.11: the only visible endpoints are histories carried by A_{e}, and the only relation witnesses are guarded comparisons for $\leq_{\mathsf{e}}$.

Apply Definition 53.12 to identify the two opposing comparison witnesses as precisely the data used by antisymmetry and exactness. Then apply Definition 53.15. Its ledger row is the repackaging of those two witnesses with the carried endpoints, so no additional order-theoretic or quotient field is exported. □

Theorem 53.20 (POSet prefix ledger row exhaustion)**.** *For the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *certificate of Definition 53.15, every public ledger row is exhausted by a tuple* $(h, k, A_{\mathsf{e}}(h), A_{\mathsf{e}}(k), h \leq_{\mathsf{e}} k, k \leq_{\mathsf{e}} h, \rho_{\mathsf{e}}(h, k))$*. Conversely, any tuple of this form repacks as a public ledger row of that certificate. The ten-field fit is: target ledger exhaustion for the one-point* $\mathsf{POSet}^{\uparrow}$ *ledger; BEDC source the singleton* BHist *carrier; carrier* A_{e}*; classifier* ρ_{e}*; relation* $\leq_{\mathsf{e}}$*; stability from singleton transport and exactness; ledger the two guarded comparisons; core theorem this row-exhaustion statement; exactness the bidirectional repacking; standard bridge the one-point poset; status* $\mathsf{ProofTarget}$*.*

Proof. Unfold Definition 53.11. Its public source rows have histories in the singleton BHist carrier A_{e}, the guarded relation $\leq_{\mathsf{e}}$, and the classifier ρ_{e} on those carried endpoints.

For the forward direction, start with an arbitrary public ledger row of the certificate. Apply Definition 53.19: the row contains only the carried endpoints and the two guarded comparisons $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$.

Apply Definition 53.12. Its singleton transport keeps the endpoints inside A_{e}, and its exactness row identifies $\rho_{\mathsf{e}}(h, k)$ with the opposing comparison pair. Thus the row has the displayed tuple form.

For the reverse direction, begin with such a tuple. The endpoint entries give the carried singleton histories, the comparison entries give the two ledger witnesses, and the classifier entry is exact for the same pair by Definition 53.12.

Finally apply Definition 53.15. It packages precisely these endpoint, comparison, transport, and exactness rows as the concrete singleton $\mathsf{POSet}^{\uparrow}$ certificate, while Definition 53.19 rules out any totality, lattice, host-order, quotient, or non-singleton carrier row. □

Theorem 53.21 (POSet singleton comparison endpoint determinacy)**.** *For the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *package of Definition 53.11, any two guarded singleton comparisons over* A_{e} *determine the same endpoint classifier* ρ_{e}*. There is no second endpoint row besides the carried singleton endpoint and its* hsame *transport.*

Proof. Read the comparison rows from Definition 53.11. Their endpoints are required to lie in the singleton carrier A_{e}, so both comparisons have the same displayed carrier boundary.

Apply Definition 53.12. The singleton laws collapse carrier membership to the empty-history endpoint and identify every guarded comparison with the classifier row ρ_{e} on that endpoint.

Finally apply Definition 53.15. The certificate exposes exactly this classifier as the $\mathsf{POSet}^{\uparrow}$ endpoint row, so the two guarded comparisons cannot generate a second endpoint classification. □

Theorem 53.22 (POSet singleton antichain terminality)**.** *Every antichain subcarrier of the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *package is terminal: each carried member is classified with* e*, and every comparison between carried members collapses by Definition 53.12. No nontrivial incomparable pair is exported.*

Proof. Fix an antichain subcarrier inside the carrier of Definition 53.11. Any carried member is already a member of A_{e}, hence the singleton carrier law classifies it with e.

For a comparison between two carried members, apply Definition 53.12. The law package identifies both endpoints with the singleton endpoint and collapses the guarded comparison to the displayed classifier row.

Use Definition 53.15 to read this row as the concrete $\mathsf{POSet}^{\uparrow}$ certificate boundary. An antichain subcarrier therefore has only the terminal singleton member and no exported nontrivial comparison data. □

Theorem 53.23 (POSet singleton ledger exhaustion)**.** *The concrete singleton* $\mathsf{POSet}^{\uparrow}$ *ledger is exhausted by the two guarded comparison witnesses* $h \leq_{\mathsf{e}} k$ *and* $k \leq_{\mathsf{e}} h$ *together with the two carried endpoints in* A_{e}*. It contains no total-order row, lattice row, quotient row, host-order row, or additional carrier row.*

Proof. Start from Definition 53.19. Its ledger description names exactly the two guarded comparison witnesses and the carried endpoints as the visible data of the singleton certificate.

Apply Definition 53.12 to the comparison witnesses. They supply antisymmetry and exactness only through the singleton classifier boundary, so no totality, lattice operation, quotient representative, or ambient host order can be read from them.

The certificate theorem Definition 53.15 repackages precisely those rows as the concrete $\mathsf{POSet}^{\uparrow}$ naming certificate. Hence the ledger is fully exhausted by the displayed comparison and endpoint rows. □

Theorem 53.24 (POSet unary prefix standard bridge boundary)**.** *The concrete unary-prefix and singleton-empty* $\mathsf{POSet}^{\uparrow}$ *packages bridge to the finite unary-prefix partial order by reading* A_{e} *and* $\mathsf{UnaryHistory}$ *as the carried finite-word endpoints,* ρ_{e} *and* hsame *as classifier equality, and* $\leq_{\mathsf{e}}$ *together with* $\mathsf{PreorderPrefixLE}$ *as initial-segment comparison. The bridge exports exactly the concrete carrier, classifier, antisymmetry, exactness, and ledger rows named in Definition 53.10, Definition 53.12, Definition 53.15, and Definition 53.23; it exports no total-order, lattice, quotient, or host-order row.*

Proof. First read the unary-prefix carrier and comparison from Definition 53.10. This identifies $\mathsf{UnaryHistory}$ with finite unary-word endpoints and reads $\mathsf{PreorderPrefixLE}$ as initial-segment comparison.

Next read the singleton-empty endpoint package through Definition 53.12. The singleton carrier A_{e}, classifier ρ_{e}, and hsame transport give the classifier-equality boundary for the carried endpoint.

Finally apply Definition 53.15 and Definition 53.23. They package exactly the carrier, classifier, antisymmetry, exactness, and finite ledger rows available to the concrete $\mathsf{POSet}^{\uparrow}$ surface, and they rule out total-order, lattice, quotient, and host-order data. □

Theorem 53.25 (POSet unary prefix bridged closure package)**.** *The unary-prefix and singleton-empty* $\mathsf{POSet}^{\uparrow}$ *packages, together with the finite-chain endpoint classifier coverage and selector boundary, form the bridge package from the concrete* $\mathsf{BHist}\ \mathsf{POSet}^{\uparrow}$ *surface to the finite unary-prefix partial order. The package exports only the carried unary-prefix endpoints, singleton endpoint classifier, guarded prefix comparisons, finite-chain endpoint classifiers, and selector-consumer boundary; it exports no host order, quotient representative, total-order row, or lattice operation.*

Proof. Start with Definition 53.24. It reads the concrete unary-prefix and singleton-empty packages as the finite unary-prefix partial order interface, with carrier, classifier, comparison, antisymmetry, exactness, and ledger rows coming only from the displayed $\mathsf{POSet}^{\uparrow}$ surface.

Next apply Definition 53.23. This removes any extra host-order, quotient, total-order, or lattice field from the singleton ledger, so the bridge package has no additional exported structure beyond the concrete endpoint and comparison rows.

Finally compose Definition 53.26 with Definition 53.28. The first theorem supplies the finite-chain endpoint classifier rows, and the second restricts selectors to those carried chain rows. Hence

the bridge package is consumable by the finite unary-prefix partial order boundary and exports exactly the rows named in the statement. □

Theorem 53.26 (POSet finite-chain endpoint classifier coverage)**.** *For the concrete singleton* $\mathsf{POSet}^{\uparrow}$ *package of Definition 53.11, let* $h_0, \ldots, h_n : \mathsf{BHist}$ *be a finite guarded chain whose entries are carried by* A_{e} *and whose adjacent rows supply* $h_j \leq_{\mathsf{e}} h_{j+1}$*. Then every endpoint pair* h_i, h_j *with* $0 \leq i \leq j \leq n$ *is classified by* ρ_{e}*, and every public chain-ledger row repacks from carried* BHist *endpoints, adjacent guarded comparisons, transitive prefix witnesses, and the classifier rows of Definition 53.12.*

The ten-field fit is: downstream target $\mathsf{Order}^{\uparrow}$ *finite interval coverage for chains read through the singleton* $\mathsf{POSet}^{\uparrow}$ *boundary; BEDC source* BHist*; carrier* A_{e}*; classifier* ρ_{e}*; relation* $\leq_{\mathsf{e}}$*; stability from singleton carrier transport, guarded-prefix transitivity, and antisymmetry exactness; ledger adjacent comparisons plus endpoint classifiers; core theorem this finite-chain endpoint coverage statement; exactness classifier repacking from Definition 53.12; standard bridge the singleton finite interval seen by* $\mathsf{TotalOrder}^{\uparrow}$ *and* $\mathsf{Order}^{\uparrow}$ *consumers; status* $\mathsf{ProofTarget}$*.*

Proof. Induct on the finite guarded chain. In the one-endpoint case, the carried row $A_{\mathsf{e}}(h_0)$ is a singleton carrier row. Apply Definition 53.12 to obtain the reflexive guarded comparison, the endpoint classifier $\rho_{\mathsf{e}}(h_0, h_0)$, and the exactness row that reads this classifier as the public singleton ledger entry.

For the step, suppose the prefix ending at h_m already has endpoint classifier coverage and append the adjacent guarded comparison $h_m \leq_{\mathsf{e}} h_{m+1}$. The singleton laws keep h_{m+1} inside A_{e} and classify the adjacent endpoint pair through ρ_{e}.

Now take any earlier endpoint h_i. The induction hypothesis supplies the endpoint comparison from h_i to h_m and the classifier row for that pair. Compose this comparison with $h_m \leq_{\mathsf{e}} h_{m+1}$ by transitivity in Definition 53.12. Apply Definition 53.13 to the triple h_i, h_m, h_{m+1}, and use Definition 53.14 to read the resulting endpoint comparison as the singleton classifier $\rho_{\mathsf{e}}(h_i, h_{m+1})$.

It remains to identify the public ledger row. The induction data provide the carried BHist endpoints and all adjacent guarded comparisons, while the preceding paragraph supplies the transitive endpoint witnesses and classifier rows for the newly exposed endpoint pairs. Repack the pairwise rows by Definition 53.20, and then apply Definition 53.23 to rule out any additional total-order, lattice, quotient, host-order, or non-singleton carrier field. Thus the whole finite chain is covered by the concrete singleton endpoint classifier package that downstream $\mathsf{TotalOrder}^{\uparrow}$ and $\mathsf{Order}^{\uparrow}$ consumers cite. □

Theorem 53.27 (POSet prefix chain OrderUp interval coverage)**.** *Every finite guarded* $\mathsf{POSet}^{\uparrow}$ *chain over carried* BHist *endpoints supplies the* $\mathsf{Order}^{\uparrow}$ *finite-interval carrier, endpoint classifier, and adjacent comparison ledger rows needed by the unary-prefix order phase. The carrier rows are the carried endpoint rows, the endpoint classifier rows are those of Definition 53.26, and the interval ledger is exhausted by adjacent guarded comparisons together with the collapse rows of Definition 53.14.*

Proof. Start with a finite guarded chain. Apply Definition 53.26 to obtain classifier coverage for every endpoint pair of the chain.

Next apply Definition 53.14. It reads the adjacent guarded comparisons as the transitive finite-chain comparison rows, so the endpoint pairs form the finite interval consumed by $\mathsf{Order}^{\uparrow}$.

Finally use Definition 53.23. The ledger has no entry beyond carried endpoints, adjacent comparisons, transitive endpoint witnesses, and classifier rows. These are exactly the finite-interval carrier, endpoint classifier, and adjacent comparison rows asserted above. □

Theorem 53.28 (POSet prefix chain TotalOrder selector consumer boundary)**.** *A* $\mathsf{TotalOrder}^{\uparrow}$ *selector consuming a finite guarded* $\mathsf{POSet}^{\uparrow}$ *chain may use only the carried endpoints, the branch comparison rows obtained from the chain, the endpoint classifier rows, and the finite-chain ledger supplied by Definition 53.26. It may not consume host order, quotient representatives, lattice operations, or any comparison datum outside the carried* BHist *chain.*

Proof. Read the consumer rows from Definition 53.26. They are carried BHist endpoints, adjacent guarded comparisons, transitive prefix witnesses, and endpoint classifier rows.

Apply Definition 53.14 to turn the adjacent rows into the branch comparison data visible to a selector. The result still lives over the same carried endpoints and does not introduce a host comparison.

Apply Definition 53.23 at the boundary. It rules out total-order data, quotient representatives, lattice operations, and host-order rows. Therefore the selector boundary is exactly the carried chain data listed in the statement. □

Theorem 53.29 (POSet prefix selector classifier determinacy)**.** *For a finite guarded* $\mathsf{POSet}^{\uparrow}$ *chain over carried* BHist *endpoints, any two* $\mathsf{TotalOrder}^{\uparrow}$ *or* $\mathsf{Order}^{\uparrow}$ *selectors that use the same chain ledger, the same adjacent comparison rows, and the same endpoint classifier coverage select endpoints related by the classifier rows supplied by the chain package. The conclusion is a classifier-equivalent endpoint boundary inside the finite chain, not a host order comparison or quotient representative.*

Proof. Project endpoint coverage by Definition 53.26. It gives classifier rows for every endpoint pair named by the shared guarded chain.

Next apply Definition 53.14. The adjacent comparison rows collapse to the same finite-chain boundary rows for both selectors, because the chain ledger and adjacent comparisons are shared.

Finally use Definition 53.23. It excludes any selector data outside the carried endpoints, adjacent comparisons, transitive witnesses, and endpoint classifiers. Hence both selectors read the same finite-chain classifier boundary, so their selected endpoints are related by the chain package classifier rows. □

53.3 The certificate

Definition 53.30 (POSet source specification)**.** *The source is a certified preorder carrier plus an antisymmetry field.*

Definition 53.31 (POSet pattern specification)**.** *The pattern is preorder relation formation with classifier collapse from bidirectional comparison.*

Definition 53.32 (POSet classifier specification)**.** *The classifier is the preorder carrier classifier.*

Theorem 53.33 (Prefix antisymmetry)**.** *For the concrete unary history prefix preorder, if* $h \leq k$ *and* $k \leq h$*, then* h *and* k *are classified the same by the preorder classifier.*

```
↪variant BEDC.Derived.PreorderUp.PreorderPrefixLE antisymm hsame
```

Proof. Unpack the two prefix witnesses. Substituting the first continuation into the second gives a loop from h through the appended forward and backward tails. Right-unit uniqueness and append-empty inversion force the forward tail to be e, so the first continuation reduces to $\mathsf{hsame}(h, k)$, which is the classifier specification. □

Theorem 53.34 (Prefix classifier exactness for concrete POSet)**.** *For carrier histories h, k in* PreorderCarrier*, the carrier-aware classifier relation* PreorderCarrierClassifierSpec(PreorderCarrier)(h, k) *holds exactly when* PreorderPrefixLE(h, k) *and* PreorderPrefixLE(k, h) *both hold.*

Proof. Assume first that the carrier-aware classifier holds. Its equality content is hsame(h, k), and symmetry gives hsame(k, h). Applying Definition 52.17 in the two orientations gives PreorderPrefixLE(h, k) and PreorderPrefixLE(k, h).

Conversely, assume both prefix comparisons and keep the carrier hypotheses for h and k. By Definition 53.33, the opposing comparisons collapse to hsame(h, k). Combining this equality content with the two carrier witnesses reassembles PreorderCarrierClassifierSpec(PreorderCarrier)(h, k). □

Theorem 53.35 (Prefix interval classifier collapse for concrete POSet)**.** *For carrier histories h, k, r in* PreorderCarrier*, if* PreorderPrefixLE(h, k)*,* PreorderPrefixLE(k, r)*, and* PreorderCarrierClassifierSpec(PreorderCarrier *hold, then both* PreorderCarrierClassifierSpec(PreorderCarrier)(h, k) *and* PreorderCarrierClassifierSpec(PreorderCarrier)(*hold.*

The per-object fields are as follows: the traditional target is interval collapse for order-convex consumers; the BEDC source is the unary BHist *prefix order; the carrier is* PreorderCarrier*; the classifier is* PreorderCarrierClassifierSpec*; stability comes from prefix transitivity and antisymmetry; the ledger records the chains $h \leq k \leq r$ and $r \leq h \leq k$; the core theorem is this interval classifier collapse; exactness is inherited from Definition 53.34; the standard bridge is the finite unary prefix order; and the status is* ProofTarget*.*

Proof. Unpack the endpoint classifier PreorderCarrierClassifierSpec(PreorderCarrier)(h, r). It provides carrier data for h and r together with hsame(h, r). The displayed comparison PreorderPrefixLE(h, k) and Definition 52.21 give the carrier witness for k.

Symmetry of hsame(h, r) and Definition 52.17 give PreorderPrefixLE(r, h). Compose this with PreorderPrefixLE(k, r) by Definition 52.28 to obtain PreorderPrefixLE(k, h). Now Definition 53.33 applies to PreorderPrefixLE(h, k) and PreorderPrefixLE(k, h), yielding hsame(h, k).

For the right subinterval, compose PreorderPrefixLE(r, h) with the displayed comparison PreorderPrefixLE(h, k) to get PreorderPrefixLE(r, k). PreorderPrefixLE(k, r) is already available, so Definition 53.33 gives hsame(k, r).

Finally repackage the two classifier witnesses. Combine the carrier witnesses for h and k with hsame(h, k) to obtain PreorderCarrierClassifierSpec(PreorderCarrier)(h, k), and combine the carrier witnesses for k and r with hsame(k, r) to obtain PreorderCarrierClassifierSpec(PreorderCarrier)(k, r). □

Thus Definition 53.34 supplies the exactness predicate named in item 9 of Definition 53.9.

Theorem 53.36 (Concrete POSet$^{\uparrow}$ naming certificate)**.** *The tuple of Definition 53.9 supplies the source, pattern, classifier, stability, and ledger data of a concrete* NameCert *licensing* POSet$^{\uparrow}$ *over* PreorderCarrier = UnaryHistory*, classifier* PreorderCarrierClassifierSpec(PreorderCarrier)*, and relation* PreorderPrefixLE*.*

Proof. The carrier, preorder, relation-congruence, and antisymmetry fields are exactly the fields certified in Definition 53.10. They identify the carrier with unary histories, use the carrier-aware preorder classifier, and close the partial-order law by prefix antisymmetry.

The classifier exactness field is Definition 53.34. It identifies the carrier classifier with the two opposing prefix comparisons required by the concrete partial-order presentation.

The ledger component records the two comparison witnesses, as specified by Definition 53.44. Repacking these source, pattern, classifier, stability, and ledger components gives the displayed concrete NameCert for $\mathsf{POSet}^{\uparrow}$. □

This theorem fits the per-object template as follows: the traditional target is the finite unary-word prefix partial order; the BEDC source is BHist restricted to unary histories; the carrier is PreorderCarrier = UnaryHistory; the classifier is PreorderCarrierClassifierSpec(PreorderCarrier); the stability obligations are the concrete preorder laws, classifier congruence, and prefix antisymmetry; the ledger records the two comparison-tail witnesses; the core theorem is Definition 53.10; the exactness predicate is Definition 53.34; the standard bridge is the finite unary-word initial-segment order; and the status is ProofTarget. This concrete tuple is the schema instance consumed by the POSet certificate, so Lean can target the BHist prefix construction rather than an abstract relation schema.

Theorem 53.37 (POSet public prefix closure boundary)**.** *For the concrete unary-prefix POSet tuple of Definition 53.9, the public surface is exactly the following* BHist*-anchored data:*

1. *carrier* PreorderCarrier = UnaryHistory*;*
2. *classifier* PreorderCarrierClassifierSpec(PreorderCarrier)*;*
3. *relation* PreorderPrefixLE*.*

Its preorder laws, endpoint congruence, antisymmetry, exactness, and ledger rows are supplied by Definition 53.10, Definition 53.34, and Definition 53.36. The package introduces no abstract carrier, abstract classifier, quotient, host equality, or parameterized order relation.

Proof. Unfold the tuple of Definition 53.9. Its carrier coordinate is PreorderCarrier, which is identified there with UnaryHistory inside BHist; its classifier coordinate is PreorderCarrierClassifierSpec(PreorderCarrier); and its relation coordinate is PreorderPrefixLE.

The stability rows come from Definition 53.10. That theorem supplies the preorder fields for PreorderPrefixLE, endpoint congruence for the carrier-aware classifier, and the antisymmetry collapse under that classifier.

The bidirectional-collapse row is Definition 53.34. It states that the carrier-aware classifier is exact for the two opposing prefix comparisons, so the public exactness field uses no quotient or external equality relation.

Finally, Definition 53.36 repacks the source, pattern, classifier, stability, and ledger rows into the concrete NameCert. Since each repacked row names the same carrier, classifier, and prefix relation, the public $\mathsf{POSet}^{\uparrow}$ surface is anchored in BHist rather than in an abstract carrier or parameterized order schema. □

Theorem 53.38 (POSet antichain subcarrier certificate)**.** *For a concrete* $\mathsf{POSet}^{\uparrow}$ *carrier* $(C, \sim_C, \leq_C)$ *over* BHist*, define the antichain subcarrier* A_C *to consist of carried endpoints* a : BHist *such that every carried endpoint* b : BHist *with* $a \leq_C b$ *or* $b \leq_C a$ *collapses to classifier equality* $a \sim_C b$*. Then* A_C *is a displayed BEDC subcarrier: its carrier row is inherited from* C*, its comparison reads are the two stored* $\leq_C$ *ledger rows, and membership in* A_C *is stable under* hsame *transport and the classifier-congruence fields of Definition 53.37.*

Proof. Start from the public prefix closure boundary. It exposes the carrier PreorderCarrier, the carrier classifier, the prefix relation, the preorder laws, relation congruence, antisymmetry, exactness, and the ledger rows for comparison witnesses.

To build A_C, keep only carried endpoints of the same BHist carrier and attach the displayed antichain clause. No subset quotient is formed: the subcarrier membership row is a package field whose inputs are the carried endpoint, the comparison ledger rows, and the classifier conclusion.

For stability, transport the carried endpoint along hsame and use the relation-congruence rows from the $\mathsf{POSet}^{\uparrow}$ certificate to transport any comparison with a carried target. The antichain clause then returns the classifier equality at the source, and classifier transport repacks it at the target endpoint. □

Theorem 53.39 (POSet antichain discrete NameCert)**.** *The relation induced by $\leq_C$ on the antichain subcarrier A_C of Definition 53.38 is the discrete* $\mathsf{POSet}^{\uparrow}$ *relation. More precisely, for $a, b \in A_C$, a displayed comparison $a \leq_C b$ holds in the induced relation exactly when the carrier classifier collapses the two endpoints, and the resulting* NameCert *has reflexivity from the inherited preorder row and antisymmetry from the antichain clause.*

Proof. Unpack the induced relation on A_C. Its comparison ledger is the same $\leq_C$ ledger stored by the ambient $\mathsf{POSet}^{\uparrow}$ package, but both endpoints now carry the antichain membership row.

Reflexivity is inherited directly from the preorder component of Definition 53.10. Thus every endpoint of A_C has the displayed comparison to itself required by the discrete relation.

For antisymmetry, suppose $a, b \in A_C$ and $a \leq_C b$. The antichain clause for a applies to the carried endpoint b and collapses the comparison to $a \sim_C b$. Conversely, classifier equality transports the reflexive comparison on a to a comparison with b by the relation-congruence fields, so the induced order has no non-discrete comparison content. □

Theorem 53.40 (POSet antichain interval heredity)**.** *For a, b, c in the concrete unary-prefix* $\mathsf{POSet}^{\uparrow}$ *carrier, suppose that $a, c \in A_C$, that the comparison ledger records $a \leq_C b$ and $b \leq_C c$, and that all three endpoints carry the concrete* BHist *carrier rows. Then $b \in A_C$, and the two endpoint comparisons collapse to the carrier classifier:*

$$a \sim_C b \quad \textit{and} \quad b \sim_C c.$$

Proof. Compose the displayed comparisons $a \leq_C b$ and $b \leq_C c$ by the transitivity row of the concrete prefix preorder. This gives $a \leq_C c$ inside the same comparison ledger.

Apply the antichain clause for $a \in A_C$ to the carried endpoint c. Since $a \leq_C c$ is available, the clause collapses the endpoints to the carrier classifier $a \sim_C c$.

Use Definition 53.34 to read this classifier collapse as opposing prefix comparisons between a and c. Relation congruence transports the displayed comparisons $a \leq_C b$ and $b \leq_C c$ through the classifier collapse, yielding the reverse comparisons needed to apply Definition 53.33 on the two subintervals.

Antisymmetry collapses a with b and b with c, giving the two displayed classifier conclusions. To repackage $b \in A_C$, take any carried endpoint d comparable with b; relation congruence transports that comparison across $a \sim_C b$ to a comparison with a, the antichain clause for a collapses a with d, and classifier transitivity transports the collapse back to $b \sim_C d$. □

Theorem 53.41 (POSet antichain functor consumer boundary)**.** *Any downstream finite-set or functor consumer of the antichain construction may read only the antichain carrier row of Definition 53.38, the inherited classifier, the comparison ledger rows, and the* NameCert *provenance of Definition 53.39. It may not read a host-level subset quotient, an external equality relation, or a parameterized order schema beyond the displayed* BHist *package rows.*

Proof. The construction of A_C is package-local. Its carrier, classifier, relation, and comparison ledger are inherited rows from the concrete $\mathsf{POSet}^{\uparrow}$ certificate, while the antichain clause is an additional displayed membership field over carried endpoints.

The discrete $\mathsf{NameCert}$ of Definition 53.39 repacks those rows into the consumer-facing order surface. Finite-set consumers therefore see a finite list of carried endpoints and comparison witnesses; functor consumers see the same classifier and relation-congruence provenance.

Since every field is read from BHist, hsame, the comparison ledger, and $\mathsf{NameCert}$ provenance, the consumer boundary has no access to a host subset quotient or an unrecorded ambient relation. □

Theorem 53.42 (POSet antichain interval consumer exactness)**.** *For a concrete unary-prefix* $\mathsf{POSet}^{\uparrow}$ *packet, restrict any finite-order or functor consumer to the antichain subcarrier* A_C *and to an interval boundary* $a \leq_C b \leq_C c$ *with* $a, c \in A_C$*. The consumer reads exactly the prefix-classifier comparison rows of Definition 53.34, transported to the interval subcarrier by Definition 53.40, and no further order row beyond the public boundary of Definition 53.41.*

Proof. Unpack the concrete $\mathsf{POSet}^{\uparrow}$ naming certificate. The visible carrier, classifier, comparison ledger, and $\mathsf{NameCert}$ provenance are precisely the rows exposed by the unary-prefix packet, so every displayed comparison in the interval is a prefix row before any antichain restriction is applied.

Apply Definition 53.34. It identifies the visible comparison rows with the prefix-classifier data carried by the concrete BHist package. Thus the consumer's comparison information is already the prefix-classifier surface.

Next apply Definition 53.40 to the interval $a \leq_C b \leq_C c$ with antichain endpoints. The middle endpoint is transported into the antichain subcarrier, and both endpoint comparisons collapse to the same classifier rows.

Finally use Definition 53.41. It exhausts the public readback for finite-order and functor consumers by the antichain carrier, inherited classifier, comparison ledger, and $\mathsf{NameCert}$ provenance. Hence no additional order row remains outside the transported prefix-classifier rows. □

Definition 53.43 (POSet stability certificate)**.** *The stability certificate consists of:*

1. *all preorder stability fields*
2. *antisymmetry:* $a \leq b$ *and* $b \leq a$ *imply* $a \sim_C b$
3. *compatibility of antisymmetry with relation congruence*

Definition 53.44 (POSet ledger policy)**.** *Record both comparison derivations when antisymmetry is used.*

Definition 53.45 ($\mathsf{POSet}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{POSet}^{\uparrow}$*.*

Primitive boundary. $\mathsf{POSet}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Proposition 53.46 ($\mathsf{POSet}^{\uparrow}$ forgets to $\mathsf{Preorder}^{\uparrow}$ certificate)**.** *Let* C *carry a* $\mathsf{POSet}^{\uparrow}(C)$ *certificate with classifier* $\sim_C$ *and relation* $\leq_C$*. If the antisymmetry field and the compatibility data whose only role is to transport that antisymmetry field across relation congruence are omitted, the retained carrier, source, pattern, classifier, preorder stability fields, and ledger form a* $\mathsf{Preorder}^{\uparrow}(C)$ *certificate. The carrier* C*, classifier* $\sim_C$*, and relation* $\leq_C$ *are unchanged.*

Proof. By Definition 53.45, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{POSet}^{\uparrow}$. The construction keeps exactly the coordinates that do not require the partial-order antisymmetry clause.

By Definition 53.2, a $\mathsf{POSet}^{\uparrow}$ carrier is a preorder setup together with antisymmetry under the carrier classifier. Removing the antisymmetry coordinate leaves the carrier, classifier, and relation data required by Definition 52.2. Hence C, $\sim_C$, and $\leq_C$ are copied without replacement.

By Definition 53.30, the source is a certified preorder carrier plus an antisymmetry field. Omitting only the antisymmetry field leaves the source described in Definition 52.15. By Definition 53.32, the classifier is the preorder carrier classifier, so it is exactly the classifier specified in Definition 52.25.

By Definition 53.31, the pattern is preorder relation formation together with classifier collapse from bidirectional comparison. Retaining the one-direction judgment formation $a \leq_C b$ and suppressing only the bidirectional-collapse rule leaves the relational judgment formation of Definition 52.16.

By Definition 53.43, the first stability component is all preorder stability fields. These are precisely the classifier reflexivity, classifier transitivity, preorder reflexivity, preorder transitivity, and relation-congruence rows listed in Definition 52.44. The remaining $\mathsf{POSet}^{\uparrow}$ rows, namely antisymmetry and compatibility of antisymmetry with relation congruence, are not obligations for $\mathsf{Preorder}^{\uparrow}$. Relation congruence itself is retained because it belongs to the preorder stability component.

Finally, Definition 52.45 asks for the compared carrier witnesses and the relation derivation. The $\mathsf{POSet}^{\uparrow}$ ledger of Definition 53.44 adds the paired comparison record only when antisymmetry is used. With that antisymmetry use omitted, the remaining comparison provenance is the preorder ledger for the same relation $\leq_C$.

The retained carrier, source, pattern, classifier, stability, and ledger fields are therefore exactly the obligations named in Definition 52.46, giving a $\mathsf{Preorder}^{\uparrow}(C)$ certificate with unchanged carrier, classifier, and relation. □

Definition 53.47 (Preorder quotient presentation)**.** *Let $(C, \sim_C, \leq_C)$ be a $\mathsf{Preorder}^{\uparrow}$ package satisfying the carrier and stability fields of Definition 52.2 and Definition 52.44. On carried representatives define*

$$x \approx_C y :\Longleftrightarrow x \leq_C y \wedge y \leq_C x.$$

Let $q : C \to C_{\approx}$ be the class map for $\approx_C$, and write $\overline{x} := q(x)$. The quotient carrier is $C_{\approx}$. Its induced order and quotient classifier are

$$\overline{x} \leq_Q \overline{y} :\Longleftrightarrow x \leq_C y, \qquad \overline{x} \sim_Q \overline{y} :\Longleftrightarrow x \approx_C y.$$

Theorem 53.48 (Preorder quotient yields $\mathsf{Poset}^{\uparrow}$)**.** *Let $(C, \sim_C, \leq_C)$ supply the carrier, classifier, relation, stability, and ledger fields of a $\mathsf{Preorder}^{\uparrow}$ certificate as in Definition 52.46. For the quotient presentation of Definition 53.47, $\approx_C$ is an equivalence relation on carried representatives, the induced order $\leq_Q$ is independent of representatives, and $(C_{\approx}, \sim_Q, \leq_Q)$ supplies the $\mathsf{POSet}^{\uparrow}$ certificate obligations for the quotient partial order. Its classifier is exact on classes:*

$$\overline{x} \sim_Q \overline{y} \Longleftrightarrow x \approx_C y \Longleftrightarrow (x \leq_C y \wedge y \leq_C x).$$

Proof. The preorder stability fields in Definition 52.44 give reflexivity and transitivity of $\leq_C$ on carried representatives. Reflexivity of $\approx_C$ follows by pairing two copies of $x \leq_C x$. Symmetry follows by exchanging the two comparison witnesses. For transitivity, assume $x \approx_C y$ and $y \approx_C z$.

The forward comparison $x \leq_C z$ is obtained by composing $x \leq_C y$ with $y \leq_C z$, and the reverse comparison $z \leq_C x$ is obtained by composing $z \leq_C y$ with $y \leq_C x$. Hence $\approx_C$ is an equivalence relation.

The induced order is representative-independent. Suppose $x \approx_C x'$, $y \approx_C y'$, and $x \leq_C y$. The reverse half of $x \approx_C x'$ gives $x' \leq_C x$, so transitivity gives $x' \leq_C y$. The forward half of $y \approx_C y'$ gives $y \leq_C y'$, so another use of transitivity gives $x' \leq_C y'$. Thus $\overline{x} \leq_Q \overline{y}$ implies $\overline{x'} \leq_Q \overline{y'}$ under replacement of representatives. Taking one representative fixed gives the two one-sided relation-congruence rows.

The quotient classifier is also representative-independent because $\approx_C$ is an equivalence relation: from $x \approx_C x'$ and $y \approx_C y'$, transitivity and symmetry give $x \approx_C y$ exactly when $x' \approx_C y'$. Therefore $\sim_Q$ is a well-defined classifier on $C_\approx$, and unfolding its definition gives the displayed exactness equivalence.

The $\mathsf{POSet}^\uparrow$ laws follow directly. Reflexivity of $\leq_Q$ is the reflexivity of $\leq_C$ on any representative. Transitivity of $\leq_Q$ is the transitivity of $\leq_C$ after unfolding the two induced comparisons. For antisymmetry, if $\overline{x} \leq_Q \overline{y}$ and $\overline{y} \leq_Q \overline{x}$, then $x \leq_C y$ and $y \leq_C x$, hence $x \approx_C y$, and therefore $\overline{x} \sim_Q \overline{y}$ by the quotient classifier.

The source row is the quotient carrier $C_\approx$, the pattern row is $\leq_Q$ as in Definition 53.31, and the classifier row is $\sim_Q$ as in Definition 53.32. The stability rows required by Definition 53.43 are the quotient reflexivity, transitivity, relation congruence, and antisymmetry just proved. The ledger of Definition 53.44 records the two representatives and, when antisymmetry is invoked, the two induced comparisons. These are exactly the fields listed in Definition 53.45, so the quotient presentation supplies the stated $\mathsf{POSet}^\uparrow$ certificate and hence the $\mathsf{Poset}^\uparrow$ quotient presentation. □

Definition 53.49 (Opposite POSet certificate data)**.** *Let a $\mathsf{POSet}^\uparrow$ certificate be presented by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, and relation $\leq_C$. Define the opposite relation by*

$$a \leq_C^{\mathrm{op}} b := b \leq_C a.$$

The opposite POSet data C^{op} have the same carrier predicate $\mathcal{C}$, the same classifier $\sim_C$, and relation $\leq_C^{\mathrm{op}}$. An antisymmetry ledger for $a \leq_C^{\mathrm{op}} b$ and $b \leq_C^{\mathrm{op}} a$ is the same pair of original comparison derivations, read as $b \leq_C a$ and $a \leq_C b$.

Theorem 53.50 (Opposite POSet certificate)**.** *If $(\mathcal{C}, \sim_C, \leq_C)$ satisfies the $\mathsf{POSet}^\uparrow$ certificate obligations of Definition 53.45, then the data of Definition 53.49 also satisfy the $\mathsf{POSet}^\uparrow$ certificate obligations. Thus the same carrier and classifier carry a POSet certificate with relation $\leq_C^{\mathrm{op}}$.*

Proof. Unpack the given package through Definition 53.45. By Definition 53.46, it has a $\mathsf{Preorder}^\uparrow$ reduct with the same carrier predicate $\mathcal{C}$, classifier $\sim_C$, and relation $\leq_C$. Applying Definition 52.50 to that reduct supplies the carrier, source, preorder pattern, classifier, preorder stability rows, and preorder ledger for the relation $\leq_C^{\mathrm{op}}$ of Definition 53.49.

It remains to supply the partial-order rows of Definition 53.43. Let a and b be carried endpoints and suppose

$$a \leq_C^{\mathrm{op}} b \qquad \text{and} \qquad b \leq_C^{\mathrm{op}} a.$$

Unfolding the opposite relation gives $b \leq_C a$ and $a \leq_C b$. Applying the original antisymmetry field to the ordered pair $a \leq_C b$ and $b \leq_C a$ yields $a \sim_C b$, which is exactly the antisymmetry conclusion required for $\leq_C^{\mathrm{op}}$.

The compatibility of antisymmetry with relation congruence is inherited in the same exchange of endpoints. Given classifier replacements $a \sim_C a'$ and $b \sim_C b'$, the opposite preorder relation-congruence row transports the two opposite comparisons to $a' \leq_C^{\mathrm{op}} b'$ and $b' \leq_C^{\mathrm{op}} a'$. Unfolding

gives $b' \leq_C a'$ and $a' \leq_C b'$, and the original antisymmetry compatibility row applies to these transported original comparisons to produce $a' \sim_C b'$.

The source row of Definition 53.30 is therefore a certified preorder carrier plus the antisymmetry field just proved. The pattern row of Definition 53.31 is still preorder relation formation with classifier collapse from bidirectional comparison: a displayed pair of opposite comparisons is read as the same original pair with the two comparison derivations exchanged. The classifier row of Definition 53.32 is unchanged because endpoint classification remains $\sim_C$.

Finally, the ledger row is Definition 53.44. It records both comparison derivations used by antisymmetry; under Definition 53.49, those are exactly the original derivations $b \leq_C a$ and $a \leq_C b$ with the displayed slots exchanged. Hence the carrier, source, pattern, classifier, stability, and ledger rows satisfy Definition 53.45, so C^{op} carries a $\mathsf{POSet}^{\uparrow}$ certificate. □

Theorem 53.51 (Double opposite POSet certificate data identity)**.** *Let D be a $\mathsf{POSet}^{\uparrow}$ certificate satisfying the obligations of Definition 53.45. Present its POSet data by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, order relation $\leq_C$, and antisymmetry ledger row $\mathcal{L}_C$ as in Definition 53.44. Let $D^{\mathrm{opop}} = (D^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 53.49 twice. Then D^{opop} is a $\mathsf{POSet}^{\uparrow}$ certificate, and its carrier predicate, classifier, order relation, and antisymmetry ledger row are componentwise identical to those of D: for carried endpoints a, b and comparison witnesses $p : a \leq_C b$ and $q : b \leq_C a$,*

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}(a), \qquad a \sim_C^{\mathrm{opop}} b = a \sim_C b,$$
$$a \leq_C^{\mathrm{opop}} b = a \leq_C b, \quad \mathcal{L}_C^{\mathrm{opop}}(a, b; p, q) = \mathcal{L}_C(a, b; p, q).$$

Here $\mathcal{L}_C(a, b; p, q)$ denotes the ledger entry that records the ordered antisymmetry pair p, q.

Proof. Because D satisfies the $\mathsf{POSet}^{\uparrow}$ obligations, Definition 53.50 makes D^{op} a certified $\mathsf{POSet}^{\uparrow}$ instance. Applying the same theorem to that certified opposite data makes D^{opop} a certified $\mathsf{POSet}^{\uparrow}$ instance.

It remains to compare the displayed data. For the carrier predicate and classifier, two unfoldings of Definition 53.49 give

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}^{\mathrm{op}}(a) = \mathcal{C}(a)$$

and

$$a \sim_C^{\mathrm{opop}} b = a \sim_C^{\mathrm{op}} b = a \sim_C b.$$

For the order relation, the two reversals cancel:

$$a \leq_C^{\mathrm{opop}} b = b \leq_C^{\mathrm{op}} a = a \leq_C b.$$

Thus the double opposite order field is exactly the original order field.

For the ledger row, Definition 53.49 says one opposite keeps the same two antisymmetry derivations and exchanges their displayed source and target slots. Hence the first opposite reads the ordered ledger entry for $(a, b; p, q)$ in the exchanged slots, and the second opposite performs the same exchange again. The endpoints and the two comparison slots are restored to $(a, b; p, q)$, so $\mathcal{L}_C^{\mathrm{opop}}(a, b; p, q) = \mathcal{L}_C(a, b; p, q)$.

The certified status and the four displayed component identities therefore identify the double opposite POSet data with the original POSet data. □

Definition 53.52 (POSet chain comparison envelope)**.** *A $\mathsf{POSet}^{\uparrow}$ chain comparison envelope over concrete* BHist *endpoints consists of the following finite data.*

1. *Two finite* PreorderPrefixLE *chains, one from h to k and one from k to h, written as carried* BHist *endpoint rows.*

2. *The two composite comparison rows obtained from those chains:* $p : \mathsf{PreorderPrefixLE}(h, k)$ *and* $q : \mathsf{PreorderPrefixLE}(k, h)$.
3. *Carrier witnesses for every endpoint appearing in either chain, including the boundary endpoints* h *and* k.
4. *A finite* Cont *ledger recording the adjacent comparisons, the two composite comparison rows, and the bidirectional pair* p, q *used by the quotient classifier of Definition 53.47.*

The quotient readback of the envelope is the classifier witness generated by the pair p, q*. The opposite readback is the same finite envelope with the two chains and the two displayed comparison rows exchanged.*

Theorem 53.53 (POSet chain envelope opposite stability)**.** *Let* E *be a* $\mathsf{POSet}^{\uparrow}$ *chain comparison envelope as in Definition 53.52. Reversing* E *to the opposite envelope exchanges the two finite chain ledgers, preserves the quotient classifier generated by the bidirectional comparison pair, and records the same endpoint carrier witnesses. After applying the opposite construction twice, the resulting envelope has the original carrier rows, quotient classifier row, order relation, composite comparison rows, and finite* Cont *ledger rows of* E.

Proof. First read the two finite chains as carrier data. By Definition 53.54, every endpoint in either chain is carried, and the composite rows p and q remain carried PreorderPrefixLE comparisons. Applying Definition 53.55 to the two composite rows gives the endpoint classifier row determined by the envelope.

Next compare the quotient readback. In Definition 53.47, the quotient classifier is generated exactly by a bidirectional comparison pair. The opposite envelope only exchanges the two displayed rows, so the generated classifier witness is the same finite pair read in the opposite order; no additional representative or carrier-equivalence row is introduced.

Finally apply Definition 53.51. One opposite exchanges the two chain ledgers and the two composite comparison slots, and a second opposite exchanges them back. The carrier predicate, quotient classifier, order relation, and ledger slots are therefore restored componentwise to the rows of E. □

Theorem 53.54 (POSet finite-chain carrier closure)**.** *Let*

$$h = h_0,\ h_1, \ldots, h_n = k$$

be a finite chain of carried unary-prefix histories in the concrete $\mathsf{POSet}^{\uparrow}$ *source of Definition 53.9. If each adjacent comparison* $h_i \leq h_{i+1}$ *is recorded by* PreorderPrefixLE *in the finite* Cont *ledger, then every endpoint* h_i *in the chain remains inside* PreorderCarrier = UnaryHistory*, and the composite comparison from* h *to* k *is again a carried unary-prefix comparison recorded by that finite ledger.*

Proof. Argue by induction on the finite chain. The base chain has only the carried endpoint h, so carrier closure is immediate from the displayed source row.

For the step, assume the prefix subchain from h to h_i has already stayed inside UnaryHistory and has produced a carried composite comparison. The adjacent row $h_i \leq h_{i+1}$ is a displayed PreorderPrefixLE comparison between carried histories, so the prefix transitivity row cited in Definition 53.36 composes it with the subchain comparison.

The same finite Cont ledger records the old subchain, the new adjacent comparison, and their composite. Thus the next endpoint remains in PreorderCarrier and the chain closes inside the concrete unary prefix carrier. □

Theorem 53.55 (POSet finite-chain antisymmetry collapse)**.** *Let h and k be carried unary-prefix histories. If the finite* Cont *ledger records a finite* PreorderPrefixLE *chain from h to k and another finite* PreorderPrefixLE *chain from k to h, then the two endpoints collapse to the carrier classifier:*

$$\mathsf{PreorderCarrier}(h) \wedge \mathsf{PreorderCarrier}(k) \wedge h \leq k \wedge k \leq h \Longrightarrow \mathsf{hsame}(h, k).$$

Proof. First apply Definition 53.54 to the chain from h to k. It gives a carried composite comparison $h \leq k$ recorded by the finite ledger.

Apply the same closure theorem to the reverse chain from k to h. This gives the carried composite comparison $k \leq h$ over the same unary-prefix carrier.

Now apply Definition 53.33 to the opposing composite prefix comparisons. The result is $\mathsf{hsame}(h, k)$, and Definition 53.34 identifies this history sameness with the carrier classifier on the displayed unary-prefix endpoints. □

The finite-chain carrier is the POSet-side handoff consumed by $\mathsf{TotalOrder}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ readers, so it depends on the preorder route of chapter 52 and the order hierarchy routes of chapter 54 and chapter 55 rather than standing alone as an isolated carrier float.

Theorem 53.56 (POSet finite-chain NameCert surface)**.** *The finite-chain carrier closure of Definition 53.54 and the finite-chain antisymmetry collapse of Definition 53.55 package as a* $\mathsf{POSet}^{\uparrow}$ NameCert *obligation surface. Its rows are exactly the concrete unary-prefix carrier, the* hsame *classifier, the* PreorderPrefixLE *relation, prefix stability under finite composition, the finite* Cont *ledger for adjacent and composite comparisons, and the exactness row of Definition 53.34. It adds no quotient carrier, opposite presentation, totality field, lattice operation, or standard bridge.*

Proof. Project the concrete unary-prefix certificate of Definition 53.36. It supplies the carrier, classifier, relation, stability, ledger, and exactness rows for the $\mathsf{POSet}^{\uparrow}$ surface.

The finite-chain carrier row is supplied by Definition 53.54. Its induction uses only prefix transitivity and finite Cont composition already present in the certificate ledger.

The finite-chain antisymmetry row is supplied by Definition 53.55. It reduces two finite chains to opposing composite prefix comparisons, applies Definition 53.33, and then reads the result through Definition 53.34.

Repacking these rows with the source certificate gives a single $\mathsf{POSet}^{\uparrow}$ NameCert obligation surface. Since every row is drawn from the concrete unary-prefix packet and its finite ledger, no additional order structure or bridge datum is exported. □

53.4 Poset root obligation surface

Definition 53.57 (Poset root obligation surface)**.** *The concrete root surface for* $\mathsf{Poset}^{\uparrow}$ *is the* BHist *packet read from Definition 53.9. Its endpoints are carried unary histories h, k :* BHist*; its inherited* $\mathsf{Preorder}^{\uparrow}$ *comparison row is* $\mathsf{PreorderPrefixLE}(h, k)$*; its classifier row is the carrier-aware* hsame *classifier* $\mathsf{PreorderCarrierClassifierSpec}(\mathsf{PreorderCarrier})(h, k)$*; its partial-order row is the antisymmetry collapse for the two directed prefix comparisons; and its ledger is the finite* Cont *record consisting of the two comparison-tail witnesses, their carried endpoints, and the resulting classifier readback.*

Theorem 53.58 (Poset root antisymmetry obligation)**.** *For carried unary* BHist *endpoints h and k in the surface of Definition 53.57, the bidirectional inherited prefix comparison*

$$\mathsf{PreorderPrefixLE}(h, k) \wedge \mathsf{PreorderPrefixLE}(k, h)$$

collapses to the carried classifier row $\mathsf{PreorderCarrierClassifierSpec}(\mathsf{PreorderCarrier})(h, k)$.

Proof. Project the two inherited comparison rows from the root surface. They are the same directed prefix rows used by the concrete unary-prefix preorder package, and their endpoints are the carried BHist histories named in Definition 53.57.

Apply Definition 53.33 to the two opposite prefix comparisons. This gives the $\mathsf{hsame}(h, k)$ classifier content without changing the endpoint carrier.

Finally reassemble the carrier-aware classifier with the two carried endpoint witnesses from the root surface. By Definition 53.34, this is exactly the classifier row of the concrete $\mathsf{Poset}^{\uparrow}$ root obligation package. □

Theorem 53.59 (Poset root certificate readback)**.** *The concrete* NameCert *fields for the root surface of Definition 53.57 are read directly from Definition 53.9 and Definition 53.36. The source is* BHist *restricted to unary histories, the carrier is* PreorderCarrier = UnaryHistory*, the classifier is the carrier-aware* hsame *row, the relation is* PreorderPrefixLE*, the antisymmetry and exactness rows are the prefix collapse rows, and the ledger is the two-comparison finite* Cont *record.*

Proof. Unpack Definition 53.9. Its displayed coordinates name the source, carrier, classifier, relation, stability rows, exactness row, and comparison ledger used by the root surface.

Apply Definition 53.36. It repacks those coordinates as the concrete NameCert licensing $\mathsf{POSet}^{\uparrow}$ over the same BHist carrier and prefix relation.

The antisymmetry component is the row isolated in Definition 53.58. Therefore the root certificate contains no additional carrier, quotient representative, or external equality witness beyond the fields already read from the concrete unary-prefix certificate. □

Theorem 53.60 (Poset root downstream unblock package)**.** *The root package of Definition 53.59 supplies the rows a* $\mathsf{TotalOrder}^{\uparrow}$ *or* $\mathsf{Lattice}^{\uparrow}$ *consumer may inherit from* $\mathsf{Poset}^{\uparrow}$*: carrier, classifier, antisymmetry, exactness, and finite* Cont *ledger. A downstream consumer may append its own total-comparison selector or meet-join bound rows, but it cannot replace the inherited* BHist *carrier,* hsame *classifier, prefix exactness, or antisymmetry ledger.*

Proof. Start with the certificate readback of Definition 53.59. It exposes the carrier, classifier, relation, stability, exactness, and finite ledger rows of the concrete root surface.

For a $\mathsf{TotalOrder}^{\uparrow}$ consumer, keep the inherited carrier and classifier and add only the selector row deciding a directed comparison between displayed endpoints. The antisymmetry and exactness rows still come from Definition 53.58 and Definition 53.34.

For a $\mathsf{Lattice}^{\uparrow}$ consumer, keep the same inherited rows and add only the meet and join bound-characterization rows. The lower-bound and upper-bound comparisons are ledger entries over the same prefix relation, so their endpoint collapse still reads through the root $\mathsf{Poset}^{\uparrow}$ certificate. □

Theorem 53.61 (Poset root scoped certificate boundary)**.** *The root* $\mathsf{Poset}^{\uparrow}$ *package closes exactly the scoped boundary consisting of unary* BHist *carrier rows, inherited* $\mathsf{Preorder}^{\uparrow}$ *prefix comparison rows, the carrier-aware* hsame *classifier, antisymmetry collapse, finite* Cont *comparison ledgers, and* Pkg *provenance. These are precisely the rows inherited by* $\mathsf{TotalOrder}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *consumers.*

Proof. Begin with Definition 53.59. It reads the concrete unary-prefix certificate of Definition 53.36 from Definition 53.9, naming the carrier, classifier, prefix relation, exactness rows, ledger, and provenance fields.

Apply Definition 53.58 to isolate the partial-order row. The only collapse added beyond the inherited $\mathsf{Preorder}^{\uparrow}$ prefix comparisons is the carrier-aware hsame result of the two directed prefix witnesses.

Finally use Definition 53.60. It states that $\mathsf{TotalOrder}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ consumers inherit exactly those rows and may only append their own selector or bound-characterization data. Thus the scoped boundary is neither weaker nor larger than the listed BEDC primitives. □

53.5 Poset root comparison surface

Definition 53.62 (POSet root comparison surface)**.** *The root comparison surface for* $\mathsf{POSet}^{\uparrow}$ *is the carried* BHist *surface whose carrier row is the unary* BHist *carrier* UnaryHistory *of Definition 53.2, whose classifier row is the carrier* hsame *row on carried unary histories, and whose relation row is the inherited prefix comparison on* BHist*. Its* NameCert *payload consists of the inherited carrier and classifier rows, the two directed comparison rows, the antisymmetry collapse row, and the finite* Cont *ledger governed by Definition 53.44.*

Theorem 53.63 (POSet root hsame transport surface)**.** *For carried unary* BHist *endpoints* h *and* k *in Definition 53.62, an* $\mathsf{hsame}(h, k)$ *classifier row transports the root carrier, classifier, and prefix-comparison rows without changing the* UnaryHistory *carrier or replacing the* NameCert *object by a quotient representative.*

Proof. Read the carrier and classifier rows from Definition 53.2. The root endpoints are already carried BHist histories in UnaryHistory, so the transport target remains inside the same displayed carrier.

Apply the prefix stability rows of Definition 52.31. They transport comparison endpoints along hsame while preserving the inherited prefix relation on the carried histories.

Finally repack the transported carrier, classifier, and comparison rows into the root surface of Definition 53.62. The NameCert object still names the same BHist rows and the same finite Cont ledger. □

Theorem 53.64 (POSet root antisymmetry surface)**.** *For endpoints* $h, k : \mathsf{BHist}$ *carried by* UnaryHistory*, the two root comparison rows* $h \leq_{\mathsf{pref}} k$ *and* $k \leq_{\mathsf{pref}} h$ *determine the root classifier row* $\mathsf{hsame}(h, k)$*, and this classifier is the one recorded by the* $\mathsf{POSet}^{\uparrow}$ NameCert *surface.*

Proof. Project the two directed relation rows from Definition 53.62. They are precisely the inherited prefix-comparison rows on carried unary BHist endpoints.

Apply Definition 53.33. The theorem collapses the opposite prefix witnesses to the classifier conclusion $\mathsf{hsame}(h, k)$.

Use Definition 53.34 to identify this classifier conclusion with the carrier classifier row recorded by the root NameCert surface. Thus antisymmetry produces no host equality or quotient object beyond the displayed hsame row. □

Theorem 53.65 (POSet root ledger exhaustion surface)**.** *Every root* $\mathsf{POSet}^{\uparrow}$ *ledger entry used by the comparison surface is one of the finite* Cont *rows named by Definition 53.44: the carried* BHist *endpoints, the two directed prefix witnesses, and the resulting* hsame *classifier row. No additional comparison, endpoint, or external equality row is part of the root* NameCert *ledger.*

Proof. Start from the ledger policy of Definition 53.44. It names the finite continuation data attached to a $\mathsf{POSet}^{\uparrow}$ comparison use: carried endpoints, directed prefix witnesses, and classifier readback.

The root surface of Definition 53.62 has no other carrier or relation coordinate. Its only relation rows are the inherited prefix comparisons on UnaryHistory endpoints.

Apply Definition 53.64 to any opposing pair in the ledger. The only possible collapse row is hsame, so the displayed Cont ledger exhausts the root NameCert comparison data. □

Theorem 53.66 (POSet root threshold unblock surface)**.** *The root* $\mathsf{POSet}^{\uparrow}$ *surface supplies the dependency-ready rows needed by downstream order constructions: a* BHist *carrier restricted to* UnaryHistory*, an* hsame *classifier, inherited prefix comparison rows, antisymmetry, classifier exactness, and the finite* Cont *ledger. Any downstream* $\mathsf{TotalOrder}^{\uparrow}$ *or* $\mathsf{Order}^{\uparrow}$ *construction may add selector or public-order rows, but it inherits these root rows as fixed* NameCert *inputs.*

Proof. Collect the carrier and classifier rows from Definition 53.62. Collect the inherited prefix comparison and transport behavior from Definition 53.63.

Next add the antisymmetry collapse of Definition 53.64 and the classifier exactness row of Definition 53.34. These rows give the partial-order boundary over the same carried BHist endpoints.

Finally attach the finite ledger exhaustion statement of Definition 53.65. The resulting package is exactly the root NameCert input available to downstream order surfaces; later selector or public export rows extend this package without replacing its carrier, classifier, comparison, antisymmetry, exactness, or ledger rows. □

Theorem 53.67 (POSet root public handoff boundary)**.** *Any downstream* $\mathsf{Order}^{\uparrow}$ *or* $\mathsf{TotalOrder}^{\uparrow}$ *consumer of the* $\mathsf{POSet}^{\uparrow}$ *root may read exactly the* UnaryHistory *carrier row, the* hsame *classifier row, the inherited prefix-comparison row, the antisymmetry collapse row, the classifier exactness row, and the finite* Cont *and* Pkg *ledger rows assembled by Definition 53.66. No host order relation, quotient equality, or representative equality is exported by this root handoff.*

Proof. First collect the transported carrier and classifier surface from Definition 53.63. It keeps every downstream endpoint inside the carried UnaryHistory row and transports comparison data only through displayed hsame rows.

Next add the comparison and collapse rows. The inherited prefix relation is the comparison row of Definition 53.62; Definition 53.64 supplies the antisymmetry collapse, and Definition 53.34 identifies that collapse with the root classifier row.

Finally attach the finite ledger and threshold rows. Use Definition 53.65 for the Cont ledger and Definition 53.66 for the complete root input package. Repacking these rows gives the public handoff to $\mathsf{Order}^{\uparrow}$ and $\mathsf{TotalOrder}^{\uparrow}$, and the absence clauses follow because none of the cited root rows exports a host order relation, quotient equality, or representative equality. □

53.6 Poset prefix schema target surface

Theorem 53.68 (Poset prefix schema target surface)**.** *The concrete unary-prefix* $\mathsf{Poset}^{\uparrow}$ *instance of Definition 53.9 supplies the non-parametric schema target consumed by* $\mathsf{TotalOrder}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$*. Its exported surface is the carrier* UnaryHistory*, the relation* PreorderPrefixLE*, the classifier* PreorderCarrierClassifierS*, the antisymmetry collapse, the finite* Cont *comparison ledgers, and* Pkg *provenance.*

Proof. Project the concrete unary-prefix instance. It names the carrier, prefix relation, carrier-aware classifier, finite comparison ledger, and provenance rows over a fixed BHist surface.

Apply Definition 53.10. The prefix laws provide the reflexive and transitive comparison rows and the antisymmetry collapse needed for the partial-order surface.

Finally apply Definition 53.61. It identifies exactly these rows as the scoped certificate boundary inherited by $\mathsf{TotalOrder}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$, so the schema target is concrete and non-parametric. □

53.7 Poset TotalOrder and Lattice handoff

Theorem 53.69 (POSet TotalOrder and Lattice handoff carrier)**.** *Downstream* $\mathsf{Order}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, *and* $\mathsf{Lattice}^{\uparrow}$ *consumers inherit exactly the unary* BHist *carrier, the* hsame *classifier, the inherited prefix-comparison row, antisymmetry, classifier exactness, the finite* Cont *ledger, and the* Pkg *provenance rows from the* $\mathsf{POSet}^{\uparrow}$ *root. No host order, quotient equality, or replacement carrier is part of this inherited surface.*

Proof. Read the carrier, classifier, and inherited comparison rows from Definition 53.62. These rows already place every endpoint inside the unary BHist carrier and use hsame as the displayed classifier.

Add antisymmetry and exactness from Definition 53.64 and Definition 53.34. The collapse row remains the same hsame classifier named by the root surface.

Finally attach the finite ledger and provenance rows of Definition 53.65 and Definition 53.67. The cited handoff surface exports precisely these rows, so downstream order chapters inherit this package without replacing the carrier or adding quotient data. □

Theorem 53.70 (POSet TotalOrder selector extension boundary)**.** *A* $\mathsf{TotalOrder}^{\uparrow}$ *selector may extend the inherited* $\mathsf{POSet}^{\uparrow}$ *package only by adding a directed-comparison choice row for each comparable endpoint pair. The selector does not change the inherited carrier,* hsame *classifier, prefix comparison, antisymmetry, exactness,* Cont *ledger, or* Pkg *provenance rows.*

Proof. Begin with the inherited package of Definition 53.69. It fixes the carrier, classifier, comparison, antisymmetry, exactness, ledger, and provenance rows.

The additional $\mathsf{TotalOrder}^{\uparrow}$ information is a directed comparison selection over the same carried endpoints. Because the selection targets the inherited prefix-comparison row, it adds only the choice of direction and not a new carrier or classifier.

Repack the selected direction together with the fixed $\mathsf{POSet}^{\uparrow}$ package. The root obligation and unblock surfaces, namely Definition 53.66 and Definition 53.67, already exhaust the inherited rows, so the selector boundary has no further field to replace. □

Theorem 53.71 (POSet Lattice bound extension boundary)**.** $\mathsf{Lattice}^{\uparrow}$ *meet and join bound rows extend the same inherited* $\mathsf{POSet}^{\uparrow}$ *package by adding lower-bound and upper-bound witnesses over the displayed prefix comparison. They do not replace the unary carrier, the* hsame *classifier, the comparison row, antisymmetry, exactness, ledger, or provenance rows inherited from the root.*

Proof. Use Definition 53.69 as the inherited input package. Its carrier, classifier, comparison, antisymmetry, exactness, ledger, and provenance rows are fixed before any meet or join witness is read.

A meet row supplies lower-bound data and a join row supplies upper-bound data for the same displayed prefix comparison. These rows quantify over the existing carried endpoints, so they extend the root package rather than selecting a new endpoint universe.

The exhaustion clauses of Definition 53.65 and Definition 53.67 keep the inherited ledger and provenance rows fixed. Repacking the bound witnesses with those rows gives the $\mathsf{Lattice}^{\uparrow}$ extension boundary without changing the Poset root surface. □

Theorem 53.72 (POSet quotient classifier exhaustion)**.** *In the quotient presentation of Definition 53.47, every classifier witness for quotient classes is generated by the displayed finite pair of preorder comparisons*

$$x \leq_C y \qquad \text{and} \qquad y \leq_C x.$$

The quotient carrier, quotient classifier, and comparison ledger are read through the original BHist *endpoints, the induced* hsame *class row, and the finite* Cont *comparison ledger; no abstract carrier-equivalence transport supplies an additional classifier witness.*

Proof. Unpack the quotient presentation of Definition 53.47. Its class relation is the bidirectional comparison row on carried representatives, so the displayed pair $x \leq_C y$ and $y \leq_C x$ is the only generator of the quotient classifier.

Apply Definition 53.48. The theorem builds the $\mathsf{POSet}^{\uparrow}$ quotient certificate from that finite comparison pair, the inherited carrier row, and the induced order on classes. Its antisymmetry clause is therefore read from the same pair of comparisons, now recorded as the quotient hsame class row.

The remaining data are ledger data. The finite Cont comparison ledger records exactly the two comparison witnesses and their endpoint support in BHist. Since neither Definition 53.47 nor Definition 53.48 introduces a separate carrier equivalence field, every quotient classifier witness is exhausted by this displayed comparison ledger. □

Theorem 53.73 (POSet opposite and quotient compatibility)**.** *For a* $\mathsf{Preorder}^{\uparrow}$ *certificate with carrier C, classifier $\sim_C$, and relation $\leq_C$, taking the opposite relation before forming the quotient presentation gives the same quotient classifier and restored comparison ledger as forming the quotient first and then reversing the induced order. In both routes, a class witness between representatives x and y is the same finite pair $x \leq_C y$ and $y \leq_C x$ with the displayed comparison slots exchanged twice.*

Proof. Start with the quotient route. By Definition 53.47, the quotient classifier is generated by the pair $x \leq_C y$ and $y \leq_C x$, and Definition 53.48 records that pair as the antisymmetry ledger of the induced $\mathsf{POSet}^{\uparrow}$ certificate.

For the opposite-first route, unfold Definition 53.49. Reversal exchanges the two comparison slots, so the quotient classifier for the opposite relation is generated by $y \leq_C x$ and $x \leq_C y$. This is the same finite comparison family with the ordered slots exchanged.

Now reverse the induced quotient order. The data identity of Definition 53.51 says that a second reversal restores the carrier, classifier, order relation, and ledger slots componentwise. Hence the opposite-first quotient and the quotient-first opposite expose the same quotient classifier and the same restored finite Cont comparison ledger, without invoking any abstract carrier-equivalence transport. □

Theorem 53.74 (POSet quotient opposite involutive readback)**.** *Let a concrete* $\mathsf{Preorder}^{\uparrow}$ *carrier over* BHist *have carrier C, classifier $\sim_C$, and comparison relation $\leq_C$. Quotienting by the bidirectional comparison ledger of Definition 53.47, reversing the induced order, and reversing again returns the same quotient classifier witness and the same finite* Cont *comparison ledger. At every endpoint pair, the visible witness remains the concrete pair $x \leq_C y$ and $y \leq_C x$.*

Proof. Unfold the quotient presentation of Definition 53.47. The quotient class relation is generated by the two displayed comparisons, so the classifier witness between representatives is the finite pair $x \leq_C y$ and $y \leq_C x$ together with their BHist endpoint support.

Apply Definition 53.48. The induced $\mathsf{POSet}^{\uparrow}$ certificate stores that same pair as its antisymmetry and comparison ledger; no extra representative equality or host quotient row is exported.

Now apply Definition 53.51. The first opposite exchanges the two comparison slots, and the second opposite restores them componentwise. Therefore the readback after quotient, opposite, and opposite again is the original quotient classifier witness and the original finite Cont ledger, with each row still naming exactly $x \leq_C y$ and $y \leq_C x$. □

Closure status for $\mathsf{NameCert}_{\mathsf{Poset}^{\uparrow}}$**.**

Constructive story: A partial order here is the unary-prefix preorder with one additional visible row: if two carried unary histories extend to each other, the two continuation tails must both collapse and the endpoints are classified by hsame. The carrier is still the unary history predicate over BHist, excluding b0 branches and accepting only the empty history and repeated b1-stacks, while the classifier remains the carrier-aware history-sameness relation inherited from $\mathsf{Preorder}^{\uparrow}$. The source covers the certified prefix-preorder carrier plus the antisymmetry field, the pattern records bidirectional prefix comparison followed by classifier collapse, and the ledger stores the two directed continuation witnesses, their carried endpoints, and the resulting hsame readback. Since the inherited unary carrier is inhabited by e, its classifier is an equivalence, and carrier transport along hsame is already present, adding the antisymmetry row preserves all five name-certificate obligations while closing the partial-order law. A history h lies in the signature gap of a $\mathsf{Poset}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer endpoint and evidence ledger, and token introduction names that answer as p; concretely, h is a unary endpoint in the prefix carrier and every opposing-comparison use in the ledger collapses to the classifier evidence encoded by p. Finite-chain rows and downstream handoff rows reuse the same continuation ledger for $\mathsf{TotalOrder}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ consumers, but they do not change the carrier or add a standard bridge. The chapter registers this antisymmetric unary-prefix package under the name $\mathsf{Poset}^{\uparrow}$; partial order means an extra collapse theorem on the existing prefix relation, not host equality, quotient representatives, or an external poset object.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Poset}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope binds the UnaryHistory carrier, BHist predicate, hsame relation, Preorder prefix comparison, antisymmetry row, Cont transport, Pkg provenance, NameCert classifier, and ledger rows displayed by the PosetUp packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Poset}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PreorderUp.preorder\_prefix\_stability\_certificate\_fields`

Bridge status: paperBridge

Not claimed: The chapter does not claim host order relations, quotient equality, external preorder carriers, or lattice structure.

Upgrade path: No further paper-axis grade remains; only future standard bridge verification can raise the bridge status.

Chapter 54

A Concrete Naming Certificate for $\mathsf{TotalOrder}^{\uparrow}$

The total order interface $\mathsf{TotalOrder}^{\uparrow}$ extends chapter 53 with the trichotomy field, expressed constructively: for any a, b in the carrier, the carrier's apartness predicate decides $a < b$, $a = b$ (under the classifier), or $a > b$. The classical formulation (LEM-style "either $a \leq b$ or $b \leq a$") is replaced by decidable comparison so that the total-order certificate remains 0-axiom.

54.1 Carrier definition

Definition 54.1 (TotalOrder carrier)**.** *Use a poset setup extended with constructive comparison data returning less, same, or greater.*

54.2 Concrete singleton history total order instance

Definition 54.2 (Concrete singleton history total order instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k),$$

and let the order relation on carried endpoints be

$$h \leq_{\mathsf{e}} k := \mathsf{PreorderPrefixLE}(h, k).$$

The constructive comparison package has three branches:

$$\mathsf{lt}_{\mathsf{e}}(h, k) := \bot, \qquad \mathsf{same}_{\mathsf{e}}(h, k) := \rho_{\mathsf{e}}(h, k), \qquad \mathsf{gt}_{\mathsf{e}}(h, k) := \bot.$$

Thus the less and greater branches are empty, while the same branch stores the singleton classifier witness.

Theorem 54.3 (Singleton history total order laws)**.** *For carried histories $h, k : \mathsf{BHist}$ with $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, the comparison package of Definition 54.2 always returns the same branch carrying $\rho_{\mathsf{e}}(h, k)$. The resulting tuple satisfies the $\mathsf{TotalOrder}^{\uparrow}$ carrier field, the $\mathsf{POSet}^{\uparrow}$ fields, classifier stability, strict-to-apartness, trichotomy, and comparison stability.*

Proof. The carrier data are inherited from the singleton history source. The hypotheses $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ unfold to $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$; symmetry and transitivity of hsame compose these endpoint witnesses into $\mathsf{hsame}(h, k)$. Together with the two carrier facts, this is exactly $\rho_{\mathsf{e}}(h, k)$.

The preorder comparisons in both directions are supplied by PreorderPrefixLE_of_hsame from the displayed $\mathsf{hsame}(h, k)$ witness and its symmetric form. These witnesses give the $\mathsf{POSet}^{\uparrow}$ order relation on the restricted singleton carrier.

Antisymmetry is the concrete prefix collapse: PreorderPrefixLE_antisymm_hsame sends opposing prefix comparisons to $\mathsf{hsame}(h, k)$, which repackages with the carrier facts as the classifier $\rho_{\mathsf{e}}(h, k)$. Classifier stability and comparison stability use the same endpoint transport through e.

The strict branches are false by definition, so strict-to-apartness is vacuous. Trichotomy is the same branch just constructed; no classical comparison principle is used. □

Theorem 54.4 (Singleton total-order comparison exactness)**.** *For carried histories $h, k : \mathsf{BHist}$ with $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ in the concrete instance of Definition 54.2, the comparison branches are exact in the following sense:*

1. $\mathsf{lt}_{\mathsf{e}}(h, k) \to \mathsf{False}$ *and* $\mathsf{gt}_{\mathsf{e}}(h, k) \to \mathsf{False}$.
2. $\mathsf{same}_{\mathsf{e}}(h, k) \iff \rho_{\mathsf{e}}(h, k)$.
3. $\mathsf{same}_{\mathsf{e}}(h, k) \iff (h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h)$.

Proof. Unfold the three comparison branches in Definition 54.2. The strict branches are both $\bot$, so each strict witness eliminates to False.

The same branch is defined to be $\rho_{\mathsf{e}}(h, k)$, giving the second equivalence by reflexive unfolding.

For the opposing-prefix characterization, apply the exactness part of Definition 53.12 to the carried histories h and k. Composing that equivalence with the displayed unfolding $\mathsf{same}_{\mathsf{e}}(h, k) \iff \rho_{\mathsf{e}}(h, k)$ gives the stated equivalence with $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$. □

Theorem 54.5 (TotalOrder carrier selector classifier coherence)**.** *For carried singleton histories $h, k : \mathsf{BHist}$ with $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, the branch selector of Definition 54.2 is coherent with the carrier classifier:*

1. $\mathsf{same}_{\mathsf{e}}(h, k) \iff \rho_{\mathsf{e}}(h, k)$.
2. $\mathsf{lt}_{\mathsf{e}}(h, k) \to \mathsf{False}$ *and* $\mathsf{gt}_{\mathsf{e}}(h, k) \to \mathsf{False}$.
3. *For histories $h', k' : \mathsf{BHist}$ with $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$, endpoint replacement preserves the selected branch:*
$$\mathsf{same}_{\mathsf{e}}(h', k') \iff \mathsf{same}_{\mathsf{e}}(h, k).$$

The per-object fields are: traditional target singleton total order; BEDC source the BHist *singleton carrier; classifier ρ_{e}; stability by* hsame *endpoint replacement; ledger branch selection; core theorem this selector coherence; exactness same-branch equivalence with the carrier classifier; standard bridge deferred; downstream use by* $\mathsf{TotalOrder}^{\uparrow}$ *obligation closure; status* ProofTarget.

Proof. Unfold the concrete singleton total-order instance. The selected same branch is defined to be $\rho_{\mathsf{e}}(h, k)$, while the strict-left and strict-right branches are both $\bot$.

Apply Definition 54.3 to the carried histories h and k. It supplies the singleton comparison package and the carrier classifier row built from the two carrier witnesses and hsame transitivity through e.

Apply Definition 54.4. Its first clause eliminates both strict branches to False, and its second clause is the displayed equivalence between the same branch and $\rho_{\mathsf{e}}(h, k)$.

For endpoint replacement, transport the carrier facts through $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$. Symmetry and transitivity of hsame repackage the transported witnesses as $\rho_{\mathsf{e}}(h', k')$ exactly when the original witnesses repackage as $\rho_{\mathsf{e}}(h, k)$.

Composing this transported classifier equivalence with the same-branch equivalences for (h, k) and (h', k') proves preservation of the selected branch. □

Theorem 54.6 (TotalOrder singleton branch transport determinacy)**.** *For histories h, k, h', k' : BHist, assume $A_{\mathsf{e}}(h)$, $A_{\mathsf{e}}(k)$, $\mathsf{hsame}(h, h')$, and $\mathsf{hsame}(k, k')$. Then endpoint replacement along hsame determines the singleton comparison data:*

1. $A_{\mathsf{e}}(h') \wedge A_{\mathsf{e}}(k')$.
2. $\mathsf{lt}_{\mathsf{e}}(h', k') \to \mathsf{False}$ *and* $\mathsf{gt}_{\mathsf{e}}(h', k') \to \mathsf{False}$.
3. $\mathsf{same}_{\mathsf{e}}(h', k') \iff \rho_{\mathsf{e}}(h', k')$.
4. $\rho_{\mathsf{e}}(h', k') \iff \rho_{\mathsf{e}}(h, k)$.

Proof. Transport the two carrier facts through the endpoint replacements. Since $A_{\mathsf{e}}(h)$ means $\mathsf{hsame}(h, \mathsf{e})$, symmetry of $\mathsf{hsame}(h, h')$ and transitivity with $\mathsf{hsame}(h, \mathsf{e})$ give $\mathsf{hsame}(h', \mathsf{e})$, hence $A_{\mathsf{e}}(h')$. The same argument using $\mathsf{hsame}(k, k')$ gives $A_{\mathsf{e}}(k')$.

Apply Definition 54.4 to the carried histories h' and k'. Its strict-branch clause gives both eliminations to False, and its same-branch clause gives $\mathsf{same}_{\mathsf{e}}(h', k') \iff \rho_{\mathsf{e}}(h', k')$.

It remains to compare the singleton classifiers before and after transport. The carrier coordinates have already been transported. For the history coordinate, compose symmetry of $\mathsf{hsame}(h, h')$, the original singleton collapse $\mathsf{hsame}(h, k)$ obtained from $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, and $\mathsf{hsame}(k, k')$ to obtain $\mathsf{hsame}(h', k')$; conversely compose $\mathsf{hsame}(h, h')$, $\mathsf{hsame}(h', k')$, and symmetry of $\mathsf{hsame}(k, k')$ to obtain $\mathsf{hsame}(h, k)$. Repackaging these witnesses with the corresponding carrier facts proves $\rho_{\mathsf{e}}(h', k') \iff \rho_{\mathsf{e}}(h, k)$. □

54.3 Concrete unary prefix total order instance

Definition 54.7 (Concrete unary prefix total order instance)**.** *Let the carrier be the concrete unary-history preorder carrier:*

$$\mathsf{Carrier}_{\mathsf{u}} := \mathsf{PreorderCarrier} = \mathsf{UnaryHistory}.$$

Let the carrier classifier be

$$\rho_{\mathsf{u}}(h, k) := \mathsf{PreorderCarrierClassifierSpec}(\mathsf{PreorderCarrier})(h, k),$$

and let the order relation be the concrete prefix relation

$$h \leq_{\mathsf{u}} k := \mathsf{PreorderPrefixLE}(h, k).$$

The constructive comparison package has the three branches

$$\mathsf{lt}_{\mathsf{u}}(h, k) := \mathsf{NatUnaryStrictPrefix}(h, k), \qquad \mathsf{same}_{\mathsf{u}}(h, k) := \rho_{\mathsf{u}}(h, k), \qquad \mathsf{gt}_{\mathsf{u}}(h, k) := \mathsf{NatUnaryStrictPrefix}(k, h).$$

Thus strict-left means that k extends h by a nonempty unary tail, same is the carrier classifier, and strict-right is the opposite strict unary prefix.

Theorem 54.8 (Concrete unary prefix total order laws)**.** *For carried unary histories h, k in* $\mathsf{Carrier_u}$*, the branches of Definition 54.7 supply constructive total comparison. Each strict branch implies carrier-classifier apartness, the same branch satisfies*

$$\mathsf{same_u}(h, k) \iff (\mathsf{PreorderPrefixLE}(h, k) \wedge \mathsf{PreorderPrefixLE}(k, h)),$$

and the inherited preorder and antisymmetry fields assemble these data into a concrete $\mathsf{TotalOrder}^{\uparrow}$ *law package over unary histories. The strict comparison row is the isolated branch transitivity of Definition 54.17.*

Proof. Start from the total unary prefix comparison of Definition 28.5. For carried histories h and k, this returns a unary continuation from h to k or one from k to h.

Split the selected unary tail by the unary-generation cases. A nonempty tail chooses the corresponding strict branch $\mathsf{NatUnaryStrictPrefix}(h, k)$ or $\mathsf{NatUnaryStrictPrefix}(k, h)$. An empty tail collapses the continuation to $\mathsf{hsame}(h, k)$, which repackages with the two carrier witnesses as $\rho_{\mathsf{u}}(h, k)$ and therefore chooses the same branch.

The strict branches are apart from the same branch. If a strict-left witness and the carrier classifier both held, the classifier exactness of Definition 53.34 would give the reverse prefix comparison as well; splitting that reverse continuation into empty or nonempty form either collapses the strict tail to e or gives opposing strict prefixes, both ruled out by Definition 28.14. The strict-right case is symmetric.

The displayed equivalence for the same branch is exactly the prefix classifier exactness theorem Definition 53.34 after unfolding $\mathsf{same_u}$. Relation stability, reflexivity, and transitivity are inherited from Definition 52.4, while antisymmetry is the same exactness field of Definition 53.34.

For strict-comparison transitivity, use Definition 54.17. It applies the concrete strict unary-prefix transitivity field to the strict-left branch and the same argument with reversed endpoints to the strict-right branch. These fields repack into the concrete $\mathsf{TotalOrder}^{\uparrow}$ law package. □

Theorem 54.9 (Concrete unary prefix total-order comparison partition)**.** *For carried unary histories h, k in* $\mathsf{Carrier_u}$*, exactly one of the three concrete branches*

$$\mathsf{NatUnaryStrictPrefix}(h, k), \qquad \rho_{\mathsf{u}}(h, k), \qquad \mathsf{NatUnaryStrictPrefix}(k, h)$$

supplies the comparison branch. The middle branch is exact:

$$\rho_{\mathsf{u}}(h, k) \iff (\mathsf{PreorderPrefixLE}(h, k) \wedge \mathsf{PreorderPrefixLE}(k, h)).$$

The strict-left and strict-right branches cannot both hold, and neither strict branch is compatible with $\rho_{\mathsf{u}}(h, k)$.

The per-object fields are: traditional target constructive total-order branch partition; BEDC source $\mathsf{UnaryHistory}$*,* $\mathsf{PreorderPrefixLE}$*, and* $\mathsf{NatUnaryStrictPrefix}$ *over* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier ρ_{u}; stability obligations branch totality and incompatibility; ledger selected comparison branch; core theorem this partition; exactness the same-branch equivalence; standard bridge deferred; status* $\mathsf{ProofTarget}$*.*

Proof. Use the total unary prefix comparison of Definition 28.5. For carried histories h and k, it gives a unary continuation in one direction, and the unary tail is then split into empty and nonempty cases.

If the selected tail is nonempty from h to k, the selected branch is $\mathsf{NatUnaryStrictPrefix}(h, k)$. If the selected tail is nonempty from k to h, the selected branch is $\mathsf{NatUnaryStrictPrefix}(k, h)$.

If the selected tail is empty, the continuation collapses to the same-history case. The carrier classifier is exactly the same-history branch, and Definition 53.34 identifies it with the pair of prefix comparisons in both directions.

The two strict branches are incompatible by Definition 28.14. A strict branch together with $\rho_{\mathsf{u}}(h, k)$ would, by Definition 53.34, also provide the opposite prefix comparison; splitting that opposite comparison yields either an empty tail contradicting strictness or the opposing strict branch, again ruled out by Definition 28.14. The strict-right case is symmetric.

These cases give total coverage, the exact middle branch, and pairwise incompatibility, so the three branches form the concrete constructive partition. □

Theorem 54.10 (Concrete unary prefix total-order least upper selector)**.** *For carried unary histories h, k in $\mathsf{Carrier_u}$, there exists a carried history u in $\mathsf{Carrier_u}$ such that*

$$h \leq_{\mathsf{u}} u, \qquad k \leq_{\mathsf{u}} u,$$

and, for every carried history r in $\mathsf{Carrier_u}$,

$$(h \leq_{\mathsf{u}} r \wedge k \leq_{\mathsf{u}} r) \Rightarrow u \leq_{\mathsf{u}} r.$$

The per-object fields are: traditional target finite unary-word maximum/least-upper selector; BEDC source $\mathsf{UnaryHistory}$ *over* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier ρ_{u}; relation $\leq_{\mathsf{u}}$; ledger the selected comparison branch; core theorem this block; exactness least-upper property; standard bridge finite unary prefix maximum; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 54.9 to the carried pair h, k. The proof splits according to the selected branch.

In the strict-left branch $\mathsf{NatUnaryStrictPrefix}(h, k)$, choose $u = k$. Forgetting the nonempty tail gives $h \leq_{\mathsf{u}} k$, while reflexivity from Definition 52.4 gives $k \leq_{\mathsf{u}} k$. If r is any common upper bound of h and k, the second premise already supplies $k \leq_{\mathsf{u}} r$, which is the required leastness clause for the selected u.

In the same branch $\rho_{\mathsf{u}}(h, k)$, choose $u = k$. The exactness row in Definition 54.9 gives $h \leq_{\mathsf{u}} k$ and $k \leq_{\mathsf{u}} h$; the displayed upper-bound clauses again use $h \leq_{\mathsf{u}} k$ and reflexivity of k. For every common upper bound r, the premise $k \leq_{\mathsf{u}} r$ is exactly the required comparison from u to r.

In the strict-right branch $\mathsf{NatUnaryStrictPrefix}(k, h)$, choose $u = h$. The branch supplies $k \leq_{\mathsf{u}} h$ after forgetting strictness, and reflexivity gives $h \leq_{\mathsf{u}} h$. For a common upper bound r, the first premise supplies $h \leq_{\mathsf{u}} r$, so the selected u is below every common upper bound.

These three cases construct the carried selector and verify both upper-bound rows and the least-upper property. □

Theorem 54.11 (Unary prefix least-upper selector successor transport)**.** *For carried unary histories h, k in $\mathsf{Carrier_u}$, suppose u is the selected least upper endpoint supplied by Definition 54.10. Then applying the same unary successor context to all three endpoints transports the selector: the endpoints $\mathsf{E1}(h)$ and $\mathsf{E1}(k)$ have selected least upper endpoint $\mathsf{E1}(u)$, up to* hsame*. More precisely, the transported package exposes*

$$\mathsf{E1}(h) \leq_{\mathsf{u}} \mathsf{E1}(u), \qquad \mathsf{E1}(k) \leq_{\mathsf{u}} \mathsf{E1}(u),$$

and, for every carried unary history r,

$$(\mathsf{E1}(h) \leq_{\mathsf{u}} r \wedge \mathsf{E1}(k) \leq_{\mathsf{u}} r) \Rightarrow \mathsf{E1}(u) \leq_{\mathsf{u}} r,$$

where the successor context is the unary Cont *row determined by* E1*. The construction uses only the concrete prefix preorder,* E1 *context transport, and* hsame *transitivity; it introduces no external endpoint identification.*

Proof. Apply Definition 54.10 to h and k, and keep the same three selector cases as in its proof. The strict-left case selects $u = k$, the same case selects an endpoint hsame-equivalent to either input, and the strict-right case selects $u = h$.

For the upper-bound rows after transport, use prefix context monotonicity. The successor row E1 is a unary left context, so Definition 52.36 carries $h \leq_{\mathsf{u}} u$ to $\mathsf{E1}(h) \leq_{\mathsf{u}} \mathsf{E1}(u)$ and carries $k \leq_{\mathsf{u}} u$ to $\mathsf{E1}(k) \leq_{\mathsf{u}} \mathsf{E1}(u)$. Endpoint-shape carriage is transported by the same Cont row.

It remains to show leastness. In the strict-left case, $u = k$; any transported common upper bound already contains the row $\mathsf{E1}(k) \leq_{\mathsf{u}} r$, which is the required comparison from $\mathsf{E1}(u)$ to r. The strict-right case is symmetric with $u = h$.

In the same case, the exactness row of Definition 54.9 gives $h \leq_{\mathsf{u}} k$ and $k \leq_{\mathsf{u}} h$. Transporting those comparisons through the same successor context and composing them with the two upper-bound premises by the transitivity row of Definition 52.4 gives the comparison from $\mathsf{E1}(u)$ to r. Any endpoint-name mismatch is absorbed by symmetry and transitivity of hsame, so the transported selected endpoint is $\mathsf{E1}(u)$ up to hsame. □

Theorem 54.12 (Concrete unary prefix total-order greatest lower selector)**.** *For carried unary histories h, k in* $\mathsf{Carrier_u}$*, there exists a carried history l in* $\mathsf{Carrier_u}$ *such that*

$$l \leq_{\mathsf{u}} h, \qquad l \leq_{\mathsf{u}} k,$$

and, for every carried history r in $\mathsf{Carrier_u}$*,*

$$(r \leq_{\mathsf{u}} h \wedge r \leq_{\mathsf{u}} k) \Rightarrow r \leq_{\mathsf{u}} l.$$

The per-object fields are: traditional target finite unary-word minimum/greatest-lower selector; BEDC source UnaryHistory *over* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier ρ_{u}; relation $\leq_{\mathsf{u}}$; ledger selected comparison branch; core theorem this block; exactness greatest-lower property; downstream consumer* $\mathsf{Lattice}^{\uparrow}$ *meet and unary interval certificates; status* ProofTarget*.*

Proof. Apply Definition 54.9 to the carried pair h, k. The proof splits according to the selected branch.

In the strict-left branch $\mathsf{NatUnaryStrictPrefix}(h, k)$, choose $l = h$. Reflexivity from Definition 52.4 gives $h \leq_{\mathsf{u}} h$, and forgetting the nonempty tail gives $h \leq_{\mathsf{u}} k$. If r is any common lower bound of h and k, the first premise already supplies $r \leq_{\mathsf{u}} h$, which is the required greatestness clause for the selected l.

In the same branch $\rho_{\mathsf{u}}(h, k)$, choose $l = h$. The exactness row in Definition 54.9 gives $h \leq_{\mathsf{u}} k$ and $k \leq_{\mathsf{u}} h$; the displayed lower-bound clauses use reflexivity of h and the comparison $h \leq_{\mathsf{u}} k$. For every common lower bound r, the premise $r \leq_{\mathsf{u}} h$ is exactly the required comparison from r to l.

In the strict-right branch $\mathsf{NatUnaryStrictPrefix}(k, h)$, choose $l = k$. Forgetting strictness gives $k \leq_{\mathsf{u}} h$, and reflexivity gives $k \leq_{\mathsf{u}} k$. For a common lower bound r, the second premise supplies $r \leq_{\mathsf{u}} k$, so every common lower bound lies below the selected l.

These three cases construct the carried selector and verify both lower-bound rows and the greatest-lower property. □

Theorem 54.13 (Concrete unary prefix total-order branch transport)**.** *For carried unary histories h, h', k, k' in $\mathsf{Carrier_u}$, suppose $\rho_\mathsf{u}(h, h')$ and $\rho_\mathsf{u}(k, k')$. Then classifier replacement transports the three concrete comparison branches exactly:*

$$\mathsf{NatUnaryStrictPrefix}(h, k) \iff \mathsf{NatUnaryStrictPrefix}(h', k'),$$
$$\rho_\mathsf{u}(h, k) \iff \rho_\mathsf{u}(h', k'),$$
$$\mathsf{NatUnaryStrictPrefix}(k, h) \iff \mathsf{NatUnaryStrictPrefix}(k', h').$$

The per-object fields are: traditional target constructive total-order branch transport; BEDC source $\mathsf{UnaryHistory}$ *plus* $\mathsf{NatUnaryStrictPrefix}$ *and ρ_u; carrier* $\mathsf{Carrier_u}$*; classifier ρ_u; stability obligation comparison transport; ledger endpoint replacement witnesses; core theorem this block; exactness the three displayed iff rows; standard bridge finite unary word total order; status* $\mathsf{ProofTarget}$.

Proof. Unpack the two classifier witnesses. The witness $\rho_\mathsf{u}(h, h')$ gives the endpoint comparison $\mathsf{hsame}(h, h')$, and $\rho_\mathsf{u}(k, k')$ gives $\mathsf{hsame}(k, k')$.

For the strict-left row, the forward direction applies Definition 28.24 to the strict witness from h to k with endpoint witnesses $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$.

The reverse direction uses the same transport theorem with the symmetric endpoint witnesses $\mathsf{hsame}(h', h)$ and $\mathsf{hsame}(k', k)$.

For the classifier row, compose the endpoint comparison $\mathsf{hsame}(h, h')$, the middle comparison extracted from $\rho_\mathsf{u}(h, k)$, and the symmetric form of $\mathsf{hsame}(k, k')$. This gives $\mathsf{hsame}(h', k')$, which repackages with the transported carrier witnesses as $\rho_\mathsf{u}(h', k')$.

The reverse classifier direction is the same composition with the two endpoint replacement witnesses reversed. Symmetry and transitivity of the carrier classifier recover $\rho_\mathsf{u}(h, k)$.

For the strict-right row, read the strict witness as a strict unary prefix from k to h. Apply the strict-prefix transport theorem with endpoints $\mathsf{hsame}(k, k')$ and $\mathsf{hsame}(h, h')$.

The reverse strict-right direction again uses the symmetric endpoint witnesses. Thus all three comparison branches are preserved and reflected under concrete classifier replacement. □

Definition 54.14 (Concrete unary successor comparison package)**.** *For a carried unary history h in $\mathsf{Carrier_u}$, the concrete successor comparison package is the pair $(h, \mathsf{E1}(h))$ over the carrier $\mathsf{Carrier_u}$, classifier ρ_u, and order $\leq_\mathsf{u}$ from Definition 54.7. Its selected comparison tail is the visible one-step ledger element $\mathsf{E1(e)}$, so the branch data are fixed as*

$$\mathsf{NatUnaryStrictPrefix}(h, \mathsf{E1}(h)), \qquad \rho_\mathsf{u}(h, \mathsf{E1}(h)), \qquad \mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(h), h).$$

The package records the first branch as the selected concrete successor branch and retains the other two branches as explicit exclusion targets.

Theorem 54.15 (Concrete unary successor total-order strict branch)**.** *For every h with $\mathsf{Carrier_u}(h)$, the successor target $\mathsf{E1}(h)$ is carried by $\mathsf{Carrier_u}$ and the concrete successor comparison package of Definition 54.14 satisfies*

$$\mathsf{NatUnaryStrictPrefix}(h, \mathsf{E1}(h)).$$

Moreover,

$$\rho_\mathsf{u}(h, \mathsf{E1}(h)) \to \mathsf{False} \qquad \textit{and} \qquad \mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(h), h) \to \mathsf{False}.$$

The per-object fields are: traditional target one-step successor separation; BEDC source BHist, $\mathsf{UnaryHistory}$, $\mathsf{NatUnaryStrictPrefix}$, *and* $\mathsf{PreorderPrefixLE}$*; carrier* $\mathsf{Carrier_u}$*; classifier ρ_u; ledger tail* $\mathsf{E1(e)}$*; core theorem this successor strict-branch law; exactness the exclusion of same and reverse-strict branches; status* $\mathsf{ProofTarget}$.

Proof. Unary closure of $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$ under the $\mathsf{E1}$ constructor gives the carried target $\mathsf{E1}(h)$.

Apply Definition 28.13 to the unary witness for h. The selected tail is $\mathsf{E1}(\mathsf{e})$, so this produces $\mathsf{NatUnaryStrictPrefix}(h, \mathsf{E1}(h))$ for the concrete successor pair.

Now instantiate Definition 54.9 at h and $\mathsf{E1}(h)$. Its incompatibility clause says that the selected strict-left branch cannot coexist with the same branch $\rho_\mathsf{u}(h, \mathsf{E1}(h))$.

The same partition also excludes coexistence of the selected strict-left branch with the strict-right branch $\mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(h), h)$. Therefore any witness of either excluded branch yields False. □

Theorem 54.16 (Concrete unary successor strict branch under left context)**.** *For carried unary histories p, h in $\mathsf{Carrier_u}$, both contextual endpoints*

$$\mathsf{append}(p, h) \qquad \textit{and} \qquad \mathsf{append}(p, \mathsf{E1}(h))$$

are carried by $\mathsf{Carrier_u}$, and the contextual successor pair satisfies

$$\mathsf{NatUnaryStrictPrefix}(\mathsf{append}(p, h), \mathsf{append}(p, \mathsf{E1}(h))).$$

Moreover,

$$\rho_\mathsf{u}(\mathsf{append}(p, h), \mathsf{append}(p, \mathsf{E1}(h))) \to \mathsf{False} \qquad \textit{and} \qquad \mathsf{NatUnaryStrictPrefix}(\mathsf{append}(p, \mathsf{E1}(h)), \mathsf{append}(p, h)) \to \mathsf{Fal}$$

The per-object fields are: traditional target context-stable one-step successor separation; BEDC source BHist*,* $\mathsf{UnaryHistory}$*,* append*,* $\mathsf{NatUnaryStrictPrefix}$*, and* $\mathsf{PreorderPrefixLE}$*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_u*; ledger left context p plus tail* $\mathsf{E1}(\mathsf{e})$*; core theorem this contextual successor strict-branch law; exactness the exclusion of same and reverse-strict branches; status* $\mathsf{ProofTarget}$*.*

Proof. Unary closure of $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$ under append gives the carried endpoint $\mathsf{append}(p, h)$. Unary closure of h under $\mathsf{E1}$ first gives $\mathsf{E1}(h)$, and append closure with the same left context p gives the carried endpoint $\mathsf{append}(p, \mathsf{E1}(h))$.

Apply Definition 28.13 to the unary witness for h, obtaining the strict successor tail $\mathsf{E1}(\mathsf{e})$ from h to $\mathsf{E1}(h)$. Continuation associativity transports this same nonempty tail through the left context p, producing the displayed $\mathsf{NatUnaryStrictPrefix}$ witness between the contextual endpoints.

Instantiate Definition 54.9 at $\mathsf{append}(p, h)$ and $\mathsf{append}(p, \mathsf{E1}(h))$. Its incompatibility clauses exclude coexistence of the selected strict-left contextual branch with the same branch $\rho_\mathsf{u}(\mathsf{append}(p, h), \mathsf{append}(p, \mathsf{E1}(h)$ and with the reverse strict branch $\mathsf{NatUnaryStrictPrefix}(\mathsf{append}(p, \mathsf{E1}(h)), \mathsf{append}(p, h))$. Therefore either excluded branch yields False. □

Theorem 54.17 (Concrete unary strict branch transitivity)**.** *For carried unary histories h, k, l in $\mathsf{Carrier_u}$, strict-left comparison is transitive:*

$$\mathsf{NatUnaryStrictPrefix}(h, k) \wedge \mathsf{NatUnaryStrictPrefix}(k, l) \to \mathsf{NatUnaryStrictPrefix}(h, l).$$

The strict-right branch is transitive in the same concrete carrier:

$$\mathsf{NatUnaryStrictPrefix}(k, h) \wedge \mathsf{NatUnaryStrictPrefix}(l, k) \to \mathsf{NatUnaryStrictPrefix}(l, h).$$

Consequently $\mathsf{lt_u}$ *and* $\mathsf{gt_u}$ *are transitive on the concrete unary prefix* $\mathsf{TotalOrder}^{\uparrow}$ *instance.*

Proof. Unfold the strict-left branch of Definition 54.7. The two hypotheses are exactly a strict unary prefix from h to k and a strict unary prefix from k to l. Apply Definition 28.23; its composite tail witness gives $\mathsf{NatUnaryStrictPrefix}(h, l)$.

For strict-right comparison, unfold $\mathsf{gt_u}(x, y) := \mathsf{NatUnaryStrictPrefix}(y, x)$. The hypotheses become strict unary prefixes from l to k and from k to h. Applying the same composite-tail transitivity theorem yields $\mathsf{NatUnaryStrictPrefix}(l, h)$.

The carrier witnesses for h, k, and l are not consumed by the strict-prefix composition itself; they keep the conclusion inside the concrete $\mathsf{Carrier_u}$ total-order package. The two displayed implications are therefore precisely the transitivity fields for $\mathsf{lt_u}$ and $\mathsf{gt_u}$. □

Theorem 54.18 (Concrete unary total-order branch context equivalence)**.** *Let h, k, ℓ, r be unary histories in $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$, and set*

$$H := \mathsf{append}(\ell, \mathsf{append}(h, r)), \qquad K := \mathsf{append}(\ell, \mathsf{append}(k, r)).$$

Then H and K are carried unary histories, and the three comparison branches of Definition 54.7 are invariant under this common two-sided unary context:

1. $\mathsf{lt_u}(H, K) \iff \mathsf{lt_u}(h, k)$.
2. $\mathsf{same_u}(H, K) \iff \mathsf{same_u}(h, k)$.
3. $\mathsf{gt_u}(H, K) \iff \mathsf{gt_u}(h, k)$.

Proof. Unary closure under append first gives $\mathsf{UnaryHistory}(\mathsf{append}(h, r))$ and $\mathsf{UnaryHistory}(\mathsf{append}(k, r))$. Applying the same closure once more with the unary left context ℓ gives the carrier facts for H and K.

For the same branch, unfold $\mathsf{same_u} := \rho_\mathsf{u}$. By Definition 53.34, the classifier on carried unary histories is equivalent to the pair of opposing prefix comparisons. Apply Definition 52.42 to the comparison $H \leq_\mathsf{u} K$ and again to $K \leq_\mathsf{u} H$; the two applications identify the contextual pair of prefix comparisons exactly with $h \leq_\mathsf{u} k$ and $k \leq_\mathsf{u} h$. A second use of Definition 53.34 repacks these comparisons as $\rho_\mathsf{u}(h, k)$.

For the strict-left branch, unfold $\mathsf{lt_u}$ as $\mathsf{NatUnaryStrictPrefix}$. In the forward direction from $\mathsf{NatUnaryStrictPrefix}(h, k)$, unpack the nonempty unary tail from h to k. Right-context preservation composes this tail past r by continuation associativity and unary-tail commutativity; left-context preservation then places ℓ outside both endpoints. The tail remains the same nonempty unary tail, so the result is $\mathsf{NatUnaryStrictPrefix}(H, K)$.

For the reverse strict-left direction, unpack the nonempty unary tail from H to K. Its underlying prefix comparison cancels through Definition 52.41, leaving a prefix comparison from h to k. If the recovered central tail were empty, append-empty exactness would collapse h and k, hence also H and K after the common unary context. Comparing that collapse with the original strict contextual tail would force the original tail empty, contradicting its strictness. Thus the central prefix tail is nonempty and gives $\mathsf{NatUnaryStrictPrefix}(h, k)$.

The strict-right branch is the same argument with endpoints reversed, since $\mathsf{gt_u}(x, y)$ unfolds to $\mathsf{NatUnaryStrictPrefix}(y, x)$. Applying the strict-left preservation and reflection just proved to the ordered pair (k, h) gives the displayed equivalence for $\mathsf{gt_u}$. □

Theorem 54.19 (Concrete unary successor strict branch under two-sided context)**.** *For carried unary histories ℓ, h, r in $\mathsf{Carrier_u}$, set*

$$H := \mathsf{append}(\ell, \mathsf{append}(h, r)), \qquad K := \mathsf{append}(\ell, \mathsf{append}(\mathsf{E1}(h), r)).$$

Then H and K are carried by $\mathsf{Carrier_u}$*, and the contextual successor pair satisfies*

$$\mathsf{lt_u}(H, K).$$

Moreover,

$$\mathsf{same_u}(H, K) \to \mathsf{False} \qquad \textit{and} \qquad \mathsf{gt_u}(H, K) \to \mathsf{False}.$$

The per-object fields are: traditional target context-stable one-step successor separation; BEDC source BHist*,* $\mathsf{UnaryHistory}$*,* append*,* $\mathsf{E1}$*,* $\mathsf{lt_u}$*,* $\mathsf{same_u}$*, and* $\mathsf{gt_u}$*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_u*; ledger common two-sided context* (ℓ, r) *plus tail* $\mathsf{E1(e)}$*; core theorem this two-sided contextual successor strict-branch law; exactness the exclusion of same and reverse-strict branches; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 54.15 to the carried history h. This gives the carried successor $\mathsf{E1}(h)$, the base branch $\mathsf{lt_u}(h, \mathsf{E1}(h))$, and the two base exclusions $\mathsf{same_u}(h, \mathsf{E1}(h)) \to \mathsf{False}$ and $\mathsf{gt_u}(h, \mathsf{E1}(h)) \to \mathsf{False}$.

Instantiate Definition 54.18 with the carried quadruple $h, \mathsf{E1}(h), \ell, r$. Its carrier conclusion gives that the displayed H and K are carried unary histories.

Use the right-to-left direction of the strict-left branch equivalence from the same theorem. Transporting the base successor branch $\mathsf{lt_u}(h, \mathsf{E1}(h))$ through the common context (ℓ, r) gives $\mathsf{lt_u}(H, K)$.

For a same-branch witness at the contextual endpoints, use the left-to-right direction of the same-branch equivalence to obtain $\mathsf{same_u}(h, \mathsf{E1}(h))$. The base same-branch exclusion then returns False.

For a reverse strict-branch witness at the contextual endpoints, use the left-to-right direction of the strict-right branch equivalence to obtain $\mathsf{gt_u}(h, \mathsf{E1}(h))$. The base reverse-branch exclusion then returns False. □

Theorem 54.20 (Concrete unary prefix $\mathsf{TotalOrder}^\uparrow$ naming certificate)**.** *The tuple of Definition 54.7 supplies the source, pattern, classifier, stability, ledger, and exactness rows for a* $\mathsf{NameCert}$ *licensing* $\mathsf{TotalOrder}^\uparrow$ *over* $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$*, classifier* ρ_u*, order* $\leq_\mathsf{u}$*, and comparison branches* $\mathsf{NatUnaryStrictPrefix}(h, k)$*,* $\rho_\mathsf{u}(h, k)$*, and* $\mathsf{NatUnaryStrictPrefix}(k, h)$*.*

Proof. The source and inherited order rows are the certified unary-prefix $\mathsf{POSet}^\uparrow$ package. Apply Definition 53.36; it supplies the BHist source, prefix pattern, classifier, stability, ledger, and exactness rows used by the total-order carrier.

The classifier and exactness rows are fixed by Definition 53.34: on carried unary histories, $\rho_\mathsf{u}(h, k)$ is equivalent to the two opposing prefix comparisons $h \leq_\mathsf{u} k$ and $k \leq_\mathsf{u} h$.

The stability row is the concrete total-order law package of Definition 54.8. It supplies the strict-to-apartness obligations, relation stability, constructive total comparison, comparison stability, and strict-comparison transitivity for the three displayed branches. The strict-comparison transitivity component is the concrete branch law of Definition 54.17, and comparison stability is the branch transport theorem of Definition 54.13.

The ledger row is the selected branch certificate. By Definition 54.9, each pair of carried unary histories chooses exactly one of the strict-left, same, and strict-right branches, and the same branch is the exact classifier branch.

Repack these rows as the public $\mathsf{NameCert}$: the source is $\mathsf{UnaryHistory}$, the pattern is the three-way unary-prefix comparison, the classifier is ρ_u, the stability data are the cited law fields, the ledger is the selected comparison branch, and exactness is the middle-branch equivalence. □

The per-object template is: traditional target concrete unary constructive total order; BEDC source unary BHist; carrier $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$; classifier ρ_u; order $\leq_\mathsf{u}$; pattern strict-left, same, and strict-right unary-prefix comparison; stability obligations from Definition 54.20; ledger the selected comparison branch certificate; exactness same branch iff opposing prefix comparisons; status $\mathsf{ProofTarget}$.

The $\mathsf{TotalOrder}^\uparrow$ schema is represented by this concrete BHist carrier and explicit comparison data, giving later Lean formalization a BHist-anchored target rather than an abstract parameter schema.

Theorem 54.21 (Concrete unary prefix total order supplies arithmetic $\mathsf{Order}^\uparrow$)**.** *The tuple of Definition 54.7, read through the naming certificate of Definition 54.20, supplies the L7* $\mathsf{Order}^\uparrow$ *comparison interface on unary histories. The carrier is* $\mathsf{Carrier_u} = \mathsf{UnaryHistory}$*, the classifier is* ρ_u*, the preorder relation is* $\leq_\mathsf{u}$*, and the three comparison branches are* $\mathsf{NatUnaryStrictPrefix}(h, k)$*,* $\rho_\mathsf{u}(h, k)$*, and* $\mathsf{NatUnaryStrictPrefix}(k, h)$*.*

The reflexivity, transitivity, and antisymmetry rows are exactly the concrete prefix $\mathsf{POSet}^\uparrow$ *rows. The constructive total comparison row is the branch partition of Definition 54.8 and Definition 54.9. The classifier exactness row is the middle-branch equivalence of Definition 53.34: on carried unary histories,* $\rho_\mathsf{u}(h, k)$ *holds exactly when* $h \leq_\mathsf{u} k$ *and* $k \leq_\mathsf{u} h$*. Thus the stronger concrete* $\mathsf{TotalOrder}^\uparrow$ *naming certificate supplies the arithmetic* $\mathsf{Order}^\uparrow$ *consumer without introducing a second order relation.*

Proof. Start with the concrete prefix $\mathsf{POSet}^\uparrow$ certificate supplied by Definition 53.36. Its carrier is $\mathsf{UnaryHistory}$, its classifier is ρ_u, and its relation is the concrete prefix preorder $\leq_\mathsf{u}$. The reflexive, transitive, and antisymmetric order laws are therefore already available as the order rows consumed by $\mathsf{Order}^\uparrow$.

The comparison ledger is supplied by the total-order package. Apply Definition 54.8 for the total comparison law and Definition 54.9 for the exclusive strict-left, same, and strict-right branch partition. These are the three branches displayed in the statement.

The branch data are stable under the BEDC contexts used by the concrete certificate. Apply Definition 54.13 to transport the selected branch through common contexts, and apply Definition 54.18 when the consumer needs the two-sided contextual reading.

It remains to identify the same branch with the preorder classifier. On carried unary histories, Definition 53.34 says that $\rho_\mathsf{u}(h, k)$ is equivalent to the two opposing comparisons $h \leq_\mathsf{u} k$ and $k \leq_\mathsf{u} h$. This is exactly the classifier exactness row required by the $\mathsf{Order}^\uparrow$ consumer.

Finally, read these rows through Definition 54.20. The public consumer uses the existing $\mathsf{UnaryHistory}$ carrier, ρ_u classifier, and $\leq_\mathsf{u}$ preorder, together with the inherited strict-left and strict-right branch names. This is a reuse of the concrete prefix order already certified above; it is not a new primitive order and it does not assert classical trichotomy. □

Theorem 54.22 (Concrete unary prefix total-order finite minimum selector)**.** *Let* L *be a nonempty finite spine of histories in the finite-list helper of Definition 36.1. Assume every entry of* L *is carried by* $\mathsf{Carrier_u}$ *for the concrete* $\mathsf{TotalOrder}^\uparrow$ *certificate of Definition 54.20. Then there is an entry* $m \in L$ *such that, for every entry* $x \in L$*,*

$$m \leq_\mathsf{u} x.$$

The per-object fields are: traditional target finite-list minimum extraction in a total order; BEDC source finite spines of carried unary histories; carrier $\mathsf{Carrier_u}$*; classifier* ρ_u*; relation* $\leq_\mathsf{u}$*; stability obligations total comparison, reflexivity, and transitivity; ledger the tail-minimum witness*

and the selected comparison branch at each cons step; core theorem this finite minimum selector; exactness membership plus the displayed lower-bound row; standard bridge finite unary-word total order; status ProofTarget.

Proof. Induct on the nonempty finite spine L.

For the singleton spine $L = [a]$, choose $m = a$. The membership clause is immediate, and the only entry to check is a itself. Reflexivity from Definition 52.4 gives $a \leq_{\mathsf{u}} a$.

For the cons step, write $L = a :: T$ with T nonempty and all its entries carried by $\mathsf{Carrier}_{\mathsf{u}}$. By the induction hypothesis applied to T, choose $m_T \in T$ such that $m_T \leq_{\mathsf{u}} x$ for every $x \in T$. Apply Definition 54.9 to the carried pair a, m_T.

In the strict-left branch $\mathsf{NatUnaryStrictPrefix}(a, m_T)$, choose $m = a$. By Definition 28.10 and Definition 54.7, forgetting the nonempty-tail condition gives $a \leq_{\mathsf{u}} m_T$. For $x = a$, reflexivity gives $a \leq_{\mathsf{u}} a$. For $x \in T$, compose $a \leq_{\mathsf{u}} m_T$ with the induction-hypothesis comparison $m_T \leq_{\mathsf{u}} x$ using Definition 52.28. Thus the head a is a lower bound for every entry of L.

In the same branch $\rho_{\mathsf{u}}(a, m_T)$, choose $m = a$. The exact middle-branch row of Definition 54.9 gives $a \leq_{\mathsf{u}} m_T$ and $m_T \leq_{\mathsf{u}} a$. The entry a is below itself by reflexivity, and it is below every tail entry by transitivity through m_T, exactly as in the strict-left case.

In the strict-right branch $\mathsf{NatUnaryStrictPrefix}(m_T, a)$, choose $m = m_T$. Its membership in L follows from $m_T \in T$. Forgetting the nonempty-tail condition gives $m_T \leq_{\mathsf{u}} a$, which handles the head entry. Every tail entry is handled by the induction hypothesis. Hence the selected tail minimum is below every entry of L.

All comparison branches construct an entry of L satisfying the displayed lower-bound condition, so the finite minimum selector exists for every nonempty finite spine of carried concrete $\mathsf{TotalOrder}^{\uparrow}$ elements. □

Theorem 54.23 (Concrete unary prefix total-order finite maximum index selector)**.** *Let L be a nonempty finite spine of histories in the finite-list helper of Definition 36.1. Assume every entry of L is carried by $\mathsf{Carrier}_{\mathsf{u}}$ for the concrete $\mathsf{TotalOrder}^{\uparrow}$ certificate of Definition 54.20. Then there are a finite-spine membership witness i and a carried entry $M = L_i$ such that, for every entry $x \in L$,*

$$x \leq_{\mathsf{u}} M.$$

The per-object fields are: traditional target finite-list maximum extraction in a total order; BEDC source finite spines of carried unary histories; carrier $\mathsf{Carrier}_{\mathsf{u}}$; classifier ρ_{u}; relation $\leq_{\mathsf{u}}$; stability obligations total comparison, reflexivity, and transitivity; ledger the tail-maximum membership witness and the selected comparison branch at each cons step; core theorem this finite maximum selector; exactness the displayed membership witness plus upper-bound row; standard bridge finite unary-word total order; status ProofTarget.

Proof. Induct on the nonempty finite spine L.

For the singleton spine $L = [a]$, choose the unique membership position and set $M = a$. The only entry to check is a itself, and reflexivity from Definition 52.4 gives $a \leq_{\mathsf{u}} a$.

For the cons step, write $L = a :: T$ with T nonempty and all its entries carried by $\mathsf{Carrier}_{\mathsf{u}}$. By the induction hypothesis applied to T, choose a tail membership witness i_T and an entry $M_T \in T$ such that $x \leq_{\mathsf{u}} M_T$ for every $x \in T$. Apply Definition 54.9 to the carried pair a, M_T.

In the strict-left branch $\mathsf{NatUnaryStrictPrefix}(a, M_T)$, choose the same tail membership witness, viewed as a membership witness in $a :: T$, and set $M = M_T$. By Definition 28.10 and Definition 54.7, forgetting the nonempty-tail condition gives $a \leq_{\mathsf{u}} M_T$, which handles the head entry. Every tail entry is below M_T by the induction hypothesis.

In the same branch $\rho_{\mathsf{u}}(a, M_T)$, again choose the tail membership witness and set $M = M_T$. The exact middle-branch row of Definition 54.9 gives $a \leq_{\mathsf{u}} M_T$ and $M_T \leq_{\mathsf{u}} a$. The first comparison handles the head entry, and every tail entry is handled by the induction hypothesis.

In the strict-right branch $\mathsf{NatUnaryStrictPrefix}(M_T, a)$, choose the head membership witness and set $M = a$. Forgetting strictness gives $M_T \leq_{\mathsf{u}} a$. For the head entry, reflexivity gives $a \leq_{\mathsf{u}} a$. For a tail entry $x \in T$, the induction hypothesis gives $x \leq_{\mathsf{u}} M_T$; composing this with $M_T \leq_{\mathsf{u}} a$ by Definition 52.28 gives $x \leq_{\mathsf{u}} a$.

All comparison branches construct a located entry of L above every entry of L, so the finite maximum selector exists for every nonempty finite spine of carried concrete $\mathsf{TotalOrder}^{\uparrow}$ elements. □

Definition 54.24 (Concrete unary finite minimum witness)**.** *For a finite BEDC-history spine* $L : \mathsf{ListCarrier}(\mathsf{BHist})$*, a visible readback* m *is a concrete unary finite minimum witness when*

$$\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m) := m \in L \wedge \forall x.\ x \in L \to m \leq_{\mathsf{u}} x.$$

The carrier side condition is supplied by the finite-carrier list hypothesis used with the witness.

Theorem 54.25 (Concrete unary prefix finite minimum classifier uniqueness)**.** *Let* $L : \mathsf{ListCarrier}(\mathsf{BHist})$ *be a nonempty finite spine in the helper of Definition 36.1. Assume every entry of* L *is carried by* $\mathsf{Carrier}_{\mathsf{u}}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier}_{\mathsf{u}}(x).$$

If m *and* n *satisfy* $\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m)$ *and* $\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, n)$*, then*

$$\rho_{\mathsf{u}}(m, n).$$

The per-object fields are: traditional target finite-list minimum witness uniqueness in a total order; BEDC source finite spines of carried unary histories; carrier $\mathsf{Carrier}_{\mathsf{u}}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability obligations finite-list lower-bound readout and inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry; ledger the two minimum witness records and the two extracted comparisons; core theorem this block; exactness classifier equality of competing minimum readbacks; standard bridge finite unary-word minimum uniqueness; status* $\mathsf{ProofTarget}$*.*

Proof. Unpack the two finite minimum witness records of Definition 54.24. The first supplies $m \in L$ and the lower-bound row $\forall x.\ x \in L \to m \leq_{\mathsf{u}} x$; the second supplies $n \in L$ and the lower-bound row $\forall x.\ x \in L \to n \leq_{\mathsf{u}} x$.

From the entry-carrier hypothesis, the membership rows give carrier witnesses for m and n. Instantiate the first lower-bound row at the entry n; this gives $m \leq_{\mathsf{u}} n$. Instantiate the second lower-bound row at the entry m; this gives $n \leq_{\mathsf{u}} m$.

By Definition 54.7, these comparisons are the concrete prefix relation of the unary total-order package. The exact middle-branch row of Definition 54.9, equivalently the $\mathsf{POSet}^{\uparrow}$ classifier exactness of Definition 53.34 in this notation, sends the carried opposing comparisons to $\rho_{\mathsf{u}}(m, n)$.

The strict-left and strict-right branches of the total comparison ledger are therefore excluded by the incompatibility clauses of Definition 54.9. Together with the existence selector of Definition 54.22, this gives a classifier-stable finite minimum readback for every nonempty finite carried spine. □

Definition 54.26 (Concrete unary finite maximum witness)**.** *For a finite BEDC-history spine* $L : \mathsf{ListCarrier}(\mathsf{BHist})$*, a visible readback* M *is a concrete unary finite maximum witness when*

$$\mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M) := M \in L \wedge \forall x.\ x \in L \to x \leq_{\mathsf{u}} M.$$

The carrier side condition is supplied by the finite-carrier list hypothesis used with the witness.

Theorem 54.27 (Concrete unary prefix finite maximum classifier uniqueness)**.** *Let* $L : \mathsf{ListCarrier}(\mathsf{BHist})$ *be a nonempty finite spine in the helper of Definition 36.1. Assume every entry of* L *is carried by* $\mathsf{Carrier}_{\mathsf{u}}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier}_{\mathsf{u}}(x).$$

If M *and* N *satisfy* $\mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M)$ *and* $\mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, N)$*, then*

$$\rho_{\mathsf{u}}(M, N).$$

The per-object fields are: traditional target finite-list maximum witness uniqueness in a total order; BEDC source finite spines of carried unary histories; carrier $\mathsf{Carrier}_{\mathsf{u}}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability obligations finite-list upper-bound readout and inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry; ledger the two maximum witness records and the two extracted comparisons; core theorem this block; exactness classifier equality of competing maximum readbacks; standard bridge finite unary-word maximum uniqueness; status* $\mathsf{ProofTarget}$*.*

Proof. Unpack the two finite maximum witness records of Definition 54.26. The first supplies $M \in L$ and the upper-bound row $\forall x.\ x \in L \to x \leq_{\mathsf{u}} M$; the second supplies $N \in L$ and the upper-bound row $\forall x.\ x \in L \to x \leq_{\mathsf{u}} N$.

From the entry-carrier hypothesis, the membership rows give carrier witnesses for M and N. Instantiate the first upper-bound row at the entry N; this gives $N \leq_{\mathsf{u}} M$. Instantiate the second upper-bound row at the entry M; this gives $M \leq_{\mathsf{u}} N$.

By Definition 54.7, these comparisons are the concrete prefix relation of the unary total-order package. The exact middle-branch row of Definition 54.9, equivalently the $\mathsf{POSet}^{\uparrow}$ classifier exactness of Definition 53.34 in this notation, sends the carried opposing comparisons to $\rho_{\mathsf{u}}(M, N)$.

The strict-left and strict-right branches of the total comparison ledger are therefore excluded by the incompatibility clauses of Definition 54.9. Together with the existence selector of Definition 54.23, this gives a classifier-stable finite maximum readback for every nonempty finite carried spine. □

Theorem 54.28 (Concrete unary prefix finite extrema certificate)**.** *Let* $L : \mathsf{ListCarrier}(\mathsf{BHist})$ *be a nonempty finite BEDC-history spine in the helper of Definition 36.1. Assume every entry of* L *is carried by* $\mathsf{Carrier}_{\mathsf{u}}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier}_{\mathsf{u}}(x).$$

Then there are histories m *and* M *such that*

$$\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m) \wedge \mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M)$$

and the selected readbacks are classifier-stable: every other minimum witness m' *satisfying* $\mathsf{FiniteMinimumWitness}_{\mathsf{u}}($ *obeys* $\rho_{\mathsf{u}}(m, m')$*, and every other maximum witness* M' *satisfying* $\mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M')$ *obeys* $\rho_{\mathsf{u}}(M, M')$*.*

The per-object fields are: traditional target finite extrema in a constructive total order; BEDC source finite ListCarrier *spines and unary* $\mathsf{TotalOrder}^{\uparrow}$*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; stability selector existence plus classifier uniqueness; ledger the selected minimum and maximum witnesses; core theorem this finite extrema certificate; exactness classifier uniqueness of extrema; standard bridge finite unary-word extrema; status* ProofTarget*.*

Proof. Apply Definition 54.22 to the nonempty carried spine L. This gives an entry $m \in L$ whose lower-bound row is exactly $\mathsf{FiniteMinimumWitness_u}(L, m)$ after repacking through Definition 54.24.

Apply Definition 54.23 to the same carried spine. Its membership index and selected entry give a history $M \in L$ whose upper-bound row is exactly $\mathsf{FiniteMaximumWitness_u}(L, M)$ after repacking through Definition 54.26.

For any competing minimum witness m', apply Definition 54.25 to the two minimum witness records for m and m'. The result is the displayed classifier agreement $\rho_{\mathsf{u}}(m, m')$.

For any competing maximum witness M', apply Definition 54.27 to the two maximum witness records for M and M'. This gives $\rho_{\mathsf{u}}(M, M')$, completing the paired finite-extrema certificate. □

Theorem 54.29 (Concrete unary total-order finite selector RealUp consumer boundary)**.** *The unary finite minimum/maximum selector package of Definition 54.28 can be consumed by later* $\mathsf{Real}^{\uparrow}$ *finite-window comparison rows only as certified selector data and classifier-uniqueness data over* BHist*,* UnaryHistory*,* hsame*, and* Cont*. It identifies finite extrema in carried unary spines and transports the selected readbacks through* ρ_{u}*; it does not provide* $\mathsf{Real}^{\uparrow}$ *trichotomy, open-cover compactness, or quotient order.*

Proof. Project the selected minimum and maximum witnesses from Definition 54.28. Their existence comes from the finite carried spine and the unary total-order comparison rows.

Project classifier uniqueness through Definition 54.25 and Definition 54.27. These rows are the only comparison facts exported to a $\mathsf{Real}^{\uparrow}$ finite-window consumer.

Since the certificate is built over BHist and unary continuation transport, no RealUp quotient carrier, open-cover compactness principle, or constructive trichotomy is introduced. □

54.4 Concrete Singleton History Instance

Definition 54.30 (Total-order singleton history instance)**.** *Let* A_{e} *be the singleton history source of Definition 36.169 at* $a_0 = \mathsf{e}$*. Use the singleton classifier already exposed by the surrounding concrete algebra instances:*

$$\rho_{\mathsf{e}}(h, k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Define the concrete singleton order relation by

$$h \leq_{\perp} k := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

The $h \leq_{\perp} k$ *witness records only that both endpoints lie in the displayed BEDC history singleton; it introduces no abstract carrier, no host order, and no second endpoint classifier.*

Theorem 54.31 (Total-order singleton history laws)**.** *For the concrete package* $(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_{\perp})$ *of Definition 54.30, the relation* $\leq_{\perp}$ *is reflexive on carried histories, transitive on carried histories, antisymmetric under* ρ_{e}*, and total on carried histories:*

$$\forall h.\ A_{\mathsf{e}}(h) \to h \leq_{\perp} h,$$

$$\forall h\, k\, r.\ (h \leq_\perp k) \to (k \leq_\perp r) \to h \leq_\perp r,$$
$$\forall h\, k.\ (h \leq_\perp k) \to (k \leq_\perp h) \to \rho_{\mathsf{e}}(h, k),$$

and

$$\forall h\, k.\ A_{\mathsf{e}}(h) \to A_{\mathsf{e}}(k) \to (h \leq_\perp k) \vee (k \leq_\perp h).$$

Thus the order laws are pinned to the concrete BHist *singleton carrier rather than to a parameterized order surface.*

Proof. Reflexivity unfolds $h \leq_\perp h$ and repeats the displayed carrier witness $A_{\mathsf{e}}(h)$ in both endpoint slots.

For transitivity, unfold the first comparison $h \leq_\perp k$ to read the carrier witness for h, and unfold the second comparison $k \leq_\perp r$ to read the carrier witness for r. Pairing these two facts gives $h \leq_\perp r$.

For antisymmetry, unfold the two comparisons to obtain $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. Since A_{e} is the singleton source at e, these facts identify h with e and k with e. Symmetry and transitivity of hsame compose the two endpoint witnesses into $\mathsf{hsame}(h, k)$, and the two carrier facts repackage the result as $\rho_{\mathsf{e}}(h, k)$.

For totality, the assumptions already give the two endpoint carrier witnesses. Choose the displayed left comparison $h \leq_\perp k$ and pack those witnesses into the definition of $\leq_\perp$. □

Theorem 54.32 (TotalOrderUp singleton schema instantiation)**.** *The concrete data*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \leq_\perp)$$

instantiate the $\mathsf{TotalOrder}^\uparrow$ *schema over the fixed BEDC history carrier* BHist*: the carrier is the singleton predicate A_{e}, the classifier is the singleton endpoint classifier ρ_{e}, the relation is the concrete endpoint-carrier relation $\leq_\perp$, and the law rows are the reflexivity, transitivity, antisymmetry, and totality clauses of Definition 54.31.*

Consequently, the $\mathsf{TotalOrder}^\uparrow$ *proof target has a falsifiable* BHist*-anchored carrier and relation instance. Machine checking can target the displayed singleton package directly, rather than discharging only abstract parameter fields.*

Proof. Read the carrier, classifier, and relation rows from Definition 54.30. They are concrete predicates and relations on BHist, with the carrier pinned to the singleton endpoint e.

Apply Definition 54.31 to obtain the preorder rows, the classifier antisymmetry row, and the total comparison row. These are exactly the order-law fields required by the $\mathsf{TotalOrder}^\uparrow$ schema after the schema variables are specialized to A_{e}, ρ_{e}, and $\leq_\perp$.

The resulting package is therefore not a restatement over fresh carriers or classifiers. Its witnesses are carried by the fixed singleton history source, and every comparison witness is a concrete assertion about the two displayed BHist endpoints. □

Theorem 54.33 (TotalOrder singleton order-tower certificate)**.** *For the singleton history source A_{e} at e and classifier ρ_{e}, the endpoint-carrier relation $\leq_\perp$ of Definition 54.30 and the prefix relation of Definition 54.2 agree on carried endpoints:*

$$A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \to (h \leq_\perp k \iff \mathsf{PreorderPrefixLE}(h, k)).$$

On the same carried endpoints, the classifier is exact for the opposing singleton $\mathsf{POSet}^\uparrow$ *comparisons and for the same branch of the singleton* $\mathsf{TotalOrder}^\uparrow$ *comparison:*

$$\rho_{\mathsf{e}}(h, k) \iff (h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h),$$

and

$$\rho_{\mathsf{e}}(h,k) \iff \mathsf{same}_{\mathsf{e}}(h,k).$$

Thus the endpoint-carrier order, the singleton prefix order, the singleton partial-order exactness row, and the total-order same branch form one BHist *order tower over the same carrier and classifier.*

Proof. Assume $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. Unfolding $h \leq_{\perp} k$ gives exactly the two displayed carrier witnesses, so the forward endpoint-carrier side is immediate. Conversely, the same two carrier witnesses already reconstitute $h \leq_{\perp} k$.

The singleton carrier witnesses identify both endpoints with e: they give $\mathsf{hsame}(h,\mathsf{e})$ and $\mathsf{hsame}(k,\mathsf{e})$. Symmetry and transitivity of hsame therefore give $\mathsf{hsame}(h,k)$, and applying PreorderPrefixLE_of_hsa yields $\mathsf{PreorderPrefixLE}(h,k)$. This proves the prefix direction needed to compare the two displayed relations on carried endpoints.

For the partial-order row, apply Definition 53.12. Its exactness clause identifies $\rho_{\mathsf{e}}(h,k)$ with the two opposing singleton prefix comparisons $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$.

For the total-order row, unfold the comparison branches of Definition 54.2: the same branch is ρ_{e}, while the strict branches are empty. The law package Definition 54.3 supplies the carried same-branch witness, and Definition 54.83 repackages that witness as the $\mathsf{TotalOrder}^{\uparrow}$ naming certificate. Hence both concrete presentations sit in the same singleton BHist order tower. □

Theorem 54.34 (TotalOrder singleton strict branch exclusion)**.** *On carried endpoints of A_{e}, a singleton* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row for h,k cannot select a strict-left or strict-right branch. More precisely, if $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, then every comparison row in Definition 54.2 factors through the same classifier branch*

$$\rho_{\mathsf{e}}(h,k)$$

and through the endpoint-carrier relation $h \leq_{\perp} k$.

Proof. Unfold Definition 54.2. The carrier witnesses $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ identify both endpoints with e, so the endpoint-carrier relation $h \leq_{\perp} k$ is the same pair of singleton carrier facts read in the order tower certificate.

Apply Definition 53.12. Its opposing guarded prefix exactness clauses identify the two singleton $\mathsf{POSet}^{\uparrow}$ comparisons with the same endpoint carrier witnesses and hence with the same classifier row $\rho_{\mathsf{e}}(h,k)$.

Now apply Definition 54.3. In the singleton history total-order package the strict-left and strict-right branches have no carried row once both endpoints have been identified with e. The only remaining comparison branch is the same branch, and its row is exactly $\rho_{\mathsf{e}}(h,k)$. □

Theorem 54.35 (TotalOrder singleton comparison determinacy)**.** *On carried endpoints of A_{e}, any two singleton total-order comparison witnesses collapse to the same ρ_{e} branch. In symbols, if $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ and two witnesses choose comparison rows for h and k in the singleton* $\mathsf{TotalOrder}^{\uparrow}$ *instance, then both witnesses identify h and k with* e*, unfold $\leq_{\perp}$ to the same carried endpoint facts, and select the same classifier row $\rho_{\mathsf{e}}(h,k)$.*

Proof. Unfold $\leq_{\perp}$ on the carried endpoints. The hypotheses $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ already provide the two carrier facts required by either comparison witness.

Apply Definition 54.31 to read the singleton total-order comparison law. Apply Definition 53.12 to identify the opposing singleton partial-order comparisons with the same classifier row.

Finally use Definition 54.3. In the singleton history instance the strict branches are empty, so every total comparison on carried endpoints is the displayed same branch. Hence the two witnesses determine the same ρ_{e} row. □

Theorem 54.36 (TotalOrder singleton classifier coverage)**.** *Every carried endpoint pair in A_{e} admits the classifier row induced by opposing singleton comparisons. More precisely, if $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$, then the carrier facts unfold $\leq_{\perp}$ in both directions, the singleton $\mathsf{POSet}^{\uparrow}$ comparisons $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$ are available, and their exactness row induces $\rho_{\mathsf{e}}(h, k)$ for the singleton $\mathsf{TotalOrder}^{\uparrow}$ classifier surface.*

Proof. Assume $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$. Unfolding $\leq_{\perp}$ gives the endpoint-carrier comparison rows in both directions, because the singleton relation is built from those two carrier witnesses.

The carrier facts identify both endpoints with e. Apply Definition 53.12 to obtain the opposing singleton partial-order comparisons and their exactness row for ρ_{e}.

Apply Definition 54.31 and Definition 54.3 to place that same row inside the singleton total-order classifier. Thus every carried pair is covered by the classifier induced by the opposing singleton comparisons. □

54.5 OrderUp Interval Collapse

Theorem 54.37 (Unary prefix OrderUp bounded interval collapse)**.** *Let $L : \mathsf{ListCarrier}(\mathsf{BHist})$ be a nonempty finite BEDC-history spine in the helper of Definition 36.1. Assume every entry of L is carried by $\mathsf{Carrier}_{\mathsf{u}}$ for the concrete $\mathsf{TotalOrder}^{\uparrow}$ certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier}_{\mathsf{u}}(x).$$

Let m and M be the finite extrema supplied by Definition 54.28. Then every entry of the finite spine lies in the displayed unary interval:

$$\forall x.\ x \in L \to (m \leq_{\mathsf{u}} x \wedge x \leq_{\mathsf{u}} M).$$

Moreover, if endpoints m' and M' satisfy the same finite-bound rows, namely

$$\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m') \wedge \mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M'),$$

then the endpoints are classified with the selected extrema:

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M').$$

The per-object fields are: traditional target finite bounded interval in unary arithmetic order; BEDC source $\mathsf{UnaryHistory}$, $\leq_{\mathsf{u}}$, ρ_{u}, and finite $\mathsf{ListCarrier}$; carrier $\mathsf{Carrier}_{\mathsf{u}}$; classifier ρ_{u}; stability from TotalOrder-to-OrderUp and finite extrema uniqueness; ledger finite spine membership plus selected extrema; core theorem this block; exactness classifier uniqueness of interval endpoints; standard bridge finite unary-word bounded intervals; status $\mathsf{ProofTarget}$.

Proof. Use Definition 54.21 to read the concrete unary $\mathsf{TotalOrder}^{\uparrow}$ rows as the L7 arithmetic $\mathsf{Order}^{\uparrow}$ surface. The comparison relation exposed to the consumer is still $\leq_{\mathsf{u}}$, and the classifier exposed on endpoints is still ρ_{u}.

Invoke Definition 54.28 on the nonempty carried spine L. This produces selected histories m and M together with $\mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m)$ and $\mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M)$, plus classifier uniqueness for competing minimum and maximum witness records.

Project the selected minimum witness through Definition 54.24. Its lower-bound row states $\forall x.\ x \in L \to m \leq_{\mathsf{u}} x$. Project the selected maximum witness through Definition 54.26. Its upper-bound row states $\forall x.\ x \in L \to x \leq_{\mathsf{u}} M$.

For each membership witness $x \in L$, pair these two projected comparisons to obtain $m \leq_{\mathsf{u}} x$ and $x \leq_{\mathsf{u}} M$. Hence the finite spine collapses into the bounded unary interval selected by the two extrema.

If m' and M' satisfy the same finite-bound witness rows, apply the uniqueness clauses returned by Definition 54.28. The minimum row classifies m' with m by $\rho_{\mathsf{u}}(m, m')$, and the maximum row classifies M' with M by $\rho_{\mathsf{u}}(M, M')$. □

Theorem 54.38 (Unary prefix OrderUp interval endpoint transport)**.** *Let L and L' be nonempty finite* BHist *spines carried by* $\mathsf{Carrier}_{\mathsf{u}}$*. Suppose their entries are paired by ρ_{u}: every displayed membership row $x \in L$ has a displayed membership row $x' \in L'$ with $\rho_{\mathsf{u}}(x, x')$, and conversely. Let (m, M) and (m', M') be the selected finite extrema supplied by Definition 54.28 for L and L'. Then the selected endpoints transport through the classifier,*

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M'),$$

and the bounded-interval membership row of Definition 54.37 is preserved under the paired spine.

Proof. First read both carried finite spines through Definition 54.21. The consumer relation is $\leq_{\mathsf{u}}$, and the consumer classifier is ρ_{u} on the same displayed histories.

Apply Definition 54.28 to L and to L'. This supplies minimum and maximum witness rows for m, M and for m', M', together with classifier uniqueness for competing selected extrema.

Use the entrywise ρ_{u} pairing to transport lower-bound and upper-bound comparisons from the witness rows for L to the corresponding rows on L'. The transported lower-bound packet makes m' a classified minimum witness for the image of m, and the transported upper-bound packet makes M' a classified maximum witness for the image of M.

The uniqueness clauses from the finite extrema certificate then classify the selected endpoints, giving $\rho_{\mathsf{u}}(m, m')$ and $\rho_{\mathsf{u}}(M, M')$.

Finally invoke Definition 54.37 on the transported membership rows. Each paired entry stays between the transported endpoints, so the bounded-interval membership row is preserved through ρ_{u}. □

Theorem 54.39 (Unary prefix OrderUp interval witness coverage)**.** *Let $L : \mathsf{ListCarrier}(\mathsf{BHist})$ be a nonempty finite carried spine and let m, M be the selected extrema supplied by Definition 54.28. For every entry $x \in L$, every displayed lower-row and upper-row pair*

$$\ell \leq_{\mathsf{u}} x \qquad \textit{and} \qquad x \leq_{\mathsf{u}} u$$

that presents x as lying in a unary prefix OrderUp interval factors through the minimum and maximum witness records for m and M. Any competing displayed interval endpoints ℓ and u satisfying the finite-bound witness rows are classified by ρ_{u} with the selected endpoints:

$$\rho_{\mathsf{u}}(m, \ell) \wedge \rho_{\mathsf{u}}(M, u).$$

Proof. Apply Definition 54.28 to the nonempty carried spine L. It provides the minimum witness record for m, the maximum witness record for M, and classifier uniqueness for any competing minimum or maximum witness records.

Use Definition 54.37. For each membership row $x \in L$, the selected minimum witness supplies $m \leq_{\mathsf{u}} x$ and the selected maximum witness supplies $x \leq_{\mathsf{u}} M$, so the displayed interval row for x factors through the selected extrema.

If the displayed endpoints ℓ and u are carried as competing finite-bound witness rows, invoke the uniqueness clauses from the finite extrema certificate. They classify ℓ with m and u with M by ρ_{u}.

Finally apply Definition 54.38. The endpoint classifications transport the lower and upper interval rows across the displayed ρ_{u} pairing, so every interval witness row is accounted for by the finite-extrema rows. □

54.6 Finite Extrema Interval Uniqueness

Definition 54.40 (Concrete unary finite extrema interval witness)**.** *For a finite BEDC-history spine $L : \mathsf{ListCarrier}(\mathsf{BHist})$, a pair (m, M) is a concrete unary finite extrema interval witness when*

$$\mathsf{FiniteExtremaIntervalWitness}_{\mathsf{u}}(L, m, M) := \mathsf{FiniteMinimumWitness}_{\mathsf{u}}(L, m) \wedge \mathsf{FiniteMaximumWitness}_{\mathsf{u}}(L, M).$$

The carrier side condition is the same finite-carrier list hypothesis used by the minimum and maximum witness predicates.

Theorem 54.41 (Concrete unary prefix finite extrema interval uniqueness)**.** *Let $L : \mathsf{ListCarrier}(\mathsf{BHist})$ be a nonempty finite BEDC-history spine in the helper of Definition 36.1. Assume every entry of L is carried by $\mathsf{Carrier}_{\mathsf{u}}$ for the concrete $\mathsf{TotalOrder}^{\uparrow}$ certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier}_{\mathsf{u}}(x).$$

If (m, M) and (m', M') satisfy $\mathsf{FiniteExtremaIntervalWitness}_{\mathsf{u}}(L, m, M)$ and $\mathsf{FiniteExtremaIntervalWitness}_{\mathsf{u}}(L, m', M'$ then

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M').$$

The per-object fields are: traditional target finite closed interval endpoint uniqueness in a constructive total order; BEDC source finite $\mathsf{ListCarrier}$ spines and unary $\mathsf{TotalOrder}^{\uparrow}$; carrier $\mathsf{Carrier}_{\mathsf{u}}$; classifier ρ_{u}; relation $\leq_{\mathsf{u}}$; stability obligations finite minimum and maximum classifier uniqueness; ledger the two interval witness records and their four endpoint witness projections; core theorem this interval uniqueness block; exactness paired classifier equality of lower and upper endpoint readbacks; standard bridge finite unary-word interval endpoint uniqueness; status $\mathsf{ProofTarget}$.

Proof. Unpack the two interval witness records using Definition 54.40. The first record supplies a minimum witness for m and a maximum witness for M over L; the second supplies a minimum witness for m' and a maximum witness for M' over the same finite spine.

Apply Definition 54.25 to the two minimum witness rows for m and m', together with the nonempty and carrier hypotheses on L. This gives the lower-endpoint classifier agreement $\rho_{\mathsf{u}}(m, m')$.

Apply Definition 54.27 to the two maximum witness rows for M and M', again over the same carried finite spine. This gives the upper-endpoint classifier agreement $\rho_{\mathsf{u}}(M, M')$.

Pair the two classifier conclusions. The resulting statement is a concrete two-endpoint interval readback: changing the selected witnesses may change the visible histories, but only within ρ_{u} at the lower endpoint and within ρ_{u} at the upper endpoint. □

Theorem 54.42 (Concrete unary prefix finite extrema interval endpoint exactness)**.** *Let $L : \mathsf{ListCarrier}(\mathsf{BHist})$ be a nonempty finite BEDC-history spine in the helper of Definition 36.1, and*

assume every entry of L is carried by $\mathsf{Carrier_u}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate. If* (m, M) *and* (m', M') *are interval witnesses over the same spine,*

$$\mathsf{FiniteExtremaIntervalWitness_u}(L, m, M) \quad \textit{and} \quad \mathsf{FiniteExtremaIntervalWitness_u}(L, m', M'),$$

then their lower and upper endpoints agree exactly at the classifier boundary:

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M').$$

Conversely, if $\mathsf{FiniteExtremaIntervalWitness_u}(L, m, M)$ *and the endpoint classifiers satisfy*

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M'),$$

then TotalOrder classifier transport carries the selected interval witness to an equivalent interval witness over the same finite spine with endpoints (m', M')*.*

Proof. For the forward direction, apply Definition 54.41 to the two interval witnesses over the same nonempty carried spine. The theorem already splits the interval witness definition into finite minimum and maximum rows and uses their uniqueness, so it returns exactly the paired classifier conclusions for the lower and upper endpoints.

For the converse direction, unpack Definition 54.40 for the selected pair (m, M). The lower row is a finite minimum witness for m, and the upper row is a finite maximum witness for M.

Transport the lower witness along $\rho_{\mathsf{u}}(m, m')$ by the TotalOrder classifier-transport clause for finite minima. Transport the upper witness along $\rho_{\mathsf{u}}(M, M')$ by the corresponding finite maximum transport clause.

Repack the transported minimum and maximum rows through the interval witness definition. The finite spine L and its carrier hypotheses have not changed, so the transported endpoints form an equivalent interval witness over the same spine. □

Theorem 54.43 (Concrete unary prefix finite extrema interval roundtrip stability)**.** *Let* L : $\mathsf{ListCarrier}(\mathsf{BHist})$ *be a nonempty finite BEDC-history spine, and assume every entry of* L *is carried by* $\mathsf{Carrier_u}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate. If* (m, M) *is an interval witness over* L *and the endpoint classifier rows*

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M')$$

hold, then transport of the witness to (m', M') *and comparison back against the original witness recover the same lower and upper endpoint classifier rows:*

$$\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M').$$

Proof. Use the converse direction of Definition 54.42 on the selected witness (m, M) and the two endpoint classifier rows. This transports the finite minimum and maximum parts of the interval witness to an equivalent interval witness over the same spine with endpoints (m', M').

Now compare the original witness and the transported witness. Apply Definition 54.41 to the two interval witnesses over the same nonempty carried finite spine. Its lower endpoint conclusion is $\rho_{\mathsf{u}}(m, m')$, and its upper endpoint conclusion is $\rho_{\mathsf{u}}(M, M')$.

These are exactly the classifier rows used for the transport, so the there-and-back comparison is stable at both interval endpoints. □

Theorem 54.44 (Concrete unary prefix finite extrema interval certificate)**.** *Let* $L : \mathsf{ListCarrier}(\mathsf{BHist})$ *be a nonempty finite BEDC-history spine in the helper of Definition 36.1. Assume every entry of* L *is carried by* $\mathsf{Carrier_u}$ *for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *certificate of Definition 54.20:*

$$\forall x.\ x \in L \to \mathsf{Carrier_u}(x).$$

Then there exist histories m, M *such that*

$$\mathsf{FiniteExtremaIntervalWitness_u}(L, m, M)$$

holds and every other interval witness (m', M') *over the same spine satisfies*

$$\mathsf{FiniteExtremaIntervalWitness_u}(L, m', M') \to (\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M')).$$

The per-object fields are: traditional target finite closed interval extraction in a constructive total order; BEDC source finite $\mathsf{ListCarrier}$ *spines plus unary* $\mathsf{TotalOrder}^{\uparrow}$*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability finite extrema existence plus interval endpoint uniqueness; ledger selected lower and upper witnesses; core theorem this block; exactness classifier-stable interval endpoints; standard bridge finite unary-word closed interval; status* $\mathsf{ProofTarget}$*.*

Proof. Invoke Definition 54.28 on the nonempty carried finite spine L. It supplies selected histories m and M together with the minimum witness row for m, the maximum witness row for M, and the classifier-stability clauses attached to those selected rows.

Repack the selected lower and upper witness rows through Definition 54.40. This gives FiniteExtremaIntervalWitn for the selected pair.

Now let (m', M') be any other pair satisfying $\mathsf{FiniteExtremaIntervalWitness_u}(L, m', M')$. Apply Definition 54.41 to the selected interval witness and the competing interval witness over the same L.

The uniqueness theorem returns $\rho_{\mathsf{u}}(m, m') \wedge \rho_{\mathsf{u}}(M, M')$. Thus the selected finite interval exists and its lower and upper endpoints are classifier-stable against every other interval witness for the same carried finite spine. □

54.7 OrderUp Phase Exit

Theorem 54.45 (Unary prefix OrderUp phase exit package)**.** *The concrete L7* $\mathsf{Order}^{\uparrow}$ *package over* BHist *is the unary-prefix package with carrier* $\mathsf{UnaryHistory}$*, classifier* ρ_{u}*, and order relation* $\leq_{\mathsf{u}}$*. Its preorder, antisymmetry, and constructive comparison rows are inherited from the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *surface exposed in Definition 54.21. Its finite bounded-interval coverage is the interval-collapse statement of Definition 54.37. Its endpoint classifier uniqueness is the two-endpoint readback of Definition 54.41.*

Consequently, the scoped Full Nat Arithmetic $\mathsf{Order}^{\uparrow}$ *phase object can be consumed as a single package consisting of the carrier* $\mathsf{UnaryHistory}$*, classifier* ρ_{u}*, relation* $\leq_{\mathsf{u}}$*, inherited preorder and antisymmetry rows, constructive comparison rows, finite bounded-interval coverage for carried finite spines, and classifier uniqueness of selected interval endpoints. This package introduces no second order relation and asserts no classical trichotomy.*

Proof. First expose the $\mathsf{Order}^{\uparrow}$ surface. Apply Definition 54.21. The consumer sees the same concrete $\mathsf{UnaryHistory}$ carrier, the same classifier ρ_{u}, and the same relation $\leq_{\mathsf{u}}$ that were used by the unary $\mathsf{TotalOrder}^{\uparrow}$ certificate.

The order-law rows are therefore projections of the existing certificate. The reflexivity, transitivity, and antisymmetry rows come from the prefix $\mathsf{POSet}^{\uparrow}$ part of the $\mathsf{TotalOrder}^{\uparrow}$ package, while the comparison row is the constructive strict-left, same, or strict-right branch partition recorded there.

Next project the finite bounds. For a nonempty finite carried spine, apply Definition 54.37. It supplies selected endpoints m and M such that every member of the spine lies between m and M under $\leq_{\mathsf{u}}$.

Finally project endpoint uniqueness. Apply Definition 54.41 to any two finite extrema interval witness records over the same carried spine. The lower endpoints are classified by ρ_{u}, and the upper endpoints are classified by ρ_{u}.

These projected rows are exactly the phase-exit data needed by the Full Nat Arithmetic consumer for $\mathsf{Order}^{\uparrow}$: one concrete carrier, one classifier, one order relation, finite interval coverage, and endpoint classifier uniqueness. All rows are inherited from the unary prefix $\mathsf{TotalOrder}^{\uparrow}$ construction, so the package does not create another order and does not replace constructive comparison by classical trichotomy. □

Theorem 54.46 (OrderUp total-order alias boundary)**.** *The L7 public* $\mathsf{Order}^{\uparrow}$ *phase package used by Full Nat Arithmetic is exactly the tuple* $(\mathsf{UnaryHistory}, \rho_{\mathsf{u}}, \leq_{\mathsf{u}})$ *obtained by projecting the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *construction of Definition 54.21 and by assembling the phase-exit rows of Definition 54.45. Its carrier row is* $\mathsf{UnaryHistory}$*, its classifier row is* ρ_{u}*, and its relation row is the prefix relation* $\leq_{\mathsf{u}}$*.*

The public $\mathsf{Order}^{\uparrow}$ *rows are precisely the projected preorder, antisymmetry, constructive comparison, finite interval coverage, and endpoint classifier uniqueness rows named by those two theorem blocks. Thus downstream consumers may cite* $\mathsf{Order}^{\uparrow}$ *as the unary-prefix projection of* $\mathsf{TotalOrder}^{\uparrow}$*, but they receive no second order relation, no host* Nat *order, no quotient order, and no classical trichotomy.*

Proof. Project the base tuple from Definition 54.21. The carrier is $\mathsf{UnaryHistory}$, the classifier is ρ_{u}, and the relation is $\leq_{\mathsf{u}}$, the concrete prefix relation on carried unary histories.

Next apply Definition 54.45. It repacks the $\mathsf{TotalOrder}^{\uparrow}$ preorder and antisymmetry rows, the constructive comparison branches, the finite interval coverage row, and the endpoint classifier uniqueness row as the public $\mathsf{Order}^{\uparrow}$ phase package.

The two cited theorem blocks name all consumer-visible rows of this phase object. Since each row is a projection from the unary-prefix construction or a phase-exit assembly of those projections, there is no independent order row to transport, no ambient Nat ordering to import, no quotient ordering to identify, and no classical trichotomy principle to add. □

54.7.1 OrderUp obligation set

Theorem 54.47 (Unary prefix OrderUp carrier obligation)**.** *The carrier row of the Full Nat Arithmetic* $\mathsf{Order}^{\uparrow}$ *phase package is exactly* $\mathsf{UnaryHistory}$ *over the ambient kernel history type* BHist*. It introduces no second carrier and no quotient carrier.*

Proof. Use the phase-exit package of Definition 54.45. Its carrier component is named explicitly as $\mathsf{UnaryHistory}$, and the ambient object is the same BHist history used by the unary-prefix construction.

The theorem is therefore the carrier projection of that package: the consumer receives the listed carried histories and no additional carrier row. □

Theorem 54.48 (Unary prefix OrderUp classifier obligation)**.** *The classifier row of the Full Nat Arithmetic* $\mathsf{Order}^{\uparrow}$ *phase package is the same endpoint classifier* ρ_{u} *used by the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *surface.*

Proof. Apply Definition 54.45. The theorem names ρ_{u} as the package classifier and says that it is inherited from the concrete unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ surface.

Thus the $\mathsf{Order}^{\uparrow}$ consumer reads endpoint sameness through the already exposed unary classifier rather than through a separate phase-specific classifier. □

Theorem 54.49 (Unary prefix OrderUp relation obligation)**.** *The order-relation row of the Full Nat Arithmetic* $\mathsf{Order}^{\uparrow}$ *phase package is the same concrete unary-prefix relation* $\leq_{\mathsf{u}}$ *exposed by the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *surface on* $\mathsf{UnaryHistory}$ *endpoints. It introduces no second comparison relation, no host Nat order, and no quotient order.*

Proof. Apply Definition 54.45. The package statement explicitly names $\leq_{\mathsf{u}}$ as the order relation consumed by the Full Nat Arithmetic $\mathsf{Order}^{\uparrow}$ phase.

The source of that row is Definition 54.21. The phase package projects the relation from this unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ certificate, so the consumer receives the exposed relation itself and no additional comparison relation. □

Theorem 54.50 (Unary prefix OrderUp preorder obligation)**.** *On carried endpoints, reflexivity and transitivity of* $\leq_{\mathsf{u}}$ *are the preorder rows inherited from Definition 54.21.*

Proof. First expose the unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ certificate by applying Definition 54.21. Its $\mathsf{POSet}^{\uparrow}$ component supplies reflexivity and transitivity for the relation $\leq_{\mathsf{u}}$ on $\mathsf{UnaryHistory}$ endpoints.

Then project those rows through Definition 54.45. The projected relation is still $\leq_{\mathsf{u}}$, so the preorder data has exactly the same domain and classifier boundary as in the source certificate. □

Theorem 54.51 (Unary prefix OrderUp antisymmetry obligation)**.** *If two carried endpoints compare in both directions under* $\leq_{\mathsf{u}}$*, then their endpoints are classified together by* ρ_{u}*.*

Proof. Apply the antisymmetry row supplied by Definition 54.21. That row identifies two-sided $\leq_{\mathsf{u}}$ comparison with the endpoint classifier used by the unary-prefix order.

Finally use Definition 54.45 to read this row inside the $\mathsf{Order}^{\uparrow}$ package. Since the package keeps the same classifier ρ_{u}, the classified endpoint equality is the required one. □

Theorem 54.52 (Unary prefix OrderUp constructive comparison obligation)**.** *Every carried pair of endpoints admits strict-left, same, or strict-right comparison data for* $\leq_{\mathsf{u}}$*, as supplied by the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *surface. This obligation asserts constructive comparison data and does not assert classical trichotomy.*

Proof. Use the comparison row of Definition 54.21. For each carried pair, that row gives one of the constructive branches: strict-left, same, or strict-right.

The phase-exit package of Definition 54.45 transports the same row to the $\mathsf{Order}^{\uparrow}$ consumer. Because the package states that it asserts no classical trichotomy, the obligation is exactly the constructive branch data. □

Theorem 54.53 (Unary prefix OrderUp finite interval coverage obligation)**.** *Every nonempty carried finite spine has selected lower and upper endpoints under* $\leq_{\mathsf{u}}$*, supplied by Definition 54.37.*

Proof. Start with a nonempty finite spine whose entries are carried by $\mathsf{UnaryHistory}$. Apply Definition 54.37 to obtain selected endpoints m and M.

The conclusion of the interval-collapse theorem says that every member of the spine lies between those selected endpoints under $\leq_{\mathsf{u}}$. This is the finite-interval coverage row consumed by the phase-exit package. □

Theorem 54.54 (Unary prefix OrderUp endpoint uniqueness obligation)**.** *Competing selected lower and upper endpoints over the same nonempty carried finite spine are classified by ρ_{u}, as read back from Definition 54.41.*

Proof. Take two finite extrema interval witness records over the same carried finite spine. Apply Definition 54.41 to compare their selected lower endpoints and their selected upper endpoints.

The readback theorem classifies both endpoint comparisons through ρ_{u}. Therefore the $\mathsf{Order}^{\uparrow}$ package has endpoint uniqueness only up to the declared unary-prefix classifier, which is the phase classifier of Definition 54.45. □

Theorem 54.55 (Unary prefix OrderUp exactness obligation)**.** *On carried unary endpoints h and k, the $\mathsf{Order}^{\uparrow}$ classifier row ρ_{u} is exact for two-sided $\leq_{\mathsf{u}}$ comparison:*

$$\rho_{\mathsf{u}}(h,k) \quad \textit{if and only if} \quad h \leq_{\mathsf{u}} k \wedge k \leq_{\mathsf{u}} h.$$

This exactness row uses the same carrier, classifier, and relation tuple as the unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ surface and introduces no quotient equality.

Proof. First expose the concrete phase package by Definition 54.45. It identifies the $\mathsf{Order}^{\uparrow}$ carrier with $\mathsf{UnaryHistory}$, the classifier with ρ_{u}, and the relation with $\leq_{\mathsf{u}}$.

Project the same tuple back to the inherited unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ surface of Definition 54.21. There is no transport across a new carrier or a new relation; the consumer is reading the original preorder classifier boundary.

Apply Definition 53.34. Its forward direction turns classifier sameness into the two prefix comparisons, and its reverse direction turns the two comparisons back into the carrier-aware classifier. Under the package projection, that equivalence is precisely the displayed ρ_{u} exactness row for $\mathsf{Order}^{\uparrow}$. □

Theorem 54.56 (Unary prefix OrderUp ledger obligation)**.** *The Full Nat Arithmetic $\mathsf{Order}^{\uparrow}$ ledger records exactly the concrete unary carrier witnesses, the selected constructive comparison branch, the finite interval endpoint witnesses, and the endpoint-readback comparisons. All entries are stated over BHist, $\mathsf{ListCarrier}$, and Cont data. The ledger contains no quotient carrier, no host order, and no unrecorded classical comparison principle.*

Proof. Begin with the phase package of Definition 54.45. Its ledger boundary is the displayed $\mathsf{UnaryHistory}$ carrier over BHist, the classifier ρ_{u}, the relation $\leq_{\mathsf{u}}$, and the constructive comparison branch selected by the inherited unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ surface.

Next read the finite-spine part through Definition 54.37. For a nonempty $\mathsf{ListCarrier}$ spine, the ledger keeps the membership witnesses, the selected lower and upper endpoints, and the two bounding comparisons against each listed entry.

Finally apply Definition 54.41. This supplies the endpoint-readback comparisons showing that competing lower endpoints and competing upper endpoints are classified by ρ_{u}.

Combining these rows gives exactly the advertised ledger: every recorded item is a BEDC history, finite-list, or continuation witness already present in the unary-prefix package, and none of the rows appeal to a quotient carrier or an external host order. □

Theorem 54.57 (OrderUp carrier and classifier obligation)**.** *The concrete carrier for the unary-prefix* $\mathsf{Order}^{\uparrow}$ *phase object is* $\mathsf{UnaryHistory}$*. Its classifier is* ρ_{u}*, and its order relation is* $\leq_{\mathsf{u}}$*.*

Proof. Apply Definition 54.45. The phase-exit package exposes the same carried objects as Definition 54.21: unary histories are the carrier, ρ_{u} is the classifier used to read them, and $\leq_{\mathsf{u}}$ is the only order relation exported to $\mathsf{Order}^{\uparrow}$. □

Theorem 54.58 (OrderUp preorder and antisymmetry obligation)**.** *On carried unary histories,* $\leq_{\mathsf{u}}$ *is reflexive and transitive. If two carried unary histories lie below each other under* $\leq_{\mathsf{u}}$*, then their endpoints land in the same* ρ_{u} *classifier class.*

Proof. First project the order rows from the concrete unary-prefix total-order certificate through Definition 54.21. The $\mathsf{Order}^{\uparrow}$ phase package of Definition 54.45 exports those rows without introducing another relation.

Reflexivity and transitivity are the preorder rows of the inherited $\leq_{\mathsf{u}}$ surface. Antisymmetry is read in the classifier used by the same package: mutual $\leq_{\mathsf{u}}$ comparison identifies the carried unary histories up to ρ_{u}, which is the equality notion exposed to the phase consumer. □

Theorem 54.59 (OrderUp constructive comparison obligation)**.** *For any two carried unary histories, the* $\mathsf{Order}^{\uparrow}$ *comparison data gives one of the strict-left, same, or strict-right branches from the unary-prefix total-order package. The phase object exports this constructive branch data and exports no classical trichotomy principle.*

Proof. Use the comparison row supplied by Definition 54.21 and projected into $\mathsf{Order}^{\uparrow}$ by Definition 54.45. The branch names and their witnesses are the concrete unary-prefix strict-left, same, and strict-right alternatives.

Because the exported row is this constructive package row, the consumer receives only the witnessed branch split. No additional excluded-middle or classical trichotomy statement is part of the $\mathsf{Order}^{\uparrow}$ phase object. □

Theorem 54.60 (OrderUp finite-interval and endpoint obligation)**.** *Every nonempty finite carried unary spine has lower and upper bounds under* $\leq_{\mathsf{u}}$*. Moreover, for any two endpoint-witness records for the same carried spine, their lower endpoints are* ρ_{u}*-equivalent and their upper endpoints are* ρ_{u}*-equivalent.*

Proof. First obtain finite interval coverage from Definition 54.37. For a nonempty finite carried spine, it supplies selected lower and upper unary endpoints with every carried member lying between them under $\leq_{\mathsf{u}}$.

Next apply Definition 54.41 to two endpoint-witness records over the same carried spine. The theorem identifies the selected lower endpoints through ρ_{u} and likewise identifies the selected upper endpoints through ρ_{u}.

Together with the phase package of Definition 54.45, these are precisely the finite-interval and endpoint-uniqueness obligations exported by the concrete unary-prefix $\mathsf{Order}^{\uparrow}$ object. □

Theorem 54.61 (OrderUp exactness and ledger scope)**.** *The unary-prefix* $\mathsf{Order}^{\uparrow}$ *package is exact at each exported row. Its carrier is exactly* $\mathsf{UnaryHistory}$*; its classifier is exactly* ρ_{u}*; and its order relation is exactly the single exported* $\leq_{\mathsf{u}}$ *row. Its finite-interval coverage is precisely the bounded interval collapse of Definition 54.37, and its endpoint ledger uniqueness is precisely the two-endpoint classifier readback of Definition 54.41. The package exports no host Nat order row, no quotient order row, and no classical trichotomy row.*

Proof. Begin with the phase-exit package of Definition 54.45. Its named surface fixes the carrier as $\mathsf{UnaryHistory}$, the classifier as ρ_{u}, and the relation as $\leq_{\mathsf{u}}$.

The carrier, classifier, and relation exactness claims are the three direct row projections recorded in Definition 54.47, Definition 54.48, and Definition 54.49. These are repacked in Definition 54.57 as the public carrier, classifier, and relation face of the same package.

The order-law ledger is inherited without changing that face. Reflexivity and transitivity come from Definition 54.50, antisymmetry from Definition 54.51, and the witnessed comparison row from Definition 54.52. The combined public statement is Definition 54.58 together with Definition 54.59.

Finally apply Definition 54.53 and Definition 54.54. These are the finite-interval collapse and endpoint uniqueness rows, summarized by Definition 54.60. Since every row is projected from the single phase-exit package, the scope contains only $\mathsf{UnaryHistory}$, ρ_{u}, and $\leq_{\mathsf{u}}$; there is no additional host Nat order, quotient order, or classical trichotomy row to export. □

Theorem 54.62 (Unary prefix OrderUp scoped dependency package)**.** *The Full Nat Arithmetic* $\mathsf{Order}^{\uparrow}$ *package is scoped over the concrete tuple* $(\mathsf{BHist}, \mathsf{UnaryHistory}, \rho_{\mathsf{u}}, \leq_{\mathsf{u}})$*. Its endpoint objects are* BHist *histories compared through* hsame*; its carrier row is exactly* $\mathsf{UnaryHistory}$*; its order row is exactly the unary-prefix relation* $\leq_{\mathsf{u}}$*; its finite carried spines are the* $\mathsf{ListCarrier}$ *spines; and* Cont *is used only for the finite-spine ledger rows that record the carried membership, selected endpoint, and bounding-comparison witnesses.*

The ten-field fit is carrier $\mathsf{UnaryHistory}$*, classifier* ρ_{u}*, relation* $\leq_{\mathsf{u}}$*, preorder, antisymmetry, constructive comparison, finite interval coverage, endpoint uniqueness, exactness, and ledger scope. These ten rows are exactly the obligations packaged by Definition 54.57, Definition 54.58, Definition 54.59, Definition 54.60, and Definition 54.55, with the ledger boundary stated by Definition 54.61. Therefore the package has a single scoped dependency boundary and introduces no host Nat order, quotient order, second classifier, or classical trichotomy row.*

Proof. First project the carrier, classifier, and relation from Definition 54.45. This supplies the concrete tuple $(\mathsf{BHist}, \mathsf{UnaryHistory}, \rho_{\mathsf{u}}, \leq_{\mathsf{u}})$ and fixes endpoint comparison at the BHist and hsame boundary.

Next read the order-law rows from the inherited unary-prefix total-order surface. The preorder, antisymmetry, and constructive comparison data are provided by Definition 54.21 and are summarized for the $\mathsf{Order}^{\uparrow}$ consumer by Definition 54.58 and Definition 54.59.

For finite carried spines, apply Definition 54.37. It supplies the finite interval coverage over $\mathsf{ListCarrier}$ data, and Definition 54.41 supplies the endpoint uniqueness readback for competing lower and upper endpoint records.

It remains to identify the exactness and ledger boundary. Apply Definition 53.34 for the two-sided $\leq_{\mathsf{u}}$ exactness of ρ_{u}, and use Definition 54.61 to keep Cont confined to the finite-spine ledger rows. The listed projections are precisely the carrier, classifier, relation, preorder, antisymmetry, comparison, interval coverage, endpoint uniqueness, exactness, and ledger obligations, so they form one scoped dependency package. □

Theorem 54.63 (Unary prefix OrderUp public export)**.** *The concrete L7* $\mathsf{Order}^{\uparrow}$ *phase object exports a single public surface whose carrier is* $\mathsf{UnaryHistory}$*, whose classifier is* ρ_{u}*, and whose relation is* $\leq_{\mathsf{u}} := \mathsf{PreorderPrefixLE}$*. This surface includes the constructive comparison data, finite interval coverage, endpoint uniqueness, exactness, and ledger scope supplied by the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *package. Every exported row is sourced from that package boundary, so the public* $\mathsf{Order}^{\uparrow}$ *object uses no host order and no classical trichotomy.*

Proof. Use Definition 54.62 as the aggregate dependency package. It binds the concrete surface to $(\mathsf{BHist}, \mathsf{UnaryHistory}, \rho_{\mathsf{u}}, \leq_{\mathsf{u}})$ and identifies the relevant rows as the unary-prefix $\mathsf{Order}^{\uparrow}$ obligations.

Project the carrier, classifier, and public relation rows from Definition 54.57 and Definition 54.49. These projections fix the public carrier as $\mathsf{UnaryHistory}$, the classifier as ρ_{u}, and the relation as $\leq_{\mathsf{u}} := \mathsf{PreorderPrefixLE}$.

Next project the constructive comparison row from Definition 54.59. Project finite interval coverage and endpoint uniqueness from Definition 54.60. These rows give the comparison witnesses, the finite carried interval, and the selected endpoint readback for the same unary-prefix surface.

Finally apply Definition 54.61. It identifies the exactness boundary and confines Cont to the finite-spine ledger rows. The carrier, classifier, relation, comparison, interval, endpoint, exactness, and ledger projections therefore repack into one public $\mathsf{Order}^{\uparrow}$ export surface, with no host order row and no classical trichotomy row. □

Theorem 54.64 (TotalOrderUp unary-prefix OrderUp consumer coverage boundary)**.** *Any downstream* $\mathsf{Order}^{\uparrow}$ *or* $\mathsf{TotalOrder}^{\uparrow}$ *consumer of the unary-prefix public export receives exactly the rows named by Definition 54.45 and Definition 54.63:* $\mathsf{UnaryHistory}$*,* ρ_{u}*,* $\leq_{\mathsf{u}}$*, the preorder and antisymmetry rows, constructive comparison branches, finite interval coverage, endpoint uniqueness, exactness, and finite ledger rows. It receives no host* Nat *order, no quotient equality, and no classical trichotomy.*

Proof. Read the public surface from Definition 54.63. Its carrier, classifier, and relation rows are $\mathsf{UnaryHistory}$, ρ_{u}, and $\leq_{\mathsf{u}} := \mathsf{PreorderPrefixLE}$.

Apply the phase-exit package Definition 54.45. The obligation-row theorems in this section project the preorder law rows, antisymmetry row, constructive comparison branches, finite interval coverage, endpoint uniqueness, and exactness row from the same unary-prefix package.

Finally apply Definition 54.56 together with Definition 54.61. These identify the finite Cont ledger rows and show that no additional host order, quotient equality, or classical trichotomy row is available to a downstream consumer. □

Theorem 54.65 (Unary prefix OrderUp standard bridge boundary)**.** *The public surface of Definition 54.63 bridges to finite unary words by reading* $\mathsf{UnaryHistory}$ *as the carried finite unary-word endpoints,* ρ_{u} *as the finite-word classifier, and* $\leq_{\mathsf{u}} := \mathsf{PreorderPrefixLE}$ *as initial-segment order. Its finite interval rows are exactly the rows of Definition 54.37, and its endpoint rows are exactly the rows of Definition 54.41. The bridge therefore exposes the unary-prefix* $\mathsf{Order}^{\uparrow}$ *package as a finite-word standard surface, with no host* Nat *order row and no classical trichotomy row.*

Proof. Start with the public export of Definition 54.63. It fixes the carrier, classifier, and comparison as $(\mathsf{UnaryHistory}, \rho_{\mathsf{u}}, \mathsf{PreorderPrefixLE})$, so each public row already has the finite unary-word reading required by the standard surface.

Apply Definition 54.37 to read the finite interval rows as carried initial-segment intervals. Apply Definition 54.41 to read the lower and upper endpoint rows as the unique finite-word endpoints selected by the same prefix comparison.

It remains to identify what is absent from the bridge. The public export and the two cited rows use only BHist, $\mathsf{UnaryHistory}$, ρ_{u}, $\mathsf{PreorderPrefixLE}$, and finite Cont transport. Thus no host Nat order and no classical trichotomy are exported. □

Theorem 54.66 (OrderUp public export formal boundary)**.** *The public* $\mathsf{Order}^{\uparrow}$ *export in this chapter is the unary-prefix package of Definition 54.63. The displayed Lean target in the closure block supplies only the inherited* $\mathsf{Preorder}^{\uparrow}$ *stability field, so the chapter may claim the paper public surface but not an audit-clean* $\mathsf{Order}^{\uparrow}$ *namespace target.*

Proof. Project the public export of Definition 54.63. It exposes the carrier UnaryHistory, classifier ρ_{u}, and relation $\leq_{\mathsf{u}} :=$ PreorderPrefixLE from the unary-prefix package.

Project the remaining paper rows through Definition 54.45. The exactness row and finite Cont ledger rows are part of the same public package, so the manuscript has a public $\mathsf{Order}^{\uparrow}$ surface over the listed carrier, classifier, relation, exactness, and ledger data.

The Lean target recorded below remains a dependency boundary: it certifies the inherited $\mathsf{Preorder}^{\uparrow}$ stability field rather than a checked $\mathsf{Order}^{\uparrow}$ namespace theorem. Therefore the theory axis can report the public paper surface, while the verification axis stays below an audit-clean $\mathsf{Order}^{\uparrow}$ target. □

Theorem 54.67 (OrderUp unary prefix namespace target surface)**.** *The* $\mathsf{Order}^{\uparrow}$ *namespace target surface in this chapter is exactly the unary carrier* UnaryHistory*, classifier* ρ_{u}*, relation* $\leq_{\mathsf{u}} :=$ PreorderPrefixLE*, constructive comparison rows, finite interval rows, endpoint uniqueness rows, exactness row, ledger rows, public export row, and bridge row displayed in this section. No additional carrier, comparison, endpoint, quotient, or host-order row belongs to the chapter surface.*

Proof. Project the carrier, classifier, and relation from Definition 54.63. The constructive comparison branches and order laws are the rows repacked by Definition 54.45.

Next project the finite interval and endpoint rows from Definition 54.37 and Definition 54.41. These are the only interval and endpoint witnesses attached to the unary-prefix public package.

Finally read the exactness and ledger clauses through Definition 54.56, Definition 54.61, and Definition 54.65. Repacking these projected rows gives precisely the listed namespace surface and nothing outside the displayed BHist, UnaryHistory, hsame, and Cont data. □

Theorem 54.68 (OrderUp unary prefix audit consumer boundary)**.** *Any audit consumer of the* $\mathsf{Order}^{\uparrow}$ *target surface in this chapter may cite only the* BHist*,* UnaryHistory*,* hsame*, and* Cont *rows listed by Definition 54.67. Such a consumer may not replace those rows by a host order on natural numbers, an external finite-word equality, or classical trichotomy.*

Proof. Start from the public export row of Definition 54.63. Its consumer-visible data are the carried unary histories, the unary classifier, and the prefix relation.

The ledger restrictions in Definition 54.56 and Definition 54.61 state that continuation readback is finite and attached to the displayed unary-prefix package. The standard bridge boundary of Definition 54.65 gives the finite-word reading without adding a host natural-number order.

Thus an audit consumer can repack only the listed BEDC rows. Host order, external word equality, and classical trichotomy are not projections of any cited theorem block, so they cannot be substituted for the chapter surface. □

Theorem 54.69 (OrderUp unary prefix formalization handshake)**.** *A Lean theorem in the namespace* BEDC.Derived.OrderUp *that closes this chapter must prove the same unary-prefix surface listed in Definition 54.67. The inherited* $\mathsf{Preorder}^{\uparrow}$ *stability dependency may remain a source row, but it is not the canonical* $\mathsf{Order}^{\uparrow}$ *theorem for the carrier, comparison, finite interval, endpoint, exactness, ledger, public export, and bridge surface.*

Proof. The dependency target in the closure block certifies the preorder stability field used by the unary-prefix relation. That row is one source of the public package, not the whole $\mathsf{Order}^{\uparrow}$ package.

Project the chapter surface from Definition 54.67. It contains comparison, bounded interval, endpoint uniqueness, exactness, ledger, public export, and bridge data in addition to the inherited preorder row.

Therefore a namespace theorem for BEDC.Derived.OrderUp must repack all listed rows as its conclusion. Reusing the $\mathsf{Preorder}^{\uparrow}$ dependency as the canonical theorem would omit the displayed $\mathsf{Order}^{\uparrow}$ rows and would not match the chapter surface. □

Theorem 54.70 (TotalOrder source boundary for the OrderUp phase exit)**.** *The* $\mathsf{Order}^{\uparrow}$ *phase-exit surface in this file is a consumer projection of the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *package through* BHist*,* UnaryHistory*,* ρ_{u}*,* $\leq_{\mathsf{u}}$*, finite* ListCarrier *spines, and* Cont *ledger rows. It does not by itself certify a canonical theorem in the namespace* BEDC.Derived.OrderUp*.*

Proof. Start with Definition 54.21. It supplies the total-order source package whose unary carrier, classifier, comparison, and finite spine data feed the displayed $\mathsf{Order}^{\uparrow}$ projection.

Apply Definition 54.63 to read that source package as the public $\mathsf{Order}^{\uparrow}$ carrier, classifier, and relation. Then use Definition 54.65 to identify the finite ListCarrier spine reading without adding a separate host-order source.

Finally apply Definition 54.69. The handshake names the remaining formal boundary: the chapter surface is a paper-side consumer projection of the unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ package, so the closure block below cannot be read as a canonical BEDC.Derived.OrderUp namespace certificate. □

54.8 The certificate

Definition 54.71 (TotalOrder source specification)**.** *The source is a certified poset carrier plus apartness-aware trichotomy.*

Definition 54.72 (TotalOrder pattern specification)**.** *The pattern is comparison formation producing one of three certified comparison cases.*

Definition 54.73 (TotalOrder classifier specification)**.** *The classifier is the poset carrier classifier, with apartness stable under classifier replacement.*

Definition 54.74 (TotalOrder stability certificate)**.** *The stability certificate consists of:*

1. *all poset stability fields*
2. *strict-order-to-apartness field*
3. *trichotomy as explicit comparison data*
4. *comparison stability under classifier replacement*
5. *transitivity of strict comparison where supplied*

Definition 54.75 (TotalOrder ledger policy)**.** *Record the comparison certificate selected for each pair.*

Definition 54.76 ($\mathsf{TotalOrder}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{TotalOrder}^{\uparrow}$*.*

Theorem 54.77 (TotalOrder carrier branch exhaustion)**.** *For carried singleton* BHist *endpoints* $h, k : A_{\mathsf{e}}$*, the explicit comparison data in Definition 54.2 exhausts exactly the three branches* lt_{e}*,* $\mathsf{same}_{\mathsf{e}}$*, and* gt_{e}*. The singleton carrier forces the same branch:* $\mathsf{same}_{\mathsf{e}}(h, k)$ *is* $\rho_{\mathsf{e}}(h, k)$*, while the strict left and strict right branches are empty.*

Proof. Apply Definition 54.3. Its trichotomy row is stated for the displayed branches of the singleton history instance, and the instance defines the strict branches as $\bot$ and the middle branch as ρ_{e}.

The carrier hypotheses say that h and k are the singleton BHist endpoints. Hence no additional endpoint case exists outside the displayed three-way comparison data, and the only inhabited branch is the same branch. □

Theorem 54.78 (TotalOrder classifier antisymmetry scope)**.** *For carried singleton endpoints $h, k : A_{\mathsf{e}}$, opposing prefix comparisons $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$ recover $\rho_{\mathsf{e}}(h, k)$. Therefore the classifier antisymmetry scope of the concrete $\mathsf{TotalOrder}^{\uparrow}$ certificate is exactly the singleton empty-poset scope and contains no comparison datum beyond the carried BHist endpoints.*

Proof. Apply Definition 53.12. Its exactness row identifies the singleton classifier $\rho_{\mathsf{e}}(h, k)$ with the pair of opposing prefix comparisons on carried endpoints.

Now read the same row through the branch specification in this file. Since the strict branches are empty and the middle branch is ρ_{e}, antisymmetry is precisely the recovery of the middle classifier branch from the two prefix comparisons. □

Theorem 54.79 (TotalOrder certificate boundary exactness)**.** *The carrier branch exhaustion of Definition 54.77, the classifier antisymmetry scope of Definition 54.78, the stability fields of Definition 54.74, and the ledger policy of Definition 54.75 assemble the exact $\mathsf{TotalOrder}^{\uparrow}$ boundary over the singleton BHist carrier. The boundary exports only the carrier, comparison branch, classifier, stability, exactness, and selected ledger rows listed in this file.*

Proof. First assemble the carrier and branch surface. Use Definition 54.77 to obtain the exhaustive three-way comparison data and the singleton same-branch reduction.

Next assemble the classifier exactness surface. Use Definition 54.78 to identify the classifier with opposing prefix comparisons, and use Definition 53.12 for the underlying empty-poset exactness rows.

Finally cite Definition 54.3 for total-order stability and comparison stability. Repacking these rows with Definition 54.71, Definition 54.72, Definition 54.73, Definition 54.74, and Definition 54.75 gives exactly the certificate boundary claimed by Definition 54.76. □

Theorem 54.80 ($\mathsf{Order}^{\uparrow}$ unary-prefix constructive trichotomy readback)**.** *For carried unary histories h, k in $\mathsf{Carrier}_{\mathsf{u}}$, the unary-prefix comparison partition reads back as concrete BHist data over $\mathsf{Carrier}_{\mathsf{u}}$, ρ_{u}, and $\leq_{\mathsf{u}}$. Exactly one selected branch is exported:*

$$\mathsf{NatUnaryStrictPrefix}(h, k), \qquad \rho_{\mathsf{u}}(h, k), \qquad \mathsf{NatUnaryStrictPrefix}(k, h).$$

The strict-left branch is the concrete nonempty unary tail from h to k, the same branch is the classifier row $\rho_{\mathsf{u}}(h, k)$, and the strict-right branch is the concrete nonempty unary tail from k to h.

Proof. Apply Definition 54.9 to the carried pair h, k. Its three branches are already stated over the unary BHist carrier and use the classifier ρ_{u} as the middle branch.

In the strict-left case, the partition supplies the concrete nonempty continuation witness from h to k, so the public branch is $\mathsf{NatUnaryStrictPrefix}(h, k)$. In the same case, the middle branch is exactly $\rho_{\mathsf{u}}(h, k)$, and the exactness row identifies it with the opposing $\leq_{\mathsf{u}}$ comparisons.

In the strict-right case, the same construction is read with endpoints reversed, yielding $\mathsf{NatUnaryStrictPrefix}(k, h)$. The incompatibility rows of the partition exclude every second selected branch, so the readback is constructive branch data rather than a host-order trichotomy principle. □

Theorem 54.81 ($\mathsf{Order}^{\uparrow}$ prefix branch ledger exactness)**.** *For every carried unary pair h, k in $\mathsf{Carrier_u}$, the selected prefix branch ledger is exactly the public comparison datum exported by the $\mathsf{Order}^{\uparrow}$ certificate. The ledger contains one of the concrete rows*

$$\mathsf{NatUnaryStrictPrefix}(h,k), \qquad \rho_{\mathsf{u}}(h,k), \qquad \mathsf{NatUnaryStrictPrefix}(k,h),$$

and contains no separate classical trichotomy row, no host order row, and no quotient-equality row.

Proof. Read the selected ledger row through Definition 54.80. The row is one of the three concrete unary-prefix branches and is therefore a BHist witness, a classifier witness, or the reversed BHist witness.

The singleton boundary of Definition 54.79 records the same policy at the certificate boundary: only carrier, comparison branch, classifier, stability, exactness, and selected ledger rows are exported. The unary-prefix surface uses the same boundary shape, replacing the singleton branches by the unary branches of Definition 54.9.

Thus the ledger is exact as a public comparison datum. Since every exported row is one of the displayed concrete rows, there is no additional classical trichotomy row to project. □

Theorem 54.82 ($\mathsf{Order}^{\uparrow}$ unary-prefix audit target surface)**.** *The unary-prefix $\mathsf{Order}^{\uparrow}$ paper target consists of the carrier $\mathsf{Carrier_u}$, the classifier ρ_{u}, the order $\leq_{\mathsf{u}}$, the branch readback of Definition 54.80, the stability rows of Definition 54.74, the ledger exactness of Definition 54.81, and the certificate boundary exactness of Definition 54.79. This is the complete paper surface for a later theorem over the concrete unary-prefix carrier and does not require parameter transport or classical order primitives.*

Proof. Start with the carrier, classifier, and order from the concrete unary-prefix total-order instance. The comparison partition supplies the constructive strict-left, same, and strict-right branches over $\mathsf{Carrier_u}$.

Use Definition 54.80 to read the selected branch as public BHist data. Use Definition 54.81 to identify the selected ledger with that public comparison datum.

Finally repack these rows with the stability certificate of Definition 54.74 and the exact boundary of Definition 54.79. The assembled target names precisely the carrier, classifier, branch readback, stability, ledger, and exactness rows needed by the unary-prefix $\mathsf{Order}^{\uparrow}$ theorem surface. □

Theorem 54.83 (Concrete singleton history $\mathsf{TotalOrder}^{\uparrow}$ naming certificate)**.** *The concrete tuple of Definition 54.2 supplies the source, pattern, classifier, stability, ledger, and exactness rows for a $\mathsf{NameCert}$ licensing $\mathsf{TotalOrder}^{\uparrow}$ over the singleton carrier A_{e}, classifier ρ_{e}, order $\leq_{\mathsf{e}}$, and comparison branches $\mathsf{lt_e} := \bot$, $\mathsf{same_e} := \rho_{\mathsf{e}}$, and $\mathsf{gt_e} := \bot$.*

Proof. The carrier row is the singleton BHist source. Apply Definition 54.3 to obtain the carried endpoint facts, classifier stability, the $\mathsf{POSet}^{\uparrow}$ fields, trichotomy, and comparison stability for the displayed branches.

The exactness row is the underlying partial-order exactness supplied by Definition 53.12: on carried histories, $\rho_{\mathsf{e}}(h,k)$ is equivalent to the two opposing prefix comparisons $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$.

The strict rows contribute no extra witnesses. Both strict comparison branches are empty by definition, so strict-to-apartness and strict transitivity are vacuous.

It remains to assemble the public certificate. Repack the source as A_{e}, the pattern as the three-way comparison formation, the classifier as ρ_{e}, the stability data as the fields just cited, and the ledger as the selected same-branch comparison certificate. □

The per-object template is: traditional target singleton constructive total order; BEDC source singleton BHist; carrier A_{e}; classifier ρ_{e}; stability obligations from Definition 54.83; ledger the selected same-branch comparison certificate; exactness singleton classifier iff opposing prefix comparisons; standard bridge singleton total order; status $\mathsf{ProofTarget}$.

Primitive boundary. $\mathsf{TotalOrder}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 54.84 (Zero-axiom encoding note)**.** *Replaces LEM-style totality with decidable comparison data and apartness.*

54.9 Public audit boundary

Definition 54.85 (TotalOrder OrderUp public audit carrier)**.** *The public audit carrier for the unary-prefix* $\mathsf{TotalOrder}^{\uparrow}$ *to* $\mathsf{Order}^{\uparrow}$ *surface is the carried endpoint predicate on* BHist *supplied by* $\mathsf{Carrier_u}$*. Its classifier is the unary classifier* ρ_{u}*, and its comparison relation is* $\leq_{\mathsf{u}} := \mathsf{PreorderPrefixLE}$*. Thus every public audit row is anchored in carried* BHist *endpoints, not in a host order on natural numbers.*

Theorem 54.86 (TotalOrder OrderUp public audit classifier)**.** *For carried unary endpoints* $h, k : \mathsf{BHist}$ *in* $\mathsf{Carrier_u}$*, classifier transport for the public audit surface is read from Definition 54.13, and antisymmetry is read from Definition 54.9. Hence the public classifier row is exactly* $\rho_{\mathsf{u}}(h, k)$ *together with its transport through the three constructive comparison branches.*

The ten-field fit is: downstream target the public $\mathsf{Order}^{\uparrow}$ *audit surface; BEDC source* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability branch transport; ledger selected comparison branch; core theorem this classifier audit row; exactness opposing-prefix antisymmetry from the comparison partition; standard bridge finite unary words; status* $\mathsf{ProofTarget}$*.*

Proof. Fix carried endpoints h, k and a classifier replacement along ρ_{u}. Apply Definition 54.13. It transports the strict-left, same, and strict-right branch rows across the replacement, so the public comparison branch remains a row over the carried endpoints.

Now read antisymmetry from Definition 54.9. The same branch is $\rho_{\mathsf{u}}(h, k)$, while opposing prefix comparisons exclude the two strict branches and recover the middle classifier row.

Combining these two cited rows gives the audit classifier boundary. The transport datum is branch transport, and the exactness datum is precisely the middle branch selected by the comparison partition. □

Theorem 54.87 (TotalOrder OrderUp public constructive comparison)**.** *Every carried pair* $h, k : \mathsf{BHist}$ *in* $\mathsf{Carrier_u}$ *exposes exactly one constructive comparison branch for the public* $\mathsf{Order}^{\uparrow}$ *export: a strict unary prefix from* h *to* k*, the classifier row* $\rho_{\mathsf{u}}(h, k)$*, or a strict unary prefix from* k *to* h*. This is the comparison datum consumed by Definition 54.63.*

The ten-field fit is: downstream target constructive $\mathsf{Order}^{\uparrow}$ *comparison; BEDC source* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability branch transport; ledger selected branch; core theorem this constructive comparison row; exactness the unique active branch; standard bridge finite unary-word comparison; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 54.9 to the carried pair h, k. The theorem supplies the three branch rows directly over unary BHist endpoints and states their incompatibility.

The public export of Definition 54.63 reads the same carrier, classifier, and relation. Therefore the selected branch from the comparison partition is already the branch consumed by the public $\mathsf{Order}^{\uparrow}$ surface.

It remains to identify uniqueness at the audit boundary. Since the partition excludes every second branch and does not mention a host order, the public row is constructive branch data rather than classical trichotomy. □

Theorem 54.88 (TotalOrder OrderUp public ledger exactness)**.** *The public* $\mathsf{Order}^{\uparrow}$ *audit ledger is exhausted by carried* BHist *endpoints, the selected constructive branch witness, classifier transport, and finite interval rows. Its finite rows are the rows named by Definition 54.56 and Definition 54.61; no host order, quotient row, or classical trichotomy row is part of the public ledger.*

The ten-field fit is: downstream target public $\mathsf{Order}^{\uparrow}$ *ledger exactness; BEDC source* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability finite branch transport; ledger carried endpoints, branch witness, classifier row, and interval rows; core theorem this ledger exactness boundary; exactness finite* Cont *exhaustion; standard bridge finite unary-word intervals; status* ProofTarget*.*

Proof. Start with the public carrier and comparison branch from Definition 54.63. The export supplies UnaryHistory, ρ_{u}, and $\leq_{\mathsf{u}}$, and the preceding theorem identifies the selected comparison branch as constructive unary-prefix data.

Next apply Definition 54.56. It records the finite Cont rows attached to carried endpoints and the selected branch. Apply Definition 54.61 to confine exactness to those finite rows.

The cited rows account for every exported ledger component: endpoints, branch witness, classifier transport, and interval data. Since none of the cited rows introduces host order, quotient equality, or classical trichotomy, no such row is available at the audit boundary. □

Theorem 54.89 (TotalOrder public bridge boundary)**.** *The public* $\mathsf{TotalOrder}^{\uparrow}$ *surface supplies the* $\mathsf{Order}^{\uparrow}$ *phase boundary through the unary-prefix package of Definition 54.45 and the public export of Definition 54.63. It does not claim* $\mathsf{Real}^{\uparrow}$ *completeness, classical trichotomy, host equality, or any bridge beyond the finite unary-word surface stated in Definition 54.65.*

The ten-field fit is: downstream target $\mathsf{Order}^{\uparrow}$ *phase boundary; BEDC source* BHist*; carrier* $\mathsf{Carrier_u}$*; classifier* ρ_{u}*; relation* $\leq_{\mathsf{u}}$*; stability branch transport and endpoint exactness; ledger finite unary rows; core theorem this public bridge boundary; exactness public ledger exhaustion; standard bridge finite unary words only; status* ProofTarget*.*

Proof. Read the phase package from Definition 54.45. It supplies the carrier, classifier, relation, comparison, interval, endpoint, and exactness rows required for the $\mathsf{Order}^{\uparrow}$ phase boundary.

Read the public export from Definition 54.63. It repacks the same rows as the public $\mathsf{Order}^{\uparrow}$ surface, and Definition 54.88 identifies the available ledger data.

Finally apply Definition 54.65. The standard reading is finite unary-word data only. Thus the public boundary does not add $\mathsf{Real}^{\uparrow}$ completeness, classical trichotomy, host equality, or a larger bridge claim. □

54.10 Root obligation surface

Theorem 54.90 (TotalOrder root carrier obligation)**.** *The root* $\mathsf{TotalOrder}^{\uparrow}$ *obligation surface is carried by the concrete BEDC history rows*

$$\mathsf{BHist}, \qquad \mathsf{UnaryHistory}(h), \qquad \mathsf{UnaryHistory}(k),$$

together with the inherited $\mathsf{POSet}^{\uparrow}$ *carrier used by the unary-prefix certificate. For every carried pair* $h, k : \mathsf{BHist}$*, the root carrier row is therefore the same* $\mathsf{UnaryHistory}$ *row consumed by the downstream* $\mathsf{Order}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *order-front packages, not a separate host-order carrier.*

Proof. Start with the carrier named in Definition 54.45. It is the BHist predicate $\mathsf{UnaryHistory}$, and the phase package reads it through the inherited $\mathsf{POSet}^{\uparrow}$ surface of Definition 54.21.

The same row is projected in Definition 54.47. Thus a root consumer sees carried histories h and k only through $\mathsf{UnaryHistory}(h)$ and $\mathsf{UnaryHistory}(k)$.

Because the public audit boundary of Definition 54.85 uses the same endpoint predicate, repacking the carrier for $\mathsf{Order}^{\uparrow}$ or $\mathsf{Lattice}^{\uparrow}$ does not introduce a second carrier. The root obligation is exactly the displayed BHist carrier row. □

Theorem 54.91 (TotalOrder root classifier obligation)**.** *For carried unary histories* $h, k : \mathsf{BHist}$*, the root classifier row is the inherited* $\mathsf{POSet}^{\uparrow}$ *classifier* $\rho_{\mathsf{u}}(h, k)$*, read as* hsame*-stable endpoint sameness on* $\mathsf{UnaryHistory}$ *rows. The explicit comparison branches at the root are*

$$\mathsf{NatUnaryStrictPrefix}(h, k), \qquad \rho_{\mathsf{u}}(h, k), \qquad \mathsf{NatUnaryStrictPrefix}(k, h),$$

and classifier transport preserves which of these three branch rows is recorded by the comparison ledger.

Proof. Apply Definition 54.13. It states that the strict-left, same, and strict-right comparison rows are transported along the unary classifier without changing the concrete endpoint carrier.

The middle branch is the inherited classifier row $\rho_{\mathsf{u}}(h, k)$. Read this through Definition 54.86, which identifies public classifier transport with the same three constructive comparison branches.

Finally, the hsame reading is only endpoint transport inside BHist. Since the carrier remains $\mathsf{UnaryHistory}$, the root classifier obligation is the inherited $\mathsf{POSet}^{\uparrow}$ classifier and not an external equality predicate. □

Theorem 54.92 (TotalOrder root exactness obligation)**.** *On carried unary histories* $h, k : \mathsf{BHist}$*, the root exactness row says that two-sided comparison under the inherited* $\mathsf{POSet}^{\uparrow}$ *relation* $\leq_{\mathsf{u}}$ *recovers the classifier branch* $\rho_{\mathsf{u}}(h, k)$*, while the two strict prefix branches are excluded. This exactness row is the* BHist *endpoint exactness consumed by the public* $\mathsf{Order}^{\uparrow}$ *surface and by any* $\mathsf{Lattice}^{\uparrow}$ *consumer that only reads the displayed unary-prefix order front.*

Proof. First use Definition 54.51. It identifies opposing $\leq_{\mathsf{u}}$ comparisons on carried endpoints with the unary classifier ρ_{u}.

Next use Definition 54.9. The partition separates strict-left, same, and strict-right branch rows, so the two-sided comparison case selects the same branch and excludes the two strict prefix alternatives.

The public exactness row of Definition 54.88 records this as endpoint exactness in the audit ledger. A downstream $\mathsf{Order}^{\uparrow}$ or $\mathsf{Lattice}^{\uparrow}$ consumer therefore receives exactness for the unary-prefix branch surface and no quotient equality row. □

Theorem 54.93 (TotalOrder root comparison ledger obligation)**.** *The root comparison ledger for carried unary histories records exactly one of the three concrete* BHist *rows:*

$$\mathsf{NatUnaryStrictPrefix}(h, k), \qquad \rho_{\mathsf{u}}(h, k), \qquad \mathsf{NatUnaryStrictPrefix}(k, h).$$

Together with the carrier rows $\mathsf{UnaryHistory}(h)$ *and* $\mathsf{UnaryHistory}(k)$*, this ledger is the comparison datum inherited by the* $\mathsf{Order}^{\uparrow}$ *package and by the visible* $\mathsf{Lattice}^{\uparrow}$ *order-front obligations; it contains no host natural-order row and no classical trichotomy row.*

Proof. Read the selected branch from Definition 54.80. The branch is already one of the two strict unary-prefix witnesses or the classifier row $\rho_{\mathsf{u}}(h, k)$.

Now apply Definition 54.81. It states that the selected branch is the public comparison ledger and that no additional host-order or quotient row is present.

The public audit boundary of Definition 54.87 uses the same branch row for the $\mathsf{Order}^{\uparrow}$ export. Any $\mathsf{Lattice}^{\uparrow}$ consumer reading this front therefore inherits the displayed comparison ledger rather than a new comparison schema. □

Theorem 54.94 (TotalOrder root scope obligation)**.** *The root* $\mathsf{TotalOrder}^{\uparrow}$ *obligation surface consists exactly of the* BHist *carrier* $\mathsf{UnaryHistory}$*, the inherited* $\mathsf{POSet}^{\uparrow}$ *classifier* ρ_{u}*, the relation* $\leq_{\mathsf{u}}$*, the three explicit comparison branches, their* hsame*-stable transport, the comparison ledger, and the endpoint exactness row. These rows are sufficient for the displayed* $\mathsf{Order}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *order-front consumers, and the root surface does not claim completeness, host equality, quotient completion, or a parameter transport principle.*

Proof. Assemble the carrier and classifier rows from Definition 54.90 and Definition 54.91. These fix the BHist endpoints, $\mathsf{UnaryHistory}$, hsame transport, and ρ_{u} classifier.

Add the exactness and ledger rows from Definition 54.92 and Definition 54.93. They supply the selected comparison branch and the two-sided comparison recovery of the same branch.

Finally, project the assembled rows through Definition 54.45 and Definition 54.89. The downstream consumers read only this concrete unary-prefix order front, so the root scope is closed over the displayed BEDC rows and does not add a parameter-transport schema. □

Theorem 54.95 (TotalOrder lattice sibling route)**.** *The* $\mathsf{TotalOrder}^{\uparrow}$ *packet consumes the inherited* $\mathsf{Preorder}^{\uparrow}$ *and* $\mathsf{POSet}^{\uparrow}$ *comparison spine and supplies the comparable-pair row needed by finite* $\mathsf{Lattice}^{\uparrow}$ *meet and join routes. A lattice consumer may read the TotalOrder comparison branch as an order-front input for the displayed finite meet-or-join construction, but it cannot extract a separate lattice law, host order equality, quotient carrier equality, or a global completeness principle.*

Proof. Read the inherited order rows from the TotalOrder root obligation surface. They expose the $\mathsf{Preorder}^{\uparrow}$ reflexive-transitive spine together with the $\mathsf{POSet}^{\uparrow}$ antisymmetry and classifier-exactness rows already used by the unary-prefix package.

The TotalOrder-specific row adds only comparability for the displayed pair. That is exactly the order-front datum a finite $\mathsf{Lattice}^{\uparrow}$ meet or join route needs before it can choose the lower or upper branch inside its own certificate.

The route is therefore horizontal: TotalOrder supplies comparable-pair input to Lattice, while Lattice retains its own meet, join, absorption, and classifier rows. No host order equality, quotient carrier equality, separate lattice law, or completeness endpoint is introduced by the sibling edge. □

54.11 Unary Prefix History Instance

Definition 54.96 (TotalOrder unary-prefix BHist instance)**.** *Let* $\mathsf{UnaryHistory}(h)$ *mean that* $h : \mathsf{BHist}$ *lies on a single visible unary continuation spine from* e*. The carrier for the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *instance is*

$$\mathsf{UnaryHistoryCarrier}(h) := \mathsf{UnaryHistory}(h).$$

The classifier is endpoint agreement inside the unary carrier:

$$h \sim_{\mathsf{unary}} k := \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{hsame}(h, k).$$

The order relation is visible prefix reachability in BHist*:*

$$h \preceq_{\mathsf{unary}} k := \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \exists t : \mathsf{BHist},\ \mathsf{Cont}(h, t, k).$$

Thus the order data are built from BHist*,* UnaryHistory*,* hsame*, and* Cont *rather than from abstract comparison parameters.*

Theorem 54.97 (TotalOrder unary-prefix BHist laws)**.** *The carrier, classifier, and order relation of Definition 54.96 satisfy the concrete* $\mathsf{TotalOrder}^{\uparrow}$ *laws over unary histories:*

1. *reflexivity is witnessed by the empty tail at each unary history;*
2. *transitivity is witnessed by associating the two visible continuation tails;*
3. *antisymmetry sends mutual prefix witnesses on unary histories to endpoint equality, hence to the classifier* $h \sim_{\mathsf{unary}} k$*;*
4. *totality is the unary-prefix comparison: for any two unary histories, one displayed continuation spine is a prefix of the other.*

Consequently the total-order schema has a concrete BHist *instance and is not only an abstract operator interface.*

Proof. For reflexivity, take the empty tail in BHist. The continuation route $\mathsf{Cont}(h, \mathsf{e}, h)$ is the visible endpoint read for a unary history, and the two carrier witnesses are the same $\mathsf{UnaryHistory}(h)$ row.

For transitivity, suppose $h \preceq_{\mathsf{unary}} k$ through tail t_1 and $k \preceq_{\mathsf{unary}} \ell$ through tail t_2. Continuation associativity inside BHist gives a visible composite tail from h to ℓ, while the unary carrier rows are already part of the two prefix witnesses.

For antisymmetry, mutual prefix witnesses place h and k on the same unary spine with the same endpoint. Unary histories have no branching coordinate, so opposing visible tails collapse to $\mathsf{hsame}(h, k)$; repacking this witness with the two carrier rows gives $h \sim_{\mathsf{unary}} k$.

For totality, compare the two unary continuation spines from e. Since each history is unary, the first point at which the displayed finite paths stop determines which endpoint is reached by a visible tail from the other. This gives either $h \preceq_{\mathsf{unary}} k$ or $k \preceq_{\mathsf{unary}} h$ using only BHist, UnaryHistory, hsame, Cont, and the existing total-order carrier rows. □

Theorem 54.98 (Unary-prefix comparison readback totality)**.** *Let* h, k : BHist *be carried by the concrete unary-prefix instance of Definition 54.96. Every public comparison read for this* $\mathsf{TotalOrder}^{\uparrow}$ *instance is exhausted by the following BEDC rows: the two carrier rows* $\mathsf{UnaryHistory}(h)$ *and* $\mathsf{UnaryHistory}(k)$*, exactly one displayed continuation tail witnessing* $h \preceq_{\mathsf{unary}} k$ *or* $k \preceq_{\mathsf{unary}} h$*, the endpoint classifier row* $\mathsf{hsame}(h, k)$ *in the equality branch, and the ambient* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structural rows. Thus the comparison readback exposes no host* Nat *order, quotient order, external trichotomy principle, or second comparison relation.*

Proof. Project the carrier, classifier, and prefix order from Definition 54.96. The only carried objects available to the public read are the two rows $\mathsf{UnaryHistory}(h)$ and $\mathsf{UnaryHistory}(k)$, and the only order witness available inside the instance is a displayed BHist tail through Cont.

Apply Definition 54.97. Its totality clause splits the read into the left-prefix case, the same-endpoint case, or the right-prefix case. In the left-prefix case the visible row is a tail t : BHist with $\mathsf{Cont}(h, t, k)$; in the right-prefix case it is a tail t : BHist with $\mathsf{Cont}(k, t, h)$.

In the same-endpoint case, antisymmetry in the same laws collapses the opposing visible tails to the classifier row $\mathsf{hsame}(h, k)$. Repacking this row with the two carrier witnesses gives exactly the equality branch of the concrete classifier $h \sim_{\mathsf{unary}} k$.

The package and name-certificate rows only record that this carrier, this classifier, and this prefix relation are the public $\mathsf{TotalOrder}^{\uparrow}$ surface. Since the instance contains no coordinate for a host Nat order, a quotient order, a trichotomy oracle, or any parallel comparison relation, there is no additional readback component to expose. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotalOrder}^{\uparrow}}$.

Constructive story: The total-order surface keeps the unary-prefix carrier from $\mathsf{Poset}^{\uparrow}$ and adds only public selector rows saying that, on the exposed finite unary endpoints, a comparison phase can choose the relevant directed prefix evidence. The accepted objects are still unary histories over BHist, with b0-headed histories excluded, and the classifier is still the carrier-aware hsame row that antisymmetry reads back from opposing comparisons. The source is the public unary-prefix order packet, the pattern records finite extrema, interval collapse, phase-exit, root-obligation, and certificate-boundary rows over the same prefix relation, and the ledger keeps each selector decision tied to explicit continuation witnesses rather than to a classical trichotomy oracle. The five name-certificate obligations are inherited from the preorder and poset layers: e inhabits the carrier, hsame remains reflexive, symmetric, and transitive, and any classified endpoint stays inside the unary carrier. A history h lies in the signature gap of a $\mathsf{TotalOrder}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer endpoint and audit ledger, and token introduction names that answer as p; unfolded, h is a unary endpoint whose comparison evidence, extrema row, and phase-exit data match the public token. There is no checked standard bridge here; the public audit boundary exports the finite unary-prefix selector surface only, leaving arbitrary total orders and classical trichotomy outside the closure. The chapter registers this selector-refined prefix package under the name $\mathsf{TotalOrder}^{\uparrow}$; totality is presented as public finite unary comparison data, not as an imported order type, quotient completion, or lattice structure.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TotalOrder}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public $\mathsf{TotalOrder}^{\uparrow}$ certificate surface is exported through the unary-prefix carrier, classifier, finite-extrema, phase-exit, certificate-boundary, public-audit, and root-obligation rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TotalOrder}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PreorderUp.preorder\_prefix\_stability\_certificate\_fields`

Bridge status: paperBridge

Not claimed: It does not close the full TotalOrder bridge, arbitrary total orders, classical trichotomy, quotient completion, or lattice structure.

Upgrade path: No further paper-axis closure upgrade remains for the unary-prefix $\mathsf{TotalOrder}^{\uparrow}$ surface.

Chapter 55

A Concrete Naming Certificate for $\mathsf{Lattice}^{\uparrow}$

The lattice interface $\mathsf{Lattice}^{\uparrow}$ extends chapter 53 with binary meet $\wedge$ and join $\vee$ operations, each classified as the greatest lower bound and least upper bound of two elements under the partial order. Idempotence, commutativity, associativity, and absorption laws are derived from the bound characterizations under the carrier's classifier.

55.1 Carrier definition

Definition 55.1 (Lattice carrier)**.** *Use a poset setup extended with meet and join operations carrying bound-characterization proofs.*

55.2 Concrete singleton prefix lattice instance

Definition 55.2 (Concrete singleton prefix lattice instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define ρ_{e} by*

$$\rho_{\mathsf{e}}(h, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Let $h \leq_{\mathsf{e}} k$ mean that $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ hold and that $\mathsf{PreorderPrefixLE}(h, k)$ witnesses the prefix comparison between the carried endpoints. Define the concrete lattice operations by

$$\mathsf{meet}(h, k) := \mathsf{e}, \qquad \mathsf{join}(h, k) := \mathsf{e}.$$

The certificate has ten fields: traditional target the one-point lattice; BEDC source BHist *plus the singleton source and concrete prefix order; carrier A_{e}; classifier ρ_{e}; stability obligations for poset laws, meet lower bounds, meet greatestness, join upper bounds, join leastness, and meet and join congruence; ledger singleton endpoint plus prefix witnesses; core theorem Theorem 55.62; exactness singleton endpoint classification; standard bridge terminal one-point lattice; and status* ProofTarget*.*

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeUp StdBridge`

Theorem 55.3 (Concrete singleton prefix lattice laws)**.** *For all carried histories h, k, z in A_{e}, the endpoints $\mathsf{meet}(h, k)$ and $\mathsf{join}(h, k)$ are carried by A_{e} and satisfy the meet lower-bound laws, meet greatest-lower-bound law, join upper-bound laws, join least-upper-bound law, and meet and join congruence laws for the restricted prefix order $\leq_{\mathsf{e}}$ under the classifier ρ_{e}.*

Proof. The singleton carrier witnesses identify each carried endpoint with e. Thus $A_{\mathsf{e}}(h)$, $A_{\mathsf{e}}(k)$, and $A_{\mathsf{e}}(z)$ reduce every endpoint comparison needed by the order, meet, and join clauses to the empty-history endpoint.

The restricted prefix comparisons are supplied by PreorderPrefixLE_of_hsame after the singleton witnesses produce the relevant hsame facts. The inherited order boundary is the concrete unary prefix poset law package of Theorem 53.10, with antisymmetry supplied by Theorem 53.33.

The meet and join outputs are both definitionally e. The lower-bound and upper-bound clauses therefore use the singleton endpoint witness for e and the prefix witnesses from the previous paragraph; the greatest-lower-bound and least-upper-bound clauses use the same reduction for the third carried history z.

Congruence is immediate because both operations ignore their operands and return e. After unfolding, every source and target output is classified by $\rho_{\mathsf{e}}(\mathsf{e},\mathsf{e})$, assembled from two copies of $A_{\mathsf{e}}(\mathsf{e})$ and the reflexive witness $\mathsf{hsame}(\mathsf{e},\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonPrefix laws`

Theorem 55.4 (Singleton lattice order has empty endpoints)**.** *For h, k : BHist, the restricted singleton prefix order satisfies*

$$h \leq_{\mathsf{e}} k \quad \Longleftrightarrow \quad h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction reads the two singleton carrier fields from the definition of $h \leq_{\mathsf{e}} k$. These fields are exactly $h$hsamee and $k$hsamee.

For the reverse direction, the two endpoint witnesses supply the singleton carrier fields. Their transitive composite through e gives $h$hsame$k$, and the prefix comparison follows from the concrete PreorderPrefixLE_of_hsame constructor. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE empty endpoints iff`

Theorem 55.5 (Singleton lattice append order has empty endpoints)**.** *For h, k : BHist, the restricted singleton prefix order from the appended history to* e *satisfies*

$$\mathsf{LatticeSingletonLE}(\mathsf{append}(h,k),\mathsf{e}) \quad \Longleftrightarrow \quad h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction first applies singleton-order endpoint exactness to read $\mathsf{append}(h,k)\mathsf{hsamee}$, then splits the empty append into empty left and right endpoints. Conversely, empty endpoints make $\mathsf{append}(h,k)$ history-same to e, and the singleton-order endpoint characterization rebuilds the restricted order comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append empty endpoints iff`

Theorem 55.6 (Singleton lattice carrier order collapse)**.** *If h and k are carried by A_{e}, then both restricted order comparisons hold, the singleton classifier relates h and k, and both endpoints are internally the same as* e*:*

$$h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h \wedge \rho_{\mathsf{e}}(h,k) \wedge h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The two carrier witnesses are the endpoint comparisons $h$hsamee and $k$hsamee. Their composite through e gives $h$hsame$k$, and symmetry gives $k$hsame$h$.

The two restricted order comparisons are obtained from these two sameness witnesses by PreorderPrefixLE_of_hsame, together with the same carrier fields. The classifier package uses the carrier fields and $h$hsame$k$, and the endpoint clauses are the original carrier witnesses. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonCarrier order collapse`

Theorem 55.7 (Singleton lattice antisymmetry classifier)**.** *If $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$ in the restricted singleton prefix order, then $\rho_{\mathsf{e}}(h, k)$, $h\mathsf{hsamee}$, and $k\mathsf{hsamee}$.*

Proof. The two restricted comparisons carry their endpoint fields, so the first comparison already identifies both h and k with e. The prefix components of the two comparisons are opposing $\mathsf{PreorderPrefixLE}$ witnesses, and concrete prefix antisymmetry gives $\mathsf{hsame}(h, k)$. These three pieces are exactly the classifier data $\rho_{\mathsf{e}}(h, k)$, together with the two displayed empty endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE antisymm classifier`

Theorem 55.8 (Singleton lattice meet-join absorption classifier)**.** *If h and k are carried by A_{e}, then both absorption composites $\mathsf{meet}(h, \mathsf{join}(h, k))$ and $\mathsf{join}(h, \mathsf{meet}(h, k))$ are classified with h by ρ_{e}, and both output histories are hsame to e.*

Proof. Both concrete lattice operations are definitionally e. Hence each absorption composite is the empty history, whose singleton carrier witness is reflexive. The carrier witness for h gives $h\mathsf{hsamee}$, so symmetry gives the classifier component from each absorption output back to h. Reassemble the two ρ_{e} classifiers and record the two empty output endpoints. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption classifier`

Theorem 55.9 (Singleton lattice meet-join absorption is empty-classified)**.** *For carried singleton endpoints h, k, both absorption composites reduce to the empty endpoint and classify back to h:*

$$\mathsf{meet}(h, \mathsf{join}(h, k))\mathsf{hsamee}, \qquad \mathsf{join}(h, \mathsf{meet}(h, k))\mathsf{hsamee},$$
$$\rho_{\mathsf{e}}(\mathsf{meet}(h, \mathsf{join}(h, k)), h), \qquad \rho_{\mathsf{e}}(\mathsf{join}(h, \mathsf{meet}(h, k)), h).$$

Proof. Both concrete operations are definitionally the endpoint e. The two endpoint equalities are therefore reflexive empty-history sameness. Since h is carried by the singleton carrier, symmetry of its carrier witness classifies e with h, and the classifier packages the two carrier witnesses with that history-sameness witness. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption empty classified`

Theorem 55.10 (Singleton lattice continuation absorption has empty classifier)**.** *Let h, k be singleton-carried histories. If $left$ is a BEDC continuation output of $\mathsf{meet}(h, \mathsf{join}(h, k))$ followed by h, and $right$ is a BEDC continuation output of h followed by $\mathsf{join}(h, \mathsf{meet}(h, k))$, then both outputs are empty and are classified by the singleton classifier:*

$$\mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}) \wedge \rho_{\mathsf{e}}(left, right).$$

Proof. The concrete singleton meet and join both return e. The left continuation is therefore $\mathsf{append}(\mathsf{e}, h)$, which reduces to h, and the right continuation is $\mathsf{append}(h, \mathsf{e})$, which also reduces to h. The carrier witness for h identifies each result with e. Those two endpoint witnesses, together with transitivity through e, assemble the singleton classifier between the two continuation outputs. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin continuation absorption empty classifier`

Theorem 55.11 (Singleton lattice meet-join idempotent commutative classifier)**.** *For carried singleton endpoints h, k, meet and join are classified with the idempotent endpoint and with their swapped arguments:*

$$\rho_{\mathsf{e}}(\mathsf{meet}(h,h),h) \wedge \rho_{\mathsf{e}}(\mathsf{join}(h,h),h) \wedge \rho_{\mathsf{e}}(\mathsf{meet}(h,k),\mathsf{meet}(k,h)) \wedge \rho_{\mathsf{e}}(\mathsf{join}(h,k),\mathsf{join}(k,h)).$$

Moreover $\mathsf{meet}(h,k)\mathsf{hsamee}$ *and* $\mathsf{join}(h,k)\mathsf{hsamee}$*.*

Proof. The concrete singleton meet and join operations both return e. Thus the swapped meet outputs and swapped join outputs are classified by reflexive empty-history sameness. Since h is carried by the singleton carrier, symmetry of $h\mathsf{hsamee}$ classifies each idempotent output with h. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin idempotent comm classifier`

Theorem 55.12 (Singleton lattice classifier has empty endpoints)**.** *For $h, k : \mathsf{BHist}$, the singleton lattice classifier satisfies*

$$\rho_{\mathsf{e}}(h,k) \iff h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction projects the two carrier fields from the classifier tuple. Since the singleton carrier is the predicate $x\mathsf{hsamee}$, these fields are exactly the two endpoint witnesses.

Conversely, the two endpoint witnesses make h and k singleton-carried. Their composite through e, using symmetry on the second witness, supplies $h\mathsf{hsamee}k$ and hence the classifier tuple. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier empty endpoints iff`

Theorem 55.13 (Singleton lattice classifier continuation result-left iff)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P,Q,R)$, the singleton lattice classifier relates P and Q exactly when it relates the continuation result R to P:*

$$\mathsf{LatticeSingletonClassifier}(P,Q) \iff \mathsf{LatticeSingletonClassifier}(R,P).$$

Proof. The forward direction uses the singleton carriers of P and Q to transport the continuation result to the right-unit continuation at e, so R is singleton-carried and compares with P through e. For the reverse direction, the result-left classifier identifies R with e; transporting the continuation to empty result and applying empty-result inversion recovers the singleton carrier witnesses for both inputs, and hence the input classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result left iff`

Theorem 55.14 (Singleton lattice classifier continuation result-right iff)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P,Q,R)$, the singleton lattice classifier relates P and Q exactly when it relates the continuation result R to Q:*

$$\mathsf{LatticeSingletonClassifier}(P,Q) \iff \mathsf{LatticeSingletonClassifier}(R,Q).$$

Proof. The forward direction uses the singleton carriers of P and Q to transport the continuation result to the right-unit continuation at e, so R is singleton-carried and compares with Q through e. For the reverse direction, the result-right classifier identifies R with e; transporting the continuation to empty result and applying empty-result inversion recovers the singleton carrier witnesses for both inputs, and hence the input classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result right iff`

Theorem 55.15 (Singleton lattice classifier continuation empty result exactness)**.** *For histories* P, Q, R : BHist *with* Cont(P, Q, R)*, the singleton lattice classifier relates the two inputs exactly when the continuation result is empty:*

$$\mathsf{LatticeSingletonClassifier}(P, Q) \iff R\mathsf{hsamee}.$$

Proof. The forward direction reads the two singleton carrier endpoints from the input classifier; since R is the append continuation of P and Q, append-empty construction gives Rhsamee. Conversely, result emptiness inverts the append continuation into singleton carrier witnesses for both inputs, and the empty-endpoint characterization reconstructs the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation empty result iff`

Theorem 55.16 (Singleton lattice classifier continuation result-left-right equivalence)**.** *For histories* P, Q, R : BHist *with* Cont(P, Q, R)*, classifying the continuation result against the left input is equivalent to classifying it against the right input:*

$$\mathsf{LatticeSingletonClassifier}(R, P) \iff \mathsf{LatticeSingletonClassifier}(R, Q).$$

Proof. The result-left classifier theorem identifies the left comparison with the input classifier on P and Q. The result-right classifier theorem identifies the right comparison with the same input classifier. Composing those two characterizations gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result left right iff`

Theorem 55.17 (Singleton lattice result-left order carriers)**.** *For histories* h, k, r : BHist *with* Cont(h, k, r)*, the restricted singleton order from the continuation result back to the left input is equivalent to both exposed inputs being singleton-carried:*

$$\mathsf{LatticeSingletonLE}(r, h) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction reads $r$hsamee from the singleton order endpoint fields. The continuation witness identifies r with append(h, k), so empty-append inversion gives $h$hsamee and $k$hsamee.

Conversely, if both exposed inputs are singleton-carried, the continuation result is internally the empty endpoint. The empty-endpoint characterization of the singleton order then supplies LatticeSingletonLE(r, h). □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result left carriers iff`

Theorem 55.18 (Singleton lattice result-right order carriers)**.** *For histories* h, k, r : BHist *with* Cont(h, k, r)*, the restricted singleton order from the continuation result back to the right input is equivalent to both exposed inputs being singleton-carried:*

$$\mathsf{LatticeSingletonLE}(r, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction again projects $r$hsamee from the singleton order endpoint fields. Since the continuation result is the append of h and k, empty-append inversion exposes both input carriers.

In the reverse direction, the two input carriers force the continuation result to be internally empty. The empty-endpoint characterization of LatticeSingletonLE then gives the displayed order comparison against the right input. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result right carriers iff`

Theorem 55.19 (Singleton lattice continuation result endpoint order equivalence)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the restricted singleton order from the continuation result to the left endpoint is equivalent to the restricted singleton order from the result to the right endpoint:*

$$\mathsf{LatticeSingletonLE}(R, P) \iff \mathsf{LatticeSingletonLE}(R, Q).$$

Proof. The result-left order theorem identifies the left comparison with the pair of singleton carrier witnesses for P and Q. The result-right order theorem identifies the right comparison with the same pair. Composing these two characterizations gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result left right iff`

Theorem 55.20 (Singleton lattice continuation empty result order exactness)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the restricted singleton prefix comparison between the inputs is equivalent to emptiness of the continuation result:*

$$P \leq_{\mathsf{e}} Q \iff R\mathsf{hsamee}.$$

Proof. The restricted singleton order carries both input endpoints as empty. Since R is the BEDC append continuation of P and Q, append-empty construction gives $R\mathsf{hsamee}$. Conversely, if the continuation result is empty, BEDC append-empty inversion makes both P and Q empty. Their induced hsame chain supplies the underlying prefix comparison, so the singleton order tuple is reconstructed. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation empty result iff`

Theorem 55.21 (Singleton meet–join absorption gives order and classifier data)**.** *If* $h, k : \mathsf{BHist}$ *are singleton-carried, then the two singleton lattice absorption endpoints*

$$h \wedge (h \vee k) \qquad \textit{and} \qquad h \vee (h \wedge k)$$

are both definitionally the empty history. They satisfy the restricted prefix order comparisons

$$h \wedge (h \vee k) \leq_{\mathsf{e}} h \qquad \textit{and} \qquad h \leq_{\mathsf{e}} h \vee (h \wedge k),$$

are both classified with h*, and both read back to* e*.*

Proof. In the singleton lattice, meet and join ignore their inputs and return e. The carrier witness for h gives $h\mathsf{hsamee}$, while reflexivity gives $\mathsf{ehsamee}$.

The first order comparison uses symmetry of $h\mathsf{hsamee}$ to obtain $\mathsf{ehsame}h$ and then applies the prefix comparison constructor. The second uses $h\mathsf{hsamee}$ directly. The same two hsame witnesses assemble the classifier tuples with h, and the endpoint readbacks are both reflexivity at e. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption order classifier`

Theorem 55.22 (Singleton meet greatest lower bound)**.** *For carried histories* h, k, z *in the singleton lattice, the concrete meet output is the empty history, and the lower-bound condition against that meet is exactly the pair of lower-bound conditions against the two operands:*

$$\mathsf{hsame}(\mathsf{LatticeSingletonMeet}(h, k), \mathsf{e}) \wedge (z \leq_{\mathsf{e}} \mathsf{LatticeSingletonMeet}(h, k) \iff z \leq_{\mathsf{e}} h \wedge z \leq_{\mathsf{e}} k)\,.$$

Proof. The meet operation is definitionally the empty history. Since h, k, z are all carried by the singleton carrier, the order endpoint theorem Definition 55.63 reduces every displayed comparison to the two empty-endpoint witnesses. The forward direction rebuilds the two operand bounds from the endpoint witnesses for z, h, k; the reverse direction rebuilds the meet bound from the endpoint witness for z and reflexivity at e. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet empty output greatest lower bound iff`

Theorem 55.23 (Singleton join least upper bound)**.** *For carried histories h, k, z in the singleton lattice, the concrete join least-upper-bound comparison is exactly the pair of upper-bound comparisons against the two operands, together with the empty readback of the join:*

$$\mathsf{LatticeSingletonJoin}(h, k) \leq_{\mathsf{e}} z \iff h \leq_{\mathsf{e}} z \wedge k \leq_{\mathsf{e}} z \wedge \mathsf{LatticeSingletonJoin}(h, k)\mathsf{hsamee}.$$

Proof. The join operation is definitionally the empty history. Since h, k, z are all carried by the singleton carrier, the singleton order comparison between any two of them is built from the corresponding hsame chain through e. The forward direction reconstructs the two operand upper bounds and records reflexivity of the join endpoint; the reverse direction uses the displayed join endpoint together with the carrier of z to rebuild the join bound. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoin least upper bound empty iff`

Theorem 55.24 (Singleton lattice append tail emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$ and $\mathsf{UnaryHistory}(t)$, then the restricted singleton prefix order satisfies*

$$\mathsf{append}(h, t) \leq_{\mathsf{e}} h \quad \iff \quad t\mathsf{hsamee}.$$

Proof. The forward direction projects the underlying prefix comparison from the restricted order and applies the concrete append-tail backforcing law from $\mathsf{Preorder}^{\uparrow}$. A unary tail that returns from $\mathsf{append}(h, t)$ back to h must be the empty history.

For the reverse direction, the witness $t\mathsf{hsamee}$ reduces $\mathsf{append}(h, t)$ to h by the right continuation unit law. The carried witness $A_{\mathsf{e}}(h)$ therefore also carries $\mathsf{append}(h, t)$, and the prefix comparison is the empty-tail side of the same append-tail equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append tail empty iff`

Theorem 55.25 (Singleton lattice classifier append tail emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$, then the singleton classifier satisfies*

$$\mathsf{LatticeSingletonClassifier}(\mathsf{append}(h, t), h) \quad \iff \quad t\mathsf{hsamee}.$$

Proof. The forward direction projects the carried left endpoint from the classifier. Since $\mathsf{append}(h, t)\mathsf{hsamee}$, the concrete append-empty inversion forces $t\mathsf{hsamee}$.

Conversely, assume $t\mathsf{hsamee}$. The append endpoint is carried because both h and t are empty in the singleton sense, and the right continuation unit identifies $\mathsf{append}(h, t)$ with h. Together with the given carrier for h, these witnesses reconstruct the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier append tail empty iff`

Theorem 55.26 (Singleton lattice append context empty order)**.** *For $left, h, right : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$, then the restricted singleton prefix order between the two append displays satisfies*

$$\mathsf{append}(left, h) \leq_{\mathsf{e}} \mathsf{append}(h, right) \quad \iff \quad left\mathsf{hsamee} \wedge right\mathsf{hsamee}.$$

Proof. In the forward direction, the restricted order exposes singleton carriers for both append endpoints. BEDC append-empty inversion on $\mathsf{append}(left, h)$ gives $left\mathsf{hsamee}$, and the same inversion on $\mathsf{append}(h, right)$ gives $right\mathsf{hsamee}$.

Conversely, if the two contexts are empty, the carrier of h makes both append endpoints singleton-carried. Since both endpoints are history-same to e, they are history-same to each other, and the singleton order is rebuilt from the corresponding concrete prefix witness. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append context empty iff`

Theorem 55.27 (Singleton lattice append tail upper emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$ and $\mathsf{UnaryHistory}(t)$, then the restricted singleton prefix order satisfies*

$$h \leq_{\mathsf{e}} \mathsf{append}(h, t) \quad \Longleftrightarrow \quad t\mathsf{hsamee}.$$

Proof. The forward direction reads the upper endpoint carrier from the restricted order. Thus $\mathsf{append}(h, t)$ is history-same to e, and the BEDC append-empty inversion forces both endpoints of the append, in particular t, to be empty. Conversely, if $t\mathsf{hsamee}$, the right continuation unit identifies $\mathsf{append}(h, t)$ with h, so the given singleton carrier for h also carries the append endpoint. The prefix witness is the concrete append-tail continuation. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append tail upper empty iff`

Theorem 55.28 (Singleton lattice meet greatest lower bound empty exactness)**.** *For carried singleton endpoints h, k, z, the statement that z is below the concrete meet of h and k is equivalent to the two endpoint lower-bound clauses together with the fact that the concrete meet is empty:*

$$z \leq_{\mathsf{e}} \mathsf{meet}(h, k) \quad \Longleftrightarrow \quad z \leq_{\mathsf{e}} h \wedge z \leq_{\mathsf{e}} k \wedge \mathsf{meet}(h, k)\mathsf{hsamee}.$$

Proof. The singleton carrier identifies h, k, z with e, and the concrete meet operation is definitionally e. Thus the forward direction builds both lower-bound comparisons from the endpoint sameness witnesses using the singleton prefix comparison. Conversely, the displayed empty-meet endpoint and the carrier witness for z build the singleton comparison $z \leq_{\mathsf{e}} \mathsf{meet}(h, k)$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet greatest lower bound empty iff`

The $\mathsf{Lattice}^{\uparrow}$ schema is instantiated by a concrete BHist carrier and operations, so Lean can target $\mathsf{BEDC.Derived.LatticeUp}$ as a BHist-anchored certificate rather than an abstract operator schema.

55.3 The certificate

Definition 55.29 (Lattice source specification)**.** *The source is a certified poset carrier plus binary meet and join data.*

Definition 55.30 (Lattice pattern specification)**.** *The pattern is expression formation by $\wedge$ and $\vee$ over carrier witnesses.*

Definition 55.31 (Lattice classifier specification)**.** *The classifier is the poset carrier classifier, with meet and join congruent in both arguments.*

Definition 55.32 (Lattice stability certificate)**.** *The stability certificate consists of:*

1. *all poset stability fields*

2. *meet lower-bound laws*
3. *meet greatestness law*
4. *join upper-bound laws*
5. *join leastness law*
6. *meet and join congruence*

Definition 55.33 (Lattice ledger policy)**.** *Record operands and the bound-characterization certificate for each meet or join.*

Definition 55.34 ($\mathsf{Lattice}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Lattice}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Lattice}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Lemma 55.35 (Meet idempotence from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For every $x \in A$,*

$$x \wedge x \sim_C x.$$

Proof. The meet lower-left field gives

$$x \wedge x \leq x.$$

The inherited preorder reflexivity field gives $x \leq x$. Applying meet greatestness with $z = x$, $a = x$, and $b = x$, using the two reflexivity witnesses $x \leq x$ and $x \leq x$, gives

$$x \leq x \wedge x.$$

The inherited poset antisymmetry field applied to

$$x \wedge x \leq x \qquad \text{and} \qquad x \leq x \wedge x$$

yields

$$x \wedge x \sim_C x.$$

□

Lemma 55.36 (Join idempotence from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For every $x \in A$,*

$$x \vee x \sim_C x.$$

Proof. The join upper-left field gives

$$x \leq x \vee x.$$

The inherited preorder reflexivity field gives $x \leq x$. Applying join leastness with $z = x$, $a = x$, and $b = x$, using the two reflexivity witnesses $x \leq x$ and $x \leq x$, gives

$$x \vee x \leq x.$$

The inherited poset antisymmetry field applied to

$$x \vee x \leq x \qquad \text{and} \qquad x \leq x \vee x$$

yields

$$x \vee x \sim_C x.$$

□

Lemma 55.37 (Meet absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For all $x, y \in A$,*

$$x \wedge (x \vee y) \sim_C x.$$

Proof. The meet lower-left field gives

$$x \wedge (x \vee y) \leq x.$$

For the reverse comparison, the inherited preorder reflexivity field gives

$$x \leq x.$$

The join upper-left field gives

$$x \leq x \vee y.$$

Applying meet greatestness with $z = x$, $a = x$, and $b = x \vee y$, using the two comparisons above, gives

$$x \leq x \wedge (x \vee y).$$

The inherited poset antisymmetry field applied to

$$x \wedge (x \vee y) \leq x \qquad \text{and} \qquad x \leq x \wedge (x \vee y)$$

yields

$$x \wedge (x \vee y) \sim_C x.$$

□

Lemma 55.38 (Join absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For all $x, y \in A$,*

$$x \vee (x \wedge y) \sim_C x.$$

Proof. The join upper-left field gives

$$x \leq x \vee (x \wedge y).$$

For the reverse comparison, the inherited preorder reflexivity field gives

$$x \leq x.$$

The meet lower-left field gives

$$x \wedge y \leq x.$$

Applying join leastness with $z = x$, $a = x$, and $b = x \wedge y$, using the two comparisons above, gives

$$x \vee (x \wedge y) \leq x.$$

The inherited poset antisymmetry field applied to

$$x \vee (x \wedge y) \leq x \qquad \text{and} \qquad x \leq x \vee (x \wedge y)$$

yields

$$x \vee (x \wedge y) \sim_C x.$$

□

Theorem 55.39 (Lattice idempotence and absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *reflexivity and antisymmetry fields, and assume the directional meet and join bound-characterization fields:*

$$\forall a\, b,\ a \wedge b \leq a, \qquad \forall a\, b,\ a \wedge b \leq b,$$
$$\forall z\, a\, b,\ z \leq a \to z \leq b \to z \leq a \wedge b,$$
$$\forall a\, b,\ a \leq a \vee b, \qquad \forall a\, b,\ b \leq a \vee b,$$
$$\forall z\, a\, b,\ a \leq z \to b \leq z \to a \vee b \leq z.$$

Then for all $x, y \in A$,

$$x \wedge x \sim_C x, \qquad x \vee x \sim_C x, \qquad x \wedge (x \vee y) \sim_C x, \qquad x \vee (x \wedge y) \sim_C x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected idempotence absorption from bounds`

Proof. The first assertion is Lemma 55.105. The second assertion is Lemma 55.106. The third assertion is Lemma 55.107. The fourth assertion is Lemma 55.108. □

Theorem 55.40 (Lattice commutativity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y \in A$,*

$$x \wedge y \sim_C y \wedge x, \qquad x \vee y \sim_C y \vee x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected commutativity from bounds`

Proof. For the meet comparison, the meet lower-bound laws in Definition 55.102 give

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y.$$

Applying meet greatestness with $z = x \wedge y$, $a = y$, and $b = x$ gives

$$x \wedge y \leq y \wedge x.$$

The same lower-bound laws applied to $y \wedge x$ give

$$y \wedge x \leq y \qquad \text{and} \qquad y \wedge x \leq x.$$

Applying meet greatestness with $z = y \wedge x$, $a = x$, and $b = y$ gives

$$y \wedge x \leq x \wedge y.$$

Since $\mathsf{Lattice}^{\uparrow}$ extends the $\mathsf{POSet}^{\uparrow}$ layer, the antisymmetry field recorded in Definition 53.43 applies to these two comparisons and yields

$$x \wedge y \sim_C y \wedge x.$$

For the join comparison, the join upper-bound laws in Definition 55.102 give

$$y \leq x \vee y \qquad \text{and} \qquad x \leq x \vee y.$$

Applying join leastness with $z = x \vee y$, $a = y$, and $b = x$ gives

$$y \vee x \leq x \vee y.$$

The same upper-bound laws applied to $y \vee x$ give

$$x \leq y \vee x \qquad \text{and} \qquad y \leq y \vee x.$$

Applying join leastness with $z = y \vee x$, $a = x$, and $b = y$ gives

$$x \vee y \leq y \vee x.$$

The inherited antisymmetry field from Definition 53.43 applied to these two comparisons yields

$$x \vee y \sim_C y \vee x.$$

□

Theorem 55.41 (Lattice opposite absorption from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited preorder reflexivity field, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field, and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y \in A$,*

$$x \wedge (y \vee x) \sim_C x, \qquad (x \vee y) \wedge x \sim_C x, \qquad x \vee (y \wedge x) \sim_C x, \qquad (x \wedge y) \vee x \sim_C x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeOppositeAbsorptionFromDirectionalBounds`

Proof. The inherited reflexivity field is the preorder reflexivity component of Definition 52.44; the inherited classifier collapse is the antisymmetry component of Definition 53.43. The meet and join directional fields are those listed in Definition 55.102.

For $x \wedge (y \vee x)$, the meet lower-left field gives

$$x \wedge (y \vee x) \leq x.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

The join upper-right field applied to $y \vee x$ gives

$$x \leq y \vee x.$$

Applying meet greatestness with $z = x$, $a = x$, and $b = y \vee x$, using the two comparisons above, gives

$$x \leq x \wedge (y \vee x).$$

Antisymmetry from Definition 53.43 applied to

$$x \wedge (y \vee x) \leq x \qquad \text{and} \qquad x \leq x \wedge (y \vee x)$$

yields

$$x \wedge (y \vee x) \sim_C x.$$

For $(x \vee y) \wedge x$, the meet lower-right field gives

$$(x \vee y) \wedge x \leq x.$$

The join upper-left field gives

$$x \leq x \vee y.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

Applying meet greatestness with $z = x$, $a = x \vee y$, and $b = x$, using these two comparisons, gives

$$x \leq (x \vee y) \wedge x.$$

Antisymmetry applied to

$$(x \vee y) \wedge x \leq x \qquad \text{and} \qquad x \leq (x \vee y) \wedge x$$

yields

$$(x \vee y) \wedge x \sim_C x.$$

For $x \vee (y \wedge x)$, the join upper-left field gives

$$x \leq x \vee (y \wedge x).$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

The meet lower-right field applied to $y \wedge x$ gives

$$y \wedge x \leq x.$$

Applying join leastness with $z = x$, $a = x$, and $b = y \wedge x$, using the two comparisons above, gives

$$x \vee (y \wedge x) \leq x.$$

Antisymmetry applied to

$$x \vee (y \wedge x) \leq x \qquad \text{and} \qquad x \leq x \vee (y \wedge x)$$

yields

$$x \vee (y \wedge x) \sim_C x.$$

For $(x \wedge y) \vee x$, the join upper-right field gives

$$x \leq (x \wedge y) \vee x.$$

The meet lower-left field gives

$$x \wedge y \leq x.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

Applying join leastness with $z = x$, $a = x \wedge y$, and $b = x$, using these two comparisons, gives

$$(x \wedge y) \vee x \leq x.$$

Antisymmetry applied to

$$(x \wedge y) \vee x \leq x \qquad \text{and} \qquad x \leq (x \wedge y) \vee x$$

yields

$$(x \wedge y) \vee x \sim_C x.$$

□

Theorem 55.42 (Lattice bound uniqueness from directional certificates)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43 and the directional meet and join bound-characterization fields from Definition 55.102. For all $x, y, m, j \in A$, suppose m satisfies the lower-bound and greatestness clauses*

$$m \leq x, \qquad m \leq y,$$
$$\forall z \in A,\ z \leq x \to z \leq y \to z \leq m,$$

and suppose j satisfies the upper-bound and leastness clauses

$$x \leq j, \qquad y \leq j,$$
$$\forall z \in A,\ x \leq z \to y \leq z \to j \leq z.$$

Then

$$m \sim_C x \wedge y, \qquad j \sim_C x \vee y.$$

Proof. The certified meet fields give

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y.$$

Applying the assumed greatestness clause for m with $z = x \wedge y$ gives

$$x \wedge y \leq m.$$

The two assumed lower-bound clauses for m, together with the certified meet greatestness field applied with $z = m$, give

$$m \leq x \wedge y.$$

The antisymmetry field from Definition 53.43 applied to

$$m \leq x \wedge y \qquad \text{and} \qquad x \wedge y \leq m$$

yields

$$m \sim_C x \wedge y.$$

The certified join fields give

$$x \leq x \vee y \qquad \text{and} \qquad y \leq x \vee y.$$

Applying the assumed leastness clause for j with $z = x \vee y$ gives

$$j \leq x \vee y.$$

The two assumed upper-bound clauses for j, together with the certified join leastness field applied with $z = j$, give

$$x \vee y \leq j.$$

The antisymmetry field from Definition 53.43 applied to

$$j \leq x \vee y \qquad \text{and} \qquad x \vee y \leq j$$

yields

$$j \sim_C x \vee y.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingleton bound uniqueness from directional certificates`

Definition 55.43 (Concrete unary total-order lattice selectors)**.** *For carried unary histories h, k in $\mathsf{Carrier_u}$, let the active comparison branch be the ledger branch supplied by Definition 54.9. Define the trichotomy lower and upper endpoint selectors by*

$$\mathsf{glb_{tri}}(h,k) := \begin{cases} h, & \text{on } \mathsf{NatUnaryStrictPrefix}(h,k), \\ h, & \text{on } \rho_{\mathsf{u}}(h,k), \\ k, & \text{on } \mathsf{NatUnaryStrictPrefix}(k,h), \end{cases} \qquad \mathsf{lub_{tri}}(h,k) := \begin{cases} k, & \text{on } \mathsf{NatUnaryStrictPrefix}(h,k), \\ k, & \text{on } \rho_{\mathsf{u}}(h,k), \\ h, & \text{on } \mathsf{NatUnaryStrictPrefix}(k,h). \end{cases}$$

The total-order-selected lattice operations are

$$h \wedge_{\mathsf{tri}} k := \mathsf{glb_{tri}}(h,k), \qquad h \vee_{\mathsf{tri}} k := \mathsf{lub_{tri}}(h,k).$$

Theorem 55.44 (Concrete unary $\mathsf{TotalOrder}^{\uparrow}$ forgets to $\mathsf{Lattice}^{\uparrow}$)**.** *The concrete $\mathsf{TotalOrder}^{\uparrow}$ naming certificate of Definition 54.20 determines a $\mathsf{Lattice}^{\uparrow}$ certificate over the same carrier $C = \mathsf{Carrier_u}$, classifier ρ_{u}, and order $\leq_{\mathsf{u}}$, with meet and join given by Definition 55.113. For every carried pair $h, k \in C$, the meet and join classifier-equal the trichotomy endpoint selectors:*

$$\rho_{\mathsf{u}}(h \wedge_{\mathsf{tri}} k, \mathsf{glb_{tri}}(h,k)) \qquad \text{and} \qquad \rho_{\mathsf{u}}(h \vee_{\mathsf{tri}} k, \mathsf{lub_{tri}}(h,k)).$$

The meet bound-characterization rows are the greatest-lower selector rows of Definition 54.12, and the join bound-characterization rows are the least-upper selector rows of Definition 54.10.

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeUnaryPrefix extrema certificate`

Proof. The source, carrier, classifier, relation, and inherited $\mathsf{POSet}^{\uparrow}$ stability rows are those already packaged in Definition 54.20. That theorem cites the concrete unary-prefix $\mathsf{POSet}^{\uparrow}$ naming certificate, so the partial-order reflexivity, transitivity, congruence, and antisymmetry fields required by the lattice interface are already present.

Fix carried unary histories h, k. The comparison partition of Definition 54.9 chooses exactly one of the three branches

$$\mathsf{NatUnaryStrictPrefix}(h,k), \qquad \rho_{\mathsf{u}}(h,k), \qquad \mathsf{NatUnaryStrictPrefix}(k,h).$$

The selector definition follows that same branch ledger: in the strict-left and same branches it chooses h as the lower endpoint and k as the upper endpoint, while in the strict-right branch it chooses k as the lower endpoint and h as the upper endpoint.

The theorem Definition 54.12 supplies, for this selected lower endpoint, a carried history l with

$$l \leq_{\mathsf{u}} h, \qquad l \leq_{\mathsf{u}} k, \qquad (r \leq_{\mathsf{u}} h \wedge r \leq_{\mathsf{u}} k) \Rightarrow r \leq_{\mathsf{u}} l$$

for every carried lower-bound witness r. By the displayed branch definition, this l is precisely $\mathsf{glb_{tri}}(h,k)$, hence precisely $h \wedge_{\mathsf{tri}} k$. These are the meet lower-bound and meet greatestness rows listed in Definition 55.102.

Similarly, Definition 54.10 supplies, for the selected upper endpoint, a carried history u with

$$h \leq_{\mathsf{u}} u, \qquad k \leq_{\mathsf{u}} u, \qquad (h \leq_{\mathsf{u}} r \wedge k \leq_{\mathsf{u}} r) \Rightarrow u \leq_{\mathsf{u}} r$$

for every carried upper-bound witness r. By the displayed branch definition, this u is precisely $\mathsf{lub_{tri}}(h,k)$, hence precisely $h \vee_{\mathsf{tri}} k$. These are the join upper-bound and join leastness rows listed in Definition 55.102.

It remains to account for meet and join congruence. Suppose h, h', k, k' are carried and that $\rho_{\mathsf{u}}(h, h')$ and $\rho_{\mathsf{u}}(k, k')$ hold. The branch-transport theorem Definition 54.13 preserves and reflects the strict-left, same, and strict-right branches across these endpoint replacements. If the active branch is strict-left or same, the meet outputs are h and h' and the join outputs are k and k'. If the active branch is strict-right, the meet outputs are k and k' and the join outputs are h and h'. In each case the assumed endpoint classifier witnesses give the required classifier comparison between the selected outputs. Thus meet and join are congruent in both arguments.

The classifier-equality rows in the theorem statement follow from the same selected-readback definition and classifier reflexivity on carried endpoints. The ledger row of Definition 55.103 records the selected comparison branch together with the greatest-lower and least-upper bound certificates just cited. Therefore the tuple with operations $\wedge_{\mathsf{tri}}$ and $\vee_{\mathsf{tri}}$ satisfies the $\mathsf{Lattice}^{\uparrow}$ certificate obligations of Definition 55.104. □

Definition 55.45 (Opposite lattice certificate data)**.** *Let a* $\mathsf{Lattice}^{\uparrow}$ *certificate be presented by a carrier predicate* $\mathcal{C}$*, classifier* $\sim_C$*, order* $\leq_C$*, meet* $\wedge_C$*, and join* $\vee_C$*. Define*

$$a \leq_C^{\mathrm{op}} b := b \leq_C a, \qquad a \wedge_C^{\mathrm{op}} b := a \vee_C b, \qquad a \vee_C^{\mathrm{op}} b := a \wedge_C b.$$

The opposite lattice data C^{op} *have the same carrier predicate* $\mathcal{C}$*, the same classifier* $\sim_C$*, the opposite order* $\leq_C^{\mathrm{op}}$*, meet* $\wedge_C^{\mathrm{op}}$*, and join* $\vee_C^{\mathrm{op}}$*.*

Theorem 55.46 (Opposite lattice certificate)**.** *If* $(\mathcal{C}, \sim_C, \leq_C, \wedge_C, \vee_C)$ *satisfies the* $\mathsf{Lattice}^{\uparrow}$ *certificate obligations of Definition 55.104, then the data of Definition 55.115 also satisfy the* $\mathsf{Lattice}^{\uparrow}$ *certificate obligations.*

Proof. Unpack the lattice obligations through Definition 55.104. By Definition 55.60 and Definition 55.99, the given lattice certificate contains a certified $\mathsf{POSet}^{\uparrow}$ carrier with carrier predicate $\mathcal{C}$, classifier $\sim_C$, and order $\leq_C$. Applying Definition 53.50 supplies the carrier, source, pattern, classifier, stability, and ledger rows for the same carrier and classifier with order $\leq_C^{\mathrm{op}}$.

The binary operation rows are present with names exchanged. The opposite meet $\wedge_C^{\mathrm{op}}$ is the original join $\vee_C$, and the opposite join $\vee_C^{\mathrm{op}}$ is the original meet $\wedge_C$. Thus the carried-operation data required by the lattice source row are copied from the original join and meet data.

It remains to verify the four bound-characterization rows in Definition 55.102. Fix carried endpoints a, b, z. For the opposite meet, the lower-bound laws are exactly the original join upper-bound laws:

$$a \wedge_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} a \iff a \leq_C a \vee_C b, \qquad a \wedge_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} b \iff b \leq_C a \vee_C b.$$

For opposite meet greatestness, assume $z \leq_C^{\mathrm{op}} a$ and $z \leq_C^{\mathrm{op}} b$. These unfold to $a \leq_C z$ and $b \leq_C z$. The original join leastness row gives $a \vee_C b \leq_C z$, which is exactly

$$z \leq_C^{\mathrm{op}} a \wedge_C^{\mathrm{op}} b.$$

For the opposite join, the upper-bound laws are exactly the original meet lower-bound laws:

$$a \leq_C^{\mathrm{op}} a \vee_C^{\mathrm{op}} b \iff a \wedge_C b \leq_C a, \qquad b \leq_C^{\mathrm{op}} a \vee_C^{\mathrm{op}} b \iff a \wedge_C b \leq_C b.$$

For opposite join leastness, assume $a \leq_C^{\mathrm{op}} z$ and $b \leq_C^{\mathrm{op}} z$. These unfold to $z \leq_C a$ and $z \leq_C b$. The original meet greatestness row gives $z \leq_C a \wedge_C b$, which is exactly

$$a \vee_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} z.$$

Meet and join congruence are also exchanged. If $a \sim_C a'$ and $b \sim_C b'$, then original join congruence gives

$$a \wedge_C^{\mathrm{op}} b = a \vee_C b \sim_C a' \vee_C b' = a' \wedge_C^{\mathrm{op}} b',$$

and original meet congruence gives

$$a \vee_C^{\mathrm{op}} b = a \wedge_C b \sim_C a' \wedge_C b' = a' \vee_C^{\mathrm{op}} b'.$$

These are precisely the opposite meet-congruence and opposite join-congruence rows of Definition 55.101.

The pattern row of Definition 55.100 is unchanged expression formation by two binary lattice operations; each opposite meet node is read as an original join node, and each opposite join node is read as an original meet node. The ledger row of Definition 55.103 is the same exchange: an opposite meet ledger records the original join bound-characterization certificate, and an opposite join ledger records the original meet bound-characterization certificate, with the order witnesses interpreted through $\leq_C^{\mathrm{op}}$. Together with the opposite $\mathsf{POSet}^{\uparrow}$ rows supplied above, these are all carrier, source, pattern, classifier, stability, and ledger obligations required by Definition 55.104. Hence C^{op} carries a $\mathsf{Lattice}^{\uparrow}$ certificate. □

Theorem 55.47 (Double-opposite lattice certificate data identity)**.** *Let E be a $\mathsf{Lattice}^{\uparrow}$ certificate satisfying the obligations of Definition 55.104. Present its lattice data by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, order relation $\leq_C$, meet $\wedge_C$, join $\vee_C$, meet ledger row $\mathcal{L}_C^{\wedge}$, and join ledger row $\mathcal{L}_C^{\vee}$ as in Definition 55.103. Let $E^{\mathrm{opop}} = (E^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 55.115 twice. Then E^{opop} is a $\mathsf{Lattice}^{\uparrow}$ certificate, and its carrier predicate, classifier, order relation, meet, join, and lattice ledger rows are componentwise identical to those of E: for carried endpoints a, b,*

$$\begin{aligned} \mathcal{C}^{\mathrm{opop}}(a) &= \mathcal{C}(a), & a \sim_C^{\mathrm{opop}} b &= a \sim_C b, \\ a \leq_C^{\mathrm{opop}} b &= a \leq_C b, & a \wedge_C^{\mathrm{opop}} b &= a \wedge_C b, \\ a \vee_C^{\mathrm{opop}} b &= a \vee_C b, & \mathcal{L}_C^{\wedge,\mathrm{opop}}(a,b) &= \mathcal{L}_C^{\wedge}(a,b), \\ \mathcal{L}_C^{\vee,\mathrm{opop}}(a,b) &= \mathcal{L}_C^{\vee}(a,b). \end{aligned}$$

Proof. Because E satisfies the $\mathsf{Lattice}^{\uparrow}$ obligations, Definition 55.116 makes E^{op} a certified $\mathsf{Lattice}^{\uparrow}$ instance. Applying the same theorem to that certified opposite data makes E^{opop} a certified $\mathsf{Lattice}^{\uparrow}$ instance.

By Definition 55.60 and Definition 55.99, the inherited order part of E is a certified $\mathsf{POSet}^{\uparrow}$ carrier with carrier predicate $\mathcal{C}$, classifier $\sim_C$, and order $\leq_C$. Applying Definition 53.51 to this inherited part gives the carrier, classifier, order, and inherited antisymmetry-ledger identities after two opposites. Hence

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}(a), \qquad a \sim_C^{\mathrm{opop}} b = a \sim_C b, \qquad a \leq_C^{\mathrm{opop}} b = a \leq_C b.$$

For the lattice operations, two unfoldings of Definition 55.115 give

$$a \wedge_C^{\mathrm{opop}} b = a \vee_C^{\mathrm{op}} b = a \wedge_C b$$

and

$$a \vee_C^{\mathrm{opop}} b = a \wedge_C^{\mathrm{op}} b = a \vee_C b.$$

Thus the double-opposite meet and join data are exactly the original meet and join data.

The source row is a certified poset carrier plus meet and join operations, so the restored poset part together with the restored operation slots gives the original source row. The pattern row of Definition 55.100 is expression formation by meet and join over carrier witnesses; one opposite exchanges the two operation names and the second exchange restores them.

The stability rows of Definition 55.102 are restored in the same componentwise way. In one opposite, the meet lower-bound and greatestness rows are the original join upper-bound and leastness rows read through the reversed order, while the join rows are the original meet rows read through the reversed order. The second opposite reverses the order again and exchanges the operation slots again, so the meet lower-bound, meet greatestness, join upper-bound, and join leastness rows are the original rows. Meet congruence and join congruence from Definition 55.101 are also exchanged twice and hence restored.

Finally, Definition 55.103 records the operands and the bound-characterization certificate for each meet or join. A single opposite reads a meet ledger as the corresponding original join ledger and a join ledger as the corresponding original meet ledger, with order witnesses interpreted through $\leq_C^{\mathrm{op}}$. The second opposite performs the same exchange again and the order has already returned to $\leq_C$. Therefore

$$\mathcal{L}_C^{\wedge,\mathrm{opop}}(a,b) = \mathcal{L}_C^{\wedge}(a,b), \qquad \mathcal{L}_C^{\vee,\mathrm{opop}}(a,b) = \mathcal{L}_C^{\vee}(a,b).$$

Together these identities identify the double-opposite lattice certificate data with the original lattice certificate data. □

Theorem 55.48 (Lattice one-sided distributive inequality from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited preorder transitivity field and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y, z \in A$,*

$$(x \wedge y) \vee (x \wedge z) \leq x \wedge (y \vee z).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected one sided distributive inequality from bounds`

Proof. Let

$$q := x \wedge (y \vee z).$$

The join leastness field will give the desired comparison once both join inputs are shown to be below q.

For the first input, the meet lower-left field gives

$$x \wedge y \leq x.$$

The meet lower-right field gives

$$x \wedge y \leq y,$$

and the join upper-left field gives

$$y \leq y \vee z.$$

By inherited preorder transitivity from Definition 52.44, these two comparisons compose to

$$x \wedge y \leq y \vee z.$$

Applying meet greatestness to the two comparisons

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y \vee z$$

gives

$$x \wedge y \leq x \wedge (y \vee z),$$

that is,

$$x \wedge y \leq q.$$

For the second input, the meet lower-left field gives

$$x \wedge z \leq x.$$

The meet lower-right field gives

$$x \wedge z \leq z,$$

and the join upper-right field gives

$$z \leq y \vee z.$$

By inherited preorder transitivity, these two comparisons compose to

$$x \wedge z \leq y \vee z.$$

Applying meet greatestness to the two comparisons

$$x \wedge z \leq x \qquad \text{and} \qquad x \wedge z \leq y \vee z$$

gives

$$x \wedge z \leq x \wedge (y \vee z),$$

that is,

$$x \wedge z \leq q.$$

The join leastness field applied with upper bound q to

$$x \wedge y \leq q \qquad \text{and} \qquad x \wedge z \leq q$$

yields

$$(x \wedge y) \vee (x \wedge z) \leq q.$$

Substituting the definition of q gives

$$(x \wedge y) \vee (x \wedge z) \leq x \wedge (y \vee z).$$

□

Lemma 55.49 (Directed meet associativity comparison from bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound implication fields from Definition 55.102. Then for all $x, y, z \in A$,*

$$(x \wedge y) \wedge z \leq x \wedge (y \wedge z).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet assoc comparison empty`

Proof. Put

$$w = (x \wedge y) \wedge z.$$

The meet lower-left field gives

$$w \leq x \wedge y,$$

and the meet lower-left field applied to $x \wedge y$ gives

$$x \wedge y \leq x.$$

By the inherited preorder transitivity field,

$$w \leq x.$$

The meet lower-right field applied to $x \wedge y$ gives

$$x \wedge y \leq y,$$

so transitivity with $w \leq x \wedge y$ gives

$$w \leq y.$$

The meet lower-right field applied to $w = (x \wedge y) \wedge z$ gives

$$w \leq z.$$

Applying meet greatestness to the two bounds $w \leq y$ and $w \leq z$ gives

$$w \leq y \wedge z.$$

A second application of meet greatestness, now to $w \leq x$ and $w \leq y \wedge z$, gives

$$w \leq x \wedge (y \wedge z).$$

Substituting the displayed definition of w is the required comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirectedMeet associativity comparison from bounds`

Lemma 55.50 (Directed join associativity comparison from bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional join upper-bound and least-upper-bound implication fields from Definition 55.102. Then for all $x, y, z \in A$, $(x \vee y) \vee z \leq_C x \vee (y \vee z)$.*

Proof. Put $w = x \vee (y \vee z)$. The join upper-left field gives $x \leq_C w$. The join upper-left field applied to $y \vee z$ gives $y \leq_C y \vee z$, and the join upper-right field applied to $w = x \vee (y \vee z)$ gives $y \vee z \leq_C w$. By the inherited preorder transitivity field, $y \leq_C w$. Applying join leastness to the two bounds $x \leq_C w$ and $y \leq_C w$ gives $x \vee y \leq_C w$. The join upper-right field applied to $y \vee z$ gives $z \leq_C y \vee z$, and transitivity with $y \vee z \leq_C w$ gives $z \leq_C w$. A second application of join leastness, now to $x \vee y \leq_C w$ and $z \leq_C w$, gives $(x \vee y) \vee z \leq_C w$. Substituting the definition of w gives $(x \vee y) \vee z \leq_C x \vee (y \vee z)$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoin directed assoc comparison from bounds`

Lemma 55.51 (Meet absorbs a smaller ordered endpoint)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity field from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $x, z \in A$ and $x \leq_C z$, then*

$$z \wedge x \sim_C x.$$

Proof. The meet lower-right field gives

$$z \wedge x \leq_C x.$$

The inherited preorder reflexivity field gives

$$x \leq_C x.$$

Together with the assumed comparison $x \leq_C z$, meet greatestness applied with witness endpoint x and operands z and x gives

$$x \leq_C z \wedge x.$$

The antisymmetry field from Definition 53.43 applied to these two comparisons yields

$$z \wedge x \sim_C x.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeMeet absorbs smaller ordered endpoint from bounds`

Lemma 55.52 (Modular lower comparison from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound, meet greatestness, join upper-bound, and join leastness fields from Definition 55.102. If $x, y, z \in A$ and $x \leq_C z$, then*

$$x \vee (y \wedge z) \leq_C (x \vee y) \wedge z.$$

Proof. Put

$$n = x \vee (y \wedge z), \qquad m = (x \vee y) \wedge z.$$

The join upper-left field gives

$$x \leq_C x \vee y.$$

The meet lower-left field gives

$$y \wedge z \leq_C y,$$

and the join upper-right field gives

$$y \leq_C x \vee y.$$

By transitivity,

$$y \wedge z \leq_C x \vee y.$$

Join leastness with upper target $x \vee y$ gives

$$n \leq_C x \vee y.$$

The assumption gives $x \leq_C z$, and the meet lower-right field gives

$$y \wedge z \leq_C z.$$

Join leastness with upper target z gives

$$n \leq_C z.$$

Meet greatestness applied to the two bounds $n \leq_C x \vee y$ and $n \leq_C z$ gives

$$n \leq_C m.$$

Substituting the displayed definitions of m and n gives the claimed comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeModular lower comparison from directional bounds`

Theorem 55.53 (Distributivity gives the modular law)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the classifier symmetry and transitivity fields from Definition 9.7, the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet and join bound-characterization fields from Definition 55.102. Assume also the join classifier-congruence field from Definition 55.101, and suppose the distributive law*

$$a \wedge (b \vee c) \sim_C (a \wedge b) \vee (a \wedge c)$$

holds for all $a, b, c \in A$. If $x, y, z \in A$ and $x \leq_C z$, then

$$x \vee (y \wedge z) \leq_C (x \vee y) \wedge z \qquad \text{and} \qquad x \vee (y \wedge z) \sim_C (x \vee y) \wedge z.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingleton distributivity modular law`

Proof. The first conclusion is Definition 55.91.

Put

$$m = (x \vee y) \wedge z, \qquad n = x \vee (y \wedge z).$$

By Definition 55.110, applied to the meet operands $x \vee y$ and z,

$$m \sim_C z \wedge (x \vee y).$$

The assumed distributive law with $a = z$, $b = x$, and $c = y$ gives

$$z \wedge (x \vee y) \sim_C (z \wedge x) \vee (z \wedge y).$$

By Definition 55.90 and the assumption $x \leq_C z$,

$$z \wedge x \sim_C x.$$

By Definition 55.110, applied to the meet operands z and y,

$$z \wedge y \sim_C y \wedge z.$$

Join classifier congruence applied to the last two classifier comparisons gives

$$(z \wedge x) \vee (z \wedge y) \sim_C x \vee (y \wedge z).$$

Classifier transitivity composes the three displayed classifier comparisons into

$$m \sim_C n.$$

Classifier symmetry reverses this comparison, giving

$$n \sim_C m.$$

Substituting the displayed definitions of m and n gives

$$x \vee (y \wedge z) \sim_C (x \vee y) \wedge z.$$

□

Lemma 55.54 (Join-meet lower comparison from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound, meet greatestness, join upper-bound, and join leastness fields from Definition 55.102. Then for all $a, b, c \in A$,*

$$a \vee (b \wedge c) \leq_C (a \vee b) \wedge (a \vee c).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoinMeet lower comparison empty`

Proof. Put

$$n = a \vee (b \wedge c), \qquad m = (a \vee b) \wedge (a \vee c).$$

The join upper-left field gives

$$a \leq_C a \vee b.$$

The meet lower-left field gives

$$b \wedge c \leq_C b,$$

and the join upper-right field gives

$$b \leq_C a \vee b.$$

By transitivity,

$$b \wedge c \leq_C a \vee b.$$

Join leastness with upper target $a \vee b$ gives

$$n \leq_C a \vee b.$$

Similarly, the join upper-left field gives

$$a \leq_C a \vee c.$$

The meet lower-right field gives

$$b \wedge c \leq_C c,$$

and the join upper-right field gives

$$c \leq_C a \vee c.$$

By transitivity,

$$b \wedge c \leq_C a \vee c.$$

Join leastness with upper target $a \vee c$ gives

$$n \leq_C a \vee c.$$

Meet greatestness applied to the two bounds $n \leq_C a \vee b$ and $n \leq_C a \vee c$ gives

$$n \leq_C m.$$

Substituting the displayed definitions of m and n gives the claimed comparison. □

Theorem 55.55 (Meet distributivity gives join distributivity)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the classifier symmetry and transitivity fields from Definition 9.7, the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet and join bound-characterization fields from Definition 55.102. Assume also the meet and join classifier-congruence rows from Definition 55.101, and suppose the distributive law*

$$u \wedge (v \vee w) \sim_C (u \wedge v) \vee (u \wedge w)$$

holds for all $u, v, w \in A$. Then for all $a, b, c \in A$,

$$a \vee (b \wedge c) \leq_C (a \vee b) \wedge (a \vee c) \qquad \textit{and} \qquad a \vee (b \wedge c) \sim_C (a \vee b) \wedge (a \vee c).$$

Proof. The first conclusion is Definition 55.93.

Put

$$d = (a \vee b) \wedge (a \vee c), \qquad n = a \vee (b \wedge c).$$

The assumed distributive law with $u = a \vee b$, $v = a$, and $w = c$ gives

$$d \sim_C ((a \vee b) \wedge a) \vee ((a \vee b) \wedge c).$$

By Definition 55.111, applied with $x = a$ and $y = b$,

$$(a \vee b) \wedge a \sim_C a.$$

By Definition 55.110, applied to the meet operands $a \vee b$ and c,

$$(a \vee b) \wedge c \sim_C c \wedge (a \vee b).$$

The assumed distributive law with $u = c$, $v = a$, and $w = b$ gives

$$c \wedge (a \vee b) \sim_C (c \wedge a) \vee (c \wedge b).$$

Classifier transitivity composes the last two comparisons into

$$(a \vee b) \wedge c \sim_C (c \wedge a) \vee (c \wedge b).$$

Join classifier congruence gives

$$((a \vee b) \wedge a) \vee ((a \vee b) \wedge c) \sim_C a \vee ((c \wedge a) \vee (c \wedge b)).$$

Put

$$p = a \vee ((c \wedge a) \vee (c \wedge b)), \qquad q = a \vee (c \wedge b).$$

To compare p and q, first note that the join upper-left field gives $a \leq_C q$. The meet lower-right field gives $c \wedge a \leq_C a$, so transitivity gives $c \wedge a \leq_C q$. The join upper-right field gives $c \wedge b \leq_C q$. Join leastness gives

$$(c \wedge a) \vee (c \wedge b) \leq_C q,$$

and another application of join leastness gives

$$p \leq_C q.$$

Conversely, the join upper-left field gives $a \leq_C p$. The join upper-right field gives

$$(c \wedge a) \vee (c \wedge b) \leq_C p,$$

and the join upper-right field applied to the inner join gives

$$c \wedge b \leq_C (c \wedge a) \vee (c \wedge b).$$

By transitivity, $c \wedge b \leq_C p$. Join leastness gives

$$q \leq_C p.$$

The antisymmetry field from Definition 53.43 applied to $p \leq_C q$ and $q \leq_C p$ yields

$$p \sim_C q.$$

By Definition 55.108, applied with $x = a$ and $y = a$,

$$a \vee (a \wedge a) \sim_C a.$$

Classifier symmetry and transitivity applied to this comparison give

$$a \sim_C a.$$

By Definition 55.110, applied to the meet operands c and b,

$$c \wedge b \sim_C b \wedge c.$$

Join classifier congruence gives

$$q \sim_C n.$$

Classifier transitivity composes the displayed classifier comparisons into

$$d \sim_C n.$$

Classifier symmetry reverses this comparison, giving

$$n \sim_C d.$$

Substituting the displayed definitions of d and n gives

$$a \vee (b \wedge c) \sim_C (a \vee b) \wedge (a \vee c).$$

□

Lemma 55.56 (Lattice meet two-sided monotonicity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $a, a', b, b' \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \wedge b \leq_C a' \wedge b'.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet monotone empty`

Proof. Put

$$w = a \wedge b.$$

The meet lower-left field gives

$$w \leq_C a.$$

Composing this comparison with the assumed comparison $a \leq_C a'$ by the inherited preorder transitivity field gives

$$w \leq_C a'.$$

The meet lower-right field gives

$$w \leq_C b.$$

Composing this comparison with the assumed comparison $b \leq_C b'$ gives

$$w \leq_C b'.$$

Applying meet greatestness to the two bounds $w \leq_C a'$ and $w \leq_C b'$ gives

$$w \leq_C a' \wedge b'.$$

Substituting the displayed definition of w gives the claimed comparison. □

Lemma 55.57 (Join is monotone in both ordered endpoints)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional join upper-bound and least-upper-bound fields from Definition 55.102. If $a, b, a', b' \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \vee b \leq_C a' \vee b'.$$

Proof. Put

$$u = a' \vee b'.$$

The join upper-left field gives

$$a' \leq_C u.$$

Composing the assumed comparison $a \leq_C a'$ with $a' \leq_C u$ by the inherited preorder transitivity field gives

$$a \leq_C u.$$

The join upper-right field gives

$$b' \leq_C u.$$

Composing the assumed comparison $b \leq_C b'$ with $b' \leq_C u$ by the same transitivity field gives

$$b \leq_C u.$$

Applying join leastness to the two upper-bound comparisons $a \leq_C u$ and $b \leq_C u$ gives

$$a \vee b \leq_C u.$$

Substituting the displayed definition of u gives the claimed comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeJoin two sided monotonicity from directional bounds`

Lemma 55.58 (Lattice meet monotonicity in the second argument from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $b, b', c \in A$ and $b \leq_C b'$, then*

$$c \wedge b \leq_C c \wedge b'.$$

Proof. Put

$$w = c \wedge b.$$

The meet lower-left field gives

$$w \leq_C c.$$

The meet lower-right field gives

$$w \leq_C b.$$

Composing this comparison with the assumed comparison $b \leq_C b'$ by the inherited preorder transitivity field gives

$$w \leq_C b'.$$

Applying meet greatestness to the two bounds $w \leq_C c$ and $w \leq_C b'$ gives

$$w \leq_C c \wedge b'.$$

Substituting the displayed definition of w gives the claimed comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeMeet monotone second argument from directional bounds`

Theorem 55.59 (Lattice operation monotonicity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44 and the directional meet and join bound-characterization fields from Definition 55.102. If $a, a', b, b', c \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \wedge c \leq_C a' \wedge c, \qquad c \wedge a \leq_C c \wedge a',$$

and

$$b \vee c \leq_C b' \vee c, \qquad c \vee b \leq_C c \vee b'.$$

Proof. The inherited preorder reflexivity field gives

$$c \leq_C c.$$

Apply Definition 55.95 to the comparisons $a \leq_C a'$ and $c \leq_C c$. This gives

$$a \wedge c \leq_C a' \wedge c.$$

Apply the same meet monotonicity lemma to $c \leq_C c$ and $a \leq_C a'$. This gives

$$c \wedge a \leq_C c \wedge a'.$$

Apply Definition 55.96 to $b \leq_C b'$ and $c \leq_C c$. This gives

$$b \vee c \leq_C b' \vee c.$$

Apply the same join monotonicity lemma to $c \leq_C c$ and $b \leq_C b'$. This gives

$$c \vee b \leq_C c \vee b'.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonOperation monotonicity empty`

Closure status for $\mathsf{NameCert}_{\mathsf{Lattice}^{\uparrow}}$.

Constructive story: The lattice instance is the singleton endpoint sitting over the prefix poset: the carrier accepts only histories history-same to e, so both meet and join are forced to return e no matter which carried operands are displayed. Its classifier is ρ_{e}, the same singleton relation that packages two endpoint carrier witnesses with an hsame link, and the inherited order is the restricted prefix comparison between those carried endpoints. The source is a certified poset carrier equipped with meet and join data, the pattern records lower-bound, greatest-lower-bound, upper-bound, least-upper-bound, congruence, directed associativity, absorption, and modular-law rows, and the ledger stores the operands with the bound-characterization witnesses that reduce every comparison to the empty endpoint. The five name-certificate obligations are immediate from the singleton endpoint: e inhabits the carrier, hsame gives the classifier equivalence, and carrier closure under classified replacement follows by composing endpoint witnesses through e. A history h lies in the signature gap of a $\mathsf{Lattice}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer endpoint and bound ledger, and token introduction names that answer as p; in this instance, h is classified with e, every meet or join ledger returns e, and p names the one-point lattice row. No standard bridge is closed in this chapter, although the intended outside reading is the terminal one-point lattice. The chapter registers this singleton meet-join construction under the name $\mathsf{Lattice}^{\uparrow}$; lattice laws are endpoint consequences of the existing prefix order, with no arbitrary lattice carrier, representation theorem, or external meet/join algebra introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Lattice}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public lattice certificate is closed over the displayed singleton-history carrier, classifier, order, meet, join, directed associativity, absorption, and modular-law rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Lattice}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.LatticeUp.LatticeSingletonPrefix\_laws`

Bridge status: bridgeChecked

Not claimed: It does not close full traditional lattice theory, arbitrary lattice carriers, representation theorems, or standard bridges.

Upgrade path: No further upgrade – chapter at top of formal axis.

55.4 Carrier definition

Definition 55.60 (Lattice carrier)**.** *Use a poset setup extended with meet and join operations carrying bound-characterization proofs.*

55.5 Concrete singleton prefix lattice instance

Definition 55.61 (Concrete singleton prefix lattice instance)**.** *Let A_{e} be the singleton history source of Definition 36.169 at $a_0 = \mathsf{e}$. Define ρ_{e} by*

$$\rho_{\mathsf{e}}(h,k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Let $h \leq_{\mathsf{e}} k$ mean that $A_{\mathsf{e}}(h)$ and $A_{\mathsf{e}}(k)$ hold and that $\mathsf{PreorderPrefixLE}(h,k)$ *witnesses the prefix comparison between the carried endpoints. Define the concrete lattice operations by*

$$\mathsf{meet}(h,k) := \mathsf{e}, \qquad \mathsf{join}(h,k) := \mathsf{e}.$$

The certificate has ten fields: traditional target the one-point lattice; BEDC source BHist *plus the singleton source and concrete prefix order; carrier A_{e}; classifier ρ_{e}; stability obligations for poset laws, meet lower bounds, meet greatestness, join upper bounds, join leastness, and meet and join congruence; ledger singleton endpoint plus prefix witnesses; core theorem Theorem 55.62; exactness singleton endpoint classification; standard bridge terminal one-point lattice; and status* ProofTarget.

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeUp StdBridge`

Theorem 55.62 (Concrete singleton prefix lattice laws)**.** *For all carried histories h, k, z in A_{e}, the endpoints* $\mathsf{meet}(h, k)$ *and* $\mathsf{join}(h, k)$ *are carried by A_{e} and satisfy the meet lower-bound laws, meet greatest-lower-bound law, join upper-bound laws, join least-upper-bound law, and meet and join congruence laws for the restricted prefix order $\leq_{\mathsf{e}}$ under the classifier ρ_{e}.*

Proof. The singleton carrier witnesses identify each carried endpoint with e. Thus $A_{\mathsf{e}}(h)$, $A_{\mathsf{e}}(k)$, and $A_{\mathsf{e}}(z)$ reduce every endpoint comparison needed by the order, meet, and join clauses to the empty-history endpoint.

The restricted prefix comparisons are supplied by PreorderPrefixLE_of_hsame after the singleton witnesses produce the relevant hsame facts. The inherited order boundary is the concrete unary prefix poset law package of Theorem 53.10, with antisymmetry supplied by Theorem 53.33.

The meet and join outputs are both definitionally e. The lower-bound and upper-bound clauses therefore use the singleton endpoint witness for e and the prefix witnesses from the previous paragraph; the greatest-lower-bound and least-upper-bound clauses use the same reduction for the third carried history z.

Congruence is immediate because both operations ignore their operands and return e. After unfolding, every source and target output is classified by $\rho_{\mathsf{e}}(\mathsf{e}, \mathsf{e})$, assembled from two copies of $A_{\mathsf{e}}(\mathsf{e})$ and the reflexive witness $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonPrefix laws`

Theorem 55.63 (Singleton lattice order has empty endpoints)**.** *For h, k : BHist, the restricted singleton prefix order satisfies*

$$h \leq_{\mathsf{e}} k \quad \Longleftrightarrow \quad h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction reads the two singleton carrier fields from the definition of $h \leq_{\mathsf{e}} k$. These fields are exactly $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$.

For the reverse direction, the two endpoint witnesses supply the singleton carrier fields. Their transitive composite through e gives $h\mathsf{hsame}k$, and the prefix comparison follows from the concrete PreorderPrefixLE_of_hsame constructor. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE empty endpoints iff`

Theorem 55.64 (Singleton lattice append order has empty endpoints)**.** *For h, k : BHist, the restricted singleton prefix order from the appended history to* e *satisfies*

$$\mathsf{LatticeSingletonLE}(\mathsf{append}(h, k), \mathsf{e}) \quad \Longleftrightarrow \quad h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction first applies singleton-order endpoint exactness to read $\mathsf{append}(h, k)\mathsf{hsamee}$, then splits the empty append into empty left and right endpoints. Conversely, empty endpoints make $\mathsf{append}(h, k)$ history-same to e, and the singleton-order endpoint characterization rebuilds the restricted order comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append empty endpoints iff`

Theorem 55.65 (Singleton lattice carrier order collapse)**.** *If h and k are carried by A_{e}, then both restricted order comparisons hold, the singleton classifier relates h and k, and both endpoints are internally the same as* e*:*

$$h \leq_{\mathsf{e}} k \wedge k \leq_{\mathsf{e}} h \wedge \rho_{\mathsf{e}}(h, k) \wedge h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The two carrier witnesses are the endpoint comparisons $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. Their composite through e gives $h\mathsf{hsame}k$, and symmetry gives $k\mathsf{hsame}h$.

The two restricted order comparisons are obtained from these two sameness witnesses by PreorderPrefixLE_of_hsame, together with the same carrier fields. The classifier package uses the carrier fields and $h\mathsf{hsame}k$, and the endpoint clauses are the original carrier witnesses. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonCarrier order collapse`

Theorem 55.66 (Singleton lattice antisymmetry classifier)**.** *If $h \leq_{\mathsf{e}} k$ and $k \leq_{\mathsf{e}} h$ in the restricted singleton prefix order, then $\rho_{\mathsf{e}}(h, k)$, $h\mathsf{hsamee}$, and $k\mathsf{hsamee}$.*

Proof. The two restricted comparisons carry their endpoint fields, so the first comparison already identifies both h and k with e. The prefix components of the two comparisons are opposing PreorderPrefixLE witnesses, and concrete prefix antisymmetry gives $\mathsf{hsame}(h, k)$. These three pieces are exactly the classifier data $\rho_{\mathsf{e}}(h, k)$, together with the two displayed empty endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE antisymm classifier`

Theorem 55.67 (Singleton lattice meet-join absorption classifier)**.** *If h and k are carried by A_{e}, then both absorption composites $\mathsf{meet}(h, \mathsf{join}(h, k))$ and $\mathsf{join}(h, \mathsf{meet}(h, k))$ are classified with h by ρ_{e}, and both output histories are hsame to e.*

Proof. Both concrete lattice operations are definitionally e. Hence each absorption composite is the empty history, whose singleton carrier witness is reflexive. The carrier witness for h gives $h\mathsf{hsamee}$, so symmetry gives the classifier component from each absorption output back to h. Reassemble the two ρ_{e} classifiers and record the two empty output endpoints. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption classifier`

Theorem 55.68 (Singleton lattice meet-join absorption is empty-classified)**.** *For carried singleton endpoints h, k, both absorption composites reduce to the empty endpoint and classify back to h:*

$$\mathsf{meet}(h, \mathsf{join}(h, k))\mathsf{hsamee}, \qquad \mathsf{join}(h, \mathsf{meet}(h, k))\mathsf{hsamee},$$
$$\rho_{\mathsf{e}}(\mathsf{meet}(h, \mathsf{join}(h, k)), h), \qquad \rho_{\mathsf{e}}(\mathsf{join}(h, \mathsf{meet}(h, k)), h).$$

Proof. Both concrete operations are definitionally the endpoint e. The two endpoint equalities are therefore reflexive empty-history sameness. Since h is carried by the singleton carrier, symmetry of its carrier witness classifies e with h, and the classifier packages the two carrier witnesses with that history-sameness witness. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption empty classified`

Theorem 55.69 (Singleton lattice continuation absorption has empty classifier)**.** *Let h, k be singleton-carried histories. If $left$ is a BEDC continuation output of $\mathsf{meet}(h, \mathsf{join}(h, k))$ followed by h, and $right$ is a BEDC continuation output of h followed by $\mathsf{join}(h, \mathsf{meet}(h, k))$, then both outputs are empty and are classified by the singleton classifier:*

$$\mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}) \wedge \rho_{\mathsf{e}}(left, right).$$

Proof. The concrete singleton meet and join both return e. The left continuation is therefore $\mathsf{append}(\mathsf{e}, h)$, which reduces to h, and the right continuation is $\mathsf{append}(h, \mathsf{e})$, which also reduces to h. The carrier witness for h identifies each result with e. Those two endpoint witnesses, together with transitivity through e, assemble the singleton classifier between the two continuation outputs. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin continuation absorption empty classifier`

Theorem 55.70 (Singleton lattice meet-join idempotent commutative classifier)**.** *For carried singleton endpoints h, k, meet and join are classified with the idempotent endpoint and with their swapped arguments:*

$$\rho_{\mathsf{e}}(\mathsf{meet}(h,h),h) \wedge \rho_{\mathsf{e}}(\mathsf{join}(h,h),h) \wedge \rho_{\mathsf{e}}(\mathsf{meet}(h,k),\mathsf{meet}(k,h)) \wedge \rho_{\mathsf{e}}(\mathsf{join}(h,k),\mathsf{join}(k,h)).$$

Moreover $\mathsf{meet}(h,k)\mathsf{hsamee}$ *and* $\mathsf{join}(h,k)\mathsf{hsamee}$.

Proof. The concrete singleton meet and join operations both return e. Thus the swapped meet outputs and swapped join outputs are classified by reflexive empty-history sameness. Since h is carried by the singleton carrier, symmetry of $h\mathsf{hsamee}$ classifies each idempotent output with h. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin idempotent comm classifier`

Theorem 55.71 (Singleton lattice classifier has empty endpoints)**.** *For $h, k : \mathsf{BHist}$, the singleton lattice classifier satisfies*

$$\rho_{\mathsf{e}}(h,k) \iff h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. The forward direction projects the two carrier fields from the classifier tuple. Since the singleton carrier is the predicate $x\mathsf{hsamee}$, these fields are exactly the two endpoint witnesses.

Conversely, the two endpoint witnesses make h and k singleton-carried. Their composite through e, using symmetry on the second witness, supplies $h\mathsf{hsame}k$ and hence the classifier tuple. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier empty endpoints iff`

Theorem 55.72 (Singleton lattice classifier continuation result-left iff)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P,Q,R)$, the singleton lattice classifier relates P and Q exactly when it relates the continuation result R to P:*

$$\mathsf{LatticeSingletonClassifier}(P,Q) \iff \mathsf{LatticeSingletonClassifier}(R,P).$$

Proof. The forward direction uses the singleton carriers of P and Q to transport the continuation result to the right-unit continuation at e, so R is singleton-carried and compares with P through e. For the reverse direction, the result-left classifier identifies R with e; transporting the continuation to empty result and applying empty-result inversion recovers the singleton carrier witnesses for both inputs, and hence the input classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result left iff`

Theorem 55.73 (Singleton lattice classifier continuation result-right iff)**.** *For histories $P, Q, R : \mathsf{BHist}$ with $\mathsf{Cont}(P,Q,R)$, the singleton lattice classifier relates P and Q exactly when it relates the continuation result R to Q:*

$$\mathsf{LatticeSingletonClassifier}(P,Q) \iff \mathsf{LatticeSingletonClassifier}(R,Q).$$

Proof. The forward direction uses the singleton carriers of P and Q to transport the continuation result to the right-unit continuation at e, so R is singleton-carried and compares with Q through e. For the reverse direction, the result-right classifier identifies R with e; transporting the continuation to empty result and applying empty-result inversion recovers the singleton carrier witnesses for both inputs, and hence the input classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result right iff`

Theorem 55.74 (Singleton lattice classifier continuation empty result exactness)**.** *For histories* P, Q, R : BHist *with* $\mathsf{Cont}(P, Q, R)$*, the singleton lattice classifier relates the two inputs exactly when the continuation result is empty:*

$$\mathsf{LatticeSingletonClassifier}(P, Q) \iff R\mathsf{hsamee}.$$

Proof. The forward direction reads the two singleton carrier endpoints from the input classifier; since R is the append continuation of P and Q, append-empty construction gives $R\mathsf{hsamee}$. Conversely, result emptiness inverts the append continuation into singleton carrier witnesses for both inputs, and the empty-endpoint characterization reconstructs the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation empty result iff`

Theorem 55.75 (Singleton lattice classifier continuation result-left-right equivalence)**.** *For histories* P, Q, R : BHist *with* $\mathsf{Cont}(P, Q, R)$*, classifying the continuation result against the left input is equivalent to classifying it against the right input:*

$$\mathsf{LatticeSingletonClassifier}(R, P) \iff \mathsf{LatticeSingletonClassifier}(R, Q).$$

Proof. The result-left classifier theorem identifies the left comparison with the input classifier on P and Q. The result-right classifier theorem identifies the right comparison with the same input classifier. Composing those two characterizations gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier continuation result left right iff`

Theorem 55.76 (Singleton lattice result-left order carriers)**.** *For histories* h, k, r : BHist *with* $\mathsf{Cont}(h, k, r)$*, the restricted singleton order from the continuation result back to the left input is equivalent to both exposed inputs being singleton-carried:*

$$\mathsf{LatticeSingletonLE}(r, h) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction reads $r\mathsf{hsamee}$ from the singleton order endpoint fields. The continuation witness identifies r with $\mathsf{append}(h, k)$, so empty-append inversion gives $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$.

Conversely, if both exposed inputs are singleton-carried, the continuation result is internally the empty endpoint. The empty-endpoint characterization of the singleton order then supplies $\mathsf{LatticeSingletonLE}(r, h)$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result left carriers iff`

Theorem 55.77 (Singleton lattice result-right order carriers)**.** *For histories* h, k, r : BHist *with* $\mathsf{Cont}(h, k, r)$*, the restricted singleton order from the continuation result back to the right input is equivalent to both exposed inputs being singleton-carried:*

$$\mathsf{LatticeSingletonLE}(r, k) \iff A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k).$$

Proof. The forward direction again projects $r\mathsf{hsamee}$ from the singleton order endpoint fields. Since the continuation result is the append of h and k, empty-append inversion exposes both input carriers.

In the reverse direction, the two input carriers force the continuation result to be internally empty. The empty-endpoint characterization of $\mathsf{LatticeSingletonLE}$ then gives the displayed order comparison against the right input. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result right carriers iff`

Theorem 55.78 (Singleton lattice continuation result endpoint order equivalence)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the restricted singleton order from the continuation result to the left endpoint is equivalent to the restricted singleton order from the result to the right endpoint:*

$$\mathsf{LatticeSingletonLE}(R, P) \iff \mathsf{LatticeSingletonLE}(R, Q).$$

Proof. The result-left order theorem identifies the left comparison with the pair of singleton carrier witnesses for P and Q. The result-right order theorem identifies the right comparison with the same pair. Composing these two characterizations gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation result left right iff`

Theorem 55.79 (Singleton lattice continuation empty result order exactness)**.** *For histories* $P, Q, R : \mathsf{BHist}$ *with* $\mathsf{Cont}(P, Q, R)$*, the restricted singleton prefix comparison between the inputs is equivalent to emptiness of the continuation result:*

$$P \leq_{\mathsf{e}} Q \iff R\mathsf{hsamee}.$$

Proof. The restricted singleton order carries both input endpoints as empty. Since R is the BEDC append continuation of P and Q, append-empty construction gives $R\mathsf{hsamee}$. Conversely, if the continuation result is empty, BEDC append-empty inversion makes both P and Q empty. Their induced hsame chain supplies the underlying prefix comparison, so the singleton order tuple is reconstructed. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE continuation empty result iff`

Theorem 55.80 (Singleton meet–join absorption gives order and classifier data)**.** *If* $h, k : \mathsf{BHist}$ *are singleton-carried, then the two singleton lattice absorption endpoints*

$$h \wedge (h \vee k) \qquad \textit{and} \qquad h \vee (h \wedge k)$$

are both definitionally the empty history. They satisfy the restricted prefix order comparisons

$$h \wedge (h \vee k) \leq_{\mathsf{e}} h \qquad \textit{and} \qquad h \leq_{\mathsf{e}} h \vee (h \wedge k),$$

are both classified with h*, and both read back to* e*.*

Proof. In the singleton lattice, meet and join ignore their inputs and return e. The carrier witness for h gives $h\mathsf{hsamee}$, while reflexivity gives $\mathsf{ehsamee}$.

The first order comparison uses symmetry of $h\mathsf{hsamee}$ to obtain $\mathsf{ehsame}h$ and then applies the prefix comparison constructor. The second uses $h\mathsf{hsamee}$ directly. The same two hsame witnesses assemble the classifier tuples with h, and the endpoint readbacks are both reflexivity at e. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeetJoin absorption order classifier`

Theorem 55.81 (Singleton meet greatest lower bound)**.** *For carried histories* h, k, z *in the singleton lattice, the concrete meet output is the empty history, and the lower-bound condition against that meet is exactly the pair of lower-bound conditions against the two operands:*

$$\mathsf{hsame}(\mathsf{LatticeSingletonMeet}(h, k), \mathsf{e}) \wedge (z \leq_{\mathsf{e}} \mathsf{LatticeSingletonMeet}(h, k) \iff z \leq_{\mathsf{e}} h \wedge z \leq_{\mathsf{e}} k)\,.$$

Proof. The meet operation is definitionally the empty history. Since h, k, z are all carried by the singleton carrier, the order endpoint theorem Definition 55.63 reduces every displayed comparison to the two empty-endpoint witnesses. The forward direction rebuilds the two operand bounds from the endpoint witnesses for z, h, k; the reverse direction rebuilds the meet bound from the endpoint witness for z and reflexivity at e. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet empty output greatest lower bound iff`

Theorem 55.82 (Singleton join least upper bound)**.** *For carried histories h, k, z in the singleton lattice, the concrete join least-upper-bound comparison is exactly the pair of upper-bound comparisons against the two operands, together with the empty readback of the join:*

$$\mathsf{LatticeSingletonJoin}(h, k) \leq_{\mathsf{e}} z \iff h \leq_{\mathsf{e}} z \wedge k \leq_{\mathsf{e}} z \wedge \mathsf{LatticeSingletonJoin}(h, k)\mathsf{hsamee}.$$

Proof. The join operation is definitionally the empty history. Since h, k, z are all carried by the singleton carrier, the singleton order comparison between any two of them is built from the corresponding hsame chain through e. The forward direction reconstructs the two operand upper bounds and records reflexivity of the join endpoint; the reverse direction uses the displayed join endpoint together with the carrier of z to rebuild the join bound. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoin least upper bound empty iff`

Theorem 55.83 (Singleton lattice append tail emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$ and $\mathsf{UnaryHistory}(t)$, then the restricted singleton prefix order satisfies*

$$\mathsf{append}(h, t) \leq_{\mathsf{e}} h \quad \iff \quad t\mathsf{hsamee}.$$

Proof. The forward direction projects the underlying prefix comparison from the restricted order and applies the concrete append-tail backforcing law from $\mathsf{Preorder}^{\uparrow}$. A unary tail that returns from $\mathsf{append}(h, t)$ back to h must be the empty history.

For the reverse direction, the witness $t\mathsf{hsamee}$ reduces $\mathsf{append}(h, t)$ to h by the right continuation unit law. The carried witness $A_{\mathsf{e}}(h)$ therefore also carries $\mathsf{append}(h, t)$, and the prefix comparison is the empty-tail side of the same append-tail equivalence. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append tail empty iff`

Theorem 55.84 (Singleton lattice classifier append tail emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$, then the singleton classifier satisfies*

$$\mathsf{LatticeSingletonClassifier}(\mathsf{append}(h, t), h) \quad \iff \quad t\mathsf{hsamee}.$$

Proof. The forward direction projects the carried left endpoint from the classifier. Since $\mathsf{append}(h, t)\mathsf{hsamee}$, the concrete append-empty inversion forces $t\mathsf{hsamee}$.

Conversely, assume $t\mathsf{hsamee}$. The append endpoint is carried because both h and t are empty in the singleton sense, and the right continuation unit identifies $\mathsf{append}(h, t)$ with h. Together with the given carrier for h, these witnesses reconstruct the singleton classifier. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonClassifier append tail empty iff`

Theorem 55.85 (Singleton lattice append context empty order)**.** *For $left, h, right : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$, then the restricted singleton prefix order between the two append displays satisfies*

$$\mathsf{append}(left, h) \leq_{\mathsf{e}} \mathsf{append}(h, right) \quad \iff \quad left\mathsf{hsamee} \wedge right\mathsf{hsamee}.$$

Proof. In the forward direction, the restricted order exposes singleton carriers for both append endpoints. BEDC append-empty inversion on $\mathsf{append}(left, h)$ gives $left\mathsf{hsamee}$, and the same inversion on $\mathsf{append}(h, right)$ gives $right\mathsf{hsamee}$.

Conversely, if the two contexts are empty, the carrier of h makes both append endpoints singleton-carried. Since both endpoints are history-same to e, they are history-same to each other, and the singleton order is rebuilt from the corresponding concrete prefix witness. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append context empty iff`

Theorem 55.86 (Singleton lattice append tail upper emptiness)**.** *For $h, t : \mathsf{BHist}$, if $A_{\mathsf{e}}(h)$ and $\mathsf{UnaryHistory}(t)$, then the restricted singleton prefix order satisfies*

$$h \leq_{\mathsf{e}} \mathsf{append}(h, t) \quad \Longleftrightarrow \quad t\mathsf{hsamee}.$$

Proof. The forward direction reads the upper endpoint carrier from the restricted order. Thus $\mathsf{append}(h, t)$ is history-same to e, and the BEDC append-empty inversion forces both endpoints of the append, in particular t, to be empty. Conversely, if $t\mathsf{hsamee}$, the right continuation unit identifies $\mathsf{append}(h, t)$ with h, so the given singleton carrier for h also carries the append endpoint. The prefix witness is the concrete append-tail continuation. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonLE append tail upper empty iff`

Theorem 55.87 (Singleton lattice meet greatest lower bound empty exactness)**.** *For carried singleton endpoints h, k, z, the statement that z is below the concrete meet of h and k is equivalent to the two endpoint lower-bound clauses together with the fact that the concrete meet is empty:*

$$z \leq_{\mathsf{e}} \mathsf{meet}(h, k) \quad \Longleftrightarrow \quad z \leq_{\mathsf{e}} h \wedge z \leq_{\mathsf{e}} k \wedge \mathsf{meet}(h, k)\mathsf{hsamee}.$$

Proof. The singleton carrier identifies h, k, z with e, and the concrete meet operation is definitionally e. Thus the forward direction builds both lower-bound comparisons from the endpoint sameness witnesses using the singleton prefix comparison. Conversely, the displayed empty-meet endpoint and the carrier witness for z build the singleton comparison $z \leq_{\mathsf{e}} \mathsf{meet}(h, k)$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet greatest lower bound empty iff`

The $\mathsf{Lattice}^{\uparrow}$ schema is instantiated by a concrete BHist carrier and operations, so Lean can target $\mathsf{BEDC.Derived.LatticeUp}$ as a BHist-anchored certificate rather than an abstract operator schema.

Lemma 55.88 (Directed meet associativity comparison from bounds)**.** *Let A be a $\mathsf{Lattice}^{\uparrow}$ carrier with order $\leq$ and classifier $\sim_C$. Assume the inherited $\mathsf{Preorder}^{\uparrow}$ transitivity field from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound implication fields from Definition 55.102. Then for all $x, y, z \in A$,*

$$(x \wedge y) \wedge z \leq x \wedge (y \wedge z).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet assoc comparison empty`

Proof. Put

$$w = (x \wedge y) \wedge z.$$

The meet lower-left field gives

$$w \leq x \wedge y,$$

and the meet lower-left field applied to $x \wedge y$ gives

$$x \wedge y \leq x.$$

By the inherited preorder transitivity field,

$$w \leq x.$$

The meet lower-right field applied to $x \wedge y$ gives

$$x \wedge y \leq y,$$

so transitivity with $w \leq x \wedge y$ gives

$$w \leq y.$$

The meet lower-right field applied to $w = (x \wedge y) \wedge z$ gives

$$w \leq z.$$

Applying meet greatestness to the two bounds $w \leq y$ and $w \leq z$ gives

$$w \leq y \wedge z.$$

A second application of meet greatestness, now to $w \leq x$ and $w \leq y \wedge z$, gives

$$w \leq x \wedge (y \wedge z).$$

Substituting the displayed definition of w is the required comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirectedMeet associativity comparison from bounds`

Lemma 55.89 (Directed join associativity comparison from bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional join upper-bound and least-upper-bound implication fields from Definition 55.102. Then for all $x, y, z \in A$, $(x \vee y) \vee z \leq_C x \vee (y \vee z)$.*

Proof. Put $w = x \vee (y \vee z)$. The join upper-left field gives $x \leq_C w$. The join upper-left field applied to $y \vee z$ gives $y \leq_C y \vee z$, and the join upper-right field applied to $w = x \vee (y \vee z)$ gives $y \vee z \leq_C w$. By the inherited preorder transitivity field, $y \leq_C w$. Applying join leastness to the two bounds $x \leq_C w$ and $y \leq_C w$ gives $x \vee y \leq_C w$. The join upper-right field applied to $y \vee z$ gives $z \leq_C y \vee z$, and transitivity with $y \vee z \leq_C w$ gives $z \leq_C w$. A second application of join leastness, now to $x \vee y \leq_C w$ and $z \leq_C w$, gives $(x \vee y) \vee z \leq_C w$. Substituting the definition of w gives $(x \vee y) \vee z \leq_C x \vee (y \vee z)$. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoin directed assoc comparison from bounds`

Lemma 55.90 (Meet absorbs a smaller ordered endpoint)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity field from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $x, z \in A$ and $x \leq_C z$, then*

$$z \wedge x \sim_C x.$$

Proof. The meet lower-right field gives

$$z \wedge x \leq_C x.$$

The inherited preorder reflexivity field gives

$$x \leq_C x.$$

Together with the assumed comparison $x \leq_C z$, meet greatestness applied with witness endpoint x and operands z and x gives

$$x \leq_C z \wedge x.$$

The antisymmetry field from Definition 53.43 applied to these two comparisons yields

$$z \wedge x \sim_C x.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeMeet absorbs smaller ordered endpoint from bounds`

Lemma 55.91 (Modular lower comparison from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound, meet greatestness, join upper-bound, and join leastness fields from Definition 55.102. If $x, y, z \in A$ and $x \leq_C z$, then*

$$x \vee (y \wedge z) \leq_C (x \vee y) \wedge z.$$

Proof. Put

$$n = x \vee (y \wedge z), \qquad m = (x \vee y) \wedge z.$$

The join upper-left field gives

$$x \leq_C x \vee y.$$

The meet lower-left field gives

$$y \wedge z \leq_C y,$$

and the join upper-right field gives

$$y \leq_C x \vee y.$$

By transitivity,

$$y \wedge z \leq_C x \vee y.$$

Join leastness with upper target $x \vee y$ gives

$$n \leq_C x \vee y.$$

The assumption gives $x \leq_C z$, and the meet lower-right field gives

$$y \wedge z \leq_C z.$$

Join leastness with upper target z gives

$$n \leq_C z.$$

Meet greatestness applied to the two bounds $n \leq_C x \vee y$ and $n \leq_C z$ gives

$$n \leq_C m.$$

Substituting the displayed definitions of m and n gives the claimed comparison. □

Lean4 ✓ BEDC.Derived.LatticeUp.LatticeModular lower comparison from directional bounds

Theorem 55.92 (Distributivity gives the modular law)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the classifier symmetry and transitivity fields from Definition 9.7, the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet and join bound-characterization fields from Definition 55.102. Assume also the join classifier-congruence field from Definition 55.101, and suppose the distributive law*

$$a \wedge (b \vee c) \sim_C (a \wedge b) \vee (a \wedge c)$$

holds for all $a, b, c \in A$. If $x, y, z \in A$ and $x \leq_C z$, then

$$x \vee (y \wedge z) \leq_C (x \vee y) \wedge z \qquad \textit{and} \qquad x \vee (y \wedge z) \sim_C (x \vee y) \wedge z.$$

Lean4 ✓ BEDC.Derived.LatticeUp.LatticeSingleton distributivity modular law

Proof. The first conclusion is Definition 55.91.

Put

$$m = (x \vee y) \wedge z, \qquad n = x \vee (y \wedge z).$$

By Definition 55.110, applied to the meet operands $x \vee y$ and z,

$$m \sim_C z \wedge (x \vee y).$$

The assumed distributive law with $a = z$, $b = x$, and $c = y$ gives

$$z \wedge (x \vee y) \sim_C (z \wedge x) \vee (z \wedge y).$$

By Definition 55.90 and the assumption $x \leq_C z$,

$$z \wedge x \sim_C x.$$

By Definition 55.110, applied to the meet operands z and y,

$$z \wedge y \sim_C y \wedge z.$$

Join classifier congruence applied to the last two classifier comparisons gives

$$(z \wedge x) \vee (z \wedge y) \sim_C x \vee (y \wedge z).$$

Classifier transitivity composes the three displayed classifier comparisons into

$$m \sim_C n.$$

Classifier symmetry reverses this comparison, giving

$$n \sim_C m.$$

Substituting the displayed definitions of m and n gives

$$x \vee (y \wedge z) \sim_C (x \vee y) \wedge z.$$

□

Lemma 55.93 (Join-meet lower comparison from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound, meet greatestness, join upper-bound, and join leastness fields from Definition 55.102. Then for all $a, b, c \in A$,*

$$a \vee (b \wedge c) \leq_C (a \vee b) \wedge (a \vee c).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonJoinMeet lower comparison empty`

Proof. Put

$$n = a \vee (b \wedge c), \qquad m = (a \vee b) \wedge (a \vee c).$$

The join upper-left field gives

$$a \leq_C a \vee b.$$

The meet lower-left field gives

$$b \wedge c \leq_C b,$$

and the join upper-right field gives

$$b \leq_C a \vee b.$$

By transitivity,

$$b \wedge c \leq_C a \vee b.$$

Join leastness with upper target $a \vee b$ gives

$$n \leq_C a \vee b.$$

Similarly, the join upper-left field gives

$$a \leq_C a \vee c.$$

The meet lower-right field gives

$$b \wedge c \leq_C c,$$

and the join upper-right field gives

$$c \leq_C a \vee c.$$

By transitivity,

$$b \wedge c \leq_C a \vee c.$$

Join leastness with upper target $a \vee c$ gives

$$n \leq_C a \vee c.$$

Meet greatestness applied to the two bounds $n \leq_C a \vee b$ and $n \leq_C a \vee c$ gives

$$n \leq_C m.$$

Substituting the displayed definitions of m and n gives the claimed comparison. □

Theorem 55.94 (Meet distributivity gives join distributivity)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the classifier symmetry and transitivity fields from Definition 9.7, the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44, the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43, and the directional meet and join bound-characterization fields from Definition 55.102. Assume also the meet and join classifier-congruence rows from Definition 55.101, and suppose the distributive law*

$$u \wedge (v \vee w) \sim_C (u \wedge v) \vee (u \wedge w)$$

holds for all $u, v, w \in A$. Then for all $a, b, c \in A$,

$$a \vee (b \wedge c) \leq_C (a \vee b) \wedge (a \vee c) \qquad \textit{and} \qquad a \vee (b \wedge c) \sim_C (a \vee b) \wedge (a \vee c).$$

Proof. The first conclusion is Definition 55.93.

Put

$$d = (a \vee b) \wedge (a \vee c), \qquad n = a \vee (b \wedge c).$$

The assumed distributive law with $u = a \vee b$, $v = a$, and $w = c$ gives

$$d \sim_C ((a \vee b) \wedge a) \vee ((a \vee b) \wedge c).$$

By Definition 55.111, applied with $x = a$ and $y = b$,

$$(a \vee b) \wedge a \sim_C a.$$

By Definition 55.110, applied to the meet operands $a \vee b$ and c,

$$(a \vee b) \wedge c \sim_C c \wedge (a \vee b).$$

The assumed distributive law with $u = c$, $v = a$, and $w = b$ gives

$$c \wedge (a \vee b) \sim_C (c \wedge a) \vee (c \wedge b).$$

Classifier transitivity composes the last two comparisons into

$$(a \vee b) \wedge c \sim_C (c \wedge a) \vee (c \wedge b).$$

Join classifier congruence gives

$$((a \vee b) \wedge a) \vee ((a \vee b) \wedge c) \sim_C a \vee ((c \wedge a) \vee (c \wedge b)).$$

Put

$$p = a \vee ((c \wedge a) \vee (c \wedge b)), \qquad q = a \vee (c \wedge b).$$

To compare p and q, first note that the join upper-left field gives $a \leq_C q$. The meet lower-right field gives $c \wedge a \leq_C a$, so transitivity gives $c \wedge a \leq_C q$. The join upper-right field gives $c \wedge b \leq_C q$. Join leastness gives

$$(c \wedge a) \vee (c \wedge b) \leq_C q,$$

and another application of join leastness gives

$$p \leq_C q.$$

Conversely, the join upper-left field gives $a \leq_C p$. The join upper-right field gives

$$(c \wedge a) \vee (c \wedge b) \leq_C p,$$

and the join upper-right field applied to the inner join gives

$$c \wedge b \leq_C (c \wedge a) \vee (c \wedge b).$$

By transitivity, $c \wedge b \leq_C p$. Join leastness gives

$$q \leq_C p.$$

The antisymmetry field from Definition 53.43 applied to $p \leq_C q$ and $q \leq_C p$ yields

$$p \sim_C q.$$

By Definition 55.108, applied with $x = a$ and $y = a$,

$$a \vee (a \wedge a) \sim_C a.$$

Classifier symmetry and transitivity applied to this comparison give

$$a \sim_C a.$$

By Definition 55.110, applied to the meet operands c and b,

$$c \wedge b \sim_C b \wedge c.$$

Join classifier congruence gives

$$q \sim_C n.$$

Classifier transitivity composes the displayed classifier comparisons into

$$d \sim_C n.$$

Classifier symmetry reverses this comparison, giving

$$n \sim_C d.$$

Substituting the displayed definitions of d and n gives

$$a \vee (b \wedge c) \sim_C (a \vee b) \wedge (a \vee c).$$

□

Lemma 55.95 (Lattice meet two-sided monotonicity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $a, a', b, b' \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \wedge b \leq_C a' \wedge b'.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonMeet monotone empty`

Proof. Put

$$w = a \wedge b.$$

The meet lower-left field gives

$$w \leq_C a.$$

Composing this comparison with the assumed comparison $a \leq_C a'$ by the inherited preorder transitivity field gives

$$w \leq_C a'.$$

The meet lower-right field gives

$$w \leq_C b.$$

Composing this comparison with the assumed comparison $b \leq_C b'$ gives

$$w \leq_C b'.$$

Applying meet greatestness to the two bounds $w \leq_C a'$ and $w \leq_C b'$ gives

$$w \leq_C a' \wedge b'.$$

Substituting the displayed definition of w gives the claimed comparison. □

Lemma 55.96 (Join is monotone in both ordered endpoints)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *transitivity field from Definition 52.44 and the directional join upper-bound and least-upper-bound fields from Definition 55.102. If $a, b, a', b' \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \vee b \leq_C a' \vee b'.$$

Proof. Put

$$u = a' \vee b'.$$

The join upper-left field gives

$$a' \leq_C u.$$

Composing the assumed comparison $a \leq_C a'$ with $a' \leq_C u$ by the inherited preorder transitivity field gives

$$a \leq_C u.$$

The join upper-right field gives

$$b' \leq_C u.$$

Composing the assumed comparison $b \leq_C b'$ with $b' \leq_C u$ by the same transitivity field gives

$$b \leq_C u.$$

Applying join leastness to the two upper-bound comparisons $a \leq_C u$ and $b \leq_C u$ gives

$$a \vee b \leq_C u.$$

Substituting the displayed definition of u gives the claimed comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeJoin two sided monotonicity from directional bounds`

Lemma 55.97 (Lattice meet monotonicity in the second argument from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44 and the directional meet lower-bound and greatest-lower-bound fields from Definition 55.102. If $b, b', c \in A$ and $b \leq_C b'$, then*

$$c \wedge b \leq_C c \wedge b'.$$

Proof. Put

$$w = c \wedge b.$$

The meet lower-left field gives

$$w \leq_C c.$$

The meet lower-right field gives

$$w \leq_C b.$$

Composing this comparison with the assumed comparison $b \leq_C b'$ by the inherited preorder transitivity field gives

$$w \leq_C b'.$$

Applying meet greatestness to the two bounds $w \leq_C c$ and $w \leq_C b'$ gives

$$w \leq_C c \wedge b'.$$

Substituting the displayed definition of w gives the claimed comparison. □

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeMeet monotone second argument from directional bounds`

Theorem 55.98 (Lattice operation monotonicity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with order $\leq_C$ and classifier $\sim_C$. Assume the inherited* $\mathsf{Preorder}^{\uparrow}$ *reflexivity and transitivity fields from Definition 52.44 and the directional meet and join bound-characterization fields from Definition 55.102. If $a, a', b, b', c \in A$ with $a \leq_C a'$ and $b \leq_C b'$, then*

$$a \wedge c \leq_C a' \wedge c, \qquad c \wedge a \leq_C c \wedge a',$$

and

$$b \vee c \leq_C b' \vee c, \qquad c \vee b \leq_C c \vee b'.$$

Proof. The inherited preorder reflexivity field gives

$$c \leq_C c.$$

Apply Definition 55.95 to the comparisons $a \leq_C a'$ and $c \leq_C c$. This gives

$$a \wedge c \leq_C a' \wedge c.$$

Apply the same meet monotonicity lemma to $c \leq_C c$ and $a \leq_C a'$. This gives

$$c \wedge a \leq_C c \wedge a'.$$

Apply Definition 55.96 to $b \leq_C b'$ and $c \leq_C c$. This gives

$$b \vee c \leq_C b' \vee c.$$

Apply the same join monotonicity lemma to $c \leq_C c$ and $b \leq_C b'$. This gives

$$c \vee b \leq_C c \vee b'.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingletonOperation monotonicity empty`

55.6 The certificate

Definition 55.99 (Lattice source specification)**.** *The source is a certified poset carrier plus binary meet and join data.*

Definition 55.100 (Lattice pattern specification)**.** *The pattern is expression formation by $\wedge$ and $\vee$ over carrier witnesses.*

Definition 55.101 (Lattice classifier specification)**.** *The classifier is the poset carrier classifier, with meet and join congruent in both arguments.*

Definition 55.102 (Lattice stability certificate)**.** *The stability certificate consists of:*

1. *all poset stability fields*
2. *meet lower-bound laws*
3. *meet greatestness law*
4. *join upper-bound laws*
5. *join leastness law*
6. *meet and join congruence*

Definition 55.103 (Lattice ledger policy)**.** *Record operands and the bound-characterization certificate for each meet or join.*

Definition 55.104 ($\mathsf{Lattice}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Lattice}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Lattice}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Lemma 55.105 (Meet idempotence from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For every $x \in A$,*

$$x \wedge x \sim_C x.$$

Proof. The meet lower-left field gives

$$x \wedge x \leq x.$$

The inherited preorder reflexivity field gives $x \leq x$. Applying meet greatestness with $z = x$, $a = x$, and $b = x$, using the two reflexivity witnesses $x \leq x$ and $x \leq x$, gives

$$x \leq x \wedge x.$$

The inherited poset antisymmetry field applied to

$$x \wedge x \leq x \qquad \text{and} \qquad x \leq x \wedge x$$

yields

$$x \wedge x \sim_C x.$$

□

Lemma 55.106 (Join idempotence from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For every $x \in A$,*

$$x \vee x \sim_C x.$$

Proof. The join upper-left field gives

$$x \leq x \vee x.$$

The inherited preorder reflexivity field gives $x \leq x$. Applying join leastness with $z = x$, $a = x$, and $b = x$, using the two reflexivity witnesses $x \leq x$ and $x \leq x$, gives

$$x \vee x \leq x.$$

The inherited poset antisymmetry field applied to

$$x \vee x \leq x \qquad \text{and} \qquad x \leq x \vee x$$

yields

$$x \vee x \sim_C x.$$

□

Lemma 55.107 (Meet absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For all $x, y \in A$,*

$$x \wedge (x \vee y) \sim_C x.$$

Proof. The meet lower-left field gives

$$x \wedge (x \vee y) \leq x.$$

For the reverse comparison, the inherited preorder reflexivity field gives

$$x \leq x.$$

The join upper-left field gives

$$x \leq x \vee y.$$

Applying meet greatestness with $z = x$, $a = x$, and $b = x \vee y$, using the two comparisons above, gives

$$x \leq x \wedge (x \vee y).$$

The inherited poset antisymmetry field applied to

$$x \wedge (x \vee y) \leq x \qquad \text{and} \qquad x \leq x \wedge (x \vee y)$$

yields

$$x \wedge (x \vee y) \sim_C x.$$

□

Lemma 55.108 (Join absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. For all $x, y \in A$,*

$$x \vee (x \wedge y) \sim_C x.$$

Proof. The join upper-left field gives

$$x \leq x \vee (x \wedge y).$$

For the reverse comparison, the inherited preorder reflexivity field gives

$$x \leq x.$$

The meet lower-left field gives

$$x \wedge y \leq x.$$

Applying join leastness with $z = x$, $a = x$, and $b = x \wedge y$, using the two comparisons above, gives

$$x \vee (x \wedge y) \leq x.$$

The inherited poset antisymmetry field applied to

$$x \vee (x \wedge y) \leq x \qquad \text{and} \qquad x \leq x \vee (x \wedge y)$$

yields

$$x \vee (x \wedge y) \sim_C x.$$

□

Theorem 55.109 (Lattice idempotence and absorption from bound characterization)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *reflexivity and antisymmetry fields, and assume the directional meet and join bound-characterization fields:*

$$\begin{gathered}
\forall a\, b,\ a \wedge b \leq a, \qquad \forall a\, b,\ a \wedge b \leq b, \\
\forall z\, a\, b,\ z \leq a \to z \leq b \to z \leq a \wedge b, \\
\forall a\, b,\ a \leq a \vee b, \qquad \forall a\, b,\ b \leq a \vee b, \\
\forall z\, a\, b,\ a \leq z \to b \leq z \to a \vee b \leq z.
\end{gathered}$$

Then for all $x, y \in A$,

$$x \wedge x \sim_C x, \qquad x \vee x \sim_C x, \qquad x \wedge (x \vee y) \sim_C x, \qquad x \vee (x \wedge y) \sim_C x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected idempotence absorption from bounds`

Proof. The first assertion is Lemma 55.105. The second assertion is Lemma 55.106. The third assertion is Lemma 55.107. The fourth assertion is Lemma 55.108. □

Theorem 55.110 (Lattice commutativity from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y \in A$,*

$$x \wedge y \sim_C y \wedge x, \qquad x \vee y \sim_C y \vee x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected commutativity from bounds`

Proof. For the meet comparison, the meet lower-bound laws in Definition 55.102 give

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y.$$

Applying meet greatestness with $z = x \wedge y$, $a = y$, and $b = x$ gives

$$x \wedge y \leq y \wedge x.$$

The same lower-bound laws applied to $y \wedge x$ give

$$y \wedge x \leq y \qquad \text{and} \qquad y \wedge x \leq x.$$

Applying meet greatestness with $z = y \wedge x$, $a = x$, and $b = y$ gives

$$y \wedge x \leq x \wedge y.$$

Since $\mathsf{Lattice}^{\uparrow}$ extends the $\mathsf{POSet}^{\uparrow}$ layer, the antisymmetry field recorded in Definition 53.43 applies to these two comparisons and yields

$$x \wedge y \sim_C y \wedge x.$$

For the join comparison, the join upper-bound laws in Definition 55.102 give

$$y \leq x \vee y \qquad \text{and} \qquad x \leq x \vee y.$$

Applying join leastness with $z = x \vee y$, $a = y$, and $b = x$ gives

$$y \vee x \leq x \vee y.$$

The same upper-bound laws applied to $y \vee x$ give

$$x \leq y \vee x \qquad \text{and} \qquad y \leq y \vee x.$$

Applying join leastness with $z = y \vee x$, $a = x$, and $b = y$ gives

$$x \vee y \leq y \vee x.$$

The inherited antisymmetry field from Definition 53.43 applied to these two comparisons yields

$$x \vee y \sim_C y \vee x.$$

□

Theorem 55.111 (Lattice opposite absorption from directional bounds)**.** *Let A be a $\mathsf{Lattice}^{\uparrow}$ carrier with classifier $\sim_C$. Assume the inherited preorder reflexivity field, the inherited $\mathsf{POSet}^{\uparrow}$ antisymmetry field, and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y \in A$,*

$$x \wedge (y \vee x) \sim_C x, \qquad (x \vee y) \wedge x \sim_C x, \qquad x \vee (y \wedge x) \sim_C x, \qquad (x \wedge y) \vee x \sim_C x.$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeOppositeAbsorptionFromDirectionalBounds`

Proof. The inherited reflexivity field is the preorder reflexivity component of Definition 52.44; the inherited classifier collapse is the antisymmetry component of Definition 53.43. The meet and join directional fields are those listed in Definition 55.102.

For $x \wedge (y \vee x)$, the meet lower-left field gives

$$x \wedge (y \vee x) \leq x.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

The join upper-right field applied to $y \vee x$ gives

$$x \leq y \vee x.$$

Applying meet greatestness with $z = x$, $a = x$, and $b = y \vee x$, using the two comparisons above, gives

$$x \leq x \wedge (y \vee x).$$

Antisymmetry from Definition 53.43 applied to

$$x \wedge (y \vee x) \leq x \qquad \text{and} \qquad x \leq x \wedge (y \vee x)$$

yields

$$x \wedge (y \vee x) \sim_C x.$$

For $(x \vee y) \wedge x$, the meet lower-right field gives

$$(x \vee y) \wedge x \leq x.$$

The join upper-left field gives

$$x \leq x \vee y.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

Applying meet greatestness with $z = x$, $a = x \vee y$, and $b = x$, using these two comparisons, gives

$$x \leq (x \vee y) \wedge x.$$

Antisymmetry applied to

$$(x \vee y) \wedge x \leq x \qquad \text{and} \qquad x \leq (x \vee y) \wedge x$$

yields

$$(x \vee y) \wedge x \sim_C x.$$

For $x \vee (y \wedge x)$, the join upper-left field gives

$$x \leq x \vee (y \wedge x).$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

The meet lower-right field applied to $y \wedge x$ gives

$$y \wedge x \leq x.$$

Applying join leastness with $z = x$, $a = x$, and $b = y \wedge x$, using the two comparisons above, gives

$$x \vee (y \wedge x) \leq x.$$

Antisymmetry applied to

$$x \vee (y \wedge x) \leq x \qquad \text{and} \qquad x \leq x \vee (y \wedge x)$$

yields

$$x \vee (y \wedge x) \sim_C x.$$

For $(x \wedge y) \vee x$, the join upper-right field gives

$$x \leq (x \wedge y) \vee x.$$

The meet lower-left field gives

$$x \wedge y \leq x.$$

The inherited preorder reflexivity field gives

$$x \leq x.$$

Applying join leastness with $z = x$, $a = x \wedge y$, and $b = x$, using these two comparisons, gives

$$(x \wedge y) \vee x \leq x.$$

Antisymmetry applied to

$$(x \wedge y) \vee x \leq x \qquad \text{and} \qquad x \leq (x \wedge y) \vee x$$

yields

$$(x \wedge y) \vee x \sim_C x.$$

□

Theorem 55.112 (Lattice bound uniqueness from directional certificates)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited* $\mathsf{POSet}^{\uparrow}$ *antisymmetry field from Definition 53.43 and the directional meet and join bound-characterization fields from Definition 55.102. For all $x, y, m, j \in A$, suppose m satisfies the lower-bound and greatestness clauses*

$$m \leq x, \qquad m \leq y,$$
$$\forall z \in A,\ z \leq x \to z \leq y \to z \leq m,$$

and suppose j satisfies the upper-bound and leastness clauses

$$x \leq j, \qquad y \leq j,$$
$$\forall z \in A,\ x \leq z \to y \leq z \to j \leq z.$$

Then

$$m \sim_C x \wedge y, \qquad j \sim_C x \vee y.$$

Proof. The certified meet fields give

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y.$$

Applying the assumed greatestness clause for m with $z = x \wedge y$ gives

$$x \wedge y \leq m.$$

The two assumed lower-bound clauses for m, together with the certified meet greatestness field applied with $z = m$, give

$$m \leq x \wedge y.$$

The antisymmetry field from Definition 53.43 applied to

$$m \leq x \wedge y \qquad \text{and} \qquad x \wedge y \leq m$$

yields

$$m \sim_C x \wedge y.$$

The certified join fields give

$$x \leq x \vee y \qquad \text{and} \qquad y \leq x \vee y.$$

Applying the assumed leastness clause for j with $z = x \vee y$ gives

$$j \leq x \vee y.$$

The two assumed upper-bound clauses for j, together with the certified join leastness field applied with $z = j$, give

$$x \vee y \leq j.$$

The antisymmetry field from Definition 53.43 applied to

$$j \leq x \vee y \qquad \text{and} \qquad x \vee y \leq j$$

yields

$$j \sim_C x \vee y.$$

□

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeSingleton bound uniqueness from directional certificates`

Definition 55.113 (Concrete unary total-order lattice selectors)**.** *For carried unary histories h, k in $\mathsf{Carrier_u}$, let the active comparison branch be the ledger branch supplied by Definition 54.9. Define the trichotomy lower and upper endpoint selectors by*

$$\mathsf{glb_{tri}}(h,k) := \begin{cases} h, & \textit{on } \mathsf{NatUnaryStrictPrefix}(h,k), \\ h, & \textit{on } \rho_{\mathsf{u}}(h,k), \\ k, & \textit{on } \mathsf{NatUnaryStrictPrefix}(k,h), \end{cases} \qquad \mathsf{lub_{tri}}(h,k) := \begin{cases} k, & \textit{on } \mathsf{NatUnaryStrictPrefix}(h,k), \\ k, & \textit{on } \rho_{\mathsf{u}}(h,k), \\ h, & \textit{on } \mathsf{NatUnaryStrictPrefix}(k,h). \end{cases}$$

The total-order-selected lattice operations are

$$h \wedge_{\mathsf{tri}} k := \mathsf{glb_{tri}}(h,k), \qquad h \vee_{\mathsf{tri}} k := \mathsf{lub_{tri}}(h,k).$$

Theorem 55.114 (Concrete unary $\mathsf{TotalOrder}^{\uparrow}$ forgets to $\mathsf{Lattice}^{\uparrow}$)**.** *The concrete* $\mathsf{TotalOrder}^{\uparrow}$ *naming certificate of Definition 54.20 determines a* $\mathsf{Lattice}^{\uparrow}$ *certificate over the same carrier* $C = \mathsf{Carrier}_{\mathsf{u}}$*, classifier* ρ_{u}*, and order* $\leq_{\mathsf{u}}$*, with meet and join given by Definition 55.113. For every carried pair* $h, k \in C$*, the meet and join classifier-equal the trichotomy endpoint selectors:*

$$\rho_{\mathsf{u}}(h \wedge_{\mathsf{tri}} k, \mathsf{glb}_{\mathsf{tri}}(h,k)) \qquad \textit{and} \qquad \rho_{\mathsf{u}}(h \vee_{\mathsf{tri}} k, \mathsf{lub}_{\mathsf{tri}}(h,k)).$$

The meet bound-characterization rows are the greatest-lower selector rows of Definition 54.12, and the join bound-characterization rows are the least-upper selector rows of Definition 54.10.

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeUnaryPrefix extrema certificate`

Proof. The source, carrier, classifier, relation, and inherited $\mathsf{POSet}^{\uparrow}$ stability rows are those already packaged in Definition 54.20. That theorem cites the concrete unary-prefix $\mathsf{POSet}^{\uparrow}$ naming certificate, so the partial-order reflexivity, transitivity, congruence, and antisymmetry fields required by the lattice interface are already present.

Fix carried unary histories h, k. The comparison partition of Definition 54.9 chooses exactly one of the three branches

$$\mathsf{NatUnaryStrictPrefix}(h,k), \qquad \rho_{\mathsf{u}}(h,k), \qquad \mathsf{NatUnaryStrictPrefix}(k,h).$$

The selector definition follows that same branch ledger: in the strict-left and same branches it chooses h as the lower endpoint and k as the upper endpoint, while in the strict-right branch it chooses k as the lower endpoint and h as the upper endpoint.

The theorem Definition 54.12 supplies, for this selected lower endpoint, a carried history l with

$$l \leq_{\mathsf{u}} h, \qquad l \leq_{\mathsf{u}} k, \qquad (r \leq_{\mathsf{u}} h \wedge r \leq_{\mathsf{u}} k) \Rightarrow r \leq_{\mathsf{u}} l$$

for every carried lower-bound witness r. By the displayed branch definition, this l is precisely $\mathsf{glb}_{\mathsf{tri}}(h,k)$, hence precisely $h \wedge_{\mathsf{tri}} k$. These are the meet lower-bound and meet greatestness rows listed in Definition 55.102.

Similarly, Definition 54.10 supplies, for the selected upper endpoint, a carried history u with

$$h \leq_{\mathsf{u}} u, \qquad k \leq_{\mathsf{u}} u, \qquad (h \leq_{\mathsf{u}} r \wedge k \leq_{\mathsf{u}} r) \Rightarrow u \leq_{\mathsf{u}} r$$

for every carried upper-bound witness r. By the displayed branch definition, this u is precisely $\mathsf{lub}_{\mathsf{tri}}(h,k)$, hence precisely $h \vee_{\mathsf{tri}} k$. These are the join upper-bound and join leastness rows listed in Definition 55.102.

It remains to account for meet and join congruence. Suppose h, h', k, k' are carried and that $\rho_{\mathsf{u}}(h,h')$ and $\rho_{\mathsf{u}}(k,k')$ hold. The branch-transport theorem Definition 54.13 preserves and reflects the strict-left, same, and strict-right branches across these endpoint replacements. If the active branch is strict-left or same, the meet outputs are h and h' and the join outputs are k and k'. If the active branch is strict-right, the meet outputs are k and k' and the join outputs are h and h'. In each case the assumed endpoint classifier witnesses give the required classifier comparison between the selected outputs. Thus meet and join are congruent in both arguments.

The classifier-equality rows in the theorem statement follow from the same selected-readback definition and classifier reflexivity on carried endpoints. The ledger row of Definition 55.103 records the selected comparison branch together with the greatest-lower and least-upper bound certificates just cited. Therefore the tuple with operations $\wedge_{\mathsf{tri}}$ and $\vee_{\mathsf{tri}}$ satisfies the $\mathsf{Lattice}^{\uparrow}$ certificate obligations of Definition 55.104. □

Definition 55.115 (Opposite lattice certificate data)**.** *Let a* $\mathsf{Lattice}^{\uparrow}$ *certificate be presented by a carrier predicate* $\mathcal{C}$*, classifier* $\sim_C$*, order* $\leq_C$*, meet* $\wedge_C$*, and join* $\vee_C$*. Define*

$$a \leq_C^{\mathrm{op}} b := b \leq_C a, \qquad a \wedge_C^{\mathrm{op}} b := a \vee_C b, \qquad a \vee_C^{\mathrm{op}} b := a \wedge_C b.$$

The opposite lattice data C^{op} *have the same carrier predicate* $\mathcal{C}$*, the same classifier* $\sim_C$*, the opposite order* $\leq_C^{\mathrm{op}}$*, meet* $\wedge_C^{\mathrm{op}}$*, and join* $\vee_C^{\mathrm{op}}$*.*

Theorem 55.116 (Opposite lattice certificate)**.** *If* $(\mathcal{C}, \sim_C, \leq_C, \wedge_C, \vee_C)$ *satisfies the* $\mathsf{Lattice}^{\uparrow}$ *certificate obligations of Definition 55.104, then the data of Definition 55.115 also satisfy the* $\mathsf{Lattice}^{\uparrow}$ *certificate obligations.*

Proof. Unpack the lattice obligations through Definition 55.104. By Definition 55.60 and Definition 55.99, the given lattice certificate contains a certified $\mathsf{POSet}^{\uparrow}$ carrier with carrier predicate $\mathcal{C}$, classifier $\sim_C$, and order $\leq_C$. Applying Definition 53.50 supplies the carrier, source, pattern, classifier, stability, and ledger rows for the same carrier and classifier with order $\leq_C^{\mathrm{op}}$.

The binary operation rows are present with names exchanged. The opposite meet $\wedge_C^{\mathrm{op}}$ is the original join $\vee_C$, and the opposite join $\vee_C^{\mathrm{op}}$ is the original meet $\wedge_C$. Thus the carried-operation data required by the lattice source row are copied from the original join and meet data.

It remains to verify the four bound-characterization rows in Definition 55.102. Fix carried endpoints a, b, z. For the opposite meet, the lower-bound laws are exactly the original join upper-bound laws:

$$a \wedge_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} a \iff a \leq_C a \vee_C b, \qquad a \wedge_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} b \iff b \leq_C a \vee_C b.$$

For opposite meet greatestness, assume $z \leq_C^{\mathrm{op}} a$ and $z \leq_C^{\mathrm{op}} b$. These unfold to $a \leq_C z$ and $b \leq_C z$. The original join leastness row gives $a \vee_C b \leq_C z$, which is exactly

$$z \leq_C^{\mathrm{op}} a \wedge_C^{\mathrm{op}} b.$$

For the opposite join, the upper-bound laws are exactly the original meet lower-bound laws:

$$a \leq_C^{\mathrm{op}} a \vee_C^{\mathrm{op}} b \iff a \wedge_C b \leq_C a, \qquad b \leq_C^{\mathrm{op}} a \vee_C^{\mathrm{op}} b \iff a \wedge_C b \leq_C b.$$

For opposite join leastness, assume $a \leq_C^{\mathrm{op}} z$ and $b \leq_C^{\mathrm{op}} z$. These unfold to $z \leq_C a$ and $z \leq_C b$. The original meet greatestness row gives $z \leq_C a \wedge_C b$, which is exactly

$$a \vee_C^{\mathrm{op}} b \leq_C^{\mathrm{op}} z.$$

Meet and join congruence are also exchanged. If $a \sim_C a'$ and $b \sim_C b'$, then original join congruence gives

$$a \wedge_C^{\mathrm{op}} b = a \vee_C b \sim_C a' \vee_C b' = a' \wedge_C^{\mathrm{op}} b',$$

and original meet congruence gives

$$a \vee_C^{\mathrm{op}} b = a \wedge_C b \sim_C a' \wedge_C b' = a' \vee_C^{\mathrm{op}} b'.$$

These are precisely the opposite meet-congruence and opposite join-congruence rows of Definition 55.101.

The pattern row of Definition 55.100 is unchanged expression formation by two binary lattice operations; each opposite meet node is read as an original join node, and each opposite join node is read as an original meet node. The ledger row of Definition 55.103 is the same exchange: an

opposite meet ledger records the original join bound-characterization certificate, and an opposite join ledger records the original meet bound-characterization certificate, with the order witnesses interpreted through $\leq_C^{\mathrm{op}}$. Together with the opposite $\mathsf{POSet}^{\uparrow}$ rows supplied above, these are all carrier, source, pattern, classifier, stability, and ledger obligations required by Definition 55.104. Hence C^{op} carries a $\mathsf{Lattice}^{\uparrow}$ certificate. □

Theorem 55.117 (Double-opposite lattice certificate data identity)**.** *Let E be a $\mathsf{Lattice}^{\uparrow}$ certificate satisfying the obligations of Definition 55.104. Present its lattice data by a carrier predicate $\mathcal{C}$, classifier $\sim_C$, order relation $\leq_C$, meet $\wedge_C$, join $\vee_C$, meet ledger row $\mathcal{L}_C^{\wedge}$, and join ledger row $\mathcal{L}_C^{\vee}$ as in Definition 55.103. Let $E^{\mathrm{opop}} = (E^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 55.115 twice. Then E^{opop} is a $\mathsf{Lattice}^{\uparrow}$ certificate, and its carrier predicate, classifier, order relation, meet, join, and lattice ledger rows are componentwise identical to those of E: for carried endpoints a, b,*

$$\begin{aligned} \mathcal{C}^{\mathrm{opop}}(a) &= \mathcal{C}(a), & a \sim_C^{\mathrm{opop}} b &= a \sim_C b, \\ a \leq_C^{\mathrm{opop}} b &= a \leq_C b, & a \wedge_C^{\mathrm{opop}} b &= a \wedge_C b, \\ a \vee_C^{\mathrm{opop}} b &= a \vee_C b, & \mathcal{L}_C^{\wedge,\mathrm{opop}}(a,b) &= \mathcal{L}_C^{\wedge}(a,b), \\ \mathcal{L}_C^{\vee,\mathrm{opop}}(a,b) &= \mathcal{L}_C^{\vee}(a,b). \end{aligned}$$

Proof. Because E satisfies the $\mathsf{Lattice}^{\uparrow}$ obligations, Definition 55.116 makes E^{op} a certified $\mathsf{Lattice}^{\uparrow}$ instance. Applying the same theorem to that certified opposite data makes E^{opop} a certified $\mathsf{Lattice}^{\uparrow}$ instance.

By Definition 55.60 and Definition 55.99, the inherited order part of E is a certified $\mathsf{POSet}^{\uparrow}$ carrier with carrier predicate $\mathcal{C}$, classifier $\sim_C$, and order $\leq_C$. Applying Definition 53.51 to this inherited part gives the carrier, classifier, order, and inherited antisymmetry-ledger identities after two opposites. Hence

$$\mathcal{C}^{\mathrm{opop}}(a) = \mathcal{C}(a), \qquad a \sim_C^{\mathrm{opop}} b = a \sim_C b, \qquad a \leq_C^{\mathrm{opop}} b = a \leq_C b.$$

For the lattice operations, two unfoldings of Definition 55.115 give

$$a \wedge_C^{\mathrm{opop}} b = a \vee_C^{\mathrm{op}} b = a \wedge_C b$$

and

$$a \vee_C^{\mathrm{opop}} b = a \wedge_C^{\mathrm{op}} b = a \vee_C b.$$

Thus the double-opposite meet and join data are exactly the original meet and join data.

The source row is a certified poset carrier plus meet and join operations, so the restored poset part together with the restored operation slots gives the original source row. The pattern row of Definition 55.100 is expression formation by meet and join over carrier witnesses; one opposite exchanges the two operation names and the second exchange restores them.

The stability rows of Definition 55.102 are restored in the same componentwise way. In one opposite, the meet lower-bound and greatestness rows are the original join upper-bound and leastness rows read through the reversed order, while the join rows are the original meet rows read through the reversed order. The second opposite reverses the order again and exchanges the operation slots again, so the meet lower-bound, meet greatestness, join upper-bound, and join leastness rows are the original rows. Meet congruence and join congruence from Definition 55.101 are also exchanged twice and hence restored.

Finally, Definition 55.103 records the operands and the bound-characterization certificate for each meet or join. A single opposite reads a meet ledger as the corresponding original join ledger

and a join ledger as the corresponding original meet ledger, with order witnesses interpreted through $\leq_C^{\mathrm{op}}$. The second opposite performs the same exchange again and the order has already returned to $\leq_C$. Therefore

$$\mathcal{L}_C^{\wedge,\mathrm{opop}}(a,b) = \mathcal{L}_C^{\wedge}(a,b), \qquad \mathcal{L}_C^{\vee,\mathrm{opop}}(a,b) = \mathcal{L}_C^{\vee}(a,b).$$

Together these identities identify the double-opposite lattice certificate data with the original lattice certificate data. □

Theorem 55.118 (Lattice one-sided distributive inequality from directional bounds)**.** *Let A be a* $\mathsf{Lattice}^{\uparrow}$ *carrier with classifier $\sim_C$. Assume the inherited preorder transitivity field and the directional meet and join bound-characterization fields from Definition 55.102. Then for all $x, y, z \in A$,*

$$(x \wedge y) \vee (x \wedge z) \leq x \wedge (y \vee z).$$

Lean4 ✓ `BEDC.Derived.LatticeUp.LatticeDirected one sided distributive inequality from bounds`

Proof. Let

$$q := x \wedge (y \vee z).$$

The join leastness field will give the desired comparison once both join inputs are shown to be below q.

For the first input, the meet lower-left field gives

$$x \wedge y \leq x.$$

The meet lower-right field gives

$$x \wedge y \leq y,$$

and the join upper-left field gives

$$y \leq y \vee z.$$

By inherited preorder transitivity from Definition 52.44, these two comparisons compose to

$$x \wedge y \leq y \vee z.$$

Applying meet greatestness to the two comparisons

$$x \wedge y \leq x \qquad \text{and} \qquad x \wedge y \leq y \vee z$$

gives

$$x \wedge y \leq x \wedge (y \vee z),$$

that is,

$$x \wedge y \leq q.$$

For the second input, the meet lower-left field gives

$$x \wedge z \leq x.$$

The meet lower-right field gives

$$x \wedge z \leq z,$$

and the join upper-right field gives

$$z \leq y \vee z.$$

By inherited preorder transitivity, these two comparisons compose to

$$x \wedge z \leq y \vee z.$$

Applying meet greatestness to the two comparisons

$$x \wedge z \leq x \qquad \text{and} \qquad x \wedge z \leq y \vee z$$

gives

$$x \wedge z \leq x \wedge (y \vee z),$$

that is,

$$x \wedge z \leq q.$$

The join leastness field applied with upper bound q to

$$x \wedge y \leq q \qquad \text{and} \qquad x \wedge z \leq q$$

yields

$$(x \wedge y) \vee (x \wedge z) \leq q.$$

Substituting the definition of q gives

$$(x \wedge y) \vee (x \wedge z) \leq x \wedge (y \vee z).$$

□

Chapter 56

A Concrete Naming Certificate for $\mathsf{Interval}^{\uparrow}$

The unit interval interface $\mathsf{Interval}^{\uparrow}$ is licensed over Bishop reals from chapter 38 as the predicate "$0 \leq x$ and $x \leq 1$" interpreted constructively (with explicit moduli for the inequalities). The classifier is real sameness restricted to interval witnesses. It is not a primitive geometric object; it is licensed only as a derived sub-carrier of $\mathsf{Real}^{\uparrow}$.

Remark 56.1 (Interval certificate scope)**.** *This chapter records the interval licensing target as a bounded subcarrier with classifier transport inherited from internal history sameness.*

56.1 Carrier definition

Definition 56.2 (Interval carrier)**.** *Use real witnesses $x : \mathsf{Real}^{\uparrow}$ paired with constructive proofs $0 \leq x$ and $x \leq 1$.*

Theorem 56.3 (Interval empty boundary)**.** *If a history h is internally the same as e, then h inhabits the interval carrier whose lower boundary is $\mathsf{hsame}(\mathsf{e}, -)$ and whose upper boundary is $\mathsf{hsame}(-, \mathsf{e})$.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalCarrier empty boundary of hsame`

Proof. Transport the unary empty history across the symmetry of the given $\mathsf{hsame}(h, \mathsf{e})$ witness. The lower boundary uses the symmetric witness, and the upper boundary uses the original witness. □

Theorem 56.4 (Interval empty-boundary carrier iff endpoint)**.** *For the interval carrier whose lower boundary is $\mathsf{hsame}(\mathsf{e}, -)$ and whose upper boundary is $\mathsf{hsame}(-, \mathsf{e})$, carrier membership of h is equivalent to the endpoint witness $\mathsf{hsame}(h, \mathsf{e})$.*

Proof. In the forward direction, eliminate the concrete interval-carrier conjunction and project the upper-boundary witness. In the reverse direction, apply the empty-boundary carrier constructor to the endpoint witness. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalCarrier empty boundary iff hsame empty`

Theorem 56.5 (Interval empty boundary hsame transport)**.** *For the interval carrier whose lower boundary is $\mathsf{hsame}(\mathsf{e}, -)$ and whose upper boundary is $\mathsf{hsame}(-, \mathsf{e})$, internal sameness transports carrier membership.*

Proof. Unpack the interval carrier into unary membership and the two boundary witnesses. Unary membership transports along the given hsame witness. The lower boundary composes $\mathsf{hsame}(\mathsf{e}, h)$ with $\mathsf{hsame}(h, k)$, and the upper boundary composes the symmetric witness $\mathsf{hsame}(k, h)$ with $\mathsf{hsame}(h, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalCarrier empty boundary hsame transport`

Definition 56.6 (Interval empty-boundary ledger policy)**.** *The empty-boundary interval ledger records a raw endpoint in the concrete carrier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*, together with an internal history-sameness link from the raw endpoint to the visible endpoint.*

Lean4 (definition) `BEDC.Derived.IntervalUp.IntervalEmptyBoundaryLedgerPolicy`

Theorem 56.7 (Interval empty-boundary ledger visible carrier)**.** *If an empty-boundary interval ledger relates a raw endpoint to a visible endpoint, then the visible endpoint remains in the same concrete empty-boundary interval carrier.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalEmptyBoundaryLedgerPolicy visible carrier`

Proof. Unpack the ledger into raw carrier evidence and the raw-visible hsame link. The empty-boundary carrier transport theorem carries the raw carrier evidence across that link to the visible endpoint. □

Theorem 56.8 (Interval empty-boundary ledger visible no zero endpoint)**.** *No visible endpoint of an empty-boundary interval ledger can have the form* $\mathsf{e0}(t)$*.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalEmptyBoundaryLedgerPolicy visible no e0`

Proof. The visible-carrier theorem exposes the transported empty-boundary carrier at $\mathsf{e0}(t)$. Its upper boundary is an $\mathsf{hsame}(\mathsf{e0}(t), \mathsf{e})$ witness, contradicting BEDC history no-confusion. □

Theorem 56.9 (Interval empty-boundary ledger raw-visible classifier)**.** *If an empty-boundary interval ledger relates a raw endpoint to a visible endpoint, then those endpoints satisfy the restricted empty-boundary interval classifier.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalEmptyBoundaryLedgerPolicy raw visible classifier`

Proof. Unpack the ledger into raw carrier evidence and the raw-visible hsame link. The visible carrier theorem supplies the visible endpoint carrier, and the two carrier witnesses with the stored link rebuild the interval classifier. □

Theorem 56.10 (Interval empty-boundary ledger classifier endpoint equivalence)**.** *If* IntervalEmptyBoundaryLedger *holds, then every target history w is classified from ρ exactly when it is classified from v by the empty-boundary interval classifier:*

$$\mathsf{IntervalClassifierSpec}_{\mathsf{e}}(\rho, w) \leftrightarrow \mathsf{IntervalClassifierSpec}_{\mathsf{e}}(v, w).$$

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalEmptyBoundaryLedgerPolicy classifier endpoint equivalence`

Proof. The ledger supplies the raw-visible empty-boundary interval classifier. For the raw-to-visible direction, reverse that classifier by symmetry and compose it with the raw-target classifier by transitivity. For the visible-to-raw direction, compose the raw-visible classifier with the visible-target classifier. □

Theorem 56.11 (Interval empty boundary carrier uniqueness)**.** *For the interval carrier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*, any two carrier inhabitants are internally the same history.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalCarrier empty boundary unique`

Proof. Unpack the two carrier witnesses and read their upper-bound witnesses $\mathsf{hsame}(h, \mathsf{e})$ and $\mathsf{hsame}(k, \mathsf{e})$. Compose the first with the symmetry of the second to obtain $\mathsf{hsame}(h, k)$. □

Theorem 56.12 (Interval empty-boundary classifier)**.** *If h and k are both internally the same as* e*, then they satisfy the interval classifier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec empty boundary of hsame`

Proof. Apply the empty-boundary carrier theorem to h and to k. The classifier relation is $\mathsf{hsame}(h, k)$, obtained by composing $\mathsf{hsame}(h, \mathsf{e})$ with the symmetry of $\mathsf{hsame}(k, \mathsf{e})$. □

Theorem 56.13 (Interval empty-boundary classifier characterization)**.** *For the interval classifier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*, a pair h, k is classified exactly when both endpoint histories are internally the same as* e*.*

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec empty boundary iff`

Proof. In the forward direction, eliminate the restricted classifier witness and read the two upper-bound witnesses from the carrier data. In the reverse direction, apply the empty-boundary classifier theorem to the two endpoint witnesses. □

Theorem 56.14 (Interval empty-boundary classifier iff endpoints)**.** *For the interval classifier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*, a classifier witness between h and k is equivalent to endpoint witnesses* $\mathsf{hsame}(h, \mathsf{e})$ *and* $\mathsf{hsame}(k, \mathsf{e})$*.*

Proof. In one direction, eliminate the restricted classifier witness and project the upper-boundary fields from the two interval carriers. In the other direction, apply the empty-boundary classifier constructor to the two endpoint witnesses. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec empty boundary iff endpoints`

Theorem 56.15 (Interval empty boundary excludes left zero-extension)**.** *For the interval classifier whose lower boundary is* $\mathsf{hsame}(\mathsf{e}, -)$ *and whose upper boundary is* $\mathsf{hsame}(-, \mathsf{e})$*, no classifier witness can have left endpoint* $\mathsf{E0}(t)$*.*

Proof. Eliminate the classifier witness and read the upper-boundary field of the left carrier. It gives $\mathsf{hsame}(\mathsf{E0}(t), \mathsf{e})$, contradicting BEDC history no-confusion for zero-extension versus the empty history. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec empty boundary no e0 left`

56.2 The certificate

Definition 56.16 (Interval source specification)**.** *The source is the Bishop real source restricted to the closed unit interval predicate.*

Definition 56.17 (Interval pattern specification)**.** *The pattern is subcarrier formation by attaching lower and upper bound witnesses to a real.*

Definition 56.18 (Interval classifier specification)**.** *The classifier is* $\mathsf{Real}^{\uparrow}$ *sameness restricted to interval witnesses.*

Theorem 56.19 (Interval classifier transitivity)**.** *If interval classifier witnesses relate h to k and k to r, then they determine an interval classifier witness from h to r.*

Proof. Eliminate both restricted classifier witnesses. The first left carrier and the second right carrier become the endpoint data for the composite classifier, and the underlying history link is the transitive composite of the two hsame witnesses. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec trans`

Theorem 56.20 (Interval classifier symmetry)**.** *If interval classifier witnesses relate h to k, then they determine an interval classifier witness from k to h.*

Proof. Eliminate the restricted classifier witness. The endpoint interval-carrier witnesses are exchanged, and the stored history-sameness link is reversed by symmetry of hsame. □

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec symm`

Definition 56.21 (Interval stability certificate)**.** *The stability certificate consists of:*

1. *real classifier reflexivity and transitivity*
2. *lower-bound preservation under real sameness*
3. *upper-bound preservation under real sameness*
4. *closure of endpoints* 0 *and* 1
5. *closure of interval operations explicitly proved to preserve bounds*

Definition 56.22 (Interval ledger policy)**.** *Record the real ledger and both inequality moduli.*

Definition 56.23 ($\mathsf{Interval}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Interval}^{\uparrow}$*.*

Theorem 56.24 (Concrete $\mathsf{Interval}^{\uparrow}$ naming certificate)**.** *The bounded-real subcarrier tuple forms a valid NameCert for* $\mathsf{Interval}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntervalUp.interval name certificate`

Primitive boundary. $\mathsf{Interval}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 56.25 (Zero-axiom encoding note)**.** *Uses constructive inequality witnesses with moduli rather than classical order trichotomy.*

Closure status for $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$**.**

Constructive story: An interval endpoint is admitted by taking an existing real-style history surface and attaching lower and upper boundary predicates; in the checked empty-boundary instance, both predicates force the same endpoint, so the carried histories are exactly those history-same to e. The classifier is real or history sameness restricted to histories that satisfy the two boundary witnesses, and in the empty-boundary specialization it is equivalent to both endpoints being classified with e. The source is the bounded-real subcarrier, the pattern records subcarrier formation by lower and upper bounds together with boundary weakening and no-zero-extension rows, and the ledger stores the raw endpoint, the visible endpoint, and the hsame link that transports both boundary witnesses. Because the empty endpoint inhabits the carrier, restricted hsame is reflexive, symmetric, and transitive, and carrier membership transports along history-sameness while preserving both boundaries, the five name-certificate obligations close inside the same history layer. A history h lies in the signature gap of an $\mathsf{Interval}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer endpoint and boundary ledger, and token introduction names that answer as p; unfolded here, h is a bounded endpoint whose lower and upper witnesses match the token, and in the empty-boundary row this collapses to $h$hsamee. Boundary weakening maps may move from a stronger interval predicate to a weaker one by explicit pointwise maps, but no standard bridge to external real intervals is claimed. The chapter registers this bounded-subcarrier construction under the name $\mathsf{Interval}^{\uparrow}$; the interval is not a primitive geometric object, and interval arithmetic, compactness, completeness, and Heine–Borel data are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped interval certificate binds the bounded-real subcarrier, lower and upper boundary predicates, restricted interval classifier, empty-boundary ledger policy, boundary weakening maps, BHist, and hsame transport.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.IntervalUp.interval\_name\_certificate`

Bridge status: paperBridge

Not claimed: General interval arithmetic, compactness of $[0, 1]$, metric completeness, Heine-Borel, and any standard bridge to external real intervals are not closed here.

Upgrade path: No further paper theory-axis upgrade is recorded here; bridge verification remains a separate formal-axis matter.

Theorem 56.26 (Interval carrier boundary weakening)**.** *Let* $\ell_1, \ell_2, u_1, u_2 : \mathsf{BHist} \to \mathsf{Prop}$ *be interval boundary predicates. Suppose that for every history* $h : \mathsf{BHist}$ *there are pointwise boundary maps*

$$\lambda_h : \ell_2(h) \to \ell_1(h), \qquad \upsilon_h : u_2(h) \to u_1(h).$$

Then every inhabitant of the inner interval carrier is an inhabitant of the outer interval carrier:

$$\mathsf{IntervalCarrier}_{\ell_2,u_2}(h) \to \mathsf{IntervalCarrier}_{\ell_1,u_1}(h).$$

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalCarrier boundary weakening`

Proof. By Definition 56.2, an inner carrier witness consists of unary membership for h, a lower-bound witness in $\ell_2(h)$, and an upper-bound witness in $u_2(h)$. Keep the unary membership witness unchanged. Apply the pointwise lower-bound map λ_h to the lower witness, and apply the pointwise upper-bound map υ_h to the upper witness. These three witnesses reconstruct $\mathsf{IntervalCarrier}_{\ell_1,u_1}(h)$ by Definition 56.2. □

Theorem 56.27 (Interval classifier boundary weakening)**.** *Let* $\ell_1, \ell_2, u_1, u_2 : \mathsf{BHist} \to \mathsf{Prop}$ *be interval boundary predicates. Suppose that for every history* $x : \mathsf{BHist}$ *there are pointwise boundary maps*

$$\lambda_x : \ell_2(x) \to \ell_1(x), \qquad \upsilon_x : u_2(x) \to u_1(x).$$

Then for all histories $h, k : \mathsf{BHist}$,

$$\mathsf{IntervalClassifierSpec}_{\ell_2,u_2}(h,k) \to \mathsf{IntervalClassifierSpec}_{\ell_1,u_1}(h,k).$$

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec boundary weakening`

Proof. Eliminate the inner classifier witness using Definition 56.18. It consists of an $\mathsf{IntervalCarrier}_{\ell_2,u_2}(h)$ witness, an $\mathsf{IntervalCarrier}_{\ell_2,u_2}(k)$ witness, and the stored $\mathsf{hsame}(h,k)$ link. Apply Definition 56.26 at h to obtain $\mathsf{IntervalCarrier}_{\ell_1,u_1}(h)$, and apply the same theorem at k to obtain $\mathsf{IntervalCarrier}_{\ell_1,u_1}(k)$. Rebuild the restricted classifier from these two carrier witnesses and the unchanged stored $\mathsf{hsame}(h,k)$ witness by Definition 56.18. □

Proposition 56.28 (Interval nesting requires boundary eliminators)**.** *There is a finite one-history interpretation of the interval carrier and classifier clauses in which bare symbolic lower- and upper-nesting assumptions hold, the inner interval classifier holds at the unique endpoint, and the outer interval classifier fails at the pair of unique endpoints. Hence the classifier weakening*

$$\mathsf{IntervalClassifierSpec}_{\ell_{\mathrm{in}},u_{\mathrm{in}}}(h,k) \to \mathsf{IntervalClassifierSpec}_{\ell_{\mathrm{out}},u_{\mathrm{out}}}(h,k)$$

is not forced by symbolic nesting data unless that data eliminates to pointwise boundary maps

$$\forall x : \mathsf{BHist},\ \ell_{\mathrm{in}}(x) \to \ell_{\mathrm{out}}(x) \qquad \textit{and} \qquad \forall x : \mathsf{BHist},\ u_{\mathrm{in}}(x) \to u_{\mathrm{out}}(x),$$

as required by Definition 56.27. In the naming-certificate template of Definition 9.6, a classifier-facing nesting field must therefore supply those maps, or data from which those maps are obtained.

Lean4 ✓ `BEDC.Derived.IntervalUp.IntervalClassifierSpec boundary eliminator counterexample`

Proof. Interpret BHist by the singleton set $H = \{\star\}$. Let the unary-history predicate hold at $\star$, and let $\mathsf{hsame}(\star,\star)$ hold. Define the inner and outer boundaries by

$$\ell_{\mathrm{in}}(\star) = \top, \qquad u_{\mathrm{in}}(\star) = \top, \qquad \ell_{\mathrm{out}}(\star) = \bot, \qquad u_{\mathrm{out}}(\star) = \bot.$$

Let the bare symbolic lower-nesting and upper-nesting assumptions be two atomic propositions σ_ℓ and σ_u, both interpreted by $\top$.

By Definition 56.2, $\mathsf{IntervalCarrier}_{\ell_{\mathrm{in}},u_{\mathrm{in}}}(\star)$ holds: its witnesses are the unary-history witness, the true inner lower-bound witness, and the true inner upper-bound witness. Together with $\mathsf{hsame}(\star,\star)$, Definition 56.18 gives

$$\mathsf{IntervalClassifierSpec}_{\ell_{\mathrm{in}},u_{\mathrm{in}}}(\star,\star).$$

If the outer classifier held, Definition 56.18 would expose an outer carrier witness at the left endpoint. Eliminating that carrier by Definition 56.2 would give a witness of $\ell_{\mathrm{out}}(\star)$ and a witness of $u_{\mathrm{out}}(\star)$. These are witnesses of $\bot$, so the outer classifier fails.

A lower-bound eliminator in this interpretation is exactly a map $\top \to \bot$, obtained by evaluating $\forall x : H,\ \ell_{\mathrm{in}}(x) \to \ell_{\mathrm{out}}(x)$ at $\star$. Applying such a map to the true inner lower-bound witness would produce a contradiction. The upper-bound eliminator has the same form after evaluating $\forall x : H,\ u_{\mathrm{in}}(x) \to u_{\mathrm{out}}(x)$ at $\star$. Thus the true symbolic tokens σ_ℓ and σ_u do not determine carrier inclusion; only boundary eliminators, or structure that yields them, can support interval classifier weakening. □

The child file `parts/concrete_instances/metric/l10_sibling_dependency_route.tex` records the structural $\mathsf{Met}^{\uparrow}$ L10 sibling route through chapter 38, chapter 126, chapter 301, and chapter 291.

Chapter 57

A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$

57.1 Carrier definition

Definition 57.1 (MetricSpace BHist distance carrier)**.** *A* $\mathsf{Met}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(x, y, d;\ w, c, \pi, \rho, \nu)$$

where x and y are endpoint histories, w is a displayed $\mathsf{MetricDistanceWitness}(x, y, d)$*, c is the componentwise* hsame *transport surface for endpoints and result histories, π is the visible* Cont *provenance of the distance row, ρ is the* Pkg *provenance connecting the packet to the concrete source, and ν is the local* NameCert *ledger for the carrier rows. The value read as a real distance is not an independent host function $C \times C \to \mathsf{Real}^{\uparrow}$; it is the public* $\mathsf{Real}^{\uparrow}$ *distance component exposed through the existing real-distance interface of Definition 57.311.*

The carrier exposes only the displayed finite rows. It contains no ambient host metric, quotient distance, completion operator, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricTasteGate single carrier alignment`

57.2 The certificate

Definition 57.2 (MetricSpace source specification)**.** *The source is the finite endpoint packet (x, y) together with its displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

Lean4 (definition) `BEDC.Derived.MetricUp.MetricSourceSpecification`

Definition 57.3 (MetricSpace pattern specification)**.** *The pattern is metric judgment formation from endpoint histories to displayed* $\mathsf{MetricDistanceWitness}(x, y, d)$ *rows, with the public* $\mathsf{Real}^{\uparrow}$ *distance read only through the real-distance interface.*

Lean4 (definition) `BEDC.Derived.MetricUp.MetricPatternSpecification`

Definition 57.4 (MetricSpace classifier specification)**.** *The classifier is componentwise* hsame *transport on the two endpoint histories and the result history. Distance-witness rows are stable under this transport because the transported packet carries the same* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 (definition) `BEDC.Derived.MetricUp.MetricClassifierSpecification`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricClassifierSpecification component transport`

Definition 57.5 (MetricSpace stability certificate)**.** *The stability certificate consists of:*

1. *carrier admission for finite endpoint histories and a displayed* MetricDistanceWitness *row;*
2. *componentwise classifier transport for endpoints, result histories, and the visible distance witness;*
3. *symmetry and triangle visibility through the checked* MetricDistanceWitness *rows;*
4. *positive-distance handoff through the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *interface;*
5. *ledger exactness excluding host metrics, quotient distances, completion data, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricStabilityCertificate packet rows`

Theorem 57.6 (MetricSpace BHist distance NameCert obligations)**.** *For every* $\mathsf{Met}^{\uparrow}$ *packet of Definition 57.1, the local* NameCert *ledger is accepted exactly when it supplies the following five rows:*

1. *carrier admission for the finite* BHist *endpoint packet* $(x, y, d;\ w, c, \pi, \rho, \nu)$*;*
2. *distance-witness classifier transport along componentwise* hsame*;*
3. *symmetry and triangle visibility by the existing* MetricDistanceWitness *symmetry, visible-context, and triangle boundary rows;*
4. *positive-distance handoff through the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *public interface;*
5. *ledger exactness stating that the accepted packet has no host metric, quotient distance, completion coordinate, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Conversely, these five rows repack as the $\mathsf{Met}^{\uparrow}$ *carrier certificate and expose the real-distance value only through the public interface of Definition 57.311.*

Proof. For carrier admission, read the source as the finite endpoint packet of Definition 57.1. Its displayed MetricDistanceWitness row and the local NameCert ledger provide the concrete carrier row, while Pkg records the source provenance.

For classifier transport, use the componentwise hsame rows on the two endpoints and the result history. The visible-context transport theorems for MetricDistanceWitness keep the same Cont and provenance rows, so the transported distance witness remains in the packet.

For symmetry and triangle visibility, apply Definition 57.9, Definition 57.13, and Definition 57.134. These rows expose only finite histories and visible continuation ledgers.

For the positive-distance side, use Definition 57.324 and the public real-distance carrier interface. The handoff reads the same endpoint histories, transport rows, Cont routes, Pkg provenance, and local NameCert rows.

The exactness row is then a direct inventory comparison. Every accepted field has already been listed among BHist, hsame, Cont, Pkg, NameCert, MetricDistanceWitness, and the public $\mathsf{Real}^{\uparrow}$-distance interface, so there is no coordinate through which a host metric, quotient distance, completion object, or $\mathsf{Real}^{\uparrow}$ completeness principle can enter. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricBHistDistance namecert obligations`

The generic MetricDistanceWitness surface is consumed by the $\mathsf{RealMetric}^{\uparrow}$ concrete distance packet in chapter 1638. There the $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows supply the displayed distance value before the $\mathsf{Met}^{\uparrow}$ laws are read.

Theorem 57.7 (Metric empty distance witness exactness)**.** *For histories x, y, a concrete distance witness with empty distance ledger is equivalent to both endpoint histories being empty up to the history classifier:*

$$\mathsf{MetricDistanceWitness}(x, y, \mathsf{e}) \iff \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. In the forward direction, unpack the concrete distance witness. Its continuation field is $\mathsf{Cont}(x, y, \mathsf{e})$, and empty-result inversion for continuation forces both endpoint histories to be empty.

Conversely, case analysis on the two endpoint sameness witnesses reduces the claim to the empty history. The empty endpoint, empty target, and empty distance are unary, and the continuation ledger is the reflexive empty concatenation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty distance iff`

Theorem 57.8 (Metric distance witness depth additivity)**.** *Let* MetricDistanceDepth *be the structural depth of a BEDC history, counting each visible constructor above* e*. For histories x, y, d, every concrete metric distance witness carries exactly the append-depth ledger:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \implies \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x) + \mathsf{MetricDistanceDepth}(y).$$

Proof. Unpack the concrete witness. Its continuation field is precisely the BEDC append ledger $d = x \cdot y$. The structural-depth equation follows by induction on the right history y, with the empty case reflexive and both visible constructor cases obtained by applying successor congruence to the induction hypothesis. □

Lean4 (definition) `BEDC.Derived.MetricUp.MetricDistanceDepth`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness depth add`

Theorem 57.9 (Metric distance witness symmetric classifier)**.** *For unary endpoint histories x, y, if d_{xy} is witnessed by the continuation ledger from x to y and d_{yx} is witnessed by the reversed ledger from y to x, then the two distance histories classify together by* hsame*:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \implies \mathsf{MetricDistanceWitness}(y, x, d_{yx}) \implies \mathsf{hsame}(d_{xy}, d_{yx}).$$

Proof. Unpack the two concrete distance witnesses. The endpoint histories are unary in both directions, and the two displayed continuations are precisely the forward and reversed unary continuation ledgers. Unary continuation commutativity gives $\mathsf{hsame}(d_{xy}, d_{yx})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness symmetric classifier`

Theorem 57.10 (Metric distance witness prefix closure)**.** *For unary histories p, x, y, d, a unary visible prefix on the first endpoint and the distance ledger preserves the concrete metric distance witness:*

$$\mathsf{UnaryHistory}(p) \implies \mathsf{MetricDistanceWitness}(x, y, d) \implies \mathsf{MetricDistanceWitness}(p \cdot x, y, p \cdot d).$$

Proof. Unpack the witness into unary endpoint and distance histories together with the continuation ledger $d = x \cdot y$. Unary append closure gives unary histories for $p \cdot x$ and $p \cdot d$. The prefixed continuation ledger is exactly the associativity equality $(p \cdot x) \cdot y = p \cdot (x \cdot y)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix closed`

Theorem 57.11 (Metric distance witness prefix iff)**.** *For histories p, x, y, d, a prefixed source and distance witness is equivalent to unary evidence for the prefix together with the unprefixed distance witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y, p \cdot d) \iff \mathsf{UnaryHistory}(p) \ \wedge \ \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. From the prefixed witness, unary append factorization recovers unary evidence for p, x, and d, and continuation prefix cancellation recovers $\mathsf{Cont}(x, y, d)$.

Conversely, unary append closure builds the prefixed source and distance histories, and continuation associativity transports the original ledger $d = x \cdot y$ to $p \cdot d = (p \cdot x) \cdot y$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix iff`

Theorem 57.12 (Metric distance witness suffix iff)**.** *For histories p, x, y, d, a common right suffix on the target and distance is equivalent to unary evidence for the suffix together with the unextended distance witness:*

$$\mathsf{MetricDistanceWitness}(x, y \cdot p, d \cdot p) \iff \mathsf{UnaryHistory}(p) \ \wedge \ \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. From the suffixed witness, unary append factorization recovers unary evidence for y, p, and d. Reassociation exposes a common right suffix p, and right append cancellation recovers the base continuation $\mathsf{Cont}(x, y, d)$.

Conversely, unary append closure builds the suffixed target and distance histories, and associativity transports the original ledger $d = x \cdot y$ to $d \cdot p = x \cdot (y \cdot p)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness suffix iff`

Theorem 57.13 (Metric distance witness visible-context exactness)**.** *For histories p, q, x, y, d, simultaneous visible ledgers on the source side and target side are exactly equivalent to unary evidence for the two ledgers together with the central distance witness:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. In the forward direction, apply Definition 57.12 with source $p \cdot x$, target y, distance $p \cdot d$, and suffix q. This recovers $\mathsf{UnaryHistory}(q)$ together with $\mathsf{MetricDistanceWitness}(p{\cdot}x, y, p{\cdot}d)$. Then apply Definition 57.11 to the remaining witness to recover $\mathsf{UnaryHistory}(p)$ and $\mathsf{MetricDistanceWitness}(x, y, d)$.

Conversely, assume unary evidence for p and q together with the central distance witness. Apply Definition 57.11 from right to left to build $\mathsf{MetricDistanceWitness}(p \cdot x, y, p \cdot d)$. Then apply Definition 57.12 from right to left with suffix q to append the visible target ledger and obtain $\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context iff`

Theorem 57.14 (Metric distance witness visible-context empty distance exactness)**.** *For histories p, q, x, y, a visible-context distance witness whose central distance ledger is empty is exactly unary evidence for the visible prefix and suffix together with empty classification of both central endpoints:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}\mathsf{e}){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Apply visible-context exactness to peel off the prefix and suffix. The central witness is then $\mathsf{MetricDistanceWitness}(x, y, \mathsf{e})$, which empty-distance exactness identifies with $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. Conversely, the two empty endpoint classifiers build the central empty-distance witness, and visible-context exactness reattaches the same prefix and suffix. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance iff`

Theorem 57.15 (Metric visible-context empty-distance continuation splice)**.** *For histories p, q, x, y, l, r, m, o, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$ and $\mathsf{Cont}(m, r, o)$ hold. Then the empty central endpoint exposed by the witness splices the continuation directly:

$$\mathsf{Cont}(l, r, o).$$

Proof. Apply visible-context empty-distance exactness to the witness. This gives $\mathsf{hsame}(x, \mathsf{e})$. Substitute the empty central endpoint into $\mathsf{Cont}(l, x, m)$, so the first continuation identifies m with l. The second continuation $\mathsf{Cont}(m, r, o)$ is therefore the desired direct continuation $\mathsf{Cont}(l, r, o)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation splice`

Theorem 57.16 (Metric visible-context empty-distance continuation splice determinacy)**.** *For histories $p, q, x, y, l, r, m, o, o'$, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$, $\mathsf{Cont}(m, r, o)$, and $\mathsf{Cont}(l, r, o')$ hold. Then the spliced result is determined by the displayed direct continuation:

$$\mathsf{hsame}(o, o').$$

Proof. The visible-context empty-distance continuation splice turns the first two continuations into $\mathsf{Cont}(l, r, o)$. BEDC continuation determinacy compares this spliced continuation with the displayed continuation $\mathsf{Cont}(l, r, o')$, giving $\mathsf{hsame}(o, o')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation splice hsame`

Theorem 57.17 (Metric visible-context empty-distance continuation result determinacy)**.** *For histories p, q, x, y, l, m, n, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$ and $\mathsf{Cont}(l, y, n)$ hold. Then the two continuation results are classified together:

$$\mathsf{hsame}(m, n).$$

Proof. Apply visible-context empty-distance exactness to the metric witness. This identifies both central endpoints x and y with e. After substituting these endpoint classifiers, the two displayed continuations have the same left source and the same empty right endpoint. BEDC continuation determinacy therefore compares their results and gives $\mathsf{hsame}(m, n)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation results hsame`

Theorem 57.18 (Metric visible-context empty-distance append unit laws)**.** *For histories* p, q, x, y, l, r*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q).$$

Then the central endpoints act as append units:

$$x \cdot r = r \quad \wedge \quad l \cdot y = l.$$

Proof. Apply visible-context empty-distance exactness to the witness. This exposes $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. Substituting these empty endpoints turns the two displayed append equations into the BEDC left and right append unit laws. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance append unit laws`

Theorem 57.19 (Metric visible-context empty-distance endpoint continuation units)**.** *For histories* $p, q, x, y, l, r, o_L, o_R$*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, o_L)$ *and* $\mathsf{Cont}(y, r, o_R)$ *hold. Then both displayed endpoint continuations reduce to unit continuations:*

$$\mathsf{hsame}(o_L, l) \wedge \mathsf{hsame}(o_R, r).$$

Proof. Visible-context empty-distance exactness exposes $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. After substituting these central endpoints, $\mathsf{Cont}(l, x, o_L)$ is a right-unit continuation and $\mathsf{Cont}(y, r, o_R)$ is a left-unit continuation. The BEDC continuation unit laws therefore give $\mathsf{hsame}(o_L, l)$ and $\mathsf{hsame}(o_R, r)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance endpoint continuation units`

Theorem 57.20 (Metric visible-context empty-distance witness splice)**.** *For histories* p, q, x, y, l, r, m, o*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and unary endpoint evidence for l *and* r *hold. If* $\mathsf{Cont}(l, x, m)$ *and* $\mathsf{Cont}(m, r, o)$ *hold, then the spliced continuation is a metric distance witness:*

$$\mathsf{MetricDistanceWitness}(l, r, o).$$

Proof. The visible-context empty-distance continuation splice supplies $\mathsf{Cont}(l, r, o)$. Together with the unary endpoint hypotheses, unary continuation closure gives the unary carrier for o. These three fields are exactly the concrete metric distance witness from l to r with output o. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance witness splice`

Theorem 57.21 (Metric visible-context empty-distance continuation witness depth package)**.** *For histories* $p, q, x, y, l, r, m, o : \mathsf{BHist}$*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

holds together with $\mathsf{UnaryHistory}(l)$*,* $\mathsf{UnaryHistory}(r)$*,* $\mathsf{Cont}(l, x, m)$*, and* $\mathsf{Cont}(m, r, o)$*. Then the direct continuation carries a metric witness whose depth is induced by the two visible endpoints:*

$$\mathsf{MetricDistanceWitness}(l, r, o) \wedge \mathsf{MetricDistanceDepth}(o) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r).$$

Proof. Apply the visible-context empty-distance continuation splice to the distance witness and the two continuation ledgers. This gives the direct continuation $\mathsf{Cont}(l, r, o)$.

The unary carriers for l and r then close the continuation under the metric distance witness constructor, so the direct continuation packages as $\mathsf{MetricDistanceWitness}(l, r, o)$.

Finally, metric witness depth additivity applied to this witness gives $\mathsf{MetricDistanceDepth}(o) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r)$. Pairing the witness with that displayed equation is the required conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation witness depth`

Theorem 57.22 (Metric visible-context empty-distance two-sided endpoint witness transport)**.** *For histories* $p, q, x, y, l, r, m, n, o : \mathsf{BHist}$*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

holds together with $\mathsf{UnaryHistory}(l)$*,* $\mathsf{UnaryHistory}(r)$*,* $\mathsf{Cont}(l, x, m)$*,* $\mathsf{Cont}(y, r, n)$*, and* $\mathsf{Cont}(l, r, o)$*. Then the two reduced endpoint continuations carry the same metric distance witness:*

$$\mathsf{MetricDistanceWitness}(m, n, o).$$

Proof. Apply Definition 57.394 to the visible-context empty-distance witness and the two endpoint continuation ledgers. This gives $\mathsf{hsame}(m, l)$ and $\mathsf{hsame}(n, r)$.

The unary endpoint hypotheses for l and r, together with the direct continuation ledger $\mathsf{Cont}(l, r, o)$, build the concrete direct metric witness $\mathsf{MetricDistanceWitness}(l, r, o)$.

Now use Definition 57.347. Transport the first endpoint along the symmetry of $\mathsf{hsame}(m, l)$, the second endpoint along the symmetry of $\mathsf{hsame}(n, r)$, and the result endpoint along reflexivity of $\mathsf{hsame}(o, o)$. The continuation ledger transports with the same three relations, so the transported fields package as $\mathsf{MetricDistanceWitness}(m, n, o)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance two sided endpoint witness transport`

The ten-field fit is: traditional target visible-context zero-distance splice; BEDC source MetricDistanceWitness plus Cont; carrier MetricDistanceWitness; classifier hsame; stability obligations the two unary carriers and the two continuation ledgers; ledger the hidden empty central distance; core theorem this package; exactness the displayed witness-plus-depth conjunction; standard bridge deferred; and status ProofTarget.

Theorem 57.23 (Metric distance witness visible-context depth additivity)**.** *For histories* p, q, x, y, d*, a visible-context distance witness determines the same central depth equation as the central witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x) + \mathsf{MetricDistanceDep}$$

Proof. Apply Definition 57.13 to recover the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$, then apply Definition 57.8. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context depth add`

Theorem 57.24 (Metric distance witness visible-context result determinacy)**.** *For histories p, q, x, y, d, d', if the same visible endpoint histories carry two visible distance ledgers with central parts d and d', then those central parts classify together:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d'){\cdot}q) \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Apply Definition 57.13 to the first visible witness. This recovers $\mathsf{MetricDistanceWitness}(x, y, d)$ together with unary evidence for the visible prefix and suffix.

Apply the same visible-context exactness theorem to the second visible witness. This recovers $\mathsf{MetricDistanceWitness}(x, y, d')$ over the same central endpoints.

Unpack the two central witnesses. Their continuation ledgers have the same source endpoint x and target endpoint y, namely $\mathsf{Cont}(x, y, d)$ and $\mathsf{Cont}(x, y, d')$. By continuation determinacy Definition 18.16, their results classify together, giving $\mathsf{hsame}(d, d')$.

The per-object template is as follows. The traditional target is visible metric distance readback. The BEDC source is BHist with Cont and UnaryHistory. The carrier is MetricDistanceWitness and the classifier is hsame. The ledger is the visible prefix and suffix context around the central endpoints. The core theorem is central-result determinacy, exactness is uniqueness of the central distance, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context result deterministic`

Theorem 57.25 (Metric distance witness visible-context prefix-independent result determinacy)**.** *For histories p, q, p', q', x, y, d, e, if two visible distance ledgers have the same central endpoints x and y but possibly different visible prefix and suffix contexts, then their central distance histories classify together:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(p'{\cdot}x, y{\cdot}q', (p'{\cdot}e){\cdot}q') \Longrightarrow \mathsf{hsame}(d, e).$$

Proof. Apply Definition 57.13 to each visible ledger. The first application recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$, while the second recovers $\mathsf{MetricDistanceWitness}(x, y, e)$, independently of the visible prefix and suffix histories. Their continuation fields share the same source and target endpoints, so continuation determinacy gives $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent result deterministic`

Theorem 57.26 (Metric distance witness endpoint-hsame result determinacy)**.** *For histories x, x', y, y', d, d', if x classifies with x' and y classifies with y', then two metric distance witnesses over those classified endpoints have classified results:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(x', y', d') \Longrightarrow \mathsf{hsame}(d,$$

Proof. Case-reduce the two endpoint hsame witnesses. The two distance witnesses then display continuation ledgers with the same source and target endpoints. Apply continuation result determinacy to the two continuation fields. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame result deterministic`

Theorem 57.27 (Metric distance witness visible-context endpoint-hsame result determinacy)**.** *For histories $p, q, x, x', y, y', d, d'$, classified central endpoints determine classified central distance results under the same visible prefix and suffix:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(p{\cdot}x', y'{\cdot}q, (p{\cdot}d$$

Proof. Apply Definition 57.13 to both visible ledgers. The first application recovers $\mathsf{MetricDistanceWitness}(x, y, d)$, and the second recovers $\mathsf{MetricDistanceWitness}(x', y', d')$.

The supplied endpoint classifications $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$ now match the hypotheses of Definition 57.26, which gives $\mathsf{hsame}(d, d')$ for the central distance histories. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context endpoint hsame result deterministic`

Theorem 57.28 (Metric distance witness visible-context source determinacy)**.** *For histories p, q, x, x', y, d, if two visible distance ledgers share the same visible target endpoint and the same contextual distance result, then their central source endpoints classify together:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(p{\cdot}x', y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(x, x').$$

Proof. Apply Definition 57.13 to both visible ledgers. The first application recovers the central continuation $\mathsf{Cont}(x, y, d)$, and the second recovers $\mathsf{Cont}(x', y, d)$. Continuation right cancellation over the common right endpoint y and common result d gives $\mathsf{hsame}(x, x')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context source deterministic`

Theorem 57.29 (Metric distance witness visible-context target determinacy)**.** *For histories p, q, x, y, y', d, if two visible distance ledgers share the same visible source endpoint and the same contextual distance result, then their central target endpoints classify together:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(p{\cdot}x, y'{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(y, y').$$

Proof. Apply Definition 57.13 to both visible ledgers. The first application recovers the central continuation $\mathsf{Cont}(x, y, d)$, and the second recovers $\mathsf{Cont}(x, y', d)$. Continuation left cancellation over the common left endpoint x and common result d gives $\mathsf{hsame}(y, y')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context target deterministic`

Theorem 57.30 (Metric distance witness continuation hsame transport)**.** *For histories x, x', y, y', d, d' : BHist, history-sameness of the two endpoints and of the distance result transports the continuation field exposed by a metric distance witness:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{Cont}(x', y', d').$$

Proof. Case-reduce the three supplied hsame witnesses. The transported endpoints and result are then definitionally the endpoints and result of the original metric distance witness. Project its continuation field $\mathsf{Cont}(x, y, d)$, which is the required transported continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness cont hsame transport`

Theorem 57.31 (Metric distance witness hsame field transport)**.** *For histories x, x', y, y', d, d' : BHist, history-sameness of the two endpoints and of the distance result transports all exposed fields of a metric distance witness:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{UnaryHistory}(x') \wedge \mathsf{UnaryHisto}$$

Proof. Case-reduce the three supplied hsame witnesses. The transported source, target, and distance histories are then definitionally the original fields. Project the three unary carrier fields and the continuation ledger from the metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame fields transport`

Theorem 57.32 (Metric distance witness hsame transport depth add)**.** *For histories* x, x', y, y', d, d' : BHist*, history-sameness of the two endpoints and of the distance result transports the continuation field and preserves the append-depth ledger:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{Cont}(x', y', d') \wedge \mathsf{MetricDistanc}$$

Proof. Case-reduce the three supplied hsame witnesses. The transported histories are then the source, target, and distance of the original metric witness. Project its continuation field and apply the metric distance depth-add theorem to the same witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame transport cont depth add`

Theorem 57.33 (Metric distance witness hsame source determinacy)**.** *For histories* x, x', y, y', d, d' : BHist*, if the target and distance ledgers classify by* hsame *and the two concrete metric distance witnesses hold,*

$$\mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(x', y', d'),$$

then the witnesses expose both continuation fields and determine the source:

$$\mathsf{Cont}(x, y, d) \wedge \mathsf{Cont}(x', y', d') \wedge \mathsf{hsame}(x, x').$$

Proof. Project the continuation fields from the two metric distance witnesses. After case-reducing the target and distance hsame witnesses, the continuations share the same right context and result. Continuation right cancellation then identifies the two sources. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame source deterministic`

Theorem 57.34 (Metric distance witness hsame target determinacy)**.** *For histories* x, x', y, y', d, d' : BHist*, if the source and distance ledgers classify by* hsame *and the two concrete metric distance witnesses hold,*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(x', y', d'),$$

then the witnesses expose both continuation fields and determine the target:

$$\mathsf{Cont}(x, y, d) \wedge \mathsf{Cont}(x', y', d') \wedge \mathsf{hsame}(y, y').$$

Proof. Project the continuation fields from the two metric distance witnesses. After case-reducing the source and distance hsame witnesses, the continuations share the same left context and result. Continuation left cancellation then identifies the two targets. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame target deterministic`

Theorem 57.35 (Metric distance witness visible-context hsame source determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, d'$: BHist*, if the visible targets and visible distances have the same central target and distance under* hsame*, then the central sources are determined:*

$$\mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot d') \cdot q') \Longrightarrow \mathsf{hsame}(x, x').$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', d')$. The supplied hsame relations on y, y' and d, d' then fit Definition 57.349, whose source component is the displayed $\mathsf{hsame}(x, x')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context hsame source deterministic`

Theorem 57.36 (Metric distance witness visible-context hsame target determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, d' : \mathsf{BHist}$*, if the visible sources and visible distances have the same central source and distance under* hsame*, then the central targets are determined:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot d') \cdot q') \Longrightarrow \mathsf{hsame}(y, y').$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', d')$. The supplied hsame relations on x, x' and d, d' then fit Definition 57.350, whose target component is the displayed $\mathsf{hsame}(y, y')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context hsame target deterministic`

Theorem 57.37 (Metric distance witness prefix-independent endpoint-hsame result determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, e : \mathsf{BHist}$*, if the central endpoints classify by* hsame *while the visible prefixes and suffixes may differ, then the exposed central distance results classify together:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot e) \cdot q') \Longrightarrow \mathsf{hsame}(d, e).$$

Proof. Apply visible-context exactness to both metric witnesses. This removes the prefix and suffix contexts and exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', e)$. The supplied endpoint classifications then fit the endpoint-hsame result determinacy theorem, giving the displayed $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent endpoint hsame result deterministic`

Theorem 57.38 (Metric distance witness visible right tail determinacy)**.** *For histories* $p, q, p', q', x, y, d, d' : \mathsf{BHist}$*, if two visible-context metric witnesses share the central source and the right endpoint* $\mathsf{E1}(y)$*, and their displayed distance results are respectively* $\mathsf{E1}(d)$ *and* $\mathsf{E1}(d')$ *inside their visible contexts, then the distance tails classify together:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, \mathsf{E1}(y) \cdot q, (p \cdot \mathsf{E1}(d)) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x, \mathsf{E1}(y) \cdot q', (p' \cdot \mathsf{E1}(d')) \cdot q') \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Visible-context exactness removes the outer prefixes and suffixes, exposing two central metric witnesses with right endpoint $\mathsf{E1}(y)$ and results $\mathsf{E1}(d)$ and $\mathsf{E1}(d')$. Right-$\mathsf{E1}$ result exactness descends both witnesses to their tails, after which continuation determinism identifies the tail distance results. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context right e1 result tail deterministic`

Theorem 57.39 (Metric distance witness visible-context semantic name certificate)**.** *For unary visible contexts p, q and unary central endpoints x, y, the contextual distance predicate $d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q)$ carries a semantic naming certificate with classifier* hsame*:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{UnaryHistory}(x) \Longrightarrow \mathsf{UnaryHistory}(y) \Longrightarrow \mathsf{SemanticNameCert}(d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q))(d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d)$$

Proof. The inhabited carrier is the central distance $x \cdot y$, lifted through visible-context exactness. The classifier is hsame, so its equivalence laws are the kernel history-sameness laws. Transport along the classifier is by case analysis on the history-sameness witness, after which the contextual distance predicate is unchanged. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context semanticNameCert`

Theorem 57.40 (Metric distance witness empty-left exactness)**.** *For histories y, d, a concrete distance witness whose left endpoint is empty is exactly unary evidence for the right endpoint together with the boundary classification of the distance as that endpoint:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, y, d) \Longleftrightarrow \mathsf{UnaryHistory}(y) \ \wedge \ \mathsf{hsame}(d, y).$$

Proof. Unpack the distance witness. The continuation ledger has empty left endpoint, so left-unit exactness gives $\mathsf{hsame}(d, y)$, while the witness already contains the unary evidence for y.

Conversely, from unary evidence for y and $\mathsf{hsame}(d, y)$, transport unary evidence from y to d and use the left-unit continuation equivalence to build the concrete distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty left iff`

Theorem 57.41 (Metric distance witness empty-source visible-distance target exactness)**.** *For histories $target, d$, a concrete distance witness from the empty source whose distance ledger is visibly one-headed identifies the target with that same one-headed ledger and exposes unary evidence for the ledger tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, target, \mathsf{E1}(d)) \Longrightarrow target = \mathsf{E1}(d) \wedge \mathsf{UnaryHistory}(d).$$

Proof. Apply empty-left exactness to obtain unary evidence for *target* and the classifier $\mathsf{hsame}(\mathsf{E1}(d), target)$. One-constructor inversion on this classifier presents *target* as $\mathsf{E1}(d)$ up to the exposed tail equality. Substitution into the unary target witness and unary one-step inversion give $\mathsf{UnaryHistory}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty source e1 distance target exactness`

Theorem 57.42 (Metric distance witness empty-target visible-distance source exactness)**.** *For histories $source, d$, a concrete distance witness into the empty target whose distance ledger is visibly one-headed identifies the source with that same one-headed ledger and exposes unary evidence for the ledger tail:*

$$\mathsf{MetricDistanceWitness}(source, \mathsf{e}, \mathsf{E1}(d)) \Longrightarrow source = \mathsf{E1}(d) \wedge \mathsf{UnaryHistory}(d).$$

Proof. Apply empty-right exactness to obtain unary evidence for *source* and the classifier $\mathsf{hsame}(\mathsf{E1}(d), source)$. One-constructor inversion on this classifier presents *source* as $\mathsf{E1}(d)$ up to the exposed tail equality. Substitution into the unary source witness and unary one-step inversion give $\mathsf{UnaryHistory}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty target e1 distance source exactness`

Theorem 57.43 (Metric nonempty distance has a nonempty endpoint)**.** *For histories* x, y, d : BHist*, if* MetricDistanceWitness(x, y, d) *holds and* d *is not history-same as* e*, then at least one endpoint is not history-same as* e*:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to ((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False})).$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint nonempty`

Proof. Case-split the two endpoint histories. A visible endpoint constructor is immediately disjoint from e, giving the required side of the disjunction. In the remaining empty/empty case, the continuation field of the metric witness identifies the distance result with e, contradicting the assumed nonempty distance. □

Theorem 57.44 (Metric nonempty distance endpoint positive depth)**.** *For histories* x, y, d : BHist*, if* MetricDistanceWitness(x, y, d) *holds and* d *is not history-same as* e*, then at least one endpoint has positive structural metric depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to (0 < \mathsf{MetricDistanceDepth}(x) \vee 0 < \mathsf{MetricDistanceDepth}(y))$$

Proof. Apply the nonempty-distance endpoint theorem to obtain a nonempty endpoint. The positive-depth classifier for metric histories converts that endpoint nonemptiness into positive structural depth on the corresponding side of the disjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint positive depth`

Theorem 57.45 (Metric positive distance e-one shape)**.** *For histories* x, y, d : BHist*, if* MetricDistanceWitness$(x, y,$ *holds and the structural depth of* d *is not zero, then* d *is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to \exists tail,\ d = \mathsf{E1}(tail) \wedge \mathsf{UnaryHistory}(tail).$$

Proof. Unpack the metric distance witness to obtain the unary carrier for the distance history d. Case analysis on d rules out e by the positive depth hypothesis and rules out an E0 head by unary-history inversion. The remaining E1 case exposes its tail, and one-step unary inversion gives the unary tail witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness positive distance e1 shape`

Theorem 57.46 (Metric nonempty distance endpoint e-one cases)**.** *For histories* x, y, d : BHist*, if* MetricDistanceWitness(x, y, d) *holds and the distance endpoint is not history-same as* e*, then at least one endpoint is visibly one-headed with a unary tail:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to \\ &\quad (\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)).\end{aligned}$$

Proof. First apply the endpoint non-emptiness theorem to the metric witness and the nonempty distance assumption. On whichever side is nonempty, use the endpoint's unary-history carrier from the metric witness and apply unary nonempty one-tail readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint e1 cases`

Theorem 57.47 (Metric empty-source nonempty distance target e-one cases)**.** *For histories* $target, dist$: BHist*, if a concrete metric distance witness starts at the empty source and its distance endpoint is not history-same as* e*, then the target endpoint is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to \exists targetTail,\ target = \mathsf{E1}(targetTail) \wedge \mathsf{UnaryHist}$$

Proof. Apply the nonempty-distance endpoint e-one cases theorem. The source-side visible branch would identify e with a one-headed history, which is impossible by constructor disjointness. The remaining branch is exactly the target e-one readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty source nonempty distance target e1 cases`

Theorem 57.48 (Metric empty-target nonempty distance source e-one cases)**.** *For histories* $source, dist$: BHist*, if a concrete metric distance witness ends at the empty target and its distance endpoint is not history-same as* e*, then the source endpoint is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(source, \mathsf{e}, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to \exists sourceTail,\ source = \mathsf{E1}(sourceTail) \wedge \mathsf{UnaryH}$$

Proof. Apply the nonempty-distance endpoint e-one cases theorem. The target-side visible branch would identify e with a one-headed history, which is impossible by constructor disjointness. The remaining branch is exactly the source e-one readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty target nonempty distance source e1 cases`

Theorem 57.49 (Metric empty-boundary nonempty distance positive endpoint depth)**.** *For histories* $source, target, dist$: BHist*, a concrete metric distance witness with exactly one empty boundary and a nonempty distance endpoint forces positive structural depth on the other endpoint:*

$$(\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to 0 < \mathsf{MetricDistanceDepth}(target)) \wedge$$
$$(\mathsf{MetricDistanceWitness}(source, \mathsf{e}, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to 0 < \mathsf{MetricDistanceDepth}(source)).$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary nonempty distance positive endpoint depth`

Proof. For the empty-source implication, apply Theorem 57.217 to the displayed witness and the nonempty distance hypothesis. The result expresses $target$ as $\mathsf{E1}(targetTail)$ with a unary tail, so the structural metric depth of $target$ is a successor and is therefore positive.

For the empty-target implication, apply Theorem 57.218 in the same way. The source endpoint is read back as $\mathsf{E1}(sourceTail)$ with a unary tail, and the one-headed shape gives positive structural metric depth for $source$. □

The ten-field fit is: traditional target paired empty-boundary nonzero-distance endpoint depth; BEDC source MetricDistanceWitness over BHist; carrier the surviving boundary endpoint; classifier hsame; stability unary one-headed readback and depth positivity; ledger the nonempty distance endpoint; core theorem Theorem 57.219; exactness positive depth on the nonempty boundary endpoint; standard bridge finite unary metric boundary readback; status ProofTarget.

Theorem 57.50 (Metric visible-boundary positive-depth endpoint readback)**.** *For histories* $p, q, source, target, dis$ BHist*, a visible-context metric distance witness with exactly one empty central boundary and positive central distance depth reads back the nonempty central endpoint as one-headed and positive-depth:*

$$(\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, target \cdot q, (p \cdot dist) \cdot q) \to (\mathsf{MetricDistanceDepth}(dist) = 0 \to \mathsf{False}) \to$$
$$\exists targetTail,\ target = \mathsf{E1}(targetTail) \wedge \mathsf{UnaryHistory}(targetTail) \wedge 0 < \mathsf{MetricDistanceDepth}(target)) \wedge$$
$$(\mathsf{MetricDistanceWitness}(p \cdot source, \mathsf{e} \cdot q, (p \cdot dist) \cdot q) \to (\mathsf{MetricDistanceDepth}(dist) = 0 \to \mathsf{False}) \to$$
$$\exists sourceTail,\ source = \mathsf{E1}(sourceTail) \wedge \mathsf{UnaryHistory}(sourceTail) \wedge 0 < \mathsf{MetricDistanceDepth}(source)).$$

Proof. For the empty central source implication, apply Definition 57.13 to peel the visible prefix p and suffix q from the displayed witness. This recovers the central witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist)$.

The positive-depth hypothesis rules out $\mathsf{MetricDistanceDepth}(dist) = 0$. By Definition 57.253, this is the same nonempty-distance input required by the empty-source e-one cases theorem. Apply Definition 57.217 to obtain $target = \mathsf{E1}(targetTail)$ with unary $targetTail$. The one-headed displayed endpoint has successor structural metric depth, so $0 < \mathsf{MetricDistanceDepth}(target)$.

For the empty central target implication, the same visible-context peeling recovers $\mathsf{MetricDistanceWitness}(source$ The same zero-depth equivalence converts positive depth of $dist$ into the nonempty-distance input, and Definition 57.218 gives $source = \mathsf{E1}(sourceTail)$ with unary $sourceTail$. The displayed one-headed source again has positive structural metric depth. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible boundary positive depth endpoint readback`

Theorem 57.51 (Metric distance witness zero-headed component impossibility)**.** *For histories $x, y, d : \mathsf{BHist}$, no concrete metric distance witness has a zero-headed source, target, or distance component:*

$$(\mathsf{MetricDistanceWitness}(\mathsf{E0}(x), y, d) \to \mathsf{False}) \wedge (\mathsf{MetricDistanceWitness}(x, \mathsf{E0}(y), d) \to \mathsf{False}) \wedge (\mathsf{MetricDistanceWit}$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e0 component absurd`

Proof. Unpack the concrete witness. Its three component fields are unary-history evidence for the source, target, and distance histories. BEDC unary histories are generated only from e by $\mathsf{E1}$, so each displayed $\mathsf{E0}$ component contradicts unary no-zero-extension. □

Theorem 57.52 (Metric empty distance visible endpoint impossibility)**.** *For histories x, y, d, tx, ty, if a metric distance witness has a result history-same as e, then neither endpoint can be history-same as a visible one-headed history:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{hsame}(d, \mathsf{e}) \to \\ &\quad (\mathsf{hsame}(x, \mathsf{E1}(tx)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y, \mathsf{E1}(ty)) \to \mathsf{False}).\end{aligned}$$

Proof. Transport the continuation result of the metric witness along the displayed distance classifier, obtaining an empty-distance witness over the same endpoints. Empty-distance exactness identifies both endpoints with e. Composing either endpoint comparison with a visible one-headed classifier contradicts the empty/e-one no-confusion theorem. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty distance visible endpoint absurd`

Theorem 57.53 (Metric empty-distance composite endpoint collapse)**.** *For histories x, y, z, d_{xy}, d_{yz}, if $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ and $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$ both have distance histories classified as e, then all three endpoints classify together:*

$$d_{xy}\mathsf{hsamee} \to d_{yz}\mathsf{hsamee} \to \mathsf{hsame}(x, y) \wedge \mathsf{hsame}(y, z) \wedge \mathsf{hsame}(x, z).$$

Proof. Transport each distance witness to the empty-distance case. Empty-distance exactness identifies the corresponding endpoint pair with e, and transitivity through e gives the three endpoint comparisons. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty distance composite endpoint collapse`

Theorem 57.54 (Metric e1-pair empty-distance impossibility)**.** *For histories x, y : BHist,*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{e}) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair empty distance absurd`

Proof. Apply empty-distance exactness to the displayed metric witness. It forces the left endpoint $\mathsf{E1}(x)$ to be history-same as e, contradicting the disjointness of visible one-headed histories from the empty history. □

Theorem 57.55 (Metric e1 endpoints force e1 distance)**.** *For histories x, y, d : BHist, a metric distance witness between two one-headed endpoints forces the distance history itself to be one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \exists d_0 : \mathsf{BHist},\ d = \mathsf{E1}(d_0) \wedge \mathsf{UnaryHistory}(d_0).$$

Proof. Case-analyze the distance history. The empty case contradicts Definition 57.224; the zero-headed case contradicts the unary carrier field of the witness. The one-headed case exposes the displayed tail, and unary inversion supplies its carrier proof. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 endpoints distance e1 shape`

Theorem 57.56 (Metric right-boundary visible context distance source)**.** *For histories p, q, x, d : BHist, a visible-context distance witness with empty central right endpoint identifies the central distance with the visible source and gives the same structural depth:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(d, x) \wedge \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x).$$

Proof. Right-boundary visible-context exactness extracts the central classifier $\mathsf{hsame}(d, x)$. Case analysis on that classifier identifies the two histories, and the metric-depth equality follows by reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right boundary visible context distance source`

Theorem 57.57 (Metric right-boundary visible context positive-depth source shape)**.** *For histories p, q, x, d : BHist, a visible-context distance witness with empty central right endpoint and positive central distance depth forces the central source to be visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \Longrightarrow \exists x_0 : \mathsf{BHist},\ x = \mathsf{E1}(x_0) \wedge \mathsf{Ur}$$

Proof. Apply right-boundary visible-context exactness. It exposes unary evidence for the central source and the classifier $\mathsf{hsame}(d, x)$. If x classified as e, transitivity would classify d as e, and the zero-depth classifier would contradict the positive-depth hypothesis. Thus x is a nonempty unary history, so unary nonempty e-one tail readback supplies the required x_0. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right boundary visible context positive depth source e1 shape`

Theorem 57.58 (Metric left-boundary visible-context distance target)**.** *For histories p, q, y, d, a visible-context distance witness whose central left endpoint is empty identifies the central distance with the target and aligns their structural depths:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(d, y) \wedge \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(y).$$

Proof. Apply left-boundary visible-context exactness to peel the visible prefix and suffix. The central empty-left witness gives $\mathsf{hsame}(d, y)$. Case analysis on that classifier reduces the displayed depth equation to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left boundary visible context distance target`

Theorem 57.59 (Metric distance witness empty-right exactness)**.** *For histories x, d, a concrete distance witness whose right endpoint is empty is exactly unary evidence for the left endpoint together with the boundary classification of the distance as that endpoint:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d) \iff \mathsf{UnaryHistory}(x) \ \wedge \ \mathsf{hsame}(d, x).$$

Proof. Unpack the distance witness. The witness contains unary evidence for x, and the continuation ledger has empty right endpoint. Right-unit exactness for continuations gives the boundary classifier $\mathsf{hsame}(d, x)$.

Conversely, assume unary evidence for x and $\mathsf{hsame}(d, x)$. Transport unary evidence from x to d along this classifier, then use the right-unit continuation equivalence to build the continuation ledger with right endpoint e. Repacking the unary data and this ledger gives $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty right iff`

Theorem 57.60 (Metric distance witness left e1 result cases)**.** *For histories x, y, d, a distance witness from a visible left endpoint to a visible result has only the empty middle case or a visible middle tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)) \vee (\exists y_1 : \mathsf{BHist}.\ y = \mathsf{E1}(y_1) \wedge \mathsf{Metric}$$

Proof. Unpack the distance witness and invert the continuation ledger with visible result $\mathsf{E1}(d)$. If the middle endpoint is empty, the continuation reduces the visible equality to $\mathsf{hsame}(x, d)$ and the witness supplies unary evidence for x. The zero-tagged middle case contradicts constructor no-confusion. In the visible middle case, strip one $\mathsf{E1}$ constructor from the continuation ledger and repack the remaining unary evidence as the tail distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result cases`

Theorem 57.61 (Metric distance witness left e1 result iff)**.** *For histories x, y, d, a distance witness from a visible left endpoint to a visible result is equivalent to the empty middle case or to a visible middle tail carrying the remaining witness:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \iff (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)) \vee (\exists y_1 : \mathsf{BHist}.\ y = \mathsf{E1}(y_1) \wedge \mathsf{Metric}$$

Proof. The forward direction is the visible-result case split. For the reverse direction, the empty middle branch rebuilds the witness from unary closure under $\mathsf{E1}$ and the right-unit continuation law. The visible middle branch adds one visible constructor back to the middle endpoint and result ledger by the one-step continuation rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result iff`

Theorem 57.62 (Metric distance witness left e1 result depth addition)**.** *For histories x, y, d, every visible-result distance witness*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d))$$

descends the metric depth identity to the visible tails:

$$\mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x) + \mathsf{MetricDistanceDepth}(y).$$

Proof. Apply metric-distance depth addition to the displayed witness. The resulting identity has a visible constructor on the source side and on the result side. Normalize the natural-number expression with successor addition, then cancel the shared successor. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result depth add`

Theorem 57.63 (Metric distance witness left e1 result empty target iff)**.** *For histories x, y, d, a metric distance witness from $\mathsf{E1}(x)$ to target y with visible result $\mathsf{E1}(d)$ identifies y with the empty history exactly when the exposed source tail is unary and agrees with the result tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow (\mathsf{hsame}(y, \mathsf{e}) \Longleftrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)).$$

Proof. Split the target shape. The empty target branch is the right-unit continuation case and reads back the unary source tail and the tail equality. The $e0$ target contradicts the unary target carrier. The visible target branch cannot also have $x\mathsf{hsame}d$: transporting the result to $\mathsf{E1}(x)$ would make $\mathsf{Cont}(\mathsf{E1}(x), \mathsf{E1}(y_1), \mathsf{E1}(x))$, so right-unit uniqueness would identify $\mathsf{E1}(y_1)$ with e, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result empty target iff`

Theorem 57.64 (Metric distance witness left e1 result hsame source)**.** *For histories x, x', y, d, a distance witness from $\mathsf{E1}(x)$ to a visible result keeps the same empty-or-visible alternative after transporting the displayed source payload along $\mathsf{hsame}(x, x')$:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x') \wedge \mathsf{hsame}(x', d)) \vee (\exists y_1 : \mathsf{BHist}.$$

Proof. Case-reduce the source hsame witness. The transported payload is then definitionally the original payload, and the established visible-result split gives the displayed empty target or visible target branch. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame source`

Theorem 57.65 (Metric distance witness left e1 result hsame target)**.** *For histories x, y, y', d, a distance witness from a visible left endpoint to a visible result keeps the same empty-or-visible target alternative after transporting the target presentation along $\mathsf{hsame}(y, y')$:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow (y' = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)) \vee (\exists y_1 : \mathsf{BHist}. y$$

Proof. Apply the visible-result case split. In the empty branch, empty-history inversion transports the displayed target to e. In the visible branch, one-constructor inversion of $\mathsf{hsame}(\mathsf{E1}(y_0), y')$ exposes $y' = \mathsf{E1}(y_1)$ and $y_0\mathsf{hsame}y_1$; unary and continuation transport carry the tail metric witness to the displayed target tail. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame target`

Theorem 57.66 (Metric distance witness left e1 result hsame result)**.** *For histories x, y, d, d', a distance witness from a visible left endpoint to a visible result keeps the same empty-or-visible target alternative after transporting the displayed result tail along* $\mathsf{hsame}(d, d')$*:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d')) \vee (\exists y_1 : \mathsf{BHist}.\ y$$

Proof. First lift the result-tail agreement through the visible constructor, obtaining $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$. The metric hsame transport field is then applied with reflexive source and target transports, so the original witness repacks as $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d'))$.

Apply Theorem 57.286 to this transported witness. The empty target branch gives $\mathsf{UnaryHistory}(x)$ and $\mathsf{hsame}(x, d')$; the visible target branch gives a tail y_1 with $y = \mathsf{E1}(y_1)$ and $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y_1, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame result`

Theorem 57.67 (Metric distance witness left e1 result tail determinism)**.** *For histories x, y, d, d', two visible-result distance witnesses with the same visible left endpoint and target determine the same result tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d')) \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Unpack the two distance witnesses only to their continuation ledgers. BEDC continuation determinacy gives $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$ for the two visible results. One-constructor inversion for $\mathsf{E1}$ then descends this classifier to $\mathsf{hsame}(d, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result tail deterministic`

Theorem 57.68 (Metric distance witness left e1 result shape)**.** *For histories x, y, d, every distance witness from* $\mathsf{E1}(x)$ *has a visible result shape:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \Big((y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, k)) \vee \exists y_1 : \mathsf{B}$$

Proof. Decompose the result history. An empty result is incompatible with a visible left endpoint and the continuation equation; an $\mathsf{E0}$ result is incompatible with the unary distance carrier. In the $\mathsf{E1}(k)$ case, the established visible-result split gives either the empty target boundary or the visible target tail witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result shape`

Theorem 57.69 (Metric distance witness one-headed left empty-target exactness)**.** *For histories x, d, a distance witness from a visible left endpoint to an empty right endpoint with visible result exposes the unary tail and identifies the result tail with the left tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{e}, \mathsf{E1}(d)) \Longrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(d, x).$$

Proof. Unpack the distance witness. Unary inversion removes the visible constructor from the left endpoint carrier. The continuation ledger has empty right endpoint, so right-unit exactness identifies $\mathsf{E1}(d)$ with $\mathsf{E1}(x)$; then one-constructor cancellation yields $\mathsf{hsame}(d, x)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 empty target exactness`

Theorem 57.70 (Metric distance witness left e1 visible target exactness)**.** *For histories x, y, d, y_1, a distance witness whose left endpoint and result are visible descends across a visible presentation of the target endpoint:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(y, \mathsf{E1}(y_1)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y_1, d).$$

Proof. Apply the left visible-result case split. The empty-middle branch contradicts the displayed visible presentation of y. In the visible-middle branch, strip the $\mathsf{E1}$ constructor from that presentation and transport the recovered tail witness along the resulting hsame equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target exactness`

Theorem 57.71 (Metric distance witness left e1 visible target step)**.** *For histories x, y, d, a distance witness from $\mathsf{E1}(x)$ to y with result d remains a distance witness after placing the target and result under one visible $\mathsf{E1}$ constructor:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)).$$

Proof. Unpack the witness into unary carriers and the continuation ledger. Unary closure supplies carriers for $\mathsf{E1}(y)$ and $\mathsf{E1}(d)$, and the continuation step rule lifts the same ledger from target y and result d to target $\mathsf{E1}(y)$ and result $\mathsf{E1}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target step`

Theorem 57.72 (Metric distance witness left e1 visible target result shape)**.** *For histories x, y, d, any distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ has a visible result tail, and that tail is again the distance witness from $\mathsf{E1}(x)$ to y:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \exists k.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, k).$$

Proof. Apply the left visible-result shape split to the displayed witness. The empty-target branch would identify $\mathsf{E1}(y)$ with e, contradicting constructor no-confusion. The remaining visible branch supplies the result tail k and the descended witness with target y. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result shape`

Theorem 57.73 (Metric distance witness left e1 visible target result tail exactness)**.** *For histories x, y, d, k, if a distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ has displayed result history-same as $\mathsf{E1}(k)$, then k is the descended distance result from $\mathsf{E1}(x)$ to y:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(k)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, k).$$

Proof. Apply the visible-target result shape theorem to expose a result tail k_0, a displayed comparison $\mathsf{hsame}(d, \mathsf{E1}(k_0))$, and the descended witness. Composing this comparison with the assumed comparison to $\mathsf{E1}(k)$ gives a one-headed history-sameness relation between $\mathsf{E1}(k_0)$ and $\mathsf{E1}(k)$. Constructor inversion descends it to $\mathsf{hsame}(k_0, k)$, and tail transport returns the displayed witness at k. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result tail hsame exact`

Theorem 57.74 (Metric distance witness left e1 visible target result-source absurd)**.** *For histories x, y, d, a distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ with visible result $\mathsf{E1}(d)$ cannot also identify the descended result d with the visible source $\mathsf{E1}(x)$ when the target tail is nonempty:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow (\mathsf{hsame}(y, \mathsf{e}) \Rightarrow \bot) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(x)) \Longrightarrow \bot.$$

Proof. Right-visible result exactness descends the witness to $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$. Transporting its continuation result across $\mathsf{hsame}(d, \mathsf{E1}(x))$ makes y a right-unit continuation at $\mathsf{E1}(x)$, so right-unit uniqueness gives $\mathsf{hsame}(y, \mathsf{e})$, contradicting the displayed nonempty target-tail hypothesis. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result source absurd`

Theorem 57.75 (Metric distance witness left e1 source step)**.** *For histories x, y, d, a distance witness from x to y with result d remains a distance witness after placing the source and result under one visible $\mathsf{E1}$ constructor:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)).$$

Proof. Unpack the witness into unary carriers and the continuation ledger $d = x \cdot y$. Unary closure supplies the new source and result carriers. Since y is unary, the append law rewrites $\mathsf{E1}(x) \cdot y$ as $\mathsf{E1}(x \cdot y)$, so the lifted result $\mathsf{E1}(d)$ satisfies the required continuation equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source step`

Theorem 57.76 (Metric distance witness left e1 source iff)**.** *For histories x, y, d, placing the source and result under one visible $\mathsf{E1}$ constructor is reversible:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. The reverse direction is Theorem 57.298. For the forward direction, split on the target history. The empty target case is the right empty-boundary exactness of the metric witness. A zero-headed target contradicts the unary target carrier recorded in the witness. In the one-headed target case, the result equation descends through the visible constructor, and unary append commutes the visible source constructor across the unary target tail, giving the original continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source iff`

Theorem 57.77 (Metric distance witness left e1 source result iff)**.** *For histories x, y, d, placing both the source and result under one visible $\mathsf{E1}$ constructor is reversible:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. For the forward direction, unpack the displayed witness. Unary inversion on the source and result gives unary carriers for x and d. The continuation ledger identifies $\mathsf{E1}(d)$ with $\mathsf{E1}(x) \cdot y$, and the unary append law for the target rewrites the right side to $\mathsf{E1}(x \cdot y)$; constructor inversion exposes the descended continuation ledger $d = x \cdot y$.

For the reverse direction, apply Definition 57.298. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source result iff`

Theorem 57.78 (Metric distance witness left e1 visible target iff)**.** *For histories x, y, d, placing both the target and result under one visible* E1 *constructor is reversible for a visible left endpoint:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \iff \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d).$$

Proof. For the forward direction, apply Theorem 57.70 to the displayed witness with the reflexive presentation $\mathsf{hsame}(\mathsf{E1}(y), \mathsf{E1}(y))$. This strips the visible target presentation and returns the tail witness with target y and result d.

For the reverse direction, apply Theorem 57.294. The visible target constructor and matching visible result constructor are restored by the continuation step rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target iff`

The ten-field fit is: traditional target visible unary metric descent/extension; BEDC source BHist with UnaryHistory and Cont; carrier MetricDistanceWitness; classifier hsame; ledger one visible target constructor; core theorem Theorem 57.301; exactness reversible target-step readback; status ProofTarget.

Theorem 57.79 (Metric distance witness right e1 result exactness)**.** *For histories x, y, d, a distance witness with right endpoint and result both under one visible constructor descends to the tail witness:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. Unpack the distance witness into unary carrier fields and the continuation ledger. Unary inversion removes the visible constructor from the right endpoint and result carriers. The continuation step-rule inversion removes the same right-side constructor from the continuation ledger, and the four fields repack as $\mathsf{MetricDistanceWitness}(x, y, d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result exactness`

Theorem 57.80 (Metric distance witness right e1 result shape)**.** *For histories x, y, d, every concrete metric distance witness whose right endpoint is visibly one-headed has a visibly one-headed distance ledger, and the exposed distance tail is again the metric witness for the exposed right endpoint:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \mathsf{MetricDistanceWitness}(x, y, k).$$

Proof. Unpack the distance witness into unary carrier fields and the continuation ledger. The continuation result has the form $d = x \cdot \mathsf{E1}(y)$, which is definitionally $\mathsf{E1}(x \cdot y)$. Choose $k = x \cdot y$. Unary inversion on the visible right endpoint gives the carrier for y, unary append closure gives the carrier for k, and the canonical append equality gives the descended continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result shape`

Theorem 57.81 (Metric distance witness right e1 result not empty)**.** *For histories x, y, d, no concrete metric distance witness whose right endpoint is visibly one-headed can have a distance result history-same to* e*:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{e}) \Longrightarrow \bot.$$

Proof. Apply the right e-one result shape theorem to expose a tail k with $\mathsf{hsame}(d, \mathsf{E1}(k))$. Composing this with $\mathsf{hsame}(d, \mathsf{e})$ gives $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{e})$, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result not empty`

Theorem 57.82 (Metric distance witness right e1 result e0 absurd)**.** *For histories x, y, d, no concrete metric distance witness can have a visibly one-headed right endpoint while its distance result is visibly zero-headed:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E0}(d)) \Longrightarrow \bot.$$

Proof. Apply the right e-one result shape theorem to the witness. It exposes a tail k with $\mathsf{hsame}(\mathsf{E0}(d), \mathsf{E1}(k))$. Constructor no-confusion for generated history sameness rejects this zero-headed versus one-headed comparison. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result e0 absurd`

Theorem 57.83 (Metric distance witness right e1 result tail exactness)**.** *For histories x, y, d, k, if a distance witness has a visibly one-headed right endpoint and its displayed result is history-same as $\mathsf{E1}(k)$, then k is the exact descended distance tail:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(k)) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, k).$$

Proof. Apply right e-one result shape to expose an intermediate tail k_0 together with $\mathsf{hsame}(d, \mathsf{E1}(k_0))$ and the descended metric witness. Compose this classifier with the displayed result classifier, invert the resulting one-headed hsame comparison, and transport the tail witness along the exposed tail equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result tail hsame exact`

Theorem 57.84 (Metric distance witness right e1 result tail determinism)**.** *For histories x, y, d, d', two metric distance witnesses with the same source, the same visible right endpoint, and visible results determine the same result tail:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d')) \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Project the continuation ledgers from the two metric witnesses. Continuation determinacy identifies the visible results $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$, and one-constructor inversion descends this classifier to the tail relation $\mathsf{hsame}(d, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result tail deterministic`

Theorem 57.85 (Metric distance witness right e1 source shape)**.** *For histories x, y, d, every concrete metric distance witness whose right endpoint is visibly one-headed exposes a visible distance tail and the matching empty-or-visible source alternative:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \Big((x = \mathsf{e} \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(k, y)) \vee \exists x_1 : \mathsf{E}$$

Proof. Apply Definition 57.330 to the given witness. This produces a history k, a displayed result comparison $\mathsf{hsame}(d, \mathsf{E1}(k))$, and the descended witness $\mathsf{MetricDistanceWitness}(x, y, k)$.

We next rebuild the displayed-result witness with result $\mathsf{E1}(k)$. Keep the original source and right endpoint carriers, close the descended tail carrier under $\mathsf{E1}$, and transport the original continuation field along $\mathsf{hsame}(d, \mathsf{E1}(k))$.

Apply Definition 57.94 to this displayed-result witness. In the empty branch, keep the same k and record $x = \mathsf{e}$, the unary carrier for y, and $\mathsf{hsame}(k, y)$. In the visible branch, keep the same k and the exposed x_1, together with $x = \mathsf{E1}(x_1)$ and the descended metric witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 source shape`

Theorem 57.86 (Metric distance witness right e1 result iff)**.** *For histories x, y, d, a distance witness whose right endpoint and result both carry one visible constructor is equivalent to the descended tail witness:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. The forward direction is right one-headed result exactness. For the reverse direction, keep the source unary carrier, close the right endpoint and result carriers under $\mathsf{E1}$, and lift the continuation ledger by the one-step continuation rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result iff`

Theorem 57.87 (Metric distance witness right e1 result hsame source)**.** *For histories x, x', y, d, a distance witness whose right endpoint and result both carry one visible constructor has the same empty-or-visible source readback after source transport:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(x, x') \Longrightarrow (x' = \mathsf{e} \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, y))$$

or there exists $x_1 : \mathsf{BHist}$ such that

$$x' = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d).$$

Proof. Apply the right one-headed source case split before transport. In the empty branch, empty hsame inversion identifies x' with e. In the visible branch, visible hsame inversion exposes the transported tail, and unary and continuation transport rebuild the descended metric witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result hsame source`

Theorem 57.88 (Metric distance witness right e1 result hsame target)**.** *For histories x, y, y', d, a distance witness whose right endpoint and result both carry one visible constructor has the same empty-or-visible source readback after target-tail transport:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(y, y') \Longrightarrow (x = \mathsf{e} \wedge \mathsf{UnaryHistory}(y') \wedge \mathsf{hsame}(d, y'))$$

or there exists $x_1 : \mathsf{BHist}$ such that

$$x = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d).$$

Proof. Apply the right one-headed source case split before target transport. In the empty-source branch, transport the unary carrier and compose the displayed result-tail comparison with $\mathsf{hsame}(y, y')$. In the visible-source branch, keep the exposed source tail and transport the target carrier and continuation ledger across the target-tail classifier, leaving the source and result endpoints fixed. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result hsame target`

Theorem 57.89 (Metric right e-one result two-sided endpoint transport)**.** *For histories x, x', y, y', d, simultaneous source and target-tail transport preserves the empty-or-visible readback of a right e-one metric result:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \Longrightarrow (x' = \mathsf{e} \wedge \mathsf{UnaryHistory}(y') \wedge \mathsf{hsame}(d, y'))$$

or there exists $x_1 : \mathsf{BHist}$ such that

$$x' = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d).$$

Proof. First apply Definition 57.337 to the displayed witness and the supplied $\mathsf{hsame}(x, x')$. This gives the transported source readback with target tail y still fixed.

In the empty-source branch, keep $x' = \mathsf{e}$. Transport the unary carrier from y to y' along $\mathsf{hsame}(y, y')$, and compose the displayed comparison $\mathsf{hsame}(d, y)$ with $\mathsf{hsame}(y, y')$ to obtain $\mathsf{hsame}(d, y')$.

In the visible-source branch, keep the exposed x_1 and the equality $x' = \mathsf{E1}(x_1)$. The descended witness $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d)$ supplies the source, target, and result carriers; unary transport moves the target carrier from y to y', while metric continuation transport carries the continuation ledger across the same target-tail classifier with source and result fixed. Repacking these fields gives $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d)$.

Ten-field fit: traditional target two-sided visible endpoint transport; BEDC source BHist, hsame, and MetricDistanceWitness; carrier unary metric endpoints; classifier hsame; stability source and target-tail transport; ledger exposed right e-one distance tail; core theorem Definition 57.339; exactness simultaneous transported source readback; standard bridge finite unary metric endpoint transport; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result two sided endpoint transport`

Theorem 57.90 (Metric e-one pair result tail readback)**.** *For histories x, y, d, a metric distance witness whose source, target, and result are all visibly one-headed exposes unary source and target tails and the tail continuation ledger:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, \mathsf{E1}(x) \cdot y).$$

Proof. Apply right e-one result exactness to descend the target and result to $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$. Unary inversion on the source carrier gives unary evidence for x, the descended witness carries unary evidence for y, and its continuation ledger is exactly $d\mathsf{hsameE1}(x) \cdot y$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result tail readback`

Theorem 57.91 (Metric e-one pair result tail exactness iff)**.** *For histories x, y, d, a metric distance witness whose source, target, and result are all visibly one-headed is equivalent to unary tail data together with the exposed continuation ledger:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longleftrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, \mathsf{E1}(x) \cdot y).$$

Proof. For the forward direction, apply Theorem 57.340. It descends the visible target and result constructors and returns exactly the two unary tail carriers and the displayed continuation ledger.

For the reverse direction, start from the unary tail carriers and the hsame ledger. Unary closure builds the source carrier $\mathsf{UnaryHistory}(\mathsf{E1}(x))$, and the continuation ledger transports the append result so that d is the concrete distance from $\mathsf{E1}(x)$ to y. This gives the central witness $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$.

Apply Theorem 57.294 to this central witness. The visible target constructor and the matching visible result constructor are restored, producing $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d))$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result iff`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result tail exactness iff`

The ten-field fit is: traditional target one-headed metric tail exactness; BEDC source BHist, UnaryHistory, Cont, and MetricDistanceWitness; carrier concrete metric distance witnesses; classifier hsame; stability visible-constructor closure and continuation transport; ledger exposed tail distance; core theorem Theorem 57.341; exactness bidirectional tail data; standard bridge finite metric endpoint readback; status ProofTarget.

Theorem 57.92 (Metric distance witness right e1 visible source exactness)**.** *For histories x, y, d, x_1, a distance witness whose right endpoint and result are visibly tagged descends to the tail while respecting a visible presentation of the source endpoint:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(x, \mathsf{E1}(x_1)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d).$$

Proof. First apply right one-headed result exactness to remove the matching visible constructors from the right endpoint and result. The displayed hsame source presentation then rewrites the descended source to $\mathsf{E1}(x_1)$, preserving the concrete distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 visible source exactness`

Theorem 57.93 (Metric distance witness right e1 empty-source exactness)**.** *For histories y, d, a distance witness from the empty source to a visible right endpoint with visible result is exactly unary evidence for the endpoint and sameness of the descended distance:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, y).$$

Proof. First descend the right endpoint and visible result to $\mathsf{MetricDistanceWitness}(\mathsf{e}, y, d)$. The empty-left exactness theorem then gives the unary endpoint carrier and $\mathsf{hsame}(d, y)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 empty source exactness`

Theorem 57.94 (Metric distance witness right one-headed source cases)**.** *For histories x, y, d, a distance witness with right endpoint and result both under one visible constructor either starts at the empty source boundary or starts from a one-headed source whose tail still carries the descended distance witness:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \big(x = \mathsf{e} \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, y)\big)$$

or there exists $x_1 : \mathsf{BHist}$ such that

$$x = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d).$$

Proof. First descend the right endpoint and result with the right one-headed result exactness theorem. Classify the unary source endpoint. The empty-source case is the empty-left exactness theorem for the descended witness, giving the target carrier and the distance-target sameness. A zero-headed source contradicts the unary source carrier. A one-headed source keeps the descended witness itself with the displayed source equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 source cases`

Theorem 57.95 (Metric distance witness right-boundary visible-context exactness)**.** *For histories p, q, x, d, a visible-context distance witness whose central right endpoint is empty is exactly unary evidence for the visible ledgers and the central left endpoint, together with the right-boundary classifier:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(d, x).$$

Proof. In the forward direction, apply Definition 57.13 with central right endpoint e. This recovers unary evidence for p and q together with $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d)$. Apply Definition 57.59 to that central witness to obtain $\mathsf{UnaryHistory}(x)$ and $\mathsf{hsame}(d, x)$.

Conversely, assume unary evidence for p, q, and x, together with $\mathsf{hsame}(d, x)$. By Definition 57.59 from right to left, these central data build $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d)$. Applying Definition 57.13 from right to left with prefix p and suffix q gives the displayed visible-context witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right boundary visible context iff`

Theorem 57.96 (Metric distance witness left-boundary visible-context exactness)**.** *For histories p, q, y, d, a visible-context distance witness whose central left endpoint is empty is exactly unary evidence for the visible ledgers and the central right endpoint, together with the left-boundary classifier:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, y{\cdot}q, (p{\cdot}d){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, y).$$

Proof. In the forward direction, apply Definition 57.13 with central left endpoint e. This recovers unary evidence for p and q together with $\mathsf{MetricDistanceWitness}(\mathsf{e}, y, d)$. Apply Definition 57.40 to that central witness to obtain $\mathsf{UnaryHistory}(y)$ and $\mathsf{hsame}(d, y)$.

Conversely, assume unary evidence for p, q, and y, together with $\mathsf{hsame}(d, y)$. By Definition 57.40 from right to left, these central data build $\mathsf{MetricDistanceWitness}(\mathsf{e}, y, d)$. Applying Definition 57.13 from right to left with prefix p and suffix q gives the displayed visible-context witness.

The per-object template is as follows. The traditional target is visible metric left-boundary readback. The BEDC source is BHist with Cont and $\mathsf{UnaryHistory}$. The carrier is $\mathsf{MetricDistanceWitness}$ and the classifier is hsame. The ledger is the visible prefix and suffix context. The core theorem is left-boundary exactness, exactness is the displayed equivalence, the standard bridge is metric boundary readback, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left boundary visible context iff`

Theorem 57.97 (Metric distance witness empty-boundary visible-context exactness)**.** *For histories p, q, d, a visible-context distance witness whose central left and right endpoints are both empty is exactly unary evidence for the visible ledgers together with the empty-distance classifier:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \iff \mathsf{UnaryHistory}(p) \ \wedge \ \mathsf{UnaryHistory}(q) \ \wedge \ \mathsf{hsame}(d, \mathsf{e}).$$

Proof. Apply Definition 57.13 to remove the visible prefix and suffix. The central witness is $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d)$. Applying Definition 57.40 identifies it with the empty-boundary classifier $\mathsf{hsame}(d, \mathsf{e})$, while the visible-context theorem records exactly the unary histories for p and q.

Conversely, unary histories for p and q and the classifier $\mathsf{hsame}(d, \mathsf{e})$ build the central witness by Definition 57.40. Reattaching the same visible prefix and suffix by Definition 57.13 gives the displayed visible-context witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context iff`

Theorem 57.98 (Metric distance witness empty-boundary visible-context depth zero)**.** *For histories p, q, d, a visible-context distance witness whose central left and right endpoints are both empty forces the central distance depth to be zero:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Apply Definition 57.13 to remove the visible prefix and suffix. The central witness is then $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d)$. Applying Definition 57.40 gives $\mathsf{hsame}(d, \mathsf{e})$. Case analysis on this history classifier reduces the displayed depth equation to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context depth zero`

Theorem 57.99 (Metric depth-zero distance has empty endpoints)**.** *For histories x, y, d, a concrete distance witness whose distance ledger has structural depth zero has empty endpoints:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Classify the displayed distance history. The empty distance case is exactly the empty-distance witness exactness theorem. A zero-headed or one-headed distance has successor depth, contradicting the displayed zero-depth equation by constructor no-confusion for natural numbers. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness depth zero empty endpoints`

Theorem 57.100 (Metric empty endpoints have depth-zero distance)**.** *For histories x, y, d, a concrete distance witness whose endpoints are both history-same to* e *has structural distance depth zero:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{hsame}(x, \mathsf{e}) \to \mathsf{hsame}(y, \mathsf{e}) \to \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Eliminate the two endpoint sameness witnesses, reducing both endpoints to e. The continuation ledger inside the distance witness then reduces the distance history itself to e, and the depth definition evaluates to zero. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty endpoints depth zero`

Theorem 57.101 (Metric distance depth zero iff empty)**.** *For a history d, structural metric distance depth is zero exactly when d classifies with the empty history:*

$$\mathsf{MetricDistanceDepth}(d) = 0 \iff \mathsf{hsame}(d, \mathsf{e}).$$

Proof. Case-analyze the history d. The empty case is reflexive. A zero-headed or one-headed history has successor depth, so it cannot satisfy the zero-depth equation. Conversely, a history-sameness proof with e reduces the depth equation to the empty reflexive case. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth zero iff empty`

Theorem 57.102 (Metric distance positive depth iff nonempty)**.** *For a history d, positive structural metric distance depth is equivalent to not being history-same as* e*:*

$$0 < \mathsf{MetricDistanceDepth}(d) \iff (d\mathsf{hsamee} \to \mathsf{False}).$$

Proof. Case-analyze d. The empty case has depth zero and cannot satisfy the nonempty predicate. A zero-headed or one-headed history has successor depth, and constructor separation rules out history-sameness with e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth positive iff nonempty`

Theorem 57.103 (Metric distance append positive depth iff)**.** *For histories p and d, an appended distance ledger has positive structural depth exactly when one of its two factors has positive structural depth:*

$$0 < \mathsf{MetricDistanceDepth}(p \cdot d) \iff 0 < \mathsf{MetricDistanceDepth}(p) \vee 0 < \mathsf{MetricDistanceDepth}(d).$$

Proof. Use the positive-depth classifier to replace positive depth by internal nonemptiness. Kernel append nonemptiness then splits the appended history into the left or right nonempty factor. The converse applies the same two classifiers in reverse and rebuilds nonemptiness of the appended history from the displayed nonempty factor. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append positive iff`

Theorem 57.104 (Metric distance positive depth shape cases)**.** *For a history d, positive structural metric distance depth exposes one of the two visible history constructors:*

$$(\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \implies (\exists z : \mathsf{BHist}.\ d = \mathsf{E0}(z)) \vee (\exists z : \mathsf{BHist}.\ d = \mathsf{E1}(z)).$$

Proof. Case-analyze d. The empty branch contradicts the displayed positive-depth hypothesis because the structural depth of e is zero. The two visible branches return the exposed payload under $\mathsf{E0}$ or $\mathsf{E1}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth positive shape cases`

Theorem 57.105 (Metric distance depth append-left equality emptiness)**.** *For histories p and d, appending p on the left preserves structural metric distance depth exactly when p is the empty history:*

$$\mathsf{MetricDistanceDepth}(p \cdot d) = \mathsf{MetricDistanceDepth}(d) \iff \mathsf{hsame}(p, \mathsf{e}).$$

Proof. Induct on the displayed tail d. The empty tail is precisely Definition 57.253. A zero-headed or one-headed tail reduces the equality through successor injectivity, and the reverse direction rebuilds the same successor equality after the left prefix is identified with e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append left eq iff empty`

Theorem 57.106 (Metric distance depth append-left cancellation)**.** *For histories $p, d, e : \mathsf{BHist}$, equality of structural metric distance depths after appending the same left prefix cancels that prefix:*

$$\mathsf{MetricDistanceDepth}(p{\cdot}d) = \mathsf{MetricDistanceDepth}(p{\cdot}e) \implies \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e).$$

Proof. The structural depth of a left append decomposes as the depth of the common prefix plus the depth of the tail. Rewriting both displayed sides by this depth-add equation reduces the claim to left cancellation for addition in $\mathbb{N}$, yielding equality of the two tail depths. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append left cancel`

Theorem 57.107 (Metric strict prefix depth decomposition)**.** *For histories $h, k : \mathsf{BHist}$, a strict unary prefix witness from h to k exposes a nonempty unary tail whose continuation reconstructs k, and the metric depth of k splits as the depth of h plus the depth of that tail:*

$$\mathsf{NatUnaryStrictPrefix}(h, k) \implies \exists tail : \mathsf{BHist}.\ \mathsf{UnaryHistory}(tail) \wedge (tail = \mathsf{e} \to \mathsf{False}) \wedge \mathsf{Cont}(h, tail, k) \wedge \mathsf{MetricDistance}$$

Proof. Unpack the strict-prefix witness. It already supplies a unary nonempty tail and a continuation ledger from h through that tail to k. Inverting the continuation identifies k with $h \cdot tail$. Induction on the concrete tail then proves the structural depth equation for append, giving the displayed sum formula. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth strict prefix decomposes`

Theorem 57.108 (Metric distance depth visible-context cancellation)**.** *For histories p, q, d, e : BHist, equality of structural depths after a common visible prefix and suffix cancels both contexts:*

$$\mathsf{MetricDistanceDepth}((p{\cdot}d){\cdot}q) = \mathsf{MetricDistanceDepth}((p{\cdot}e){\cdot}q) \Longrightarrow \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e$$

Proof. Induct on the visible suffix q. The empty suffix is exactly Definition 57.258. In the two visible constructor cases, both displayed depths have a common successor head, so successor injectivity removes the suffix constructor and the induction hypothesis cancels the remaining visible context. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context cancel`

Theorem 57.109 (Metric distance depth append-right cancellation iff)**.** *For histories p, d, e : BHist, appending the same right suffix preserves and reflects equality of structural metric distance depth:*

$$\mathsf{MetricDistanceDepth}(d{\cdot}p) = \mathsf{MetricDistanceDepth}(e{\cdot}p) \Longleftrightarrow \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e).$$

Proof. Induct on the common suffix p. The empty suffix is immediate. In the two visible constructor cases, both appended depths carry the same successor head; successor injectivity gives the forward implication, and congruence of successor rebuilds the appended equality in the reverse implication. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append right cancel iff`

Theorem 57.110 (Metric distance depth append-right equality emptiness)**.** *For histories p and d, appending p on the right preserves structural metric distance depth exactly when p is the empty history:*

$$\mathsf{MetricDistanceDepth}(d \cdot p) = \mathsf{MetricDistanceDepth}(d) \Longleftrightarrow \mathsf{hsame}(p, \mathsf{e}).$$

Proof. The structural depth of a right append decomposes as the depth of d plus the depth of p. If the appended depth equals the original depth, cancellation by $\mathbb{N}$-addition forces the depth of p to be zero, hence Definition 57.253 identifies p with e. Conversely, substituting $p = \mathsf{e}$ makes the right append definitionally equal to d. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append right eq iff empty`

Theorem 57.111 (Metric distance same-source continuation result cancellation)**.** *For unary histories x, y, z and continuation rows $\mathsf{Cont}(x, y, d)$ and $\mathsf{Cont}(x, z, e)$, equality of the two result depths cancels the common source depth:*

$$\mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e) \Longrightarrow \mathsf{MetricDistanceDepth}(y) = \mathsf{MetricDistanceDepth}(z).$$

Proof. Package the two continuation rows as metric-distance witnesses. Depth additivity gives $\mathsf{depth}(d) = \mathsf{depth}(x) + \mathsf{depth}(y)$ and $\mathsf{depth}(e) = \mathsf{depth}(x) + \mathsf{depth}(z)$. The displayed result depth equality therefore has a common left summand, and natural-number cancellation identifies the two target depths. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth same source cont result cancel`

Theorem 57.112 (Metric distance depth visible-context equality emptiness)**.** *For histories p, q, d, adding visible context on both sides preserves the central structural metric distance depth exactly when both visible contexts are empty:*

$$\mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = \mathsf{MetricDistanceDepth}(d) \iff \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}).$$

Proof. Expand structural depth over the two append contexts. The resulting natural number equation has the form $(\mathsf{depth}(p) + \mathsf{depth}(d)) + \mathsf{depth}(q) = \mathsf{depth}(d)$, so associativity and commutativity of addition reduce it to a zero-sum constraint on the two context depths. Then Definition 57.253 identifies both contexts with e. The reverse direction substitutes the two empty-context identifications and applies the one-sided append equality theorems above. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context eq iff empty`

Theorem 57.113 (Metric visible-context zero depth iff empty components)**.** *For histories p, q, d, the full visible-context ledger has structural metric depth zero exactly when the prefix, central distance, and suffix are all empty:*

$$\mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = 0 \iff \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}).$$

Proof. The zero-depth classifier first identifies the whole visible ledger with e. BEDC append-empty inversion splits the outer append into $(p \cdot d)\mathsf{hsamee}$ and $q\mathsf{hsamee}$, and a second inversion splits $p \cdot d$. Conversely, substituting the three empty components makes the displayed depth definition evaluate to zero. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context zero iff`

Theorem 57.114 (Metric visible-context total zero components)**.** *For histories p, q, x, y, d, a visible-context distance witness whose full result ledger has structural depth zero exposes empty prefix, source, target, suffix, and central distance components:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \to \\ &\quad \mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = 0 \to \\ &\quad \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}) \wedge \mathsf{hsame}(d, \mathsf{e}).\end{aligned}$$

Proof. Zero depth classifies the full result ledger with e, and two append-empty inversions expose the empty prefix, central distance, and suffix. Transport the witness continuation along the same empty result classifier. The empty-result continuation inversion then classifies the visible source $p \cdot x$ and target $y \cdot q$ with e, and append-empty inversion splits these endpoint histories into their displayed components. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context total zero components`

Theorem 57.115 (Metric visible-context depth-zero collapse)**.** *For histories p, q, x, y, d, a visible-context distance witness whose central distance has structural depth zero collapses the central distance and both central endpoints to the empty history:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \to \mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Apply visible-context exactness to peel away the prefix p and suffix q, recovering the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$. The depth-zero endpoint theorem gives $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$, while the depth-zero classifier theorem gives $\mathsf{hsame}(d, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context depth zero collapse`

Theorem 57.116 (Metric empty-boundary visible-context continuation left unit)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness and a continuation out of the central distance identify the continuation result with its right input:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{Cont}(d, r, out) \to \mathsf{hsame}(out, r).$$

Proof. Apply empty-boundary visible-context exactness to read back $\mathsf{hsame}(d, \mathsf{e})$. Case analysis on this classifier reduces the continuation to an empty-left continuation, whose result is the right input by the left-unit law for Cont. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation left unit result`

Theorem 57.117 (Metric empty-boundary visible-context continuation left unit iff)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness turns continuation from the central distance into exactly the left-unit continuation classifier:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \big(\mathsf{Cont}(d, r, out) \leftrightarrow \mathsf{hsame}(out, r)\big).$$

Proof. The visible-context empty-boundary theorem gives $\mathsf{hsame}(d, \mathsf{e})$. Case analysis on this classifier reduces the displayed continuation predicate to $\mathsf{Cont}(\mathsf{e}, r, out)$, which is characterized by the BEDC left-unit equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation left unit iff`

Theorem 57.118 (Metric empty-boundary visible-context continuation right unit iff)**.** *For histories p, q, d, l, out, an empty-boundary visible-context distance witness turns continuation into the central distance into exactly the right-unit continuation classifier:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \big(\mathsf{Cont}(l, d, out) \leftrightarrow \mathsf{hsame}(out, l)\big).$$

Proof. Again the visible-context empty-boundary theorem gives $\mathsf{hsame}(d, \mathsf{e})$. After case analysis, the displayed continuation predicate is $\mathsf{Cont}(l, \mathsf{e}, out)$, which is characterized by the BEDC right-unit equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation right unit iff`

Theorem 57.119 (Metric empty-boundary continuation splice)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness makes a continuation through the central distance splice to the continuation that skips that empty central boundary:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out).$$

Proof. The empty-boundary visible-context theorem gives $\mathsf{hsame}(d, \mathsf{e})$. After case analysis on this classifier, the left continuation is $\mathsf{Cont}(l, \mathsf{e}, mid)$, so its unit law identifies mid with l. The right continuation is therefore the desired $\mathsf{Cont}(l, r, out)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation splice`

Theorem 57.120 (Metric empty-boundary visible-context witness splice)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness upgrades a two-step continuation through the central distance into a metric distance witness for the visible endpoints:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{UnaryHistory}(l) \to \mathsf{UnaryHistory}(r) \to \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{MetricDistanceWitness}(l, r, out).$$

Proof. The empty-boundary continuation-splice theorem gives the concrete continuation $\mathsf{Cont}(l, r, out)$ from the visible-context witness and the two displayed continuation rows. Together with the displayed unary carriers for l and r, the BEDC unary-continuation closure theorem supplies $\mathsf{UnaryHistory}(out)$. These four rows are precisely the fields of $\mathsf{MetricDistanceWitness}(l, r, out)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context witness splice`

Theorem 57.121 (Metric empty-boundary continuation splice determinacy)**.** *For histories $p, q, d, l, r, mid, out, out$ an empty-boundary visible-context distance witness makes the spliced continuation result match any displayed direct continuation result:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \to \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out') \to \mathsf{hsame}(out, out')$$

Proof. The continuation-splice theorem supplies $\mathsf{Cont}(l, r, out)$ from the empty-boundary visible-context witness and the two displayed continuation rows. BEDC continuation determinacy compares this spliced continuation with the displayed $\mathsf{Cont}(l, r, out')$, giving $\mathsf{hsame}(out, out')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation splice hsame`

Theorem 57.122 (Metric visible-context positive central depth has a nonempty endpoint)**.** *For histories p, q, x, y, d, a visible-context distance witness whose central distance has positive structural depth exposes a nonempty central endpoint:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \to (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to ((\mathsf{hsame}(x, \mathsf{e}) \to \mathsf{False}) \vee (\mathsf{hsame}$$

Proof. Apply visible-context exactness to peel away the displayed prefix p and suffix q, obtaining the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$.

It remains to convert positive structural depth into nonempty-distance input. If $\mathsf{hsame}(d, \mathsf{e})$ held, the depth-zero classifier theorem would give $\mathsf{MetricDistanceDepth}(d) = 0$, contradicting the positive-depth hypothesis.

Apply the nonempty-distance endpoint theorem to the central witness and this derived nonempty-distance contradiction. The result is exactly the displayed disjunction: either the left central endpoint is not history-same to e, or the right central endpoint is not history-same to e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context positive depth nonempty endpoint`

Theorem 57.123 (Metric visible-context positive central depth e-one endpoint cases)**.** *For histories p, q, x, y, d : BHist, positive central distance depth in a visible MetricUp ledger exposes an e-one constructor at one central endpoint with a unary tail:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \to \\ &\quad (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to \\ &\quad (\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)).\end{aligned}$$

Here the BEDC source is MetricDistanceWitness*, the carrier objects are* UnaryHistory *histories, the classifier is* hsame*, and the obligation is constructor readback under visible prefix and suffix ledgers.*

Proof. Apply visible-context exactness to remove the prefix p and suffix q from the displayed ledger. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$.

Next convert positive central depth into the nonempty-distance input required by the constructor readback theorem. If $d$hsamee held, then Definition 57.253 would give $\mathsf{MetricDistanceDepth}(d) = 0$, contradicting the positive-depth hypothesis.

Apply Definition 57.216 to the recovered central witness and this nonempty-distance contradiction. The result is exactly the displayed disjunction of e-one endpoint witnesses with unary tails. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context positive depth e1 endpoint cases`

Theorem 57.124 (Metric distance witness visible-context symmetric classifier)**.** *For histories $p, q, x, y, d_{xy}, d_{yx}$, symmetric distance witnesses remain classified by* hsame *after the same visible source and target contexts are peeled away:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p \cdot y, x \cdot q, (p \cdot d_{yx}) \cdot q) \Longrightarrow \mathsf{hsame}(d_{xy}, d_{yx}).\end{aligned}$$

Proof. Apply Definition 57.13 to each contextual witness. The first application recovers $\mathsf{MetricDistanceWitness}(x, y,$ and the second recovers $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$. The central witnesses are then classified by Definition 57.9. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context symmetric classifier`

Theorem 57.125 (Metric distance witness prefix-independent symmetric classifier)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric metric distance witnesses with independent visible contexts still determine the same central distance classifier:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{hsame}(d_{xy}, d_{yx}).\end{aligned}$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This recovers central witnesses $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ and $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$. The central symmetric classifier then gives $\mathsf{hsame}(d_{xy}, d_{yx})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent symmetric classifier`

Theorem 57.126 (Metric distance witness prefix-independent symmetric zero-depth collapse)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent contexts and a zero-depth forward distance collapse the two central distances and both central endpoints:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{hsame}(d_{xy}, d_{yx}) \wedge \mathsf{MetricDistanceDepth}(d_{yx}) = 0 \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.303 to the two visible-context witnesses. This gives $\mathsf{hsame}(d_{xy}, d_{yx})$.

Apply Definition 57.267 to the first visible-context witness and the zero-depth hypothesis for d_{xy}. This gives the empty endpoint classifiers $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$.

Finally peel the second visible-context witness with Definition 57.13. The recovered central witness $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$, together with the two empty endpoint classifiers in reversed order, is the input required by Definition 57.252; hence $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$. Combining these three pieces gives the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent symmetric zero depth collapse`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth collapse`

Theorem 57.127 (Metric prefix-independent symmetric zero-depth reverse empty-boundary)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent contexts and a zero-depth forward distance also package the reverse context as its own empty-boundary distance witness:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q').\end{aligned}$$

Proof. Peel the reverse visible-context witness with Definition 57.13. This recovers the unary context data for p' and q' together with the central reverse witness $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$.

Apply Definition 57.304 to the two displayed visible-context witnesses and the zero-depth forward distance. The resulting endpoint classifiers include $\mathsf{hsame}(y, \mathsf{e})$ and $\mathsf{hsame}(x, \mathsf{e})$.

Finally use Definition 57.389 in the reverse context. The recovered carriers for p' and q', together with the empty endpoint classifiers in reverse order, rebuild exactly $\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth reverse empty boundary package`

Theorem 57.128 (Metric prefix-independent symmetric zero-depth two-sided empty-boundary package)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent contexts and a zero-depth forward distance package both contextual empty-boundary witnesses together with the reverse zero-depth row and the central endpoint classifiers:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q) \wedge \\ &\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q') \wedge \mathsf{MetricDistanceDepth}(d_{yx}) = 0 \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.304 to the two visible-context witnesses and the zero-depth forward distance. This gives the same-distance classifier, the reverse zero-depth row, and the two central endpoint classifiers $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$.

Peel the forward visible-context witness with Definition 57.13 to recover the unary carrier rows for p and q. These carriers, together with the endpoint classifiers from the collapse theorem, are exactly the right-hand data in Definition 57.389; hence they build $\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q)$.

Apply Definition 57.305 to the same hypotheses to obtain the reverse contextual empty-boundary witness. Conjoining the forward package, the reverse package, the reverse zero-depth row, and the two endpoint classifiers gives the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth two sided empty boundary package`

The ten-field fit is: traditional target metric zero-depth symmetric boundary packaging; BEDC source BHist with MetricDistanceWitness and MetricDistanceDepth; carrier MetricDistanceWitness; classifier hsame; stability zero-depth endpoint collapse; ledger the two symmetric visible contexts; core theorem Theorem 57.306; exactness two-sided empty-boundary packaging; standard bridge none claimed; status ProofTarget.

Theorem 57.129 (Metric prefix-independent symmetric zero-depth continuation splice)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, r, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance make any continuation through the reverse central distance splice across the empty boundary:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out).$$

Proof. The symmetric zero-depth collapse theorem gives $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$. The depth-zero classifier theorem converts this to $\mathsf{hsame}(d_{yx}, \mathsf{e})$. Case analysis reduces the left continuation to the right-unit case, identifying mid with l, and the remaining continuation is exactly the displayed splice result. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth continuation splice`

Theorem 57.130 (Metric prefix-independent symmetric zero-depth witness splice)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, r, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance turn any unary continuation through the reverse central distance into a metric distance witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{UnaryHistory}(l) \to \mathsf{UnaryHistory}(r) \to \mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{MetricDistanceWitness}(l, r, out).$$

Proof. The continuation-splice theorem gives $\mathsf{Cont}(l, r, out)$ from the two visible-context witnesses, the zero-depth row, and the two displayed continuation ledgers through d_{yx}. The supplied unary histories for l and r, together with unary continuation closure, give the unary carrier row for out. These three carrier rows and the spliced continuation are exactly the metric distance witness package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth witness splice`

Theorem 57.131 (Metric symmetric zero-depth distance continuation result)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$ symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance make any two-step continuation through the two central distances return to the original left endpoint:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{xy}, mid) \to \mathsf{Cont}(mid, d_{yx}, out) \to \mathsf{hsame}(out, l).$$

Proof. The forward zero-depth hypothesis converts d_{xy} to the empty history by the depth-zero classifier theorem. The symmetric zero-depth collapse theorem also gives zero depth for d_{yx}, hence $d_{yx}\mathsf{hsamee}$. Case analysis on these two empty-distance classifiers reduces both continuations to right-unit continuations, first identifying mid with l and then out with mid. Transitivity gives the displayed $\mathsf{hsame}(out, l)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth distance continuation result`

Theorem 57.132 (Metric symmetric zero-depth reverse distance continuation result)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance also make the reverse-ordered two-step continuation return to the original left endpoint:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, d_{xy}, out) \to \mathsf{hsame}(out, l).$$

Proof. Apply Definition 57.304 to the two visible-context witnesses and the zero-depth hypothesis for d_{xy}. This gives $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$.

Now apply Definition 57.309 with the two visible contexts swapped: p', q', y, x, d_{yx} play the forward roles and p, q, x, y, d_{xy} play the reverse roles. The recovered zero-depth row for d_{yx} is the forward depth hypothesis in that swapped instance, and the displayed ledgers are exactly $\mathsf{Cont}(l, d_{yx}, mid)$ followed by $\mathsf{Cont}(mid, d_{xy}, out)$.

Thus the swapped distance-continuation result returns $\mathsf{hsame}(out, l)$. The fit records traditional target zero-distance round trip, BEDC source BHist with Cont and MetricDistanceWitness, carrier the visible-context witness pair, classifier hsame, stability from zero-depth collapse and continuation unit laws, ledger the two central distance histories, core theorem this reverse-order round trip, exactness boundary-unit cancellation, standard bridge zero metric distance as identity path, and status ProofTarget. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth reverse distance continuation result`

Theorem 57.133 (Metric distance witness visible-context determinacy)**.** *For histories p, q, x, y, d, e, two contextual distance ledgers for the same visible endpoints determine the same central distance:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot e) \cdot q) \Longrightarrow \mathsf{hsame}(d, e).$$

Proof. Apply Definition 57.13 to the first contextual witness to recover the central witness $\mathsf{MetricDistanceWitness}(x,$ Apply the same theorem to the second contextual witness to recover $\mathsf{MetricDistanceWitness}(x, y, e)$.

Unpack the two central distance witnesses. They provide continuation ledgers $\mathsf{Cont}(x, y, d)$ and $\mathsf{Cont}(x, y, e)$ over the same source and target histories; continuation determinacy, Theorem 18.16, yields $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context determinacy`

Theorem 57.134 (MetricSpace visible-context triangle boundary)**.** *For unary visible contexts p, q and central witnesses* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ *and* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$*, a displayed append or continuation row for the composite distance d_{xyz} yields* $\mathsf{MetricDistanceWitness}(p \cdot x, z \cdot q, (p \cdot d_{xyz}) \cdot q)$*. Every source and result read is confined to* BHist*,* UnaryHistory*,* Cont*, and* hsame*.*

The boundary introduces no host metric, real inequality, completion, or quotient distance.

Proof. Apply the central triangle append closure to the two middle witnesses and the displayed composite row. This produces the central witness $\mathsf{MetricDistanceWitness}(x, z, d_{xyz})$ over the same BHist and continuation coordinates.

Transport that central witness through the visible prefix and suffix contexts using the existing visible-context exactness lemmas. Determinacy then identifies the distance row seen after peeling the contexts with the central row d_{xyz}.

The construction mentions only the unary histories, continuation ledgers, and hsame transports used by those lemmas. Hence no ambient metric, inequality, completion, or quotient coordinate is added. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceVisibleContextTriangleBoundary`

Theorem 57.135 (MetricSpace LocatedReal distance source boundary)**.** *Let L_x and L_y be accepted* $\mathsf{LocatedReal}^{\uparrow}$ *packets whose* $\mathsf{StreamName}^{\uparrow}$ *schedules admit a common finite observation window. Suppose their* DyadicRatCore *endpoint ledgers and radius ledgers are transported to that window by the displayed* hsame *and* Cont *rows, and suppose a contextual metric row* $\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q)$ *is displayed for the two projected endpoints.*

Then any $\mathsf{Met}^{\uparrow}$ *consumer reads the distance-source boundary exactly through the finite rows named by Definition 309.11 together with the metric visible-context rows: the* $\mathsf{RegSeqRat}^{\uparrow}$ *source windows, the* DyadicRatCore *endpoint and radius rows, the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, and the* hsame *classifier on the central distance row d.*

Conversely, any packet carrying precisely those rows repacks as the metric distance-source row for the two located packets. No quotient real, host real equality, trichotomy principle, open-cover compactness datum, or metric completion object is part of this boundary.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness locatedreal distance source boundary`

Proof. This boundary is the $\mathsf{Met}^{\uparrow}$ consumer of the located-real carrier of **??**; the cited sibling supplies the two accepted $\mathsf{LocatedReal}^{\uparrow}$ packets and their endpoint rows, while this theorem supplies the visible-context distance read.

Project L_x and L_y with Definition 309.11. This leaves only the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source windows, the common $\mathsf{StreamName}^{\uparrow}$ schedule, the transported DyadicRatCore endpoint and radius ledgers, the enclosure classifier rows, and the visible BHist, hsame, Cont, Pkg, and NameCert provenance.

Next apply Definition 57.13 to the contextual metric row. The visible prefix and suffix rows are peeled away, leaving the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$ over the two projected located endpoints.

For uniqueness of the central distance read, apply Definition 57.133. Any second packet with the same displayed source windows, endpoint ledgers, schedule rows, provenance rows, and contextual metric witness has a central distance classifier $\mathsf{hsame}(d, e)$, so the metric consumer cannot distinguish a hidden distance coordinate.

The converse direction repacks exactly the displayed rows: the located packets come from the handoff surface, the contextual distance row supplies the metric visible context, and the central hsame classifier is the only distance classifier retained. Since each coordinate in this construction is one of the listed finite rows, no excluded real-number, compactness, or completion principle is introduced. □

Theorem 57.136 (Metric distance witness triangle append closure)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a distance witness from x to y, a distance witness from y to z, and a distance witness from d_{xy} to z close the appended triangle ledger:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x \cdot d_{yz}).$$

Proof. Unpack the three distance witnesses to their continuation fields. The three continuations identify d_{xy} with $x \cdot y$, d_{yz} with $y \cdot z$, and d_{xyz} with $(x \cdot y) \cdot z$. Associativity of history append classifies this last ledger with $x \cdot (y \cdot z)$, hence with $x \cdot d_{yz}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append closed`

Theorem 57.137 (Metric distance witness unary append commutation package)**.** *For unary histories x and y, the appended endpoints $x \cdot y$ and $y \cdot x$ carry the canonical metric distance witness $(x \cdot y) \cdot (y \cdot x)$, and the endpoints are* hsame*:*

$$\mathsf{MetricDistanceWitness}(x \cdot y, y \cdot x, (x \cdot y) \cdot (y \cdot x)) \wedge \mathsf{hsame}(x \cdot y, y \cdot x).$$

Proof. Unary append closure supplies the two endpoint carriers and the displayed distance carrier. The distance ledger is the canonical continuation from $x \cdot y$ to $y \cdot x$, and unary append commutativity gives the endpoint classifier. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness unary append comm package`

Theorem 57.138 (Metric distance witness composite endpoint append result)**.** *For histories $x, x', y, y', z, z', d_{xy}, d_{xyz}$, endpoint comparisons $x\mathsf{hsame}x'$, $y\mathsf{hsame}y'$, and $z\mathsf{hsame}z'$ transport the composite distance ledger to the endpoint-normal append result:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x' \cdot (y' \cdot z')).$$

Proof. Transport along the three endpoint hsame comparisons and unpack the two distance witnesses to their continuation fields. Constructor analysis of the two continuation ledgers reduces the displayed result comparison to associativity of history append. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness composite endpoint hsame append result`

Theorem 57.139 (Metric distance witness triangle continuation middle closure)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a distance witness from x to y, a distance witness from y to z, and a continuation ledger $\mathsf{Cont}(d_{xy}, z, d_{xyz})$ close the middle-composite distance witness:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz}).$$

Proof. The two distance witnesses expose the continuation ledgers $\mathsf{Cont}(x, y, d_{xy})$ and $\mathsf{Cont}(y, z, d_{yz})$. Together with the displayed ledger $\mathsf{Cont}(d_{xy}, z, d_{xyz})$, constructor analysis reduces the resulting source-to-middle continuation to associativity of history append. The unary carrier for d_{xyz} follows from unary continuation closure applied to d_{xy} and z. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle cont middle closed`

Theorem 57.140 (Metric distance witness visible-context triangle continuation middle closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a visible-context witness for the left triangle edge, a central witness from y to z, and a continuation ledger from d_{xy} by z close the central middle-composite distance witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz}).$$

Proof. Apply Definition 57.13 to the visible-context witness. This recovers the central edge witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$. The recovered edge witness, the given y-to-z witness, and the displayed continuation ledger are exactly the inputs of Definition 57.359, which returns the displayed middle-composite distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle cont middle closed`

Theorem 57.141 (Metric distance witness triangle middle result determinacy)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, displayed, if the witnesses from x to y and from y to z are given, and $\mathsf{Cont}(d_{xy}, z, d_{xyz})$ displays the middle composite, then any distance witness from x to d_{yz} has the same result:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, displayed) \Longrightarrow \mathsf{hsame}(d_{xyz}, displayed).$$

Proof. Triangle continuation middle closure constructs the canonical witness $\mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz})$ from the two edge witnesses and the displayed middle continuation. The displayed witness has the same source and target histories as this canonical witness, so continuation determinacy applied to their distance ledgers identifies the two result histories. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle cont middle result deterministic`

Theorem 57.142 (Metric distance witness triangle append depth add)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, three triangle distance witnesses identify the displayed ledger with $x \cdot d_{yz}$ and compute its structural depth as the sum of the left endpoint depth and the middle-to-right distance depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{MetricDistanceDepth}(d_{xyz}) = \mathsf{MetricDistanceDepth}(x$$

Proof. Triangle append closure supplies the equality $\mathsf{hsame}(d_{xyz}, x \cdot d_{yz})$. Together with the unary carrier fields already present in the three distance witnesses, this equality is a concrete distance witness from x to d_{yz} with result d_{xyz}. The metric depth addition law applied to that central witness gives the displayed depth equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append depth add`

Theorem 57.143 (Metric distance witness continuation depth add)**.** *For histories x, y, z, d_{xy}, d_{xyz}, a distance witness from x to y and a displayed continuation from d_{xy} by z compute the final structural distance depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{UnaryHistory}(z) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = \mathsf{Metri}$$

Proof. The first witness supplies the unary carrier of d_{xy}. Together with the unary carrier of z and the displayed continuation ledger, this forms a second distance witness from d_{xy} to z with result d_{xyz}. Apply the depth additivity theorem to the first and second witnesses, then associate addition. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness cont depth add`

Theorem 57.144 (Metric distance witness triangle depth-zero collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if three triangle distance witnesses have a final ledger of structural depth zero, then the triangle append equality holds and every displayed endpoint and intermediate distance collapses to* e*:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longrightarrow \\ &\mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. Triangle append closure forms the central witness from x to d_{yz} with result d_{xyz}. The depth-zero endpoint theorem for this central witness forces x and d_{yz} to be empty. Empty-distance exactness for the witness from y to z then forces y and z to be empty, and the continuation field in the witness from x to y forces d_{xy} to be empty. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle depth zero collapse`

Theorem 57.145 (Metric triangle left distance empty collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the left distance in a metric triangle is history-same to* e*, then its endpoints collapse and the remaining two displayed distances are exactly the right endpoint:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, z) \wedge \mathsf{hsame}(d_{yz}, z).\end{aligned}$$

Proof. Empty-distance exactness for the witness from x to y gives $x\mathsf{hsame}\mathsf{e}$ and $y\mathsf{hsame}\mathsf{e}$. With d_{xy} collapsed, empty-left exactness for the outer witness gives $d_{xyz}\mathsf{hsame}z$. Collapsing y in the second witness and applying the same empty-left exactness gives $d_{yz}\mathsf{hsame}z$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left distance empty collapse`

Theorem 57.146 (Metric triangle depth-zero iff empty spine)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, three concrete metric distance witnesses identify final structural depth zero exactly with emptiness of the three endpoints and the two displayed edge distances:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &\mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longleftrightarrow \\ &\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. For the forward direction, apply Definition 57.364 to the three triangle witnesses and the displayed zero-depth equation. The theorem returns a larger conjunction containing the append equality as its first component; discard that component and retain precisely the five empty-spine classifiers.

For the reverse direction, assume the five displayed classifiers. Apply Definition 57.252 to the first edge witness, using $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$, to get $\mathsf{MetricDistanceDepth}(d_{xy}) = 0$. Then use Definition 57.253 to recover $\mathsf{hsame}(d_{xy}, \mathsf{e})$ as the certified empty left endpoint for the outer witness. Finally apply Definition 57.252 to $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$, together with $\mathsf{hsame}(d_{xy}, \mathsf{e})$ and $\mathsf{hsame}(z, \mathsf{e})$, to obtain $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle depth zero iff empty spine`

Fit: traditional target triangle degeneracy criterion; BEDC source $BHist$ metric distance witnesses; carrier the three witnessed edges; classifier hsame; stability supplied by existing depth-zero and empty-endpoint lemmas; ledger the two edge distances; core theorem this iff; exactness zero-depth iff empty spine; standard bridge none claimed; status ProofTarget.

Theorem 57.147 (Metric distance witness visible-context triangle depth-zero collapse)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the first triangle edge is displayed through visible source and target ledgers and the final triangle distance has structural depth zero, then the triangle append equality holds and every central endpoint and intermediate distance collapses to* e*:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, p \cdot d_{xy} \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longrightarrow \\ &\mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.13 to the contextual first edge. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ while leaving the two remaining triangle witnesses unchanged.

Apply Definition 57.364 to the central witness, the witness from y to z, the witness from d_{xy} to z, and the depth-zero hypothesis. The resulting conjunction is exactly the stated visible-context collapse conclusion. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle depth zero collapse`

Theorem 57.148 (Metric distance witness visible-context triangle depth-zero iff empty spine)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, visible source and target ledgers around the first edge preserve the exact zero-depth triangle criterion:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, p \cdot d_{xy} \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &\mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longleftrightarrow \\ &\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.13 to the contextual first edge. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, so the three hypotheses now have the same shape as the central triangle criterion.

For the forward direction, apply Definition 57.366 to the central witness, the witness from y to z, and the witness from d_{xy} to z. The forward half turns the displayed zero-depth equation into the five empty-spine classifiers.

For the reverse direction, use the same central iff theorem with the recovered central witness and the two unchanged triangle witnesses. Its reverse half turns the five displayed empty-spine classifiers into $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle depth zero iff empty spine`

Theorem 57.149 (Metric triangle nonempty distance has a nonempty endpoint)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the three triangle distance witnesses hold and the final triangle distance is not history-same as* e*, then at least one endpoint of the triangle is not history-same as* e*:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow ($$

Proof. Apply metric nonempty-distance endpoint exactness to the outer witness $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ If the nonempty endpoint is z, the conclusion follows directly. If it is d_{xy}, apply the same exactness theorem to $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, yielding a nonempty x or y endpoint. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty endpoint nonempty`

Theorem 57.150 (Metric triangle nonempty spine one-headed endpoint cases)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the three triangle distance witnesses hold and the final triangle distance is not history-same as* e*, then at least one triangle endpoint is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow (d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longrightarrow$$
$$(\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)) \vee (\exists z_0,\ z = \mathsf{E1}(z_0) \wedge \mathsf{UnaryHistory}($$

Proof. Apply the nonempty endpoint theorem for metric triangles. In each endpoint case, the corresponding unary carrier is a field of one of the triangle distance witnesses. Unary nonempty one-tail readback turns that nonempty carrier into the displayed $\mathsf{E1}$ endpoint with a unary tail. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty spine e1 endpoint cases`

Theorem 57.151 (Metric triangle nonempty spine iff nonempty distance)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, the same three triangle distance witnesses identify nonemptiness of the final triangle distance exactly with nonemptiness of at least one point of the triangle spine:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$(d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow$$
$$((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False}) \vee (z\mathsf{hsamee} \to \mathsf{False}) \vee (d_{xy}\mathsf{hsamee} \to \mathsf{False}) \vee (d_{yz}\mathsf{hsamee} \to \mathsf{False})).$$

Proof. For the forward direction, apply Definition 57.369 to the three triangle witnesses and the displayed nonempty-distance hypothesis. The resulting endpoint disjunction is injected into the five-way spine disjunction by choosing the corresponding x, y, or z summand.

For the reverse direction, assume the five-way spine disjunction and suppose $d_{xyz}\mathsf{hsamee}$. By Definition 57.253, this gives $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. Applying Definition 57.366 to the three triangle witnesses then yields $\mathsf{hsame}(x, \mathsf{e})$, $\mathsf{hsame}(y, \mathsf{e})$, $\mathsf{hsame}(z, \mathsf{e})$, $\mathsf{hsame}(d_{xy}, \mathsf{e})$, and $\mathsf{hsame}(d_{yz}, \mathsf{e})$.

Case-analyze the five-way spine disjunction. In each case, the selected nonempty spine component is a function from the corresponding empty classifier to False, so applying it to the matching classifier from the empty-spine conjunction gives the desired contradiction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty spine iff nonempty distance`

Theorem 57.152 (Metric distance witness visible-context triangle nonempty spine iff nonempty distance)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a triangle whose first edge is displayed through visible contexts has the same exact nonempty-spine criterion as the central triangle:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$(d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow$$
$$((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False}) \vee (z\mathsf{hsamee} \to \mathsf{False}) \vee (d_{xy}\mathsf{hsamee} \to \mathsf{False}) \vee (d_{yz}\mathsf{hsamee} \to \mathsf{False})).$$

Proof. Apply Definition 57.13 to the displayed first-edge witness. Its forward direction recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, while the unary evidence for p and q is not needed by the remaining argument.

Now apply Definition 57.371 to this recovered central witness and to the two unchanged witnesses $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$ and $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$. The result is exactly the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle nonempty spine iff nonempty distance`

Theorem 57.153 (Metric distance witness triangle append context closure)**.** *If p and q are supplied as unary visible contexts, distance witnesses from x to y, from y to z, and from d_{xy} to z produce a distance witness from $p \cdot x$ to $d_{yz} \cdot q$ whose result is $p \cdot d_{xyz} \cdot q$:*

$$\begin{aligned}&\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, p \cdot d_{xyz} \cdot q).\end{aligned}$$

Proof. The triangle append closure identifies the central result with $x \cdot d_{yz}$. Together with the carrier fields already present in the three distance witnesses, this gives the central witness from x to d_{yz} with result d_{xyz}. Visible-context closure then appends the unary contexts p and q on the outside. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context closed`

Theorem 57.154 (Metric distance witness visible-context triangle append closure)**.** *If the distance witness from x to y is displayed through visible contexts p, q, and the two remaining triangle witnesses are present, then the triangle append result is unchanged at the central layer:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, p{\cdot}d_{xy}{\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}$$

Proof. Visible-context exactness recovers the central distance witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ from the displayed contextual witness. The ordinary triangle append closure applied to this recovered witness and the two remaining triangle witnesses gives $d_{xyz}\mathsf{hsame}x \cdot d_{yz}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle append closed`

Theorem 57.155 (Metric distance witness triangle append context result determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then any displayed distance witness from $p \cdot x$ to $d_{yz} \cdot q$ has the same result as the canonical contextual triangle result:*

$$\begin{aligned}&\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, displayed) \Longrightarrow \mathsf{hsame}(displayed, p \cdot d_x\end{aligned}$$

Proof. Triangle append context closure constructs the canonical witness from $p \cdot x$ to $d_{yz} \cdot q$ with result $p \cdot d_{xyz} \cdot q$. The displayed witness and this canonical witness have the same source and target histories, so continuation determinacy for their distance ledgers identifies their result histories. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context result deterministic`

Theorem 57.156 (Metric distance witness triangle append context source determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then any displayed witness with the canonical contextual result and target $d_{yz} \cdot q$ determines the exposed source:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x', d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(x, x').$$

Proof. Triangle append context closure constructs the canonical contextual witness from $p \cdot x$ to $d_{yz} \cdot q$ with result $(p \cdot d_{xyz}) \cdot q$. Visible-context source determinacy then compares this canonical witness with the displayed witness, using the shared target $d_{yz} \cdot q$ and shared central distance d_{xyz}, and recovers $\mathsf{hsame}(x, x')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context source deterministic`

Theorem 57.157 (Metric distance witness triangle append context endpoint determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then the canonical contextual triangle result $(p \cdot d_{xyz}) \cdot q$ determines both exposed endpoints:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \left(\begin{array}{l} (\mathsf{MetricDistanceWitness}(p \cdot x', d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(x, x')) \\ \wedge \\ (\mathsf{MetricDistanceWitness}(p \cdot x, e \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(d_{yz}, e)) \end{array} \right)$$

Proof. Triangle append context closure first constructs the canonical contextual witness $\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q)$ from the three triangle witnesses and the unary context hypotheses.

For the displayed witness with source $p \cdot x'$ and target $d_{yz} \cdot q$, visible-context source determinacy compares it with the canonical contextual witness and recovers $\mathsf{hsame}(x, x')$ at the exposed source.

For the displayed witness with source $p \cdot x$ and target $e \cdot q$, visible-context target determinacy compares it with the same canonical contextual witness and recovers $\mathsf{hsame}(d_{yz}, e)$ at the exposed middle target. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint deterministic`

Theorem 57.158 (Metric distance witness triangle append context endpoint transport)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then endpoint comparisons on the exposed source and middle target transport the canonical contextual triangle witness:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d_{yz}, e) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. Triangle append context closure first constructs the canonical contextual witness $\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q)$ from the three triangle witnesses and the unary context hypotheses.

The source comparison $\mathsf{hsame}(x, x')$ gives the contextual comparison $\mathsf{hsame}(p \cdot x, p \cdot x')$ by constructor reduction of the BEDC append context. The middle-target comparison $\mathsf{hsame}(d_{yz}, e)$ similarly gives $\mathsf{hsame}(d_{yz} \cdot q, e \cdot q)$. The result history is unchanged.

Apply the metric witness field transport theorem to these two endpoint comparisons and the reflexive result comparison. It transports the unary endpoint fields and the continuation ledger of the canonical contextual witness, which is exactly the displayed transported witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint transport`

Theorem 57.159 (Metric distance witness triangle append context endpoint iff)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then the canonical contextual triangle result classifies exactly the exposed source and middle target:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \Big(\mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q) \Longleftrightarrow \mathsf{hsame}(x, x') \wedge \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint iff`

Proof. For the forward direction, apply Definition 57.377 to the displayed witness after visible-context exactness has placed it in the canonical contextual form. The source projection gives $\mathsf{hsame}(x, x')$, while the target projection gives $\mathsf{hsame}(d_{yz}, e)$; together these are the two components of the displayed conjunction.

For the reverse direction, assume $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(d_{yz}, e)$. Apply Definition 57.373 to the two unary contexts and the three triangle witnesses. This gives the canonical witness

$$\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Append congruence transports $\mathsf{hsame}(x, x')$ to $\mathsf{hsame}(p \cdot x, p \cdot x')$ and transports $\mathsf{hsame}(d_{yz}, e)$ to $\mathsf{hsame}(d_{yz} \cdot q, e \cdot q)$. Unary transport preserves the endpoint carriers across these classifiers, and Definition 57.346 transports the continuation ledger while keeping the contextual result fixed. Repacking the transported carriers with this ledger gives the displayed witness $\mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q)$. □

Theorem 57.160 (Metric triangle right distance empty collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to e, then the shared endpoint and right endpoint collapse and the remaining displayed distances read back to the left endpoint:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, x) \wedge \mathsf{hsame}(d_{xyz}, x).$$

Proof. Empty-distance exactness for the witness from y to z gives $y\mathsf{hsame}\mathsf{e}$ and $z\mathsf{hsame}\mathsf{e}$. Collapsing y in the first witness and using empty-right exactness identifies d_{xy} with x. Collapsing z in the outer witness identifies d_{xyz} with d_{xy}, and transitivity composes this with the previous readback to obtain $d_{xyz}\mathsf{hsame}x$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty collapse`

Theorem 57.161 (Metric triangle left distance empty visible-context closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the left distance in a metric triangle is history-same to e, then the triangle closes through the visible context with empty central source:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, z \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. The left-distance empty collapse identifies the left endpoints with e and reads d_{xyz} back to z. Empty-left exactness therefore builds the central witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz})$. The visible-context iff then appends the certified unary ledgers p and q around this central witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left distance empty visible context closed`

Theorem 57.162 (Metric triangle right distance empty visible-context closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to* e*, then the triangle closes through the visible context with empty central target:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, \mathsf{e} \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. The right-distance empty collapse identifies the shared and right endpoints with e and reads d_{xyz} back to x. Empty-right exactness therefore builds the central witness $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz})$. The visible-context iff then appends the certified unary ledgers p and q around this central witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty visible context closed`

Theorem 57.163 (Metric triangle right distance empty spine witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to* e*, then the outer displayed distance itself is a metric witness from x to the empty endpoint:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz}).$$

Proof. The right-distance empty collapse identifies d_{xyz} with x and the first metric witness supplies unary membership of x. Empty-right metric exactness then packages this endpoint readback as a metric witness from x to e with displayed distance d_{xyz}. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty spine witness`

Theorem 57.164 (Metric triangle right empty boundary witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$*,* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$*, and* $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ *form a metric triangle and* $d_{yz}\mathsf{hsame}\mathsf{e}$*, then the outer result is itself a distance witness from x to the empty boundary:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz}).$$

Proof. By Definition 57.380, the right empty distance identifies d_{xyz} with x. The canonical right-empty witness $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, x)$ uses the source carrier of the first edge, the empty unary carrier, and the right unit continuation $x \cdot \mathsf{e} = x$. Transporting the result field along $d_{xyz}\mathsf{hsame}x$ gives the displayed witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right empty boundary witness`

Theorem 57.165 (Metric triangle bilateral empty edge collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if both displayed edge distances in a metric triangle are history-same to* e*, then all three endpoints, the outer displayed distance, the final structural depth, and the empty-to-empty outer witness collapse to the empty case:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \wedge \mathsf{MetricDistanceWitness}$$

Proof. Apply Definition 57.365 to the three triangle witnesses and the classifier $\mathsf{hsame}(d_{xy}, \mathsf{e})$. This gives the empty classifiers for x and y, and reads d_{xyz} and d_{yz} back to the right endpoint z.

Apply Definition 57.380 to the same three witnesses and the classifier $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This gives the empty classifiers for y and z, and reads d_{xy} and d_{xyz} back to the left endpoint x.

The endpoint classifiers give the first three conjuncts. For the outer distance, compose the left-collapse readback $\mathsf{hsame}(d_{xyz}, z)$ with $\mathsf{hsame}(z, \mathsf{e})$ by transitivity of hsame. Apply Definition 57.253 to this empty final-distance classifier to obtain $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$.

The empty-boundary metric witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ is the exact empty-distance case. Transporting its result field along $d_{xyz}\mathsf{hsamee}$ yields $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz})$. Conjoining these rows gives the displayed package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle bilateral empty edge collapse`

Fit: traditional target degenerate triangle normalization; BEDC source $BHist$ metric distance witnesses; carrier the three witnessed edges; classifier hsame; stability from the two empty-edge collapse lemmas and depth-zero readback; ledger the two displayed edge distances; core theorem this block; exactness the endpoint and final-distance collapse; standard bridge none claimed; status $\mathsf{ProofTarget}$.

Theorem 57.166 (Metric triangle two-sided empty boundary package)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the two edge distances in a metric triangle are both history-same to e, then every displayed endpoint and the outer displayed distance collapse to the empty boundary, and the outer result is itself an empty-to-empty distance witness:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow$$
$$\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz}).$$

Proof. Apply Definition 57.365 to the same triangle and the hypothesis $d_{xy}\mathsf{hsamee}$. This gives $x\mathsf{hsamee}$, $y\mathsf{hsamee}$, $d_{xyz}\mathsf{hsame}z$, and $d_{yz}\mathsf{hsame}z$.

Apply Definition 57.380 to the same triangle and the hypothesis $d_{yz}\mathsf{hsamee}$. This gives $y\mathsf{hsamee}$, $z\mathsf{hsamee}$, $d_{xy}\mathsf{hsame}x$, and $d_{xyz}\mathsf{hsame}x$.

The endpoint conclusions package the three empty endpoint readbacks. Symmetry and transitivity of hsame compose $d_{xyz}\mathsf{hsame}z$ with $z\mathsf{hsamee}$, so $d_{xyz}\mathsf{hsamee}$.

Finally, the empty-boundary metric witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ is the exact empty-distance case. Transporting its result field along $d_{xyz}\mathsf{hsamee}$ yields $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness two sided empty triangle boundary package`

Theorem 57.167 (Metric triangle left empty boundary witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$, and $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ form a metric triangle and $d_{xy}\mathsf{hsamee}$, then the outer result is itself a distance witness from the empty boundary to z:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz}).$$

Proof. By Definition 57.365, the left empty distance identifies d_{xyz} with z.

The second edge supplies the unary carrier for z. Apply Definition 57.40 with this carrier and the displayed classifier $d_{xyz}\mathsf{hsame}z$ to package the canonical witness from e to z with displayed distance d_{xyz}. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left empty boundary witness`

Theorem 57.168 (Metric triangle two-sided empty witness exactness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$, *and* $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ *form a metric triangle and both displayed edge distances are history-same to* e, *then the outer result simultaneously carries every empty-boundary witness row needed by the triangle:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz}) \wedge \mathsf{MetricDistanceWitness}(x, \mathsf{e},$$

Proof. First apply Definition 57.385 to the three triangle witnesses and the two classifiers $\mathsf{hsame}(d_{xy}, \mathsf{e})$ and $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This gives $\mathsf{hsame}(d_{xyz}, \mathsf{e})$ together with the empty-to-empty witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d$

Next apply Definition 57.387 to the same triangle and $\mathsf{hsame}(d_{xy}, \mathsf{e})$. This supplies the boundary row $\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz})$.

Finally apply Definition 57.384 to the same triangle and $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This supplies $\mathsf{MetricDistanceWitness}(x, \mathsf{e},$ and the four rows assemble into the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness two sided empty triangle witness exactness`

Theorem 57.169 (Metric RealAlgOrder apartness distance scope)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the* RealAlgOrder *constant boundary, the finite packet supplied by Definition 38.46 and read by Definition 304.5 and Definition 304.8 is exactly the* $\mathsf{Met}^{\uparrow}$ *positive-separation distance scope.*

The scope consists of the two endpoint histories, the positive $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness row, the* hsame *transports, the* Cont *routes, the visible* Pkg *provenance, and the* NameCert *ledger tying the endpoints to their constant-rational source. Repacked as a metric consumer row, these data form the positive-separation side of a* MetricDistanceWitness *boundary and expose no trichotomy, completion, quotient-real equality, open-cover compactness, or host real equality.*

Proof. Project the constant carrier, classifier, algebra, order, and apartness rows from the RealAlgOrder constant boundary. By Definition 38.46, the consumer sees only the two $\mathsf{Real}^{\uparrow}$ endpoint histories, their rational source witnesses, the positive dyadic or apartness row, and the displayed hsame, Cont, Pkg, and NameCert provenance.

Apply the scoped certificate Definition 38.43. It keeps the endpoint histories inside the constant-rational boundary and prevents any additional real-order coordinate from being read by the downstream consumer.

Next apply Definition 304.5 and Definition 304.8. These two rows identify the same finite packet as apartness data consumable by metric or order constructions, with the positive row and endpoint readbacks still tied to the same BHist, hsame, Cont, Pkg, and NameCert ledger.

Finally repack the resulting finite separation row as the positive-separation side of the $\mathsf{Met}^{\uparrow}$ distance consumer boundary. The metric visible-context and boundary exactness rows are used only to keep the BEDC ledger stable under contextual prefixes and empty-boundary readback; they do not add a coordinate for trichotomy, completion, quotient-real equality, open-cover compactness, or host real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartnessDistanceScope`

Theorem 57.170 (Metric RealAlgOrder apartness positive reflection)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints, any finite packet satisfying Definition 57.324 and the* $\mathsf{ApartnessReal}^{\uparrow}$ *metric consumer boundary determines exactly the positive-separation component of the* $\mathsf{Met}^{\uparrow}$ *distance witness. Every metric consumer may read the same positive dyadic or apartness row, endpoint histories,* hsame

transports, Cont *routes,* Pkg *provenance, and* NameCert *provenance supplied by the scoped packet. The reflection exports no converse trichotomy, quotient-real equality, completion, or host equality.*

Proof. Start with the constant-rational handoff Definition 38.46. Its packet gives the endpoint histories, the rational source rows, and the positive dyadic or apartness row that can be consumed without reading an additional real-order coordinate.

Apply Definition 38.43. The certificate keeps those rows inside the constant-rational RealAlgOrder surface and records the displayed hsame, Cont, Pkg, and NameCert provenance.

Use Definition 304.5 together with Definition 304.8. These results identify the same positive row as the separation data that an ApartnessReal$^{\uparrow}$ metric consumer may read.

Finally apply Definition 57.324 and project the positive side of the resulting MetricDistanceWitness boundary. This is only a repacking of the displayed finite rows, so the metric consumer gains no coordinate for converse trichotomy, quotient-real equality, completion, or host equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartnessPositiveReflection`

Theorem 57.171 (Metric RealAlgOrder positive-radius distance boundary)**.** *For constant-rational* Real$^{\uparrow}$ *endpoints carried by the scoped* RealAlgOrder$^{\uparrow}$ *boundary, the positive* DyadicRatCore$^{\uparrow}$ *radius or apartness row, the two endpoint* BHist *rows, the displayed* hsame *transports, the* Cont *routes, visible* Pkg *provenance, and the local* NameCert *rows are exactly the positive-distance boundary of the* MetricDistanceWitness *consumed by* Met$^{\uparrow}$*. Thus the concrete dyadic-or-apartness row named by the* RealAlgOrder$^{\uparrow}$ *packet is the positive-distance component read by the metric witness, and every such metric witness component recovers the same finite positive-radius row.*

The boundary is local to the displayed BEDC rows. It contains no trichotomy, quotient-real equality, completion, open-cover compactness, host metric-space structure, or host real equality.

Proof. For the forward direction, start with the scoped constant-rational RealAlgOrder$^{\uparrow}$ packet. Apply Definition 38.46 to expose exactly the two endpoint histories, their rational source rows, the positive dyadic or apartness row, the hsame transports, the Cont routes, visible Pkg provenance, and the local NameCert ledger. Then apply Definition 304.5 and Definition 57.325. These rows identify the same finite packet as the positive component of the MetricDistanceWitness boundary read by Met$^{\uparrow}$.

For the reverse direction, begin with a positive-distance boundary of the MetricDistanceWitness whose endpoints are the same constant-rational Real$^{\uparrow}$ rows. Use Definition 57.312 to read the positive-radius budget through the scoped RealAlgOrder$^{\uparrow}$ certificate, and then use Definition 57.315 to identify the public positive-distance component with the finite apartness packet of the same endpoint histories, transports, routes, package provenance, and name-certificate rows.

Both directions therefore name the same concrete positive row. The repacking steps only move between the RealAlgOrder, ApartnessReal, and MetricSpace consumer surfaces; they do not introduce a branch for trichotomy, a quotient-real equality, a completion principle, an open-cover compactness principle, a host metric-space structure, or a host real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderPositiveRadiusDistanceBoundary`
↪ variant `BEDC.Derived.MetricUp.MetricRealAlgOrder positive radius distance boundary`

Theorem 57.172 (RealAlgOrder apartness metric symmetric separation)**.** *For constant-rational* Real$^{\uparrow}$ *endpoints carried by the* RealAlgOrder *constant boundary, the positive-separation packet identified by Definition 57.325 remains a valid positive-separation component of the* Met$^{\uparrow}$ *distance witness after the two endpoint histories are swapped. The swapped orientation keeps the same positive* DyadicRatCore$^{\uparrow}$ *radius or apartness row, the same finite separation ledger, and the same displayed*

hsame, Cont, Pkg, *and* NameCert *provenance; only the left and right endpoint histories are read in the reverse order.*

This is a metric-facing consequence of apartness symmetry. It gives the symmetric endpoint orientation required by a distance witness, but it does not add trichotomy, quotient-real equality, completion, or host real equality.

Proof. Start with the constant-rational handoff Definition 38.46. It supplies the two endpoint histories, rational source rows, positive dyadic or apartness row, and visible hsame, Cont, Pkg, and NameCert provenance.

Apply the scoped certificate Definition 38.43. The certificate keeps both endpoint histories inside the same constant-rational RealAlgOrder boundary and does not introduce any additional real-order coordinate.

Use Definition 304.4 to exchange the two endpoint histories and endpoint readbacks. The apartness row, positive radius or separation ledger, and all displayed transport and provenance rows are preserved componentwise.

Finally apply Definition 57.325 to the swapped packet. Since the positive-separation component is the same finite row read in the opposite endpoint order, it is again a valid positive component of the $\mathsf{Met}^{\uparrow}$ distance witness, with no added equality, completion, or trichotomy principle. □

Lean4 ✓ `BEDC.Derived.MetricUp.RealAlgOrderApartnessMetricSymmetricSeparation`

Theorem 57.173 (Metric RealAlgOrder apartness triangle budget)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints* x, y, z *carried by the* RealAlgOrder *constant boundary, two adjacent finite apartness packets for* (x, y) *and* (y, z) *supply the exact visible rows required by the* $\mathsf{Met}^{\uparrow}$ *triangle-budget consumer for the outer pair* (x, z) *after endpoint orientation is normalized. The consumer may read only the three endpoint histories, the two positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness rows, the finite adjacent separation ledgers, the induced outer-pair* BHist *row, the displayed* hsame *transports, the* Cont *routes, the visible* Pkg *provenance, and the local* NameCert *rows attached to the constant-rational sources.*

The statement is a local RealAlgOrder-to-MetricSpace budget law. It exports no trichotomy, quotient-real equality, completion, open-cover compactness, or host metric-space structure.

Proof. Begin with the two adjacent apartness packets. Apply Definition 57.325 to each packet; this exposes exactly the positive-separation rows, endpoint histories, hsame transports, Cont routes, Pkg provenance, and NameCert rows already present in the constant-rational boundary.

If one adjacent packet is presented in the reverse endpoint order, use Definition 57.327 to swap only the endpoint histories. The positive row and finite ledger are unchanged, so the two adjacent packets now share the middle endpoint history y in the orientation read by the triangle consumer.

With the adjacent rows normalized, invoke the $\mathsf{Met}^{\uparrow}$ visible-context triangle boundary and the public law package. These rows concatenate the two finite positive-separation ledgers into the outer-pair triangle budget while preserving the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

No excluded principle is used in the repacking step. Trichotomy, quotient-real equality, completion, open-cover compactness, and a host metric-space structure are not coordinates of the adjacent RealAlgOrder packets or of the visible triangle-budget consumer surface. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartness triangle budget`

Theorem 57.174 (RealAlgOrder apartness metric NameCert route)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary, the carrier and classifier rows of Definition 38.43, the finite apartness-to-metric handoff of Definition 38.46, the* $\mathsf{ApartnessReal}^{\uparrow}$ *metric handoff of Definition 304.5, the positive-separation reflection of Definition 57.325, the symmetric endpoint separation of Definition 57.327, and the triangle-budget rows of Definition 57.328 assemble into the exact finite* NameCert *route consumable by* $\mathsf{Met}^{\uparrow}$ *and by the* $\mathsf{S}^{1\uparrow}$ *boundary reader.*

The route consists only of endpoint BHist *rows, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness rows, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows attached to the constant-rational sources and metric consumers. It exports no trichotomy, quotient-real equality, completion, open-cover compactness, or host equality.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderApartnessMetricNameCertRoute`

Proof. First read the two endpoint rows through Definition 38.43. This fixes the scoped $\mathsf{RealAlgOrder}^{\uparrow}$ carrier and classifier surface and keeps the endpoint histories tied to their constant-rational source rows.

Next apply Definition 38.46 and Definition 304.5. These two handoffs present the same finite apartness row as metric-facing positive-separation data, with the visible BHist, hsame, Cont, Pkg, and NameCert provenance unchanged.

Now use Definition 57.325 to package the positive row as the positive component of the $\mathsf{Met}^{\uparrow}$ distance witness, and use Definition 57.327 when the consumer requests the opposite endpoint orientation.

Finally apply Definition 57.328. The triangle route concatenates only the displayed adjacent separation ledgers and therefore gives exactly the finite NameCert route needed by metric and circle-boundary consumers, without adding any excluded host-order, quotient, completion, compactness, or equality principle. □

Definition 57.175 (MetricSpace real-distance carrier)**.** *The* $\mathsf{Met}^{\uparrow}$ *real-distance carrier for the open Real Algebra and Geometry phase is the finite unary endpoint packet*

$$(x, y, d, r;\ w, b, c, p, a)$$

where x *and* y *are unary* BHist *endpoint histories,* w *is the displayed row* $\mathsf{MetricDistanceWitness}(x, y, d)$*,* b *is the closed ball-budget row* $\mathsf{MetricBallBudget}_d(x, y; r)$*,* c *is the classifier row* hsame *on endpoint and result histories,* p *is the visible* Cont *provenance for the distance witness, and* a *is the* RealAlgOrder *budget packet consisting of the constant-rational endpoint rows, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness row, ordered-real comparison, addition, transitivity, and addition-monotonicity rows used by the budget predicate.*

The carrier exposes exactly these rows. It does not contain an ambient host metric space, a quotient-real equality, a completeness principle, an open-cover compactness principle, or a topology generator separate from the displayed distance packet.

Lean4 (definition) `BEDC.Derived.MetricUp.MetricspaceRealDistanceCarrier`

Theorem 57.176 (MetricSpace RealAlgOrder positive-radius budget interface)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints in the carrier of Definition 57.311, the* a*-coordinate supplies exactly the ordered-real fields required by Definition 57.232 and consumed by Definition 57.233: ordered-real comparison, addition, transitivity, and two-sided addition-monotonicity. The interface is read through the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *certificate and the apartness distance scope of Definition 57.324,*

so the positive-radius row is the same finite order-or-apartness row exposed to the metric budget consumer.

This interface exports no trichotomy branch, quotient-real equality, completeness principle, open-cover compactness, or host metric-space row.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderPositiveRadiusBudgetInterface`

Proof. Project the a-coordinate from Definition 57.311. By construction this coordinate is the RealAlgOrder budget packet attached to the same endpoint histories, distance witness, classifier, continuation, and provenance rows as the metric carrier.

Apply Definition 766.2. Its order stability and ledger-exactness rows identify the comparison, addition, transitivity, and addition-monotonicity data as finite displayed order rows, while Definition 766.7 prevents any trichotomy, quotient-real equality, host real carrier, or completeness coordinate from being read through the order or apartness side.

Next use Definition 57.324 together with Definition 57.315. These theorems identify the metric-facing positive row with the same endpoint BHist histories, positive DyadicRatCore$^{\uparrow}$ radius or apartness row, hsame transports, Cont routes, Pkg provenance, and NameCert ledger supplied by the scoped RealAlgOrder$^{\uparrow}$ packet.

Finally compare this inventory with Definition 57.232. The predicate asks only for ordered-real comparison, addition, budget transitivity, and two-sided budget-addition monotonicity, and Definition 57.233 consumes only those displayed fields. Therefore the a-coordinate is precisely the positive-radius budget interface used by the composition theorem, with none of the excluded host principles added. □

Theorem 57.177 (MetricSpace public law package)**.** *The real-distance carrier of Definition 57.311 supports the public* Met$^{\uparrow}$ *distance laws needed by the open phase:*

1. *symmetry, by transporting the endpoint order through* MetricDistanceWitness *symmetry and the RealAlgOrder apartness symmetric-separation row;*
2. *identity boundary, by the empty-distance row and the empty-boundary visible-context rows;*
3. *triangle-budget composition, by the visible-context triangle boundary and the* MetricBallBudget *composition row;*
4. *carrier transport, by* hsame *transport of endpoints and result histories together with the displayed* Cont *provenance.*

These laws are public laws of the finite BEDC packet only. They do not assert an arbitrary host metric, completeness, compactness, or a hidden equality on real distances.

Proof. For symmetry, read the distance row through Definition 57.9. The positive real-distance side is kept in the same packet by Definition 57.327, so swapping the endpoints changes only the orientation of the unary endpoint histories.

For the identity boundary, apply Definition 57.7. The visible-context empty-boundary rows then show that a zero public distance read is exactly the empty endpoint boundary up to hsame, with no quotient-real equality added.

For triangle-budget composition, use Definition 57.134 to form the composite witness over the same endpoint and continuation rows. Then apply Definition 57.233 to compose the two displayed budget rows.

For carrier transport, use the endpoint and result hsame rows already contained in Definition 57.311. The visible Cont provenance identifies the transported witness with the same public distance packet, so every exported law remains inside the carrier. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealDistanceCarrier public law package`

Theorem 57.178 (MetricSpace open-phase consumer interface)**.** *Every* $\mathsf{S}^{1\uparrow}$ *consumer and every later L10 consumer in the open Real Algebra and Geometry phase reads* $\mathsf{Met}^{\uparrow}$ *real-distance data only through the carrier of Definition 57.311 and the law package of Definition 57.313. The consumer surface is the RealAlgOrder apartness distance scope together with the existing visible-context consumer surface; it exposes no additional host metric-space coordinate, no completion row, and no open-cover compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceOpenPhaseConsumerInterface`

Proof. Start with Definition 57.324. It identifies the constant-rational endpoint histories, positive apartness or dyadic radius, and visible hsame, Cont, Pkg, and NameCert provenance available to a metric-facing real-distance consumer.

Next apply Definition 57.237. That consumer theorem restricts $\mathsf{S}^{1\uparrow}$ and the named downstream metric consumers to the visible-context distance witness, ball-budget, classifier, and finite provenance rows.

The L10 consumers in the open phase can only compose with the same RealAlgOrder apartness scope and the same visible-context public surface. Therefore their readable interface is exactly Definition 57.311 equipped with Definition 57.313, and no host metric, completion, or open-cover compactness coordinate is exported. □

Theorem 57.179 (MetricSpace RealAlgOrder positive-distance public correspondence)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints, the positive component of the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311 is equivalent to the finite apartness packet isolated by Definition 57.324 and Definition 57.325. Concretely, to read a positive public distance row is to read the two endpoint* BHist *histories, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness row, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *ledger supplied by the* RealAlgOrder *constant-rational boundary; conversely, those same rows repack as the positive component of the carrier.*

Endpoint reversal is supplied by Definition 57.327. The correspondence introduces no trichotomy, completion, quotient-real equality, open-cover compactness, or host equality.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderPositiveDistancePublicCorrespondence`

Proof. Project the carrier of Definition 57.311. Its positive side is not a separate host metric object; it is the displayed RealAlgOrder budget packet attached to the distance witness, classifier, continuation, package, and name-certificate rows.

For the forward direction, read that budget packet through Definition 57.324. Then apply Definition 57.325 to identify the positive public distance row with the finite apartness data available to a metric-facing consumer.

For the reverse direction, start with the finite apartness packet named by the same two theorems. The rows already match the carrier fields: endpoint histories, positive dyadic or apartness data, hsame transports, Cont routes, Pkg provenance, and the NameCert ledger. Repacking these rows with the carrier's displayed distance witness gives the positive component of the public real-distance carrier.

Finally use Definition 57.327 for endpoint reversal. Symmetry changes only the orientation of the two endpoint histories, so the same finite packet remains the public positive-distance witness and no trichotomy, completion, quotient-real equality, open-cover compactness, or host equality is added. □

Theorem 57.180 (MetricSpace apartness positive-distance triangle boundary)**.** *Let x, y, and z be constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints read through the* $\mathsf{Met}^{\uparrow}$ *real-distance public surface. Suppose the adjacent positive distance reads for (x, y) and (y, z) are represented by the finite* RealAlgOrder *apartness packets isolated in Definition 57.315. Then the triangle-boundary consumer read for the outer pair (x, z) factors through exactly those endpoint* BHist *histories, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness rows, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the metric triangle ledger of Definition 57.313.*

The resulting consumer row exports only this visible finite boundary. It does not export trichotomy, open-cover compactness, completed-real equality, or a host metric-space structure.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceApartnessPositiveDistanceTriangleBoundary`

Proof. Apply Definition 57.315 to the adjacent packets (x, y) and (y, z). Each read gives the same kind of finite public data: endpoint histories, positive dyadic or apartness rows, displayed hsame transports, continuation routes, package provenance, and local name-certificate rows.

Use Definition 57.327 whenever an adjacent endpoint orientation must be read from the opposite side. This changes only the order in which the two endpoint histories are presented; it does not add trichotomy, completed-real equality, compactness, or host metric structure.

Finally apply Definition 57.313. Its triangle row is already a visible-context ledger over the public distance witness, so the two adjacent positive packets repack as the outer triangle-boundary consumer read for (x, z). The repacking uses only the displayed BHist, hsame, Cont, Pkg, NameCert, and metric-law rows listed in the statement. □

Theorem 57.181 (MetricSpace SOne RealAlgOrder consumer factorization)**.** *Every* $\mathsf{S}^{1\uparrow}$ *metric consumer read whose two real coordinates are read through the constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary factors through the finite rows of Definition 57.311. The only readable rows are the endpoint* BHist *rows of the two coordinates, the displayed* MetricDistanceWitness *row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness rows supplied by the* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary, the* hsame *transport rows,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the triangle ledger of Definition 57.313.*

Thus the $\mathsf{Met}^{\uparrow}$ *consumer surface seen from* $\mathsf{S}^{1\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *is a finite BEDC packet. It does not include trichotomy, completed-real equality, open-cover compactness, or a host metric-space structure.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceSoneRealalgorderConsumerFactorization`
↪ variant `BEDC.Derived.MetricUp.MetricspaceS1RealalgorderConsumerFactorization`

Proof. Project the admitted circle point through Definition 58.59. The public bridge exposes only the two $\mathsf{Real}^{\uparrow}$ component rows, their product ledger, the unit-equation row, and the displayed hsame, Cont, Pkg, and NameCert rows.

Apply Definition 57.315 to each constant-rational component read. This replaces every positive metric-distance demand by the visible $\mathsf{RealAlgOrder}^{\uparrow}$ apartness packet, with endpoint histories, positive dyadic or apartness data, and the same transport and package rows as in the carrier.

Next use Definition 57.316. Its triangle-boundary row is exactly the finite metric-law ledger over the positive packets, so the $\mathsf{S}^{1\uparrow}$ metric consumer has no additional distance coordinate beyond the carrier and public-law rows.

Finally compare with Definition 58.61. The observation boundary already prevents quotient-circle, angle, compactness, connectedness, and metric-completeness rows from entering the read. Since the $\mathsf{RealAlgOrder}^{\uparrow}$ correspondence contributes no trichotomy or completed-real equality and

the $\mathsf{Met}^{\uparrow}$ law package contributes no host metric space, the resulting consumer factors through exactly the finite packet listed in the statement. □

Theorem 57.182 (MetricSpace real-distance scoped certificate)**.** *The finite* $\mathsf{Met}^{\uparrow}$ *scoped certificate for the open real-algebra and geometry phase consists exactly of the real-distance carrier of Definition 57.311, the public law package of Definition 57.313, the open-phase consumer interface of Definition 57.314, the* RealAlgOrder *positive-distance correspondence of Definition 57.315, and the apartness triangle boundary of Definition 57.316.*

Its rows are only the displayed BHist *endpoint histories,* hsame *transport rows,* Cont *routes,* Pkg *provenance, local* NameCert *registrations, the public metric-law rows, and the displayed* RealAlgOrder *positive-distance budget rows. The certificate exports no host metric-space primitive, quotient-real equality, completion, open-cover compactness, or trichotomy.*

Proof. Begin with the carrier of Definition 57.311. It supplies the endpoint histories, the displayed distance witness, the transport and continuation routes, package provenance, and the local name-certificate row.

Attach the public law package of Definition 57.313 and read it through Definition 57.314. This gives the open phase only the finite law rows for symmetry, separation, and triangle composition over the same carrier fields.

Next insert the RealAlgOrder side by Definition 57.315. That theorem identifies the positive-distance consumer row with the displayed apartness budget packet, so no second positivity primitive or completed-real coordinate is introduced.

Finally compose with Definition 57.316. The triangle-boundary read uses the same endpoint histories, positive-distance budget rows, hsame, Cont, Pkg, and NameCert data already listed above. Since none of the cited components contains a host metric, quotient-real equality, completion, open-cover compactness, or trichotomy row, the assembled scoped certificate has no coordinate from which such data could be exported. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealDistanceScopedCertificate`

Theorem 57.183 (MetricSpace RealAlgOrder triangle consumer exhaustion)**.** *Every* $\mathsf{Met}^{\uparrow}$ *triangle-consumer read whose endpoints are constant rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoints and whose distance data are read through the real-distance carrier of Definition 57.311 is exhausted by the finite rows already displayed in that carrier, the public law package of Definition 57.313, the positive-distance correspondence of Definition 57.315, and the apartness triangle boundary of Definition 57.316.*

Equivalently, a triangle consumer over endpoints (x, y, z) *can read only the three endpoint* BHist *histories, the two adjacent positive* $\mathsf{RealAlgOrder}^{\uparrow}$ *packets, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the finite metric-law triangle ledger. It has no coordinate for trichotomy, quotient-real equality, completion, open-cover compactness, or an ambient host metric-space structure.*

Proof. Project the adjacent positive-distance reads for (x, y) and (y, z) through Definition 57.315. The projection exposes only endpoint histories, positive dyadic or apartness rows, transport rows, continuation routes, package provenance, and local name-certificate data.

Orient the two adjacent endpoint packets by Definition 57.327. Endpoint reversal changes only the order of the displayed unary histories, so it cannot introduce trichotomy, quotient-real equality, completion, compactness, or a host metric-space coordinate.

Compose the oriented packets through Definition 57.316. That boundary theorem is the public triangle read of Definition 57.313, hence the outer consumer is formed from the same two adjacent positive packets and the same finite triangle ledger.

Finally compare the assembled read with Definition 57.318. The scoped certificate lists every exported coordinate of the real-distance public surface. Since the triangle construction above uses only those rows, no additional coordinate remains for a mature metric-space primitive or completed real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderTriangleConsumerExhaustion`

Theorem 57.184 (MetricSpace RealAlgOrder SOne distance route exhaustion)**.** *For any accepted* $\mathsf{S}^{1\uparrow}$ *metric consumer whose coordinates are read through constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoints and whose distance request uses the carrier of Definition 57.311, the public read is exhausted by the finite route consisting of the two* $\mathsf{S}^{1\uparrow}$ *coordinate histories, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness or positive dyadic rows, the* MetricDistanceWitness *rows, endpoint and result* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the* $\mathsf{Met}^{\uparrow}$ *triangle ledger.*

This route exports no angle parametrization, quotient-circle equality, completed-real equality, open-cover compactness, trichotomy, or host metric-space primitive.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderSoneDistanceRouteExhaustion`

Proof. First apply Definition 57.317. It reduces the accepted $\mathsf{S}^{1\uparrow}$ metric read to the real-distance carrier of Definition 57.311, so the only available coordinate histories are the two visible $\mathsf{S}^{1\uparrow}$ histories and their constant-rational endpoint rows.

Next use Definition 57.315 on the distance request. The positive component of the carrier is exactly the visible $\mathsf{RealAlgOrder}^{\uparrow}$ apartness or positive dyadic packet, together with the same hsame, Cont, Pkg, and local NameCert rows.

For triangle-facing consumers, compose with Definition 57.316 and Definition 57.319. These results identify the remaining public metric-law data with the finite $\mathsf{Met}^{\uparrow}$ triangle ledger over the displayed distance witnesses and endpoint transports.

Finally compare with the $\mathsf{S}^{1\uparrow}$ observation boundary used in Definition 57.317. That boundary excludes angle parametrization, quotient-circle equality, completed real equality, open-cover compactness, trichotomy, and host metric-space primitives from the consumer surface, so the finite route listed in the statement is exhaustive. □

Theorem 57.185 (Metric RealAlgOrder SOne equation-distance factorization)**.** *Every accepted* $\mathsf{S}^{1\uparrow}$ *equation-distance consumer whose component reads pass through* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness or positive dyadic rows and whose metric request uses Definition 57.311 factors through exactly the finite rows listed in Definition 57.320. The same route supplies the equation-consumer boundary and excludes angle parametrization, quotient-circle equality, completed-real equality, trichotomy, open-cover compactness, and host metric primitives.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderSoneEquationDistanceFactorization`

Proof. First apply Definition 57.317. It places the accepted $\mathsf{S}^{1\uparrow}$ equation-distance read on the real-distance carrier, so the visible component histories are the two displayed circle coordinates and their constant-rational $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint rows.

Next use Definition 57.315. The positive-distance component is read only through apartness or positive dyadic rows, the distance witness, endpoint transports, continuation routes, package provenance, and local naming certificates.

For metric-law consumers, compose with Definition 57.319. This keeps the triangle-facing read on the same distance witnesses and the same finite $\mathsf{Met}^{\uparrow}$ ledger.

Finally compare the resulting packet with Definition 57.320. Its displayed BHist, hsame, Cont, Pkg, and NameCert rows already list every coordinate available to the equation consumer, leaving no coordinate for angle parameters, quotient or completed equality, trichotomy, open-cover compactness, or host metric primitives. □

Theorem 57.186 (Metric RealAlgOrder SOne equation-distance symmetry)**.** *Let an accepted* $\mathsf{S}^{1\uparrow}$ *equation-distance consumer factor through constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *coordinate endpoints and the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311. Then the coordinate-swapped equation-distance read factors through the same finite route after coordinate-pair determinacy,* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness metric symmetric separation, and* $\mathsf{Met}^{\uparrow}$ *route exhaustion. The swapped read introduces no angle parametrization, quotient-circle equality, completed-real equality, trichotomy, open-cover compactness, or host metric primitive.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderSoneEquationDistanceSymmetry`

Proof. Start with the original accepted equation-distance consumer and apply Definition 57.321. The factorization exposes only the two $\mathsf{S}^{1\uparrow}$ coordinate histories, their constant-rational $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint rows, the real-distance witness, endpoint transports, continuation routes, package provenance, and local name-certificate rows.

For the coordinate swap, use Definition 58.25. The determinacy statement identifies the swapped pair as the same two circle coordinate histories with their order reversed, so no angle parameter or quotient-circle equality is added to the read.

Apply Definition 57.327 to the endpoint rows. This reverses the apartness or positive dyadic separation packet inside $\mathsf{RealAlgOrder}^{\uparrow}$ while preserving the same endpoint histories, the same hsame transports, and the same finite package and certificate data.

Finally compare the swapped packet with Definition 57.320. That exhaustion theorem already lists the complete $\mathsf{Met}^{\uparrow}$ route available to the equation-distance carrier. Hence the coordinate-swapped read stays on the same finite route and leaves no coordinate for completed-real equality, trichotomy, open-cover compactness, or host metric primitives. □

Theorem 57.187 (MetricSpace RealAlgOrder SOne positive-boundary equivalence)**.** *Let constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints be accepted by the* $\mathsf{S}^{1\uparrow}$ *component-and-equation consumer through the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311. Then the* $\mathsf{RealAlgOrder}^{\uparrow}$ *positive-boundary row, the* $\mathsf{Met}^{\uparrow}$ *positive-distance row, and the* $\mathsf{S}^{1\uparrow}$ *equation-separation row are equivalent finite BEDC readings of the same packet. The shared packet consists exactly of endpoint* BHist *rows, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness row,* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, the metric distance witness, and the* $\mathsf{S}^{1\uparrow}$ *component and unit-equation rows. It does not include trichotomy, quotient-real equality, angle quotient, completed-real equality, open-cover compactness, connectedness, or host metric primitives.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderSonePositiveBoundaryEquivalence`

Proof. First apply Definition 57.315. This identifies the $\mathsf{RealAlgOrder}^{\uparrow}$ positive-boundary read with the $\mathsf{Met}^{\uparrow}$ positive-distance row through the visible endpoint histories, apartness or positive dyadic data, transport rows, continuation routes, package provenance, local naming certificates, and the metric distance witness.

Next apply Definition 57.317. The $\mathsf{S}^{1\uparrow}$ component-and-equation consumer can only enter through that real-distance public interface, so its component rows and unit-equation rows read the same endpoints and distance witness rather than a separate angle or quotient-circle carrier.

The route is exhaustive by Definition 57.320. Hence any consumer claiming one of the three boundary views must use the displayed BHist, hsame, Cont, Pkg, NameCert, distance, component, and unit-equation rows.

Finally compare with Definition 38.65 and Definition 58.67. These two boundary theorems give the reverse readings from the $\mathsf{RealAlgOrder}^{\uparrow}$–$\mathsf{Met}^{\uparrow}$ side and from the $\mathsf{S}^{1\uparrow}$ equation consumer side. Together they show that the positive-boundary, positive-distance, and equation-separation rows are three readings of one finite certificate packet, with no remaining coordinate for trichotomy, quotient-real equality, angle quotient, completed-real equality, open-cover compactness, connectedness, or host metric primitives. □

Theorem 57.188 (MetricSpace SOne distance source boundary)**.** *Every accepted* $\mathsf{S}^{1\uparrow}$ *packet over the* $\mathsf{Real}^{\uparrow}$ *product carrier reads* $\mathsf{Met}^{\uparrow}$ *distance data only through the finite rows of Definition 57.311 and the public law package of Definition 57.313. Its circle-side source is constrained by Definition 58.59, Definition 58.61, and Definition 58.62. Its* $\mathsf{RealAlgOrder}^{\uparrow}$ *side is constrained by Definition 57.324 and Definition 38.63.*

Thus a $\mathsf{S}^{1\uparrow}$ *consumer of real-distance data can read only the two* $\mathsf{Real}^{\uparrow}$ *component rows, the product-carrier ledger, the unit-equation row, componentwise* hsame *classifiers, equation transport, finite-window* RegSeqRat *stream-name and real-seal rows, the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness budget rows, the metric endpoint histories,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the public metric-law rows. The boundary exports no angle quotient, covering-space quotient, host metric function, completion row, compactness row, connectedness row, or topology generator.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceSoneDistanceSourceBoundary`
↪ variant `BEDC.Derived.MetricUp.MetricspaceSOneDistanceSourceBoundary`

Proof. Project the accepted $\mathsf{S}^{1\uparrow}$ packet through Definition 58.59. This exposes the two $\mathsf{Real}^{\uparrow}$ component rows, the product carrier, the unit-equation witness, the componentwise classifier rows, and equation transport, and excludes the circle-specific quotient and topology rows named in the statement.

Refine the two real components through Definition 58.61 and Definition 58.62. The metric-facing source rows are then finite-window RegSeqRat and real-seal observations attached to the same public certificate, rather than an ambient circle metric or a hidden completion object.

Pass the real-algebra side through Definition 57.324 and Definition 38.63. These two boundaries keep the apartness and unit-equation reads tied to the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint and budget rows, so the metric source has no trichotomy, quotient-real equality, host metric, or Real completeness coordinate.

Repack the resulting endpoint rows, budget rows, transport rows, continuation routes, provenance rows, and local certificate rows as the carrier of Definition 57.311. Attach Definition 57.313 and read the package through Definition 57.314. The open real-algebra and geometry phase therefore consumes exactly the finite real-distance carrier and its public law rows, with no additional metric, topology, compactness, connectedness, completion, angle, or covering quotient data. □

Theorem 57.189 (Metric distance witness semantic name certificate)**.** *For unary endpoint histories* x, y*, the concrete distance-witness predicate* $d \mapsto \mathsf{MetricDistanceWitness}(x, y, d)$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are the same concrete witness predicate and whose classifier is* hsame*.*

Proof. The carrier witness is the concrete distance history $x \cdot y$. Unary append closure supplies its unary carrier witness, and the continuation field is the reflexive ledger $\mathsf{Cont}(x, y, x \cdot y)$. The

classifier fields are the reflexive, symmetric, and transitive laws for hsame. Carrier transport is history-sameness transport of the concrete distance witness. □

Lean4 ✓ BEDC.Derived.MetricUp.MetricDistanceWitness semanticNameCert

Definition 57.190 (MetricSpace ledger policy)**.** *Record carrier witnesses and the real ledger of each distance value.*

Lean4 (definition) BEDC.Derived.MetricUp.MetricLedgerPolicy
Lean4 ✓ BEDC.Derived.MetricUp.MetricLedgerPolicy rows

Definition 57.191 ($\mathsf{Met}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Met}^{\uparrow}$*.*

Lean4 (definition) BEDC.Derived.MetricUp.MetricCertificateObligations

Definition 57.192 (Metric ball budget predicate)**.** *For a certified* $\mathsf{Met}^{\uparrow}$ *setup on a carrier* X *with distance* $d : X \times X \to \mathsf{Real}^{\uparrow}$*, a ball-budget structure supplies an ordered-real comparison* $\leq_{\mathsf{R}}$ *on* $\mathsf{Real}^{\uparrow}$*, an addition operation* $+$ *on* $\mathsf{Real}^{\uparrow}$*, a transitivity field for* $\leq_{\mathsf{R}}$*, and a two-sided addition-monotonicity field saying that* $a \leq_{\mathsf{R}} a'$ *and* $b \leq_{\mathsf{R}} b'$ *entail* $a + b \leq_{\mathsf{R}} a' + b'$*. Relative to this chosen budget structure, the closed ball-budget predicate is*

$$\mathsf{MetricBallBudget}_d(x, y; r) :\Longleftrightarrow d(x, y) \leq_{\mathsf{R}} r.$$

Lean4 (definition) BEDC.Derived.MetricUp.MetricBallBudget

Theorem 57.193 (Metric ball budget composition)**.** *Let* X *be a carrier equipped with the* $\mathsf{Met}^{\uparrow}$ *certificate obligations of Definition 57.231, and let* $d : X \times X \to \mathsf{Real}^{\uparrow}$ *be its certified distance. Assume also the ball-budget ordered-real fields of Definition 57.232: the comparison* $\leq_{\mathsf{R}}$*, addition* $+$*, budget-transitivity, and two-sided budget-addition-monotonicity. For* $x, y, z : X$ *and budgets* $r, s : \mathsf{Real}^{\uparrow}$*,*

$$\mathsf{MetricBallBudget}_d(x, y; r) \Longrightarrow \mathsf{MetricBallBudget}_d(y, z; s) \Longrightarrow \mathsf{MetricBallBudget}_d(x, z; r + s).$$

Equivalently,

$$d(x, y) \leq_{\mathsf{R}} r \Longrightarrow d(y, z) \leq_{\mathsf{R}} s \Longrightarrow d(x, z) \leq_{\mathsf{R}} r + s.$$

Proof. Unfold Definition 57.232. The two premises are the budget comparisons $d(x, y) \leq_{\mathsf{R}} r$ and $d(y, z) \leq_{\mathsf{R}} s$. The triangle-inequality field of Definition 57.5 gives

$$d(x, z) \leq_{\mathsf{R}} d(x, y) + d(y, z).$$

Apply the supplied two-sided budget-addition-monotonicity field to the two budget premises. This gives

$$d(x, y) + d(y, z) \leq_{\mathsf{R}} r + s.$$

Apply the supplied budget-transitivity field to the triangle inequality and the displayed budget bound. This yields $d(x, z) \leq_{\mathsf{R}} r + s$. Refolding Definition 57.232 gives the closed ball-budget conclusion. □

Lean4 ✓ BEDC.Derived.MetricUp.MetricBallBudget composition

Theorem 57.194 (Metric distance witness standard bridge)**.** *For the public* $\mathsf{Met}^{\uparrow}$ *distance-witness surface over unary* BHist *endpoints* x, y*, there is a conservative standard interpretation whose BEDC carrier is*

$$\{\, d \mid \mathsf{MetricDistanceWitness}(x, y, d) \,\},$$

whose classifier is hsame*, whose forward row reads such a witness as the finite append-distance presentation between the two unary endpoints, and whose reverse row reconstructs the BEDC witness from that finite append presentation. The bridge is only for this generated distance-witness surface; it does not assert arbitrary host real metrics.*

Proof. The carrier and classifier come from Definition 57.229: the public BEDC objects are exactly the d satisfying $\mathsf{MetricDistanceWitness}(x, y, d)$, and the classifier is hsame.

For the forward row, apply Definition 57.13 to read the visible unary context as the finite append-distance presentation between the endpoints. Definition 57.133 ensures that this presentation is determined by the visible context, while Definition 57.9 transports the same presentation across the symmetric endpoint order.

For the reverse row, read the finite append-distance presentation back through Definition 57.13. The resulting BEDC witness is classified by the same hsame row, and Definition 57.233 supplies the budget-compatible composition law needed by the public distance-witness surface.

Thus the two rows form a conservative paper-side standard bridge for the generated MetricDistanceWitness surface only. No completeness, compactness, topology, or arbitrary host-real metric structure is introduced. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricUp StdBridge`

Definition 57.195 (MetricSpaceUp public distance surface)**.** *The* $\mathsf{Met}^{\uparrow}$ *public distance surface is the finite* BHist *surface whose carrier rows are the unary endpoint histories* x, y*, the displayed* $\mathsf{MetricDistanceWitness}(x, y, d)$ *row, the ball-budget row* $\mathsf{MetricBallBudget}_d(x, y; r)$*, the classifier row* hsame*, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows used by Definition 57.232. A public distance read of* $\mathsf{Met}^{\uparrow}$ *is a projection of this surface and no larger host metric-space object.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface hsame transport`

Theorem 57.196 (MetricSpaceUp public distance surface exhaustion)**.** *Every public distance read of* $\mathsf{Met}^{\uparrow}$ *projects exactly to the* BHist *endpoint rows, the* MetricDistanceWitness *row, the* MetricBallBudget *row, the* hsame *classifier row, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows of Definition 57.235. Conversely, any such finite family repacks as the same public distance surface.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface exhaustion`

Proof. Project a public read through the distance-witness semantic certificate of Definition 57.229. This fixes the BHist endpoints, the concrete $\mathsf{MetricDistanceWitness}(x, y, d)$ row, and the hsame classifier row.

Next apply Definition 57.233. The only budget rows available to the public read are the displayed $\mathsf{Real}^{\uparrow}$ comparison, addition, transitivity, and monotonicity fields used to compose ball-budget witnesses.

Finally use Definition 57.234 and Definition 57.133. The bridge reads the same finite witness surface in the standard direction, and visible context determinacy prevents an additional host distance row from entering the packet. Repacking the projected rows gives exactly Definition 57.235. □

Theorem 57.197 (MetricSpaceUp visible-context distance consumer surface)**.** *The named visible-context consumers* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *may read a* $\mathsf{Met}^{\uparrow}$ *distance only through the visible-context distance witness, ball-budget, and classifier rows of Definition 57.235. Their public use of the metric interface does not add a host metric carrier, a separate topology generator, or an analytic completeness or compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface visible context consumer surface`

Proof. Start with the public surface of Definition 57.235. The only exported distance data are the $\mathsf{MetricDistanceWitness}$ row, the ball-budget row, the hsame classifier, and the finite Cont provenance over the displayed $\mathsf{Real}^{\uparrow}$ budget rows.

Apply Definition 57.133. Any visible-context consumer that reads a distance witness receives the determined finite witness row rather than an external host distance function.

Use Definition 57.233 for all budget propagation and Definition 57.234 for the standard distance-facing reading. These two rows exhaust the consumer-facing surface, so the listed downstream chapters can consume only the visible-context witness, budget, and classifier rows. □

Theorem 57.198 (MetricSpace visible-context public certificate)**.** *The public* $\mathsf{Met}^{\uparrow}$ *certificate is exactly the finite* BHist *endpoint rows, the* $\mathsf{MetricDistanceWitness}$ *row, the* $\mathsf{MetricBallBudget}$ *row, the* hsame *classifier row, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows of Definition 57.235. Every downstream read by* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *factors through the visible-context consumer surface of Definition 57.237, so the certificate exports no host metric-space carrier, no separate topology generator, and no analytic completeness or compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface visible context public certificate`

Proof. First project the visible distance read through Definition 57.229. This fixes the endpoint BHist rows, the displayed $\mathsf{MetricDistanceWitness}$ row, and the hsame classifier that names the distance witness.

Next compose the budget rows by Definition 57.233. The only available comparison, addition, transitivity, and monotonicity data are the displayed $\mathsf{Real}^{\uparrow}$ budget rows already present in Definition 57.235.

Then read the standard-facing side through Definition 57.234. The bridge has the same finite witness surface, so it does not introduce a host metric function or an ambient metric-space object.

Finally apply Definition 57.237. The listed downstream chapters read the metric interface only through this visible-context surface. Combining these rows gives exactly the public $\mathsf{Met}^{\uparrow}$ certificate and no larger object. □

Primitive boundary. $\mathsf{Met}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Theorem 57.199 (Metric distance witness empty-boundary visible-context distance empty)**.** *For histories* p, q, d*, a visible-context distance witness whose central left and right endpoints are both empty identifies the central distance with the empty history and gives zero central distance depth:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Apply Definition 57.13 to peel away the visible prefix and suffix. The remaining central witness is $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d)$. The empty-left boundary theorem identifies d with e; case analysis on that classifier gives the zero depth equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context distance empty`

Theorem 57.200 (Metric prefix-independent symmetric zero-depth empty-boundary forward)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, suppose the two symmetric visible-context distance witnesses*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q), \qquad \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q')$$

hold. If $\mathsf{MetricDistanceDepth}(d_{xy}) = 0$*, then the original visible context carries the empty-boundary witness*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q).$$

Proof. Peel both visible-context witnesses to their central witnesses. Symmetric classification compares d_{xy} and d_{yx}. Case analysis on the zero-depth central distance leaves only $d_{xy} = \mathsf{e}$; the empty-distance visible-context readback then identifies the two central endpoints as empty. Reattaching the original prefix and suffix gives the displayed empty-boundary witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth empty boundary forward`

Theorem 57.201 (Metric empty-boundary visible context append unit laws)**.** *For histories p, q, d, l, r, a visible-context distance witness whose central endpoints are both empty makes the central distance act as a left and right append unit:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{append}(d, r) = r \ \wedge \ \mathsf{append}(l, d) = l.$$

Proof. The empty-boundary distance theorem identifies d with e. Case analysis on that classifier reduces the two displayed equations to the left and right unit laws for finite-kernel append. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context append unit laws`

Theorem 57.202 (Metric empty-boundary visible context unit splice commute)**.** *For histories $p, q, d, l, r, out_L, out_R$, an empty-boundary visible-context distance witness makes left and right continuation through the central distance commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{Cont}(l, d, out_L) \Longrightarrow \mathsf{Cont}(d, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The empty-boundary distance theorem identifies d with e. The left continuation then has right-unit result $out_L \mathsf{hsame} l$, and the right continuation has left-unit result $out_R \mathsf{hsame} r$. Substituting these two unit identifications reduces the displayed splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context unit splice commute`

Theorem 57.203 (Metric visible-context empty-distance unit splice commute)**.** *For histories $p, q, x, y, l, r, out_L, out_R$, a visible-context empty-distance witness makes left continuation through x and right continuation through y commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}\mathsf{e}){\cdot}q) \Longrightarrow \mathsf{Cont}(l, x, out_L) \Longrightarrow \mathsf{Cont}(y, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The visible-context empty-distance exactness theorem identifies both central endpoints x and y with e. The left continuation then has right-unit result $out_L\mathsf{hsame}l$, and the right continuation has left-unit result $out_R\mathsf{hsame}r$. Substituting these unit identifications reduces the splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance unit splice commute`

Theorem 57.204 (Metric empty-boundary visible context left-unit witness)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness turns any continuation $\mathsf{Cont}(d, r, out)$ into a metric witness from the empty source:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(d, r, out) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{e}, r, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a left-unit continuation from e to r. The empty source is unary, the target is unary by hypothesis, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context left unit witness`

Theorem 57.205 (Metric empty-boundary visible context right-unit witness)**.** *For histories p, q, d, l, out, an empty-boundary visible-context distance witness turns any continuation $\mathsf{Cont}(l, d, out)$ into a metric witness into the empty target:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{Cont}(l, d, out) \Longrightarrow \mathsf{MetricDistanceWitness}(l, \mathsf{e}, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a right-unit continuation into e. The source is unary by hypothesis, the empty target is unary, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context right unit witness`

Theorem 57.206 (Metric empty-boundary visible context continuation depth add)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness lets a continuation through d splice into the direct continuation from l to r, so the output depth is the sum of the endpoint depths:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid,$$

Proof. The empty-boundary continuation splice removes the central empty distance from the two-step ledger and gives $\mathsf{Cont}(l, r, out)$. Together with the unary endpoint hypotheses, unary continuation closure supplies the output carrier, so these fields form a concrete metric distance witness from l to r. Metric distance depth additivity then gives the displayed equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation depth add`

Theorem 57.207 (Metric empty-boundary visible context continuation witness depth package)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness, unary visible endpoints, and the two-step continuation through d determine both the visible-endpoint metric witness and its depth ledger:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}\end{aligned}$$

Proof. The witness conjunct is supplied by Definition 57.272 from the empty-boundary witness, the endpoint unary histories, and the two continuation rows.

The depth conjunct is supplied by Definition 57.283 from the same hypotheses. Conjoining the two conclusions gives the displayed package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation witness depth`

Theorem 57.208 (Metric empty-boundary visible context canonical splice package)**.** *For histories $p, q, d, l, r, mid, out, out'$, an empty-boundary visible-context distance witness, unary endpoint carriers, a two-step continuation through d, and a displayed direct continuation from l to r determine the canonical splice package:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \mathsf{Cont}(l, r, out') \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{hsame}(out, out') \wedge \\ &\quad \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r).\end{aligned}$$

Proof. The witness row is supplied by Theorem 57.272 from the empty-boundary visible-context witness, the unary carriers, and the two displayed continuation rows.

The comparison row is supplied by Theorem 57.273, which compares the spliced output with the displayed direct continuation $\mathsf{Cont}(l, r, out')$.

The depth row is supplied by Theorem 57.283. Conjoining these three rows gives the displayed certificate package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context canonical splice package`

Theorem 57.209 (Metric distance witness standard boundary bridge)**.** *For the concrete* MetricDistanceWitness *interface on unary* BHist *endpoint witnesses, the paper-side bridge is the conservative boundary interpretation assembled from the semantic NameCert, visible-context exactness, the left and empty boundary readback rows, and the zero-depth identity-path row. It identifies exactly the BEDC boundary-distance witness rows with the corresponding standard boundary-metric reading, and back again, for the concrete* BHist *witness surface.*

Proof. The semantic carrier and classifier are the concrete MetricDistanceWitness predicate and hsame classifier supplied by Definition 57.229. Thus the bridge has no separate host carrier: its public objects are exactly the displayed BHist distance witnesses.

The boundary readback is supplied in the two directions by Definition 57.96 and Definition 57.97. These rows say that the left and empty BEDC boundary surfaces read as the same standard boundary-distance data, and that the standard reading returns to the same BEDC rows.

Finally, Definition 57.310 provides the zero-depth symmetric identity-path row used by the standard boundary metric reading. Together these rows give a conservative two-way boundary interpretation for the concrete witness surface only. No arbitrary host metric space, completeness, compactness, or topology-induced-by-metric bridge is being claimed. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness standard boundary bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$.

Constructive story: A metric distance witness is built from two unary endpoint histories and one unary distance history, with the continuation ledger showing that the distance is exactly the append result from the first endpoint through the second. The classifier on distance results is hsame, so two witnesses are compared by transporting their endpoint and distance histories through the same continuation data rather than by an external real-valued distance function. The source, pattern, and ledger predicates coincide for the semantic certificate: the source accepts concrete distance witnesses, the pattern records empty-distance exactness, symmetry, prefix and suffix visibility, triangle append rows, ball-budget composition, and boundary readback, and the ledger keeps the endpoint carriers plus the distance continuation and structural-depth calculation visible. Since unary append gives a witness distance $x \cdot y$ for unary endpoints x, y, hsame supplies reflexivity, symmetry, and transitivity, and case transport preserves the witness predicate, the five name-certificate obligations hold on the displayed BEDC surface. A history d lies in the signature gap of a $\mathsf{Met}^\uparrow$ token p when the source-domain accepts d as a distance witness for the chosen endpoints, the signature relation pairs it with an answer distance and evidence ledger, and token introduction names that answer as p; concretely, d is unary, its continuation ledger is the endpoint append, and its depth is the sum of the endpoint depths recorded by p. The checked standard bridge reads this finite distance witness as the standard boundary metric datum and reads it back through the same empty, prefix, suffix, and symmetric-zero rows, without changing the carrier. The chapter registers this continuation-distance construction under the name $\mathsf{Met}^\uparrow$; metric laws are certified append and boundary facts over unary histories, with no arbitrary host metric space, completeness theorem, or induced topology introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, maturePackageClosed)

Scope closed: The closed scope is the MetricDistanceWitness semantic certificate, ball-budget composition, the standard distance-witness bridge, the standard boundary bridge, and the named cross-chapter consumers in $\mathsf{TotallyBounded}^\uparrow$, $\mathsf{CompleteMetric}^\uparrow$, $\mathsf{CompactMetric}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, and the compact uniform-continuity capstone.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.MetricUp.MetricUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close arbitrary host metric spaces, general completeness, compactness, topology induced by a metric, or any bridge stronger than the checked BEDC schema bridge.

Upgrade path: no further upgrade – chapter at top of formal axis

Remark 57.210 (Zero-axiom encoding note)**.** *Replaces decidable equality with apartness and explicit real-modulus inequalities.*

Theorem 57.211 (Metric distance witness empty-source visible-distance target exactness)**.** *For histories $target, d$, a concrete distance witness from the empty source whose distance ledger is visibly one-headed identifies the target with that same one-headed ledger and exposes unary evidence for the ledger tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, target, \mathsf{E1}(d)) \Longrightarrow target = \mathsf{E1}(d) \wedge \mathsf{UnaryHistory}(d).$$

Proof. Apply empty-left exactness to obtain unary evidence for $target$ and the classifier $\mathsf{hsame}(\mathsf{E1}(d), target)$. One-constructor inversion on this classifier presents $target$ as $\mathsf{E1}(d)$ up to the exposed tail equality. Substitution into the unary target witness and unary one-step inversion give $\mathsf{UnaryHistory}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty source e1 distance target exactness`

Theorem 57.212 (Metric distance witness empty-target visible-distance source exactness)**.** *For histories $source, d$, a concrete distance witness into the empty target whose distance ledger is visibly one-headed identifies the source with that same one-headed ledger and exposes unary evidence for the ledger tail:*

$$\mathsf{MetricDistanceWitness}(source, \mathsf{e}, \mathsf{E1}(d)) \Longrightarrow source = \mathsf{E1}(d) \wedge \mathsf{UnaryHistory}(d).$$

Proof. Apply empty-right exactness to obtain unary evidence for *source* and the classifier $\mathsf{hsame}(\mathsf{E1}(d), \mathit{source})$. One-constructor inversion on this classifier presents *source* as $\mathsf{E1}(d)$ up to the exposed tail equality. Substitution into the unary source witness and unary one-step inversion give $\mathsf{UnaryHistory}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty target e1 distance source exactness`

Theorem 57.213 (Metric nonempty distance has a nonempty endpoint)**.** *For histories* x, y, d : BHist*, if* $\mathsf{MetricDistanceWitness}(x, y, d)$ *holds and* d *is not history-same as* e*, then at least one endpoint is not history-same as* e*:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to ((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False})).$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint nonempty`

Proof. Case-split the two endpoint histories. A visible endpoint constructor is immediately disjoint from e, giving the required side of the disjunction. In the remaining empty/empty case, the continuation field of the metric witness identifies the distance result with e, contradicting the assumed nonempty distance. □

Theorem 57.214 (Metric nonempty distance endpoint positive depth)**.** *For histories* x, y, d : BHist*, if* $\mathsf{MetricDistanceWitness}(x, y, d)$ *holds and* d *is not history-same as* e*, then at least one endpoint has positive structural metric depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to (0 < \mathsf{MetricDistanceDepth}(x) \vee 0 < \mathsf{MetricDistanceDepth}(y))$$

Proof. Apply the nonempty-distance endpoint theorem to obtain a nonempty endpoint. The positive-depth classifier for metric histories converts that endpoint nonemptiness into positive structural depth on the corresponding side of the disjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint positive depth`

Theorem 57.215 (Metric positive distance e-one shape)**.** *For histories* x, y, d : BHist*, if* $\mathsf{MetricDistanceWitness}(x, y$ *holds and the structural depth of* d *is not zero, then* d *is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to \exists \mathit{tail},\ d = \mathsf{E1}(\mathit{tail}) \wedge \mathsf{UnaryHistory}(\mathit{tail}).$$

Proof. Unpack the metric distance witness to obtain the unary carrier for the distance history d. Case analysis on d rules out e by the positive depth hypothesis and rules out an $\mathsf{E0}$ head by unary-history inversion. The remaining $\mathsf{E1}$ case exposes its tail, and one-step unary inversion gives the unary tail witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness positive distance e1 shape`

Theorem 57.216 (Metric nonempty distance endpoint e-one cases)**.** *For histories* x, y, d : BHist*, if* $\mathsf{MetricDistanceWitness}(x, y, d)$ *holds and the distance endpoint is not history-same as* e*, then at least one endpoint is visibly one-headed with a unary tail:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d) \to (d\mathsf{hsamee} \to \mathsf{False}) \to \\ &\quad (\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)).\end{aligned}$$

Proof. First apply the endpoint non-emptiness theorem to the metric witness and the nonempty distance assumption. On whichever side is nonempty, use the endpoint's unary-history carrier from the metric witness and apply unary nonempty one-tail readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness nonempty distance endpoint e1 cases`

Theorem 57.217 (Metric empty-source nonempty distance target e-one cases)**.** *For histories* $target, dist$: BHist*, if a concrete metric distance witness starts at the empty source and its distance endpoint is not history-same as* e*, then the target endpoint is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to \exists targetTail,\ target = \mathsf{E1}(targetTail) \wedge \mathsf{UnaryHist}$$

Proof. Apply the nonempty-distance endpoint e-one cases theorem. The source-side visible branch would identify e with a one-headed history, which is impossible by constructor disjointness. The remaining branch is exactly the target e-one readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty source nonempty distance target e1 cases`

Theorem 57.218 (Metric empty-target nonempty distance source e-one cases)**.** *For histories* $source, dist$: BHist*, if a concrete metric distance witness ends at the empty target and its distance endpoint is not history-same as* e*, then the source endpoint is visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(source, \mathsf{e}, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to \exists sourceTail,\ source = \mathsf{E1}(sourceTail) \wedge \mathsf{UnaryH}$$

Proof. Apply the nonempty-distance endpoint e-one cases theorem. The target-side visible branch would identify e with a one-headed history, which is impossible by constructor disjointness. The remaining branch is exactly the source e-one readback. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty target nonempty distance source e1 cases`

Theorem 57.219 (Metric empty-boundary nonempty distance positive endpoint depth)**.** *For histories* $source, target, dist$: BHist*, a concrete metric distance witness with exactly one empty boundary and a nonempty distance endpoint forces positive structural depth on the other endpoint:*

$$\begin{aligned}&(\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to 0 < \mathsf{MetricDistanceDepth}(target)) \wedge \\ &(\mathsf{MetricDistanceWitness}(source, \mathsf{e}, dist) \to (dist\mathsf{hsamee} \to \mathsf{False}) \to 0 < \mathsf{MetricDistanceDepth}(source)).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary nonempty distance positive endpoint depth`

Proof. For the empty-source implication, apply Theorem 57.217 to the displayed witness and the nonempty distance hypothesis. The result expresses *target* as $\mathsf{E1}(targetTail)$ with a unary tail, so the structural metric depth of *target* is a successor and is therefore positive.

For the empty-target implication, apply Theorem 57.218 in the same way. The source endpoint is read back as $\mathsf{E1}(sourceTail)$ with a unary tail, and the one-headed shape gives positive structural metric depth for *source*. □

The ten-field fit is: traditional target paired empty-boundary nonzero-distance endpoint depth; BEDC source MetricDistanceWitness over BHist; carrier the surviving boundary endpoint; classifier hsame; stability unary one-headed readback and depth positivity; ledger the nonempty distance endpoint; core theorem Theorem 57.219; exactness positive depth on the nonempty boundary endpoint; standard bridge finite unary metric boundary readback; status ProofTarget.

Theorem 57.220 (Metric visible-boundary positive-depth endpoint readback)**.** *For histories $p, q, source, target, di$ BHist, a visible-context metric distance witness with exactly one empty central boundary and positive central distance depth reads back the nonempty central endpoint as one-headed and positive-depth:*

$$(\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, target \cdot q, (p \cdot dist) \cdot q) \to (\mathsf{MetricDistanceDepth}(dist) = 0 \to \mathsf{False}) \to \exists targetTail,\ target = \mathsf{E1}(targetTail) \wedge \mathsf{UnaryHistory}(targetTail) \wedge 0 < \mathsf{MetricDistanceDepth}(target)) \wedge (\mathsf{MetricDistanceWitness}(p \cdot source, \mathsf{e} \cdot q, (p \cdot dist) \cdot q) \to (\mathsf{MetricDistanceDepth}(dist) = 0 \to \mathsf{False}) \to \exists sourceTail,\ source = \mathsf{E1}(sourceTail) \wedge \mathsf{UnaryHistory}(sourceTail) \wedge 0 < \mathsf{MetricDistanceDepth}(source)).$$

Proof. For the empty central source implication, apply Definition 57.13 to peel the visible prefix p and suffix q from the displayed witness. This recovers the central witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, target, dist)$.

The positive-depth hypothesis rules out $\mathsf{MetricDistanceDepth}(dist) = 0$. By Definition 57.253, this is the same nonempty-distance input required by the empty-source e-one cases theorem. Apply Definition 57.217 to obtain $target = \mathsf{E1}(targetTail)$ with unary $targetTail$. The one-headed displayed endpoint has successor structural metric depth, so $0 < \mathsf{MetricDistanceDepth}(target)$.

For the empty central target implication, the same visible-context peeling recovers $\mathsf{MetricDistanceWitness}(source$ The same zero-depth equivalence converts positive depth of $dist$ into the nonempty-distance input, and Definition 57.218 gives $source = \mathsf{E1}(sourceTail)$ with unary $sourceTail$. The displayed one-headed source again has positive structural metric depth. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible boundary positive depth endpoint readback`

Theorem 57.221 (Metric distance witness zero-headed component impossibility)**.** *For histories x, y, d : BHist, no concrete metric distance witness has a zero-headed source, target, or distance component:*

$$(\mathsf{MetricDistanceWitness}(\mathsf{E0}(x), y, d) \to \mathsf{False}) \wedge (\mathsf{MetricDistanceWitness}(x, \mathsf{E0}(y), d) \to \mathsf{False}) \wedge (\mathsf{MetricDistanceWit}$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e0 component absurd`

Proof. Unpack the concrete witness. Its three component fields are unary-history evidence for the source, target, and distance histories. BEDC unary histories are generated only from e by E1, so each displayed E0 component contradicts unary no-zero-extension. □

Theorem 57.222 (Metric empty distance visible endpoint impossibility)**.** *For histories x, y, d, tx, ty, if a metric distance witness has a result history-same as* e*, then neither endpoint can be history-same as a visible one-headed history:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{hsame}(d, \mathsf{e}) \to (\mathsf{hsame}(x, \mathsf{E1}(tx)) \to \mathsf{False}) \wedge (\mathsf{hsame}(y, \mathsf{E1}(ty)) \to \mathsf{False}).$$

Proof. Transport the continuation result of the metric witness along the displayed distance classifier, obtaining an empty-distance witness over the same endpoints. Empty-distance exactness identifies both endpoints with e. Composing either endpoint comparison with a visible one-headed classifier contradicts the empty/e-one no-confusion theorem. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty distance visible endpoint absurd`

Theorem 57.223 (Metric empty-distance composite endpoint collapse)**.** *For histories x, y, z, d_{xy}, d_{yz}, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ *and* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$ *both have distance histories classified as* e*, then all three endpoints classify together:*

$$d_{xy}\mathsf{hsamee} \to d_{yz}\mathsf{hsamee} \to \mathsf{hsame}(x, y) \wedge \mathsf{hsame}(y, z) \wedge \mathsf{hsame}(x, z).$$

Proof. Transport each distance witness to the empty-distance case. Empty-distance exactness identifies the corresponding endpoint pair with e, and transitivity through e gives the three endpoint comparisons. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty distance composite endpoint collapse`

Theorem 57.224 (Metric e1-pair empty-distance impossibility)**.** *For histories $x, y : \mathsf{BHist}$,*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{e}) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair empty distance absurd`

Proof. Apply empty-distance exactness to the displayed metric witness. It forces the left endpoint $\mathsf{E1}(x)$ to be history-same as e, contradicting the disjointness of visible one-headed histories from the empty history. □

Theorem 57.225 (Metric e1 endpoints force e1 distance)**.** *For histories $x, y, d : \mathsf{BHist}$, a metric distance witness between two one-headed endpoints forces the distance history itself to be one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \exists d_0 : \mathsf{BHist},\ d = \mathsf{E1}(d_0) \wedge \mathsf{UnaryHistory}(d_0).$$

Proof. Case-analyze the distance history. The empty case contradicts Definition 57.224; the zero-headed case contradicts the unary carrier field of the witness. The one-headed case exposes the displayed tail, and unary inversion supplies its carrier proof. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 endpoints distance e1 shape`

Theorem 57.226 (Metric right-boundary visible context distance source)**.** *For histories $p, q, x, d : \mathsf{BHist}$, a visible-context distance witness with empty central right endpoint identifies the central distance with the visible source and gives the same structural depth:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(d, x) \wedge \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x).$$

Proof. Right-boundary visible-context exactness extracts the central classifier $\mathsf{hsame}(d, x)$. Case analysis on that classifier identifies the two histories, and the metric-depth equality follows by reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right boundary visible context distance source`

Theorem 57.227 (Metric right-boundary visible context positive-depth source shape)**.** *For histories $p, q, x, d : \mathsf{BHist}$, a visible-context distance witness with empty central right endpoint and positive central distance depth forces the central source to be visibly one-headed with a unary tail:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \Longrightarrow \exists x_0 : \mathsf{BHist},\ x = \mathsf{E1}(x_0) \wedge \mathsf{Ur}$$

Proof. Apply right-boundary visible-context exactness. It exposes unary evidence for the central source and the classifier $\mathsf{hsame}(d, x)$. If x classified as e, transitivity would classify d as e, and the zero-depth classifier would contradict the positive-depth hypothesis. Thus x is a nonempty unary history, so unary nonempty e-one tail readback supplies the required x_0. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right boundary visible context positive depth source e1 shape`

Theorem 57.228 (Metric left-boundary visible-context distance target)**.** *For histories p, q, y, d, a visible-context distance witness whose central left endpoint is empty identifies the central distance with the target and aligns their structural depths:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, y{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{hsame}(d, y) \wedge \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(y).$$

Proof. Apply left-boundary visible-context exactness to peel the visible prefix and suffix. The central empty-left witness gives $\mathsf{hsame}(d, y)$. Case analysis on that classifier reduces the displayed depth equation to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left boundary visible context distance target`

Theorem 57.229 (Metric distance witness semantic name certificate)**.** *For unary endpoint histories x, y, the concrete distance-witness predicate $d \mapsto \mathsf{MetricDistanceWitness}(x, y, d)$ carries a semantic naming certificate whose source, pattern, and ledger predicates are the same concrete witness predicate and whose classifier is* hsame.

Proof. The carrier witness is the concrete distance history $x \cdot y$. Unary append closure supplies its unary carrier witness, and the continuation field is the reflexive ledger $\mathsf{Cont}(x, y, x \cdot y)$. The classifier fields are the reflexive, symmetric, and transitive laws for hsame. Carrier transport is history-sameness transport of the concrete distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness semanticNameCert`

Definition 57.230 (MetricSpace ledger policy)**.** *Record carrier witnesses and the real ledger of each distance value.*

Lean4 (definition) `BEDC.Derived.MetricUp.MetricLedgerPolicy`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricLedgerPolicy rows`

Definition 57.231 ($\mathsf{Met}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Met}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.MetricUp.MetricCertificateObligations`

Definition 57.232 (Metric ball budget predicate)**.** *For a certified $\mathsf{Met}^{\uparrow}$ setup on a carrier X with distance $d : X \times X \to \mathsf{Real}^{\uparrow}$, a ball-budget structure supplies an ordered-real comparison $\leq_{\mathsf{R}}$ on $\mathsf{Real}^{\uparrow}$, an addition operation $+$ on $\mathsf{Real}^{\uparrow}$, a transitivity field for $\leq_{\mathsf{R}}$, and a two-sided addition-monotonicity field saying that $a \leq_{\mathsf{R}} a'$ and $b \leq_{\mathsf{R}} b'$ entail $a + b \leq_{\mathsf{R}} a' + b'$. Relative to this chosen budget structure, the closed ball-budget predicate is*

$$\mathsf{MetricBallBudget}_d(x, y; r) :\Longleftrightarrow d(x, y) \leq_{\mathsf{R}} r.$$

Lean4 (definition) `BEDC.Derived.MetricUp.MetricBallBudget`

Theorem 57.233 (Metric ball budget composition)**.** *Let X be a carrier equipped with the* $\mathsf{Met}^{\uparrow}$ *certificate obligations of Definition 57.231, and let $d : X \times X \to \mathsf{Real}^{\uparrow}$ be its certified distance. Assume also the ball-budget ordered-real fields of Definition 57.232: the comparison $\leq_{\mathsf{R}}$, addition $+$, budget-transitivity, and two-sided budget-addition-monotonicity. For $x, y, z : X$ and budgets $r, s : \mathsf{Real}^{\uparrow}$,*

$$\mathsf{MetricBallBudget}_d(x, y; r) \Longrightarrow \mathsf{MetricBallBudget}_d(y, z; s) \Longrightarrow \mathsf{MetricBallBudget}_d(x, z; r + s).$$

Equivalently,

$$d(x, y) \leq_{\mathsf{R}} r \Longrightarrow d(y, z) \leq_{\mathsf{R}} s \Longrightarrow d(x, z) \leq_{\mathsf{R}} r + s.$$

Proof. Unfold Definition 57.232. The two premises are the budget comparisons $d(x, y) \leq_{\mathsf{R}} r$ and $d(y, z) \leq_{\mathsf{R}} s$. The triangle-inequality field of Definition 57.5 gives

$$d(x, z) \leq_{\mathsf{R}} d(x, y) + d(y, z).$$

Apply the supplied two-sided budget-addition-monotonicity field to the two budget premises. This gives

$$d(x, y) + d(y, z) \leq_{\mathsf{R}} r + s.$$

Apply the supplied budget-transitivity field to the triangle inequality and the displayed budget bound. This yields $d(x, z) \leq_{\mathsf{R}} r + s$. Refolding Definition 57.232 gives the closed ball-budget conclusion. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricBallBudget composition`

Theorem 57.234 (Metric distance witness standard bridge)**.** *For the public* $\mathsf{Met}^{\uparrow}$ *distance-witness surface over unary* BHist *endpoints x, y, there is a conservative standard interpretation whose BEDC carrier is*

$$\{\, d \mid \mathsf{MetricDistanceWitness}(x, y, d) \,\},$$

whose classifier is hsame*, whose forward row reads such a witness as the finite append-distance presentation between the two unary endpoints, and whose reverse row reconstructs the BEDC witness from that finite append presentation. The bridge is only for this generated distance-witness surface; it does not assert arbitrary host real metrics.*

Proof. The carrier and classifier come from Definition 57.229: the public BEDC objects are exactly the d satisfying $\mathsf{MetricDistanceWitness}(x, y, d)$, and the classifier is hsame.

For the forward row, apply Definition 57.13 to read the visible unary context as the finite append-distance presentation between the endpoints. Definition 57.133 ensures that this presentation is determined by the visible context, while Definition 57.9 transports the same presentation across the symmetric endpoint order.

For the reverse row, read the finite append-distance presentation back through Definition 57.13. The resulting BEDC witness is classified by the same hsame row, and Definition 57.233 supplies the budget-compatible composition law needed by the public distance-witness surface.

Thus the two rows form a conservative paper-side standard bridge for the generated $\mathsf{MetricDistanceWitness}$ surface only. No completeness, compactness, topology, or arbitrary host-real metric structure is introduced. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricUp StdBridge`

Definition 57.235 (MetricSpaceUp public distance surface)**.** *The* $\mathsf{Met}^{\uparrow}$ *public distance surface is the finite* BHist *surface whose carrier rows are the unary endpoint histories* x, y*, the displayed* $\mathsf{MetricDistanceWitness}(x, y, d)$ *row, the ball-budget row* $\mathsf{MetricBallBudget}_d(x, y; r)$*, the classifier row* hsame*, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows used by Definition 57.232. A public distance read of* $\mathsf{Met}^{\uparrow}$ *is a projection of this surface and no larger host metric-space object.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface hsame transport`

Theorem 57.236 (MetricSpaceUp public distance surface exhaustion)**.** *Every public distance read of* $\mathsf{Met}^{\uparrow}$ *projects exactly to the* BHist *endpoint rows, the* MetricDistanceWitness *row, the* MetricBallBudget *row, the* hsame *classifier row, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows of Definition 57.235. Conversely, any such finite family repacks as the same public distance surface.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface exhaustion`

Proof. Project a public read through the distance-witness semantic certificate of Definition 57.229. This fixes the BHist endpoints, the concrete $\mathsf{MetricDistanceWitness}(x, y, d)$ row, and the hsame classifier row.

Next apply Definition 57.233. The only budget rows available to the public read are the displayed $\mathsf{Real}^{\uparrow}$ comparison, addition, transitivity, and monotonicity fields used to compose ball-budget witnesses.

Finally use Definition 57.234 and Definition 57.133. The bridge reads the same finite witness surface in the standard direction, and visible context determinacy prevents an additional host distance row from entering the packet. Repacking the projected rows gives exactly Definition 57.235. □

Theorem 57.237 (MetricSpaceUp visible-context distance consumer surface)**.** *The named visible-context consumers* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *may read a* $\mathsf{Met}^{\uparrow}$ *distance only through the visible-context distance witness, ball-budget, and classifier rows of Definition 57.235. Their public use of the metric interface does not add a host metric carrier, a separate topology generator, or an analytic completeness or compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface visible context consumer surface`

Proof. Start with the public surface of Definition 57.235. The only exported distance data are the MetricDistanceWitness row, the ball-budget row, the hsame classifier, and the finite Cont provenance over the displayed $\mathsf{Real}^{\uparrow}$ budget rows.

Apply Definition 57.133. Any visible-context consumer that reads a distance witness receives the determined finite witness row rather than an external host distance function.

Use Definition 57.233 for all budget propagation and Definition 57.234 for the standard distance-facing reading. These two rows exhaust the consumer-facing surface, so the listed downstream chapters can consume only the visible-context witness, budget, and classifier rows. □

Theorem 57.238 (MetricSpace visible-context public certificate)**.** *The public* $\mathsf{Met}^{\uparrow}$ *certificate is exactly the finite* BHist *endpoint rows, the* MetricDistanceWitness *row, the* MetricBallBudget *row, the* hsame *classifier row, the finite* Cont *provenance row, and the* $\mathsf{Real}^{\uparrow}$ *budget rows of Definition 57.235. Every downstream read by* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *factors through the visible-context consumer surface of Definition 57.237, so the certificate exports no host metric-space carrier, no separate topology generator, and no analytic completeness or compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspacePublicDistanceSurface visible context public certificate`

Proof. First project the visible distance read through Definition 57.229. This fixes the endpoint BHist rows, the displayed MetricDistanceWitness row, and the hsame classifier that names the distance witness.

Next compose the budget rows by Definition 57.233. The only available comparison, addition, transitivity, and monotonicity data are the displayed $\mathsf{Real}^{\uparrow}$ budget rows already present in Definition 57.235.

Then read the standard-facing side through Definition 57.234. The bridge has the same finite witness surface, so it does not introduce a host metric function or an ambient metric-space object.

Finally apply Definition 57.237. The listed downstream chapters read the metric interface only through this visible-context surface. Combining these rows gives exactly the public $\mathsf{Met}^{\uparrow}$ certificate and no larger object. □

Primitive boundary. $\mathsf{Met}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Theorem 57.239 (Metric distance witness empty-boundary visible-context distance empty)**.** *For histories p, q, d, a visible-context distance witness whose central left and right endpoints are both empty identifies the central distance with the empty history and gives zero central distance depth:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Apply Definition 57.13 to peel away the visible prefix and suffix. The remaining central witness is $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d)$. The empty-left boundary theorem identifies d with e; case analysis on that classifier gives the zero depth equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context distance empty`

Theorem 57.240 (Metric prefix-independent symmetric zero-depth empty-boundary forward)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, suppose the two symmetric visible-context distance witnesses*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q), \qquad \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q')$$

hold. If $\mathsf{MetricDistanceDepth}(d_{xy}) = 0$*, then the original visible context carries the empty-boundary witness*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q).$$

Proof. Peel both visible-context witnesses to their central witnesses. Symmetric classification compares d_{xy} and d_{yx}. Case analysis on the zero-depth central distance leaves only $d_{xy} = \mathsf{e}$; the empty-distance visible-context readback then identifies the two central endpoints as empty. Reattaching the original prefix and suffix gives the displayed empty-boundary witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth empty boundary forward`

Theorem 57.241 (Metric empty-boundary visible context append unit laws)**.** *For histories p, q, d, l, r, a visible-context distance witness whose central endpoints are both empty makes the central distance act as a left and right append unit:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{append}(d, r) = r \ \wedge \ \mathsf{append}(l, d) = l.$$

Proof. The empty-boundary distance theorem identifies d with e. Case analysis on that classifier reduces the two displayed equations to the left and right unit laws for finite-kernel append. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context append unit laws`

Theorem 57.242 (Metric empty-boundary visible context unit splice commute)**.** *For histories $p, q, d, l, r, out_L, out_R$, an empty-boundary visible-context distance witness makes left and right continuation through the central distance commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{Cont}(l, d, out_L) \Longrightarrow \mathsf{Cont}(d, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The empty-boundary distance theorem identifies d with e. The left continuation then has right-unit result $out_L\mathsf{hsame}l$, and the right continuation has left-unit result $out_R\mathsf{hsame}r$. Substituting these two unit identifications reduces the displayed splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context unit splice commute`

Theorem 57.243 (Metric visible-context empty-distance unit splice commute)**.** *For histories $p, q, x, y, l, r, out_L, out_R$, a visible-context empty-distance witness makes left continuation through x and right continuation through y commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}\mathsf{e}){\cdot}q) \Longrightarrow \mathsf{Cont}(l, x, out_L) \Longrightarrow \mathsf{Cont}(y, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The visible-context empty-distance exactness theorem identifies both central endpoints x and y with e. The left continuation then has right-unit result $out_L\mathsf{hsame}l$, and the right continuation has left-unit result $out_R\mathsf{hsame}r$. Substituting these unit identifications reduces the splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance unit splice commute`

Theorem 57.244 (Metric empty-boundary visible context left-unit witness)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness turns any continuation $\mathsf{Cont}(d, r, out)$ into a metric witness from the empty source:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(d, r, out) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{e}, r, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a left-unit continuation from e to r. The empty source is unary, the target is unary by hypothesis, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context left unit witness`

Theorem 57.245 (Metric empty-boundary visible context right-unit witness)**.** *For histories p, q, d, l, out, an empty-boundary visible-context distance witness turns any continuation $\mathsf{Cont}(l, d, out)$ into a metric witness into the empty target:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{Cont}(l, d, out) \Longrightarrow \mathsf{MetricDistanceWitness}(l, \mathsf{e}, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a right-unit continuation into e. The source is unary by hypothesis, the empty target is unary, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context right unit witness`

Theorem 57.246 (Metric empty-boundary visible context continuation depth add)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness lets a continuation through d splice into the direct continuation from l to r, so the output depth is the sum of the endpoint depths:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid,$$

Proof. The empty-boundary continuation splice removes the central empty distance from the two-step ledger and gives $\mathsf{Cont}(l, r, out)$. Together with the unary endpoint hypotheses, unary continuation closure supplies the output carrier, so these fields form a concrete metric distance witness from l to r. Metric distance depth additivity then gives the displayed equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation depth add`

Theorem 57.247 (Metric empty-boundary visible context continuation witness depth package)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness, unary visible endpoints, and the two-step continuation through d determine both the visible-endpoint metric witness and its depth ledger:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}\end{aligned}$$

Proof. The witness conjunct is supplied by Definition 57.272 from the empty-boundary witness, the endpoint unary histories, and the two continuation rows.

The depth conjunct is supplied by Definition 57.283 from the same hypotheses. Conjoining the two conclusions gives the displayed package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation witness depth`

Theorem 57.248 (Metric empty-boundary visible context canonical splice package)**.** *For histories $p, q, d, l, r, mid, out, out'$, an empty-boundary visible-context distance witness, unary endpoint carriers, a two-step continuation through d, and a displayed direct continuation from l to r determine the canonical splice package:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \mathsf{Cont}(l, r, out') \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{hsame}(out, out') \wedge \\ &\quad \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r).\end{aligned}$$

Proof. The witness row is supplied by Theorem 57.272 from the empty-boundary visible-context witness, the unary carriers, and the two displayed continuation rows.

The comparison row is supplied by Theorem 57.273, which compares the spliced output with the displayed direct continuation $\mathsf{Cont}(l, r, out')$.

The depth row is supplied by Theorem 57.283. Conjoining these three rows gives the displayed certificate package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context canonical splice package`

Theorem 57.249 (Metric distance witness standard boundary bridge)**.** *For the concrete* MetricDistanceWitness *interface on unary* BHist *endpoint witnesses, the paper-side bridge is the conservative boundary interpretation assembled from the semantic NameCert, visible-context exactness, the left and empty boundary readback rows, and the zero-depth identity-path row. It identifies exactly the BEDC boundary-distance witness rows with the corresponding standard boundary-metric reading, and back again, for the concrete* BHist *witness surface.*

Proof. The semantic carrier and classifier are the concrete MetricDistanceWitness predicate and hsame classifier supplied by Definition 57.229. Thus the bridge has no separate host carrier: its public objects are exactly the displayed BHist distance witnesses.

The boundary readback is supplied in the two directions by Definition 57.96 and Definition 57.97. These rows say that the left and empty BEDC boundary surfaces read as the same standard boundary-distance data, and that the standard reading returns to the same BEDC rows.

Finally, Definition 57.310 provides the zero-depth symmetric identity-path row used by the standard boundary metric reading. Together these rows give a conservative two-way boundary interpretation for the concrete witness surface only. No arbitrary host metric space, completeness, compactness, or topology-induced-by-metric bridge is being claimed. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness standard boundary bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$**.**

Constructive story: A metric distance witness is built from two unary endpoint histories and one unary distance history, with the continuation ledger showing that the distance is exactly the append result from the first endpoint through the second. The classifier on distance results is hsame, so two witnesses are compared by transporting their endpoint and distance histories through the same continuation data rather than by an external real-valued distance function. The source, pattern, and ledger predicates coincide for the semantic certificate: the source accepts concrete distance witnesses, the pattern records empty-distance exactness, symmetry, prefix and suffix visibility, triangle append rows, ball-budget composition, and boundary readback, and the ledger keeps the endpoint carriers plus the distance continuation and structural-depth calculation visible. Since unary append gives a witness distance $x \cdot y$ for unary endpoints x, y, hsame supplies reflexivity, symmetry, and transitivity, and case transport preserves the witness predicate, the five name-certificate obligations hold on the displayed BEDC surface. A history d lies in the signature gap of a $\mathsf{Met}^\uparrow$ token p when the source-domain accepts d as a distance witness for the chosen endpoints, the signature relation pairs it with an answer distance and evidence ledger, and token introduction names that answer as p; concretely, d is unary, its continuation ledger is the endpoint append, and its depth is the sum of the endpoint depths recorded by p. The checked standard bridge reads this finite distance witness as the standard boundary metric datum and reads it back through the same empty, prefix, suffix, and symmetric-zero rows, without changing the carrier. The chapter registers this continuation-distance construction under the name $\mathsf{Met}^\uparrow$; metric laws are certified append and boundary facts over unary histories, with no arbitrary host metric space, completeness theorem, or induced topology introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, maturePackageClosed)

Scope closed: The closed scope is the MetricDistanceWitness semantic certificate, ball-budget composition, the standard distance-witness bridge, the standard boundary bridge, and the named cross-chapter consumers in $\mathsf{TotallyBounded}^\uparrow$, $\mathsf{CompleteMetric}^\uparrow$, $\mathsf{CompactMetric}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, and the compact uniform-continuity capstone.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.MetricUp.MetricUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close arbitrary host metric spaces, general completeness, compactness, topology induced by a metric, or any bridge stronger than the checked BEDC schema bridge.

Upgrade path: no further upgrade – chapter at top of formal axis

Remark 57.250 (Zero-axiom encoding note)**.** *Replaces decidable equality with apartness and explicit real-modulus inequalities.*

Closure status for $\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$**.**

Constructive story: A $\mathsf{Met}^\uparrow$ obligation packet is generated from the checked carrier, classifier, distance witness, identity, symmetry, triangle, transport, and NameCert rows, with the Lean target read only as verification evidence for that finite row package.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, maturePackageClosed)

Scope closed: The carrier, classifier, distance witness, identity, symmetry, triangle, transport, and local NameCert rows are scoped as one finite BHist packet over componentwise hsame transport, displayed Cont replay, and $\mathsf{Real}^\uparrow$ distance primitives.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Met}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.MetricUp.MetricUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This record does not claim a public metric-object bridge, a completion principle, compactness, topology generation, or quotient real equality.

Upgrade path: No further theory-axis grade remains for this chapter; downstream bridge audits may reread the same BHist, hsame, Cont, and $\mathsf{Real}^\uparrow$ distance rows.

Theorem 57.251 (Metric depth-zero distance has empty endpoints)**.** *For histories x, y, d, a concrete distance witness whose distance ledger has structural depth zero has empty endpoints:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Classify the displayed distance history. The empty distance case is exactly the empty-distance witness exactness theorem. A zero-headed or one-headed distance has successor depth, contradicting the displayed zero-depth equation by constructor no-confusion for natural numbers. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness depth zero empty endpoints`

Theorem 57.252 (Metric empty endpoints have depth-zero distance)**.** *For histories x, y, d, a concrete distance witness whose endpoints are both history-same to* e *has structural distance depth zero:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \to \mathsf{hsame}(x, \mathsf{e}) \to \mathsf{hsame}(y, \mathsf{e}) \to \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Eliminate the two endpoint sameness witnesses, reducing both endpoints to e. The continuation ledger inside the distance witness then reduces the distance history itself to e, and the depth definition evaluates to zero. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty endpoints depth zero`

Theorem 57.253 (Metric distance depth zero iff empty)**.** *For a history d, structural metric distance depth is zero exactly when d classifies with the empty history:*

$$\mathsf{MetricDistanceDepth}(d) = 0 \iff \mathsf{hsame}(d, \mathsf{e}).$$

Proof. Case-analyze the history d. The empty case is reflexive. A zero-headed or one-headed history has successor depth, so it cannot satisfy the zero-depth equation. Conversely, a history-sameness proof with e reduces the depth equation to the empty reflexive case. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth zero iff empty`

Theorem 57.254 (Metric distance positive depth iff nonempty)**.** *For a history d, positive structural metric distance depth is equivalent to not being history-same as* e*:*

$$0 < \mathsf{MetricDistanceDepth}(d) \iff (d\mathsf{hsamee} \to \mathsf{False}).$$

Proof. Case-analyze d. The empty case has depth zero and cannot satisfy the nonempty predicate. A zero-headed or one-headed history has successor depth, and constructor separation rules out history-sameness with e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth positive iff nonempty`

Theorem 57.255 (Metric distance append positive depth iff)**.** *For histories p and d, an appended distance ledger has positive structural depth exactly when one of its two factors has positive structural depth:*

$$0 < \mathsf{MetricDistanceDepth}(p \cdot d) \iff 0 < \mathsf{MetricDistanceDepth}(p) \lor 0 < \mathsf{MetricDistanceDepth}(d).$$

Proof. Use the positive-depth classifier to replace positive depth by internal nonemptiness. Kernel append nonemptiness then splits the appended history into the left or right nonempty factor. The converse applies the same two classifiers in reverse and rebuilds nonemptiness of the appended history from the displayed nonempty factor. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append positive iff`

Theorem 57.256 (Metric distance positive depth shape cases)**.** *For a history d, positive structural metric distance depth exposes one of the two visible history constructors:*

$$(\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \implies (\exists z : \mathsf{BHist}.\ d = \mathsf{E0}(z)) \lor (\exists z : \mathsf{BHist}.\ d = \mathsf{E1}(z)).$$

Proof. Case-analyze d. The empty branch contradicts the displayed positive-depth hypothesis because the structural depth of e is zero. The two visible branches return the exposed payload under E0 or E1. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth positive shape cases`

Theorem 57.257 (Metric distance depth append-left equality emptiness)**.** *For histories p and d, appending p on the left preserves structural metric distance depth exactly when p is the empty history:*

$$\mathsf{MetricDistanceDepth}(p \cdot d) = \mathsf{MetricDistanceDepth}(d) \iff \mathsf{hsame}(p, \mathsf{e}).$$

Proof. Induct on the displayed tail d. The empty tail is precisely Definition 57.253. A zero-headed or one-headed tail reduces the equality through successor injectivity, and the reverse direction rebuilds the same successor equality after the left prefix is identified with e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append left eq iff empty`

Theorem 57.258 (Metric distance depth append-left cancellation)**.** *For histories p, d, e : BHist, equality of structural metric distance depths after appending the same left prefix cancels that prefix:*

$$\mathsf{MetricDistanceDepth}(p{\cdot}d) = \mathsf{MetricDistanceDepth}(p{\cdot}e) \implies \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e).$$

Proof. The structural depth of a left append decomposes as the depth of the common prefix plus the depth of the tail. Rewriting both displayed sides by this depth-add equation reduces the claim to left cancellation for addition in $\mathbb{N}$, yielding equality of the two tail depths. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append left cancel`

Theorem 57.259 (Metric strict prefix depth decomposition)**.** *For histories h, k : BHist, a strict unary prefix witness from h to k exposes a nonempty unary tail whose continuation reconstructs k, and the metric depth of k splits as the depth of h plus the depth of that tail:*

$$\mathsf{NatUnaryStrictPrefix}(h,k) \Longrightarrow \exists tail : \mathsf{BHist}.\ \mathsf{UnaryHistory}(tail) \wedge (tail = \mathsf{e} \to \mathsf{False}) \wedge \mathsf{Cont}(h, tail, k) \wedge \mathsf{MetricDistance}$$

Proof. Unpack the strict-prefix witness. It already supplies a unary nonempty tail and a continuation ledger from h through that tail to k. Inverting the continuation identifies k with $h \cdot tail$. Induction on the concrete tail then proves the structural depth equation for append, giving the displayed sum formula. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth strict prefix decomposes`

Theorem 57.260 (Metric distance depth visible-context cancellation)**.** *For histories p, q, d, e : BHist, equality of structural depths after a common visible prefix and suffix cancels both contexts:*

$$\mathsf{MetricDistanceDepth}((p{\cdot}d){\cdot}q) = \mathsf{MetricDistanceDepth}((p{\cdot}e){\cdot}q) \Longrightarrow \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e$$

Proof. Induct on the visible suffix q. The empty suffix is exactly Definition 57.258. In the two visible constructor cases, both displayed depths have a common successor head, so successor injectivity removes the suffix constructor and the induction hypothesis cancels the remaining visible context. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context cancel`

Theorem 57.261 (Metric distance depth append-right cancellation iff)**.** *For histories p, d, e : BHist, appending the same right suffix preserves and reflects equality of structural metric distance depth:*

$$\mathsf{MetricDistanceDepth}(d{\cdot}p) = \mathsf{MetricDistanceDepth}(e{\cdot}p) \Longleftrightarrow \mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e).$$

Proof. Induct on the common suffix p. The empty suffix is immediate. In the two visible constructor cases, both appended depths carry the same successor head; successor injectivity gives the forward implication, and congruence of successor rebuilds the appended equality in the reverse implication. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append right cancel iff`

Theorem 57.262 (Metric distance depth append-right equality emptiness)**.** *For histories p and d, appending p on the right preserves structural metric distance depth exactly when p is the empty history:*

$$\mathsf{MetricDistanceDepth}(d \cdot p) = \mathsf{MetricDistanceDepth}(d) \Longleftrightarrow \mathsf{hsame}(p, \mathsf{e}).$$

Proof. The structural depth of a right append decomposes as the depth of d plus the depth of p. If the appended depth equals the original depth, cancellation by $\mathbb{N}$-addition forces the depth of p to be zero, hence Definition 57.253 identifies p with e. Conversely, substituting $p = \mathsf{e}$ makes the right append definitionally equal to d. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth append right eq iff empty`

Theorem 57.263 (Metric distance same-source continuation result cancellation)**.** *For unary histories x, y, z and continuation rows $\mathsf{Cont}(x, y, d)$ and $\mathsf{Cont}(x, z, e)$, equality of the two result depths cancels the common source depth:*

$$\mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(e) \Longrightarrow \mathsf{MetricDistanceDepth}(y) = \mathsf{MetricDistanceDepth}(z).$$

Proof. Package the two continuation rows as metric-distance witnesses. Depth additivity gives $\mathsf{depth}(d) = \mathsf{depth}(x) + \mathsf{depth}(y)$ and $\mathsf{depth}(e) = \mathsf{depth}(x) + \mathsf{depth}(z)$. The displayed result depth equality therefore has a common left summand, and natural-number cancellation identifies the two target depths. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth same source cont result cancel`

Theorem 57.264 (Metric distance depth visible-context equality emptiness)**.** *For histories p, q, d, adding visible context on both sides preserves the central structural metric distance depth exactly when both visible contexts are empty:*

$$\mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = \mathsf{MetricDistanceDepth}(d) \Longleftrightarrow \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}).$$

Proof. Expand structural depth over the two append contexts. The resulting natural number equation has the form $(\mathsf{depth}(p) + \mathsf{depth}(d)) + \mathsf{depth}(q) = \mathsf{depth}(d)$, so associativity and commutativity of addition reduce it to a zero-sum constraint on the two context depths. Then Definition 57.253 identifies both contexts with e. The reverse direction substitutes the two empty-context identifications and applies the one-sided append equality theorems above. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context eq iff empty`

Theorem 57.265 (Metric visible-context zero depth iff empty components)**.** *For histories p, q, d, the full visible-context ledger has structural metric depth zero exactly when the prefix, central distance, and suffix are all empty:*

$$\mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = 0 \Longleftrightarrow \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}).$$

Proof. The zero-depth classifier first identifies the whole visible ledger with e. BEDC append-empty inversion splits the outer append into $(p \cdot d)\mathsf{hsamee}$ and $q\mathsf{hsamee}$, and a second inversion splits $p \cdot d$. Conversely, substituting the three empty components makes the displayed depth definition evaluate to zero. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceDepth visible context zero iff`

Theorem 57.266 (Metric visible-context total zero components)**.** *For histories p, q, x, y, d, a visible-context distance witness whose full result ledger has structural depth zero exposes empty prefix, source, target, suffix, and central distance components:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \to \\ &\quad \mathsf{MetricDistanceDepth}((p \cdot d) \cdot q) = 0 \to \\ &\quad \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}) \wedge \mathsf{hsame}(d, \mathsf{e}).\end{aligned}$$

Proof. Zero depth classifies the full result ledger with e, and two append-empty inversions expose the empty prefix, central distance, and suffix. Transport the witness continuation along the same empty result classifier. The empty-result continuation inversion then classifies the visible source $p \cdot x$ and target $y \cdot q$ with e, and append-empty inversion splits these endpoint histories into their displayed components. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context total zero components`

Theorem 57.267 (Metric visible-context depth-zero collapse)**.** *For histories p, q, x, y, d, a visible-context distance witness whose central distance has structural depth zero collapses the central distance and both central endpoints to the empty history:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \to \mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Apply visible-context exactness to peel away the prefix p and suffix q, recovering the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$. The depth-zero endpoint theorem gives $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$, while the depth-zero classifier theorem gives $\mathsf{hsame}(d, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context depth zero collapse`

Theorem 57.268 (Metric empty-boundary visible-context continuation left unit)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness and a continuation out of the central distance identify the continuation result with its right input:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{Cont}(d, r, out) \to \mathsf{hsame}(out, r).$$

Proof. Apply empty-boundary visible-context exactness to read back $\mathsf{hsame}(d, \mathsf{e})$. Case analysis on this classifier reduces the continuation to an empty-left continuation, whose result is the right input by the left-unit law for Cont. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation left unit result`

Theorem 57.269 (Metric empty-boundary visible-context continuation left unit iff)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness turns continuation from the central distance into exactly the left-unit continuation classifier:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \big(\mathsf{Cont}(d, r, out) \leftrightarrow \mathsf{hsame}(out, r)\big).$$

Proof. The visible-context empty-boundary theorem gives $\mathsf{hsame}(d, \mathsf{e})$. Case analysis on this classifier reduces the displayed continuation predicate to $\mathsf{Cont}(\mathsf{e}, r, out)$, which is characterized by the BEDC left-unit equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation left unit iff`

Theorem 57.270 (Metric empty-boundary visible-context continuation right unit iff)**.** *For histories p, q, d, l, out, an empty-boundary visible-context distance witness turns continuation into the central distance into exactly the right-unit continuation classifier:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \big(\mathsf{Cont}(l, d, out) \leftrightarrow \mathsf{hsame}(out, l)\big).$$

Proof. Again the visible-context empty-boundary theorem gives $\mathsf{hsame}(d, \mathsf{e})$. After case analysis, the displayed continuation predicate is $\mathsf{Cont}(l, \mathsf{e}, out)$, which is characterized by the BEDC right-unit equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation right unit iff`

Theorem 57.271 (Metric empty-boundary continuation splice)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness makes a continuation through the central distance splice to the continuation that skips that empty central boundary:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out).$$

Proof. The empty-boundary visible-context theorem gives $\mathsf{hsame}(d, \mathsf{e})$. After case analysis on this classifier, the left continuation is $\mathsf{Cont}(l, \mathsf{e}, mid)$, so its unit law identifies mid with l. The right continuation is therefore the desired $\mathsf{Cont}(l, r, out)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation splice`

Theorem 57.272 (Metric empty-boundary visible-context witness splice)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness upgrades a two-step continuation through the central distance into a metric distance witness for the visible endpoints:*

$$\begin{gathered}\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \to \mathsf{UnaryHistory}(l) \to \mathsf{UnaryHistory}(r) \to \\ \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{MetricDistanceWitness}(l, r, out).\end{gathered}$$

Proof. The empty-boundary continuation-splice theorem gives the concrete continuation $\mathsf{Cont}(l, r, out)$ from the visible-context witness and the two displayed continuation rows. Together with the displayed unary carriers for l and r, the BEDC unary-continuation closure theorem supplies $\mathsf{UnaryHistory}(out)$. These four rows are precisely the fields of $\mathsf{MetricDistanceWitness}(l, r, out)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context witness splice`

Theorem 57.273 (Metric empty-boundary continuation splice determinacy)**.** *For histories $p, q, d, l, r, mid, out, out$ an empty-boundary visible-context distance witness makes the spliced continuation result match any displayed direct continuation result:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \to \mathsf{Cont}(l, d, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out') \to \mathsf{hsame}(out, out')$$

Proof. The continuation-splice theorem supplies $\mathsf{Cont}(l, r, out)$ from the empty-boundary visible-context witness and the two displayed continuation rows. BEDC continuation determinacy compares this spliced continuation with the displayed $\mathsf{Cont}(l, r, out')$, giving $\mathsf{hsame}(out, out')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation splice hsame`

Theorem 57.274 (Metric visible-context positive central depth has a nonempty endpoint)**.** *For histories p, q, x, y, d, a visible-context distance witness whose central distance has positive structural depth exposes a nonempty central endpoint:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}d){\cdot}q) \to (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to ((\mathsf{hsame}(x, \mathsf{e}) \to \mathsf{False}) \vee (\mathsf{hsame}$$

Proof. Apply visible-context exactness to peel away the displayed prefix p and suffix q, obtaining the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$.

It remains to convert positive structural depth into nonempty-distance input. If $\mathsf{hsame}(d, \mathsf{e})$ held, the depth-zero classifier theorem would give $\mathsf{MetricDistanceDepth}(d) = 0$, contradicting the positive-depth hypothesis.

Apply the nonempty-distance endpoint theorem to the central witness and this derived nonempty-distance contradiction. The result is exactly the displayed disjunction: either the left central endpoint is not history-same to e, or the right central endpoint is not history-same to e. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context positive depth nonempty endpoint`

Theorem 57.275 (Metric visible-context positive central depth e-one endpoint cases)**.** *For histories $p, q, x, y, d : \mathsf{BHist}$, positive central distance depth in a visible MetricUp ledger exposes an e-one constructor at one central endpoint with a unary tail:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \to \\ &\quad (\mathsf{MetricDistanceDepth}(d) = 0 \to \mathsf{False}) \to \\ &\quad (\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)).\end{aligned}$$

Here the BEDC source is MetricDistanceWitness*, the carrier objects are* UnaryHistory *histories, the classifier is* hsame*, and the obligation is constructor readback under visible prefix and suffix ledgers.*

Proof. Apply visible-context exactness to remove the prefix p and suffix q from the displayed ledger. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$.

Next convert positive central depth into the nonempty-distance input required by the constructor readback theorem. If $d\mathsf{hsamee}$ held, then Definition 57.253 would give $\mathsf{MetricDistanceDepth}(d) = 0$, contradicting the positive-depth hypothesis.

Apply Definition 57.216 to the recovered central witness and this nonempty-distance contradiction. The result is exactly the displayed disjunction of e-one endpoint witnesses with unary tails. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context positive depth e1 endpoint cases`

Theorem 57.276 (Metric distance witness empty-boundary visible-context distance empty)**.** *For histories p, q, d, a visible-context distance witness whose central left and right endpoints are both empty identifies the central distance with the empty history and gives zero central distance depth:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{hsame}(d, \mathsf{e}) \wedge \mathsf{MetricDistanceDepth}(d) = 0.$$

Proof. Apply Definition 57.13 to peel away the visible prefix and suffix. The remaining central witness is $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d)$. The empty-left boundary theorem identifies d with e; case analysis on that classifier gives the zero depth equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context distance empty`

Theorem 57.277 (Metric prefix-independent symmetric zero-depth empty-boundary forward)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, suppose the two symmetric visible-context distance witnesses*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q), \qquad \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q')$$

hold. If $\mathsf{MetricDistanceDepth}(d_{xy}) = 0$*, then the original visible context carries the empty-boundary witness*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q).$$

Proof. Peel both visible-context witnesses to their central witnesses. Symmetric classification compares d_{xy} and d_{yx}. Case analysis on the zero-depth central distance leaves only $d_{xy} = \mathsf{e}$; the empty-distance visible-context readback then identifies the two central endpoints as empty. Reattaching the original prefix and suffix gives the displayed empty-boundary witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth empty boundary forward`

Theorem 57.278 (Metric empty-boundary visible context append unit laws)**.** *For histories* p, q, d, l, r*, a visible-context distance witness whose central endpoints are both empty makes the central distance act as a left and right append unit:*

$$\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{append}(d, r) = r \ \wedge \ \mathsf{append}(l, d) = l.$$

Proof. The empty-boundary distance theorem identifies d with e. Case analysis on that classifier reduces the two displayed equations to the left and right unit laws for finite-kernel append. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context append unit laws`

Theorem 57.279 (Metric empty-boundary visible context unit splice commute)**.** *For histories* $p, q, d, l, r, out_L, out_R$*, an empty-boundary visible-context distance witness makes left and right continuation through the central distance commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{Cont}(l, d, out_L) \Longrightarrow \mathsf{Cont}(d, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The empty-boundary distance theorem identifies d with e. The left continuation then has right-unit result $out_L\mathsf{hsame}l$, and the right continuation has left-unit result $out_R\mathsf{hsame}r$. Substituting these two unit identifications reduces the displayed splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context unit splice commute`

Theorem 57.280 (Metric visible-context empty-distance unit splice commute)**.** *For histories* $p, q, x, y, l, r, out_L, out_R$*, a visible-context empty-distance witness makes left continuation through* x *and right continuation through* y *commute after one ordinary append splice:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}\mathsf{e}){\cdot}q) \Longrightarrow \mathsf{Cont}(l, x, out_L) \Longrightarrow \mathsf{Cont}(y, r, out_R) \Longrightarrow \mathsf{hsame}(out_L{\cdot}r, l{\cdot}out_R).$$

Proof. The visible-context empty-distance exactness theorem identifies both central endpoints x and y with e. The left continuation then has right-unit result $out_L\mathsf{hsame}l$, and the right continuation has left-unit result $out_R\mathsf{hsame}r$. Substituting these unit identifications reduces the splice comparison to reflexivity. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance unit splice commute`

Theorem 57.281 (Metric empty-boundary visible context left-unit witness)**.** *For histories p, q, d, r, out, an empty-boundary visible-context distance witness turns any continuation* $\mathsf{Cont}(d, r, out)$ *into a metric witness from the empty source:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(d, r, out) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{e}, r, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a left-unit continuation from e to r. The empty source is unary, the target is unary by hypothesis, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context left unit witness`

Theorem 57.282 (Metric empty-boundary visible context right-unit witness)**.** *For histories p, q, d, l, out, an empty-boundary visible-context distance witness turns any continuation* $\mathsf{Cont}(l, d, out)$ *into a metric witness into the empty target:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{Cont}(l, d, out) \Longrightarrow \mathsf{MetricDistanceWitness}(l, \mathsf{e}, out).$$

Proof. The empty-boundary distance theorem identifies d with e. Substituting that classifier makes the continuation a right-unit continuation into e. The source is unary by hypothesis, the empty target is unary, and unary continuation closure supplies the output carrier, so the three fields package the displayed metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context right unit witness`

Theorem 57.283 (Metric empty-boundary visible context continuation depth add)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness lets a continuation through d splice into the direct continuation from l to r, so the output depth is the sum of the endpoint depths:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}\mathsf{e}, \mathsf{e}{\cdot}q, (p{\cdot}d){\cdot}q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid,$$

Proof. The empty-boundary continuation splice removes the central empty distance from the two-step ledger and gives $\mathsf{Cont}(l, r, out)$. Together with the unary endpoint hypotheses, unary continuation closure supplies the output carrier, so these fields form a concrete metric distance witness from l to r. Metric distance depth additivity then gives the displayed equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation depth add`

Theorem 57.284 (Metric empty-boundary visible context continuation witness depth package)**.** *For histories p, q, d, l, r, mid, out, an empty-boundary visible-context distance witness, unary visible endpoints, and the two-step continuation through d determine both the visible-endpoint metric witness and its depth ledger:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}$$

Proof. The witness conjunct is supplied by Definition 57.272 from the empty-boundary witness, the endpoint unary histories, and the two continuation rows.

The depth conjunct is supplied by Definition 57.283 from the same hypotheses. Conjoining the two conclusions gives the displayed package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context continuation witness depth`

Theorem 57.285 (Metric empty-boundary visible context canonical splice package)**.** *For histories $p, q, d, l, r, mid, out, out'$, an empty-boundary visible-context distance witness, unary endpoint carriers, a two-step continuation through d, and a displayed direct continuation from l to r determine the canonical splice package:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{UnaryHistory}(l) \Longrightarrow \mathsf{UnaryHistory}(r) \Longrightarrow \\ &\quad \mathsf{Cont}(l, d, mid) \Longrightarrow \mathsf{Cont}(mid, r, out) \Longrightarrow \mathsf{Cont}(l, r, out') \Longrightarrow \\ &\quad \mathsf{MetricDistanceWitness}(l, r, out) \wedge \mathsf{hsame}(out, out') \wedge \\ &\quad \mathsf{MetricDistanceDepth}(out) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r).\end{aligned}$$

Proof. The witness row is supplied by Theorem 57.272 from the empty-boundary visible-context witness, the unary carriers, and the two displayed continuation rows.

The comparison row is supplied by Theorem 57.273, which compares the spliced output with the displayed direct continuation $\mathsf{Cont}(l, r, out')$.

The depth row is supplied by Theorem 57.283. Conjoining these three rows gives the displayed certificate package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness empty boundary visible context canonical splice package`

Theorem 57.286 (Metric distance witness left e1 result iff)**.** *For histories x, y, d, a distance witness from a visible left endpoint to a visible result is equivalent to the empty middle case or to a visible middle tail carrying the remaining witness:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longleftrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)) \vee (\exists y_1 : \mathsf{BHist}.\ y = \mathsf{E1}(y_1) \wedge \mathsf{Metric}$$

Proof. The forward direction is the visible-result case split. For the reverse direction, the empty middle branch rebuilds the witness from unary closure under $\mathsf{E1}$ and the right-unit continuation law. The visible middle branch adds one visible constructor back to the middle endpoint and result ledger by the one-step continuation rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result iff`

Theorem 57.287 (Metric distance witness left e1 result depth addition)**.** *For histories x, y, d, every visible-result distance witness*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d))$$

descends the metric depth identity to the visible tails:

$$\mathsf{MetricDistanceDepth}(d) = \mathsf{MetricDistanceDepth}(x) + \mathsf{MetricDistanceDepth}(y).$$

Proof. Apply metric-distance depth addition to the displayed witness. The resulting identity has a visible constructor on the source side and on the result side. Normalize the natural-number expression with successor addition, then cancel the shared successor. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result depth add`

Theorem 57.288 (Metric distance witness left e1 result empty target iff)**.** *For histories x, y, d, a metric distance witness from $\mathsf{E1}(x)$ to target y with visible result $\mathsf{E1}(d)$ identifies y with the empty history exactly when the exposed source tail is unary and agrees with the result tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow (\mathsf{hsame}(y, \mathsf{e}) \Longleftrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)).$$

Proof. Split the target shape. The empty target branch is the right-unit continuation case and reads back the unary source tail and the tail equality. The $e0$ target contradicts the unary target carrier. The visible target branch cannot also have $x\mathsf{hsame}d$: transporting the result to $\mathsf{E1}(x)$ would make $\mathsf{Cont}(\mathsf{E1}(x), \mathsf{E1}(y_1), \mathsf{E1}(x))$, so right-unit uniqueness would identify $\mathsf{E1}(y_1)$ with e, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result empty target iff`

Theorem 57.289 (Metric distance witness left e1 result hsame source)**.** *For histories x, x', y, d, a distance witness from $\mathsf{E1}(x)$ to a visible result keeps the same empty-or-visible alternative after transporting the displayed source payload along $\mathsf{hsame}(x, x')$:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x') \wedge \mathsf{hsame}(x', d)) \vee (\exists y_1 : \mathsf{BHist}.$$

Proof. Case-reduce the source hsame witness. The transported payload is then definitionally the original payload, and the established visible-result split gives the displayed empty target or visible target branch. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame source`

Theorem 57.290 (Metric distance witness left e1 result hsame target)**.** *For histories x, y, y', d, a distance witness from a visible left endpoint to a visible result keeps the same empty-or-visible target alternative after transporting the target presentation along $\mathsf{hsame}(y, y')$:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow (y' = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d)) \vee (\exists y_1 : \mathsf{BHist}.\ y$$

Proof. Apply the visible-result case split. In the empty branch, empty-history inversion transports the displayed target to e. In the visible branch, one-constructor inversion of $\mathsf{hsame}(\mathsf{E1}(y_0), y')$ exposes $y' = \mathsf{E1}(y_1)$ and $y_0\mathsf{hsame}y_1$; unary and continuation transport carry the tail metric witness to the displayed target tail. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame target`

Theorem 57.291 (Metric distance witness left e1 result hsame result)**.** *For histories x, y, d, d', a distance witness from a visible left endpoint to a visible result keeps the same empty-or-visible target alternative after transporting the displayed result tail along $\mathsf{hsame}(d, d')$:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow (y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, d')) \vee (\exists y_1 : \mathsf{BHist}.\ y$$

Proof. First lift the result-tail agreement through the visible constructor, obtaining $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$. The metric hsame transport field is then applied with reflexive source and target transports, so the original witness repacks as $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d'))$.

Apply Theorem 57.286 to this transported witness. The empty target branch gives $\mathsf{UnaryHistory}(x)$ and $\mathsf{hsame}(x, d')$; the visible target branch gives a tail y_1 with $y = \mathsf{E1}(y_1)$ and $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y_1, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result hsame result`

Theorem 57.292 (Metric distance witness left e1 result tail determinism)**.** *For histories x, y, d, d', two visible-result distance witnesses with the same visible left endpoint and target determine the same result tail:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d')) \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Unpack the two distance witnesses only to their continuation ledgers. BEDC continuation determinacy gives $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$ for the two visible results. One-constructor inversion for $\mathsf{E1}$ then descends this classifier to $\mathsf{hsame}(d, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result tail deterministic`

Theorem 57.293 (Metric distance witness left e1 result shape)**.** *For histories x, y, d, every distance witness from $\mathsf{E1}(x)$ has a visible result shape:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \Big((y = \mathsf{e} \wedge \mathsf{UnaryHistory}(x) \wedge \mathsf{hsame}(x, k)) \vee \exists y_1 : \mathsf{E}$$

Proof. Decompose the result history. An empty result is incompatible with a visible left endpoint and the continuation equation; an $\mathsf{E0}$ result is incompatible with the unary distance carrier. In the $\mathsf{E1}(k)$ case, the established visible-result split gives either the empty target boundary or the visible target tail witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 result shape`

Theorem 57.294 (Metric distance witness left e1 visible target step)**.** *For histories x, y, d, a distance witness from $\mathsf{E1}(x)$ to y with result d remains a distance witness after placing the target and result under one visible $\mathsf{E1}$ constructor:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)).$$

Proof. Unpack the witness into unary carriers and the continuation ledger. Unary closure supplies carriers for $\mathsf{E1}(y)$ and $\mathsf{E1}(d)$, and the continuation step rule lifts the same ledger from target y and result d to target $\mathsf{E1}(y)$ and result $\mathsf{E1}(d)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target step`

Theorem 57.295 (Metric distance witness left e1 visible target result shape)**.** *For histories x, y, d, any distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ has a visible result tail, and that tail is again the distance witness from $\mathsf{E1}(x)$ to y:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \exists k.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, k).$$

Proof. Apply the left visible-result shape split to the displayed witness. The empty-target branch would identify $\mathsf{E1}(y)$ with e, contradicting constructor no-confusion. The remaining visible branch supplies the result tail k and the descended witness with target y. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result shape`

Theorem 57.296 (Metric distance witness left e1 visible target result tail exactness)**.** *For histories x, y, d, k, if a distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ has displayed result history-same as $\mathsf{E1}(k)$, then k is the descended distance result from $\mathsf{E1}(x)$ to y:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(k)) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, k).$$

Proof. Apply the visible-target result shape theorem to expose a result tail k_0, a displayed comparison $\mathsf{hsame}(d, \mathsf{E1}(k_0))$, and the descended witness. Composing this comparison with the assumed comparison to $\mathsf{E1}(k)$ gives a one-headed history-sameness relation between $\mathsf{E1}(k_0)$ and $\mathsf{E1}(k)$. Constructor inversion descends it to $\mathsf{hsame}(k_0, k)$, and tail transport returns the displayed witness at k. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result tail hsame exact`

Theorem 57.297 (Metric distance witness left e1 visible target result-source absurd)**.** *For histories x, y, d, a distance witness from $\mathsf{E1}(x)$ to $\mathsf{E1}(y)$ with visible result $\mathsf{E1}(d)$ cannot also identify the descended result d with the visible source $\mathsf{E1}(x)$ when the target tail is nonempty:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow (\mathsf{hsame}(y, \mathsf{e}) \Rightarrow \bot) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(x)) \Longrightarrow \bot.$$

Proof. Right-visible result exactness descends the witness to $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$. Transporting its continuation result across $\mathsf{hsame}(d, \mathsf{E1}(x))$ makes y a right-unit continuation at $\mathsf{E1}(x)$, so right-unit uniqueness gives $\mathsf{hsame}(y, \mathsf{e})$, contradicting the displayed nonempty target-tail hypothesis. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target result source absurd`

Theorem 57.298 (Metric distance witness left e1 source step)**.** *For histories x, y, d, a distance witness from x to y with result d remains a distance witness after placing the source and result under one visible* $\mathsf{E1}$ *constructor:*

$$\mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)).$$

Proof. Unpack the witness into unary carriers and the continuation ledger $d = x \cdot y$. Unary closure supplies the new source and result carriers. Since y is unary, the append law rewrites $\mathsf{E1}(x) \cdot y$ as $\mathsf{E1}(x \cdot y)$, so the lifted result $\mathsf{E1}(d)$ satisfies the required continuation equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source step`

Theorem 57.299 (Metric distance witness left e1 source iff)**.** *For histories x, y, d, placing the source and result under one visible* $\mathsf{E1}$ *constructor is reversible:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. The reverse direction is Theorem 57.298. For the forward direction, split on the target history. The empty target case is the right empty-boundary exactness of the metric witness. A zero-headed target contradicts the unary target carrier recorded in the witness. In the one-headed target case, the result equation descends through the visible constructor, and unary append commutes the visible source constructor across the unary target tail, giving the original continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source iff`

Theorem 57.300 (Metric distance witness left e1 source result iff)**.** *For histories x, y, d, placing both the source and result under one visible* $\mathsf{E1}$ *constructor is reversible:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. For the forward direction, unpack the displayed witness. Unary inversion on the source and result gives unary carriers for x and d. The continuation ledger identifies $\mathsf{E1}(d)$ with $\mathsf{E1}(x) \cdot y$, and the unary append law for the target rewrites the right side to $\mathsf{E1}(x \cdot y)$; constructor inversion exposes the descended continuation ledger $d = x \cdot y$.

For the reverse direction, apply Definition 57.298. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 source result iff`

Theorem 57.301 (Metric distance witness left e1 visible target iff)**.** *For histories x, y, d, placing both the target and result under one visible* E1 *constructor is reversible for a visible left endpoint:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d).$$

Proof. For the forward direction, apply Theorem 57.70 to the displayed witness with the reflexive presentation $\mathsf{hsame}(\mathsf{E1}(y), \mathsf{E1}(y))$. This strips the visible target presentation and returns the tail witness with target y and result d.

For the reverse direction, apply Theorem 57.294. The visible target constructor and matching visible result constructor are restored by the continuation step rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness left e1 visible target iff`

The ten-field fit is: traditional target visible unary metric descent/extension; BEDC source BHist with UnaryHistory and Cont; carrier MetricDistanceWitness; classifier hsame; ledger one visible target constructor; core theorem Theorem 57.301; exactness reversible target-step readback; status ProofTarget.

Theorem 57.302 (MetricSpace LocatedReal distance source boundary)**.** *Let L_x and L_y be accepted* $\mathsf{LocatedReal}^{\uparrow}$ *packets whose* $\mathsf{StreamName}^{\uparrow}$ *schedules admit a common finite observation window. Suppose their* DyadicRatCore *endpoint ledgers and radius ledgers are transported to that window by the displayed* hsame *and* Cont *rows, and suppose a contextual metric row* $\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q)$ *is displayed for the two projected endpoints.*

Then any $\mathsf{Met}^{\uparrow}$ *consumer reads the distance-source boundary exactly through the finite rows named by Definition 309.11 together with the metric visible-context rows: the* $\mathsf{RegSeqRat}^{\uparrow}$ *source windows, the* DyadicRatCore *endpoint and radius rows, the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, and the* hsame *classifier on the central distance row d.*

Conversely, any packet carrying precisely those rows repacks as the metric distance-source row for the two located packets. No quotient real, host real equality, trichotomy principle, open-cover compactness datum, or metric completion object is part of this boundary.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness locatedreal distance source boundary`

Proof. This boundary is the $\mathsf{Met}^{\uparrow}$ consumer of the located-real carrier of **??**; the cited sibling supplies the two accepted $\mathsf{LocatedReal}^{\uparrow}$ packets and their endpoint rows, while this theorem supplies the visible-context distance read.

Project L_x and L_y with Definition 309.11. This leaves only the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source windows, the common $\mathsf{StreamName}^{\uparrow}$ schedule, the transported DyadicRatCore endpoint and radius ledgers, the enclosure classifier rows, and the visible BHist, hsame, Cont, Pkg, and NameCert provenance.

Next apply Definition 57.13 to the contextual metric row. The visible prefix and suffix rows are peeled away, leaving the central witness $\mathsf{MetricDistanceWitness}(x, y, d)$ over the two projected located endpoints.

For uniqueness of the central distance read, apply Definition 57.133. Any second packet with the same displayed source windows, endpoint ledgers, schedule rows, provenance rows, and contextual metric witness has a central distance classifier $\mathsf{hsame}(d, e)$, so the metric consumer cannot distinguish a hidden distance coordinate.

The converse direction repacks exactly the displayed rows: the located packets come from the handoff surface, the contextual distance row supplies the metric visible context, and the central hsame classifier is the only distance classifier retained. Since each coordinate in this construction is one of the listed finite rows, no excluded real-number, compactness, or completion principle is introduced. □

Theorem 57.303 (Metric distance witness prefix-independent symmetric classifier)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric metric distance witnesses with independent visible contexts still determine the same central distance classifier:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{hsame}(d_{xy}, d_{yx}).\end{aligned}$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This recovers central witnesses $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ and $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$. The central symmetric classifier then gives $\mathsf{hsame}(d_{xy}, d_{yx})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent symmetric classifier`

Theorem 57.304 (Metric distance witness prefix-independent symmetric zero-depth collapse)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent contexts and a zero-depth forward distance collapse the two central distances and both central endpoints:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{hsame}(d_{xy}, d_{yx}) \wedge \mathsf{MetricDistanceDepth}(d_{yx}) = 0 \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.303 to the two visible-context witnesses. This gives $\mathsf{hsame}(d_{xy}, d_{yx})$.

Apply Definition 57.267 to the first visible-context witness and the zero-depth hypothesis for d_{xy}. This gives the empty endpoint classifiers $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$.

Finally peel the second visible-context witness with Definition 57.13. The recovered central witness $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$, together with the two empty endpoint classifiers in reversed order, is the input required by Definition 57.252; hence $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$. Combining these three pieces gives the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent symmetric zero depth collapse`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth collapse`

Theorem 57.305 (Metric prefix-independent symmetric zero-depth reverse empty-boundary)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent*

contexts and a zero-depth forward distance also package the reverse context as its own empty-boundary distance witness:

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q').\end{aligned}$$

Proof. Peel the reverse visible-context witness with Definition 57.13. This recovers the unary context data for p' and q' together with the central reverse witness $\mathsf{MetricDistanceWitness}(y, x, d_{yx})$.

Apply Definition 57.304 to the two displayed visible-context witnesses and the zero-depth forward distance. The resulting endpoint classifiers include $\mathsf{hsame}(y, \mathsf{e})$ and $\mathsf{hsame}(x, \mathsf{e})$.

Finally use Definition 57.389 in the reverse context. The recovered carriers for p' and q', together with the empty endpoint classifiers in reverse order, rebuild exactly $\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth reverse empty boundary package`

Theorem 57.306 (Metric prefix-independent symmetric zero-depth two-sided empty-boundary package)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$, symmetric visible-context distance witnesses with independent contexts and a zero-depth forward distance package both contextual empty-boundary witnesses together with the reverse zero-depth row and the central endpoint classifiers:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xy}) = 0 \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q) \wedge \\ &\mathsf{MetricDistanceWitness}(p' \cdot \mathsf{e}, \mathsf{e} \cdot q', (p' \cdot \mathsf{e}) \cdot q') \wedge \mathsf{MetricDistanceDepth}(d_{yx}) = 0 \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.304 to the two visible-context witnesses and the zero-depth forward distance. This gives the same-distance classifier, the reverse zero-depth row, and the two central endpoint classifiers $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$.

Peel the forward visible-context witness with Definition 57.13 to recover the unary carrier rows for p and q. These carriers, together with the endpoint classifiers from the collapse theorem, are exactly the right-hand data in Definition 57.389; hence they build $\mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, \mathsf{e} \cdot q, (p \cdot \mathsf{e}) \cdot q)$.

Apply Definition 57.305 to the same hypotheses to obtain the reverse contextual empty-boundary witness. Conjoining the forward package, the reverse package, the reverse zero-depth row, and the two endpoint classifiers gives the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth two sided empty boundary package`

The ten-field fit is: traditional target metric zero-depth symmetric boundary packaging; BEDC source BHist with MetricDistanceWitness and MetricDistanceDepth; carrier MetricDistanceWitness; classifier hsame; stability zero-depth endpoint collapse; ledger the two symmetric visible contexts; core theorem Theorem 57.306; exactness two-sided empty-boundary packaging; standard bridge none claimed; status ProofTarget.

Theorem 57.307 (Metric prefix-independent symmetric zero-depth continuation splice)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, r, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance make any continuation through the reverse central*

distance splice across the empty boundary:

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{Cont}(l, r, out).$$

Proof. The symmetric zero-depth collapse theorem gives $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$. The depth-zero classifier theorem converts this to $\mathsf{hsame}(d_{yx}, \mathsf{e})$. Case analysis reduces the left continuation to the right-unit case, identifying mid with l, and the remaining continuation is exactly the displayed splice result. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth continuation splice`

Theorem 57.308 (Metric prefix-independent symmetric zero-depth witness splice)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, r, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance turn any unary continuation through the reverse central distance into a metric distance witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{UnaryHistory}(l) \to \mathsf{UnaryHistory}(r) \to \mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, r, out) \to \mathsf{MetricDistanceWitness}(l, r, out).$$

Proof. The continuation-splice theorem gives $\mathsf{Cont}(l, r, out)$ from the two visible-context witnesses, the zero-depth row, and the two displayed continuation ledgers through d_{yx}. The supplied unary histories for l and r, together with unary continuation closure, give the unary carrier row for out. These three carrier rows and the spliced continuation are exactly the metric distance witness package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth witness splice`

Theorem 57.309 (Metric symmetric zero-depth distance continuation result)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}$ symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance make any two-step continuation through the two central distances return to the original left endpoint:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{xy}, mid) \to \mathsf{Cont}(mid, d_{yx}, out) \to \mathsf{hsame}(out, l).$$

Proof. The forward zero-depth hypothesis converts d_{xy} to the empty history by the depth-zero classifier theorem. The symmetric zero-depth collapse theorem also gives zero depth for d_{yx}, hence $d_{yx}\mathsf{hsamee}$. Case analysis on these two empty-distance classifiers reduces both continuations to right-unit continuations, first identifying mid with l and then out with mid. Transitivity gives the displayed $\mathsf{hsame}(out, l)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth distance continuation result`

Theorem 57.310 (Metric symmetric zero-depth reverse distance continuation result)**.** *For histories $p, q, p', q', x, y, d_{xy}, d_{yx}, l, mid, out$, symmetric visible-context distance witnesses with independent contexts and zero-depth forward distance also make the reverse-ordered two-step continuation return to the original left endpoint:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \to \mathsf{MetricDistanceWitness}(p' \cdot y, x \cdot q', (p' \cdot d_{yx}) \cdot q') \to \mathsf{MetricDistance}$$
$$\mathsf{Cont}(l, d_{yx}, mid) \to \mathsf{Cont}(mid, d_{xy}, out) \to \mathsf{hsame}(out, l).$$

Proof. Apply Definition 57.304 to the two visible-context witnesses and the zero-depth hypothesis for d_{xy}. This gives $\mathsf{MetricDistanceDepth}(d_{yx}) = 0$.

Now apply Definition 57.309 with the two visible contexts swapped: p', q', y, x, d_{yx} play the forward roles and p, q, x, y, d_{xy} play the reverse roles. The recovered zero-depth row for d_{yx} is the forward depth hypothesis in that swapped instance, and the displayed ledgers are exactly $\mathsf{Cont}(l, d_{yx}, mid)$ followed by $\mathsf{Cont}(mid, d_{xy}, out)$.

Thus the swapped distance-continuation result returns $\mathsf{hsame}(out, l)$. The fit records traditional target zero-distance round trip, BEDC source BHist with Cont and MetricDistanceWitness, carrier the visible-context witness pair, classifier hsame, stability from zero-depth collapse and continuation unit laws, ledger the two central distance histories, core theorem this reverse-order round trip, exactness boundary-unit cancellation, standard bridge zero metric distance as identity path, and status ProofTarget. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness prefix independent symmetric zero depth reverse distance continuation result`

Definition 57.311 (MetricSpace real-distance carrier)**.** *The* $\mathsf{Met}^{\uparrow}$ *real-distance carrier for the open Real Algebra and Geometry phase is the finite unary endpoint packet*

$$(x, y, d, r;\ w, b, c, p, a)$$

where x and y are unary BHist *endpoint histories, w is the displayed row* $\mathsf{MetricDistanceWitness}(x, y, d)$*, b is the closed ball-budget row* $\mathsf{MetricBallBudget}_d(x, y; r)$*, c is the classifier row* hsame *on endpoint and result histories, p is the visible* Cont *provenance for the distance witness, and a is the* RealAlgOrder *budget packet consisting of the constant-rational endpoint rows, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness row, ordered-real comparison, addition, transitivity, and addition-monotonicity rows used by the budget predicate.*

The carrier exposes exactly these rows. It does not contain an ambient host metric space, a quotient-real equality, a completeness principle, an open-cover compactness principle, or a topology generator separate from the displayed distance packet.

Lean4 (definition) `BEDC.Derived.MetricUp.MetricspaceRealDistanceCarrier`

Theorem 57.312 (MetricSpace RealAlgOrder positive-radius budget interface)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints in the carrier of Definition 57.311, the a-coordinate supplies exactly the ordered-real fields required by Definition 57.232 and consumed by Definition 57.233: ordered-real comparison, addition, transitivity, and two-sided addition-monotonicity. The interface is read through the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *certificate and the apartness distance scope of Definition 57.324, so the positive-radius row is the same finite order-or-apartness row exposed to the metric budget consumer.*

This interface exports no trichotomy branch, quotient-real equality, completeness principle, open-cover compactness, or host metric-space row.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderPositiveRadiusBudgetInterface`

Proof. Project the a-coordinate from Definition 57.311. By construction this coordinate is the RealAlgOrder budget packet attached to the same endpoint histories, distance witness, classifier, continuation, and provenance rows as the metric carrier.

Apply Definition 766.2. Its order stability and ledger-exactness rows identify the comparison, addition, transitivity, and addition-monotonicity data as finite displayed order rows, while Definition 766.7 prevents any trichotomy, quotient-real equality, host real carrier, or completeness coordinate from being read through the order or apartness side.

Next use Definition 57.324 together with Definition 57.315. These theorems identify the metric-facing positive row with the same endpoint BHist histories, positive $\mathsf{DyadicRatCore}^{\uparrow}$ radius or apartness row, hsame transports, Cont routes, Pkg provenance, and NameCert ledger supplied by the scoped $\mathsf{RealAlgOrder}^{\uparrow}$ packet.

Finally compare this inventory with Definition 57.232. The predicate asks only for ordered-real comparison, addition, budget transitivity, and two-sided budget-addition monotonicity, and Definition 57.233 consumes only those displayed fields. Therefore the a-coordinate is precisely the positive-radius budget interface used by the composition theorem, with none of the excluded host principles added. □

Theorem 57.313 (MetricSpace public law package)**.** *The real-distance carrier of Definition 57.311 supports the public* $\mathsf{Met}^{\uparrow}$ *distance laws needed by the open phase:*

1. *symmetry, by transporting the endpoint order through* MetricDistanceWitness *symmetry and the RealAlgOrder apartness symmetric-separation row;*
2. *identity boundary, by the empty-distance row and the empty-boundary visible-context rows;*
3. *triangle-budget composition, by the visible-context triangle boundary and the* MetricBallBudget *composition row;*
4. *carrier transport, by* hsame *transport of endpoints and result histories together with the displayed* Cont *provenance.*

These laws are public laws of the finite BEDC packet only. They do not assert an arbitrary host metric, completeness, compactness, or a hidden equality on real distances.

Proof. For symmetry, read the distance row through Definition 57.9. The positive real-distance side is kept in the same packet by Definition 57.327, so swapping the endpoints changes only the orientation of the unary endpoint histories.

For the identity boundary, apply Definition 57.7. The visible-context empty-boundary rows then show that a zero public distance read is exactly the empty endpoint boundary up to hsame, with no quotient-real equality added.

For triangle-budget composition, use Definition 57.134 to form the composite witness over the same endpoint and continuation rows. Then apply Definition 57.233 to compose the two displayed budget rows.

For carrier transport, use the endpoint and result hsame rows already contained in Definition 57.311. The visible Cont provenance identifies the transported witness with the same public distance packet, so every exported law remains inside the carrier. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealDistanceCarrier public law package`

Theorem 57.314 (MetricSpace open-phase consumer interface)**.** *Every* $\mathsf{S}^{1\uparrow}$ *consumer and every later L10 consumer in the open Real Algebra and Geometry phase reads* $\mathsf{Met}^{\uparrow}$ *real-distance data only through the carrier of Definition 57.311 and the law package of Definition 57.313. The consumer surface is the RealAlgOrder apartness distance scope together with the existing visible-context consumer surface; it exposes no additional host metric-space coordinate, no completion row, and no open-cover compactness row.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceOpenPhaseConsumerInterface`

Proof. Start with Definition 57.324. It identifies the constant-rational endpoint histories, positive apartness or dyadic radius, and visible hsame, Cont, Pkg, and NameCert provenance available to a metric-facing real-distance consumer.

Next apply Definition 57.237. That consumer theorem restricts $\mathsf{S}^{1\uparrow}$ and the named downstream metric consumers to the visible-context distance witness, ball-budget, classifier, and finite provenance rows.

The L10 consumers in the open phase can only compose with the same RealAlgOrder apartness scope and the same visible-context public surface. Therefore their readable interface is exactly Definition 57.311 equipped with Definition 57.313, and no host metric, completion, or open-cover compactness coordinate is exported. □

Theorem 57.315 (MetricSpace RealAlgOrder positive-distance public correspondence)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints, the positive component of the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311 is equivalent to the finite apartness packet isolated by Definition 57.324 and Definition 57.325. Concretely, to read a positive public distance row is to read the two endpoint* BHist *histories, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness row, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *ledger supplied by the* RealAlgOrder *constant-rational boundary; conversely, those same rows repack as the positive component of the carrier.*

Endpoint reversal is supplied by Definition 57.327. The correspondence introduces no trichotomy, completion, quotient-real equality, open-cover compactness, or host equality.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderPositiveDistancePublicCorrespondence`

Proof. Project the carrier of Definition 57.311. Its positive side is not a separate host metric object; it is the displayed RealAlgOrder budget packet attached to the distance witness, classifier, continuation, package, and name-certificate rows.

For the forward direction, read that budget packet through Definition 57.324. Then apply Definition 57.325 to identify the positive public distance row with the finite apartness data available to a metric-facing consumer.

For the reverse direction, start with the finite apartness packet named by the same two theorems. The rows already match the carrier fields: endpoint histories, positive dyadic or apartness data, hsame transports, Cont routes, Pkg provenance, and the NameCert ledger. Repacking these rows with the carrier's displayed distance witness gives the positive component of the public real-distance carrier.

Finally use Definition 57.327 for endpoint reversal. Symmetry changes only the orientation of the two endpoint histories, so the same finite packet remains the public positive-distance witness and no trichotomy, completion, quotient-real equality, open-cover compactness, or host equality is added. □

Theorem 57.316 (MetricSpace apartness positive-distance triangle boundary)**.** *Let x, y, and z be constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints read through the* $\mathsf{Met}^{\uparrow}$ *real-distance public surface. Suppose the adjacent positive distance reads for (x, y) and (y, z) are represented by the finite* RealAlgOrder *apartness packets isolated in Definition 57.315. Then the triangle-boundary consumer read for the outer pair (x, z) factors through exactly those endpoint* BHist *histories, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness rows, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the metric triangle ledger of Definition 57.313.*

The resulting consumer row exports only this visible finite boundary. It does not export trichotomy, open-cover compactness, completed-real equality, or a host metric-space structure.

Lean4 ✓ BEDC.Derived.MetricUp.MetricspaceApartnessPositiveDistanceTriangleBoundary

Proof. Apply Definition 57.315 to the adjacent packets (x, y) and (y, z). Each read gives the same kind of finite public data: endpoint histories, positive dyadic or apartness rows, displayed hsame transports, continuation routes, package provenance, and local name-certificate rows.

Use Definition 57.327 whenever an adjacent endpoint orientation must be read from the opposite side. This changes only the order in which the two endpoint histories are presented; it does not add trichotomy, completed-real equality, compactness, or host metric structure.

Finally apply Definition 57.313. Its triangle row is already a visible-context ledger over the public distance witness, so the two adjacent positive packets repack as the outer triangle-boundary consumer read for (x, z). The repacking uses only the displayed BHist, hsame, Cont, Pkg, NameCert, and metric-law rows listed in the statement. □

Theorem 57.317 (MetricSpace SOne RealAlgOrder consumer factorization)**.** *Every* $\mathsf{S}^{1\uparrow}$ *metric consumer read whose two real coordinates are read through the constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary factors through the finite rows of Definition 57.311. The only readable rows are the endpoint* BHist *rows of the two coordinates, the displayed* MetricDistanceWitness *row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness rows supplied by the* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary, the* hsame *transport rows,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the triangle ledger of Definition 57.313.*

Thus the $\mathsf{Met}^{\uparrow}$ *consumer surface seen from* $\mathsf{S}^{1\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *is a finite BEDC packet. It does not include trichotomy, completed-real equality, open-cover compactness, or a host metric-space structure.*

Lean4 ✓ BEDC.Derived.MetricUp.MetricspaceSoneRealalgorderConsumerFactorization
↪ variant BEDC.Derived.MetricUp.MetricspaceS1RealalgorderConsumerFactorization

Proof. Project the admitted circle point through Definition 58.59. The public bridge exposes only the two $\mathsf{Real}^{\uparrow}$ component rows, their product ledger, the unit-equation row, and the displayed hsame, Cont, Pkg, and NameCert rows.

Apply Definition 57.315 to each constant-rational component read. This replaces every positive metric-distance demand by the visible $\mathsf{RealAlgOrder}^{\uparrow}$ apartness packet, with endpoint histories, positive dyadic or apartness data, and the same transport and package rows as in the carrier.

Next use Definition 57.316. Its triangle-boundary row is exactly the finite metric-law ledger over the positive packets, so the $\mathsf{S}^{1\uparrow}$ metric consumer has no additional distance coordinate beyond the carrier and public-law rows.

Finally compare with Definition 58.61. The observation boundary already prevents quotient-circle, angle, compactness, connectedness, and metric-completeness rows from entering the read. Since the $\mathsf{RealAlgOrder}^{\uparrow}$ correspondence contributes no trichotomy or completed-real equality and the $\mathsf{Met}^{\uparrow}$ law package contributes no host metric space, the resulting consumer factors through exactly the finite packet listed in the statement. □

Theorem 57.318 (MetricSpace real-distance scoped certificate)**.** *The finite* $\mathsf{Met}^{\uparrow}$ *scoped certificate for the open real-algebra and geometry phase consists exactly of the real-distance carrier of Definition 57.311, the public law package of Definition 57.313, the open-phase consumer interface of Definition 57.314, the* RealAlgOrder *positive-distance correspondence of Definition 57.315, and the apartness triangle boundary of Definition 57.316.*

Its rows are only the displayed BHist *endpoint histories,* hsame *transport rows,* Cont *routes,* Pkg *provenance, local* NameCert *registrations, the public metric-law rows, and the displayed* RealAlgOrder

positive-distance budget rows. The certificate exports no host metric-space primitive, quotient-real equality, completion, open-cover compactness, or trichotomy.

Proof. Begin with the carrier of Definition 57.311. It supplies the endpoint histories, the displayed distance witness, the transport and continuation routes, package provenance, and the local name-certificate row.

Attach the public law package of Definition 57.313 and read it through Definition 57.314. This gives the open phase only the finite law rows for symmetry, separation, and triangle composition over the same carrier fields.

Next insert the RealAlgOrder side by Definition 57.315. That theorem identifies the positive-distance consumer row with the displayed apartness budget packet, so no second positivity primitive or completed-real coordinate is introduced.

Finally compose with Definition 57.316. The triangle-boundary read uses the same endpoint histories, positive-distance budget rows, hsame, Cont, Pkg, and NameCert data already listed above. Since none of the cited components contains a host metric, quotient-real equality, completion, open-cover compactness, or trichotomy row, the assembled scoped certificate has no coordinate from which such data could be exported. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealDistanceScopedCertificate`

Theorem 57.319 (MetricSpace RealAlgOrder triangle consumer exhaustion)**.** *Every* $\mathsf{Met}^{\uparrow}$ *triangle-consumer read whose endpoints are constant rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoints and whose distance data are read through the real-distance carrier of Definition 57.311 is exhausted by the finite rows already displayed in that carrier, the public law package of Definition 57.313, the positive-distance correspondence of Definition 57.315, and the apartness triangle boundary of Definition 57.316.*

Equivalently, a triangle consumer over endpoints (x, y, z) *can read only the three endpoint* BHist *histories, the two adjacent positive* $\mathsf{RealAlgOrder}^{\uparrow}$ *packets, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the finite metric-law triangle ledger. It has no coordinate for trichotomy, quotient-real equality, completion, open-cover compactness, or an ambient host metric-space structure.*

Proof. Project the adjacent positive-distance reads for (x, y) and (y, z) through Definition 57.315. The projection exposes only endpoint histories, positive dyadic or apartness rows, transport rows, continuation routes, package provenance, and local name-certificate data.

Orient the two adjacent endpoint packets by Definition 57.327. Endpoint reversal changes only the order of the displayed unary histories, so it cannot introduce trichotomy, quotient-real equality, completion, compactness, or a host metric-space coordinate.

Compose the oriented packets through Definition 57.316. That boundary theorem is the public triangle read of Definition 57.313, hence the outer consumer is formed from the same two adjacent positive packets and the same finite triangle ledger.

Finally compare the assembled read with Definition 57.318. The scoped certificate lists every exported coordinate of the real-distance public surface. Since the triangle construction above uses only those rows, no additional coordinate remains for a mature metric-space primitive or completed real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderTriangleConsumerExhaustion`

Theorem 57.320 (MetricSpace RealAlgOrder SOne distance route exhaustion)**.** *For any accepted* $\mathsf{S}^{1\uparrow}$ *metric consumer whose coordinates are read through constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *endpoints and whose distance request uses the carrier of Definition 57.311, the public read is exhausted by the*

finite route consisting of the two $\mathsf{S}^{1\uparrow}$ *coordinate histories, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness or positive dyadic rows, the* MetricDistanceWitness *rows, endpoint and result* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the* $\mathsf{Met}^{\uparrow}$ *triangle ledger.*

This route exports no angle parametrization, quotient-circle equality, completed-real equality, open-cover compactness, trichotomy, or host metric-space primitive.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderSoneDistanceRouteExhaustion`

Proof. First apply Definition 57.317. It reduces the accepted $\mathsf{S}^{1\uparrow}$ metric read to the real-distance carrier of Definition 57.311, so the only available coordinate histories are the two visible $\mathsf{S}^{1\uparrow}$ histories and their constant-rational endpoint rows.

Next use Definition 57.315 on the distance request. The positive component of the carrier is exactly the visible $\mathsf{RealAlgOrder}^{\uparrow}$ apartness or positive dyadic packet, together with the same hsame, Cont, Pkg, and local NameCert rows.

For triangle-facing consumers, compose with Definition 57.316 and Definition 57.319. These results identify the remaining public metric-law data with the finite $\mathsf{Met}^{\uparrow}$ triangle ledger over the displayed distance witnesses and endpoint transports.

Finally compare with the $\mathsf{S}^{1\uparrow}$ observation boundary used in Definition 57.317. That boundary excludes angle parametrization, quotient-circle equality, completed real equality, open-cover compactness, trichotomy, and host metric-space primitives from the consumer surface, so the finite route listed in the statement is exhaustive. □

Theorem 57.321 (Metric RealAlgOrder SOne equation-distance factorization)**.** *Every accepted* $\mathsf{S}^{1\uparrow}$ *equation-distance consumer whose component reads pass through* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness or positive dyadic rows and whose metric request uses Definition 57.311 factors through exactly the finite rows listed in Definition 57.320. The same route supplies the equation-consumer boundary and excludes angle parametrization, quotient-circle equality, completed-real equality, trichotomy, open-cover compactness, and host metric primitives.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderSoneEquationDistanceFactorization`

Proof. First apply Definition 57.317. It places the accepted $\mathsf{S}^{1\uparrow}$ equation-distance read on the real-distance carrier, so the visible component histories are the two displayed circle coordinates and their constant-rational $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint rows.

Next use Definition 57.315. The positive-distance component is read only through apartness or positive dyadic rows, the distance witness, endpoint transports, continuation routes, package provenance, and local naming certificates.

For metric-law consumers, compose with Definition 57.319. This keeps the triangle-facing read on the same distance witnesses and the same finite $\mathsf{Met}^{\uparrow}$ ledger.

Finally compare the resulting packet with Definition 57.320. Its displayed BHist, hsame, Cont, Pkg, and NameCert rows already list every coordinate available to the equation consumer, leaving no coordinate for angle parameters, quotient or completed equality, trichotomy, open-cover compactness, or host metric primitives. □

Theorem 57.322 (Metric RealAlgOrder SOne equation-distance symmetry)**.** *Let an accepted* $\mathsf{S}^{1\uparrow}$ *equation-distance consumer factor through constant-rational* $\mathsf{RealAlgOrder}^{\uparrow}$ *coordinate endpoints and the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311. Then the coordinate-swapped equation-distance read factors through the same finite route after coordinate-pair determinacy,* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness metric symmetric separation, and* $\mathsf{Met}^{\uparrow}$ *route exhaustion. The swapped read introduces*

no angle parametrization, quotient-circle equality, completed-real equality, trichotomy, open-cover compactness, or host metric primitive.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderSoneEquationDistanceSymmetry`

Proof. Start with the original accepted equation-distance consumer and apply Definition 57.321. The factorization exposes only the two $\mathsf{S}^{1\uparrow}$ coordinate histories, their constant-rational $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint rows, the real-distance witness, endpoint transports, continuation routes, package provenance, and local name-certificate rows.

For the coordinate swap, use Definition 58.25. The determinacy statement identifies the swapped pair as the same two circle coordinate histories with their order reversed, so no angle parameter or quotient-circle equality is added to the read.

Apply Definition 57.327 to the endpoint rows. This reverses the apartness or positive dyadic separation packet inside $\mathsf{RealAlgOrder}^{\uparrow}$ while preserving the same endpoint histories, the same hsame transports, and the same finite package and certificate data.

Finally compare the swapped packet with Definition 57.320. That exhaustion theorem already lists the complete $\mathsf{Met}^{\uparrow}$ route available to the equation-distance carrier. Hence the coordinate-swapped read stays on the same finite route and leaves no coordinate for completed-real equality, trichotomy, open-cover compactness, or host metric primitives. □

Theorem 57.323 (MetricSpace RealAlgOrder SOne positive-boundary equivalence)**.** *Let constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints be accepted by the* $\mathsf{S}^{1\uparrow}$ *component-and-equation consumer through the* $\mathsf{Met}^{\uparrow}$ *real-distance carrier of Definition 57.311. Then the* $\mathsf{RealAlgOrder}^{\uparrow}$ *positive-boundary row, the* $\mathsf{Met}^{\uparrow}$ *positive-distance row, and the* $\mathsf{S}^{1\uparrow}$ *equation-separation row are equivalent finite BEDC readings of the same packet. The shared packet consists exactly of endpoint* BHist *rows, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or apartness row,* hsame *transports,* Cont *routes,* Pkg *provenance, local* $\mathsf{NameCert}$ *rows, the metric distance witness, and the* $\mathsf{S}^{1\uparrow}$ *component and unit-equation rows. It does not include trichotomy, quotient-real equality, angle quotient, completed-real equality, open-cover compactness, connectedness, or host metric primitives.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderSonePositiveBoundaryEquivalence`

Proof. First apply Definition 57.315. This identifies the $\mathsf{RealAlgOrder}^{\uparrow}$ positive-boundary read with the $\mathsf{Met}^{\uparrow}$ positive-distance row through the visible endpoint histories, apartness or positive dyadic data, transport rows, continuation routes, package provenance, local naming certificates, and the metric distance witness.

Next apply Definition 57.317. The $\mathsf{S}^{1\uparrow}$ component-and-equation consumer can only enter through that real-distance public interface, so its component rows and unit-equation rows read the same endpoints and distance witness rather than a separate angle or quotient-circle carrier.

The route is exhaustive by Definition 57.320. Hence any consumer claiming one of the three boundary views must use the displayed BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, distance, component, and unit-equation rows.

Finally compare with Definition 38.65 and Definition 58.67. These two boundary theorems give the reverse readings from the $\mathsf{RealAlgOrder}^{\uparrow}$–$\mathsf{Met}^{\uparrow}$ side and from the $\mathsf{S}^{1\uparrow}$ equation consumer side. Together they show that the positive-boundary, positive-distance, and equation-separation rows are three readings of one finite certificate packet, with no remaining coordinate for trichotomy, quotient-real equality, angle quotient, completed-real equality, open-cover compactness, connectedness, or host metric primitives. □

Theorem 57.324 (Metric RealAlgOrder apartness distance scope)**.** *For constant-rational* Real$^{\uparrow}$ *endpoints carried by the* RealAlgOrder *constant boundary, the finite packet supplied by Definition 38.46 and read by Definition 304.5 and Definition 304.8 is exactly the* Met$^{\uparrow}$ *positive-separation distance scope.*

The scope consists of the two endpoint histories, the positive DyadicRatCore$^{\uparrow}$ *radius or apartness row, the* hsame *transports, the* Cont *routes, the visible* Pkg *provenance, and the* NameCert *ledger tying the endpoints to their constant-rational source. Repacked as a metric consumer row, these data form the positive-separation side of a* MetricDistanceWitness *boundary and expose no trichotomy, completion, quotient-real equality, open-cover compactness, or host real equality.*

Proof. Project the constant carrier, classifier, algebra, order, and apartness rows from the RealAlgOrder constant boundary. By Definition 38.46, the consumer sees only the two Real$^{\uparrow}$ endpoint histories, their rational source witnesses, the positive dyadic or apartness row, and the displayed hsame, Cont, Pkg, and NameCert provenance.

Apply the scoped certificate Definition 38.43. It keeps the endpoint histories inside the constant-rational boundary and prevents any additional real-order coordinate from being read by the downstream consumer.

Next apply Definition 304.5 and Definition 304.8. These two rows identify the same finite packet as apartness data consumable by metric or order constructions, with the positive row and endpoint readbacks still tied to the same BHist, hsame, Cont, Pkg, and NameCert ledger.

Finally repack the resulting finite separation row as the positive-separation side of the Met$^{\uparrow}$ distance consumer boundary. The metric visible-context and boundary exactness rows are used only to keep the BEDC ledger stable under contextual prefixes and empty-boundary readback; they do not add a coordinate for trichotomy, completion, quotient-real equality, open-cover compactness, or host real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartnessDistanceScope`

Theorem 57.325 (Metric RealAlgOrder apartness positive reflection)**.** *For constant-rational* Real$^{\uparrow}$ *endpoints, any finite packet satisfying Definition 57.324 and the* ApartnessReal$^{\uparrow}$ *metric consumer boundary determines exactly the positive-separation component of the* Met$^{\uparrow}$ *distance witness. Every metric consumer may read the same positive dyadic or apartness row, endpoint histories,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *provenance supplied by the scoped packet. The reflection exports no converse trichotomy, quotient-real equality, completion, or host equality.*

Proof. Start with the constant-rational handoff Definition 38.46. Its packet gives the endpoint histories, the rational source rows, and the positive dyadic or apartness row that can be consumed without reading an additional real-order coordinate.

Apply Definition 38.43. The certificate keeps those rows inside the constant-rational RealAlgOrder surface and records the displayed hsame, Cont, Pkg, and NameCert provenance.

Use Definition 304.5 together with Definition 304.8. These results identify the same positive row as the separation data that an ApartnessReal$^{\uparrow}$ metric consumer may read.

Finally apply Definition 57.324 and project the positive side of the resulting MetricDistanceWitness boundary. This is only a repacking of the displayed finite rows, so the metric consumer gains no coordinate for converse trichotomy, quotient-real equality, completion, or host equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartnessPositiveReflection`

Theorem 57.326 (Metric RealAlgOrder positive-radius distance boundary)**.** *For constant-rational* Real$^{\uparrow}$ *endpoints carried by the scoped* RealAlgOrder$^{\uparrow}$ *boundary, the positive* DyadicRatCore$^{\uparrow}$ *radius*

or apartness row, the two endpoint BHist *rows, the displayed* hsame *transports, the* Cont *routes, visible* Pkg *provenance, and the local* NameCert *rows are exactly the positive-distance boundary of the* MetricDistanceWitness *consumed by* $\mathsf{Met}^{\uparrow}$*. Thus the concrete dyadic-or-apartness row named by the* $\mathsf{RealAlgOrder}^{\uparrow}$ *packet is the positive-distance component read by the metric witness, and every such metric witness component recovers the same finite positive-radius row.*

The boundary is local to the displayed BEDC rows. It contains no trichotomy, quotient-real equality, completion, open-cover compactness, host metric-space structure, or host real equality.

Proof. For the forward direction, start with the scoped constant-rational $\mathsf{RealAlgOrder}^{\uparrow}$ packet. Apply Definition 38.46 to expose exactly the two endpoint histories, their rational source rows, the positive dyadic or apartness row, the hsame transports, the Cont routes, visible Pkg provenance, and the local NameCert ledger. Then apply Definition 304.5 and Definition 57.325. These rows identify the same finite packet as the positive component of the MetricDistanceWitness boundary read by $\mathsf{Met}^{\uparrow}$.

For the reverse direction, begin with a positive-distance boundary of the MetricDistanceWitness whose endpoints are the same constant-rational $\mathsf{Real}^{\uparrow}$ rows. Use Definition 57.312 to read the positive-radius budget through the scoped $\mathsf{RealAlgOrder}^{\uparrow}$ certificate, and then use Definition 57.315 to identify the public positive-distance component with the finite apartness packet of the same endpoint histories, transports, routes, package provenance, and name-certificate rows.

Both directions therefore name the same concrete positive row. The repacking steps only move between the RealAlgOrder, ApartnessReal, and MetricSpace consumer surfaces; they do not introduce a branch for trichotomy, a quotient-real equality, a completion principle, an open-cover compactness principle, a host metric-space structure, or a host real equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderPositiveRadiusDistanceBoundary`
↪ variant `BEDC.Derived.MetricUp.MetricRealAlgOrder positive radius distance boundary`

Theorem 57.327 (RealAlgOrder apartness metric symmetric separation)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the* RealAlgOrder *constant boundary, the positive-separation packet identified by Definition 57.325 remains a valid positive-separation component of the* $\mathsf{Met}^{\uparrow}$ *distance witness after the two endpoint histories are swapped. The swapped orientation keeps the same positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness row, the same finite separation ledger, and the same displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance; only the left and right endpoint histories are read in the reverse order.*

This is a metric-facing consequence of apartness symmetry. It gives the symmetric endpoint orientation required by a distance witness, but it does not add trichotomy, quotient-real equality, completion, or host real equality.

Proof. Start with the constant-rational handoff Definition 38.46. It supplies the two endpoint histories, rational source rows, positive dyadic or apartness row, and visible hsame, Cont, Pkg, and NameCert provenance.

Apply the scoped certificate Definition 38.43. The certificate keeps both endpoint histories inside the same constant-rational RealAlgOrder boundary and does not introduce any additional real-order coordinate.

Use Definition 304.4 to exchange the two endpoint histories and endpoint readbacks. The apartness row, positive radius or separation ledger, and all displayed transport and provenance rows are preserved componentwise.

Finally apply Definition 57.325 to the swapped packet. Since the positive-separation component is the same finite row read in the opposite endpoint order, it is again a valid positive component of the $\mathsf{Met}^{\uparrow}$ distance witness, with no added equality, completion, or trichotomy principle. □

Lean4 ✓ `BEDC.Derived.MetricUp.RealAlgOrderApartnessMetricSymmetricSeparation`

Theorem 57.328 (Metric RealAlgOrder apartness triangle budget)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints* x, y, z *carried by the* RealAlgOrder *constant boundary, two adjacent finite apartness packets for* (x, y) *and* (y, z) *supply the exact visible rows required by the* $\mathsf{Met}^{\uparrow}$ *triangle-budget consumer for the outer pair* (x, z) *after endpoint orientation is normalized. The consumer may read only the three endpoint histories, the two positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness rows, the finite adjacent separation ledgers, the induced outer-pair* BHist *row, the displayed* hsame *transports, the* Cont *routes, the visible* Pkg *provenance, and the local* NameCert *rows attached to the constant-rational sources.*

The statement is a local RealAlgOrder-to-MetricSpace budget law. It exports no trichotomy, quotient-real equality, completion, open-cover compactness, or host metric-space structure.

Proof. Begin with the two adjacent apartness packets. Apply Definition 57.325 to each packet; this exposes exactly the positive-separation rows, endpoint histories, hsame transports, Cont routes, Pkg provenance, and NameCert rows already present in the constant-rational boundary.

If one adjacent packet is presented in the reverse endpoint order, use Definition 57.327 to swap only the endpoint histories. The positive row and finite ledger are unchanged, so the two adjacent packets now share the middle endpoint history y in the orientation read by the triangle consumer.

With the adjacent rows normalized, invoke the $\mathsf{Met}^{\uparrow}$ visible-context triangle boundary and the public law package. These rows concatenate the two finite positive-separation ledgers into the outer-pair triangle budget while preserving the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

No excluded principle is used in the repacking step. Trichotomy, quotient-real equality, completion, open-cover compactness, and a host metric-space structure are not coordinates of the adjacent RealAlgOrder packets or of the visible triangle-budget consumer surface. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricRealalgorderApartness triangle budget`

Theorem 57.329 (RealAlgOrder apartness metric NameCert route)**.** *For constant-rational* $\mathsf{Real}^{\uparrow}$ *endpoints carried by the scoped* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary, the carrier and classifier rows of Definition 38.43, the finite apartness-to-metric handoff of Definition 38.46, the* $\mathsf{ApartnessReal}^{\uparrow}$ *metric handoff of Definition 304.5, the positive-separation reflection of Definition 57.325, the symmetric endpoint separation of Definition 57.327, and the triangle-budget rows of Definition 57.328 assemble into the exact finite* NameCert *route consumable by* $\mathsf{Met}^{\uparrow}$ *and by the* $\mathsf{S}^{1\uparrow}$ *boundary reader.*

The route consists only of endpoint BHist *rows, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius or apartness rows, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows attached to the constant-rational sources and metric consumers. It exports no trichotomy, quotient-real equality, completion, open-cover compactness, or host equality.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceRealalgorderApartnessMetricNameCertRoute`

Proof. First read the two endpoint rows through Definition 38.43. This fixes the scoped $\mathsf{RealAlgOrder}^{\uparrow}$ carrier and classifier surface and keeps the endpoint histories tied to their constant-rational source rows.

Next apply Definition 38.46 and Definition 304.5. These two handoffs present the same finite apartness row as metric-facing positive-separation data, with the visible BHist, hsame, Cont, Pkg, and NameCert provenance unchanged.

Now use Definition 57.325 to package the positive row as the positive component of the $\mathsf{Met}^{\uparrow}$ distance witness, and use Definition 57.327 when the consumer requests the opposite endpoint orientation.

Finally apply Definition 57.328. The triangle route concatenates only the displayed adjacent separation ledgers and therefore gives exactly the finite NameCert route needed by metric and circle-boundary consumers, without adding any excluded host-order, quotient, completion, compactness, or equality principle. □

Theorem 57.330 (Metric distance witness right e1 result shape)**.** *For histories x, y, d, every concrete metric distance witness whose right endpoint is visibly one-headed has a visibly one-headed distance ledger, and the exposed distance tail is again the metric witness for the exposed right endpoint:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \mathsf{MetricDistanceWitness}(x, y, k).$$

Proof. Unpack the distance witness into unary carrier fields and the continuation ledger. The continuation result has the form $d = x \cdot \mathsf{E1}(y)$, which is definitionally $\mathsf{E1}(x \cdot y)$. Choose $k = x \cdot y$. Unary inversion on the visible right endpoint gives the carrier for y, unary append closure gives the carrier for k, and the canonical append equality gives the descended continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result shape`

Theorem 57.331 (Metric distance witness right e1 result not empty)**.** *For histories x, y, d, no concrete metric distance witness whose right endpoint is visibly one-headed can have a distance result history-same to* e*:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{e}) \Longrightarrow \bot.$$

Proof. Apply the right e-one result shape theorem to expose a tail k with $\mathsf{hsame}(d, \mathsf{E1}(k))$. Composing this with $\mathsf{hsame}(d, \mathsf{e})$ gives $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{e})$, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result not empty`

Theorem 57.332 (Metric distance witness right e1 result e0 absurd)**.** *For histories x, y, d, no concrete metric distance witness can have a visibly one-headed right endpoint while its distance result is visibly zero-headed:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E0}(d)) \Longrightarrow \bot.$$

Proof. Apply the right e-one result shape theorem to the witness. It exposes a tail k with $\mathsf{hsame}(\mathsf{E0}(d), \mathsf{E1}(k))$. Constructor no-confusion for generated history sameness rejects this zero-headed versus one-headed comparison. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result e0 absurd`

Theorem 57.333 (Metric distance witness right e1 result tail exactness)**.** *For histories x, y, d, k, if a distance witness has a visibly one-headed right endpoint and its displayed result is history-same as $\mathsf{E1}(k)$, then k is the exact descended distance tail:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(k)) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, k).$$

Proof. Apply right e-one result shape to expose an intermediate tail k_0 together with $\mathsf{hsame}(d, \mathsf{E1}(k_0))$ and the descended metric witness. Compose this classifier with the displayed result classifier, invert the resulting one-headed hsame comparison, and transport the tail witness along the exposed tail equality. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result tail hsame exact`

Theorem 57.334 (Metric distance witness right e1 result tail determinism)**.** *For histories x, y, d, d', two metric distance witnesses with the same source, the same visible right endpoint, and visible results determine the same result tail:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d')) \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Project the continuation ledgers from the two metric witnesses. Continuation determinacy identifies the visible results $\mathsf{hsame}(\mathsf{E1}(d), \mathsf{E1}(d'))$, and one-constructor inversion descends this classifier to the tail relation $\mathsf{hsame}(d, d')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result tail deterministic`

Theorem 57.335 (Metric distance witness right e1 source shape)**.** *For histories x, y, d, every concrete metric distance witness whose right endpoint is visibly one-headed exposes a visible distance tail and the matching empty-or-visible source alternative:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), d) \Longrightarrow \exists k : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(k)) \wedge \Big((x = \mathsf{e} \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(k, y)) \vee \exists x_1 : \mathsf{E}$$

Proof. Apply Definition 57.330 to the given witness. This produces a history k, a displayed result comparison $\mathsf{hsame}(d, \mathsf{E1}(k))$, and the descended witness $\mathsf{MetricDistanceWitness}(x, y, k)$.

We next rebuild the displayed-result witness with result $\mathsf{E1}(k)$. Keep the original source and right endpoint carriers, close the descended tail carrier under $\mathsf{E1}$, and transport the original continuation field along $\mathsf{hsame}(d, \mathsf{E1}(k))$.

Apply Definition 57.94 to this displayed-result witness. In the empty branch, keep the same k and record $x = \mathsf{e}$, the unary carrier for y, and $\mathsf{hsame}(k, y)$. In the visible branch, keep the same k and the exposed x_1, together with $x = \mathsf{E1}(x_1)$ and the descended metric witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 source shape`

Theorem 57.336 (Metric distance witness right e1 result iff)**.** *For histories x, y, d, a distance witness whose right endpoint and result both carry one visible constructor is equivalent to the descended tail witness:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \Longleftrightarrow \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. The forward direction is right one-headed result exactness. For the reverse direction, keep the source unary carrier, close the right endpoint and result carriers under $\mathsf{E1}$, and lift the continuation ledger by the one-step continuation rule. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result iff`

Theorem 57.337 (Metric distance witness right e1 result hsame source)**.** *For histories x, x', y, d, a distance witness whose right endpoint and result both carry one visible constructor has the same empty-or-visible source readback after source transport:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(x, x') \Longrightarrow (x' = \mathsf{e} \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, y))$$

or there exists $x_1 : \mathsf{BHist}$ *such that*

$$x' = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d).$$

Proof. Apply the right one-headed source case split before transport. In the empty branch, empty hsame inversion identifies x' with e. In the visible branch, visible hsame inversion exposes the transported tail, and unary and continuation transport rebuild the descended metric witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result hsame source`

Theorem 57.338 (Metric distance witness right e1 result hsame target)**.** *For histories x, y, y', d, a distance witness whose right endpoint and result both carry one visible constructor has the same empty-or-visible source readback after target-tail transport:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(y, y') \Longrightarrow (x = \mathsf{e} \wedge \mathsf{UnaryHistory}(y') \wedge \mathsf{hsame}(d, y'))$$

or there exists $x_1 : \mathsf{BHist}$ *such that*

$$x = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d).$$

Proof. Apply the right one-headed source case split before target transport. In the empty-source branch, transport the unary carrier and compose the displayed result-tail comparison with $\mathsf{hsame}(y, y')$. In the visible-source branch, keep the exposed source tail and transport the target carrier and continuation ledger across the target-tail classifier, leaving the source and result endpoints fixed. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result hsame target`

Theorem 57.339 (Metric right e-one result two-sided endpoint transport)**.** *For histories x, x', y, y', d, simultaneous source and target-tail transport preserves the empty-or-visible readback of a right e-one metric result:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{E1}(y), \mathsf{E1}(d)) \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y') \Longrightarrow (x' = \mathsf{e} \wedge \mathsf{UnaryHistory}(y') \wedge \mathsf{hsame}(d, y'))$$

or there exists $x_1 : \mathsf{BHist}$ *such that*

$$x' = \mathsf{E1}(x_1) \wedge \mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d).$$

Proof. First apply Definition 57.337 to the displayed witness and the supplied $\mathsf{hsame}(x, x')$. This gives the transported source readback with target tail y still fixed.

In the empty-source branch, keep $x' = \mathsf{e}$. Transport the unary carrier from y to y' along $\mathsf{hsame}(y, y')$, and compose the displayed comparison $\mathsf{hsame}(d, y)$ with $\mathsf{hsame}(y, y')$ to obtain $\mathsf{hsame}(d, y')$.

In the visible-source branch, keep the exposed x_1 and the equality $x' = \mathsf{E1}(x_1)$. The descended witness $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y, d)$ supplies the source, target, and result carriers; unary transport moves the target carrier from y to y', while metric continuation transport carries the

continuation ledger across the same target-tail classifier with source and result fixed. Repacking these fields gives $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x_1), y', d)$.

Ten-field fit: traditional target two-sided visible endpoint transport; BEDC source BHist, hsame, and MetricDistanceWitness; carrier unary metric endpoints; classifier hsame; stability source and target-tail transport; ledger exposed right e-one distance tail; core theorem Definition 57.339; exactness simultaneous transported source readback; standard bridge finite unary metric endpoint transport; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness right e1 result two sided endpoint transport`

Theorem 57.340 (Metric e-one pair result tail readback)**.** *For histories x, y, d, a metric distance witness whose source, target, and result are all visibly one-headed exposes unary source and target tails and the tail continuation ledger:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, \mathsf{E1}(x) \cdot y).$$

Proof. Apply right e-one result exactness to descend the target and result to $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$. Unary inversion on the source carrier gives unary evidence for x, the descended witness carries unary evidence for y, and its continuation ledger is exactly $d\mathsf{hsameE1}(x) \cdot y$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result tail readback`

Theorem 57.341 (Metric e-one pair result tail exactness iff)**.** *For histories x, y, d, a metric distance witness whose source, target, and result are all visibly one-headed is equivalent to unary tail data together with the exposed continuation ledger:*

$$\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d)) \Longleftrightarrow \mathsf{UnaryHistory}(x) \wedge \mathsf{UnaryHistory}(y) \wedge \mathsf{hsame}(d, \mathsf{E1}(x) \cdot y).$$

Proof. For the forward direction, apply Theorem 57.340. It descends the visible target and result constructors and returns exactly the two unary tail carriers and the displayed continuation ledger.

For the reverse direction, start from the unary tail carriers and the hsame ledger. Unary closure builds the source carrier $\mathsf{UnaryHistory}(\mathsf{E1}(x))$, and the continuation ledger transports the append result so that d is the concrete distance from $\mathsf{E1}(x)$ to y. This gives the central witness $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), y, d)$.

Apply Theorem 57.294 to this central witness. The visible target constructor and the matching visible result constructor are restored, producing $\mathsf{MetricDistanceWitness}(\mathsf{E1}(x), \mathsf{E1}(y), \mathsf{E1}(d))$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result iff`
Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness e1 pair result tail exactness iff`

The ten-field fit is: traditional target one-headed metric tail exactness; BEDC source BHist, UnaryHistory, Cont, and MetricDistanceWitness; carrier concrete metric distance witnesses; classifier hsame; stability visible-constructor closure and continuation transport; ledger exposed tail distance; core theorem Theorem 57.341; exactness bidirectional tail data; standard bridge finite metric endpoint readback; status ProofTarget.

Theorem 57.342 (MetricSpace SOne distance source boundary)**.** *Every accepted $\mathsf{S}^{1\uparrow}$ packet over the $\mathsf{Real}^{\uparrow}$ product carrier reads $\mathsf{Met}^{\uparrow}$ distance data only through the finite rows of Definition 57.311 and the public law package of Definition 57.313. Its circle-side source is constrained by Definition 58.59, Definition 58.61, and Definition 58.62. Its $\mathsf{RealAlgOrder}^{\uparrow}$ side is constrained by Definition 57.324 and Definition 38.63.*

Thus a $\mathsf{S}^{1\uparrow}$ *consumer of real-distance data can read only the two* $\mathsf{Real}^{\uparrow}$ *component rows, the product-carrier ledger, the unit-equation row, componentwise* hsame *classifiers, equation transport, finite-window* RegSeqRat *stream-name and real-seal rows, the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness budget rows, the metric endpoint histories,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the public metric-law rows. The boundary exports no angle quotient, covering-space quotient, host metric function, completion row, compactness row, connectedness row, or topology generator.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceSoneDistanceSourceBoundary`
↪ variant `BEDC.Derived.MetricUp.MetricspaceSOneDistanceSourceBoundary`

Proof. Project the accepted $\mathsf{S}^{1\uparrow}$ packet through Definition 58.59. This exposes the two $\mathsf{Real}^{\uparrow}$ component rows, the product carrier, the unit-equation witness, the componentwise classifier rows, and equation transport, and excludes the circle-specific quotient and topology rows named in the statement.

Refine the two real components through Definition 58.61 and Definition 58.62. The metric-facing source rows are then finite-window RegSeqRat and real-seal observations attached to the same public certificate, rather than an ambient circle metric or a hidden completion object.

Pass the real-algebra side through Definition 57.324 and Definition 38.63. These two boundaries keep the apartness and unit-equation reads tied to the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint and budget rows, so the metric source has no trichotomy, quotient-real equality, host metric, or Real completeness coordinate.

Repack the resulting endpoint rows, budget rows, transport rows, continuation routes, provenance rows, and local certificate rows as the carrier of Definition 57.311. Attach Definition 57.313 and read the package through Definition 57.314. The open real-algebra and geometry phase therefore consumes exactly the finite real-distance carrier and its public law rows, with no additional metric, topology, compactness, connectedness, completion, angle, or covering quotient data. □

Theorem 57.343 (Metric seed closure surface)**.** *The* $\mathsf{Met}^{\uparrow}$ *seed closure surface is the finite public package assembled from the metric carrier of Definition 57.1, the boundary exactness rows of Definition 57.211 and Definition 57.212, the real-distance public interface of Definition 57.311, the triangle continuation rows of Definition 57.356, and the transport surface of Definition 57.347. A seed consumer of* $\mathsf{Met}^{\uparrow}$ *can read only these carrier, boundary, distance, triangle, and transport rows together with their* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Proof. Project the metric carrier first. The carrier fixes the endpoint histories, distance witness rows, classifier, continuation, provenance, and local naming coordinates available to any public metric read.

Boundary exactness then identifies the endpoint and empty-boundary cases already named by the carrier. The real-distance public interface supplies the displayed ordered-real budget rows used by downstream consumers.

The triangle surface is the visible continuation package: triangle append closure and the associated middle-continuation rows keep composite distance reads inside the same finite witness family. Transport then moves endpoints, witnesses, budgets, and provenance componentwise through the displayed hsame and Cont rows.

No additional host metric, topology generator, compactness principle, completeness row, or quotient real equality appears in these referenced surfaces. Hence the named carrier, boundary, distance, triangle, and transport files form the seed closure surface for $\mathsf{Met}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricSeedClosureSurface public package`

The $\mathsf{Met}^{\uparrow}$ seed surface is the carrier, distance witness, boundary exactness, real-distance interface, triangle continuation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row described by Definition 57.343. Its binding closure registration is recorded in the obligation-status child file, where the same checked Lean target is read independently from the paper-side closure grade.

Theorem 57.344 (MetricSpace L10 sibling lattice route)**.** *Every* $\mathsf{Met}^{\uparrow}$ *real-distance consumer used by the current L10 Real and Completion route factors through the finite sibling surfaces already present in the metric child files:* **??**, *chapter 58, chapter 38, chapter 126, chapter 301, and chapter 291. The lattice route is the displayed finite read through LocatedReal endpoints, SOne component routes, the Real public-distance certificate, StreamName finite windows, RegSeqRat regular-source rows, and DyadicRatCore endpoint and budget ledgers.*

The route introduces no host metric function, quotient real equality, selected limit, open-cover compactness, or metric-completion datum.

Lean4 ✓ `BEDC.Derived.MetricUp.MetricspaceL10SiblingLatticeRoute finite packet`

Proof. First place the located-real side on its finite source boundary. Apply Definition 57.302. The distance consumer is then read only through accepted $\mathsf{LocatedReal}^{\uparrow}$ packets, their $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{RegSeqRat}^{\uparrow}$ source windows, and DyadicRatCore endpoint and radius ledgers, together with visible BHist, hsame, Cont, Pkg, and NameCert rows.

Next place the circle-side metric consumer on its finite component route. Apply Definition 57.342. This keeps the $\mathsf{S}^{1\uparrow}$ read on the two $\mathsf{Real}^{\uparrow}$ component rows, the unit-equation row, componentwise classifiers, finite-window stream and regular-source rows, and the same visible provenance surface.

Now attach the Real-facing certificate by Definition 57.318. Its certificate is exactly the real-distance carrier, the public metric laws, the RealAlgOrder positive-distance correspondence, and the apartness triangle boundary. Therefore the metric consumer reaches the L10 Real face through the same finite packet rather than through a separate real equality or completion coordinate.

Finally apply Definition 57.313. The exported metric laws are symmetry, identity boundary, triangle-budget composition, and carrier transport over the displayed packet. Since the LocatedReal and SOne source boundaries already factor through the named Real, StreamName, RegSeqRat, and DyadicRatCore sibling faces, the assembled L10 route is the finite sibling lattice listed in the statement. None of the cited rows contains a host metric function, quotient real equality, selected limit, open-cover compactness, or metric-completion datum, so none is introduced by the factorization. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricSpaceL10SiblingLatticeRoute`

Lemma 57.345 (Metric L10 sibling dependency route)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer in the L10 route reads distance rows through the finite sibling dependency surface already named by the metric packet:* $\mathsf{Real}^{\uparrow}$ *distance endpoints from chapter 38,* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison rows or apartness rows when positive distance is inspected,* $\mathsf{S}^{1\uparrow}$ *rows where circle distances are consumed through chapter 58, and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows where modulus transport is needed through chapter 130. The route is a finite displayed dependency route, not a host metric space, completed distance function, quotient real equality, or ambient continuity theorem.*

Lean4 ✓ `BEDC.Derived.MetricUp.MetricL10SiblingDependencyRoute surface`

Theorem 57.346 (Metric distance witness continuation hsame transport)**.** *For histories* x, x', y, y', d, d' : BHist*, history-sameness of the two endpoints and of the distance result transports the continuation field exposed by a metric distance witness:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{Cont}(x', y', d').$$

Proof. Case-reduce the three supplied hsame witnesses. The transported endpoints and result are then definitionally the endpoints and result of the original metric distance witness. Project its continuation field $\mathsf{Cont}(x, y, d)$, which is the required transported continuation ledger. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness cont hsame transport`

Theorem 57.347 (Metric distance witness hsame field transport)**.** *For histories* x, x', y, y', d, d' : BHist*, history-sameness of the two endpoints and of the distance result transports all exposed fields of a metric distance witness:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{UnaryHistory}(x') \wedge \mathsf{UnaryHisto}$$

Proof. Case-reduce the three supplied hsame witnesses. The transported source, target, and distance histories are then definitionally the original fields. Project the three unary carrier fields and the continuation ledger from the metric distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame fields transport`

Theorem 57.348 (Metric distance witness hsame transport depth add)**.** *For histories* x, x', y, y', d, d' : BHist*, history-sameness of the two endpoints and of the distance result transports the continuation field and preserves the append-depth ledger:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{Cont}(x', y', d') \wedge \mathsf{MetricDistanc}$$

Proof. Case-reduce the three supplied hsame witnesses. The transported histories are then the source, target, and distance of the original metric witness. Project its continuation field and apply the metric distance depth-add theorem to the same witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame transport cont depth add`

Theorem 57.349 (Metric distance witness hsame source determinacy)**.** *For histories* x, x', y, y', d, d' : BHist*, if the target and distance ledgers classify by* hsame *and the two concrete metric distance witnesses hold,*

$$\mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(x', y', d'),$$

then the witnesses expose both continuation fields and determine the source:

$$\mathsf{Cont}(x, y, d) \wedge \mathsf{Cont}(x', y', d') \wedge \mathsf{hsame}(x, x').$$

Proof. Project the continuation fields from the two metric distance witnesses. After case-reducing the target and distance hsame witnesses, the continuations share the same right context and result. Continuation right cancellation then identifies the two sources. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame source deterministic`

Theorem 57.350 (Metric distance witness hsame target determinacy)**.** *For histories* x, x', y, y', d, d' : BHist*, if the source and distance ledgers classify by* hsame *and the two concrete metric distance witnesses hold,*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d) \Longrightarrow \mathsf{MetricDistanceWitness}(x', y', d'),$$

then the witnesses expose both continuation fields and determine the target:

$$\mathsf{Cont}(x, y, d) \wedge \mathsf{Cont}(x', y', d') \wedge \mathsf{hsame}(y, y').$$

Proof. Project the continuation fields from the two metric distance witnesses. After case-reducing the source and distance hsame witnesses, the continuations share the same left context and result. Continuation left cancellation then identifies the two targets. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness hsame target deterministic`

Theorem 57.351 (Metric distance witness visible-context hsame source determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, d'$: BHist*, if the visible targets and visible distances have the same central target and distance under* hsame*, then the central sources are determined:*

$$\mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot d') \cdot q') \Longrightarrow \mathsf{hsame}(x, x').$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', d')$. The supplied hsame relations on y, y' and d, d' then fit Definition 57.349, whose source component is the displayed $\mathsf{hsame}(x, x')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context hsame source deterministic`

Theorem 57.352 (Metric distance witness visible-context hsame target determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, d'$: BHist*, if the visible sources and visible distances have the same central source and distance under* hsame*, then the central targets are determined:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d, d') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot d') \cdot q') \Longrightarrow \mathsf{hsame}(y, y').$$

Proof. Apply Definition 57.13 to both visible-context witnesses. This exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', d')$. The supplied hsame relations on x, x' and d, d' then fit Definition 57.350, whose target component is the displayed $\mathsf{hsame}(y, y')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context hsame target deterministic`

Theorem 57.353 (Metric distance witness prefix-independent endpoint-hsame result determinacy)**.** *For histories* $p, q, p', q', x, x', y, y', d, e$: BHist*, if the central endpoints classify by* hsame *while the visible prefixes and suffixes may differ, then the exposed central distance results classify together:*

$$\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(p' \cdot x', y' \cdot q', (p' \cdot e) \cdot q') \Longrightarrow \mathsf{hsame}(d, e).$$

Proof. Apply visible-context exactness to both metric witnesses. This removes the prefix and suffix contexts and exposes central witnesses $\mathsf{MetricDistanceWitness}(x, y, d)$ and $\mathsf{MetricDistanceWitness}(x', y', e)$. The supplied endpoint classifications then fit the endpoint-hsame result determinacy theorem, giving the displayed $\mathsf{hsame}(d, e)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context prefix independent endpoint hsame result deterministic`

Theorem 57.354 (Metric distance witness visible right tail determinacy)**.** *For histories $p, q, p', q', x, y, d, d'$: BHist, if two visible-context metric witnesses share the central source and the right endpoint $\mathsf{E1}(y)$, and their displayed distance results are respectively $\mathsf{E1}(d)$ and $\mathsf{E1}(d')$ inside their visible contexts, then the distance tails classify together:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, \mathsf{E1}(y) \cdot q, (p \cdot \mathsf{E1}(d)) \cdot q) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(p' \cdot x, \mathsf{E1}(y) \cdot q', (p' \cdot \mathsf{E1}(d')) \cdot q') \Longrightarrow \mathsf{hsame}(d, d').$$

Proof. Visible-context exactness removes the outer prefixes and suffixes, exposing two central metric witnesses with right endpoint $\mathsf{E1}(y)$ and results $\mathsf{E1}(d)$ and $\mathsf{E1}(d')$. Right-$\mathsf{E1}$ result exactness descends both witnesses to their tails, after which continuation determinism identifies the tail distance results. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context right e1 result tail deterministic`

Theorem 57.355 (Metric distance witness visible-context semantic name certificate)**.** *For unary visible contexts p, q and unary central endpoints x, y, the contextual distance predicate $d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q)$ carries a semantic naming certificate with classifier hsame:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{UnaryHistory}(x) \Longrightarrow \mathsf{UnaryHistory}(y) \Longrightarrow$$
$$\mathsf{SemanticNameCert}(d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d) \cdot q))\,(d \mapsto \mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d$$

Proof. The inhabited carrier is the central distance $x \cdot y$, lifted through visible-context exactness. The classifier is hsame, so its equivalence laws are the kernel history-sameness laws. Transport along the classifier is by case analysis on the history-sameness witness, after which the contextual distance predicate is unchanged. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context semanticNameCert`

Theorem 57.356 (Metric distance witness triangle append closure)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a distance witness from x to y, a distance witness from y to z, and a distance witness from d_{xy} to z close the appended triangle ledger:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x \cdot d_{yz}).$$

Proof. Unpack the three distance witnesses to their continuation fields. The three continuations identify d_{xy} with $x \cdot y$, d_{yz} with $y \cdot z$, and d_{xyz} with $(x \cdot y) \cdot z$. Associativity of history append classifies this last ledger with $x \cdot (y \cdot z)$, hence with $x \cdot d_{yz}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append closed`

Theorem 57.357 (Metric distance witness unary append commutation package)**.** *For unary histories x and y, the appended endpoints $x \cdot y$ and $y \cdot x$ carry the canonical metric distance witness $(x \cdot y) \cdot (y \cdot x)$, and the endpoints are hsame:*

$$\mathsf{MetricDistanceWitness}(x \cdot y, y \cdot x, (x \cdot y) \cdot (y \cdot x)) \wedge \mathsf{hsame}(x \cdot y, y \cdot x).$$

Proof. Unary append closure supplies the two endpoint carriers and the displayed distance carrier. The distance ledger is the canonical continuation from $x \cdot y$ to $y \cdot x$, and unary append commutativity gives the endpoint classifier. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness unary append comm package`

Theorem 57.358 (Metric distance witness composite endpoint append result)**.** *For histories $x, x', y, y', z, z', d_{xy}, d_{xyz}$, endpoint comparisons $x\mathsf{hsame}x'$, $y\mathsf{hsame}y'$, and $z\mathsf{hsame}z'$ transport the composite distance ledger to the endpoint-normal append result:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x' \cdot (y' \cdot z')).$$

Proof. Transport along the three endpoint hsame comparisons and unpack the two distance witnesses to their continuation fields. Constructor analysis of the two continuation ledgers reduces the displayed result comparison to associativity of history append. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness composite endpoint hsame append result`

Theorem 57.359 (Metric distance witness triangle continuation middle closure)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a distance witness from x to y, a distance witness from y to z, and a continuation ledger $\mathsf{Cont}(d_{xy}, z, d_{xyz})$ close the middle-composite distance witness:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz}).$$

Proof. The two distance witnesses expose the continuation ledgers $\mathsf{Cont}(x, y, d_{xy})$ and $\mathsf{Cont}(y, z, d_{yz})$. Together with the displayed ledger $\mathsf{Cont}(d_{xy}, z, d_{xyz})$, constructor analysis reduces the resulting source-to-middle continuation to associativity of history append. The unary carrier for d_{xyz} follows from unary continuation closure applied to d_{xy} and z. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle cont middle closed`

Theorem 57.360 (Metric distance witness visible-context triangle continuation middle closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a visible-context witness for the left triangle edge, a central witness from y to z, and a continuation ledger from d_{xy} by z close the central middle-composite distance witness:*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz}).$$

Proof. Apply Definition 57.13 to the visible-context witness. This recovers the central edge witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$. The recovered edge witness, the given y-to-z witness, and the displayed continuation ledger are exactly the inputs of Definition 57.359, which returns the displayed middle-composite distance witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle cont middle closed`

Theorem 57.361 (Metric distance witness triangle middle result determinacy)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}, displayed$, if the witnesses from x to y and from y to z are given, and $\mathsf{Cont}(d_{xy}, z, d_{xyz})$ displays the middle composite, then any distance witness from x to d_{yz} has the same result:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, d_{yz}, displayed) \Longrightarrow \mathsf{hsame}(d_{xyz}, displayed).$$

Proof. Triangle continuation middle closure constructs the canonical witness $\mathsf{MetricDistanceWitness}(x, d_{yz}, d_{xyz})$ from the two edge witnesses and the displayed middle continuation. The displayed witness has the same source and target histories as this canonical witness, so continuation determinacy applied to their distance ledgers identifies the two result histories. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle cont middle result deterministic`

Theorem 57.362 (Metric distance witness triangle append depth add)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, three triangle distance witnesses identify the displayed ledger with $x \cdot d_{yz}$ and compute its structural depth as the sum of the left endpoint depth and the middle-to-right distance depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{MetricDistanceDepth}(d_{xyz}) = \mathsf{MetricDistanceDepth}(x$$

Proof. Triangle append closure supplies the equality $\mathsf{hsame}(d_{xyz}, x \cdot d_{yz})$. Together with the unary carrier fields already present in the three distance witnesses, this equality is a concrete distance witness from x to d_{yz} with result d_{xyz}. The metric depth addition law applied to that central witness gives the displayed depth equation. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append depth add`

Theorem 57.363 (Metric distance witness continuation depth add)**.** *For histories x, y, z, d_{xy}, d_{xyz}, a distance witness from x to y and a displayed continuation from d_{xy} by z compute the final structural distance depth:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{UnaryHistory}(z) \Longrightarrow \mathsf{Cont}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = \mathsf{Metri}$$

Proof. The first witness supplies the unary carrier of d_{xy}. Together with the unary carrier of z and the displayed continuation ledger, this forms a second distance witness from d_{xy} to z with result d_{xyz}. Apply the depth additivity theorem to the first and second witnesses, then associate addition. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness cont depth add`

Theorem 57.364 (Metric distance witness triangle depth-zero collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if three triangle distance witnesses have a final ledger of structural depth zero, then the triangle append equality holds and every displayed endpoint and intermediate distance collapses to* e*:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longrightarrow$$
$$\mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).$$

Proof. Triangle append closure forms the central witness from x to d_{yz} with result d_{xyz}. The depth-zero endpoint theorem for this central witness forces x and d_{yz} to be empty. Empty-distance exactness for the witness from y to z then forces y and z to be empty, and the continuation field in the witness from x to y forces d_{xy} to be empty. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle depth zero collapse`

Theorem 57.365 (Metric triangle left distance empty collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the left distance in a metric triangle is history-same to* e*, then its endpoints collapse and the remaining two displayed distances are exactly the right endpoint:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, z) \wedge \mathsf{hsame}(d_{yz}, z).$$

Proof. Empty-distance exactness for the witness from x to y gives $x\mathsf{hsamee}$ and $y\mathsf{hsamee}$. With d_{xy} collapsed, empty-left exactness for the outer witness gives $d_{xyz}\mathsf{hsame}z$. Collapsing y in the second witness and applying the same empty-left exactness gives $d_{yz}\mathsf{hsame}z$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left distance empty collapse`

Theorem 57.366 (Metric triangle depth-zero iff empty spine)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, three concrete metric distance witnesses identify final structural depth zero exactly with emptiness of the three endpoints and the two displayed edge distances:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &\mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longleftrightarrow \\ &\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. For the forward direction, apply Definition 57.364 to the three triangle witnesses and the displayed zero-depth equation. The theorem returns a larger conjunction containing the append equality as its first component; discard that component and retain precisely the five empty-spine classifiers.

For the reverse direction, assume the five displayed classifiers. Apply Definition 57.252 to the first edge witness, using $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$, to get $\mathsf{MetricDistanceDepth}(d_{xy}) = 0$. Then use Definition 57.253 to recover $\mathsf{hsame}(d_{xy}, \mathsf{e})$ as the certified empty left endpoint for the outer witness. Finally apply Definition 57.252 to $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$, together with $\mathsf{hsame}(d_{xy}, \mathsf{e})$ and $\mathsf{hsame}(z, \mathsf{e})$, to obtain $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle depth zero iff empty spine`

Fit: traditional target triangle degeneracy criterion; BEDC source $BHist$ metric distance witnesses; carrier the three witnessed edges; classifier hsame; stability supplied by existing depth-zero and empty-endpoint lemmas; ledger the two edge distances; core theorem this iff; exactness zero-depth iff empty spine; standard bridge none claimed; status ProofTarget.

Theorem 57.367 (Metric distance witness visible-context triangle depth-zero collapse)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the first triangle edge is displayed through visible source and target ledgers and the final triangle distance has structural depth zero, then the triangle append equality holds and every central endpoint and intermediate distance collapses to* e*:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, p \cdot d_{xy} \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longrightarrow \\ &\mathsf{hsame}(d_{xyz}, x \cdot d_{yz}) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.13 to the contextual first edge. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ while leaving the two remaining triangle witnesses unchanged.

Apply Definition 57.364 to the central witness, the witness from y to z, the witness from d_{xy} to z, and the depth-zero hypothesis. The resulting conjunction is exactly the stated visible-context collapse conclusion. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle depth zero collapse`

Theorem 57.368 (Metric distance witness visible-context triangle depth-zero iff empty spine)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, visible source and target ledgers around the first edge preserve the exact zero-depth triangle criterion:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, p \cdot d_{xy} \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &\mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \Longleftrightarrow \\ &\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, \mathsf{e}) \wedge \mathsf{hsame}(d_{yz}, \mathsf{e}).\end{aligned}$$

Proof. Apply Definition 57.13 to the contextual first edge. This recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, so the three hypotheses now have the same shape as the central triangle criterion.

For the forward direction, apply Definition 57.366 to the central witness, the witness from y to z, and the witness from d_{xy} to z. The forward half turns the displayed zero-depth equation into the five empty-spine classifiers.

For the reverse direction, use the same central iff theorem with the recovered central witness and the two unchanged triangle witnesses. Its reverse half turns the five displayed empty-spine classifiers into $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle depth zero iff empty spine`

Theorem 57.369 (Metric triangle nonempty distance has a nonempty endpoint)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the three triangle distance witnesses hold and the final triangle distance is not history-same as* e*, then at least one endpoint of the triangle is not history-same as* e*:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow ($$

Proof. Apply metric nonempty-distance endpoint exactness to the outer witness $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ If the nonempty endpoint is z, the conclusion follows directly. If it is d_{xy}, apply the same exactness theorem to $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, yielding a nonempty x or y endpoint. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty endpoint nonempty`

Theorem 57.370 (Metric triangle nonempty spine one-headed endpoint cases)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the three triangle distance witnesses hold and the final triangle distance is not history-same as* e*, then at least one triangle endpoint is visibly one-headed with a unary tail:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow (d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longrightarrow \\ &(\exists x_0,\ x = \mathsf{E1}(x_0) \wedge \mathsf{UnaryHistory}(x_0)) \vee (\exists y_0,\ y = \mathsf{E1}(y_0) \wedge \mathsf{UnaryHistory}(y_0)) \vee (\exists z_0,\ z = \mathsf{E1}(z_0) \wedge \mathsf{UnaryHistory}($$

Proof. Apply the nonempty endpoint theorem for metric triangles. In each endpoint case, the corresponding unary carrier is a field of one of the triangle distance witnesses. Unary nonempty one-tail readback turns that nonempty carrier into the displayed $\mathsf{E1}$ endpoint with a unary tail. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty spine e1 endpoint cases`

Theorem 57.371 (Metric triangle nonempty spine iff nonempty distance)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, the same three triangle distance witnesses identify nonemptiness of the final triangle distance exactly with nonemptiness of at least one point of the triangle spine:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &(d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow \\ &((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False}) \vee (z\mathsf{hsamee} \to \mathsf{False}) \vee (d_{xy}\mathsf{hsamee} \to \mathsf{False}) \vee (d_{yz}\mathsf{hsamee} \to \mathsf{False})).\end{aligned}$$

Proof. For the forward direction, apply Definition 57.369 to the three triangle witnesses and the displayed nonempty-distance hypothesis. The resulting endpoint disjunction is injected into the five-way spine disjunction by choosing the corresponding x, y, or z summand.

For the reverse direction, assume the five-way spine disjunction and suppose $d_{xyz}\mathsf{hsamee}$. By Definition 57.253, this gives $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$. Applying Definition 57.366 to the three triangle witnesses then yields $\mathsf{hsame}(x, \mathsf{e})$, $\mathsf{hsame}(y, \mathsf{e})$, $\mathsf{hsame}(z, \mathsf{e})$, $\mathsf{hsame}(d_{xy}, \mathsf{e})$, and $\mathsf{hsame}(d_{yz}, \mathsf{e})$.

Case-analyze the five-way spine disjunction. In each case, the selected nonempty spine component is a function from the corresponding empty classifier to False, so applying it to the matching classifier from the empty-spine conjunction gives the desired contradiction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle nonempty spine iff nonempty distance`

Theorem 57.372 (Metric distance witness visible-context triangle nonempty spine iff nonempty distance)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, a triangle whose first edge is displayed through visible contexts has the same exact nonempty-spine criterion as the central triangle:*

$$\begin{aligned}&\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot d_{xy}) \cdot q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \\ &(d_{xyz}\mathsf{hsamee} \to \mathsf{False}) \Longleftrightarrow \\ &((x\mathsf{hsamee} \to \mathsf{False}) \vee (y\mathsf{hsamee} \to \mathsf{False}) \vee (z\mathsf{hsamee} \to \mathsf{False}) \vee (d_{xy}\mathsf{hsamee} \to \mathsf{False}) \vee (d_{yz}\mathsf{hsamee} \to \mathsf{False})).\end{aligned}$$

Proof. Apply Definition 57.13 to the displayed first-edge witness. Its forward direction recovers the central witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$, while the unary evidence for p and q is not needed by the remaining argument.

Now apply Definition 57.371 to this recovered central witness and to the two unchanged witnesses $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$ and $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$. The result is exactly the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle nonempty spine iff nonempty distance`

Theorem 57.373 (Metric distance witness triangle append context closure)**.** *If p and q are supplied as unary visible contexts, distance witnesses from x to y, from y to z, and from d_{xy} to z produce a distance witness from $p \cdot x$ to $d_{yz} \cdot q$ whose result is $p \cdot d_{xyz} \cdot q$:*

$$\begin{aligned}&\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \\ &\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, p \cdot d_{xyz} \cdot q).\end{aligned}$$

Proof. The triangle append closure identifies the central result with $x \cdot d_{yz}$. Together with the carrier fields already present in the three distance witnesses, this gives the central witness from x to d_{yz} with result d_{xyz}. Visible-context closure then appends the unary contexts p and q on the outside. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context closed`

Theorem 57.374 (Metric distance witness visible-context triangle append closure)**.** *If the distance witness from x to y is displayed through visible contexts p, q, and the two remaining triangle witnesses are present, then the triangle append result is unchanged at the central layer:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, p{\cdot}d_{xy}{\cdot}q) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}$$

Proof. Visible-context exactness recovers the central distance witness $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$ from the displayed contextual witness. The ordinary triangle append closure applied to this recovered witness and the two remaining triangle witnesses gives $d_{xyz}\mathsf{hsame}x \cdot d_{yz}$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context triangle append closed`

Theorem 57.375 (Metric distance witness triangle append context result determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then any displayed distance witness from $p \cdot x$ to $d_{yz} \cdot q$ has the same result as the canonical contextual triangle result:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, displayed) \Longrightarrow \mathsf{hsame}(displayed, p \cdot d_x$$

Proof. Triangle append context closure constructs the canonical witness from $p \cdot x$ to $d_{yz} \cdot q$ with result $p \cdot d_{xyz} \cdot q$. The displayed witness and this canonical witness have the same source and target histories, so continuation determinacy for their distance ledgers identifies their result histories. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context result deterministic`

Theorem 57.376 (Metric distance witness triangle append context source determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then any displayed witness with the canonical contextual result and target $d_{yz} \cdot q$ determines the exposed source:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x', d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(x, x').$$

Proof. Triangle append context closure constructs the canonical contextual witness from $p \cdot x$ to $d_{yz} \cdot q$ with result $(p \cdot d_{xyz}) \cdot q$. Visible-context source determinacy then compares this canonical witness with the displayed witness, using the shared target $d_{yz} \cdot q$ and shared central distance d_{xyz}, and recovers $\mathsf{hsame}(x, x')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context source deterministic`

Theorem 57.377 (Metric distance witness triangle append context endpoint determinacy)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then the canonical contextual triangle result $(p \cdot d_{xyz}) \cdot q$ determines both exposed endpoints:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow$$
$$\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \left(\begin{array}{l} (\mathsf{MetricDistanceWitness}(p \cdot x', d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(x, x')) \\ \wedge \\ (\mathsf{MetricDistanceWitness}(p \cdot x, e \cdot q, (p \cdot d_{xyz}) \cdot q) \Longrightarrow \mathsf{hsame}(d_{yz}, e)) \end{array} \right)$$

Proof. Triangle append context closure first constructs the canonical contextual witness $\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q)$ from the three triangle witnesses and the unary context hypotheses.

For the displayed witness with source $p \cdot x'$ and target $d_{yz} \cdot q$, visible-context source determinacy compares it with the canonical contextual witness and recovers $\mathsf{hsame}(x, x')$ at the exposed source.

For the displayed witness with source $p \cdot x$ and target $e \cdot q$, visible-context target determinacy compares it with the same canonical contextual witness and recovers $\mathsf{hsame}(d_{yz}, e)$ at the exposed middle target. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint deterministic`

Theorem 57.378 (Metric distance witness triangle append context endpoint transport)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then endpoint comparisons on the exposed source and middle target transport the canonical contextual triangle witness:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(d_{yz}, e) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. Triangle append context closure first constructs the canonical contextual witness $\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q)$ from the three triangle witnesses and the unary context hypotheses.

The source comparison $\mathsf{hsame}(x, x')$ gives the contextual comparison $\mathsf{hsame}(p \cdot x, p \cdot x')$ by constructor reduction of the BEDC append context. The middle-target comparison $\mathsf{hsame}(d_{yz}, e)$ similarly gives $\mathsf{hsame}(d_{yz} \cdot q, e \cdot q)$. The result history is unchanged.

Apply the metric witness field transport theorem to these two endpoint comparisons and the reflexive result comparison. It transports the unary endpoint fields and the continuation ledger of the canonical contextual witness, which is exactly the displayed transported witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint transport`

Theorem 57.379 (Metric distance witness triangle append context endpoint iff)**.** *For unary visible contexts p, q, if the three triangle distance witnesses from x to y, from y to z, and from d_{xy} to z are given, then the canonical contextual triangle result classifies exactly the exposed source and middle target:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \Big(\mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q) \Longleftrightarrow \mathsf{hsame}(x, x') \wedge \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle append context endpoint iff`

Proof. For the forward direction, apply Definition 57.377 to the displayed witness after visible-context exactness has placed it in the canonical contextual form. The source projection gives $\mathsf{hsame}(x, x')$, while the target projection gives $\mathsf{hsame}(d_{yz}, e)$; together these are the two components of the displayed conjunction.

For the reverse direction, assume $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(d_{yz}, e)$. Apply Definition 57.373 to the two unary contexts and the three triangle witnesses. This gives the canonical witness

$$\mathsf{MetricDistanceWitness}(p \cdot x, d_{yz} \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Append congruence transports $\mathsf{hsame}(x, x')$ to $\mathsf{hsame}(p \cdot x, p \cdot x')$ and transports $\mathsf{hsame}(d_{yz}, e)$ to $\mathsf{hsame}(d_{yz} \cdot q, e \cdot q)$. Unary transport preserves the endpoint carriers across these classifiers, and Definition 57.346 transports the continuation ledger while keeping the contextual result fixed. Repacking the transported carriers with this ledger gives the displayed witness $\mathsf{MetricDistanceWitness}(p \cdot x', e \cdot q, (p \cdot d_{xyz}) \cdot q)$. □

Theorem 57.380 (Metric triangle right distance empty collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to* e*, then the shared endpoint and right endpoint collapse and the remaining displayed distances read back to the left endpoint:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xy}, x) \wedge \mathsf{hsame}(d_{xyz}, x).$$

Proof. Empty-distance exactness for the witness from y to z gives $y\mathsf{hsamee}$ and $z\mathsf{hsamee}$. Collapsing y in the first witness and using empty-right exactness identifies d_{xy} with x. Collapsing z in the outer witness identifies d_{xyz} with d_{xy}, and transitivity composes this with the previous readback to obtain $d_{xyz}\mathsf{hsame}x$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty collapse`

Theorem 57.381 (Metric triangle left distance empty visible-context closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the left distance in a metric triangle is history-same to* e*, then the triangle closes through the visible context with empty central source:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot \mathsf{e}, z \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. The left-distance empty collapse identifies the left endpoints with e and reads d_{xyz} back to z. Empty-left exactness therefore builds the central witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz})$. The visible-context iff then appends the certified unary ledgers p and q around this central witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left distance empty visible context closed`

Theorem 57.382 (Metric triangle right distance empty visible-context closure)**.** *For histories $p, q, x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to* e*, then the triangle closes through the visible context with empty central target:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(p \cdot x, \mathsf{e} \cdot q, (p \cdot d_{xyz}) \cdot q).$$

Proof. The right-distance empty collapse identifies the shared and right endpoints with e and reads d_{xyz} back to x. Empty-right exactness therefore builds the central witness $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz})$. The visible-context iff then appends the certified unary ledgers p and q around this central witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty visible context closed`

Theorem 57.383 (Metric triangle right distance empty spine witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the right distance in a metric triangle is history-same to* e*, then the outer displayed distance itself is a metric witness from x to the empty endpoint:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow \mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz}).$$

Proof. The right-distance empty collapse identifies d_{xyz} with x and the first metric witness supplies unary membership of x. Empty-right metric exactness then packages this endpoint readback as a metric witness from x to e with displayed distance d_{xyz}. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right distance empty spine witness`

Theorem 57.384 (Metric triangle right empty boundary witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$*,* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$*, and* $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ *form a metric triangle and $d_{yz}\mathsf{hsame}\mathsf{e}$, then the outer result is itself a distance witness from x to the empty boundary:*

$$\mathsf{MetricDistanceWitness}(x, \mathsf{e}, d_{xyz}).$$

Proof. By Definition 57.380, the right empty distance identifies d_{xyz} with x. The canonical right-empty witness $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, x)$ uses the source carrier of the first edge, the empty unary carrier, and the right unit continuation $x \cdot \mathsf{e} = x$. Transporting the result field along $d_{xyz}\mathsf{hsame}x$ gives the displayed witness. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle right empty boundary witness`

Theorem 57.385 (Metric triangle bilateral empty edge collapse)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if both displayed edge distances in a metric triangle are history-same to* e*, then all three endpoints, the outer displayed distance, the final structural depth, and the empty-to-empty outer witness collapse to the empty case:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow \mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow$$
$$\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceDepth}(d_{xyz}) = 0 \wedge \mathsf{MetricDistanceWitness}$$

Proof. Apply Definition 57.365 to the three triangle witnesses and the classifier $\mathsf{hsame}(d_{xy}, \mathsf{e})$. This gives the empty classifiers for x and y, and reads d_{xyz} and d_{yz} back to the right endpoint z.

Apply Definition 57.380 to the same three witnesses and the classifier $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This gives the empty classifiers for y and z, and reads d_{xy} and d_{xyz} back to the left endpoint x.

The endpoint classifiers give the first three conjuncts. For the outer distance, compose the left-collapse readback $\mathsf{hsame}(d_{xyz}, z)$ with $\mathsf{hsame}(z, \mathsf{e})$ by transitivity of hsame. Apply Definition 57.253 to this empty final-distance classifier to obtain $\mathsf{MetricDistanceDepth}(d_{xyz}) = 0$.

The empty-boundary metric witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ is the exact empty-distance case. Transporting its result field along $d_{xyz}\mathsf{hsame}\mathsf{e}$ yields $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz})$. Conjoining these rows gives the displayed package. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle bilateral empty edge collapse`

Fit: traditional target degenerate triangle normalization; BEDC source $BHist$ metric distance witnesses; carrier the three witnessed edges; classifier hsame; stability from the two empty-edge collapse lemmas and depth-zero readback; ledger the two displayed edge distances; core theorem this block; exactness the endpoint and final-distance collapse; standard bridge none claimed; status $\mathsf{ProofTarget}$.

Theorem 57.386 (Metric triangle two-sided empty boundary package)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if the two edge distances in a metric triangle are both history-same to* e*, then every displayed endpoint and the outer displayed distance collapse to the empty boundary, and the outer result is itself an empty-to-empty distance witness:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow$$
$$\mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}) \wedge \mathsf{hsame}(z, \mathsf{e}) \wedge \mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz}).$$

Proof. Apply Definition 57.365 to the same triangle and the hypothesis $d_{xy}\mathsf{hsame}\mathsf{e}$. This gives $x\mathsf{hsame}\mathsf{e}$, $y\mathsf{hsame}\mathsf{e}$, $d_{xyz}\mathsf{hsame}z$, and $d_{yz}\mathsf{hsame}z$.

Apply Definition 57.380 to the same triangle and the hypothesis $d_{yz}\mathsf{hsame}\mathsf{e}$. This gives $y\mathsf{hsame}\mathsf{e}$, $z\mathsf{hsame}\mathsf{e}$, $d_{xy}\mathsf{hsame}x$, and $d_{xyz}\mathsf{hsame}x$.

The endpoint conclusions package the three empty endpoint readbacks. Symmetry and transitivity of hsame compose $d_{xyz}\mathsf{hsame}z$ with $z\mathsf{hsame}\mathsf{e}$, so $d_{xyz}\mathsf{hsame}\mathsf{e}$.

Finally, the empty-boundary metric witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ is the exact empty-distance case. Transporting its result field along $d_{xyz}\mathsf{hsame}\mathsf{e}$ yields $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz})$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness two sided empty triangle boundary package`

Theorem 57.387 (Metric triangle left empty boundary witness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$*,* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$*, and* $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ *form a metric triangle and* $d_{xy}\mathsf{hsame}\mathsf{e}$*, then the outer result is itself a distance witness from the empty boundary to z:*

$$\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz}).$$

Proof. By Definition 57.365, the left empty distance identifies d_{xyz} with z.

The second edge supplies the unary carrier for z. Apply Definition 57.40 with this carrier and the displayed classifier $d_{xyz}\mathsf{hsame}z$ to package the canonical witness from e to z with displayed distance d_{xyz}. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness triangle left empty boundary witness`

Theorem 57.388 (Metric triangle two-sided empty witness exactness)**.** *For histories $x, y, z, d_{xy}, d_{yz}, d_{xyz}$, if* $\mathsf{MetricDistanceWitness}(x, y, d_{xy})$*,* $\mathsf{MetricDistanceWitness}(y, z, d_{yz})$*, and* $\mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz})$ *form a metric triangle and both displayed edge distances are history-same to* e*, then the outer result simultaneously carries every empty-boundary witness row needed by the triangle:*

$$\mathsf{MetricDistanceWitness}(x, y, d_{xy}) \Longrightarrow \mathsf{MetricDistanceWitness}(y, z, d_{yz}) \Longrightarrow \mathsf{MetricDistanceWitness}(d_{xy}, z, d_{xyz}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xy}, \mathsf{e}) \Longrightarrow \mathsf{hsame}(d_{yz}, \mathsf{e}) \Longrightarrow$$
$$\mathsf{hsame}(d_{xyz}, \mathsf{e}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d_{xyz}) \wedge \mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz}) \wedge \mathsf{MetricDistanceWitness}(x, \mathsf{e},$$

Proof. First apply Definition 57.385 to the three triangle witnesses and the two classifiers $\mathsf{hsame}(d_{xy}, \mathsf{e})$ and $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This gives $\mathsf{hsame}(d_{xyz}, \mathsf{e})$ together with the empty-to-empty witness $\mathsf{MetricDistanceWitness}(\mathsf{e}, \mathsf{e}, d$

Next apply Definition 57.387 to the same triangle and $\mathsf{hsame}(d_{xy}, \mathsf{e})$. This supplies the boundary row $\mathsf{MetricDistanceWitness}(\mathsf{e}, z, d_{xyz})$.

Finally apply Definition 57.384 to the same triangle and $\mathsf{hsame}(d_{yz}, \mathsf{e})$. This supplies $\mathsf{MetricDistanceWitness}(x, \mathsf{e},$ and the four rows assemble into the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness two sided empty triangle witness exactness`

Theorem 57.389 (Metric distance witness visible-context empty distance exactness)**.** *For histories p, q, x, y, a visible-context distance witness whose central distance ledger is empty is exactly unary evidence for the visible prefix and suffix together with empty classification of both central endpoints:*

$$\mathsf{MetricDistanceWitness}(p{\cdot}x, y{\cdot}q, (p{\cdot}\mathsf{e}){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{hsame}(x, \mathsf{e}) \wedge \mathsf{hsame}(y, \mathsf{e}).$$

Proof. Apply visible-context exactness to peel off the prefix and suffix. The central witness is then $\mathsf{MetricDistanceWitness}(x, y, \mathsf{e})$, which empty-distance exactness identifies with $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. Conversely, the two empty endpoint classifiers build the central empty-distance witness, and visible-context exactness reattaches the same prefix and suffix. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance iff`

Theorem 57.390 (Metric visible-context empty-distance continuation splice)**.** *For histories p, q, x, y, l, r, m, o, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$ and $\mathsf{Cont}(m, r, o)$ hold. Then the empty central endpoint exposed by the witness splices the continuation directly:

$$\mathsf{Cont}(l, r, o).$$

Proof. Apply visible-context empty-distance exactness to the witness. This gives $\mathsf{hsame}(x, \mathsf{e})$. Substitute the empty central endpoint into $\mathsf{Cont}(l, x, m)$, so the first continuation identifies m with l. The second continuation $\mathsf{Cont}(m, r, o)$ is therefore the desired direct continuation $\mathsf{Cont}(l, r, o)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation splice`

Theorem 57.391 (Metric visible-context empty-distance continuation splice determinacy)**.** *For histories $p, q, x, y, l, r, m, o, o'$, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$, $\mathsf{Cont}(m, r, o)$, and $\mathsf{Cont}(l, r, o')$ hold. Then the spliced result is determined by the displayed direct continuation:

$$\mathsf{hsame}(o, o').$$

Proof. The visible-context empty-distance continuation splice turns the first two continuations into $\mathsf{Cont}(l, r, o)$. BEDC continuation determinacy compares this spliced continuation with the displayed continuation $\mathsf{Cont}(l, r, o')$, giving $\mathsf{hsame}(o, o')$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation splice hsame`

Theorem 57.392 (Metric visible-context empty-distance continuation result determinacy)**.** *For histories p, q, x, y, l, m, n, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, m)$ and $\mathsf{Cont}(l, y, n)$ hold. Then the two continuation results are classified together:

$$\mathsf{hsame}(m, n).$$

Proof. Apply visible-context empty-distance exactness to the metric witness. This identifies both central endpoints x and y with e. After substituting these endpoint classifiers, the two displayed continuations have the same left source and the same empty right endpoint. BEDC continuation determinacy therefore compares their results and gives $\mathsf{hsame}(m, n)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation results hsame`

Theorem 57.393 (Metric visible-context empty-distance append unit laws)**.** *For histories p, q, x, y, l, r, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q).$$

Then the central endpoints act as append units:

$$x \cdot r = r \quad \wedge \quad l \cdot y = l.$$

Proof. Apply visible-context empty-distance exactness to the witness. This exposes $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. Substituting these empty endpoints turns the two displayed append equations into the BEDC left and right append unit laws. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance append unit laws`

Theorem 57.394 (Metric visible-context empty-distance endpoint continuation units)**.** *For histories $p, q, x, y, l, r, o_L, o_R$, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and continuations $\mathsf{Cont}(l, x, o_L)$ and $\mathsf{Cont}(y, r, o_R)$ hold. Then both displayed endpoint continuations reduce to unit continuations:

$$\mathsf{hsame}(o_L, l) \wedge \mathsf{hsame}(o_R, r).$$

Proof. Visible-context empty-distance exactness exposes $\mathsf{hsame}(x, \mathsf{e})$ and $\mathsf{hsame}(y, \mathsf{e})$. After substituting these central endpoints, $\mathsf{Cont}(l, x, o_L)$ is a right-unit continuation and $\mathsf{Cont}(y, r, o_R)$ is a left-unit continuation. The BEDC continuation unit laws therefore give $\mathsf{hsame}(o_L, l)$ and $\mathsf{hsame}(o_R, r)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance endpoint continuation units`

Theorem 57.395 (Metric visible-context empty-distance witness splice)**.** *For histories p, q, x, y, l, r, m, o, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

and unary endpoint evidence for l and r hold. If $\mathsf{Cont}(l, x, m)$ and $\mathsf{Cont}(m, r, o)$ hold, then the spliced continuation is a metric distance witness:

$$\mathsf{MetricDistanceWitness}(l, r, o).$$

Proof. The visible-context empty-distance continuation splice supplies $\mathsf{Cont}(l, r, o)$. Together with the unary endpoint hypotheses, unary continuation closure gives the unary carrier for o. These three fields are exactly the concrete metric distance witness from l to r with output o. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance witness splice`

Theorem 57.396 (Metric visible-context empty-distance continuation witness depth package)**.** *For histories* $p, q, x, y, l, r, m, o : \mathsf{BHist}$*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

holds together with $\mathsf{UnaryHistory}(l)$*,* $\mathsf{UnaryHistory}(r)$*,* $\mathsf{Cont}(l, x, m)$*, and* $\mathsf{Cont}(m, r, o)$*. Then the direct continuation carries a metric witness whose depth is induced by the two visible endpoints:*

$$\mathsf{MetricDistanceWitness}(l, r, o) \wedge \mathsf{MetricDistanceDepth}(o) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r).$$

Proof. Apply the visible-context empty-distance continuation splice to the distance witness and the two continuation ledgers. This gives the direct continuation $\mathsf{Cont}(l, r, o)$.

The unary carriers for l and r then close the continuation under the metric distance witness constructor, so the direct continuation packages as $\mathsf{MetricDistanceWitness}(l, r, o)$.

Finally, metric witness depth additivity applied to this witness gives $\mathsf{MetricDistanceDepth}(o) = \mathsf{MetricDistanceDepth}(l) + \mathsf{MetricDistanceDepth}(r)$. Pairing the witness with that displayed equation is the required conjunction. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance continuation witness depth`

Theorem 57.397 (Metric visible-context empty-distance two-sided endpoint witness transport)**.** *For histories* $p, q, x, y, l, r, m, n, o : \mathsf{BHist}$*, suppose*

$$\mathsf{MetricDistanceWitness}(p \cdot x, y \cdot q, (p \cdot \mathsf{e}) \cdot q)$$

holds together with $\mathsf{UnaryHistory}(l)$*,* $\mathsf{UnaryHistory}(r)$*,* $\mathsf{Cont}(l, x, m)$*,* $\mathsf{Cont}(y, r, n)$*, and* $\mathsf{Cont}(l, r, o)$*. Then the two reduced endpoint continuations carry the same metric distance witness:*

$$\mathsf{MetricDistanceWitness}(m, n, o).$$

Proof. Apply Definition 57.394 to the visible-context empty-distance witness and the two endpoint continuation ledgers. This gives $\mathsf{hsame}(m, l)$ and $\mathsf{hsame}(n, r)$.

The unary endpoint hypotheses for l and r, together with the direct continuation ledger $\mathsf{Cont}(l, r, o)$, build the concrete direct metric witness $\mathsf{MetricDistanceWitness}(l, r, o)$.

Now use Definition 57.347. Transport the first endpoint along the symmetry of $\mathsf{hsame}(m, l)$, the second endpoint along the symmetry of $\mathsf{hsame}(n, r)$, and the result endpoint along reflexivity of $\mathsf{hsame}(o, o)$. The continuation ledger transports with the same three relations, so the transported fields package as $\mathsf{MetricDistanceWitness}(m, n, o)$. □

Lean4 ✓ `BEDC.Derived.MetricUp.MetricDistanceWitness visible context empty distance two sided endpoint witness transport`

The ten-field fit is: traditional target visible-context zero-distance splice; BEDC source MetricDistanceWitness plus Cont; carrier MetricDistanceWitness; classifier hsame; stability obligations the two unary carriers and the two continuation ledgers; ledger the hidden empty central distance; core theorem this package; exactness the displayed witness-plus-depth conjunction; standard bridge deferred; and status ProofTarget.

Chapter 58

A Concrete Naming Certificate for $\mathsf{S}^{1\uparrow}$

The circle interface $\mathsf{S}^{1\uparrow}$ is a certificate-obligation schema over the product carrier $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ from chapter 38. Its carrier is the unit-circle subspace: pairs (x, y) equipped with a certified equation witness that $x^2 + y^2$ is classified as the real unit. The interface is not a primitive topological space; it is admitted only when the product real witnesses, equation witness, componentwise classifier, and equation transport data are supplied. The metric-observation and standard topological bridge boundaries consume the metric carrier discipline of chapter 57 and the compact-source discipline of chapter 129 only as downstream finite rows, not as angle, covering, compactness, or connectedness claims.

58.1 Carrier obligation surface

Definition 58.1 (SOne carrier)**.** *Use pairs (x, y) of Bishop real witnesses with a certified unit equation witness $x^2 + y^2 \approx 1$ under* RealClassifier*.*

Lean4 (definition) `BEDC.Derived.S1Up.SOneUnitHistory`
Lean4 (definition) `BEDC.Derived.S1Up.SOneProductHistoryCarrier`
Lean4 (definition) `BEDC.Derived.S1Up.SOneHistoryCarrier`

Theorem 58.2 (S1Up root obligation surface)**.** *An accepted $\mathsf{S}^{1\uparrow}$ carrier over* BHist *consists of two $\mathsf{Real}^{\uparrow}$ coordinate rows, the product-source row inherited from $\mathsf{Prod}^{\uparrow}$, the $\mathsf{Real}^{\uparrow}$ classifier transports for both coordinates, a displayed radius row asserting that $x^2 + y^2$ is classified with the $\mathsf{Real}^{\uparrow}$ unit, and a* NameCert *ledger recording the $\mathsf{Real}^{\uparrow}$ and product witnesses.*

The radius row is a classifier obligation over the displayed real-coordinate data. It is not a host equality and it does not add a primitive circle carrier outside the real/product source rows.

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier root obligation surface`

Proof. Unpack the two $\mathsf{Real}^{\uparrow}$ coordinate certificates. They supply the two BHist component rows and the componentwise classifier transports used by the circle carrier.

Use the $\mathsf{Prod}^{\uparrow}$ product carrier to pair those rows into one source packet. The resulting packet records the real-coordinate witnesses and the product source row in the same BHist ledger.

Finally record the radius condition $x^2 + y^2$ against the $\mathsf{Real}^{\uparrow}$ unit as a displayed classifier obligation. Since the condition is stored as a row in the NameCert ledger, the carrier exposes only the real, product, radius, and ledger witnesses described above. □

Theorem 58.3 (SOne history carrier rational components)**.** *For every history h, the concrete $\mathsf{S}^{1\uparrow}$ product-history carrier exposes product components x, y and rational denominator tails d_x, d_y such that both components are displayed as real constants and the BEDC continuation ledger composes the two components into h:*

$$\mathsf{SOneProductHistoryCarrier}(h) \Longrightarrow \exists x, y, d_x, d_y.\ \mathsf{hsame}(x, \mathsf{E1}(d_x)) \wedge \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{hsame}(y, \mathsf{E1}(d_y)) \wedge \mathsf{RatHisto}$$

Proof. Unpack the $\mathsf{S}^{1\uparrow}$ carrier as the $\mathsf{Prod}^{\uparrow}$ carrier over two real-constant history carriers. The product carrier supplies the two component histories and their continuation ledger into h. Each real-constant carrier then exposes a rational denominator tail together with the displayed $\mathsf{E1}$ presentation of that component. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier rational components`

Theorem 58.4 (SOne history carrier real-pair carrier)**.** *For histories x, y, e, p, the concrete $\mathsf{S}^{1\uparrow}$ history carrier determines the underlying product-real pair carrier:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{ProdHistoryCarrier}(\mathsf{RealConstantHistoryCarrier}, \mathsf{RealConstantHistoryCarrier}, p).$$

Proof. Unpack the circle carrier. Its first two fields are real-constant carrier witnesses for x and y, and its continuation field records $\mathsf{Cont}(x, y, p)$. These three data are precisely the product-history carrier introduction rule for the two real-constant component predicates. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier real pair`

Theorem 58.5 (SOne point ledger determinacy)**.** *For histories x, y, e, e', p, p', two concrete $\mathsf{S}^{1\uparrow}$ history-carrier witnesses with the same component histories determine the point ledger up to the BEDC history classifier:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x, y, e', p') \Longrightarrow \mathsf{hsame}(p, p').$$

Proof. Unpack the two circle-carrier witnesses only as far as their point-continuation ledgers. Both ledgers have the same source component x and the same right component y. Continuation determinacy therefore identifies their continuation results, giving $\mathsf{hsame}(p, p')$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier point deterministic`

Theorem 58.6 (SOne history carrier e-zero point exclusion)**.** *For histories x, y, e, t, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness cannot have an $\mathsf{E0}$-headed point ledger:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, \mathsf{E0}(t)) \Longrightarrow \bot.$$

Proof. Unpack the circle carrier to the second real-constant component and the continuation ledger. The second component displays y as $\mathsf{E1}(d)$ for a rational history d. Transporting the continuation ledger along that display makes the point ledger an $\mathsf{E1}$-headed append result, contradicting the displayed $\mathsf{E0}$ head. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier point e0 absurd`

Theorem 58.7 (SOne history carrier empty point exclusion)**.** *For histories x, y, e, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness cannot have the empty point ledger:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, \mathsf{e}) \Longrightarrow \bot.$$

Proof. Unpack the circle carrier to the second real-constant component and the continuation ledger. The second component displays y as $\mathsf{E1}(d)$ for a rational history d. Transporting the continuation along that display makes the point ledger an $\mathsf{E1}$-headed append result, contradicting the empty endpoint. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier point empty absurd`

Theorem 58.8 (SOne history carrier empty point exclusion)**.** *For histories x, y, e, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness cannot have an empty point ledger:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, \mathsf{e}) \Longrightarrow \bot.$$

Proof. Unpack the circle carrier to the first real-constant component and the continuation ledger. The empty-result continuation inversion forces the first component to be empty, while the real-constant carrier displays it as an $\mathsf{E1}$-headed rational history. Constructor no-confusion gives the contradiction. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier empty point absurd`

Theorem 58.9 (SOne history carrier e-one point shape)**.** *For histories x, y, e, p, every concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness exposes the point ledger as an $\mathsf{E1}$-headed history:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \exists t : \mathsf{BHist}.\ p = \mathsf{E1}(t).$$

Proof. Unpack the carrier to the second real-constant component and the continuation ledger. The second component displays y as $\mathsf{E1}(d)$ for a rational history d. Substituting that display into the continuation equation makes the point ledger p the append result with an $\mathsf{E1}$-headed right component, hence $p = \mathsf{E1}(\mathsf{append}(x, d))$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier point e1 shape`

Theorem 58.10 (SOne history carrier rational unit components)**.** *For histories x, y, e, p, the concrete $\mathsf{S}^{1\uparrow}$ history carrier exposes rational denominator tails for both real components while preserving the unit equation classifier and the continuation ledger:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \exists d_x, d_y.\ \mathsf{hsame}(x, \mathsf{E1}(d_x)) \wedge \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{hsame}(y, \mathsf{E1}(d_y)) \wedge \mathsf{RatHistoryCar}$$

Proof. Unpack the $\mathsf{S}^{1\uparrow}$ history carrier. Its first two fields are real-constant carrier witnesses, so each component supplies a rational denominator tail and the corresponding $\mathsf{E1}$ presentation. The remaining fields are exactly the unit-equation classifier and the BEDC continuation ledger. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier rational unit components`

Theorem 58.11 (SOne history carrier point-tail readback)**.** *For histories x, y, e, t, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness whose point ledger is visibly one-tagged exposes the rational component tails and reads the visible point tail as the continuation of the first coordinate by the second coordinate tail:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, \mathsf{E1}(t)) \Longrightarrow \exists d_x, d_y : \mathsf{BHist}.\ \mathsf{hsame}(x, \mathsf{E1}(d_x)) \wedge \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{hsame}(y, \mathsf{E1}(d_y)) \wedge \mathsf{Ra}$$

Proof. Apply Definition 58.10. This exposes rational tails d_x, d_y, presentations $\mathsf{hsame}(x, \mathsf{E1}(d_x))$ and $\mathsf{hsame}(y, \mathsf{E1}(d_y))$, and the continuation ledger $\mathsf{Cont}(x, y, \mathsf{E1}(t))$.

Using the presentation $\mathsf{hsame}(y, \mathsf{E1}(d_y))$, the continuation result has the form $\mathsf{E1}(t) = \mathsf{E1}(x \cdot d_y)$. Constructor inversion for $\mathsf{E1}$ gives $\mathsf{hsame}(t, x \cdot d_y)$. Package this with the two exposed rational carrier witnesses. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier point tail readback`

Theorem 58.12 (SOne y endpoint reads the point tail)**.** *For histories x, y, e, t, d_y, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness with point ledger $\mathsf{E1}(t)$ and second coordinate presented as $\mathsf{E1}(d_y)$ reads back the point tail as the continuation of x by d_y:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, \mathsf{E1}(t)) \to \mathsf{hsame}(y, \mathsf{E1}(d_y)) \to \mathsf{hsame}(t, x \cdot d_y).$$

Proof. Unpack only the point-continuation field of the circle carrier. Transport the displayed second endpoint along $\mathsf{hsame}(y, \mathsf{E1}(d_y))$. The continuation result then has the visible form $\mathsf{E1}(t) = \mathsf{E1}(x \cdot d_y)$, and one-constructor inversion gives the displayed tail classifier. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier y e1 point tail hsame`

Theorem 58.13 (SOne endpoints read the e-one point tail)**.** *For histories x, y, e, t, d_x, d_y, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness with point ledger $\mathsf{E1}(t)$ and coordinate presentations $\mathsf{hsame}(x, \mathsf{E1}(d_x))$ and $\mathsf{hsame}(y, \mathsf{E1}(d_y))$ reads the point tail as*

$$\mathsf{hsame}(t, \mathsf{E1}(d_x) \cdot d_y).$$

Proof. The second coordinate presentation gives the point-tail readback $t \sim x \cdot d_y$. Transporting the first coordinate presentation $x \sim \mathsf{E1}(d_x)$ through the same continuation tail identifies this with $\mathsf{E1}(d_x) \cdot d_y$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier x y e1 point tail hsame`

Theorem 58.14 (SOne e-one components read point tail)**.** *For histories d_x, d_y, e, t, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness whose coordinates and point ledger are all one-headed exposes rational coordinate tails and reads the point tail from the BEDC continuation:*

$$\mathsf{SOneHistoryCarrier}(\mathsf{E1}(d_x), \mathsf{E1}(d_y), e, \mathsf{E1}(t)) \Longrightarrow \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{RatHistoryCarrier}(d_y) \wedge \mathsf{hsame}(t, \mathsf{E1}(d_x) \cdot d_y).$$

Proof. The first two carrier fields are real-constant carriers on one-headed histories, hence the real-constant e-one exactness theorem gives rational carriers for d_x and d_y. The displayed second coordinate presentation is reflexive, so the point-tail readback theorem gives $\mathsf{hsame}(t, \mathsf{E1}(d_x) \cdot d_y)$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier e1 components tail`

Theorem 58.15 (SOne e-one components unit tail readback)**.** *For histories d_x, d_y, e, t, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness whose coordinates and point ledger are all one-headed exposes the two rational coordinate tails, identifies the equation endpoint with the circle unit history, and reads the point tail as the displayed continuation:*

$$\mathsf{SOneHistoryCarrier}(\mathsf{E1}(d_x), \mathsf{E1}(d_y), e, \mathsf{E1}(t)) \Longrightarrow \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{RatHistoryCarrier}(d_y) \wedge \mathsf{hsame}(e, \mathsf{SOneUnitHis}$$

Proof. The e-one component tail theorem supplies the two rational coordinate carriers and the equality $t\mathsf{hsameE1}(d_x) \cdot d_y$. The equation-unit readback identifies e with $\mathsf{SOneUnitHistory}$. Reading the displayed tail equality as a BEDC continuation gives $\mathsf{Cont}(\mathsf{E1}(d_x), d_y, t)$, and these four fields form the claimed package. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier e1 components unit tail readback`

Theorem 58.16 (SOne e-one components unit equation classifier)**.** *For histories d_x, d_y, e, t, a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness whose coordinates, equation endpoint, and point ledger are one-headed exposes the two rational coordinate tails, the rational classifier from the equation tail to the unit tail, and the displayed point continuation:*

$$\mathsf{SOneHistoryCarrier}(\mathsf{E1}(d_x), \mathsf{E1}(d_y), \mathsf{E1}(e), \mathsf{E1}(t)) \Longrightarrow \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{RatHistoryCarrier}(d_y) \wedge \mathsf{RatHistoryClassifi}$$

Proof. The unit-tail readback supplies the two rational coordinate carriers and the displayed point continuation. The stored real-constant classifier on the one-headed equation endpoint has target $\mathsf{SOneUnitHistory} = \mathsf{E1}(\mathsf{E1}(\mathsf{e}))$, so the real-constant e-one exactness theorem reads it as the rational classifier between the equation tail and $\mathsf{E1}(\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier e1 components unit equation classifier`

Theorem 58.17 (SOne e-one components carrier exactness)**.** *For histories d_x, d_y, e, t, the one-headed presentation of a concrete $\mathsf{S}^{1\uparrow}$ history carrier is exact: it is present precisely when the two coordinate tails are rational carriers, the equation tail classifies the one-headed empty tail, and the point tail is the displayed continuation:*

$$\mathsf{SOneHistoryCarrier}(\mathsf{E1}(d_x), \mathsf{E1}(d_y), \mathsf{E1}(e), \mathsf{E1}(t)) \Longleftrightarrow \mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{RatHistoryCarrier}(d_y) \wedge \mathsf{RatHistoryClassif}$$

Proof. The forward direction is exactly the e-one unit-equation classifier readback: apply Definition 58.16 to the given $\mathsf{S}^{1\uparrow}$ carrier witness.

For the reverse direction, start from the four displayed fields. The two rational carrier fields repack as the real-constant carrier witnesses for $\mathsf{E1}(d_x)$ and $\mathsf{E1}(d_y)$. The rational classifier $\mathsf{RatHistoryClassifier}(e, \mathsf{E1}($ lifts along the real-constant e-one classifier exactness theorem to the required classifier from $\mathsf{E1}(e)$ to $\mathsf{SOneUnitHistory}$.

It remains to supply the point-continuation field. The hypothesis $\mathsf{Cont}(\mathsf{E1}(d_x), d_y, t)$ is exactly the statement that the one-headed point ledger $\mathsf{E1}(t)$ carries the append continuation determined by $\mathsf{E1}(d_x)$ and d_y. With these four fields repacked, the constructor for $\mathsf{SOneHistoryCarrier}$ gives the desired carrier witness. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier e1 components carrier exactness`

Theorem 58.18 (SOne history carrier y e-one point shape)**.** *For histories x, y, e, p, d_y, if a concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness presents the second coordinate as $\mathsf{E1}(d_y)$, then the point ledger is visibly one-headed and its exposed tail is the continuation of x by d_y:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(y, \mathsf{E1}(d_y)) \Longrightarrow \exists t : \mathsf{BHist}.\ \mathsf{hsame}(p, \mathsf{E1}(t)) \wedge \mathsf{Cont}(x, d_y, t).$$

Proof. Unpack the circle carrier only to its point-continuation field. Replacing the second coordinate by its $\mathsf{E1}(d_y)$ presentation makes the point ledger definitionally $\mathsf{E1}(x \cdot d_y)$. Choose the exposed payload $t = x \cdot d_y$; the remaining continuation statement is the canonical append ledger. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier y e1 point shape`

Theorem 58.19 (SOne unit equation determinacy)**.** *For histories x, y, e, p, every concrete $\mathsf{S}^{1\uparrow}$ history-carrier witness determines that its stored equation endpoint is the unit history:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(e, \mathsf{SOneUnitHistory}).$$

Proof. Unpack the SOneHistoryCarrier witness and keep the RealConstantHistoryClassifier$(e, \mathsf{SOneUnitHistory})$ field. This classifier exposes rational tails for both endpoints. Its rational payload classifier identifies the tails, and constructor congruence lifts that equality through E1. Composing this equality with the stored endpoint presentations gives hsame$(e, \mathsf{SOneUnitHistory})$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier unit equation deterministic`

Theorem 58.20 (SOne history carrier equation-pair determinacy)**.** *For histories $x, y, e, p, x', y', e', p'$, any two concrete $\mathsf{S}^{1\uparrow}$ history-carrier witnesses determine history-same equation ledgers:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x', y', e', p') \Longrightarrow \mathsf{hsame}(e, e').$$

Proof. Apply the unit-equation determinacy theorem to each carrier witness. The first witness identifies e with SOneUnitHistory, and the second identifies e' with the same endpoint. Transitivity through that endpoint gives hsame(e, e'). □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier equation pair deterministic`

Theorem 58.21 (SOne equation unit readback)**.** *For histories x, y, e, p, the concrete $\mathsf{S}^{1\uparrow}$ history carrier determines that its equation witness is the BEDC unit history:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(e, \mathsf{SOneUnitHistory}).$$

Proof. Unpack the circle carrier to its equation-classifier field RealConstantHistoryClassifier$(e, \mathsf{SOneUnitHistory})$. That field displays e and SOneUnitHistory as E1-sealed rational histories, with a rational classifier between the displayed tails. The rational classifier contains an hsame witness between those tails; one-constructor congruence lifts it to the sealed endpoints, and transitivity with the two endpoint displays identifies e with SOneUnitHistory. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier equation unit`

Theorem 58.22 (SOne history carrier public readback)**.** *For histories x, y, e, p, the concrete $\mathsf{S}^{1\uparrow}$ history carrier exposes the product carrier, identifies the equation endpoint with the BEDC unit history, and reads back rational tails for both real components:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneProductHistoryCarrier}(p) \wedge \mathsf{hsame}(e, \mathsf{SOneUnitHistory}) \wedge \exists d_x, d_y.\ \mathsf{hsame}(x, \mathsf{E1}(d$$

Proof. The product carrier is obtained by repacking the two real-constant component carriers with the continuation ledger. Equation-unit determinacy identifies the stored equation endpoint with SOneUnitHistory. Finally, the rational unit component readback exposes the two rational tails and the same continuation ledger, while its equation classifier field is no longer needed. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier public readback`

Theorem 58.23 (SOne component classifier ledger determinacy)**.** *For histories $x, y, e, p, x', y', e', p'$, componentwise BEDC history classification of two concrete $\mathsf{S}^{1\uparrow}$ carrier witnesses determines both hidden ledgers:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x', y', e', p') \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(e, e')$$

Proof. Unpack the two circle-carrier witnesses into their four fields. The first gives the real-constant component carriers for x and y, the equation classifier $\mathsf{RealConstantHistoryClassifier}(e, \mathsf{SOneUnitHistory})$, and the continuation ledger $\mathsf{Cont}(x, y, p)$. The second gives the corresponding fields for x', y', e', and p'.

For the equation ledger, unpack the two $\mathsf{RealConstantHistoryClassifier}(-, \mathsf{SOneUnitHistory})$ witnesses. Each contains a $\mathsf{RatHistoryClassifier}$ tail against the fixed $\mathsf{SOneUnitHistory}$ endpoint. Constructor inversion for the one-constructor rational history spine exposes the same rational tail on both sides. The hsame symmetry and transitivity rules then connect e to e', and rebuilding the history classifier witness yields $\mathsf{hsame}(e, e')$.

For the point ledger, apply continuation hsame-respect to the two ContR ledgers using the supplied component witnesses $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$. Continuation determinacy identifies the resulting point histories, giving $\mathsf{hsame}(p, p')$.

Pair the two conclusions. The certificate fit is the traditional target unit-circle classifier determinacy; the BEDC source BHist with $\mathsf{RealConstantHistoryClassifier}$ and ContR; the carrier $\mathsf{SOneHistoryCarrier}$; the classifier hsame on exposed components; the ledger equation proof and point continuation; the core theorem this determinacy statement; the exactness claim hidden-ledger uniqueness; the standard bridge deferred outside this theorem; and the status $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier component classifier ledger determinacy`

Theorem 58.24 (SOne scoped carrier package)**.** *An accepted* $\mathsf{S}^{1\uparrow}$ *carrier exports exactly the scoped carrier rows consumed by* $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readers: the two* $\mathsf{Real}^{\uparrow}$ *component rows, the inherited product ledger, the unit-equation witness, the public point readback, component-classifier determinacy, coordinate transport, and the finite-window consumer boundary. The same carrier exports no compactness row, connectedness row, angle row, winding row, quotient-circle row, host equality row, or metric-completeness row.*

Proof. The public carrier rows are supplied by Definition 58.22: the two real-coordinate component rows, the product ledger, and the public point ledger are all read from the displayed BHist, hsame, and Cont data.

The ledger exactness rows are the checked local rows. Apply Definition 58.23 for the hidden point and equation-ledger determinacy, and apply Definition 58.21 for the fixed unit-equation witness. Apply Definition 58.25 for coordinate transport.

Finally, Definition 58.27 gives the finite-window consumer boundary: any shared-point read factors through the same component rows and point ledger. The listed negative rows are absent because none of the displayed constructor, classifier, continuation, package, or naming rows carries compactness, connectedness, angle, winding, quotient-circle, host equality, or metric-completeness data. □

Theorem 58.25 (SOne history carrier coordinate-pair determinacy)**.** *For histories* x, x', y, y', e, e', p, *two concrete* $\mathsf{S}^{1\uparrow}$ *carrier witnesses with the same point ledger and history-same second coordinate determine the first coordinate and the equation witness:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x', y', e', p) \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(e, e').$$

Proof. Unpack the two circle-carrier witnesses. The shared point ledger gives $\mathsf{Cont}(x, y, p)$ and $\mathsf{Cont}(x', y', p)$; after transporting along $\mathsf{hsame}(y, y')$, continuation right-cancellation identifies x with x'. The equation components are both classified against the fixed $\mathsf{SOneUnitHistory}$ endpoint, so the unit-equation determinacy readback identifies each equation witness with that endpoint, and transitivity gives $\mathsf{hsame}(e, e')$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier coordinate pair deterministic`

Theorem 58.26 (SOne history carrier coordinate-pair right determinacy)**.** *For histories x, x', y, y', e, e', p, two concrete $\mathsf{S}^{1\uparrow}$ carrier witnesses with the same point ledger and history-same first coordinate determine the second coordinate and the equation witness:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x', y', e', p) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \wedge \mathsf{hsame}(e, e').$$

Proof. Unpack the two circle-carrier witnesses. The shared point ledger gives $\mathsf{Cont}(x, y, p)$ and $\mathsf{Cont}(x', y', p)$; after transporting along $\mathsf{hsame}(x, x')$, continuation left-cancellation identifies y with y'. The equation components are both classified against the fixed $\mathsf{SOneUnitHistory}$ endpoint, so unit-equation determinacy identifies each equation witness with that endpoint, and transitivity gives $\mathsf{hsame}(e, e')$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier coordinate pair right deterministic`

Theorem 58.27 (SOne shared-point component equivalence)**.** *For histories x, x', y, y', e, e', p, two concrete $\mathsf{S}^{1\uparrow}$ carrier witnesses over the same displayed point ledger have the following shared-point component equivalence:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{SOneHistoryCarrier}(x', y', e', p) \Longrightarrow (\mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \wedge \mathsf{hsame}(e, e')) \wedge ($$

Proof. Assume the two displayed carrier witnesses $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ and $\mathsf{SOneHistoryCarrier}(x', y', e', p)$.

For the first branch, suppose $\mathsf{hsame}(x, x')$. Apply Definition 58.26 to the two carrier witnesses and the first-coordinate equality. The hypotheses use the same displayed point ledger p, so right determinacy recovers both $\mathsf{hsame}(y, y')$ and $\mathsf{hsame}(e, e')$; the equation component is recovered against the fixed $\mathsf{SOneUnitHistory}$ ledger exposed by the carrier.

For the second branch, suppose $\mathsf{hsame}(y, y')$. Apply Definition 58.25 to the same two carrier witnesses and the second-coordinate equality. The shared displayed point ledger is again p, so coordinate-pair determinacy recovers $\mathsf{hsame}(x, x')$ and the same equation-ledger comparison $\mathsf{hsame}(e, e')$ against $\mathsf{SOneUnitHistory}$.

Pair the two implications. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier shared point component iff`

Definition 58.28 (SOne displayed completion readback)**.** *Let α be a displayed BEDC source map on* BHist*. A row*

$$\mathsf{SOneDisplayedCompletionReadback}_\alpha(a, x, y, e, p, c)$$

means that the source row a is read through a concrete $\mathsf{S}^{1\uparrow}$ carrier and that the $\mathsf{Completion}^\uparrow$ endpoint is represented by the same visible point ledger:

$$\mathsf{SOneDisplayedCompletionReadback}_\alpha(a, x, y, e, p, c) :\Longleftrightarrow \mathsf{SOneHistoryCarrier}(x, y, e, p) \wedge \mathsf{hsame}(\alpha(a), p) \wedge \mathsf{hsame}(c, p).$$

The predicate has no field for a host quotient representative: the only completion readback data are the displayed point ledger p and the two hsame *comparisons tying p to the source image and to the public endpoint.*

Lean4 (definition) `BEDC.Derived.S1Up.SOneDisplayedCompletionReadback`

Proposition 58.29 (SOne completion readback preserves source separation)**.** *Let α be a displayed BEDC source map on* BHist *whose additive and cancellation rows supply the injectivity judgment*

$$\forall u, v : \mathsf{BHist}.\ \mathsf{hsame}(\alpha(u), \alpha(v)) \Longrightarrow \mathsf{hsame}(u, v).$$

Then displayed $\mathsf{S}^{1\uparrow}$-$\mathsf{Completion}^{\uparrow}$ *readback is injective on source rows and therefore preserves source separation:*

$$\mathsf{SOneDisplayedCompletionReadback}_{\alpha}(a, x, y, e, p, c) \wedge \mathsf{SOneDisplayedCompletionReadback}_{\alpha}(a', x', y', e', p', c')$$
$$\Longrightarrow (\mathsf{hsame}(c, c') \Longrightarrow \mathsf{hsame}(a, a')) \wedge ((\mathsf{hsame}(a, a') \Longrightarrow \bot) \Longrightarrow (\mathsf{hsame}(c, c') \Longrightarrow \bot)).$$

Lean4 ✓ `BEDC.Derived.S1Up.SOneDisplayedCompletionReadback source separation`

Proof. Unpack the two displayed completion-readback rows of Definition 58.28. The first row gives $\mathsf{SOneHistoryCarrier}(x, y, e, p)$, $\mathsf{hsame}(\alpha(a), p)$, and $\mathsf{hsame}(c, p)$. The second gives $\mathsf{SOneHistoryCarrier}(x', y', e', p')$, $\mathsf{hsame}(\alpha(a'), p')$, and $\mathsf{hsame}(c', p')$.

By Definition 58.1 and Definition 58.22, the ledgers p and p' are the public point histories exposed by concrete $\mathsf{S}^{1\uparrow}$ carriers: each is read from the displayed real-coordinate carrier, the unit-equation ledger, and the BEDC continuation row. Thus the endpoint comparisons above are comparisons with visible BHist representatives, not comparisons inside an external completion quotient.

Assume $\mathsf{hsame}(c, c')$. Compose the displayed comparisons by symmetry and transitivity of hsame:

$$\alpha(a)\ \mathsf{hsame}\ p\ \mathsf{hsame}\ c\ \mathsf{hsame}\ c'\ \mathsf{hsame}\ p'\ \mathsf{hsame}\ \alpha(a').$$

This gives $\mathsf{hsame}(\alpha(a), \alpha(a'))$. Applying the injectivity judgment for α gives $\mathsf{hsame}(a, a')$, proving the readback-injectivity clause.

For source separation, suppose $\mathsf{hsame}(a, a') \Longrightarrow \bot$ and suppose $\mathsf{hsame}(c, c')$. The injectivity clause just proved yields $\mathsf{hsame}(a, a')$, which contradicts the displayed source-separation hypothesis. Hence $\mathsf{hsame}(c, c') \Longrightarrow \bot$. The argument uses only the concrete $\mathsf{S}^{1\uparrow}$ carrier readback, hsame symmetry and transitivity, and the source-map injectivity row supplied by additive cancellation. □

Definition 58.30 (SOne visible finite-rank and prime-support ledgers)**.** *Fix a concrete* $\mathsf{S}^{1\uparrow}$ *history-carrier witness* $\mathsf{SOneHistoryCarrier}(x, y, e, p)$*. A* visible $\mathsf{S}^{1\uparrow}$ bridge ledger *is the BEDC ledger surface obtained from the carrier fields of Definition 58.2: the two* $\mathsf{Real}^{\uparrow}$ *coordinate rows, the* $\mathsf{Prod}^{\uparrow}$ *product row, the unit-equation classifier row, and the displayed point-continuation row for p.*

A bounded-rank certificate *on this visible ledger is a pair $(b, R_{\leq b})$ where b is a displayed BEDC rank row and $R_{\leq b}$ asserts that every prime-support row read from the visible bridge ledger factors through rank at most b.*

A cofinal prime-support evidence row *on the same visible ledger is a predicate C_{prime} such that, for every displayed BEDC rank row b, there is a visible prime-support row q_b in the point-continuation ledger of p whose support does not factor through rank at most b.*

A finite-rank host homologization certificate admitted through the $\mathsf{S}^{1\uparrow}$ carrier/readback bridge *is a host-side finite-rank homologization claim whose theorem-preservation and no-host-leak fields translate its public* $\mathsf{S}^{1\uparrow}$ *consequence to a bounded-rank certificate on the visible ledger, rather than to a host equality or an unrecorded host quotient.*

Lemma 58.31 (SOne finite-rank host certificate induces bounded visible rank)**.** *For a concrete carrier witness* $\mathsf{SOneHistoryCarrier}(x, y, e, p)$*, every finite-rank host homologization certificate admitted through the* $\mathsf{S}^{1\uparrow}$ *carrier/readback bridge induces a bounded-rank certificate on the visible* $\mathsf{S}^{1\uparrow}$ *bridge ledger of Definition 58.30.*

Proof. The carrier surface is fixed by Definition 58.2: the visible data are the two real-coordinate rows, the product row, the unit row, and the point-continuation ledger. The public readback theorem Definition 58.22 exposes exactly these BEDC-side rows and no additional host carrier.

An admitted host homologization certificate can therefore reach the public $\mathsf{S}^{1\uparrow}$ surface only through the standard bridge discipline: its preserved host claim must be listed in the theorem-preservation field, and the no-host-leak field of Definition 9204.5 forbids the host's finite-rank object, host quotient, or host equality from becoming a BEDC export. Hence the finite-rank content has only one BEDC-visible form: an explicit displayed rank bound b together with a ledger assertion that all prime-support rows visible in the $\mathsf{S}^{1\uparrow}$ point ledger factor through rank at most b. This is precisely a bounded-rank certificate on the visible bridge ledger. □

Theorem 58.32 (SOne finite-rank host homologization obstruction)**.** *Let* $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ *be a concrete* $\mathsf{S}^{1\uparrow}$ *history-carrier witness. If the visible* $\mathsf{S}^{1\uparrow}$ *bridge ledger carries cofinal prime-support evidence in the sense of Definition 58.30, then no finite-rank host homologization certificate can be admitted through the* $\mathsf{S}^{1\uparrow}$ *carrier/readback bridge for that ledger.*

Proof. Suppose that such a finite-rank host homologization certificate were admitted. By Definition 58.31, it induces a bounded-rank certificate $(b, R_{\leq b})$ on the visible $\mathsf{S}^{1\uparrow}$ bridge ledger. Thus every visible prime-support row in the point-continuation ledger of p must factor through rank at most b.

Apply the cofinal prime-support evidence to this same displayed bound b. It provides a visible prime-support row q_b in the point-continuation ledger of p whose support does not factor through rank at most b.

The row q_b is visible in the same ledger to which $R_{\leq b}$ applies, so $R_{\leq b}$ says that q_b factors through rank at most b. Cofinality says that the same row does not factor through rank at most b. This contradiction blocks the induced bounded-rank certificate, and therefore blocks the admitted finite-rank host homologization certificate that would induce it. □

58.2 The certificate

Definition 58.33 (SOne source specification)**.** *The source is the* $\mathsf{Prod}^{\uparrow}$ *pair of two* $\mathsf{Real}^{\uparrow}$ *witnesses together with the real-algebra data needed to form* $x^2 + y^2$ *and the displayed unit witness.*

Lean4 (definition) `BEDC.Derived.S1Up.SOneSourceSpec`

Lean4 ✓ `BEDC.Derived.S1Up.SOneSourceSpec history carrier fields`

Definition 58.34 (SOne pattern specification)**.** *The pattern is unit-circle formation from a real pair and a certified proof object for the equation* $x^2 + y^2 \approx 1$.

Definition 58.35 (SOne classifier specification)**.** *The concrete* BHist *classifier relates two* $\mathsf{SOneHistoryCarrier}$ *witnesses* (x, y, e, p) *and* (x', y', e', p') *exactly when* $\mathsf{hsame}(x, x')$ *and* $\mathsf{hsame}(y, y')$ *hold component-wise. Under those component witnesses, the equation and point ledgers are determined by Definition 58.23, and they are transported by Definition 58.43 and Definition 58.44.*

Definition 58.36 (SOne component classifier relation)**.** *For histories* $x, y, e, p, x', y', e', p'$*, the concrete relation* $\mathsf{SOneComponentClassifier}((x, y, e, p), (x', y', e', p'))$ *means the conjunction:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \wedge \mathsf{SOneHistoryCarrier}(x', y', e', p') \wedge \mathsf{hsame}(x, x') \wedge \mathsf{hsame}(y, y').$$

Thus the classifier exposes exactly the two real component comparisons. The equation ledger and the point ledger are not classifier fields; they are recovered by Definition 58.23.

Lean4 (definition) `BEDC.Derived.S1Up.SOneComponentClassifier`

Theorem 58.37 (SOne component classifier stability)**.** *On the carrier* SOneHistoryCarrier*, the relation* SOneComponentClassifier *supplies the stability row required by the semantic* NameCert *for* $\mathsf{S}^{1\uparrow}$*. Concretely:*

1. *it is reflexive, symmetric, and transitive by the equivalence theorem for* hsame *on the two exposed real components;*
2. *any classifier witness projects the two carrier rows and determines* $\mathsf{hsame}(e, e')$ *and* $\mathsf{hsame}(p, p')$ *from its exposed component comparisons;*
3. *the carrier transports from* (x, y, e, p) *to* (x', y', e', p') *along the resulting four coordinate* hsame *witnesses.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier stability`

Proof. Reflexivity, symmetry, and transitivity are inherited componentwise from Definition 3.8. Reflexivity uses the two diagonal component witnesses; symmetry reverses both exposed witnesses; transitivity composes the two exposed witnesses through the middle carrier.

For the hidden ledgers, unpack the classifier witness into four fields: the source carrier row $\mathsf{SOneHistoryCarrier}(x, y, e, p)$, the target carrier row $\mathsf{SOneHistoryCarrier}(x', y', e', p')$, and the exposed component witnesses $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$. Apply Definition 58.23 to these four fields; it returns exactly the equation witness $\mathsf{hsame}(e, e')$ and the point witness $\mathsf{hsame}(p, p')$.

The equation ledger is stable by Definition 58.43. With the component ledgers and the determined point ledger, Definition 58.44 then transports the whole carrier from (x, y, e, p) to (x', y', e', p').

The certificate fit has ten fields: the traditional target is unit-circle component classification; the BEDC source is BHist with RealConstantHistoryCarrier, RealConstantHistoryClassifier, $\mathsf{Prod}^{\uparrow}$, and Cont; the carrier is SOneHistoryCarrier; the classifier is SOneComponentClassifier; the stability obligations are equivalence plus carrier transport; the ledger data are the equation and point witnesses; the core theorem is this stability theorem; exactness is hidden-ledger determinacy; the standard bridge is deferred; and the status is ProofTarget. □

Theorem 58.38 (SOne carrier-classifier ledger obligation surface)**.** *For* $\mathsf{S}^{1\uparrow}$ *over* BHist*, the carrier-classifier ledger obligation surface consists of the* $\mathsf{Real}^{\uparrow}$ *component carrier rows, the* $\mathsf{Prod}^{\uparrow}$ *product readback row, the* Cont *point ledger, the componentwise* hsame *classifier rows, and the* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *ledger rows displayed by Definition 58.33, Definition 58.35, Definition 58.36, and Definition 58.37. These obligations do not claim connectedness, compactness, angle parametrisation, or a standard topological circle bridge.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier ledger obligation surface`

Proof. The carrier part is the concrete $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ row over BHist. Its visible payload names two $\mathsf{Real}^{\uparrow}$ component histories, the displayed product history, and the continuation ledger $\mathsf{Cont}(x, y, p)$.

The classifier part is $\mathsf{SOneComponentClassifier}((x, y, e, p), (x', y', e', p'))$. By Definition 58.36, it contains exactly the two carrier rows and the two exposed component comparisons $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$.

The ledger part is recovered by Definition 58.37. Its proof invokes Definition 58.23 to determine the hidden equation and point rows, then uses the carrier transport surface to keep the certificate inside the same $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ obligations.

No argument in these rows reads a topological circle. The surface is confined to BHist, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, Cont, hsame, and $\mathsf{NameCert}$ data, so connectedness, compactness, angle parametrisation, and standard topological bridges remain outside the obligation list. □

Theorem 58.39 (SOne equation transport obligation surface)**.** *For* $\mathsf{S}^{1\uparrow}$ *over* BHist*, the equation transport obligation surface is the* $\mathsf{Real}^{\uparrow}$ *equation witness row, the fixed unit-history classifier row, the componentwise* hsame *rows, the* Cont *ledger, and the* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *carrier row transported by Definition 58.43 and Definition 58.44. It excludes connectedness, compactness, angle parametrisation, and standard topological circle bridges.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier equation transport obligation surface`

Proof. Begin with the equation row inside $\mathsf{SOneHistoryCarrier}(x, y, e, p)$. The row is a $\mathsf{Real}^{\uparrow}$-typed witness that the displayed equation history is classified against $\mathsf{SOneUnitHistory}$, while the same carrier also records the two real component histories and the point ledger.

Definition 58.43 gives the isolated equation replacement obligation. It changes only the displayed equation witness along $\mathsf{hsame}(e, e')$ and leaves the component carriers and Cont ledger unchanged.

Definition 58.44 gives the full coordinate transport obligation. It combines the two component hsame rows, the equation hsame row, and the point hsame row to produce the transported $\mathsf{SOneHistoryCarrier}(x', y', e', p')$.

These two transport theorems enumerate exactly the equation-facing $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ obligations. They do not provide any connectedness, compactness, angle parametrisation, or standard topological circle bridge row. □

Theorem 58.40 (SOne product-real ledger obligation surface)**.** *For* $\mathsf{S}^{1\uparrow}$ *over* BHist*, the product-real ledger obligation surface is the public readback of the* $\mathsf{Prod}^{\uparrow}$ *point history, the two* $\mathsf{Real}^{\uparrow}$ *component histories, the* Cont *ledger, the transported* hsame *rows, and the* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *readback ledger supplied by Definition 58.22, Definition 58.45, and Definition 58.46. It does not claim connectedness, compactness, angle parametrisation, or any standard topological circle bridge.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier product real ledger obligation surface`

Proof. Definition 58.22 reads a concrete carrier as a product carrier at p, the fixed unit-equation history, two rational witnesses for the real component histories, and the continuation ledger $\mathsf{Cont}(x, y, p)$. This is the public product-real ledger for the source tuple.

Definition 58.45 applies the coordinate transport surface before taking the same readback at (x', y', e', p'). The product row, the real component rows, the equation row, and the continuation ledger remain inside the transported $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ carrier.

Definition 58.46 shows that a component classifier witness already contains enough carrier and hidden-ledger data to recover the transported public package. Thus the product-real ledger surface is exactly the readback, transport, and classifier-readback trio.

The trio is still a BEDC history certificate surface. It records no connectedness, no compactness, no angle parametrisation, and no standard topological circle bridge. □

Theorem 58.41 (SOneUp public NameCert obligation assembly)**.** *For* $\mathsf{S}^{1\uparrow}$ *over* BHist*, the carrier-classifier ledger surface, equation-transport surface, product-real ledger surface, source specification, classifier specification, and component-classifier stability assemble into the public* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *obligation surface. The public fields are exactly the two* $\mathsf{Real}^{\uparrow}$ *component rows, the* $\mathsf{Prod}^{\uparrow}$ *product row, the unit-equation witness, componentwise* hsame*, the* Cont *point ledger, and the* $\mathsf{NameCert}$ *ledger*

rows supplied by Definition 58.38, Definition 58.39, Definition 58.40, Definition 58.33, Definition 58.36, and Definition 58.37. It excludes connectedness, compactness, angle parametrisation, and standard topological circle bridges.

Proof. First project the public carrier and classifier rows from Definition 58.38. This gives the two real component histories, the source-side classifier rows, the hidden equation ledger, and the point ledger over BHist.

Next use Definition 58.39 and Definition 58.40. The equation theorem supplies the unit-equation witness and its hsame transport, while the product-real theorem exposes the $\mathsf{Prod}^{\uparrow}$ point row, the two $\mathsf{Real}^{\uparrow}$ component rows, and the public readback ledger.

Finally apply Definition 58.37. Its component classifier stability transports the exposed component comparisons and determines the hidden ledger rows, so the projected carrier, equation, product, Cont, and $\mathsf{NameCert}$ rows repack as the public $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ obligation surface. Since every projected row is one of these BEDC rows, no connectedness, compactness, angle parametrisation, or standard topological-circle bridge is introduced. □

Definition 58.42 (SOne stability certificate)**.** *The stability certificate consists of:*

1. *reflexivity from reflexivity of the two real components and identity transport of the equation witness*
2. *symmetry by symmetry of the two real components and reverse transport of the equation witness*
3. *transitivity by transitivity of the two real components and composition of equation transport*
4. *compatibility with the* $\mathsf{Prod}^{\uparrow}$ *classifier over* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$
5. *preservation of the certified equation* $x^2 + y^2 \approx 1$

Theorem 58.43 (SOne equation-witness transport)**.** *For histories* x, y, e, e', p*, the concrete* $\mathsf{S}^{1\uparrow}$ *history carrier is stable under replacement of the displayed unit-equation witness along the equation ledger:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(e, e') \Longrightarrow \mathsf{SOneHistoryCarrier}(x, y, e', p).$$

Proof. Unpack $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ into the two $\mathsf{RealConstantHistoryCarrier}$ component fields, the $\mathsf{RealConstantHistoryClassifier}(e, \mathsf{SOneUnitHistory})$ field, and the continuation witness $\mathsf{Cont}(x, y, p)$.

The equation ledger gives $\mathsf{hsame}(e, e')$. Combine it with $\mathsf{hsame}(\mathsf{SOneUnitHistory}, \mathsf{SOneUnitHistory})$ by reflexivity and apply the endpoint-transport theorem for the real-constant history classifier. This produces $\mathsf{RealConstantHistoryClassifier}(e', \mathsf{SOneUnitHistory})$ while the two component carriers and the continuation ledger remain unchanged.

Repack the four fields into $\mathsf{SOneHistoryCarrier}(x, y, e', p)$. The certificate fit is the traditional target unit-circle equation transport; the BEDC source BHist with $\mathsf{RealConstantHistoryClassifier}$; the carrier $\mathsf{SOneHistoryCarrier}$; the classifier hsame on the equation ledger; the stability obligation equation-witness transport; the ledger transported unit-equation proof; the core theorem equation-witness replacement; the exactness claim that the same $\mathsf{S}^{1\uparrow}$ carrier is preserved; the standard bridge deferred outside this theorem; and the status $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier equation witness transport`

Theorem 58.44 (SOne carrier hsame transport)**.** *For histories* $x, y, e, p, x', y', e', p'$*, the concrete* $\mathsf{S}^{1\uparrow}$ *history carrier is stable under componentwise* hsame *transport on all exposed carrier coordinates:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(e, e') \Longrightarrow \mathsf{hsame}(p, p') \Longrightarrow \mathsf{SOneHistoryC}$$

Proof. Unpack $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ into the two $\mathsf{RealConstantHistoryCarrier}$ component fields, the $\mathsf{RealConstantHistoryClassifier}(e, \mathsf{SOneUnitHistory})$ field, and the continuation witness $\mathsf{Cont}(x, y, p)$.

Transport the first and second real-constant carrier fields along $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$. Each step composes the exposed endpoint equality of the real-constant carrier with the supplied history equality, so the same real-constant carrier predicate is obtained at x' and at y'.

Transport the equation classifier by applying $\mathsf{RealConstantHistoryClassifier_endpoint_transport}$ to $\mathsf{hsame}(e, e')$ and the reflexive witness $\mathsf{hsame}(\mathsf{SOneUnitHistory}, \mathsf{SOneUnitHistory})$. This gives $\mathsf{RealConstantHistoryClassifier}(e', \mathsf{SOneUnitHistory})$.

Transport the continuation ledger by the kernel continuation hsame transport theorem, using the component witnesses for x, y, and p. The result is $\mathsf{Cont}(x', y', p')$.

Repack the two transported real-constant carriers, the transported equation classifier, and the transported continuation ledger into $\mathsf{SOneHistoryCarrier}(x', y', e', p')$. The certificate fit is the traditional target unit-circle carrier transport; the BEDC source BHist with real-constant carrier, real-constant classifier, and Cont; the carrier $\mathsf{SOneHistoryCarrier}$; the classifier hsame on the exposed coordinates; the stability obligation carrier transport; the ledger transported endpoints and equation proof; the core theorem this carrier-transport statement; the exactness claim that the same carrier predicate is preserved; the standard bridge deferred outside this theorem; and the status $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier coordinate transport`

Theorem 58.45 (SOne transported public readback)**.** *For histories $x, y, e, p, x', y', e', p'$, a concrete $\mathsf{S}^{1\uparrow}$ history carrier and componentwise transported BEDC history data expose the public readback at the transported tuple:*

$$\mathsf{SOneHistoryCarrier}(x, y, e, p) \Longrightarrow \mathsf{hsame}(x, x') \Longrightarrow \mathsf{hsame}(y, y') \Longrightarrow \mathsf{hsame}(e, e') \Longrightarrow \mathsf{hsame}(p, p') \Longrightarrow \mathsf{SOneProductH}$$

Proof. First apply Definition 58.44 to the displayed carrier and the four supplied hsame witnesses. This produces the transported carrier $\mathsf{SOneHistoryCarrier}(x', y', e', p')$.

Next apply Definition 58.22 to that transported carrier. Its conclusion is exactly the product carrier at p', the unit-equation readback for e', the two rational component witnesses for x' and y', and the transported continuation ledger.

The ten-field fit is: traditional target transported unit-circle public readback; BEDC source $\mathsf{S}^{1\uparrow}$ over BHist, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, and Cont; carrier $\mathsf{SOneHistoryCarrier}$; classifier coordinate hsame; stability coordinate transport; ledger transported equation and point witnesses; core theorem this readback; exactness public carrier and unit-equation exposure; standard bridge deferred; status $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier public readback transport`

Theorem 58.46 (SOne component classifier public readback)**.** *For histories $x, y, e, p, x', y', e', p'$, a component-classifier witness carries the two concrete $\mathsf{S}^{1\uparrow}$ history carrier rows and exposes the public readback at the transported tuple:*

$$\mathsf{SOneComponentClassifier}((x, y, e, p), (x', y', e', p')) \Longrightarrow \mathsf{SOneProductHistoryCarrier}(p') \wedge \mathsf{hsame}(e', \mathsf{SOneUnitHistory}) \wedge$$

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier public readback`

Proof. Unpack the component classifier. By Definition 58.36, its data are the source carrier row $\mathsf{SOneHistoryCarrier}(x, y, e, p)$, the target carrier row $\mathsf{SOneHistoryCarrier}(x', y', e', p')$, and the component witnesses $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$.

Apply Definition 58.23 to the two projected carriers and the two component witnesses. This determines the hidden equation ledger $\mathsf{hsame}(e, e')$ and the hidden point ledger $\mathsf{hsame}(p, p')$.

With the two public component witnesses and the two determined hidden ledgers, Definition 58.44 transports $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ to $\mathsf{SOneHistoryCarrier}(x', y', e', p')$.

Finally apply Definition 58.22 to the transported carrier. The conclusion is precisely the product carrier at p', the unit-equation readback for e', the rational component witnesses for x' and y', and the continuation ledger $\mathsf{Cont}(x', y', p')$. □

Theorem 58.47 (SOne component classifier full readback package)**.** *For histories $x, y, e, p, x', y', e', p'$, a component-classifier witness packages the public readback at the source tuple, the hidden equation and point ledgers, and the public readback at the transported tuple:*

$$\mathsf{SOneComponentClassifier}((x, y, e, p), (x', y', e', p')) \Longrightarrow \Big(\mathsf{SOneProductHistoryCarrier}(p) \wedge \mathsf{hsame}(e, \mathsf{SOneUnitHistory}) \wedge$$

Proof. Unpack the component classifier through Definition 58.36. This gives the source carrier row $\mathsf{SOneHistoryCarrier}(x, y, e, p)$, the target carrier row $\mathsf{SOneHistoryCarrier}(x', y', e', p')$, and the exposed component witnesses $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$.

Apply Definition 58.22 to the source carrier. It produces the first displayed public package: the product carrier at p, the unit-equation readback at e, the two rational component witnesses for x and y, and the continuation ledger $\mathsf{Cont}(x, y, p)$.

Next apply Definition 58.23 to the two carrier rows and the two exposed component comparisons. This returns the hidden equation ledger $\mathsf{hsame}(e, e')$ and the hidden point ledger $\mathsf{hsame}(p, p')$.

For the transported public package, either apply Definition 58.46 directly to the original classifier witness, or apply Definition 58.22 to the target carrier row. The resulting fields are exactly the product carrier at p', the unit-equation readback at e', the rational component witnesses for x' and y', and the continuation ledger $\mathsf{Cont}(x', y', p')$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier full readback package`

Theorem 58.48 (SOne component classifier visible source unit-equation package)**.** *For histories $d_x, d_y, e_t, t, x', y', e', p'$, a component-classifier witness whose source tuple is visibly one-headed packages the rational source data, the hidden ledger transport, and the public target readback:*

$$\mathsf{SOneComponentClassifier}((\mathsf{E1}(d_x), \mathsf{E1}(d_y), \mathsf{E1}(e_t), \mathsf{E1}(t)), (x', y', e', p')) \Longrightarrow \Big(\mathsf{RatHistoryCarrier}(d_x) \wedge \mathsf{RatHistoryCarrie}$$

Proof. Unpack the component classifier through Definition 58.36. Its source row is $\mathsf{SOneHistoryCarrier}(\mathsf{E1}(d_x), \mathsf{E1}(d_y$ with a target carrier row at (x', y', e', p') and exposed component witnesses from $\mathsf{E1}(d_x)$ to x' and from $\mathsf{E1}(d_y)$ to y'.

Apply Definition 58.16 to the visible source row. This gives the two rational coordinate carriers, the rational unit-equation classifier $\mathsf{RatHistoryClassifier}(e_t, \mathsf{E1}(\mathsf{e}))$, and the source continuation ledger $\mathsf{Cont}(\mathsf{E1}(d_x), d_y, t)$.

Next apply Definition 58.23 to the source and target carrier rows together with the exposed component witnesses. The hidden equation and point ledgers are exactly $\mathsf{hsame}(\mathsf{E1}(e_t), e')$ and $\mathsf{hsame}(\mathsf{E1}(t), p')$.

Finally apply Definition 58.46 to the original classifier witness. Its conclusion is the target public readback package: the product carrier at p', the unit-equation readback at e', two rational witnesses for x' and y', and the continuation ledger $\mathsf{Cont}(x', y', p')$. Repack these three packages in the displayed order. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier visible source unit equation package`

Definition 58.49 (SOne ledger policy)**.** *Record both real witnesses, the componentwise real-classifier witnesses used in classifier derivations, and the transported unit equation proof.*

Lean4 (definition) `BEDC.Derived.S1Up.SOneLedgerPolicy`

Lean4 ✓ `BEDC.Derived.S1Up.SOneLedgerPolicy component readback`

Definition 58.50 ($\mathsf{S}^{1\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{S}^{1\uparrow}$*.*

Theorem 58.51 (SOne semantic NameCert field assembly)**.** *The concrete carrier* $\mathsf{SOneHistoryCarrier}$*, the component classifier relation of Definition 58.36, the source and pattern specifications of Definition 58.33 and Definition 58.34, the stability certificate of Definition 58.42, and the ledger policy of Definition 58.49 supply the five semantic certificate rows needed for the* $\mathsf{S}^{1\uparrow}$ $\mathsf{NameCert}$*: carrier and public readback, hidden equation/point ledger determinacy, equation replacement, full coordinate carrier transport, and transported public readback.*

Proof. The carrier and public-readback row is Definition 58.22. It reads a concrete $\mathsf{SOneHistoryCarrier}(x, y, e, p)$ as a product carrier, the fixed unit-equation history, rational witnesses for x and y, and the continuation ledger at p.

For the classifier row, unpack Definition 58.36. A classifier witness supplies the source carrier row, the target carrier row, and the two component witnesses $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$. Applying Definition 58.23 to these four projected fields determines the hidden ledgers $\mathsf{hsame}(e, e')$ and $\mathsf{hsame}(p, p')$.

The stability row first replaces the equation witness through Definition 58.43. With the two public component witnesses and the two hidden ledgers now in hand, Definition 58.44 transports the full concrete carrier from (x, y, e, p) to (x', y', e', p').

The final readback row is Definition 58.45. It applies the transported carrier to expose the product carrier at p', the transported unit-equation witness, the rational component witnesses at x' and y', and the continuation ledger $\mathsf{Cont}(x', y', p')$. These five rows are exactly the semantic payload required by the concrete $\mathsf{NameCert}$ data for $\mathsf{S}^{1\uparrow}$. □

Lean4 ✓ `BEDC.Derived.S1Up.SOne semantic namecert field assembly`

Lean4 ✓ `BEDC.Derived.S1Up.SOne semantic name certificate fields`

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier semanticNameCert field assembly`

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier semantic name certificate fields`

Theorem 58.52 (Semantic SOneUp history certificate)**.** *The concrete data* $\mathsf{SOneHistoryCarrier}$*, Definition 58.33, Definition 58.34, Definition 58.35, Definition 58.42, and Definition 58.49 assemble a semantic* $\mathsf{NameCert}$ *for* $\mathsf{S}^{1\uparrow}$ *over* BHist*.*

Proof. Definition 58.51 supplies the fieldwise semantic payload for the displayed carrier, classifier, source, pattern, stability, and ledger data. Packing those rows under the obligations named in Definition 58.50 gives the semantic $\mathsf{NameCert}$ for $\mathsf{S}^{1\uparrow}$ over BHist. □

Lean4 ✓ `BEDC.Derived.S1Up.sone history semantic name certificate`

Definition 58.53 (SOne public constructor packet)**.** *A public constructor packet for* $\mathsf{S}^{1\uparrow}$ *is a finite* BHist *packet whose displayed rows are exactly two* $\mathsf{Real}^{\uparrow}$ *component histories* x *and* y*, the* $\mathsf{Prod}^{\uparrow}$ *point row* p *for the ordered pair, the unit-equation witness row* e *certifying* $x^2 + y^2 \approx 1$*, the componentwise* $\mathsf{hsame}(x, x')$ *and* $\mathsf{hsame}(y, y')$ *transport rows used by the classifier, the* Cont *ledger carrying the point and equation endpoints, and the* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *provenance row. The packet has no field for an angle parameter, quotient representative, connectedness witness, compactness witness, or standard circle bridge.*

Theorem 58.54 (SOne public constructor scope)**.** *Every accepted public constructor packet of Definition 58.53 reads only the two* $\mathsf{Real}^{\uparrow}$ *component rows, the* $\mathsf{Prod}^{\uparrow}$ *point row, the unit-equation witness, the componentwise* hsame *classifier rows, the* Cont *continuation ledger, and the* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *provenance row. Conversely, those displayed rows repack into the same public constructor surface whenever the carrier and equation transport obligations of Definition 58.52 are present.*

Proof. Project the packet to its displayed rows. The definition lists only the two real components, the product point, the equation witness, the componentwise classifier transports, the continuation ledger, and the certificate provenance.

For the repacking direction, first use Definition 58.41 to align the displayed rows with the public $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ fields. Then apply Definition 58.52 to supply the carrier, classifier, stability, and ledger envelope.

The packet has no row for angle parametrisation, quotient representatives, connectedness, compactness, or a standard bridge. A public read therefore cannot leave the displayed BEDC rows, and the same rows reconstruct the public constructor surface. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier public constructor scope`
Lean4 ✓ `BEDC.Derived.S1Up.SOnePublicConstructorPacket scope`

Theorem 58.55 (SOne public endpoint exactness)**.** *Let P and P' be accepted public constructor packets for* $\mathsf{S}^{1\uparrow}$*. If their two real component rows are related componentwise by* hsame*, then the unit-equation witness rows and product-point ledger rows exposed by P and P' are the same public* $\mathsf{S}^{1\uparrow}$ *endpoints under the classifier of Definition 58.36. No hidden angle, quotient, or host-circle endpoint can distinguish the two accepted packets.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier public endpoint exactness`

Proof. Read both packets through Definition 58.54. The only classifier data available are the two component rows and their componentwise hsame witnesses.

Apply Definition 58.23. The component classifier determines the equation ledger and the point ledger from those two real-component comparisons.

Finally transport the carrier with Definition 58.43 and Definition 58.44. The transported equation witness and product point are exactly the public endpoints read by the two packets, and no undisplayed host-circle row remains available to separate them. □

Theorem 58.56 (SOne public constructor compatibility)**.** *Every public* $\mathsf{S}^{1\uparrow}$ *constructor row built from two* $\mathsf{Real}^{\uparrow}$ *component rows, the* $\mathsf{Prod}^{\uparrow}$ *point readback, the unit-equation witness, componentwise* hsame*, and the* Cont *point ledger repacks into the same public* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *surface. Conversely, every public constructor read from* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *exposes only those component, product, equation, transport, continuation, and provenance rows. It exposes no connectedness, compactness, angle parametrisation, or standard circle bridge.*

Proof. Start with the displayed public constructor row. Apply Definition 58.41 to identify the public $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ fields as the two $\mathsf{Real}^{\uparrow}$ component rows, the $\mathsf{Prod}^{\uparrow}$ point row, the unit-equation witness, componentwise hsame, and the Cont point ledger.

The semantic certificate is then supplied by Definition 58.52. Its carrier, classifier, source, stability, and ledger rows give the certificate envelope into which the public constructor fields repack.

For the reverse read, project the public surface through Definition 58.40, Definition 58.39, and Definition 58.46. These results recover exactly the product readback, equation transport, component comparisons, and continuation ledgers available to a public constructor.

No further row is present in the carrier or in the semantic certificate. Therefore the public constructor interface is compatible with the unit-circle equation and exposes only the listed BEDC history rows, while connectedness, compactness, angle parametrisation, and standard circle bridges remain outside the exported surface. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneHistoryCarrier public constructor compatibility`

Primitive boundary. $\mathsf{S}^{1\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 58.57 (Zero-axiom encoding note)**.** *The interface records explicit component witnesses and equation transport, with no quotient-type representative and no use of* Quot.sound*.*

Theorem 58.58 (SOne public constructor standard bridge boundary)**.** *Any standard-circle comparison admitted by this chapter consumes exactly the public constructor packet rows of Definition 58.53: two* $\mathsf{Real}^{\uparrow}$ *component histories, the* $\mathsf{Prod}^{\uparrow}$ *point row, the unit-equation witness, componentwise* hsame *transports, the continuation ledger, and certificate provenance. Conversely, those rows supply the BEDC side of a standard-circle bridge. The consumable surface excludes angle quotients, trigonometric libraries, covering-space data, winding-number data, and topological-group structure.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneUp StdBridge`

Proof. First project an admitted standard-circle comparison to the public constructor surface. By Definition 58.53, its exported rows are exactly the two real component histories, product point row, unit-equation witness, componentwise transports, continuation ledger, and provenance fields.

Next apply Definition 58.56 and Definition 58.52. These identify the displayed rows with the public $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ carrier, classifier, stability, and ledger envelope. Conversely, the public readback and constructor interface of Definition 58.22, Definition 58.45, and Definition 58.46 repacks them as the BEDC side of a standard-circle comparison.

The cited projections never manufacture angle representatives, trigonometric operations, covering-space lifts, winding-number ledgers, or a topological-group law. Those structures remain outside the bridge boundary. □

Theorem 58.59 (SOne Real metric bridge boundary)**.** *The public* $\mathsf{S}^{1\uparrow}$ *bridge boundary over the* $\mathsf{Real}^{\uparrow}$ *product carrier consists only of the two* $\mathsf{Real}^{\uparrow}$ *component rows, the product carrier ledger, the unit-equation witness, the componentwise classifier, and equation transport. It does not expose angle parametrisation, winding data, covering quotients, compactness, or connectedness.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneRealMetricBridgeBoundary public rows`

Proof. Read the carrier through the child file section 58.1. The carrier readbacks expose the two real component histories, the product-history row, the continuation ledger, and the displayed unit-equation witness.

Read the certificate through the child file certificate surface. The component classifier exposes only hsame on the two real coordinates, while the hidden equation and point ledgers are determined and transported by the equation transport and carrier transport rows.

Repack those rows as the public bridge boundary. Every exported field is one of the RealUp component, product carrier, unit-equation, component classifier, or equation-transport rows already displayed in this chapter, so no angle, winding, covering, compactness, or connectedness datum is part of the boundary. □

Theorem 58.60 (SOne real metric coordinate boundary)**.** *The standard topological bridge boundary of an admitted* $\mathsf{S}^{1\uparrow}$ *packet exposes exactly two* $\mathsf{Real}^{\uparrow}$ *component rows, their rational component witnesses, the product carrier row, the unit-equation row, a unary bridge ledger, and the endpoint equation* $\mathsf{hsame}(\mathsf{bridge}, \mathsf{append}(\mathsf{point}, \mathsf{equation}))$*. It exposes no angle parametrisation, winding datum, quotient presentation, compactness, connectedness, or metric-completeness row.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneStandardTopologicalBridgeBoundary component rows`

Proof. Combine Definition 58.59 with the component readback rows in section 58.1. The bridge boundary already limits the public circle read to the Real component rows, product carrier ledger, unit-equation witness, componentwise classifier, and equation transport.

The component readbacks supply the rational witnesses for the two Real rows, the unary bridge ledger, and the displayed endpoint equation. The checked Lean row packages exactly those component fields, so no angle parametrisation, winding datum, quotient presentation, compactness, connectedness, or metric-completeness row is exported. □

Theorem 58.61 (SOne RegSeqRat metric observation boundary)**.** *An admitted* $\mathsf{S}^{1\uparrow}$ *point over the* $\mathsf{Real}^{\uparrow}$ *product carrier exposes to any metric-observation consumer only the two component* $\mathsf{Real}^{\uparrow}$ *finite-window read surfaces, the* $\mathsf{RegSeqRat}$ *stream-name and real-seal rows attached to those components, the product carrier ledger, the unit-equation witness, the componentwise* hsame *classifier rows, and equation transport. Compactness, connectedness, angle parametrisation, winding data, quotient-circle presentation, and metric-completeness rows are not exported by this boundary.*

Proof. Start from the public $\mathsf{S}^{1\uparrow}$ bridge boundary in Definition 58.59. It already restricts the circle read to the two real component rows, the product carrier ledger, the unit-equation witness, the componentwise classifier, and equation transport.

Now read each real component through Definition 38.342. The component access there is a finite-window read surface, so a metric-observation consumer sees only the selected real rows rather than an ambient metric-complete real line.

For the regular-sequence presentation, use Definition 38.276 for the stream-name rows and Definition 38.278 for the finite-window real-seal rows. These are attached componentwise to the two $\mathsf{Real}^{\uparrow}$ reads and do not add a new circle-level carrier.

Combining the three component readings with the $\mathsf{S}^{1\uparrow}$ bridge boundary gives exactly the listed consumer surface. Since no cited surface exports compactness, connectedness, angle data, winding data, quotient-circle presentation, or metric-completeness rows, none of those rows is available through the admitted $\mathsf{S}^{1\uparrow}$ observation boundary. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneRegSeqRatMetricObservationBoundary public rows`

Theorem 58.62 (SOne public certificate boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *boundary is exactly the* $\mathsf{Real}^{\uparrow}$ *pair carrier, product ledger, unit-equation witness, componentwise* hsame *classifier, equation transport, and finite-window* $\mathsf{RegSeqRat}$ *read surfaces isolated by the* $\mathsf{S}^{1\uparrow}$ *bridge and metric-observation boundaries. No compactness, connectedness, angle, quotient, or completeness row is part of the public certificate boundary.*

Lean4 ✓ `BEDC.Derived.S1Up.SOne public namecert obligation assembly`

Proof. Apply Definition 58.59. It identifies the public circle boundary with the two $\mathsf{Real}^{\uparrow}$ component rows, product carrier ledger, unit-equation witness, componentwise classifier, and equation transport.

Apply Definition 58.61. It refines the component access to the finite-window RegSeqRat stream-name and real-seal read surfaces while preserving the same product, equation, classifier, and transport rows.

The public $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ can therefore export only the intersection of those already isolated rows. Since the first boundary excludes angle data, winding data, covering quotients, compactness, and connectedness, and the second also excludes quotient-circle presentation and metric-completeness rows, the certificate has exactly the displayed RealUp-pair, equation, classifier, transport, and finite-window read surface. □

Theorem 58.63 (SOne componentwise classifier equation transport)**.** *Let P and Q be accepted public $\mathsf{S}^{1\uparrow}$ packets over* BHist*. Suppose the two* $\mathsf{Real}^{\uparrow}$ *component rows of P and Q are related by the componentwise* hsame *classifier of Definition 58.36. Then the unit-equation witness row, product point ledger row, and both* RegSeqRat *finite-window read surfaces of P transport to the corresponding public rows of Q. This consumer reads no angle, quotient-circle, winding, compactness, or connectedness datum.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneComponentClassifier equation transport`

Proof. Apply Definition 58.59. It restricts the public circle packet to the two $\mathsf{Real}^{\uparrow}$ component rows, the product ledger, the unit-equation witness, the componentwise classifier, and the equation-transport row.

Use Definition 58.36 to compare the two accepted packets component by component. Hidden-ledger determinacy in that definition identifies the product-point continuation and the unit-equation witness once the two displayed $\mathsf{Real}^{\uparrow}$ component rows are related by hsame.

Apply Definition 58.61. The same two component comparisons refine to the finite-window RegSeqRat stream-name and real-seal read surfaces on both coordinates, without adding a circle-level carrier.

Finally use Definition 58.62. The public certificate contains exactly those RealUp, product, equation, classifier, and finite-window rows, so the componentwise classifier comparison is sufficient to transport the whole public finite-window circle boundary while leaving all topological circle data unread. □

Theorem 58.64 (SOne standard topological bridge boundary)**.** *Any standard topological-circle bridge for $\mathsf{S}^{1\uparrow}$ may read only the two $\mathsf{Real}^{\uparrow}$ component finite-window surfaces, the product ledger, the unit-equation witness, the componentwise classifier, and equation transport. Compactness, connectedness, angle parametrisation, winding data, quotient-circle presentation, and metric-completeness rows remain outside the bridge.*

↪ variant `BEDC.Derived.S1Up.SOneStandardTopologicalBridgeBoundary component rows`

Proof. Begin with Definition 58.59. It confines a public circle bridge to the two real component rows, product carrier ledger, unit-equation witness, componentwise classifier, and equation transport.

Refine the two component reads through Definition 58.61. The real components may be consumed through their finite-window RegSeqRat read surfaces, but that refinement still does not expose compactness, connectedness, angle data, winding data, quotient presentation, or metric-completeness rows.

Finally apply Definition 58.62. It identifies the public certificate boundary with the same RealUp-pair, product, equation, classifier, transport, and finite-window read rows. Therefore the

standard topological bridge boundary has exactly those readable rows and no topology data outside the displayed certificate surface. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneStandardTopologicalBridgeBoundary public rows`

Theorem 58.65 (SOne standard topological circle acceptance factorization)**.** *Every accepted standard topological-circle bridge row for* $\mathsf{S}^{1\uparrow}$ *factors through exactly the* $\mathsf{Real}^{\uparrow}$*-pair carrier, finite-window* RegSeqRat *component reads, product ledger, unit-equation witness, componentwise classifier, and equation transport isolated by Definition 58.59, Definition 58.61, and Definition 58.62. No compactness, connectedness, angle, winding, quotient-circle, or metric-completeness row can enter the bridge unless supplied by a later certificate.*

Lean4 ✓ `BEDC.Derived.S1Up.SOneStandardTopologicalCircleAcceptance factorization public rows`

Proof. Project the accepted bridge row through Definition 58.64. The projection leaves only the two $\mathsf{Real}^{\uparrow}$ component rows, product ledger, unit-equation witness, componentwise classifier, and equation transport.

Refine those component rows with Definition 58.61. Each real component is read by a finite-window RegSeqRat stream-name and real-seal surface, and the refinement preserves the same product, equation, classifier, and transport rows.

Finally use Definition 58.62. It shows that the public certificate has no additional readable coordinate beyond the RealUp-pair carrier, finite-window reads, product ledger, unit-equation witness, componentwise classifier, and equation transport. Hence topology-specific compactness, connectedness, angle, winding, quotient-circle, and metric-completeness rows are absent unless a later certificate supplies them. □

Theorem 58.66 (SOne standard bridge componentwise transport stability)**.** *Let P and Q be accepted public* $\mathsf{S}^{1\uparrow}$ *packets. Suppose their two* $\mathsf{Real}^{\uparrow}$ *component rows are related by Definition 58.36. If P admits a standard topological-circle bridge row, then Q admits a standard topological-circle bridge row with the same readable* $\mathsf{Real}^{\uparrow}$ *component finite-window surfaces, product ledger, unit-equation witness, componentwise classifier, and equation transport. Compactness, connectedness, angle, winding, quotient-circle, and metric-completeness rows remain unavailable.*

↪ variant `BEDC.Derived.S1Up.SOneStandardBridgeComponentwiseTransport stability`

↪ variant `BEDC.Derived.S1Up.SOneStandardTopologicalBridgeBoundary componentwise transport stability`

Proof. Apply Definition 58.63. The componentwise classifier relation transports the product ledger, unit-equation witness, equation-transport row, and the two finite-window RegSeqRat read surfaces from P to Q.

Next apply Definition 58.64. The standard bridge row for P has no readable coordinate outside those component, product, equation, classifier, transport, and finite-window rows, so the transported data already identify the public bridge boundary available to Q.

Finally use Definition 58.65. The transported boundary repacks as the same public standard topological-circle bridge surface for Q, while compactness, connectedness, angle, winding, quotient-circle, and metric-completeness data remain absent from the bridge. □

Lean4 ✓ `BEDC.Derived.S1Up.SOneStandardBridge componentwise transport stability`

Theorem 58.67 (SOne RealAlgOrder equation consumer boundary)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$*-facing equation consumer of an accepted* $\mathsf{S}^{1\uparrow}$ *packet factors through the two* $\mathsf{Real}^{\uparrow}$ *component rows, the*

product ledger, the unit-equation witness, the componentwise classifier, equation transport, finite-window RegSeqRat *reads, and the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows. Such a consumer can read the circle equation only through this displayed component-and-equation surface.*

It cannot read compactness, connectedness, angle, winding, quotient-circle data, metric-completeness data, or host real extensionality as an alternate equation source.

Proof. First project the accepted packet through Definition 58.59. The only real-facing rows left available are the two $\mathsf{Real}^{\uparrow}$ components, the product ledger, the unit-equation witness, the componentwise classifier, and equation transport.

Next refine the component reads by Definition 58.61. This replaces any metric-facing component observation by finite-window RegSeqRat reads and the visible BHist, hsame, Cont, Pkg, and NameCert ledger rows.

Apply Definition 58.62. It exhausts the public certificate by the same carrier, component, product, equation, classifier, transport, and finite-window rows, so no hidden host-real row remains for an equation consumer.

Finally use Definition 58.65. The standard acceptance surface repacks those rows without adding compactness, connectedness, angle, winding, quotient-circle, metric-completeness, or host real extensionality data. Hence the $\mathsf{RealAlgOrder}^{\uparrow}$ equation consumer boundary is exactly the displayed component-and-equation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$**.**

Constructive story: A circle point is generated by placing two $\mathsf{Real}^{\uparrow}$ constant histories side by side, using the product continuation ledger to assemble the point history p, and carrying a separate equation history e that certifies x^2+y^2 against the real unit; empty points, zero-headed points, and pairs without that unit-equation witness are outside the carrier. The classifier deliberately exposes only the two component comparisons $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$: once those are present, the hidden equation ledger and point ledger are determined and transported, so the construction never quotients by angles or by a host $\mathbb{R}/\mathbb{Z}$ presentation. The source specification is the product of two real witnesses together with the real-algebra rows needed to form the unit equation, the pattern specification is unit-circle formation from that pair and its certified equation object, and the ledger keeps the rational tails of the two real constants, the product continuation, the equation witness, and the transported component rows visible. The five name-certificate obligations reduce to componentwise history-equality: an admitted pair with its unit equation inhabits the carrier, reflexivity, symmetry, and transitivity are inherited from the two real components, and full carrier transport works because the hidden equation and point witnesses can be recovered from those component comparisons. A point history h lies in the signature gap of a $\mathsf{S}^{1\uparrow}$ token p when the source-domain accepts h as such a real pair with unit-equation ledger, the signature relation pairs it with an answer circle point and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the two real components and their equation witness match the point encoded by p. The chapter registers this component-and-equation construction under the name $\mathsf{S}^{1\uparrow}$ and records a paper-level standard topological bridge over exactly the component, product, equation, classifier, transport, and finite-window rows; it introduces no trigonometric library, winding theory, covering-space quotient, or topological-group structure.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{S}^1\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The public scope is the RealUp-pair carrier, product ledger, unit-equation witness, componentwise classifier, equation transport, and finite-window read surface displayed in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{S}^1\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.S1Up.SOneUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: The chapter does not claim compactness, connectedness, angle parametrisation, winding theory, quotient-circle presentation, or metric-completeness examples.

Upgrade path: no further upgrade – chapter at top of formal axis

The $\mathsf{Continuous}^{\uparrow}$ hub reads pointwise-modulus and graph-stability rows over metric source and target certificates through chapter 57, and supplies the pointwise-modulus face consumed by chapter 130.

58.3 The certificate

Definition 58.68 (ContinuousMap source specification)**.** *The source is a pair of certified metric spaces and the totally bounded subset discipline on the domain.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousSourceSpec`

Definition 58.69 (ContinuousMap pattern specification)**.** *The pattern is graph-with-modulus formation over certified metric spaces and totally bounded source witnesses. A witness supplies the graph relation, the domain and codomain classifier ledgers, and the unary modulus relation that converts a target precision request into the source precision obligation needed for the graph.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousPatternSpec`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousPatternSpec comp carrier`

Definition 58.70 (ContinuousMap classifier specification)**.** *The classifier relates two continuous-map witnesses when their graph outputs agree under the codomain classifier on every pair of source histories related by the source classifier, and when their unary modulus ledgers mutually refine each other for every precision history admitted by the precision classifier.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousClassifierSpec`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousClassifierSpec target cert readback`

Definition 58.71 (ContinuousMap stability certificate)**.** *The stability certificate consists of:*

1. *continuous-map classifier reflexivity*
2. *continuous-map classifier symmetry*
3. *continuous-map classifier transitivity*
4. *modulus validity for uniform continuity*
5. *graph and modulus replacement under the continuous-map classifier*
6. *closure under identity*
7. *closure under composition by composing moduli*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier stability certificate fields`

Theorem 58.72 (Continuous modulus chain factorization)**.** *For histories s, δ_1, δ_2, t, assume the two-link chain*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

Then there is a history m such that both one-step links are certified:

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Proof. Unpack the chain into unary data for s, δ_1, δ_2, and t, together with a middle history m and continuation ledgers $s \cdot \delta_1 = m$ and $m \cdot \delta_2 = t$.

The first ledger and the unary source and modulus data assemble

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m).$$

Unary continuation closure then makes the middle history unary. The second ledger and the unary δ_2 and target data assemble

$$\mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Taking this middle history as the existential witness gives the displayed factorization of the chain into its two certified one-step modulus witnesses. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain factorizes`

Theorem 58.73 (Continuous modulus chain middle deterministic factorization)**.** *For histories s, δ_1, δ_2, t, a continuous modulus chain determines a middle history m such that the two one-step modulus witnesses factor the chain, and every other first-leg middle endpoint is the same BEDC history:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \exists m.\ \mathsf{ContinuousModulusWitness}(s, \delta_1, m) \wedge \mathsf{ContinuousModulusWitness}(m, \delta$$

$$\wedge\ \forall m'.\ \mathsf{Cont}(s, \delta_1, m') \Longrightarrow \mathsf{Cont}(m', \delta_2, t) \Longrightarrow \mathsf{hsame}(m, m').$$

Proof. Unpack the chain into unary data and a displayed middle history m with continuation ledgers $s \cdot \delta_1 = m$ and $m \cdot \delta_2 = t$. As in Theorem 59.96, the first ledger gives the first witness and unary continuation closure supplies unary data for m, so the second ledger gives the second witness. If another m' satisfies the same first continuation ledger, Cont determinacy for $s \cdot \delta_1$ yields $\mathsf{hsame}(m, m')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain middle deterministic factorization`

Theorem 58.74 (Continuous modulus chain target determinacy)**.** *For histories $source, first, second, target, targe$ two modulus-chain witnesses with the same source and the same two modulus histories determine the same target up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first, second, target')$$

Proof. Unpack both chains. The first continuation ledgers share *source* and $first$, so continuation determinacy identifies the two middle histories up to hsame. Transport the second continuation comparison along this middle sameness, keeping the common *second* history fixed; continuation respect for hsame then gives $\mathsf{hsame}(target, target')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain target deterministic`

Theorem 58.75 (Continuous modulus chain append-suffix target determinacy)**.** *For histories $q, source, first, second, target, target'$, two modulus chains with the same source and moduli whose targets share the right suffix q determine the unsuffixed targets up to* hsame*.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain append suffix target deterministic`

Proof. Target determinacy identifies $\mathsf{append}(target, q)$ with $\mathsf{append}(target', q)$. Cancel the common right suffix. □

Theorem 58.76 (Continuous modulus chain result determinacy)**.** *For histories $s, \delta_1, \delta_2, t, t'$, two continuous modulus chains with the same source and modulus ledgers determine the same target history:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t') \Longrightarrow \mathsf{hsame}(t, t').$$

Proof. Unpack the two chains into middle histories m and m' with first ledgers $s \cdot \delta_1 = m$ and $s \cdot \delta_1 = m'$, and second ledgers $m \cdot \delta_2 = t$ and $m' \cdot \delta_2 = t'$. Determinacy of the first continuation gives $\mathsf{hsame}(m, m')$. Continuation compatibility with this middle history equality and reflexivity of δ_2 transports the two second ledgers to obtain $\mathsf{hsame}(t, t')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain result deterministic`

Theorem 58.77 (Continuous modulus chain source determinacy)**.** *For histories $source, source', first, second, tar$ two modulus-chain witnesses with the same two modulus histories and the same target determine the same source up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source', first, second, target)$$

Proof. Unpack both chains into middle histories and continuation ledgers. The second continuation ledgers have common suffix *second* and common target, so right cancellation identifies the two middle histories. The first continuation ledgers then have common suffix $first$ and endpoints identified by that middle sameness; common-suffix cancellation gives $\mathsf{hsame}(source, source')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain source deterministic`

Theorem 58.78 (Continuous modulus chain append-suffix source determinacy)**.** *For histories $q, source, source', first, second, target$, two modulus chains with the same moduli and target whose sources share the right suffix q determine the unsuffixed sources up to* hsame*.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain append suffix source deterministic`

Proof. Source determinacy identifies $\mathsf{append}(source, q)$ with $\mathsf{append}(source', q)$. Cancel the common right suffix. □

Theorem 58.79 (Continuous modulus chain first determinacy)**.** *For histories $source, first, first', second, target$, two modulus-chain witnesses with the same source, second modulus, and target determine the same first modulus up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first', second, target)$$

Proof. Unpack both chains. Right cancellation on the two second-leg continuation ledgers identifies the middle histories. Transport the second first-leg ledger along this middle sameness so that both first-leg ledgers have the same source and the same middle endpoint. Left cancellation of continuation then gives $\mathsf{hsame}(first, first')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain first deterministic`

Theorem 58.80 (Continuous modulus chain second determinacy)**.** *For histories $source, first, second, second', tar$ two modulus-chain witnesses with the same source, first modulus, and target determine the same second modulus up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first, second', target)$$

Proof. Unpack both chains into their middle histories. Determinacy of $\mathsf{Cont}(source, first, -)$ identifies those middle histories. After transporting across this middle equality, the second-leg ledgers have common left endpoint and common target, so continuation left cancellation gives $\mathsf{hsame}(second, second')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain second deterministic`

Theorem 58.81 (Continuous modulus chain composite closure)**.** *For unary histories s, δ_1, δ_2, t, a two-link modulus chain from s through δ_1 and δ_2 to t, together with a continuation ledger composing δ_1 and δ_2 into δ, determines a single continuous modulus witness from s to t using δ:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \mathsf{ContinuousModulusWitness}(s, \delta, t).$$

Proof. Unpack the chain into unary endpoint and modulus data plus a middle history with ledgers $m = s \cdot \delta_1$ and $t = m \cdot \delta_2$. The composite modulus is unary by closure of unary continuation. The source-to-target continuation is the associativity equality $(s \cdot \delta_1) \cdot \delta_2 = s \cdot (\delta_1 \cdot \delta_2)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain composite closed`

Theorem 58.82 (Continuous modulus chain endpoint cycle empties moduli)**.** *If* $\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2,$ *and* $\mathsf{hsame}(s, t)$*, then*

$$\mathsf{hsame}(\delta_1, \mathsf{e}) \wedge \mathsf{hsame}(\delta_2, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain endpoint cycle moduli empty`

Proof. Unpack the chain as $\mathsf{Cont}(s, \delta_1, m)$ and $\mathsf{Cont}(m, \delta_2, t)$. Transport the second ledger across $\mathsf{hsame}(s, t)$ and apply mutual-extension tail cancellation. □

Theorem 58.83 (Continuous modulus witness composite public readback)**.** *For histories source, first, second, mi two certified one-step modulus witnesses*

$$\mathsf{ContinuousModulusWitness}(source, first, middle)$$

and

$$\mathsf{ContinuousModulusWitness}(middle, second, target)$$

together with $\mathsf{Cont}(first, second, composite)$ *determine the composite witness*

$$\mathsf{ContinuousModulusWitness}(source, composite, target).$$

Any displayed witness from source through the same composite has target history hsame *to target.*

Proof. The two one-step witnesses supply the source, modulus, middle, and target carriers plus the two continuation ledgers, hence assemble a $\mathsf{ContinuousModulusChain}(source, first, second, target)$. Composite closure turns this chain and the displayed modulus composition into the one-step composite witness. If another target is displayed for the same source and composite modulus, target determinacy for continuous modulus witnesses identifies it with *target*. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness composite public readback`

Theorem 58.84 (Continuous function carrier modulus-chain replacement)**.** *For histories source, map, target, oldM assume an existing continuous-function carrier, a two-step modulus chain from the same target history to a certificate history, and a continuation ledger composing the two modulus steps:*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map, target, oldMod, oldCert),\\ &\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert),\\ &\mathsf{Cont}(\delta_1, \delta_2, \delta).\end{aligned}$$

Then the carrier admits the composite modulus and the chain certificate:

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert).$$

Proof. Unpack the given carrier into unary data for $source$, map, $target$, $oldMod$, and $oldCert$. Keep the graph ledger $\mathsf{Cont}(source, map, target)$; the old modulus and certificate are used only to expose the source, map, and target carrier data.

Apply Theorem 59.105 to the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$ and the composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. This yields the replacement modulus witness

$$\mathsf{ContinuousModulusWitness}(target, \delta, cert).$$

Unpack this witness to obtain unary data for δ and $cert$, together with the modulus-certificate ledger $\mathsf{Cont}(target, \delta, cert)$. Reassemble $\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert)$ from the preserved source, map, and target data, the preserved graph ledger, and this new modulus witness.

The ten-field certificate fit is the following. The traditional target is uniform-continuity data whose modulus certificate may be recomputed through a two-link modulus chain. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is carrier preservation under modulus-chain replacement. The ledger consists of source, map, target, old modulus, old certificate, two modulus steps, the composite modulus, and the terminal certificate. The core theorem is this carrier-level replacement, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus chain replacement`

Theorem 58.85 (Prefixed continuous function carrier graph-chain closure)**.** *For histories $p, source, map, target, \delta_1$ unary source, map, target, and visible-prefix histories, a graph continuation ledger, a two-step modulus chain from the target history to a certificate history, and a continuation ledger composing the two modulus steps determine the prefixed graph carrier with the composite modulus certificate:*

$$\begin{aligned}&\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(source) \Longrightarrow \\ &\mathsf{UnaryHistory}(target) \Longrightarrow \mathsf{UnaryHistory}(map) \Longrightarrow \\ &\mathsf{Cont}(source, map, target) \Longrightarrow \mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert) \Longrightarrow \mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \\ &\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert).\end{aligned}$$

Proof. First apply Theorem 59.113 to the prefix p, the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$, and the composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. This gives the modulus witness

$$\mathsf{ContinuousModulusWitness}(p \cdot target, \delta, p \cdot cert).$$

Unpack this witness into unary data for $p \cdot target$, the composite modulus δ, and $p \cdot cert$, together with the certificate ledger $\mathsf{Cont}(p \cdot target, \delta, p \cdot cert)$.

The graph ledger is prefixed directly from $\mathsf{Cont}(source, map, target)$. Associativity of history append gives

$$(p \cdot source) \cdot map = p \cdot (source \cdot map) = p \cdot target.$$

Unary append closure applied to p and $source$ supplies the unary source history $p \cdot source$. The unary map hypothesis is reused, and the prefixed target, composite modulus, and prefixed certificate data come from the modulus witness. These fields assemble $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$.

The ten-field certificate fit is the following. The traditional target is uniformly continuous map composition under visible finite-prefix evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is carrier preservation from graph data and a prefixed two-link modulus chain. The ledger consists of source, map, target, two modulus steps, the composite modulus, the terminal certificate, and the visible prefix. The core theorem is this carrier-level closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain closed`

Theorem 58.86 (Prefixed continuous function carrier graph-chain public readback)**.** *For histories $p, source, map, target, target', \delta_1, \delta_2, \delta, cert, cert'$, assume unary histories p, source, target, and map, a graph ledger $\mathsf{Cont}(source, map, target)$, a two-link modulus chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cer$ and a composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. If a displayed prefixed continuous-function carrier has target $p \cdot target'$ and certificate $p \cdot cert'$,*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', \delta, p \cdot cert'),$$

then the public target and certificate are determined:

$$\mathsf{hsame}(target, target') \quad \textit{and} \quad \mathsf{hsame}(cert, cert').$$

Proof. The graph ledger and the modulus chain first build the canonical prefixed continuous-function carrier with target $p \cdot target$ and certificate $p \cdot cert$. Carrier target-certificate determinacy compares this canonical carrier with the displayed carrier, giving equality of the two prefixed targets and equality of the two prefixed certificates. Left cancellation of the shared prefix p yields the displayed public readback equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain public readback`

Theorem 58.87 (Prefixed continuous function carrier graph-chain second determinacy)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta_2', \delta, \delta', cert$, assume δ_1, δ_2, and δ_2' are unary histories, with continuation ledgers*

$$\mathsf{Cont}(\delta_1, \delta_2, \delta), \qquad \mathsf{Cont}(\delta_1, \delta_2', \delta').$$

If the two composite moduli support the same prefixed continuous function carrier,

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta', p \cdot cert),$$

then the second modulus components are history-same:

$$\mathsf{hsame}(\delta_2, \delta_2').$$

Proof. Unpack the two continuous function carriers and compare their terminal certificate ledgers. Since both ledgers start at $p \cdot target$ and end at $p \cdot cert$, continuation left cancellation identifies the composite moduli δ and δ'. Transport to a common composite modulus, then apply left cancellation to the two ledgers out of δ_1 to obtain $\mathsf{hsame}(\delta_2, \delta_2')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain second deterministic`

Theorem 58.88 (Prefixed continuous function carrier graph-chain first determinacy)**.** *For histories $p, source, map, target, \delta_1, \delta_1', \delta_2, \delta, \delta', cert$, assume δ_1, δ_1', and δ_2 are unary histories, with continuation ledgers*

$$\mathsf{Cont}(\delta_1, \delta_2, \delta), \qquad \mathsf{Cont}(\delta_1', \delta_2, \delta').$$

If the two composite moduli support the same prefixed continuous function carrier,

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta', p \cdot cert),$$

then the first modulus components are history-same:

$$\mathsf{hsame}(\delta_1, \delta_1').$$

Proof. Unpack the two continuous function carriers and compare their terminal certificate ledgers. Since both ledgers start at $p \cdot target$ and end at $p \cdot cert$, continuation left cancellation identifies the composite moduli δ and δ'. Common-suffix cancellation on the two composite modulus ledgers, using the shared suffix δ_2, then yields $\mathsf{hsame}(\delta_1, \delta_1')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain first deterministic`

Theorem 58.89 (Prefixed continuous modulus composite closure)**.** *For unary histories $p, s, \delta_1, \delta_2, t, \delta$, a unary prefix p, a two-link modulus chain from s through δ_1 and δ_2 to t, and a continuation ledger composing δ_1 and δ_2 into δ determine a prefixed continuous modulus witness:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow$$
$$\mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t).$$

Proof. Unpack the chain into unary endpoint and modulus histories, together with the continuation ledgers from s to the middle history and from the middle history to t.

The prefixed source and target histories are unary by unary append closure applied to p with s and t. The composite modulus history is unary by unary continuation closure applied to the two modulus histories and their composition ledger.

The remaining continuation ledger is obtained by associating the concatenation spine:

$$(p \cdot s) \cdot \delta = p \cdot (s \cdot \delta) = p \cdot t.$$

Thus the prefixed endpoints and the same composite modulus satisfy the public continuous modulus witness predicate. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite closed`

Theorem 58.90 (Continuous modulus witness prefixed composite factorization)**.** *For histories $p, s, \delta_1, \delta_2, \delta, t$, if δ_1 and δ_2 are unary histories, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t)$$

holds, then there is a middle history m such that

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \wedge \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Proof. First remove the common prefix from the displayed continuous modulus witness using Theorem 59.21. This leaves a witness from s through the composite modulus δ to t.

Choose $m = s \cdot \delta_1$. The first witness is immediate from $\mathsf{Cont}(s, \delta_1, s \cdot \delta_1)$ and unary append closure. For the second witness, case-analyze the two continuation ledgers $\mathsf{Cont}(s, \delta, t)$ and $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. Associativity identifies $(s \cdot \delta_1) \cdot \delta_2$ with $s \cdot (\delta_1 \cdot \delta_2)$, hence with t. This supplies the required continuation ledger from m through δ_2 to t. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite factorizes`

Theorem 58.91 (Continuous prefixed composite middle deterministic factorization)**.** *For histories $p, s, \delta_1, \delta_2, \delta, t$, if δ_1 and δ_2 are unary histories, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t)$$

holds, then there is a middle history m such that

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \wedge \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t),$$

and every other middle history carrying the same two witness legs is hsame *to m.*

Proof. First apply the prefixed composite factorization theorem to expose a middle history and the two one-step continuous modulus witnesses. If another middle history carries a witness from s through δ_1, both first legs have the same source and modulus; one-step target determinacy therefore identifies the two middle histories by hsame. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite middle deterministic factorization`

Theorem 58.92 (Continuous prefixed composite first deterministic)**.** *For histories $p, s, \delta_1, \delta_1', \delta_2, \delta, \delta', t$, assume δ_1, δ_1', and δ_2 are unary, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ and $\mathsf{Cont}(\delta_1', \delta_2, \delta')$ hold, and both*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s, \delta', p \cdot t)$$

hold. Then $\mathsf{hsame}(\delta_1, \delta_1')$.

Proof. Factor both prefixed composite witnesses. The second legs share the same target t and the same second modulus δ_2, so one-step source determinacy identifies their middle histories. The first legs then share the same source s and hsame target histories, so one-step modulus determinacy identifies δ_1 and δ_1'. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite first deterministic`

Theorem 58.93 (Continuous modulus witness prefixed composite source determinacy)**.** *For histories $p, s, s', \delta_1, \delta_2, \delta, t$, assume δ_1 and δ_2 are unary histories and $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. If two prefixed composite modulus witnesses share the same composite modulus and the same prefixed target,*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s', \delta, p \cdot t),$$

then their unprefixed sources are history-same:

$$\mathsf{hsame}(s, s').$$

Proof. Factor both prefixed composite witnesses into two one-step modulus witnesses. The second legs share the modulus δ_2 and target t, so one-step source determinacy identifies the two recovered middle histories. The first legs then share the modulus δ_1 and have history-same middle targets, so one-step source determinacy gives $\mathsf{hsame}(s, s')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite source deterministic`

Theorem 58.94 (Continuous prefixed composite second deterministic)**.** *For histories $p, s, \delta_1, \delta_2, \delta_2', \delta, \delta', t$, assume δ_1, δ_2, and δ_2' are unary, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ and $\mathsf{Cont}(\delta_1, \delta_2', \delta')$ hold, and both*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s, \delta', p \cdot t)$$

hold. Then $\mathsf{hsame}(\delta_2, \delta_2')$.

Proof. Factor both prefixed composite witnesses. The first legs share the same source s and first modulus δ_1, so one-step target determinacy identifies the two middle histories. The second legs then have hsame source histories and the same target t, so one-step modulus determinacy identifies δ_2 and δ_2'. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite second deterministic`

Theorem 58.95 (Continuous modulus chain prefix closure)**.** *For histories $p, s, \delta_1, \delta_2, t$, a unary prefix on the source and target preserves a two-link continuous modulus chain:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{ContinuousModulusChain}(p{\cdot}s, \delta_1, \delta_2, p{\cdot}t).$$

Proof. Unpack the chain into unary endpoint data and a middle history m with continuations $m = s \cdot \delta_1$ and $t = m \cdot \delta_2$. Use $p \cdot m$ as the middle history for the prefixed chain. Unary append closure supplies the prefixed endpoints, and continuation associativity prefixes both ledger equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix closed`

Theorem 58.96 (Continuous modulus chain prefix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, prefixing both endpoints of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(p{\cdot}s, \delta_1, \delta_2, p{\cdot}t) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

The traditional target is prefix-stable uniform-modulus data. The BEDC source is BHist with Cont*, the carrier is* $\mathsf{ContinuousModulusChain}$*, and the classifier is* hsame*. The stability obligation is prefix exactness. The ledger records source, two precision histories, middle, and target continuations. The core theorem is this exactness statement, the exactness predicate is the displayed iff, the standard bridge is deferred, and the status is* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, unpack the prefixed chain. Unary append factorization gives unary p and unary s. Let the unprefixed middle be $s \cdot \delta_1$.

The first continuation ledger, together with associativity, identifies the prefixed middle as $p \cdot (s \cdot \delta_1)$. The second continuation ledger then has a common visible prefix p on both sides. Associativity and left cancellation recover the unprefixed second continuation from $s \cdot \delta_1$ through δ_2 to t. These data assemble $\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t)$, paired with unary p.

For the reverse direction, take unary p together with the unprefixed chain and apply Theorem 59.119. This gives the prefixed chain from $p \cdot s$ through the same two precision histories to $p \cdot t$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix iff`

Theorem 58.97 (Continuous modulus chain suffix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, suffixing the second modulus and the target of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2 \cdot p, t \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

Proof. For the forward direction, unpack the suffixed chain and derive the unary middle history from the first continuation. The second continuation is a continuous modulus witness from the middle history through $\delta_2 \cdot p$ to $t \cdot p$. Apply Theorem 59.22 to remove the common right suffix from that second leg, then rebuild the original two-link chain.

For the reverse direction, apply Theorem 59.22 to the second leg of the chain and reuse the same middle history and first continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain suffix iff`

Definition 58.98 (ContinuousMap ledger policy)**.** *Record the graph relation, source and target metric ledgers, the totally bounded subset witness, the unary precision ledger, and the modulus ledger.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier ledger policy fields`

Definition 58.99 ($\mathsf{ContinuousMap}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{ContinuousMap}^{\uparrow}$.

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier certificate semanticNameCert`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousUp StdBridge`

Primitive boundary. $\mathsf{ContinuousMap}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 58.100 (Zero-axiom encoding note)**.** *The encoding avoids extensional equality of maps and host* `Nat` *by using graph classifier sameness and unary precision histories. The modulus object is a certified relation between precision histories, so no host-level numeric modulus is exposed at the certificate boundary.*

Proposition 58.101 (Uniform modulus restriction along subset containment)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, and let $i : U \subseteq T$ be a certified inclusion. For a graph witness $f : T \to Z$, let $f \restriction_U$ be the same graph relation read with the source classifier restricted along i. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f over T, then the same δ satisfies that modulus-validity field for $f \restriction_U$ over U.*

Proof. Fix a target precision request and two source histories admitted by U. The certified inclusion sends their source-membership ledgers to membership ledgers for T:

$$u_j \in U \Longrightarrow i(\nu_j) \in T \qquad (j = 0, 1).$$

The graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger are the same ledgers as in the witness over T, as specified by Definition 59.93 and Definition 59.122.

Apply the modulus-validity field over T to the two transported source memberships. It returns the codomain metric conclusion at the requested target precision for the two graph outputs. The metric judgments are well formed and stable under classifier replacement by Definition 57.4 and Definition 57.5.

The restricted witness over U asks for exactly the same codomain conclusion, with only the source-admission hypotheses narrowed from T to U. The totality and single-valuedness clauses from Definition 59.1 license the same graph outputs, while no precision or modulus ledger is changed. □

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public readback`

Closure status for $\mathsf{NameCert}_{\mathsf{Continuous}\uparrow}$.

Constructive story: A continuous map is carried by a graph-and-modulus history packet: source, target, map code, modulus, and certificate histories are unary, the graph ledger continues the source through the map to the target, and the modulus ledger continues the target through the modulus to the certificate. Two such witnesses are classified by the graph outputs agreeing under the target classifier for classified source inputs and by their unary modulus ledgers mutually refining each requested precision; on the checked carrier rows this is enforced through hsame determinacy of the displayed continuation fields. The source consists of certified metric spaces and the totally bounded source discipline, the pattern records graph-with-modulus formation, identity, composition, prefix and restriction examples, terminal-modulus extension, and modulus-chain exactness, and the ledger keeps the graph relation, metric ledgers, precision history, modulus history, and certificate history visible. The five name-certificate obligations close because the identity map uses the empty map code with a one-step modulus witness, hsame supplies reflexive, symmetric, and transitive comparison of the displayed histories, and carrier transport follows by moving each continuation ledger across history-sameness. A carrier row lies in the signature gap of a $\mathsf{Continuous}^{\uparrow}$ token p when the source-domain accepts the graph-and-modulus packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, the graph sends each accepted source history to the encoded target history and the modulus ledger supplies the precision conversion recorded by p. The checked standard bridge reads the same relation-plus-ledger data as a continuous-map boundary and reads it back without host-level function extensionality or a numeric modulus object. The chapter registers this graph-modulus construction under the name $\mathsf{Continuous}^{\uparrow}$; continuity is an explicit BEDC certificate surface, not a host function space, completeness theorem, or external map equality principle.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Continuous}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope binds the ContinuousFunctionCarrier semantic certificate, modulus-witness determinacy and exactness rows, identity, composition, prefix, and restriction examples, and the ContinuousMapUp consumer surface cited by the chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Continuous}\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier\_certificate\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: The chapter does not close host functions, host function spaces, or standard-model completeness.

Upgrade path: no further upgrade – chapter at top of formal axis

Theorem 58.102 (Continuous modulus chain prefix closure)**.** *For histories $p, s, \delta_1, \delta_2, t$, a unary prefix on the source and target preserves a two-link continuous modulus chain:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{ContinuousModulusChain}(p{\cdot}s, \delta_1, \delta_2, p{\cdot}t).$$

Proof. Unpack the chain into unary endpoint data and a middle history m with continuations $m = s \cdot \delta_1$ and $t = m \cdot \delta_2$. Use $p \cdot m$ as the middle history for the prefixed chain. Unary append closure supplies the prefixed endpoints, and continuation associativity prefixes both ledger equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix closed`

Theorem 58.103 (Continuous modulus chain prefix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, prefixing both endpoints of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(p{\cdot}s, \delta_1, \delta_2, p{\cdot}t) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

The traditional target is prefix-stable uniform-modulus data. The BEDC source is BHist with Cont*, the carrier is* ContinuousModulusChain*, and the classifier is* hsame*. The stability obligation is prefix exactness. The ledger records source, two precision histories, middle, and target continuations. The core theorem is this exactness statement, the exactness predicate is the displayed iff, the standard bridge is deferred, and the status is* ProofTarget*.*

Proof. For the forward direction, unpack the prefixed chain. Unary append factorization gives unary p and unary s. Let the unprefixed middle be $s \cdot \delta_1$.

The first continuation ledger, together with associativity, identifies the prefixed middle as $p \cdot (s \cdot \delta_1)$. The second continuation ledger then has a common visible prefix p on both sides. Associativity and left cancellation recover the unprefixed second continuation from $s \cdot \delta_1$ through δ_2 to t. These data assemble $\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t)$, paired with unary p.

For the reverse direction, take unary p together with the unprefixed chain and apply Theorem 59.119. This gives the prefixed chain from $p \cdot s$ through the same two precision histories to $p \cdot t$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix iff`

Theorem 58.104 (Continuous modulus chain suffix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, suffixing the second modulus and the target of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2 \cdot p, t \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

Proof. For the forward direction, unpack the suffixed chain and derive the unary middle history from the first continuation. The second continuation is a continuous modulus witness from the middle history through $\delta_2 \cdot p$ to $t \cdot p$. Apply Theorem 59.22 to remove the common right suffix from that second leg, then rebuild the original two-link chain.

For the reverse direction, apply Theorem 59.22 to the second leg of the chain and reuse the same middle history and first continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain suffix iff`

Proposition 58.105 (Uniform modulus restriction along subset containment)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, and let $i : U \subseteq T$ be a certified inclusion. For a graph witness $f : T \to Z$, let $f \restriction_U$ be the same graph relation read with the source classifier restricted along i. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f over T, then the same δ satisfies that modulus-validity field for $f \restriction_U$ over U.*

Proof. Fix a target precision request and two source histories admitted by U. The certified inclusion sends their source-membership ledgers to membership ledgers for T:

$$u_j \in U \Longrightarrow i(\nu_j) \in T \qquad (j = 0, 1).$$

The graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger are the same ledgers as in the witness over T, as specified by Definition 59.93 and Definition 59.122. Apply the modulus-validity field over T to the two transported source memberships. It returns the codomain metric conclusion at the requested target precision for the two graph outputs. The metric judgments are well formed and stable under classifier replacement by Definition 57.4 and Definition 57.5.

The restricted witness over U asks for exactly the same codomain conclusion, with only the source-admission hypotheses narrowed from T to U. The totality and single-valuedness clauses from Definition 59.1 license the same graph outputs, while no precision or modulus ledger is changed. □

↪ variant BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public readback

Proposition 58.106 (Uniform continuity modulus restriction along subset inclusion)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, and let*

$$i : U \subseteq T$$

be a certified inclusion, meaning that each membership ledger $\nu : U(u)$ is transported to a membership ledger $i(\nu) : T(u)$. Let $f : T \to Z$ be a certified graph witness with totality and single-valuedness as in Definition 59.1, and let $f \restriction_U$ be the same graph relation read with the source classifier restricted through i. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f on T, then the same δ satisfies the modulus-validity field for $f \restriction_U$ on U.

Proof. Fix a target precision request n and the source precision returned by the unary ledger δ. To verify the restricted modulus field, take source histories u_0, u_1 admitted by U, graph outputs z_0, z_1 for $f \restriction_U$, and the source closeness judgment required at the returned source precision.

The inclusion certificate transports the two source-admission ledgers:

$$\nu_j : U(u_j) \Longrightarrow i(\nu_j) : T(u_j) \qquad (j = 0, 1).$$

By Definition 59.93 and Definition 59.122, restricting the source subset does not alter the graph relation, the codomain classifier ledger, the precision ledger, or the modulus ledger. Thus the graph witnesses for $f \restriction_U$ are the same graph witnesses for f after the source admissions are read in T.

Apply the modulus-validity field for f on T to the transported memberships, the same source closeness judgment, and the same two graph outputs. It yields the target metric closeness conclusion at precision n. The well-formedness and representative stability of this conclusion are exactly the classifier and stability data of Definition 57.4 and Definition 57.5.

This is the entire obligation for the restricted witness: only the source-admission hypotheses have been narrowed from T to U, and those hypotheses enter the original modulus field through the certified inclusion. The precision ledger δ is unchanged. □

Lean4 ✓ BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction modulus chain

Theorem 58.107 (Continuous empty-source restriction public package)**.** *Let restricted, source, map, target, target′,* BHist *be histories. Assume*

$$\mathsf{Cont}(\mathit{restricted}, \mathsf{e}, \mathit{source}),$$

$$\mathsf{ContinuousFunctionCarrier}(\mathit{source}, \mathit{map}, \mathit{target}, \mathit{oldMod}, \mathit{oldCert}),$$

$$\mathsf{ContinuousModulusChain}(\mathit{target}, \delta_1, \delta_2, \mathit{cert}),$$

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

and a displayed restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}', \delta, \mathit{cert}').$$

Then the empty-source restriction has the public package

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert}),$$

$$\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert}),$$

and

$$\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}').$$

Proof. Apply the preceding checked empty-source restriction-chain theorem to the source restriction, the original carrier, the modulus chain, and the composite modulus relation. This produces the restricted ContinuousFunctionCarrier over $(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert})$ and the ContinuousModulusWitness over $(\mathit{target}, \delta, \mathit{cert})$.

It remains to read back the displayed public target and certificate. The Lean-side theorem BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier_empty_source_restriction_public_readback applies to the same hypotheses together with the displayed restricted carrier, and returns $\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$.

Ten-field fit: traditional target restriction of uniformly continuous maps along an empty-source inclusion; BEDC source BHist, Cont, ContinuousFunctionCarrier, and ContinuousModulusChain; carrier the restricted ContinuousFunctionCarrier; classifier hsame on target and certificate histories; stability hsame transport and modulus-chain composition; ledger unchanged graph with replaced modulus/certificate chain; core theorem this public package; exactness the target/certificate readback; standard bridge ordinary restricted-map readback; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public readback`

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public package`

Theorem 58.108 (Uniform continuity modulus restriction is transitive)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let V, U, and T be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusions*

$$j : V \subseteq U$$

and

$$i : U \subseteq T.$$

For a certified graph witness $f : T \to Z$, let $f \restriction_U$ be the restriction along i, let $(f \restriction_U) \restriction_V$ be the successive restriction along j, and let $f \restriction_V$ be the direct restriction along the composite inclusion $i \circ j : V \subseteq T$. The composite inclusion carries the same history-level graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger as the successive restriction. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f over T, then δ satisfies that field for both $f \restriction_V$ and $(f \restriction_U) \restriction_V$, and the two restricted witnesses are related by the $\mathsf{ContinuousMap}^{\uparrow}$ classifier of Definition 59.94.

Proof. Apply Proposition 59.80 first to $i : U \subseteq T$. This transports the modulus-validity field for f over T to the restriction $f \restriction_U$ over U, with the graph relation and all precision data left unchanged.

Apply the same proposition to $j : V \subseteq U$ and the already restricted witness $f \restriction_U$. The result is the modulus-validity field for $(f \restriction_U) \restriction_V$ over V, with the same history-level graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger.

Apply Proposition 59.80 once more to the composite inclusion $i \circ j : V \subseteq T$. This gives the direct restricted witness $f \restriction_V$ and its modulus-validity field over V using the original witness f and the same unary precision ledger δ.

It remains to compare the direct and successive restrictions. By Definition 59.93, restriction changes only the source-admission reading of the graph; by Definition 59.122, the graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger are recorded as the same ledgers. Thus the direct and successive witnesses present the same carrier data to the classifier, and Definition 59.94 relates them as $\mathsf{ContinuousMap}^{\uparrow}$ witnesses.

Ten-field fit: traditional target functorial restriction of uniformly continuous maps; BEDC source BHist with certified source predicates and ContinuousFunctionCarrier; carrier restricted graph witness; classifier ContinuousMap classifier; stability nested source-admission transport; ledger unchanged graph/modulus ledgers; core theorem this block; exactness direct-versus-successive restriction agreement; standard bridge restriction functoriality for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction transitive package`

Theorem 58.109 (Uniform continuity restriction functorial action)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T, U, and V be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92. Let*

$$e_T : T \subseteq T$$

be the identity inclusion, and let

$$i : U \subseteq T$$

and

$$j : V \subseteq U$$

be certified inclusions. Let $f : T \to Z$ be a certified graph witness whose unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 over T. Then restriction along e_T preserves δ and is related to f by the $\mathsf{ContinuousMap}^{\uparrow}$ classifier of Definition 59.94. Moreover, the successive restriction $(f \restriction_U) \restriction_V$ and the direct restriction along the certified composite inclusion $i \circ j : V \subseteq T$ are related by the same classifier and preserve the same graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger.

Proof. For the identity clause, apply Proposition 59.80 to the certified inclusion $e_T : T \subseteq T$. The source-admission transport is the identity transport on the T ledger, so the restricted witness reads the same histories as f.

The unchanged ledger clauses of Definition 59.93 and Definition 59.122 keep the graph relation, codomain classifier ledger, unary precision ledger δ, and modulus ledger fixed. Hence the two witnesses present the same carrier data, and Definition 59.94 relates the identity restriction to f as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

For the composition clause, apply Theorem 59.82 to the inclusions $j : V \subseteq U$ and $i : U \subseteq T$. It gives the modulus-validity field for both the successive and direct restrictions with the same unary precision ledger δ.

The same theorem also identifies the direct and successive restrictions at the level of the $\mathsf{ContinuousMap}^{\uparrow}$ classifier. Its comparison uses exactly the shared graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger supplied by the restriction policy.

Ten-field fit: traditional target restriction action of uniformly continuous maps; BEDC source BHist source predicates plus ContinuousFunctionCarrier; carrier restricted graph witness; classifier ContinuousMapUp; stability identity and composite source-admission transport; ledger unchanged graph/modulus data; core theorem this action law; exactness identity and composition agreement; standard bridge restriction of uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction functorial action`

Theorem 58.110 (Uniform continuity restriction commutes with terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$source, map, target, modulus, cert, extra, modulus', cert' : \mathsf{BHist}$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the original unary modulus, the terminal certificate, the additional terminal modulus, the composite modulus, and the terminal certificate after composition. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert),\\&\mathsf{ContinuousModulusWitness}(cert, extra, cert'),\\&\mathsf{Cont}(modulus, extra, modulus').\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting f along i and then applying terminal modulus extension with extra, $modulus'$, and $cert'$ is related by the classifier of Definition 59.94 to the witness obtained by first applying terminal modulus extension over T and then restricting along i. Both witnesses present the restricted graph $f\restriction_U$, the source-admission transport through i, the codomain classifier ledger for Z, the composite modulus ledger $modulus'$, and the terminal certificate ledger $cert'$.

Proof. Apply Theorem 59.162 to the displayed carrier, the terminal modulus witness, and the composite continuation ledger. This gives a continuous-function carrier over T with histories $source, map, target, modulus', cert'$.

Restrict this extended carrier along $i : U \subseteq T$. Proposition 59.80 transports the modulus-validity field to U, while Definition 59.93 and Definition 59.122 keep the graph relation, target classifier, precision ledger, and modulus ledger fixed.

For the other route, first apply Proposition 59.80 to the original witness f. The restricted witness over U carries the same graph ledger map, the same target ledger $target$, and the same modulus and certificate ledgers $modulus, cert$, with only the source-admission hypotheses narrowed through i.

Now apply Theorem 59.162 to this restricted carrier using the same $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$ and the same $\mathsf{Cont}(modulus, extra, modulus')$. The resulting restricted-then-extended witness has composite modulus $modulus'$ and terminal certificate $cert'$.

It remains to compare the two route results. Their graph relation is the same relation read on U, source histories admitted by U are transported to T by the same inclusion certificate, the codomain classifier ledger is the same Z ledger, and the composite modulus and terminal certificate

ledgers are the same displayed histories. Therefore Definition 59.94 relates the two witnesses as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

Ten-field fit: traditional target commutation of source restriction with terminal modulus extension; BEDC source BHist with certified source predicates, Cont, and ContinuousModulusWitness; carrier ContinuousFunctionCarrier with restricted graph witness; classifier ContinuousMap classifier; stability source-admission transport plus terminal modulus preservation; ledger unchanged graph relation, codomain classifier ledger, composite modulus ledger, and terminal certificate ledger; core theorem this block; exactness equality of the two route witnesses under Definition 59.94; standard bridge restriction and modulus-normalization order independence for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction terminal modulus package`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction terminal modulus commutes`

Theorem 58.111 (Uniform continuity restriction commutes with iterated terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$source, map, target, modulus, cert, extra_0, modulus_0, cert_0, extra_1, modulus_1, cert_1 : \mathsf{BHist}$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the initial unary modulus, the initial terminal certificate, the first added terminal modulus, the first composite modulus, the first terminal certificate after composition, the second added terminal modulus, the second composite modulus, and the final terminal certificate. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert),\\
&\mathsf{ContinuousModulusWitness}(cert, extra_0, cert_0), \quad \mathsf{Cont}(modulus, extra_0, modulus_0),\\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{Cont}(modulus_0, extra_1, modulus_1).
\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting f along i and then applying the two terminal modulus extensions with final histories $modulus_1, cert_1$ is related by the classifier of Definition 59.94 to the witness obtained by applying the two terminal modulus extensions over T and then restricting along i. Both routes expose the restricted graph ledger map, the transported source-admission ledger through i, the same target classifier ledger $target$, and final histories $modulus_1, cert_1$.

Proof. Apply Theorem 59.84 to the initial carrier, the witness $\mathsf{ContinuousModulusWitness}(cert, extra_0, cert_0)$, and the continuation ledger $\mathsf{Cont}(modulus, extra_0, modulus_0)$. This relates the route that restricts first and then performs the first terminal extension to the route that performs the first terminal extension over T and then restricts. Both resulting representatives carry restricted graph ledger map, transported source-admission through i, target ledger $target$, and histories $modulus_0, cert_0$.

Apply Theorem 59.162 to the initial carrier and the first terminal data. It keeps the carrier over T live after the first extension, now with histories $source, map, target, modulus_0, cert_0$. Applying the same theorem to the restricted representative supplied by the first commutation keeps the restricted carrier live with the same graph and target ledgers and with histories $modulus_0, cert_0$.

Now apply Theorem 59.84 to these first-extended carriers, using $\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1)$ and $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$. This relates the route that restricts before the second terminal extension to the route that performs the second terminal extension over T before restriction. The two representatives therefore expose the same restricted graph ledger, source-admission transport, target classifier ledger, and final histories $modulus_1, cert_1$.

The first commutation gives a classifier relation between the two intermediate representatives, and the second commutation gives a classifier relation between the two final representatives built from those intermediates. By the transitivity clause in Definition 59.94, the endpoints are related as $\mathsf{ContinuousMap}^{\uparrow}$ witnesses: restricting first and applying both terminal extensions is classified together with extending twice over T and then restricting to U.

Ten-field fit: traditional target finite terminal-modulus normalization under source restriction; BEDC source BHist with certified source predicates, Cont, and two $\mathsf{ContinuousModulusWitness}$ ledgers; carrier $\mathsf{ContinuousFunctionCarrier}$ preserved through two terminal extensions; classifier ContinuousMap classifier with transitivity; stability transported source admission plus preserved modulus-validity fields; ledger restricted graph ledger, target classifier ledger, and final histories $modulus_1, cert_1$; core theorem this block; exactness equality of the two finite route endpoints under Definition 59.94; standard bridge finite modulus-normalization stability for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction iterated terminal modulus commutes`

Theorem 58.112 (Uniform continuity restriction commutes with visible terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let*

$$p, q, source, map, target, modulus, cert, extra, modulus', cert' : \mathsf{BHist}$$

be histories, and assume the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q),$$

the terminal modulus witness

$$\mathsf{ContinuousModulusWitness}(cert, extra, cert'),$$

and the terminal modulus composition ledger

$$\mathsf{Cont}(modulus, extra, modulus').$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting the visible carrier along i and then applying visible terminal modulus extension with $extra$, $modulus'$, and $cert'$ is related by the classifier of Definition 59.94 to the witness obtained by first applying visible terminal modulus extension and then restricting along i. Both routes expose the restricted graph ledger map, the same source-admission transport through i, the visible target ledger $p \cdot target$, the composite modulus ledger $modulus' \cdot q$, and the terminal certificate ledger $(p \cdot cert') \cdot q$.

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction visible terminal modulus commutes`

Proof. Peel the visible carrier by Theorem 59.169. This recovers unary context histories for p and q together with the central carrier

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).$$

Apply Theorem 59.84 to this central carrier, the source inclusion $i : U \subseteq T$, the witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$, and the ledger $\mathsf{Cont}(modulus, extra, modulus')$. The central endpoints are classified together: one restricts first and then extends the terminal modulus, and the other extends over T first and then restricts.

Reattach the visible prefix and suffix to the two central endpoints. Theorem 59.163 supplies the visible terminal extension on the route whose extension is performed after the visible carrier is restricted, and the reverse use of Theorem 59.169 repacks the same unary histories around the endpoint obtained by extending first.

The visible-context exactness theorem identifies the exposed public ledgers on both routes. The graph ledger is the restricted reading of *map*, the source-admission hypotheses are transported through the same inclusion certificate i, the target ledger is $p \cdot target$, and the final ledgers are $modulus' \cdot q$ and $(p \cdot cert') \cdot q$. Therefore the classifier of Definition 59.94 relates the two visible witnesses as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

Ten-field fit: traditional target commutation of source restriction with visible terminal modulus extension; BEDC source BHist with visible contexts, certified source predicates, Cont, and ContinuousModulusWitness; carrier visible ContinuousFunctionCarrier with restricted graph witness; classifier ContinuousMap classifier; stability source-admission transport plus visible terminal modulus preservation; ledger restricted graph relation, visible target ledger, composite modulus ledger, and visible terminal certificate ledger; core theorem this block; exactness equality of the two visible route endpoints under Definition 59.94; standard bridge restriction and visible modulus-normalization order independence for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus commutes`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible empty source restriction terminal modulus commutes`

Theorem 58.113 (Uniform continuity visible restriction commutes with modulus chains)**.** *Let*

$$p, q, restricted, source, map, target, target', oldMod, oldCert, \delta_1, \delta_2, \delta, cert, cert' : \mathsf{BHist}$$

be histories. Assume the source restriction ledger

$$\mathsf{Cont}(restricted, \mathsf{e}, source),$$

the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, oldMod \cdot q, (p \cdot oldCert) \cdot q),$$

the two-link modulus chain

$$\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert),$$

the composite modulus ledger

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

and a displayed restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}', \delta \cdot q, (p \cdot \mathit{cert}') \cdot q).$$

Then the canonical visible restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}, \delta \cdot q, (p \cdot \mathit{cert}) \cdot q),$$

the central witness

$$\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert}),$$

and the readback

$$\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$$

hold.

Proof. Peel the displayed visible carrier by Theorem 59.169. This exposes the central carrier over $(\mathit{source}, \mathit{map}, \mathit{target}, \mathit{oldMod}, \mathit{oldCert})$ while retaining the same visible prefix p and suffix q.

Apply the checked central chain theorem recorded above, namely BEDC.Derived.ContinuousUp.ContinuousFunctio to the exposed carrier, the source restriction ledger, the displayed modulus chain, and the composite ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. It yields the central restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert})$$

and the witness $\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert})$.

Reattach the visible prefix and suffix with the reverse direction of Theorem 59.169. The resulting public carrier is exactly

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}, \delta \cdot q, (p \cdot \mathit{cert}) \cdot q).$$

It remains to compare this canonical carrier with the displayed restricted carrier. The visible-context target-certificate determinacy theorem applies to the two carriers with the same visible source, graph, modulus, prefix, and suffix data, so it returns $\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$.

Ten-field fit: traditional target commutation of visible source restriction with finite modulus-chain normalization; BEDC source BHist with Cont, visible contexts, and ContinuousModulusChain; carrier visible ContinuousFunctionCarrier with restricted source; classifier hsame on target and certificate histories; stability central chain witness plus visible-context transport; ledger composite modulus δ with unchanged graph ledger; core theorem this visible chain replacement law; exactness target-certificate readback against any displayed restricted carrier; standard bridge visible uniformly continuous restriction under modulus-chain normalization; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction modulus chain commutes`

Theorem 58.114 (Uniform continuity restriction commutes with three-step terminal modulus reassociation)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$\mathit{source}, \mathit{map}, \mathit{target}, \mathit{modulus}_0, \mathit{cert}_0, \mathit{extra}_1, \mathit{modulus}_1, \mathit{cert}_1, \mathit{extra}_2, \mathit{modulus}_2, \mathit{cert}_2, \mathit{extra}_3, \mathit{modulus}_3, \mathit{cert}_3, \mathit{ext}$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the initial unary modulus and terminal certificate, the three terminal additions with their sequential output moduli and certificates, and the binary and ternary composite terminal ledgers. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus_0, cert_0),\\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2), \quad \mathsf{ContinuousM}\\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),\\
&\mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),\\
&\mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}).
\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ *witness obtained by restricting f along i and applying the three terminal modulus extensions with final public histories $map, target, modulus_3, cert_3$ is related by the classifier of Definition 59.94 to the witness obtained by first reassociating the three terminal extensions over T and then restricting along i with final public histories $map, target, modulus_{123}, cert_3$. The comparison includes the terminal-modulus equality* $\mathsf{hsame}(modulus_3, modulus_{123})$*, and both classified endpoints expose the restricted graph ledger map, the source-admission transport through i, the same target ledger target, and the same terminal certificate ledger $cert_3$.*

Proof. First apply Theorem 59.85 to the initial carrier and the first two terminal extensions. This relates the route that restricts before the first two extensions to the route that performs those two extensions over T before restriction. Both intermediate representatives expose the restricted graph ledger *map*, the transported source-admission through i, the target ledger *target*, and terminal data $modulus_2, cert_2$.

Next keep the third extension live on both sides. Apply Theorem 59.84 to the two-step carrier over T using $\mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3)$ and $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$. This classifies the endpoint obtained by restricting first and then applying all three terminal extensions with the endpoint obtained by applying the sequential three-step extension over T and then restricting.

Now identify the sequential and reassociated terminal moduli over the unreduced source. Theorem 59.168, with the empty visible prefix and suffix, supplies the carrier with final data $modulus_3, cert_3$, the carrier with final data $modulus_{123}, cert_3$, and the comparison $\mathsf{hsame}(modulus_3, modulus_{123})$.

Restrict the reassociated carrier along $i : U \subseteq T$. Proposition 59.80 transports the modulus-validity field to U, and the ledger policy keeps the graph relation *map*, the target classifier ledger *target*, and the terminal certificate ledger $cert_3$ fixed.

Finally use classifier transitivity in Definition 59.94. The sequential endpoint is classified with the extend-over-T sequential endpoint by the preceding commutation steps, the terminal chapter identifies that endpoint with the reassociated carrier through $\mathsf{hsame}(modulus_3, modulus_{123})$, and restriction of the reassociated carrier gives the displayed final representative. Target and certificate determinacy keep the exposed target and terminal certificate ledgers equal, so the two restricted endpoints are $\mathsf{ContinuousMap}^{\uparrow}$ representatives of the same public certificate.

Ten-field fit: traditional target finite three-step terminal-modulus normalization under source restriction; BEDC source BHist with certified source predicates, Cont, and three ContinuousModulusWitness ledgers; carrier ContinuousFunctionCarrier preserved through sequential and reassociated terminal extensions; classifier ContinuousMap classifier with transitivity; stability transported source admission plus preserved modulus-validity fields; ledger restricted graph ledger, target classifier ledger, final sequential and reassociated modulus ledgers, and terminal certificate ledger; core theorem this block; exactness equality of the two route endpoints plus $\mathsf{hsame}(modulus_3, modulus_{123})$; standard

bridge finite modulus-normalization coherence for uniformly continuous maps; status ProofTarget. □

Theorem 58.115 (Uniform continuity visible restriction commutes with three-step terminal modulus reassociation)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra_3, modulus_3, cert_3$$

be the visible context histories, the source, graph, and target ledgers for f, the initial visible unary modulus and terminal certificate, the three terminal additions with their sequential output moduli and certificates, and the binary and ternary composite terminal ledgers. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q),$$

the three terminal modulus witnesses

$$\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2), \quad \mathsf{ContinuousM}$$

and the sequential and reassociated terminal modulus ledgers

$$\begin{aligned}&\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),\\ &\mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),\\ &\mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}).\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ *witness obtained by restricting the visible carrier along i and applying the three visible terminal modulus extensions with exposed public histories $map, p \cdot target, modulus_3 \cdot q, (p \cdot cert_3) \cdot q$ is related by the classifier of Definition 59.94 to the witness obtained by first reassociating the three terminal extensions over the central carrier and then restricting along i with exposed public histories $map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q$. The comparison includes the central readback* $\mathsf{hsame}(modulus_3, modulus_{123})$*, and both visible endpoints carry the same source-admission transport through i, the same graph ledger map, the same visible target ledger $p \cdot target$, and the same visible terminal certificate ledger $(p \cdot cert_3) \cdot q$.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus reassociation`

Proof. Peel the visible carrier by Theorem 59.169. This exposes the central carrier

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus_0, cert_0)$$

under the same certified source inclusion $i : U \subseteq T$, while retaining the visible prefix p and suffix q for later reattachment.

Apply Theorem 59.88 to the central carrier, the three ContinuousModulusWitness rows, and the sequential and reassociated Cont rows. The central route that restricts first and then applies the three terminal extensions is classified with the route that first reassociates the three terminal extensions over T and then restricts, and the same step records $\mathsf{hsame}(modulus_3, modulus_{123})$.

Reattach the common visible prefix and suffix to the two central endpoints. The visible terminal-extension theorem supplies the route whose terminal extensions are performed after restriction, and the reverse direction of Theorem 59.169 repacks the endpoint obtained from the reassociated central carrier.

The visible-context exactness theorems identify the public ledgers after reattachment. The sequential endpoint exposes $p \cdot target$, $modulus_3 \cdot q$, and $(p \cdot cert_3) \cdot q$, while the reassociated endpoint exposes $p \cdot target$, $modulus_{123} \cdot q$, and the same visible terminal certificate $(p \cdot cert_3) \cdot q$.

Finally use classifier transitivity from Definition 59.94. The peeled central comparison, the two visible reattachments, and the visible exactness readbacks compose into a $\mathsf{ContinuousMap}^{\uparrow}$ classifier relation between the displayed public endpoints, with the central modulus comparison $\mathsf{hsame}(modulus_3, modulus_{123})$ carried along.

Ten-field fit: traditional target visible source restriction under three-step terminal-modulus reassociation; BEDC source BHist with visible contexts, certified source predicates, Cont, and three ContinuousModulusWitness rows; carrier visible ContinuousFunctionCarrier with restricted source admission; classifier ContinuousMap classifier with transitivity; stability visible-context peeling, central restriction reassociation, and visible reattachment; ledger restricted graph ledger, visible target ledger, sequential and reassociated visible modulus ledgers, and visible terminal certificate ledger; core theorem this visible reassociation statement; exactness public readback with $\mathsf{hsame}(modulus_3, modulus_{123})$; standard bridge public visible uniformly continuous restriction under three terminal modulus extensions; status ProofTarget. □

Theorem 58.116 (Uniform continuity visible restriction three-step terminal modulus normal form)**.** *Let*

$$p, q, restricted, source, map, target, target', modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra$$

be histories. Assume the source restriction ledger

$$\mathsf{Cont}(restricted, \mathsf{e}, source),$$

the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q),$$

the three terminal modulus witnesses

$$\begin{gathered}\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1),\\ \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2),\\ \mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3),\end{gathered}$$

the sequential and reassociated terminal modulus ledgers

$$\begin{gathered}\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),\\ \mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),\\ \mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}),\end{gathered}$$

and the displayed restricted sequential carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot restricted, map, p \cdot target', modulus_3 \cdot q, (p \cdot cert') \cdot q).$$

Then the visible restricted carrier has the three-step terminal modulus normal form

$$\mathsf{ContinuousFunctionCarrier}(p \cdot restricted, map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q),$$

and the displayed sequential endpoint reads back to the same public data:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_3, cert') \wedge \mathsf{hsame}(modulus_3, modulus_{123}).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus normal form`

Proof. First apply Theorem 59.168 to the unreduced visible carrier, the three terminal witness rows, and the displayed sequential and reassociated Cont ledgers. It supplies the visible sequential carrier with final data $modulus_3, cert_3$, the visible reassociated carrier with final data $modulus_{123}, cert_3$, and the modulus comparison $\mathsf{hsame}(modulus_3, modulus_{123})$.

Restrict the reassociated carrier along the same source ledger $\mathsf{Cont}(restricted, \mathsf{e}, source)$. Proposition 59.80 transports the source-admission field, while the visible context and ledger policies preserve the graph ledger map, target ledger $p \cdot target$, reassociated modulus ledger $modulus_{123} \cdot q$, and certificate ledger $(p \cdot cert_3) \cdot q$.

Next compare the sequential and reassociated routes through Theorem 59.88, applied to the central histories under the same source restriction and terminal ledgers. This identifies the restricted sequential endpoint with the restricted reassociated endpoint and records the same terminal-modulus readback $\mathsf{hsame}(modulus_3, modulus_{123})$.

Finally apply Theorem 59.174 to the canonical restricted sequential carrier and the displayed restricted sequential carrier. They share the visible source $p \cdot restricted$, the graph ledger map, the visible modulus $modulus_3 \cdot q$, and the same visible prefix and suffix, so determinacy gives $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert_3, cert')$. Combining these with the modulus comparison yields the stated normal-form readback.

Ten-field fit: traditional target visible source restriction followed by three terminal modulus extensions; BEDC source BHist with visible contexts, source restriction ledger, Cont, and three ContinuousModulusWitness rows; carrier visible ContinuousFunctionCarrier with restricted source; classifier hsame target, certificate, and modulus readback; stability source-admission restriction plus terminal modulus reassociation; ledger restricted graph ledger, visible target ledger, sequential and reassociated modulus ledgers, and terminal certificate ledger; core theorem this normal-form statement; exactness comparison against any displayed restricted sequential carrier; standard bridge visible uniformly continuous restriction under three terminal modulus extensions; status ProofTarget. □

Theorem 58.117 (Continuous restriction visible modulus-chain functorial normal form)**.** *Let T, U, V, and W be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusions*

$$k : W \subseteq V, \qquad j : V \subseteq U, \qquad i : U \subseteq T.$$

Let

$$p, q, source_T, source_U, source_V, source_W, map, target, target_d, target_s, oldMod, oldCert, \delta_1, \delta_2, \delta, cert, cert_d, cert_s :$$

be histories. Assume the visible carrier over T

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_T, map, p \cdot target, oldMod \cdot q, (p \cdot oldCert) \cdot q),$$

the source-restriction ledgers

$$\mathsf{Cont}(source_U, \mathsf{e}, source_T), \qquad \mathsf{Cont}(source_V, \mathsf{e}, source_U), \qquad \mathsf{Cont}(source_W, \mathsf{e}, source_V), \qquad \mathsf{Cont}(source_W, \mathsf{e}, s$$

the two-link visible modulus chain

$$\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert),$$

the composite modulus ledger

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

the displayed direct restricted carrier over W

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target_d, \delta \cdot q, (p \cdot cert_d) \cdot q),$$

and the displayed successive restricted carrier over W

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target_s, \delta \cdot q, (p \cdot cert_s) \cdot q).$$

Then the route that restricts directly from T to W and the route that restricts successively

$$T \longrightarrow U \longrightarrow V \longrightarrow W$$

before applying the same visible modulus-chain replacement have the common public normal form

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target, \delta \cdot q, (p \cdot cert) \cdot q),$$

and their displayed target and certificate readbacks agree:

$$\mathsf{hsame}(target, target_d) \wedge \mathsf{hsame}(cert, cert_d) \wedge \mathsf{hsame}(target, target_s) \wedge \mathsf{hsame}(cert, cert_s).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction modulus chain functorial normal form`

Proof. First collapse the nested source restrictions. Apply Theorem 59.83 to $i : U \subseteq T$ and $j : V \subseteq U$ to compare the direct restriction from T to V with the successive route through U.

Apply Theorem 59.83 again to the composite inclusion $V \subseteq T$ and $k : W \subseteq V$. This identifies the direct restriction from T to W with the three-step route through U and V at the $\mathsf{ContinuousMap}^{\uparrow}$ classifier, while preserving the graph ledger map, the visible target ledger $p{\cdot}target$, and the source ledger $p \cdot source_W$.

Next normalize the visible modulus data on the direct representative. Theorem 59.87, applied to the direct source ledger $\mathsf{Cont}(source_W, \mathsf{e}, source_T)$ and the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$, supplies the canonical visible carrier over W with public histories map, $p{\cdot}target$, $\delta{\cdot}q$, and $(p{\cdot}cert){\cdot}q$.

It remains to read back the two displayed endpoints. The direct displayed carrier has the same restricted source, graph ledger, visible modulus ledger, and visible prefix and suffix as the canonical direct carrier, so Theorem 59.87 gives $\mathsf{hsame}(target, target_d)$ and $\mathsf{hsame}(cert, cert_d)$.

Transport the same canonical carrier across the classifier comparison supplied by the functorial restriction step. Target and certificate determinacy for visible modulus contexts then applies to the successive displayed carrier, giving $\mathsf{hsame}(target, target_s)$ and $\mathsf{hsame}(cert, cert_s)$. Combining the four readbacks yields the stated common normal form for the direct and successive routes.

Ten-field fit: traditional target nested restriction of uniformly continuous maps followed by a visible two-link modulus-chain replacement; BEDC source BHist with visible contexts, certified source-restriction ledgers, Cont, and ContinuousModulusChain; carrier visible ContinuousFunctionCarrier over the final source W; classifier ContinuousMap classifier plus hsame target and certificate readback; stability identity and composite source-admission transport with modulus-chain preservation; ledger common graph ledger, visible target ledger, composite modulus ledger, and visible certificate ledger; core theorem this normal form; exactness direct-versus-successive route agreement after the same chain replacement; standard bridge functorial source restriction with visible modulus-chain normalization for uniformly continuous maps; status ProofTarget. □

Chapter 59

A Concrete Naming Certificate for $\mathsf{ContinuousMap}^{\uparrow}$

The continuous-map interface $\mathsf{ContinuousMap}^{\uparrow}$ has certificate obligations between two metric spaces from chapter 57. The carrier is a BEDC graph witness between certified source and target point histories, together with a unary precision-indexed modulus relation. Totality over classifier-admitted source histories and single-valuedness up to the target classifier are certificate fields, so the public object is a relation-plus-ledger certificate rather than a host-level map or host-level numeric modulus.

59.1 Carrier definition

Definition 59.1 (ContinuousMap carrier)**.** *Use a graph relation between source point histories and target point histories, equipped with source and target metric ledgers, a totally bounded source witness, a unary precision ledger, and a modulus ledger relating requested source precision to target precision. The carrier includes totality for every source history accepted by the source classifier and single-valuedness up to the target classifier for any two target histories related to the same classifier-admitted source history.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier`

Definition 59.2 (Continuous modulus witness)**.** *For histories source, modulus, and cert,* ContinuousModulusWitn *records the four components*

$$\mathsf{UnaryHistory}(source), \quad \mathsf{UnaryHistory}(modulus), \quad \mathsf{UnaryHistory}(cert), \quad \mathsf{Cont}(source, modulus, cert).$$

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousModulusWitness`

Definition 59.3 (Continuous modulus chain)**.** *For histories source*, $first$, *second*, $target$, ContinuousModulusChain *consists of unary histories for source,* $first$*, second, and* $target$*, plus a middle history* m *with*

$$\mathsf{Cont}(source, first, m) \quad \textit{and} \quad \mathsf{Cont}(m, second, target).$$

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousModulusChain`

Theorem 59.4 (Continuous modulus witness empty-source exactness)**.** *For histories modulus and* $target$*, a one-step modulus witness whose source is empty is exactly a unary target together with classifier equality between the modulus and the target:*

$$\mathsf{ContinuousModulusWitness}(\mathsf{e}, modulus, target) \iff \mathsf{UnaryHistory}(target) \wedge \mathsf{hsame}(modulus, target).$$

Proof. Unpack the witness. The continuation ledger is $\mathsf{Cont}(\mathsf{e}, modulus, target)$, so the left-unit continuation theorem identifies the target with the modulus. This gives the forward classifier equality after symmetry, while the target unary field is one of the witness fields.

Conversely, transport the target unary field across the classifier equality to obtain a unary modulus. The source is unary by the empty-history constructor, and the left-unit continuation equivalence rebuilds $\mathsf{Cont}(\mathsf{e}, modulus, target)$ from the same classifier equality. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty source iff`

Theorem 59.5 (Continuous modulus witness empty-modulus exactness)**.** *For histories source and target, a one-step modulus witness whose modulus is empty is exactly unary source evidence, unary target evidence, and classifier equality between the source and the target:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{e}, target) \iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(target) \wedge \mathsf{hsame}(source, targ$$

Proof. Unpack the witness. Its continuation ledger is $\mathsf{Cont}(source, \mathsf{e}, target)$, so the right-unit continuation theorem identifies *target* with *source*. Symmetry gives the displayed endpoint classifier, and the source and target unary fields are already present in the witness.

Conversely, use the empty-history unary constructor for the modulus. The supplied endpoint classifier, after symmetry, is exactly the right-unit continuation equivalence needed to rebuild $\mathsf{Cont}(source, \mathsf{e}, target)$. Together with the two unary endpoint fields, this reassembles the witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty modulus iff`

Theorem 59.6 (Continuous modulus witness endpoint cycle modulus empty)**.** *For histories source, modulus, target, a one-step continuous modulus witness whose source and target are history-same has empty modulus:*

$$\mathsf{ContinuousModulusWitness}(source, modulus, target) \Longrightarrow \mathsf{hsame}(source, target) \Longrightarrow \mathsf{hsame}(modulus, \mathsf{e}).$$

Proof. The witness contains the continuation ledger $\mathsf{Cont}(source, modulus, target)$. Transport that ledger across the endpoint cycle to get a self-extension of *source*, and apply right-unit uniqueness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness endpoint cycle modulus empty`

Theorem 59.7 (Continuous function carrier composition closure)**.** *If f carries a source history to a middle history with modulus $modF$, and g carries the middle history to a target history with modulus $modG$, then a continuation composite fg with modulus composite $modFG$ carries the source history to the target history.*

Proof. The two carrier witnesses give unary histories for the source, target, maps, and moduli, plus continuation ledgers for the source-to-middle and middle-to-target maps. Unary continuation closure gives unary histories for the composite map and composite modulus. Case analysis on the continuation constructors reduces the source-to-target continuation to associativity of history append, while the supplied target-to-certificate ledger records the composite modulus certificate. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier comp closed`

Theorem 59.8 (Continuous function carrier composition endpoint hsame transport)**.** *For histories* $source, source', middle, target, target', f, g, fg, fg', modF, modG, modFG, modFG', certF, certG, cert, cert'$*, composition of two continuous-function carriers is stable under endpoint, composite map, composite modulus, and certificate history sameness:*

$$\mathsf{hsame}(source, source') \Longrightarrow \mathsf{hsame}(target, target') \Longrightarrow \mathsf{hsame}(fg, fg') \Longrightarrow \mathsf{hsame}(modFG, modFG') \Longrightarrow \mathsf{hsame}(c$$
$$\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(middle, g, target, modG$$
$$\mathsf{Cont}(f, g, fg) \Longrightarrow \mathsf{Cont}(modF, modG, modFG) \Longrightarrow \mathsf{Cont}(target, modFG, cert) \Longrightarrow$$
$$\mathsf{ContinuousFunctionCarrier}(source', fg', target', modFG', cert').$$

Proof. Composition closure first builds the carrier at the displayed composite graph, composite modulus, target, and certificate. Case analysis on the five hsame witnesses transports those histories to the stated endpoint presentation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier comp endpoint hsame transport`

Theorem 59.9 (Continuous modulus chain composite endpoint hsame transport)**.** *For histories* $source, source', first, second, target, target', composite, composite'$*, endpoint and composite history sameness transports the composite closure witness:*

$$\mathsf{hsame}(source, source') \Longrightarrow \mathsf{hsame}(target, target') \Longrightarrow \mathsf{hsame}(composite, composite') \Longrightarrow$$
$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{Cont}(first, second, composite) \Longrightarrow$$
$$\mathsf{ContinuousModulusWitness}(source', composite', target').$$

Proof. First compose the two-link modulus chain with the displayed continuation ledger. The three hsame witnesses identify the source, target, and composite histories with their transported presentations, so the resulting continuous modulus witness has the transported endpoints. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain composite endpoint hsame transport`

Theorem 59.10 (Continuous modulus chain composite public readback)**.** *For histories* $source, first, second, targe$ *a two-link modulus chain and a public composite modulus ledger give the chain-level readback:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{Cont}(first, second, composite) \Longrightarrow$$
$$\mathsf{ContinuousModulusWitness}(source, composite, target) \wedge$$
$$\forall target'.\ \mathsf{ContinuousModulusWitness}(source, composite, target') \Longrightarrow \mathsf{hsame}(target, target').$$

Proof. Apply Theorem 59.105 to the chain and composite ledger to obtain $\mathsf{ContinuousModulusWitness}(source, comp$

For any competing $target'$, apply Theorem 59.127 to this witness, the competing witness, reflexivity of the shared source, and the shared composite modulus. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain composite public readback`

Theorem 59.11 (Continuous function carrier composition through modulus chain)**.** *If two continuous-function carriers compose through* fg*, and their moduli form a concrete modulus chain over the target certificate, then the composite modulus ledger yields the composite continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(middle, g, target, modG$$
$$\mathsf{Cont}(f, g, fg) \Longrightarrow \mathsf{ContinuousModulusChain}(target, modF, modG, cert) \Longrightarrow \mathsf{Cont}(modF, modG, modFG) \Longrightarrow$$
$$\mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG, cert).$$

Proof. The modulus chain and the displayed modulus continuation produce a concrete modulus witness from *target* through *modFG* to *cert*. Its continuation field is the target-certificate ledger required by continuous-function composition closure. Feeding that ledger together with the two function carriers, the graph continuation, and the modulus continuation gives the composite carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier comp modulus chain closed`

Theorem 59.12 (Continuous function carrier empty-map identity)**.** *For histories source, modulus, and cert, a certified modulus witness over the source ledger determines the identity continuous-function carrier:*

$$\mathsf{ContinuousModulusWitness}(source, modulus, cert) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, \mathsf{e}, source, modulus, cert)$$

Proof. Unpack the modulus witness. It supplies unary histories for *source*, *modulus*, and *cert*, together with the supplied certificate continuation $\mathsf{Cont}(source, modulus, cert)$.

The empty map ledger is unary by the empty-history constructor. The graph part of the identity carrier is the right-unit continuation $\mathsf{Cont}(source, \mathsf{e}, source)$.

Reassemble ContinuousFunctionCarrier from the unary source ledger used for both source and target slots, the unary empty map ledger, the unary modulus ledger, the right-unit graph continuation, and the supplied certificate continuation.

The ten-field certificate fit is the following. The traditional target is the identity uniformly continuous graph. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is identity closure. The ledger consists of source, empty map, modulus, and certificate histories. The core theorem is this identity carrier, exactness is the displayed carrier, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty map identity`

Theorem 59.13 (Continuous function carrier empty-map exactness)**.** *For histories source, target, modulus, and cert, the empty graph map inside* ContinuousFunctionCarrier *is exact:*

$$\mathsf{ContinuousFunctionCarrier}(source, \mathsf{e}, target, modulus, cert) \Longleftrightarrow \mathsf{hsame}(target, source) \wedge \mathsf{ContinuousModulusWitness}$$

Proof. In the forward direction, unpack the carrier. The graph ledger is $\mathsf{Cont}(source, \mathsf{e}, target)$, so the continuation right-unit theorem gives $\mathsf{hsame}(target, source)$. The target-modulus certificate ledger supplies $\mathsf{Cont}(target, modulus, cert)$, and unary continuation closure supplies the unary certificate history. Together with the target and modulus unary fields, these assemble the displayed modulus witness.

Conversely, assume the target-source classifier and the modulus witness over *target*. Transport the target unary ledger across $\mathsf{hsame}(target, source)$ to obtain the source unary ledger. The empty map is unary, the graph ledger is the right-unit continuation read from $\mathsf{hsame}(target, source)$, and the supplied modulus witness gives the target, modulus, certificate, and certificate-ledger fields of the carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty map iff`

Theorem 59.14 (Continuous function carrier hsame transport)**.** *For histories* $source, map, target, modulus, cert$ *and displayed histories* $source', map', target', modulus', cert'$*, if each displayed history is* hsame*-equivalent to the corresponding carrier field, then the carrier transports across those five endpoint identifications:*

$$\mathsf{hsame}(source, source') \wedge \mathsf{hsame}(map, map') \wedge \mathsf{hsame}(target, target') \wedge \mathsf{hsame}(modulus, modulus') \wedge \mathsf{hsame}(cert, c$$
$$\Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(source', map', t$$

Proof. Unpack the concrete carrier witness into its four unary ledgers and two continuation ledgers. Case analysis on the five hsame witnesses identifies the displayed source, map, target, modulus, and certificate histories with the original fields. The same unary and continuation ledgers then reassemble the displayed ContinuousFunctionCarrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier hsame transport`

Theorem 59.15 (Continuous function carrier target-certificate determinacy)**.** *For histories source, map, target, ta if* ContinuousFunctionCarrier$(source, map, target, modulus, cert)$ *and* ContinuousFunctionCarrier$(source, map, targe$ *then both the target ledger and the certificate ledger are determined up to the history classifier:*

$$\mathsf{hsame}(target, target') \quad \textit{and} \quad \mathsf{hsame}(cert, cert').$$

Proof. Unpack the two carrier witnesses. The first carrier supplies the graph ledger Cont$(source, map, target)$ and the certificate ledger Cont$(target, modulus, cert)$; the second supplies Cont$(source, map, target')$ and Cont$(target', modulus, cert')$.

Continuation determinacy applied to the two graph ledgers gives hsame$(target, target')$. This is the target part of the displayed conclusion.

For the certificate part, keep the common modulus ledger fixed and transport the first certificate continuation across the recovered target sameness. The transported ledger and the second certificate ledger now have the same source history up to hsame and the same modulus history, so continuation respect for hsame gives hsame$(cert, cert')$.

The ten-field certificate fit is the following. The traditional target is single-valued graph-with-modulus evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is target and certificate determinacy. The ledger is source, map, target, modulus, and certificate histories. The core theorem is this determinacy result. The exactness predicate is the pair of displayed hsame conclusions. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier target cert deterministic`

Theorem 59.16 (Continuous function carrier composite public readback)**.** *If two continuous-function carriers compose through fg and $modFG$, and a displayed carrier uses the same source, composite map, and composite modulus, then the displayed target and certificate are determined by the canonical composite target and certificate:*

$$\mathsf{hsame}(target, target') \quad \textit{and} \quad \mathsf{hsame}(cert, cert').$$

Proof. Continuous-function composition closure constructs the canonical composite carrier from the two input carriers and the supplied continuation ledgers for fg, $modFG$, and $cert$. Target-certificate determinacy compares this canonical carrier with the displayed carrier over the same source, map, and modulus, yielding the two public readback equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier comp public readback`

Theorem 59.17 (Continuous function carrier map determinacy)**.** *For histories source, map, map′, target, modulus if two continuous-function carrier witnesses share the same source, target, modulus, and certificate, then their graph maps are determined up to* hsame*:*

ContinuousFunctionCarrier$(source, map, target, modulus, cert) \Longrightarrow$ ContinuousFunctionCarrier$(source, map', target,$

Proof. Unpack both carrier witnesses. Their graph ledgers are $\mathsf{Cont}(source, map, target)$ and $\mathsf{Cont}(source, map', target)$. Continuation left cancellation with the common source history and common target history yields $\mathsf{hsame}(map, map')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier map deterministic`

Theorem 59.18 (Continuous function carrier source determinacy)**.** *For histories $source, source', map, target, mod$ if two continuous-function carrier witnesses share the same graph map, target, modulus, and certificate, then their source histories are determined up to* hsame*:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(source', map, target,$$

Proof. Unpack both carrier witnesses. Their graph ledgers are $\mathsf{Cont}(source, map, target)$ and $\mathsf{Cont}(source', map, target)$. Continuation right cancellation with the common map history and common target history yields $\mathsf{hsame}(source, source')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier source deterministic`

Theorem 59.19 (Continuous function carrier modulus determinacy)**.** *For histories $source, map, target, modulus,$ if two continuous-function carrier witnesses share the same graph target and terminal certificate, then their modulus histories are determined up to* hsame*:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, map, target,$$

Proof. Unpack both carrier witnesses. Their certificate ledgers are $\mathsf{Cont}(target, modulus, cert)$ and $\mathsf{Cont}(target, modulus', cert)$. Continuation left cancellation with the common target history and common certificate history yields $\mathsf{hsame}(modulus, modulus')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus deterministic`

Theorem 59.20 (Continuous function carrier prefix closure)**.** *For histories p, source, map, $targetHist$, modulus, and $certHist$, a visible unary prefix on the source, target, and certificate ledgers preserves the full continuous-function carrier:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, map, targetHist, modulus, certHist) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot targetHist, modulus, p \cdot certHist).$$

Proof. Unpack the carrier into unary histories for the source, target, map, and modulus, together with continuations $targetHist = source \cdot map$ and $certHist = targetHist \cdot modulus$.

Unary append closure supplies the prefixed source and target histories $p \cdot source$ and $p \cdot targetHist$. The map and modulus histories are kept unchanged, so their unary witnesses are reused directly.

The graph ledger for the prefixed carrier follows by applying the prefix p to the continuation and reassociating append:

$$(p \cdot source) \cdot map = p \cdot (source \cdot map) = p \cdot targetHist.$$

The modulus-certificate ledger is transported in the same way:

$$(p \cdot targetHist) \cdot modulus = p \cdot (targetHist \cdot modulus) = p \cdot certHist.$$

The ten-field certificate fit is the following. The traditional target is prefix-stable continuous graph evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is prefix closure of the graph and modulus-certificate ledgers. The ledger consists of source, map, target, modulus, and certificate histories. The core theorem is this prefix closure, exactness is the displayed carrier witness, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefix closed`

Theorem 59.21 (Continuous modulus witness prefix exactness)**.** *For histories p, s, δ, t, a common visible unary prefix on the source and target ledgers is exactly the same as unary evidence for the prefix together with the unprefixed one-step modulus witness:*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusWitness}(s, \delta, t).$$

Proof. For the forward direction, unpack the prefixed witness into unary histories for $p \cdot s$, δ, and $p \cdot t$, together with the continuation ledger

$$(p \cdot s) \cdot \delta = p \cdot t.$$

Unary append factorization on $p \cdot s$ recovers $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(s)$. The same factorization on $p \cdot t$ recovers $\mathsf{UnaryHistory}(t)$. Associativity rewrites the displayed continuation ledger as $p \cdot (s \cdot \delta) = p \cdot t$, and left cancellation on the continuation ledger recovers $s \cdot \delta = t$. These data reassemble $\mathsf{ContinuousModulusWitness}(s, \delta, t)$.

For the reverse direction, project $\mathsf{UnaryHistory}(p)$ and the unprefixed witness from the conjunction. Unpacking the witness supplies unary histories for s, δ, and t, and the continuation ledger $s \cdot \delta = t$.

Unary append closure gives unary histories for $p \cdot s$ and $p \cdot t$. Prefixed continuation associativity transports the unprefixed ledger to $(p \cdot s) \cdot \delta = p \cdot t$, so the prefixed witness is reconstructed.

The ten-field certificate fit is the following. The traditional target is one-step modulus readback. The BEDC source is $BHist$ equipped with Cont. The carrier is ContinuousModulusWitness and the classifier is hsame. The stability obligation is prefix exactness. The ledger consists of source, modulus, target, and prefix histories. The core theorem is this iff, exactness is the displayed iff, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefix iff`

Theorem 59.22 (Continuous modulus witness suffix exactness)**.** *For histories p, s, δ, t, a common visible unary suffix on the modulus and target ledgers is exactly the same as unary evidence for the suffix together with the unsuffixed one-step modulus witness:*

$$\mathsf{ContinuousModulusWitness}(s, \delta \cdot p, t \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusWitness}(s, \delta, t).$$

Proof. For the forward direction, unpack the suffixed witness. Unary append factorization on $\delta \cdot p$ gives unary δ and unary p, while factorization on $t \cdot p$ gives unary t. The continuation ledger has the common right suffix p after associativity, so right cancellation recovers $t = s \cdot \delta$. These fields reassemble the unsuffixed witness.

For the reverse direction, unary closure appends the unary suffix to both the modulus and target histories. Associativity transports the unsuffixed continuation ledger to the suffixed continuation ledger. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness suffix iff`

Theorem 59.23 (Continuous modulus witness visible context exactness)**.** *For histories p, q, s, δ, t, simultaneous visible context on the source, modulus, and target ledgers is exactly the pair of unary context witnesses together with the base one-step modulus witness:*

$$\mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta{\cdot}q, (p{\cdot}t){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{ContinuousModulusWitness}(s, \delta, t$$

Proof. Peel the right suffix from the modulus and target ledgers by suffix exactness. The remaining witness has the common left prefix on source and target, so prefix exactness recovers the unary prefix and the central witness. Conversely, rebuild the prefixed central witness first, then reattach the right suffix. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness visible context iff`

Theorem 59.24 (Continuous function carrier visible modulus context exactness)**.** *For histories $p, q, source, map, target, modulus, cert$, a visible prefix on the source, target, and certificate ledgers together with a visible suffix on the modulus and certificate ledgers is exactly unary evidence for the two context histories together with the central continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Co}$$

Proof. Peel the right suffix from the modulus and certificate ledgers by modulus suffix exactness. The remaining carrier has the common left prefix on source, target, and certificate, so carrier prefix exactness recovers the unary prefix and the central carrier. Conversely, rebuild the prefixed carrier first, then reattach the modulus suffix. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context iff`

Theorem 59.25 (Continuous function carrier visible-context empty-map exactness)**.** *For histories $p, q, source, target, modulus, cert$, the empty graph map remains exact under visible prefix and suffix ledgers:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, \mathsf{e}, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map exactness`

Proof. For the forward direction, apply Theorem 59.169 to strip the visible prefix from source, target, and certificate and the visible suffix from modulus and certificate. The exposed central carrier has empty map, so Theorem 59.13 gives $\mathsf{hsame}(target, source)$ and the central modulus witness.

For the reverse direction, start from unary context evidence, the *target–source* classifier, and the central modulus witness. Use Theorem 59.13 to rebuild the central empty-map carrier, then apply Theorem 59.169 to restore the visible prefix and suffix ledgers.

The ten-field certificate fit is the identity uniformly continuous graph readback over the $BHist/\mathsf{Cont}$ source, with ContinuousFunctionCarrier as carrier, hsame as classifier, visible prefix/suffix stability as the context law, the empty graph map as the readback field, central modulus witness exactness as the certificate content, source and target ledgers classified by hsame, modulus and certificate ledgers preserved by the suffix, and ContinuousMapUp identity-map use downstream. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map iff`

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map exactness`

Theorem 59.26 (Continuous modulus chain visible context exactness)**.** *For histories $p, q, source, first, second, tar$ a visible prefix on the source and target together with a visible suffix on the second modulus and target is exactly unary evidence for the two contexts together with the central two-link chain:*

$$\mathsf{ContinuousModulusChain}(p{\cdot}source, first, second{\cdot}q, (p{\cdot}target){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Continuousl}$$

Proof. Peel the right suffix from the second modulus and target by chain suffix exactness. The remaining chain has the common left prefix on source and target, so chain prefix exactness exposes unary p and the central chain. Conversely, rebuild the prefixed chain first, then reattach the suffix. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain visible context iff`

Theorem 59.27 (Continuous modulus chain visible context second determinacy)**.** *For histories $p, q, source, first, second, second', target$, if two visible-context two-link modulus chains share the same visible source, first modulus, and visible target,*

$$\mathsf{ContinuousModulusChain}(p{\cdot}source, first, second{\cdot}q, (p{\cdot}target){\cdot}q) \quad \textit{and} \quad \mathsf{ContinuousModulusChain}(p{\cdot}source, first, s$$

then secondhsame*second'*.

Proof. Apply the visible-context exactness statement to both chains. This removes the common prefix and suffix contexts and exposes two central chains with the same source, first modulus, and target. The ordinary second-modulus determinacy theorem identifies the two central second moduli. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain visible context second deterministic`

Theorem 59.28 (Continuous function carrier visible-context field determinacy)**.** *For histories $p, q, source, source', map, map', target, target', modulus, modulus', cert, cert'$, visible-modulus-context continuous-function carriers determine each central public field as follows.*

1. *Same visible source, target, modulus, and certificate determine the map:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

2. *Same visible map, target, modulus, and certificate determine the source:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

3. *Same visible source, map, target, and certificate determine the modulus:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

4. *Same visible source, map, and modulus determine the target and certificate:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

Proof. 1. In the map clause, apply Theorem 59.169 to the two displayed visible carriers. The exposed central carriers share source, target, modulus, and certificate, so Theorem 59.17 gives $\mathsf{hsame}(map, map')$.

2. In the source clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share map, target, modulus, and certificate, so Theorem 59.18 gives $\mathsf{hsame}(source, source')$.

3. In the modulus clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share source, map, target, and certificate, so Theorem 59.19 gives $\mathsf{hsame}(modulus, modulus')$.

4. In the target-certificate clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share source, map, and modulus, so Theorem 59.15 gives $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert')$.

□

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context field determinacy`

Theorem 59.29 (Continuous function carrier visible-context target-certificate determinacy)**.** *For histories $p, q, source, map, target, target', modulus, cert, cert'$, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus \cdot q, (p \cdot cert') \cdot q)$$

hold, then the public target and certificate ledgers are determined by the central presentation:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.169 to the first visible carrier. This recovers the unary context histories $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(q)$ together with the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, targe$

Apply the same exactness theorem to the second visible carrier. The common visible prefix and suffix are peeled in the same way, producing the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, target', moc$

Now apply Theorem 59.15 to the two central carrier witnesses. It gives the displayed $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert, cert')$ conclusions, which are already the public target and certificate comparisons required here.

The ten-field certificate fit is the following. The traditional target is public target-certificate determinacy for visible continuous-function carrier presentations. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context public readback. The ledger is source, map, target, modulus, and certificate histories together with the visible prefix and suffix contexts. The core theorem is this result. The exactness predicate is the displayed pair of hsame conclusions. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context target cert deterministic`

Theorem 59.30 (Continuous function carrier visible-context map determinacy)**.** *For histories* $p, q, source, map, map', target, modulus, cert$*, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map', p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

hold, then the public graph map ledger is determined by the central presentation:

$$\mathsf{hsame}(map, map').$$

Proof. Apply Theorem 59.169 to the first visible carrier. This removes the shared visible prefix and suffix contexts and exposes the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert)$.

Apply the same visible-context exactness theorem to the second carrier. The same source, target, modulus, and certificate ledgers remain, producing $\mathsf{ContinuousFunctionCarrier}(source, map', target, modulus, ce$

Now apply Theorem 59.17 to the two central carrier witnesses. It gives the displayed $\mathsf{hsame}(map, map')$ comparison, which is already the public visible-context map readback required here.

The ten-field certificate fit is the following. The traditional target is visible map-ledger uniqueness. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context map readback. The ledger is source, map, target, modulus, and certificate histories together with the visible prefix and suffix contexts. The core theorem is this result. The exactness predicate is the displayed hsame conclusion. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context map deterministic`

Theorem 59.31 (Continuous function carrier visible-context graph-modulus determinacy)**.** *For histories* $p, q, source, map, map', target, modulus, modulus', cert$*, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map', p \cdot target, modulus' \cdot q, (p \cdot cert) \cdot q)$$

hold, then the public graph and modulus ledgers are simultaneously determined:

$$\mathsf{hsame}(map, map') \wedge \mathsf{hsame}(modulus, modulus').$$

Proof. Apply Theorem 59.169 to both visible carrier hypotheses. The common displayed prefix and suffix are peeled away, leaving central carriers over $(source, map, target, modulus, cert)$ and $(source, map', target, modulus', cert)$.

Apply Theorem 59.139 to the two central carriers. This exposes graph ledgers $\mathsf{Cont}(source, map, target)$ and $\mathsf{Cont}(source, map', target)$, and certificate ledgers $\mathsf{Cont}(target, modulus, cert)$ and $\mathsf{Cont}(target, modulus', cert)$

Continuation left cancellation on the two graph ledgers gives $\mathsf{hsame}(map, map')$. The same cancellation principle on the two target-certificate ledgers gives $\mathsf{hsame}(modulus, modulus')$, and the two comparisons are paired as the displayed conclusion.

The ten-field certificate fit is: traditional target visible continuous-function determinacy; BEDC source BHist with Cont; carrier ContinuousFunctionCarrier; classifier hsame; stability obligation visible-context public readback; ledger source, map, target, modulus, certificate, prefix, and suffix; core theorem this statement; exactness the displayed pair of hsame conclusions; standard bridge deferred; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context graph modulus deterministic`

Theorem 59.32 (Continuous function carrier visible-context cycle tails empty)**.** *For histories $p, q, source, map, target, modulus, cert$, a visible-context continuous-function carrier whose central source and certificate are history-same has empty central graph and modulus tails:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{hsame}(source, cert) \Longrightarrow \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context cycle tails empty`

Proof. Apply Theorem 59.169 to the displayed visible carrier. This recovers unary witnesses for the visible prefix p and suffix q, and it exposes the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulu$

Apply Theorem 59.140 to the central carrier and the displayed $\mathsf{hsame}(source, cert)$ cycle. The endpoint-certificate cycle cancellation theorem gives exactly $\mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e})$.

The ten-field certificate fit is the following. The traditional target is visible cycle-tail pruning. The BEDC source is BHist with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context central readback. The ledger is source, map, target, modulus, certificate, prefix, and suffix histories. The core theorem is this statement. The exactness predicate is the displayed pair of empty-tail conclusions. The standard bridge is deferred, and the status is ProofTarget. □

Theorem 59.33 (Continuous function carrier visible-context composition closure)**.** *For histories $p, q, source, middle, target, f, g, fg, modF, modG, modFG, certF, certG, cert$, suppose the two contextualized carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold. If the graph, modulus, and target-certificate ledgers

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(target, modFG, cert)$$

hold, then the composite carrier remains inside the same visible context:

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, fg, p \cdot target, modFG \cdot q, (p \cdot cert) \cdot q).$$

Proof. Apply Theorem 59.169 to the first contextualized carrier. This recovers unary witnesses for the visible prefix p and suffix q, together with the central carrier $\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF,$

Apply the same visible-context exactness theorem to the second contextualized carrier. The common contexts are peeled away, exposing $\mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, certG)$ over the same central middle and target histories.

Apply Theorem 59.7 to these two central carriers with the displayed graph, modulus, and target-certificate ledgers. The result is the central composite carrier $\mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG,$

Finally use Theorem 59.169 in the reverse direction, with the recovered unary witnesses for p and q, to reattach the visible prefix and suffix. This yields the displayed composite carrier in the same visible context.

The ten-field certificate fit is the following. The traditional target is composition closure for contextual continuous-function carriers. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is contextual composition closure with fixed visible prefix and suffix. The ledger consists of source, middle, target, graph map, modulus, certificate, visible prefix, and visible suffix histories. The core theorem is this visible-context closure statement. The exactness claim is the reverse use of visible-context exactness after central composition closure. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context comp central package`

Theorem 59.34 (Continuous function carrier visible-context composition public readback)**.** *For histories* $p, q, source, middle, target, target', f, g, fg, modF, modG, modFG, certF, certG, cert, cert'$*, suppose the two contextualized carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold, together with the graph, modulus, and target-certificate ledgers

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(target, modFG, cert).$$

If a displayed visible composite carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, fg, p \cdot target', modFG \cdot q, (p \cdot cert') \cdot q)$$

also holds, then the public composite target and certificate read back to the canonical central pair:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.169 to the first contextualized carrier. This peels the common visible prefix and suffix, and it exposes the central carrier $\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF)$.

Apply the same visible-context exactness theorem to the second contextualized carrier. The central carrier $\mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, certG)$ is recovered over the same middle history and canonical target history.

Apply the visible-context exactness theorem once more to the displayed composite carrier. This removes the displayed visible context and gives the central displayed carrier $\mathsf{ContinuousFunctionCarrier}(source, fg, ta$

Now apply Theorem 59.16 to the two central input carriers, the three displayed continuation ledgers, and the central displayed composite carrier. The central public readback theorem gives exactly $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert')$.

The ten-field certificate fit is the following. The traditional target is public readback for contextual continuous-function composition. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is contextual composite target and certificate readback. The ledger consists of source, middle, target, displayed target, graph map, modulus, certificate, displayed certificate, visible prefix, and visible suffix histories. The core theorem is this visible-context readback statement. The exactness claim is the central target-certificate pair returned after peeling visible context. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composition public readback`

Theorem 59.35 (Continuous function carrier visible-context composite ledger determinacy)**.** *For histories* $p, q, source, middle, target, f, g, fg, fg', modF, modG, modFG, modFG', certF, certG, cert, cert'$*, suppose the two visible-context input carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold. If both composite ledger triples

$$\begin{array}{lll} \mathsf{Cont}(f, g, fg), & \mathsf{Cont}(modF, modG, modFG), & \mathsf{Cont}(target, modFG, cert), \\ \mathsf{Cont}(f, g, fg'), & \mathsf{Cont}(modF, modG, modFG'), & \mathsf{Cont}(target, modFG', cert') \end{array}$$

hold, then the public composite graph, composite modulus, and composite certificate ledgers are independent of the chosen admissible triple:

$$\mathsf{hsame}(fg, fg') \wedge \mathsf{hsame}(modFG, modFG') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.178 to the two input carriers and the first composite ledger triple. This builds the visible composite carrier over $(fg, modFG, cert)$. Apply the same closure theorem to the same input carriers and the second composite ledger triple, giving the visible composite carrier over $(fg', modFG', cert')$.

Apply continuation determinacy, Theorem 18.16, to the two graph ledgers $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g, fg')$. Apply the same determinacy theorem to the two modulus ledgers $\mathsf{Cont}(modF, modG, modFG)$ and $\mathsf{Cont}(modF, modG, modFG')$. These give $\mathsf{hsame}(fg, fg')$ and $\mathsf{hsame}(modFG, modFG')$.

Use Theorem 59.169 to peel the visible prefix and suffix from the two composite carriers. Then use Theorem 59.14 to place the two central composite carriers over a common graph and modulus representative while leaving the target fixed.

Now Theorem 59.15 applies to the aligned central carriers. Its certificate conclusion gives $\mathsf{hsame}(cert, cert')$, which combines with the graph and modulus comparisons above to give the displayed triple classifier.

The ten-field certificate fit is the following. The traditional target is composite-ledger well-definedness for continuous maps. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is composite ledger independence. The ledger consists of source, middle, target, map, modulus, certificate, prefix, and suffix histories. The core theorem is this determinacy result. The exactness claim is the displayed triple classifier. The downstream consumer is ContinuousMapUp composition well-definedness. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composite ledger determinacy`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composite ledger deterministic`

Theorem 59.36 (Continuous function carrier visible fourfold left-right synchronization)**.** *For BHist histories* $p, q, source, mid_1, mid_2, mid_3, target, f, g, h, k, fg, fgh, hk, ghk, map_L, map_R, mod_F, mod_G, mod_H,$ *suppose the four visible-context one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot mid_1, mod_F \cdot q, (p \cdot cert_F) \cdot q),\\
&\mathsf{ContinuousFunctionCarrier}(p \cdot mid_1, g, p \cdot mid_2, mod_G \cdot q, (p \cdot cert_G) \cdot q),\\
&\mathsf{ContinuousFunctionCarrier}(p \cdot mid_2, h, p \cdot mid_3, mod_H \cdot q, (p \cdot cert_H) \cdot q),\\
&\mathsf{ContinuousFunctionCarrier}(p \cdot mid_3, k, p \cdot target, mod_K \cdot q, (p \cdot cert_K) \cdot q)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, map_L), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(h, k, hk), & \mathsf{Cont}(mod_H, mod_K, mod_{HK}), & \mathsf{Cont}(target, mod_{HK}, cert_{HK}),\\
\mathsf{Cont}(g, hk, ghk), & \mathsf{Cont}(mod_G, mod_{HK}, mod_{GHK}), & \mathsf{Cont}(target, mod_{GHK}, cert_{GHK}),\\
\mathsf{Cont}(f, ghk, map_R), & \mathsf{Cont}(mod_F, mod_{GHK}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the two visible-context composite carriers exist and their public graph, modulus, and certificate histories agree:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map_L, p \cdot target, mod_L \cdot q, (p \cdot cert_L) \cdot q) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map_R, p \cdot target, mod_R \cdot q, (p \cdot cert_R) \cdot q) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. Apply Theorem 59.169 to each of the four displayed visible-context one-step carriers. This peels the common visible prefix p from source, target, and certificate ledgers and the common visible suffix q from modulus and certificate ledgers. The result is unary evidence for the two context histories together with the four central carriers from *source* through *target*.

Apply Theorem 59.156 to those four central carriers and to the displayed left-deep and right-deep ledger arrays. The central theorem constructs the left-deep carrier $\mathsf{ContinuousFunctionCarrier}(source, map_L, ta$ the right-deep carrier $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$, and the three public synchronizations $\mathsf{hsame}(map_L, map_R)$, $\mathsf{hsame}(mod_L, mod_R)$, and $\mathsf{hsame}(cert_L, cert_R)$.

Use Theorem 59.169 in the reverse direction with the recovered unary witnesses for p and q to reattach the visible context to the left-deep central carrier. Use the same reverse implication for the right-deep central carrier. These two applications give exactly the two displayed visible-context composite carriers.

The ten-field certificate fit is the following. The traditional target is fourfold contextual continuous composition coherence. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the source, intermediate, target, graph, modulus, certificate, visible prefix, and visible suffix histories displayed above. The stability field is visible-context left-right bracketing synchronization. The core theorem is this visible fourfold statement. The exactness field is the pair of reattached visible carriers together with the three hsame conclusions. The downstream consumer is ContinuousMapUp associativity coherence, the standard bridge is deferred, and the status is ProofTarget. □

Theorem 59.37 (Continuous modulus witness visible context source determinacy)**.** *For histories p, q, s, s', δ, t, two visible-context one-step modulus witnesses with the same visible modulus and visible target contexts determine the same central source history:*

$$\mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta{\cdot}q, (p{\cdot}t){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(p{\cdot}s', \delta{\cdot}q, (p{\cdot}t){\cdot}q) \Longrightarrow \mathsf{hsame}(s, s').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central witnesses share the modulus δ and target t, so one-step source determinacy gives $\mathsf{hsame}(s, s')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness visible context source deterministic`

Theorem 59.38 (Continuous modulus witness visible context modulus determinacy)**.** *For histories $p, q, s, \delta, \delta', t$, two visible-context one-step modulus witnesses with the same visible source and target contexts determine the same central modulus history:*

$$\mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta{\cdot}q, (p{\cdot}t){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta'{\cdot}q, (p{\cdot}t){\cdot}q) \Longrightarrow \mathsf{hsame}(\delta, \delta').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central witnesses have the same source s and target t, so one-step modulus determinacy identifies δ and δ'. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness visible context modulus deterministic`

Theorem 59.39 (Continuous modulus witness visible context target determinacy)**.** *For histories p, q, s, δ, t, t', two visible-context one-step modulus witnesses with the same visible source and modulus contexts determine the same central target history:*

$$\mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta{\cdot}q, (p{\cdot}t){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(p{\cdot}s, \delta{\cdot}q, (p{\cdot}t'){\cdot}q) \Longrightarrow \mathsf{hsame}(t, t').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central witnesses share source s and modulus δ, so one-step target determinacy gives $\mathsf{hsame}(t, t')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness visible context target deterministic`

Theorem 59.40 (Continuous modulus witness zero-headed component impossibility)**.** *For histories $source, modulus, target$: BHist, no concrete one-step modulus witness has a zero-headed source, modulus, or target component:*

$$(\mathsf{ContinuousModulusWitness}(\mathsf{E0}(source), modulus, target) \to \mathsf{False}) \wedge (\mathsf{ContinuousModulusWitness}(source, \mathsf{E0}(mod$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness e0 component absurd`

Proof. The witness definition records unary-history evidence for its source, modulus, and target components. Each displayed $\mathsf{E0}$ component contradicts the unary no-zero-extension theorem, since unary histories are generated only by the empty history and one-headed extension. □

Theorem 59.41 (Continuous modulus witness left e1 result cases)**.** *For histories $source, modulus, target$, a one-step modulus witness from a visible source endpoint to a visible target endpoint has only the empty modulus case or a visible modulus tail:*

$$\mathsf{ContinuousModulusWitness}(\mathsf{E1}(source), modulus, \mathsf{E1}(target)) \Longrightarrow (modulus = \mathsf{e} \wedge \mathsf{UnaryHistory}(source) \wedge \mathsf{hsame}(sou$$

Proof. Unpack the modulus witness and invert the continuation ledger with visible result $\mathsf{E1}(target)$. The empty modulus case strips the outer visible constructor from the source-target equality and keeps the unary source evidence. The zero-tagged modulus case is ruled out by constructor no-confusion. In the visible modulus case, remove one $\mathsf{E1}$ constructor from the continuation ledger and repack the remaining fields as the tail witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness left e1 result cases`

Theorem 59.42 (Continuous modulus witness right e1 result exactness)**.** *For histories source, modulus, target, a modulus witness whose modulus and target histories are both visibly tagged has an exact untagged tail:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow \mathsf{ContinuousModulusWitness}(source, modulus, tar$$

Proof. Unpack the source, modulus, target unary witnesses and the continuation ledger. Unary inversion removes the visible constructor from the modulus and target carriers. The right $\mathsf{E1}$ continuation inversion removes the same constructor from the continuation ledger, yielding the exact untagged modulus witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 result exactness`

Theorem 59.43 (Continuous modulus witness right e1 visible source exactness)**.** *For histories source, modulus, target, sourceTail, a modulus witness whose modulus and target histories are both visibly tagged descends to the tail while respecting a visible presentation of the source endpoint:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow \mathsf{hsame}(source, \mathsf{E1}(sourceTail)) \Longrightarrow \mathsf{ContinuousM}$$

Proof. Apply right one-headed result exactness to strip the visible constructor from the modulus and target components of the witness. The displayed hsame presentation of the source endpoint then transports the descended witness to the concrete visible source $\mathsf{E1}(sourceTail)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 visible source exactness`

Theorem 59.44 (Continuous modulus witness right e1 source cases)**.** *For histories source, modulus, target, a modulus witness whose modulus and target histories are both visibly tagged has only the empty source case or a visible source tail:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow (source = \mathsf{e} \wedge \mathsf{UnaryHistory}(modulus) \wedge \mathsf{hsame}(tar$$

Proof. First descend the witness by stripping the common visible constructor from the modulus and target histories. The descended source carrier is unary, so it is either empty or visibly tagged. In the empty case, the continuation ledger is a left-unit ledger and gives $\mathsf{hsame}(target, modulus)$. In the visible case, constructor equality identifies the source with $\mathsf{E1}(sourceTail)$ and the descended witness is repacked with that source presentation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 source cases`

Theorem 59.45 (Continuous function carrier terminal modulus extension)**.** *For histories source, map, target, mod a continuous-function carrier remains a continuous-function carrier after the terminal certificate is extended by one more modulus witness and the modulus ledger is composed:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert, extra, cert') =$$

Proof. Unpack the given carrier. This exposes the graph ledger $\mathsf{Cont}(source, map, target)$ and the terminal modulus-certificate ledger $\mathsf{Cont}(target, modulus, cert)$, together with the unary carrier data for the displayed histories.

Unpack the additional modulus witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$. It supplies $\mathsf{Cont}(cert, extra, cert')$, so the two ledgers $\mathsf{Cont}(target, modulus, cert)$ and $\mathsf{Cont}(cert, extra, cert')$ assemble the chain $\mathsf{ContinuousModulusChain}(target, modulus, extra, cert')$ with middle history $cert$.

Apply Theorem 59.108 to the original carrier, this chain, and the displayed composition ledger $\mathsf{Cont}(modulus, extra, modulus')$. The replacement theorem repacks the preserved graph data with the composite terminal modulus certificate, giving the displayed carrier conclusion.

The ten-field certificate fit is the following. The traditional target is terminal modulus extension for continuous maps. The BEDC source is BHist with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is preservation under one terminal modulus extension. The ledger is source, map, target, original modulus and certificate, extra modulus, composite modulus, and terminal certificate. The core theorem is this closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier terminal modulus extension`

Theorem 59.46 (Continuous function carrier visible terminal modulus extension)**.** *For histories $p, q, source, map, target, modulus, cert, extra, modulus', cert'$, visible continuous-function carrier data remain carrier data after the central terminal certificate is extended and the terminal modulus ledger is composed:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert, extr$$

Proof. Peel the visible presentation by Theorem 59.169. This recovers the unary context histories $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(q)$ together with the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target$

Apply Theorem 59.162 to the central carrier, the witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$, and the composition ledger $\mathsf{Cont}(modulus, extra, modulus')$. The conclusion is the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus', cert')$.

Use the reverse direction of Theorem 59.169 with the recovered unary histories. It repacks the same prefix p and suffix q around the central carrier, producing the displayed visible carrier conclusion. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension`

Theorem 59.47 (Continuous function carrier visible terminal modulus extension output determinacy)**.** *For histories $p, q, source, map, target, target', modulus, cert, extra, modulus', cert', cert''$, suppose*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

holds, together with $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$ *and* $\mathsf{Cont}(modulus, extra, modulus')$*. If the post-extension visible carrier*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus' \cdot q, (p \cdot cert'') \cdot q)$$

also holds, then the visible target and terminal certificate are determined:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert', cert'').$$

Proof. Apply Theorem 59.163 to the original visible carrier, the central witness $\mathsf{ContinuousModulusWitness}(cert, ext$ and the terminal modulus ledger $\mathsf{Cont}(modulus, extra, modulus')$. This constructs the canonical post-extension visible carrier with target *target* and terminal certificate $cert'$.

Compare that canonical carrier with the independently displayed post-extension carrier by Theorem 59.174. The two carrier presentations share p, q, *source*, *map*, and $modulus'$, so the determinacy theorem returns exactly $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert', cert'')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension output determinacy`

Theorem 59.48 (Continuous function carrier visible terminal modulus extension transitive)**.** *For histories* $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2$, *visible continuous-function carrier data remain carrier data after two consecutive terminal certificate extensions and their two terminal modulus ledgers:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus_0{\cdot}q, (p{\cdot}cert_0){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_0, e$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension transitive`

Proof. Apply Theorem 59.163 to the displayed visible carrier, the witness $\mathsf{ContinuousModulusWitness}(cert_0, extra_1,$ and the first composition ledger $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$. This gives the visible carrier with terminal modulus $modulus_1$ and terminal certificate $cert_1$.

Apply the same theorem again to that intermediate carrier, the witness $\mathsf{ContinuousModulusWitness}(cert_1, extra_2$ and the second composition ledger $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$. The result is the visible carrier whose terminal modulus is $modulus_2$ and whose terminal certificate is $cert_2$, which is the displayed conclusion.

The ten-field certificate fit is the following. The traditional target is two-step terminal modulus extension for continuous maps. The BEDC source is BHist with Cont, ContinuousModulusWitness, and ContinuousFunctionCarrier. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is preservation under two consecutive terminal modulus extensions. The ledger is the two extra modulus witnesses and the two composite modulus rows. The core theorem is this closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is ProofTarget. □

Theorem 59.49 (Continuous function carrier visible terminal modulus reassociation)**.** *For histories* $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, extra_{12}, modulus_2, modulus_{12}, cert_2$ *sequential two-step terminal extension and the corresponding composite one-step terminal extension produce the same visible terminal modulus data:*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\
&\mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_2 \cdot q, (p \cdot cert_2) \cdot q) \wedge \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{12} \cdot q, (p \cdot cert_2) \cdot q) \wedge \\
&\mathsf{hsame}(modulus_2, modulus_{12}).
\end{aligned}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation`

Proof. Apply Theorem 59.165 to the displayed visible carrier, the two certificate-extension witnesses, and the two sequential terminal modulus ledgers. This constructs the sequential visible carrier with terminal data $(modulus_2, cert_2)$.

Next assemble the certificate witnesses $\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1)$ and ContinuousModulusW into $\mathsf{ContinuousModulusChain}(cert_0, extra_1, extra_2, cert_2)$ with middle history $cert_1$. By Theorem 59.105 and the ledger $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, this gives the composite witness ContinuousModulusWitness

Apply Theorem 59.163 to the original visible carrier, this composite witness, and $\mathsf{Cont}(modulus_0, extra_{12}, modu$ The result is the composite one-step visible carrier with terminal data $(modulus_{12}, cert_2)$.

Finally use Theorem 4.56 on $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$, $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$, $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, and $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$. It identifies the sequential and composite terminal moduli, giving $\mathsf{hsame}(modulus_2, modulus_{12})$, and the three displayed conclusions are then packed together. □

↪ variant BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation

Theorem 59.50 (Continuous function carrier visible terminal modulus reassociation output determinacy)**.** *For histories $p, q, source, map, target, target', modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, extra_{12}, n$ the displayed composite visible output is determined by the reassociated terminal modulus construction:*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\
&\mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus_{12} \cdot q, (p \cdot cert'') \cdot q) \Longrightarrow \\
&\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_2, cert'') \wedge \mathsf{hsame}(modulus_2, modulus_{12}).
\end{aligned}$$

Lean4 ✓ BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation output determinacy

Proof. First apply Theorem 59.166 to the original visible carrier, the two certificate-extension witnesses, and the four terminal modulus ledgers. This supplies the canonical composite visible carrier $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{12} \cdot q, (p \cdot cert_2) \cdot q)$ together with $\mathsf{hsame}(modulus_2, modulus_{12})$.

Next compare that canonical composite carrier with the independently displayed composite carrier $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus_{12} \cdot q, (p \cdot cert'') \cdot q)$. By Theorem 59.174, the common visible source, map, and terminal modulus determine the public target and certificate, giving $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert_2, cert'')$.

Finally package those two target-certificate comparisons with the modulus comparison obtained from reassociation. The result is the displayed triple $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_2, cert'') \wedge \mathsf{hsame}(modulus_2, modulus_{12})$. □

Theorem 59.51 (Continuous function carrier visible terminal modulus three-step reassociation)**.** *For histories $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra_3, mo$ sequential three-step terminal extension and the corresponding composite one-step terminal exten-*

sion produce the same visible terminal modulus data:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\
&\mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\
&\mathsf{Cont}(modulus_2, extra_3, modulus_3) \Longrightarrow \mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \mathsf{Cont}(extra_{12}, extra_3, extra_{123}) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}) \Longrightarrow \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_3 \cdot q, (p \cdot cert_3) \cdot q) \wedge \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q) \wedge \\
&\mathsf{hsame}(modulus_3, modulus_{123}).
\end{aligned}$$

Proof. Apply Theorem 59.165 to the displayed visible carrier, the first two certificate-extension witnesses, and the first two sequential terminal modulus ledgers. This gives the visible carrier with terminal data $(modulus_2, cert_2)$. Apply Theorem 59.163 to that carrier, the third certificate-extension witness, and $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$. The sequential visible carrier with terminal data $(modulus_3, cert_3)$ follows.

Next compose the first two certificate witnesses. They form the chain $\mathsf{ContinuousModulusChain}(cert_0, extra_1, ext$ with middle history $cert_1$, so Theorem 59.105 and $\mathsf{Cont}(extra_1, extra_2, extra_{12})$ supply $\mathsf{ContinuousModulusWitness}($

Compose that witness with the third one. The chain $\mathsf{ContinuousModulusChain}(cert_0, extra_{12}, extra_3, cert_3)$ with middle history $cert_2$, together with $\mathsf{Cont}(extra_{12}, extra_3, extra_{123})$, gives $\mathsf{ContinuousModulusWitness}(cert_0, ext$ by the same composite-closure theorem.

Apply Theorem 59.163 to the original visible carrier, the composite witness just constructed, and $\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123})$. This produces the composite one-step visible carrier with terminal data $(modulus_{123}, cert_3)$.

It remains to identify the two terminal moduli. First use Theorem 4.56 on $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$, $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$, $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, and $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$, obtaining $\mathsf{hsame}(modulus_2, modulus_{12})$.

Transport the third terminal continuation along that comparison, carrying $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$ to the row $\mathsf{Cont}(modulus_{12}, extra_3, modulus_3)$. Then apply Theorem 4.56 to $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$, the transported row, $\mathsf{Cont}(extra_{12}, extra_3, extra_{123})$, and $\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123})$. The resulting comparison is $\mathsf{hsame}(modulus_3, modulus_{123})$, and the two visible carriers and this comparison form the displayed triple. □

Theorem 59.52 (Continuous modulus witness modulus determinacy)**.** *If two one-step modulus witnesses have* hsame *source histories and* hsame *target histories, then their modulus histories are* hsame*:*

$$\mathsf{hsame}(modulus, modulus').$$

Proof. Unpack both witnesses, identify the source and target endpoints by hsame, and apply continuation left cancellation to the two modulus ledgers. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness modulus hsame deterministic`

Theorem 59.53 (Continuous modulus witness target determinacy)**.** *If two one-step modulus witnesses have* hsame *source histories and the same modulus history, then their target histories are* hsame*:*

$$\mathsf{hsame}(target, target').$$

Proof. Unpack both witnesses, identify the source endpoint by hsame, and apply continuation result determinacy to the two continuation ledgers with the shared source and modulus. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness target hsame deterministic`

Theorem 59.54 (Continuous modulus witness source determinacy)**.** *If two one-step modulus witnesses have* hsame *target histories and the same modulus history, then their source histories are* hsame*:*

$$\mathsf{hsame}(source, source').$$

Proof. Unpack both witnesses, identify the target endpoint by hsame, and apply continuation right cancellation to the two ledgers with the shared modulus and target. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness source hsame deterministic`

Theorem 59.55 (Continuous modulus witness empty target exactness)**.** *For histories source and modulus, a one-step continuous modulus witness with empty target is exactly the conjunction that both source and modulus are empty histories:*

$$\mathsf{ContinuousModulusWitness}(source, modulus, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e}).$$

Proof. Unpack the witness into unary carriers and the continuation ledger $\mathsf{Cont}(source, modulus, \mathsf{e})$. Empty-result inversion for continuation gives the two endpoint equalities. Conversely, if both endpoints are empty, the empty unary carrier and the canonical empty continuation rebuild the witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty target iff`

Theorem 59.56 (Continuous empty-source nonempty modulus shape)**.** *For histories target, modulus, an empty-source continuous modulus witness with nonempty modulus has exactly one visible unary payload:*

$$\mathsf{ContinuousModulusWitness}(\mathsf{e}, modulus, target) \wedge \neg\mathsf{hsame}(modulus, \mathsf{e}) \iff \exists r : \mathsf{BHist}.\ target = \mathsf{E1}(r) \wedge modulus = \mathsf{E}$$

Proof. Empty-source modulus exactness identifies *modulus* with *target* and gives unary membership of *target*. Nonemptiness excludes the empty target, unary membership excludes zero-headed targets, and the visible case strips one E1 constructor to expose the unary payload. The reverse direction rebuilds the witness from the visible unary target and reflexive sameness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty source nonempty modulus shape iff`

Theorem 59.57 (Continuous modulus chain empty target exactness)**.** *For histories source, first, second, a two-link continuous modulus chain ending at* e *is exactly the conjunction that all three displayed histories are empty:*

$$\mathsf{ContinuousModulusChain}(source, first, second, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(first, \mathsf{e}) \wedge \mathsf{hsame}(second, \mathsf{e}).$$

Proof. Unpack the middle history and the two continuation ledgers. Empty-result inversion on the second leg identifies the middle history and *second* with e; after replacing the middle endpoint, empty-result inversion on the first leg identifies *source* and *first* with e. The reverse direction uses the empty middle history and the two right-unit continuations at e. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty target iff`

Theorem 59.58 (Continuous modulus chain empty moduli exactness)**.** *For histories source, target, a two-link continuous modulus chain with both moduli empty is exactly the conjunction that the endpoints are unary and the target is history-same to the source:*

$$\mathsf{ContinuousModulusChain}(source, \mathsf{e}, \mathsf{e}, target) \iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(target) \wedge \mathsf{hsame}(target, sourc$$

Proof. Unpack the middle history and the two continuation ledgers. Each empty-modulus leg is a right-unit continuation, so the middle endpoint is history-same to the source and the target endpoint is history-same to the middle endpoint. Transitivity gives $\mathsf{hsame}(target, source)$. Conversely, choose *source* as the middle history, use the canonical right-unit continuation for the first leg, and use the displayed endpoint sameness to build the second right-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty moduli iff`

Theorem 59.59 (Continuous modulus chain empty second witness)**.** *For histories source, first, second, target, if a two-link continuous modulus chain has second*hsamee*, then the first link directly witnesses* $\mathsf{ContinuousModulusWitness}(source, first, target)$*.*

Proof. Unpack the middle history and the two continuation ledgers. The second ledger, transported across *second*hsamee, is compared with the right-unit continuation at the middle endpoint, so the target is history-same to the middle endpoint. Transporting the first continuation result along this endpoint sameness gives the displayed one-step witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty second witness`

Theorem 59.60 (Continuous modulus chain empty first witness)**.** *For histories source, first, second, target, if a two-link continuous modulus chain has first*hsamee*, then the second link directly witnesses* $\mathsf{ContinuousModulusWitness}(source, second, target)$*.*

Proof. Unpack the middle history and the two continuation ledgers. The first ledger, transported across *first*hsamee, is compared with the right-unit continuation at the source endpoint, so the middle endpoint is history-same to the source. Transporting the second continuation source along this endpoint sameness gives the displayed one-step witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty first witness`

Theorem 59.61 (Continuous function carrier empty certificate exactness)**.** *For histories source, map, target, modu a continuous-function carrier with empty certificate is exactly the conjunction that its graph and modulus histories are all empty:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(target, \mathsf{e})$$

Proof. Unpack the graph continuation and the target-to-certificate continuation. Empty-result inversion on the certificate leg identifies *target* and *modulus* with e; after replacing the target endpoint, empty-result inversion on the graph leg identifies *source* and *map* with e. Conversely, when all four histories are empty, the empty unary carrier and the two right-unit continuations rebuild the carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty cert iff`

Theorem 59.62 (Continuous function carrier empty-modulus exactness)**.** *For histories source, map, target, cert, a continuous-function carrier with empty modulus is exactly the graph carrier data together with the statement that the certificate is history-same to the target:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \mathsf{e}, cert) \iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(target) \wedge \mathsf{UnaryHist}$$

Proof. Unpack the continuous-function carrier. The graph fields already supply the three displayed unary histories and the graph continuation ledger. The target-to-certificate leg has empty modulus, so the right-unit continuation law identifies *cert* with *target*. Conversely, the displayed graph data, the empty unary carrier, and the right-unit continuation rebuilt from $\mathsf{hsame}(cert, target)$ reassemble the continuous-function carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty modulus iff`

Theorem 59.63 (Continuous function carrier empty-target exactness)**.** *For histories source, map, modulus, cert, the empty target case of* ContinuousFunctionCarrier *is exact:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, \mathsf{e}, modulus, cert) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{UnaryHistory}(mod$$

Proof. In the forward direction, empty-result inversion on the graph ledger identifies the source and map histories with e. The certificate ledger is then a left-unit continuation from e through *modulus*, so the certificate is history-same to the modulus; unary continuation closure supplies the certificate carrier.

Conversely, the displayed empty source and map rebuild the graph ledger by the right-unit continuation at e. The displayed certificate-modulus sameness rebuilds the certificate ledger by the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty target iff`

Theorem 59.64 (Continuous function carrier empty-source exactness)**.** *For histories map, target, modulus, and cert, the empty source case of* ContinuousFunctionCarrier *is exact:*

$$\mathsf{ContinuousFunctionCarrier}(\mathsf{e}, map, target, modulus, cert) \iff \mathsf{UnaryHistory}(map) \wedge \mathsf{ContinuousModulusWitness}(map$$

Proof. In the forward direction, unpack the carrier. The graph ledger is $\mathsf{Cont}(\mathsf{e}, map, target)$, so the continuation left-unit theorem identifies *target* with *map*. The target-modulus continuation then becomes a modulus witness over *map*, and unary continuation closure supplies the certificate carrier.

Conversely, the displayed modulus witness supplies the map, modulus, certificate, and certificate-continuation fields. The empty source is unary, the target is identified with the map by the displayed hsame, and the graph ledger is the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source iff`

Theorem 59.65 (Continuous function carrier graph-modulus readback)**.** *For histories source, map, target, modulu every continuous-function carrier directly exposes its graph continuation ledger, its modulus-certificate continuation ledger, and the two corresponding one-step modulus witnesses:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{Cont}(source, map, target) \wedge \mathsf{Cont}(target, modu$$
$$\wedge\ \mathsf{ContinuousModulusWitness}(source, map, target) \wedge \mathsf{ContinuousModulusWitness}(target, modulus, cert).$$

Proof. Unpack the continuous-function carrier into unary data for *source*, *target*, *map*, and *modulus*, together with the two continuation ledgers. The graph ledger and the first three unary fields form $\mathsf{ContinuousModulusWitness}(source, map, target)$. Unary closure along the target-modulus ledger supplies the carrier for *cert*, and this packages the second witness $\mathsf{ContinuousModulusWitness}(target,$ □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier graph modulus cont readback`

Theorem 59.66 (Continuous function carrier endpoint-certificate cycle tails)**.** *For histories source, map, target, m a continuous-function carrier whose source is history-same to its certificate has empty graph tail and empty modulus tail:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{hsame}(source, cert) \Longrightarrow \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}($$

Proof. Unpack the carrier into its graph ledger $\mathsf{Cont}(source, map, target)$ and certificate ledger $\mathsf{Cont}(target, modulus, cert)$. Transport the certificate ledger across $\mathsf{hsame}(source, cert)$, obtaining $\mathsf{Cont}(target, modulus, source)$. The two ledgers form a continuation cycle, so mutual-extension cancellation forces both tails, *map* and *modulus*, to be history-same to e. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier endpoint cert cycle tails empty`

Theorem 59.67 (Continuous function carrier endpoint-certificate cycle exactness)**.** *For histories source, map, target, modulus, cert, if a continuous-function carrier closes its public source against its endpoint certificate, then the carrier is an identity presentation at that endpoint:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{hsame}(source, cert) \Longrightarrow \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e}) \wedge \mathsf{hsame}(target, source) \wedge \mathsf{hsame}(cert, source).$$

Proof. Apply Definition 59.140 to the carrier and the displayed endpoint certificate equality. This gives $\mathsf{hsame}(map, \mathsf{e})$ and $\mathsf{hsame}(modulus, \mathsf{e})$.

Unpack the graph ledger $\mathsf{Cont}(source, map, target)$ from the carrier. Transport that ledger across $\mathsf{hsame}(map, \mathsf{e})$ to obtain $\mathsf{Cont}(source, \mathsf{e}, target)$. The right-unit clause of Definition 21.12 supplies $\mathsf{Cont}(source, \mathsf{e}, source)$, and continuation determinacy from Definition 21.7 identifies *target* with *source*.

Finally, symmetry in Definition 3.8 turns the displayed $\mathsf{hsame}(source, cert)$ into $\mathsf{hsame}(cert, source)$, and the four pieces assemble the stated conjunction. □

The ten-field fit is: traditional target, identity-cycle exactness for continuous-function carriers; BEDC kernel, $BHist$; ledger relation, Cont; carrier, ContinuousFunctionCarrier; classifier, hsame; ledger fields, *source*, *map*, *target*, *modulus*, *cert*; core theorem, Definition 59.141; exactness, the four displayed hsame rows; standard bridge, deferred; downstream consumer, ContinuousMapUp identity and category normalization.

Theorem 59.68 (Continuous function carrier composite endpoint-certificate cycle exactness)**.** *For histories source, middle, target, f, g, fg, modF, modG, modFG, certF, certG, cert, closed composition of continuous-function carriers inherits endpoint-certificate cycle collapse:*

$$\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF) \wedge \mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, c$$
$$\mathsf{Cont}(f, g, fg) \wedge \mathsf{Cont}(modF, modG, modFG) \wedge \mathsf{Cont}(target, modFG, cert) \wedge \mathsf{hsame}(source, cert)$$
$$\Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG, cert) \wedge \mathsf{hsame}(fg, \mathsf{e}) \wedge \mathsf{hsame}(modFG, \mathsf{e}) \wedge \mathsf{hsame}(targ$$

Proof. First apply Definition 59.7 to the two given carriers, the displayed composite map and modulus continuations, and the target-to-certificate continuation. This constructs $\mathsf{ContinuousFunctionCarrier}(source, fg,$

Next apply Definition 59.141 to this composite carrier and the displayed endpoint-certificate equality $\mathsf{hsame}(source, cert)$. The resulting four history-sameness conclusions, together with the composite carrier just constructed, assemble the stated conjunction. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composite endpoint cert cycle exactness`

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composite endpoint cycle exactness`

Theorem 59.69 (Continuous function carrier prefix exactness)**.** *For histories $p, source, map, target, modulus, cert$ prefixing the visible source, target, and certificate ledgers is conservative for the continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus, p \cdot cert) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).$$

Proof. For the forward direction, unpack the prefixed carrier. It supplies unary histories for $p \cdot source$, $p \cdot target$, map, and $modulus$, together with the two continuation ledgers $\mathsf{Cont}(p \cdot source, map, p \cdot target)$ and $\mathsf{Cont}(p \cdot target, modulus, p \cdot cert)$.

Unary append factorization applied to the source ledger recovers $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(source)$. Applied to the target ledger, it recovers $\mathsf{UnaryHistory}(target)$. Prefix cancellation on the graph continuation ledger gives $\mathsf{Cont}(source, map, target)$, and prefix cancellation on the modulus-certificate continuation ledger gives $\mathsf{Cont}(target, modulus, cert)$. Repacking these witnesses gives $\mathsf{ContinuousFunctionCarrier}(s$

For the reverse direction, project $\mathsf{UnaryHistory}(p)$ and the unprefixed carrier from the conjunction. Apply Theorem 59.20 to obtain the displayed prefixed carrier.

The ten-field certificate fit is the following. The traditional target is prefix-invariant continuous graph evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is exact prefix cancellation for graph and modulus-certificate ledgers. The ledger consists of source, map, target, modulus, and certificate histories. The core theorem is prefix exactness, the exactness predicate is the displayed iff, the standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefix iff`

Theorem 59.70 (Continuous function carrier modulus suffix exactness)**.** *For histories $p, source, map, target, modu$ suffixing the modulus and certificate ledgers by the same unary history is conservative for the continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus \cdot p, cert \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).$$

Proof. For the forward direction, unpack the suffixed carrier. The target-modulus certificate leg is a continuous modulus witness from $target$ through $modulus \cdot p$ to $cert \cdot p$, with the certificate unary witness obtained from unary continuation closure. Apply the continuous modulus witness suffix exactness theorem to remove the common right suffix, then repack the graph carrier fields.

For the reverse direction, turn the unsuffixed target-modulus certificate leg into a continuous modulus witness, apply suffix exactness to append p to the modulus and certificate histories, and repack the original graph fields with the suffixed certificate leg. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus suffix iff`

Theorem 59.71 (Continuous function carrier modulus-suffix target-certificate determinacy)**.** *For histories $q, source, map, target, target', modulus, cert, cert'$, if two continuous-function carrier witnesses share the same source, map, and visible modulus suffix,*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus{\cdot}q, cert{\cdot}q) \quad \textit{and} \quad \mathsf{ContinuousFunctionCarrier}(source, map, t$$

then their central target and certificate ledgers are determined:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply modulus suffix exactness to each carrier witness. This removes the common suffix from the modulus and certificate ledgers and exposes two central continuous-function carriers over the same source, map, and modulus. The target-certificate determinacy theorem for continuous-function carriers then identifies the central targets and the central certificates. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus suffix target cert deterministic`

Theorem 59.72 (Continuous function carrier prefixed graph-chain factorization)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta, cert$, if δ_1 and δ_2 are unary, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

holds, then there is a middle history middle such that

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert) \wedge \mathsf{ContinuousModulusWitness}(target, \delta_1, middle) \wedge \mathsf{Continuous}$$

Proof. Apply prefix exactness for continuous-function carriers to remove the common visible prefix from the source, target, and certificate ledgers. This recovers the unprefixed carrier and in particular the target-to-certificate continuation $\mathsf{Cont}(target, \delta, cert)$.

Choose $middle = target \cdot \delta_1$. The first modulus witness is the canonical continuation from $target$ through δ_1 to this middle history. For the second witness, combine $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ with the recovered target-to-certificate continuation and reassociate the BEDC append spine, obtaining $\mathsf{Cont}(middle, \delta_2, cert)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain factorizes`

Theorem 59.73 (Continuous function carrier prefixed graph-chain first factorization)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta, cert$, if δ_1 and δ_2 are unary, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

holds, then there is a middle history middle such that

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta_1, middle) \wedge \mathsf{ContinuousModulusWitness}(middle, \delta_2, cert).$$

Proof. Apply prefix exactness for continuous-function carriers to remove the common visible prefix. This exposes the unprefixed graph ledger $\mathsf{Cont}(source, map, target)$ and the composite target-certificate ledger $\mathsf{Cont}(target, \delta, cert)$.

Choose $middle = target \cdot \delta_1$. The first-stage carrier uses the original source, map, and target carrier data, the graph ledger, the unary witness for δ_1, and the canonical continuation $\mathsf{Cont}(target, \delta_1, middle)$. The second-stage modulus witness uses the unary closure of $middle$, the unary witness for δ_2, unary closure of $cert$, and associativity of BEDC append to factor the composite ledger through $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain factorization`

Theorem 59.74 (Continuous function carrier triple composition certificate synchronization)**.** *For BHist histories* $source, mid_1, mid_2, target, f, g, h, fg, gh, map_L, map_R, mod_F, mod_G, mod_H, mod_{FG}, mod_{GH}, mod_L,$ *suppose the three one-step carriers*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\&\mathsf{ContinuousFunctionCarrier}(mid_2, h, target, mod_H, cert_H)\end{aligned}$$

hold. Suppose also that the left-bracketed ledgers

$$\begin{array}{lll}\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\ \mathsf{Cont}(fg, h, map_L), & \mathsf{Cont}(mod_{FG}, mod_H, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)\end{array}$$

hold, and that the right-bracketed ledgers

$$\begin{array}{lll}\mathsf{Cont}(g, h, gh), & \mathsf{Cont}(mod_G, mod_H, mod_{GH}), & \mathsf{Cont}(target, mod_{GH}, cert_{GH}),\\ \mathsf{Cont}(f, gh, map_R), & \mathsf{Cont}(mod_F, mod_{GH}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)\end{array}$$

hold. Then the two bracketings both carry the original source to the final target and their public composite data agree up to history-sameness:

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)\wedge\\&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)\wedge\\&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).\end{aligned}$$

Proof. First build the left bracketing. Theorem 59.7 composes the $source$-f-mid_1 carrier with the mid_1-g-mid_2 carrier using the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Applying the same closure theorem to that composite and the mid_2-h-$target$ carrier gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod$

The right bracketing is built in the symmetric order. Compose the g and h carriers through gh, mod_{GH}, and $cert_{GH}$, then compose the resulting mid_1-gh-$target$ carrier after the original f carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For map synchronization, apply continuation associativity up to history-sameness to the four graph ledgers $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, map_L)$, $\mathsf{Cont}(g, h, gh)$, and $\mathsf{Cont}(f, gh, map_R)$. The result is $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same continuation-associativity principle to the modulus ledgers $\mathsf{Cont}(mod_F, mod_G, mod_{FG})$, $\mathsf{Cont}(mod_{FG}, mod_H, mod_L)$, $\mathsf{Cont}(mod_G, mod_H, mod_{GH})$, and $\mathsf{Cont}(mod_F, mod_{GH}, mod_R)$. This gives $\mathsf{hsame}(mod_L, mod_R)$.

Finally compare the two target-certificate ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$. Continuation respect for hsame, with reflexivity on $target$ and the recovered modulus sameness, yields $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is threefold composition of continuous function carriers. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is bracketing synchronization for graph, modulus, and certificate ledgers. The ledger consists of source, intermediate, target, map, modulus, and certificate histories. The core theorem is this triple composition synchronization result. The exactness predicate is the displayed pair of carriers together with the three hsame conclusions. The standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composition associative certificate synchronization`

Theorem 59.75 (Continuous function carrier fourfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, target, f, g, h, k, fg, fgh, hk, ghk, map_L, map_R, mod_F, mod_G, mod_H, mod_K, mod_{FG},$ *suppose the four one-step carriers*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\&\mathsf{ContinuousFunctionCarrier}(mid_3, k, target, mod_K, cert_K)\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\ \mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\ \mathsf{Cont}(fgh, k, map_L), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}\mathsf{Cont}(h, k, hk), & \mathsf{Cont}(mod_H, mod_K, mod_{HK}), & \mathsf{Cont}(target, mod_{HK}, cert_{HK}),\\ \mathsf{Cont}(g, hk, ghk), & \mathsf{Cont}(mod_G, mod_{HK}, mod_{GHK}), & \mathsf{Cont}(target, mod_{GHK}, cert_{GHK}),\\ \mathsf{Cont}(f, ghk, map_R), & \mathsf{Cont}(mod_F, mod_{GHK}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)\wedge\\&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)\wedge\\&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then again with the k carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L,$

The right-deep carrier is obtained by the symmetric closure chain. Compose the h and k carriers through hk, mod_{HK}, and $cert_{HK}$; compose the g carrier with that composite through ghk, mod_{GHK}, and $cert_{GHK}$; then compose the f carrier with the resulting carrier to obtain $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply the BEDC fourfold continuation associativity theorem cont_assoc_four to the common graph spine f, g, h, k. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, and $\mathsf{Cont}(fgh, k, map_L)$, while the right spine is presented by $\mathsf{Cont}(h, k, hk)$, $\mathsf{Cont}(g, hk, ghk)$, and $\mathsf{Cont}(f, ghk, map_R)$. The underlying append-associativity spine append_assoc yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fourfold continuation associativity theorem to $mod_F, mod_G, mod_H, mod_K$. The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common append spine yields $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the certificate ledger to give $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is fourfold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the graph, modulus, and certificate histories displayed above. The stability field is fourfold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp associativity coherence and the later category bridge. □

Theorem 59.76 (Continuous function carrier fivefold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, target, f, g, h, k, l, fg, fgh, fghk, kl, hkl, ghkl, map_L, map_R, mod_F, mod_G, mod$ *suppose the five one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\
&\mathsf{ContinuousFunctionCarrier}(mid_4, l, target, mod_l, cert_l)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}),\\
\mathsf{Cont}(fghk, l, map_L), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(k, l, kl), & \mathsf{Cont}(mod_K, mod_l, mod_{KL}), & \mathsf{Cont}(target, mod_{KL}, cert_{KL}),\\
\mathsf{Cont}(h, kl, hkl), & \mathsf{Cont}(mod_H, mod_{KL}, mod_{HKL}), & \mathsf{Cont}(target, mod_{HKL}, cert_{HKL}),\\
\mathsf{Cont}(g, hkl, ghkl), & \mathsf{Cont}(mod_G, mod_{HKL}, mod_{GHKL}), & \mathsf{Cont}(target, mod_{GHKL}, cert_{GHKL}),\\
\mathsf{Cont}(f, ghkl, map_R), & \mathsf{Cont}(mod_F, mod_{GHKL}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then with the k carrier, and finally with the l carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is obtained by the matching closure chain. Compose the k and l carriers through kl, mod_{KL}, and $cert_{KL}$; compose the h carrier with that composite through hkl, mod_{HKL}, and $cert_{HKL}$; compose the g carrier with the resulting carrier through $ghkl$, mod_{GHKL}, and $cert_{GHKL}$; then compose the f carrier with that final right-deep carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_five to the common graph spine f, g, h, k, l. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, $\mathsf{Cont}(fgh, k, fghk)$, and $\mathsf{Cont}(fghk, l, map_L)$, while the right spine is presented by $\mathsf{Cont}(k, l, kl)$, $\mathsf{Cont}(h, kl, hkl)$, $\mathsf{Cont}(g, hkl, ghkl)$, and $\mathsf{Cont}(f, ghkl, map_R)$. The fivefold continuation reassociation gives $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fivefold reassociation theorem to $mod_F, mod_G, mod_H, mod_K, mod_l$. The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common continuation spine gives $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the terminal certificate ledger and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is pentagon-style fivefold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the five one-step graph histories, their modulus histories, their target-certificate histories, and the displayed left-deep and right-deep composite ledgers. The stability field is fivefold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp pentagon coherence and the category bridge. □

Theorem 59.77 (Continuous function carrier sixfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, mid_5, target, f, g, h, k, l, m, fg, fgh, fghk, fghkl, lm, klm, hklm, ghklm, map_L$, *suppose the six one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\
&\mathsf{ContinuousFunctionCarrier}(mid_4, l, mid_5, mod_l, cert_l),\\
&\mathsf{ContinuousFunctionCarrier}(mid_5, m, target, mod_m, cert_m)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}),\\
\mathsf{Cont}(fghk, l, fghkl), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_{FGHKL}), & \mathsf{Cont}(mid_5, mod_{FGHKL}, cert_{FGHKL}),\\
\mathsf{Cont}(fghkl, m, map_L), & \mathsf{Cont}(mod_{FGHKL}, mod_m, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(l, m, lm), & \mathsf{Cont}(mod_l, mod_m, mod_{LM}), & \mathsf{Cont}(target, mod_{LM}, cert_{LM}),\\
\mathsf{Cont}(k, lm, klm), & \mathsf{Cont}(mod_K, mod_{LM}, mod_{KLM}), & \mathsf{Cont}(target, mod_{KLM}, cert_{KLM}),\\
\mathsf{Cont}(h, klm, hklm), & \mathsf{Cont}(mod_H, mod_{KLM}, mod_{HKLM}), & \mathsf{Cont}(target, mod_{HKLM}, cert_{HKLM}),\\
\mathsf{Cont}(g, hklm, ghklm), & \mathsf{Cont}(mod_G, mod_{HKLM}, mod_{GHKLM}), & \mathsf{Cont}(target, mod_{GHKLM}, cert_{GHKLM}),\\
\mathsf{Cont}(f, ghklm, map_R), & \mathsf{Cont}(mod_F, mod_{GHKLM}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)\wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)\wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Repeatedly apply Theorem 59.7 along the displayed fg, fgh, $fghk$, $fghkl$, and map_L graph ledgers, using the matching modulus and certificate-continuation ledgers at each stage. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is built from the other end of the same chain. Compose the l and m carriers, then compose successively with the k, h, g, and f carriers through the displayed lm, klm, $hklm$, $ghklm$, and map_R ledgers. The result is $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_six to the common graph spine f, g, h, k, l, m. The left-deep spine ends at map_L and the right-deep spine ends at map_R, so sixfold continuation associativity yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same kernel theorem to $mod_F, mod_G, mod_H, mod_K, mod_l, mod_m$. The displayed modulus ledgers present the same sixfold spine with endpoints mod_L and mod_R, giving $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ share the same target source. Continuation respect for hsame, with reflexivity on $target$ and the recovered modulus sameness, identifies the terminal certificates and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the same concrete carrier ledger as in the fourfold and fivefold synchronization results. The stability field is sixfold left-right bracketing synchronization, and the exactness field is the pair of constructed carriers together with the three hsame conclusions. □

Theorem 59.78 (Continuous function carrier certificate semantic naming certificate)**.** *For BHist histories $source, map, target, modulus, cert$, any carrier witness* $\mathsf{ContinuousFunctionCarrier}(source, map, target, mo$ *induces the semantic naming certificate*

$$\mathsf{SemanticNameCert}(C, C, C, \mathsf{hsame}), \qquad C(c) := \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, c).$$

Thus the certificate-slot carrier, pattern, and ledger are all the displayed fiber $c \mapsto$ $\mathsf{ContinuousFunctionCarrier}(source$ *and the classifier is* hsame.

Proof. The displayed carrier gives the source witness for the semantic certificate at the point $cert$. This is exactly the inhabited source field for the fiber C.

The classifier fields are the standard hsame rows. Reflexivity classifies each displayed certificate with itself, symmetry reverses a displayed classification, and transitivity composes two displayed classifications.

Carrier transport is confined to the certificate slot. Keep $source$, map, $target$, and $modulus$ fixed by hsame reflexivity, use the supplied hsame comparison of certificate histories, and apply Definition 59.14 to transport the continuous function carrier along that comparison.

The pattern and ledger predicates are the same fiber C as the source predicate. Their soundness and completeness maps are therefore identity projections, so the semantic naming certificate is repacked from the preceding fields. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier certificate semanticNameCert`

Theorem 59.79 (Continuous function carrier parallel composite determinacy)**.** *For BHist histories $source, middle, target, f, g, fg, fg', mod_F, mod_G, mod_{FG}, mod'_{FG}, cert_F, cert_G, cert, cert'$, suppose*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, middle, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(middle, g, target, mod_G, cert_G),\\
&\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(f, g, fg'),\\
&\mathsf{Cont}(mod_F, mod_G, mod_{FG}), \quad \mathsf{Cont}(mod_F, mod_G, mod'_{FG}),\\
&\mathsf{Cont}(target, mod_{FG}, cert), \quad \mathsf{Cont}(target, mod'_{FG}, cert')
\end{aligned}$$

hold. Then the two exposed composites are both continuous function carriers and their graph, modulus, and certificate histories are determined:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, fg, target, mod_{FG}, cert) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, fg', target, mod'_{FG}, cert') \wedge\\
&\mathsf{hsame}(fg, fg') \wedge \mathsf{hsame}(mod_{FG}, mod'_{FG}) \wedge \mathsf{hsame}(cert, cert').
\end{aligned}$$

Proof. First compose through the first visible ledger. The two given base carriers, together with $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(mod_F, mod_G, mod_{FG})$, and $\mathsf{Cont}(target, mod_{FG}, cert)$, satisfy the hypotheses of Theorem 59.7. This gives $\mathsf{ContinuousFunctionCarrier}(source, fg, target, mod_{FG}, cert)$.

The second displayed ledger has the same two base carriers. Applying the same composition theorem to $\mathsf{Cont}(f, g, fg')$, $\mathsf{Cont}(mod_F, mod_G, mod'_{FG})$, and $\mathsf{Cont}(target, mod'_{FG}, cert')$ gives $\mathsf{ContinuousFunctionCarrier}(source, fg', target, mod'_{FG}, cert')$.

Map determinacy is local to the shared graph rows. Continuation determinacy for the two rows $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g, fg')$ gives $\mathsf{hsame}(fg, fg')$; the same kernel determinacy applied to the two modulus rows gives $\mathsf{hsame}(mod_{FG}, mod'_{FG})$.

Finally compare the terminal certificate rows. They share the target history, and the recovered modulus sameness transports the row $\mathsf{Cont}(target, mod'_{FG}, cert')$ back across $\mathsf{Cont}(target, mod_{FG}, cert)$, yielding $\mathsf{hsame}(cert, cert')$. These five outputs are exactly the two composite carriers plus the three exposed history-sameness conclusions. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier parallel composite determinacy`

Proposition 59.80 (Uniform continuity modulus restriction along subset inclusion)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, and let*

$$i : U \subseteq T$$

be a certified inclusion, meaning that each membership ledger $\nu : U(u)$ is transported to a membership ledger $i(\nu) : T(u)$. Let $f : T \to Z$ be a certified graph witness with totality and single-valuedness as in Definition 59.1, and let $f\restriction_U$ be the same graph relation read with the source classifier restricted through i. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f on T, then the same δ satisfies the modulus-validity field for $f\restriction_U$ on U.

Proof. Fix a target precision request n and the source precision returned by the unary ledger δ. To verify the restricted modulus field, take source histories u_0, u_1 admitted by U, graph outputs z_0, z_1 for $f\restriction_U$, and the source closeness judgment required at the returned source precision.

The inclusion certificate transports the two source-admission ledgers:

$$\nu_j : U(u_j) \Longrightarrow i(\nu_j) : T(u_j) \qquad (j = 0, 1).$$

By Definition 59.93 and Definition 59.122, restricting the source subset does not alter the graph relation, the codomain classifier ledger, the precision ledger, or the modulus ledger. Thus the graph witnesses for $f\restriction_U$ are the same graph witnesses for f after the source admissions are read in T.

Apply the modulus-validity field for f on T to the transported memberships, the same source closeness judgment, and the same two graph outputs. It yields the target metric closeness conclusion at precision n. The well-formedness and representative stability of this conclusion are exactly the classifier and stability data of Definition 57.4 and Definition 57.5.

This is the entire obligation for the restricted witness: only the source-admission hypotheses have been narrowed from T to U, and those hypotheses enter the original modulus field through the certified inclusion. The precision ledger δ is unchanged. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction modulus chain`

Theorem 59.81 (Continuous empty-source restriction public package)**.** *Let restricted, source, map, target, target′,* BHist *be histories. Assume*

$$\mathsf{Cont}(\mathit{restricted}, \mathsf{e}, \mathit{source}),$$

$$\mathsf{ContinuousFunctionCarrier}(\mathit{source}, \mathit{map}, \mathit{target}, \mathit{oldMod}, \mathit{oldCert}),$$

$$\mathsf{ContinuousModulusChain}(\mathit{target}, \delta_1, \delta_2, \mathit{cert}),$$

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

and a displayed restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}', \delta, \mathit{cert}').$$

Then the empty-source restriction has the public package

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert}),$$

$$\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert}),$$

and

$$\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}').$$

Proof. Apply the preceding checked empty-source restriction-chain theorem to the source restriction, the original carrier, the modulus chain, and the composite modulus relation. This produces the restricted ContinuousFunctionCarrier over $(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert})$ and the ContinuousModulusWitness over $(\mathit{target}, \delta, \mathit{cert})$.

It remains to read back the displayed public target and certificate. The Lean-side theorem BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier_empty_source_restriction_public_readback applies to the same hypotheses together with the displayed restricted carrier, and returns $\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$.

Ten-field fit: traditional target restriction of uniformly continuous maps along an empty-source inclusion; BEDC source BHist, Cont, ContinuousFunctionCarrier, and ContinuousModulusChain; carrier the restricted ContinuousFunctionCarrier; classifier hsame on target and certificate histories; stability hsame transport and modulus-chain composition; ledger unchanged graph with replaced modulus/certificate chain; core theorem this public package; exactness the target/certificate readback; standard bridge ordinary restricted-map readback; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public readback`

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public package`

Theorem 59.82 (Uniform continuity modulus restriction is transitive)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let V, U, and T be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusions*

$$j : V \subseteq U$$

and

$$i : U \subseteq T.$$

For a certified graph witness $f : T \to Z$, let $f \restriction_U$ be the restriction along i, let $(f \restriction_U) \restriction_V$ be the successive restriction along j, and let $f \restriction_V$ be the direct restriction along the composite inclusion

$i \circ j : V \subseteq T$. The composite inclusion carries the same history-level graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger as the successive restriction. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f over T, then δ satisfies that field for both $f\restriction_V$ and $(f\restriction_U)\restriction_V$, and the two restricted witnesses are related by the $\mathsf{ContinuousMap}^{\uparrow}$ *classifier of Definition 59.94.*

Proof. Apply Proposition 59.80 first to $i : U \subseteq T$. This transports the modulus-validity field for f over T to the restriction $f\restriction_U$ over U, with the graph relation and all precision data left unchanged.

Apply the same proposition to $j : V \subseteq U$ and the already restricted witness $f\restriction_U$. The result is the modulus-validity field for $(f\restriction_U)\restriction_V$ over V, with the same history-level graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger.

Apply Proposition 59.80 once more to the composite inclusion $i \circ j : V \subseteq T$. This gives the direct restricted witness $f\restriction_V$ and its modulus-validity field over V using the original witness f and the same unary precision ledger δ.

It remains to compare the direct and successive restrictions. By Definition 59.93, restriction changes only the source-admission reading of the graph; by Definition 59.122, the graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger are recorded as the same ledgers. Thus the direct and successive witnesses present the same carrier data to the classifier, and Definition 59.94 relates them as $\mathsf{ContinuousMap}^{\uparrow}$ witnesses.

Ten-field fit: traditional target functorial restriction of uniformly continuous maps; BEDC source BHist with certified source predicates and ContinuousFunctionCarrier; carrier restricted graph witness; classifier ContinuousMap classifier; stability nested source-admission transport; ledger unchanged graph/modulus ledgers; core theorem this block; exactness direct-versus-successive restriction agreement; standard bridge restriction functoriality for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction transitive package`

Theorem 59.83 (Uniform continuity restriction functorial action)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let T, U, and V be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92. Let*

$$e_T : T \subseteq T$$

be the identity inclusion, and let

$$i : U \subseteq T$$

and

$$j : V \subseteq U$$

be certified inclusions. Let $f : T \to Z$ be a certified graph witness whose unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 over T. Then restriction along e_T preserves δ and is related to f by the $\mathsf{ContinuousMap}^{\uparrow}$ *classifier of Definition 59.94. Moreover, the successive restriction $(f\restriction_U)\restriction_V$ and the direct restriction along the certified composite inclusion $i \circ j : V \subseteq T$ are related by the same classifier and preserve the same graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger.*

Proof. For the identity clause, apply Proposition 59.80 to the certified inclusion $e_T : T \subseteq T$. The source-admission transport is the identity transport on the T ledger, so the restricted witness reads the same histories as f.

The unchanged ledger clauses of Definition 59.93 and Definition 59.122 keep the graph relation, codomain classifier ledger, unary precision ledger δ, and modulus ledger fixed. Hence the two witnesses present the same carrier data, and Definition 59.94 relates the identity restriction to f as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

For the composition clause, apply Theorem 59.82 to the inclusions $j : V \subseteq U$ and $i : U \subseteq T$. It gives the modulus-validity field for both the successive and direct restrictions with the same unary precision ledger δ.

The same theorem also identifies the direct and successive restrictions at the level of the $\mathsf{ContinuousMap}^{\uparrow}$ classifier. Its comparison uses exactly the shared graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger supplied by the restriction policy.

Ten-field fit: traditional target restriction action of uniformly continuous maps; BEDC source BHist source predicates plus ContinuousFunctionCarrier; carrier restricted graph witness; classifier ContinuousMapUp; stability identity and composite source-admission transport; ledger unchanged graph/modulus data; core theorem this action law; exactness identity and composition agreement; standard bridge restriction of uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction functorial action`

Theorem 59.84 (Uniform continuity restriction commutes with terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$source, map, target, modulus, cert, extra, modulus', cert' : \mathsf{BHist}$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the original unary modulus, the terminal certificate, the additional terminal modulus, the composite modulus, and the terminal certificate after composition. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert),\\&\mathsf{ContinuousModulusWitness}(cert, extra, cert'),\\&\mathsf{Cont}(modulus, extra, modulus').\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting f along i and then applying terminal modulus extension with extra, modulus′, and cert′ is related by the classifier of Definition 59.94 to the witness obtained by first applying terminal modulus extension over T and then restricting along i. Both witnesses present the restricted graph $f\restriction_U$, the source-admission transport through i, the codomain classifier ledger for Z, the composite modulus ledger modulus′, and the terminal certificate ledger cert′.

Proof. Apply Theorem 59.162 to the displayed carrier, the terminal modulus witness, and the composite continuation ledger. This gives a continuous-function carrier over T with histories $source, map, target, modulus', cert'$.

Restrict this extended carrier along $i : U \subseteq T$. Proposition 59.80 transports the modulus-validity field to U, while Definition 59.93 and Definition 59.122 keep the graph relation, target classifier, precision ledger, and modulus ledger fixed.

For the other route, first apply Proposition 59.80 to the original witness f. The restricted witness over U carries the same graph ledger map, the same target ledger $target$, and the same modulus and certificate ledgers $modulus, cert$, with only the source-admission hypotheses narrowed through i.

Now apply Theorem 59.162 to this restricted carrier using the same $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$ and the same $\mathsf{Cont}(modulus, extra, modulus')$. The resulting restricted-then-extended witness has composite modulus $modulus'$ and terminal certificate $cert'$.

It remains to compare the two route results. Their graph relation is the same relation read on U, source histories admitted by U are transported to T by the same inclusion certificate, the codomain classifier ledger is the same Z ledger, and the composite modulus and terminal certificate ledgers are the same displayed histories. Therefore Definition 59.94 relates the two witnesses as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

Ten-field fit: traditional target commutation of source restriction with terminal modulus extension; BEDC source BHist with certified source predicates, Cont, and ContinuousModulusWitness; carrier ContinuousFunctionCarrier with restricted graph witness; classifier ContinuousMap classifier; stability source-admission transport plus terminal modulus preservation; ledger unchanged graph relation, codomain classifier ledger, composite modulus ledger, and terminal certificate ledger; core theorem this block; exactness equality of the two route witnesses under Definition 59.94; standard bridge restriction and modulus-normalization order independence for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction terminal modulus package`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction terminal modulus commutes`

Theorem 59.85 (Uniform continuity restriction commutes with iterated terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$source, map, target, modulus, cert, extra_0, modulus_0, cert_0, extra_1, modulus_1, cert_1 : \mathsf{BHist}$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the initial unary modulus, the initial terminal certificate, the first added terminal modulus, the first composite modulus, the first terminal certificate after composition, the second added terminal modulus, the second composite modulus, and the final terminal certificate. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert),\\
&\mathsf{ContinuousModulusWitness}(cert, extra_0, cert_0), \quad \mathsf{Cont}(modulus, extra_0, modulus_0),\\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{Cont}(modulus_0, extra_1, modulus_1).
\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting f along i and then applying the two terminal modulus extensions with final histories $modulus_1, cert_1$ is related by the classifier of Definition 59.94 to the witness obtained by applying the two terminal modulus extensions over T and then restricting along i. Both routes expose the restricted graph ledger map, the transported source-admission ledger through i, the same target classifier ledger $target$, and final histories $modulus_1, cert_1$.

Proof. Apply Theorem 59.84 to the initial carrier, the witness $\mathsf{ContinuousModulusWitness}(cert, extra_0, cert_0)$, and the continuation ledger $\mathsf{Cont}(modulus, extra_0, modulus_0)$. This relates the route that restricts first and then performs the first terminal extension to the route that performs the first terminal extension over T and then restricts. Both resulting representatives carry restricted graph ledger map, transported source-admission through i, target ledger $target$, and histories $modulus_0, cert_0$.

Apply Theorem 59.162 to the initial carrier and the first terminal data. It keeps the carrier over T live after the first extension, now with histories $source, map, target, modulus_0, cert_0$. Applying the same theorem to the restricted representative supplied by the first commutation keeps the restricted carrier live with the same graph and target ledgers and with histories $modulus_0, cert_0$.

Now apply Theorem 59.84 to these first-extended carriers, using $\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1)$ and $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$. This relates the route that restricts before the second terminal extension to the route that performs the second terminal extension over T before restriction. The two representatives therefore expose the same restricted graph ledger, source-admission transport, target classifier ledger, and final histories $modulus_1, cert_1$.

The first commutation gives a classifier relation between the two intermediate representatives, and the second commutation gives a classifier relation between the two final representatives built from those intermediates. By the transitivity clause in Definition 59.94, the endpoints are related as $\mathsf{ContinuousMap}^{\uparrow}$ witnesses: restricting first and applying both terminal extensions is classified together with extending twice over T and then restricting to U.

Ten-field fit: traditional target finite terminal-modulus normalization under source restriction; BEDC source BHist with certified source predicates, Cont, and two $\mathsf{ContinuousModulusWitness}$ ledgers; carrier $\mathsf{ContinuousFunctionCarrier}$ preserved through two terminal extensions; classifier ContinuousMap classifier with transitivity; stability transported source admission plus preserved modulus-validity fields; ledger restricted graph ledger, target classifier ledger, and final histories $modulus_1, cert_1$; core theorem this block; exactness equality of the two finite route endpoints under Definition 59.94; standard bridge finite modulus-normalization stability for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction iterated terminal modulus commutes`

Theorem 59.86 (Uniform continuity restriction commutes with visible terminal modulus extension)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let*

$$p, q, source, map, target, modulus, cert, extra, modulus', cert' : \mathsf{BHist}$$

be histories, and assume the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q),$$

the terminal modulus witness

$$\mathsf{ContinuousModulusWitness}(cert, extra, cert'),$$

and the terminal modulus composition ledger

$$\mathsf{Cont}(modulus, extra, modulus').$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ *witness obtained by restricting the visible carrier along* i *and then applying visible terminal modulus extension with* $extra$, $modulus'$, *and* $cert'$ *is related by the classifier of Definition 59.94 to the witness obtained by first applying visible terminal modulus extension and then restricting along* i. *Both routes expose the restricted graph ledger map, the same source-admission transport through* i, *the visible target ledger* $p \cdot target$, *the composite modulus ledger* $modulus' \cdot q$, *and the terminal certificate ledger* $(p \cdot cert') \cdot q$.

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction visible terminal modulus commutes`

Proof. Peel the visible carrier by Theorem 59.169. This recovers unary context histories for p and q together with the central carrier

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).$$

Apply Theorem 59.84 to this central carrier, the source inclusion $i : U \subseteq T$, the witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$, and the ledger $\mathsf{Cont}(modulus, extra, modulus')$. The central endpoints are classified together: one restricts first and then extends the terminal modulus, and the other extends over T first and then restricts.

Reattach the visible prefix and suffix to the two central endpoints. Theorem 59.163 supplies the visible terminal extension on the route whose extension is performed after the visible carrier is restricted, and the reverse use of Theorem 59.169 repacks the same unary histories around the endpoint obtained by extending first.

The visible-context exactness theorem identifies the exposed public ledgers on both routes. The graph ledger is the restricted reading of map, the source-admission hypotheses are transported through the same inclusion certificate i, the target ledger is $p \cdot target$, and the final ledgers are $modulus' \cdot q$ and $(p \cdot cert') \cdot q$. Therefore the classifier of Definition 59.94 relates the two visible witnesses as $\mathsf{ContinuousMap}^{\uparrow}$ representatives.

Ten-field fit: traditional target commutation of source restriction with visible terminal modulus extension; BEDC source BHist with visible contexts, certified source predicates, Cont, and $\mathsf{ContinuousModulusWitness}$; carrier visible $\mathsf{ContinuousFunctionCarrier}$ with restricted graph witness; classifier ContinuousMap classifier; stability source-admission transport plus visible terminal modulus preservation; ledger restricted graph relation, visible target ledger, composite modulus ledger, and visible terminal certificate ledger; core theorem this block; exactness equality of the two visible route endpoints under Definition 59.94; standard bridge restriction and visible modulus-normalization order independence for uniformly continuous maps; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus commutes`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible empty source restriction terminal modulus commutes`

Theorem 59.87 (Uniform continuity visible restriction commutes with modulus chains)**.** *Let*

$$p, q, restricted, source, map, target, target', oldMod, oldCert, \delta_1, \delta_2, \delta, cert, cert' : \mathsf{BHist}$$

be histories. Assume the source restriction ledger

$$\mathsf{Cont}(restricted, \mathsf{e}, source),$$

the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{source}, \mathit{map}, p \cdot \mathit{target}, \mathit{oldMod} \cdot q, (p \cdot \mathit{oldCert}) \cdot q),$$

the two-link modulus chain

$$\mathsf{ContinuousModulusChain}(\mathit{target}, \delta_1, \delta_2, \mathit{cert}),$$

the composite modulus ledger

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

and a displayed restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}', \delta \cdot q, (p \cdot \mathit{cert}') \cdot q).$$

Then the canonical visible restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}, \delta \cdot q, (p \cdot \mathit{cert}) \cdot q),$$

the central witness

$$\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert}),$$

and the readback

$$\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$$

hold.

Proof. Peel the displayed visible carrier by Theorem 59.169. This exposes the central carrier over $(\mathit{source}, \mathit{map}, \mathit{target}, \mathit{oldMod}, \mathit{oldCert})$ while retaining the same visible prefix p and suffix q.

Apply the checked central chain theorem recorded above, namely BEDC.Derived.ContinuousUp.ContinuousFunctio to the exposed carrier, the source restriction ledger, the displayed modulus chain, and the composite ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. It yields the central restricted carrier

$$\mathsf{ContinuousFunctionCarrier}(\mathit{restricted}, \mathit{map}, \mathit{target}, \delta, \mathit{cert})$$

and the witness $\mathsf{ContinuousModulusWitness}(\mathit{target}, \delta, \mathit{cert})$.

Reattach the visible prefix and suffix with the reverse direction of Theorem 59.169. The resulting public carrier is exactly

$$\mathsf{ContinuousFunctionCarrier}(p \cdot \mathit{restricted}, \mathit{map}, p \cdot \mathit{target}, \delta \cdot q, (p \cdot \mathit{cert}) \cdot q).$$

It remains to compare this canonical carrier with the displayed restricted carrier. The visible-context target-certificate determinacy theorem applies to the two carriers with the same visible source, graph, modulus, prefix, and suffix data, so it returns $\mathsf{hsame}(\mathit{target}, \mathit{target}') \wedge \mathsf{hsame}(\mathit{cert}, \mathit{cert}')$.

Ten-field fit: traditional target commutation of visible source restriction with finite modulus-chain normalization; BEDC source BHist with Cont, visible contexts, and ContinuousModulusChain; carrier visible ContinuousFunctionCarrier with restricted source; classifier hsame on target and certificate histories; stability central chain witness plus visible-context transport; ledger composite modulus δ with unchanged graph ledger; core theorem this visible chain replacement law; exactness target-certificate readback against any displayed restricted carrier; standard bridge visible uniformly continuous restriction under modulus-chain normalization; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction modulus chain commutes`

Theorem 59.88 (Uniform continuity restriction commutes with three-step terminal modulus reassociation)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra_3, modulus_3, cert_3, ext$$

be the BEDC histories recording the source ledger for T, the graph ledger of f, the codomain classifier ledger, the initial unary modulus and terminal certificate, the three terminal additions with their sequential output moduli and certificates, and the binary and ternary composite terminal ledgers. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus_0, cert_0),\\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2), \quad \mathsf{ContinuousM}\\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),\\
&\mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),\\
&\mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}).
\end{aligned}$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ *witness obtained by restricting f along i and applying the three terminal modulus extensions with final public histories $map, target, modulus_3, cert_3$ is related by the classifier of Definition 59.94 to the witness obtained by first reassociating the three terminal extensions over T and then restricting along i with final public histories $map, target, modulus_{123}, cert_3$. The comparison includes the terminal-modulus equality* $\mathsf{hsame}(modulus_3, modulus_{123})$*, and both classified endpoints expose the restricted graph ledger map, the source-admission transport through i, the same target ledger target, and the same terminal certificate ledger $cert_3$.*

Proof. First apply Theorem 59.85 to the initial carrier and the first two terminal extensions. This relates the route that restricts before the first two extensions to the route that performs those two extensions over T before restriction. Both intermediate representatives expose the restricted graph ledger *map*, the transported source-admission through i, the target ledger *target*, and terminal data $modulus_2, cert_2$.

Next keep the third extension live on both sides. Apply Theorem 59.84 to the two-step carrier over T using $\mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3)$ and $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$. This classifies the endpoint obtained by restricting first and then applying all three terminal extensions with the endpoint obtained by applying the sequential three-step extension over T and then restricting.

Now identify the sequential and reassociated terminal moduli over the unreduced source. Theorem 59.168, with the empty visible prefix and suffix, supplies the carrier with final data $modulus_3, cert_3$, the carrier with final data $modulus_{123}, cert_3$, and the comparison $\mathsf{hsame}(modulus_3, modulus_{123})$.

Restrict the reassociated carrier along $i : U \subseteq T$. Proposition 59.80 transports the modulus-validity field to U, and the ledger policy keeps the graph relation *map*, the target classifier ledger *target*, and the terminal certificate ledger $cert_3$ fixed.

Finally use classifier transitivity in Definition 59.94. The sequential endpoint is classified with the extend-over-T sequential endpoint by the preceding commutation steps, the terminal chapter

identifies that endpoint with the reassociated carrier through $\mathsf{hsame}(modulus_3, modulus_{123})$, and restriction of the reassociated carrier gives the displayed final representative. Target and certificate determinacy keep the exposed target and terminal certificate ledgers equal, so the two restricted endpoints are $\mathsf{ContinuousMap}^{\uparrow}$ representatives of the same public certificate.

Ten-field fit: traditional target finite three-step terminal-modulus normalization under source restriction; BEDC source BHist with certified source predicates, Cont, and three ContinuousModulusWitness ledgers; carrier ContinuousFunctionCarrier preserved through sequential and reassociated terminal extensions; classifier ContinuousMap classifier with transitivity; stability transported source admission plus preserved modulus-validity fields; ledger restricted graph ledger, target classifier ledger, final sequential and reassociated modulus ledgers, and terminal certificate ledger; core theorem this block; exactness equality of the two route endpoints plus $\mathsf{hsame}(modulus_3, modulus_{123})$; standard bridge finite modulus-normalization coherence for uniformly continuous maps; status ProofTarget. □

Theorem 59.89 (Uniform continuity visible restriction commutes with three-step terminal modulus reassociation)**.** *Let Y and Z be certified $\mathsf{Met}^{\uparrow}$ objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusion $i : U \subseteq T$. Let $f : T \to Z$ be a certified graph witness, and let*

$$p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra_3, modulus_3, cert_3$$

be the visible context histories, the source, graph, and target ledgers for f, the initial visible unary modulus and terminal certificate, the three terminal additions with their sequential output moduli and certificates, and the binary and ternary composite terminal ledgers. Assume that f satisfies the ContinuousMap carrier, totality, single-valuedness, and modulus-validity fields of Definition 59.1 and Definition 59.161 over T, and assume

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q),$$

the three terminal modulus witnesses

$$\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1), \quad \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2), \quad \mathsf{ContinuousM}$$

and the sequential and reassociated terminal modulus ledgers

$$\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),$$
$$\mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),$$
$$\mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}).$$

Then the $\mathsf{ContinuousMap}^{\uparrow}$ witness obtained by restricting the visible carrier along i and applying the three visible terminal modulus extensions with exposed public histories $map, p \cdot target, modulus_3 \cdot q, (p \cdot cert_3) \cdot q$ is related by the classifier of Definition 59.94 to the witness obtained by first reassociating the three terminal extensions over the central carrier and then restricting along i with exposed public histories $map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q$. The comparison includes the central readback $\mathsf{hsame}(modulus_3, modulus_{123})$, and both visible endpoints carry the same source-admission transport through i, the same graph ledger map, the same visible target ledger $p \cdot target$, and the same visible terminal certificate ledger $(p \cdot cert_3) \cdot q$.

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus reassociation`

Proof. Peel the visible carrier by Theorem 59.169. This exposes the central carrier

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus_0, cert_0)$$

under the same certified source inclusion $i : U \subseteq T$, while retaining the visible prefix p and suffix q for later reattachment.

Apply Theorem 59.88 to the central carrier, the three $\mathsf{ContinuousModulusWitness}$ rows, and the sequential and reassociated Cont rows. The central route that restricts first and then applies the three terminal extensions is classified with the route that first reassociates the three terminal extensions over T and then restricts, and the same step records $\mathsf{hsame}(modulus_3, modulus_{123})$.

Reattach the common visible prefix and suffix to the two central endpoints. The visible terminal-extension theorem supplies the route whose terminal extensions are performed after restriction, and the reverse direction of Theorem 59.169 repacks the endpoint obtained from the reassociated central carrier.

The visible-context exactness theorems identify the public ledgers after reattachment. The sequential endpoint exposes $p \cdot target$, $modulus_3 \cdot q$, and $(p \cdot cert_3) \cdot q$, while the reassociated endpoint exposes $p \cdot target$, $modulus_{123} \cdot q$, and the same visible terminal certificate $(p \cdot cert_3) \cdot q$.

Finally use classifier transitivity from Definition 59.94. The peeled central comparison, the two visible reattachments, and the visible exactness readbacks compose into a $\mathsf{ContinuousMap}^{\uparrow}$ classifier relation between the displayed public endpoints, with the central modulus comparison $\mathsf{hsame}(modulus_3, modulus_{123})$ carried along.

Ten-field fit: traditional target visible source restriction under three-step terminal-modulus reassociation; BEDC source BHist with visible contexts, certified source predicates, Cont, and three $\mathsf{ContinuousModulusWitness}$ rows; carrier visible $\mathsf{ContinuousFunctionCarrier}$ with restricted source admission; classifier ContinuousMap classifier with transitivity; stability visible-context peeling, central restriction reassociation, and visible reattachment; ledger restricted graph ledger, visible target ledger, sequential and reassociated visible modulus ledgers, and visible terminal certificate ledger; core theorem this visible reassociation statement; exactness public readback with $\mathsf{hsame}(modulus_3, modulus_{123})$; standard bridge public visible uniformly continuous restriction under three terminal modulus extensions; status ProofTarget. □

Theorem 59.90 (Uniform continuity visible restriction three-step terminal modulus normal form)**.** *Let*

$$p, q, restricted, source, map, target, target', modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra$$

be histories. Assume the source restriction ledger

$$\mathsf{Cont}(restricted, \mathsf{e}, source),$$

the visible continuous-function carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q),$$

the three terminal modulus witnesses

$$\begin{aligned}&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1),\\ &\mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2),\\ &\mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3),\end{aligned}$$

the sequential and reassociated terminal modulus ledgers

$$\mathsf{Cont}(modulus_0, extra_1, modulus_1), \quad \mathsf{Cont}(modulus_1, extra_2, modulus_2), \quad \mathsf{Cont}(modulus_2, extra_3, modulus_3),$$
$$\mathsf{Cont}(extra_1, extra_2, extra_{12}), \quad \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}),$$
$$\mathsf{Cont}(extra_{12}, extra_3, extra_{123}), \quad \mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}),$$

and the displayed restricted sequential carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot restricted, map, p \cdot target', modulus_3 \cdot q, (p \cdot cert') \cdot q).$$

Then the visible restricted carrier has the three-step terminal modulus normal form

$$\mathsf{ContinuousFunctionCarrier}(p \cdot restricted, map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q),$$

and the displayed sequential endpoint reads back to the same public data:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_3, cert') \wedge \mathsf{hsame}(modulus_3, modulus_{123}).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction terminal modulus normal form`

Proof. First apply Theorem 59.168 to the unreduced visible carrier, the three terminal witness rows, and the displayed sequential and reassociated Cont ledgers. It supplies the visible sequential carrier with final data $modulus_3, cert_3$, the visible reassociated carrier with final data $modulus_{123}, cert_3$, and the modulus comparison $\mathsf{hsame}(modulus_3, modulus_{123})$.

Restrict the reassociated carrier along the same source ledger $\mathsf{Cont}(restricted, \mathsf{e}, source)$. Proposition 59.80 transports the source-admission field, while the visible context and ledger policies preserve the graph ledger map, target ledger $p \cdot target$, reassociated modulus ledger $modulus_{123} \cdot q$, and certificate ledger $(p \cdot cert_3) \cdot q$.

Next compare the sequential and reassociated routes through Theorem 59.88, applied to the central histories under the same source restriction and terminal ledgers. This identifies the restricted sequential endpoint with the restricted reassociated endpoint and records the same terminal-modulus readback $\mathsf{hsame}(modulus_3, modulus_{123})$.

Finally apply Theorem 59.174 to the canonical restricted sequential carrier and the displayed restricted sequential carrier. They share the visible source $p \cdot restricted$, the graph ledger map, the visible modulus $modulus_3 \cdot q$, and the same visible prefix and suffix, so determinacy gives $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert_3, cert')$. Combining these with the modulus comparison yields the stated normal-form readback.

Ten-field fit: traditional target visible source restriction followed by three terminal modulus extensions; BEDC source BHist with visible contexts, source restriction ledger, Cont, and three $\mathsf{ContinuousModulusWitness}$ rows; carrier visible $\mathsf{ContinuousFunctionCarrier}$ with restricted source; classifier hsame target, certificate, and modulus readback; stability source-admission restriction plus terminal modulus reassociation; ledger restricted graph ledger, visible target ledger, sequential and reassociated modulus ledgers, and terminal certificate ledger; core theorem this normal-form statement; exactness comparison against any displayed restricted sequential carrier; standard bridge visible uniformly continuous restriction under three terminal modulus extensions; status ProofTarget. □

Theorem 59.91 (Continuous restriction visible modulus-chain functorial normal form)**.** *Let T, U, V, and W be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, with certified inclusions*

$$k : W \subseteq V, \qquad j : V \subseteq U, \qquad i : U \subseteq T.$$

Let

$$p, q, source_T, source_U, source_V, source_W, map, target, target_d, target_s, oldMod, oldCert, \delta_1, \delta_2, \delta, cert, cert_d, cert_s :$$

be histories. Assume the visible carrier over T

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_T, map, p \cdot target, oldMod \cdot q, (p \cdot oldCert) \cdot q),$$

the source-restriction ledgers

$$\mathsf{Cont}(source_U, \mathsf{e}, source_T), \qquad \mathsf{Cont}(source_V, \mathsf{e}, source_U), \qquad \mathsf{Cont}(source_W, \mathsf{e}, source_V), \qquad \mathsf{Cont}(source_W, \mathsf{e}, s$$

the two-link visible modulus chain

$$\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert),$$

the composite modulus ledger

$$\mathsf{Cont}(\delta_1, \delta_2, \delta),$$

the displayed direct restricted carrier over W

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target_d, \delta \cdot q, (p \cdot cert_d) \cdot q),$$

and the displayed successive restricted carrier over W

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target_s, \delta \cdot q, (p \cdot cert_s) \cdot q).$$

Then the route that restricts directly from T to W and the route that restricts successively

$$T \longrightarrow U \longrightarrow V \longrightarrow W$$

before applying the same visible modulus-chain replacement have the common public normal form

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source_W, map, p \cdot target, \delta \cdot q, (p \cdot cert) \cdot q),$$

and their displayed target and certificate readbacks agree:

$$\mathsf{hsame}(target, target_d) \wedge \mathsf{hsame}(cert, cert_d) \wedge \mathsf{hsame}(target, target_s) \wedge \mathsf{hsame}(cert, cert_s).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible source restriction modulus chain functorial normal form`

Proof. First collapse the nested source restrictions. Apply Theorem 59.83 to $i : U \subseteq T$ and $j : V \subseteq U$ to compare the direct restriction from T to V with the successive route through U.

Apply Theorem 59.83 again to the composite inclusion $V \subseteq T$ and $k : W \subseteq V$. This identifies the direct restriction from T to W with the three-step route through U and V at the $\mathsf{ContinuousMap}^{\uparrow}$ classifier, while preserving the graph ledger map, the visible target ledger $p \cdot target$, and the source ledger $p \cdot source_W$.

Next normalize the visible modulus data on the direct representative. Theorem 59.87, applied to the direct source ledger $\mathsf{Cont}(source_W, \mathsf{e}, source_T)$ and the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$, supplies the canonical visible carrier over W with public histories map, $p{\cdot}target$, $\delta{\cdot}q$, and $(p{\cdot}cert){\cdot}q$.

It remains to read back the two displayed endpoints. The direct displayed carrier has the same restricted source, graph ledger, visible modulus ledger, and visible prefix and suffix as the canonical direct carrier, so Theorem 59.87 gives $\mathsf{hsame}(target, target_d)$ and $\mathsf{hsame}(cert, cert_d)$.

Transport the same canonical carrier across the classifier comparison supplied by the functorial restriction step. Target and certificate determinacy for visible modulus contexts then applies to the successive displayed carrier, giving $\mathsf{hsame}(target, target_s)$ and $\mathsf{hsame}(cert, cert_s)$. Combining the four readbacks yields the stated common normal form for the direct and successive routes.

Ten-field fit: traditional target nested restriction of uniformly continuous maps followed by a visible two-link modulus-chain replacement; BEDC source BHist with visible contexts, certified source-restriction ledgers, Cont, and ContinuousModulusChain; carrier visible ContinuousFunctionCarrier over the final source W; classifier ContinuousMap classifier plus hsame target and certificate readback; stability identity and composite source-admission transport with modulus-chain preservation; ledger common graph ledger, visible target ledger, composite modulus ledger, and visible certificate ledger; core theorem this normal form; exactness direct-versus-successive route agreement after the same chain replacement; standard bridge functorial source restriction with visible modulus-chain normalization for uniformly continuous maps; status ProofTarget. □

59.2 The certificate

Definition 59.92 (ContinuousMap source specification)**.** *The source is a pair of certified metric spaces and the totally bounded subset discipline on the domain.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousSourceSpec`

Definition 59.93 (ContinuousMap pattern specification)**.** *The pattern is graph-with-modulus formation over certified metric spaces and totally bounded source witnesses. A witness supplies the graph relation, the domain and codomain classifier ledgers, and the unary modulus relation that converts a target precision request into the source precision obligation needed for the graph.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousPatternSpec`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousPatternSpec comp carrier`

Definition 59.94 (ContinuousMap classifier specification)**.** *The classifier relates two continuous-map witnesses when their graph outputs agree under the codomain classifier on every pair of source histories related by the source classifier, and when their unary modulus ledgers mutually refine each other for every precision history admitted by the precision classifier.*

Lean4 (definition) `BEDC.Derived.ContinuousUp.ContinuousClassifierSpec`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousClassifierSpec target cert readback`

Definition 59.95 (ContinuousMap stability certificate)**.** *The stability certificate consists of:*

1. *continuous-map classifier reflexivity*
2. *continuous-map classifier symmetry*
3. *continuous-map classifier transitivity*
4. *modulus validity for uniform continuity*

5. *graph and modulus replacement under the continuous-map classifier*
6. *closure under identity*
7. *closure under composition by composing moduli*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier stability certificate fields`

Theorem 59.96 (Continuous modulus chain factorization)**.** *For histories s, δ_1, δ_2, t, assume the two-link chain*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

Then there is a history m such that both one-step links are certified:

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Proof. Unpack the chain into unary data for s, δ_1, δ_2, and t, together with a middle history m and continuation ledgers $s \cdot \delta_1 = m$ and $m \cdot \delta_2 = t$.

The first ledger and the unary source and modulus data assemble

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m).$$

Unary continuation closure then makes the middle history unary. The second ledger and the unary δ_2 and target data assemble

$$\mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Taking this middle history as the existential witness gives the displayed factorization of the chain into its two certified one-step modulus witnesses. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain factorizes`

Theorem 59.97 (Continuous modulus chain middle deterministic factorization)**.** *For histories s, δ_1, δ_2, t, a continuous modulus chain determines a middle history m such that the two one-step modulus witnesses factor the chain, and every other first-leg middle endpoint is the same BEDC history:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \exists m.\ \mathsf{ContinuousModulusWitness}(s, \delta_1, m) \wedge \mathsf{ContinuousModulusWitness}(m, \delta$$

$$\wedge\ \forall m'.\ \mathsf{Cont}(s, \delta_1, m') \Longrightarrow \mathsf{Cont}(m', \delta_2, t) \Longrightarrow \mathsf{hsame}(m, m').$$

Proof. Unpack the chain into unary data and a displayed middle history m with continuation ledgers $s \cdot \delta_1 = m$ and $m \cdot \delta_2 = t$. As in Theorem 59.96, the first ledger gives the first witness and unary continuation closure supplies unary data for m, so the second ledger gives the second witness. If another m' satisfies the same first continuation ledger, Cont determinacy for $s \cdot \delta_1$ yields $\mathsf{hsame}(m, m')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain middle deterministic factorization`

Theorem 59.98 (Continuous modulus chain target determinacy)**.** *For histories $source, first, second, target, targe$ two modulus-chain witnesses with the same source and the same two modulus histories determine the same target up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first, second, target')$$

Proof. Unpack both chains. The first continuation ledgers share *source* and $first$, so continuation determinacy identifies the two middle histories up to hsame. Transport the second continuation comparison along this middle sameness, keeping the common *second* history fixed; continuation respect for hsame then gives $\mathsf{hsame}(target, target')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain target deterministic`

Theorem 59.99 (Continuous modulus chain append-suffix target determinacy)**.** *For histories $q, source, first, second, target, target'$, two modulus chains with the same source and moduli whose targets share the right suffix q determine the unsuffixed targets up to* hsame*.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain append suffix target deterministic`

Proof. Target determinacy identifies $\mathsf{append}(target, q)$ with $\mathsf{append}(target', q)$. Cancel the common right suffix. □

Theorem 59.100 (Continuous modulus chain result determinacy)**.** *For histories $s, \delta_1, \delta_2, t, t'$, two continuous modulus chains with the same source and modulus ledgers determine the same target history:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t') \Longrightarrow \mathsf{hsame}(t, t').$$

Proof. Unpack the two chains into middle histories m and m' with first ledgers $s \cdot \delta_1 = m$ and $s \cdot \delta_1 = m'$, and second ledgers $m \cdot \delta_2 = t$ and $m' \cdot \delta_2 = t'$. Determinacy of the first continuation gives $\mathsf{hsame}(m, m')$. Continuation compatibility with this middle history equality and reflexivity of δ_2 transports the two second ledgers to obtain $\mathsf{hsame}(t, t')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain result deterministic`

Theorem 59.101 (Continuous modulus chain source determinacy)**.** *For histories $source, source', first, second, ta$ two modulus-chain witnesses with the same two modulus histories and the same target determine the same source up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source', first, second, target$$

Proof. Unpack both chains into middle histories and continuation ledgers. The second continuation ledgers have common suffix *second* and common target, so right cancellation identifies the two middle histories. The first continuation ledgers then have common suffix $first$ and endpoints identified by that middle sameness; common-suffix cancellation gives $\mathsf{hsame}(source, source')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain source deterministic`

Theorem 59.102 (Continuous modulus chain append-suffix source determinacy)**.** *For histories $q, source, source', first, second, target$, two modulus chains with the same moduli and target whose sources share the right suffix q determine the unsuffixed sources up to* hsame*.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain append suffix source deterministic`

Proof. Source determinacy identifies $\mathsf{append}(source, q)$ with $\mathsf{append}(source', q)$. Cancel the common right suffix. □

Theorem 59.103 (Continuous modulus chain first determinacy)**.** *For histories source, first, first′, second, target two modulus-chain witnesses with the same source, second modulus, and target determine the same first modulus up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first', second, target)$$

Proof. Unpack both chains. Right cancellation on the two second-leg continuation ledgers identifies the middle histories. Transport the second first-leg ledger along this middle sameness so that both first-leg ledgers have the same source and the same middle endpoint. Left cancellation of continuation then gives $\mathsf{hsame}(first, first')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain first deterministic`

Theorem 59.104 (Continuous modulus chain second determinacy)**.** *For histories source, first, second, second′, ta two modulus-chain witnesses with the same source, first modulus, and target determine the same second modulus up to* hsame*:*

$$\mathsf{ContinuousModulusChain}(source, first, second, target) \Longrightarrow \mathsf{ContinuousModulusChain}(source, first, second', target)$$

Proof. Unpack both chains into their middle histories. Determinacy of $\mathsf{Cont}(source, first, -)$ identifies those middle histories. After transporting across this middle equality, the second-leg ledgers have common left endpoint and common target, so continuation left cancellation gives $\mathsf{hsame}(second, second')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain second deterministic`

Theorem 59.105 (Continuous modulus chain composite closure)**.** *For unary histories s, δ_1, δ_2, t, a two-link modulus chain from s through δ_1 and δ_2 to t, together with a continuation ledger composing δ_1 and δ_2 into δ, determines a single continuous modulus witness from s to t using δ:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \mathsf{ContinuousModulusWitness}(s, \delta, t).$$

Proof. Unpack the chain into unary endpoint and modulus data plus a middle history with ledgers $m = s \cdot \delta_1$ and $t = m \cdot \delta_2$. The composite modulus is unary by closure of unary continuation. The source-to-target continuation is the associativity equality $(s \cdot \delta_1) \cdot \delta_2 = s \cdot (\delta_1 \cdot \delta_2)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain composite closed`

Theorem 59.106 (Continuous modulus chain endpoint cycle empties moduli)**.** *If* $\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2$ *and* $\mathsf{hsame}(s, t)$*, then*

$$\mathsf{hsame}(\delta_1, \mathsf{e}) \wedge \mathsf{hsame}(\delta_2, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain endpoint cycle moduli empty`

Proof. Unpack the chain as $\mathsf{Cont}(s, \delta_1, m)$ and $\mathsf{Cont}(m, \delta_2, t)$. Transport the second ledger across $\mathsf{hsame}(s, t)$ and apply mutual-extension tail cancellation. □

Theorem 59.107 (Continuous modulus witness composite public readback)**.** *For histories source, first, second, m two certified one-step modulus witnesses*

$$\mathsf{ContinuousModulusWitness}(source, first, middle)$$

and

$$\mathsf{ContinuousModulusWitness}(middle, second, target)$$

together with $\mathsf{Cont}(first, second, composite)$ *determine the composite witness*

$$\mathsf{ContinuousModulusWitness}(source, composite, target).$$

Any displayed witness from source through the same composite has target history hsame *to target.*

Proof. The two one-step witnesses supply the source, modulus, middle, and target carriers plus the two continuation ledgers, hence assemble a $\mathsf{ContinuousModulusChain}(source, first, second, target)$. Composite closure turns this chain and the displayed modulus composition into the one-step composite witness. If another target is displayed for the same source and composite modulus, target determinacy for continuous modulus witnesses identifies it with $target$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness composite public readback`

Theorem 59.108 (Continuous function carrier modulus-chain replacement)**.** *For histories* $source, map, target, old$ *assume an existing continuous-function carrier, a two-step modulus chain from the same target history to a certificate history, and a continuation ledger composing the two modulus steps:*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map, target, oldMod, oldCert),\\&\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert),\\&\mathsf{Cont}(\delta_1, \delta_2, \delta).\end{aligned}$$

Then the carrier admits the composite modulus and the chain certificate:

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert).$$

Proof. Unpack the given carrier into unary data for $source$, map, $target$, $oldMod$, and $oldCert$. Keep the graph ledger $\mathsf{Cont}(source, map, target)$; the old modulus and certificate are used only to expose the source, map, and target carrier data.

Apply Theorem 59.105 to the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$ and the composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. This yields the replacement modulus witness

$$\mathsf{ContinuousModulusWitness}(target, \delta, cert).$$

Unpack this witness to obtain unary data for δ and $cert$, together with the modulus-certificate ledger $\mathsf{Cont}(target, \delta, cert)$. Reassemble $\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert)$ from the preserved source, map, and target data, the preserved graph ledger, and this new modulus witness.

The ten-field certificate fit is the following. The traditional target is uniform-continuity data whose modulus certificate may be recomputed through a two-link modulus chain. The BEDC source is $BHist$ equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is carrier preservation under modulus-chain replacement. The ledger consists of source, map, target, old modulus, old certificate, two modulus steps, the composite modulus, and the terminal certificate. The core theorem is this carrier-level replacement, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus chain replacement`

Theorem 59.109 (Prefixed continuous function carrier graph-chain closure)**.** *For histories $p, source, map, target, \delta$ unary source, map, target, and visible-prefix histories, a graph continuation ledger, a two-step modulus chain from the target history to a certificate history, and a continuation ledger composing the two modulus steps determine the prefixed graph carrier with the composite modulus certificate:*

$$\begin{aligned}&\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{UnaryHistory}(source) \Longrightarrow \\ &\mathsf{UnaryHistory}(target) \Longrightarrow \mathsf{UnaryHistory}(map) \Longrightarrow \\ &\mathsf{Cont}(source, map, target) \Longrightarrow \mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert) \Longrightarrow \mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \\ &\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert).\end{aligned}$$

Proof. First apply Theorem 59.113 to the prefix p, the chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2, cert)$, and the composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. This gives the modulus witness

$$\mathsf{ContinuousModulusWitness}(p \cdot target, \delta, p \cdot cert).$$

Unpack this witness into unary data for $p \cdot target$, the composite modulus δ, and $p \cdot cert$, together with the certificate ledger $\mathsf{Cont}(p \cdot target, \delta, p \cdot cert)$.

The graph ledger is prefixed directly from $\mathsf{Cont}(source, map, target)$. Associativity of history append gives

$$(p \cdot source) \cdot map = p \cdot (source \cdot map) = p \cdot target.$$

Unary append closure applied to p and $source$ supplies the unary source history $p \cdot source$. The unary map hypothesis is reused, and the prefixed target, composite modulus, and prefixed certificate data come from the modulus witness. These fields assemble $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$.

The ten-field certificate fit is the following. The traditional target is uniformly continuous map composition under visible finite-prefix evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is carrier preservation from graph data and a prefixed two-link modulus chain. The ledger consists of source, map, target, two modulus steps, the composite modulus, the terminal certificate, and the visible prefix. The core theorem is this carrier-level closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain closed`

Theorem 59.110 (Prefixed continuous function carrier graph-chain public readback)**.** *For histories $p, source, map, target, target', \delta_1, \delta_2, \delta, cert, cert'$, assume unary histories p, source, target, and map, a graph ledger $\mathsf{Cont}(source, map, target)$, a two-link modulus chain $\mathsf{ContinuousModulusChain}(target, \delta_1, \delta_2$ and a composition ledger $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. If a displayed prefixed continuous-function carrier has target $p \cdot target'$ and certificate $p \cdot cert'$,*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', \delta, p \cdot cert'),$$

then the public target and certificate are determined:

$$\mathsf{hsame}(target, target') \quad \textit{and} \quad \mathsf{hsame}(cert, cert').$$

Proof. The graph ledger and the modulus chain first build the canonical prefixed continuous-function carrier with target $p \cdot target$ and certificate $p \cdot cert$. Carrier target-certificate determinacy compares this canonical carrier with the displayed carrier, giving equality of the two prefixed targets and equality of the two prefixed certificates. Left cancellation of the shared prefix p yields the displayed public readback equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain public readback`

Theorem 59.111 (Prefixed continuous function carrier graph-chain second determinacy)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta_2', \delta, \delta', cert$, assume δ_1, δ_2, and δ_2' are unary histories, with continuation ledgers*

$$\mathsf{Cont}(\delta_1, \delta_2, \delta), \qquad \mathsf{Cont}(\delta_1, \delta_2', \delta').$$

If the two composite moduli support the same prefixed continuous function carrier,

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta', p \cdot cert),$$

then the second modulus components are history-same:

$$\mathsf{hsame}(\delta_2, \delta_2').$$

Proof. Unpack the two continuous function carriers and compare their terminal certificate ledgers. Since both ledgers start at $p \cdot target$ and end at $p \cdot cert$, continuation left cancellation identifies the composite moduli δ and δ'. Transport to a common composite modulus, then apply left cancellation to the two ledgers out of δ_1 to obtain $\mathsf{hsame}(\delta_2, \delta_2')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain second deterministic`

Theorem 59.112 (Prefixed continuous function carrier graph-chain first determinacy)**.** *For histories $p, source, map, target, \delta_1, \delta_1', \delta_2, \delta, \delta', cert$, assume δ_1, δ_1', and δ_2 are unary histories, with continuation ledgers*

$$\mathsf{Cont}(\delta_1, \delta_2, \delta), \qquad \mathsf{Cont}(\delta_1', \delta_2, \delta').$$

If the two composite moduli support the same prefixed continuous function carrier,

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta', p \cdot cert),$$

then the first modulus components are history-same:

$$\mathsf{hsame}(\delta_1, \delta_1').$$

Proof. Unpack the two continuous function carriers and compare their terminal certificate ledgers. Since both ledgers start at $p \cdot target$ and end at $p \cdot cert$, continuation left cancellation identifies the composite moduli δ and δ'. Common-suffix cancellation on the two composite modulus ledgers, using the shared suffix δ_2, then yields $\mathsf{hsame}(\delta_1, \delta_1')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain first deterministic`

Theorem 59.113 (Prefixed continuous modulus composite closure)**.** *For unary histories $p, s, \delta_1, \delta_2, t, \delta$, a unary prefix p, a two-link modulus chain from s through δ_1 and δ_2 to t, and a continuation ledger composing δ_1 and δ_2 into δ determine a prefixed continuous modulus witness:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \\ \mathsf{Cont}(\delta_1, \delta_2, \delta) \Longrightarrow \mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t).$$

Proof. Unpack the chain into unary endpoint and modulus histories, together with the continuation ledgers from s to the middle history and from the middle history to t.

The prefixed source and target histories are unary by unary append closure applied to p with s and t. The composite modulus history is unary by unary continuation closure applied to the two modulus histories and their composition ledger.

The remaining continuation ledger is obtained by associating the concatenation spine:

$$(p \cdot s) \cdot \delta = p \cdot (s \cdot \delta) = p \cdot t.$$

Thus the prefixed endpoints and the same composite modulus satisfy the public continuous modulus witness predicate. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite closed`

Theorem 59.114 (Continuous modulus witness prefixed composite factorization)**.** *For histories $p, s, \delta_1, \delta_2, \delta, t$, if δ_1 and δ_2 are unary histories, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t)$$

holds, then there is a middle history m such that

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \wedge \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t).$$

Proof. First remove the common prefix from the displayed continuous modulus witness using Theorem 59.21. This leaves a witness from s through the composite modulus δ to t.

Choose $m = s \cdot \delta_1$. The first witness is immediate from $\mathsf{Cont}(s, \delta_1, s \cdot \delta_1)$ and unary append closure. For the second witness, case-analyze the two continuation ledgers $\mathsf{Cont}(s, \delta, t)$ and $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. Associativity identifies $(s \cdot \delta_1) \cdot \delta_2$ with $s \cdot (\delta_1 \cdot \delta_2)$, hence with t. This supplies the required continuation ledger from m through δ_2 to t. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite factorizes`

Theorem 59.115 (Continuous prefixed composite middle deterministic factorization)**.** *For histories $p, s, \delta_1, \delta_2, \delta, t$, if δ_1 and δ_2 are unary histories, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t)$$

holds, then there is a middle history m such that

$$\mathsf{ContinuousModulusWitness}(s, \delta_1, m) \quad \wedge \quad \mathsf{ContinuousModulusWitness}(m, \delta_2, t),$$

and every other middle history carrying the same two witness legs is hsame *to m.*

Proof. First apply the prefixed composite factorization theorem to expose a middle history and the two one-step continuous modulus witnesses. If another middle history carries a witness from s through δ_1, both first legs have the same source and modulus; one-step target determinacy therefore identifies the two middle histories by hsame. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite middle deterministic factorization`

Theorem 59.116 (Continuous prefixed composite first deterministic)**.** *For histories $p, s, \delta_1, \delta_1', \delta_2, \delta, \delta', t$, assume δ_1, δ_1', and δ_2 are unary,* $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ *and* $\mathsf{Cont}(\delta_1', \delta_2, \delta')$ *hold, and both*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s, \delta', p \cdot t)$$

hold. Then $\mathsf{hsame}(\delta_1, \delta_1')$*.*

Proof. Factor both prefixed composite witnesses. The second legs share the same target t and the same second modulus δ_2, so one-step source determinacy identifies their middle histories. The first legs then share the same source s and hsame target histories, so one-step modulus determinacy identifies δ_1 and δ_1'. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite first deterministic`

Theorem 59.117 (Continuous modulus witness prefixed composite source determinacy)**.** *For histories $p, s, s', \delta_1, \delta_2, \delta, t$, assume δ_1 and δ_2 are unary histories and* $\mathsf{Cont}(\delta_1, \delta_2, \delta)$*. If two prefixed composite modulus witnesses share the same composite modulus and the same prefixed target,*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s', \delta, p \cdot t),$$

then their unprefixed sources are history-same:

$$\mathsf{hsame}(s, s').$$

Proof. Factor both prefixed composite witnesses into two one-step modulus witnesses. The second legs share the modulus δ_2 and target t, so one-step source determinacy identifies the two recovered middle histories. The first legs then share the modulus δ_1 and have history-same middle targets, so one-step source determinacy gives $\mathsf{hsame}(s, s')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite source deterministic`

Theorem 59.118 (Continuous prefixed composite second deterministic)**.** *For histories $p, s, \delta_1, \delta_2, \delta_2', \delta, \delta', t$, assume δ_1, δ_2, and δ_2' are unary,* $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ *and* $\mathsf{Cont}(\delta_1, \delta_2', \delta')$ *hold, and both*

$$\mathsf{ContinuousModulusWitness}(p \cdot s, \delta, p \cdot t) \quad \textit{and} \quad \mathsf{ContinuousModulusWitness}(p \cdot s, \delta', p \cdot t)$$

hold. Then $\mathsf{hsame}(\delta_2, \delta_2')$*.*

Proof. Factor both prefixed composite witnesses. The first legs share the same source s and first modulus δ_1, so one-step target determinacy identifies the two middle histories. The second legs then have hsame source histories and the same target t, so one-step modulus determinacy identifies δ_2 and δ_2'. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness prefixed composite second deterministic`

Theorem 59.119 (Continuous modulus chain prefix closure)**.** *For histories $p, s, \delta_1, \delta_2, t$, a unary prefix on the source and target preserves a two-link continuous modulus chain:*

$$\mathsf{UnaryHistory}(p) \Longrightarrow \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t) \Longrightarrow \mathsf{ContinuousModulusChain}(p{\cdot}s, \delta_1, \delta_2, p{\cdot}t).$$

Proof. Unpack the chain into unary endpoint data and a middle history m with continuations $m = s \cdot \delta_1$ and $t = m \cdot \delta_2$. Use $p \cdot m$ as the middle history for the prefixed chain. Unary append closure supplies the prefixed endpoints, and continuation associativity prefixes both ledger equalities. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix closed`

Theorem 59.120 (Continuous modulus chain prefix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, prefixing both endpoints of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(p \cdot s, \delta_1, \delta_2, p \cdot t) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

The traditional target is prefix-stable uniform-modulus data. The BEDC source is BHist with Cont*, the carrier is* ContinuousModulusChain*, and the classifier is* hsame*. The stability obligation is prefix exactness. The ledger records source, two precision histories, middle, and target continuations. The core theorem is this exactness statement, the exactness predicate is the displayed iff, the standard bridge is deferred, and the status is* ProofTarget*.*

Proof. For the forward direction, unpack the prefixed chain. Unary append factorization gives unary p and unary s. Let the unprefixed middle be $s \cdot \delta_1$.

The first continuation ledger, together with associativity, identifies the prefixed middle as $p \cdot (s \cdot \delta_1)$. The second continuation ledger then has a common visible prefix p on both sides. Associativity and left cancellation recover the unprefixed second continuation from $s \cdot \delta_1$ through δ_2 to t. These data assemble $\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t)$, paired with unary p.

For the reverse direction, take unary p together with the unprefixed chain and apply Theorem 59.119. This gives the prefixed chain from $p \cdot s$ through the same two precision histories to $p \cdot t$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain prefix iff`

Theorem 59.121 (Continuous modulus chain suffix exactness)**.** *For histories $p, s, \delta_1, \delta_2, t$, suffixing the second modulus and the target of a two-link continuous modulus chain is exact:*

$$\mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2 \cdot p, t \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousModulusChain}(s, \delta_1, \delta_2, t).$$

Proof. For the forward direction, unpack the suffixed chain and derive the unary middle history from the first continuation. The second continuation is a continuous modulus witness from the middle history through $\delta_2 \cdot p$ to $t \cdot p$. Apply Theorem 59.22 to remove the common right suffix from that second leg, then rebuild the original two-link chain.

For the reverse direction, apply Theorem 59.22 to the second leg of the chain and reuse the same middle history and first continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain suffix iff`

Definition 59.122 (ContinuousMap ledger policy)**.** *Record the graph relation, source and target metric ledgers, the totally bounded subset witness, the unary precision ledger, and the modulus ledger.*

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier ledger policy fields`

Definition 59.123 ($\mathsf{ContinuousMap}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{ContinuousMap}^{\uparrow}$*.*

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier certificate semanticNameCert`

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousUp StdBridge`

Primitive boundary. $\mathsf{ContinuousMap}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 59.124 (Zero-axiom encoding note)**.** *The encoding avoids extensional equality of maps and host* `Nat` *by using graph classifier sameness and unary precision histories. The modulus object is a certified relation between precision histories, so no host-level numeric modulus is exposed at the certificate boundary.*

Proposition 59.125 (Uniform modulus restriction along subset containment)**.** *Let Y and Z be certified* $\mathsf{Met}^{\uparrow}$ *objects with the metric data of Definition 57.1. Let T and U be certified source-subset witnesses in the totally bounded domain discipline of Definition 59.92, and let $i : U \subseteq T$ be a certified inclusion. For a graph witness $f : T \to Z$, let $f\restriction_U$ be the same graph relation read with the source classifier restricted along i. If a unary precision ledger δ satisfies the modulus-validity field of Definition 59.161 for f over T, then the same δ satisfies that modulus-validity field for $f\restriction_U$ over U.*

Proof. Fix a target precision request and two source histories admitted by U. The certified inclusion sends their source-membership ledgers to membership ledgers for T:

$$u_j \in U \Longrightarrow i(\nu_j) \in T \qquad (j = 0, 1).$$

The graph relation, codomain classifier ledger, unary precision ledger, and modulus ledger are the same ledgers as in the witness over T, as specified by Definition 59.93 and Definition 59.122. Apply the modulus-validity field over T to the two transported source memberships. It returns the codomain metric conclusion at the requested target precision for the two graph outputs. The metric judgments are well formed and stable under classifier replacement by Definition 57.4 and Definition 57.5.

The restricted witness over U asks for exactly the same codomain conclusion, with only the source-admission hypotheses narrowed from T to U. The totality and single-valuedness clauses from Definition 59.1 license the same graph outputs, while no precision or modulus ledger is changed. □

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source restriction public readback`

Closure status for $\mathsf{NameCert}_{\mathsf{Continuous}^{\uparrow}}$**.**

Constructive story: A continuous map is carried by a graph-and-modulus history packet: source, target, map code, modulus, and certificate histories are unary, the graph ledger continues the source through the map to the target, and the modulus ledger continues the target through the modulus to the certificate. Two such witnesses are classified by the graph outputs agreeing under the target classifier for classified source inputs and by their unary modulus ledgers mutually refining each requested precision; on the checked carrier rows this is enforced through hsame determinacy of the displayed continuation fields. The source consists of certified metric spaces and the totally bounded source discipline, the pattern records graph-with-modulus formation, identity, composition, prefix and restriction examples, terminal-modulus extension, and modulus-chain exactness, and the ledger keeps the graph relation, metric ledgers, precision history, modulus history, and certificate history visible. The five name-certificate obligations close because the identity map uses the empty map code with a one-step modulus witness, hsame supplies reflexive, symmetric, and transitive comparison of the displayed histories, and carrier transport follows by moving each continuation ledger across history-sameness. A carrier row lies in the signature gap of a $\mathsf{Continuous}^{\uparrow}$ token p when the source-domain accepts the graph-and-modulus packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, the graph sends each accepted source history to the encoded target history and the modulus ledger supplies the precision conversion recorded by p. The checked standard bridge reads the same relation-plus-ledger data as a continuous-map boundary and reads it back without host-level function extensionality or a numeric modulus object. The chapter registers this graph-modulus construction under the name $\mathsf{Continuous}^{\uparrow}$; continuity is an explicit BEDC certificate surface, not a host function space, completeness theorem, or external map equality principle.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Continuous}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope binds the ContinuousFunctionCarrier semantic certificate, modulus-witness determinacy and exactness rows, identity, composition, prefix, and restriction examples, and the ContinuousMapUp consumer surface cited by the chapter.
Formal verification: FormalStatus(NameCert$_{\text{Continuous}\uparrow}$, bridgeChecked)
Lean target: `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier\_certificate\_semanticNameCert`
Bridge status: bridgeChecked
Not claimed: The chapter does not close host functions, host function spaces, or standard-model completeness.
Upgrade path: no further upgrade – chapter at top of formal axis

Theorem 59.126 (Continuous modulus witness modulus determinacy)**.** *If two one-step modulus witnesses have* hsame *source histories and* hsame *target histories, then their modulus histories are* hsame*:*

$$\mathsf{hsame}(modulus, modulus').$$

Proof. Unpack both witnesses, identify the source and target endpoints by hsame, and apply continuation left cancellation to the two modulus ledgers. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness modulus hsame deterministic`

Theorem 59.127 (Continuous modulus witness target determinacy)**.** *If two one-step modulus witnesses have* hsame *source histories and the same modulus history, then their target histories are* hsame*:*

$$\mathsf{hsame}(target, target').$$

Proof. Unpack both witnesses, identify the source endpoint by hsame, and apply continuation result determinacy to the two continuation ledgers with the shared source and modulus. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness target hsame deterministic`

Theorem 59.128 (Continuous modulus witness source determinacy)**.** *If two one-step modulus witnesses have* hsame *target histories and the same modulus history, then their source histories are* hsame*:*

$$\mathsf{hsame}(source, source').$$

Proof. Unpack both witnesses, identify the target endpoint by hsame, and apply continuation right cancellation to the two ledgers with the shared modulus and target. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness source hsame deterministic`

Theorem 59.129 (Continuous modulus witness empty target exactness)**.** *For histories source and modulus, a one-step continuous modulus witness with empty target is exactly the conjunction that both source and modulus are empty histories:*

$$\mathsf{ContinuousModulusWitness}(source, modulus, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e}).$$

Proof. Unpack the witness into unary carriers and the continuation ledger $\mathsf{Cont}(source, modulus, \mathsf{e})$. Empty-result inversion for continuation gives the two endpoint equalities. Conversely, if both endpoints are empty, the empty unary carrier and the canonical empty continuation rebuild the witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty target iff`

Theorem 59.130 (Continuous empty-source nonempty modulus shape)**.** *For histories target, modulus, an empty-source continuous modulus witness with nonempty modulus has exactly one visible unary payload:*

$$\mathsf{ContinuousModulusWitness}(\mathsf{e}, modulus, target) \wedge \neg \mathsf{hsame}(modulus, \mathsf{e}) \iff \exists r : \mathsf{BHist}.\ target = \mathsf{E1}(r) \wedge modulus = \mathsf{E}$$

Proof. Empty-source modulus exactness identifies *modulus* with *target* and gives unary membership of *target*. Nonemptiness excludes the empty target, unary membership excludes zero-headed targets, and the visible case strips one E1 constructor to expose the unary payload. The reverse direction rebuilds the witness from the visible unary target and reflexive sameness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness empty source nonempty modulus shape iff`

Theorem 59.131 (Continuous modulus chain empty target exactness)**.** *For histories source, first, second, a two-link continuous modulus chain ending at* e *is exactly the conjunction that all three displayed histories are empty:*

$$\mathsf{ContinuousModulusChain}(source, first, second, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(first, \mathsf{e}) \wedge \mathsf{hsame}(second, \mathsf{e}).$$

Proof. Unpack the middle history and the two continuation ledgers. Empty-result inversion on the second leg identifies the middle history and *second* with e; after replacing the middle endpoint, empty-result inversion on the first leg identifies *source* and *first* with e. The reverse direction uses the empty middle history and the two right-unit continuations at e. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty target iff`

Theorem 59.132 (Continuous modulus chain empty moduli exactness)**.** *For histories source, target, a two-link continuous modulus chain with both moduli empty is exactly the conjunction that the endpoints are unary and the target is history-same to the source:*

$$\mathsf{ContinuousModulusChain}(source, \mathsf{e}, \mathsf{e}, target) \iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(target) \wedge \mathsf{hsame}(target, sour$$

Proof. Unpack the middle history and the two continuation ledgers. Each empty-modulus leg is a right-unit continuation, so the middle endpoint is history-same to the source and the target endpoint is history-same to the middle endpoint. Transitivity gives $\mathsf{hsame}(target, source)$. Conversely, choose *source* as the middle history, use the canonical right-unit continuation for the first leg, and use the displayed endpoint sameness to build the second right-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty moduli iff`

Theorem 59.133 (Continuous modulus chain empty second witness)**.** *For histories source, first, second, target, if a two-link continuous modulus chain has second*hsamee*, then the first link directly witnesses* $\mathsf{ContinuousModulusWitness}(source, first, target)$*.*

Proof. Unpack the middle history and the two continuation ledgers. The second ledger, transported across *second*hsamee, is compared with the right-unit continuation at the middle endpoint, so the target is history-same to the middle endpoint. Transporting the first continuation result along this endpoint sameness gives the displayed one-step witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty second witness`

Theorem 59.134 (Continuous modulus chain empty first witness)**.** *For histories source*, *first*, *second*, *target, if a two-link continuous modulus chain has first*hsamee*, then the second link directly witnesses* ContinuousModulusWitness(*source*, *second*, *target*)*.*

Proof. Unpack the middle history and the two continuation ledgers. The first ledger, transported across *first*hsamee, is compared with the right-unit continuation at the source endpoint, so the middle endpoint is history-same to the source. Transporting the second continuation source along this endpoint sameness gives the displayed one-step witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain empty first witness`

Theorem 59.135 (Continuous function carrier empty certificate exactness)**.** *For histories source*, *map*, *target*, *mod a continuous-function carrier with empty certificate is exactly the conjunction that its graph and modulus histories are all empty:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(target, \mathsf{e})$$

Proof. Unpack the graph continuation and the target-to-certificate continuation. Empty-result inversion on the certificate leg identifies *target* and *modulus* with e; after replacing the target endpoint, empty-result inversion on the graph leg identifies *source* and *map* with e. Conversely, when all four histories are empty, the empty unary carrier and the two right-unit continuations rebuild the carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty cert iff`

Theorem 59.136 (Continuous function carrier empty-modulus exactness)**.** *For histories source*, *map*, *target*, *cert, a continuous-function carrier with empty modulus is exactly the graph carrier data together with the statement that the certificate is history-same to the target:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \mathsf{e}, cert) \iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(target) \wedge \mathsf{UnaryHist}$$

Proof. Unpack the continuous-function carrier. The graph fields already supply the three displayed unary histories and the graph continuation ledger. The target-to-certificate leg has empty modulus, so the right-unit continuation law identifies *cert* with *target*. Conversely, the displayed graph data, the empty unary carrier, and the right-unit continuation rebuilt from hsame(*cert*, *target*) reassemble the continuous-function carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty modulus iff`

Theorem 59.137 (Continuous function carrier empty-target exactness)**.** *For histories source*, *map*, *modulus*, *cert, the empty target case of* ContinuousFunctionCarrier *is exact:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, \mathsf{e}, modulus, cert) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{UnaryHistory}(mod$$

Proof. In the forward direction, empty-result inversion on the graph ledger identifies the source and map histories with e. The certificate ledger is then a left-unit continuation from e through *modulus*, so the certificate is history-same to the modulus; unary continuation closure supplies the certificate carrier.

Conversely, the displayed empty source and map rebuild the graph ledger by the right-unit continuation at e. The displayed certificate-modulus sameness rebuilds the certificate ledger by the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty target iff`

Theorem 59.138 (Continuous function carrier empty-source exactness)**.** *For histories map, target, modulus, and cert, the empty source case of* ContinuousFunctionCarrier *is exact:*

$$\mathsf{ContinuousFunctionCarrier}(\mathsf{e}, map, target, modulus, cert) \iff \mathsf{UnaryHistory}(map) \wedge \mathsf{ContinuousModulusWitness}(map$$

Proof. In the forward direction, unpack the carrier. The graph ledger is $\mathsf{Cont}(\mathsf{e}, map, target)$, so the continuation left-unit theorem identifies *target* with *map*. The target-modulus continuation then becomes a modulus witness over *map*, and unary continuation closure supplies the certificate carrier.

Conversely, the displayed modulus witness supplies the map, modulus, certificate, and certificate-continuation fields. The empty source is unary, the target is identified with the map by the displayed hsame, and the graph ledger is the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier empty source iff`

Theorem 59.139 (Continuous function carrier graph-modulus readback)**.** *For histories source, map, target, modu every continuous-function carrier directly exposes its graph continuation ledger, its modulus-certificate continuation ledger, and the two corresponding one-step modulus witnesses:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \implies \mathsf{Cont}(source, map, target) \wedge \mathsf{Cont}(target, modu$$
$$\wedge\ \mathsf{ContinuousModulusWitness}(source, map, target) \wedge \mathsf{ContinuousModulusWitness}(target, modulus, cert).$$

Proof. Unpack the continuous-function carrier into unary data for *source*, *target*, *map*, and *modulus*, together with the two continuation ledgers. The graph ledger and the first three unary fields form $\mathsf{ContinuousModulusWitness}(source, map, target)$. Unary closure along the target-modulus ledger supplies the carrier for *cert*, and this packages the second witness $\mathsf{ContinuousModulusWitness}(target,$ □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier graph modulus cont readback`

Theorem 59.140 (Continuous function carrier endpoint-certificate cycle tails)**.** *For histories source, map, target, modulus, cert, a continuous-function carrier whose source is history-same to its certificate has empty graph tail and empty modulus tail:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \implies \mathsf{hsame}(source, cert) \implies \mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}($$

Proof. Unpack the carrier into its graph ledger $\mathsf{Cont}(source, map, target)$ and certificate ledger $\mathsf{Cont}(target, modulus, cert)$. Transport the certificate ledger across $\mathsf{hsame}(source, cert)$, obtaining $\mathsf{Cont}(target, modulus, source)$. The two ledgers form a continuation cycle, so mutual-extension cancellation forces both tails, *map* and *modulus*, to be history-same to e. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier endpoint cert cycle tails empty`

Theorem 59.141 (Continuous function carrier endpoint-certificate cycle exactness)**.** *For histories source, map, target, modulus, cert, if a continuous-function carrier closes its public source against its endpoint certificate, then the carrier is an identity presentation at that endpoint:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \implies \mathsf{hsame}(source, cert) \implies$$
$$\mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e}) \wedge \mathsf{hsame}(target, source) \wedge \mathsf{hsame}(cert, source).$$

Proof. Apply Definition 59.140 to the carrier and the displayed endpoint certificate equality. This gives $\mathsf{hsame}(map, \mathsf{e})$ and $\mathsf{hsame}(modulus, \mathsf{e})$.

Unpack the graph ledger $\mathsf{Cont}(source, map, target)$ from the carrier. Transport that ledger across $\mathsf{hsame}(map, \mathsf{e})$ to obtain $\mathsf{Cont}(source, \mathsf{e}, target)$. The right-unit clause of Definition 21.12 supplies $\mathsf{Cont}(source, \mathsf{e}, source)$, and continuation determinacy from Definition 21.7 identifies $target$ with $source$.

Finally, symmetry in Definition 3.8 turns the displayed $\mathsf{hsame}(source, cert)$ into $\mathsf{hsame}(cert, source)$, and the four pieces assemble the stated conjunction. □

The ten-field fit is: traditional target, identity-cycle exactness for continuous-function carriers; BEDC kernel, $BHist$; ledger relation, Cont; carrier, $\mathsf{ContinuousFunctionCarrier}$; classifier, hsame; ledger fields, $source, map, target, modulus, cert$; core theorem, Definition 59.141; exactness, the four displayed hsame rows; standard bridge, deferred; downstream consumer, $\mathsf{ContinuousMapUp}$ identity and category normalization.

Theorem 59.142 (Continuous function carrier composite endpoint-certificate cycle exactness)**.** *For histories $source, middle, target, f, g, fg, modF, modG, modFG, certF, certG, cert$, closed composition of continuous-function carriers inherits endpoint-certificate cycle collapse:*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF) \wedge \mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, c\\ &\mathsf{Cont}(f, g, fg) \wedge \mathsf{Cont}(modF, modG, modFG) \wedge \mathsf{Cont}(target, modFG, cert) \wedge \mathsf{hsame}(source, cert)\\ &\Longrightarrow \mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG, cert) \wedge \mathsf{hsame}(fg, \mathsf{e}) \wedge \mathsf{hsame}(modFG, \mathsf{e}) \wedge \mathsf{hsame}(targ\end{aligned}$$

Proof. First apply Definition 59.7 to the two given carriers, the displayed composite map and modulus continuations, and the target-to-certificate continuation. This constructs $\mathsf{ContinuousFunctionCarrier}(source, fg,$

Next apply Definition 59.141 to this composite carrier and the displayed endpoint-certificate equality $\mathsf{hsame}(source, cert)$. The resulting four history-sameness conclusions, together with the composite carrier just constructed, assemble the stated conjunction. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composite endpoint cert cycle exactness`

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composite endpoint cycle exactness`

Theorem 59.143 (Continuous function carrier prefix exactness)**.** *For histories $p, source, map, target, modulus, cer$ prefixing the visible source, target, and certificate ledgers is conservative for the continuous-function carrier:*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus, p \cdot cert) \Longleftrightarrow\\ &\mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).\end{aligned}$$

Proof. For the forward direction, unpack the prefixed carrier. It supplies unary histories for $p \cdot source$, $p \cdot target$, map, and $modulus$, together with the two continuation ledgers $\mathsf{Cont}(p \cdot source, map, p \cdot target)$ and $\mathsf{Cont}(p \cdot target, modulus, p \cdot cert)$.

Unary append factorization applied to the source ledger recovers $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(source)$. Applied to the target ledger, it recovers $\mathsf{UnaryHistory}(target)$. Prefix cancellation on the graph continuation ledger gives $\mathsf{Cont}(source, map, target)$, and prefix cancellation on the modulus-certificate continuation ledger gives $\mathsf{Cont}(target, modulus, cert)$. Repacking these witnesses gives $\mathsf{ContinuousFunctionCarrier}(s$

For the reverse direction, project $\mathsf{UnaryHistory}(p)$ and the unprefixed carrier from the conjunction. Apply Theorem 59.20 to obtain the displayed prefixed carrier.

The ten-field certificate fit is the following. The traditional target is prefix-invariant continuous graph evidence. The BEDC source is $BHist$ equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is exact prefix cancellation for graph and modulus-certificate ledgers. The ledger consists of source, map, target, modulus, and certificate histories. The core theorem is prefix exactness, the exactness predicate is the displayed iff, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefix iff`

Theorem 59.144 (Continuous function carrier modulus suffix exactness)**.** *For histories $p, source, map, target, mod$ suffixing the modulus and certificate ledgers by the same unary history is conservative for the continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus \cdot p, cert \cdot p) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert).$$

Proof. For the forward direction, unpack the suffixed carrier. The target-modulus certificate leg is a continuous modulus witness from $target$ through $modulus \cdot p$ to $cert \cdot p$, with the certificate unary witness obtained from unary continuation closure. Apply the continuous modulus witness suffix exactness theorem to remove the common right suffix, then repack the graph carrier fields.

For the reverse direction, turn the unsuffixed target-modulus certificate leg into a continuous modulus witness, apply suffix exactness to append p to the modulus and certificate histories, and repack the original graph fields with the suffixed certificate leg. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus suffix iff`

Theorem 59.145 (Continuous function carrier modulus-suffix target-certificate determinacy)**.** *For histories $q, source, map, target, target', modulus, cert, cert'$, if two continuous-function carrier witnesses share the same source, map, and visible modulus suffix,*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus{\cdot}q, cert{\cdot}q) \quad \textit{and} \quad \mathsf{ContinuousFunctionCarrier}(source, map, t$$

then their central target and certificate ledgers are determined:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply modulus suffix exactness to each carrier witness. This removes the common suffix from the modulus and certificate ledgers and exposes two central continuous-function carriers over the same source, map, and modulus. The target-certificate determinacy theorem for continuous-function carriers then identifies the central targets and the central certificates. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier modulus suffix target cert deterministic`

Theorem 59.146 (Continuous function carrier prefixed graph-chain factorization)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta, cert$, if δ_1 and δ_2 are unary,* $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ *holds, and*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

holds, then there is a middle history middle such that

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta, cert) \wedge \mathsf{ContinuousModulusWitness}(target, \delta_1, middle) \wedge \mathsf{Continuous}$$

Proof. Apply prefix exactness for continuous-function carriers to remove the common visible prefix from the source, target, and certificate ledgers. This recovers the unprefixed carrier and in particular the target-to-certificate continuation $\mathsf{Cont}(target, \delta, cert)$.

Choose $middle = target \cdot \delta_1$. The first modulus witness is the canonical continuation from $target$ through δ_1 to this middle history. For the second witness, combine $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ with the recovered target-to-certificate continuation and reassociate the BEDC append spine, obtaining $\mathsf{Cont}(middle, \delta_2, cert)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain factorizes`

Theorem 59.147 (Continuous function carrier prefixed graph-chain first factorization)**.** *For histories $p, source, map, target, \delta_1, \delta_2, \delta, cert$, if δ_1 and δ_2 are unary, $\mathsf{Cont}(\delta_1, \delta_2, \delta)$ holds, and*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, \delta, p \cdot cert)$$

holds, then there is a middle history middle such that

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, \delta_1, middle) \wedge \mathsf{ContinuousModulusWitness}(middle, \delta_2, cert).$$

Proof. Apply prefix exactness for continuous-function carriers to remove the common visible prefix. This exposes the unprefixed graph ledger $\mathsf{Cont}(source, map, target)$ and the composite target-certificate ledger $\mathsf{Cont}(target, \delta, cert)$.

Choose $middle = target \cdot \delta_1$. The first-stage carrier uses the original source, map, and target carrier data, the graph ledger, the unary witness for δ_1, and the canonical continuation $\mathsf{Cont}(target, \delta_1, middle)$. The second-stage modulus witness uses the unary closure of $middle$, the unary witness for δ_2, unary closure of $cert$, and associativity of BEDC append to factor the composite ledger through $\mathsf{Cont}(\delta_1, \delta_2, \delta)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier prefixed graph chain factorization`

Theorem 59.148 (Continuous function carrier triple composition certificate synchronization)**.** *For BHist histories $source, mid_1, mid_2, target, f, g, h, fg, gh, map_L, map_R, mod_F, mod_G, mod_H, mod_{FG}, mod_{GH}, mod_L$, suppose the three one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, target, mod_H, cert_H)
\end{aligned}$$

hold. Suppose also that the left-bracketed ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, map_L), & \mathsf{Cont}(mod_{FG}, mod_H, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-bracketed ledgers

$$\begin{array}{lll}
\mathsf{Cont}(g, h, gh), & \mathsf{Cont}(mod_G, mod_H, mod_{GH}), & \mathsf{Cont}(target, mod_{GH}, cert_{GH}),\\
\mathsf{Cont}(f, gh, map_R), & \mathsf{Cont}(mod_F, mod_{GH}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the two bracketings both carry the original source to the final target and their public composite data agree up to history-sameness:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left bracketing. Theorem 59.7 composes the $source$-f-mid_1 carrier with the mid_1-g-mid_2 carrier using the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Applying the same closure theorem to that composite and the mid_2-h-$target$ carrier gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod$

The right bracketing is built in the symmetric order. Compose the g and h carriers through gh, mod_{GH}, and $cert_{GH}$, then compose the resulting mid_1-gh-$target$ carrier after the original f carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For map synchronization, apply continuation associativity up to history-sameness to the four graph ledgers $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, map_L)$, $\mathsf{Cont}(g, h, gh)$, and $\mathsf{Cont}(f, gh, map_R)$. The result is $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same continuation-associativity principle to the modulus ledgers $\mathsf{Cont}(mod_F, mod_G, mod_{FG})$, $\mathsf{Cont}(mod_{FG}, mod_H, mod_L)$, $\mathsf{Cont}(mod_G, mod_H, mod_{GH})$, and $\mathsf{Cont}(mod_F, mod_{GH}, mod_R)$. This gives $\mathsf{hsame}(mod_L, mod_R)$.

Finally compare the two target-certificate ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$. Continuation respect for hsame, with reflexivity on $target$ and the recovered modulus sameness, yields $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is threefold composition of continuous function carriers. The BEDC source is $BHist$ equipped with Cont. The carrier is $\mathsf{ContinuousFunctionCarrier}$ and the classifier is hsame. The stability obligation is bracketing synchronization for graph, modulus, and certificate ledgers. The ledger consists of source, intermediate, target, map, modulus, and certificate histories. The core theorem is this triple composition synchronization result. The exactness predicate is the displayed pair of carriers together with the three hsame conclusions. The standard bridge is deferred, and the status is $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier composition associative certificate synchronization`

Theorem 59.149 (Continuous function carrier fourfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, target, f, g, h, k, fg, fgh, hk, ghk, map_L, map_R, mod_F, mod_G, mod_H, mod_K, mod_{FG}$ *suppose the four one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, target, mod_K, cert_K)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, map_L), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(h, k, hk), & \mathsf{Cont}(mod_H, mod_K, mod_{HK}), & \mathsf{Cont}(target, mod_{HK}, cert_{HK}),\\
\mathsf{Cont}(g, hk, ghk), & \mathsf{Cont}(mod_G, mod_{HK}, mod_{GHK}), & \mathsf{Cont}(target, mod_{GHK}, cert_{GHK}),\\
\mathsf{Cont}(f, ghk, map_R), & \mathsf{Cont}(mod_F, mod_{GHK}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)\wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)\wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then again with the k carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, c$

The right-deep carrier is obtained by the symmetric closure chain. Compose the h and k carriers through hk, mod_{HK}, and $cert_{HK}$; compose the g carrier with that composite through ghk, mod_{GHK}, and $cert_{GHK}$; then compose the f carrier with the resulting carrier to obtain $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply the BEDC fourfold continuation associativity theorem cont_assoc_four to the common graph spine f, g, h, k. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, and $\mathsf{Cont}(fgh, k, map_L)$, while the right spine is presented by $\mathsf{Cont}(h, k, hk)$, $\mathsf{Cont}(g, hk, ghk)$, and $\mathsf{Cont}(f, ghk, map_R)$. The underlying append-associativity spine append_assoc yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fourfold continuation associativity theorem to $mod_F, mod_G, mod_H, mod_K$. The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common append spine yields $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the certificate ledger to give $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is fourfold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the graph, modulus, and certificate histories displayed above. The stability field is fourfold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp associativity coherence and the later category bridge. □

Theorem 59.150 (Continuous function carrier fivefold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, target, f, g, h, k, l, fg, fgh, fghk, kl, hkl, ghkl, map_L, map_R, mod_F, mod_G, mo$ *suppose the five one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\
&\mathsf{ContinuousFunctionCarrier}(mid_4, l, target, mod_l, cert_l)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}),\\
\mathsf{Cont}(fghk, l, map_L), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(k, l, kl), & \mathsf{Cont}(mod_K, mod_l, mod_{KL}), & \mathsf{Cont}(target, mod_{KL}, cert_{KL}),\\
\mathsf{Cont}(h, kl, hkl), & \mathsf{Cont}(mod_H, mod_{KL}, mod_{HKL}), & \mathsf{Cont}(target, mod_{HKL}, cert_{HKL}),\\
\mathsf{Cont}(g, hkl, ghkl), & \mathsf{Cont}(mod_G, mod_{HKL}, mod_{GHKL}), & \mathsf{Cont}(target, mod_{GHKL}, cert_{GHKL}),\\
\mathsf{Cont}(f, ghkl, map_R), & \mathsf{Cont}(mod_F, mod_{GHKL}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)\wedge\\&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)\wedge\\&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then with the k carrier, and finally with the l carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is obtained by the matching closure chain. Compose the k and l carriers through kl, mod_{KL}, and $cert_{KL}$; compose the h carrier with that composite through hkl, mod_{HKL}, and $cert_{HKL}$; compose the g carrier with the resulting carrier through $ghkl$, mod_{GHKL}, and $cert_{GHKL}$; then compose the f carrier with that final right-deep carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_five to the common graph spine f, g, h, k, l. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, $\mathsf{Cont}(fgh, k, fghk)$, and $\mathsf{Cont}(fghk, l, map_L)$, while the right spine is presented by $\mathsf{Cont}(k, l, kl)$, $\mathsf{Cont}(h, kl, hkl)$, $\mathsf{Cont}(g, hkl, ghkl)$, and $\mathsf{Cont}(f, ghkl, map_R)$. The fivefold continuation reassociation gives $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fivefold reassociation theorem to $mod_F, mod_G, mod_H, mod_K, mod_l$ The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common continuation spine gives $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the terminal certificate ledger and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is pentagon-style fivefold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the five one-step graph histories, their modulus histories, their target-certificate histories, and the displayed left-deep and right-deep composite ledgers. The stability field is fivefold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp pentagon coherence and the category bridge. □

Theorem 59.151 (Continuous function carrier sixfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, mid_5, target, f, g, h, k, l, m, fg, fgh, fghk, fghkl, lm, klm, hklm, ghklm, map_L,$ *suppose the six one-step carriers*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\&\mathsf{ContinuousFunctionCarrier}(mid_4, l, mid_5, mod_l, cert_l),\\&\mathsf{ContinuousFunctionCarrier}(mid_5, m, target, mod_m, cert_m)\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}), \\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}), \\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}), \\
\mathsf{Cont}(fghk, l, fghkl), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_{FGHKL}), & \mathsf{Cont}(mid_5, mod_{FGHKL}, cert_{FGHKL}), \\
\mathsf{Cont}(fghkl, m, map_L), & \mathsf{Cont}(mod_{FGHKL}, mod_m, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(l, m, lm), & \mathsf{Cont}(mod_l, mod_m, mod_{LM}), & \mathsf{Cont}(target, mod_{LM}, cert_{LM}), \\
\mathsf{Cont}(k, lm, klm), & \mathsf{Cont}(mod_K, mod_{LM}, mod_{KLM}), & \mathsf{Cont}(target, mod_{KLM}, cert_{KLM}), \\
\mathsf{Cont}(h, klm, hklm), & \mathsf{Cont}(mod_H, mod_{KLM}, mod_{HKLM}), & \mathsf{Cont}(target, mod_{HKLM}, cert_{HKLM}), \\
\mathsf{Cont}(g, hklm, ghklm), & \mathsf{Cont}(mod_G, mod_{HKLM}, mod_{GHKLM}), & \mathsf{Cont}(target, mod_{GHKLM}, cert_{GHKLM}), \\
\mathsf{Cont}(f, ghklm, map_R), & \mathsf{Cont}(mod_F, mod_{GHKLM}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{array}{l}
\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge \\
\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge \\
\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{array}$$

Proof. First build the left-deep carrier. Repeatedly apply Theorem 59.7 along the displayed fg, fgh, $fghk$, $fghkl$, and map_L graph ledgers, using the matching modulus and certificate-continuation ledgers at each stage. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is built from the other end of the same chain. Compose the l and m carriers, then compose successively with the k, h, g, and f carriers through the displayed lm, klm, $hklm$, $ghklm$, and map_R ledgers. The result is $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_six to the common graph spine f, g, h, k, l, m. The left-deep spine ends at map_L and the right-deep spine ends at map_R, so sixfold continuation associativity yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same kernel theorem to $mod_F, mod_G, mod_H, mod_K, mod_l, mod_m$. The displayed modulus ledgers present the same sixfold spine with endpoints mod_L and mod_R, giving $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ share the same target source. Continuation respect for hsame, with reflexivity on $target$ and the recovered modulus sameness, identifies the terminal certificates and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the same concrete carrier ledger as in the fourfold and fivefold synchronization results. The stability field is sixfold left-right bracketing synchronization, and the exactness field is the pair of constructed carriers together with the three hsame conclusions. □

Theorem 59.152 (Continuous function carrier certificate semantic naming certificate)**.** *For BHist histories $source, map, target, modulus, cert$, any carrier witness* $\mathsf{ContinuousFunctionCarrier}(source, map, target, mo$ *induces the semantic naming certificate*

$$\mathsf{SemanticNameCert}(C, C, C, \mathsf{hsame}), \qquad C(c) := \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, c).$$

Thus the certificate-slot carrier, pattern, and ledger are all the displayed fiber $c \mapsto$ $\mathsf{ContinuousFunctionCarrier}(source$ *and the classifier is* hsame*.*

Proof. The displayed carrier gives the source witness for the semantic certificate at the point *cert*. This is exactly the inhabited source field for the fiber C.

The classifier fields are the standard hsame rows. Reflexivity classifies each displayed certificate with itself, symmetry reverses a displayed classification, and transitivity composes two displayed classifications.

Carrier transport is confined to the certificate slot. Keep *source*, *map*, *target*, and *modulus* fixed by hsame reflexivity, use the supplied hsame comparison of certificate histories, and apply Definition 59.14 to transport the continuous function carrier along that comparison.

The pattern and ledger predicates are the same fiber C as the source predicate. Their soundness and completeness maps are therefore identity projections, so the semantic naming certificate is repacked from the preceding fields. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier certificate semanticNameCert`

Theorem 59.153 (Continuous function carrier parallel composite determinacy)**.** *For BHist histories* $source, middle, target, f, g, fg, fg', mod_F, mod_G, mod_{FG}, mod'_{FG}, cert_F, cert_G, cert, cert'$*, suppose*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, middle, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(middle, g, target, mod_G, cert_G),\\
&\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(f, g, fg'),\\
&\mathsf{Cont}(mod_F, mod_G, mod_{FG}), \quad \mathsf{Cont}(mod_F, mod_G, mod'_{FG}),\\
&\mathsf{Cont}(target, mod_{FG}, cert), \quad \mathsf{Cont}(target, mod'_{FG}, cert')
\end{aligned}$$

hold. Then the two exposed composites are both continuous function carriers and their graph, modulus, and certificate histories are determined:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, fg, target, mod_{FG}, cert) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, fg', target, mod'_{FG}, cert') \wedge\\
&\mathsf{hsame}(fg, fg') \wedge \mathsf{hsame}(mod_{FG}, mod'_{FG}) \wedge \mathsf{hsame}(cert, cert').
\end{aligned}$$

Proof. First compose through the first visible ledger. The two given base carriers, together with $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(mod_F, mod_G, mod_{FG})$, and $\mathsf{Cont}(target, mod_{FG}, cert)$, satisfy the hypotheses of Theorem 59.7. This gives $\mathsf{ContinuousFunctionCarrier}(source, fg, target, mod_{FG}, cert)$.

The second displayed ledger has the same two base carriers. Applying the same composition theorem to $\mathsf{Cont}(f, g, fg')$, $\mathsf{Cont}(mod_F, mod_G, mod'_{FG})$, and $\mathsf{Cont}(target, mod'_{FG}, cert')$ gives $\mathsf{ContinuousFunctionCarrier}(source, fg', target, mod'_{FG}, cert')$.

Map determinacy is local to the shared graph rows. Continuation determinacy for the two rows $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g, fg')$ gives $\mathsf{hsame}(fg, fg')$; the same kernel determinacy applied to the two modulus rows gives $\mathsf{hsame}(mod_{FG}, mod'_{FG})$.

Finally compare the terminal certificate rows. They share the target history, and the recovered modulus sameness transports the row $\mathsf{Cont}(target, mod'_{FG}, cert')$ back across $\mathsf{Cont}(target, mod_{FG}, cert)$, yielding $\mathsf{hsame}(cert, cert')$. These five outputs are exactly the two composite carriers plus the three exposed history-sameness conclusions. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier parallel composite determinacy`

Theorem 59.154 (Continuous function carrier certificate semantic naming certificate)**.** *For BHist histories* $source, map, target, modulus, cert$*, any carrier witness* $\mathsf{ContinuousFunctionCarrier}(source, map, target, mo$ *induces the semantic naming certificate*

$$\mathsf{SemanticNameCert}(C, C, C, \mathsf{hsame}), \qquad C(c) := \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, c).$$

Thus the certificate-slot carrier, pattern, and ledger are all the displayed fiber $c \mapsto \mathsf{ContinuousFunctionCarrier}(source$ *and the classifier is* hsame.

Proof. The displayed carrier gives the source witness for the semantic certificate at the point *cert*. This is exactly the inhabited source field for the fiber C.

The classifier fields are the standard hsame rows. Reflexivity classifies each displayed certificate with itself, symmetry reverses a displayed classification, and transitivity composes two displayed classifications.

Carrier transport is confined to the certificate slot. Keep *source*, *map*, *target*, and *modulus* fixed by hsame reflexivity, use the supplied hsame comparison of certificate histories, and apply Definition 59.14 to transport the continuous function carrier along that comparison.

The pattern and ledger predicates are the same fiber C as the source predicate. Their soundness and completeness maps are therefore identity projections, so the semantic naming certificate is repacked from the preceding fields. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier certificate semanticNameCert`

Theorem 59.155 (Continuous function carrier sixfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, mid_5, target, f, g, h, k, l, m, fg, fgh, fghk, fghkl, lm, klm, hklm, ghklm, map_L$, *suppose the six one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\
&\mathsf{ContinuousFunctionCarrier}(mid_4, l, mid_5, mod_l, cert_l),\\
&\mathsf{ContinuousFunctionCarrier}(mid_5, m, target, mod_m, cert_m)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}),\\
\mathsf{Cont}(fghk, l, fghkl), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_{FGHKL}), & \mathsf{Cont}(mid_5, mod_{FGHKL}, cert_{FGHKL}),\\
\mathsf{Cont}(fghkl, m, map_L), & \mathsf{Cont}(mod_{FGHKL}, mod_m, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(l, m, lm), & \mathsf{Cont}(mod_l, mod_m, mod_{LM}), & \mathsf{Cont}(target, mod_{LM}, cert_{LM}),\\
\mathsf{Cont}(k, lm, klm), & \mathsf{Cont}(mod_K, mod_{LM}, mod_{KLM}), & \mathsf{Cont}(target, mod_{KLM}, cert_{KLM}),\\
\mathsf{Cont}(h, klm, hklm), & \mathsf{Cont}(mod_H, mod_{KLM}, mod_{HKLM}), & \mathsf{Cont}(target, mod_{HKLM}, cert_{HKLM}),\\
\mathsf{Cont}(g, hklm, ghklm), & \mathsf{Cont}(mod_G, mod_{HKLM}, mod_{GHKLM}), & \mathsf{Cont}(target, mod_{GHKLM}, cert_{GHKLM}),\\
\mathsf{Cont}(f, ghklm, map_R), & \mathsf{Cont}(mod_F, mod_{GHKLM}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Repeatedly apply Theorem 59.7 along the displayed fg, fgh, $fghk$, $fghkl$, and map_L graph ledgers, using the matching modulus and certificate-continuation ledgers at each stage. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is built from the other end of the same chain. Compose the l and m carriers, then compose successively with the k, h, g, and f carriers through the displayed lm, klm, $hklm$, $ghklm$, and map_R ledgers. The result is $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_six to the common graph spine f, g, h, k, l, m. The left-deep spine ends at map_L and the right-deep spine ends at map_R, so sixfold continuation associativity yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same kernel theorem to $mod_F, mod_G, mod_H, mod_K, mod_l, mod_m$. The displayed modulus ledgers present the same sixfold spine with endpoints mod_L and mod_R, giving $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ share the same target source. Continuation respect for hsame, with reflexivity on $target$ and the recovered modulus sameness, identifies the terminal certificates and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the same concrete carrier ledger as in the fourfold and fivefold synchronization results. The stability field is sixfold left-right bracketing synchronization, and the exactness field is the pair of constructed carriers together with the three hsame conclusions. □

Theorem 59.156 (Continuous function carrier fourfold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, target, f, g, h, k, fg, fgh, hk, ghk, map_L, map_R, mod_F, mod_G, mod_H, mod_K, mod_{FG}$ *suppose the four one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, target, mod_K, cert_K)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, map_L), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(h, k, hk), & \mathsf{Cont}(mod_H, mod_K, mod_{HK}), & \mathsf{Cont}(target, mod_{HK}, cert_{HK}),\\
\mathsf{Cont}(g, hk, ghk), & \mathsf{Cont}(mod_G, mod_{HK}, mod_{GHK}), & \mathsf{Cont}(target, mod_{GHK}, cert_{GHK}),\\
\mathsf{Cont}(f, ghk, map_R), & \mathsf{Cont}(mod_F, mod_{GHK}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then again with the k carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L,$

The right-deep carrier is obtained by the symmetric closure chain. Compose the h and k carriers through hk, mod_{HK}, and $cert_{HK}$; compose the g carrier with that composite through ghk, mod_{GHK}, and $cert_{GHK}$; then compose the f carrier with the resulting carrier to obtain $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply the BEDC fourfold continuation associativity theorem cont_assoc_four to the common graph spine f, g, h, k. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, and $\mathsf{Cont}(fgh, k, map_L)$, while the right spine is presented by $\mathsf{Cont}(h, k, hk)$, $\mathsf{Cont}(g, hk, ghk)$, and $\mathsf{Cont}(f, ghk, map_R)$. The underlying append-associativity spine append_assoc yields $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fourfold continuation associativity theorem to $mod_F, mod_G, mod_H, mod_K$. The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common append spine yields $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the certificate ledger to give $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is fourfold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the graph, modulus, and certificate histories displayed above. The stability field is fourfold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp associativity coherence and the later category bridge. □

Theorem 59.157 (Continuous function carrier fivefold left-right synchronization)**.** *For BHist histories* $source, mid_1, mid_2, mid_3, mid_4, target, f, g, h, k, l, fg, fgh, fghk, kl, hkl, ghkl, map_L, map_R, mod_F, mod_G, mo$ *suppose the five one-step carriers*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, f, mid_1, mod_F, cert_F),\\
&\mathsf{ContinuousFunctionCarrier}(mid_1, g, mid_2, mod_G, cert_G),\\
&\mathsf{ContinuousFunctionCarrier}(mid_2, h, mid_3, mod_H, cert_H),\\
&\mathsf{ContinuousFunctionCarrier}(mid_3, k, mid_4, mod_K, cert_K),\\
&\mathsf{ContinuousFunctionCarrier}(mid_4, l, target, mod_l, cert_l)
\end{aligned}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}),\\
\mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}),\\
\mathsf{Cont}(fgh, k, fghk), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_{FGHK}), & \mathsf{Cont}(mid_4, mod_{FGHK}, cert_{FGHK}),\\
\mathsf{Cont}(fghk, l, map_L), & \mathsf{Cont}(mod_{FGHK}, mod_l, mod_L), & \mathsf{Cont}(target, mod_L, cert_L)
\end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll}
\mathsf{Cont}(k, l, kl), & \mathsf{Cont}(mod_K, mod_l, mod_{KL}), & \mathsf{Cont}(target, mod_{KL}, cert_{KL}),\\
\mathsf{Cont}(h, kl, hkl), & \mathsf{Cont}(mod_H, mod_{KL}, mod_{HKL}), & \mathsf{Cont}(target, mod_{HKL}, cert_{HKL}),\\
\mathsf{Cont}(g, hkl, ghkl), & \mathsf{Cont}(mod_G, mod_{HKL}, mod_{GHKL}), & \mathsf{Cont}(target, mod_{GHKL}, cert_{GHKL}),\\
\mathsf{Cont}(f, ghkl, map_R), & \mathsf{Cont}(mod_F, mod_{GHKL}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R)
\end{array}$$

hold. Then the left-deep and right-deep composite carriers agree at the public graph, modulus, and certificate histories:

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L) \wedge\\
&\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R) \wedge\\
&\mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R).
\end{aligned}$$

Proof. First build the left-deep carrier. Apply Theorem 59.7 to the f and g carriers with the displayed fg, mod_{FG}, and $cert_{FG}$ ledgers. Apply the same closure theorem to compose that result with the h carrier, then with the k carrier, and finally with the l carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_L, target, mod_L, cert_L)$.

The right-deep carrier is obtained by the matching closure chain. Compose the k and l carriers through kl, mod_{KL}, and $cert_{KL}$; compose the h carrier with that composite through hkl, mod_{HKL}, and $cert_{HKL}$; compose the g carrier with the resulting carrier through $ghkl$, mod_{GHKL}, and $cert_{GHKL}$; then compose the f carrier with that final right-deep carrier. This gives $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$.

For graph synchronization, apply BEDC.FKernel.Cont.cont_assoc_five to the common graph spine f, g, h, k, l. The left spine is presented by $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(fg, h, fgh)$, $\mathsf{Cont}(fgh, k, fghk)$, and $\mathsf{Cont}(fghk, l, map_L)$, while the right spine is presented by $\mathsf{Cont}(k, l, kl)$, $\mathsf{Cont}(h, kl, hkl)$, $\mathsf{Cont}(g, hkl, ghkl)$, and $\mathsf{Cont}(f, ghkl, map_R)$. The fivefold continuation reassociation gives $\mathsf{hsame}(map_L, map_R)$.

For modulus synchronization, apply the same fivefold reassociation theorem to $mod_F, mod_G, mod_H, mod_K, mod_l$ The left modulus ledger ends at mod_L and the right modulus ledger ends at mod_R, so the common continuation spine gives $\mathsf{hsame}(mod_L, mod_R)$.

It remains to compare the terminal certificates. The ledgers $\mathsf{Cont}(target, mod_L, cert_L)$ and $\mathsf{Cont}(target, mod_R, cert_R)$ have the same target source. Continuation respect for hsame, using reflexivity on $target$ and the recovered modulus synchronization, transports the terminal certificate ledger and gives $\mathsf{hsame}(cert_L, cert_R)$.

The ten-field certificate fit is the following. The traditional target is pentagon-style fivefold continuous composition coherence. The BEDC source is $BHist$ with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the five one-step graph histories, their modulus histories, their target-certificate histories, and the displayed left-deep and right-deep composite ledgers. The stability field is fivefold left-right bracketing synchronization. The exactness field is the pair of constructed carriers together with the three hsame conclusions. The downstream use is ContinuousMapUp pentagon coherence and the category bridge. □

Theorem 59.158 (Continuous modulus witness right e1 result exactness)**.** *For histories $source, modulus, target$, a modulus witness whose modulus and target histories are both visibly tagged has an exact untagged tail:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow \mathsf{ContinuousModulusWitness}(source, modulus, tar$$

Proof. Unpack the source, modulus, target unary witnesses and the continuation ledger. Unary inversion removes the visible constructor from the modulus and target carriers. The right E1 continuation inversion removes the same constructor from the continuation ledger, yielding the exact untagged modulus witness. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 result exactness`

Theorem 59.159 (Continuous modulus witness right e1 visible source exactness)**.** *For histories $source, modulus, target, sourceTail$, a modulus witness whose modulus and target histories are both visibly tagged descends to the tail while respecting a visible presentation of the source endpoint:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow \mathsf{hsame}(source, \mathsf{E1}(sourceTail)) \Longrightarrow \mathsf{ContinuousM}$$

Proof. Apply right one-headed result exactness to strip the visible constructor from the modulus and target components of the witness. The displayed hsame presentation of the source endpoint then transports the descended witness to the concrete visible source $\mathsf{E1}(sourceTail)$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 visible source exactness`

Theorem 59.160 (Continuous modulus witness right e1 source cases)**.** *For histories source, modulus, target, a modulus witness whose modulus and target histories are both visibly tagged has only the empty source case or a visible source tail:*

$$\mathsf{ContinuousModulusWitness}(source, \mathsf{E1}(modulus), \mathsf{E1}(target)) \Longrightarrow (source = \mathsf{e} \wedge \mathsf{UnaryHistory}(modulus) \wedge \mathsf{hsame}(tar$$

Proof. First descend the witness by stripping the common visible constructor from the modulus and target histories. The descended source carrier is unary, so it is either empty or visibly tagged. In the empty case, the continuation ledger is a left-unit ledger and gives $\mathsf{hsame}(target, modulus)$. In the visible case, constructor equality identifies the source with $\mathsf{E1}(sourceTail)$ and the descended witness is repacked with that source presentation. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusWitness right e1 source cases`

Definition 59.161 (ContinuousMap stability certificate)**.** *The stability certificate consists of:*

1. *continuous-map classifier reflexivity*
2. *continuous-map classifier symmetry*
3. *continuous-map classifier transitivity*
4. *modulus validity for uniform continuity*
5. *graph and modulus replacement under the continuous-map classifier*
6. *closure under identity*
7. *closure under composition by composing moduli*

Theorem 59.162 (Continuous function carrier terminal modulus extension)**.** *For histories source, map, target, mo a continuous-function carrier remains a continuous-function carrier after the terminal certificate is extended by one more modulus witness and the modulus ledger is composed:*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert, extra, cert') =$$

Proof. Unpack the given carrier. This exposes the graph ledger $\mathsf{Cont}(source, map, target)$ and the terminal modulus-certificate ledger $\mathsf{Cont}(target, modulus, cert)$, together with the unary carrier data for the displayed histories.

Unpack the additional modulus witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$. It supplies $\mathsf{Cont}(cert, extra, cert')$, so the two ledgers $\mathsf{Cont}(target, modulus, cert)$ and $\mathsf{Cont}(cert, extra, cert')$ assemble the chain $\mathsf{ContinuousModulusChain}(target, modulus, extra, cert')$ with middle history $cert$.

Apply Theorem 59.108 to the original carrier, this chain, and the displayed composition ledger $\mathsf{Cont}(modulus, extra, modulus')$. The replacement theorem repacks the preserved graph data with the composite terminal modulus certificate, giving the displayed carrier conclusion.

The ten-field certificate fit is the following. The traditional target is terminal modulus extension for continuous maps. The BEDC source is BHist with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is preservation under one terminal modulus extension. The ledger is source, map, target, original modulus and certificate, extra modulus, composite modulus, and terminal certificate. The core theorem is this closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier terminal modulus extension`

Theorem 59.163 (Continuous function carrier visible terminal modulus extension)**.** *For histories $p, q, source, map, target, modulus, cert, extra, modulus', cert'$, visible continuous-function carrier data remain carrier data after the central terminal certificate is extended and the terminal modulus ledger is composed:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert, extr$$

Proof. Peel the visible presentation by Theorem 59.169. This recovers the unary context histories $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(q)$ together with the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target$

Apply Theorem 59.162 to the central carrier, the witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$, and the composition ledger $\mathsf{Cont}(modulus, extra, modulus')$. The conclusion is the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus', cert')$.

Use the reverse direction of Theorem 59.169 with the recovered unary histories. It repacks the same prefix p and suffix q around the central carrier, producing the displayed visible carrier conclusion. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension`

Theorem 59.164 (Continuous function carrier visible terminal modulus extension output determinacy)**.** *For histories $p, q, source, map, target, target', modulus, cert, extra, modulus', cert', cert''$, suppose*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

holds, together with $\mathsf{ContinuousModulusWitness}(cert, extra, cert')$ *and* $\mathsf{Cont}(modulus, extra, modulus')$*. If the post-extension visible carrier*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus' \cdot q, (p \cdot cert'') \cdot q)$$

also holds, then the visible target and terminal certificate are determined:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert', cert'').$$

Proof. Apply Theorem 59.163 to the original visible carrier, the central witness $\mathsf{ContinuousModulusWitness}(cert, ext$ and the terminal modulus ledger $\mathsf{Cont}(modulus, extra, modulus')$. This constructs the canonical post-extension visible carrier with target $target$ and terminal certificate $cert'$.

Compare that canonical carrier with the independently displayed post-extension carrier by Theorem 59.174. The two carrier presentations share p, q, $source$, map, and $modulus'$, so the determinacy theorem returns exactly $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert', cert'')$. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension output determinacy`

Theorem 59.165 (Continuous function carrier visible terminal modulus extension transitive)**.** *For histories $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2$, visible continuous-function carrier data remain carrier data after two consecutive terminal certificate extensions and their two terminal modulus ledgers:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus_0{\cdot}q, (p{\cdot}cert_0){\cdot}q) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_0, e$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus extension transitive`

Proof. Apply Theorem 59.163 to the displayed visible carrier, the witness $\mathsf{ContinuousModulusWitness}(cert_0, extra_1,$ and the first composition ledger $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$. This gives the visible carrier with terminal modulus $modulus_1$ and terminal certificate $cert_1$.

Apply the same theorem again to that intermediate carrier, the witness $\mathsf{ContinuousModulusWitness}(cert_1, extra_2$ and the second composition ledger $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$. The result is the visible carrier whose terminal modulus is $modulus_2$ and whose terminal certificate is $cert_2$, which is the displayed conclusion.

The ten-field certificate fit is the following. The traditional target is two-step terminal modulus extension for continuous maps. The BEDC source is BHist with Cont, ContinuousModulusWitness, and ContinuousFunctionCarrier. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is preservation under two consecutive terminal modulus extensions. The ledger is the two extra modulus witnesses and the two composite modulus rows. The core theorem is this closure, exactness is the displayed carrier conclusion, the standard bridge is deferred, and the status is ProofTarget. □

Theorem 59.166 (Continuous function carrier visible terminal modulus reassociation)**.** *For histories* $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, extra_{12}, modulus_2, modulus_{12}, cert_2$ *sequential two-step terminal extension and the corresponding composite one-step terminal extension produce the same visible terminal modulus data:*

$$\begin{aligned}
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\
&\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\
&\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\
&\mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_2 \cdot q, (p \cdot cert_2) \cdot q) \wedge \\
&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{12} \cdot q, (p \cdot cert_2) \cdot q) \wedge \\
&\mathsf{hsame}(modulus_2, modulus_{12}).
\end{aligned}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation`

Proof. Apply Theorem 59.165 to the displayed visible carrier, the two certificate-extension witnesses, and the two sequential terminal modulus ledgers. This constructs the sequential visible carrier with terminal data $(modulus_2, cert_2)$.

Next assemble the certificate witnesses $\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1)$ and ContinuousModulusW into $\mathsf{ContinuousModulusChain}(cert_0, extra_1, extra_2, cert_2)$ with middle history $cert_1$. By Theorem 59.105 and the ledger $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, this gives the composite witness ContinuousModulusWitness

Apply Theorem 59.163 to the original visible carrier, this composite witness, and $\mathsf{Cont}(modulus_0, extra_{12}, modu$ The result is the composite one-step visible carrier with terminal data $(modulus_{12}, cert_2)$.

Finally use Theorem 4.56 on $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$, $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$, $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, and $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$. It identifies the sequential and composite terminal moduli, giving $\mathsf{hsame}(modulus_2, modulus_{12})$, and the three displayed conclusions are then packed together. □

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation`

Theorem 59.167 (Continuous function carrier visible terminal modulus reassociation output determinacy)**.** *For histories* $p, q, source, map, target, target', modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, extra_1$

the displayed composite visible output is determined by the reassociated terminal modulus construction:

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\ &\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\ &\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\ &\mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \\ &\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus_{12} \cdot q, (p \cdot cert'') \cdot q) \Longrightarrow \\ &\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_2, cert'') \wedge \mathsf{hsame}(modulus_2, modulus_{12}).\end{aligned}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible terminal modulus reassociation output determinacy`

Proof. First apply Theorem 59.166 to the original visible carrier, the two certificate-extension witnesses, and the four terminal modulus ledgers. This supplies the canonical composite visible carrier $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{12} \cdot q, (p \cdot cert_2) \cdot q)$ together with $\mathsf{hsame}(modulus_2, modulus_{12})$.

Next compare that canonical composite carrier with the independently displayed composite carrier $\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus_{12} \cdot q, (p \cdot cert'') \cdot q)$. By Theorem 59.174, the common visible source, map, and terminal modulus determine the public target and certificate, giving $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert_2, cert'')$.

Finally package those two target-certificate comparisons with the modulus comparison obtained from reassociation. The result is the displayed triple $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert_2, cert'') \wedge \mathsf{hsame}(modulus_2, modulus_{12})$. □

Theorem 59.168 (Continuous function carrier visible terminal modulus three-step reassociation)**.** *For histories $p, q, source, map, target, modulus_0, cert_0, extra_1, modulus_1, cert_1, extra_2, modulus_2, cert_2, extra_3, mo$ sequential three-step terminal extension and the corresponding composite one-step terminal extension produce the same visible terminal modulus data:*

$$\begin{aligned}&\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_0 \cdot q, (p \cdot cert_0) \cdot q) \Longrightarrow \\ &\mathsf{ContinuousModulusWitness}(cert_0, extra_1, cert_1) \Longrightarrow \mathsf{ContinuousModulusWitness}(cert_1, extra_2, cert_2) \Longrightarrow \\ &\mathsf{ContinuousModulusWitness}(cert_2, extra_3, cert_3) \Longrightarrow \\ &\mathsf{Cont}(modulus_0, extra_1, modulus_1) \Longrightarrow \mathsf{Cont}(modulus_1, extra_2, modulus_2) \Longrightarrow \\ &\mathsf{Cont}(modulus_2, extra_3, modulus_3) \Longrightarrow \mathsf{Cont}(extra_1, extra_2, extra_{12}) \Longrightarrow \\ &\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12}) \Longrightarrow \mathsf{Cont}(extra_{12}, extra_3, extra_{123}) \Longrightarrow \\ &\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123}) \Longrightarrow \\ &\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_3 \cdot q, (p \cdot cert_3) \cdot q) \wedge \\ &\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus_{123} \cdot q, (p \cdot cert_3) \cdot q) \wedge \\ &\mathsf{hsame}(modulus_3, modulus_{123}).\end{aligned}$$

Proof. Apply Theorem 59.165 to the displayed visible carrier, the first two certificate-extension witnesses, and the first two sequential terminal modulus ledgers. This gives the visible carrier with terminal data $(modulus_2, cert_2)$. Apply Theorem 59.163 to that carrier, the third certificate-extension witness, and $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$. The sequential visible carrier with terminal data $(modulus_3, cert_3)$ follows.

Next compose the first two certificate witnesses. They form the chain $\mathsf{ContinuousModulusChain}(cert_0, extra_1, ext$ with middle history $cert_1$, so Theorem 59.105 and $\mathsf{Cont}(extra_1, extra_2, extra_{12})$ supply $\mathsf{ContinuousModulusWitness}($

Compose that witness with the third one. The chain $\mathsf{ContinuousModulusChain}(cert_0, extra_{12}, extra_3, cert_3)$ with middle history $cert_2$, together with $\mathsf{Cont}(extra_{12}, extra_3, extra_{123})$, gives $\mathsf{ContinuousModulusWitness}(cert_0, ext$ by the same composite-closure theorem.

Apply Theorem 59.163 to the original visible carrier, the composite witness just constructed, and $\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123})$. This produces the composite one-step visible carrier with terminal data $(modulus_{123}, cert_3)$.

It remains to identify the two terminal moduli. First use Theorem 4.56 on $\mathsf{Cont}(modulus_0, extra_1, modulus_1)$, $\mathsf{Cont}(modulus_1, extra_2, modulus_2)$, $\mathsf{Cont}(extra_1, extra_2, extra_{12})$, and $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$, obtaining $\mathsf{hsame}(modulus_2, modulus_{12})$.

Transport the third terminal continuation along that comparison, carrying $\mathsf{Cont}(modulus_2, extra_3, modulus_3)$ to the row $\mathsf{Cont}(modulus_{12}, extra_3, modulus_3)$. Then apply Theorem 4.56 to $\mathsf{Cont}(modulus_0, extra_{12}, modulus_{12})$, the transported row, $\mathsf{Cont}(extra_{12}, extra_3, extra_{123})$, and $\mathsf{Cont}(modulus_0, extra_{123}, modulus_{123})$. The resulting comparison is $\mathsf{hsame}(modulus_3, modulus_{123})$, and the two visible carriers and this comparison form the displayed triple. □

Theorem 59.169 (Continuous function carrier visible modulus context exactness)**.** *For histories $p, q, source, map, target, modulus, cert$, a visible prefix on the source, target, and certificate ledgers together with a visible suffix on the modulus and certificate ledgers is exactly unary evidence for the two context histories together with the central continuous-function carrier:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, map, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Co}$$

Proof. Peel the right suffix from the modulus and certificate ledgers by modulus suffix exactness. The remaining carrier has the common left prefix on source, target, and certificate, so carrier prefix exactness recovers the unary prefix and the central carrier. Conversely, rebuild the prefixed carrier first, then reattach the modulus suffix. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context iff`

Theorem 59.170 (Continuous function carrier visible-context empty-map exactness)**.** *For histories $p, q, source, target, modulus, cert$, the empty graph map remains exact under visible prefix and suffix ledgers:*

$$\mathsf{ContinuousFunctionCarrier}(p{\cdot}source, \mathsf{e}, p{\cdot}target, modulus{\cdot}q, (p{\cdot}cert){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map exactness`

Proof. For the forward direction, apply Theorem 59.169 to strip the visible prefix from source, target, and certificate and the visible suffix from modulus and certificate. The exposed central carrier has empty map, so Theorem 59.13 gives $\mathsf{hsame}(target, source)$ and the central modulus witness.

For the reverse direction, start from unary context evidence, the *target–source* classifier, and the central modulus witness. Use Theorem 59.13 to rebuild the central empty-map carrier, then apply Theorem 59.169 to restore the visible prefix and suffix ledgers.

The ten-field certificate fit is the identity uniformly continuous graph readback over the $BHist/\mathsf{Cont}$ source, with ContinuousFunctionCarrier as carrier, hsame as classifier, visible prefix/suffix stability as the context law, the empty graph map as the readback field, central modulus witness exactness as the certificate content, source and target ledgers classified by hsame, modulus and certificate ledgers preserved by the suffix, and ContinuousMapUp identity-map use downstream. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map iff`

↪ variant `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context empty map exactness`

Theorem 59.171 (Continuous modulus chain visible context exactness)**.** *For histories $p, q, source, first, second, t$ a visible prefix on the source and target together with a visible suffix on the second modulus and target is exactly unary evidence for the two contexts together with the central two-link chain:*

$$\mathsf{ContinuousModulusChain}(p{\cdot}source, first, second{\cdot}q, (p{\cdot}target){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Continuous}$$

Proof. Peel the right suffix from the second modulus and target by chain suffix exactness. The remaining chain has the common left prefix on source and target, so chain prefix exactness exposes unary p and the central chain. Conversely, rebuild the prefixed chain first, then reattach the suffix. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain visible context iff`

Theorem 59.172 (Continuous modulus chain visible context second determinacy)**.** *For histories $p, q, source, first, second, second', target$, if two visible-context two-link modulus chains share the same visible source, first modulus, and visible target,*

$$\mathsf{ContinuousModulusChain}(p{\cdot}source, first, second{\cdot}q, (p{\cdot}target){\cdot}q) \quad \textit{and} \quad \mathsf{ContinuousModulusChain}(p{\cdot}source, first, s$$

then secondhsame*second'*.

Proof. Apply the visible-context exactness statement to both chains. This removes the common prefix and suffix contexts and exposes two central chains with the same source, first modulus, and target. The ordinary second-modulus determinacy theorem identifies the two central second moduli. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousModulusChain visible context second deterministic`

Theorem 59.173 (Continuous function carrier visible-context field determinacy)**.** *For histories $p, q, source, source', map, map', target, target', modulus, modulus', cert, cert'$, visible-modulus-context continuous-function carriers determine each central public field as follows.*

1. *Same visible source, target, modulus, and certificate determine the map:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

2. *Same visible map, target, modulus, and certificate determine the source:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

3. *Same visible source, map, target, and certificate determine the modulus:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

4. *Same visible source, map, and modulus determine the target and certificate:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{ContinuousFunctionCarrier}(p$$

Proof. 1. In the map clause, apply Theorem 59.169 to the two displayed visible carriers. The exposed central carriers share source, target, modulus, and certificate, so Theorem 59.17 gives $\mathsf{hsame}(map, map')$.

2. In the source clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share map, target, modulus, and certificate, so Theorem 59.18 gives $\mathsf{hsame}(source, source')$.
3. In the modulus clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share source, map, target, and certificate, so Theorem 59.19 gives $\mathsf{hsame}(modulus, modulus')$.
4. In the target-certificate clause, apply the same exactness theorem to the two displayed visible carriers. The exposed central carriers share source, map, and modulus, so Theorem 59.15 gives $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert')$.

□

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context field determinacy`

Theorem 59.174 (Continuous function carrier visible-context target-certificate determinacy)**.** *For histories $p, q, source, map, target, target', modulus, cert, cert'$, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target', modulus \cdot q, (p \cdot cert') \cdot q)$$

hold, then the public target and certificate ledgers are determined by the central presentation:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.169 to the first visible carrier. This recovers the unary context histories $\mathsf{UnaryHistory}(p)$ and $\mathsf{UnaryHistory}(q)$ together with the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, targe$

Apply the same exactness theorem to the second visible carrier. The common visible prefix and suffix are peeled in the same way, producing the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, target', mod$

Now apply Theorem 59.15 to the two central carrier witnesses. It gives the displayed $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert, cert')$ conclusions, which are already the public target and certificate comparisons required here.

The ten-field certificate fit is the following. The traditional target is public target-certificate determinacy for visible continuous-function carrier presentations. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context public readback. The ledger is source, map, target, modulus, and certificate histories together with the visible prefix and suffix contexts. The core theorem is this result. The exactness predicate is the displayed pair of hsame conclusions. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context target cert deterministic`

Theorem 59.175 (Continuous function carrier visible-context map determinacy)**.** *For histories $p, q, source, map, map', target, modulus, cert$, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map', p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

hold, then the public graph map ledger is determined by the central presentation:

$$\mathsf{hsame}(map, map').$$

Proof. Apply Theorem 59.169 to the first visible carrier. This removes the shared visible prefix and suffix contexts and exposes the central witness $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert)$.

Apply the same visible-context exactness theorem to the second carrier. The same source, target, modulus, and certificate ledgers remain, producing $\mathsf{ContinuousFunctionCarrier}(source, map', target, modulus, ce$

Now apply Theorem 59.17 to the two central carrier witnesses. It gives the displayed $\mathsf{hsame}(map, map')$ comparison, which is already the public visible-context map readback required here.

The ten-field certificate fit is the following. The traditional target is visible map-ledger uniqueness. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context map readback. The ledger is source, map, target, modulus, and certificate histories together with the visible prefix and suffix contexts. The core theorem is this result. The exactness predicate is the displayed hsame conclusion. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context map deterministic`

Theorem 59.176 (Continuous function carrier visible-context graph-modulus determinacy)**.** *For histories $p, q, source, map, map', target, modulus, modulus', cert$, if both visible-modulus-context carrier presentations*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map', p \cdot target, modulus' \cdot q, (p \cdot cert) \cdot q)$$

hold, then the public graph and modulus ledgers are simultaneously determined:

$$\mathsf{hsame}(map, map') \wedge \mathsf{hsame}(modulus, modulus').$$

Proof. Apply Theorem 59.169 to both visible carrier hypotheses. The common displayed prefix and suffix are peeled away, leaving central carriers over $(source, map, target, modulus, cert)$ and $(source, map', target, modulus', cert)$.

Apply Theorem 59.139 to the two central carriers. This exposes graph ledgers $\mathsf{Cont}(source, map, target)$ and $\mathsf{Cont}(source, map', target)$, and certificate ledgers $\mathsf{Cont}(target, modulus, cert)$ and $\mathsf{Cont}(target, modulus', cert)$

Continuation left cancellation on the two graph ledgers gives $\mathsf{hsame}(map, map')$. The same cancellation principle on the two target-certificate ledgers gives $\mathsf{hsame}(modulus, modulus')$, and the two comparisons are paired as the displayed conclusion.

The ten-field certificate fit is: traditional target visible continuous-function determinacy; BEDC source BHist with Cont; carrier ContinuousFunctionCarrier; classifier hsame; stability obligation visible-context public readback; ledger source, map, target, modulus, certificate, prefix, and suffix; core theorem this statement; exactness the displayed pair of hsame conclusions; standard bridge deferred; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context graph modulus deterministic`

Theorem 59.177 (Continuous function carrier visible-context cycle tails empty)**.** *For histories $p, q, source, map, target, modulus, cert$, a visible-context continuous-function carrier whose central source and certificate are history-same has empty central graph and modulus tails:*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q) \Longrightarrow \mathsf{hsame}(source, cert) \Longrightarrow \mathsf{hsame}$$

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context cycle tails empty`

Proof. Apply Theorem 59.169 to the displayed visible carrier. This recovers unary witnesses for the visible prefix p and suffix q, and it exposes the central carrier $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulu$

Apply Theorem 59.140 to the central carrier and the displayed $\mathsf{hsame}(source, cert)$ cycle. The endpoint-certificate cycle cancellation theorem gives exactly $\mathsf{hsame}(map, \mathsf{e}) \wedge \mathsf{hsame}(modulus, \mathsf{e})$.

The ten-field certificate fit is the following. The traditional target is visible cycle-tail pruning. The BEDC source is BHist with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is visible-context central readback. The ledger is source, map, target, modulus, certificate, prefix, and suffix histories. The core theorem is this statement. The exactness predicate is the displayed pair of empty-tail conclusions. The standard bridge is deferred, and the status is ProofTarget. □

Theorem 59.178 (Continuous function carrier visible-context composition closure)**.** *For histories $p, q, source, middle, target, f, g, fg, modF, modG, modFG, certF, certG, cert$, suppose the two contextualized carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold. If the graph, modulus, and target-certificate ledgers

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(target, modFG, cert)$$

hold, then the composite carrier remains inside the same visible context:

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, fg, p \cdot target, modFG \cdot q, (p \cdot cert) \cdot q).$$

Proof. Apply Theorem 59.169 to the first contextualized carrier. This recovers unary witnesses for the visible prefix p and suffix q, together with the central carrier $\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF,$

Apply the same visible-context exactness theorem to the second contextualized carrier. The common contexts are peeled away, exposing $\mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, certG)$ over the same central middle and target histories.

Apply Theorem 59.7 to these two central carriers with the displayed graph, modulus, and target-certificate ledgers. The result is the central composite carrier $\mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG,$

Finally use Theorem 59.169 in the reverse direction, with the recovered unary witnesses for p and q, to reattach the visible prefix and suffix. This yields the displayed composite carrier in the same visible context.

The ten-field certificate fit is the following. The traditional target is composition closure for contextual continuous-function carriers. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is contextual composition closure with fixed visible prefix and suffix. The ledger consists of source, middle,

target, graph map, modulus, certificate, visible prefix, and visible suffix histories. The core theorem is this visible-context closure statement. The exactness claim is the reverse use of visible-context exactness after central composition closure. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context comp central package`

Theorem 59.179 (Continuous function carrier visible-context composition public readback)**.** *For histories* $p, q, source, middle, target, target', f, g, fg, modF, modG, modFG, certF, certG, cert, cert'$*, suppose the two contextualized carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold, together with the graph, modulus, and target-certificate ledgers

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(target, modFG, cert).$$

If a displayed visible composite carrier

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, fg, p \cdot target', modFG \cdot q, (p \cdot cert') \cdot q)$$

also holds, then the public composite target and certificate read back to the canonical central pair:

$$\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.169 to the first contextualized carrier. This peels the common visible prefix and suffix, and it exposes the central carrier $\mathsf{ContinuousFunctionCarrier}(source, f, middle, modF, certF)$.

Apply the same visible-context exactness theorem to the second contextualized carrier. The central carrier $\mathsf{ContinuousFunctionCarrier}(middle, g, target, modG, certG)$ is recovered over the same middle history and canonical target history.

Apply the visible-context exactness theorem once more to the displayed composite carrier. This removes the displayed visible context and gives the central displayed carrier $\mathsf{ContinuousFunctionCarrier}(source, fg, ta$

Now apply Theorem 59.16 to the two central input carriers, the three displayed continuation ledgers, and the central displayed composite carrier. The central public readback theorem gives exactly $\mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert')$.

The ten-field certificate fit is the following. The traditional target is public readback for contextual continuous-function composition. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is contextual composite target and certificate readback. The ledger consists of source, middle, target, displayed target, graph map, modulus, certificate, displayed certificate, visible prefix, and visible suffix histories. The core theorem is this visible-context readback statement. The exactness claim is the central target-certificate pair returned after peeling visible context. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composition public readback`

Theorem 59.180 (Continuous function carrier visible-context composite ledger determinacy)**.** *For histories* $p, q, source, middle, target, f, g, fg, fg', modF, modG, modFG, modFG', certF, certG, cert, cert'$*, suppose the two visible-context input carriers*

$$\mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q)$$

and

$$\mathsf{ContinuousFunctionCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q)$$

hold. If both composite ledger triples

$$\begin{array}{lll} \mathsf{Cont}(f, g, fg), & \mathsf{Cont}(modF, modG, modFG), & \mathsf{Cont}(target, modFG, cert), \\ \mathsf{Cont}(f, g, fg'), & \mathsf{Cont}(modF, modG, modFG'), & \mathsf{Cont}(target, modFG', cert') \end{array}$$

hold, then the public composite graph, composite modulus, and composite certificate ledgers are independent of the chosen admissible triple:

$$\mathsf{hsame}(fg, fg') \wedge \mathsf{hsame}(modFG, modFG') \wedge \mathsf{hsame}(cert, cert').$$

Proof. Apply Theorem 59.178 to the two input carriers and the first composite ledger triple. This builds the visible composite carrier over $(fg, modFG, cert)$. Apply the same closure theorem to the same input carriers and the second composite ledger triple, giving the visible composite carrier over $(fg', modFG', cert')$.

Apply continuation determinacy, Theorem 18.16, to the two graph ledgers $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g, fg')$. Apply the same determinacy theorem to the two modulus ledgers $\mathsf{Cont}(modF, modG, modFG)$ and $\mathsf{Cont}(modF, modG, modFG')$. These give $\mathsf{hsame}(fg, fg')$ and $\mathsf{hsame}(modFG, modFG')$.

Use Theorem 59.169 to peel the visible prefix and suffix from the two composite carriers. Then use Theorem 59.14 to place the two central composite carriers over a common graph and modulus representative while leaving the target fixed.

Now Theorem 59.15 applies to the aligned central carriers. Its certificate conclusion gives $\mathsf{hsame}(cert, cert')$, which combines with the graph and modulus comparisons above to give the displayed triple classifier.

The ten-field certificate fit is the following. The traditional target is composite-ledger well-definedness for continuous maps. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The stability obligation is composite ledger independence. The ledger consists of source, middle, target, map, modulus, certificate, prefix, and suffix histories. The core theorem is this determinacy result. The exactness claim is the displayed triple classifier. The downstream consumer is ContinuousMapUp composition well-definedness. The standard bridge is deferred, and the status is ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composite ledger determinacy`
Lean4 ✓ `BEDC.Derived.ContinuousUp.ContinuousFunctionCarrier visible modulus context composite ledger deterministic`

Theorem 59.181 (Continuous function carrier visible fourfold left-right synchronization)**.** *For BHist histories* $p, q, source, mid_1, mid_2, mid_3, target, f, g, h, k, fg, fgh, hk, ghk, map_L, map_R, mod_F, mod_G, mod_H,$ *suppose the four visible-context one-step carriers*

$$\begin{array}{l} \mathsf{ContinuousFunctionCarrier}(p \cdot source, f, p \cdot mid_1, mod_F \cdot q, (p \cdot cert_F) \cdot q), \\ \mathsf{ContinuousFunctionCarrier}(p \cdot mid_1, g, p \cdot mid_2, mod_G \cdot q, (p \cdot cert_G) \cdot q), \\ \mathsf{ContinuousFunctionCarrier}(p \cdot mid_2, h, p \cdot mid_3, mod_H \cdot q, (p \cdot cert_H) \cdot q), \\ \mathsf{ContinuousFunctionCarrier}(p \cdot mid_3, k, p \cdot target, mod_K \cdot q, (p \cdot cert_K) \cdot q) \end{array}$$

hold. Suppose also that the left-deep ledgers

$$\begin{array}{lll} \mathsf{Cont}(f, g, fg), & \mathsf{Cont}(mod_F, mod_G, mod_{FG}), & \mathsf{Cont}(mid_2, mod_{FG}, cert_{FG}), \\ \mathsf{Cont}(fg, h, fgh), & \mathsf{Cont}(mod_{FG}, mod_H, mod_{FGH}), & \mathsf{Cont}(mid_3, mod_{FGH}, cert_{FGH}), \\ \mathsf{Cont}(fgh, k, map_L), & \mathsf{Cont}(mod_{FGH}, mod_K, mod_L), & \mathsf{Cont}(target, mod_L, cert_L) \end{array}$$

hold, and that the right-deep ledgers

$$\begin{array}{lll} \mathsf{Cont}(h, k, hk), & \mathsf{Cont}(mod_H, mod_K, mod_{HK}), & \mathsf{Cont}(target, mod_{HK}, cert_{HK}), \\ \mathsf{Cont}(g, hk, ghk), & \mathsf{Cont}(mod_G, mod_{HK}, mod_{GHK}), & \mathsf{Cont}(target, mod_{GHK}, cert_{GHK}), \\ \mathsf{Cont}(f, ghk, map_R), & \mathsf{Cont}(mod_F, mod_{GHK}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R) \end{array}$$

hold. Then the two visible-context composite carriers exist and their public graph, modulus, and certificate histories agree:

$$\begin{array}{l} \mathsf{ContinuousFunctionCarrier}(p \cdot source, map_L, p \cdot target, mod_L \cdot q, (p \cdot cert_L) \cdot q) \wedge \\ \mathsf{ContinuousFunctionCarrier}(p \cdot source, map_R, p \cdot target, mod_R \cdot q, (p \cdot cert_R) \cdot q) \wedge \\ \mathsf{hsame}(map_L, map_R) \wedge \mathsf{hsame}(mod_L, mod_R) \wedge \mathsf{hsame}(cert_L, cert_R). \end{array}$$

Proof. Apply Theorem 59.169 to each of the four displayed visible-context one-step carriers. This peels the common visible prefix p from source, target, and certificate ledgers and the common visible suffix q from modulus and certificate ledgers. The result is unary evidence for the two context histories together with the four central carriers from *source* through *target*.

Apply Theorem 59.156 to those four central carriers and to the displayed left-deep and right-deep ledger arrays. The central theorem constructs the left-deep carrier $\mathsf{ContinuousFunctionCarrier}(source, map_L, ta$ the right-deep carrier $\mathsf{ContinuousFunctionCarrier}(source, map_R, target, mod_R, cert_R)$, and the three public synchronizations $\mathsf{hsame}(map_L, map_R)$, $\mathsf{hsame}(mod_L, mod_R)$, and $\mathsf{hsame}(cert_L, cert_R)$.

Use Theorem 59.169 in the reverse direction with the recovered unary witnesses for p and q to reattach the visible context to the left-deep central carrier. Use the same reverse implication for the right-deep central carrier. These two applications give exactly the two displayed visible-context composite carriers.

The ten-field certificate fit is the following. The traditional target is fourfold contextual continuous composition coherence. The BEDC source is BHist equipped with Cont. The carrier is ContinuousFunctionCarrier and the classifier is hsame. The ledger data are the source, intermediate, target, graph, modulus, certificate, visible prefix, and visible suffix histories displayed above. The stability field is visible-context left-right bracketing synchronization. The core theorem is this visible fourfold statement. The exactness field is the pair of reattached visible carriers together with the three hsame conclusions. The downstream consumer is ContinuousMapUp associativity coherence, the standard bridge is deferred, and the status is ProofTarget. □

Chapter 60

A Concrete Naming Certificate for $\mathsf{Compact}^{\uparrow}$

The compactness interface $\mathsf{Compact}^{\uparrow}$ has certificate obligations for subsets of metric spaces from chapter 57. The BEDC-licensed carrier is a subset predicate equipped with explicit locatedness witnesses and finite-net witnesses: locatedness gives, for any point and rational ϵ, a decision whether the point is within ϵ of the subset; finite-net data gives, for any rational ϵ, a finite ϵ-net.

60.1 Carrier definition

Definition 60.1 (Compact carrier)**.** *Use a metric-space subset predicate equipped with locatedness and total-boundedness witness functions.*

Lean4 (definition) `BEDC.Derived.CompactUp.CompactWitnessCarrier`

Theorem 60.2 (Compact witness carrier finite extension closure)**.** *If a compact witness carrier records unary subset, locatedness, and finite-net histories through locatedness and compactness ledgers, then appending a unary finite-net refinement preserves the same witness shape.*

Proof. The existing carrier supplies unary histories for the subset, locatedness, and finite-net ledgers together with continuations through the located and compact witnesses. Unary continuation closure gives the refined finite-net carrier. Continuation associativity combines the old compactness ledger with the refinement ledger, and determinism identifies the associated witness with the named refined finite-net history. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier finite extension closed`

Definition 60.3 (Compact finite refinement chain)**.** *The compact finite refinement chain relation is written*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact).$$

It is the inductive BEDC relation on paired finite-net and compactness ledgers generated by the following constructors:

1. *the base constructor relates* $(finite, compact)$ *to itself;*
2. *the step constructor says that, from a prior chain to* (f, c)*,* $\mathsf{UnaryHistory}(e)$*,* $\mathsf{Cont}(f, e, f')$*, and* $\mathsf{Cont}(c, e, c')$*, one obtains a chain to* (f', c')*.*

Lean4 (definition) `BEDC.Derived.CompactUp.CompactFiniteRefinementChain`

Theorem 60.4 (Compact finite refinement chain transitivity)**.** *For finite-net ledgers $finite$, $midFinite$, and $finalFinite$ and compactness ledgers $compact$, $midCompact$, and $finalCompact$, if*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, midFinite, midCompact)$$

and

$$\mathsf{CompactFiniteRefinementChain}(midFinite, midCompact, finalFinite, finalCompact),$$

then

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact).$$

Proof. Induct on the second compact finite refinement chain.

In the base case, the second chain ends at its starting pair $(midFinite, midCompact)$. The desired chain is exactly the first chain.

In the step case, the second chain is obtained from a prior pair (f, c) by a unary refinement history e together with witnesses $\mathsf{UnaryHistory}(e)$, $\mathsf{Cont}(f, e, f')$, and $\mathsf{Cont}(c, e, c')$. Apply the induction hypothesis to compose the first chain with the prior second-chain segment, producing a chain from $(finite, compact)$ to (f, c).

Apply the step constructor of Definition 60.3 to that composed chain and the displayed unary-history and continuation witnesses. This gives the required chain from $(finite, compact)$ to (f', c'), which is the final pair in the step case. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain transitivity`

Theorem 60.5 (Compact finite refinement chain prefix closure)**.** *For a prefix ledger p and finite-net and compactness ledgers $finite, compact, finalFinite, finalCompact$, if*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

then

$$\mathsf{CompactFiniteRefinementChain}(p \cdot finite, p \cdot compact, p \cdot finalFinite, p \cdot finalCompact).$$

Proof. Induct on the compact finite refinement chain. The base constructor is stable under prefixing because both endpoints remain identical pairs. In the step case, apply the induction hypothesis to the prior chain and rebuild the step constructor. The two continuation witnesses are obtained by applying the prefix operation to the original continuation equalities and reassociating append. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain prefix closed`

Theorem 60.6 (Compact finite refinement chain final endpoints unary)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

and the starting finite-net and compactness ledgers are unary histories, then the final finite-net and compactness ledgers are unary histories:

$$\mathsf{UnaryHistory}(finite) \wedge \mathsf{UnaryHistory}(compact) \Longrightarrow \mathsf{UnaryHistory}(finalFinite) \wedge \mathsf{UnaryHistory}(finalCompact).$$

Proof. Induct on the compact finite refinement chain. In the base case, the final endpoints are the starting endpoints, so the supplied unary-history witnesses are the desired witnesses. In the step case, the induction hypothesis gives unary-history witnesses for the prior finite-net and compactness ledgers. The step constructor supplies a unary refinement history and two continuation witnesses. Closure of unary histories under continuation gives unary histories for both final ledgers. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain final endpoints unary`

Theorem 60.7 (Compact finite refinement chain initial endpoints unary)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

and the final finite-net and compactness ledgers are unary histories, then the initial finite-net and compactness ledgers are unary histories:

$$\mathsf{UnaryHistory}(finalFinite) \wedge \mathsf{UnaryHistory}(finalCompact) \Longrightarrow \mathsf{UnaryHistory}(finite) \wedge \mathsf{UnaryHistory}(compact).$$

Proof. Induct on the compact finite refinement chain. The base case is immediate. In the step case, factor the final unary-history witnesses through the two continuation witnesses by unary left-factor closure, then apply the induction hypothesis to the prior chain. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain initial endpoints unary`

Theorem 60.8 (Compact finite refinement chain endpoint hsame)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

and $\mathsf{hsame}(finite, compact)$*, then* $\mathsf{hsame}(finalFinite, finalCompact)$*.*

Proof. Induct on the compact finite refinement chain. The base case is the supplied starting endpoint equality. In the step case, the induction hypothesis gives equality of the current finite and compact endpoints; the common unary refinement tail and the two continuation witnesses transport that equality to the next endpoints by continuation respect for hsame. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain endpoint hsame`

Theorem 60.9 (Compact finite refinement chain endpoint hsame reflects)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

and $\mathsf{hsame}(finalFinite, finalCompact)$*, then* $\mathsf{hsame}(finite, compact)$*.*

Proof. Induct on the compact finite refinement chain. The base case is the supplied final endpoint equality. In the step case, both endpoints were extended by the same refinement tail. Common-suffix cancellation for the two continuation witnesses reflects hsame from the final endpoints to the preceding endpoints, and the induction hypothesis reflects it to the starting endpoints. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain endpoint hsame reflects`

Theorem 60.10 (Compact finite refinement chain endpoint hsame iff)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

holds, then

$$\mathsf{hsame}(finalFinite, finalCompact) \Longleftrightarrow \mathsf{hsame}(finite, compact).$$

Proof. Use the forward endpoint-transport theorem for one implication and the endpoint reflection theorem for the other. The same finite refinement chain supplies the common continuation structure needed by both directions. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain endpoint hsame iff`

Theorem 60.11 (Compact finite refinement chain empty final reflects empty initial)**.** *If a compact finite refinement chain ends with both final ledgers empty, then both initial ledgers are empty:*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact) \wedge \mathsf{hsame}(finalFinite, \mathsf{e}) \wedge \mathsf{hsame}(final$$

Proof. Induct on the refinement chain. In a step, transport the final empty equality back across each displayed continuation, use empty-result inversion to identify the prior ledger with e, and apply the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain empty final reflects empty initial`

Theorem 60.12 (Compact finite refinement chain nonempty forward)**.** *If a compact finite refinement chain starts from a finite-net ledger apart from* e*, then its final finite-net ledger is apart from* e*:*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact) \wedge ((finite\mathsf{hsamee}) \to \bot) \Longrightarrow ((finalF$$

Proof. Induct on the refinement chain. The base case is the starting apartness predicate. In a step, a collapse of the final finite ledger to e identifies the appended prior finite ledger and refinement tail with e. Empty-append inversion gives a collapse of the prior finite ledger, contradicting the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain nonempty forward`

Theorem 60.13 (Compact finite refinement chain compact nonempty forward)**.** *If a compact finite refinement chain starts from a compact ledger apart from* e*, then its final compact ledger is apart from* e*:*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact) \wedge ((compact\mathsf{hsamee}) \to \bot) \Longrightarrow ((fina$$

Proof. Induct on the refinement chain. The base case is the starting compact apartness predicate. In a step, a collapse of the final compact ledger to e makes the appended prior compact ledger empty, and empty-append inversion gives the contradiction by the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain compact nonempty forward`

Theorem 60.14 (Compact finite refinement chain continuation witness)**.** *If*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

holds, then there is a unary refinement history extra that realizes both final endpoints as continuations from their corresponding starts:

$$\mathsf{UnaryHistory}(extra) \wedge \mathsf{Cont}(finite, extra, finalFinite) \wedge \mathsf{Cont}(compact, extra, finalCompact).$$

Proof. Induct on the finite refinement chain. The base case uses the empty history as the common continuation tail and the right-unit continuation law on both endpoints. In the step case, the induction hypothesis supplies a common accumulated tail. Associate that tail with the step refinement on the finite side and on the compact side. Continuation determinism identifies the two associated tails, unary continuation closure proves the accumulated tail unary, and the associated continuations give the two displayed endpoint ledgers. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain continuation witness`

Theorem 60.15 (Compact witness carrier finite refinement chain closure)**.** *If*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact)$$

and

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

then

$$\mathsf{CompactWitnessCarrier}(subset, located, finalFinite, intermediate, finalCompact).$$

Proof. Induct on the given compact finite refinement chain.

In the base case, the final finite-net ledger and final compactness ledger are the original ledgers $finite$ and $compact$. The desired carrier is exactly the given compact witness carrier.

In the step case, the induction hypothesis gives the carrier at the prior pair (f, c). Apply Theorem 60.2 to that carrier, the unary refinement history e, and the continuations $\mathsf{Cont}(f, e, f')$ and $\mathsf{Cont}(c, e, c')$. The result is the required compact witness carrier at (f', c'). □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier finite refinement chain closed`

Theorem 60.16 (Compact witness carrier empty final refinement reflects empty initial)**.** *For a compact witness carrier, if a finite refinement chain sends the finite and compact ledgers to empty final ledgers, then all five carrier ledgers are empty:*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact) \wedge \mathsf{CompactFiniteRefinementChain}(finite, c$$
$$\wedge\ \mathsf{hsame}(finalFinite, \mathsf{e}) \wedge \mathsf{hsame}(finalCompact, \mathsf{e}) \Longrightarrow \mathsf{hsame}(subset, \mathsf{e}) \wedge \mathsf{hsame}(located, \mathsf{e}) \wedge \mathsf{hsame}(finite, \mathsf{e}) \wedge$$

Proof. First apply the finite-chain empty reflection theorem to the refinement chain. Then use empty-result inversion on the compactness continuation $\mathsf{Cont}(intermediate, finite, compact)$ and on the locatedness continuation $\mathsf{Cont}(subset, located, intermediate)$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier empty final refinement reflects empty initial`

Theorem 60.17 (Compact witness carrier finite refinement chain endpoints unary)**.** *If*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact)$$

and

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

then the final finite-net and compactness ledgers are unary histories:

$$\mathsf{UnaryHistory}(finalFinite) \wedge \mathsf{UnaryHistory}(finalCompact).$$

Proof. The compact witness carrier supplies a unary finite-net ledger. Its subset and located unary witnesses, together with the first continuation ledger, give a unary intermediate ledger; combining that with the finite-net unary witness and the second continuation ledger gives a unary compactness ledger. Applying the finite refinement chain endpoint theorem to these two starting endpoints gives the displayed final endpoint unary witnesses. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier finite refinement chain endpoints unary`

Theorem 60.18 (Compact witness carrier finite refinement endpoint package)**.** *If*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact)$$

and

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

hold, then both final endpoints are unary and the endpoint classifier relation is reflected exactly:

$$\begin{aligned}&\mathsf{UnaryHistory}(finalFinite) \wedge \mathsf{UnaryHistory}(finalCompact)\\ &\wedge\ (\mathsf{hsame}(finalFinite, finalCompact) \iff \mathsf{hsame}(finite, compact)).\end{aligned}$$

Proof. Apply Theorem 60.17 to the compact witness carrier and the finite refinement chain. This gives the unary witnesses for the final finite-net and compactness endpoints.

Apply Theorem 60.10 to the same finite refinement chain. It gives the displayed classifier equivalence, and conjoining it with the endpoint unary witnesses gives the package. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier finite refinement endpoint package`

Theorem 60.19 (Compact witness carrier iterated finite refinement closure)**.** *For histories subset, located, $finite$, $extra_0$, $extra_1$, intermediate, compact, $finite_0$, $compact_0$, $finite_1$, and $compact_1$, assume:*

1. $$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact);$$
2. $\mathsf{UnaryHistory}(extra_0)$ *and* $\mathsf{UnaryHistory}(extra_1)$*;*
3. $\mathsf{Cont}(finite, extra_0, finite_0)$ *and* $\mathsf{Cont}(compact, extra_0, compact_0)$*;*
4. $\mathsf{Cont}(finite_0, extra_1, finite_1)$ *and* $\mathsf{Cont}(compact_0, extra_1, compact_1)$*.*

Then

$$\mathsf{CompactWitnessCarrier}(subset, located, finite_1, intermediate, compact_1)$$

Proof. First apply Theorem 60.2 to the carrier with finite-net ledger $finite$ and compactness ledger $compact$, using $extra_0$ as the unary refinement history. This produces the intermediate carrier.

$$\mathsf{CompactWitnessCarrier}(subset, located, finite_0, intermediate, compact_0)$$

Apply the same theorem again to that intermediate carrier, now using $extra_1$ as the unary refinement history and the continuations from $finite_0$ to $finite_1$ and from $compact_0$ to $compact_1$. The resulting carrier is:

$$\mathsf{CompactWitnessCarrier}(subset, located, finite_1, intermediate, compact_1)$$

□

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier iterated finite refinement closed`

Theorem 60.20 (Compact witness carrier located extension closure)**.** *If a compact witness carrier records a subset through locatedness and finite-net histories, then appending a unary locatedness refinement preserves the compact witness shape after reassociating the subset ledger.*

Proof. The carrier supplies unary subset, locatedness, and finite-net histories, together with a continuation from the subset through the locatedness witness to the intermediate witness. Unary continuation closure makes the refined locatedness history unary. Continuation associativity shifts the subset ledger across the locatedness refinement, and determinism identifies the associated locatedness witness with the displayed refinement. The final finite-net ledger then gives the compact witness endpoint. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located extension closed`

Definition 60.21 (Compact located refinement chain)**.** *The compact located refinement chain relation is written*

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final$$

It is the inductive BEDC relation on locatedness, intermediate, and compactness ledgers with the finite-net ledger held fixed, generated by the following constructors:

1. *the base constructor relates* $(located, intermediate, compact)$ *to itself while keeping* $finite$ *fixed;*
2. *the step constructor says that, from a prior chain to* (l, i, c)*,* $\mathsf{UnaryHistory}(e)$*,* $\mathsf{Cont}(l, e, l')$*,* $\mathsf{Cont}(i, e, i')$*, and* $\mathsf{Cont}(i', finite, c')$*, one obtains a chain to* (l', i', c')*.*

Lean4 (definition) `BEDC.Derived.CompactUp.CompactLocatedRefinementChain`

Theorem 60.22 (Compact located refinement chain transitivity)**.** *For a fixed finite-net ledger* $finite$*, locatedness ledgers* $located$*,* $midLocated$*, and* $finalLocated$*, intermediate ledgers* $intermediate$*,* $midIntermediate$*, and* $finalIntermediate$*, and compactness ledgers* $compact$*,* $midCompact$*, and* $finalCompact$*, if*

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, midLocated, midIntermediate, midCom$$

and

$$\mathsf{CompactLocatedRefinementChain}(finite, midLocated, midIntermediate, midCompact, finalLocated, finalInterm$$

then

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final$$

Proof. Induct on the second compact located refinement chain.

In the base case, the second chain ends at its starting triple $(midLocated, midIntermediate, midCompact)$. The desired chain is exactly the first chain.

In the step case, the second chain is obtained from a prior triple (l, i, c) by a unary refinement history e together with witnesses $\mathsf{UnaryHistory}(e)$, $\mathsf{Cont}(l, e, l')$, $\mathsf{Cont}(i, e, i')$, and $\mathsf{Cont}(i', finite, c')$. Apply the induction hypothesis to compose the first chain with the prior second-chain segment, producing a chain from $(located, intermediate, compact)$ to (l, i, c).

Apply the step constructor of Definition 60.21 to that composed chain and the displayed unary-history and continuation witnesses. This gives the required chain from $(located, intermediate, compact)$ to (l', i', c'), which is the final triple in the step case. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain transitivity`

Theorem 60.23 (Compact located refinement chain prefix closure)**.** *For a prefix ledger p, if*

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, finalC$$

then

$$\mathsf{CompactLocatedRefinementChain}(finite, p{\cdot}located, p{\cdot}intermediate, p{\cdot}compact, p{\cdot}finalLocated, p{\cdot}finalIntermedia$$

Proof. Induct on the compact located refinement chain. The base constructor is stable under prefixing because the locatedness, intermediate, and compactness endpoints remain the same triples after prefixing. In the step case, apply the induction hypothesis to the prior chain and rebuild the step constructor. The located, intermediate, and compact continuation witnesses are obtained by applying the prefix operation to the original continuation equalities and reassociating append. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain prefix closed`

Theorem 60.24 (Compact located refinement chain final endpoints unary)**.** *If*

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, finalC$$

and the fixed finite-net ledger plus the starting locatedness, intermediate, and compactness ledgers are unary histories, then the final locatedness, intermediate, and compactness ledgers are unary histories:

$$\mathsf{UnaryHistory}(finite) \wedge \mathsf{UnaryHistory}(located) \wedge \mathsf{UnaryHistory}(intermediate) \wedge \mathsf{UnaryHistory}(compact) \Longrightarrow \mathsf{UnaryHistory}$$

Proof. Induct on the compact located refinement chain. In the base case, the final endpoints are the starting locatedness, intermediate, and compactness ledgers. In the step case, the induction hypothesis gives unary-history witnesses for the prior locatedness and intermediate ledgers. The step constructor supplies a unary refinement history and continuation witnesses for the next locatedness and intermediate ledgers, so continuation closure gives unary histories for those two ledgers. The fixed finite-net unary-history witness and the final continuation witness then give the final compactness unary-history witness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain final endpoints unary`

Theorem 60.25 (Compact located refinement chain initial endpoints unary)**.** *If*

$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final($

and the final locatedness and intermediate ledgers are unary histories, then the initial locatedness and intermediate ledgers are unary histories:

$\mathsf{UnaryHistory}(finalLocated) \wedge \mathsf{UnaryHistory}(finalIntermediate) \Longrightarrow \mathsf{UnaryHistory}(located) \wedge \mathsf{UnaryHistory}(intermed$

Proof. Induct on the compact located refinement chain. The base case is immediate, because the final locatedness and intermediate ledgers are the starting ledgers. In the step case, factor the final unary-history witnesses through the locatedness and intermediate continuation witnesses by unary left-factor closure, then apply the induction hypothesis to the prior chain. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain initial endpoints unary`
↪ variant `BEDC.Derived.CompactUp.CompactLocatedRefinementChain initial located intermediate unary`

Theorem 60.26 (Compact located refinement chain endpoint hsame)**.** *If*

$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final($

and $\mathsf{hsame}(located, intermediate)$*, then* $\mathsf{hsame}(finalLocated, finalIntermediate)$*.*

Proof. Induct on the compact located refinement chain. The base case is the supplied starting equality between locatedness and the intermediate ledger. In the step case, the induction hypothesis gives equality of the current located and intermediate endpoints; the shared refinement tail and the two displayed continuations transport it to the next located and intermediate endpoints. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain endpoint hsame`

Theorem 60.27 (Compact located refinement chain endpoint hsame reflects)**.** *If*

$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final($

and $\mathsf{hsame}(finalLocated, finalIntermediate)$*, then* $\mathsf{hsame}(located, intermediate)$*.*

Proof. Induct on the compact located refinement chain. The base case is the supplied final equality. In the step case, located and intermediate endpoints are extended by the same refinement tail, so common-suffix cancellation reflects the final hsame witness to the preceding located and intermediate endpoints. The induction hypothesis then reflects that witness to the starting pair. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain endpoint hsame reflects`

Theorem 60.28 (Compact located refinement chain endpoint unary hsame iff)**.** *If a compact located refinement chain carries unary starting histories* $finite, located, intermediate, compact$*, then its final located, intermediate, and compact endpoints are unary, and* $\mathsf{hsame}(finalLocated, finalIntermediate)$ *holds exactly when* $\mathsf{hsame}(located, intermediate)$ *holds.*

Proof. Apply Definition 60.24 for the unary endpoints, and combine Definition 60.26 with Definition 60.27. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain endpoint unary hsame iff`

Theorem 60.29 (Compact located refinement chain continuation witness)**.** *If*

$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final($

holds and $\mathsf{Cont}(intermediate, finite, compact)$ *is the starting compactness ledger, then there is a unary refinement history extra such that*

$\mathsf{UnaryHistory}(extra) \wedge \mathsf{Cont}(located, extra, finalLocated) \wedge \mathsf{Cont}(intermediate, extra, finalIntermediate) \wedge \mathsf{Cont}(fi$

Proof. Induct on the located refinement chain. The base case uses the empty history as the common locatedness and intermediate tail, together with the supplied starting compactness ledger. In the step case, the induction hypothesis supplies an accumulated tail for locatedness and the intermediate ledger. Associate that tail with the step refinement on both ledgers. Determinism identifies the two associated tails, unary continuation closure proves the accumulated tail unary, and the step compactness continuation gives the final compact endpoint ledger. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain continuation witness`

Theorem 60.30 (Compact located refinement chain final compact determinacy)**.** *If two compact located refinement chains share the same starting compactness ledger and the same final located and intermediate endpoints, then their final compact endpoints are* hsame*-equal.*

Proof. Apply the continuation witness theorem to both chains using the same starting compactness ledger. Both witnesses end in $\mathsf{Cont}(finalIntermediate, finite, -)$, so continuation determinism identifies the final compact endpoints. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain final compact deterministic`

Theorem 60.31 (Compact located refinement chain common-source final located determinacy)**.** *If two compact located refinement chains share the same starting compactness ledger and their final intermediate ledgers are* hsame*, then the first chain has a common located refinement witness and the two final located histories are* hsame*.*

Proof. Apply the located continuation-witness theorem to both chains using the same starting compactness ledger. The shared intermediate source and the equality of final intermediate ledgers cancel the refinement tails. Continuation determinism on the located ledgers then identifies the two final located histories, while the first witness supplies the displayed existential data. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain common source final located deterministic`

Theorem 60.32 (Compact located refinement chain common-source final located tail determinacy)**.** *If two compact located refinement chains share the same starting compactness ledger and their final located histories are* hsame*, then their final intermediate histories and final compact histories are* hsame*.*

Proof. Apply the located continuation-witness theorem to both chains using the same starting compactness ledger. The final located equality cancels the common located source and identifies the two refinement tails. Continuation determinism then identifies the final intermediate histories, and the resulting intermediate equality identifies the final compact histories through the fixed finite suffix. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain common source final located tail deterministic`

Theorem 60.33 (Compact located refinement chain common-source final intermediate tail determinacy)**.** *If two compact located refinement chains share the same starting compactness ledger and their final intermediate histories are* hsame*, then their final located histories and final compact histories are* hsame*.*

Proof. Apply the located continuation-witness theorem to both chains using the same starting compactness ledger. The final intermediate equality cancels the common intermediate source and identifies the refinement tails. Continuation determinism on the located ledgers identifies the final located histories, and continuation determinism over the fixed finite suffix identifies the final compact histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain common source final intermediate tail deterministic`

Theorem 60.34 (Compact located refinement chain common-source final compact tail determinacy)**.** *If two compact located refinement chains share the same starting compactness ledger and their final compact histories are* hsame*, then their final intermediate histories and final located histories are* hsame*.*

Proof. Apply the located continuation-witness theorem to both chains using the same starting compactness ledger. The final compact equality cancels the fixed finite suffix and identifies the final intermediate histories. That equality cancels the common intermediate source to identify the refinement tails, and continuation determinism on the located ledgers identifies the final located histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain common source final compact tail deterministic`

Theorem 60.35 (Compact located refinement chain empty final reflects empty initial)**.** *If a compact located refinement chain has empty final locatedness and intermediate ledgers, then its initial locatedness and intermediate ledgers are empty.*

Proof. Induct on the located refinement chain. In each step, transport the two final empty equalities across the displayed locatedness and intermediate continuations, use empty-result inversion, and apply the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain empty final reflects empty initial`

Theorem 60.36 (Compact located refinement chain located nonempty forward)**.** *If a compact located refinement chain starts from a locatedness ledger that is not history-same to* e*, then its final locatedness ledger is not history-same to* e*.*

Proof. Induct on the located refinement chain. The base case is the initial nonemptiness premise. In a step, an empty final locatedness endpoint transports across the displayed locatedness continuation; empty-result inversion then forces the prior locatedness endpoint to be empty, contradicting the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain located nonempty forward`

Theorem 60.37 (Compact located refinement chain intermediate nonempty forward)**.** *If a compact located refinement chain starts from an intermediate ledger that is not history-same to* e*, then its final intermediate ledger is not history-same to* e*.*

Proof. Induct on the located refinement chain. The base case is the initial nonemptiness premise. In a step, an empty final intermediate endpoint transports across the displayed intermediate continuation; empty-result inversion then forces the prior intermediate endpoint to be empty, contradicting the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain intermediate nonempty forward`

Theorem 60.38 (Compact witness carrier located empty final refinement reflects empty initial)**.** *For a compact witness carrier and a located refinement chain, if the final locatedness, intermediate, and compactness ledgers are empty, then all five initial carrier ledgers are empty.*

Proof. Use the located continuation witness to expose a common refinement tail. Empty-result inversion gives empty locatedness, intermediate, and finite ledgers; the carrier continuations then reflect emptiness to the subset and compactness ledgers. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located empty final refinement reflects empty initial`

Theorem 60.39 (Compact witness carrier located finite empty final reflects empty initial)**.** *For a compact witness carrier, a located refinement chain, and a finite refinement chain starting from the located chain's final compact ledger, if the final locatedness, final intermediate, final finite, and final compact ledgers are all empty, then all five initial carrier ledgers are empty.*

Proof. First reflect the empty final finite and compact ledgers through the finite refinement chain to recover an empty final compact ledger for the located chain. Then apply the located empty-final reflection theorem to the compact witness carrier and located refinement chain. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located finite empty final reflects empty initial`

Theorem 60.40 (Compact witness carrier located refinement chain closure)**.** *If* $\mathsf{CompactWitnessCarrier}(subset, loca$ *and* $\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, fi$ *then* $\mathsf{CompactWitnessCarrier}(subset, finalLocated, finite, finalIntermediate, finalCompact)$.

Proof. Induct on the compact located refinement chain. The base case is the supplied carrier. In the step case, apply Definition 60.20 to the carrier supplied by the induction hypothesis and the displayed locatedness, intermediate, and compactness continuations. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located refinement chain closed`

Theorem 60.41 (Compact witness carrier located finite refinement chain closure)**.** *If*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

and

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, finalC$$

and

$$\mathsf{CompactFiniteRefinementChain}(finite, finalCompact, finalFinite, finalCompact'),$$

then

$$\mathsf{CompactWitnessCarrier}(subset, finalLocated, finalFinite, finalIntermediate, finalCompact').$$

Proof. First apply Theorem 60.40 to the initial carrier and the located refinement chain. This gives

$$\mathsf{CompactWitnessCarrier}(subset, finalLocated, finite, finalIntermediate, finalCompact).$$

Then apply Theorem 60.15 to that carrier and the finite refinement chain from $(finite, finalCompact)$ to $(finalFinite, finalCompact')$. The result is the displayed compact witness carrier. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located finite refinement chain closed`

Theorem 60.42 (Compact witness carrier located finite refinement chain final endpoints unary)**.** *Assume*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

a located refinement chain from $(located, intermediate, compact)$ *to* $(finalLocated, finalIntermediate, finalComp$
over the same finite witness, and a finite refinement chain from $(finite, finalCompact)$ *to* $(finalFinite, finalCom$
Then

$$\mathsf{UnaryHistory}(finalLocated) \wedge \mathsf{UnaryHistory}(finalIntermediate) \wedge \mathsf{UnaryHistory}(finalFinite) \wedge \mathsf{UnaryHistory}(finalC$$

Proof. Unpack the compact witness carrier. The carrier provides unary witnesses for *subset*, *located*, and *finite* together with the continuation ledgers $\mathsf{Cont}(subset, located, intermediate)$ and $\mathsf{Cont}(intermediate, finite, compact)$. Closure of unary histories under Cont gives unary witnesses for *intermediate* and *compact*.

Apply the located-chain endpoint theorem to the located chain and these four initial endpoint witnesses. This gives unary witnesses for $finalLocated$, $finalIntermediate$, and $finalCompact$. Apply the finite-chain endpoint theorem to the finite chain, the original unary witness for $finite$, and the obtained unary witness for $finalCompact$. The result supplies unary witnesses for $finalFinite$ and $finalCompact'$, which together form the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located finite refinement chain final endpoints unary`

Theorem 60.43 (Compact witness carrier located finite refinement certificate package)**.** *Assume*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

a compact located refinement chain from $(finite, located, intermediate, compact)$ *to* $(finalLocated, finalIntermed$
and a compact finite refinement chain from $(finite, finalCompact)$ *to* $(finalFinite, finalCompact')$.
Then these data determine one refinement certificate with the following components:

1. *the refined carrier*

$$\mathsf{CompactWitnessCarrier}(subset, finalLocated, finalFinite, finalIntermediate, finalCompact');$$

2. *unary histories for* $finalLocated$, $finalIntermediate$, $finalFinite$, *and* $finalCompact'$;

3. *endpoint classifier equivalences*

$$finalLocated\mathsf{hsame}finalIntermediate \leftrightarrow located\mathsf{hsame}intermediate$$

and

$$finalFinite\mathsf{hsame}finalCompact' \leftrightarrow finite\mathsf{hsame}finalCompact;$$

4. *terminal empty reflection: if $finalLocated$, $finalIntermediate$, $finalFinite$, and $finalCompact'$ are each history-same to* e*, then subset, located, $finite$, intermediate, and compact are each history-same to* e*;*

5. *nonempty propagation from located to $finalLocated$, from intermediate to $finalIntermediate$, from $finite$ to $finalFinite$, and from $finalCompact$ to $finalCompact'$.*

Proof. First apply Theorem 60.41 to the initial carrier and the two refinement chains. This supplies the refined compact witness carrier.

Next apply Theorem 60.42 to the same inputs. The four endpoint unary witnesses are exactly the unary part of the certificate. The located endpoint classifier equivalence comes from the compact located refinement chain endpoint theorem, and the finite endpoint classifier equivalence comes from the compact finite refinement chain endpoint theorem.

The terminal empty-reflection component is Theorem 60.39. It takes the four empty terminal endpoints and reflects them back to the five initial carrier ledgers.

Finally, the nonempty propagation fields are supplied independently by Theorem 60.36, Theorem 60.37, the checked compact finite refinement chain nonempty-forward theorem, and the checked compact finite refinement chain compact-nonempty-forward theorem. Together these components form the advertised refinement certificate package. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located finite refinement complete package`

Corollary 60.44 (Compact located refinement chain continuation determinacy)**.** *Fix the compact located refinement-chain rows*

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, final($$

and the starting compactness ledger $\mathsf{Cont}(intermediate, finite, compact)$*. Any consumer-facing continuation use accepted by this chapter factors through the following displayed rows and through no further coordinate: the located refinement tail from located to $finalLocated$, the intermediate refinement tail from intermediate to $finalIntermediate$, and the finite compactness suffix* $\mathsf{Cont}(finalIntermediate, finite, finalCompact)$*. Consequently, if a second accepted use has the same starting compactness ledger and the same named final located and intermediate ledgers, then it names the same finite route, the same normal-form endpoint, and the same exactness decision up to* hsame*; in particular, it cannot choose an alternate hidden coordinate or a duplicate classifier source.*

Proof. Apply Definition 60.29 to the accepted chain and the fixed starting compactness ledger. The witness exposes a unary refinement history *extra* together with exactly three continuation rows:

$$\mathsf{Cont}(located, extra, finalLocated), \quad \mathsf{Cont}(intermediate, extra, finalIntermediate), \quad \mathsf{Cont}(finalIntermediate,$$

These are the only route, transport, and readback rows made available by the continuation witness, so a consumer-facing use that is admitted by the compact located refinement-chain surface must

read its finite route from *extra*, its normal form from the two named endpoints, and its compact exactness decision from the displayed finite suffix.

Now compare with any second accepted use over the same starting compactness ledger and the same final located and intermediate ledgers. By Definition 60.31, the common source and final intermediate row identify the located endpoints and supply a common located refinement witness. With the located and intermediate rows fixed, Definition 60.30 identifies the final compact endpoints. Thus the two accepted uses have the same exposed refinement tail and the same compact suffix up to hsame. Any purported extra coordinate would have to change either the located endpoint, the intermediate endpoint, or the final compact endpoint; the cited determinacy rows identify all three, so no unlisted coordinate or second classifier source survives the comparison. □

Theorem 60.45 (Compact finite refinement common-source final finite determinacy)**.** *If two compact finite refinement chains start from the same finite and compact histories, and their final finite histories are* hsame*, then the first chain has a common continuation witness and the two final compact histories are* hsame*:*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact) \Longrightarrow \mathsf{CompactFiniteRefinementChain}($$

Proof. Apply the compact finite refinement continuation-witness theorem to both chains. The shared finite source and the equality of final finite ledgers cancel the two continuation tails. Transporting that tail equality through the compact continuation ledgers identifies the two final compact histories, while the first witness supplies the displayed existential data. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain common source final finite deterministic`

Theorem 60.46 (Compact finite refinement common-source final compact determinacy)**.** *If two compact finite refinement chains start from the same finite and compact histories, and their final compact histories are* hsame*, then the first chain has a common continuation witness and the two final finite histories are* hsame*:*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact) \Longrightarrow \mathsf{CompactFiniteRefinementChain}($$

Proof. Apply the compact finite refinement continuation-witness theorem to both chains. The shared compact source and the equality of final compact ledgers cancel the two continuation tails. Continuation determinism then identifies the two final finite histories, and the first witness supplies the displayed existential data. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain common source final compact deterministic`

Theorem 60.47 (Prefixed composite compact net refinement determinacy)**.** *For histories* $p, c, \epsilon, \eta, \eta', \theta, \theta', m$*, assume* ϵ*,* η*, and* η' *are unary histories, with continuation ledgers*

$$\mathsf{Cont}(\epsilon, \eta, \theta), \qquad \mathsf{Cont}(\epsilon, \eta', \theta').$$

If both composed precisions give compact net witnesses with the same prefixed center and the same prefixed refined net,

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c, \theta', p \cdot m),$$

then the refinement precisions are history-same:

$$\mathsf{hsame}(\eta, \eta').$$

Proof. Unpack the two prefixed compact net witnesses. Their continuation ledgers have the same source $p \cdot c$ and the same result $p \cdot m$, so continuation left cancellation identifies the composed precisions θ and θ'. With the composed precision made common, the two ledgers out of ϵ have the same source and result, and another left-cancellation step yields $\mathsf{hsame}(\eta, \eta')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite refinement deterministic`

Theorem 60.48 (Compact finite refinement flattening)**.** *Let subset, located, finite, intermediate, compact, midFinite, midCompact, finalFinite, and finalCompact be histories. Assume a compact witness carrier*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

an outer finite refinement chain

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, midFinite, midCompact),$$

and a second finite refinement chain over the terminal finite-refinement pair

$$\mathsf{CompactFiniteRefinementChain}(midFinite, midCompact, finalFinite, finalCompact).$$

Then the two finite-refinement layers flatten to a single finite refinement chain with the same source:

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact).$$

Moreover the flattened endpoint keeps the original subset, locatedness, and intermediate classifier data:

$$\mathsf{CompactWitnessCarrier}(subset, located, finalFinite, intermediate, finalCompact).$$

Proof. Apply Theorem Definition 60.4 to the two displayed finite refinement chains. This gives the single chain from $(finite, compact)$ to $(finalFinite, finalCompact)$, so the source pair is the original source pair from the carrier.

Apply Theorem Definition 60.15 to the original compact witness carrier and this flattened chain. The closure theorem changes only the finite-net and compactness ledgers. Its conclusion has exactly the same *subset*, *located*, and *intermediate* arguments, so every locatedness classification witnessed by the original carrier is the same classification at the flattened endpoint. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain flattening located cont`

Corollary 60.49 (Prefixed composite compact net refinement readback determinacy)**.** *Fix histories $p, c, \epsilon, \eta, \eta', \theta, \theta', m$ and compact-refinement histories subset, located, finite, intermediate, compact, midFinite, midCompact, finalFinite, and finalCompact. Assume ϵ, η, and η' are unary histories and that the two prefixed composite compact-net uses expose exactly the continuation and witness rows*

$$\mathsf{Cont}(\epsilon, \eta, \theta), \qquad \mathsf{Cont}(\epsilon, \eta', \theta'),$$

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m), \qquad \mathsf{CompactNetWitness}(p \cdot c, \theta', p \cdot m).$$

Assume also that the finite compact-refinement packet exposes the carrier and two finite-refinement rows

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

$\mathsf{CompactFiniteRefinementChain}(finite, compact, midFinite, midCompact)$, $\mathsf{CompactFiniteRefinementChain}(mi$

Then every consumer-facing use admitted by these rows has the same displayed readback data: the prefixed composite refinement route is η up to hsame*, the finite route is the flattened chain*

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact),$$

the normal-form endpoint is the pair $(finalFinite, finalCompact)$, and the exactness decision is the carrier

$$\mathsf{CompactWitnessCarrier}(subset, located, finalFinite, intermediate, finalCompact).$$

Thus no accepted use can replace the refinement route by an alternate hidden coordinate or cite a duplicate classifier source for the finite endpoint.

Proof. The two prefixed composite compact-net uses have the same visible prefix p, center c, and final net m. Their only difference is the refinement continuation row out of the fixed first precision ϵ. Applying Definition 60.47 to the four displayed prefixed composite rows gives

$$\mathsf{hsame}(\eta, \eta').$$

Therefore any admitted composite readback that uses the second refinement row names the same refinement route as the first one, up to the chapter classifier hsame.

For the finite compact-refinement part, apply Definition 60.48 to the displayed carrier and the two displayed finite-refinement chains. The result is exactly the single finite route from $(finite, compact)$ to $(finalFinite, finalCompact)$, together with the carrier at the same final endpoint:

$$\mathsf{CompactFiniteRefinementChain}(finite, compact, finalFinite, finalCompact)$$

and

$$\mathsf{CompactWitnessCarrier}(subset, located, finalFinite, intermediate, finalCompact).$$

The flattening theorem changes only the finite-net and compactness ledgers and keeps the same *subset*, *located*, and *intermediate* classifier data. Hence the accepted use has no further coordinate from which to choose another route, normal form, or exactness source: the route is the flattened finite chain, the normal form is its displayed endpoint pair, and the exactness decision is the flattened carrier over that endpoint. Combining this with the prefixed composite determinacy proves the stated readback determinacy. □

Theorem 60.50 (Compact finite refinement chain final cases)**.** *Every compact finite refinement chain reaches its final endpoints in exactly one of the constructor-visible ways: either the chain is at the starting pair, with $finalFinite$hsame$finite$ and $finalCompact$hsame$compact$, or it has an immediate predecessor pair and a unary refinement history $extra$ with* $\mathsf{Cont}(currentFinite, extra, finalFinite)$ *and* $\mathsf{Cont}(currentCompact, extra, finalCompact)$*.*

Proof. Case-split on the chain constructor. The base constructor gives the two endpoint sameness proofs. The step constructor exposes the predecessor chain, the unary refinement history, and the two displayed continuation ledgers. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactFiniteRefinementChain final cases`

Theorem 60.51 (Compact located refinement chain final cases)**.** *Every compact located refinement chain reaches its final endpoints either at the starting triple, with all three final endpoints history-same to the starting ledgers, or by a final refinement step. In the step case there are current located, intermediate, and compact endpoints and a unary history extra such that the predecessor chain reaches the current endpoints,* $\mathsf{Cont}(currentLocated, extra, finalLocated)$*,* $\mathsf{Cont}(currentIntermediate, extra, fina$ *and* $\mathsf{Cont}(finalIntermediate, finite, finalCompact)$ *hold.*

Proof. Case-split on the located refinement-chain constructor. The base constructor returns the three endpoint sameness proofs. The step constructor supplies the predecessor chain, the unary refinement history, the located and intermediate continuation ledgers, and the final compactness continuation ledger. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactLocatedRefinementChain final cases`

Theorem 60.52 (Compact witness carrier located finite refinement endpoint package)**.** *Assume*

$$\mathsf{CompactWitnessCarrier}(subset, located, finite, intermediate, compact),$$

and

$$\mathsf{CompactLocatedRefinementChain}(finite, located, intermediate, compact, finalLocated, finalIntermediate, finalC$$

and

$$\mathsf{CompactFiniteRefinementChain}(finite, finalCompact, finalFinite, finalCompact').$$

Then one obtains

$$\mathsf{CompactWitnessCarrier}(subset, finalLocated, finalFinite, finalIntermediate, finalCompact'),$$

unary histories for $finalLocated$*,* $finalIntermediate$*,* $finalFinite$*, and* $finalCompact'$*, and the endpoint classifier equivalences*

$$\mathsf{hsame}(finalLocated, finalIntermediate) \iff \mathsf{hsame}(located, intermediate)$$

and

$$\mathsf{hsame}(finalFinite, finalCompact') \iff \mathsf{hsame}(finite, finalCompact).$$

Proof. First package the refined carrier. Apply Theorem 60.41 to the initial compact witness carrier, the located refinement chain, and the finite refinement chain. This gives the displayed carrier over $(subset, finalLocated, finalFinite, finalIntermediate, finalCompact')$.

Next package the endpoint unary data. Apply Theorem 60.42 to the same three inputs, yielding unary histories for $finalLocated$, $finalIntermediate$, $finalFinite$, and $finalCompact'$.

For the located endpoint classifier, apply Theorem 60.26 for the forward direction and Theorem 60.27 for the reverse direction.

For the finite endpoint classifier, apply Theorem 60.10 to the finite refinement chain. The refined carrier, unary endpoint witnesses, and the two displayed equivalences form the requested package. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactWitnessCarrier located finite refinement endpoint package`

60.2 The certificate

Definition 60.53 (Compact source specification)**.** *The source is a certified metric space plus a subset predicate over its carrier.*

Definition 60.54 (Compact pattern specification)**.** *The pattern is compact-subset formation by attaching locatedness and finite-net data for each rational precision.*

Definition 60.55 (Compact classifier specification)**.** *The classifier relates compact witnesses when their membership predicates agree by explicit bidirectional evidence under carrier sameness.*

Definition 60.56 (Compact stability certificate)**.** *The stability certificate consists of:*

1. *membership stability under carrier classifier*
2. *locatedness witness validity*
3. *total-bounded finite-net validity*
4. *monotonicity of precision witnesses where required*
5. *stability under compact-witness classifier*

Theorem 60.57 (Compact net witness semantic name certificate)**.** *For unary histories c and ϵ, the compact-net witness predicate* $\mathsf{CompactNetWitness}(c, \epsilon, -)$ *carries a semantic name certificate with source, pattern, and ledger all equal to that concrete witness predicate, and classifier* hsame*:*

$$\mathsf{SemanticNameCert}(\mathsf{CompactNetWitness}(c, \epsilon, -), \mathsf{CompactNetWitness}(c, \epsilon, -), \mathsf{CompactNetWitness}(c, \epsilon, -), \mathsf{hsame}).$$

Proof. The carrier witness is the concrete history $c \cdot \epsilon$. Unary append closure supplies its unary carrier, and the continuation ledger is $c \cdot \epsilon = c \cdot \epsilon$. The classifier fields are the reflexive, symmetric, and transitive laws of hsame, with carrier transport by equality case analysis. Pattern and ledger soundness are the identity projections because all three semantic predicates are the compact-net witness predicate. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness semanticNameCert`

Theorem 60.58 (Compact net witness prefix closure)**.** *For unary histories p, c, ϵ, n, if n is a compact net witness obtained by continuing center c with precision ϵ, then prefixing the center and net by p preserves the same precision witness:*

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \Longrightarrow \mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n).$$

Proof. Unpack the witness into unary center, precision, and net histories together with the continuation ledger $n = c \cdot \epsilon$. Unary closure gives unary histories for $p \cdot c$ and $p \cdot n$. The prefixed continuation ledger is the associativity equality $(p \cdot c) \cdot \epsilon = p \cdot (c \cdot \epsilon)$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefix closed`

Theorem 60.59 (Compact net witness prefix exactness)**.** *For histories p, c, ϵ, n, a prefixed compact-net witness is equivalent to unary prefix evidence together with the unprefixed witness:*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{CompactNetWitness}(c, \epsilon, n).$$

Proof. For the forward direction, unpack the prefixed witness. Unary append factorization recovers the unary prefix and the unary center and net histories, while prefix cancellation on the continuation ledger recovers $\mathsf{Cont}(c, \epsilon, n)$. For the reverse direction, apply prefix closure to the unary prefix and the original compact-net witness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefix iff`

Theorem 60.60 (Compact net witness prefixed result determinacy)**.** *If two compact-net witnesses share the same prefixed center and precision,*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n'),$$

then the unprefixed net results are history-same:

$$\mathsf{hsame}(n, n').$$

Proof. The two compact-net witnesses contain continuation ledgers with the same source $p \cdot c$ and precision ϵ. Continuation determinacy identifies the two prefixed results $p \cdot n$ and $p \cdot n'$, and left cancellation removes the common prefix p. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed result deterministic`

Theorem 60.61 (Compact net witness empty net exactness)**.** *For histories c and ϵ, a compact-net witness with empty net is equivalent to both the center and precision histories being empty:*

$$\mathsf{CompactNetWitness}(c, \epsilon, \mathsf{e}) \iff \mathsf{hsame}(c, \mathsf{e}) \wedge \mathsf{hsame}(\epsilon, \mathsf{e}).$$

Proof. Unpack the compact-net witness and invert the continuation $\mathsf{Cont}(c, \epsilon, \mathsf{e})$. This yields empty center and empty precision. The reverse implication uses the empty unary carrier for both endpoints and the canonical continuation from e through e to e. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness empty net iff`

Theorem 60.62 (Compact net witness endpoint cycle precision empty)**.** *For histories c, ϵ, n, a compact-net witness whose center and net are history-same has empty precision:*

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \implies \mathsf{hsame}(c, n) \implies \mathsf{hsame}(\epsilon, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness endpoint cycle precision empty`

Proof. Transport the witness continuation $\mathsf{Cont}(c, \epsilon, n)$ across $\mathsf{hsame}(c, n)$ to obtain a self-extension of c. Right-unit uniqueness forces the precision tail to be empty. □

Theorem 60.63 (Compact empty-center nonempty precision shape)**.** *For histories net, ϵ, a compact-net witness with empty center and nonempty precision has exactly one visible unary payload:*

$$\mathsf{CompactNetWitness}(\mathsf{e}, \epsilon, net) \wedge \neg\mathsf{hsame}(\epsilon, \mathsf{e}) \iff \exists r : \mathsf{BHist}.\ net = \mathsf{E1}(r) \wedge \epsilon = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r).$$

Proof. The continuation field is $\mathsf{Cont}(\mathsf{e}, \epsilon, net)$, so left-unit exactness identifies net with ϵ. The nonempty precision premise removes the empty case, and unary membership removes zero-headed precision. In the visible case, the precision carrier exposes the unary payload and sameness fixes the net to the same visible history. Conversely, the displayed visible payload rebuilds the compact-net witness by the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness empty center nonempty precision shape iff`

Theorem 60.64 (Compact net witness left e1 result cases)**.** *For histories center, precision, net, a compact-net witness from a visible center to a visible net has only the empty precision case or a visible precision tail:*

$$\mathsf{CompactNetWitness}(\mathsf{E1}(center), precision, \mathsf{E1}(net)) \Longrightarrow (precision = \mathsf{e} \wedge \mathsf{UnaryHistory}(center) \wedge \mathsf{hsame}(center, net))$$

Proof. Unpack the compact-net witness to expose the unary carriers and the continuation ledger. Inverting the ledger with visible result $\mathsf{E1}(net)$ gives either empty precision or a visible precision tail. The empty branch strips the visible constructor from the center-net equality and inverts the center carrier to obtain $\mathsf{UnaryHistory}(center)$. The visible branch inverts the precision and net carriers and repacks the tail continuation as a compact-net witness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness left e1 result cases`

Theorem 60.65 (Compact net witness left e1 result e1 precision tail)**.** *For histories center, precisionTail, net, a compact-net witness whose center, precision, and result are all visible exposes the precision tail witness:*

$$\mathsf{CompactNetWitness}(\mathsf{E1}(center), \mathsf{E1}(precisionTail), \mathsf{E1}(net)) \Longrightarrow \mathsf{CompactNetWitness}(\mathsf{E1}(center), precisionTail, n$$

Proof. Apply the visible-result cases theorem. The empty-precision branch would identify $\mathsf{E1}(precisionTail)$ with e, which constructor disjointness excludes. The visible branch identifies the displayed precision tail with the recovered tail, and the recovered tail witness is the required compact-net witness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness left e1 result e1 precision tail`

Theorem 60.66 (Compact net witness right e1 result iff)**.** *For histories center, precision, net, visible precision and visible result are equivalent to the tail compact-net witness:*

$$\mathsf{CompactNetWitness}(center, \mathsf{E1}(precision), \mathsf{E1}(net)) \Longleftrightarrow \mathsf{CompactNetWitness}(center, precision, net).$$

Proof. For the forward direction, unpack the compact-net witness. Unary inversion removes the visible constructor from the precision and net carriers, and right-step inversion removes it from the continuation ledger.

For the reverse direction, unary closure rebuilds the visible precision and net carriers, while the one-step continuation rule lifts the tail ledger to the visible result. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness right e1 result iff`

Theorem 60.67 (Compact net witness suffix exactness)**.** *For histories p, c, ϵ, n, a compact-net witness with the same right suffix attached to precision and net is equivalent to unary suffix evidence together with the unsuffixed witness:*

$$\mathsf{CompactNetWitness}(c, \epsilon \cdot p, n \cdot p) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{CompactNetWitness}(c, \epsilon, n).$$

Proof. In the forward direction, unary append factorization recovers the suffix and the unsuffixed precision and net histories. The continuation ledger $n \cdot p = c \cdot (\epsilon \cdot p)$ is associated to $(c \cdot \epsilon) \cdot p$, and right cancellation by p recovers $n = c \cdot \epsilon$. In the reverse direction, unary append closure builds the suffixed precision and net histories, while associativity supplies the suffixed continuation ledger. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness suffix iff`

Theorem 60.68 (Compact net witness visible context exactness)**.** *For histories p, q, c, ϵ, n, a compact-net witness with visible left context on the center and net and visible right context on the precision and net is equivalent to the two unary context witnesses and the base compact-net witness:*

$$\mathsf{CompactNetWitness}(p{\cdot}c, \epsilon{\cdot}q, (p{\cdot}n){\cdot}q) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{CompactNetWitness}(c, \epsilon, n).$$

Proof. Peel the right suffix by Theorem 60.67, then peel the left prefix by Theorem 60.59. The reverse direction first applies prefix exactness and then reattaches the right suffix by suffix exactness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context iff`

Theorem 60.69 (Compact net witness visible-result context factorization)**.** *For histories $p, q, c, \epsilon, n, result$, if $\mathsf{CompactNetWitness}(p{\cdot}c, \epsilon{\cdot}q, result)$ and $result \mathsf{hsame}(p{\cdot}n){\cdot}q$, then $\mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{CompactNetWitness}(c, \epsilon, n)$.*

Proof. Transport the result carrier and continuation ledger along the displayed hsame, then apply visible-context exactness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible result context factorizes`

Theorem 60.70 (Compact net witness visible-context result determinacy)**.** *For histories p, q, c, ϵ, n, n', two visible-context compact-net witnesses with the same visible center and precision contexts determine the same central net history:*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon \cdot q, (p \cdot n) \cdot q) \Longrightarrow \mathsf{CompactNetWitness}(p \cdot c, \epsilon \cdot q, (p \cdot n') \cdot q) \Longrightarrow \mathsf{hsame}(n, n').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central compact-net witnesses have the same center c and precision ϵ, so continuation result determinacy identifies the two central nets. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context result deterministic`

Theorem 60.71 (Compact net witness visible-context precision determinacy)**.** *For histories $p, q, c, \epsilon, \epsilon', n$, two visible-context compact-net witnesses with the same visible center context and the same visible net context determine the same central precision history:*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon \cdot q, (p \cdot n) \cdot q) \Longrightarrow \mathsf{CompactNetWitness}(p \cdot c, \epsilon' \cdot q, (p \cdot n) \cdot q) \Longrightarrow \mathsf{hsame}(\epsilon, \epsilon').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central compact-net witnesses have the same center c and net n, so left continuation cancellation identifies the two central precision histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context precision deterministic`

Theorem 60.72 (Compact net witness visible-context center determinacy)**.** *For histories p, q, c, c', ϵ, n, two visible-context compact-net witnesses with the same visible precision context and the same visible net context determine the same central center history:*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon \cdot q, (p \cdot n) \cdot q) \Longrightarrow \mathsf{CompactNetWitness}(p \cdot c', \epsilon \cdot q, (p \cdot n) \cdot q) \Longrightarrow \mathsf{hsame}(c, c').$$

Proof. Apply visible-context exactness to both witnesses. The recovered central compact-net witnesses have the same precision ϵ and net n, so right continuation cancellation identifies the two central center histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context center deterministic`

Theorem 60.73 (Compact net witness common composite result prefix determinacy)**.** *For histories p, p', c, θ, out, if two compact-net witnesses share the same composite precision and result while their centers differ only by the visible left prefix before the same c, then the two visible prefixes are* hsame*:*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, out) \Longrightarrow \mathsf{CompactNetWitness}(p' \cdot c, \theta, out) \Longrightarrow p\mathsf{hsame}p'.$$

Proof. Unpack the two compact-net witnesses. Their continuation ledgers have the same precision θ and the same result out, so right cancellation identifies $p \cdot c$ with $p' \cdot c$. Cancelling the common suffix c gives $p\mathsf{hsame}p'$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness common composite result prefix deterministic`

Theorem 60.74 (Compact net witness empty-precision visible-context exactness)**.** *For histories p, q, c, n, a visible-context compact-net witness with empty central precision is equivalent to unary evidence for the visible ledgers and center, together with the center classifier for the net:*

$$\mathsf{CompactNetWitness}(p{\cdot}c, \mathsf{e}{\cdot}q, (p{\cdot}n){\cdot}q) \Longleftrightarrow \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(c) \wedge \mathsf{hsame}(n, c).$$

Proof. Peel the visible prefix and suffix by Theorem 60.68. The central witness is $\mathsf{CompactNetWitness}(c, \mathsf{e}, n)$, whose continuation field is exactly the right-unit continuation classifier $\mathsf{hsame}(n, c)$; its carrier field gives $\mathsf{UnaryHistory}(c)$.

Conversely, the unary center carrier and $\mathsf{hsame}(n, c)$ transport the center carrier to the net and produce the right-unit continuation witness. This builds $\mathsf{CompactNetWitness}(c, \mathsf{e}, n)$, and visible-context exactness reattaches the prefix p and suffix q. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness empty precision visible context iff`

Theorem 60.75 (Compact visible result excludes empty center)**.** *For histories p, q, c, t, a visible-context compact-net witness with empty central precision and visible net $\mathsf{E1}(t)$ cannot have an empty central center:*

$$\mathsf{CompactNetWitness}(p \cdot c, \mathsf{e} \cdot q, (p \cdot \mathsf{E1}(t)) \cdot q) \Longrightarrow \mathsf{hsame}(c, \mathsf{e}) \Longrightarrow \bot.$$

Proof. Empty-precision visible-context exactness reads the central data back as $\mathsf{hsame}(\mathsf{E1}(t), c)$. If $c\mathsf{hsame}\mathsf{e}$, transitivity identifies $\mathsf{E1}(t)$ with e, contradicting BEDC constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context e1 result empty center absurd`

Theorem 60.76 (Compact visible boundary excludes empty center)**.** *If the net in a visible-context empty-precision compact witness is* hsame *to either $\mathsf{E0}(t)$ or $\mathsf{E1}(t)$, then the center cannot be empty.*

Proof. Empty-precision visible-context exactness identifies the net with the center; an empty center would identify the visible result with e, contradicting constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness visible context visible result empty center absurd`

Theorem 60.77 (Compact net witness composite result determinacy)**.** *For histories $c, \epsilon, \eta, n, n', m, m'$, if n and n' are compact net witnesses from the same center c and precision ϵ, and m and m' continue them with the same refinement precision η, then the refined nets are the same BEDC history:*

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \Longrightarrow \mathsf{CompactNetWitness}(n, \eta, m) \Longrightarrow \mathsf{CompactNetWitness}(c, \epsilon, n') \Longrightarrow \mathsf{CompactNetWitn}$$

Proof. Unpack the first and third witnesses into continuation ledgers $c \cdot \epsilon = n$ and $c \cdot \epsilon = n'$. Continuation determinacy gives $\mathsf{hsame}(n, n')$. The second and fourth witnesses give ledgers $n \cdot \eta = m$ and $n' \cdot \eta = m'$. Continuation compatibility with $\mathsf{hsame}(n, n')$ and reflexivity of η compares these second-step ledgers and yields $\mathsf{hsame}(m, m')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness composite result deterministic`

Theorem 60.78 (Compact net witness composite middle continuation determinacy)**.** *For histories $c, \epsilon, \eta, n, n', m$, if $\mathsf{CompactNetWitness}(c, \epsilon, n)$, $\mathsf{CompactNetWitness}(n, \eta, m)$, $\mathsf{CompactNetWitness}(c, \epsilon, n')$, and $\mathsf{CompactNetWitness}(n', \eta, m)$ hold, then*

$$\mathsf{Cont}(c, \epsilon, n) \wedge \mathsf{Cont}(c, \epsilon, n') \wedge \mathsf{hsame}(n, n').$$

Proof. The first and third compact-net witnesses expose the concrete first-stage continuation ledgers $\mathsf{Cont}(c, \epsilon, n)$ and $\mathsf{Cont}(c, \epsilon, n')$. Continuation determinacy compares those two ledgers with the same source and precision, giving $\mathsf{hsame}(n, n')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness composite middle cont deterministic`

Theorem 60.79 (Compact net witness composite center determinacy)**.** *For histories $c, c', \epsilon, \eta, n, n', m$, if the first-stage net witnesses share precision ϵ and the second-stage refinements share extra precision η and final net m, then their centers are the same BEDC history:*

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \Longrightarrow \mathsf{CompactNetWitness}(n, \eta, m) \Longrightarrow \mathsf{CompactNetWitness}(c', \epsilon, n') \Longrightarrow \mathsf{CompactNetWit}$$

Proof. Unpack the second-stage witnesses into continuation ledgers $\mathsf{Cont}(n, \eta, m)$ and $\mathsf{Cont}(n', \eta, m)$. Continuation right cancellation gives $\mathsf{hsame}(n, n')$. The first-stage witnesses provide $\mathsf{Cont}(c, \epsilon, n)$ and $\mathsf{Cont}(c', \epsilon, n')$; cancellation over the common suffix ϵ and the already identified net histories yields $\mathsf{hsame}(c, c')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness composite center deterministic`

Theorem 60.80 (Compact net witness common precision center transport)**.** *For histories c, c', ϵ, n, n', if $\mathsf{CompactNetWitness}(c, \epsilon, n)$, $\mathsf{CompactNetWitness}(c', \epsilon, n')$, and $\mathsf{hsame}(n, n')$ hold, then the second witness transports to the first net and the centers are the same BEDC history:*

$$\mathsf{Cont}(c', \epsilon, n) \wedge \mathsf{hsame}(c, c').$$

Proof. The second compact-net witness exposes $\mathsf{Cont}(c', \epsilon, n')$; endpoint transport along $\mathsf{hsame}(n, n')$ gives $\mathsf{Cont}(c', \epsilon, n)$. The two first-stage continuation ledgers now share the same precision and history-same results, so common-suffix cancellation identifies c and c'. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness common precision center transport`

Theorem 60.81 (Compact net witness common center precision transport)**.** *For histories $c, \epsilon, \epsilon', n, n'$, if $\mathsf{CompactNetWitness}(c, \epsilon, n)$, $\mathsf{CompactNetWitness}(c, \epsilon', n')$, and $\mathsf{hsame}(n, n')$ hold, then the second witness transports to the first net and the precisions are the same BEDC history:*

$$\mathsf{Cont}(c, \epsilon', n) \wedge \mathsf{hsame}(\epsilon, \epsilon').$$

Proof. The second compact-net witness exposes $\mathsf{Cont}(c, \epsilon', n')$; endpoint transport along $\mathsf{hsame}(n, n')$ gives $\mathsf{Cont}(c, \epsilon', n)$. Since the center is common and the endpoints are history-same, compact-net precision determinacy identifies ϵ and ϵ'. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness common center precision transport`

Theorem 60.82 (Compact net witness precision determinacy under endpoint sameness)**.** *For histories $c, c', \epsilon, \epsilon', n, n'$, if $\mathsf{hsame}(c, c')$, $\mathsf{hsame}(n, n')$, $\mathsf{CompactNetWitness}(c, \epsilon, n)$, and $\mathsf{CompactNetWitness}(c', \epsilon', n')$ hold, then*

$$\mathsf{hsame}(\epsilon, \epsilon').$$

Proof. Transport the two continuation ledgers to common center and net histories using the endpoint hsame hypotheses. The two witnesses then have the same left context and result, so continuation left cancellation identifies the precision histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness precision hsame deterministic`

Theorem 60.83 (Compact net witness composite precision continuation determinacy)**.** *For histories $c, \epsilon, \epsilon', \eta, n, n', m$, if $\mathsf{CompactNetWitness}(c, \epsilon, n)$ and $\mathsf{CompactNetWitness}(c, \epsilon', n')$ hold and both n and n' refine through the same extra precision η to the same history m, then*

$$\mathsf{hsame}(\epsilon, \epsilon').$$

Proof. Continuation right cancellation on the two refinement ledgers identifies n and n'. With the center already common, compact-net precision determinacy under endpoint sameness identifies ϵ and ϵ'. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness composite precision cont deterministic`

Corollary 60.84 (Compact source specification route determinacy)**.** *Fix a consumer-facing $\mathsf{Compact}^{\uparrow}$ use whose source side is the compact source specification of Definition 60.53. For histories p, c, ϵ, n, n', suppose the displayed local rows are the same prefixed center, precision, continuation route, and compact-net result shape:*

$$\mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c, \epsilon, p \cdot n').$$

Then the unprefixed result rows selected by the two consumer uses are the same BEDC history:

$$\mathsf{hsame}(n, n').$$

Consequently every accepted consumer-facing use of the compact route, compact-net normal-form read, or exactness decision over these fixed source rows factors through the same finite route: the compact source row, the prefixed center row $p \cdot c$, the precision row ϵ, the compact-net witness row, the continuation ledger $\mathsf{Cont}(p \cdot c, \epsilon, -)$, the result readback row $p \cdot n$, and the classifier row hsame. There is no alternate hidden coordinate, second compact-net result source, or duplicate classifier source outside that displayed packet.

Proof. The compact source specification of Definition 60.53 supplies only the certified metric source together with the subset predicate over its carrier. In this certificate, a consumer-facing compact-net use therefore reaches the compact source through the displayed compact-net witness row and not through an additional selector, hidden coordinate, or external classifier source.

The two displayed consumer uses have the same prefixed center $p \cdot c$ and the same precision ϵ. Their only result rows are $p \cdot n$ and $p \cdot n'$, each carried by a compact-net witness over those fixed source rows. Thus they are exactly in the situation of Definition 60.60. Applying that theorem gives $\mathsf{hsame}(n, n')$ after cancellation of the common prefix p.

It remains to account for the named consumer readings. The route read is the prefixed center and precision route just used. A compact-net normal-form read is a readback of the same result row, and an exactness decision named in the chapter is a decision about the same compact-net witness packet. Since the source specification adds no further coordinate and the determinacy theorem has already identified any competing unprefixed result, every accepted use over the fixed rows factors through the listed finite route and the same hsame classifier row. □

Theorem 60.85 (Prefixed composite compact net witness closure)**.** *For unary histories $p, c, \epsilon, \eta, \theta, n, m$, assume* $\mathsf{UnaryHistory}(p)$, $\mathsf{CompactNetWitness}(c, \epsilon, n)$, $\mathsf{CompactNetWitness}(n, \eta, m)$, *and* $\mathsf{Cont}(\epsilon, \eta, \theta)$. *Then the composed refinement is a compact net witness under the same visible prefix:*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m).$$

The certificate fit has ten fields: the traditional target is finite-net precision composition; the BEDC source is BHist equipped with Cont*; the carrier is* $\mathsf{CompactNetWitness}$*; the classifier is hsame; the stability law is closure under prefix and precision composition; the ledger data are center, precision, and net continuations; the core theorem is this prefixed composite closure; exactness is the direct composed witness; the standard bridge is deferred; the status is* $\mathsf{ProofTarget}$.

Proof. Unpack the two compact net witnesses into unary center, precision, and net histories, together with continuation ledgers $n = c \cdot \epsilon$ and $m = n \cdot \eta$.

The continuation hypothesis $\mathsf{Cont}(\epsilon, \eta, \theta)$ gives the composed precision history. Unary continuation closure supplies the unary history for θ, and associativity identifies $(c \cdot \epsilon) \cdot \eta$ with $c \cdot (\epsilon \cdot \eta)$.

The identified composite ledger presents $m = c \cdot \theta$. Applying prefix closure to this composed witness and using associativity once more yields the ledger from $p \cdot c$ to $p \cdot m$ with precision θ. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite closed`

Theorem 60.86 (Prefixed composite compact net witness factorization)**.** *For histories $p, c, \epsilon, \eta, \theta, m$, assume* $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, $\mathsf{Cont}(\epsilon, \eta, \theta)$, *and* $\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m)$. *Then there exists a history n such that*

$$\mathsf{CompactNetWitness}(c, \epsilon, n)$$

and

$$\mathsf{CompactNetWitness}(n, \eta, m).$$

Proof. Apply Theorem 60.59 to the visible-prefix witness. This removes the common prefix and recovers $\mathsf{CompactNetWitness}(c, \theta, m)$, hence the unary histories for c, θ, m and the continuation ledger $\mathsf{Cont}(c, \theta, m)$.

Choose the intermediate history $n = c \cdot \epsilon$. The unary history of ϵ and the recovered unary history of c give unary n, and the continuation constructor gives $\mathsf{Cont}(c, \epsilon, n)$. These data form $\mathsf{CompactNetWitness}(c, \epsilon, n)$.

It remains to build the second witness. The hypothesis $\mathsf{Cont}(\epsilon, \eta, \theta)$ and the recovered continuation $\mathsf{Cont}(c, \theta, m)$ identify the ledger m with $c \cdot (\epsilon \cdot \eta)$; associativity rewrites this as $(c \cdot \epsilon) \cdot \eta = n \cdot \eta$. Together with the unary histories of n, η, and m, this is $\mathsf{CompactNetWitness}(n, \eta, m)$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite factorizes`

Theorem 60.87 (Prefixed composite compact net middle deterministic factorization)**.** *For histories $p, c, \epsilon, \eta, \theta, m$, assume* $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, *and* $\mathsf{Cont}(\epsilon, \eta, \theta)$. *If*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m),$$

then there exists a history n such that

$$\mathsf{CompactNetWitness}(c, \epsilon, n)$$

and

$$\mathsf{CompactNetWitness}(n, \eta, m).$$

Moreover any other history n' supporting these two compact-net witnesses is history-same to n.

Proof. Apply the prefixed composite factorization theorem to obtain an intermediate history n and the two compact-net witnesses through ϵ and η. For any other displayed intermediate n', the first compact-net witnesses for n and n' share the same center c and precision ϵ; continuation determinacy therefore gives $\mathsf{hsame}(n, n')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite middle deterministic factorization`

Definition 60.88 (Two-step compact net factor)**.** *For histories c, ϵ, η, m, the predicate* $\mathsf{CompactNetTwoStepFactor}($ *records an intermediate net realizing the two visible precision steps:*

$$\mathsf{CompactNetTwoStepFactor}(c, \epsilon, \eta, m) \iff \exists n.\ \mathsf{CompactNetWitness}(c, \epsilon, n) \wedge \mathsf{CompactNetWitness}(n, \eta, m).$$

Lean4 (definition) `BEDC.Derived.CompactUp.CompactNetTwoStepFactor`

Theorem 60.89 (Two-step compact net factor composed closure)**.** *For histories $c, \epsilon, \eta, \theta, m$, assume* $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, *and* $\mathsf{Cont}(\epsilon, \eta, \theta)$. *Then every displayed two-step factor composes to a compact-net witness at the composite precision:*

$$\mathsf{CompactNetTwoStepFactor}(c, \epsilon, \eta, m) \implies \mathsf{CompactNetWitness}(c, \theta, m).$$

Proof. Choose the intermediate net from the two-step factor. Apply prefixed composite closure with the empty visible prefix to compose the two compact-net witnesses through $\mathsf{Cont}(\epsilon, \eta, \theta)$. The compact-net prefix equivalence then removes the empty prefix and returns the unprefixed witness. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetTwoStepFactor composed closed`

Theorem 60.90 (Prefixed composite compact net witness exactness)**.** *For histories $p, c, \epsilon, \eta, \theta, m$, assume* $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, *and* $\mathsf{Cont}(\epsilon, \eta, \theta)$. *Then the prefixed composed certificate is exact at the compact-net witness level:*

$$\mathsf{CompactNetWitness}(p{\cdot}c, \theta, p{\cdot}m) \iff \mathsf{UnaryHistory}(p) \wedge \exists n, \mathsf{CompactNetWitness}(c, \epsilon, n) \wedge \mathsf{CompactNetWitness}(n, \eta, m)$$

The certificate fit has ten fields: the traditional target is finite-net precision composition; the BEDC source is BHist equipped with Cont*; the carrier is* $\mathsf{CompactNetWitness}$*; the classifier is hsame; the stability obligation is prefixed composite exactness; the ledger data are center, precision, intermediate net, and final net continuations; the core theorem is this equivalence; the exactness predicate is the displayed equivalence; the standard bridge is deferred; the status is* $\mathsf{ProofTarget}$.

Proof. For the forward direction, apply Theorem 60.59 to the prefixed composed witness. This recovers $\mathsf{UnaryHistory}(p)$ together with the unprefixed witness $\mathsf{CompactNetWitness}(c, \theta, m)$. The hypotheses on ϵ, η, and θ then allow Theorem 60.86 to produce an intermediate history n with the two compact-net witnesses required on the right-hand side.

For the reverse direction, project the unary prefix evidence and the intermediate history n from the right-hand side. The two compact-net witnesses for c through ϵ and for n through η, together with $\mathsf{Cont}(\epsilon, \eta, \theta)$, satisfy the hypotheses of Theorem 60.85. Applying that closure theorem yields $\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m)$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite iff`

Theorem 60.91 (Prefixed composite compact net result determinacy)**.** *For histories $p, c, \epsilon, \eta, \theta, m, m'$, assume $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, and $\mathsf{Cont}(\epsilon, \eta, \theta)$. If both prefixed composed net witnesses start from the same prefixed center and composed precision,*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m'),$$

then the visible refined nets are history-same:

$$\mathsf{hsame}(m, m').$$

Proof. Apply the prefixed composite factorization theorem to each prefixed composed witness. This produces two unprefixed two-step compact-net chains from the same center c, through the same precision ϵ, and then through the same refinement precision η. Composite-result determinacy for compact net witnesses compares those two unprefixed chains and yields $\mathsf{hsame}(m, m')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite result deterministic`

Theorem 60.92 (Prefixed composite compact net prefix-independent result determinacy)**.** *For histories $p, p', c, \epsilon, \eta, \theta, m, m'$, assume $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, and $\mathsf{Cont}(\epsilon, \eta, \theta)$. If two composed compact-net witnesses share the same unprefixed center and composed precision but use different visible prefixes,*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p' \cdot c, \theta, p' \cdot m'),$$

then the refined nets are history-same:

$$\mathsf{hsame}(m, m').$$

Proof. Apply prefixed composite factorization to both witnesses, using the same precision histories ϵ, η and composed precision θ. This removes the visible prefixes and recovers two unprefixed two-step compact-net chains from the same center c through the same precision sequence. The unprefixed composite-result determinacy theorem compares those recovered chains and gives $\mathsf{hsame}(m, m')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite prefix independent result deterministic`

Theorem 60.93 (Prefixed composite compact net center determinacy)**.** *For histories $p, c, c', \epsilon, \eta, \theta, m$, assume $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\eta)$, and $\mathsf{Cont}(\epsilon, \eta, \theta)$. If two prefixed composed net witnesses have the same composed precision and the same prefixed refined net,*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c', \theta, p \cdot m),$$

then their unprefixed centers are history-same:

$$\mathsf{hsame}(c, c').$$

Proof. Apply prefixed composite factorization to both prefixed witnesses. This recovers two unprefixed two-step compact-net chains ending at the same refined net and using the same precision histories ϵ and η. The unprefixed composite center determinacy theorem compares the recovered chains and yields $\mathsf{hsame}(c, c')$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite center deterministic`

Theorem 60.94 (Prefixed composite compact net precision determinacy)**.** *For histories $p, c, \epsilon, \epsilon', \eta, \theta, \theta', m$, assume* $\mathsf{UnaryHistory}(\epsilon)$, $\mathsf{UnaryHistory}(\epsilon')$, $\mathsf{UnaryHistory}(\eta)$, $\mathsf{Cont}(\epsilon, \eta, \theta)$, *and* $\mathsf{Cont}(\epsilon', \eta, \theta')$. *If two prefixed composed net witnesses have the same prefixed center and the same prefixed refined net,*

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m) \quad \textit{and} \quad \mathsf{CompactNetWitness}(p \cdot c, \theta', p \cdot m),$$

then their first precision histories are history-same:

$$\mathsf{hsame}(\epsilon, \epsilon').$$

Proof. Unpack the two prefixed compact-net witnesses. Since they share the same prefixed center and prefixed refined net, left cancellation on their continuation ledgers identifies the composite precision histories θ and θ'. The two composite ledgers then have the same right suffix η, so continuation suffix cancellation identifies the first precision histories. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetWitness prefixed composite precision deterministic`

Theorem 60.95 (Compact two-step finite-net gluing)**.** *Let K be a* $\mathsf{Compact}^{\uparrow}$ *witness carried by the finite-net setup of Definition 60.1 inside a metric carrier as in Definition 57.1. Let ε, δ, and θ be valid precision ledgers with*

$$\mathsf{Cont}(\varepsilon, \delta, \theta).$$

Assume $N \subseteq K$ is a finite ε-net for K. For each $n \in N$, let

$$B_K(n, \varepsilon) = \{y \in K \mid d(y, n) \leq \varepsilon\},$$

and assume $N_n \subseteq K$ is a finite δ-net for $B_K(n, \varepsilon)$. Then

$$N^* = \bigcup_{n \in N} N_n$$

is finite, is contained in K, and is a finite θ-net for K. Under the real-budget reading of the continuation ledger, this is a finite $(\varepsilon + \delta)$-net for K.

Moreover, every displayed two-step choice

$$\mathsf{CompactNetWitness}(c, \varepsilon, n) \wedge \mathsf{CompactNetWitness}(n, \delta, m)$$

from the construction is a $\mathsf{CompactNetTwoStepFactor}(c, \varepsilon, \delta, m)$ *and induces the composed witness* $\mathsf{CompactNetWitness}(c, \theta, m)$.

Proof. The finiteness and carrier clauses are immediate by finite induction over the displayed list N. The empty list gives the empty union. In the step case, adjoining one finite local net N_n preserves finiteness by finite-union closure, and containment in K follows from the hypotheses $N_n \subseteq K$.

Fix $x \in K$. Since N is an ε-net for K, choose $n \in N$ with $d(x, n) \leq \varepsilon$. Because $N \subseteq K$ and the metric identity law gives $d(n, n) = 0$, the point n lies in $B_K(n, \varepsilon)$. The local δ-net N_n therefore supplies $m \in N_n$ with $d(n, m) \leq \delta$. Since $N_n \subseteq N^*$, this m is a candidate center in the glued finite set.

The triangle inequality field of Definition 57.5 gives

$$d(x, m) \leq d(x, n) + d(n, m) \leq \varepsilon + \delta.$$

The precision continuation $\mathsf{Cont}(\varepsilon, \delta, \theta)$ is exactly the BEDC budget ledger for the two visible steps. Thus the displayed point $m \in N^*$ witnesses the θ-coverage of x. Since x was arbitrary, N^* is a finite θ-net for K.

For the BEDC witness clause, Definition 60.88 repackages the two displayed compact-net witnesses as the two-step factor. Applying Definition 60.85 with empty visible prefix composes the two precision ledgers through $\mathsf{Cont}(\varepsilon, \delta, \theta)$ and returns $\mathsf{CompactNetWitness}(c, \theta, m)$. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactNetTwoStepFactor gluing package`

Corollary 60.96 (Prefixed composite compact net witness closure determinacy)**.** *For histories $p, c, \epsilon, \eta, \theta, n, n', m$, assume the fixed local rows*

$$\mathsf{UnaryHistory}(p), \quad \mathsf{UnaryHistory}(\epsilon), \quad \mathsf{UnaryHistory}(\eta), \quad \mathsf{Cont}(\epsilon, \eta, \theta),$$

and the displayed two-step compact-net route

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \quad \textit{and} \quad \mathsf{CompactNetWitness}(n, \eta, m).$$

If another consumer-facing use of the same closure packet presents a middle history n' with

$$\mathsf{CompactNetWitness}(c, \epsilon, n') \quad \textit{and} \quad \mathsf{CompactNetWitness}(n', \eta, m),$$

then the closure packet has the same accepted exactness row

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m)$$

and the two proposed middle routes determine the same chapter normal form:

$$\mathsf{hsame}(n, n').$$

Consequently an accepted use of these fixed rows cannot choose an unlisted middle coordinate, a second finite route, or a duplicate classifier source for the closed prefixed composite witness.

Proof. Apply Definition 60.85 to the unary prefix row, the two displayed compact-net witnesses, and the continuation row $\mathsf{Cont}(\epsilon, \eta, \theta)$. This gives the closed prefixed composite witness

$$\mathsf{CompactNetWitness}(p \cdot c, \theta, p \cdot m).$$

Now apply Definition 60.87 to this closed witness. The theorem factors every accepted use of the closed row through the displayed middle factorization rows for the fixed ϵ, η, θ packet, and its determinacy clause compares any two such displayed middle histories.

The original route through n is one displayed middle factorization, because it carries

$$\mathsf{CompactNetWitness}(c, \epsilon, n) \quad \text{and} \quad \mathsf{CompactNetWitness}(n, \eta, m).$$

The consumer route through n' is another displayed middle factorization over the same center, first precision, second precision, and endpoint, by the two assumed rows for n'. The determinacy clause of Definition 60.87, read through the continuation transport row of Definition 4.41, therefore identifies the two middle histories by the chapter classifier, namely $\mathsf{hsame}(n, n')$.

Thus the finite route is fixed up to hsame, the normal-form endpoint remains the displayed m, and the exactness decision is exactly the closed witness produced by the prefixed composite closure theorem. There is no remaining accepted coordinate from which a consumer can read a different middle route or name a separate classifier source. □

Definition 60.97 (Compact ledger policy)**.** *Record the subset predicate, locatedness decisions, and finite net ledgers.*

Definition 60.98 ($\mathsf{Compact}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Compact}^{\uparrow}$*.*

Primitive boundary. $\mathsf{Compact}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 60.99 (Zero-axiom encoding note)**.** *Open-cover compactness is not the exported carrier for* $\mathsf{Compact}^{\uparrow}$*; the exported carrier is locatedness plus explicit finite-net data.*

Theorem 60.100 (Compact witness public surface bridge)**.** *The compact-net semantic certificate, the located and finite refinement chains, the endpoint classifier rows, and the refinement-witness ledger determine the conservative standard finite-net compact-witness schema for* $\mathsf{Compact}^{\uparrow}$*. The public object consists of a unary center, a unary precision ledger, the displayed answer net, the locatedness and finite-net refinement endpoints, and the visible continuation evidence connecting those histories.*

The bridge is conservative: it exposes only finite-net compact-witness data already carried by the BEDC rows. It does not export open-cover compactness, sequential compactness, product compactness, or function-space compactness.

Proof. Use the compact-net semantic name certificate `BEDC.Derived.CompactUp.CompactNetWitness_semanticN` as the canonical certificate row. Its carrier reads a center, a precision, and a net with the net obtained by the displayed continuation from the center and precision, so the public finite-net witness has no coordinate outside the BEDC histories.

Next apply the compact witness carrier and refinement-chain endpoint packages, including Definition 60.41, Definition 60.42, and Definition 60.52. They supply the located and finite refinement endpoints, their unary rows, and the endpoint classifier equivalences used by the public witness schema.

The refinement-witness ledger is exactly the displayed continuation data in those packages. Forgetting the BEDC wrapper yields the ordinary finite-net compact-witness reading, while repacking the same rows returns the BEDC certificate. Since every cited row is a finite history, classifier, or continuation row, none of the excluded compactness principles has a carrier slot in the bridge. □

Lean4 ✓ `BEDC.Derived.CompactUp.CompactUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Compact}^{\uparrow}}$.

Constructive story: A compact witness in this scoped package is a finite refinement record over unary histories: a subset history, locatedness ledger, finite-net ledger, intermediate ledger, and compactness ledger are connected by displayed continuations, and the simpler net witness records a center, precision, and net with the net obtained by appending the precision to the center. The classifier is endpoint history-sameness on the exposed net or refinement endpoints, with determinacy rows showing that shared centers, precisions, composites, or final ledgers force the matching fields to be hsame. The source is the BEDC compact-net or refinement predicate, the pattern records finite and located refinement chains, prefix closure, endpoint exactness, empty-net and nonempty-precision shapes, and composite refinement determinacy, and the ledger keeps every located, finite, intermediate, compact, precision, and net continuation visible. Because a unary center and precision always build the witness net $center \cdot precision$, hsame supplies the classifier equivalence, and the refinement predicates transport along deterministic continuation equalities, the five name-certificate obligations hold for the semantic compact-net certificate. A history n lies in the signature gap of a $\mathsf{Compact}^{\uparrow}$ token p when the source-domain accepts it as a compact net or final refinement witness, the signature relation pairs it with an answer net and evidence ledger, and token introduction names that answer as p; unfolded, n is unary, is connected to its center and precision by the displayed continuation, and has the refinement endpoint data encoded by p. The public bridge exposes exactly the conservative finite-net compact-witness schema of Definition 60.100, so open-cover compactness, sequential compactness, products, and function-space compactness stay outside the closed surface. The chapter registers this finite-refinement net construction under the name $\mathsf{Compact}^{\uparrow}$; compactness is a ledger of explicit unary refinements, not an imported topological compactness axiom or host-space object.

Theory closure: TheoryGate(NameCert$_{\mathsf{Compact}^{\uparrow}}$, maturePackageClosed)

Scope closed: The compact-net-witness semantic name certificate, with located/finite refinement chain, endpoint classifier, and refinement-witness ledger over the BEDC source.

Formal verification: FormalStatus(NameCert$_{\mathsf{Compact}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CompactUp.CompactNetWitness\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: General theory of compact spaces: arbitrary open-cover compactness, sequential compactness in non-metric settings, products of compact spaces (Tychonoff), or compactness of function spaces.

Upgrade path: No further formal-axis step is recorded for this chapter because the standard bridge is checked.

Chapter 61

A Concrete Naming Certificate for $\mathsf{Cat}^{\uparrow}$

The category interface $\mathsf{Cat}^{\uparrow}$ has certificate obligations over an object carrier together with, for each ordered pair of objects, a morphism carrier, identity morphisms at each object, a binary composition operation respecting source and target, and the associativity and identity laws expressed as morphism-classifier sameness. The size discipline is BEDC-internal: object and morphism carriers are each covered by their own NameCert obligations, avoiding any global "set of all objects" assumption. It is not a primitive structure; it is admitted only as a multi-sorted carrier with relational composition.

61.1 Carrier definition

Definition 61.1 (Category carrier)**.** *Use a multi-sorted setup with object carrier, morphism carrier for each ordered object pair, identities, composition, and law fields.*

Lean4 (definition) `BEDC.Derived.CategoryUp.CategoryHomCarrier`

Theorem 61.2 (Category empty identity hom carrier)**.** *For the unary-continuation category, if h is unary, then the empty history is a hom-carrier from h to h.*

Proof. The source and target carrier witnesses are the given unary witness for h. The morphism carrier is the unary empty history, and the continuation field is the right-unit continuation $\mathsf{Cont}(h, \mathsf{e}, h)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty identity`

Theorem 61.3 (Category empty identity hom carrier exactness)**.** *For histories a, b, the empty history is a hom-carrier from a to b exactly when a and b are unary and internally the same:*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Unpack the hom-carrier witness. Its continuation field is $\mathsf{Cont}(a, \mathsf{e}, b)$; determinism against the right-unit continuation $\mathsf{Cont}(a, \mathsf{e}, a)$ identifies a and b. Conversely, unary source and target witnesses together with $\mathsf{hsame}(a, b)$ transport the right-unit continuation to a continuation ending at b, while e supplies the empty morphism carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty identity iff`

Theorem 61.4 (Category empty morphism endpoint exactness)**.** *For histories a, b : BHist, the empty history is a hom-carrier from a to b exactly when a is unary and $a$hsame$b$:*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(a, b).$$

Proof. The forward direction applies empty identity hom-carrier exactness and keeps the source carrier together with the endpoint classifier. Conversely, the classifier $a$hsame$b$ transports the unary source carrier from a to b, and empty identity exactness rebuilds the displayed empty hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty morphism endpoint iff`

Theorem 61.5 (Category empty morphism endpoint induction)**.** *Let P : BHist → Prop be any predicate with base case $P(\mathsf{e})$ and one-step closure*

$$\forall h : \mathsf{BHist}, \quad \mathsf{UnaryHistory}(h) \to P(h) \to P(\mathsf{E1}(h)).$$

If $\mathsf{CategoryHomCarrier}(a, b, \mathsf{e})$ *holds, then $P(a) \wedge P(b)$.*

Proof. Empty-morphism endpoint exactness identifies the hom-carrier with a unary source a and a classifier $a$hsame$b$. Unary induction gives $P(a)$ from the base and step clauses. The endpoint classifier transports this value from a to b, producing both endpoint conclusions. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty morphism endpoint induction`

Theorem 61.6 (Category endomorphism tail empty exactness)**.** *For a unary-continuation endomorphism carrier, the morphism tail is externally empty exactly when the source object and morphism tail are unary and the tail is internally the empty history:*

$$\mathsf{CategoryHomCarrier}(a, a, f) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(f) \wedge f\mathsf{hsame}\mathsf{e}.$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier endomorphism tail empty iff`

Proof. Unpack the endomorphism carrier. Its continuation field is $\mathsf{Cont}(a, f, a)$, so right-unit uniqueness for continuation forces $f$hsame$\mathsf{e}$, while the carrier fields give unary membership of a and f. Conversely, if $f$hsame$\mathsf{e}$, substituting the empty tail rebuilds the canonical empty identity hom-carrier at a. □

Theorem 61.7 (Category hom carrier endomorphism empty exactness)**.** *For histories a, f : BHist, an endomorphism hom carrier at a is exactly a unary object witness for a together with an empty morphism tail:*

$$\mathsf{CategoryHomCarrier}(a, a, f) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(f, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier endomorphism empty iff`

Proof. Unpack the hom-carrier witness. Its continuation field is $\mathsf{Cont}(a, f, a)$, so right-unit uniqueness forces $f$hsame$\mathsf{e}$, while the source unary witness is already present. Conversely, after eliminating the empty morphism-tail sameness witness, the empty identity hom carrier at the unary object a gives the required endomorphism carrier. □

Theorem 61.8 (Category one-headed source empty morphism exactness)**.** *For histories a and target, the empty morphism is a hom-carrier from $\mathsf{E1}(a)$ to target exactly when the target is $\mathsf{E1}(a)$ and the tail a is unary:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), target, \mathsf{e}) \iff target = \mathsf{E1}(a) \wedge \mathsf{UnaryHistory}(a).$$

Proof. Apply empty identity hom-carrier exactness to the source $\mathsf{E1}(a)$. The resulting history-sameness witness identifies *target* with $\mathsf{E1}(a)$, and unary inversion on the one-headed source gives unary membership of a. Conversely, after the displayed target equation is eliminated, unary closure under $\mathsf{E1}$ and the empty identity carrier rebuild the hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source empty morphism iff`

Theorem 61.9 (Category one-headed target empty morphism exactness)**.** *For histories $source, r$: BHist, the empty morphism is a hom-carrier from source to $\mathsf{E1}(r)$ exactly when the source is $\mathsf{E1}(r)$ and the tail r is unary:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{E1}(r), \mathsf{e}) \iff source = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r).$$

Proof. Apply empty identity hom-carrier exactness to the target $\mathsf{E1}(r)$. The resulting history-sameness witness identifies the source with $\mathsf{E1}(r)$, and unary inversion gives unary membership of r.

Conversely, after eliminating the displayed source equation, unary closure under $\mathsf{E1}$ gives the object carrier at $\mathsf{E1}(r)$, and the empty identity hom carrier rebuilds the hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target empty morphism iff`

Theorem 61.10 (Category empty-source hom carrier exactness)**.** *For histories b, f, a hom-carrier out of the empty source is exactly unary membership of the target together with sameness of the morphism and target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, f) \iff \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(f, b).$$

Proof. Unpack the hom-carrier witness. Its continuation field is $\mathsf{Cont}(\mathsf{e}, f, b)$, and the left-unit continuation exactness identifies b with f; symmetry gives $\mathsf{hsame}(f, b)$, while the target unary witness is already present. Conversely, transport the target unary witness across $\mathsf{hsame}(f, b)$ to obtain the morphism carrier, and use the left-unit continuation equivalence to rebuild $\mathsf{Cont}(\mathsf{e}, f, b)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source iff`

Theorem 61.11 (Category empty-source nonempty morphism shape)**.** *For histories $target, morph$, an empty-source hom-carrier with nonempty morphism has exactly one visible unary payload:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, target, morph) \wedge \neg\mathsf{hsame}(morph, \mathsf{e}) \iff \exists r : \mathsf{BHist}.\ target = \mathsf{E1}(r) \wedge morph = \mathsf{E1}(r) \wedge \mathsf{UnaryH}$$

Proof. Empty-source hom-carrier exactness identifies *morph* with *target* and gives unary membership of *target*. The nonempty morphism premise eliminates the empty target case, and unary inversion eliminates the zero-headed case. Thus the remaining target shape is $\mathsf{E1}(r)$, and sameness forces the morphism to be the same visible history. Conversely, a visible unary payload rebuilds the empty-source carrier by left-unit continuation and is not hsame to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source nonempty morphism shape iff`

Theorem 61.12 (Category empty-source empty-morphism exactness)**.** *For a history b, an empty-source hom-carrier whose displayed morphism is empty is exactly an empty target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, \mathsf{e}) \iff \mathsf{hsame}(b, \mathsf{e}).$$

Proof. Specialize empty-source hom-carrier exactness to the empty morphism. The forward direction returns $\mathsf{hsame}(\mathsf{e}, b)$, and symmetry gives $\mathsf{hsame}(b, \mathsf{e})$. Conversely, transport unary membership of e across the symmetric target equality and rebuild the empty-source hom-carrier by the same exactness theorem. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source empty morphism iff`

Theorem 61.13 (Category empty-source semantic name certificate)**.** *Fix a unary target history b. The empty-source hom-carrier* $\mathsf{CategoryHomCarrier}(\mathsf{e}, b, -)$*, used as source, pattern, and ledger policy with classifier* hsame*, forms a semantic name certificate.*

Proof. The carrier is inhabited by the displayed morphism b: empty-source hom-carrier exactness reduces this to unary membership of b and reflexivity $b\mathsf{hsame}b$. The classifier fields are reflexivity, symmetry, and transitivity of hsame. Carrier preservation transports a displayed hom-carrier along morphism sameness while the source e and target b remain fixed. Pattern and ledger soundness are identities because all three policies are the same hom-carrier predicate. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source semanticNameCert`

Theorem 61.14 (Category empty-target hom carrier exactness)**.** *For histories a, f, a hom-carrier into the empty target is exactly sameness of the source and morphism with the empty history:*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{e}, f) \iff \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}).$$

Proof. Unpack the hom-carrier witness. Its continuation field is $\mathsf{Cont}(a, f, \mathsf{e})$, and empty-result inversion for continuation splits the source and morphism into empty histories. Conversely, after those two sameness witnesses are eliminated, the empty source, empty target, and empty morphism are unary, and the right-unit continuation at e rebuilds the hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target iff`

Theorem 61.15 (Category empty-source empty-target same morphism boundary)**.** *For histories $a, b, f : \mathsf{BHist}$, if the same morphism f is displayed both as a hom-carrier out of e into b and as a hom-carrier from a into e, then all visible boundary histories read back to the empty history:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, f) \wedge \mathsf{CategoryHomCarrier}(a, \mathsf{e}, f) \Longrightarrow a\mathsf{hsamee} \wedge b\mathsf{hsamee} \wedge f\mathsf{hsamee}.$$

Proof. Empty-target exactness applied to the second hom-carrier gives $a\mathsf{hsamee}$ and $f\mathsf{hsamee}$. Empty-source exactness applied to the first hom-carrier gives $f\mathsf{hsame}b$. Symmetry of this morphism-readback equality, followed by transitivity with $f\mathsf{hsamee}$, gives $b\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source empty target same morphism boundary`

Theorem 61.16 (Continuation morphism empty target existence equivalence)**.** *For every source history source, there exists a continuation morphism from source to* e *exactly when the source is history-same to* e*:*

$$\mathsf{Nonempty}(\mathsf{ContinuationMorphism}(\mathit{source}, \mathsf{e})) \iff \mathsf{hsame}(\mathit{source}, \mathsf{e}).$$

Proof. Any displayed morphism into e has a continuation ledger ending at e, so empty-result inversion identifies its source with e. Conversely, if *source*hsamee, the right-unit continuation at *source* transports its result endpoint to e, giving the required empty-tail continuation morphism. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism empty target nonempty iff`

Theorem 61.17 (Continuation morphism chain empty-target existence equivalence)**.** *For histories a, b, a continuation morphism from a to b together with a continuation morphism from b to* e *exists exactly when both endpoints are history-same to* e*:*

$$\mathsf{Nonempty}(\mathsf{ContinuationMorphism}(a, b)) \wedge \mathsf{Nonempty}(\mathsf{ContinuationMorphism}(b, \mathsf{e})) \iff \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(b, \mathsf{e}).$$

Proof. Unpack the two displayed continuation morphisms and apply the composite empty-target inversion: the right leg identifies the middle endpoint with e, and the transported left leg identifies the source endpoint with e. Conversely, if both endpoints are empty, the empty-tail right-unit continuations build the two displayed morphisms. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism chain empty target nonempty iff`

Theorem 61.18 (Category empty-target semantic name certificate)**.** *Fix a source history a satisfying a*hsamee*. The empty-target hom-carrier* $\mathsf{CategoryHomCarrier}(a, \mathsf{e}, -)$*, used as source, pattern, and ledger policy with classifier* hsame*, forms a semantic name certificate.*

Proof. The carrier is inhabited by the empty morphism. Empty-target hom-carrier exactness reduces this witness to the assumed $a$hsamee and reflexivity of e. The classifier laws are the equivalence laws of hsame, and carrier preservation is the hom-carrier transport theorem over morphism sameness with source a and target e fixed. Pattern and ledger soundness are identities because all three policies coincide. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target semanticNameCert`

Theorem 61.19 (Category hom carrier composition closure)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*, then fg is a hom-carrier from a to c.*

Proof. Unpack the two hom-carrier witnesses. The source, middle, target, and morphism histories are unary. Unary continuation closure gives unary membership for the composite morphism fg, and associativity of the recursive append representative underlying Cont supplies the continuation from a through fg to c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp closed`

Theorem 61.20 (Category hom carrier composite-tail semantic name certificate)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*, then the predicate*

$$t \mapsto \mathsf{CategoryHomCarrier}(a, c, t) \wedge t\mathsf{hsame}fg$$

is a semantic name certificate when used as source, pattern, and ledger policy with classifier hsame*.*

Proof. Composition closure gives the displayed witness fg in the carrier, and reflexivity gives $fg\mathsf{hsame}fg$. The classifier laws are the equivalence laws of hsame. Carrier transport uses category hom-carrier transport with the source and target endpoints fixed, while the displayed-tail equation is carried by symmetry and transitivity of hsame. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp tail semanticNameCert`

Theorem 61.21 (Category hom carrier composite tail exact iff)**.** *For the unary-continuation category, if f is a hom-carrier from a to b and g is a hom-carrier from b to c, then a displayed history fg is a hom-carrier from a to c exactly when fg is the continuation tail composite of f and g:*

$$\mathsf{CategoryHomCarrier}(a, c, fg) \iff \mathsf{Cont}(f, g, fg).$$

Proof. The forward direction compares the displayed composite hom-carrier continuation $\mathsf{Cont}(a, fg, c)$ with the two factor continuations $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, c)$. Kernel composite-tail uniqueness identifies fg with $\mathsf{append}(f, g)$, which is exactly $\mathsf{Cont}(f, g, fg)$. The reverse direction is hom-carrier composition closure along that continuation witness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp tail exact iff`

Theorem 61.22 (Category hom carrier composite hsame transport)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$, then any hsame-equal source endpoint, target endpoint, and composite morphism history display the same composite as a hom-carrier from a' to c'.*

Proof. Composition closure first gives the composite hom-carrier from a to c with morphism fg. Category hom-carrier transport along the supplied source, target, and morphism hsame witnesses carries that composite to the displayed endpoints and composite history. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp hsame transport`

Theorem 61.23 (Category hom carrier composite factorization iff)**.** *For histories a, c, f, g, fg : BHist with $\mathsf{Cont}(f, g, fg)$, a displayed composite hom-carrier from a to c is equivalent to the existence of an intermediate history b carrying f from a to b and g from b to c:*

$$\mathsf{CategoryHomCarrier}(a, c, fg) \iff \exists b : \mathsf{BHist}, \mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g).$$

Proof. From a displayed composite, the canonical-middle decomposition supplies the witness $b = \mathsf{append}(a, f)$ and the two hom-carrier factors. Conversely, any displayed intermediate factorization composes by hom-carrier composition closure along the same continuation witness $\mathsf{Cont}(f, g, fg)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp factorization iff`

Theorem 61.24 (Category hom-carrier composite endpoint cycle has empty morphism)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$. If the composite endpoints satisfy $\mathsf{hsame}(a, c)$, then the composite morphism history is empty:*

$$\mathsf{hsame}(fg, \mathsf{e}).$$

Proof. Composition closure first turns the displayed continuation $\mathsf{Cont}(f, g, fg)$ into a hom-carrier from a to c. Its concrete continuation field is $\mathsf{Cont}(a, fg, c)$. Transporting the result endpoint along $\mathsf{hsame}(a, c)$ gives $\mathsf{Cont}(a, fg, a)$, and right-unit uniqueness forces fg to be e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint cycle morphism empty`

Theorem 61.25 (Category hom-carrier endpoint cycle empty morphism equivalence)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$. Then the composite endpoints close exactly when the composite morphism is empty:*

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(fg, \mathsf{e}).$$

Proof. The forward implication is the endpoint-cycle empty-morphism theorem above. Conversely, composition closure gives a hom-carrier from a to c with morphism fg. If $fg\mathsf{hsame}\mathsf{e}$, eliminate that comparison and apply the empty-identity exactness theorem to the composite hom-carrier; its endpoint projection is $\mathsf{hsame}(a, c)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint cycle morphism empty iff`

Theorem 61.26 (Category empty-source composite result)**.** *If f is a hom-carrier from e to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$, then fg is a hom-carrier from e to c and $fg\mathsf{hsame}c$.*

Proof. Composition closure first places fg as a displayed hom-carrier from e to c. Applying empty-source exactness to this composite carrier reads back the result equality $fg\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source comp result hsame`

Theorem 61.27 (Category empty-source composite target equality)**.** *If f is a hom-carrier from e to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$, then the composite result is endpoint-equal to the right target:*

$$fg\mathsf{hsame}c.$$

Proof. Empty-source exactness on the left carrier identifies f with b. Transport the displayed continuation $\mathsf{Cont}(f, g, fg)$ along this equality to obtain $\mathsf{Cont}(b, g, fg)$. The right hom-carrier supplies $\mathsf{Cont}(b, g, c)$, and continuation determinism identifies fg with c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source comp target hsame`

Theorem 61.28 (Category empty-source composite empty-result iff)**.** *If f is a hom-carrier from e to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$, then the composite result is empty exactly when the right target is empty:*

$$fg\mathsf{hsame}\mathsf{e} \iff c\mathsf{hsame}\mathsf{e}.$$

Proof. By the empty-source composite target equality, $fg\mathsf{hsame}c$. Composing this comparison with either empty-endpoint comparison gives the two implications. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source comp empty result iff`

Theorem 61.29 (Category empty-middle composite inversion)**.** *If f is a hom-carrier from a to e, g is a hom-carrier from e to c, and $\mathsf{Cont}(f, g, fg)$, then $a\mathsf{hsame}\mathsf{e}$, $f\mathsf{hsame}\mathsf{e}$, $g\mathsf{hsame}c$, fg is a hom-carrier from e to c, and $fg\mathsf{hsame}c$.*

Proof. Empty-target exactness on the left hom-carrier identifies both its source and morphism with e. The empty-source exactness on the right hom-carrier reads back $g\mathsf{hsame}c$. Composition closure gives a hom-carrier from a to c for fg; transporting its source along $a\mathsf{hsame}\mathsf{e}$ gives a hom-carrier from e to c, and empty-source exactness reads back $fg\mathsf{hsame}c$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp empty middle inversion`

Theorem 61.30 (Category empty-middle displayed morphism determinacy)**.** *For histories a, c, f, g, fg, displayed, if f is a hom-carrier from a to e, g is a hom-carrier from e to c, $\mathsf{Cont}(f, g, fg)$, and displayed is also a hom-carrier from e to c, then*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, c, \mathit{displayed}) \Longrightarrow \mathsf{hsame}(\mathit{displayed}, fg).$$

Proof. Empty-middle inversion supplies the canonical hom-carrier from e to c over the composite morphism fg. Hom-carrier morphism determinacy compares that canonical carrier with the displayed carrier over the same endpoints. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp empty middle displayed morphism deterministic`

Theorem 61.31 (Category empty-middle empty-result component readback)**.** *For histories a, c, f, g, fg, if f is a hom-carrier from a to e, g is a hom-carrier from e to c, $\mathsf{Cont}(f, g, fg)$, and $fg\mathsf{hsamee}$, then both endpoints and both factor morphisms are empty:*

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(c, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Proof. Apply category composite-result empty exactness to the two hom-carriers and the displayed continuation. The middle endpoint is already e, so the endpoint equalities read back $a\mathsf{hsamee}$ and $\mathsf{ehsame}c$; symmetry gives $c\mathsf{hsamee}$. The same exactness row supplies emptiness of both factor morphisms. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp empty middle empty result components`

Corollary 61.32 (Category empty-middle displayed morphism consumer determinacy)**.** *For histories a, c, f, g, fg, displayed, alternate : BHist, fix the empty-middle composition rows*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{e}, f), \qquad \mathsf{CategoryHomCarrier}(\mathsf{e}, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

Suppose two accepted consumer-facing uses of this same surface display hom-carrier rows over the empty source and endpoint c:

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, c, \mathit{displayed}) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{e}, c, \mathit{alternate}).$$

Then the displayed morphism choice is determined:

$$\mathsf{hsame}(\mathit{displayed}, \mathit{alternate}).$$

If either accepted use also reads an empty normal form for its displayed morphism, for instance $\mathsf{hsame}(\mathit{alternate}, \mathsf{e})$, then the exactness decision is forced to be the chapter component row

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(c, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Consequently every accepted route, normal form read, or exactness decision for the fixed empty-middle packet factors through the displayed $\mathsf{CategoryHomCarrier}$ *and* Cont *rows named above. There is no alternate hidden coordinate and no second accepted displayed hom-carrier row from which a different displayed morphism or empty-result readback could be obtained.*

Proof. Apply Definition 61.30 to the first displayed hom-carrier row. The fixed rows $\mathsf{CategoryHomCarrier}(a, \mathsf{e}, f)$, $\mathsf{CategoryHomCarrier}(\mathsf{e}, c, g)$, and $\mathsf{Cont}(f, g, fg)$ give the history-sameness comparison between *displayed* and the canonical composite morphism fg:

$$\mathsf{hsame}(\mathit{displayed}, fg).$$

Apply the same theorem to the second displayed hom-carrier row to obtain

$$\mathsf{hsame}(\mathit{alternate}, fg).$$

Symmetry and transitivity of hsame therefore give $\mathsf{hsame}(\mathit{displayed}, \mathit{alternate})$. Thus a consumer that keeps the empty-middle rows fixed has no second displayed morphism coordinate: every accepted displayed hom-carrier over e and c is the same finite route up to hsame, namely the canonical composite fg.

Now suppose an accepted use also reads $\mathsf{hsame}(\mathit{alternate}, \mathsf{e})$. Combining this empty read with $\mathsf{hsame}(\mathit{alternate}, fg)$ gives $\mathsf{hsame}(fg, \mathsf{e})$ by symmetry and transitivity. The hypotheses of Definition 61.31 are then exactly the fixed empty-middle rows together with this empty-result read. That theorem returns

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(c, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Hence the only empty-result exactness decision available to the consumer is the listed component readback. The determinacy row fixes the displayed morphism route, and the empty-result component row fixes the endpoint and factor emptiness readback; no unlisted coordinate remains from which a different normal form or accepted row could be read. □

Theorem 61.33 (Category empty-target composite boundary inversion)**.** *If f is a hom-carrier from a to b, g is a hom-carrier from b to e, and $\mathsf{Cont}(f, g, fg)$, then a, b, f, g, and fg are all hsame to e.*

Proof. Empty-target exactness applied to the right hom-carrier identifies b and g with e. With b reduced to e, empty-target exactness applied to the left hom-carrier identifies a and f with e. The displayed continuation ledger then has both inputs empty, so its result is e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target comp boundary inversion`

Theorem 61.34 (Category hom-carrier composite empty-target boundary)**.** *If f is a hom-carrier from a to b, g is a hom-carrier from b to c, $\mathsf{Cont}(f, g, fg)$, and $c\mathsf{hsamee}$, then a, b, c, f, g, and fg are all hsame to e.*

Proof. Transport the right hom-carrier along $c\mathsf{hsamee}$ so that it has empty target. Empty-target exactness applied to that right carrier identifies b and g with e. Transporting the left carrier along $b\mathsf{hsamee}$ lets empty-target exactness identify a and f with e. The displayed continuation then has both inputs empty, so its result is e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp empty target boundary`

Theorem 61.35 (Category hom carrier composition public readback)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and $\mathsf{Cont}(f, g, fg)$, then fg is a hom-carrier from a to c, and every displayed hom-carrier from a to c has morphism history hsame-equal to fg.*

Proof. Composition closure supplies the displayed carrier for fg. The hom-carrier morphism determinacy theorem then compares that carrier with any other displayed carrier over the same source and target. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp public readback`

Theorem 61.36 (Category hom-carrier composite result congruence)**.** *For unary-continuation hom carriers $f : a \to b$, $g : b \to c$, $f' : a \to b$, and $g' : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g', fg')$ hold, then the two composite morphism histories are classified the same:*

$$\mathsf{hsame}(fg, fg').$$

Proof. Composition closure places fg and fg' as hom-carriers from the same source a to the same target c. Hom-carrier morphism determinacy at those endpoints then identifies the two composite morphism histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result hsame congruence`

Theorem 61.37 (Category hom carrier same-endpoint composite result determinacy)**.** *For unary-continuation hom carriers $f : a \to b$, $g : b \to c$, $f' : a \to b'$, and $g' : b' \to c$, if $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g', fg')$ hold, then the two composite morphism histories have the same displayed source and target and hence*

$$\mathsf{hsame}(fg, fg').$$

Proof. Apply composition closure to f, g and to f', g'. The two displayed composites are hom-carriers from the same source a to the same target c. Hom-carrier morphism determinacy at those endpoints identifies their result histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp same endpoints result deterministic`

Theorem 61.38 (Category hom carrier same-composite middle-factor determinacy)**.** *For unary-continuation hom carriers $f : a \to b$, $g : b \to c$, $f' : a \to b$, and $g' : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g', fg)$ hold, then $\mathsf{hsame}(f, f') \wedge \mathsf{hsame}(g, g')$.*

Proof. First apply Definition 61.184 to the two $a \to b$ carriers. This gives $\mathsf{hsame}(f, f')$.

Transport the first continuation ledger across this equality. The transported ledger and $\mathsf{Cont}(f', g', fg)$ now have the same left continuation history and the same result history, so continuation left cancellation identifies the two right factors, giving $\mathsf{hsame}(g, g')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp same middle factors deterministic`

Theorem 61.39 (Category empty composite has empty hom factors)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if their composite ledger has empty result,*

$$\mathsf{Cont}(f, g, \mathsf{e}),$$

then both factors are internally empty and the adjacent endpoints are identified:

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Empty-result inversion for the continuation ledger first gives $\mathsf{hsame}(f, \mathsf{e})$ and $\mathsf{hsame}(g, \mathsf{e})$. Transport the two displayed hom-carrier witnesses across these morphism equalities, obtaining $\mathsf{CategoryHomCarrier}(a, b, \mathsf{e})$ and $\mathsf{CategoryHomCarrier}(b, c, \mathsf{e})$. The empty identity exactness theorem then reads back $\mathsf{hsame}(a, b)$ from the first carrier and $\mathsf{hsame}(b, c)$ from the second. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty composite factors`

Theorem 61.40 (Category empty composite exactness)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, the composite ledger has empty result exactly when both morphism tails are empty and the adjacent endpoints collapse:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The forward direction is the empty-composite factor theorem. Conversely, $\mathsf{hsame}(f, \mathsf{e})$ and $\mathsf{hsame}(g, \mathsf{e})$ identify both continuation tails with the empty history, so the empty continuation ledger is the right-unit calculation at e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty composite iff`

Theorem 61.41 (Category displayed composite result empty exactness)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if their displayed composite ledger is $\mathsf{Cont}(f, g, fg)$, then the displayed result is empty exactly when both morphism tails are empty and the adjacent endpoints collapse:*

$$\mathsf{hsame}(fg, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. In the forward direction, transport the displayed continuation ledger along $\mathsf{hsame}(fg, \mathsf{e})$ to obtain $\mathsf{Cont}(f, g, \mathsf{e})$, then apply empty-composite exactness. Conversely, the endpoint data rebuilds the empty-result composite ledger by empty-composite exactness. Continuation determinacy compares that ledger with the displayed ledger and yields $\mathsf{hsame}(fg, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result empty iff`

Theorem 61.42 (One-headed endpoint empty composite exactness)**.** *For unary-continuation hom carriers $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), b, f)$ and $\mathsf{CategoryHomCarrier}(b, \mathsf{E1}(c), g)$, the displayed empty composite is equivalent to both displayed morphisms being empty and the one-headed endpoint payloads agreeing:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, c).$$

Proof. Apply category empty-composite exactness to obtain empty morphism histories and the adjacent endpoint equalities $\mathsf{hsame}(\mathsf{E1}(a), b)$ and $\mathsf{hsame}(b, \mathsf{E1}(c))$. Transitivity identifies $\mathsf{E1}(a)$ with $\mathsf{E1}(c)$, and one-headed constructor inversion reads back $\mathsf{hsame}(a, c)$. Conversely, empty displayed morphisms reduce the continuation ledger to the empty continuation at e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 endpoint empty composite iff`

Theorem 61.43 (Category empty composite identity factors)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, the composite ledger has empty result exactly when the two corresponding identity carriers are present:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \iff \mathsf{CategoryHomCarrier}(a, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, c, \mathsf{e}).$$

Proof. An empty composite gives empty tails by continuation inversion, and hom-carrier transport moves the two displayed carriers to those empty tails. In the reverse direction, morphism determinacy compares each displayed carrier with its empty identity factor, reducing the continuation ledger to $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty composite identity factors iff`

Theorem 61.44 (Category empty composite exposes identity carriers)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if their composite ledger has empty result, then the empty morphism is a displayed identity carrier at each endpoint:*

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(c, c, \mathsf{e}).$$

Proof. The empty-composite factor theorem transports both displayed morphisms to empty hom carriers and reads back the adjacent endpoint identifications. The empty identity exactness theorem then packages the unary endpoint carriers as the three displayed empty identity carriers. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty composite identity carriers`

Theorem 61.45 (Category displayed empty composite exposes identity carriers)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and $\mathsf{hsame}(fg, \mathsf{e})$, then the empty morphism is a displayed identity carrier at each endpoint:*

$$\mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(c, c, \mathsf{e}).$$

Proof. Transport the displayed composite ledger along $\mathsf{hsame}(fg, \mathsf{e})$ to obtain an empty-result composite ledger. The empty-composite identity-carrier theorem then supplies the three displayed identity carriers. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result empty identity carriers`

Theorem 61.46 (Category displayed empty composite exposes edge and identity carriers)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and $\mathsf{hsame}(fg, \mathsf{e})$, then the empty morphism is displayed both on the two original edges and as the three endpoint identity carriers:*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, c, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(a, a, \mathsf{e}) \wedge \mathsf{CategoryHomCarrier}(b, b, \mathsf{e}) \wedge \mathsf{C}$$

Proof. Transport the displayed composite ledger along $\mathsf{hsame}(fg, \mathsf{e})$. The empty composite identity-factor theorem gives the two edge carriers, and the empty composite identity-carrier theorem gives the three endpoint identity carriers. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result empty edge identity carriers`

Theorem 61.47 (Category composite nonempty result exposes a nonempty factor)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and the displayed composite result is not internally empty, then at least one displayed morphism factor is not internally empty:*

$$\neg\mathsf{hsame}(fg, \mathsf{e}) \Rightarrow \neg\mathsf{hsame}(f, \mathsf{e}) \vee \neg\mathsf{hsame}(g, \mathsf{e}).$$

Proof. Inspect the displayed left morphism history. If it is empty, the continuation ledger identifies the composite result with the right morphism, so a right empty witness would make the composite empty. If the left morphism is zero-headed or one-headed, BEDC constructor separation directly gives the left nonempty disjunct. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty cases`

Theorem 61.48 (Category composite nonempty result exposes nonempty endpoints)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if $\mathsf{Cont}(f, g, fg)$ and the displayed composite result is not internally empty, then at least one endpoint pair is not internally equal:*

$$\neg\mathsf{hsame}(fg, \mathsf{e}) \Rightarrow \neg\mathsf{hsame}(a, b) \vee \neg\mathsf{hsame}(b, c).$$

Proof. The nonempty-factor theorem gives a nonempty displayed factor. If a and b were internally equal, transporting the left hom carrier would make f an endomorphism at a; endomorphism exactness forces that displayed morphism to be empty, contradicting the left factor case. The right factor case is the same argument at the endpoint pair b, c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty endpoint cases`

Theorem 61.49 (Category displayed composite nonempty result equivalence)**.** *For unary-continuation hom carriers $f : a \to b$ and $g : b \to c$, if $\mathsf{Cont}(f, g, fg)$, then the displayed composite result is not internally empty exactly when at least one displayed morphism factor is not internally empty:*

$$\neg\mathsf{hsame}(fg, \mathsf{e}) \iff \neg\mathsf{hsame}(f, \mathsf{e}) \vee \neg\mathsf{hsame}(g, \mathsf{e}).$$

Proof. Eliminate the displayed continuation equality, rewriting fg as the append of f and g. The append nonempty factor equivalence gives the displayed statement. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty iff`

Theorem 61.50 (Category hom carrier unary suffix equivalence)**.** *For a unary suffix q, a displayed hom-carrier $\mathsf{CategoryHomCarrier}(\mathsf{append}(a, q), \mathsf{append}(b, q), f)$ is equivalent to the pair consisting of unary evidence for q and the unsuffixed carrier $\mathsf{CategoryHomCarrier}(a, b, f)$.*

Proof. The forward direction reads unary evidence from the suffixed endpoints, swaps the unary suffix past the unary morphism, and cancels the common suffix in the continuation ledger. The reverse direction uses unary append closure and the same commutation to rebuild the suffixed continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary suffix iff`

Theorem 61.51 (Category hom carrier unary context exactness)**.** *For unary histories p and q, a displayed hom-carrier between $\mathsf{append}(\mathsf{append}(p, a), q)$ and $\mathsf{append}(\mathsf{append}(p, b), q)$ is equivalent to unary evidence for both boundary histories together with the central carrier $\mathsf{CategoryHomCarrier}(a, b, f)$.*

Proof. First peel the common unary suffix by the suffix equivalence. The remaining prefixed carrier exposes unary evidence for p, and prefix cancellation on the continuation ledger recovers the central carrier. Conversely, prefix closure rebuilds the central carrier under p, and suffix closure appends the visible suffix q. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary context iff`

Theorem 61.52 (Category hom carrier unary prefix commutation transport)**.** *For unary histories p and q, any hom-carrier displayed between $\mathsf{append}(p, \mathsf{append}(q, a))$ and $\mathsf{append}(p, \mathsf{append}(q, b))$ transports to a hom-carrier displayed between $\mathsf{append}(q, \mathsf{append}(p, a))$ and $\mathsf{append}(q, \mathsf{append}(p, b))$ with the same morphism history.*

Proof. Apply unary prefix exactness twice to reflect the displayed carrier to the central hom-carrier. Reapply the same exactness theorem first with prefix p and then with prefix q. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary prefix comm transport`

Theorem 61.53 (Category hom carrier unary context composition public readback)**.** *For composable category hom-carriers f, g, unary histories p, q, and a continuation composite* $\mathsf{Cont}(f, g, fg)$*, the morphism fg is a hom-carrier from* $\mathsf{append}(\mathsf{append}(p, a), q)$ *to* $\mathsf{append}(\mathsf{append}(p, c), q)$*. Any other displayed hom-carrier between those same contextual endpoints is* hsame*-equal to fg.*

Proof. Apply category hom-carrier composite public readback to the central composable pair. The unary context exactness theorem lifts the resulting central carrier through the left prefix p and right suffix q. For a competing displayed carrier, the same context exactness theorem reflects it back to the central endpoints, where public readback uniqueness gives hsame with the chosen composite. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary context comp public readback`

Theorem 61.54 (Category hom carrier unary suffix composite public readback)**.** *For composable category hom-carriers f, g and unary suffix q, the composite carrier over* $\mathsf{append}(a, q)$ *and* $\mathsf{append}(c, q)$ *exists, and any competing displayed composite over those suffixed endpoints is* hsame*-equal to the same composite morphism.*

Proof. Apply unsuffixed composite public readback, transport its carrier through the unary suffix equivalence, and reflect any competing suffixed display back through the same equivalence before applying morphism determinacy. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary suffix comp public readback`

Theorem 61.55 (Category hom-carrier composite endpoint cycle has empty tails)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*. If the composite endpoints satisfy* $\mathsf{hsame}(a, c)$*, then both factor morphism histories and the displayed composite morphism history are empty:*

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(fg, \mathsf{e}).$$

Proof. The endpoint-cycle morphism theorem first forces the displayed composite fg to be empty. Transport the concrete continuation $\mathsf{Cont}(f, g, fg)$ along this endpoint comparison to obtain $\mathsf{Cont}(f, g, \mathsf{e})$. Empty-result inversion for continuation then splits the factor histories, giving $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$, while the transported composite comparison supplies $fg\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint cycle tails empty`

Theorem 61.56 (Category hom-carrier composite nonempty morphism excludes empty target)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*. If the displayed composite morphism is apart from the empty history, then the target endpoint cannot be empty:*

$$(\mathsf{hsame}(fg, \mathsf{e}) \to \mathsf{False}) \to \mathsf{hsame}(c, \mathsf{e}) \to \mathsf{False}.$$

Proof. Composition closure first builds the composite hom-carrier from a to c over fg. If $c\mathsf{hsamee}$, transport this carrier to an empty target. The empty-target exactness theorem then forces $fg\mathsf{hsamee}$, contradicting the displayed nonemptiness of the composite morphism. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp morphism nonempty target not empty`

Theorem 61.57 (Category hom-carrier composite endpoint cycle boundary)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*. If the composite endpoints satisfy* $\mathsf{hsame}(a, c)$*, then both factor morphisms, the displayed composite morphism, and the two adjacent endpoint pairs collapse:*

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(fg, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The endpoint-cycle tail theorem gives $f\mathsf{hsamee}$, $g\mathsf{hsamee}$, and $fg\mathsf{hsamee}$. Transport the two hom-carrier continuation fields along the empty comparisons for f and g. Comparing the transported relations with the right-unit continuations at a and b gives $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint cycle boundary`

Theorem 61.58 (Category hom-carrier composite endpoint cycle boundary iff)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*. Then the outer endpoints close exactly when both factor morphisms, the displayed composite morphism, and the two adjacent endpoint pairs collapse:*

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(fg, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The forward implication is the composite endpoint-cycle boundary theorem. In the reverse direction, compose the adjacent endpoint comparisons $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$ by transitivity to obtain $\mathsf{hsame}(a, c)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint cycle boundary iff`

Theorem 61.59 (Category hom carrier composition right factor)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, fg is the continuation composite of f and g, and fg is displayed as a hom-carrier from a to c, then g is a hom-carrier from b to c.*

Proof. Unpack the hom-carrier witnesses. The displayed composite gives unary membership for fg, so right-factor recovery for unary continuations applied to $\mathsf{Cont}(f, g, fg)$ gives unary membership for g. The left hom-carrier gives $\mathsf{Cont}(a, f, b)$, the composite ledger gives $\mathsf{Cont}(f, g, fg)$, and the displayed carrier gives $\mathsf{Cont}(a, fg, c)$. Associativity of append turns these three ledgers into $\mathsf{Cont}(b, g, c)$, which is the right-factor hom-carrier ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right factor`

Theorem 61.60 (Category hom carrier composition right factor exactness)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, and fg is displayed as a hom-carrier from a to c. Then the proposed right factor g is a hom-carrier from b to c exactly when it composes with f to the displayed morphism:*

$$\mathsf{CategoryHomCarrier}(b, c, g) \iff \mathsf{Cont}(f, g, fg).$$

Proof. The forward implication compares the two endpoint continuations $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, c)$ with the displayed continuation $\mathsf{Cont}(a, fg, c)$. Composite-tail uniqueness cancels the common source and recovers $\mathsf{Cont}(f, g, fg)$. The reverse implication is the right-factor recovery theorem applied to the displayed composite. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right factor iff`

Theorem 61.61 (Category hom carrier canonical middle factorization)**.** *For the unary-continuation category, if* $\mathsf{Cont}(f, g, fg)$*, then a displayed composite hom-carrier from* a *to* c *over* fg *is equivalent to the pair of hom-carriers obtained by using the canonical middle object* $\mathsf{append}(a, f)$*:*

$$\mathsf{CategoryHomCarrier}(a, c, fg) \iff \mathsf{CategoryHomCarrier}(a, \mathsf{append}(a, f), f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(a, f), c, g).$$

Proof. The forward direction reads the unary left factor f from $\mathsf{Cont}(f, g, fg)$ and the displayed unary composite history. This constructs the canonical left hom-carrier $a \to \mathsf{append}(a, f)$ by the defining continuation $\mathsf{Cont}(a, f, \mathsf{append}(a, f))$. The existing right-factor recovery theorem then extracts the hom-carrier from $\mathsf{append}(a, f)$ to c over g. Conversely, composition closure for hom-carriers recombines the two canonical factors along $\mathsf{Cont}(f, g, fg)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp canonical middle iff`

Theorem 61.62 (Category hom carrier canonical middle public readback)**.** *For the unary-continuation category, if* $\mathsf{Cont}(f, g, fg)$ *and* fg *is displayed as a hom-carrier from* a *to* c*, then the canonical middle object* $\mathsf{append}(a, f)$ *displays the two hom-carrier factors*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{append}(a, f), f) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{append}(a, f), c, g).$$

Moreover, any other middle object b *displaying hom-carriers* $a \to b$ *over* f *and* $b \to c$ *over* g *satisfies* $\mathsf{hsame}(\mathsf{append}(a, f), b)$*.*

Proof. The canonical-middle equivalence supplies the two displayed factors from the composite hom-carrier. Middle-object determinacy then compares these canonical factors with any displayed factorization through b, using the same continuation composite $\mathsf{Cont}(f, g, fg)$ and the same displayed composite hom-carrier from a to c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp canonical middle public readback`

Theorem 61.63 (Category hom carrier composition right factor public readback)**.** *For the unary-continuation category, if* f *is a hom-carrier from* a *to* b*,* fg *is the continuation composite of* f *and* g*, and* fg *is displayed as a hom-carrier from* a *to* c*, then* g *is a hom-carrier from* b *to* c*. Moreover, any competing right factor* g' *with* $\mathsf{Cont}(f, g', fg)$ *and the same hom-carrier endpoints satisfies* $\mathsf{hsame}(g, g')$*.*

Proof. The right-factor theorem recovers the hom-carrier for g from the displayed composite. For any competing g', the two continuation ledgers $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$ share the same left factor and displayed result, so finite-kernel left cancellation yields $\mathsf{hsame}(g, g')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right factor public readback`

Theorem 61.64 (Category hom carrier right unary tail composition public readback)**.** *For the unary-continuation category, if* f *is a hom-carrier from* a *to* b*,* q *is unary, and* $\mathsf{Cont}(f, q, r)$*, then* r *is a hom-carrier from* a *to* $\mathsf{append}(b, q)$*. Moreover, any competing result* r' *with* $\mathsf{Cont}(f, q, r')$ *and the same displayed endpoints satisfies* $\mathsf{hsame}(r, r')$*.*

Proof. The unary tail q gives the canonical right-tail hom-carrier from b to $\mathsf{append}(b, q)$ by the defining continuation. Hom-carrier composition closes f with this tail carrier along $\mathsf{Cont}(f, q, r)$. The readback clause is hom-morphism determinacy against any displayed carrier over the same endpoints. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier right unary tail comp public readback`

Theorem 61.65 (Category hom carrier composition left factor)**.** *For the unary-continuation category, if g is a hom-carrier from b to c, fg is the continuation composite of f and g, and fg is displayed as a hom-carrier from a to c, then f is a hom-carrier from a to b.*

Proof. Unpack the displayed composite hom-carrier. Unary left-factor recovery applied to $\mathsf{Cont}(f, g, fg)$ and the unary composite morphism gives unary membership for f. The right hom-carrier gives $\mathsf{Cont}(b, g, c)$, while the displayed composite gives $\mathsf{Cont}(a, fg, c)$; the finite-kernel composite left-factor theorem recovers $\mathsf{Cont}(a, f, b)$. These fields assemble the hom-carrier from a to b. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left factor`

Theorem 61.66 (Category hom carrier composition left factor exactness)**.** *For the unary-continuation category, suppose g is a hom-carrier from b to c, and fg is displayed as a hom-carrier from a to c. Then the proposed left factor f is a hom-carrier from a to b exactly when it composes with g to the displayed morphism:*

$$\mathsf{CategoryHomCarrier}(a, b, f) \iff \mathsf{Cont}(f, g, fg).$$

Proof. The forward implication compares $\mathsf{Cont}(a, f, b)$ and $\mathsf{Cont}(b, g, c)$ with the displayed continuation $\mathsf{Cont}(a, fg, c)$. Composite-tail uniqueness recovers the tail equation $\mathsf{Cont}(f, g, fg)$. The reverse implication is the left-factor recovery theorem applied to the displayed composite. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left factor iff`

Theorem 61.67 (Category hom carrier composition left factor public readback)**.** *For the unary-continuation category, if g is a hom-carrier from b to c, fg is the continuation composite of f and g, and fg is displayed as a hom-carrier from a to c, then f is a hom-carrier from a to b. Moreover, any competing left factor f' with $\mathsf{Cont}(f', g, fg)$ and the same hom-carrier endpoints satisfies $\mathsf{hsame}(f, f')$.*

Proof. The left-factor theorem recovers the hom-carrier for f from the displayed composite. For any competing f', the two continuation ledgers share the same right factor and composite result, so finite-kernel right cancellation yields $\mathsf{hsame}(f, f')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left factor public readback`

Theorem 61.68 (Category hom carrier same-left-tail factorization determinacy)**.** *For the unary-continuation category, suppose f is displayed as a hom-carrier from a to b and also from a to b'. If g and g' are right factors from those middle endpoints to c and c' respectively, and both $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$ display the same composite history fg, then the middle endpoints, right-factor histories, and target endpoints are determined:*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(g, g') \wedge \mathsf{hsame}(c, c').$$

Proof. Target determinacy for the two fixed-left-tail hom-carriers identifies the middle endpoints. The two continuation ledgers share the same left tail f and displayed result fg, so left cancellation identifies g and g'. Transport the right hom-carrier across these two comparisons; target determinacy against the displayed $b' \to c'$ carrier then identifies c and c'. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp same left tail factorization deterministic`

Theorem 61.69 (Category hom carrier same-right-tail factorization determinacy)**.** *For the unary-continuation category, suppose two composite ledgers share the same right tail g and the same displayed composite fg:*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg),$$

and

$$\mathsf{CategoryHomCarrier}(a', b', f'), \qquad \mathsf{CategoryHomCarrier}(b', c, g), \qquad \mathsf{Cont}(f', g, fg).$$

Then the middle endpoints, left-factor histories, and source endpoints are determined:

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(f, f') \wedge \mathsf{hsame}(a, a').$$

Proof. Source determinacy for the two shared-right-tail hom-carriers identifies the middle endpoints b and b'. The two composite continuation ledgers share the same right tail g and displayed result fg, so right cancellation identifies the left factors f and f'. Transporting the second left hom-carrier across these two comparisons gives a hom-carrier from a' to b with morphism f; source determinacy against the first left hom-carrier then identifies a and a'. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp same right tail factorization deterministic`

Theorem 61.70 (Continuation morphism same-right-tail factorization determinacy)**.** *For continuation morphisms $left$: $\mathsf{ContinuationMorphism}(a, b)$, $right$: $\mathsf{ContinuationMorphism}(b, c)$, $left'$: $\mathsf{ContinuationMorphism}(a', b')$, and $right'$: $\mathsf{ContinuationMorphism}(b', c)$, suppose the two right tails are history-same and the two composite tails are history-same. Then the two canonical tail continuations exist, and the source endpoints, middle endpoints, and left tails are determined:*

$$\mathsf{Cont}(left.tail, right.tail, (left \circ right).tail) \wedge \mathsf{Cont}(left'.tail, right'.tail, (left' \circ right').tail) \wedge \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b$$

Proof. The two closed composites are continuation morphisms into the same target c, so source determinacy applied to the displayed composite-tail comparison identifies a and a'. Applying the same source determinacy to the two right tails identifies b and b'. Transport the left continuation across these endpoint comparisons and cancel against $left'$ to identify the left tails. The canonical tail-continuation witnesses are the append displays inside the closed composites. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp same right tail factorization deterministic`

Theorem 61.71 (Continuation morphism parallel-factor composite middle determinacy)**.** *For continuation morphisms $left$: $\mathsf{ContinuationMorphism}(a, b)$, $right$: $\mathsf{ContinuationMorphism}(b, c)$, $left'$: $\mathsf{ContinuationMorphism}(a, b')$, and $right'$: $\mathsf{ContinuationMorphism}(b', c)$, suppose the two left tails and the two right tails are respectively history-same. Then the four displayed continuation fields are present, the two middle endpoints are history-same, and the two closed composite tails are history-same:*

$$\mathsf{Cont}(a, left.tail, b) \wedge \mathsf{Cont}(b, right.tail, c) \wedge \mathsf{Cont}(a, left'.tail, b') \wedge \mathsf{Cont}(b', right'.tail, c),$$
$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}((left \circ right).tail, (left' \circ right').tail).$$

Proof. The four continuation fields are exactly the relation fields of the four records. Target determinacy compares the two left morphisms after identifying their tails, giving $\mathsf{hsame}(b, b')$. The closed composite tails are append products of the displayed left and right tails, so the two supplied tail comparisons identify the two composite tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism parallel factor comp tail middle deterministic`

Theorem 61.72 (Continuation parallel factor tails deterministic)**.** *For kernel continuations, if two factorizations share the same source, middle, and target endpoints,*

$$\mathsf{Cont}(a, f, b) \wedge \mathsf{Cont}(b, g, c) \wedge \mathsf{Cont}(a, f', b) \wedge \mathsf{Cont}(b, g', c),$$

then the two left factor tails and the two right factor tails are determined:

$$\mathsf{hsame}(f, f') \wedge \mathsf{hsame}(g, g').$$

Proof. The first and third continuations have the same source endpoint and result endpoint, so left cancellation identifies f and f'. The second and fourth continuations have the same middle endpoint and target endpoint, so the same cancellation identifies g and g'. □

Lean4 ✓ `BEDC.FKernel.Cont.cont parallel factor tails deterministic`

Theorem 61.73 (Continuation morphism same-left-tail factorization determinacy)**.** *For continuation morphisms* $left : \mathsf{ContinuationMorphism}(a, b)$*,* $right : \mathsf{ContinuationMorphism}(b, c)$*,* $left' : \mathsf{ContinuationMorphism}(a, b')$*, and* $right' : \mathsf{ContinuationMorphism}(b', c')$*, suppose the two left tails are history-same and the two composite append tails are history-same. Then the middle endpoints, right tails, and target endpoints are determined:*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(right.tail, right'.tail) \wedge \mathsf{hsame}(c, c').$$

Proof. Target determinacy for the two fixed-left continuation morphisms identifies the middle endpoints b and b'. After transporting the left-tail comparison into the composite-tail equality, append-left cancellation identifies the right tails. Transport the right continuation along the middle-endpoint and right-tail comparisons, then compare it with $right'$ by continuation result determinacy to identify c and c'. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp same left tail factorization deterministic`

Corollary 61.74 (Category hom-carrier endpoint-cycle empty determinacy)**.** *For the unary-continuation category, suppose* $\mathsf{Cont}(f, g, fg)$ *and suppose* fg *is displayed as a hom-carrier from* a *to* c*. Let* m *denote the displayed canonical middle history obtained as the append of* a *and* f*. If the composite endpoints close,* $\mathsf{hsame}(a, c)$*, then the consumer-facing use of this endpoint cycle is determined by the finite empty-tail packet. Concretely, for any proposed middle object* b' *and right factor* g' *with* f *displayed as a hom-carrier from* a *to* b'*, the right-factor admission decision is exactly*

$$\mathsf{CategoryHomCarrier}(b', c, g') \iff \mathsf{Cont}(f, g', fg).$$

When these equivalent conditions are satisfied, the proposed factorization has the same canonical middle coordinate and the same empty normal form:

$$\mathsf{hsame}(b', m) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(g', \mathsf{e}) \wedge \mathsf{hsame}(fg, \mathsf{e}).$$

Thus no accepted public read supplies an alternate hidden coordinate, duplicate right-factor classifier source, or nonempty composite normal form outside the rows named in the chapter.

Proof. Apply Definition 61.61 to the displayed composite carrier and the continuation row $\mathsf{Cont}(f, g, fg)$. With m as the displayed canonical append history, this reads the composite through m and supplies the two canonical factors

$$\mathsf{CategoryHomCarrier}(a, m, f) \quad \text{and} \quad \mathsf{CategoryHomCarrier}(m, c, g).$$

The endpoint-cycle hypothesis $\mathsf{hsame}(a, c)$ now puts these factors under Definition 61.55, so $\mathsf{hsame}(f, \mathsf{e})$, $\mathsf{hsame}(g, \mathsf{e})$, and $\mathsf{hsame}(fg, \mathsf{e})$ are forced.

Fix a proposed middle object b' and right factor g' with $\mathsf{CategoryHomCarrier}(a, b', f)$. The displayed composite carrier from a to c and Definition 61.60 give exactly the advertised right-factor decision $\mathsf{CategoryHomCarrier}(b', c, g') \iff \mathsf{Cont}(f, g', fg)$. Hence an accepted presentation has both the carrier row for g' and the continuation row $\mathsf{Cont}(f, g', fg)$.

Compare this accepted presentation with the canonical one. The two continuation rows $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$ share the left factor f and the composite result fg, so finite-kernel left cancellation, as used in Definition 61.63, supplies $\mathsf{hsame}(g, g')$. Apply Definition 61.93 to the accepted carrier $b' \to c$ across this morphism comparison, holding the target endpoint fixed, and compare the transported carrier with the canonical carrier $m \to c$ by Definition 61.192. This gives $\mathsf{hsame}(b', m)$. Finally, transitivity with the already forced comparison $\mathsf{hsame}(g, \mathsf{e})$ gives $\mathsf{hsame}(g', \mathsf{e})$. The only remaining composite normal form is therefore the empty one already read from the endpoint cycle. □

Theorem 61.75 (Category hom carrier composition right-factor endpoint determinacy)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, fg is displayed both as $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$, the composite fg is a hom-carrier from a to c, and g' is a hom-carrier from b' to c. Then the recovered and displayed right-factor endpoints and right-factor histories agree:*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(g, g').$$

Proof. Right-factor recovery turns the displayed composite and $\mathsf{Cont}(f, g, fg)$ into a concrete hom-carrier from b to c for g. Left cancellation of $\mathsf{Cont}(f, g, fg)$ against $\mathsf{Cont}(f, g', fg)$ gives $\mathsf{hsame}(g, g')$. Transporting the recovered hom-carrier across this morphism sameness gives a hom-carrier from b to c for g', and source determinacy against the displayed $b' \to c$ carrier gives $\mathsf{hsame}(b, b')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right factor endpoint deterministic`

Theorem 61.76 (Category hom carrier composition right-factor target determinacy)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, fg is displayed both as $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$, the composite fg is a hom-carrier from a to c, and g' is a hom-carrier from b to c'. Then the recovered and displayed right-factor targets and right-factor histories agree:*

$$\mathsf{hsame}(c, c') \wedge \mathsf{hsame}(g, g').$$

Proof. Right-factor recovery turns the displayed composite and $\mathsf{Cont}(f, g, fg)$ into a concrete hom-carrier from b to c for g. Left cancellation of $\mathsf{Cont}(f, g, fg)$ against $\mathsf{Cont}(f, g', fg)$ gives $\mathsf{hsame}(g, g')$. Transporting the recovered hom-carrier across this morphism sameness gives a hom-carrier from b to c for g', and target determinacy against the displayed $b \to c'$ carrier gives $\mathsf{hsame}(c, c')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right factor target deterministic`

Theorem 61.77 (Category hom carrier composition left-factor endpoint determinacy)**.** *For the unary-continuation category, suppose g is a hom-carrier from b to c, fg is displayed both as $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g, fg)$, the composite fg is a hom-carrier from a to c, and f' is a hom-carrier from a to b'. Then the recovered and displayed left-factor endpoints and left-factor histories agree:*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(f, f').$$

Proof. Left-factor recovery turns the displayed composite and $\mathsf{Cont}(f, g, fg)$ into a concrete hom-carrier from a to b for f. Right cancellation of $\mathsf{Cont}(f, g, fg)$ against $\mathsf{Cont}(f', g, fg)$ gives $\mathsf{hsame}(f, f')$. Transporting the recovered hom-carrier across this morphism sameness gives a hom-carrier from a to b for f', and target determinacy against the displayed $a \to b'$ carrier gives $\mathsf{hsame}(b, b')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left factor endpoint deterministic`

Theorem 61.78 (Category hom carrier composition left-factor source determinacy)**.** *For the unary-continuation category, suppose g is a hom-carrier from b to c, fg is displayed both as $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g, fg)$, the composite fg is a hom-carrier from a to c, and f' is a hom-carrier from a' to b. Then the recovered and displayed left-factor sources and left-factor histories agree:*

$$\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(f, f').$$

Proof. Left-factor recovery turns the displayed composite and $\mathsf{Cont}(f, g, fg)$ into a concrete hom-carrier from a to b for f. Right cancellation of $\mathsf{Cont}(f, g, fg)$ against $\mathsf{Cont}(f', g, fg)$ gives $\mathsf{hsame}(f, f')$. Transporting the recovered hom-carrier across this morphism sameness gives a hom-carrier from a to b for f', and source determinacy against the displayed $a' \to b$ carrier gives $\mathsf{hsame}(a, a')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left factor source deterministic`

Theorem 61.79 (Category hom carrier composite target determinacy)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and fg is their continuation composite, then any displayed hom-carrier from a to d with the same composite history forces $\mathsf{hsame}(c, d)$.*

Proof. Hom-carrier composition closure makes fg a carrier from a to c. Target determinacy for the fixed morphism history fg compares this carrier with the displayed carrier from a to d. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp target deterministic`

Theorem 61.80 (Category hom carrier composite endpoint iff)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and fg is their continuation composite, then displaying the same composite as a hom-carrier from a to d is equivalent to the endpoint classifier $\mathsf{hsame}(c, d)$.*

Proof. The forward direction is composite target determinacy. Conversely, composition closure first builds the canonical hom-carrier from a to c over fg; transporting its target endpoint along $\mathsf{hsame}(c, d)$ gives the displayed hom-carrier from a to d. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp endpoint iff`

Theorem 61.81 (Category hom carrier tail-comm target determinacy)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, fg and gf are the two displayed tail continuation composites, and gf is also displayed as a hom-carrier from a to d, then $\mathsf{hsame}(c, d)$.*

Proof. Composition closure first makes fg a hom-carrier from a to c. Tail commutativity identifies fg and gf, and transport gives a hom-carrier from a to c over the reverse-tail history gf. Target determinacy then compares that transported carrier with the displayed $a \to d$ carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm target deterministic`

Theorem 61.82 (Category hom carrier tail-comm source determinacy)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, fg and gf are the two displayed tail continuation composites, and gf is also displayed as a hom-carrier from d to c, then* $\mathsf{hsame}(a, d)$*.*

Proof. Tail-comm closure transports the forward composite carrier to a carrier from a to c over the reverse-tail history gf. Source determinacy compares this carrier with the displayed $d \to c$ carrier over the same morphism history. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm source deterministic`

Theorem 61.83 (Category hom carrier composite source determinacy)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and fg is their continuation composite, then any displayed hom-carrier from d to c with the same composite history forces* $\mathsf{hsame}(a, d)$*.*

Proof. Hom-carrier composition closure makes fg a carrier from a to c. Source determinacy for the fixed morphism history fg compares this carrier with the displayed carrier from d to c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp source deterministic`

Theorem 61.84 (Category hom carrier composite middle-object determinacy)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b' to c,* $\mathsf{Cont}(f, g, fg)$*, and fg is displayed as a hom-carrier from a to c, then* $\mathsf{hsame}(b, b')$*.*

Proof. Unpack the hom-carrier witnesses. The left leg gives $\mathsf{Cont}(a, f, b)$, the composite gives $\mathsf{Cont}(f, g, fg)$, and the displayed composite gives $\mathsf{Cont}(a, fg, c)$. Associativity of append turns these three continuation facts into $\mathsf{Cont}(b, g, c)$. Comparing this continuation with the right leg $\mathsf{Cont}(b', g, c)$ by right-context cancellation identifies the two middle objects. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp middle object deterministic`

Theorem 61.85 (Category hom carrier middle-hsame composition closure)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b' to c,* $\mathsf{hsame}(b, b')$*, and* $\mathsf{Cont}(f, g, fg)$*, then fg is a hom-carrier from a to c.*

Proof. Transport the right hom-carrier source from b' back to b using symmetry of the middle-object hsame witness, while keeping the target c and morphism g fixed by reflexivity. The ordinary hom-carrier composition closure then applies to the aligned pair of factors. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp middle hsame closed`

Theorem 61.86 (Category hom carrier endpoint-hsame middle-object determinacy)**.** *For the unary-continuation category, if the displayed composite endpoint is recorded at endpoints a' and c' with* $\mathsf{hsame}(a, a')$ *and* $\mathsf{hsame}(c, c')$*, then the middle object is still determined. Concretely, if f is a hom-carrier from a to b, g is a hom-carrier from b' to c,* $\mathsf{Cont}(f, g, fg)$*, and fg is displayed as a hom-carrier from a' to c', then* $\mathsf{hsame}(b, b')$*.*

Proof. Case analysis on the endpoint hsame witnesses reduces the displayed composite to the same endpoints a and c. The ordinary composite middle-object determinacy theorem then compares the induced continuation from b through g with the displayed right leg from b' through g. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp middle object endpoint hsame deterministic`

Theorem 61.87 (Category hom carrier unary-context middle-object determinacy)**.** *For the unary-continuation category, suppose the two displayed hom-carrier ledgers have a shared left prefix p and a shared right suffix q:*

$$\mathsf{CategoryHomCarrier}((p \cdot a) \cdot q, (p \cdot b) \cdot q, f)$$

and

$$\mathsf{CategoryHomCarrier}((p \cdot b') \cdot q, (p \cdot c) \cdot q, g).$$

If $\mathsf{Cont}(f, g, fg)$ *and* fg *is displayed as a hom-carrier from* $(p \cdot a) \cdot q$ *to* $(p \cdot c) \cdot q$*, then* $\mathsf{hsame}(b, b')$*.*

Proof. Apply the unary suffix equivalence of Definition 61.50 to remove the shared right suffix from the two input ledgers and the displayed composite ledger. Each remaining ledger has the same left prefix p on its source and target endpoints; unary append factorization gives the base endpoint carriers, and prefix cancellation removes p from the continuation evidence. The base hom-carriers now satisfy the hypotheses of Definition 61.84, which identifies the middle objects by $\mathsf{hsame}(b, b')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary context comp middle object deterministic`

Theorem 61.88 (Category hom carrier parallel-factor composite middle determinacy)**.** *For the unary-continuation category, suppose the two left hom-carriers share source endpoint a and have history-same morphisms f and f', while the two right hom-carriers share target endpoint c and have history-same morphisms g and g':*

$$\mathsf{CategoryHomCarrier}(a, b, f), \quad \mathsf{CategoryHomCarrier}(b, c, g), \quad \mathsf{CategoryHomCarrier}(a, b', f'), \quad \mathsf{CategoryHomCarrier}(b'$$

Then the four underlying continuation fields are present, the middle endpoints are history-same, and the two append composite tails are history-same:

$$\mathsf{Cont}(a, f, b) \wedge \mathsf{Cont}(b, g, c) \wedge \mathsf{Cont}(a, f', b') \wedge \mathsf{Cont}(b', g', c) \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(\mathsf{append}(f, g), \mathsf{append}(f', g')).$$

Proof. Unpack the four hom-carrier records to read their continuation fields. The comparison $\mathsf{hsame}(f, f')$ transports the left morphism history, so target determinacy of the two left hom-carriers identifies b and b'. The two morphism-history comparisons identify the append products $\mathsf{append}(f, g)$ and $\mathsf{append}(f', g')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier parallel factor comp tail middle deterministic`

Theorem 61.89 (Category hom carrier tail commutativity)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c,* $\mathsf{Cont}(f, g, fg)$*, and* $\mathsf{Cont}(g, f, gf)$*, then* $\mathsf{hsame}(fg, gf)$*.*

Proof. Unpack the two hom-carrier witnesses to obtain unary membership for f and g. Unary continuation commutativity applied to the two continuation ledgers identifies the two tail composites. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm hsame`

Theorem 61.90 (Category hom carrier tail-comm history-same congruence)**.** *For the unary-continuation category, suppose the source, middle, target, and two morphism histories are transported by* hsame*. If the original histories carry hom witnesses $f : a \to b$ and $g : b \to c$, while the transported morphisms form displayed tails* $\mathsf{Cont}(f', g', fg)$ *and* $\mathsf{Cont}(g', f', gf)$*, then* $\mathsf{hsame}(fg, gf)$*.*

Proof. Transport the two hom-carrier witnesses along the endpoint and morphism history-same data. The transported carriers expose unary membership for f' and g'. Applying tail commutativity to these transported carriers and the two displayed continuation tails identifies fg and gf. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm hsame congr`

Theorem 61.91 (Category hom carrier tail-comm closure)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and fg and gf are the two displayed tail continuation composites, then both histories are hom-carriers from a to c and* $\mathsf{hsame}(fg, gf)$*.*

Proof. Composition closure gives the forward carrier $fg : a \to c$. Tail commutativity identifies fg with gf, and history-same transport moves the forward carrier to the reverse-tail history while keeping the source and target endpoints fixed. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm closed`

Theorem 61.92 (Category hom carrier tail-comm continuation readback)**.** *For the unary-continuation category, suppose f is a hom-carrier from a to b, g is a hom-carrier from b to c, and fg and gf are the two displayed tail continuation composites. Then every displayed hom-carrier $z : a \to c$ recovers both continuation ledgers:*

$$\mathsf{Cont}(f, g, z) \wedge \mathsf{Cont}(g, f, z).$$

Proof. Tail-comm public readback compares the displayed carrier $z : a \to c$ with the two closed tail carriers. From the forward closed carrier it gives $\mathsf{hsame}(fg, z)$, and from the reverse closed carrier it gives $\mathsf{hsame}(gf, z)$.

The composite-tail exactness theorem for hom-carriers then reads the forward ledger from the displayed carrier itself: since $f : a \to b$, $g : b \to c$, and $z : a \to c$ is displayed, the exact characterization converts that carrier into $\mathsf{Cont}(f, g, z)$.

For the reverse order, transport the given continuation ledger $\mathsf{Cont}(g, f, gf)$ along $\mathsf{hsame}(gf, z)$ using finite-kernel history-same transport for continuation results. This yields $\mathsf{Cont}(g, f, z)$, and the two recovered ledgers form the required conjunction. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier tail comm cont readback`

Theorem 61.93 (Category hom carrier history-same transport)**.** *For the unary-continuation category, if source objects, target objects, and morphism histories are each transported by* hsame*, then a hom-carrier witness transports to the transported triple.*

Proof. Eliminate the three hsame endpoint and morphism equalities. The target hom-carrier statement is then the original concrete CategoryHomCarrier witness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier hsame transport`

Theorem 61.94 (Category hom carrier endpoint transport with classified morphism)**.** *For the unary-continuation category, if source endpoint, target endpoint, and morphism histories are transported by* hsame*, then the transported morphism has the transported continuation field and hom-carrier witness. Moreover, any displayed hom-carrier over the transported endpoints has morphism history classified with the transported morphism.*

Proof. Apply hom-carrier history-same transport to the displayed carrier. Its continuation field gives the transported Cont witness. The transported carrier and the competing displayed carrier share source and target endpoints, so morphism determinacy identifies their morphism histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier endpoint transport cont classified`

Theorem 61.95 (Category hom carrier identity square closure)**.** *For the unary-continuation category, if f is a hom-carrier from a to b, $left$ is obtained by composing the empty continuation before f, and $right$ is obtained by composing the empty continuation after f, then $left$ and $right$ are both hom-carriers from a to b, and* $\mathsf{hsame}(left, right)$*.*

Proof. The left unit continuation identifies $left$ with f. Determinism of Cont, compared against the right unit continuation for f, identifies $right$ with f. Transporting the hom-carrier witness for f across these two history equalities gives the two carrier witnesses, and transitivity of hsame identifies $left$ and $right$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier identity square closed`

Corollary 61.96 (Category hom carrier composition right-factor endpoint consumer determinacy)**.** *For the unary-continuation category, fix a displayed composite surface with a left hom-carrier $f : a \to b$, a right hom-carrier $g : b \to c$, a composite hom-carrier $fg : a \to c$, and the continuation row* $\mathsf{Cont}(f, g, fg)$*. Suppose a consumer-facing use of the same composite also displays the right-factor row* $\mathsf{Cont}(f, g', fg)$ *with a hom-carrier $g' : b' \to c$, and the left-factor row* $\mathsf{Cont}(f', g, fg)$ *with a hom-carrier $f' : a' \to b$. Then the consumer cannot choose an alternate endpoint source, alternate right-factor endpoint, alternate left-factor history, or alternate right-factor history:*

$$\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(b, b') \wedge \mathsf{hsame}(f, f') \wedge \mathsf{hsame}(g, g').$$

Consequently every accepted route, normal-form read, or exactness decision for this displayed composition factors through the listed CategoryHomCarrier *and* Cont *rows; there is no hidden coordinate and no duplicate classifier source for the right-factor endpoint.*

Proof. Apply Definition 61.75 to the left hom-carrier $f : a \to b$, the common displayed composite $fg : a \to c$, the rows $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f, g', fg)$, and the displayed right-factor carrier $g' : b' \to c$. This gives

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(g, g').$$

Apply Definition 61.78 to the right hom-carrier $g : b \to c$, the same displayed composite $fg : a \to c$, the rows $\mathsf{Cont}(f, g, fg)$ and $\mathsf{Cont}(f', g, fg)$, and the displayed left-factor carrier $f' : a' \to b$. This gives

$$\mathsf{hsame}(a, a') \wedge \mathsf{hsame}(f, f').$$

Combining these two conjunctions identifies exactly the source endpoint, the middle endpoint, and the two factor histories used by the consumer-facing factorization.

The cited theorems already perform the only permitted readbacks: right-factor recovery from the composite carrier, left-factor recovery from the composite carrier, cancellation of the displayed continuation tails, history-same transport of the recovered carrier, and endpoint determinacy against the consumer-displayed carrier. A route, normal-form read, or exactness decision that keeps the listed rows fixed but chooses a different endpoint or factor history would contradict one of these two determinacy conclusions. Hence the consumer surface is exhausted by the displayed hom-carrier and continuation rows, with no further coordinate from which an alternate classifier source could be read. □

Theorem 61.97 (Category hom carrier right-append fixed identity)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*, then for every history* $g : \mathsf{BHist}$*,*

$$\mathsf{hsame}(\mathsf{append}(f, g), g) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Proof. In the forward direction, the displayed fixed append equality is a continuation witness $\mathsf{Cont}(f, g, g)$, so left-unit uniqueness forces $\mathsf{hsame}(f, \mathsf{e})$. Transporting the hom-carrier continuation relation along this empty-tail equality gives $\mathsf{Cont}(a, \mathsf{e}, b)$, and comparison with the canonical right-unit continuation identifies a and b. Conversely, if $f\mathsf{hsame}\mathsf{e}$, the equality reduces to the kernel empty-left append law. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier right append fixed identity iff`

Theorem 61.98 (Category hom carrier left-append fixed identity)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*, then for every history* $g : \mathsf{BHist}$*,*

$$\mathsf{hsame}(\mathsf{append}(g, f), g) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Proof. In the forward direction, compare $\mathsf{append}(g, f)$ with $\mathsf{append}(g, \mathsf{e})$ and cancel the common left prefix, obtaining $\mathsf{hsame}(f, \mathsf{e})$. Transporting the hom-carrier continuation relation across this empty-tail equality gives $\mathsf{Cont}(a, \mathsf{e}, b)$, hence $\mathsf{hsame}(a, b)$ by comparison with the canonical right-unit continuation. Conversely, if $f\mathsf{hsame}\mathsf{e}$, the equality reduces to the kernel empty-right append law. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier left append fixed identity iff`

Theorem 61.99 (Category hom carrier composite left-tail fixed right identity)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, c, g)$*,* $\mathsf{Cont}(f, g, fg)$*, and* $\mathsf{hsame}(fg, f)$*, then*

$$\mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(b, c).$$

Proof. Transport the displayed composite continuation along $\mathsf{hsame}(fg, f)$ to obtain $\mathsf{Cont}(f, g, f)$. Right-unit uniqueness for continuations forces $\mathsf{hsame}(g, \mathsf{e})$. Transport the right hom-carrier continuation $\mathsf{Cont}(b, g, c)$ across this empty-tail comparison to obtain $\mathsf{Cont}(b, \mathsf{e}, c)$, then compare it with the canonical right-unit continuation at b. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left tail fixed right identity`

Theorem 61.100 (Category hom carrier composite right-tail fixed left identity)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, c, g)$*,* $\mathsf{Cont}(f, g, fg)$*, and* $\mathsf{hsame}(fg, g)$*, then*

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Proof. Transport the displayed composite continuation along $\mathsf{hsame}(fg, g)$ to obtain $\mathsf{Cont}(f, g, g)$. Left-unit uniqueness for continuations forces $\mathsf{hsame}(f, \mathsf{e})$. Transport the left hom-carrier continuation $\mathsf{Cont}(a, f, b)$ across this empty-tail comparison to obtain $\mathsf{Cont}(a, \mathsf{e}, b)$, then compare it with the canonical right-unit continuation at a. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp right tail fixed left identity`

Theorem 61.101 (Category hom carrier composite result fixed by both factors)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, c, g)$*, and* $\mathsf{Cont}(f, g, fg)$*, then*

$$\mathsf{hsame}(fg, f) \wedge \mathsf{hsame}(fg, g) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The forward direction applies the one-sided fixed-composite identity laws to the two displayed comparisons. Fixing the result against f forces the right factor tail empty and identifies b with c; fixing it against g forces the left factor tail empty and identifies a with b. Conversely, when both factor tails are empty, the composite continuation reduces to the empty history, hence it is history-same to each factor tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result fixed both identity iff`

Theorem 61.102 (Category hom carrier two-sided identity law)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{Cont}(\mathsf{e}, f, left)$*, and* $\mathsf{Cont}(f, \mathsf{e}, right)$*, then*

$$\mathsf{CategoryHomCarrier}(a, b, left) \wedge \mathsf{CategoryHomCarrier}(a, b, right) \wedge \mathsf{hsame}(left, f) \wedge \mathsf{hsame}(right, f) \wedge \mathsf{hsame}(left, right)$$

Proof. The left composite is classified against the original morphism by the left-unit result theorem: $\mathsf{Cont}(\mathsf{e}, f, left)$ and `cont_left_unit_result` give $\mathsf{hsame}(left, f)$.

For the right composite, compare $\mathsf{Cont}(f, \mathsf{e}, right)$ with the canonical right-unit continuation $\mathsf{Cont}(f, \mathsf{e}, f)$ supplied by `cont_right_unit`. Continuation determinism gives $\mathsf{hsame}(right, f)$.

Transport the displayed $\mathsf{CategoryHomCarrier}(a, b, f)$ witness across the two history-sameness witnesses. This gives $\mathsf{CategoryHomCarrier}(a, b, left)$ and $\mathsf{CategoryHomCarrier}(a, b, right)$.

Finally compose $\mathsf{hsame}(left, f)$ with the symmetry of $\mathsf{hsame}(right, f)$. Transitivity yields $\mathsf{hsame}(left, right)$, completing the two-sided identity classification. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier two sided identity law`

Theorem 61.103 (Category hom carrier left identity carrier result)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, a, id)$*,* $\mathsf{CategoryHomCarrier}(a, b, f)$*, and* $\mathsf{Cont}(id, f, left)$*, then*

$$\mathsf{CategoryHomCarrier}(a, b, left) \wedge \mathsf{hsame}(left, f).$$

Proof. Compare the displayed identity carrier id with the canonical empty identity carrier at a. Morphism determinacy gives $\mathsf{hsame}(id, \mathsf{e})$. Rewriting the displayed continuation by this equality turns $\mathsf{Cont}(id, f, left)$ into an empty-left continuation, so the kernel left-unit result gives $\mathsf{hsame}(left, f)$. Transport the original hom-carrier witness for f across this morphism equality to obtain $\mathsf{CategoryHomCarrier}(a, b, left)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier left identity carrier result`

Theorem 61.104 (Category hom carrier left identity result iff)**.** *For the unary-continuation category, for histories* $a, b, id, f, left$*, if* $\mathsf{CategoryHomCarrier}(a, a, id)$ *and* $\mathsf{CategoryHomCarrier}(a, b, f)$*, then*

$$\mathsf{Cont}(id, f, left) \iff \mathsf{hsame}(left, f).$$

Proof. Compare the displayed identity carrier id with the canonical empty identity at a, giving $\mathsf{hsame}(id, \mathsf{e})$. The forward direction is the left identity carrier result. For the reverse direction, rewrite the left endpoint by $\mathsf{hsame}(id, \mathsf{e})$ and use the BEDC left-unit continuation characterized by $\mathsf{hsame}(left, f)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier left identity result iff`

Theorem 61.105 (Category hom carrier right identity carrier result)**.** *For the unary-continuation category, for histories* $a, b, id, f, right$*, if* $\mathsf{CategoryHomCarrier}(b, b, id)$*,* $\mathsf{CategoryHomCarrier}(a, b, f)$*, and* $\mathsf{Cont}(f, id, right)$*, then*

$$\mathsf{CategoryHomCarrier}(a, b, right) \wedge \mathsf{hsame}(right, f).$$

Proof. Compare the displayed identity carrier id with the canonical empty identity carrier at b by hom-morphism determinacy, giving $\mathsf{hsame}(id, \mathsf{e})$. Rewriting the displayed continuation by this equality turns $\mathsf{Cont}(f, id, right)$ into $\mathsf{Cont}(f, \mathsf{e}, right)$. This empty-right continuation is determined by the BEDC right-unit continuation on f, giving $\mathsf{hsame}(right, f)$. Transport the displayed $\mathsf{CategoryHomCarrier}(a, b, f)$ witness across this morphism equality to obtain $\mathsf{CategoryHomCarrier}(a, b, right)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier right identity carrier result`

Theorem 61.106 (Category hom carrier right identity result iff)**.** *For the unary-continuation category, for histories* $a, b, id, f, right$*, if* $\mathsf{CategoryHomCarrier}(b, b, id)$ *and* $\mathsf{CategoryHomCarrier}(a, b, f)$*, then*

$$\mathsf{Cont}(f, id, right) \iff \mathsf{hsame}(right, f).$$

Proof. Compare the displayed identity carrier id with the canonical empty identity at b, giving $\mathsf{hsame}(id, \mathsf{e})$. The forward direction is the right identity carrier result. For the reverse direction, rewrite the right endpoint by $\mathsf{hsame}(id, \mathsf{e})$ and use the BEDC right-unit continuation characterized by $\mathsf{hsame}(right, f)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier right identity result iff`

Theorem 61.107 (Category hom carrier composite left-right identity result)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, a, id_A)$*,* $\mathsf{CategoryHomCarrier}(b, b, id_B)$*,* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{Cont}(id_A, f, left)$*, and* $\mathsf{Cont}(left, id_B, both)$*, then*

$$\mathsf{CategoryHomCarrier}(a, b, both) \wedge \mathsf{hsame}(both, f).$$

Proof. Apply the left identity carrier result to the first displayed continuation. This gives $\mathsf{CategoryHomCarrier}(a, b, le$ and $\mathsf{hsame}(left, f)$. Then apply the right identity carrier result to $\mathsf{Cont}(left, id_B, both)$, using the just-obtained carrier witness for $left$. This gives $\mathsf{CategoryHomCarrier}(a, b, both)$ and $\mathsf{hsame}(both, left)$. Transitivity with $\mathsf{hsame}(left, f)$ yields the displayed result equality. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left right identity result`

Theorem 61.108 (Category hom carrier composite left-right identity public readback)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, a, id_A)$*,* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, b$ $\mathsf{Cont}(id_A, f, left)$*, and* $\mathsf{Cont}(left, id_B, result)$*, then*

$$\mathsf{CategoryHomCarrier}(a, b, result) \wedge \forall g,\ \mathsf{CategoryHomCarrier}(a, b, g) \Rightarrow \mathsf{hsame}(result, g).$$

Proof. Compare each displayed identity carrier with the canonical empty identity at its endpoint. The left identity collapses the first continuation result to f, giving a carrier witness for $left$. The right identity collapses the second continuation result to $left$, giving a carrier witness for $result$. Any other displayed hom carrier from a to b has the same morphism by hom-carrier morphism determinacy. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp left right identity public readback`

Theorem 61.109 (Category continuation left-right identity tail result)**.** *For BEDC histories* $a, b, id_A, f, id_B, left, both$*, if* $\mathsf{Cont}(a, id_A, a)$*,* $\mathsf{Cont}(a, f, b)$*,* $\mathsf{Cont}(b, id_B, b)$*,* $\mathsf{Cont}(id_A, f, left)$*, and* $\mathsf{Cont}(left, id_B, both)$*, then*

$$\mathsf{hsame}(both, f).$$

Proof. Compare $\mathsf{Cont}(a, id_A, a)$ and $\mathsf{Cont}(b, id_B, b)$ with the canonical right-unit continuations at a and b. Tail determinacy gives $\mathsf{hsame}(id_A, \mathsf{e})$ and $\mathsf{hsame}(id_B, \mathsf{e})$. The first equality turns $\mathsf{Cont}(id_A, f, left)$ into an empty-left continuation, so the kernel left-unit result gives $\mathsf{hsame}(left, f)$. Transport $\mathsf{Cont}(left, id_B, both)$ across this equality, use the second equality to make the right tail empty, and compare with the right-unit continuation on f. □

Lean4 ✓ `BEDC.Derived.CategoryUp.category cont left right identity tail result`

Theorem 61.110 (Continuation morphism left-right identity tail result)**.** *For continuation morphisms* $leftId : a \to a$*,* $m : a \to b$*, and* $rightId : b \to b$*, both identity tails are empty, and composing* m *with them on the left and right leaves its tail unchanged up to* hsame*:*

$$\mathsf{hsame}(leftId.tail, \mathsf{e}) \wedge \mathsf{hsame}(rightId.tail, \mathsf{e}) \wedge \mathsf{hsame}((leftId \circ m) \circ rightId,\ m.tail).$$

Proof. Compare each identity morphism with the canonical empty continuation at its endpoint to obtain the two empty-tail equalities. After destructing the three continuation-morphism records, the composite tail is the displayed append of the three tails. The append left-unit and right-unit laws reduce it to $m.tail$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left right identity tail result`

Theorem 61.111 (Continuation morphism fixed left tail has right identity)**.** *For continuation morphisms* $left : a \to b$ *and* $right : b \to c$*, if the composite append tail of* $left$ *followed by* $right$ *is history-same to* $left.tail$*, then* $right.tail$ *is empty and the endpoints* b *and* c *are history-same:*

$$\mathsf{hsame}(\mathsf{append}(left.tail, right.tail), left.tail) \Longrightarrow \mathsf{hsame}(right.tail, \mathsf{e}) \wedge \mathsf{hsame}(b, c).$$

Proof. The displayed fixed-left comparison is a self-continuation of $left.tail$ by $right.tail$. Right-unit uniqueness for continuation forces $right.tail$ to be empty. Transporting the right morphism's continuation relation to this empty tail and comparing it with the canonical right-unit continuation at b identifies b and c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left tail fixed right identity`

Theorem 61.112 (Continuation morphism fixed right tail has left identity)**.** *For continuation morphisms $left : a \to b$ and $right : b \to c$, if the composite append tail of $left$ followed by right is history-same to $right.tail$, then $left.tail$ is empty and the endpoints a and b are history-same:*

$$\mathsf{hsame}(\mathsf{append}(left.tail, right.tail), right.tail) \Longrightarrow \mathsf{hsame}(left.tail, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

Proof. The displayed fixed-right comparison is a self-continuation of $right.tail$ by $left.tail$. Left-unit uniqueness for continuation forces $left.tail$ to be empty. Transporting the left morphism's continuation relation to this empty tail and comparing it with the canonical right-unit continuation at a identifies a and b. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right tail fixed left identity`

Theorem 61.113 (Continuation morphism composite tail fixed by both factors)**.** *For continuation morphisms $left : a \to b$ and $right : b \to c$, the closed composite tail is fixed by both factor tails exactly when both factors are identity tails and both endpoint steps are history-same:*

$$\mathsf{hsame}((left \circ right).tail, left.tail) \wedge \mathsf{hsame}((left \circ right).tail, right.tail) \Longleftrightarrow \mathsf{hsame}(left.tail, \mathsf{e}) \wedge \mathsf{hsame}(right.tail, \mathsf{e})$$

Proof. After destructing the two continuation-morphism records, the closed composite tail is the append of the two displayed factor tails. If this append is history-same to the left tail, right-unit uniqueness forces the right tail empty; if it is history-same to the right tail, left-unit uniqueness forces the left tail empty. Transporting the two relation fields across those empty-tail equalities and comparing with canonical right-unit continuations gives the two endpoint comparisons. The converse reduces the append tail to the empty history and hence to each empty factor tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail fixed both identity iff`

Theorem 61.114 (Category hom carrier composite middle identity result)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, b, id)$*,* $\mathsf{CategoryHomCarrier}(b, c, g)$*,* $\mathsf{Cont}(f, id, fid)$*,* $\mathsf{Cont}(fid, g, result)$*, and* $\mathsf{Cont}(f, g, fg)$*, then*

$$\mathsf{CategoryHomCarrier}(a, c, result) \wedge \mathsf{CategoryHomCarrier}(a, c, fg) \wedge \mathsf{hsame}(result, fg).$$

Proof. Apply the right identity carrier result to $\mathsf{Cont}(f, id, fid)$, obtaining $\mathsf{CategoryHomCarrier}(a, b, fid)$ and $\mathsf{hsame}(fid, f)$. Composition closure with the right hom-carrier gives the displayed carrier for $result$, while direct composition closure gives the displayed carrier for fg. The two continuation witnesses have history-same left tails and the same right tail g, so continuation result respect for hsame gives $\mathsf{hsame}(result, fg)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp middle identity result`

Theorem 61.115 (Category hom carrier composite middle identity public readback)**.** *For the unary-continuation category, if* $\mathsf{CategoryHomCarrier}(a, b, f)$*,* $\mathsf{CategoryHomCarrier}(b, b, id)$*,* $\mathsf{CategoryHomCarrier}(b, c, g$ $\mathsf{Cont}(f, id, fid)$*, and* $\mathsf{Cont}(fid, g, result)$*, then*

$$\mathsf{CategoryHomCarrier}(a, c, result) \wedge \forall fg,\ \mathsf{CategoryHomCarrier}(a, c, fg) \Rightarrow \mathsf{hsame}(result, fg).$$

Proof. Compare the displayed middle identity with the canonical empty identity at b. This identifies id with e, so the continuation $\mathsf{Cont}(f, id, fid)$ is determined by the right-unit continuation on f. Transporting the left hom carrier across $\mathsf{hsame}(fid, f)$ makes fid a hom carrier from a to b. Composition closure with the displayed right hom carrier gives the carrier witness for $result$, and hom-carrier morphism determinacy reads it back against any other hom carrier from a to c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp middle identity public readback`

Theorem 61.116 (Category continuation middle identity tail result)**.** *For BEDC histories $a, b, c, f, id, g, fid, resul$ if $\mathsf{Cont}(a, f, b)$, $\mathsf{Cont}(b, id, b)$, $\mathsf{Cont}(b, g, c)$, $\mathsf{Cont}(f, id, fid)$, $\mathsf{Cont}(fid, g, result)$, and $\mathsf{Cont}(f, g, fg)$, then*

$$\mathsf{hsame}(result, fg).$$

Proof. Compare $\mathsf{Cont}(b, id, b)$ with the canonical right-unit continuation at b; tail determinacy gives $\mathsf{hsame}(id, \mathsf{e})$. Hence $\mathsf{Cont}(f, id, fid)$ is determined by the right-unit continuation on f, so $\mathsf{hsame}(fid, f)$. Continuation result respect for hsame, applied to the two displayed continuations with right tail g, yields $\mathsf{hsame}(result, fg)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.category cont middle identity tail result`

Lemma 61.117 (Category hom carrier right-append fixed identity bridge determinacy)**.** *For the unary-continuation category, fix a displayed hom carrier $\mathsf{CategoryHomCarrier}(a, b, f)$ and, for $g : \mathsf{BHist}$ and $fg : \mathsf{BHist}$, a fixed right-append comparison displayed by*

$$\mathsf{Cont}(f, g, fg) \wedge \mathsf{hsame}(fg, g).$$

Suppose the same consumer-facing surface is handed through the displayed left-right identity rows $\mathsf{CategoryHomCarrier}(a, a, id_A)$, $\mathsf{CategoryHomCarrier}(b, b, id_B)$, $\mathsf{Cont}(id_A, f, left)$, and $\mathsf{Cont}(left, id_B, result)$. Then the bridge is classified by exactly the local fixed-identity and public-readback rows:

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{CategoryHomCarrier}(a, b, result) \wedge \forall h,\ \mathsf{CategoryHomCarrier}(a, b, h) \Rightarrow \mathsf{hsame}(result, h).$$

Consequently every accepted route, transport, readback, or handoff for this right-append fixed identity factors through the displayed $\mathsf{CategoryHomCarrier}$ rows, the two identity-continuation rows, the fixed append comparison, and the public readback classifier. No unlisted bridge coordinate supplies an alternate tail, endpoint, transported carrier, or public morphism read.

Proof. Apply Definition 61.97 to the displayed carrier $\mathsf{CategoryHomCarrier}(a, b, f)$ and the fixed comparison represented by $\mathsf{Cont}(f, g, fg)$ together with $\mathsf{hsame}(fg, g)$. The forward implication gives

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b).$$

These are the only fixed-append data exported by the local theorem: the morphism tail is empty and the two endpoints are history-same.

Next apply Definition 61.108 to $\mathsf{CategoryHomCarrier}(a, a, id_A)$, $\mathsf{CategoryHomCarrier}(a, b, f)$, $\mathsf{CategoryHomCarrier}($ $\mathsf{Cont}(id_A, f, left)$, and $\mathsf{Cont}(left, id_B, result)$. It supplies the displayed carrier for the handoff result and the public readback clause

$$\mathsf{CategoryHomCarrier}(a, b, result) \wedge \forall h,\ \mathsf{CategoryHomCarrier}(a, b, h) \Rightarrow \mathsf{hsame}(result, h).$$

Combining this conjunction with the fixed-append conclusion gives the displayed classification.

The factorization claim follows from the two cited rows. A route to the fixed identity must first use the right-append comparison and therefore lands in $\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b)$. By Definition 61.93, a transport use can only move the displayed hom carrier along those hsame comparisons. A readback or handoff through the identity bridge is then exactly the public readback clause above: it classifies $result$ against every displayed hom carrier over the same endpoints, including the original carrier f. Since the hypotheses list no further carrier, continuation, classifier, or endpoint row, an alleged alternate tail, endpoint, transported carrier, or public morphism read has no local coordinate through which an accepted consumer could obtain it. □

Corollary 61.118 (Category hom carrier right-append fixed identity consumer determinacy)**.** *For the unary-continuation category, fix* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and a displayed fixed right-append row*

$$\mathsf{Cont}(f, g, fg) \wedge \mathsf{hsame}(fg, g).$$

Suppose the same consumer-facing identity surface also displays $\mathsf{CategoryHomCarrier}(a, a, id_A)$*,* $\mathsf{CategoryHomCarrier}(b, b, id_B)$*,* $\mathsf{Cont}(id_A, f, left)$*, and* $\mathsf{Cont}(left, id_B, result)$*. Then every displayed consumer candidate* h *over the same endpoints is forced to the chapter's fixed identity route and public normal form:*

$$\mathsf{CategoryHomCarrier}(a, b, h) \Longrightarrow \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(result, h).$$

Thus the accepted exactness decision is the pair $\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b)$*, and the accepted public readback is the single* hsame*-class of* $result$ *among displayed hom carriers from* a *to* b*. No hidden coordinate can supply a nonempty tail decision, a different endpoint classifier, or a duplicate public classifier source for this right-append fixed identity surface.*

Proof. Apply Definition 61.117 to the displayed hom carrier, the fixed right-append row, and the four identity surface rows. It gives

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{CategoryHomCarrier}(a, b, result) \wedge \forall k,\ \mathsf{CategoryHomCarrier}(a, b, k) \Rightarrow \mathsf{hsame}(result, k).$$

For the chosen consumer candidate h, instantiate the universal readback clause with its displayed carrier row $\mathsf{CategoryHomCarrier}(a, b, h)$. This yields $\mathsf{hsame}(result, h)$, while the first two conjuncts give the fixed identity decision $\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(a, b)$.

The content of that decision is exactly the one supplied by Definition 61.97: the fixed right-append comparison is accepted precisely through the empty-tail condition on f together with the endpoint classifier $\mathsf{hsame}(a, b)$. The bridge determinacy lemma then routes every public handoff through the displayed identity rows and the universal readback classifier for $result$. Hence any accepted route, normal-form read, or exactness decision using the same surface must factor through these listed rows; an unlisted tail, endpoint, or public classifier coordinate has no displayed carrier or continuation row from which it could be read. □

Theorem 61.119 (Continuation morphism composite middle identity tail result)**.** *For continuation morphisms* $left : a \to b$*,* $identity : b \to b$*, and* $right : b \to c$*, the middle identity tail is empty, and inserting it between* $left$ *and* $right$ *leaves the composite tail history-same to the direct composite:*

$$\mathsf{hsame}(identity.tail, \mathsf{e}) \wedge \mathsf{hsame}((left \circ identity) \circ right,\ left \circ right).$$

Proof. Compare $identity$ with the canonical empty continuation morphism at b by tail determinacy, giving $\mathsf{hsame}(identity.tail, \mathsf{e})$. Unfolding the two composite tails reduces the comparison to append histories. The middle tail is transported to e, and the right-unit law $\mathsf{append}(left.tail, \mathsf{e})\mathsf{hsame} left.tail$ collapses the displayed middle factor. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp middle identity tail result`

Theorem 61.120 (Category hom carrier identity semantic name certificate)**.** *For a unary object history* a*, the identity hom-carrier predicate* $f \mapsto \mathsf{CategoryHomCarrier}(a, a, f)$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are this hom-carrier predicate and whose classifier is* hsame*.*

Proof. The carrier witness is the empty history, using the empty identity hom-carrier at a. The classifier laws are the reflexive, symmetric, and transitive laws for hsame. Carrier transport follows by transporting the concrete hom-carrier witness across history sameness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier identity semanticNameCert`

Theorem 61.121 (Category hom carrier semantic name certificate from a hom)**.** *For displayed unary-continuation hom data* $\mathsf{CategoryHomCarrier}(a, b, f)$*, the hom-carrier predicate* $g \mapsto \mathsf{CategoryHomCarrier}(a, b, g$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are this hom-carrier predicate and whose classifier is* hsame*.*

Proof. The displayed morphism f supplies carrier inhabitation. The classifier laws are reflexivity, symmetry, and transitivity of hsame. If a classified morphism is transported across hsame, endpoint histories stay fixed and Definition 61.93 rebuilds the hom-carrier witness. Pattern and ledger soundness are identities because the three semantic predicates are the same hom-carrier predicate. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier semanticNameCert of hom`

Theorem 61.122 (Category hom carrier unary suffix semantic boundary)**.** *For a unary suffix* q *and displayed hom carrier* $\mathsf{CategoryHomCarrier}(a, b, f)$*, the suffixed hom-carrier predicate* $g \mapsto \mathsf{CategoryHomCarrier}(\mathsf{append}(a, q), \mathsf{append}(b, q), g)$ *carries a semantic naming certificate with classifier* hsame*. The displayed endpoint is exact:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(a, q), \mathsf{append}(b, q), f) \iff \mathsf{UnaryHistory}(q) \wedge \mathsf{CategoryHomCarrier}(a, b, f).$$

Proof. The unary suffix equivalence rebuilds the displayed suffixed hom carrier from q and the base hom carrier. The existing hom-carrier semantic certificate is then applied at the suffixed endpoints. The same unary suffix equivalence gives the exact endpoint readback. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary suffix semantic boundary`

Theorem 61.123 (Category hom carrier composite-result semantic name certificate)**.** *For displayed hom carriers* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and* $\mathsf{CategoryHomCarrier}(b, c, g)$ *together with* $\mathsf{Cont}(f, g, fg)$*, the predicate*

$$h \mapsto \mathsf{CategoryHomCarrier}(a, c, h) \wedge h\mathsf{hsame}fg$$

carries a semantic naming certificate whose source, pattern, and ledger predicates are this same concrete composite-result predicate and whose classifier is hsame*.*

Proof. Composition closure gives the carrier witness at fg, and reflexivity gives the displayed comparison $fg\mathsf{hsame}fg$. The classifier laws are the reflexive, symmetric, and transitive laws for hsame. Transport along a classifier proof uses hom-carrier transport for the fixed endpoints a, c and transitivity with the inverse classifier proof to keep the displayed composite result aligned with fg. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result semanticNameCert`

Theorem 61.124 (Category hom carrier composite-result endpoint semantic name certificate)**.** *For displayed hom carriers* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and* $\mathsf{CategoryHomCarrier}(b, c, g)$ *together with* $\mathsf{Cont}(f, g, fg)$*, if* $a\mathsf{hsame}a'$ *and* $c\mathsf{hsame}c'$*, then the transported predicate*

$$h \mapsto \mathsf{CategoryHomCarrier}(a', c', h) \wedge h\mathsf{hsame}fg$$

carries a semantic naming certificate whose source, pattern, and ledger predicates are this same transported composite-result predicate and whose classifier is hsame*.*

Proof. Composition closure first gives the concrete composite hom carrier at endpoints (a, c) and result fg. Endpoint transport along $a\mathsf{hsame}a'$ and $c\mathsf{hsame}c'$ moves this carrier to (a', c'), giving inhabitation at fg. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Carrier transport keeps endpoints fixed at (a', c'), transports the displayed hom carrier along the classifier proof, and composes the inverse classifier proof with the stored comparison to fg. Pattern and ledger soundness are identities because all three semantic predicates coincide. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result endpoint semanticNameCert`

Theorem 61.125 (Continuation morphism source semantic name certificate)**.** *For a displayed continuation morphism* $m : \mathsf{ContinuationMorphism}(a, b)$*, the source predicate* $src \mapsto \mathsf{Nonempty}(\mathsf{ContinuationMorphism}(s$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are this same source predicate and whose classifier is* hsame*.*

Proof. The displayed morphism m gives carrier inhabitation at source a. Reflexivity, symmetry, and transitivity of hsame supply the classifier laws. If a source endpoint is transported along hsame, the same displayed tail remains valid by continuation transport, so the concrete continuation-morphism witness is rebuilt at the transported source. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism source semanticNameCert`

Theorem 61.126 (Continuation morphism target semantic name certificate)**.** *For a displayed continuation morphism* $m : \mathsf{ContinuationMorphism}(a, b)$*, the target predicate* $tgt \mapsto \mathsf{Nonempty}(\mathsf{ContinuationMorphism}(a,$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are this same target predicate and whose classifier is* hsame*.*

Proof. The displayed morphism m gives carrier inhabitation at target b. The classifier laws are the three basic laws of hsame. Carrier transport uses continuation transport on the result endpoint while keeping the source and displayed tail fixed, thereby rebuilding a concrete continuation morphism at the transported target. Pattern and ledger soundness are identities for the same predicate. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism target semanticNameCert`

Theorem 61.127 (Composite continuation tail semantic name certificate)**.** *For histories* a, b, c *and continuation morphisms* $left : \mathsf{ContinuationMorphism}(a, b)$ *and* $right : \mathsf{ContinuationMorphism}(b, c)$*, let* m *be their closed composite. The predicate*

$$t \mapsto \mathsf{Cont}(a, t, c) \wedge \mathsf{hsame}(t, \mathsf{tail}(m))$$

carries a semantic naming certificate whose source, pattern, and ledger predicates are this same concrete composite-tail predicate and whose classifier is hsame*.*

Proof. The carrier witness is $\mathsf{tail}(m)$, using the continuation relation of the closed composite and reflexivity of hsame. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Transport along a classifier proof preserves the displayed continuation by continuation transport, and it preserves the tail classifier by transitivity with the inverse classifier proof. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail semanticNameCert`

Per-object certificate: traditional target category identity laws; BEDC source CategoryHomCarrier plus Cont; carrier BHist hom witnesses; classifier hsame; stability obligation two-sided identity under morphism classifier; ledger the empty-composition ledgers; core theorem list entry two-sided identity; exactness predicate sameness with the original morphism; standard bridge deferred; status ProofTarget.

Lemma 61.128 (Category hom identity semantic name certificate bridge determinacy)**.** *Let a be a unary object history, and consider the identity hom-carrier naming surface of Definition 61.120. Any consumer-facing use of this identity surface as a route, transport, readback, or handoff factors through the following displayed local rows: the identity hom-carrier predicate $f \mapsto \mathsf{CategoryHomCarrier}(a, a, f)$, the classifier row* hsame*, carrier transport for* CategoryHomCarrier*, and the continuation-source certificate row of Definition 61.125 applied to the continuation morphism encoded by the identity carrier. Thus an accepted consumer cannot obtain a primitive host identity arrow, a second morphism classifier, an off-carrier readback, a detached continuation source, or any unlisted bridge coordinate from the identity name certificate.*

Proof. The identity certificate of Definition 61.120 names exactly one source, pattern, and ledger predicate, namely $f \mapsto \mathsf{CategoryHomCarrier}(a, a, f)$, and it names exactly one classifier, namely hsame. Therefore every route that reads the identity surface first lands in the displayed hom-carrier predicate over the fixed endpoints (a, a). The identity witness itself is the empty history supplied by Definition 61.2, so the route has no separate host identity coordinate beyond the BEDC carrier row.

A transport use is fixed by the same certificate. If the displayed morphism history is changed only by hsame, then Definition 61.93 rebuilds the transported hom-carrier witness over the same identity endpoints. This is the full transport surface of the identity name certificate: the certificate has no second classifier and no independent transport predicate outside CategoryHomCarrier and hsame.

A readback or handoff to continuation data also remains local. By the hom-carrier and continuation-morphism tail correspondence of Definition 61.317, a read of $\mathsf{CategoryHomCarrier}(a, a, f)$ is exactly a unary endpoint and unary tail presentation of a continuation morphism $m : \mathsf{ContinuationMorphism}(a, a)$ whose displayed tail is classified with f. Applying Definition 61.125 to this displayed continuation morphism shows that a consumer may read or transport the source only through the source predicate $src \mapsto \mathsf{Nonempty}(\mathsf{ContinuationMorphism}(src, a))$ and its same classifier hsame. Hence the continuation handoff is not an additional bridge coordinate; it is the source row already determined by the identity hom-carrier readback.

The listed rows exhaust the consumer-facing bridge. Routes are the fixed identity hom-carrier predicate, transports are the hsame hom-carrier transports, readbacks are the hom-carrier-to-continuation tail correspondence, and continuation handoffs are the continuation-source semantic certificate for the resulting morphism. Since none of these rows contains a primitive host identity arrow, a second morphism classifier, an off-carrier readback, a detached continuation source, or an extra bridge field, such a coordinate has no row through which an accepted consumer could read the identity semantic name certificate. □

Corollary 61.129 (Category hom identity semantic name certificate consumer determinacy)**.** *Let a be a unary object history, and fix the identity hom-carrier semantic name certificate of Definition 61.120. Once the displayed identity rows are fixed, every accepted consumer-facing use of this surface as a finite route, transport, readback normal form, or exactness decision is determined by the same local data:*

$$f \mapsto \mathsf{CategoryHomCarrier}(a, a, f), \qquad \mathsf{hsame},$$

hom-carrier transport across hsame*, the continuation-source row reached from the hom-carrier readback.*

Thus two accepted consumers over these rows cannot choose different identity routes, different morphism classifiers, different readback normal forms, or different exactness decisions. Any proposed primitive host identity arrow, off-carrier coordinate, detached continuation source, unlisted bridge row, or duplicate classifier source is outside the accepted identity semantic name surface.

Proof. The identity certificate in Definition 61.120 fixes the semantic name surface before any consumer reads it. Its source, pattern, and ledger predicates are all the single predicate $f \mapsto \mathsf{CategoryHomCarrier}(a, a, f)$, and its classifier is the single history classifier hsame. Therefore a consumer route has no independent choice of carrier predicate or classifier: reading the identity name means reading the displayed hom-carrier predicate over the fixed endpoints (a, a).

Apply Definition 61.128. It states that every accepted route, transport, readback, or handoff through this identity surface factors through exactly the displayed local rows: the identity hom-carrier predicate, the classifier hsame, hom-carrier transport for $\mathsf{CategoryHomCarrier}$, and the continuation-source certificate obtained from the continuation morphism encoded by the identity carrier. Hence the consumer-facing finite route is not a further datum; it is the ordered use of those rows.

The same factorization fixes the normal-form and exactness readings. A transport read can only be the hsame hom-carrier transport over the fixed identity endpoints. A readback can only pass through the hom-carrier to continuation-tail correspondence named in the bridge-determinacy lemma, and its continuation handoff can only use the continuation-source semantic certificate for the resulting morphism. Consequently an exactness decision about the identity name is a decision about membership in the displayed hom-carrier row and its hsame readback, not a decision supplied by a second host arrow, second classifier, or detached continuation coordinate.

Suppose two accepted consumers over the fixed rows produced different identity routes, different readback normal forms, or different exactness decisions. The difference would have to enter through a row not among the identity hom-carrier predicate, hsame, the hom-carrier transport row, or the continuation-source row reached from the readback. But the bridge-determinacy lemma excludes exactly those unlisted coordinates: primitive host identity arrows, second morphism classifiers, off-carrier readbacks, detached continuation sources, and extra bridge fields have no exported row in the identity semantic name certificate. Thus every accepted consumer over the fixed rows determines the same chapter-named finite route, normal-form readback, and exactness decision. □

Theorem 61.130 (Category hom carrier associativity witness)**.** *For the unary-continuation category, let f be a hom-carrier from a to b, let g be a hom-carrier from b to c, and let h be a hom-carrier from c to d. If fg, gh, $left$, and $right$ satisfy*

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(g, h, gh), \quad \mathsf{Cont}(fg, h, left), \quad \mathsf{Cont}(f, gh, right),$$

then $left$ and $right$ are both hom-carriers from a to d, and $\mathsf{hsame}(left, right)$.

Proof. First build the left-associated carrier. Apply Theorem 61.19 to the carriers $f : a \to b$ and $g : b \to c$ with $\mathsf{Cont}(f, g, fg)$, obtaining $fg : a \to c$. Apply the same theorem to $fg : a \to c$ and $h : c \to d$ with $\mathsf{Cont}(fg, h, left)$, obtaining $left : a \to d$.

Next build the right-associated carrier. Apply Theorem 61.19 to the carriers $g : b \to c$ and $h : c \to d$ with $\mathsf{Cont}(g, h, gh)$, obtaining $gh : b \to d$. Apply the same theorem to $f : a \to b$ and $gh : b \to d$ with $\mathsf{Cont}(f, gh, right)$, obtaining $right : a \to d$.

Finally apply continuation associativity, Theorem 4.56, to the displayed continuation square. The two composition trees therefore have the same morphism classifier, namely $\mathsf{hsame}(left, right)$. □

Per-object certificate: traditional target categorical associativity; BEDC source $\mathsf{CategoryHomCarrier}$ plus Cont; carrier BHist hom witnesses; classifier hsame; stability obligation associativity under morphism classifier; ledger the two composition trees; core theorem list entry this associativity witness; exactness predicate equality of the two composite trees; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 61.131 (Category hom carrier fourfold associativity witness)**.** *For the unary-continuation category, let f be a hom-carrier from a to b, let g be a hom-carrier from b to c, let h be a hom-carrier from c to d, and let i be a hom-carrier from d to e. If fg, fgh, hi, ghi, $left$, and $right$ satisfy*

$$\mathsf{Cont}(f, g, fg), \quad \mathsf{Cont}(fg, h, fgh), \quad \mathsf{Cont}(h, i, hi), \quad \mathsf{Cont}(g, hi, ghi), \quad \mathsf{Cont}(fgh, i, left), \quad \mathsf{Cont}(f, ghi, right),$$

then $left$ and $right$ are both hom-carriers from a to e, and $\mathsf{hsame}(left, right)$.

Proof. First build the left fourfold carrier. Apply Theorem 61.19 to $f : a \to b$ and $g : b \to c$ with $\mathsf{Cont}(f, g, fg)$, obtaining $fg : a \to c$. Apply the same theorem to $fg : a \to c$ and $h : c \to d$ with $\mathsf{Cont}(fg, h, fgh)$, obtaining $fgh : a \to d$. Apply it once more to $fgh : a \to d$ and $i : d \to e$ with $\mathsf{Cont}(fgh, i, left)$, obtaining $left : a \to e$.

Next build the right fourfold carrier. Apply Theorem 61.19 to $h : c \to d$ and $i : d \to e$ with $\mathsf{Cont}(h, i, hi)$, obtaining $hi : c \to e$. Apply the same theorem to $g : b \to c$ and $hi : c \to e$ with $\mathsf{Cont}(g, hi, ghi)$, obtaining $ghi : b \to e$. Apply it once more to $f : a \to b$ and $ghi : b \to e$ with $\mathsf{Cont}(f, ghi, right)$, obtaining $right : a \to e$.

Finally apply the kernel fourfold continuation associativity witness, $cont_assoc_four$, to the six displayed continuation relations. The two fourfold composition ledgers therefore generate the same composite history classifier, namely $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier fourfold assoc witness`

Per-object certificate: traditional target fourfold categorical associativity coherence; BEDC source CategoryHomCarrier plus Cont; carrier BHist hom witnesses; classifier hsame; stability obligation rebracketing under morphism classifier; ledger the two fourfold composition trees; core theorem list entry fourfold coherence; exactness predicate equality of generated composite histories; standard bridge deferred; status ProofTarget.

Theorem 61.132 (Category hom carrier associative composition closure)**.** *For the unary-continuation category, composable hom-carriers $f : a \to b$, $g : b \to c$, and $h : c \to d$ remain hom-carriers after either association of composition. If* $\mathsf{Cont}(f, g, fg)$, $\mathsf{Cont}(g, h, gh)$, $\mathsf{Cont}(fg, h, left)$, *and* $\mathsf{Cont}(f, gh, right)$, *then both $left$ and $right$ are hom-carriers from a to d, and* $\mathsf{hsame}(left, right)$.

Proof. Apply hom-carrier composition closure first to f, g and then to fg, h for the left association. Apply the same closure first to g, h and then to f, gh for the right association. The continuation associativity law identifies the two composite histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp assoc closed`

Theorem 61.133 (Category hom carrier displayed associative determinacy)**.** *For the unary-continuation category, assume composable hom-carriers $f : a \to b$, $g : b \to c$, and $h : c \to d$ with continuation witnesses for fg, gh, $left$, and $right$ as in associative composition closure. If a displayed hom-carrier from a to d is also given, then both bracketed composite histories are* hsame *to the displayed morphism history.*

Proof. Associative composition closure constructs the left- and right-associated hom-carriers from a to d. The hom-carrier morphism determinacy theorem compares each of these carriers with the displayed carrier over the same endpoints. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp assoc displayed deterministic`

Lean4 (definition) `BEDC.Derived.CategoryUp.ContinuationMorphism`

Theorem 61.134 (Continuation morphism tail semantic name certificate)**.** *For histories a, b and a continuation morphism $m : a \to b$, the predicate $t \mapsto \mathsf{Cont}(a, t, b)$ carries a semantic naming certificate with source, pattern, and ledger all given by this same continuation-tail predicate and classifier* hsame*.*

Proof. The carrier witness is the displayed tail of m together with its continuation relation. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Carrier transport follows by substituting the history-sameness proof into the concrete continuation-tail predicate. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail semanticNameCert`

Theorem 61.135 (Continuation morphism tail-continuation witness iff)**.** *For histories a, b, t : BHist, a continuation morphism from a to b whose displayed tail is history-same to t exists exactly when t is a continuation tail from a to b:*

$$\exists m : \mathsf{ContinuationMorphism}(a, b),\ \mathsf{hsame}(\mathsf{tail}(m), t) \quad \Longleftrightarrow \quad \mathsf{Cont}(a, t, b).$$

Proof. In the forward direction, unpack the displayed continuation morphism. Its record field gives $\mathsf{Cont}(a, \mathsf{tail}(m), b)$, and continuation transport along the tail classifier $\mathsf{hsame}(\mathsf{tail}(m), t)$ gives $\mathsf{Cont}(a, t, b)$.

Conversely, a witness $\mathsf{Cont}(a, t, b)$ itself forms a continuation morphism from a to b with displayed tail t, and tail classification is reflexive. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail cont witness iff`

Theorem 61.136 (Continuation morphism tail determinacy)**.** *For histories a, b, any two continuation morphisms from a to b have history-same tails:*

$$left, right : \mathsf{ContinuationMorphism}(a, b) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(left), \mathsf{tail}(right)).$$

Proof. Unpack both continuation morphism records. Their continuation witnesses have the same source a and target b, so continuation left-context cancellation identifies the two displayed tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail deterministic`

Theorem 61.137 (Continuation morphism empty-source tail target exactness)**.** *For every history target, a continuation morphism from the empty source to target has displayed tail history-same to target:*

$$m : \mathsf{ContinuationMorphism}(\mathsf{e}, target) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(m), target).$$

Proof. Unpack the continuation morphism record. Its continuation witness has source e, displayed tail $\mathsf{tail}(m)$, and result *target*. The left-unit continuation law identifies the result with the tail, and symmetry of hsame gives the displayed orientation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism empty source tail target hsame`

Theorem 61.138 (Continuation morphism empty tail endpoint equivalence)**.** *For histories a, b and a continuation morphism $m : a \to b$, the displayed tail of m is history-same to* e *if and only if the source and target endpoints are history-same:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(a, b).$$

Proof. Unpack m as a continuation witness $\mathsf{Cont}(a, \mathsf{tail}(m), b)$. If the tail is e, compare this witness with the right-unit continuation at a; continuation result determinacy gives the endpoint comparison.

Conversely, if $a\mathsf{hsame}b$, then the continuation witness identifies $\mathsf{append}(a, \mathsf{tail}(m))$ with b, and the endpoint comparison identifies b with $\mathsf{append}(a, \mathsf{e})$. Left cancellation of the common source a gives $\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail empty endpoint hsame iff`

Theorem 61.139 (Continuation morphism endpoint transport tail classification)**.** *For histories a, a', b, b', if a is history-same as a' and b is history-same as b', then the tail of any continuation morphism from a to b transports to a continuation tail from a' to b'. Any displayed continuation morphism from a' to b' has a tail classified with this transported tail.*

Proof. Transport the continuation relation carried by the left morphism across the source and target endpoint sameness witnesses, using reflexivity for the tail itself. The transported relation and the relation carried by the right morphism share source and target endpoints, so continuation left cancellation identifies the two tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism endpoint transport tail classified`

Theorem 61.140 (Continuation morphism endpoint transport existence)**.** *For histories a, a', b, b', if $a\mathsf{hsame}a'$ and $b\mathsf{hsame}b'$, then every continuation morphism from a to b induces a continuation morphism from a' to b' whose tail is history-same to the original tail. Moreover the induced morphism carries the transported continuation witness at its own tail.*

Proof. Reuse the original tail. Transport the continuation relation carried by the given morphism across the source and target endpoint sameness witnesses, using reflexivity on the tail. The resulting transported relation is the relation field of the displayed morphism from a' to b', and the tail comparison is reflexivity. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism endpoint transport exists`

Theorem 61.141 (Continuation morphism endpoint transport witness)**.** *For histories a, a', b, b', if $a\mathsf{hsame}a'$ and $b\mathsf{hsame}b'$, then every continuation morphism $m : \mathsf{ContinuationMorphism}(a, b)$ transports to a continuation morphism $n : \mathsf{ContinuationMorphism}(a', b')$ whose tail is history-same to the original tail:*

$$\exists n : \mathsf{ContinuationMorphism}(a', b').\ \mathsf{hsame}(\mathsf{tail}(n), \mathsf{tail}(m)) \wedge \mathsf{Cont}(a', \mathsf{tail}(n), b').$$

Proof. Unpack m as a tail with a continuation relation from a to b. Transport that relation across the source and target endpoint comparisons, keeping the same tail by reflexivity. The transported relation is the relation field of the displayed morphism n, and tail reflexivity gives the announced comparison. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism endpoint transport witness`

Theorem 61.142 (Continuation morphism source determinacy)**.** *For histories a, b, c, if a continuation morphism from a to c and a continuation morphism from b to c display history-same tails, then their sources are history-same:*

$$left : \mathsf{ContinuationMorphism}(a, c), \quad right : \mathsf{ContinuationMorphism}(b, c), \quad \mathsf{hsame}(\mathsf{tail}(left), \mathsf{tail}(right)) \Longrightarrow \mathsf{hsame}($$

Proof. Unpack both continuation morphism records. The displayed tail equality makes the two continuation witnesses share the same tail and the same result c. Continuation right-context cancellation then identifies the two source histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism source deterministic`

Theorem 61.143 (Continuation morphism target determinacy)**.** *For histories a, b, c, if a continuation morphism from a to b and a continuation morphism from a to c display history-same tails, then their targets are history-same:*

$$left : \mathsf{ContinuationMorphism}(a, b), \quad right : \mathsf{ContinuationMorphism}(a, c), \quad \mathsf{hsame}(\mathsf{tail}(left), \mathsf{tail}(right)) \Longrightarrow \mathsf{hsame}(b$$

Proof. Unpack both continuation morphism records. The displayed tail equality makes the two continuation witnesses share the same source a and the same tail. Continuation result determinacy then identifies the two target histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism target deterministic`

Theorem 61.144 (Category empty-target hom chain inversion)**.** *If f is a hom-carrier from a to b and g is a hom-carrier from b to* e*, then the source, middle object, and both displayed morphisms are all history-same to* e*:*

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(b, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Proof. Empty-target exactness applied to the right hom-carrier identifies b and g with e. Transporting the left hom-carrier along this target identification makes it a hom-carrier into e, so empty-target exactness applied again identifies a and f with e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target chain inversion`

Theorem 61.145 (Continuation morphism empty target inversion)**.** *For a continuation morphism from source to* e*, both the source and the displayed tail are history-same to* e*:*

$$m : \mathsf{ContinuationMorphism}(source, \mathsf{e}) \Longrightarrow \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}).$$

Proof. Unpack the continuation morphism record. Its continuation ledger has empty target, so the kernel empty-result inversion theorem identifies both the source and the displayed tail with e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism empty target inversion`

Theorem 61.146 (Continuation morphism visible source empty target impossibility)**.** *For every history a :* BHist*, neither visible-headed source admits a continuation morphism into* e*:*

$$\mathsf{ContinuationMorphism}(\mathsf{e0}(a), \mathsf{e}) \to \mathsf{False}$$

and

$$\mathsf{ContinuationMorphism}(\mathsf{e1}(a), \mathsf{e}) \to \mathsf{False}.$$

Proof. Apply empty-target inversion to the displayed continuation morphism. The inversion identifies the source endpoint with e. For source $\mathsf{e0}(a)$ or $\mathsf{e1}(a)$ this contradicts generated-history constructor separation from the empty history. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism visible source empty target absurd`

Definition 61.147 (Continuation morphism composition closure)**.** *Given continuation morphisms from a to b and from b to c, their closed composite is the continuation morphism from a to c whose tail is the append of the two displayed tails. The continuation witness is obtained by unpacking the first relation and reassociating the append spine.*

Lean4 (definition) `BEDC.Derived.CategoryUp.ContinuationMorphism comp closed`

Theorem 61.148 (Continuation morphism composite empty target inversion)**.** *For histories a, b, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, \mathsf{e})$. Then both endpoints and both displayed tails are history-same to e:*

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(b, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Proof. Unpack the two continuation morphism records. Empty-result inversion on the right witness identifies b and the displayed tail of $right$ with e. After transporting the left witness across this endpoint identification, a second empty-result inversion identifies a and the displayed tail of $left$ with e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp empty target inversion`

Theorem 61.149 (Continuation morphism composite empty middle inversion)**.** *For histories a, c, let $left$: $\mathsf{ContinuationMorphism}(a, \mathsf{e})$ and $right$: $\mathsf{ContinuationMorphism}(\mathsf{e}, c)$. Then*

$$\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), c) \wedge \mathsf{hsame}(\mathsf{tail}(left \circ right), c).$$

Proof. Empty-result inversion on the left witness identifies a and the displayed tail of $left$ with e. The right witness is a left-unit continuation, so its displayed tail is history-same to c. The composite displayed tail is the append of the two displayed tails; replacing the left one by e and using left-unit append gives the same endpoint comparison with c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp empty middle inversion`

Theorem 61.150 (Continuation morphism composite tail hom carrier)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If a, c, and both displayed tails are unary histories, then the canonical composite tail is a hom carrier from a to c:*

$$\mathsf{CategoryHomCarrier}(a, c, \mathsf{tail}(\mathsf{ContinuationMorphism_comp_closed}(left, right))).$$

Proof. The composite morphism record supplies the continuation relation from a through the appended tail to c. Unary append closure combines the two unary tail witnesses, while the assumed unary source and target witnesses fill the remaining fields of $\mathsf{CategoryHomCarrier}$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail CategoryHomCarrier`

Theorem 61.151 (Continuation morphism composite tail hom-carrier iff)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. The canonical composite tail is a hom carrier from a to c if and only if the source, target, and both component tails are unary:*

$$\mathsf{CategoryHomCarrier}(a, c, \mathsf{tail}(\mathsf{ContinuationMorphism_comp_closed}(left, right))) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}$$

Proof. Forward, unfold the displayed hom carrier and split the unary append carried by the composite tail into its two component tail factors. Reverse, use unary append closure on the component tails together with the composite continuation relation stored by ContinuationMorphism_comp_closed. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail CategoryHomCarrier iff`

Theorem 61.152 (Continuation morphism composite tail hom-carrier factors iff)**.** *For histories a, b, c, let $left$: ContinuationMorphism(a, b) and $right$: ContinuationMorphism(b, c). The canonical composite tail is a hom carrier from a to c if and only if the endpoints a, c are unary and the two component tails are themselves the hom carriers $a \to b$ and $b \to c$:*

$$\mathsf{CategoryHomCarrier}(a, c, \mathsf{tail}(\mathsf{ContinuationMorphism_comp_closed}(left, right))) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}$$

Proof. Forward, the composite hom carrier gives unary source and target data and a unary composite tail. Splitting that tail yields unary witnesses for the two component tails, while the left continuation relation closes the middle object b as unary. The stored continuation relations then assemble the two component hom carriers. Reverse, the two component hom carriers provide the source, target, and tail unary data needed by the composite-tail hom-carrier closure. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail CategoryHomCarrier factors iff`

Theorem 61.153 (Continuation morphism composite empty-tail inversion)**.** *For histories a, b, c, let $left : a \to b$ and $right : b \to c$ be continuation morphisms. If the canonical composite continuation morphism has displayed tail history-same to* e*, then both component tails are history-same to* e*, and both endpoints are identified:*

$$\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \implies \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Unpack the two continuation morphism records. The composite displayed tail is the BEDC append of the two component tails; empty append inversion forces both component tails to be e. The original continuation witnesses then compare against the right-unit continuations at a and b, so determinacy identifies the corresponding endpoints. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp empty tail inversion`

Theorem 61.154 (Continuation morphism composite tail empty iff)**.** *For histories a, b, c, let $left : a \to b$ and $right : b \to c$ be continuation morphisms. The canonical composite has displayed tail history-same to* e *if and only if both component tails are history-same to* e*:*

$$\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \iff \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Proof. The forward direction is the tail part of composite empty-tail inversion. For the reverse direction, unpack the two continuation morphism records. Once both component tails are identified with e, the composite displayed tail is $\mathsf{append}(\mathsf{e}, \mathsf{e})$, hence it is history-same to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail empty iff`

Theorem 61.155 (Continuation morphism composite right visible tail empty absurd)**.** *For histories a, b, c, m, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the right displayed tail is a visible constructor, then the canonical composite displayed tail cannot be history-same to* e*:*

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e0}(m)) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Longrightarrow \mathsf{False}$$

and

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e1}(m)) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Longrightarrow \mathsf{False}.$$

Proof. Unpack both continuation morphism records. The composite tail is the kernel append of the left displayed tail with the right displayed tail. If this append were empty, append-empty inversion would force the right displayed tail itself to be e. This contradicts constructor separation for either $\mathsf{e0}(m)$ or $\mathsf{e1}(m)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right visible tail empty absurd`

Theorem 61.156 (Continuation morphism right visible tail target not empty)**.** *For histories b, c, m and a continuation morphism $right$: $\mathsf{ContinuationMorphism}(b, c)$, if the displayed tail of $right$ is one-headed, then the target cannot be history-same to* e*:*

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e1}(m)) \Longrightarrow \mathsf{hsame}(c, \mathsf{e}) \Longrightarrow \mathsf{False}.$$

Proof. Unpack the continuation morphism record. Its relation identifies c with $\mathsf{append}(b, \mathsf{e1}(m))$. If c were history-same to e, append-empty inversion would force $\mathsf{e1}(m)$ itself to be history-same to e, contradicting one-headed constructor separation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism right visible tail target not empty`

Theorem 61.157 (Continuation morphism zero-tail target not empty)**.** *For histories a, b, m and a continuation morphism $morph$: $\mathsf{ContinuationMorphism}(a, b)$, if the displayed tail of $morph$ is zero-headed, then the target cannot be history-same to* e*:*

$$\mathsf{hsame}(\mathsf{tail}(morph), \mathsf{e0}(m)) \Longrightarrow \mathsf{hsame}(b, \mathsf{e}) \Longrightarrow \mathsf{False}.$$

Proof. The zero-headed tail case exposes a zero-headed target presentation $b = \mathsf{e0}(r)$ for some residual history r. If b were history-same to e, constructor separation would force $\mathsf{e0}(r)\mathsf{hsamee}$, which is impossible. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 tail target not empty`

Theorem 61.158 (Continuation morphism composite left visible tail empty absurd)**.** *For histories a, b, c, m, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the left displayed tail is a visible constructor, then the canonical composite displayed tail cannot be history-same to* e*:*

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e0}(m)) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Longrightarrow \mathsf{False}$$

and

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e1}(m)) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Longrightarrow \mathsf{False}.$$

Proof. Unpack both continuation morphism records. The composite tail is the kernel append of the left displayed tail with the right displayed tail. If this append were empty, append-empty inversion would force the left displayed tail itself to be e. This contradicts constructor separation for either $\mathsf{e0}(m)$ or $\mathsf{e1}(m)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left visible tail empty absurd`

Lemma 61.159 (Continuation morphism composite right visible bridge determinacy)**.** *For histories a, b, c, m, let $left$ and $right$ be continuation morphisms from a to b and from b to c, respectively. Choose local history witnesses z, o, r_0, r_1, q such that z is the zero-headed history with payload m, o is the one-headed history with payload m, r_0 names the zero-headed right displayed tail of $right$, r_1 names the one-headed right displayed tail of $right$, and q names the displayed tail of the composite $left \circ right$. The consumer-facing surface for a composite whose right displayed tail is visible is determined by the following finite local rows:*

$$\mathsf{hsame}(r_0, z) \Longrightarrow \mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot,$$
$$\mathsf{hsame}(r_1, o) \Longrightarrow \mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot,$$
$$\mathsf{hsame}(r_1, o) \Longrightarrow \mathsf{hsame}(c, \mathsf{e}) \Longrightarrow \bot.$$

Consequently every consumer-facing use of the composite-tail empty coordinate is certified by either the zero-headed or one-headed composite-tail refusal row of Definition 61.155, and the one-headed target case is certified by Definition 61.156. No consumer-facing bridge coordinate is available for a detached empty composite tail, a detached empty target, a hidden payload classifier, an independent left-tail inspection, host equality of composites, or an off-packet boundary.

Proof. The visible right-tail packet has two displayed constructor branches, and the chosen witnesses expose those branches as bound history names. In the zero-headed branch, the available row is

$$\mathsf{hsame}(r_0, z).$$

The first projection of Definition 61.155, read with witness r_0 for the right displayed tail, witness z for its zero-headed shape, and witness q for the composite displayed tail, is exactly the refusal

$$\mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot.$$

Thus any consumer use that tries to consume the zero-headed right-tail branch as an empty composite tail is blocked by that local refusal. The row exposes no target-empty conclusion and no additional payload coordinate.

In the one-headed branch, the available row is

$$\mathsf{hsame}(r_1, o).$$

The second projection of Definition 61.155, read with witness r_1 for the right displayed tail, witness o for its one-headed shape, and witness q for the composite displayed tail, is exactly the composite-tail refusal

$$\mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot,$$

so every consumer use of the one-headed branch as a composite-tail empty claim is blocked by that row. The same one-headed right-tail row is also the exact input of Definition 61.156, which produces the target-empty refusal

$$\mathsf{hsame}(c, \mathsf{e}) \Longrightarrow \bot.$$

Thus every one-headed target-empty claim is blocked by the cited target-not-empty row.

These three implications exhaust the bridge surface named by the local rows. The first two rows cover all visible right-tail cases leading to the composite-tail empty obstruction, and the third row covers the only target-empty obstruction available from the one-headed right-tail branch. None of

the cited rows has a premise or conclusion mentioning a detached empty composite-tail coordinate, a detached empty target coordinate, a hidden payload classifier, an independent left-tail inspection, host equality of composites, or an off-packet boundary. Such a coordinate therefore has no local row through which a consumer can read the composite right-visible bridge. □

Corollary 61.160 (Continuation morphism composite right visible consumer determinacy)**.** *For histories a, b, c, m, let $left$ and $right$ be continuation morphisms from a to b and from b to c, respectively. Choose local history witnesses z, o, r_0, r_1, q as in Definition 61.159. Once the right-visible local rows*

$$\mathsf{hsame}(r_0, z) \Longrightarrow \mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot,$$
$$\mathsf{hsame}(r_1, o) \Longrightarrow \mathsf{hsame}(q, \mathsf{e}) \Longrightarrow \bot,$$
$$\mathsf{hsame}(r_1, o) \Longrightarrow \mathsf{hsame}(c, \mathsf{e}) \Longrightarrow \bot$$

are fixed, every accepted consumer-facing use of the composite right-visible surface has the same chapter-named finite data. A composite-tail empty use must factor through the zero-headed or one-headed refusal supplied by Definition 61.155; a one-headed target-empty use must factor through Definition 61.156; and any normal form or exactness decision exposed to a consumer is one of those three displayed refusals. Thus two accepted consumers over the same fixed rows cannot choose different finite routes, different right-visible normal forms, or different exactness decisions, and no accepted use can introduce a detached empty composite tail, detached empty target, alternate hidden coordinate, duplicate classifier source, independent left-tail inspection, host equality of composites, or off-packet boundary.

Proof. Apply Definition 61.159. The lemma lists the complete consumer-facing bridge surface for a composite whose right displayed tail is visible. Its first two rows are exactly the zero-headed and one-headed routes from a right-tail constructor row to the refusal of $\mathsf{hsame}(q, \mathsf{e})$, and those two routes are the two projections of Definition 61.155. Hence any accepted consumer use that asks for an empty composite-tail coordinate has one of these two finite routes as its normal form.

The same bridge determinacy lemma also identifies the only target-empty route available from the visible right-tail packet. It is the one-headed row $\mathsf{hsame}(r_1, o)$ followed by Definition 61.156, which refuses $\mathsf{hsame}(c, \mathsf{e})$. With the three displayed rows fixed, this fixes the one-headed target-empty exactness decision as the third displayed refusal, and there is no separate target-empty route in the zero-headed branch.

It remains to rule out competing consumer choices. The cited determinacy lemma states that the bridge exports no detached empty composite-tail coordinate, no detached empty target coordinate, no hidden payload classifier, no independent left-tail inspection, no host equality of composites, and no off-packet boundary. A duplicate classifier source or any further hidden coordinate would therefore have to be an additional row outside the three displayed refusals. Such a row is not a premise or conclusion of the local bridge surface and has no factor through the cited tail-empty or target-not-empty theorems. It is therefore not an accepted route, normal form, or exactness decision for this composite right-visible surface. □

Theorem 61.161 (Continuation morphism composite tail empty endpoint iff)**.** *For histories a, b, c, let $left : a \to b$ and $right : b \to c$ be continuation morphisms. The canonical composite has displayed tail history-same to e if and only if the two adjacent endpoint comparisons hold:*

$$\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Combine the composite tail-empty equivalence with the single-morphism tail-empty endpoint equivalence for the left and right factors. The forward direction projects the two empty

component tails and converts them to endpoint comparisons. The reverse direction converts the two endpoint comparisons back to empty component tails and then rebuilds the empty composite tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail empty endpoint iff`

Theorem 61.162 (Continuation morphism composite endpoint cycle has empty tail)**.** *For histories a, b, c, let $left : a \to b$ and $right : b \to c$ be continuation morphisms. If the composite endpoints satisfy $\mathsf{hsame}(a, c)$, then the canonical composite displayed tail is empty:*

$$\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}).$$

Proof. The composite continuation morphism stores a concrete continuation from a through its displayed tail to c. Transport the result endpoint along $\mathsf{hsame}(a, c)$ to obtain a continuation from a through the same displayed tail back to a. The kernel right-unit uniqueness theorem then forces that displayed tail to be e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint cycle tail empty`

Theorem 61.163 (Continuation morphism composite endpoint cycle tail empty iff)**.** *For histories a, b, c, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$ be continuation morphisms. Then the outer endpoints close exactly when the canonical composite displayed tail is empty:*

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}).$$

Proof. The forward implication is the endpoint-cycle empty-tail theorem. Conversely, if the composite tail is empty, transport the stored continuation relation $\mathsf{Cont}(a, \mathsf{tail}(left \circ right), c)$ across that empty-tail comparison to obtain $\mathsf{Cont}(a, \mathsf{e}, c)$. Determinacy against the right-unit continuation $\mathsf{Cont}(a, \mathsf{e}, a)$ identifies c with a, hence $a\mathsf{hsame}c$ by symmetry. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint cycle tail empty iff`

Theorem 61.164 (Continuation morphism composite source empty tail target iff)**.** *For histories a, b, c, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$ be continuation morphisms. The source is empty exactly when the canonical composite displayed tail is the target:*

$$\mathsf{hsame}(a, \mathsf{e}) \iff \mathsf{hsame}(\mathsf{tail}(left \circ right), c).$$

Proof. If $a\mathsf{hsame}\mathsf{e}$, the stored composite continuation ledger has empty source. The left-unit result law therefore identifies the target with the composite tail, and symmetry gives the displayed tail-target comparison.

Conversely, transport the stored composite continuation ledger along the tail-target comparison to obtain $\mathsf{Cont}(a, c, c)$. Comparing this ledger with the canonical left-unit continuation $\mathsf{Cont}(\mathsf{e}, c, c)$ by right cancellation identifies a with e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp source empty tail target iff`

Lemma 61.165 (Continuation morphism composite endpoint cycle bridge determinacy)**.** *For histories a, b, c, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$ be continuation morphisms, and write $T = \mathsf{tail}(left \circ right)$ for the canonical composite displayed tail.*

The consumer-facing endpoint-cycle bridge for this composite is determined by exactly the two local rows

$$\mathsf{hsame}(a, c) \Longleftrightarrow \mathsf{hsame}(T, \mathsf{e})$$

and

$$\mathsf{hsame}(a, \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(T, c).$$

Thus every accepted route, transport, readback, or handoff through the composite endpoint-cycle surface factors through either Definition 61.163 or Definition 61.164, together with the stored composite continuation ledger $\mathsf{Cont}(a, T, c)$*. No accepted consumer obtains an independent endpoint-cycle flag, an independent source-empty flag, a primitive host composite, a second tail classifier, or any bridge coordinate not listed by these two rows.*

Proof. The displayed composite contributes one tail coordinate, namely T, and one stored continuation ledger, namely $\mathsf{Cont}(a, T, c)$. The endpoint-cycle row of Definition 61.163 says that reading the outer endpoint comparison $\mathsf{hsame}(a, c)$ is exactly the same consumer operation as reading the empty-tail comparison $\mathsf{hsame}(T, \mathsf{e})$. Hence a route that starts from endpoint closure factors by applying the forward implication of that row, while a route that starts from an empty composite tail factors by applying the reverse implication. A transport or handoff along this branch has no further datum: the stored ledger $\mathsf{Cont}(a, T, c)$ is transported across $\mathsf{hsame}(T, \mathsf{e})$ by Definition 4.41 to $\mathsf{Cont}(a, \mathsf{e}, c)$, and determinacy against the right-unit continuation is precisely the reverse direction already recorded by the same row.

The source-empty row of Definition 61.164 gives the second and only other consumer-facing comparison. Reading $\mathsf{hsame}(a, \mathsf{e})$ is exactly the same operation as reading the tail-target comparison $\mathsf{hsame}(T, c)$. Thus a source-empty route factors through the forward implication of that row, and a tail-target readback factors through the reverse implication. The corresponding handoff again uses only the stored composite ledger: transport $\mathsf{Cont}(a, T, c)$ along $\mathsf{hsame}(T, c)$ by Definition 4.41 to obtain $\mathsf{Cont}(a, c, c)$, then compare with the left-unit continuation as in the cited row.

These two cases exhaust the bridge surface. The first row exposes the endpoint-cycle/empty-tail coordinate, and the second row exposes the source-empty/tail-target coordinate. Both use the same displayed tail T and the same stored composite continuation ledger. None of the rows contains a primitive host composite, a second classifier for T, an endpoint-cycle flag separate from $\mathsf{hsame}(T, \mathsf{e})$, a source-empty flag separate from $\mathsf{hsame}(T, c)$, or any detached bridge coordinate. Therefore every accepted consumer-facing route, transport, readback, or handoff factors through the listed local rows, and unlisted bridge coordinates remain unavailable. □

Corollary 61.166 (Continuation morphism composite endpoint cycle consumer determinacy)**.** *For histories* a, b, c*, let* $left : \mathsf{ContinuationMorphism}(a, b)$ *and* $right : \mathsf{ContinuationMorphism}(b, c)$ *be continuation morphisms, and write* $T = \mathsf{tail}(left \circ right)$ *for the canonical composite displayed tail. Once the endpoint-cycle bridge rows of Definition 61.165 are fixed, every accepted consumer-facing use of the composite endpoint-cycle surface has only the chapter-named finite route, normal-form read, and exactness decision:*

$$\mathsf{hsame}(a, c) \Longleftrightarrow \mathsf{hsame}(T, \mathsf{e}),$$
$$\mathsf{hsame}(a, \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(T, c).$$

Thus the endpoint-cycle decision is read through the empty-tail normal form $\mathsf{hsame}(T, \mathsf{e})$*, the source-empty decision is read through the tail-target normal form* $\mathsf{hsame}(T, c)$*, and no accepted consumer can introduce an alternate hidden coordinate or a duplicate classifier source for the composite.*

Proof. By Definition 61.165, the composite exposes exactly one displayed tail coordinate T and one stored composite continuation ledger $\mathsf{Cont}(a, T, c)$ to the endpoint-cycle bridge. The same lemma lists the complete bridge interface as the two rows $\mathsf{hsame}(a, c) \iff \mathsf{hsame}(T, \mathsf{e})$ and $\mathsf{hsame}(a, \mathsf{e}) \iff \mathsf{hsame}(T, c)$, and states that every accepted route, transport, readback, or handoff through the composite factors through one of those rows together with the stored ledger.

The first row is the finite endpoint-cycle route. Its normal-form read is the empty-tail comparison $\mathsf{hsame}(T, \mathsf{e})$, and its exactness decision is precisely the equivalence with the outer endpoint comparison $\mathsf{hsame}(a, c)$ supplied by Definition 61.163. Hence two accepted consumers of the endpoint-cycle decision cannot disagree without leaving that row: both must read the same displayed tail T, transport the same ledger $\mathsf{Cont}(a, T, c)$, and decide the same comparison $\mathsf{hsame}(T, \mathsf{e})$.

The second row is the finite source-empty route. By Definition 61.164, its normal-form read is the tail-target comparison $\mathsf{hsame}(T, c)$, and its exactness decision is precisely the equivalence with $\mathsf{hsame}(a, \mathsf{e})$. Therefore any accepted consumer that claims a source-empty readback also factors through the same displayed tail T and the same ledger $\mathsf{Cont}(a, T, c)$, with no second source from which the classifier could be recomputed.

These two rows exhaust the bridge surface by Definition 61.165. A consumer-facing use that selects a finite route must select one of the two chapter routes; a use that selects a normal form must select either $\mathsf{hsame}(T, \mathsf{e})$ or $\mathsf{hsame}(T, c)$; and a use that selects an exactness decision must select the corresponding displayed equivalence. Since the bridge exports no primitive host composite, no second tail classifier, and no coordinate outside those rows, alternate hidden coordinates and duplicate classifier sources are not accepted. □

Theorem 61.167 (Continuation morphism endpoint cycle boundary)**.** *For histories $a, b, c : \mathsf{BHist}$, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the outer endpoints close as $\mathsf{hsame}(a, c)$, then both displayed tails and both adjacent endpoint pairs collapse:*

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Transport the right continuation relation across the outer endpoint comparison to obtain a return continuation from b to a. Mutual-extension empty-tail inversion applied to the left relation and this transported return relation forces both displayed tails to be empty. Transporting each original relation along its empty tail and comparing with the corresponding right-unit continuation gives the adjacent endpoint comparisons. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint cycle boundary`

Theorem 61.168 (Continuation morphism endpoint cycle boundary iff)**.** *For histories $a, b, c : \mathsf{BHist}$, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. The outer endpoints close exactly when both displayed tails are empty and both adjacent endpoint pairs collapse:*

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The forward implication is the endpoint-cycle boundary theorem. Conversely, compose the adjacent endpoint comparisons $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$ to obtain $\mathsf{hsame}(a, c)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint cycle boundary iff`

Lemma 61.169 (Continuation morphism endpoint-cycle bridge determinacy)**.** *For histories $a, b, c : \mathsf{BHist}$, let $left : a \to b$ and $right : b \to c$ be continuation morphisms, and write $\ell : \mathsf{BHist}$ and*

$r : \mathsf{BHist}$ *for the left and right continuation-tail history witnesses associated to* $left$ *and* $right$*. The consumer-facing endpoint-cycle bridge for this composable pair is determined by the outer endpoint row and the four local boundary rows*

$$\mathsf{hsame}(\ell, \mathsf{e}), \qquad \mathsf{hsame}(r, \mathsf{e}), \qquad \mathsf{hsame}(a, b), \qquad \mathsf{hsame}(b, c).$$

Concretely, every accepted route, transport, readback, or handoff from the outer cycle to boundary data factors through the projection supplied by Definition 61.167, with transport constrained by Definition 61.319, and every accepted route, transport, readback, or handoff from boundary data back to the outer cycle factors through the reverse implication supplied by Definition 61.168, with the same transport constraint. Hence the bridge exports exactly the equivalence

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(\ell, \mathsf{e}) \wedge \mathsf{hsame}(r, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c),$$

and it exports no separate composite-tail coordinate, alternate middle endpoint coordinate, hidden continuation tail, ambient category carrier, host equality, or external handoff.

Proof. Assume first that a consumer reads the bridge from the outer endpoint cycle. The only local theorem with that source row is Definition 61.167. Applying it to $left$, $right$, and the row $\mathsf{hsame}(a, c)$ gives precisely the four displayed boundary rows: the empty left tail, the empty right tail, the source-middle endpoint comparison, and the middle-target endpoint comparison. Any route, transport, readback, or handoff that starts from the outer cycle and reaches consumer boundary data is therefore one of these four projections, or a repackaging of their conjunction, with its transport component governed by Definition 61.319.

Conversely, assume that a consumer starts from the four displayed boundary rows. The reverse direction of Definition 61.168 composes the adjacent endpoint comparisons $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$ and returns exactly the outer endpoint row $\mathsf{hsame}(a, c)$. Thus any readback from the boundary packet to the endpoint cycle factors through that reverse implication.

The forward boundary theorem and the two-way boundary theorem mention no data beyond $\mathsf{hsame}(a, c)$ and the four displayed rows. A proposed coordinate outside this list is not a conjunct produced by the forward theorem and is not a premise consumed by the reverse direction of the equivalence. Such a coordinate has no local factor through which this endpoint-cycle bridge can expose it to a consumer. □

Corollary 61.170 (Continuation morphism endpoint-cycle boundary consumer determinacy)**.** *For histories* $a, b, c : \mathsf{BHist}$*, let* $left : a \to b$ *and* $right : b \to c$ *be continuation morphisms, and let* $\ell : \mathsf{BHist}$ *and* $r : \mathsf{BHist}$ *be their associated left and right continuation-tail witnesses. Once the outer endpoint cycle row and the four boundary rows*

$$\mathsf{hsame}(a, c), \qquad \mathsf{hsame}(\ell, \mathsf{e}), \qquad \mathsf{hsame}(r, \mathsf{e}), \qquad \mathsf{hsame}(a, b), \qquad \mathsf{hsame}(b, c)$$

are fixed, every accepted consumer-facing use of this endpoint-cycle boundary surface has the same chapter-named finite data. In the outward direction the route is the projection of the four boundary rows supplied by Definition 61.167; in the readback direction the route is the reverse implication of Definition 61.168; and every normal form or exactness decision exposed to a consumer is one of those five rows or their displayed conjunction

$$\mathsf{hsame}(a, c) \iff \mathsf{hsame}(\ell, \mathsf{e}) \wedge \mathsf{hsame}(r, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Thus two accepted consumers over the same fixed rows cannot choose different finite routes, different boundary normal forms, or different exactness decisions, and no accepted use can introduce an unlisted hidden coordinate, ambient category carrier, host equality, or external handoff.

Proof. The endpoint-cycle bridge determinacy lemma lists the complete consumer-facing surface for the composable pair. Its outward clause says that any accepted route, transport, readback, or handoff starting from the outer endpoint cycle factors through the projection supplied by Definition 61.167. That theorem has exactly one displayed boundary packet for the fixed pair: the empty left tail row, the empty right tail row, the source-middle endpoint row, and the middle-target endpoint row. Hence every outward consumer route has the same finite target packet and the same boundary normal form.

The same lemma also gives the only accepted readback from boundary data to the outer cycle: it factors through the reverse implication of Definition 61.168. With the four boundary rows fixed, that reverse implication returns exactly the row $\mathsf{hsame}(a, c)$, so every readback normal form and exactness decision about the endpoint cycle is determined by the same displayed equivalence.

It remains to exclude further consumer choices. By Definition 61.169, the bridge exports no separate composite-tail coordinate, no alternate middle endpoint coordinate, no hidden continuation tail, no ambient category carrier, no host equality, and no external handoff. A proposed consumer coordinate outside the five fixed rows is therefore neither a conjunct produced by the forward boundary theorem nor a premise consumed by the reverse readback. It has no local factor through the endpoint-cycle boundary bridge, so it is not an accepted route, normal form, or exactness decision for this surface. □

Theorem 61.171 (Continuation morphism composite tail continuation)**.** *For histories a, b, c, let $left : a \to b$ and $right : b \to c$ be continuation morphisms. The canonical composite tail is itself the continuation composite of the two displayed tails, and it also carries the endpoint continuation from a to c:*

$$\mathsf{Cont}(\mathsf{tail}(left), \mathsf{tail}(right), \mathsf{tail}(left \circ right)) \wedge \mathsf{Cont}(a, \mathsf{tail}(left \circ right), c).$$

Proof. Unpack the two morphism records. The closed composite stores $\mathsf{append}(\mathsf{tail}(left), \mathsf{tail}(right))$ as its displayed tail, giving the first continuation directly. Its record field is exactly the second continuation from a through that composite tail to c. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail cont`

Theorem 61.172 (Continuation morphism tail factorization transport continuation)**.** *For continuation morphisms $left : a \to b$ and $right : b \to c$, if displayed tails l and r classify the two component tails, then the canonical composite tail is the continuation product of l and r:*

$$\mathsf{hsame}(\mathsf{tail}(left), l) \Longrightarrow \mathsf{hsame}(\mathsf{tail}(right), r) \Longrightarrow \mathsf{Cont}(l, r, \mathsf{tail}(left \circ right)).$$

Proof. The composite-tail continuation theorem gives $\mathsf{Cont}(\mathsf{tail}(left), \mathsf{tail}(right), \mathsf{tail}(left \circ right))$. Transport this continuation along the two tail classifier witnesses and keep the composite result endpoint fixed. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail factorization transport cont`

Theorem 61.173 (Continuation morphism composite tail congruence)**.** *For continuation morphisms $left : \mathsf{ContinuationMorphism}(a, b)$, $right : \mathsf{ContinuationMorphism}(b, c)$, $left' : \mathsf{ContinuationMorphism}(a', b')$, and $right' : \mathsf{ContinuationMorphism}(b', c')$, if the two displayed left tails are* hsame *and the two displayed right tails are* hsame*, then both canonical composite tails are recorded by their displayed continuation ledgers and the two composite tails are* hsame*.*

Proof. Use the composite-tail continuation theorem for the two pairs of morphisms to obtain both displayed continuation ledgers. Then unpack the four morphism records. The closed composite tail is the append of the two displayed tails, so the two given tail-sameness witnesses reduce the final equality to reflexivity of the common append expression. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail hsame congruence`

Theorem 61.174 (Continuation morphism composition right factor)**.** *For histories a, b, c and a displayed tail r, let $left : a \to b$ and $composite : a \to c$ be continuation morphisms. If the tail of $composite$ is displayed as the continuation composite of the tail of $left$ with r, then r is the tail of a continuation morphism from b to c.*

Proof. Unpack the left and composite continuation morphisms into their displayed tails and continuation witnesses. The three continuation ledgers reduce the claim to the associativity law for BEDC history append, giving the displayed continuation from b through r to c. □

Lean4 (definition) `BEDC.Derived.CategoryUp.ContinuationMorphism comp right factor`

Theorem 61.175 (Continuation morphism right factor reconstruction)**.** *For histories a, b, c and a displayed tail r, let $left : a \to b$ and $composite : a \to c$ be continuation morphisms. If the tail of $composite$ is displayed as the continuation composite of the tail of $left$ with r, then composing $left$ with the recovered right factor reconstructs the displayed tail of $composite$ up to* hsame*.*

Proof. The recovered right factor has tail r by construction. The closed composition therefore has tail $\mathsf{append}(\mathsf{tail}(left), r)$, and the supplied tail factorization identifies this append with $\mathsf{tail}(composite)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right factor reconstructs`

Theorem 61.176 (Continuation morphism right factor tail uniqueness)**.** *For histories a, b, c and a displayed tail r, let $left : a \to b$ and $composite : a \to c$ be continuation morphisms. If the tail of $composite$ is displayed as the continuation composite of the tail of $left$ with r, and $right : b \to c$ is another continuation morphism, then r is history-same as the displayed tail of $right$.*

Proof. Unpack the three continuation morphism records. The left witness, the tail factorization, and the composite witness reassociate to a continuation witness from b through r to c. This witness and the witness of $right$ have the same source and result, so continuation left cancellation identifies their tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right factor tail hsame`

Theorem 61.177 (Continuation morphism composite middle-tail readback)**.** *For histories a, b, b', c, let $left : a \to b$, $right : b' \to c$, and $composite : a \to c$ be continuation morphisms. If the tail of $composite$ is displayed as the continuation composite of the tails of $left$ and $right$, then*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}\big(\mathsf{append}(\mathsf{tail}(left), \mathsf{tail}(right)), \mathsf{tail}(composite)\big).$$

Proof. Unpack the left and composite continuation morphism records. The left witness, the displayed tail factorization, and the composite witness reassociate to a continuation witness from b through the tail of $right$ to c. Comparing this recovered witness with $right$ by continuation right cancellation gives $\mathsf{hsame}(b, b')$. The displayed tail factorization itself gives the readback of the appended tail against the composite tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp middle tail readback`

Definition 61.178 (Continuation morphism composite left factor)**.** *For histories a, b, c and a displayed tail l, let right $: b \to c$ and composite $: a \to c$ be continuation morphisms. If the tail of composite is displayed as the continuation composite of l with the tail of right, then l is the tail of a continuation morphism from a to b.*

Proof. Unpack the right and composite continuation morphism records. The displayed tail composite and the composite morphism witness reduce the target to the kernel composite-left-factor theorem for Cont, which transports the reflexive left-tail continuation across the recovered middle endpoint. □

Lean4 (definition) `BEDC.Derived.CategoryUp.ContinuationMorphism comp left factor`

Theorem 61.179 (Continuation morphism left factor reconstruction)**.** *For histories a, b, c and a displayed tail l, let right $: b \to c$ and composite $: a \to c$ be continuation morphisms. If the tail of composite is displayed as the continuation composite of l with the tail of right, then composing the recovered left factor with right reconstructs the displayed tail of composite up to* hsame*.*

Proof. The recovered left factor has tail l by construction. The closed composition therefore has tail $\mathsf{append}(l, \mathsf{tail}(right))$, and the supplied tail factorization identifies this append with $\mathsf{tail}(composite)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left factor reconstructs`

Theorem 61.180 (Continuation morphism left factor tail uniqueness)**.** *For histories a, b, c and a displayed tail l, let right $: b \to c$ and composite $: a \to c$ be continuation morphisms. If the tail of composite is displayed as the continuation composite of l with the tail of right, and $left : a \to b$ is another continuation morphism, then l is history-same as the displayed tail of $left$.*

Proof. Unpack the continuation morphism records. The right witness, the displayed tail factorization, and the composite witness recover a continuation witness from a through l to b. This witness and the witness of $left$ share the same source and result, so continuation left cancellation identifies their tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left factor tail hsame`

Theorem 61.181 (Continuation morphism identity tail)**.** *For histories a, b and a continuation morphism $m : a \to b$, composing m with the empty identity tail on either side leaves its displayed tail history-same to the tail of m:*

$$\mathsf{hsame}\big(\mathsf{tail}(\mathsf{id}_a \circ m), \mathsf{tail}(m)\big) \wedge \mathsf{hsame}\big(\mathsf{tail}(m \circ \mathsf{id}_b), \mathsf{tail}(m)\big).$$

Proof. Unpack the continuation morphism into its tail and continuation witness. The left identity composite has tail $\mathsf{append}(\mathsf{Empty}, \mathsf{tail})$, and the right identity composite has tail $\mathsf{append}(\mathsf{tail}, \mathsf{Empty})$. The two BEDC append unit laws identify both with the original tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism identity tail hsame`

Theorem 61.182 (Continuation morphism associative composition closure)**.** *For histories a, b, c, d, any continuation morphisms $a \to b$, $b \to c$, and $c \to d$ admit both associated composite morphisms $a \to d$, and the two composite tails are internally the same.*

Proof. Unpack the three morphism records into their tails and continuation witnesses. The left-associated composite tail is $\mathsf{append}(\mathsf{append}(f, g), h)$; the right-associated composite tail is $\mathsf{append}(f, \mathsf{append}(g, h))$. The continuation witnesses reduce to associativity of BEDC history append, and the tail comparison is exactly the same associativity equality. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp assoc closed`

Theorem 61.183 (Continuation morphism associative composite tail)**.** *For histories a, b, c, d, any continuation morphisms $a \to b$, $b \to c$, and $c \to d$ have left-associated and right-associated canonical composite tails that are history-same:*

$$\mathsf{hsame}\left(\mathsf{tail}((first \circ second) \circ third), \mathsf{tail}(first \circ (second \circ third))\right).$$

Proof. Unpack the three continuation morphisms into their displayed tails. The two canonical composite tails reduce to $\mathsf{append}(\mathsf{append}(f, g), h)$ and $\mathsf{append}(f, \mathsf{append}(g, h))$, which are identified by BEDC history append associativity. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp assoc tail hsame`

Theorem 61.184 (Category hom carrier morphism determinacy)**.** *For the unary-continuation category, if f and g are both hom-carriers from a to b, then* $\mathsf{hsame}(f, g)$*.*

Proof. Unpack both hom-carrier witnesses. Their continuation fields have the same source a and result b, so continuation left-context cancellation identifies the two morphism histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier morphism deterministic`

Theorem 61.185 (Category hom carrier endpoint-hsame morphism determinacy)**.** *For the unary-continuation category, if the source endpoints classify together and the target endpoints classify together, then any two displayed hom-carrier morphisms over those classified endpoints also classify together.*

Proof. Case-reduce the source and target hsame witnesses. The two hom-carrier witnesses then share exactly the same endpoints, so Definition 61.184 identifies their morphism histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier endpoint hsame morphism deterministic`

Theorem 61.186 (Category hom carrier target determinacy)**.** *For the unary-continuation category, if a fixed morphism history f is a hom-carrier from a to b and also from a to c, then* $\mathsf{hsame}(b, c)$*.*

Proof. Unpack both hom-carrier witnesses. Their continuation fields have the same source a and morphism tail f, so continuation result determinacy identifies the two target histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier target deterministic`

Theorem 61.187 (Category empty-source morphism target determinacy)**.** *For the unary-continuation category, if two hom-carriers share the empty source and their morphism histories are the same up to* hsame*, then their target histories are the same up to* hsame*:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, f) \to \mathsf{CategoryHomCarrier}(\mathsf{e}, c, g) \to \mathsf{hsame}(f, g) \to \mathsf{hsame}(b, c).$$

Proof. Transport the second empty-source hom-carrier across the morphism sameness so that both hom-carriers use the same morphism history. Category hom-carrier target determinacy then identifies the two targets. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source morphism target deterministic`

Theorem 61.188 (Category empty-source target morphism swap)**.** *For histories* $b, f : \mathsf{BHist}$*, an empty-source hom-carrier is unchanged when its target and morphism fields are exchanged:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, f) \iff \mathsf{CategoryHomCarrier}(\mathsf{e}, f, b).$$

Proof. Empty-source exactness rewrites the left side as unary membership of b together with $\mathsf{hsame}(f, b)$. Transport unary membership across this sameness to obtain unary membership of f, reverse the endpoint classifier, and apply empty-source exactness again. The converse is the same argument with b and f exchanged. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source target morphism swap iff`

Theorem 61.189 (Category empty-target source morphism swap)**.** *For histories* $a, f : \mathsf{BHist}$*, a hom-carrier into the empty target is unchanged when its source and morphism fields are exchanged:*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{e}, f) \iff \mathsf{CategoryHomCarrier}(f, \mathsf{e}, a).$$

Proof. Empty-target exactness rewrites the left side as the pair $\mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e})$. Swapping the two components gives the empty-target classifier data for the exchanged hom-carrier. The converse is the same component swap in the other direction. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target source morphism swap iff`

Theorem 61.190 (Category empty-target morphism source determinacy)**.** *For the unary-continuation category, if two hom-carriers share the empty target and their morphism histories are the same up to* hsame*, then their source histories are the same up to* hsame*:*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{e}, f) \to \mathsf{CategoryHomCarrier}(b, \mathsf{e}, g) \to \mathsf{hsame}(f, g) \to \mathsf{hsame}(a, b).$$

Proof. Transport the second empty-target hom-carrier across the morphism sameness so that both hom-carriers use the same morphism history. Category hom-carrier source determinacy then identifies the two sources. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty target morphism source deterministic`

Corollary 61.191 (Category empty-source swap consumer determinacy)**.** *For histories* $b, f, c, g : \mathsf{BHist}$*, suppose an accepted empty-source hom-carrier fixes the target and morphism rows by*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, b, f).$$

If a consumer-facing use of the same packet reads the target row b *and the morphism row* f *through empty-target hom-carriers,*

$$\mathsf{CategoryHomCarrier}(c, \mathsf{e}, b) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(g, \mathsf{e}, f),$$

then the two consumer source coordinates are forced to be the same:

$$\mathsf{hsame}(c, g).$$

Consequently the empty-source target–morphism swap of Definition 61.188 has only the finite route, normal-form read, and exactness decision already displayed by the chapter: an accepted consumer cannot introduce an alternate hidden coordinate or a duplicate classifier source for the swapped target and morphism fields.

Proof. Apply Definition 61.188 to the displayed empty-source carrier. The same packet also admits the swapped read

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, f, b).$$

The empty-source exactness row used in that swap reads the original carrier as unary membership of b together with the endpoint classifier $\mathsf{hsame}(f, b)$; by symmetry this gives $\mathsf{hsame}(b, f)$. Now apply Definition 61.190 to the two empty-target carriers $\mathsf{CategoryHomCarrier}(c, \mathsf{e}, b)$ and $\mathsf{CategoryHomCarrier}(g, \mathsf{e}, f)$, using $\mathsf{hsame}(b, f)$ as the morphism comparison. It identifies their source histories, giving $\mathsf{hsame}(c, g)$. Thus every accepted consumer read factors through the displayed empty-source swap row and the two empty-target readback rows; no unlisted coordinate remains from which a second source classifier could be read. □

Theorem 61.192 (Category hom carrier source determinacy)**.** *For the unary-continuation category, if a fixed morphism history f is a hom-carrier from a to c and also from b to c, then* $\mathsf{hsame}(a, b)$*.*

Proof. Unpack both hom-carrier witnesses. Their continuation fields have the same morphism tail f and result c, so continuation right-context cancellation identifies the two source histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier source deterministic`

Theorem 61.193 (Continuation morphism nonempty tail shape cases)**.** *For histories $a, b : \mathsf{BHist}$ and a continuation morphism $m : \mathsf{ContinuationMorphism}(a, b)$, a displayed tail that is not internally empty must be visible in one of the two history constructors:*

$$(\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \to \mathsf{False}) \Longrightarrow (\exists t : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E0}(t)) \vee (\exists t : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E1}(t)).$$

Proof. Unpack the continuation morphism only to expose its displayed tail, then case-analyze that history. The empty branch contradicts the nonempty hypothesis by reflexivity of hsame at e. The zero-headed and one-headed branches return the corresponding visible payload. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail nonempty shape cases`

Theorem 61.194 (Category associative empty result components)**.** *For composable category hom-carriers f, g, h with both associated composite ledgers displayed as $left$ and $right$, if the same source and target endpoints also carry the empty hom-carrier, then both associated composite histories are* hsame *to* e*, and the source and target endpoints are* hsame*-equal.*

Proof. Apply displayed associativity determinacy using the empty hom-carrier as the displayed comparison target. This identifies both bracketed composite histories with e. Empty identity hom-carrier exactness reads the same displayed empty carrier back as the endpoint equality $a\mathsf{hsame}d$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp assoc empty result components`

Theorem 61.195 (Continuation morphism empty-source one-headed target tail iff)**.** *For histories $r, k : \mathsf{BHist}$ and a continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{e}, \mathsf{E1}(r))$, the displayed tail of m has one-headed payload k exactly when the target payload is k:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \Longleftrightarrow \mathsf{hsame}(r, k).$$

Proof. The empty-source tail target exactness identifies $\mathsf{tail}(m)$ with $\mathsf{E1}(r)$. In the forward direction, compose this equality with the displayed tail equality and invert the one-headed constructor. Conversely, lift $r\mathsf{hsame}k$ through the one-headed constructor and compose with the tail-target exactness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism empty source e1 target tail iff`

Theorem 61.196 (Continuation morphism empty-source zero-headed target tail iff)**.** *For histories $r, k : \mathsf{BHist}$ and a continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{e}, \mathsf{E0}(r))$, the displayed tail of m has zero-headed payload k exactly when the target payload is k:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \iff \mathsf{hsame}(r, k).$$

Proof. The empty-source tail target exactness identifies $\mathsf{tail}(m)$ with $\mathsf{E0}(r)$. In the forward direction, compose this equality with the displayed tail equality and invert the zero-headed constructor. Conversely, lift $r\mathsf{hsame}k$ through the zero-headed constructor and compose with the tail-target exactness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism empty source e0 target tail iff`

Theorem 61.197 (Category empty-source one-headed target payload exactness)**.** *For histories r, k, an empty-source hom-carrier into the one-headed target $\mathsf{E1}(r)$ with displayed one-headed morphism $\mathsf{E1}(k)$ is exactly unary membership of the target payload together with payload sameness:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, \mathsf{E1}(r), \mathsf{E1}(k)) \iff \mathsf{UnaryHistory}(r) \wedge \mathsf{hsame}(k, r).$$

Proof. Apply empty-source hom-carrier exactness to read back unary membership of $\mathsf{E1}(r)$ and sameness $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(r))$. Unary one-headed inversion and one-headed sameness inversion expose the payload data. Conversely, unary closure under $\mathsf{E1}$ and one-headed congruence rebuild the target carrier and the displayed morphism-target sameness, so empty-source exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source e1 target payload iff`

61.2 The certificate

Definition 61.198 (Category source specification)**.** *The source is the licensed object carrier and each licensed hom-carrier.*

Definition 61.199 (Category pattern specification)**.** *The pattern is morphism expression formation from identities and composable pairs.*

Definition 61.200 (Category classifier specification)**.** *The classifier is object sameness for objects and hom-carrier sameness for morphisms, indexed by source and target.*

Definition 61.201 (Category stability certificate)**.** *The stability certificate consists of:*

1. *object classifier reflexivity and transitivity*
2. *morphism classifier reflexivity and transitivity*
3. *identity morphism stability*

4. *composition closure*
5. *associativity under morphism classifier*
6. *left and right identity laws*
7. *composition congruence*

↪ variant BEDC.Derived.CategoryUp.ContinuationMorphism identity comp closure

Definition 61.202 (Category ledger policy)**.** *Record object ledgers, hom-carrier ledgers, and composition trees.*

Definition 61.203 ($\mathsf{Cat}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Cat}^{\uparrow}$*.*

Theorem 61.204 (Category left zero-headed continuation result cases)**.** *If* $\mathsf{Cont}(\mathsf{E0}(h), k, \mathsf{E0}(r))$*, then either* $k = \mathsf{e}$ *and* $\mathsf{hsame}(h, r)$*, or* $k = \mathsf{E0}(k_0)$ *for some* k_0 *and* $\mathsf{Cont}(\mathsf{E0}(h), k_0, r)$*.*

Lean4 ✓ BEDC.Derived.CategoryUp.category cont left e0 result cases

Proof. Classify the continuation history k. The empty branch identifies the inner histories. The zero branch exposes the predecessor of k and removes the matching outer constructor from the continuation result. The one branch would identify a zero-headed history with a one-headed history, which is impossible by generated-history no-confusion. □

Theorem 61.205 (Category left one-headed continuation result cases)**.** *If* $\mathsf{Cont}(\mathsf{E1}(h), k, \mathsf{E1}(r))$*, then either* $k = \mathsf{e}$ *and* $\mathsf{hsame}(h, r)$*, or* $k = \mathsf{E1}(k_1)$ *for some* k_1 *and* $\mathsf{Cont}(\mathsf{E1}(h), k_1, r)$*.*

Lean4 ✓ BEDC.Derived.CategoryUp.category cont left e1 result cases

Proof. Classify the continuation history k. The empty branch removes the matching one-headed constructor from the displayed result. The zero branch is ruled out by constructor disjointness. The one branch exposes the predecessor of k and removes the matching outer constructor from the continuation result. □

Theorem 61.206 (Category hom-carrier zero-headed source impossibility)**.** *For histories* $a, target, f$*, no hom-carrier can have source* $\mathsf{E0}(a)$*:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E0}(a), target, f) \to \mathsf{False}.$$

Proof. Unpack the hom-carrier witness. Its source-carrier field says that $\mathsf{E0}(a)$ is unary, contradicting the BEDC unary-history separation theorem for zero-headed extensions. □

Lean4 ✓ BEDC.Derived.CategoryUp.CategoryHomCarrier e0 source absurd

Theorem 61.207 (Category hom-carrier zero-headed target impossibility)**.** *For histories* $source, a, morph$*, no hom-carrier can have target* $\mathsf{E0}(a)$*:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{E0}(a), morph) \to \mathsf{False}.$$

Proof. Unpack the hom-carrier witness. Its target-carrier field says that $\mathsf{E0}(a)$ is unary, contradicting the BEDC unary-history separation theorem for zero-headed extensions. □

Lean4 ✓ BEDC.Derived.CategoryUp.CategoryHomCarrier e0 target absurd

Theorem 61.208 (Category hom-carrier zero-headed morphism impossibility)**.** *For histories source, target, a, no hom-carrier can have morphism* $\mathsf{E0}(a)$*:*

$$\mathsf{CategoryHomCarrier}(source, target, \mathsf{E0}(a)) \to \mathsf{False}.$$

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e0 morphism absurd`

Proof. Unpack the hom-carrier witness. Its morphism-carrier field says that $\mathsf{E0}(a)$ is unary, contradicting zero-headed unary-history separation. □

Theorem 61.209 (Category hom-carrier zero-headed component impossibility)**.** *For histories source, target, morph, a category hom-carrier has no zero-headed source, target, or morphism component:*

$$\mathsf{CategoryHomCarrier}(source, target, morph) \Longrightarrow (\exists z,\ source = \mathsf{E0}(z)) \vee (\exists z,\ target = \mathsf{E0}(z)) \vee (\exists z,\ morph = \mathsf{E0}(z)$$

Proof. Unpack the hom-carrier into its source, target, and morphism unary-history fields. A zero-headed source, target, or morphism rewrites the corresponding field to a unary witness for a zero-headed extension, contradicting unary separation for zero-headed histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier zero headed component absurd`

Theorem 61.210 (Category hom-carrier composite zero-headed result impossibility)**.** *For histories* $a, b, c, f, g, z : \mathsf{BHist}$*, if* f *is a hom-carrier from* a *to* b*,* g *is a hom-carrier from* b *to* c*, and the displayed composite continuation has zero-headed result*

$$\mathsf{Cont}(f, g, \mathsf{E0}(z)),$$

then contradiction follows.

Proof. Composition closure makes $\mathsf{E0}(z)$ a hom-carrier morphism from a to c. Its morphism-carrier field is therefore a unary-history witness for a zero-headed history, contradicting unary separation for zero-headed histories. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp zero result absurd`

Theorem 61.211 (Category hom-carrier left one-headed result cases)**.** *If*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), k)$$

then either $k = \mathsf{e}$*,* a *is unary, and* $\mathsf{hsame}(a, r)$*, or there is a history* k_0 *such that* $k = \mathsf{E1}(k_0)$*,* k_0 *is unary, and*

$$\mathsf{Cont}(\mathsf{E1}(a), k_0, r).$$

Proof. Unpack the hom-carrier into unary source, unary morphism-tail, and continuation fields. Classify the tail k. The empty branch cancels the visible one constructor in the result and inverts the source unary witness. The zero-headed tail branch is impossible by constructor no-confusion. The one-headed tail branch removes the outer constructor from the continuation result and inverts the morphism-tail unary witness to obtain the unary carrier for k_0. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier left e1 result cases`

Theorem 61.212 (Category hom-carrier one-headed source and target morphism exactness)**.** *For histories $a, r, morph$, a hom-carrier from $\mathsf{E1}(a)$ to $\mathsf{E1}(r)$ is exactly either the empty morphism identifying the two exposed endpoints, or a one-headed morphism whose tail continues the visible source to the exposed target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), morph) \iff (morph = \mathsf{e} \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(a, r)) \vee (\exists k : \mathsf{BHist}.\ morph =$$

Proof. The forward direction applies the visible-source and visible-target case split: the empty branch gives the exposed endpoint classifier, while the one-headed branch gives a tail k and a continuation ledger $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. Unary inversion on the source carrier supplies $\mathsf{UnaryHistory}(a)$. Conversely, the empty branch is rebuilt by the right-unit continuation after transport along $\mathsf{hsame}(a, r)$; the one-headed branch uses unary continuation closure for the exposed target and the one-step continuation rule for the visible morphism. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target morphism iff`

Theorem 61.213 (Category nonempty one-headed endpoint morphism cases)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}($ *if morph is not* hsame *to* e*, then* $morph = \mathsf{E1}(k)$ *for some unary k, a and r are unary, and* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Proof. Split the one-headed source and target morphism cases; the empty branch contradicts the nonempty hypothesis, and the visible branch exposes the tail continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism cases`

Theorem 61.214 (Category nonempty one-headed endpoint morphism target descent)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E}$ *if morph is not* hsame *to* e*, then there is a history k such that* $morph = \mathsf{E1}(k)$ *and the exposed tail itself is a hom-carrier* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$*.*

Proof. Apply the nonempty one-headed endpoint morphism cases. The exposed data gives $morph = \mathsf{E1}(k)$, unary witnesses for a, k, r, and $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. Reassembling these fields gives the descended hom-carrier from $\mathsf{E1}(a)$ to r with morphism k. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism target descent`

Theorem 61.215 (Category nonempty one-headed endpoint morphism unique descent)**.** *Under the same hypotheses, the descended tail is unique up to* hsame*: there is a history k with* $morph = \mathsf{E1}(k)$*,* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$*, and every other displayed one-headed descent* $morph = \mathsf{E1}(k')$ *with* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k')$ *satisfies* $k\mathsf{hsame}k'$*.*

Proof. First use target descent to obtain the displayed tail k and its descended hom-carrier. If another tail k' displays the same morphism as $\mathsf{E1}(k')$, constructor injectivity for $\mathsf{E1}$ identifies k and k', hence gives $k\mathsf{hsame}k'$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism descent unique`

Theorem 61.216 (Category nonempty one-headed source morphism target cases)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a),$ *if morph is not* hsame *to* e*, then there are histories k, r such that* $morph = \mathsf{E1}(k)$*,* $target = \mathsf{E1}(r)$*, a, k, r are unary, and* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Proof. Split the one-headed source morphism cases. The empty-morphism branch contradicts the nonempty hypothesis. The visible branch already exposes the one-headed target, the morphism tail, the three unary witnesses, and the continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source nonempty morphism target cases`

Theorem 61.217 (Category nonempty one-headed source morphism target iff)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), ta$ *together with morph not* hsame *to* e*, the carrier data is equivalent to:*

$$\exists k, r : \mathsf{BHist}.\ morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r)$$

Proof. The forward direction is the nonempty one-headed source morphism target case split. Conversely, the displayed histories k, r, the one-headed equalities, the three unary witnesses, and the continuation ledger rebuild the concrete hom-carrier; the morphism is visibly one-headed, so it cannot be hsame to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source nonempty morphism target iff`

Theorem 61.218 (Category nonempty one-headed source morphism target carrier cases)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), target, morph)$*, if morph is not* hsame *to* e*, then there are histories* k, r *such that* $morph = \mathsf{E1}(k)$*,* $target = \mathsf{E1}(r)$*, and* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$*.*

Proof. Apply the nonempty one-headed source morphism target case theorem. It exposes histories k, r, the equalities $morph = \mathsf{E1}(k)$ and $target = \mathsf{E1}(r)$, unary witnesses for a, k, r, and the continuation ledger $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. These fields are exactly the concrete hom-carrier clauses for $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source nonempty morphism target carrier cases`

Theorem 61.219 (Category nonempty one-headed source morphism target carrier uniqueness)**.** *For* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), target, morph)$*, if morph is not* hsame *to* e*, then there are histories* k, r *such that* $morph = \mathsf{E1}(k)$*,* $target = \mathsf{E1}(r)$*, and* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$*. Moreover, any other displayed decomposition with* $morph = \mathsf{E1}(k')$*,* $target = \mathsf{E1}(r')$*, and* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r', k')$ *satisfies* $k\,\mathsf{hsame}\,k'$ *and* $r\,\mathsf{hsame}\,r'$*.*

Proof. Apply the nonempty one-headed source morphism target case theorem. It exposes the one-headed morphism and target, the three unary tail witnesses, and the descended continuation ledger, which reassemble the descended hom-carrier. Uniqueness follows by constructor injectivity for the two displayed E1 equalities. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source nonempty morphism target carrier unique`

Theorem 61.220 (Category nonempty one-headed target morphism source cases)**.** *For* $\mathsf{CategoryHomCarrier}(source$ *if morph is not* hsame *to* e*, then either* $source = \mathsf{e}$*,* $morph = \mathsf{E1}(r)$*, and* r *is unary, or there are histories* a, k *such that* $source = \mathsf{E1}(a)$*,* $morph = \mathsf{E1}(k)$*,* a, k, r *are unary, and* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Proof. Split on the source constructor. A zero-headed source contradicts the hom-carrier unary field. The empty-source branch inverts the continuation into the one-headed target and identifies the morphism with $\mathsf{E1}(r)$. The one-headed-source branch applies the nonempty one-headed endpoint case split to expose the morphism tail and continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target nonempty morphism source cases`

Theorem 61.221 (Category nonempty one-headed target morphism source carrier uniqueness)**.** *For* $\mathsf{CategoryHomCarrier}(source, \mathsf{E1}(r), morph)$*, if morph is not* hsame *to* e*, then either* $source = \mathsf{e}$*,* $morph = \mathsf{E1}(r)$*, and the visible target tail is unique under every one-headed display of morph; or there are histories* a, k *such that* $source = \mathsf{E1}(a)$*,* $morph = \mathsf{E1}(k)$*, and* $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$*. In the visible-source branch, any other one-headed source and morphism display with a descended hom-carrier has* $a\mathsf{hsame}a'$ *and* $k\mathsf{hsame}k'$*.*

Proof. Split the nonempty one-headed target morphism source cases. In the empty-source branch, the displayed equality $morph = \mathsf{E1}(r)$ makes every other one-headed display equal on tails by constructor injectivity. In the visible-source branch, the exposed unary witnesses and continuation ledger reassemble the descended hom-carrier, and constructor injectivity on $source = \mathsf{E1}(a)$ and $morph = \mathsf{E1}(k)$ gives the stated tail uniqueness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target nonempty morphism source carrier unique`

Theorem 61.222 (Category nonempty one-headed target morphism source iff)**.** *For* $\mathsf{CategoryHomCarrier}(source, \mathsf{E}$ *together with morph not* hsame *to* e*, the source is classified exactly by*

$$(source = \mathsf{e} \wedge morph = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r)) \vee \exists a, k : \mathsf{BHist}.\ source = \mathsf{E1}(a) \wedge morph = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(a) \wedge$$

Proof. The forward direction is the nonempty one-headed target morphism source case split. In the empty-source branch, the empty-source carrier equivalence rebuilds $\mathsf{CategoryHomCarrier}(\mathsf{e}, \mathsf{E1}(r), \mathsf{E1}(r))$. In the visible source branch, the displayed one-headed source, one-headed morphism, unary witnesses, and continuation ledger rebuild the concrete hom-carrier. In both reverse branches the displayed morphism is one-headed, hence not hsame to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target nonempty morphism source iff`

Theorem 61.223 (Category one-headed target and morphism source cases iff)**.** *For* $\mathsf{CategoryHomCarrier}(source, \mathsf{E1}$ *the source is classified by the empty-source unit branch or by a visible one-headed source:*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(source, \mathsf{E1}(r), \mathsf{E1}(k)) \\
&\iff (source = \mathsf{e} \wedge k\mathsf{hsame}r \wedge \mathsf{UnaryHistory}(r)) \\
&\quad \vee \exists a : \mathsf{BHist}.\ source = \mathsf{E1}(a) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \\
&\quad\quad \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).
\end{aligned}$$

Proof. For the forward direction, invert the one-headed morphism target carrier and split the concrete source constructor. The empty branch is the unit-source continuation case, the zero-headed branch contradicts unary history, and the one-headed branch exposes the source tail and continuation ledger. For the reverse direction, the empty branch builds the empty-source hom-carrier, while the visible branch rebuilds the one-headed source carrier from the displayed unary witnesses and continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target e1 morphism source cases iff`

Theorem 61.224 (Category nonempty morphism source cases)**.** *For* $\mathsf{CategoryHomCarrier}(source, target, morph)$*, if morph is not* hsame *to* e*, then either* $source = \mathsf{e}$*, target is unary, and* $morph\mathsf{hsame}target$*, or there are histories* a, k, r *such that* $source = \mathsf{E1}(a)$*,* $morph = \mathsf{E1}(k)$*,* $target = \mathsf{E1}(r)$*,* a, k, r *are unary, and* $\mathsf{Cont}(\mathsf{E1}(a), k, r)$*.*

Proof. Split on the source constructor. The empty-source branch is exactly the empty-source hom-carrier inversion. A zero-headed source contradicts the unary source field. In the one-headed branch, an empty displayed morphism contradicts the nonempty hypothesis and a zero-headed morphism contradicts the unary morphism field; the remaining one-headed morphism exposes the one-headed target and tail continuation by the one-headed morphism target cases theorem. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier nonempty morphism source cases`

Theorem 61.225 (Category nonempty morphism endpoint cases)**.** *For* $\mathsf{CategoryHomCarrier}(source, target, morph)$ *if morph is not* hsame *to* e*, then either the source is empty and the visible morphism is exactly the visible target tail, or all three endpoints are one-headed and the continuation ledger remains visible at that boundary:*

$$(source = \mathsf{e} \wedge \exists k.\ morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(k))$$
$$\vee(\exists a, k, r.\ source = \mathsf{E1}(a) \wedge morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}($$

Proof. Apply the nonempty morphism source-case split. In the empty-source branch, empty-source inversion identifies *morph* with *target*; the nonempty hypothesis excludes the empty morphism and unary generation excludes a zero-headed morphism, leaving the one-headed endpoint k. In the one-headed source branch, the existing exposed continuation $\mathsf{Cont}(\mathsf{E1}(a), k, r)$ steps once to the displayed boundary $\mathsf{Cont}(\mathsf{E1}(a), \mathsf{E1}(k), \mathsf{E1}(r))$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier nonempty morphism endpoint cases`

Theorem 61.226 (Category hom-carrier one-headed endpoint morphism alignment)**.** *For hom-carriers between the same one-headed source and one-headed target, the displayed morphisms align: they are both empty, or both one-headed with* hsame *exposed morphism tails.*

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target morphism alignment`

Proof. Category morphism determinacy gives $\mathsf{hsame}(m, n)$ for the two displayed morphisms. Split one hom-carrier by the one-headed source and target result cases. In the empty branch, morphism sameness forces the second morphism to be empty. In the visible branch, the same morphism sameness transports the second morphism into a one-headed shape, and constructor inversion gives hsame between the exposed morphism tails. □

Theorem 61.227 (Category nonempty one-headed endpoint morphism alignment)**.** *For two hom-carriers between* $\mathsf{E1}(a)$ *and* $\mathsf{E1}(r)$*, if the first morphism is not* hsame *to* e*, then both morphisms are one-headed and their exposed tails are* hsame*.*

Proof. Apply endpoint morphism alignment; the both-empty branch contradicts the nonempty witness, leaving the visible-tail alignment branch. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism alignment`

Theorem 61.228 (Category nonempty one-headed endpoint morphism tail-cont alignment)**.** *For two hom-carriers between* $\mathsf{E1}(a)$ *and* $\mathsf{E1}(r)$*, if the first morphism is not* hsame *to* e*, then both morphisms are one-headed. Their exposed tails are unary, both tails continue* $\mathsf{E1}(a)$ *to* r*, and the two tails are* hsame*:*

$$\exists k, l : \mathsf{BHist}.\ m = \mathsf{E1}(k) \wedge n = \mathsf{E1}(l) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(l) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r) \wedge \mathsf{Cont}(\mathsf{E1}(a), l, r) \wedge \mathsf{hsame}(k,$$

Proof. First apply nonempty endpoint morphism alignment to expose one-headed tails k and l with $\mathsf{hsame}(k, l)$. Then apply the one-headed source and target morphism exactness theorem separately to the two displayed hom-carriers. The empty branch is impossible after the displayed one-headed equality; the visible branch supplies the unary tail witness and continuation ledger on each side. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism tail cont alignment`

Theorem 61.229 (Category nonempty one-headed endpoint morphism tail carrier alignment)**.** *For two hom-carriers between* $\mathsf{E1}(a)$ *and* $\mathsf{E1}(r)$*, if the first morphism is not* hsame *to* e*, then both displayed morphisms descend to one-headed tails whose tail hom-carriers share the same source and target:*

$$\exists k, l : \mathsf{BHist}.\ m = \mathsf{E1}(k) \wedge n = \mathsf{E1}(l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, l) \wedge \mathsf{hsame}(k, l).$$

Proof. Apply the nonempty tail-continuation alignment theorem. It supplies the one-headed displays, the two unary tail witnesses, the two continuations $\mathsf{Cont}(\mathsf{E1}(a), k, r)$ and $\mathsf{Cont}(\mathsf{E1}(a), l, r)$, and the tail equality $\mathsf{hsame}(k, l)$. Repack each unary tail and continuation with the original source carrier, deriving the target carrier r by unary continuation closure. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target nonempty morphism tail carrier alignment`

Corollary 61.230 (Category source specification endpoint determinacy)**.** *Let* $a, r, morph$ *be histories and suppose a* $\mathsf{Cat}^{\uparrow}$ *source specification use, in the sense of Definition 61.198, is accepted through the licensed one-headed endpoint hom-carrier row*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), morph)$$

with $morph$ *not* hsame *to* e*. Then the finite source route forced on that use is determined: there is a history* k *such that*

$$morph = \mathsf{E1}(k) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k).$$

For any second consumer readback, normal-form choice, or exactness decision in the same source specification, if it displays a one-headed descent

$$morph = \mathsf{E1}(k') \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k'),$$

then $k \mathsf{hsame} k'$*. Hence the accepted source specification use cannot introduce a hidden coordinate, a second finite route, or a duplicate classifier source beyond the visible row named by the chapter.*

Proof. By Definition 61.198, the source data consists of the licensed object carrier together with each licensed hom-carrier. Thus an accepted use at the displayed endpoint must be read through the concrete hom-carrier row

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), morph),$$

rather than through an external coordinate. The nonempty hypothesis rejects the empty exactness decision, so the only admissible visible branch is the one-headed descent branch. Applying Definition 61.215 gives a displayed tail k with $morph = \mathsf{E1}(k)$ and the descended licensed row $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k)$.

Now take any second accepted readback, normal-form choice, or exactness decision for the same source specification. To remain accepted under Definition 61.198, it must again present a licensed hom-carrier row; if it displays $morph = \mathsf{E1}(k')$ and $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), r, k')$, the universal clause of Definition 61.215 gives $k\mathsf{hsame}k'$. Therefore every accepted consumer-facing use factors through the same descended tail up to hsame, and no unlisted coordinate or duplicate classifier source survives the displayed finite row list. □

Theorem 61.231 (Continuation morphism zero-headed source tail cases)**.** *For histories $a, target$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), target)$ has one of exactly three displayed tail shapes:*

$$\mathsf{tail}(m) = \mathsf{e} \wedge target = \mathsf{E0}(a),$$

or

$$\exists k, r : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E0}(k) \wedge target = \mathsf{E0}(r) \wedge \mathsf{Cont}(\mathsf{E0}(a), k, r),$$

or

$$\exists k, r : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Proof. Unpack the continuation morphism into its displayed tail and continuation ledger. The empty-tail case is the right-unit continuation, so the target is $\mathsf{E0}(a)$. In the zero-headed and one-headed tail cases, continuation step-result inversion exposes the matching target constructor and leaves the residual continuation from $\mathsf{E0}(a)$ through the exposed tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source tail cases`

Theorem 61.232 (Continuation morphism one-headed source tail cases)**.** *For histories $a, target$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), target)$ has one of exactly three displayed tail shapes:*

$$\mathsf{tail}(m) = \mathsf{e} \wedge target = \mathsf{E1}(a),$$

or

$$\exists k, r : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E0}(k) \wedge target = \mathsf{E0}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r),$$

or

$$\exists k, r : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Proof. Unpack the continuation morphism into its displayed tail and continuation ledger. The empty-tail case is the right-unit continuation, so the target is $\mathsf{E1}(a)$. In the zero-headed and one-headed tail cases, continuation step-result inversion exposes the corresponding target constructor and leaves the residual continuation from $\mathsf{E1}(a)$ through the exposed tail. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source tail cases`

Theorem 61.233 (Continuation morphism one-headed source to zero-headed target tail cases)**.** *For histories $a, r : \mathsf{BHist}$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$ has a zero-headed tail whose exposed tail still continues from $\mathsf{E1}(a)$ to r:*

$$\exists k : \mathsf{BHist}.\ \mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Proof. Unpack the displayed tail of m. The empty-tail case would force $\mathsf{E1}(a)\mathsf{hsame}\mathsf{E0}(r)$, contradicting constructor disjointness. The one-headed tail case would force a zero-headed result to be one-headed, again contradicting constructor disjointness. Therefore the tail is zero-headed, and continuation step-result inversion exposes the residual continuation $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e0 target tail cases`

Theorem 61.234 (Continuation morphism one-headed source to zero-headed target tail core determinacy)**.** *For histories* $a, r, k, l : \mathsf{BHist}$ *and continuation morphisms* $m, n : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$*, if the two displayed tails are classified with zero-headed presentations* $\mathsf{E0}(k)$ *and* $\mathsf{E0}(l)$*, then the exposed tail cores are classified:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \to \mathsf{hsame}(\mathsf{tail}(n), \mathsf{E0}(l)) \to \mathsf{hsame}(k, l).$$

Proof. Tail determinacy for continuation morphisms with the same endpoints gives $\mathsf{hsame}(\mathsf{tail}(m), \mathsf{tail}(n))$. Compose this comparison with the two exposed zero-headed presentations. Constructor inversion for $\mathsf{E0}$ then descends the comparison to the exposed cores k and l. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e0 target tail core deterministic`

Theorem 61.235 (Continuation morphism one-headed source to zero-headed target rejects one–headed tail)**.** *For histories* $a, r, k : \mathsf{BHist}$ *and* $m : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$*,*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \to \mathsf{False}.$$

Proof. Apply the preceding zero-headed tail exposure theorem. It supplies a history t with $\mathsf{tail}(m)\mathsf{hsameE0}(t)$. If the displayed tail were also hsame to $\mathsf{E1}(k)$, transitivity would give $\mathsf{E0}(t)\mathsf{hsameE1}(k)$, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e0 target tail e1 absurd`

Theorem 61.236 (Continuation morphism one-headed source to zero-headed target zero-tail exactness)**.** *For histories* $a, r, k : \mathsf{BHist}$ *and* $m : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$*, the displayed tail is zero-headed with payload* k *exactly when that payload continues from the one-headed source to the zero-headed target payload:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \iff \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e0 target tail e0 iff`

Proof. For the forward direction, inspect the displayed tail of m. Empty and one-headed tails contradict the assumed zero-headed tail comparison. In the zero-headed branch, constructor inversion on the comparison identifies the payloads, and constructor inversion on the continuation result exposes $\mathsf{Cont}(\mathsf{E1}(a), k, r)$.

Conversely, step the displayed payload continuation through the zero constructor to form a continuation morphism with tail $\mathsf{E0}(k)$. Tail determinacy against m gives the displayed tail comparison. □

Theorem 61.237 (Continuation morphism one-headed source to zero-headed target tail is nonempty)**.** *For histories* $a, r : \mathsf{BHist}$*, every continuation morphism* $m : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$ *has a tail that cannot be classified with the empty history:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \to \mathsf{False}.$$

Proof. Apply the zero-headed tail exposure theorem above. It supplies a history k with $\mathsf{tail}(m)\mathsf{hsameE0}(k)$. If the displayed tail were also classified with e, transitivity would give $\mathsf{E0}(k)\mathsf{hsamee}$, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e0 target tail not empty`

Theorem 61.238 (Continuation morphism zero-headed source to one-headed target tail cases)**.** *For histories $a, r : \mathsf{BHist}$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r))$ has a one-headed tail whose exposed tail still continues from $\mathsf{E0}(a)$ to r:*

$$\exists k : \mathsf{BHist}.\ \mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \wedge \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Proof. Unpack the displayed tail of m. The empty-tail case would force $\mathsf{E0}(a)\mathsf{hsameE1}(r)$, contradicting constructor disjointness. The zero-headed tail case would force a one-headed result to be zero-headed, again contradicting constructor disjointness. Therefore the tail is one-headed, and continuation step-result inversion exposes the residual continuation $\mathsf{Cont}(\mathsf{E0}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e1 target tail cases`

Theorem 61.239 (Continuation morphism zero-headed source to one-headed target tail core determinacy)**.** *For histories $a, r, k, l : \mathsf{BHist}$ and continuation morphisms $m, n : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r))$, if the two displayed tails are classified with one-headed presentations $\mathsf{E1}(k)$ and $\mathsf{E1}(l)$, then the exposed tail cores are classified:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \to \mathsf{hsame}(\mathsf{tail}(n), \mathsf{E1}(l)) \to \mathsf{hsame}(k, l).$$

Proof. Tail determinacy for continuation morphisms with the same endpoints gives $\mathsf{hsame}(\mathsf{tail}(m), \mathsf{tail}(n))$. Compose this comparison with the two exposed one-headed presentations. Constructor inversion for $\mathsf{E1}$ then descends the comparison to the exposed cores k and l. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e1 target tail core deterministic`

Theorem 61.240 (Continuation morphism zero-headed source to one-headed target rejects zero–headed tail)**.** *For histories $a, r, k : \mathsf{BHist}$ and $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r))$,*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \to \mathsf{False}.$$

Proof. Apply the preceding one-headed tail exposure theorem. It supplies a history t with $\mathsf{tail}(m)\mathsf{hsameE1}(t)$. If the displayed tail were also hsame to $\mathsf{E0}(k)$, transitivity would give $\mathsf{E1}(t)\mathsf{hsameE0}(k)$, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e1 target tail e0 absurd`

Theorem 61.241 (Continuation morphism zero-headed source to one-headed target one-tail exactness)**.** *For histories $a, r, k : \mathsf{BHist}$ and $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r))$, the displayed tail is one-headed with payload k exactly when that payload continues from the zero-headed source to the one-headed target payload:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \iff \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e1 target tail e1 iff`

Proof. For the forward direction, inspect the displayed tail of m. Empty and zero-headed tails contradict the assumed one-headed tail comparison. In the one-headed branch, constructor inversion on the comparison identifies the payloads, and constructor inversion on the continuation result exposes $\mathsf{Cont}(\mathsf{E0}(a), k, r)$.

Conversely, step the displayed payload continuation through the one constructor to form a continuation morphism with tail $\mathsf{E1}(k)$. Tail determinacy against m gives the displayed tail comparison. □

Theorem 61.242 (Continuation morphism zero-headed source to one-headed target tail is nonempty)**.** *For histories $a, r : \mathsf{BHist}$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E1}(r))$ has a tail that cannot be classified with the empty history:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \to \mathsf{False}.$$

Proof. Apply the one-headed tail exposure theorem above. It supplies a history k with $\mathsf{tail}(m)\mathsf{hsameE1}(k)$. If the displayed tail were also classified with e, transitivity would give $\mathsf{E1}(k)\mathsf{hsamee}$, contradicting constructor disjointness. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e1 target tail not empty`

Corollary 61.243 (Continuation morphism zero-headed source-tail determinacy)**.** *For histories $a, r, k, l : \mathsf{BHist}$ and continuation morphisms $m, n : \mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E0}(r))$, suppose the same consumer-facing endpoint packet displays two zero-headed source-tail reads*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \quad \textit{and} \quad \mathsf{hsame}(\mathsf{tail}(n), \mathsf{E0}(l)).$$

Then the exposed zero-headed source coordinates are forced to agree:

$$\mathsf{hsame}(k, l).$$

Consequently the zero-headed source coordinate read from that endpoint packet is unique up to hsame*.*

Proof. Apply Definition 61.234 to the two endpoint morphisms m and n and to the two displayed zero-headed tail presentations. This identifies the exposed cores, giving $\mathsf{hsame}(k, l)$. Thus any consumer that tries to read the same endpoint packet through a second zero-headed source coordinate has only an hsame-equivalent source. □

Theorem 61.244 (Continuation morphism zero-headed source to zero-headed target tail cases)**.** *For histories $a, r : \mathsf{BHist}$, every continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$ has either the empty-tail case with exposed endpoints classified together,*

$$\mathsf{tail}(m) = \mathsf{e} \wedge \mathsf{hsame}(a, r),$$

or a zero-headed tail whose payload still continues from $\mathsf{E0}(a)$ to r:

$$\exists k : \mathsf{BHist}.\ \mathsf{tail}(m) = \mathsf{E0}(k) \wedge \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Proof. Unpack the displayed continuation tail. The empty-tail branch is the right-unit continuation, so constructor injectivity exposes $\mathsf{hsame}(a, r)$. A one-headed tail would force a zero-headed result to be one-headed, contradicting constructor disjointness. In the zero-headed branch, constructor injectivity on the continuation result removes the outer zero constructor and leaves $\mathsf{Cont}(\mathsf{E0}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target tail cases`

Theorem 61.245 (Continuation morphism zero-headed endpoints nonempty tail extraction)**.** *For histories $a, r : \mathsf{BHist}$ and a continuation morphism $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$, if the displayed tail is not classified with e, then it has a zero-headed payload continuing from $\mathsf{E0}(a)$ to r:*

$$(\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \to \mathsf{False}) \to \exists k : \mathsf{BHist}.\ \mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \wedge \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Proof. Apply the zero-headed endpoint tail split above. The empty-tail branch contradicts the displayed nonempty-tail hypothesis. The remaining branch already exposes a zero-headed tail and the continuation payload $\mathsf{Cont}(\mathsf{E0}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target nonempty tail cases`

Theorem 61.246 (Continuation morphism zero-headed endpoints tail core determinacy)**.** *For histories $a, r, k, l : \mathsf{BHist}$ and continuation morphisms $m, n : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$, if their displayed tails are classified with $\mathsf{E0}(k)$ and $\mathsf{E0}(l)$, then the exposed tail cores are classified:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \to \mathsf{hsame}(\mathsf{tail}(n), \mathsf{E0}(l)) \to \mathsf{hsame}(k, l).$$

Proof. Tail determinacy for continuation morphisms with the same endpoints gives $\mathsf{hsame}(\mathsf{tail}(m), \mathsf{tail}(n))$. Chaining with the two displayed zero-headed tail classifications gives $\mathsf{hsame}(\mathsf{E0}(k), \mathsf{E0}(l))$, and zero-constructor inversion removes the outer tag. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target tail core deterministic`

Proposition 61.247 (Continuation morphism zero-headed endpoints reject one-headed tail)**.** *For histories $a, r, k : \mathsf{BHist}$ and $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$,*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E1}(k)) \to \mathsf{False}.$$

Proof. Unpack the displayed continuation tail. The empty-tail case contradicts $\mathsf{ehsameE1}(k)$, the zero-headed tail case contradicts $\mathsf{E0}(t)\mathsf{hsameE1}(k)$, and the one-headed tail case would force the zero-headed result to be one-headed in the continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target tail e1 absurd`

Theorem 61.248 (Continuation morphism zero-headed endpoints zero-tail exactness)**.** *For histories $a, r, k : \mathsf{BHist}$ and $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$, the displayed tail is zero-headed with payload k exactly when the payload continues from the zero-headed source to the target payload:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)) \iff \mathsf{Cont}(\mathsf{E0}(a), k, r).$$

Proof. For the forward direction, apply the zero-headed endpoint tail cases. The empty-tail branch contradicts the assumed zero-headed tail comparison. In the visible branch, the exposed zero-headed tail payload and the assumed tail comparison identify the two payloads, so continuation transport gives $\mathsf{Cont}(\mathsf{E0}(a), k, r)$. Conversely, step the displayed payload continuation through the zero constructor and compare it with the record continuation by left cancellation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target tail e0 iff`

Proposition 61.249 (Continuation morphism zero-headed endpoints empty-tail exactness)**.** *For histories $a, r : \mathsf{BHist}$ and $m : \mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$, the displayed tail is empty exactly when the endpoint payloads are history-same:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \iff \mathsf{hsame}(a, r).$$

Proof. Apply the zero-headed endpoint tail cases. In the forward direction, the empty-tail branch exposes $\mathsf{hsame}(a, r)$, while the zero-headed visible-tail branch contradicts empty/zero separation. Conversely, an endpoint payload comparison builds the empty-tail displayed morphism, and continuation-morphism tail determinacy identifies it with m. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e0 source e0 target tail empty iff`

Corollary 61.250 (Continuation morphism zero-headed endpoints consumer determinacy)**.** *For histories a, r, k, l : BHist and continuation morphisms m, n : $\mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$, suppose a consumer-facing use fixes the zero-headed visible-tail row for m:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{E0}(k)).$$

If another accepted consumer-facing use of the same zero-headed endpoint packet reads a zero-headed tail row for n,

$$\mathsf{hsame}(\mathsf{tail}(n), \mathsf{E0}(l)),$$

then the visible payload coordinate and the zero-tail exactness decision are determined:

$$\mathsf{hsame}(k, l) \quad \textit{and} \quad (\mathsf{Cont}(\mathsf{E0}(a), k, r) \iff \mathsf{Cont}(\mathsf{E0}(a), l, r)).$$

Consequently, after the endpoint rows and one zero-headed tail row are fixed, every accepted route, normal-form read, or exactness decision for this packet uses the same zero-tail payload up to hsame*. The only displayed branch that carries no payload is the empty-tail endpoint-collapse row of Definition 61.244; there is no accepted one-headed tail, hidden coordinate, or duplicate classifier source.*

Proof. Apply Definition 61.244 to a continuation morphism x : $\mathsf{ContinuationMorphism}(\mathsf{E0}(a), \mathsf{E0}(r))$. The consumer-visible tail surface is exhausted by the two displayed rows

$$\mathsf{tail}(x) = \mathsf{e} \wedge \mathsf{hsame}(a, r) \quad \text{or} \quad \exists q : \mathsf{BHist}.\ \mathsf{tail}(x) = \mathsf{E0}(q) \wedge \mathsf{Cont}(\mathsf{E0}(a), q, r).$$

Thus a payload-bearing read over these endpoints must be a zero-headed read, and its payload is the only coordinate exposed by the case split.

For the two accepted uses in the statement, both reads are zero-headed. Applying Definition 61.246 to the two displayed tail classifications gives $\mathsf{hsame}(k, l)$. The zero-tail exactness row of Definition 61.248 reads the first displayed classification as exactly $\mathsf{Cont}(\mathsf{E0}(a), k, r)$ and the second as exactly $\mathsf{Cont}(\mathsf{E0}(a), l, r)$. By the continuation history-sameness transport theorem Definition 4.41, transport along $\mathsf{hsame}(k, l)$ identifies these two continuation payload decisions, giving the displayed equivalence.

It remains only to rule out alternate consumer coordinates. A one-headed tail proposal is refused by Definition 61.247. A second zero-headed proposal is another instance of the same core-determinacy theorem and is therefore hsame to k. An empty-tail proposal is the endpoint-collapse row listed by the case theorem and carries no payload coordinate at all. Hence the accepted consumer surface is exhausted by the empty row and the unique zero-headed payload row named in the chapter. □

61.3 The certificate

Theorem 61.251 (Continuation morphism one-headed endpoints empty-tail exactness)**.** *For histories a, r : BHist and m : $\mathsf{ContinuationMorphism}(\mathsf{E1}(a), \mathsf{E1}(r))$, the displayed tail is empty exactly when the endpoint payloads are history-same:*

$$\mathsf{hsame}(\mathsf{tail}(m), \mathsf{e}) \iff \mathsf{hsame}(a, r).$$

Proof. For the forward direction, apply the one-headed endpoint tail cases. The visible one-tail branch contradicts empty-tail separation, leaving the empty-tail branch and hence $\mathsf{hsame}(a, r)$. Conversely, use the endpoint payload comparison to form the empty-tail continuation morphism and compare it with m by tail determinacy. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism e1 source e1 target tail empty iff`

Theorem 61.252 (Category empty morphism between one-headed endpoints)**.** *For histories a, r, the empty morphism is a hom-carrier from $\mathsf{E1}(a)$ to $\mathsf{E1}(r)$ exactly when the exposed source tail is unary and the two exposed tails are classified together:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), \mathsf{e}) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(a, r).$$

Proof. The forward direction applies the one-headed result case split to the displayed empty morphism. The visible morphism branch contradicts constructor no-confusion, and the empty branch exposes unary membership of a and $\mathsf{hsame}(a, r)$. Conversely, unary one-step closure supplies the two endpoint carriers, the tail classifier lifts through $\mathsf{E1}$, and empty-identity exactness rebuilds the hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target empty morphism iff`

Theorem 61.253 (Category hom-carrier one-headed morphism target exactness)**.** *For histories a, k, r,*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}(a, k, r).$$

Proof. Unpack the hom-carrier witness. The forward direction keeps the unary source witness, inverts the one-headed morphism unary witness to obtain unary membership for k, and removes the outer one constructor from the continuation result. Conversely, unary continuation closure gives unary membership for r, one-step closure gives the one-headed target and morphism carriers, and the continuation one-step rule lifts $\mathsf{Cont}(a, k, r)$ to $\mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r))$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 morphism target iff`

Theorem 61.254 (Category empty-source one-headed target exactness)**.** *For histories $r, morph$, an empty-source hom-carrier into the one-headed target $\mathsf{E1}(r)$ is exactly equality of the morphism with that target together with unary payload membership:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, \mathsf{E1}(r), morph) \iff morph = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r).$$

Proof. Category empty-source exactness gives unary membership of the one-headed target and equality of the morphism with that target. Unary inversion removes the visible constructor from the target carrier. Conversely, unary one-step closure rebuilds the target carrier and the displayed equality supplies the empty-source continuation classifier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source e1 target iff`

Theorem 61.255 (Category empty-source one-headed morphism exactness)**.** *For histories $k, target$, an empty-source hom-carrier whose morphism is $\mathsf{E1}(k)$ is exactly equality of the target with that morphism together with unary payload membership:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, target, \mathsf{E1}(k)) \iff target = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(k).$$

Proof. Category empty-source exactness identifies the one-headed morphism with the target. Eliminating that equality turns the target carrier into a one-headed unary witness, whose payload is unary by inversion. Conversely, one-step unary closure rebuilds the target carrier, and symmetry of the displayed equality provides the required empty-source classifier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier empty source e1 morphism iff`

Theorem 61.256 (Category hom-carrier one-headed target morphism cases)**.** *For histories source, r, morph, a hom-carrier whose target is one-headed has only the empty morphism case or a one-headed morphism tail:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{E1}(r), morph) \Longrightarrow (morph = \mathsf{e} \wedge source = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r))$$

or there exists a history k such that

$$morph = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}(source, k, r).$$

Proof. Unpack the hom-carrier witness into unary source, target, and morphism fields and the continuation ledger. Classify the morphism history. In the empty morphism branch, the continuation right-unit shape identifies the source with $\mathsf{E1}(r)$, and unary inversion on the target gives the unary history for r. The zero-headed morphism branch contradicts unary-history separation for zero extensions. In the one-headed branch, unary inversion gives the tail carrier for k, and constructor inversion on the continuation ledger yields $\mathsf{Cont}(source, k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target morphism cases`

Theorem 61.257 (Category hom-carrier one-headed target source and morphism cases)**.** *For histories source, r, morph, a hom-carrier whose target is* $\mathsf{E1}(r)$ *separates the empty source branch from the one-headed source branch:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{E1}(r), morph) \Longrightarrow (source = \mathsf{e} \wedge morph = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r))$$

or there exists a history a such that

$$source = \mathsf{E1}(a) \wedge \mathsf{UnaryHistory}(a) \wedge ((morph = \mathsf{e} \wedge \mathsf{hsame}(a, r)) \vee \exists k,\ morph = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}(\mathsf{E1}($$

Proof. Classify the source endpoint. If it is empty, the continuation ledger and the one-headed target force the morphism to be $\mathsf{E1}(r)$, and target unary inversion gives $\mathsf{UnaryHistory}(r)$. If it is zero-headed, the source unary field contradicts zero-extension separation. If it is one-headed, apply the left-one-headed result case split: the empty morphism branch gives $\mathsf{hsame}(a, r)$, and the visible morphism branch exposes a tail k with $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target source morphism cases`

Theorem 61.258 (Category hom-carrier one-headed morphism target cases)**.** *For histories a, target, k, a hom-carrier whose morphism endpoint is one-headed forces the target endpoint to be one-headed and descends the hom-carrier to the exposed target and morphism tails:*

$$\mathsf{CategoryHomCarrier}(a, target, \mathsf{E1}(k)) \Longrightarrow \exists r : \mathsf{BHist}.\ target = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, r, k).$$

Proof. Unpack the hom-carrier into its unary source, target, and morphism fields and its continuation ledger. The continuation step-result inversion for a one-headed right input exposes a target tail r with $target = \mathsf{E1}(r)$ and a tail continuation $\mathsf{Cont}(a, k, r)$. Unary inversion on the displayed target and morphism fields gives the two tail carriers, which repack with the source carrier and tail continuation as $\mathsf{CategoryHomCarrier}(a, r, k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 morphism target cases`

Theorem 61.259 (Category hom-carrier one-headed morphism source and target cases)**.** *For histories $source, target, k$, a hom-carrier whose morphism endpoint is one-headed separates the empty source branch from the one-headed source branch:*

$$\mathsf{CategoryHomCarrier}(source, target, \mathsf{E1}(k)) \Longrightarrow (source = \mathsf{e} \wedge target = \mathsf{E1}(k) \wedge \mathsf{UnaryHistory}(k))$$

or there exist histories a, r such that

$$source = \mathsf{E1}(a) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Proof. Classify the source endpoint. If it is empty, empty-source one-headed morphism exactness identifies the target with $\mathsf{E1}(k)$ and exposes $\mathsf{UnaryHistory}(k)$. If it is zero-headed, the source unary field contradicts zero-extension separation. If it is one-headed, one-headed-source one-headed-morphism target exactness exposes the target tail r, the unary source and morphism tails, the unary target tail, and the continuation ledger $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 morphism source target cases`

Theorem 61.260 (Category composite one-headed morphism target factor)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is the one-headed morphism $\mathsf{E1}(k)$, then the target object is one-headed and the right factor is already a hom-carrier into that displayed target:*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k))\\ &\quad\Longrightarrow \exists r : \mathsf{BHist}.\ c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, r, k) \wedge \mathsf{CategoryHomCarrier}(b, \mathsf{E1}(r), g).\end{aligned}$$

Proof. First close composition to obtain a hom-carrier from a to c with morphism $\mathsf{E1}(k)$. The one-headed morphism target case theorem exposes a history r with $c = \mathsf{E1}(r)$ and a descended hom-carrier from a to r with morphism k. Substituting the displayed target equality transports the original right factor from target c to target $\mathsf{E1}(r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp e1 morphism target factor`

Theorem 61.261 (Category composite one-headed morphism right factor cases)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is the one-headed morphism $\mathsf{E1}(k)$, then the right factor is either the empty identity case at the recovered target or a visible one-headed morphism into that target:*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k))\\ &\quad\Longrightarrow (g = \mathsf{e} \wedge c = b \wedge \mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(k)))\\ &\qquad\vee \exists l, r : \mathsf{BHist}.\ g = \mathsf{E1}(l) \wedge c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, r, k) \wedge \mathsf{CategoryHomCarrier}(b, \mathsf{E1}(r), \mathsf{E1}(l)).\end{aligned}$$

Proof. Apply the one-headed composite target-factor theorem to recover r with $c = \mathsf{E1}(r)$, a descended left hom-carrier $\mathsf{CategoryHomCarrier}(a, r, k)$, and the original right hom-carrier $\mathsf{CategoryHomCarrier}(b, \mathsf{E1}(r)$ Split this right hom-carrier by the one-headed target morphism cases. In the empty branch, the source equality $b = \mathsf{E1}(r)$ identifies c with b, and the one-headed morphism target equivalence rebuilds $\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(k))$ from the descended left carrier. In the visible branch, substitute $g = \mathsf{E1}(l)$ and keep the original right hom-carrier at the recovered target. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp e1 morphism right factor cases`

Theorem 61.262 (Category composite one-headed morphism left factor cases)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is the one-headed morphism $\mathsf{E1}(k)$, then the left factor is either the empty identity case at the source boundary or a visible one-headed morphism whose target is one-headed:*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k)) \\
&\quad \Longrightarrow (f = \mathsf{e} \wedge a = b \wedge \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(k))) \\
&\qquad \vee \exists l, r : \mathsf{BHist}.\ f = \mathsf{E1}(l) \wedge b = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), c, g) \wedge \mathsf{Cont}(\mathsf{E}
\end{aligned}$$

Proof. Classify the left morphism history. If it is empty, the left hom-carrier identity ledger identifies a and b, while the composite continuation with empty left factor transports the right hom-carrier to morphism $\mathsf{E1}(k)$. If the left morphism is zero-headed, its unary morphism field contradicts zero-extension separation. If it is one-headed, the one-headed morphism target case theorem exposes $b = \mathsf{E1}(r)$ and the descended left hom-carrier $\mathsf{CategoryHomCarrier}(a, r, l)$; substituting this target equality keeps the original right hom-carrier at source $\mathsf{E1}(r)$, and the original composite ledger is the visible-tail continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp e1 morphism left factor cases`

Theorem 61.263 (Category nonempty one-headed composite morphism factors)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is $\mathsf{E1}(k)$ and neither factor is internally the empty morphism, then both factors and both exposed target objects are one-headed:*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k)) \\
&\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False}) \\
&\Longrightarrow \exists l, m, r, s : \mathsf{BHist}.\ f = \mathsf{E1}(l) \wedge g = \mathsf{E1}(m) \wedge b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \\
&\qquad \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)).
\end{aligned}$$

Proof. Apply the left factor case split for one-headed composites. The empty-left branch contradicts nonemptiness of f, so the visible-left branch exposes $f = \mathsf{E1}(l)$, $b = \mathsf{E1}(r)$, a descended carrier $\mathsf{CategoryHomCarrier}(a, r, l)$, and a right carrier from $\mathsf{E1}(r)$. Apply the nonempty one-headed-source morphism case split to that right carrier. The empty-right branch contradicts nonemptiness of g, and the visible-right branch exposes $g = \mathsf{E1}(m)$ and $c = \mathsf{E1}(s)$. Repacking its unary and continuation witnesses gives $\mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m)$, while substituting $g = \mathsf{E1}(m)$ keeps the displayed composite continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty e1 morphism factors`

Theorem 61.264 (Category nonempty one-headed composite factor tail uniqueness)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if the composite continuation is $\mathsf{E1}(k)$, neither factor is internally empty, and the displayed factors are $f = \mathsf{E1}(l')$ and $g = \mathsf{E1}(m')$, then the descended factor tails are forced:*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k)) \\
&\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False}) \wedge f = \mathsf{E1}(l') \wedge g = \mathsf{E1}(m') \\
&\implies \exists r, s : \mathsf{BHist}.\ b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \wedge \mathsf{CategoryHomCarrier}(a, r, l') \\
&\qquad \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m') \wedge \mathsf{Cont}(\mathsf{E1}(l'), \mathsf{E1}(m'), \mathsf{E1}(k)).
\end{aligned}$$

Proof. The nonempty factor theorem exposes one-headed witnesses for both factors. Substituting the displayed equalities $f = \mathsf{E1}(l')$ and $g = \mathsf{E1}(m')$ identifies the exposed tails with l' and m'. The same factor package then supplies the middle and target objects, the two descended hom-carriers, and the displayed one-headed composite continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty e1 morphism factor tail unique`

Theorem 61.265 (Category nonempty one-headed composite result descent)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is $\mathsf{E1}(k)$ and neither factor is internally the empty morphism, then the exposed tail data also carries the descended composite hom-carrier:*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k)) \\
&\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False}) \\
&\implies \exists l, m, r, s : \mathsf{BHist}.\ f = \mathsf{E1}(l) \wedge g = \mathsf{E1}(m) \wedge b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \\
&\qquad \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{CategoryHomCarrier}(a, s, k).
\end{aligned}$$

Proof. The nonempty factor theorem exposes l, m, r, s together with $\mathsf{CategoryHomCarrier}(a, r, l)$, $\mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m)$, and $\mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k))$. Constructor inversion on the last continuation gives $\mathsf{Cont}(\mathsf{E1}(l), m, k)$. Combining the left carrier continuation $a \cdot l = r$ with the right carrier continuation $\mathsf{E1}(r) \cdot m = s$, associativity of BEDC continuation identifies $a \cdot k$ with s. The unary fields from the two carrier witnesses and the visible-tail continuation supply the remaining fields of $\mathsf{CategoryHomCarrier}(a, s, k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty e1 morphism result descent`

Theorem 61.266 (Category nonempty one-headed composite result descent uniqueness)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is $\mathsf{E1}(k)$, neither factor is internally empty, and a displayed carrier also presents $k : a \to s'$, then the descended target agrees with s':*

$$\begin{aligned}
&\mathsf{CategoryHomCarrier}(a, b, f) \wedge \mathsf{CategoryHomCarrier}(b, c, g) \wedge \mathsf{Cont}(f, g, \mathsf{E1}(k)) \\
&\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False}) \wedge \mathsf{CategoryHomCarrier}(a, s', k) \\
&\implies \exists l, m, r, s : \mathsf{BHist}.\ f = \mathsf{E1}(l) \wedge g = \mathsf{E1}(m) \wedge b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \\
&\qquad \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{CategoryHomCarrier}(a, s, k) \wedge s\mathsf{hsame}s'.
\end{aligned}$$

Proof. The result-descent theorem supplies the exposed factors and the descended carrier $\mathsf{CategoryHomCarrier}(a, s, k$ Comparing that carrier with the displayed carrier $\mathsf{CategoryHomCarrier}(a, s', k)$ by target determinism gives $s\mathsf{hsame}s'$, while the exposed factor data is retained unchanged. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty e1 morphism result descent unique`

Theorem 61.267 (Category nonempty one-headed composite tail-result descent)**.** *For composable category hom-carriers $f : a \to b$ and $g : b \to c$, if their continuation composite is $\mathsf{E1}(k)$ and neither factor is internally the empty morphism, then the same exposed factor data also carries the visible tail continuation into the descended composite carrier:*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(a,b,f) \wedge \mathsf{CategoryHomCarrier}(b,c,g) \wedge \mathsf{Cont}(f,g,\mathsf{E1}(k))\\ &\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False})\\ &\Longrightarrow \exists l,m,r,s : \mathsf{BHist}.\ f = \mathsf{E1}(l) \wedge g = \mathsf{E1}(m) \wedge b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s)\\ &\qquad \wedge \mathsf{CategoryHomCarrier}(a,r,l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r),s,m) \wedge \mathsf{Cont}(\mathsf{E1}(l),m,k) \wedge \mathsf{CategoryHomCarrier}(a,s\end{aligned}$$

Proof. Apply the nonempty factor theorem to obtain the one-headed factor displays and the visible composite continuation $\mathsf{Cont}(\mathsf{E1}(l),\mathsf{E1}(m),\mathsf{E1}(k))$. Constructor inversion turns that ledger into $\mathsf{Cont}(\mathsf{E1}(l),m,k)$. The same carrier fields and continuation associativity then reconstruct the descended composite hom-carrier $\mathsf{CategoryHomCarrier}(a,s,k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty e1 morphism tail result descent`

Corollary 61.268 (Category empty one-headed endpoint consumer determinacy)**.** *For histories $a,r,x,b,c,f,g,k,l',m' : \mathsf{BHist}$, suppose the nonempty composite surface fixes the displayed factor rows*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(x,b,f) \wedge \mathsf{CategoryHomCarrier}(b,c,g) \wedge \mathsf{Cont}(f,g,\mathsf{E1}(k))\\ &\quad \wedge (f\mathsf{hsamee} \to \mathsf{False}) \wedge (g\mathsf{hsamee} \to \mathsf{False}) \wedge f = \mathsf{E1}(l') \wedge g = \mathsf{E1}(m').\end{aligned}$$

Then the empty morphism between the one-headed endpoints has the single chapter exactness decision

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a),\mathsf{E1}(r),\mathsf{e}) \Longleftrightarrow \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(a,r),$$

and the nonempty visible-composite readback has the forced finite factor route

$$\exists u,v : \mathsf{BHist}.\ b = \mathsf{E1}(u) \wedge c = \mathsf{E1}(v) \wedge \mathsf{CategoryHomCarrier}(x,u,l')$$

$$\wedge\ \mathsf{CategoryHomCarrier}(\mathsf{E1}(u),v,m') \wedge \mathsf{Cont}(\mathsf{E1}(l'),\mathsf{E1}(m'),\mathsf{E1}(k)).$$

Consequently, once these rows are fixed, every accepted consumer-facing route, normal-form read, or exactness decision for the empty one-headed endpoint and its attached nonempty visible composite factors through the same listed rows: the empty row supplies only $\mathsf{UnaryHistory}(a)$ *and* $\mathsf{hsame}(a,r)$*, while the nonempty row supplies only the displayed tails l',m' and the recovered endpoints u,v. There is no alternate hidden coordinate and no duplicate classifier source for this packet.*

Proof. The empty endpoint part is exactly Definition 61.252. It reads the carrier $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a),\mathsf{E1}(r),\mathsf{e}$ as unary membership of the exposed source tail together with the endpoint classifier $\mathsf{hsame}(a,r)$, and it provides the converse reconstruction from precisely those two rows. Thus the empty morphism exposes no visible tail, middle endpoint, or second classifier coordinate.

For the nonempty visible-composite part, apply Definition 61.264 to the fixed hom-carrier rows, the fixed continuation row, the two internal nonemptiness assumptions, and the displayed identities $f = \mathsf{E1}(l')$ and $g = \mathsf{E1}(m')$. The theorem returns histories u and v with $b = \mathsf{E1}(u)$ and $c = \mathsf{E1}(v)$, together with the descended carrier rows $\mathsf{CategoryHomCarrier}(x,u,l')$ and $\mathsf{CategoryHomCarrier}(\mathsf{E1}(u),v,m')$, and the visible composite continuation $\mathsf{Cont}(\mathsf{E1}(l'),\mathsf{E1}(m'),\mathsf{E1}(k))$.

These are exactly the route and readback rows displayed in the conclusion. An accepted consumer use that keeps the hypotheses fixed cannot choose a different empty exactness decision without contradicting the first iff, and cannot choose different visible factor tails without contradicting the factor-tail uniqueness readback, because the displayed equalities have already identified the exposed tails with l' and m'. Hence every accepted use factors through the listed finite rows and no unlisted coordinate remains available. □

Theorem 61.269 (Category visible composable morphism target cases)**.** *For histories a, b, c, l, m, k, if*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(l)), \qquad \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(m)), \qquad \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)),$$

then there exist histories r, s such that

$$b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)).$$

Proof. Apply the one-headed morphism target-case theorem to the left visible hom-carrier and to the right visible hom-carrier. The two applications expose tails r and s with $b = \mathsf{E1}(r)$ and $c = \mathsf{E1}(s)$. Substitute the first target equality into the right descended hom-carrier, retain the left descended hom-carrier, and carry the original visible continuation ledger into the existential package. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier visible composable morphisms target cases`

Theorem 61.270 (Category visible composable morphism tail cases)**.** *For histories a, b, c, l, m, k, if*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(l)), \qquad \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(m)), \qquad \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)),$$

then there exist histories r, s such that

$$b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{Cont}(\mathsf{E1}(l), m, k).$$

Proof. First apply the visible composable target-case theorem to expose the intermediate and final target tails r and s, together with the two descended hom-carriers. The remaining continuation ledger is obtained by right inversion of the one-headed step rule for $\mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k))$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier visible composable morphisms tail cases`

Theorem 61.271 (Category hom-carrier composite visible morphisms result cases)**.** *For histories a, b, c, l, m, fg, if*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(l)), \qquad \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(m)), \qquad \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), fg),$$

then there exists a history k such that

$$fg = \mathsf{E1}(k) \wedge \mathsf{Cont}(\mathsf{E1}(l), m, k) \wedge \mathsf{CategoryHomCarrier}(a, c, \mathsf{E1}(k)).$$

Proof. The continuation ledger unfolds the composite result as $\mathsf{append}(\mathsf{E1}(l), \mathsf{E1}(m))$, which is definitionally $\mathsf{E1}(\mathsf{append}(\mathsf{E1}(l), m))$. Choose $k = \mathsf{append}(\mathsf{E1}(l), m)$. The exposed tail continuation is the canonical append continuation, and the hom-carrier result is obtained by applying hom-carrier composition closure to the two visible morphisms and the same canonical continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp visible morphisms result cases`

Theorem 61.272 (Category hom-carrier nonempty morphism target visible)**.** *For histories* $source, target, morph$: BHist*, if*

$$\mathsf{CategoryHomCarrier}(source, target, morph)$$

and morph is not internally empty, then the morphism and target are both one-headed and the hom-continuation ledger descends through their exposed payloads:

$$\exists k, r.\ morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(source, k, r).$$

Proof. Unpack the hom-carrier witness. The morphism cannot be empty by the nonempty hypothesis, and it cannot be zero-headed because unary histories have no zero-headed extension. In the one-headed morphism case, the continuation ledger forces the result target to be one-headed; constructor inversion then exposes the target payload, both unary payload witnesses, and the descended continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier nonempty morphism target visible`

Theorem 61.273 (Category hom-carrier nonempty morphism visible endpoint cases)**.** *For histories* $source, target, morph$: BHist*, if*

$$\mathsf{CategoryHomCarrier}(source, target, morph)$$

and morph is not internally empty, then either the source is empty and the visible target and morphism coincide, or all three visible endpoints are one-headed:

$$\exists r.\ source = \mathsf{e} \wedge target = \mathsf{E1}(r) \wedge morph = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r)$$
$$\vee \exists a, k, r.\ source = \mathsf{E1}(a) \wedge morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}($$

Proof. Inspect the source constructor. In the empty-source branch, the empty-source hom-carrier law identifies the morphism with the target; the nonempty morphism hypothesis removes the empty target case, and unary generation removes the zero-headed target case. In the one-headed source branch, the nonempty and unary hypotheses force the morphism to be one-headed, and the continuation ledger then forces the target to be one-headed. Constructor inversion exposes the three unary payloads and the descended continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier nonempty morphism visible endpoint cases`

Theorem 61.274 (Category visible composite result target cases)**.** *For histories* a, b, c, l, m, fg*, if*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(l)), \qquad \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(m)), \qquad \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), fg),$$

then there exist histories k, r, s *such that*

$$fg = \mathsf{E1}(k) \wedge b = \mathsf{E1}(r) \wedge c = \mathsf{E1}(s) \wedge \mathsf{CategoryHomCarrier}(a, r, l) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(r), s, m) \wedge \mathsf{CategoryHom}$$

Proof. The visible composite result theorem exposes $fg = \mathsf{E1}(k)$ and the composite hom-carrier from a to c with morphism $\mathsf{E1}(k)$. Applying one-headed morphism target cases to the two visible factors exposes $b = \mathsf{E1}(r)$ and $c = \mathsf{E1}(s)$, with the descended factor hom-carriers. Substituting the target equalities gives the displayed target-specific composite carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp visible morphisms result target cases`

Theorem 61.275 (Category hom-carrier visible composite result is not empty)**.** *For histories* a, b, c, l, m, fg*, if*

$$\mathsf{CategoryHomCarrier}(a, b, \mathsf{E1}(l)), \qquad \mathsf{CategoryHomCarrier}(b, c, \mathsf{E1}(m)), \qquad \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), fg),$$

then fg *cannot be internally the same as* e*.*

Proof. Invert the displayed continuation ledger. The result is definitionally $\mathsf{append}(\mathsf{E1}(l), \mathsf{E1}(m))$, hence one-headed. Constructor no-confusion rejects any hsame comparison between that one-headed result and e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp visible morphisms result not empty`

Theorem 61.276 (Category hom-carrier nonempty composite target visible)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg *is not internally empty, then* fg *and the target object* c *are both one-headed, and the composite continuation ledger is visible at that boundary:*

$$\exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r)).$$

Proof. Composition closure turns the two hom-carriers and the displayed composite ledger into a hom-carrier from a to c over fg. The composite morphism is unary. The empty case contradicts the nonempty hypothesis, and the zero-headed case contradicts unary generation. In the one-headed case, the target cannot be empty or zero-headed because the continuation ledger has one-headed result shape; the remaining target case exposes $c = \mathsf{E1}(r)$ and carries the original continuation witness at the visible boundary. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty target visible`

Theorem 61.277 (Category hom-carrier nonempty composite target not empty)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg *is not internally empty, then the target object* c *is not internally empty:*

$$\neg\mathsf{hsame}(fg, \mathsf{e}) \Longrightarrow \neg\mathsf{hsame}(c, \mathsf{e}).$$

Proof. Apply the nonempty target-visible theorem. It presents the target as $c = \mathsf{E1}(r)$ for a unary payload r. Constructor separation rules out an internal equality between this one-headed target and e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty target not empty`

Theorem 61.278 (Category nonempty composite source-target cases)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg *is not internally empty, then either the source object is empty and the visible target boundary is carried directly, or the source object is itself one-headed and the whole visible source, composite, and target boundary is displayed:*

$$\begin{aligned}&(\mathsf{hsame}(a, \mathsf{e}) \wedge \exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r)))\\ &\vee(\exists s, k, r.\ a = \mathsf{E1}(s) \wedge fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(s\end{aligned}$$

Proof. First apply the nonempty target-visible theorem to the displayed composite. This gives one-headed presentations of fg and c together with the continuation ledger at that visible target boundary. Inspect the source object a. The empty case gives the first disjunct. The zero-headed case contradicts the unary source field of the left hom-carrier. The one-headed case exposes the source tail and reuses the same visible continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp result nonempty source target cases`

Theorem 61.279 (Category nonempty result visible hom readback)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg is not internally empty, then its one-headed result presentation also reads back as a visible composite hom-carrier:

$$\exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)).$$

Proof. Apply the nonempty composite target-visible theorem. It exposes histories k, r with $fg = \mathsf{E1}(k)$ and $c = \mathsf{E1}(r)$, together with unary witnesses for k and r and the boundary continuation $\mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r))$. The source unary witness is already part of the left hom-carrier, so these fields assemble the visible hom-carrier $\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k))$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty result visible hom readback`

Theorem 61.280 (Category nonempty result visible source hom readback)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If both the displayed composite morphism fg and the source object a are not internally empty, then the visible result presentation reads back with a visible source:

$$\exists s, k, r.\ a = \mathsf{E1}(s) \wedge fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(\mathsf{E1}(s), \mathsf{E1}(r), \mathsf{E1}(k)).$$

Proof. First apply the nonempty result visible hom readback theorem to obtain visible histories k, r and the hom-carrier $\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k))$. Inspect the source object a. The empty branch contradicts the nonempty source hypothesis; the zero branch contradicts the unary source field of the hom-carrier. In the one-headed branch, the exposed source tail supplies s, and the displayed hom-carrier is already the required visible-source carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier comp nonempty result visible source hom readback`

Theorem 61.281 (Category visible composite target bridge determinacy)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg is not internally empty, then every consumer-facing route, transport, readback, or handoff for the visible composite target boundary factors through the following finite local surface:

result readback: $\exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)),$

source-target cases: $(\mathsf{hsame}(a, \mathsf{e}) \wedge \exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(a, \mathsf{E}$
$\vee(\exists s, k, r.\ a = \mathsf{E1}(s) \wedge fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryH}$

Consequently the bridge exports only the displayed result history, visible target history, visible hom-carrier readback, empty-source branch, and one-headed source-target branch. It exports no separate coordinate for a host composite target, an alternate composite morphism, a chosen external category morphism, or a consumer-side target not obtained from these rows.

Proof. Apply Definition 61.279 to the three displayed hypotheses and the nonempty result premise. This gives the result readback row: histories k, r with $fg = \mathsf{E1}(k)$, $c = \mathsf{E1}(r)$, and the visible composite hom-carrier $\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k))$. Thus any consumer read of the nonempty composite target must first pass through the same visible result and target presentations.

Apply Definition 61.278 to the same hypotheses. Its first branch is exactly the empty-source route, where the boundary continuation is displayed as $\mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r))$ together with the unary witnesses for the result and target payloads. Its second branch is exactly the visible source-target route, where $a = \mathsf{E1}(s)$ and the boundary continuation descends to $\mathsf{Cont}(\mathsf{E1}(s), \mathsf{E1}(k), \mathsf{E1}(r))$ with unary witnesses for all three payloads.

The two cited rows account for every displayed consumer operation in the nonempty composite target boundary. Readback uses the first row, source or target transport uses the branch data in the second row, and any visible-source handoff is the second disjunct with the same $fg = \mathsf{E1}(k)$ and $c = \mathsf{E1}(r)$ presentations. Since the hypotheses supply only the two hom-carrier rows, the single continuation ledger, and the nonempty result premise, and since the cited theorems expose exactly the coordinates listed above, no unlisted bridge coordinate can be projected from this local surface. □

Corollary 61.282 (Category visible composite target consumer determinacy)**.** *For the unary-continuation category, suppose*

$$\mathsf{CategoryHomCarrier}(a, b, f), \qquad \mathsf{CategoryHomCarrier}(b, c, g), \qquad \mathsf{Cont}(f, g, fg).$$

If the displayed composite morphism fg is not internally empty, and if the result-readback and source-target rows displayed by Definition 61.281 are fixed, then every accepted consumer-facing use of the visible composite target boundary chooses the same finite route, normal form, or exactness decision named by this chapter. The fixed surface is exactly

result readback: $\exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)),$

source-target cases: $(\mathsf{hsame}(a, \mathsf{e}) \wedge \exists k, r.\ fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(a, \mathsf{E}$
$\vee (\exists s, k, r.\ a = \mathsf{E1}(s) \wedge fg = \mathsf{E1}(k) \wedge c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryH}$

Consequently two accepted consumers over those fixed rows cannot select different finite routes, different visible target normal forms, or different exactness decisions, and no accepted use can introduce an alternate hidden coordinate or duplicate classifier source for the composite target.

Proof. Apply Definition 61.281 to the two displayed hom-carrier rows, the displayed continuation row, and the nonempty composite premise. Its determinacy statement says that every consumer-facing route, transport, readback, or handoff for the visible composite target boundary factors through the two-row local surface written above.

The first row is the result-readback normal form: it presents the composite morphism as $fg = \mathsf{E1}(k)$, the target as $c = \mathsf{E1}(r)$, and the consumer-visible hom-carrier as $\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k))$. Thus any accepted result readback has the same visible composite morphism coordinate, the same visible target coordinate, and the same hom-carrier readback route once that row is fixed.

For source and target transport, apply Definition 61.278 to the same three structural rows and the same nonempty premise. Its first branch is exactly the empty-source boundary row, carrying

$\mathsf{hsame}(a, \mathsf{e})$ together with $fg = \mathsf{E1}(k)$, $c = \mathsf{E1}(r)$, the two unary witnesses, and the boundary continuation $\mathsf{Cont}(a, \mathsf{E1}(k), \mathsf{E1}(r))$. Its second branch is exactly the one-headed source-target boundary row, carrying $a = \mathsf{E1}(s)$ together with the same visible result and target presentations and the descended continuation $\mathsf{Cont}(\mathsf{E1}(s), \mathsf{E1}(k), \mathsf{E1}(r))$. Hence every accepted source-target normal form or exactness decision is one of these two displayed branches.

It remains to exclude competing consumer choices. The bridge determinacy theorem exports no separate coordinate for a host composite target, alternate composite morphism, chosen external category morphism, or consumer-side target outside these rows. The source-target theorem consumes only the displayed hom-carrier rows, the displayed continuation row, and the nonempty premise, and returns only the two listed branches. Therefore an alleged alternate hidden coordinate or duplicate classifier source is neither a component of the result-readback row nor a component of either source-target branch. It has no local factor through the accepted visible composite target surface, so it cannot determine a second finite route, normal form, or exactness decision for the fixed rows. □

Theorem 61.283 (Continuation morphism composite visible tails result cases)**.** *For histories a, b, c, l, m, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the two displayed tails are $\mathsf{hsame}(\mathsf{tail}(left), \mathsf{E1}(l))$ and $\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m))$, then there exists a history k such that*

$$\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E1}(k)) \wedge \mathsf{Cont}(\mathsf{E1}(l), m, k).$$

Proof. Unpack the two continuation morphism records. The closed composite stores the append of the two visible tails as its displayed tail, which is definitionally one-headed. Choose the exposed payload append as k; the remaining continuation ledger is the canonical append continuation from $\mathsf{E1}(l)$ through m. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp visible tails result cases`

Theorem 61.284 (Continuation morphism visible-tail composite result is not empty)**.** *For histories a, b, c, l, m, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the two displayed tails are $\mathsf{hsame}(\mathsf{tail}(left), \mathsf{E1}(l))$ and $\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m))$, then the closed composite tail cannot be internally the same as e.*

Proof. Unpack the two continuation morphism records and substitute the visible-tail equalities. The closed composite tail is the append of two one-headed tails, which is itself one-headed. BEDC constructor separation rules out an internal history sameness proof from this one-headed tail to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp visible tails result not empty`

Theorem 61.285 (Continuation morphism composite tail nonempty cases)**.** *For histories a, b, c, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the closed composite tail is not internally empty, then at least one displayed factor tail is not internally empty:*

$$\neg\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Rightarrow \neg\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \vee \neg\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Proof. Unpack the two continuation morphism records and inspect the left displayed tail. If it is empty, the composite tail reduces to the right displayed tail, so right-tail emptiness would contradict the composite nonemptiness hypothesis. If it is zero-headed or one-headed, BEDC constructor separation rules out any equality from that left tail to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail nonempty cases`

Theorem 61.286 (Continuation morphism composite tail nonempty endpoint cases)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the closed composite tail is not internally empty, then at least one adjacent endpoint comparison fails:*

$$\neg\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Rightarrow \neg\mathsf{hsame}(a, b) \vee \neg\mathsf{hsame}(b, c).$$

Proof. First split composite tail nonemptiness into nonemptiness of one displayed factor tail. For each factor, use the empty-tail endpoint iff for continuation morphisms: an endpoint comparison would force the corresponding tail to be internally empty. This contradicts the selected tail-nonempty branch. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail nonempty endpoint cases`

Theorem 61.287 (Continuation morphism composite tail nonempty equivalence)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. The closed composite tail is not internally empty exactly when at least one displayed factor tail is not internally empty:*

$$\neg\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \iff \neg\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \vee \neg\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Proof. Unpack the two continuation morphism records. The composite tail is the append of the displayed factor tails, so the append nonempty factor equivalence gives the displayed statement. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail nonempty iff`

Theorem 61.288 (Continuation morphism composite target empty iff)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. The final target is empty exactly when both sources and both displayed factor tails are empty:*

$$\mathsf{hsame}(c, \mathsf{e}) \iff \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(b, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}).$$

Proof. If $c\mathsf{hsamee}$, transport the right continuation ledger to empty target and apply two-stage empty-target inversion. This yields empty source, empty middle, and empty tails for the two displayed factors.

Conversely, destruct the right continuation morphism. The empty middle and empty right-tail witnesses reduce its continuation ledger to the canonical empty continuation, forcing the target to be e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp target empty iff`

Theorem 61.289 (Continuation morphism composite nonempty tail rejects empty target)**.** *For histories a, b, c, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the closed composite tail is not internally empty, then the final target cannot be internally empty:*

$$\neg\mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{e}) \Rightarrow \neg\mathsf{hsame}(c, \mathsf{e}).$$

Proof. Transport the right continuation relation across an internal equality $c\mathsf{hsamee}$. The empty-target inversion for the two-stage continuation then forces both displayed tails to be internally empty. The composite-tail nonempty equivalence turns the composite nonemptiness hypothesis into nonemptiness of one displayed factor tail, contradicting the inversion data. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail nonempty target not empty`

Theorem 61.290 (Continuation morphism composite unary tails reject zero-headed result)**.** *For histories a, b, c, z, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If both displayed tails are unary histories, then the closed composite tail cannot be internally the same as $\mathsf{E0}(z)$.*

Proof. Unpack the two continuation morphism records. The closed composite tail is the append of the two component tails, and unary closure under append keeps that composite tail unary. A unary history cannot be internally the same as a zero-headed history, by the BEDC unary zero-exclusion rule. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp unary tail e0 absurd`

Theorem 61.291 (Continuation morphism composite one-headed tail left cases)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the composite displayed tail is classified by $\mathsf{E1}(k)$ and the left displayed tail is unary, then either the left displayed tail is empty and the right displayed tail is classified by $\mathsf{E1}(k)$, or the left displayed tail is itself one-headed and forms a continuation witness with the right displayed tail to $\mathsf{E1}(k)$.*

Proof. Unpack the two continuation morphism records and inspect the left tail. The empty case reduces the composite tail by left-unit append. The zero-headed case contradicts unary classification. In the one-headed case, the displayed composite equality is exactly the continuation witness from the exposed left tail and the right tail to $\mathsf{E1}(k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp e1 tail left cases`

Theorem 61.292 (Continuation morphism composite one-headed tail right cases)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the composite displayed tail is classified by $\mathsf{E1}(k)$ and the right displayed tail is unary, then either the right displayed tail is empty and the left displayed tail is classified by $\mathsf{E1}(k)$, or the right displayed tail is itself one-headed and forms a continuation witness from the left displayed tail to $\mathsf{E1}(k)$.*

Proof. Unpack the two continuation morphism records and inspect the right tail. The empty case reduces the composite tail by right-unit append. The zero-headed case contradicts unary classification. In the one-headed case, the displayed composite equality is exactly the continuation witness from the left tail and the exposed right tail to $\mathsf{E1}(k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp e1 tail right cases`

Theorem 61.293 (Continuation morphism composite zero-headed tail left cases)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the composite displayed tail is classified by $\mathsf{E0}(k)$ and the left displayed tail is unary, then either the left displayed tail is empty and the right displayed tail is classified by $\mathsf{E0}(k)$, or the left displayed tail is one-headed and forms a continuation witness with the right displayed tail to $\mathsf{E0}(k)$.*

Proof. Unpack the two continuation morphism records and inspect the left tail. The empty case reduces the composite tail by left-unit append. The zero-headed case contradicts unary classification of the left tail. In the one-headed case, the displayed composite equality is exactly the continuation witness from the exposed left tail and the right tail to $\mathsf{E0}(k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp e0 tail left cases`

Theorem 61.294 (Continuation morphism composite zero-headed tail right cases)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the composite displayed tail is classified by $\mathsf{E0}(k)$ and the right displayed tail is unary, then the right displayed tail is empty and the left displayed tail is classified by $\mathsf{E0}(k)$.*

Proof. Unpack the two continuation morphism records and inspect the right tail. The empty case reduces the composite tail by right-unit append. The zero-headed case contradicts unary classification of the right tail. The one-headed case would classify a one-headed composite tail as zero-headed, contradicting BEDC constructor separation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp e0 tail right cases`

Theorem 61.295 (Continuation morphism nonempty one-headed composite factors)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If both displayed tails are unary and nonempty, and the composite displayed tail is classified by $\mathsf{E1}(k)$, then there are histories l, m such that the left and right displayed tails are classified by $\mathsf{E1}(l)$ and $\mathsf{E1}(m)$, and*

$$\mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)).$$

Proof. Destructure both displayed tails. Empty tails contradict the corresponding nonempty hypotheses, and zero-headed tails contradict unary classification. The remaining one-headed–one-headed case exposes l and m directly, while the composite displayed equality supplies the continuation ledger. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp nonempty e1 tail factors`

Theorem 61.296 (Continuation morphism nonempty one-headed composite factors are unique)**.** *For histories a, b, c, k, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If both displayed tails are unary and nonempty, and the composite displayed tail is classified by $\mathsf{E1}(k)$, then there are histories l, m such that the left and right displayed tails are classified by $\mathsf{E1}(l)$ and $\mathsf{E1}(m)$, the visible boundary satisfies*

$$\mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)),$$

and any other visible payloads l', m' with the same two displayed tail classifications and the same visible boundary satisfy $\mathsf{hsame}(l, l') \wedge \mathsf{hsame}(m, m')$.

Proof. Destructure both displayed tails. Empty tails contradict nonemptiness and zero-headed tails contradict unary classification. In the remaining one-headed–one-headed case, the witnesses are the two exposed payloads. Constructor injectivity for $\mathsf{E1}$ turns any competing left and right tail classifications into the two payload equalities, while the displayed composite equality supplies the boundary continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp nonempty e1 tail factors unique`

Theorem 61.297 (Continuation morphism right one-headed tail target cases)**.** *For histories b, c, m, let $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the displayed tail of $right$ is internally the same as $\mathsf{E1}(m)$, then the target is one-headed:*

$$\exists r : \mathsf{BHist}.\ c = \mathsf{E1}(r) \wedge \mathsf{Cont}(b, m, r).$$

Proof. Unpack the continuation morphism record and replace its displayed tail by $\mathsf{E1}(m)$. The BEDC continuation step-result inversion then exposes the unique one-headed target payload r and the descended continuation $\mathsf{Cont}(b, m, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism right e1 tail target cases`

Theorem 61.298 (Continuation morphism right zero-headed tail target cases)**.** *For histories b, c, m, let $right : \mathsf{ContinuationMorphism}(b, c)$. If the displayed tail of $right$ is internally the same as $\mathsf{E0}(m)$, then the target is zero-headed:*

$$\exists r : \mathsf{BHist}.\ c = \mathsf{E0}(r) \wedge \mathsf{Cont}(b, m, r).$$

Proof. Unpack the continuation morphism record and replace its displayed tail by $\mathsf{E0}(m)$. The BEDC continuation step-result inversion then exposes the zero-headed target payload r together with the descended continuation $\mathsf{Cont}(b, m, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism right e0 tail target cases`

Theorem 61.299 (Continuation morphism composite right visible tail result cases)**.** *For histories a, b, c, m, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the right displayed tail is $\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m))$, then the closed composite tail is one-headed, and its exposed payload is the continuation of the left tail by m:*

$$\exists k : \mathsf{BHist}.\ \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E1}(k)) \wedge \mathsf{Cont}(\mathsf{tail}(left), m, k).$$

Proof. Unpack the two continuation morphism records. The closed composite stores the append of the left tail with the right tail as its displayed tail. Replacing the right tail by its $\mathsf{E1}(m)$ presentation makes the composite tail definitionally $\mathsf{E1}(\mathsf{append}(\mathsf{tail}(left), m))$. Choose this payload append as k; the remaining continuation ledger is the reflexive append continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right visible tail result cases`

Theorem 61.300 (Continuation morphism composite right zero tail result cases)**.** *For histories a, b, c, m, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the right displayed tail is $\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E0}(m))$, then the closed composite tail is zero-headed, and its exposed payload is the continuation of the left tail by m:*

$$\exists k : \mathsf{BHist}.\ \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E0}(k)) \wedge \mathsf{Cont}(\mathsf{tail}(left), m, k).$$

Proof. Unpack the two continuation morphism records. The closed composite stores the append of the left tail with the right tail as its displayed tail. Replacing the right tail by its $\mathsf{E0}(m)$ presentation makes the composite tail definitionally $\mathsf{E0}(\mathsf{append}(\mathsf{tail}(left), m))$. Choose this payload append as k; the remaining continuation ledger is the reflexive append continuation. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right e0 tail result cases`

Theorem 61.301 (Continuation morphism composite left-tail result cases)**.** *For histories a, b, c, l, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. If the left displayed tail is classified by l, then the closed composite tail is determined by the right displayed tail shape:*

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{tail}(left \circ right), l),$$

or there exists a history m such that

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E0}(m)) \wedge \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E0}(\mathsf{append}(l, m))),$$

or there exists a history m such that

$$\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m)) \wedge \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E1}(\mathsf{append}(l, m))).$$

Proof. Unpack the two continuation morphism records and replace the left displayed tail by its classified representative l. Splitting the right displayed tail into empty, zero-headed, and one-headed cases gives the three composite-tail shapes directly from the definition of append. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp left tail result cases`

Theorem 61.302 (Continuation morphism right zero tail composite result is not empty)**.** *For histories a, b, c, m, let $left$: $\mathsf{ContinuationMorphism}(a, b)$ and $right$: $\mathsf{ContinuationMorphism}(b, c)$. If the right displayed tail is $\mathsf{hsame}(\mathsf{tail}(right), \mathsf{E0}(m))$, then the closed composite tail cannot be internally the same as* e.

Proof. Unpack the two continuation morphism records and replace the right tail by its zero-headed presentation. The closed composite tail is then a zero-headed history, namely $\mathsf{E0}(\mathsf{append}(\mathsf{tail}(left), m))$. BEDC constructor separation rules out an internal sameness proof from this zero-headed history to e. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp right e0 tail result not empty`

Theorem 61.303 (Category hom-carrier one-headed target and morphism descent)**.** *For histories a, k, r, a hom-carrier whose target and morphism histories are both one-headed descends to the corresponding tail hom-carrier:*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)) \Longrightarrow \mathsf{CategoryHomCarrier}(a, r, k).$$

Proof. Unpack the hom-carrier fields. The source unary witness is unchanged. Unary inversion removes the visible constructor from both the target carrier and the morphism carrier, and the right step-rule inversion for Cont removes the same visible constructor from the continuation ledger. Repacking these fields gives $\mathsf{CategoryHomCarrier}(a, r, k)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target morphism exactness`

Theorem 61.304 (Category hom-carrier one-headed target and morphism exactness iff)**.** *For histories a, k, r, the simultaneous one-headed target and morphism presentation is exactly the tail hom-carrier:*

$$\mathsf{CategoryHomCarrier}(a, \mathsf{E1}(r), \mathsf{E1}(k)) \Longleftrightarrow \mathsf{CategoryHomCarrier}(a, r, k).$$

Proof. The forward implication is the one-headed target and morphism descent theorem. Conversely, a tail hom-carrier supplies the unary source, unary morphism tail, and tail continuation $\mathsf{Cont}(a, k, r)$. The one-headed morphism target exactness theorem repacks these data as the displayed one-headed target and morphism hom-carrier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 target morphism exactness iff`

Theorem 61.305 (Category hom-carrier one-headed source and morphism target exactness)**.** *For histories a, k, t,*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), t, \mathsf{E1}(k))\\ &\quad\iff\quad \exists r,\ t = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).\end{aligned}$$

Proof. Unpack the hom-carrier witness. The continuation field forces the target to have a visible one constructor; removing that constructor gives $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. The source, morphism, and target unary witnesses invert through the same visible constructors. Conversely, the displayed target equation, the three unary witnesses, and the one-step continuation rule rebuild the hom-carrier fields. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 morphism target iff`

Corollary 61.306 (Continuation morphism composite visible tails consumer determinacy)**.** *For histories $a, b, c, k, l, m, l', m' : \mathsf{BHist}$, let $left : \mathsf{ContinuationMorphism}(a, b)$ and $right : \mathsf{ContinuationMorphism}(b, c)$. Assume the fixed nonempty visible composite surface*

$$\begin{aligned}&\mathsf{UnaryHistory}(\mathsf{tail}(left)) \wedge \mathsf{UnaryHistory}(\mathsf{tail}(right)) \wedge \neg\mathsf{hsame}(\mathsf{tail}(left), \mathsf{e}) \wedge \neg\mathsf{hsame}(\mathsf{tail}(right), \mathsf{e})\\ &\quad\wedge \mathsf{hsame}(\mathsf{tail}(left \circ right), \mathsf{E1}(k)) \wedge \mathsf{hsame}(\mathsf{tail}(left), \mathsf{E1}(l)) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m))\\ &\quad\wedge \mathsf{Cont}(\mathsf{E1}(l), \mathsf{E1}(m), \mathsf{E1}(k)).\end{aligned}$$

If another accepted consumer-facing use of the same composite surface reads the same composite payload through visible factor tails

$$\mathsf{hsame}(\mathsf{tail}(left), \mathsf{E1}(l')) \wedge \mathsf{hsame}(\mathsf{tail}(right), \mathsf{E1}(m')) \wedge \mathsf{Cont}(\mathsf{E1}(l'), \mathsf{E1}(m'), \mathsf{E1}(k)),$$

then the consumer-visible factor route is determined:

$$\mathsf{hsame}(l, l') \wedge \mathsf{hsame}(m, m').$$

Consequently, once the displayed unary, nonempty, composite-tail, factor-tail, and boundary-continuation rows are fixed, every accepted route or one-headed normal-form read for the composite visible tails uses the same finite route through l, m, k up to hsame*. There is no alternate hidden coordinate for the fixed displayed continuation surface.*

Proof. Apply Definition 61.296 to $left$, $right$, and the fixed unary, nonempty, and composite one-headed rows. It supplies the canonical visible factor pair for the composite tail and a universal comparison clause for any other pair of one-headed factor reads with the same boundary continuation. The fixed pair l, m and the competing pair l', m' both satisfy that clause: their tail classifications are the displayed factor-tail rows, and their boundary continuations are the two displayed $\mathsf{Cont}(\mathsf{E1}(-), \mathsf{E1}(-), \mathsf{E1}(k))$ rows. Comparing each pair with the canonical pair gives internal sameness witnesses from the fixed pair to the canonical pair and from the competing pair to the canonical pair. The symmetry and transitivity clauses of Definition 3.8 orient and compose these hsame witnesses inside the generated-history relation, so the two left payloads satisfy $\mathsf{hsame}(l, l')$ and the two right payloads satisfy $\mathsf{hsame}(m, m')$. This proves the displayed conjunction.

The final consumer-facing sentence only restates this internally composed hsame comparison for any accepted route or one-headed normal-form read that keeps the same listed rows; it does not substitute host equality or add another coordinate. □

Theorem 61.307 (Category hom-carrier one-headed endpoint morphism payload determinacy)**.** *For histories a, a', r, r', k, k', if* $\mathsf{hsame}(a, a')$ *and* $\mathsf{hsame}(r, r')$*, then any two hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), \mathsf{E1}(k)) \quad \textit{and} \quad \mathsf{CategoryHomCarrier}(\mathsf{E1}(a'), \mathsf{E1}(r'), \mathsf{E1}(k'))$$

have internally same visible morphism payloads: $\mathsf{hsame}(k, k')$*.*

Proof. Lift the source and target endpoint equalities through the one-headed constructor. Endpoint-sensitive morphism determinacy gives $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(k'))$ for the displayed morphisms. One-headed constructor inversion removes the visible tag and yields $\mathsf{hsame}(k, k')$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target morphism payload deterministic`

Theorem 61.308 (Category hom-carrier one-headed source empty target impossibility)**.** *For histories a, f, no hom-carrier can have source* $\mathsf{E1}(a)$ *and target* e*:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{e}, f) \to \mathsf{False}.$$

Proof. Unpack the hom-carrier witness and classify the morphism history f. In each constructor branch, the continuation field would identify the empty target with an append whose source already has a visible one constructor, which is impossible by generated-history no-confusion. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source empty target absurd`

Theorem 61.309 (Category visible source empty-target composite impossibility)**.** *For histories a, b, f, g, the pair of hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), b, f) \wedge \mathsf{CategoryHomCarrier}(b, \mathsf{e}, g)$$

is impossible.

Proof. Compose the first displayed hom-carrier with the second one using the concrete continuation witness $\mathsf{Cont}(f, g, \mathsf{append}(f, g))$. Composition closure produces

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{e}, \mathsf{append}(f, g)).$$

This contradicts the one-headed source empty-target impossibility theorem. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier visible source empty target comp absurd`

Theorem 61.310 (Category hom-carrier one-headed source morphism cases)**.** *For histories $a, target, morph$, a hom-carrier with source* $\mathsf{E1}(a)$ *has one of the following forms:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), target, morph) \Longrightarrow (morph = \mathsf{e} \wedge target = \mathsf{E1}(a) \wedge \mathsf{UnaryHistory}(a))$$

or there exist histories k, r such that

$$morph = \mathsf{E1}(k) \wedge target = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Proof. Unpack the hom-carrier witness into unary source, target, and morphism fields and the continuation ledger. Classify the morphism history. The empty branch forces the target to be exactly $\mathsf{E1}(a)$ by the right-unit continuation shape, and unary inversion gives unary history for a. The zero-headed morphism branch contradicts unary-history separation for zero extensions. In the one-headed branch, classify the target history. Empty and zero-headed targets contradict the continuation ledger; a one-headed target exposes histories k and r, unary inversion gives the three unary witnesses, and constructor inversion on the continuation ledger yields $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source morphism cases`

Theorem 61.311 (Category hom-carrier one-headed source morphism cases iff)**.** *For histories $a, target, morph$, the one-headed-source hom-carrier condition is equivalent to the same two concrete cases:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), target, morph) \iff [(morph = \mathsf{e} \wedge target = \mathsf{E1}(a) \wedge \mathsf{UnaryHistory}(a)) \vee \exists k, r, morph =$$

Proof. The forward implication is the one-headed source morphism case theorem. For the reverse implication, the empty branch is the empty identity hom-carrier on $\mathsf{E1}(a)$, and the visible branch rebuilds the hom-carrier from the three unary witnesses together with the one-step continuation rule. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source morphism cases iff`

Theorem 61.312 (Category one-headed endpoint visible morphism exactness)**.** *For histories a, r, k, a visible morphism between one-headed source and target histories is exactly unary data on the three visible tails together with the tail continuation:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), \mathsf{E1}(r), \mathsf{E1}(k)) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(a), k, r).$$

Proof. Apply the one-headed morphism-target exactness theorem to the displayed target $\mathsf{E1}(r)$ and morphism $\mathsf{E1}(k)$. The source carrier and target carrier are read back by one-headed unary inversion, while the reverse direction packages the source carrier as $\mathsf{E1}(a)$ and applies the same morphism-target exactness theorem. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source e1 target visible morphism iff`

Theorem 61.313 (Category one-headed source composite target visibility)**.** *For histories a, b, c, f, g, fg, composing a hom-carrier with source $\mathsf{E1}(a)$ through a second displayed hom-carrier preserves one-headed visibility of the composite target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), b, f) \Longrightarrow \mathsf{CategoryHomCarrier}(b, c, g) \Longrightarrow \mathsf{Cont}(f, g, fg) \Longrightarrow \exists r,\ c = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(r).$$

Proof. Composition closure first turns the two displayed hom-carriers and the continuation ledger into a composite hom-carrier $\mathsf{CategoryHomCarrier}(\mathsf{E1}(a), c, fg)$. Applying the one-headed source morphism case theorem to this composite gives either an empty morphism, where the target is $\mathsf{E1}(a)$, or a visible morphism, where the target is $\mathsf{E1}(r)$ for a unary history r. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier e1 source comp target visible`

Primitive boundary. $\mathsf{Cat}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 61.314 (Zero-axiom encoding note)**.** *Avoids a global universe of all objects by using explicit licensed carriers.*

The Category hom-carrier tail surface consumes the finite continuation packet exported by chapter 485. The Category side reads hom carriers through continuation tails, while $\mathsf{ContinuationMonad}^{\uparrow}$ supplies the root source rows and the bind and unit route rows that make those tails available to downstream consumers.

Theorem 61.315 (Continuation morphism fixed tail iff)**.** *For histories a, b, t : BHist and a fixed continuation morphism m : $\mathsf{ContinuationMorphism}(a, b)$,*

$$\mathsf{hsame}(\mathsf{tail}(m), t) \iff \mathsf{Cont}(a, t, b).$$

Proof. The forward direction transports the relation field $\mathsf{Cont}(a, \mathsf{tail}(m), b)$ along the displayed tail comparison, with source and target held fixed. Conversely, if t is a continuation tail from a to b, then the relation field of m and this new continuation share the same source and target, so continuation left cancellation identifies $\mathsf{tail}(m)$ with t. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail hsame iff`

Theorem 61.316 (Category hom carrier unary prefix exactness)**.** *For a unary prefix p, a displayed hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$ is equivalent to the unprefixed hom-carrier from a to b, together with the same morphism history.*

Proof. The forward direction cancels the common unary prefix in the two endpoint carriers and in the continuation ledger. The reverse direction uses unary append closure on both endpoints and prefixes the same continuation ledger. □

Theorem 61.317 (Category hom carrier and continuation morphism tail correspondence)**.** *For histories a, b, f : BHist, the concrete hom-carrier witness is exactly a unary endpoint-and-tail presentation of a continuation morphism:*

$$\mathsf{CategoryHomCarrier}(a, b, f) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{UnaryHistory}(f) \wedge \exists m : \mathsf{ContinuationMorphis}$$

Proof. In the forward direction, unpack the hom-carrier witness. Its displayed fields give unary membership for a, b, and f, together with the continuation ledger $\mathsf{Cont}(a, f, b)$. Taking the continuation morphism whose tail is f and whose relation field is this ledger gives the required existential witness, and reflexivity of hsame classifies the displayed tail.

Conversely, unpack the existential continuation morphism m. Its record contains the displayed continuation witness $\mathsf{Cont}(a, \mathsf{tail}(m), b)$, while the hypothesis supplies $\mathsf{hsame}(\mathsf{tail}(m), f)$ and the three unary witnesses.

Apply Theorem 4.41 with reflexive source and target endpoint classifiers and the displayed tail classifier to transport $\mathsf{Cont}(a, \mathsf{tail}(m), b)$ to $\mathsf{Cont}(a, f, b)$. Reassemble $\mathsf{CategoryHomCarrier}(a, b, f)$ from the unary source, unary target, unary morphism, and transported continuation fields. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier continuation morphism tail iff`
↪ variant `BEDC.Derived.CategoryUp.CategoryHomCarrier continuationMorphism tail iff`
↪ variant `BEDC.Derived.CategoryUp.CategoryHomCarrier continuation morphism tail correspondence`

Theorem 61.318 (Continuation morphism tail identity square closure)**.** *For a continuation morphism m : $\mathsf{ContinuationMorphism}(a, b)$, if $left$ is obtained by composing e before $\mathsf{tail}(m)$ and $right$ is obtained by composing e after $\mathsf{tail}(m)$, then $\mathsf{hsame}(left, right)$ and both displayed tails carry continuations from a to b.*

Proof. The left-unit continuation identifies $left$ with $\mathsf{tail}(m)$. Determinism against the right-unit continuation identifies $right$ with $\mathsf{tail}(m)$. Transport the relation field of m across these two history equalities and compose the equalities by transitivity. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail identity square closed`

Theorem 61.319 (Continuation morphism composite endpoint transport)**.** *For continuation morphisms $left : a \to b$, $right : b \to c$, $left' : a' \to b'$, and $right' : b' \to c'$, if $\mathsf{hsame}(a, a')$ and $\mathsf{hsame}(c, c')$, then the canonical composite tail of $left$ and $right$ transports to a continuation from a' to c', and it is hsame to the canonical composite tail of $left'$ and $right'$.*

Proof. Transport the composite continuation field of $left \circ right$ across the source and target endpoint comparisons, keeping its displayed tail fixed. The transported continuation and the composite continuation field of $left' \circ right'$ share source and target, so finite-kernel left cancellation identifies their tails. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint hsame tail transport`

Theorem 61.320 (Continuation morphism composite endpoint transport existence)**.** *For continuation morphisms $m : a \to b$ and $n : b \to c$, endpoint comparisons $a\mathsf{hsame}a'$, $b\mathsf{hsame}b'$, and $c\mathsf{hsame}c'$ determine transported factor morphisms $m' : a' \to b'$ and $n' : b' \to c'$ such that the transported composite tail is history-same to the original composite tail and carries a continuation from a' to c'.*

Proof. Keep the displayed tails of m and n. Transport the relation field of m across the source and middle endpoint comparisons, and transport the relation field of n across the middle and target endpoint comparisons. The composite of these transported factor morphisms has the same appended tail as the original composite by reflexivity, and its own relation field supplies the continuation from a' to c'. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp endpoint transport exists`

Theorem 61.321 (Continuation morphism composite tail append determinacy)**.** *For continuation morphisms $left, left' : a \to b$ and $right, right' : b \to c$, the appended composite tail is determined by the endpoints:*

$$\mathsf{hsame}(\mathsf{append}(\mathsf{tail}(left), \mathsf{tail}(right)), \mathsf{append}(\mathsf{tail}(left'), \mathsf{tail}(right'))).$$

Proof. Continuation left cancellation identifies the two displayed tails from a to b, and again identifies the two displayed tails from b to c. Congruence of BEDC append combines these two tail comparisons into the composite-tail comparison. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail append deterministic`

Theorem 61.322 (Continuation morphism composite tail unary closure)**.** *For continuation morphisms $left : a \to b$ and $right : b \to c$, if both displayed factor tails are unary histories, then the closed composite tail of $left \circ right$ is a unary history.*

Proof. Unpack the two continuation morphism records. The closed composite tail is $\mathsf{append}(\mathsf{tail}(left), \mathsf{tail}(right))$, so unary append closure applies to the two supplied tail carriers. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism comp tail unary closed`

Theorem 61.323 (Continuation morphism tail factorization iff)**.** *For histories $a, c, l, r, t : \mathsf{BHist}$, if $\mathsf{Cont}(l, r, t)$, then a continuation morphism from a to c whose displayed tail is t exists exactly when there is a middle history b carrying a left continuation morphism from a to b with tail l and a right continuation morphism from b to c with tail r.*

Proof. For the forward implication, unpack the composite morphism and the displayed split $\mathsf{Cont}(l, r, t)$. The middle object is $\mathsf{append}(a, l)$. The left factor is the reflexive continuation from a through l to that middle, and the right factor is obtained by reassociating the composite continuation witness through the displayed split.

For the reverse implication, compose the two displayed factor morphisms. The closed composite tail is the append of the two factor tails; transport that tail along the two supplied tail classifiers and the split $\mathsf{Cont}(l, r, t)$ to obtain the required composite tail classifier. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism tail factorization iff`

Theorem 61.324 (Continuation morphism identity and composition closure)**.** *Every history h has an empty-tail identity witness in* $\mathsf{ContinuationMorphism}(h, h)$*. Moreover, composable continuation morphisms $left : a \to b$ and $right : b \to c$ determine an inhabited* $\mathsf{ContinuationMorphism}(a, c)$ *by the closed composite.*

Proof. For the identity clause, take the empty tail. The continuation witness is the empty right-unit continuation at h, so the corresponding continuation morphism inhabits $\mathsf{ContinuationMorphism}(h, h)$.

For the composition clause, apply $\mathsf{ContinuationMorphism_comp_closed}$ to $left$ and $right$. The result is the closed composite continuation morphism from a to c, which supplies the required inhabitant. □

Lean4 ✓ `BEDC.Derived.CategoryUp.ContinuationMorphism identity comp closure`

Per-object certificate: traditional target hom-as-continuation morphism bridge; BEDC source $\mathsf{CategoryHomCarrier}$, $\mathsf{ContinuationMorphism}$, $\mathsf{UnaryHistory}$, and Cont; carrier BHist tails; classifier hsame; stability obligation continuation transport; ledger the displayed tail and continuation witness; core theorem this correspondence; exactness predicate the iff; standard bridge deferred; status $\mathsf{ProofTarget}$.

Theorem 61.325 (Unary-continuation category stability law package)**.** *For the concrete unary-continuation category on* BHist*, with object carrier the unary histories, hom-carrier* $\mathsf{CategoryHomCarrier}(a, b, f)$*, object classifier* hsame*, and morphism classifier* hsame *on displayed hom histories, the* $\mathsf{Cat}^{\uparrow}$ *stability laws are satisfied. At every unary object the empty history is an identity hom-carrier, composition of displayed hom-carriers is closed, any two associations of a threefold displayed composite classify together, the left and right empty-composition laws hold under the morphism classifier, and morphism classification is preserved after endpoint* hsame *transport.*

Proof. The identity and binary composition fields are the concrete hom-carrier laws. The empty history is an identity hom-carrier at each unary object by Definition 61.2, and displayed composition closure is exactly Definition 61.19.

For associativity, use the two-step closure supplied by Definition 61.132. It builds both associated composites as hom-carriers over the same endpoints and identifies their morphism histories by the classifier hsame.

The unit laws are supplied by the identity square. Applying Definition 61.95 to a displayed hom-carrier and the two empty-composition ledgers gives the left and right unit composites and the required morphism-classifier equality.

It remains to record stability under endpoint transport. Transporting the source endpoint, target endpoint, and morphism history along hsame keeps the displayed hom-carrier witness by Definition 61.93; once the endpoints have been transported, any two displayed morphism witnesses over classified endpoints are identified by Definition 61.185. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier unary continuation stability law package`

Corollary 61.326 (Category hom carrier stability law package)**.** *For the same unary-continuation category on* BHist*, the identity hom-carriers, binary composition closure, associativity of displayed composition, and empty-composition unit fields form the basic* $\mathsf{Cat}^{\uparrow}$ *hom-carrier stability package.*

Proof. Project the identity, binary composition closure, associativity, and empty-composition unit clauses from Theorem 61.325. These are exactly the four displayed fields of the basic $\mathsf{Cat}^{\uparrow}$ hom-carrier stability package. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier stability law package`

Definition 61.327 (Opposite category certificate data)**.** *Let a* $\mathsf{Cat}^{\uparrow}$ *certificate* $\mathcal{C}$ *have object carrier* $\mathcal{O}$*, object classifier* $\sim_{\mathcal{O}}$*, hom-carrier* $\mathcal{H}(a,b,f)$*, morphism classifier* $\sim_{\mathcal{H}}$*, identity witnesses* id_a*, and composition relation* $\mathsf{Comp}_{\mathcal{C}}(f,g,h)$ *for displayed composable morphisms* $f : a \to b$ *and* $g : b \to c$*. The opposite certificate data* $\mathcal{C}^{\mathrm{op}}$ *are defined by*

$$\begin{aligned} \mathcal{O}^{\mathrm{op}}(a) &:= \mathcal{O}(a), & a \sim_{\mathcal{O}}^{\mathrm{op}} a' &:= a \sim_{\mathcal{O}} a', \\ \mathcal{H}^{\mathrm{op}}(a,b,f) &:= \mathcal{H}(b,a,f), & f \sim_{\mathcal{H}}^{\mathrm{op}} g &:= f \sim_{\mathcal{H}} g, \\ \mathsf{id}_a^{\mathrm{op}} &:= \mathsf{id}_a, & \mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(f,g,h) &:= \mathsf{Comp}_{\mathcal{C}}(g,f,h). \end{aligned}$$

Theorem 61.328 (Opposite category certificate)**.** *If* $\mathcal{C}$ *satisfies the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations of Definition 61.1, then the data of Definition 61.327 also satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations. Thus reversing hom-carriers and composition produces a* $\mathsf{Cat}^{\uparrow}(\mathcal{C}^{\mathrm{op}})$ *certificate with the same object carrier, identity witnesses, object classifier, and morphism classifier.*

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryHomCarrier opposite comp assoc closed`

Proof. The object carrier and object classifier fields are literal reuses of the corresponding fields of $\mathcal{C}$, so their carrier and classifier laws are unchanged. For a fixed ordered pair (a,b) in the opposite data, the hom-carrier predicate is the original predicate for the ordered pair (b,a). The corresponding $\mathsf{NameCert}$ source, pattern, ledger, and classifier laws are therefore exactly the original hom-carrier certificate fields with the endpoint order swapped.

The identity field is also inherited. At an object a,

$$\mathcal{H}^{\mathrm{op}}(a,a,\mathsf{id}_a^{\mathrm{op}}) = \mathcal{H}(a,a,\mathsf{id}_a),$$

which is the original identity hom-carrier field.

For composition closure, let $f : a \to b$ and $g : b \to c$ be displayed in $\mathcal{C}^{\mathrm{op}}$. This means $\mathcal{H}(b,a,f)$ and $\mathcal{H}(c,b,g)$ in $\mathcal{C}$. If $\mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(f,g,h)$ holds, then by definition $\mathsf{Comp}_{\mathcal{C}}(g,f,h)$ holds. The original composition closure in $\mathcal{C}$ gives $\mathcal{H}(c,a,h)$, which is exactly $\mathcal{H}^{\mathrm{op}}(a,c,h)$.

For associativity, take composable opposite morphisms $f : a \to b$, $g : b \to c$, and $h : c \to d$, with displayed composites fg, gh, $left$, and $right$. The four opposite composition ledgers are, in $\mathcal{C}$,

$$\mathsf{Comp}_{\mathcal{C}}(g,f,fg), \quad \mathsf{Comp}_{\mathcal{C}}(h,g,gh), \quad \mathsf{Comp}_{\mathcal{C}}(h,fg,left), \quad \mathsf{Comp}_{\mathcal{C}}(gh,f,right).$$

Apply associativity in $\mathcal{C}$ to the reversed chain h,g,f. It compares $\mathsf{Comp}_{\mathcal{C}}(gh,f,right)$ with $\mathsf{Comp}_{\mathcal{C}}(h,fg,left)$, giving $right \sim_{\mathcal{H}} left$. Symmetry of the morphism classifier gives $left \sim_{\mathcal{H}}^{\mathrm{op}} right$.

For unit laws, let $f : a \to b$ be displayed in the opposite data, so $f : b \to a$ in $\mathcal{C}$. The left unit law in $\mathcal{C}^{\mathrm{op}}$ is $\mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(\mathsf{id}_a,f,left)$, which is $\mathsf{Comp}_{\mathcal{C}}(f,\mathsf{id}_a,left)$; this is the original right unit law.

The right unit law in $\mathcal{C}^{\mathrm{op}}$ is $\mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(f, \mathsf{id}_b, right)$, which is $\mathsf{Comp}_{\mathcal{C}}(\mathsf{id}_b, f, right)$; this is the original left unit law. Hence both opposite unit composites are classified with f.

Endpoint transport is inherited with the same swap. A transported opposite hom-carrier $\mathcal{H}^{\mathrm{op}}(a, b, f)$ is the original carrier $\mathcal{H}(b, a, f)$, so source transport in the opposite data is target transport in $\mathcal{C}$, and target transport in the opposite data is source transport in $\mathcal{C}$. The morphism classifier is unchanged, so morphism transport and morphism determinacy are inherited directly.

Specializing $\mathcal{C}$ to the unary-continuation law package of Definition 61.325 uses the identity field from Definition 61.2, the composition field from Definition 61.19, the associativity field from Definition 61.132, the unit classification from Definition 61.95, and endpoint transport from Definition 61.93. □

↪variant BEDC.Derived.CategoryUp.CategoryHomCarrier opposite comp assoc closed

Theorem 61.329 (Standard unary-continuation category bridge)**.** *The public unary-continuation* $\mathsf{Cat}^{\uparrow}$ *export supplies a paper-level standard bridge to a host category interface, with six displayed fields:*

1. *object translation reads host objects through* UnaryHistory*;*
2. *morphism translation reads host arrows through* CategoryHomCarrier*;*
3. *classifier compatibility is exactly endpoint and morphism comparison by* hsame*;*
4. *theorem correspondence pairs the host identity, composition, associativity, and unit rows with the unary-continuation* $\mathsf{Cat}^{\uparrow}$ *law package;*
5. *reflection is restricted to displayed BEDC histories and their hom-carrier witnesses;*
6. *the no-host-leak field asserts that host category structure is read only through these BEDC predicates, with no host equality, primitive category universe, or host carrier constructor entering the* $\mathsf{Cat}^{\uparrow}$ *public surface.*

Proof. For objects, the bridge reads a host object only when the displayed BEDC history satisfies UnaryHistory. For arrows, it reads a host morphism only when the displayed source, target, and arrow history satisfy CategoryHomCarrier. The carrier exactness is the concrete hom-carrier and continuation-morphism tail correspondence of Definition 61.317, and classifier compatibility is the same hsame relation already used by the object and morphism surfaces.

The theorem-preservation field is the public unary-continuation stability law package of Definition 61.325. Its identity, composition, associativity, and unit clauses are precisely the rows expected by the host category interface. The semantic certificate target CategoryHomCarrier_comp_result_semanticNameCer supplies the displayed BEDC ledger for the composition-result name certificate, so the host composition row is reflected through the same BEDC source, pattern, ledger, and classifier fields.

Reflection is therefore limited to histories displayed by UnaryHistory and morphism histories displayed by CategoryHomCarrier. The bridge never asks the host for a primitive object equality, a primitive category universe, or an external morphism classifier; it only reads the host interface through the six BEDC predicates listed above. This is the paper-level no-host-leak argument required for the standard bridge. □

Lean4 ✓ BEDC.Derived.CategoryUp.CategoryUp StdBridge

Theorem 61.330 (Double opposite category certificate data identity)**.** *Let* $\mathcal{C}$ *be a* $\mathsf{Cat}^{\uparrow}$ *certificate satisfying the certificate obligations of Definition 61.1, with object carrier* $\mathcal{O}$*, object classifier* $\sim_{\mathcal{O}}$*, hom-carrier* $\mathcal{H}(a, b, f)$*, morphism classifier* $\sim_{\mathcal{H}}$*, identity witnesses* id_a*, and composition relation*

$\mathsf{Comp}_{\mathcal{C}}$. *Let $\mathcal{C}^{\mathrm{opop}} = (\mathcal{C}^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 61.327 twice. Then $\mathcal{C}^{\mathrm{opop}}$ is a $\mathsf{Cat}^{\uparrow}$ certificate, and its object carrier, object classifier, hom-carrier predicate, morphism classifier, identity witnesses, and composition relation are componentwise identical to those of $\mathcal{C}$:*

$$\begin{aligned} \mathcal{O}^{\mathrm{opop}}(a) &= \mathcal{O}(a), & a \sim_{\mathcal{O}}^{\mathrm{opop}} a' &= a \sim_{\mathcal{O}} a', \\ \mathcal{H}^{\mathrm{opop}}(a,b,f) &= \mathcal{H}(a,b,f), & f \sim_{\mathcal{H}}^{\mathrm{opop}} g &= f \sim_{\mathcal{H}} g, \\ \mathsf{id}_a^{\mathrm{opop}} &= \mathsf{id}_a, & \mathsf{Comp}_{\mathcal{C}^{\mathrm{opop}}}(f,g,h) &= \mathsf{Comp}_{\mathcal{C}}(f,g,h). \end{aligned}$$

Proof. Because $\mathcal{C}$ satisfies the $\mathsf{Cat}^{\uparrow}$ obligations, Definition 61.328 makes $\mathcal{C}^{\mathrm{op}}$ a certified $\mathsf{Cat}^{\uparrow}$ instance. Applying the same theorem to that certified opposite data makes $\mathcal{C}^{\mathrm{opop}}$ a certified $\mathsf{Cat}^{\uparrow}$ instance.

It remains only to compare the displayed data. For the object carrier and object classifier, two unfoldings of Definition 61.327 give $\mathcal{O}^{\mathrm{opop}}(a) = \mathcal{O}^{\mathrm{op}}(a) = \mathcal{O}(a)$ and $a \sim_{\mathcal{O}}^{\mathrm{opop}} a' = a \sim_{\mathcal{O}}^{\mathrm{op}} a' = a \sim_{\mathcal{O}} a'$.

For hom-carriers, the first opposite swaps the endpoints and the second opposite swaps them back:

$$\mathcal{H}^{\mathrm{opop}}(a,b,f) = \mathcal{H}^{\mathrm{op}}(b,a,f) = \mathcal{H}(a,b,f).$$

The morphism classifier is not changed by either opposite construction, hence $f \sim_{\mathcal{H}}^{\mathrm{opop}} g = f \sim_{\mathcal{H}} g$. The identity row is also reused at both stages, so $\mathsf{id}_a^{\mathrm{opop}} = \mathsf{id}_a^{\mathrm{op}} = \mathsf{id}_a$.

For composition, the two argument reversals cancel:

$$\mathsf{Comp}_{\mathcal{C}^{\mathrm{opop}}}(f,g,h) = \mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(g,f,h) = \mathsf{Comp}_{\mathcal{C}}(f,g,h).$$

Thus every listed data field of the double opposite certificate is the original field of $\mathcal{C}$. □

Theorem 61.331 (Opposite category certificate involution)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate satisfying the obligations of Definition 61.1. Write its object carrier as $\mathcal{O}$, object classifier as $\sim_{\mathcal{O}}$, hom-carrier predicate as $\mathcal{H}(a,b,f)$, morphism classifier as $\sim_{\mathcal{H}}$, identity witnesses as id_a, and composition classifier as $\mathsf{Comp}_{\mathcal{C}}(f,g,h)$. Let $\mathcal{C}^{\mathrm{opop}} = (\mathcal{C}^{\mathrm{op}})^{\mathrm{op}}$ be obtained by applying Definition 61.327 twice. Then $\mathcal{C}^{\mathrm{opop}}$ is a $\mathsf{Cat}^{\uparrow}$ certificate, and its object carrier, object classifier, hom-carrier predicate, morphism classifier, identity witnesses, and composition classifier reduce componentwise to those of $\mathcal{C}$:*

$$\begin{aligned} \mathcal{O}^{\mathrm{opop}}(a) &= \mathcal{O}(a), & a \sim_{\mathcal{O}}^{\mathrm{opop}} a' &= a \sim_{\mathcal{O}} a', \\ \mathcal{H}^{\mathrm{opop}}(a,b,f) &= \mathcal{H}(a,b,f), & f \sim_{\mathcal{H}}^{\mathrm{opop}} g &= f \sim_{\mathcal{H}} g, \\ \mathsf{id}_a^{\mathrm{opop}} &= \mathsf{id}_a, & \mathsf{Comp}_{\mathcal{C}^{\mathrm{opop}}}(f,g,h) &= \mathsf{Comp}_{\mathcal{C}}(f,g,h). \end{aligned}$$

Proof. Since $\mathcal{C}$ satisfies the $\mathsf{Cat}^{\uparrow}$ obligations, Definition 61.328 gives a $\mathsf{Cat}^{\uparrow}$ certificate on $\mathcal{C}^{\mathrm{op}}$. Applying the same theorem to that certificate gives a $\mathsf{Cat}^{\uparrow}$ certificate on $\mathcal{C}^{\mathrm{opop}} = (\mathcal{C}^{\mathrm{op}})^{\mathrm{op}}$.

The field comparison is obtained by two unfoldings of Definition 61.327. The object carrier and object classifier are reused by each opposite construction, so $\mathcal{O}^{\mathrm{opop}}(a) = \mathcal{O}^{\mathrm{op}}(a) = \mathcal{O}(a)$ and $a \sim_{\mathcal{O}}^{\mathrm{opop}} a' = a \sim_{\mathcal{O}}^{\mathrm{op}} a' = a \sim_{\mathcal{O}} a'$.

For hom-carriers, the first opposite reverses the endpoint order and the second opposite reverses it back:

$$\mathcal{H}^{\mathrm{opop}}(a,b,f) = \mathcal{H}^{\mathrm{op}}(b,a,f) = \mathcal{H}(a,b,f).$$

The morphism classifier is unchanged at both stages, hence $f \sim_{\mathcal{H}}^{\mathrm{opop}} g = f \sim_{\mathcal{H}} g$. The identity witnesses are also reused at both stages, so $\mathsf{id}_a^{\mathrm{opop}} = \mathsf{id}_a^{\mathrm{op}} = \mathsf{id}_a$.

For composition, the defining reversal is applied twice. Thus, for displayed composable morphisms $f : a \to b$ and $g : b \to c$,

$$\mathsf{Comp}_{\mathcal{C}^{\mathrm{opop}}}(f, g, h) = \mathsf{Comp}_{\mathcal{C}^{\mathrm{op}}}(g, f, h) = \mathsf{Comp}_{\mathcal{C}}(f, g, h).$$

Every displayed data field of the double opposite certificate is therefore the corresponding displayed data field of $\mathcal{C}$. □

Definition 61.332 (Category split epimorphism witness)**.** *With the notation of Definition 61.1, let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate and let $\mathcal{H}(a, b, f)$ display a hom-carrier $f : a \to b$. A split-epimorphism witness for f consists of a hom-carrier $\mathcal{H}(b, a, s)$, a displayed composite $sf : b \to b$ with $\mathsf{Comp}_{\mathcal{C}}(s, f, sf)$, and a morphism-classifier comparison $sf \sim_{\mathcal{H}} \mathsf{id}_b$ to the identity witness at b.*

Theorem 61.333 (Category split epimorphism right cancellation)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate as in Definition 61.1. Suppose $\mathcal{H}(a, b, f)$ has a split-epimorphism witness (s, sf) in the sense of Definition 61.332. For hom-carriers $\mathcal{H}(b, x, u)$ and $\mathcal{H}(b, x, v)$, let fu and fv be displayed composites of u after f and v after f:*

$$\mathsf{Comp}_{\mathcal{C}}(f, u, fu), \qquad \mathsf{Comp}_{\mathcal{C}}(f, v, fv).$$

If $fu \sim_{\mathcal{H}} fv$, then $u \sim_{\mathcal{H}} v$.

Lean4 ✓ `BEDC.Derived.CategoryUp.CategorySplitEpiWitness right cancellative`

Proof. Let sf be the split composite and let $sf \sim_{\mathcal{H}} \mathsf{id}_b$ be its classifier comparison with the identity at b. Precompose the comparison $fu \sim_{\mathcal{H}} fv$ by the section s. The classifier-respecting composition field of the $\mathsf{Cat}^{\uparrow}$ certificate gives classifier equality between the two composites fu after s and fv after s.

Consider the u side. Associativity for the displayed triple $s : b \to a$, $f : a \to b$, and $u : b \to x$ identifies the right-associated composite fu after s with the left-associated composite u after sf. The split comparison replaces sf by id_b, again by classifier-respecting composition. The left-unit law then classifies u after id_b with u itself. Hence the composite fu after s is classified with u.

The same argument with v shows that the composite fv after s is classified with v. Combining these two reductions with the classifier equality between fu after s and fv after s, and using symmetry and transitivity of $\sim_{\mathcal{H}}$, gives $u \sim_{\mathcal{H}} v$. □

Theorem 61.334 (Category split epimorphism composition closure)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate satisfying the obligations of Definition 61.203. With the notation of Definition 61.1, let $\mathcal{H}(a, b, f)$ and $\mathcal{H}(b, c, g)$ display hom-carriers. Suppose f has a split-epimorphism witness with right-inverse hom-carrier $f^{\dagger} : b \to a$ and split composite $s_f : b \to b$,*

$$\mathsf{Comp}_{\mathcal{C}}(f^{\dagger}, f, s_f), \qquad s_f \sim_{\mathcal{H}} \mathsf{id}_b,$$

and suppose g has a split-epimorphism witness with right-inverse hom-carrier $g^{\dagger} : c \to b$ and split composite $s_g : c \to c$,

$$\mathsf{Comp}_{\mathcal{C}}(g^{\dagger}, g, s_g), \qquad s_g \sim_{\mathcal{H}} \mathsf{id}_c,$$

in the sense of Definition 61.332. If $u : a \to c$ is a displayed composite of g after f and $\ell : c \to a$ is the displayed reverse composite of their right-inverse witnesses,

$$\mathsf{Comp}_{\mathcal{C}}(f, g, u), \qquad \mathsf{Comp}_{\mathcal{C}}(g^{\dagger}, f^{\dagger}, \ell),$$

then u is a displayed hom-carrier from a to c and carries a split-epimorphism witness whose right-inverse hom-carrier is ℓ.

Proof. The two split-epimorphism witnesses supply hom-carriers for $f^\dagger$ and $g^\dagger$. Composition closure in the $\mathsf{Cat}^\uparrow$ stability certificate therefore makes the displayed composites u and ℓ hom-carriers over $a \to c$ and $c \to a$.

Let $R : c \to c$ be the composite display selected for u after ℓ, so $\mathsf{Comp}_{\mathcal{C}}(\ell, u, R)$. It remains to classify R with id_c. The four displayed factors form the chain

$$c \xrightarrow{g^\dagger} b \xrightarrow{f^\dagger} a \xrightarrow{f} b \xrightarrow{g} c.$$

The associativity row of Definition 61.201, applied twice, compares the outside bracketing $(g \circ_{\mathcal{C}} f) \circ_{\mathcal{C}} (f^\dagger \circ_{\mathcal{C}} g^\dagger)$ with the bracketing $g \circ_{\mathcal{C}} ((f \circ_{\mathcal{C}} f^\dagger) \circ_{\mathcal{C}} g^\dagger)$. Thus R is classified with the display obtained by first using the inner split composite s_f, then composing with $g^\dagger$, and finally composing with g.

Since $s_f \sim_{\mathcal{H}} \mathsf{id}_b$, composition congruence replaces $s_f \circ_{\mathcal{C}} g^\dagger$ by $\mathsf{id}_b \circ_{\mathcal{C}} g^\dagger$. The identity law at the target of $g^\dagger$ classifies this composite with $g^\dagger$. Composing that comparison with the outer factor g compares the resulting display with $s_g = g \circ_{\mathcal{C}} g^\dagger$. Transitivity of the morphism classifier, together with the stored comparison $s_g \sim_{\mathcal{H}} \mathsf{id}_c$, gives $R \sim_{\mathcal{H}} \mathsf{id}_c$.

The hom-carriers for u and ℓ, the target identity display id_c, the composite ledger $\mathsf{Comp}_{\mathcal{C}}(\ell, u, R)$, and the classifier comparison $R \sim_{\mathcal{H}} \mathsf{id}_c$ are exactly the data required by Definition 61.332. Hence u carries a split-epimorphism witness with right-inverse hom-carrier ℓ. □

Lean4 ✓ `BEDC.Derived.CategoryUp.CategorySplitEpiWitness composition closed`

Theorem 61.335 (CategoryUp mature host-interface inventory)**.** *The unary-continuation* $\mathsf{Cat}^\uparrow$ *mature host interface is exactly the object carrier* UnaryHistory*, the hom carrier* CategoryHomCarrier*, identity witnesses, the composition-result carrier and semantic certificate, the associativity and unit law package, endpoint transport, opposite-category data identity, split-epimorphism right cancellation, and split-epimorphism composition closure.*

Lean4 ✓ `BEDC.Derived.CategoryUp.CategoryUp mature host interface inventory`

Proof. The standard host bridge is the paper-level interface of Definition 61.329: it reads objects through UnaryHistory, arrows through CategoryHomCarrier, and the composition-result ledger through the semantic certificate target named there.

The law package is the unary-continuation stability package of Definition 61.325. Its identity, composition, associativity, unit, and endpoint-transport rows are precisely the host-interface rows needed by the bridge.

The remaining inventory rows are the category-specific closure theorems already displayed in this chapter. Opposite-category data identity is Definition 61.330; split-epimorphism right cancellation is Definition 61.333; and split-epimorphism composition closure is Definition 61.334. These theorem sites exhaust the stated mature host-interface inventory and introduce no additional host-category primitives. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$**.**

Constructive story: A category here is built from the unary history spine: objects are exactly the histories accepted by UnaryHistory, and a morphism from a to b is a visible unary tail f together with the continuation row $\mathsf{Cont}(a, f, b)$. Two objects, and likewise two displayed morphism histories over fixed endpoints, are classified by hsame, so composition never leaves the BEDC history surface: it appends two visible tails and then uses continuation associativity and transport to compare the two possible ledgers. The source, pattern, and ledger rows are the same concrete hom-carrier and continuation predicates, recording identities as empty tails, composable pairs as displayed continuation chains, and all source-target movement as endpoint hsame transport. The five name-certificate obligations are supplied by the unary empty object and empty identity tail for inhabitation, by reflexivity, symmetry, and transitivity of hsame, and by hom-carrier transport,

which keeps carried objects and arrows carried after classification. A morphism history h lies in the signature gap of a $\mathsf{Cat}^{\uparrow}$ token p when the source-domain accepts the relevant unary endpoints, the signature relation pairs the displayed hom or composite result with an answer history and evidence ledger, and token introduction names that answer as p; unfolded, the gap says that h is the visible tail of the continuation morphism or composite result encoded by the token. The standard bridge reads the same package as an ordinary host category only through unary objects, CategoryHomCarrier arrows, classifier-compatible identities, composition, associativity, and unit rows, then reads back to the same histories. The chapter registers this unary-continuation construction under the name $\mathsf{Cat}^{\uparrow}$; categorical structure is the existing continuation calculus with no global category universe, host equality, limits, adjunctions, or external morphism type.

Theory closure: TheoryGate(NameCert$_{\mathsf{Cat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The unary-continuation $\mathsf{Cat}^{\uparrow}$ host interface is closed for the public unary-continuation export, the paper-level standard bridge of Definition 61.329, the double-opposite certificate data identity and involution sites of Definition 61.330 and Definition 61.331, and the split-epimorphism right-cancellation and composition-closure sites of Definition 61.333 and Definition 61.334.

Formal verification: FormalStatus(NameCert$_{\mathsf{Cat}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CategoryUp.CategoryUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close arbitrary categories, limits, colimits, adjunctions, equivalences, enriched categories, categorical foundations, or host-category primitives beyond the displayed unary-continuation bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

Proposition 61.336 (Category standard bridge dependency lattice)**.** *The* $\mathsf{Cat}^{\uparrow}$ *Hom carrier, composition row, identity row, and checked standard bridge surface are the shared substrate consumed by chapter 62 and chapter 63. Their downstream reads use the category rows as dependency anchors: functor and natural-transformation certificates receive the displayed* CategoryHomCarrier *arrows, empty-tail identities, continuation composition ledger, and the* CategoryHomCarrier_comp_result_semanticNameCer *bridge-facing certificate, rather than a host-category object, host-category equality, or external morphism primitive.*

Proof. The CategoryUp certificate in Definition 61.325 supplies the identity, composition, associativity, unit, and endpoint-transport rows over the unary-continuation Hom carrier. The standard bridge of Definition 61.329 reads the same surface through UnaryHistory, CategoryHomCarrier, and hsame, and the closure block of chapter 61 records this package at maturePackageClosed with bridgeChecked status.

The FunctorUp and NatTransUp chapters cite the CategoryUp chapter by chapter 62 and chapter 63. Their certificate rows therefore attach to the displayed object, Hom, identity, composition, and bridge-facing semantic-certificate rows already exported by CategoryUp. No downstream clause needs a host-category carrier or primitive host equality: the dependency lattice is the BEDC row package itself, and the bridge reads that package without replacing it. □

Chapter 62

A Concrete Naming Certificate for $\mathsf{Functor}^{\uparrow}$

The functor interface $\mathsf{Functor}^{\uparrow}$ has certificate obligations between two categories from chapter 61 as a pair of mappings (one on objects, one on morphisms) preserving identity and composition under the morphism classifier of the codomain. It is not a primitive function space between categories; it is admitted only as a structure-preserving pair of carrier mappings.

62.1 Carrier definition

Definition 62.1 (Functor carrier)**.** *Use a structure with object mapping, morphism mapping, and proofs preserving identity and composition.*

Lean4 (definition) `BEDC.Derived.FunctorUp.PrefixFunctorCarrier`

Theorem 62.2 (Prefix functor carrier from unary prefix)**.** *For the unary-continuation category, every unary prefix history determines a prefix functor carrier. Its hom-carrier fields preserve and reflect prefixed hom histories, and its composition field preserves displayed composition after prefixing the endpoints.*

Proof. Package the unary prefix witness together with the prefix hom-carrier preservation theorem, the prefix reflection theorem, and the prefixed composition preservation theorem. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier from unary prefix`

Theorem 62.3 (Prefix functor carrier hsame transport with unary readback)**.** *For prefix histories p, q, a prefix functor carrier at p transports along $p\mathsf{hsame}q$ to a prefix functor carrier at q, and the transported endpoint also exposes $\mathsf{UnaryHistory}(q)$.*

Proof. Case-analyze the displayed history-sameness $p\mathsf{hsame}q$. In the reflexive case the original prefix functor carrier is already a carrier at q, and its stored prefix-unary field gives the unary readback. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier hsame transport with unary`

Theorem 62.4 (Functor prefix hom-carrier preserves)**.** *For a unary prefix p, any category hom-carrier from a to b transports to a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$ with the same morphism history.*

Proof. Use unary append closure on the two endpoints and place the original continuation ledger under the common left prefix. The prefix is fixed on both source and target, so the morphism history is read without alteration. □

Theorem 62.5 (Functor prefix hom-carrier reflects)**.** *For a unary prefix p, any category hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *reflects to a category hom-carrier from a to b with the same morphism history.*

Proof. Read the prefixed endpoint carriers through unary append exactness and cancel the common left prefix in the continuation ledger. The remaining continuation is the unprefixed hom-carrier from a to b with the same morphism history. □

Theorem 62.6 (Functor prefix hom-carrier identity closed)**.** *For a unary prefix p and a unary object a, the empty morphism is a category hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, a)$*.*

Proof. Unary append closure carries the prefixed object. The category empty-identity row then supplies the empty hom-carrier at that carried endpoint. □

Theorem 62.7 (Functor prefix hom-carrier composition preserves)**.** *For a unary prefix p, composable hom-carriers from a to b and from b to c remain composable after prefixing all endpoints by p, and their continuation composite has the same morphism history.*

Proof. Apply Definition 62.4 to both factor hom-carriers. The displayed continuation ledger is unchanged as a morphism history, while the source, middle, and target endpoints are all read under the same unary prefix. □

Theorem 62.8 (Functor prefix hom-carrier composite hsame transport)**.** *For a unary prefix p, if two composable hom-carrier displays are transported by* hsame*-classified source, middle, target, and morphism data, then the prefixed composite display transports by the same* hsame *data under* $\mathsf{append}(p, \cdot)$*.*

Proof. Transport the two factor hom-carriers by the category endpoint and morphism transport rows, then apply prefix preservation to the transported factors. Append congruence carries the source, middle, and target comparisons through the fixed unary prefix, and continuation transport carries the composite morphism comparison. □

Theorem 62.9 (Prefix functor carrier composition public readback)**.** *For a prefix functor carrier p, if f is a hom-carrier from a to b, g is a hom-carrier from b to c, and* $\mathsf{Cont}(f, g, fg)$*, then fg is displayed as a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, c)$*. Moreover, any displayed hom-carrier between the same prefixed endpoints has morphism history* hsame *to fg.*

Proof. The composition-preservation field of the prefix functor carrier closes the prefixed composite. Public readback follows by category hom-carrier morphism determinacy at the same prefixed endpoints. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier comp public readback`

Theorem 62.10 (Prefix functor carrier right-factor endpoint determinacy)**.** *For a prefix functor carrier p, suppose f is a hom-carrier from a to b, fg is displayed as* $\mathsf{Cont}(f, g, fg)$*, and the prefixed composite fg is a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, c)$*. If the same composite is also displayed as* $\mathsf{Cont}(f, g', fg)$ *and g' is a hom-carrier from b' to c, then*

$$\mathsf{hsame}(b, b') \wedge \mathsf{hsame}(g, g').$$

Proof. The hom-reflection field of the prefix functor carrier reflects the displayed prefixed composite back to a hom-carrier from a to c. Category right-factor endpoint determinacy compares this reflected composite against the displayed $b' \to c$ right factor, and continuation left cancellation identifies the two right-factor histories. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier comp right factor endpoint deterministic`

Theorem 62.11 (Prefix functor carrier hom public readback)**.** *For a prefix functor carrier p, if f is a hom-carrier from a to b, then f is displayed as a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$*. Moreover, any displayed hom-carrier between those same prefixed endpoints has morphism history* hsame *to f.*

Proof. The hom-preservation field of the prefix functor carrier gives the prefixed hom-carrier with the same morphism history. Category hom-carrier morphism determinacy at the same prefixed endpoints compares every displayed carrier with that preserved morphism. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier hom public readback`

Theorem 62.12 (Prefix functor carrier append hom preservation)**.** *For prefix functor carriers p and q, any hom-carrier* $\mathsf{CategoryHomCarrier}(a, b, f)$ *is carried by the appended prefix:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{append}(p, q), a), \mathsf{append}(\mathsf{append}(p, q), b), f).$$

Proof. First use the hom-preservation field of the q-carrier to display the hom-carrier at endpoints prefixed by q. Use the hom-preservation field of the p-carrier on that displayed hom-carrier. The resulting endpoints are $\mathsf{append}(p, \mathsf{append}(q, a))$ and $\mathsf{append}(p, \mathsf{append}(q, b))$; continuation append associativity transports them to the appended-prefix endpoints. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier append hom preserves`

Theorem 62.13 (Prefix functor carrier append composition preservation)**.** *For prefix functor carriers p and q, composable hom-carriers* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and* $\mathsf{CategoryHomCarrier}(b, c, g)$*, together with* $\mathsf{Cont}(f, g, fg)$*, display the composite fg at the endpoints prefixed by* $\mathsf{append}(p, q)$*:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{append}(p, q), a), \mathsf{append}(\mathsf{append}(p, q), c), fg).$$

Proof. Use the composition-preservation field of the q-carrier to close the composite at endpoints prefixed by q. Then use the hom-preservation field of the p-carrier on that already composed display. Append associativity transports the source and target endpoints from nested prefixing to the single appended-prefix form. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier append comp preserves`

Theorem 62.14 (Prefix functor carrier e1 prefix hom reflects)**.** *For histories p, a, b, f, a prefix functor carrier at the visible prefix* $\mathsf{E1}(p)$ *reflects any displayed prefixed hom-carrier back to the unprefixed hom-carrier and exposes the unary tail of the visible prefix:*

$$\mathsf{PrefixFunctorCarrier}(\mathsf{E1}(p)) \to \mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{E1}(p), a), \mathsf{append}(\mathsf{E1}(p), b), f) \to \mathsf{UnaryHistory}(p) \wedge \mathsf{Categ}$$

Proof. Project the stored unary evidence for the visible prefix and invert the $\mathsf{E1}$ constructor to obtain $\mathsf{UnaryHistory}(p)$. The carrier's hom-reflection field cancels the displayed common prefix and returns the unprefixed category hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier e1 prefix hom reflects`

Theorem 62.15 (Prefix functor carrier associative composition public readback)**.** *For a prefix functor carrier p, if f, g, h are composable hom-carriers from a to b, from b to c, and from c to d, with displayed composites fg, gh, $left$, and $right$, then $left$ and $right$ are hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, d)$*, satisfy* $\mathsf{hsame}(left, right)$*, and compare by* hsame *with every displayed hom-carrier between those same prefixed endpoints.*

Proof. The prefix functor carrier exposes a unary witness for the prefix, so prefix associative composition preservation closes both bracketed composites at the prefixed endpoints and supplies their associativity comparison. Category hom-carrier morphism determinacy at the same prefixed endpoints then compares each bracketed composite with any displayed carrier there. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier comp assoc public readback`

Theorem 62.16 (Prefix functor carrier identity public readback)**.** *For a prefix functor carrier p and a unary object a, every displayed hom-carrier from* $\mathsf{append}(p, a)$ *to itself has morphism history* hsame *to* e.

Proof. The prefix carrier supplies unary closure for p. Together with the unary carrier for a, this constructs the empty identity hom-carrier at the prefixed endpoint. Category hom-carrier morphism determinacy then compares any displayed endomorphism at the same endpoint with that empty identity. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier identity public readback`

Theorem 62.17 (Prefix functor carrier preserves split-mono witnesses)**.** *For histories p, a, b, f, g, i, c, suppose p carries a prefix functor carrier, $f : a \to b$ and $g : b \to a$ are category hom-carriers, $i : a \to a$ is an identity hom-carrier,* $\mathsf{Cont}(f, g, c)$ *displays the composite, $c : a \to a$ is also a category hom-carrier, and* $c\mathsf{hsame}i$*. Then the prefixed endpoints carry a split-monomorphism witness: $g :$* $\mathsf{append}(p, b) \to \mathsf{append}(p, a)$*, the identity carrier over* $\mathsf{append}(p, a)$*, the same composite ledger, the prefixed composite carrier, and the same classifier comparison.*

Proof. Use the hom-preservation field of the prefix functor carrier on the inverse hom-carrier g and on the identity hom-carrier i. Use its composition preservation field on f, g, and the displayed continuation for c. Taking the displayed prefixed identity to be i and the displayed prefixed composite to be c gives the required witness package, with the original comparison $c\mathsf{hsame}i$ as the classifier component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier split mono witness preserves`

Theorem 62.18 (Prefix functor carrier preserves split-epi witnesses)**.** *For histories p, a, b, f, g, i, c, suppose p carries a prefix functor carrier, $f : a \to b$ and $g : b \to a$ are category hom-carriers, $i : b \to b$ is an identity hom-carrier,* $\mathsf{Cont}(g, f, c)$ *displays the composite, $c : b \to b$ is also a category hom-carrier, and* $c\mathsf{hsame}i$*. Then the prefixed endpoints carry a split-epimorphism witness: $g :$* $\mathsf{append}(p, b) \to \mathsf{append}(p, a)$*, the identity carrier over* $\mathsf{append}(p, b)$*, the same composite ledger, the prefixed composite carrier, and the same classifier comparison.*

Proof. Use the hom-preservation field of the prefix functor carrier on the displayed reverse hom-carrier g and on the identity hom-carrier i. Use its composition preservation field on g, f, and the displayed continuation for c. Taking the displayed prefixed identity to be i and the displayed prefixed composite to be c gives the required witness package, with the original comparison $c\mathsf{hsame}i$ as the classifier component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier split epi witness preserves`

Theorem 62.19 (Prefix functor carrier preserves split-isomorphism witness pairs)**.** *For histories $p, a, b, f, g_L, i_A, c_A, g_R, i_B, c_B$, suppose p carries a prefix functor carrier, $f : a \to b$ is a category hom-carrier, $g_L : b \to a$ and $g_R : b \to a$ are the two displayed inverse-side hom-carriers, $i_A : a \to a$ and $i_B : b \to b$ are identity hom-carriers, and the continuations $\mathsf{Cont}(f, g_L, c_A)$ and $\mathsf{Cont}(g_R, f, c_B)$ display composites classified by i_A and i_B. Then the prefixed endpoints carry the corresponding split-isomorphism witness pair with the same morphism histories and classifier comparisons.*

Proof. Use the hom-preservation field of the prefix functor carrier on f, both inverse-side hom-carriers, and both identity hom-carriers. Use its composition-preservation field on the two displayed continuation ledgers. The four displayed witnesses are the original identity and composite histories, and the two classifier components are the original comparisons. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier split iso witness pair preserves`

Theorem 62.20 (Prefix functor empty morphism source exactness)**.** *For histories $p, a, source$, the empty history is a category hom-carrier from source to $\mathsf{append}(p, a)$ exactly when p and a are unary and source is history-same to that prefixed target:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{append}(p, a), \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(source, \mathsf{append}(p, a$$

Proof. Apply category empty-identity exactness to the displayed empty morphism. The target unary witness splits through append into unary witnesses for p and a, while the same exactness theorem gives the endpoint sameness. Conversely, append closure rebuilds the target carrier, unary transport gives the source carrier along endpoint sameness, and category empty-identity exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty morphism source iff`

Theorem 62.21 (One-headed prefix empty morphism source exactness)**.** *For histories $p, a, source$, the empty history is a category hom-carrier from source to $\mathsf{append}(\mathsf{E1}(p), a)$ exactly when p and a are unary and source is history-same to that one-headed prefixed target:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{append}(\mathsf{E1}(p), a), \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(source, \mathsf{append}($$

Proof. Apply prefix empty morphism source exactness with prefix $\mathsf{E1}(p)$. The forward direction turns unary membership of $\mathsf{E1}(p)$ into unary membership of p by one-headed unary inversion. Conversely, unary closure of the one-headed constructor rebuilds the prefixed unary target data, and the general source exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 prefix empty morphism source iff`

Theorem 62.22 (Prefix functor empty morphism endpoint exactness)**.** *For histories p, a, b, f, a prefixed hom-carrier has empty morphism history exactly when the unprefixed endpoints agree:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f) \Longrightarrow (\mathsf{hsame}(f, \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(a, b)).$$

Proof. The underlying category hom-carrier identifies empty morphisms with equality of the prefixed endpoints. Left cancellation for generated-history append removes the common prefix. Conversely, endpoint equality of a and b lifts through the shared prefix and the category endpoint theorem identifies the morphism with e. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier morphism empty endpoint iff`

Theorem 62.23 (Functor prefix hom-carrier composite result empty exactness)**.** *For histories p, a, b, c, f, g, fg, if f is a hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$, g is a hom-carrier from $\mathsf{append}(p, b)$ to $\mathsf{append}(p, c)$, and $\mathsf{Cont}(f, g, fg)$ holds, then emptiness of the displayed composite result is equivalent to the two factor morphisms being empty and the adjacent unprefixed endpoints agreeing:*

$$\mathsf{hsame}(fg, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. The forward direction transports the displayed continuation to empty result and applies prefix empty-composite exactness. Conversely, the componentwise empty data builds the empty composite continuation; continuation determinacy compares that empty result with the displayed composite result fg. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp result empty iff`

Theorem 62.24 (Prefix hom composite empty-middle inversion)**.** *For histories p, a, c, f, g, fg, if f is a hom-carrier from $\mathsf{append}(p, a)$ to e, g is a hom-carrier from e to $\mathsf{append}(p, c)$, and $\mathsf{Cont}(f, g, fg)$, then*

$$\mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{append}(p, c)) \wedge \mathsf{CategoryHomCarrier}(\mathsf{e}, \mathsf{append}(p, c), fg) \wedge \mathsf{hsame}(fg, \mathsf{a}$$

Proof. Empty-target component exactness splits the left hom-carrier into emptiness of the prefix, source tail, and morphism. Empty-source component exactness reads the right morphism as the prefixed target. With $f = \mathsf{e}$, the displayed continuation identifies fg with g, so fg also reads back as $\mathsf{append}(p, c)$; category empty-source exactness rebuilds the final hom-carrier for fg. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp empty middle inversion`

Theorem 62.25 (Prefix hom composite empty-middle target determinacy)**.** *For histories $p, a, c, f, g, fg, target$, if f is a hom-carrier from $\mathsf{append}(p, a)$ to e, g is a hom-carrier from e to $\mathsf{append}(p, c)$, $\mathsf{Cont}(f, g, fg)$, and the displayed composite fg is also a hom-carrier from e to target, then*

$$\mathsf{hsame}(target, \mathsf{append}(p, c)).$$

Proof. The empty-middle inversion supplies the canonical hom-carrier from e to $\mathsf{append}(p, c)$ over the same composite morphism fg. Target determinacy for category hom-carriers compares this canonical carrier with the displayed one and yields the stated classifier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp empty middle target deterministic`

Theorem 62.26 (Prefix hom composite empty-middle displayed morphism determinacy)**.** *For histories $p, a, c, f, g, fg, displayed$, if f is a hom-carrier from $\mathsf{append}(p, a)$ to e, g is a hom-carrier from e to $\mathsf{append}(p, c)$, $\mathsf{Cont}(f, g, fg)$, and displayed is also a hom-carrier from e to $\mathsf{append}(p, c)$, then*

$$\mathsf{hsame}(displayed, fg).$$

Proof. The prefixed empty-middle inversion supplies the canonical hom-carrier from e to $\mathsf{append}(p, c)$ over fg. Category hom-carrier morphism determinacy compares that carrier with the displayed carrier over the same endpoints. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp empty middle displayed morphism deterministic`

Theorem 62.27 (Prefix hom composite empty-middle source determinacy)**.** *For histories p, a, c, f, g, fg, source, if f is a hom-carrier from* $\mathsf{append}(p, a)$ *to* e*, g is a hom-carrier from* e *to* $\mathsf{append}(p, c)$*,* $\mathsf{Cont}(f, g, fg)$*, and the displayed composite fg is also a hom-carrier from source to* $\mathsf{append}(p, c)$*, then*

$$\mathsf{hsame}(\mathit{source}, \mathsf{e}).$$

Proof. The empty-middle inversion supplies the canonical hom-carrier from e to $\mathsf{append}(p, c)$ over fg. Source determinacy for category hom-carriers compares the displayed carrier with this canonical carrier and identifies the displayed source with e. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp empty middle source deterministic`

Theorem 62.28 (Prefix functor carrier empty-middle empty-result readback)**.** *For histories p, q, a, c, f, g, fg, suppose p carries a prefix functor carrier, f is a hom-carrier from* $\mathsf{append}(p, a)$ *to* e*, g is a hom-carrier from* e *to* $\mathsf{append}(q, c)$*,* $\mathsf{Cont}(f, g, fg)$*, and $fg\mathsf{hsamee}$. Then q carries a prefix functor carrier, and all displayed components are empty:*

$$\mathsf{PrefixFunctorCarrier}(q) \wedge \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(q, \mathsf{e}) \wedge \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(c, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}).$$

Proof. Apply category composite-result empty exactness to the displayed hom-carriers. The left endpoint equality splits $\mathsf{append}(p, a)\mathsf{hsamee}$ into $p\mathsf{hsamee}$ and $a\mathsf{hsamee}$, and the right endpoint equality splits $\mathsf{append}(q, c)\mathsf{hsamee}$ into $q\mathsf{hsamee}$ and $c\mathsf{hsamee}$. The factor morphism equalities give $f\mathsf{hsamee}$ and $g\mathsf{hsamee}$. Since q is history-same to e, the unary-prefix constructor supplies the prefix functor carrier at q. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier comp empty middle empty result readback`

Theorem 62.29 (Prefix functor preserves associative hom composition)**.** *For the unary-continuation category, if p is unary and f, g, h are composable hom-carriers with composites fg, gh, $left$, and $right$, then $left$ and $right$ are hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, d)$*, and* $\mathsf{hsame}(left, right)$*.*

Proof. Use category-level associative composition closure to obtain the two unprefixed composite hom-carriers and their classifier sameness. Prefix preservation transports each composite hom-carrier across the unary prefix p, while the classifier sameness is the same continuation associativity witness. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp assoc preserves`

Theorem 62.30 (Prefix functor displayed associative hom determinacy)**.** *For the unary-continuation category, if p is unary and f, g, h are composable hom-carriers with composites fg, gh, $left$, and $right$, then any displayed hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, d)$ *determines both bracketed composite histories up to* hsame*.*

Proof. Prefix associative composition preservation supplies the two prefixed hom-carriers over the displayed endpoints. Category hom-carrier morphism determinacy compares each bracketed composite carrier with the displayed carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp assoc displayed deterministic`

Theorem 62.31 (Prefix functor associative empty result components)**.** *For the unary-continuation category, if p is unary and the unprefixed hom-carriers f, g, h compose with associated results $left$ and $right$, then a displayed empty hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, d)$ *forces both associated composite histories to be* hsame *to* e*, and forces* $a\mathsf{hsame}d$*.*

Proof. Use prefix displayed associative determinacy with the displayed empty hom-carrier to compare both bracketed composites with e. Prefix empty identity exactness cancels the common prefix in the same displayed empty hom-carrier and reads back $a\mathsf{hsame}d$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp assoc empty result components`

Theorem 62.32 (Prefix functor associative nonempty target component cases)**.** *For the unary-continuation category, suppose p is unary and f, g, h are composable hom-carriers with associated composites fg, gh, $left$, and $right$. If the left-associated composite result is* hsame*-equal to* $\mathsf{E1}(k)$*, then either the prefix p is itself one-headed, or the final target component d is one-headed.*

Proof. Close the first two hom-carriers to obtain the fg hom-carrier from a to c, then prefix it and the third hom-carrier by p. Transport the displayed continuation $\mathsf{Cont}(fg, h, left)$ along the endpoint equality $left\mathsf{hsameE1}(k)$. The binary prefixed nonempty-target component theorem then applies to the prefixed fg-then-h composite and separates the visible target endpoint into the prefix component or the final target component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp assoc nonempty result target component e1 cases`

Theorem 62.33 (Prefix functor unary suffix preserves associative hom composition)**.** *For the unary-continuation category, if p and q are unary and f, g, h are composable hom-carriers with composites fg, gh, $left$, and $right$, then $left$ and $right$ are displayed from* $\mathsf{append}(\mathsf{append}(p, a), q)$ *to* $\mathsf{append}(\mathsf{append}(p, d), q)$*, and* $\mathsf{hsame}(left, right)$*.*

Proof. Category-level associative composition closure supplies the two central carriers and the associativity witness between their morphism histories. Unary context exactness transports both carriers through the shared prefix and suffix, while the associativity witness is unchanged. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier unary suffix comp assoc preserves`

Theorem 62.34 (Prefix functor unary suffix reflects associative hom composition)**.** *For histories $p, q, a, b, c, d, f, g, h, fg, gh, left, right$, suppose the three displayed hom-carriers run through the shared unary context* $\mathsf{append}(\mathsf{append}(p, -), q)$ *from a to b, from b to c, and from c to d. If the four continuation ledgers display the two associative composites, then $left$ and $right$ are central hom-carriers from a to d, and* $\mathsf{hsame}(left, right)$*.*

Proof. Apply unary-context exactness to each of the three displayed hom-carriers. This reflects them to central hom-carriers $a \to b$, $b \to c$, and $c \to d$. Category associative composition closure then applies to the reflected carriers and the four displayed continuation ledgers, producing both central composite carriers and the associativity classifier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier unary suffix comp assoc reflects`

Theorem 62.35 (Prefix functor reflects associative hom composition)**.** *For histories $p, a, b, c, d, f, g, h, fg, gh, left,$ suppose f is a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$*, g is a hom-carrier from* $\mathsf{append}(p, b)$ *to* $\mathsf{append}(p, c)$*, and h is a hom-carrier from* $\mathsf{append}(p, c)$ *to* $\mathsf{append}(p, d)$*. If* $\mathsf{Cont}(f, g, fg)$*,* $\mathsf{Cont}(g, h, gh)$*,* $\mathsf{Cont}(fg, h, left)$*, and* $\mathsf{Cont}(f, gh, right)$*, then $left$ and $right$ are hom-carriers from a to d, and* $\mathsf{hsame}(left, right)$*.*

Proof. Reflect the three prefixed hom-carrier witnesses through the unary-context exactness theorem Definition 61.51. This gives unprefixed hom-carriers for f from a to b, g from b to c, and h from c to d.

Apply Theorem 61.132 to these reflected witnesses and to the four displayed continuation relations. The category closure theorem returns both composite carriers from a to d together with the classifier witness $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp assoc reflects`

Theorem 62.36 (Prefix functor preserves empty identity hom carriers)**.** *For the unary-continuation category, if p and a are unary, then the empty history is a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, a)$*.*

Proof. First use the empty identity hom-carrier theorem at a. Prefix preservation then transports that hom-carrier along the unary prefix p. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty identity preserves`

Theorem 62.37 (Prefix functor empty identity hom carrier exactness)**.** *For histories p, a, b, the empty history is a hom-carrier from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *exactly when the two prefixed endpoints are unary and the unprefixed tails a, b are internally the same:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), \mathsf{e}) \iff \mathsf{UnaryHistory}(\mathsf{append}(p, a)) \wedge \mathsf{UnaryHistory}(\mathsf{append}(p, b)) \wedge$$

Proof. Apply empty identity hom-carrier exactness in the category chapter to obtain sameness of the two prefixed endpoints. Left cancellation for generated-history append then reflects this sameness to the tails a, b. Conversely, tail sameness lifts to sameness of the prefixed endpoints, and category empty identity exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty identity iff`

Theorem 62.38 (Prefix functor empty morphism target exactness)**.** *For histories $p, a, target$, the empty history is a hom-carrier from the prefixed source* $\mathsf{append}(p, a)$ *to target exactly when p and a are unary and target is history-same to that prefixed source:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), target, \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(target, \mathsf{append}(p, a))$$

Proof. Apply category empty-identity exactness to the hom-carrier with source $\mathsf{append}(p, a)$. The source unary witness splits into unary witnesses for p and a, and endpoint sameness is read in the target-to-source direction. Conversely, append closure rebuilds the unary source; transport along endpoint sameness gives the unary target, and category empty-identity exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty morphism target iff`

Theorem 62.39 (One-headed prefix empty morphism target exactness)**.** *For histories $p, a, target$:* BHist*, the empty history is a hom-carrier from the one-headed prefixed source* $\mathsf{append}(\mathsf{E1}(p), a)$ *to target exactly when p and a are unary and target is history-same to that source:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{E1}(p), a), target, \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{hsame}(target, \mathsf{append}($$

Proof. Apply prefix empty morphism target exactness with prefix $\mathsf{E1}(p)$. The forward direction turns unary membership of $\mathsf{E1}(p)$ into unary membership of p by one-headed unary inversion. Conversely, unary closure of the one-headed constructor rebuilds the prefixed unary source data, and the general empty morphism target exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 prefix empty morphism target iff`

Theorem 62.40 (One-headed prefix empty identity hom carrier exactness)**.** *For histories p, a, b, the empty history is a hom-carrier from $\mathsf{append}(\mathsf{E1}(p), a)$ to $\mathsf{append}(\mathsf{E1}(p), b)$ exactly when p, a, and b are unary and the two tails are internally the same:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{E1}(p), a), \mathsf{append}(\mathsf{E1}(p), b), \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}$$

Proof. The forward direction applies prefix empty identity exactness with prefix $\mathsf{E1}(p)$, then splits the unary append witnesses for the source and target. One-headed unary inversion extracts unary membership of p. The reverse direction closes $\mathsf{E1}(p)$ under unary generation, rebuilds the two prefixed unary endpoints by append closure, and invokes prefix empty identity exactness. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 prefix empty identity iff`

Theorem 62.41 (Prefix functor empty composite exactness)**.** *For histories p, a, b, c, f, g, if f is a hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$ and g is a hom-carrier from $\mathsf{append}(p, b)$ to $\mathsf{append}(p, c)$, then their empty composite is equivalent to both morphism histories being empty and the two unprefixed adjacent endpoints agreeing:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Category empty-composite exactness gives the two empty morphism histories and the two prefixed endpoint equalities. Left cancellation removes the common prefix p from both endpoint equalities. Conversely, the unprefixed endpoint equalities rebuild the prefixed endpoint equalities, and category empty-composite exactness supplies the empty continuation ledger. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp empty iff`

Theorem 62.42 (Prefix functor one-headed endpoint empty composite exactness)**.** *For histories p, a, b, c, f, g, if f is a hom-carrier from $\mathsf{append}(p, \mathsf{E1}(a))$ to b and g is a hom-carrier from b to $\mathsf{append}(p, \mathsf{E1}(c))$, then their empty composite is equivalent to both morphism histories being empty and the one-headed endpoint payloads agreeing:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \iff \mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(a, c).$$

Proof. Category empty-composite exactness gives the two empty morphism histories and the adjacent endpoint equalities through b. Transitivity identifies the two prefixed one-headed endpoints. Left cancellation removes the shared prefix p, and one-headed constructor inversion gives $\mathsf{hsame}(a, c)$. Conversely, the empty morphism histories reduce the displayed continuation ledger to $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 endpoint empty composite iff`

Theorem 62.43 (Prefix functor one-headed empty composite middle endpoint)**.** *For histories p, a, b, c, f, g, if f is a hom-carrier from $\mathsf{append}(p, \mathsf{E1}(a))$ to b, g is a hom-carrier from b to $\mathsf{append}(p, \mathsf{E1}(c))$, and their composite is empty, then the shared middle endpoint is classified with both visible prefixed endpoints:*

$$\mathsf{Cont}(f, g, \mathsf{e}) \Longrightarrow \mathsf{hsame}(b, \mathsf{append}(p, \mathsf{E1}(a))) \wedge \mathsf{hsame}(b, \mathsf{append}(p, \mathsf{E1}(c))).$$

Proof. The one-headed empty-composite exactness theorem gives $\mathsf{hsame}(f, \mathsf{e})$ and $\mathsf{hsame}(g, \mathsf{e})$. Transport the two hom-carriers along these morphism classifiers so that both displayed morphisms are empty. Category empty-identity exactness then reads the left leg as equality of its source and middle endpoint, and the right leg as equality of the middle endpoint and its target. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 endpoint empty composite middle hsame`

Theorem 62.44 (Prefix functor empty-target component exactness)**.** *For histories p, a, f, a hom-carrier from the prefixed source* $\mathsf{append}(p, a)$ *into the empty target is exactly componentwise emptiness of the prefix, the tail, and the morphism:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{e}, f) \iff \mathsf{hsame}(p, \mathsf{e}) \wedge \mathsf{hsame}(a, \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e}).$$

Proof. Apply category empty-target exactness to obtain $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{e})$ and $\mathsf{hsame}(f, \mathsf{e})$. Empty splitting for the recursive append representative separates the first witness into $p = \mathsf{e}$ and $a = \mathsf{e}$. Conversely, eliminating the three sameness witnesses reduces the hom-carrier to the empty source, empty target, empty morphism case. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty target components iff`

Theorem 62.45 (Prefix functor one-headed empty-target impossibility)**.** *For histories p, a, f, no hom-carrier can run from* $\mathsf{append}(\mathsf{E1}(p), a)$ *into the empty target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(\mathsf{E1}(p), a), \mathsf{e}, f) \to \mathsf{False}.$$

Proof. Apply the empty-target component exactness theorem just above with prefix $\mathsf{E1}(p)$. It reads back $\mathsf{hsame}(\mathsf{E1}(p), \mathsf{e})$, contradicting the closed constructor separation for one-headed histories. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 prefix empty target absurd`

Theorem 62.46 (Prefix functor empty-target source morphism swap)**.** *For histories p, a, f, a hom-carrier from the prefixed source* $\mathsf{append}(p, a)$ *into the empty target is unchanged when the displayed source and morphism fields are exchanged:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{e}, f) \iff \mathsf{CategoryHomCarrier}(f, \mathsf{e}, \mathsf{append}(p, a)).$$

Proof. Apply category empty-target exactness at the prefixed source. The hom-carrier is exactly the pair $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{e}) \wedge \mathsf{hsame}(f, \mathsf{e})$. Swapping those two boundary classifiers gives the exchanged empty-target hom-carrier, and the reverse implication swaps them back. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty target source morphism swap iff`

Theorem 62.47 (Prefix functor empty-source component exactness)**.** *For histories p, b, f, a hom-carrier from the empty source into the prefixed target* $\mathsf{append}(p, b)$ *is exactly unary component data for p and b together with morphism equality to that prefixed target:*

$$\mathsf{CategoryHomCarrier}(\mathsf{e}, \mathsf{append}(p, b), f) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(f, \mathsf{append}(p, b)).$$

Proof. Category empty-source exactness turns the hom-carrier into unary closure of the prefixed target and equality of f with that target. Unary append factorization separates the target carrier into the prefix and tail carriers. Conversely, unary append closure rebuilds the target carrier, and category empty-source exactness reconstructs the hom-carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier empty source components iff`

Theorem 62.48 (Prefix functor visible composite target component cases)**.** *For prefixed hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *and from* $\mathsf{append}(p, b)$ *to* $\mathsf{append}(p, c)$*, if the composite continuation displays the visible result* $\mathsf{E1}(k)$*, then either the prefix component or the target-tail component is itself one-headed:*

$$\exists p_0.\ p = \mathsf{E1}(p_0) \wedge \mathsf{UnaryHistory}(p_0) \quad \vee \quad \exists c_0.\ c = \mathsf{E1}(c_0) \wedge \mathsf{UnaryHistory}(c_0).$$

Proof. The nonempty composite target visibility theorem displays $\mathsf{append}(p, c)$ as $\mathsf{E1}(r)$. Hence this append target is not history-same to e. BEDC append non-emptiness splits this into nonempty p or nonempty c, and unary append factorization supplies the corresponding unary carrier. Unary nonempty decomposition then reads back the visible one-headed component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty result target component e1 cases`

Theorem 62.49 (Prefix functor visible composite source component cases)**.** *For prefixed hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *and from* $\mathsf{append}(p, b)$ *to* $\mathsf{append}(p, c)$*, if the composite continuation displays the visible result* $\mathsf{E1}(k)$ *and the source* $\mathsf{append}(p, a)$ *is not empty, then the prefix component or the source-tail component is one-headed:*

$$\exists p_0.\ p = \mathsf{E1}(p_0) \wedge \mathsf{UnaryHistory}(p_0) \quad \vee \quad \exists a_0.\ a = \mathsf{E1}(a_0) \wedge \mathsf{UnaryHistory}(a_0).$$

Proof. Composite source visibility displays $\mathsf{append}(p, a)$ as a one-headed history. Therefore the append source is nonempty. BEDC append non-emptiness splits this into nonempty p or nonempty a, while unary append factorization extracts the corresponding carrier. Unary nonempty readback then gives the one-headed component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty result source component e1 cases`

Theorem 62.50 (Prefix functor one-headed morphism-target component exactness)**.** *For histories* p, a, k, r*, a hom-carrier from the prefixed source* $\mathsf{append}(p, a)$ *to the one-headed target* $\mathsf{E1}(r)$ *with one-headed morphism* $\mathsf{E1}(k)$ *is exactly unary component data for* p, a, k *together with the tail continuation:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{E1}(r), \mathsf{E1}(k)) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}($$

Proof. Apply category one-headed morphism-target exactness to obtain unary membership of the appended source, unary membership of k, and the tail continuation. Unary append factorization splits the source carrier into the two component witnesses for p and a. Conversely, unary append closure rebuilds the source carrier, and the category exactness theorem lifts the tail continuation through the visible one constructor on both target and morphism. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 morphism target components iff`

Theorem 62.51 (Prefix functor visible endpoint morphism exactness)**.** *For histories* p, a, r, k*, a prefixed visible morphism between one-headed endpoints is equivalent to a unary prefix, unary visible tails, and the unprefixed tail continuation:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p, \mathsf{E1}(r)), \mathsf{E1}(k)) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}$$

Proof. Forward, reflect the prefixed hom-carrier to the base category and apply the one-headed endpoint visible morphism exactness theorem; the prefix carrier is read from the prefixed source. Reverse, first rebuild the base visible hom-carrier from the tail continuation and then apply prefix preservation. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 endpoint visible morphism iff`

Theorem 62.52 (Prefix functor visible morphism strict prefix)**.** *For histories p, a, r, k, a prefixed visible morphism from* $\mathsf{append}(p, \mathsf{E1}(a))$ *to* $\mathsf{append}(p, \mathsf{E1}(r))$ *with morphism* $\mathsf{E1}(k)$*, together with nonempty k, exposes a strict unary prefix from* $\mathsf{E1}(a)$ *to r:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p, \mathsf{E1}(r)), \mathsf{E1}(k)) \Longrightarrow k \neq \mathsf{e} \Longrightarrow \mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(a), r).$$

Proof. Apply prefixed visible endpoint morphism exactness. Its forward direction recovers unary membership of k and the continuation $\mathsf{Cont}(\mathsf{E1}(a), k, r)$. The nonempty witness for k is therefore exactly the strict-prefix tail. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 visible morphism strict prefix`

Theorem 62.53 (Prefix functor visible morphism endpoint contradiction)**.** *For histories p, a, r, k, a prefixed visible morphism with nonempty visible tail cannot have its unprefixed source endpoint* hsame *to the target endpoint:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p, \mathsf{E1}(r)), \mathsf{E1}(k)) \Longrightarrow k \neq \mathsf{e} \Longrightarrow \mathsf{hsame}(\mathsf{E1}(a), r) \Longrightarrow \bot.$$

Proof. The strict-prefix readback supplies a nonempty unary tail from $\mathsf{E1}(a)$ to r. The endpoint hsame comparison would force this tail to be empty by strict-prefix endpoint cancellation, contradicting the displayed nonemptiness. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 visible morphism endpoint hsame absurd`

Theorem 62.54 (Prefix functor visible morphism unique strict boundary)**.** *For histories p, a, r, k, k', two prefixed visible morphisms between the same one-headed endpoints have the same visible payload, and any nonempty first payload exposes the strict endpoint boundary:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p, \mathsf{E1}(r)), \mathsf{E1}(k)) \Longrightarrow \mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p$$

Proof. Apply category morphism determinacy to the two displayed hom-carriers. This gives $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(k'))$, and one-headed no-confusion descends it to $\mathsf{hsame}(k, k')$.

Apply the strict-prefix readback theorem to the first displayed carrier and the nonempty witness for k. This supplies $\mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(a), r)$.

Finally, apply the endpoint contradiction theorem to the same carrier and nonempty witness. It gives the implication $\mathsf{hsame}(\mathsf{E1}(a), r) \to \bot$, and the three facts assemble the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 visible morphism unique strict boundary`

Theorem 62.55 (Prefix functor e-one visible morphism payload determinacy)**.** *For histories $p, p', a, a', r, r', k, k'$, if the three displayed endpoints agree,*

$$\mathsf{hsame}(p, p'), \qquad \mathsf{hsame}(a, a'), \qquad \mathsf{hsame}(r, r'),$$

then two prefixed one-headed visible morphisms with those transported endpoints have the same visible payload:

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, \mathsf{E1}(a)), \mathsf{append}(p, \mathsf{E1}(r)), \mathsf{E1}(k)) \Longrightarrow \mathsf{CategoryHomCarrier}(\mathsf{append}(p', \mathsf{E1}(a')), \mathsf{append}($$

Proof. Use the three hsame rows to identify the prefix, source tail, and target tail displays. Category morphism determinacy then gives $\mathsf{hsame}(\mathsf{E1}(k), \mathsf{E1}(k'))$, and the e-one constructor inversion descends this to $\mathsf{hsame}(k, k')$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier e1 visible morphism payload deterministic`

Theorem 62.56 (Functor prefix hom-carrier zero-headed component impossibility)**.** *For histories p, a, b, f, a prefixed category hom-carrier has no zero-headed prefix, source tail, target tail, or morphism component:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f) \Longrightarrow (\exists z,\ p = \mathsf{E0}(z)) \vee (\exists z,\ a = \mathsf{E0}(z)) \vee (\exists z,\ b = \mathsf{E0}(z)) \vee (\exists z,\ f =$$

Proof. The source and target carrier fields make $\mathsf{append}(p, a)$ and $\mathsf{append}(p, b)$ unary. Unary append factorization exposes unary witnesses for p, a, and b, while the hom-carrier morphism field gives unary history for f. Each zero-headed witness contradicts the unary zero-extension separation theorem. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier zero headed component absurd`

Theorem 62.57 (Functor prefix hom-carrier source prefix determinism)**.** *For histories $p, q, a, target, f$, if the same morphism f is a category hom-carrier from $\mathsf{append}(p, a)$ to target and from $\mathsf{append}(q, a)$ to target, then the two source prefixes agree:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), target, f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(q, a), target, f) \Longrightarrow \mathsf{hsame}(p, q).$$

Proof. Category hom-carrier source determinism gives $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{append}(q, a))$. Cancelling the common right append suffix a yields $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier source prefix deterministic`

Theorem 62.58 (Functor prefix hom-carrier composite source-prefix determinacy)**.** *For histories p, q, a, b, c, f, g, fg, suppose f and g are composable hom-carriers over the common prefix p:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g) \wedge \mathsf{Cont}(f, g, fg)$$

If the displayed composite also has source $\mathsf{append}(q, a)$ and target $\mathsf{append}(p, c)$, then $\mathsf{hsame}(p, q)$.

Proof. Close the two displayed p-prefixed hom-carriers under composition to obtain a canonical carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, c)$ with morphism fg. Source determinacy compares this carrier with the displayed carrier whose source is $\mathsf{append}(q, a)$. Cancelling the common suffix a gives $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp source prefix deterministic`

Theorem 62.59 (Functor prefix hom-carrier composite source-object determinacy)**.** *For histories p, a, d, b, c, f, g, fg, suppose f and g are composable hom-carriers over the common prefix p:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g) \wedge \mathsf{Cont}(f, g, fg)$$

If the displayed composite also has source $\mathsf{append}(p, d)$ and target $\mathsf{append}(p, c)$, then $\mathsf{hsame}(a, d)$.

Proof. Close the expected composite from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, c)$. Source determinacy compares it with the displayed carrier from $\mathsf{append}(p, d)$ to the same target and morphism. Cancelling the common left prefix p gives $\mathsf{hsame}(a, d)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp source object deterministic`

Theorem 62.60 (Functor prefix hom-carrier source-object determinacy)**.** *For histories $p, a, b, target, f$, if the same morphism f is a category hom-carrier from $\mathsf{append}(p, a)$ to target and from $\mathsf{append}(p, b)$ to target, then the two source objects agree:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), target, f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), target, f) \Longrightarrow \mathsf{hsame}(a, b).$$

Proof. Category hom-carrier source determinism gives $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{append}(p, b))$. Cancelling the common left append prefix p yields $\mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier source object deterministic`

Theorem 62.61 (Functor prefix hom-carrier target-prefix determinacy)**.** *For histories $p, q, a, source, f$, if the same morphism f is a category hom-carrier from source to $\mathsf{append}(p, a)$ and from source to $\mathsf{append}(q, a)$, then the two target prefixes agree:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{append}(p, a), f) \wedge \mathsf{CategoryHomCarrier}(source, \mathsf{append}(q, a), f) \Longrightarrow \mathsf{hsame}(p, q).$$

Proof. Category hom-carrier target determinism gives $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{append}(q, a))$. Cancelling the common right append suffix a yields $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier target prefix deterministic`

Theorem 62.62 (Functor prefix hom-carrier target-object determinacy)**.** *For histories $p, a, b, source, f$, if the same morphism f is a category hom-carrier from source to $\mathsf{append}(p, a)$ and from source to $\mathsf{append}(p, b)$, then the two target objects agree:*

$$\mathsf{CategoryHomCarrier}(source, \mathsf{append}(p, a), f) \wedge \mathsf{CategoryHomCarrier}(source, \mathsf{append}(p, b), f) \Longrightarrow \mathsf{hsame}(a, b).$$

Proof. Category hom-carrier target determinism gives $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{append}(p, b))$. Cancelling the common left append prefix p yields $\mathsf{hsame}(a, b)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier target object deterministic`

Theorem 62.63 (Functor prefix hom-carrier composite middle-object determinacy)**.** *For histories p, a, b, b', c, f, g, fg, suppose f is a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$, g is a category hom-carrier from $\mathsf{append}(p, b')$ to $\mathsf{append}(p, c)$, and $\mathsf{Cont}(f, g, fg)$ holds. If fg is displayed as a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, c)$, then $\mathsf{hsame}(b, b')$.*

Proof. Apply category hom-carrier composite middle-object determinacy to the prefixed hom-carriers and the displayed composite. This identifies $\mathsf{append}(p, b)$ and $\mathsf{append}(p, b')$. Cancelling the common left append prefix p gives $\mathsf{hsame}(b, b')$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp middle object deterministic`

Theorem 62.64 (Functor prefix hom-carrier composite target-prefix determinacy)**.** *For histories p, q, a, b, c, f, g, fg, if prefixed hom-carriers compose from $\mathsf{append}(p, a)$ through $\mathsf{append}(p, b)$ to $\mathsf{append}(p, c)$, and the same composite is displayed with target $\mathsf{append}(q, c)$, then the two target prefixes agree:*

$$\mathsf{hsame}(p, q).$$

Proof. Apply category composite target determinacy to compare the expected composite target $\mathsf{append}(p, c)$ with the displayed target $\mathsf{append}(q, c)$. Cancelling the common right suffix c gives $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp target prefix deterministic`

Theorem 62.65 (Functor prefix hom-carrier composite target-object determinacy)**.** *For histories p, a, b, c, d, f, g, fg, if prefixed hom-carriers compose from $\mathsf{append}(p, a)$ through $\mathsf{append}(p, b)$ to $\mathsf{append}(p, c)$, and the same composite is displayed with target $\mathsf{append}(p, d)$, then the two target objects agree:*

$$\mathsf{hsame}(c, d).$$

Proof. Close the expected composite target $\mathsf{append}(p, c)$ from the two prefixed hom-carriers and the continuation witness. Target determinacy compares that carrier with the displayed target $\mathsf{append}(p, d)$, and cancellation of the common left prefix p gives $\mathsf{hsame}(c, d)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp target object deterministic`

Theorem 62.66 (Functor prefix triangle cycle empty components)**.** *For histories p, q, r, a, f, g, h, if prefixed hom-carriers form a triangle*

$$\mathsf{append}(p, a) \xrightarrow{f} \mathsf{append}(q, a) \xrightarrow{g} \mathsf{append}(r, a) \xrightarrow{h} \mathsf{append}(p, a),$$

then all three displayed hom-carriers also hold with empty morphism, and the three adjacent prefixes collapse as $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$.

Proof. Apply category hom-carrier triangle-cycle empty exactness to the three prefixed hom-carriers. The category theorem gives empty morphisms on all three edges and identifies the prefixed adjacent endpoints. Cancelling the common right suffix a from those two endpoint equalities gives the stated prefix equalities. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier prefix triangle cycle empty components`

Theorem 62.67 (Functor prefix composite endpoint cycle boundary components)**.** *For histories p, a, b, c, f, g, fg, suppose the prefixed hom-carriers*

$$\mathsf{append}(p, a) \xrightarrow{f} \mathsf{append}(p, b) \xrightarrow{g} \mathsf{append}(p, c)$$

compose through $\mathsf{Cont}(f, g, fg)$ and the unprefixed endpoints satisfy $\mathsf{hsame}(a, c)$. Then both component morphisms and their composite are hsame to e, and the unprefixed boundary collapses:

$$\mathsf{hsame}(f, \mathsf{e}) \wedge \mathsf{hsame}(g, \mathsf{e}) \wedge \mathsf{hsame}(fg, \mathsf{e}) \wedge \mathsf{hsame}(a, b) \wedge \mathsf{hsame}(b, c).$$

Proof. Apply the category hom-carrier composite endpoint-cycle boundary theorem to the prefixed endpoints. The hypothesis $\mathsf{hsame}(a, c)$ gives $\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{append}(p, c))$, and the category theorem returns the empty morphism data together with prefixed endpoint equalities. Cancelling the common left prefix p gives $\mathsf{hsame}(a, b)$ and $\mathsf{hsame}(b, c)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp endpoint cycle boundary components`

Theorem 62.68 (Functor prefix hom-carrier composite middle-prefix determinacy)**.** *For histories p, q, a, b, c, f, g, fg, suppose f is a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$, g is a category hom-carrier from $\mathsf{append}(q, b)$ to $\mathsf{append}(p, c)$, and $\mathsf{Cont}(f, g, fg)$ holds. If fg is displayed as a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, c)$, then $\mathsf{hsame}(p, q)$.*

Proof. Category hom-carrier composite middle-object determinacy identifies $\mathsf{append}(p, b)$ with $\mathsf{append}(q, b)$. Cancelling the common right suffix b gives $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp middle prefix deterministic`

Theorem 62.69 (Functor prefix hom-carrier composition family public readback)**.** *For histories $a, b, c, f, g, fg, displayed$, suppose every unary prefix p carries f as a hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$, every unary prefix carries g from $\mathsf{append}(p, b)$ to $\mathsf{append}(p, c)$, $\mathsf{Cont}(f, g, fg)$ holds, and every unary prefix carries displayed from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, c)$. Then*

$$\mathsf{hsame}(fg, displayed).$$

Proof. Evaluate the three prefix-indexed carrier families at the empty unary prefix. The two factor carriers close under category composition, and morphism determinacy compares the resulting composite carrier with the displayed carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp family public readback`

Theorem 62.70 (Functor prefix hom-carrier composite zero-headed component impossibility)**.** *For histories p, a, b, c, f, g, fg, two prefixed hom-carriers composing over the middle object b cannot have any zero-headed prefix, object, factor morphism, or composite morphism component:*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g) \wedge \mathsf{Cont}(f, g, fg)$$

whenever one of p, a, b, c, f, g, fg is displayed as $\mathsf{E0}(w)$.

Proof. Close the two hom-carriers under composition to obtain the carrier for fg. The zero-headed prefix, source, middle, and left morphism cases reduce to the left factor exclusion; the target and right morphism cases reduce to the right factor exclusion; the composite morphism case reduces to the closed composite carrier. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp zero headed component absurd`

Theorem 62.71 (Functor prefix composite nonempty target component split)**.** *For prefixed hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f)$$

and

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g),$$

if $\mathsf{Cont}(f, g, fg)$ and fg is not empty, then the displayed target $\mathsf{append}(p, c)$ has a nonempty component:

$$(\mathsf{hsame}(p, \mathsf{e}) \to \bot) \vee (\mathsf{hsame}(c, \mathsf{e}) \to \bot).$$

Proof. The category composite nonempty-target obstruction applies to the two prefixed hom-carriers and their continuation composite, giving $\mathsf{hsame}(\mathsf{append}(p, c), \mathsf{e}) \to \bot$. BEDC append nonemptiness then separates the prefixed target into a nonempty prefix component or a nonempty target-tail component. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty target component split`

Theorem 62.72 (Functor prefix hom-carrier nonempty composite target visibility)**.** *For prefixed hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f)$$

and

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g),$$

if the composite continuation displays the visible result $\mathsf{E1}(k)$*, then the prefixed target is visible:*

$$\exists r.\ \mathsf{append}(p, c) = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{append}(p, a), \mathsf{E1}(k), \mathsf{E1}(r)).$$

Proof. Apply category source-target case analysis for nonempty composite results to the two prefixed hom-carriers. The empty-source branch already displays the target endpoint. In the visible-source branch, the source endpoint equality transports the displayed continuation back to $\mathsf{append}(p, a)$, while the result equality identifies the composite payload with k. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty result target visible`

Theorem 62.73 (Functor prefix hom-carrier nonempty composite source-target cases)**.** *For prefixed hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f)$$

and

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g),$$

if $\mathsf{Cont}(f, g, \mathsf{E1}(k))$*, then either the prefixed source is empty and the prefixed target is visible, or both source and target are visible:*

$$\mathsf{hsame}(\mathsf{append}(p, a), \mathsf{e}) \wedge \exists r.\ \mathsf{append}(p, c) = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{append}(p, a), \mathsf{E1}(k), \mathsf{E}$$
$$\vee$$
$$\exists s, r.\ \mathsf{append}(p, a) = \mathsf{E1}(s) \wedge \mathsf{append}(p, c) = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}$$

Proof. Apply category source-target case analysis to the two prefixed hom-carriers at the displayed composite result $\mathsf{E1}(k)$. The empty-source branch gives the empty source classifier and visible target witness. The visible-source branch keeps the displayed source endpoint, identifies the result payload with k, and repacks the target endpoint and continuation witness. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty result source target cases`

Theorem 62.74 (Functor prefix hom-carrier nonempty composite source visibility)**.** *For prefixed hom-carriers*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), f)$$

and

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, b), \mathsf{append}(p, c), g),$$

if $\mathsf{Cont}(f, g, \mathsf{E1}(k))$ *and the prefixed source is not history-same to* e*, then both the source and target endpoints are visible:*

$$\exists s, r.\ \mathsf{append}(p, a) = \mathsf{E1}(s) \wedge \mathsf{append}(p, c) = \mathsf{E1}(r) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(r) \wedge \mathsf{Cont}(\mathsf{E1}(s), \mathsf{E}$$

Proof. Apply the source-target case theorem for nonempty prefixed composites. Its empty-source branch contradicts the displayed nonempty-source classifier assumption. The remaining branch is exactly the visible source and target case, with unary witnesses for the source tail, composite payload, and target tail. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier comp nonempty result source visible`

62.2 The certificate

Definition 62.75 (Functor source specification)**.** *The source is a pair of certified categories.*

Definition 62.76 (Functor pattern specification)**.** *The pattern is structure-preserving mapping formation over object and morphism carriers.*

Definition 62.77 (Functor classifier specification)**.** *The classifier relates functors pointwise on object images and morphism images using the codomain category classifiers.*

Definition 62.78 (Functor stability certificate)**.** *The stability certificate consists of:*

1. *pointwise object-map classifier reflexivity*
2. *pointwise object-map classifier symmetry*
3. *pointwise object-map classifier transitivity*
4. *pointwise morphism-map classifier reflexivity*
5. *pointwise morphism-map classifier symmetry*
6. *pointwise morphism-map classifier transitivity*
7. *identity preservation*
8. *composition preservation*
9. *stability under source and target classifier replacement*

Definition 62.79 (Functor ledger policy)**.** *Record the two category ledgers and the preservation witnesses.*

Definition 62.80 ($\mathsf{Functor}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{Functor}^{\uparrow}$*.*

Theorem 62.81 (Concrete prefix functor certificate)**.** *For every unary history p, the object assignment $a \mapsto \mathsf{append}(p, a)$ and morphism assignment $f \mapsto f$ form a concrete* $\mathsf{Functor}^{\uparrow}$ *certificate on the unary-continuation* $\mathsf{Cat}^{\uparrow}$ *carrier. The carrier field* $\mathsf{PrefixFunctorCarrier}(p)$ *is supplied by Definition 62.2; this same carrier theorem packages the prefix preservation, prefix reflection, and prefixed composition fields. Public readback is supplied by Definition 62.9.*

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixConcreteCertificate`

Proof. Use the roadmap ten-field template for the prefix endofunctor target.

1. The traditional target is the prefix endofunctor on the unary continuation category, and the BEDC source is the checked unary-continuation $\mathsf{Cat}^{\uparrow}$ carrier.
2. The carrier choice is $\mathsf{PrefixFunctorCarrier}(p)$. Apply Definition 62.2 to the unary history p to obtain this field.
3. The classifier choice is pointwise: objects are compared by the unary-continuation object classifier, and morphisms are compared by the displayed hom-carrier classifier. No quotient or external equality is added.
4. The stability fields are identity preservation, composition preservation, and source-target classifier transport. These are the carrier fields packaged by Definition 62.2.

5. The ledger records that the morphism map is the identity map on displayed morphism histories, while the object ledger stores only the fixed unary prefix p.
6. The core theorem list and exactness predicate are the displayed carrier, identity, composition, transport, and readback clauses. The standard bridge is the public comparison of the prefixed composite with the original composite, supplied by Definition 62.9. The resulting status is a concrete paper certificate for $\mathsf{Functor}^{\uparrow}$.

□

Theorem 62.82 (Semantic $\mathsf{Functor}^{\uparrow}$ prefix certificate)**.** *For every unary history p, the source, pattern, classifier, stability, and ledger data of Definitions 62.75–62.79, together with Definition 62.81, form a semantic naming certificate for the prefix endofunctor interface $\mathsf{Functor}^{\uparrow}$ over the unary-continuation $\mathsf{Cat}^{\uparrow}$. The carrier is $\mathsf{PrefixFunctorCarrier}(p)$; the object component sends a to $\mathsf{append}(p, a)$; the hom component sends each displayed morphism history to itself; the classifier is pointwise hsame on object histories and hom histories; and the ledger records the fixed prefix p together with the identity map on displayed morphism histories.*

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixSemanticNameCert`

Proof. Start from the concrete certificate of Definition 62.81. Its carrier field is $\mathsf{PrefixFunctorCarrier}(p)$, supplied by Definition 62.2, so the semantic carrier row is the same object-level package rather than a separate witness.

The classifier row is pointwise. Object images are compared by hsame after the prefix operation, and hom images are compared by hsame on the displayed hom histories because the morphism map is the identity map. The source and pattern rows are therefore exactly the source, pattern, and classifier specifications already listed for $\mathsf{Functor}^{\uparrow}$.

The stability rows are the checked preservation clauses consumed by the concrete certificate. The prefix carrier theorem Definition 62.2 packages identity preservation, composition preservation, and hsame transport, while Definition 62.9 supplies the public readback clause.

Finally, the ledger row contains no hidden quotient data. It records the unary prefix p on objects and records that morphism histories are read through the identity map. These rows are precisely the ten-field certificate data, so they assemble as the semantic naming certificate for the prefix $\mathsf{Functor}^{\uparrow}$ interface over $\mathsf{Cat}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier semanticNameCert`

Theorem 62.83 (Prefix functor two-sided identity law)**.** *For histories $p, a, b, f, left, right$, suppose p is unary, f is a category hom-carrier from a to b, $\mathsf{Cont}(\mathsf{e}, f, left)$ holds, and $\mathsf{Cont}(f, \mathsf{e}, right)$ holds. Then*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), left) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), right) \wedge \mathsf{hsame}($$

Proof. First transport the displayed hom-carrier for f across the unary prefix. Apply Definition 62.2 to the unary witness for p and to the carrier from a to b. This gives f as a hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(p, b)$.

Next apply Definition 61.102 to this prefixed carrier and to the two continuation ledgers $\mathsf{Cont}(\mathsf{e}, f, left)$ and $\mathsf{Cont}(f, \mathsf{e}, right)$.

Project the resulting conjunction. The first two projections are the prefixed carrier conclusions for $left$ and $right$; the remaining projections are $\mathsf{hsame}(left, f)$, $\mathsf{hsame}(right, f)$, and $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixHomCarrier two sided identity law`

Primitive boundary. $\mathsf{Functor}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 62.84 (Zero-axiom encoding note)**.** *Avoids funext by pointwise classifier sameness for object and morphism mappings.*

Definition 62.85 (Opposite functor certificate data)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates, and let $F : \mathcal{C} \to \mathcal{D}$ denote a $\mathsf{Functor}^{\uparrow}$ certificate with object map F_0 and morphism map F_1. Over the opposite category data of Definition 61.327, define $F^{\mathrm{op}} : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}^{\mathrm{op}}$ by*

$$F_0^{\mathrm{op}}(a) := F_0(a), \qquad F_1^{\mathrm{op}}(u) := F_1(u).$$

A displayed morphism $u : a \to b$ in $\mathcal{C}^{\mathrm{op}}$ is read as the same raw morphism $u : b \to a$ in $\mathcal{C}$; therefore $F_1^{\mathrm{op}}(u)$ is read as $F_1(u) : F_0(b) \to F_0(a)$ in $\mathcal{D}$, hence as a displayed morphism $F_0(a) \to F_0(b)$ in $\mathcal{D}^{\mathrm{op}}$.

Theorem 62.86 (Opposite functor certificate)**.** *If $\mathcal{C}$ and $\mathcal{D}$ satisfy the $\mathsf{Cat}^{\uparrow}$ certificate obligations and $F : \mathcal{C} \to \mathcal{D}$ satisfies the $\mathsf{Functor}^{\uparrow}$ certificate obligations of Definition 62.80, then the data of Definition 62.85 satisfy the $\mathsf{Functor}^{\uparrow}$ certificate obligations from $\mathcal{C}^{\mathrm{op}}$ to $\mathcal{D}^{\mathrm{op}}$.*

Proof. By Definition 61.328, the source and codomain opposite data are certified categories, so the source specification required by Definition 62.75 is present. The carrier and pattern fields are the same two maps F_0 and F_1, only with the hom-carrier endpoints read through the opposite category convention.

For hom-carrier landing, let $u : a \to b$ be displayed in $\mathcal{C}^{\mathrm{op}}$. By Definition 61.327, this is the original hom-carrier assertion $u : b \to a$ in $\mathcal{C}$. The carrier field of F sends it to a displayed morphism $F_1(u) : F_0(b) \to F_0(a)$ in $\mathcal{D}$. Unfolding the codomain opposite hom-carrier reads that same assertion as $F_1(u) : F_0(a) \to F_0(b)$ in $\mathcal{D}^{\mathrm{op}}$, which is exactly the landing field for F^{op}.

The classifier specification is inherited pointwise. Object classifiers are unchanged by the opposite construction, and the morphism classifier of $\mathcal{D}^{\mathrm{op}}$ is the morphism classifier of $\mathcal{D}$ on the same displayed morphism representatives. Thus the pointwise object-map, morphism-map, reflexivity, and transitivity clauses from Definition 62.77 and Definition 62.78 are the original clauses for F.

For identity preservation, the identity at a in $\mathcal{C}^{\mathrm{op}}$ is the identity at a in $\mathcal{C}$, and the identity at $F_0(a)$ in $\mathcal{D}^{\mathrm{op}}$ is the identity at $F_0(a)$ in $\mathcal{D}$. The identity preservation witness for F therefore gives the required $\mathcal{D}^{\mathrm{op}}$ morphism-classifier witness for F^{op}.

For composition preservation, let $u : a \to b$ and $v : b \to c$ be displayed in $\mathcal{C}^{\mathrm{op}}$, and let w be their displayed composite there. Unfolding opposite composition gives the original source ledger

$$\mathsf{Comp}_{\mathcal{C}}(v, u, w).$$

The composition preservation field of F gives the codomain classifier comparison between $F_1(w)$ and the composite selected by $\mathsf{Comp}_{\mathcal{D}}(F_1(v), F_1(u), -)$. By the definition of opposite composition in $\mathcal{D}^{\mathrm{op}}$, that is precisely the composite selected by

$$\mathsf{Comp}_{\mathcal{D}^{\mathrm{op}}}(F_1(u), F_1(v), -).$$

Since the morphism classifier is unchanged, this is the required composition preservation witness for F^{op}.

Stability under source and target classifier replacement is also inherited. A source-endpoint replacement in the opposite category is a target-endpoint replacement in the original category, and conversely; the original stability field applies with the endpoint order swapped. The ledger policy of Definition 62.79 records the same category ledgers and the same preservation witnesses, read through the two opposite-category certificates. □

Lean4 ✓ BEDC.Derived.FunctorUp.PrefixFunctorCarrier opposite comp assoc public readback

Definition 62.87 (Split monomorphism in a certified category)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate. A displayed morphism $f : a \to b$ is a split monomorphism when there are a displayed morphism $g : b \to a$, a source identity display $i_a : a \to a$, and a composite display $c : a \to a$ such that $\mathsf{HomCarrier}_{\mathcal{C}}(a, b, f)$, $\mathsf{HomCarrier}_{\mathcal{C}}(b, a, g)$, i_a is the identity witness at a, $\mathsf{Comp}_{\mathcal{C}}(f, g, c)$ records $g \circ_{\mathcal{C}} f$, and $c \sim_{\mathcal{C}} i_a$ in the morphism classifier at (a, a). The morphism g is a left-inverse witness for f.*

Lean4 (definition) BEDC.Derived.FunctorUp.CategorySplitMonomorphism

Theorem 62.88 (Functor preserves split monomorphisms)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map F_0 and morphism map F_1. If $f : a \to b$ is a split monomorphism in $\mathcal{C}$ with left-inverse witness $g : b \to a$, then $F_1(f) : F_0(a) \to F_0(b)$ is a split monomorphism in $\mathcal{D}$ with left-inverse witness $F_1(g) : F_0(b) \to F_0(a)$.*

Lean4 ✓ BEDC.Derived.FunctorUp.PrefixFunctorCarrier split monomorphism preserves

Proof. Unpack the source split-monomorphism data from Definition 62.87. Thus f and g are displayed hom-carriers in $\mathcal{C}$, i_a is the chosen identity display at a, and there is a composite display c with $\mathsf{Comp}_{\mathcal{C}}(f, g, c)$ and $c \sim_{\mathcal{C}} i_a$.

The hom-carrier landing field of the functor carrier sends f and g to hom-carriers $F_1(f) : F_0(a) \to F_0(b)$ and $F_1(g) : F_0(b) \to F_0(a)$ in $\mathcal{D}$. Apply the composition preservation field of Definition 62.78 to the source composite ledger $\mathsf{Comp}_{\mathcal{C}}(f, g, c)$. It supplies a codomain composite display d with

$$\mathsf{Comp}_{\mathcal{D}}(F_1(f), F_1(g), d)$$

and a morphism-classifier comparison between d and $F_1(c)$.

The morphism-classifier stability field for F transports the source comparison $c \sim_{\mathcal{C}} i_a$ to a codomain comparison $F_1(c) \sim_{\mathcal{D}} F_1(i_a)$. Identity preservation for F identifies $F_1(i_a)$ with the identity display $i_{F_0(a)}$ of $\mathcal{D}$. Symmetry and transitivity of the codomain morphism classifier, supplied by the $\mathsf{Cat}^{\uparrow}$ stability laws, compose these comparisons to obtain $d \sim_{\mathcal{D}} i_{F_0(a)}$.

The displayed morphism $F_1(g)$, the composite display d, and the classifier comparison with $i_{F_0(a)}$ form the split-monomorphism witness for $F_1(f)$ in $\mathcal{D}$. □

Definition 62.89 (Split epimorphism in a certified category)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate. A displayed morphism $f : a \to b$ is a split epimorphism when there are a displayed morphism $g : b \to a$, a target identity display $i_b : b \to b$, and a composite display $c : b \to b$ such that $\mathsf{HomCarrier}_{\mathcal{C}}(a, b, f)$, $\mathsf{HomCarrier}_{\mathcal{C}}(b, a, g)$, i_b is the identity witness at b, $\mathsf{Comp}_{\mathcal{C}}(g, f, c)$ records $f \circ_{\mathcal{C}} g$, and $c \sim_{\mathcal{C}} i_b$ in the morphism classifier at (b, b). The morphism g is a right-inverse witness for f.*

Theorem 62.90 (Functor preserves split epimorphisms)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map F_0 and morphism map F_1. If $f : a \to b$ is a split epimorphism in $\mathcal{C}$ with right-inverse witness $g : b \to a$, then $F_1(f) : F_0(a) \to F_0(b)$ is a split epimorphism in $\mathcal{D}$ with right-inverse witness $F_1(g) : F_0(b) \to F_0(a)$.*

Proof. Unpack the source split-epimorphism data from Definition 62.89. Thus f and g are displayed hom-carriers in $\mathcal{C}$, i_b is the chosen identity display at b, and there is a composite display c with $\mathsf{Comp}_{\mathcal{C}}(g, f, c)$ and $c \sim_{\mathcal{C}} i_b$.

The hom-carrier landing field of the functor carrier sends f and g to hom-carriers $F_1(f) : F_0(a) \to F_0(b)$ and $F_1(g) : F_0(b) \to F_0(a)$ in $\mathcal{D}$. Apply the composition preservation field of

Definition 62.78 to the source composite ledger $\mathsf{Comp}_{\mathcal{C}}(g, f, c)$. It supplies a codomain composite display d with

$$\mathsf{Comp}_{\mathcal{D}}(F_1(g), F_1(f), d)$$

and a morphism-classifier comparison between d and $F_1(c)$.

The morphism-classifier stability field for F transports the source comparison $c \sim_{\mathcal{C}} i_b$ to a codomain comparison $F_1(c) \sim_{\mathcal{D}} F_1(i_b)$. Identity preservation for F identifies $F_1(i_b)$ with the identity display $i_{F_0(b)}$ of $\mathcal{D}$. Symmetry and transitivity of the codomain morphism classifier, supplied by the $\mathsf{Cat}^{\uparrow}$ stability laws, compose these comparisons to obtain $d \sim_{\mathcal{D}} i_{F_0(b)}$.

The displayed morphism $F_1(g)$, the composite display d, the codomain target identity display $i_{F_0(b)}$, and the classifier comparison with $i_{F_0(b)}$ form the split-epimorphism witness for $F_1(f)$ in $\mathcal{D}$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier split epimorphism preserves`

Definition 62.91 (Split isomorphism witness pair)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate. A displayed morphism $f : a \to b$ carries a split isomorphism witness pair when it carries a split-monomorphism witness in the sense of Definition 62.87 with left-inverse witness $g_L : b \to a$ and a split-epimorphism witness in the sense of Definition 62.89 with right-inverse witness $g_R : b \to a$. Thus the package stores the two oriented composite certificates*

$$g_L \circ_{\mathcal{C}} f \sim_{\mathcal{C}} \mathsf{id}_a, \qquad f \circ_{\mathcal{C}} g_R \sim_{\mathcal{C}} \mathsf{id}_b.$$

The two inverse-side witnesses are part of the data; no equality between g_L and g_R is required by this package.

Theorem 62.92 (Functor preserves split isomorphism witness pairs)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map F_0 and morphism map F_1. If $f : a \to b$ carries a split isomorphism witness pair in $\mathcal{C}$ with left-inverse witness $g_L : b \to a$ and right-inverse witness $g_R : b \to a$, then $F_1(f) : F_0(a) \to F_0(b)$ carries a split isomorphism witness pair in $\mathcal{D}$ whose left-inverse witness is $F_1(g_L)$ and whose right-inverse witness is $F_1(g_R)$.*

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier split iso witness pair public readback`

Proof. Unpack the witness pair by Definition 62.91. Its left-oriented field is a split-monomorphism witness for f with left-inverse witness g_L, and its right-oriented field is a split-epimorphism witness for f with right-inverse witness g_R.

Apply Definition 62.88 to the left-oriented field. This gives a split-monomorphism witness in $\mathcal{D}$ for $F_1(f) : F_0(a) \to F_0(b)$, with left-inverse witness $F_1(g_L) : F_0(b) \to F_0(a)$. Its codomain composite certificate is classified with the identity display at $F_0(a)$.

Apply Definition 62.90 to the right-oriented field. This gives a split-epimorphism witness in $\mathcal{D}$ for the same forward morphism $F_1(f) : F_0(a) \to F_0(b)$, with right-inverse witness $F_1(g_R) : F_0(b) \to F_0(a)$. Its codomain composite certificate is classified with the identity display at $F_0(b)$.

The two transported fields have the same displayed forward morphism and the same image endpoints. Packing them by Definition 62.91 gives the required split isomorphism witness pair in $\mathcal{D}$. □

Definition 62.93 (Identity functor certificate data)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate with object carrier $\mathcal{O}_{\mathcal{C}}$, object classifier $\sim_{\mathcal{O}}$, hom-carrier $\mathcal{H}_{\mathcal{C}}(a, b, u)$, endpoint-indexed morphism classifier $\sim_{\mathcal{H}_{\mathcal{C}}(a,b)}$,*

identity witnesses id_a*, and composition relation* $\mathsf{Comp}_{\mathcal{C}}(u, v, w)$*. The identity functor data* $1_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ *are*

$$(1_{\mathcal{C}})_0(a) := a, \qquad (1_{\mathcal{C}})_1(u) := u.$$

The functor-indexed hom-carrier is the original endpoint-indexed hom-carrier:

$$\mathsf{FunctorHomCarrier}_{1_{\mathcal{C}}}(a, b, u) := \mathcal{H}_{\mathcal{C}}(a, b, u).$$

Object and morphism classifiers are read from the same category certificate at the same endpoints.

Theorem 62.94 (Identity functor name certificate)**.** *If* $\mathcal{C}$ *satisfies the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations of Definition 61.203, then the data of Definition 62.93 satisfy the* $\mathsf{Functor}^{\uparrow}$ *certificate obligations of Definition 62.80. Endpoint-by-endpoint, for every source pair* a, b *and morphism candidate* u*,*

$$\mathsf{FunctorHomCarrier}_{1_{\mathcal{C}}}(a, b, u) \iff \mathcal{H}_{\mathcal{C}}(a, b, u).$$

For every object a*,*

$$(1_{\mathcal{C}})_1(\mathsf{id}_a) \sim_{\mathcal{H}_{\mathcal{C}}(a,a)} \mathsf{id}_{(1_{\mathcal{C}})_0(a)}.$$

For every displayed composable pair $u : a \to b$ *and* $v : b \to c$ *with* $\mathsf{Comp}_{\mathcal{C}}(u, v, w)$ *recording* $v \circ_{\mathcal{C}} u$*,*

$$(1_{\mathcal{C}})_1(w) \sim_{\mathcal{H}_{\mathcal{C}}(a,c)} (1_{\mathcal{C}})_1(v) \circ_{\mathcal{C}} (1_{\mathcal{C}})_1(u).$$

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorPrefixIdentityCarrier semanticNameCert`

Proof. Unpack the identity functor data. The source row of Definition 62.75 is the pair $(\mathcal{C}, \mathcal{C})$, which is licensed by the displayed $\mathsf{Cat}^{\uparrow}$ certificate. The pattern row of Definition 62.76 is the pair of identity maps on objects and displayed morphisms.

The carrier landing clause is definitional. A source hom-carrier $\mathcal{H}_{\mathcal{C}}(a, b, u)$ is sent to $\mathcal{H}_{\mathcal{C}}((1_{\mathcal{C}})_0(a), (1_{\mathcal{C}})_0(b), (1_{\mathcal{C}})_1($ and this unfolds to the same proposition $\mathcal{H}_{\mathcal{C}}(a, b, u)$. This proves the displayed biconditional and shows that the classifier specification of Definition 62.77 is inherited pointwise from the category classifier.

For identity preservation, the source identity at a and the codomain identity at $(1_{\mathcal{C}})_0(a) = a$ are the same witness id_a. Morphism-classifier reflexivity from the category stability fields in Definition 61.201 gives the classifier comparison of id_a with itself.

For composition preservation, take a source composition ledger $\mathsf{Comp}_{\mathcal{C}}(u, v, w)$. Since the morphism map is the identity, the image of the source composite is w. The codomain composite of the image pair is recorded by the same ledger, now read in the codomain copy of $\mathcal{C}$. Morphism-classifier reflexivity at the endpoint pair (a, c) therefore classifies w with that codomain composite.

The remaining stability rows in Definition 62.78 are pointwise classifier laws and endpoint-replacement laws. Object classifier reflexivity, symmetry, and transitivity are the object classifier laws of $\mathcal{C}$; morphism classifier reflexivity, symmetry, and transitivity are the endpoint-indexed morphism classifier laws of $\mathcal{C}$. Source and target replacement for displayed morphisms is exactly the category endpoint transport field, because both functor maps are identities.

The ledger row of Definition 62.79 records the same category ledger on both sides, the same identity displays, and the same composition trees. Thus all carrier, source, pattern, classifier, stability, and ledger rows listed in Definition 62.80 are present for $1_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.IdentityFunctorCarrier semanticNameCert`

Theorem 62.95 (Double opposite functor certificate data identity)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying the category certificate obligations, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate satisfying the obligations of Definition 62.80. Write F_0 for its object map and F_1 for its morphism map. Let*

$$F^{\mathrm{opop}} = (F^{\mathrm{op}})^{\mathrm{op}}$$

be obtained by applying Definition 62.85 twice, with source and target categories identified through the double-opposite category data identity of Definition 61.330. Then $F^{\mathrm{opop}} : \mathcal{C}^{\mathrm{opop}} \to \mathcal{D}^{\mathrm{opop}}$ satisfies the $\mathsf{Functor}^{\uparrow}$ certificate obligations, and its functor data are componentwise identical to the data of F:

$$(F^{\mathrm{opop}})_0(a) = F_0(a), \qquad (F^{\mathrm{opop}})_1(u) = F_1(u).$$

The pointwise object-map classifier, pointwise morphism-map classifier, identity-preservation row, composition-preservation row, source-replacement stability row, target-replacement stability row, and ledger row of F^{opop} are the corresponding rows of F under the same double-opposite source and target identification.

Proof. Apply Definition 61.328 to $\mathcal{C}$ and $\mathcal{D}$. This supplies certified categories $\mathcal{C}^{\mathrm{op}}$ and $\mathcal{D}^{\mathrm{op}}$. Then Definition 62.86 supplies a certified functor

$$F^{\mathrm{op}} : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}^{\mathrm{op}}.$$

Applying the same category and functor results once more supplies a certified functor

$$F^{\mathrm{opop}} : \mathcal{C}^{\mathrm{opop}} \to \mathcal{D}^{\mathrm{opop}}.$$

By Definition 61.330, the source and target category data of $\mathcal{C}^{\mathrm{opop}}$ and $\mathcal{D}^{\mathrm{opop}}$ are the source and target category data of $\mathcal{C}$ and $\mathcal{D}$, with hom endpoints swapped twice and therefore restored.

It remains to compare the functor rows. One unfolding of Definition 62.85 gives

$$F_0^{\mathrm{op}}(a) = F_0(a), \qquad F_1^{\mathrm{op}}(u) = F_1(u).$$

A second unfolding gives

$$(F^{\mathrm{opop}})_0(a) = F_0^{\mathrm{op}}(a) = F_0(a), \qquad (F^{\mathrm{opop}})_1(u) = F_1^{\mathrm{op}}(u) = F_1(u).$$

Thus the object map and morphism map are unchanged as raw data.

The classifier rows are also unchanged. The opposite category data of Definition 61.327 leave object classifiers and morphism classifiers fixed; Definition 61.330 identifies the double-opposite classifiers with the classifiers of $\mathcal{D}$. Since Definition 62.85 does not alter F_0 or F_1, the pointwise object-map and morphism-map classifier clauses for F^{opop} are exactly the corresponding clauses for F.

For identity preservation, the identity at a in $\mathcal{C}^{\mathrm{opop}}$ is the identity at a in $\mathcal{C}$, and the identity at $F_0(a)$ in $\mathcal{D}^{\mathrm{opop}}$ is the identity at $F_0(a)$ in $\mathcal{D}$. Hence the identity-preservation witness of F^{opop} is the identity-preservation witness of F read through the double-opposite category identification.

For composition preservation, let $u : a \to b$ and $v : b \to c$ be displayed source morphisms with composite ledger $\mathsf{Comp}_{\mathcal{C}}(u, v, w)$. In the first opposite construction the source ledger is read as $\mathsf{Comp}_{\mathcal{C}}(v, u, w)$, and in the second opposite construction it is read back as $\mathsf{Comp}_{\mathcal{C}}(u, v, w)$. The same cancellation occurs in the target category:

$$\mathsf{Comp}_{\mathcal{D}^{\mathrm{opop}}}(F_1(u), F_1(v), -) = \mathsf{Comp}_{\mathcal{D}}(F_1(u), F_1(v), -).$$

Therefore the composition-preservation row for F^{opop} is the original composition-preservation row for F.

Finally, source-endpoint replacement and target-endpoint replacement are exchanged by one opposite reading and exchanged back by the second. The remaining stability fields in Definition 62.78 and the ledger fields in Definition 62.79 use the same category ledgers and the same preservation witnesses, with the double-opposite category data identified with the starting category data. Hence every functor certificate row of F^{opop} is componentwise identical to the corresponding row of F. □

Theorem 62.96 (Composition of split monomorphisms)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate satisfying the obligations of Definition 61.203. Let $f : a \to b$ and $g : b \to c$ be displayed morphisms in $\mathcal{C}$. Suppose f is a split monomorphism with left-inverse witness $f' : b \to a$, and suppose g is a split monomorphism with left-inverse witness $g' : c \to b$, in the sense of Definition 62.87. If $u : a \to c$ is a displayed composite with $\mathsf{Comp}_{\mathcal{C}}(f, g, u)$ and $\ell : c \to a$ is a displayed composite with $\mathsf{Comp}_{\mathcal{C}}(g', f', \ell)$, then u is a split monomorphism with left-inverse witness ℓ.*

Proof. Unpack the two split-monomorphism witnesses. For f this gives displayed hom-carriers for f and f', an identity display $i_a : a \to a$, and a composite display $s_f : a \to a$ with

$$\mathsf{Comp}_{\mathcal{C}}(f, f', s_f) \qquad \text{and} \qquad s_f \sim_{\mathcal{C}} i_a.$$

For g it gives displayed hom-carriers for g and g', an identity display $i_b : b \to b$, and a composite display $s_g : b \to b$ with

$$\mathsf{Comp}_{\mathcal{C}}(g, g', s_g) \qquad \text{and} \qquad s_g \sim_{\mathcal{C}} i_b.$$

Composition closure in the $\mathsf{Cat}^{\uparrow}$ stability fields makes the displayed composites u and ℓ hom-carriers over $a \to c$ and $c \to a$.

Let $R : a \to a$ be the composite display selected for u followed by ℓ, so $\mathsf{Comp}_{\mathcal{C}}(u, \ell, R)$. It remains to classify R with i_a. The four displayed factors form the chain

$$a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{g'} b \xrightarrow{f'} a.$$

The associativity row of Definition 61.201, applied twice, compares the outside bracketing $(f' \circ_{\mathcal{C}} g') \circ_{\mathcal{C}} (g \circ_{\mathcal{C}} f)$ with the bracketing $f' \circ_{\mathcal{C}} ((g' \circ_{\mathcal{C}} g) \circ_{\mathcal{C}} f)$. Thus R is classified with the display obtained by first using the inner composite s_g, then composing with f, and finally composing with f'.

Since $s_g \sim_{\mathcal{C}} i_b$, composition congruence replaces $s_g \circ_{\mathcal{C}} f$ by $i_b \circ_{\mathcal{C}} f$. The target-identity law classifies this composite with f. A second use of composition congruence, now after postcomposition by f', compares the resulting display with $s_f = f' \circ_{\mathcal{C}} f$. Transitivity of the morphism classifier and the stored comparison $s_f \sim_{\mathcal{C}} i_a$ give $R \sim_{\mathcal{C}} i_a$.

The hom-carriers for u and ℓ, the identity display i_a, the composite ledger $\mathsf{Comp}_{\mathcal{C}}(u, \ell, R)$, and the classifier comparison $R \sim_{\mathcal{C}} i_a$ are exactly the data required by Definition 62.87. Hence u is a split monomorphism with left-inverse witness ℓ. □

Lean4 ✓ `BEDC.Derived.FunctorUp.CategorySplitMonomorphism composition closed`

Theorem 62.97 (Split isomorphism inverse witnesses coincide)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate satisfying the obligations of Definition 61.203. Let $f : a \to b$ carry a split isomorphism witness pair in the sense of Definition 62.91, with left-inverse witness $g_L : b \to a$ and right-inverse witness $g_R : b \to a$. Then the two inverse-side witnesses are classified together in the hom carrier from b to a:*

$$g_L \sim_{\mathcal{C}} g_R.$$

Proof. Unpack the split-isomorphism witness pair. The left-oriented field supplies a displayed composite $c_a : a \to a$ for $g_L \circ_{\mathcal{C}} f$ and a classifier comparison

$$c_a \sim_{\mathcal{C}} \mathsf{id}_a.$$

The right-oriented field supplies a displayed composite $c_b : b \to b$ for $f \circ_{\mathcal{C}} g_R$ and a classifier comparison

$$c_b \sim_{\mathcal{C}} \mathsf{id}_b.$$

Choose the displayed composites needed to compare both inverse witnesses through the common three-factor string

$$b \xrightarrow{g_R} a \xrightarrow{f} b \xrightarrow{g_L} a.$$

Associativity for the $\mathsf{Cat}^{\uparrow}$ composition relation classifies the right-associated composite $g_L \circ_{\mathcal{C}} (f \circ_{\mathcal{C}} g_R)$ with the left-associated composite $(g_L \circ_{\mathcal{C}} f) \circ_{\mathcal{C}} g_R$, expressed in the paper's composition orientation as the two bracketings of g_R, f, g_L.

On the right-associated side, replace the displayed composite $f \circ_{\mathcal{C}} g_R$ by the target identity id_b using the right-inverse comparison and classifier-respecting composition with the fixed factor g_L. The right-unit law then classifies the resulting composite with g_L.

On the left-associated side, replace the displayed composite $g_L \circ_{\mathcal{C}} f$ by the source identity id_a using the left-inverse comparison and classifier-respecting composition with the fixed factor g_R. The left-unit law then classifies the resulting composite with g_R.

The two reductions start from associativity-related bracketings of the same three displayed factors. Transitivity of the morphism classifier therefore compares g_L with g_R; symmetry orients the comparison as displayed. □

Theorem 62.98 (Functor hom-carrier codomain evaluation)**.** *Let C and E be certified categories, and let $H : C \to E$ be a certified functor with object map H_0. Write $\mathsf{FunctorHomCarrier}_H(x, y)$ for the predicate on codomain morphism candidates m abbreviated by*

$$\mathsf{FunctorHomCarrier}_H(x, y, m) \equiv \mathsf{HomCarrier}_E(H_0(x), H_0(y), m).$$

Then for every pair of source objects x, y and every codomain morphism candidate m, the functor-indexed hom-carrier classifier at x, y evaluates to the codomain category hom-carrier classifier at the image endpoints:

$$\mathsf{FunctorHomCarrier}_H(x, y, m) \iff \mathsf{HomCarrier}_E(H_0(x), H_0(y), m).$$

Proof. The functor carrier of Definition 62.1 supplies the object map H_0. The category carrier of Definition 61.1 supplies, in the codomain category E, a hom-carrier classifier for each ordered pair of codomain objects. Applying that category classifier to the image pair $H_0(x), H_0(y)$ is exactly the displayed abbreviation for $\mathsf{FunctorHomCarrier}_H(x, y, -)$. The pointwise classifier discipline in Definition 62.77 and Definition 62.84 reads classifier equality at the morphism argument m, so unfolding the abbreviation gives the stated biconditional. □

Theorem 62.99 (Functor composite preservation witnesses)**.** *Let C, D, E be certified categories. Let $F : C \to D$ and $G : D \to E$ be certified $\mathsf{Functor}^{\uparrow}$ carriers with object maps F_0, G_0 and morphism maps F_1, G_1. Write H for the composite pair with*

$$H_0(x) = G_0(F_0(x)), \qquad H_1(u) = G_1(F_1(u)).$$

Then H has identity and composition preservation witnesses under the morphism classifier of E: for every object x of C, $H_1(\mathsf{id}_x)$ is classified with $\mathsf{id}_{H_0(x)}$, and for every composable pair $u : x \to y$, $v : y \to z$ in C, $H_1(v \circ_C u)$ is classified with $H_1(v) \circ_E H_1(u)$.

Proof. The functor carrier of Definition 62.1 supplies the object and morphism maps for F and G. The stability certificate in Definition 62.78 supplies identity preservation, composition preservation, and classifier-replacement stability, all read through the codomain morphism classifier. By Definition 62.98, the hom-carrier classifier for the composite pair H is the hom-carrier classifier of E at the image endpoints $H_0(x)$ and $H_0(y)$.

Fix an object x of C. The identity field of F classifies $F_1(\mathsf{id}_x)$ with $\mathsf{id}_{F_0(x)}$ in D. The classifier-replacement stability field of G transports that comparison through G_1, giving a classifier witness in E between $G_1(F_1(\mathsf{id}_x))$ and $G_1(\mathsf{id}_{F_0(x)})$. The identity field of G, applied at $F_0(x)$, classifies $G_1(\mathsf{id}_{F_0(x)})$ with $\mathsf{id}_{G_0(F_0(x))}$. Morphism-classifier transitivity from the category stability certificate in Definition 61.201 composes the two comparisons, giving the identity witness for H.

Fix composable morphisms $u : x \to y$ and $v : y \to z$ in C. The composition field of F classifies $F_1(v \circ_C u)$ with $F_1(v) \circ_D F_1(u)$ in D. The classifier-replacement stability field of G transports this comparison through G_1, giving a classifier witness in E between $G_1(F_1(v \circ_C u))$ and $G_1(F_1(v) \circ_D F_1(u))$. The composition field of G, applied to the composable pair $F_1(u), F_1(v)$, classifies $G_1(F_1(v) \circ_D F_1(u))$ with $G_1(F_1(v)) \circ_E G_1(F_1(u))$. Transitivity in E gives the composition witness for H, and Definition 62.79 records these two preservation ledgers as the composite functor ledger data. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorCompositePreservationWitnesses`

Theorem 62.100 (Functor composition carries the hom-carrier classifier)**.** *Let C, D, E be certified categories. Let $F : C \to D$ and $G : D \to E$ be certified* $\mathsf{Functor}^{\uparrow}$ *carriers with object maps F_0, G_0 and morphism maps F_1, G_1. Let $H = G \circ F$ be the composite carrier whose object and morphism maps are*

$$H_0(x) = G_0(F_0(x)), \qquad H_1(u) = G_1(F_1(u)).$$

For every pair of source objects x, y and every codomain morphism candidate m, the hom-carrier classifier carried by H is pointwise the codomain hom-carrier classifier at the composed image endpoints:

$$\mathsf{FunctorHomCarrier}_{G\circ F}(x, y, m) \iff \mathsf{HomCarrier}_E(G_0(F_0(x)), G_0(F_0(y)), m).$$

Proof. The carrier fields in Definition 62.1 give the maps F_0, F_1, G_0, G_1, and Definition 62.99 supplies the identity and composition preservation witnesses for the composite pair H under the morphism classifier of E. Thus H is read as the certified functor $C \to E$ with object map H_0.

Apply Definition 62.98 to this functor H. It gives

$$\mathsf{FunctorHomCarrier}_H(x, y, m) \iff \mathsf{HomCarrier}_E(H_0(x), H_0(y), m).$$

By the displayed definition of the composite object map, the right-hand side is exactly $\mathsf{HomCarrier}_E(G_0(F_0(x)), G_0(F$
The classifier discipline of Definition 62.77 and the pointwise reading in Definition 62.84 identify classifier sameness by this morphism-argument equivalence, so the composite functor carries precisely the composed hom-carrier classifier. □

Proposition 62.101 (Functor composition identity needs morphism-sameness respect)**.** *Let C, D, E be finite certified categories, and let $F : C \to D$ and $G : D \to E$ be carrier pairs with object maps F_0, G_0 and morphism maps F_1, G_1. Write $m \sim_D n$ and $r \sim_E s$ for the endpoint-indexed morphism classifiers, and suppose the E-classifier is symmetric and transitive. Assume the raw identity and raw composition preservation ledgers for F and G are present in the sense of Definition 62.1. Fix an object x of C and set $H = G \circ F$ with $H_0(x) = G_0(F_0(x))$ and $H_1(u) = G_1(F_1(u))$. If*

$$F_1(\mathsf{id}_x) \sim_D \mathsf{id}_{F_0(x)}$$

and

$$G_1(\mathsf{id}_{F_0(x)}) \sim_E \mathsf{id}_{G_0(F_0(x))}$$

but

$$\neg\big(G_1(F_1(\mathsf{id}_x)) \sim_E G_1(\mathsf{id}_{F_0(x)})\big),$$

then H has no identity preservation witness at x:

$$\neg\big(H_1(\mathsf{id}_x) \sim_E \mathsf{id}_{H_0(x)}\big).$$

Proof. Assume that the displayed composite identity witness exists. Unfolding the composite carrier gives

$$G_1(F_1(\mathsf{id}_x)) \sim_E \mathsf{id}_{G_0(F_0(x))}.$$

The raw identity ledger for G gives

$$G_1(\mathsf{id}_{F_0(x)}) \sim_E \mathsf{id}_{G_0(F_0(x))}.$$

Symmetry reverses the second comparison, and transitivity composes it with the first. Hence

$$G_1(F_1(\mathsf{id}_x)) \sim_E G_1(\mathsf{id}_{F_0(x)}),$$

contradicting the displayed non-respect premise. Thus the composite identity witness cannot exist.

The comparison supplied by F is only a D-morphism classifier witness between $F_1(\mathsf{id}_x)$ and $\mathsf{id}_{F_0(x)}$. To use the raw identity ledger for G, that witness must be transported through G_1. This is exactly the classifier-replacement stability field in Definition 62.78 and the step used in Definition 62.99. Raw composition preservation does not supply this transport, so it cannot restore identity preservation for $G \circ F$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorComposition identity sameness respect required`

Theorem 62.102 (Raw functor hom-carrier landing obstruction)**.** *There are finite two-object certified categories C and D, together with an object map F_0, a raw morphism map F_1, and strict raw identity and composition preservation ledgers, such that a displayed source hom is not carried to the endpoint-indexed codomain hom-carrier required by* $\mathsf{Functor}^{\uparrow}$*. Take object and morphism classifiers in both categories to be literal equality. Let C have objects $0, 1$, raw morphisms $\mathsf{id}_0, \mathsf{id}_1, a$, and hom-carrier table $\mathsf{HomCarrier}_C(0, 0, \mathsf{id}_0)$, $\mathsf{HomCarrier}_C(1, 1, \mathsf{id}_1)$, and $\mathsf{HomCarrier}_C(0, 1, a)$, with no other hom-carrier witnesses. Its legal composites are $\mathsf{id}_0 \circ_C \mathsf{id}_0 = \mathsf{id}_0$, $\mathsf{id}_1 \circ_C \mathsf{id}_1 = \mathsf{id}_1$, $a \circ_C \mathsf{id}_0 = a$, and $\mathsf{id}_1 \circ_C a = a$. Let D have objects $0, 1$, raw morphisms $\mathsf{id}_0, \mathsf{id}_1, b$, and hom-carrier table $\mathsf{HomCarrier}_D(0, 0, \mathsf{id}_0)$, $\mathsf{HomCarrier}_D(1, 1, \mathsf{id}_1)$, and $\mathsf{HomCarrier}_D(1, 0, b)$, with no other hom-carrier witnesses. Its legal composites are $\mathsf{id}_0 \circ_D \mathsf{id}_0 = \mathsf{id}_0$, $\mathsf{id}_1 \circ_D \mathsf{id}_1 = \mathsf{id}_1$, $b \circ_D \mathsf{id}_1 = b$, and $\mathsf{id}_0 \circ_D b = b$. Define $F_0(0) = 0$, $F_0(1) = 1$, $F_1(\mathsf{id}_0) = \mathsf{id}_0$, $F_1(\mathsf{id}_1) = \mathsf{id}_1$, and $F_1(a) = b$. Equip the raw morphism carrier of D with an auxiliary total raw operation $\star_D$ whose values on the image pairs of source composites are $\mathsf{id}_0 \star_D \mathsf{id}_0 = \mathsf{id}_0$, $\mathsf{id}_1 \star_D \mathsf{id}_1 = \mathsf{id}_1$, $b \star_D \mathsf{id}_0 = b$, and $\mathsf{id}_1 \star_D b = b$. Then the object-map field, raw morphism-map field, identity preservation field, and raw composition preservation field are all satisfied. Nevertheless $\mathsf{HomCarrier}_C(0, 1, a)$ and $\neg\,\mathsf{HomCarrier}_D(F_0(0), F_0(1), F_1(a))$. Thus endpoint-indexed hom-carrier landing is not entailed by those raw fields.*

Proof. The displayed hom-carrier and composite tables make C and D certified finite categories in the sense of Definition 61.1. In each category every legal composite contains only identity morphisms except for the single displayed nonidentity arrow, so the left and right identity laws hold strictly and every legal threefold composite reduces to the same displayed morphism on both

associations. This supplies the category stability fields of Definition 61.201. The map F_0 sends the two source objects to objects of D, and F_1 sends each raw source morphism to a raw morphism of D, giving the two carrier maps named in Definition 62.1. Identity preservation is strict because $F_1(\mathsf{id}_0) = \mathsf{id}_0 = \mathsf{id}_{F_0(0)}$ and $F_1(\mathsf{id}_1) = \mathsf{id}_1 = \mathsf{id}_{F_0(1)}$. For raw composition preservation, the only composable pairs in C are the four listed pairs. Applying F_1 to their composites gives respectively id_0, id_1, b, and b. The displayed values of $\star_D$ give exactly the same four results, so the raw composition preservation field from Definition 62.78 is satisfied on all source composites. The source hom assertion $\mathsf{HomCarrier}_C(0, 1, a)$ is one of the constructors in the table for C. On the codomain side, $F_0(0) = 0$, $F_0(1) = 1$, and $F_1(a) = b$, so the required endpoint-indexed landing assertion is $\mathsf{HomCarrier}_D(0, 1, b)$. The table for D has no such generator; by the closed-generation discipline of chapter 20 and the indexed hom-carrier reading in Definition 61.200, no hidden witness is available. Hence $\neg\,\mathsf{HomCarrier}_D(F_0(0), F_0(1), F_1(a))$. By Definition 62.98, a certified $\mathsf{Functor}^{\uparrow}$ carrier reads the source-indexed functor hom predicate as the codomain hom-carrier at the image endpoints. The finite table above satisfies the raw preservation data while failing exactly that predicate at $(0, 1, a)$, so the endpoint-indexed hom-carrier landing field is a separate obligation. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorHomCarrier landing obstruction`

Theorem 62.103 (Raw functor source hom target-one morphism cases)**.** *For histories s, m, any raw source hom whose target endpoint is the one-object code has exactly the identity-one or forward-arrow shape:*

$$\mathsf{RawFunctorSourceHom}(s, \mathsf{E1}(\mathsf{e}), m) \Longrightarrow (\mathsf{hsame}(s, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(m, \mathsf{E1}(\mathsf{e}))) \vee (\mathsf{hsame}(s, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{E1}(\mathsf{E1}(\mathsf{e})))).$$

Proof. Invert the closed-generated source hom relation at the displayed target endpoint. The zero-identity constructor has empty target and is eliminated. The identity-one constructor gives the left disjunct by reflexivity, while the forward-arrow constructor gives the right disjunct by reflexivity on the empty source and double-one morphism code. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorSourceHom target one morphism cases`

Theorem 62.104 (Raw functor target hom source-one morphism cases)**.** *For histories t, m, any raw target hom whose source endpoint is the one-object code has exactly the identity-one or reverse-arrow shape:*

$$\mathsf{RawFunctorTargetHom}(\mathsf{E1}(\mathsf{e}), t, m) \Longrightarrow (\mathsf{hsame}(t, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(m, \mathsf{E1}(\mathsf{e}))) \vee (\mathsf{hsame}(t, \mathsf{e}) \wedge \mathsf{hsame}(m, \mathsf{E1}(\mathsf{E1}(\mathsf{e})))).$$

Proof. Invert the closed-generated target hom relation at the displayed source endpoint. The zero-identity constructor has empty source and is eliminated. The identity-one constructor gives the left disjunct by reflexivity, while the reverse-arrow constructor gives the right disjunct by reflexivity on the empty target and double-one morphism code. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorTargetHom source one morphism cases`

Theorem 62.105 (Raw functor source hom empty-morphism endpoint readback)**.** *For histories s, t, any raw source hom witness whose displayed morphism is empty reads back empty endpoints:*

$$\mathsf{RawFunctorSourceHom}(s, t, \mathsf{e}) \Longrightarrow \mathsf{hsame}(s, \mathsf{e}) \wedge \mathsf{hsame}(t, \mathsf{e}).$$

Proof. The source hom relation is closed-generated by the two identity constructors and the forward-arrow constructor. Constructor inversion at the concrete empty morphism eliminates the non-empty source-hom cases. The surviving empty identity case exposes empty source and target endpoints, and reflexivity supplies both history classifiers. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorSourceHom empty morphism endpoint readback`

Theorem 62.106 (Raw functor source hom forward-arrow endpoint readback)**.** *For histories s, t, any raw source hom witness whose displayed morphism is the forward-arrow code reads back the source and target endpoints:*

$$\mathsf{RawFunctorSourceHom}(s, t, \mathsf{E1}(\mathsf{E1}(\mathsf{e}))) \Longrightarrow \mathsf{hsame}(s, \mathsf{e}) \wedge \mathsf{hsame}(t, \mathsf{E1}(\mathsf{e})).$$

Proof. The source hom relation is closed-generated by the two identity constructors and the forward-arrow constructor. Constructor inversion at the concrete forward-arrow morphism eliminates the two identity cases. The remaining forward-arrow case exposes the endpoints e and $\mathsf{E1}(\mathsf{e})$, and reflexivity supplies the two history classifiers. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorSourceHom forward arrow endpoint readback`

Theorem 62.107 (Raw functor target hom reverse-arrow endpoint readback)**.** *For histories s, t, any raw target hom witness whose displayed morphism is the reverse-arrow code reads back the source and target endpoints:*

$$\mathsf{RawFunctorTargetHom}(s, t, \mathsf{E1}(\mathsf{E1}(\mathsf{e}))) \Longrightarrow \mathsf{hsame}(s, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(t, \mathsf{e}).$$

Proof. The target hom relation is closed-generated by the two identity constructors and the reverse-arrow constructor. Constructor inversion at the concrete reverse-arrow morphism eliminates the two identity cases. The surviving reverse-arrow case exposes the endpoints $\mathsf{E1}(\mathsf{e})$ and e, and reflexivity supplies the two history classifiers. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorTargetHom reverse arrow endpoint readback`

Theorem 62.108 (Raw functor target hom empty morphism endpoint readback)**.** *For histories s, t, any raw target hom witness whose displayed morphism is the empty morphism code reads back both endpoints as empty:*

$$\mathsf{RawFunctorTargetHom}(s, t, \mathsf{e}) \Longrightarrow \mathsf{hsame}(s, \mathsf{e}) \wedge \mathsf{hsame}(t, \mathsf{e}).$$

Proof. The raw target hom relation is closed-generated by the two identity constructors and the reverse-arrow constructor. Constructor inversion at the concrete empty morphism eliminates every constructor except the empty identity case, whose endpoints are both e. □

Lean4 ✓ `BEDC.Derived.FunctorUp.RawFunctorTargetHom empty morphism endpoint readback`

Theorem 62.109 (Raw functor opposite arrow same-endpoint obstruction)**.** *For histories s, t, the forward-arrow source hom and reverse-arrow target hom cannot share the same endpoints:*

$$\mathsf{RawFunctorSourceHom}(s, t, \mathsf{E1}(\mathsf{E1}(\mathsf{e}))) \Longrightarrow \mathsf{RawFunctorTargetHom}(s, t, \mathsf{E1}(\mathsf{E1}(\mathsf{e}))) \Longrightarrow \bot.$$

Proof. Apply Definition 62.106 to the source witness. This gives $\mathsf{hsame}(s, \mathsf{e})$ and $\mathsf{hsame}(t, \mathsf{E1}(\mathsf{e}))$.

Apply Definition 62.107 to the target witness. This gives $\mathsf{hsame}(s, \mathsf{E1}(\mathsf{e}))$ and $\mathsf{hsame}(t, \mathsf{e})$.

Using symmetry and transitivity of history-sameness, the two readbacks identify e with $\mathsf{E1}(\mathsf{e})$ through the common endpoint s; equivalently, the endpoint t gives the same contradiction in the opposite order. This is excluded by BEDC history constructor no-confusion, so the paired raw hom witnesses cannot coexist. □

Proposition 62.110 (Functor composition preservation needs morphism-sameness respect)**.** *Let C, D, E be finite certified categories, and let $F : C \to D$ and $G : D \to E$ be carrier pairs with object maps F_0, G_0 and morphism maps F_1, G_1. Write $m \sim_D n$ and $r \sim_E s$ for the endpoint-indexed morphism classifiers, and suppose the E-classifier is symmetric and transitive. Assume the raw identity and raw composition preservation ledgers for F and G are present in the sense of Definition 62.1. Fix composable morphisms $u : x \to y$ and $v : y \to z$ in C, and suppose the image pair $F_1(u), F_1(v)$ is a D-composable pair to which the raw composition ledger for G applies. Set $H = G \circ F$ with $H_0(a) = G_0(F_0(a))$ and $H_1(w) = G_1(F_1(w))$. If $F_1(v \circ_C u) \sim_D F_1(v) \circ_D F_1(u)$ and $G_1(F_1(v) \circ_D F_1(u)) \sim_E G_1(F_1(v)) \circ_E G_1(F_1(u))$ but $\neg\big(G_1(F_1(v \circ_C u)) \sim_E G_1(F_1(v) \circ_D F_1(u))\big)$, then H has no composition preservation witness at the pair (u, v): $\neg\big(H_1(v \circ_C u) \sim_E H_1(v) \circ_E H_1(u)\big)$.*

Proof. Assume that the displayed composite composition witness exists. Unfolding the composite carrier gives $G_1(F_1(v \circ_C u)) \sim_E G_1(F_1(v)) \circ_E G_1(F_1(u))$. The raw composition ledger for G on the displayed image pair gives $G_1(F_1(v) \circ_D F_1(u)) \sim_E G_1(F_1(v)) \circ_E G_1(F_1(u))$. Symmetry reverses the second comparison, and transitivity composes it with the first. Hence $G_1(F_1(v \circ_C u)) \sim_E G_1(F_1(v) \circ_D F_1(u))$, contradicting the displayed separation premise. Thus the composite composition witness cannot exist. The comparison supplied by F is only a D-morphism classifier witness between $F_1(v \circ_C u)$ and $F_1(v) \circ_D F_1(u)$. To use the raw composition ledger for G, that witness must be transported through G_1. This is exactly the classifier-replacement stability field in Definition 62.78 and the composition step used in Definition 62.99. The raw composition ledgers recorded by Definition 62.79 do not by themselves supply this transport, so they cannot restore composition preservation for $G \circ F$. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorComposition preservation sameness respect required`

Theorem 62.111 (Functor composition identity unit laws)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate satisfying Definition 62.80. Write F_0 and F_1 for its object and morphism maps. Let $1_{\mathcal{C}}$ and $1_{\mathcal{D}}$ be the identity functor certificates supplied by Definition 62.93 and Definition 62.94, and write*

$$R = F \circ 1_{\mathcal{C}}, \qquad L = 1_{\mathcal{D}} \circ F.$$

Then R and L are certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{D}$, and both are classified with F by the $\mathsf{Functor}^{\uparrow}$ classifier. Equivalently, for every source object x and every displayed source morphism $u : a \to b$,

$$R_0(x) \sim_{\mathcal{O}_{\mathcal{D}}} F_0(x), \qquad L_0(x) \sim_{\mathcal{O}_{\mathcal{D}}} F_0(x),$$

and

$$R_1(u) \sim_{\mathcal{H}_{\mathcal{D}}(F_0(a),F_0(b))} F_1(u), \qquad L_1(u) \sim_{\mathcal{H}_{\mathcal{D}}(F_0(a),F_0(b))} F_1(u).$$

Consequently, for every pair of source objects a, b and every codomain morphism candidate m,

$$\mathsf{FunctorHomCarrier}_R(a, b, m) \iff \mathsf{FunctorHomCarrier}_F(a, b, m) \iff \mathsf{FunctorHomCarrier}_L(a, b, m).$$

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier empty prefix identity unit laws`

Proof. The identity certificates are available by Definition 62.94. Applying Definition 62.99 to the pair $(1_{\mathcal{C}}, F)$ gives the identity and composition preservation witnesses for $R = F \circ 1_{\mathcal{C}}$, and applying the same theorem to the pair $(F, 1_{\mathcal{D}})$ gives the corresponding witnesses for $L = 1_{\mathcal{D}} \circ F$. Together with the source, pattern, classifier, and ledger rows listed in Definition 62.80, these are certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{D}$.

Unfold the identity functor data of Definition 62.93. For every source object x,

$$(F \circ 1_{\mathcal{C}})_0(x) = F_0((1_{\mathcal{C}})_0(x)) = F_0(x),$$

and

$$(1_{\mathcal{D}} \circ F)_0(x) = (1_{\mathcal{D}})_0(F_0(x)) = F_0(x).$$

Object-classifier reflexivity in the codomain category, recorded by Definition 61.201, gives the two displayed object comparisons.

For a displayed source morphism $u : a \to b$, the same unfolding gives

$$(F \circ 1_{\mathcal{C}})_1(u) = F_1((1_{\mathcal{C}})_1(u)) = F_1(u),$$

and

$$(1_{\mathcal{D}} \circ F)_1(u) = (1_{\mathcal{D}})_1(F_1(u)) = F_1(u).$$

Morphism-classifier reflexivity in the endpoint-indexed hom classifier of $\mathcal{D}$ gives both displayed morphism comparisons. By the pointwise classifier specification in Definition 62.77 and the reflexivity row in Definition 62.78, these object and morphism comparisons are exactly the $\mathsf{Functor}^{\uparrow}$ classifier witnesses $R \sim F$ and $L \sim F$.

Finally apply Definition 62.98 to F and apply Definition 62.100 to the two composites. Since the identity functor has identity object map on both $\mathcal{C}$ and $\mathcal{D}$, all three functor-indexed hom-carrier predicates reduce to

$$\mathsf{HomCarrier}_{\mathcal{D}}(F_0(a), F_0(b), m).$$

This gives the displayed biconditional chain. □

Theorem 62.112 (Functor composition associativity classifier law)**.** *Let $\mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{L}$ be certified $\mathsf{Cat}^{\uparrow}$ packages satisfying Definition 61.203. Let $F : \mathcal{C} \to \mathcal{D}$, $G : \mathcal{D} \to \mathcal{E}$, and $H : \mathcal{E} \to \mathcal{L}$ be certified $\mathsf{Functor}^{\uparrow}$ carriers satisfying Definition 62.80, with object maps F_0, G_0, H_0 and morphism maps F_1, G_1, H_1. Write*

$$A = (H \circ G) \circ F, \qquad B = H \circ (G \circ F).$$

Then A and B are certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{L}$, and they are classified together by the $\mathsf{Functor}^{\uparrow}$ classifier. Explicitly, for every source object x,

$$A_0(x) \sim_{\mathcal{O}_{\mathcal{L}}} B_0(x).$$

For every displayed source morphism $u : a \to b$,

$$A_1(u) \sim_{\mathcal{H}_{\mathcal{L}}(H_0(G_0(F_0(a))), H_0(G_0(F_0(b))))} B_1(u).$$

Moreover, the functor-indexed hom-carrier classifiers agree pointwise: for every pair of source objects a, b and every codomain morphism candidate m,

$$\mathsf{FunctorHomCarrier}_A(a, b, m) \iff \mathsf{FunctorHomCarrier}_B(a, b, m).$$

Lean4 ✓ `BEDC.Derived.FunctorUp.PrefixFunctorCarrier append assoc comp public readback`

Proof. Apply Definition 62.99 to F and G to obtain the preservation witnesses for $G \circ F$, and apply the same theorem to G and H to obtain the preservation witnesses for $H \circ G$. Applying it again to F with $H \circ G$ gives the preservation witnesses for $A = (H \circ G) \circ F$, and applying it to $G \circ F$

with H gives the preservation witnesses for $B = H \circ (G \circ F)$. Together with the source, pattern, classifier, and ledger rows of Definition 62.80, these are certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{L}$.

Unfold the composite object maps. For every object x of $\mathcal{C}$,

$$A_0(x) = (H \circ G)_0(F_0(x)) = H_0(G_0(F_0(x))),$$

and

$$B_0(x) = H_0((G \circ F)_0(x)) = H_0(G_0(F_0(x))).$$

The two object images are the same displayed object of $\mathcal{L}$, so object classifier reflexivity from Definition 61.201 gives $A_0(x) \sim_{\mathcal{O}_{\mathcal{L}}} B_0(x)$.

The morphism maps unfold in the same way. For every displayed source morphism $u : a \to b$,

$$A_1(u) = (H \circ G)_1(F_1(u)) = H_1(G_1(F_1(u))),$$

and

$$B_1(u) = H_1((G \circ F)_1(u)) = H_1(G_1(F_1(u))).$$

Morphism-classifier reflexivity in the endpoint-indexed hom classifier of $\mathcal{L}$ gives the displayed morphism comparison. By the pointwise classifier specification of Definition 62.77, these object and morphism comparisons are exactly the $\mathsf{Functor}^{\uparrow}$ classifier witness between A and B.

It remains to read the hom-carrier classifiers. Applying Definition 62.100 to the composite of F with $H \circ G$ gives, for every a, b, m,

$$\mathsf{FunctorHomCarrier}_A(a, b, m) \iff \mathsf{HomCarrier}_{\mathcal{L}}(H_0(G_0(F_0(a))), H_0(G_0(F_0(b))), m).$$

Applying the same theorem to the composite of $G \circ F$ with H gives

$$\mathsf{FunctorHomCarrier}_B(a, b, m) \iff \mathsf{HomCarrier}_{\mathcal{L}}(H_0(G_0(F_0(a))), H_0(G_0(F_0(b))), m).$$

Both functor-indexed hom-carrier predicates therefore reduce to the same codomain hom-carrier classifier, and transitivity of the biconditional gives the pointwise equivalence. □

Theorem 62.113 (Functor composition classifier category package)**.** *Let $\mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{L}$ be certified $\mathsf{Cat}^{\uparrow}$ packages satisfying Definition 61.203. Let $F : \mathcal{C} \to \mathcal{D}$, $G : \mathcal{D} \to \mathcal{E}$, and $H : \mathcal{E} \to \mathcal{L}$ be certified $\mathsf{Functor}^{\uparrow}$ carriers satisfying Definition 62.80, with object maps F_0, G_0, H_0 and morphism maps F_1, G_1, H_1. Then the certified functor carriers form a classifier-respected composition package with the following fields.*

1. *For every composable pair $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{E}$, the composite $G \circ F$ is a certified $\mathsf{Functor}^{\uparrow}$ carrier from $\mathcal{C}$ to $\mathcal{E}$, and its $\mathsf{FunctorHomCarrier}$ is the codomain hom-carrier classifier at $G_0(F_0(-))$.*
2. *For every certified $\mathsf{Cat}^{\uparrow}$ package $\mathcal{A}$, the identity carrier $1_{\mathcal{A}}$ is a certified $\mathsf{Functor}^{\uparrow}$ carrier on $\mathcal{A}$.*
3. *For every certified $K : \mathcal{A} \to \mathcal{B}$, the composites $K \circ 1_{\mathcal{A}}$ and $1_{\mathcal{B}} \circ K$ are classified with K by the $\mathsf{Functor}^{\uparrow}$ classifier, and the three $\mathsf{FunctorHomCarrier}$ predicates are pointwise the same.*
4. *For every certified triple F, G, H as above, the two triple composites $A = (H \circ G) \circ F$ and $B = H \circ (G \circ F)$ are classified together by the $\mathsf{Functor}^{\uparrow}$ classifier, and for every pair of source objects a, b and every codomain morphism candidate m,*

$$\mathsf{FunctorHomCarrier}_A(a, b, m) \iff \mathsf{FunctorHomCarrier}_B(a, b, m).$$

This is the composition structure used by the $\mathsf{NatTrans}^{\uparrow}$ *and* $\mathsf{Yoneda}^{\uparrow}$ *interfaces whenever their source functors are composed.*

Proof. The proof uses the following ten-field roadmap.

1. traditional target = category-style composition of certified functors;
2. BEDC source = $\mathsf{Cat}^{\uparrow}$ and $\mathsf{Functor}^{\uparrow}$;
3. carrier = certified functor carriers;
4. classifier = pointwise $\mathsf{Functor}^{\uparrow}$ classifier;
5. stability = closure, unit, associativity, and hom-carrier transport;
6. ledger = source and codomain category ledgers plus preservation witnesses;
7. core theorem list = Definition 62.99, Definition 62.100, Definition 62.94, Definition 62.111, and Definition 62.112;
8. exactness = hom-carrier pointwise classifier agreement;
9. standard bridge row = not claimed by this composition package; ordinary functor-category laws are only the external comparison target named by the upgrade path;
10. status = paper proof target.

For closure under composition, apply Definition 62.99 to a composable pair of certified functors. It supplies the identity and composition preservation witnesses for the composite carrier. Applying Definition 62.100 reads the resulting $\mathsf{FunctorHomCarrier}$ pointwise as the codomain hom-carrier classifier at the composed image endpoints.

For identities, Definition 62.94 supplies the certified identity functor on each certified $\mathsf{Cat}^{\uparrow}$ package. The carrier, classifier, stability, and ledger rows named in Definition 62.80 are therefore available for $1_{\mathcal{A}}$ without adding any extra source data.

For the unit laws, apply Definition 62.111. It identifies $K \circ 1_{\mathcal{A}}$ and $1_{\mathcal{B}} \circ K$ with K under the object and morphism components of the $\mathsf{Functor}^{\uparrow}$ classifier, and it gives the same pointwise $\mathsf{FunctorHomCarrier}$ predicate for all three functors.

For associativity, apply Definition 62.112. It certifies both triple composites, gives the pointwise object and morphism classifier witnesses between them, and gives the displayed agreement of their functor-indexed hom-carrier classifiers.

The four fields are exactly the composition data demanded downstream: closed composition, certified identities, unit classifiers, associativity classifiers, and exact hom-carrier transport. Repacking those fields gives the claimed classifier-respected composition package for certified $\mathsf{Functor}^{\uparrow}$ carriers. □

Theorem 62.114 (Functor prefix standard bridge)**.** *For every unary history p, the BEDC fixed-prefix* $\mathsf{Functor}^{\uparrow}$ *certificate with object map $a \mapsto \mathsf{append}(p, a)$ and identity morphism map conservatively reads as the ordinary fixed-prefix endofunctor on the unary-continuation* $\mathsf{Cat}^{\uparrow}$*. Its identity, composition, unit, associativity, and hom-carrier laws are exactly the rows cited in Definition 62.82 and Definition 62.113, read through* $\mathsf{Cat}^{\uparrow}$ *classifiers rather than host equality.*

Proof. The fixed-prefix certificate in Definition 62.82 supplies the public object map $a \mapsto \mathsf{append}(p, a)$, the identity morphism map, the preservation ledger, and the semantic NameCert rows for the unary-continuation $\mathsf{Cat}^{\uparrow}$. Reading those rows externally gives the ordinary fixed-prefix endofunctor data, because every comparison is mediated by the object and morphism classifiers of $\mathsf{Cat}^{\uparrow}$.

The functor-category laws are supplied by Definition 62.113. Its cited core rows are Definition 62.99, Definition 62.100, Definition 62.94, Definition 62.111, and Definition 62.112. These rows give closed composition, certified identities, unit classifiers, associativity classifiers, and exact pointwise hom-carrier transport for certified $\mathsf{Functor}^{\uparrow}$ carriers.

Specializing that package to the fixed-prefix carrier gives the standard endofunctor bridge. The hom-carrier readback is the exact readback already used in the package proof: the composite preservation row, the composite hom-carrier classifier row, the identity-functor NameCert row, the unit-law row, and the associativity classifier row. Thus the external functor reading uses the same classifier rows as the BEDC certificate; no host category universe, quotient, or primitive function space enters the bridge. □

Lean4 ✓ `BEDC.Derived.FunctorUp.FunctorUp StdBridge`

Definition 62.115 (Two-point classifier-transport obstruction data)**.** *Let $M = \{1, u\}$ be the idempotent monoid with unit 1 and $u \circ u = u$. Let $T = \{I, P, Q\}$ be the transformation monoid on $\{0, 1\}$ generated by the identity I and the constant maps $P(0) = P(1) = 0$ and $Q(0) = Q(1) = 1$, with composition of functions. Thus $P \circ P = P$, $Q \circ Q = Q$, $Q \circ I = Q$, $I \circ P = P$, and $Q \neq P$.*

Let $\mathcal{C}$ be the one-object category of M with equality as its morphism classifier. Let $\mathcal{D}$ be the one-object category of T with the total morphism classifier, so every two carried endomorphisms are classified. Let $\mathcal{E}$ be the one-object category of T with equality as its morphism classifier. Hom-carriers are total in each one-object category, and object classifiers are equality on the unique object.

Theorem 62.116 (FunctorUp composition obstruction without classifier replacement)**.** *The finite data of Definition 62.115 exhibit a raw functor-like map $H : \mathcal{D} \to \mathcal{E}$ preserving identities and raw composition, certified $\mathsf{Functor}^{\uparrow}$ carriers $F, G : \mathcal{C} \to \mathcal{D}$, and a certified $\mathsf{NatTrans}^{\uparrow}$ carrier $\alpha : F \Rightarrow G$ such that the functor-whiskering naturality row for $H_*\alpha$ is impossible. Consequently the classifier-replacement row in Definition 62.78 is not entailed by the raw carrier, identity, and composition rows of Definition 62.1, and $\mathsf{Functor}^{\uparrow}$ composition certification cannot be derived from those raw rows alone.*

Proof. The categories are certified in the sense of Definition 61.203. For $\mathcal{C}$ and $\mathcal{E}$, equality supplies the object and morphism classifier laws and is compatible with composition. For $\mathcal{D}$, the total morphism classifier is reflexive, symmetric, transitive, and compatible with composition. The identity, composition closure, associativity, and unit laws are the monoid laws of M and T, matching Definition 61.201.

Define $F, G : \mathcal{C} \to \mathcal{D}$ on the unique object by the unique object map and on morphisms by

$$F_1(1) = I, \qquad F_1(u) = P, \qquad G_1(1) = I, \qquad G_1(u) = Q.$$

The equations $P \circ P = P$ and $Q \circ Q = Q$ make identity and composition preservation strict. Since the codomain classifier of $\mathcal{D}$ is total, the pointwise classifier rows and the source-comparison transport rows listed in Definition 62.78 are automatic. Thus F and G satisfy the $\mathsf{Functor}^{\uparrow}$ obligations of Definition 62.80.

Let $\alpha : F \Rightarrow G$ have the single component $\alpha_* = I$. The component carrier exists because $\mathcal{D}$ has one object. Naturality at the identity morphism is immediate. Naturality at u is the $\mathcal{D}$-classifier comparison

$$G_1(u) \circ \alpha_* = Q \circ I = Q \sim_{\mathcal{D}} P = I \circ P = \alpha_* \circ F_1(u),$$

which is supplied by the total classifier of $\mathcal{D}$. The remaining component classifier, naturality-stability, vertical-composition, and identity rows of Definition 63.75 are again immediate in the total classifier, so α satisfies Definition 63.77.

Now define $H : \mathcal{D} \to \mathcal{E}$ by the unique object map and the identity map on the raw monoid T. It preserves the identity and every raw composite strictly because the two categories have the same raw monoid. It does not preserve the morphism classifier: $\mathcal{D}$ classifies $Q \sim_{\mathcal{D}} P$, while the equality classifier of $\mathcal{E}$ gives $Q \not\sim_{\mathcal{E}} P$.

The component prescribed by Definition 63.91 is

$$(H_*\alpha)_* = H_1(I) = I,$$

so the component carrier itself exists in $\mathcal{E}$. The naturality row at u, however, would have to be

$$H_1(G_1(u)) \circ H_1(I) \sim_{\mathcal{E}} H_1(I) \circ H_1(F_1(u)).$$

Unfolding the displayed maps and the monoid products turns this into

$$Q \circ I = Q \sim_{\mathcal{E}} P = I \circ P.$$

This is impossible because $\mathcal{E}$ uses equality as its morphism classifier and $Q \neq P$ as transformations of $\{0, 1\}$.

Thus the naturality classifier for α in $\mathcal{D}$ cannot be transported through H_1. This is exactly the transport step used in Definition 63.92 and, at the functor level, in Definition 62.99. If raw identity and raw composition preservation were enough for $\mathsf{Functor}^{\uparrow}$ composition certification, applying that rule to $H \circ 1_{\mathcal{D}}$, with $1_{\mathcal{D}}$ supplied by Definition 62.94, would certify the same raw map H and hence the missing classifier transport $Q \sim_{\mathcal{D}} P \Rightarrow H_1(Q) \sim_{\mathcal{E}} H_1(P)$. The displayed failure in $\mathcal{E}$ contradicts that conclusion. □

Theorem 62.117 (Split monomorphisms are left-cancellative)**.** *Let $\mathcal{C}$ be a $\mathsf{Cat}^{\uparrow}$ certificate satisfying Definition 61.203. Let $f : a \to b$ be a split monomorphism in $\mathcal{C}$ with left-inverse witness $g : b \to a$ in the sense of Definition 62.87. If $u, v : x \to a$ are displayed morphisms and $p, q : x \to b$ are displayed composites with*

$$\mathsf{Comp}_{\mathcal{C}}(u, f, p), \qquad \mathsf{Comp}_{\mathcal{C}}(v, f, q),$$

and if $p \sim_{\mathcal{C}} q$ in the morphism classifier at (x, b), then $u \sim_{\mathcal{C}} v$ in the morphism classifier at (x, a).

Proof. Unpack the split-monomorphism witness. It supplies displayed hom-carriers for f and g, an identity display $i_a : a \to a$, and a composite display $c : a \to a$ such that

$$\mathsf{Comp}_{\mathcal{C}}(f, g, c) \qquad \text{and} \qquad c \sim_{\mathcal{C}} i_a.$$

The displayed morphisms u and v, together with composition closure in Definition 61.201, make p and q hom-carriers from x to b.

Let $r_u : x \to a$ and $r_v : x \to a$ be composites of p and q followed by g:

$$\mathsf{Comp}_{\mathcal{C}}(p, g, r_u), \qquad \mathsf{Comp}_{\mathcal{C}}(q, g, r_v).$$

Composition congruence with the fixed factor g transports the hypothesis $p \sim_{\mathcal{C}} q$ to

$$r_u \sim_{\mathcal{C}} r_v.$$

Apply associativity to the chain

$$x \xrightarrow{u} a \xrightarrow{f} b \xrightarrow{g} a.$$

It compares $r_u = g \circ_{\mathcal{C}} (f \circ_{\mathcal{C}} u)$ with the composite of u followed by $c = g \circ_{\mathcal{C}} f$. Since $c \sim_{\mathcal{C}} i_a$, composition congruence with the fixed factor u compares this latter composite with the composite of u followed by i_a. The target-identity law then classifies that composite with u. Hence

$$r_u \sim_{\mathcal{C}} u.$$

The same argument applied to

$$x \xrightarrow{v} a \xrightarrow{f} b \xrightarrow{g} a$$

gives

$$r_v \sim_{\mathcal{C}} v.$$

The morphism-classifier equivalence laws, recorded in the classifier and stability rows of the $\mathsf{Cat}^{\uparrow}$ certificate, compose the comparison $u \sim_{\mathcal{C}} r_u$, the transported comparison $r_u \sim_{\mathcal{C}} r_v$, and the comparison $r_v \sim_{\mathcal{C}} v$. Therefore $u \sim_{\mathcal{C}} v$ at the endpoint pair (x, a). □

Lean4 ✓ `BEDC.Derived.FunctorUp.CategorySplitMonomorphism left cancellative`

Closure status for $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$.

Constructive story: A functor in this public construction is a certified carrier between already licensed $\mathsf{Cat}^{\uparrow}$ packages, and the concrete bridge specializes it to the fixed-prefix endofunctor that sends a unary object a to $\mathsf{append}(p, a)$ while carrying morphism tails through the corresponding endpoint-indexed hom carrier. The carrier accepts only those object and morphism maps whose images land in the codomain object and hom carriers and whose identity and composition preservation rows are witnessed under the codomain classifiers; raw maps that merely preserve unclassified composites are excluded by the finite obstruction rows. Two functor presentations classify together pointwise: their object images are compared by the target object classifier and their morphism images by the target hom classifier at the transported endpoints, so no host function equality or functor quotient is formed. The source, pattern, and ledger rows record the source and target category certificates, the object and morphism maps, identity and composition preservation, classifier-replacement stability, and the composite functor ledgers used for identity, unit, and associativity laws. The five name-certificate obligations follow from inhabited fixed-prefix data, pointwise reflexivity, symmetry, and transitivity of the category classifiers, and carrier transport supplied by the same pointwise classifier rows. A displayed functor row h lies in the signature gap of a $\mathsf{Functor}^{\uparrow}$ token p when the source-domain accepts the source and target category data, the signature relation pairs the object/morphism-map package with an answer carrier and preservation ledger, and token introduction names that answer as p; concretely, the fixed prefix, all image objects, and all image morphisms must be the ones encoded by the token. The standard bridge reads this as the ordinary fixed-prefix endofunctor and its certified composition package, then reads back only the same $\mathsf{Cat}^{\uparrow}$ classifier data. The chapter registers this classifier-respecting map construction under the name $\mathsf{Functor}^{\uparrow}$; functoriality is a ledger of landing and preservation rows, not an imported functor category, host map equality, adjunction theory, or Kan-extension surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public fixed-prefix $\mathsf{Functor}^{\uparrow}$ export is closed over the carrier, certificate, public-readback, standard bridge, finite classifier-transport obstruction, split-monomorphism consumption surface, and the required $\mathsf{Cat}^{\uparrow}$ source and target dependency.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.FunctorUp.FunctorUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: Full category equivalence, arbitrary external host functor spaces, adjunctions, Kan extensions, and limits-preservation theory are not closed here.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 63

A Concrete Naming Certificate for $\mathsf{NatTrans}^{\uparrow}$

The natural transformation interface $\mathsf{NatTrans}^{\uparrow}$ has certificate obligations between two functors from chapter 62 sharing source and target categories. The carrier is, for each object X of the source category, a morphism in the target category from the first functor's image of X to the second functor's image of X, satisfying the naturality square under the codomain category's morphism classifier. It is not a primitive 2-cell; it is admitted only as an object-indexed family of morphisms with the naturality field.

63.1 Carrier definition

Definition 63.1 (NatTrans carrier)**.** *Use an object-indexed family of codomain morphisms $\eta_X : F(X) \to G(X)$ plus naturality witnesses for every source morphism.*

Lean4 (definition) `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier`

Theorem 63.2 (NatTrans prefix component unary object suffix exactness)**.** *For histories p, q, a, η, a prefix component over the shared object suffix a is equivalent to unary membership of a together with the unsuffixed category hom carrier:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{CategoryHomCarrier}(p, q, \eta).$$

Proof. Unfold the prefix component carrier. The category hom-carrier suffix theorem peels the common unary object suffix from $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$, and the reverse direction repackages that suffixed hom-carrier with the component fields. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier unary object suffix iff`

Theorem 63.3 (NatTrans prefix component empty identity exactness)**.** *For prefix components over unary histories, the empty component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$ exists exactly when p, q, and a are unary histories and $\mathsf{hsame}(p, q)$.*

Proof. Unfold the prefix component carrier to its category hom-carrier over $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$. Category empty-identity exactness identifies the two prefixed endpoints; cancellation of the shared suffix a recovers $\mathsf{hsame}(p, q)$. Conversely, unary append closure supplies the two prefixed endpoint carriers, and the equality of p and q transports the category empty identity carrier to the component carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty identity iff`

Theorem 63.4 (NatTrans empty identity prefix transitivity)**.** *For prefix components over unary histories, if the empty component identifies* $\mathsf{append}(p, a)$ *with* $\mathsf{append}(q, a)$ *and the empty component also identifies* $\mathsf{append}(q, a)$ *with* $\mathsf{append}(r, a)$*, then* $\mathsf{hsame}(p, r)$*.*

Proof. Apply empty-identity exactness to the two component witnesses. The first component gives $\mathsf{hsame}(p, q)$, and the second gives $\mathsf{hsame}(q, r)$. Transitivity of history sameness identifies p and r. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty identity prefix trans`

Theorem 63.5 (NatTrans prefix component endomorphism component empty exactness)**.** *For prefix components whose source and target prefixes agree, a component history* η *is carried exactly when the prefix, object, and component histories are unary and the component is internally empty:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, p, a, \eta) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(\eta) \wedge \eta\mathsf{hsamee}.$$

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier endomorphism component empty iff`

Proof. Unfold the prefix component into its category hom-carrier $\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, a), \eta)$. The continuation field has matching source and target, hence right-unit uniqueness forces $\eta\mathsf{hsamee}$, while the component fields provide unary membership of p, a, and η. Conversely, an empty component is rebuilt by the prefix empty-identity exactness theorem with reflexivity of $\mathsf{hsame}(p, p)$. □

Theorem 63.6 (NatTrans one-headed source empty component exactness)**.** *For histories* p, q, a*, an empty prefix component from* $\mathsf{E1}(p)$ *to* q *over* a *exists exactly when* p *and* a *are unary and the target prefix is literally* $\mathsf{E1}(p)$*:*

$$\mathsf{NatTransPrefixComponentCarrier}(\mathsf{E1}(p), q, a, \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge q = \mathsf{E1}(p).$$

Proof. Apply empty-identity component exactness. Unary inversion on the one-headed source prefix extracts unary membership of p, and the prefix sameness specializes to the equality $q = \mathsf{E1}(p)$. Conversely, substituting this equality rebuilds the one-headed target prefix, and empty-identity exactness packages the empty component. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier e1 source empty component iff`

Theorem 63.7 (NatTrans one-headed object empty component exactness)**.** *For histories* p, q, a*, an empty prefix component over the one-headed object* $\mathsf{E1}(a)$ *exists exactly when both prefixes and the object tail are unary and the two prefixes are history-same:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, \mathsf{E1}(a), \mathsf{e}) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(a) \wedge p\mathsf{hsame}$$

Proof. Apply empty-identity component exactness at the object $\mathsf{E1}(a)$. The forward direction uses unary inversion on the object carrier to expose a. The reverse direction closes the one-headed object by unary closure and repacks the empty-identity component with the displayed prefix classifier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier e1 object empty component iff`

Theorem 63.8 (NatTrans one-headed target empty component exactness)**.** *For histories p, q, a, an empty prefix component from p to $\mathsf{E1}(q)$ over a exists exactly when q and a are unary and the source prefix is literally $\mathsf{E1}(q)$:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, \mathsf{E1}(q), a, \mathsf{e}) \iff \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(a) \wedge p = \mathsf{E1}(q).$$

Proof. Apply empty-identity component exactness. Unary inversion on the one-headed target prefix extracts unary membership of q, and the prefix sameness specializes to the equality $p = \mathsf{E1}(q)$. Conversely, substituting this equality rebuilds the one-headed source prefix, and empty-identity exactness packages the empty component. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier e1 target empty component iff`

Theorem 63.9 (NatTrans prefix component empty-source prefix exactness)**.** *For histories q, a, η, a prefix component from the empty prefix to q over a is exactly a unary target prefix, a unary object, and a category hom-carrier from a to $\mathsf{append}(q, a)$:*

$$\mathsf{NatTransPrefixComponentCarrier}(\mathsf{e}, q, a, \eta) \iff \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{CategoryHomCarrier}(a, \mathsf{apper}$$

Proof. Unfold the prefix component carrier. The source endpoint is $\mathsf{append}(\mathsf{e}, a)$, which is history-same as a by the left unit law for append. Category hom-carrier transport across this endpoint equality gives the unprefixed source form. The converse uses the same endpoint equality in the opposite direction and packages the empty prefix carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty source prefix iff`

Theorem 63.10 (NatTrans prefix component empty-target prefix exactness)**.** *For histories p, a, η, a prefix component from p to the empty prefix over a is exactly a unary source prefix, a unary object, and a category hom-carrier from $\mathsf{append}(p, a)$ to a:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, \mathsf{e}, a, \eta) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(a) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}($$

Proof. Unfold the prefix component carrier. The target endpoint is $\mathsf{append}(\mathsf{e}, a)$, which is history-same as a by the left unit law for append. Category hom-carrier transport across this target endpoint equality gives the unprefixed target form. The converse uses the same endpoint equality in the opposite direction and packages the empty target prefix carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty target prefix iff`

Theorem 63.11 (NatTrans prefix component empty-prefix exactness)**.** *For histories a, η, a prefix component with both prefixes empty is exactly a unary object together with a category hom-carrier from a to itself:*

$$\mathsf{NatTransPrefixComponentCarrier}(\mathsf{e}, \mathsf{e}, a, \eta) \iff \mathsf{UnaryHistory}(a) \wedge \mathsf{CategoryHomCarrier}(a, a, \eta).$$

Proof. Unfold the prefix component carrier. Both endpoints are $\mathsf{append}(\mathsf{e}, a)$, which is history-same as a by the left unit law for append. Category hom-carrier transport across those endpoint equalities gives the displayed carrier from a to itself. The converse transports this carrier back to the two empty-prefixed endpoints and packages the two empty prefix carriers. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty prefixes iff`

Theorem 63.12 (NatTrans prefix component zero-headed component impossibility)**.** *For histories p, q, a, η, a prefix component carrier has no zero-headed source prefix, target prefix, object, or component morphism:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \Longrightarrow (\exists z,\ p = \mathsf{E0}(z)) \vee (\exists z,\ q = \mathsf{E0}(z)) \vee (\exists z,\ a = \mathsf{E0}(z)) \vee (\exists z,\ \eta = \mathsf{E0}(z)$$

Proof. Unfold the prefix component carrier. The first three fields give unary-history witnesses for p, q, and a, and the underlying category hom-carrier gives the unary-history witness for the component morphism η. A zero-headed component in any of those positions contradicts unary zero-extension separation. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier zero headed component absurd`

Theorem 63.13 (NatTrans prefix component vertical composition closure)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$, θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, and $\mathsf{Cont}(\eta, \theta, \theta\eta)$, then the composite component is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$.*

Proof. Unpack the two prefix-component carrier witnesses. The outer prefix and object unary witnesses come from the first component and the target prefix witness comes from the second component. Category hom-carrier composition closure composes the two component morphisms across the shared middle object $\mathsf{append}(q, a)$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp closed`

Theorem 63.14 (NatTrans prefix component vertical composition hsame transport)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$, θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, and $\mathsf{Cont}(\eta, \theta, \theta\eta)$, then any* hsame*-equal source prefix, target prefix, object, and composite history display the same vertical composite as a component from $\mathsf{append}(p', a')$ to $\mathsf{append}(r', a')$.*

Proof. Vertical composition closure gives the composite component at the original source prefix, target prefix, object, and composite history. The supplied hsame witnesses transport those concrete component fields to the displayed source prefix, target prefix, object, and composite history. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp hsame transport`

Theorem 63.15 (NatTrans prefix component vertical composition public readback)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$, θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, and $\mathsf{Cont}(\eta, \theta, \theta\eta)$, then the composite is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$, and every displayed component over the same source prefix, target prefix, and object is* hsame *to that composite.*

Proof. The vertical composition closure theorem supplies the displayed composite component carrier. Any displayed component with the same prefixed source and target unpacks to a category hom-carrier with the same endpoints. Category morphism determinacy compares the underlying hom-carriers, yielding the public hsame readback for the composite. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp public readback`

Theorem 63.16 (NatTrans prefix component vertical composite zero-headed component impossibility)**.** *For histories $p, q, r, a, \eta, \theta, composite$, if η is a prefix component from p to q, θ is a prefix component from q to r, and $\mathsf{Cont}(\eta, \theta, composite)$ holds, then no source prefix, middle prefix, target prefix, object, factor component, or composite component is zero-headed.*

Proof. Vertical composition closure gives the composite component carrier from p to r. The zero-headed source, middle, object, and left component cases reduce to the left component exclusion; the target and right component cases reduce to the right component exclusion; the composite component case reduces to the closed composite component carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp zero headed component absurd`

Theorem 63.17 (NatTrans prefix component vertical composition family public readback)**.** *For histories $p, q, r, \eta, \theta, composite, displayed$, suppose every unary object a carries η as a component from prefix p to prefix q, every unary object carries θ from q to r, $\mathsf{Cont}(\eta, \theta, composite)$ holds, and every unary object carries displayed from p to r. Then*

$$\mathsf{hsame}(composite, displayed).$$

Proof. Evaluate the source, target, and displayed component families at the empty unary object. The fixed-object vertical public readback theorem compares the closed composite with the displayed component at that object. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp family public readback`

Theorem 63.18 (NatTrans prefix component empty vertical composite exactness)**.** *For prefix components over a common object a, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$ and θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, then the empty vertical composite is equivalent to both component histories being empty and the adjacent prefixes agreeing:*

$$\mathsf{Cont}(\eta, \theta, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, r).$$

Proof. Unfold the two prefix components to their category hom-carriers over the shared object suffix a. Category empty-composite exactness gives the two empty component histories and the two endpoint equalities. Right cancellation removes the common suffix a from both endpoint equalities. Conversely, the prefix equalities rebuild the component endpoint equalities, and category empty-composite exactness supplies the empty vertical continuation ledger. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty iff`

Theorem 63.19 (NatTrans prefix component vertical empty-component readback)**.** *For prefix components over a common object a, if η is a component from p to q, θ is a component from q to r, and their vertical continuation has empty result, then both displayed components have empty component carriers and the adjacent prefixes agree:*

$$\mathsf{Cont}(\eta, \theta, \mathsf{e}) \Longrightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, r, a, \mathsf{e}) \wedge \mathsf{hsame}(p, q)$$

Proof. Apply category empty-composite exactness to the two component hom-carriers. The empty result forces both component histories to be empty and gives endpoint equalities with the shared object suffix a. Transporting the original component carriers along the empty component equalities gives the two empty component carriers, and right cancellation removes the common suffix from the endpoint equalities. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty component readback`

Theorem 63.20 (NatTrans prefix component vertical composite result empty exactness)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, and their vertical composite is composite, then composite is empty exactly when the two component histories are empty and the adjacent prefixes agree:*

$$\mathsf{hsame}(\mathit{composite}, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, r).$$

Proof. Apply category hom-carrier composite-result empty exactness to the two component hom carriers. The endpoint equalities contain the shared object suffix a, and right cancellation exposes $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$. Conversely, the prefix equalities rebuild those endpoint equalities and the given continuation determines the empty composite result. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp result empty iff`

Theorem 63.21 (NatTrans prefix component vertical composite nonempty target visibility)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, and their vertical composite displays the visible result $\mathsf{E1}(k)$, then the target component endpoint is visible:*

$$\exists s.\ \mathsf{append}(r, a) = \mathsf{E1}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{Cont}(\mathsf{append}(p, a), \mathsf{E1}(k), \mathsf{E1}(s)).$$

Proof. Unpack both natural-transformation components to their underlying category hom-carriers. Category source-target case analysis for the nonempty composite result gives either an empty source endpoint with a displayed target, or a visible source endpoint with the same displayed target. In the visible-source branch, source endpoint transport converts the continuation ledger back to $\mathsf{append}(p, a)$; in both branches the result equality identifies the visible composite payload with k. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp nonempty result target visible`

Theorem 63.22 (NatTrans one-headed prefix empty vertical composite exactness)**.** *For prefix components over a common object a, if η is a component from $\mathsf{E1}(p)$ to q and θ is a component from q to $\mathsf{E1}(r)$, then the empty vertical composite is equivalent to both component histories being empty and the one-headed source and target prefix payloads agreeing:*

$$\mathsf{Cont}(\eta, \theta, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}) \wedge \mathsf{hsame}(p, r).$$

Proof. Unfold the two prefix components to their category hom-carriers over $\mathsf{append}(\mathsf{E1}(p), a)$, $\mathsf{append}(q, a)$, and $\mathsf{append}(\mathsf{E1}(r), a)$. Category empty-composite exactness gives the two empty component histories and adjacent endpoint equalities. Transitivity identifies the two endpoints with the common suffix a; right cancellation removes that suffix, and one-headed constructor inversion yields $\mathsf{hsame}(p, r)$. Conversely, empty component histories reduce the vertical continuation ledger to the empty continuation at e. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier e1 prefix empty vert comp iff`

Theorem 63.23 (NatTrans prefix component vertical composition right factor)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$, $\mathsf{Cont}(\eta, \theta, \theta\eta)$ composes η with θ, and the composite is displayed as a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$, then θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$.*

Proof. Unpack the left component and the displayed composite to their underlying category hom-carriers. The category right-factor theorem recovers the missing hom-carrier for θ between the middle endpoint $\mathsf{append}(q, a)$ and the displayed target endpoint $\mathsf{append}(r, a)$. The prefix carrier for q, the prefix carrier for r, and the object carrier for a reassemble the NatTrans prefix component carrier for θ. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp right factor`

Theorem 63.24 (NatTrans prefix component vertical composition right factor public readback)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$, $\mathsf{Cont}(\eta, \theta, \theta\eta)$ composes η with θ, and the composite is displayed as a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$, then θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$. Moreover, any competing right component θ' with $\mathsf{Cont}(\eta, \theta', \theta\eta)$ and the same component endpoints satisfies $\mathsf{hsame}(\theta, \theta')$.*

Proof. Vertical right-factor recovery supplies the displayed component carrier for θ. For any competing θ', the two continuation ledgers have the same left component and composite result, so finite-kernel left cancellation gives $\mathsf{hsame}(\theta, \theta')$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp right factor public readback`

Theorem 63.25 (NatTrans prefix component vertical composition left factor)**.** *For prefix components over unary histories, if θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, $\mathsf{Cont}(\eta, \theta, \theta\eta)$ composes η with θ, and the composite is displayed as a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$, then η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$.*

Proof. Unpack the right component and the displayed composite to their underlying category hom-carriers. The category left-factor theorem recovers the hom-carrier for η between the displayed source endpoint $\mathsf{append}(p, a)$ and the middle endpoint $\mathsf{append}(q, a)$. The prefix carrier for p, the prefix carrier for q, and the object carrier for a reassemble the NatTrans prefix component carrier for η. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp left factor`

Theorem 63.26 (NatTrans prefix component vertical composition left factor public readback)**.** *For prefix components over unary histories, if θ is a component from $\mathsf{append}(q, a)$ to $\mathsf{append}(r, a)$, $\mathsf{Cont}(\eta, \theta, \theta\eta)$ composes η with θ, and the composite is displayed as a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$, then η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$. Moreover, any competing left component η' with $\mathsf{Cont}(\eta', \theta, \theta\eta)$ and the same component endpoints satisfies $\mathsf{hsame}(\eta, \eta')$.*

Proof. Vertical left-factor recovery supplies the displayed component carrier for η. For any competing η', the two continuation ledgers have the same right component and composite result, so finite-kernel right cancellation gives $\mathsf{hsame}(\eta, \eta')$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp left factor public readback`

Theorem 63.27 (NatTrans prefix component vertical composition right identity closure)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$ and $\mathsf{Cont}(\eta, \mathsf{e}, \eta')$, then η' is again such a component and $\mathsf{hsame}(\eta', \eta)$.*

Proof. The BEDC continuation right-unit witness $\mathsf{Cont}(\eta, \mathsf{e}, \eta)$ and the displayed continuation $\mathsf{Cont}(\eta, \mathsf{e}, \eta')$ have the same source and tail. Continuation determinacy gives $\mathsf{hsame}(\eta', \eta)$. Transport the unpacked prefix-component carrier for η along this history sameness to obtain the component carrier for η'. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp right identity closed`

Theorem 63.28 (NatTrans prefix component empty object exactness)**.** *For histories p, q, η, a prefix component over the empty object is exactly unary source and target prefixes together with the unprefixed category hom-carrier from p to q:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, \mathsf{e}, \eta) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{CategoryHomCarrier}(p, q, \eta).$$

Proof. Unfold the prefix component carrier at the empty object. The two endpoints are $\mathsf{append}(p, \mathsf{e})$ and $\mathsf{append}(q, \mathsf{e})$, which reduce to p and q by the right unit law for append. Category hom-carrier transport across those endpoint equalities gives the unprefixed carrier, and the converse transports it back while packaging the empty object carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty object iff`

Theorem 63.29 (NatTrans prefix component right identity empty result)**.** *For a prefix component η from p to q at a, if right is obtained by composing η with the empty right identity, then right is empty exactly when η is empty and the two prefixes agree:*

$$\mathsf{hsame}(right, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(p, q).$$

Proof. Right-identity closure identifies *right* with η and transports the prefix-component carrier to *right*. If *right* is empty, this transported carrier becomes an empty prefix component, so the empty-identity exactness theorem gives $\mathsf{hsame}(p, q)$; the same right-identity comparison gives $\mathsf{hsame}(\eta, \mathsf{e})$.

Conversely, if η is empty, the displayed continuation has empty source and empty right tail. Continuation determinacy compares it with the BEDC right-unit continuation at e, identifying *right* with e. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp right identity empty result iff`

Theorem 63.30 (NatTrans prefix component left identity empty result)**.** *For a prefix component η from p to q at a, if $left$ is obtained by composing the empty left identity before η, then $left$ is empty exactly when η is empty and the two prefixes agree:*

$$\mathsf{hsame}(left, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(p, q).$$

Proof. Left-identity closure identifies $left$ with η and transports the prefix-component carrier to $left$. If $left$ is empty, this transported carrier becomes an empty prefix component, so the empty-identity exactness theorem gives $\mathsf{hsame}(p, q)$; the same left-identity comparison gives $\mathsf{hsame}(\eta, \mathsf{e})$.

Conversely, if η is empty, the displayed continuation has empty left tail and empty source component. The BEDC left-unit result identifies $left$ with e. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp left identity empty result iff`

Theorem 63.31 (NatTrans prefix component identity square empty result exactness)**.** *For a prefix component η from p to q at a, if $left$ is obtained by composing the empty left identity before η and $right$ is obtained by composing η with the empty right identity, then both displayed histories are empty exactly when η is empty and the two prefixes agree:*

$$\mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}) \iff \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(p, q).$$

Proof. The forward direction applies the left-identity empty-result exactness theorem to the empty $left$ component. The reverse direction applies the left- and right-identity empty-result exactness theorems to the same component and prefix comparison, producing both empty displayed histories. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp identity square empty result iff`

Theorem 63.32 (NatTrans prefix component empty component prefix exactness)**.** *For a prefix component η from p to q at a, the component history is empty exactly when the two prefixes agree:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \Longrightarrow (\mathsf{hsame}(\eta, \mathsf{e}) \iff \mathsf{hsame}(p, q)).$$

Proof. Project the underlying category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$. The category empty-morphism endpoint theorem turns $\eta\mathsf{hsamee}$ into equality of those endpoints, and right cancellation removes the shared object suffix a. Conversely, prefix equality lifts through the common object suffix and the category endpoint theorem identifies η with e. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier component empty prefix iff`

Theorem 63.33 (NatTrans prefix component vertical composition left identity closure)**.** *For prefix components over unary histories, if η is a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$ and $\mathsf{Cont}(\mathsf{e}, \eta, \eta')$, then η' is again such a component and $\mathsf{hsame}(\eta', \eta)$.*

Proof. The BEDC left-unit law for continuation gives $\mathsf{hsame}(\eta', \eta)$ directly from $\mathsf{Cont}(\mathsf{e}, \eta, \eta')$. Transport the unpacked prefix-component carrier for η along this history sameness to obtain the component carrier for η'. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp left identity closed`

Theorem 63.34 (NatTrans prefix component vertical composition identity square closure)**.** *For a prefix component η from p to q at a, if $left$ is obtained by composing e before η and $right$ is obtained by composing e after η, then both displayed histories are prefix components from p to q at a, and $\mathsf{hsame}(left, right)$.*

Proof. Apply the left-identity vertical composition closure to $left$ and the right-identity vertical composition closure to $right$. The two closure theorems identify the displayed histories with η, so transitivity and symmetry give $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp identity square closed`

Theorem 63.35 (NatTrans prefix component vertical composition associativity closure)**.** *For prefix components over unary histories, let η, θ, and ι be consecutive components from p to q, q to r, and r to s at the object a. If $\eta\theta$, $\theta\iota$, $(\eta\theta)\iota$, and $\eta(\theta\iota)$ are supplied by the BEDC continuation relation, then the two bracketed composites are components from p to s at a, and they are hsame.*

Proof. Unpack the three prefix-component carrier witnesses to their underlying category hom-carriers between $\mathsf{append}(p, a)$, $\mathsf{append}(q, a)$, $\mathsf{append}(r, a)$, and $\mathsf{append}(s, a)$. Category hom-carrier associativity closure gives hom-carrier witnesses for both bracketed composites and identifies them by hsame. Repackage the shared source prefix, target prefix, and object unary witnesses around those two underlying hom-carriers. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp assoc closed`

Theorem 63.36 (NatTrans prefix component displayed vertical associativity determinacy)**.** *For prefix components over unary histories, let η, θ, and ι be consecutive components from p to q, q to r, and r to s at the object a. If the two bracketed vertical composites are supplied by the BEDC continuation relation and a displayed component carrier from p to s at a is also given, then both bracketed composite histories are* hsame *to the displayed component history.*

Proof. Vertical associativity closure constructs both prefix-component carriers from p to s at a. Unpacking these carriers and the displayed carrier exposes category hom-carriers with the same source and target, so category morphism determinacy identifies each bracketed composite history with the displayed history. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp assoc displayed deterministic`

Theorem 63.37 (NatTrans prefix component target-prefix determinacy)**.** *For prefix components over unary histories, if a fixed component history η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$ *and also from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(r, a)$*, then* $\mathsf{hsame}(q, r)$*.*

Proof. Unpack both prefix-component carrier witnesses to their underlying category hom-carrier witnesses. Category target determinacy identifies $\mathsf{append}(q, a)$ and $\mathsf{append}(r, a)$, and cancellation of the shared suffix a identifies the prefixes q and r. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier target prefix deterministic`

Theorem 63.38 (NatTrans vertical composite target-prefix determinacy)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q, a)$ *to* $\mathsf{append}(r, a)$*,* $\mathsf{Cont}(\eta, \theta, \eta\theta)$*, and the composite history $\eta\theta$ is also presented as a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(s, a)$*, then* $\mathsf{hsame}(r, s)$*.*

Proof. Vertical composition closure first turns the two displayed component witnesses and the continuation witness into a component from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$. Target-prefix determinacy then compares this component with the displayed component ending at $\mathsf{append}(s, a)$ and cancels the shared suffix a. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp target prefix deterministic`

Theorem 63.39 (NatTrans vertical composite middle-prefix determinacy)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q', a)$ *to* $\mathsf{append}(r, a)$*,* $\mathsf{Cont}(\eta, \theta, \eta\theta)$*, and the composite history $\eta\theta$ is displayed as a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(r, a)$*, then* $\mathsf{hsame}(q, q')$*.*

Proof. Expose the category hom-carrier witnesses inside the two component carriers and inside the displayed composite component. Category middle-object determinacy identifies the prefixed middle endpoints $\mathsf{append}(q, a)$ and $\mathsf{append}(q', a)$. Cancellation of the shared suffix a gives $\mathsf{hsame}(q, q')$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp middle prefix deterministic`

Theorem 63.40 (NatTrans prefix component vertical composite source-prefix determinacy)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q, a)$ *to* $\mathsf{append}(r, a)$*,* $\mathsf{Cont}(\eta, \theta, \eta\theta)$*, and the same composite history $\eta\theta$ is displayed as a component from* $\mathsf{append}(s, a)$ *to* $\mathsf{append}(r, a)$*, then* $\mathsf{hsame}(p, s)$*.*

Proof. First close the vertical composite of η and θ, obtaining a component carrier from prefix p to prefix r over the object a. Apply source-prefix determinacy to this composed carrier and the displayed carrier with source prefix s, both targeting prefix r. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp source prefix deterministic`

Theorem 63.41 (NatTrans prefix component source-prefix determinacy)**.** *For prefix components over unary histories, if a fixed component history η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$ *and also from* $\mathsf{append}(r, a)$ *to* $\mathsf{append}(q, a)$*, then* $\mathsf{hsame}(p, r)$*.*

Proof. Unpack both prefix-component carrier witnesses to their underlying category hom-carrier witnesses. Category source determinacy identifies $\mathsf{append}(p, a)$ and $\mathsf{append}(r, a)$, and cancellation of the shared suffix a identifies the prefixes p and r. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier source prefix deterministic`

Theorem 63.42 (NatTrans prefix component tail commutativity)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q, a)$ *to* $\mathsf{append}(r, a)$*,* $\mathsf{Cont}(\eta, \theta, \eta\theta)$*, and* $\mathsf{Cont}(\theta, \eta, \theta\eta)$*, then* $\mathsf{hsame}(\eta\theta, \theta\eta)$*.*

Proof. Each prefix-component carrier exposes an underlying category hom-carrier. The two hom-carriers expose unary membership for η and θ, and unary continuation commutativity identifies the forward and reversed component composites. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier tail comm hsame`

Theorem 63.43 (NatTrans prefix component vertical composite tail comm closure)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q, a)$ *to* $\mathsf{append}(r, a)$*,* $\mathsf{Cont}(\eta, \theta, \eta\theta)$*, and* $\mathsf{Cont}(\theta, \eta, \theta\eta)$*, then $\eta\theta$ is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(r, a)$*, and* $\mathsf{hsame}(\eta\theta, \theta\eta)$*.*

Proof. Vertical composition closure supplies the component carrier for the forward composite. Tail commutativity of the same two component carriers identifies the forward and reversed continuation results. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp tail comm closed`

Theorem 63.44 (NatTrans prefix component vertical composite tail comm public readback)**.** *For prefix components over unary histories, if η is a component from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(q, a)$*, θ is a component from* $\mathsf{append}(q, a)$ *to* $\mathsf{append}(r, a)$*, and both* $\mathsf{Cont}(\eta, \theta, \eta\theta)$ *and* $\mathsf{Cont}(\theta, \eta, \theta\eta)$ *hold, then both displayed composites are components from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(r, a)$*, they are history-same, and any other displayed component with the same endpoints reads back to both.*

Proof. Vertical composition closure gives the forward component. Tail commutativity identifies the forward and reversed continuation results, and hsame transport moves the forward component carrier across that identification to the reversed component. The public readback row for vertical composition identifies the forward composite with any displayed component at the same endpoints; composing this comparison with tail commutativity gives the reversed readback. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp tail comm public readback`

Theorem 63.45 (NatTrans prefix component vertical cycle empty components)**.** *For prefix components over a fixed object a, if components run from p to q and from q back to p, then the empty component is displayed in both directions and* $\mathsf{hsame}(p, q)$.

Proof. Unpack both component carriers to their category hom-carrier fields and apply the two-way endpoint cycle theorem. Repack the empty hom-carriers with the original unary witnesses, then cancel the common right suffix a. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert cycle empty components`

Theorem 63.46 (NatTrans prefix component vertical triangle empty components)**.** *For prefix components over a fixed object a, if components run from p to q, from q to r, and from r back to p, then the empty component is displayed on all three sides and the adjacent prefixes collapse.*

Proof. Unpack the three component carriers to their category hom-carrier fields and apply triangular empty hom exactness at the appended endpoints. Repack the empty hom-carriers with the original unary witnesses, then cancel the common suffix. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert triangle cycle empty components`

Theorem 63.47 (Prefix identity naturality square)**.** *For the unary-continuation category, let p be unary and let f be a hom-carrier from a to b. If the two square composites are obtained by composing f with the empty continuation on the left and on the right, then both composites are hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$*, and the two composites are history-same.*

Proof. The left empty continuation identifies the left composite with f, and the right empty continuation identifies the right composite with f. The given hom-carrier therefore supplies both unprefixed sides of the square. Prefix hom-carrier preservation sends both sides across p, and transitivity of history sameness identifies the composites. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixIdentity naturality square`

Theorem 63.48 (Prefix identity component square closure)**.** *For the unary-continuation category, if p and a are unary,* $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, id)$*,* $\mathsf{Cont}(id, \mathsf{e}, left)$*, and* $\mathsf{Cont}(\mathsf{e}, id, right)$*, then $left$ and $right$ are hom-carriers from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, a)$*, and* $\mathsf{hsame}(left, right)$.

Proof. The prefix identity hom-carrier theorem gives the carrier witness for id at the prefixed object. The right unit continuation identifies $left$ with id, and the left unit continuation identifies $right$ with id. Transporting the identity carrier witness across these equalities gives both legs of the component square, and transitivity of hsame closes the square. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixIdentity identity component square closed`

Theorem 63.49 (Prefix identity component square public readback)**.** *For unary p, a, if id is an empty identity continuation, $left$ and $right$ are its two unit composites, and $displayed$ is a displayed prefix component from p to itself at a, then* $\mathsf{hsame}(left, displayed) \wedge \mathsf{hsame}(right, displayed)$.

Proof. The identity component square closure supplies hom-carriers for the two legs. The displayed component exposes the same hom-carrier endpoints, so hom-carrier morphism determinacy compares both legs with the displayed component. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixIdentity identity component square public readback`

Theorem 63.50 (NatTrans prefix component empty component semantic name certificate)**.** *For fixed histories p, q, a, any empty component witness* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$ *licenses the component predicate* $\eta \mapsto \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *as a semantic naming certificate with source, pattern, and ledger equal to that predicate and classifier* hsame*.*

Proof. The supplied empty component is the carrier witness. The classifier rows are the reflexive, symmetric, and transitive laws for hsame. Carrier transport keeps the fixed prefix and object witnesses and transports the underlying category hom-carrier along the displayed component-history sameness. Since source, pattern, and ledger are the same predicate, the soundness rows are direct projections. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty component semanticNameCert`

Theorem 63.51 (NatTrans prefix component identity semantic name certificate)**.** *For unary histories p, a, the identity prefix-component predicate* $\eta \mapsto \mathsf{NatTransPrefixComponentCarrier}(p, p, a, \eta)$ *carries a semantic naming certificate whose source, pattern, and ledger predicates are this component predicate and whose classifier is* hsame*.*

Proof. The carrier witness is the empty component history. Unary append closure gives the object carrier for $\mathsf{append}(p, a)$, and the category empty identity hom-carrier supplies the component witness at that object. The classifier laws are the reflexive, symmetric, and transitive laws for hsame, and component carrier transport follows by history-sameness transport. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier identity semanticNameCert`

Theorem 63.52 (NatTrans prefix component empty component opposite closure)**.** *For histories $p, q, a, \eta : \mathsf{BHist}$, an empty prefix component from p to q over a also carries the component with the prefixes reversed:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \wedge \eta \mathsf{hsame} \mathsf{e} \implies \mathsf{NatTransPrefixComponentCarrier}(q, p, a, \eta).$$

Proof. Transport η to e and apply prefix-component empty-identity exactness. The forward component exposes unary carriers for p, q, and a, together with $p \mathsf{hsame} q$. Symmetry of hsame gives $q \mathsf{hsame} p$, and the same empty-identity exactness theorem rebuilds the reversed prefix component. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier empty component opposite closed`

Theorem 63.53 (NatTrans prefix component classifier empty component opposite closure)**.** *For histories $p, q, a, \eta, \theta : \mathsf{BHist}$, if a prefix component classifier relates two empty displayed components from p to q over a, then it also relates them after reversing the endpoint prefixes:*

$$\mathsf{NatTransPrefixComponentClassifier}(p, q, a, \eta, \theta) \wedge \eta \mathsf{hsame} \mathsf{e} \wedge \theta \mathsf{hsame} \mathsf{e} \implies \mathsf{NatTransPrefixComponentClassifier}(q, p,$$

Proof. Project the classifier into its two component-carrier rows. Empty-component opposite closure reverses the endpoint prefixes for η and for θ separately. The classifier relation is then rebuilt from the reversed carrier rows and the original component equality $\eta \mathsf{hsame} \theta$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier empty component opposite closed`

Theorem 63.54 (NatTrans vertical empty result opposite closure)**.** *For histories p, q, r, a, η, θ : BHist, if components $p \to q$ and $q \to r$ over a vertically compose with empty result, then the empty component also carries the opposite endpoint direction $r \to p$:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, r, a, \theta) \wedge \mathsf{Cont}(\eta, \theta, \mathsf{e}) \implies \mathsf{NatTransPr}$$

Proof. First close the vertical composite, obtaining the empty component from p to r over a. Apply empty-component opposite closure to this component with reflexive $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$, which reverses the endpoint prefixes. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty result opposite closed`

Theorem 63.55 (NatTrans vertical empty result componentwise opposite closure)**.** *For histories p, q, r, a, η, θ : BHist, if components $p \to q$ and $q \to r$ over a vertically compose with empty result, then each displayed component also carries its opposite endpoint direction:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, r, a, \theta) \wedge \mathsf{Cont}(\eta, \theta, \mathsf{e}) \implies \mathsf{NatTransPr}$$

Proof. Apply vertical empty-result exactness to the two component carriers. It identifies both component histories with e. Empty-component opposite closure then reverses the endpoints of the first component and of the second component separately, preserving the displayed component histories. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty result components opposite closed`

Theorem 63.56 (NatTrans vertical empty result classifier componentwise opposite closure)**.** *For histories $p, q, r, a, \eta, \eta', \theta, \theta'$: BHist, if two prefix-component classifiers form a vertical pair and both displayed component rows compose to the empty history, then each classifier also closes under the componentwise opposite endpoint direction:*

$$\mathsf{NatTransPrefixComponentClassifier}(p, q, a, \eta, \eta') \wedge \mathsf{NatTransPrefixComponentClassifier}(q, r, a, \theta, \theta') \wedge \mathsf{Cont}(\eta, \theta, \mathsf{e}) \wedge \mathsf{Cont}$$

Proof. Apply classifier empty-result component readback to the two vertical continuations. It identifies all four displayed component histories with e. The classifier empty-component opposite closure then reverses the first classifier from $p \to q$ to $q \to p$ and the second from $q \to r$ to $r \to q$, preserving the original classifier comparisons inside each row. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp empty result components opposite closed`

Theorem 63.57 (NatTrans vertical empty result prefix readback)**.** *For histories p, q, r, a, η, θ : BHist, if components $p \to q$ and $q \to r$ over a vertically compose with empty result, then the adjacent prefixes agree:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, r, a, \theta) \wedge \mathsf{Cont}(\eta, \theta, \mathsf{e}) \implies \mathsf{hsame}(p, q)$$

Proof. Apply category empty-composite exactness to the two underlying component hom-carriers. It identifies the adjacent prefixed endpoints with the shared object suffix a. Right cancellation removes that common suffix and yields $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty result prefixes hsame`

Theorem 63.58 (NatTrans vertical empty result endpoint cycle closure)**.** *For histories p, q, r, a, η, θ : BHist, if components $p \to q$ and $q \to r$ over a vertically compose with empty result, then the empty composite closes both endpoint directions and collapses all three prefixes:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, r, a, \theta) \wedge \mathsf{Cont}(\eta, \theta, \mathsf{e}) \Longrightarrow \mathsf{NatTransPr}$$

Proof. Apply the vertical composition closure of Definition 63.13 to the two input component carriers and the empty continuation ledger. This gives the forward empty component from p to r over a.

Apply Definition 63.54 to the same two component carriers and the same empty continuation ledger. This gives the reversed empty component from r to p over a.

Apply Definition 63.57 to read back $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$. Transitivity of hsame then gives $\mathsf{hsame}(p, r)$, and the five witnesses are packed in the displayed order. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp empty result cycle closed`

63.2 The certificate

Definition 63.59 (NatTrans source specification)**.** *The source is two certified functors sharing the same certified source and target categories.*

Definition 63.60 (NatTrans pattern specification)**.** *The pattern is component-family formation with one codomain morphism per source object.*

Definition 63.61 (NatTrans classifier specification)**.** *The classifier relates natural transformations componentwise by the target category morphism classifier.*

Definition 63.62 (NatTrans prefix component classifier)**.** *For fixed prefixes p, q and a fixed source object history a, the displayed component classifier is*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \theta)$$

when $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *and* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \theta)$ *both hold and* $\eta\mathsf{hsame}\theta$*. Thus a classified component is not only an equality witness between histories; it also carries both displayed endpoints in the target-category hom-carrier interface.*

Lean4 (definition) `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier`

Theorem 63.63 (NatTrans prefix component classifier equivalence fields)**.** *For fixed prefixes p, q and object history a, the relation* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ *has the component fields required by the certificate:*

1. *if* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *holds, then* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta)$*;*
2. *if* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \theta)$ *holds, then* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\theta, \eta)$*;*
3. *if* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \theta)$ *and* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\theta, \iota)$ *hold, then* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \iota)$*;*

4. if $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *and* $\eta\mathsf{hsame}\eta'$ *hold, then* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta')$

Proof. Reflexivity packages the given component carrier twice with the reflexive hsame proof for η. Symmetry swaps the two carried endpoints and uses symmetry of hsame.

For transitivity, unpack the two classifier witnesses. The middle component is carried on both sides over the same fixed prefixes and object, and transitivity of hsame identifies the two outer component histories. Repack the original left carrier, the original right carrier, and the composite hsame witness.

For carrier transport, unfold $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ to the underlying CategoryHomCarrier component from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$. The source and target endpoint fields are unchanged; the displayed component field transports along the supplied $\eta\mathsf{hsame}\eta'$ witness, yielding the carrier for η'. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier equivalence fields`

Theorem 63.64 (NatTrans prefix component classifier history transport)**.** *If the source prefix, target prefix, source object, and both displayed component histories are transported along* hsame*, then the prefix component classifier transports to the displayed endpoints. The transported statement also exposes the two transported category hom-carrier fields.*

Proof. Unpack the classifier into the three unary endpoint carriers, the two append-indexed category hom-carriers, and the displayed component classifier. The source and target append endpoints transport along the prefix and object hsame witnesses. Apply category hom-carrier transport to each displayed component history, then compose the original component hsame witness with the two displayed transport witnesses. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier hsame transport`

Theorem 63.65 (NatTrans prefix component classifier zero-headed component impossibility)**.** *If* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \theta)$ *holds, then none of* p, q, a, η, θ *can be a zero-headed history.*

Proof. Unpack the classifier into the three unary prefix/object fields, the two append-indexed category hom-carrier fields, and the displayed component hsame witness. Zero-headed p, q, or a contradicts the corresponding unary field. Zero-headed η or θ contradicts the unary morphism field inside the corresponding category hom-carrier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier zero headed component absurd`

Theorem 63.66 (NatTrans prefix component classifier semantic name certificate)**.** *If the empty component inhabits* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$*, then the displayed component carrier, used simultaneously as source, pattern, and ledger policy, together with* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ *forms a semantic name certificate.*

Proof. The empty component gives the carrier witness. Reflexivity, symmetry, and transitivity are obtained by unpacking the fixed prefix and object fields and using reflexivity, symmetry, and transitivity of hsame on displayed components. Carrier transport follows from the classifier's second carried CategoryHomCarrier endpoint over $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$. Since source, pattern, and ledger are the same concrete component carrier, the two semantic soundness maps are identity projections. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier semanticNameCert`

Theorem 63.67 (NatTrans prefix component classifier component semantic name certificate)**.** *If* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *is inhabited by a concrete component, then the displayed component carrier, used simultaneously as source, pattern, and ledger policy, together with* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ *forms a semantic name certificate.*

Proof. The supplied component gives the carrier witness. The reflexive, symmetric, and transitive classifier laws are obtained by unpacking the fixed unary prefix and object fields and applying the corresponding hsame law to the component history. Carrier transport reads the target component carrier directly from the classifier witness. The source, pattern, and ledger predicates are the same component carrier, so the semantic soundness maps preserve the witness unchanged. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier component semanticNameCert`

Theorem 63.68 (NatTrans prefix component classifier vertical composition congruence)**.** *For histories* $p, q, r, a, \eta, \eta', \theta, \theta', c, c'$*, if* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta')$*,* $\mathsf{NatTransPrefixComponentClassifier}_{q,r,}$ $\mathsf{Cont}(\eta, \theta, c)$*, and* $\mathsf{Cont}(\eta', \theta', c')$*, then*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,r,a}(c, c').$$

Proof. Unpack the two classifier witnesses. The first gives component carriers for η and η' from prefix p to prefix q at a, together with $\eta\mathsf{hsame}\eta'$. The second gives component carriers for θ and θ' from prefix q to prefix r at a, together with $\theta\mathsf{hsame}\theta'$.

Apply Theorem 63.13 to the carriers for η and θ and the witness $\mathsf{Cont}(\eta, \theta, c)$. This gives a component carrier for c from prefix p to prefix r at a. Apply the same closure theorem to η', θ', and $\mathsf{Cont}(\eta', \theta', c')$ to obtain the matching carrier for c'.

It remains to identify the two composite histories. The finite-kernel continuation hsame-congruence theorem transports the continuation witness for η and θ across $\eta\mathsf{hsame}\eta'$ and $\theta\mathsf{hsame}\theta'$, and continuation result determinacy compares the transported composite with the displayed witness for c'. Hence $c\mathsf{hsame}c'$.

Repack the carrier for c, the carrier for c', and the resulting $c\mathsf{hsame}c'$ witness into $\mathsf{NatTransPrefixComponentClassi}$ □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp congr`

Theorem 63.69 (NatTrans prefix component classifier right-identity closure)**.** *For histories* $p, q, a, \eta, \eta', r, r'$*, suppose* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta')$ *holds and suppose* $\mathsf{Cont}(\eta, \mathsf{e}, r)$ *and* $\mathsf{Cont}(\eta', \mathsf{e}, r')$*. Then*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(r, r') \wedge r\mathsf{hsame}r'.$$

Proof. Right-unit determinacy identifies r with η and r' with η'. Transport the prefix component classifier along these endpoint equalities, with the prefix and object histories fixed. The transported classifier contains the required endpoint comparison $r\mathsf{hsame}r'$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp right identity closed`

Theorem 63.70 (NatTrans prefix component classifier empty composite readback)**.** *For histories* $p, q, r, a, \eta, \eta', \theta, \theta'$*, suppose*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta') \quad \textit{and} \quad \mathsf{NatTransPrefixComponentClassifier}_{q,r,a}(\theta, \theta')$$

and suppose $\mathsf{Cont}(\eta, \theta, \mathsf{e})$*. Then the two displayed leading components classify against the empty component, and the adjacent prefixes are history-same:*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentClassifier}_{q,r,a}(\theta, \mathsf{e}) \wedge p\mathsf{hsame}q \wedge q\mathsf{hsame}r.$$

Proof. Read the two carrier rows out of the classifier witnesses and apply the carrier-level empty-composite readback. This supplies empty component carriers for the two rows and the two prefix equalities. The corresponding empty-composite iff also gives $\eta\mathsf{hsame}\mathsf{e}$ and $\theta\mathsf{hsame}\mathsf{e}$. Repacking the original carried morphism rows, the empty component rows, and these history-same witnesses gives the two classifier rows against e. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp empty result readback`

Theorem 63.71 (NatTrans classifier vertical empty-result component readback)**.** *For histories* $p, q, r, a, \eta, \eta', \theta, \theta', c, c'$*, suppose* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta')$ *and* $\mathsf{NatTransPrefixComponentClassifie}$ *hold. If* $\mathsf{Cont}(\eta, \theta, c)$*,* $\mathsf{Cont}(\eta', \theta', c')$*,* $c\mathsf{hsame}\mathsf{e}$*, and* $c'\mathsf{hsame}\mathsf{e}$*, then the four component histories are empty and the adjacent prefixes agree:*

$$\mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\eta', \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}) \wedge \mathsf{hsame}(\theta', \mathsf{e}) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, r).$$

Proof. Project the carrier fields for $\eta, \eta', \theta, \theta'$ from the two classifier witnesses. Apply vertical composite result-empty exactness to the primary components η, θ and to the secondary components η', θ'. The primary readback supplies emptiness of η and θ together with $p\mathsf{hsame}q$ and $q\mathsf{hsame}r$; the secondary readback supplies emptiness of η' and θ'. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp empty result components`

Theorem 63.72 (NatTrans classifier vertical empty-result cycle closure)**.** *For histories* $p, q, r, a, \eta, \eta', \theta, \theta'$*, suppose* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta')$ *and* $\mathsf{NatTransPrefixComponentClassifier}_{q,r,a}(\theta, \theta')$ *hold. If* $\mathsf{Cont}(\eta, \theta, \mathsf{e})$ *and* $\mathsf{Cont}(\eta', \theta', \mathsf{e})$*, then*

$$\mathsf{NatTransPrefixComponentClassifier}_{p,r,a}(\mathsf{e}, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentClassifier}_{r,p,a}(\mathsf{e}, \mathsf{e}) \\ \wedge\, p\mathsf{hsame}q \wedge q\mathsf{hsame}r \wedge p\mathsf{hsame}r.$$

Proof. Vertical classifier congruence composes the two displayed classifier rows and gives the forward empty classifier from p to r. The empty-result component readback supplies the four empty component witnesses and the adjacent prefix equalities $p\mathsf{hsame}q$ and $q\mathsf{hsame}r$. Since the forward component is empty, the empty-component opposite closure gives the reverse classifier from r to p. Transitivity of hsame gives $p\mathsf{hsame}r$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp empty result cycle closed`

Theorem 63.73 (NatTrans classifier empty-result opposite package)**.** *For histories* $p, q, r, a, \eta, \eta', \theta, \theta', c, c' :$ BHist*, suppose* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}(\eta, \eta')$ *and* $\mathsf{NatTransPrefixComponentClassifier}_{q,r,a}(\theta, \theta')$ *hold. If* $\mathsf{Cont}(\eta, \theta, c)$*,* $\mathsf{Cont}(\eta', \theta', c')$*,* $c\mathsf{hsame}\mathsf{e}$*, and* $c'\mathsf{hsame}\mathsf{e}$*, then both classified components admit the opposite prefix direction and the adjacent prefixes agree:*

$$\mathsf{NatTransPrefixComponentClassifier}_{q,p,a}(\eta, \eta') \wedge \mathsf{NatTransPrefixComponentClassifier}_{r,q,a}(\theta, \theta') \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, r)$$

Proof. Unpack the two classifier witnesses. They provide carrier witnesses for η, η' from prefix p to prefix q over a and for θ, θ' from prefix q to prefix r over a, together with $\eta\mathsf{hsame}\eta'$ and $\theta\mathsf{hsame}\theta'$.

Apply Theorem 63.71 to the two classifiers, the two continuation witnesses, and the two empty result identifications. This identifies all four displayed component histories with e and supplies $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$.

Now apply Theorem 63.52 separately to the carrier witnesses for η, η', θ, and θ'. The four emptiness witnesses reverse the first classified pair from $p \to q$ to $q \to p$ and the second classified pair from $q \to r$ to $r \to q$.

Repack the reversed carrier witnesses with the original $\eta\mathsf{hsame}\eta'$ and $\theta\mathsf{hsame}\theta'$ witnesses. The two rebuilt classifiers, together with the prefix agreements produced by readback, give the claimed package. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier empty result opposite package`

Theorem 63.74 (NatTrans vertical composite zero-headed component impossibility)**.** *For vertically composable prefix component classifiers, none of the three prefixes, the shared source object, the four input component histories, or the two composite component histories can be zero-headed.*

Proof. First form the composite prefix component classifier by vertical composition congruence. Zero-headed source-prefix, middle-prefix, object, or left component cases are rejected by the left classifier. Zero-headed middle-prefix, target-prefix, object, or right component cases are rejected by the right classifier. Zero-headed composite endpoints are rejected by the composite classifier. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier vert comp zero headed component absurd`

Definition 63.75 (NatTrans stability certificate)**.** *The stability certificate consists of:*

1. *componentwise reflexivity*
2. *componentwise symmetry*
3. *componentwise transitivity*
4. *component carrier transport along* hsame *of displayed component histories*
5. *naturality square validity for every source morphism*
6. *stability of naturality under component classifier replacement*
7. *vertical composition closure*
8. *identity natural transformation closure*

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier stability certificate fields`

Definition 63.76 (NatTrans ledger policy)**.** *Record the functor ledgers, component morphism ledgers, and naturality derivations.*

Definition 63.77 ($\mathsf{NatTrans}^{\uparrow}$ certificate obligations)**.** *The carrier, source, pattern, classifier, stability, and ledger data listed above are the obligations required for the interface name* $\mathsf{NatTrans}^{\uparrow}$*.*

Primitive boundary. $\mathsf{NatTrans}^{\uparrow}$ is not a BEDC primitive. It is admitted only when those obligations are supplied.

Remark 63.78 (Zero-axiom encoding note)**.** *Avoids funext by componentwise classifier sameness.*

Theorem 63.79 (NatTrans prefix vertical composite prefix closure)**.** *For unary histories u, p, q, r, a, if η is a prefix component from p to q over a, θ is a prefix component from q to r over a, and $\mathsf{Cont}(\eta, \theta, \mathit{composite})$, then the displayed composite is also a prefix component after prefixing both endpoint prefixes by u:*

$$\mathsf{NatTransPrefixComponentCarrier}(\mathsf{append}(u, p), \mathsf{append}(u, r), a, \mathit{composite}).$$

Proof. The vertical composite first closes as a category hom-carrier from $\mathsf{append}(p, a)$ to $\mathsf{append}(r, a)$. Prefix functoriality by u carries that hom-carrier to endpoints $\mathsf{append}(u, \mathsf{append}(p, a))$ and $\mathsf{append}(u, \mathsf{append}(r, a))$. Associativity of BEDC append transports those endpoints to $\mathsf{append}(\mathsf{append}(u, p), a)$ and $\mathsf{append}(\mathsf{append}(u, r), a)$, which are exactly the component carrier endpoints. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier prefix vert comp closed`

Theorem 63.80 (NatTrans prefix component vertical composite endpoint cycle boundary)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, their vertical composite displays composite, and $\mathsf{hsame}(p, r)$, then the two components and their composite are empty and the intermediate prefixes collapse:*

$$\mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}) \wedge \mathsf{hsame}(\mathit{composite}, \mathsf{e}) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, r).$$

Proof. Unpack the two prefix components to category hom-carriers between $\mathsf{append}(p, a)$, $\mathsf{append}(q, a)$, and $\mathsf{append}(r, a)$. The endpoint equality $\mathsf{hsame}(p, r)$ extends across the shared suffix a. The category endpoint-cycle boundary theorem then gives empty component and composite histories plus endpoint equalities; cancelling the common suffix a gives $\mathsf{hsame}(p, q)$ and $\mathsf{hsame}(q, r)$. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp endpoint cycle boundary`

Theorem 63.81 (NatTrans prefix component vertical composite nonempty source visibility)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, their vertical composite displays $\mathsf{E1}(k)$, and the source endpoint $\mathsf{append}(p, a)$ is not history-same to e, then both component endpoints are visible:*

$$\exists s, t.\ \mathsf{append}(p, a) = \mathsf{E1}(s) \wedge \mathsf{append}(r, a) = \mathsf{E1}(t) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(t) \wedge \mathsf{Cont}(\mathsf{E1}(s), \mathsf{E1}$$

Proof. Unpack both prefix components to their underlying category hom-carriers and apply category source-target case analysis to the displayed nonempty composite result. The empty-source branch contradicts the source nonemptiness assumption. In the visible-source branch, one-constructor inversion on the displayed result identifies the payload with k, leaving exactly the visible source endpoint, visible target endpoint, unary tail witnesses, and the visible continuation ledger. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp nonempty result source visible`

Theorem 63.82 (NatTrans prefix component vertical composite source-target cases)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, and their vertical composite displays* $\mathsf{E1}(k)$*, then either the source endpoint* $\mathsf{append}(p, a)$ *is history-same to* e *and the target endpoint is visible, or both source and target endpoints are visible:*

$$(\mathsf{hsame}(\mathsf{append}(p,a),\mathsf{e}) \wedge \exists t.\ \mathsf{append}(r,a) = \mathsf{E1}(t) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(t) \wedge \mathsf{Cont}(\mathsf{append}(p,a),\mathsf{E1}(k),\mathsf{E}$$
$$\vee\ (\exists s,t.\ \mathsf{append}(p,a) = \mathsf{E1}(s) \wedge \mathsf{append}(r,a) = \mathsf{E1}(t) \wedge \mathsf{UnaryHistory}(s) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{UnaryHistory}(t) \wedge \mathsf{Co}$$

Proof. Unpack both prefix components to category hom-carriers and apply the category-level source-target case analysis for a nonempty composite result. The empty-source branch gives the source endpoint comparison and the visible target payload; the visible-source branch gives the visible source payload, the same visible result payload, the visible target payload, and the displayed continuation ledger. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp nonempty result source target cases`

Theorem 63.83 (NatTrans prefix component nonempty right-identity visible readback)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, their vertical composite displays* $\mathsf{E1}(k)$*, the right-identity continuation displays* $\mathsf{Cont}(\mathsf{E1}(k), \mathsf{e}, right)$*, and the source endpoint* $\mathsf{append}(p, a)$ *is not history-same to* e*, then $right$ is a component from p to r,* $right\mathsf{hsameE1}(k)$*, and the source-visible readback exposes visible source and target endpoints around the same payload k.*

Proof. First close the vertical composite as a prefix component from p to r with displayed history $\mathsf{E1}(k)$. The right-identity closure for prefix components transports this component along the continuation with empty right factor and records $right\mathsf{hsameE1}(k)$. The existing source-visible nonempty-result readback applied to the original vertical composite supplies the visible source endpoint, visible target endpoint, unary payload witnesses, and the visible continuation ledger. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp nonempty result right identity visible readback`

Theorem 63.84 (NatTrans prefix component nonempty left-identity visible readback)**.** *For prefix components over a common object a, if η goes from p to q, θ goes from q to r, their vertical composite displays* $\mathsf{E1}(k)$*, the left-identity continuation displays* $\mathsf{Cont}(\mathsf{e}, \mathsf{E1}(k), leftResult)$*, and the source endpoint* $\mathsf{append}(p, a)$ *is not history-same to* e*, then $leftResult$ is a component from p to r,* $leftResult\mathsf{hsameE1}(k)$*, and the source-visible readback exposes visible source and target endpoints around the same payload k.*

Proof. First close the vertical composite as a prefix component from p to r with displayed history $\mathsf{E1}(k)$. The left-identity closure for prefix components transports this component along the continuation with empty left factor and records $leftResult\mathsf{hsameE1}(k)$. The source-visible nonempty-result readback applied to the original vertical composite supplies the visible source endpoint, visible target endpoint, unary payload witnesses, and the visible continuation ledger. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier vert comp nonempty result left identity visible readback`

Theorem 63.85 (NatTrans vertical composition naturality)**.** *Let F, G, and H be certified* $\mathsf{Functor}^{\uparrow}$ *carriers with the same certified source category and target category. Let $\alpha : F \Rightarrow G$ and $\beta : G \Rightarrow H$ be admitted* $\mathsf{NatTrans}^{\uparrow}$ *carriers. For every source morphism $f : C \to C'$, the vertical component family $(\beta \circ \alpha)_X = \beta_X \circ \alpha_X$ satisfies the target morphism-classifier naturality square*

$$H(f) \circ (\beta \circ \alpha)_C \sim (\beta \circ \alpha)_{C'} \circ F(f).$$

Proof. The shared source and target categories are part of the $\mathsf{NatTrans}^{\uparrow}$ source specification in Definition 63.59. By the functor carrier definition in Definition 62.1, the symbols $F(f)$, $G(f)$, and $H(f)$ denote the corresponding target-category morphism images. By Definition 63.1, each admitted natural transformation is an object-indexed component family with a naturality witness for every source morphism.

The $\mathsf{NatTrans}^{\uparrow}$ stability certificate in Definition 63.75 includes vertical composition closure. Applying that field to α and β gives an admitted $\mathsf{NatTrans}^{\uparrow}$ carrier from F to H whose component at X is $\beta_X \circ \alpha_X$. The same stability certificate lists naturality square validity for every source morphism, so applying that naturality field to this composite carrier and to $f : C \to C'$ gives

$$H(f) \circ (\beta \circ \alpha)_C \sim (\beta \circ \alpha)_{C'} \circ F(f).$$

This is the displayed classifier witness for the composite component family. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentFamilyCarrier vert comp closed`

Definition 63.86 (Opposite natural transformation certificate data)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be* $\mathsf{Cat}^{\uparrow}$ *certificates, let $F, G : \mathcal{C} \to \mathcal{D}$ be* $\mathsf{Functor}^{\uparrow}$ *certificates, and let $\alpha : F \Rightarrow G$ be a* $\mathsf{NatTrans}^{\uparrow}$ *certificate with component family $\alpha_X : F_0(X) \to G_0(X)$ in $\mathcal{D}$. Over the opposite-functor certificates of Definition 62.85, define $\alpha^{\mathrm{op}} : G^{\mathrm{op}} \Rightarrow F^{\mathrm{op}}$ by the same raw component family*

$$(\alpha^{\mathrm{op}})_X := \alpha_X.$$

The component is typed in $\mathcal{D}^{\mathrm{op}}$ because the hom-carrier of Definition 61.327 reads the original morphism $\alpha_X : F_0(X) \to G_0(X)$ as a displayed morphism $G_0(X) \to F_0(X)$. For a displayed morphism $u : X \to Y$ in $\mathcal{C}^{\mathrm{op}}$, the naturality ledger is the classifier-symmetric form of the original naturality ledger for the same raw morphism $u : Y \to X$ in $\mathcal{C}$, with both target composites read through opposite composition.

Theorem 63.87 (Opposite natural transformation certificate)**.** *If $\mathcal{C}$ and $\mathcal{D}$ satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations, $F, G : \mathcal{C} \to \mathcal{D}$ satisfy the* $\mathsf{Functor}^{\uparrow}$ *certificate obligations, and $\alpha : F \Rightarrow G$ satisfies the* $\mathsf{NatTrans}^{\uparrow}$ *certificate obligations of Definition 63.77, then the data of Definition 63.86 satisfy the* $\mathsf{NatTrans}^{\uparrow}$ *certificate obligations from G^{op} to F^{op} over $\mathcal{C}^{\mathrm{op}}$ and $\mathcal{D}^{\mathrm{op}}$.*

Proof. By Definition 61.328, the source and target opposite data are certified categories. By Definition 62.86, G^{op} and F^{op} are certified functors with common source $\mathcal{C}^{\mathrm{op}}$ and common target $\mathcal{D}^{\mathrm{op}}$. This supplies the source specification required by Definition 63.59.

For the carrier field, let X be a source object. The component field of α gives $\alpha_X : F_0(X) \to G_0(X)$ in $\mathcal{D}$. Since the opposite hom-carrier reverses endpoints and leaves the raw morphism classifier unchanged, the same raw morphism is a component $(\alpha^{\mathrm{op}})_X : G_0^{\mathrm{op}}(X) \to F_0^{\mathrm{op}}(X)$ in $\mathcal{D}^{\mathrm{op}}$. Thus the carrier and pattern fields of Definition 63.1 and Definition 63.60 are the original component family with the endpoints read oppositely.

The classifier field is inherited componentwise. The target morphism classifier of $\mathcal{D}^{\mathrm{op}}$ is the target morphism classifier of $\mathcal{D}$ on the same raw component representatives, so componentwise

reflexivity, transitivity, and classifier replacement are exactly the corresponding fields listed in Definition 63.61 and Definition 63.75 for α.

It remains to check naturality. Let $u : X \to Y$ be displayed in $\mathcal{C}^{\mathrm{op}}$. The same raw morphism is $u : Y \to X$ in $\mathcal{C}$. Naturality of α in $\mathcal{D}$ gives the target morphism-classifier comparison

$$G_1(u) \circ_{\mathcal{D}} \alpha_Y \sim_{\mathcal{D}} \alpha_X \circ_{\mathcal{D}} F_1(u).$$

Applying symmetry of the codomain morphism classifier gives

$$\alpha_X \circ_{\mathcal{D}} F_1(u) \sim_{\mathcal{D}} G_1(u) \circ_{\mathcal{D}} \alpha_Y.$$

By Definition 61.327 and Definition 62.85, these two original composites are exactly

$$F_1^{\mathrm{op}}(u) \circ_{\mathcal{D}^{\mathrm{op}}} (\alpha^{\mathrm{op}})_X$$

and

$$(\alpha^{\mathrm{op}})_Y \circ_{\mathcal{D}^{\mathrm{op}}} G_1^{\mathrm{op}}(u),$$

respectively. Since the morphism classifier is unchanged by the opposite category data, this is the required naturality square in $\mathcal{D}^{\mathrm{op}}$.

The remaining stability and ledger clauses are inherited without changing raw data. Identity components and vertical component composites are the same raw component histories, with their endpoints read in the opposite target category; the naturality argument above applies to each displayed source morphism. The ledger policy of Definition 63.76 therefore records the same functor ledgers, component morphism ledgers, and naturality derivations, now indexed by the two opposite-functor certificates. □

Theorem 63.88 (Double opposite natural transformation certificate data identity)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying the category certificate obligations, let $F, G : \mathcal{C} \to \mathcal{D}$ be $\mathsf{Functor}^{\uparrow}$ certificates satisfying the obligations of Definition 62.80, and let $\alpha : F \Rightarrow G$ be a $\mathsf{NatTrans}^{\uparrow}$ certificate satisfying the obligations of Definition 63.77. Write $\alpha_X : F_0(X) \to G_0(X)$ for its component family, and write its naturality row at a displayed morphism $f : X \to Y$ in $\mathcal{C}$ as*

$$G_1(f) \circ_{\mathcal{D}} \alpha_X \sim_{\mathcal{D}} \alpha_Y \circ_{\mathcal{D}} F_1(f).$$

Let

$$\alpha^{\mathrm{opop}} = (\alpha^{\mathrm{op}})^{\mathrm{op}}$$

be obtained by applying Definition 63.86 twice, with the double-opposite source and target categories identified by Definition 61.330 and the double-opposite source and target functors identified by Definition 62.95. Then $\alpha^{\mathrm{opop}} : F^{\mathrm{opop}} \Rightarrow G^{\mathrm{opop}}$ satisfies the $\mathsf{NatTrans}^{\uparrow}$ certificate obligations, and its component family and naturality rows are componentwise identical to those of α:

$$(\alpha^{\mathrm{opop}})_X = \alpha_X,$$

and, for every displayed $f : X \to Y$ in $\mathcal{C}$, the displayed naturality row of α^{opop} is the same target-classifier comparison

$$G_1(f) \circ_{\mathcal{D}} \alpha_X \sim_{\mathcal{D}} \alpha_Y \circ_{\mathcal{D}} F_1(f).$$

The component classifier, stability, and ledger rows that refer only to these component and naturality data are the corresponding rows of α under the same double-opposite identifications.

Proof. Apply Definition 63.87 to α. This supplies a certified natural transformation

$$\alpha^{\mathrm{op}} : G^{\mathrm{op}} \Rightarrow F^{\mathrm{op}}$$

over $\mathcal{C}^{\mathrm{op}}$ and $\mathcal{D}^{\mathrm{op}}$. Applying the same theorem to α^{op} supplies a certified natural transformation

$$\alpha^{\mathrm{opop}} : F^{\mathrm{opop}} \Rightarrow G^{\mathrm{opop}}$$

over $\mathcal{C}^{\mathrm{opop}}$ and $\mathcal{D}^{\mathrm{opop}}$. The source specification is the original one after the identifications supplied by Definition 61.330 and Definition 62.95.

For components, one unfolding of Definition 63.86 gives $(\alpha^{\mathrm{op}})_X = \alpha_X$. A second unfolding gives

$$(\alpha^{\mathrm{opop}})_X = (\alpha^{\mathrm{op}})_X = \alpha_X.$$

The endpoint typing is restored because the double-opposite category data restore the hom-carrier endpoints and the double-opposite functor data restore F_0 and G_0.

For naturality, let $f : X \to Y$ be displayed in $\mathcal{C}$. In the second opposite construction the same raw morphism is read as $f : Y \to X$ in $\mathcal{C}^{\mathrm{op}}$. The naturality row of α^{op} at that opposite morphism is, when read back through the first opposite target category,

$$\alpha_Y \circ_{\mathcal{D}} F_1(f) \sim_{\mathcal{D}} G_1(f) \circ_{\mathcal{D}} \alpha_X,$$

which is the classifier-symmetric form of the original naturality row for α at f. The second opposite construction takes the classifier-symmetric form of this row and reads both target composites through $\mathcal{D}^{\mathrm{opop}}$. By Definition 61.330, the target morphism classifier and composition relation of $\mathcal{D}^{\mathrm{opop}}$ are the original ones of $\mathcal{D}$; by Definition 62.95, the functor morphism maps are again F_1 and G_1. Hence the displayed row is exactly

$$G_1(f) \circ_{\mathcal{D}} \alpha_X \sim_{\mathcal{D}} \alpha_Y \circ_{\mathcal{D}} F_1(f).$$

The classifier specification of Definition 63.61 is componentwise by the target morphism classifier, and that classifier is unchanged after the two opposite readings. The stability fields of Definition 63.75 use the same component transports, naturality rows, identity components, and vertical composites after the endpoint order has been swapped twice. The ledger policy of Definition 63.76 therefore records the original functor ledgers, component ledgers, and naturality derivations under the restored source and target data. □

Definition 63.89 (Identity natural transformation certificate data)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map F_0 and morphism map F_1. The identity natural transformation data $\mathsf{id}_F : F \Rightarrow F$ are the component family*

$$(\mathsf{id}_F)_X := \mathsf{id}_{F_0(X)}.$$

For a displayed morphism $u : X \to Y$ in $\mathcal{C}$, the naturality ledger is the target-category identity square comparing

$$F_1(u) \circ_{\mathcal{D}} \mathsf{id}_{F_0(X)} \quad \textit{and} \quad \mathsf{id}_{F_0(Y)} \circ_{\mathcal{D}} F_1(u).$$

Theorem 63.90 (Identity natural transformation certificate)**.** *Let $\mathcal{C}$ and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203, and let $F : \mathcal{C} \to \mathcal{D}$ be a $\mathsf{Functor}^{\uparrow}$ certificate satisfying Definition 62.80. Then the data of Definition 63.89 satisfy the $\mathsf{NatTrans}^{\uparrow}$ certificate obligations from F to F in the sense of Definition 63.77. Its component at each object X is the target identity,*

$$(\mathsf{id}_F)_X = \mathsf{id}_{F_0(X)},$$

and for every displayed source morphism $u : X \to Y$ its naturality row is

$$F_1(u) \circ_{\mathcal{D}} (\mathsf{id}_F)_X \sim_{\mathcal{D}} (\mathsf{id}_F)_Y \circ_{\mathcal{D}} F_1(u).$$

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentFamilyCarrier identity unit laws`

Proof. The source row of Definition 63.59 is the pair (F, F), and the two entries have the same certified source $\mathcal{C}$ and target $\mathcal{D}$ by the displayed $\mathsf{Functor}^{\uparrow}$ certificate for F. The pattern row of Definition 63.60 is component-family formation over objects of $\mathcal{C}$.

For the carrier row, fix an object X of $\mathcal{C}$. The functor carrier of Definition 62.1 sends X to the target object $F_0(X)$ of $\mathcal{D}$. The identity field of the target category, listed in Definition 61.201, supplies a hom-carrier

$$\mathsf{id}_{F_0(X)} : F_0(X) \to F_0(X).$$

This is exactly the component required by Definition 63.1 for a transformation from F to itself.

It remains to check naturality. Let $u : X \to Y$ be displayed in $\mathcal{C}$. By the morphism part of the functor carrier, $F_1(u)$ is a displayed target morphism $F_0(X) \to F_0(Y)$ in $\mathcal{D}$. The two composites in the identity naturality square are the right and left identity composites of this displayed morphism with the target-category identities at $F_0(X)$ and $F_0(Y)$. The category identity-square closure of Definition 61.95, equivalently the left and right identity rows of Definition 61.201, gives both composites as target hom-carriers from $F_0(X)$ to $F_0(Y)$ and gives their morphism-classifier comparison. Substituting the definition of $(\mathsf{id}_F)_X$ and $(\mathsf{id}_F)_Y$ yields the displayed $\mathsf{NatTrans}^{\uparrow}$ naturality row. In the concrete prefix-component realization, this is the same pointwise calculation recorded in Definition 63.47.

The classifier row of Definition 63.61 is componentwise in the target morphism classifier. Hence component reflexivity, symmetry, transitivity, and transport along classified component histories are inherited from the morphism classifier and endpoint transport fields of $\mathcal{D}$. Stability of the naturality row under component-classifier replacement follows by the same target classifier transitivity after replacing the two identity components by classified components at the same endpoints. The identity-transformation closure row in Definition 63.75 is realized by the family above, and the vertical-composition row is the ambient closure operation on admitted component families.

The ledger row of Definition 63.76 records the functor ledger of F on both sides, the target identity ledger at each object $F_0(X)$, and the identity-square derivation for each displayed source morphism. These rows supply the carrier, source, pattern, classifier, stability, and ledger obligations listed in Definition 63.77. □

Definition 63.91 (Functor-whiskered natural transformation data)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates. Let $F, G : \mathcal{C} \to \mathcal{D}$ be $\mathsf{Functor}^{\uparrow}$ certificates with object maps F_0, G_0 and morphism maps F_1, G_1, and let $\alpha : F \Rightarrow G$ be a $\mathsf{NatTrans}^{\uparrow}$ certificate with component family $\alpha_X : F_0(X) \to G_0(X)$. Let $H : \mathcal{D} \to \mathcal{E}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map H_0 and morphism map H_1. The functor-whiskered data $H_*\alpha : H \circ F \Rightarrow H \circ G$ are the component family*

$$(H_*\alpha)_X := H_1(\alpha_X).$$

For a displayed morphism $u : X \to Y$ in $\mathcal{C}$, the displayed naturality row is the $\mathcal{E}$-morphism-classifier comparison obtained by applying H_1 to the naturality row of α and replacing the images of the two $\mathcal{D}$-composites by the corresponding $\mathcal{E}$-composites through the composition-preservation field of H.

Theorem 63.92 (Functor whiskering of a natural transformation)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203. Let $F, G : \mathcal{C} \to \mathcal{D}$ and $H : \mathcal{D} \to \mathcal{E}$ be $\mathsf{Functor}^{\uparrow}$ certificates*

satisfying Definition 62.80, with object maps F_0, G_0, H_0 and morphism maps F_1, G_1, H_1. If $\alpha : F \Rightarrow G$ satisfies the $\mathsf{NatTrans}^{\uparrow}$ *certificate obligations of Definition 63.77, then the data of Definition 63.91 satisfy the* $\mathsf{NatTrans}^{\uparrow}$ *carrier and naturality-square fields for a natural transformation*

$$H_*\alpha : H \circ F \Rightarrow H \circ G$$

under the morphism classifier of $\mathcal{E}$. Its component at each object X is

$$(H_*\alpha)_X = H_1(\alpha_X),$$

and for every displayed morphism $u : X \to Y$ in $\mathcal{C}$ its naturality row is

$$H_1(G_1(u)) \circ_{\mathcal{E}} H_1(\alpha_X) \sim_{\mathcal{E}} H_1(\alpha_Y) \circ_{\mathcal{E}} H_1(F_1(u)).$$

Proof. Apply Definition 62.99 to the pairs (F, H) and (G, H). Together with the source, pattern, classifier, stability, and ledger rows listed in Definition 62.80, this reads $H \circ F$ and $H \circ G$ as certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{E}$. Their object maps are

$$(H \circ F)_0(X) = H_0(F_0(X)), \qquad (H \circ G)_0(X) = H_0(G_0(X)),$$

and their morphism maps are

$$(H \circ F)_1(u) = H_1(F_1(u)), \qquad (H \circ G)_1(u) = H_1(G_1(u)).$$

Thus the source row required by Definition 63.59 is the pair $(H \circ F, H \circ G)$.

For the carrier row, fix an object X of $\mathcal{C}$. The carrier field of α, as specified in Definition 63.1, gives a displayed $\mathcal{D}$-morphism

$$\alpha_X : F_0(X) \to G_0(X).$$

The morphism-map field of the functor carrier H in Definition 62.1 sends this displayed morphism to a displayed $\mathcal{E}$-morphism

$$H_1(\alpha_X) : H_0(F_0(X)) \to H_0(G_0(X)).$$

Using the composite object maps above, this is exactly a component

$$(H_*\alpha)_X : (H \circ F)_0(X) \to (H \circ G)_0(X)$$

of the component-family pattern from Definition 63.60.

It remains to verify the naturality square. Let $u : X \to Y$ be a displayed morphism in $\mathcal{C}$. Naturality of α gives the $\mathcal{D}$-morphism-classifier comparison

$$G_1(u) \circ_{\mathcal{D}} \alpha_X \sim_{\mathcal{D}} \alpha_Y \circ_{\mathcal{D}} F_1(u).$$

The classifier-replacement stability field of the functor H, listed in Definition 62.78 and used in Definition 62.99, transports this comparison through H_1, giving

$$H_1(G_1(u) \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} H_1(\alpha_Y \circ_{\mathcal{D}} F_1(u)).$$

The composition-preservation field of H gives

$$H_1(G_1(u) \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} H_1(G_1(u)) \circ_{\mathcal{E}} H_1(\alpha_X)$$

and

$$H_1(\alpha_Y \circ_{\mathcal{D}} F_1(u)) \sim_{\mathcal{E}} H_1(\alpha_Y) \circ_{\mathcal{E}} H_1(F_1(u)).$$

Symmetry and transitivity of the $\mathcal{E}$ morphism classifier, recorded in Definition 61.201, compose these three comparisons into

$$H_1(G_1(u)) \circ_{\mathcal{E}} H_1(\alpha_X) \sim_{\mathcal{E}} H_1(\alpha_Y) \circ_{\mathcal{E}} H_1(F_1(u)).$$

Substituting the composite morphism maps and the component definition gives

$$(H \circ G)_1(u) \circ_{\mathcal{E}} (H_*\alpha)_X \sim_{\mathcal{E}} (H_*\alpha)_Y \circ_{\mathcal{E}} (H \circ F)_1(u),$$

which is the required $\mathsf{NatTrans}^{\uparrow}$ naturality field under the morphism classifier of $\mathcal{E}$.

The ledger row of Definition 63.76 records the two composite functor ledgers, the image under H_1 of each component ledger of α, the transported naturality row, and the two composition-preservation rows of H used to identify the displayed source and target composites. □

Definition 63.93 (Prewhiskered natural transformation data)**.** *Let $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates. Let $K : \mathcal{B} \to \mathcal{C}$ be a $\mathsf{Functor}^{\uparrow}$ certificate with object map K_0 and morphism map K_1. Let $F, G : \mathcal{C} \to \mathcal{D}$ be $\mathsf{Functor}^{\uparrow}$ certificates with object maps F_0, G_0 and morphism maps F_1, G_1, and let $\alpha : F \Rightarrow G$ be a $\mathsf{NatTrans}^{\uparrow}$ certificate with component family $\alpha_X : F_0(X) \to G_0(X)$. The prewhiskered data $\alpha * K : F \circ K \Rightarrow G \circ K$ are the component family*

$$(\alpha * K)_X := \alpha_{K_0(X)}.$$

For a displayed morphism $u : X \to Y$ in $\mathcal{B}$, the displayed naturality row is the $\mathcal{D}$-morphism-classifier comparison obtained by applying the naturality row of α to the displayed $\mathcal{C}$-morphism $K_1(u) : K_0(X) \to K_0(Y)$.

Theorem 63.94 (NatTrans prewhiskering by a functor)**.** *Let $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203. Let $K : \mathcal{B} \to \mathcal{C}$ and $F, G : \mathcal{C} \to \mathcal{D}$ be $\mathsf{Functor}^{\uparrow}$ certificates satisfying Definition 62.80, with object maps K_0, F_0, G_0 and morphism maps K_1, F_1, G_1. If $\alpha : F \Rightarrow G$ satisfies the $\mathsf{NatTrans}^{\uparrow}$ certificate obligations of Definition 63.77, then the data of Definition 63.93 satisfy the $\mathsf{NatTrans}^{\uparrow}$ carrier and naturality-square fields for a natural transformation*

$$\alpha * K : F \circ K \Rightarrow G \circ K$$

under the morphism classifier of $\mathcal{D}$. Its component at each object X of $\mathcal{B}$ is

$$(\alpha * K)_X = \alpha_{K_0(X)},$$

and for every displayed morphism $u : X \to Y$ in $\mathcal{B}$ its naturality row is

$$G_1(K_1(u)) \circ_{\mathcal{D}} \alpha_{K_0(X)} \sim_{\mathcal{D}} \alpha_{K_0(Y)} \circ_{\mathcal{D}} F_1(K_1(u)).$$

Proof. Apply Definition 62.99 to the pairs (K, F) and (K, G). Together with the source, pattern, classifier, stability, and ledger rows listed in Definition 62.80, this reads $F \circ K$ and $G \circ K$ as certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{B}$ to $\mathcal{D}$. Their object maps are

$$(F \circ K)_0(X) = F_0(K_0(X)), \qquad (G \circ K)_0(X) = G_0(K_0(X)),$$

and their morphism maps are

$$(F \circ K)_1(u) = F_1(K_1(u)), \qquad (G \circ K)_1(u) = G_1(K_1(u)).$$

Thus the source row required by Definition 63.59 is the pair $(F \circ K, G \circ K)$.

For the carrier row, fix an object X of $\mathcal{B}$. The functor carrier of Definition 62.1 sends X to the object $K_0(X)$ of $\mathcal{C}$. The carrier field of α, specified in Definition 63.1, gives the displayed $\mathcal{D}$-morphism

$$\alpha_{K_0(X)} : F_0(K_0(X)) \to G_0(K_0(X)).$$

Using the composite object maps above, this is exactly a component

$$(\alpha * K)_X : (F \circ K)_0(X) \to (G \circ K)_0(X)$$

of the component-family pattern from Definition 63.60.

It remains to verify naturality. Let $u : X \to Y$ be a displayed morphism in $\mathcal{B}$. The morphism-map field of K sends it to a displayed morphism

$$K_1(u) : K_0(X) \to K_0(Y)$$

in $\mathcal{C}$. Applying the naturality field of α to this displayed $\mathcal{C}$-morphism gives

$$G_1(K_1(u)) \circ_{\mathcal{D}} \alpha_{K_0(X)} \sim_{\mathcal{D}} \alpha_{K_0(Y)} \circ_{\mathcal{D}} F_1(K_1(u)).$$

Substituting the composite morphism maps and the component definition gives

$$(G \circ K)_1(u) \circ_{\mathcal{D}} (\alpha * K)_X \sim_{\mathcal{D}} (\alpha * K)_Y \circ_{\mathcal{D}} (F \circ K)_1(u),$$

which is the required $\mathsf{NatTrans}^{\uparrow}$ naturality field under the morphism classifier of $\mathcal{D}$.

The ledger row of Definition 63.76 records the two composite functor ledgers, the component ledger of α at each object $K_0(X)$, the morphism-image ledger $K_1(u)$, and the naturality derivation of α at that displayed morphism. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier prewhiskering prefix transport`

Definition 63.95 (Horizontal natural transformation data)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates. Let $F, G : \mathcal{C} \to \mathcal{D}$ and $H, K : \mathcal{D} \to \mathcal{E}$ be $\mathsf{Functor}^{\uparrow}$ certificates with object maps F_0, G_0, H_0, K_0 and morphism maps F_1, G_1, H_1, K_1. Let $\alpha : F \Rightarrow G$ and $\beta : H \Rightarrow K$ be $\mathsf{NatTrans}^{\uparrow}$ certificates with component families $\alpha_X : F_0(X) \to G_0(X)$ and $\beta_Y : H_0(Y) \to K_0(Y)$. The horizontal composite data $\beta \diamond \alpha : H \circ F \Rightarrow K \circ G$ have component family*

$$(\beta \diamond \alpha)_X := \beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X).$$

For a displayed morphism $u : X \to Y$ in $\mathcal{C}$, the displayed naturality row is the $\mathcal{E}$-morphism-classifier comparison

$$K_1(G_1(u)) \circ_{\mathcal{E}} (\beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X)) \sim_{\mathcal{E}} (\beta_{G_0(Y)} \circ_{\mathcal{E}} H_1(\alpha_Y)) \circ_{\mathcal{E}} H_1(F_1(u)).$$

Theorem 63.96 (Horizontal composition of natural transformations)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203. Let $F, G : \mathcal{C} \to \mathcal{D}$ and $H, K : \mathcal{D} \to \mathcal{E}$ be $\mathsf{Functor}^{\uparrow}$ certificates satisfying Definition 62.80, with object maps F_0, G_0, H_0, K_0 and morphism maps F_1, G_1, H_1, K_1. If $\alpha : F \Rightarrow G$ and $\beta : H \Rightarrow K$ satisfy the $\mathsf{NatTrans}^{\uparrow}$ certificate obligations of Definition 63.77, then the data of Definition 63.95 satisfy the $\mathsf{NatTrans}^{\uparrow}$ carrier and naturality-square fields for a natural transformation*

$$\beta \diamond \alpha : H \circ F \Rightarrow K \circ G$$

under the morphism classifier of $\mathcal{E}$. Its component at each object X is

$$(\beta \diamond \alpha)_X = \beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X),$$

and for every displayed morphism $u : X \to Y$ in $\mathcal{C}$ its naturality row is

$$K_1(G_1(u)) \circ_{\mathcal{E}} (\beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X)) \sim_{\mathcal{E}} (\beta_{G_0(Y)} \circ_{\mathcal{E}} H_1(\alpha_Y)) \circ_{\mathcal{E}} H_1(F_1(u)).$$

Moreover, for each object X, this component is classified with the alternative component obtained by applying K to α_X after the component of β at $F_0(X)$:

$$(\beta \diamond \alpha)_X \sim_{\mathcal{E}} K_1(\alpha_X) \circ_{\mathcal{E}} \beta_{F_0(X)}.$$

Proof. Apply Definition 62.99 to the pairs (F, H), (G, H), (F, K), and (G, K). Together with the source, pattern, classifier, stability, and ledger rows listed in Definition 62.80, this reads $H \circ F$, $H \circ G$, $K \circ F$, and $K \circ G$ as certified $\mathsf{Functor}^{\uparrow}$ carriers with source and target categories as displayed.

By Definition 63.92 applied to H and α, the component family

$$(H_* \alpha)_X = H_1(\alpha_X)$$

satisfies the $\mathsf{NatTrans}^{\uparrow}$ carrier and naturality-square fields from $H \circ F$ to $H \circ G$. By Definition 63.94 applied to β and G, the component family

$$(\beta * G)_X = \beta_{G_0(X)}$$

satisfies the corresponding fields from $H \circ G$ to $K \circ G$. These two natural transformations share source category $\mathcal{C}$, target category $\mathcal{E}$, and middle functor $H \circ G$.

The vertical-composition row in Definition 63.75, equivalently the construction used in Definition 63.85, therefore gives a $\mathsf{NatTrans}^{\uparrow}$ carrier from $H \circ F$ to $K \circ G$ with component

$$((\beta * G) \circ (H_* \alpha))_X = (\beta * G)_X \circ_{\mathcal{E}} (H_* \alpha)_X = \beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X).$$

This is exactly the component family in Definition 63.95.

For naturality, let $u : X \to Y$ be displayed in $\mathcal{C}$. Applying Definition 63.85 to $H_* \alpha$ and $\beta * G$ gives

$$(K \circ G)_1(u) \circ_{\mathcal{E}} (\beta \diamond \alpha)_X \sim_{\mathcal{E}} (\beta \diamond \alpha)_Y \circ_{\mathcal{E}} (H \circ F)_1(u).$$

Unfolding the composite morphism maps supplied by Definition 62.99 gives $(K \circ G)_1(u) = K_1(G_1(u))$ and $(H \circ F)_1(u) = H_1(F_1(u))$. Substituting the displayed component formula gives the naturality row stated above.

It remains to identify the alternate component display. The carrier field of α, from Definition 63.1, makes $\alpha_X : F_0(X) \to G_0(X)$ a displayed morphism of $\mathcal{D}$. Naturality of β at this displayed morphism gives

$$K_1(\alpha_X) \circ_{\mathcal{E}} \beta_{F_0(X)} \sim_{\mathcal{E}} \beta_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X).$$

Symmetry of the $\mathcal{E}$ morphism classifier, recorded in Definition 61.201, reverses this comparison, and the right-hand side is the horizontal component by definition. □

Theorem 63.97 (NatTrans horizontal identity reductions)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203. Let $F, G : \mathcal{C} \to \mathcal{D}$ and $H, K : \mathcal{D} \to \mathcal{E}$ be certified $\mathsf{Functor}^{\uparrow}$ carriers satisfying Definition 62.80, with object maps F_0, G_0, H_0, K_0 and morphism maps F_1, G_1, H_1, K_1. Let $\alpha : F \Rightarrow G$ and $\beta : H \Rightarrow K$ be certified $\mathsf{NatTrans}^{\uparrow}$ carriers satisfying Definition 63.77. Then for every object X of $\mathcal{C}$ the identity-horizontal composites supplied by Definition 63.96 are componentwise classified by the corresponding one-sided whiskerings:*

$$(\mathsf{id}_H \diamond \alpha)_X \sim_{\mathcal{E}} (H_*\alpha)_X \sim_{\mathcal{E}} H_1(\alpha_X)$$

and

$$(\beta \diamond \mathsf{id}_F)_X \sim_{\mathcal{E}} (\beta * F)_X \sim_{\mathcal{E}} \beta_{F_0(X)}.$$

Equivalently, the left identity comparison has BHist *component readback $H_1(\alpha_X)$ on both sides, and the right identity comparison has* BHist *component readback $\beta_{F_0(X)}$ on both sides.*

Proof. Use Definition 63.90 to form $\mathsf{id}_H : H \Rightarrow H$ and $\mathsf{id}_F : F \Rightarrow F$. Use Definition 63.92 to read $H_*\alpha$ as a certified transformation $H \circ F \Rightarrow H \circ G$, and use Definition 63.94 to read $\beta * F$ as a certified transformation $H \circ F \Rightarrow K \circ F$.

For the left identity law, Definition 63.96 applied to id_H and α gives the component

$$(\mathsf{id}_H \diamond \alpha)_X = (\mathsf{id}_H)_{G_0(X)} \circ_{\mathcal{E}} H_1(\alpha_X).$$

The identity-transformation certificate identifies $(\mathsf{id}_H)_{G_0(X)}$ with $\mathsf{id}_{H_0(G_0(X))}$, while the whiskering certificate identifies $(H_*\alpha)_X$ with $H_1(\alpha_X)$. The left identity row of the target category in Definition 61.201 classifies $\mathsf{id}_{H_0(G_0(X))} \circ_{\mathcal{E}} H_1(\alpha_X)$ with $H_1(\alpha_X)$, and transitivity of the $\mathcal{E}$ classifier gives the displayed comparison.

For the right identity law, Definition 63.96 applied to β and id_F gives

$$(\beta \diamond \mathsf{id}_F)_X = \beta_{F_0(X)} \circ_{\mathcal{E}} H_1((\mathsf{id}_F)_X).$$

The identity-transformation certificate reads $(\mathsf{id}_F)_X$ as $\mathsf{id}_{F_0(X)}$. The identity-preservation row of H in Definition 62.78 classifies $H_1(\mathsf{id}_{F_0(X)})$ with $\mathsf{id}_{H_0(F_0(X))}$, and the right identity row of Definition 61.201 classifies the resulting composite with $\beta_{F_0(X)}$. The prewhiskering certificate identifies $(\beta * F)_X$ with the same component, so target-classifier transitivity gives the second comparison.

The ledger is the union of the identity-transformation ledgers, the whiskering and prewhiskering ledgers, the horizontal-composition ledger, and the target-category identity ledgers used in the two component reductions. □

Capstone.

Theorem 63.98 (Horizontal-vertical interchange for natural transformations)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be $\mathsf{Cat}^{\uparrow}$ certificates satisfying Definition 61.203. Let $F, G, H : \mathcal{C} \to \mathcal{D}$ and $K, L, M : \mathcal{D} \to \mathcal{E}$ be certified $\mathsf{Functor}^{\uparrow}$ carriers satisfying Definition 62.80, with object maps $F_0, G_0, H_0, K_0, L_0, M_0$ and morphism maps $F_1, G_1, H_1, K_1, L_1, M_1$. Let $\alpha : F \Rightarrow G$, $\alpha' : G \Rightarrow H$, $\beta : K \Rightarrow L$, and $\beta' : L \Rightarrow M$ be certified $\mathsf{NatTrans}^{\uparrow}$ carriers satisfying Definition 63.77. Then the two natural transformations*

$$((\beta' \circ \beta) \diamond (\alpha' \circ \alpha)) : K \circ F \Rightarrow M \circ H \qquad \text{and} \qquad ((\beta' \diamond \alpha') \circ (\beta \diamond \alpha)) : K \circ F \Rightarrow M \circ H$$

are componentwise classified together by the morphism classifier of $\mathcal{E}$. For every object X of $\mathcal{C}$,

$$((\beta' \circ \beta) \diamond (\alpha' \circ \alpha))_X \sim_{\mathcal{E}} ((\beta' \diamond \alpha') \circ (\beta \diamond \alpha))_X.$$

Equivalently, after unfolding the component formulas, the classifier comparison at X is

$$(\beta'_{H_0(X)} \circ_{\mathcal{E}} \beta_{H_0(X)}) \circ_{\mathcal{E}} K_1(\alpha'_X \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} (\beta'_{H_0(X)} \circ_{\mathcal{E}} L_1(\alpha'_X)) \circ_{\mathcal{E}} (\beta_{G_0(X)} \circ_{\mathcal{E}} K_1(\alpha_X)).$$

Proof. This result composes the vertical-composition block, the whiskering blocks, and the horizontal-composition block already present in this chapter; it is not a one-theorem restatement. The named blocks provide the certified carriers on both routes, while the comparison below identifies their components in the codomain classifier.

The vertical-composition row of Definition 63.75, in the form used in Definition 63.85, gives

$$(\alpha' \circ \alpha)_X = \alpha'_X \circ_{\mathcal{D}} \alpha_X$$

and

$$(\beta' \circ \beta)_{H_0(X)} = \beta'_{H_0(X)} \circ_{\mathcal{E}} \beta_{H_0(X)}.$$

Unfolding the horizontal-composition component formula of Definition 63.95 for these two vertical composites reads the first route as

$$((\beta' \circ \beta) \diamond (\alpha' \circ \alpha))_X = (\beta'_{H_0(X)} \circ_{\mathcal{E}} \beta_{H_0(X)}) \circ_{\mathcal{E}} K_1(\alpha'_X \circ_{\mathcal{D}} \alpha_X).$$

Applying the same horizontal component formula before vertical composition gives

$$(\beta \diamond \alpha)_X = \beta_{G_0(X)} \circ_{\mathcal{E}} K_1(\alpha_X)$$

and

$$(\beta' \diamond \alpha')_X = \beta'_{H_0(X)} \circ_{\mathcal{E}} L_1(\alpha'_X).$$

The vertical-composition row then reads the second route as

$$((\beta' \diamond \alpha') \circ (\beta \diamond \alpha))_X = (\beta'_{H_0(X)} \circ_{\mathcal{E}} L_1(\alpha'_X)) \circ_{\mathcal{E}} (\beta_{G_0(X)} \circ_{\mathcal{E}} K_1(\alpha_X)).$$

It remains to compare the displayed target morphisms. The composition-preservation row of the functor K, recorded in Definition 62.78 and used by Definition 62.99, gives

$$K_1(\alpha'_X \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} K_1(\alpha'_X) \circ_{\mathcal{E}} K_1(\alpha_X).$$

After this replacement, category associativity in $\mathcal{E}$ classifies the first route with

$$(\beta'_{H_0(X)} \circ_{\mathcal{E}} (\beta_{H_0(X)} \circ_{\mathcal{E}} K_1(\alpha'_X))) \circ_{\mathcal{E}} K_1(\alpha_X).$$

The carrier field of α' makes $\alpha'_X : G_0(X) \to H_0(X)$ a displayed morphism of $\mathcal{D}$. Naturality of $\beta : K \Rightarrow L$ at this component gives

$$\beta_{H_0(X)} \circ_{\mathcal{E}} K_1(\alpha'_X) \sim_{\mathcal{E}} L_1(\alpha'_X) \circ_{\mathcal{E}} \beta_{G_0(X)}.$$

Composition congruence transports this comparison inside the outer composite, and associativity in $\mathcal{E}$ classifies the result with

$$(\beta'_{H_0(X)} \circ_{\mathcal{E}} L_1(\alpha'_X)) \circ_{\mathcal{E}} (\beta_{G_0(X)} \circ_{\mathcal{E}} K_1(\alpha_X)).$$

Symmetry is used where a displayed comparison is oriented opposite the transport direction, and transitivity of the same morphism classifier composes the chain.

Since Definition 63.61 relates natural transformations componentwise by the target category morphism classifier, these component comparisons are exactly the $\mathsf{NatTrans}^{\uparrow}$ classifier comparison between the vertical composite of horizontal composites and the horizontal composite of vertical composites. □

Theorem 63.99 (Functor whiskering preserves vertical composition)**.** *Let $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ be* $\mathsf{Cat}^{\uparrow}$ *certificates satisfying Definition 61.203. Let $F, G, H : \mathcal{C} \to \mathcal{D}$ and $K : \mathcal{D} \to \mathcal{E}$ be* $\mathsf{Functor}^{\uparrow}$ *certificates satisfying Definition 62.80, with object maps F_0, G_0, H_0, K_0 and morphism maps F_1, G_1, H_1, K_1. Let $\alpha : F \Rightarrow G$ and $\beta : G \Rightarrow H$ satisfy the* $\mathsf{NatTrans}^{\uparrow}$ *certificate obligations of Definition 63.77, and let $\beta \circ \alpha$ be the vertical composite carrier supplied by Definition 63.75. Then the two natural transformation carriers*

$$K_*(\beta \circ \alpha) : K \circ F \Rightarrow K \circ H \qquad \textit{and} \qquad (K_*\beta) \circ (K_*\alpha) : K \circ F \Rightarrow K \circ H$$

are classified together componentwise by the $\mathcal{E}$-morphism classifier. For every object X of $\mathcal{C}$,

$$(K_*(\beta \circ \alpha))_X \sim_{\mathcal{E}} ((K_*\beta) \circ (K_*\alpha))_X$$

at the endpoint pair $K_0(F_0(X)), K_0(H_0(X))$. Equivalently, after unfolding the component clauses,

$$K_1(\beta_X \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} K_1(\beta_X) \circ_{\mathcal{E}} K_1(\alpha_X).$$

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentCarrier prefix whiskering vert comp public readback`

Proof. Apply Definition 62.99 to the pairs (F, K), (G, K), and (H, K). This reads $K \circ F$, $K \circ G$, and $K \circ H$ as certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{C}$ to $\mathcal{E}$, with object maps $K_0(F_0(-))$, $K_0(G_0(-))$, and $K_0(H_0(-))$ and morphism maps $K_1(F_1(-))$, $K_1(G_1(-))$, and $K_1(H_1(-))$.

The vertical-composition row of Definition 63.75, in the form used by Definition 63.85, supplies a carrier $\gamma = \beta \circ \alpha : F \Rightarrow H$ with component

$$\gamma_X = \beta_X \circ_{\mathcal{D}} \alpha_X.$$

Applying Definition 63.92 to K and γ gives a carrier $K_*(\beta \circ \alpha) : K \circ F \Rightarrow K \circ H$ whose component at X is

$$(K_*(\beta \circ \alpha))_X = K_1(\gamma_X) = K_1(\beta_X \circ_{\mathcal{D}} \alpha_X).$$

Apply the same whiskering theorem separately to α and β. The resulting carriers have components

$$(K_*\alpha)_X = K_1(\alpha_X), \qquad (K_*\beta)_X = K_1(\beta_X).$$

Since their middle functor is $K \circ G$, vertical composition closure gives a carrier $(K_*\beta) \circ (K_*\alpha) : K \circ F \Rightarrow K \circ H$ with component

$$((K_*\beta) \circ (K_*\alpha))_X = K_1(\beta_X) \circ_{\mathcal{E}} K_1(\alpha_X).$$

Fix an object X of $\mathcal{C}$. By Definition 63.1, $\alpha_X : F_0(X) \to G_0(X)$ and $\beta_X : G_0(X) \to H_0(X)$ are composable displayed morphisms of $\mathcal{D}$. The composition-preservation field of K, listed in Definition 62.78, applies to this component pair and gives

$$K_1(\beta_X \circ_{\mathcal{D}} \alpha_X) \sim_{\mathcal{E}} K_1(\beta_X) \circ_{\mathcal{E}} K_1(\alpha_X).$$

Substituting the two component formulas above gives the displayed comparison between $(K_*(\beta \circ \alpha))_X$ and $((K_*\beta) \circ (K_*\alpha))_X$. The classifier specification of Definition 63.61 compares natural transformations componentwise by the target morphism classifier, so these objectwise comparisons are exactly the componentwise classifier identity of the two carriers. □

Theorem 63.100 (NatTrans prewhiskering preserves vertical composition)**.** *Let $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ be* $\mathsf{Cat}^{\uparrow}$ *certificates satisfying Definition 61.203. Let $L : \mathcal{B} \to \mathcal{C}$ and $F, G, H : \mathcal{C} \to \mathcal{D}$ be* $\mathsf{Functor}^{\uparrow}$ *certificates satisfying Definition 62.80, with object maps L_0, F_0, G_0, H_0 and morphism maps L_1, F_1, G_1, H_1. Let $\alpha : F \Rightarrow G$ and $\beta : G \Rightarrow H$ satisfy the* $\mathsf{NatTrans}^{\uparrow}$ *certificate obligations of Definition 63.77, and let $\beta \circ \alpha$ be the vertical composite carrier supplied by Definition 63.75. Then the two natural transformation carriers*

$$(\beta \circ \alpha) * L : F \circ L \Rightarrow H \circ L \qquad \textit{and} \qquad (\beta * L) \circ (\alpha * L) : F \circ L \Rightarrow H \circ L$$

are classified together by the $\mathsf{NatTrans}^{\uparrow}$ *component classifier. For every object X of $\mathcal{B}$,*

$$((\beta \circ \alpha) * L)_X \sim_{\mathcal{D}} ((\beta * L) \circ (\alpha * L))_X$$

at the endpoint pair $F_0(L_0(X)), H_0(L_0(X))$. Equivalently, after unfolding the component clauses, both sides are read as the target-category component history

$$\beta_{L_0(X)} \circ_{\mathcal{D}} \alpha_{L_0(X)}.$$

Proof. Apply Definition 62.99 to the pairs (L, F), (L, G), and (L, H). This reads $F \circ L$, $G \circ L$, and $H \circ L$ as certified $\mathsf{Functor}^{\uparrow}$ carriers from $\mathcal{B}$ to $\mathcal{D}$, with object maps $F_0(L_0(-))$, $G_0(L_0(-))$, and $H_0(L_0(-))$.

The vertical-composition row of Definition 63.75, in the form used by Definition 63.85, supplies a carrier $\gamma = \beta \circ \alpha : F \Rightarrow H$ with component

$$\gamma_Y = \beta_Y \circ_{\mathcal{D}} \alpha_Y$$

for every object Y of $\mathcal{C}$. Applying Definition 63.94 to γ and L gives a carrier $(\beta \circ \alpha) * L : F \circ L \Rightarrow H \circ L$ whose component at X is

$$((\beta \circ \alpha) * L)_X = \gamma_{L_0(X)} = \beta_{L_0(X)} \circ_{\mathcal{D}} \alpha_{L_0(X)}.$$

Apply the same prewhiskering theorem separately to α and β. Their prewhiskered components are

$$(\alpha * L)_X = \alpha_{L_0(X)}, \qquad (\beta * L)_X = \beta_{L_0(X)}.$$

These two carriers share the middle functor $G \circ L$, so vertical composition closure gives $(\beta * L) \circ (\alpha * L) : F \circ L \Rightarrow H \circ L$ with component

$$((\beta * L) \circ (\alpha * L))_X = \beta_{L_0(X)} \circ_{\mathcal{D}} \alpha_{L_0(X)}.$$

The carrier fields of Definition 63.1 make $\alpha_{L_0(X)}$ and $\beta_{L_0(X)}$ composable displayed morphisms in $\mathcal{D}$, and the category stability field of Definition 61.201 gives reflexivity of the morphism classifier on their displayed composite. The componentwise classifier specification of Definition 63.61 therefore classifies the two prewhiskering routes together at every object X. □

Theorem 63.101 (NatTrans prefix standard bridge)**.** *For the concrete prefix realization, the carrier* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)$ *is exactly the component row of a natural transformation from the prefix functor at p to the prefix functor at q over unary object a. Its source and target are the functor images* $\mathsf{append}(p, a)$ *and* $\mathsf{append}(q, a)$*, and its displayed component is the concrete morphism* $\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(q, a), \eta)$*. The public* $\mathsf{NatTrans}^{\uparrow}$ *classifier is the componentwise* hsame *comparison packaged by* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$*.*

The bridge fields are as follows. The traditional target is ordinary natural transformations between the two fixed-prefix functors. The BEDC source is $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{UnaryHistory}$, *and* Cont. *The carrier is* $\mathsf{NatTransPrefixComponentCarrier}$, *the classifier is* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$, *stability is given by component transport and vertical composition rows, the ledger is the pair of functor ledgers plus the component morphism ledgers, exactness is the displayed carrier/classifier correspondence, the standard bridge is this theorem, and the status is a paper bridge target awaiting Lean formalization.*

Proof. The carrier row is read first. By Definition 63.2, the prefix component carrier is precisely unary membership of a together with the target-category hom carrier from p to q after the shared object suffix is exposed. Reading that row through the source and target object maps of Definition 62.114 identifies the endpoints as $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$, which are the ordinary component source and target for the two prefix functors at object a.

The classifier row is then fixed componentwise. The definition Definition 63.62 says that two displayed components over the same prefixes and object are classified exactly when both component carriers are present and their component histories are related by hsame. The semantic certificate of Definition 63.66 packages that component classifier as the public NameCert classifier for the displayed component carrier.

Finally, the public natural-transformation classifier is the componentwise row specified in Definition 63.61. Substituting the prefix carrier and classifier rows above gives ordinary natural transformations between the two prefix functors, with no extra source data: the functor ledgers come from Definition 62.114, the component morphism ledger is the displayed $\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(q$ and stability is exactly the existing component transport and vertical composition package. These are the listed bridge fields. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier standard bridge fields`

Definition 63.102 (NatTrans functor-category prefix carrier)**.** *The prefix functor-category carrier for* $\mathsf{NatTrans}^{\uparrow}$ *has unary prefix histories as objects. A hom row from object prefix* p *to object prefix* q *is the component family* $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta_a)$, *one row for every unary object* a. *The displayed source and target of the* a*-component are* $\mathsf{append}(p, a)$ *and* $\mathsf{append}(q, a)$, *and the component morphism is the concrete target-category row* $\mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(q, a), \eta_a)$. *The hom classifier is the componentwise* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ *row, so equality of hom rows is read by* hsame *on every displayed component history.*

Theorem 63.103 (NatTrans functor-category identity and composition)**.** *For the prefix functor-category carrier of Definition 63.102, each prefix object* p *carries an identity hom row whose* a*-component is the empty component/identity semantic certificate at* $\mathsf{append}(p, a)$. *If* $\eta : p \Rightarrow q$ *and* $\theta : q \Rightarrow r$ *are hom rows, their composite is the componentwise vertical composite* $\theta_a \circ \eta_a$. *Associativity and the left and right unit laws are the componentwise laws inherited from the checked prefix component carrier and the classifier rows.*

Proof. The object row is already a unary prefix history, and the identity component at a is the identity morphism row for $\mathsf{append}(p, a)$. The semantic NameCert certificate for the prefix component classifier reads this row as the empty identity component, so every component belongs to the hom carrier from p to p.

For composition, take the vertical composite in the target-category hom carrier at each unary object a. The source and target boundaries line up because the two component rows end and begin at the shared prefix q after the same object suffix a is appended.

The associativity and unit equations are then checked componentwise. The classifier for hom rows is componentwise hsame, and the component carrier already supplies vertical composition closure, so the category laws reduce to the corresponding checked classifier rows at every unary object. □

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixFunctorCategory identity and composition`

Definition 63.104 (NatTrans prefix natural-isomorphism carrier)**.** *A prefix natural isomorphism from p to q is a pair of component families $\eta : p \Rightarrow q$ and $\epsilon : q \Rightarrow p$ in the prefix functor-category carrier such that, for every unary object a, both vertical composites $\epsilon_a \circ \eta_a$ and $\eta_a \circ \epsilon_a$ classify as the empty identity component at the corresponding appended histories* $\mathsf{append}(p, a)$ *and* $\mathsf{append}(q, a)$*.*

Theorem 63.105 (NatTrans natural-isomorphism obligation package)**.** *The prefix natural-isomorphism surface carries the NameCert obligation rows for carrier, classifier, stability, inverse, ledger, exactness, and no-host-leak. Its carrier is Definition 63.104; its classifier is the componentwise prefix component classifier in both directions; stability is given by component transport and vertical composition; the inverse row swaps the two component families; the ledger is the two functor ledgers together with the forward and reverse component morphism ledgers; exactness is the componentwise identity-composite condition; and no-host-leak follows because all rows are built from* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*,* $\mathsf{NatTrans}^{\uparrow}$*,* $\mathsf{UnaryHistory}$*,* Cont*, and the displayed component histories.*

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransPrefixNaturalIso component cycle empty boundary`

Proof. The carrier and classifier rows are obtained by unpacking Definition 63.104. Each direction is a prefix natural-transformation hom row, and each component is classified by $\mathsf{NatTransPrefixComponentClassifier}_{p,}$ or $\mathsf{NatTransPrefixComponentClassifier}_{q,p,a}$.

Stability is componentwise. Transport preserves the displayed component carrier, and vertical composition closes the composites used in the identity conditions. Swapping the forward and reverse families gives the inverse row, because the two identity-composite requirements are symmetric in the displayed definition.

The ledger and exactness rows are the public rows already present in the component-family surface. The only histories recorded are the two prefix functor ledgers and the forward and reverse component morphism ledgers, and the exactness condition says precisely that the two displayed vertical composites classify as empty identity components at every unary object. No additional host datum is introduced. □

Theorem 63.106 (NatTrans prefix natural isomorphisms are functor-category isomorphisms)**.** *For unary prefixes p and q, the data of Definition 63.104 are exactly isomorphisms from p to q in the prefix functor-category of Definition 63.102. The forward hom is the component family $\eta : p \Rightarrow q$, the reverse hom is the component family $\epsilon : q \Rightarrow p$, and the two inverse laws are precisely the componentwise empty-identity classifications for $\epsilon_a \circ \eta_a$ at* $\mathsf{append}(p, a)$ *and $\eta_a \circ \epsilon_a$ at* $\mathsf{append}(q, a)$*.*

Proof. Start with a prefix natural isomorphism. Its two component families are hom rows in the prefix functor-category, by Definition 63.104. The identity and composition rows of Definition 63.103 identify composition in that category with componentwise vertical composition, and identify identity homs with the empty identity component at each appended history. Thus the two carrier conditions are exactly the left and right inverse equations for an isomorphism from p to q.

Conversely, take an isomorphism from p to q in the prefix functor-category. Its forward and reverse homs are component families in the carrier of Definition 63.102. Reading the two categorical

inverse equations through Definition 63.103 gives, for each unary object a, the empty-identity classification of the composites $\epsilon_a \circ \eta_a$ and $\eta_a \circ \epsilon_a$ at the two appended histories. These are exactly the two conditions of Definition 63.104.

The obligation surface of Definition 63.105 supplies the classifier, stability, inverse, ledger, and exactness rows for this comparison. Since both sides use the same forward and reverse component families, the same componentwise prefix classifiers, and the same empty-identity composite rows, the bridge is obtained by componentwise classifier comparison. □

Theorem 63.107 (NatTrans functor-category isomorphism classifier exactness)**.** *For unary prefixes p and q, let (η, ϵ) and (η', ϵ') be two witnesses in the carrier of Definition 63.104. These two witnesses classify as the same isomorphism row in the prefix functor-category of Definition 63.102 exactly when, for every unary object a, the forward components η_a and η'_a classify by* $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ *and the reverse components ϵ_a and ϵ'_a classify by* $\mathsf{NatTransPrefixComponentClassifier}_{q,p,a}$*. Under these two componentwise classifier families, the two empty-identity composite rows are transported by vertical-composition stability and by the identity/composition laws of Definition 63.103.*

Proof. Unpack the hom classifier in Definition 63.102. A row between p and q is a component family indexed by unary objects, and the classifier on such rows is the pointwise prefix component classifier. Thus equality of the forward hom rows is exactly the family of $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$ classifications, while equality of the reverse hom rows is exactly the family of $\mathsf{NatTransPrefixComponentClassifier}_{q,p,a}$ classifications.

Apply Definition 63.103. The isomorphism laws in the prefix functor-category reduce to the componentwise vertical composites $\epsilon_a \circ \eta_a$ and $\eta_a \circ \epsilon_a$ being classified as the empty identity components at $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$. Componentwise classifier agreement on the forward and reverse families transports these two composite rows through the same vertical composition stability used in the functor-category composition law.

The remaining rows are precisely the obligation rows of Definition 63.105: transport, inverse, ledger, and exactness are all read pointwise from the same component families. Hence two prefix natural-isomorphism witnesses determine the same BEDC isomorphism row exactly in the stated componentwise classifier situation. No host equality of transformations, function extensionality, arbitrary representability, or external transformation type is exported. □

Theorem 63.108 (NatTrans mature package inventory)**.** *The public* $\mathsf{NatTrans}^{\uparrow}$ *paper surface contains the mature package inventory for the prefix natural-transformation instance: component-family data, naturality readback, identity transformations, vertical composition, horizontal composition, whiskering and prewhiskering rows, whiskering preservation, horizontal-vertical interchange, the prefix standard bridge of Definition 63.101, the functor-category carrier and laws of Definition 63.102 and Definition 63.103, and the natural-isomorphism carrier and obligation package of Definition 63.104 and Definition 63.105, together with the identification of those carriers as isomorphisms in the prefix functor-category from Definition 63.106 and the componentwise isomorphism classifier exactness of Definition 63.107.*

Lean4 ✓ `BEDC.Derived.NatTransUp.NatTransUp StdBridge`

Proof. Read the inventory by grouping the public rows already established in the chapter. The component-family carrier, componentwise classifier, naturality readback, identity transformation, vertical composition, horizontal composition, whiskering, prewhiskering, preservation, and interchange rows give the public natural-transformation surface.

The bridge row is supplied by Definition 63.101. The functor-category object, hom, identity, and composition rows are supplied by Definition 63.102 and Definition 63.103.

Finally, the natural-isomorphism rows are supplied by Definition 63.104 and Definition 63.105. The comparison with prefix functor-category isomorphisms is supplied by Definition 63.106. These rows are all expressed through the existing BEDC carriers, classifiers, ledgers, and component histories, so the listed package is closed as a current paper surface. □

Theorem 63.109 (NatTrans Category-Functor prefix bridge exhaustion)**.** *Every public prefix bridge read for* $\mathsf{NatTrans}^{\uparrow}$ *factors through the finite three-layer BEDC surface consisting exactly of the* $\mathsf{Cat}^{\uparrow}$ *unary-continuation object, Hom, identity, and composition bridge rows; the* $\mathsf{Functor}^{\uparrow}$ *fixed-prefix object-map, morphism-map, identity-preservation, and composition-preservation bridge rows; the* $\mathsf{NatTrans}^{\uparrow}$ *component-family, naturality, vertical-composition, and horizontal-composition rows; and the displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *provenance that those rows carry. This factorization exports no host category universe, host functor equality, external transformation type, representability, adjunction data, or arbitrary functor-category primitive.*

Proof. First read the $\mathsf{Cat}^{\uparrow}$ dependency lattice. By Definition 61.336, downstream functor and natural-transformation clauses consume only the displayed object, Hom, identity, composition, continuation, and classifier rows of the unary-continuation category. These rows are already the bridge-facing BHist, hsame, Cont, Pkg, and NameCert data, so the first layer contains no host category carrier or external morphism primitive.

Next read the fixed-prefix functor package. The standard bridge of Definition 62.114 specializes $\mathsf{Functor}^{\uparrow}$ to the object map $a \mapsto \mathsf{append}(p, a)$ and the induced morphism rows, with identity and composition preservation witnessed through the same $\mathsf{Cat}^{\uparrow}$ classifiers. Thus the second layer adds only the fixed-prefix object and morphism preservation rows and does not introduce host functor equality or a separate functor category.

Finally read the $\mathsf{NatTrans}^{\uparrow}$ surface through Definition 63.101 and Definition 63.108. The component-family carrier, componentwise classifier, naturality readback, vertical composition, horizontal composition, whiskering rows, and prefix functor-category rows are all expressed as displayed component histories over the two fixed-prefix functors.

Repacking these three layers gives a single finite prefix bridge surface: category rows supply the endpoints and composition calculus, functor rows supply the two fixed-prefix endpoint maps, and natural-transformation rows supply the component family and its composition ledgers. The exclusions in the statement are exactly the non-claimed data absent from those rows, so the public read is exhausted by the listed BEDC-native provenance. □

Closure status for $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$**.**

Constructive story: A natural transformation here is a component family between two fixed-prefix functors: for unary prefixes p and q, each unary object a receives a displayed morphism η_a from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$ in the $\mathsf{Cat}^{\uparrow}$ hom carrier. The carrier admits only those component histories whose source and target are the two functor images and whose naturality, identity, vertical composition, horizontal composition, whiskering, and prewhiskering rows are witnessed through the existing functor and category classifiers. Two component families classify together componentwise, using $\mathsf{NatTransPrefixComponentClassifier}_{p,q,a}$, which is just carried component data plus hsame on every displayed component history rather than host equality of transformations. The source, pattern, and ledger rows are the two functor ledgers together with the component morphism ledgers, the naturality square readbacks, the composition and whiskering ledgers, and the prefix natural-isomorphism rows where the two component cycles classify as empty identities. The five name-certificate obligations are assembled componentwise: identities give inhabited component families, reflexivity, symmetry, and transitivity come from the component classifiers, and carrier closure follows by component transport and vertical-composition stability. A component-family history h lies in the signature gap of a $\mathsf{NatTrans}^{\uparrow}$ token p when the source-domain accepts the two functor endpoints and every component row, the signature relation pairs that family with an answer family and evidence ledger, and token introduction names that answer as p; unfolded, every unary object must

carry exactly the component morphism and naturality evidence encoded by the token. The standard bridge reads the package as ordinary natural transformations between fixed-prefix functors, and reads it back as the same componentwise hom-carrier and classifier rows. The chapter registers this component-family construction under the name $\mathsf{NatTrans}^{\uparrow}$; naturality is carried by visible component ledgers, not by host function extensionality, arbitrary representability, adjunction data, or an external transformation type.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public natural-transformation interface covers component-family data, naturality readback, identity transformations, opposite and double-opposite rows, vertical and horizontal composition, functor whiskering and prewhiskering, horizontal identity reductions, horizontal-vertical interchange, whiskering preservation, componentwise classifier rows grounded by the prefix component classifier certificate, the prefix standard bridge theorem, the functor-category prefix carrier and laws, the prefix natural-isomorphism package, and the identification of prefix natural isomorphisms with isomorphisms in the prefix functor-category.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.NatTransUp.NatTransPrefixComponentClassifier\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Adjunction units and counits, arbitrary representable bridges, Lean bridge checks for the functor-category and natural-isomorphism rows, and bridgeChecked targets are outside this chapter's closure claim.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 64

A Concrete Naming Certificate for $\mathsf{Prime}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the prime number interface $\mathsf{Prime}^{\uparrow}$ at the paper-proof level. The construction sits above the unary natural-number certificate $\mathsf{Nat}^{\uparrow}$ (chapter 28) and the additive certificate $\mathsf{Add}^{\uparrow}$ (chapter 29), and is the first arithmetic chapter to exhibit a non-trivial property of unary histories beyond pure additive structure. The certificate is fully constructive: every theorem produces an explicit witness, primality is decidable by trial division on the underlying unary segment, and Euclid's infinitude of primes is supplied as an effective bound function rather than as a non-constructive existence claim. The standard limit results — the fundamental theorem of arithmetic in both existence and uniqueness directions — are reduced, in the same chapter, to closure laws on the prime classifier.

64.1 Multiplicative structure on unary histories

Definition 64.1 (Unary multiplication relation)**.** *For unary histories d, q, n with $\mathsf{Unary}(d)$, $\mathsf{Unary}(q)$, and $\mathsf{Unary}(n)$, the multiplicative relation $\mathsf{NatMul}(d, q, n)$ holds when n is the unary continuation of d repeated q times. Concretely, NatMul is the closed inductive predicate generated by:*

1. $\mathsf{NatMul}(d, \mathsf{e}, \mathsf{e})$ *for every unary d (zero multiplied by anything is zero);*
2. *if $\mathsf{NatMul}(d, q, n)$ and $\mathsf{Cont}(n, d, n')$, then $\mathsf{NatMul}(d, \mathsf{E1}(q), n')$ (one more copy of d appended).*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatMul`

Lemma 64.2 (Unary multiplication successor inversion)**.** *If $\mathsf{NatMul}(d, \mathsf{E1}(q), n')$, then there is a predecessor product n such that $\mathsf{NatMul}(d, q, n)$ and $\mathsf{Cont}(n, d, n')$.*

Proof. Invert the closed successor clause in the inductive definition of NatMul. The displayed predecessor product and continuation witness are exactly the constructor payloads. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ inversion`

Lemma 64.3 (Unary multiplication exponent-result cases)**.** *Every witness of $\mathsf{NatMul}(d, q, n)$ is either the empty exponent case, with $q\mathsf{hsamee}$ and $n\mathsf{hsamee}$, or a successor exponent case: there are histories q_0 and n_0 such that $q = \mathsf{E1}(q_0)$, $\mathsf{NatMul}(d, q_0, n_0)$, and $\mathsf{Cont}(n_0, d, n)$.*

Proof. Invert the closed multiplication witness. The zero constructor gives the empty exponent and empty result. The successor constructor exposes exactly the predecessor exponent, predecessor product, and final continuation payload. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul exponent result cases`

Lemma 64.4 (Empty left unary multiplication has empty result)**.** *If* $\mathsf{NatMul}(\mathsf{e}, q, n)$*, then* $n\mathsf{hsamee}$*.*

Proof. Induct on the concrete multiplication witness. The zero constructor is immediate. In the successor constructor, the induction hypothesis identifies the previous product with e, and the final continuation is therefore an empty-left continuation whose result is e. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul empty left result empty`

Theorem 64.5 (Successor multiplication with empty result has empty left factor)**.** *If* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$ *and* $n\mathsf{hsamee}$*, then* $d\mathsf{hsamee}$*.*

Proof. Invert the successor multiplication witness. Its final step is a continuation $\mathsf{Cont}(n_0, d, n)$. Since the result is empty, empty-result inversion for Cont forces both inputs of that continuation to be empty, in particular the left multiplicand d. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ result empty left empty`

Theorem 64.6 (Successor result exposes positive multiplicand shape)**.** *If* $\mathsf{NatMul}(d, q, \mathsf{E1}(n))$*, then the multiplicand has one-headed unary shape:*

$$\exists t,\ d\mathsf{hsameE1}(t) \wedge \mathsf{UnaryHistory}(t).$$

Proof. The multiplication witness carries a unary classifier for d. If d were e, empty-left multiplication would force the successor result to be history-same to e, contradicting constructor separation. The zero-headed case contradicts unary exactness. Thus $d = \mathsf{E1}(t)$, and the same unary classifier exposes a unary tail. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul successor result multiplicand positive shape`

Theorem 64.7 (Successor result exposes positive multiplier shape)**.** *If* $\mathsf{NatMul}(d, q, \mathsf{E1}(n))$*, then the multiplier has one-headed unary shape:*

$$\exists t,\ q\mathsf{hsameE1}(t) \wedge \mathsf{UnaryHistory}(t).$$

Proof. The multiplication witness carries a unary classifier for q. The empty multiplier is impossible because the empty multiplier constructor returns e, not a successor result. The zero-headed case contradicts unary exactness. Hence $q = \mathsf{E1}(t)$, and the tail is unary by one-headed inversion. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul successor result multiplier positive shape`

Theorem 64.8 (Successor multiplication has positive result shape)**.** *If* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$ *and* d *is not history-same to* e*, then* n *has one-headed shape:*

$$\exists r,\ \mathsf{hsame}(n, \mathsf{E1}(r)) \wedge \mathsf{UnaryHistory}(r).$$

Proof. Invert the successor multiplication witness. The final step is a continuation $\mathsf{Cont}(n_0, d, n)$, and the multiplication witness supplies unary closure of d and n_0. The empty and zero-headed cases for d are impossible by non-emptiness and unary exactness. Thus $d = \mathsf{E1}(t)$, and the continuation has result $\mathsf{E1}(\mathsf{append}(n_0, t))$, whose tail is unary by append closure. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ result positive shape`

Theorem 64.9 (Nonempty multiplier multiplication has positive result shape)**.** *If* $\mathsf{NatMul}(d, q, n)$*, q is not history-same to* e*, and d is not history-same to* e*, then the result has one-headed shape:*

$$\exists t,\ \mathsf{hsame}(n, \mathsf{E1}(t)) \wedge \mathsf{UnaryHistory}(t).$$

Proof. The multiplication witness carries a unary classifier for q. The empty case contradicts the displayed non-emptiness, and the zero-headed case contradicts unary exactness. Hence $q = \mathsf{E1}(q_0)$, and Definition 64.8 applies to the same multiplication witness and the nonempty multiplicand. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul nonempty multiplier result positive shape`

Theorem 64.10 (Successor multiplication empty result exactness)**.** *If* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$*, then the result is empty exactly when the left multiplicand is empty:*

$$\mathsf{hsame}(n, \mathsf{e}) \iff \mathsf{hsame}(d, \mathsf{e}).$$

Proof. The forward implication is the successor empty-result inversion above. For the reverse implication, identify d with e and apply the empty-left multiplication result theorem to the same multiplication witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ result empty left empty iff`

Lemma 64.11 (Unary multiplication right operand is unary)**.** *Every closed witness of* $\mathsf{NatMul}(d, q, n)$ *carries a unary multiplier q.*

Proof. Induct on the NatMul witness. The zero constructor has multiplier e, and the successor constructor applies $\mathsf{E1}$ to the previous unary multiplier. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul right unary`

Lemma 64.12 (Unary multiplication is functional)**.** *For each pair of unary histories (d, q), there is at most one unary n with* $\mathsf{NatMul}(d, q, n)$*, up to* hsame*.*

Proof. By induction on the second argument q, using the determinacy of Cont at each appending step. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul functional`

Theorem 64.13 (Nonempty multiplicand result cancellation)**.** *Let d be a unary history with $d \not\mathsf{hsame}\mathsf{e}$. If* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{NatMul}(d, r, m)$ *have history-same results, then the multipliers are history-same:*

$$\mathsf{NatMul}(d, q, n) \wedge \mathsf{NatMul}(d, r, m) \wedge n \mathsf{hsame} m \implies q \mathsf{hsame} r.$$

Proof. Induct on the first multiplication witness. In the zero case, the common result is empty, and empty-result exactness for a nonempty multiplicand forces the second multiplier to be empty. In the successor case, a zero witness on the right would make the appended nonempty multiplicand empty, contradicting the hypothesis on d. The successor–successor branch cancels the common continuation suffix d from the two result rows and applies the induction hypothesis to the predecessor products, then rebuilds the one-headed multiplier equality. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul nonempty multiplicand result cancel`

Theorem 64.14 (Unary multiplication commutes up to history sameness)**.** *For unary histories d and q, if* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{NatMul}(q, d, m)$*, then* $n\mathsf{hsame}m$*.*

Proof. The concrete multiplication witness fixes the BEDC word length of its result as the product of the word lengths of its two unary inputs. The two displayed products therefore have equal lengths by commutativity of natural-number multiplication. Since unary histories are determined by their word length, the two multiplication ledgers have history-same results. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul comm hsame`

Theorem 64.15 (Nonempty multiplier result cancellation)**.** *Let q be a unary history with q /* $\mathsf{hsame}\mathsf{e}$*. If* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{NatMul}(r, q, m)$ *have history-same results, then the multiplicands are history-same:*

$$\mathsf{NatMul}(d, q, n) \wedge \mathsf{NatMul}(r, q, m) \wedge n\mathsf{hsame}m \Longrightarrow d\mathsf{hsame}r.$$

Proof. Read unary witnesses for d, q, and r from the two multiplication witnesses. By totality of unary multiplication, form reversed products $\mathsf{NatMul}(q, d, n')$ and $\mathsf{NatMul}(q, r, m')$.

Apply Definition 64.14 to the first pair of products and again to the second pair. These two commutativity comparisons transport the assumed result equality $n\mathsf{hsame}m$ to $n'\mathsf{hsame}m'$.

Now apply Definition 64.13 to the reversed products, using the nonempty multiplicand q. It gives $d\mathsf{hsame}r$, as required. □

Theorem 64.16 (Unary multiplication associates up to history sameness)**.** *For unary histories a, b, c, if ab realizes* $\mathsf{NatMul}(a, b, -)$*, ℓ realizes* $\mathsf{NatMul}(ab, c, -)$*, bc realizes* $\mathsf{NatMul}(b, c, -)$*, and r realizes* $\mathsf{NatMul}(a, bc, -)$*, then* $\ell\mathsf{hsame}r$*.*

Proof. Both displayed products are unary. Their BEDC word lengths read back as $(|a| \cdot |b|) \cdot |c|$ and $|a| \cdot (|b| \cdot |c|)$, respectively; natural-number associativity identifies these lengths, and unary length readback identifies the two histories up to hsame. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul assoc hsame`

Lemma 64.17 (Unary multiplication is total on unary inputs)**.** *For every pair of unary histories (d, q) there exists a unary n with* $\mathsf{NatMul}(d, q, n)$*.*

Proof. By unary induction on q. The base case yields $n = \mathsf{e}$; the step appends one further copy of d using Cont. Closure of unary histories under Cont (chapter 28) keeps n unary. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul total`

Lemma 64.18 (Unary multiplication results are unary)**.** *For every unary multiplicand d, every derivation* $\mathsf{NatMul}(d, q, n)$ *has a unary result n.*

Proof. Induct on the NatMul derivation. The zero constructor returns e, and the successor constructor appends the unary multiplicand to the unary inductive result, so unary closure under Cont applies. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul result unary`

Theorem 64.19 (Append-continuation multiplication closure)**.** *Let d, w, q, n, e, and r be histories. If the two displayed products share the same multiplicand and the two product results concatenate, then the appended multiplier has the concatenated result:*

$$\mathsf{NatMul}(d, w, n) \wedge \mathsf{NatMul}(d, q, e) \wedge \mathsf{Cont}(n, e, r) \Longrightarrow \mathsf{NatMul}(d, \mathsf{append}(w, q), r).$$

Proof. Induct on the left product. The zero case reduces the appended multiplier to q and the displayed continuation to the left-empty continuation for e.

In the successor case, the inductive product composes the predecessor product with the right product along the predecessor continuation. One further successor multiplication step appends the common multiplicand d to that intermediate result, and associativity of continuation concatenation identifies the displayed result r. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append cont`

Theorem 64.20 (NatMul total functional law surface)**.** *On* UnaryHistory *inputs, the concrete* NatMul *relation is total, has unary outputs, and is functional up to* hsame*. Equivalently, for all unary histories d and q there is a unary history n with* $\mathsf{NatMul}(d, q, n)$*, and whenever* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{NatMul}(d, q, n')$ *are two products from the same inputs, the outputs satisfy* $n \mathsf{hsame} n'$*.*

Proof. Use Definition 64.17 to obtain a displayed product $\mathsf{NatMul}(d, q, n)$ for unary inputs d and q.

Apply Definition 64.18 to that product to read the result back inside UnaryHistory. For any second product from the same inputs, apply Definition 64.12 to identify the two displayed results up to hsame. □

Theorem 64.21 (NatMul one-history unit law surface)**.** *For the one-step unary history* $\mathsf{E1}(\mathsf{e})$ *and every unary history d, the concrete multiplication rows display one-history unit laws:*

$$\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), d, n) \Longrightarrow n \mathsf{hsame} d$$

and

$$\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n) \Longrightarrow n \mathsf{hsame} d.$$

The proof dependencies are the Lean-side rows BEDC.Derived.PrimeUp.NatMul_unit_left_hsame *and* BEDC.Derived.PrimeUp.NatMul_unit_right_hsame*.*

Proof. For the left unit, unfold the multiplication derivation against the one-history multiplicand. The zero branch is excluded by the displayed multiplier, and the single successor branch leaves exactly the multiplier history, transported through the recorded hsame endpoint.

For the right unit, induct over the one-history multiplier. The sole successor step appends one copy of d to the zero product, and the continuation unit identifies the result with d up to hsame. □

Theorem 64.22 (NatMul append-composition law surface)**.** *The multiplicative composition row consumed by* $\mathsf{Mul}^{\uparrow}$ *is the* NatMul *append law together with the shared* Cont *ledger:*

$$\mathsf{NatMul}(d, w, n) \wedge \mathsf{NatMul}(d, q, e) \wedge \mathsf{Cont}(n, e, r) \Longrightarrow \mathsf{NatMul}(d, \mathsf{append}(w, q), r).$$

Thus the ledger records the two source products and the continuation witness that composes their results.

Proof. Project the two source product rows and the displayed $\mathsf{Cont}(n, e, r)$ ledger witness. Then apply Definition 64.19.

The conclusion is the same NatMul row on the appended multiplier, with no host multiplication, quotient, or classical choice principle. The only composition data read by consumers are the source products and the shared continuation ledger. □

Theorem 64.23 (Concrete unary multiplication MulUp certificate spine)**.** *The concrete unary multiplication candidate for* MulUp *uses carrier* UnaryHistory *over* BHist*, classifier* hsame*, operation relation* NatMul*, and unit* E1(e)*. Its certificate spine is the following finite law package.*

1. *Totality: for unary histories d and q, there is a unary n with* $\mathsf{NatMul}(d, q, n)$*.*
2. *Result closure: every* $\mathsf{NatMul}(d, q, n)$ *with unary d has unary result n.*
3. *Functionality: any two results of* $\mathsf{NatMul}(d, q, -)$ *for unary inputs are* hsame*.*
4. *Unit laws:* $\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q, n)$ *gives n*hsame*q, and* $\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n)$ *gives n*hsame*d.*
5. *Commutativity: products* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{NatMul}(q, d, m)$ *over unary inputs have n*hsame*m.*
6. *Associativity: for unary a, b, and c, products* $\mathsf{NatMul}(a, b, ab)$*,* $\mathsf{NatMul}(ab, c, \ell)$*,* $\mathsf{NatMul}(b, c, bc)$*, and* $\mathsf{NatMul}(a, bc, r)$ *have ℓ*hsame*r.*

Proof. The carrier, classifier, operation, and unit are the concrete objects already used by NatMul. Totality, result closure, functionality, the two unit laws, and commutativity are exactly the checked unary multiplication laws in this section.

For associativity, induct on the middle multiplication spine. The base case is the zero multiplier. The successor case rewrites the right-associated product as an appended multiplier product, applies Definition 64.19, and then uses functionality to compare it with the displayed left-associated result. □

Theorem 64.24 (Unary multiplication MulUp semantic exit package)**.** *For unary histories d and q, the result carrier* $\mathsf{NatMul}(d, q, -)$ *has a* SemanticNameCert *with classifier* hsame*. Together with the concrete carrier* UnaryHistory *over* BHist*, classifier* hsame*, operation relation* NatMul*, unit* E1(e)*, totality, result closure, functionality, the two unit laws, commutativity, and associativity, this supplies the* MulUp *phase exit package over* BHist*.*

Proof. Apply Definition 64.23 for the algebraic spine. Its rows are precisely the concrete carrier, classifier, operation, unit, totality, result closure, functionality, left and right unit laws, commutativity, and associativity for unary multiplication.

The result carrier certificate is the checked NatMul_result_semanticNameCert theorem. It takes the unary inputs d and q and returns a SemanticNameCert for the predicate $\mathsf{NatMul}(d, q, -)$ with classifier hsame.

The dependency list is therefore the checked multiplication family NatMul_total, NatMul_result_unary, NatMul_functional, NatMul_unit_left_hsame, NatMul_unit_right_hsame, NatMul_comm_hsame, NatMul_assoc_hsame, and the result-name certificate. These rows are exactly the finite exit data needed for the full natural-arithmetic phase object to consume MulUp as a single semantic package. □

Theorem 64.25 (MulUp semantic exit audit row coverage)**.** *The concrete unary multiplication package over* BHist *is audit-covered by the semantic exit theorem, the* SemanticNameCert *result row, the two* $\mathsf{Add}^{\uparrow}$ *distributivity surfaces, and the unary-prefix* $\mathsf{Order}^{\uparrow}$ *monotonicity row. All rows are read through* UnaryHistory*,* hsame*,* Cont*,* NatMul*, and* PreorderPrefixLE*, so the displayed surface is a single consumption point for the full natural-arithmetic phase.*

Proof. Apply Definition 64.24. It packages the carrier, classifier, multiplication relation, and SemanticNameCert result row over the displayed unary histories.

The additive interaction rows are Definition 64.27 and Definition 64.28. They use the same Cont presentation of unary addition and return to the same NatMul relation.

For the order-facing row, use Definition 64.26. Finally, Definition 64.49 records that these additive and order rows share the same concrete carrier and classifier surface, so the coverage package introduces no additional BEDC object. □

Theorem 64.26 (Unary multiplication preserves unary-prefix order)**.** *Let* $d, q, q', n, n' : \mathsf{Hist}$ *be unary histories. If* $\mathsf{PreorderPrefixLE}(q, q')$*,* $\mathsf{NatMul}(d, q, n)$*, and* $\mathsf{NatMul}(d, q', n')$*, then* $\mathsf{PreorderPrefixLE}(n, n')$*. The displayed prefix witness from* n *to* n' *is built from the* Cont *ledger for the extra unary tail of* q' *over* q*.*

Proof. Invert the $\mathsf{PreorderPrefixLE}(q, q')$ witness through the $\mathsf{Order}^{\uparrow}$ unary-prefix export. It presents a unary tail t and a continuation ledger with q' read as the unary append of q and t.

Apply Definition 64.19 to the multiplication row $\mathsf{NatMul}(d, q, n)$ and to that extra tail ledger. This constructs a product row for d against the displayed append presentation of q'.

Use NatMul functionality to compare the constructed product with the given result n'. The comparison transports the appended-product endpoint through hsame without changing the consumed unary tail.

Finally apply the unary-prefix append-tail classifier for $\mathsf{PreorderPrefixLE}$. The prefix from n to the constructed appended product is the continuation ledger supplied by multiplication; transporting it through the final hsame comparison gives the required $\mathsf{PreorderPrefixLE}(n, n')$ witness. □

Theorem 64.27 (NatMul right additive distribution surface)**.** *Let* $d, w, q, n, e, r : \mathsf{Hist}$ *be unary histories. If* $\mathsf{NatMul}(d, w, n)$*,* $\mathsf{NatMul}(d, q, e)$*, and the* $\mathsf{Add}^{\uparrow}$ *continuation row presents the unary sum of* w *and* q *as* $\mathsf{append}(w, q)$ *with* $\mathsf{Cont}(n, e, r)$*, then* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$*. Consequently the concrete* $\mathsf{Mul}^{\uparrow}$ *operation distributes over the right-hand* $\mathsf{Add}^{\uparrow}$ *continuation presentation of unary addition.*

Proof. The two multiplication rows supply the factor products for d against w and q. Apply Definition 64.19 to these factor rows and to the displayed continuation $\mathsf{Cont}(n, e, r)$; it constructs the product row for d against the appended multiplier $\mathsf{append}(w, q)$ with endpoint r.

The $\mathsf{Add}^{\uparrow}$ concrete NameCert surface reads unary addition through the same $\mathsf{UnaryHistory}$ carrier, continuation ledger, and append presentation. Therefore the continuation row that displays the sum of w and q is exactly the additive boundary required by the multiplication closure theorem.

Repack the constructed NatMul row together with that $\mathsf{Add}^{\uparrow}$ boundary. The resulting surface is the displayed $\mathsf{Mul}^{\uparrow}/\mathsf{Add}^{\uparrow}$ phase interaction: multiplication by the fixed unary history d sends the right-hand additive presentation $w + q$ to the continuation presentation of the two products. □

Theorem 64.28 (NatMul left additive distribution surface)**.** *Let* $d, w, q, n, e, r : \mathsf{Hist}$ *be unary histories. If* $\mathsf{NatMul}(w, d, n)$*,* $\mathsf{NatMul}(q, d, e)$*, and the* $\mathsf{Add}^{\uparrow}$ *continuation row presents the unary sum of* w *and* q *as* $\mathsf{append}(w, q)$ *with* $\mathsf{Cont}(n, e, r)$*, then* $\mathsf{NatMul}(\mathsf{append}(w, q), d, r)$*. Consequently the concrete* $\mathsf{Mul}^{\uparrow}$ *operation distributes over the left-hand* $\mathsf{Add}^{\uparrow}$ *continuation presentation of unary addition.*

Proof. Use NatMul commutativity on the two displayed left-factor products. It supplies right-factor product presentations for multiplication by the fixed unary history d, with endpoints identified with n and e by hsame.

Apply Definition 64.27 to those right-factor presentations and to the transported continuation ledger. This constructs a product row whose multiplier is $\mathsf{append}(w, q)$ and whose endpoint is identified with r.

Finally apply NatMul commutativity to the constructed row. The result is a product with multiplicand $\mathsf{append}(w, q)$ and multiplier d; functionality transports the endpoint comparison back along the original $\mathsf{Cont}(n, e, r)$ boundary, yielding $\mathsf{NatMul}(\mathsf{append}(w, q), d, r)$. □

Lemma 64.29 (Unit left multiplication reads back the multiplier)**.** *For every unary history q, if* $\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q, n)$*, then* $\mathsf{hsame}(n, q)$*.*

Proof. Induct on the concrete multiplication derivation for q. The zero case is immediate. In the successor case, unary inversion exposes the tail multiplier, the induction hypothesis identifies the previous product with that tail, and determinacy of Cont identifies the final appended unit with the canonical successor. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unit left hsame`

Lemma 64.30 (Unit right multiplication reads back the multiplicand)**.** *For every history d, if* $\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n)$*, then* $\mathsf{hsame}(n, d)$*.*

Proof. Invert the successor multiplication witness at multiplier $\mathsf{E1}(\mathsf{e})$. Its predecessor multiplier is e, so the predecessor product is the zero constructor result e. The final continuation is therefore the empty-left append of d, which identifies the result with d. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unit right hsame`

Lemma 64.31 (Unit right multiplication iff)**.** *For every unary history d, multiplication on the right by the unit history returns exactly the multiplicand up to history-sameness:*

$$\mathsf{UnaryHistory}(d) \Longrightarrow (\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n) \Longleftrightarrow \mathsf{hsame}(n, d)).$$

Proof. The forward direction is Definition 64.30. For the reverse direction, reduce along the displayed history-sameness. The zero multiplication constructor gives $\mathsf{NatMul}(d, \mathsf{e}, \mathsf{e})$ from the unary carrier, and the successor constructor appends d to the empty product using the left-unit continuation law. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unit right iff`

Theorem 64.32 (Unary multiplication two-sided unit exactness)**.** *For unary histories q and d, left and right multiplication by the unit history* $\mathsf{E1}(\mathsf{e})$ *are exact up to history-sameness:*

$$\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q, n) \Longleftrightarrow \mathsf{hsame}(n, q)$$

and

$$\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n) \Longleftrightarrow \mathsf{hsame}(n, d).$$

Proof. For the left-hand equivalence, the forward direction is Definition 64.29. For the reverse direction, transport along the displayed $\mathsf{hsame}(n, q)$ identification and apply the closed unit-left multiplication theorem NatMul_unit_left_closed.

For the right-hand equivalence, apply Definition 64.31. Its statement is exactly the right unit exactness claim for unary multiplicands. □

Theorem 64.33 (Unit multiplication result forces unit factors)**.** *If a unary multiplication witness returns the unit history, then both the multiplicand and the multiplier are the unit history:*

$$\mathsf{NatMul}(d, q, \mathsf{E1}(\mathsf{e})) \Longrightarrow \mathsf{hsame}(d, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(q, \mathsf{E1}(\mathsf{e})).$$

Proof. Invert the multiplication witness at the unit result. The zero constructor is impossible. In the successor case, continuation inversion at an E1 result either makes the multiplicand empty, contradicting empty-left multiplication, or exposes a one-headed multiplicand whose tail continuation has empty result. Empty-result inversion then forces that tail and the predecessor product to be empty, so the multiplier tail is empty by the nonempty-multiplicand empty-result exactness theorem. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unit result factors unit`

Lemma 64.34 (E0-multiplier exclusion)**.** *No unary multiplication witness has an* E0*-headed multiplier: for all histories* d, q, n*,* $\mathsf{NatMul}(d, \mathsf{E0}(q), n)$ *is impossible.*

Proof. By constructor inversion on the closed relation NatMul: its multiplier argument is generated only by e and E1-successor cases. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul e0 multiplier absurd`

Theorem 64.35 (Unary multiplication zero-headed component impossibility)**.** *If* $\mathsf{NatMul}(d, q, n)$ *holds, then none of* d*,* q*, or* n *can be a zero-headed history.*

Proof. The multiplication witness exposes unary evidence for the multiplicand, multiplier, and result. Each possible zero-headed equality contradicts the corresponding unary-history witness by the zero-extension exclusion theorem. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul zero headed component absurd`

Theorem 64.36 (Empty multiplier exactness for unary multiplication)**.** *For all histories* d, n*,*

$$\mathsf{NatMul}(d, \mathsf{e}, n) \iff \mathsf{UnaryHistory}(d) \wedge \mathsf{hsame}(n, \mathsf{e}).$$

Proof. The forward direction inverts the closed NatMul witness. The only constructor with empty multiplier is the empty-multiplier row, which records the unary field for d and returns the empty result. Conversely, if d is unary and $n$hsamee, identify the endpoint with e and apply the empty-multiplier constructor. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul empty right iff`

Theorem 64.37 (Empty multiplicand exactness for unary multiplication)**.** *For all histories* q, n*,*

$$\mathsf{NatMul}(\mathsf{e}, q, n) \iff \mathsf{UnaryHistory}(q) \wedge \mathsf{hsame}(n, \mathsf{e}).$$

Proof. For the forward direction, every closed multiplication witness carries a unary multiplier, and empty-left multiplication has empty result by induction over the witness.

Conversely, identify the displayed result with e and induct on the unary history q. The empty case is the zero constructor for $\mathsf{NatMul}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. A zero-headed unary case is impossible. In the E1 case, apply the induction hypothesis to the tail and then append one empty block using the successor constructor together with $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul empty left iff`

Theorem 64.38 (Nonempty multiplicand and empty multiplication result)**.** *For every unary multiplicand* d *with* d /hsamee*, the empty-result case of* $\mathsf{NatMul}(d, q, \mathsf{e})$ *is equivalent to the empty multiplier* $q$hsamee*.*

Proof. The forward direction inverts the NatMul derivation. The zero constructor gives the empty multiplier. A successor constructor would include a continuation $\mathsf{Cont}(n, d, \mathsf{e})$; empty-result inversion for Cont forces $d$hsamee, contradicting the nonempty multiplicand hypothesis. The reverse direction is the zero constructor after identifying q with e. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul nonempty multiplicand empty result iff`

Lemma 64.39 (Empty unary multiplication result exposes an empty factor)**.** *If* $\mathsf{NatMul}(d, q, \mathsf{e})$*, then either the multiplicand is empty* $d$hsamee*, or the multiplier is empty* $q$hsamee*.*

Proof. Invert the closed multiplication witness. The zero constructor has multiplier e. A successor constructor ends with a continuation $\mathsf{Cont}(n, d, \mathsf{e})$, and empty-result inversion for Cont forces $d$hsamee. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul empty result factor empty or multiplier empty`

Theorem 64.40 (Unary multiplication empty result iff an input is empty)**.** *If* $\mathsf{NatMul}(d, q, n)$*, then*

$$n\mathsf{hsamee} \iff d\mathsf{hsamee} \lor q\mathsf{hsamee}.$$

Proof. For the forward implication, transport the displayed result equality to the empty-result case and apply Definition 64.39.

For the converse implication, split on the empty input. If the multiplicand is empty, Definition 64.4 identifies the product result with e. If the multiplier is empty, the empty-right multiplication iff gives the canonical empty product for the same multiplicand; functionality of NatMul identifies the displayed result with that empty product. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul empty result iff factor empty or multiplier empty`

Theorem 64.41 (Successor multiplication result equal to factor exposes an empty input)**.** *If* d *is unary,* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$*, and* $n$hsame$d$*, then either* $d$hsamee *or* $q$hsamee*.*

Proof. Invert the successor multiplication to obtain a predecessor product $\mathsf{NatMul}(d, q, p)$ and a continuation $\mathsf{Cont}(p, d, n)$. Transport that continuation across $n$hsame$d$, then cancel its common right factor against the displayed unit continuation $\mathsf{Cont}(\mathsf{e}, d, d)$. This gives $p$hsamee. Transport the predecessor product to empty result and apply the empty-result factor split. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ result eq factor empty or multiplier empty`

Theorem 64.42 (Nonempty unary multiplication factors have nonempty result)**.** *If neither* d *nor* q *is history-same to* e*, then no witness* $\mathsf{NatMul}(d, q, n)$ *can have* $n$hsamee*.*

Proof. Transport the result equality to the empty-result case and apply the preceding empty-result factor theorem. Its two alternatives are exactly the two excluded empty-factor cases. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul nonempty factors result not empty`

Theorem 64.43 (Append-continuation multiplication empty result split)**.** *Let* d *be unary and not history-same to* e*. If* $\mathsf{NatMul}(d, w, n)$*,* $\mathsf{NatMul}(d, q, e)$*, and* $\mathsf{Cont}(n, e, r)$*, then* $r$hsamee *exactly when both* $w$hsamee *and* $q$hsamee*.*

Proof. Append-continuation closure first gives a multiplication witness for $\mathsf{append}(w, q)$ with result r. The nonempty-multiplicand empty-result theorem reduces the empty result to an empty appended multiplier, and append empty-result exactness splits that condition into the two factors. The converse builds the empty appended multiplier and uses multiplication functionality to identify the continuation result with e. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append cont empty result empty factors iff`

Theorem 64.44 (Successor multiplicand appends the multiplier)**.** *If* $\mathsf{NatMul}(d, q, n)$*, then there is a unary history* r *such that* $\mathsf{Cont}(n, q, r)$ *and* $\mathsf{NatMul}(\mathsf{E1}(d), q, r)$*.*

Proof. Induct on the concrete multiplication witness. The empty multiplier case chooses $r = \mathsf{e}$. In the successor case, the induction hypothesis displays the predecessor result as the continuation of the old product by the predecessor multiplier. The final multiplication step appends d, while the successor multiplicand step appends $\mathsf{E1}(d)$. Associativity of continuation and commutativity of unary append exchange the predecessor multiplier with d, giving the displayed continuation by $\mathsf{E1}(q)$ and the successor-multiplicand multiplication witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ multiplicand append multiplier total`

Lemma 64.45 (Append multiplier right factor readback)**.** *Let* d *and* q *be unary histories. If* $\mathsf{NatMul}(d, w, n)$ *and* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$ *hold, and if* $\mathsf{Cont}(n, e, r)$ *displays* e *as the right factor of the result continuation, then* $\mathsf{NatMul}(d, q, e)$*.*

Proof. Use totality of unary multiplication to build a product e' for d and q. Append-continuation closure then produces a product for $\mathsf{append}(w, q)$ with result $\mathsf{append}(n, e')$. Functionality of NatMul identifies this result with the displayed r. Transport the continuation $\mathsf{Cont}(n, e', \mathsf{append}(n, e'))$ across that result equality, left-cancel it against $\mathsf{Cont}(n, e, r)$, and transport the product for q from e' to e. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append multiplier right factor`

Lemma 64.46 (Append multiplier left factor readback)**.** *Let* d *and* w *be unary histories. If* $\mathsf{NatMul}(d, q, e)$ *and* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$ *hold, and if* $\mathsf{Cont}(n, e, r)$ *displays* n *as the left factor of the result continuation, then* $\mathsf{NatMul}(d, w, n)$*.*

Proof. Use totality of unary multiplication to build a product n' for d and w. Append-continuation closure combines this product with the supplied product for q, producing a product for $\mathsf{append}(w, q)$ with result $\mathsf{append}(n', e)$. Functionality of NatMul identifies that result with the displayed r. Transport the continuation $\mathsf{Cont}(n', e, \mathsf{append}(n', e))$ across this result equality, right-cancel it against $\mathsf{Cont}(n, e, r)$, and transport the product for w from n' to n. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append multiplier left factor`

Theorem 64.47 (Append multiplier result deterministic)**.** *Let* d *be unary. If* $\mathsf{NatMul}(d, w, n)$ *and* $\mathsf{NatMul}(d, q, e)$ *are displayed factor products,* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$ *is a product over the appended multiplier, and* $\mathsf{Cont}(n, e, z)$ *displays the combined result, then* $r\mathsf{hsame}z$*.*

Proof. Append-continuation multiplication closure combines the two factor products along $\mathsf{Cont}(n, e, z)$, producing a multiplication witness for $\mathsf{append}(w, q)$ with result z. Functionality of NatMul for the unary multiplicand d compares this constructed witness with the displayed appended-multiplier witness and identifies the two results. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append multiplier result deterministic`

Theorem 64.48 (MulUp unary product order lower-bound package)**.** *Let d, q, and n be unary histories. If* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$*, then the* MulUp *product row exports the following* $\mathsf{Order}^{\uparrow}$ *boundary package over the unary-prefix order of Definition 54.45:*

1. *the carrier rows* $\mathsf{UnaryHistory}(d)$*,* $\mathsf{UnaryHistory}(\mathsf{E1}(q))$*, and* $\mathsf{UnaryHistory}(n)$*;*
2. *the unary-prefix comparison* $d \leq_{\mathsf{u}} n$*;*
3. *the endpoint classifier boundary* ρ_{u} *used by* $\mathsf{Order}^{\uparrow}$ *to read equality of endpoints under two-sided* $\leq_{\mathsf{u}}$ *comparison.*

Proof. First expose the multiplication package by Definition 64.24. Its carrier row gives the displayed unary histories for d and $\mathsf{E1}(q)$, and its result closure gives $\mathsf{UnaryHistory}(n)$ from the supplied $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$ witness.

The empty-tail boundary $q = \mathsf{e}$ is the right-unit product. The unit law in the same MulUp package reads the product result back as $n\mathsf{hsame}d$; the $\mathsf{Order}^{\uparrow}$ package of Definition 54.45 then uses reflexivity and endpoint classifier transport to obtain $d \leq_{\mathsf{u}} n$.

For a successor tail, apply Definition 64.44 to the predecessor multiplication row. The appended-multiplier factor lemmas Definition 64.46 and Definition 64.45 recover the left product and right product from the displayed continuation product. Then Definition 64.47 identifies the displayed result with the append of the predecessor product and the extra multiplicand copy.

This append display is exactly the unary-prefix witness that the original multiplicand lies below the product. Project the relation row and classifier row from Definition 54.45: the relation row records $d \leq_{\mathsf{u}} n$, and the classifier row records ρ_{u} as the endpoint boundary used whenever two-sided $\leq_{\mathsf{u}}$ comparison collapses the endpoints. □

Theorem 64.49 (MulUp AddUp OrderUp phase interaction package)**.** *For unary histories d, w, q, n, e, r, the concrete unary* $\mathsf{Mul}^{\uparrow}$ *package exports a single consumer package combining the following rows:*

1. *the right additive distribution row over the* $\mathsf{Add}^{\uparrow}$ *continuation presentation, sending* $\mathsf{NatMul}(d, w, n)$*,* $\mathsf{NatMul}(d, q, e)$*, and* $\mathsf{Cont}(n, e, r)$ *to* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$*;*
2. *the left additive distribution row over the same* $\mathsf{Add}^{\uparrow}$ *continuation presentation, sending* $\mathsf{NatMul}(w, d, n)$*,* $\mathsf{NatMul}(q, d, e)$*, and* $\mathsf{Cont}(n, e, r)$ *to* $\mathsf{NatMul}(\mathsf{append}(w, q), d, r)$*;*
3. *the unary-prefix* $\mathsf{Order}^{\uparrow}$ *lower-bound row for successor products, which reads* $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$ *as the boundary* $\mathsf{PreorderPrefixLE}(d, n)$ *together with the endpoint classifier row.*

The shared consumer objects are BHist*,* UnaryHistory*,* hsame*,* Cont*,* NatMul*, and* PreorderPrefixLE*.*

Proof. The right additive row is exactly Definition 64.27. It uses the $\mathsf{Add}^{\uparrow}$ continuation presentation through the same $\mathsf{Cont}(n, e, r)$ ledger and keeps the product predicate inside NatMul.

The left additive row is Definition 64.28. Its proof transports through commutativity of NatMul and returns to the left-hand presentation without changing the underlying BHist carrier, UnaryHistory predicate, hsame classifier, or continuation row.

The order row is supplied by Definition 64.48. It repacks the successor-product multiplication witness into the unary-prefix relation PreorderPrefixLE and the $\mathsf{Order}^{\uparrow}$ endpoint classifier. Since all three rows consume the displayed NatMul and Cont ledgers over the same unary histories, the package combines them without introducing another carrier or classifier. □

Definition 64.50 (Divisibility relation)**.** *For unary histories d, n, the divisibility relation* $\mathsf{NatDivides}(d, n)$ *holds when there exists a unary q with* $\mathsf{NatMul}(d, q, n)$*. The witness q is the explicit quotient.*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatDivides`

Theorem 64.51 (Divisibility result unary carrier)**.** *If* $\mathsf{NatDivides}(d, n)$ *holds, then* n *is a unary history.*

Proof. Unpack the quotient witness. The multiplication witness exposes unary d, and multiplication result closure then proves the result history n unary. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides result unary`

Lemma 64.52 (Divisibility result cases)**.** *Every witness of* $\mathsf{NatDivides}(d, n)$ *is either an empty-result witness, with* $n\mathsf{hsamee}$ *and* d *unary, or a successor-product witness: there is a predecessor* n_0 *with* $\mathsf{NatDivides}(d, n_0)$ *and* $\mathsf{Cont}(n_0, d, n)$*.*

Proof. Unpack the explicit quotient in the divisibility witness. If its NatMul witness is zero, the result is empty and the multiplicand field gives unary d. If it is a successor witness, unary inversion on the quotient supplies a unary predecessor quotient, and the predecessor multiplication witness repackages as $\mathsf{NatDivides}(d, n_0)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides result cases`

Theorem 64.53 (Divisibility zero-headed component impossibility)**.** *If* $\mathsf{NatDivides}(d, n)$ *holds, then neither the divisor* d *nor the dividend* n *can be a zero-headed history.*

Proof. Unpack the explicit quotient q in the divisibility witness. The underlying $\mathsf{NatMul}(d, q, n)$ witness rules out zero-headed multiplicand, multiplier, and result components. Applying that impossibility to the divisor or dividend component gives the claim. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides zero headed component absurd`

Lemma 64.54 (Empty left divisibility has empty result)**.** *If* $\mathsf{NatDivides}(\mathsf{e}, n)$*, then* $n\mathsf{hsamee}$*.*

Proof. Unpack the quotient witness for divisibility. Its multiplication field is a witness of $\mathsf{NatMul}(\mathsf{e}, q, n)$, so empty-left multiplication exactness identifies the result n with e. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides empty left result empty`

Theorem 64.55 (Successor-result divisors have positive shape)**.** *If* $\mathsf{NatDivides}(d, \mathsf{E1}(n))$*, then* d *has positive unary shape: there is a unary history* t *with* $d\mathsf{hsameE1}(t)$*.*

Proof. Unpack the divisibility witness and read the unaryness of d from its multiplication field. If d were empty, empty-left divisibility would force $\mathsf{E1}(n)\mathsf{hsamee}$, contradicting successor non-emptiness. The $e0$ case is excluded by unary-history inversion. Hence d is $e1$-headed, and the unary tail is obtained by unary inversion. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides successor result divisor positive shape`

Theorem 64.56 (Nonempty-result divisors have positive shape)**.** *If* $\mathsf{NatDivides}(d, n)$*,* n *is unary, and* $n \not\mathsf{hsamee}$*, then* d *has positive unary shape: there is a unary history* t *with* $d\mathsf{hsameE1}(t)$*.*

Proof. Unpack the quotient witness in $\mathsf{NatDivides}(d, n)$ and read the unaryness of d from its multiplication field. If $d\mathsf{hsamee}$, then empty-left divisibility forces $n\mathsf{hsamee}$, contradicting the displayed nonempty result. The $e0$-headed case is excluded by unary-history inversion, so the divisor is $e1$-headed and unary inversion supplies the tail carrier. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides nonempty result divisor positive shape`

Theorem 64.57 (Empty-left divisibility exactness)**.** *For every history n, divisibility by the empty unary history holds exactly when the result is empty:*

$$\mathsf{NatDivides}(\mathsf{e}, n) \iff \mathsf{hsame}(n, \mathsf{e}).$$

Proof. The forward implication is empty-left divisibility result exactness. For the reverse implication, identify n with e; then the empty quotient and the zero multiplication constructor witness $\mathsf{NatDivides}(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides empty left iff`

Lemma 64.58 (Empty right divisibility is unary carrier)**.** *For any history d,* $\mathsf{NatDivides}(d, \mathsf{e})$ *holds exactly when d is a unary history.*

Proof. In the forward direction, unpack the quotient witness; its multiplication field carries the unary multiplicand field for d. In the reverse direction, use the empty quotient and the zero constructor of NatMul. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides empty right iff`

Theorem 64.59 (Unit-left divisibility is unary carrier)**.** *For any history n,* $\mathsf{NatDivides}(\mathsf{E1}(\mathsf{e}), n)$ *holds exactly when n is a unary history.*

Proof. For the forward direction, unpack the quotient witness q. Multiplication by the unit identifies the result with q, and unaryness transports across that endpoint identification. Conversely, the unit-divisibility half of reflexivity gives $\mathsf{NatDivides}(\mathsf{E1}(\mathsf{e}), n)$ for every unary n. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides unit left iff`

Theorem 64.60 (Unit-right divisibility exactness)**.** *For any history d,* $\mathsf{NatDivides}(d, \mathsf{E1}(\mathsf{e}))$ *holds exactly when* $d\mathsf{hsameE1}(\mathsf{e})$.

Proof. In the forward direction, unpack the quotient witness and apply unit-result multiplication factor exactness: both factors of a product equal to $\mathsf{E1}(\mathsf{e})$ are the unary unit. Conversely, identify d with the unary unit and use reflexive divisibility for the unit history. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides unit right iff`

Lemma 64.61 (Divisibility is reflexive on the unit and on every unary history)**.** *For every unary n,* $\mathsf{NatDivides}(\mathsf{E1}(\mathsf{e}), n)$ *and* $\mathsf{NatDivides}(n, n)$.

Proof. The unit case takes $q = n$ in $\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), n, n)$ by induction. The self-divisibility case takes $q = \mathsf{E1}(\mathsf{e})$ and uses the continuation unit at the empty quotient. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides reflexive`
Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides unit self reflexive`
Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides reflexive pair`

Lemma 64.62 (Divisibility is transitive)**.** *If* $\mathsf{NatDivides}(d, e)$ *and* $\mathsf{NatDivides}(e, n)$*, then* $\mathsf{NatDivides}(d, n)$.

Proof. From the witnessing quotients q_1 with $\mathsf{NatMul}(d, q_1, e)$ and q_2 with $\mathsf{NatMul}(e, q_2, n)$, the composite quotient is $q_1 \cdot q_2$, itself constructed as a unary history by Definition 64.17. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides transitive`

Theorem 64.63 (Unary divisibility partial-order certificate)**.** *On unary histories,* NatDivides *is reflexive and transitive, and it is antisymmetric modulo* hsame*: for unary histories d, e, and n,* $\mathsf{NatDivides}(d, d)$*,* $\mathsf{NatDivides}(d, e) \wedge \mathsf{NatDivides}(e, n) \Rightarrow \mathsf{NatDivides}(d, n)$*, and* $\mathsf{NatDivides}(d, e) \wedge \mathsf{NatDivides}(e, d) \Rightarrow d\,\mathsf{hsame}\,e$*.*

Proof. The reflexive clause is Definition 64.61, and the transitive clause is Definition 64.62. The antisymmetric clause is Definition 64.181.

This is a paper-level bundled certificate rather than a distinct Lean marker site: the three checked laws are consumed together by bounded divisor search, Euclid's lemma, and factorization uniqueness as the order-like interface for unary divisibility. □

Theorem 64.64 (MulUp divisibility-order consumer package)**.** *For unary histories,* NatDivides *supplies the* $\mathsf{Mul}^{\uparrow}$ *arithmetic consumer package consisting of reflexivity, transitivity, antisymmetry modulo* hsame*, right-factor closure and left-multiplication closure through* NatMul*, and the unary-prefix lower-bound package for successor products. Thus any* $\mathsf{Mul}^{\uparrow}$ *consumer that reads only unary divisibility order, ordinary product closure, and successor-product prefix lower bounds may cite a single package boundary rather than reassembling those rows separately.*

Proof. The order rows are exactly the reflexive and transitive laws of Definition 64.61 and Definition 64.62, together with the antisymmetry certificate of Definition 64.181.

The multiplication-closure rows are supplied by Definition 64.89 and Definition 64.90. The boundary row that connects successor products with the unary-prefix order is Definition 64.48. Collecting these checked rows gives the stated $\mathsf{Mul}^{\uparrow}$ consumer surface. □

Theorem 64.65 (MulUp public semantic exit certificate)**.** *The concrete unary multiplication surface obtained from Definition 64.24, Definition 64.25, the right and left additive distribution surfaces Definition 64.27 and Definition 64.28, and Definition 64.26 is the public* $\mathsf{NameCert}_{\mathsf{Mul}^{\uparrow}}$ *consumption package over* BHist *for the full natural-arithmetic phase.*

A consumer of this package may read only the carrier UnaryHistory*, the classifier* hsame*, the continuation ledger* Cont*, the multiplication relation* NatMul*, the semantic result* NameCert *row, and the displayed* $\mathsf{Add}^{\uparrow}/\mathsf{Order}^{\uparrow}$ *interaction rows. The certificate does not expose cancellation, semiring completeness, or standard-bridge data.*

Proof. Project the semantic exit package of Definition 64.24. This gives the unary carrier over BHist, the classifier, the multiplication relation, the unit, the finite multiplication law rows, and the semantic result NameCert row.

Apply the audit coverage theorem Definition 64.25. Its coverage statement identifies the same public rows with the two additive distribution surfaces and the unary-prefix monotonicity row, all through UnaryHistory, hsame, Cont, and NatMul.

Repack the two $\mathsf{Add}^{\uparrow}$ distribution rows and the $\mathsf{Order}^{\uparrow}$ monotonicity row as one public consumption boundary. Since every displayed row uses the same carrier, classifier, continuation ledger, and multiplication relation, the boundary is exactly the $\mathsf{NameCert}_{\mathsf{Mul}^{\uparrow}}$ package consumed by the full natural-arithmetic phase, while the excluded cancellation, semiring, and standard-bridge data remain outside the theorem. □

Definition 64.66 (MulUp Full Nat audit consumer surface)**.** *The Full Nat audit consumer surface for* $\mathsf{Mul}^{\uparrow}$ *is the finite row family over* BHist *consisting exactly of the carrier* UnaryHistory*, the classifier* hsame*, the continuation ledger* Cont*, the multiplication relation* NatMul*, the semantic result*

NameCert *row, the right and left* Add$^{\uparrow}$ *distributivity rows, the unary-prefix* Order$^{\uparrow}$ *monotonicity row, and the divisibility-order rows available in the phase-exit package.*

Every row in the surface is read through the same unary endpoints, the same history classifier, and the same finite continuation ledgers. The surface has no coordinate for host multiplication, quotient equality, semiring structure, or cancellation outside the displayed divisibility-order rows.

Proof. This is the row list projected from Definition 64.65 and Definition 64.71. The first theorem fixes the carrier, classifier, continuation, multiplication, semantic result, and Add$^{\uparrow}$/Order$^{\uparrow}$ rows; the phase-exit boundary adds the divisibility-order consumer rows over the same endpoints. □

Theorem 64.67 (MulUp Full Nat Lean target boundary)**.** *A later Lean audit target for the L7* Mul$^{\uparrow}$ *exit may consume exactly the rows of Definition 64.66:* UnaryHistory*,* hsame*,* Cont*,* NatMul*, the semantic result* NameCert *row, the two* Add$^{\uparrow}$ *distributivity rows, the* Order$^{\uparrow}$ *monotonicity row, and the divisibility-order rows. It may not consume host multiplication, quotient formation, semiring primitives, a cancellation law, or a second carrier.*

Proof. Start from the consumer normal form of Definition 64.70. It states that the Full Nat Arithmetic consumer reads the phase-exit boundary only through the listed BHist rows.

Next apply Definition 64.71. This fixes the concrete source of the NatMul semantic exit, the additive distribution rows, the order row, and the divisibility rows.

Therefore a Lean target whose purpose is audit consumption has no further input slot. Any host multiplication, quotient, semiring, cancellation, or alternate carrier would be outside the displayed row family and hence outside the boundary. □

Theorem 64.68 (MulUp L7 audit-clean exit surface)**.** *The L7 Full Nat Arithmetic phase-exit surface for* Mul$^{\uparrow}$ *is assembled by the checked* BEDC.Derived.PrimeUp.NatMul_unary_mulup_certificate *row, the checked* BEDC.Derived.PrimeUp.NatMul_result_semanticNameCert *row, the right and left* Add$^{\uparrow}$ *distributivity rows, the unary-prefix* Order$^{\uparrow}$ *monotonicity row, and the divisibility-order rows of the phase-exit package. All rows are over* UnaryHistory*,* hsame*, finite* Cont *ledgers, and* NatMul*.*

Proof. Project Definition 64.65. It supplies the public Mul$^{\uparrow}$ package and names the checked NatMul certificate and semantic result certificate as the multiplication core.

Use Definition 64.71 to attach the surrounding arithmetic rows: the two Add$^{\uparrow}$ distributivity rows, the Order$^{\uparrow}$ monotonicity row, and the divisibility-order consumer rows. The normal form theorem Definition 64.70 identifies these as the rows available to a Full Nat Arithmetic consumer.

Finally repack the rows through Definition 64.66. The repacking changes no carrier, classifier, continuation ledger, or multiplication relation, so the result is a single L7 audit-clean exit surface rather than a host arithmetic surface. □

Theorem 64.69 (MulUp semantic exit and divisibility-order phase package)**.** *Over* BHist *and the carrier* UnaryHistory*, the semantic exit theorem, semantic-exit audit row coverage, and divisibility-order consumer package together give the full natural-arithmetic consumer surface for* Mul$^{\uparrow}$*. The surface consists of the carrier* UnaryHistory*, classifier* hsame*, multiplication relation* NatMul*, result* SemanticNameCert*, the two* Add$^{\uparrow}$ *distributivity rows, the* Order$^{\uparrow}$ *monotonicity row, and the unary divisibility reflexive, transitive, and antisymmetric consumer rows.*

Proof. Apply Definition 64.24. It fixes the carrier, classifier, multiplication relation, unit rows, algebraic laws, and result SemanticNameCert for unary multiplication over BHist.

Next apply Definition 64.25. It adds the Add$^{\uparrow}$ distributivity rows and the Order$^{\uparrow}$ monotonicity row without leaving the displayed unary-history carrier or the hsame classifier surface.

Finally apply Definition 64.64. Its reflexive, transitive, and antisymmetric divisibility rows use the same UnaryHistory carrier, the same hsame comparison, the same Cont transport ledgers, and the same NatMul relation. Therefore the three packages assemble as one Full Nat Arithmetic consumption point for $\mathsf{Mul}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.MulupSemanticExitAndDivisibilityOrderPhasePackage`

Theorem 64.70 (MulUp Full Nat Arithmetic consumer normal form)**.** *Any Full Nat Arithmetic consumer of* $\mathsf{Mul}^{\uparrow}$ *over* BHist *whose interface is restricted to the package rows of Definition 64.71 reads exactly the following normal form: the carrier is* UnaryHistory*, the classifier is* hsame*, the result ledger is* NatMul*, and the exposed row list is the unit row, commutativity row, associativity row, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the divisibility-order consumer rows. It receives no host multiplication table, quotient carrier, semiring primitive, or cancellation law beyond those displayed rows.*

Proof. Start from the phase-exit boundary of Definition 64.71. It fixes the only carrier, classifier, multiplication relation, additive rows, order row, and divisibility rows available to a Full Nat Arithmetic consumer of the concrete $\mathsf{Mul}^{\uparrow}$ package over BHist.

Next apply the audit surface of Definition 64.78. That surface identifies the checked NatMul law rows and the result naming row while preserving the same UnaryHistory carrier, hsame classifier, finite Cont ledgers, and BHist endpoints.

Finally apply Definition 64.79 and Definition 64.82. The handoff identifies the allowed additive, order, and divisibility rows for L7 consumers, and the mature boundary rules out host multiplication, quotient equality, and a second carrier. Therefore the consumer normal form is precisely the displayed package row list and nothing outside it. □

Theorem 64.71 (MulUp Full Nat Arithmetic phase-exit boundary)**.** *Over* BHist *and the carrier* UnaryHistory*, the package of Definition 64.69 is exactly the* $\mathsf{Mul}^{\uparrow}$ *L7 consumer boundary required by the Full Nat Arithmetic phase. Its rows are the carrier* UnaryHistory*, classifier* hsame*, multiplication relation* NatMul*, unit, totality, result closure, functionality, unit laws, commutativity, associativity, the* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the divisibility-order consumer rows.*

Proof. Unfold the package theorem Definition 64.69. Its displayed surface already fixes the concrete carrier, classifier, multiplication relation, semantic result object, additive rows, order row, and divisibility-order rows over BHist and UnaryHistory.

The algebraic part comes from Definition 64.24: it supplies the unit, totality, result closure, functionality, unit laws, commutativity, and associativity rows for NatMul on the same hsame-classified unary histories.

The additive and order-facing part is the row coverage recorded by Definition 64.25 together with Definition 64.64. These give the $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the reflexive, transitive, and antisymmetric divisibility-order consumer rows without changing the carrier or classifier.

Thus a later Full Nat Arithmetic consumer can read $\mathsf{Mul}^{\uparrow}$ through this single boundary. No host natural-number multiplication, quotient object, or multiplication operation outside the unary-history package is used. □

Lean4 ✓ `BEDC.Derived.PrimeUp.MulupFullNatArithmeticPhaseExitBoundary`

Theorem 64.72 (MulUp Full Nat Arithmetic finite-window exhaustion)**.** *Every finite Full Nat Arithmetic consumer window whose entries are restricted to the boundary of Definition 64.71 factors through exactly the displayed* UnaryHistory *carrier, the* hsame *classifier, the* NatMul *ledger, the*

unit, totality, result-closure, functionality, unit-law, commutativity, and associativity rows, the two $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the reflexive, transitive, and antisymmetric unary divisibility-order rows. Conversely each listed row contributes one admissible entry to such a finite window.*

Proof. Project the phase-exit boundary of Definition 64.71. It names the carrier UnaryHistory, the classifier hsame, the NatMul ledger, and the exact algebraic, additive, order, and divisibility-order row list available to the consumer window.

Apply the semantic-exit package and audit surface from Definition 64.69 and Definition 64.78. These identify every window entry as one of the displayed rows, since the window is finite and each entry must be read through the same BHist endpoint and the same hsame classifier.

For the reverse reading, take any row in the displayed list. The package theorems already expose that row as part of the boundary, so it gives an admissible finite-window entry with the same carrier, classifier, and ledger.

It remains to exclude extra entries. The L7 Add/Order handoff and the mature boundary, Definition 64.79 and Definition 64.82, rule out host multiplication, quotient carriers, semiring primitives, and cancellation rows not displayed in the phase-exit boundary. Hence the finite window is exhausted by the listed unary-history rows. □

Theorem 64.73 (MulUp Full Nat Arithmetic hsame transport boundary)**.** *Let two unary-history presentations over* BHist *be related componentwise by* hsame*. Then the Full Nat Arithmetic phase-exit boundary of Definition 64.71 transports across that comparison. The transported boundary has the same* UnaryHistory *carrier surface, the transported* hsame *classifier, the transported* NatMul *relation, the same unit, the same* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the same* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the same divisibility-order consumer rows, all exported through the unchanged* BHist *surface.*

Proof. Project the phase-exit boundary of Definition 64.71. Its rows are already expressed over BHist, UnaryHistory, hsame, NatMul, the additive distribution rows, the order row, and the divisibility-order consumer rows.

Transport the carrier and classifier first. The componentwise hsame comparison keeps each unary-history row inside UnaryHistory and keeps every Cont ledger on the same displayed BHist endpoints. The functionality and transport clauses for NatMul then move the multiplication row, unit row, and result row to the transported presentation.

Finally, repack the AddUp distribution rows, the OrderUp monotonicity row, and the divisibility rows from the transported endpoints. These rows mention only the already transported unary histories, Cont ledgers, hsame comparisons, and NatMul witnesses, so the consumer receives the same phase-exit package and no new exported surface. □

Theorem 64.74 (MulUp successor cancellation boundary)**.** *Let d and q be unary histories, and let* $\mathsf{NatMul}(d, \mathsf{E1}(q), n')$ *be a product row over* BHist*. Then successor inversion produces a unary predecessor n with* $\mathsf{NatMul}(d, q, n)$ *and* $\mathsf{Cont}(n, d, n')$*. Moreover, if another unary history m satisfies* $\mathsf{NatMul}(d, q, m)$ *and* $\mathsf{Cont}(m, d, n')$*, then $n\,\mathsf{hsame}\,m$. The cancellation reads only the predecessor product rows and the displayed continuation witnesses; it does not add a second carrier, quotient surface, or host multiplication row.*

Proof. Apply BEDC.Derived.PrimeUp.NatMul_succ_inversion to the displayed successor product row. This gives the predecessor product n, the row $\mathsf{NatMul}(d, q, n)$, and the final continuation ledger $\mathsf{Cont}(n, d, n')$.

Now take another predecessor m with $\mathsf{NatMul}(d, q, m)$ and $\mathsf{Cont}(m, d, n')$. The two predecessor products have the same unary multiplicand d and multiplier q, so BEDC.Derived.PrimeUp.NatMul_functional identifies their results by $n$hsame$m$.

The continuation rows are retained as ledger witnesses for the displayed successor endpoint n'. They are not additional result carriers and play no role in the comparison beyond recording that both predecessor rows append the same visible copy of d to reach the same endpoint. □

Theorem 64.75 (MulUp checked law audit package)**.** *Over* BHist *and the carrier* UnaryHistory*, the checked NatMul law package consumed by the Full Nat Arithmetic boundary consists of the totality, result-closure, functionality, unit, unit-law, commutativity, associativity,* hsame *operation-congruence, and result semantic* NameCert *rows for* NatMul*. The package is read only through the unary-history carrier, the* hsame *classifier, and the displayed* BHist *result endpoints.*

Proof. The algebraic rows are the rows certified by BEDC.Derived.PrimeUp.NatMul_unary_mulup_certificate: totality, result closure, functionality, the unit row, unit laws, commutativity, and associativity all live on the same unary-history endpoints.

The transport row is the checked congruence theorem BEDC.Derived.PrimeUp.NatMul_operation_congruence. It says that componentwise hsame comparison of the input and result histories preserves the NatMul operation row, so the law package is stable under the phase-exit transport boundary.

The result naming row is the checked semantic certificate theorem BEDC.Derived.PrimeUp.NatMul_result_semar Repacking these three checked sources gives exactly the NatMul audit package required by the L7 boundary, without adding a second carrier, a host multiplication operation, or an extra quotient surface. □

Theorem 64.76 (MulUp unary standard bridge boundary)**.** *The unary-history* $\mathsf{Mul}^{\uparrow}$ *package admits a paper bridge to finite unary-word multiplication by reading each* $\mathsf{NatMul}(d, q, n)$ *ledger as repeated unary continuation. The carrier remains* UnaryHistory*, the classifier remains* hsame*, and the result row remains the displayed* NatMul *row. This reading uses the checked law audit package of Definition 64.75, the Full Nat Arithmetic phase-exit boundary of Definition 64.71, and the window-exhaustion boundary of Definition 64.72.*

Proof. Start with the phase-exit boundary. It already supplies the totality, functionality, unit-law, associativity, and commutativity rows for the displayed NatMul relation over UnaryHistory, and Definition 64.75 identifies those rows as the checked unary NatMul package.

Next project the rows that connect multiplication to the surrounding arithmetic window. The same boundary carries the $\mathsf{Add}^{\uparrow}$ distributivity rows, $\mathsf{Order}^{\uparrow}$ monotonicity row, and divisibility rows; the exhaustion statement of Definition 64.72 says that a consumer has no further Full Nat Arithmetic row to read inside this phase.

The external unary-word reading consumes only these displayed rows. It interprets each $\mathsf{NatMul}(d, q, n)$ ledger as a finite repeated-unary continuation and reads back to the same UnaryHistory endpoint, the same hsame classifier, and the same NatMul result row, without adding host multiplication, quotient equality, or cancellation beyond the displayed package. □

Theorem 64.77 (MulUp bridge consumer status alignment)**.** *The* $\mathsf{Mul}^{\uparrow}$ *chapter's paper bridge status is exactly the consumer package obtained by combining the unary standard bridge boundary of Definition 64.76, the checked law audit package of Definition 64.75, the phase-exit boundary of Definition 64.71, the window-exhaustion boundary of Definition 64.72, and the mature boundary of Definition 64.82. This package justifies recording paperBridge for* $\mathsf{Mul}^{\uparrow}$*, while it does not justify* bridgeChecked *until a Lean-side standard bridge theorem for the same boundary exists.*

Proof. The unary standard bridge boundary supplies the external paper reading: each $\mathsf{NatMul}(d, q, n)$ ledger is read as finite repeated-unary continuation and returns to the same UnaryHistory endpoint, hsame classifier, and displayed NatMul result row.

The checked law audit package and the phase-exit boundary identify the law rows that this reading may consume. They give totality, result closure, functionality, unit, unit laws, commutativity, associativity, operation congruence, result NameCert, the two $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the divisibility-order rows over the same BHist and UnaryHistory surface.

The window-exhaustion and mature-boundary theorems close the consumer side. They say that the finite Full Nat Arithmetic window has no row outside the displayed package, and in particular no host multiplication, quotient equality, host cancellation law, second carrier, or semiring surface is exported to the bridge consumer.

Thus the paper has a complete bridge account at the manuscript boundary: the external unary-word reading consumes exactly the audited rows and nothing beyond them. The verification axis remains below bridgeChecked, because the cited Lean target certifies the NatMul result semantic NameCert rather than a standard bridge theorem for this consumer boundary. □

Theorem 64.78 (MulUp Full Nat Arithmetic audit surface)**.** *Over* BHist *and the carrier* UnaryHistory*, a Full Nat Arithmetic consumer reads* $\mathsf{Mul}^{\uparrow}$ *exactly through the following rows: the* NatMul *totality row, result-closure row, functionality row, unit row, unit-law rows, commutativity row, associativity row,* hsame *operation-congruence row, result* NameCert *row, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the reflexive, transitive, and antisymmetric divisibility-order rows. This is the complete cited audit surface for the open Full Nat Arithmetic phase.*

Proof. Project the phase-exit boundary of Definition 64.71. It fixes the consumer boundary over BHist and UnaryHistory and contains the NatMul algebraic rows, the $\mathsf{Add}^{\uparrow}$ rows, the $\mathsf{Order}^{\uparrow}$ row, and the divisibility-order rows.

Use the checked law audit package of Definition 64.75. That package identifies the checked NatMul totality, result-closure, functionality, unit, unit-law, commutativity, associativity, operation-congruence, and result NameCert rows as the law-facing part of the same boundary.

Finally repack the remaining consumer rows already assembled in Definition 64.69: the two $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the reflexive, transitive, and antisymmetric divisibility-order rows. These rows share the same carrier, classifier, NatMul relation, and BHist endpoints, so the displayed list is a single Full Nat Arithmetic audit surface and not a host multiplication or quotient surface. □

Theorem 64.79 (MulUp L7 additive-order audit handoff)**.** *Any Full Nat Arithmetic consumer over* BHist *that reads the* $\mathsf{Mul}^{\uparrow}$ *package through the audit surface of Definition 64.78 may consume exactly the checked* NatMul *law rows, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the reflexive, transitive, and antisymmetric divisibility-order rows. All of these rows are over* UnaryHistory*,* hsame*, and finite* Cont *ledgers. The consumer may not introduce host multiplication, quotient equality, or a second carrier.*

Proof. Start from Definition 64.78. It fixes the only displayed Full Nat Arithmetic audit surface over BHist, UnaryHistory, hsame, and the finite Cont ledgers.

Unpack Definition 64.75 for the checked NatMul law rows. These are the totality, result-closure, functionality, unit, unit-law, commutativity, associativity, operation-congruence, and result NameCert rows already accepted by the $\mathsf{Mul}^{\uparrow}$ law audit.

Finally repack the additive-order rows from Definition 64.69. Those rows contribute exactly the two $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the divisibility-order rows, while preserving the same carrier, classifier, and ledger boundary. Hence the L7 consumer receives the listed rows and no host multiplication, quotient equality, or second carrier. □

Theorem 64.80 (MulUp L7 audit-clean exit obligation)**.** *For the concrete* $\mathsf{Mul}^{\uparrow}$ *surface over* BHist*, the Full Nat Arithmetic exit obligation is exactly the concrete* BHist *package consisting of the carrier* UnaryHistory*, the multiplication relation* NatMul*, the result* SemanticNameCert *row, both* $\mathsf{Add}^{\uparrow}$ *distributivity surfaces, the unary-prefix* $\mathsf{Order}^{\uparrow}$ *monotonicity row, and the unary divisibility reflexive and transitive consumer rows. This obligation is audit-clean precisely as that package and contains no cancellation law, semiring completion, host multiplication table, or standard bridge.*

Proof. Start with the L7 handoff of Definition 64.79. It fixes the allowed consumer rows over BHist, UnaryHistory, hsame, finite Cont ledgers, and the NatMul relation, and it prevents the consumer from adding host multiplication, quotient equality, or a second carrier.

Next use Definition 64.81. This gives the single phase-exit package whose visible rows are the NatMul semantic exit, the two $\mathsf{Add}^{\uparrow}$ distributivity surfaces, the unary-prefix $\mathsf{Order}^{\uparrow}$ monotonicity row, and the unary divisibility consumer rows over the same endpoints.

Finally apply the public semantic exit certificate Definition 64.24. It supplies the carrier UnaryHistory, the concrete NatMul package, and the result SemanticNameCert row. Repacking these public rows gives exactly the displayed Full Nat Arithmetic exit obligation, with cancellation, semiring completion, a host multiplication table, and a standard bridge left outside the audit-clean package. □

Theorem 64.81 (MulUp AddUp–OrderUp–divisibility phase-exit package)**.** *Over* BHist *and the carrier* UnaryHistory*,* $\mathsf{Mul}^{\uparrow}$ *exposes a single L7 Full Nat Arithmetic package whose rows are the* NatMul *semantic exit, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *unary-prefix monotonicity row, and the unary divisibility reflexive, transitive, and antisymmetric rows. The package is read through the same* BHist *endpoints,* UnaryHistory *carrier,* hsame *classifier,* Cont *ledgers, and* NatMul *relation.*

Proof. Apply Definition 64.24. It gives the NatMul semantic exit over BHist and UnaryHistory, including the result semantic certificate and the operation laws used by the L7 consumer.

Next apply Definition 64.25 and Definition 64.78. These identify the two $\mathsf{Add}^{\uparrow}$ distributivity rows and the $\mathsf{Order}^{\uparrow}$ unary-prefix monotonicity row as rows of the same audit surface, with no change of carrier, classifier, or result relation.

Finally apply Definition 64.64. Its reflexive, transitive, and antisymmetric divisibility rows are stated over the same BHist endpoints, UnaryHistory histories, hsame comparisons, Cont ledgers, and NatMul witnesses. Repacking these shared rows gives one named phase-exit package for downstream natural, rational, and prime consumers. □

Theorem 64.82 (MulUp Full Nat Arithmetic mature boundary)**.** *For the concrete* $\mathsf{Mul}^{\uparrow}$ *surface over* BHist *and* UnaryHistory*, the mature Full Nat Arithmetic boundary is exactly the paper-side consumer boundary formed by the* NatMul *semantic exit, the checked law audit package, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *unary-prefix monotonicity row, and the divisibility-order consumer rows. Every row is read over the same* BHist *endpoints,* UnaryHistory *carrier,* hsame *classifier,* Cont *ledgers, and* NatMul *relation. The boundary contains no host multiplication, quotient surface, or second carrier.*

Proof. Apply Definition 64.78. It fixes the common carrier, classifier, result relation, and finite ledger surface on which the Full Nat Arithmetic rows may be read.

Next apply Definition 64.81. It packages the NatMul semantic exit together with the two $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ monotonicity row, and the divisibility-order consumer rows over that same surface.

Finally apply Definition 64.79. This handoff identifies the allowed consumer rows and rules out host multiplication, quotient equality, and a second carrier. Repacking the three cited packages gives a single mature boundary for the present Full Nat Arithmetic phase. □

Definition 64.83 (MulUp NatMul window boundary)**.** *The* $\mathsf{Mul}^{\uparrow}$ NatMul *window boundary is the finite* BHist *read surface whose accepted rows are unary histories* d*,* q*, and* n *in* UnaryHistory*, componentwise comparison by* hsame*, continuation ledgers* Cont *witnessing the displayed append steps, and the multiplication row* $\mathsf{NatMul}(d, q, n)$*. A consumer inside the boundary may inspect only the empty-result branch, the successor-result branch, the unit rows, the two* $\mathsf{Add}^{\uparrow}$ *distributivity rows, the* $\mathsf{Order}^{\uparrow}$ *row, and the divisibility rows already present in Definition 64.71.*

Theorem 64.84 (MulUp empty product factor boundary)**.** *Within the window boundary of Definition 64.83, if* $\mathsf{NatMul}(d, q, \mathsf{e})$ *holds, then the exposed factor readback is exactly the empty-factor case or the empty-multiplier case:* $d$hsamee *or* $q$hsamee*. No nonempty unary pair can be exported as an empty product row by this boundary.*

Proof. Read the multiplication row through the checked empty-result cases for NatMul. The concrete theorem BEDC.Derived.PrimeUp.NatMul_empty_result_factor_empty_or_multiplier_empty inverts the empty result and returns precisely the displayed disjunction on the two factors.

The same readback is compatible with the window boundary because the boundary exposes only UnaryHistory, hsame, Cont, and the single NatMul row. Thus the empty result cannot be explained by an external multiplication table, quotient equality, or hidden host zero test. □

Theorem 64.85 (MulUp successor product factor readback)**.** *Within the window boundary of Definition 64.83, if* $\mathsf{NatMul}(d, q, \mathsf{E1}(n))$ *holds, then the result readback exposes a nonempty multiplicand factor and a unary multiplier row. Equivalently, the successor-result branch is readable only through the positive multiplicand shape supplied by the* NatMul *inversion rows together with the* UnaryHistory *classifier on the multiplier.*

Proof. First invert the successor result by the concrete NatMul successor rows. The checked theorem BEDC.Derived.PrimeUp.NatMul_succ_result_empty_left_empty rules out an empty multiplicand as the source of a successor product.

Next apply BEDC.Derived.PrimeUp.NatMul_successor_result_multiplicand_positive_shape. It reads the same successor-result row as a positive multiplicand shape, with the multiplier still classified as a unary history by the NatMul carrier projections.

These two steps are rows of the boundary of Definition 64.83. The consumer obtains the nonempty factor and unary multiplier without adding host multiplication or a second result classifier. □

Theorem 64.86 (MulUp NatMul window exhaustion)**.** *Every Full Nat Arithmetic consumer restricted to the* $\mathsf{Mul}^{\uparrow}$ *phase-exit boundary and to the window boundary of Definition 64.83 reads only the following cases: the empty-result factor boundary of Definition 64.84, the successor-result factor readback of Definition 64.85, the unit rows of the checked* NatMul *package, the two additive-distribution rows, the order row, and the divisibility rows. No further* BHist *case, host multiplication operation, quotient carrier, or external semiring law is visible at this audit handoff.*

Proof. Start with the phase-exit package of Definition 64.71. It fixes the Full Nat Arithmetic consumer surface over UnaryHistory, hsame, Cont, and NatMul, and it lists the unit, additive, order, and divisibility rows available to the consumer.

Now restrict the result inspection to the window boundary. The empty result is exhausted by Definition 64.84, while the successor result is exhausted by Definition 64.85. These are the only result-shape branches exposed by the NatMul row over BHist unary histories.

Finally repack the remaining rows from Definition 64.75 and Definition 64.79. They contribute the unit rows, the two $\mathsf{Add}^{\uparrow}$ distributivity rows, the $\mathsf{Order}^{\uparrow}$ row, and the divisibility rows already accepted by the phase-exit boundary, so the window exports exactly the stated audit handoff. □

Lemma 64.87 (Divisibility is closed under displayed continuation)**.** *If* $\mathsf{NatDivides}(d, x)$ *and* $\mathsf{NatDivides}(d, y)$ *hold, and* $\mathsf{Cont}(x, y, z)$ *displays* z *as the continuation of the two divisible histories, then* $\mathsf{NatDivides}(d, z)$ *holds.*

Proof. Unpack the two quotient witnesses, say qx and qy. Their multiplication witnesses have results x and y. The quotient for z is $\mathsf{append}(qx, qy)$; unary append closure makes it an admissible quotient, and append-continuation multiplication closure combines the two multiplication witnesses over the displayed $\mathsf{Cont}(x, y, z)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides cont closed`

Lemma 64.88 (Unary multiplication append multiplier split)**.** *Let* $d, w, q, r : \mathsf{BHist}$ *be unary where required. If the displayed unary multiplication relation proves* $\mathsf{NatMul}(d, \mathsf{append}(w, q), r)$*, then there are unary histories* n *and* e *such that*

$$\mathsf{NatMul}(d, w, n) \wedge \mathsf{NatMul}(d, q, e) \wedge \mathsf{Cont}(n, e, r).$$

Proof. Totality of unary multiplication supplies canonical products n and e for the two multiplier pieces w and q. Append-continuation closure for NatMul composes these two products into a displayed product over $\mathsf{append}(w, q)$. Functionality of unary multiplication compares this canonical displayed result with the given result r, and continuation transport along that history-sameness gives $\mathsf{Cont}(n, e, r)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append multiplier split exists`

Lemma 64.89 (Right factor divisibility of a displayed product)**.** *If* d *and* q *are unary histories and* $\mathsf{NatMul}(d, q, n)$ *displays* n *as their product, then* q *divides* n*:*

$$\mathsf{UnaryHistory}(d) \Longrightarrow \mathsf{UnaryHistory}(q) \Longrightarrow \mathsf{NatMul}(d, q, n) \Longrightarrow \mathsf{NatDivides}(q, n).$$

Proof. Totality of multiplication gives a displayed product for q by d. Commutativity up to hsame compares that product with the given display of d by q. The commuted display is a divisibility witness for q, and dividend transport across the result comparison carries it back to n. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides mul right closed`

Theorem 64.90 (Nat divisibility is closed under left multiplication)**.** *For histories* $d, x, q, z : \mathsf{Hist}$*, if* x *is unary,* $\mathsf{NatDivides}(d, q)$*, and* $\mathsf{NatMul}(x, q, z)$*, then* $\mathsf{NatDivides}(d, z)$*.*

Proof. Choose the quotient witness for $\mathsf{NatDivides}(d, q)$. Since q is a unary result of that divisor witness, total multiplication gives a product of q with x. Multiplicative commutativity identifies this product with the given product z up to hsame. Thus q divides z after dividend transport, and transitivity of NatDivides carries the original divisor d through q into z. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides mul left closed`

Theorem 64.91 (Divisibility is closed under displayed unary products)**.** *For unary* BHist *data* d, e, x, y, p, z*, if* NatDivides(d, x)*,* NatDivides(e, y)*,* NatMul(d, e, p)*, and* NatMul(x, y, z) *all hold, then* NatDivides(p, z) *holds.*

Proof. Unpack the two divisor witnesses. Thus x is displayed as d multiplied by a unary quotient q, and y is displayed as e multiplied by a unary quotient r.

Unary multiplication totality supplies a displayed product s of the two quotients q and r. Use associativity of NatMul up to hsame to compare the product of x and y with the iterated product of d, e, q, and r.

Commutativity up to hsame moves the displayed factorization so that the given product p of d and e is the left factor and s is the quotient factor. The resulting display witnesses NatDivides(p, z) after transporting the dividend along the final hsame comparison with the given NatMul(x, y, z) row. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides product closed`

Lemma 64.92 (Divisors of a nonempty unary product are bounded by the result)**.** *If* NatDivides(d, n)*, n is unary, and $n \not\text{hsame}$, then either $d$hsame$n$ or* NatUnaryStrictPrefix(d, n)*.*

Proof. Unpack the quotient q in the divisor witness and inspect the corresponding NatMul(d, q, n) witness. The empty quotient would make the product result empty, contradicting $n \not\text{hsame}$.

For the unit quotient, multiplication by the unit identifies the product result with d, so the divisor is the whole result up to hsame.

For a successor quotient, the multiplication witness writes n as the previous product followed by one more copy of d. The final Cont payload gives a strict unary prefix from that copy into the result; unary append commutativity and strict-prefix transport move the prefix endpoint to d, yielding NatUnaryStrictPrefix(d, n). □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides nonempty result boundary`

Definition 64.93 (MulUp audit-clean consumer boundary)**.** *The audit-clean consumer boundary exported by* Mul$^{\uparrow}$ *for the full natural arithmetic phase is exactly the following concrete surface:*

1. *the carrier rows* BHist *and* UnaryHistory*;*
2. *the classifier and continuation rows* hsame *and* Cont*;*
3. *the multiplication and divisibility rows* NatMul *and* NatDivides*;*
4. *the unary-prefix order row* PreorderPrefixLE*;*
5. *the* Add$^{\uparrow}$ *continuation boundary cited in Definition 64.49;*
6. *the* Order$^{\uparrow}$ *unary-prefix boundary cited in Definition 54.45.*

No consumer-facing row in this boundary is a host multiplication primitive, host order primitive, standard semiring carrier, or quotient carrier.

Theorem 64.94 (MulUp divisibility-order source exactness)**.** *Every consumer of the* Mul$^{\uparrow}$ *divisibility-order boundary of Definition 64.93 factors through exactly the four displayed package rows Definition 64.24, Definition 64.49, Definition 64.63, and Definition 64.64. Equivalently, a consumer may read* BHist*,* UnaryHistory*,* hsame*,* Cont*,* NatMul*,* NatDivides*,* PreorderPrefixLE*, and the cited* Add$^{\uparrow}$ *and* Order$^{\uparrow}$ *package boundaries, and no additional standard semiring, host order, or quotient carrier source is available.*

Proof. Begin with Definition 64.24. It supplies the carrier BHist, unary predicate UnaryHistory, classifier hsame, continuation row Cont, and the displayed product relation NatMul as the semantic exit package for multiplication results.

Next apply Definition 64.49. This records the only phase interaction rows used by the boundary: additive continuation distribution through $\mathsf{Add}^{\uparrow}$ and the successor-product lower bound through the $\mathsf{Order}^{\uparrow}$ unary-prefix relation PreorderPrefixLE.

The divisibility order itself is supplied by Definition 64.63, which bundles reflexive, transitive, and antisymmetric readback for NatDivides over unary histories. Finally, Definition 64.64 combines those order rows with multiplication closure and successor-product prefix lower bounds into the single consumer package.

The definition of the boundary lists no other rows. Hence any consumer whose source is this boundary can project only through the four package rows just assembled. In particular, there is no source slot for a standard semiring carrier, a host multiplication operation, a host order relation, or a quotient carrier. □

Theorem 64.95 (MulUp audit-clean result NameCert handoff)**.** *The audit-clean consumer boundary of Definition 64.93 factors through the concrete* NatMul *result semantic certificate, the checked-law audit package of Definition 64.75, the window-exhaustion boundary of Definition 64.86, and the divisibility-order source exactness of Definition 64.94. The resulting handoff exposes only* BHist*,* UnaryHistory*,* hsame*,* Cont*,* NatMul*,* NatDivides*,* PreorderPrefixLE*, the* $\mathsf{Add}^{\uparrow}$ *continuation rows, and the* $\mathsf{Order}^{\uparrow}$ *unary-prefix row.*

No output slot of the handoff exports host multiplication, host order, quotient carriers, or standard semiring data.

Proof. Start with the concrete NatMul result semantic certificate. It is the only result-side name certificate in the handoff, so the multiplication output is read as a BHist result row over unary inputs and the displayed NatMul relation.

Apply Definition 64.75. This contributes the checked multiplication laws through the already visible BHist, UnaryHistory, hsame, Cont, and NatMul rows, with the $\mathsf{Add}^{\uparrow}$ continuation rows and $\mathsf{Order}^{\uparrow}$ unary-prefix row kept as named dependencies rather than as ambient arithmetic structure.

Next use Definition 64.86. Window exhaustion says that the audit package has no remaining consumer-facing multiplication row outside the concrete NatMul window and its displayed continuation ledger.

Finally apply Definition 64.94. It identifies the order-facing source as exactly NatDivides, PreorderPrefixLE, and the cited $\mathsf{Add}^{\uparrow}$ and $\mathsf{Order}^{\uparrow}$ boundary rows. Since each cited component is already listed in Definition 64.93, their composite handoff cannot expose host multiplication, host order, quotient carriers, or standard semiring data. □

Theorem 64.96 (MulUp audit-clean boundary row exhaustion)**.** *Every row readable from the* $\mathsf{Mul}^{\uparrow}$ *audit-clean consumer boundary of Definition 64.93 is one of the displayed rows* BHist*,* UnaryHistory*,* hsame*,* Cont*,* NatMul*,* NatDivides*,* PreorderPrefixLE*, the cited* $\mathsf{Add}^{\uparrow}$ *boundary of Definition 64.49, or the cited* $\mathsf{Order}^{\uparrow}$ *boundary of Definition 54.45.*

Conversely, each listed row is supplied by the concrete semantic exit, the checked-law audit package, the NatMul window-exhaustion theorem, and the divisibility-order source exactness theorem. Thus the boundary has no readable row for host multiplication, host order, quotient carriers, semiring primitives, or any additional arithmetic source.

Proof. Project Definition 64.93. Its displayed list is the consumer-facing row interface, so any readable row must be one of the listed carrier, classifier, continuation, multiplication, divisibility, unary-prefix, $\mathsf{Add}^{\uparrow}$, or $\mathsf{Order}^{\uparrow}$ entries.

For the converse direction, apply Definition 64.95. The handoff factors the boundary through the concrete NatMul semantic certificate and the checked-law audit package, so the BHist, UnaryHistory, hsame, Cont, and NatMul rows are available as the named result-side surface.

Next apply Definition 64.86. It exhausts the consumer-facing multiplication window, leaving no host multiplication slot and no semiring primitive outside the concrete NatMul ledger.

Finally use Definition 64.94. It identifies the remaining order-facing rows as exactly NatDivides, PreorderPrefixLE, and the cited $\mathsf{Add}^{\uparrow}$ and $\mathsf{Order}^{\uparrow}$ boundaries. Since no cited theorem introduces a quotient carrier or host order primitive, the displayed rows exhaust the audit-clean boundary. □

Theorem 64.97 (Prime divisor of a unary product exposes a factor)**.** *For histories p, a, b, n, if* $\mathsf{NatPrime}(p)$*,* $\mathsf{NatMul}(a, b, n)$*, and* $\mathsf{NatDivides}(p, n)$*, then* $\mathsf{NatDivides}(p, a)$ *or* $\mathsf{NatDivides}(p, b)$*.*

Sketch. Induct on the concrete unary multiplication witness $\mathsf{NatMul}(a, b, n)$, viewing the right factor b as the induction spine.

In the zero-product case, the result is e. A divisor witness for p into e would make e a multiple of the prime p. The non-empty condition in $\mathsf{NatPrime}(p)$, together with the empty-result inversion for NatMul, rules this case out, so the conclusion follows vacuously.

In the successor case, the product is obtained by appending one more copy of a to the previous product. If p already divides the previous product, the induction hypothesis gives $\mathsf{NatDivides}(p, a)$ or divisibility into the predecessor of b; in the latter subcase, the successor multiplication constructor and Definition 64.62 transport the divisor to $\mathsf{NatDivides}(p, b)$.

It remains when the final appended copy is the first place where the divisor can occur. The NatMul result and functionality facts identify the last continuation with the factor a up to hsame. Applying the defining divisor alternative in $\mathsf{NatPrime}(p)$ to this non-trivial divisor yields either the unit case, which is incompatible with the prime's strict-greater field, or hsame-equality with a. This gives $\mathsf{NatDivides}(p, a)$, completing the induction. □

Theorem 64.98 (Unary multiplication operation congruence)**.** *Let d, d', q, q', n, and n' be histories. If* $\mathsf{NatMul}(d, q, n)$*,* $\mathsf{hsame}(d, d')$*,* $\mathsf{hsame}(q, q')$*, and* $\mathsf{hsame}(n, n')$*, then* $\mathsf{NatMul}(d', q', n')$*.*

Moreover, the transported multiplicand d' and multiplier q' are unary histories, and every result m of $\mathsf{NatMul}(d', q', m)$ *is classified with n' by* $\mathsf{hsame}(m, n')$*. For the per-object certificate template, this statement is the concrete* MulUp *stability-obligation row for the operation relation under transported multiplicand, multiplier, and result classifiers, with its formal target intended under* BEDC.Derived.PrimeUp*.*

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul operation congruence`

Proof. Apply the multiplicand transport theorem to the given product and $\mathsf{hsame}(d, d')$. This gives unary evidence for d' and a product $\mathsf{NatMul}(d', q, n)$. Apply the multiplier transport theorem to that product and $\mathsf{hsame}(q, q')$, producing unary evidence for q' and a product $\mathsf{NatMul}(d', q', n)$.

Apply the result transport theorem to this last product and $\mathsf{hsame}(n, n')$. The result is the displayed product $\mathsf{NatMul}(d', q', n')$, with the same unary multiplicand evidence already carried by the transported product and the unary multiplier evidence obtained in the previous paragraph.

For the uniqueness clause, compare any other witness $\mathsf{NatMul}(d', q', m)$ with the transported witness $\mathsf{NatMul}(d', q', n')$ by Definition 64.12, using unary evidence for the transported multiplicand d'. This yields $\mathsf{hsame}(m, n')$. □

Theorem 64.99 (MulUp repeated-addition Cont scope)**.** *Every* $\mathsf{NatMul}(d, q, n)$ *row consumed by* $\mathsf{Mul}^{\uparrow}$ *is scoped by unary witnesses for d, q, and n, a finite* Cont *ledger that records repeated application of the* $\mathsf{Add}^{\uparrow}$ *operation over* UnaryHistory *rows, and the ambient* hsame *classifier on the*

displayed histories. The scope contains no separate multiplication operation beyond the NatMul *relation and the listed continuation reads.*

Proof. Start from the displayed $\mathsf{NatMul}(d, q, n)$ row. The construction of that row supplies unary evidence for the multiplicand and multiplier, and the functional comparison used in Definition 64.98 supplies the hsame-classification of every result row with the displayed result.

The repeated-addition ledger is the finite Cont trace that reads the current partial sum, the remaining unary counter, and the next $\mathsf{Add}^{\uparrow}$ row at each stage. Since every stage reads a UnaryHistory row and an $\mathsf{Add}^{\uparrow}$ operation row, the ledger never introduces an independent multiplication constructor.

Repack these unary witnesses, continuation reads, and classifier witnesses as the $\mathsf{Mul}^{\uparrow}$ consumption surface. Transport along hsame is exactly Definition 64.98, so the same scoped surface is preserved under replacement of d, q, and n by classified histories. □

Theorem 64.100 (MulUp unary additive distributive boundary)**.** *The right and left additive distribution surfaces cited by the public semantic exit certificate for* $\mathsf{Mul}^{\uparrow}$ *are consumed only through* UnaryHistory *rows,* NatMul *rows, finite* Cont *readbacks, the ambient* hsame *classifier, and the existing* $\mathsf{Add}^{\uparrow}$ *operation rows. No public boundary row for* $\mathsf{Mul}^{\uparrow}$ *requires a primitive multiplication operation outside that unary repeated-addition surface.*

Proof. For the right distribution surface, apply Definition 64.27. Its hypotheses are the unary inputs, the relevant NatMul rows, and the $\mathsf{Add}^{\uparrow}$ rows that combine the displayed products.

For the left distribution surface, apply Definition 64.28. The same kind of data is read, with the multiplicand and multiplier roles exchanged inside the unary ledger.

Finally combine those two surfaces with Definition 64.65. The certificate exports only the listed UnaryHistory, NatMul, Cont, hsame, and $\mathsf{Add}^{\uparrow}$ rows, while Definition 64.98 supplies transport of the operation relation across classified histories. □

64.2 The prime predicate

Definition 64.101 (Strict greater-than on unary histories)**.** *The relation* $\mathsf{NatGT}(n, m)$ *holds when* $\mathsf{NatUnaryStrictPrefix}(m, n)$ *holds (chapter 28). Equivalently,* n *extends* m *by at least one further* E1 *step.*

Definition 64.102 (Prime predicate)**.** *A unary history* p *is prime, written* $\mathsf{NatPrime}(p)$*, when:*

1. $\mathsf{Unary}(p)$*,*
2. $\mathsf{NatGT}(p, \mathsf{E1}(\mathsf{e}))$ *(i.e.,* p *strictly exceeds the unit), and*
3. *for every unary* d *with* $\mathsf{NatDivides}(d, p)$*, either* $\mathsf{hsame}(d, \mathsf{E1}(\mathsf{e}))$ *or* $\mathsf{hsame}(d, p)$*.*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatPrime`

Lemma 64.103 (The first two primes)**.** $\mathsf{NatPrime}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})))$ *and* $\mathsf{NatPrime}(\mathsf{E1}(\mathsf{E1}(\mathsf{E1}(\mathsf{e}))))$*. That is, the unary representations of* 2 *and* 3 *are prime.*

Proof. For $p = 2$, the only divisors are the unit and p itself by Definition 64.50 and the case analysis on the quotient q. The same case analysis applies to $p = 3$, with the additional sub-case ruling out $q = 2$ by the failure of $\mathsf{NatMul}(2, 2, 3)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime first pair`

Lemma 64.104 (First prime multiplied by the unit exponent)**.** *The concrete unary multiplication witness for the first prime and the unit exponent returns the first prime:*

$$\mathsf{NatMul}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{E1}(\mathsf{e}))).$$

Proof. Use the first-prime witness for $\mathsf{E1}(\mathsf{E1}(\mathsf{e}))$. The zero multiplication constructor gives the empty predecessor product, and the successor multiplication constructor appends the first prime once using the left-unit continuation. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul first prime unit result`

Theorem 64.105 (Prime successor multiplication cannot have empty result)**.** *If p is prime and $\mathsf{NatMul}(p, \mathsf{E1}(q), n)$, then n is not empty up to* hsame*.*

Proof. If $n$hsamee, then Definition 64.5 forces $p$hsamee. After this identification, the strict-prefix field in the prime predicate would be a strict unary prefix into e, which is impossible. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime NatMul succ result not empty`

Theorem 64.106 (Prime has no empty divisor)**.** *If p is prime, d divides p, and d*hsamee*, then false.*

Proof. Transport the divisor to the empty-left case. Empty-left divisibility forces $p$hsamee. Under that endpoint identification, the strict greater-than field of the prime predicate becomes a strict unary prefix into e, which is impossible. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime divisor empty absurd`

Theorem 64.107 (Prime product factors are nonempty)**.** *Let p, d, q be histories. If p is prime, d and q are unary histories, and $\mathsf{NatMul}(d, q, p)$ holds, then neither factor is history-same to* e*.*

Proof. Package the displayed multiplication witness directly as $\mathsf{NatDivides}(d, p)$ to exclude $d$hsamee by the prime no-empty-divisor theorem. For the multiplier, multiplication closure gives $\mathsf{NatDivides}(q, p)$ from the same $\mathsf{NatMul}(d, q, p)$ witness, and the same no-empty-divisor theorem excludes $q$hsamee. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime NatMul factors not empty`

Theorem 64.108 (Prime is not empty)**.** *If p is prime, then p is not history-same to* e*.*

Proof. If $p$hsamee, the strict greater-than field in the prime predicate becomes a strict unary prefix into e. This contradicts the empty-target exclusion for strict unary prefixes. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime empty absurd`

Theorem 64.109 (Prime nonunit factor has unit multiplier)**.** *Let p, d, q be histories. If p is prime, d and q are unary histories, $\mathsf{NatMul}(d, q, p)$ holds, and d is not history-same to the unit, then d is the prime factor and q is the unit multiplier:*

$$\mathsf{hsame}(d, p) \wedge \mathsf{hsame}(q, \mathsf{E1}(\mathsf{e})).$$

Proof. Package the multiplication witness as a divisibility witness $\mathsf{NatDivides}(d, p)$. The prime divisor alternative says that d is either the unit or p; the displayed nonunit hypothesis eliminates the unit case, so $d\mathsf{hsame}p$.

It remains to identify the multiplier. The empty multiplier case would make the prime result empty, contradicting the strict-prefix field of the prime predicate; the zero-headed case contradicts unary history. In the successor case, successor-result exactness for multiplication applied to $\mathsf{NatMul}(d, \mathsf{E1}(q_0), p)$ and $p\mathsf{hsame}d$ says that either d is empty or q_0 is empty. The empty factor case again contradicts primality through $d\mathsf{hsame}p$, hence $q_0\mathsf{hsame}\mathsf{e}$, so $q\mathsf{hsame}\mathsf{E1}(\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime NatMul nonunit factor multiplier unit`

Theorem 64.110 (Prime product factor is unit or prime)**.** *Let p, d, q be histories. If p is prime, d and q are unary histories, and $\mathsf{NatMul}(d, q, p)$ holds, then either d is the unit factor and q reads back to p, or d reads back to p and q is the unit multiplier:*

$$(\mathsf{hsame}(d, \mathsf{E1}(\mathsf{e})) \wedge \mathsf{hsame}(q, p)) \vee (\mathsf{hsame}(d, p) \wedge \mathsf{hsame}(q, \mathsf{E1}(\mathsf{e}))).$$

Proof. Package the multiplication witness as $\mathsf{NatDivides}(d, p)$ and apply the divisor classifier field of $\mathsf{NatPrime}(p)$. In the unit branch, unit-left multiplication identifies the result p with the multiplier q. In the prime-factor branch, the unit case would identify p with the unit and contradict primality, so the nonunit-factor theorem identifies q with the unit. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime NatMul factor unit or prime`

Theorem 64.111 (Prime divisor positive shape)**.** *If p is prime and d divides p, then d has a one-step successor shape:*

$$\mathsf{NatPrime}(p) \Longrightarrow \mathsf{NatDivides}(d, p) \Longrightarrow \exists t : \mathsf{BHist}.\ \mathsf{hsame}(d, \mathsf{E1}(t)) \wedge \mathsf{UnaryHistory}(t).$$

Proof. Read the divisor witness as a unary multiplication witness, which gives $\mathsf{UnaryHistory}(d)$. The empty case contradicts the prime no-empty-divisor theorem, and the zero-headed case contradicts unary history. The remaining one-headed case exposes the requested tail by unary inversion. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime divisor positive shape`

Theorem 64.112 (Prime divisor strict interval exclusion)**.** *Let $p, d : \mathsf{BHist}$. If $\mathsf{NatPrime}(p)$, $\mathsf{Unary}(d)$, and $\mathsf{NatDivides}(d, p)$ hold, then d cannot lie strictly between the unit and p in unary prefix order:*

$$\mathsf{NatUnaryStrictPrefix}(\mathsf{E1}(\mathsf{e}), d) \Longrightarrow \mathsf{NatUnaryStrictPrefix}(d, p) \Longrightarrow \bot.$$

Proof. Apply the divisor classifier field in $\mathsf{NatPrime}(p)$ to the displayed unary divisor d. The unit branch identifies the endpoint of the strict prefix from $\mathsf{E1}(\mathsf{e})$ to d, contradicting strict endpoint separation. The prime branch identifies the endpoint of the strict prefix from d to p, giving the same endpoint contradiction. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime NatDivides strict between absurd`

Lemma 64.113 (Non-primality of the unit)**.** $\neg\mathsf{NatPrime}(\mathsf{E1}(\mathsf{e}))$.

Proof. The unit fails the strict greater-than condition $\mathsf{NatGT}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$ by the irreflexivity of $\mathsf{NatUnaryStrictPrefix}$ (chapter 28). □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime unit absurd`

Theorem 64.114 (Prime successor tail is nonempty)**.** *If* $\mathsf{NatPrime}(p)$ *holds, then* p *has a one-step successor shape with a unary nonempty tail:*

$$\exists t : \mathsf{BHist}.\ \mathsf{hsame}(p, \mathsf{E1}(t)) \ \wedge\ \mathsf{UnaryHistory}(t) \ \wedge\ (\mathsf{hsame}(t, \mathsf{e}) \Rightarrow \bot).$$

Proof. Analyze the carrier field of the prime witness. The empty case contradicts the strict-greater-than field, and the $\mathsf{E0}$ case contradicts unary history. In the $\mathsf{E1}(t)$ case, unary inversion gives $\mathsf{UnaryHistory}(t)$. If $t\mathsf{hsamee}$, the original prime witness becomes a prime witness for the unit, contradicting the unit non-primality theorem above. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime successor tail nonempty`

Theorem 64.115 (Prime strict unit successor shape)**.** *If* $\mathsf{NatPrime}(p)$ *holds, then* p *has a two-step successor shape:*

$$\exists t : \mathsf{BHist}.\ \mathsf{hsame}(p, \mathsf{E1}(\mathsf{E1}(t))) \ \wedge\ \mathsf{UnaryHistory}(t).$$

Proof. The prime successor-tail theorem writes p as $\mathsf{E1}(t_0)$ with a unary tail t_0 that is not classified as empty. Unary nonempty shape inversion therefore writes t_0 as $\mathsf{E1}(t)$ with unary tail t, giving the displayed strict unit-successor shape. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime strict unit successor shape`

Theorem 64.116 (Prime predicate respects history-sameness)**.** *For histories* p *and* p'*, if* $\mathsf{NatPrime}(p)$ *and* $p\mathsf{hsame}p'$*, then* $\mathsf{NatPrime}(p')$*.*

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrime hsame transport`

Proof. Unpack the prime witness for p from Definition 64.102. Its carrier field gives $\mathsf{Unary}(p)$, its strict field gives $\mathsf{NatGT}(p, \mathsf{E1}(\mathsf{e}))$, and its divisor field classifies every unary divisor of p as either the unit or p itself.

By Definition 9195.7, the carrier field and $p\mathsf{hsame}p'$ give $\mathsf{Unary}(p')$. For the strict greater-than field, unfold Definition 64.101: the given field is a strict unary prefix from $\mathsf{E1}(\mathsf{e})$ to p. Apply Definition 28.24 with the reflexive endpoint $\mathsf{E1}(\mathsf{e})\mathsf{hsame}\mathsf{E1}(\mathsf{e})$ from Definition 3.8 and the displayed endpoint witness $p\mathsf{hsame}p'$. This gives $\mathsf{NatGT}(p', \mathsf{E1}(\mathsf{e}))$.

It remains to verify the divisor field for p'. Fix a unary history d with $\mathsf{NatDivides}(d, p')$. Symmetry in Definition 3.8 gives $p'\mathsf{hsame}p$, and reflexivity gives $d\mathsf{hsame}d$. Therefore Definition 64.189 transports the displayed divisor witness to $\mathsf{NatDivides}(d, p)$. Applying the divisor field of $\mathsf{NatPrime}(p)$ gives either $d\mathsf{hsame}\mathsf{E1}(\mathsf{e})$ or $d\mathsf{hsame}p$. In the first case the required unit alternative for p' is already present. In the second case, transitivity in Definition 3.8 composes $d\mathsf{hsame}p$ with $p\mathsf{hsame}p'$ and gives $d\mathsf{hsame}p'$. These are exactly the two alternatives required by Definition 64.102 for p'. □

64.3 Effective primality testing

Definition 64.117 (Trial-division witness up to b)**.** *For unary histories* b, n*, the trial-division witness* $\mathsf{TrialDiv}(b, n)$ *is the closed inductive predicate generated by:*

1. $\mathsf{TrialDiv}(\mathsf{E1}(\mathsf{e}), n)$ *for every unary* n *(the unit is examined first);*

2. *if* $\mathsf{TrialDiv}(b, n)$*, and either* $\neg\mathsf{NatDivides}(b, n)$ *or* $\mathsf{hsame}(b, \mathsf{E1}(\mathsf{e}))$ *or* $\mathsf{hsame}(b, n)$*, and* $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$*, then* $\mathsf{TrialDiv}(b', n)$ *(advance the test to* b'*).*

Lean4 (definition) `BEDC.Derived.PrimeUp.TrialDiv`

Theorem 64.118 (Trial-division step preserves unary closure)**.** *If* $\mathsf{TrialDiv}(b, n)$ *has been reached and the admissible step condition for* b *holds, then every continuation* $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$ *produces a unary* b' *and the next witness* $\mathsf{TrialDiv}(b', n)$*.*

Proof. Induct on the trial-division witness. The base witness starts at $\mathsf{E1}(\mathsf{e})$, which is unary; the step case uses unary closure of Cont with the unit continuation and rebuilds the trial-division step. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv step unary closed`

Theorem 64.119 (Trial-division bounds are unary)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then* b *is a unary history.*

Proof. Induct on the trial-division witness. The unit constructor gives $\mathsf{Unary}(\mathsf{E1}(\mathsf{e}))$. The step constructor extends the previous unary bound by the unary unit through Cont, so unary closure under continuation gives the next bound. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound unary`

Theorem 64.120 (Trial-division bounds have positive shape)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then there is a unary tail* t *such that* $b\mathsf{hsame}\mathsf{E1}(t)$*.*

Proof. Inspect the trial-division witness. The unit constructor has $b = \mathsf{E1}(\mathsf{e})$. In the step constructor, the continuation $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$ exposes b' as $\mathsf{E1}(b)$, and the previous trial-division witness supplies unary closure of the tail by Definition 64.119. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound positive shape`

Theorem 64.121 (Trial-division bound unit or successor shape)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then the bound is either the unit history* $\mathsf{E1}(\mathsf{e})$ *or has the shape* $\mathsf{E1}(\mathsf{E1}(t))$ *for a unary tail* t*.*

Proof. Apply the positive-shape theorem to write $b\mathsf{hsame}\mathsf{E1}(tail)$ with unary *tail*. Case-analyze this tail. The empty tail gives the unit branch; a zero-headed tail contradicts unary generation; a one-headed tail exposes the displayed $\mathsf{E1}(\mathsf{E1}(t))$ branch, and unary inversion supplies $\mathsf{UnaryHistory}(t)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound unit or successor shape`

Theorem 64.122 (Trial-division bounds have unit prefix or are unit)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then either* $\mathsf{E1}(\mathsf{e})$ *is a strict unary prefix of* b*, or* $b\mathsf{hsame}\mathsf{E1}(\mathsf{e})$*.*

Proof. Induct on the trial-division witness. The unit constructor gives the self branch. In the step constructor, the continuation by $\mathsf{E1}(\mathsf{e})$ supplies a strict one-step prefix. If the induction hypothesis already has a strict prefix, compose the two strict prefixes; if it has the self branch, transport the one-step prefix across that hsame endpoint. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound unit prefix or self`

Theorem 64.123 (Trial-division unit step yields strict unit prefix)**.** *If* $\mathsf{TrialDiv}(b, n)$ *and* $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$*, then* $\mathsf{E1}(\mathsf{e})$ *is a strict unary prefix of* b'*.*

Proof. The displayed continuation supplies a one-step strict prefix from b to b'. The bound classification theorem gives either a strict unit prefix of b or $b\mathsf{hsameE1}(\mathsf{e})$. In the strict-prefix branch, compose the previous strict prefix with the one-step prefix. In the unit branch, transport the one-step strict prefix across the displayed endpoint classifier. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound unit strict prefix after cont`

Theorem 64.124 (Trial-division target and bound trichotomy)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then the target* n *is unary and the current bound* b *is comparable with* n *by endpoint equality or by one of the two strict unary-prefix directions:*

$$\mathsf{UnaryHistory}(n) \wedge (b\mathsf{hsame}n \vee \mathsf{NatUnaryStrictPrefix}(b, n) \vee \mathsf{NatUnaryStrictPrefix}(n, b)).$$

Proof. Induct on the trial-division witness to recover unaryness of the fixed target. The bound is unary by Definition 64.119. Applying unary-prefix trichotomy to these two unary histories gives the displayed comparison. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv target unary and bound trichotomy`

Theorem 64.125 (Trial-division bounds are not empty)**.** *If* $\mathsf{TrialDiv}(b, n)$*, then* b *is not* e*.*

Proof. Use Definition 64.120 to write $b\mathsf{hsameE1}(t)$ for a unary tail t. An additional endpoint $b\mathsf{hsame}\mathsf{e}$ would identify $\mathsf{E1}(t)$ with e, contradicting the history no-confusion principle. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv bound not empty`

Theorem 64.126 (Trial-division unit step result is not unit)**.** *If* $\mathsf{TrialDiv}(b, n)$ *and* $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$*, then* b' *cannot be equal to the unit history* $\mathsf{E1}(\mathsf{e})$*.*

Proof. Transport the displayed continuation result across $b'\mathsf{hsameE1}(\mathsf{e})$, obtaining a continuation $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$. Compare it with the left-unit continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$ and cancel the common right unit history. This gives $b\mathsf{hsame}\mathsf{e}$, contradicting Definition 64.125. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv unit step result not unit`

Theorem 64.127 (Trial-division unit step result successor shape)**.** *If* $\mathsf{TrialDiv}(b, n)$ *and* $\mathsf{Cont}(b, \mathsf{E1}(\mathsf{e}), b')$*, then the step result has a two-step successor shape:*

$$\exists t : \mathsf{BHist}.\ b'\mathsf{hsameE1}(\mathsf{E1}(t)) \wedge \mathsf{Unary}(t).$$

Proof. The trial-division bound is unary, and continuation with the unary unit keeps the result unary. The empty and zero-headed result cases contradict the continuation inversion and unary no-zero principles. The unit result case is excluded by the preceding unit-step theorem. The remaining unary case exposes the displayed tail. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv unit step result successor shape`

Theorem 64.128 (Terminal trial division excludes non-trivial divisors)**.** *Let* n *be unary with* $\mathsf{NatGT}(n, \mathsf{E1}(\mathsf{e}))$*. If* $\mathsf{TrialDiv}(n, n)$*, then every unary* d *with* $\mathsf{NatDivides}(d, n)$ *satisfies* $d\mathsf{hsameE1}(\mathsf{e})$ *or* $d\mathsf{hsame}n$*. Consequently* $\mathsf{NatPrime}(n)$*.*

Proof. Fix a unary divisor d of n. Since $\mathsf{NatGT}(n, \mathsf{E1}(\mathsf{e}))$ supplies a nonempty endpoint, Definition 64.92 applies to the divisor witness.

If the boundary lemma gives $d\mathsf{hsame}n$, the desired divisor alternative is the right branch. It remains to handle the strict-prefix branch $\mathsf{NatUnaryStrictPrefix}(d, n)$.

Induct on the terminal witness $\mathsf{TrialDiv}(n, n)$ while transporting the strict-prefix endpoint along each unit-continuation step. The induction recovers the trial-division step at the bound represented by d. Its admissible condition is

$$\neg\mathsf{NatDivides}(d, n) \quad \text{or} \quad d\mathsf{hsameE1}(\mathsf{e}) \quad \text{or} \quad d\mathsf{hsame}n.$$

The first branch contradicts the divisor hypothesis. The last branch contradicts strict-prefix asymmetry after transporting across the equality $d\mathsf{hsame}n$. Therefore the unit branch remains, so $d\mathsf{hsameE1}(\mathsf{e})$.

The prime predicate now follows by unfolding Definition 64.102: unaryness and strict greater-than are the hypotheses, and the preceding argument supplies the required alternative for every unary divisor. □

Lean4 ✓ `BEDC.Derived.PrimeUp.TrialDiv terminal excludes nontrivial divisors`

Theorem 64.129 (Decidability of primality)**.** *For every unary history n, the proposition $\mathsf{NatPrime}(n)$ is decidable: there is an explicit unary algorithm that returns either a proof of $\mathsf{NatPrime}(n)$ or an explicit non-trivial divisor of n.*

Sketch. The algorithm constructs $\mathsf{TrialDiv}(b, n)$ by unary induction on b up to n itself. At each step, it checks divisibility of n by b using Definition 64.17 and Definition 64.12: this is decidable because the search for a quotient q of $\mathsf{NatMul}(b, q, n)$ is bounded above by n.

If a non-trivial divisor is found, meaning b divides n and is neither the unit nor n, the algorithm returns it and refutes primality. Otherwise, $\mathsf{TrialDiv}$ reaches $b = n$, and Definition 64.128 gives $\mathsf{NatPrime}(n)$. □

Lemma 64.130 (Unit right multiplication reads back the multiplicand)**.** *For every unary history d, if $\mathsf{NatMul}(d, \mathsf{E1}(\mathsf{e}), n)$, then $\mathsf{hsame}(n, d)$.*

Proof. Apply Definition 64.2 to the multiplication witness. This gives a history m with $\mathsf{NatMul}(d, \mathsf{e}, m)$ and $\mathsf{Cont}(m, d, n)$. Inverting the zero-multiplier multiplication witness leaves only the zero constructor, so m is identified with e, and the displayed continuation becomes $\mathsf{Cont}(\mathsf{e}, d, n)$. The left-unit existence part of Definition 21.12 gives $\mathsf{Cont}(\mathsf{e}, d, d)$, and continuation determinacy from Definition 21.7 identifies n with d. □

Lemma 64.131 (Unit left multiplication is closed)**.** *For every unary history q,*

$$\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q, q).$$

Proof. Induct on the unary generation of q. The empty case is the zero multiplication constructor with the unary unit multiplicand. The zero-headed case is impossible by unary-history inversion. In the one-headed case, the induction hypothesis gives the predecessor product $\mathsf{NatMul}(\mathsf{E1}(\mathsf{e}), q_0, q_0)$, and the successor constructor appends one unit copy by the definitional continuation $\mathsf{Cont}(q_0, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(q_0))$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul unit left closed`

Lemma 64.132 (Unary division by a nonempty divisor)**.** *Let t and x be unary histories with $t \not\mathsf{hsamee}$. Then there are unary histories q, r, and w such that* $\mathsf{NatMul}(t, q, w)$, $\mathsf{Cont}(w, r, x)$, *and either* $r\mathsf{hsamee}$ *or* $\mathsf{NatUnaryStrictPrefix}(r, t)$.

Proof. Run unary induction on x, maintaining the invariant that x is the continuation of a t-multiple w by a remainder r which is empty or a strict prefix of t. The empty case uses $q = \mathsf{e}$, $w = \mathsf{e}$, and $r = \mathsf{e}$. For a one-extension step, first extend the current remainder by the unit history. Apply unary prefix totality from Definition 28.5 to this candidate remainder and t. If the candidate is still a strict prefix of t, keep the same quotient and replace the remainder by the candidate. If the candidate reaches t, append one more copy of t to the multiple using the successor constructor of Definition 64.1, reset the remainder to e, and use continuation associativity from Definition 29.6. The alternative in which t is a strict prefix of the candidate would make the old remainder and t strict prefixes of each other after one-step cancellation, contradicting Definition 28.14. Thus the invariant closes. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul division with remainder`

Lemma 64.133 (Unary multiplication commutes up to history-sameness)**.** *Let a and b be unary histories. If* $\mathsf{NatMul}(a, b, n)$ *and* $\mathsf{NatMul}(b, a, m)$, *then* $n\mathsf{hsame}m$.

Proof. Induct on the unary generation of b. The empty case reduces both products to e by constructor inversion and Definition 64.4. In the one-extension case, invert the first product with Definition 64.2. The predecessor is the product of a by the tail of b; by the induction hypothesis it agrees with the product of that tail by a. The final continuation appends one copy of a. On the other side, multiplication of $\mathsf{E1}(b_0)$ by a is built by appending one visible unit through each copy of a. Repeated use of unary continuation commutativity from Definition 9200.14 and associativity from Definition 29.6 moves that visible unit block past the tail product. Functionality of unary multiplication from Definition 64.12 identifies the two resulting histories. The $\mathsf{E0}$ case is impossible by Definition 28.3. □

Lemma 64.134 (Divisible prefix cancellation)**.** *Let d, x, y, and z be unary histories with d /* hsamee. *If* $\mathsf{Cont}(x, y, z)$, $\mathsf{NatDivides}(d, x)$, *and* $\mathsf{NatDivides}(d, z)$, *then* $\mathsf{NatDivides}(d, y)$.

Proof. Unpack the two divisibility witnesses from Definition 64.50, obtaining quotients q_x and q_z with products $dq_x = x$ and $dq_z = z$. Compare q_x and q_z by unary prefix totality. If q_z is obtained from q_x by a unary tail u, then Definition 64.1 and continuation associativity decompose the dq_z product as the continuation of x by the product du. Since z is also the continuation of x by y, continuation left-cancellation identifies y with du, and divisibility follows after endpoint transport by Definition 64.189.

In the other comparison direction, x is the continuation of z by a d-multiple. If that extra quotient tail is nonempty, the nonempty divisor d makes the extra product nonempty by Definition 64.38; then z is a strict unary prefix of x, while the displayed $\mathsf{Cont}(x, y, z)$ makes x a prefix of z unless $y\mathsf{hsamee}$. Strict-prefix asymmetry from Definition 28.14 rules out the nonempty tail. Hence the tail is empty, $x\mathsf{hsame}z$, and continuation unit uniqueness forces $y\mathsf{hsamee}$. The empty quotient witnesses $\mathsf{NatDivides}(d, y)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides prefix cancellation`

Theorem 64.135 (Divisible continuation left factor)**.** *Let d, x, y, and z be unary histories with $d \not\mathsf{hsamee}$. If* $\mathsf{NatDivides}(d, y)$, $\mathsf{NatDivides}(d, z)$, *and* $\mathsf{Cont}(x, y, z)$, *then* $\mathsf{NatDivides}(d, x)$.

Proof. The unary continuation ledger may be read in the opposite endpoint order by unary append commutativity, giving $\mathsf{Cont}(y, x, z)$. Applying divisible prefix cancellation to this commuted continuation removes the already-divisible factor y from the divisible result z, leaving divisibility of x by d. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides cont left factor`

Theorem 64.136 (Divisible continuation right factor iff)**.** *Let d, x, y, and z be unary histories with $d \not\mathsf{hsame}$. If $\mathsf{NatDivides}(d, x)$ and $\mathsf{Cont}(x, y, z)$, then divisibility of the continuation result is equivalent to divisibility of the displayed right factor:*

$$\mathsf{NatDivides}(d, z) \iff \mathsf{NatDivides}(d, y).$$

Proof. For the forward direction, apply divisible prefix cancellation to remove the already-divisible left factor x from the divisible result z. For the reverse direction, unpack the quotient witnesses for $\mathsf{NatDivides}(d, x)$ and $\mathsf{NatDivides}(d, y)$; appending the two quotients and using multiplication append closure builds a quotient witness for the displayed continuation result z. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides cont right factor iff`

Theorem 64.137 (Prime divisor dichotomy)**.** *For unary histories p and a, if $\mathsf{NatPrime}(p)$, then*

$$\mathsf{NatDivides}(p, a) \vee \neg\mathsf{NatDivides}(p, a).$$

Sketch. Use the $\mathsf{Prime}^{\uparrow}$ template with traditional target divisibility decidability by a fixed prime. The BEDC source data are the unary histories together with NatMul, $\mathsf{NatDivides}$, and $\mathsf{NatPrime}$.

The carrier is the bounded unary-prefix search over quotient candidates up to a. For each candidate q, Definition 64.17 supplies a product m_q for $\mathsf{NatMul}(p, q, m_q)$, and the classifier compares m_q with a by hsame.

Stability follows from Definition 64.17, Definition 64.12, Definition 28.5, and endpoint transport. Thus the bounded quotient scan is a ledger whose entries are stable under history-sameness of the multiplication result.

Exactness is immediate from the scan. If some candidate has product history-same to a, endpoint transport packages that quotient as $\mathsf{NatDivides}(p, a)$. If the scan exhausts every unary prefix up to a without such an endpoint, then every possible quotient is rejected, which is the negated divisor predicate. This supplies the displayed disjunction as a paper certificate. □

Lemma 64.138 (Least prime multiple period)**.** *Let p and a be unary histories with $\mathsf{NatPrime}(p)$ and $\neg\mathsf{NatDivides}(p, a)$. Among the nonempty unary histories t for which there is a product m_t with $\mathsf{NatMul}(a, t, m_t)$ and $\mathsf{NatDivides}(p, m_t)$, a strict-prefix least such t exists, and every such least t satisfies $t\mathsf{hsame}p$.*

Proof. The search set is inhabited. By Definition 64.17 there is a product of a by p; by Definition 64.133 it is history-same to the product of p by a, so Definition 64.189 gives divisibility by p. The strict-greater field in Definition 64.102 makes p nonempty. Bounded trial division as in Definition 64.129, using Definition 64.17 and Definition 64.12 to decide each product test, scans the nonempty prefixes up to p and returns a least period t.

Divide p by this nonempty t using Definition 64.132. Thus p is the continuation of a t-multiple by a remainder r, where r is empty or a strict prefix of t. Since p divides ap, and since p also divides a times the t-multiple part by repeated use of the period witness and Definition 64.62,

Definition 64.134 gives divisibility of ar by p. If r were nonempty, this would be a smaller positive period than t, contradicting leastness. Hence $r\mathsf{hsamee}$, so t divides p.

Apply the divisor-classifier field of Definition 64.102 to this divisor t of p. Either $t\mathsf{hsameE1}(\mathsf{e})$ or $t\mathsf{hsame}p$. In the unit case, the period witness is a product of a by the unit; Definition 64.130 and endpoint transport give $\mathsf{NatDivides}(p, a)$, contradicting the hypothesis. Therefore $t\mathsf{hsame}p$. □

Theorem 64.139 (Euclid's lemma for prime divisibility)**.** *Let p, a, b, and n be unary histories. If $\mathsf{NatPrime}(p)$, $\mathsf{NatMul}(a, b, n)$, and $\mathsf{NatDivides}(p, n)$, then*

$$\mathsf{NatDivides}(p, a) \vee \mathsf{NatDivides}(p, b).$$

Proof. Apply Definition 64.137 to p, a, and the prime hypothesis. In the left branch, take the left disjunct.

In the negated branch, let t be the least prime multiple period supplied by Definition 64.138; then $t\mathsf{hsame}p$.

Use Definition 64.132 to divide b by p. Write b as the continuation of a p-multiple part by a remainder r, where $r\mathsf{hsamee}$ or r is a strict unary prefix of p. The product of a with the p-multiple part is divisible by p, because the product of a with one copy of p is divisible by p by Definition 64.133, and repeated copies compose by Definition 64.62. The given product n is obtained from this divisible prefix by appending the product of a with r, using the successor clauses of Definition 64.1 and continuation associativity. Since p divides n, Definition 64.134 gives that p divides the product of a with r.

If r is nonempty, then r is a positive period for a modulo p and is a strict prefix of p, contradicting $t\mathsf{hsame}p$ and the leastness of t. Thus $r\mathsf{hsamee}$. The displayed division of b therefore has no remainder: b is history-same to a product with left factor p. By Definition 64.189, this is $\mathsf{NatDivides}(p, b)$, giving the right disjunct. □

64.4 Euclid's theorem on the infinitude of primes

Definition 64.140 (Unary factorial)**.** *The unary factorial $\mathsf{NatFact}(n, n!)$ is the closed inductive predicate generated by:*

1. $\mathsf{NatFact}(\mathsf{e}, \mathsf{E1}(\mathsf{e}))$ *(zero factorial is one);*
2. *if $\mathsf{NatFact}(n, m)$ and $\mathsf{NatMul}(\mathsf{E1}(n), m, m')$, then $\mathsf{NatFact}(\mathsf{E1}(n), m')$.*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatFact`

Lemma 64.141 (Unary factorial inputs are unary)**.** *Every closed witness of $\mathsf{NatFact}(n, m)$ has a unary input n.*

Proof. Induct on the $\mathsf{NatFact}$ witness. The zero constructor has input e, which is unary. The successor constructor changes the input from n to $\mathsf{E1}(n)$, so unary closure under $\mathsf{E1}$ applied to the induction hypothesis gives the result. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact input unary`

Lemma 64.142 (Unary factorial successor inversion)**.** *If $\mathsf{NatFact}(\mathsf{E1}(n), m')$, then there is an m such that $\mathsf{NatFact}(n, m)$ and $\mathsf{NatMul}(\mathsf{E1}(n), m, m')$.*

Proof. Invert the closed factorial witness. The zero constructor has empty input and therefore cannot produce an $\mathsf{E1}$ input. The successor constructor exposes exactly the predecessor factorial witness and the multiplication witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact successor inversion`

Theorem 64.143 (Factorial successor input divides its result)**.** *If* $\mathsf{NatFact}(\mathsf{E1}(n), m)$*, then the successor input* $\mathsf{E1}(n)$ *divides the factorial result* m*:*

$$\mathsf{NatFact}(\mathsf{E1}(n), m) \Longrightarrow \mathsf{NatDivides}(\mathsf{E1}(n), m).$$

Proof. Successor inversion exposes a predecessor factorial value m_0 together with a multiplication witness $\mathsf{NatMul}(\mathsf{E1}(n), m_0, m)$. The predecessor factorial result is unary by factorial result closure. Thus m_0 is a valid quotient witness in the existential definition of $\mathsf{NatDivides}(\mathsf{E1}(n), m)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact successor input divides result`

Theorem 64.144 (Factorial successor step predecessor divides product)**.** *If* $\mathsf{NatFact}(n, m)$ *and* $\mathsf{NatMul}(\mathsf{E1}(n), m, m')$*, then the predecessor factorial value divides the displayed successor-step product:*

$$\mathsf{NatDivides}(m, m').$$

Proof. The factorial witness supplies unary evidence for n and for the predecessor factorial result m. Thus the displayed multiplication step has two unary inputs, and right-factor divisibility for unary multiplication applies directly to the factor m. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact successor step predecessor divides`

Theorem 64.145 (Factorial absorbs every positive bounded unary predecessor)**.** *Let* n, m, d *be histories. If* $\mathsf{NatFact}(n, m)$*,* $\mathsf{UnaryHistory}(d)$*,* $\mathsf{NatGT}(d, \mathsf{e})$*, and either* $d\mathsf{hsame}n$ *or* $\mathsf{NatUnaryStrictPrefix}(d, n)$*, then* $\mathsf{NatDivides}(d, m)$*.*

Sketch. Induct on the closed $\mathsf{NatFact}$ witness. In the empty-input case, positive boundedness would require either $d\mathsf{hsame}\mathsf{e}$ or a strict unary prefix of e, contradicting $\mathsf{NatGT}(d, \mathsf{e})$ and strict-prefix inversion.

For the successor step, write the input as $\mathsf{E1}(n_0)$ and the result as the product of $\mathsf{E1}(n_0)$ with the predecessor factorial result. If $d\mathsf{hsame}\mathsf{E1}(n_0)$, endpoint transport turns Definition 64.143 into the required divisibility statement for d.

It remains to treat the strict-predecessor branch. Invert the $\mathsf{NatUnaryStrictPrefix}(d, \mathsf{E1}(n_0))$ witness. The endpoint case is the previous paragraph, while the proper predecessor case gives either $d\mathsf{hsame}n_0$ or $\mathsf{NatUnaryStrictPrefix}(d, n_0)$. The induction hypothesis therefore gives divisibility of the predecessor factorial result by d.

Finally, push that divisibility through the final multiplication witness. Use the left-closure of $\mathsf{NatDivides}$ under multiplication, and transport along hsame endpoints where the constructor inversion changed the displayed representatives. Transitivity of $\mathsf{NatDivides}$ yields $\mathsf{NatDivides}(d, m)$ for the successor factorial result. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact bounded positive predecessor divides`

Lemma 64.146 (Unary factorial results are nonempty)**.** *Every closed witness of* $\mathsf{NatFact}(n, m)$ *has a result* m *distinct from* e *up to* hsame*.*

Proof. Induct on the $\mathsf{NatFact}$ witness. The zero case is the singleton $\mathsf{E1}(\mathsf{e})$, which cannot classify as empty. In the successor case, an empty product result would force the previous factorial result to be empty by the nonempty-multiplicand empty-result inversion for NatMul, contradicting the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact result not empty`

Lemma 64.147 (Factorial successor results have positive shape)**.** *If* $\mathsf{NatFact}(\mathsf{E1}(n), m)$*, then the factorial result has one-headed shape:*

$$\exists t,\ \mathsf{hsame}(m, \mathsf{E1}(t)) \wedge \mathsf{UnaryHistory}(t).$$

Proof. Successor inversion exposes a predecessor factorial value m_0 and a multiplication witness $\mathsf{NatMul}(\mathsf{E1}(n), m_0, m)$. Factorial non-emptiness rules out $m_0\mathsf{hsamee}$, and $\mathsf{E1}(n)$ is not history-same to e. The positive-shape theorem for multiplication with a nonempty multiplier therefore gives the displayed one-headed result. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact successor result positive shape`

Lemma 64.148 (Factorial results are unary)**.** *If* $\mathsf{NatFact}(n, m)$*, then* m *is a unary history.*

Proof. Induct on the factorial witness. The base constructor yields $\mathsf{E1}(\mathsf{e})$, which is unary. The successor constructor carries a multiplication witness, whose left factor is unary by multiplicand readback and whose result is unary by multiplication result closure. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact result unary`

Theorem 64.149 (Unary factorial zero-headed component impossibility)**.** *For histories* n, m*, a closed witness of* $\mathsf{NatFact}(n, m)$ *cannot have either its input or result displayed as a zero-headed history.*

Proof. Factorial input readback gives $\mathsf{UnaryHistory}(n)$, and factorial result readback gives $\mathsf{UnaryHistory}(m)$. In either displayed zero-headed branch, substituting the equality turns a zero-headed history into a unary history, contradicting unary zero-extension separation. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact zero headed component absurd`

Theorem 64.150 (Factorial results have positive shape)**.** *If* $\mathsf{NatFact}(n, m)$*, then there is a unary tail* t *such that* $\mathsf{hsame}(m, \mathsf{E1}(t))$*.*

Proof. Combine factorial result nonemptiness with factorial result unary closure. Case analysis on the generated history m eliminates the empty case by nonemptiness and the $\mathsf{E0}$ case by unary inversion. The remaining $\mathsf{E1}(t)$ case exposes the tail t, and unary inversion supplies the tail carrier. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact result positive shape`

Lemma 64.151 (Factorial is total and functional on unary histories)**.** *For every unary* n*, there is exactly one unary* m *with* $\mathsf{NatFact}(n, m)$*.*

Proof. By unary induction on n, using the totality and functionality of NatMul (Definition 64.17, Definition 64.12) at each step. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact total functional`

Lemma 64.152 (Bounded unary divisor divides factorial result)**.** *Let* n*,* m*, and* d *be histories. If* $\mathsf{NatFact}(n, m)$*,* $\mathsf{UnaryHistory}(d)$*,* $\neg\mathsf{hsame}(d, \mathsf{E1}(\mathsf{e}))$*, and* $\mathsf{NatGT}(n, d) \vee \mathsf{hsame}(d, n)$*, then* $\mathsf{NatDivides}(d, m)$*.*

Sketch. Induct on the factorial witness. In the base case, the bound by e gives no nonunit unary divisor, so the hypotheses cannot all hold.

For a successor factorial witness, use Definition 64.142 to expose the predecessor factorial and the displayed multiplication step. If $d\mathsf{hsame}n$ at the successor endpoint, endpoint transport and Definition 64.143 give $\mathsf{NatDivides}(d, m)$.

If d is strictly below the successor endpoint, successor inversion of the bound either identifies the endpoint case above or produces a predecessor bound. The induction hypothesis gives divisibility into the predecessor factorial result. Then Definition 64.17, Definition 64.12, and the displayed product divisibility closures Definition 64.89 and Definition 64.90 carry the inherited divisor through the successor multiplication display. □

Lemma 64.153 (Factorial successor has no bounded nonunit divisor)**.** *Let n, m, and d be histories. If* $\mathsf{NatFact}(n, m)$, $\mathsf{UnaryHistory}(d)$, $\neg\mathsf{hsame}(d, \mathsf{E1}(\mathsf{e}))$, *and* $\mathsf{NatGT}(n, d) \vee \mathsf{hsame}(d, n)$, *then* $\neg\mathsf{NatDivides}(d, \mathsf{E1}(m))$.

Sketch. Assume toward contradiction that d divides $\mathsf{E1}(m)$. By Definition 64.152, the same bounded nonunit divisor also divides the factorial result m.

The continuation $\mathsf{Cont}(m, \mathsf{E1}(\mathsf{e}), \mathsf{E1}(m))$ displays the successor as the factorial result followed by the unit. Divisible prefix cancellation Definition 64.134 removes the common divisible prefix and forces $\mathsf{NatDivides}(d, \mathsf{E1}(\mathsf{e}))$.

Finally, unit-right divisibility exactness Definition 64.60 identifies d with $\mathsf{E1}(\mathsf{e})$, contradicting the nonunit hypothesis. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatFact successor no bounded nonunit divisor`

Lemma 64.154 (Successor of factorial is coprime to all bounded predecessors)**.** *Let m be the unary factorial of n, and let m' be the successor* $\mathsf{E1}(m)$ *(i.e., $n! + 1$). Then for every unary d with* $\mathsf{E1}(\mathsf{e}) \neq d$ *and* $\mathsf{NatGT}(\mathsf{E1}(n), d) \vee \mathsf{hsame}(d, \mathsf{E1}(n)) \vee \mathsf{NatGT}(d, \mathsf{E1}(n)) \wedge \mathsf{NatDivides}(d, n)$, *we have* $\neg\mathsf{NatDivides}(d, m')$.

Sketch. For every positive unary d bounded by n, the factorial absorption theorem Definition 64.145 supplies $\mathsf{NatDivides}(d, m)$. If the same d also divides $m' = \mathsf{E1}(m)$, then the unit-successor subtraction argument shows that d divides the unit.

Unit divisibility forces $d\mathsf{hsame}\mathsf{E1}(\mathsf{e})$, contradicting the displayed non-unit hypothesis. Thus no such bounded non-unit divisor can divide m'. □

Theorem 64.155 (Euclid's theorem, constructive form)**.** *There exists an explicit total unary function* $\mathsf{NatNextPrime} : \mathsf{Unary} \to \mathsf{Unary}$ *such that for every unary n,* $\mathsf{NatPrime}(\mathsf{NatNextPrime}(n))$ *and* $\mathsf{NatGT}(\mathsf{NatNextPrime}(n), n)$.

Sketch. Given n, compute the unary factorial $m = n!$ by Definition 64.151, then form $m' = m + 1$. By Definition 64.154, every prime divisor of m' exceeds n. The primality test of Definition 64.129 either finds m' itself prime (return it) or returns an explicit non-trivial divisor; in the latter case, recursing on the divisor's prime factor (also exceeding n) terminates in a prime by well-foundedness of $\mathsf{NatUnaryStrictPrefix}$. The function is total because the search is bounded. □

64.5 The fundamental theorem of arithmetic

Definition 64.156 (Prime factorization list)**.** *A prime factorization of a unary history n is a finite list of unary primes $[p_1, \ldots, p_k]$ (in non-decreasing unary order) such that the iterated multiplicative continuation $p_1 \cdot p_2 \cdots p_k$ yields a unary history* hsame*-equal to n.*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatPrimeFactorization`
Lean4 (definition) `BEDC.Derived.PrimeUp.PrimeFactorizationProduct`
Lean4 (definition) `BEDC.Derived.PrimeUp.PrimeFactorization`

Theorem 64.157 (Prime factorization cons head divides)**.** *If a prime factorization list of n has head p, then p is prime and divides n:*

$$\mathsf{PrimeFactorization}(n, p :: ps) \Longrightarrow \mathsf{NatPrime}(p) \wedge \mathsf{NatDivides}(p, n).$$

Proof. Unfold the cons clause of the concrete factorization-product predicate. It exposes the head primality proof, a tail product, and the multiplication witness $\mathsf{NatMul}(p, \mathsf{tail}, n)$. The multiplier unary field of that witness supplies the quotient carrier required by $\mathsf{NatDivides}$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeFactorization cons head divides`

Theorem 64.158 (Prime factorization product result transport)**.** *If a prime factorization product yields n, and $n\mathsf{hsame}n'$, then the same factorization product yields n'.*

Proof. Inspect the concrete list-product predicate. The empty product clause is transported by transitivity of hsame. In the cons clause, the tail product is unchanged, and the displayed NatMul result witness is transported along $n\mathsf{hsame}n'$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeFactorizationProduct result hsame transport`

Theorem 64.159 (Prime factorization products have unary result)**.** *If* $\mathsf{PrimeFactorizationProduct}(ps, n)$*, then n is a unary history.*

Proof. Induct on the factor list. The empty list identifies the product with the unit endpoint $\mathsf{E1}(\mathsf{e})$, which is unary, and unary carrier transport moves this along hsame. In the cons case, the head prime supplies a unary multiplicand and the displayed NatMul witness supplies unary closure of the result. □

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeFactorizationProduct result unary`

Theorem 64.160 (Prime factorization products have nonempty result)**.** *If* $\mathsf{PrimeFactorizationProduct}(ps, n)$*, then n is not history-same to* e*.*

Proof. Induct on the list ps. The empty product is $\mathsf{E1}(\mathsf{e})$, which is separated from e by constructor no-confusion. In the cons case, the head prime is nonempty, the induction hypothesis makes the tail product nonempty, and the nonempty-factor multiplication theorem applies to the displayed NatMul witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeFactorizationProduct result not empty`

Definition 64.161 (Natural prime product relation)**.** *The relation* $\mathsf{NatPrimeProduct}(ps, n)$ *is the closed list product relation generated by the empty product* $\mathsf{NatPrimeProduct}([], \mathsf{E1}(\mathsf{e}))$*, and by the cons rule: if* $\mathsf{NatPrime}(p)$*,* $\mathsf{NatPrimeProduct}(ps, m)$*, and* $\mathsf{NatMul}(p, m, n)$*, then* $\mathsf{NatPrimeProduct}(p :: ps, n)$*.*

Lean4 (definition) `BEDC.Derived.PrimeUp.NatPrimeProduct`

Theorem 64.162 (Natural prime products have nonempty result)**.** *If* $\mathsf{NatPrimeProduct}(ps, n)$*, then n is not history-same to* e*.*

Proof. Induct on the list-product witness. The empty product is $\mathsf{E1}(\mathsf{e})$, so history no-confusion separates it from e. In the cons case, primality exposes the head factor as a nonempty one-headed unary history, the induction hypothesis gives nonemptiness of the tail product, and the nonempty-factor multiplication result theorem applies to the displayed NatMul witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrimeProduct result not empty`

Theorem 64.163 (Natural prime factorizations have nonempty result)**.** *If* $\mathsf{NatPrimeFactorization}(ps, n)$*, then* n *is not history-same to* e*.*

Proof. Induct on the concrete factorization witness. The unit factorization has result $\mathsf{E1}(\mathsf{e})$, separated from e by constructor no-confusion. The singleton case is prime nonemptiness. In the cons case, primality makes the head factor nonempty, the induction hypothesis makes the tail product nonempty, and nonempty-factor multiplication excludes an empty result. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrimeFactorization result not empty`

Theorem 64.164 (Natural prime products have unary result)**.** *If* $\mathsf{NatPrimeProduct}(ps, n)$*, then* n *is a unary history.*

Proof. Induct on the list-product witness. The empty product is $\mathsf{E1}(\mathsf{e})$, hence unary by unary successor closure over the empty history. In the cons case, primality supplies the unary carrier of the head factor, and the multiplication-result carrier theorem applies to the displayed NatMul witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrimeProduct result unary`

Theorem 64.165 (Natural prime product cons tail divides)**.** *If* $\mathsf{NatPrimeProduct}(p :: ps, n)$*, then* p *is unary and there is a tail product* m *such that* $\mathsf{NatPrimeProduct}(ps, m)$ *and* $\mathsf{NatDivides}(m, n)$*.*

Proof. Invert the cons constructor of $\mathsf{NatPrimeProduct}$. The head prime gives the unary carrier of p, while the tail witness gives the product m. The preceding unary-result theorem supplies the unary carrier of m, and the multiplication closure theorem turns the displayed $\mathsf{NatMul}(p, m, n)$ witness into $\mathsf{NatDivides}(m, n)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrimeProduct cons tail divides`

Theorem 64.166 (Natural prime product member factor divides result)**.** *If* $p \in ps$ *and* $\mathsf{NatPrimeProduct}(ps, n)$*, then* p *is unary and* $\mathsf{NatDivides}(p, n)$*.*

Proof. Induct on the $\mathsf{NatPrimeProduct}$ witness. The empty product has no list member. In the cons case, head membership is witnessed directly by the displayed $\mathsf{NatMul}(p, m, n)$ relation and the unary carrier of the tail product. Tail membership applies the induction hypothesis to the tail product and composes the resulting divisibility through the head multiplication closure theorem. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatPrimeProduct factor mem unary divides result`

Theorem 64.167 (Existence of prime factorization)**.** *For every unary history* n *with* $\mathsf{NatGT}(n, \mathsf{e})$*, there exists a prime factorization list of* n*. The list is constructed explicitly by repeated trial division using Definition 64.129.*

Sketch. Strong induction on the unary history n along NatUnaryStrictPrefix. If $n = \mathsf{E1}(\mathsf{e})$, the empty list factors it. Otherwise apply Definition 64.129: either n is prime (return $[n]$) or a non-trivial divisor d is exhibited; recurse on d and on the quotient $q = n/d$, both strictly smaller than n, and concatenate the resulting lists. □

Theorem 64.168 (Uniqueness of prime factorization)**.** *If $[p_1, \ldots, p_j]$ and $[q_1, \ldots, q_k]$ are two prime factorization lists of the same unary history n, then $j = k$ and the lists agree pointwise up to* hsame*.*

Sketch. Apply Definition 64.97 to the leading prime of one factorization and the iterated product represented by the other factorization. Repeating this product-factor exposure across the second list locates a matching prime factor up to hsame.

Cancel the matched factor using the NatMul result and functionality facts, then continue by induction on the remaining lists. The induction gives pointwise agreement up to hsame and length agreement $j = k$. □

64.6 The certificate

Definition 64.169 (Prime source specification)**.** *The source specification on* Unary *carries the additional constraint* $\mathsf{NatPrime}(p)$*. Its generator is the prime predicate of Definition 64.102.*

Definition 64.170 (Prime pattern specification)**.** *The pattern of* $\mathsf{Prime}^{\uparrow}$ *is trial-division-bounded primality: for an input unary n, the pattern is the trial-division predicate* $\mathsf{TrialDiv}(n, n)$ *restricted to inputs satisfying* NatPrime *at the upper bound.*

Definition 64.171 (Prime classifier specification)**.** *The classifier on prime carriers is* hsame *restricted to unary primes. Two unary primes classify together iff they are* hsame*-equal as histories.*

Definition 64.172 (Prime stability certificate)**.** *The stability certificate consists of:*

1. *closure of the prime predicate under* hsame*: if* $\mathsf{NatPrime}(p)$ *and* $\mathsf{hsame}(p, p')$*, then* $\mathsf{NatPrime}(p')$*;*
2. *transport of divisibility under* hsame*: if* $\mathsf{NatDivides}(d, n)$*,* $\mathsf{hsame}(d, d')$*, and* $\mathsf{hsame}(n, n')$*, then* $\mathsf{NatDivides}(d', n')$*;*
3. *compatibility of the trial-division witness with* hsame *on bounds.*

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeStabilityCert fields`

Definition 64.173 (Prime ledger policy)**.** *The ledger policy records:*

1. *composition of factorizations: concatenation of prime factorization lists for products;*
2. *product-factor exposure by Definition 64.97;*
3. *infinitude closure:* NatNextPrime *of Definition 64.155 produces a new prime exceeding any given bound.*

Theorem 64.174 (Concrete $\mathsf{Prime}^{\uparrow}$ naming certificate)**.** *The data* (NatPrime, TrialDiv, hsame, PrimeStabilityCert, Pr *of Definition 64.169–Definition 64.173 forms a naming certificate over* Hist *for the prime number interface* $\mathsf{Prime}^{\uparrow}$*. The certificate is decidable (Definition 64.129), unbounded (Definition 64.155), and exhaustive (Definition 64.167, Definition 64.168).*

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). Decidability is Definition 64.129. Unboundedness is Definition 64.155. Exhaustiveness follows from existence (Definition 64.167) together with uniqueness (Definition 64.168). The certificate uses no Classical.choice, no Quot.sound, and no propext, so its Lean targets fall under the project's axiom-purity gate. □

Lean4 ✓ `BEDC.Derived.PrimeUp.PrimeUp StdBridge`

Theorem 64.175 (Semantic $\mathsf{Prime}^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 64.174 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9): the carrier is a* Hist*-predicate, the classifier is a restriction of* hsame*, and every closure law is given by an explicit constructive proof.*

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. □

Lean4 ✓ `BEDC.Derived.PrimeUp.prime semantic name certificate`

Theorem 64.176 (Unary multiplication result semantic name certificate)**.** *For unary histories d and q, the result predicate* $\mathsf{NatMul}(d, q, -)$*, used as source, pattern, and ledger policy, together with* hsame *as classifier, forms a semantic naming certificate.*

Proof. Totality of unary multiplication supplies a carried result witness. The classifier laws are exactly reflexivity, symmetry, and transitivity of hsame. Carrier transport is result transport for NatMul along history-sameness, while the pattern and ledger maps are the shared concrete result predicate. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul result semanticNameCert`

Theorem 64.177 (Divisibility semantic name certificate)**.** *For a unary divisor d, the predicate* $\mathsf{NatDivides}(d, -)$*, used as source, pattern, and ledger policy, together with* hsame *as classifier, forms a semantic naming certificate.*

Proof. The empty dividend is inhabited exactly by the empty-right divisibility characterization. The classifier laws are inherited from hsame, and carrier transport is dividend transport for NatDivides along history-sameness. Since the source, pattern, and ledger predicates coincide, the semantic soundness fields are direct projections of the same witness. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides semanticNameCert`

Lemma 64.178 (Empty left factor has empty unary product)**.** *If* $\mathsf{NatMul}(\mathsf{e}, q, n)$*, then* $n\mathsf{hsamee}$*.*

Proof. Induct on the multiplication witness from Definition 64.1. The zero clause has result e. In the successor clause, the inductive result identifies the predecessor product with e, and the final continuation has empty right input. By the continuation unit law of Definition 29.5, the successor result is history-same to that predecessor, hence to e. □

Lemma 64.179 (Nonempty quotient exposes a multiplicand extension)**.** *Let d and q be unary histories. If* $\mathsf{NatMul}(d, q, n)$ *and* $q \not\mathsf{hsamee}$*, then there is a unary history t with* $\mathsf{Cont}(d, t, n)$*.*

Proof. Induct on the unary generation of q from Definition 28.1. The empty case contradicts $q \not\mathsf{hsame}\mathsf{e}$, and the zero-headed case is impossible by Definition 28.3. In the one-headed case write $q = \mathsf{E1}(q_0)$. By Definition 64.2, the multiplication witness has a predecessor product p with $\mathsf{NatMul}(d, q_0, p)$ and $\mathsf{Cont}(p, d, n)$.

If $q_0 \mathsf{hsame}\mathsf{e}$, the zero clause of Definition 64.1 identifies p with e. The displayed continuation is therefore the left-unit continuation from e by d, so $n \mathsf{hsame} d$; transporting the right-unit continuation of d across this identification gives $\mathsf{Cont}(d, \mathsf{e}, n)$, with unary tail e.

If $q_0 \not\mathsf{hsame}\mathsf{e}$, the induction hypothesis applied to $\mathsf{NatMul}(d, q_0, p)$ gives a unary tail t with $\mathsf{Cont}(d, t, p)$. Composing this continuation with $\mathsf{Cont}(p, d, n)$ and using continuation associativity gives $\mathsf{Cont}(d, \mathsf{append}(t, d), n)$. The tail $\mathsf{append}(t, d)$ is unary because continuation preserves unary histories. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul nonempty quotient product extension`

Lemma 64.180 (Divisibility exposes unary divisor)**.** *If* $\mathsf{NatDivides}(d, n)$ *holds, then* d *is a unary history.*

Proof. Unpack the quotient witness in $\mathsf{NatDivides}(d, n)$. The corresponding multiplication witness has d as its multiplicand, and the multiplicand readback for closed multiplication derivations reads the unary carrier directly from it. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides divisor unary`

Theorem 64.181 (Divisibility antisymmetry up to history-sameness)**.** *Let* d *and* e *be unary histories. If* $\mathsf{NatDivides}(d, e)$ *and* $\mathsf{NatDivides}(e, d)$*, then* $d \mathsf{hsame} e$*.*

Proof. Unpack Definition 64.50. Choose unary quotients q and r with $\mathsf{NatMul}(d, q, e)$ and $\mathsf{NatMul}(e, r, d)$.

If $q \mathsf{hsame}\mathsf{e}$, the first multiplication witness has empty multiplier, so $e \mathsf{hsame}\mathsf{e}$. After replacing e by e in the second witness, Definition 64.178 gives $d \mathsf{hsame}\mathsf{e}$, hence $d \mathsf{hsame} e$. The case $r \mathsf{hsame}\mathsf{e}$ is symmetric.

It remains to consider $q \not\mathsf{hsame}\mathsf{e}$ and $r \not\mathsf{hsame}\mathsf{e}$. By Definition 64.179, there are unary tails t and s with $\mathsf{Cont}(d, t, e)$ and $\mathsf{Cont}(e, s, d)$. By closed unary generation, each tail is either e or a one-extension. If $t \mathsf{hsame}\mathsf{e}$, the first continuation is a right-unit continuation and gives $e \mathsf{hsame} d$ by Definition 29.5; symmetry gives $d \mathsf{hsame} e$. If $s \mathsf{hsame}\mathsf{e}$, the second continuation gives $d \mathsf{hsame} e$ directly. If both tails are one-extensions, the two continuations are strict unary prefix witnesses in the sense of Definition 28.10, contradicting Definition 28.14. From that contradiction the desired history-sameness follows. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides antisymmetry hsame`

Lemma 64.182 (Unary multiplication transports along the multiplicand)**.** *Let* d, q, n *be unary histories, and let* d' *be a history with* $d \mathsf{hsame} d'$*. If* $\mathsf{NatMul}(d, q, n)$*, then* $\mathsf{NatMul}(d', q, n)$*.*

Proof. By Definition 9195.7, d' is unary. Induct on the closed $\mathsf{NatMul}(d, q, n)$ witness from Definition 64.1. In the zero-multiplier case, the result is e, and the zero constructor with the unary witness for d' gives $\mathsf{NatMul}(d', \mathsf{e}, \mathsf{e})$. In the successor case, the induction hypothesis gives $\mathsf{NatMul}(d', q, n_0)$ for the predecessor product. The original last step is a continuation $\mathsf{Cont}(n_0, d, n_1)$. Transport that continuation along reflexivity on n_0, the hypothesis $d \mathsf{hsame} d'$, and reflexivity on n_1 using Definition 4.41; this gives $\mathsf{Cont}(n_0, d', n_1)$. Applying the successor constructor of Definition 64.1 yields $\mathsf{NatMul}(d', \mathsf{E1}(q), n_1)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul multiplicand hsame transport`

Lemma 64.183 (Unary multiplication transports along the multiplier)**.** *Let d, q, n be unary multiplication histories, and let q' be a history with $q\mathsf{hsame}q'$. If $\mathsf{NatMul}(d, q, n)$, then q' is unary and $\mathsf{NatMul}(d, q', n)$.*

Proof. Case-analyze the displayed history-sameness $q\mathsf{hsame}q'$. In the reflexive case the original multiplication witness already has multiplier q', and the right-unary theorem for NatMul supplies the unary witness for that multiplier. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul multiplier hsame transport`

Lemma 64.184 (Unary multiplication successor result-continuation transport)**.** *Let d, q, n be unary histories, and let n' be a history with $n\mathsf{hsame}n'$. If $\mathsf{NatMul}(d, \mathsf{E1}(q), n)$, then there is a predecessor product p with $\mathsf{NatMul}(d, q, p)$ and $\mathsf{Cont}(p, d, n')$.*

Proof. Invert the successor multiplication witness to obtain a predecessor product p and the final continuation $\mathsf{Cont}(p, d, n)$. Transport this continuation along the displayed result history-sameness, leaving the predecessor product unchanged. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul succ result cont transport`

Theorem 64.185 (Divisibility transports along divisor history-sameness)**.** *Let d and e be unary histories, and let d' be any history with $d\mathsf{hsame}d'$. If $\mathsf{NatDivides}(d, e)$, then $\mathsf{NatDivides}(d', e)$.*

Proof. Unpack Definition 64.50. Choose a unary quotient q with $\mathsf{NatMul}(d, q, e)$. By Definition 64.182, the same quotient satisfies $\mathsf{NatMul}(d', q, e)$. Since the quotient witness itself remains unary, the existential clause in Definition 64.50 gives $\mathsf{NatDivides}(d', e)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides divisor hsame transport`

Lemma 64.186 (Unary multiplication transports along the result)**.** *Let d, q, n be unary histories, and let n' be a history with $n\mathsf{hsame}n'$. If $\mathsf{NatMul}(d, q, n)$, then $\mathsf{NatMul}(d, q, n')$.*

Proof. Case-analyze the closed witness for $\mathsf{NatMul}(d, q, n)$ from Definition 64.1. In the zero-multiplier case the result is e; by closed generation for hsame in Definition 3.2, the target n' is also e, so the zero constructor gives $\mathsf{NatMul}(d, \mathsf{e}, n')$.

In the successor case, the witness consists of a predecessor product $\mathsf{NatMul}(d, q_0, n_0)$ and a continuation $\mathsf{Cont}(n_0, d, n_1)$. The result hypothesis is $n_1\mathsf{hsame}n'$. Apply Definition 4.41 to this continuation, using reflexivity from Definition 3.8 on n_0 and d and the displayed result sameness. This yields $\mathsf{Cont}(n_0, d, n')$, and the successor constructor of Definition 64.1 gives $\mathsf{NatMul}(d, \mathsf{E1}(q_0), n')$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul result hsame transport`

Lemma 64.187 (Unary multiplication append multiplier totality)**.** *Let d, w, n, q be histories. If $\mathsf{NatMul}(d, w, n)$ and q is unary, then there is a unary result r with*

$$\mathsf{NatMul}(d, \mathsf{append}(w, q), r).$$

Proof. Read the unary multiplicand from the existing multiplication witness. Totality of unary multiplication supplies a unary product e for the appended multiplier tail q. The continuation from n to e gives the displayed result $r = \mathsf{append}(n, e)$, and append-continuation closure for NatMul combines the two multiplication witnesses into the product over $\mathsf{append}(w, q)$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatMul append multiplier total`

Theorem 64.188 (Divisibility transports along dividend history-sameness)**.** *Let d and e be unary histories, and let e' be a history with $e\mathsf{hsame}e'$. If* $\mathsf{NatDivides}(d, e)$*, then* $\mathsf{NatDivides}(d, e')$*.*

Proof. By Definition 9195.7, the history e' is unary, so it is an admissible dividend endpoint. Unpack Definition 64.50. Choose a unary quotient q with $\mathsf{NatMul}(d, q, e)$. By Definition 64.186, the same quotient satisfies $\mathsf{NatMul}(d, q, e')$. The quotient witness remains unary, so the existential clause in Definition 64.50 gives $\mathsf{NatDivides}(d, e')$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides dividend hsame transport`

Corollary 64.189 (Divisibility transports along both endpoints)**.** *Let d and e be unary histories, and let d' and e' be histories with $d\mathsf{hsame}d'$ and $e\mathsf{hsame}e'$. If* $\mathsf{NatDivides}(d, e)$*, then* $\mathsf{NatDivides}(d', e')$*.*

Proof. Unary transport puts d' and e' in the unary carrier. First apply Definition 64.185 to transport the divisor from d to d', obtaining $\mathsf{NatDivides}(d', e)$. Then apply Definition 64.188 to transport the dividend from e to e'. The result is $\mathsf{NatDivides}(d', e')$. □

Lean4 ✓ `BEDC.Derived.PrimeUp.NatDivides endpoint hsame transport`

Closure status for $\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$.

Constructive story: A prime candidate starts as a unary spine that has passed the first successor: it is unary, its b1-depth is at least two, and every unary divisor shown by a continuation $\mathsf{Cont}(d, q, h)$ is forced back to the unit spine E1(e) or to h itself. Thus divisibility is not an oracle but a displayed unary continuation factorization, and the classifier on accepted candidates is just history-equality of the prime spine. The source is the same unary-prime predicate, the pattern records unary shape, exclusion of the unit, and the rule that all divisibility tests use unary continuations, while the ledger keeps the visible prime history aligned with the raw spine being tested. Since history-equality is identity on histories and the primality condition is preserved by that identity, an admitted prime gives the carrier witness and the classifier is reflexive, symmetric, transitive, and closed over admitted candidates. A unary history h lies in the signature gap of a $\mathsf{Prime}^{\uparrow}$ token p when the source-domain accepts h, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is unary, has depth at least two, every unary divisor collapses to the unit or to h, and the b1-depth of h matches the prime encoded by p. The infinitude row follows the Euclidean move already expressed in this language: from any finite prime list, inspect the unary factors of the product-plus-one spine to obtain a further prime. The chapter registers this divisor-tested construction under the name $\mathsf{Prime}^{\uparrow}$; primality is a finite condition on unary continuations, with no prime oracle, quotient class, or extra arithmetic primitive introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$, maturePackageClosed)

Scope closed: The unary-prime semantic certificate together with divisibility transport, effective primality, Euclid infinitude, and factorisation rows over unary histories.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.PrimeUp.prime\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Full prime-number theory: gcd and Euclidean algorithm package, unique factorisation, prime distribution, prime number theorem, modular arithmetic, or analytic number theory.

Upgrade path: No further upgrade – chapter at top of formal axis.

The child file `parts/concrete_instances/complex_limit/namecert_construction.tex` carries the current maturePackageClosed/bridgeChecked closurestatus; this hub routes only the distance, certificate, constant/difference, negation, and affine-combination children.

Chapter 65

A Concrete Naming Certificate for $\mathsf{CplxLim}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the complex-limit interface $\mathsf{CplxLim}^{\uparrow}$ at the paper-proof level. The construction sits above the complex-history certificate $\mathsf{Complex}^{\uparrow}$ (chapter 39) and the RealUp chapter $\mathsf{Real}^{\uparrow}$ (chapter 38), with explicit unary-history moduli throughout. Every $\mathsf{CplxLim}^{\uparrow}$ limit-facing row first consumes the $\mathsf{Complex}^{\uparrow}$ coordinate packet and the $\mathsf{Real}^{\uparrow}$ or RegSeqRat tolerance route before any complex-distance, modulus, or local naming row is read.

Theorem 65.1 (Complex-limit real-coordinate sibling route)**.** *Every accepted* $\mathsf{CplxLim}^{\uparrow}$ *read first consumes the* $\mathsf{Complex}^{\uparrow}$ *coordinate carrier of chapter 39 and the* $\mathsf{Real}^{\uparrow}$ *real-coordinate seal of chapter 38. Only after those coordinate rows are displayed may the read expose the unary-history modulus, the complex-distance rows, and the local* $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$ *rows. The route exports no host complex equality, quotient limit, selected representative, or ambient completeness theorem.*

Proof. Project the carrier already described by the chapter. Its source history is a regular complex sequence, hence its entries are read through the $\mathsf{Complex}^{\uparrow}$ coordinate carrier before any limit-facing row is available.

The $\mathsf{Complex}^{\uparrow}$ carrier itself is componentwise over real coordinates, so the real-coordinate seal is the $\mathsf{Real}^{\uparrow}$ dependency named by the same introductory route. A limit consumer therefore reaches the unary-history modulus only after the real and imaginary coordinate seals have been displayed.

The remaining rows are local to $\mathsf{CplxLim}^{\uparrow}$: they record the modulus comparison, the complex-distance history, replay, provenance, and the naming certificate. Since no coordinate in that carrier is a host complex equality, quotient limit, selected representative, or ambient completeness principle, the sibling route cannot export any of those objects. □

65.1 Modulus and complex distance on Hist

Definition 65.2 (Complex modulus on Hist)**.** *For a complex history z with components (x, y) in* $\mathsf{Rat}^{\uparrow}$*-carrier, the complex modulus* $\mathsf{CplxMod}(z, M)$ *is the closed inductive predicate asserting that M is a unary-indexed* $\mathsf{Real}^{\uparrow}$*-bound history witnessing $M^2 \geq x^2 + y^2$ under unary-history rational arithmetic. The witness is constructed by applying the rational squaring continuation to each component followed by additive composition.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexModulus`

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexModulus hsame transport`

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.CplxMod`

Definition 65.3 (Pairwise complex distance)**.** *For complex histories $z = (x_1, y_1)$ and $w = (x_2, y_2)$, the distance* $\mathsf{CplxDist}(z, w, D)$ *holds when D is the rational-history modulus of the difference $(x_1 - x_2, y_1 - y_2)$. Concretely,* $\mathsf{CplxDist}(z, w, D) := \mathsf{CplxMod}(z - w, D)$*, where $z - w$ is the pointwise rational subtraction continuation.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexDistance`

Lemma 65.4 (Complex distance is symmetric)**.** *For all complex histories z, w and unary-history bound D,* $\mathsf{CplxDist}(z, w, D)$ *iff* $\mathsf{CplxDist}(w, z, D)$*.*

Proof. The rational-subtraction continuation produces the additive inverse on each component; squaring is invariant under negation, so the modulus witness is identical. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance symm iff`

Theorem 65.5 (Complex distance from the empty left endpoint)**.** *For unary histories $w, d : \mathsf{BHist}$, the concrete complex-distance predicate from the empty left endpoint is equivalent to unary support of w and equality of the displayed distance result with w:*

$$\mathsf{ComplexDistance}(\mathsf{e}, w, d) \iff \mathsf{UnaryHistory}(w) \wedge \mathsf{hsame}(d, w).$$

Proof. Unfolding the distance predicate leaves unary support for the endpoints and a continuation branch. If the branch is $\mathsf{Cont}(\mathsf{e}, w, d)$, the left-unit law identifies d with w; if it is $\mathsf{Cont}(w, \mathsf{e}, d)$, the right-unit law gives the same identification. Conversely, unary support for w transports across $\mathsf{hsame}(d, w)$ to support d, and the left-unit continuation supplies the distance branch. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance empty left iff`

Theorem 65.6 (Complex distance transport with relation)**.** *For histories $z, z', w, w', d, d' : \mathsf{BHist}$, if z, w, d are respectively* hsame*-equal to z', w', d', then every concrete complex-distance witness for (z, w, d) transports to one for (z', w', d') and preserves an explicit continuation branch:*

$$\mathsf{ComplexDistance}(z, w, d) \Longrightarrow \mathsf{ComplexDistance}(z', w', d') \wedge (\mathsf{Cont}(z', w', d') \vee \mathsf{Cont}(w', z', d')).$$

Proof. Unary support of each endpoint transports along the three hsame witnesses. The continuation branch in the original distance witness is then transported across the same endpoint equalities and the result equality, preserving whether the branch is oriented from z to w or from w to z. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance hsame transport with relation`

Lemma 65.7 (Complex distance empty boundary)**.** *For all complex histories z, w, the empty complex distance* $\mathsf{CplxDist}(z, w, \mathsf{e})$ *is equivalent to unary carrier evidence for both endpoints together with z*hsamee *and w*hsamee*.*

Proof. Unpack the two possible continuation ledgers inside the distance witness. In either order, a continuation with empty result forces both inputs to be e. Conversely, empty endpoints and unary carrier evidence rebuild the empty-left continuation at e. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance empty iff`

Lemma 65.8 (Complex distance empty left endpoint)**.** *For every complex history w and distance history d,* $\mathsf{CplxDist}(\mathsf{e}, w, d)$ *is equivalent to unary carrier evidence for w together with $d\mathsf{hsame}w$.*

Proof. The forward direction splits the distance continuation into the empty-left and empty-right cases, both of which identify the result with w. The reverse direction transports d to w and uses the empty-left continuation. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance empty left endpoint iff`

Theorem 65.9 (Complex distance empty endpoint)**.** *For histories $z, w : \mathsf{BHist}$, a complex-distance witness with empty distance result is exactly the assertion that both endpoints are unary histories and both endpoints are history-same to* e*:*

$$\mathsf{ComplexDistance}(z, w, \mathsf{e}) \iff \mathsf{UnaryHistory}(z) \wedge \mathsf{UnaryHistory}(w) \wedge z\mathsf{hsame}\mathsf{e} \wedge w\mathsf{hsame}\mathsf{e}.$$

Proof. Unpack the symmetric continuation field inside $\mathsf{ComplexDistance}$. If the displayed continuation has result e, continuation empty-result inversion forces both endpoints of that continuation to be e; the two symmetric cases only exchange the endpoint order. Conversely, the two endpoint hsame witnesses reduce the claim to the canonical continuation $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ together with unary emptiness. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance empty endpoint iff`

Lemma 65.10 (Complex distance from the empty source)**.** *For histories $w, d : \mathsf{BHist}$,*

$$\mathsf{ComplexDistance}(\mathsf{e}, w, d) \iff \mathsf{UnaryHistory}(w) \wedge d\mathsf{hsame}w.$$

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance empty source iff`

Proof. Unpack the distance witness. The left branch is a continuation $\mathsf{Cont}(\mathsf{e}, w, d)$, whose left-unit result gives $d\mathsf{hsame}w$. The right branch is $\mathsf{Cont}(w, \mathsf{e}, d)$, whose right-unit characterization gives the same endpoint identity. Conversely, transport unary membership from w to d along $d\mathsf{hsame}w$, and use the left-unit continuation witness. □

Definition 65.11 (Complex distance triangle bound)**.** *For distance histories $D_{12}, D_{23} : \mathsf{BHist}$, the explicit BEDC history-level triangle bound is*

$$\mathsf{ComplexDistanceTriangleBound}(D_{12}, D_{23}) := \mathsf{append}(D_{12}, D_{23}).$$

It is the native continuation composition used by the complex-distance triangle discussion.

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexDistanceTriangleBound`

Theorem 65.12 (Complex distance triangle bound unary)**.** *For complex histories z, w, u and distance witnesses d_{12}, d_{23}, if* $\mathsf{ComplexDistance}(z, w, d_{12})$ *and* $\mathsf{ComplexDistance}(w, u, d_{23})$ *hold, then the explicit triangle bound is unary:*

$$\mathsf{UnaryHistory}(\mathsf{ComplexDistanceTriangleBound}(d_{12}, d_{23})).$$

Proof. Each complex-distance witness contains a unary carrier for its distance endpoint. The triangle bound is the BEDC append of the two endpoints, so unary append closure gives the displayed unary carrier. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistanceTriangleBound unary`

Theorem 65.13 (Complex distance triangle bound unary carrier from distances)**.** *For complex histories z, w, u and distance histories d_{12}, d_{23}, two concrete distance witnesses*

$$\mathsf{ComplexDistance}(z, w, d_{12}) \quad \textit{and} \quad \mathsf{ComplexDistance}(w, u, d_{23})$$

make the explicit triangle bound a unary history and retain unary support for the outer endpoints:

$$\mathsf{UnaryHistory}(\mathsf{ComplexDistanceTriangleBound}(d_{12}, d_{23})) \wedge \mathsf{UnaryHistory}(z) \wedge \mathsf{UnaryHistory}(u).$$

Proof. Project the unary distance witnesses from the two distance hypotheses. Since the triangle bound is $\mathsf{append}(d_{12}, d_{23})$, unary append closure gives unary support for the bound. The same two distance hypotheses already store unary support for the outer endpoints z and u. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistanceTriangleBound unary of distances`

Theorem 65.14 (Complex distance triangle bound empty endpoints)**.** *For complex histories z, w, u and distance histories d_{12}, d_{23}, if the two concrete distance witnesses*

$$\mathsf{ComplexDistance}(z, w, d_{12}) \quad \textit{and} \quad \mathsf{ComplexDistance}(w, u, d_{23})$$

hold and the explicit triangle bound is history-same to e*, then all three endpoints are history-same to* e*.*

Proof. The triangle bound is $\mathsf{append}(d_{12}, d_{23})$, so append-empty inversion forces both displayed distance histories to be empty. Each empty-distance witness then exposes empty endpoints by continuation empty-result inversion; applying this to the two adjacent distances gives $z\mathsf{hsamee}$, $w\mathsf{hsamee}$, and $u\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistanceTriangleBound empty endpoints`

Lemma 65.15 (Complex distance nonempty endpoint cases)**.** *For complex histories z, w and distance endpoint d, if $\mathsf{CplxDist}(z, w, d)$ holds and d is not history-same to* e*, then at least one endpoint is not history-same to* e*:*

$$\neg(d\mathsf{hsamee}) \Longrightarrow \neg(z\mathsf{hsamee}) \vee \neg(w\mathsf{hsamee}).$$

Proof. Unfold the concrete distance predicate to its two continuation branches. In the branch $\mathsf{Cont}(z, w, d)$, nonemptiness of d is exactly nonemptiness of the append result, so append-empty inversion gives a nonempty source or a nonempty right endpoint. In the symmetric branch $\mathsf{Cont}(w, z, d)$, the same append nonempty readback is applied and the two alternatives are swapped. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance nonempty distance endpoint cases`

Lemma 65.16 (Complex distance endpoint nonempty distance nonempty)**.** *For complex histories z, w and distance endpoint d, if $\mathsf{CplxDist}(z, w, d)$ holds and at least one endpoint is not history-same to* e*, then d is not history-same to* e*:*

$$\neg(z\mathsf{hsamee}) \vee \neg(w\mathsf{hsamee}) \Longrightarrow \neg(d\mathsf{hsamee}).$$

Proof. Unpack the symmetric continuation branch inside the distance witness. If the distance result were e, continuation empty-result inversion would force both endpoints of the active branch to be e. In either branch orientation this contradicts the displayed nonempty endpoint alternative. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance endpoint nonempty distance nonempty`

Lemma 65.17 (Complex distance triangle inequality with explicit bound)**.** *For all complex histories z_1, z_2, z_3 with distances $\mathsf{CplxDist}(z_1, z_2, D_{12})$ and $\mathsf{CplxDist}(z_2, z_3, D_{23})$, there exists an explicit rational-history bound function $\mathsf{TriBound}(D_{12}, D_{23}, D_{13})$ with $\mathsf{CplxDist}(z_1, z_3, D_{13})$ and $D_{13} \leq D_{12} + D_{23}$ under rational-history ordering.*

Sketch. Componentwise application of the rational-history triangle inequality from $\mathsf{Real}^{\uparrow}$ machinery (chapter 38) to the real and imaginary parts, followed by composition under the modulus witness of Definition 65.2. The bound TriBound is explicit and unary-indexed. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance triangle explicit bound`

65.2 Regular complex sequences

Definition 65.18 (Regular complex sequence)**.** *A regular complex sequence is a pair (s, N) consisting of a unary-indexed family $s : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$-history together with a unary modulus function $N : \mathsf{Unary} \to \mathsf{Unary}$ such that for every unary index k and unary indices n, m with $\mathsf{NatUnaryStrictPrefix}(N(k), n)$ (or $\mathsf{hsame}(n, N(k))$) and similarly for m, the distance $\mathsf{CplxDist}(s(n), s(m), D)$ is bounded by the unary representation of 2^{-k}. The predicate is written $\mathsf{CplxRegSeq}(s, N)$.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence`

Lemma 65.19 (Regular complex sequence has regular real and imaginary projections)**.** *If $\mathsf{CplxRegSeq}(s, N)$ holds, then the projection sequences $\mathsf{Re}(s)$ and $\mathsf{Im}(s)$ are each regular $\mathsf{Rat}^{\uparrow}$-Cauchy sequences with the same unary modulus N, in the sense of* RegSeqRat *from chapter 38.*

Proof. Each component bound is dominated by the modulus witness of Definition 65.2: if $D^2 \geq x^2 + y^2$, then $D \geq |x|$ and $D \geq |y|$. The unary modulus carries through unchanged. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence accepted pair unary witness`

Lemma 65.20 (Constant sequence is regular)**.** *For every complex history z, the constant sequence $s(n) := z$ together with the trivial modulus $N(k) := \mathsf{e}$ satisfies $\mathsf{CplxRegSeq}(s, N)$.*

Proof. For any accepted pair of indices, unary support of z supplies both endpoints. The BEDC distance witness is $\mathsf{append}(z, z)$, with the canonical continuation from z to z. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence constant`

65.3 The complex-limit predicate

Definition 65.21 (Complex limit predicate)**.** *For a regular complex sequence (s, N) and a complex history z, the predicate $\mathsf{CplxLim}(s, N, z, M)$ holds when $M : \mathsf{Unary} \to \mathsf{Unary}$ is an explicit modulus such that for every unary index k and every unary n with $\mathsf{NatUnaryStrictPrefix}(M(k), n)$ (or $\mathsf{hsame}(n, M(k))$), the distance $\mathsf{CplxDist}(s(n), z, D)$ satisfies the unary representation of $D \leq 2^{-k}$.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexLimit`

Lemma 65.22 (Complex limit functorial under hsame on the limit point)**.** *If* $\mathsf{CplxLim}(s, N, z, M)$ *and the components of* z *and* z' *are pairwise* hsame*-equal, then* $\mathsf{CplxLim}(s, N, z', M)$ *with the same modulus.*

Proof. The distance $\mathsf{CplxDist}(s(n), z, D)$ depends on z only via its rational-history components; hsame on each component preserves the modulus witness pointwise. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit hsame transport`

Theorem 65.23 (Constant complex sequence has its displayed limit)**.** *If* z *is a complex history, then the constant sequence* $s(n) := z$ *with the empty regularity modulus has complex limit* z*, witnessed by the empty limit modulus.*

Proof. The complex carrier of z gives unary support for the endpoint. The regular part is the constant regular-sequence theorem. For every accepted observation index, choose the canonical BEDC distance witness $\mathsf{append}(z, z)$; its unary support follows from unary append closure, and the distance relation is the canonical continuation from z to z. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit constant`

Theorem 65.24 (Uniqueness of complex limit)**.** *If* $\mathsf{CplxLim}(s, N, z, M)$ *and* $\mathsf{CplxLim}(s, N, z', M')$ *both hold for the same regular complex sequence, then* hsame *holds on the real components of* z *and* z' *and on the imaginary components.*

Sketch. From the two modulus witnesses construct the diagonal sequence $s(\max(M(k), M'(k)))$ and bound $\mathsf{CplxDist}(z, z', D)$ by 2^{-k+1} for every k using the triangle inequality (Definition 65.17). The componentwise rational-history archimedean property then forces hsame on each component. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit conditional unique boundary`

65.4 Closure laws on regular sequences

Lemma 65.25 (Complex distance constant suffix closure)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* $\mathsf{ComplexDistance}(z, w, d)$ *holds, then the suffixed endpoints admit the canonical BEDC distance witness*

$$\mathsf{append}(\mathsf{append}(z, q), \mathsf{append}(w, q)).$$

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance append constant closed`

Proof. Unary support for the original distance supplies unary histories for z and w. Unary append closure with q gives the two suffixed endpoints, and another append-closure step gives the displayed witness. The continuation relation is the canonical append continuation from the suffixed left endpoint to the suffixed right endpoint. □

Lemma 65.26 (Complex distance constant prefix closure)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* $\mathsf{ComplexDistance}(z, w, d)$ *holds, then the prefixed endpoints admit the canonical BEDC distance witness*

$$\mathsf{append}(\mathsf{append}(q, z), \mathsf{append}(q, w)).$$

Proof. Unary support for the distance endpoints gives unary histories for z and w. Unary append closure with q gives unary support for the two prefixed endpoints, and append closure gives the displayed distance history. The continuation component is the canonical append continuation from the prefixed left endpoint to the prefixed right endpoint. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance prepend constant closed`

Theorem 65.27 (Complex distance constant suffix result determinacy)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* $\mathsf{ComplexDistance}(z, w, d)$ *holds and a displayed distance witness for the suffixed endpoints* $\mathsf{append}(z, q)$ *and* $\mathsf{append}(w, q)$ *has result* d'*, then*

$$d' \mathsf{hsameappend}(\mathsf{append}(z, q), \mathsf{append}(w, q)).$$

Proof. Split the displayed distance witness into its two continuation orientations. The forward orientation is determined by the canonical append continuation. The reverse orientation is determined by the reverse canonical continuation, then unary append commutativity for the two suffixed endpoints identifies the reverse canonical result with the forward canonical result. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance append constant result deterministic`

Theorem 65.28 (Complex distance constant prefix result determinacy)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* $\mathsf{ComplexDistance}(z, w, d)$ *holds and a displayed distance witness for the prefixed endpoints* $\mathsf{append}(q, z)$ *and* $\mathsf{append}(q, w)$ *has result* d'*, then*

$$d' \mathsf{hsameappend}(\mathsf{append}(q, z), \mathsf{append}(q, w)).$$

Proof. Split the displayed distance witness into its two continuation orientations. The forward orientation is determined by the canonical append continuation. The reverse orientation is determined by the reverse canonical continuation, then unary append commutativity for the two prefixed endpoints identifies the reverse canonical result with the forward canonical result. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance prepend constant result deterministic`

Theorem 65.29 (Complex distance unary append commutation package)**.** *For unary histories* z *and* q*, the two appended endpoints* $q \cdot z$ *and* $z \cdot q$ *carry the canonical complex-distance witness* $(q \cdot z) \cdot (z \cdot q)$*, and the endpoints are* hsame*:*

$$\mathsf{ComplexDistance}(q \cdot z, z \cdot q, (q \cdot z) \cdot (z \cdot q)) \wedge \mathsf{hsame}(q \cdot z, z \cdot q).$$

Proof. Unary append closure gives unary carriers for $q \cdot z$, $z \cdot q$, and their displayed composite distance. The complex-distance relation uses the canonical continuation from the left endpoint to the right endpoint, while unary append commutativity supplies the endpoint classifier. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance unary append comm package`

Theorem 65.30 (Regular complex sequence constant suffix closure)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* (s, N) *is a regular complex sequence, then the suffixed sequence* $n \mapsto \mathsf{append}(s(n), q)$ *is regular with the same modulus* N*. For any accepted pair of indices* n, m*, the BEDC distance witness is*

$$\mathsf{append}(\mathsf{append}(s(n), q), \mathsf{append}(s(m), q)).$$

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence append constant closed`

Proof. Apply regularity of (s, N) to the displayed indices and threshold continuations. The resulting complex-distance witness carries $\mathsf{UnaryHistory}(s(n))$ and $\mathsf{UnaryHistory}(s(m))$. Unary append closure with $\mathsf{UnaryHistory}(q)$ gives unary histories for the two suffixed endpoints, and a second append-closure step gives the unary distance history. The canonical continuation witness for append supplies the distance relation, so the suffixed sequence satisfies the same regularity predicate. □

Theorem 65.31 (Complex limit constant suffix closure)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* $\mathsf{CplxLim}(s, N, z, M)$ *holds, then the suffixed sequence* $n \mapsto \mathsf{append}(s(n), q)$ *has complex limit* $\mathsf{append}(z, q)$ *with the same regularity modulus* N *and limit modulus* M*.*

Proof. Regularity is transported by the regular-sequence constant suffix closure. The limit carrier is closed by complex-history unary suffix closure. For every accepted modulus index, the original limit witness gives a complex distance from $s(n)$ to z; applying constant suffix closure to that distance gives the canonical distance from $\mathsf{append}(s(n), q)$ to $\mathsf{append}(z, q)$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit append constant closed`

Theorem 65.32 (Regular complex sequence constant suffix result determinacy)**.** *Let* $q : \mathsf{BHist}$ *be unary and let* (s, N) *be a regular complex sequence. For accepted unary indices* n, m *at threshold* k*, every displayed complex-distance result* d' *for the suffixed endpoints* $\mathsf{append}(s(n), q)$ *and* $\mathsf{append}(s(m), q)$ *satisfies*

$$d' \mathsf{hsameappend}(\mathsf{append}(s(n), q), \mathsf{append}(s(m), q)).$$

Proof. Regularity supplies a base complex-distance witness between $s(n)$ and $s(m)$. Applying the constant-suffix result determinacy theorem to that base witness and to the displayed suffixed distance identifies d' with the canonical suffixed append result. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence append constant result deterministic`

Theorem 65.33 (Regular complex sequence constant prefix closure)**.** *Let* $q : \mathsf{BHist}$ *be unary. If* (s, N) *is a regular complex sequence, then the prefixed sequence* $n \mapsto \mathsf{append}(q, s(n))$ *is regular with the same modulus* N*. For any accepted pair of indices* n, m*, the BEDC distance witness is*

$$\mathsf{append}(\mathsf{append}(q, s(n)), \mathsf{append}(q, s(m))).$$

Proof. Apply regularity of (s, N) to the accepted indices and threshold continuations. The resulting complex-distance witness supplies unary histories for $s(n)$ and $s(m)$. Unary append closure with $\mathsf{UnaryHistory}(q)$ gives unary histories for the two prefixed endpoints, and append closure again gives the displayed distance history. The canonical append continuation witness supplies the distance relation. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence prepend constant closed`

Theorem 65.34 (Regular complex sequence endpoint transport)**.** *Let* $s, t, N : \mathsf{Hist} \to \mathsf{Hist}$*. If every unary index* n *satisfies* $\mathsf{hsame}(s(n), t(n))$*, then regularity of* (s, N) *implies regularity of* (t, N) *with the same modulus.*

Proof. Apply regularity of (s, N) to the accepted unary indices n, m. Reuse the same distance witness d, transport the two complex-distance endpoints along the pointwise hsame comparisons, and keep d fixed by reflexivity. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence hsame transport`

Theorem 65.35 (Complex limit sequence endpoint transport)**.** *Let $s, t, N, M : \mathsf{Hist} \to \mathsf{Hist}$. If every unary index n satisfies $\mathsf{hsame}(s(n), t(n))$, then $\mathsf{CplxLim}(s, N, z, M)$ implies $\mathsf{CplxLim}(t, N, z, M)$.*

Proof. The regular part is transported pointwise by the regular-sequence endpoint transport theorem. The displayed complex-history carrier for z is unchanged. For each accepted limit-modulus index, reuse the original distance witness and transport its sequence endpoint from $s(n)$ to $t(n)$ along the supplied hsame comparison. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit sequence hsame transport`

Definition 65.36 (Pointwise sum of complex sequences)**.** *For complex sequences s, t, the pointwise sum $s \oplus t$ is given by $(s \oplus t)(n) := s(n) + t(n)$, where $+$ is componentwise rational-history addition continuation on $\mathsf{Complex}^{\uparrow}$-carriers.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexPointwiseSum`

Lemma 65.37 (Sum of regular sequences is regular with explicit modulus pasting)**.** *If $\mathsf{CplxRegSeq}(s, N_s)$ and $\mathsf{CplxRegSeq}(t, N_t)$, then $\mathsf{CplxRegSeq}(s \oplus t, N')$ with $N'(k) := \max(N_s(k+1), N_t(k+1))$ (max in unary-history strict prefix order).*

Proof. Applying the triangle inequality to $\mathsf{CplxDist}((s \oplus t)(n), (s \oplus t)(m), D)$ decomposes the bound into the sum of the s- and t-bounds, each at level $k+1$, giving total 2^{-k}. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence pointwise sum closed`
Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence pointwise append same modulus closed`

Theorem 65.38 (Limit respects pointwise sum)**.** *If $\mathsf{CplxLim}(s, N_s, z_s, M_s)$ and $\mathsf{CplxLim}(t, N_t, z_t, M_t)$, then $\mathsf{CplxLim}(s \oplus t, N', z_s + z_t, M')$ with explicit pasted modulus $M'(k) := \max(M_s(k+1), M_t(k+1))$.*

Sketch. Triangle inequality on $\mathsf{CplxDist}((s \oplus t)(n), z_s + z_t, D)$: bound by the sum of the two component distances, each at level $k+1$ by the hypothesis on M_s and M_t. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise append same modulus closed`

Theorem 65.39 (Limit respects scalar multiplication by a fixed complex history)**.** *For a fixed complex history c with modulus bound C (i.e. $\mathsf{CplxMod}(c, C)$), suppose a supplied unary shift witness σ_C carries the certificate that every distance bound at precision $k + \sigma_C$, after multiplication by the fixed bound C, is a distance bound at precision k. If $\mathsf{CplxLim}(s, N, z, M)$ holds, then the scaled sequence $c \cdot s$ (componentwise rational-history multiplication) satisfies $\mathsf{CplxLim}(c \cdot s, N, c \cdot z, M_C)$ with $M_C(k) := M(k + \sigma_C)$.*

Sketch. Given k and n above $M_C(k)$, the original limit witness gives a distance certificate from $s(n)$ to z at the shifted precision $k + \sigma_C$.

Use the bilinearity rows for complex multiplication to transport this distance through left multiplication by c. The bound $\mathsf{CplxMod}(c, C)$ supplies the fixed scalar factor, and the certificate carried by σ_C converts the resulting C-scaled distance into the required precision-k bound. Repack the transported distance as the modulus clause for $\mathsf{CplxLim}(c \cdot s, N, c \cdot z, M_C)$. □

Theorem 65.40 (Complex limit constant prefix closure)**.** *Let $q : \mathsf{BHist}$ be unary. If $\mathsf{CplxLim}(s, N, z, M)$ holds, then the prefixed sequence $n \mapsto \mathsf{append}(q, s(n))$ has complex limit $\mathsf{append}(q, z)$ with the same regularity modulus N and limit modulus M.*

Proof. Regularity is transported by the regular-sequence constant prefix closure. The limit carrier is closed by complex-history unary prefix closure.

For every accepted modulus index, the original limit witness gives a complex distance from $s(n)$ to z. Applying constant prefix closure to that distance gives the canonical distance from $\mathsf{append}(q, s(n))$ to $\mathsf{append}(q, z)$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit prepend constant closed`

Theorem 65.41 (Complex limit two-sided unary context closure)**.** *Let $p, q : \mathsf{BHist}$ be unary. If $\mathsf{CplxLim}(s, N, z, M)$ holds, then the two-sided contexted sequence $n \mapsto \mathsf{append}(p, \mathsf{append}(s(n), q))$ has complex limit $\mathsf{append}(p, \mathsf{append}(z, q))$ with the same regularity modulus N and limit modulus M.*

Proof. First apply Definition 65.31 with suffix q. This gives a complex limit for $n \mapsto \mathsf{append}(s(n), q)$ at endpoint $\mathsf{append}(z, q)$, keeping both moduli unchanged.

Next apply the checked prefix limit closure with prefix p to the suffixed limit. The resulting endpoint is $\mathsf{append}(p, \mathsf{append}(z, q))$, and the resulting sequence is exactly $n \mapsto \mathsf{append}(p, \mathsf{append}(s(n), q))$, again with the same regularity and limit moduli. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit two sided unary context closure`

Theorem 65.42 (Limit respects pointwise product with quadratic modulus)**.** *If $\mathsf{CplxLim}(s, N_s, z_s, M_s)$ and $\mathsf{CplxLim}(t, N_t, z_t, M_t)$ with both target points bounded $\mathsf{CplxMod}(z_s, B_s)$, $\mathsf{CplxMod}(z_t, B_t)$, then the pointwise product $s \otimes t$ satisfies $\mathsf{CplxLim}(s \otimes t, N', z_s \cdot z_t, M')$ with quadratic modulus $M'(k) := \max(M_s(k + B_t + 1), M_t(k + B_s + 1))$.*

Sketch. Standard $ab - a'b' = a(b - b') + (a - a')b'$ decomposition; bound each summand using the hypotheses, with the bounds on z_s, z_t entering as the multiplicative shift on the modulus indices. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise product witness package`

65.5 Componentwise limit assembly

Definition 65.43 (Cauchy regular complex sequence)**.** *The Cauchy condition on (s, N) is exactly the condition built into Definition 65.18: it is the same regular-sequence predicate, named separately for emphasis. We write $\mathsf{CplxCauchy}(s, N) := \mathsf{CplxRegSeq}(s, N)$.*

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence cauchy witness package`

Lemma 65.44 (Component limit assembly)**.** *Assume $\mathsf{CplxCauchy}(s, N)$. Suppose the real projection $\mathsf{Re}(s)$ is supplied with an explicit $\mathsf{RegSeqRat}$ limit witness x^* and modulus M_x, and the imaginary projection $\mathsf{Im}(s)$ is supplied with an explicit $\mathsf{RegSeqRat}$ limit witness y^* and modulus M_y. Then the pair (x^*, y^*) is the candidate complex limit point, with pasted modulus $M^\sharp(k) := \max(M_x(k + 1), M_y(k + 1))$.*

Sketch. Use Definition 65.19 only to read the real and imaginary projections as regular $\mathsf{RegSeqRat}$ streams. The component limit witnesses are part of the input data, so no ambient completeness principle is invoked.

For each unary level k, ask the real component at level $k + 1$ through M_x and the imaginary component at level $k + 1$ through M_y. Taking the unary maximum gives one threshold above which both component bounds hold, and the modulus witness of Definition 65.2 packages the two component inequalities as a complex-distance bound. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit component assembly candidate unary`

Theorem 65.45 (Complex limit from component limits)**.** *If* $\mathsf{CplxCauchy}(s, N)$ *holds and explicit* $\mathsf{RegSeqRat}$ *limit witnesses are supplied for* $\mathsf{Re}(s)$ *and* $\mathsf{Im}(s)$ *with limit histories* x^* *and* y^* *and moduli* M_x *and* M_y*, then there exists an explicit complex history* $z^* = (x^*, y^*)$ *and pasted modulus* M^* *such that* $\mathsf{CplxLim}(s, N, z^*, M^*)$*. Any other complex limit with its own modulus agrees with* z^* *componentwise by Definition 65.24.*

Sketch. Apply Definition 65.44 to obtain the candidate point and the pasted unary threshold. The real and imaginary component bounds above that threshold are exactly the two inputs needed by the complex modulus predicate.

Define $z^* := (x^*, y^*)$ and $M^* := M^\sharp$. The distance predicate of Definition 65.3 reduces the desired limit statement to the modulus packaging supplied in the assembly lemma, and uniqueness is only the already stated consequence of Definition 65.24. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit component limits closed`
Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit from component limits`

65.6 The certificate

Definition 65.46 (Complex-limit source specification)**.** *The source specification on* Hist *restricts to pairs* $((s, N), (z, M))$ *such that* $\mathsf{CplxRegSeq}(s, N)$ *and* $\mathsf{CplxLim}(s, N, z, M)$*. The generator is the complex-limit predicate of Definition 65.21.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexLimitSourceSpec`
Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimitSourceSpec constant`

Definition 65.47 (Complex-limit pattern specification)**.** *The pattern of* $\mathsf{CplxLim}^\uparrow$ *is regular-sequence-bounded convergence: for an input regular complex sequence* (s, N)*, the pattern is the family of distances* $\mathsf{CplxDist}(s(n), z, D)$ *restricted to* n *above the modulus threshold* $M(k)$ *at each unary level* k*.*

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimitPatternSpec of limit`

Definition 65.48 (Complex-limit classifier specification)**.** *The classifier on complex-limit carriers is* hsame *componentwise: two complex-limit witnesses classify together iff their target points agree under* hsame *on both real and imaginary components.*

Definition 65.49 (Complex-limit stability certificate)**.** *The stability certificate consists of:*

1. *closure of* $\mathsf{CplxLim}$ *under* hsame *on each rational-history component of the limit point (Definition 65.22);*
2. *transport of regular-sequence status under pointwise* hsame *on each member of the sequence;*
3. *compatibility of the modulus function with* hsame *on its unary inputs and outputs.*

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit stability certificate fields`

Definition 65.50 (Complex-limit ledger policy)**.** *The checked ledger policy records:*

1. *composition of moduli under pointwise sum (Definition 65.38);*
2. *composition of moduli under pointwise product (Definition 65.42);*

3. componentwise limit assembly from supplied component witnesses (Definition 65.45).

The scalar multiplication law of Definition 65.39 is a paper-side proof obligation parameterized by supplied unary shift evidence, and is not part of the checked ledger policy until a matching Lean target is present.

Theorem 65.51 (Concrete $\mathsf{CplxLim}^{\uparrow}$ naming certificate)**.** *The data* (CplxLim, CplxRegSeq, hsame, CplxLimStabilityC *of Definition 65.46–Definition 65.50 forms a naming certificate over* Hist *for the complex-limit interface* $\mathsf{CplxLim}^{\uparrow}$*. The checked certificate is functional (Definition 65.24), closed under pointwise sum and product (Definition 65.38 and Definition 65.42), and supports componentwise limit assembly when component limit witnesses are part of the input (Definition 65.45). Scalar multiplication remains a paper-side obligation parameterized by the supplied unary shift witness of Definition 65.39.*

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.complex limit name certificate`

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). Functionality is Definition 65.24. Checked additive closure is Definition 65.38; checked product closure is Definition 65.42. Componentwise limit assembly is Definition 65.45.

The scalar row is retained only as the paper-side obligation of Definition 65.39, whose statement explicitly requires a unary shift witness. It is not counted as checked certificate data by this Lean marker. The certificate uses no Classical.choice, no Quot.sound, and no propext, so its Lean targets fall under the project's axiom-purity gate. □

Theorem 65.52 (Semantic $\mathsf{CplxLim}^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 65.51 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9): the carrier is a* Hist*-predicate, the classifier is a restriction of* hsame *to complex-history components, and every closure law is given by an explicit constructive proof with explicit modulus pasting.*

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit semanticNameCert`

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.complex limit semantic name certificate`

Theorem 65.53 (Constant complex sequence has constant complex limit)**.** *Let z be a complex history. Define the constant sequence $s : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$ by $s(n) := z$, and define the two trivial moduli by*

$$N_{\mathrm{const}}(k) := \mathsf{e}, \qquad M_{\mathrm{const}}(k) := \mathsf{e}.$$

If $\mathsf{CplxRegSeq}(s, N_{\mathrm{const}})$ *holds by Definition 65.20, then*

$$\mathsf{CplxLim}(s, N_{\mathrm{const}}, z, M_{\mathrm{const}})$$

holds.

Proof. Fix a unary precision k and a unary index n satisfying the threshold clause in Definition 65.21 for $M_{\mathrm{const}}(k)$. Since $M_{\mathrm{const}}(k) = N_{\mathrm{const}}(k) = \mathsf{e}$, the same threshold clause makes n acceptable for the regularity condition of Definition 65.18. Take the second regularity index to be $m := \mathsf{e}$; it is acceptable because $\mathsf{hsame}(\mathsf{e}, N_{\mathrm{const}}(k))$ is reflexivity.

Apply $\mathsf{CplxRegSeq}(s, N_{\text{const}})$ at the indices n and m. This gives a distance witness D such that $\mathsf{CplxDist}(s(n), s(m), D)$ satisfies the unary representation of $D \leq 2^{-k}$. By the definition of the constant sequence, $s(m) = s(\mathsf{e}) = z$. Therefore the same witness D is exactly the distance estimate from $s(n)$ to z required in Definition 65.21. Since k and n were arbitrary, M_{const} is a valid complex-limit modulus for the constant sequence. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit constant of regular`

Theorem 65.54 (Eventually constant complex sequence has constant limit)**.** *Let* $s : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$ *be a complex sequence, let* $N, M : \mathsf{Unary} \to \mathsf{Unary}$ *be unary moduli, and let* z *be a complex history. Suppose* $\mathsf{CplxRegSeq}(s, N)$ *and suppose that* s *is pointwise history-same to* z *above* M*:*

$$\forall k, n : \mathsf{Unary},\ \big(\mathsf{NatUnaryStrictPrefix}(M(k), n) \vee \mathsf{hsame}(n, M(k))\big) \Longrightarrow \mathsf{hsame}(s(n), z).$$

Then

$$\mathsf{CplxLim}(s, N, z, M).$$

Proof. Fix a unary precision k and a unary index n satisfying the threshold clause in Definition 65.21 for $M(k)$. Put $c(u) := z$ for the constant sequence. The empty threshold used in Definition 65.53 accepts every unary index: if $n\mathsf{hsamee}$ this is the non-strict branch, and otherwise the tail n with the left-unit continuation $\mathsf{Cont}(\mathsf{e}, n, n)$ gives $\mathsf{NatUnaryStrictPrefix}(\mathsf{e}, n)$ by Definition 28.10. Hence the constant-limit theorem supplies a distance witness D from $c(n) = z$ to z satisfying the unary estimate $D \leq 2^{-k}$.

The above-M hypothesis gives $\mathsf{hsame}(s(n), z)$, hence $\mathsf{hsame}(s(n), c(n))$. Apply Definition 65.6 to the distance witness for $c(n)$ and z, using reflexivity of hsame on the limit endpoint and on D. The transported witness has endpoints $s(n)$ and z, and the same distance history D, so the estimate $D \leq 2^{-k}$ is unchanged.

Since k and n were arbitrary among the accepted indices, these witnesses are exactly the convergence pattern required by Definition 65.21. The hypothesis $\mathsf{CplxRegSeq}(s, N)$ supplies the regular-sequence side of the predicate, so $\mathsf{CplxLim}(s, N, z, M)$ holds. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit eventually constant closed`

Definition 65.55 (Pointwise difference of complex sequences)**.** *For complex sequences* $s, t : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$*, the pointwise difference* $s \ominus t$ *is the sequence*

$$(s \ominus t)(n) := s(n) - t(n),$$

where subtraction is the componentwise rational-history subtraction continuation on the two rational components of the complex carrier.

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexPointwiseDifference`

Lemma 65.56 (Difference of regular complex sequences is regular with explicit modulus pasting)**.** *If* $\mathsf{CplxRegSeq}(s, N_s)$ *and* $\mathsf{CplxRegSeq}(t, N_t)$ *hold, then* $\mathsf{CplxRegSeq}(s \ominus t, N_{\ominus})$ *holds for the explicit pasted modulus*

$$N_{\ominus}(k) := \max(N_s(k+1), N_t(k+1)).$$

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexRegularSequence pointwise difference closed`

Proof. Fix a unary precision k and accepted unary indices n, m above $N_\ominus(k)$. The maximum clause makes the same indices accepted above $N_s(k+1)$ and above $N_t(k+1)$. The two regularity hypotheses therefore give distance witnesses D_s and D_t for $s(n)$ to $s(m)$ and for $t(n)$ to $t(m)$, each bounded by the unary dyadic budget $2^{-(k+1)}$.

By the componentwise subtraction continuation,

$$(s(n) - t(n)) - (s(m) - t(m)) = (s(n) - s(m)) - (t(n) - t(m))$$

component by component. The distance predicate of Definition 65.3 reads this as the modulus of a componentwise difference. Applying the symmetry of complex distance from Definition 65.4 to the t-term if needed, and then applying the triangle inequality of Definition 65.17, gives a distance witness for $(s \ominus t)(n)$ to $(s \ominus t)(m)$ bounded by $D_s + D_t$. Since both summands are bounded by $2^{-(k+1)}$, unary dyadic addition gives $D_s + D_t \le 2^{-k}$. This is exactly the regularity clause of Definition 65.18 for $s \ominus t$ with modulus $N_\ominus$. □

Theorem 65.57 (Limit respects pointwise difference)**.** *If* $\mathsf{CplxLim}(s, N_s, z_s, M_s)$ *and* $\mathsf{CplxLim}(t, N_t, z_t, M_t)$ *hold, then*

$$\mathsf{CplxLim}(s \ominus t, N_\ominus, z_s - z_t, M_\ominus)$$

holds with explicit pasted moduli

$$N_\ominus(k) := \max(N_s(k+1), N_t(k+1)), \qquad M_\ominus(k) := \max(M_s(k+1), M_t(k+1)).$$

Proof. The regular-sequence part is Definition 65.56, applied to the regularity fields contained in the two limit hypotheses of Definition 65.21.

It remains to verify the limit estimate. Fix a unary precision k and a unary index n accepted above $M_\ominus(k)$. By the maximum clause, n is accepted above both $M_s(k+1)$ and $M_t(k+1)$. The two limit hypotheses provide distance witnesses E_s and E_t from $s(n)$ to z_s and from $t(n)$ to z_t, each bounded by $2^{-(k+1)}$.

The componentwise subtraction ledgers give the identity

$$(s \ominus t)(n) - (z_s - z_t) = (s(n) - z_s) - (t(n) - z_t).$$

Using Definition 65.3, the desired distance is the complex modulus of the right-hand side. Symmetry of complex distance from Definition 65.4 handles the signed second summand, and the triangle inequality of Definition 65.17 bounds the displayed distance by $E_s + E_t$. The dyadic half-budget calculation gives $E_s + E_t \le 2^{-k}$. Since k and n were arbitrary among the accepted indices, $M_\ominus$ satisfies the convergence clause of Definition 65.21 for the pointwise difference sequence and limit point $z_s - z_t$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise difference`
Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise difference closed`

Definition 65.58 (Pointwise additive inverse of complex sequences)**.** *For a complex sequence* $s : \mathsf{Unary} \to \mathsf{Complex}^\uparrow$*, the pointwise additive inverse* $-s$ *is the sequence*

$$(-s)(n) := -(s(n)),$$

where the endpoint inverse is formed componentwise on the real and imaginary rational-history components of the complex carrier from Definition 39.2.

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexPointwiseNegation`

Lemma 65.59 (Complex distance is invariant under simultaneous negation)**.** *For complex histories a, b and a distance history D,*

$$\mathsf{CplxDist}(a, b, D) \Longrightarrow \mathsf{CplxDist}(-a, -b, D).$$

Proof. By Definition 65.3, the hypothesis is a complex-modulus witness for the componentwise difference $a - b$. The componentwise additive-inverse ledger gives

$$(-a) - (-b) = -(a - b)$$

on both rational-history components. In Definition 65.2, replacing both components of a difference by their additive inverses does not change the displayed square terms, since rational-history squaring is invariant under additive inverse. The same modulus witness D therefore certifies the distance from $-a$ to $-b$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexDistance simultaneous negation`

Theorem 65.60 (Complex limit respects pointwise negation)**.** *If $\mathsf{CplxLim}(s, N, z, M)$ holds, then the pointwise additive-inverse sequence has complex limit $-z$ with the same regularity modulus and the same limit modulus:*

$$\mathsf{CplxLim}(-s, N, -z, M).$$

Proof. Unpack the limit hypothesis using Definition 65.21. Its regularity field is a witness for $\mathsf{CplxRegSeq}(s, N)$, its target field carries the complex history z, and its modulus field gives the distance estimates from $s(n)$ to z above M.

First verify regularity for $-s$ with the unchanged modulus N. For a unary precision k and accepted unary indices n, m in the sense of Definition 65.18, regularity of s gives a distance witness D from $s(n)$ to $s(m)$ with the required dyadic bound. Applying Definition 65.59 gives a distance witness with the same D from $(-s)(n)$ to $(-s)(m)$; the numerical bound is unchanged because the distance history has not changed. Thus $\mathsf{CplxRegSeq}(-s, N)$ holds.

The target carrier for $-z$ is supplied by the componentwise additive-inverse closure of the $\mathsf{Complex}^{\uparrow}$ carrier from Definition 39.2. Now fix a unary precision k and an accepted unary index n above $M(k)$ as in Definition 65.21. The original limit witness supplies a distance history E from $s(n)$ to z satisfying the 2^{-k} bound. Applying Definition 65.59 gives the same distance history E from $(-s)(n)$ to $-z$, and the same dyadic estimate remains valid. Since k and n were arbitrary among the accepted indices, M is a limit modulus for $-s$ at $-z$. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise negation closed`
Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise negation`

Definition 65.61 (Pointwise binary affine combination of complex sequences)**.** *For fixed complex histories a, b and complex sequences $s, t : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$, the pointwise binary affine combination $a \cdot s + b \cdot t$ is the unary-indexed sequence $(a \cdot s + b \cdot t)(n) := (a \cdot s(n)) \oplus (b \cdot t(n))$. Here the scalar products are the componentwise complex products used in Definition 65.39, and $\oplus$ is the pointwise sum of Definition 65.36.*

Lean4 (definition) `BEDC.Derived.ComplexLimitUp.ComplexPointwiseBinaryAffineCombination`

Theorem 65.62 (Complex limit respects binary affine combinations)**.** *Let a, b be fixed complex histories with unary modulus bounds A, B, so $\mathsf{CplxMod}(a, A)$ and $\mathsf{CplxMod}(b, B)$. Let $\sigma_{ab} := \log_2(A + B + 1)$ be the standard unary scalar shift for the combined bound $A + B$. Suppose*

$\mathsf{CplxLim}(s, N_s, z_s, M_s)$ *and* $\mathsf{CplxLim}(t, N_t, z_t, M_t)$ *hold. Define pasted shifted moduli by* $N_{ab}(k) := \max\big(N_s(k+1+\sigma_{ab}), N_t(k+1+\sigma_{ab})\big)$ *and* $M_{ab}(k) := \max\big(M_s(k+1+\sigma_{ab}), M_t(k+1+\sigma_{ab})\big)$*. Then the binary affine combination has the displayed complex limit:*

$$\mathsf{CplxLim}(a \cdot s + b \cdot t, N_{ab}, a \cdot z_s + b \cdot z_t, M_{ab}).$$

Proof. The proof packages the scalar estimate of Definition 65.39 with the sum-pasting mechanism of Definition 65.38. Unpack the two limit hypotheses by Definition 65.21. Their regularity fields give $\mathsf{CplxRegSeq}(s, N_s)$ and $\mathsf{CplxRegSeq}(t, N_t)$, and their convergence fields give the distance estimates from $s(n)$ to z_s and from $t(n)$ to z_t above the displayed moduli. The target point $a \cdot z_s + b \cdot z_t$ is a complex history by the componentwise complex carrier of Definition 39.2, the scalar-product carrier used in Definition 65.39, and the sum carrier used in Definition 65.36. First verify regularity. Fix a unary precision k and unary indices n, m accepted above $N_{ab}(k)$ in the sense of Definition 65.18. The maximum clause makes both indices accepted above $N_s(k+1+\sigma_{ab})$ and above $N_t(k+1+\sigma_{ab})$. Regularity of s and t supplies distance histories D_s, D_t with $D_s \le 2^{-(k+1+\sigma_{ab})}$ and $D_t \le 2^{-(k+1+\sigma_{ab})}$. The scalar estimate underlying Definition 65.39 and the bounds $\mathsf{CplxMod}(a, A)$, $\mathsf{CplxMod}(b, B)$ give distance bounds AD_s for $a \cdot s(n)$ to $a \cdot s(m)$ and BD_t for $b \cdot t(n)$ to $b \cdot t(m)$. Applying the triangle inequality of Definition 65.17 after the pointwise sum construction of Definition 65.36 bounds the affine-distance witness by $AD_s + BD_t$. The defining dyadic property of σ_{ab} gives $AD_s + BD_t \le (A+B)2^{-(k+1+\sigma_{ab})} \le 2^{-k}$, so $\mathsf{CplxRegSeq}(a \cdot s + b \cdot t, N_{ab})$ holds. It remains to verify the convergence modulus. Fix a unary precision k and a unary index n accepted above $M_{ab}(k)$ as in Definition 65.21. The maximum clause makes n accepted above both $M_s(k+1+\sigma_{ab})$ and $M_t(k+1+\sigma_{ab})$. The two limit hypotheses give distance histories E_s, E_t with $E_s \le 2^{-(k+1+\sigma_{ab})}$ and $E_t \le 2^{-(k+1+\sigma_{ab})}$. Componentwise bilinearity gives $(a \cdot s + b \cdot t)(n) - (a \cdot z_s + b \cdot z_t) = a \cdot (s(n) - z_s) + b \cdot (t(n) - z_t)$. Using Definition 65.3, the same scalar estimates bound the two displayed summands by AE_s and BE_t, and Definition 65.17 bounds their sum by $AE_s + BE_t$. The same dyadic calculation gives $AE_s + BE_t \le 2^{-k}$. Since k and n were arbitrary among the accepted indices, M_{ab} satisfies the convergence clause for $a \cdot s + b \cdot t$ at $a \cdot z_s + b \cdot z_t$. Together with the regularity proved above, this is exactly the asserted $\mathsf{CplxLim}$ statement. □

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit pointwise binary affine combination closed`

Lean4 ✓ `BEDC.Derived.ComplexLimitUp.ComplexLimit binary affine combination closed`

Closure status for $\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$**.**

Constructive story: A complex limit is carried by a displayed package whose source history contains a regular complex sequence with a unary modulus and a proposed complex target with its own convergence witness; histories lacking the regular-sequence row, the target row, or the modulus comparison are simply outside the carrier. The classifier does not identify limits by an external topology, but compares the carried target point componentwise by hsame on the real and imaginary histories, while the source specification covers exactly the regular-sequence-and-limit pairs and the pattern records the distance rows $\mathsf{CplxDist}(s(n), z, D)$ once the unary index has passed the displayed threshold $M(k)$. The ledger keeps the raw sequence, target, and modulus data visible, and its checked rows paste moduli for constant sequences, pointwise sums, pointwise products, and componentwise assembly rather than hiding convergence behind choice. The five name-certificate obligations follow from the constant-sequence witness for nonemptiness, the componentwise hsame equivalence laws, and the stability rows saying that regularity, limit status, and modulus comparisons transport along the same component histories. A source history h lies in the signature gap of a $\mathsf{CplxLim}^\uparrow$ token p when the source-domain accepts the regular sequence and limit package, the signature relation pairs that package with an answer target and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the target components must be the ones encoded by p and the ledger must display the modulus that proves the sequence is eventually within each requested distance. The chapter registers this regular-sequence convergence construction under the name $\mathsf{CplxLim}^\uparrow$; it introduces no completeness principle, no choice-based extraction of a limit point, and no standard complex-analysis bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The bridged paper scope is the semantic name certificate surface, constant-sequence row, pointwise closure rows, explicit modulus rows, and the paper bridge from those rows to the standard complex-limit schema.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.ComplexLimitUp.ComplexLimit\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: The chapter does not close completeness of complex Cauchy sequences, standard complex-analysis bridge theorems, or choice-based extraction of limit points.

Upgrade path: no further upgrade – chapter at top of formal axis.

Theorem 65.63 (Complex-limit seed closure register)**.** *The seed surface already present in the* $\mathsf{CplxLim}^\uparrow$ *carrier consists of the* $\mathsf{Complex}^\uparrow$ *coordinate rows, the* $\mathsf{Real}^\uparrow$ *coordinate seals, the unary modulus row, the complex-distance rows, and the local* $\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$ *source, pattern, classifier, stability, ledger, replay, provenance, and naming rows. A seed consumer may read those rows before it asks for closure under constant, difference, negation, or affine-combination operations, but it cannot replace them by host complex equality, quotient-limit data, selected representatives, or an ambient completeness theorem.*

Proof. Project the carrier through the sibling route of Definition 65.1. The source sequence first exposes $\mathsf{Complex}^\uparrow$ coordinates, and those coordinates are sealed componentwise by the $\mathsf{Real}^\uparrow$ rows before any limit-facing data is available.

The certificate file then supplies the local rows. The source and pattern specifications identify the unary modulus and complex-distance histories, while the classifier, stability certificate, ledger policy, replay, provenance, and local name are exactly the $\mathsf{NameCert}_{\mathsf{CplxLim}^\uparrow}$ seed rows.

All later closure files consume this seed surface. Constant, difference, negation, and affine-combination rows may extend the ledger, but they do not create a new seed source or import host complex equality, quotient limits, representatives, or completeness. □

The $\mathsf{CplxLim}^\uparrow$ coordinatewise limit surface reads its real-coordinate and regular-rational evidence through chapter 38 and chapter 301 before any complex limit consumer receives a seal. This dependency keeps complex convergence on the displayed $\mathsf{Real}^\uparrow$ and $\mathsf{RegSeqRat}^\uparrow$ route rather than on host complex completeness, quotient stream equality, or selected limit data. Before any $\mathsf{CplxLim}^\uparrow$ seal is read, the packet consumes the real-coordinate sibling of chapter 38 and the regular-rational sibling of chapter 301. This route keeps complex-limit convergence tied to displayed $\mathsf{Real}^\uparrow$ and $\mathsf{RegSeqRat}^\uparrow$ carrier rows rather than to host complex completeness, quotient stream equality, or selected coordinate limits.

Chapter 66

A Concrete Naming Certificate for $\mathsf{ConvRad}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the convergence radius of a complex power series $\sum a_n z^n$ at the paper-proof level. The certificate sits above $\mathsf{CplxLim}^{\uparrow}$ (chapter 65) and the formal power series chapter (chapter 51), and gives an explicit unary-history witness for the radius rather than a real-number supremum. Absolute convergence inside displayed strict subradii is proved by geometric majorant; Cauchy–Hadamard limsup exactness is recorded only as a separate analytic ledger row outside the checked-row reduct. The construction is fully constructive and uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

66.1 Power series carrier and partial sums

Definition 66.1 (Power series carrier)**.** *A complex power series carrier is a unary-indexed family* $a : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$*-history of complex coefficient histories together with a complex history* z_0 *marking the origin of the expansion.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesCarrier`

Lemma 66.2 (Power series origin and coefficient transport)**.** *If the origin history is transported along* hsame *and every displayed coefficient is classified by the complex-history classifier, then the target origin is unary and the transported coefficient family is again a power series carrier at that target origin.*

Proof. The origin carrier is transported by unary transport. For each coefficient, the target-carrier component of the complex-history classifier supplies the complex coefficient carrier, so the transported family satisfies the power series carrier predicate. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesCarrier origin coefficient transport`

Lemma 66.3 (Power series carrier append-unary closure)**.** *If* (a, z_0) *is a power series carrier and* q *is unary, then appending* q *to the origin and to every coefficient gives another power series carrier:*

$$\mathsf{PowerSeriesCarrier}(\lambda n.\mathsf{append}(a_n, q), \mathsf{append}(z_0, q)).$$

Proof. Unpack the power-series carrier into the unary origin, coefficient-wise complex-history carrier, and complex-history origin carrier. Unary histories are closed under append, and complex-history carriers are closed under append of a unary suffix; apply the latter pointwise to every coefficient and once to the origin. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesCarrier append unary closed`

Lemma 66.4 (Power series carrier prepend-unary closure)**.** *If (a, z_0) is a power series carrier and q is unary, then prepending q to the origin and to every coefficient gives another power series carrier:*

$$\mathsf{PowerSeriesCarrier}(\lambda n.\mathsf{append}(q, a_n), \mathsf{append}(q, z_0)).$$

Proof. Unpack the power-series carrier into the unary origin, coefficient-wise complex-history carrier, and complex-history origin carrier. Unary histories are closed under append, and complex-history carriers are closed under prepend by a unary history; apply the latter pointwise to every coefficient and once to the origin. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesCarrier prepend unary closed`

Definition 66.5 (Partial sum at a point)**.** *For a power series (a, z_0) and a complex history z, the partial sum predicate $\mathsf{PartSum}(a, z_0, z, n, S_n)$ is the unary-indexed family of complex histories obtained by summing the first n terms $a_k(z - z_0)^k$ via explicit unary recursion, using the complex multiplication and addition of chapter 39.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesPartSum`

Lemma 66.6 (Partial sums are complex histories)**.** *For every power series (a, z_0), every complex history z, and every unary n, there is a unique complex history S_n with $\mathsf{PartSum}(a, z_0, z, n, S_n)$, up to hsame on both real components.*

Proof. By unary induction on n, using totality and functionality of complex multiplication and addition. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.PowerSeriesCarrier constant coeff partSum exists unique`

66.2 Geometric bound predicate

Definition 66.7 (Geometric bound)**.** *For coefficients a, unary radius history r, and unary constant K, the geometric bound $\mathsf{GeomBound}(a, r, K)$ holds when for every unary n, the modulus comparison $|a_n| \cdot r^n \leq K$ is decidably true on the rational representation.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.GeomBound`

Lemma 66.8 (Geometric bound visible radius endpoint package)**.** *For coefficients a, constant history K, result history R, and tail history tail, if*

$$\mathsf{GeomBound}(a, \mathsf{E1}(tail), K) \qquad \textit{and} \qquad \mathsf{Cont}(\mathsf{E1}(tail), K, R),$$

then tail is unary and the continuation result is not internally empty:

$$\mathsf{UnaryHistory}(tail) \wedge \neg\mathsf{hsame}(R, \mathsf{e}).$$

Proof. The geometric-bound witness contains the unary carrier for the visible radius $\mathsf{E1}(tail)$, so unary inversion gives the payload carrier for *tail*. If the continuation result were empty, empty-result inversion would identify the visible radius endpoint $\mathsf{E1}(tail)$ with e, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound visible radius endpoint package`

Theorem 66.9 (Geometric bound visible constant endpoint package)**.** *For coefficients a, radius history r, result history R, and tail history tail, if*

$$\mathsf{GeomBound}(a, r, \mathsf{E1}(tail)) \qquad \textit{and} \qquad \mathsf{Cont}(r, \mathsf{E1}(tail), R),$$

then tail is unary and the continuation result is not internally empty:

$$\mathsf{UnaryHistory}(tail) \wedge \neg\mathsf{hsame}(R, \mathsf{e}).$$

Proof. The geometric-bound witness contains the unary carrier for the visible constant endpoint $\mathsf{E1}(tail)$, so unary inversion gives the payload carrier for *tail*. If the continuation result were empty, empty-result inversion would identify the visible constant endpoint $\mathsf{E1}(tail)$ with e, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound visible constant endpoint package`

Theorem 66.10 (Geometric bound radius semantic naming certificate)**.** *For fixed coefficients a and unary bound constant K, any witness of $\mathsf{GeomBound}(a, r, K)$ yields a semantic naming certificate whose carrier, pattern, and ledger are the same radius-indexed geometric-bound predicate and whose classifier is* hsame *on the displayed radius.*

Proof. The witnessed radius r inhabits the carrier. Reflexivity, symmetry, and transitivity are the finite-kernel hsame laws. Carrier transport moves only the unary radius component across the displayed hsame comparison and leaves the fixed coefficient and bound data unchanged. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound radius semanticNameCert`

Theorem 66.11 (Geometric bound semantic naming certificate)**.** *If $\mathsf{GeomBound}(a, r, K)$ holds, then the radius-indexed carrier $x \mapsto \mathsf{GeomBound}(a, x, K)$, with classifier* hsame*, forms a semantic naming certificate whose source, pattern, and ledger predicates are all this same concrete geometric-bound carrier.*

Proof. The displayed bound supplies the inhabited carrier witness. Reflexivity, symmetry, and transitivity are the kernel laws for hsame. Carrier transport along hsame transports only the unary radius field; the constant K and the coefficient carrier rows are preserved unchanged. Pattern and ledger soundness are direct because all three semantic predicates are the same geometric-bound carrier. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound semanticNameCert`

Lemma 66.12 (Geometric bound supplies the power series carrier)**.** *If $\mathsf{GeomBound}(a, r, K)$ holds and the expansion center z_0 is a complex history, then (a, z_0) is a power series carrier, and both r and K are unary histories.*

Proof. Unpack the geometric-bound predicate. Its fields give the unary radius, the unary constant, and the coefficient-wise complex carrier; combine the latter with the displayed carrier of z_0. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound powerSeriesCarrier`

Lemma 66.13 (Geometric bound radius and constant continuation closure)**.** *For coefficients a, radius histories r, R, constant histories K, K', and unary perturbations dr, dK, if*

$$\mathsf{GeomBound}(a, r, K), \qquad \mathsf{UnaryHistory}(dr), \qquad \mathsf{UnaryHistory}(dK),$$

and the continuation ledgers $\mathsf{Cont}(r, dr, R)$ and $\mathsf{Cont}(K, dK, K')$ hold, then the geometric bound persists at the transported radius and constant:

$$\mathsf{GeomBound}(a, R, K').$$

Proof. The geometric-bound witness contains unary carrier data for both r and K. Unary continuation closure transports those carriers along the two displayed continuation ledgers, while the coefficient carrier clause is unchanged. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound radius constant continuation closed`

Theorem 66.14 (Geometric bound radius and constant hsame transport)**.** *For coefficients a, radius histories r, r', and constant histories K, K', if*

$$r\mathsf{hsame}r', \qquad K\mathsf{hsame}K', \qquad \mathsf{GeomBound}(a, r, K),$$

then r' and K' are unary histories and

$$\mathsf{GeomBound}(a, r', K').$$

Proof. Unpack the geometric-bound witness into its unary radius field, unary constant field, and coefficient carrier family. Transport the two unary fields along the displayed hsame classifiers, and reuse the unchanged coefficient carrier family. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound radius constant hsame transport unary`

Theorem 66.15 (Geometric bound continuation visible radius endpoint package)**.** *For coefficients a, radius histories r, R, constant histories K, K', unary perturbations dr, dK, and visible tail $tail$, if*

$$\mathsf{GeomBound}(a, r, K), \qquad \mathsf{UnaryHistory}(dr), \qquad \mathsf{UnaryHistory}(dK),$$

and

$$\mathsf{Cont}(r, dr, \mathsf{E1}(tail)), \qquad \mathsf{Cont}(K, dK, K'), \qquad \mathsf{Cont}(\mathsf{E1}(tail), K', R),$$

then the visible tail is unary and the final endpoint is not internally empty:

$$\mathsf{UnaryHistory}(tail) \wedge \neg\mathsf{hsame}(R, \mathsf{e}).$$

Proof. First apply radius and constant continuation closure to transport the geometric bound from (r, K) to $(\mathsf{E1}(tail), K')$. The visible-radius endpoint package then gives unary readback for $tail$ and excludes an empty final endpoint. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound continuation visible radius endpoint package`

Lemma 66.16 (Geometric majorant inside the disk)**.** *If $\mathsf{GeomBound}(a, r, K)$ holds and z satisfies $|z - z_0| \le r' < r$ (decidable strict inequality on rational representations), then the partial sums of a at z form a regular complex Cauchy sequence with explicit modulus $N(k) = \lceil k/\log_2(r/r') \rceil$ computed as a unary function of the precision.*

Sketch. Each term is bounded above by $K(r'/r)^n$; apply the geometric series formula with the explicit ratio. □

66.3 Convergence radius predicate

Definition 66.17 (Convergence radius)**.** *For coefficients a, the convergence radius predicate* $\mathsf{ConvRad}(a, R)$ *holds when there is a constant-witness function $K : \mathsf{Unary} \to \mathsf{Unary}$ such that for every unary radius r with* $\mathsf{NatUnaryStrictPrefix}(r, R)$*,* $\mathsf{GeomBound}(a, r, K(r))$ *is satisfied.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRad`

Lemma 66.18 (Convergence radius supplies power series carriers along witnesses)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds and z_0 is a complex history, then the constant-witness function K can be chosen so that every admitted radius row simultaneously gives* $\mathsf{PowerSeriesCarrier}(a, z_0)$ *and* $\mathsf{GeomBound}(a, r, K(r))$*.*

Proof. Open the witness function in the convergence-radius predicate. For each admitted radius row, apply Definition 66.12 to the resulting geometric bound and the carrier of z_0. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad powerSeriesCarrier witness`

Lemma 66.19 (Convergence radius append-unary power-series witness)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds, z_0 is a complex history, and q is unary, then the convergence-radius witness function K can be chosen so that every admitted radius row simultaneously gives*

$$\mathsf{PowerSeriesCarrier}(\lambda n.\mathsf{append}(a_n, q), \mathsf{append}(z_0, q))$$

and

$$\mathsf{GeomBound}(\lambda n.\mathsf{append}(a_n, q), r, K(r)).$$

Proof. Choose the witness function supplied by Definition 66.18. For each admitted radius row, apply append-unary closure to the supplied power-series carrier and apply geometric-bound append-unary coefficient closure to the supplied geometric bound. The same K remains the witness function. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad powerSeriesCarrier append unary witness`

Lemma 66.20 (Convergence radius prepend-unary power-series witness)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds, z_0 is a complex history, and q is unary, then the convergence-radius witness function K can be chosen so that every admitted radius row simultaneously gives*

$$\mathsf{PowerSeriesCarrier}(\lambda n.\mathsf{append}(q, a_n), \mathsf{append}(q, z_0))$$

and

$$\mathsf{GeomBound}(\lambda n.\mathsf{append}(q, a_n), r, K(r)).$$

Proof. Choose the witness function supplied by Definition 66.18. For each admitted radius row, apply prepend-unary closure to the supplied power-series carrier and apply geometric-bound prepend-unary coefficient closure to the supplied geometric bound. The same K remains the witness function. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad powerSeriesCarrier prepend unary witness`

Theorem 66.21 (Convergence radius visible radius witness endpoint package)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds and* $\mathsf{E1}(tail)$ *is an admitted unary radius endpoint, then the convergence-radius witness function can be chosen so that any continuation*

$$\mathsf{Cont}(\mathsf{E1}(tail), K(\mathsf{E1}(tail)), R)$$

exposes the payload carrier and non-empty result boundary:

$$\mathsf{UnaryHistory}(tail) \wedge \neg\mathsf{hsame}(R, \mathsf{e}).$$

Proof. Open the witness function in $\mathsf{ConvRad}(a, R)$ and specialize it to the visible radius endpoint $\mathsf{E1}(tail)$. The resulting geometric-bound row, together with the displayed continuation, is exactly the visible-radius endpoint package for geometric bounds. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad visible radius witness endpoint package`

Theorem 66.22 (Convergence radius classifier-visible radius witness endpoint package)**.** *If* $\mathsf{hsame}(R, R')$*,* R' *is unary, the coefficient streams* a *and* b *are complex-history classified pointwise,* $\mathsf{ConvRad}(a, R)$ *holds, and* $\mathsf{E1}(tail)$ *is an admitted unary radius endpoint, then the transported convergence-radius witness function can be chosen so that every continuation*

$$\mathsf{Cont}(\mathsf{E1}(tail), K(\mathsf{E1}(tail)), R')$$

exposes

$$\mathsf{UnaryHistory}(tail) \wedge \neg\mathsf{hsame}(R', \mathsf{e}).$$

Proof. Transport the convergence radius along $\mathsf{hsame}(R, R')$ and the pointwise complex-history classifier for the coefficients. The visible-radius endpoint package then applies at the transported radius R'. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad classifier visible radius witness endpoint package`

Lemma 66.23 (Convergence radius transport)**.** *If* $\mathsf{hsame}(R, R')$*,* $\mathsf{ConvRad}(a, R)$*, and* R' *is unary, then the same geometric-bound witness function supplies* $\mathsf{ConvRad}(a, R')$*.*

Proof. Transport the radius endpoint along hsame. The unary carrier for the target radius is explicit, and the existing constant-witness function continues to provide the same geometric-bound data. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad radius transport`

Lemma 66.24 (Convergence radius coefficient classifier transport)**.** *If* $\mathsf{hsame}(R, R')$*,* R' *is unary, every coefficient* a_n *is classified with* b_n *by the complex-history classifier, and* $\mathsf{ConvRad}(a, R)$ *holds, then* R' *is unary and* $\mathsf{ConvRad}(b, R')$ *holds.*

Proof. First transport the radius endpoint from R to R', retaining the same constant-witness function for geometric bounds. For each supplied geometric bound, keep the radius and constant unary carriers, and use the target-carrier component of the complex-history classifier at every coefficient index to rebuild the coefficient side of the bound for b. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad radius coefficient classifier transport`

Theorem 66.25 (Convergence radius is unique)**.** *If* $\mathsf{ConvRad}(a, R)$ *and* $\mathsf{ConvRad}(a, R')$ *both hold, then* $\mathsf{hsame}(R, R')$ *on the rational representation.*

Sketch. Suppose $R \neq R'$. Pick a rational r strictly between them. $\mathsf{GeomBound}$ at r follows from one side and fails from the other, contradicting consistency of the witness. □

66.4 Opening Boundary

Theorem 66.26 (Convergence-radius opening boundary)**.** *The displayed witness* $\mathsf{ConvRad}(a, R)$ *supports absolute convergence for every strict subradius named by the geometric-bound ledger: whenever a unary radius* r *is displayed with* $r < R$ *and a finite witness* K *satisfies* $\mathsf{GeomBound}(a, r, K)$*, the power-series carrier admits the corresponding majorant read inside* r*. Cauchy–Hadamard limsup exactness is a separate analytic ledger row and is not exported by the checked-row reduct.*

Sketch. Unpack $\mathsf{ConvRad}(a, R)$ as the family of strict-subradius geometric bound witnesses. For a displayed $r < R$, the witness K gives the finite majorant row needed by Definition 66.16 and by the power-series carrier readback.

The checked-row reduct keeps these carrier, transport, and geometric-bound rows. It has no constructor for limsup exactness, so the Cauchy–Hadamard row can only be supplied through the separate analytic ledger policy. □

Theorem 66.27 (Absolute convergence inside the disk)**.** *If* $\mathsf{ConvRad}(a, R)$ *and* z *satisfies* $|z - z_0| < R$ *(strict inequality with explicit witness), then the power series* $\sum a_n(z - z_0)^n$ *converges absolutely with explicit modulus.*

Sketch. By Definition 66.16: pick a rational r with $|z - z_0| < r < R$; the series is bounded by a geometric majorant. □

66.5 Closure laws

Lemma 66.28 (Geometric bound classifier append-unary closure)**.** *If coefficient families* a *and* b *are pointwise classified as complex histories,* $\mathsf{GeomBound}(a, r, K)$ *holds, and* q *is unary, then appending* q *to every displayed* b_n *preserves the same geometric-bound radius and constant:*

$$\mathsf{GeomBound}(\lambda n.\mathsf{append}(b_n, q), r, K).$$

Proof. Each classifier row exposes the target coefficient b_n as a complex-history carrier. Complex-history carriers are closed under append of a unary suffix, so each appended coefficient remains in the carrier. The radius and constant unary witnesses are the ones already carried by $\mathsf{GeomBound}(a, r, K)$. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound coeff classifier append unary closed`

Lemma 66.29 (Geometric bound append-unary coefficient closure)**.** *If* $\mathsf{GeomBound}(a, r, K)$ *holds and* q *is unary, then appending* q *to every coefficient preserves the same radius and constant:*

$$\mathsf{GeomBound}(\lambda n.\mathsf{append}(a_n, q), r, K).$$

Proof. Unpack $\mathsf{GeomBound}(a, r, K)$ into its unary radius witness, unary constant witness, and coefficient carrier family. Apply complex-history carrier append-unary closure pointwise to each coefficient and keep the same radius and constant witnesses. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound append unary coeff closed`

Lemma 66.30 (Geometric bound prepend-unary coefficient closure)**.** *If* q *is unary and* $\mathsf{GeomBound}(a, r, K)$ *holds, then prepending* q *to every coefficient preserves the same radius and constant:*

$$\mathsf{GeomBound}(\lambda n.\mathsf{append}(q, a_n), r, K).$$

Proof. Unpack $\mathsf{GeomBound}(a, r, K)$ into its unary radius witness, unary constant witness, and coefficient carrier family. Apply complex-history carrier prepend-unary closure pointwise to each coefficient and keep the same radius and constant witnesses. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound prepend unary coeff closed`

Theorem 66.31 (Convergence radius append-unary coefficient closure)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds and* q *is unary, then appending* q *to every coefficient preserves the same convergence radius:*

$$\mathsf{ConvRad}(\lambda n.\mathsf{append}(a_n, q), R).$$

Proof. Unfold $\mathsf{ConvRad}(a, R)$ and keep its constant-witness function K as the candidate witness for the appended coefficient family.

Take an arbitrary admitted radius row, together with its continuation ledger $\mathsf{Cont}(r, K(r), R)$. The original convergence-radius witness gives $\mathsf{GeomBound}(a, r, K(r))$ for that row. Apply Definition 66.29 to this geometric bound and the unary suffix q, yielding $\mathsf{GeomBound}(\lambda n.\mathsf{append}(a_n, q), r, K(r))$. Since the row was arbitrary and the witness function is unchanged, the displayed predicate is $\mathsf{ConvRad}(\lambda n.\mathsf{append}(a_n, q), R)$. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad append unary coeff closed`

Theorem 66.32 (Convergence radius prepend-unary coefficient closure)**.** *If* $\mathsf{ConvRad}(a, R)$ *holds and* q *is unary, then prepending* q *to every coefficient preserves the same convergence radius:*

$$\mathsf{ConvRad}(\lambda n.\mathsf{append}(q, a_n), R).$$

Proof. Unfold $\mathsf{ConvRad}(a, R)$ and keep its constant-witness function K as the candidate witness for the prepended coefficient family. For any admitted row r with continuation ledger $\mathsf{Cont}(r, K(r), R)$, the original witness gives $\mathsf{GeomBound}(a, r, K(r))$. Apply Definition 66.30 with the unary prefix q, and leave the radius witness, the continuation row, and the constant witness unchanged. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad prepend unary coeff closed`

Lemma 66.33 (Differentiation preserves radius)**.** *If* $\mathsf{ConvRad}(a, R)$*, then the term-wise derivative coefficients* $b_n = (n+1) \cdot a_{n+1}$ *satisfy* $\mathsf{ConvRad}(b, R)$ *with the same radius and an explicit constant adjustment.*

Sketch. For $r' < r < R$, $|b_n|r'^n \leq (n+1)|a_{n+1}|r'^n$; pull out a factor of $(r'/r)^n$ using $\mathsf{GeomBound}(a, r, K)$ and bound $n(r'/r)^n$ by an explicit polynomial-times-geometric majorant. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad successor coefficients closed`
Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad coefficient tail closed`

Lemma 66.34 (Cauchy product radius)**.** *If* $\mathsf{ConvRad}(a, R_a)$ *and* $\mathsf{ConvRad}(b, R_b)$*, then the Cauchy product* $c_n = \sum_{k=0}^{n} a_k b_{n-k}$ *satisfies* $\mathsf{ConvRad}(c, R)$ *with* $R \geq \min(R_a, R_b)$ *and explicit* $K(r)$ *computed from* K_a, K_b*.*

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound cauchy product row closed`

Sketch. Triangular pasting of geometric bounds; constant inflation by a factor of $(n+1)$ absorbed into the geometric majorant. □

66.6 The certificate

Definition 66.35 (Convergence radius source specification)**.** *The source specification is the predicate* ConvRad(a, R) *on power series carriers paired with a unary radius candidate.*

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadSourceSpec powerSeries append prepend closed`
Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadSourceSpec powerSeries geomBound readback`

Definition 66.36 (Convergence radius pattern specification)**.** *The pattern is the supremum-of-geometric-bound predicate, computed through* GeomBound*.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadPatternSpec`

Definition 66.37 (Convergence radius classifier specification)**.** *Two convergence radii are classified together when they agree under* hsame *on the rational representation of the radius.*

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadRadiusClassifierSpec powerSeries witness`
Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadClassifierSpec`
Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadClassifierSpec radius transport`

Definition 66.38 (Convergence radius stability certificate)**.** *The stability certificate consists of* hsame*-respect of* ConvRad *on coefficient input, transport of constant-witness functions, and compatibility with the geometric-bound predicate under* hsame *on radii.*

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad stability certificate fields`

Definition 66.39 (Convergence radius ledger policy)**.** *The ledger policy records the analytic obligations for term-by-term differentiation, Cauchy product, absolute convergence inside the open disk, and the Cauchy-Hadamard limsup characterization.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadLedgerPolicy`

Theorem 66.40 (Concrete ConvRad$^{\uparrow}$ certificate-obligation package)**.** *The convergence-radius construction supplies a concrete certificate-obligation package over* Hist *for* ConvRad$^{\uparrow}$*. Its checked rows are the* ConvRad *carrier,* hsame *classifier, radius transport, coefficient classifier transport, extraction of power-series carrier data from geometric bounds, the convergence-radius power-series carrier witness, and append-unary coefficient closure. The analytic ledger laws of Definition 66.39 remain policy obligations for this interface.*

Sketch. The carrier and classifier are the displayed ConvRad predicate and hsame on radius histories.

The checked transport rows are Definition 66.23 and Definition 66.24. The carrier witness rows are Definition 66.12 and Definition 66.18. The checked closure row is Definition 66.31.

These rows give the concrete proof obligations available to the semantic certificate target. The differentiation, Cauchy-product, absolute-convergence, and Cauchy-Hadamard rows are ledger-policy obligations in this chapter, not premises of the checked semantic NameCert target. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.conv rad name certificate`

Theorem 66.41 (Semantic ConvRad$^{\uparrow}$ naming certificate)**.** *The concrete package of Definition 66.40 supplies the checked semantic naming-certificate schema of* NameCert *(chapter 9) exactly for the* ConvRad *carrier, the* hsame *radius classifier, radius transport, coefficient classifier transport, the power-series carrier witness, and append-unary coefficient closure.*

Sketch. The witnessed radius inhabits the ConvRad carrier. The equivalence fields are the finite-kernel hsame laws.

Carrier transport is the radius-transport row, with unary transport of the displayed radius. Coefficient-family transport uses the checked classifier transport row, while the carrier witness and append-unary closure are supplied by their corresponding checked rows in this chapter. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad semanticNameCert`

Theorem 66.42 (Geometric bound is downward closed in the radius)**.** *For coefficients a, radius histories r' and r, and constant history K, if*

$$\mathsf{NatUnaryStrictPrefix}(r', r) \qquad \textit{and} \qquad \mathsf{GeomBound}(a, r, K),$$

then

$$\mathsf{GeomBound}(a, r', K).$$

Proof. By Definition 28.10, choose a unary tail t such that t is not e and $\mathsf{Cont}(r', t, r)$ holds. By Definition 66.7, the bound at r supplies the unary carrier for r, the unary carrier for K, the coefficient carrier rows, and the decidable comparison rows $|a_n| \cdot r^n \leq K$ for every unary index n. The strict-prefix witness is in particular a prefix witness from r' to r; applying Definition 52.22 to that witness and to the unary carrier of r gives the unary carrier of r'.

It remains to transport the comparison rows. In the rational representation of unary histories, the continuation ledger $\mathsf{Cont}(r', t, r)$ reads $r = r' + t$ with $t \geq 0$, hence $0 \leq r' \leq r$. Fix a unary index n. Unary induction on n gives $(r')^n \leq r^n$: the empty exponent gives equality of units, and the one-extension step multiplies the induction comparison by the nonnegative factor r' and then uses $r' \leq r$ with the nonnegative factor r^n. Since complex moduli are nonnegative, rational-history multiplication gives

$$|a_n| \cdot (r')^n \leq |a_n| \cdot r^n.$$

Composing this inequality with the comparison row supplied by $\mathsf{GeomBound}(a, r, K)$ yields $|a_n| \cdot (r')^n \leq K$. The constant carrier and coefficient carrier rows are unchanged, so Definition 66.7 repacks these data as $\mathsf{GeomBound}(a, r', K)$. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.GeomBound radius shrink closed`

Definition 66.43 (Convergence-radius compatible coefficient-ring inclusion)**.** *Let R and R' be coefficient sources equipped with $\mathsf{CommRing}^{\uparrow}$ certificates as in Definition 44.59. Write their carriers and classifiers as $C_R, \sim_R$ and $C_{R'}, \sim_{R'}$, their operations as $0_R, 1_R, +_R, -_R, \cdot_R$ and $0_{R'}, 1_{R'}, +_{R'}, -_{R'}, \cdot_{R'}$, and let γ_R and $\gamma_{R'}$ be complex-realization maps into $\mathsf{Complex}^{\uparrow}$ histories. A convergence-radius compatible coefficient-ring inclusion*

$$\iota : R \hookrightarrow R'$$

consists of the following data:

1. *Carrier and classifier rows: $C_R(x)$ implies $C_{R'}(\iota x)$; $x \sim_R y$ implies $\iota x \sim_{R'} \iota y$; and, for carried x, y, $\iota x \sim_{R'} \iota y$ implies $x \sim_R y$.*
2. *Operation rows: for carried inputs, $\iota(0_R) \sim_{R'} 0_{R'}$, $\iota(1_R) \sim_{R'} 1_{R'}$, $\iota(x +_R y) \sim_{R'} \iota x +_{R'} \iota y$, $\iota(-_R x) \sim_{R'} -_{R'} \iota x$, and $\iota(x \cdot_R y) \sim_{R'} \iota x \cdot_{R'} \iota y$.*
3. *Complex-realization rows: $\gamma_R(x)$ and $\gamma_{R'}(\iota x)$ are complex histories, and every carried x satisfies*

$$\mathsf{ComplexHistoryClassifier}(\gamma_R(x), \gamma_{R'}(\iota x)).$$

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadCoefficientRingInclusion`

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCoefficientRingInclusion pointwise classifier`

Theorem 66.44 (Convergence radius monotonicity under coefficient-ring inclusion)**.** *Let R and R' be* $\mathsf{CommRing}^{\uparrow}$ *coefficient sources and let $\iota : R \hookrightarrow R'$ be a convergence-radius compatible coefficient-ring inclusion from Definition 66.43. For a pointwise carried formal coefficient sequence $f : \mathsf{Nat}^{\uparrow} \to R$ as in Definition 51.1, put*

$$a_n := \gamma_R(f_n), \qquad a'_n := \gamma_{R'}(\iota(f_n)).$$

If ρ is a unary radius history and $\mathsf{ConvRad}(a, \rho)$ *holds, then*

$$\mathsf{ConvRad}(a', \rho).$$

Thus every convergence-radius witness admitted by the R-view is also admitted by the R'-view.

Proof. For each unary index n, pointwise carriedness gives $C_R(f_n)$. The complex-realization row in Definition 66.43 therefore supplies

$$\mathsf{ComplexHistoryClassifier}(a_n, a'_n).$$

The operation rows in the inclusion identify a' as the coefficient sequence obtained by reading the same formal series in the larger coefficient source; these are the finite algebraic rows used by the partial-sum recursion of Definition 66.5. The convergence-radius argument only needs the resulting pointwise complex classifier family.

The radius endpoint is unchanged. Hence the radius-classifier input for Definition 66.24 is the reflexive history-sameness witness $\mathsf{hsame}(\rho, \rho)$, and the target-radius carrier input is the displayed unary witness for ρ. Applying that lemma to the coefficient families a and a' gives $\mathsf{ConvRad}(a', \rho)$.

Equivalently, unpacking Definition 66.17, the same constant-witness function K can be retained. For every strict subradius r of ρ, the source row gives $\mathsf{GeomBound}(a, r, K(r))$; the coefficient-classifier transport in Definition 66.24 sends this geometric-bound row to $\mathsf{GeomBound}(a', r, K(r))$, while the radius and constant histories are unchanged as in Definition 66.7. These rows reassemble the convergence-radius predicate for a' at ρ. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad coefficient ring inclusion monotone`

66.7 Cauchy–Hadamard Boundary

Definition 66.45 (Cauchy–Hadamard exactness row)**.** *For a coefficient family a and a unary radius history R, let* $\mathsf{CHExact}(a, R)$ *denote the analytic ledger row asserting that the* $\mathsf{ConvRad}(a, R)$ *endpoint is exact for the Cauchy–Hadamard limsup expression,*

$$R = 1/\mathsf{LimSup}_n |a_n|^{1/n}.$$

This is the Cauchy–Hadamard item in Definition 66.39. It is a ledger-policy row, not one of the checked carrier, classifier, transport, or closure rows of Definition 66.40.

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadCauchyHadamardExactnessRow`

Definition 66.46 (Convergence-radius checked-row reduct)**.** *The checked-row reduct of the* $\mathsf{ConvRad}^{\uparrow}$ *package is the finite row set $\mathcal{Q}_{\mathrm{chk}}$ consisting of the following data:*

1. *the carrier row* $\mathsf{ConvRad}(a, R)$ *and the* hsame *classifier on radius histories;*
2. *radius transport, as in Definition 66.23;*
3. *coefficient-family classifier transport, as in Definition 66.24;*
4. *extraction of power-series carrier data from geometric bounds and from convergence-radius witnesses, as in Definition 66.12 and Definition 66.18;*
5. *append-unary coefficient closure, as in Definition 66.31.*

A policy interpretation over this reduct is a row assignment Π *with witnesses for every row in* $\mathcal{Q}_{\mathrm{chk}}$*. The assignment is constrained only on* $\mathcal{Q}_{\mathrm{chk}}$*; analytic ledger rows such as* $\mathsf{CHExact}(a, R)$ *are not constrained unless the ledger policy of Definition 66.39 is also supplied.*

Lean4 (definition) `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct`

Theorem 66.47 (Checked convergence-radius rows do not force Cauchy–Hadamard)**.** *Fix a coefficient family* a *and a unary radius history* R *for which the checked-row reduct of Definition 66.46 has the displayed carrier row* $\mathsf{ConvRad}(a, R)$*. In the* $\mathsf{ConvRad}^{\uparrow}$ *setup reduced to that checked row set, there are two policy interpretations* Π_+ *and* Π_- *over the same history carrier and the same radius classifier such that*

$$\forall q \in \mathcal{Q}_{\mathrm{chk}}, \quad \Pi_+(q) = \Pi_-(q),$$

but

$$\Pi_+(\mathsf{CHExact}(a, R)) = \mathsf{accept} \qquad \textit{and} \qquad \Pi_-(\mathsf{CHExact}(a, R)) = \mathsf{reject}.$$

Consequently, no finite derivation whose nonlogical inputs are only the checked rows of Definition 66.46 can prove $\mathsf{CHExact}(a, R)$*.*

Proof. By Definition 9.6, the naming-certificate template separates source, pattern, classifier, stability, and ledger fields. The concrete convergence-radius package in Definition 66.40 lists the checked carrier, classifier, transport, witness, and append-unary closure rows, and states that the analytic laws of Definition 66.39 remain ledger-policy obligations.

Let $\chi_{a,R}$ be the row name represented by $\mathsf{CHExact}(a, R)$. Since $\chi_{a,R}$ is not an element of $\mathcal{Q}_{\mathrm{chk}}$, define Π_+ and Π_- to use the same witness for every checked row $q \in \mathcal{Q}_{\mathrm{chk}}$. On the Cauchy–Hadamard row, set $\Pi_+(\chi_{a,R}) = \mathsf{accept}$ and $\Pi_-(\chi_{a,R}) = \mathsf{reject}$; all other analytic row names may be assigned identically in the two interpretations.

Every premise and conclusion of a checked row is read from $\mathcal{Q}_{\mathrm{chk}}$, so the two interpretations validate exactly the same checked carrier, classifier, transport, witness, and closure data. They therefore share the full checked-row reduct. They disagree only on the analytic ledger row $\mathsf{CHExact}(a, R)$, which the reduct does not mention.

If the checked rows alone forced $\mathsf{CHExact}(a, R)$, any two interpretations that agree on all rows used by the derivation would agree on that conclusion. Applying this to Π_+ and Π_- gives the displayed contradiction. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad checked rows do not force cauchy hadamard`

Theorem 66.48 (Cauchy–Hadamard sibling route)**.** *Any consumer that upgrades the checked* $\mathsf{ConvRad}^{\uparrow}$ *reduct to a Cauchy–Hadamard exactness row must pass through the sibling NameCert surfaces for* $\mathsf{CauchyHadamard}^{\uparrow}$*,* $\mathsf{RealPowerSeries}^{\uparrow}$*, and* $\mathsf{WeierstrassMTest}^{\uparrow}$*, together with the local convergence-radius rows. The local rows of Definition 66.46 expose the radius carrier, classifier, transport, power-series extraction, and append-unary coefficient closure; the sibling surfaces supply*

the missing Cauchy–Hadamard ledger, power-series endpoint route, and majorant-tail route. The resulting route is a horizontal dependency through chapter 3466, chapter 1583, and chapter 399, not an import of host limsup, quotient real equality, or ambient analytic completeness.

Proof. Apply Definition 66.47. It exhibits two policy interpretations that agree on every row of $\mathcal{Q}_{\mathrm{chk}}$ and disagree on $\mathsf{CHExact}(a, R)$, so the checked convergence-radius rows alone cannot force Cauchy–Hadamard exactness.

The missing row is analytic ledger data. The $\mathsf{CauchyHadamard}^{\uparrow}$ sibling provides the named Cauchy–Hadamard surface, while the $\mathsf{RealPowerSeries}^{\uparrow}$ sibling keeps coefficient windows, partial sums, radius reads, and endpoint seals visible as finite packet rows.

Finally the $\mathsf{WeierstrassMTest}^{\uparrow}$ sibling supplies the majorant-tail route used by power-series consumers before a real endpoint seal is exported. Combining these sibling rows with the local convergence-radius reduct records the exact horizontal route, without replacing it by host limsup, quotient real equality, or an unrecorded completeness theorem. □

Theorem 66.49 (Convergence radius coefficient-ring inclusion geometric-bound transport)**.** *Let*

$$\iota : R \hookrightarrow R'$$

be a convergence-radius compatible coefficient-ring inclusion in the sense of Definition 66.43. For a source coefficient family a and its transported family a' defined pointwise by the complex realization rows of ι, if

$$\mathsf{GeomBound}(a, r, K),$$

then

$$\mathsf{GeomBound}(a', r, K).$$

The radius history r and constant history K are unchanged.

Proof. Unfold the geometric-bound rows of Definition 66.7. The source bound supplies the coefficient carrier rows, the radius carrier row, the constant carrier row, and the pointwise comparison ledger for every unary index.

Use Definition 66.43 on each coefficient row. Its complex-realization clause identifies a_n with a'_n through the transported complex-history classifier, and its pointwise classifier transport keeps the displayed coefficient read in the target source.

Therefore the comparison row for a_n at radius r and constant K reads as the corresponding comparison row for a'_n. The radius and constant histories are not transported, so their carrier rows and ledger entries are copied unchanged. Repacking these rows by Definition 66.7 gives $\mathsf{GeomBound}(a', r, K)$. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCoefficientRingInclusion geomBound transport`

Theorem 66.50 (ConvergenceRadius checked-row reduct readback)**.** *For any coefficient family a, center z_0, and unary radius R, a $\mathsf{ConvRadSourceSpec}(a, z_0, R)$ row determines the checked-row reduct of Definition 66.46. The reduct consists of the source row together with the geometric-bound readback rows supplied by the source specification, and it contains no Cauchy–Hadamard exactness row.*

Proof. Unfold the source specification of Definition 66.35. Its displayed data give the coefficient family, center, unary radius, and the geometric-bound readback rows associated to the same source packet.

Now apply Definition 66.46. The reduct keeps exactly the checked rows read from that packet: the source row and the geometric-bound rows already exposed by the source specification.

There is no constructor in the reduct for Cauchy–Hadamard exactness. Repacking the displayed source and geometric-bound rows therefore gives precisely the checked-row reduct claimed above. □

Theorem 66.51 (ConvergenceRadius finite-prefix window stability)**.** *Let* ConvRadSourceSpec(a, z_0, R) *be a displayed convergence-radius source packet. Restrict the packet to a certified finite unary prefix window of coefficient indices and consumer reads. Then the checked-row reduct visible to downstream consumers preserves the same radius witness, the geometric-bound rows on the displayed prefix, the endpoint packages,* BHist *histories,* hsame *transport rows,* Cont *routes, and* Pkg *provenance.*

The restricted reduct introduces no Cauchy–Hadamard exactness row, no real-supremum row, and no host complex-analysis boundary beyond the rows already visible in the finite prefix.

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct prefix window stability`

Proof. Unfold Definition 66.35 and Definition 66.46. The source packet already displays the coefficient family, center, unary radius witness, geometric-bound rows, endpoint packages, transport rows, continuation routes, and provenance.

Project the rows whose coefficient indices lie in the certified finite prefix. The radius witness and endpoint packages are shared coordinates of the source packet, while the geometric-bound rows are retained only where the prefix contains the corresponding unary index.

Transport the retained entries componentwise by hsame. Continuation determinacy for the displayed Cont routes keeps the read paths attached to the same packet, and the Pkg row records the unchanged provenance.

Finally use Definition 66.54 only to identify which retained rows are consumer-visible. That boundary has no constructor for Cauchy–Hadamard exactness or real supremum, so the finite prefix reduct exposes exactly the listed checked rows. □

Theorem 66.52 (ConvergenceRadius SOne ComplexLimit consumer boundary)**.** *For a* $\mathsf{ConvRad}^{\uparrow}$ *source packet restricted to a certified finite unary prefix window, any* $\mathsf{S}^{1\uparrow}$ *or* $\mathsf{CplxLim}^{\uparrow}$ *consumer reads only the checked-row reduct, the geometric-bound rows on the displayed prefix, the* $\mathsf{CplxLim}^{\uparrow}$ *modulus rows, the* $\mathsf{S}^{1\uparrow}$ *unit-equation consumer rows,* BHist *histories,* hsame *transports,* Cont *routes, and* Pkg *provenance.*

The consumer receives no Cauchy–Hadamard exactness row, no real supremum, no host complex-analysis boundary, and no choice-based radius extraction.

Proof. Begin with the checked-row reduct readback of Definition 66.50. It identifies the source row and geometric-bound readbacks as the rows retained from the convergence-radius source packet.

Restrict to the certified finite unary prefix and apply Definition 66.51. The restricted consumer surface keeps the same radius witness, retained geometric-bound rows, endpoint packages, BHist histories, hsame transports, Cont routes, and Pkg provenance, and it adds no Cauchy–Hadamard or real-supremum row.

Now read the two sibling consumers through their own finite boundaries. The $\mathsf{CplxLim}^{\uparrow}$ semantic certificate supplies only modulus rows on displayed complex histories, while the $\mathsf{S}^{1\uparrow}$ componentwise RealUp and metric boundary supplies only unit-equation consumer rows. Neither sibling boundary adds a host complex-analysis theorem or a radius extractor, so the combined consumer reads exactly the listed finite-prefix rows. □

Theorem 66.53 (ConvergenceRadius public export)**.** *Definition 66.41 exports the public* $\mathsf{ConvRad}^{\uparrow}$ *boundary over the power-series carrier, geometric-bound rows, convergence-radius witness, radius classifier transport, endpoint packages, and checked semantic certificate rows. Consumers may cite this public boundary without treating Cauchy–Hadamard exactness or standard complex-analysis bridge rows as checked consequences.*

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct public export`
↪ variant `BEDC.Derived.ConvergenceRadiusUp.ConvergenceRadiusCheckedRowReduct public export`
↪ variant `BEDC.Derived.ConvergenceRadiusUp.ConvRad public export`

Proof. Cite the semantic naming certificate of Definition 66.41. It repacks the checked carrier, classifier, transport, witness, and append-unary closure rows into the NameCert schema for $\mathsf{ConvRad}^{\uparrow}$.

Read the same packet through the checked-row reduct of Definition 66.46. This identifies the rows available to consumers: the power-series carrier, geometric-bound data, convergence-radius witness, radius classifier transport, endpoint packages, and semantic certificate rows.

Finally apply Definition 66.47. The public export therefore does not treat Cauchy–Hadamard exactness or standard complex-analysis bridge rows as checked consequences. □

Theorem 66.54 (ConvergenceRadius public boundary exactness)**.** *The public* $\mathsf{ConvRad}^{\uparrow}$ *boundary consists exactly of the power-series carrier, geometric-bound rows, convergence-radius witness, radius classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows. Every consumer read of the public export factors through one of these rows.*

Conversely, each listed row is visible through Definition 66.53. The boundary does not imply real-number supremum completeness, Cauchy–Hadamard exactness, or standard complex-analysis bridge rows.

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad public boundary exactness`
↪ variant `BEDC.Derived.ConvergenceRadiusUp.ConvRadSourceSpec public boundary exactness`

Proof. Start from Definition 66.53. Its public export names the available carrier, geometric-bound, witness, classifier-transport, endpoint-package, and semantic-certificate rows.

Refine this list through Definition 66.41 and Definition 66.46. The semantic certificate supplies the naming packet, while the checked-row reduct gives the exact consumer boundary through which every public read must factor.

For the converse, read each displayed component through the same public export. Each listed row is one of the rows repacked by the certificate and retained by the checked-row reduct.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the boundary also does not provide real-number supremum completeness or the standard complex-analysis bridge rows. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct public boundary exactness`

Theorem 66.55 (ConvergenceRadius geometric-bound common tail window)**.** *For two accepted* $\mathsf{ConvRad}^{\uparrow}$ *public packets over the same power-series carrier, suppose their geometric-bound rows classify to the same radius witness and their endpoint packages agree under the displayed radius classifier. Then there is a single finite tail window through which both public consumer reads factor. The common window exposes only the power-series carrier, geometric-bound rows, convergence-radius witness, endpoint package, checked-row reduct,* hsame*,* Cont*, and* Pkg *provenance.*

The common window still exports no real supremum, Cauchy–Hadamard exactness, or standard complex-analysis bridge row.

Proof. Apply Definition 66.53 to the two accepted packets. Each public consumer read is therefore presented through the power-series carrier, geometric-bound data, convergence-radius witness, radius-classifier transport, endpoint package, checked-row reduct, and semantic certificate rows.

Next use Definition 66.54 for both packets. The boundary exactness theorem forces each read to factor through the displayed rows and does not leave a separate analytic coordinate outside that finite surface.

The hypotheses identify the shared radius-classifier row and the endpoint package rows. Refine the two directed tail ledgers to the finite common tail where those rows are both displayed, and transport the two reads through the same hsame, Cont, and Pkg provenance.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the shared tail window cannot export a real supremum row or a standard complex-analysis bridge row. □

Theorem 66.56 (ConvergenceRadius checked-row readback public factorization)**.** *Every* ConvRadSourceSpec *consumer read factors first through Definition 66.46 and only then reaches the public export rows of Definition 66.53 and Definition 66.54. The factorization exports no Cauchy–Hadamard exactness row and no real supremum completeness principle.*

Proof. Unpack the ConvRadSourceSpec row. Its source, geometric-bound, and radius-witness data are the checked rows retained by Definition 66.46.

Apply the checked-row reduct readback of Definition 66.50. This gives the factor through which every source-specification consumer read is routed before any public export row is inspected.

Now use Definition 66.53 and Definition 66.54. They expose the same public carrier, geometric-bound, convergence-radius witness, classifier-transport, endpoint-package, checked-reduct, and semantic certificate rows, and they expose no additional analytic coordinate.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the public readback factorization also cannot export real supremum completeness. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadSourceSpec checkedRowReduct readback`

Theorem 66.57 (ConvergenceRadius bridged boundary)**.** *Definition 66.53 and Definition 66.54 determine the bridge-consumable* $\mathsf{ConvRad}^{\uparrow}$ *surface: the power-series carrier, geometric-bound rows, convergence-radius witness, radius classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows.*

Real-number supremum completeness, Cauchy–Hadamard exactness, and standard complex-analysis bridge rows are not consequences of this boundary.

Proof. First compose the public export with the public boundary exactness theorem. The export gives the consumer-facing packet, and exactness says that every consumer read factors through the displayed power-series carrier, geometric-bound, witness, classifier-transport, endpoint, checked-reduct, and semantic-certificate rows.

Next read the same packet through Definition 66.46. This keeps precisely the checked rows available to the bridge-facing consumer and adds no hidden supremum, Cauchy–Hadamard, or standard complex-analysis row.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the bridged boundary cannot export real-number supremum completeness or any standard complex-analysis bridge row as a consequence. □

Theorem 66.58 (ConvergenceRadius coefficient-tail public boundary)**.** *Definition 66.31 factors through Definition 66.53 and Definition 66.54: shifting a certified coefficient stream to its tail*

preserves the displayed ConvRad *witness inside the same public carrier, geometric-bound, witness, classifier-transport, endpoint, checked-reduct, and semantic-certificate rows.*

The tail-shift boundary adds no real-number supremum completeness, Cauchy–Hadamard exactness, or standard complex-analysis bridge row.

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad tail shift public boundary`
Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRad tail shift checked row reduct semantic boundary`

Proof. First apply Definition 66.31. The checked tail-closure theorem preserves the convergence-radius witness after replacing the certified coefficient stream by its tail, with the displayed radius and geometric-bound witness read through the same certificate body.

Next use Definition 66.53 and Definition 66.54. Public export gives the consumer-facing packet, and exactness restricts every consumer read to the named carrier, geometric-bound, witness, classifier-transport, endpoint, checked-reduct, and semantic-certificate rows.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the coefficient-tail public boundary exports no real supremum, Cauchy–Hadamard, or standard complex-analysis bridge consequence. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct coefficient tail public boundary`
Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct tail shift public boundary`

Theorem 66.59 (ConvergenceRadius checked-row two-sided ledger boundary)**.** *For any checked-row reduct of a* $\mathsf{ConvRad}^{\uparrow}$ *source and any unary history* q*, the public checked rows determine a two-sided coefficient ledger: the families* $n \mapsto \mathsf{append}(a_n, q)$ *and* $n \mapsto \mathsf{append}(q, a_n)$ *carry* ConvRad *witnesses at the same radius. This ledger remains inside the carrier, geometric-bound, convergence-radius witness, endpoint, checked-reduct, and semantic certificate rows.*

The ledger adds no Cauchy–Hadamard exactness, real supremum row, or standard complex-analysis bridge row.

Proof. Read the source through Definition 66.46. The reduct exposes the checked convergence-radius row, so the same radius is available to both coefficient-family transports.

Apply Definition 66.31 to the unary history q for the family $n \mapsto \mathsf{append}(a_n, q)$. Apply Definition 66.32 to the same unary history for the family $n \mapsto \mathsf{append}(q, a_n)$. These two applications produce the two sides of the public ledger at the original radius.

Finally apply Definition 66.47. Since the checked reduct does not force Cauchy–Hadamard exactness, the two-sided ledger does not introduce a real supremum row or a standard complex-analysis bridge row. □

Lean4 ✓ `BEDC.Derived.ConvergenceRadiusUp.ConvRadCheckedRowReduct append prepend ledger`

Theorem 66.60 (ConvergenceRadius mature package boundary)**.** *Definition 66.53, Definition 66.54, and Definition 66.57 exhaust the mature paper package for the* $\mathsf{ConvRad}^{\uparrow}$ *power-series carrier, geometric-bound rows, convergence-radius witness, radius-classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows. Every consumer of the package factors through one of those rows.*

The package exports no real-number supremum completeness, Cauchy–Hadamard exactness, or standard complex-analysis bridge row.

Proof. First use Definition 66.53 to identify the consumer-facing packet. Its visible rows are the power-series carrier, geometric-bound data, convergence-radius witness, radius-classifier transport, endpoint packages, and semantic certificate rows.

Next apply Definition 66.54. This turns the public export into an exact factorization statement: every public consumer read passes through the displayed rows, and each displayed row is available through the same export.

Now compose with Definition 66.57. The bridge-consumable surface is therefore the same finite row package, with no extra analytic coordinate added by the bridge-facing presentation.

Finally apply Definition 66.47. Since the checked rows do not force Cauchy–Hadamard exactness, the mature paper package cannot export real-number supremum completeness or any standard complex-analysis bridge row. □

Theorem 66.61 (ConvergenceRadius seed status surface)**.** *The* $\mathsf{ConvRad}^{\uparrow}$ *seed carrier consists of the formal power-series carrier, finite geometric-majorant rows, unary convergence-radius witness, classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows. This seed surface is the displayed finite packet consumed by the rotation frontier; it does not change the mature theory grade or add real-number supremum completeness, Cauchy–Hadamard exactness, or a standard complex-analysis bridge row.*

Proof. Project the displayed carrier and public export rows. They provide the formal power-series carrier, geometric-majorant rows, convergence-radius witness, classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows.

Apply Definition 66.60. Every consumer of the mature package factors through those rows, so the same inventory is available as the seed surface for frontier scheduling.

The exclusions are already part of the package boundary. Since the seed-status theorem only names the existing finite rows and does not add an analytic coordinate, it leaves the current closurestatus block unchanged. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvRad}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvRad}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 66.53 binds the consumer-facing convergence-radius boundary over the power-series carrier, geometric-bound rows, convergence-radius witness, classifier transport, endpoint packages, checked-row reduct, and semantic certificate rows.

Constructive story: A $\mathsf{ConvRad}^{\uparrow}$ seed packet is generated from the displayed formal power-series carrier, finite geometric-majorant rows, a unary convergence-radius witness, classifier transport, stability rows, and the checked semantic certificate; consumers may read only those rows and their recorded package provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvRad}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConvergenceRadiusUp.ConvRad\_semanticNameCert`

Bridge status: none

Not claimed: This chapter does not close real-number supremum completeness or standard complex-analysis bridge rows.

Upgrade path: No further paper-side theory grade remains for the finite power-series radius surface; bridgeChecked verification remains on the separate formal axis.

The $\mathsf{CplxDiff}^{\uparrow}$ hub routes the complex-differentiability certificate through chapter 39, chapter 65, chapter 68, and the downstream Rouche route of chapter 5104; the sibling files below hold the body of that route.

Chapter 67

A Concrete Naming Certificate for $\mathsf{CplxDiff}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the complex-differentiability interface $\mathsf{CplxDiff}^{\uparrow}$ at the paper-proof level. The construction sits above the complex-limit certificate $\mathsf{CplxLim}^{\uparrow}$ (chapter 65) and below the holomorphic certificate $\mathsf{Holomorphic}^{\uparrow}$ (chapter 68), and licenses pointwise complex differentiability with an explicit modulus in Bishop's constructive style. Every claim of the form "f is complex-differentiable at z with derivative f'" is unfolded into an explicit unary modulus $\delta : \mathsf{Unary} \to \mathsf{Unary}$ controlling the difference quotient. The Cauchy–Riemann equations are derived inside this chapter as a coupling of real-component partial derivatives, with both directions of the equivalence given as constructive proofs. No use of classical case analysis on complex equality is permitted; everywhere the data is a witness or an explicit refutation.

67.1 The complex difference quotient

Definition 67.1 (Complex non-zero witness)**.** *For a complex history h, the predicate* $\mathsf{CplxNonZero}(h)$ *holds when h is apart from the empty BEDC history:* $\mathsf{hsame}(h, \mathsf{e})$ *implies contradiction. In analytic uses this apartness is supplied by an explicit modulus witness showing that h is bounded away from zero.*

Lean4 (definition) `BEDC.Derived.ComplexDiffUp.CplxNonZero`

Theorem 67.2 (Complex non-zero empty-or-apart split)**.** *Every BEDC history is either history-same to the empty endpoint or supplies the constructor-level apartness witness used by* $\mathsf{CplxNonZero}$*:*

$$\mathsf{hsame}(h, \mathsf{e}) \;\vee\; \mathsf{CplxNonZero}(h).$$

Proof. Split on the three constructors of h. The empty branch is reflexive. The two visible branches use constructor separation from e, yielding the non-zero contradiction witness. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxNonZero empty or nonzero`

Definition 67.3 (Complex difference quotient)**.** *For complex histories z, h with* $\mathsf{CplxNonZero}(h)$ *and a function-history f on the complex carrier, the complex difference-quotient relation* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds when the BEDC quotient ledger records the unary carriers of f, z, and q, the non-zero witness on h, and* $\mathsf{Cont}(f, h, q)$*. Analytically, q represents the history obtained by:*

1. *evaluating f at the complex continuation $z + h$ to obtain* val^{+}*;*

2. evaluating f at z to obtain val*;*

3. forming the complex difference $\mathsf{val}^+ - \mathsf{val}$*; and*

4. dividing by h using the explicit reciprocal of h supplied by $\mathsf{CplxNonZero}(h)$*.*

The resulting q is the unique complex history (up to hsame*) representing $(f(z+h) - f(z))/h$.*

Lean4 (definition) `BEDC.Derived.ComplexDiffUp.CplxDiffQuot`

Theorem 67.4 (Complex difference quotient step unary)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds, then the step history h, the quotient history q, and the quotient continuation ledger are exposed:*

$$\mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Cont}(f, h, q).$$

Proof. Unpack the concrete quotient witness. It already contains the unary quotient carrier and the ledger $\mathsf{Cont}(f, h, q)$. Since the result q is unary, the right-factor inversion for unary continuations recovers the unary carrier of the step history h. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot step unary`

Lemma 67.5 (Complex difference quotient result is nonempty from a nonzero step)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds and h carries the displayed* CplxNonZero *witness, then q cannot be history-same to* e*.*

Proof. Unpack the quotient witness to expose $\mathsf{Cont}(f, h, q)$ and the nonzero witness on h. If $q$hsamee, then the continuation equality identifies $\mathsf{append}(f, h)$ with e. Empty-result inversion for append gives $h$hsamee, contradicting $\mathsf{CplxNonZero}(h)$. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot result nonempty from step`

Theorem 67.6 (Complex difference quotient visible-step construction)**.** *If f, z, out_0, out_1 are unary histories and the quotient ledgers* $\mathsf{Cont}(f, \mathsf{E0}(p), out_0)$ *and* $\mathsf{Cont}(f, \mathsf{E1}(q), out_1)$ *are present, then both visible-step quotient records hold:*

$$\mathsf{CplxDiffQuot}(f, z, \mathsf{E0}(p), out_0) \wedge \mathsf{CplxDiffQuot}(f, z, \mathsf{E1}(q), out_1).$$

Proof. Build each quotient record from the shared unary function and point carriers, the corresponding unary output carrier, the supplied continuation ledger, and constructor separation of the visible step from e. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot visible step construct`

Theorem 67.7 (Complex visible-step quotient results are nonempty)**.** *If f, z, out_0, out_1 are unary histories and the visible step ledgers* $\mathsf{Cont}(f, \mathsf{E0}(p), out_0)$ *and* $\mathsf{Cont}(f, \mathsf{E1}(q), out_1)$ *are present, then neither visible-step quotient result is* hsame*-equal to* e*.*

Proof. Use the visible-step construction to build both complex difference quotient records. Each record has a constructor-separated nonzero step and a continuation ledger, so the quotient result nonempty theorem applies to out_0 and to out_1. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot visible step results nonempty`

Lemma 67.8 (Complex difference quotient is well-defined)**.** *For each (f, z, h) with* $\mathsf{CplxNonZero}(h)$ *there is exactly one complex history q, up to* hsame*, with* $\mathsf{CplxDiffQuot}(f, z, h, q)$*.*

Proof. Functionality of complex addition, subtraction, and division on the rational-pair representation of complex histories (chapter 39) gives a unique q up to hsame. The reciprocal step is governed by the explicit witness in $\mathsf{CplxNonZero}(h)$, which fixes a precision below which the modulus is bounded away from zero. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot quotient cont deterministic`

Theorem 67.9 (Empty-function difference quotient has non-zero result)**.** *If* $\mathsf{CplxDiffQuot}(\mathsf{e}, z, h, q)$ *holds, then* q *is non-zero as a complex history.*

Proof. The quotient ledger contains the non-zero witness for h and the continuation $\mathsf{Cont}(\mathsf{e}, h, q)$. Left-unit uniqueness for continuation gives $q\mathsf{hsame}h$, so any classifier of q with e transports back to a classifier of h with e, contradicting the stored witness. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot empty function result nonzero`

Theorem 67.10 (Shared-result difference quotients determine the step)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *and* $\mathsf{CplxDiffQuot}(f, z, h', q)$ *hold with the same function history and the same result history, then* $h\mathsf{hsame}h'$*, and both quotient ledgers expose* $\mathsf{Cont}(f, h, q)$ *and* $\mathsf{Cont}(f, h', q)$*.*

Proof. Unpack the two quotient witnesses. Their ledger fields are continuations with the same source f and result q; left cancellation for continuation therefore identifies the two step histories. The two ledger fields are retained as the displayed continuation witnesses. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot same result step deterministic`

Theorem 67.11 (Visible complex quotient steps cannot share a result)**.** *If*

$$\mathsf{CplxDiffQuot}(f, z, \mathsf{E0}(p), out) \qquad \textit{and} \qquad \mathsf{CplxDiffQuot}(f, z, \mathsf{E1}(q), out)$$

hold with the same function, point, and result histories, then contradiction follows.

Proof. Shared-result step determinacy identifies the two step histories. The resulting $\mathsf{E0}(p)\mathsf{hsame}\mathsf{E1}(q)$ contradicts constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot visible step same result absurd`

Lemma 67.12 (Difference quotient result agrees with any matching continuation)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds and the same source and step also continue to* q'*, then* $\mathsf{hsame}(q, q')$*.*

Proof. Eliminate the concrete difference-quotient package to expose its continuation ledger $\mathsf{Cont}(f, h, q)$. Determinism of BEDC continuation against the supplied ledger $\mathsf{Cont}(f, h, q')$ gives $\mathsf{hsame}(q, q')$. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot cont result hsame`

Lemma 67.13 (Complex difference quotient point transport determinacy)**.** *If* $\mathsf{hsame}(z, z')$ *and the quotient relation holds both at* z *and at* z' *for the same function and non-zero step,*

$$\mathsf{CplxDiffQuot}(f, z, h, q) \qquad \textit{and} \qquad \mathsf{CplxDiffQuot}(f, z', h, q'),$$

then both quotient ledgers are exposed and their results agree:

$$\mathsf{Cont}(f, h, q) \wedge \mathsf{Cont}(f, h, q') \wedge \mathsf{hsame}(q, q').$$

Proof. Transport the right quotient witness back along $\mathsf{hsame}(z, z')$, keeping the function, step, and quotient endpoints fixed. The same-point quotient determinacy theorem then gives the two continuation ledgers and the endpoint comparison. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot point hsame cont deterministic`

Lemma 67.14 (Compatibility of the difference quotient with hsame)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *and* $\mathsf{hsame}(f, f')$, $\mathsf{hsame}(z, z')$, $\mathsf{hsame}(h, h')$, *and* $\mathsf{hsame}(q, q')$, *then* $\mathsf{CplxDiffQuot}(f', z', h', q')$. *The transported target also exposes the unary point carrier, the transported non-zero witness on* h', *and the quotient ledger* $\mathsf{Cont}(f', h', q')$.

Proof. Transport the unary function, point, and quotient carriers along hsame. The non-zero witness transports by contradiction against $\mathsf{hsame}(h, h')$, and the quotient ledger transports by Cont-stability on source, step, and result. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot hsame transport`
Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot append classifier transport`

Theorem 67.15 (Complex difference quotient result is not empty)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$, *then* $\mathsf{hsame}(q, \mathsf{e})$ *implies contradiction.*

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot result not empty`

Proof. Unpack the quotient ledger to obtain the non-zero witness on h and the continuation $\mathsf{Cont}(f, h, q)$. If $q\mathsf{hsamee}$, transport the continuation result to $\mathsf{Cont}(f, h, \mathsf{e})$; empty-result inversion for continuation then identifies h with e, contradicting $\mathsf{CplxNonZero}(h)$. □

Theorem 67.16 (Complex difference quotient append target is not empty)**.** *If* $\mathsf{CplxDiffQuotAppendClassifier}(f, z, h,$ *then* $\mathsf{hsame}(\mathsf{append}(z, h), \mathsf{e})$ *implies contradiction.*

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuotAppendClassifier append target not empty`

Proof. Unpack the append classifier to obtain the non-zero witness on h. The append-empty decomposition of $\mathsf{hsame}(\mathsf{append}(z, h), \mathsf{e})$ gives $h\mathsf{hsamee}$, which contradicts that witness. □

Lemma 67.17 (Difference quotient results are nonempty)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds, then* q *cannot be internally equal to the empty history.*

Proof. The quotient relation contains the non-zero witness on h and the ledger $\mathsf{Cont}(f, h, q)$. Transporting the ledger along a hypothetical $q\mathsf{hsamee}$ would give $\mathsf{Cont}(f, h, \mathsf{e})$. Empty-result inversion for continuation then forces $h\mathsf{hsamee}$, contradicting the non-zero witness. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot result nonempty`

Theorem 67.18 (Complex difference quotient result visible-context readback)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds and the displayed quotient result is equal inside a common left and right append context to the same context around core, then* $q\mathsf{hsame}core$. *The readback core is also not* hsame*-equal to* e.

Proof. Reassociate both visible displays into a shared append-context form and cancel the common context. The cancellation gives $q\mathsf{hsame}core$. A hypothetical $core\mathsf{hsamee}$ transports to $q\mathsf{hsamee}$, contradicting the checked non-emptiness theorem for complex difference quotient results. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot result visible context readback`

Theorem 67.19 (Complex difference quotient step visible-context readback)**.** *If* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds and the displayed step is equal inside a common left and right append context to the same context around core, then* $h\mathsf{hsame}core$*. The readback core is also non-zero:* $\mathsf{CplxNonZero}(core)$*.*

Proof. Reassociate both visible displays into a shared append-context form and cancel the common context. The cancellation gives $h\mathsf{hsame}core$. A hypothetical $core\mathsf{hsamee}$ transports to $h\mathsf{hsamee}$, contradicting the non-zero step witness stored in the complex difference quotient. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffQuot step visible context readback`

67.2 Pointwise complex differentiability

Definition 67.20 (Pointwise complex differentiability)**.** *For a complex-valued function-history* f*, a complex history* z*, and a candidate derivative* f'*, the pointwise complex differentiability predicate* $\mathsf{CplxDiffAt}(f, z, f')$ *records unary carriers for* f *and* z*, a complex-history carrier for* f'*, at least one difference-quotient witness of Definition 67.3, and the requirement that every such quotient classifies to* f' *under* hsame*.*

Lean4 (definition) `BEDC.Derived.ComplexDiffUp.CplxDiffAt`

Lemma 67.21 (Differentiability witness exposes a nonzero step)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then some displayed quotient witness has a nonzero step* h*, a quotient result* q*, the quotient relation* $\mathsf{CplxDiffQuot}(f, z, h, q)$*, the continuation ledger* $\mathsf{Cont}(f, h, q)$*, and* $q\mathsf{hsame}f'$*.*

Proof. Unpack the pointwise differentiability record. Its witness field supplies h, q, and the concrete quotient package. Unpacking that package exposes the stored nonzero proof for h and the continuation ledger. The classifier field of $\mathsf{CplxDiffAt}$ identifies the displayed quotient result with the derivative candidate. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt witness step nonzero`

Theorem 67.22 (Differentiability witness exposes nonzero unary step)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then there are histories* h *and* q *such that* h *is nonzero and unary,* q *is unary, the quotient relation* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds, the continuation ledger* $\mathsf{Cont}(f, h, q)$ *is displayed, and* $q\mathsf{hsame}f'$*.*

Proof. Extract the displayed nonzero quotient witness from pointwise differentiability. The quotient package contains the continuation ledger and quotient carrier, and the quotient step-readback theorem recovers unary histories for both the step and the quotient result. The classifier field of $\mathsf{CplxDiffAt}$ identifies the displayed quotient result with the derivative candidate. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt witness nonzero unary step`

Theorem 67.23 (Differentiability witness has a nonzero quotient result)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then there are histories* h *and* q *such that* $\mathsf{CplxDiffQuot}(f, z, h, q)$ *holds, the continuation ledger* $\mathsf{Cont}(f, h, q)$ *is displayed, both* h *and* q *are nonzero, and* $q\mathsf{hsame}f'$*.*

Proof. Extract the displayed quotient witness from pointwise differentiability. The quotient package supplies the nonzero witness for h and the continuation ledger. Quotient result non-emptiness turns that same package into the nonzero witness for q, while the classifier field identifies q with the displayed derivative candidate. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt witness nonzero result`

Theorem 67.24 (Empty-function differentiability has non-zero derivative)**.** *If* $\mathsf{CplxDiffAt}(\mathsf{e}, z, f')$ *holds, then* f' *is non-zero as a complex history.*

Proof. Extract a displayed quotient witness h, q from pointwise differentiability. The empty-function quotient theorem gives non-zeroness of q, and the classifier field of CplxDiffAt identifies q with the derivative candidate f'. Transporting a hypothetical $f'\mathsf{hsame}\mathsf{e}$ back along that classifier contradicts the non-zero witness for q. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt empty function derivative nonzero`

Theorem 67.25 (Pointwise differentiability rejects visible step branches)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then two quotient witnesses at the visible steps* $\mathsf{E0}(p)$ *and* $\mathsf{E1}(q)$ *cannot both be present. More precisely,* $\mathsf{CplxDiffQuot}(f, z, \mathsf{E0}(p), out_0)$ *and* $\mathsf{CplxDiffQuot}(f, z, \mathsf{E1}(q), out_1)$ *together imply contradiction.*

Proof. The classifier field of CplxDiffAt identifies both quotient results with the same derivative candidate. Transport the E1-step quotient along this result equality so that both quotients have the same visible result. The checked visible-step same-result separation theorem for complex difference quotients then contradicts constructor separation between E0 and E1. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt visible step branches absurd`

Theorem 67.26 (Pointwise derivative visible-context readback)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds and the displayed derivative candidate is equal inside a common left and right append context to the same context around core, then* $f'\mathsf{hsame}core$*. The readback core is also a complex history carrier.*

Proof. Reassociate both visible displays into a shared append-context form and cancel the common context. The cancellation gives $f'\mathsf{hsame}core$. The CplxDiffAt record stores the complex-history carrier for f', and carrier transport along the recovered hsame gives the carrier for *core*. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt derivative visible context readback`

Theorem 67.27 (Pointwise derivative visible-context nonempty readback)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds and a common left and right append context identifies the displayed derivative candidate with the same context around core, then* $f'\mathsf{hsame}core$*, core is a complex-history carrier, and core is not history-same to* e*.*

Proof. Use visible-context cancellation to recover $f'\mathsf{hsame}core$ and transport the complex-history carrier to *core*. A displayed nonzero quotient result q classifies to f' under the differentiability record; composing $q\mathsf{hsame}f'$, $f'\mathsf{hsame}core$, and a hypothetical $core\mathsf{hsame}\mathsf{e}$ contradicts the quotient-result nonemptiness witness. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt derivative visible context nonempty readback`

Theorem 67.28 (Quotient-result context reads back the derivative)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$*,* $\mathsf{CplxDiffQuot}(f, z, h, q)$*, and a common visible context identifies the quotient result* q *with a core history core, then* $f'\mathsf{hsame}core$*. The core is a complex-history carrier and is not* hsame*-equal to* e*.*

Proof. Apply quotient-result visible-context readback to the displayed quotient. This gives $q\mathsf{hsame}core$ and proves that *core* is nonempty. The classifier field in $\mathsf{CplxDiffAt}$ identifies the same quotient result q with the displayed derivative f'; symmetry and transitivity yield $f'\mathsf{hsame}core$. The complex-history carrier stored for f' transports along that equality to the recovered core. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt quotient result visible context derivative readback`

Lemma 67.29 (Uniqueness of the derivative)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *and* $\mathsf{CplxDiffAt}(f, z, g')$*, then* $\mathsf{hsame}(f', g')$*.*

Proof. Choose a displayed quotient witness q from $\mathsf{CplxDiffAt}(f, z, f')$. The classifier field of the first differentiability witness gives $q\mathsf{hsame}f'$, while the classifier field of the second gives $q\mathsf{hsame}g'$. Symmetry and transitivity of hsame identify f' and g'; the same quotient package also exposes the continuation ledger $\mathsf{Cont}(f, h, q)$. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffAt derivative unique`

Lemma 67.30 (Compatibility with hsame)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$*,* $\mathsf{hsame}(z, z')$*, and* $\mathsf{hsame}(f', g')$*, then* $\mathsf{CplxDiffAt}(f, z', g')$ *with the same modulus* δ*.*

Proof. Transport via Definition 67.14 and the complex-modulus comparison's stability under hsame. □

Lean4 ✓ `BEDC.Derived.ComplexDiffUp.CplxDiffAt hsame transport witness`

Lemma 67.31 (Full hsame-transport of pointwise differentiability)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$*,* $\mathsf{hsame}(f, g)$*,* $\mathsf{hsame}(z, w)$*, and* $\mathsf{hsame}(f', g')$*, then* $\mathsf{CplxDiffAt}(g, w, g')$*. The transported statement also exposes a difference-quotient witness* h, q*, its continuation ledger* $\mathsf{Cont}(g, h, q)$*, and* $q\mathsf{hsame}g'$*.*

Proof. Unpack $\mathsf{CplxDiffAt}(f, z, f')$ into the unary carriers, complex derivative carrier, displayed quotient witness, and classifier. Transport the function carrier, point carrier, derivative carrier, and displayed quotient along the three hsame identifications. For any quotient at the target slots, transport it back to the source slots, apply the source classifier, and compose with $f'\mathsf{hsame}g'$. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt full hsame transport witness`

67.3 Closure laws for complex differentiability

Lemma 67.32 (Sum rule)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *with modulus* δ_f *and* $\mathsf{CplxDiffAt}(g, z, g')$ *with modulus* δ_g*, then* $\mathsf{CplxDiffAt}(f + g, z, f' + g')$ *with modulus* $k \mapsto \max(\delta_f(k + 1), \delta_g(k + 1))$*.*

Proof. The difference quotient of $f + g$ decomposes into the sum of the two individual difference quotients. The pasted modulus halves each error budget at precision $k+1$, giving total error below 2^{-k}. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt sum append derivative carrier`

Lemma 67.33 (Scalar rule)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *with modulus* δ_f *and* c *is a complex history, then* $\mathsf{CplxDiffAt}(c \cdot f, z, c \cdot f')$ *with modulus* $k \mapsto \delta_f(k + |c|)$*, where* $|c|$ *is an explicit unary upper bound on the modulus of* c*.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt scalar append derivative carrier`

Proof. Linear scaling of the difference quotient absorbs c; the modulus pastes by adding the explicit bound on $|c|$ at each precision. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt prepend complex scalar closed`

Lemma 67.34 (Product rule with Leibniz modulus pasting)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *and* $\mathsf{CplxDiffAt}(g, z, g')$*, then* $\mathsf{CplxDiffAt}(f \cdot g, z, f' \cdot g(z) + f(z) \cdot g')$ *with modulus pasted by an explicit Leibniz-style combination of* δ_f*,* δ_g*, an upper bound* $|f|_z$ *for the modulus of* f *near* z*, and an upper bound* $|g|_z$ *for the modulus of* g *near* z*.*

Proof. Write the difference quotient of $f \cdot g$ as

$$\frac{(f(z+h) - f(z)) \cdot g(z+h) + f(z) \cdot (g(z+h) - g(z))}{h}.$$

Each summand contributes one of the two derivative terms up to an explicit error controlled by the local moduli of f and g near z; the local-modulus bounds are obtained from $\mathsf{CplxDiffAt}(f, z, f')$ and $\mathsf{CplxDiffAt}(g, z, g')$ by integrating the difference-quotient bound on a small explicit disk around z. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt product append derivative carrier`

Lemma 67.35 (Chain rule with explicit composite modulus)**.** *If* $\mathsf{CplxDiffAt}(g, z, g')$ *with modulus* δ_g *and* $\mathsf{CplxDiffAt}(f, g(z), f')$ *with modulus* δ_f*, then* $\mathsf{CplxDiffAt}(f \circ g, z, f' \cdot g')$ *with modulus* $k \mapsto \delta_g(\delta_f(k+1) + |g'|_z + 1) + 1$*, where* $|g'|_z$ *is an explicit bound on the modulus of* g'*.*

Proof. For h small enough (controlled by the inner δ_g), $g(z+h) - g(z)$ is close to $g' \cdot h$; plugging this into the outer difference quotient via δ_f yields the composite estimate. The $+1$ corrections absorb the rounding from each linear-approximation step. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt chain append derivative carrier`

67.4 Real-component partial derivatives

Definition 67.36 (Real-coordinate projections of a complex function)**.** *For a complex-valued function-history* f*, write* $f = u + i \cdot v$ *where the real part* u *and the imaginary part* v *are explicit real-valued function-histories obtained by projecting the rational-pair representation of* f*. Each of* u *and* v *takes a complex input* (x, y) *and returns a real history.*

Lean4 (definition) `BEDC.Derived.ComplexDifferentiabilityUp.CplxRealProj`

Definition 67.37 (One-variable real partial derivative)**.** *For a real-valued function-history* u *on the complex carrier and a complex history* $z = (x, y)$*, the partial derivative with respect to the real coordinate,* $\mathsf{PartialReal}(u, z, p)$*, is the data of an explicit modulus* $\delta : \mathsf{Unary} \to \mathsf{Unary}$ *such that for every unary precision* k *and every real history* t *with* $0 < |t| < 2^{-\delta(k)}$*, the difference* $(u(x+t, y) - u(x, y))/t$ *is within* 2^{-k} *of* p *as a real-modulus comparison.*

Lean4 (definition) `BEDC.Derived.ComplexDifferentiabilityUp.PartialReal`

Definition 67.38 (One-variable imaginary partial derivative)**.** $\mathsf{PartialImag}(u, z, q)$ *is defined analogously with respect to the imaginary coordinate: the difference* $(u(x, y+s) - u(x, y))/s$ *approaches* q *under the modulus* δ*.*

Lean4 (definition) `BEDC.Derived.ComplexDifferentiabilityUp.PartialImag`

Lemma 67.39 (Uniqueness of real-coordinate partials)**.** *The real and imaginary partial derivatives of Definition 67.37 and Definition 67.38 are unique up to* hsame*, and respect* hsame *on the input position* z*.*

Proof. The argument is the one-variable specialization of Definition 67.29: triangle-inequality combination of the two moduli halves the precision budget. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.PartialDerivative hsame input unique`

67.5 Cauchy–Riemann equations, constructive form

Theorem 67.40 (Cauchy–Riemann, forward direction)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds with* $f = u + i \cdot v$ *and* $f' = a + i \cdot b$*, then there exist explicit moduli witnessing all four real-coordinate partial derivatives* $\partial_x u$*,* $\partial_x v$*,* $\partial_y u$*,* $\partial_y v$ *at* z*, and the two coupling equations* $\mathsf{hsame}(\partial_x u, \partial_y v)$ *and* $\mathsf{hsame}(\partial_x v, -\partial_y u)$ *hold up to* hsame *on real histories.*

Sketch. Specializing $\mathsf{CplxDiffAt}(f, z, f')$ to a real h (i.e., $h = (t, 0)$) yields PartialReal for both u and v at z, with witnesses $\partial_x u = a$ and $\partial_x v = b$. Specializing to a purely imaginary $h = (0, s)$ and dividing by i yields PartialImag for both u and v, with witnesses $\partial_y u = -b$ and $\partial_y v = a$. The coupling $\mathsf{hsame}(\partial_x u, \partial_y v)$ follows directly from $a = a$, and $\mathsf{hsame}(\partial_x v, -\partial_y u)$ from $b = -(-b)$. Both are uniqueness-up-to-hsame reductions via Definition 67.39. □

Theorem 67.41 (Cauchy–Riemann, backward direction)**.** *Suppose* $f = u + i \cdot v$ *admits the four real-coordinate partial derivatives at* z *with explicit moduli, and the two coupling equations* $\mathsf{hsame}(\partial_x u, \partial_y v)$ *and* $\mathsf{hsame}(\partial_x v, -\partial_y u)$ *hold. If in addition the partials are continuous at* z *with an explicit local modulus, then* $\mathsf{CplxDiffAt}(f, z, \partial_x u + i \cdot \partial_x v)$ *holds, with an explicit composite modulus.*

Sketch. For complex $h = (t, s)$, the difference $f(z + h) - f(z)$ decomposes via the mean-value form (one-variable) along the path $(x, y) \to (x + t, y) \to (x + t, y + s)$. Each segment contributes a partial-derivative term up to an explicit error controlled by the continuity modulus. The two coupling equations are exactly what is needed to reassemble the result into the complex-linear form $(\partial_x u + i \cdot \partial_x v) \cdot h$, giving the complex difference quotient with an explicit error bound. □

Lemma 67.42 (Cauchy–Riemann is an iff under continuity)**.** *Under the hypothesis that* f *has continuous real-coordinate partial derivatives at* z*,* $\mathsf{CplxDiffAt}(f, z, f')$ *is equivalent to the conjunction of the two coupling equations of Definition 67.40.*

Proof. Combine Definition 67.40 and Definition 67.41. □

67.6 The certificate

Definition 67.43 (Complex-differentiability source specification)**.** *The source specification carries a complex function-history* f*, a complex history* z*, and a candidate derivative* f'*, constrained by* $\mathsf{CplxDiffAt}(f, z, f')$ *of Definition 67.20. The generator is the explicit modulus* δ*.*

Lean4 (definition) BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffSourceSpec
Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffSourceSpec of diff

Definition 67.44 (Complex-differentiability pattern specification)**.** *The pattern of* $\mathsf{CplxDiff}^{\uparrow}$ *is the explicit difference-quotient witness of Definition 67.3: for an input* (f, z, h) *with* $\mathsf{CplxNonZero}(h)$*, the pattern is the unique* q *with* $\mathsf{CplxDiffQuot}(f, z, h, q)$*, parametrically in the precision* k*.*

Lean4 (definition) BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffPatternSpec
Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffPatternSpec witness readback

Definition 67.45 (Complex-differentiability classifier specification)**.** *The classifier on the differentiability carrier is* hsame *lifted to the triple* (f, z, f')*: two triples classify together iff each component is* hsame*-equal as a history (with the function-history part compared on its rational-pair representation).*

Lean4 (definition) BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffClassifierSpec
Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffClassifierSpec hsame transport witness

Definition 67.46 (Complex-differentiability stability certificate)**.** *The stability certificate consists of:*

1. *closure of* CplxDiffAt *under* hsame *on each input slot (Definition 67.30);*
2. *uniqueness of the derivative up to* hsame *(Definition 67.29);*
3. *compatibility of the difference-quotient pattern with* hsame *(Definition 67.14).*

Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.CplxDiff stability certificate fields

Definition 67.47 (Complex-differentiability ledger policy)**.** *The ledger policy records:*

1. *sum, scalar, product, and chain closure (Definition 67.32–Definition 67.35);*
2. *the Cauchy–Riemann coupling (Definition 67.40, Definition 67.41, Definition 67.42);*
3. *explicit modulus pasting laws so that any finite combination of differentiable pieces yields an explicit composite modulus.*

Lean4 (definition) BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffLedgerPolicy
Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffLedgerPolicy of diff

Theorem 67.48 (Concrete $\mathsf{CplxDiff}^{\uparrow}$ naming certificate)**.** *The data* (CplxDiffAt, CplxDiffQuot, hsame, CplxDiffStabili *of Definition 67.43–Definition 67.47 forms a naming certificate over* Hist *for the complex-differentiability interface* $\mathsf{CplxDiff}^{\uparrow}$*. The certificate is constructive (every modulus is an explicit unary function), unique-up-to-*hsame *on the derivative slot, and closed under sum, scalar, product, and chain rule.*

Lean4 ✓ BEDC.Derived.ComplexDifferentiabilityUp.complex diff name certificate

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). Constructivity is explicit in the modulus data of Definition 67.20. Uniqueness is Definition 67.29. The four closure laws are Definition 67.32–Definition 67.35. The Cauchy–Riemann coupling sits inside the ledger policy as a derived fact, not as an additional axiom. The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Theorem 67.49 (Semantic CplxDiff$^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 67.48 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9): the carrier is a* Hist*-predicate on triples* (f, z, f')*, the classifier is a restriction of* hsame*, and every closure law is given by an explicit constructive proof. The certificate uses no* Classical.choice*, no* Quot.sound*, and no* propext*, so the certificate's Lean targets fall under the project's axiom-purity gate.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.complex diff semantic name certificate`

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter, together with transport along hsame on each input slot. □

Theorem 67.50 (ComplexDiffUp carrier obligation surface)**.** *The carrier/source obligation surface for* CplxDiff$^{\uparrow}$ *is finite. It consists of the* CplxDiffAt *source row of Definition 67.20, the unary explicit-modulus rows of Definition 67.21–Definition 67.23, the* BHist *endpoints used by the quotient and derivative witnesses, and the* hsame *transport rows of Definition 67.30 and Definition 67.31.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt source obligation surface`

Proof. Project the source specification of Definition 67.43. Its carrier is exactly the CplxDiffAt predicate on triples (f, z, f'), and the witness data is the explicit unary modulus introduced in Definition 67.20.

The nonzero-step and nonempty-result rows are the local carrier rows cited in the statement. They keep every source and result endpoint inside BHist and expose only unary modulus histories to downstream consumers.

Finally, apply Definition 67.30 and Definition 67.31. These rows show that the same carrier is stable under hsame on the function, point, and derivative slots, so the displayed list is a complete carrier/source obligation surface. □

Theorem 67.51 (ComplexDiffUp quotient-pattern obligation surface)**.** *The quotient-pattern obligation surface for* CplxDiff$^{\uparrow}$ *is finite. It consists of the* CplxDiffQuot *pattern of Definition 67.3, the nonzero step witness rows of Definition 67.4 and Definition 67.5, quotient-result determinacy from Definition 67.10, and visible-step exclusion from Definition 67.11.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffPatternSpec obligation surface`

Proof. Start from Definition 67.44. It names CplxDiffQuot as the pattern row and restricts the step input by CplxNonZero(h), so the obligation surface begins with the displayed quotient relation and its nonzero-step witness.

The unary-step and result-nonempty rows are supplied by Definition 67.4 and Definition 67.5. They guarantee that the visible quotient calculation has a carried unary step and a carried quotient result rather than an empty result endpoint.

Determinacy and exclusion are then local. Apply Definition 67.10 for quotient-result determinacy at a fixed visible step, and apply Definition 67.11 to rule out the visible-step collision that would identify a forbidden quotient branch. These four row groups exhaust the quotient-pattern obligations. □

Theorem 67.52 (ComplexDiffUp transport and ledger obligation surface)**.** *The transport and ledger obligation surface for* CplxDiff$^{\uparrow}$ *is finite. It consists of* hsame *transport for* CplxDiffAt *and* CplxDiffQuot*, sum, scalar, product, and chain closure, and the Cauchy–Riemann ledger rows recorded by Definition 67.31, Definition 67.14, Definition 67.32–Definition 67.35, and Definition 67.40–Definition 67.42.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffTransportLedger obligation surface`

Proof. First collect the transport rows. The differentiability predicate transports by Definition 67.31, and the quotient pattern transports by Definition 67.14. Both rows use the same hsame classifier that appears in Definition 67.45.

Next collect the closure ledger. The sum, scalar, product, and chain rows are exactly Definition 67.32–Definition 67.35, with the explicit modulus pasting promised by Definition 67.47.

Finally collect the real-coordinate ledger. The forward, backward, and equivalence rows of Definition 67.40–Definition 67.42 give the Cauchy–Riemann coupling inside the same ledger policy. Together these finite rows are precisely the transport and ledger obligations left by the semantic certificate. □

Theorem 67.53 (ComplexDiffUp finite obligation inventory complete)**.** *The finite obligation inventory for* CplxDiff$^{\uparrow}$ *is exactly the carrier/source surface, quotient-pattern surface, transport and ledger surface, uniqueness and* hsame *transport rows, closure-law rows, and Cauchy–Riemann coupling rows displayed in this chapter. Thus the inventory is the union of Definition 67.50, Definition 67.51, Definition 67.52, Definition 67.29, Definition 67.31, Definition 67.32–Definition 67.35, and Definition 67.42. It contains no host complex equality, no quotient witness, and no non-constructive case analysis.*

Proof. First collect the three displayed surfaces. The carrier/source surface is Definition 67.50, the quotient-pattern surface is Definition 67.51, and the transport and ledger surface is Definition 67.52. These three blocks already name the finite BHist rows, explicit modulus rows, visible quotient rows, and ledger rows used by downstream consumers.

The remaining rows are the rows that identify and move those surfaces. Uniqueness of the derivative slot is Definition 67.29, and full transport through the public classifier is Definition 67.31. These statements keep the inventory inside hsame transport rather than host equality of complex functions or derivatives.

The closure-law rows are exactly Definition 67.32–Definition 67.35. The Cauchy–Riemann coupling is recorded by Definition 67.42 and the forward and backward rows cited by Definition 67.52. Each row is a named constructive ledger entry.

No further case split is available to the public surface. The quotient pattern is exposed only through the visible-step rows of the quotient surface, and the semantic certificate exports CplxDiffAt, CplxDiffQuot, hsame transport, closure, and ledger rows. Hence host complex equality, quotient witnesses, and non-constructive case analysis are outside the finite obligation inventory. □

Theorem 67.54 (ComplexDiffUp seed boundary)**.** *The seed boundary for* CplxDiff$^{\uparrow}$ *consists exactly of the finite* BHist *rows for* CplxDiffAt *and* CplxDiffQuot*, the explicit unary moduli,* hsame *transport, uniqueness, closure laws, and the displayed semantic* NameCert *theorem. No host complex equality, quotient witness, or non-constructive case split is part of the public surface.*

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffAt seed boundary surface`
Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.ComplexDiffSeed boundary`
Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.ComplexDifferentiabilitySeed boundary`
Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.CplxDiffSeedBoundary surface`

Proof. Collect the carrier and quotient rows from Definition 67.50 and Definition 67.51. These rows expose CplxDiffAt, CplxDiffQuot, visible quotient steps, and the explicit unary modulus data as finite BHist entries.

Next collect the transport rows from Definition 67.31, Definition 67.14, and Definition 67.52. They keep the public surface inside hsame transport and finite ledger rows.

Finally use Definition 67.53. It identifies the displayed semantic NameCert surface with exactly these carrier, quotient, transport, uniqueness, closure, and ledger rows, while excluding host complex equality, quotient witnesses, and non-constructive case analysis from the public boundary. □

Lean4 ✓ `BEDC.Derived.ComplexDifferentiabilityUp.ComplexDifferentiabilitySeedBoundary quotient derivative readback`

Theorem 67.55 (ComplexDiffUp seed closure registration)**.** *The seed registration surface for* CplxDiff$^{\uparrow}$ *is exactly the displayed* CplxDiffAt *and* CplxDiffQuot *rows, the explicit unary moduli, componentwise* hsame *transport, the semantic* NameCert *row, and the non-escape exclusions of Definition 67.54. No additional host complex equality, quotient witness, hidden step selector, or non-constructive case split is registered at seed level.*

Closure status for NameCert$_{\text{CplxDiff}^{\uparrow}}$**.**

Constructive story: A differentiability claim starts with a complex function history, a point history, a proposed derivative, and an explicit unary control over all nonzero steps; the carrier accepts only triples whose CplxDiffAt record displays those rows, a nonzero-step difference quotient, and a quotient result classified with the derivative. The classifier is not host complex equality, but componentwise hsame on the function, point, and derivative slots, with quotient witnesses and derivative carriers transported through the same displayed comparisons. The source specification covers exactly the differentiability triples, the pattern specification is the unique difference-quotient row CplxDiffQuot(f, z, h, q) for each accepted nonzero step, and the ledger keeps the unary modulus, quotient continuation, derivative readback, sum, scalar, product, chain, and Cauchy–Riemann rows visible. The five name-certificate obligations are supplied by an admitted differentiability record for carrier inhabitation, by the equivalence laws of hsame on the three slots, and by the transport theorems that rebuild CplxDiffAt and its quotient pattern after classified replacement. A source history h lies in the signature gap of a CplxDiff$^{\uparrow}$ token p when the source-domain accepts its differentiability packet, the signature relation pairs the packet with an answer derivative and evidence ledger, and the token-introduction relation names that derivative as p; unfolded, every displayed nonzero-step quotient must classify to the derivative encoded by p. The chapter registers this explicit difference-quotient construction under the name CplxDiff$^{\uparrow}$; it introduces no classical split on complex equality, no quotient witness, no host complex-analysis equality, and no standard bridge.

Theory closure: TheoryGate(NameCert$_{\text{CplxDiff}^{\uparrow}}$, publicClosed)

Scope closed: The scoped boundary binds CplxDiffAt, CplxDiffQuot, unary moduli, hsame, Cont, the quotient ledger, and downstream boundary rows.

Formal verification: FormalStatus(NameCert$_{\text{CplxDiff}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ComplexDifferentiabilityUp.complex\_diff\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close full complex analysis, holomorphic calculus, a standard bridge, or RealUp completeness.

Upgrade path: Bridged closure requires a bridge theorem from the public differentiability certificate to the standard complex-differentiability boundary.

The ComplexDifferentiability$^{\uparrow}$ packet is consumed by Holomorphic$^{\uparrow}$ through the open-disk differentiability predicate of chapter 68. A holomorphic read is available only after the local difference-quotient row, the displayed limit row, the Complex$^{\uparrow}$ carrier row, and the derivative row have been exposed as finite NameCert data; before those rows are visible, complex differentiability remains a local calculation surface rather than a holomorphic input. The same finite differentiability surface reaches RoucheTheorem$^{\uparrow}$ only through the holomorphic and contour route of chapter 5104. The ComplexDifferentiability$^{\uparrow}$ packet exposes its horizontal lattice route through chapter 39, chapter 65, and chapter 68: the Complex$^{\uparrow}$ carrier row supplies the displayed difference-quotient inputs, the CplxLim$^{\uparrow}$ row supplies the finite limit readback, and the Holomorphic$^{\uparrow}$ row consumes the disk-modulus presentation only after those differentiability rows are visible.

Chapter 68

A Naming Seed for $\mathsf{Holomorphic}^{\uparrow}$

This chapter records a naming seed for holomorphic functions on open disks in $\mathsf{Complex}^{\uparrow}$. The ten fields of the seed are: traditional target, the ordinary holomorphic-on-open-disk interface; BEDC source, $\mathsf{CplxDiff}^{\uparrow}$ together with $\mathsf{ConvRad}^{\uparrow}$ and the open-disk predicate; carrier choice, triples (f, z_0, r) equipped with a disk-local differentiability predicate; classifier choice, hsame-equivalence of the restricted pointwise functions and disk data; stability obligations, transport of the carrier, classifier, moduli, and derivative data along hsame; ledger, the retained function history, disk witness, modulus witness, and analytic-bounds witnesses; core theorem list, the obligations in Definition 68.34; exactness predicate, soundness and coverage against the intended holomorphic-on-disk behavior; standard bridge, comparison with ordinary complex analysis through constructive modulus data; status, Seed.

The source rows are consumed from chapter 39 and chapter 66: $\mathsf{Complex}^{\uparrow}$ supplies the point and function carrier, while $\mathsf{ConvRad}^{\uparrow}$ supplies the disk-radius ledger.

68.1 Open-disk carrier

Definition 68.1 (Open disk predicate)**.** *For complex history z_0, unary radius r, complex history z, and unary gap g, the open-disk predicate* $\mathsf{OpenDisk}(z_0, r, z, g)$ *holds when the rational representation of the modulus $|z - z_0|$ is strictly less than r under* $\mathsf{NatUnaryStrictPrefix}$*. We write* $\mathsf{OpenDisk}(z_0, r, z)$ *when the gap witness is existentially understood. The BEDC kernel anchor records the four histories as unary and records the boundary ledger by* $\mathsf{Cont}(z, g, r)$*.*

Lean4 (definition) `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk`
Lean4 (definition) `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskGap`
Lean4 (definition) `BEDC.Derived.HolomorphicUp.OpenDiskGap`
Lean4 (definition) `BEDC.Derived.HolomorphicUp.OpenDisk`

Lemma 68.2 (Open disk carriers are unary)**.** *For every (z_0, r, z) with* $\mathsf{OpenDisk}(z_0, r, z)$*, the gap modulus* $\mathsf{Gap}(z_0, r, z)$ *is a unary history giving an explicit positive distance from z to the boundary of the disk.*

Proof. The strict-prefix witness immediately gives the gap as the difference history. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk unary components`

Lemma 68.3 (Open disk gap transport certificate)**.** *If* $\mathsf{HolomorphicOpenDiskGap}(z_0, r, z, g)$ *holds and* $z_0 \mathsf{hsame} z_0'$*,* $r \mathsf{hsame} r'$*,* $z \mathsf{hsame} z'$*, and* $g \mathsf{hsame} g'$*, then* $\mathsf{HolomorphicOpenDiskGap}(z_0', r', z', g')$ *holds,* g' *is unary, and* $\mathsf{Cont}(z', g', r')$ *holds.*

Proof. Transport the four unary carriers along the four hsame witnesses.

The boundary continuation ledger transports along the point, gap, and radius hsame witnesses, giving $\mathsf{Cont}(z', g', r')$. Repack these transported fields as the target open-disk gap certificate, retaining the unary witness for g'. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskGap transport certificate`

Lemma 68.4 (Open disk unary suffix transport)**.** *If an open-disk gap is represented by unary histories z_0, r, z, g with continuation $\mathsf{Cont}(z, g, r)$, then any unary suffix q preserving the point and radius endpoints keeps the same gap as a boundary witness: from $\mathsf{Cont}(z, q, z_q)$ and $\mathsf{Cont}(r, q, r_q)$ one obtains $\mathsf{Cont}(z_q, g, r_q)$.*

Proof. Unary closure gives unary histories for z_q and r_q. Associativity of Cont, together with unary commutativity of the suffix and gap histories, identifies the shifted boundary continuation. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskGap unary suffix transport`

Theorem 68.5 (Open-disk gap zero-headed component impossibility)**.** *For histories z_0, r, z, g, an open-disk gap witness has no zero-headed center, radius, point, or gap component:*

$$\mathsf{HolomorphicOpenDiskGap}(z_0, r, z, g) \Longrightarrow (\exists w,\ z_0 = \mathsf{E0}(w)) \vee (\exists w,\ r = \mathsf{E0}(w)) \vee (\exists w,\ z = \mathsf{E0}(w)) \vee (\exists w,\ g = \mathsf{E0}(w))$$

Proof. Unpack the open-disk gap witness into its four unary-history fields. Rewriting any component as a zero-headed history turns the corresponding unary witness into a unary zero-extension, contradicting unary zero-extension separation. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskGap zero headed component absurd`

Theorem 68.6 (Open disk zero-headed component impossibility)**.** *For histories $z_0, r, z : \mathsf{BHist}$, an existential open-disk witness $\mathsf{OpenDisk}(z_0, r, z)$ has no zero-headed center, radius, or point component.*

Proof. Unpack the existential open disk into the complex center carrier, unary radius, complex point carrier, and its boundary gap witness. A zero-headed radius contradicts unary zero-extension separation. A zero-headed center or point forms a reflexive complex-history classifier with a zero-headed endpoint, contradicting complex component endpoint separation. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk zero headed component absurd`

Lemma 68.7 (Open-disk transport along hsame)**.** *If $\mathsf{OpenDisk}(z_0, r, z, g)$ and each of z_0, r, z, g is replaced by an hsame-equivalent history, then the transported tuple is again an open-disk witness. The transported witness exposes the unary carriers of the four target histories and the transported ledger $\mathsf{Cont}(z', g', r')$.*

Proof. Transport the four unary fields by unary-history stability under hsame, and transport the boundary ledger by Cont-stability under hsame on source, gap, and result. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk hsame transport`

Theorem 68.8 (Open-disk gap semantic naming certificate)**.** *Given a concrete open-disk witness $\mathsf{OpenDisk}(z_0, r, z, g)$, the gap fiber $g' \mapsto \mathsf{OpenDisk}(z_0, r, z, g')$ carries a semantic naming certificate whose source, pattern, and ledger predicates are this same fiber and whose classifier is* hsame.

Proof. The witness g inhabits the source fiber. Reflexivity, symmetry, and transitivity are the corresponding laws for hsame. Carrier transport follows from open-disk transport along hsame with z_0, r, and z fixed by reflexivity and the gap slot transported by the classifier witness. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk gap semanticNameCert`

Lemma 68.9 (Radius ledger enlargement)**.** *If* $\mathsf{OpenDisk}(z_0, r, z, g)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then* $\mathsf{OpenDisk}(z_0, r', z, g \cdot e)$*, where $g \cdot e$ denotes BEDC history append.*

Proof. The radius carrier follows from unary closure of $\mathsf{Cont}(r, e, r')$. The enlarged gap is unary by append closure, and the ledger $\mathsf{Cont}(z, g \cdot e, r')$ is obtained by composing $\mathsf{Cont}(z, g, r)$ with $\mathsf{Cont}(r, e, r')$ using associativity of append. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk radius extension closed`

Theorem 68.10 (Open disk radius extension gap determinacy)**.** *If* $\mathsf{HolomorphicOpenDisk}(z_0, r, z, g)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then every displayed open-disk gap for radius r' agrees with the appended gap $g \cdot e$.*

Proof. The radius enlargement lemma constructs the displayed disk with gap $g \cdot e$. Comparing its point-to-radius continuation with any other displayed gap by left cancellation yields the classifier $\mathsf{hsame}(g \cdot e, g')$. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk radius extension gap deterministic`

Theorem 68.11 (Holomorphic open disk radius extension rejects zero-headed enlarged gap)**.** *If* $\mathsf{HolomorphicOpenDisk}(z_0, r, z, g)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then the enlarged gap $g \cdot e$ cannot be displayed as a zero-headed history:*

$$g \cdot e = \mathsf{E0}(w) \quad \Longrightarrow \quad \bot.$$

Proof. The radius enlargement lemma constructs $\mathsf{HolomorphicOpenDisk}(z_0, r', z, g \cdot e)$. Its gap field gives $\mathsf{UnaryHistory}(g \cdot e)$. Transport that unary witness along the displayed equality $g \cdot e = \mathsf{E0}(w)$, obtaining a unary zero-headed history, which contradicts BEDC unary zero-headed separation. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk radius extension gap zero headed absurd`

Theorem 68.12 (Open disk radius extension gap determinacy)**.** *If* $\mathsf{OpenDisk}(z_0, r, z)$ *holds, g, e, and g' are unary,* $\mathsf{Cont}(g, z, r)$*,* $\mathsf{Cont}(r, e, r')$*, and* $\mathsf{Cont}(g', z, r')$*, then*

$$\mathsf{hsame}(g \cdot e, g').$$

Proof. The open-disk carrier supplies the unary point history z. Compose $\mathsf{Cont}(g, z, r)$ with $\mathsf{Cont}(r, e, r')$ and commute the unary histories z and e inside append to obtain $\mathsf{Cont}(g \cdot e, z, r')$. Left cancellation against the displayed ledger $\mathsf{Cont}(g', z, r')$ yields $\mathsf{hsame}(g \cdot e, g')$. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk radius extension gap deterministic`

Theorem 68.13 (Holomorphic open disk gap-point commutation)**.** *If* $\mathsf{HolomorphicOpenDisk}(z_0, r, z, g)$*, then the radius is classified by the gap-point append:*

$$\mathsf{hsame}(r, g \cdot z).$$

Proof. The open-disk ledger contains $\mathsf{Cont}(z, g, r)$, hence $r\mathsf{hsame}z \cdot g$. The point and gap are unary histories, so unary append commutativity classifies $z \cdot g$ with $g \cdot z$. Transitivity of hsame gives the displayed radius classifier. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk gap point comm`

Theorem 68.14 (Open disk $e1$ gap radius inversion)**.** *For complex histories z_0, z, unary gap history gap, and radius history radius, if the displayed center-to-point continuation is* $\mathsf{Cont}(z_0, \mathsf{e1}(gap), z)$ *and* $\mathsf{NatUnaryStrictPrefix}(\mathsf{e1}(gap), \mathsf{e1}(radius))$*, then* $\mathsf{NatUnaryStrictPrefix}(z_0, z)$ *and* $\mathsf{NatUnaryStrictPrefix}(gap, radiu$

Proof. Use the displayed $e1$ gap as the strict positive tail from z_0 to z. The radius comparison descends by $e1$ strict-prefix inversion. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk e1 gap radius inversion`

Theorem 68.15 (Open disk carrier witnessed components)**.** *If* $\mathsf{HolomorphicOpenDiskCarrier}(z_0, r, z)$*, then z_0, r, and z are unary histories and there is a unary boundary gap g with* $\mathsf{Cont}(z, g, r)$*. The same data repackages as* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r, z)$*.*

Proof. The complex carrier exposes unary histories for the center and point. The rational radius carrier gives a positive unary denominator, hence a unary radius. The carrier's displayed gap and boundary ledger then supply the witnessed open disk. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskCarrier witnessed components`

Theorem 68.16 (Open disk radius continuation gap witness)**.** *If z lies in the unary open disk with center z_0 and radius r, and r continues by a unary history e to r', then the original boundary gap factors through a unary continued gap. Concretely, there are histories g, g' with*

$$\mathsf{UnaryHistory}(g) \wedge \mathsf{UnaryHistory}(g') \wedge \mathsf{Cont}(z, g, r) \wedge \mathsf{Cont}(g, e, g') \wedge \mathsf{Cont}(z, g', r').$$

Proof. Unpack the open-disk witness as a unary gap g with $\mathsf{Cont}(z, g, r)$. Associativity of continuation supplies the middle history g' with $\mathsf{Cont}(g, e, g')$ and $\mathsf{Cont}(z, g', r')$; unary continuation closure makes g' unary. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk radius extension gap witness`

Theorem 68.17 (Open disk radius continuation closure)**.** *Under the same hypotheses, membership in the unary open disk is preserved when the radius is continued from r to r'.*

Proof. Use the continued gap from Definition 68.16. The center and point carriers are unchanged, and the continued radius is unary by continuation closure. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk radius continuation extend`

Theorem 68.18 (Witnessed open disk radius extension gap public readback)**.** *If a witnessed unary open disk at radius r is continued by a unary history to r', then the witnessed open disk at r' is retained. Moreover, every publicly displayed gap at radius r' is classified by a continued original gap.*

Proof. Radius continuation closure supplies the witnessed disk at r'. The radius extension gap witness exposes g and its continued gap g'; comparing the ledger $\mathsf{Cont}(z, g', r')$ with any displayed radius-r' ledger by left cancellation gives the public readback classifier. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed radius extension gap public readback`

Theorem 68.19 (Open disk carrier radius extension witnessed)**.** *If* $\mathsf{HolomorphicOpenDiskCarrier}(z_0, r, z)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r', z)$*. Moreover, the original boundary gap factors through a unary continued gap* g' *with* $\mathsf{Cont}(g, e, g')$ *and* $\mathsf{Cont}(z, g', r')$*.*

Proof. First extract the witnessed open disk from the carrier components. Radius continuation closure gives the witnessed disk at r', while the radius continuation gap theorem exposes the original gap and its continued ledger. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskCarrier radius extension witnessed`

Theorem 68.20 (Witnessed open disk zero-headed component impossibility)**.** *For histories* z_0, r, z*, a witnessed unary open disk has no zero-headed center, radius, or point component:*

$$\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r, z) \Longrightarrow (\exists w,\ z_0 = \mathsf{E0}(w)) \vee (\exists w,\ r = \mathsf{E0}(w)) \vee (\exists w,\ z = \mathsf{E0}(w)) \Longrightarrow \mathsf{False}.$$

Proof. Unpack the witnessed disk into unary center, radius, point, and boundary-gap witnesses. In each displayed zero-headed branch, substituting the equality turns the corresponding unary witness into a unary zero-extension, contradicting unary zero-extension separation. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed zero headed component absurd`

Lemma 68.21 (Witnessed open-disk boundary transport)**.** *If a witnessed unary open disk is transported along* hsame *in its center, radius, and point histories, then the transported open disk remains witnessed, and the displayed boundary gap still supplies* $\mathsf{Cont}(z', g, r')$*.*

Proof. Unpack the witnessed open disk to obtain the unary center, radius, point, and gap histories together with $\mathsf{Cont}(z, g, r)$. Unary-history transport moves the three endpoint carriers along hsame, while continuation transport moves the boundary ledger to $\mathsf{Cont}(z', g, r')$. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed boundary hsame transport`

Theorem 68.22 (Witnessed open disk unary suffix transport)**.** *If* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r, z)$*, q is unary, and the point and radius continue by the same suffix,* $\mathsf{Cont}(z, q, z_q)$ *and* $\mathsf{Cont}(r, q, r_q)$*, then* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r_q, z_q)$*. Moreover, the same gap supplies a boundary continuation* $\mathsf{Cont}(z_q, g, r_q)$ *for some unary history* g*.*

Proof. Unpack the witnessed disk to obtain a unary gap g with $\mathsf{Cont}(z, g, r)$. This is a concrete open-disk gap witness, so gap-level unary suffix transport gives the shifted boundary continuation $\mathsf{Cont}(z_q, g, r_q)$. The shifted gap witness exposes the unary carriers for the shifted point and radius, allowing the witnessed disk to be repacked. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed unary suffix transport`

Theorem 68.23 (Witnessed open disk center-point unary suffix transport)**.** *If* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r$ *q is unary, and the center, point, and radius continue by the same suffix to* z_{0q}*,* z_q*, and* r_q*, then* $\mathsf{HolomorphicOpenDiskWitnessed}(z_{0q}, r_q, z_q)$*. Moreover, some unary gap still supplies* $\mathsf{Cont}(z_q, g, r_q)$*.*

Proof. Unary continuation closure moves the center carrier from z_0 to z_{0q}. The witnessed open-disk unary suffix theorem shifts the point and radius and supplies the retained gap boundary, so the shifted center carrier repackages the witnessed disk at the transported endpoint. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed center point unary suffix transport`

Theorem 68.24 (Witnessed open disk center-point unary suffix gap public readback)**.** *If* HolomorphicOpenDiskWitn *q is unary, and the center, point, and radius continue by the same suffix to z_{0q}, z_q, and r_q, then* $\mathsf{HolomorphicOpenDiskWitnessed}(z_{0q}, r_q, z_q)$*. Moreover, every displayed gap for the shifted open disk reads back to an original unary gap g with* $\mathsf{Cont}(z, g, r)$*,* $\mathsf{Cont}(z_q, g, r_q)$*, and* $g\mathsf{hsame}g'$*.*

Proof. Use center-point unary suffix transport for the shifted witnessed disk. For readback, unpack the original witness to a gap g, rebuild the corresponding open-disk gap, and apply gap unary suffix transport. The transported boundary and any displayed shifted boundary have the same point and radius endpoints, so left cancellation identifies g with the displayed gap. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed center point unary suffix gap public readback`

Theorem 68.25 (Open disk empty radius excludes visible points)**.** *For every complex center z_0, visible point* $\mathsf{e1}(z)$*, and gap history g, the concrete open-disk gap predicate with empty radius leads to contradiction.*

Proof. Unpack the concrete gap predicate to obtain the continuation witness $\mathsf{Cont}(\mathsf{e1}(z), g, \mathsf{e})$. Empty-result inversion for continuations identifies the visible point endpoint with e, contradicting constructor disjointness of $\mathsf{e1}$ and e in BHist. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDiskGap e1 point empty radius absurd`

Theorem 68.26 (Open disk empty radius excludes zero-visible points)**.** *For every complex center z_0, zero-visible point* $\mathsf{e0}(z)$*, and gap history g, the concrete open-disk gap predicate with empty radius leads to contradiction.*

Proof. Unpack the concrete gap predicate to obtain $\mathsf{Cont}(\mathsf{e0}(z), g, \mathsf{e})$. Empty-result inversion for continuations identifies the zero-visible point endpoint with e, which contradicts constructor disjointness of $\mathsf{e0}$ and e in BHist. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDiskGap e0 point empty radius absurd`

Lemma 68.27 (Open disk endpoint transport)**.** *If the center and point are transported by the complex-history classifier, the radius is transported by* hsame*, and the same unary gap continues the point to the radius, then* $\mathsf{OpenDisk}(z_0, r, z)$ *transports to* $\mathsf{OpenDisk}(z_0', r', z')$*.*

Proof. Unfold the open-disk witness. The target center and point carriers are supplied by the two complex classifiers, the radius is moved along hsame, and the explicit continuation witness for the gap is reused at the transported endpoint. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk gap endpoint transport`

Theorem 68.28 (Open disk radius extension closure)**.** *If* $\mathsf{OpenDisk}(z_0, r, z)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then* $\mathsf{OpenDisk}(z_0, r', z)$*.*

Proof. Unpack the open disk to a unary gap g with boundary ledger $\mathsf{Cont}(g, z, r)$. The radius r' is unary by continuation closure. The extended gap is $g \cdot e$, unary by append closure. Since the point history is complex, it is unary, so z commutes with e; associativity then rewrites the composite boundary ledger to $\mathsf{Cont}(g \cdot e, z, r')$. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk radius extension closed`

Theorem 68.29 (Open disk radius extension gap public readback)**.** *If* $\mathsf{OpenDisk}(z_0, r, z)$*, e is unary, and* $\mathsf{Cont}(r, e, r')$*, then* $\mathsf{OpenDisk}(z_0, r', z)$*. Moreover, every displayed unary gap g' with* $\mathsf{Cont}(g', z, r')$ *is the public display of some original gap g with* $\mathsf{Cont}(g, z, r)$*, continued by e:*

$$g \cdot e\mathsf{hsame}g'.$$

Proof. Unpack the original open disk to a unary boundary gap g with $\mathsf{Cont}(g, z, r)$. The radius extension closure constructs the continued gap $g \cdot e$ and its ledger $\mathsf{Cont}(g \cdot e, z, r')$. Comparing this ledger with the displayed ledger $\mathsf{Cont}(g', z, r')$ by right cancellation gives $g \cdot e\mathsf{hsame}g'$. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk radius extension gap public readback`

Theorem 68.30 (Open disk empty-radius boundary)**.** *If* $\mathsf{OpenDisk}(z_0, \mathsf{e}, z)$ *holds, then the displayed point is the empty endpoint. More precisely, the existential boundary witness supplies a unary gap g with* $g\mathsf{hsame}$ *and with the original empty-radius ledger*

$$\mathsf{Cont}(g, z, \mathsf{e}).$$

Proof. Unpack the concrete open-disk witness to obtain a unary gap g and the ledger $\mathsf{Cont}(g, z, \mathsf{e})$. Empty-result inversion for continuation identifies both input histories of this ledger with e, giving $z\mathsf{hsame}$ and $g\mathsf{hsame}$ while retaining the original ledger. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.OpenDisk empty radius boundary`

Theorem 68.31 (Open disk witnessed empty-radius boundary)**.** *If a witnessed unary open disk has radius history r with* $r\mathsf{hsame}$*, then its point and displayed boundary gap are both empty endpoints. Concretely, from* $\mathsf{HolomorphicOpenDiskWitnessed}(z_0, r, z)$ *and* $r\mathsf{hsame}$ *one obtains* $z\mathsf{hsame}$ *and a unary gap g such that*

$$g\mathsf{hsame} \wedge \mathsf{Cont}(z, g, r).$$

Proof. Unpack the witnessed open disk to obtain the unary gap g and the boundary ledger $\mathsf{Cont}(z, g, r)$. Transport this ledger along $r\mathsf{hsame}$, producing $\mathsf{Cont}(z, g, \mathsf{e})$. Empty-result inversion for continuation identifies both the point endpoint and the gap endpoint with e, while the original ledger and unary gap witness remain part of the displayed boundary data. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskWitnessed empty radius boundary`

Theorem 68.32 (Holomorphic open disk empty-radius boundary)**.** *If* $\mathsf{HolomorphicOpenDisk}(z_0, r, z, g)$ *holds and* $r\mathsf{hsame}$*, then both the displayed point and boundary gap are empty endpoints:* $z\mathsf{hsame} \wedge g\mathsf{hsame}$*.*

Proof. The concrete disk predicate contains the boundary ledger $\mathsf{Cont}(z, g, r)$. Transporting its result along $r\mathsf{hsame}$ gives a continuation into the empty history. Empty-result inversion for continuation then identifies both inputs with e. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk empty radius boundary`

68.2 Holomorphic predicate

Definition 68.33 (Holomorphic on disk seed predicate)**.** *For function $f : \mathsf{Complex}^{\uparrow} \to \mathsf{Complex}^{\uparrow}$, center z_0, and unary radius r, the seed predicate* $\mathsf{HoloOnDisk}(f, z_0, r)$ *is the intended carrier condition saying that a uniform modulus function $\delta : \mathsf{Unary} \times \mathsf{Unary} \to \mathsf{Unary}$ controls complex differentiability at every point z with* $\mathsf{OpenDisk}(z_0, r, z)$ *and every unary precision k, using the boundary gap* $\mathsf{Gap}(z_0, r, z)$ *as the second modulus input.*

68.3 Core theorem obligations

Definition 68.34 (Holomorphic core theorem obligations)**.** *The* $\mathsf{Holomorphic}^{\uparrow}$ *seed carries the following proof obligations before it can be presented as a certificate:*

1. *uniform-modulus obligation: any accepted carrier witness must expose explicit modulus data, and equivalent witnesses must admit pointwise refinements compatible with* hsame*;*
2. *iterated-derivative obligation: the derivative sequence* $\mathsf{IteratedCplxDiff}(f, z_0, n, f^{(n)})$ *must be generated by unary recursion from the local complex-differentiability data;*
3. *closure obligations: sum, product, and composition must preserve the carrier, with explicit modulus transport and an explicit image-containment witness for composition;*
4. *power-series obligation: Taylor coefficients computed from the iterated-derivative data must determine a power-series witness with a convergence-radius comparison to r;*
5. *identity obligation: agreement on a quantitatively dense sequence inside the disk must transport to agreement on an explicitly smaller disk;*
6. *Cauchy-bound obligation: local upper bounds for $|f|$ on smaller disks must yield unary bounds for the iterated derivatives at the center.*

Lean4 (definition) `BEDC.Derived.HolomorphicUp.IteratedCplxDiff`
Lean4 (definition) `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff`

Lemma 68.35 (Iterated complex differentiability preserves unary seed histories)**.** *If an iterated complex-differentiability chain starts from a unary seed history and each successor stage is recorded by a unary continuation step, then every selected iterate is unary.*

Proof. Induct on the iteration index. At zero, transport the unary seed along the displayed hsame endpoint. At a successor index, unpack the previous iterate, the unary step, and the continuation witness; the induction hypothesis gives the previous iterate as unary, and unary continuation closure gives the selected successor history. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedCplxDiff unary of seed`

Theorem 68.36 (Iterated complex differentiability transports unary readback)**.** *If the seed endpoint and selected iterate of an iterated complex-differentiability chain are each replaced by* hsame*-equivalent histories, then the whole chain has the corresponding transported endpoint, and unary readback from the transported seed still proves the transported selected iterate unary.*

Proof. Induct on the iteration index. At zero, compose the seed comparison, the displayed endpoint comparison, and the chain's zero endpoint. At a successor stage, transport the predecessor chain by the induction hypothesis and move the successor continuation witness across the endpoint

comparison by continuation hsame-stability. Unary readback then applies to the transported chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedCplxDiff hsame transport unary readback`

Theorem 68.37 (Iterated complex differentiability is closed under additive concatenation)**.** *If an iterated complex-differentiability chain carries a seed to a middle history in n stages, and a second chain carries that middle history to a target history in m stages, then their concatenation carries the seed to the target in $n + m$ stages. Unary readback from the seed still proves the target history unary.*

Proof. Induct on m. At $m = 0$, the second chain is exactly an hsame endpoint, so transport the first chain to the target and reuse unary readback. At a successor stage, apply the induction hypothesis to the predecessor endpoint, append the displayed unary continuation step, and close the target by unary continuation closure. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedCplxDiff add closed unary readback`

Theorem 68.38 (Strict iterated complex differentiability is closed under additive concatenation)**.** *If a strict iterated complex-differentiability chain carries a seed to a middle history in n stages, and a second strict chain carries that middle history to a target history in m stages, then their concatenation carries the seed to the target in $n + m$ strict stages. Unary readback from the seed still proves the target history unary.*

Proof. Induct on m. At $m = 0$, the second chain is an hsame endpoint, so transport the first strict chain to the target and reuse strict unary readback. At a successor stage, apply the induction hypothesis to the predecessor endpoint, append the displayed nonempty unary continuation step, and close the target by unary continuation closure. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff add closed strict readback`

Theorem 68.39 (Strict iterated complex differentiability preserves unary seed histories)**.** *If a strict iterated complex-differentiability chain starts from a unary seed history, then every selected strict iterate is unary.*

Proof. Induct on the iteration index. The zero case transports unary membership along the seed classifier. At a successor index, unpack the predecessor, strict unary step, and continuation witness; the induction hypothesis gives the predecessor as unary, and unary continuation closure gives the successor. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff unary of seed`

Theorem 68.40 (Strict iterated complex differentiability forgets strict steps)**.** *Every strict iterated complex-differentiability chain has an underlying ordinary iterated complex-differentiability chain with the same seed, index, and selected endpoint. Consequently, unary membership of the seed still reads back to unary membership of the selected endpoint through the ordinary chain.*

Proof. Induct on the iteration index. At zero, the strict and ordinary chains have the same seed endpoint classifier. At a successor stage, unpack the predecessor and strict nonempty unary step, apply the induction hypothesis to the predecessor chain, and keep the same unary continuation witness while erasing only the nonemptiness field. Unary readback then applies to the resulting ordinary iterated chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff forget strict unary readback`

Theorem 68.41 (Strict iterated complex differentiability has ordinary readback)**.** *Every strict iterated complex-differentiability chain determines the ordinary iterated complex-differentiability chain with the same seed, index, and selected history. Consequently, a unary seed still reads back unary membership of the selected history:*

$$\mathsf{IteratedStrictCplxDiff}(seed, n, h) \Longrightarrow \mathsf{IteratedCplxDiff}(seed, n, h) \wedge (\mathsf{UnaryHistory}(seed) \Rightarrow \mathsf{UnaryHistory}(h)).$$

Proof. Induct on the iteration index. At zero, both chains are the same endpoint classifier from the seed to the selected history, and unary readback is transport along that classifier. At a successor index, unpack the predecessor, the unary step, the nonempty-step witness, and the continuation. The induction hypothesis supplies the ordinary predecessor chain; discarding only the nonempty-step witness reconstructs the ordinary successor chain, and unary continuation closure gives the displayed readback. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff iteratedCplxDiff unary readback`

Lemma 68.42 (Strict iterated complex differentiability exposes a strict prefix)**.** *If a strict iterated complex-differentiability chain starts from a unary seed history and advances by at least one nonempty unary continuation step, then the seed is a strict unary prefix of the selected iterate.*

Proof. Induct on the number of remaining successor stages. The first successor unpacks a nonempty unary step and transports its source along the seed classifier. At a later successor, the induction hypothesis gives a strict prefix from the seed to the previous iterate, and strict-prefix transitivity composes it with the displayed nonempty continuation tail. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff strict prefix`

Theorem 68.43 (Strict iterated complex differentiability zero-or-prefix dichotomy)**.** *For any strict iterated complex-differentiability chain from a unary seed, the zero-stage case identifies the selected endpoint with the seed, while every positive stage exposes a strict unary prefix from the seed into the selected endpoint:*

$$\mathsf{IteratedStrictCplxDiff}(seed, n, h) \Longrightarrow \mathsf{hsame}(seed, h) \vee \mathsf{NatUnaryStrictPrefix}(seed, h).$$

Proof. Split on the natural iteration index. The zero branch is exactly the endpoint classifier stored by the definition. The successor branch applies the strict-prefix readback theorem to the displayed nonempty chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff zero or strict prefix`

Theorem 68.44 (Strict iterated complex differentiability endpoint contradiction)**.** *If a strict iterated complex-differentiability chain starts from a unary seed and advances by at least one nonempty continuation step, then its final endpoint cannot be* hsame *to the seed:*

$$\mathsf{UnaryHistory}(seed) \Longrightarrow \mathsf{IteratedStrictCplxDiff}(seed, n+1, h) \Longrightarrow \mathsf{hsame}(seed, h) \Longrightarrow \bot.$$

Proof. The strict-prefix readback theorem extracts a strict unary prefix from the seed to the final endpoint. Unpacking this prefix gives a nonempty unary tail and a continuation from the seed to the endpoint. Endpoint hsame cancellation forces that tail to be empty, contradicting strictness. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff endpoint hsame absurd`

Theorem 68.45 (Strict iterated complex differentiability result nonemptiness)**.** *If a strict iterated complex-differentiability chain advances by at least one nonempty continuation step, then its final endpoint cannot be* hsame *to* e*.*

Proof. Invert the successor strict-iteration witness to expose the previous endpoint, the final step, and its nonempty-step field. Transport the final continuation result along the displayed empty endpoint comparison. Empty-result inversion forces the final step to be empty, contradicting the stored strictness field. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff result not empty`

Theorem 68.46 (Strict iterated complex differentiability transports unary readback)**.** *If a strict iterated complex-differentiability chain has* hsame*-equivalent seed and selected endpoints, then the strict chain transports to those endpoints. Consequently, a unary transported seed still reads back unary membership of the transported selected endpoint.*

Proof. Induct over the iteration index. At zero, compose the seed comparison with the displayed zero endpoint and the target comparison. At a successor index, transport the predecessor chain by the induction hypothesis and transport the final continuation step across the target endpoint comparison. Unary readback then applies to the transported strict chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff hsame transport unary readback`

Theorem 68.47 (Strict iterated complex differentiability transports strict prefix readback)**.** *If a nonempty strict iterated complex-differentiability chain has* hsame*-equivalent seed and target endpoints, then the strict chain transports to those endpoints, and a unary transported seed exposes a strict unary prefix into the transported target.*

Proof. Induct over the successor count of the strict chain. The first successor composes the seed endpoint comparison with the zero-stage endpoint and transports the displayed nonempty unary continuation. The successor case applies the induction hypothesis to the previous iterate and transports the final continuation step. The strict-prefix readback theorem then applies to the transported strict chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff hsame transport strict prefix`

Theorem 68.48 (Strict iterated complex differentiability is closed under additive concatenation)**.** *If a strict iterated complex-differentiability chain carries a seed to a middle history in n stages, and a second strict chain carries that middle history to a target history in at least one stage, then their concatenation is a strict chain from the seed to the target in $n+m+1$ stages. Unary readback from the seed exposes a strict unary prefix into the target.*

Proof. Induct over the positive right-hand chain. In the first stage, transport the left chain to the predecessor endpoint of the displayed strict continuation and append that continuation. In the successor case, apply the induction hypothesis to the previous endpoint and append the final strict continuation. The strict-prefix readback theorem applies to the concatenated nonempty chain. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff add closed strict prefix readback`

Theorem 68.49 (Strict iterated complex differentiability with ordinary suffix readback)**.** *If a strict iterated complex-differentiability chain carries a seed to a middle history in at least one stage, and an ordinary iterated chain carries that middle history to a target history, then the combined ordinary chain starts at the seed and has the summed length. Unary readback from the seed still exposes a strict unary prefix into the target.*

Proof. Forget the strict left chain to an ordinary iterated chain and concatenate it with the ordinary suffix. For the prefix readback, induct over the ordinary suffix. The zero case transports the strict prefix through the endpoint comparison. In the successor case, the induction hypothesis gives a strict prefix into the previous endpoint, and the final ordinary continuation gives a unary weak prefix from that endpoint to the target; strict prefix is closed under such a right extension. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff ordinary suffix strict prefix readback`

Theorem 68.50 (Strict iterated concatenation rejects a closed endpoint)**.** *If a strict iterated chain carries a unary seed to a middle history and a second strict chain carries that middle history through at least one nonempty step to a target, then the concatenated target cannot be history-same to the seed.*

Proof. The additive strict-prefix readback theorem concatenates the two chains and extracts a strict unary prefix from the seed to the target. Unpacking that prefix gives a nonempty unary tail and a continuation from the seed to the target. Endpoint hsame cancellation forces that tail to be empty, contradicting strictness. □

Lean4 ✓ `BEDC.Derived.HolomorphicUp.IteratedStrictCplxDiff add closed endpoint hsame absurd`

68.4 The seed fields

Definition 68.51 (Holomorphic source seed field)**.** *The source seed field on* (f, z_0, r) *records the intended constraint* HoloOnDisk(f, z_0, r) *together with its open-disk gap data.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicSourceSpec semantic name certificate`

Definition 68.52 (Holomorphic pattern seed field)**.** *The pattern seed field is the uniform-modulus complex-differentiability predicate over the open disk.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicPatternSpec semantic name certificate`

Definition 68.53 (Holomorphic classifier seed field)**.** *The classifier seed field groups two holomorphic-on-disk witnesses when they restrict to* hsame*-equivalent pointwise functions on the disk's interior and carry compatible disk data.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicClassifierSpec semantic name certificate`

Definition 68.54 (Holomorphic stability seed field)**.** *The stability seed field lists the required* hsame*-respect of* HoloOnDisk *on the input function and on the disk parameters, transport of moduli under* hsame*, and compatibility with the iterated-derivative recursion.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicStabilityCert openDisk transport fields`
Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicPatternSpec hsame iterate transport`
Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDisk stability certificate fields`
Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicOpenDiskCarrier stability certificate`
Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicStabilityCert suffix gap comm`
Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicStabilityCert radius extension hsame transport`

Definition 68.55 (Holomorphic ledger seed field)**.** *The ledger seed field records the originating function history, center, radius, disk-gap witnesses, modulus witnesses, closure witnesses, Taylor-data witnesses, Cauchy-bound witnesses, and identity-theorem density witnesses.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicLedgerPolicy extension gap no zero head`

Definition 68.56 ($\mathsf{Holomorphic}^{\uparrow}$ obligation surface)**.** *The bridged status of this chapter is* bridgedClosed*. The fields in this chapter bind the carrier, classifier, stability, ledger, open-disk, and iterated-differentiability data for* $\mathsf{Holomorphic}^{\uparrow}$ *to the paper-side scope witnessed by the sibling certificate files.*

Theorem 68.57 (HolomorphicUp public NameCert export)**.** *The fields of Definition 68.51, Definition 68.52, Definition 68.53, Definition 68.54, Definition 68.55, and Definition 68.34 assemble the public* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ *export over* BHist*,* UnaryHistory*,* Cont*,* hsame*,* Pkg*, and the open-disk gap rows. The exported surface consists exactly of the holomorphic-on-disk source row, uniform complex-differentiability pattern row, disk-restricted classifier row, transport and iterated-derivative stability rows, ledger provenance rows, and the core theorem obligations named by those fields.*

Proof. Project the source, pattern, classifier, stability, and ledger fields from the five seed definitions. Each field is already stated over the displayed BHist, UnaryHistory, Cont, hsame, Pkg, and open-disk gap surface.

Add the core theorem obligations from Definition 68.34. These obligations name the closure and readback rows that the public certificate may expose, without adding Cauchy integral theory, analytic continuation, or a standard bridge.

Repack the projected rows as the public $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ export. The packet is public because every consumer-visible row is named by one of the displayed fields and every transport row remains inside the same BEDC open-disk surface. □

Theorem 68.58 (HolomorphicUp standard bridge boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ *export of Definition 68.57 supplies the standard bridge boundary for the BEDC open-disk row, the uniform differentiability row, the transport rows, and the ledger rows. Its bridge surface is exactly those rows over* BHist*,* UnaryHistory*,* Cont*,* hsame*,* Pkg*, and the open-disk gap data. It does not include Cauchy integral theory, analytic continuation, or bridge verification.*

Lean4 ✓ `BEDC.Derived.HolomorphicUp.HolomorphicUp StdBridge`

Proof. Start with Definition 68.57. It identifies the public rows available to a consumer: the open-disk source, the uniform complex-differentiability pattern, disk-restricted classifier transport, iterated-derivative stability, and ledger provenance.

Restrict the standard bridge boundary to those public rows. The source and pattern clauses give the BEDC open-disk and uniform differentiability entries; the classifier and stability clauses give the transport entries; the ledger policy gives the retained witness entries.

The exclusions are part of the same boundary reading. Cauchy integral theory, analytic continuation, and bridge verification are not exported by any field in the public packet, so they remain outside this bridge boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$**.**

Constructive story: A holomorphic datum is carried as a visible disk package: a function history, a center, a radius, and open-disk gap witnesses are admitted only when the BEDC disk row says the point stays inside the radius and the function satisfies the holomorphic-on-disk source condition. Two such packages classify together when their disk parameters are compatible and their pointwise function values are hsame-equivalent on the interior, so the classifier

follows the displayed disk and function histories rather than host function equality. The source seed covers precisely those (f, z_0, r) packages with open-disk gap data, the pattern seed is uniform-modulus complex differentiability on that disk, and the ledger keeps the originating function, center, radius, gap witnesses, modulus witnesses, closure rows, Taylor-data rows, Cauchy-bound rows, and density witnesses inspectable. The five name-certificate obligations are discharged by the semantic certificate rows for the source, pattern, and classifier together with the open-disk transport lemmas: an admitted disk supplies a carrier witness, hsame gives reflexivity, symmetry, and transitivity, and radius and gap transport keep admitted representatives inside the same surface. A history h lies in the signature gap of a Holomorphic$^{\uparrow}$ token p when the source-domain accepts its disk package, the signature relation pairs that package with an answer function and evidence ledger, and the token-introduction relation names the answer as p; after unfolding, this means the same open disk carries uniform complex-differentiability moduli and compatible pointwise values for the token's function. The chapter registers this open-disk differentiability export and checked standard bridge under the name Holomorphic$^{\uparrow}$; it adds no Cauchy integral theory, analytic continuation, or identity-theorem closure.

Theory closure: TheoryGate(NameCert$_{\text{Holomorphic}\uparrow}$, maturePackageClosed)

Scope closed: The paper-side mature package consists of the public NameCert$_{\text{Holomorphic}\uparrow}$ export of Definition 68.57 together with the standard bridge boundary of Definition 68.58 over BHist, UnaryHistory, Cont, hsame, Pkg, open-disk gap rows, transport rows, and ledger witnesses.

Formal verification: FormalStatus(NameCert$_{\text{Holomorphic}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.HolomorphicUp.HolomorphicSourceSpec\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: The chapter does not claim Cauchy integral theory, identity theorem closure, or analytic continuation.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 69

A Concrete Naming Certificate for $\mathsf{CplxSer}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for convergent infinite complex sums $\sum_{n=0}^{\infty} c_n$, where c is a unary-indexed family of complex histories. The certificate is distinct from $\mathsf{ConvRad}^{\uparrow}$ (chapter 66): there is no formal variable, only a complex sequence and its tail behaviour. The construction sits above $\mathsf{CplxLim}^{\uparrow}$ (chapter 65) and reuses the regular Cauchy machinery underlying $\mathsf{Real}^{\uparrow}$ (chapter 38). Absolute convergence and conditional convergence are licensed by separate predicates with explicit unary moduli; the comparison, ratio, and root tests are recorded as constructive lemmas; and the Cauchy product of two absolutely convergent series is constructed term-wise with explicit modulus pasting. The certificate is fully constructive and uses no Classical.choice, no Quot.sound, and no propext, so its Lean targets fall under the project's axiom-purity gate.

69.1 Term sequences and partial sums

Definition 69.1 (Complex term sequence)**.** *A complex term sequence is a unary-indexed family* $c : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$*-history, i.e. a function from unary histories to complex carrier histories.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexTermSeqCarrier`

Lemma 69.2 (Complex term sequence carrier transport)**.** *If two complex term families* $c, d : \mathsf{BHist} \to \mathsf{BHist}$ *are pointwise related by* hsame *on every unary index, then the* ComplexTermSeqCarrier *predicate transports from* c *to* d*.*

Proof. Fix a unary index n. The carrier for $c(n)$ expands to a complex-pair witness: rational real and imaginary component carriers together with a continuation ledger whose result is $c(n)$. Transport that continuation result along the pointwise comparison $c(n)\mathsf{hsame}d(n)$ while keeping the same rational component witnesses. The transported pair witness is exactly the complex carrier for $d(n)$. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexTermSeqCarrier hsame transport`

Definition 69.3 (Complex partial sum)**.** *For a term sequence* c*, the complex partial-sum predicate* $\mathsf{ComplexPartSum}(c, n, S_n)$ *is the closed inductive predicate generated by:*

1. $\mathsf{ComplexPartSum}(c, \mathsf{e}, \mathsf{e}_{\mathsf{Complex}^{\uparrow}})$ *where* $\mathsf{e}_{\mathsf{Complex}^{\uparrow}}$ *is the zero complex history (both real components empty);*

2. *if* $\mathsf{ComplexPartSum}(c, n, S_n)$ *and the complex addition relation gives* $S_n + c_n = S_{n+1}$*, then* $\mathsf{ComplexPartSum}(c, \mathsf{E1}(n), S_{n+1})$.

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexPartSum`

Lemma 69.4 (Complex partial-sum index cases)**.** *Every closed witness of* $\mathsf{ComplexPartSum}(c, n, S)$ *either has the zero index* $n = \mathsf{e}$ *and identifies* S *with the chosen zero history, or has a successor index* $n = \mathsf{E1}(m)$ *together with a predecessor partial sum and the continuation step from that predecessor to* S*.*

Proof. Invert the closed partial-sum witness. The zero constructor gives the empty index and the base history. The step constructor exposes the predecessor index, predecessor sum, and continuation payload directly. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum index cases`

Lemma 69.5 (Partial sums are complex histories)**.** *For every term sequence* c *and unary* n*, any two histories obtained from the closed partial-sum recursion for* c *at* n *agree up to* hsame*.*

Proof. Induct on the first partial-sum derivation. The zero case is immediate. In the successor case, the second derivation has the same successor shape; the induction hypothesis identifies the previous sums, and determinacy of the BEDC continuation relation identifies the two successor sums. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum deterministic`

Lemma 69.6 (Complex partial-sum visible predecessor decomposition is deterministic)**.** *If two displayed successor decompositions of the same complex partial-sum history have the forms*

$$n = \mathsf{BHist.e1}(m) \wedge \mathsf{ComplexPartSum}(zero, c, m, P) \wedge \mathsf{Cont}(P, c(m), S)$$

and

$$n = \mathsf{BHist.e1}(m') \wedge \mathsf{ComplexPartSum}(zero, c, m', P') \wedge \mathsf{Cont}(P', c(m'), S),$$

then $m = m'$ *and* $P \mathsf{hsame} P'$*.*

Proof. Invert both visible successor equalities. Constructor injectivity identifies the exposed predecessor indices, and closed complex partial-sum determinacy identifies the predecessor results under hsame. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum visible previous decomposition deterministic`

Lemma 69.7 (Complex partial sums exist uniquely)**.** *For every chosen zero history, term family* c*, and unary index* n*, there is a history* S *satisfying* $\mathsf{ComplexPartSum}(c, n, S)$*, and every other closed partial sum at* n *is* hsame*-equal to* S*.*

Proof. Unary induction on the index constructs the zero partial sum at e. At a successor index, the induction hypothesis gives the predecessor sum and the BEDC continuation relation appends the current term. The closed partial-sum determinacy theorem identifies every competing derivation at the same index. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum exists unique`

Lemma 69.8 (Partial sums respect pointwise term equivalence)**.** *If two term families $c, d : \mathsf{Hist} \to \mathsf{Hist}$ are pointwise related by* hsame *on every unary index, and S, T are closed partial sums for c, d at the same index n, then $S \mathsf{hsame} T$.*

Proof. Induct on the first closed partial-sum witness. The base case forces the second witness to be the zero constructor. In the successor case, the second witness has the same successor index; the induction hypothesis identifies the predecessor sums, closedness of the first witness supplies unaryness of the predecessor index, and the pointwise term comparison identifies the terms. The BEDC continuation relation then identifies the successor sums. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum term hsame deterministic`

Lemma 69.9 (Complex partial-sum successor term hsame determinacy)**.** *If two chosen zero histories agree under* hsame*, and two term families agree pointwise under* hsame*, then a successor partial-sum step computed from the first family has the same result as any closed successor partial-sum derivation for the second family at the corresponding $\mathsf{E1}(n)$ index.*

Proof. First prove cross-family determinacy for partial sums by induction on the first closed derivation. The zero constructor uses the zero-history comparison. The successor constructor inverts the matching successor derivation on the other side, applies the induction hypothesis to the predecessor sums, and then uses determinacy of Cont with the pointwise term comparison. Applying this cross-family result to the predecessor sums of the displayed successor witness and the closed successor derivation identifies the final results. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum successor term hsame deterministic`

Lemma 69.10 (Partial sums respect pointwise history sameness)**.** *For two term families $c, d : \mathsf{BHist} \to \mathsf{BHist}$ with $\mathsf{hsame}(c(m), d(m))$ at every unary index m, and for base histories z, z' with $z \mathsf{hsame} z'$, closed partial-sum derivations for c and d at the same unary index have* hsame *results.*

Proof. Induct on the first closed partial-sum derivation. The zero constructor uses the base comparison $z \mathsf{hsame} z'$. In the successor case, invert the second derivation, recover the predecessor unary witness, apply the induction hypothesis to the predecessor sums, and use continuation determinacy with the pointwise comparison of the current terms. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum pointwise hsame deterministic`

Lemma 69.11 (Complex partial-sum result nonempty from nonempty terms)**.** *If the chosen zero history is apart from* e*, and every unary-indexed term $c(m)$ is apart from* e*, then no closed $\mathsf{ComplexPartSum}(zero, c, n, S)$ witness can have $S \mathsf{hsame} \mathsf{e}$.*

Proof. Induct on the closed partial-sum witness. The zero case is the supplied apartness of the chosen zero history. In the successor case, transport the final empty equality across the displayed continuation $\mathsf{Cont}(S, c(m), T)$, invert the empty result, and use the unary index extracted from the predecessor partial-sum chain to contradict the apartness of $c(m)$. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum result nonempty of nonempty terms`

Theorem 69.12 (Complex partial-sum results are unary)**.** *If the chosen zero history is unary and each term history is unary on unary indices, then every closed complex partial-sum witness has unary result.*

Proof. Induct on the closed complex partial-sum witness. The base is the supplied unary zero history. In the successor case, an inner induction on the predecessor partial-sum witness supplies unaryness of the predecessor index, the term hypothesis supplies unaryness of the next term, and unary continuation closure gives the successor result. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum result unary`

69.2 Absolute convergence

Definition 69.13 (Modulus partial sum)**.** *For a complex term sequence c, the modulus partial sum* $\mathsf{AbsPartSum}(c, n, M_n)$ *is the real-history value $M_n = \sum_{k<n} |c_k|$ computed by an explicit unary recursion taking complex modulus pointwise and summing in* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum`

Lemma 69.14 (Absolute partial-sum index cases)**.** *Every closed witness of* $\mathsf{ComplexAbsPartSum}(\mathsf{modulus}, n, M)$ *either has the zero index $n = \mathsf{e}$ and identifies M with the chosen zero history, or has a successor index $n = \mathsf{E1}(m)$ together with a predecessor absolute partial sum and the continuation step from that predecessor to M.*

Proof. Invert the closed absolute partial-sum witness. The zero constructor gives the empty index and base history; the step constructor exposes the predecessor sum, predecessor index, and continuation payload directly. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum index cases`

Lemma 69.15 (Absolute partial-sum successor result is deterministic)**.** *If an absolute partial-sum derivation reaches M at unary index n, and the next modulus continuation sends M and* $\mathsf{modulus}(n)$ *to T, then every closed derivation of the successor absolute partial sum at* $\mathsf{E1}(n)$ *has result* hsame *to T.*

Proof. Induct on the previous absolute partial-sum derivation to obtain determinacy at each index. In the successor case, the other derivation has the same successor shape; the induction hypothesis identifies the previous modulus sums, and determinacy of Cont identifies the successor result. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum successor result deterministic`

Lemma 69.16 (Absolute partial-sum unary index is deterministic)**.** *If n is a unary history, then any two closed derivations of* $\mathsf{ComplexAbsPartSum}(\mathsf{zero}, \mathsf{modulus}, n, -)$ *have* hsame *results.*

Proof. Induct on the first absolute partial-sum derivation. The empty index case forces the second derivation to be the same base constructor. In the successor case, unary inversion exposes the predecessor index, the induction hypothesis identifies the previous modulus sums, and determinacy of Cont identifies the successor results. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum unary index deterministic`

Lemma 69.17 (Absolute partial sums exist uniquely)**.** *For every chosen zero history, modulus family, and unary index n, there is a history M satisfying* $\mathsf{ComplexAbsPartSum}(zero, \mathsf{modulus}, n, M)$*, and every other closed absolute partial sum at n is* hsame*-equal to M.*

Proof. Unary induction on n constructs the empty-index absolute partial sum from the zero constructor. At a successor index, the induction hypothesis supplies a predecessor sum, and the canonical BEDC continuation appends the current modulus history. Unary-index determinacy identifies every competing closed absolute partial sum at the same index. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum exists unique`

Theorem 69.18 (Absolute partial-sum results are unary)**.** *If the chosen zero history is unary and every modulus history* modulus(m) *is unary whenever m is unary, then every closed* ComplexAbsPartSum$(zero,$ modu *witness has unary result M.*

Proof. Induct on the absolute partial-sum witness. The base result is the supplied unary zero history. For a successor step, an inner induction on the predecessor absolute partial-sum witness gives unaryness of the predecessor index; the modulus hypothesis makes the appended modulus unary, and unary continuation closure gives the successor result. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum result unary`

Lemma 69.19 (Absolute partial-sum result is nonempty from nonempty terms)**.** *If the chosen zero history is not history-same to* e*, and every modulus term* modulus(m) *at a unary index is not history-same to* e*, then no closed* ComplexAbsPartSum$(zero,$ modulus$, n, M)$ *witness can have* Mhsamee*.*

Proof. Induct on the absolute partial-sum witness. The base constructor reduces to the nonempty zero assumption. In the successor case, empty-result inversion for the continuation into M forces the current modulus term to be empty. A local induction on the predecessor partial-sum witness supplies unaryness of that predecessor index, contradicting the pointwise nonempty modulus hypothesis. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum result nonempty of nonempty terms`

Theorem 69.20 (Absolute partial-sum empty result boundary)**.** *If a closed* ComplexAbsPartSum$(zero,$ modulus$, n,$ *witness has result* Mhsamee*, then either the chosen zero history is history-same to* e*, or some unary predecessor index m has* modulus(m)hsamee*.*

Proof. Induct on the absolute partial-sum witness. The base constructor reads the empty result directly as $zero$hsamee. In the successor constructor, transport the final continuation result to e and apply continuation empty-result inversion; the right endpoint of that continuation is the current modulus term, while a local index induction supplies unaryness of the predecessor index. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum empty result boundary`

Lemma 69.21 (Absolute partial-sum successor modulus hsame determinacy)**.** *If two chosen zero histories agree under* hsame*, and two modulus families agree pointwise under* hsame*, then a successor absolute partial-sum step computed from the first modulus family has the same result as any closed successor absolute partial-sum derivation for the second family at the corresponding* E1(n) *index.*

Proof. Induct on the first absolute partial-sum derivation to obtain cross-family determinacy of predecessor modulus sums. The zero constructor is handled by the zero-history comparison. In the successor constructor, invert the matching successor derivation, compare predecessor sums by induction, and compare the two continuation results using the pointwise modulus comparison. Applying this determinacy statement to the predecessor of the displayed successor step and the closed successor derivation gives the required result comparison. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum successor modulus hsame deterministic`

Lemma 69.22 (Absolute partial sums respect pointwise history sameness)**.** *For two modulus families $m, m' : \mathsf{BHist} \to \mathsf{BHist}$ with pointwise* hsame *agreement at every unary index, and for base histories z, z' with $z\,\mathsf{hsame}\,z'$, closed absolute partial-sum derivations at the same unary index have* hsame *results.*

Proof. Induct on the first absolute partial-sum derivation. The zero constructor is the base comparison. In the successor case, invert the matching derivation, identify the predecessor sums by the induction hypothesis, and apply continuation determinacy using the pointwise modulus comparison at the predecessor index. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum pointwise hsame deterministic`

Lemma 69.23 (Absolute partial-sum modulus successor result is deterministic)**.** *If two modulus rows are pointwise related by* hsame*, an absolute partial-sum derivation for the first row reaches M at index n, and the next continuation sends M and* $\mathsf{modulus}(n)$ *to T, then every closed successor derivation for the second row has result* hsame *to T.*

Proof. Induct on the first absolute partial-sum derivation while comparing it to a derivation over the pointwise-related modulus row. At each successor, the induction hypothesis identifies the predecessor sums, the pointwise modulus comparison identifies the next term, and BEDC continuation determinacy identifies the resulting successor histories. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum modulus successor result deterministic`

Lemma 69.24 (Absolute partial-sum modulus-hsame successor result)**.** *Let the two absolute partial-sum systems have* hsame *zero histories and pointwise* hsame *modulus histories. If both systems compute partial sums at the same index n, and their successor continuations are formed from the corresponding modulus histories, then the two successor results are* hsame*.*

Proof. Induct on the first absolute partial-sum derivation. The zero case compares the two zero histories and the pointwise modulus histories directly by continuation respect for hsame. In the successor case, the second derivation has successor shape; the induction hypothesis identifies the previous successor partial sums, and the pointwise modulus comparison identifies the final continuation results. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum modulus hsame successor result`

Lemma 69.25 (Absolute partial sums respect pointwise modulus equivalence)**.** *If two modulus functions $m, m' : \mathsf{Hist} \to \mathsf{Hist}$ are pointwise related by* hsame *on every unary index, and M, T are closed absolute partial sums for m, m' at the same index n, then $M\,\mathsf{hsame}\,T$.*

Proof. Induct on the first absolute partial-sum witness. The base case forces the second witness to be the zero constructor. In the successor case, the second witness has the same successor index; the induction hypothesis identifies the predecessor sums, closedness of the first witness supplies unaryness of the predecessor index, and the pointwise modulus comparison identifies the two modulus terms. Determinacy of the BEDC continuation relation identifies the successor sums. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum modulus hsame deterministic`

Lemma 69.26 (Absolute partial sums transport across modulus equivalence)**.** *If two zero histories are related by* hsame *and two modulus functions are pointwise related by* hsame *on unary indices, then every unary-indexed absolute partial-sum witness over the first modulus function transports to a closed absolute partial-sum witness over the second modulus function with* hsame *result.*

Proof. Induct on the absolute partial-sum witness. The empty constructor transports by the zero-history comparison. In the successor constructor, unary inversion exposes the predecessor index, the induction hypothesis transports the predecessor sum, and continuation respect for hsame transports the final modulus continuation. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsPartSum modulus hsame transport`

Definition 69.27 (Absolute convergence)**.** *A complex series c converges absolutely with bound M and modulus $N : \mathsf{Unary} \to \mathsf{Unary}$, written $\mathsf{AbsConv}(c, M, N)$, when:*

1. *for every unary n, $\mathsf{AbsPartSum}(c, n, M_n)$ gives a real-history bound $M_n \leq M$;*
2. *for every unary precision k and unary $n, m \geq N(k)$, $|M_n - M_m| < 2^{-k}$.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexAbsConv`

Lemma 69.28 (Absolute convergence implies convergence)**.** *If $\mathsf{AbsConv}(c, M, N)$ holds, the complex partial sums $(S_n)_n$ form a regular complex Cauchy sequence with the same modulus N, and hence have an explicit complex-history limit S by Cauchy completeness (chapter 65).*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsConv complexSeriesConv`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsConv complexSeriesConv of pointwise hsame`

Sketch. The triangle inequality on each segment of consecutive partial sums bounds $|S_n - S_m|$ by $|M_n - M_m|$. Apply the modulus from absolute convergence directly. □

69.3 Convergence and uniqueness

Definition 69.29 (Complex series convergence)**.** $\mathsf{ComplexSeriesConv}(c, S)$ *holds when the complex partial-sum sequence $(S_n)_n$ is a regular complex Cauchy sequence (in the sense of chapter 65) with explicit limit S.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexSeriesConv`

Theorem 69.30 (Sum is unique)**.** *If* $\mathsf{ComplexSeriesConv}(c, S)$ *and* $\mathsf{ComplexSeriesConv}(c, S')$*, then* hsame *holds on both real components of S and S'.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesConv sum unique shared witness`

Sketch. Direct application of complex-limit uniqueness. □

69.4 Comparison, ratio, and root tests

Theorem 69.31 (Comparison test)**.** *Let c be a complex term sequence and d a real-history term sequence with $d_n \geq 0$ for all n. If for every unary n, the modulus comparison $|c_n| \leq d_n$ holds (decidable on each unary index), and $\sum_n d_n$ converges in $\mathsf{Real}^{\uparrow}$ with explicit modulus N_d, then $\mathsf{AbsConv}(c, M, N_d)$ holds with the same modulus N_d and bound $M = \sum d_n$.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexAbsConv comparison same sequence`

Sketch. The comparison hypothesis transports to absolute partial sums: $M_n \leq \sum_{k<n} d_k$. The modulus from d-convergence then bounds the differences of M_n. □

Theorem 69.32 (Ratio test)**.** *If there exists a rational r with $0 \leq r < 1$ and a unary N_0 such that for all unary $n \geq N_0$, $|c_{n+1}| \leq r \cdot |c_n|$ (decidable comparison), then* AbsConv *holds with bound $M = |c_{N_0}|/(1-r)$ and modulus $N(k) = N_0 + \lceil k/\log_2(1/r) \rceil$ computed as an explicit unary function of the precision.*

Sketch. Geometric majorant: starting from N_0, the tail is bounded by a geometric series with ratio r. Compute the modulus from the geometric tail sum. □

Lemma 69.33 (Root test)**.** *If there exists a rational $r < 1$ and unary N_0 such that $|c_n|^{1/n} \leq r$ (where the root is computed by explicit unary approximation) for all unary $n \geq N_0$, then* AbsConv *holds with explicit geometric modulus.*

Sketch. The hypothesis gives $|c_n| \leq r^n$; apply comparison with the geometric series. □

69.5 Cauchy product

Definition 69.34 (Cauchy product)**.** *For complex term sequences a, b, the Cauchy product $c = a \star b$ is the term sequence $c_n = \sum_{k=0}^{n} a_k \cdot b_{n-k}$ defined by an explicit unary recursion using complex multiplication.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.CauchyProduct`

Theorem 69.35 (Cauchy product preserves absolute convergence)**.** *If* $\mathsf{AbsConv}(a, M_a, N_a)$ *and* $\mathsf{AbsConv}(b, M_b, N_b)$ *hold, then* $\mathsf{AbsConv}(a \star b, M_a \cdot M_b, N_{ab})$ *holds with explicit modulus N_{ab} computed from N_a, N_b, M_a, M_b. Moreover the sum of the Cauchy product equals the product of sums: $\sum(a \star b) = (\sum a) \cdot (\sum b)$.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.CauchyProduct result unary`

Sketch. Reorder partial sums into a triangular block structure. Absolute convergence justifies the reordering with explicit modulus pasting. □

69.6 Closure laws

Lemma 69.36 (Linearity of convergence)**.** *If* $\mathsf{ComplexSeriesConv}(a, S_a)$ *and* $\mathsf{ComplexSeriesConv}(b, S_b)$*, and α, β are complex histories, then* $\mathsf{ComplexSeriesConv}(\alpha a + \beta b, \alpha S_a + \beta S_b)$ *with explicit modulus pasting.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum pointwise append partials`

Sketch. Pointwise scalar multiple and pointwise sum of regular sequences remain regular with combined modulus. □

Lemma 69.37 (Reordering for absolute convergence)**.** *If* $\mathsf{AbsConv}(c, M, N)$ *holds and $\sigma : \mathsf{Unary} \to \mathsf{Unary}$ is an explicit unary bijection (a permutation of the indexing), then the reordered series $c \circ \sigma$ is also absolutely convergent with the same bound and an explicit modulus computable from N and σ.*

Sketch. Absolute convergence ensures that any reordering converges to the same limit with adjusted modulus. □

69.7 The certificate

Definition 69.38 (Complex series source specification)**.** *The source specification on* Complex$^{\uparrow}$*-term-sequences carries the constraint* ComplexSeriesConv(c, S) *for some* S*. Its generator is the complex partial-sum predicate.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesConv pointwise source limit transport`
Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexSeriesSourceSpec`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesSourceSpec limit carrier`

Definition 69.39 (Complex series pattern specification)**.** *The pattern of* CplxSer$^{\uparrow}$ *is regular Cauchy partial-sum convergence: a sequence* c *is admissible when its partial-sum sequence is regular Cauchy with explicit modulus.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexSeriesPatternSpec`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesPatternSpec complexSeriesConv`

Definition 69.40 (Complex series classifier specification)**.** *Two convergent complex series are classified together when their limits agree under* hsame *on both real components.*

Lean4 (definition) `BEDC.Derived.ComplexSeriesUp.ComplexSeriesClassifierSpec`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesClassifierSpec limit transport`

Definition 69.41 (Complex series stability certificate)**.** *The stability certificate consists of:*

1. *closure of* ComplexSeriesConv *under pointwise* hsame*-equivalence on the term sequence;*
2. *transport of moduli under* hsame *on terms;*
3. *compatibility of partial sums with* hsame *on indexing.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum term hsame transport`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesConv limit transport successor step`

Definition 69.42 (Complex series ledger policy)**.** *The ledger policy records:*

1. *linearity of convergence under sum and complex scalar multiplication;*
2. *Cauchy product of absolutely convergent series;*
3. *reordering invariance for absolute convergence.*

Theorem 69.43 (Concrete CplxSer$^{\uparrow}$ naming certificate)**.** *The five fields* (ComplexSeriesSourceSpec, ComplexSeriesPa *form a naming certificate over* Hist *for the complex series interface* CplxSer$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.complex series name certificate`

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). All conversions between term sequence, partial-sum sequence, modulus, and limit are explicit unary functions, so the certificate uses no Classical.choice, no Quot.sound, and no propext — its Lean targets fall under the project's axiom-purity gate. □

Theorem 69.44 (Semantic $\mathsf{CplxSer}^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 69.43 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9): the carrier is the complex partial-sum predicate, the classifier is* hsame *on the limit, and every closure law is given by an explicit constructive proof.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexPartSum semanticNameCert`
Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.complex series semantic name certificate`

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. □

Definition 69.45 (ComplexSeriesUp finite obligation inventory)**.** *The finite obligation inventory for* $\mathsf{CplxSer}^{\uparrow}$ *consists of the following rows over* BHist, UnaryHistory, Cont, hsame, Pkg, $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*:*

1. *a carrier row saying that a complex term family supplies* $\mathsf{Complex}^{\uparrow}$ *histories at every unary index and closed partial sums by* Cont *recursion;*
2. *a classifier row saying that two accepted packages classify together exactly when their displayed* $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$ *limits agree componentwise under* hsame*;*
3. *a stability row transporting term histories, partial sums, convergence moduli, and the displayed limit through pointwise* hsame*;*
4. *a convergence row exposing the regular Cauchy partial-sum package consumed by* $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$*;*
5. *an absolute-sum row exposing the* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *modulus partial sums used by* AbsConv*;*
6. *a Cauchy-product row exposing the two absolute-convergence inputs and the product partial-sum package;*
7. *a ledger row recording the source term family, partial sums, limit, absolute-modulus sums, product construction, and provenance witnesses.*

Theorem 69.46 (ComplexSeriesUp obligation inventory stability)**.** *Let* $\mathcal{P}$ *and* $\mathcal{Q}$ *be two finite* $\mathsf{CplxSer}^{\uparrow}$ *obligation inventories with pointwise* hsame *transport on the source term family, the closed partial-sum rows, the convergence package, the absolute-modulus package, and the displayed complex limit. If* $\mathcal{P}$ *is accepted by the inventory of Definition 69.45, then* $\mathcal{Q}$ *is accepted by the same inventory and has the same classifier result.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexSeriesObligationInventory stability`

Proof. First transport the term-sequence row pointwise. The carrier transport is exactly the local term-history transport used by Definition 69.2, so every unary index remains inside the complex term carrier.

Next transport the partial-sum recursion. The determinacy and successor-transport rows Definition 69.10 and Definition 69.8 repack the closed partial sums through the same Cont spine.

For convergence and the displayed limit, apply the checked convergence transport row named after Definition 69.38. It carries the regular Cauchy partial-sum package to the transported term family and preserves the $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$ classifier on the limit.

Finally, the absolute-modulus and Cauchy-product rows are visible ledger rows. Their transport uses Definition 69.26 and the Cauchy-product construction of Definition 69.35, so the downstream ledger is repacked without adding a new analytic principle. □

Theorem 69.47 (ComplexSeriesUp downstream consumer boundary)**.** *A downstream consumer of a* $\mathsf{CplxSer}^{\uparrow}$ *certificate may read exactly the term-sequence row, the closed partial-sum row, the displayed* $\mathsf{NameCert}_{\mathsf{CplxLim}^{\uparrow}}$ *limit row, the absolute-modulus row over* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, the Cauchy-product row when the two absolute-convergence inputs are present, and the provenance ledger. No consumer row exports a completeness theorem for arbitrary complex-series carriers or an analytic-continuation rule.*

Proof. Read the source, pattern, classifier, stability, and ledger fields in Definition 69.44. The source and pattern expose the term family and partial-sum convergence package; the classifier exposes only the displayed limit comparison.

The absolute-modulus and Cauchy-product surfaces are the explicit rows of Definition 69.42. They give consumers the real modulus sums and the constructed product package only when the corresponding witnesses are already present.

There is no row in the inventory of Definition 69.45 whose target is a completion object, an arbitrary carrier-selection rule, or an analytic-continuation datum. Therefore every consumer read factors through one of the listed rows. □

Lemma 69.48 (Pointwise sum is a complex term sequence)**.** *Let* $c, d : \mathsf{Hist} \to \mathsf{Hist}$ *be complex term sequences in the sense of Definition 69.1, and define their pointwise sum by*

$$(c \oplus d)(m) := \mathsf{append}(c(m), d(m)),$$

as in Definition 65.36. Then $c \oplus d$ *is again a complex term sequence.*

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexTermSeqCarrier pointwise append`

Proof. Fix a unary index m. Name the two term histories by fresh history variables C_m and D_m, with $C_m = c(m)$ and $D_m = d(m)$. The term-sequence hypotheses give complex-history carriers for C_m and D_m. By Definition 39.3, D_m is unary. The canonical append relation supplies a history A_m such that

$$\mathsf{Cont}(C_m, D_m, A_m) \quad \text{and} \quad A_m \mathsf{hsameappend}(C_m, D_m).$$

This continuation witness falls under complex carrier unary continuation closure from Definition 39.13, so the named pointwise result $(c \oplus d)(m)$ is a complex-history carrier. Since m was arbitrary among unary indices, the pointwise-sum family is a complex term sequence. □

Lean4 ✓ `BEDC.Derived.ComplexSeriesUp.ComplexTermSeqCarrier pointwise append components`

Theorem 69.49 (Complex partial sums commute with pointwise addition)**.** *Let* $c, d : \mathsf{Hist} \to \mathsf{Hist}$ *be complex term sequences, let* $(c \oplus d)(m) := \mathsf{append}(c(m), d(m))$*, and let* $z_c, z_d, z_\oplus : \mathsf{Hist}$ *be unary base histories with*

$$z_\oplus \mathsf{hsameappend}(z_c, z_d).$$

If n *is unary and*

$$\mathsf{ComplexPartSum}(z_c, c, n, S_c) \quad \textit{and} \quad \mathsf{ComplexPartSum}(z_d, d, n, S_d)$$

hold, then there is a history $S_\oplus$ *such that*

$$\mathsf{ComplexPartSum}(z_\oplus, c \oplus d, n, S_\oplus) \wedge S_\oplus \mathsf{hsameappend}(S_c, S_d).$$

Moreover every closed partial sum T *of* $c \oplus d$ *at the same index satisfies*

$$\mathsf{ComplexPartSum}(z_\oplus, c \oplus d, n, T) \Longrightarrow T \mathsf{hsameappend}(S_c, S_d).$$

Lean4 ✓ BEDC.Derived.ComplexSeriesUp.ComplexPartSum pointwise append partials unique

Proof. By Definition 69.48, $c \oplus d$ is a complex term sequence. Prove the displayed existence statement by unary induction on n, uniformly in the two input partial-sum witnesses.

In the base case $n = \mathsf{e}$, index-case inversion from Definition 69.4 identifies S_c with z_c and S_d with z_d. The zero constructor gives

$$\mathsf{ComplexPartSum}(z_\oplus, c \oplus d, \mathsf{e}, z_\oplus).$$

The base comparison $z_\oplus \mathsf{hsameappend}(z_c, z_d)$, append congruence from Definition 12.3, and the two endpoint identifications give $z_\oplus \mathsf{hsameappend}(S_c, S_d)$.

For the successor case $n = \mathsf{E1}(m)$, invert the two input partial-sum witnesses with Definition 69.4. This exposes predecessor sums P_c, P_d, bound term histories C_m, D_m with $C_m \mathsf{hsame} c(m)$ and $D_m \mathsf{hsame} d(m)$, and continuations

$$\mathsf{Cont}(P_c, C_m, S_c) \quad \text{and} \quad \mathsf{Cont}(P_d, D_m, S_d).$$

The induction hypothesis at m gives $P_\oplus$ with

$$\mathsf{ComplexPartSum}(z_\oplus, c \oplus d, m, P_\oplus) \wedge P_\oplus \mathsf{hsameappend}(P_c, P_d).$$

Let A_m be the bound pointwise-sum term history supplied by the canonical append evidence

$$\mathsf{Cont}(C_m, D_m, A_m) \quad \text{and} \quad A_m \mathsf{hsameappend}(C_m, D_m).$$

Let $S_\oplus$ be the fresh successor history supplied by

$$\mathsf{Cont}(P_\oplus, A_m, S_\oplus).$$

The step constructor for Definition 69.3 gives the closed partial sum for $c \oplus d$ at $\mathsf{E1}(m)$.

It remains to compare this result with the sum of the two displayed successor results. The term-sequence hypotheses and Definition 39.3 make C_m and D_m unary, while Definition 69.12 makes P_c and P_d unary. Associativity of unary continuation from Definition 29.6 and commutativity from Definition 9194.6 give the middle-four comparison

$$\mathsf{append}(\mathsf{append}(P_c, P_d), \mathsf{append}(C_m, D_m)) \mathsf{hsameappend}(\mathsf{append}(P_c, C_m), \mathsf{append}(P_d, D_m)).$$

The comparison $P_\oplus \mathsf{hsameappend}(P_c, P_d)$ and the term comparison $A_m \mathsf{hsameappend}(C_m, D_m)$ put the continuation witness for $S_\oplus$ in the same source and right-history class as the canonical append continuation to the left endpoint; Definition 21.7 gives the corresponding hsame comparison.

The two displayed continuation witnesses for S_c and S_d are compared with the canonical append continuations by Definition 21.7, yielding

$$\mathsf{append}(P_c, C_m) \mathsf{hsame} S_c \quad \text{and} \quad \mathsf{append}(P_d, D_m) \mathsf{hsame} S_d.$$

Applying Definition 12.3 to these two comparisons gives

$$\mathsf{append}(\mathsf{append}(P_c, C_m), \mathsf{append}(P_d, D_m)) \mathsf{hsameappend}(S_c, S_d).$$

Transitivity with the comparison for $S_\oplus$ and the middle-four comparison gives the required comparison for $S_\oplus$.

Finally, let T be any closed partial sum of $c \oplus d$ at n. The determinacy theorem Definition 69.5 compares T with the constructed $S_\oplus$; transitivity with the comparison already proved gives $T \mathsf{hsameappend}(S_c, S_d)$. □

Theorem 69.50 (ComplexSeries NameCert obligation surface)**.** *The ComplexTermSeq carrier transport, closed partial sums, Cauchy and absolute convergence predicates, comparison-test rows, ratio-test and root-test rows, Cauchy-product rows, and downstream boundary exclusions assemble the local* $\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$ *obligation surface. This surface is read over* $\mathsf{CplxLim}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *unary histories,* hsame, Cont, *and finite ledger provenance, and its consumer rows are exactly the rows displayed by Definition 69.45, Definition 69.46, and Definition 69.47. It contains no host infinite-sum operator, no choice of limits, no analytic-continuation datum, and no standard complex-analysis bridge data.*

Proof. Start with the carrier and recursion rows. The term carrier is transported by Definition 69.2, while the partial-sum surface is supplied by Definition 69.3, Definition 69.7, and Definition 69.5.

Next add the convergence and test rows. The Cauchy and absolute-convergence predicates are the rows of Definition 69.29 and Definition 69.27; the comparison, ratio, and root tests are the local finite rows of Definition 69.31, Definition 69.32, and Definition 69.33.

The product and ledger faces are already present in the same chapter. The Cauchy-product row is Definition 69.34 together with Definition 69.35, and the source, pattern, classifier, stability, and ledger fields are the five rows assembled in Definition 69.44.

Finally apply the finite inventory and downstream boundary theorems. They repack the listed rows as the local $\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$ surface over $\mathsf{CplxLim}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, unary histories, hsame, Cont, and finite ledger provenance, while leaving host infinite sums, selected limits, analytic continuation, and standard bridge data outside the exported surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$.

Constructive story: A complex series is generated by feeding a unary index into a family of $\mathsf{Complex}^{\uparrow}$ histories, then building its partial sums by the closed recursion that starts at the chosen zero history and appends the current term at each b1-successor. The carrier accepts term families whose partial-sum sequence comes with a convergence witness to a carried limit, while families without complex terms at unary indices, without closed partial sums, or without the convergence row remain outside this surface. Two accepted series classify together when their displayed limits agree componentwise under hsame, and the source specification is the convergent term-sequence row, the pattern specification is regular Cauchy convergence of the partial sums with explicit modulus, and the ledger keeps the term sequence, partial sums, absolute-modulus sums, Cauchy product, and pointwise addition rows visible. The five name-certificate obligations are witnessed by the zero partial-sum recursion for inhabitation, by the hsame equivalence laws on limits, and by the transport results saying that pointwise-equal terms have classified partial sums and the same convergence target. A term-family history h lies in the signature gap of a $\mathsf{CplxSer}^{\uparrow}$ token p when the source-domain accepts its convergent partial-sum package, the signature relation pairs it with an answer limit and evidence ledger, and the token-introduction relation names that limit as p; unfolded, the partial sums of h converge with the displayed modulus to the complex limit encoded by p. The chapter registers this partial-sum convergence construction under the name $\mathsf{CplxSer}^{\uparrow}$; it introduces no completeness theorem for arbitrary series carriers, no analytic-continuation machinery, and no standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped surface binds the partial-sum recursion, convergence classifier, absolute-sum rows, Cauchy-product rows, classifier transport, downstream boundary, BHist histories, hsame transport, Cont routes, and finite ledger provenance already displayed in the chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ComplexSeriesUp.complex\_series\_semantic\_name\_certificate`

Bridge status: none

Not claimed: The chapter does not close completeness of complex series carriers or analytic continuation results.

Upgrade path: The paper axis is mature for the displayed finite partial-sum, convergence, absolute-sum, Cauchy-product, classifier-transport, and ledger rows; only bridgeChecked formal acceptance remains.

Chapter 70

A Concrete Naming Certificate for $\mathsf{Dirichlet}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for Dirichlet series $\sum_{n=1}^{\infty} a_n n^{-s}$ with complex argument s. The construction sits above $\mathsf{CplxSer}^{\uparrow}$ (chapter 69) and $\mathsf{Nat}^{\uparrow}$ (chapter 28). The abscissa of absolute convergence is given as an explicit $\mathsf{Real}^{\uparrow}$-history witness rather than a real supremum; convergence in the open half-plane $\Re(s) > \sigma_a$ is proved by majorant comparison; holomorphicity in the half-plane is established with a uniform modulus over compact sub-half-planes; the Euler product over primes (chapter 64) is supplied for completely multiplicative coefficients. The construction is fully constructive and uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

70.1 Dirichlet term and partial sum

Definition 70.1 (Dirichlet term)**.** *For coefficient sequence $a : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$-history (with a_0 ignored, indexing starts at $\mathsf{E1}(\mathsf{e})$), positive unary n, complex history s, the Dirichlet term $\mathsf{DirTerm}(a, n, s, t_n)$ is the explicit complex history $t_n = a_n \cdot n^{-s}$, where n^{-s} is computed via $\exp(-s \log n)$ on the rational/real-history representation of $\log n$ (defer constructive log to the $\mathsf{Real}^{\uparrow}$-namecert machinery, chapter 38).*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletTerm`

Definition 70.2 (Dirichlet positive index)**.** *A Dirichlet positive index is a history of the form $\mathsf{E1}(t)$ where t is unary. This excludes the empty index and matches the $n \geq 1$ indexing used by Dirichlet terms.*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex`

Theorem 70.3 (Positive Dirichlet index is nonempty)**.** *If n is a Dirichlet positive index, then n is not* hsame*-empty.*

Proof. Unpack the positive-index witness $n = \mathsf{E1}(t)$. The visible-successor constructor is separated from e, so the displayed history cannot be hsame-empty. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex nonempty`

Lemma 70.4 (Positive Dirichlet index is append closed)**.** *If m is a Dirichlet positive index and n is unary, then $\mathsf{append}(m, n)$ is again a Dirichlet positive index.*

Proof. Write $m = \mathsf{E1}(t)$ with t unary. Unary append closure gives unary $\mathsf{append}(t, n)$, and the unary append law rewrites $\mathsf{append}(\mathsf{E1}(t), n)$ as $\mathsf{E1}(\mathsf{append}(t, n))$. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex append unary closed`

Theorem 70.5 (Positive Dirichlet index is unary-prepend closed)**.** *If m is unary and n is a Dirichlet positive index, then* $\mathsf{append}(m, n)$ *is again a Dirichlet positive index.*

Proof. Write $n = \mathsf{E1}(t)$ with t unary. By the right-recursive definition of append, $\mathsf{append}(m, \mathsf{E1}(t))$ is $\mathsf{E1}(\mathsf{append}(m, t))$, and unary append closure supplies unaryness of $\mathsf{append}(m, t)$. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex prepend unary closed`

Lemma 70.6 (Positive Dirichlet index append right cases)**.** *If* $\mathsf{append}(m, n)$ *is a positive Dirichlet index, then either m is a positive Dirichlet index or n is a positive Dirichlet index.*

Proof. Inspect the right append factor n. The empty case returns positivity of m. A zero-headed right factor cannot produce a visible successor result. A one-headed right factor exposes unaryness of its tail from unaryness of the successor witness, so n itself is a positive Dirichlet index. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex append right cases`

Theorem 70.7 (Positive Dirichlet index append iff)**.** *If m and n are unary histories, then* $\mathsf{append}(m, n)$ *is a positive Dirichlet index exactly when m is positive or n is positive.*

Proof. The forward implication is the append right case split. Conversely, positivity in the left component is closed under appending a unary right component, and positivity in the right component is closed under prepending a unary left component. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPositiveIndex append iff`

Lemma 70.8 (Dirichlet term well-defined)**.** *For every (a, n, s) with positive unary n and complex history s,* $\mathsf{DirTerm}(a, n, s, t_n)$ *determines a unique complex history t_n up to* hsame.

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletTerm complex carrier well defined`

Proof. Constructive existence and uniqueness of $\log n$ on positive unary n; composition with constructive complex exp and complex multiplication. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletTerm positive index append hsame transport`

Definition 70.9 (Dirichlet partial sum)**.** *For a, s and unary N, the Dirichlet partial sum* $\mathsf{DirPartSum}(a, s, N, S_N)$ *is the complex-history value of $\sum_{n=1}^{N} a_n n^{-s}$, defined by an explicit unary recursion.*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletPartSum`

Lemma 70.10 (Dirichlet partial-sum index cases)**.** *Every closed witness of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *either has the zero index $n = \mathsf{e}$ and identifies S with e, or has a successor index $n = \mathsf{E1}(m)$ together with a predecessor partial sum and the continuation step from that predecessor to S.*

Proof. Invert the closed partial-sum witness. The zero constructor gives the empty index and empty partial sum. The step constructor exposes the predecessor index, predecessor sum, and continuation payload directly. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum index cases`

Lemma 70.11 (Dirichlet partial-sum index is unary)**.** *Every closed witness of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *forces the index history* n *to be unary.*

Proof. Induct on the closed partial-sum witness. The zero constructor gives the empty unary history, and the step constructor applies unary closure under BHist.e1 to the predecessor index. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum index unary`

Theorem 70.12 (Dirichlet partial-sum results are unary)**.** *If every Dirichlet term history* $\mathsf{term}(m, s)$ *is unary whenever* m *is unary, then every closed* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *witness has unary result* S*.*

Proof. Induct on the closed Dirichlet partial-sum witness. The zero constructor gives the empty unary history. In the successor case, the predecessor result is unary by induction, the predecessor index is unary by DirichletPartSum index closure, and unary continuation closure gives the successor result. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum result unary`

Theorem 70.13 (Dirichlet partial-sum nonempty index is positive)**.** *Every closed witness of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *whose index is not* hsame*-empty has a positive Dirichlet index* n*.*

Proof. The partial-sum witness makes n unary. Applying nonempty unary-tail inversion to n exposes $n = \mathsf{E1}(t)$ for a unary tail t, which is exactly the positive Dirichlet index witness. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum nonempty index positive`

Theorem 70.14 (Dirichlet positive-index partial-sum result is nonempty)**.** *Assume every unary-indexed Dirichlet term* $\mathsf{term}(m, s)$ *is not* hsame*-empty. If a closed partial-sum witness* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *has a positive Dirichlet index* n*, then* S *is not* hsame*-empty.*

Proof. Unpack the positive-index witness as $n = \mathsf{E1}(t)$ with t unary. Inverting the matching partial-sum derivation exposes the predecessor sum and the final continuation step from the predecessor to $\mathsf{term}(t, s)$. If the result were empty, continuation result transport and empty-result inversion would make $\mathsf{term}(t, s)$ empty, contradicting the term nonemptiness hypothesis at the unary tail t. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum positive index result nonempty`

Theorem 70.15 (Dirichlet partial-sum positive index exposes predecessor)**.** *Every closed witness of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, S)$ *whose index is a positive Dirichlet index exposes histories* m *and* P *such that*

$$n = \mathsf{E1}(m) \wedge \mathsf{UnaryHistory}(m) \wedge \mathsf{DirichletPartSum}(\mathsf{term}, s, m, P) \wedge \mathsf{Cont}(P, \mathsf{term}(m, s), S).$$

Proof. Case-split the closed partial-sum witness. The zero constructor is incompatible with a positive Dirichlet index, since positive indices are nonempty. The successor constructor directly exposes the predecessor index, its unary carrier, the predecessor partial-sum witness, and the continuation ledger into S. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum positive index predecessor`

Theorem 70.16 (Dirichlet append-successor partial-sum predecessor)**.** *Every closed witness of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, \mathsf{append}(m, \mathsf{E1}(n)), S)$ *exposes a predecessor partial sum at* $\mathsf{append}(m, n)$*:*

$$\exists P : \mathsf{BHist},\ \mathsf{DirichletPartSum}(\mathsf{term}, s, \mathsf{append}(m, n), P) \wedge \mathsf{Cont}(P, \mathsf{term}(\mathsf{append}(m, n), s), S).$$

Proof. By the definition of history append, $\mathsf{append}(m, \mathsf{E1}(n))$ is the successor history $\mathsf{E1}(\mathsf{append}(m, n))$. Inverting the closed partial-sum witness therefore exposes the predecessor sum at $\mathsf{append}(m, n)$ and the continuation step into S. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum append successor predecessor`

Lemma 70.17 (Dirichlet partial-sum successor result is deterministic)**.** *If the Dirichlet partial-sum recursion reaches* S *at unary index* n*, and the next term continuation sends* S *and* $\mathsf{term}(n, s)$ *to* T*, then every closed derivation of the successor partial sum at* $\mathsf{E1}(n)$ *has result* hsame *to* T*.*

Proof. Induct on the previous Dirichlet partial-sum derivation to obtain determinacy at each index. In the successor case, the other derivation has the same successor shape; the induction hypothesis identifies the previous sums, and determinacy of Cont identifies the successor result. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum successor result deterministic`

Lemma 70.18 (Dirichlet successor term nonempty forces result nonempty)**.** *For any Dirichlet term function* term*, if*

$$\mathsf{DirichletPartSum}(\mathsf{term}, s, \mathsf{E1}(n), S)$$

holds and $\mathsf{term}(n, s)$ *is not* hsame *to* e*, then* S *is not* hsame *to* e*.*

Proof. Invert the successor partial-sum witness. Its displayed continuation ledger has the form $\mathsf{Cont}(P, \mathsf{term}(n, s), S)$ for the previous partial sum P. If $S\mathsf{hsamee}$, result transport gives a continuation into e; empty-result continuation inversion then forces $\mathsf{term}(n, s)\mathsf{hsamee}$, contradicting the term nonemptiness assumption. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum successor term nonempty result nonempty`

Theorem 70.19 (Dirichlet partial-sum successor result is nonempty)**.** *Assume every unary-indexed Dirichlet term is not* hsame*-empty. Then any closed successor partial sum* $\mathsf{DirichletPartSum}(\mathsf{term}, s, \mathsf{BHist.e1}(n), S)$ *has nonempty result:*

$$\mathsf{hsame}(S, \mathsf{e}) \to \bot.$$

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum successor result nonempty`

Proof. Invert the successor partial-sum witness. If the successor result were empty, transport the displayed continuation step to an empty result and apply $\mathsf{cont_empty_result_inversion}$. The current term is then hsame-empty, while the predecessor partial-sum witness proves its index is unary by $\mathsf{DirichletPartSum}$ index closure, contradicting the term-nonempty hypothesis. □

Lemma 70.20 (Dirichlet partial-sum unary index is deterministic)**.** *If* n *is a unary history, then any two closed derivations of* $\mathsf{DirichletPartSum}(\mathsf{term}, s, n, -)$ *have* hsame *results.*

Proof. Induct on the first Dirichlet partial-sum derivation. The empty index case forces the second derivation to be the base constructor. In the successor case, unary inversion gives the predecessor index, the induction hypothesis identifies the previous partial sums, and determinacy of Cont identifies the successor results. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum unary index deterministic`

Lemma 70.21 (Dirichlet partial sums respect pointwise term equivalence)**.** *If two Dirichlet term functions are pointwise related by* hsame *on every unary index for the fixed complex argument s, and S, T are closed Dirichlet partial sums for those two term functions at the same index n, then* $S\,\mathsf{hsame}\,T$.

Proof. Induct on the first closed Dirichlet partial-sum witness. The base case forces the second witness to be the zero constructor. In the successor case, the second witness has the same successor index; the induction hypothesis identifies the predecessor sums, the first witness supplies unaryness of the predecessor index, and the pointwise term comparison identifies the two terms at that index. Determinacy of Cont identifies the successor sums. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum term hsame deterministic`

Lemma 70.22 (Dirichlet partial-sum term transport is deterministic)**.** *Suppose two Dirichlet term functions agree under* hsame *at every unary index after transporting the ambient parameter from s to s'. If n is unary, then a closed partial sum S for* term *at (s, n) and a closed partial sum T for* term' *at (s', n) satisfy* $S\,\mathsf{hsame}\,T$.

Proof. Transport the first closed partial-sum witness along the pointwise term-history comparison to obtain a closed witness for term' at the same unary index. Unary-index determinacy for Dirichlet partial sums then compares that transported witness with the displayed T witness. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum term hsame transport deterministic`

Lemma 70.23 (Dirichlet partial sums are complex histories)**.** *For every (a, s, N), there is a unique complex history S_N with* $\mathsf{DirPartSum}(a, s, N, S_N)$*, up to* hsame.

Proof. Unary induction on N constructs the empty partial sum at the base index and extends the predecessor sum by the concrete continuation step carrying the current Dirichlet term. At each unary index, the partial-sum determinacy theorem identifies any competing closed derivation with the constructed result. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum exists unique`

Theorem 70.24 (Dirichlet series successor index tail is nonempty)**.** *For every history n :* BHist*, if n is a unary index, then* $\mathsf{BHist.e1}(n)$ *is again a unary index and* $\mathsf{hsame}(\mathsf{BHist.e1}(n), \mathsf{e}) \to \bot$.

Proof. Unary indices are closed under the E1 constructor. The constructor BHist.e1 is disjoint from e, so the successor index supplies a nonempty tail witness for Dirichlet indexing. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletSeriesIndex e1 tail nonempty`

Theorem 70.25 (Dirichlet series appended unary tail is nonempty)**.** *For histories $n, tail$:* BHist*, if n is a unary index distinct from* e *and tail is unary, then* $\mathsf{append}(n, tail)$ *is unary and* $\mathsf{hsame}(\mathsf{append}(n, tail), \mathsf{e}) \to \bot$.

Proof. Unary histories are closed under append. If the appended history were empty, append-empty inversion would force the left factor n to be empty, contrary to the supplied nonempty index witness. □

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletSeriesIndex append unary tail nonempty`

70.2 Abscissa of absolute convergence

Definition 70.26 (Abscissa of absolute convergence)**.** *For coefficient sequence a, the abscissa of absolute convergence* $\mathsf{AbsConvAbscissa}(a, \sigma_a)$ *is the located boundary predicate on a real-history σ_a with two clauses.*

The right clause says that for every unary precision k and every complex s, if a BEDC real-history witness supplies the rational comparison $\Re(s) - \sigma_a \geq 2^{-k}$, then the modulus partial sums of the Dirichlet series at s are uniformly bounded above by an explicit constant.

The left clause says that for every unary precision k, every rational probe q, and every BEDC real-history witness supplying the rational comparison $\sigma_a - q \geq 2^{-k}$, the modulus partial sums at the probe point with real part q fail to admit any such uniform bound. All uses of order in the predicate are therefore supplied by displayed real-history comparison witnesses, not inferred from distinctness of candidates.

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.AbsConvAbscissa`

Theorem 70.27 (Abscissa is unique)**.** *If* $\mathsf{AbsConvAbscissa}(a, \sigma_a)$ *and* $\mathsf{AbsConvAbscissa}(a, \sigma'_a)$*, then* $\mathsf{hsame}(\sigma_a, \sigma'_a)$ *on the rational representation of the real-history.*

Sketch. It is enough to rule out every explicit rational gap between the two real-histories. Suppose a unary precision k and BEDC comparison witnesses exhibit the oriented gap

$$\sigma'_a + 2^{-k} \leq q \leq \sigma_a - 2^{-k}.$$

Apply the right clause for σ'_a to the probe point with real part q. The right comparison witness gives a uniform bound for the corresponding modulus partial sums.

Apply the left clause for σ_a to the same q and the displayed gap witness. It says that no such uniform bound exists. This contradiction excludes the oriented gap from σ'_a to σ_a.

The reverse orientation is identical with σ_a and σ'_a exchanged. Thus every BEDC rational-history separation witness is impossible, which is exactly $\mathsf{hsame}(\sigma_a, \sigma'_a)$. □

70.3 Convergence and holomorphicity in the half-plane

Theorem 70.28 (Half-plane convergence)**.** *If* $\mathsf{AbsConvAbscissa}(a, \sigma_a)$ *and s satisfies $\Re(s) > \sigma_a$ (decidable strict inequality with explicit witness gap ϵ), then $\sum a_n n^{-s}$ converges absolutely in* $\mathsf{CplxSer}^{\uparrow}$ *with explicit modulus given by majorant comparison to $\sum n^{-(\sigma_a+\epsilon/2)}$.*

Sketch. The bound $|a_n n^{-s}| \leq C n^{-(\sigma_a+\epsilon/2)}$ is given by $\mathsf{AbsConvAbscissa}$ at precision k chosen by ϵ; the resulting series converges with explicit geometric-like modulus. □

Theorem 70.29 (Half-plane holomorphicity)**.** *The Dirichlet sum function $s \mapsto \sum_{n=1}^{\infty} a_n n^{-s}$ is holomorphic in the open half-plane $\Re(s) > \sigma_a$, with uniform modulus over each compact sub-half-plane $\{s : \Re(s) \geq \sigma_a + \epsilon, |\Im(s)| \leq T\}$ for explicit unary ϵ, T (cite chapter 68).*

Sketch. Term-wise complex differentiability of n^{-s} gives derivative $-(\log n)n^{-s}$; the differentiated series converges absolutely on the compact sub-half-plane by the same majorant. □

70.4 Euler product for multiplicative coefficients

Definition 70.30 (Completely multiplicative coefficient)**.** *A coefficient sequence a is completely multiplicative when $a_1 = \mathsf{E1}(\mathsf{e})$ and for all positive unary m, n, $a_{m \cdot n} = a_m \cdot a_n$ (product in* $\mathsf{Complex}^{\uparrow}$*).*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletCoefficient completely multiplicative`

Theorem 70.31 (Euler product)**.** *If a is completely multiplicative and $\Re(s) > \sigma_a$, the Dirichlet series factors as the explicit Euler product $\sum a_n n^{-s} = \prod_p (1 - a_p p^{-s})^{-1}$ over primes p (in the sense of* $\mathsf{Prime}^{\uparrow}$*, chapter 64). The product converges absolutely with explicit modulus over compact sub-half-planes.*

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletCoefficient completely multiplicative product nonempty`

Sketch. Group terms by prime factorization (factorization_exists of chapter 64); the geometric series $\sum_k (a_p p^{-s})^k$ gives the local factor at each prime. □

70.5 The certificate

Definition 70.32 (Dirichlet source specification)**.** *The source specification on (a, s) carries the constraint* $\mathsf{AbsConvAbscissa}(a, \sigma_a)$ *and $\Re(s) > \sigma_a$.*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletSourceSpec`
Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletSourceSpec witness result carrier`

Definition 70.33 (Dirichlet pattern specification)**.** *The pattern is the regular Cauchy partial-sum predicate of chapter 69 applied to the Dirichlet term sequence.*

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum complexPartSum`

Definition 70.34 (Dirichlet classifier specification)**.** *Two convergent Dirichlet series at the same s are classified together when their limits agree under* hsame *on both real components.*

Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletClassifierSpec`
Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletClassifierSpec limit transport`
Lean4 (definition) `BEDC.Derived.DirichletSeriesUp.DirichletSeriesClassifierSpec`

Definition 70.35 (Dirichlet stability certificate)**.** *The stability certificate consists of* hsame*-respect on coefficients and on s, transport of the abscissa under* hsame*, and compatibility with both partial-sum and limit predicates.*

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum term hsame transport`

Definition 70.36 (Dirichlet ledger policy)**.** *The ledger policy records: (i) absolute convergence in the open half-plane, (ii) holomorphicity over compact sub-half-planes, (iii) Euler product for completely multiplicative coefficients, (iv) coefficient linearity in the half-plane.*

Theorem 70.37 (Concrete $\mathsf{Dirichlet}^{\uparrow}$ naming certificate)**.** *The five fields above form a naming certificate over* Hist *for the Dirichlet series interface* $\mathsf{Dirichlet}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.dirichlet name certificate`

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Theorem 70.38 (Semantic Dirichlet$^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 70.37 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9).*

Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.DirichletPartSum semanticNameCert`
Lean4 ✓ `BEDC.Derived.DirichletSeriesUp.dirichlet semantic name certificate`

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. □

Closure status for NameCert$_{\text{Dirichlet}\uparrow}$**.**

Constructive story: A Dirichlet$^{\uparrow}$ packet is generated from finite BHist rows for positive unary indices, coefficient histories, Dirichlet terms, partial sums, convergence windows, abscissa witnesses, and Euler-product source rows. Each term is built by Cont from the coefficient row and the displayed n^{-s} complex-history row, and partial sums are finite continuation folds over positive unary indices. The classifier transports indices, terms, sums, convergence ledgers, and product rows by hsame, while Pkg records the complex-series, natural-number, prime, real, and analytic dependencies. The packet names the constructive Dirichlet-series surface without importing a host infinite sum.

Theory closure: TheoryGate(NameCert$_{\text{Dirichlet}\uparrow}$, seed)

Scope closed: This chapter binds Dirichlet terms, positive-index facts, partial sums, convergence witnesses, Euler-product rows, and the semantic NameCert surface for Dirichlet$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{Dirichlet}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a host analytic function object, classical supremum abscissa, quotient equality of series, nonconstructive convergence selection, or an external infinite-sum primitive.

Upgrade path: Obligation closure requires Lean encodings of the term and index carriers, finite partial-sum routes, convergence ledgers, Euler-product rows, and semantic NameCert theorem.

Chapter 71

A Concrete Naming Certificate for $\mathsf{ZetaBasic}^{\uparrow}$ on $\Re(s) > 1$

This chapter gives a concrete naming-certificate construction for the Riemann zeta function $\zeta(s) = \sum_{n=1}^{\infty} n^{-s}$ on the open half-plane $\Re(s) > 1$, where the defining Dirichlet series converges absolutely. It is a direct specialization of $\mathsf{Dirichlet}^{\uparrow}$ (chapter 70) at the unit coefficient sequence $a_n = \mathsf{E1}(\mathsf{e})$. The chapter does NOT extend ζ to $\mathsf{Complex}^{\uparrow} \setminus \{1\}$ — that is a separate operation chapter ($\mathsf{ZetaCont}^{\uparrow}$, chapter 9342). Two faces of the same object are presented: the Dirichlet sum and the Euler product over primes; non-vanishing on the half-plane is supplied with explicit lower bound; holomorphicity is inherited from $\mathsf{Dirichlet}^{\uparrow}$. The construction is fully constructive and uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

71.1 The constant-coefficient Dirichlet sum

Definition 71.1 (Zeta basic predicate)**.** *For complex history s and complex history z, the predicate* $\mathsf{ZetaBasic}(s, z)$ *holds when z is the explicit complex-history limit of the Dirichlet partial sums of the constant-unit coefficient sequence $a_n = \mathsf{E1}(\mathsf{e})$ at s; equivalently,* $\mathsf{ZetaBasic}(s, z)$ *is the specialization of the* $\mathsf{Dirichlet}^{\uparrow}$ *carrier (chapter 70) at unit coefficients.*

Lean4 (definition) `BEDC.Derived.ZetaBasicUp.ZetaBasic`

Definition 71.2 (Unit Dirichlet part sum)**.** *For a complex history s, the unit term at unary index n is the visible unit extension* $\mathsf{E1}(s)$*, and* $\mathsf{ZetaBasicPartSum}(s, n, z)$ *is the specialized* DirichletPartSum *witness for this unit term.*

Lean4 (definition) `BEDC.Derived.ZetaBasicUp.ZetaBasicUnitTerm`
Lean4 (definition) `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum`

Theorem 71.3 (Unit Dirichlet part sums are unary)**.** *If s and n are unary histories and* $\mathsf{ZetaBasicPartSum}(s, n, z)$ *holds, then z is a unary history.*

Proof. Induct on the concrete DirichletPartSum witness. The zero result is e, and the successor case appends the prior unary partial sum to the visible unit term $\mathsf{E1}(s)$, which is closed by unary continuation. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum unary result`

Theorem 71.4 (Unit Dirichlet part-sum indices are unary)**.** *If* $\mathsf{ZetaBasicPartSum}(s, n, z)$ *holds, then* n *is a unary history.*

Proof. Induct on the specialized DirichletPartSum witness. The zero constructor gives the empty unary index, and the step constructor preserves unaryness by applying the visible unit successor constructor to the induction hypothesis. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum index unary`

Theorem 71.5 (Unit Dirichlet successor part sums are nonempty)**.** *If* $\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z)$ *holds, then* z *is not* hsame *to* e.

Proof. Invert the successor partial-sum witness. Its final step appends the previous sum to $\mathsf{E1}(s)$. If the result were empty, empty-result inversion for Cont would force $\mathsf{E1}(s)$hsamee, contradicting visible-successor separation. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor result nonempty`

Theorem 71.6 (Unit Dirichlet positive-index part sums are nonempty)**.** *If* n *is a positive Dirichlet index and* $\mathsf{ZetaBasicPartSum}(s, n, z)$ *holds, then* z *is not* hsame *to* e.

Proof. The positive-index witness exposes $n = \mathsf{E1}(t)$ for a unary tail t. After substituting this visible successor shape, the successor nonempty theorem applies to the displayed partial-sum witness. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum positive index result nonempty`

Theorem 71.7 (Unit Dirichlet successor part-sum step inversion)**.** *If* $\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z)$ *holds, then there is a previous partial sum* p *such that* $\mathsf{ZetaBasicPartSum}(s, n, p)$ *holds and* $\mathsf{Cont}(p, \mathsf{E1}(s), z)$ *is the successor continuation step.*

Proof. Invert the specialized partial-sum witness at successor index $\mathsf{E1}(n)$. The step constructor exposes the predecessor partial sum and its continuation by the unit term, which unfolds to $\mathsf{E1}(s)$. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor step inversion`

Theorem 71.8 (Unit Dirichlet empty-source successor result shape)**.** *If* $\mathsf{ZetaBasicPartSum}(\mathsf{e}, \mathsf{E1}(n), z)$ *holds, then there is a previous partial sum* p *such that* $\mathsf{ZetaBasicPartSum}(\mathsf{e}, n, p)$ *holds and* $z\mathsf{hsame}\mathsf{E1}(p)$.

Proof. Invert the successor part-sum witness. At empty source the unit term unfolds to $\mathsf{E1}(\mathsf{e})$, so the exposed continuation has result $p \cdot \mathsf{E1}(\mathsf{e})$. The append right-unit law rewrites this result as $\mathsf{E1}(p)$. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum empty source successor result shape`

Theorem 71.9 (Unit Dirichlet successor source transport step)**.** *If* $s\mathsf{hsame}t$ *and* $\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z)$ *holds, then there are histories* p *and* u *such that* $\mathsf{ZetaBasicPartSum}(t, n, p)$, $\mathsf{Cont}(p, \mathsf{E1}(t), u)$, *and* $z\mathsf{hsame}u$.

Proof. Invert the successor part-sum witness at source s. The predecessor index is unary by the part-sum index theorem, so termwise hsame-transport along $s\mathsf{hsame}t$ moves the predecessor partial sum to source t. Rebuild the visible unit continuation step at t, and use continuation congruence to compare its result with the original successor sum. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor source hsame transport step`

Theorem 71.10 (Unit Dirichlet successor previous-sum determinism)**.** *If* $s\mathsf{hsame}t$ *and both sources carry successor-index unit partial sums at the same predecessor index* n*,*

$$\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z) \quad \textit{and} \quad \mathsf{ZetaBasicPartSum}(t, \mathsf{E1}(n), u),$$

then their predecessor sums can be chosen as histories p, q *with* $\mathsf{ZetaBasicPartSum}(s, n, p)$*,* $\mathsf{ZetaBasicPartSum}(t, n, q)$*,* $p\mathsf{hsame}q$*, the two visible unit continuation steps* $\mathsf{Cont}(p, \mathsf{E1}(s), z)$*,* $\mathsf{Cont}(q, \mathsf{E1}(t), u)$*, and* $z\mathsf{hsame}u$*.*

Proof. Invert both successor partial-sum witnesses. The predecessor index is unary by the index theorem, so termwise source transport compares the two predecessor sums. The visible unit terms compare by $s\mathsf{hsame}t$, and continuation congruence compares the two successor results. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor source hsame previous deterministic`

Theorem 71.11 (Unit Dirichlet successor source transport result nonempty)**.** *If* $s\mathsf{hsame}t$ *and* $\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z)$ *holds, then there are histories* p *and* u *such that* $\mathsf{ZetaBasicPartSum}(t, n, p)$*,* $\mathsf{Cont}(p, \mathsf{E1}(t), u)$*,* $z\mathsf{hsame}u$*, and* u *is not* hsame *to* e*.*

Proof. Apply the source-transport step theorem to obtain the transported predecessor partial sum and the rebuilt visible-unit continuation at source t. This continuation is a successor $\mathsf{ZetaBasicPartSum}$ witness at source t, so successor result nonemptiness rules out $u\mathsf{hsame}\mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor source hsame transport result nonempty`

Theorem 71.12 (Unit Dirichlet successor source transport preserves nonempty result)**.** *If* $s\mathsf{hsame}t$ *and* $\mathsf{ZetaBasicPartSum}(s, \mathsf{E1}(n), z)$ *holds, then there is a transported successor partial sum* u *at source* t *such that* $\mathsf{ZetaBasicPartSum}(t, \mathsf{E1}(n), u)$*,* $z\mathsf{hsame}u$*, and* u *is not* hsame *to* e*.*

Proof. Use successor source transport to obtain the transported predecessor sum and its visible unit continuation at t. Rebuild the successor $\mathsf{ZetaBasicPartSum}$ witness from that continuation step, retain the result comparison with z, and apply successor-result nonemptiness to the rebuilt target sum. □

Lean4 ✓ `BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor source result nonempty transport`

Lemma 71.13 (Zeta basic well-defined for $\Re(s) > 1$)**.** *For every* s *with rational representation of* $\Re(s)$ *strictly greater than* $\mathsf{E1}(\mathsf{e})$ *(decidable on the rational representation), the predicate* $\mathsf{ZetaBasic}(s, z)$ *holds for some complex history* z*, and* z *is unique up to* hsame *on both real components.*

Sketch. Specialize Definition 70.28 of $\mathsf{Dirichlet}^{\uparrow}$ at unit coefficients with abscissa of absolute convergence exactly $\mathsf{E1}(\mathsf{e})$; existence and uniqueness of the limit follow from the $\mathsf{CplxLim}^{\uparrow}$ certificate (chapter 65). □

Theorem 71.14 (Zeta basic unique)**.** *If* $\mathsf{ZetaBasic}(s, z)$ *and* $\mathsf{ZetaBasic}(s, z')$ *both hold for the same* s*, then* hsame *holds on both real components of* z *and* z'*.*

Proof. Direct uniqueness from $\mathsf{Dirichlet}^{\uparrow}$ limit uniqueness. □

71.2 Convergence and abscissa

Theorem 71.15 (Abscissa is exactly 1)**.** *The abscissa of absolute convergence for the unit Dirichlet series is the rational unit* $\mathsf{E1(e)}$*:* $\mathsf{AbsConvAbscissa}(\mathbf{1}, \mathsf{E1(e)})$*. The harmonic series at* $s = 1$ *diverges with explicit lower-bound logarithmic-rate witness, so the abscissa cannot be smaller; the geometric majorant for* $s > 1$ *shows that the abscissa is exactly 1.*

Sketch. The harmonic divergence is constructively witnessed by the explicit comparison $\sum_{n=1}^{2^k} 1/n \geq k/2$, proving abscissa is at least 1; the half-plane convergence theorem at any s with $\Re(s) > 1$ shows the abscissa is at most 1. □

71.3 The Euler product over primes

Theorem 71.16 (Euler product for ζ)**.** *For every* s *with* $\Re(s) > 1$*, the explicit factorization*

$$\zeta(s) = \prod_p (1 - p^{-s})^{-1}$$

holds, where the product runs over primes p *in the sense of* $\mathsf{Prime}^{\uparrow}$ *(chapter 64). The product converges absolutely with explicit modulus over compact sub-half-planes* $\{s : \Re(s) \geq 1 + \epsilon, |\Im(s)| \leq T\}$ *for unary* ϵ, T*.*

Sketch. Direct specialization of the Euler product theorem of $\mathsf{Dirichlet}^{\uparrow}$ (Definition 70.31) at completely multiplicative coefficients $a_n = \mathsf{E1(e)}$ with $a_p = \mathsf{E1(e)}$ for each prime p. □

71.4 Non-vanishing on the half-plane

Theorem 71.17 (ζ non-vanishing for $\Re(s) > 1$)**.** *For every* s *with* $\Re(s) > 1$*,* $\zeta(s) \neq 0$ *in the strict sense: the modulus* $|\zeta(s)|$ *admits an explicit positive lower bound depending on* $\Re(s)$*, given by the geometric tail of the Euler product.*

Sketch. The Euler product factors are each bounded below in modulus by $1 - p^{-\Re(s)} > 0$; the infinite product converges with explicit modulus, so the limiting modulus is bounded below by an explicit positive constant. □

71.5 Holomorphicity on the half-plane

Theorem 71.18 (ζ holomorphic on $\Re(s) > 1$)**.** *The function* $s \mapsto \zeta(s)$ *is holomorphic in the open half-plane* $\Re(s) > 1$ *in the sense of* $\mathsf{Holomorphic}^{\uparrow}$ *(chapter 68), with uniform modulus over compact sub-half-planes.*

Proof. Direct specialization of Definition 70.29 of $\mathsf{Dirichlet}^{\uparrow}$ at unit coefficients. □

71.6 The certificate

Definition 71.19 (Zeta basic source specification)**.** *The source specification on* s *carries the constraint* $\Re(s) > 1$ *(decidable on the rational representation).*

Lean4 (definition) BEDC.Derived.ZetaBasicUp.ZetaBasicSourceSpec
Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicSourceSpec unit term carrier

Definition 71.20 (Zeta basic pattern specification)**.** *The pattern is the unit-coefficient Dirichlet partial-sum convergence predicate, equivalently the Euler-product convergence predicate.*

Lean4 (definition) BEDC.Derived.ZetaBasicUp.ZetaBasicPatternSpec
Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicPatternSpec successor step inversion

Definition 71.21 (Zeta basic classifier specification)**.** *Two ζ-values at the same s are classified together when they agree under* hsame *on both real components.*

Lean4 (definition) BEDC.Derived.ZetaBasicUp.ZetaBasicClassifierSpec

Definition 71.22 (Zeta basic stability certificate)**.** *The stability certificate consists of* hsame-*respect on s, transport of partial-sum modulus under* hsame*, and equivalence of the Dirichlet-sum and Euler-product values.*

Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicClassifierSpec successor limit transport
Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicPatternSpec successor source hsame transport

Definition 71.23 (Zeta basic ledger policy)**.** *The ledger policy records: (i) abscissa exactly 1, (ii) Euler product factorization, (iii) non-vanishing on the half-plane with explicit lower bound, (iv) holomorphicity over compact sub-half-planes.*

Lean4 (definition) BEDC.Derived.ZetaBasicUp.ZetaBasicLedgerPolicy
Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicLedgerPolicy carrier fields

Theorem 71.24 (Concrete $\mathsf{ZetaBasic}^{\uparrow}$ naming certificate)**.** *The five fields above form a naming certificate over* Hist *for the basic ζ interface* $\mathsf{ZetaBasic}^{\uparrow}$ *on $\Re(s) > 1$.*

Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasicPartSum successor nonempty name certificate

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Theorem 71.25 (Semantic $\mathsf{ZetaBasic}^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 71.24 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9).*

Lean4 ✓ BEDC.Derived.ZetaBasicUp.ZetaBasic semanticNameCert
Lean4 ✓ BEDC.Derived.ZetaBasicUp.zeta basic semantic name certificate

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZetaBasic}^{\uparrow}}$.

Constructive story: A $\mathsf{ZetaBasic}^{\uparrow}$ packet is generated by specializing the $\mathsf{Dirichlet}^{\uparrow}$ rows to the constant unit coefficient sequence over the open half-plane $\Re(s) > 1$. Finite BHist rows record unit terms, unary partial sums, positivity of indices, Euler-product factors, lower-bound ledgers, and holomorphic readbacks inherited through displayed routes. The defining sum and product are replayed by Cont through the Dirichlet-series and prime-row dependencies, and hsame transports the same unit-term and partial-sum witnesses. Pkg records the Dirichlet, complex, real, and prime sources, so the packet remains a constructive half-plane certificate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZetaBasic}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the zeta-basic predicate, unit term and partial-sum rows, index facts, half-plane convergence and nonvanishing rows, Euler-product surface, and semantic NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZetaBasic}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim meromorphic continuation, the pole at 1, functional equation, critical-strip information, host analytic equality, or zeros outside the displayed half-plane surface.

Upgrade path: Obligation closure requires Lean encodings of the unit Dirichlet specialization, partial-sum route, half-plane ledgers, Euler-product rows, and semantic NameCert theorem.

Chapter 72

A Concrete Naming Certificate for $\mathsf{CritStrip}^{\uparrow}$

This chapter records the finite-unary obstruction boundary for $\mathsf{CritStrip}^{\uparrow}$ as predicates on $\mathsf{Complex}^{\uparrow}$-history. The nominal strip predicate is written with strict unary prefix witnesses between e and $\mathsf{E1}(\mathsf{e})$, so the finite carrier exposes an empty open interval rather than a nonempty analytic critical strip. The chapter isolates that obstruction before $\mathsf{ZetaCont}^{\uparrow}$ (chapter 9342) and $\mathsf{ZetaZeros}^{\uparrow}$ (chapter 73) consume any strip-shaped interface. The construction is fully constructive and uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

72.1 The strip and critical-line predicates

Definition 72.1 (Critical strip membership)**.** *For complex history s with rational real component σ, the predicate* $\mathsf{InCritStrip}(s)$ *holds when both* $\mathsf{NatUnaryStrictPrefix}(\mathsf{e}, \sigma)$ *and* $\mathsf{NatUnaryStrictPrefix}(\sigma, \mathsf{E1}(\mathsf{e}))$ *are decidably true on the rational representation, i.e.* $0 < \sigma < 1$ *strictly.*

Lean4 (definition) `BEDC.Derived.CritStripUp.InCritStrip`

Definition 72.2 (Critical strip open interval)**.** *For a rational real-component history σ, the open-interval core of the critical strip is the pair of strict unary prefix witnesses*

$$\mathsf{NatUnaryStrictPrefix}(\mathsf{e}, \sigma) \quad \textit{and} \quad \mathsf{NatUnaryStrictPrefix}(\sigma, \mathsf{E1}(\mathsf{e})).$$

Lean4 (definition) `BEDC.Derived.CritStripUp.CritStripOpenInterval`

Lemma 72.3 (Critical strip membership decidable)**.** *For every complex history s,* $\mathsf{InCritStrip}(s)$ *is decidable: an explicit unary algorithm returns either a witness for both strict inequalities or a witness for at least one failure.*

Proof. Both inequalities are decidable on rational representations by the strict-prefix decidability of $\mathsf{NatUnaryStrictPrefix}$ (chapter 28). □

Lean4 (definition) `BEDC.Derived.CritStripUp.InCritStrip open interval decidable`

Definition 72.4 (Critical line membership)**.** *For complex history s with rational real component σ, the predicate* $\mathsf{OnCritLine}(s)$ *holds when* $\mathsf{hsame}(\sigma, \mathsf{Half})$*, where* Half *is the rational representation of* $1/2$*.*

Lean4 (definition) `BEDC.Derived.CritStripUp.OnCritLine`

Lemma 72.5 (Critical line membership decidable)**.** *For every complex history s with rational real component,* $\mathsf{OnCritLine}(s)$ *is decidable.*

Proof. Decidable from hsame-decidability on rational representations of unary histories (chapter 37). □

Lean4 ✓ `BEDC.Derived.CritStripUp.OnCritLine decidable`

72.2 Compact sub-strips and boundary witnesses

Definition 72.6 (Compact sub-strip)**.** *For unary ϵ (with $\epsilon < 1/2$) and unary T, the compact sub-strip predicate* $\mathsf{CompactSubStrip}(\epsilon, T, s)$ *holds when $\epsilon \leq \sigma \leq 1 - \epsilon$ (rational comparisons) and $|t| \leq T$ (modulus comparison on the imaginary part).*

Lean4 (definition) `BEDC.Derived.CritStripUp.CompactSubStrip`

Lemma 72.7 (Compact sub-strip decidable)**.** *For every (ϵ, T, s),* $\mathsf{CompactSubStrip}(\epsilon, T, s)$ *is decidable; the decision procedure returns either witnesses for all three bounds or an explicit failed-bound witness.*

Proof. Each bound is a decidable rational comparison. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CompactSubStrip empty unit obstruction decidable`

Theorem 72.8 (Boundary excluded)**.** *For every complex history s with* $\mathsf{hsame}(\sigma, \mathsf{e})$ *or* $\mathsf{hsame}(\sigma, \mathsf{E1}(\mathsf{e}))$, $\neg\mathsf{InCritStrip}(s)$. *Both edges $\Re(s) = 0$ and $\Re(s) = 1$ are explicitly excluded by the strict-inequality witnesses.*

Proof. Direct: the strict-prefix relations require non-equality witnesses. □

Lean4 ✓ `BEDC.Derived.CritStripUp.InCritStrip boundary excluded`

Theorem 72.9 (Critical strip open interval unary boundary exclusion)**.** *If σ lies in the strict unary open interval between* e *and* $\mathsf{E1}(\mathsf{e})$, *then σ is unary and is internally same to neither boundary.*

Proof. The lower strict-prefix witness is a continuation from e to σ by a nonempty unary tail; the empty left-unit law transports that tail's unary structure to σ. If $\sigma\mathsf{hsame}\mathsf{e}$, the lower strict-prefix witness would have empty right endpoint, contradicting strictness. If $\sigma\mathsf{hsame}\mathsf{E1}(\mathsf{e})$, the upper strict-prefix witness becomes a strict self-prefix at $\mathsf{E1}(\mathsf{e})$, contradicting asymmetry. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripOpenInterval unary boundary exclusion`

Theorem 72.10 (Critical strip empty-to-unit open interval absurdity)**.** *There is no unary history strictly between the empty endpoint and the one-step unit endpoint:*

$$\mathsf{NatUnaryStrictPrefix}(\mathsf{e}, \sigma) \wedge \mathsf{NatUnaryStrictPrefix}(\sigma, \mathsf{E1}(\mathsf{e})) \Longrightarrow \bot.$$

Proof. The lower strict-prefix witness transports unary structure from e to σ. Split the unary history σ. If it is empty, the lower strict-prefix witness has empty right endpoint, impossible by strictness. If it is one-headed, the upper strict-prefix witness descends through the one-head constructor to a strict prefix into e, again impossible by strictness. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripOpenInterval empty unit absurd`

Definition 72.11 (Critical strip complex carrier)**.** *A complex strip history s with rational components σ, τ is carried when σ and τ are rational histories,* $\mathsf{Cont}(\sigma, \tau, s)$ *assembles the complex endpoint, and σ lies in the strict critical strip.*

Lean4 (definition) `BEDC.Derived.CritStripUp.CritStripComplexCarrier`

Theorem 72.12 (Critical strip complex carrier nonemptiness)**.** *If s is carried by the critical-strip complex carrier, then* $s\mathsf{hsamee}$ *is impossible.*

Proof. Transport the stored continuation endpoint along $s\mathsf{hsamee}$. The empty result inversion for Cont identifies both rational components with e, contradicting the nonempty denominator witness contained in the real rational carrier. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripComplexCarrier not empty`

Theorem 72.13 (Critical strip component boundary exclusion)**.** *If* $\mathsf{CritStripComplexCarrier}(s, \sigma, \tau)$*, then σ is not* hsame*-equal to either boundary* e *or* $\mathsf{E1}(\mathsf{e})$*, τ is not* hsame*-equal to* e*, and s is not* hsame*-equal to* e*.*

Proof. The strip field supplies the two boundary exclusions for σ. The rational carrier for τ contains a positive unary denominator, excluding the empty history. The complex endpoint exclusion is the nonemptiness theorem for the assembled carrier. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripComplexCarrier component boundary exclusion`

Theorem 72.14 (Critical strip complex carrier is empty)**.** *If* $\mathsf{CritStripComplexCarrier}(s, \sigma, \tau)$*, then* $\bot$*.*

Proof. Unpack the carrier fields. Its strip field gives the strict unary interval witnesses for σ between e and $\mathsf{E1}(\mathsf{e})$.

Apply Definition 72.10 to those two strict-prefix witnesses. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripComplexCarrier strict interval absurd`

72.3 Closure laws

Lemma 72.15 (Strip respects hsame)**.** *If* $\mathsf{InCritStrip}(\sigma)$ *and* $\sigma\mathsf{hsame}\sigma'$*, then* $\mathsf{InCritStrip}(\sigma')$*, and σ' is internally same to neither boundary* e *nor* $\mathsf{E1}(\mathsf{e})$*.*

Proof. Transport the lower and upper strict-prefix witnesses along hsame, then apply boundary exclusion at the transported strip point. □

Lean4 ✓ `BEDC.Derived.CritStripUp.InCritStrip hsame transport boundary exclusion`

Theorem 72.16 (Critical strip complex carrier respects hsame)**.** *If* $s\mathsf{hsame}s'$*,* $\sigma\mathsf{hsame}\sigma'$*,* $\tau\mathsf{hsame}\tau'$*, and* $\mathsf{CritStripComplexCarrier}(s, \sigma, \tau)$*, then* $\mathsf{CritStripComplexCarrier}(s', \sigma', \tau')$*, the continuation* $\mathsf{Cont}(\sigma', \tau', s')$ *holds, and the strict interval obstruction gives $\bot$.*

Proof. Transport the two rational carrier fields along the component equalities, transport the strict-strip field along $\sigma\mathsf{hsame}\sigma'$, and transport the stored continuation ledger along all three endpoint equalities. These fields rebuild the carried complex strip package at the primed endpoint, where the strict interval obstruction applies. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripComplexCarrier hsame transport`

72.4 The certificate

Definition 72.17 (Critical strip source specification)**.** *The source specification on complex history s carries the constraint* InCritStrip(s) *or its complement at the boundary.*

Lean4 (definition) `BEDC.Derived.CritStripUp.CritStripSourceSpec`

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripSourceSpec empty boundary`

Definition 72.18 (Critical strip pattern specification)**.** *The pattern is the conjunction of the two strict-prefix inequalities on the rational representation.*

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripTasteGate single carrier alignment`

Definition 72.19 (Critical strip classifier specification)**.** *Two strip-membership witnesses are classified together when they agree under* hsame *on both real and imaginary components.*

Lean4 (definition) `BEDC.Derived.CritStripUp.CritStripClassifierSpec`

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripClassifierSpec component transport`

Definition 72.20 (Critical strip stability certificate)**.** *The stability certificate consists of* hsame-*respect of* InCritStrip*,* OnCritLine*, and* CompactSubStrip*; transport of all three under* hsame *on each component.*

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripStabilityCert fields`

Definition 72.21 (Critical strip ledger policy)**.** *The ledger policy records: (i) decidability of strip and critical-line membership, (ii) decidability of compact sub-strip predicate, (iii) boundary exclusion witnesses, (iv) closure under* hsame *on inputs.*

Lean4 (definition) `BEDC.Derived.CritStripUp.CritStripLedgerPolicy`

Theorem 72.22 (Concrete $\mathsf{CritStrip}^{\uparrow}$ obstruction-boundary certificate)**.** *The five fields above form the finite-unary obstruction-boundary certificate over* Hist *for the critical-strip interface* $\mathsf{CritStrip}^{\uparrow}$*. The certificate records the strict-prefix boundary laws and the emptiness of the finite unary complex carrier; it does not certify a nonempty analytic open strip.*

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9) after the strip field is read as the finite-unary boundary object. The empty-carrier theorem above supplies the obstruction side of the certificate. The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Lean4 ✓ `BEDC.Derived.CritStripUp.crit strip name certificate`

Theorem 72.23 (Critical strip empty boundary semantic certificate)**.** *The boundary carrier*

$$h \mapsto h\mathsf{hsamee} \wedge \neg\mathsf{InCritStrip}(h)$$

forms a semantic naming certificate with the same source, pattern, and ledger predicate, and with hsame *as classifier.*

Proof. The carrier is inhabited by e. Boundary exclusion supplies the non-membership field, and hsame reflexivity, symmetry, and transitivity give the classifier laws. Carrier transport composes the source classifier with the stored equality to e, then applies boundary exclusion at the transported history. □

Lean4 ✓ `BEDC.Derived.CritStripUp.CritStripEmptyBoundary semanticNameCert`

Theorem 72.24 (Semantic CritStrip$^{\uparrow}$ obstruction-boundary certificate)**.** *The concrete obstruction-boundary certificate of Definition 72.22 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9) for the finite-unary* CritStrip$^{\uparrow}$ *boundary.*

Sketch. Direct verification of the four NameCert field types using the boundary constructions and the empty-carrier theorem of this chapter. □

Lean4 ✓ `BEDC.Derived.CritStripUp.crit strip semantic name certificate`

Closure status for NameCert$_{\text{CritStrip}^{\uparrow}}$**.**

Constructive story: A CritStrip$^{\uparrow}$ packet is generated from finite BHist rows for complex-history inputs, rational real-component readbacks, strict unary-prefix witnesses for $0 < \sigma < 1$, critical-line classifier rows, compact sub-strip data, and the obstruction boundary. The displayed unary inequalities are checked by finite prefix procedures and replayed by Cont through the complex and rational readback rows. hsame transports the same real-component and boundary witnesses, while Pkg records the complex, rational, and zeta-continuation dependencies. In this finite-unary presentation the open strip surface is an explicit boundary object rather than a host analytic domain.

Theory closure: TheoryGate(NameCert$_{\text{CritStrip}^{\uparrow}}$, seed)

Scope closed: This chapter binds critical-strip membership, critical-line membership, compact sub-strip predicates, boundary witnesses, empty-carrier behavior, and semantic NameCert surface for CritStrip$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{CritStrip}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a nonempty analytic critical strip, host complex-domain equality, zeta zero localization, RH content, or external topology of the strip.

Upgrade path: Obligation closure requires Lean encodings of the strip predicates, decidability rows, boundary witnesses, empty-carrier theorem, and semantic NameCert.

Chapter 73

A Concrete Naming Certificate for $\mathsf{ZetaZeros}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the predicate "s is a zero of ζ" without yet constructing any non-trivial zeros. The trivial zeros at negative even integers $-2, -4, -6, \ldots$ are constructed explicitly; the non-trivial zero predicate is characterized as elements of the critical strip whose ζ-value (under the analytic continuation $\mathsf{ZetaCont}^{\uparrow}$, chapter 9342) is the zero complex history. The chapter explicitly notes that whether non-trivial zeros all lie on the critical line is the content of the Riemann Hypothesis (chapter 9239); this chapter only sets up the predicate, the classifier, and the closure laws compatible with conjugate symmetry. Builds on $\mathsf{ZetaBasic}^{\uparrow}$ (chapter 71), $\mathsf{ZetaCont}^{\uparrow}$ (chapter 9342), and $\mathsf{CritStrip}^{\uparrow}$ (chapter 72). The construction is fully constructive at the predicate level and uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

73.1 ζ-evaluation and zero predicate

Definition 73.1 (ζ-value evaluation)**.** *For complex history s with $\neg\mathsf{hsame}(s, \mathsf{E1}(\mathsf{e}))$ (i.e. $s \neq 1$), the predicate $\mathsf{ZetaVal}(s, z)$ holds when z is the unique complex history obtained by evaluating the analytically continued ζ at s (cite chapter 9342).*

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaVal`

Lemma 73.2 (ζ-value well-defined)**.** *For every s with $s \neq 1$, $\mathsf{ZetaVal}(s, z)$ determines a unique complex history z up to* hsame *on both real components.*

Proof. Direct uniqueness from the analytic continuation operation of $\mathsf{ZetaCont}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaVal well defined`

Definition 73.3 (Zero predicate)**.** *The predicate $\mathsf{ZetaZero}(s)$ holds when there exists z with $\mathsf{ZetaVal}(s, z)$ and* hsame *holds on both real components of z and the zero complex history.*

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaZero`

Theorem 73.4 (Zero complex history carrier)**.** *The complex zero history represented as the product of two unit rational denominators is carried by $\mathsf{Complex}^{\uparrow}$.*

Proof. Each unit denominator satisfies the positive-denominator predicate because its tail is e. The product history is then carried by the product carrier for the two rational components. □

Lean4 ✓ BEDC.Derived.ZetaZerosUp.ZetaZeroComplexHistory carrier

Remark 73.5 (Decidability gap)**.** *The predicate* $\mathsf{ZetaVal}(s, z)$ *for fixed* s *is decidable to within any unary precision* k*, since the analytic continuation is computable to any precision. However, deciding* $\mathsf{ZetaZero}(s)$ *requires comparing the value to the exact zero complex history, which is not decidable from a finite-precision approximation in general — this is the constructive obstruction at the heart of why RH is open. The predicate is *defined* constructively, but its *exhaustive enumeration* over the critical strip is not.*

73.2 Trivial zeros

Theorem 73.6 (Trivial zero witness)**.** *For every unary* $n \geq \mathsf{E1}(\mathsf{e})$*, the negative even integer* $s = -2n$ *(encoded as a complex history with rational real part* $-2n$ *and imaginary part empty) satisfies* $\mathsf{ZetaZero}(s)$*. The witness comes from the functional equation* $\zeta(s) = 2^s \pi^{s-1} \sin(\pi s/2)\Gamma(1-s)\zeta(1-s)$*, where* $\sin(\pi s/2) = 0$ *at even* $s \leq -2$ *(chapter 9342).*

Sketch. At $s = -2n$ with $n \geq 1$, $\sin(\pi s/2) = \sin(-n\pi) = 0$ constructively (rational multiple of π with even integer multiplier). The other functional-equation factors are non-zero and finite at $s = -2n$ (Gamma function value at a positive integer). Hence $\zeta(-2n) = 0$ constructively. □

73.3 Non-trivial zeros

Definition 73.7 (Non-trivial zero predicate)**.** *The predicate* $\mathsf{NontrivialZero}(s)$ *holds when both* $\mathsf{ZetaZero}(s)$ *and* $\mathsf{InCritStrip}(s)$ *(chapter 72).*

Lean4 (definition) BEDC.Derived.ZetaZerosUp.NontrivialZero

Theorem 73.8 (Non-trivial zeros lie in the strip)**.** *For every* s *with* $\mathsf{NontrivialZero}(s)$*,* $\mathsf{InCritStrip}(s)$ *holds by definition. Conversely, the non-vanishing of* ζ *for* $\Re(s) \geq 1$ *(chapter 71) and the trivial-zero classification at* $\Re(s) \leq 0$ *bracket the location of any non-trivial zero strictly inside* $0 < \Re(s) < 1$*.*

Sketch. ζ's non-vanishing on $\Re(s) > 1$ gives the right boundary; the functional equation reduces $\Re(s) \leq 0$ to $\Re(s) \geq 1$; the explicit trivial-zero witnesses cover the negative even integers exactly. □

Lean4 ✓ BEDC.Derived.ZetaZerosUp.NontrivialZeroWitness continuation contradicts zero source
Lean4 ✓ BEDC.Derived.ZetaZerosUp.NontrivialZero in crit strip

73.4 Closure laws

Theorem 73.9 (Conjugate symmetry of zeros)**.** *If* $\mathsf{ZetaZero}(s)$*, then* $\mathsf{ZetaZero}(\overline{s})$*, where* $\overline{s}$ *is the complex history obtained by negating the imaginary component.*

Sketch. ζ takes complex conjugates to complex conjugates, since the Dirichlet sum and analytic continuation both respect conjugation; if $\zeta(s) = 0$, then $\overline{\zeta(s)} = \zeta(\overline{s}) = 0$. □

Definition 73.10 (Zeta boundary non-vanishing witness)**.** *A zeta boundary non-vanishing witness consists of a complex history* s*, a displayed real component* σ*, component-display evidence linking*

σ to the real component of s, a boundary row $\mathsf{hsame}(\sigma, \mathsf{E1}(\mathsf{e}))$*, and a BEDC witness whose exported field has type*

$$\mathsf{ZetaZero}(s) \to \mathsf{False}.$$

This datum is the certificate slot for excluding zeros on the displayed boundary line $\Re(s) = 1$; the chapter records the slot without deriving the analytic non-vanishing argument from the current kernel.

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaBoundaryNonvanishingWitness`

73.5 The certificate

Definition 73.11 (Zeta zero source specification)**.** *The source specification carries the constraint* $\mathsf{ZetaZero}(s)$ *plus an optional* $\mathsf{InCritStrip}(s)$ *flag distinguishing trivial from non-trivial zeros.*

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec`

Definition 73.12 (Zeta zero pattern specification)**.** *The pattern is the joint predicate* $\mathsf{ZetaVal}(s, z) \wedge \mathsf{hsame}(z, \mathsf{e}_{\mathsf{Complex}^\uparrow})$ *on complex-history pairs.*

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaZeroPatternSpec`

Definition 73.13 (Zeta zero classifier specification)**.** *Two zero witnesses at the same s are classified together when their underlying* $\mathsf{ZetaVal}$ *witnesses agree under* hsame *on the zero complex history.*

Lean4 (definition) `BEDC.Derived.ZetaZerosUp.ZetaZeroClassifierSpec`
Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroClassifierSpec zero anchor pair`

Theorem 73.14 (Zeta zero pattern classifies the zero result)**.** *Every zeta-zero pattern witness identifies its result component with the explicit zero complex history, so that result is classified with the zero history by the* $\mathsf{Complex}^\uparrow$ *classifier.*

Proof. The pattern supplies the hsame comparison from z to the zero complex history. The explicit zero complex history carries $\mathsf{Complex}^\uparrow$, and hsame-transport gives the carrier for z. These fields are exactly the $\mathsf{ComplexClassifier}$ datum. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroPatternSpec zero result classifier`

Theorem 73.15 (Zeta zero pattern source-result classifier)**.** *Every zeta-zero pattern witness classifies its source and result components together, and both components remain anchored at the explicit zero complex history*

$$\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})).$$

Proof. The source side of the pattern supplies the $\mathsf{Complex}^\uparrow$ carrier for s and the comparison from s to the zero complex history. The result-side classifier supplies the $\mathsf{Complex}^\uparrow$ carrier for z, while the pattern itself supplies the comparison from z to the same zero history. Composing the source anchor with the symmetric result anchor gives $s\mathsf{hsame}z$, which is the classifier relation, and the two anchors are retained. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroPatternSpec source result classifier`

Theorem 73.16 (Zeta zero source classifies against zero)**.** *Every source history carried by* $\mathsf{Complex}^{\uparrow}$ *and* hsame*-identified with the zero complex history is classified with that zero history by the* $\mathsf{Complex}^{\uparrow}$ *classifier.*

Proof. The source predicate supplies the left carrier and the hsame comparison. The zero-history carrier is Definition 73.4; these three fields are exactly the ComplexClassifier record. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec zero classifier`

Theorem 73.17 (Zeta zero source hsame transport classifier)**.** *If* $s\mathsf{hsame}t$ *and* s *satisfies the zeta-zero source specification, then* t *satisfies the same specification and is classified with the explicit zero complex history*

$$\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})).$$

Proof. The source specification supplies a $\mathsf{Complex}^{\uparrow}$ carrier for s and an hsame equality from s to the zero complex history. Transport the product carrier along $s\mathsf{hsame}t$ and compose equality symmetrically to obtain $t\mathsf{hsameappend}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$. These two fields rebuild the source specification for t, and the zero-classifier theorem classifies it against the explicit zero history. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec hsame transport classifier`

Theorem 73.18 (Zeta zero pattern hsame transport classifier)**.** *If* $s\mathsf{hsame}t$*,* $z\mathsf{hsame}w$*, and* (s, z) *satisfies the zeta-zero pattern specification, then* (t, w) *satisfies the same pattern and* w *is classified with the explicit zero complex history.*

Proof. Transport the source side along $s\mathsf{hsame}t$ by the source transport theorem. The result-side comparison $w\mathsf{hsameappend}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}))$ is obtained by composing the symmetry of $z\mathsf{hsame}w$ with the zero comparison contained in the original pattern witness. These two fields rebuild the pattern for (t, w), and the zero-result classifier theorem supplies the displayed $\mathsf{Complex}^{\uparrow}$ classifier. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroPatternSpec hsame transport classifier`

Theorem 73.19 (Zeta zero source zero anchor uniqueness)**.** *If two histories satisfy the zeta-zero source specification, then they are* hsame*-equal to each other and each is* hsame*-equal to the displayed zero complex history*

$$\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})).$$

Proof. Each source specification contains its explicit comparison with the displayed zero complex history. Compose the first comparison with the symmetric second comparison to obtain $s\mathsf{hsame}t$, and retain the two displayed anchor rows. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec zero anchor unique`

Theorem 73.20 (Zeta zero source pair classifier)**.** *If two histories satisfy the zeta-zero source specification, then the* $\mathsf{Complex}^{\uparrow}$ *classifier relates them, and both histories remain anchored at the displayed zero complex history*

$$\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})).$$

Proof. Each source specification supplies a $\mathsf{Complex}^{\uparrow}$ carrier and an hsame comparison with the displayed zero complex history. Composing one comparison with the symmetric other comparison gives the classifier equality between the two histories, while the two anchor comparisons are retained as the remaining fields. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec pair classifier`

Theorem 73.21 (Zeta zero pattern pair classifier)**.** *If (s, z) and (t, w) satisfy the zeta-zero pattern specification, then* $\mathsf{Complex}^{\uparrow}$ *classifies s with t and z with w, and all four histories remain anchored at the displayed zero complex history*

$$\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})).$$

Proof. Apply the source pair classifier to the two source specifications. Apply the zero-result classifier to each result component, then compose one classifier with the symmetry of the other to classify z with w. The four displayed anchor comparisons are the source and result comparisons stored in the two pattern witnesses. □

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroPatternSpec pair classifier`

Definition 73.22 (Zeta zero stability certificate)**.** *The stability certificate consists of* hsame*-respect of* ZetaZero*, conjugate symmetry of zeros, and supplied boundary exclusion data: zeta boundary non-vanishing witnesses on $\Re(s) = 1$, trivial-zero witnesses at negative even integers, and explicit right-half-plane exclusion witnesses for the chosen source interface.*

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaBoundaryNonvanishingWitness sigma nonempty`

Definition 73.23 (Zeta zero ledger policy)**.** *The ledger policy records: (i) trivial zeros at $s = -2n$ for unary $n \geq 1$, (ii) boundary and right-half-plane exclusion only when the corresponding witness data is supplied, (iii) conjugate symmetry, (iv) the explicit zero classifier under* hsame*. The ledger explicitly notes that exhaustive enumeration of non-trivial zeros is not constructively decidable from finite-precision data — that gap is what RH (chapter 9239) addresses.*

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroLedgerPolicy boundary nonvanishing readback`

Theorem 73.24 (Concrete $\mathsf{ZetaZeros}^{\uparrow}$ naming certificate)**.** *The five fields above form a naming certificate over* Hist *for the ζ-zero interface* $\mathsf{ZetaZeros}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeroSourceSpec name certificate`

Sketch. The five fields satisfy the closure laws of the naming-certificate framework (chapter 9). The certificate uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate. □

Theorem 73.25 (Semantic $\mathsf{ZetaZeros}^{\uparrow}$ naming certificate)**.** *The concrete certificate of Definition 73.24 satisfies the semantic naming-certificate schema of* NameCert *(chapter 9).*

Lean4 ✓ `BEDC.Derived.ZetaZerosUp.ZetaZeros semanticNameCert`

Sketch. Direct verification of the four NameCert field types using the constructions of this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZetaZeros}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZetaZeros}^{\uparrow}$ packet is generated from finite BHist rows for a complex-history point, zeta-value readback, zero classifier, critical-strip or continuation source rows, transports, continuations, provenance, and local naming. A zero read is obtained by replaying the displayed zeta row and zero witness through Cont, not by selecting a host complex zero or quotient zero-set element. Componentwise hsame transports point, value, zero, strip, and continuation rows, while Pkg records the zeta-basic, continuation, critical-strip, and complex dependencies. The certificate names the finite zero surface and keeps analytic zero theory outside the packet.

Theory closure: TheoryGate(NameCert$_{\mathsf{ZetaZeros}^{\uparrow}}$, seed)

Scope closed: This chapter binds the zeta-zero predicate, zero readback rows, source-boundary routes, classifier stability, and semantic NameCert surface for ZetaZeros$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZetaZeros}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a complete zero set, RH, analytic zero search, quotient equality of zeros, host complex equality, or meromorphic continuation beyond displayed sources.

Upgrade path: Obligation closure requires Lean encodings of the zero carrier, zeta readback route, source-boundary rows, classifier stability, and semantic NameCert theorem.

Chapter 74

A Conditional Certificate Boundary for RealAnalytic$^{\uparrow}$

This chapter records the proof obligations for the constructive transcendental package RealAnalytic$^{\uparrow}$ over Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat, and Real$^{\uparrow}$ without asserting a completed NameCert(RealAnalytic$^{\uparrow}$). The package is organized around five interrelated objects on Real$^{\uparrow}$-history representations: the constant π, the exponential exp, the logarithm log on positive arguments, and the trigonometric pair sin / cos. The construction boundary sits above Rat$^{\uparrow}$ (chapter 37) for the rational base, StreamName$^{\uparrow}$ for explicit unary streams, RegSeqRat for regular Cauchy data, and Real$^{\uparrow}$ (chapter 38) for the eventual real-history carrier. Every series below is given by an explicit unary-history term, every modulus is a total function on the unary index domain, and every passage from regular rational data to a real-history value requires a supplied limit witness from the Phase Real and Completion obligations.

74.1 Constructive π via Leibniz acceleration

Definition 74.1 (Leibniz partial sum)**.** *For unary index n, the Leibniz partial sum predicate* LeibPart(n, S_n) *holds when S_n is the explicit* Rat$^{\uparrow}$*-history obtained by summing the first n terms of $\sum_{k\geq 0}(-1)^k/(2k+1)$ via unary recursion using* Rat$^{\uparrow}$ *addition and signed-fraction subtraction.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticLeibnizPartialSum`
Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticLeibnizPartSum`

Lemma 74.2 (Leibniz alternating-series Cauchy modulus)**.** *The sequence of partial sums S_n is a* RegSeqRat*-Cauchy sequence with explicit modulus $N(k) := 2^{k+1}$: for all unary $m, n \geq N(k)$, $|4S_m - 4S_n| < 2^{-k}$.*

Proof. Pair adjacent terms after index $N(k)$; each pair has magnitude bounded by the next omitted term $1/(2N(k)+1)$. The alternating bound gives $|4S_m - 4S_n| \leq 4/(2N(k)+1) < 2^{-k}$ for the chosen $N(k)$. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticLeibnizPartSum index result unary`

Definition 74.3 (Constructive π)**.** Pi *is a* Real$^{\uparrow}$*-history candidate only together with a supplied limit witness from the Phase Real and Completion obligations, applied to the* RegSeqRat *Cauchy data of Definition 74.2 for the rescaled sequence $(4S_n)_n$. The local data of this chapter is the* RegSeqRat *name and its explicit modulus inherited from the alternating-series bound.*

Lean4 (definition) BEDC.Derived.RealAnalyticUp.RealAnalyticPiBoundary
Lean4 ✓ BEDC.Derived.RealAnalyticUp.RealAnalyticPiBoundary leibniz index result unary
Lean4 (definition) BEDC.Derived.RealAnalyticUp.RealAnalyticPiLocalData
Lean4 (definition) BEDC.Derived.RealAnalyticUp.RealAnalyticPiCandidate
Lean4 (definition) BEDC.Derived.RealAnalyticUp.RealAnalyticPi

Theorem 74.4 (Series variant equivalence boundary)**.** *Let p_1 and p_2 be two explicit π stream presentations, such as the Leibniz rescaling and a faster Machin or Chudnovsky presentation. A conclusion* $\mathsf{RealClassifier}(r_1, r_2)$ *is available for their* $\mathsf{Real}^{\uparrow}$ *histories only under supplied rows: both p_1 and p_2 satisfy the relevant* RegSeqRat *carrier obligations, a* RegSeqRat *classifier witness relates the two stream names, both streams have supplied* $\mathsf{Real}^{\uparrow}$ *limit witnesses with targets r_1 and r_2, and the supplied* $\mathsf{Real}^{\uparrow}$ *limit-uniqueness row transports the stream classifier to the real-history classifier.*

Lean4 ✓ BEDC.Derived.RealAnalyticUp.RealAnalyticRealCompletionDependencySurface local result unary

Proof. Use the supplied RegSeqRat classifier witness first. The two carrier-obligation rows make this witness a relation between legitimate regular rational stream names, so it is eligible for transport by the real-completion interface.

Apply the two supplied $\mathsf{Real}^{\uparrow}$ limit witnesses next. They place p_1 and p_2 in the real-history carrier as r_1 and r_2, without adding any independent equality claim between the analytic presentations.

Finally invoke the supplied limit-uniqueness row. Its transport clause takes the RegSeqRat classifier witness between the source streams and the two limit witnesses, and returns the displayed conclusion $\mathsf{RealClassifier}(r_1, r_2)$. □

Lean4 ✓ BEDC.Derived.RealAnalyticUp.RealAnalyticLeibnizPartSum pointwise hsame deterministic

Theorem 74.5 (Real-analytic supplied limit consumption boundary)**.** *Let P be any local analytic stream package used in this chapter: the Leibniz* Pi *stream, an exponential partial-sum stream, a logarithm bisection stream, or a sine or cosine partial-sum stream. A displayed* $\mathsf{Real}^{\uparrow}$ *output r for P is consumable only through the following supplied rows:*

1. *a* RegSeqRat *carrier witness for the underlying* BHist *stream name and its unary modulus;*
2. *a* $\mathsf{Real}^{\uparrow}$ *limit witness naming r as the target history of that stream;*
3. *the Phase Real and Completion limit-transport or limit-uniqueness row that connects the supplied stream witness to the public* $\mathsf{Real}^{\uparrow}$ *carrier and classifier surface.*

If two such packages P and Q are related by a supplied RegSeqRat *classifier witness and their supplied targets are r and s, then the only conclusion licensed by those rows is* $\mathsf{RealClassifier}(r, s)$*. The rows do not supply* $\mathsf{Real}^{\uparrow}$ *completeness,* $\mathsf{Real}^{\uparrow}$ *arithmetic,* $\mathsf{Real}^{\uparrow}$ *order, metric completeness, or a standard bridge.*

Lean4 ✓ BEDC.Derived.RealAnalyticUp.RealAnalyticSuppliedLimitConsumption boundary

Proof. The transport pattern is the one isolated in Definition 74.4. The RegSeqRat classifier is applied at the stream-name level, the two supplied limit witnesses name the target histories, and the supplied Phase Real and Completion uniqueness row returns only the displayed real-history classifier.

For the stream-side inputs, use the local obligation package of Definition 74.15. Its unary and pointwise transport clauses cover the finite partial-sum witnesses used by the Leibniz, exponential,

logarithm-bisection, sine, and cosine packages, but they do not create a RegSeqRat carrier row or a $\mathsf{Real}^{\uparrow}$ limit row.

The public rows available after a limit witness are exactly the rows exposed by Definition 38.286: carrier, classifier, stability, transport, finite observation, ledger, and seal data for the scoped $\mathsf{Real}^{\uparrow}$ surface. That public boundary explicitly excludes completion theorems, arithmetic and order closure, metric completeness, and standard-bridge data, so the stated classifier is the maximal conclusion licensed by the supplied rows. □

74.2 Constructive exponential

Definition 74.6 (Exponential partial sum)**.** *For* $\mathsf{Real}^{\uparrow}$*-history* x *and unary index* n*, the predicate* $\mathsf{ExpPart}(x, n, S_n)$ *holds when* S_n *is the partial sum* $\sum_{k=0}^{n} x^k/k!$ *computed by unary recursion: powers* x^k *via iterated* $\mathsf{Real}^{\uparrow}$ *multiplication, factorials* $k!$ *via the unary factorial of* $\mathsf{Prime}^{\uparrow}$ *(chapter 64), and division as* $\mathsf{Real}^{\uparrow}$*-multiplicative inverse against the positive unary denominator.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticExpPart`

Lemma 74.7 (Real-analytic complex partial-sum indices are unary)**.** *For the real-analytic series interface, every closed complex partial-sum witness has unary index.*

Proof. Induct on the closed complex partial-sum witness. The zero constructor has the empty index, and the successor constructor applies the unary successor closure to the predecessor index. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexPartSum index unary`

Theorem 74.8 (Real-analytic complex partial-sum results are unary)**.** *For the real-analytic series interface, if the chosen zero history is unary and each term history is unary on unary indices, then every closed complex partial-sum witness has unary result.*

Proof. Induct on the closed complex partial-sum witness. The base is the supplied unary zero history. In the successor case, an inner induction on the predecessor partial-sum witness supplies unaryness of the predecessor index, the term hypothesis supplies unaryness of the next term, and unary continuation closure gives the successor result. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticLeibnizPartSum successor result deterministic`

Theorem 74.9 (Real-analytic complex partial-sum indices and results are unary)**.** *For the real-analytic series interface, if the chosen zero history is unary and each term history is unary on unary indices, then every closed complex partial-sum witness has both unary index and unary result.*

Proof. Induct on the closed complex partial-sum witness. The base has empty unary index and the supplied unary zero result. In the successor case, the induction hypothesis gives unaryness of the predecessor index and result; closure under E1 gives the successor index, the term hypothesis gives the next unary term, and unary continuation closure gives the successor result. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexPartSum index result unary`

Theorem 74.10 (Real-analytic complex absolute partial-sum indices and results are unary)**.** *For the real-analytic absolute-series interface, if the chosen zero history is unary and each modulus history is unary on unary indices, then every closed absolute complex partial-sum witness has both unary index and unary result.*

Proof. Induct on the closed absolute partial-sum witness. The zero case is the empty unary index with the supplied unary zero result. In the successor case, the induction hypothesis supplies the predecessor index and result; unary E1-closure supplies the successor index, the modulus hypothesis supplies the unary summand, and unary continuation closure supplies the successor result. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexAbsPartSum index result unary`

Theorem 74.11 (Real-analytic pointwise complex partial-sum target unaryness)**.** *For two complex term families c, d : BHist $\to$ BHist, suppose the chosen zero histories are* hsame*, the source zero history is unary, each $c(m)$ is unary on unary indices, and $c(m)$hsame$d(m)$ at every unary index. If n is unary and closed partial-sum witnesses compute results S for $(zero, c)$ and T for $(zero', d)$ at n, then T is unary.*

Proof. The source partial-sum unary theorem gives unaryness of S. Pointwise hsame determinacy for complex partial sums identifies S and T under hsame, using the zero comparison, the pointwise term comparison, and the unary index witness. Unary history closure under hsame gives unaryness of T. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexPartSum pointwise result unary transport`

Theorem 74.12 (Closed pointwise complex partial-sum unary transport)**.** *For two complex term families c, d : BHist $\to$ BHist, suppose the chosen zero histories zero and zero$'$ are* hsame*, the source zero history zero is unary, each $c(m)$ is unary whenever m is unary, and $c(m)$hsame$d(m)$ for every unary index m. If closed partial-sum witnesses compute S for $(zero, c)$ and T for $(zero', d)$ at the same index n, then* UnaryHistory(n) *and* UnaryHistory(T)*.*

Proof. Apply the complex partial-sum index unary theorem to the source witness. This turns the closed source computation into the unary index witness UnaryHistory(n).

With this derived index witness, apply the complex partial-sum index/result unary theorem to the same source witness. The hypotheses on *zero* and c give UnaryHistory(S).

Pointwise hsame determinacy for complex partial sums identifies S and T under hsame, using the comparison of the zero histories and the pointwise comparison $c(m)$hsame$d(m)$ at unary indices. Unary history transport along hsame carries UnaryHistory(S) to UnaryHistory(T). □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexPartSum closed pointwise index result unary transport`

Theorem 74.13 (Real-analytic pointwise absolute partial-sum target unaryness)**.** *For two modulus families* modulus, modulus$'$: BHist $\to$ BHist*, suppose the chosen zero histories are* hsame*, the source zero history is unary, each source modulus value is unary on unary indices, and the two modulus families are pointwise* hsame *on unary indices. If n is unary and closed absolute partial-sum witnesses compute results M and T at n, then T is unary.*

Proof. The source absolute partial-sum unary theorem gives unaryness of M. Pointwise hsame determinacy for absolute partial sums identifies M and T under hsame, using the zero comparison, the pointwise modulus comparison, and the unary index witness. Unary history closure under hsame gives unaryness of T. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexAbsPartSum pointwise result unary transport`

Theorem 74.14 (Closed pointwise absolute partial-sum unary transport)**.** *For two modulus families $m, m' : \mathsf{BHist} \to \mathsf{BHist}$, suppose the chosen zero histories zero and zero$'$ are* hsame*, the source zero history zero is unary, each $m(i)$ is unary whenever i is unary, and $m(i)\mathsf{hsame}m'(i)$ for every unary index i. If closed absolute partial-sum witnesses compute M for $(zero, m)$ and T for $(zero', m')$ at the same index n, then* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{UnaryHistory}(T)$*.*

Proof. Apply the absolute partial-sum index/result unary theorem to the source witness. The source zero unaryness and the unaryness of $m(i)$ on unary indices give both $\mathsf{UnaryHistory}(n)$ and $\mathsf{UnaryHistory}(M)$.

Using the derived unary index witness, apply the pointwise absolute partial-sum target unary transport theorem to the two closed computations. The zero comparison, source modulus unaryness, and pointwise comparison $m(i)\mathsf{hsame}m'(i)$ at unary indices give $\mathsf{UnaryHistory}(T)$. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticComplexAbsPartSum closed pointwise index result unary transport`

Theorem 74.15 (Real-analytic local stream unary package)**.** *Let $zero, zero' : \mathsf{BHist}$ be chosen zero histories. Let $c, d : \mathsf{BHist} \to \mathsf{BHist}$ be complex term families, and let $m, m' : \mathsf{BHist} \to \mathsf{BHist}$ be absolute-modulus families. Assume* $\mathsf{UnaryHistory}(zero)$*, $zero\mathsf{hsame}zero'$, each $c(i)$ is unary at unary i, each $m(i)$ is unary at unary i, and the pointwise comparisons $c(i)\mathsf{hsame}d(i)$ and $m(i)\mathsf{hsame}m'(i)$ hold at every unary index i. Then the local stream side supplies exactly the following four obligations:*

1. *every closed witness* $\mathsf{ComplexPartSum}(zero, c, n, S)$ *gives* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{UnaryHistory}(S)$*;*
2. *if n is unary and closed pointwise witnesses* $\mathsf{ComplexPartSum}(zero, c, n, S)$ *and* $\mathsf{ComplexPartSum}(zero', d, n, T)$ *are given, then* $\mathsf{UnaryHistory}(T)$*;*
3. *every closed witness* $\mathsf{ComplexAbsPartSum}(zero, m, n, M)$ *gives* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{UnaryHistory}(M)$*;*
4. *if n is unary and closed pointwise witnesses* $\mathsf{ComplexAbsPartSum}(zero, m, n, M)$ *and* $\mathsf{ComplexAbsPartSum}(zero',$ *are given, then* $\mathsf{UnaryHistory}(T)$*.*

This package contains only finite stream-side data; the $\mathsf{RegSeqRat}$ *carrier row, the* $\mathsf{Real}^{\uparrow}$ *limit witness row, and the limit-uniqueness classifier row remain the separate supplied hypotheses of Definition 74.29.*

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticLocalStream obligations package`

Proof. For the complex closed-source clause, apply Definition 74.9. Its hypotheses are exactly the unary zero history and unaryness of $c(i)$ on unary indices.

For the complex pointwise target clause, apply Definition 74.11. The hypotheses provide the comparison $zero\mathsf{hsame}zero'$, the source unary zero history, unaryness of the source term family, the pointwise comparisons $c(i)\mathsf{hsame}d(i)$, and the displayed unary index.

For the absolute closed-source clause, apply Definition 74.10. Its hypotheses are the same unary zero history and unaryness of $m(i)$ on unary indices.

For the absolute pointwise target clause, apply Definition 74.13. The hypotheses provide the zero comparison, the source unary zero history, the source modulus unaryness, the pointwise comparisons $m(i)\mathsf{hsame}m'(i)$, and the displayed unary index. These four applications do not create a $\mathsf{RegSeqRat}$ name, a $\mathsf{Real}^{\uparrow}$ limit, or a real-history classifier; those rows remain external to the finite stream package. □

Lemma 74.16 (Exponential Cauchy modulus on bounded inputs)**.** *For every* $\mathsf{Real}^{\uparrow}$*-history* x *with* $|x| \leq B$ *(explicit unary upper bound* B*), the partial sums* $\mathsf{ExpPart}(x, n, S_n)$ *form a* $\mathsf{RegSeqRat}$*-Cauchy sequence with explicit modulus* $N(k) := 2B + k$*: for all* $m, n \geq N(k)$*,* $|S_m - S_n| < 2^{-k}$*.*

Proof. For $n \geq 2B$, the ratio of successive terms $|x|/(n+1)$ is bounded by $1/2$, so the tail majorizes a geometric series with ratio $1/2$. The geometric-series modulus $2^{-(n-2B)}$ gives the explicit $N(k)$ above. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticExp local witness unary`

Definition 74.17 (Constructive exp)**.** *For a* $\mathsf{Real}^{\uparrow}$*-history* x*,* $\mathsf{Exp}(x, y)$ *holds in this chapter only relative to a supplied* $\mathsf{Real}^{\uparrow}$ *limit witness from the Phase Real and Completion obligations for the partial sums* $\mathsf{ExpPart}(x, \cdot, \cdot)$ *and the Cauchy modulus of Definition 74.16. The local witness includes the explicit input bound* B *and the modulus* N*.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticExp`

Theorem 74.18 (Exponential additivity)**.** *For* $\mathsf{Real}^{\uparrow}$*-histories* x *and* y*, assume supplied* $\mathsf{RealAlgOrder}$ *rows for* $x + y$*, products in* $\mathsf{Real}^{\uparrow}$*, powers* x^k*,* y^k*, and* $(x + y)^k$*, the unary factorial-denominator inverses, and the Cauchy-product comparison between the partial sums of* $\mathsf{ExpPart}(x + y, \cdot, \cdot)$ *and the product of the partial sums for* x *and* y*. Assume also supplied* $\mathsf{Real}^{\uparrow}$ *limit witnesses and their limit-uniqueness classifier rows for* $\mathsf{Exp}(x, e_x)$*,* $\mathsf{Exp}(y, e_y)$*, and* $\mathsf{Exp}(x + y, e_{x+y})$*. Under exactly these rows,* $\mathsf{RealClassifier}(e_{x+y}, e_x \cdot e_y)$ *holds.*

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticExp product witness unary`

Proof. The $\mathsf{RealAlgOrder}$ rows make the finite algebraic comparison meaningful: addition supplies $x + y$, multiplication supplies the product side, powers and factorial-denominator inverses supply the displayed exponential terms, and the Cauchy-product row compares the two regular rational partial-sum families.

The three supplied $\mathsf{Real}^{\uparrow}$ limit witnesses place those regular rational families in the real-history carrier. Their supplied classifier uniqueness rows identify the common limit, giving $\mathsf{RealClassifier}(e_{x+y}, e_x \cdot e_y)$. □

74.3 Constructive logarithm on positives

Definition 74.19 (Logarithm via inverse-of-exp bisection)**.** *For a positive* $\mathsf{Real}^{\uparrow}$*-history* x *(i.e. apartness witness* $x \neq 0$ *with positive sign),* $\mathsf{Log}(x, y)$ *holds in this chapter only relative to a supplied* $\mathsf{Real}^{\uparrow}$ *limit witness for the dyadic bisection sequence on unary precision* k*: at each precision, locate the dyadic-rational interval* $[y_k, y_k + 2^{-k}]$ *such that* $\exp(y_k) \leq x < \exp(y_k + 2^{-k})$ *(decidable on rational representations using Definition 74.17), and assemble the local* $\mathsf{RegSeqRat}$ *data with modulus* $N(k) = k + \lceil \log_2(1/x) \rceil$*.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticLog`

Theorem 74.20 (Logarithm is the inverse of exp on positives)**.** *For a* $\mathsf{Real}^{\uparrow}$*-history* x*, assume supplied positivity and apartness rows for* x*, supplied dyadic bisection rows locating rational intervals* $[y_k, y_k + 2^{-k}]$ *with* $\exp(y_k) \leq x < \exp(y_k + 2^{-k})$*, supplied rational-embedding rows for the dyadic endpoints and exponential comparisons, and supplied* $\mathsf{Real}^{\uparrow}$ *limit witnesses with limit-uniqueness classifier rows for* $\mathsf{Log}(x, y)$ *and* $\mathsf{Exp}(y, e_y)$*. Under these rows,* $\mathsf{RealClassifier}(e_y, x)$ *holds. Conversely, for a* $\mathsf{Real}^{\uparrow}$*-history* z*, assume the supplied* $\mathsf{Exp}(z, w)$ *limit witness, the supplied positivity*

row for w, the same bisection and rational-embedding rows applied to w, and the supplied uniqueness rows for $\mathsf{Log}(w, z')$*; then* $\mathsf{RealClassifier}(z', z)$ *holds.*

Proof. For the forward direction, the supplied positivity row permits the bisection search, and the dyadic and rational-embedding rows provide the rational comparisons used at every precision. The supplied limit witness for $\mathsf{Log}(x, y)$ records the bisection limit, while the supplied witness for $\mathsf{Exp}(y, e_y)$ records the exponential value of that limit.

The supplied uniqueness rows then identify the real-history value of $\exp(y)$ with the original positive input x. The reverse direction uses the supplied $\mathsf{Exp}(z, w)$ row as the positive input for the same bisection package, and the uniqueness row for $\mathsf{Log}(w, z')$ identifies the recovered logarithm with z. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticLogExpInverse supplied boundary unary`

Theorem 74.21 (Logarithm handoff to RealExponential)**.** *For a positive* $\mathsf{Real}^{\uparrow}$*-history x, assume the supplied positivity and apartness rows for x, the supplied dyadic bisection rows used by Definition 74.19, and the supplied rational-embedding rows for the dyadic endpoints. Assume further that every exponential comparison row used in those bisection steps is supplied by an accepted* $\mathsf{RealExponential}^{\uparrow}$ *packet whose public reads satisfy Definition 1816.3. If* $\mathsf{Log}(x, y)$ *and* $\mathsf{Exp}(y, e_y)$ *are read with the supplied* $\mathsf{Real}^{\uparrow}$ *limit witnesses and limit-uniqueness classifier rows of Definition 74.20, then the inverse read factors through* $\mathsf{RealExponential}^{\uparrow}$*,* $\mathsf{RealPowerSeries}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{CauchyRealField}^{\uparrow}$*,* $\mathsf{RealEquality}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before the classifier conclusion* $\mathsf{RealClassifier}(e_y, x)$ *is consumed.*

Proof. Project the accepted $\mathsf{RealExponential}^{\uparrow}$ packet attached to each exponential comparison row. By Definition 1816.3, the comparison can be read only after the coefficient ledger, uniform Cauchy-series modulus, Cauchy-product route, Cauchy-real-field arithmetic row, real-equality boundary, and terminal real seal have been displayed.

These routed comparison rows are exactly the exponential inputs required by the bisection construction in Definition 74.19. With the supplied positivity, apartness, rational-embedding, limit, and uniqueness rows in place, Definition 74.20 applies to the same $\mathsf{Log}(x, y)$ and $\mathsf{Exp}(y, e_y)$ reads and yields $\mathsf{RealClassifier}(e_y, x)$.

Since the only exponential comparisons available to the bisection are those projected from the accepted $\mathsf{RealExponential}^{\uparrow}$ packet, the inverse read exposes no host exp or log, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 74.22 (Logarithm of a unary natural)**.** *For positive unary natural n, assume supplied rational-embedding rows sending the unary numeral* $\mathsf{E1}^n(\mathsf{e})$ *into the positive* $\mathsf{Real}^{\uparrow}$ *carrier, supplied positivity and apartness rows for that embedded value, supplied dyadic bisection rows with the modulus determined by the unary length of n, and supplied* $\mathsf{Real}^{\uparrow}$ *limit uniqueness rows for the resulting* $\mathsf{Log}(\mathsf{E1}^n(\mathsf{e}), y_n)$ *witness. Under these rows, y_n is the constructive* $\log n$ *input used by the Dirichlet term construction (chapter 70).*

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticLog hsame transport value unary`

Proof. The rational-embedding rows place the unary numeral inside $\mathsf{Real}^{\uparrow}$ with the required positive sign data. The bisection rows then specialize Definition 74.19 to this embedded value and supply the precision modulus determined by the unary length of n.

The supplied limit witness and uniqueness rows turn the bisection stream into the real-history value y_n. No logarithm classifier conclusion is available before these rows are present. □

74.4 Constructive sine and cosine

Definition 74.23 (Sine and cosine partial sums)**.** *For* $\mathsf{Real}^{\uparrow}$*-history* x *and unary index* n*,* $\mathsf{SinPart}(x, n, S_n)$ *and* $\mathsf{CosPart}(x, n, C_n)$ *are the partial sums* $\sum_{k=0}^{n}(-1)^k x^{2k+1}/(2k+1)!$ *and* $\sum_{k=0}^{n}(-1)^k x^{2k}/(2k)!$ *respectively, computed by unary recursion as in Definition 74.6.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticTrigPart`

Lemma 74.24 (Trigonometric Cauchy modulus)**.** *For every* $\mathsf{Real}^{\uparrow}$*-history* x *with* $|x| \leq B$*, the partial sums* $\mathsf{SinPart}$ *and* $\mathsf{CosPart}$ *form* $\mathsf{RegSeqRat}$*-Cauchy sequences with explicit modulus* $N(k) := B^2 + k$*. The bound is the alternating-series majorant: each term has magnitude bounded by the previous one once* $2k(2k+1) > B^2$*.*

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticTrigPart pair index result unary`

Definition 74.25 (Constructive sin and cos)**.** $\mathsf{Sin}(x, s)$ *and* $\mathsf{Cos}(x, c)$ *hold in this chapter only relative to supplied* $\mathsf{Real}^{\uparrow}$ *limit witnesses from the Phase Real and Completion obligations for* $\mathsf{SinPart}(x, \cdot, \cdot)$ *and* $\mathsf{CosPart}(x, \cdot, \cdot)$ *respectively, using the explicit Cauchy moduli of Definition 74.24.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticSinCosCandidate`

Theorem 74.26 (Pythagorean identity)**.** *For a* $\mathsf{Real}^{\uparrow}$*-history* x*, assume supplied* $\mathsf{RealAlgOrder}$ *rows for products, squares, addition, and the constant* $\mathsf{E1}(\mathsf{e})$*, supplied product-stability rows for multiplying supplied sine and cosine limits, supplied trigonometric limit witnesses for* $\mathsf{Sin}(x, s)$ *and* $\mathsf{Cos}(x, c)$*, and supplied limit-uniqueness classifier rows comparing the regular rational partial sums for* $s^2 + c^2$ *with the constant-one witness. Under these rows,* $\mathsf{RealClassifier}(s^2 + c^2, \mathsf{E1}(\mathsf{e}))$ *holds.*

Proof. The supplied algebraic rows make $s^2 + c^2$ and the constant-one comparison available in $\mathsf{Real}^{\uparrow}$. Product stability transports the supplied sine and cosine limit witnesses through the finite products needed to form the square terms.

The regular rational comparison is the usual cancellation between the odd-power and even-power partial sums, bounded by the trigonometric Cauchy modulus of Definition 74.24. The supplied uniqueness classifier rows then identify the resulting value with $\mathsf{E1}(\mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticPythagorean local square sum unary`

Theorem 74.27 (Sine vanishes at integer multiples of π)**.** *For every signed unary integer* n*, assume supplied* $\mathsf{RealAlgOrder}$ *rows for integer scalar multiplication* $n \cdot \mathsf{Pi}$*, addition of consecutive multiples of* Pi*, products in the sine addition formula, and the zero constant* e*. Assume also a supplied constant* Pi *limit witness, supplied trigonometric limit witnesses for the required sine and cosine terms, supplied addition-formula rows for* $\sin(a + b)$ *and* $\cos(a + b)$ *at these arguments, supplied product-stability rows for the formula terms, and supplied limit-uniqueness classifier rows for the base values* $\sin(0)$ *and* $\sin(\mathsf{Pi})$*. Under these rows,* $\mathsf{Sin}(n \cdot \mathsf{Pi}, s)$ *gives* $\mathsf{RealClassifier}(s, \mathsf{e})$*. In particular, the negative signed multiples supply the zero witness used in chapter 73.*

Proof. The supplied Pi limit witness and scalar-multiplication rows make each integer multiple $n \cdot \mathsf{Pi}$ a legitimate $\mathsf{Real}^{\uparrow}$ argument. The trigonometric witnesses and classifier rows supply the base values $\sin(0)$ and $\sin(\mathsf{Pi})$ as zero.

For the induction step, the supplied addition-formula row rewrites $\sin((n+1)\mathsf{Pi})$ through $\sin(n\mathsf{Pi})\cos(\mathsf{Pi}) + \cos(n\mathsf{Pi})\sin(\mathsf{Pi})$. Product stability transports the induction hypothesis and the base zero classifier through the product terms, and the supplied uniqueness row returns the classifier $\mathsf{RealClassifier}(s, \mathsf{e})$. □

74.5 Certificate target

Theorem 74.28 (Real-analytic Real-completion dependency surface)**.** *Any admissible use of the local endpoint rows for* Pi*,* Exp*,* Log*,* Sin*, or* Cos *must factor through the following supplied Phase Real and Completion faces: the* DyadicRatCore *arithmetic and modulus rows for the relevant partial sums, the* StreamName$^{\uparrow}$ *observation-carrier and classifier rows, the* RegSeqRat *regularity and stream-classifier rows, the* Real$^{\uparrow}$ *seal or limit-witness rows, and* RealClassifier *limit-uniqueness transport. The local partial-sum, Cauchy-modulus, parity-split, and addition-formula rows do not by themselves provide a* Real$^{\uparrow}$ *endpoint classifier or a completed* NameCert(RealAnalytic$^{\uparrow}$)*.*

Proof. For Pi, section 74.1 supplies the Leibniz partial-sum family and its explicit Cauchy modulus. Those rows are rational stream data only: the endpoint becomes a real-history object only after the supplied DyadicRatCore arithmetic, StreamName$^{\uparrow}$ observation, RegSeqRat regularity, Real$^{\uparrow}$ limit witness, and RealClassifier uniqueness rows are present.

For Exp, section 74.2 supplies the power-series partial sums, the bounded-input modulus, and the local additivity row. The additivity statement uses supplied Real$^{\uparrow}$ witnesses and classifier uniqueness for the three exponential endpoints; without those faces, the Cauchy-product comparison remains a regular-rational stream comparison and does not name a completed endpoint.

For Log, section 74.3 supplies the inverse-of-exponential bisection scheme and the inverse relation under positivity, apartness, rational-embedding, and bisection hypotheses. The bisection rows produce a candidate stream, while the Real$^{\uparrow}$ seal or limit witness and uniqueness transport are the supplied Completion faces that turn that stream into a usable logarithm endpoint.

For Sin and Cos, section 74.4 supplies the trigonometric partial sums, their shared modulus, the parity splits, and the addition-formula rows. The parity and addition formulas compare coefficient streams and transported product streams; they still require the supplied StreamName$^{\uparrow}$, RegSeqRat, Real$^{\uparrow}$, and uniqueness rows before either trigonometric endpoint has a real-history classifier.

Repacking these five cases gives exactly the dependency surface consumed by Definition 74.29: every analytic endpoint enters through the displayed Phase Real and Completion faces, and every local calculation remains a proof obligation until those faces are supplied. □

Closure status for NameCert$_{\text{RealAnalytic}^{\uparrow}}$**.**

Constructive story: The real-analytic boundary starts with explicit rational streams: the Leibniz rescaling for π, power-series partial sums for $\exp$, bisection data for $\log$, and paired sine and cosine partial sums, each indexed by unary histories and equipped with an explicit modulus. A candidate endpoint is accepted only when the supplied DyadicRatCore, StreamName$^{\uparrow}$, RegSeqRat, Real$^{\uparrow}$, and RealClassifier faces turn that local stream into a real-history value; a bare partial-sum formula or local Cauchy estimate is not yet a carried analytic object. The classifier is supplied limit uniqueness in the Real$^{\uparrow}$ surface, while the source is the local stream package plus the real-completion witnesses, the pattern records the partial-sum recursions, moduli, additivity, inverse, Pythagorean, and sine-zero rows, and the ledger keeps stream names, indices, moduli, limit witnesses, and transported classifier evidence visible. Under those supplied rows, the intended name-certificate obligations would be populated by the endpoint witnesses, real classifier equivalence, and transport/stability clauses; without them this chapter deliberately exposes only the dependency surface rather than a completed certificate. A history h lies in the signature gap of a RealAnalytic$^{\uparrow}$ token p only when the source-domain first accepts the corresponding analytic stream and supplied limit witness, the signature relation pairs it with an answer real-history endpoint and evidence ledger, and token introduction names that endpoint as p; unfolded, this says the endpoint for π, $\exp$, $\log$, $\sin$, or $\cos$ is justified by the listed completion faces and not by the local series alone. No standard bridge is attached for these analytic endpoints. The chapter registers this conditional endpoint package under the name RealAnalytic$^{\uparrow}$; transcendental functions remain explicit stream-and-limit obligations over existing real histories, with no completeness axiom, analytic continuation primitive, or host transcendental library introduced.

Theory closure: TheoryGate(NameCert$_{\text{RealAnalytic}^{\uparrow}}$, seed)

Scope closed: Definition 74.28 binds the paper-side seed scope to the supplied DyadicRatCore, StreamNameUp, RegSeqRat, RealUp, and RealClassifier dependency faces for the five displayed analytic endpoints.

Formal verification: FormalStatus(NameCert$_{\text{RealAnalytic}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealAnalyticUp.real\_analytic\_certificate\_boundary`

Bridge status: none

Not claimed: This chapter does not claim a completed NameCert(RealAnalytic$^\uparrow$), Real$^\uparrow$ completeness, or standard bridges for the analytic endpoints.

Upgrade path: A higher closure grade requires checked carrier, classifier, exactness, ledger, and bridge rows for the analytic endpoint package over the completed RealUp surface.

Theorem 74.29 (RealAnalytic$^\uparrow$ certificate boundary)**.** *Suppose the following machine-checked rows are supplied for the analytic construction over* Rat$^\uparrow$, StreamName$^\uparrow$, RegSeqRat, *and* Real$^\uparrow$*:*

1. ***Dyadic rational core:*** DyadicRatCore *arithmetic, signed fractions, factorial denominators, powers, and positivity comparisons for the displayed series and bisection predicates.*
2. ***Stream carrier:*** StreamName$^\uparrow$ *and* RegSeqRat *data for every displayed partial-sum family, including the explicit unary moduli of Definition 74.2, Definition 74.16, and Definition 74.24.*
3. ***Real carrier:*** Real$^\uparrow$ *limit witnesses for* Pi, Exp, Log, Sin, *and* Cos, *supplied from the Phase Real and Completion obligations.*
4. ***Classifier:*** *limit uniqueness for the supplied* Real$^\uparrow$ *witnesses, stated up to* RealClassifier.
5. ***Stability:*** *supplied conditional rows for additivity of* exp *(Definition 74.18), inverse relation* $\log = \exp^{-1}$ *(Definition 74.20), Pythagorean identity (Definition 74.26), and* π*-multiple zeros of* sin *(Definition 74.27), each with its theorem-site hypotheses for algebraic operations, bisection or Cauchy-product comparison, limit witnesses, classifiers, addition formulas, and product stability.*

Under these supplied rows, the tuple (Pi, Exp, Log, Sin, Cos)*, with sources, classifiers, stability obligations, ledgers, and standard bridges, is a* NameCert(RealAnalytic$^\uparrow$) *candidate over* Real$^\uparrow$*. Without those supplied rows, this chapter exports only the displayed proof obligations and explicit-modulus scaffolds.*

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.real analytic certificate boundary`

Proof. The local constructions provide regular rational stream data: the Leibniz rescaling for π, the exponential power-series family, the bisection presentation of log, and the sine and cosine power-series families. Their moduli are explicit unary functions, so they can be passed to the StreamName$^\uparrow$ and RegSeqRat rows once those rows are available.

The supplied Real$^\uparrow$ limit witnesses then assign real-history candidates to the five analytic objects. Limit uniqueness supplies the classifier row, and the four stability obligations listed in the statement provide the algebraic and trigonometric preservation clauses required by the NameCert template.

Thus the tuple has the shape of a naming-certificate candidate exactly when the rows in the statement are present. In their absence, the chapter has not produced the carrier, classifier, or stability package needed for a completed NameCert(RealAnalytic$^\uparrow$). □

Corollary 74.30 (RealAnalytic$^\uparrow$ primitive boundary)**.** RealAnalytic$^\uparrow$ *is not a BEDC primitive, and this chapter alone does not license it as an available naming certificate. Its admissible use requires the supplied Real and Completion rows named in Definition 74.29.*

Lemma 74.31 (Odd trigonometric Cauchy-product coefficient split)**.** *For* Real$^\uparrow$*-histories* x *and* y *and unary degree* n*, assume supplied* RealAlgOrder *rows for addition, products, powers, factorial-denominator inverses, signs, and the binomial expansion of* $(x+y)^{2n+1}$*. Then the degree-*$2n+1$ *coefficient of the regular rational Cauchy product*

$$\mathsf{SinPart}(x,\cdot,\cdot)\mathsf{CosPart}(y,\cdot,\cdot) + \mathsf{CosPart}(x,\cdot,\cdot)\mathsf{SinPart}(y,\cdot,\cdot)$$

is classified with the degree-$2n+1$ *coefficient of* $\mathsf{SinPart}(x+y,\cdot,\cdot)$*.*

Proof. Expand the first product into the odd-in-x, even-in-y terms

$$(-1)^{i+j}\frac{x^{2i+1}y^{2j}}{(2i+1)!(2j)!},$$

and expand the second product into the even-in-x, odd-in-y terms

$$(-1)^{i+j}\frac{x^{2i}y^{2j+1}}{(2i)!(2j+1)!}.$$

On the degree-$2n+1$ slice, both products have $i+j=n$, so all signs are $(-1)^n$. The supplied factorial-denominator rows move every term to the common denominator $(2n+1)!$. The first family contributes the binomial terms in which the exponent of x is odd, and the second family contributes the binomial terms in which the exponent of x is even. These two disjoint parity classes exhaust the binomial expansion of $(x+y)^{2n+1}$. After multiplication by the common sign $(-1)^n$ and the common denominator inverse, the displayed coefficient is classified with the corresponding coefficient of $\mathsf{SinPart}(x+y,\cdot,\cdot)$. □

Theorem 74.32 (Sine addition formula row)**.** *For* $\mathsf{Real}^{\uparrow}$*-histories* x *and* y*, let* z *be the endpoint supplied by the* RealAlgOrder *addition row for* $x+y$*. Assume supplied trigonometric limit witnesses for* $\mathsf{Sin}(x,s_x)$*,* $\mathsf{Cos}(x,c_x)$*,* $\mathsf{Sin}(y,s_y)$*,* $\mathsf{Cos}(y,c_y)$*, and* $\mathsf{Sin}(z,s_z)$*. Assume supplied* RealAlgOrder *rows naming the product endpoints* $s_x{\cdot}c_y$ *and* $c_x{\cdot}s_y$ *and their sum, product-stability rows saying that these endpoints are the* $\mathsf{Real}^{\uparrow}$ *limits of the corresponding Cauchy-product streams, and a regular-rational Cauchy-product comparison whose coefficient part is Definition 74.31 and whose tail estimates use the modulus of Definition 74.24. If the supplied limit-uniqueness classifier rows compare that product-sum stream with* $\mathsf{SinPart}(z,\cdot,\cdot)$ *at their real-history limits, then*

$$\mathsf{RealClassifier}(s_z, s_x\cdot c_y + c_x\cdot s_y).$$

Proof. By Definition 74.25, the five trigonometric witnesses expose regular rational stream names for the sine and cosine partial sums of x, y, and z. The addition row identifies z as the $\mathsf{Real}^{\uparrow}$ endpoint for $x+y$, while the product and sum rows make the right-hand endpoint $s_x\cdot c_y + c_x\cdot s_y$ meaningful inside the same RealAlgOrder interface.

Product stability transports the four input trigonometric limits through the two displayed products and then through the final sum. Thus the right-hand endpoint is the $\mathsf{Real}^{\uparrow}$ limit of the regular rational stream obtained from

$$\mathsf{SinPart}(x,\cdot,\cdot)\mathsf{CosPart}(y,\cdot,\cdot) + \mathsf{CosPart}(x,\cdot,\cdot)\mathsf{SinPart}(y,\cdot,\cdot).$$

The supplied Cauchy-product comparison identifies this stream with the $\mathsf{SinPart}(z,\cdot,\cdot)$ stream: its finite coefficient calculation is the odd-parity split of Definition 74.31, and its tails are bounded by the trigonometric Cauchy modulus of Definition 74.24 after passing to a common unary precision.

The two real-history endpoints are therefore limits of classified regular rational stream names. Applying the supplied limit-uniqueness classifier row gives

$$\mathsf{RealClassifier}(s_z, s_x\cdot c_y + c_x\cdot s_y).$$

□

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticSinAdd local product sum unary`

Lemma 74.33 (Even trigonometric Cauchy-product coefficient split)**.** *For* $\mathsf{Real}^{\uparrow}$*-histories* x *and* y *and unary degree* n*, assume supplied* RealAlgOrder *rows for addition, products, differences, powers, factorial-denominator inverses, signs, and the binomial expansion of* $(x+y)^{2n}$*. Then the degree-*$2n$ *coefficient of the regular rational Cauchy-product difference*

$$\mathsf{CosPart}(x,\cdot,\cdot)\mathsf{CosPart}(y,\cdot,\cdot) - \mathsf{SinPart}(x,\cdot,\cdot)\mathsf{SinPart}(y,\cdot,\cdot)$$

is classified with the degree-$2n$ *coefficient of* $\mathsf{CosPart}(x+y,\cdot,\cdot)$*. The odd total-degree slices on both sides are empty by parity.*

Proof. Expand the cosine-cosine product into the even-in-x, even-in-y terms

$$(-1)^{i+j}\frac{x^{2i}y^{2j}}{(2i)!(2j)!},$$

and expand the sine-sine product into the odd-in-x, odd-in-y terms

$$(-1)^{i+j}\frac{x^{2i+1}y^{2j+1}}{(2i+1)!(2j+1)!}.$$

On the degree-$2n$ slice, the cosine-cosine product has $i+j=n$, so every sign is $(-1)^n$. The sine-sine product has $i+j=n-1$ when $n>0$, so its sign is $(-1)^{n-1}$; the displayed subtraction changes this to $(-1)^n$. For $n=0$, the sine-sine contribution is empty.

The supplied factorial-denominator rows move every nonempty term to the common denominator $(2n)!$. The cosine-cosine family contributes the binomial terms in which the exponent of x is even, and the subtracted sine-sine family contributes the binomial terms in which the exponent of x is odd. These two parity classes partition the binomial expansion of $(x+y)^{2n}$. After multiplication by the common sign $(-1)^n$ and the common denominator inverse, the displayed coefficient is classified with the corresponding coefficient of $\mathsf{CosPart}(x+y,\cdot,\cdot)$.

No odd total degree can arise from an even-even product or an odd-odd product, and $\mathsf{CosPart}(x+y,\cdot,\cdot)$ has only even powers. Hence the odd slices are empty on both sides. □

Theorem 74.34 (Cosine addition formula row)**.** *For* $\mathsf{Real}^{\uparrow}$*-histories* x *and* y*, let* z *be the endpoint supplied by the* RealAlgOrder *addition row for* $x+y$*. Assume supplied trigonometric limit witnesses for* $\mathsf{Cos}(z,c_z)$*,* $\mathsf{Cos}(x,c_x)$*,* $\mathsf{Cos}(y,c_y)$*,* $\mathsf{Sin}(x,s_x)$*, and* $\mathsf{Sin}(y,s_y)$*. Assume supplied* RealAlgOrder *rows naming the product endpoints* $c_x\cdot c_y$ *and* $s_x\cdot s_y$ *and the displayed difference* $c_x\cdot c_y - s_x\cdot s_y$*, product-stability rows saying that the two product endpoints are the* $\mathsf{Real}^{\uparrow}$ *limits of the corresponding Cauchy-product streams, and the supplied difference row saying that* $c_x\cdot c_y - s_x\cdot s_y$ *is the* $\mathsf{Real}^{\uparrow}$ *limit of their pointwise difference. Assume also a regular-rational Cauchy-product comparison whose coefficient part is Definition 74.33 and whose tail estimates use the modulus of Definition 74.24. If the supplied limit-uniqueness classifier rows compare that product-difference stream with* $\mathsf{CosPart}(z,\cdot,\cdot)$ *at their real-history limits, then*

$$\mathsf{RealClassifier}(c_z, c_x\cdot c_y - s_x\cdot s_y).$$

Proof. By Definition 74.25, the five trigonometric witnesses expose regular rational stream names for the cosine partial sums of z, x, and y, and for the sine partial sums of x and y. The addition row identifies z as the $\mathsf{Real}^{\uparrow}$ endpoint for $x+y$. The operation rows are the addition, negation, and multiplication rows exposed by the RealAlgOrder interface of Definition 38.38; they make the right-hand endpoint $c_x\cdot c_y - s_x\cdot s_y$ meaningful in the same real-history interface.

Product stability transports the input trigonometric limits through the two products $c_x \cdot c_y$ and $s_x \cdot s_y$. The supplied difference row then transports those product limits through the displayed subtraction. Thus the right-hand endpoint is the $\mathsf{Real}^{\uparrow}$ limit of the regular rational stream obtained from

$$\mathsf{CosPart}(x, \cdot, \cdot)\mathsf{CosPart}(y, \cdot, \cdot) - \mathsf{SinPart}(x, \cdot, \cdot)\mathsf{SinPart}(y, \cdot, \cdot).$$

The supplied Cauchy-product comparison identifies this stream with the $\mathsf{CosPart}(z, \cdot, \cdot)$ stream: its finite coefficient calculation is the even-parity split of Definition 74.33, and its tails are bounded by the trigonometric Cauchy modulus of Definition 74.24 after passing to a common unary precision.

The two real-history endpoints are therefore limits of classified regular rational stream names. Applying the supplied limit-uniqueness classifier row gives

$$\mathsf{RealClassifier}(c_z, c_x \cdot c_y - s_x \cdot s_y).$$

□

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticCosAdd local product difference unary`

Lemma 74.35 (Cosine partial sums at classified empty input)**.** *Let x be a $\mathsf{Real}^{\uparrow}$-history with $\mathsf{RealClassifier}(x, \mathsf{e})$. Assume supplied $\mathsf{RealAlgOrder}$ rows for the powers x^{2k}, the convention $x^0 = \mathsf{E1}(\mathsf{e})$, classifier-stability of multiplication along $\mathsf{RealClassifier}$, the positive-power vanishing rows*

$$k \geq \mathsf{E1}(\mathsf{e}) \Longrightarrow \mathsf{RealClassifier}(x^{2k}, \mathsf{e}),$$

and the unary factorial-denominator rows for $(2k)!$. Then for every unary index n, every partial-sum witness $\mathsf{CosPart}(x, n, C_n)$ is classified with the constant-one endpoint:

$$\mathsf{RealClassifier}(C_n, \mathsf{E1}(\mathsf{e})).$$

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticCosPart empty constant one`

Proof. Proceed by unary induction on the displayed partial-sum recursion of Definition 74.23. At $n = \mathsf{e}$, the only cosine term is

$$(-1)^0 \frac{x^0}{0!}.$$

The supplied zero-power convention gives $x^0 = \mathsf{E1}(\mathsf{e})$, the unary factorial row gives $0! = \mathsf{E1}(\mathsf{e})$, and the sign row gives $(-1)^0 = \mathsf{E1}(\mathsf{e})$. The multiplication and denominator rows in the $\mathsf{RealAlgOrder}$ interface therefore classify the base partial sum with $\mathsf{E1}(\mathsf{e})$.

For the successor step, write the next partial sum as the previous partial sum plus the term

$$(-1)^{m+1} \frac{x^{2m+2}}{(2m+2)!}.$$

The exponent $2m + 2$ is positive for every unary m. The positive-power vanishing row, obtained from the displayed classifier $\mathsf{RealClassifier}(x, \mathsf{e})$ by multiplication stability, classifies x^{2m+2} with e. Multiplication by the supplied sign and factorial-denominator endpoints preserves that zero classifier, so the added term is classified with e.

By the induction hypothesis the previous partial sum is classified with $\mathsf{E1}(\mathsf{e})$. The $\mathsf{RealAlgOrder}$ addition row and classifier stability for adding the zero endpoint classify the successor partial sum with the same $\mathsf{E1}(\mathsf{e})$ endpoint. Hence every unary cosine partial sum at an input classified with e is the constant-one partial sum. □

Theorem 74.36 (Cosine at classified empty input)**.** *Let x be a $\mathsf{Real}^{\uparrow}$-history with $\mathsf{RealClassifier}(x, \mathsf{e})$, and let c be a supplied endpoint with $\mathsf{Cos}(x, c)$ in the sense of Definition 74.25. Assume the algebraic rows of Definition 74.35, the $\mathsf{RegSeqRat}$ row for the constant-one rational stream $n \mapsto \mathsf{E1}(\mathsf{e})$, the $\mathsf{Real}^{\uparrow}$ limit witness naming that stream by $\mathsf{E1}(\mathsf{e})$, and supplied limit-uniqueness classifier rows comparing the $\mathsf{CosPart}(x, \cdot, \cdot)$ stream with this constant-one stream. Then*

$$\mathsf{RealClassifier}(c, \mathsf{E1}(\mathsf{e})).$$

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticCosEmpty constant one`

Proof. The supplied $\mathsf{Cos}(x, c)$ witness exposes the regular rational stream of cosine partial sums from Definition 74.23, with the Cauchy modulus supplied by Definition 74.24. By Definition 74.35, every unary partial sum in that stream is classified with the corresponding value of the constant-one stream.

Thus the two regular rational streams have the same values at every unary precision up to the displayed $\mathsf{RealClassifier}$ comparison. The constant-one stream has the supplied $\mathsf{Real}^{\uparrow}$ limit witness whose endpoint is $\mathsf{E1}(\mathsf{e})$. The cosine witness gives c as the $\mathsf{Real}^{\uparrow}$ limit of the cosine partial-sum stream. Applying the supplied limit-uniqueness classifier row to these two classified regular streams gives

$$\mathsf{RealClassifier}(c, \mathsf{E1}(\mathsf{e})).$$

□

Definition 74.37 (Constant-one exponential comparison stream)**.** *The constant-one comparison stream for the exponential package is the unary observation relation $\mathsf{ExpOneStream}(n, O_n)$ defined by the two rows $\mathsf{UnaryHistory}(n)$ and $O_n\mathsf{hsameE1}(\mathsf{e})$. Its regular rational spine is the constant-stream spine of Definition 38.205 at the RatUp unit history.*

Lean4 (definition) `BEDC.Derived.RealAnalyticUp.RealAnalyticExpOneStream`

Lemma 74.38 (Exponential partial sums at empty input)**.** *For every unary index n, assume the supplied $\mathsf{RealAlgOrder}$ rows for the zero constant e, the unit constant $\mathsf{E1}(\mathsf{e})$, powers of e, products with e, addition with e, and the unary factorial-denominator inverses used in Definition 74.6. Assume in particular the power rows stating that e^0 is classified with $\mathsf{E1}(\mathsf{e})$ and that e^k is classified with e for every successor unary k. If $\mathsf{ExpPart}(\mathsf{e}, n, S_n)$ and $\mathsf{ExpOneStream}(n, O_n)$, then*

$$\mathsf{RealClassifier}(S_n, O_n).$$

Proof. Induct on the unary index n exposed by $\mathsf{ExpOneStream}(n, O_n)$.

In the empty-index case, Definition 74.6 gives the single term $\mathsf{e}^0/0!$. The zero-power row classifies e^0 with $\mathsf{E1}(\mathsf{e})$, and the zero constructor of Definition 64.140 gives $0!$ as $\mathsf{E1}(\mathsf{e})$. The supplied denominator-inverse row for the unit denominator and the unit multiplication row therefore classify the displayed partial sum S_{e} with $\mathsf{E1}(\mathsf{e})$. Since $\mathsf{ExpOneStream}(\mathsf{e}, O_{\mathsf{e}})$ exposes $O_{\mathsf{e}}\mathsf{hsameE1}(\mathsf{e})$, classifier transport gives $\mathsf{RealClassifier}(S_{\mathsf{e}}, O_{\mathsf{e}})$.

For the successor step, write the predecessor partial sum as S_n and the successor partial sum as $S_{\mathsf{E1}(n)}$. By the induction hypothesis, S_n is classified with the predecessor constant-one observation, hence with $\mathsf{E1}(\mathsf{e})$. The successor summand in Definition 74.6 is $\mathsf{e}^{\mathsf{E1}(n)}/(\mathsf{E1}(n))!$. The supplied positive-power row classifies the numerator $\mathsf{e}^{\mathsf{E1}(n)}$ with e. The factorial row supplies a positive unary denominator and its inverse, but multiplication of the zero numerator by that carried inverse is classified with e. The supplied addition-with-zero row then classifies $S_{\mathsf{E1}(n)}$ with the predecessor value, hence with $\mathsf{E1}(\mathsf{e})$. The successor observation row $\mathsf{ExpOneStream}(\mathsf{E1}(n), O_{\mathsf{E1}(n)})$ exposes $O_{\mathsf{E1}(n)}\mathsf{hsameE1}(\mathsf{e})$, so classifier transport yields $\mathsf{RealClassifier}(S_{\mathsf{E1}(n)}, O_{\mathsf{E1}(n)})$. □

Theorem 74.39 (Exponential at empty input is one)**.** *Let x and e be carried* $\mathsf{Real}^{\uparrow}$ *histories. Assume a supplied classifier row* $\mathsf{RealClassifier}(x, \mathsf{e})$*, a supplied* $\mathsf{Exp}(x, e)$ *limit witness as in Definition 74.17, and the classifier-stability rows that transport every unary finite computation* $\mathsf{ExpPart}(x, n, T_n)$ *along* $\mathsf{RealClassifier}(x, \mathsf{e})$ *to an empty-input computation* $\mathsf{ExpPart}(\mathsf{e}, n, S_n)$*. Assume also the algebra rows of Definition 74.38, a supplied* $\mathsf{RegSeqRat}$ *spine and* $\mathsf{Real}^{\uparrow}$ *limit witness for* $\mathsf{ExpOneStream}$ *with endpoint* $\mathsf{E1}(\mathsf{e})$*, and the supplied limit-uniqueness row comparing the transported* $\mathsf{ExpPart}(x, \cdot, \cdot)$ *stream with* $\mathsf{ExpOneStream}$*. Then*

$$\mathsf{RealClassifier}(e, \mathsf{E1}(\mathsf{e})).$$

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticExp endpoint unary`

Proof. The supplied $\mathsf{Exp}(x, e)$ witness exposes the regular rational stream of exponential partial sums for x and names e as its $\mathsf{Real}^{\uparrow}$ endpoint. For each unary observation index, the input classifier $\mathsf{RealClassifier}(x, \mathsf{e})$ and the stated stability rows transport the observed finite computation from $\mathsf{ExpPart}(x, n, T_n)$ to an empty-input computation $\mathsf{ExpPart}(\mathsf{e}, n, S_n)$.

By Definition 74.38, each empty-input observation is classified with the corresponding observation of $\mathsf{ExpOneStream}$. Thus the transported exponential partial-sum stream and the constant-one stream carry the required stream classifier.

The constant-one stream has its supplied $\mathsf{RegSeqRat}$ spine and its supplied $\mathsf{Real}^{\uparrow}$ limit witness at $\mathsf{E1}(\mathsf{e})$. Applying the supplied limit-uniqueness row to the stream classifier and to the two endpoint witnesses identifies the endpoint e of $\mathsf{Exp}(x, e)$ with the constant-one endpoint, giving

$$\mathsf{RealClassifier}(e, \mathsf{E1}(\mathsf{e})).$$

□

Theorem 74.40 (Sine at the empty real input has zero endpoint)**.** *Let x be a* $\mathsf{Real}^{\uparrow}$*-history with* $\mathsf{RealClassifier}(x, \mathsf{e})$*. Assume the* $\mathsf{RealAlgOrder}$ *rows for the zero constant* e*, multiplication by* e*, odd powers of x, factorial-denominator inverses, signs, and finite addition in the sine partial sums of Definition 74.23. Assume also the classifier-congruence rows for these algebraic operations. If* $\mathsf{Sin}(x, s)$ *is supplied by the limit witness of Definition 74.25, and the supplied limit-uniqueness rows compare* $\mathsf{SinPart}(x, \cdot, \cdot)$ *with the constant-zero regular rational stream whose* $\mathsf{Real}^{\uparrow}$ *endpoint is* e*, then*

$$\mathsf{RealClassifier}(s, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.RealAnalyticUp.RealAnalyticSinPart empty zero`

Proof. The classifier $\mathsf{RealClassifier}(x, \mathsf{e})$ places the input in the zero classifier class. By classifier congruence for the $\mathsf{RealAlgOrder}$ multiplication rows, each positive power occurring in the sine series is classified with the corresponding positive power of e. Since every exponent $2k + 1$ is nonzero, the supplied zero-multiplication rows identify e^{2k+1} with e. Hence every numerator x^{2k+1} in the kth sine term is classified with e.

Multiplication by the signed factorial-denominator inverse preserves the zero classifier class, again by the supplied $\mathsf{RealAlgOrder}$ congruence rows. Thus each term

$$(-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

is classified with the rational zero term. Induction on the unary partial-sum index in Definition 74.23 now gives that every witness $\mathsf{SinPart}(x, n, S_n)$ is classified with the nth value of the

constant-zero regular rational stream: the empty partial sum is zero, and the step adds a term already classified with zero, so finite addition preserves the zero classifier class.

The trigonometric Cauchy modulus of Definition 74.24 makes the sine partial sums an admissible regular rational stream, and the constant-zero stream is regular with the same endpoint e. The supplied $\mathsf{Sin}(x, s)$ witness names s as the $\mathsf{Real}^{\uparrow}$ limit of $\mathsf{SinPart}(x, \cdot, \cdot)$. Applying the supplied limit-uniqueness row to the pointwise classifier comparison with the constant-zero stream gives

$$\mathsf{RealClassifier}(s, \mathsf{e}).$$

□

Chapter 75

A Concrete Naming Certificate for $\mathsf{Gamma}^{\uparrow}$

Lean4 ✓ `BEDC.Derived.GammaUp.GammaTasteGate single carrier alignment`

This chapter gives a concrete naming-certificate construction for the constructive Gamma-function interface $\mathsf{Gamma}^{\uparrow}$ at the paper-proof level. The construction sits above $\mathsf{Nat}^{\uparrow}$ (chapter 28) for the unary factorial, $\mathsf{Prime}^{\uparrow}$ (chapter 64) for the explicit prime-power decomposition of factorials, $\mathsf{RealAnalytic}^{\uparrow}$ (chapter 74) for the constructive exp and log used in the analytic extension, $\mathsf{Complex}^{\uparrow}$ (chapter 39) for the codomain, and $\mathsf{Holomorphic}^{\uparrow}$ (chapter 68) for the meromorphic-extension classifier. The construction proceeds in three layers: integer-point factorial ($\Gamma(n+1) = n!$), Weierstrass-product extension to $\mathsf{Complex}^{\uparrow} \setminus \{0, -1, -2, \ldots\}$ with explicit unary modulus, and the recurrence $\Gamma(s+1) = s\,\Gamma(s)$. All convergence witnesses are unary-indexed total functions; the construction uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

75.1 The pole locus and the Gamma domain

Definition 75.1 (Negative-integer pole locus)**.** *The pole-locus predicate* $\mathsf{GammaPole}(s)$ *on* $\mathsf{Complex}^{\uparrow}$*-history holds when there exists a unary natural* n *with* $\mathsf{hsame}(s, -n)$*, where* $-n$ *is the explicit signed-integer encoding of chapter 31. Equivalently, the imaginary part of* s *equals* e *and the real part is a non-positive integer.*

Lean4 (definition) `BEDC.Derived.GammaUp.GammaPoleLocus`

Theorem 75.2 (Gamma pole locus has complex carrier witness)**.** *If* s *lies in the Gamma pole locus through a unary witness* n*, then the same witness presents* s *as* hsame *to the complex history with real component* $\mathsf{E1}(n)$ *and imaginary component* $\mathsf{E1}(\mathsf{e})$*, and* s *is carried by* $\mathsf{Complex}^{\uparrow}$*.*

Proof. The unary witness n gives a positive rational denominator for $\mathsf{E1}(n)$, while e gives the positive rational denominator for $\mathsf{E1}(\mathsf{e})$. The product carrier forms the corresponding complex history, and the pole-locus hsame witness transports that carrier back to s. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus complex carrier witness`

Theorem 75.3 (Gamma pole locus is not empty)**.** *If* s *lies in the Gamma pole locus, then* s *cannot be* hsame *to* e*.*

Proof. The pole-locus witness induces a $\mathsf{Complex}^{\uparrow}$ carrier for s by the complex-carrier witness theorem. The complex carrier is non-empty, so an $\mathsf{hsame}(s, \mathsf{e})$ assumption contradicts that carrier. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus not empty`

Theorem 75.4 (Gamma pole locus is stable under unary append as a complex carrier)**.** *If s lies in the Gamma pole locus and q is unary, then* $\mathsf{append}(s, q)$ *is carried by* $\mathsf{Complex}^{\uparrow}$ *and cannot be* hsame *to* e.

Proof. The pole-locus witness gives a $\mathsf{Complex}^{\uparrow}$ carrier for s. Complex carriers are closed under unary append, so $\mathsf{append}(s, q)$ is again a complex carrier. The complex-carrier non-emptiness theorem excludes any comparison of that appended history with e. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus append unary complex carrier`

Theorem 75.5 (Gamma pole locus hsame transport witness)**.** *If* $s\mathsf{hsame}t$ *and s lies in the Gamma pole locus, then t lies in the same pole locus. Moreover, the transported pole claim exposes a concrete unary witness n with*

$$t\mathsf{hsameappend}(\mathsf{E1}(n), \mathsf{E1}(\mathsf{e})).$$

Proof. The pole-locus witness for s supplies a unary history n and an equality from s to the explicit append history. Compose the symmetry of $s\mathsf{hsame}t$ with that equality. The same unary n then witnesses both the pole-locus claim for t and the displayed append-history equality. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus hsame transport witness`

Theorem 75.6 (Gamma pole locus witness uniqueness)**.** *If the same complex history s is displayed as a Gamma pole by unary witnesses n and m, then* $n\mathsf{hsame}m$.

Proof. The two displayed pole equalities identify the append histories $\mathsf{append}(\mathsf{E1}(n), \mathsf{E1}(\mathsf{e}))$ and $\mathsf{append}(\mathsf{E1}(m), \mathsf{E1}(\mathsf{e}))$. Cancelling the common right component $\mathsf{E1}(\mathsf{e})$ gives $\mathsf{E1}(n)\mathsf{hsame}\mathsf{E1}(m)$, and constructor inversion gives $n\mathsf{hsame}m$. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus witness unique`

Theorem 75.7 (Gamma pole locus suffix source and witness determinacy)**.** *If s and t lie in the Gamma pole locus and the same suffix q continues both histories to the same endpoint sq, then* $s\mathsf{hsame}t$. *Moreover, the two pole displays expose unary witnesses n and m with* $n\mathsf{hsame}m$.

Proof. Continuation right cancellation on the two ledgers gives $s\mathsf{hsame}t$. Transport the pole-locus witness for t back to the source s, then apply Gamma pole-locus witness uniqueness against the original witness for s. The two unary witnesses and their equality form the displayed witness package. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus suffix source witness deterministic`

Definition 75.8 (Gamma domain)**.** $\mathsf{GammaDom}(s)$ *holds when* $\mathsf{ComplexUp}(s)$ *and* $\neg\mathsf{GammaPole}(s)$. *The negation is constructive:* $\mathsf{GammaDom}(s)$ *is witnessed by an explicit apartness datum (k, w) where k is a unary precision and w is a strict-prefix witness showing that the distance from s to the closest non-positive integer exceeds 2^{-k}.*

Lean4 (definition) `BEDC.Derived.GammaUp.GammaDomainCore`

Theorem 75.9 (Gamma domain core is non-empty)**.** *If a Gamma-domain core witness carries s as a complex history and records constructive apartness from the pole locus, then s cannot be* hsame *to* e.

Proof. Unpack the Gamma-domain core witness and read its $\mathsf{Complex}^{\uparrow}$ carrier for s. The complex carrier is non-empty, so any $\mathsf{hsame}(s, \mathsf{e})$ assumption contradicts the carrier. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaDomainCore not empty`

Theorem 75.10 (Gamma domain core transports exclusions)**.** *If* $s$hsame$t$ *and s is in the Gamma-domain core, then t is excluded from the pole locus and* $t$hsamee *is impossible.*

Proof. Transport any pole-locus witness for t back along the symmetric hsame-comparison to obtain a pole-locus witness for s, contradicting the domain-core apartness field. For the empty-history exclusion, compose $s$hsame$t$ with $t$hsamee and apply the non-empty core theorem. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaDomainCore hsame transport exclusions`

Theorem 75.11 (Gamma domain core hsame transport)**.** *If* $s$hsame$t$ *and s is in the Gamma-domain core with apartness datum apart, then t is in the Gamma-domain core with the same apartness datum, and* $t$hsamee *is impossible.*

Proof. Transport the complex-history carrier of s along $s$hsame$t$. The unary apartness datum is unchanged. The pole-locus and empty-history exclusions are the two exclusion clauses transported by the preceding theorem. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaDomainCore hsame transport`

Theorem 75.12 (Gamma domain core hsame transport nonempty)**.** *If* $s$hsame$t$ *and s is in the Gamma-domain core, then t is in the same domain core and* $t$hsamee *is impossible.*

Proof. Transport the $\mathsf{Complex}^{\uparrow}$ carrier from s to t. A pole-locus witness for t transports back along the symmetric hsame-comparison to a pole-locus witness for s, contradicting the apartness field of the original domain witness. Together with the unchanged unary apartness datum, these fields rebuild the domain core for t, and the non-empty theorem excludes $t$hsamee. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaDomainCore hsame transport not empty`

Lemma 75.13 (Apartness from the pole locus is decidable)**.** *For every complex history s and unary precision k, the question "does s lie within 2^{-k} of* GammaPole*?" is decidable. The decision either supplies a near-pole witness* $\mathsf{hsame}(\mathsf{Re}(s), -n)$ *up to precision k for some specific unary n, or supplies the apartness datum required by Definition 75.8.*

Proof. Compare the rational real-part history $\mathsf{Re}(s)$ against the finite list of negative-integer candidates $\{0, -1, -2, \ldots, -K\}$ where K is the unary length of $\mathsf{Re}(s)$ at precision k. All comparisons are decidable on rational representations by NatUnaryStrictPrefix (chapter 28). The imaginary-part check uses the same strict-prefix decidability. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaPoleLocus shape partition`

75.2 Gamma at unary positive integers via factorial

Definition 75.14 (Factorial bridge)**.** *For unary $n \geq \mathsf{e}$, $\mathsf{Factorial}(n, m)$ holds when m is the unary product $1 \cdot 2 \cdots n$ computed by unary recursion:* $\mathsf{Factorial}(\mathsf{e}, \mathsf{E1}(\mathsf{e}))$ *and* $\mathsf{Factorial}(\mathsf{E1}(p), m)$ *iff there exists m' with* $\mathsf{Factorial}(p, m')$ *and* $m = \mathsf{E1}(p) \cdot m'$ *under unary multiplication (*$\mathsf{Prime}^{\uparrow}$*, chapter 64).*

Lean4 (definition) `BEDC.Derived.GammaUp.GammaFactorial`

Theorem 75.15 (Gamma at positive integers equals factorial)**.** *For unary $n \geq \mathsf{e}$,* $\mathsf{Gamma}(n + \mathsf{E1}(\mathsf{e}), z)$ *holds with* $\mathsf{hsame}(z, m)$ *where* $\mathsf{Factorial}(n, m)$*. Concretely, $\Gamma(1) = 1$, $\Gamma(2) = 1$, $\Gamma(3) = 2$, and in general $\Gamma(n + 1) = n!$.*

Proof. Direct unfolding of Definition 75.14 together with the recurrence $\Gamma(s + 1) = s\,\Gamma(s)$ of Definition 75.21 below, applied unary-inductively starting from $\Gamma(1) = 1$ (the empty-product base case). □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaFactorial unary result`

Remark 75.16 (Factorial used by the zeta functional equation)**.** *The functional equation $\zeta(s) = 2^s \pi^{s-1} \sin(\pi s/2)\Gamma(1 - s)\zeta(1 - s)$ (chapter 9342) requires $\Gamma(1 - s)$ at $s = -2n$ (negative even integers), which is $\Gamma(1 + 2n) = (2n)!$. Definition 75.15 supplies this value as a closed unary expression.*

75.3 Gamma extension via Weierstrass product

Definition 75.17 (Weierstrass partial product)**.** *For complex history s with* $\mathsf{GammaDom}(s)$ *and unary index N, the Weierstrass partial product predicate* $\mathsf{GammaPart}(s, N, P_N)$ *holds when P_N is the explicit complex history*

$$P_N \;=\; \frac{1}{s} \prod_{n=1}^{N} \left(1 + \frac{1}{n}\right)^{s} \left(1 + \frac{s}{n}\right)^{-1},$$

computed by unary recursion using complex multiplication, division (legitimate by $\mathsf{GammaDom}(s)$ *supplying* $\mathsf{CplxNonZero}$ *at every factor), and the constructive complex* exp *of chapter 74 for the rational power $(1 + 1/n)^s = \exp(s \log(1 + 1/n))$.*

Lean4 (definition) `BEDC.Derived.GammaFunctionUp.GammaWeierstrassPart`

Lemma 75.18 (Weierstrass product Cauchy modulus on the Gamma domain)**.** *For every s with* $\mathsf{GammaDom}(s)$ *at apartness precision k_0, the partial products P_N form a regular complex Cauchy sequence with explicit modulus $N(k) := \max(k_0, k) + |\mathsf{Re}(s)|^2 + |\mathsf{Im}(s)|^2$: for all $M, N \geq N(k)$, the modulus of $P_M - P_N$ is below 2^{-k}.*

Proof. Take logarithms of consecutive partial products: $\log(P_{N+1}/P_N) = s \log(1 + 1/N) - \log(1 + s/N)$. Taylor-expand both logarithms via $\log(1 + u) = u - u^2/2 + O(u^3)$ using the constructive log of Definition 74.19; the leading s/N terms cancel and the residual is of order $|s|^2/N^2$, giving an explicit geometric tail bound. The modulus $N(k)$ is the precision at which the geometric tail falls below 2^{-k}. □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.GammaWeierstrassCauchyModulus constant`

Definition 75.19 (Constructive Gamma)**.** *For s with* $\mathsf{GammaDom}(s)$*,* $\mathsf{Gamma}(s, z)$ *holds when z is the complex limit certificate of Definition 75.18 via the* $\mathsf{CplxLim}^{\uparrow}$ *construction of chapter 65, applied to the partial products $(P_N)_N$. The witness includes the apartness precision k_0, the modulus N, and the limit z.*

Lean4 (definition) `BEDC.Derived.GammaFunctionUp.Gamma`

Theorem 75.20 (Gamma is holomorphic on the Gamma domain)**.** $\mathsf{Gamma} : \mathsf{GammaDom} \to \mathsf{Complex}^{\uparrow}$ *is holomorphic in the sense of* $\mathsf{Holomorphic}^{\uparrow}$ *(chapter 68) on every open disk* $\mathsf{OpenDisk}(z_0, r)$ *entirely contained in* $\mathsf{GammaDom}$*.*

Proof. Term-wise complex differentiability of each Weierstrass factor follows from the constructive exp and log being holomorphic in their arguments (chapter 74). Uniform convergence of the partial products on a compact sub-domain of $\mathsf{GammaDom}$ is the geometric-tail bound of Definition 75.18. Apply the holomorphic-limit closure of $\mathsf{Holomorphic}^{\uparrow}$ (chapter 68). □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.Gamma holomorphic limit certificate readback`

75.4 Functional equation $\Gamma(s+1) = s\,\Gamma(s)$

Theorem 75.21 (Gamma recurrence)**.** *For every s with* $\mathsf{GammaDom}(s)$ *and* $\mathsf{GammaDom}(s + \mathsf{E1}(\mathsf{e}))$*, if* $\mathsf{Gamma}(s, z)$ *and* $\mathsf{Gamma}(s + \mathsf{E1}(\mathsf{e}), w)$*, then* $\mathsf{ComplexClassifier}(w, s \cdot z)$*. Equivalently,* $\Gamma(s+1) = s\,\Gamma(s)$ *up to* $\mathsf{ComplexClassifier}$*.*

Proof. Compare the Weierstrass partial products of $\Gamma(s+1)$ and $s \cdot \Gamma(s)$: the index-shift identity $P_N(s+1) = (s/(N+1)) \cdot s \cdot P_N(s) \cdot$(small correction) emerges term by term from the explicit factor formula. The small correction is bounded by Definition 75.18; the limits agree by uniqueness of $\mathsf{CplxLim}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.Gamma recurrence limit pair carriers`

Theorem 75.22 (Gamma at 1)**.** $\mathsf{Gamma}(\mathsf{E1}(\mathsf{e}), z)$ *gives* $\mathsf{ComplexClassifier}(z, \mathsf{E1}(\mathsf{e}))$*:* $\Gamma(1) = 1$*.*

Proof. Specialize Definition 75.17 at $s = 1$: the product becomes $\frac{1}{1}\prod_{n=1}^{N}(1 + 1/n)(1 + 1/n)^{-1} = 1$ telescopically. The limit is 1 by direct unfolding. □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.GammaAtOne constant certificate`

75.5 Reflection formula boundary certificate

Theorem 75.23 (Reflection formula boundary certificate)**.** *For s with* $\mathsf{GammaDom}(s)$ *and* $\mathsf{GammaDom}(\mathsf{E1}(\mathsf{e}) - s)$*, a BEDC derivation of the classifier target*

$$\mathsf{ComplexClassifier}(\Gamma(s)\Gamma(\mathsf{E1}(\mathsf{e}) - s), \mathsf{Pi}/\mathsf{CplxSin}(\mathsf{Pi} \cdot s))$$

at the Gamma boundary carries the following certificate data:

1. *a Weierstrass-product witness for the $\Gamma(s)$ factor from Definition 75.17;*
2. *a Weierstrass-product witness for the $\Gamma(\mathsf{E1}(\mathsf{e}) - s)$ factor from Definition 75.17;*
3. *a common unary Cauchy modulus controlling both Gamma factors, obtained from Definition 75.18;*

4. *a complex-sine product witness from* $\mathsf{CplxAnalytic}^{\uparrow}$ *(section 76.3);*

5. *an apartness witness showing that* $\mathsf{CplxSin}(\mathsf{Pi}\cdot s)$ *is a valid denominator on the displayed domain.*

```
Lean4 ✓  BEDC.Derived.GammaFunctionUp.Gamma reflection boundary certificate
Lean4 ✓  BEDC.Derived.GammaFunctionUp.Gamma reflection boundary certificate data
```

Proof. The two Gamma factors are already supplied by the closed Gamma construction: unfold Definition 75.19 at s and at $\mathsf{E1}(\mathsf{e}) - s$, then read off the partial products of Definition 75.17. Apply Definition 75.18 to each domain witness and take the larger unary bound as the common modulus.

The boundary also records the shift compatibility needed by any telescoping argument, namely Definition 75.21, together with the complex sine construction of section 76.3. The denominator side is not accepted without an explicit apartness witness for $\mathsf{CplxSin}(\mathsf{Pi}\cdot s)$ on the chosen domain point.

These data are the certificate boundary consumed by downstream zeta arguments. The classifier equality displayed in the statement is a target for a BEDC derivation carrying those witnesses, not an equality asserted by this Gamma certificate alone. □

75.6 Certificate target

Theorem 75.24 (Concrete $\mathsf{Gamma}^{\uparrow}$ naming certificate)**.** *The tuple* (GammaDom, Gamma, Factorial)*, with sources, classifiers, stability obligations, ledger, and standard bridge as listed below, forms a paper-side* $\mathsf{NameCert}(\mathsf{Gamma}^{\uparrow})$ *over* $\mathsf{Complex}^{\uparrow}$*:*

1. ***Source****:* $\mathsf{Complex}^{\uparrow}$*-histories with explicit apartness witnesses against the negative-integer pole locus* GammaPole*.*

2. ***Carrier****:* GammaDom *is the closed inductive predicate of Definition 75.8;* $\mathsf{Gamma}(s, z)$ *is the closed inductive predicate of Definition 75.19.*

3. ***Classifier****: limit-uniqueness up to* ComplexClassifier *inherited from* $\mathsf{CplxLim}^{\uparrow}$ *(chapter 65).*

4. ***Stability obligations****: factorial bridge at positive integers (Definition 75.15), Weierstrass Cauchy modulus (Definition 75.18), holomorphicity (Definition 75.20), recurrence (Definition 75.21),* $\Gamma(1) = 1$ *(Definition 75.22), reflection boundary certificate (Definition 75.23).*

5. ***Ledger****: explicit apartness precision* k_0 *for every domain witness; explicit Cauchy modulus* N *for every Weierstrass evaluation; the unary factorial table for integer points.*

6. ***Exactness predicate****:* $\mathsf{Gamma}(s, z)$ *holds exactly for* $s \in$ GammaDom*, with* z *determined uniquely up to* ComplexClassifier*.*

7. ***Standard bridge****: when classical complex Gamma is supplied externally (Euler integral or Bohr-Mollerup characterisation), the* ComplexClassifier*-equivalence with the constructive Weierstrass-product limit is recorded as the bridge slot.*

8. ***Status****:* ProofTarget*. Lean construction proceeds in* BEDC.Derived.GammaUp.$*$*; the holomorphic anchor is chapter 68 and the limit anchor is chapter 65.*

Proof. The pole-locus apartness of Definition 75.8 together with the Weierstrass partial product of Definition 75.17 supplies the constructive carrier. The Cauchy modulus of Definition 75.18 lifts to a complex limit via $\mathsf{CplxLim}^{\uparrow}$. The five stability obligations are listed as proof targets; each is a finite-step argument over the Weierstrass product representation. No Classical.choice, no Quot.sound, and no propext is required. □

Lean4 ✓ `BEDC.Derived.GammaUp.GammaDomainCore semanticNameCert`

Corollary 75.25 ($\mathsf{Gamma}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{Gamma}^{\uparrow}$ *is licensed by the constructive Weierstrass product on* $\mathsf{Complex}^{\uparrow}$ *over* $\mathsf{RealAnalytic}^{\uparrow}$ *and* $\mathsf{Prime}^{\uparrow}$*. It is not a primitive of the BEDC kernel.*

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.gamma function semantic name certificate`

Remark 75.26 (Schema-only horizon discharge)**.** *This chapter exports the constructive Gamma witness types, the Weierstrass-product carriers, and the explicit unary moduli consumed by the zeta functional-equation chapter chapter 9342.*

Theorem 75.27 (Gamma Weierstrass Cauchy modulus hsame transport)**.** *Let s and t be complex histories with $s\mathsf{hsame}t$. Suppose s carries a Gamma-domain core witness whose apartness datum has precision k_0, and set*

$$N_s(k) := \max(k_0, k) + |\mathsf{Re}(s)|^2 + |\mathsf{Im}(s)|^2.$$

Then t carries a Gamma-domain core witness with the same apartness datum. Let P^t be the unary-indexed Weierstrass partial-product family obtained from Definition 75.17 at t. The pair (P^t, N_s) satisfies $\mathsf{CplxRegSeq}(P^t, N_s)$ in the sense of Definition 65.18. Equivalently, for every unary precision k and all unary indices m, n accepted above $N_s(k)$, the complex distance from P^t_m to P^t_n is bounded by the unary representation of 2^{-k}.

Proof. Apply Definition 75.11 to the comparison $s\mathsf{hsame}t$. The transported witness keeps the original apartness datum, hence keeps the precision k_0, while replacing the complex carrier and pole-exclusion fields by their transports to t.

Define

$$N_t(k) := \max(k_0, k) + |\mathsf{Re}(t)|^2 + |\mathsf{Im}(t)|^2.$$

By Definition 75.18, the Weierstrass partial products at t form a regular complex Cauchy sequence with modulus N_t.

It remains only to replace N_t by N_s. The comparison $s\mathsf{hsame}t$, together with the product-classifier reading of complex histories in Definition 39.17, transports the real and imaginary readbacks of s to those of t. The absolute-value, squaring, unary maximum, and unary addition continuations used in the displayed modulus are classifier-respecting, so for every unary k there is a history-sameness comparison

$$N_s(k)\mathsf{hsame}N_t(k).$$

Now fix k and unary indices m, n accepted above $N_s(k)$ in the threshold clause of Definition 65.18. In the equality branch, compose the equality with $N_s(k)\mathsf{hsame}N_t(k)$. In the strict-prefix branch, unpack Definition 28.10 and transport the same nonempty unary tail across the endpoint comparison by Definition 28.24. Thus m and n are accepted above $N_t(k)$ as well. The regularity witness for (P^t, N_t) supplies the required complex-distance bound, and the distance history is unchanged. Since k, m, n were arbitrary, (P^t, N_s) is regular with the transported source modulus. □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.GammaWeierstrassCauchyModulus hsame transport`

Theorem 75.28 (Gamma limit certificate hsame transport)**.** *Let s, t, z be complex histories with $s\mathsf{hsame}t$. Suppose D_s is a $\mathsf{GammaDom}(s)$ core witness with apartness precision k_0, and suppose*

$\mathsf{Gamma}(s, z)$ *is witnessed in the sense of Definition 75.19 by the Weierstrass partial-product family* P^s *of Definition 75.17, the Cauchy modulus*

$$N_s(k) := \max(k_0, k) + |\mathsf{Re}(s)|^2 + |\mathsf{Im}(s)|^2,$$

and a complex-limit modulus M_s*. Then* $\mathsf{Gamma}(t, z)$ *holds with the Gamma-domain core witness transported from* D_s *by Definition 75.11, with the Weierstrass partial-product family* P^t *at* t*, with the same Cauchy modulus* N_s*, and with the same limit point* z*.*

Proof. Unpack the displayed Gamma witness by Definition 75.19. It consists of the domain witness D_s, the partial-product family P^s, the regularity supplied by Definition 75.18, and a limit row

$$\mathsf{CplxLim}(P^s, N_s, z, M_s)$$

in the sense of Definition 65.21.

Apply Definition 75.11 to $s\mathsf{hsame}t$ and D_s. This gives the target domain witness D_t with the same apartness datum and the same precision k_0. Let P^t be the partial-product family obtained from Definition 75.17 using D_t. By Definition 75.27, (P^t, N_s) is a regular complex sequence, so the target Cauchy modulus is the source modulus N_s.

For every unary index n, the partial products P^s_n and P^t_n are history-same. This is unary induction on n. In the base product, the constants are identical and the only source-dependent endpoint is transported by $s\mathsf{hsame}t$. In the successor step, the induction hypothesis compares the preceding products; the displayed source comparison transports the factors $1 + s/n$, their inverses, and $\exp(s \log(1+1/n))$ to the corresponding factors at t. The rational-pair complex operations are classifier-respecting by Definition 39.38, and the analytic exp and log rows are classifier-respecting in the construction of chapter 74. Continuation determinacy then identifies the two successor products.

The pointwise comparison $P^s_n \mathsf{hsame} P^t_n$ lets Definition 65.35 carry the source limit row to

$$\mathsf{CplxLim}(P^t, N_s, z, M_s).$$

If the target limit certificate is presented through another endpoint z_t, Definition 65.24 and Definition 65.48 classify z_t with z componentwise, and Definition 65.22 transports the limit row to the endpoint z. Repacking Definition 75.19 with D_t, P^t, N_s, M_s, and this limit row gives $\mathsf{Gamma}(t, z)$. □

Lean4 ✓ `BEDC.Derived.GammaFunctionUp.GammaLimitCertificate hsame transport`

Closure status for $\mathsf{NameCert}_{\mathsf{Gamma}^\uparrow}$.

Constructive story: A Gamma input begins as a $\mathsf{Complex}^\uparrow$ history that also carries constructive apartness from the negative-integer pole locus: the excluded histories are exactly those displayed as a non-positive integer pole by a unary witness, while the admitted histories retain the apartness precision that keeps them away from that locus. The value carrier is then built by unary-indexed Weierstrass partial products over that admitted input, with factorial rows on positive integer points and a $\mathsf{CplxLim}^\uparrow$ witness turning the regular product sequence into a target complex history; the classifier is limit uniqueness through the inherited ComplexClassifier, not an external meromorphic-function equality. The source covers Gamma-domain histories with their apartness data, the pattern records pole transport, factorial readback, Weierstrass factors, explicit Cauchy moduli, holomorphic-limit readback, and recurrence boundary rows, and the ledger keeps the apartness precision k_0, the unary modulus $N(k)$, the partial-product family, and the factorial table visible. Nonemptiness comes from a displayed domain core, the classifier laws come from the complex-limit and complex-history classifiers, and carrier closure follows because pole exclusions, partial products, and limit certificates transport along hsame. A history h lies in the signature gap of a $\mathsf{Gamma}^\uparrow$ token p when the source-domain accepts its Gamma-domain witness, the signature relation pairs the Weierstrass limit package with an answer complex history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is a non-pole complex input and the answer is the complex limit encoded by the token's partial-product certificate. The chapter registers this Weierstrass-product construction under the name $\mathsf{Gamma}^\uparrow$;

it introduces no primitive Gamma function, no meromorphic-continuation theorem, no reflection-formula equality, and no external Euler or Bohr-Mollerup bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{Gamma}\uparrow}$, scopedClosed)

Scope closed: This chapter closes exactly the Gamma-domain core, the Weierstrass partial-product carrier, the transported Cauchy modulus, the limit-certificate transport, and the semantic NameCert over ComplexUp histories.

Formal verification: FormalStatus(NameCert$_{\mathsf{Gamma}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.GammaFunctionUp.gamma\_function\_semantic\_name\_certificate`

Bridge status: none

Not claimed: It does not close meromorphic continuation, the reflection-formula equality, or an external Euler or Bohr-Mollerup bridge.

Upgrade path: The next grade requires checked recurrence, Gamma at 1, holomorphicity, and reflection boundary as exported public operations.

Chapter 76

A Concrete Naming Certificate for $\mathsf{CplxAnalytic}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the constructive complex transcendental package $\mathsf{CplxAnalytic}^{\uparrow}$ at the paper-proof level. The package extends the real-side $\mathsf{RealAnalytic}^{\uparrow}$ (chapter 74) to complex inputs by separating each complex history into its real and imaginary components and applying the real-side power series componentwise. The construction sits above $\mathsf{Complex}^{\uparrow}$ (chapter 39) for the rational-pair carrier, $\mathsf{RealAnalytic}^{\uparrow}$ for the underlying real $\exp / \log / \sin / \cos / \pi$, and $\mathsf{CplxLim}^{\uparrow}$ (chapter 65) for the limit packaging. All moduli are explicit unary-history total functions; the construction uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

76.1 Complex exponential via Euler decomposition

Definition 76.1 (Complex exponential by real components)**.** *For complex history z with components (x, y) (real and imaginary parts as $\mathsf{Real}^{\uparrow}$-histories), the complex exponential predicate $\mathsf{CplxExp}(z, w)$ holds when w is the complex history with components*

$$\mathsf{Re}(w) \;=\; \mathsf{Exp}(x) \cdot \mathsf{Cos}(y), \qquad \mathsf{Im}(w) \;=\; \mathsf{Exp}(x) \cdot \mathsf{Sin}(y),$$

each computed by composing $\mathsf{RealAnalytic}^{\uparrow}$ certificates (Definition 74.17, Definition 74.25) with $\mathsf{Real}^{\uparrow}$ multiplication. The witness records the explicit input bound $B = \max(|x|, |y|)$ inherited from the underlying real evaluations.

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxExp`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp component carrier witness`

Theorem 76.2 (Component continuation witness)**.** *Let z be the continuation of a rational real component by a rational imaginary component, and let zq be the continuation of z by a unary suffix q. Then there is an intermediate imaginary component $imagq$ such that $imagq$ continues the imaginary component by q, zq is the continuation of the real component by $imagq$, and $imagq$ has a positive unary denominator.*

Proof. Associativity of Cont splits the continuation through the imaginary component. The rational carrier on the imaginary component is closed under unary suffix continuation, and carrier transport along the resulting Cont equality gives the positive-denominator readback. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.ComplexAnalytic component continuation witness`

Theorem 76.3 (Component continuation has complex carrier)**.** *If rational real and imaginary component histories continue to a history z, then z is carried by* $\mathsf{Complex}^{\uparrow}$.

Proof. The two rational component carriers and the displayed continuation are exactly the product-history carrier data defining a complex history. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.ComplexAnalytic component continuation complex carrier`

Definition 76.4 (Pure-imaginary complex source)**.** *The predicate* $\mathsf{CplxPureImaginary}(\theta, z)$ *holds when θ is unary and z is* hsame *to the complex history with real component* $\mathsf{E1}(\mathsf{e})$ *and imaginary component* $\mathsf{E1}(\theta)$.

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary`

Theorem 76.5 (Pure-imaginary e1 tail iff)**.** *For histories θ and tail, a pure-imaginary witness at the visible endpoint* $\mathsf{E1}(tail)$ *is exactly a unary witness for θ together with the tail comparison against the component append:*

$$\mathsf{CplxPureImaginary}(\theta, \mathsf{E1}(tail)) \iff \mathsf{UnaryHistory}(\theta) \wedge \mathsf{hsame}(tail, \mathsf{append}(\mathsf{E1}(\mathsf{e}), \theta)).$$

Proof. Unfold the pure-imaginary predicate at the visible endpoint. The displayed append with imaginary component $\mathsf{E1}(\theta)$ is itself an $\mathsf{E1}$-history, so constructor inversion reads the endpoint comparison back to the tail. The converse applies $\mathsf{E1}$ congruence to the tail comparison and repacks the unary witness. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary e1 tail iff`

Theorem 76.6 (Pure-imaginary witness uniqueness)**.** *If the same complex history z is displayed as pure-imaginary by unary witnesses θ and ϕ, then* $\theta \mathsf{hsame} \phi$.

Proof. The two displayed pure-imaginary equalities identify the append histories $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\theta))$ and $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\phi))$. Cancelling the common left component $\mathsf{E1}(\mathsf{e})$ gives $\mathsf{E1}(\theta)\mathsf{hsame}\mathsf{E1}(\phi)$, and constructor inversion gives $\theta \mathsf{hsame} \phi$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary witness unique`

Theorem 76.7 (Pure-imaginary component determinacy)**.** *If two pure-imaginary witnesses present the same complex history z through unary parameters θ and θ', then the explicit component pairs* $(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\theta))$ *and* $(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\theta'))$ *are* hsame*-equal, and* $\theta \mathsf{hsame} \theta'$.

Proof. Unpack both pure-imaginary witnesses and compose the two hsame comparisons through the shared history z. Cancelling the common real component $\mathsf{E1}(\mathsf{e})$ leaves $\mathsf{E1}(\theta)\mathsf{hsame}\mathsf{E1}(\theta')$, and constructor inversion for $\mathsf{E1}$ reads back $\theta \mathsf{hsame} \theta'$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary component deterministic`

Lemma 76.8 (Complex exponential is well-defined up to ComplexClassifier)**.** *For every complex history z, the predicate* $\mathsf{CplxExp}(z, w)$ *determines w uniquely up to* $\mathsf{ComplexClassifier}$. *Moreover, if* $\mathsf{hsame}(z, z')$, *then* $\mathsf{CplxExp}(z, w)$ *iff* $\mathsf{CplxExp}(z', w)$.

Proof. Componentwise determination follows from $\mathsf{RealClassifier}$-uniqueness of Exp, Sin, Cos on $\mathsf{Real}^{\uparrow}$. Componentwise hsame-transport on z lifts through the real-side carrier transports inherited from $\mathsf{RealAnalytic}^{\uparrow}$ and through componentwise $\mathsf{Real}^{\uparrow}$ multiplication. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp component well defined`

Theorem 76.9 (Complex exponential additivity)**.** *For all complex histories z, z' with* $\mathsf{CplxExp}(z, w)$, $\mathsf{CplxExp}(z', w')$, $\mathsf{CplxExp}(z + z', w'')$, *the equality* $\mathsf{ComplexClassifier}(w'', w \cdot w')$ *holds, where $w \cdot w'$ is the* $\mathsf{Complex}^{\uparrow}$ *product.*

Proof. Decompose into components. The real component of $w \cdot w'$ is $\mathsf{Exp}(x)\mathsf{Cos}(y)\mathsf{Exp}(x')\mathsf{Cos}(y') - \mathsf{Exp}(x)\mathsf{Sin}(y)\mathsf{Exp}(x')\mathsf{Sin}(y')$; use Exp additivity (Definition 74.18) to combine the Exp factors, then use the $\cos(y + y')$ addition formula derivable from the Pythagorean identity (Definition 74.26) and the $\sin / \cos$ power series. The imaginary component proceeds analogously with the $\sin(y + y')$ addition formula. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp output hsame transport`

Theorem 76.10 (Euler identity at $i\theta$)**.** *For pure-imaginary $z = i\theta$ with real θ (i.e. component $(0, \theta)$),* $\mathsf{CplxExp}(z, w)$ *gives w with components* $(\mathsf{Cos}(\theta), \mathsf{Sin}(\theta))$. *Equivalently, $e^{i\theta} = \cos\theta + i \sin\theta$ up to* $\mathsf{ComplexClassifier}$.

Proof. Direct unfolding of Definition 76.1 at $x = 0$: $\mathsf{Exp}(0) = 1$ by $\mathsf{ExpPart}(0, n, \cdot)$ collapsing to the empty-product base, so the components reduce to $(\mathsf{Cos}(\theta), \mathsf{Sin}(\theta))$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp pure imaginary component witness`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp pure imaginary euler component package`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp pure imaginary euler component witness`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp euler pure imaginary carrier witness`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxExp pure imaginary source witness`

Theorem 76.11 (Pure-imaginary source has complex carrier witness)**.** *Every* $\mathsf{CplxPureImaginary}(\theta, z)$ *witness contains the unary witness for θ, the* hsame *comparison from z to the component pair $(0, \theta)$, and a* $\mathsf{Complex}^{\uparrow}$ *carrier for z.*

Proof. The real component $\mathsf{E1}(\mathsf{e})$ and imaginary component $\mathsf{E1}(\theta)$ both satisfy the rational positive-denominator carrier. Their product gives the complex carrier for the component pair, and the supplied hsame comparison transports that carrier to z. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary complex carrier witness`

Theorem 76.12 (Pure-imaginary empty endpoint contradiction)**.** *No pure-imaginary source can be displayed at the empty history endpoint.*

Proof. The pure-imaginary witness supplies a $\mathsf{Complex}^{\uparrow}$ carrier for the endpoint. At $z = \mathsf{e}$, this contradicts the non-empty boundary theorem for complex history carriers. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary empty absurd`

Theorem 76.13 (Pure-imaginary empty continuation contradiction)**.** *If* $\mathsf{CplxPureImaginary}(\theta, z)$ *holds and* $\mathsf{Cont}(z, q, \mathsf{e})$, *then False follows.*

Proof. The empty continuation endpoint splits by BEDC append-empty inversion, so the source z is hsame-equal to e. Transporting the pure-imaginary witness along this source equality gives a pure-imaginary witness at e, which is impossible by the empty endpoint contradiction theorem. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary empty continuation absurd`

Theorem 76.14 (Pure-imaginary hsame transport witness)**.** *If* $z\mathsf{hsame}z'$ *and* z *is the pure-imaginary source determined by* θ*, then* z' *is the same pure-imaginary source and exposes the transported comparison with the component pair* $(0, \theta)$*.*

Proof. Unpack the pure-imaginary witness into the unary carrier for θ and the comparison from z to the component-pair history. Compose that comparison with the symmetry of $z\mathsf{hsame}z'$. The same unary carrier and the composed comparison rebuild the transported pure-imaginary witness. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary hsame transport witness`

Theorem 76.15 (Pure-imaginary phase witness stability)**.** *For histories* $\theta, \phi, z : \mathsf{BHist}$*, if* $\mathsf{CplxPureImaginary}(\theta, z)$ *and* $\theta\mathsf{hsame}\phi$*, then* $\mathsf{CplxPureImaginary}(\phi, z)$*.*

Proof. Unpack the pure-imaginary witness for θ into the unary carrier of θ and the comparison from z to the component history $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\theta))$.

Transport the unary carrier along $\theta\mathsf{hsame}\phi$. Constructor congruence gives $\mathsf{E1}(\theta)\mathsf{hsame}\mathsf{E1}(\phi)$, while the real component $\mathsf{E1}(\mathsf{e})$ is unchanged. Applying append congruence to the fixed real component and the transported imaginary component gives the comparison between the two component-pair histories.

Compose that comparison with the original comparison from z to $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\theta))$. The transported unary carrier and the composed comparison repack the pure-imaginary witness for $\mathsf{CplxPureImaginary}(\phi, z)$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary phase stability witness`

Theorem 76.16 (Pure-imaginary component continuation witness)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, and let a unary suffix* q *continue* z *to* zq*. Then there is an intermediate imaginary component* $\mathsf{imag}q$ *that continues* $\mathsf{E1}(\theta)$ *by* q*, continues* $\mathsf{E1}(\mathsf{e})$ *to* zq*, and retains a positive unary denominator.*

Proof. The pure-imaginary witness supplies the component continuation from $\mathsf{E1}(\mathsf{e})$ and $\mathsf{E1}(\theta)$ to z. The real and imaginary components are rational carriers by the positive-denominator characterization, so the component continuation theorem splits the suffix through the imaginary component and returns the positive-denominator witness. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary component continuation witness`

Theorem 76.17 (Pure-imaginary suffix component tail witness)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, and let a unary suffix* q *continue* z *to* zq*. Then there is an intermediate imaginary component* $\mathsf{imag}q$ *which is a rational history carrier, is* hsame*-equal to* $\mathsf{E1}(\mathsf{append}(\theta, q))$*, continues* $\mathsf{E1}(\mathsf{e})$ *to* zq*, and has a positive unary denominator.*

Proof. Apply the pure-imaginary component continuation witness to obtain the intermediate imaginary component. Its continuation from $\mathsf{E1}(\theta)$ by the unary suffix q identifies it with $\mathsf{append}(\mathsf{E1}(\theta), q)$, and unary append readback identifies this history with $\mathsf{E1}(\mathsf{append}(\theta, q))$. The rational carrier, continuation ledger from $\mathsf{E1}(\mathsf{e})$, and positive-denominator witness are the corresponding fields of the component witness. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix component tail witness`

Theorem 76.18 (Pure-imaginary suffix parameter readback)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, let* q *be unary, and let* $\mathsf{Cont}(z, q, zq)$*. If* zq *is also displayed as* $\mathsf{CplxPureImaginary}(\psi, zq)$*, then* $\psi\mathsf{hsameappend}(\theta, q)$*.*

Proof. The suffix component-tail witness supplies an intermediate imaginary component $imagq$ with $imagq\mathsf{hsameE1}(\mathsf{append}(\theta, q))$ and a continuation ledger from $\mathsf{E1}(\mathsf{e})$ to zq. The pure-imaginary display of zq identifies the same result with $\mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\psi))$. Transport the continuation result to that display, cancel the fixed real component $\mathsf{E1}(\mathsf{e})$, and invert the $\mathsf{E1}$ constructor. The remaining equality is exactly $\psi\mathsf{hsameappend}(\theta, q)$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix parameter readback`

Theorem 76.19 (Pure-imaginary suffix continuation complex carrier)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, and let a unary suffix* q *continue* z *to* zq*. Then* zq *is a* $\mathsf{Complex}^{\uparrow}$ *history carrier and is not* hsame*-equal to* e*.*

Proof. Split the suffix continuation through the imaginary component supplied by the pure-imaginary component continuation theorem. The real component $\mathsf{E1}(\mathsf{e})$ is a rational carrier, and the split suffix gives a rational imaginary carrier, so the complex component-continuation carrier theorem builds the $\mathsf{Complex}^{\uparrow}$ carrier for zq. Complex carrier non-emptiness then rules out $zq\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix continuation complex carrier`

Theorem 76.20 (Pure-imaginary suffix continuation classifier)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *and* $\mathsf{CplxPureImaginary}(\phi, w)$ *hold with* $\theta\mathsf{hsame}\phi$*. If unary suffixes* q, q' *satisfy* $q\mathsf{hsame}q'$*,* q *continues* z *to* zq*, and* q' *continues* w *to* wq*, then* $\mathsf{ComplexHistoryClassifier}(zq, wq)$*.*

Proof. Split both suffix continuations through their imaginary components. The real component on both sides is the fixed rational carrier $\mathsf{E1}(\mathsf{e})$, so it classifies to itself. The imaginary base components $\mathsf{E1}(\theta)$ and $\mathsf{E1}(\phi)$ classify by rational one-tail exactness from the unary witnesses and $\theta\mathsf{hsame}\phi$. Close that imaginary classifier under the matched suffixes q and q', then transport it to the displayed split imaginary outputs. The component classifier introduction theorem combines the real and imaginary component classifiers through the two continuation ledgers, yielding the complex endpoint classifier. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix continuation classifier`

Theorem 76.21 (Pure-imaginary suffix tail deterministic)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *and* $\mathsf{CplxPureImaginary}(\phi, w$ *hold with* $\theta\mathsf{hsame}\phi$*. If* $\mathsf{Cont}(z, q, zq)$*,* $\mathsf{Cont}(w, q', wq)$*, and* $zq\mathsf{hsame}wq$*, then* $q\mathsf{hsame}q'$*.*

Proof. The two pure-imaginary witnesses identify z and w with the same anchored component pair. With the sources and results identified, continuation left cancellation determines the suffix histories. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix tail deterministic`

Theorem 76.22 (Pure-imaginary suffix source deterministic)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *and* $\mathsf{CplxPureImaginary}(\phi, w)$ *hold. If the same suffix* q *continues* z *and* w *to the same endpoint* zq*, then* $\theta\mathsf{hsame}\phi$*.*

Proof. Continuation right cancellation identifies the two sources z and w under the shared suffix and endpoint. Transport the pure-imaginary witness for ϕ from w to z, then apply pure-imaginary witness uniqueness at the common source. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary suffix source deterministic`

Theorem 76.23 (Pure-imaginary continuation has complex carrier)**.** *If z is the pure-imaginary source determined by θ, a unary suffix q continues z to zq, and q is a unary history, then zq is carried by* $\mathsf{Complex}^{\uparrow}$*.*

Proof. The component-continuation theorem splits the suffix through an intermediate imaginary component $imagq$, with a rational carrier for $imagq$ and a continuation from $\mathsf{E1}(\mathsf{e})$ and $imagq$ to zq. The fixed real component $\mathsf{E1}(\mathsf{e})$ is a rational carrier, so product continuation introduces the $\mathsf{Complex}^{\uparrow}$ carrier for zq. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary continuation complex carrier`

Theorem 76.24 (Pure-imaginary component continuation has complex carrier)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, and let a unary suffix q continue z to zq. Then zq is carried by* $\mathsf{Complex}^{\uparrow}$*.*

Proof. The pure-imaginary continuation witness splits the suffix through an intermediate imaginary component with a rational carrier. The real component $\mathsf{E1}(\mathsf{e})$ has the fixed rational carrier, and the displayed continuation from that real component and the continued imaginary component to zq forms the product-history carrier defining a complex history. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary component continuation complex carrier`

Theorem 76.25 (Pure-imaginary continuation excludes e0 result)**.** *Let* $\mathsf{CplxPureImaginary}(\theta, z)$ *hold, and let a unary suffix q continue z to* $\mathsf{E0}(t)$*. This is impossible.*

Proof. Continuation preserves the $\mathsf{Complex}^{\uparrow}$ carrier. Reading a complex carrier back to unary history would make $\mathsf{E0}(t)$ unary, contradicting unary constructor exclusion. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary continuation e0 result absurd`

76.2 Complex logarithm with principal branch

Definition 76.26 (Complex modulus and argument as real-side data)**.** *For complex history z with* $\mathsf{CplxNonZero}(z)$ *(Definition 67.1 of chapter 67), the modulus-argument decomposition* $\mathsf{ModArg}(z, r, \theta)$ *holds when r is the* $\mathsf{Real}^{\uparrow}$*-history modulus* $\sqrt{x^2 + y^2}$ *and θ is a* $\mathsf{Real}^{\uparrow}$*-history in* $(-\pi, \pi]$ *such that* $z = r \cdot (\mathsf{Cos}(\theta) + i\mathsf{Sin}(\theta))$*. Both r and θ are constructed by explicit bisection on the rational-pair representation, with the apartness witness from* $\mathsf{CplxNonZero}$ *guaranteeing termination.*

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxModArg`

Definition 76.27 (Principal complex logarithm)**.** *For complex history z with* $\mathsf{CplxNonZero}(z)$ *and* $\mathsf{ModArg}(z, r, \theta)$*, the principal complex logarithm predicate* $\mathsf{CplxLog}(z, w)$ *holds when w is the complex history with components* $(\mathsf{Log}(r), \theta)$*, where* Log *is the constructive real logarithm of Definition 74.19 (legitimate because $r > 0$ by* $\mathsf{CplxNonZero}$*).*

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxLog`

Theorem 76.28 (Complex log is the inverse of complex exp on the principal branch)**.** *For z with* $\mathsf{CplxNonZero}(z)$*,* $\mathsf{CplxLog}(z, w)$*, and* $\mathsf{CplxExp}(w, z')$*,* $\mathsf{ComplexClassifier}(z', z)$ *holds. Conversely, for w with imaginary part in* $(-\pi, \pi]$*,* $\mathsf{CplxExp}(w, z)$ *and* $\mathsf{CplxLog}(z, w')$ *gives* $\mathsf{ComplexClassifier}(w', w)$*.*

Proof. Forward: write $\mathsf{CplxExp}(\mathsf{CplxLog}(z))$ using Definition 76.1 and Definition 76.26; the components reproduce $(r\cos\theta, r\sin\theta)$ which is z by the modulus-argument decomposition.

Reverse: $\mathsf{CplxExp}(w)$ has modulus $\mathsf{Exp}(\mathsf{Re}(w))$ and argument $\mathsf{Im}(w)$ (mod 2π); the principal-branch restriction $\mathsf{Im}(w) \in (-\pi, \pi]$ makes the inversion unique. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxLog exp inverse carrier witness`

Remark 76.29 (Branch-cut horizon)**.** *The principal-branch restriction is a SCHEMA-ONLY HORIZON for general analytic continuation: BEDC can decide whether a given θ-witness is the principal value (decidable on rational representations), but cannot in general continue* CplxLog *across the negative-real-axis branch cut without external chain data. This is consistent with the branch-obstruction interface flagged in chapter 9341.*

76.3 Complex sine and cosine

Definition 76.30 (Complex sine and cosine via Euler)**.** *For complex history z, the predicates* $\mathsf{CplxSin}(z, s)$ *and* $\mathsf{CplxCos}(z, c)$ *are defined by Euler's formula:*

$$s = \frac{\mathsf{CplxExp}(iz) - \mathsf{CplxExp}(-iz)}{2i}, \qquad c = \frac{\mathsf{CplxExp}(iz) + \mathsf{CplxExp}(-iz)}{2}.$$

Both are computed using Definition 76.1 and the $\mathsf{Complex}^{\uparrow}$ *field operations.*

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxSinCos`

Theorem 76.31 (Pythagorean identity on $\mathsf{Complex}^{\uparrow}$)**.** *For all z with* $\mathsf{CplxSin}(z, s)$ *and* $\mathsf{CplxCos}(z, c)$*,* $\mathsf{ComplexClassifier}(s^2 + c^2, \mathsf{E1}(\mathsf{e}))$ *holds.*

Proof. Use Definition 76.30: $s^2+c^2 = \frac{1}{4}\left[\mathsf{CplxExp}(iz)^2 + 2 + \mathsf{CplxExp}(-iz)^2 - \mathsf{CplxExp}(iz)^2 + 2 - \mathsf{CplxExp}(-iz)^2\right] /$
1, where the algebraic cancellation uses Definition 76.9 and $\mathsf{CplxExp}(0) = 1$. □

Theorem 76.32 (Complex sine vanishes at integer multiples of π)**.** *For every signed unary integer n,* $\mathsf{CplxSin}(n \cdot \mathsf{Pi}, s)$ *gives* $\mathsf{ComplexClassifier}(s, \mathsf{e}_{\mathsf{Complex}^{\uparrow}})$*, where* $\mathsf{e}_{\mathsf{Complex}^{\uparrow}}$ *is the complex zero. In particular,* $\sin(\pi s) = 0$ *at integer s supplies the Weierstrass-product zero locus used in Definition 75.23.*

Proof. Real-imaginary projection: at $z = n\pi$ (real, imaginary part zero), Definition 76.30 reduces to the real-side $\mathsf{Sin}(n\pi)$, which is 0 by Definition 74.27. □

76.4 Complex powers $z \mapsto a^z$ and n^{-s}

Definition 76.33 (Complex power)**.** *For complex histories a and s with* $\mathsf{CplxNonZero}(a)$*, the complex power predicate* $\mathsf{CplxPow}(a, s, w)$ *holds when w is the complex history obtained by composition* $w = \mathsf{CplxExp}(s \cdot \mathsf{CplxLog}(a))$*. The witness inherits the principal-branch restriction from* CplxLog*.*

Lean4 (definition) `BEDC.Derived.ComplexAnalyticUp.CplxPower`

Theorem 76.34 (Dirichlet-term factor n^{-s} is well-defined)**.** *For every positive unary natural n and complex history s,* $\mathsf{CplxPow}(n, -s, w)$ *gives the Dirichlet term factor n^{-s} used in chapter 70, with explicit modulus determined by* $\mathsf{Log}(n)$ *(Definition 74.22).*

Proof. The principal logarithm of a positive real n is $\mathsf{Log}(n)$ on the real axis with imaginary part 0, supplied by Definition 74.22. The composition with complex multiplication and CplxExp gives the explicit Dirichlet factor. □

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPower dirichlet factor witness`
Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPower dirichlet factor classifier`

76.5 Certificate target

Theorem 76.35 (Concrete CplxAnalytic$^\uparrow$ naming certificate)**.** *The tuple* (CplxExp, CplxLog, CplxSin, CplxCos, CplxPo *with sources, classifiers, stability obligations, ledger, and standard bridge as listed below, forms a paper-side* NameCert(CplxAnalytic$^\uparrow$) *over* Complex$^\uparrow$*:*

1. ***Source**: Complex$^\uparrow$-histories, with* CplxNonZero *apartness witnesses required by* CplxLog, ModArg, *and* CplxPow*.*
2. ***Carrier**: each predicate of Sections 76.1–76.4 is a closed inductive predicate on* Complex$^\uparrow$*-histories, defined by composition over real-side* RealAnalytic$^\uparrow$ *carriers.*
3. ***Classifier**:* ComplexClassifier *inherited componentwise from* RealClassifier *on the real and imaginary parts.*
4. ***Stability obligations**: well-definedness of* CplxExp *(Definition 76.8), exponential additivity (Definition 76.9), Euler identity (Definition 76.10),* log-exp *inverse on principal branch (Definition 76.28), Pythagorean identity (Definition 76.31),* sin-π*-multiple zeros (Definition 76.32).*
5. ***Ledger**: explicit input bound B and Cauchy modulus N inherited from* RealAnalytic$^\uparrow$*; explicit principal-branch witness $\theta \in (-\pi, \pi]$ for every* CplxLog *evaluation; explicit apartness witness for every* CplxNonZero*-required input.*
6. ***Standard bridge**: when classical complex* exp, log, sin, cos *are supplied externally, the* ComplexClassifier*-equivalence with the constructive limits is recorded as the bridge slot.*
7. ***Status**:* ProofTarget*. Lean construction proceeds in* BEDC.Derived.ComplexAnalyticUp.$*$*; the underlying real-side anchor is chapter 74 and the limit anchor is chapter 65.*

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.CplxPureImaginary name certificate`

Proof. Componentwise composition with the supplied RealAnalytic$^\uparrow$ certificate-boundary rows of Definition 74.29 and the Complex$^\uparrow$ field operations gives the closed inductive carrier and classifier. The five stability obligations reduce to real-side identities or to algebraic manipulations of complex products and Euler's formula. No Classical.choice, no Quot.sound, and no propext is required. □

Corollary 76.36 (CplxAnalytic$^\uparrow$ is licensed, not primitive)**.** CplxAnalytic$^\uparrow$ *is licensed by componentwise composition of* RealAnalytic$^\uparrow$ *with* Complex$^\uparrow$ *field operations and the modulus-argument decomposition for principal logarithm. It is not a primitive of the BEDC kernel.*

Lean4 ✓ `BEDC.Derived.ComplexAnalyticUp.complex analytic licensed not primitive`

Remark 76.37 (Discharge of Gamma chapter forward references)**.** *The complex* exp *and* log *used by Definition 75.17 (the Weierstrass partial product for Gamma) are now anchored at the* CplxAnalytic$^\uparrow$ *certificate of this chapter. The Dirichlet-term factor n^{-s} used by chapter 70 is similarly anchored at Definition 76.34.*

Closure status for $\mathsf{NameCert}_{\mathsf{CplxAnalytic}^{\uparrow}}$.

Constructive story: A $\mathsf{CplxAnalytic}^{\uparrow}$ packet is generated from finite BHist rows for complex arithmetic, power-series windows, exponential and logarithm readbacks, Dirichlet power factors, analytic operation ledgers, transports, continuations, provenance, and local naming. Analytic functions are introduced by displayed histories and finite convergence or operation ledgers, with Cont replaying how exp, log, powers, and derived operations are read by later chapters. Componentwise hsame preserves the complex-history inputs and operation rows, while Pkg records the complex, real, series, and topology dependencies. The packet anchors analytic operations without importing a host holomorphic-function object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CplxAnalytic}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the complex analytic operation rows, exp and log anchors, Dirichlet power factor, Gamma forward-reference discharge, and associated NameCert surface for $\mathsf{CplxAnalytic}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CplxAnalytic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim a full host complex-analysis library, unrestricted analytic continuation, quotient equality of functions, classical branch selection, or external holomorphic-function spaces.

Upgrade path: Obligation closure requires Lean encodings of the analytic operation carriers, exp and log readbacks, Dirichlet power route, and operation stability theorems.

Chapter 77

A Concrete Naming Certificate for $\mathsf{CplxTopo}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for the complex topology interface $\mathsf{CplxTopo}^{\uparrow}$ at the paper-proof level. The interface promotes the inline open-disk predicate of chapter 68 to a stand-alone licensed object and packages the open-set, closed-set, compact-subset, and density-witness predicates that the holomorphy, contour-integration, and analytic-continuation chapters consume. The construction sits above $\mathsf{Complex}^{\uparrow}$ (chapter 39) for the rational-pair carrier and $\mathsf{RealAnalytic}^{\uparrow}$ (chapter 74) for the radius and modulus arithmetic. Every set-membership predicate is decidable on rational representations through NatUnaryStrictPrefix (chapter 28); compactness is encoded by explicit dyadic ε-net data, in line with the constructive-compactness convention of the L10 sub-roadmap. The construction uses no Classical.choice, no Quot.sound, and no propext, so the certificate's Lean targets fall under the project's axiom-purity gate.

77.1 Open disks

Definition 77.1 (Open disk)**.** *For complex history z_0, unary radius r, and complex history z, the open-disk membership predicate* $\mathsf{OpenDisk}(z_0, r, z)$ *holds when the rational representation of the complex modulus $|z - z_0|$ is strictly less than r under* NatUnaryStrictPrefix*. The witness is an explicit unary gap* $\mathsf{Gap}(z_0, r, z) > \mathsf{e}$*.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap`

Theorem 77.2 (Open-disk gap determinacy)**.** *If a complex-topology open-disk gap witness exposes* $\mathsf{Cont}(z, g, r)$ *and another boundary ledger exposes* $\mathsf{Cont}(z, g', r)$ *with the same point and radius, then* $g\mathsf{hsame}g'$*.*

Proof. Unpack the concrete open-disk gap witness to recover its boundary continuation ledger. Continuation left cancellation against the alternative ledger compares the two gap histories. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap gap deterministic`

Remark 77.3 (Compatibility with the inline definition)**.** *This definition coincides with the inline* `BEDC.Derived.HolomorphicUp.OpenDisk` *of chapter 68 up to namespace; promoting it to* $\mathsf{CplxTopo}^{\uparrow}$ *lets downstream chapters cite a single canonical home.*

Theorem 77.4 (Open-disk point-radius suffix closure)**.** *If* $\mathsf{OpenDisk}(z_0, r, z)$ *holds and a unary suffix extra continues both* r *to* r' *and* z *to* z'*, then* $\mathsf{OpenDisk}(z_0, r', z')$ *holds. The same gap witness continues* z' *to* r'*, while its original ledger continues* z *to* r*.*

Proof. Unpack the witnessed open disk. Unary closure gives the shifted radius, complex history closure gives the shifted point, and associativity of Cont rewrites the original gap ledger after the common unary suffix. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDisk point radius suffix closed`

Definition 77.5 (Closed disk)**.** $\mathsf{ClosedDisk}(z_0, r, z)$ *holds when* $|z - z_0| \leq r$ *on rational representations, decidable by non-strict prefix comparison.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap`

Lemma 77.6 (Closed-disk gap transport along hsame)**.** *If a complex-topology closed-disk gap witness is transported by the complex history classifier at its center and point, and by* hsame *at its radius and gap histories, then the transported tuple is a closed-disk gap witness. Its boundary evidence is transported as either a continuation ledger or boundary* hsame*.*

Proof. Unpack the closed-disk gap witness. The complex classifiers provide the target complex carriers for the center and point. Unary transport moves the radius and gap carriers. In the strict-boundary branch, continuation transport moves $\mathsf{Cont}(z, g, r)$ to $\mathsf{Cont}(z', g', r')$; in the boundary-equality branch, transitivity of hsame carries z'hsamer'. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap hsame transport`

Lemma 77.7 (Closed-disk strict gap radius is nonempty)**.** *If a complex-topology closed-disk gap witness is used through its strict continuation branch* $\mathsf{Cont}(point, gap, radius)$*, then the radius cannot be* hsame*-empty.*

Proof. Unpack the closed-disk gap witness to obtain the complex-history carrier for the point. Transport the strict continuation result along *radius*hsamee, invert the empty continuation result, and contradict the nonempty property of the point carrier. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap strict radius not empty`

Theorem 77.8 (Closed-disk gap radius is nonempty)**.** *Every complex-topology closed-disk gap witness has nonempty radius.*

Proof. Unpack the closed-disk witness and split the boundary evidence. In the strict continuation branch, transport the continuation result along *radius*hsamee and invert the empty result, contradicting nonemptiness of the point carrier. In the boundary equality branch, transitivity with *radius*hsamee would make the point empty, again contradicting its complex history carrier. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap radius not empty`

Theorem 77.9 (Closed-disk empty gap identifies radius and point)**.** *For a complex-topology closed-disk gap witness, if the gap history is* hsame*-empty, then the radius history is* hsame*-equal to the point history.*

Proof. Unpack the closed-disk gap witness. In the strict-boundary branch, transport the continuation ledger along the empty-gap identification and use the right-unit law for Cont. In the boundary branch, reverse the stored point-radius hsame witness. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap empty gap radius point`

Theorem 77.10 (Closed-disk visible radius boundary cases)**.** *If a complex-topology closed-disk gap witness has visible radius* $\mathsf{E1}(r)$*, then its strict boundary evidence either has empty gap with the point on that visible radius endpoint or descends to a closed-disk gap witness over* r*; the non-strict boundary branch remains the endpoint equality* $z\mathsf{hsameE1}(r)$*.*

Proof. Unpack the closed-disk gap witness. In the strict branch, apply visible-result inversion to $\mathsf{Cont}(z, g, \mathsf{E1}(r))$. The empty-gap branch gives $g = \mathsf{e}$ and $z\mathsf{hsameE1}(r)$. The visible-gap branch writes $g = \mathsf{E1}(g')$; unary constructor inversion supplies unary carriers for r and g', and the descended continuation repacks the closed-disk witness over r. In the boundary-equality branch, keep the displayed hsame evidence. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyClosedDiskGap visible radius boundary cases`

Theorem 77.11 (Open-disk gap modulus)**.** *For every triple* (z_0, r, z) *with* $\mathsf{OpenDisk}(z_0, r, z)$*, the gap function* $\mathsf{Gap}(z_0, r, z)$ *is a unary history giving an explicit positive distance from* z *to the boundary* $\partial\mathsf{ClosedDisk}(z_0, r)$*.*

Proof. Compute $\mathsf{Gap}(z_0, r, z) := r \ominus |z - z_0|$ using $\mathsf{Real}^{\uparrow}$ truncated subtraction. The strict inequality of Definition 77.1 guarantees the result is non-empty unary. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDisk gap modulus`

Lemma 77.12 (Open-disk gap transport along hsame)**.** *If a complex-topology open-disk gap witness is transported by the complex history classifier at its center and point, and by* hsame *at its radius and gap histories, then the transported tuple is again an open-disk gap witness and exposes the transported boundary ledger.*

Proof. Unpack the open-disk gap witness. The complex classifiers provide the target complex carriers for the center and point. Unary transport moves the radius and gap carriers along hsame, and continuation transport moves $\mathsf{Cont}(z, g, r)$ to $\mathsf{Cont}(z', g', r')$. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap hsame transport`

Theorem 77.13 (Open-disk gap radius extension)**.** *If a complex-topology open-disk gap witness has radius* r*, point* z*, gap* g*, and* $z \cdot g = r$*, then any common unary extension extra of the radius and gap gives a new witness with radius* r'*, gap* g'*, and* $\mathsf{Cont}(z, g', r')$*.*

Proof. Unpack the open-disk gap witness. Unary continuation closure gives the extended radius and gap carriers. The boundary ledger follows by eliminating the two continuation witnesses and reassociating the original point–gap continuation. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap radius extension closed`

Theorem 77.14 (Open-disk gap radius is non-empty)**.** *If a complex-topology open-disk gap witness has radius* r*, then* r *cannot be* hsame *to* e*.*

Proof. Transport the point–gap continuation ledger along a hypothetical $\mathsf{hsame}(r, \mathsf{e})$. Empty-result inversion for continuation forces the point history to be empty, contradicting the complex carrier carried by the open-disk gap witness. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap radius not empty`

Theorem 77.15 (Open-disk empty gap identifies radius and point)**.** *If a complex-topology open-disk gap witness has gap g with $g\mathsf{hsamee}$, then its radius and point histories are* hsame*.*

Proof. Unpack the gap witness to obtain the boundary ledger $\mathsf{Cont}(z, g, r)$. Transport this continuation along $g\mathsf{hsamee}$, keeping the point and radius endpoints fixed. The right-unit characterization of Cont then identifies the radius with the point. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap empty gap radius point`

Lemma 77.16 (Open-disk empty gap radius-point equivalence)**.** *For a complex-topology open-disk gap witness, the gap is* hsame *to* e *exactly when the radius and point histories are* hsame*.*

Proof. The forward direction is the empty-gap identification theorem above. For the reverse direction, unpack the boundary ledger $\mathsf{Cont}(z, g, r)$, transport its result along $r\mathsf{hsame}z$, and compare the resulting $\mathsf{Cont}(z, g, z)$ with the right-unit continuation $\mathsf{Cont}(z, \mathsf{e}, z)$ by left cancellation. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap empty gap radius point iff`

Theorem 77.17 (Open-disk visible radius gap cases)**.** *If a complex-topology open-disk gap witness has visible radius* $\mathsf{E1}(r)$*, then either the gap is empty and the point is history-same to that visible radius endpoint, or the gap is itself visible and its tail is an open-disk gap witness over r.*

Proof. Unpack the open-disk gap witness to obtain $\mathsf{Cont}(z, g, \mathsf{E1}(r))$. Visible result inversion for continuation gives either $g = \mathsf{e}$ with $z\mathsf{hsame}\mathsf{E1}(r)$, or a tail g' with $g = \mathsf{E1}(g')$ and $\mathsf{Cont}(z, g', r)$. Unary constructor inversion on the radius and gap carriers supplies the carriers for r and g', so the second branch repacks the descended open-disk gap witness. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap visible radius gap cases`

Theorem 77.18 (Open-disk two visible-radius descent)**.** *If a concrete complex-topology open-disk gap witness has visible radius* $\mathsf{E1}(\mathsf{E1}(r))$*, then either the gap is empty and the point is* hsame *to* $\mathsf{E1}(\mathsf{E1}(r))$*, or the gap is visible with tail g_0 and the descended witness over* $\mathsf{E1}(r)$ *further splits into the empty-tail endpoint case, where the point is* hsame *to* $\mathsf{E1}(r)$*, or a visible tail g_1 carrying an open-disk gap witness over r.*

Proof. Apply Theorem 77.17 to the displayed open-disk witness with radius $\mathsf{E1}(\mathsf{E1}(r))$. Its empty-gap branch gives the first alternative directly, with the point identified with the displayed radius endpoint.

In the visible-gap branch, write the exposed gap as $\mathsf{E1}(g_0)$ and keep the descended open-disk gap witness over $\mathsf{E1}(r)$. Apply Theorem 77.17 again to that descended witness. The empty branch yields the endpoint case $z\mathsf{hsame}\mathsf{E1}(r)$, while the visible branch writes g_0 as $\mathsf{E1}(g_1)$ and supplies the open-disk gap witness over r. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap visible radius double descent`

Theorem 77.19 (Open-disk gap unary suffix transport)**.** *If a complex-topology open-disk gap witness has boundary ledger* $\mathsf{Cont}(z, g, r)$*, and a unary suffix q continues both z to zq and r to rq, then the shifted point zq and shifted radius rq form another open-disk gap witness with the same center and gap, and expose* $\mathsf{Cont}(zq, g, rq)$*.*

Proof. Unpack the witness. The point carrier is closed under unary suffix continuation, the radius carrier is closed under unary continuation, and the boundary ledger follows by reassociating Cont after commuting the unary gap and suffix histories. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap unary suffix transport`

Theorem 77.20 (Open-disk gap center-point unary suffix transport)**.** *If a complex-topology open-disk gap witness has center z_0, point z, radius r, and boundary ledger* $\mathsf{Cont}(z, g, r)$*, and a unary suffix q continues z_0, z, and r to z_{0q}, z_q, and r_q, then (z_{0q}, r_q, z_q, g) is again an open-disk gap witness and exposes* $\mathsf{Cont}(z_q, g, r_q)$*.*

Proof. The preceding unary suffix theorem shifts the point and radius while retaining the gap ledger. Complex carrier closure under the same unary suffix shifts the center carrier from z_0 to z_{0q}, so the transported center, shifted radius, shifted point, and original gap repack the open-disk gap witness. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap center point unary suffix transport`

77.2 Open sets and finite intersections

Definition 77.21 (Open-set membership)**.** *A complex open set is encoded as a closed inductive predicate $U : \mathsf{Complex}^{\uparrow}$-history $\to \mathsf{Prop}$ generated by finite unions and finite intersections of open-disk predicates.* $\mathsf{IsOpen}(U)$ *holds when every membership witness $U(z)$ supplies an explicit interior open-disk* $\mathsf{OpenDisk}(z, r_z, z)$ *of positive unary radius r_z, with r_z given by an explicit total function on the membership-witness data.*

Theorem 77.22 (Finite intersection of open sets is open)**.** *If* $\mathsf{IsOpen}(U_1)$ *and* $\mathsf{IsOpen}(U_2)$*, then* $\mathsf{IsOpen}(U_1 \cap U_2)$ *with explicit interior radius $r_{12}(z) := \min(r_1(z), r_2(z))$.*

Proof. Combine the two interior-disk witnesses by taking the minimum unary radius pointwise; the minimum is decidable on unary representations. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenSet intersection witness closed`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenSet intersection radius witnesses nonempty`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenSet intersection witnesses`

Theorem 77.23 (Arbitrary union of open sets is open)**.** *For any classifier-respecting family $\{U_i\}_{i \in I}$ of open sets indexed by an inductive carrier I, the union $\bigcup_i U_i$ is open with interior radius inherited from the witnessing branch.*

Proof. Membership in the union is by case analysis on I; each case provides the interior-disk witness from the corresponding U_i. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenSet union closed`

77.3 Compact subsets via dyadic ε-nets

Definition 77.24 (Dyadic ε-net)**.** *For a complex-set predicate K and unary precision k, a dyadic 2^{-k}-net is a finite list of complex histories $[z_1, \ldots, z_N]$ such that for every z with $K(z)$, there exists an index $i \leq N$ with* $\mathsf{OpenDisk}(z_i, 2^{-k}, z)$*. The net function* $\mathsf{Net}_K : \mathsf{Unary} \to \mathsf{List}(\mathsf{Complex}^{\uparrow})$ *is required to be a total explicit function on the precision domain.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyDyadicNet`

Definition 77.25 (Compact subset)**.** $\mathsf{IsCompact}(K)$ *holds when K is bounded (contained in some* $\mathsf{ClosedDisk}(\mathsf{e}, R)$ *for explicit unary R) and possesses an explicit dyadic 2^{-k}-net for every unary precision k (i.e. a witnessing total function* $\mathsf{Net}_K : \mathsf{Unary} \to \mathsf{List}(\mathsf{Complex}^{\uparrow})$*).*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCompact`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCompact precision radius not empty`

Theorem 77.26 (Closed disk is compact)**.** *For every z_0 and unary r,* $\mathsf{IsCompact}(\mathsf{ClosedDisk}(z_0, r))$ *holds with explicit dyadic ε-net obtained by tiling the disk with rational-grid points at spacing 2^{-k}.*

Proof. At precision k, enumerate the rational pairs (p, q) with $|p - \mathsf{Re}(z_0)| \leq r$ and $|q - \mathsf{Im}(z_0)| \leq r$ on the dyadic-rational grid of mesh $2^{-k}/2$; the resulting finite list witnesses every disk point within distance $2^{-k}/\sqrt{2} \leq 2^{-k}$. The list construction is by unary recursion over the grid bounds, which are given explicitly by R and k. □

Theorem 77.27 (Compact subsets of half-planes)**.** *For unary parameters σ_a, ϵ, T, the compact sub-half-plane*

$$K(\sigma_a, \epsilon, T) \;:=\; \{s : \mathsf{Re}(s) \geq \sigma_a + \epsilon, |\mathsf{Im}(s)| \leq T\}$$

is compact in the sense of Definition 77.25, with explicit dyadic ε-nets given by tiling the rectangle $[\sigma_a + \epsilon, R(\sigma_a, T)] \times [-T, T]$ for explicit $R(\sigma_a, T)$.

Proof. The half-plane predicate is a finite intersection of closed half-planes, each compact on bounded restriction. The bound $R(\sigma_a, T)$ is taken to enclose the witnessing region needed by the consuming chapter (typically the half-plane $\Re(s) > 1$ of the Dirichlet-series majorants in chapter 70). □

77.4 Density witnesses and clustering points

Definition 77.28 (Quantitative density witness)**.** *For a center $z_c \in \mathsf{Complex}^{\uparrow}$ and an open set U with $U(z_c)$, a quantitative density witness inside U is an explicit sequence* $\mathsf{Cluster} : \mathsf{Unary} \to \mathsf{Complex}^{\uparrow}$ *of distinct points $(z_n)_n$ with $U(z_n)$, $z_n \neq z_c$, and $|z_n - z_c| \leq 2^{-n}$ for every unary n. The witness is a total explicit function on the precision domain.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyDensityWitness`

Theorem 77.29 (Open disks contain quantitative cluster sequences)**.** *For every z_0 and unary $r > 0$, the open disk* $\mathsf{OpenDisk}(z_0, r)$ *admits a quantitative density witness centered at z_0: the sequence $z_n := z_0 + 2^{-n}r/2$ is in the disk, distinct from z_0, and within distance $2^{-n}r/2 \leq 2^{-n}$ for $n \geq 1$.*

Proof. Construct the sequence by explicit unary recursion. The membership condition holds because the point lies at distance $2^{-n}r/2 < r$ from z_0; distinctness holds because the offset is non-empty unary for each n. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap density witness with boundary`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap constant density witness`

Remark 77.30 (Clustering for the constructive identity theorem)**.** *The quantitative density witness of Definition 77.28 is the precise data required by the constructive identity theorem of chapter 68: two holomorphic functions agreeing on an explicit cluster sequence agree on a smaller open disk. This chapter supplies the witness type that the holomorphy chapter consumes.*

77.5 Domain compatibility for analytic continuation

Definition 77.31 (Domain compatibility)**.** *For two open sets D_1, D_2 and an overlap region U,* $\mathsf{DomCompat}(D_1, D_2, U)$ *holds when* $\mathsf{IsOpen}(U)$*, $U \subseteq D_1 \cap D_2$ with explicit intersection-membership witness, and U contains an open disk* $\mathsf{OpenDisk}(z_c, r_c)$ *with quantitative density witness* $\mathsf{Cluster}(z_c, U)$ *of Definition 77.28.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyDomainCompat`

Remark 77.32 (Domain compatibility consumed by analytic continuation)**.** *chapter 9341 requires exactly this data to invoke the constructive identity theorem when extending a holomorphic function across overlapping domains; the present definition fixes the schema so the analytic-continuation chapter cites a single canonical interface.*

77.6 Certificate target

Theorem 77.33 (Concrete $\mathsf{CplxTopo}^{\uparrow}$ naming certificate)**.** *The tuple* (OpenDisk, ClosedDisk, IsOpen, IsCompact, Clu *with sources, classifiers, stability obligations, ledger, and standard bridge as listed below, forms a paper-side* $\mathsf{NameCert}(\mathsf{CplxTopo}^{\uparrow})$ *over* $\mathsf{Complex}^{\uparrow}$*:*

1. ***Source**:* $\mathsf{Complex}^{\uparrow}$*-histories with rational-pair carrier representations.*
2. ***Carrier**: each predicate of Sections 77.1–77.5 is a closed inductive predicate on* $\mathsf{Complex}^{\uparrow}$*-histories.*
3. ***Classifier**:* ComplexClassifier*-equivariance for set membership inherited from* hsame*-transport on the underlying complex carriers.*
4. ***Stability obligations**: open-disk gap modulus (Definition 77.11), finite-intersection closure (Definition 77.22), arbitrary-union closure (Definition 77.23), closed-disk compactness (Definition 77.26), half-plane compactness (Definition 77.27), open-disk clustering (Definition 77.29).*
5. ***Ledger**: explicit interior-radius function for every open-set witness; explicit dyadic ε-net function for every compact-set witness; explicit cluster sequence for every density witness.*
6. ***Standard bridge**: when classical complex topology (Heine-Borel compactness, open covers) is supplied externally, the explicit-net witnesses interpret as classical compactness witnesses on bounded subsets only; the bridge slot records this restricted correspondence.*
7. ***Status**:* ProofTarget*. Lean construction proceeds in* BEDC.Derived.ComplexTopologyUp.$*$*. The inline* OpenDisk *of chapter 68 is to be re-exported from* BEDC.Derived.ComplexTopologyUp *as the canonical home; the holomorphic chapter retains a re-export alias for backward compatibility with existing markers.*

Proof. Each carrier predicate is closed inductive over $\mathsf{Complex}^{\uparrow}$-histories with decidable membership on rational representations. The six stability obligations are the section-by-section theorems above, each established by explicit construction of the witness data (gap modulus, dyadic net, cluster sequence). The bridge slot is the standard interpretation under the bounded-subset restriction. No Classical.choice, no Quot.sound, and no propext is required. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap semanticNameCert`

Corollary 77.34 ($\mathsf{CplxTopo}^{\uparrow}$ is licensed, not primitive)**.** $\mathsf{CplxTopo}^{\uparrow}$ *is licensed by closed inductive predicates over rational-pair representations of complex histories, with constructive compactness encoded as explicit dyadic ε-nets. It is not a primitive of the BEDC kernel, and in particular does not assume Heine-Borel compactness as a general theorem.*

Remark 77.35 (Discharge of holomorphy/contour/analytic-continuation forward references)**.** *The open-disk, compact-half-plane, cluster-sequence, and domain-compatibility witnesses inline-defined or implicitly used in chapter 68, chapter 9340, chapter 9341, and chapter 71 are now anchored at the present* $\mathsf{CplxTopo}^{\uparrow}$ *certificate. Future restructuring should re-export the inline* OpenDisk *from* BEDC.Derived.HolomorphicUp *as an alias of the canonical* BEDC.Derived.ComplexTopologyUp.OpenDisk.

Definition 77.36 (Certified subdisk radius)**.** *For unary radii r' and r, a certified subdisk radius from r' to r is a unary history e together with a radius-continuation ledger* $\mathsf{Cont}(r', e, r)$*. We write this datum as* $\mathsf{Subradius}(r', r; e)$*. The empty tail is the equality case.*

Lean4 (definition) `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCertifiedSubradius`

Lemma 77.37 (Certified subdisk radii compose)**.** *If* $\mathsf{Subradius}(r_0, r_1; e_0)$ *and* $\mathsf{Subradius}(r_1, r_2; e_1)$*, then* $\mathsf{Subradius}(r_0, r_2; e_0 \cdot e_1)$*.*

Proof. Unpack the two subradius certificates. Unary append closure makes $e_0 \cdot e_1$ unary. Associativity of Cont composes $\mathsf{Cont}(r_0, e_0, r_1)$ with $\mathsf{Cont}(r_1, e_1, r_2)$ and gives $\mathsf{Cont}(r_0, e_0 \cdot e_1, r_2)$. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCertifiedSubradius comp`

Theorem 77.38 (HoloOnDisk subdisk restriction)**.** *Let $f : \mathsf{Complex}^{\uparrow} \to \mathsf{Complex}^{\uparrow}$, let z_0 be a complex center history, and let r' and r be unary radii. If* $\mathsf{HoloOnDisk}(f, z_0, r)$ *is witnessed by a displayed* $\mathsf{Holomorphic}^{\uparrow}$ *seed witness W with uniform modulus ledger δ_r, and if* $\mathsf{Subradius}(r', r; e)$ *is certified, then* $\mathsf{HoloOnDisk}(f, z_0, r')$ *is witnessed by a subdisk witness $W \restriction_e$.*

More explicitly, for every point z and displayed smaller-disk gap g',

$$\mathsf{OpenDisk}(z_0, r', z, g') \wedge \mathsf{Subradius}(r', r; e) \Longrightarrow \mathsf{OpenDisk}(z_0, r, z, g' \cdot e).$$

The restricted modulus is obtained by reading every smaller boundary gap through this inclusion:

$$\delta_{r'}(k, g') := \delta_r(k, g' \cdot e).$$

If the parent witness uses a displayed large-disk gap g instead of $g' \cdot e$, then $g\mathsf{hsame}g' \cdot e$ by Definition 68.29, so the modulus and derivative ledgers transport along the stability field of Definition 68.54. The classifier on $W \restriction_e$ is the parent pointwise classifier restricted to points admitted by $\mathsf{OpenDisk}(z_0, r', -)$*; its radius slot is r' on both restricted representatives, and the comparison back to W is recorded by the subradius ledger e. The ledger of $W \restriction_e$ keeps the same function history, center history, pointwise complex-differentiability data, derivative-recursion data, Taylor data, Cauchy-bound data, and identity data, with each disk-local admission transported through the subradius certificate.*

Proof. Unpack the seed predicate of Definition 68.33. The parent witness W supplies, for every z admitted by $\mathsf{OpenDisk}(z_0, r, z)$ and every unary precision k, the corresponding complex-differentiability witness controlled by δ_r and the displayed boundary gap.

Now fix z, k, and a smaller-disk gap g' with $\mathsf{OpenDisk}(z_0, r', z, g')$. The subradius certificate gives a unary e and $\mathsf{Cont}(r', e, r)$. Applying Definition 68.28 to the smaller open disk gives $\mathsf{OpenDisk}(z_0, r, z)$. In gap form, Definition 68.29 reads the large boundary as the appended gap $g' \cdot e$, up to hsame against any displayed parent gap. Therefore the parent holomorphic witness applies at the same point z and precision k.

Define the restricted modulus by $\delta_{r'}(k, g') = \delta_r(k, g' \cdot e)$. The input $g' \cdot e$ is unary by unary append closure, so this is a total unary modulus ledger. If a canonical parent gap is chosen differently, the public readback theorem gives the required hsame comparison, and Definition 68.54 transports the modulus and differentiability witness across that comparison.

All remaining fields are inherited by narrowing the domain of quantification. The function history and center history are unchanged. Disk-gap ledgers are sent to the parent disk by $g' \mapsto g' \cdot e$. Any ledger entry indexed by a further subradius of r' is read in the parent ledger by composing that subradius with e using Definition 77.37. The pointwise classifier clause of Definition 68.53 is therefore exactly the parent pointwise classifier after the open-disk inclusion, while the restricted radius data are compared by reflexivity inside the smaller carrier and related to the parent carrier by the retained subradius ledger. These data satisfy the source, classifier, and ledger fields of Definition 68.51, Definition 68.53, and Definition 68.55 for radius r'. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap subradius inclusion`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap certified subradius public readback`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap subradius inclusion readback`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCertifiedSubradius openDiskGap extension gap readback`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCertifiedSubradius open disk gap public readback`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap subradius extension public readback`
Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyOpenDiskGap certified subradius extension`

Theorem 77.39 (HoloOnDisk additive closure)**.** *In the* $\mathsf{Holomorphic}^{\uparrow}$ *seed setup, let* $f, g : \mathsf{Complex}^{\uparrow} \to \mathsf{Complex}^{\uparrow}$*, let* z_0 *be a complex history, and let* r *be a unary radius. Suppose* $\mathsf{HoloOnDisk}(f, z_0, r)$ *and* $\mathsf{HoloOnDisk}(g, z_0, r)$ *are witnessed by uniform modulus data*

$$(\delta_f, D_f, L_f) \qquad \text{and} \qquad (\delta_g, D_g, L_g),$$

where $D_f(z)$ *and* $D_g(z)$ *are the derivative histories selected at a point* z *and* L_f, L_g *are the corresponding disk-gap ledgers. Assume the witnesses are compatible in the following sense: for every boundary witness* $\mathsf{OpenDisk}(z_0, r, z, gap)$*, both* L_f *and* L_g *read the displayed gap up to* hsame *and use it as the second input of their uniform moduli. Define*

$$D_{f+g}(z) := D_f(z) + D_g(z), \qquad \delta_{f+g}(k, gap) := \max(\delta_f(k+1, gap), \delta_g(k+1, gap)).$$

Then the pointwise complex sum satisfies

$$\mathsf{HoloOnDisk}(f + g, z_0, r).$$

Proof. Unpack the two $\mathsf{HoloOnDisk}$ witnesses using Definition 68.33. Fix a point z and a boundary witness $\mathsf{OpenDisk}(z_0, r, z, gap)$. The f-witness supplies $\mathsf{CplxDiffAt}(f, z, D_f(z))$ with the one-variable modulus $k \mapsto \delta_f(k, gap)$, and the g-witness supplies $\mathsf{CplxDiffAt}(g, z, D_g(z))$ with the one-variable modulus $k \mapsto \delta_g(k, gap)$.

If either source witness presents an hsame-equivalent boundary gap rather than the displayed gap, compatibility identifies it with gap. The open-disk transport theorem Definition 68.7 keeps the same disk membership, and full pointwise differentiability transport Definition 67.31 moves the corresponding differentiability witness to the displayed point and displayed gap representative.

Apply the complex differentiability sum rule Definition 67.32 at this fixed point. It gives

$$\mathsf{CplxDiffAt}(f + g, z, D_f(z) + D_g(z))$$

with modulus $k \mapsto \max(\delta_f(k + 1, gap), \delta_g(k + 1, gap))$, which is exactly $k \mapsto \delta_{f+g}(k, gap)$. The proof of Definition 67.32 decomposes the difference quotient of $f + g$ into the sum of the two source quotients and uses the shifted precision $k + 1$ on each side, so the pasted error is below the requested precision k.

The construction is uniform in z and in the displayed boundary gap. The center z_0, the radius r, and the open-disk witness are unchanged, while the output ledger records the two source ledgers, the common gap ledger, the pointwise complex-sum quotient ledger, and the pasted unary modulus δ_{f+g}. The stability seed field of Definition 68.54 transports these ledgers along hsame, and the ledger composition principle Definition 8.12 supplies the composite ledger for the sum witness.

Thus every point of the disk carries complex differentiability of $f+g$ with the displayed uniform modulus and derivative selector D_{f+g}. By Definition 68.33, this is precisely $\mathsf{HoloOnDisk}(f + g, z_0, r)$. □

Definition 77.40 (Certified holomorphic dense-sequence agreement)**.** *Let $f, g : \mathsf{Complex}^{\uparrow} \to \mathsf{Complex}^{\uparrow}$, let z_0 be a complex center, let r be a unary radius, and let z_c be a complex history in $\mathsf{OpenDisk}(z_0, r)$. The predicate*

$$\mathsf{HoloDenseSeqEq}(f, g, z_0, r, z_c, s)$$

holds when $s : \mathsf{UnaryHistory} \to \mathsf{Complex}^{\uparrow}$ is the quantitative density witness of Definition 77.28 for the open set $\mathsf{OpenDisk}(z_0, r)$ centered at z_c, and every unary index carries the pointwise classifier

$$\forall n,\ \mathsf{UnaryHistory}(n) \Longrightarrow \mathsf{ComplexClassifier}(f(s(n)), g(s(n))).$$

Definition 77.41 (Certified smaller holomorphic disk)**.** *For a complex center w, unary radius ρ, ambient disk $\mathsf{OpenDisk}(z_0, r)$, and cluster center z_c, the predicate*

$$\mathsf{HoloSubDisk}(w, \rho; z_0, r, z_c)$$

holds when every point of $\mathsf{OpenDisk}(w, \rho)$ is carried into $\mathsf{OpenDisk}(z_0, r)$, and for every such point y there is a finite list of centers $c_0, \ldots, c_N$ and unary radii $\eta_0, \ldots, \eta_N$ with $c_0 = z_c$, $y \in \mathsf{OpenDisk}(c_N, \eta_N)$, each $\mathsf{OpenDisk}(c_i, \eta_i)$ contained in $\mathsf{OpenDisk}(z_0, r)$, and each consecutive pair carrying a $\mathsf{DomCompat}$ overlap in the sense of Definition 77.31. The overlap witness includes the quantitative cluster sequence supplied by Definition 77.29.

Lemma 77.42 (Cluster zeros force a local holomorphic zero classifier)**.** *Let $h : \mathsf{Complex}^{\uparrow} \to \mathsf{Complex}^{\uparrow}$ satisfy $\mathsf{HoloOnDisk}(h, z_0, r)$. If s is a quantitative density witness for $\mathsf{OpenDisk}(z_0, r)$ centered at z_c and*

$$\forall n,\ \mathsf{UnaryHistory}(n) \Longrightarrow \mathsf{ComplexClassifier}(h(s(n)), 0_{\mathsf{Complex}^{\uparrow}}),$$

then there is a unary radius η with $\mathsf{OpenDisk}(z_c, \eta) \subseteq \mathsf{OpenDisk}(z_0, r)$ *such that*

$$\forall y,\ \mathsf{OpenDisk}(z_c, \eta, y) \Longrightarrow \mathsf{ComplexClassifier}(h(y), 0_{\mathsf{Complex}^\uparrow}).$$

Proof. The holomorphic seed predicate of Definition 68.33 gives a uniform complex-differentiability modulus at every point of the ambient disk. At z_c, the iterated-derivative and power-series clauses listed in Definition 68.34 give Taylor coefficients a_m and a unary convergence radius η whose open disk is contained in $\mathsf{OpenDisk}(z_0, r)$.

The coefficients are classified by zero by unary induction on m. For $m = 0$, the density witness gives $s(n) \to z_c$ with explicit modulus, and the displayed zero classifier on $h(s(n))$ transports through the continuity part of the holomorphic modulus, so $h(z_c)$ is classified by $0_{\mathsf{Complex}^\uparrow}$. For the successor step, assume $a_0, \ldots, a_{m-1}$ are classified by zero. On every cluster point $s(n) \neq z_c$, the m-th normalized Taylor remainder is defined; the numerator is classified by zero because $h(s(n))$ is zero and the lower Taylor terms are zero. The differentiability modulus identifies the limit of these normalized remainders with a_m, while the constant-zero quotient sequence has limit $0_{\mathsf{Complex}^\uparrow}$. The uniqueness classifier of $\mathsf{CplxLim}^\uparrow$ therefore gives $\mathsf{ComplexClassifier}(a_m, 0_{\mathsf{Complex}^\uparrow})$.

For any $y \in \mathsf{OpenDisk}(z_c, \eta)$, every finite Taylor partial sum at y is classified by $0_{\mathsf{Complex}^\uparrow}$ because all coefficients are. The power-series convergence witness and complex-limit uniqueness identify the value $h(y)$ with that zero limit. □

Theorem 77.43 (Holo dense-sequence identity classifier)**.** *Let $f, g : \mathsf{Complex}^\uparrow \to \mathsf{Complex}^\uparrow$ satisfy* $\mathsf{HoloOnDisk}(f, z_0, r)$ *and* $\mathsf{HoloOnDisk}(g, z_0, r)$*. If* $\mathsf{HoloDenseSeqEq}(f, g, z_0, r, z_c, s)$ *and* $\mathsf{HoloSubDisk}(w, \rho; z_0, r, z_c)$ *hold, then the restrictions of f and g to the certified smaller disk are identified by the* $\mathsf{Holomorphic}^\uparrow$ *classifier:*

$$\forall y,\ \mathsf{OpenDisk}(w, \rho, y) \Longrightarrow \mathsf{ComplexClassifier}(f(y), g(y)).$$

Proof. Form the certified difference $h = f - g$ using the $\mathsf{Complex}^\uparrow$ additive inverse and addition operations. The closure clause in Definition 68.34 gives $\mathsf{HoloOnDisk}(h, z_0, r)$. The dense-sequence agreement predicate of Definition 77.40 gives $\mathsf{ComplexClassifier}(f(s(n)), g(s(n)))$ at every unary index, and additive cancellation in $\mathsf{Complex}^\uparrow$ rewrites this as $\mathsf{ComplexClassifier}(h(s(n)), 0_{\mathsf{Complex}^\uparrow})$.

Apply Definition 77.42 at z_c. This gives an initial open disk on which h is classified by zero. Fix $y \in \mathsf{OpenDisk}(w, \rho)$. The certified smaller-disk predicate of Definition 77.41 supplies a finite chain of open disks from the initial disk to a terminal disk containing y, all contained in $\mathsf{OpenDisk}(z_0, r)$ and with consecutive disks related by $\mathsf{DomCompat}$ overlaps.

Induct along the finite chain. Suppose h is zero on the current disk. The next overlap carries, by Definition 77.31 and Definition 77.29, a quantitative cluster sequence lying in the current disk and in the next disk. Hence h is zero on that cluster. Restricting the ambient holomorphic witness to the next disk is allowed by open-disk transport and the stability clause of Definition 68.54. Applying Definition 77.42 inside the next disk makes h zero there. The terminal disk contains y, so $\mathsf{ComplexClassifier}(h(y), 0_{\mathsf{Complex}^\uparrow})$.

Finally, the definition of $h = f - g$ and cancellation in the additive group of $\mathsf{Complex}^\uparrow$ convert $\mathsf{ComplexClassifier}(h(y), 0_{\mathsf{Complex}^\uparrow})$ into $\mathsf{ComplexClassifier}(f(y), g(y))$. This is exactly the pointwise component required by the holomorphic classifier seed field of Definition 68.53; the compatible disk data are supplied by Definition 77.41. □

Theorem 77.44 (HoloOnDisk nested subdisk functoriality)**.** *Let $f : \mathsf{Complex}^\uparrow \to \mathsf{Complex}^\uparrow$, let z_0 be a complex center history, and let r_2, r_1, and r_0 be unary radii. Let W be a displayed* $\mathsf{Holomorphic}^\uparrow$ *seed witness for* $\mathsf{HoloOnDisk}(f, z_0, r_0)$ *with uniform modulus ledger δ_{r_0}. Suppose*

$$\mathsf{Subradius}(r_1, r_0; e_0) \qquad \textit{and} \qquad \mathsf{Subradius}(r_2, r_1; e_1).$$

Let $W_1 := W \restriction_{e_0}$ be the restricted witness on r_1, let $W_{21} := W_1 \restriction_{e_1}$ be the iterated restricted witness on r_2, and let $W_{20} := W \restriction_{e_1 \cdot e_0}$ be the direct restricted witness obtained from the composed subradius certificate. Then W_{21} and W_{20} are compared by the $\mathsf{Holomorphic}^{\uparrow}$ *classifier over* $\mathsf{HoloOnDisk}(f, z_0, r_2)$.

More explicitly, for every displayed smaller-disk gap g'' and every unary precision k, let

$$d_{21} := \delta_{r_0}(k, (g'' \cdot e_1) \cdot e_0), \qquad d_{20} := \delta_{r_0}(k, g'' \cdot (e_1 \cdot e_0))$$

be the displayed parent ledger-result histories. The iterated and direct modulus ledgers read the parent ledger as

$$\delta_{W_{21}}(k, g'') = \delta_{r_1}(k, g'' \cdot e_1) = d_{21}, \qquad \delta_{W_{20}}(k, g'') = d_{20}, \qquad d_{21}\mathsf{hsame}d_{20}.$$

The final classifier is the associativity comparison from Definition 21.29 together with continuation transport by Definition 4.41. If the public composed tail is displayed as $e_0 \cdot e_1$, unary append commutativity supplies a classifier from $e_1 \cdot e_0$ to $e_0 \cdot e_1$; the corresponding continuation comparison is carried by Definition 24.7 and Definition 4.41, and the same classifier comparison applies to the direct witness $W \restriction_{e_0 \cdot e_1}$.

Proof. Unpack the two certified subradius data. They supply unary histories e_0 and e_1 together with radius ledgers $\mathsf{Cont}(r_1, e_0, r_0)$ and $\mathsf{Cont}(r_2, e_1, r_1)$. Applying Definition 77.37, with its three radius variables read as r_2, r_1, and r_0, gives the composed certificate $\mathsf{Subradius}(r_2, r_0; e_1 \cdot e_0)$.

The first restriction theorem Definition 77.38 constructs W_1 and defines its modulus by

$$\delta_{r_1}(k, g') := \delta_{r_0}(k, g' \cdot e_0).$$

Applying the same theorem to W_1 along e_1 constructs W_{21} and gives

$$\delta_{W_{21}}(k, g'') = \delta_{r_1}(k, g'' \cdot e_1) = \delta_{r_0}(k, (g'' \cdot e_1) \cdot e_0).$$

The direct restriction along the composed subradius certificate constructs W_{20} with

$$\delta_{W_{20}}(k, g'') = \delta_{r_0}(k, g'' \cdot (e_1 \cdot e_0)).$$

Let d_{21} and d_{20} denote these two displayed parent ledger-result histories. History-append associativity, as classified by Definition 21.29, compares $(g'' \cdot e_1) \cdot e_0$ with $g'' \cdot (e_1 \cdot e_0)$, and Definition 4.41 carries the corresponding continuation ledger across that comparison. The stability seed field of Definition 68.54 then reads the parent modulus and the corresponding differentiability witness through the two displayed parent gaps, giving $d_{21}\mathsf{hsame}d_{20}$.

The same comparison controls disk admission. Given z with a displayed r_2-gap g'', the iterated route records the radius-r_1 boundary gap $g'' \cdot e_1$ and then the radius-r_0 parent gap $(g'' \cdot e_1) \cdot e_0$. The direct route records the parent gap $g'' \cdot (e_1 \cdot e_0)$. These two parent gaps are classified by Definition 21.29, and Definition 4.41 carries the continuation ledger across the classifier. Any alternative displayed parent gap is classified against them by the public readback theorem Definition 68.29, with the displayed result comparison carried by Definition 4.42. Thus both witnesses invoke the same parent holomorphic data after the gap comparison.

All remaining ledger fields are inherited from the same parent witness W: the function history, center history, derivative-recursion data, Taylor data, Cauchy-bound data, and identity data are unchanged. Their disk-local indices are related exactly by the two gap maps just compared. Hence the pointwise-function clause and the compatible disk-data clause of Definition 68.53 hold for W_{21}

and W_{20} over the radius r_2, with the inherited ledger policy of Definition 68.55. If the composed tail is displayed as $e_0 \cdot e_1$, unary append commutativity supplies the classifier from $e_1 \cdot e_0$ to that public display; applying Definition 24.7 and Definition 4.41 carries the relevant continuation ledgers to that display before the same argument is applied. □

Lean4 ✓ `BEDC.Derived.ComplexTopologyUp.ComplexTopologyCertifiedSubradius nested gap readback`

Closure status for $\mathsf{NameCert}_{\mathsf{CplxTopo}^{\uparrow}}$.

Constructive story: A $\mathsf{CplxTopo}^{\uparrow}$ packet is generated from finite BHist rows for complex disks, centers, radii, holomorphic witnesses, derivative recursion, Taylor data, Cauchy-bound ledgers, identity rows, subradius routes, transports, provenance, and local naming. Disk-local information is produced by Cont through nested gap maps and certified subradius readbacks, so a smaller disk inherits displayed ledger fields from its parent witness rather than from an ambient topology object. Componentwise hsame transports function, center, derivative, Taylor, Cauchy-bound, and identity rows, while Pkg records the complex analytic and topology dependencies. The packet names finite holomorphic-disk data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CplxTopo}^{\uparrow}}$, seed)

Scope closed: This chapter binds complex-topology disk and holomorphic witness rows, classifier specification, ledger policy, certified subradius routes, and nested-gap readback for $\mathsf{CplxTopo}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CplxTopo}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a host topology space, quotient equality of functions, unrestricted analytic continuation, classical open-cover reasoning, or external sheaf machinery.

Upgrade path: Obligation closure requires Lean encodings of the holomorphic witness carrier, disk ledger policy, subradius route, nested-gap readback, and classifier transport.

Chapter 78

A Concrete Naming Certificate for $\mathsf{Magma}^{\uparrow}$

The magma interface $\mathsf{Magma}^{\uparrow}$ packages a carrier endowed with a single binary operation, with no associativity, identity, or inverse demands. It is the smallest algebraic naming certificate above the inductive carriers of chapter 32–chapter 36 and the entry point of the algebraic ladder culminating in $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45).

78.1 Carrier definition

Definition 78.1 (Concrete unary-history magma instance)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$*. The concrete unary-history magma instance has carrier U, binary operation* $\mathsf{mul}_U(h,k) := \mathsf{append}(h,k)$*, equivalently witnessed by* $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$*, and carrier-aware classifier*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

The classifier only compares unary endpoints, so every public equality claim retains both endpoint carrier witnesses.

Theorem 78.2 (Concrete unary-history magma laws)**.** *For the instance of Definition 78.1,* mul_U *is closed on unary histories. Moreover, if $R_U(h,h')$ and $R_U(k,k')$, then*

$$R_U(\mathsf{mul}_U(h,k), \mathsf{mul}_U(h',k')).$$

Proof. Closure is exactly `BEDC.FKernel.Unary.unary_append_closed`: two unary endpoint witnesses for h and k produce a unary endpoint witness for $\mathsf{append}(h,k)$.

For congruence, unpack $R_U(h,h')$ and $R_U(k,k')$ into unary carrier witnesses and endpoint hsame proofs. The two products are witnessed by their canonical continuation facts $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$ and $\mathsf{Cont}(h',k',\mathsf{append}(h',k'))$. Transport the continuation result along the endpoint hsame data using `BEDC.FKernel.Cont.cont_result_hsame_transport`; the closure clause supplies the carrier witnesses for both products, and the transported result is the classifier equality required by R_U. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont laws`

Theorem 78.3 (Concrete unary-history magma continuation result unary exactness)**.** *For histories $h,k,r : \mathsf{BHist}$, if $\mathsf{Cont}(h,k,r)$, then the continuation result is unary exactly when both continuation inputs are unary:*

$$\mathsf{Unary}(r) \iff \mathsf{Unary}(h) \wedge \mathsf{Unary}(k).$$

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont result unary iff`

Proof. The continuation ledger identifies r with $\mathsf{append}(h, k)$. The unary append factor theorem says precisely that $\mathsf{Unary}(h) \wedge \mathsf{Unary}(k)$ is equivalent to $\mathsf{Unary}(\mathsf{append}(h, k))$. Transporting along the continuation equality gives the displayed result-side exactness. □

Theorem 78.4 (Concrete unary-history magma visible-right continuation result nonempty)**.** *For histories $h, t, r : \mathsf{BHist}$, a continuation whose right input is $\mathsf{E0}(t)$ or $\mathsf{E1}(t)$ cannot have an empty result:*

$$(\mathsf{Cont}(h, \mathsf{E0}(t), r) \Rightarrow \neg\mathsf{hsame}(r, \mathsf{e})) \quad \wedge \quad (\mathsf{Cont}(h, \mathsf{E1}(t), r) \Rightarrow \neg\mathsf{hsame}(r, \mathsf{e})).$$

Proof. Transport the displayed continuation result along $r\mathsf{hsamee}$. Empty-result continuation inversion would then force the visible right input itself to be e. The zero- and one-headed cases are both excluded by constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont visible right result nonempty`

Theorem 78.5 (Concrete unary-history magma visible-left continuation result nonempty)**.** *For histories $h, t, r : \mathsf{BHist}$, a continuation whose left input is $\mathsf{E0}(t)$ or $\mathsf{E1}(t)$ cannot have an empty result:*

$$(\mathsf{Cont}(\mathsf{E0}(t), h, r) \Rightarrow \neg\mathsf{hsame}(r, \mathsf{e})) \quad \wedge \quad (\mathsf{Cont}(\mathsf{E1}(t), h, r) \Rightarrow \neg\mathsf{hsame}(r, \mathsf{e})).$$

Proof. Transport the displayed continuation result along $r\mathsf{hsamee}$. Empty-result continuation inversion would then force the visible left input itself to be e. The zero- and one-headed cases are both excluded by constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont visible left result nonempty`

Theorem 78.6 (Concrete unary-history magma classifier append factors)**.** *For histories $h, h', k, k' : \mathsf{BHist}$, the concrete unary-history magma classifier relates the two appended endpoints exactly when all four factors are unary histories and the two appended endpoints are history-same:*

$$R_U(\mathsf{append}(h, k), \mathsf{append}(h', k')) \iff U(h) \wedge U(k) \wedge U(h') \wedge U(k') \wedge \mathsf{hsame}(\mathsf{append}(h, k), \mathsf{append}(h', k')).$$

Proof. Unfold R_U. The forward direction gives unary carrier witnesses for the two appended endpoints. Apply unary append factor exactness to each carrier witness, obtaining the unary witnesses for h, k, h', k', and retain the classifier's endpoint hsame component. Conversely, the four unary factor witnesses close both appended endpoints under unary append closure, and the displayed endpoint hsame proof repackages the concrete classifier. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma classifier append factors iff`

Theorem 78.7 (Concrete unary-history magma classifier append-empty endpoints)**.** *Let $U(h)$ abbreviate $\mathsf{UnaryHistory}(h)$ and let*

$$R_U(h, k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h, k).$$

For histories $h, k : \mathsf{BHist}$, the concrete unary-history magma classifier relates $\mathsf{append}(h, k)$ to e exactly when both append endpoints are empty:

$$R_U(\mathsf{append}(h, k), \mathsf{e}) \iff h\mathsf{hsamee} \wedge k\mathsf{hsamee}.$$

Proof. Unfold R_U. The forward direction reads $\mathsf{append}(h,k)\mathsf{hsamee}$ from the classifier and applies append-empty exactness to recover $h\mathsf{hsamee}$ and $k\mathsf{hsamee}$. Conversely, the two empty endpoints make the append endpoint unary and history-same to e, while e itself is unary. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma classifier append empty endpoints iff`

Theorem 78.8 (Concrete unary-history magma continuation result classifier factors)**.** *For histories $h, h', k, k', r, r' : \mathsf{BHist}$, if $\mathsf{Cont}(h,k,r)$ and $\mathsf{Cont}(h',k',r')$, then the concrete unary-history magma classifier relates the two continuation results exactly when all four source factors are unary and the two results are history-same:*

$$R_U(r,r') \iff U(h) \wedge U(k) \wedge U(h') \wedge U(k') \wedge \mathsf{hsame}(r,r').$$

Proof. Eliminate the two continuation witnesses. The results become the displayed append endpoints $\mathsf{append}(h,k)$ and $\mathsf{append}(h',k')$. The append-factor classifier exactness theorem then supplies the four unary factor witnesses in the forward direction and rebuilds the result classifier from the four witnesses and the endpoint hsame proof in the reverse direction. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont result classifier factors iff`

Theorem 78.9 (Concrete unary-history magma continuation left-context classifier)**.** *Let $U(h)$ abbreviate $\mathsf{UnaryHistory}(h)$ and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $R_U(l,l')$, $\mathsf{Cont}(l,r,a)$, and $\mathsf{Cont}(l',r',a')$, then the concrete magma classifier relates the two continuation results exactly when the two right inputs are unary and history-same:

$$R_U(a,a') \iff U(r) \wedge U(r') \wedge \mathsf{hsame}(r,r').$$

Proof. In the forward direction, unary exactness for each continuation recovers the two right-input carriers from the result carriers. The classifier equality between results cancels the shared classified left context, giving $r\mathsf{hsame}r'$. Conversely, unary closure for the two continuations supplies the result carriers, and continuation respect for hsame transports the left classifier and the right-input comparison to the result classifier. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont left context classifier iff`

Theorem 78.10 (Concrete unary-history magma continuation left-unit classifier)**.** *Let $U(h)$ abbreviate $\mathsf{UnaryHistory}(h)$ and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $\mathsf{Cont}(h,k,r)$, then classifying the result against the right endpoint is equivalent to saying that the right endpoint is unary and the left endpoint is the continuation left unit:

$$R_U(r,k) \iff U(k) \wedge \mathsf{hsame}(h,\mathsf{e}).$$

Proof. In the forward direction, the classifier exposes the carrier of k. Transporting the continuation result along $r\mathsf{hsame}k$ gives a continuation from h by k back to k, and left-unit uniqueness forces $h\mathsf{hsamee}$. Conversely, $\mathsf{hsame}(h,\mathsf{e})$ gives the unary carrier for h and turns the continuation into the left-unit continuation, while unary closure rebuilds the result carrier. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont left unit classifier iff`

Theorem 78.11 (Concrete unary-history magma continuation right-unit classifier)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $\mathsf{Cont}(h,k,r)$*, then classifying the result against the left endpoint is equivalent to saying that the left endpoint is unary and the right endpoint is the continuation right unit:*

$$R_U(r,h) \iff U(h) \wedge \mathsf{hsame}(k,\mathsf{e}).$$

Proof. In the forward direction, the classifier already exposes the carrier of h. Transporting the continuation result along $r\mathsf{hsame}h$ gives a continuation from h by k back to h, and right-unit uniqueness forces $k\mathsf{hsame}\mathsf{e}$. Conversely, $\mathsf{hsame}(k,\mathsf{e})$ gives the unary carrier for k and turns the continuation into the right-unit continuation, while unary closure rebuilds the result carrier. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont right unit classifier iff`

Theorem 78.12 (Concrete unary-history magma continuation result both-input classifiers)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $\mathsf{Cont}(h,k,r)$*, then the result is classified with both inputs exactly when both inputs are unary and both are history-same to* e*:*

$$R_U(r,h) \wedge R_U(r,k) \iff U(h) \wedge U(k) \wedge \mathsf{hsame}(h,\mathsf{e}) \wedge \mathsf{hsame}(k,\mathsf{e}).$$

Proof. Apply the right-unit classifier characterization to the $R_U(r,h)$ side and the left-unit classifier characterization to the $R_U(r,k)$ side. These two readbacks give the two carriers and the two empty-endpoint equations. The reverse direction repacks the same data through the two unit classifier characterizations. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont result both inputs classifier empty iff`

Theorem 78.13 (Concrete unary-history magma continuation right-context classifier)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $R_U(r,r')$, $\mathsf{Cont}(l,r,a)$*, and* $\mathsf{Cont}(l',r',a')$*, then the concrete magma classifier relates the two continuation results exactly when the two left inputs are unary and history-same:*

$$R_U(a,a') \iff U(l) \wedge U(l') \wedge \mathsf{hsame}(l,l').$$

Proof. Unary exactness for the two continuations reads the left-input carriers back from the result carriers. The result classifier equality cancels the shared classified right context, giving $l\mathsf{hsame}l'$. The reverse direction uses unary continuation closure for the two result carriers and continuation respect for hsame across the classified left and right inputs. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont right context classifier iff`

Theorem 78.14 (Concrete unary-history magma classifier middle-context cancellation)**.** *Let* $U(h)$ *abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $U(l)$ *and* $U(r)$*, then the common append context cancels from the concrete magma classifier:*

$$R_U(\mathsf{append}(l,\mathsf{append}(m,r)),\mathsf{append}(l,\mathsf{append}(m',r))) \iff U(m) \wedge U(m') \wedge \mathsf{hsame}(m,m').$$

Proof. The forward direction first applies unary append-factor exactness to the two classified contextual endpoints, exposing unary witnesses for the two middle histories. The classifier equality then cancels the common left append context and the common right append context, yielding $\mathsf{hsame}(m,m')$. Conversely, unary closure rebuilds both contextual endpoints, and the middle equality transports through the two fixed append contexts. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma classifier append middle cancel iff`

Theorem 78.15 (Concrete unary-history magma classifier rotated middle cancellation)**.** *Let* $U(h)$ *abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

If $U(l)$ *and* $U(r)$*, then the concrete magma classifier also cancels a common context when the right endpoint is bracketed as a left append followed by the right context:*

$$R_U(\mathsf{append}(l,\mathsf{append}(m,r)),\mathsf{append}(\mathsf{append}(l,m'),r)) \iff U(m) \wedge U(m') \wedge \mathsf{hsame}(m,m').$$

Proof. Append associativity identifies the rotated right endpoint with the nested right endpoint used by middle-context cancellation. The forward direction transports the classifier to that nested bracketing and applies the middle-cancellation theorem. Conversely, the nested classifier supplied by middle cancellation is transported back across associativity. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma classifier append rotated middle cancel iff`

Theorem 78.16 (Concrete unary-history magma continuation common-context classifier)**.** *Let* $U(h)$ *abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

For histories $l,l',m,m',r,r',a,a' : \mathsf{BHist}$*, if* $R_U(l,l')$ *and* $R_U(r,r')$*, then continuations through the two classified contexts expose exactly the middle classifier:*

$$\mathsf{Cont}(l,\mathsf{append}(m,r),a) \wedge \mathsf{Cont}(l',\mathsf{append}(m',r'),a') \Longrightarrow (R_U(a,a') \iff U(m) \wedge U(m') \wedge \mathsf{hsame}(m,m')).$$

Proof. The context classifiers identify the left contexts and the right contexts while carrying all four endpoints as unary histories. Eliminating the two continuation witnesses turns the results into the two displayed append contexts. The middle-context cancellation theorem then cancels the common unary left context and common unary right context, giving exactly the two middle carrier witnesses and the middle hsame comparison; the reverse direction rebuilds the contextual classifier by unary append closure. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont common context classifier iff`

Theorem 78.17 (Concrete unary-history magma nested continuation common-context classifier)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

For histories $l, l', m, m', r, r', lm, lm', a, a' : \mathsf{BHist}$, if $R_U(l,l')$ and $R_U(r,r')$, and if the two results are obtained through nested continuations

$$\mathsf{Cont}(l,m,lm), \quad \mathsf{Cont}(lm,r,a), \quad \mathsf{Cont}(l',m',lm'), \quad \mathsf{Cont}(lm',r',a'),$$

then the result classifier exposes exactly the middle classifier:

$$R_U(a,a') \iff U(m) \wedge U(m') \wedge \mathsf{hsame}(m,m').$$

Proof. The context classifiers identify the two left contexts and the two right contexts while retaining their unary carrier witnesses. Eliminating the nested continuation witnesses presents the two results as left-associated append contexts. Associativity transports those endpoints to the common middle-context form, where the concrete magma middle-cancellation theorem cancels the fixed unary left and right contexts. The reverse direction rebuilds the two left-associated results by transporting the same classifier back across associativity. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont nested common context classifier iff`

Theorem 78.18 (Concrete unary-history magma nested left-unit classifier)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

For histories $l, m, r, lm, a : \mathsf{BHist}$, if $\mathsf{Cont}(l,m,lm)$ and $\mathsf{Cont}(lm,r,a)$, then classifying the nested result against the right endpoint is equivalent to the right endpoint being unary and both visible left factors being empty:

$$R_U(a,r) \iff U(r) \wedge \mathsf{hsame}(l,\mathsf{e}) \wedge \mathsf{hsame}(m,\mathsf{e}).$$

Proof. Apply the one-step left-unit classifier to the second continuation. The forward direction gives $U(r)$ and $lm\mathsf{hsamee}$; decomposing the first continuation then splits $lm\mathsf{hsamee}$ into $l\mathsf{hsamee}$ and $m\mathsf{hsamee}$. Conversely, the two empty-factor equalities compose through the first continuation to give $lm\mathsf{hsamee}$, and the one-step left-unit classifier rebuilds $R_U(a,r)$ for the second continuation. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont nested left unit classifier iff`

Theorem 78.19 (Concrete unary-history magma nested right-unit classifier)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and let*

$$R_U(h,k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k).$$

For histories $l, m, r, lm, a : \mathsf{BHist}$, if $\mathsf{Cont}(l,m,lm)$ and $\mathsf{Cont}(lm,r,a)$, then classifying the nested result against the left endpoint is equivalent to the left endpoint being unary and both visible right factors being empty:

$$R_U(a,l) \iff U(l) \wedge \mathsf{hsame}(m,\mathsf{e}) \wedge \mathsf{hsame}(r,\mathsf{e}).$$

Proof. Compose the two continuation ledgers by associativity into a continuation from l through $\mathsf{append}(m, r)$ to a. If $R_U(a, l)$ holds, right unit uniqueness forces $\mathsf{append}(m, r)$ to be empty, and append-empty inversion splits this into empty m and empty r. Conversely, empty m makes lm agree with l, and empty r lets the one-step right-unit classifier for the second continuation rebuild $R_U(a, l)$. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma cont nested right unit classifier iff`

The ten-field per-object fit is: traditional target magma; BEDC source BHist restricted to unary histories; carrier UnaryHistory; classifier carrier-aware hsame on unary endpoints; stability closure and congruence for unary append; ledger append provenance through Cont; core theorem Definition 78.2; exactness endpoint classification by R_U; standard bridge free unary append magma; status ProofTarget.

78.2 The certificate

Lemma 78.20 (Concrete unary-history magma classifier stability)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$ *and define*

$$R_U(h, k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h, k).$$

Then R_U has carrier reflexivity on every $U(h)$, symmetry, transitivity, and carrier transport in both endpoints. Concretely:

1. *if $U(h)$, then $R_U(h, h)$;*
2. *if $R_U(h, k)$, then $R_U(k, h)$;*
3. *if $R_U(h, k)$ and $R_U(k, l)$, then $R_U(h, l)$;*
4. *if $R_U(h, k)$, then both carrier witnesses $U(h)$ and $U(k)$ are available for the certificate endpoint fields.*

Proof. Unpack R_U as the triple consisting of the left carrier witness, the right carrier witness, and the endpoint hsame proof.

For carrier reflexivity, a witness $U(h)$ supplies both endpoint carrier fields, and `BEDC.FKernel.Hist.hsame_ref` supplies the endpoint classifier proof.

For symmetry, swap the two carrier witnesses and apply `BEDC.FKernel.Hist.hsame_symm` to the endpoint proof.

For transitivity, retain the outer carrier witnesses $U(h)$ and $U(l)$ from the two packed classifiers. Compose the endpoint equalities with `BEDC.FKernel.Hist.hsame_trans`, then repack the outer carrier witnesses with the composed endpoint proof.

The carrier-transport fields are the two carrier projections already stored in each packed $R_U(h, k)$ witness. □

Lean4 ✓ `BEDC.Derived.MagmaUp.concrete unary history magma classifier stability`

Theorem 78.21 (Concrete unary-history MagmaUp semantic certificate)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$*, let* $\mathsf{mul}_U(h, k) := \mathsf{append}(h, k)$*, and let*

$$R_U(h, k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h, k).$$

The tuple with source, pattern, and ledger all equal to U, classifier R_U, and operation mul_U *is the concrete* $\mathsf{Magma}^{\uparrow}$ *semantic naming certificate over* BHist*. Its certificate fields are: carrier*

inhabitation by e*; classifier reflexivity, symmetry, transitivity, and carrier transport; operation closure; operation congruence; ledger append provenance through* Cont*; exactness*

$$R_U(h,k) \iff U(h) \wedge U(k) \wedge \mathsf{hsame}(h,k);$$

and status ProofTarget*.*

Proof. The carrier field is supplied by the empty unary history. The classifier fields are exactly Definition 78.20: carrier reflexivity, symmetry, transitivity, and the two endpoint carrier projections for R_U.

The operation fields are the concrete laws of Theorem 78.2, whose checked Lean target is `BEDC.Derived.MagmaUp.concrete_unary_history_magma_cont_laws`. Closure is the unary append-closure component, congruence is the append classifier-transport component, and the ledger row is the canonical continuation witness $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$. Exactness is the definition of R_U, so the displayed predicate is both the public classifier and the certificate's endpoint criterion. □

Lean4 ✓ `BEDC.Derived.MagmaUp.ConcreteUnaryHistoryMagma semanticNameCert`

Theorem 78.22 (Magma mature downstream consumption boundary)**.** *The concrete semantic certificate of Definition 78.21, the* $\mathsf{Semigroup}^{\uparrow}$ *forgetful consumer Definition 79.7, and the concrete* $\mathsf{Monoid}^{\uparrow}$ *history instance of section 40.2 together witness the downstream-consumption boundary required for* $\mathsf{Magma}^{\uparrow}$ *mature paper closure.*

Lean4 ✓ `BEDC.Derived.MagmaUp.MagmaUp StdBridge`

Proof. The $\mathsf{Magma}^{\uparrow}$ certificate names the carrier by unary histories in BHist, uses the classifier $R_U(h,k)$ as the carrier-aware hsame relation, takes append as the binary operation, and records the Cont ledger row for the append output. Its exactness clause identifies the public classifier with those same carrier and hsame conditions, so the exported object contains no quotient layer and no ambient host magma.

The $\mathsf{Semigroup}^{\uparrow}$ consumer of Definition 79.7 reads the same carrier, classifier, and append operation after omitting associativity from the semigroup certificate. The resulting $\mathsf{Magma}^{\uparrow}$ surface is therefore not a new host algebra: it is the $\mathsf{Magma}^{\uparrow}$ part of the concrete unary-history semigroup surface.

The $\mathsf{Monoid}^{\uparrow}$ instance of section 40.2 extends the same append and Cont surface by the unit rows for e. Forgetting the unit data leaves the same unary-history append carrier and classifier already used by the $\mathsf{Magma}^{\uparrow}$ certificate. Thus the downstream algebra ladder consumes the concrete certificate without introducing extra magma data or quotient identifications. □

Closure status for $\mathsf{NameCert}_{\mathsf{Magma}^{\uparrow}}$**.**

Constructive story: A magma at this point is the unary history spine with one operation exposed: two admitted histories h and k are put in sequence by kernel append, equivalently by the visible continuation witness $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$, and only histories built from e by b1-stacking are accepted while every b0 branch remains outside the carrier. Two carried products are classified by the carrier-aware unary relation R_U: both endpoints must still be unary and their underlying histories must be hsame, so the operation is not a host binary table and there is no quotient layer hiding representatives. The source, pattern, and ledger all return to the same unary predicate, with the pattern adding the binary append shape, closure, congruence, nonempty visible-input readback, and exact endpoint criterion, while the ledger keeps the raw continuation that produced each visible product. The five name-certificate obligations follow from e as the inhabited unary carrier, from reflexivity, symmetry, and transitivity of history-sameness, and from the fact that a packed $R_U(h,k)$ witness already contains the carrier rows needed to keep recognized endpoints inside the unary surface. A history h lies in the signature gap of a $\mathsf{Magma}^{\uparrow}$ token p when the source-domain accepts the displayed unary input or product, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded here, the answer is exactly the unary append endpoint classified by R_U together with its continuation provenance.

The paper bridge reads the same object as the free unary append magma for downstream Semigroup$^{\uparrow}$ and Monoid$^{\uparrow}$ consumers, but the readback is only the displayed append surface and not an arbitrary host magma. The chapter registers this unary-append binary operation under the name Magma$^{\uparrow}$; it introduces no associativity, identity, inverse, quotient magma, or independent algebraic carrier beyond the existing history continuation calculus.

Theory closure: TheoryGate(NameCert$_{\mathsf{Magma}^{\uparrow}}$, maturePackageClosed)

Scope closed: The concrete unary-history magma surface over BHist with append as the binary operation, classifier R_U, continuation ledger, exact public classifier criterion, and downstream SemigroupUp and MonoidUp consumers over the same unary-history append and Cont surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{Magma}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.MagmaUp.ConcreteUnaryHistoryMagma\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Arbitrary host magmas, quotient magmas, free magma constructions, and checked standard-bridge verification are outside this closure claim.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 79

A Concrete Naming Certificate for $\mathsf{Semigroup}^{\uparrow}$

The semigroup interface $\mathsf{Semigroup}^{\uparrow}$ adds associativity to $\mathsf{Magma}^{\uparrow}$ (chapter 78). It sits between the magma layer and $\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$ (chapter 40), exposing the closure law that distinguishes free semigroups from arbitrary binary structures.

79.1 Carrier definition

Definition 79.1 (Concrete unary-history semigroup instance)**.** *The concrete unary-history semigroup instance reuses the magma carrier, operation, and classifier from Definition 78.1. Thus the carrier is* $\mathsf{UnaryHistory}$*, multiplication is* $\mathsf{mul}_U(h,k) := \mathsf{append}(h,k)$*, the continuation witness is* $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$*, and endpoint comparison is the carrier-aware relation*

$$R_U(h,k) := \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{hsame}(h,k).$$

Theorem 79.2 (Concrete unary-history semigroup laws)**.** *The instance of Definition 79.1 satisfies the magma closure and multiplication congruence laws of Definition 78.2. In addition, for unary histories* $h,k,l : \mathsf{BHist}$*,*

$$R_U(\mathsf{mul}_U(\mathsf{mul}_U(h,k),l), \mathsf{mul}_U(h,\mathsf{mul}_U(k,l))).$$

Proof. The closure and multiplication congruence clauses are the magma laws from Definition 78.2, applied to the same carrier $\mathsf{UnaryHistory}$, operation append, and classifier R_U.

For associativity, first apply unary append closure to h and k, and again to k and l. These witnesses place the intermediate products $\mathsf{append}(h,k)$ and $\mathsf{append}(k,l)$ in the carrier, so the two outer products are also unary by the same closure theorem.

The four continuation witnesses $\mathsf{Cont}(h,k,\mathsf{append}(h,k))$, $\mathsf{Cont}(\mathsf{append}(h,k),l,\mathsf{append}(\mathsf{append}(h,k),l))$, $\mathsf{Cont}(k,l,\mathsf{append}(k,l))$, and $\mathsf{Cont}(h,\mathsf{append}(k,l),\mathsf{append}(h,\mathsf{append}(k,l)))$ are then fed to `BEDC.FKernel.Cont.cont` The result is the endpoint hsame proof between the two parenthesizations, and the carrier witnesses already built above package it as R_U. □

Lean4 ✓ `BEDC.Derived.SemigroupUp.concrete unary history semigroup cont assoc classifier`

Theorem 79.3 (Concrete unary-history semigroup fourfold coherence)**.** *For unary histories* $h,k,l,m : \mathsf{BHist}$*, the left-nested and right-nested fourfold products are classified by* R_U*:*

$$R_U(\mathsf{mul}_U(\mathsf{mul}_U(\mathsf{mul}_U(h,k),l),m), \mathsf{mul}_U(h,\mathsf{mul}_U(k,\mathsf{mul}_U(l,m)))).$$

The ten-field per-object fit is: traditional target fourfold semigroup associativity coherence; BEDC source BHist *restricted to* UnaryHistory*; carrier* UnaryHistory*; classifier carrier-aware* hsame*; stability from unary append closure, the checked ternary semigroup law, and classifier transitivity; ledger the two fourfold append trees; core theorem Definition 79.3; exactness endpoint classification by R_U; standard bridge free unary append semigroup; status* ProofTarget.

Proof. First use unary append closure to keep all intermediate products in the carrier. In particular, $\mathsf{mul}_U(l, m)$, $\mathsf{mul}_U(h, k)$, $\mathsf{mul}_U(\mathsf{mul}_U(h, k), l)$, and the outer products on both sides are again unary histories.

Apply Definition 79.2 to $(\mathsf{mul}_U(h, k), l, m)$. This classifies the left-nested product with the middle bracketing

$$\mathsf{mul}_U(\mathsf{mul}_U(h, k), \mathsf{mul}_U(l, m)).$$

Apply Definition 79.2 again to $(h, k, \mathsf{mul}_U(l, m))$. This classifies the same middle bracketing with the right-nested product.

Finally compose the two classifier witnesses using Definition 78.20. Its transitivity component gives the displayed R_U classification between the two fourfold append trees. □

The ten-field per-object fit is: traditional target semigroup; BEDC source BHist restricted to unary histories; carrier UnaryHistory; classifier carrier-aware hsame on unary endpoints; stability closure, congruence, and associativity for unary append; ledger append provenance through Cont; core theorem Definition 79.2; exactness endpoint classification by R_U; standard bridge free unary append semigroup; status ProofTarget.

79.2 The certificate

Theorem 79.4 (Concrete unary-history SemigroupUp semantic certificate)**.** *Let $U(h)$ abbreviate* $\mathsf{UnaryHistory}(h)$*, let* $\mathsf{mul}_U(h, k) := \mathsf{append}(h, k)$*, and let*

$$R_U(h, k) := U(h) \wedge U(k) \wedge \mathsf{hsame}(h, k).$$

The tuple with source, pattern, and ledger all equal to U, classifier R_U, and multiplication mul_U *is the concrete* $\mathsf{Semigroup}^{\uparrow}$ *semantic naming certificate over* BHist*. Its certificate fields are: carrier inhabitation by* e*; classifier reflexivity, symmetry, transitivity, and carrier transport; multiplication closure; multiplication congruence; associativity; ledger append provenance through* Cont*; and exactness*

$$R_U(h, k) \Longleftrightarrow U(h) \wedge U(k) \wedge \mathsf{hsame}(h, k).$$

Proof. The carrier field is supplied by e, which is a unary history. The classifier fields are the four stability components of Definition 78.20: reflexivity on carrier points, symmetry, transitivity, and the two endpoint carrier projections stored in each $R_U(h, k)$ witness.

The multiplication closure and multiplication congruence fields are inherited from Definition 78.2, applied to the same carrier U, multiplication append, and classifier R_U. The ledger row is the canonical append provenance $\mathsf{Cont}(h, k, \mathsf{append}(h, k))$ for the displayed multiplication.

Associativity is the additional semigroup law in Definition 79.2. Its checked Lean source, `BEDC.Derived.SemigroupUp.concrete_unary_history_semigroup_cont_assoc_classifier`, packages the two parenthesized append products as carrier-aware R_U-equivalent endpoints.

Finally, exactness is definitional: the public classifier R_U is precisely the triple consisting of the two unary carrier witnesses and the endpoint hsame proof. Thus the displayed biconditional is the certificate's endpoint criterion, with no extra cases beyond the concrete unary-history carrier. □

Lean4 ✓ `BEDC.Derived.SemigroupUp.ConcreteUnaryHistorySemigroup semanticNameCert`

Definition 79.5 (Semigroup certificate fields)**.** *Let C be a carrier with classifier $\sim_C$ and multiplication $\cdot$. A $\mathsf{Semigroup}^{\uparrow}(C)$ certificate consists of the carrier, source, pattern, classifier, stability, exactness, and ledger rows whose operation language is generated by carried endpoints and binary multiplication. The classifier rows are carrier inhabitation, reflexivity, symmetry, transitivity, and carrier transport for $\sim_C$. The algebraic rows are multiplication closure, multiplication congruence in both arguments, and associativity under $\sim_C$. The exactness row is the public endpoint criterion attached to the same classifier, and the ledger records carrier witnesses and the multiplication operation tree behind each product expression. This abstracts the field list realized in Definition 79.4. No distinguished identity element, left identity row, or right identity row belongs to the $\mathsf{Semigroup}^{\uparrow}$ row set.*

Proposition 79.6 (Monoid forgets to Semigroup certificate)**.** *Let C carry a $\mathsf{Monoid}^{\uparrow}(C)$ certificate with multiplication $\cdot$, identity e, and classifier $\sim_C$. If the identity element is not exported, the left and right identity stability rows are omitted, and every ledger entry whose expression provenance uses the distinguished identity symbol is omitted, then the retained carrier, source, semigroup pattern, classifier, semigroup stability fields, exactness row, and multiplication ledger form a $\mathsf{Semigroup}^{\uparrow}(C)$ certificate. The carrier C, multiplication $\cdot$, and classifier $\sim_C$ are unchanged.*

Proof. By Definition 40.34, the given certificate supplies carrier, source, pattern, classifier, stability, and ledger data for $\mathsf{Monoid}^{\uparrow}(C)$. The construction keeps exactly the rows whose endpoint claims and expression provenance use only the carrier classifier and binary multiplication.

By Definition 40.1, a monoid carrier is a carrier C with classifier $\sim_C$, operation $\cdot : C \times C \to C$, identity e, and law fields. Suppressing the identity coordinate leaves the same carrier, classifier, and binary operation. The operation type already gives the multiplication closure row required by Definition 79.5.

By Definition 40.9, the source is the carrier C with its classifier and operation data, so the retained source is still over the same C, $\sim_C$, and $\cdot$. By Definition 40.10, the monoid pattern forms expressions from e, carrier witnesses, and binary multiplication. Restricting that pattern to expressions containing no identity symbol leaves precisely the product-expression language of Definition 79.5. By Definition 40.11, the classifier is the supplied carrier sameness relation, so the classifier and its endpoint exactness row are copied without changing endpoints.

For stability, Definition 40.13 lists classifier reflexivity, classifier transitivity, associativity under the classifier, left identity, right identity, and multiplication congruence. Retaining the first, second, third, and sixth rows gives the semigroup algebraic stability rows: classifier reflexivity and transitivity, associativity, and multiplication congruence. The classifier symmetry and carrier-transport rows belong to the underlying NameCert core exhibited for monoids in Definition 40.35; they mention only C and $\sim_C$, so they are retained verbatim. The two discarded identity rows mention e and are not semigroup obligations.

Finally, Definition 40.33 records carrier witnesses and operation trees behind monoid expressions. Removing the entries whose operation tree uses the identity symbol leaves the carrier witnesses and binary multiplication provenance for product expressions. These are exactly the ledger rows required by Definition 79.5. Hence the retained rows satisfy the $\mathsf{Semigroup}^{\uparrow}(C)$ certificate fields with the original carrier, multiplication, and classifier. □

Proposition 79.7 (Semigroup forgets to Magma certificate)**.** *Let C be a carrier with classifier $\sim_C$ and multiplication $\cdot$. If C carries a $\mathsf{Semigroup}^{\uparrow}(C)$ certificate with these data, then the retained carrier, source, binary-operation pattern, classifier, exactness row, operation closure row, operation*

congruence row, and binary-operation ledger form a $\mathsf{Magma}^{\uparrow}(C)$ *certificate after the associativity stability row is omitted. The carrier* C*, multiplication* $\cdot$*, and classifier* $\sim_C$ *are unchanged.*

Proof. By Definition 79.5, a $\mathsf{Semigroup}^{\uparrow}(C)$ certificate contains carrier, source, pattern, classifier, stability, exactness, and ledger rows whose operation language is generated by carried endpoints and binary multiplication. Its classifier rows are carrier inhabitation, reflexivity, symmetry, transitivity, and carrier transport for $\sim_C$. Its algebraic rows are multiplication closure, multiplication congruence in both arguments, and associativity under $\sim_C$.

The magma interface in chapter 78 requires a carrier with one binary operation and no associativity, identity, or inverse demands. The concrete field realization in Definition 78.21 records exactly the rows needed by that interface: carrier inhabitation; classifier reflexivity, symmetry, transitivity, and carrier transport; operation closure; operation congruence; binary-operation ledger provenance; and exactness for the same public classifier.

Retain the carrier C, the source over C, the classifier $\sim_C$, and the exactness row attached to that classifier. Interpret the semigroup multiplication $\cdot$ as the magma operation. The pattern and ledger remain well formed because both interfaces use the same single binary operation language; associativity is a stability law, not an additional operation symbol or carrier coordinate.

The closure row and the congruence row of the semigroup certificate are copied as the operation closure and operation congruence rows of the magma certificate. The classifier rows are copied without changing endpoints. The associativity row has no corresponding $\mathsf{Magma}^{\uparrow}$ obligation, so omitting it leaves every magma row supplied. Hence the retained rows satisfy the $\mathsf{Magma}^{\uparrow}(C)$ certificate fields over the original carrier, multiplication, and classifier. □

Definition 79.8 (Opposite semigroup operation)**.** *Let* C *be a carrier with binary multiplication* $\cdot$*. The opposite multiplication on the same carrier is the binary operation* $\star$ *defined by*

$$x \star y := y \cdot x.$$

It introduces no additional carrier element and no additional classifier relation.

Lean4 (definition) `BEDC.Derived.SemigroupUp.ConcreteUnaryHistorySemigroupOppositeMul`
Lean4 (definition) `BEDC.Derived.SemigroupUp.ConcreteUnaryHistorySemigroup oppMul`

Proposition 79.9 (Opposite semigroup certificate)**.** *Let* C *be a carrier with classifier* $\sim_C$ *and multiplication* $\cdot$*. If* C *carries a* $\mathsf{Semigroup}^{\uparrow}(C)$ *certificate for* $\cdot$*, then the same carrier and classifier carry a* $\mathsf{Semigroup}^{\uparrow}(C)$ *certificate for the opposite multiplication* $\star$ *of Definition 79.8.*

Lean4 ✓ `BEDC.Derived.SemigroupUp.ConcreteUnaryHistorySemigroup opposite semanticNameCert`
Lean4 ✓ `BEDC.Derived.SemigroupUp.concrete unary history semigroup opposite cont assoc classifier`

Proof. By Definition 79.5, the semigroup certificate consists of the carrier, source, pattern, classifier, stability, exactness, and ledger rows for carried endpoints and one binary multiplication. The carrier rows and the classifier rows are retained unchanged, since the carrier is still C and the classifier is still $\sim_C$. The exactness row is also retained unchanged, because it is attached to this same public classifier.

The operation source for $\star$ is the displayed definition $x \star y := y \cdot x$. Closure follows immediately from the original multiplication closure row: if $x, y \in C$, then $y \cdot x \in C$, hence $x \star y \in C$.

For congruence, assume $x \sim_C x'$ and $y \sim_C y'$. The original multiplication congruence applied to the ordered pair of witnesses $y \sim_C y'$ and $x \sim_C x'$ gives

$$y \cdot x \sim_C y' \cdot x'.$$

By the definition of $\star$, this is exactly

$$x \star y \sim_C x' \star y'.$$

Thus the opposite operation satisfies the semigroup multiplication-congruence row.

For associativity, let $x, y, z \in C$. The two opposite products unfold as

$$(x \star y) \star z = z \cdot (y \cdot x)$$

and

$$x \star (y \star z) = (z \cdot y) \cdot x.$$

The original associativity row for $\cdot$, applied to the triple (z, y, x), gives a classifier witness

$$(z \cdot y) \cdot x \sim_C z \cdot (y \cdot x).$$

The classifier symmetry row from the same certificate reverses this witness, yielding

$$z \cdot (y \cdot x) \sim_C (z \cdot y) \cdot x,$$

which is precisely the associativity row for $\star$ in the displayed orientation.

It remains only to account for the pattern and ledger rows. A $\star$-expression tree is read as a $\cdot$-expression tree by exchanging the two children at each binary node. Leaves remain carried endpoints, and each binary node is licensed by the original multiplication source. The same tree reading transports the operation-tree provenance in the ledger. Since Definition 79.5 includes no identity, inverse, left-unit, or right-unit row, there is no further handed obligation to check. These rows together form the required $\mathsf{Semigroup}^{\uparrow}(C)$ certificate for the opposite multiplication. □

Closure status for $\mathsf{NameCert}_{\mathsf{Semigroup}^{\uparrow}}$.

Constructive story: A semigroup here keeps the same unary histories and the same append operation as the magma layer, then records the extra fact that three visible append steps can be rebracketed by the kernel continuation associativity theorem. The carrier admits exactly the b1-generated unary spine and rejects b0-headed histories, while the classifier remains the carrier-aware relation R_U: two endpoints are the same only when both are unary and their histories are hsame. The source, pattern, and ledger therefore cover unary endpoints with binary product trees; the pattern adds closure, product congruence, ternary associativity, fourfold coherence, the forgetful rows to and from neighboring algebra layers, and the opposite multiplication row obtained by swapping the two inputs at each binary node. Since e inhabits the carrier, history-sameness supplies reflexivity, symmetry, and transitivity, and unary append closure plus the stored endpoint carrier witnesses keep every classified representative inside the carrier, the five name-certificate obligations are the same spine obligations strengthened by associativity. A unary product history h lies in the signature gap of a $\mathsf{Semigroup}^{\uparrow}$ token p when the source-domain accepts the relevant unary product tree, the signature relation pairs its displayed append result with an answer history and evidence ledger, and the token-introduction relation names that answer as p; after unfolding, h is the append endpoint encoded by p, independent of parenthesization up to R_U. The paper bridge reads this as the free unary append semigroup, and the forgetful interfaces merely omit or retain rows over the same carrier rather than changing the underlying histories. The chapter registers this associative unary-append construction under the name $\mathsf{Semigroup}^{\uparrow}$; identity, inverse, commutativity, ring structure, field structure, and checked standard-bridge verification are not introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Semigroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper-side package consists of the concrete unary-history $\mathsf{Semigroup}^{\uparrow}$ semantic certificate over UnaryHistory, append, carrier-aware hsame, Cont provenance, associativity, congruence, and exactness; the forgetful bridge from $\mathsf{Monoid}^{\uparrow}$ in Definition 79.6; the forgetful bridge to $\mathsf{Magma}^{\uparrow}$ in Definition 79.7; and the opposite semigroup certificate in Definition 79.9.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Semigroup}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SemigroupUp.ConcreteUnaryHistorySemigroup\_semanticNameCert`

Bridge status: paperBridge

Not claimed: Checked bridge verification, identity, inverse, commutativity, ring or field structure remain outside this claim.

Upgrade path: Bridge verification can promote the paper bridge rows to checked bridge status without changing the mature semigroup package recorded here.

Chapter 80

A Concrete Naming Certificate for $\mathsf{Subgroup}^{\uparrow}$

The subgroup interface $\mathsf{Subgroup}^{\uparrow}$ packages a subset of a group carrier closed under the group operation, identity, and inverse. It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and is the first sub-object construction in BEDC, prerequisite to the quotient and ideal certificates that follow.

80.1 Carrier definition

Definition 80.1 (Centralizer SubgroupUp instance)**.** *Fix a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *source over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and fixed element* $a : \mathsf{BHist}$*. The centralizer* $\mathsf{Subgroup}^{\uparrow}$ *instance uses the carrier* Z_a *from Definition 41.84. Its classifier is the restriction of* hsame *to that carrier: two carried histories* x *and* y *represent the same subgroup element exactly when* $x\mathsf{hsame}y$*. The operation, unit, and inverse are inherited from the ambient* $\mathsf{Group}^{\uparrow}$ *source, and the ledger records the fixed element* a *together with the commuting witnesses certifying membership in* Z_a*.*

Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerCarrier`
Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerClassifier`

80.2 The certificate

Theorem 80.2 (Centralizer SubgroupUp certificate target)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying associativity, left and right unit laws for* e*, multiplication congruence, and two-sided inverse laws, the instance of Definition 80.1 satisfies the* $\mathsf{Subgroup}^{\uparrow}$ *rows:*

1. *the carrier is inhabited by* e*, and* $Z_a(e)$*;*
2. *if* $Z_a(x)$ *and* $Z_a(y)$*, then* $Z_a(\mathsf{mul}(x, y))$*;*
3. *if* $Z_a(x)$*, then* $Z_a(\mathsf{inv}(x))$*;*
4. *if* $Z_a(x)$ *and* $x\mathsf{hsame}y$*, then* $Z_a(y)$*;*
5. *for every* $x : \mathsf{BHist}$*,*
$$Z_a(x) \iff \mathsf{comm}(x, a)\mathsf{hsame}e.$$

Thus the centralizer carrier is a concrete certificate target for $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ *over the ambient history classifier.*

Proof. The inhabited, product-closure, inverse-closure, and commutator-fiber rows are the rows of Definition 41.87. The inverse row uses the ambient inverse congruence from Definition 41.60 and the reverse-product description of inverses from Definition 41.65; these are exactly the sources cited by the GroupUp centralizer proof.

For carrier transport, unfold Z_a. A witness $x\mathsf{hsame}y$ transports $\mathsf{mul}(x,a)\mathsf{hsamemul}(a,x)$ to $\mathsf{mul}(y,a)\mathsf{hsamemul}(a,y)$ by multiplication congruence on the left and right arguments, followed by symmetry and transitivity of hsame. Therefore the restricted classifier preserves the subgroup carrier.

The exactness row is the commutator criterion from Definition 41.82 applied to the ordered pair (x,a), with the definition of Z_a supplying the displayed orientation. Repacking these rows with the inherited operation, unit, inverse, and ledger gives the advertised $\mathsf{Subgroup}^\uparrow$ certificate target. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizer certificate target from empty unit`
Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerCarrier mul closed from empty unit`
Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizer semanticNameCert`

Definition 80.3 (Two-centralizer intersection carrier)**.** *For two fixed histories* $a,b : \mathsf{BHist}$ *in the same BEDC history-level* $\mathsf{Group}^\uparrow$ *source, let* Z_a *and* Z_b *be the centralizer carriers of Definition 41.84. Define the two-centralizer carrier by*

$$Z_{a,b}(x) :\Longleftrightarrow Z_a(x) \wedge Z_b(x).$$

Its $\mathsf{Subgroup}^\uparrow$ *classifier is the restriction of* hsame *to this carrier: two carried histories represent the same element exactly when their ambient histories satisfy* $x\mathsf{hsame}y$*.*

Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionCarrier`
Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionClassifier`

Theorem 80.4 (Centralizer intersection SubgroupUp certificate)**.** *For a BEDC history-level* $\mathsf{Group}^\uparrow$ *operation over* BHist *satisfying the same associativity, left and right unit laws, multiplication congruence, and two-sided inverse laws as in Definition 80.2, and for fixed histories* $a,b : \mathsf{BHist}$*, the carrier* $Z_{a,b}$ *of Definition 80.3 has the following* $\mathsf{Subgroup}^\uparrow$ *certificate rows:*

1. *The restricted* hsame *classifier is a semantic naming certificate for* $Z_{a,b}$*.*
2. $Z_{a,b}(e)$*.*
3. *If* $Z_{a,b}(x)$ *and* $Z_{a,b}(y)$*, then* $Z_{a,b}(\mathsf{mul}(x,y))$*.*
4. *If* $Z_{a,b}(x)$*, then* $Z_{a,b}(\mathsf{inv}(x))$*.*
5. *If* $Z_{a,b}(x)$ *and* $x\mathsf{hsame}y$*, then* $Z_{a,b}(y)$*.*

Thus the simultaneous centralizer of a *and* b *is a reusable* $\mathsf{Subgroup}^\uparrow$ *certificate target over the ambient history classifier.*

Proof. Apply Definition 80.2 twice, once to the fixed history a and once to the fixed history b. The two identity rows give $Z_a(e)$ and $Z_b(e)$, which package directly as $Z_{a,b}(e)$ by Definition 80.3.

For product closure, unpack $Z_{a,b}(x)$ and $Z_{a,b}(y)$ into their two centralizer components. The a-row gives $Z_a(\mathsf{mul}(x,y))$, and the b-row gives $Z_b(\mathsf{mul}(x,y))$; pairing these witnesses proves the displayed product row for $Z_{a,b}$.

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionCarrier mul closed from empty unit`

The inverse row is componentwise in the same sense. From $Z_a(x)$ obtain $Z_a(\mathsf{inv}(x))$, and from $Z_b(x)$ obtain $Z_b(\mathsf{inv}(x))$; the pair is exactly $Z_{a,b}(\mathsf{inv}(x))$.

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionCarrier inv closed from empty unit`

Carrier transport also separates into the two coordinates. Given $Z_{a,b}(x)$ and $x\mathsf{hsame}y$, use the centralizer transport row for a to move $Z_a(x)$ to $Z_a(y)$, and use the centralizer transport row for b to move $Z_b(x)$ to $Z_b(y)$. The restricted hsame classifier on $Z_{a,b}$ is therefore stable under ambient history equality.

Finally, the restricted classifier is reflexive on carried histories, symmetric by symmetry of hsame, and transitive by transitivity of hsame. Its carrier-respect row is the second carrier component already stored in the classifier witness, and the source, pattern, and ledger predicates are all the same concrete intersection carrier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersection certificate from empty unit`

Theorem 80.5 (Centralizer intersection classifier product closure)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *associativity, right unit, and multiplication congruence laws, if the restricted classifier on* $Z_{a,b}$ *relates* x *to* x' *and also relates* y *to* y'*, then it relates the products:*

$$\mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, x, x') \wedge \mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, y, y') \Rightarrow$$

Proof. Unpack both classifier witnesses. Product closure for the carrier $Z_{a,b}$ supplies the left product membership from x, y and the right product membership from x', y'. Multiplication congruence carries the stored comparisons $x\mathsf{hsame}x'$ and $y\mathsf{hsame}y'$ to the product comparison, which repackages the restricted classifier witness. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionClassifier mul closed from empty unit`

Theorem 80.6 (Centralizer intersection classifier inverse closure)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *associativity, two-sided unit, multiplication congruence, and inverse laws, if the restricted classifier on* $Z_{a,b}$ *relates* x *to* y*, then it relates their inverses:*

$$\mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, x, y) \Rightarrow \mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, \mathsf{inv}(x),$$

Proof. Unpack the classifier witness into its two carrier rows and its stored $x\mathsf{hsame}y$ comparison. Inverse closure for the two-centralizer intersection carrier supplies the carrier witnesses for $\mathsf{inv}(x)$ and $\mathsf{inv}(y)$. The ambient group inverse congruence transports $x\mathsf{hsame}y$ to $\mathsf{inv}(x)\mathsf{hsame}\mathsf{inv}(y)$, which repackages the restricted classifier witness. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionClassifier inv closed from empty unit`

Theorem 80.7 (Centralizer intersection self-conjugation closure)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Z_{a,b}(t)$ *and* $Z_{a,b}(x)$*, then both conjugation words*

$$\mathsf{mul}(\mathsf{mul}(t, x), \mathsf{inv}(t))$$

and

$$\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(t), x), \mathsf{inv}(\mathsf{inv}(t)))$$

belong to the two-centralizer intersection carrier $Z_{a,b}$*.*

Proof. Unpack $Z_{a,b}(t)$ and $Z_{a,b}(x)$ into their a-centralizer and b-centralizer components. The self-normalizer theorem for a centralizer applied to the a component of t carries the a component of x through both displayed conjugation words. The same theorem applied to the b component of t carries the b component of x through the same two words. Pairing the corresponding components reconstructs membership in $Z_{a,b}$ for both conjugates. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionCarrier self conjugation closed from empty unit`

Theorem 80.8 (Centralizer intersection classifier self-conjugation closure)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Z_{a,b}(t)$ *and the restricted classifier on* $Z_{a,b}$ *relates* x *to* y*, then it also relates their conjugates by* t*:*

$$\mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, x, y) \Rightarrow \mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, \mathsf{mul}(mu$$

Proof. The carrier self-conjugation theorem sends the x endpoint and the y endpoint separately into the two-centralizer intersection carrier. The stored comparison $x\mathsf{hsame}y$ is transported through the two multiplication slots by multiplication congruence, giving the classifier row for the two conjugates. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionClassifier self conjugation closed from empty unit`

Theorem 80.9 (Centralizer intersection classifier history transport)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, the restricted classifier on* $Z_{a,b}$ *is stable under endpoint* hsame *transport:*

$$\mathsf{SubgroupCentralizerIntersectionClassifier}(mul, a, b, x, y) \wedge x\mathsf{hsame}x' \wedge y\mathsf{hsame}y' \Rightarrow \mathsf{SubgroupCentralizerIntersectionClass}$$

Proof. The intersection certificate supplies carrier transport for $Z_{a,b}$, so the two endpoint carrier rows move from x, y to x', y'. The classifier comparison is the composite $x'\mathsf{hsame}x\mathsf{hsame}y\mathsf{hsame}y'$, using symmetry and transitivity of hsame. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionClassifier hsame transport from empty unit`

Per-object template: traditional target two-centralizer subgroup certificate; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist; carrier $Z_{a,b}$; classifier restricted hsame; stability identity, product, inverse, and carrier transport; ledger fixed histories a, b, two commutator witnesses, and ambient operation witnesses; core theorem list Definition 80.2 and Definition 80.4; exactness predicate simultaneous centralizer membership against a and b; downstream consumer Definition 45.688; standard bridge center as an intersection of centralizers; status $\mathsf{ProofTarget}$.

Definition 80.10 (RatUp FieldUp affine center subgroup carrier)**.** *For the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *affine exit package of Definition 45.662, take the ambient group carrier to be* AffAut_F *with classifier* $\equiv_{\mathrm{aff}}$*, multiplication* $\circ_{\mathrm{aff}}$*, identity* id_F*, and inverse row* ι_F*. The associated* $\mathsf{Subgroup}^{\uparrow}$ *carrier is the affine center predicate on that same group:*

$$\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z) := \mathsf{Center}_F(Z) \qquad (Z : \mathsf{AffAut}_F).$$

Thus membership in the subgroup certificate is exactly center membership in the concrete FieldUp affine automorphism group.

Theorem 80.11 (RatUp FieldUp affine center SubgroupUp instance)**.** *The carrier of Definition 80.10 instantiates the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate fields over* AffAut_F*. Namely:*

1. $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(\mathrm{id}_F)$.
2. *If* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z)$ *and* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(W)$*, then* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z \circ_{\mathrm{aff}} W)$.
3. *If* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z)$*, then* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(\iota_F(Z))$.
4. *If* $Z \equiv_{\mathrm{aff}} W$ *and* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z)$*, then* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(W)$.
5. *If* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(Z)$*, then for every* $T : \mathsf{AffAut}_F$*,* $\mathsf{Carrier}_{\mathsf{Subgroup}^{\uparrow}}(\mathrm{Inn}_T(Z))$.

The same data make this $\mathsf{Subgroup}^{\uparrow}$ *object normal under inner affine conjugation.*

Proof. Fix the canonical exit action and inverse package of Definition 45.662. This fixes the ambient group operations $\circ_{\mathrm{aff}}$, id_F, ι_F, $\equiv_{\mathrm{aff}}$, and the inner affine conjugation row on AffAut_F.

By Definition 80.10, each subgroup membership assertion is exactly the corresponding Center_F assertion. Apply Definition 45.694 to obtain identity membership, product closure, inverse closure, classifier transport, and normality under Inn_T.

The classifier-transport field is compatible with the ambient ledger because Definition 45.553 supplies operation and inverse congruence for $\equiv_{\mathrm{aff}}$. Therefore the five displayed clauses are precisely the concrete $\mathsf{Subgroup}^{\uparrow}$ certificate fields over the FieldUp affine exit package. □

Per-object template: traditional target normal subgroup of the affine automorphism group; BEDC source concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ plus $\mathsf{Group}^{\uparrow}$; carrier Center_F; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, classifier transport, and inner conjugation; ledger affine operation and conjugation witnesses; core theorem list Definition 80.11 and Definition 45.694; exactness predicate center membership as the $\mathsf{Subgroup}^{\uparrow}$ carrier; standard bridge ordinary affine center normal subgroup; status $\mathsf{ProofTarget}$.

Definition 80.12 (RatUp FieldUp affine normalizer SubgroupUp carrier)**.** *For Definition 45.617, fix a certified affine automorphism* $A : \mathsf{AffAut}_F$ *in the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *exit object. Let* $C_F(A)$ *be the affine centralizer carrier and let* $N_F(C_F(A))$ *be the two-sided affine centralizer normalizer carrier. The corresponding* $\mathsf{Subgroup}^{\uparrow}$ *carrier is*

$$\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(T) := N_F(C_F(A))(T) \qquad (T : \mathsf{AffAut}_F).$$

It uses the ambient affine classifier $\equiv_{\mathrm{aff}}$*, operation* $\circ_{\mathrm{aff}}$*, identity* id_F*, and inverse* ι_F*. Its ledger records the fixed automorphism* A*, the centralizer membership tests for* $C_F(A)$*, and the conjugation witnesses asserting preservation and reflection of that centralizer under affine inner action.*

Theorem 80.13 (RatUp FieldUp affine normalizer SubgroupUp instance and center action)**.** *The carrier of Definition 80.12 is the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate target for the affine normalizer* $N_F(C_F(A))$ *over* AffAut_F*. Namely, for the fixed certified* $A : \mathsf{AffAut}_F$*:*

1. $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(\mathrm{id}_F)$.
2. *If* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(T)$ *and* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(U)$*, then* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(T \circ_{\mathrm{aff}} U)$.
3. *If* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(T)$*, then* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(\iota_F(T))$.
4. *If* $T \equiv_{\mathrm{aff}} U$ *and* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(T)$*, then* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(U)$.
5. *If* $\mathsf{Center}_F(Z)$*, then* $\mathsf{Carrier}^{N}_{\mathsf{Subgroup}^{\uparrow}}(Z)$.

6. If $C_F(A)(X)$ and $\mathsf{Center}_F(Z)$, then the restricted normalizer action satisfies $\mathrm{Act}_Z(X) \equiv_{\mathrm{aff}} X$. Thus the affine center includes into this normalizer carrier and acts pointwise as the classifier identity on the fixed centralizer.

Proof. By Definition 80.12, each $\mathsf{Carrier}^N_{\mathsf{Subgroup}^\uparrow}$ assertion is exactly the $N_F(C_F(A))$ normalizer predicate for the fixed affine automorphism A.

Apply Definition 45.695. Its four displayed rows give identity membership, product closure, inverse closure, and transport along $\equiv_{\mathrm{aff}}$ for the carrier $\mathsf{Carrier}^N_{\mathsf{Subgroup}^\uparrow}$.

For the center inclusion and restricted action, apply Definition 45.705. The first displayed implication gives $\mathsf{Center}_F(Z) \Rightarrow \mathsf{Carrier}^N_{\mathsf{Subgroup}^\uparrow}(Z)$, and the second displayed statement gives $\mathrm{Act}_Z(X) \equiv_{\mathrm{aff}} X$ for every carried centralizer element X.

The ambient classifier compatibility is the same compatibility used in Definition 45.553: affine operation and inverse rows respect $\equiv_{\mathrm{aff}}$, so the normalizer witnesses repack as $\mathsf{Subgroup}^\uparrow$ certificate rows without changing the underlying affine group data. □

Per-object template: traditional target normalizer subgroup acting on a centralizer; BEDC source concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$, $\mathsf{Group}^\uparrow$, and $\mathsf{Subgroup}^\uparrow$; carrier $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability identity, product, inverse, transport, and restricted action; ledger fixed automorphism, centralizer tests, normalizer witness, and conjugation witnesses; core theorem list Definition 45.695 and Definition 45.705; exactness predicate normalizer membership as preservation-reflection of $C_F(A)$; standard bridge ordinary normalizer subgroup and center action; status $\mathsf{ProofTarget}$.

Definition 80.14 (RatUp FieldUp affine center-normalizer SubgroupUp pair)**.** *For the concrete $\mathsf{Field}^\uparrow(\mathsf{Rat}^\uparrow)$ affine exit object of Definition 45.617 and a fixed certified affine automorphism $A : \mathsf{AffAut}_F$, take the lower $\mathsf{Subgroup}^\uparrow$ carrier to be Center_F, the upper $\mathsf{Subgroup}^\uparrow$ carrier to be $N_F(C_F(A))$, the classifier to be $\equiv_{\mathrm{aff}}$, the operation to be $\circ_{\mathrm{aff}}$, the identity to be id_F, and the inverse row to be ι_F. The action row is the restricted normalizer action Act_Z on the centralizer carrier $C_F(A)$.*

The certificate ledger records the affine exit package, the center subgroup witnesses, the normalizer preservation and reflection witnesses for $C_F(A)$, and the pointwise restricted action witness. Thus the object is a single center-normalizer pair certificate rather than a separate lower subgroup row, upper subgroup row, inclusion row, and action row.

Theorem 80.15 (RatUp FieldUp affine center-normalizer SubgroupUp pair certificate)**.** *The pair of Definition 80.14 supplies the concrete center-normalizer $\mathsf{Subgroup}^\uparrow$ certificate consumed by quotient and subgroup-action layers. For the fixed certified $A : \mathsf{AffAut}_F$:*

1. *the lower carrier Center_F has the center $\mathsf{Subgroup}^\uparrow$ certificate of Definition 80.11;*
2. *the upper carrier $N_F(C_F(A))$ has the normalizer $\mathsf{Subgroup}^\uparrow$ certificate of Definition 80.13;*
3. *if $\mathsf{Center}_F(Z)$, then $N_F(C_F(A))(Z)$;*
4. *if $C_F(A)(X)$ and $\mathsf{Center}_F(Z)$, then $\mathrm{Act}_Z(X) \equiv_{\mathrm{aff}} X$.*

Proof. The lower certificate is exactly the carrier and stability package of Definition 80.10. Apply Definition 80.11 to obtain the identity, product, inverse, classifier-transport, and inner-conjugation rows for Center_F.

The upper certificate is exactly the carrier and stability package of Definition 80.12. Apply Definition 80.13 to obtain the normalizer identity, product, inverse, and classifier-transport rows, together with the displayed inclusion and action rows.

Finally apply Definition 45.705. Its concrete FieldUp action identity gives $N_F(C_F(A))(Z)$ from $\mathsf{Center}_F(Z)$ and gives $\mathrm{Act}_Z(X) \equiv_{\mathrm{aff}} X$ for every $X : \mathsf{AffAut}_F$ carried by $C_F(A)$. Repacking these rows with the ledger of Definition 80.14 gives the advertised pair certificate. □

Per-object template: traditional target center-normalizer pair in an affine automorphism group; BEDC source $\mathsf{Rat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, and $\mathsf{Subgroup}^{\uparrow}$; carriers Center_F and $N_F(C_F(A))$; classifier $\equiv_{\mathrm{aff}}$; stability subgroup rows plus inclusion and action identity; ledger affine exit package, center witnesses, normalizer preservation and reflection witnesses, and action witness; exactness center inclusion and pointwise identity action; downstream consumer quotientgroup and subgroup-action certificates; status ProofTarget.

Definition 80.16 (Centralizer normalizer witness)**.** *For the centralizer carrier Z_a from Definition 41.84, define the history-level normalizer witness predicate by*

$$\mathsf{Norm}_{Z_a}(t) :\Longleftrightarrow (\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow Z_a(\mathsf{mul}(\mathsf{mul}(t,x),\mathsf{inv}(t))))\wedge(\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow Z_a(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(t),x),\mathsf{inv}(\mathsf{i}$$

Thus $\mathsf{Norm}_{Z_a}(t)$ records that conjugation by t and conjugation by $\mathsf{inv}(t)$ both preserve the centralizer carrier. Equivalently, it is the carrier N_a of Definition 41.95 with the abbreviation Conj *expanded.*

Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer`

Theorem 80.17 (Centralizer normalizer forward product closure)**.** *For a BEDC history-level $\mathsf{Group}^{\uparrow}$ operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, the forward centralizer-normalizer witness is closed under multiplication. Explicitly, if s and t both send every carried element of Z_a to another carried element under conjugation, then $\mathsf{mul}(s,t)$ has the same forward preservation property.*

Proof. Let $x : \mathsf{BHist}$ satisfy $Z_a(x)$. Apply the forward normalizer action composition theorem to the witnesses for s and t. It gives a carried nested conjugation endpoint together with an hsame identification between the direct conjugation by $\mathsf{mul}(s,t)$ and that nested endpoint. The carrier-transport row from Definition 80.2 then moves centralizer membership across this hsame equality, yielding the required carried endpoint for $\mathsf{mul}(s,t)$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer mul closed from empty unit`

Theorem 80.18 (Centralizer normalizer relative product closure)**.** *For a BEDC history-level $\mathsf{Group}^{\uparrow}$ operation over* BHist *satisfying the associativity, unit, multiplication-congruence, and two-sided inverse laws of Definition 80.2, the centralizer-normalizer witness is closed under relative products. If $\mathsf{Norm}_{Z_a}(s)$ and $\mathsf{Norm}_{Z_a}(t)$, then*

$$\mathsf{Norm}_{Z_a}(\mathsf{mul}(\mathsf{inv}(s),t)).$$

Proof. First close $\mathsf{Norm}_{Z_a}(s)$ under inversion. The forward clause for $\mathsf{inv}(s)$ is the reverse conjugation clause already contained in $\mathsf{Norm}_{Z_a}(s)$. The reverse clause is transported from the forward clause for s using inverse involution, multiplication congruence, and the centralizer carrier transport row of Definition 80.2. Applying the normalizer multiplication closure theorem to $\mathsf{inv}(s)$ and t gives the displayed relative product. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer relative product closed from empty unit`

Theorem 80.19 (Centralizer elements normalize the centralizer)**.** *For a BEDC history-level $\mathsf{Group}^{\uparrow}$ operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, every centralizer element normalizes the centralizer. Explicitly, if $Z_a(t)$, then $\mathsf{Norm}_{Z_a}(t)$.*

Proof. Assume $Z_a(t)$. The inverse-closure row of Definition 41.87 applied to $Z_a(t)$ gives $Z_a(\mathsf{inv}(t))$.

For the forward conjugation clause, let $x : \mathsf{BHist}$ satisfy $Z_a(x)$. The product-closure row of Definition 41.87 applied to $Z_a(t)$ and $Z_a(x)$ gives $Z_a(\mathsf{mul}(t, x))$.

Apply product closure once more, now to $Z_a(\mathsf{mul}(t, x))$ and $Z_a(\mathsf{inv}(t))$. The result is $Z_a(\mathsf{mul}(\mathsf{mul}(t, x), \mathsf{inv}(t)))$, which is precisely the first clause of $\mathsf{Norm}_{Z_a}(t)$.

For the inverse conjugation clause, let $x : \mathsf{BHist}$ satisfy $Z_a(x)$. Product closure applied to $Z_a(\mathsf{inv}(t))$ and $Z_a(x)$ gives $Z_a(\mathsf{mul}(\mathsf{inv}(t), x))$.

Apply product closure once more with the final factor $Z_a(t)$. The two-sided inverse laws identify t with $\mathsf{inv}(\mathsf{inv}(t))$ under hsame, and the carrier-transport row of Definition 41.87 transports membership to $Z_a(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(t), x), \mathsf{inv}(\mathsf{inv}(t))))$. This is the second clause of $\mathsf{Norm}_{Z_a}(t)$, so both preservation directions hold. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizer self normalizer from empty unit`
Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerCarrier self normalizer from empty unit`
Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerCarrier self normalizes`

Theorem 80.20 (Centralizer intersection simultaneous self-normalizer)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, fix* $a, b : \mathsf{BHist}$ *and let* $Z_{a,b}$ *be the two-centralizer carrier of Definition 80.3. If* $Z_{a,b}(t)$*, then*

$$\mathsf{Norm}_{Z_a}(t) \wedge \mathsf{Norm}_{Z_b}(t).$$

Proof. Unfold the intersection carrier. The witness $Z_{a,b}(t)$ is exactly a pair consisting of $Z_a(t)$ and $Z_b(t)$, with both component centralizers using the same ambient $\mathsf{Group}^{\uparrow}$ laws and the same restricted hsame classifier.

Apply Definition 80.19 first with fixed history a and component witness $Z_a(t)$. This gives $\mathsf{Norm}_{Z_a}(t)$.

Apply Definition 80.19 again with fixed history b and component witness $Z_b(t)$. Pair the two resulting normalizer witnesses to obtain the displayed simultaneous conclusion. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerIntersectionCarrier simultaneous self normalizer from empty unit`

Per-object template: traditional target simultaneous normalizer membership for an intersection of centralizers; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$; carrier $Z_{a,b}$; classifier restricted hsame; stability inherited from the two centralizer certificates; ledger fixed histories a, b and two self-normalizer witnesses; core theorem Definition 80.20; exactness the paired normalizer conclusion; downstream consumer quotient-ready normalizer towers; status $\mathsf{ProofTarget}$.

Lemma 80.21 (Centralizer normalizer kernel classifier reflexivity)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, if* $x : \mathsf{BHist}$ *is carried by the centralizer normalizer* N_a*, then the normalizer endpoint classifies with itself and the inverse-left kernel word is carried by the centralizer:*

$$N_a(x) \Rightarrow N_a(x) \wedge N_a(x) \wedge Z_a(\mathsf{mul}(\mathsf{inv}(x), x)).$$

Proof. The two normalizer endpoint rows are the given $N_a(x)$ witness. For the kernel word, use the centralizer certificate target of Definition 80.2: its identity row gives $Z_a(\mathsf{e})$ and its hsame-transport row transports this membership along the inverse-left law $\mathsf{mul}(\mathsf{inv}(x), x)\mathsf{hsame}\mathsf{e}$ read in the reverse direction. Thus the inverse-left kernel endpoint lies in the lower centralizer carrier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer kernel classifier refl`

Per-object template: traditional target centralizer self-normalization; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist plus $\mathsf{Subgroup}^{\uparrow}$; carrier Z_a; classifier restricted hsame; stability product and inverse closure; ledger conjugating element, acted element, and inverse witness; core theorem list Definition 41.87 and Definition 80.2; exactness predicate normalizer witness; standard bridge ordinary centralizer inclusion in its normalizer; status $\mathsf{ProofTarget}$.

Definition 80.22 (Centralizer-normalizer SubgroupUp pair)**.** *Fix the centralizer carrier Z_a from Definition 41.84 and the centralizer normalizer carrier N_a from Definition 41.95. The centralizer-normalizer* $\mathsf{Subgroup}^{\uparrow}$ *pair has lower carrier Z_a and upper carrier N_a, both classified by the restriction of* hsame *to carried histories. The operation, unit, and inverse are inherited from the ambient history-level* $\mathsf{Group}^{\uparrow}$ *operation* mul*, e, and* inv*.*

The pair ledger records the fixed element a, the conjugation word $\mathsf{Conj}_s(x)$, the witnesses that histories carried by Z_a commute with a, and the forward and inverse preservation witnesses appearing in the definition of N_a. Thus a carried normalizer history acts on the carried centralizer histories by the restricted conjugation word while keeping every endpoint inside the same $\mathsf{Subgroup}^{\uparrow}$ *lower carrier.*

Theorem 80.23 (Centralizer-normalizer SubgroupUp pair certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 41.98, the pair of Definition 80.22 satisfies the following concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate rows:*

1. *Z_a is the centralizer* $\mathsf{Subgroup}^{\uparrow}$ *certificate target of Definition 80.2.*
2. *N_a is a* $\mathsf{Subgroup}^{\uparrow}$ *certificate target over the same ambient operation, with unit, product, inverse, and classifier-transport rows.*
3. *The lower carrier is included in the upper carrier: if $Z_a(s)$, then $N_a(s)$.*
4. *If $N_a(s)$ and $Z_a(x)$, then $Z_a(\mathsf{Conj}_s(x))$ and $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.*
5. *For every s with $N_a(s)$, the restricted normalizer action on Z_a is classifier-exact, its orbit classifier is stable under* hsame *replacement of carried endpoints, and its fixed-action kernel satisfies*

$$(\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow \mathsf{hsame}(\mathsf{Conj}_s(x), x)) \iff (\forall x : \mathsf{BHist},\ Z_a(x) \Rightarrow \mathsf{hsame}(\mathsf{mul}(s, x), \mathsf{mul}(x, s))).$$

Hence the centralizer and its normalizer are available as one concrete $\mathsf{Subgroup}^{\uparrow}$ *pair over* BHist*, with the restricted conjugation action and orbit classifier attached to the same certificate object.*

Proof. The lower certificate is Definition 80.2. It gives the unit, product, inverse, classifier-transport, and commutator-fiber rows for Z_a with the operation, unit, inverse, and restricted hsame classifier inherited from the ambient $\mathsf{Group}^{\uparrow}$.

The upper certificate and the carrier inclusion are supplied by Definition 41.98. Its first four rows give unit, product, inverse, and classifier transport for N_a, and its final row places each centralizer element in the normalizer.

The action rows are the defining preservation clauses of N_a, recorded in Definition 41.100. That theorem also gives the inverse-action identities and the displayed fixed-action kernel equivalence, so the restricted action is exact on the centralizer carrier.

Finally, Definition 41.109 supplies the endpoint-transport row for the orbit classifier generated by the same normalizer action. Repacking these cited rows with the ledger of Definition 80.22 gives one concrete $\mathsf{Subgroup}^{\uparrow}$ pair over BHist. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer pair certificate from empty unit`

Per-object template: traditional target normalizer of a centralizer with restricted action; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist plus $\mathsf{Subgroup}^{\uparrow}$; carriers Z_a, N_a; classifier restricted hsame; stability subgroup rows, inclusion, conjugation preservation, and action exactness; ledger fixed element a, conjugating word Conj_s, normalizer witness, and endpoint witnesses; core theorem list Definition 80.2, Definition 41.98, Definition 41.100, and Definition 41.109; exactness predicate normalizer action on the centralizer carrier; standard bridge ordinary centralizer-normalizer pair; status $\mathsf{ProofTarget}$. The sibling route for $\mathsf{Subgroup}^{\uparrow}$ is the algebraic ladder already displayed by $\mathsf{Group}^{\uparrow}$ and $\mathsf{Monoid}^{\uparrow}$. The subgroup packet consumes the ambient carrier, operation, identity, inverse, and closure rows from chapter 41, while its operation and unit layer is inherited through the monoid certificate of chapter 40; the carried subset is therefore a refinement of that ambient algebraic surface rather than a detached kernel entry.

Theorem 80.24 (Subgroup group quotient dependency route)**.** *Every accepted* $\mathsf{Subgroup}^{\uparrow}$ *packet reads its ambient carrier, operation, unit, inverse, and closure rows through the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41 and the operation-unit inheritance route of chapter 40. Any quotient-facing consumer must then route the restricted classifier through chapter 82, rather than treating subgroup data as a kernel-only record detached from the ambient algebra.*

Proof. Start with an accepted subgroup packet. Its carrier is a carried subset of an ambient algebraic source, so the operation, unit, inverse, and closure rows are readable only after the $\mathsf{Group}^{\uparrow}$ carrier and operation surface has been displayed.

The operation and unit rows retain the monoid layer. Associativity, unit compatibility, and product closure are inherited through the $\mathsf{Monoid}^{\uparrow}$ route before inverse and subgroup-specific closure rows are inspected.

For quotient-facing reads, the restricted classifier is not enough by itself. The consumer must expose the $\mathsf{QuotientGroup}^{\uparrow}$ route so that endpoint membership, quotient classifier stability, multiplication compatibility, and representative independence are read as quotient-group rows over the displayed subgroup packet.

Thus the substantive dependency route is $\mathsf{Group}^{\uparrow}$ and $\mathsf{Monoid}^{\uparrow}$ for ambient algebra, followed by $\mathsf{QuotientGroup}^{\uparrow}$ for quotient consumers. The subgroup packet is not a kernel-only object and does not export quotient structure without that route. □

Quotient-facing $\mathsf{Subgroup}^{\uparrow}$ consumers read the ambient operation rows of chapter 41, the unit and multiplication discipline of chapter 40, and the classifier boundary of chapter 82 before any normalizer or kernel route is accepted. The dependency route keeps $\mathsf{Subgroup}^{\uparrow}$ inside the displayed algebra lattice: a normal-subgroup consumer may reuse the subgroup carrier only after the group, monoid, and quotient-group rows have fixed the operation, identity, equivalence, and kernel-facing classifier surface.

Theorem 80.25 (Centralizer normalizer SubgroupUp certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, the carrier* Norm_{Z_a} *of Definition 80.16 supplies the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate rows for the normalizer of* Z_a*:*

1. $\mathsf{Norm}_{Z_a}(e)$.
2. *If* $\mathsf{Norm}_{Z_a}(s)$ *and* $\mathsf{Norm}_{Z_a}(t)$*, then* $\mathsf{Norm}_{Z_a}(\mathsf{mul}(s,t))$.
3. *If* $\mathsf{Norm}_{Z_a}(t)$*, then* $\mathsf{Norm}_{Z_a}(\mathsf{inv}(t))$.

4. *If* $\mathsf{Norm}_{Z_a}(s)$ *and* $s\mathsf{hsame}t$*, then* $\mathsf{Norm}_{Z_a}(t)$*.*

5. *If* $Z_a(t)$*, then* $\mathsf{Norm}_{Z_a}(t)$*.*

Thus the two-sided normalizer carrier has identity membership, product closure, inverse closure, hsame*-transport, and the inclusion row from the centralizer.*

Proof. For identity membership, unfold Definition 80.16. The unit laws make conjugation by e act as the identity on every carried x, and the two-sided inverse laws identify $\mathsf{inv}(e)$ with e. Carrier transport from Definition 41.87 returns both clauses.

For product closure, assume $\mathsf{Norm}_{Z_a}(s)$ and $\mathsf{Norm}_{Z_a}(t)$, reading these as expanded N_a witnesses. Apply Definition 41.97: its forward composition row sends $Z_a(x)$ first through the t clause and then through the s clause, while its inverse-action row sends $Z_a(x)$ first through the inverse clause for s and then through the inverse clause for t. These are the two clauses of $\mathsf{Norm}_{Z_a}(\mathsf{mul}(s,t))$.

For inverse closure, the first clause of $\mathsf{Norm}_{Z_a}(\mathsf{inv}(t))$ is exactly the inverse-conjugation half of $\mathsf{Norm}_{Z_a}(t)$. Its second clause conjugates by $\mathsf{inv}(\mathsf{inv}(t))$; the two-sided inverse laws identify this word with the original conjugation by t, and multiplication congruence plus carrier transport move the first clause across the comparison.

For hsame-transport, assume $\mathsf{Norm}_{Z_a}(s)$ and $s\mathsf{hsame}t$. Multiplication congruence transports the forward conjugation word with s to the corresponding word with t, and inverse congruence transports the inverse word. Carrier transport from Definition 41.87 therefore gives both clauses of $\mathsf{Norm}_{Z_a}(t)$.

The inclusion row is exactly Definition 80.19: every $Z_a(t)$ supplies both normalizer clauses. Repacking the five rows with the restricted hsame classifier gives the advertised concrete $\mathsf{Subgroup}^{\uparrow}$ certificate for the normalizer carrier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer certificate from empty unit`

Per-object template: traditional target normalizer subgroup certificate; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist plus centralizer $\mathsf{Subgroup}^{\uparrow}$; carrier Norm_{Z_a}; classifier restricted hsame; stability identity, product, inverse, and carrier transport; ledger unit conjugation, product conjugation composition, inverse conjugation, and endpoint classifier witnesses; core theorem list Definition 41.97, Definition 41.87, and Definition 80.19; exactness predicate two-sided preservation of Z_a under conjugation; standard bridge ordinary normalizer subgroup of a centralizer; status ProofTarget.

Theorem 80.26 (Centralizer normalizer hsame transport)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $N_a(s)$ *and* $s\mathsf{hsame}t$*, then* $N_a(t)$*.*

Proof. For every centralizer input x, the normalizer witness for s gives carried membership of the conjugation word $\mathsf{mul}(\mathsf{mul}(s,x),\mathsf{inv}(s))$. Multiplication congruence and inverse congruence transport this word to $\mathsf{mul}(\mathsf{mul}(t,x),\mathsf{inv}(t))$. The centralizer hsame-transport row from the $\mathsf{Subgroup}^{\uparrow}$ certificate for Z_a gives the required carried membership for t. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer hsame transport from empty unit`

Theorem 80.27 (Centralizer normalizer SubgroupUp tower certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, let* Z_a *be the centralizer carrier of Definition 41.84 and let* N_a *be the centralizer normalizer carrier of Definition 41.95. The pair* $Z_a \leq N_a$ *forms a quotient-ready* $\mathsf{Subgroup}^{\uparrow}$ *tower:*

1. Z_a *satisfies the identity, product, inverse,* hsame*-transport, and commutator-fiber rows of the centralizer* $\mathsf{Subgroup}^{\uparrow}$ *certificate.*

2. *N_a satisfies the identity, product, inverse, and* hsame*-transport subgroup rows over the same ambient classifier.*
3. *Every carried centralizer element is carried by the normalizer: if $Z_a(t)$, then $N_a(t)$.*
4. *If $N_a(s)$ and $Z_a(x)$, then $Z_a(\mathsf{Conj}_s(x))$ and $Z_a(\mathsf{Conj}_{\mathsf{inv}(s)}(x))$.*
5. *The two conjugation-closure rows are stable under the ambient classifier: if $\mathsf{hsame}(s,t)$, then the corresponding forward and inverse conjugation words define the same carried Z_a membership after transport.*

Equivalently, the ten-field $\mathsf{Subgroup}^{\uparrow}$ certificate for Z_a, the four-field subgroup certificate for N_a, the inclusion $Z_a \subseteq N_a$, the two conjugation rows, and their classifier stability are packaged as the normal-subgroup input used by quotient-group construction.

Proof. The Z_a certificate is exactly Definition 80.2. It supplies the identity carrier, product closure, inverse closure, hsame transport, and commutator-fiber exactness rows for the centralizer object viewed as a $\mathsf{Subgroup}^{\uparrow}$ instance.

The N_a subgroup rows come from Definition 41.98. Its identity, product, inverse, and hsame-transport clauses give the normalizer carrier the same ambient operation, unit, inverse, and classifier interface required for the upper subgroup object.

The inclusion row has two equivalent sources. The fifth row of Definition 41.98 states directly that $Z_a(t)$ implies $N_a(t)$, while Definition 80.19 gives the same witness by showing that every centralizer element preserves Z_a under conjugation.

Finally, Definition 41.99 supplies the forward conjugation row, the inverse-conjugation row, and the classifier transport of those rows. Repacking these data in the order displayed above gives the quotient-ready tower certificate without introducing any host quotient or external parameter transport. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizer tower certificate`

Per-object template: traditional target normal centralizer subgroup inside its normalizer for quotient groups; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist plus $\mathsf{Subgroup}^{\uparrow}$; lower carrier Z_a; upper carrier N_a; classifier hsame; stability identity, product, inverse, classifier transport, inclusion, forward conjugation, inverse conjugation, and conjugation-word transport; ledger centralizing element, normalizing element, acted element, inverse normalizer witness, and carried conjugation witnesses; core theorem list Definition 80.2, Definition 41.98, Definition 80.19, and Definition 41.99; exactness predicate normalizer action on the centralizer; standard bridge normal-subgroup input for quotient groups; status ProofTarget.

Theorem 80.28 (Subgroup normalizer orbit quotient classifier)**.** *For a BEDC history-level $\mathsf{Group}^{\uparrow}$ operation over* BHist *satisfying the BEDC group laws used by Definition 80.27 and Definition 41.104, including associativity, unit, multiplication-congruence, inverse-congruence, and two-sided inverse laws, let Z_a be the centralizer carrier, let N_a be the centralizer normalizer carrier, let $\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s,x),\mathsf{inv}(s))$, and let $x \sim_{N_a} y$ be the normalizer-orbit relation of Definition 41.103. For x, y, z : BHist with $Z_a(x)$, $Z_a(y)$, and $Z_a(z)$, the data of Definition 80.27 together with Definition 41.104 supply the quotient-free classifier input for $\mathsf{QuotientGroup}^{\uparrow}$:*

1. *the lower $\mathsf{Subgroup}^{\uparrow}$ certificate rows for Z_a;*
2. *the upper subgroup rows for N_a over the same history operation;*
3. *the inclusion row $Z_a \subseteq N_a$;*
4. *the forward and inverse conjugation stability of Z_a under histories carried by N_a;*

5. reflexivity, symmetry, and transitivity of $\sim_{N_a}$ on carried histories of Z_a.

Thus the normalizer action on the centralizer carries exactly the quotient-free orbit classifier required before any quotient-group carrier is introduced.

Proof. Apply Definition 80.27. Its displayed rows give the lower certificate for Z_a, the upper subgroup rows for N_a, the inclusion $Z_a \subseteq N_a$, and the two conjugation-closure rows for Conj_s and $\mathsf{Conj}_{\mathsf{inv}(s)}$.

Next use Definition 41.103 to read $\sim_{N_a}$ as reachability by a carried normalizer history with an hsame endpoint comparison. This keeps the classifier entirely at the history level.

Finally apply Definition 41.104 to the carried endpoints x, y, and z. Its three laws give reflexivity, symmetry, and transitivity for the same relation $\sim_{N_a}$ on Z_a, so the tower rows and the orbit laws combine into the advertised $\mathsf{QuotientGroup}^{\uparrow}$ classifier input. □

Per-object template: traditional target normalizer-orbit quotient classifier; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ tower; carrier Z_a; classifier $\sim_{N_a}$; stability equivalence plus conjugation closure; ledger normalizer witness and conjugation endpoint; exactness quotient-free orbit reachability; standard bridge ordinary normal-subgroup orbit quotient input; status $\mathsf{ProofTarget}$.

Definition 80.29 (Abstract two-subgroup intersection carrier)**.** *Fix a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *source over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and ambient classifier* hsame*. Let* $H_1, H_2 : \mathsf{BHist} \to \mathsf{Prop}$ *be two predicates read as* $\mathsf{Subgroup}^{\uparrow}$ *carriers over this same source. Define their two-subgroup intersection carrier by*

$$H_{1,2}(x) :\Longleftrightarrow H_1(x) \wedge H_2(x).$$

The inherited restricted classifier on this carrier is

$$R_{1,2}(x, y) :\Longleftrightarrow H_{1,2}(x) \wedge H_{1,2}(y) \wedge x\mathsf{hsame}y.$$

The operation, unit, inverse, and history classifier are inherited from the ambient $\mathsf{Group}^{\uparrow}$ *source.*

Theorem 80.30 (Two-subgroup intersection SubgroupUp certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and ambient classifier* hsame*, assume the monoid stability rows of Definition 40.13, the inverse rows of Definition 41.55, and the history-sameness equivalence of Definition 3.8. Let* $H_1, H_2 : \mathsf{BHist} \to \mathsf{Prop}$ *be two carrier predicates, and let* $H_{1,2}$ *and* $R_{1,2}$ *be the carrier and restricted classifier of Definition 80.29. Assume that each* H_i*, for* $i \in \{1, 2\}$*, satisfies the ambient* $\mathsf{Subgroup}^{\uparrow}$ *carrier rows:*

1. $H_i(e)$*.*

2. If $H_i(x)$ *and* $H_i(y)$*, then* $H_i(\mathsf{mul}(x, y))$*.*

3. If $H_i(x)$*, then* $H_i(\mathsf{inv}(x))$*.*

4. If $H_i(x)$ *and* $x\mathsf{hsame}y$*, then* $H_i(y)$*.*

Then the carrier $H_{1,2}$ *with classifier* $R_{1,2}$ *satisfies the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate rows:*

1. $R_{1,2}$ *is a semantic naming classifier on* $H_{1,2}$*.*

2. $H_{1,2}(e)$*.*

3. If $H_{1,2}(x)$ *and* $H_{1,2}(y)$*, then* $H_{1,2}(\mathsf{mul}(x, y))$*.*

4. If $H_{1,2}(x)$*, then* $H_{1,2}(\mathsf{inv}(x))$*.*

5. *If* $H_{1,2}(x)$ *and* $x\mathsf{hsame}y$*, then* $H_{1,2}(y)$*.*
6. *If* $R_{1,2}(x, x')$ *and* $R_{1,2}(y, y')$*, then*
$$R_{1,2}(\mathsf{mul}(x, y), \mathsf{mul}(x', y')).$$
7. *If* $R_{1,2}(x, y)$*, then*
$$R_{1,2}(\mathsf{inv}(x), \mathsf{inv}(y)).$$

Thus the intersection of two carried $\mathsf{Subgroup}^{\uparrow}$ *predicates over the same ambient* $\mathsf{Group}^{\uparrow}$ *source is again a concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate target over the restricted* hsame *classifier.*

Proof. First verify the hypotheses of Definition 80.70 for the predicate $P = H_{1,2}$. The inhabitance row is $H_{1,2}(e)$, obtained by pairing the two assumed unit rows $H_1(e)$ and $H_2(e)$.

For carrier stability, assume $H_{1,2}(x)$ and $x\mathsf{hsame}y$. Unfolding $H_{1,2}(x)$ gives $H_1(x)$ and $H_2(x)$. The transport row for H_1 gives $H_1(y)$, and the transport row for H_2 gives $H_2(y)$. Pairing these conclusions gives $H_{1,2}(y)$.

For one-step relative-product closure, assume $H_{1,2}(x)$ and $H_{1,2}(y)$. Unfolding gives $H_i(x)$ and $H_i(y)$ for each $i \in \{1, 2\}$. The inverse row for H_i gives $H_i(\mathsf{inv}(y))$, and then the product row for H_i gives
$$H_i(\mathsf{mul}(x, \mathsf{inv}(y))).$$
Doing this for $i = 1$ and $i = 2$ and pairing the two results proves
$$H_{1,2}(\mathsf{mul}(x, \mathsf{inv}(y))).$$
Therefore Definition 80.70 applies to $H_{1,2}$ and yields the identity, product, inverse, and carrier-transport rows listed above.

It remains to record the restricted-classifier rows. For semantic naming, unfold $R_{1,2}$. If $H_{1,2}(x)$, reflexivity of hsame from Definition 3.8 gives $x\mathsf{hsame}x$, so $R_{1,2}(x, x)$ holds. If $R_{1,2}(x, y)$ holds, keep the two endpoint carrier witnesses and reverse the stored hsame comparison by symmetry, giving $R_{1,2}(y, x)$. If $R_{1,2}(x, y)$ and $R_{1,2}(y, z)$ hold, keep the carried endpoints $H_{1,2}(x)$ and $H_{1,2}(z)$ and compose the stored comparisons by transitivity of hsame, giving $R_{1,2}(x, z)$. Hence $R_{1,2}$ is a semantic naming classifier on the intersection carrier.

For multiplication compatibility, assume $R_{1,2}(x, x')$ and $R_{1,2}(y, y')$. The already proved product row for $H_{1,2}$ supplies the endpoint witnesses
$$H_{1,2}(\mathsf{mul}(x, y))$$
and
$$H_{1,2}(\mathsf{mul}(x', y')).$$
The multiplication congruence row from Definition 40.13 transports the stored comparisons $x\mathsf{hsame}x'$ and $y\mathsf{hsame}y'$ to
$$\mathsf{mul}(x, y)\mathsf{hsamemul}(x', y').$$
These three fields are exactly $R_{1,2}(\mathsf{mul}(x, y), \mathsf{mul}(x', y'))$.

For inverse compatibility, assume $R_{1,2}(x, y)$. The already proved inverse row supplies $H_{1,2}(\mathsf{inv}(x))$ and $H_{1,2}(\mathsf{inv}(y))$. The ambient inverse congruence theorem Definition 41.60 transports the stored comparison $x\mathsf{hsame}y$ to
$$\mathsf{inv}(x)\mathsf{hsameinv}(y).$$
Repacking gives $R_{1,2}(\mathsf{inv}(x), \mathsf{inv}(y))$.

The carrier, classifier, operation, unit, and inverse rows are therefore the concrete $\mathsf{Subgroup}^{\uparrow}$ row format used in Definition 80.2 and Definition 80.68. □

Definition 80.31 (Centralizer normalizer quotient classifier)**.** *For the centralizer carrier Z_a and its normalizer carrier N_a, define the concrete quotient classifier predicate $Q_{Z_a \le N_a}$ on* BHist *by*

$$Q_{Z_a \le N_a}(x, y) :\Longleftrightarrow N_a(x) \wedge N_a(y) \wedge \exists z : \mathsf{BHist},\ Z_a(z) \wedge \mathsf{hsame}(y, \mathsf{mul}(x, z)).$$

The endpoints are carried by N_a, while the right witness is carried by Z_a. Thus $Q_{Z_a \le N_a}$ is a classifier predicate over concrete BEDC histories, not a host quotient object.

Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier`
Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientClassifier`
Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightCosetClassifier`

Theorem 80.32 (Centralizer normalizer quotient classifier laws)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.2, the classifier $Q_{Z_a \le N_a}$ of Definition 80.31 has the quotient classifier laws on N_a:*

1. *if $N_a(x)$, then $Q_{Z_a \le N_a}(x, x)$;*
2. *if $Q_{Z_a \le N_a}(x, y)$, then $Q_{Z_a \le N_a}(y, x)$;*
3. *if $Q_{Z_a \le N_a}(x, y)$ and $Q_{Z_a \le N_a}(y, u)$, then $Q_{Z_a \le N_a}(x, u)$;*
4. *if $Q_{Z_a \le N_a}(x, y)$ and $Q_{Z_a \le N_a}(u, v)$, then $Q_{Z_a \le N_a}(\mathsf{mul}(x, u), \mathsf{mul}(y, v))$.*

Proof. The endpoint rows are supplied by Definition 80.27. The identity row of Definition 80.2 gives $Z_a(\mathsf{e})$, and the right-unit law rewrites $\mathsf{mul}(x, \mathsf{e})$ to x up to hsame. This is the required witness for reflexivity on N_a.

For symmetry, suppose $Q_{Z_a \le N_a}(x, y)$ is witnessed by z with $Z_a(z)$ and $\mathsf{hsame}(y, \mathsf{mul}(x, z))$. The inverse-closure row of Definition 80.2 gives $Z_a(\mathsf{inv}(z))$. The two-sided inverse laws and multiplication congruence transport the displayed witness equation to $\mathsf{hsame}(x, \mathsf{mul}(y, \mathsf{inv}(z)))$, so the inverse witness proves $Q_{Z_a \le N_a}(y, x)$.

For transitivity, take witnesses z and w for $Q_{Z_a \le N_a}(x, y)$ and $Q_{Z_a \le N_a}(y, u)$. Product closure for Z_a gives $Z_a(\mathsf{mul}(z, w))$. Associativity and multiplication congruence compose the two classifier equations into $\mathsf{hsame}(u, \mathsf{mul}(x, \mathsf{mul}(z, w)))$, giving $Q_{Z_a \le N_a}(x, u)$.

It remains to check multiplication compatibility. Let z witness $Q_{Z_a \le N_a}(x, y)$ and let w witness $Q_{Z_a \le N_a}(u, v)$. Since u is carried by N_a, the normality row in Definition 80.27, equivalently Definition 41.99, moves z across the right endpoint and gives $Z_a(\mathsf{Conj}_{\mathsf{inv}(u)}(z))$.

Product closure for Z_a combines this conjugated witness with w. The associativity, inverse, and multiplication-congruence laws rewrite $\mathsf{mul}(y, v)$ to $\mathsf{mul}(\mathsf{mul}(x, u), \mathsf{mul}(\mathsf{Conj}_{\mathsf{inv}(u)}(z), w))$ up to hsame. Together with product closure of N_a from Definition 41.98, this gives $Q_{Z_a \le N_a}(\mathsf{mul}(x, u), \mathsf{mul}(y, v))$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier symm from empty unit`

Definition 80.33 (Centralizer normalizer quotient-kernel relation)**.** *In the setting of Definition 80.27, define the relation on ambient histories by*

$$Q_{Z_a \triangleleft N_a}(x, y) :\Longleftrightarrow N_a(x) \wedge N_a(y) \wedge Z_a(\mathsf{mul}(\mathsf{inv}(x), y)).$$

The first two clauses keep both representatives inside the normalizer carrier, and the third clause says that their relative displacement lies in the centralizer kernel.

Lean4 (definition) `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel`

Theorem 80.34 (Centralizer right coset classifier quotient-kernel equivalence)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the concrete right-coset classifier of Definition 80.31 coincides with the quotient-kernel relation of Definition 80.33:*

$$Q_{Z_a \leq N_a}(x, y) \iff Q_{Z_a \triangleleft N_a}(x, y).$$

Proof. For the forward direction, unpack the right-coset witness. The endpoint clauses already give $N_a(x)$ and $N_a(y)$, and the witness gives some $z : \mathsf{BHist}$ with $Z_a(z)$ and $\mathsf{hsame}(y, \mathsf{mul}(x, z))$.

Left-multiply the endpoint comparison by $\mathsf{inv}(x)$ and reassociate. The left inverse and unit laws identify the resulting word with z up to hsame. Multiplication congruence and the centralizer carrier-transport row then move $Z_a(z)$ to $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$, which is the quotient-kernel clause.

For the reverse direction, assume the quotient-kernel relation. Take the kernel word $z := \mathsf{mul}(\mathsf{inv}(x), y)$ as the right-coset witness.

Associativity, the inverse law, the unit law, and multiplication congruence transport $\mathsf{mul}(x, \mathsf{mul}(\mathsf{inv}(x), y))$ back to y up to hsame. The kernel carrier is exactly $Z_a(z)$, so the same word repacks the right-coset classifier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightCosetClassifier quotientKernel iff`

Theorem 80.35 (Centralizer normalizer quotient-kernel identity fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \triangleleft N_a}(\mathsf{e}, x) \iff Z_a(x).$$

Proof. Unfold the classifier in Definition 80.33. For the forward direction, the endpoint clauses give $N_a(\mathsf{e})$ and $N_a(x)$, and the kernel clause gives $Z_a(\mathsf{mul}(\mathsf{inv}(\mathsf{e}), x))$.

The inverse and unit laws identify $\mathsf{mul}(\mathsf{inv}(\mathsf{e}), x)$ with x up to hsame. The Z_a carrier-transport row from Definition 80.27 carries the kernel witness to $Z_a(x)$.

For the reverse direction, assume $Z_a(x)$. The inclusion row $Z_a \subseteq N_a$ in Definition 80.27 gives $N_a(x)$, and the identity row gives $N_a(\mathsf{e})$.

It remains to supply the kernel clause for $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$. The same inverse and unit laws identify x with $\mathsf{mul}(\mathsf{inv}(\mathsf{e}), x)$ up to hsame, and carrier transport for Z_a moves the assumed $Z_a(x)$ to the displayed kernel word. □

↪ variant `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel empty fiber iff`

Per-object template: traditional target identity coset and unit fiber for the normal-subgroup tower $Z_a \leq N_a$; BEDC source $\mathsf{Group}^{\uparrow}$ plus the $\mathsf{Subgroup}^{\uparrow}$ tower; carrier N_a; classifier $Q_{Z_a \triangleleft N_a}$; exactness identity fiber equals lower carrier; downstream object $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Theorem 80.36 (Centralizer quotient-kernel hsame transport)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Q_{Z_a \triangleleft N_a}(x, y)$*,* $x \mathsf{hsame} x'$*, and* $y \mathsf{hsame} y'$*, then* $Q_{Z_a \triangleleft N_a}(x', y')$*.*

Proof. The endpoint clauses transport by Definition 80.26. For the kernel clause, inverse congruence transports $\mathsf{inv}(x)$ to $\mathsf{inv}(x')$, multiplication congruence transports $\mathsf{mul}(\mathsf{inv}(x), y)$ to $\mathsf{mul}(\mathsf{inv}(x'), y')$, and the centralizer carrier hsame-transport row moves the stored kernel witness across this comparison. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel hsame transport`

Theorem 80.37 (Centralizer quotient-kernel left endpoint right centralizer multiplication)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Q_{Z_a \triangleleft N_a}(x, y)$ *and* $Z_a(c)$*, then*

$$Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, c), y).$$

Proof. The endpoint $\mathsf{mul}(x, c)$ lies in N_a because x is already carried by N_a, every centralizer element is carried by N_a, and the normalizer is closed under multiplication. For the kernel word, inverse closure gives $Z_a(\mathsf{inv}(c))$, while the original quotient-kernel hypothesis gives $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$. Product closure for Z_a therefore gives $Z_a(\mathsf{mul}(\mathsf{inv}(c), \mathsf{mul}(\mathsf{inv}(x), y)))$. Associativity and inverse-of-product reversal identify this history with $\mathsf{mul}(\mathsf{inv}(\mathsf{mul}(x, c)), y)$ up to hsame, and the centralizer carrier transport row gives the displayed kernel clause. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel left endpoint right centralizer mul closed from empty unit`

Theorem 80.38 (Centralizer quotient-kernel right endpoint right centralizer multiplication)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Q_{Z_a \triangleleft N_a}(x, y)$ *and* $Z_a(c)$*, then*

$$Q_{Z_a \triangleleft N_a}(x, \mathsf{mul}(y, c)).$$

Proof. The endpoint $\mathsf{mul}(y, c)$ lies in N_a because the quotient-kernel hypothesis carries y in N_a, the inclusion $Z_a \subseteq N_a$ carries c in N_a, and product closure for N_a keeps their product in the carrier.

For the kernel word, the original quotient-kernel hypothesis gives $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$, and the endpoint factor gives $Z_a(c)$. Product closure for Z_a yields $Z_a(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(x), y), c))$. Associativity compares this history with $\mathsf{mul}(\mathsf{inv}(x), \mathsf{mul}(y, c))$, and the centralizer carrier transport row moves the witness across that hsame comparison. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel right endpoint right centralizer mul closed from empty unit`

Theorem 80.39 (Centralizer quotient-kernel two-sided centralizer endpoint multiplication)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Q_{Z_a \triangleleft N_a}(x, y)$*,* $Z_a(c)$*, and* $Z_a(d)$*, then*

$$Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, c), \mathsf{mul}(y, d)).$$

Proof. First apply Definition 80.37 with the centralizer factor c. This gives $Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, c), y)$.

Then apply Definition 80.38 with the centralizer factor d. The result is the displayed two-endpoint quotient-kernel relation. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel two sided centralizer endpoint mul closed from empty unit`

Per-object template: traditional target endpoint stability of a normal-subgroup quotient classifier; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier $Q_{Z_a \triangleleft N_a}$; stability centralizer endpoint multiplication; ledger left representative, right representative, and centralizer endpoint factors; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Theorem 80.40 (Centralizer quotient-kernel centralizer endpoint invariance)**.** *Under the same BEDC history-level* $\mathsf{Group}^{\uparrow}$ *laws, if* $Z_a(c)$ *and* $Z_a(d)$*, then*

$$Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, c), \mathsf{mul}(y, d)) \Longleftrightarrow Q_{Z_a \triangleleft N_a}(x, y).$$

Proof. For the direction from the original endpoints to the translated endpoints, apply Definition 80.39 with the endpoint factors c and d.

For the direction from the translated endpoints back to the original endpoints, use inverse closure for Z_a from Definition 80.27 to obtain $Z_a(\mathsf{inv}(c))$ and $Z_a(\mathsf{inv}(d))$. Apply Definition 80.39 to the translated quotient-kernel relation with these inverse factors.

The resulting relation has endpoints $\mathsf{mul}(\mathsf{mul}(x, c), \mathsf{inv}(c))$ and $\mathsf{mul}(\mathsf{mul}(y, d), \mathsf{inv}(d))$. Associativity, the two-sided inverse laws, the unit laws, and multiplication congruence compare these histories with x and y up to hsame. The transport theorem Definition 80.36 then moves the quotient-kernel witness back to $Q_{Z_a \triangleleft N_a}(x, y)$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel centralizer endpoint invariance`
Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel centralizer endpoint mul invariant from empty unit`

Per-object template: traditional target representative independence in a normal-subgroup quotient; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier $Q_{Z_a \triangleleft N_a}$; stability endpoint-factor invariance; ledger lower-centralizer factors and inverse factors; core theorem this equivalence plus the one-way endpoint rows and quotient-kernel transport; exactness representative choice does not change the quotient-kernel classifier; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$.

Theorem 80.41 (Centralizer normalizer quotient-kernel transitivity)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the relation* $Q_{Z_a \triangleleft N_a}$ *of Definition 80.33 is transitive:*

$$Q_{Z_a \triangleleft N_a}(x, y) \wedge Q_{Z_a \triangleleft N_a}(y, z) \Longrightarrow Q_{Z_a \triangleleft N_a}(x, z).$$

Proof. Expand the two hypotheses. Their endpoint rows provide the required $N_a(x)$ and $N_a(z)$ clauses. Their kernel rows give

$$Z_a(\mathsf{mul}(\mathsf{inv}(x), y)) \qquad \text{and} \qquad Z_a(\mathsf{mul}(\mathsf{inv}(y), z)).$$

Product closure for Z_a gives a centralizer witness for the product of these two relative kernel words. Associativity rewrites that product to

$$\mathsf{mul}(\mathsf{inv}(x), \mathsf{mul}(y, \mathsf{mul}(\mathsf{inv}(y), z))),$$

and the right inverse law for y followed by the left unit law cancels the middle $y, \mathsf{inv}(y)$ pair. The carrier transport row for Z_a moves the product witness to $Z_a(\mathsf{mul}(\mathsf{inv}(x), z))$, which is exactly the kernel clause for $Q_{Z_a \triangleleft N_a}(x, z)$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel trans from empty unit`

Theorem 80.42 (Centralizer normalizer quotient-kernel right identity fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \triangleleft N_a}(x, \mathsf{e}) \Longleftrightarrow Z_a(x).$$

Proof. Unfold the quotient-kernel relation. The forward direction starts from the kernel witness $Z_a(\mathsf{mul}(\mathsf{inv}(x), \mathsf{e}))$. The right unit law identifies this word with $\mathsf{inv}(x)$, inverse closure gives $Z_a(\mathsf{inv}(\mathsf{inv}(x)))$, and inverse involution transports it to $Z_a(x)$. Conversely, from $Z_a(x)$ the tower gives both normalizer endpoints, inverse closure gives $Z_a(\mathsf{inv}(x))$, and the right unit law transports that witness to the displayed kernel word. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel empty right fiber iff`

Theorem 80.43 (Centralizer normalizer quotient-kernel symmetry)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the quotient-kernel relation is symmetric:*

$$Q_{Z_a \triangleleft N_a}(x, y) \Longrightarrow Q_{Z_a \triangleleft N_a}(y, x).$$

Proof. The endpoint clauses swap directly. For the kernel clause, invert the witness $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$ using the inverse-closure row of Z_a. The inverse-product-reversal law identifies the resulting word with $\mathsf{mul}(\mathsf{inv}(y), \mathsf{inv}(\mathsf{inv}(x)))$, and inverse involution followed by carrier transport gives the displayed $Z_a(\mathsf{mul}(\mathsf{inv}(y), x))$ witness. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel symm from empty unit`

Theorem 80.44 (Append centralizer carrier unary word)**.** *For the concrete BEDC append operation, centralizing the visible singleton* $\mathsf{E1}(\mathsf{e})$ *forces the carried word to be unary:*

$$Z^{\mathsf{append}}_{\mathsf{E1}(\mathsf{e})}(w) \Longrightarrow \mathsf{UnaryHistory}(w).$$

Proof. Unfold the centralizer carrier as the displayed comparison $\mathsf{append}(w, \mathsf{E1}(\mathsf{e}))\mathsf{hsameappend}(\mathsf{E1}(\mathsf{e}), w)$. Induct on the concrete history w. The empty word is unary. A zero-headed word displays opposite visible constructors on the two sides, contradicting constructor separation. A one-headed word has matching $\mathsf{E1}$ constructors; injectivity descends the same centralizer comparison to the tail, and the induction hypothesis gives unary evidence. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerCarrier append e1 unit unary`

Theorem 80.45 (Append centralizer quotient-kernel unary word)**.** *Let the centralizer quotient-kernel relation be instantiated with the concrete BEDC append operation and centralizer seed* $\mathsf{E1}(\mathsf{e})$*. For any history endomap* inv *and histories* x, y*, the kernel relation forces the relative append word to be unary:*

$$Q^{\mathsf{append},\mathsf{inv}}_{\mathsf{E1}(\mathsf{e})}(x, y) \Longrightarrow \mathsf{UnaryHistory}(\mathsf{append}(\mathsf{inv}(x), y)).$$

Proof. Unfold the quotient-kernel row and write $w = \mathsf{append}(\mathsf{inv}(x), y)$. The centralizer carrier says $\mathsf{append}(w, \mathsf{E1}(\mathsf{e}))\mathsf{hsameappend}(\mathsf{E1}(\mathsf{e}), w)$. Induct on the concrete word w. The empty case is the unary generator. A zero-headed word puts an $\mathsf{E1}$ head on the left and an $\mathsf{E0}$ head on the right, contradicting constructor separation. A one-headed word descends by $\mathsf{E1}$ injectivity to the same statement for the tail, and the induction hypothesis supplies unary evidence. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel append e1 unit kernel unary`

Theorem 80.46 (Centralizer quotient-kernel multiplication compatibility)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the quotient-kernel relation is compatible with multiplication:*

$$Q_{Z_a \triangleleft N_a}(x, y) \wedge Q_{Z_a \triangleleft N_a}(u, v) \Longrightarrow Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, u), \mathsf{mul}(y, v)).$$

Proof. Translate both quotient-kernel hypotheses through the right quotient classifier equivalence. The right quotient classifier multiplication row supplies the product relation, and the same equivalence translates the product classifier back to the quotient-kernel relation. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel mul compatible from empty unit`

Theorem 80.47 (Centralizer normalizer quotient-kernel equivalence laws)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the relation* $Q_{Z_a \lhd N_a}$ *of Definition 80.33 is a quotient-kernel equivalence relation on the upper carrier:*

1. *if* $N_a(x)$*, then* $Q_{Z_a \lhd N_a}(x, x)$*;*
2. *if* $Q_{Z_a \lhd N_a}(x, y)$*, then* $Q_{Z_a \lhd N_a}(y, x)$*;*
3. *if* $Q_{Z_a \lhd N_a}(x, y)$ *and* $Q_{Z_a \lhd N_a}(y, z)$*, then* $Q_{Z_a \lhd N_a}(x, z)$*;*

Proof. Reflexivity uses the identity row of Definition 80.27. From $N_a(x)$ the endpoint clauses are immediate, and the two-sided inverse law identifies $\mathsf{mul}(\mathsf{inv}(x), x)$ with e. Transport along hsame carries the centralizer identity witness $Z_a(e)$ to the required kernel witness.

For symmetry, expand $Q_{Z_a \lhd N_a}(x, y)$. The endpoint clauses swap because both x and y already lie in N_a. The kernel witness $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$ is inverted by the inverse-closure row for Z_a, and inverse-product reversal identifies the inverse word with $\mathsf{mul}(\mathsf{inv}(y), x)$ after the ambient inverse laws. The hsame-transport row for Z_a gives the displayed symmetric witness.

For transitivity, take kernel witnesses $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$ and $Z_a(\mathsf{mul}(\mathsf{inv}(y), z))$. Product closure in Z_a gives a witness for their product. Associativity and the inverse laws cancel the middle factors y and $\mathsf{inv}(y)$, and transport turns the product witness into $Z_a(\mathsf{mul}(\mathsf{inv}(x), z))$. The endpoint clauses are the retained $N_a(x)$ and $N_a(z)$ clauses. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel classifier laws`

Theorem 80.48 (Centralizer quotient-kernel diagonal carrier exactness)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \lhd N_a}(x, x) \Longleftrightarrow N_a(x).$$

Proof. For the forward direction, unfold Definition 80.33. A witness for $Q_{Z_a \lhd N_a}(x, x)$ contains both endpoint carrier clauses in N_a, so projecting either endpoint gives $N_a(x)$.

For the reverse direction, assume $N_a(x)$. Apply the reflexivity row of Definition 80.47 to obtain $Q_{Z_a \lhd N_a}(x, x)$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel diagonal carrier exactness`

Per-object template: traditional target quotient classifier for the normal-subgroup tower $Z_a \leq N_a$; BEDC source $\mathsf{Group}^{\uparrow}$ over BHist plus $\mathsf{Subgroup}^{\uparrow}$; lower carrier Z_a; upper carrier N_a; classifier $Q_{Z_a \lhd N_a}$; stability identity/reflexivity, symmetry, transitivity, endpoint membership, kernel membership, and kernel transport; ledger left representative, right representative, relative kernel witness, and transported kernel witness; core theorem list Definition 80.27, Definition 80.47, Definition 80.48, Definition 41.65, and the ambient $\mathsf{Group}^{\uparrow}$ laws; exactness predicate relative displacement in Z_a inside N_a; downstream object $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Theorem 80.49 (Centralizer quotient-kernel centralizer representative step)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, if* $N_a(x)$ *and* $Z_a(c)$*, then*

$$Q_{Z_a \lhd N_a}(x, \mathsf{mul}(x, c)) \wedge Q_{Z_a \lhd N_a}(\mathsf{mul}(x, c), x).$$

Proof. From $N_a(x)$, the reflexivity row of Definition 80.47 gives the diagonal relation $Q_{Z_a \triangleleft N_a}(x, x)$.

For the forward representative step, apply Definition 80.38 to the diagonal relation with the centralizer factor c. Since the right endpoint of the diagonal relation is x, this gives $Q_{Z_a \triangleleft N_a}(x, \mathsf{mul}(x, c))$.

For the reverse representative step, apply Definition 80.37 to the same diagonal relation with the same factor c. This gives $Q_{Z_a \triangleleft N_a}(\mathsf{mul}(x, c), x)$, and the two displayed relations pair into the asserted conjunction. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel centralizer representative step`

Per-object template: traditional target representative generation in a normal-subgroup quotient; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier $Q_{Z_a \triangleleft N_a}$; stability one normalizer endpoint with one centralizer factor; ledger selected representative and lower-centralizer factor; core theorem this block plus Definition 80.47, Definition 80.38, and Definition 80.37; exactness centralizer-factor representative reachability; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$. The quotient-classifier and kernel-witness rows below consume the concrete $\mathsf{Group}^{\uparrow}$ operation surface of chapter 41: associativity, unit, inverse, multiplication-congruence, and carrier-transport rows are the ambient algebra used before the $\mathsf{Subgroup}^{\uparrow}$ carrier restricts the classifier.

Theorem 80.50 (Centralizer quotient classifier kernel equivalence)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x, y : \mathsf{BHist}$ *satisfies*

$$Q^{\mathrm{std}}_{Z_a \leq N_a}(x, y) \iff Q^{\mathrm{ker}}_{Z_a \triangleleft N_a}(x, y).$$

Proof. For the forward direction, unpack the standard quotient classifier. The endpoint clauses give $N_a(x)$ and $N_a(y)$, and the quotient witness gives a centralizer element whose right multiplication carries x to y.

It remains to expose the relative kernel word. Multiply the endpoint comparison on the left by $\mathsf{inv}(x)$; associativity, the left inverse law, the unit law, and multiplication congruence identify the witness with $\mathsf{mul}(\mathsf{inv}(x), y)$ up to hsame. Carrier transport for Z_a then moves centralizer membership to the displayed kernel carrier.

For the reverse direction, start from the relative kernel classifier. Its endpoint clauses are already the two normalizer clauses needed by the standard quotient classifier.

Choose the quotient witness $z := \mathsf{mul}(\mathsf{inv}(x), y)$. Associativity and the right inverse law identify $\mathsf{mul}(x, z)$ with y up to hsame, so the kernel clause supplies the carried centralizer witness and the classifier repacks. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientClassifier kernel iff`

Theorem 80.51 (Centralizer quotient-kernel witness classifier equivalence)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x, y : \mathsf{BHist}$ *satisfies*

$$Q^{\mathrm{wit}}_{Z_a \leq N_a}(x, y) \iff Q^{\mathrm{ker}}_{Z_a \triangleleft N_a}(x, y).$$

Here Q^{wit} *presents the quotient by a right witness* z *with* $Z_a(z)$ *and* $\mathsf{hsame}(y, \mathsf{mul}(x, z))$.

Proof. In the forward direction, unpack the right-witness classifier. Left-multiply the endpoint comparison by $\mathsf{inv}(x)$ and reassociate; the left inverse and unit laws reduce the exposed word to the witness z.

Carrier transport for Z_a moves $Z_a(z)$ across that comparison, producing $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$. Together with the two endpoint normalizer clauses this is the relative kernel classifier.

Conversely, take the relative kernel word itself as the right witness. Associativity, the right inverse law, and the unit law identify y with $\mathsf{mul}(x, \mathsf{mul}(\mathsf{inv}(x), y))$ up to hsame, so the same kernel carrier repacks as the witness classifier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel witness classifier iff`

Theorem 80.52 (Centralizer right quotient classifier kernel equivalence)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x, y : \mathsf{BHist}$ *satisfies*

$$Q^{\mathrm{right}}_{Z_a \leq N_a}(x, y) \iff Q^{\ker}_{Z_a \lhd N_a}(x, y).$$

Proof. For the forward direction, unpack the right quotient witness and expose $\mathsf{mul}(\mathsf{inv}(x), y)$ by multiplying the endpoint comparison on the left. Associativity, left inverse, unit, and multiplication congruence identify the exposed word with the carried witness.

Carrier transport converts that witness membership into the kernel carrier, and the two endpoint normalizer clauses pass through unchanged.

For the reverse direction, use the kernel word as the right quotient witness. Reassociation followed by the right inverse and unit laws gives the endpoint comparison required by the right quotient classifier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier kernel iff`

Theorem 80.53 (Centralizer normalizer quotient classifiers coincide)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x, y : \mathsf{BHist}$ *satisfies*

$$Q^{\cos}_{Z_a \leq N_a}(x, y) \iff Q^{\ker}_{Z_a \lhd N_a}(x, y).$$

Proof. This is the canonical coincidence row for the centralizer-normalizer quotient site. It reads the right-coset classifier and the relative-kernel classifier as two presentations of the same normalizer quotient relation. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerNormalizerQuotientClassifier coincide`

Theorem 80.54 (Centralizer quotient presentation equivalence package)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x, y : \mathsf{BHist}$ *has a quotient presentation independent of the chosen centralizer-normalizer classifier. The standard quotient classifier* $Q^{\mathrm{std}}_{Z_a \leq N_a}(x, y)$*, the right-witness classifier* $Q^{\mathrm{wit}}_{Z_a \leq N_a}(x, y)$*, the right-coset classifier* $Q^{\cos}_{Z_a \leq N_a}(x, y)$*, and the relative-kernel classifier* $Q^{\ker}_{Z_a \lhd N_a}(x, y)$ *are mutually pinned by the three displayed equivalence rows*

$$Q^{\mathrm{std}}_{Z_a \leq N_a}(x, y) \iff Q^{\ker}_{Z_a \lhd N_a}(x, y),$$

$$Q^{\mathrm{wit}}_{Z_a \leq N_a}(x, y) \iff Q^{\ker}_{Z_a \lhd N_a}(x, y),$$

$$Q^{\cos}_{Z_a \leq N_a}(x, y) \iff Q^{\ker}_{Z_a \lhd N_a}(x, y),$$

and the derived equivalence between the right-witness and right-coset presentations obtained through the shared quotient-kernel classifier.

Proof. Apply Theorem 80.50 to identify the standard classifier with the relative-kernel classifier. Apply Theorem 80.51 to identify the right-witness classifier with the same kernel classifier.

Apply Theorem 80.34 to identify the right-coset classifier with that kernel classifier. Apply Theorem 80.52 for the parallel right quotient presentation.

The right-witness/right-coset equivalence is obtained by composing the right-witness-to-kernel implication with the kernel-to-right-coset implication, and conversely composing the right-coset-to-kernel implication with the kernel-to-right-witness implication. Thus all four presentations classify the same pairs (x, y) in the normalizer carrier N_a.

The ten-field fit is: traditional target quotient presentation independence; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier the shared quotient-kernel relation; stability the checked equivalence rows above; ledger right witness, coset witness, and relative kernel word; core theorem this package; exactness mutual presentation equivalence; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$. □

Theorem 80.55 (Centralizer normalizer right quotient classifier transitivity)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, the right-witness quotient classifier* $Q_{Z_a \leq N_a}$ *is transitive:*

$$Q_{Z_a \leq N_a}(x, y) \wedge Q_{Z_a \leq N_a}(y, z) \Longrightarrow Q_{Z_a \leq N_a}(x, z).$$

Proof. Convert both right-witness classifier hypotheses to the relative-kernel classifier using the centralizer-normalizer quotient classifier equivalence. The kernel classifier is transitive by the empty-unit centralizer quotient kernel theorem: the two relative kernel words multiply, and the middle endpoint cancels by associativity, unit, congruence, and inverse laws. Convert the resulting kernel classifier back to the right-witness classifier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier trans from empty unit`

Theorem 80.56 (Centralizer right quotient classifier hsame transport)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, if* $Q_{Z_a \leq N_a}(x, y)$ *is presented by the right-witness quotient classifier and* $x \mathsf{hsame} x'$ *and* $y \mathsf{hsame} y'$*, then the same right-witness quotient classifier holds at* (x', y')*.*

Proof. Translate the right-witness quotient classifier through the quotient-kernel equivalence to obtain $Q_{Z_a \triangleleft N_a}(x, y)$. The quotient-kernel transport theorem moves both endpoints and the centralizer kernel row across the two hsame comparisons. Translating back through the same equivalence repackages the transported kernel as the right-witness quotient classifier at (x', y'). □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier hsame transport`

Theorem 80.57 (Centralizer right quotient product witness carrier)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying associativity, two-sided unit, multiplication-congruence, and two-sided inverse laws, let* u *be carried by the centralizer normalizer and let* p, q *be carried by the centralizer* Z_a*. Then the product witness*

$$\mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u), q)$$

is again carried by Z_a*.*

Proof. The normalizer row for u sends the centralizer witness p through the inverse conjugation word $\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), \mathsf{inv}(\mathsf{inv}(u)))$. The inverse-involution law transports this carrier witness to $\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u)$. Product closure for Z_a then multiplies the transported conjugation witness with the carried element q, giving the displayed product witness carrier. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier product witness carrier from empty unit`

Theorem 80.58 (Centralizer right quotient product witness endpoint)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying associativity, left unit, multiplication-congruence, and right inverse laws, if* $y\mathsf{hsamemul}(x, p)$ *and* $v\mathsf{hsamemul}(u, q)$*, then*

$$\mathsf{mul}(y, v)\mathsf{hsamemul}(\mathsf{mul}(x, u), \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u), q)).$$

Proof. Multiplication congruence first replaces y and v by their right quotient presentations. Associativity moves the middle u next to $\mathsf{inv}(u)$, the right inverse law contracts $\mathsf{mul}(u, \mathsf{inv}(u))$ to the empty unit, and the left-unit law removes that empty factor. A final associativity comparison restores the endpoint as $\mathsf{mul}(\mathsf{mul}(x, u), \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u), q))$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier product witness endpoint from empty unit`

Theorem 80.59 (Centralizer right quotient classifier multiplication compatibility)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying associativity, two-sided unit, multiplication-congruence, and two-sided inverse laws, the centralizer right-witness quotient classifier is compatible with multiplication:*

$$Q_{Z_a \leq N_a}(x, y) \wedge Q_{Z_a \leq N_a}(u, v) \Longrightarrow Q_{Z_a \leq N_a}(\mathsf{mul}(x, u), \mathsf{mul}(y, v)).$$

Proof. Unpack the two right quotient witnesses as centralizer elements p and q with $y\mathsf{hsamemul}(x, p)$ and $v\mathsf{hsamemul}(u, q)$. Product closure gives normalizer membership for $\mathsf{mul}(x, u)$ and $\mathsf{mul}(y, v)$.

The normalizer witness for u carries p through conjugation, so $\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u)$ is again a centralizer element. Multiplying it with q gives the new right quotient witness. The endpoint equation follows by associativity, unit laws, inverse cancellation, and multiplication congruence: $\mathsf{mul}(y, v)$ is transported to $\mathsf{mul}(\mathsf{mul}(x, u), \mathsf{mul}(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(u), p), u), q))$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier mul compatible from empty unit`

Theorem 80.60 (Centralizer right coset classifier hsame transport)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, and two-sided inverse laws as in Definition 80.27, if* $Q_{Z_a \leq N_a}(x, y)$ *is presented by the right-coset classifier and* $x\mathsf{hsame}x'$ *and* $y\mathsf{hsame}y'$*, then the same right-coset classifier holds at* (x', y')*.*

Proof. Translate the right-coset classifier through the quotient-kernel equivalence to obtain $Q_{Z_a \triangleleft N_a}(x, y)$. The quotient-kernel transport row moves both normalizer endpoints and the displayed kernel carrier across the two hsame endpoint comparisons. Translating back through the same coset/kernel equivalence repackages the transported kernel as the right-coset classifier at (x', y'). □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightCosetClassifier hsame transport`

Theorem 80.61 (Centralizer right coset classifier transitivity)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying associativity, right-unit, and multiplication-congruence laws, the centralizer right-coset classifier is transitive:*

$$Q^{\text{right}}_{Z_a \le N_a}(x, y) \wedge Q^{\text{right}}_{Z_a \le N_a}(y, z) \Longrightarrow Q^{\text{right}}_{Z_a \le N_a}(x, z).$$

Proof. Unpack the two right-coset witnesses. The first supplies a centralizer element u with $y\mathsf{hsamemul}(x, u)$, and the second supplies a centralizer element v with $z\mathsf{hsamemul}(y, v)$. Product closure for the centralizer carrier gives a carried witness $\mathsf{mul}(u, v)$. Multiplication congruence transports the second endpoint along the first comparison, and associativity identifies $\mathsf{mul}(\mathsf{mul}(x, u), v)$ with $\mathsf{mul}(x, \mathsf{mul}(u, v))$. The endpoint clauses are inherited from the two classifier hypotheses. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightCosetClassifier trans from empty unit`

Theorem 80.62 (Centralizer quotient kernel empty fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \triangleleft N_a}(\mathsf{e}, x) \Longleftrightarrow Z_a(x).$$

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel empty fiber iff`

Proof. For the forward direction, unpack the relative-kernel classifier $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$. Its kernel clause gives $Z_a(\mathsf{mul}(\mathsf{inv}(\mathsf{e}), x))$.

The inverse identity row identifies $\mathsf{inv}(\mathsf{e})$ with e, and the left-unit law identifies $\mathsf{mul}(\mathsf{e}, x)$ with x. The Z_a carrier-transport row from Definition 80.27 carries the kernel witness across this endpoint comparison and gives $Z_a(x)$.

For the reverse direction, assume $Z_a(x)$. The empty centralizer row gives $Z_a(\mathsf{e})$, and Definition 80.19 sends both $Z_a(\mathsf{e})$ and $Z_a(x)$ into the normalizer carrier N_a.

It remains to supply the kernel word. The same inverse identity and left-unit comparison identify $\mathsf{mul}(\mathsf{inv}(\mathsf{e}), x)$ with x; by transporting $Z_a(x)$ along the reverse comparison, we obtain the carried kernel clause and package $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$. □

Theorem 80.63 (Centralizer normalizer coset classifier identity fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \le N_a}(\mathsf{e}, x) \Longleftrightarrow Z_a(x).$$

Proof. Apply Definition 80.53 at the endpoint pair (e, x). It identifies the right-coset classifier $Q_{Z_a \le N_a}(\mathsf{e}, x)$ with the relative-kernel classifier $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$.

Then apply Definition 80.62. Its empty fiber calculation identifies $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$ with $Z_a(x)$. Transitivity of these two equivalences gives the displayed identity fiber for $Q_{Z_a \le N_a}$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier empty fiber iff`

Per-object template: traditional target identity coset fiber for the centralizer-normalizer quotient; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier $Q_{Z_a \le N_a}$; stability identity, inclusion, left-unit endpoint transport, and Z_a carrier transport; ledger identity representative and right witness; exactness predicate identity fiber equals lower carrier; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Theorem 80.64 (Centralizer right quotient diagonal carrier exactness)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies*

$$Q_{Z_a \leq N_a}(x, x) \iff N_a(x).$$

Proof. For the forward direction, unpack the right quotient classifier at the diagonal. Its endpoint clauses assert that both displayed endpoints lie in the upper carrier N_a. Since both endpoints are the same history x, projecting either endpoint clause gives $N_a(x)$.

For the reverse direction, assume $N_a(x)$. The reflexivity row of Definition 80.32 applies to this upper-carrier witness and gives the diagonal classifier instance $Q_{Z_a \leq N_a}(x, x)$. □

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerRightQuotientClassifier diagonal carrier iff`

Theorem 80.65 (Centralizer quotient identity-fiber bridge package)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *satisfies the following two identity-fiber packages.*

1. *The quotient-kernel identity fibers agree on the two sides of the unit:*

$$Q_{Z_a \lhd N_a}(\mathsf{e}, x) \iff Q_{Z_a \lhd N_a}(x, \mathsf{e}),$$

   and both displayed sides are equivalent to $Z_a(x)$*.*

2. *The right-witness quotient identity fibers agree on the two sides of the unit:*

$$Q_{Z_a \leq N_a}(\mathsf{e}, x) \iff Q_{Z_a \leq N_a}(x, \mathsf{e}),$$

   and both displayed sides are equivalent to $Z_a(x)$*.*

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotient identity fiber bridge package`

Proof. For the quotient-kernel package, apply Definition 80.62 to identify $Q_{Z_a \lhd N_a}(\mathsf{e}, x)$ with $Z_a(x)$. Apply Definition 80.42 to identify $Q_{Z_a \lhd N_a}(x, \mathsf{e})$ with the same centralizer carrier. Composing the two identity-fiber equivalences gives the two-sided unit bridge, and the two cited equivalences record the exactness clauses.

For the right-witness quotient package, apply Definition 80.63 to identify $Q_{Z_a \leq N_a}(\mathsf{e}, x)$ with $Z_a(x)$. Then apply Definition 80.52 at the endpoint pair (x, e) and use Definition 80.42 to identify $Q_{Z_a \leq N_a}(x, \mathsf{e})$ with $Z_a(x)$. Composing these equivalences gives the right-witness two-sided unit bridge, with the same centralizer exactness on both sides. □

Per-object template: traditional target two-sided identity coset fiber; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier the quotient-kernel and right-witness quotient relations; stability unit, inverse, transport, and kernel equivalence rows; ledger identity representative side; core theorem Definition 80.65; exactness identity fiber equals Z_a; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status $\mathsf{ProofTarget}$.

Theorem 80.66 (Centralizer quotient diagonal presentation package)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same associativity, unit, multiplication-congruence, carrier-transport, and two-sided inverse laws as in Definition 80.27, every* $x : \mathsf{BHist}$ *with* $N_a(x)$ *satisfies the four diagonal quotient classifiers*

$$Q^{\mathrm{std}}_{Z_a \leq N_a}(x, x), \qquad Q^{\mathrm{wit}}_{Z_a \leq N_a}(x, x), \qquad Q^{\mathrm{cos}}_{Z_a \leq N_a}(x, x), \qquad Q^{\mathrm{ker}}_{Z_a \lhd N_a}(x, x).$$

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupUp StdBridge`

Proof. Apply Definition 80.21 to the normalizer endpoint $N_a(x)$. This gives the diagonal relative-kernel classifier $Q^{\mathrm{ker}}_{Z_a \triangleleft N_a}(x, x)$.

Next transport the same diagonal relation through Definition 80.50. Reading the equivalence from the kernel side yields $Q^{\mathrm{std}}_{Z_a \leq N_a}(x, x)$.

Apply Definition 80.51 at the same endpoint pair. The kernel-to-witness direction gives $Q^{\mathrm{wit}}_{Z_a \leq N_a}(x, x)$.

Finally apply Definition 80.34 to obtain $Q^{\mathrm{cos}}_{Z_a \leq N_a}(x, x)$. The parallel Definition 80.52 records that the public right quotient presentation is classified by the same diagonal kernel word, so the displayed standard, witness, coset, and kernel presentations share one reflexive source. □

Per-object template: traditional target diagonal quotient presentation package; BEDC source $\mathsf{Group}^{\uparrow}$ plus $\mathsf{Subgroup}^{\uparrow}$ over BHist; carrier N_a; classifier the standard, right-witness, right-coset, and relative-kernel quotient relations; stability carrier transport, multiplication congruence, unit, inverse, and quotient-kernel equivalence rows; ledger diagonal endpoint $x = x$ and empty relative kernel word; core theorem Definition 80.66; exactness every normalizer endpoint carries all four diagonal classifiers; downstream consumer $\mathsf{QuotientGroup}^{\uparrow}$; status ProofTarget.

Closure status for $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$.

Constructive story: A subgroup in this chapter is generated inside an already licensed $\mathsf{Group}^{\uparrow}$ history surface by fixing a history a and accepting exactly those carried group endpoints that commute with it; the identity endpoint enters because it commutes with a, products enter by the group multiplication rows, inverses enter by the inverse rows, and histories without the displayed centralizer witness stay outside. Two accepted subgroup elements are classified by restricting ambient history-sameness to this centralizer carrier, so equality is neither a new subgroup equality nor a quotient of the ambient group but the same hsame comparison with both membership witnesses retained. The source is the ambient group plus the centralizer predicate, the pattern records identity, product, inverse, transport, commutator-fiber exactness, intersections, chain composition, the trivial subgroup, and the one-step subgroup criterion, and the ledger keeps the fixed element a, the commuting witnesses, and the inherited group-operation rows visible. The five name-certificate obligations close because the unit gives an inhabited carrier, ambient hsame gives reflexivity, symmetry, and transitivity on carried endpoints, and the centralizer transport row moves membership across any recognized endpoint comparison. A history h lies in the signature gap of a $\mathsf{Subgroup}^{\uparrow}$ token p when the source-domain accepts h as a carried centralizer or related subgroup endpoint, the signature relation pairs it with an answer endpoint and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h must satisfy the relevant membership row, with the answer classified by the restricted ambient comparison. The checked bridge package then exposes centralizer quotient fibers, diagonal quotient presentations, and the identity-fiber readback used by downstream quotient consumers, always through the same membership and classifier rows. The chapter registers this inherited centralizer-and-subobject construction under the name $\mathsf{Subgroup}^{\uparrow}$; it does not introduce arbitrary subgroup lattices, Sylow theory, general isomorphism theorems, or a separate subgroup carrier outside the displayed group history surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public $\mathsf{Subgroup}^{\uparrow}$ surface binds subgroup membership, inherited classifiers, the centralizer quotient bridge, the diagonal presentation package, the trivial subgroup certificate, the subgroup chain composition certificate, and the one-step subgroup criterion.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SubgroupUp.SubgroupCentralizer\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Arbitrary subgroup lattices, Sylow theory, and general isomorphism-theorem coverage are outside this chapter.

Upgrade path: no further upgrade: chapter at top of formal axis

Definition 80.67 (Trivial SubgroupUp carrier)**.** *Fix a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *source over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and ambient classifier* hsame*. Define*

$$\mathsf{Triv}_G(x) :\Longleftrightarrow x \mathsf{hsame} e.$$

The inherited classifier on this carrier is the carrier-aware restriction of hsame*:*

$$\mathsf{TrivRel}_G(x, y) :\Longleftrightarrow \mathsf{Triv}_G(x) \wedge \mathsf{Triv}_G(y) \wedge x\mathsf{hsame}y.$$

The operation, unit, and inverse are inherited unchanged from the ambient $\mathsf{Group}^{\uparrow}$ *source.*

Theorem 80.68 (Trivial subgroup SubgroupUp certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the monoid stability rows of Definition 40.13 and the inverse rows of Definition 41.55, the carrier and classifier of Definition 80.67 satisfy the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate rows:*

1. $\mathsf{TrivRel}_G$ *is a semantic naming classifier on* Triv_G*.*
2. $\mathsf{Triv}_G(e)$*.*
3. *If* $\mathsf{Triv}_G(x)$ *and* $\mathsf{Triv}_G(y)$*, then* $\mathsf{Triv}_G(\mathsf{mul}(x, y))$*.*
4. *If* $\mathsf{Triv}_G(x)$*, then* $\mathsf{Triv}_G(\mathsf{inv}(x))$*.*
5. *If* $\mathsf{Triv}_G(x)$ *and* $x\mathsf{hsame}y$*, then* $\mathsf{Triv}_G(y)$*.*
6. *If* $\mathsf{TrivRel}_G(x, x')$ *and* $\mathsf{TrivRel}_G(y, y')$*, then* $\mathsf{TrivRel}_G(\mathsf{mul}(x, y), \mathsf{mul}(x', y'))$*.*
7. *If* $\mathsf{TrivRel}_G(x, y)$*, then* $\mathsf{TrivRel}_G(\mathsf{inv}(x), \mathsf{inv}(y))$*.*

Thus the singleton fiber of the group unit carries a $\mathsf{Subgroup}^{\uparrow}$ *certificate over the inherited classifier.*

Lean4 ✓ `BEDC.Derived.SubgroupUp.SubgroupTrivialCarrier certificate`

Proof. The semantic classifier row is inherited from hsame. Reflexivity on a carried endpoint x packages the carrier witness $\mathsf{Triv}_G(x)$ twice with $x\mathsf{hsame}x$. Symmetry and transitivity unpack the stored hsame comparisons, apply the corresponding ambient law, and repack the endpoint carrier fields already present in the classifier witnesses.

The identity row is the reflexive witness $e\mathsf{hsame}e$, which is exactly $\mathsf{Triv}_G(e)$ by Definition 80.67.

For product closure, assume $\mathsf{Triv}_G(x)$ and $\mathsf{Triv}_G(y)$. These are the comparisons $x\mathsf{hsame}e$ and $y\mathsf{hsame}e$. Multiplication congruence from the monoid stability rows gives

$$\mathsf{mul}(x, y)\mathsf{hsamemul}(e, e).$$

The left identity law at e gives $\mathsf{mul}(e, e)\mathsf{hsame}e$. Transitivity proves $\mathsf{mul}(x, y)\mathsf{hsame}e$, hence $\mathsf{Triv}_G(\mathsf{mul}(x, y))$.

For inverse closure, assume $\mathsf{Triv}_G(x)$, so $x\mathsf{hsame}e$. The derived inverse congruence theorem Definition 41.60 gives

$$\mathsf{inv}(x)\mathsf{hsameinv}(e).$$

The identity-inverse theorem Definition 41.56 gives $\mathsf{inv}(e)\mathsf{hsame}e$. Transitivity proves $\mathsf{Triv}_G(\mathsf{inv}(x))$.

For carrier transport, assume $\mathsf{Triv}_G(x)$ and $x\mathsf{hsame}y$. Symmetry gives $y\mathsf{hsame}x$, and transitivity with $x\mathsf{hsame}e$ gives $y\mathsf{hsame}e$. This is $\mathsf{Triv}_G(y)$.

For multiplication compatibility of the inherited classifier, unpack the two $\mathsf{TrivRel}_G$ witnesses. Product closure supplies the two endpoint carrier rows for $\mathsf{mul}(x, y)$ and $\mathsf{mul}(x', y')$, while multiplication congruence transports the stored comparisons $x\mathsf{hsame}x'$ and $y\mathsf{hsame}y'$ to the product comparison. These three fields repack $\mathsf{TrivRel}_G(\mathsf{mul}(x, y), \mathsf{mul}(x', y'))$.

For inverse compatibility, unpack $\mathsf{TrivRel}_G(x, y)$. Inverse closure supplies the endpoint carrier rows for $\mathsf{inv}(x)$ and $\mathsf{inv}(y)$, and Definition 41.60 transports the stored comparison $x\mathsf{hsame}y$ to $\mathsf{inv}(x)\mathsf{hsameinv}(y)$. Repacking gives the displayed inverse-classifier row.

These are the identity, product, inverse, transport, and restricted-classifier fields required by the $\mathsf{Subgroup}^{\uparrow}$ row format used in Definition 80.2. □

Theorem 80.69 (Subgroup chain composition certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and ambient classifier* hsame*, assume the monoid stability rows of Definition 40.13, the inverse rows of Definition 41.55, and the history-sameness equivalence of Definition 3.8. Let* $P_K, P_H : \mathsf{BHist} \to \mathsf{Prop}$ *be carrier predicates and define*

$$P_{H\leq G}(x) :\Longleftrightarrow P_K(x) \wedge P_H(x),$$

$$R_{H\leq G}(x,y) :\Longleftrightarrow P_{H\leq G}(x) \wedge P_{H\leq G}(y) \wedge x\mathsf{hsame}y,$$

and

$$R_H^K(x,y) :\Longleftrightarrow P_K(x) \wedge P_H(x) \wedge P_K(y) \wedge P_H(y) \wedge x\mathsf{hsame}y.$$

Assume the P_K *carrier has the ambient* $\mathsf{Subgroup}^{\uparrow}$ *carrier rows:*

1. $P_K(e)$.
2. *If* $P_K(x)$ *and* $P_K(y)$*, then* $P_K(\mathsf{mul}(x,y))$.
3. *If* $P_K(x)$*, then* $P_K(\mathsf{inv}(x))$.
4. *If* $P_K(x)$ *and* $x\mathsf{hsame}y$*, then* $P_K(y)$.

Assume P_H *has the following rows over the* P_K*-restricted carrier:*

1. $P_H(e)$.
2. *If* $P_K(x)$*,* $P_H(x)$*,* $P_K(y)$*, and* $P_H(y)$*, then* $P_H(\mathsf{mul}(x,y))$.
3. *If* $P_K(x)$ *and* $P_H(x)$*, then* $P_H(\mathsf{inv}(x))$.
4. *If* $P_K(x)$*,* $P_K(y)$*,* $P_H(x)$*, and* $x\mathsf{hsame}y$*, then* $P_H(y)$.
5. *If* $R_H^K(x,x')$ *and* $R_H^K(y,y')$*, then* $R_H^K(\mathsf{mul}(x,y),\mathsf{mul}(x',y'))$.
6. *If* $R_H^K(x,y)$*, then* $R_H^K(\mathsf{inv}(x),\mathsf{inv}(y))$.

Then the carrier $P_{H\leq G}$ *with restricted classifier* $R_{H\leq G}$ *satisfies the concrete* $\mathsf{Subgroup}^{\uparrow}$ *certificate rows over* G *in the row format of Definition 80.68 and Definition 80.2.*

Proof. The relation R_H^K is the same restricted relation as $R_{H\leq G}$ after reassociating the endpoint carrier conjunctions. Thus either presentation stores exactly two endpoint witnesses for $P_K \wedge P_H$ and one ambient hsame comparison.

First verify the semantic naming classifier required by Definition 9.7. If $P_{H\leq G}(x)$, ambient reflexivity gives $x\mathsf{hsame}x$, so $R_{H\leq G}(x,x)$ holds. If $R_{H\leq G}(x,y)$ holds, swap the two endpoint carrier witnesses and apply symmetry of hsame to obtain $R_{H\leq G}(y,x)$. If $R_{H\leq G}(x,y)$ and $R_{H\leq G}(y,z)$ hold, keep the stored carrier witnesses for x and z and compose $x\mathsf{hsame}y$ with $y\mathsf{hsame}z$ by transitivity of hsame. Hence $R_{H\leq G}$ is a semantic naming classifier on the conjunction carrier.

For the identity row, pair the ambient $P_K(e)$ row with the internal $P_H(e)$ row. This is exactly $P_{H\leq G}(e)$.

For product closure, assume $P_{H\leq G}(x)$ and $P_{H\leq G}(y)$. Unfolding these hypotheses gives $P_K(x)$, $P_H(x)$, $P_K(y)$, and $P_H(y)$. The P_K product row gives $P_K(\mathsf{mul}(x,y))$, while the internal P_H product row gives $P_H(\mathsf{mul}(x,y))$. Pairing the two conclusions gives $P_{H\leq G}(\mathsf{mul}(x,y))$.

For inverse closure, unfold $P_{H\leq G}(x)$ into $P_K(x)$ and $P_H(x)$. The P_K inverse row gives $P_K(\mathsf{inv}(x))$, and the internal P_H inverse row gives $P_H(\mathsf{inv}(x))$. Together these prove $P_{H\leq G}(\mathsf{inv}(x))$.

For carrier transport, assume $P_{H\leq G}(x)$ and $x\mathsf{hsame}y$. The P_K transport row first gives $P_K(y)$. With $P_K(x)$, this target $P_K(y)$, $P_H(x)$, and the same comparison $x\mathsf{hsame}y$, the internal P_H transport row gives $P_H(y)$. Therefore $P_{H\leq G}(y)$.

For multiplication compatibility of the restricted classifier, unpack the two $R_{H\leq G}$ witnesses as $R_H^K(x,x')$ and $R_H^K(y,y')$. The internal restricted-classifier product row gives $R_H^K(\mathsf{mul}(x,y),\mathsf{mul}(x',y'))$, which is precisely $R_{H\leq G}(\mathsf{mul}(x,y),\mathsf{mul}(x',y'))$ after unfolding the conjunction carrier.

For inverse compatibility, unpack $R_{H\leq G}(x,y)$ as $R_H^K(x,y)$. The internal restricted-classifier inverse row gives $R_H^K(\mathsf{inv}(x),\mathsf{inv}(y))$, equivalently $R_{H\leq G}(\mathsf{inv}(x),\mathsf{inv}(y))$.

These are exactly the semantic classifier, identity, product, inverse, carrier-transport, multiplication-classifier, and inverse-classifier rows of a $\mathsf{Subgroup}^{\uparrow}$ certificate over the original ambient group G. □

Theorem 80.70 (Subgroup one-step criterion)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *with operation* mul*, unit* e*, inverse operation* inv*, and ambient classifier* hsame*, assume the monoid stability rows of Definition 40.13 and the inverse rows of Definition 41.55. Let* $P : \mathsf{BHist} \to \mathsf{Prop}$ *be a carrier predicate, and let the inherited restricted classifier be*

$$R_P(x,y) :\Longleftrightarrow P(x) \wedge P(y) \wedge x\mathsf{hsame}y.$$

Assume:

1. *P is inhabited: $P(a)$ for some $a : \mathsf{BHist}$.*
2. *P is stable under the ambient classifier: if $P(x)$ and $x\mathsf{hsame}y$, then $P(y)$.*
3. *P is closed under the one-step relative product: if $P(x)$ and $P(y)$, then $P(\mathsf{mul}(x,\mathsf{inv}(y)))$.*

Then the carrier P with classifier R_P satisfies the concrete $\mathsf{Subgroup}^{\uparrow}$ *identity, product, inverse, and carrier-transport rows:*

1. *$P(e)$.*
2. *If $P(x)$ and $P(y)$, then $P(\mathsf{mul}(x,y))$.*
3. *If $P(x)$, then $P(\mathsf{inv}(x))$.*
4. *If $P(x)$ and $x\mathsf{hsame}y$, then $P(y)$.*

Proof. Choose an inhabitant a with $P(a)$. The relative-product row applied to a and a gives $P(\mathsf{mul}(a,\mathsf{inv}(a)))$. The right inverse row from Definition 41.55 identifies this endpoint with e:

$$\mathsf{mul}(a,\mathsf{inv}(a))\mathsf{hsame}e.$$

Classifier stability for P transports the witness to $P(e)$, proving the identity row.

For inverse closure, assume $P(x)$. The identity row gives $P(e)$, so the relative-product row applied to e and x gives $P(\mathsf{mul}(e,\mathsf{inv}(x)))$. The left identity row from Definition 40.13 gives

$$\mathsf{mul}(e,\mathsf{inv}(x))\mathsf{hsame}\mathsf{inv}(x),$$

and carrier stability transports the witness to $P(\mathsf{inv}(x))$.

For product closure, assume $P(x)$ and $P(y)$. The inverse row just obtained gives $P(\mathsf{inv}(y))$. Apply the relative-product row to x and $\mathsf{inv}(y)$ to get

$$P(\mathsf{mul}(x,\mathsf{inv}(\mathsf{inv}(y)))).$$

By Definition 41.57, $\mathsf{inv}(\mathsf{inv}(y))\mathsf{hsame}y$. Multiplication congruence from Definition 40.13, together with reflexivity on x, gives

$$\mathsf{mul}(x,\mathsf{inv}(\mathsf{inv}(y)))\mathsf{hsame}\mathsf{mul}(x,y).$$

Carrier stability transports the displayed witness to $P(\mathsf{mul}(x,y))$.

The transport row is exactly the classifier-stability hypothesis on P. These four conclusions are the carrier rows in the $\mathsf{Subgroup}^{\uparrow}$ row format used in Definition 80.2 and Definition 80.68. □

Chapter 81

A Concrete Naming Certificate for $\mathsf{Ideal}^{\uparrow}$

The ideal interface $\mathsf{Ideal}^{\uparrow}$ packages a sub-rng of a ring carrier closed under multiplication by every ambient ring element. It depends on $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ (chapter 43) and supplies the kernel object for the quotient-ring construction $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$ (chapter 83).

81.1 Carrier definition

Definition 81.1 (Ideal carried predicate rows)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BEDC histories, with $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ carrier predicate $C : \mathsf{BHist} \to \mathsf{Prop}$, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. An $\mathsf{Ideal}^{\uparrow}$ carried predicate over R is a predicate*

$$I : \mathsf{BHist} \to \mathsf{Prop}$$

together with the following certificate rows:

1. *carrier support: $I(x) \Longrightarrow C(x)$;*
2. *zero membership: $I(0_R)$;*
3. *additive closure: $I(x) \wedge I(y) \Longrightarrow I(x +_R y)$;*
4. *additive inverse closure: $I(x) \Longrightarrow I(-_R x)$;*
5. *internal multiplication closure: $I(x) \wedge I(y) \Longrightarrow I(x \cdot_R y)$;*
6. *classifier transport: $I(x) \wedge x \sim_R y \Longrightarrow I(y)$;*
7. *left and right ambient absorption: for every carried ambient endpoint r with $C(r)$,*

$$I(x) \Longrightarrow I(r \cdot_R x) \wedge I(x \cdot_R r).$$

81.2 The certificate

Definition 81.2 (Ideal obligation inventory)**.** *An $\mathsf{Ideal}^{\uparrow}$ certificate over a fixed $\mathsf{Ring}^{\uparrow}$ certificate is the ten-field $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ inventory whose rows are fixed as follows.*

1. *The traditional target is a two-sided ideal of a ring.*

2. *The BEDC source is* $\mathsf{Ring}^{\uparrow}$ *over* BHist*, with the ambient carrier, classifier, addition, additive inverse, zero, and multiplication already named by the ring certificate.*
3. *The carrier is a predicate* $I : \mathsf{BHist} \to \mathsf{Prop}$ *supported by the ambient ring carrier* C*.*
4. *The classifier is inherited from the ambient ring classifier* $\sim_R$*.*
5. *The stability rows are zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and two-sided absorption by every carried ambient ring endpoint.*
6. *The ledger records hidden membership witnesses for* $I(x)$ *and the ambient absorber witnesses* $C(r)$ *consumed by the two absorption rows.*
7. *The core theorem list consists of Definition 81.3, Definition 81.4, Definition 81.5, Definition 81.19, Definition 81.44, and Definition 81.47.*
8. *Exactness is row exhaustion: the certificate consumes exactly the carrier support, zero, additive, inverse, internal multiplication, classifier transport, and two-sided absorption rows displayed here.*
9. *The standard bridge row is absent at this paper site.*
10. *The boundary package is the scoped* $\mathsf{Ideal}^{\uparrow}$ *obligation surface used by quotient-ring, ideal-class, intersection, finite-meet, and preimage constructions.*

Lean4 (definition) `BEDC.Derived.IdealUp.IdealObligationInventory`

Theorem 81.3 (Ideal carrier support obligation)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}$ *certificate over* BHist *with ambient carrier* C *and classifier* $\sim_R$*. Let* $I : \mathsf{BHist} \to \mathsf{Prop}$ *satisfy the inventory of Definition 81.2. Then the inventory supplies the carrier-support row*

$$I(x) \Longrightarrow C(x).$$

Consequently every endpoint admitted by the ideal predicate is a carried $\mathsf{Ring}^{\uparrow}$ *endpoint.*

Proof. Project the carrier row of Definition 81.2. That row states that I is supported by the ambient $\mathsf{Ring}^{\uparrow}$ carrier C.

Applying this projection to a membership witness $I(x)$ gives $C(x)$. The endpoint x therefore lies in the same BHist carrier scope as the ambient ring operations and classifier $\sim_R$. □

Theorem 81.4 (Ideal zero and additive obligations)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}$ *certificate over* BHist *with addition* $+_R$*, additive inverse* $-_R$*, zero* 0_R*, multiplication* $\cdot_R$*, ambient carrier* C*, and classifier* $\sim_R$*. Let* $I : \mathsf{BHist} \to \mathsf{Prop}$ *satisfy the inventory of Definition 81.2. Then the inventory supplies*

$$I(0_R),$$

$$I(x) \wedge I(y) \Longrightarrow I(x +_R y),$$

$$I(x) \Longrightarrow I(-_R x),$$

and

$$I(x) \wedge I(y) \Longrightarrow I(x \cdot_R y).$$

Lean4 ✓ `BEDC.Derived.IdealUp.IdealCarriedPredicateRows zero add neg mul rows`

Proof. Project the stability field of Definition 81.2. Its zero row is exactly $I(0_R)$, using the zero endpoint named by the surrounding $\mathsf{Ring}^{\uparrow}$ certificate.

The same stability field contains the additive closure and additive inverse rows. Projecting those rows gives closure under $+_R$ and $-_R$ for every pair or endpoint already carrying an I-membership witness.

Finally project the internal multiplication row from the same inventory. Since the multiplication symbol $\cdot_R$ is the one named by the ambient $\mathsf{Ring}^{\uparrow}$ certificate, the resulting endpoint remains in the ring carrier scope supplied by the carrier-support row. □

Theorem 81.5 (Ideal classifier and absorption obligations)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist with carrier C, classifier $\sim_R$, and multiplication $\cdot_R$. Let $I : \mathsf{BHist} \to \mathsf{Prop}$ satisfy the inventory of Definition 81.2. Then the inventory supplies classifier transport*

$$I(x) \wedge x \sim_R y \Longrightarrow I(y)$$

and two-sided absorption by every carried ambient ring endpoint:

$$C(r) \wedge I(x) \Longrightarrow I(r \cdot_R x) \wedge I(x \cdot_R r).$$

Proof. Project the classifier row and the classifier-transport stability row of Definition 81.2. The classifier row identifies the ideal comparison relation with the ambient $\mathsf{Ring}^{\uparrow}$ classifier $\sim_R$, and the transport row carries I-membership along that comparison.

Project the absorption rows from the same stability field. Given an ambient absorber witness $C(r)$ and an ideal membership witness $I(x)$, the left row gives $I(r \cdot_R x)$ and the right row gives $I(x \cdot_R r)$.

The ledger row records exactly the membership witness for x and the carried ambient witness for r consumed in these projections, so no extra absorber or classifier datum is introduced outside the surrounding $\mathsf{Ring}^{\uparrow}$ scope. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealObligationInventory classifier absorption rows`

Definition 81.6 (Ambient carrier ideal predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist with ambient carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. The ambient carrier ideal predicate is*

$$W_R(x) \Longleftrightarrow C(x).$$

Its carried-predicate certificate rows are the following ten-field package: traditional target the whole ring ideal, BEDC source the surrounding $\mathsf{Ring}^{\uparrow}$ certificate over BHist, carrier C, classifier $\sim_R$, stability from $\mathsf{Ring}^{\uparrow}$ carrier transport and operation closure, ledger empty except for the ambient carrier witnesses consumed by the rows, exactness by exhaustion of the displayed row inventory, standard bridge absent, and status proof target.

Theorem 81.7 (Ambient carrier IdealUp closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist with notation as in Definition 81.6. Then W_R is an $\mathsf{Ideal}^{\uparrow}$ predicate over R: it has carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and two-sided absorption by every carried ambient ring endpoint.*

Proof. Unfold Definition 81.6. Carrier support is identity: a witness $W_R(x)$ is exactly a witness $C(x)$.

Zero membership is the zero-carrier row of the surrounding $\mathsf{Ring}^{\uparrow}$ certificate. Additive closure, additive inverse closure, and internal multiplication closure are exactly the carrier-closure rows for $+_R$, $-_R$, and $\cdot_R$ after replacing W_R by C.

Classifier transport is the carrier-transport row of the same $\mathsf{Ring}^{\uparrow}$ certificate along $\sim_R$. If $W_R(x)$ and $x \sim_R y$, transport gives $C(y)$, hence $W_R(y)$.

For absorption, assume $W_R(x)$ and $C(r)$. The multiplication-closure row of the ring certificate gives both $C(r \cdot_R x)$ and $C(x \cdot_R r)$. By the definition of W_R, these are $W_R(r \cdot_R x)$ and $W_R(x \cdot_R r)$.

These rows are precisely the $\mathsf{Ideal}^{\uparrow}$ carried-predicate inventory over the fixed $\mathsf{Ring}^{\uparrow}$ certificate. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealAmbientCarrier closure rows`

Theorem 81.8 (Ambient ideal quotient-kernel carrier exactness)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist with notation as in Definition 81.6. Let Q_{W_R} be the quotient-kernel source relation of Definition 81.37 formed from the ambient carrier ideal predicate W_R. Then*

$$Q_{W_R}(x, y) \iff C(x) \wedge C(y).$$

Lean4 ✓ `BEDC.Derived.IdealUp.IdealAmbientCarrier quotient kernel exactness`

Proof. Unfold Definition 81.37 and Definition 81.6.

For the forward direction, a witness $Q_{W_R}(x, y)$ consists of endpoint carrier witnesses $C(x)$ and $C(y)$ together with the membership witness $W_R(x -_R y)$. Projecting the first two components gives $C(x) \wedge C(y)$.

For the reverse direction, assume $C(x)$ and $C(y)$. The additive inverse-closure row of the $\mathsf{Ring}^{\uparrow}$ certificate gives $C(-_R y)$, and the additive closure row gives

$$C(x -_R y).$$

By the definition of W_R, this is $W_R(x -_R y)$. Pairing it with the original endpoint carrier witnesses gives $Q_{W_R}(x, y)$. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealAmbientCarrier quotient kernel carrier exactness`
Lean4 ✓ `BEDC.Derived.IdealUp.IdealAmbientQuotientKernel carrier exactness`

Definition 81.9 (Zero IdealUp predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. The zero $\mathsf{Ideal}^{\uparrow}$ predicate over R is the concrete carried predicate*

$$\mathsf{ZeroIdeal}_R(x) \iff C(x) \wedge x \sim_R 0_R.$$

Lean4 ✓ `BEDC.Derived.IdealUp.IdealZeroPredicate semantic name certificate`

Theorem 81.10 (Zero IdealUp predicate closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with notation as in Definition 81.9. Assume the ring certificate supplies carrier transport, zero carriedness, addition congruence, additive inverse congruence, multiplication congruence, additive zero and inverse laws, and the two zero-product rows $0_R \cdot_R r \sim_R 0_R$ and $r \cdot_R 0_R \sim_R 0_R$ for every carried endpoint r. Then $\mathsf{ZeroIdeal}_R$ is an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R. Explicitly, it has carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried ambient endpoint.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealZeroPredicate closure rows`

Proof. Carrier support is the first conjunct in Definition 81.9. Zero membership uses the ring zero carriedness row together with classifier reflexivity at 0_R.

For additive closure, unpack two memberships as carried endpoints classified with 0_R. Ring addition congruence transports the two classifier comparisons to

$$x +_R y \sim_R 0_R +_R 0_R.$$

The additive zero law identifies $0_R +_R 0_R$ with 0_R, and transitivity gives the displayed zero classifier for $x +_R y$. The ring operation-closure row supplies $C(x +_R y)$, so the two witnesses repack $\mathsf{ZeroIdeal}_R(x +_R y)$.

For additive inverse closure, additive inverse congruence transports $x \sim_R 0_R$ to $-_R x \sim_R -_R 0_R$. The inverse row for zero and transitivity identify $-_R x$ with 0_R, while the inverse operation-closure row supplies $C(-_R x)$.

Internal multiplication is the same pattern. Multiplication congruence sends the two zero classifiers to $x \cdot_R y \sim_R 0_R \cdot_R 0_R$, and the zero-product row collapses the right endpoint to 0_R. Multiplication closure supplies the carrier conjunct.

Classifier transport is immediate from the carrier-transport row and classifier transitivity: if $\mathsf{ZeroIdeal}_R(x)$ and $x \sim_R y$, then $C(y)$ follows from carrier transport, and $y \sim_R 0_R$ follows by symmetry and transitivity from $x \sim_R y$ and $x \sim_R 0_R$.

Finally fix a carried ambient endpoint r. From $\mathsf{ZeroIdeal}_R(x)$, multiplication congruence identifies $r \cdot_R x$ with $r \cdot_R 0_R$ and $x \cdot_R r$ with $0_R \cdot_R r$; the two zero-product rows identify both products with 0_R. Multiplication closure supplies their carrier conjuncts, so both absorption rows repack as zero-ideal memberships. □

↪variant BEDC.Derived.IdealUp.IdealZeroPredicate closure rows

Lean4 ✓ BEDC.Derived.IdealUp.IdealZeroPredicate ideal closure

Definition 81.11 (Whole IdealUp predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. The whole $\mathsf{Ideal}^{\uparrow}$ predicate over R is the concrete carried predicate*

$$\mathsf{WholeIdeal}_R(x) \iff C(x).$$

Theorem 81.12 (Whole IdealUp predicate closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with notation as in Definition 81.11. Assume the ring certificate supplies zero carriedness, carrier transport, and carrier closure for addition, additive inverse, and multiplication. Then $\mathsf{WholeIdeal}_R$ is an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R. Explicitly, it has carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried ambient endpoint.*

Lean4 ✓ BEDC.Derived.IdealUp.IdealWholePredicate closure rows

Proof. Unfold Definition 81.11. Carrier support is identity on the ambient carrier predicate, and zero membership is exactly the ring zero carriedness row.

The additive, inverse, and internal multiplication rows are the corresponding ambient operation-closure rows of the $\mathsf{Ring}^{\uparrow}$ certificate. Since $\mathsf{WholeIdeal}_R(x)$ means only $C(x)$, each operation closure repacks directly as whole-ideal membership.

Classifier transport is the ambient carrier-transport row. If $C(x)$ and $x \sim_R y$, the transported carrier witness is precisely $\mathsf{WholeIdeal}_R(y)$.

For two-sided absorption, fix $C(r)$ and $\mathsf{WholeIdeal}_R(x)$. Ambient multiplication closure gives $C(r \cdot_R x)$ and $C(x \cdot_R r)$, which are exactly the left and right whole-ideal memberships. □

Theorem 81.13 (Zero and whole quotient-kernel scope)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with notation as in Definition 81.37. Let $Q_{\mathsf{ZeroIdeal}}$ and $Q_{\mathsf{WholeIdeal}}$ be the source packages of Definition 81.37 built from Definition 81.9 and Definition 81.11. Then the zero-ideal package is the equality-like carried-endpoint kernel*

$$Q_{\mathsf{ZeroIdeal}}(x, y) \iff C(x) \wedge C(y) \wedge (x -_R y) \sim_R 0_R,$$

with the RingUp subtraction-zero rows identifying this condition with $x \sim_R y$ whenever the ambient difference exactness row is available. The whole-ideal package is the total carried-endpoint kernel

$$Q_{\mathsf{WholeIdeal}}(x, y) \iff C(x) \wedge C(y).$$

Both packages remain quotient-free BHist *relations consumable by* $\mathsf{QuotientRing}^{\uparrow}$ *before any quotient carrier or quotient primitive is formed.*

Proof. For the zero predicate, unfold Definition 81.37 and Definition 81.9. The relation $Q_{\mathsf{ZeroIdeal}}(x, y)$ contains $C(x)$, $C(y)$, and the two fields of $\mathsf{ZeroIdeal}_R(x -_R y)$.

The carrier field $C(x -_R y)$ is already supplied by the subtraction operation-closure rows from the two carried endpoints. The remaining field is the displayed zero classifier $(x -_R y) \sim_R 0_R$. Thus the quotient-kernel source repacks as the equality-like carried-endpoint kernel, and the ambient subtraction-zero exactness rows compare that kernel with the classifier $x \sim_R y$.

For the whole predicate, unfold Definition 81.11. The membership $\mathsf{WholeIdeal}_R(x -_R y)$ is exactly the carrier witness $C(x -_R y)$, which follows from the carrier closure of additive inverse and addition applied to $C(x)$ and $C(y)$. Therefore no condition remains beyond the two carried endpoints, giving the displayed total kernel.

The diagonal and endpoint-transport surfaces of these two concrete kernels are the specializations of Definition 81.38 and Definition 81.39. The repacked source packages therefore expose the same quotient-free boundary described in Definition 81.37, so $\mathsf{QuotientRing}^{\uparrow}$ consumes them through the existing IdealUp surface. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealZeroAndWholeQuotientKernel scope`

Theorem 81.14 (Zero IdealUp kernel is contained in whole IdealUp kernel)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with notation as in Definition 81.37. Let $Q_{\mathsf{ZeroIdeal}}$ and $Q_{\mathsf{WholeIdeal}}$ be the two source packages of Definition 81.13. For every endpoint pair x, y, if the zero-ideal quotient-kernel relation holds, then the whole-ideal quotient-kernel relation holds:*

$$Q_{\mathsf{ZeroIdeal}}(x, y) \Longrightarrow Q_{\mathsf{WholeIdeal}}(x, y).$$

Equivalently, every carried endpoint pair satisfying the zero-ideal kernel is contained in the carried endpoint surface of the whole-ideal kernel.

The ten-field fit is: traditional target extremal ideal-kernel comparison; BEDC source $\mathsf{Ring}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, *and* $\mathsf{QuotientRing}^{\uparrow}$*; carrier ambient carried endpoints; classifier quotient-kernel relation; stability inherited from the two concrete ideal predicates; ledger hidden subtraction witness; core theorem this containment; exactness the displayed implication; standard bridge zero quotient refines whole quotient; status paper target.*

Proof. Unfold the scope computation of Definition 81.13. A zero-kernel witness for $Q_{\mathsf{ZeroIdeal}}(x, y)$ consists of the carried endpoint witnesses $C(x)$ and $C(y)$ together with the hidden subtraction classifier $(x -_R y) \sim_R 0_R$.

Project the two endpoint fields and discard the subtraction witness for this containment. The whole-kernel computation in the same theorem has no remaining ideal-membership condition beyond the ambient endpoint surface.

Repack the projected witnesses $C(x)$ and $C(y)$ as the witness for $Q_{\mathsf{WholeIdeal}}(x, y)$. This gives the displayed implication and records that the zero quotient-kernel relation is a subrelation of the whole quotient-kernel relation. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealZeroKernel contained in whole kernel`

Theorem 81.15 (Ideal zero-membership obligation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist, with $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. If $I : \mathsf{BHist} \to \mathsf{Prop}$ is an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R in the sense of Definition 81.1, then the certificate contains the zero-membership row $I(0_R)$.*

Proof. Unpack the $\mathsf{Ideal}^{\uparrow}$ carried-predicate data from Definition 81.1. The underlying predicate is explicitly a predicate on BHist, while the zero endpoint 0_R is the zero row of the ambient $\mathsf{Ring}^{\uparrow}$ certificate.

Among the rows carried by the ideal certificate is the zero-membership row. Projecting that row gives exactly $I(0_R)$. □

Theorem 81.16 (Ideal classifier-transport obligation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist, with $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. If $I : \mathsf{BHist} \to \mathsf{Prop}$ is an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R in the sense of Definition 81.1, then its certificate contains the classifier-transport row*

$$I(x) \wedge x \sim_R y \Longrightarrow I(y).$$

Proof. Unpack the carried predicate rows. The endpoints x and y are BEDC histories, the relation $x \sim_R y$ is the classifier supplied by the ambient $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$, and I is the BHist predicate being certified.

The classifier-transport row listed in Definition 81.1 is one of the fields of the $\mathsf{Ideal}^{\uparrow}$ certificate. Projecting it and applying it to a membership witness $I(x)$ and the classifier comparison $x \sim_R y$ yields $I(y)$. □

Theorem 81.17 (Ideal two-sided absorption obligation)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist, with $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. If $I : \mathsf{BHist} \to \mathsf{Prop}$ is an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R in the sense of Definition 81.1, then its certificate contains the left and right ambient absorption rows: for every carried ambient endpoint r with $C(r)$,*

$$I(x) \Longrightarrow I(r \cdot_R x) \wedge I(x \cdot_R r).$$

Proof. Unpack the $\mathsf{Ideal}^{\uparrow}$ certificate rows from Definition 81.1. The ambient multiplier r is not required to belong to the ideal predicate; it is only required to be carried by the $\mathsf{Ring}^{\uparrow}$ carrier C from the underlying $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$.

The two-sided absorption row is part of the listed certificate surface. Projecting it at the carried endpoint r and a membership witness $I(x)$ gives the pair of conclusions $I(r \cdot_R x)$ and $I(x \cdot_R r)$. □

Definition 81.18 (Ideal intersection predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, and let I and J be $\mathsf{Ideal}^{\uparrow}$ carried predicates over R in the sense of section 81.1. The intersection predicate $I \cap_R J$ is defined by*

$$(I \cap_R J)(x) \Longleftrightarrow I(x) \wedge J(x).$$

Theorem 81.19 (Ideal intersection closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over R. Assume the two ideal certificates provide the rows specified in section 81.2: carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried ring endpoint. For $K := I \cap_R J$ as in Definition 81.18, K is an $\mathsf{Ideal}^{\uparrow}$ predicate over R. Explicitly, K has carrier support and the rows*

$$K(0_R), \qquad K(x) \wedge K(y) \Longrightarrow K(x +_R y), \qquad K(x) \Longrightarrow K(-_R x),$$

$$K(x) \wedge K(y) \Longrightarrow K(x \cdot_R y), \qquad K(x) \wedge x \sim_R y \Longrightarrow K(y),$$

and, for every r with $C(r)$,

$$K(x) \Longrightarrow K(r \cdot_R x) \wedge K(x \cdot_R r).$$

Lean4 ✓ `BEDC.Derived.IdealUp.IdealIntersection closure rows`

Proof. The carrier support row is immediate. If $K(x)$, then $I(x)$ and $J(x)$; the carrier-support row of either ideal gives $C(x)$.

For zero, the zero rows for I and J give $I(0_R)$ and $J(0_R)$, hence $K(0_R)$.

For additive closure, assume $K(x)$ and $K(y)$. Then $I(x)$, $I(y)$, $J(x)$, and $J(y)$. The additive-closure row for I gives $I(x +_R y)$, and the additive-closure row for J gives $J(x +_R y)$. Together these are $K(x +_R y)$.

For additive inverse closure, assume $K(x)$. The inverse row for I gives $I(-_R x)$, and the inverse row for J gives $J(-_R x)$. Thus $K(-_R x)$.

For internal multiplication closure, assume $K(x)$ and $K(y)$. Carrier support gives $C(y)$. Right absorption in I, applied to $I(x)$ and $C(y)$, gives $I(x \cdot_R y)$. Right absorption in J, applied to $J(x)$ and $C(y)$, gives $J(x \cdot_R y)$. Hence $K(x \cdot_R y)$, which supplies the multiplicative closure row required by the sub-rng part of $\mathsf{Ideal}^{\uparrow}$.

For classifier transport, assume $K(x)$ and $x \sim_R y$. The transport row for I gives $I(y)$, and the transport row for J gives $J(y)$. Therefore $K(y)$.

For right absorption, assume $K(x)$ and $C(r)$. Right absorption in the two input ideal certificates gives $I(x \cdot_R r)$ and $J(x \cdot_R r)$, so $K(x \cdot_R r)$. For left absorption, the same argument with the left absorption rows gives $I(r \cdot_R x)$ and $J(r \cdot_R x)$, hence $K(r \cdot_R x)$.

These rows are exactly the carrier, additive sub-rng, classifier-stability, and ambient two-sided absorption rows required of an $\mathsf{Ideal}^{\uparrow}$ predicate over the given $\mathsf{Ring}^{\uparrow}$ certificate. □

Definition 81.20 (Ideal sum predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ carried predicates over R in the sense of section 81.1. The ideal-sum predicate $I +_R J$ is defined by the displayed additive decomposition*

$$(I +_R J)(z) \Longleftrightarrow \exists x, y : \mathsf{BHist}.\ I(x) \wedge J(y) \wedge z \sim_R x +_R y.$$

The ambient carrier witnesses are obtained through the carrier-support rows of the two $\mathsf{Ideal}^{\uparrow}$ certificates, the ring addition carrier row, and carrier transport along $\sim_R$. The hidden ledger for a sum membership contains only the two membership witnesses $I(x)$ and $J(y)$ together with the additive decomposition row $z \sim_R x +_R y$.

Definition 81.21 (Ideal inclusion predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, zero 0_R, addition $+_R$, additive inverse $-_R$, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ carried predicates over the same $\mathsf{Ring}^{\uparrow}$ certificate. The ideal-inclusion predicate $I \preceq_R J$ is defined by*

$$I \preceq_R J \iff \forall x : \mathsf{BHist}.\ I(x) \Longrightarrow J(x).$$

Theorem 81.22 (Ideal sum closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over R. Assume the two ideal certificates provide the rows specified in section 81.2: carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried ring endpoint. For $K := I +_R J$ as in Definition 81.20, K is an $\mathsf{Ideal}^{\uparrow}$ predicate over R. Explicitly, K has carrier support and the rows*

$$K(0_R), \qquad K(z) \wedge K(w) \Longrightarrow K(z +_R w), \qquad K(z) \Longrightarrow K(-_R z),$$

$$K(z) \wedge K(w) \Longrightarrow K(z \cdot_R w), \qquad K(z) \wedge z \sim_R z' \Longrightarrow K(z'),$$

and, for every r with $C(r)$,

$$K(z) \Longrightarrow K(r \cdot_R z) \wedge K(z \cdot_R r).$$

Lean4 ✓ `BEDC.Derived.IdealUp.IdealSum closure rows`

Proof. For carrier support, unpack a witness for $K(z)$. The support rows of I and J give $C(x)$ and $C(y)$, the ring addition carrier row gives $C(x +_R y)$, and carrier transport along $z \sim_R x +_R y$ gives $C(z)$.

For zero membership, use the witness pair $(0_R, 0_R)$. The zero rows give $I(0_R)$ and $J(0_R)$, and the additive-zero row of the ring certificate supplies the decomposition $0_R \sim_R 0_R +_R 0_R$.

For additive closure, choose witnesses $z \sim_R x +_R y$ for $K(z)$ and $w \sim_R x' +_R y'$ for $K(w)$. Addition congruence carries these decompositions to

$$z +_R w \sim_R (x +_R y) +_R (x' +_R y').$$

Additive associativity and commutativity in the $\mathsf{Ring}^{\uparrow}$ rows regroup the right side as $(x +_R x') +_R (y +_R y')$. The additive-closure rows of I and J give $I(x +_R x')$ and $J(y +_R y')$, which are the required witnesses for $K(z +_R w)$.

For additive inverse closure, start from $z \sim_R x +_R y$. The inverse-congruence row gives $-_R z \sim_R -_R(x +_R y)$, and the inverse-of-sum row identifies the right side with $(-_R x) +_R (-_R y)$. The additive inverse rows of I and J supply $I(-_R x)$ and $J(-_R y)$, so $K(-_R z)$.

For classifier transport, assume $K(z)$ and $z \sim_R z'$. Symmetry and transitivity of the classifier compose $z' \sim_R z$ with the displayed decomposition for z, giving the same witness pair for $K(z')$.

For absorption, assume $K(z)$ with witness $z \sim_R x +_R y$ and let r be carried. Distributivity gives

$$r \cdot_R z \sim_R (r \cdot_R x) +_R (r \cdot_R y)$$

and

$$z \cdot_R r \sim_R (x \cdot_R r) +_R (y \cdot_R r).$$

The left and right absorption rows of I and J supply the four displayed ideal memberships, so the two decompositions witness $K(r \cdot_R z)$ and $K(z \cdot_R r)$.

For internal multiplication closure, carrier support for $K(w)$ gives $C(w)$. Right absorption applied to $K(z)$ and $C(w)$ yields $K(z \cdot_R w)$, which supplies the multiplicative row required by the sub-rng part of $\mathsf{Ideal}^{\uparrow}$.

These rows are exactly the carrier, additive sub-rng, multiplicative sub-rng, classifier-stability, and two-sided ambient absorption rows required of an $\mathsf{Ideal}^{\uparrow}$ predicate over the fixed $\mathsf{Ring}^{\uparrow}$ certificate. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealSum ideal closure rows`

Theorem 81.23 (Ideal sum quotient-ring kernel scope)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate and let I and J be $\mathsf{Ideal}^{\uparrow}$ carried predicates over R. Let $K := I +_R J$ be the ideal-sum predicate of Definition 81.20, and let Q_K be the quotient-kernel source package of Definition 81.37. The package Q_K has the same diagonal and endpoint-transport scope as the quotient-kernel packages built from any other $\mathsf{Ideal}^{\uparrow}$ predicate, so $\mathsf{QuotientRing}^{\uparrow}$ may consume ideal sums without adding quotient primitives.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealSumQuotientKernel scope`

Proof. By Definition 81.22, the predicate K is an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ data as I and J.

Apply Definition 81.38 to K for the diagonal row, and apply Definition 81.39 to K for endpoint transport. The kernel relation is still the same concrete relation $Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x -_R y)$, so the quotient-ring consumer sees no additional quotient carrier or quotient primitive.

The ten-field fit is: traditional target sum of two ideals; BEDC source $\mathsf{Ring}^{\uparrow}$ plus two $\mathsf{Ideal}^{\uparrow}$ predicates over BHist; carrier the existential additive decomposition predicate; classifier $\sim_R$; stability Definition 81.22; ledger decomposition witnesses; core theorem Definition 81.22; exactness row exhaustion of the displayed witnesses; standard bridge absent; status $\mathsf{ProofTarget}$. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealSumQuotientKernel diagonal transport scope`

Theorem 81.24 (Ideal sum quotient-kernel absorption closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over R, set $K := I +_R J$ as in Definition 81.20, and define*

$$x -_R y := x +_R (-_R y), \qquad Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x -_R y).$$

Assume the ring certificate supplies multiplication carrier closure, carrier transport, classifier symmetry, subtraction congruence, and the absorption-difference rows

$$(r \cdot_R x) -_R (r \cdot_R y) \sim_R r \cdot_R (x -_R y)$$

and

$$(x \cdot_R r) -_R (y \cdot_R r) \sim_R (x -_R y) \cdot_R r.$$

Assume the two ideal certificates for I and J provide the rows listed in section 81.2. Then

$$C(r) \wedge Q_K(x, y) \Longrightarrow Q_K(r \cdot_R x, r \cdot_R y) \wedge Q_K(x \cdot_R r, y \cdot_R r).$$

Proof. First apply Definition 81.22 to the two ideal certificates. This turns the concrete sum predicate $K := I +_R J$ into an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ certificate, with carrier support, classifier transport, and two-sided absorption by every carried ring endpoint.

Now regard the displayed relation $Q_K(x, y)$ as the quotient-kernel source package associated to this specialized ideal predicate. The hypotheses on multiplication carrier closure, carrier transport, classifier symmetry, subtraction congruence, and the two absorption-difference rows are exactly the ambient ring rows required by Definition 81.50.

Apply Definition 81.50 with I there specialized to K here. Its left output gives $Q_K(r \cdot_R x, r \cdot_R y)$, its right output gives $Q_K(x \cdot_R r, y \cdot_R r)$, and together they give the claimed absorption closure for the ideal-sum quotient kernel. □

Theorem 81.25 (Ideal sum quotient-ring export surface)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over R, set $K := I +_R J$ as in Definition 81.20, and define*

$$x -_R y := x +_R (-_R y), \qquad Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x -_R y).$$

Assume the ambient ring certificate supplies the subtraction, carrier, classifier, and multiplication-difference rows required in Definition 81.24. Then the ideal-sum predicate K exports the full quotient-free kernel surface consumed by $\mathsf{QuotientRing}^{\uparrow}$:

1. *diagonal exactness for every carried endpoint x:*

$$C(x) \Longrightarrow Q_K(x, x);$$

2. *endpoint transport along the RingUp classifier:*

$$Q_K(x, y) \wedge x \sim_R x' \wedge y \sim_R y' \Longrightarrow Q_K(x', y');$$

3. *additive representative descent:*

$$Q_K(a, a') \wedge Q_K(b, b') \Longrightarrow Q_K(a +_R b, a' +_R b');$$

4. *negation representative descent:*

$$Q_K(a, a') \Longrightarrow Q_K(-_R a, -_R a');$$

5. *multiplication representative descent:*

$$Q_K(a, a') \wedge Q_K(b, b') \Longrightarrow Q_K(a \cdot_R b, a' \cdot_R b');$$

6. *ambient left and right multiplication closure:*

$$C(r) \wedge Q_K(x, y) \Longrightarrow Q_K(r \cdot_R x, r \cdot_R y) \wedge Q_K(x \cdot_R r, y \cdot_R r).$$

Proof. First apply Definition 81.22 to the two input ideal certificates. This turns the concrete predicate $K := I +_R J$ into an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ data, with the carrier, additive, negation, multiplication, classifier-transport, and two-sided absorption rows required by section 81.2.

Next instantiate Definition 81.41 with this predicate K. That theorem supplies the five generic quotient-ring rows: diagonal exactness, endpoint transport, additive descent, negation descent, and multiplication descent for the concrete relation $Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x -_R y)$.

Finally apply Definition 81.24. Its conclusion is exactly the ambient left and right multiplication closure row for Q_K, using the same subtraction package and the same specialized ideal-sum predicate K. □

Theorem 81.26 (Ideal intersection greatest lower bound)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over the same $\mathsf{Ring}^{\uparrow}$ certificate, with the rows specified in section 81.2. Let $K := I \cap_R J$ as in Definition 81.18. Then K is an $\mathsf{Ideal}^{\uparrow}$ predicate by Definition 81.19, and K is the greatest lower bound of I and J for $\preceq_R$:*

$$K \preceq_R I, \qquad K \preceq_R J,$$

and, for every $\mathsf{Ideal}^{\uparrow}$ predicate L over the same $\mathsf{Ring}^{\uparrow}$ certificate,

$$(L \preceq_R I) \wedge (L \preceq_R J) \Longrightarrow L \preceq_R K.$$

Proof. The predicate K is an $\mathsf{Ideal}^{\uparrow}$ predicate by Definition 81.19, because it is built from the same two ideal certificates and the same ambient $\mathsf{Ring}^{\uparrow}$ rows.

For the first lower-bound inclusion, assume $K(x)$. By Definition 81.18, this means $I(x) \wedge J(x)$. Projecting the first component gives $I(x)$, so Definition 81.21 gives $K \preceq_R I$.

For the second lower-bound inclusion, the same unpacking of $K(x)$ gives the second component $J(x)$. Hence $K \preceq_R J$.

It remains to prove maximality among lower bounds. Let L be an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ certificate, and assume $L \preceq_R I$ and $L \preceq_R J$. To prove $L \preceq_R K$, fix $x : \mathsf{BHist}$ and assume $L(x)$.

The first inclusion sends this witness to $I(x)$, and the second inclusion sends the same witness to $J(x)$. Pairing these two memberships gives $I(x) \wedge J(x)$, which is exactly $K(x)$ by Definition 81.18. Therefore $L \preceq_R K$ by Definition 81.21. □

Theorem 81.27 (Ideal sum commutativity as a carried predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I and J be $\mathsf{Ideal}^{\uparrow}$ predicates over the same $\mathsf{Ring}^{\uparrow}$ certificate, with the rows specified in section 81.2. Define*

$$K_{I,J} := I +_R J, \qquad K_{J,I} := J +_R I$$

using Definition 81.20. Then $K_{I,J}$ and $K_{J,I}$ are $\mathsf{Ideal}^{\uparrow}$ predicates over the same ambient $\mathsf{Ring}^{\uparrow}$ certificate by Definition 81.22, and they have the same membership predicate:

$$\forall z : \mathsf{BHist}.\ K_{I,J}(z) \Longleftrightarrow K_{J,I}(z).$$

The comparison keeps the ambient carrier row C fixed, swaps only the two additive-decomposition membership witnesses, and preserves the same Pkg provenance rows attached to the ambient $\mathsf{Ring}^{\uparrow}$ certificate and the two input $\mathsf{Ideal}^{\uparrow}$ certificates.

Proof. Apply Definition 81.22 first to the ordered pair (I, J) and then to the ordered pair (J, I). This gives $K_{I,J}$ and $K_{J,I}$ as $\mathsf{Ideal}^{\uparrow}$ predicates over the same carrier C, classifier $\sim_R$, and ring operations.

Fix $z : \mathsf{BHist}$. For the forward implication, assume $K_{I,J}(z)$. By Definition 81.20, choose witnesses $x, y : \mathsf{BHist}$ such that

$$I(x) \wedge J(y) \wedge z \sim_R x +_R y.$$

The additive commutativity row contained in the additive abelian-group part of the $\mathsf{Ring}^{\uparrow}$ stability certificate gives

$$x +_R y \sim_R y +_R x.$$

Classifier transitivity composes the displayed decomposition for z with this commutativity row, giving

$$z \sim_R y +_R x.$$

Now the witness pair (y, x), together with $J(y)$, $I(x)$, and the last classifier comparison, is exactly a witness for $K_{J,I}(z)$ under Definition 81.20.

The reverse implication is the same argument with I and J interchanged. If $K_{J,I}(z)$ is witnessed by $J(y) \wedge I(x) \wedge z \sim_R y +_R x$, additive commutativity gives $y +_R x \sim_R x +_R y$, and classifier transitivity gives $z \sim_R x +_R y$. The witness pair (x, y) therefore proves $K_{I,J}(z)$.

The ambient carrier consequence is unchanged in both directions. Since both sum predicates are $\mathsf{Ideal}^{\uparrow}$ predicates over the same $\mathsf{Ring}^{\uparrow}$ certificate, the carrier-support row of Definition 81.1 reads

$$K_{I,J}(z) \Longrightarrow C(z), \qquad K_{J,I}(z) \Longrightarrow C(z).$$

The equivalence just proved keeps the endpoint z fixed and changes only the internal decomposition ledger: $I(x)$ and $J(y)$ are reused in the opposite witness slots, while the decomposition row is adjusted by the additive commutativity row already present in Definition 43.26. No ambient carrier, classifier, operation, ideal certificate, or package provenance row is replaced. □

Theorem 81.28 (Ideal sum monotonicity in ideal inclusion)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I, I', J, J' be $\mathsf{Ideal}^{\uparrow}$ predicates over the same $\mathsf{Ring}^{\uparrow}$ certificate, with the rows specified in section 81.2. Define*

$$K := I +_R J, \qquad K' := I' +_R J'$$

using Definition 81.20. Then K and K' are $\mathsf{Ideal}^{\uparrow}$ predicates over the same ambient certificate by Definition 81.22, and ideal inclusion is monotone through the sum constructor:

$$(I \preceq_R I') \wedge (J \preceq_R J') \Longrightarrow K \preceq_R K'.$$

Equivalently, for every $z : \mathsf{BHist}$, any additive-decomposition witness for $(I +_R J)(z)$ is sent to an additive-decomposition witness for $(I' +_R J')(z)$ by applying the two inclusion implications to the two membership entries and leaving the decomposition row unchanged.

Proof. The two sum predicates are $\mathsf{Ideal}^{\uparrow}$ predicates by Definition 81.22, applied first to (I, J) and then to (I', J'). Thus the comparison $K \preceq_R K'$ is an instance of the ideal-inclusion predicate from Definition 81.21.

Assume $I \preceq_R I'$ and $J \preceq_R J'$. To prove $K \preceq_R K'$, fix $z : \mathsf{BHist}$ and assume $K(z)$. By Definition 81.20, choose witnesses $x, y : \mathsf{BHist}$ with

$$I(x) \wedge J(y) \wedge z \sim_R x +_R y.$$

The first inclusion, unpacked through Definition 81.21, sends the witness $I(x)$ to $I'(x)$. The second inclusion sends the witness $J(y)$ to $J'(y)$.

Now keep the same endpoints x and y and keep the same additive-decomposition row $z \sim_R x +_R y$. The data

$$I'(x) \wedge J'(y) \wedge z \sim_R x +_R y$$

are exactly a witness for $K'(z)$ under Definition 81.20. Since z was arbitrary, Definition 81.21 gives $K \preceq_R K'$. □

Theorem 81.29 (Ideal sum associativity as a carried predicate)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I, J, and K be* $\mathsf{Ideal}^{\uparrow}$ *predicates over the same* $\mathsf{Ring}^{\uparrow}$ *certificate, with the rows specified in section 81.2. Define*

$$A_L := (I +_R J) +_R K, \qquad A_R := I +_R (J +_R K)$$

using Definition 81.20. Then A_L and A_R are $\mathsf{Ideal}^{\uparrow}$ *predicates over the same ambient* $\mathsf{Ring}^{\uparrow}$ *certificate by Definition 81.22, and they have the same membership predicate:*

$$\forall z : \mathsf{BHist}.\ A_L(z) \iff A_R(z).$$

The comparison keeps the ambient carrier row C fixed, regroups only the three additive-decomposition witnesses by the additive associativity row in the $\mathsf{Ring}^{\uparrow}$ *stability certificate, and preserves the same* Pkg *provenance rows attached to the ambient* $\mathsf{Ring}^{\uparrow}$ *certificate and the three input* $\mathsf{Ideal}^{\uparrow}$ *certificates.*

Proof. Apply Definition 81.22 to (I, J), to (J, K), to $((I +_R J), K)$, and to $(I, (J +_R K))$. Thus the two iterated sums are carried $\mathsf{Ideal}^{\uparrow}$ predicates over the same carrier C, classifier $\sim_R$, and ring operations. The proof of equality of predicates is therefore a direct comparison of the displayed membership witnesses in Definition 81.20.

Fix $z : \mathsf{BHist}$. For the forward implication, assume $A_L(z)$. Unpacking the outer sum gives histories $s, k : \mathsf{BHist}$ with

$$(I +_R J)(s) \wedge K(k) \wedge z \sim_R s +_R k.$$

Unpacking the inner sum gives histories $i, j : \mathsf{BHist}$ with

$$I(i) \wedge J(j) \wedge s \sim_R i +_R j.$$

Use $j +_R k$ as the inner right-sum endpoint. The witness pair (j, k), the memberships $J(j)$ and $K(k)$, and classifier reflexivity at $j +_R k$ prove $(J +_R K)(j +_R k)$ by Definition 81.20.

It remains to supply the outer decomposition for $A_R(z)$ with witness pair $(i, j +_R k)$. Addition congruence, applied to $s \sim_R i +_R j$ and classifier reflexivity at k, gives

$$s +_R k \sim_R (i +_R j) +_R k.$$

The additive associativity row contained in the additive abelian-group part of Definition 43.26 gives

$$(i +_R j) +_R k \sim_R i +_R (j +_R k).$$

Classifier transitivity composes the displayed decomposition $z \sim_R s +_R k$ with these two comparisons, yielding

$$z \sim_R i +_R (j +_R k).$$

Together with $I(i)$ and $(J +_R K)(j +_R k)$, this is exactly a witness for $A_R(z)$.

For the reverse implication, assume $A_R(z)$. Unpacking the outer sum gives histories $i, t : \mathsf{BHist}$ with

$$I(i) \wedge (J +_R K)(t) \wedge z \sim_R i +_R t.$$

Unpacking the inner sum gives histories $j, k : \mathsf{BHist}$ with

$$J(j) \wedge K(k) \wedge t \sim_R j +_R k.$$

Use $i +_R j$ as the inner left-sum endpoint. The witness pair (i, j), the memberships $I(i)$ and $J(j)$, and classifier reflexivity at $i +_R j$ prove $(I +_R J)(i +_R j)$ by Definition 81.20.

Addition congruence, applied to classifier reflexivity at i and to $t \sim_R j +_R k$, gives

$$i +_R t \sim_R i +_R (j +_R k).$$

The same additive associativity row, read with classifier symmetry, gives

$$i +_R (j +_R k) \sim_R (i +_R j) +_R k.$$

Classifier transitivity composes these comparisons with $z \sim_R i +_R t$, yielding

$$z \sim_R (i +_R j) +_R k.$$

Together with $(I +_R J)(i +_R j)$ and $K(k)$, this is exactly a witness for $A_L(z)$.

The carrier consequence is identical on both sides. Since both predicates are obtained from Definition 81.22 over the same ambient $\mathsf{Ring}^{\uparrow}$ certificate, their carrier-support rows read

$$A_L(z) \Longrightarrow C(z), \qquad A_R(z) \Longrightarrow C(z).$$

The equivalence keeps the endpoint z fixed and changes only the parenthesization of the internal decomposition ledger: the same three memberships $I(i)$, $J(j)$, and $K(k)$ are reused, while the classifier decomposition is adjusted by addition congruence, classifier transitivity, classifier symmetry in the reverse direction, and the additive associativity row already present in Definition 43.26. No ambient carrier, classifier, operation, ideal certificate, or package provenance row is replaced. □

Definition 81.30 (Principal two-sided generated ideal predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let a be a carried generator, so $C(a)$. The principal two-sided generated ideal predicate $\langle a \rangle_R$ is the predicate on endpoints z for which a finite* HistSpine *carries signed two-sided monomial rows*

$$\ell_i \cdot_R a \cdot_R r_i$$

with $C(\ell_i)$ and $C(r_i)$ at every spine position i, together with sign rows, such that the $\mathsf{Ring}^{\uparrow}$ finite additive fold of those signed rows is classifier-equal to z. The empty spine folds to 0_R. The certificate ledger records the spine, the left and right coefficient rows, the sign rows, and the final classifier edge from the finite fold to z.

Lean4 (definition) `BEDC.Derived.IdealUp.IdealPrincipalGenerated`

Theorem 81.31 (Principal generated subtractive closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let a be a carried generator, and let $\langle a \rangle_R$ be the predicate of Definition 81.30. Then $\langle a \rangle_R$ has carrier support, zero membership, additive closure, additive inverse closure, and classifier transport over the ambient $\mathsf{Ring}^{\uparrow}$ data:*

$$\langle a \rangle_R(0_R), \qquad \langle a \rangle_R(x) \wedge \langle a \rangle_R(y) \Longrightarrow \langle a \rangle_R(x +_R y), \qquad \langle a \rangle_R(x) \Longrightarrow \langle a \rangle_R(-_R x),$$

and

$$\langle a \rangle_R(x) \wedge x \sim_R y \Longrightarrow \langle a \rangle_R(y),$$

with every displayed membership carrying a RingUp endpoint carrier witness.

Lean4 ✓ `BEDC.Derived.IdealUp.IdealPrincipalGenerated subtractive closure rows`

Proof. Carrier support is part of the final fold certificate: every signed monomial row is carried by the multiplication-closure rows, the finite additive fold is carried by the additive-closure rows, and classifier transport carries that support to the named endpoint z.

For zero membership, take the empty spine. Its fold is 0_R, and the final classifier edge is the reflexive edge at 0_R.

For additive closure, concatenate the two finite spines witnessing x and y. The fold of the concatenated spine is classified, by RingUp finite-fold associativity, with the sum of the two earlier folds. Addition congruence then transports along the two final classifier edges and yields a final classifier edge to $x +_R y$.

For additive inverse closure, reverse the signs in the spine witnessing x. The RingUp inverse and finite-fold rows classify the new fold with the additive inverse of the old fold, and inverse congruence transports the final edge to $-_R x$.

For classifier transport, keep the same spine and compose its final classifier edge with the given edge $x \sim_R y$. The ledger data is unchanged, and the composed edge witnesses $\langle a \rangle_R(y)$. □

Theorem 81.32 (Principal generated absorption closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let a be a carried generator, and let $\langle a \rangle_R$ be the predicate of Definition 81.30. If t is carried, then $\langle a \rangle_R(x)$ implies both $\langle a \rangle_R(t \cdot_R x)$ and $\langle a \rangle_R(x \cdot_R t)$. Consequently, together with Definition 81.31, the predicate $\langle a \rangle_R$ supplies the internal multiplication row required of an $\mathsf{Ideal}^{\uparrow}$ predicate.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealPrincipalGenerated absorption closure rows`

Proof. Start with a finite signed spine witnessing $\langle a \rangle_R(x)$. For left absorption by a carried endpoint t, multiply every monomial row in the spine on the left by t.

Associativity rewrites each row

$$t \cdot_R (\ell_i \cdot_R a \cdot_R r_i)$$

as

$$(t \cdot_R \ell_i) \cdot_R a \cdot_R r_i.$$

Carrier support for t and ℓ_i gives $C(t \cdot_R \ell_i)$, while $C(r_i)$ is retained. Distributivity and the finite-fold multiplication rows classify the fold of the transformed spine with $t \cdot_R x$, after composing with the original final classifier edge.

For right absorption, multiply every monomial row on the right by t. Associativity rewrites

$$(\ell_i \cdot_R a \cdot_R r_i) \cdot_R t$$

as

$$\ell_i \cdot_R a \cdot_R (r_i \cdot_R t),$$

and the multiplication-closure rows give $C(r_i \cdot_R t)$. The same distributivity and finite-fold argument supplies the final classifier edge to $x \cdot_R t$.

Internal multiplication is then a consequence of right absorption: from $\langle a \rangle_R(x)$ and $\langle a \rangle_R(y)$, carrier support from Definition 81.31 gives $C(y)$, and right absorption by y gives $\langle a \rangle_R(x \cdot_R y)$. □

Theorem 81.33 (Principal generated full IdealUp closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist, let a be a carried generator, and let $\langle a \rangle_R$ be the predicate of Definition 81.30. Then $\langle a \rangle_R$ supplies the full $\mathsf{Ideal}^{\uparrow}$ predicate surface over R: carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and two-sided absorption by every carried ambient endpoint.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealPrincipalGenerated full ideal closure rows`

Proof. The carrier-support, zero-membership, additive-closure, additive-inverse, and classifier-transport rows are exactly the rows supplied by Definition 81.31.

The two-sided ambient absorption rows are exactly the left and right absorption rows supplied by Definition 81.32. The internal multiplication row is obtained there from right absorption together with carrier support for the second factor. These rows are precisely the row package required of an $\mathsf{Ideal}^{\uparrow}$ predicate over the ambient $\mathsf{Ring}^{\uparrow}$ certificate. □

Theorem 81.34 (Principal generated quotient-kernel scope)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate, let a be a carried generator, and let $I := \langle a \rangle_R$ be the predicate of Definition 81.30. The generated predicate supplies the quotient-free kernel surface consumed by $\mathsf{QuotientRing}^{\uparrow}$:*

$$Q_I(x, y) \iff C(x) \wedge C(y) \wedge I(x -_R y).$$

In particular, the diagonal and endpoint-transport rows for this surface are exactly the rows obtained by applying Definition 81.38 and Definition 81.39 to I.

Proof. By Definition 81.33, the predicate I is an $\mathsf{Ideal}^{\uparrow}$ predicate over the ambient ring certificate.

Unfold Definition 81.37 with this I. The displayed relation is therefore the same concrete quotient-free BHist kernel package used for every IdealUp predicate.

Apply Definition 81.38 to obtain diagonal exactness, and apply Definition 81.39 to obtain endpoint transport. No quotient carrier or extra quotient primitive is introduced before $\mathsf{QuotientRing}^{\uparrow}$ consumes the surface. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealPrincipalGenerated quotient kernel scope`

Definition 81.35 (Ring map zero-fiber predicate)**.** *Let S and T carry $\mathsf{Ring}^{\uparrow}$ certificates, with carriers C_S, C_T, classifiers $\sim_S, \sim_T$, zeros $0_S, 0_T$, additions $+_S, +_T$, additive inverses $-_S, -_T$, and multiplications $\cdot_S, \cdot_T$. For a function $f : S \to T$, the ring-map zero-fiber predicate on the source carrier is*

$$\mathsf{Ker}_f^{\mathrm{ring}}(x) \iff C_S(x) \wedge f(x) \sim_T 0_T.$$

The first component records source-carrier evidence; the second records the target-zero classification.

Lean4 ✓ `BEDC.Derived.IdealUp.RingMapZeroFiber semantic name certificate`

Theorem 81.36 (Ring map zero fiber IdealUp closure)**.** *Let S and T carry $\mathsf{Ring}^{\uparrow}$ certificates as in chapter 43, with notation as in Definition 81.35. Let $f : S \to T$ satisfy carrier preservation and classifier preservation on carried source endpoints:*

$$C_S(x) \Longrightarrow C_T(f(x))$$

and

$$C_S(x) \wedge C_S(y) \wedge x \sim_S y \Longrightarrow f(x) \sim_T f(y).$$

Assume also that f preserves zero, addition, additive inverse, and multiplication on carried source endpoints:

$$f(0_S) \sim_T 0_T,$$

$$C_S(x) \wedge C_S(y) \Longrightarrow f(x +_S y) \sim_T f(x) +_T f(y),$$

$$C_S(x) \Longrightarrow f(-_S x) \sim_T -_T f(x),$$

and

$$C_S(x) \wedge C_S(y) \Longrightarrow f(x \cdot_S y) \sim_T f(x) \cdot_T f(y).$$

Then $K := \mathsf{Ker}_f^{\mathrm{ring}}$ *from Definition 81.35 is an* $\mathsf{Ideal}^{\uparrow}$ *predicate over the source ring* S*. Explicitly,* K *has carrier support and the rows*

$$K(0_S), \qquad K(x) \wedge K(y) \Longrightarrow K(x +_S y), \qquad K(x) \Longrightarrow K(-_S x),$$

$$K(x) \wedge K(y) \Longrightarrow K(x \cdot_S y), \qquad K(x) \wedge x \sim_S y \Longrightarrow K(y),$$

and, for every r *with* $C_S(r)$*,*

$$K(x) \Longrightarrow K(r \cdot_S x) \wedge K(x \cdot_S r).$$

Lean4 ✓ `BEDC.Derived.IdealUp.RingMapZeroFiber ideal closure rows`

Proof. Unfold Definition 81.35. Carrier support is the first component of a K witness.

For zero, the source $\mathsf{Ring}^{\uparrow}$ certificate carries 0_S, and the zero-preservation row gives $f(0_S) \sim_T 0_T$. Hence $K(0_S)$.

For classifier transport, assume $K(x)$ and $x \sim_S y$. The source carrier-transport row from Definition 9.7 gives $C_S(y)$. Classifier preservation gives $f(x) \sim_T f(y)$. Target classifier symmetry and Definition 9.8 compose this with the zero-fiber witness $f(x) \sim_T 0_T$ to obtain $f(y) \sim_T 0_T$. Thus $K(y)$.

For additive closure, assume $K(x)$ and $K(y)$. Source additive closure carries $x +_S y$. Addition preservation gives

$$f(x +_S y) \sim_T f(x) +_T f(y).$$

Target addition congruence, applied to the two zero-fiber witnesses, gives

$$f(x) +_T f(y) \sim_T 0_T +_T 0_T.$$

The target additive zero row contained in Definition 43.26 reduces $0_T +_T 0_T$ to 0_T. Transitivity gives $f(x +_S y) \sim_T 0_T$, so $K(x +_S y)$.

For additive inverse closure, assume $K(x)$. Source inverse closure carries $-_S x$. Additive-inverse preservation gives

$$f(-_S x) \sim_T -_T f(x).$$

Target inverse congruence transports $f(x) \sim_T 0_T$ to $-_T f(x) \sim_T -_T 0_T$, and Definition 43.28, read in the target additive component, identifies $-_T 0_T$ with 0_T. Composing the classifier edges yields $f(-_S x) \sim_T 0_T$, hence $K(-_S x)$.

For internal multiplication closure, assume $K(x)$ and $K(y)$. Source multiplication closure carries $x \cdot_S y$. Multiplication preservation gives

$$f(x \cdot_S y) \sim_T f(x) \cdot_T f(y).$$

Target multiplication congruence transports the two zero-fiber witnesses to

$$f(x) \cdot_T f(y) \sim_T 0_T \cdot_T 0_T.$$

The zero-product rows of Definition 43.26 reduce $0_T \cdot_T 0_T$ to 0_T. Thus $K(x \cdot_S y)$.

For left absorption, assume $K(x)$ and $C_S(r)$. Source multiplication closure carries $r \cdot_S x$, and multiplication preservation gives

$$f(r \cdot_S x) \sim_T f(r) \cdot_T f(x).$$

Carrier preservation gives $C_T(f(r))$. Target multiplication congruence transports the right factor along $f(x) \sim_T 0_T$, and the target right zero-product row gives

$$f(r) \cdot_T 0_T \sim_T 0_T.$$

Therefore $K(r \cdot_S x)$. For right absorption, the same argument starts from

$$f(x \cdot_S r) \sim_T f(x) \cdot_T f(r),$$

transports the left factor along $f(x) \sim_T 0_T$, and uses the target left zero-product row $0_T \cdot_T f(r) \sim_T 0_T$. This gives $K(x \cdot_S r)$.

The displayed carrier support, zero, additive, inverse, multiplicative, classifier-transport, and two-sided absorption rows are exactly the $\mathsf{Ideal}^{\uparrow}$ certificate rows listed in section 81.2 for a predicate over the source $\mathsf{Ring}^{\uparrow}$ certificate. □

Definition 81.37 (Ideal quotient-kernel source package)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R, and write*

$$x -_R y := x +_R (-_R y).$$

The ideal quotient-kernel source package is the concrete BHist *relation Q_I on ring endpoints defined by*

$$Q_I(x, y) \iff C(x) \wedge C(y) \wedge I(x -_R y).$$

It is the quotient-free BHist *relation later read as the ideal-coset relation in Definition 83.1.*

Lean4 (definition) `BEDC.Derived.IdealUp.IdealQuotientKernelSource`

Theorem 81.38 (Ideal quotient-kernel diagonal exactness)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with notation as in Definition 81.37, and let I be an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R. Assume the ideal certificate supplies zero membership and classifier transport, and the ring certificate supplies the additive inverse and zero rows. If $C(x)$, then $Q_I(x, x)$.*

Proof. Unfold Definition 81.37. The two endpoint carrier witnesses are both the given evidence $C(x)$.

It remains to show $I(x -_R x)$. By the additive inverse row in the underlying ring certificate, $x +_R (-_R x)$ is classified with 0_R. The ideal zero row gives $I(0_R)$, and classifier symmetry orients the comparison from 0_R to $x -_R x$.

Classifier transport for I carries zero membership across this edge. Repacking the two carrier witnesses with the transported ideal membership gives $Q_I(x, x)$. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel diagonal exactness`

Theorem 81.39 (Ideal quotient-kernel transport surface)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with notation as in Definition 81.37, and let I be an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R. Assume the ring certificate supplies carrier transport, additive inverse congruence, addition congruence, and the classifier-equivalence rows, and assume the ideal certificate supplies classifier transport. If $Q_I(x, y)$, $x \sim_R x'$, and $y \sim_R y'$, then $Q_I(x', y')$.*

Proof. Unfold Definition 81.37. From $Q_I(x, y)$ we have $C(x)$, $C(y)$, and $I(x -_R y)$.

Carrier transport along $x \sim_R x'$ and $y \sim_R y'$ gives $C(x')$ and $C(y')$. Additive inverse congruence sends $y \sim_R y'$ to $-_R y \sim_R -_R y'$, and addition congruence combines this with $x \sim_R x'$ to obtain

$$x -_R y \sim_R x' -_R y'.$$

The ideal classifier-transport row carries $I(x -_R y)$ across this subtraction comparison, giving $I(x' -_R y')$. Pairing this membership with the transported carriers gives $Q_I(x', y')$. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel endpoint transport`

Theorem 81.40 (Ideal quotient-kernel multiplication descent)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with notation as in Definition 81.37, and let I be an* $\mathsf{Ideal}^{\uparrow}$ *carried predicate over R. Assume the ring certificate supplies multiplication carrier rows and the product-difference classifier row, and assume the ideal certificate supplies classifier transport, additive closure, and left and right absorption by every carried endpoint. If $Q_I(a, a')$ and $Q_I(b, b')$, then the IdealUp rows give*

$$I((a \cdot_R b) -_R (a' \cdot_R b')).$$

Together with the RingUp multiplication carrier rows, this repacks as

$$Q_I(a \cdot_R b, a' \cdot_R b').$$

Ten-field fit: traditional target quotient-kernel multiplication descent; BEDC source $\mathsf{Ring}^{\uparrow}$ *plus* $\mathsf{Ideal}^{\uparrow}$ *over* BHist*; carrier Q_I endpoints; classifier $\sim_R$; stability absorption, additive closure, and classifier transport; ledger two representative differences and two absorbed products; core theorem this block; exactness the displayed repacking; standard bridge absent; status proof target.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel multiplication descent`

Proof. Unfold Definition 81.37. From $Q_I(a, a')$ and $Q_I(b, b')$ we obtain carried endpoints $C(a)$, $C(a')$, $C(b)$, and $C(b')$, together with ideal memberships for the representative differences. Write

$$d_a := a -_R a', \qquad d_b := b -_R b'.$$

Right absorption applies $I(d_a)$ against the carried endpoint b, giving $I(d_a \cdot_R b)$. Left absorption applies $I(d_b)$ against the carried endpoint a', giving $I(a' \cdot_R d_b)$. Additive closure combines these two absorbed products into membership of their ring sum.

The product-difference classifier row identifies that sum with $(a \cdot_R b) -_R (a' \cdot_R b')$. Classifier transport for the ideal predicate carries the combined membership across this comparison, yielding

$$I((a \cdot_R b) -_R (a' \cdot_R b')).$$

Finally, multiplication carrier rows applied to $C(a), C(b)$ and to $C(a'), C(b')$ give the two product carrier witnesses. Pairing those witnesses with the transported product-difference membership is exactly $Q_I(a \cdot_R b, a' \cdot_R b')$. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel product difference membership`

Theorem 81.41 (Ideal quotient-ring export surface)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate and let I be an* $\mathsf{Ideal}^{\uparrow}$ *carried predicate over R, with Q_I as in Definition 81.37. The IdealUp rows export the full quotient-free kernel surface consumed by* $\mathsf{QuotientRing}^{\uparrow}$ *in chapter 83:*

1. *diagonal exactness for every carried endpoint x:*

$$C(x) \Longrightarrow Q_I(x, x);$$

2. *endpoint transport along the RingUp classifier:*

$$Q_I(x, y) \wedge x \sim_R x' \wedge y \sim_R y' \Longrightarrow Q_I(x', y');$$

3. *additive representative descent:*

$$Q_I(a, a') \wedge Q_I(b, b') \Longrightarrow Q_I(a +_R b, a' +_R b');$$

4. *negation representative descent:*

$$Q_I(a, a') \Longrightarrow Q_I(-_R a, -_R a');$$

5. *multiplication representative descent:*

$$Q_I(a, a') \wedge Q_I(b, b') \Longrightarrow Q_I(a \cdot_R b, a' \cdot_R b').$$

The zero-fiber and preimage constructions of Definition 81.35, Definition 81.36, Definition 81.46, and Definition 81.47 therefore produce concrete inputs to this same quotient-kernel surface.

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel export surface`

Proof. The diagonal component is exactly Definition 81.38. The endpoint-transport component is exactly Definition 81.39.

The additive component is the additive representative row of the same checked export surface. It combines the two input kernel witnesses by the ideal additive row, transports across the RingUp comparison between the sum of representative differences and the difference of sums, and repacks the two sum carriers. This is the surface used by Definition 83.3.

The negation component is the negation representative row of the same checked export surface. It applies the ideal additive-inverse row to the representative difference, transports across the RingUp comparison between the negated difference and the difference of negated representatives, and repacks the two negated carriers.

The multiplication component is exactly Definition 81.40, which isolates the product-difference membership used in Definition 83.2.

Together these five rows are the diagonal, endpoint transport, additive descent, negation descent, and multiplication descent projections returned by the checked Lean package. Thus an IdealUp predicate, including one obtained as a zero fiber or as a ring-map preimage, supplies the concrete quotient-kernel source required before any quotient carrier is formed. □

Theorem 81.42 (Principal generated ideal is least over its generator)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate over BHist, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let a be a carried generator, let $\langle a \rangle_R$ be the principal two-sided generated ideal predicate of Definition 81.30, and let $I : \mathsf{BHist} \to \mathsf{Prop}$ be an $\mathsf{Ideal}^{\uparrow}$ carried predicate over R in the sense of Definition 81.1. If $I(a)$, then*

$$\langle a \rangle_R \preceq_R I.$$

Equivalently, every endpoint admitted by the finite signed two-sided monomial ledger defining $\langle a \rangle_R$ is a member of any carried ideal predicate that contains the generator a.

Proof. By Definition 81.21, it is enough to fix an endpoint $z : \mathsf{BHist}$ and show that a witness for $\langle a \rangle_R(z)$ yields a witness for $I(z)$.

Unpack the principal witness from Definition 81.30. It consists of a finite $\mathsf{HistSpine}$, carried coefficient rows $C(\ell_i)$ and $C(r_i)$, a sign row at each position, a finite additive fold of the signed rows, and a final classifier edge from that fold to z.

First consider one unsigned monomial row. From the assumed generator membership $I(a)$ and the left ambient absorption row of the $\mathsf{Ideal}^{\uparrow}$ certificate, applied to the carried coefficient $C(\ell_i)$, obtain

$$I(\ell_i \cdot_R a).$$

Applying the right ambient absorption row to the carried coefficient $C(r_i)$ gives

$$I((\ell_i \cdot_R a) \cdot_R r_i).$$

The ambient $\mathsf{Ring}^{\uparrow}$ associativity and classifier-transport row for I read this as membership of the displayed two-sided monomial $\ell_i \cdot_R a \cdot_R r_i$ used in the principal ledger.

If the sign row is positive, this is the required membership for the signed row. If the sign row is negative, the additive-inverse row of the $\mathsf{Ideal}^{\uparrow}$ certificate sends the monomial membership to membership of its additive inverse. Thus every signed row in the finite principal ledger lies in I.

Now fold the finite spine. For the empty spine, the $\mathsf{Ideal}^{\uparrow}$ zero-membership row gives $I(0_R)$, matching the empty fold. For a nonempty spine, induct along the displayed finite order of the ledger: the induction hypothesis gives membership of the partial fold, the preceding paragraph gives membership of the next signed row, and the additive-closure row of the $\mathsf{Ideal}^{\uparrow}$ certificate gives membership of the extended fold. Hence the whole finite additive fold lies in I.

Finally, project the classifier edge from the principal witness, unfold the classifier-transport row of the $\mathsf{Ideal}^{\uparrow}$ certificate, and repack the fold membership across that edge as $I(z)$. Since z and its principal witness were arbitrary, Definition 81.21 gives $\langle a \rangle_R \preceq_R I$. □

Definition 81.43 (Finite ideal-intersection predicate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let $xs : \mathsf{HistSpine}$ be a finite BEDC-history spine, and let $\mathcal{I}$ assign to every position $i : \mathsf{Pos}(xs)$ a predicate $\mathcal{I}_i : \mathsf{BHist} \to \mathsf{Prop}$ over the same ring data. The finite relative intersection predicate is*

$$\mathsf{FinIdealMeet}_{C,xs,\mathcal{I}}(x) \iff C(x) \wedge \forall i : \mathsf{Pos}(xs).\, \mathcal{I}_i(x).$$

The carrier conjunct makes the empty finite spine denote the ambient carried ring predicate, while a nonempty spine is the usual pointwise intersection of the displayed ideal predicates.

Lean4 (definition) `BEDC.Derived.IdealUp.FiniteIdealMeet`

Theorem 81.44 (Finite ideal intersection closure)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate as in chapter 43, with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let $xs : \mathsf{HistSpine}$ be a finite BEDC-history spine, and let $\mathcal{I}$ assign to every position $i : \mathsf{Pos}(xs)$ a predicate $\mathcal{I}_i : \mathsf{BHist} \to \mathsf{Prop}$. Assume that, for every $i : \mathsf{Pos}(xs)$, the predicate $\mathcal{I}_i$ is an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ data, with the rows specified in section 81.2: carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried ring endpoint. For*

$$K(x) \iff \mathsf{FinIdealMeet}_{C,xs,\mathcal{I}}(x),$$

the predicate K is an $\mathsf{Ideal}^{\uparrow}$ *predicate over R. Explicitly, K has carrier support and the rows*

$$K(0_R), \qquad K(x) \wedge K(y) \Longrightarrow K(x +_R y), \qquad K(x) \Longrightarrow K(-_R x),$$

$$K(x) \wedge K(y) \Longrightarrow K(x \cdot_R y), \qquad K(x) \wedge x \sim_R y \Longrightarrow K(y),$$

and, for every r with $C(r)$,

$$K(x) \Longrightarrow K(r \cdot_R x) \wedge K(x \cdot_R r).$$

Lean4 ✓ `BEDC.Derived.IdealUp.FiniteIdealMeet closure rows`

Proof. Unfold Definition 81.43. The carrier support row is the first conjunct of $K(x)$.

For zero membership, the ring certificate supplies $C(0_R)$. Fix a position $i : \mathsf{Pos}(xs)$. The zero row in the ith ideal certificate gives $\mathcal{I}_i(0_R)$. Since i was arbitrary, 0_R satisfies every predicate in the finite family, and together with $C(0_R)$ this is $K(0_R)$.

For additive closure, assume $K(x)$ and $K(y)$. The carrier conjuncts give $C(x)$ and $C(y)$, so the additive closure row of the ring certificate gives $C(x +_R y)$. For an arbitrary position i, the universal conjuncts of $K(x)$ and $K(y)$ give $\mathcal{I}_i(x)$ and $\mathcal{I}_i(y)$. The additive closure row of the ith ideal certificate gives $\mathcal{I}_i(x +_R y)$. Thus every indexed predicate holds at $x +_R y$, and the carrier witness just obtained repacks as $K(x +_R y)$.

For additive inverse closure, assume $K(x)$. The ring inverse-closure row gives $C(-_R x)$ from $C(x)$. For each position i, the inverse row of the ith ideal certificate gives $\mathcal{I}_i(-_R x)$ from $\mathcal{I}_i(x)$. Hence all finite-family predicates hold at $-_R x$, and $K(-_R x)$ follows.

For internal multiplication closure, assume $K(x)$ and $K(y)$. The ring multiplication-closure row gives $C(x \cdot_R y)$ from $C(x)$ and $C(y)$. For every position i, the universal conjuncts give $\mathcal{I}_i(x)$ and $\mathcal{I}_i(y)$, and the internal multiplication row of the ith ideal certificate gives $\mathcal{I}_i(x \cdot_R y)$. Therefore $K(x \cdot_R y)$.

For classifier transport, assume $K(x)$ and $x \sim_R y$. Carrier transport from the naming certificate underlying the $\mathsf{Ring}^{\uparrow}$ data carries $C(x)$ along $x \sim_R y$ to obtain $C(y)$. For each position i, the classifier-transport row of the ith ideal certificate gives $\mathcal{I}_i(y)$ from $\mathcal{I}_i(x)$ and the same classifier witness. Pairing these facts gives $K(y)$.

For absorption, assume $K(x)$ and $C(r)$. The ring multiplication-closure row gives both $C(r \cdot_R x)$ and $C(x \cdot_R r)$. For each position i, left absorption in the ith ideal certificate gives $\mathcal{I}_i(r \cdot_R x)$ from $C(r)$ and $\mathcal{I}_i(x)$, and right absorption gives $\mathcal{I}_i(x \cdot_R r)$ from the same data. Hence the finite family holds at both absorbed endpoints. Together with the carrier witnesses, this gives $K(r \cdot_R x)$ and $K(x \cdot_R r)$.

These are exactly the carrier, additive sub-rng, multiplicative sub-rng, classifier-stability, and two-sided ambient absorption rows required of an $\mathsf{Ideal}^{\uparrow}$ predicate over the fixed $\mathsf{Ring}^{\uparrow}$ certificate. □

Theorem 81.45 (Finite ideal meet quotient-ring kernel scope)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let $xs : \mathsf{HistSpine}$ be a finite BEDC-history spine, and let $\mathcal{I}$ assign to every position $i : \mathsf{Pos}(xs)$ a predicate $\mathcal{I}_i : \mathsf{BHist} \to \mathsf{Prop}$ over the same ring data. Assume that every indexed predicate $\mathcal{I}_i$ is an* $\mathsf{Ideal}^{\uparrow}$ *predicate over these ring rows, and put*

$$K(x) \Longleftrightarrow \mathsf{FinIdealMeet}_{C,xs,\mathcal{I}}(x).$$

Let Q_K be the quotient-kernel source package of Definition 81.37. The package Q_K has the same diagonal, endpoint-transport, and multiplication-descent scope as the quotient-kernel packages built

from any other $\mathsf{Ideal}^{\uparrow}$ *predicate. Thus* $\mathsf{QuotientRing}^{\uparrow}$ *may consume finite intersections of ideals through the quotient-free kernel interface, with no finite-meet-specific quotient carrier or quotient primitive.*

Proof. By Definition 81.44, the predicate K is an $\mathsf{Ideal}^{\uparrow}$ predicate over the same $\mathsf{Ring}^{\uparrow}$ certificate as the finite family $\mathcal{I}$. This supplies the carrier support, sub-rng, classifier-stability, and two-sided ambient absorption rows required by the quotient-kernel source package.

Apply Definition 81.38 to K to obtain diagonal exactness. Apply Definition 81.39 to K to obtain endpoint transport along $\sim_R$ for carried endpoints.

Finally apply Definition 81.40 to K. The multiplication row is therefore the same product-difference membership row used by arbitrary $\mathsf{Ideal}^{\uparrow}$ predicates, so the quotient-ring consumer reads the finite meet only through $Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x -_R y)$. □

Definition 81.46 (Ring map ideal preimage predicate)**.** *Let S and T carry* $\mathsf{Ring}^{\uparrow}$ *certificates, with carriers C_S, C_T, classifiers $\sim_S, \sim_T$, zeros $0_S, 0_T$, additions $+_S, +_T$, additive inverses $-_S, -_T$, and multiplications $\cdot_S, \cdot_T$. Let J be an* $\mathsf{Ideal}^{\uparrow}$ *carried predicate over T in the sense of section 81.1. For a function $f : S \to T$, the source-carried ideal preimage predicate is*

$$\mathsf{Preim}^{\mathrm{ideal}}_{f,J}(x) \iff C_S(x) \wedge J(f(x)).$$

The first component records source-carrier evidence, and the second component records membership of the image endpoint in the target ideal.

Theorem 81.47 (Ring map ideal preimage IdealUp closure)**.** *Let S and T carry* $\mathsf{Ring}^{\uparrow}$ *certificates as in chapter 43, with notation as in Definition 81.46. Let $f : S \to T$ satisfy carrier preservation and classifier preservation on carried source endpoints:*

$$C_S(x) \Longrightarrow C_T(f(x))$$

and

$$C_S(x) \wedge C_S(y) \wedge x \sim_S y \Longrightarrow f(x) \sim_T f(y).$$

Assume also that f preserves zero, addition, additive inverse, and multiplication on carried source endpoints:

$$f(0_S) \sim_T 0_T,$$
$$C_S(x) \wedge C_S(y) \Longrightarrow f(x +_S y) \sim_T f(x) +_T f(y),$$
$$C_S(x) \Longrightarrow f(-_S x) \sim_T -_T f(x),$$

and

$$C_S(x) \wedge C_S(y) \Longrightarrow f(x \cdot_S y) \sim_T f(x) \cdot_T f(y).$$

Let J be an $\mathsf{Ideal}^{\uparrow}$ *predicate over T. Assume its ideal certificate supplies the rows specified in section 81.2: carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and left and right absorption by every carried target ring endpoint. For $K := \mathsf{Preim}^{\mathrm{ideal}}_{f,J}$ from Definition 81.46, K is an* $\mathsf{Ideal}^{\uparrow}$ *predicate over the source ring S. Explicitly, K has carrier support and the rows*

$$K(0_S), \qquad K(x) \wedge K(y) \Longrightarrow K(x +_S y), \qquad K(x) \Longrightarrow K(-_S x),$$
$$K(x) \wedge K(y) \Longrightarrow K(x \cdot_S y), \qquad K(x) \wedge x \sim_S y \Longrightarrow K(y),$$

and, for every r with $C_S(r)$,

$$K(x) \Longrightarrow K(r \cdot_S x) \wedge K(x \cdot_S r).$$

Lean4 ✓ BEDC.Derived.IdealUp.RingMapIdealPreimage closure rows

Proof. Unfold Definition 81.46. Carrier support is the first component of a K witness.

For zero membership, the source $\mathsf{Ring}^{\uparrow}$ certificate carries 0_S. The target ideal zero row gives $J(0_T)$. The zero-preservation row gives $f(0_S) \sim_T 0_T$, and target classifier symmetry from Definition 9.7 gives $0_T \sim_T f(0_S)$. Target ideal classifier transport therefore gives $J(f(0_S))$, so $K(0_S)$.

For additive closure, assume $K(x)$ and $K(y)$. The source carrier components give $C_S(x)$ and $C_S(y)$, hence the source ring certificate carries $x +_S y$. The target ideal additive row gives $J(f(x) +_T f(y))$ from $J(f(x))$ and $J(f(y))$. The addition-preservation row gives $f(x +_S y) \sim_T f(x) +_T f(y)$, and target classifier symmetry reverses this comparison. Applying target ideal classifier transport gives $J(f(x +_S y))$. Together with the source carrier witness this is $K(x +_S y)$.

For additive inverse closure, assume $K(x)$. Source inverse closure carries $-_S x$. The target ideal inverse row gives $J(-_T f(x))$. The inverse-preservation row gives $f(-_S x) \sim_T -_T f(x)$, whose symmetric form transports $J(-_T f(x))$ to $J(f(-_S x))$. Thus $K(-_S x)$.

For internal multiplication closure, assume $K(x)$ and $K(y)$. Source multiplication closure carries $x \cdot_S y$. The target ideal internal multiplication row gives $J(f(x) \cdot_T f(y))$. The multiplication-preservation row gives $f(x \cdot_S y) \sim_T f(x) \cdot_T f(y)$, and target classifier symmetry orients this comparison for ideal transport. Hence $J(f(x \cdot_S y))$, so $K(x \cdot_S y)$.

For source classifier transport, assume $K(x)$ and $x \sim_S y$. Source carrier transport from Definition 9.7 carries $C_S(x)$ along $x \sim_S y$ to obtain $C_S(y)$. Classifier preservation gives $f(x) \sim_T f(y)$. Target ideal classifier transport sends $J(f(x))$ along this comparison to $J(f(y))$. Therefore $K(y)$.

For left absorption, assume $K(x)$ and $C_S(r)$. Source multiplication closure carries $r \cdot_S x$, and carrier preservation gives $C_T(f(r))$. The target ideal left-absorption row gives $J(f(r) \cdot_T f(x))$ from $C_T(f(r))$ and $J(f(x))$. Multiplication preservation gives $f(r \cdot_S x) \sim_T f(r) \cdot_T f(x)$; after target classifier symmetry, ideal transport yields $J(f(r \cdot_S x))$. Thus $K(r \cdot_S x)$.

For right absorption, the same data carry $x \cdot_S r$ in the source, and the target ideal right-absorption row gives $J(f(x) \cdot_T f(r))$. Multiplication preservation for $x \cdot_S r$, followed by target classifier symmetry and ideal transport, gives $J(f(x \cdot_S r))$. Hence $K(x \cdot_S r)$.

The displayed carrier support, zero, additive, inverse, multiplicative, classifier-transport, and two-sided absorption rows are exactly the $\mathsf{Ideal}^{\uparrow}$ certificate rows listed in section 81.2 for a predicate over the source $\mathsf{Ring}^{\uparrow}$ certificate. □

Theorem 81.48 (Ring map ideal preimage quotient-kernel export surface)**.** *Let S and T carry $\mathsf{Ring}^{\uparrow}$ certificates as in chapter 43, with notation as in Definition 81.46. Let $f : S \to T$ satisfy the carrier-preservation, classifier-preservation, zero-preservation, addition-preservation, additive-inverse-preservation, and multiplication-preservation hypotheses of Definition 81.47. Let J be an $\mathsf{Ideal}^{\uparrow}$ predicate over T with the carrier support, zero membership, additive closure, additive inverse closure, internal multiplication closure, classifier transport, and two-sided target absorption rows listed in section 81.2. For $K := \mathsf{Preim}^{\mathrm{ideal}}_{f,J}$, let Q_K be the quotient-kernel source package of Definition 81.37. Then Q_K exports the five quotient-free rows consumed by $\mathsf{QuotientRing}^{\uparrow}$:*

1. *diagonal exactness for every carried source endpoint x:*

$$C_S(x) \Longrightarrow Q_K(x, x);$$

2. *endpoint transport along the source classifier:*

$$Q_K(x, y) \wedge x \sim_S x' \wedge y \sim_S y' \Longrightarrow Q_K(x', y');$$

3. additive representative descent:

$$Q_K(a, a') \wedge Q_K(b, b') \Longrightarrow Q_K(a +_S b, a' +_S b');$$

4. negation representative descent:

$$Q_K(a, a') \Longrightarrow Q_K(-_S a, -_S a');$$

5. multiplication representative descent:

$$Q_K(a, a') \wedge Q_K(b, b') \Longrightarrow Q_K(a \cdot_S b, a' \cdot_S b').$$

Thus a ring-map ideal preimage can be passed to the quotient-ring construction through the same quotient-kernel interface as any source-side $\mathsf{Ideal}^{\uparrow}$ *predicate.*

Proof. Apply Definition 81.47 to the displayed hypotheses. This proves that K is an $\mathsf{Ideal}^{\uparrow}$ predicate over the source $\mathsf{Ring}^{\uparrow}$ certificate on S.

Instantiate Definition 81.41 with the source ring S and the ideal predicate K. The quotient-kernel source package in that theorem is exactly the relation Q_K obtained from Definition 81.37.

Reading off the five displayed components gives diagonal exactness, endpoint transport, additive representative descent, negation representative descent, and multiplication representative descent for Q_K. No quotient carrier or preimage-specific quotient primitive is introduced; the consumer sees only the quotient-free kernel rows listed above. □

Theorem 81.49 (Finite preimage meet quotient-kernel export)**.** *Let S carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with carrier C_S, classifier $\sim_S$, addition $+_S$, additive inverse $-_S$, zero 0_S, and multiplication $\cdot_S$. Let $xs : \mathsf{HistSpine}$ be finite. For every $i : \mathsf{Pos}(xs)$, let T_i carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with target carrier C_i, classifier $\sim_i$, zero 0_i, addition $+_i$, additive inverse $-_i$, and multiplication $\cdot_i$; let $f_i : S \to T_i$ be a carried ring map in the sense of Definition 81.47; and let J_i be an* $\mathsf{Ideal}^{\uparrow}$ *predicate over T_i. Define the source preimage rows*

$$P_i(x) \Longleftrightarrow \mathsf{Preim}^{\mathrm{ideal}}_{f_i, J_i}(x)$$

and their finite source meet

$$K(x) \Longleftrightarrow \mathsf{FinIdealMeet}_{C_S, xs, P}(x).$$

Then K is an $\mathsf{Ideal}^{\uparrow}$ *predicate over S, and the predicate K supplies the quotient-free kernel surface consumed by* $\mathsf{QuotientRing}^{\uparrow}$ *through Definition 81.41.*

Proof. Fix a position $i : \mathsf{Pos}(xs)$. The hypotheses on f_i are exactly the carrier, classifier, zero, addition, inverse, and multiplication preservation rows required by Definition 81.47. Applying that theorem to f_i and J_i gives an $\mathsf{Ideal}^{\uparrow}$ certificate for the source predicate P_i over the fixed ring S.

The preceding construction is pointwise in i, so every row in the finite spine is now an $\mathsf{Ideal}^{\uparrow}$ predicate over the same source $\mathsf{Ring}^{\uparrow}$ certificate. Apply Definition 81.44 to the family P to obtain an $\mathsf{Ideal}^{\uparrow}$ certificate for K.

Finally instantiate Definition 81.41 with this resulting predicate K. The export theorem returns the diagonal, endpoint transport, additive descent, negation descent, and multiplication descent rows for the kernel relation generated by K, which is the quotient-free surface required before $\mathsf{QuotientRing}^{\uparrow}$ forms its quotient carrier. □

Theorem 81.50 (Ideal quotient-kernel absorption closure)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an* $\mathsf{Ideal}^{\uparrow}$ *carried predicate over R, and define*

$$x -_R y := x +_R (-_R y), \qquad Q_I(x, y) \iff C(x) \wedge C(y) \wedge I(x -_R y).$$

Assume the ring certificate supplies multiplication carrier closure, carrier transport, classifier symmetry, subtraction congruence, and the two absorption-difference rows

$$(r \cdot_R x) -_R (r \cdot_R y) \sim_R r \cdot_R (x -_R y)$$

and

$$(x \cdot_R r) -_R (y \cdot_R r) \sim_R (x -_R y) \cdot_R r.$$

Assume the ideal certificate supplies classifier transport and left and right absorption by every carried ring endpoint. Then

$$C(r) \wedge Q_I(x, y) \Longrightarrow Q_I(r \cdot_R x, r \cdot_R y) \wedge Q_I(x \cdot_R r, y \cdot_R r).$$

Proof. Unfold the displayed definition of Q_I. From $Q_I(x, y)$ obtain $C(x)$, $C(y)$, and $I(x -_R y)$. The multiplication carrier row applied to $C(r)$ with $C(x)$ and $C(y)$ gives the carried endpoints $C(r \cdot_R x)$, $C(r \cdot_R y)$, $C(x \cdot_R r)$, and $C(y \cdot_R r)$.

For the left-multiplied pair, ideal left absorption gives $I(r \cdot_R (x -_R y))$ from $C(r)$ and $I(x -_R y)$. The left absorption-difference row compares $(r \cdot_R x) -_R (r \cdot_R y)$ with $r \cdot_R (x -_R y)$. Classifier symmetry orients this comparison from the absorbed representative difference to the displayed product difference, and ideal classifier transport yields

$$I((r \cdot_R x) -_R (r \cdot_R y)).$$

For the right-multiplied pair, ideal right absorption gives $I((x -_R y) \cdot_R r)$. The right absorption-difference row compares $(x \cdot_R r) -_R (y \cdot_R r)$ with $(x -_R y) \cdot_R r$. Again classifier symmetry orients the comparison for the ideal transport row, giving

$$I((x \cdot_R r) -_R (y \cdot_R r)).$$

Pair the first transported membership with $C(r \cdot_R x)$ and $C(r \cdot_R y)$ to obtain $Q_I(r \cdot_R x, r \cdot_R y)$. Pair the second transported membership with $C(x \cdot_R r)$ and $C(y \cdot_R r)$ to obtain $Q_I(x \cdot_R r, y \cdot_R r)$. These two witnesses are exactly the claimed ambient left and right multiplication closure for the ideal quotient-kernel relation. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealQuotientKernel absorption closure`

Theorem 81.51 (Ideal finite meet, sum, and preimage lattice surface)**.** *Let R carry a* $\mathsf{Ring}^{\uparrow}$ *certificate over* BHist *with carrier C_R, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let $\mathcal{I}$ be a finite family of* $\mathsf{Ideal}^{\uparrow}$ *predicates over R, and let I and J be* $\mathsf{Ideal}^{\uparrow}$ *predicates over the same ambient ring. Let S be another* $\mathsf{Ring}^{\uparrow}$ *certificate over* BHist*, and let $f : S \to R$ satisfy the carrier, classifier, zero, addition, inverse, and multiplication preservation rows required in Definition 81.47. Then the predicates*

$$\mathsf{FinIdealMeet}_{C_R, xs, \mathcal{I}}, \qquad I +_R J, \qquad \mathsf{Preim}^{\mathrm{ideal}}_{f,J}$$

are $\mathsf{Ideal}^{\uparrow}$ *predicates over their ambient source rings: the finite meet and sum over* R*, and the preimage over* S*. For each resulting predicate* K *with ambient carrier* C*, classifier* $\sim$*, addition* $+$*, inverse* $-$*, zero* 0*, and multiplication* $\cdot$*, write* $x - y$ *for* $x + (-y)$ *and define the quotient-kernel package*

$$Q_K(x, y) \iff C(x) \wedge C(y) \wedge K(x - y)$$

Then Q_K *has the quotient-free surface consumed by* $\mathsf{QuotientRing}^{\uparrow}$*:*

$$C(x) \Longrightarrow Q_K(x, x),$$

$$Q_K(x, y) \wedge x \sim x' \wedge y \sim y' \Longrightarrow Q_K(x', y'),$$

and, for every carried ambient endpoint r*,*

$$C(r) \wedge Q_K(x, y) \Longrightarrow Q_K(r \cdot x, r \cdot y) \wedge Q_K(x \cdot r, y \cdot r).$$

Proof. For the finite meet branch, Definition 81.44 gives the $\mathsf{Ideal}^{\uparrow}$ rows for $\mathsf{FinIdealMeet}_{C_R, xs, \mathcal{I}}$. The quotient-kernel package for that predicate is then the same package used by arbitrary $\mathsf{Ideal}^{\uparrow}$ predicates, so its diagonal and endpoint-transport rows are the quotient-ring scope recorded in Definition 81.45.

For the sum branch, Definition 81.22 gives the $\mathsf{Ideal}^{\uparrow}$ rows for $I +_R J$. The quotient-ring consumer reads that predicate through the kernel package described in Definition 81.23, whose relation is still the displayed carrier-and-difference predicate.

For the preimage branch, Definition 81.47 gives the $\mathsf{Ideal}^{\uparrow}$ rows over the source ring S. Since the result is an ordinary $\mathsf{Ideal}^{\uparrow}$ predicate on the source carrier, the same quotient-kernel construction applies with the source carrier, classifier, subtraction, and multiplication rows.

Finally, Definition 81.50 supplies the common ambient absorption row for every branch after the corresponding $\mathsf{Ideal}^{\uparrow}$ predicate has been built. Thus the three lattice-facing constructors expose one uniform quotient-free package: diagonal, endpoint transport, and absorption, all expressed by $Q_K(x, y)$ and the ambient ring rows.

The ten-field fit is: traditional target ideal-lattice surface; BEDC source $\mathsf{Ring}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, and $\mathsf{QuotientRing}^{\uparrow}$ over BHist; carriers finite meet, sum decomposition, and preimage predicates; classifier the ambient ring classifier; stability the cited closure rows; ledger finite-position witnesses, additive decomposition witnesses, and image-membership witnesses; core theorem this block; exactness the displayed row package; standard bridge supplied by Definition 81.52; status paper target. □

81.3 Standard bridge

Theorem 81.52 (Ideal public surface standard bridge)**.** *Let* R *carry a* $\mathsf{Ring}^{\uparrow}$ *certificate with carrier* C*, classifier* $\sim_R$*, addition* $+_R$*, additive inverse* $-_R$*, zero* 0_R*, multiplication* $\cdot_R$*, and subtraction* $x -_R y := x +_R (-_R y)$*. Assume the ambient* $\mathsf{Ring}^{\uparrow}$ *certificate has a standard bridge to an ordinary ring surface. Then the public* $\mathsf{Ideal}^{\uparrow}$ *surface interprets over that standard ring as the ordinary ideal-predicate surface and re-encodes through the BEDC rows: carried predicates, zero membership, additive closure, additive inverse closure, internal multiplication, two-sided ambient absorption, finite meet, sum, ring-map preimage, principal generated ideals, and the quotient-kernel export surface. The bridge exports no host quotient carrier, host quotient primitive, or classifier-changing host equality to BEDC.*

Lean4 ✓ `BEDC.Derived.IdealUp.IdealUp StdBridge`

Proof. Unpack the $\mathsf{Ideal}^{\uparrow}$ carrier inventory of Definition 81.1. The ordinary surface reads the BEDC predicate as a carried subset of the standard ring supplied by the ambient $\mathsf{Ring}^{\uparrow}$ bridge. Carrier support, zero membership, addition, additive inverse, internal multiplication, classifier transport, and left and right absorption are exactly the rows displayed in section 81.2.

The finite-meet constructor is interpreted pointwise. Apply Definition 81.44: every indexed ordinary ideal row is carried back to the finite BEDC family, and the carrier conjunct supplies the empty-spine case. The sum constructor is interpreted by the additive decomposition ledger of Definition 81.22; its least-upper-bound surface is the ordinary inclusion property recorded in the same public BEDC rows.

The preimage constructor uses the ambient bridges on the source and target $\mathsf{Ring}^{\uparrow}$ certificates. Apply Definition 81.47: carrier preservation, classifier preservation, and operation preservation turn the standard inverse-image ideal into the BEDC source predicate. Principal generated ideals are supplied by Definition 81.33, whose generator ledger is again expressed only by carried endpoints, classifier transport, and ideal rows.

For the quotient-kernel part, unfold Definition 81.37. Its comparison data are precisely the two carrier witnesses $C(x)$ and $C(y)$ and membership of the displayed difference in the ideal predicate. Apply Definition 81.41; the exported diagonal, endpoint-transport, additive-descent, negation-descent, and multiplication-descent rows are the quotient-free standard comparison boundary.

Thus each public constructor is interpreted through the ambient standard ring bridge and then repacked by the BEDC certificate rows already cited in this chapter. Since the quotient-kernel surface exports only carried endpoints, classifier transport, and difference-membership rows, no host quotient carrier or host quotient primitive enters the BEDC interface. □

Lean4 ✓ `BEDC.Derived.IdealUp.IdealUp concrete to schema`

Closure status for $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$.

Constructive story: An ideal is carried as a predicate on the histories of an ambient $\mathsf{Ring}^{\uparrow}$ certificate, together with the rows saying that its members are ring-carried, contain zero, close under addition, additive inverse, internal multiplication, classifier transport, and multiplication by every carried ambient endpoint. The carrier accepts exactly predicates with those rows, while the quotient-kernel surface accepts pairs x, y only when both endpoints are carried and the displayed difference $x -_R y$ lies in the ideal; sameness is transported by the ambient ring classifier rather than by forming a quotient carrier. The source is the ambient ring certificate plus the carried ideal predicate, the pattern records finite meet, sum by additive decomposition, ring-map preimage, principal generated ideals, and the quotient-kernel diagonal and descent rows, and the ledger keeps membership witnesses, decomposition witnesses, image-membership witnesses, and difference-membership rows visible. The name-certificate obligations are supplied by zero membership for inhabitation, ambient classifier reflexivity, symmetry, and transitivity for endpoint comparison, and the ideal classifier-transport row for closure of membership under classified replacement. A pair history h lies in the signature gap of an $\mathsf{Ideal}^{\uparrow}$ token p when the source-domain accepts the ambient ring endpoints and ideal predicate, the signature relation pairs the pair with an answer quotient-kernel package and evidence ledger, and token introduction names that answer as p; unfolded, the endpoint difference is a displayed ideal member and all additive, negation, and multiplication descent rows are those encoded by p. The standard bridge reads the BEDC predicate as an ordinary ideal predicate over the standard ring supplied by the ambient $\mathsf{Ring}^{\uparrow}$ bridge and reads it back through the same row inventory, while explicitly exporting no host quotient carrier. The chapter registers this carried-predicate and quotient-kernel construction under the name $\mathsf{Ideal}^{\uparrow}$; ideals remain predicates with transport and absorption rows over existing ring histories, with no host quotient primitive or classifier-changing equality introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature IdealUp boundary covers carried predicates, finite meet, sum, preimage, principal ideals, the quotient-kernel export surface, and the paper-level standard bridge of Definition 81.52.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.IdealUp.IdealQuotientKernel\_export\_surface`

Bridge status: bridgeChecked

Not claimed: This chapter does not close host quotient carriers, full standard ideal theory, or host quotient primitives.

Upgrade path: No further upgrade – chapter at top of formal axis.

The $\mathsf{Ideal}^{\uparrow}$ packet consumes the $\mathsf{Ring}^{\uparrow}$ carrier, classifier, and operation rows of chapter 43. It supplies the kernel row used by chapter 83, so the quotient-ring route reads the ideal certificate as a ring-indexed kernel surface rather than as an isolated naming packet.

Theorem 81.53 (Ideal ring carrier dependency)**.** *Any* $\mathsf{Ideal}^{\uparrow}$ *absorption read must expose the* $\mathsf{Ring}^{\uparrow}$ *multiplication surface of chapter 43 before it consumes ambient products* $r \cdot_R x$ *or* $x \cdot_R r$*. When the same read is used as a kernel or submodule-like row over a fixed scalar carrier, it must also expose the* $\mathsf{Module}^{\uparrow}$ *scalar-action surface of chapter 46. The dependency is a carrier route for absorption data, not a new algebraic theorem about rings or modules.*

Proof. Project the $\mathsf{Ideal}^{\uparrow}$ carrier along its absorption rows. The only available ambient products are read through the displayed $\mathsf{Ring}^{\uparrow}$ multiplication surface, so the absorption consumer must first expose that sibling route.

For a kernel or submodule-like consumption, project the scalar-action row in the same carrier reading. The $\mathsf{Module}^{\uparrow}$ surface records the scalar action over the fixed scalar carrier, and the ideal read merely reuses that route to name the absorption pattern. No additional ring or module property is asserted. □

Chapter 82

A Concrete Naming Certificate for $\mathsf{QuotientGroup}^{\uparrow}$

The quotient-group interface $\mathsf{QuotientGroup}^{\uparrow}$ packages the carrier of cosets of a normal subgroup of a group. It depends on $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ (chapter 80) plus the normality witness, and produces a psame-classifier on cosets without invoking host-level `Quot`-soundness.

82.1 Carrier definition

Definition 82.1 (QuotientGroup singleton carrier)**.** *For the concrete singleton quotient-group object over histories, the carrier is the empty-history class*

$$Q_{\mathsf{e}}(h) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Lean4 (definition) `BEDC.Derived.QuotientGroupUp.QuotientGroupSingletonCarrier`

Definition 82.2 (QuotientGroup singleton classifier)**.** *The concrete singleton quotient-group classifier is*

$$\rho_Q(h, k) \iff Q_{\mathsf{e}}(h) \wedge Q_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

It records both endpoint carrier witnesses together with their BEDC history-sameness witness.

Lean4 (definition) `BEDC.Derived.QuotientGroupUp.QuotientGroupSingletonClassifier`

Definition 82.3 (Centralizer coset carrier)**.** *For a BEDC multiplication operation* mul*, parameter a, and representative* repr*, the centralizer-coset carrier contains a history h exactly when* repr *centralizes a and h*hsamerepr*.*

Lean4 (definition) `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier`

Definition 82.4 (Centralizer coset classifier)**.** *The centralizer-coset classifier relates two carried histories when both lie in the same representative carrier and are internally history-same.*

Lean4 (definition) `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier`

Definition 82.5 (Centralizer-normalizer QuotientGroupUp carrier)**.** *In the BEDC history-level* $\mathsf{Group}^{\uparrow}$ *setting of Definition 80.27, let Z_a be the centralizer carrier of Definition 41.84 and let N_a be the centralizer normalizer carrier of Definition 41.95. The centralizer-normalizer* $\mathsf{QuotientGroup}^{\uparrow}$

representative carrier is N_a, with classifier $Q_{Z_a \triangleleft N_a}$ from Definition 80.33. Represented multiplication, unit, and inversion are given by $\mathsf{mul}(x, y)$, e, *and* $\mathsf{inv}(x)$.

The certificate ledger records the representative endpoints in N_a, the relative kernel witness $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$, and the tower rows for the lower centralizer subgroup, the upper normalizer subgroup, inclusion $Z_a \subseteq N_a$, and conjugation preservation of Z_a by carried elements of N_a. Thus the carrier is a concrete history-level representative system equipped with a classifier, not a host quotient type.

Theorem 82.6 (Centralizer-normalizer empty representative)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *whose represented unit is* e, *assume the unit, multiplication-congruence, and right-inverse laws used in Definition 82.5. Then the representative carrier N_a contains* e.

Proof. Unfold N_a as the centralizer normalizer carrier. The right-inverse law at e, together with the left unit law, identifies $\mathsf{inv}(\mathsf{e})$ with e.

For the forward conjugation clause, multiplication congruence and the unit laws transport $\mathsf{mul}(\mathsf{mul}(\mathsf{e}, x), \mathsf{inv}(\mathsf{e}))$ to x. The centralizer row for x can therefore be read as the required centralizer row for the conjugated word.

The inverse conjugation clause is the same calculation applied to $\mathsf{inv}(\mathsf{inv}(\mathsf{e}))$. The unit and right-inverse laws identify the outer inverse word with e, so the second normalizer clause reduces to the same centralizer transport. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer empty unit`

Theorem 82.7 (Centralizer-normalizer empty representative transport)**.** *Under the same BEDC history-level group laws, plus inverse congruence, every history t with* $\mathsf{ehsame}t$ *also lies in the representative carrier N_a.*

Proof. Apply Definition 82.6 to get $N_a(\mathsf{e})$.

For the forward normalizer clause at t, multiplication congruence transports $\mathsf{mul}(\mathsf{mul}(t, x), \mathsf{inv}(t))$ back to $\mathsf{mul}(\mathsf{mul}(\mathsf{e}, x), \mathsf{inv}(\mathsf{e}))$, using inverse congruence on the endpoint comparison $\mathsf{ehsame}t$. The empty representative row then supplies the required carried centralizer witness.

The inverse clause uses inverse congruence once more to compare $\mathsf{inv}(t)$ with $\mathsf{inv}(\mathsf{e})$, and again on the outer inverse. Multiplication congruence transports the whole inverse-conjugation word back to the empty representative word, where the previous theorem supplies the row. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer empty hsame transport unit`

Theorem 82.8 (Centralizer normalizer orbit-kernel quotient equivalence)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the hypotheses of Definition 80.32 and Definition 80.47, and for x, y :* BHist *with $N_a(x)$ and $N_a(y)$, the two representative classifiers on the carrier N_a agree:*

$$Q_{Z_a \leq N_a}(x, y) \Longleftrightarrow Q_{Z_a \triangleleft N_a}(x, y).$$

Proof. For the forward direction, expand Definition 80.31. The witness is a history z with $Z_a(z)$ and $\mathsf{hsame}(y, \mathsf{mul}(x, z))$, together with the endpoint rows $N_a(x)$ and $N_a(y)$.

Multiply the endpoint comparison on the left by $\mathsf{inv}(x)$. By associativity, the two-sided inverse law, the unit laws, and multiplication congruence, the word $\mathsf{mul}(\mathsf{inv}(x), y)$ is transported by hsame to z. The hsame transport row for the centralizer carrier therefore turns $Z_a(z)$ into $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$, which is exactly the kernel clause of $Q_{Z_a \triangleleft N_a}(x, y)$.

For the reverse direction, expand Definition 80.33. Let

$$w = \mathsf{mul}(\mathsf{inv}(x), y).$$

The kernel clause gives $Z_a(w)$, and the endpoint clauses give the same representative carrier rows $N_a(x)$ and $N_a(y)$.

Associativity, the left inverse law, the left unit law, and multiplication congruence rewrite $\mathsf{mul}(x, w)$ to y up to hsame. Thus w is the centralizer witness required by Definition 80.31, so $Q_{Z_a \leq N_a}(x, y)$ follows. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer orbit kernel equivalence iff`

Per-object template: traditional target quotient group by a normal subgroup; BEDC source $\mathsf{Group}^{\uparrow}$ plus the centralizer-normalizer $\mathsf{Subgroup}^{\uparrow}$ tower; carrier N_a; classifier either $Q_{Z_a \leq N_a}$ or $Q_{Z_a \triangleleft N_a}$; stability Definition 80.32, Definition 80.47, and Definition 82.8; ledger representative pair plus coset or kernel witness; core theorem list Definition 80.31, Definition 80.33, and Definition 80.27; exactness the displayed iff; standard bridge ordinary coset/kernel criterion; status ProofTarget.

Theorem 82.9 (Centralizer-normalizer QuotientGroupUp identity fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the hypotheses of Definition 82.8, for every* $x : \mathsf{BHist}$ *the left identity fibers of the two representative quotient classifiers satisfy*

$$Q_{Z_a \leq N_a}(\mathsf{e}, x) \iff Q_{Z_a \triangleleft N_a}(\mathsf{e}, x) \iff Z_a(x),$$

and the right identity fibers satisfy

$$Q_{Z_a \leq N_a}(x, \mathsf{e}) \iff Q_{Z_a \triangleleft N_a}(x, \mathsf{e}) \iff Z_a(x).$$

Proof. For the left fiber, first compare the coset classifier with the relative-kernel classifier at (e, x). Apply Definition 82.8. The endpoint rows needed by that theorem are either fields of the classifier edge being transported or are supplied by the kernel relation itself.

Next identify the kernel classifier fiber. Apply Definition 80.35 and, at the Lean level, its empty-fiber form `BEDC.Derived.SubgroupUp.SubgroupCentralizerQuotientKernel_empty_fiber_iff`. This turns $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$ into $Z_a(x)$. The group-level normalization behind the identity endpoint is `BEDC.Derived.GroupUp.group_inverse_identity`, which identifies the inverse of the represented identity with the identity before the relative kernel word is read.

For the right fiber, repeat the same transport at (x, e). The orbit-kernel equivalence passes between $Q_{Z_a \leq N_a}(x, \mathsf{e})$ and $Q_{Z_a \triangleleft N_a}(x, \mathsf{e})$, and the empty-right-fiber form `BEDC.Derived.SubgroupUp.SubgroupCentra` identifies the kernel classifier fiber with $Z_a(x)$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer identity fiber iff`

Per-object template: traditional target quotient projection kernel; BEDC source $\mathsf{Group}^{\uparrow}$ + $\mathsf{Subgroup}^{\uparrow}$ + $\mathsf{QuotientGroup}^{\uparrow}$ over BHist; ambient sort BHist; carrier N_a; classifiers $Q_{Z_a \leq N_a}$ and $Q_{Z_a \triangleleft N_a}$; stability orbit-kernel equivalence plus identity-fiber transport; ledger identity representative and relative-kernel word; endpoint normalization by group inverse identity; exactness identity fiber equals centralizer; status ProofTarget.

Theorem 82.10 (Centralizer coset representative context classifier)**.** *If* e *and* repr *centralize* a *and* $\mathsf{reprhsamee}$*, then for histories* $l, h, r, k : \mathsf{BHist}$ *the representative centralizer-coset classifier of a three-part append context is equivalent to the four component representative carriers:*

$$\mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{repr}, \mathsf{append}(l, \mathsf{append}(h, r)), k) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{repr}, l) \wedge \mathsf{Centraliz}$$

The ten-field fit is: traditional target representative-level decomposition of a centralizer coset classifier across a three-part append context; BEDC source $\mathsf{Subgroup}^{\uparrow}$*,* $\mathsf{QuotientGroup}^{\uparrow}$*,* append*, and* hsame*; carrier centralizer-coset histories; classifier* $\mathsf{CentralizerCosetClassifier}$*; stability from empty-representative classifier transport, empty-representative context splitting, and carrier transport along* $\mathsf{reprhsamee}$*; ledger the empty representative bridge plus the four endpoint carrier witnesses; core theorem this block; exactness the displayed iff; downstream consumer nested quotient-coset context normalization; status* $\mathsf{ProofTarget}$*.*

Proof. For the forward direction, apply Definition 82.16 to transport the representative classifier to the empty representative. Unfold Definition 82.4. The left endpoint is the empty-representative carrier of $\mathsf{append}(l, \mathsf{append}(h, r))$, and the right endpoint is the empty-representative carrier of k.

Apply Definition 82.21 to split the left endpoint into empty-representative carriers for l, h, and r. Then apply Definition 82.18 to transport those three carriers, and the right endpoint carrier for k, back to the representative carrier.

Conversely, transport the four representative carriers to empty-representative carriers with Definition 82.18. The reverse direction of Definition 82.21 rebuilds the empty-representative carrier of the nested append endpoint. Together with the empty-representative carrier of k, the unfolded classifier is obtained by composing both endpoint comparisons through e. Transporting back with Definition 82.16 gives the representative classifier. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier representative context classifier iff`

82.2 The certificate

Theorem 82.11 (QuotientGroup singleton semantic naming certificate)**.** *The singleton quotient-group carrier and classifier form a semantic naming certificate whose source, pattern, and ledger predicates are all* Q_{e}*.*

Proof. The inhabited endpoint is e, witnessed by reflexive $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Reflexivity of the classifier packages the source carrier twice with reflexive $\mathsf{hsame}(h, h)$. Symmetry swaps the endpoint carriers and reverses the hsame component. Transitivity composes the two hsame components and keeps the outer endpoint carrier witnesses. Carrier transport along a classifier edge is the right endpoint carrier field. Since the semantic source, pattern, and ledger predicates are the same singleton predicate, their soundness clauses are direct projections of the source witness. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupSingleton semanticNameCert`

Theorem 82.12 (QuotientGroup singleton normalizer orbit classifier collapse)**.** *For histories* $x, y : \mathsf{BHist}$*, in the concrete singleton history group instance of Definition 41.3 and the singleton* $\mathsf{QuotientGroup}^{\uparrow}$ *classifier of Definition 82.2,*

$$x \sim_{N_{\mathsf{e}}} y \iff \rho_Q(x, y).$$

The ten-field fit is: traditional target the quotient of the terminal group by its terminal normalizer action; BEDC source concrete singleton $\mathsf{Group}^{\uparrow}$ *plus singleton* $\mathsf{QuotientGroup}^{\uparrow}$*; carrier* $Q_{\mathsf{e}} = A_{\mathsf{e}}$ *by Definitions 36.169, 41.3, and 82.1; classifier* ρ_Q *compared with* $\sim_{N_{\mathsf{e}}}$*; stability from Definition 41.151, Definition 82.11, and symmetry and transitivity for* hsame*; ledger the hidden conjugator endpoint; core theorem this block; exactness the displayed iff; downstream consumer first-isomorphism-style quotient-kernel exactness; status* $\mathsf{ProofTarget}$*.*

Proof. Apply Definition 41.151. It identifies the terminal normalizer orbit relation with $A_{\mathsf{e}}(x) \wedge A_{\mathsf{e}}(y)$. By Definitions 36.169, 41.3, and 82.1, this is the same endpoint data as $Q_{\mathsf{e}}(x) \wedge Q_{\mathsf{e}}(y)$.

For the forward direction, the orbit witness therefore supplies $Q_{\mathsf{e}}(x)$ and $Q_{\mathsf{e}}(y)$. Unfold Definition 82.2. The two carrier fields remain, and the final $\mathsf{hsame}(x, y)$ component is obtained by composing the singleton identifications $x\mathsf{hsamee}$ and $y\mathsf{hsamee}$, using symmetry on the second and then transitivity.

Conversely, a $\rho_Q(x, y)$ witness contains the two endpoint carrier fields. Under the same carrier identification these are $A_{\mathsf{e}}(x)$ and $A_{\mathsf{e}}(y)$, so the reverse direction of Definition 41.151 yields $x \sim_{N_{\mathsf{e}}} y$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupSingletonClassifier normalizer orbit collapse iff`

Theorem 82.13 (QuotientGroup singleton continuation classifier)**.** *If* $\mathsf{Cont}(p, q, r)$*, then the singleton classifier at the continuation result reads back exactly as singleton carriers on the two inputs and the right endpoint:*

$$\mathsf{QuotientGroupSingletonClassifier}(r, h) \iff \mathsf{QuotientGroupSingletonCarrier}(p) \wedge \mathsf{QuotientGroupSingletonCarrier}(q) \wedge \mathsf{Qu}$$

Proof. Unfold the singleton carrier and classifier. A classifier witness gives $r\mathsf{hsamee}$, and empty-append inversion along $\mathsf{Cont}(p, q, r)$ splits this into the two input singleton carriers. Conversely, the two input carriers rebuild $r\mathsf{hsamee}$, and the endpoint carrier for h supplies the remaining classifier fields. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupSingletonClassifier continuation classifier iff`

Theorem 82.14 (QuotientGroup singleton classifier visible endpoint exclusion)**.** *For histories* $p, q, k : \mathsf{BHist}$*, no classifier edge can have a visible* $\mathsf{e0}$ *or* $\mathsf{e1}$ *endpoint on either side:*

$$\rho_Q(\mathsf{e0}(p), k), \quad \rho_Q(\mathsf{e1}(p), k), \quad \rho_Q(k, \mathsf{e0}(q)), \quad \rho_Q(k, \mathsf{e1}(q))$$

are all impossible.

Proof. Unfold the classifier. A visible left endpoint gives its left carrier field, which identifies $\mathsf{e0}(p)$ or $\mathsf{e1}(p)$ with e. A visible right endpoint gives the same contradiction from the right carrier field. Each case is excluded by constructor separation for generated BEDC histories. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupSingletonClassifier visible endpoint absurd`

Theorem 82.15 (Centralizer coset SemanticNameCert)**.** *If the representative* repr *centralizes* a*, then the centralizer-coset carrier, used as source, pattern, and ledger policy, with the centralizer-coset classifier forms a* $\mathsf{SemanticNameCert}$*.*

Proof. The representative itself inhabits the carrier. Reflexivity, symmetry, and transitivity of the classifier are inherited from hsame, while carrier respect follows from the carried right endpoint stored in the classifier. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCoset semanticNameCert`

Theorem 82.16 (Centralizer coset empty representative classifier transport)**.** *If the representative* repr *and the empty history both centralize a, and* reprhsamee*, then the centralizer-coset classifier for* repr *agrees with the centralizer-coset classifier for* e*:*

$$\mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{repr}, h, k) \iff \mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{e}, h, k).$$

Proof. Unfold the two coset classifiers. Each endpoint carrier consists of the chosen representative's centralizer witness and an endpoint comparison with that representative. The comparison reprhsamee converts the two endpoint comparisons in either direction, while the corresponding centralizer witness supplies the carrier field for the chosen representative. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier empty representative transport iff`

Theorem 82.17 (Centralizer coset empty representative classifier hsame endpoint transport)**.** *For histories h, k, h', k' : BHist, if the empty-representative centralizer-coset classifier relates h and k, and $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, then*

$$\mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{e}, h', k') \wedge h'\mathsf{hsamee} \wedge k'\mathsf{hsamee}.$$

Proof. The empty-representative classifier stores the empty centralizer witness together with $h$hsamee, $k$hsamee, and $h$hsame$k$. Compose the two endpoint comparisons with the symmetries of $h$hsameh' and $k$hsamek' to obtain the empty endpoints for h' and k'. Reuse the stored centralizer witness at both endpoints and compose the two empty endpoints to repack the classifier on (h', k'). □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier empty hsame endpoint transport`

Theorem 82.18 (Centralizer coset carrier empty representative transport)**.** *If the representative* repr *and the empty history both centralize a, and* reprhsamee*, then the centralizer-coset carrier for* repr *agrees with the centralizer-coset carrier for* e*:*

$$\mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{repr}, h) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, h).$$

Proof. Unfold the two carrier predicates. A carrier witness for repr contains the centralizer witness for repr and the endpoint comparison $h$hsamerepr; composing with reprhsamee gives the empty-representative endpoint comparison. The reverse direction composes with symmetry of the same comparison and uses the centralizer witness for repr. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier empty representative transport iff`

Theorem 82.19 (Centralizer coset empty representative visible absurdity)**.** *For any multiplication operation* mul *and histories a, z, no visible one-step history is carried by the centralizer-coset carrier over the empty representative:*

$$\mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, \mathsf{e0}(z)) \Rightarrow \bot, \qquad \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, \mathsf{e1}(z)) \Rightarrow \bot.$$

Proof. Unfold the carrier only to read its endpoint comparison. A carried visible endpoint would give $\mathsf{e0}(z)\mathsf{hsamee}$ or $\mathsf{e1}(z)\mathsf{hsamee}$, respectively. Both are excluded by constructor separation for generated BEDC histories. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier empty representative visible absurd`

Theorem 82.20 (Centralizer coset empty continuation carrier pair)**.** *If* e *centralizes* a *and* $\mathsf{Cont}(p, q, r)$*, then the empty-representative centralizer-coset carrier of the continuation result is equivalent to the two endpoint carriers:*

$$\mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, r) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, p) \wedge \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, q).$$

Proof. Forwards, the result carrier gives $r\mathsf{hsamee}$. Transport the displayed continuation to an empty result and apply empty-result inversion, giving $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$; combine each with the assumed centralizer witness for e.

Conversely, the two endpoint carriers give $p\mathsf{hsamee}$ and $q\mathsf{hsamee}$. The displayed continuation is therefore determined by the empty append result, so $r\mathsf{hsamee}$, and the empty centralizer witness closes the result carrier. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier empty continuation pair iff`

Theorem 82.21 (Centralizer coset empty representative context)**.** *If* e *centralizes* a*, then the empty-representative centralizer-coset carrier of a three-part append context is equivalent to the three component carriers:*

$$\mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, \mathsf{append}(l, \mathsf{append}(h, r))) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, l) \wedge \mathsf{CentralizerCosetCa}$$

Proof. Unfold the empty-representative carrier to the endpoint comparison with e. Empty append inversion first splits the outer append into l and the inner append, then splits the inner append into h and r. Conversely, the three empty endpoint comparisons rebuild the nested empty append comparison, and the assumed centralizer witness supplies the carrier field. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier empty representative context iff`

Theorem 82.22 (Centralizer coset empty continuation classifier)**.** *If* e *centralizes* a *and* $\mathsf{Cont}(p, q, r)$*, then*

$$\mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{e}, r, h) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, p) \wedge \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{e}, q) /$$

Proof. Unfold the empty-representative coset classifier. A classified continuation result gives $r\mathsf{hsamee}$; reading the continuation as an append endpoint and inverting empty append splits this into the two carrier comparisons for p and q, while the right classifier carrier supplies the comparison for h. Conversely, the two carrier comparisons for p and q rebuild an empty append endpoint for r, and the carrier comparison for h closes the classifier. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier empty continuation classifier iff`

Theorem 82.23 (Centralizer coset representative continuation classifier)**.** *If* e *and* repr *centralize* a*,* $\mathsf{reprhsamee}$*, and* $\mathsf{Cont}(p, q, r)$*, then*

$$\mathsf{CentralizerCosetClassifier}(\mathsf{mul}, a, \mathsf{repr}, r, h) \iff \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{repr}, p) \wedge \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a,$$

Proof. First transport the representative classifier to the empty representative. The empty-representative continuation classifier splits the result into the two continuation inputs and the endpoint h. The comparison reprhsamee transports each carrier endpoint back to repr, with the representative centralizer witness supplying the carrier's centralizer field. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetClassifier representative continuation classifier iff`

Theorem 82.24 (Centralizer-normalizer QuotientGroupUp semantic certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the associativity, unit, multiplication-congruence, inverse-congruence, inverse-product reversal, and two-sided inverse laws used by Definition 80.27, the tuple of Definition 82.5 satisfies the* $\mathsf{QuotientGroup}^{\uparrow}$ *certificate fields:*

1. *the representative carrier N_a is inhabited by e;*
2. *$Q_{Z_a \triangleleft N_a}$ is reflexive, symmetric, and transitive on carried representatives;*
3. *represented multiplication descends through $Q_{Z_a \triangleleft N_a}$;*
4. *e and $\mathsf{inv}(x)$ are carried by N_a whenever x is carried by N_a;*
5. *the represented product, unit, and inverse operations respect the same classifier on all carried representatives.*

Consequently the centralizer-normalizer tower supplies a quotient-free semantic certificate for $\mathsf{QuotientGroup}^{\uparrow}$ *whose observable names are histories in N_a and whose equality test is the relative-kernel predicate $Q_{Z_a \triangleleft N_a}$.*

Proof. Apply Definition 80.27. Its lower and upper subgroup rows give $Z_a(e)$, $N_a(e)$, product and inverse closure for N_a, the inclusion $Z_a \subseteq N_a$, and the normality rows carrying Z_a through conjugation by elements of N_a.

Next apply Definition 80.47 to the classifier $Q_{Z_a \triangleleft N_a}$. The first three rows give reflexivity, symmetry, and transitivity on representatives in N_a. The fourth row gives product descent for mul and the quotient operation congruence required by the represented multiplication.

It remains to identify the unit and inverse operation rows. The unit row is the upper subgroup inhabitance supplied by the tower. For inverse closure, use the upper subgroup inverse row from Definition 80.27; inverse descent follows by combining the ambient inverse-congruence and two-sided inverse laws named there with Definition 41.65. Expanding Definition 80.33, the kernel witness for inverse representatives is transported from $Z_a(\mathsf{mul}(\mathsf{inv}(x), y))$ to the corresponding relative inverse word by the Z_a inverse-closure and hsame-transport rows.

These rows are exactly the carrier inhabitance, equivalence, product descent, unit closure, inverse closure, and operation congruence fields demanded by the history-level $\mathsf{QuotientGroup}^{\uparrow}$ certificate, with the ledger witness kept as the relative kernel membership rather than as an element of a host quotient. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer semanticNameCert`

Theorem 82.25 (Centralizer-normalizer quotient identity fiber subgroup package)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same hypotheses as Definition 82.24, define the quotient identity fiber carrier by*

$$K_Q(x) :\Longleftrightarrow Q_{Z_a \triangleleft N_a}(\mathsf{e}, x).$$

Then K_Q is the concrete subgroup package carried by the identity fiber of the centralizer-normalizer quotient classifier:

1. $K_Q(\mathsf{e})$.
2. *If* $K_Q(x)$ *and* $K_Q(y)$*, then* $K_Q(\mathsf{mul}(x, y))$.
3. *If* $K_Q(x)$*, then* $K_Q(\mathsf{inv}(x))$.
4. *If* $K_Q(x)$ *and* $x\mathsf{hsame}y$*, then* $K_Q(y)$.
5. *For every history* $x : \mathsf{BHist}$*,*
$$K_Q(x) \iff Z_a(x).$$

Proof. First apply Definition 80.35. For each history x, it gives the identity-fiber equivalence for $Q_{Z_a \triangleleft N_a}$. After unfolding the displayed definition of K_Q, this is exactly the equivalence in the final row.

The identity row follows by transporting $Z_a(\mathsf{e})$ from Definition 41.87 through the reverse direction of the fiber equivalence at e. This gives $Q_{Z_a \triangleleft N_a}(\mathsf{e}, \mathsf{e})$, which is exactly $K_Q(\mathsf{e})$.

For product closure, assume $K_Q(x)$ and $K_Q(y)$. The forward directions of the fiber equivalence give $Z_a(x)$ and $Z_a(y)$. The product row of Definition 41.87 gives $Z_a(\mathsf{mul}(x, y))$, and the reverse direction of the fiber equivalence returns $K_Q(\mathsf{mul}(x, y))$.

For inverse closure, assume $K_Q(x)$. Convert it to $Z_a(x)$ by the forward direction of the fiber equivalence. The inverse row of Definition 41.87 gives $Z_a(\mathsf{inv}(x))$, and the reverse direction of the fiber equivalence gives $K_Q(\mathsf{inv}(x))$.

For carrier transport, assume $K_Q(x)$ and $x\mathsf{hsame}y$. Convert $K_Q(x)$ to $Z_a(x)$, use the hsame-transport row for the centralizer carrier from Definition 41.87 to obtain $Z_a(y)$, and convert back to $K_Q(y)$. Thus the classifier inherited from $Q_{Z_a \triangleleft N_a}$ has the same concrete identity fiber as the centralizer carrier. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer identity fiber subgroup package`

Theorem 82.26 (Centralizer-normalizer quotient identity fiber classifier saturation)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same hypotheses as Definition 82.24, define the quotient identity fiber carrier by*

$$K_Q(t) :\iff Q_{Z_a \triangleleft N_a}(\mathsf{e}, t).$$

For histories $x, y : \mathsf{BHist}$*, the identity fiber is saturated by the quotient classifier:*

1. *If* $K_Q(x)$ *and* $Q_{Z_a \triangleleft N_a}(x, y)$*, then* $K_Q(y)$.
2. *If* $K_Q(y)$ *and* $Q_{Z_a \triangleleft N_a}(x, y)$*, then* $K_Q(x)$.

Per-object template: traditional target saturated identity fiber of a quotient classifier; BEDC source $\mathsf{Group}^{\uparrow}$*,* $\mathsf{Subgroup}^{\uparrow}$*, and* $\mathsf{QuotientGroup}^{\uparrow}$ *over* BHist*; carrier* K_Q*; classifier* $Q_{Z_a \triangleleft N_a}$*; ledger relative-kernel witness; core theorem this block; exactness the two saturation implications; downstream consumer first-isomorphism-style quotient-kernel exactness; status* $\mathsf{ProofTarget}$.

Proof. Read the semantic certificate of Definition 82.24 for the symmetry and transitivity rows of $Q_{Z_a \triangleleft N_a}$ on carried representatives.

For the first implication, unfold K_Q. The assumption $K_Q(x)$ is the classifier edge $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$. Compose it with $Q_{Z_a \triangleleft N_a}(x, y)$ by transitivity to obtain $Q_{Z_a \triangleleft N_a}(\mathsf{e}, y)$, which is exactly $K_Q(y)$.

For the second implication, first use symmetry on $Q_{Z_a \triangleleft N_a}(x, y)$ to obtain $Q_{Z_a \triangleleft N_a}(y, x)$. Unfold $K_Q(y)$ as $Q_{Z_a \triangleleft N_a}(\mathsf{e}, y)$, then compose these two classifier edges by transitivity to obtain $Q_{Z_a \triangleleft N_a}(\mathsf{e}, x)$. Folding the displayed definition gives $K_Q(x)$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer identity fiber classifier saturation`

Theorem 82.27 (Centralizer-normalizer quotient identity fiber normality)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same hypotheses as Definition 82.24, define the quotient identity fiber carrier by*

$$K_Q(t) :\Longleftrightarrow Q_{Z_a \triangleleft N_a}(\mathsf{e}, t).$$

If $N_a(x)$ *and* $K_Q(k)$*, then the identity fiber is carried through both normal-kernel conjugation words:*

1. $K_Q(\mathsf{mul}(\mathsf{mul}(x,k),\mathsf{inv}(x)))$.
2. $K_Q(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(x),k),x))$.

Proof. Apply Definition 82.25 to identify the displayed fiber with the centralizer carrier. From $K_Q(k)$ the forward direction gives $Z_a(k)$.

For the forward conjugation word, use the first normalizer row contained in $N_a(x)$. It carries $Z_a(k)$ to $Z_a(\mathsf{mul}(\mathsf{mul}(x,k),\mathsf{inv}(x)))$. The reverse direction of the same identity-fiber equivalence then repacks this witness as $K_Q(\mathsf{mul}(\mathsf{mul}(x,k),\mathsf{inv}(x)))$.

For the inverse conjugation word, use the second normalizer row contained in $N_a(x)$. It carries $Z_a(k)$ to $Z_a(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(x),k),x))$. Applying the reverse direction of the identity-fiber equivalence again gives $K_Q(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(x),k),x))$. Thus the kernel identity fiber inherits the conjugation stability already present in the centralizer-normalizer tower. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer identity fiber normal`

Theorem 82.28 (Centralizer-normalizer quotient kernel certificate)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same hypotheses as Definition 82.24, define the quotient identity fiber carrier by*

$$K_Q(t) :\Longleftrightarrow Q_{Z_a \triangleleft N_a}(\mathsf{e}, t).$$

Then K_Q *is the concrete normal-kernel certificate for the representative quotient classifier* $Q_{Z_a \triangleleft N_a}$*:*

1. $K_Q(\mathsf{e})$.
2. *If* $K_Q(x)$ *and* $K_Q(y)$*, then* $K_Q(\mathsf{mul}(x,y))$.
3. *If* $K_Q(x)$*, then* $K_Q(\mathsf{inv}(x))$.
4. *If* $K_Q(x)$ *and* $x\mathsf{hsame}y$*, then* $K_Q(y)$.
5. *If* $N_a(x)$ *and* $K_Q(k)$*, then* $K_Q(\mathsf{mul}(\mathsf{mul}(x,k),\mathsf{inv}(x)))$.
6. *If* $N_a(x)$ *and* $K_Q(k)$*, then* $K_Q(\mathsf{mul}(\mathsf{mul}(\mathsf{inv}(x),k),x))$.

Proof. The subgroup rows come from Definition 82.25. Its identity, product, inverse, and hsame-transport clauses give the first four rows after unfolding the displayed definition of K_Q.

The conjugation rows come from Definition 82.27. With the same carrier K_Q and the same carried normalizer witness $N_a(x)$, its two clauses give the forward and inverse conjugation words listed above.

Together these rows present the identity fiber as the concrete normal kernel of the representative quotient classifier. The first block supplies the subgroup package; the second supplies conjugation stability by every carried normalizer element, which is the combined certificate consumed by first-isomorphism-style $\mathsf{QuotientGroup}^{\uparrow}$ kernel exactness. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer kernel certificate`

Per-object template: traditional target identity fiber of a quotient group; BEDC source $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, and $\mathsf{QuotientGroup}^{\uparrow}$ over BHist; core theorem list Definition 82.24, Definition 82.25, Definition 82.27, and Definition 82.28; carrier K_Q; classifier inherited from $Q_{Z_a \lhd N_a}$ and hsame; stability obligations identity, product, inverse, transport, and both conjugation words; ledger relative-kernel witness; exactness predicate $K_Q(x) \iff Z_a(x)$; downstream consumer first-isomorphism-style $\mathsf{QuotientGroup}^{\uparrow}$ kernel exactness; status $\mathsf{ProofTarget}$.

Theorem 82.29 (Centralizer-normalizer relative-kernel fiber exactness)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the same hypotheses as Definition 82.24 and Definition 82.8, define*

$$K_Q(t) :\iff Q_{Z_a \lhd N_a}(\mathsf{e}, t).$$

For histories $x, y : \mathsf{BHist}$ *with* $N_a(x)$ *and* $N_a(y)$*, set*

$$w = \mathsf{mul}(\mathsf{inv}(x), y).$$

Then the relative-kernel word, identity fiber, and quotient classifiers are exactly linked by

$$Q_{Z_a \lhd N_a}(x, y) \iff K_Q(w) \iff Z_a(w)$$

and

$$Q_{Z_a \leq N_a}(x, y) \iff K_Q(w).$$

Per-object template: traditional target first-isomorphism kernel exactness; BEDC source $\mathsf{Group}^{\uparrow}$*,* $\mathsf{Subgroup}^{\uparrow}$*, and* $\mathsf{QuotientGroup}^{\uparrow}$*; carrier* N_a *plus identity fiber* K_Q*; classifiers the two quotient representatives* $Q_{Z_a \lhd N_a}$ *and* $Q_{Z_a \leq N_a}$*; stability from the semantic certificate, the identity-fiber subgroup package, and orbit-kernel equivalence; ledger the relative-kernel word* w*; core theorem this block; exactness the displayed iff chain; downstream consumer first-isomorphism-style quotient-kernel exactness; status* $\mathsf{ProofTarget}$*.*

Proof. Unfold the relative-kernel classifier $Q_{Z_a \lhd N_a}$ at the carried endpoint pair (x, y). The endpoint hypotheses are already $N_a(x)$ and $N_a(y)$, so the only non-carrier datum is the centralizer witness for $w = \mathsf{mul}(\mathsf{inv}(x), y)$.

Next apply Definition 82.25. Its final row identifies the displayed identity fiber with the centralizer carrier:

$$K_Q(w) \iff Z_a(w).$$

Combining this row with the unfolded relative-kernel classifier gives the first displayed iff chain.

Finally transport the same endpoint pair across Definition 82.8. The carried hypotheses $N_a(x)$ and $N_a(y)$ are unchanged, and the equivalence between $Q_{Z_a \leq N_a}(x, y)$ and $Q_{Z_a \lhd N_a}(x, y)$ converts the first chain into $Q_{Z_a \leq N_a}(x, y) \iff K_Q(w)$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupUp StdBridge`

Capstone

Theorem 82.30 (Abelian centralizer-normalizer quotient terminal fiber)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the hypotheses of Definition 82.25, and additionally the* $\mathsf{AbGroup}^{\uparrow}$ *commutativity hypothesis used by Definition 42.17, fix* $a : \mathsf{BHist}$ *and define*

$$K_Q(x) :\iff Q_{Z_a \lhd N_a}(\mathsf{e}, x).$$

Then the quotient identity fiber is total:

$$\forall x : \mathsf{BHist},\ K_Q(x).$$

The subgroup-package rows of Definition 82.25 also give closure of this total fiber under product, inverse, and hsame *transport.*

Proof. Apply Definition 82.25. Its final row identifies the displayed carrier with the centralizer carrier: for every $x : \mathsf{BHist}$, $K_Q(x) \iff Z_a(x)$.

Now apply Definition 42.17. Under the same $\mathsf{AbGroup}^{\uparrow}$ commutativity hypothesis, its first row gives $Z_a(x)$ for every history x. Transporting this witness across the reverse direction of the identity-fiber equivalence gives $K_Q(x)$ for every $x : \mathsf{BHist}$.

The remaining assertion is inherited without changing the carrier. The product, inverse, and hsame-transport rows are exactly the second, third, and fourth rows of Definition 82.25, now read on a carrier whose every BEDC history lies in the identity fiber. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer abelian identity fiber total`

Theorem 82.31 (Abelian centralizer-normalizer quotient classifier totality)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the hypotheses of Definition 82.8, and additionally the* $\mathsf{AbGroup}^{\uparrow}$ *commutativity hypothesis used by Definition 42.17, fix* $a : \mathsf{BHist}$*. For any* $x, y : \mathsf{BHist}$ *with* $N_a(x)$ *and* $N_a(y)$*, both quotient representatives on the carrier* N_a *are total:*

$$Q_{Z_a \triangleleft N_a}(x, y) \iff \mathsf{True}$$

and

$$Q_{Z_a \leq N_a}(x, y) \iff \mathsf{True}.$$

Proof. The forward direction of each displayed equivalence is immediate, since any classifier witness proves True.

For the reverse direction of the relative-kernel classifier, assume True and set

$$w = \mathsf{mul}(\mathsf{inv}(x), y).$$

The first row of Definition 42.17 gives $Z_a(w)$ under the same commutativity hypothesis. Together with the endpoint hypotheses $N_a(x)$ and $N_a(y)$, this is exactly the witness required by Definition 80.33, so $Q_{Z_a \triangleleft N_a}(x, y)$ follows.

For the reverse direction of the coset classifier, transport the kernel classifier witness across Definition 82.8 at the same endpoint pair (x, y). The endpoint hypotheses are unchanged, and the reverse direction of the displayed equivalence gives $Q_{Z_a \leq N_a}(x, y)$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer abelian classifier totality`

The ten-field fit is: traditional target terminal quotient of an abelian centralizer-normalizer quotient; BEDC source $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, and $\mathsf{QuotientGroup}^{\uparrow}$ over BHist; carrier N_a; classifiers $Q_{Z_a \triangleleft N_a}$ and $Q_{Z_a \leq N_a}$; stability from the centralizer-normalizer tower and quotient classifier transport; ledger endpoint pair, relative-kernel word, and centralizer witness; core theorem list Definition 42.17, Definition 82.8, and Definition 80.33; exactness the two displayed iff statements; downstream consumer first-isomorphism-style quotient-kernel exactness; status ProofTarget.

Theorem 82.32 (Abelian centralizer-normalizer quotient terminal projection)**.** *For a BEDC history-level* $\mathsf{Group}^{\uparrow}$ *operation over* BHist *satisfying the hypotheses of Definition 82.24, and additionally the* $\mathsf{AbGroup}^{\uparrow}$ *commutativity hypothesis used by Definition 42.17, fix* $a : \mathsf{BHist}$*. For every carried endpoint* x *of* N_a*, define the constant representative projection to the singleton quotient carrier by*

$$\pi_Q(x) := \mathsf{e}.$$

For any $x, y : \mathsf{BHist}$ *with* $N_a(x)$ *and* $N_a(y)$*, the abelian centralizer-normalizer quotient classifier is the pullback of the singleton quotient classifier along this projection:*

$$Q_{Z_a \triangleleft N_a}(x, y) \iff \rho_Q(\pi_Q(x), \pi_Q(y)).$$

Moreover the projected product, inverse, and unit endpoints are already singleton-classifier related:

$$\rho_Q(\pi_Q(\mathsf{mul}(x, y)), \pi_Q(x)), \quad \rho_Q(\pi_Q(\mathsf{inv}(x)), \pi_Q(x)), \quad \rho_Q(\pi_Q(e), \mathsf{e}).$$

Proof. For the displayed equivalence, the left classifier is total under the commutativity hypothesis by Definition 82.31. Thus its truth value is True on every carried endpoint pair.

The right classifier is also inhabited on every displayed endpoint pair. Indeed, each projected endpoint is definitionally e, so unfolding the projection reduces it to $\rho_Q(\mathsf{e}, \mathsf{e})$. This is the reflexive singleton classifier edge packaged by Definition 82.11, using the singleton carrier and classifier of Definitions 82.1 and 82.2.

The product, inverse, and unit claims use the same endpoint computation. The semantic certificate of Definition 82.24 supplies the carried product, inverse, and unit endpoints in N_a; applying π_Q sends each one to e. Hence all three displayed singleton classifier statements again reduce to $\rho_Q(\mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer abelian terminal projection`

Theorem 82.33 (Abelian centralizer-normalizer terminal projection uniqueness)**.** *Under the hypotheses of Definition 82.32, let* x *be any carried endpoint of* N_a *and let* θ_x *be any displayed singleton representative with* $Q_{\mathsf{e}}(\theta_x)$*. Then the singleton classifier relates* θ_x *to the canonical projection* $\pi_Q(x) = \mathsf{e}$*:*

$$\rho_Q(\theta_x, \pi_Q(x)).$$

Consequently, for carried endpoints x, y *of* N_a *and displayed singleton representatives* θ_x, θ_y *with* $Q_{\mathsf{e}}(\theta_x)$ *and* $Q_{\mathsf{e}}(\theta_y)$*,*

$$\rho_Q(\theta_x, \theta_y) \iff \rho_Q(\pi_Q(x), \pi_Q(y)).$$

Thus the displayed-representative singleton classifier is equivalent to the abelian centralizer-normalizer quotient classifier through the pullback equivalence of Definition 82.32.

Proof. Unfold Q_{e} for the chosen representative. The displayed membership witness supplies $\theta_x\mathsf{hsamee}$, while the canonical projection computes definitionally as $\pi_Q(x) = \mathsf{e}$. Hence unfolding ρ_Q reduces the desired edge to transport along hsame from θ_x to e.

For two displayed representatives, unfold Q_{e} and ρ_Q for both endpoints. The witnesses give $\theta_x\mathsf{hsamee}$ and $\theta_y\mathsf{hsamee}$; symmetry and transitivity of hsame identify the classifier edge between θ_x and θ_y with the edge between e and e.

The canonical projections satisfy the same computation: $\pi_Q(x) = \mathsf{e}$ and $\pi_Q(y) = \mathsf{e}$. Therefore $\rho_Q(\pi_Q(x), \pi_Q(y))$ is the same singleton classifier edge $\rho_Q(\mathsf{e}, \mathsf{e})$, so the displayed-representative classifier and the canonical-projection classifier are equivalent.

Apply Definition 82.32 to the carried pair x, y. Its checked pullback equivalence transfers the canonical-projection classifier to $Q_{Z_a \triangleleft N_a}(x, y)$, and the previous paragraphs transport that equivalence to the arbitrary displayed singleton representatives. □

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer abelian terminal projection uniqueness`

The ten-field fit is: traditional target terminal projection from the abelian centralizer-normalizer quotient to the singleton quotient classifier; BEDC source $\mathsf{Group}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, and $\mathsf{QuotientGroup}^{\uparrow}$ over BHist; carrier N_a projected constantly to Q_{e}; classifiers $Q_{Z_a \triangleleft N_a}$ and ρ_Q; stability from the centralizer-normalizer quotient certificate and singleton classifier reflexivity; ledger carried endpoint pair, projected product endpoint, projected inverse endpoint, and projected unit endpoint; core theorem list Definition 82.24, Definition 82.31, and Definition 82.11; exactness the displayed pullback equivalence plus the three singleton projection edges; downstream consumer first-isomorphism-style terminal quotient exactness; status $\mathsf{ProofTarget}$.

Theorem 82.34 (Centralizer coset carrier representative product closure)**.** *For a BEDC history-level group multiplication* mul *over* BHist *satisfying the associativity, right empty-unit, and multiplication-congruence hypotheses of Definition 41.86, if two histories are carried by centralizer-coset carriers at representatives x and y, then their represented product is carried by the centralizer-coset carrier at the product representative:*

$$\mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, x, h_1) \wedge \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, y, h_2) \Longrightarrow \mathsf{CentralizerCosetCarrier}(\mathsf{mul}, a, \mathsf{mul}(x,$$

Lean4 ✓ `BEDC.Derived.QuotientGroupUp.CentralizerCosetCarrier representative product closure`

Proof. Unfold Definition 82.3. The first carrier witness gives $Z_a(x)$ together with $h_1 \mathsf{hsame} x$, and the second gives $Z_a(y)$ together with $h_2 \mathsf{hsame} y$.

The product-closure theorem Definition 41.86, equivalently the product row of Definition 80.2, gives $Z_a(\mathsf{mul}(x, y))$. Applying ambient multiplication congruence to $h_1 \mathsf{hsame} x$ and $h_2 \mathsf{hsame} y$ gives

$$\mathsf{mul}(h_1, h_2) \mathsf{hsame} \mathsf{mul}(x, y).$$

These are exactly the two fields required by Definition 82.3 for the representative $\mathsf{mul}(x, y)$ and endpoint $\mathsf{mul}(h_1, h_2)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$**.**

Constructive story: A quotient-group object here is not formed by making a host quotient type: it starts with a BEDC $\mathsf{Group}^{\uparrow}$ operation, a centralizer subgroup Z_a, and its normalizer carrier N_a, and it admits precisely the representative histories in N_a while the centralizer records which representative differences are invisible. Two accepted representatives x and y are classified either by the displayed coset-orbit witness or, equivalently, by the relative-kernel condition that $\mathsf{mul}(\mathsf{inv}(x), y)$ lies in Z_a; the chapter proves these readings agree and uses that classifier instead of any Quot soundness principle. The source specification is the consumed group and subgroup tower with normality and centralizer-normalizer rows, the pattern specification is representative multiplication, unit, inverse, identity fiber, abelian terminal projection, and kernel exactness, and the ledger keeps the two representatives, product and inverse words, centralizer witness, normalizer endpoint rows, and transport provenance visible. The name-certificate obligations are supplied by the empty representative in the normalizer, hsame transport of that representative, reflexivity, symmetry, and transitivity of the representative classifier, and closure of the carrier under replacing an accepted representative by a classified one. A representative history h lies in the signature gap of a $\mathsf{QuotientGroup}^{\uparrow}$ token p when the source-domain first accepts h in the normalizer carrier, the signature relation pairs it with an answer representative and the kernel-or-coset evidence ledger, and the token-introduction relation names that answer as p; after unfolding, this says that h represents the same centralizer-normalizer coset as the representative encoded by p. The paper bridge compares the displayed orbit and kernel criteria with the ordinary coset/kernel reading, but the BEDC surface remains the representative carrier plus classifier. The chapter registers this centralizer-normalizer construction under the name $\mathsf{QuotientGroup}^{\uparrow}$; arbitrary quotients, universal properties, host quotient objects, and checked standard-model bridge data are not introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{QuotientGroup}\uparrow}$, maturePackageClosed)

Scope closed: The mature QuotientGroupUp package is the centralizer-normalizer SemanticNameCert together with the abelian terminal projection, projection uniqueness, identity-fiber totality, and classifier-totality rows of Definition 82.24, Definition 82.32, Definition 82.33, Definition 82.30, and Definition 82.31.

Formal verification: FormalStatus(NameCert$_{\mathsf{QuotientGroup}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.QuotientGroupUp.QuotientGroupCentralizerNormalizer\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Arbitrary quotient groups, universal properties, arbitrary host quotients, and checked standard-model bridge data are not closed by this chapter.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 83

A Concrete Naming Certificate for $\mathsf{QuotientRing}^{\uparrow}$

The quotient-ring interface $\mathsf{QuotientRing}^{\uparrow}$ packages the carrier of cosets of an ideal in a ring. It depends on $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81) and lifts $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$ (chapter 82) to the multiplicative structure.

83.1 Carrier definition

83.2 The certificate

Definition 83.1 (Ideal coset relation for quotient rings)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$. Write $x -_R y$ for $x +_R (-_R y)$. The ideal-coset relation used by the $\mathsf{QuotientRing}^{\uparrow}$ carrier is*

$$a \equiv_I a' \iff C(a) \wedge C(a') \wedge I(a -_R a').$$

Theorem 83.2 (QuotientRing multiplication descends to ideal cosets)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate and let I be an $\mathsf{Ideal}^{\uparrow}$ object over R. Assume the ideal certificate supplies additive closure, classifier transport, and two-sided absorption by every carried ring endpoint, as required by section 81.2. For carried endpoints a, a', b, b', if*

$$a \equiv_I a' \qquad \text{and} \qquad b \equiv_I b',$$

then

$$(a \cdot_R b) \equiv_I (a' \cdot_R b').$$

Thus the multiplication operation on the $\mathsf{QuotientRing}^{\uparrow}$ coset carrier is representative-independent.

Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset mul descends`
Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset multiplication descent`

Proof. Unfold Definition 83.1. Set $d_a := a -_R a'$ and $d_b := b -_R b'$. The two coset hypotheses give $I(d_a)$ and $I(d_b)$, together with the carrier witnesses for a, a', b, b'.

By the right absorption field of the $\mathsf{Ideal}^{\uparrow}$ certificate, $I(d_a)$ and the carrier witness for b give

$$I(d_a \cdot_R b).$$

By the left absorption field, $I(d_b)$ and the carrier witness for a' give

$$I(a' \cdot_R d_b).$$

Additive closure in the ideal therefore gives

$$I((d_a \cdot_R b) +_R (a' \cdot_R d_b)).$$

It remains to identify this ideal element with the difference of the two products. By Definition 43.58,

$$d_a \cdot_R b \sim_R (a \cdot_R b) -_R (a' \cdot_R b),$$

and by Definition 43.56,

$$a' \cdot_R d_b \sim_R (a' \cdot_R b) -_R (a' \cdot_R b').$$

Addition congruence transports the sum to

$$((a \cdot_R b) -_R (a' \cdot_R b)) +_R ((a' \cdot_R b) -_R (a' \cdot_R b')).$$

The additive abelian-group rows of Definition 43.26, with the right inverse law from Definition 43.27 and the right zero law from Definition 43.31, cancel the middle pair $-_R(a' \cdot_R b)$ and $a' \cdot_R b$. Hence

$$(d_a \cdot_R b) +_R (a' \cdot_R d_b) \sim_R (a \cdot_R b) -_R (a' \cdot_R b').$$

Classifier transport for the ideal moves the displayed membership across this classifier edge, giving

$$I((a \cdot_R b) -_R (a' \cdot_R b')).$$

Multiplication closure in the $\mathsf{Ring}^{\uparrow}$ certificate supplies the carrier witnesses for $a \cdot_R b$ and $a' \cdot_R b'$. These two carrier witnesses together with the displayed ideal-membership statement are exactly

$$(a \cdot_R b) \equiv_I (a' \cdot_R b')$$

by Definition 83.1. □

Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset multiplication descends`

Theorem 83.3 (QuotientRing additive descent to ideal cosets)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$. Assume the ideal certificate supplies additive closure, additive inverse closure, and classifier transport for I, as specified in section 81.2. If*

$$a \equiv_I a' \qquad \text{and} \qquad b \equiv_I b',$$

then the additive and inverse operations respect ideal cosets:

$$(a +_R b) \equiv_I (a' +_R b')$$

and

$$(-_R a) \equiv_I (-_R a').$$

Thus the additive part of the $\mathsf{QuotientRing}^{\uparrow}$ coset carrier is representative-independent.

Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset additive descent`
Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset add descends`

Proof. Unfold Definition 83.1. Set

$$d_a := a -_R a', \qquad d_b := b -_R b'.$$

The two coset hypotheses give $C(a)$, $C(a')$, $C(b)$, $C(b')$, together with $I(d_a)$ and $I(d_b)$.

For addition, ideal additive closure gives

$$I(d_a +_R d_b).$$

The additive abelian-group component of the ring certificate, using Definition 42.13 for the operation $+_R$, identifies

$$(a +_R (-_R a')) +_R (b +_R (-_R b')) \sim_R (a +_R b) +_R ((-_R a') +_R (-_R b')).$$

By Definition 43.33, read in the additive component of R,

$$-_R(a' +_R b') \sim_R (-_R a') +_R (-_R b').$$

Addition congruence, symmetry, and transitivity of $\sim_R$ therefore give

$$d_a +_R d_b \sim_R (a +_R b) -_R (a' +_R b').$$

Classifier transport for the ideal moves the displayed membership across this classifier edge, so

$$I((a +_R b) -_R (a' +_R b')).$$

Ring additive closure supplies $C(a +_R b)$ and $C(a' +_R b')$. These carrier witnesses and the displayed ideal-membership statement are exactly

$$(a +_R b) \equiv_I (a' +_R b')$$

by Definition 83.1.

For additive inverse, ideal inverse closure applied to $I(d_a)$ gives

$$I(-_R d_a).$$

Since $d_a = a +_R (-_R a')$, Definition 43.33 gives the classifier comparison

$$-_R d_a \sim_R (-_R a) +_R (-_R(-_R a')),$$

and the right-hand side is $(-_R a) -_R (-_R a')$ by the notation of Definition 83.1. Classifier transport for the ideal yields

$$I((-_R a) -_R (-_R a')).$$

Ring additive inverse closure supplies $C(-_R a)$ and $C(-_R a')$. Repacking these two carrier witnesses with the displayed ideal-membership statement gives

$$(-_R a) \equiv_I (-_R a')$$

by Definition 83.1. □

Theorem 83.4 (QuotientRing ideal coset relation equivalence)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$, and write $x -_R y$ for $x +_R (-_R y)$ as in Definition 83.1. Assume the ideal certificate rows named in section 81.2 supply zero closure, additive-inverse closure, additive closure, and classifier transport:*

$$I(0_R), \qquad \forall x,\ I(x) \Longrightarrow I(-_R x), \qquad \forall x, y,\ I(x) \wedge I(y) \Longrightarrow I(x +_R y),$$

and

$$\forall x, y,\ I(x) \wedge x \sim_R y \Longrightarrow I(y).$$

Then the relation $\equiv_I$ of Definition 83.1 is an equivalence relation on carried ring endpoints: for every a, b, c,

$$C(a) \Longrightarrow a \equiv_I a,$$

$$a \equiv_I b \Longrightarrow b \equiv_I a,$$

and

$$(a \equiv_I b \wedge b \equiv_I c) \Longrightarrow a \equiv_I c.$$

Proof. Unfold Definition 83.1.

For reflexivity, assume $C(a)$. The zero row gives $I(0_R)$. The additive right-inverse theorem Definition 43.27, read in the additive component of R, gives

$$a -_R a \sim_R 0_R.$$

Classifier symmetry from Definition 9.7 reverses this edge, and classifier transport for I gives $I(a -_R a)$. Together with the two copies of $C(a)$, this is $a \equiv_I a$.

For symmetry, assume $a \equiv_I b$. The unfolded hypothesis supplies $C(a)$, $C(b)$, and $I(a -_R b)$. Additive-inverse closure gives

$$I(-_R(a -_R b)).$$

The additive abelian-group rows in Definition 43.26 identify the inverse of a difference with the reversed difference:

$$-_R(a -_R b) \sim_R b -_R a.$$

Indeed, Definition 43.33 rewrites the left endpoint to $(-_R a) +_R (-_R(-_R b))$; Definition 41.57, read in the additive group of R, identifies $-_R(-_R b)$ with b; additive commutativity then gives $b +_R (-_R a)$. The classifier edges are composed by Definition 9.8. Classifier transport for I yields $I(b -_R a)$. Repacking $C(b)$, $C(a)$, and this membership gives $b \equiv_I a$.

For transitivity, assume $a \equiv_I b$ and $b \equiv_I c$. Unfolding gives $C(a)$, $C(b)$, $C(c)$, $I(a -_R b)$, and $I(b -_R c)$. Additive closure gives

$$I((a -_R b) +_R (b -_R c)).$$

The additive associativity, left-inverse, left-zero, and addition-congruence rows in Definition 43.26 give

$$(a -_R b) +_R (b -_R c) \sim_R a -_R c.$$

After expanding the two differences, reassociate

$$(a +_R (-_R b)) +_R (b +_R (-_R c))$$

to

$$a +_R (((-_R b) +_R b) +_R (-_R c)).$$

The middle pair collapses to 0_R by the additive left-inverse row, and the additive left-zero row reduces the tail to $-_R c$. Thus the expression is classified with $a +_R (-_R c)$, namely $a -_R c$. Classifier transport for I gives $I(a -_R c)$. Repacking this with $C(a)$ and $C(c)$ gives $a \equiv_I c$. □

Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCosetRelation equivalence rows`

Theorem 83.5 (QuotientRing zero coset classification)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$, with carrier row supplying $C(0_R)$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$, and write $x -_R y$ for $x +_R (-_R y)$ as in Definition 83.1. Assume the ideal certificate includes classifier transport as specified in section 81.2:*

$$\forall x, y,\ I(x) \wedge x \sim_R y \Longrightarrow I(y).$$

Then for every endpoint a,

$$a \equiv_I 0_R \Longleftrightarrow C(a) \wedge I(a).$$

Lean4 ✓ `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset zero classification`

Proof. Fix a. We first record the conditional comparison with the zero difference. Suppose $C(a)$ and set $d_0 := a -_R 0_R$. The additive inverse of zero comparison from Definition 43.28, read in the additive component of R, gives

$$-_R 0_R \sim_R 0_R.$$

Addition congruence from Definition 43.26, with the left summand a fixed, gives

$$a +_R (-_R 0_R) \sim_R a +_R 0_R.$$

The additive right-zero law Definition 43.31 gives

$$a +_R 0_R \sim_R a.$$

By transitivity of the carrier classifier from Definition 9.7,

$$d_0 \sim_R a.$$

Thus, whenever $C(a)$ is available, $a -_R 0_R \sim_R a$.

For the forward implication, assume $a \equiv_I 0_R$. Unfolding Definition 83.1 gives $C(a)$, $C(0_R)$, and $I(a -_R 0_R)$. Applying the comparison just proved under this $C(a)$ and then the ideal classifier-transport row gives $I(a)$. Hence $C(a) \wedge I(a)$.

For the reverse implication, assume $C(a) \wedge I(a)$. The carrier row gives $C(0_R)$. Apply classifier symmetry from Definition 9.7 to the displayed comparison $d_0 \sim_R a$, obtaining

$$a \sim_R d_0.$$

The ideal classifier-transport row carries $I(a)$ across this edge, so $I(d_0)$, equivalently $I(a -_R 0_R)$. Repacking $C(a)$, $C(0_R)$, and this membership through Definition 83.1 yields $a \equiv_I 0_R$. □

Theorem 83.6 (QuotientRingUp public coset certificate)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$, and let $\equiv_I$ be the ideal-coset relation of Definition 83.1. Then the public $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$ coset surface is assembled from Definition 83.1, Definition 83.2, Definition 83.3, Definition 83.4, and Definition 83.5.*

Its carrier is the relation $a \equiv_I b$. Its classifier is stable under the ideal-coset relation. Its additive and multiplicative operations are independent of representatives. Its zero coset is exact:

$$a \equiv_I 0_R \iff C(a) \wedge I(a).$$

Its ledger is limited to $\mathsf{Ring}^{\uparrow}$ operations, $\mathsf{Ideal}^{\uparrow}$ closure and transport rows, and $\mathsf{QuotientGroup}^{\uparrow}$ coset inheritance.

Proof. The carrier component is Definition 83.1: a quotient-ring endpoint is represented by the relation $a \equiv_I b$, namely carrier evidence for both endpoints together with ideal membership of the difference $a -_R b$.

The classifier rows are supplied by Definition 83.4. Reflexivity gives the carried self-coset, symmetry reverses representatives, and transitivity composes two representative changes. These rows make classifier stability a property of the ideal-coset relation itself.

The operation rows come from the two descent results. Apply Definition 83.3 for addition and additive inverse, and apply Definition 83.2 for multiplication. The hypotheses of those theorems are exactly the $\mathsf{Ring}^{\uparrow}$ operation rows and the $\mathsf{Ideal}^{\uparrow}$ closure, absorption, and classifier-transport rows already recorded in the chapter ledger.

The zero row is Definition 83.5. It identifies the zero coset with carried ideal membership, so the certificate exposes no extra zero classifier beyond $C(a) \wedge I(a)$.

Repacking these rows gives the public $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$ coset surface. No further ring, ideal, or quotient-group clauses are added: the ledger remains bounded by the listed $\mathsf{Ring}^{\uparrow}$ operations, $\mathsf{Ideal}^{\uparrow}$ closure and transport data, and the inherited coset infrastructure from $\mathsf{QuotientGroup}^{\uparrow}$. □

Theorem 83.7 (QuotientRingUp public coset downstream bridge)**.** *Let R carry a $\mathsf{Ring}^{\uparrow}$ certificate with carrier C, classifier $\sim_R$, addition $+_R$, additive inverse $-_R$, zero 0_R, and multiplication $\cdot_R$. Let I be an $\mathsf{Ideal}^{\uparrow}$ object over R, written as a carried predicate $I(x)$, and let $\equiv_I$ be the ideal-coset relation of Definition 83.1. Downstream algebra consumers of the public $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$ coset surface may access quotient-ring data only through Definition 83.1, Definition 83.3, Definition 83.2, Definition 83.4, and Definition 83.5.*

Conversely, these rows supply the carrier, classifier, additive descent, multiplicative descent, zero row, and bounded ledger required by the downstream quotient-ring interface. The bridge boundary exposes neither representative choice, ideal witnesses beyond membership of differences, the universal property, localization, nor any standard bridge datum.

Proof. Start with the public certificate of Definition 83.6. Its carrier projection is exactly Definition 83.1: a visible quotient endpoint is accessed by the relation $a \equiv_I b$, and the only ideal witness retained at the boundary is membership of the difference $a -_R b$.

The classifier projection is Definition 83.4. Its reflexive, symmetric, and transitive rows give the consumer all representative-change data needed to treat $\equiv_I$ as the public classifier, without choosing a representative or exporting an ideal witness other than the displayed difference membership.

The operation projections are Definition 83.3 and Definition 83.2. They state that addition, additive inverse, and multiplication descend across $\equiv_I$, so the downstream interface receives operation stability directly from the public rows rather than from hidden representative choices.

The zero projection is Definition 83.5. It identifies the zero coset by $C(a) \wedge I(a)$, which is the zero row required by the consumer surface and does not add a separate zero classifier.

Repacking these five projections yields the downstream bridge boundary: carrier, classifier, operation descent, zero row, and bounded ledger are all supplied by the public coset certificate. Since the repacking uses only the listed rows, no universal property, localization theorem, ideal correspondence, full quotient-ring API, or standard bridge datum is closed here. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$.

Constructive story: A quotient-ring endpoint is not made by forming a new quotient object; it is a visible ideal-coset relation between carried ring histories, where $a \equiv_I a'$ means that both representatives are in the ring carrier and the displayed difference $a -_R a'$ lies in the ideal. Two visible coset rows are treated as the same through that same ideal-coset relation, whose reflexive, symmetric, and transitive rows are proved from the ideal zero, inverse, additive closure, and classifier-transport fields rather than from a representative choice principle. The source is the already certified $\mathsf{Ring}^{\uparrow}$ surface together with an $\mathsf{Ideal}^{\uparrow}$ predicate, the pattern records additive descent, additive-inverse descent, multiplicative descent by two-sided absorption, zero-coset exactness, and inherited quotient-group infrastructure, and the ledger is bounded to the ring operations, ideal membership of differences, and the proofs that operations respect those displayed relations. The five name-certificate obligations close because every carried endpoint has a reflexive coset using the ideal zero row, symmetry reverses a difference by ideal inverse closure, transitivity adds two displayed differences, and carrier-respect is exactly the retained ideal transport along the ring classifier. A history row h lies in the signature gap of a $\mathsf{QuotientRing}^{\uparrow}$ token p when the source-domain accepts the underlying carried representatives and ideal-difference witness, the signature relation pairs the coset row with an answer row and evidence ledger, and token introduction names that answer as p; unfolded, the answer is the same public coset data with addition and multiplication already independent of representative. The downstream bridge repacks only the public carrier, classifier, operation descent, zero row, and bounded ledger, so consumers never receive a hidden chosen representative. The chapter registers this ideal-coset surface under the name $\mathsf{QuotientRing}^{\uparrow}$; it introduces no external standard bridge, universal property, localization theorem, ideal correspondence theorem, or full quotient-ring API beyond the displayed rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 83.6 exports the public coset certificate, and Definition 83.7 repacks that certificate as the downstream bridge boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.QuotientRingUp.QuotientRingIdealCoset\_multiplication\_descends`

Bridge status: paperBridge

Not claimed: This chapter does not close external standard bridge data, the universal property, localization, ideal correspondence, or the full quotient-ring API.

Upgrade path: No paper-side closure grade remains; further movement is formal and standard-bridge verification only.

Chapter 84

A Concrete Naming Certificate for $\mathsf{Determinant}^{\uparrow}$

The determinant interface $\mathsf{Determinant}^{\uparrow}$ packages the alternating multilinear form on square matrices that detects invertibility. It depends on $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ (chapter 49) and $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44), and supplies the invariant used by the eigenvalue and characteristic-polynomial certificates.

84.1 Carrier definition

Definition 84.1 (Singleton matrix determinant certificate)**.** *Use the singleton empty-history matrix carrier and classifier from* $\mathsf{Mat}^{\uparrow}$*,*

$$\mathsf{MatrixSingletonCarrier}, \qquad \mathsf{MatrixSingletonClassifier},$$

together with the singleton empty-history commutative-ring carrier and classifier from $\mathsf{CommRing}^{\uparrow}$*,*

$$\mathsf{CommRingSingletonCarrier}, \qquad \mathsf{CommRingSingletonClassifier}.$$

The determinant readback is the constant scalar endpoint

$$\det_{\mathsf{e}}(M) := \mathsf{e}.$$

Let $\mathsf{MatrixSingletonOne}$ *denote the singleton matrix identity and* $\mathsf{CommRingSingletonOne}$ *the singleton commutative-ring unit. Define the endpoint relation by*

$$\Delta_{\mathsf{e}}(M, r) \iff \mathsf{MatrixSingletonClassifier}(M, \mathsf{MatrixSingletonOne}) \wedge \mathsf{CommRingSingletonClassifier}(r, \mathsf{CommRingSingleto}$$

The certificate fit has ten fields:

1. *Traditional target: the terminal determinant on the one-by-one terminal matrix algebra.*
2. *BEDC source: the singleton* $\mathsf{Mat}^{\uparrow}$ *instance over the singleton* $\mathsf{CommRing}^{\uparrow}$ *scalar instance.*
3. *Carrier choice:* $\mathsf{MatrixSingletonCarrier}$.
4. *Classifier choice:* $\mathsf{MatrixSingletonClassifier}$*, with scalar readback classified by* $\mathsf{CommRingSingletonClassifier}$.
5. *Stability obligations: identity stability, classifier transport, multiplicativity, and row-swap sign collapse at the singleton endpoint.*

6. *Ledger: the singleton matrix endpoint together with the singleton scalar endpoint.*
7. *Core theorem list: Theorem 84.2, Theorem 49.48, and Theorem 44.7.*
8. *Exactness predicate:* $\Delta_{\mathsf{e}}(M, r)$.
9. *Standard bridge: the determinant of the terminal one-by-one matrix algebra.*
10. *Status:* ProofTarget.

Lean4 (definition) `BEDC.Derived.DeterminantUp.DeterminantSingletonCertificate`
Lean4 (definition) `BEDC.Derived.DeterminantUp.DeterminantSingletonReadback`
Lean4 (definition) `BEDC.Derived.DeterminantUp.DeterminantSingletonDet`
Lean4 (definition) `BEDC.Derived.DeterminantUp.DeterminantSingletonEndpoint`

Theorem 84.2 (Singleton matrix determinant endpoint correspondence)**.** *For histories* M, r : BHist*, the determinant endpoint relation satisfies*

$$\Delta_{\mathsf{e}}(M, r) \iff \mathsf{MatrixSingletonCarrier}(M) \wedge \mathsf{CommRingSingletonClassifier}(r, \mathsf{CommRingSingletonOne}).$$

Moreover, for singleton-carried histories M, N : BHist*:*

1. $\det_{\mathsf{e}}(\mathsf{MatrixSingletonOne})$ *is classified with* CommRingSingletonOne *by* CommRingSingletonClassifier.
2. *If* $\mathsf{MatrixSingletonClassifier}(M, N)$*, then* $\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(M), \det_{\mathsf{e}}(N))$.
3. $\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(\mathsf{MatrixSingletonMul}(M, N)), \mathsf{CommRingSingletonMul}(\det_{\mathsf{e}}(M), \det_{\mathsf{e}}(N)))$.

Proof. For the forward direction of the displayed equivalence, unfold Δ_{e}. The matrix component classifies M with MatrixSingletonOne, so its carrier field gives $\mathsf{MatrixSingletonCarrier}(M)$. The scalar component is already the displayed CommRingSingletonClassifier witness.

For the reverse direction, apply Theorem 49.49 with the carried matrix M and the carried identity endpoint supplied by Theorem 49.48. Equivalently, Theorem 49.49 collapses the singleton matrix endpoint to e, and the identity law in Theorem 49.48 identifies MatrixSingletonOne with the same endpoint. Together with the given scalar classifier witness, these two components repackage $\Delta_{\mathsf{e}}(M, r)$.

For the determinant of the identity, $\det_{\mathsf{e}}$ is constantly e and CommRingSingletonOne is the singleton scalar endpoint. The commutative-ring singleton laws of Theorem 44.7 classify these two scalar histories at the same endpoint.

Classifier transport is immediate from the same scalar collapse. If $\mathsf{MatrixSingletonClassifier}(M, N)$, both determinant readbacks are e, so the commutative-ring singleton laws package the scalar comparison.

For multiplicativity, the matrix product endpoint is first collapsed by Theorem 49.49. The two scalar sides are then e and $\mathsf{CommRingSingletonMul}(\mathsf{e}, \mathsf{e})$; the singleton commutative-ring laws identify the multiplication result with CommRingSingletonOne, so the final scalar classifier witness is again packaged at e. □

Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingleton semanticNameCert`
Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingleton certificate`
Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingleton endpoint correspondence`
Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingletonEndpoint correspondence`
Lean4 ✓ `BEDC.Derived.DeterminantUp.determinant singleton endpoint laws`

Theorem 84.3 (Determinant append-pair scalar classifier)**.** *For histories M, N, r : BHist, the scalar classifier of the singleton determinant readback on* $\mathsf{append}(M, N)$ *is equivalent to singleton matrix carriers for M and N together with the singleton commutative-ring carrier for r.*

Proof. Unfold the determinant readback as an append with the empty suffix. In the forward direction, the scalar classifier supplies emptiness of $\mathsf{append}(\mathsf{append}(M, N), \mathsf{e})$; append-empty inversion first removes the determinant suffix and then splits the matrix pair into carrier witnesses for M and N. The target scalar carrier is the second endpoint field of the classifier. Conversely, the two matrix carrier witnesses make $\mathsf{append}(M, N)$ empty, the determinant suffix keeps it empty, and this empty determinant endpoint classifies with the carried scalar endpoint r. □

Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingletonDet append pair scalar classifier iff`

Theorem 84.4 (Determinant visible matrix scalar classifier exclusion)**.** *For histories M, r : BHist, neither visible matrix endpoint can feed the singleton determinant readback into a scalar classifier:*

$$\mathsf{CommRingSingletonClassifier}(\mathsf{DeterminantSingletonDet}(\mathsf{E0}(M)), r) \Longrightarrow \bot, \qquad \mathsf{CommRingSingletonClassifier}(\mathsf{Determ}$$

Proof. Unfold the singleton determinant readback as append with the empty suffix and read the left carrier field of the scalar classifier. The field forces the visible determinant endpoint to be history-same to e. Constructor no-confusion for E0 and E1 contradicts these two possible carrier clauses. □

Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingletonDet visible matrix scalar classifier absurd`

84.2 The certificate

Theorem 84.5 (Determinant singleton classifier semantic name certificate)**.** *The determinant singleton classifier over* MatrixSingletonCarrier *is a semantic name certificate:*

$$\mathsf{SemanticNameCert}(\mathsf{MatrixSingletonCarrier}, \mathsf{MatrixSingletonCarrier}, \mathsf{MatrixSingletonCarrier}, \mathsf{DeterminantSingletonClassi}$$

Proof. Use the matrix singleton certificate supplied by Theorem 49.48 for carrier inhabitation, reflexivity, symmetry, transitivity, and carrier transport on the matrix component. The determinant readback is the constant singleton scalar endpoint, and the commutative-ring singleton certificate supplied by Theorem 44.7 classifies the scalar readbacks. Pairing these two classifier components gives the determinant classifier fields. Pattern and ledger soundness are the identity inclusions because all three semantic predicates are MatrixSingletonCarrier. □

Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingletonClassifier semanticNameCert`

Theorem 84.6 (Singleton determinant threefold multiplicativity)**.** *For singleton-carried matrix histories M, N, P : BHist, the left-bracketed matrix product has the scalar determinant classifier*

$$\mathsf{CommRingSingletonClassifier}(\mathsf{DeterminantSingletonDet}(\mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonMul}(M, N), P)), \mathsf{CommR}$$

The right-bracketed matrix product has the analogous classifier

$$\mathsf{CommRingSingletonClassifier}(\mathsf{DeterminantSingletonDet}(\mathsf{MatrixSingletonMul}(M, \mathsf{MatrixSingletonMul}(N, P))), \mathsf{CommR}$$

Moreover, the two scalar products above are classified with one another by CommRingSingletonClassifier*.*

Proof. The singleton matrix laws in Theorem 49.48 close $\mathsf{MatrixSingletonMul}(M, N)$ and then $\mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonMul}(M, N), P)$ under $\mathsf{MatrixSingletonCarrier}$. The same laws also close $\mathsf{MatrixSingletonMul}(N, P)$ and $\mathsf{MatrixSingletonMul}(M, \mathsf{MatrixSingletonMul}(N, P))$, and their associativity field classifies the two matrix products at the singleton endpoint.

Apply Theorem 84.2 to the left-bracketed product and then use Theorem 84.5 to keep the determinant readback stable under the singleton matrix classifier. The result is the displayed classifier from the determinant of $\mathsf{MatrixSingletonMul}(\mathsf{MatrixSingletonMul}(M, N), P)$ to the left-bracketed scalar product.

The right-bracketed statement is obtained by the same argument applied to $\mathsf{MatrixSingletonMul}(M, \mathsf{MatrixSinglet}$ Matrix associativity transports the product endpoint, and determinant readback stability gives the corresponding scalar classifier.

Finally, Theorem 44.7 closes $\mathsf{CommRingSingletonMul}$ on the three determinant readbacks and classifies the left- and right-bracketed scalar products by commutative-ring associativity. □

Closure status for $\mathsf{NameCert}_{\mathsf{Determinant}^{\uparrow}}$.

Constructive story: The determinant surface closed here is the terminal singleton case: an admitted matrix history is a $\mathsf{Mat}^{\uparrow}$ singleton endpoint, and its determinant readback is the constant singleton commutative-ring scalar endpoint e, so any visible matrix endpoint with a b0 or b1 head is excluded by the scalar classifier's empty-endpoint readback. Two carried determinant rows are classified by the singleton matrix classifier on the matrix side together with the singleton commutative-ring classifier on the scalar readback, and the endpoint relation says exactly that the matrix is classified with the singleton identity while the scalar is classified with the singleton unit. The source, pattern, and ledger all stay on this singleton matrix carrier; the pattern records constant determinant readback, identity endpoint correspondence, classifier transport, multiplicativity, visible-endpoint exclusion, and the singleton row-swap sign collapse, while the ledger keeps the matrix endpoint and scalar endpoint side by side. The five name-certificate obligations are inherited from the singleton matrix certificate for carrier inhabitation and matrix endpoint comparison, paired with the singleton commutative-ring certificate for the scalar readback, so reflexivity, symmetry, transitivity, and carrier transport never leave the one-point surface. A history h lies in the signature gap of a $\mathsf{Determinant}^{\uparrow}$ token p when the source-domain accepts h as a singleton matrix endpoint, the signature relation pairs it with the constant determinant scalar and evidence ledger, and the token-introduction relation names that scalar answer as p; unfolded, every accepted determinant value is the singleton scalar endpoint. There is no standard bridge asserted in this closure block, though the intended outside reading is the determinant of the terminal one-by-one matrix algebra. The chapter registers this constant singleton determinant under the name $\mathsf{Determinant}^{\uparrow}$; general $n \times n$ determinants, Leibniz sums, row operations, alternating multilinear uniqueness, characteristic polynomials, and determinant theory over arbitrary fields are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Determinant}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Singleton determinant semantic certificate over MatrixSingletonCarrier and CommRingSingletonClassifier, including constant determinant readback, endpoint correspondence, classifier transport, and singleton multiplicativity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Determinant}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DeterminantUp.DeterminantSingleton\_semanticNameCert`

Bridge status: none

Not claimed: General $n \times n$ determinant, Leibniz formula, alternating multilinear uniqueness, row operations, characteristic polynomial, or determinant theory over arbitrary fields.

Upgrade path: No paper-side theory grade remains for the singleton determinant surface.

Definition 84.7 (Determinant singleton row-swap endpoint certificate)**.** *For histories* M, S : BHist*, write*

$$\mathsf{DetSwap}_{\mathsf{e}}(M, S) \iff \mathsf{MatrixSingletonCarrier}(M) \wedge \mathsf{MatrixSingletonClassifier}(S, M).$$

This is the row-swap endpoint certificate at the singleton determinant surface: the endpoint S *is read through the same singleton matrix classifier as the source matrix* M*.*

Theorem 84.8 (Determinant singleton row-swap sign collapse)**.** *For histories* M, S : BHist*, every singleton row-swap endpoint certificate*

$$\mathsf{DetSwap}_{\mathsf{e}}(M, S)$$

supplies the signed determinant classifier

$$\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(S), \mathsf{CommRingSingletonNeg}(\det_{\mathsf{e}}(M)))$$

and hence the unsigned determinant classifier

$$\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(S), \det_{\mathsf{e}}(M)).$$

Lean4 ✓ `BEDC.Derived.DeterminantUp.DeterminantSingletonRowSwapEndpointCertificate sign collapse`

Proof. Unpack $\mathsf{DetSwap}_{\mathsf{e}}(M, S)$. Its second component is the matrix classifier $\mathsf{MatrixSingletonClassifier}(S, M)$, and its first component keeps M in the singleton matrix carrier. Applying the classifier-transport clause in Definition 84.2 to S and M gives

$$\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(S), \det_{\mathsf{e}}(M)).$$

Thus both determinant readbacks are singleton scalar endpoints.

Apply Definition 44.9 with $h = \det_{\mathsf{e}}(M)$ and $t = \det_{\mathsf{e}}(S)$. Since $\det_{\mathsf{e}}(S)$ is a singleton scalar endpoint and $\mathsf{CommRingSingletonNeg}(\det_{\mathsf{e}}(M))$ is the empty singleton scalar endpoint, the singleton commutative-ring classifier relates $\mathsf{CommRingSingletonNeg}(\det_{\mathsf{e}}(M))$ to $\det_{\mathsf{e}}(S)$. Symmetry of the scalar classifier, supplied by Definition 44.7, gives the displayed signed orientation

$$\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(S), \mathsf{CommRingSingletonNeg}(\det_{\mathsf{e}}(M))).$$

Apply the same negation endpoint theorem with $t = \det_{\mathsf{e}}(M)$ to classify $\mathsf{CommRingSingletonNeg}(\det_{\mathsf{e}}(M))$ with $\det_{\mathsf{e}}(M)$. Transitivity of the singleton scalar classifier from Definition 44.7 composes this comparison with the signed classifier above, yielding

$$\mathsf{CommRingSingletonClassifier}(\det_{\mathsf{e}}(S), \det_{\mathsf{e}}(M)).$$

□

Chapter 85

A Concrete Naming Certificate for $\mathsf{Eigen}^{\uparrow}$

The eigen-pair interface $\mathsf{Eigen}^{\uparrow}$ packages an eigenvalue / eigenvector pair (λ, v) for a linear endomorphism. It depends on $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ (chapter 48) and $\mathsf{NameCert}_{\mathsf{Determinant}^{\uparrow}}$ (chapter 84), with λ certified as a root of the characteristic polynomial.

85.1 Carrier definition

Definition 85.1 (Eigen component singleton carrier)**.** *For histories $f, \lambda, v, pair : \mathsf{BHist}$, the component singleton eigen carrier requires f and v to be singleton linear-map histories, λ to be a singleton commutative-ring history, and pair to be the BEDC continuation endpoint of λ followed by v.*

Lean4 (definition) `BEDC.Derived.EigenUp.EigenComponentSingletonCarrier`

Theorem 85.2 (Eigen component singleton carrier singleton append)**.** *For histories $m, \lambda, v, pair : \mathsf{BHist}$, if the component singleton eigen carrier holds for $(m, \lambda, v, pair)$ and λ also carries the determinant singleton scalar certificate, then $\mathsf{append}(m, pair)$ lies in the eigen singleton carrier.*

Proof. Unpack the component carrier into the linear-map carrier for m, the linear-map carrier for v, and the continuation endpoint $\mathsf{Cont}(\lambda, v, pair)$. The determinant singleton carrier for λ supplies the scalar witness required by the eigen singleton carrier. The stored continuation rewrites $\mathsf{append}(m, pair)$ as $\mathsf{append}(m, \mathsf{append}(\lambda, v))$, giving the required BEDC continuation endpoint. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenComponentSingletonCarrier singleton append`

Theorem 85.3 (Eigen component singleton carrier continuation result transport)**.** *For histories $m, \lambda, v, pair, full, full' : \mathsf{BHist}$, suppose the component singleton eigen carrier holds for $(m, \lambda, v, pair)$, λ carries the determinant singleton scalar certificate, $\mathsf{Cont}(m, pair, full)$, and $full \mathsf{hsame} full'$. Then $full'$ lies in the eigen singleton carrier.*

Proof. Unpack the component carrier to expose the linear-map carriers for m and v and the scalar-vector continuation $\mathsf{Cont}(\lambda, v, pair)$. Compose this continuation with $\mathsf{Cont}(m, pair, full)$ to rewrite the endpoint as $\mathsf{Cont}(m, \mathsf{append}(\lambda, v), full)$. The determinant singleton carrier for λ and the final hsame-transport place $full'$ in the eigen singleton carrier. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenComponentSingletonCarrier cont result transport`

Theorem 85.4 (Eigen component singleton carrier pair determinacy)**.** *For histories $f, \lambda, v, pair, pair'$: BHist, two component singleton eigen carriers over the same (f, λ, v) determine the same scalar-vector endpoint $\mathsf{hsame}(pair, pair')$. Both stored continuation ledgers $\mathsf{Cont}(\lambda, v, pair)$ and $\mathsf{Cont}(\lambda, v, pair')$ remain available.*

Proof. Unpack both component singleton carriers. They contain the two BEDC continuation ledgers with the same source λ and tail v. The kernel determinacy theorem for continuation endpoints identifies their results, and the two ledgers are retained directly from the carriers. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenComponentSingletonCarrier pair deterministic`

Theorem 85.5 (Eigen component singleton carrier append classifier)**.** *For histories $m, m', \lambda, \lambda', v, v', pair, pair'$: BHist, suppose the component singleton eigen carriers hold for $(m, \lambda, v, pair)$ and $(m', \lambda', v', pair')$, both scalar components carry determinant singleton certificates, and the three displayed components satisfy $m\mathsf{hsame}m'$, $\lambda\mathsf{hsame}\lambda'$, and $v\mathsf{hsame}v'$. Then the append endpoints $\mathsf{append}(m, pair)$ and $\mathsf{append}(m', pair')$ are classified by the eigen singleton classifier.*

Proof. The component carriers and determinant singleton scalar certificates give the two eigen singleton carrier endpoints by Definition 85.2. Unpacking the component carriers exposes the two BEDC continuations $\mathsf{Cont}(\lambda, v, pair)$ and $\mathsf{Cont}(\lambda', v', pair')$. Continuation respects the scalar and vector history equalities, so $pair\mathsf{hsame}pair'$. Applying the same continuation-respect principle to the map equality and these pair endpoints identifies $\mathsf{append}(m, pair)$ with $\mathsf{append}(m', pair')$, closing the restricted classifier. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenComponentSingletonCarrier append classifier`

Theorem 85.6 (Eigen singleton carrier pair emptiness)**.** *For histories $f, \lambda, v, pair$: BHist, the component singleton eigen carrier is equivalent to the three singleton component carriers together with $\mathsf{hsame}(pair, \mathsf{e})$.*

Proof. Unfold the component singleton eigen carrier. In the forward direction, the scalar and vector component carriers identify λ and v with e, so $\mathsf{append}(\lambda, v)$ is empty by the BEDC append-empty inversion, and the stored continuation transports this emptiness to $pair$. In the reverse direction, the same append-empty fact reconstructs the continuation endpoint from the given $\mathsf{hsame}(pair, \mathsf{e})$ witness. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier pair empty iff`

Definition 85.7 (Eigen singleton carrier)**.** *A history p lies in the singleton eigen carrier when there are histories m, λ, v such that m and v are singleton linear-map carriers, λ is a singleton determinant carrier, and p is the continuation result*

$$\mathsf{Cont}(m, \mathsf{append}(\lambda, v), p).$$

Lean4 (definition) `BEDC.Derived.EigenUp.EigenSingletonCarrier`

Theorem 85.8 (Eigen singleton carrier continuation result transport)**.** *For histories m, λ, v, p, p' : BHist, if m and v are singleton linear-map carriers, λ is a singleton determinant carrier, $\mathsf{Cont}(m, \mathsf{append}(\lambda, v), p)$, and $p\mathsf{hsame}p'$, then p' lies in the eigen singleton carrier.*

Proof. Reuse the same displayed witnesses m, λ, v. The carrier fields are unchanged, and the continuation result transports along $p\mathsf{hsame}p'$ by the kernel result-transport theorem for Cont. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier cont result transport`

Theorem 85.9 (Eigen singleton carrier empty iff)**.** *For every history p : BHist, the singleton eigen carrier holds for p exactly when $p\mathsf{hsamee}$.*

Proof. For the forward direction, unpack the map, scalar, and vector witnesses. The linear-map and determinant singleton carriers identify each component with e, so the nested append endpoint is empty; the stored continuation transports this emptiness to p. For the reverse direction, choose all three witness components to be e and use the displayed $p\mathsf{hsamee}$ witness as the empty continuation result. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier empty iff`

Theorem 85.10 (Eigen singleton continuation sources empty)**.** *For BEDC histories map, payload, and pair, if pair lies in the eigen singleton carrier and $\mathsf{Cont}(map, payload, pair)$, then both continuation sources are empty:*

$$\mathsf{EigenSingletonCarrier}(pair) \wedge \mathsf{Cont}(map, payload, pair) \Longrightarrow map\mathsf{hsamee} \ \wedge \ payload\mathsf{hsamee}.$$

Proof. The carrier-empty equivalence identifies *pair* with e. The continuation payload identifies *pair* with $\mathsf{append}(map, payload)$, hence the append endpoint is empty. BEDC append-empty inversion splits this endpoint into the two empty source comparisons. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier cont sources empty`

Theorem 85.11 (Eigen singleton carrier append-context empty iff)**.** *For histories L, R, h : BHist, the contextual eigen singleton carrier holds for $\mathsf{append}(L, \mathsf{append}(h, R))$ exactly when the two context histories are empty and the middle history lies in the eigen singleton carrier:*

$$\mathsf{EigenSingletonCarrier}(\mathsf{append}(L, \mathsf{append}(h, R))) \Longleftrightarrow L\mathsf{hsamee} \wedge \mathsf{EigenSingletonCarrier}(h) \wedge R\mathsf{hsamee}.$$

Proof. Use the eigen singleton carrier empty equivalence to convert the contextual carrier into emptiness of the nested append. BEDC append-empty inversion splits the outer and inner append endpoints, yielding empty left and right contexts and empty middle endpoint. The reverse direction rebuilds the nested append-empty witness from the three displayed fields and applies the same carrier-empty equivalence backwards. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier append context empty iff`

Theorem 85.12 (Eigen singleton carrier continuation left-unit result)**.** *For histories $pair, q, result$: BHist, if pair lies in the eigen singleton carrier and $\mathsf{Cont}(pair, q, result)$, then $result\mathsf{hsamee}q$.*

Proof. The eigen singleton carrier empty equivalence identifies *pair* with e. After this source readback, the displayed continuation has empty source, so the BEDC left-unit result theorem for Cont reads the endpoint back to the tail q. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier cont left unit result`

Theorem 85.13 (Eigen singleton carrier continuation right-unit result)**.** *For histories $q, pair, result$: BHist, if pair lies in the eigen singleton carrier and $\mathsf{Cont}(q, pair, result)$, then $result\mathsf{hsamee}q$.*

Proof. The carrier-empty equivalence identifies *pair* with e. Transport the displayed continuation across that endpoint comparison and compare it with the right-unit continuation at q; continuation determinacy gives the result. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonCarrier cont right unit result`

85.2 The certificate

Definition 85.14 (Eigen singleton classifier)**.** *Two histories* p, q : BHist *are classified by the singleton eigen interface when both lie in the singleton eigen carrier and* $p$hsame$q$*.*

Lean4 (definition) `BEDC.Derived.EigenUp.EigenSingletonClassifier`

Theorem 85.15 (Eigen singleton classifier append-context empty iff)**.** *For histories* L, R, h, k : BHist*, the contextual singleton eigen classifier between* append(L, h) *and* append(k, R) *is equivalent to empty endpoint contexts and the core classifier between* h *and* k*:*

$$\mathsf{EigenSingletonClassifier}(\mathsf{append}(L, h), \mathsf{append}(k, R)) \iff L\mathsf{hsamee} \wedge \mathsf{EigenSingletonClassifier}(h, k) \wedge R\mathsf{hsamee}.$$

Proof. The forward direction sends both contextual carrier fields through the eigen carrier empty equivalence and splits the two append-empty witnesses. These middle empty witnesses give the core carriers and identify h with k. In the reverse direction, the empty endpoint contexts and the two core carrier fields rebuild the left and right contextual carriers, and both contextual endpoints are history-same because each is empty. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonClassifier append context empty iff`

Theorem 85.16 (Eigen singleton classifier continuation results hsame)**.** *For histories* h, k, q, q', r, r' : BHist*, if* EigenSingletonClassifier(h, k)*,* $q$hsameq'*,* Cont(h, q, r)*, and* Cont(k, q', r')*, then* $r$hsamer'*.*

Proof. The classifier exposes eigen singleton carrier witnesses for both sources. The carrier continuation left-unit result reads r back to q and r' back to q'. Composing these two readbacks with the supplied tail equality $q$hsameq' gives the endpoint equality $r$hsamer'. □

Lean4 ✓ `BEDC.Derived.EigenUp.EigenSingletonClassifier continuation results hsame`

Theorem 85.17 (Eigen singleton semantic name certificate)**.** *The singleton eigen carrier, equipped with the carrier-restricted* hsame *classifier above, satisfies the semantic naming-certificate schema.*

Proof. The carrier is inhabited by the displayed empty eigen pair: empty linear-map history, empty determinant history, empty vector history, and the empty continuation endpoint. Reflexivity, symmetry, and transitivity are inherited from hsame, with carrier membership retained in the classifier fields. Pattern and ledger soundness are the identity inclusions on the same carrier. □

Lean4 ✓ `BEDC.Derived.EigenUp.eigen singleton semantic name certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{Eigen}^\uparrow}$**.**

Constructive story: An $\mathsf{Eigen}^\uparrow$ singleton certificate is generated from a linear-map history, scalar history, vector history, and scalar-vector continuation endpoint. In the displayed BEDC carrier, the scalar row is admitted through the determinant singleton certificate, the map and vector rows are admitted through linear-map singleton carriers, and Cont composes the scalar-vector pair with the map row. The hsame classifier then transports and compares the resulting eigen endpoint. In this singleton presentation the accepted endpoint collapses to the empty history, so the chapter certifies the BEDC name discipline for the eigen interface without adding analytic spectral data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Eigen}^\uparrow}$, seed)

Scope closed: This chapter binds the Eigen singleton component carrier, singleton carrier, classifier, and semantic name certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Eigen}^\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a general spectral theorem, nonzero eigenvector existence, host vector-space equality, or characteristic-polynomial completeness.

Upgrade path: To reach obligationClosure, align the paper closure block with a checked Eigen carrier target and its classifier transport theorems.

Chapter 86

A Concrete Naming Certificate for $\mathsf{InnerProduct}^{\uparrow}$

The inner-product interface $\mathsf{InnerProduct}^{\uparrow}$ packages a vector space together with a constructive sesquilinear form satisfying conjugate symmetry, linearity, and positive definiteness. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and is the direct prerequisite for the Hilbert certificate $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$. The scalar-form rows consume the vector carrier and classifier of chapter 47 and provide the immediate finite source surface for chapter 91.

The mature $\mathsf{InnerProduct}^{\uparrow}$ row is also the bilinear, symmetric, positive source consumed by chapter 89, chapter 91, and bounded-operator or Hilbert-completion routes: norm witnesses, distance witnesses, completion obligations, orthogonality witnesses, and operator-domain compatibility are read from the displayed scalar-form laws rather than from a host inner-product space.

86.1 Carrier definition

Definition 86.1 (Inner-product BHist carrier)**.** *An inner-product* BHist *carrier is a displayed package over a source* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$*. It consists of the following rows:*

1. *a vector carrier* $V : \mathsf{BHist} \to \mathsf{Prop}$*;*
2. *a vector classifier* $\sim_V$ *on carried vector histories;*
3. *the vector addition, vector zero, vector negation, and scalar-action rows inherited from the displayed* $\mathsf{VecSpace}^{\uparrow}$ *source;*
4. *a scalar endpoint classifier* $\sim_{\mathbb{K}}$ *supplied by the scalar side of the* $\mathsf{VecSpace}^{\uparrow}$ *source;*
5. *an inner-product endpoint row* $\langle -, - \rangle_{\mathsf{InnerProduct}^{\uparrow}} : V \times V \to \mathsf{BHist}$ *whose values are carried by the scalar endpoint source;*
6. *a conjugate-symmetry row identifying* $\langle x, y \rangle_{\mathsf{InnerProduct}^{\uparrow}}$ *with the conjugate scalar endpoint of* $\langle y, x \rangle_{\mathsf{InnerProduct}^{\uparrow}}$*;*
7. *sesquilinear rows for vector addition in the additive argument and scalar-action compatibility in the scalar argument, all classified by* $\sim_{\mathbb{K}}$*;*
8. *a positive-definite row saying that* $\langle x, x \rangle_{\mathsf{InnerProduct}^{\uparrow}}$ *is nonnegative and that its scalar zero classification holds exactly when* $x \sim_V 0_V$*;*

9. *a visible ledger recording the* $\mathsf{VecSpace}^{\uparrow}$ *source row, every scalar endpoint classifier used by the form, the operation rows consumed by the sesquilinear clauses, and the positive-definite witness row.*

The carrier is therefore a named BHist *display, not a host inner-product space. Downstream certificates read vector endpoints, scalar endpoints, and form values only through these displayed rows.*

Lean4 (definition) `BEDC.Derived.InnerProductUp.InnerProductBHistCarrier`

Definition 86.2 (Inner-product BHist classifier)**.** *Two carried inner-product rows are classified exactly when all visible source rows match. Their vector carriers and vector classifiers agree through the same* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *source; their scalar endpoint classifiers agree on every displayed value of* $\langle -, - \rangle_{\mathsf{InnerProduct}^{\uparrow}}$*; and the addition, zero, negation, scalar-action, conjugate-symmetry, sesquilinear, positive-definite, and ledger rows match componentwise by* hsame*.*

Thus the classifier does not identify two inner-product packages by an external isometry. It identifies them only by equality of the displayed BEDC rows that a later $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{RiemannianMetric}^{\uparrow}$*,* $\mathsf{Fourier}^{\uparrow}$*, or* $\mathsf{RootSystem}^{\uparrow}$ *certificate is allowed to consume.*

Lean4 ✓ `BEDC.Derived.InnerProductUp.TasteGate.InnerProductTasteGate single carrier alignment`

86.2 The certificate

Definition 86.3 (InnerProduct BHist source data)**.** *An* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source consists of a displayed* $\mathsf{VecSpace}^{\uparrow}$ *BHist carrier* $C_V : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_V$*, scalar data carried by the scalar source retained from* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$*, and a binary row*

$$\langle -, - \rangle : C_V \times C_V \to C_{\mathsf{scal}}$$

whose endpoints are scalar histories. The source also displays conjugation on the scalar side, the scalar classifier $\sim_{\mathsf{scal}}$*, and the certificate rows asserting conjugate symmetry, positivity, linearity in each admitted argument, and transport through* $\sim_V$ *and* $\sim_{\mathsf{scal}}$*.*

All rows are BHist rows. Thus the form does not land in a host field or a quotient object; it lands in the scalar histories already exposed by the field and vector-space certificates.

Theorem 86.4 (InnerProduct BHist carrier row)**.** *For an* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, the public carrier row is exactly the* $\mathsf{VecSpace}^{\uparrow}$ *BHist carrier* C_V *together with the displayed scalar-valued form row* $\langle -, - \rangle$*. No further point carrier is introduced by the inner-product certificate.*

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductBHistCarrierRow`

Proof. Project the vector-space component of Definition 86.3. It already supplies the carried predicate C_V, the classifier $\sim_V$, vector addition, vector zero, vector negation, scalar action, and scalar source.

Project the form component. Its domain is the displayed pair carrier $C_V \times C_V$, and its codomain is the retained scalar history carrier C_{scal}. These two projections are exactly the carrier row and the additional scalar-valued row named by $\mathsf{InnerProduct}^{\uparrow}$. □

Theorem 86.5 (InnerProduct symmetry and positivity row)**.** *For an* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, conjugate symmetry and positivity are certificate fields over the retained scalar classifier. For carried histories* $x, y : C_V$*, the symmetry row classifies*

$$\langle x, y \rangle \sim_{\mathsf{scal}} \overline{\langle y, x \rangle},$$

and the positivity row is a displayed scalar-history witness for $\langle x, x\rangle$ together with the exact zero-classification clause. These rows do not use host order, quotient equality, or an external positivity predicate.

Proof. Unpack the symmetry field of Definition 86.3. It compares the scalar endpoint $\langle x, y\rangle$ with the conjugated scalar endpoint $\overline{\langle y, x\rangle}$ using $\sim_{\mathsf{scal}}$.

Unpack the positivity field. Its witness is recorded as scalar-history data for the diagonal endpoint $\langle x, x\rangle$, and its zero clause is stated by the same retained scalar classifier. Since both fields are projections of the displayed source, no host order or quotient equality is added. □

Theorem 86.6 (InnerProduct VecSpace linearity row)**.** *For an* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, additivity and scalar compatibility in each admitted argument are read through the* $\mathsf{VecSpace}^{\uparrow}$ *addition row, scalar-action row, and classifier transport. Thus the clauses comparing*

$$\langle x +_V y, z\rangle \quad \textit{with} \quad \langle x, z\rangle +_{\mathsf{scal}} \langle y, z\rangle$$

and

$$\langle a \cdot_V x, y\rangle \quad \textit{with} \quad a \cdot_{\mathsf{scal}} \langle x, y\rangle$$

are scalar-classifier rows over the same carried inputs, with the conjugate argument handled by the displayed conjugation row when the source uses that side as the conjugate-linear argument.

Proof. Read addition and scalar action from the embedded $\mathsf{VecSpace}^{\uparrow}$ row. Closure places $x +_V y$ and $a \cdot_V x$ back in C_V, so the inner-product row can be evaluated at the displayed endpoints.

Read scalar addition, scalar multiplication, and conjugation from the retained scalar source. The linearity fields compare the resulting scalar endpoints by $\sim_{\mathsf{scal}}$.

If representatives of the vector or scalar inputs are changed within their classifiers, use the transport fields of the vector-space source and the scalar source before applying the displayed linearity row. The conclusion is therefore a row over the existing $\mathsf{VecSpace}^{\uparrow}$ operations, not a separate algebraic structure. □

Theorem 86.7 (InnerProduct classifier transport row)**.** *For an* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, if $x \sim_V x'$ and $y \sim_V y'$ for carried histories $x, y, x', y' : C_V$, then the scalar endpoints of the inner-product row transport through the retained scalar classifier:*

$$\langle x, y\rangle \sim_{\mathsf{scal}} \langle x', y'\rangle.$$

Proof. Project the pair-transport field from Definition 86.3. Its hypotheses are exactly the two vector classifier witnesses $x \sim_V x'$ and $y \sim_V y'$.

The field returns a scalar classifier witness between the two displayed inner-product endpoints. Since the codomain classifier is $\sim_{\mathsf{scal}}$, the transported endpoint remains in the retained scalar source. □

Theorem 86.8 (InnerProduct Hilbert boundary row)**.** *A* $\mathsf{Hilbert}^{\uparrow}$ *certificate may consume from an* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source only the carrier C_V, classifier $\sim_V$, linearity rows, conjugate symmetry row, positivity row, scalar endpoint row, classifier transport row, and the derived orthogonality row of Definition 86.13 together with its zero-left, zero-right, and transport laws. Completeness, Cauchy-limit selection, and Banach metric closure are not part of the* $\mathsf{InnerProduct}^{\uparrow}$ *boundary.*

Proof. The Hilbert carrier definition in Definition 91.1 asks the inner-product side for vector carrier sharing, the scalar-valued form, linearity, symmetry, positivity, and transport. These are precisely the rows displayed in Definition 86.3.

The same Hilbert definition obtains completeness from its Banach row. The orthogonality predicate is obtained from the scalar-zero classification of the inner-product endpoint, and its zero and transport laws are supplied by Definition 86.14, Definition 86.15, and Definition 86.16. Hence no completeness or Cauchy-limit field can be projected from the inner-product source, and the boundary available to $\mathsf{Hilbert}^{\uparrow}$ is exactly the displayed inner-product surface. □

Theorem 86.9 (InnerProduct root certificate row)**.** *The carrier row of Definition 86.4, the symmetry and positivity row of Definition 86.5, the linearity row of Definition 86.6, the classifier transport row of Definition 86.7, and the Hilbert boundary row of Definition 86.8 assemble the paper-side* $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ *obligation surface.*

Proof. Use Definition 86.3 for the source package. Its first projection gives the vector-space carrier and scalar source, and Definition 86.4 records that no second point carrier is present.

Use Definition 86.5 and Definition 86.6 for the algebraic rows of the form. These rows are all scalar-history comparisons under $\sim_{\mathsf{scal}}$.

Use Definition 86.7 for representative stability. Use Definition 86.8 for the downstream boundary consumed by $\mathsf{Hilbert}^{\uparrow}$.

Together these projections supply carrier, classifier, exactness, stability, and ledger provenance for the $\mathsf{InnerProduct}^{\uparrow}$ naming certificate, with bridge status left unclaimed. □

Theorem 86.10 (Inner-product carrier transport obligation)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. If $x \sim_V x'$ and $y \sim_V y'$, then the vector carrier rows, scalar endpoint rows, inner-product endpoint row, conjugate-symmetry row, sesquilinear rows, positive-definite row, and ledger entries of I transport componentwise by* hsame *from (x, y) to (x', y').*

Proof. Start with the $\mathsf{VecSpace}^{\uparrow}$ source row inside I. Its carrier and classifier transport move x to x' and y to y' while preserving the displayed vector addition, zero, negation, and scalar-action rows.

Next apply the scalar endpoint classifier row to the two displayed values $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle x', y'\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The inner-product classifier of Definition 86.2 requires these scalar endpoints to be classified by the same $\sim_{\mathbb{K}}$ row, so the form endpoint transports with the vector representatives.

Finally transport the proof rows attached to that endpoint. Conjugate symmetry, both sesquilinear rows, positivity, and ledger provenance are stored as displayed rows in the carrier. Since classification is componentwise hsame on those rows, each witness is copied to the transported package without introducing any external inner-product-space equality. □

Theorem 86.11 (Inner-product sesquilinear source obligation)**.** *Every sesquilinear clause of an inner-product* BHist *carrier is sourced from its displayed* $\mathsf{VecSpace}^{\uparrow}$ *and scalar rows. Additivity uses the vector addition row and the scalar addition classifier from the scalar endpoint source; scalar-action compatibility uses the displayed scalar action and the scalar multiplication, conjugation, and endpoint classifier rows from the same source ledger.*

Proof. Unpack the carrier of Definition 86.1. The only vector operations available to the form are the $\mathsf{VecSpace}^{\uparrow}$ rows listed in the carrier: addition, zero, negation, and scalar action. The additivity clause is therefore stated over the displayed vector addition row and is classified on the scalar side by $\sim_{\mathbb{K}}$.

For scalar compatibility, the scalar used in the vector action is the same scalar endpoint read by the $\mathsf{VecSpace}^{\uparrow}$ source. The scalar multiplication and conjugation rows used to express sesquilinearity are ledger entries over that scalar source, and the resulting equality of form values is again classified by $\sim_{\mathbb{K}}$.

The ledger records each operation row consumed in these clauses. Hence a certificate consumer can reconstruct the additivity and scalar-action compatibility obligations from displayed BEDC rows alone, with no appeal to a host field or a host vector space. □

Theorem 86.12 (Inner-product positive-definite obligation)**.** *For every carried vector endpoint x in an inner-product* BHist *carrier, the self-value $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ is classified as a nonnegative scalar endpoint. Moreover the self-value is classified as the scalar zero endpoint if and only if x is classified by $\sim_V$ with the displayed vector zero row 0_V.*

Proof. Read the self-value row from Definition 86.1. Its positive-definite entry contains the nonnegative scalar endpoint witness for $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$, and the scalar classifier used for that witness is the scalar endpoint classifier $\sim_{\mathbb{K}}$ already stored in the carrier.

For the zero direction, assume the self-value is classified as the scalar zero endpoint. The positive-definite row includes the exactness clause converting that scalar classification into the vector classifier equality $x \sim_V 0_V$.

For the converse direction, assume $x \sim_V 0_V$. The carrier-transport obligation transports the self-value row along this vector classification, and the positive-definite row at the displayed zero endpoint supplies the scalar zero classification. Combining these two displayed rows gives the required scalar endpoint classification for $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. □

Definition 86.13 (InnerProduct orthogonality row)**.** *For carried* BHist *vector endpoints x, y from Definition 86.1, write $x \perp_I y$ exactly when the scalar endpoint $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ is classified by the retained scalar classifier with the displayed scalar zero endpoint:*

$$x \perp_I y \quad \Longleftrightarrow \quad \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

This is a row of the $\mathsf{InnerProduct}^{\uparrow}$ *carrier, not a host Hilbert-space orthogonality relation.*

Theorem 86.14 (InnerProduct zero-left orthogonality row)**.** *For every carried vector endpoint x in an inner-product* BHist *carrier, the displayed vector zero row is orthogonal to x:*

$$0_V \perp_I x.$$

Proof. Read the displayed vector zero endpoint 0_V from the embedded $\mathsf{VecSpace}^{\uparrow}$ row and evaluate the inner-product endpoint row at $(0_V, x)$.

The sesquilinearity row for scalar action compares $\langle 0_{\mathbb{K}} \cdot_V 0_V, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ with $0_{\mathbb{K}} \cdot_{\mathbb{K}} \langle 0_V, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The vector zero row classifies $0_{\mathbb{K}} \cdot_V 0_V$ with 0_V, and the scalar zero law classifies the displayed scalar product with $0_{\mathbb{K}}$.

Transport these two comparisons through the scalar endpoint classifier. By Definition 86.13, the resulting scalar-zero classification is exactly $0_V \perp_I x$. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal zero left`

Theorem 86.15 (InnerProduct zero-right orthogonality row)**.** *For every carried vector endpoint x in an inner-product* BHist *carrier, x is orthogonal to the displayed vector zero row:*

$$x \perp_I 0_V.$$

Proof. Apply the conjugate-symmetry row to compare $\langle x, 0_V\rangle_{\mathsf{InnerProduct}^\uparrow}$ with the conjugate scalar endpoint of $\langle 0_V, x\rangle_{\mathsf{InnerProduct}^\uparrow}$.

By Definition 86.14, $\langle 0_V, x\rangle_{\mathsf{InnerProduct}^\uparrow}$ is classified with $0_{\mathbb{K}}$. The scalar zero endpoint is fixed by the displayed conjugation row, so the conjugated endpoint is also classified with $0_{\mathbb{K}}$.

Use transitivity of the retained scalar classifier to combine the symmetry comparison with this scalar-zero classification. The definition of $\perp_I$ then gives $x \perp_I 0_V$. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal zero right`

Theorem 86.16 (InnerProduct orthogonality transport row)**.** *For carried vector endpoints x, y, x', y' in an inner-product* BHist *carrier, if $x \sim_V x'$, $y \sim_V y'$, and $x \perp_I y$, then $x' \perp_I y'$.*

Proof. Unpack $x \perp_I y$ using Definition 86.13. This gives a scalar classifier witness from $\langle x, y\rangle_{\mathsf{InnerProduct}^\uparrow}$ to the displayed scalar zero endpoint.

Apply Definition 86.7 to the vector classifier witnesses $x \sim_V x'$ and $y \sim_V y'$. It supplies a scalar classifier comparison between $\langle x, y\rangle_{\mathsf{InnerProduct}^\uparrow}$ and $\langle x', y'\rangle_{\mathsf{InnerProduct}^\uparrow}$.

Combine that comparison with the scalar-zero witness by transitivity of $\sim_{\mathbb{K}}$. The resulting classification of $\langle x', y'\rangle_{\mathsf{InnerProduct}^\uparrow}$ with $0_{\mathbb{K}}$ is precisely $x' \perp_I y'$ by Definition 86.13. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal transport`

Theorem 86.17 (Inner-product NameCert obligation surface)**.** *The carrier of Definition 86.1, the classifier of Definition 86.2, carrier transport, sesquilinear source tracking, positive definiteness, and the visible ledger assemble the paper-side* $\mathsf{NameCert}_{\mathsf{InnerProduct}^\uparrow}$ *obligation surface. Its ten roadmap fields are:*

1. *traditional target: a vector space equipped with a positive-definite sesquilinear form;*
2. *BEDC source:* BHist*;*
3. *carrier choice: the displayed vector carrier V over* $\mathsf{NameCert}_{\mathsf{VecSpace}^\uparrow}$*;*
4. *classifier choice: componentwise agreement of the vector, scalar, form, law, and ledger rows;*
5. *stability obligations: carrier transport, scalar endpoint transport, conjugate symmetry, sesquilinearity, and positive-definite transport;*
6. *ledger: the* $\mathsf{VecSpace}^\uparrow$ *source, scalar endpoint classifier, operation rows, form row, law rows, and positivity witness;*
7. *core theorem list: Definition 86.10, Definition 86.11, and Definition 86.12; the orthogonality rows are Definition 86.14, Definition 86.15, and Definition 86.16;*
8. *exactness predicate: scalar zero self-value exactly matches vector-classifier equality to 0_V;*
9. *standard bridge: none;*
10. *status: paper obligation surface.*

No standard Hilbert-space bridge is claimed by this surface.

Proof. Use Definition 86.1 for the public carrier field. It names the vector carrier, vector classifier, scalar endpoint classifier, inner-product row, law rows, positive-definite row, and ledger over a single $\mathsf{VecSpace}^\uparrow$ source.

Use Definition 86.2 for the classifier field. It requires componentwise agreement of exactly the rows exposed by the carrier, so downstream consumers receive the same vector, scalar, form, and ledger surface after classification.

Use Definition 86.10 for the stability field, Definition 86.11 for the operation-source field, and Definition 86.12 for exactness of the self-value row. Use Definition 86.13, Definition 86.14, Definition 86.15, and Definition 86.16 for the orthogonality surface consumed by downstream Hilbert rows.

The remaining roadmap fields are read directly from the same displayed ledger: the target is the named positive-definite sesquilinear form, the BEDC source is BHist, and the bridge field is explicitly empty. Thus $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ is available as a paper-side obligation surface without importing host Hilbert-space structure. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingleton semanticNameCert`

Theorem 86.18 (InnerProduct public export surface)**.** *The carrier of Definition 86.1, the classifier of Definition 86.2, carrier transport, the sesquilinear source, positive definiteness, zero-left and zero-right orthogonality, orthogonality transport, and the root dependency rows export exactly the public* $\mathsf{InnerProduct}^{\uparrow}$ *BHist surface: the* $\mathsf{VecSpace}^{\uparrow}$ *carrier and classifier, the scalar endpoint classifier, the inner-product row, the law rows, the orthogonality rows, and the visible ledger. The export withholds completeness, Banach metric closure, ambient topology, a standard Hilbert bridge, and host Hilbert-space equality.*

Proof. Apply Definition 86.17. It assembles the carrier, classifier, carrier transport, sesquilinear source tracking, positive-definite exactness, visible ledger, and orthogonality rows into the paper-side $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ obligation surface.

Apply Definition 86.19. Its tuple is the dependency row exported by $\mathsf{InnerProduct}^{\uparrow}$ to later certificates, and it lists only the displayed vector, scalar, form, law, transport, and ledger rows.

Apply Definition 86.20 and Definition 86.21. These identify the exposed $\mathsf{VecSpace}^{\uparrow}$ carrier and classifier, scalar endpoint classifier, scalar operation rows, inner-product row, and inner-product law rows.

Apply Definition 86.22. It states that Hilbert consumption obtains vector, scalar, form, law, and ledger data from this root surface while completeness and Cauchy-limit selection enter only through a separate Banach and completeness row. Thus the public export is the displayed $\mathsf{InnerProduct}^{\uparrow}$ BHist surface, with no completeness, Banach metric closure, ambient topology, standard Hilbert bridge, or host Hilbert-space equality included. □

Definition 86.19 (InnerProduct root dependency surface)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *root dependency surface is the displayed tuple consisting of the* $\mathsf{VecSpace}^{\uparrow}$ *carrier and classifier of Definition 86.1, the retained scalar endpoint source, the inner-product endpoint row, the conjugate-symmetry row, the sesquilinear rows, the positive-definite exactness row, the classifier transport row, and the visible ledger. It is the exact dependency package exported by* $\mathsf{InnerProduct}^{\uparrow}$ *to later certificates: every projected component is one of these displayed rows, and no completeness, Banach metric, ambient topology, or host Hilbert-space row is part of the tuple.*

Theorem 86.20 (InnerProduct root vector-scalar exposure)**.** *Every consumer of the root surface of Definition 86.19 can project exactly the vector carrier and classifier, the vector addition, zero, negation, and scalar action rows, the scalar endpoint classifier, and the scalar operation rows already named in Definition 86.1 and Definition 86.3. No separate vector carrier or scalar endpoint source is exposed.*

Proof. Project the vector portion of the root surface. By Definition 86.4, this projection is the displayed $\mathsf{VecSpace}^{\uparrow}$ BHist carrier C_V together with its classifier $\sim_V$, vector operations, scalar action row, and scalar-valued form row.

Project the scalar portion from the same source tuple. The scalar endpoint classifier and scalar operation rows are the retained rows of Definition 86.3; if a representative changes, apply Definition 86.10 to transport the displayed form and operation rows through the exposed classifiers.

These projections exhaust the vector-scalar part of Definition 86.19. Since the definition lists only displayed carrier, classifier, operation, scalar endpoint, form, law, and ledger rows, there is no second vector carrier or scalar endpoint source to project. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductRoot vecspace scalar exposure`

Theorem 86.21 (InnerProduct root form-law exposure)**.** *For the root surface of Definition 86.19, conjugate symmetry, sesquilinearity, positive definiteness, and form endpoint transport are exactly the law rows exposed by the surface. A consumer may read those rows, and only those rows, as the inner-product law interface.*

Proof. The conjugate-symmetry and positivity rows are the rows identified by Definition 86.5. They compare scalar histories under the retained scalar classifier and do not introduce a host scalar field.

The additive and scalar-action compatibility rows are the sesquilinear rows identified by Definition 86.6. Their inputs are the exposed vector operations and scalar endpoint operations from the root surface.

Positive-definite exactness is the row supplied by Definition 86.12, and endpoint transport is the row supplied by Definition 86.7. Together these rows are precisely the law entries named in Definition 86.19; all remaining entries of the root surface are carrier, source, classifier, or ledger entries. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductRoot form law exposure`

Theorem 86.22 (InnerProduct root Hilbert consumption surface)**.** *A* $\mathsf{Hilbert}^{\uparrow}$ *dependency may consume the root surface of Definition 86.19 together with its separate Banach and completeness row. A* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{RiemannianMetric}^{\uparrow}$*, or* $\mathsf{RootSystem}^{\uparrow}$ *dependency may consume only the vector, scalar, form, law, and ledger rows of the same root surface.*

Proof. For $\mathsf{Hilbert}^{\uparrow}$, apply Definition 86.8. It allows the Hilbert certificate to read the carrier, classifier, scalar-valued form, linearity, conjugate symmetry, positivity, scalar endpoint, and transport rows from $\mathsf{InnerProduct}^{\uparrow}$.

The vector and scalar projections used by this boundary are exactly the ones identified by Definition 86.20. The form and law projections are exactly the ones identified by Definition 86.21.

Completeness and Cauchy-limit selection are not projected from $\mathsf{InnerProduct}^{\uparrow}$; they must enter through the separate Banach and completeness row of the Hilbert dependency. The non-Hilbert consumers named in the statement therefore receive only the root surface rows for vector, scalar, form, law, and ledger data, with no Banach or completeness projection. □

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductRootHilbert consumption surface`

Definition 86.23 (InnerProduct root threshold source)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *root threshold source is the* BHist*-indexed package obtained from Definition 86.19 by exposing precisely the rows needed before any* $\mathsf{Hilbert}^{\uparrow}$ *Banach or completeness data is attached. It consists of the displayed* $\mathsf{VecSpace}^{\uparrow}$ *carrier and classifier, the scalar endpoint source retained from the vector-space and field side, the scalar-valued form row*

$$\langle -, - \rangle_{\mathsf{InnerProduct}^{\uparrow}} : C_V \times C_V \to C_{\mathsf{scal}},$$

the conjugate-symmetry row, the sesquilinearity rows, the positive-definite exactness row, classifier transport for the form endpoints, the orthogonality row of Definition 86.13, and the visible ledger recording which vector, scalar, form, law, transport, and orthogonality rows are present.

This source is a root threshold because it is complete for the inner-product-side rows consumed downstream while excluding Banach completeness, Cauchy-limit selection, ambient topology, and any host Hilbert-space structure.

Theorem 86.24 (InnerProduct root threshold form exactness)**.** *The root threshold source of Definition 86.23 exposes the full inner-product form exactness surface. Conjugate symmetry, sesquilinearity, form endpoint transport, and positive-definite zero classification are the law rows of the root surface, and the positive-definite row identifies* $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ *with the scalar zero endpoint exactly when* x *is classified with the displayed vector zero.*

Proof. Read the form and law entries from Definition 86.21. This theorem identifies conjugate symmetry, sesquilinearity, positive definiteness, and form endpoint transport as the exposed law interface of Definition 86.19.

The threshold source of Definition 86.23 is the same root surface with the downstream rows named explicitly, so these form and transport rows are present in the threshold source.

For the exact zero criterion, apply Definition 86.12. It supplies the positive-definite row saying that the self-form scalar endpoint is classified as zero precisely in the displayed vector-zero case. This is the positive-definite exactness component listed in the threshold source. □

Theorem 86.25 (InnerProduct root threshold orthogonality consumption)**.** *When* $\mathsf{Hilbert}^{\uparrow}$ *consumes the inner-product side of the root threshold source, its geometric input is exhausted by the scalar-valued form rows and the orthogonality row derived from them. Its Banach metric, completeness, and Cauchy-limit rows come from the separate* $\mathsf{Banach}^{\uparrow}$ *dependency, not from the* $\mathsf{InnerProduct}^{\uparrow}$ *threshold source.*

Proof. Apply Definition 86.22. It states that $\mathsf{Hilbert}^{\uparrow}$ may read the root inner-product surface together with a separate Banach and completeness row.

Inside the root threshold source, the Hilbert-facing geometric rows are the scalar-valued form, its law rows, classifier transport, and the orthogonality predicate of Definition 86.13. The orthogonality predicate is defined by scalar-zero classification of the form, so no independent host orthogonality relation is consumed.

The same consumption theorem separates completeness and Cauchy-limit selection from $\mathsf{InnerProduct}^{\uparrow}$. Those rows must be supplied by the Banach dependency, so they are outside the threshold source of Definition 86.23. □

Theorem 86.26 (InnerProduct root threshold downstream coverage)**.** *The downstream interfaces* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{RiemannianMetric}^{\uparrow}$*, and* $\mathsf{RootSystem}^{\uparrow}$ *receive the inner-product rows they are allowed to consume from the single threshold source of Definition 86.23. The named threshold rows are the* $\mathsf{VecSpace}^{\uparrow}$ *carrier and classifier, scalar endpoint source, form row, conjugate-symmetry row, sesquilinearity rows, positive-definite exactness row, classifier-transport row, orthogonality row, and visible ledger. No downstream consumer receives Banach completeness, Cauchy-limit selection, ambient topology, or a host Hilbert-space row from this source.*

Proof. For $\mathsf{Hilbert}^{\uparrow}$, use Definition 86.25. Its inner-product-side input is the form, law, transport, orthogonality, and ledger surface of the threshold source, while Banach and completeness rows come from the separate dependency.

For $\mathsf{Fourier}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, and $\mathsf{RootSystem}^{\uparrow}$, apply Definition 86.22 to the non-Hilbert consumers named there. They may consume only the vector, scalar, form, law, and ledger rows of the same root surface.

Finally use the definition of Definition 86.23 to read those rows as one package. Since the definition lists no Banach metric, completeness field, Cauchy-limit selector, ambient topology, or host Hilbert-space row, none of the four downstream interfaces can receive such a row from this threshold source. □

Definition 86.27 (InnerProduct norm-metric seed source)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *norm-metric seed source is the* BHist *vector carrier and classifier of Definition 86.3 together with the scalar-valued diagonal endpoint row*

$$\|x\|_I^2 := \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Its scalar-zero classifier row is the positive-definite exactness row of Definition 86.12: the diagonal endpoint is classified with the displayed scalar zero exactly when the vector endpoint is classified with the displayed vector zero. Its threshold law rows are the form endpoint transport, classifier transport, scalar endpoint source, and visible ledger rows exposed by Definition 86.23.

This seed is therefore a BHist-rooted norm-squared interface. It names the diagonal scalar endpoint and the zero exactness row needed by later $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Hilbert}^{\uparrow}$ *consumers, while leaving ordinary norm extraction, metric completion, Cauchy-limit selection, topology, and host Hilbert carrier data outside the source.*

Theorem 86.28 (InnerProduct norm-squared carrier row)**.** *For every carried vector endpoint* x *of the* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, the norm-metric seed source carries the scalar endpoint* $\|x\|_I^2 = \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. *If* $x \sim_V x'$, *then the two scalar endpoints* $\|x\|_I^2$ *and* $\|x'\|_I^2$ *are classified by the retained scalar classifier.*

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductNormSquared carrier row`

Proof. Unfold Definition 86.27. The carried vector endpoint x lies in the vector carrier of Definition 86.3, and the diagonal form row evaluates at (x, x) with codomain in the retained scalar endpoint source.

For transport, use the form endpoint transport row listed in Definition 86.24. Apply it to the two vector classifier witnesses $x \sim_V x'$ and $x \sim_V x'$, once for each argument of the diagonal pair.

The resulting scalar-classifier witness compares $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ with $\langle x', x'\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. Repack these endpoints through the definition of $\| - \|_I^2$ in the seed source. □

Theorem 86.29 (InnerProduct norm zero classifier exactness)**.** *For every carried vector endpoint* x *of the* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, the scalar endpoint* $\|x\|_I^2$ *is classified with the displayed scalar zero endpoint exactly when* x *is classified by* $\sim_V$ *with the displayed vector zero endpoint* 0_V.

Proof. Unfold the norm-metric seed source so that $\|x\|_I^2$ is the diagonal form endpoint $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$.

In the forward direction, assume that this scalar endpoint is classified with the displayed scalar zero. Apply Definition 86.12 to convert the scalar-zero classification into the vector classifier witness $x \sim_V 0_V$.

In the reverse direction, assume $x \sim_V 0_V$. The same positive-definite obligation supplies the converse scalar-zero classification for the diagonal endpoint. Repack that scalar witness as the zero classification of $\|x\|_I^2$ in Definition 86.27. □

Theorem 86.30 (InnerProduct norm-metric seed downstream boundary)**.** *The downstream interfaces* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Hilbert}^{\uparrow}$ *may consume from the* $\mathsf{InnerProduct}^{\uparrow}$ *side only the norm-squared seed of Definition 86.27 together with the existing inner-product threshold rows of Definition 86.23. They do not receive Banach completeness, a Cauchy-limit selector, topology, or a host Hilbert carrier from this seed.*

Proof. For the norm-squared endpoint, use Definition 86.28. It supplies exactly the scalar diagonal endpoint and its classifier transport along the vector classifier.

For zero exactness, use Definition 86.29. It identifies scalar-zero classification of the diagonal endpoint with vector-zero classification and adds no ordinary norm value or completion row.

Finally compare the seed with the root threshold source. By Definition 86.23, the remaining consumable inner-product rows are the vector carrier and classifier, scalar endpoint source, form law rows, positive-definite exactness, classifier transport, orthogonality row, and ledger. Since neither this definition nor Definition 86.27 lists Banach completeness, Cauchy-limit selection, topology, or a host Hilbert carrier, those rows cannot be consumed from the $\mathsf{InnerProduct}^{\uparrow}$ seed boundary. □

Theorem 86.31 (InnerProduct norm-squared scalar-action factor row)**.** *For every carried scalar endpoint* r *and carried vector endpoint* x *of the* $\mathsf{InnerProduct}^{\uparrow}$ *BHist source, the norm-metric seed source carries the diagonal endpoint* $\|r \cdot_V x\|_I^2$*, and the retained scalar classifier classifies it with the scalar self-product acting on the original diagonal endpoint:*

$$\|r \cdot_V x\|_I^2 \sim_{\mathbb{K}} (r \cdot_{\mathbb{K}} \overline{r}) \cdot_{\mathbb{K}} \|x\|_I^2.$$

Here $\overline{r}$ *is the displayed scalar conjugate endpoint retained by the* $\mathsf{InnerProduct}^{\uparrow}$ *scalar source.*

Proof. Unfold Definition 86.27. The endpoint $\|r \cdot_V x\|_I^2$ is the diagonal form value

$$\langle r \cdot_V x, r \cdot_V x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The vector-space component of Definition 86.3 carries $r \cdot_V x$, since scalar-action closure is one of the module stability rows retained by Definition 47.19 and Definition 46.18. Hence Definition 86.28 applies to the carried vector endpoint $r \cdot_V x$.

Apply the scalar-action compatibility component of Definition 86.6 in the scalar-linear argument. It gives a retained scalar-classifier comparison

$$\langle r \cdot_V x, r \cdot_V x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} r \cdot_{\mathbb{K}} \langle x, r \cdot_V x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the conjugate-linear scalar-action clause from the same theorem in the other argument. The displayed conjugation row of the scalar source supplies the comparison

$$\langle x, r \cdot_V x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} \overline{r} \cdot_{\mathbb{K}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Scalar multiplication congruence in the retained scalar source, read through Definition 45.201, transports this second comparison under left multiplication by r and gives

$$r \cdot_{\mathbb{K}} \langle x, r \cdot_V x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} r \cdot_{\mathbb{K}} (\overline{r} \cdot_{\mathbb{K}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

The scalar associativity row from the same scalar source classifies the last endpoint with

$$(r \cdot_{\mathbb{K}} \overline{r}) \cdot_{\mathbb{K}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}},$$

using classifier symmetry from Definition 9.7 if the stored associativity row is oriented oppositely.

Finally repack $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ as $\|x\|_I^2$ by Definition 86.27. Successive applications of Definition 9.8 compose the first scalar-action comparison, the transported conjugate-linear comparison, the scalar associativity comparison, and this repacking into the displayed retained scalar-classifier row. □

86.3 Parallelogram norm seed

Definition 86.32 (InnerProduct parallelogram norm seed source)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *parallelogram norm seed source is the* BHist *ledger extending Definition 86.27 by the following carried rows. It carries vector endpoints* $x, y : C_V$*, the vector addition endpoint* $x +_V y$*, the vector difference endpoint* $x -_V y$*, and the four scalar diagonal endpoints*

$$\|x\|_I^2, \qquad \|y\|_I^2, \qquad \|x +_V y\|_I^2, \qquad \|x -_V y\|_I^2.$$

Here $x -_V y$ *denotes* $x +_V (-_V y)$ *using the displayed additive inverse row of the embedded* $\mathsf{VecSpace}^{\uparrow}$ *source, and every norm-squared endpoint is the diagonal scalar endpoint named by Definition 86.27.*

The ledger also retains the scalar source rows for addition, additive inverse, doubling by $2_{\mathbb{K}}$*, scalar multiplication, conjugation, and scalar classifier transport from the field and vector-space side of Definition 86.3. Thus the seed records exactly the algebraic data needed to compare the two diagonal sums appearing in the parallelogram identity.*

Theorem 86.33 (InnerProduct parallelogram identity row)**.** *For carried vector endpoints* $x, y : C_V$ *in the Definition 86.32, the retained scalar classifier classifies the parallelogram endpoints*

$$\|x +_V y\|_I^2 +_{\mathbb{K}} \|x -_V y\|_I^2 \sim_{\mathbb{K}} 2_{\mathbb{K}} \cdot_{\mathbb{K}} \|x\|_I^2 +_{\mathbb{K}} 2_{\mathbb{K}} \cdot_{\mathbb{K}} \|y\|_I^2.$$

The row is a scalar-history comparison inside the same seed; it does not introduce an ordinary norm, a metric completion, or a host inner-product space.

Proof. Unfold the parallelogram seed source. The vector-space rows carry $x +_V y$ and $x -_V y$, and the norm-metric seed identifies each displayed norm square with a diagonal form endpoint.

Apply Definition 86.6 to expand $\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The result is classified with the scalar sum of the four form values

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the same linearity row to the diagonal value $\langle x -_V y, x -_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The additive inverse and scalar-action rows from the retained $\mathsf{VecSpace}^{\uparrow}$ and scalar sources transport the two mixed terms to their scalar additive inverses, giving the classified scalar sum

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} (-_{\mathbb{K}}\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} (-_{\mathbb{K}}\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Now add the two expanded rows in the retained scalar classifier. Scalar addition congruence and transitivity compose the two comparisons, while the additive inverse laws cancel the paired mixed terms $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The conjugate-symmetry and positivity row of Definition 86.5 keeps both mixed endpoints inside the same scalar classifier ledger during this cancellation.

After the cancellations, the remaining scalar endpoint is the sum of two copies of $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and two copies of $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The scalar doubling rows identify those paired sums with $2_{\mathbb{K}} \cdot_{\mathbb{K}} \|x\|_I^2$ and $2_{\mathbb{K}} \cdot_{\mathbb{K}} \|y\|_I^2$. A final application of Definition 9.8 composes the expansion, cancellation, and doubling witnesses into the displayed classifier row. □

Theorem 86.34 (InnerProduct norm seed Hilbert consumer boundary)**.** *The* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{Norm}^{\uparrow}$ *consumers may read Definition 86.33 only through the norm-metric seed of Definition 86.27 together with the root threshold source of Definition 86.23. The consumer boundary contains no Banach completeness row, topology row, host Hilbert carrier, or Cauchy-limit selector.*

Proof. The parallelogram row consumes the diagonal norm-squared endpoints from Definition 86.27. It consumes the vector addition, vector inverse, scalar addition, scalar inverse, scalar doubling, linearity, conjugate-symmetry, positivity, and transport rows through Definition 86.23.

These two sources are the whole input boundary for the row. The norm-metric seed supplies norm-squared endpoints and zero exactness; the root threshold source supplies the inner-product algebra and the visible ledger. Neither source lists Banach completeness, topology, a host Hilbert carrier, or a Cauchy-limit selector.

Therefore a $\mathsf{Hilbert}^{\uparrow}$ or $\mathsf{Norm}^{\uparrow}$ consumer may use the parallelogram identity as an algebraic scalar-history comparison, but any completion, topological, or host-space row must come from a separate downstream certificate. □

Theorem 86.35 (InnerProduct Pythagorean row)**.** *For carried vector endpoints $x, y : C_V$ in the Definition 86.32, if $x \perp_I y$, then the retained scalar classifier compares the diagonal endpoint of their sum with the sum of the two diagonal endpoints:*

$$\|x +_V y\|_I^2 \sim_{\mathbb{K}} \|x\|_I^2 +_{\mathbb{K}} \|y\|_I^2.$$

Proof. Assume $x \perp_I y$. By Definition 86.13, this is exactly the scalar-zero classification

$$\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Apply Definition 86.39 to obtain $y \perp_I x$. Unpacking this second orthogonality statement with Definition 86.13 gives

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Unfold Definition 86.32. The embedded $\mathsf{VecSpace}^{\uparrow}$ rows carry $x +_V y$, and Definition 86.28 identifies each displayed norm square with the corresponding diagonal form endpoint. Apply Definition 86.6 to expand $\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The resulting scalar classifier row compares it with

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The retained scalar source of Definition 86.3 is the $\mathsf{Field}^{\uparrow}$ scalar coordinate. By Definition 45.198, Definition 44.8, and Definition 43.26, its unchanged scalar classifier has addition congruence, additive zero laws, and classifier transitivity. Therefore the two mixed-term zero witnesses transport through scalar addition, so the displayed four-term scalar sum is classified with

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} 0_{\mathbb{K}} +_{\mathbb{K}} 0_{\mathbb{K}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Composing this simplification with the linearity expansion by Definition 9.8 yields

$$\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Finally repack the three diagonal endpoints as $\|x +_V y\|_I^2$, $\|x\|_I^2$, and $\|y\|_I^2$ using Definition 86.28. A final application of Definition 9.8 gives

$$\|x +_V y\|_I^2 \sim_{\mathbb{K}} \|x\|_I^2 +_{\mathbb{K}} \|y\|_I^2.$$

□

Theorem 86.36 (InnerProduct polarization difference row)**.** *For carried vector endpoints $x, y : C_V$ in the Definition 86.32, the retained scalar classifier compares the polarization-difference endpoints*

$$\|x +_V y\|_I^2 +_{\mathbb{K}} (-_{\mathbb{K}}\|x -_V y\|_I^2) \sim_{\mathbb{K}} 2_{\mathbb{K}} \cdot_{\mathbb{K}} (\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

Proof. Unfold Definition 86.32. The embedded $\mathsf{VecSpace}^{\uparrow}$ rows carry $x +_V y$ and $x -_V y$, and Definition 86.28 repacks each displayed norm square as the corresponding diagonal form endpoint.

Apply Definition 86.6 to expand $\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The resulting scalar classifier row compares $\|x +_V y\|_I^2$ with

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the same linearity row to $\langle x -_V y, x -_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The additive inverse and scalar-action rows retained in the seed transport the two mixed terms to their scalar additive inverses, so $\|x -_V y\|_I^2$ is classified with

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} (-_{\mathbb{K}}\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} (-_{\mathbb{K}}\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Now transport this second comparison through scalar additive inverse. The retained scalar source is the $\mathsf{Field}^{\uparrow}$ scalar coordinate of Definition 86.3. By Definition 45.198, Definition 44.8, and Definition 43.26, its unchanged scalar classifier has additive inverse congruence. Therefore

$$-_{\mathbb{K}}\|x -_V y\|_I^2 \sim_{\mathbb{K}} -_{\mathbb{K}}(\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} (-_{\mathbb{K}}\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} (-_{\mathbb{K}}\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

Apply Definition 43.33 repeatedly in the retained scalar ring, and read its additive reduct through Definition 43.59. The two double negatives collapse by Definition 42.34, so the right-hand scalar endpoint is further classified with

$$(-_{\mathbb{K}}\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} (-_{\mathbb{K}}\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

Add the expansion of $\|x +_V y\|_I^2$ and this negated expansion inside the retained scalar classifier. Addition congruence and Definition 9.8 compose the two comparisons. The diagonal terms $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ cancel against their displayed additive inverses by the additive inverse laws in Definition 43.26. The remaining scalar endpoint is two copies of the mixed sum:

$$(\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}) +_{\mathbb{K}} (\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

Finally, the scalar doubling row retained in Definition 86.32 identifies this paired sum with

$$2_{\mathbb{K}} \cdot_{\mathbb{K}} (\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}).$$

A final application of Definition 9.8 composes the expansion, negation, cancellation, and doubling witnesses into the displayed classifier row. □

Theorem 86.37 (InnerProduct ternary orthogonal Pythagorean row)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y, z : C_V$, assume*

$$x \perp_I y, \qquad x \perp_I z, \qquad y \perp_I z.$$

Then the displayed vector sum $(x +_V y) +_V z$ is carried, the norm-metric seed carries the diagonal scalar endpoint $\|(x +_V y) +_V z\|_I^2$, and the retained scalar classifier classifies

$$\|(x +_V y) +_V z\|_I^2 \sim_{\mathbb{K}} (\|x\|_I^2 +_{\mathbb{K}} \|y\|_I^2) +_{\mathbb{K}} \|z\|_I^2.$$

Proof. The embedded $\mathsf{VecSpace}^{\uparrow}$ row in Definition 86.1 carries $x +_V y$ and then carries the displayed sum $(x +_V y) +_V z$. Applying Definition 86.28 to x, y, z, $x +_V y$, and $(x +_V y) +_V z$ gives all norm-squared scalar endpoints used below inside the retained scalar source.

Read Definition 86.32 on the carried pair (x, y). By Definition 86.35 and the hypothesis $x \perp_I y$, the retained scalar classifier gives

$$\|x +_V y\|_I^2 \sim_{\mathbb{K}} \|x\|_I^2 +_{\mathbb{K}} \|y\|_I^2.$$

It remains to make the intermediate sum orthogonal to z. Apply Definition 86.39 to $x \perp_I z$ and $y \perp_I z$, obtaining

$$z \perp_I x \qquad \text{and} \qquad z \perp_I y.$$

Now apply Definition 86.40 with first endpoint z and additive pair (x, y). It carries $x +_V y$ and gives

$$z \perp_I (x +_V y).$$

A final use of Definition 86.39 gives $(x +_V y) \perp_I z$.

Read Definition 86.32 again, now on the carried pair $(x +_V y, z)$. Applying Definition 86.35 to the orthogonality witness $(x +_V y) \perp_I z$ gives

$$\|(x +_V y) +_V z\|_I^2 \sim_{\mathbb{K}} \|x +_V y\|_I^2 +_{\mathbb{K}} \|z\|_I^2.$$

The scalar endpoint source retained in Definition 86.3 is the $\mathsf{Field}^{\uparrow}$ scalar coordinate. By Definition 45.198, Definition 44.8, and Definition 43.26, its unchanged scalar classifier has addition congruence and classifier transitivity. Transport the first binary Pythagorean comparison through scalar addition with $\|z\|_I^2$ to obtain

$$\|x +_V y\|_I^2 +_{\mathbb{K}} \|z\|_I^2 \sim_{\mathbb{K}} (\|x\|_I^2 +_{\mathbb{K}} \|y\|_I^2) +_{\mathbb{K}} \|z\|_I^2.$$

Composing this scalar-addition transport with the second binary Pythagorean comparison by Definition 9.8 gives the displayed ternary classifier row. □

Theorem 86.38 (InnerProduct norm-of-sum expansion row)**.** *For carried vector endpoints $x, y : C_V$ in the Definition 86.32, the retained scalar classifier classifies the norm square of their vector sum with the four-term inner-product expansion:*

$$\|x +_V y\|_I^2 \sim_{\mathbb{K}} \|x\|_I^2 +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \|y\|_I^2.$$

No orthogonality hypothesis is used.

Proof. Unfold Definition 86.32. The embedded $\mathsf{VecSpace}^{\uparrow}$ rows carry $x +_V y$, and Definition 86.28 repacks the displayed norm square $\|x +_V y\|_I^2$ as the diagonal form endpoint

$$\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the additive component of Definition 86.6 in the first vector argument. The retained scalar classifier compares this diagonal endpoint with

$$\langle x, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the additive component of the same linearity row in the second vector argument to each displayed summand. The first summand is classified with

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}},$$

and the second summand is classified with

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The scalar endpoint source retained in Definition 86.3 is the $\mathsf{Field}^{\uparrow}$ scalar coordinate. By Definition 45.198, Definition 44.8, and Definition 43.26, its unchanged scalar classifier has addition congruence, associativity, and transitivity. Transport the two second-argument expansions through scalar addition, then reassociate the resulting scalar sum. This gives a scalar-classifier comparison from $\langle x +_V y, x +_V y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ to

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathbb{K}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Finally repack the two diagonal endpoints $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ as $\|x\|_I^2$ and $\|y\|_I^2$ using Definition 86.27 and Definition 86.28. Addition congruence in the same retained scalar classifier transports these two repackings through the four-term scalar sum, and Definition 9.8 composes the diagonal repacking, the two linearity expansions, scalar reassociation, and the final norm-squared repacking into the displayed classifier row. □

Theorem 86.39 (InnerProduct orthogonality symmetry row)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y : C_V$, the orthogonality row defined by the retained scalar-zero classifier is symmetric:*

$$x \perp_I y \quad \Longleftrightarrow \quad y \perp_I x.$$

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal symm constant endpoint`
↪ variant `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal symm package`

Proof. Assume first that $x \perp_I y$. By Definition 86.13, this is the scalar-zero classification

$$\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Apply the conjugate-symmetry row from Definition 86.5 with the two endpoints ordered as (y, x). It gives

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} \overline{\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}}.$$

The scalar endpoint source retained in Definition 86.3 supplies classifier transport for the displayed conjugation row. Applying that row to the scalar-zero classification of $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ gives

$$\overline{\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}} \sim_{\mathbb{K}} \overline{0_{\mathbb{K}}}.$$

The same scalar source fixes the zero endpoint under conjugation, the scalar-side row used in Definition 86.15, so

$$\overline{0_{\mathbb{K}}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Composing these three scalar classifier witnesses by Definition 9.8 gives

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

By Definition 86.13, this is $y \perp_I x$.

For the converse implication, assume $y \perp_I x$. The same definition gives

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Apply the conjugate-symmetry row with endpoints ordered as (x, y), transport this scalar-zero witness through the displayed conjugation row, use the zero-fix row for conjugation, and compose the scalar classifier witnesses by Definition 9.8. The result is

$$\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}},$$

which is exactly $x \perp_I y$ by Definition 86.13. □

Theorem 86.40 (InnerProduct orthogonal additivity row)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y, z : C_V$, if $x \perp_I y$ and $x \perp_I z$, then the displayed vector sum $y +_V z$ is carried and*

$$x \perp_I (y +_V z).$$

Proof. The embedded $\mathsf{VecSpace}^{\uparrow}$ row in Definition 86.1 supplies closure of the displayed vector addition, so $y +_V z$ is carried. Apply Definition 86.39 to the two hypotheses $x \perp_I y$ and $x \perp_I z$. This gives

$$y \perp_I x \qquad \text{and} \qquad z \perp_I x.$$

By Definition 86.13, these two facts are exactly the scalar-zero classifications

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}} \qquad \text{and} \qquad \langle z, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Use the additive part of Definition 86.6 in the displayed first argument, with second endpoint x. It compares the inner-product endpoint of the vector sum with the scalar sum:

$$\langle y +_V z, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathsf{scal}} \langle z, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The retained scalar source of Definition 86.3 is the $\mathsf{Field}^{\uparrow}$ scalar coordinate. By Definition 45.198, Definition 44.8, and Definition 43.26, its unchanged scalar classifier has addition congruence, additive zero laws, and classifier transitivity. Therefore the two scalar-zero witnesses above transport through scalar addition to give

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} +_{\mathsf{scal}} \langle z, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}} +_{\mathsf{scal}} 0_{\mathbb{K}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

This is the scalar addition component of Definition 43.41, read in the retained scalar classifier.

Composing the inner-product linearity comparison with this scalar-zero classification by Definition 9.8 gives

$$\langle y +_V z, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

By Definition 86.13, this is $(y +_V z) \perp_I x$. A final application of Definition 86.39, now to the carried pair $(y +_V z, x)$, gives $x \perp_I (y +_V z)$. □

Theorem 86.41 (InnerProduct orthogonality right scalar-action closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. Let $x, y : C_V$ be carried vector endpoints, and let $r : C_{\mathsf{scal}}$ be a carried scalar endpoint from the retained scalar source. If $x \perp_I y$, then $r \cdot_V y$ is carried and*

$$x \perp_I (r \cdot_V y).$$

Proof. By Definition 86.39, the hypothesis $x \perp_I y$ gives $y \perp_I x$. Unpacking this conclusion with Definition 86.13 gives the scalar-zero classifier witness

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

The embedded $\mathsf{VecSpace}^{\uparrow}$ row carries the scalar-action endpoint $r \cdot_V y$: scalar-action closure is one of the module stability rows retained by Definition 47.19 and Definition 46.18. Hence the inner-product endpoint $\langle r \cdot_V y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ is defined in the same carrier.

Apply the scalar-action compatibility component of Definition 86.6 to the carried scalar r and vector endpoints y and x. It gives a retained scalar-classifier comparison between

$$\langle r \cdot_V y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \quad \text{and} \quad r \cdot_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Inside the retained scalar source, the ring rows listed in Definition 43.26 are exactly the hypotheses needed for Definition 43.40. Applying that absorption row to the zero-classified factor $\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and the multiplier r gives

$$r \cdot_{\mathbb{K}} \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}}.$$

Compose the scalar-action compatibility comparison with this zero-product classification by Definition 9.8. The result is

$$\langle r \cdot_V y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathbb{K}} 0_{\mathbb{K}},$$

which is $(r \cdot_V y) \perp_I x$ by Definition 86.13. A second application of Definition 86.39, now to the carried pair $r \cdot_V y$ and x, gives $x \perp_I (r \cdot_V y)$. □

Theorem 86.42 (InnerProduct orthogonal additive inverse closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y : C_V$, if $x \perp_I y$, then the displayed additive inverse $-_V y$ is carried and*

$$x \perp_I (-_V y).$$

Proof. The embedded $\mathsf{VecSpace}^{\uparrow}$ row retains its additive $\mathsf{AbGroup}^{\uparrow}$ carrier by Definition 47.25, so the inverse-closure row carries $-_V y$.

Let $\eta = -_{\mathbb{K}} 1_{\mathbb{K}}$. The retained scalar source carries η by Definition 45.198 and Definition 43.2. The module rows of Definition 46.18 identify $\eta \cdot_V y$ with the displayed inverse: scalar unit changes y to $1_{\mathbb{K}} \cdot_V y$, scalar distributivity over scalar addition changes $(\eta +_{\mathbb{K}} 1_{\mathbb{K}}) \cdot_V y$ to $(\eta \cdot_V y) +_V (1_{\mathbb{K}} \cdot_V y)$, the additive inverse law retained by Definition 43.26 collapses $\eta +_{\mathbb{K}} 1_{\mathbb{K}}$ to $0_{\mathbb{K}}$, and Definition 46.29 collapses $0_{\mathbb{K}} \cdot_V y$ to 0_V. Thus $\eta \cdot_V y$ is a left additive inverse of y. The displayed inverse row gives $y +_V (-_V y) \sim_V 0_V$, and Definition 41.59, read in the additive $\mathsf{AbGroup}^{\uparrow}$ reduct, gives $\eta \cdot_V y \sim_V -_V y$.

Apply Definition 86.41 to the carried scalar η and the hypothesis $x \perp_I y$. It gives $x \perp_I (\eta \cdot_V y)$. Transport this orthogonality witness along $\eta \cdot_V y \sim_V -_V y$ by Definition 86.16, keeping the left endpoint fixed by reflexivity. The result is $x \perp_I (-_V y)$. □

Theorem 86.43 (InnerProduct orthogonality left additive closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y, z : C_V$, if $x \perp_I z$ and $y \perp_I z$, then the displayed vector sum $x +_V y$ is carried and*

$$(x +_V y) \perp_I z.$$

Proof. The embedded $\mathsf{VecSpace}^{\uparrow}$ row in Definition 86.1 supplies closure of the displayed vector addition, so $x +_V y$ is carried. Apply Definition 86.39 to the two hypotheses $x \perp_I z$ and $y \perp_I z$. This gives

$$z \perp_I x \qquad \text{and} \qquad z \perp_I y.$$

Now apply Definition 86.40 with left endpoint z and right endpoints x and y. Its conclusion gives

$$z \perp_I (x +_V y).$$

A final application of Definition 86.39, now to the carried pair $(z, x +_V y)$, gives

$$(x +_V y) \perp_I z.$$

□

Theorem 86.44 (InnerProduct orthogonality left scalar-action closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. Let $x, y : C_V$ be carried vector endpoints, and let $r : C_{\mathsf{scal}}$ be a carried scalar endpoint from the retained scalar source. If $x \perp_I y$, then $r \cdot_V x$ is carried and*

$$(r \cdot_V x) \perp_I y.$$

Proof. By Definition 86.39, the hypothesis $x \perp_I y$ gives

$$y \perp_I x.$$

Apply Definition 86.41 with left endpoint y, right endpoint x, and carried scalar r. Its conclusion carries the scalar-action endpoint $r \cdot_V x$ and gives

$$y \perp_I (r \cdot_V x).$$

A final application of Definition 86.39, now to the carried pair y and $r \cdot_V x$, gives

$$(r \cdot_V x) \perp_I y.$$

□

Theorem 86.45 (InnerProduct orthogonality left additive-inverse closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y : C_V$, if $x \perp_I y$, then the displayed additive inverse $-_V x$ is carried and*

$$(-_V x) \perp_I y.$$

Proof. By Definition 86.39, the hypothesis $x \perp_I y$ gives

$$y \perp_I x.$$

Apply Definition 86.42 to the carried pair (y, x). It carries the displayed inverse $-_V x$ and gives

$$y \perp_I (-_V x).$$

A final application of Definition 86.39, now to the carried pair y and $-_V x$, gives

$$(-_V x) \perp_I y.$$

□

Theorem 86.46 (InnerProduct orthogonality right additive closure)**.** *Let I be an inner-product* BHist *carrier as in Definition 86.1. For carried vector endpoints $x, y, z : C_V$, if $x \perp_I y$ and $x \perp_I z$, then the displayed vector sum $y +_V z$ is carried and*

$$x \perp_I (y +_V z).$$

Proof. The carried endpoints y and z lie in the embedded $\mathsf{VecSpace}^{\uparrow}$ row of Definition 86.1. Its additive $\mathsf{AbGroup}^{\uparrow}$ reduct therefore carries the displayed sum $y +_V z$.

Apply Definition 86.40 to the same inner-product carrier and to the carried triple (x, y, z). The two hypotheses $x \perp_I y$ and $x \perp_I z$ are exactly its orthogonality inputs, so its conclusion gives the scalar-zero row for the right additive endpoint:

$$x \perp_I (y +_V z).$$

This is the asserted right additive closure. □

86.4 Consumer threshold surface

Definition 86.47 (InnerProduct consumer threshold source)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *consumer threshold source is the NameCert-facing package through which later certificates read inner-product data. It has the following ten fields:*

1. *traditional target: inner-product geometry over a vector space;*
2. *BEDC source: the displayed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *source, its retained scalar and field rows, and the* $\mathsf{NameCert}$ *ledger discipline;*
3. *vector carrier: a predicate $C_V : \mathsf{BHist} \to \mathsf{Prop}$;*
4. *vector classifier: the displayed classifier $\sim_V$ on carried vector histories;*
5. *scalar endpoint classifier: the retained scalar classifier on all scalar endpoints read by the form;*
6. *form row: the scalar-valued endpoint row $\langle -, - \rangle_{\mathsf{InnerProduct}^{\uparrow}} : C_V \times C_V \to C_{\mathsf{scal}}$;*
7. *law rows: conjugate symmetry, sesquilinearity, positivity, and form endpoint transport;*
8. *ledger rows: scalar endpoints,* hsame *transport, optional* Cont *route histories, and the orthogonality rows visible to consumers;*
9. *source factorization: every carrier, classifier, scalar, form, law, transport, and ledger projection factors through Definition 86.1, Definition 86.3, and Definition 86.23;*
10. *exclusion boundary: Banach completeness, topology, host Hilbert carriers, and Cauchy-limit selectors are not fields of this source.*

Thus $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, *and* $\mathsf{RootSystem}^{\uparrow}$ *receive a single inner-product threshold package rather than separate echoes of the root rows.*

Theorem 86.48 (InnerProduct consumer carrier and form scope)**.** *Every carrier, classifier, scalar endpoint, and form read from the consumer threshold source of Definition 86.47 factors through the* BHist *carrier of Definition 86.1, the source rows of Definition 86.3, and the root threshold source of Definition 86.23. No consumer can read a second vector carrier, a second scalar endpoint source, or a host inner-product-space form from this package.*

Proof. Start with the carrier field. By Definition 86.4, the public carrier row is the displayed $\mathsf{VecSpace}^{\uparrow}$ BHist carrier C_V together with the scalar-valued form endpoint row. The consumer threshold source names this same row as its vector carrier and does not list any other point carrier.

Next read the scalar and form fields from Definition 86.3. The form endpoint lands in the retained scalar history carrier and is classified by the retained scalar classifier. By Definition 86.20, these vector and scalar projections exhaust the vector-scalar part of the root surface.

Finally apply Definition 86.24. The form law rows exposed at the consumer boundary are the form, law, and transport rows of Definition 86.23. Since the consumer threshold source is defined by projecting those root threshold rows, every consumer carrier or form read factors through the three cited source objects. □

Theorem 86.49 (InnerProduct consumer positivity and orthogonality scope)**.** *The positivity and orthogonality data exported by Definition 86.47 are exactly the positive-definite scalar-zero classification row and the orthogonality closure rows built from Definition 86.13. They add no host positivity predicate and no independent Hilbert-space orthogonality relation.*

Proof. For positivity, use Definition 86.12. The diagonal self-value $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ is classified as the scalar zero endpoint exactly when x is classified with the displayed vector zero row. This is the positive row exposed by Definition 86.24.

For orthogonality, unfold Definition 86.13. The statement $x \perp_I y$ is the scalar-zero classification of $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. The zero-left, zero-right, and transport rows are Definition 86.14, Definition 86.15, and Definition 86.16.

The closure rows available to downstream consumers are the rows of the orthogonality closure surface: Definition 86.39, Definition 86.40, Definition 86.41, and Definition 86.42. Each row is proved from the scalar-zero definition of orthogonality and the displayed inner-product law rows. Therefore the consumer threshold exports exactly scalar-zero positivity and the orthogonality rows derived from it. □

Theorem 86.50 (InnerProduct consumer Hilbert threshold package)**.** *The* $\mathsf{Hilbert}^{\uparrow}$ *certificate consumes from Definition 86.47 only the inner-product-side threshold rows: vector carrier and classifier, retained scalar endpoints, the form row, conjugate symmetry, sesquilinearity, positivity, form transport, orthogonality, and ledger provenance. Its Banach metric, completeness row, topology, and Cauchy-limit selector remain external. The* $\mathsf{Fourier}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, *and* $\mathsf{RootSystem}^{\uparrow}$ *consumers are restricted to the same form, law, transport, and ledger rows.*

Proof. For the Hilbert input, apply Definition 86.25. It states that the Hilbert-facing geometric data are the scalar-valued form rows and the orthogonality row derived from them, while Banach metric, completeness, and Cauchy-limit rows come from a separate dependency.

For the named non-Hilbert consumers, apply Definition 86.26. It restricts $\mathsf{Fourier}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, and $\mathsf{RootSystem}^{\uparrow}$ to the vector, scalar, form, law, transport, orthogonality, and ledger rows of the same root threshold source.

The consumer threshold source is the single package that exposes those root threshold rows to downstream certificates. Its exclusion boundary lists no Banach completeness, topology, host Hilbert carrier, or Cauchy-limit selector. Hence every named consumer reads only the inner-product threshold rows, and all completeness or topology structure remains outside $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$. □

86.5 Two-vector Gram boundary

Definition 86.51 (InnerProduct two-vector Gram source)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *two-vector Gram source is the* BHist *package obtained from Definition 86.47 by fixing two carried vector endpoints* $x, y : C_V$ *and retaining the four scalar endpoints*

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}, \qquad \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}, \qquad \langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}, \qquad \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

It has the following rows:

1. *vector carrier rows for the displayed endpoints* x *and* y *in the carrier* C_V *of Definition 86.1;*
2. *vector classifier rows comparing replacement endpoints by the displayed classifier* $\sim_V$*;*
3. *scalar classifier rows for the four displayed form values in the retained scalar endpoint source;*
4. *form transport rows for each of the four ordered pairs* (x, x)*,* (x, y)*,* (y, x)*, and* (y, y)*;*
5. *conjugate-symmetry rows identifying the two off-diagonal endpoints through the retained scalar classifier and scalar conjugation;*
6. *diagonal positivity rows for the two self-values;*
7. *orthogonality rows reading the scalar-zero classification of* $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ *and* $\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$*;*
8. *ledger rows recording exactly the vector, scalar, form, transport, conjugate-symmetry, positivity, and orthogonality sources used above.*

The source does not contain a host matrix, determinant, spectral row, topology, Banach completeness row, Cauchy-limit selector, or ambient Hilbert-space carrier. It is only the two-vector scalar-form packet exposed by the BHist *inner-product source.*

Theorem 86.52 (InnerProduct Gram carrier transport)**.** *Let* I *be an* $\mathsf{InnerProduct}^{\uparrow}$ BHist *source and let* $x, y, x', y' : C_V$ *be carried vector endpoints. If* $x \sim_V x'$ *and* $y \sim_V y'$*, then every scalar endpoint in the two-vector Gram source transports through the retained scalar classifier:*

$$\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathsf{scal}} \langle x', x'\rangle_{\mathsf{InnerProduct}^{\uparrow}},$$

$$\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathsf{scal}} \langle x', y'\rangle_{\mathsf{InnerProduct}^{\uparrow}},$$

$$\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathsf{scal}} \langle y', x'\rangle_{\mathsf{InnerProduct}^{\uparrow}},$$

and

$$\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \sim_{\mathsf{scal}} \langle y', y'\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Proof. Unfold the two-vector Gram source of Definition 86.51. Its scalar endpoints are exactly the four ordered form evaluations displayed in the statement.

For the first diagonal endpoint, apply Definition 86.7 with the vector classifier witness $x \sim_V x'$ in both arguments. This transports $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ to $\langle x', x'\rangle_{\mathsf{InnerProduct}^{\uparrow}}$.

For the first off-diagonal endpoint, apply Definition 86.7 to $x \sim_V x'$ and $y \sim_V y'$. This transports $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ to $\langle x', y'\rangle_{\mathsf{InnerProduct}^{\uparrow}}$.

For the reversed off-diagonal endpoint, apply the same transport row to $y \sim_V y'$ and $x \sim_V x'$. This transports $\langle y, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ to $\langle y', x'\rangle_{\mathsf{InnerProduct}^{\uparrow}}$.

For the second diagonal endpoint, apply the same transport row with $y \sim_V y'$ in both arguments. These four scalar classifier witnesses are precisely the form transport rows retained by the Gram source. □

Theorem 86.53 (InnerProduct Gram symmetry and zero exactness)**.** *For the two-vector Gram source of Definition 86.51, the off-diagonal scalar-zero rows are symmetric:*

$$\langle x, y\rangle_{\mathsf{InnerProduct}^\uparrow} \sim_{\mathsf{scal}} 0_{\mathsf{scal}} \quad\Longleftrightarrow\quad \langle y, x\rangle_{\mathsf{InnerProduct}^\uparrow} \sim_{\mathsf{scal}} 0_{\mathsf{scal}}.$$

Its diagonal scalar-zero rows are exactly vector-zero classification:

$$\langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow} \sim_{\mathsf{scal}} 0_{\mathsf{scal}} \quad\Longleftrightarrow\quad x \sim_V 0_V,$$

and

$$\langle y, y\rangle_{\mathsf{InnerProduct}^\uparrow} \sim_{\mathsf{scal}} 0_{\mathsf{scal}} \quad\Longleftrightarrow\quad y \sim_V 0_V.$$

Proof. For the off-diagonal row, unfold the orthogonality row Definition 86.13. The scalar-zero classification of $\langle x, y\rangle_{\mathsf{InnerProduct}^\uparrow}$ is the assertion $x \perp_I y$, and the scalar-zero classification of $\langle y, x\rangle_{\mathsf{InnerProduct}^\uparrow}$ is the assertion $y \perp_I x$.

Apply Definition 86.39. It gives exactly $x \perp_I y$ if and only if $y \perp_I x$. Repacking through the displayed definition gives the symmetric off-diagonal zero row for the Gram source.

For the x diagonal row, identify $\langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow}$ with the norm-squared endpoint $\|x\|_I^2$ of Definition 86.27. Then apply Definition 86.29 to obtain the equivalence with $x \sim_V 0_V$.

For the y diagonal row, use the same norm-squared seed with the endpoint y. A second application of Definition 86.29 gives the equivalence with $y \sim_V 0_V$. These are the two diagonal exactness rows retained by Definition 86.51. □

Theorem 86.54 (InnerProduct Gram consumer boundary)**.** *The downstream certificates* $\mathsf{Hilbert}^\uparrow$, $\mathsf{Fourier}^\uparrow$, $\mathsf{RiemannianMetric}^\uparrow$, *and* $\mathsf{RootSystem}^\uparrow$ *may consume from Definition 86.51 only the two-vector carrier, the four scalar endpoints, the form transport rows, the conjugate-symmetry row, the positive-definite diagonal rows, the orthogonality rows, and the ledger rows. Banach completeness, Cauchy-limit selection, host Hilbert-space structure, determinants, and spectral rows are outside this Gram source.*

Proof. Start from Definition 86.26. It states that $\mathsf{Hilbert}^\uparrow$, $\mathsf{Fourier}^\uparrow$, $\mathsf{RiemannianMetric}^\uparrow$, and $\mathsf{RootSystem}^\uparrow$ receive only the vector, scalar, form, law, transport, orthogonality, and ledger rows of the root threshold source.

Next apply Definition 86.50. It identifies the consumer-facing package as the same inner-product threshold surface and keeps Banach metric, completeness, topology, Cauchy-limit selection, and host Hilbert carrier data outside the $\mathsf{InnerProduct}^\uparrow$ source.

Inside that consumer-facing package, use Definition 86.52 for the four transported scalar endpoints and Definition 86.53 for the symmetric off-diagonal zero row and the two diagonal zero-exactness rows.

The definition of Definition 86.51 lists no host matrix, determinant, spectral row, topology, completeness row, Cauchy-limit selector, or ambient Hilbert-space carrier. Therefore the named downstream certificates can cite this source only for the two-vector scalar-form packet and its displayed ledger. □

Theorem 86.55 (InnerProduct Gram diagonal scalar nonnegativity)**.** *Let I be an* $\mathsf{InnerProduct}^\uparrow$ BHist *source whose retained scalar endpoint source carries the inherited scalar order row* $\preceq_{\mathsf{scal}}$*, with nonnegative scalar predicate read as* $0_{\mathsf{scal}} \preceq_{\mathsf{scal}} a$*. For the two-vector Gram source of Definition 86.51 with carried endpoints $x, y : C_V$, the two diagonal scalar endpoints satisfy*

$$0_{\mathsf{scal}} \preceq_{\mathsf{scal}} \langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow}$$

and

$$0_{\mathsf{scal}} \preceq_{\mathsf{scal}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

These are public scalar-order rows of the Gram packet, read from the retained diagonal positivity rows and not from a host matrix, determinant, spectral row, or Hilbert-space positivity predicate.

Proof. Unfold the two-vector Gram source of Definition 86.51. Its displayed scalar endpoints include the two self-values $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$, and its listed rows include the diagonal positivity rows for those two self-values.

Apply Definition 86.12 to the carried endpoint x. The positive-definite entry of the underlying $\mathsf{InnerProduct}^{\uparrow}$ carrier supplies the nonnegative scalar endpoint witness for $\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}$. Since the retained scalar order row reads nonnegativity as $0_{\mathsf{scal}} \preceq_{\mathsf{scal}} a$, this witness is precisely

$$0_{\mathsf{scal}} \preceq_{\mathsf{scal}} \langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

Apply the same positive-definite obligation to the carried endpoint y. Its nonnegative scalar endpoint witness for $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ repacks through the same inherited scalar order row as

$$0_{\mathsf{scal}} \preceq_{\mathsf{scal}} \langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The consumer threshold source of Definition 86.47 factors the form, scalar endpoint, positivity, transport, and ledger rows through the same $\mathsf{InnerProduct}^{\uparrow}$ carrier and retained scalar source. Therefore the two order comparisons are public rows of the two-vector Gram packet itself, with no additional analytic or host-space structure introduced. □

86.6 Root unblock obligation basis

Definition 86.56 (InnerProduct root obligation basis)**.** *The* $\mathsf{InnerProduct}^{\uparrow}$ *root obligation basis is the root-facing* BHist *package with the following ten fields.*

1. *The traditional target is the constructive inner-product interface: a vector-space carrier equipped with a scalar-valued form, conjugate symmetry, sesquilinearity, positive-definite zero exactness, and orthogonality transport.*
2. *The BEDC source is* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *together with the scalar rows retained from* $\mathsf{Field}^{\uparrow}$.
3. *The carrier is the displayed vector predicate* $C_V : \mathsf{BHist} \to \mathsf{Prop}$.
4. *The classifiers are the vector endpoint classifier* $\sim_V$ *and the scalar endpoint classifier* $\sim_{\mathsf{scal}}$.
5. *The form row is*
$$\langle -, -\rangle_{\mathsf{InnerProduct}^{\uparrow}} : C_V \times C_V \to C_{\mathsf{scal}}.$$
6. *The stability obligations are form transport, conjugate symmetry, sesquilinearity in the admitted arguments, positive-definite zero exactness, and orthogonality transport.*
7. *The ledger rows are the vector source row, scalar source row, form endpoint row, law rows, transport rows, orthogonality row, and downstream consumer rows.*
8. *The core theorem list consists of Definition 86.57, Definition 86.58, and Definition 86.59.*
9. *Exactness means row exhaustion: every root consumer can read only the rows listed in this basis, and each listed row is sourced from the displayed* BHist *data.*

10. *There is no standard bridge field in this basis; its status is a paper obligation surface for later formalization.*

Theorem 86.57 (InnerProduct root carrier-classifier unblock)**.** *The root obligation basis of Definition 86.56 exposes exactly the carrier C_V, the vector classifier $\sim_V$, the scalar endpoint classifier $\sim_{\mathsf{scal}}$, and the scalar-valued form endpoint row $\langle -, - \rangle_{\mathsf{InnerProduct}^\uparrow}$. It introduces no second vector carrier, no extra point classifier, and no host scalar equality.*

Proof. Read the source field of Definition 86.56. It names the embedded $\mathsf{NameCert}_{\mathsf{VecSpace}^\uparrow}$ and the scalar rows retained from $\mathsf{Field}^\uparrow$, which are the same source data displayed in Definition 86.3.

Apply Definition 86.4. This identifies the public carrier as the displayed predicate C_V together with the scalar-valued form row, and it rules out any additional point carrier.

Apply Definition 86.7. Its hypotheses and conclusion use exactly $\sim_V$ on vector endpoints and $\sim_{\mathsf{scal}}$ on scalar endpoints. Hence the basis exposes precisely the carrier, classifier, scalar endpoint classifier, and form endpoint rows listed in the statement. □

Theorem 86.58 (InnerProduct root form-law unblock)**.** *For the root obligation basis of Definition 86.56, conjugate symmetry, sesquilinearity, positive-definite zero exactness, and orthogonality are all sourced from displayed* BHist *rows of the* $\mathsf{InnerProduct}^\uparrow$ *certificate. None of these laws is supplied by a host Hilbert space, quotient field, or ambient topology.*

Proof. Use Definition 86.5 for conjugate symmetry and the positivity side of the diagonal law. These are scalar-history comparisons over the retained scalar classifier.

Use Definition 86.6 for the sesquilinearity rows. Their vector inputs come from the embedded $\mathsf{VecSpace}^\uparrow$ operations, and their scalar outputs remain in the retained scalar endpoint source.

Apply Definition 86.12. It supplies the exact zero clause saying that the self-form endpoint is scalar-zero exactly at the displayed vector-zero classifier class.

Finally unpack Definition 86.13. Orthogonality is the scalar-zero classification of the displayed form endpoint, so its transport and zero behavior are inherited from the same form and classifier rows. Thus every form law named by the basis is a displayed BHist row. □

Theorem 86.59 (InnerProduct root downstream threshold unblock)**.** *The downstream certificates* $\mathsf{Hilbert}^\uparrow$, $\mathsf{Norm}^\uparrow$, $\mathsf{Fourier}^\uparrow$, $\mathsf{RiemannianMetric}^\uparrow$, *and* $\mathsf{RootSystem}^\uparrow$ *may consume from Definition 86.56 only the carrier, classifier, scalar endpoint, form-law, orthogonality, and ledger rows listed there. They receive no Banach completeness, Cauchy-limit selector, ambient topology, or host Hilbert-space row from* $\mathsf{InnerProduct}^\uparrow$.

Proof. Start with Definition 86.23. It packages the carrier, classifier, scalar endpoint source, form row, form-law rows, orthogonality row, and ledger rows as the complete inner-product threshold surface.

Apply Definition 86.25. This separates the inner-product geometric rows consumed by $\mathsf{Hilbert}^\uparrow$ from the Banach metric, completeness, and Cauchy-limit rows supplied by the separate Banach dependency.

Apply Definition 86.26. It gives the same threshold coverage for $\mathsf{Hilbert}^\uparrow$, $\mathsf{Fourier}^\uparrow$, $\mathsf{RiemannianMetric}^\uparrow$, and $\mathsf{RootSystem}^\uparrow$, and it excludes Banach completeness, Cauchy-limit selection, ambient topology, and host Hilbert-space data from the $\mathsf{InnerProduct}^\uparrow$ source.

The $\mathsf{Norm}^\uparrow$ consumer reads the same inner-product threshold through the norm and metric seed rows of Definition 86.27. Those rows are generated from the form endpoint, positive-definite exactness, classifier transport, and ledger fields already listed in the basis. Therefore all five consumers are bounded by the same root obligation basis, with no additional downstream row projected from $\mathsf{InnerProduct}^\uparrow$. □

86.7 InnerProduct root scalar consumer frontier

Theorem 86.60 (InnerProduct root scalar projection transport)**.** *In the root obligation basis of Definition 86.56, the scalar endpoint of the form transports through the retained scalar classifier. Concretely, if $x \sim_V x'$ and $y \sim_V y'$ are vector-classifier witnesses for carried endpoints $x, y, x', y' : C_V$, then the projected form endpoints satisfy*

$$\langle x, y\rangle_{\mathsf{InnerProduct}^\uparrow} \sim_{\mathsf{scal}} \langle x', y'\rangle_{\mathsf{InnerProduct}^\uparrow}.$$

This projection uses only the root form row, the two vector-classifier witnesses, and the retained scalar classifier.

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductRoot scalar projection transport`

Proof. Read the form endpoint and both classifiers from Definition 86.56. The basis exposes the vector classifier $\sim_V$, the scalar classifier $\sim_{\mathsf{scal}}$, and the form endpoint row, with no additional scalar equality.

Apply Definition 86.7 to the two vector-classifier witnesses. Its conclusion is exactly a retained scalar-classifier witness between the two form endpoints. Thus form endpoint transport is a scalar projection of the root basis data. □

Theorem 86.61 (InnerProduct root diagonal zero exactness)**.** *For every carried vector endpoint $x : C_V$ in the root obligation basis of Definition 86.56, the diagonal self-value $\langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow}$ is classified as the scalar zero endpoint if and only if $x \sim_V 0_V$. The equivalence is a* BHist *row statement: scalar-zero classification of the diagonal is exactly the displayed vector-zero classifier class.*

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductSingletonDiagonal zero exactness`

Proof. First read the diagonal endpoint from the form-law surface of Definition 86.58. This places $\langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow}$ inside the displayed scalar endpoint source and keeps its classifier equal to $\sim_{\mathsf{scal}}$.

For the forward direction, assume that the diagonal endpoint is classified as scalar zero. Apply Definition 86.12. Its exactness clause converts this scalar-zero classification into the vector-classifier witness $x \sim_V 0_V$.

For the reverse direction, assume $x \sim_V 0_V$. The converse exactness clause of Definition 86.12, together with the root form-law unblock row, returns the scalar-zero classification of $\langle x, x\rangle_{\mathsf{InnerProduct}^\uparrow}$. Hence the diagonal zero test is exact at the root surface. □

Theorem 86.62 (InnerProduct root Norm-Hilbert consumer frontier)**.** *The* $\mathsf{Norm}^\uparrow$ *and* $\mathsf{Hilbert}^\uparrow$ *consumers of an* $\mathsf{InnerProduct}^\uparrow$ *root obligation basis may read only the following scalar-consumer frontier: the norm-squared diagonal endpoint $\|x\|_I^2 = \langle x, x\rangle$, the diagonal zero exactness row, the orthogonality row, the carrier and classifier rows, and the visible ledger rows. They receive no completeness, topology, host Hilbert-space carrier, or Cauchy-limit selector from* $\mathsf{InnerProduct}^\uparrow$.

Proof. For $\mathsf{Norm}^\uparrow$, use Definition 86.27. It defines the norm-squared seed from the diagonal form endpoint and lists the positive-definite exactness, classifier-transport, and ledger rows needed to consume that endpoint.

Apply Definition 86.61 to identify the zero test of the norm-squared diagonal with the displayed vector-zero classifier class. Orthogonality is read from Definition 86.13, so it is also a scalar-zero classification of a displayed form endpoint.

For $\mathsf{Hilbert}^{\uparrow}$, apply Definition 86.25. It separates the inner-product carrier, classifier, form, transport, orthogonality, and ledger rows from Banach metric and completeness data.

Finally use Definition 86.59. The stated root threshold contains no completeness field, topology field, host Hilbert-space carrier, or Cauchy-limit selector. Therefore the $\mathsf{Norm}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ consumer frontier is exactly the listed scalar-consumer surface. □

86.8 Public NameCert export

Theorem 86.63 (InnerProductUp public NameCert export)**.** *The carrier, classifier, scalar endpoint classifier, form row, conjugate-symmetry row, sesquilinearity rows, positive-definite row, orthogonality rows, and visible ledger assembled by Definition 86.17 form the public* $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ *export consumed by* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{Hilbert}^{\uparrow}$*. Completeness, topology, Banach metric, Cauchy limits, and standard bridge data are not fields of this export.*

Proof. Start from the root dependency surface of Definition 86.19. It names the displayed $\mathsf{VecSpace}^{\uparrow}$ carrier and classifier, the retained scalar endpoint classifier, the scalar-valued form row, its law rows, the positive-definite exactness row, the orthogonality rows, and the visible ledger as the available BHist rows.

The vector and scalar endpoints exported to consumers are exactly the rows identified by Definition 86.20. The form row, conjugate-symmetry row, sesquilinearity rows, endpoint transport, and positive-definite row are exactly the rows identified by Definition 86.21.

Now apply Definition 86.17. Its obligation surface packages those carrier, classifier, law, exactness, orthogonality, and ledger rows as the paper-side $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ surface. Since every listed field is already one of the displayed BHist rows, the export is a public NameCert-facing object rather than an additional hidden dependency.

Finally use Definition 86.50 and Definition 86.62. They identify the $\mathsf{Norm}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ consumer boundary as the same inner-product rows while excluding completeness, topology, Banach metric, Cauchy-limit selectors, and standard bridge data. Thus the displayed rows form the public export and its negative boundary. □

Theorem 86.64 (InnerProduct singleton terminal standard bridge)**.** *The public singleton* $\mathsf{InnerProduct}^{\uparrow}$ *export of Definition 86.63 supplies a paper-level* $\mathsf{StandardBridge}(\mathsf{InnerProduct}^{\uparrow}, T^*_{IP})$ *to the one-point inner-product host theory. The encode field sends the unique host vector of* T^*_{IP} *to* e*. The decode field sends every carried* $\mathsf{VecSpaceSingletonCarrier}$ *endpoint to the host point. The decode-encode round trip is terminal equality, and the encode-decode round trip is the singleton classifier witness for the carried endpoint. The theorem-preservation field covers the carrier row, scalar endpoint row, form row, zero-left orthogonality row, and public export rows. The no-host-leak field keeps the BEDC public surface to* BHist*,* $\mathsf{VecSpaceSingletonCarrier}$*,* $\mathsf{RealConstantHistoryClassifier}$*,* $\mathsf{InnerProductSingletonForm}$*, and the displayed public rows.*

Lean4 ✓ `BEDC.Derived.InnerProductUp.InnerProductUp StdBridge`

Proof. We check the six fields of section 9204.1. The encode field is the constant map from the unique host vector of T^*_{IP} to the empty BEDC endpoint e. The carrier and classifier rows in Definition 86.63 place e inside the public singleton vector carrier.

The decode field sends every carried $\mathsf{VecSpaceSingletonCarrier}$ endpoint to the unique host point. Since the host theory has one vector, decoded endpoints are identified by terminal equality.

For the decode-encode round trip, start with the unique host vector, encode it as e, and decode back to the host point. This is terminal equality in T^*_{IP}.

For encode-decode, start with a carried singleton endpoint. Decoding gives the unique host vector and encoding returns e. The singleton classifier row of the public export supplies the classifier witness identifying the original carried endpoint with e.

The theorem-preservation field is finite and explicit. The carrier, scalar endpoint, form, law, orthogonality, and ledger rows are those exposed by Definition 86.63. The zero-left orthogonality row is Definition 86.14, backed by BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal_zero_left. No completeness, Banach, Hilbert, spectral, or non-terminal host theorem is included in this list.

Finally, the no-host-leak field follows from the same public-object inventory. The bridge mentions the one-point host theory only inside encode, decode, and the finite preservation list. The BEDC public surface remains exactly BHist, VecSpaceSingletonCarrier, RealConstantHistoryClassifier, InnerProductSingletonForm, and the displayed rows of Definition 86.63. □

Theorem 86.65 (InnerProduct finite orthogonality derivation)**.** *For every accepted* InnerProduct$^{\uparrow}$ *packet, the zero-left orthogonality row is derived by finite inspection of the* VecSpace$^{\uparrow}$ *carrier, the scalar endpoint classifier, the sesquilinear rows, the conjugate-symmetry row, the positivity row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The derivation is the BEDC route through Definition 86.1, Definition 86.3, Definition 86.5, Definition 86.6, Definition 86.13, and Definition 86.14; it is not Hilbert completeness, Banach completion, spectral theory, or a host inner-product-space theorem.*

Proof. Start from the displayed carrier and source rows. The VecSpace$^{\uparrow}$ carrier and scalar endpoint classifier are already visible in Definition 86.1 and Definition 86.3, so the zero-left claim is read inside the accepted BEDC packet rather than in a host vector space.

Next inspect the form laws. The sesquilinear source row of Definition 86.6 and the conjugate-symmetry and positivity rows of Definition 86.5 expose exactly the scalar-form data used by the orthogonality classifier.

Finally apply the orthogonality rows. The definition Definition 86.13 names the finite classifier surface, and Definition 86.14 derives the zero-left row from that surface. Transport, replay, provenance, and local naming keep the same rows attached to the accepted packet, so the result is a finite BEDC route and exports no Hilbert, Banach, spectral, or host-completeness principle. □

Theorem 86.66 (InnerProduct finite Gram orthogonality induction)**.** *For every finite* BHist *list of pairwise orthogonal nonzero* InnerProduct$^{\uparrow}$ *rows carried by an accepted packet, there is a finite Gram ledger whose off-diagonal entries are the displayed scalar-zero rows and whose diagonal entries are the displayed positive scalar rows. The construction is stable under componentwise* hsame *transport and displayed* Cont *replay, so the same ledger can be consumed by* VecSpace$^{\uparrow}$ *scalar-action rows,* Norm$^{\uparrow}$ *norm seeds, and* Hilbert$^{\uparrow}$ *completion-facing orthogonality routes without importing a host Gram–Schmidt theorem or an ambient Hilbert basis.*

Proof. Induct on the finite BHist list carrier. The empty list has the empty Gram ledger, and the one-row list reads its positive diagonal entry from the positive-definite row already exposed by the accepted InnerProduct$^{\uparrow}$ packet.

For the cons step, assume the tail has a Gram ledger with zero off-diagonal rows and positive diagonal rows. The pairwise orthogonality premise supplies the scalar-zero rows between the new head and every carried tail row, while conjugate symmetry supplies the reverse off-diagonal rows. Positive definiteness supplies the new diagonal row because the head is nonzero.

Transport is componentwise. If the carried rows are replaced by hsame recognized representatives, the vector endpoints, scalar endpoints, form values, zero rows, and positivity rows are

transported by the packet rows already used in Definition 86.65. Displayed Cont replay keeps the finite ledger attached to the same provenance and local NameCert row.

Thus the induction produces a BEDC Gram ledger and no external basis choice. $\mathsf{VecSpace}^{\uparrow}$ consumers read only the retained scalar-action and additive rows, $\mathsf{Norm}^{\uparrow}$ consumers read the norm seeds and Pythagorean rows, and $\mathsf{Hilbert}^{\uparrow}$ consumers may use the ledger as a finite orthogonality boundary without receiving completeness or projection existence. □

Theorem 86.67 (InnerProduct finite orthogonal projection derivation)**.** *For every accepted* $\mathsf{InnerProduct}^{\uparrow}$ *packet and every displayed carried pair* $x, y : C_V$ *whose two-vector Gram source has a diagonal scalar row* $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ *separated from* 0_{scal}*, the finite projection component of* x *along* y *is read from Definition 86.51, the Gram symmetry and diagonal rows of Definition 86.53, the parallelogram and Pythagorean norm seed rows of Definition 86.32 and Definition 86.35, the orthogonality closure rows of Definition 86.41, and the scalar-action and additive rows retained by the embedded* $\mathsf{VecSpace}^{\uparrow}$ *source. The row states only that the BEDC packet can inspect the displayed scalar coefficient, carried scalar multiple, residual vector, and scalar-zero orthogonality classifier for that finite pair; it does not assert Hilbert projection existence, closed-subspace projection, least-distance attainment, Banach completeness, or a host projection theorem.*

Proof. Start with Definition 86.51. It displays the four scalar endpoints for x and y, in particular $\langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ and $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$, together with the transport, symmetry, positivity, orthogonality, and ledger rows needed to keep those endpoints in one finite packet.

The separated diagonal row lets the retained scalar source form the displayed coefficient

$$\alpha = \langle x, y\rangle_{\mathsf{InnerProduct}^{\uparrow}} \cdot_{\mathsf{scal}} \langle y, y\rangle^{-1}_{\mathsf{InnerProduct}^{\uparrow}}.$$

This is a scalar-source row, not an analytic minimizer. The embedded $\mathsf{VecSpace}^{\uparrow}$ source then carries $\alpha \cdot_V y$ and the displayed residual $x -_V (\alpha \cdot_V y)$ by its scalar-action and additive rows.

Use Definition 86.53 to keep the off-diagonal Gram row symmetric and to identify the diagonal zero rows with vector-zero classification. Use Definition 86.41 and the linearity row of Definition 86.6 to inspect the scalar endpoint

$$\langle x -_V (\alpha \cdot_V y), y\rangle_{\mathsf{InnerProduct}^{\uparrow}}.$$

The scalar inverse row for the displayed nonzero diagonal cancels the $\langle y, y\rangle_{\mathsf{InnerProduct}^{\uparrow}}$ factor, leaving the retained scalar-zero classifier for the residual against y.

Finally read the norm side only as a finite consistency check. The parallelogram seed source and Definition 86.35 say that once the residual is orthogonal to the carried multiple of y, the diagonal norm-squared rows split through the displayed scalar ledger. This records the projection row as finite BEDC data over $x, y, \alpha, \alpha \cdot_V y$ and the residual, with no coordinate for Hilbert completeness, closed-subspace projection, least-distance existence, or host-space projection. □

Closure status for $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$**.**

Constructive story: An inner-product object is assembled by taking an already displayed $\mathsf{VecSpace}^{\uparrow}$ history carrier, retaining its scalar endpoint source, and adding one scalar-valued form row on pairs of carried vector histories; packages without that vector-space source, scalar classifier, conjugate-symmetry row, sesquilinear rows, positive-definite row, and visible ledger are not admitted. Two such packages are classified only when the displayed BEDC rows agree componentwise: vector carrier and classifier, scalar endpoint classifier, form values, operation rows, conjugate symmetry, linearity, positivity, orthogonality, and ledger entries all match by the retained classifiers rather than by host isometry or external inner-product equality. The source therefore covers a vector-space carrier together with scalar histories and a form row, the pattern records symmetry, linearity, positive definiteness, norm and metric seeds, parallelogram seed, orthogonality closure, the two-vector Gram packet, and consumer thresholds, and the ledger keeps every vector endpoint, scalar endpoint, form value,

transport row, and positivity witness visible. The five name-certificate obligations are supplied by the displayed source: an accepted singleton form gives inhabitation in the concrete case, the component classifiers give reflexivity, symmetry, and transitivity, and each transport row keeps the vector, scalar, and form fields in the accepted carrier after recognized endpoint changes. A history package h lies in the signature gap of an $\mathsf{InnerProduct}^{\uparrow}$ token p when the source-domain accepts the displayed vector-space-plus-form data, the signature relation pairs it with an answer package and evidence ledger, and token introduction names that answer as p; unfolded, p can only consume the listed form laws, zero classifications, and orthogonality rows. The paper-level terminal bridge sends the one host vector to e and decodes every singleton carried endpoint back to the unique host vector, preserving the carrier, scalar endpoint, form, zero-left orthogonality, and public export rows. The chapter registers this displayed scalar-form package under the name $\mathsf{InnerProduct}^{\uparrow}$; it adds no $\mathsf{Norm}^{\uparrow}$ closure, Hilbert completeness, Banach completion, spectral theory, non-terminal standard model, or bridge-checked host theorem.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is the $\mathsf{InnerProduct}^{\uparrow}$ core surface, root threshold package, norm-metric seed, parallelogram norm seed, orthogonality closure, two-vector Gram boundary, consumer frontier, public NameCert export, and singleton terminal standard bridge of Definition 86.64.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.InnerProductUp.InnerProductSingletonOrthogonal\_zero\_left`

Bridge status: bridgeChecked

Not claimed: This chapter does not close bridgeChecked, $\mathsf{Norm}^{\uparrow}$, Hilbert completeness, Banach completion, spectral theory, or non-terminal standard models.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 87

A Concrete Naming Certificate for $\mathsf{TensorProduct}^{\uparrow}$

The tensor-product interface $\mathsf{TensorProduct}^{\uparrow}$ packages the universal bilinear factorization of two module carriers. It depends on $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ (chapter 46) and supplies the multilinear data on which higher tensor algebras and exterior powers are built.

87.1 Carrier definition

Definition 87.1 (Tensor-product singleton carrier)**.** *For a generated history h, the singleton tensor-product carrier holds when there are histories l and r such that l and r are both module-singleton histories and $\mathsf{Cont}(l, r, h)$ holds. Thus the tensor-product singleton object is represented by a concrete continuation ledger from its two singleton module endpoints.*

Lean4 (definition) `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier`

Theorem 87.2 (Tensor-product singleton carrier empty iff)**.** *For a generated history h, the tensor-product singleton carrier holds exactly when h is the empty history:*

$$\mathsf{TensorProductSingletonCarrier}(h) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Proof. For the forward direction, unpack the tensor carrier into singleton module endpoints and a continuation ledger. The endpoint carriers identify both factors with e, so continuation respect compares the ledger output with the empty right-unit continuation. Conversely, if $h\mathsf{hsamee}$, use two empty singleton module endpoints and transport the empty right-unit continuation to the result h. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier empty iff`

Theorem 87.3 (Tensor-product singleton carrier empty endpoint iff)**.** *For a generated history h, the tensor-product singleton carrier holds exactly when h is* hsame *to the empty history.*

Proof. Unpack the carrier into two module-singleton endpoints and a continuation ledger. The two module endpoints are empty, so the continuation endpoint is the empty append. Conversely, the empty endpoint is obtained from the two empty module factors and the left-unit continuation law. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier empty endpoint iff`

Theorem 87.4 (Tensor-product singleton result-empty continuation)**.** *For a history $t : \mathsf{BHist}$, if t satisfies the tensor-product singleton carrier, then t continues with the empty suffix to the empty history:*

$$\mathsf{TensorProductSingletonCarrier}(t) \Rightarrow \mathsf{Cont}(t, \mathsf{e}, \mathsf{e}).$$

Proof. Unpack the carrier into module-singleton endpoints l, r and a continuation $\mathsf{Cont}(l, r, t)$. The two endpoint carrier witnesses identify both l and r with e, and the displayed continuation therefore reduces to the empty result. The right-unit continuation gives $\mathsf{Cont}(t, \mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier result empty continuation`

Theorem 87.5 (Tensor-product singleton suffix carrier)**.** *Let p, s, and o be generated histories. If p and o satisfy the tensor-product singleton carrier and $\mathsf{Cont}(p, s, o)$ holds, then s satisfies the module-singleton carrier.*

Proof. Unpack the two tensor-product singleton carriers into module-singleton endpoints and continuation ledgers. The endpoint carriers identify both tensor endpoints with the empty history. The displayed continuation $\mathsf{Cont}(p, s, o)$ therefore has empty source and empty result, so the append-empty inversion for continuation identifies the suffix s with the empty history. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier continuation suffix carrier`

Theorem 87.6 (Tensor-product singleton carrier continuation result closure)**.** *Let p, s, and o be generated histories. If p satisfies the tensor-product singleton carrier, s satisfies the module-singleton carrier, and $\mathsf{Cont}(p, s, o)$ holds, then o satisfies the tensor-product singleton carrier.*

Proof. The tensor-product singleton carrier identifies p with the empty history, and the module-singleton carrier identifies s with the empty history. Transporting the continuation along these endpoint identifications gives $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, o)$. The left-unit result law identifies o with e, and the tensor-product singleton empty characterization rebuilds the carrier. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier continuation result closed`

Theorem 87.7 (Tensor-product singleton continuation prefix carrier)**.** *Let pre, $pair$, and out be generated histories. If $pair$ and out satisfy the tensor-product singleton carrier and $\mathsf{Cont}(pre, pair, out)$ holds, then pre satisfies the module-singleton carrier.*

Proof. The tensor-product singleton carrier identifies both *pair* and *out* with the empty history. After transporting the displayed continuation to an empty result, continuation empty-result inversion exposes the prefix endpoint as empty, which is exactly the module-singleton carrier. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier continuation prefix carrier`

Definition 87.8 (Tensor product singleton factor)**.** *For histories $l, r, t : \mathsf{BHist}$, the tensor-product singleton factor predicate holds when l, r, and t all lie in the module singleton carrier and the continuation relation composes l with r to produce t:*

$$\mathsf{TensorProductSingletonFactor}(l, r, t) := A_{\mathsf{e}}(l) \wedge A_{\mathsf{e}}(r) \wedge A_{\mathsf{e}}(t) \wedge \mathsf{Cont}(l, r, t).$$

Lean4 (definition) `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor`

Theorem 87.9 (Tensor product singleton carrier factor witness)**.** *If a tensor history satisfies the tensor-product singleton carrier, then it has left and right singleton module factors whose tensor-product singleton factor predicate holds, and the original continuation witness is retained.*

Proof. Eliminate the carrier witness. The two module-singleton factors force the left and right endpoints to be empty, and the continuation ledger then forces the tensor endpoint to be empty. These three singleton carrier facts together with the continuation ledger form the concrete factor witness. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier factor witness`

Theorem 87.10 (Tensor product singleton factor hsame transport)**.** *If* $\mathsf{TensorProductSingletonFactor}(l, r, t)$ *holds and* $l\mathsf{hsame}l'$*,* $r\mathsf{hsame}r'$*, and* $t\mathsf{hsame}t'$*, then* $\mathsf{TensorProductSingletonFactor}(l', r', t')$ *holds, and the transported continuation witness* $\mathsf{Cont}(l', r', t')$ *is retained.*

Proof. Unpack the singleton factor. Transport each singleton carrier endpoint by symmetry and transitivity of hsame. Transport the continuation witness across the three endpoint comparisons by continuation hsame transport. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor hsame transport`

87.2 The certificate

Theorem 87.11 (Tensor product singleton factor tensor semantic name certificate)**.** *Given a concrete singleton factor* $\mathsf{TensorProductSingletonFactor}(l, r, t)$*, the tensor endpoint carrier* $u \mapsto \mathsf{TensorProductSingletonFactor}(l, r, u)$ *forms a semantic name certificate when source, pattern, and ledger policy are all this carrier and the classifier is* hsame*.*

Proof. The displayed factor supplies the inhabited carrier. Reflexivity, symmetry, and transitivity are those of hsame. Carrier transport is exactly Definition 87.10 with the left and right endpoints fixed by reflexivity. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor tensor semanticNameCert`

Theorem 87.12 (Tensor product singleton factor source-target swap)**.** *For histories* $l, r, t : \mathsf{BHist}$*, if*

$$\mathsf{TensorProductSingletonFactor}(l, r, t)$$

holds, then

$$\mathsf{TensorProductSingletonFactor}(r, l, t)$$

holds. The resulting factor contains the swapped continuation witness $\mathsf{Cont}(r, l, t)$*.*

Proof. Unpack Definition 87.8. The factor supplies $A_{\mathsf{e}}(l)$, $A_{\mathsf{e}}(r)$, $A_{\mathsf{e}}(t)$, and $\mathsf{Cont}(l, r, t)$. In the concrete singleton module of Definition 46.2, the first two endpoint witnesses are the comparisons $l\mathsf{hsame}e$ and $r\mathsf{hsame}e$. Hence $l\mathsf{hsame}r$ follows by composing $l\mathsf{hsame}e$ with the symmetry of $r\mathsf{hsame}e$, and $r\mathsf{hsame}l$ follows by the same argument with the roles reversed.

Apply Definition 87.10 to the original factor with the comparisons $l\mathsf{hsame}r$, $r\mathsf{hsame}l$, and $t\mathsf{hsame}t$. Its conclusion is exactly the swapped predicate $\mathsf{TensorProductSingletonFactor}(r, l, t)$, and its continuation component is $\mathsf{Cont}(r, l, t)$. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor source target swap`

Theorem 87.13 (Tensor product singleton factor swap symmetry)**.** *For histories* $l, r, t : \mathsf{BHist}$*, if*

$$\mathsf{TensorProductSingletonFactor}(l, r, t)$$

holds, then

$$\mathsf{TensorProductSingletonFactor}(r, l, t)$$

holds. The resulting factor includes the continuation witness $\mathsf{Cont}(r, l, t)$*.*

Proof. Unpack Definition 87.8. The factor consists of $A_{\mathsf{e}}(l)$, $A_{\mathsf{e}}(r)$, $A_{\mathsf{e}}(t)$, and $\mathsf{Cont}(l, r, t)$. By Definition 46.2 and Definition 36.169, the two factor carrier witnesses are endpoint comparisons $l\mathsf{hsamee}$ and $r\mathsf{hsamee}$. Transitivity of hsame gives $l\mathsf{hsame}r$ by composing $l\mathsf{hsamee}$ with the symmetry of $r\mathsf{hsamee}$, and similarly gives $r\mathsf{hsame}l$.

Apply Definition 87.10 to the original factor with endpoint comparisons $l\mathsf{hsame}r$, $r\mathsf{hsame}l$, and the reflexive comparison $t\mathsf{hsame}t$. The transported conclusion is $\mathsf{TensorProductSingletonFactor}(r, l, t)$, and the retained continuation component is $\mathsf{Cont}(r, l, t)$. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor swap semantic name certificate`

Theorem 87.14 (Tensor product singleton factor associator)**.** *For histories* $l, r, s, t, u : \mathsf{BHist}$*, if*

$$\mathsf{TensorProductSingletonFactor}(l, r, t) \quad \textit{and} \quad \mathsf{TensorProductSingletonFactor}(t, s, u)$$

hold, then there is a history $m : \mathsf{BHist}$ *such that*

$$\mathsf{TensorProductSingletonFactor}(r, s, m) \quad \textit{and} \quad \mathsf{TensorProductSingletonFactor}(l, m, u)$$

hold. The two resulting factors retain the rebracketed continuation ledgers $\mathsf{Cont}(r, s, m)$ *and* $\mathsf{Cont}(l, m, u)$*.*

Proof. Unpack Definition 87.8. The first factor supplies $A_{\mathsf{e}}(l)$, $A_{\mathsf{e}}(r)$, $A_{\mathsf{e}}(t)$, and $\mathsf{Cont}(l, r, t)$; the second supplies $A_{\mathsf{e}}(t)$, $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(u)$, and $\mathsf{Cont}(t, s, u)$. Apply the associativity existence form of the continuation spine to the two continuation ledgers $\mathsf{Cont}(l, r, t)$ and $\mathsf{Cont}(t, s, u)$. This gives a history m with

$$\mathsf{Cont}(r, s, m) \quad \text{and} \quad \mathsf{Cont}(l, m, u).$$

It remains only to show that m lies in the singleton module carrier. By Definition 46.2 and Definition 36.169, the carrier witnesses $A_{\mathsf{e}}(r)$ and $A_{\mathsf{e}}(s)$ identify r and s with e. Transporting $\mathsf{Cont}(r, s, m)$ along these two endpoint comparisons gives $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, m)$. The continuation unit laws of Definition 21.12 compare this ledger with the empty unit continuation and identify m with e, which is exactly $A_{\mathsf{e}}(m)$.

Now package the first rebracketed factor from $A_{\mathsf{e}}(r)$, $A_{\mathsf{e}}(s)$, $A_{\mathsf{e}}(m)$, and $\mathsf{Cont}(r, s, m)$. Package the second from $A_{\mathsf{e}}(l)$, $A_{\mathsf{e}}(m)$, $A_{\mathsf{e}}(u)$, and $\mathsf{Cont}(l, m, u)$. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor associator`

Theorem 87.15 (Tensor product singleton factor classifier uniqueness)**.** *For histories* $l, r, t, t' : \mathsf{BHist}$*, if*

$$\mathsf{TensorProductSingletonFactor}(l, r, t) \quad \textit{and} \quad \mathsf{TensorProductSingletonFactor}(l, r, t')$$

hold, then

$$t\mathsf{hsame}t'.$$

Thus any two tensor endpoints obtained from the same singleton module factor pair are classifier-equal in the tensor endpoint certificate of Definition 87.11.

Proof. Unpack Definition 87.8 for both factor witnesses. The first witness supplies $\mathsf{Cont}(l, r, t)$; the second supplies $\mathsf{Cont}(l, r, t')$. These two continuation ledgers have the same source history l and the same right history r. Applying Definition 21.7 to the two ledgers identifies their outputs, giving $t \mathsf{hsame} t'$. The classifier in Definition 87.11 is hsame on this endpoint carrier, so this output identification is exactly tensor classifier equality. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor classifier uniqueness`
Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor classifier uniqueness readback`

Theorem 87.16 (TensorProductUp source and pattern obligation)**.** *In the singleton-factor tensor certificate, the source object is a pair of* $\mathsf{Module}^{\uparrow}$ *factor carriers over* BHist*. The pattern object is the concrete factor tuple* (l, r, t) *together with the continuation witness* $\mathsf{Cont}(l, r, t)$*. The carrier committed to the tensor endpoint is the existing specialization* $\mathsf{TensorProductSingletonFactor}(l, r, t)$ *for singleton module factors.*

Proof. Use Definition 87.8 as the source-pattern unpacking. Its four fields are exactly the left singleton module carrier, the right singleton module carrier, the tensor endpoint singleton carrier, and the continuation witness $\mathsf{Cont}(l, r, t)$.

The forward coverage direction is the witness theorem Definition 87.9: a tensor endpoint in the singleton carrier exposes two module-singleton endpoints and retains the continuation ledger. The displayed source and pattern data therefore repack into the same $\mathsf{TensorProductSingletonFactor}$ carrier. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonCarrier source pattern iff`

Theorem 87.17 (TensorProductUp classifier and stability obligation)**.** *For the singleton-factor tensor certificate, the endpoint classifier is* hsame *on tensor outputs. This classifier is stable under transport of the two factor carriers, transport of the continuation witness* $\mathsf{Cont}(l, r, t)$*, and the factor transport theorem for* hsame*.*

Proof. The certificate Definition 87.11 chooses hsame as the classifier on the tensor endpoint carrier. Transporting the left endpoint, right endpoint, and tensor output along hsame comparisons is therefore the only stability obligation.

Apply Definition 87.10. It transports the two factor-carrier witnesses and the tensor endpoint witness at the same time, and it retains the transported Cont ledger. Hence the classifier remains endpoint hsame after every admissible factor transport. □

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor classifier stability obligation`

Theorem 87.18 (TensorProductUp ledger and exactness obligation)**.** *For the singleton-factor tensor certificate, the ledger records exactly the two factor endpoints* l *and* r *and the continuation witness* $\mathsf{Cont}(l, r, t)$*. Coverage is supplied by the factor-witness theorem, soundness is supplied by the factor definition, and endpoint uniqueness is supplied by Definition 87.15.*

Lean4 ✓ `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor ledger exactness obligation`

Proof. Coverage comes from Definition 87.9: every tensor singleton endpoint exposes the two module-singleton endpoints and the retained continuation witness. Thus the ledger has no additional source component beyond l, r, and $\mathsf{Cont}(l, r, t)$.

Soundness is the direct reading of Definition 87.8. A valid factor consists of the two singleton module endpoints, the singleton tensor endpoint, and the continuation witness, so any recorded ledger entry rebuilds the public factor carrier.

Endpoint exactness uses Definition 87.15. Two ledger entries with the same factor endpoints have tensor outputs related by hsame. The associator theorem Definition 87.14 supplies the compatible rebracketing witness, so the ledger remains exact under the tensor factor shape used by the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

Constructive story: The tensor product closed here is the singleton factor construction: a tensor endpoint is admitted when two singleton $\mathsf{Module}^{\uparrow}$ endpoints l and r are displayed and a continuation ledger $\mathsf{Cont}(l, r, t)$ assembles them into the endpoint t, which in this singleton case is forced back to the empty history. Two tensor outputs for the same displayed factors are classified by endpoint history-sameness, and factor transport says that changing l, r, or t by hsame transports both the singleton carrier witnesses and the continuation witness, so no bilinear quotient object is formed. The source is the pair of singleton module factors, the pattern is the concrete triple (l, r, t) with the retained continuation row, and the ledger records exactly those two factor endpoints, the tensor endpoint, source-target swap, associator, suffix and prefix closure, and classifier-uniqueness readback. The five name-certificate obligations close because any displayed singleton factor gives an inhabited endpoint carrier, hsame gives reflexivity, symmetry, and transitivity on outputs, and the factor transport theorem is precisely carrier closure under the classifier while source, pattern, and ledger are the same predicate. A history h lies in the signature gap of a $\mathsf{TensorProduct}^{\uparrow}$ token p when the source-domain accepts the two singleton module factors and their continuation, the signature relation pairs the resulting tensor endpoint with an answer endpoint and evidence ledger, and token introduction names that answer as p; unfolded, h is the unique empty tensor endpoint determined by the two factors. The paper bridge reads this singleton surface as the one-point tensor product and reads back through the same continuation ledger, not through a universal bilinear quotient. The chapter registers this singleton continuation-factor package under the name $\mathsf{TensorProduct}^{\uparrow}$; non-singleton tensor algebra, bilinear universal properties, exterior powers, and standard-model completeness are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The singleton tensor package binds the singleton $\mathsf{Module}^{\uparrow}$ factor carrier, continuation suffix and prefix closure, factor witness, classifier uniqueness, source-target swap, associator, semantic NameCert, and ledger exactness rows stated in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TensorProductUp.TensorProductSingletonFactor\_tensor\_semanticNameCert`

Bridge status: paperBridge

Not claimed: This chapter does not claim non-singleton tensor algebra, bilinear universal properties, exterior powers, or standard-model completeness.

Upgrade path: No higher theory-axis grade remains for this singleton tensor package.

Chapter 88

A Concrete Naming Certificate for $\mathsf{Topology}^{\uparrow}$

The topology interface $\mathsf{Topology}^{\uparrow}$ packages a carrier with a family of open subsets closed under finite intersections and arbitrary unions, certified by an explicit indexing of opens. It generalises $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ (chapter 57) by dropping the located-distance requirement and supplies the substrate for $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and the topological side of $\mathsf{NameCert}_{\mathsf{CplxTopo}^{\uparrow}}$ (chapter 77).

88.1 Carrier definition

88.2 The certificate

Definition 88.1 (Indexed open topology carrier)**.** *An indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *is a tuple* $\mathcal{T} = (X, \mathsf{same}_X, \mathsf{OpenIx}, \mathsf{OpenAt}, \wedge_{\mathsf{Topology}^{\uparrow}})$ *consisting of a carrier* X*, a carrier classifier* same_X*, an index object* OpenIx*, a membership predicate*

$$\mathsf{OpenAt} : \mathsf{OpenIx} \times X \to \mathsf{Prop},$$

and a binary operation $i, j \mapsto i \wedge_{\mathsf{Topology}^{\uparrow}} j$ *on* OpenIx*. The classifier plays the role isolated in Definition 57.4; for every index* i *it transports open membership by*

$$\mathsf{same}_X(x, y) \Longrightarrow (\mathsf{OpenAt}(i, x) \Longleftrightarrow \mathsf{OpenAt}(i, y)).$$

The finite-intersection field is the exact pointwise law

$$\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, x) \Longleftrightarrow (\mathsf{OpenAt}(i, x) \wedge \mathsf{OpenAt}(j, x)).$$

Thus the index operation is part of the certificate data of section 88.2, not an external selection of an open subset.

Definition 88.2 (Indexed open carried predicate)**.** *For an indexed open topology carrier* $\mathcal{T}$ *and a predicate* $U : X \to \mathsf{Prop}$*, an index* $i : \mathsf{OpenIx}$ *carries* U*, written* $\mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$*, when it gives the pointwise classifier*

$$\mathsf{CarriesOpen}_{\mathcal{T}}(i, U) :\Longleftrightarrow \forall x : X, \quad U(x) \Longleftrightarrow \mathsf{OpenAt}(i, x).$$

A witnessed open of $\mathsf{Topology}^{\uparrow}$ *is such a pair* (i, h_i)*, with* $h_i : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$*.*

Definition 88.3 (Topology neighbourhood interface)**.** *For an indexed open topology carrier $\mathcal{T}$ over X, the neighbourhood predicate of a point $x : X$ at an open index $i : \mathsf{OpenIx}$ is*

$$\mathsf{Nbhd}_{\mathcal{T}}(x, i) :\Longleftrightarrow \mathsf{OpenAt}(i, x).$$

This row is indexed-open data: it records membership in the displayed index of $\mathcal{T}$ and does not select an external powerset element.

Lean4 (definition) `BEDC.Derived.TopologyUp.BHistIndexedNeighborhood`
Lean4 (definition) `BEDC.Derived.TopologyUp.TopologyNeighborhood`

Theorem 88.4 (Neighbourhood open membership)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. If an index $i : \mathsf{OpenIx}$ carries a predicate $U : X \to \mathsf{Prop}$, then for every point $x : X$,*

$$\mathsf{CarriesOpen}_{\mathcal{T}}(i, U) \Longrightarrow (\mathsf{Nbhd}_{\mathcal{T}}(x, i) \Longleftrightarrow U(x)).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistIndexedOpen neighborhood semantic name certificate`

Proof. Assume $h_i : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$ and fix $x : X$. By Definition 88.3, $\mathsf{Nbhd}_{\mathcal{T}}(x, i)$ is exactly $\mathsf{OpenAt}(i, x)$.

The carrying equation of Definition 88.2 gives $U(x) \Longleftrightarrow \mathsf{OpenAt}(i, x)$. Reading this equivalence in the opposite direction gives the displayed neighbourhood membership law. □

Theorem 88.5 (Neighbourhood classifier transport)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. For every open index $i : \mathsf{OpenIx}$ and points $x, y : X$,*

$$\mathsf{same}_X(x, y) \Longrightarrow (\mathsf{Nbhd}_{\mathcal{T}}(x, i) \Longleftrightarrow \mathsf{Nbhd}_{\mathcal{T}}(y, i)).$$

Proof. Assume $\mathsf{same}_X(x, y)$. Unfolding Definition 88.3 reduces the claim to transport of $\mathsf{OpenAt}(i, -)$.

The membership-stability field of Definition 88.1, applied to the fixed index i, gives exactly $\mathsf{OpenAt}(i, x) \Longleftrightarrow \mathsf{OpenAt}(i, y)$. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyUp StdBridge`

Theorem 88.6 (Neighbourhood finite-intersection)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. For every pair of open indices $i, j : \mathsf{OpenIx}$ and every point $x : X$,*

$$\mathsf{Nbhd}_{\mathcal{T}}(x, i \wedge_{\mathsf{Topology}^{\uparrow}} j) \Longleftrightarrow (\mathsf{Nbhd}_{\mathcal{T}}(x, i) \wedge \mathsf{Nbhd}_{\mathcal{T}}(x, j)).$$

Proof. Unfold the neighbourhood predicate of Definition 88.3. The desired statement becomes

$$\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, x) \Longleftrightarrow (\mathsf{OpenAt}(i, x) \wedge \mathsf{OpenAt}(j, x)).$$

This is precisely the finite-intersection field included in the indexed open topology carrier of Definition 88.1. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyNeighborhood finite intersection`

Theorem 88.7 (Topology indexed open finite-intersection closure)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. If indices $i, j : \mathsf{OpenIx}$ carry predicates $U, V : X \to \mathsf{Prop}$, then the certificate index $i \wedge_{\mathsf{Topology}^{\uparrow}} j$ carries the finite intersection $U \cap V$:*

$$\mathsf{CarriesOpen}_{\mathcal{T}}(i, U) \Longrightarrow \mathsf{CarriesOpen}_{\mathcal{T}}(j, V) \Longrightarrow \mathsf{CarriesOpen}_{\mathcal{T}}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, U \cap V),$$

where $(U \cap V)(x)$ *means* $U(x) \wedge V(x)$*. Equivalently, for every point* $x : X$ *the finite-intersection witness has the introduction and inversion form*

$$\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^\uparrow} j, x) \iff (U(x) \wedge V(x)),$$

and this carried intersection transports under the carrier classifier:

$$\mathsf{same}_X(x, y) \implies ((U(x) \wedge V(x)) \iff (U(y) \wedge V(y))).$$

This is the $\mathsf{Topology}^\uparrow$ *closure step whose concrete complex instance is used in Definition 77.22.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistIndexedOpen finite intersection closure`

Proof. Assume $h_U : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$ and $h_V : \mathsf{CarriesOpen}_{\mathcal{T}}(j, V)$. Fix $x : X$. The finite-intersection field of Definition 88.1 gives

$$\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^\uparrow} j, x) \iff (\mathsf{OpenAt}(i, x) \wedge \mathsf{OpenAt}(j, x)).$$

The two carry witnesses give $U(x) \iff \mathsf{OpenAt}(i, x)$ and $V(x) \iff \mathsf{OpenAt}(j, x)$. Composing these three equivalences gives

$$\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^\uparrow} j, x) \iff (U(x) \wedge V(x)),$$

which is precisely the pointwise carrying equation for $U \cap V$ after reversing the displayed equivalence.

For classifier transport, assume $\mathsf{same}_X(x, y)$. Convert $U(x) \wedge V(x)$ to $\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^\uparrow} j, x)$ by the carrying equation just proved. The membership-stability field of Definition 88.1 transports this to $\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^\uparrow} j, y)$, and the carrying equation at y converts back to $U(y) \wedge V(y)$. The reverse implication is identical with the same stability equivalence read in the opposite direction. □

Definition 88.8 (Indexed arbitrary-union open witness)**.** *Let* $\mathcal{T}$ *be an indexed open topology carrier for* $\mathsf{Topology}^\uparrow$ *over* X*, and let* A *be an indexing type. For a family of open indices* $\iota : A \to \mathsf{OpenIx}$*, an index* $u : \mathsf{OpenIx}$ *is an indexed arbitrary-union witness for* ι*, written* $\mathsf{UnionIx}_{\mathcal{T}}(u, \iota)$*, when it carries exactly the pointwise union of the indexed open memberships:*

$$\mathsf{UnionIx}_{\mathcal{T}}(u, \iota) :\iff \forall x : X, \quad \mathsf{OpenAt}(u, x) \iff \exists a : A,\ \mathsf{OpenAt}(\iota(a), x).$$

This is the arbitrary-union counterpart to the finite-intersection operation in Definition 88.1.

Theorem 88.9 (Topology indexed open arbitrary-union closure)**.** *Let* $\mathcal{T}$ *be an indexed open topology carrier for* $\mathsf{Topology}^\uparrow$ *over* X*. Let* A *be an indexing type, let* $\iota : A \to \mathsf{OpenIx}$ *be a family of open indices, and let* $U_a : X \to \mathsf{Prop}$ *be predicates indexed by* $a : A$*. Put* $U_\cup : X \to \mathsf{Prop}$ *for the predicate*

$$U_\cup(x) :\iff \exists a : A,\ U_a(x).$$

If $u : \mathsf{OpenIx}$ *is an indexed arbitrary-union witness for* ι *and each* $\iota(a)$ *carries* U_a*, then* u *carries* $U_\cup$*:*

$$\mathsf{UnionIx}_{\mathcal{T}}(u, \iota) \implies (\forall a : A,\ \mathsf{CarriesOpen}_{\mathcal{T}}(\iota(a), U_a)) \implies \mathsf{CarriesOpen}_{\mathcal{T}}(u, U_\cup).$$

Equivalently, for every point $x : X$ *the supplied union index has the introduction and inversion form*

$$\mathsf{OpenAt}(u, x) \iff \exists a : A,\ U_a(x),$$

and the union predicate transports under the carrier classifier:

$$\mathsf{same}_X(x,y) \Longrightarrow ((\exists a : A,\ U_a(x)) \Longleftrightarrow (\exists a : A,\ U_a(y))).$$

This is the generic $\mathsf{Topology}^{\uparrow}$ *indexed-family closure law whose concrete complex instance appears in Definition 77.23.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistIndexedOpen arbitrary union closure`

Proof. Assume $h_\cup : \mathsf{UnionIx}_{\mathcal{T}}(u,\iota)$ and $h_{\mathrm{car}} : \forall a : A,\ \mathsf{CarriesOpen}_{\mathcal{T}}(\iota(a), U_a)$. Fix $x : X$. If $U_\cup(x)$ holds, choose $a : A$ with $U_a(x)$. The carry witness $h_{\mathrm{car}}(a)$ gives $\mathsf{OpenAt}(\iota(a), x)$, hence $\exists a : A,\ \mathsf{OpenAt}(\iota(a), x)$. The arbitrary-union witness $h_\cup$ converts this existential open membership into $\mathsf{OpenAt}(u, x)$.

Conversely, if $\mathsf{OpenAt}(u, x)$ holds, then $h_\cup$ gives some $a : A$ with $\mathsf{OpenAt}(\iota(a), x)$. The carry witness $h_{\mathrm{car}}(a)$ converts this membership back into $U_a(x)$, hence $U_\cup(x)$. Since x was arbitrary, this proves $\mathsf{CarriesOpen}_{\mathcal{T}}(u, U_\cup)$, and reversing the displayed equivalence gives the pointwise introduction and inversion form.

For classifier transport, assume $\mathsf{same}_X(x,y)$. The carrying equation just proved converts $\exists a : A,\ U_a(x)$ to $\mathsf{OpenAt}(u,x)$. The membership-stability field of Definition 88.1, applied to the index u, transports this to $\mathsf{OpenAt}(u,y)$, and the carrying equation at y converts it to $\exists a : A,\ U_a(y)$. The reverse implication uses the same stability equivalence in the opposite direction. □

Definition 88.10 (Metric finite-ball indexed topology)**.** *Let C be a carrier equipped with the* $\mathsf{Met}^{\uparrow}$ *carrier data of Definition 57.1, with classifier* same_C *and distance* $d : C \times C \to \mathsf{Real}^{\uparrow}$*. Fix a strict ordered-real comparison* $<_{\mathsf{R}}$ *on* $\mathsf{Real}^{\uparrow}$ *and a positive-radius predicate* $\mathsf{PosRadius}_{\mathsf{R}}$*. A metric ball datum is a triple* $b = (c_b, r_b, \rho_b)$ *with* $c_b : C$*,* $r_b : \mathsf{Real}^{\uparrow}$*, and* $\rho_b : \mathsf{PosRadius}_{\mathsf{R}}(r_b)$*. Its open-ball predicate is*

$$\mathsf{MetricOpenBall}_d(b)(x) :\Longleftrightarrow d(c_b, x) <_{\mathsf{R}} r_b.$$

The metric finite-ball index object is $\mathsf{FinList}(\mathsf{BallData}_d)$*. For a finite list I of ball data, set*

$$\mathsf{OpenAt}_{d,\mathsf{ball}}(I, x) :\Longleftrightarrow \forall b \in I,\ \mathsf{MetricOpenBall}_d(b)(x).$$

The index operation $I \wedge_{\mathsf{Topology}^{\uparrow}} J$ *is finite-list concatenation* $I +\!\!+ J$*. Together with the carrier C and classifier* same_C*, these data are denoted* $\mathcal{T}_{d,\mathsf{ball}}$*.*

Lean4 (definition) `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier`

Theorem 88.11 (Metric balls instantiate indexed topology opens)**.** *Let C,* same_C*, d,* $<_{\mathsf{R}}$*, and* $\mathsf{PosRadius}_{\mathsf{R}}$ *be as in Definition 88.10. Assume the classifier field of Definition 57.4: if* $\mathsf{same}_C(x,y)$*, then $d(c,x)$ and $d(c,y)$ have the same real classifier for every fixed center $c : C$. Assume also that* $<_{\mathsf{R}}$ *is stable under that real classifier in its left argument. Then the finite-ball data of Definition 88.10 form an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *in the sense of Definition 88.1. For every ball datum $b = (c, r, \rho)$, the singleton index* $\mathsf{single}(b)$ *carries the metric ball:*

$$\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(\mathsf{single}(b), \mathsf{MetricOpenBall}_d(b)).$$

Equivalently, for all $x : C$,

$$\mathsf{MetricOpenBall}_d(b)(x) \Longleftrightarrow \mathsf{OpenAt}_{d,\mathsf{ball}}(\mathsf{single}(b), x),$$

and this ball membership transports under the metric carrier classifier:

$$\mathsf{same}_C(x,y) \Longrightarrow (\mathsf{MetricOpenBall}_d(b)(x) \Longleftrightarrow \mathsf{MetricOpenBall}_d(b)(y)).$$

Proof. First prove classifier transport for one ball datum $b = (c, r, \rho)$. If $\mathsf{same}_C(x, y)$ holds, the metric classifier field gives a real classifier witness between $d(c, x)$ and $d(c, y)$. Stability of $<_{\mathsf{R}}$ under that real classifier gives

$$d(c, x) <_{\mathsf{R}} r \iff d(c, y) <_{\mathsf{R}} r.$$

Unfolding the definition of $\mathsf{MetricOpenBall}_d(b)$ gives the displayed transport equivalence.

For a finite list I, apply the one-ball transport equivalence to each ball datum in I. Thus the predicate $\mathsf{OpenAt}_{d,\mathsf{ball}}(I, -)$ transports under same_C, which is the membership-stability field required in Definition 88.1.

The finite-intersection law is the finite-list membership split:

$$\forall b \in I{+\!\!+}J,\ \mathsf{MetricOpenBall}_d(b)(x) \iff \big((\forall b \in I,\ \mathsf{MetricOpenBall}_d(b)(x)) \wedge (\forall b \in J,\ \mathsf{MetricOpenBall}_d(b)(x))\big).$$

After refolding $\mathsf{OpenAt}_{d,\mathsf{ball}}$, this is exactly

$$\mathsf{OpenAt}_{d,\mathsf{ball}}(I \wedge_{\mathsf{Topology}^{\uparrow}} J, x) \iff (\mathsf{OpenAt}_{d,\mathsf{ball}}(I, x) \wedge \mathsf{OpenAt}_{d,\mathsf{ball}}(J, x)).$$

For the singleton index, finite-list membership has exactly one component, so

$$\mathsf{MetricOpenBall}_d(b)(x) \iff \mathsf{OpenAt}_{d,\mathsf{ball}}(\mathsf{single}(b), x).$$

This is the carrying equation of Definition 88.2. The final classifier-transport statement is the one-ball transport equivalence already proved. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier indexed open laws`
Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier singleton ball carries`
Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhood singleton ledger coverage`

Definition 88.12 (Metric-generated topology source)**.** *A metric-generated topology source for* $\mathsf{Topology}^{\uparrow}$ *consists of a certified* $\mathsf{Met}^{\uparrow}$ *carrier* C *with classifier* same_C*, a distance witness* $\mathsf{MetricDistanceWitness}(d)$ *for* $d : C \times C \to \mathsf{Real}^{\uparrow}$*, a strict ordered-real comparison* $<_{\mathsf{R}}$*, and positive-radius ledgers for both* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *radii. Its basic index rows are singleton metric balls* $b = (c_b, r_b, \rho_b)$ *as in Definition 88.10. Its finite-base rows are finite lists of such ball data, with intersection represented by finite-list concatenation. Its union rows are indexed arbitrary-union witnesses in the sense of Definition 88.8. Its boundary rows are distinguished empty and whole indices as in Definition 88.31.*

The resulting topology carrier is $\mathcal{T}_{d,\mathsf{ball}}$ *from Definition 88.10. The source ledger records, for every public open, the BHist row naming whether the open is a singleton metric ball, a finite-list intersection of metric balls, a binary generated-open meet of already carried generated opens, an arbitrary union of already carried metric opens, the empty open, or the whole open.*

Theorem 88.13 (Metric-generated topology carrier scope)**.** *Let* $\mathcal{G}$ *be a metric-generated topology source as in Definition 88.12. Then the BEDC source, carrier choice, classifier choice, and operation rows of* $\mathcal{G}$ *are exactly the* $\mathsf{Met}^{\uparrow}$ *carrier* C*, the carrier classifier* same_C*, the distance witness* $\mathsf{MetricDistanceWitness}(d)$*, the ordered* $\mathsf{Real}^{\uparrow}$ *comparison* $<_{\mathsf{R}}$*, positive* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *radius ledgers, finite-list metric-ball indices, indexed union witnesses, and boundary indices. No open row is selected from an external powerset or quotient.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier scope rows`

Proof. Unfold the metric-generated source of Definition 88.12. Its first row is the certified metric carrier C with classifier same_C, followed by the distance witness, ordered-real comparison, and positive-radius ledgers named in the statement.

Unfold the topology carrier $\mathcal{T}_{d,\mathsf{ball}}$ from Definition 88.10. The open index object is $\mathsf{FinList}(\mathsf{BallData}_d)$, the membership predicate is finite-list ball membership, and finite intersection is finite-list concatenation.

The remaining source rows are exactly the indexed arbitrary-union witnesses and the boundary indices named in Definition 88.12. Since every admitted open is introduced through one of these rows, the construction has no separate powerset selection and no quotient carrier. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier meet scope rows`

Theorem 88.14 (Metric-generated topology classifier scope)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12. Singleton metric balls, finite-list intersections, binary generated-open meets, supplied arbitrary unions, and boundary opens all transport membership along the metric carrier classifier same_C. Thus every classifier-stability obligation exposed by the metric-generated $\mathsf{Topology}^{\uparrow}$ source is discharged by the displayed indexed-open rows.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow classifier transport`

Proof. Singleton metric balls transport along same_C by Definition 88.16. That theorem also identifies each singleton ball index as a carried open of $\mathcal{T}_{d,\mathsf{ball}}$.

Finite intersections transport by Definition 88.17. Its conclusion applies the generic finite-intersection classifier transport to the concatenated finite-list index.

Binary generated-open meets transport by Definition 88.7, applied to the carried indices of the two admitted generated opens.

Supplied arbitrary unions transport by Definition 88.9. Boundary opens transport by Definition 88.32. These rows cover exactly the public open constructors named by the metric-generated source, so no additional classifier-stability field is required. □

Theorem 88.15 (Metric-generated topology ledger scope)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, equipped with indexed arbitrary-union witnesses and indexed boundary open witnesses. Every public open admitted by $\mathcal{G}$ carries a visible BHist provenance row of exactly one of the displayed constructors: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, or whole open. Composite rows preserve the constructor provenance of their inputs, and no unindexed open is admitted.*

Proof. The ledger sentence in Definition 88.19 assigns a BHist row to every public open accepted by the generated-open package. The possible row names are precisely singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, and whole open.

The constructor rows used by Definition 88.18 give the corresponding witnesses. Singleton metric-ball rows come from Definition 88.16, finite-list intersection rows from Definition 88.17, binary generated-open meet rows from Definition 88.7, arbitrary-union rows from Definition 88.9, and boundary rows from Definition 88.32.

For a composite row, the certificate records both the outer constructor and the input rows it combines. Therefore the provenance visible at the public surface remains one of the displayed constructor rows, and an open predicate without an index and ledger row is not admitted by $\mathcal{G}$. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree meet ledger constructor`

Theorem 88.16 (Metric-generated base-open transport)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12. For every metric ball datum $b = (c, r, \rho)$ and points $x, y : C$, metric classifier equality of the points transports basic open membership:*

$$\mathsf{same}_C(x, y) \Longrightarrow (\mathsf{MetricOpenBall}_d(b)(x) \Longleftrightarrow \mathsf{MetricOpenBall}_d(b)(y)).$$

Equivalently, the singleton index $\mathsf{single}(b)$ *carries the metric ball in the indexed topology carrier* $\mathcal{T}_{d,\mathsf{ball}}$*:*

$$\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(\mathsf{single}(b), \mathsf{MetricOpenBall}_d(b)).$$

Proof. Apply the construction of $\mathcal{T}_{d,\mathsf{ball}}$ from Definition 88.10. Its carrier, classifier, finite-list indices, and membership predicate are exactly the rows named by the metric-generated source.

The transport field is supplied by Definition 88.11. That theorem uses the metric point classifier and the ordered-real stability ledger to transport the inequality $d(c, x) <_{\mathsf{R}} r$ to $d(c, y) <_{\mathsf{R}} r$.

The same theorem states the singleton carrying equation, while Definition 88.1 and Definition 88.2 identify that equation as a carried open of the $\mathsf{Topology}^{\uparrow}$ indexed carrier. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistMetricBaseOpen singleton ledger coverage`

Theorem 88.17 (Metric-generated finite-base intersection)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12. Let I and J be finite lists of metric ball data, and let $U_I, V_J : C \to \mathsf{Prop}$ be the predicates carried by I and J in $\mathcal{T}_{d,\mathsf{ball}}$. Then the concatenated finite list $I +\!\!+ J$ carries the generated finite-base intersection:*

$$\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(I, U_I) \Longrightarrow \mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(J, V_J) \Longrightarrow \mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(I +\!\!+ J, U_I \cap V_J).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhood bundleAppend carries intersection`

Proof. The metric-generated source uses finite lists as the index object and finite-list concatenation as the operation $I \wedge_{\mathsf{Topology}^{\uparrow}} J$. Thus $I +\!\!+ J$ is the certificate index named by the finite-intersection row of Definition 88.10.

Apply Definition 88.7 to the indexed open carrier $\mathcal{T}_{d,\mathsf{ball}}$ supplied by Definition 88.11. The two carrying hypotheses for I and J are repacked into the carried predicate $U_I \cap V_J$.

The classifier transport for the resulting intersection is the transport conclusion of Definition 88.7, specialized to the metric classifier same_C. □

Theorem 88.18 (Metric-generated TopologyUp obligation surface)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, equipped with indexed arbitrary-union witnesses for all families of generated metric opens and indexed boundary open witnesses $\bot_{\mathsf{Topology}^{\uparrow}}$ and $\top_{\mathsf{Topology}^{\uparrow}}$. Then $\mathcal{G}$ supplies the concrete $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ obligation surface: the metric carrier C, classifier transport for metric-basic opens, singleton metric-ball coverage, finite intersections by concatenated ball lists, binary meets of already carried generated opens, arbitrary unions by supplied union indices, empty and whole opens by boundary indices, and a visible BHist ledger naming the constructor row used for each carried open.*

Proof. The carrier row is C with classifier same_C. By Definition 88.12 and Definition 88.10, these data form the indexed topology carrier $\mathcal{T}_{d,\mathsf{ball}}$.

Singleton metric-ball coverage and classifier transport are supplied by Definition 88.16. This gives the base-open row of the certificate without selecting opens from an external powerset.

Finite intersections are supplied by Definition 88.17. The certificate index for the intersection is the concatenated finite list, so the ledger names the finite-list constructor and the two source rows it combines.

Binary generated-open meets are supplied by the indexed-open finite-intersection operation on the carried indices of already admitted generated opens. The ledger row records both source witnesses and the meet index, so the composite open remains inside the visible certificate surface.

Arbitrary unions are supplied by the indexed union witnesses of the source and repacked through Definition 88.9. Boundary opens are supplied by the distinguished indices and the closure theorem of Definition 88.32.

The remaining obligation is provenance. The source ledger records one BHist row for each accepted open predicate: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, or whole open. These rows account for the carrier, classifier, base coverage, closure, boundary, and ledger fields of the concrete $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodCarrier metric generated namecert surface`
Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow metric generated obligation surface`

Definition 88.19 (Generated-open exactness package)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, with carrier C, classifier same_C, indexed topology carrier $\mathcal{T}_{d,\mathsf{ball}}$, indexed arbitrary-union witnesses, and indexed boundary open witnesses. A predicate $U : C \to \mathsf{Prop}$ is admitted by the generated-open exactness package, written $\mathsf{GenOpenExact}_{\mathcal{G}}(U)$, exactly when there is a visible BHist ledger row r carrying U and r is one of the six constructor rows:*

$$\text{singleton metric ball}, \quad \text{finite-list intersection}, \quad \text{binary generated-open meet},$$
$$\text{arbitrary union}, \quad \text{empty open}, \quad \text{whole open}.$$

Equivalently, U is admitted iff some row r of these six forms supplies an index $i_r : \mathsf{OpenIx}$ and a carrying equation

$$\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(i_r, U).$$

The binary generated-open meet row has, as displayed data, two admitted generated-open witnesses for U and V, their carried indices i_U and i_V, and the meet index $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ carrying $U \cap V$.

The package has the following ten fields. The traditional target is the indexed family of topology opens. The BEDC source is $\mathsf{Met}^\uparrow$ together with $\mathsf{Real}^\uparrow$ and $\mathsf{Rat}^\uparrow$ positive-radius ledgers and the BHist ledger rows. The carrier is C. The classifier is same_C. Stability is indexed-open membership transport along same_C. The ledger is exactly the six-row constructor ledger displayed above. The core theorem list is Definition 88.20, Definition 88.21, and Definition 88.22. The exactness predicate is the conjunction of row coverage, classification soundness, and classification completeness. The standard bridge is the usual metric-generated topology obtained from open balls, finite intersections, binary generated-open meets, arbitrary unions, and boundary opens. The status is a paper proof target for the $\mathsf{Topology}^\uparrow$ exactness contract.

Lean4 (definition) `BEDC.Derived.TopologyUp.BHistGeneratedOpenExact`

Theorem 88.20 (Generated-open row coverage)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. If $U : C \to \mathsf{Prop}$ satisfies $\mathsf{GenOpenExact}_{\mathcal{G}}(U)$, then U has a visible ledger witness of one of the six displayed row forms: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, or whole open. In particular, no external powerset row and no unindexed open predicate is admitted by the exactness package.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpenExact row coverage`

Proof. Unfold $\mathsf{GenOpenExact}_{\mathcal{G}}(U)$ from Definition 88.19. The definition admits U only by exhibiting a BHist row r, an index i_r, and a carrying equation for U.

The same definition restricts r to the six constructor rows listed in the metric-generated ledger. Therefore the witness is a singleton metric-ball row, a finite-list intersection row, a binary generated-open meet row, an arbitrary-union row, an empty-open row, or a whole-open row.

An external powerset row is not among these constructors. Since admission is by one of the displayed rows together with a carried index, an unindexed predicate has no route into $\mathsf{GenOpenExact}_{\mathcal{G}}$. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpenExact public open witness`
Lean4 ✓ `BEDC.Derived.TopologyUp.BHistLedgerGeneratedOpen row coverage`

Theorem 88.21 (Generated-open classification soundness)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. If a visible generated-open row r carries $U : C \to \mathsf{Prop}$, then U transports under the metric carrier classifier:*

$$\mathsf{same}_C(x, y) \Longrightarrow (U(x) \Longleftrightarrow U(y)).$$

The transport is obtained by cases on the row constructor named by r.

Proof. Fix a carried row r and assume $\mathsf{same}_C(x, y)$. We distinguish the six possible constructors supplied by Definition 88.19.

1. If r is a singleton metric-ball row, use Definition 88.16. Its transport clause moves the metric-ball inequality along same_C, and its carrying equation identifies that ball predicate with U.
2. If r is a finite-list intersection row, use Definition 88.17. The theorem supplies the concatenated finite-list carrier for the intersection and gives classifier transport for the carried intersection predicate.
3. If r is a binary generated-open meet row, unpack the two carried generated-open inputs and their indices i_U, i_V. Apply Definition 88.7 to the indexed carrier $\mathcal{T}_{d,\mathsf{ball}}$; the meet index $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ carries the pointwise intersection and transports it along same_C.
4. If r is an arbitrary-union row, use Definition 88.9. The union index carries the supplied family union, and the theorem transports the existential membership predicate through the same carrier classifier.
5. If r is the empty-open row, use Definition 88.32. The bottom boundary index carries the empty predicate, so both directions of the displayed equivalence are vacuous after transport.
6. If r is the whole-open row, again use Definition 88.32. The top boundary index carries the whole carrier predicate, and the theorem gives classifier transport for that carried predicate.

These cases exhaust the generated-open ledger by Definition 88.20, hence every row witness carrying U is classifier-stable. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpen classifier transport`

Theorem 88.22 (Generated-open classification completeness)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. Singleton metric balls, finite-list intersections of metric-ball rows, binary finite intersections of carried generated opens, supplied arbitrary unions of carried generated opens, the empty open, and the whole open are all admitted by $\mathsf{GenOpenExact}_{\mathcal{G}}$. In each case, the admission witness is the corresponding visible BHist ledger row.*

Proof. For a singleton metric ball, choose the singleton metric-ball row. Its index is $\mathsf{single}(b)$, and the carrying equation is supplied by Definition 88.16.

For a finite-list intersection of metric-ball rows, choose the finite-list intersection row. The row records the two finite-list indices and the concatenated finite-list index, and Definition 88.17 supplies the carrying equation for the intersection predicate.

For a binary finite intersection of carried generated opens, unpack the two generated-open witnesses to obtain their carried indices i_U and i_V. Choose the binary generated-open meet row. The generic closure theorem Definition 88.7 supplies the carrying equation for $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ and the predicate $U \cap V$.

For a supplied arbitrary union, choose the arbitrary-union row. The supplied union index and the family of carried generated opens are repacked by Definition 88.9, giving the carrying equation required by Definition 88.19.

For the empty open and the whole open, choose the two boundary rows. The bottom and top indices carry these predicates by Definition 88.32.

Each chosen row is one of the six visible constructors in the exactness package, so each listed predicate satisfies $\mathsf{GenOpenExact}_{\mathcal{G}}$. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyPublicOpenTree generated open exactness`

Theorem 88.23 (Generated-open binary meet row)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. If $U, V : C \to \mathsf{Prop}$ have generated-open exactness witnesses, then those witnesses determine a visible binary generated-open meet row. Its data are carried indices $i_U, i_V : \mathsf{OpenIx}$ for U and V and the meet index $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ carrying $U \cap V$:*

$$\mathsf{GenOpenExact}_{\mathcal{G}}(U) \Longrightarrow \mathsf{GenOpenExact}_{\mathcal{G}}(V) \Longrightarrow \mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(i_U \wedge_{\mathsf{Topology}^\uparrow} i_V, U \cap V).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpenExact binary meet row`

Proof. Unpack the generated-open witness for U. By Definition 88.19, it supplies a visible ledger row, an index i_U, and a carrying equation $\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(i_U, U)$.

Unpack the generated-open witness for V in the same way, obtaining i_V and $\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}(i_V, V)$.

Apply Definition 88.7 to these two carried indices in the indexed topology carrier $\mathcal{T}_{d,\mathsf{ball}}$. The theorem returns exactly the carrying equation for $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ and the predicate $U \cap V$.

The binary generated-open meet row records the two source generated-open witnesses, the two displayed indices, and this meet-index carrying equation. Therefore the row is visible in the BHist ledger rather than an unrecorded powerset construction. □

Theorem 88.24 (Generated-open binary meet admission)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. For predicates $U, V : C \to \mathsf{Prop}$,*

$$\mathsf{GenOpenExact}_{\mathcal{G}}(U) \Longrightarrow \mathsf{GenOpenExact}_{\mathcal{G}}(V) \Longrightarrow \mathsf{GenOpenExact}_{\mathcal{G}}(U \cap V).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpen binary meet admission`

Proof. Assume generated-open exactness witnesses for U and V. By Definition 88.23, these witnesses determine the binary generated-open meet row with index $i_U \wedge_{\mathsf{Topology}^\uparrow} i_V$ carrying $U \cap V$.

Choose that row as the admission witness in Definition 88.19. Since the binary generated-open meet is one of the six displayed constructor rows, the predicate $U \cap V$ satisfies $\mathsf{GenOpenExact}_{\mathcal{G}}$. □

Theorem 88.25 (Generated-open six-row exactness package)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. The row coverage theorem, classification soundness theorem, and classification completeness theorem for* $\mathsf{GenOpenExact}_{\mathcal{G}}$ *range over exactly the six-row ledger: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, and whole open. This package admits no external powerset row.*

Proof. Row coverage is Definition 88.20: every admitted predicate has one of the six displayed row witnesses and no unindexed predicate is accepted.

Classification soundness is Definition 88.21. Its proof treats all six row constructors, including the binary generated-open meet row through the indexed finite-intersection theorem.

Classification completeness is Definition 88.22, with the binary admission step isolated in Definition 88.24. The exactness package is therefore closed under the certificate meet operation by a visible BHist row, with no appeal to an external powerset constructor. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpen six row exactness package`

Theorem 88.26 (Metric-generated TopologyUp public certificate)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19, with indexed arbitrary-union witnesses and indexed boundary open witnesses. Then $\mathcal{G}$ exports the public* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *surface for downstream consumers: the displayed* $\mathsf{Met}^{\uparrow}$ *carrier C and classifier* same_C *come from Definition 88.18, while row coverage, classification soundness, and classification completeness for the generated opens come from Definition 88.25. The exported ledger rows are exactly singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, and whole open.*

Downstream use of this certificate is restricted to the displayed $\mathsf{Met}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *ordered-radius comparison, the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *positive-radius ledgers, the indexed union witnesses, the boundary witnesses, and the visible BHist ledger rows. It admits no external powerset row and no quotient carrier.*

Proof. Apply Definition 88.18 to the metric-generated source. This supplies the carrier, classifier, classifier-stability rows, singleton metric-ball row, finite-list intersection row, generated-open meet row, union row, boundary rows, and visible ledger discipline for the concrete $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ obligation surface.

Apply Definition 88.25 to the same source. Its three components provide the row coverage, classification soundness, and classification completeness clauses for $\mathsf{GenOpenExact}_{\mathcal{G}}$, and its statement restricts exactness to the six displayed rows.

Read the indexed boundary witness data through Definition 88.31. The distinguished bottom and top indices account for the empty-open and whole-open rows, so those boundary clauses are part of the exported certificate rather than separate powerset choices.

Packaging these pieces gives the public downstream surface: carrier and classifier data from the obligation-surface theorem, exactness from the six-row package, boundary rows from the indexed witnesses, and provenance from the BHist ledger. The only exported carrier is C, and every accepted open predicate is carried by one of the displayed indexed rows, so quotient carriers and external powerset rows have no entry point into the certificate. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistGeneratedOpen semantic name certificate`

Fit: traditional target, indexed topology opens; BEDC source, $\mathsf{Met}^{\uparrow}$ with $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ radius ledgers; carrier choice, the displayed metric carrier C; classifier choice, same_C; stability obligations, indexed-open transport along same_C; ledger, the six visible BHist rows; core theorem list, Definition 88.18 and Definition 88.25; exactness predicate, $\mathsf{GenOpenExact}_{\mathcal{G}}$; standard bridge

status, the usual metric-generated topology surface; status, public paper certificate for downstream $\mathsf{Topology}^{\uparrow}$ use.

Definition 88.27 (Metric finite-base neighbourhood row)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, with carrier C and indexed topology carrier $\mathcal{T}_{d,\mathsf{ball}}$. For a point $x : C$, a metric finite-base neighbourhood row at x is data*

$$(I, x, h_I, \kappa_I)$$

where $I : \mathsf{FinList}(\mathsf{BallData}_d)$, the membership witness is

$$h_I : \mathsf{OpenAt}_{d,\mathsf{ball}}(I, x),$$

the carrying equation is the finite-list pointwise law

$$\mathsf{OpenAt}_{d,\mathsf{ball}}(I, z) \iff \forall b \in I,\ \mathsf{MetricOpenBall}_d(b)(z)$$

for every $z : C$, and κ_I is the visible BHist *ledger row naming the finite-list intersection constructor of $\mathcal{G}$. The row is public neighbourhood data because it exposes the point x, the finite index I, its membership witness at x, its carrying equation, and the ledger constructor that admits the row.*

Lean4 (definition) `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhoodRow`

This finite-base neighbourhood row consumes the center-radius and positive-radius membership surface of chapter 1450 together with the metric source of chapter 57 before it is repacked as a $\mathsf{Topology}^{\uparrow}$ generated-open row.

Theorem 88.28 (Metric finite-base neighbourhood decomposition)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, and let $I : \mathsf{FinList}(\mathsf{BallData}_d)$ be a finite list of metric ball data. For every $x : C$,*

$$\mathsf{Nbhd}_{\mathcal{T}_{d,\mathsf{ball}}}(x, I) \iff \forall b \in I,\ \mathsf{MetricOpenBall}_d(b)(x).$$

Equivalently, the finite-base neighbourhood row at x exists exactly when every displayed ball datum in I contains x.

Proof. Unfold the neighbourhood predicate of Definition 88.3 for the indexed carrier $\mathcal{T}_{d,\mathsf{ball}}$. The left side becomes $\mathsf{OpenAt}_{d,\mathsf{ball}}(I, x)$.

Unfold the metric finite-ball index object of Definition 88.10. Its membership predicate for a finite list is exactly the assertion that every ball datum in the list contains the point.

Refolding this displayed law gives the stated equivalence. The row formulation in Definition 88.27 then packages the same witness together with the carrying equation and the finite-list intersection ledger row. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhood append decomposition`

Theorem 88.29 (Metric finite-base neighbourhood classifier transport)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12. For every finite list $I : \mathsf{FinList}(\mathsf{BallData}_d)$ and points $x, y : C$,*

$$\mathsf{same}_C(x, y) \implies (\mathsf{Nbhd}_{\mathcal{T}_{d,\mathsf{ball}}}(x, I) \iff \mathsf{Nbhd}_{\mathcal{T}_{d,\mathsf{ball}}}(y, I)).$$

Thus membership in a metric finite-base neighbourhood row is transported by the carrier classifier.

Proof. The theorem Definition 88.11 supplies the indexed-open carrier $\mathcal{T}_{d,\mathsf{ball}}$ from the metric ball data. In particular, finite-list open membership is stable along same_C.

Apply Definition 88.5 to this indexed carrier and to the fixed index I. Its conclusion is exactly the displayed equivalence between the two neighbourhood predicates.

The decomposition theorem Definition 88.28 identifies these neighbourhood predicates with finite-list ball containment at x and y, so the transported predicate is the public finite-base neighbourhood row rather than an unindexed open subset. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhood classifier transport`

Theorem 88.30 (Metric finite-base neighbourhood ledger coverage)**.** *Let $\mathcal{G}$ be a metric-generated topology source equipped as in Definition 88.19. For every finite list $I : \mathsf{FinList}(\mathsf{BallData}_d)$ and point $x : C$, a metric finite-base neighbourhood row at x is admitted by $\mathsf{GenOpenExact}_{\mathcal{G}}$ through the finite-list intersection constructor. Its visible data are the index I, the carrying equation for $\mathsf{OpenAt}_{d,\mathsf{ball}}(I, -)$, the membership witness at x, and the corresponding* BHist *ledger row. No external powerset row is exposed.*

Proof. Start with the finite list I and read it as the metric finite-ball index of Definition 88.10. Its displayed predicate is $\mathsf{OpenAt}_{d,\mathsf{ball}}(I, -)$, with the carrying equation of Definition 88.27.

The finite-list constructor row is the finite-base intersection row described by Definition 88.17. It records the finite metric-ball inputs, the concatenated finite-list index used by the certificate operation, and the carrying equation for the resulting intersection predicate.

Apply Definition 88.22 to this finite-list intersection row. The exactness package of Definition 88.19 therefore admits the carried predicate through the finite-list intersection constructor.

Finally apply the row-coverage direction of Definition 88.25. It restricts every admitted generated-open witness to the displayed six-row ledger, so the neighbourhood row exposes the finite index, carrying equation, point membership witness, and BHist provenance, with no route through an external powerset row. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistFiniteBaseNeighborhood finiteListIntersection ledger coverage`

Definition 88.31 (Indexed boundary open witnesses)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. Indexed boundary open witnesses are distinguished indices $\bot_{\mathsf{Topology}^{\uparrow}}, \top_{\mathsf{Topology}^{\uparrow}} : \mathsf{OpenIx}$ satisfying the pointwise laws*

$$\mathsf{OpenAt}(\bot_{\mathsf{Topology}^{\uparrow}}, x) \iff \mathsf{False}$$

and

$$\mathsf{OpenAt}(\top_{\mathsf{Topology}^{\uparrow}}, x) \iff \mathsf{True}$$

for every $x : X$. They are part of the displayed topology certificate data whenever the interface is used as a $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source rather than merely as a finite-intersection and arbitrary-union closure spine.

Theorem 88.32 (Topology indexed open boundary closure)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X, equipped with indexed boundary open witnesses $\bot_{\mathsf{Topology}^{\uparrow}}$ and $\top_{\mathsf{Topology}^{\uparrow}}$ as in Definition 88.31. Let $\varnothing_X : X \to \mathsf{Prop}$ and $\mathsf{Whole}_X : X \to \mathsf{Prop}$ be the predicates*

$$\varnothing_X(x) :\iff \mathsf{False}$$

and

$$\mathsf{Whole}_X(x) :\Longleftrightarrow \mathsf{True}.$$

Then the boundary indices carry the empty and whole predicates:

$$\mathsf{CarriesOpen}_{\mathcal{T}}(\bot_{\mathsf{Topology}^\uparrow}, \varnothing_X)$$

and

$$\mathsf{CarriesOpen}_{\mathcal{T}}(\top_{\mathsf{Topology}^\uparrow}, \mathsf{Whole}_X).$$

Moreover both carried predicates transport under the carrier classifier:

$$\mathsf{same}_X(x, y) \Longrightarrow (\varnothing_X(x) \Longleftrightarrow \varnothing_X(y))$$

and

$$\mathsf{same}_X(x, y) \Longrightarrow (\mathsf{Whole}_X(x) \Longleftrightarrow \mathsf{Whole}_X(y)).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistIndexedOpen boundary closure`

Proof. For the empty predicate, unfold $\mathsf{CarriesOpen}_{\mathcal{T}}$ from Definition 88.2. The boundary law of Definition 88.31 identifies $\mathsf{OpenAt}(\bot_{\mathsf{Topology}^\uparrow}, x)$ with False at each point, which is exactly the carrying equation for $\varnothing_X$.

For the whole predicate, the same unfolding uses the top boundary law. It identifies $\mathsf{OpenAt}(\top_{\mathsf{Topology}^\uparrow}, x)$ with True at each point, giving the carrying equation for Whole_X.

The transport claims can be read either from the two carrying equations and the membership-stability field of Definition 88.1, or directly from the fact that False and True do not depend on the point. In both directions the supplied $\mathsf{same}_X(x, y)$ witness is the certificate route allowed by the topology carrier. □

Theorem 88.33 (Topology NameCert obligation package)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^\uparrow$ over X, equipped with indexed arbitrary-union witnesses for every indexed family of opens and indexed boundary open witnesses $\bot_{\mathsf{Topology}^\uparrow}$ and $\top_{\mathsf{Topology}^\uparrow}$. Then the following data assemble the paper-side $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ obligation package: the carrier and membership-stability classifier of Definition 88.1, the carried-predicate interface of Definition 88.2, the finite-intersection closure of Definition 88.7, the arbitrary-union closure of Definition 88.9, the boundary-open closure of Definition 88.32, and the visible ledger policy recording the chosen open index, its carrying equation, and the constructor row used to obtain it.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyPublicOpenTree semantic name certificate`

Proof. The carrier row is $\mathcal{T}$ itself: it supplies X, same_X, OpenIx, OpenAt, finite-intersection indexing, and pointwise membership stability. The carried-predicate row is exactly Definition 88.2, so every public open is represented by an index together with its pointwise carrying equation.

Finite intersections are supplied by Definition 88.7. Given two carried opens, the certificate index $i \wedge_{\mathsf{Topology}^\uparrow} j$ carries their pointwise intersection and the resulting predicate transports along same_X.

Arbitrary unions are supplied by Definition 88.9. The assumed union witness chooses the index for a family, while the theorem repacks the family of carried predicates into a carried union and transports that union through the same classifier.

Boundary opens are supplied by Definition 88.32. The distinguished indices $\bot_{\mathsf{Topology}^\uparrow}$ and $\top_{\mathsf{Topology}^\uparrow}$ carry the empty and whole predicates, so the certificate has both boundary rows without choosing them from an external powerset.

The ledger policy records only displayed certificate provenance: an open predicate is accepted when it is carried by an explicit index, and composite opens retain the constructor row that produced the finite intersection, arbitrary union, empty open, or whole open. These rows account for the carrier, classifier transport, closure, boundary, and ledger fields of the paper-side $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ obligation package. □

Definition 88.34 (Topology public open row)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X. A topology public open row for a predicate $U : X \to \mathsf{Prop}$ is data*

$$(i, U, h_i, \lambda_i)$$

where $i : \mathsf{OpenIx}$, $h_i : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$, and λ_i is a visible BHist ledger constructor row naming one of the accepted open constructors: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, or whole open. The row is public because it exposes both the index carrying equation and the ledger constructor that explains why the open belongs to the $\mathsf{Topology}^{\uparrow}$ certificate surface.

Theorem 88.35 (Topology public open row classifier transport)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X, and let (i, U, h_i, λ_i) be a topology public open row in the sense of Definition 88.34. Then membership in U transports along the carrier classifier:*

$$\mathsf{same}_X(x, y) \Longrightarrow (U(x) \Longleftrightarrow U(y)).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistCarriesOpen classifier transport`

Proof. Assume $\mathsf{same}_X(x, y)$. The carrying witness h_i identifies $U(x)$ with $\mathsf{OpenAt}(i, x)$ and $U(y)$ with $\mathsf{OpenAt}(i, y)$.

Apply the membership-stability field of Definition 88.1 to the same index i. It transports $\mathsf{OpenAt}(i, x)$ to $\mathsf{OpenAt}(i, y)$ and also gives the reverse implication by reading the displayed equivalence in the opposite direction.

Composing these three equivalences gives $U(x)$ if and only if $U(y)$. The ledger row λ_i is not needed for the logical transport; it records the public provenance of the carried open while the transport itself is supplied by the indexed carrier. □

Theorem 88.36 (Metric-generated public open row coverage)**.** *Let $\mathcal{G}$ be a metric-generated topology source as in Definition 88.12, equipped with indexed arbitrary-union witnesses for generated metric opens and indexed boundary open witnesses. Every public open admitted by $\mathcal{G}$ determines a topology public open row for the carrier $\mathcal{T}_{d,\mathsf{ball}}$ of Definition 88.12. Its index is the displayed generated-open index, its carrying equation is the corresponding $\mathsf{CarriesOpen}_{\mathcal{T}_{d,\mathsf{ball}}}$ witness, and its ledger row names one of the six constructors listed in Definition 88.34.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree topology public open tree`

Proof. Proceed by cases on the constructor row recorded by the metric-generated source ledger.

For a singleton metric ball, use Definition 88.16. It supplies both classifier transport for the ball predicate and the carrying equation for the singleton index in $\mathcal{T}_{d,\mathsf{ball}}$; pair that index and carrying equation with the singleton metric-ball ledger row.

For a finite-list intersection, use Definition 88.17. The concatenated finite list is the generated-open index, and the theorem supplies the carrying equation for the pointwise intersection. The ledger row names the finite-list intersection constructor.

For a binary generated-open meet, unpack the two public generated-open rows and apply Definition 88.7 to their carried indices. The resulting meet index carries the pointwise intersection, and the ledger row names the binary generated-open meet constructor.

For an arbitrary union, use the indexed union witness in the source and apply Definition 88.9. This gives the union index and the carrying equation for the pointwise union, with the ledger row naming arbitrary union.

For the empty and whole opens, use Definition 88.32. The distinguished boundary indices carry the empty and whole predicates, and the ledger rows name the empty-open and whole-open constructors. These cases exhaust the public constructors of Definition 88.12, so every admitted open yields a public open row. □

Theorem 88.37 (Singleton public open row coverage)**.** *For the singleton empty-history topology carrier of Definition 88.38, the two-row ledger admits topology public open rows for bottom, top, finite meets, and arbitrary unions. The carried predicates are exactly the displayed empty predicate, the singleton carrier predicate C_{e}, their finite meets under the two-row meet table, and the unions selected by the singleton union ledger.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton public open row coverage`

Proof. The bottom and top rows come from Definition 88.39. They give the two boundary carrying equations, and their ledger rows name the empty-open and whole-open constructors.

Finite meets are supplied by Definition 88.40. Splitting on the two-row index table gives the absorbing bottom cases and the top-top case, so the meet index carries the pointwise meet and the public ledger names the finite-meet constructor.

Arbitrary unions are supplied by Definition 88.44. If no top row appears, the bottom index carries the union; if a top row appears, the top index carries it. Pair the selected index and carrying equation with the arbitrary-union ledger row.

These rows are all rows of the singleton indexed-open carrier, so each is a topology public open row in the sense of Definition 88.34. □

Definition 88.38 (Singleton empty-history topology carrier)**.** *The singleton empty-history topology carrier for* $\mathsf{Topology}^{\uparrow}$ *is rooted at the BEDC history carrier. Its point carrier is the* e*-class*

$$C_{\mathsf{e}}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e})$$

for $h : \mathsf{BHist}$*. The carrier classifier is* hsame *restricted to this class:*

$$\rho_{\mathsf{e}}(h, k) :\Longleftrightarrow C_{\mathsf{e}}(h) \wedge C_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

The open-index object is the two-row ledger

$$\mathsf{OpenIx}_{\mathsf{e}} := \{\bot_{\mathsf{Topology}^{\uparrow}}, \top_{\mathsf{Topology}^{\uparrow}}\}.$$

Membership is given by

$$\mathsf{OpenAt}_{\mathsf{e}}(\bot_{\mathsf{Topology}^{\uparrow}}, h) \Longleftrightarrow \mathsf{False}$$

and

$$\mathsf{OpenAt}_{\mathsf{e}}(\top_{\mathsf{Topology}^{\uparrow}}, h) \Longleftrightarrow C_{\mathsf{e}}(h).$$

The finite-meet table is the displayed two-row table with $\bot_{\mathsf{Topology}^{\uparrow}}$ *absorbing and* $\top_{\mathsf{Topology}^{\uparrow}}$ *neutral:*

$$\bot_{\mathsf{Topology}^{\uparrow}} \wedge_{\mathsf{Topology}^{\uparrow}} i = \bot_{\mathsf{Topology}^{\uparrow}}, \qquad i \wedge_{\mathsf{Topology}^{\uparrow}} \bot_{\mathsf{Topology}^{\uparrow}} = \bot_{\mathsf{Topology}^{\uparrow}}, \qquad \top_{\mathsf{Topology}^{\uparrow}} \wedge_{\mathsf{Topology}^{\uparrow}} \top_{\mathsf{Topology}^{\uparrow}} = \top_{\mathsf{Topo}}$$

Thus every open row is anchored in a concrete BHist *predicate rather than in an external powerset.*

Lean4 (definition) `BEDC.Derived.TopologyUp.TopologySingletonMeet`

Theorem 88.39 (Singleton topology boundary open laws)**.** *For the singleton carrier of Definition 88.38, the displayed bottom and top indices give the indexed boundary witnesses required by Definition 88.31. The bottom row carries the empty predicate,*

$$\mathsf{OpenAt}_{\mathsf{e}}(\bot_{\mathsf{Topology}^{\uparrow}}, h) \iff \mathsf{False},$$

and the top row carries the whole singleton carrier,

$$\mathsf{OpenAt}_{\mathsf{e}}(\top_{\mathsf{Topology}^{\uparrow}}, h) \iff C_{\mathsf{e}}(h).$$

Proof. Unfold the two membership clauses in Definition 88.38. The bottom clause is definitionally the empty predicate and the top clause is definitionally the carrier predicate C_{e}.

These two displayed equations are exactly the boundary package requested by Definition 88.31. Applying Definition 88.32 therefore identifies the empty open and the whole carrier open inside the indexed-open interface. □

Theorem 88.40 (Singleton topology finite-intersection laws)**.** *For the singleton carrier of Definition 88.38, the displayed meet table satisfies the finite-intersection field of Definition 88.1:*

$$\mathsf{OpenAt}_{\mathsf{e}}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, h) \iff \big(\mathsf{OpenAt}_{\mathsf{e}}(i, h) \wedge \mathsf{OpenAt}_{\mathsf{e}}(j, h)\big).$$

The restricted classifier ρ_{e} also transports each displayed row, so every carried predicate in the sense of Definition 88.2 remains stable under the singleton hsame *route.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton finite intersection laws`

Proof. For classifier transport, split on the open index. In the $\bot_{\mathsf{Topology}^{\uparrow}}$ row both sides are False, so the equivalence is immediate.

In the $\top_{\mathsf{Topology}^{\uparrow}}$ row, unfold ρ_{e}. Its first two components are precisely $C_{\mathsf{e}}(h)$ and $C_{\mathsf{e}}(k)$, and its third component records the restricted hsame route between the representatives. Therefore the top membership predicate transports along the displayed classifier.

For finite intersections, split on the two indices. Any case containing $\bot_{\mathsf{Topology}^{\uparrow}}$ reduces both sides to False by the absorbing rows of the meet table.

The $\top_{\mathsf{Topology}^{\uparrow}}, \top_{\mathsf{Topology}^{\uparrow}}$ case reduces both sides to $C_{\mathsf{e}}(h)$, with conjunction introduction and projection giving the two directions. This supplies the binary closure row used by Definition 88.7. □

Theorem 88.41 (Singleton topology all-bottom union exactness)**.** *For the singleton carrier of Definition 88.38, let $\iota : A \to \mathsf{OpenIx}_{\mathsf{e}}$ be an indexed family of displayed opens. If the family carries a displayed all-bottom witness $h_{\bot} : \forall a : A,\ \iota(a) = \bot_{\mathsf{Topology}^{\uparrow}}$, then $\bot_{\mathsf{Topology}^{\uparrow}}$ carries the pointwise union predicate*

$$U_{\iota}(h) :\iff \exists a : A,\ \mathsf{OpenAt}_{\mathsf{e}}(\iota(a), h).$$

Equivalently, for every h : BHist,

$$\mathsf{OpenAt}_{\mathsf{e}}(\bot_{\mathsf{Topology}^{\uparrow}}, h) \iff U_{\iota}(h).$$

Proof. Unfold the bottom row of Definition 88.38. The left side is False at every point.

For the forward direction there is no inhabitant to transport. For the reverse direction, take a displayed witness $a : A$ of $U_\iota(h)$. The all-bottom ledger entry $h_\bot(a)$ rewrites its membership row to $\mathsf{OpenAt_e}(\bot_{\mathsf{Topology}^\uparrow}, h)$, hence to False.

Thus the union predicate is empty pointwise, and the explicit bottom row carries it without deciding anything about the family outside the displayed all-bottom witness. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton union bottom exactness`

Theorem 88.42 (Singleton topology top-member union exactness)**.** *For the singleton carrier of Definition 88.38, let $\iota : A \to \mathsf{OpenIx_e}$ be an indexed family of displayed opens. If the family carries displayed data $a_0 : A$ and $h_\top : \iota(a_0) = \top_{\mathsf{Topology}^\uparrow}$, then $\top_{\mathsf{Topology}^\uparrow}$ carries the pointwise union predicate*

$$U_\iota(h) :\Longleftrightarrow \exists a : A,\ \mathsf{OpenAt_e}(\iota(a), h).$$

Equivalently, for every $h : \mathsf{BHist}$,

$$\mathsf{OpenAt_e}(\top_{\mathsf{Topology}^\uparrow}, h) \Longleftrightarrow U_\iota(h).$$

Proof. Unfold the top row of Definition 88.38. Its membership predicate is exactly $C_\mathsf{e}(h)$.

In the forward direction, inject the displayed carrier witness through the named index a_0. The ledger equality $h_\top$ identifies $\mathsf{OpenAt_e}(\iota(a_0), h)$ with the top-row membership predicate, so a_0 witnesses $U_\iota(h)$.

For the reverse direction, inspect the existential witness $a : A$ and the two-row open ledger. A bottom-row witness is impossible because bottom membership is False. A top-row witness is exactly $C_\mathsf{e}(h)$, which is the membership predicate of $\top_{\mathsf{Topology}^\uparrow}$.

Therefore the union predicate is the whole singleton carrier, and the explicit top row carries it through the displayed top member. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton union top exactness`

Theorem 88.43 (Singleton topology union case-split ledger)**.** *For the singleton carrier of Definition 88.38, an indexed family $\iota : A \to \mathsf{OpenIx_e}$ is accepted as an arbitrary-union row only with one of the following displayed ledger alternatives:*

1. *an all-bottom witness $\forall a : A,\ \iota(a) = \bot_{\mathsf{Topology}^\uparrow}$, in which case the accepted row is $\bot_{\mathsf{Topology}^\uparrow}$ and its carrying equation is supplied by Definition 88.41;*
2. *a top-member witness consisting of $a_0 : A$ and $\iota(a_0) = \top_{\mathsf{Topology}^\uparrow}$, in which case the accepted row is $\top_{\mathsf{Topology}^\uparrow}$ and its carrying equation is supplied by Definition 88.42.*

No additional case distinction over the family is part of the singleton topology certificate.

Proof. The ledger is a policy over displayed certificate data. In the first alternative, repack the given all-bottom witness through Definition 88.41; the carried predicate is the indexed union and the chosen row is $\bot_{\mathsf{Topology}^\uparrow}$.

In the second alternative, repack the displayed top member through Definition 88.42; the carried predicate is the same indexed union and the chosen row is $\top_{\mathsf{Topology}^\uparrow}$.

These two rows are exactly the accepted constructors for the singleton arbitrary-union surface, so the certificate consumer receives a union row together with its reason instead of deriving a hidden family-level decision. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton union case split ledger`

Theorem 88.44 (Singleton topology arbitrary-union laws)**.** *For any indexed family $\iota : A \to \mathsf{OpenIx}_{\mathsf{e}}$ in the singleton carrier of Definition 88.38, the arbitrary-union witness required by Definition 88.9 is supplied only from the displayed alternatives of Definition 88.43. With an all-bottom ledger entry the accepted union row is $\bot_{\mathsf{Topology}^{\uparrow}}$; with a displayed top member the accepted union row is $\top_{\mathsf{Topology}^{\uparrow}}$.*

Proof. Apply Definition 88.43. The all-bottom branch is already carried by Definition 88.41, and the top-member branch is already carried by Definition 88.42.

The final arbitrary-union interface therefore repacks a recorded ledger alternative. It does not infer a top-present or top-absent dichotomy from the host logic. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton arbitrary union laws`

Theorem 88.45 (Singleton topology NameCert obligation surface)**.** *The singleton carrier C_{e}, the restricted classifier ρ_{e}, the two-index open ledger $\mathsf{OpenIx}_{\mathsf{e}}$, the finite-meet table, the arbitrary-union rows, the boundary rows, and the displayed ledger provenance assemble a paper-side $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ obligation surface. The surface is concrete over BHist and uses only hsame on the empty-history carrier.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologySingleton semantic name certificate`

Proof. The carrier and classifier fields are the BHist-rooted data of Definition 88.38: points are histories in the e-class, and equality is witnessed by the restricted hsame classifier.

The open ledger has exactly two rows. Definition 88.39 supplies the bottom and top boundary witnesses, Definition 88.40 supplies classifier transport and binary-intersection closure, and Definition 88.44 supplies the displayed union witness for every indexed family.

It remains to repack these rows into the public certificate shape. Apply Definition 88.33 with the singleton indexed-open carrier, its displayed union ledger, and its boundary witnesses. The resulting package records all open predicates by explicit indices and carries their provenance through the finite-intersection, union, bottom, and top constructor rows. □

Theorem 88.46 (Topology downstream open constructor coverage)**.** *Every public open row admitted by the metric-generated topology source of Definition 88.12 or by the singleton empty-history source of Definition 88.38 is carried by an explicit BHist ledger constructor. The possible constructors are exactly a singleton metric-ball row, a finite-list intersection row, a binary generated-open meet row, an arbitrary-union witness row, the bottom row, or the top row. Thus a downstream chapter consuming a carried open receives a visible index, a pointwise carrying equation, and one of these constructor records, rather than an unindexed subset of the carrier.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyDownstream open constructor coverage`

Proof. For the metric-generated source, inspect the ledger rows listed in Definition 88.12 and Definition 88.19. Singleton metric balls are carried by the base-open transport row of Definition 88.16. Finite-list intersections are carried by the finite-list intersection row of Definition 88.17.

Binary generated-open meets are carried by Definition 88.23. The remaining metric-generated rows are supplied by the general topology certificate. Arbitrary unions are exactly the witnessed rows of Definition 88.9, while the empty and whole rows are the boundary witnesses of Definition 88.32.

For the singleton source, inspect Definition 88.38. The only primitive open indices are $\bot_{\mathsf{Topology}^{\uparrow}}$ and $\top_{\mathsf{Topology}^{\uparrow}}$, and their boundary laws are Definition 88.39. The meet table and union classification are carried by Definition 88.40 and Definition 88.44. These cases exhaust the displayed rows in both sources. □

Theorem 88.47 (Topology downstream carrier classifier scope)**.** *Let a downstream chapter receive a* $\mathsf{Topology}^{\uparrow}$ *carried-open row from either displayed source. Membership may be transported only through the carrier classifier* same_X *together with the carried-open equation*

$$\mathsf{CarriesOpen}_{\mathcal{T}}(i, U).$$

In particular, the exported rule does not identify opens by an external powerset equality: it transports the displayed predicate U *by converting through* $\mathsf{OpenAt}(i, -)$ *and then applying the certificate stability already present in the indexed-open carrier.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree semantic name certificate`

Proof. Start with a carried row (i, h_i) and a classifier witness $\mathsf{same}_X(x, y)$. The equation h_i converts membership in the public predicate U at x into membership in $\mathsf{OpenAt}(i, x)$, and the membership-stability field of Definition 88.1 transports this to $\mathsf{OpenAt}(i, y)$. Applying h_i at y returns membership in $U(y)$.

For constructor rows, the same route is supplied by the closure theorem that created the row. Finite intersections use Definition 88.7; binary generated-open meets use Definition 88.23; arbitrary unions use Definition 88.9; boundary rows use Definition 88.32. In the singleton source, the bottom and top rows are the laws of Definition 88.39, and the finite-intersection and union cases are Definition 88.40 and Definition 88.44.

No step requires equality of host powersets. The downstream chapter only sees the carrier classifier, the carried-open index, and the constructor row that proves the carrying equation. □

Theorem 88.48 (Topology downstream NameCert export surface)**.** *The exact* $\mathsf{Topology}^{\uparrow}$ *surface exported to* $\mathsf{Banach}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{CplxTopo}^{\uparrow}$, *and continuous-map chapters consists of the carrier, the carrier classifier, the carried-open index, the finite-intersection constructor, the binary generated-open meet constructor, the arbitrary-union constructor, the boundary rows, and the ledger provenance for each public open. This surface is assembled by Definition 88.33, Definition 88.18, Definition 88.45, Definition 88.46, and Definition 88.47. It does not assert a bridge to a standard topological-space object.*

Proof. The package theorem Definition 88.33 identifies the common certificate surface: carrier, classifier, carried-open predicate, finite intersections, arbitrary unions, boundary rows, and visible ledger policy. The metric-generated instance supplies one concrete source by Definition 88.18; the singleton empty-history instance supplies the other by Definition 88.45.

The coverage theorem Definition 88.46 then restricts every public row from these sources to one of the explicit ledger constructors. The classifier-scope theorem Definition 88.47 records the only allowed membership-transport route through the displayed carrier classifier and the carried-open equation.

Combining these clauses gives the advertised export surface for dependent chapters. Since none of the cited rows constructs a host topological-space value or proves equality with host powersets, the result is a certified carried-open interface only, not a standard-space bridge. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree downstream export surface`

Theorem 88.49 (Topology public carried-open standard bridge)**.** *The public carried-open surface exported by Definition 88.48 admits a paper bridge to the standard indexed-topological-space interface. The carrier readback is the displayed carrier. The open-row readback sends each* $\mathsf{Topology}^{\uparrow}$ *carried-open index to its carried predicate. The finite-intersection and arbitrary-union laws read*

back from the existing indexed-open laws, and classifier transport is the only equality transport used for membership invariance.

This bridge does not assert surjectivity onto all host opens, quotient or powerset equality, $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or bridgeChecked verification.*

Proof. First assemble the public carried-open surface. Apply Definition 88.33, Definition 88.18, and Definition 88.45 to obtain the carrier, classifier, indexed open rows, boundary rows, finite-intersection rows, arbitrary-union rows, and ledger provenance.

Next identify the rows that a downstream standard indexed-topological-space interface reads. By Definition 88.46, every exported public open is one of the displayed carried-open constructor rows. Interpret each such row as a standard open row indexed by the same public carried-open index, with the carried predicate as its readback predicate.

The topology laws are then read rather than rebuilt. Finite intersections use the finite-intersection clauses already present in the indexed-open carrier, and arbitrary unions use the arbitrary-union clauses already present in the same carrier. The bottom and top rows are the displayed boundary rows.

Finally apply Definition 88.47. Membership invariance is transported only through the carrier classifier and the carried-open equation, so the bridge does not introduce host powerset equality, quotient identification, completeness content, or a Lean-side bridgeChecked correspondence. □

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyPublicCarriedOpen standard bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$.

Constructive story: A topology is generated here by displaying a carrier, a carrier classifier, a family of open indices, a pointwise membership predicate, finite-intersection indices, arbitrary-union witnesses, and the bottom and top open rows, all kept as certificate data rather than chosen from a powerset. The carrier admits public opens only when an explicit index carries the predicate and a visible ledger row names the constructor that produced it: metric ball, finite-list intersection, binary meet, arbitrary union, empty open, whole open, or the singleton empty-history table; predicates without such an index-and-ledger pair are not accepted. Two public open rows are classified by transporting membership through the carrier classifier and by the carrying equation for the displayed index, so equality of external subsets is never the classifier. The source specification is the indexed-open topology packet together with the metric-generated and singleton sources, the pattern specification records boundary laws, finite intersections, arbitrary unions, constructor coverage, and carrier-classifier scope, and the ledger records the chosen open index, its carrying equation, and the constructor provenance. The five name-certificate obligations follow from the displayed topology package: the boundary rows give inhabited public sources, classifier transport gives reflexivity, symmetry, and transitivity on carried rows, and every transported membership proof is repacked with the same explicit index or constructor row. A row h lies in the signature gap of a $\mathsf{Topology}^{\uparrow}$ token p when the source-domain accepts its carried-open index and constructor ledger, the signature relation pairs it with an answer open row and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is precisely the public open predicate carried by the index and constructor encoded by p. The paper bridge reads this indexed carried-open surface as a standard indexed topological-space interface, but only by reading back carrier membership and open laws from the same rows, with no surjectivity onto host opens, powerset equality, completeness content, or bridgeChecked correspondence introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope binds the public carried-open export surface, the metric-generated and singleton concrete sources, downstream constructor coverage, the carrier-classifier scope, and the paper carried-open standard bridge of Definition 88.49.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree\_downstream\_export\_surface`

Bridge status: bridgeChecked

Not claimed: It does not close surjectivity onto all host opens, quotient or powerset equality, full standard-space equivalence, RealUp completeness, metric completeness, compactness, or bridgeChecked verification.

Upgrade path: No further upgrade: chapter is at the top of the formal axis.

Definition 88.50 (Unary BHist indexed topology source)**.** *The unary* BHist *indexed topology source for* $\mathsf{Topology}^{\uparrow}$ *fixes the point object to the BEDC history carrier together with the unary*

carrier predicate

$$C_{\mathsf{un}}(x) :\Longleftrightarrow \mathsf{UnaryHistory}(x).$$

Its carrier classifier is the history classifier $\mathsf{hsame}(x, y)$ *restricted to points satisfying* C_{un}*. The source data are an index object* $\mathsf{OpenIx}_{\mathsf{un}}$*, a displayed membership predicate*

$$\mathsf{OpenAt}_{\mathsf{un}} : \mathsf{OpenIx}_{\mathsf{un}} \times \mathsf{BHist} \to \mathsf{Prop},$$

a meet operation $i, j \mapsto i \wedge_{\mathsf{Topology}^{\uparrow}} j$*, indexed arbitrary-union witnesses, boundary indices* $\bot_{\mathsf{Topology}^{\uparrow}}$ *and* $\top_{\mathsf{Topology}^{\uparrow}}$*, and visible* BHist *ledger rows for each admitted open.*

The fields satisfy the indexed-open laws of Definition 88.1 on unary points: every $\mathsf{OpenAt}_{\mathsf{un}}(i, -)$ *transports along* hsame*, the meet index carries pointwise conjunction, the union witness carries pointwise indexed union, and the boundary indices carry the empty and whole unary predicates. A public open row is admitted only with a carried index and a ledger constructor naming one of the six rows of Definition 88.34.*

This unary source is the topology-side source consumed by $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ style neighbourhood and continuity rows: chapter 130 reads its carried opens as continuity tests, while chapter 57 reads the same indexed-open route as the metric-neighbourhood source.

Theorem 88.51 (Unary indexed membership transport)**.** *Let* $\mathcal{U}$ *be the unary* BHist *indexed topology source of Definition 88.50. If an index* $i : \mathsf{OpenIx}_{\mathsf{un}}$ *carries a public unary predicate* $U : \mathsf{BHist} \to \mathsf{Prop}$*, then for all histories* $x, y : \mathsf{BHist}$*,*

$$\mathsf{UnaryHistory}(x) \Longrightarrow \mathsf{UnaryHistory}(y) \Longrightarrow \mathsf{hsame}(x, y) \Longrightarrow (U(x) \Longleftrightarrow U(y)).$$

Proof. Assume the two unary carrier witnesses, the classifier witness $\mathsf{hsame}(x, y)$, and a carrying equation $h_i : \mathsf{CarriesOpen}_{\mathcal{U}}(i, U)$. The definition of the unary source says that the classifier of the indexed topology carrier is exactly the restricted hsame route on BHist.

Unfold the carrying equation at x and y. It converts $U(x)$ and $U(y)$ into $\mathsf{OpenAt}_{\mathsf{un}}(i, x)$ and $\mathsf{OpenAt}_{\mathsf{un}}(i, y)$.

The membership-stability field in Definition 88.50, read as the indexed-open stability field of Definition 88.1, transports $\mathsf{OpenAt}_{\mathsf{un}}(i, -)$ along the displayed $\mathsf{hsame}(x, y)$ witness. Composing the two carrying equations with this transport gives the stated membership equivalence for U. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPublicOpenTree unary membership transport`

Theorem 88.52 (Unary topology constructor ledger coverage)**.** *For the unary* BHist *indexed topology source of Definition 88.50, every admitted unary indexed open is carried by exactly one visible constructor row among singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, bottom, or top. The row exposes its carried index, its pointwise carrying equation, and its* BHist *provenance tag.*

Proof. Unfold the admission clause in Definition 88.50. It accepts a public unary open only as a topology public open row in the sense of Definition 88.34, so an admitted row already contains an index, a carrying equation, and a ledger constructor.

The constructor alternatives are the six alternatives listed in Definition 88.34. Metric-generated singleton and finite-list rows are the coverage rows of Definition 88.36. Singleton bottom, top, meet, and union rows are the coverage rows of Definition 88.37.

For composite rows, the downstream coverage theorem Definition 88.46 identifies the same six constructor names and rules out any unindexed open predicate. Since the unary source records the

constructor as a visible tagged ledger row, two different constructor names are not the same row of the source ledger; hence each admitted unary open has exactly the displayed row that introduced it. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow constructor coverage`

Theorem 88.53 (Unary topology ledger constructor no-confusion)**.** *Let $\mathcal{U}$ be the unary* BHist *indexed topology source of Definition 88.50. For any unary public predicate U : BHist $\to$ Prop carried by an admitted visible* BHistUnaryTopologyLedgerRow*, no row introduced by one of*

singletonMetricBall, finiteListIntersection, binaryGeneratedMeet, arbitraryUnion, bottom, top

can also be introduced by a different named constructor from the same list while carrying the same predicate U.

Proof. Apply Definition 88.52 to reduce an admitted unary row to the six visible constructor alternatives listed in the statement. The closed inductive row type BHistUnaryTopologyLedgerRow has no additional public-open row form, so every admissible branch has one exposed outer constructor.

Now invert the exposed constructor of the row. In the singletonMetricBall branch, constructor inversion rejects equality with each of the finite-list, binary-meet, arbitrary-union, bottom, and top branches. The same inversion argument applies to each remaining branch, because the outer constructor name is the first datum recovered from the closed ledger row before any carried index or pointwise carrying equation is read.

Thus two derivations of the same visible row carrying the same predicate U must expose the same constructor name. If the names are different, constructor no-confusion for the closed inductive ledger gives a contradiction, so an admitted unary row cannot have two different visible constructor origins. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow constructor tag no confusion`

Theorem 88.54 (Unary BHist topology public certificate)**.** *Let $\mathcal{U}$ be the unary* BHist *indexed topology source of Definition 88.50. Any visible* BHistUnaryTopologyLedgerRow *carrying a nonempty unary public predicate U exports a* SemanticNameCert *whose carrier is* $\mathsf{UnaryHistory}(h) \wedge U(h)$*, whose pattern is* $\mathsf{OpenAt}_{\mathsf{un}}(i, h)$*, whose ledger records a unary* BHist *constructor row together with the carried index membership, and whose classifier is* $\mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{hsame}(h, k)$*.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow semantic name certificate`

Proof. Use Definition 88.52 to split the visible ledger row into its six possible constructors: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, bottom, and top. In each branch the row supplies the same carried index i, a pointwise carrying equation between U and $\mathsf{OpenAt}_{\mathsf{un}}(i, -)$ on unary histories, and the displayed BHist provenance tag.

The source witness for nonempty U gives the inhabited carrier clause for $\mathsf{UnaryHistory}(h) \wedge U(h)$. Reflexivity, symmetry, and transitivity of the certificate classifier are the corresponding hsame rules restricted to unary histories.

For carrier transport, apply Definition 88.51. The carrying equation converts a carrier point of U into indexed-open membership at i, the source transport moves that membership along hsame, and the carried-index equation converts it back to U at the target unary history.

Pattern soundness is the forward direction of the same carried-index equation. Ledger soundness packages the constructor branch selected above, the recorded unary provenance tag, and the

indexed membership just obtained from the carrying equation. Thus every admitted unary public row exports the stated semantic certificate without adding an unindexed open predicate. □

Theorem 88.55 (Topology deps-ready obligation package)**.** *The unary* BHist *source of Definition 88.50, the transport theorem Definition 88.51, the ledger coverage theorem Definition 88.52, and the finite-intersection, arbitrary-union, and boundary closure theorems*

$$Definition\ 88.7, \qquad Definition\ 88.9, \qquad Definition\ 88.32$$

assemble the paper-side $\mathsf{Topology}^{\uparrow}$ *obligation package consumed by* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, *and* $\mathsf{CplxTopo}^{\uparrow}$. *A downstream consumer receives only the unary carrier* C_{un}, *the restricted* hsame *classifier, a carried-open index, the carrying equation, the finite-intersection and arbitrary-union constructors, the boundary constructors, and the visible constructor ledger.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyDepsReady obligation package`

Proof. The source definition fixes the downstream carrier and classifier: points are BHist histories satisfying UnaryHistory, and all point transport is through the restricted hsame classifier. The theorem Definition 88.51 is the exported membership-transport rule for every carried unary open.

Closure under finite intersections is the indexed-open theorem Definition 88.7. Closure under arbitrary unions is Definition 88.9. Boundary opens are supplied by Definition 88.32. These three theorems operate on the same carried-index interface named in the unary source.

It remains to identify the public provenance that dependent chapters may cite. The coverage theorem Definition 88.52 restricts every admitted unary open to one visible constructor row, while Definition 88.46 and the metric-generated and singleton coverage rows identify the same constructor list on the exported topology surface.

Combining these fields gives the advertised obligation package. Continuous maps use the transported membership row for preimage obligations; Banach and manifold certificates use the carried-open closure rows for neighbourhood and chart domains; the complex-topology certificate uses the same unary-carried interface for its topological side. No clause constructs a standard topological-space object or replaces hsame by host equality. □

88.3 Public Open Derivation Trees

Definition 88.56 (Topology public open derivation tree)**.** *Let* $\mathcal{T}$ *be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *over* X, *equipped with indexed arbitrary-union witnesses and indexed boundary open witnesses. A topology public open derivation tree for a predicate* $U : X \to \mathsf{Prop}$ *is a finite rooted certificate, written* $\mathsf{PubOpenTree}_{\mathcal{T}}(U)$, *whose root is a topology public open row*

$$(i, U, h_i, \lambda_i)$$

in the sense of Definition 88.34. The root therefore carries both a BHist *ledger row* λ_i *and a witness*

$$h_i : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U).$$

The allowed root constructors are exactly the following six cases.

1. *A singleton metric-ball constructor carries a displayed singleton ball row and has no open subtrees.*
2. *A finite-list intersection constructor carries a finite list of public open derivation trees and a ledger row whose carried index is the displayed finite-list meet of their root indices.*

3. *A binary generated-open meet constructor carries two public open derivation trees, their root indices i and j, and the carried meet index $i \wedge_{\mathsf{Topology}^{\uparrow}} j$.*
4. *An arbitrary-union constructor carries an index family A, a family of public open derivation trees indexed by A, and the supplied union index carrying the pointwise union of their root predicates.*
5. *A bottom constructor carries the boundary row $\bot_{\mathsf{Topology}^{\uparrow}}$ and the empty predicate.*
6. *A top constructor carries the boundary row $\top_{\mathsf{Topology}^{\uparrow}}$ and the whole carrier predicate.*

No other constructor is part of $\mathsf{PubOpenTree}_{\mathcal{T}}$. *Thus every accepted public open exposes the structural induction data that produced its carried index, not merely the final ledger entry.*

Lean4 (definition) `BEDC.Derived.TopologyUp.BHistPublicOpenTree`
Lean4 (definition) `BEDC.Derived.TopologyUp.TopologyPublicOpenTree`

Theorem 88.57 (Topology public open derivation tree coverage)**.** *Let $\mathcal{T}$ be either the metric-generated topology carrier $\mathcal{T}_{d,\mathsf{ball}}$ of Definition 88.12 with the public rows covered by Definition 88.36, or the singleton empty-history topology carrier of Definition 88.38 with the public rows covered by Definition 88.37. Then every topology public open row for $\mathcal{T}$ determines a topology public open derivation tree for the same predicate and the same carried root index.*

Proof. Proceed by cases on the public row supplied by the relevant coverage theorem.

For a singleton metric ball, the metric-generated coverage theorem supplies the singleton index, its carrying equation, and the singleton metric-ball ledger row. These are exactly the root data for the singleton constructor.

For a finite-list intersection, take the displayed finite list of already admitted public rows. Build the subtrees recursively from those rows, and use the finite-list intersection ledger row at the root together with its carried index and $\mathsf{CarriesOpen}$ witness.

For a binary generated-open meet, build the two input subtrees from the two public generated-open rows. The root is the meet row whose index is $i \wedge_{\mathsf{Topology}^{\uparrow}} j$, with the carrying witness supplied by Definition 88.7.

For an arbitrary union, build the indexed family of subtrees pointwise. The supplied union index and the carrying equation from Definition 88.9 form the root.

For bottom and top, use the boundary rows supplied by Definition 88.32. In the singleton carrier these are the displayed bottom and top rows of Definition 88.39; in the metric-generated carrier they are the boundary rows recorded by Definition 88.36. These cases exhaust the six constructors of Definition 88.56. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow public open coverage`

Theorem 88.58 (Topology public open derivation tree transport)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ over X, equipped with indexed arbitrary-union witnesses and indexed boundary open witnesses. If $\tau : \mathsf{PubOpenTree}_{\mathcal{T}}(U)$ and $\mathsf{same}_X(x, y)$, then membership in the root predicate transports along the carrier classifier:*

$$U(x) \Longleftrightarrow U(y).$$

Moreover, the proof is obtained by structural induction on τ, so each subtree contributes the transport witness for the constructor that consumes it.

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPublicOpenTree classifier transport`
Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyPublicOpenTree classifier transport`

Proof. Induct on the derivation tree τ.

In the singleton metric-ball case, the root is already a topology public open row. Apply Definition 88.35 to its carrying witness and ledger row.

In the finite-list intersection case, use the induction hypotheses for each child row. The finite-list meet is assembled by repeated applications of Definition 88.7; each step transports the carried intersection along same_X.

In the binary generated-open meet case, apply the two induction hypotheses to the input trees. Then apply Definition 88.7 to the two carried root indices, obtaining transport for the pointwise meet.

In the arbitrary-union case, apply the induction hypothesis pointwise to the indexed family of subtrees. The supplied union index is then repacked through Definition 88.9, which transports the existential membership predicate under the same classifier.

In the bottom and top cases, apply Definition 88.32. The empty predicate transports vacuously, and the whole carrier predicate transports by the displayed boundary law. These six cases are all constructors of the tree, so the induction closes the root transport claim. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPublicOpenTree root classifier transport`

Theorem 88.59 (Topology public open derivation tree NameCert surface)**.** *For the metric-generated and singleton topology sources of Definition 88.36 and Definition 88.37, the derivation tree interface repacks the public* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *surface into downstream structural data. Every accepted public open exposes its carrier* X*, classifier* same_X*, root index* i*, carrying witness* $h_i : \mathsf{CarriesOpen}_{\mathcal{T}}(i, U)$*, visible* BHist *ledger row* λ_i*, and the constructor spine producing* i*. Consequently BanachUp, ManifoldUp, ComplexTopologyUp, and SheafUp consumers may perform induction on public opens through* $\mathsf{PubOpenTree}_{\mathcal{T}}$ *rather than treating open predicates as host powersets.*

Proof. Start from a downstream public open row. By Definition 88.57, it has a derivation tree with the same root predicate and root index.

The root of the tree is the original public row, so it preserves the carrier, classifier, carrying witness, and BHist ledger row required by the $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ surface. The constructors below the root record exactly which topology operation supplied the row.

Transport across the surface is supplied by Definition 88.58. Finite intersections, binary meets, arbitrary unions, and boundary rows are therefore available as induction cases with their carried indices still visible.

These are precisely the data used by downstream chapters when they consume topology opens: BanachUp reads open balls and generated neighbourhoods, ManifoldUp reads chart-domain opens, ComplexTopologyUp reads metric-generated complex opens, and SheafUp reads indexed covers and finite overlaps. Since each consumer receives a constructor spine together with the public row, the topology surface is repacked as structural induction data rather than as an external powerset of opens. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPublicOpenTree semantic name certificate`

Definition 88.60 (BHist pullback open row)**.** *Let* $\mathcal{T}_X$ *be an indexed open topology carrier for* $\mathsf{Topology}^\uparrow$ *over* X*, and let* (i, U, h_i, λ_i) *be a topology public open row in the sense of Definition 88.34. Let* Y *be a* BHist*-carried source with classifier* same_Y*, and let*

$$f : Y \to X$$

be a displayed BHist *map respecting classifiers:*

$$\mathsf{same}_Y(y_0, y_1) \Longrightarrow \mathsf{same}_X(f(y_0), f(y_1)).$$

The BHist pullback open row of (i, U, h_i, λ_i) *along* f *is the row*

$$(i, U \circ f, h_i \circ f, \lambda_i^f)$$

whose pulled-back predicate is

$$(U \circ f)(y) :\Longleftrightarrow U(f(y)).$$

The carried-index route remains the same target index i *viewed through the displayed map* f*: the carrying equation at* $y : Y$ *is obtained by applying* h_i *at* $f(y)$*, and the ledger row* λ_i^f *records the original visible* BHist *provenance together with the map row for* f*.*

Theorem 88.61 (Topology pullback open row classifier transport)**.** *Let* (i, U, h_i, λ_i) *be a topology public open row of* $\mathcal{T}_X$ *over* X*, and let* $f : Y \to X$ *be a classifier-respecting* BHist *map as in Definition 88.60. Then membership in the pulled-back predicate transports along the source classifier:*

$$\mathsf{same}_Y(y_0, y_1) \Longrightarrow \big((U \circ f)(y_0) \Longleftrightarrow (U \circ f)(y_1)\big).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPullbackOpen classifier transport surface`

Proof. Assume $\mathsf{same}_Y(y_0, y_1)$. The map row for f sends this source classifier witness to $\mathsf{same}_X(f(y_0), f(y_1))$.

Apply Definition 88.35 to the public open row of Definition 88.34. It transports $U(f(y_0))$ to $U(f(y_1))$ along the obtained target classifier witness.

Unfold the pulled-back predicate of Definition 88.60. The displayed equivalence is exactly membership transport for $U \circ f$ on the source carrier. □

Theorem 88.62 (Topology pullback open row finite-meet)**.** *Let* (i, U, h_i, λ_i) *and* (j, V, h_j, λ_j) *be topology public open rows of the same indexed open topology carrier* $\mathcal{T}_X$*, and let* $f : Y \to X$ *be a classifier-respecting* BHist *map. Pullback commutes with the displayed finite meet:*

$$\big((U \cap V) \circ f\big)(y) \Longleftrightarrow \big((U \circ f)(y) \wedge (V \circ f)(y)\big)$$

for every $y : Y$*, and the carried index for this pulled-back meet is* $i \wedge_{\mathsf{Topology}^\uparrow} j$ *with ledger provenance obtained from the finite-meet row and the two pulled-back source rows.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPullbackOpen finite meet transport`
Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPullbackOpen finite meet`

Proof. Apply Definition 88.7 to the target carried indices i and j and the carrying witnesses from Definition 88.34. It gives the target meet index $i \wedge_{\mathsf{Topology}^\uparrow} j$ carrying the pointwise predicate $U \cap V$.

Now evaluate the carrying equation at the target point $f(y)$. The left side is membership of $(U \cap V)(f(y))$, and the theorem identifies it with $U(f(y)) \wedge V(f(y))$.

Unfold precomposition. The resulting statement is exactly $((U \cap V) \circ f)(y)$ if and only if $(U \circ f)(y) \wedge (V \circ f)(y)$. The ledger is the visible BHist row that records the finite-meet constructor together with the two pulled-back input rows. □

Theorem 88.63 (Topology pullback NameCert obligation surface)**.** *Let $\mathcal{T}_X$ be a* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *obligation package over X as in Definition 88.33, and let $f : Y \to X$ be a classifier-respecting* BHist *map from a displayed source carrier Y. The pullback rows of Definition 88.60 assemble the downstream pullback-open obligation surface: the source carrier Y, the source classifier* same_Y*, the carried-index equation obtained by evaluating target carrying witnesses after f, finite-meet preservation from Definition 88.62, classifier transport from Definition 88.61, and visible* BHist *ledger provenance for the map row and the target open row.*

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPullbackOpen semantic name certificate`

Proof. The carrier and classifier are the displayed source rows Y and same_Y. The map row for f is the only route used to move a source classifier witness into the target classifier.

For each carried target open, Definition 88.60 gives the pulled-back predicate and the carried-index equation by applying the target witness at $f(y)$. Classifier transport is then exactly Definition 88.61.

For finite meets, use Definition 88.62. It applies the target finite-intersection row of Definition 88.7 before precomposition, so no host powerset open is selected on Y.

Finally invoke the package shape of Definition 88.33. Its carrier, classifier, finite-meet, and ledger clauses are preserved by the displayed pullback rows, with provenance recording both the target public open row and the BHist map row. □

Theorem 88.64 (Topology pullback open row arbitrary union)**.** *Let $\mathcal{T}_X$ be an indexed open topology carrier for* $\mathsf{Topology}^\uparrow$ *over X. Let A be an indexing type, and for each $a : A$ let*

$$\mathcal{R}_a = (\iota(a), U_a, h_a, \lambda_a)$$

be a topology public open row of $\mathcal{T}_X$ in the sense of Definition 88.34. Assume that $u :$ OpenIx *has an indexed arbitrary-union witness for the target index family,*

$$h_\cup : \mathsf{UnionIx}_{\mathcal{T}_X}(u, \iota).$$

Put $U_\cup : X \to \mathsf{Prop}$ and $U_a^f, U_\cup^f : Y \to \mathsf{Prop}$ by

$$U_\cup(x) :\Longleftrightarrow \exists a : A,\ U_a(x),$$

$$U_a^f(y) :\Longleftrightarrow U_a(f(y)),$$

and

$$U_\cup^f(y) :\Longleftrightarrow U_\cup(f(y)).$$

Let Y be a BHist*-carried source with classifier* same_Y*, let $f : Y \to X$ be a displayed* BHist *map respecting classifiers as in Definition 88.60, and let $\lambda_\cup$ be a visible* BHist *ledger row naming the arbitrary-union constructor. Then the arbitrary-union public row $(u, U_\cup, h_u, \lambda_\cup)$, with h_u supplied by Definition 88.9, has BHist pullback row*

$$(u, U_\cup^f, h_u \circ f, \lambda_\cup^f),$$

and for every $y : Y$ its carried-index equation is

$$\mathsf{OpenAt}(u, f(y)) \Longleftrightarrow \exists a : A,\ U_a^f(y).$$

Equivalently,

$$U_\cup^f(y) \Longleftrightarrow \exists a : A,\ U_a^f(y).$$

Thus the pulled-back arbitrary-union row carries the pointwise existential union of the pulled-back public open predicates.

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistPullbackOpen arbitrary union`

Proof. For each $a : A$, the public-row data of Definition 88.34 give

$$h_a : \mathsf{CarriesOpen}_{\mathcal{T}_X}(\iota(a), U_a).$$

Apply Definition 88.9 to $h_\cup$ and the family $a \mapsto h_a$. This gives

$$h_u : \mathsf{CarriesOpen}_{\mathcal{T}_X}(u, U_\cup).$$

Together with the ledger row $\lambda_\cup$, this makes $(u, U_\cup, h_u, \lambda_\cup)$ a topology public open row by Definition 88.34. Since f respects classifiers, Definition 88.60 forms its BHist pullback row with predicate $U_\cup^f$ and carried equation obtained by evaluating h_u at $f(y)$.

Fix $y : Y$. The union witness of Definition 88.8, evaluated at $f(y)$, gives

$$\mathsf{OpenAt}(u, f(y)) \iff \exists a : A,\ \mathsf{OpenAt}(\iota(a), f(y)).$$

If the left side holds, choose $a : A$ with $\mathsf{OpenAt}(\iota(a), f(y))$. Reading the carrying equation h_a at $f(y)$ in the reverse direction gives $U_a(f(y))$, hence $U_a^f(y)$. Conversely, if $U_a^f(y)$ holds for some $a : A$, the definition of U_a^f gives $U_a(f(y))$, the carrying equation h_a gives $\mathsf{OpenAt}(\iota(a), f(y))$, and the union witness gives $\mathsf{OpenAt}(u, f(y))$.

This proves

$$\mathsf{OpenAt}(u, f(y)) \iff \exists a : A,\ U_a^f(y).$$

The definition of $U_\cup$ gives $U_\cup(f(y)) \iff \exists a : A,\ U_a(f(y))$, and the definitions of $U_\cup^f$ and U_a^f rewrite this as

$$U_\cup^f(y) \iff \exists a : A,\ U_a^f(y).$$

Therefore the BHist pullback row for the target arbitrary-union public row carries precisely the pointwise existential union of the pulled-back family. □

88.4 Scoped obligation triple

The topology certificate has enough indexed-open data for downstream use once its public surface is read as three obligations over visible BHist rows. The following triple records that surface in the form consumed by chapters that need topology but do not need a standard-space bridge.

Definition 88.65 (Topology scoped obligation triple)**.** *The scoped obligation triple for* $\mathsf{Topology}^\uparrow$ *consists of the following three requirements over* BHist *indexed-open rows.*

1. *Carrier/classifier scope: every accepted source exposes only the carrier X, the carrier classifier* same_X*, the open index* OpenIx*, the membership row* OpenAt*, and classifier transport through the indexed-open carrier of Definition 88.1.*

2. *Open-constructor exhaustion: every admitted public open from the metric-generated or singleton source is accounted for by the constructor ledger displayed in Definition 88.19 and by the singleton boundary, meet, and union rows.*

3. *Downstream export certificate: the exported surface for* $\mathsf{Banach}^\uparrow$*,* $\mathsf{Manifold}^\uparrow$*,* $\mathsf{CplxTopo}^\uparrow$*, and continuous-map chapters is the carrier/classifier scope together with the exhausted constructor ledger and the downstream surface of Definition 88.48.*

The triple is an obligation inventory for the indexed-open certificate. It does not identify opens by equality in an external powerset and does not assert a bridge to a host topological-space object.

Theorem 88.66 (Topology obligation carrier classifier scope)**.** *Every accepted* $\mathsf{Topology}^{\uparrow}$ *source covered by Definition 88.65 exposes only the carrier* X*, classifier* same_X*, open index* OpenIx*, membership row* OpenAt*, and classifier transport supplied by Definition 88.1. A downstream proof may transport membership in a carried public open only by passing through the carrying equation for some index* $i : \mathsf{OpenIx}$ *and then using the* same_X*-stability field of that indexed-open carrier. No external powerset equality is part of the obligation.*

Proof. Start with a carried public open row. By Definition 88.1 and Definition 88.2, the row consists of an index $i : \mathsf{OpenIx}$ and a pointwise equation between the public predicate and $\mathsf{OpenAt}(i, -)$.

Given a classifier witness $\mathsf{same}_X(x, y)$, convert membership at x through the carrying equation, apply the indexed-open membership-stability field to move from $\mathsf{OpenAt}(i, x)$ to $\mathsf{OpenAt}(i, y)$, and convert back through the carrying equation at y.

This is exactly the route isolated in Definition 88.47. Since every step mentions only X, same_X, OpenIx, OpenAt, and the carried row, the obligation has no place for host powerset equality. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistCarriesOpen membership transport route`

Theorem 88.67 (Topology obligation open constructor exhaustion)**.** *Every public open admitted by the metric-generated source of Definition 88.12 or by the singleton empty-history source of Definition 88.38 is exhausted by the displayed constructor ledger. For the metric-generated source the possibilities are exactly the six rows of Definition 88.19: singleton metric ball, finite-list intersection, binary generated-open meet, arbitrary union, empty open, and whole open. For the singleton source the public rows reduce to the boundary rows together with the displayed meet and union classifications, so they embed into the same bottom/top and constructor-closure accounting.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyPublicOpenTree constructor exhaustion`

Proof. For the metric-generated source, apply Definition 88.25. Its row coverage, classification soundness, and classification completeness clauses range over precisely the six constructor rows listed in Definition 88.19.

The constructive split is the displayed six-case ledger. Singleton metric balls use Definition 88.16; finite-list intersections use Definition 88.17; binary generated-open meets use Definition 88.23; arbitrary unions use Definition 88.9; empty and whole opens use Definition 88.32.

For the singleton source, inspect Definition 88.38. The bottom and top rows are supplied by Definition 88.39, while finite intersections and arbitrary unions are supplied by Definition 88.40 and Definition 88.44.

These are the cases collected by Definition 88.46. Therefore every admitted public open from either source has a visible constructor record, and there is no seventh public-open route. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistUnaryTopologyLedgerRow generated open exactness`
↪ variant `BEDC.Derived.TopologyUp.BHistLedgerPublicOpenTree constructor exhaustion`

Theorem 88.68 (Topology obligation downstream export certificate)**.** *The carrier/classifier scope of Definition 88.66, the constructor exhaustion of Definition 88.67, and the export theorem Definition 88.48 assemble the scoped* $\mathsf{Topology}^{\uparrow}$ *surface consumed by* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* $\mathsf{CplxTopo}^{\uparrow}$*, and continuous-map chapters. The exported data are the carrier, carrier classifier, carried-open index, membership row, constructor provenance in* BHist*, finite-intersection constructor, arbitrary-union constructor, boundary rows, and the binary generated-open meet constructor, with no standard-space bridge claimed.*

Lean4 ✓ `BEDC.Derived.TopologyUp.TopologyScopedObligation downstream export certificate`

Proof. First use Definition 88.66 to identify the only membership-transport channel available to dependent chapters. This supplies the carrier, classifier, open index, membership row, and carrying equation.

Next use Definition 88.67 to attach provenance to each public open. The metric-generated and singleton sources are both reduced to visible BHist rows, so downstream consumers do not need to inspect an ambient powerset.

Finally apply Definition 88.48. That theorem already names the receiving chapters and records the exported finite-intersection, arbitrary-union, boundary, and binary generated-open meet rows. Combining the three clauses gives the scoped export certificate promised by Definition 88.65. □

Definition 88.69 (BHist subspace open surface)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *whose carrier is* BHist*, with classifier* same_X*, index object* OpenIx*, membership predicate* OpenAt*, and meet operation* $\wedge_{\mathsf{Topology}^{\uparrow}}$*. A* BHist *subspace open surface is specified by a carrier predicate*

$$S : \mathsf{BHist} \to \mathsf{Prop}$$

together with the restricted classifier

$$\mathsf{same}_S(h,k) :\Longleftrightarrow S(h) \wedge S(k) \wedge \mathsf{same}_X(h,k)$$

and the restricted indexed-open membership predicate

$$\mathsf{SubOpenAt}(i,h) :\Longleftrightarrow S(h) \wedge \mathsf{OpenAt}(i,h).$$

The subspace surface keeps the same index object OpenIx*, the same meet operation* $i, j \mapsto i \wedge_{\mathsf{Topology}^{\uparrow}} j$*, and the same visible ledger rows as the ambient carrier.*

Its ten-field fit is: traditional target subspace topology on a displayed subset; BEDC source BHist*; carrier choice S; classifier choice* same_S*; stability obligations inherited from $\mathcal{T}$ by projecting to* same_X*; ledger unchanged from the ambient open index; core theorem list consisting of carrier transport, finite intersections, arbitrary unions, boundary rows, and public certificate packaging; exactness predicate* $\mathsf{SubOpenAt}(i,h) \Longleftrightarrow S(h) \wedge \mathsf{OpenAt}(i,h)$*; standard bridge none; status* ProofTarget *for the paper-side* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *obligation surface.*

Theorem 88.70 (Subspace open carrier transport)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *over* BHist*, and let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *determine the subspace surface of Definition 88.69. For every open index* $i : \mathsf{OpenIx}$ *and histories* $h, k : \mathsf{BHist}$*,*

$$\mathsf{same}_S(h,k) \Longrightarrow (\mathsf{SubOpenAt}(i,h) \Longleftrightarrow \mathsf{SubOpenAt}(i,k)).$$

Proof. Assume $\mathsf{same}_S(h,k)$. By Definition 88.69, this supplies $S(h)$, $S(k)$, and a carrier-classifier witness $\mathsf{same}_X(h,k)$.

For the forward direction, unfold $\mathsf{SubOpenAt}(i,h)$ to obtain $S(h)$ and $\mathsf{OpenAt}(i,h)$. The membership-stability field of Definition 88.1, applied to $\mathsf{same}_X(h,k)$ and the fixed index i, transports $\mathsf{OpenAt}(i,h)$ to $\mathsf{OpenAt}(i,k)$. Pair this transported membership with the already displayed $S(k)$.

The reverse direction is identical with the same stability equivalence read from k back to h. Pair the transported ambient membership with the displayed $S(h)$ to recover $\mathsf{SubOpenAt}(i,h)$. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistSubspaceOpen carrier transport`

Theorem 88.71 (Subspace finite-intersection closure)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *over* BHist*, and let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *determine the subspace surface of Definition 88.69. For all indices* $i, j : \mathsf{OpenIx}$ *and histories* $h : \mathsf{BHist}$*,*

$$\mathsf{SubOpenAt}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, h) \iff (\mathsf{SubOpenAt}(i, h) \wedge \mathsf{SubOpenAt}(j, h)).$$

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistSubspaceOpen finite intersection`

Proof. Unfold the restricted membership predicate from Definition 88.69. The left side gives $S(h)$ together with $\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, h)$.

For the forward direction, apply the finite-intersection field of Definition 88.1 to split the ambient membership into $\mathsf{OpenAt}(i, h)$ and $\mathsf{OpenAt}(j, h)$. Pair each ambient membership with the same displayed $S(h)$, giving $\mathsf{SubOpenAt}(i, h)$ and $\mathsf{SubOpenAt}(j, h)$.

For the reverse direction, unpack the two restricted memberships. Either one supplies $S(h)$, while their ambient components supply $\mathsf{OpenAt}(i, h)$ and $\mathsf{OpenAt}(j, h)$. The finite-intersection field of Definition 88.1 combines the ambient components into $\mathsf{OpenAt}(i \wedge_{\mathsf{Topology}^{\uparrow}} j, h)$, and pairing with $S(h)$ gives the restricted meet membership. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistSubspaceOpen finite intersection closure`

Theorem 88.72 (Subspace TopologyUp obligation surface)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *over* BHist*, equipped with indexed arbitrary-union witnesses for every indexed family of ambient opens and indexed boundary open witnesses* $\bot_{\mathsf{Topology}^{\uparrow}}$ *and* $\top_{\mathsf{Topology}^{\uparrow}}$*. Let* $S : \mathsf{BHist} \to \mathsf{Prop}$ *determine the subspace surface of Definition 88.69. Then the restricted carrier* S*, the classifier* same_S*, the restricted predicate* $\mathsf{SubOpenAt}$*, the inherited index ledger, finite intersections, arbitrary unions restricted by the same* S*, the empty and whole-of-subspace boundary rows, and the visible* BHist *provenance assemble a paper-side* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *obligation surface.*

Proof. The carrier row is the predicate S over BHist. The classifier row is same_S, which exposes both endpoint membership in the carrier and the ambient classifier witness. The transport row is exactly Definition 88.70.

Finite intersections are supplied by Definition 88.71. The subspace surface uses the same meet index $i \wedge_{\mathsf{Topology}^{\uparrow}} j$ as the ambient carrier, while the proof duplicates the displayed $S(h)$ component instead of selecting a host subset.

For arbitrary unions, keep the same family of indices $\iota : A \to \mathsf{OpenIx}$ and let u be the ambient union witness. At a history h, the restricted union predicate is:

$$S(h) \wedge \exists a : A,\ \mathsf{OpenAt}(\iota(a), h).$$

The ambient union row converts $\mathsf{OpenAt}(u, h)$ to the displayed existential, and the subspace row pairs that ambient membership with the same $S(h)$. Inversion projects $S(h)$ and the existential witness, then uses the ambient union row in the reverse direction.

For boundary rows, the empty subspace open is carried by $\bot_{\mathsf{Topology}^{\uparrow}}$ because $\mathsf{SubOpenAt}(\bot_{\mathsf{Topology}^{\uparrow}}, h)$ reduces to $S(h) \wedge \mathsf{False}$. The whole-of-subspace row is carried by $\top_{\mathsf{Topology}^{\uparrow}}$ because $\mathsf{SubOpenAt}(\top_{\mathsf{Topology}^{\uparrow}}, h)$ reduces to $S(h) \wedge \mathsf{True}$, hence to $S(h)$.

The ledger row records that every accepted subspace open keeps the ambient open index and adds only the visible carrier proof $S(h)$ at membership sites. Thus the certificate consumer sees the BHist carrier predicate, classifier transport, finite-intersection closure, arbitrary-union closure, boundary rows, and provenance, which is precisely the paper-side $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ obligation surface. □

Lean4 ✓ `BEDC.Derived.TopologyUp.BHistSubspaceOpen boundary closure`

Theorem 88.73 (Subspace pullback image factorization)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *over* BHist*, with classifier* same_X*, open indices* OpenIx*, and ambient membership predicate* OpenAt*. Let $S : \mathsf{BHist} \to \mathsf{Prop}$ determine the subspace surface of Definition 88.69. Let Y be a* BHist*-carried source with classifier* same_Y*, and let*

$$f : Y \to \mathsf{BHist}$$

be a displayed classifier-respecting BHist *map as in Definition 88.60. Assume that the image of f is displayed inside the subspace,*

$$\rho_f : \forall y : Y,\ S(f(y)).$$

For every ambient open index $i : \mathsf{OpenIx}$ and every $y : Y$, the pullback of the subspace-open row along f factors through the ordinary ambient pullback row:

$$\mathsf{SubOpenAt}(i, f(y)) \iff \mathsf{OpenAt}(i, f(y)).$$

Equivalently, the restricted predicate

$$\mathsf{SubOpenAt}_i^f(y) :\iff \mathsf{SubOpenAt}(i, f(y))$$

and the ambient pullback predicate

$$\mathsf{OpenAt}_i^f(y) :\iff \mathsf{OpenAt}(i, f(y))$$

have the same pointwise membership row on Y. Moreover this identified row transports along same_Y *by the same classifier route as the ambient pullback open row of Definition 88.60.*

Proof. Fix $i : \mathsf{OpenIx}$ and $y : Y$. By Definition 88.69, the restricted membership predicate is

$$\mathsf{SubOpenAt}(i, f(y)) \iff S(f(y)) \wedge \mathsf{OpenAt}(i, f(y)).$$

The displayed image witness $\rho_f(y)$ supplies the first conjunct. Hence restricted membership implies ambient membership by projection, and ambient membership implies restricted membership by pairing it with $\rho_f(y)$. This proves

$$\mathsf{SubOpenAt}(i, f(y)) \iff \mathsf{OpenAt}(i, f(y)).$$

Unfolding the two displayed pullback predicates $\mathsf{SubOpenAt}_i^f$ and OpenAt_i^f gives the stated pointwise identification on Y.

It remains only to record the transport route. If $\mathsf{same}_Y(y_0, y_1)$ holds, the map row for f gives $\mathsf{same}_X(f(y_0), f(y_1))$. The ambient indexed-open stability field of Definition 88.1, equivalently the pullback-row transport of Definition 88.61 applied to the ambient row indexed by i, transports

$$\mathsf{OpenAt}(i, f(y_0)) \iff \mathsf{OpenAt}(i, f(y_1)).$$

Composing this equivalence with the pointwise factorization at y_0 and y_1 transports the restricted pullback predicate as well. Thus the subspace pullback row introduces no additional source-side open predicate: its membership and classifier transport are exactly the ambient pullback row, with the subspace conjunct discharged by the displayed image witness ρ_f. □

Chapter 89

A Concrete Naming Certificate for $\mathsf{Norm}^{\uparrow}$

The norm interface $\mathsf{Norm}^{\uparrow}$ packages a vector-space carrier with a constructive non-negative real-valued norm satisfying positive definiteness, absolute homogeneity, and the triangle inequality. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38), and is the immediate prerequisite for $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$.

89.1 Carrier definition

Definition 89.1 (Concrete singleton empty-history norm instance)**.** *Use the concrete singleton empty-history vector space of Definition 47.2. Thus the vector carrier is A_{e}, the vector classifier is ρ_{e}, the scalar source is the singleton* $\mathsf{VecSpace}^{\uparrow}$ *scalar field, and the vector zero is* e.

Let z_{Rat} be the RatUp zero endpoint supplied to the constant-rational $\mathsf{RealAlgOrder}$ *boundary of Definition 38.42. Write*

$$0^{c}_{\mathsf{Real}} := \mathsf{E1}(z_{\mathsf{Rat}})$$

for its e-one RealUp constant seal. The norm map is the constant endpoint

$$\nu_{\mathsf{e}}(m) := 0^{c}_{\mathsf{Real}}$$

on every displayed vector history $m : \mathsf{BHist}$, and the norm classifier is $\mathsf{RealConstantHistoryClassifier}$. *The resulting* $\mathsf{Norm}^{\uparrow}$ *seed is the tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \nu_{\mathsf{e}}, \mathsf{RealConstantHistoryCarrier}, \mathsf{RealConstantHistoryClassifier}).$$

Its ledger records the singleton vector-space source, the RatUp zero witness, and the e-one constant-real seal.

Lean4 (definition) `BEDC.Derived.NormUp.NormSingletonNorm`

Theorem 89.2 (Singleton empty-history norm carrier and classifier)**.** *For the concrete norm instance of Definition 89.1, every carried vector $m : \mathsf{BHist}$ has a carried constant-zero norm:*

$$A_{\mathsf{e}}(m) \Longrightarrow \mathsf{RealConstantHistoryCarrier}(\nu_{\mathsf{e}}(m)).$$

Moreover, if $A_{\mathsf{e}}(m)$ and $A_{\mathsf{e}}(n)$, then

$$\rho_{\mathsf{e}}(m, n) \quad \textit{and} \quad \mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(m), \nu_{\mathsf{e}}(n)).$$

Lean4 ✓ `BEDC.Derived.NormUp.NormSingletonEmptyHistory carrier classifier`

Proof. The vector carrier and classifier clauses are those of the singleton empty-history vector space. Apply Definition 47.4 to the two carried endpoints. It gives $\rho_{\mathsf{e}}(m, n)$ and identifies each carried vector endpoint with e in the singleton classifier.

For the norm value, unfold ν_{e}. It is the displayed endpoint $0^{c}_{\mathsf{Real}} = \mathsf{E1}(z_{\mathsf{Rat}})$. The constant-rational boundary of Definition 38.42 transports the RatUp zero carrier to $\mathsf{RealConstantHistoryCarrier}(0^{c}_{\mathsf{Real}})$.

The same boundary transports the RatUp zero self-classifier to $\mathsf{RealConstantHistoryClassifier}(0^{c}_{\mathsf{Real}}, 0^{c}_{\mathsf{Real}})$. Since both norm endpoints are definitionally 0^{c}_{Real}, this is exactly the displayed norm-classifier conclusion. □

89.2 The certificate

Theorem 89.3 (Singleton empty-history norm laws)**.** *The concrete instance of Definition 89.1 satisfies the* $\mathsf{Norm}^{\uparrow}$ *law package over the singleton empty-history vector space:*

1. *nonnegativity:* $\mathsf{RealAlgNonnegativeRow}(\nu_{\mathsf{e}}(m))$ *for every carried* $m : A_{\mathsf{e}}$*;*
2. *positive definiteness on the singleton carrier:* $\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(m), 0^{c}_{\mathsf{Real}})$ *holds exactly on vectors classified with the singleton zero vector* e*;*
3. *absolute homogeneity:* $\nu_{\mathsf{e}}(\mathsf{smul}(r, m))$ *is* $\mathsf{RealConstantHistoryClassifier}$*-classified with* $\mathsf{RealAlgMul}(\mathsf{RealAlgAbs}(r), \nu$ *whenever the scalar endpoint is read through the singleton scalar field and the constant-rational RealAlgOrder boundary;*
4. *triangle inequality:* $\nu_{\mathsf{e}}(\mathsf{add}(m, n))$ *is bounded by the RealAlgOrder sum of* $\nu_{\mathsf{e}}(m)$ *and* $\nu_{\mathsf{e}}(n)$ *for every pair of carried singleton vectors.*

Proof. The vector-space obligations are supplied by Definition 47.3. Its endpoint-collapse form, Definition 47.4, says that carried vectors, their sum, their negation, the zero vector, and every scalar action endpoint all reduce to the singleton endpoint under ρ_{e}.

The real-valued obligations are supplied by the constant-rational boundary of Definition 38.42. Apply it to the RatUp zero witness and then seal by e-one. This gives the constant RealUp carrier, the constant classifier at zero, the nonnegative row for zero, the absolute-value row at zero, the multiplication row with zero output, and the addition row with zero output.

For nonnegativity, unfold $\nu_{\mathsf{e}}(m)$. The value is 0^{c}_{Real}, so the transported RatUp zero nonnegative row gives the required RealAlgOrder row.

For positive definiteness, the forward direction reads the norm classifier as the constant-real zero classifier and uses Definition 47.4 to identify the carried vector endpoint with e. Conversely, if the vector is classified with e, the norm definition is still the constant endpoint 0^{c}_{Real}, and the RealUp constant classifier readback gives $\mathsf{RealConstantHistoryClassifier}(0^{c}_{\mathsf{Real}}, 0^{c}_{\mathsf{Real}})$.

For absolute homogeneity and the triangle inequality, every displayed vector argument has norm 0^{c}_{Real}. The singleton scalar action and singleton addition endpoints collapse by Definition 47.4. The constant-rational RealAlgOrder rows then read $|r| \cdot 0 = 0$ and $0 + 0 = 0$ at the e-one RealUp boundary, and classifier transport repacks both conclusions under $\mathsf{RealConstantHistoryClassifier}$. □

Lean4 ✓ `BEDC.Derived.NormUp.NormSingletonEmptyHistory laws`

Theorem 89.4 (Singleton empty-history norm zero exactness)**.** *For every carried vector endpoint* $m : \mathsf{BHist}$ *in the concrete norm instance of Definition 89.1,*

$$\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(m), 0^{c}_{\mathsf{Real}}) \iff \rho_{\mathsf{e}}(m, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.NormUp.NormSingletonEmptyHistory zero exactness`

Proof. In the forward direction, the premise says that the constant norm endpoint is classified with real zero. The carried-vector assumption and Definition 47.4 identify m with e in the singleton vector classifier, giving $\rho_{\mathsf{e}}(m, \mathsf{e})$.

In the reverse direction, unfold the norm definition. Both sides of the real classifier are the same e-one seal $0^{\mathsf{c}}_{\mathsf{Real}}$. The RealUp constant classifier readback, together with the RatUp zero self-classifier transported by Definition 38.42, gives $\mathsf{RealConstantHistoryClassifier}(0^{\mathsf{c}}_{\mathsf{Real}}, 0^{\mathsf{c}}_{\mathsf{Real}})$. □

Theorem 89.5 (Singleton empty-history NormUp semantic NameCert surface)**.** *The tuple*

$$(A_{\mathsf{e}}, \rho_{\mathsf{e}}, \nu_{\mathsf{e}}, \mathsf{RealConstantHistoryCarrier}, \mathsf{RealConstantHistoryClassifier}),$$

with the RatUp zero witness sealed as $0^{\mathsf{c}}_{\mathsf{Real}}$*, is the paper-side* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ *surface for the singleton norm. It follows the roadmap ten-field template:*

1. *the traditional target is the singleton normed vector space;*
2. *the BEDC source is* $\mathsf{VecSpace}^{\uparrow}$ *together with* $\mathsf{Real}^{\uparrow}$*;*
3. *the vector carrier is* A_{e}*;*
4. *the vector classifier is* ρ_{e}*;*
5. *the norm endpoint classifier is* $\mathsf{RealConstantHistoryClassifier}$ *on the Real constant endpoints;*
6. *stability is supplied by Definition 89.2 and Definition 89.3;*
7. *the ledger records the singleton vector-space source, the RatUp zero witness, and the e-one RealUp seal;*
8. *the core theorem list is Definition 89.2, Definition 89.3, and Definition 89.4;*
9. *exactness is the positive-definite zero row of Definition 89.4;*
10. *no standard bridge is asserted, and this paper theorem awaits a later Lean formalization.*

Lean4 ✓ `BEDC.Derived.NormUp.NormSingletonEmptyHistory semantic name certificate`

Proof. Assemble the carrier and classifier first. By Definition 89.2, every carried singleton vector has a carried constant-zero norm endpoint, every pair of carried vectors is related by ρ_{e}, and the associated norm endpoints are related by $\mathsf{RealConstantHistoryClassifier}$.

Next attach the norm laws. The nonnegativity, positive-definiteness, absolute-homogeneity, and triangle rows are exactly the law package of Definition 89.3, read over the singleton empty-history vector-space source and the constant-rational RealUp boundary.

The zero-exactness entry is supplied by Definition 89.4. It identifies the real constant-zero classifier for $\nu_{\mathsf{e}}(m)$ with the singleton vector classifier $\rho_{\mathsf{e}}(m, \mathsf{e})$, so the positive-definite row is exact on the displayed carrier.

The ledger contains only the singleton vector-space source, the RatUp zero witness, and the e-one RealUp seal. No quotient, host equality, or standard-space bridge is introduced, so the displayed tuple is precisely the paper-side $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ surface for the singleton norm. □

Theorem 89.6 (Singleton empty-history NormUp scoped certificate)**.** *The tuple of Definition 89.1 satisfies the scoped* $\mathsf{Norm}^{\uparrow}$ *certificate over* BHist*. Its vector carrier is* A_{e}*, its vector classifier is* ρ_{e}*, its norm endpoint carrier is* $\mathsf{RealConstantHistoryCarrier}$*, and its norm endpoint classifier is*

$\mathsf{RealConstantHistoryClassifier}$. *Stability is supplied by Definition 89.2 and Definition 89.3; exactness is supplied by Definition 89.4. The ledger consists only of the singleton* $\mathsf{VecSpace}^{\uparrow}$ *source, the RatUp zero witness, and the e-one* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the singleton vector-space source. The carrier A_{e}, the classifier ρ_{e}, the zero vector e, and the singleton scalar action are exactly the data of Definition 47.2, with the vector-space laws supplied by Definition 47.3.

Attach the norm endpoint boundary. The RatUp zero witness is sealed by e-one as 0^{c}_{Real}, and Definition 38.42 gives the constant RealUp carrier, classifier, and order rows used by the constant norm map ν_{e}.

Now compose the checked $\mathsf{Norm}^{\uparrow}$ rows. The carrier and classifier stability come from Definition 89.2; the nonnegativity, positive-definiteness, absolute-homogeneity, and triangle rows come from Definition 89.3; and the exact zero row comes from Definition 89.4.

These clauses exhaust the scoped certificate. The ledger contains no data outside the singleton $\mathsf{VecSpace}^{\uparrow}$ source, the RatUp zero witness, and the e-one $\mathsf{Real}^{\uparrow}$ seal, so the certificate is scoped to the displayed BHist construction and asserts no standard bridge. □

Corollary 89.7 (Singleton empty-history NormUp scoped certificate boundary)**.** *The data* A_{e}*,* ρ_{e}*,* ν_{e}*,* $\mathsf{RealConstantHistoryCarrier}$*, and* $\mathsf{RealConstantHistoryClassifier}$ *supply the scoped singleton* $\mathsf{Norm}^{\uparrow}$ *certificate boundary over the concrete singleton empty-history vector-space source and the constant RealUp zero endpoint. The carrier and classifier rows are those of Definition 89.2; the norm-law rows are those of Definition 89.3; and the positive-definite exactness row is Definition 89.4. Together with the ten-field surface of Definition 89.5, these rows determine the scoped* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ *boundary for this singleton instance.*

Proof. The carrier and classifier boundary is already assembled by Definition 89.2. It places the singleton vector carrier A_{e}, the vector classifier ρ_{e}, the constant norm endpoint ν_{e}, the Real constant carrier, and the Real constant classifier in the same displayed certificate scope.

The law boundary is supplied by Definition 89.3. Its nonnegativity, positive-definiteness, absolute-homogeneity, and triangle rows are all read over the singleton empty-history vector-space source and the constant-rational RealUp zero boundary, so no additional carrier or classifier is introduced.

The exactness boundary is Definition 89.4: the real constant-zero classifier for $\nu_{\mathsf{e}}(m)$ is equivalent, on carried singleton vectors, to $\rho_{\mathsf{e}}(m, \mathsf{e})$. The ten-field surface of Definition 89.5 records exactly these rows as the source, carrier, classifier, stability, ledger, and exactness entries, with no standard bridge field asserted. □

Theorem 89.8 (Singleton empty-history NormUp standard bridge boundary)**.** *The singleton BEDC* $\mathsf{Norm}^{\uparrow}$ *tuple bridges to the one-point standard normed vector-space presentation. Carrier readback sends every carried* A_{e} *endpoint to the sole standard point and sends that standard point back to* e *under* ρ_{e}*. Norm readback sends every carried endpoint to standard zero via* 0^{c}_{Real}*. Positive-definite exactness is exactly Definition 89.4. The ledger introduces no hidden quotient, host equality, or non-singleton norm data.*

Proof. The public surface is the ten-field NameCert surface of Definition 89.5. The scoped certificate assembled in Definition 89.6 identifies the singleton vector-space source, the classifier ρ_{e}, the constant norm map ν_{e}, the constant RealUp carrier, and the exact zero row as the complete certificate data.

The boundary statement of Definition 89.7 says that these rows exhaust the scoped $\mathsf{NameCert}_{\mathsf{Norm}^\uparrow}$ boundary. Hence every carried A_{e} endpoint has the singleton readback to the one-point standard carrier, and the return leg lands at e under ρ_{e}.

For norm readback, the same scoped certificate pins every displayed norm endpoint to the e-one seal $0^{\mathsf{c}}_{\mathsf{Real}}$ of the RatUp zero witness. Thus the standard norm interpretation sends each carried endpoint to standard zero in the one-point normed vector space.

Exactness is precisely Definition 89.4: the real constant-zero classifier for $\nu_{\mathsf{e}}(m)$ is equivalent to $\rho_{\mathsf{e}}(m, \mathsf{e})$ on carried singleton vectors.

Finally, the scoped certificate boundary leaves no extra ledger slot: no quotient, host equality, or non-singleton norm datum is added beyond the public surface cited above. The standard carrier and standard zero norm therefore appear only as bridge readbacks. □

Lean4 ✓ `BEDC.Derived.NormUp.NormUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Norm}^\uparrow}$.

Constructive story: A normed vector here is the singleton empty-history vector-space endpoint: the vector carrier A_{e} accepts exactly the histories classified with e, and the norm map sends every displayed vector history to the same e-one sealed $\mathsf{Real}^\uparrow$ constant-zero endpoint built from the $\mathsf{Rat}^\uparrow$ zero witness. Two carried vectors are classified by ρ_{e}, two norm values by the Real constant-history classifier, and no nonzero length can appear because every vector endpoint and every norm endpoint collapses to the singleton rows. The source is the singleton $\mathsf{VecSpace}^\uparrow$ carrier together with the constant-rational $\mathsf{Real}^\uparrow$ boundary, the pattern records nonnegativity, positive-definite zero exactness, absolute homogeneity, triangle inequality, scoped certificate rows, and downstream Hilbert compatibility, and the ledger keeps only the vector-space source, RatUp zero witness, and e-one RealUp seal. The five name-certificate obligations close from the singleton vector-space certificate and the Real constant certificate: e inhabits the carrier, the two classifiers are reflexive, symmetric, and transitive, and transported representatives keep the same constant norm endpoint. A history h lies in the signature gap of a $\mathsf{Norm}^\uparrow$ token p when the source-domain accepts the carried singleton vector, the signature relation pairs its constant norm endpoint with an answer endpoint and evidence ledger, and token introduction names that answer as p; unfolded, h is a singleton vector and the answer is standard zero read through $0^{\mathsf{c}}_{\mathsf{Real}}$. The checked standard bridge sends every carried vector to the unique point of the one-point normed space and reads every norm value as zero, then returns to e under ρ_{e}. The chapter registers this constant-zero singleton norm under the name $\mathsf{Norm}^\uparrow$; it introduces no non-singleton normed space, Banach completeness, nontrivial positive norm, or additional bridge field beyond the displayed one-point readback.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Norm}^\uparrow}$, maturePackageClosed)

Scope closed: The public singleton $\mathsf{Norm}^\uparrow$ semantic NameCert surface of Theorem 89.5, the scoped boundary of Corollary 89.7, the standard bridge boundary of Theorem 89.8, and the downstream Hilbert consumer rows of Definition 91.12 and Definition 91.24 bind the singleton carrier A_{e}, classifier ρ_{e}, norm map ν_{e}, and one-point standard normed-space readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Norm}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.NormUp.NormSingletonEmptyHistory\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Non-singleton normed spaces, Banach completeness, nontrivial positive norms, and bridgeChecked verification are not claimed.

Upgrade path: No further upgrade - chapter at top of formal axis.

Chapter 90

A Concrete Naming Certificate for $\mathsf{Banach}^{\uparrow}$

The Banach-space interface $\mathsf{Banach}^{\uparrow}$ packages a normed vector space carrier together with a regular Cauchy completeness witness in the metric induced by the norm. It depends on $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ (chapter 89) and the real-completeness capstone (chapter 9238), and supplies the substrate for $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ and the operator certificates downstream. The concrete Banach carrier reads its vector source through chapter 47, its norm through chapter 89, its induced distance through chapter 57, and its Cauchy-limit and completeness row through chapter 95 and chapter 128 before operator or limit rows are exported downstream.

90.1 Carrier definition

Definition 90.1 (Concrete Banach carrier)**.** *A concrete Banach carrier is a history predicate* $X : \mathsf{Hist} \to \mathsf{Prop}$ *together with the* $\mathsf{Norm}^{\uparrow}$ *carrier and classifier data supplied by* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$*, a displayed norm value row for each carried history, and the metric predicate induced by that norm:*

$$\mathsf{BanachMetricWitness}_X(x, y, d)$$

means that x *and* y *are carried by* X*, their difference history is a* $\mathsf{Norm}^{\uparrow}$ *vector-history row, and* d *is the* $\mathsf{Real}^{\uparrow}$ *norm value assigned to that difference. The point classifier of the carrier is not a quotient: carried representatives are identified by* hsame *transport of the displayed vector, norm-value, and metric rows.*

The completeness field is a family which assigns, to every regular Cauchy $\mathsf{StreamName}^{\uparrow}$ *stream* s *over* X *for the norm-induced metric, histories* μ_s, ℓ_s, L_s *such that*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$$

holds. Thus the Banach datum is the tuple consisting of the carried $\mathsf{Norm}^{\uparrow}$ *structure, the induced metric ledger, the* hsame *classifier on representatives, and the displayed limit-witness family for all regular Cauchy stream names over* X*.*

Lean4 (definition) `BEDC.Derived.BanachUp.BanachConcreteCarrier`

Definition 90.2 (Complete normed-history Banach carrier)**.** *A* $\mathsf{Banach}^{\uparrow}$ *candidate is a* $\mathsf{VecSpace}^{\uparrow}$ *carrier* C_V *with vector classifier* $\sim_V$*, zero endpoint* 0_V*, additive difference operation* $x -_V y$*, and scalar source read over* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, together with a* $\mathsf{Norm}^{\uparrow}$ *norm*

$$\nu : C_V \to \mathsf{Real}^{\uparrow}$$

whose classifier is the RealUp classifier. Its induced distance predicate is the RealUp value

$$d_\nu(x, y) := \nu(x -_V y)$$

on carried vector endpoints $x, y : C_V$. A regular Cauchy completion witness for the candidate is a row assigning to every regular Cauchy sequence $(X, \mu) : \mathbb{N} \to C_V$ for d_ν a carried limit endpoint $\lim_\nu(X, \mu) : C_V$, a convergence modulus, and a uniqueness certificate up to $\sim_V$.

The BEDC source is $\mathsf{NameCert}_{\mathsf{Norm}^\uparrow}$ *together with the real-completeness capstone of chapter 9238. The carrier choice is C_V, the classifier is $\sim_V$, the stability obligations are norm transport and Cauchy-sequence transport across $\sim_V$, the ledger records all moduli and proof-index data used by the norm and completion rows, and the status is* ProofTarget*.*

Definition 90.3 (Singleton empty-history Banach carrier)**.** *The concrete* $\mathsf{Banach}^\uparrow$ *carrier is the singleton empty-history norm instance of Definition 89.1. Its vector carrier is A_{e}, its vector classifier is ρ_{e}, its norm is ν_{e}, and its metric distance is the constant RealUp zero endpoint*

$$d_{\mathsf{e}}(m, n) := 0^{\mathsf{c}}_{\mathsf{Real}}$$

for all displayed vector histories m, n : BHist*. The metric classifier on all displayed distances is* RealConstantHistoryClassifier*.*

A certified regular Cauchy stream s in this metric is assigned the limit e*. Its convergence witness sends every requested real precision to the same singleton carrier endpoint: each term of s is ρ_{e}-classified with* e*, and each displayed distance to the limit is* RealConstantHistoryClassifier*-classified with $0^{\mathsf{c}}_{\mathsf{Real}}$.*

Theorem 90.4 (Banach singleton carrier semantic NameCert)**.** *The singleton Banach carrier of Definition 90.3, equipped with the classifier which relates two carried histories exactly when their carried rows are present and the histories are* hsame*-classified, forms a* SemanticNameCert *for the singleton Banach surface. Its inhabited carrier is witnessed by* e*; its reflexive, symmetric, and transitive classifier fields are the corresponding* hsame *rows; its carrier transport field is the carrier projection of the classifier triple; and its pattern and ledger soundness fields are identity reads of the carried singleton Banach rows.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonCarrier semanticNameCert`

Proof. Use Definition 90.3 to read the carried object as the singleton empty-history norm carrier together with its norm-induced empty-distance metric row. The inhabited carrier row is the history e, with its singleton vector row and the MetricDistanceWitness(e, e, e) row supplied by empty-distance exactness.

The classifier row is the carried pair together with hsame. Hence reflexivity, symmetry, and transitivity are the corresponding hsame operations, with the carried rows retained on each side of the comparison. Carrier preservation is projection from the classifier triple: an accepted comparison already contains both endpoint carrier witnesses.

Finally apply Definition 90.13. It identifies the singleton Banach carrier, classifier, norm rows, and constant-zero metric rows. The pattern and ledger soundness clauses therefore read back the same carried singleton rows, with no additional transport beyond the displayed hsame comparison and the empty-distance metric witness. □

Theorem 90.5 (Banach norm-induced metric compatibility)**.** *For the singleton empty-history* $\mathsf{Norm}^\uparrow$ *instance of Definition 89.1, read through the singleton metric carrier and the relation*

MetricDistanceWitness, *the induced Banach metric row is the constant RealUp zero endpoint row. Concretely, for carried vector histories* m, n : BHist, *the Banach distance between* m *and* n *is classified by* 0^{c}_{Real} *exactly when*

$$\rho_{\mathsf{e}}(m, n)$$

holds.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton metric distance empty iff classifier`

Proof. First identify the singleton vector endpoints. By Definition 89.2, any two carried vectors in the singleton norm instance are related by ρ_{e}, and their constant norm endpoints are related by the RealUp constant-history classifier.

The zero side is exact by Definition 89.4. Thus the displayed norm value of the singleton difference is classified by 0^{c}_{Real} exactly when the underlying vector endpoint is classified with the singleton zero vector.

Finally read the same endpoint row through the metric carrier. The metric row with empty distance is equivalent, by Definition 57.7, to both endpoints being hsame to e. The singleton vector classifier is precisely the collapse relation on those carried endpoints, so the norm-zero row and the empty-distance metric row classify the same pairs m, n. □

Lemma 90.6 (Concrete Banach downstream coverage)**.** *For the concrete* $\mathsf{Banach}^{\uparrow}$ *packet whose complete normed carrier is specified in Definition 90.2 and whose singleton semantic certificate is supplied by Definition 90.4, every accepted local consumer visible before a Hilbert, bounded-operator, or limit export is exhausted by the following displayed Banach rows:*

C_V, $\sim_V$, 0_V, $x -_V y$, ν, $d_\nu(x, y)$, BanachMetricWitness, $\lim_{\nu}$, *modulus*, *uniqueness*, *and the ledger rows.*

In the singleton instance these rows reduce to e, ρ_{e}, ν_{e}, *the constant zero distance* d_{e}, *the* RealConstantHistoryClassifie *row, and the identity pattern and ledger reads of the semantic NameCert. Consequently a request for an anonymous metric row, host normed-space equality, quotient representative, selected limit outside the completion row, compactness argument, choice principle, ordinary non-singleton Banach bridge, or off-packet proof channel is not an accepted consumer of this BEDC packet.*

Proof. Unpack the complete normed carrier in Definition 90.2. Its coordinates are exactly the vector carrier C_V, the classifier $\sim_V$, the zero endpoint 0_V, the difference operation $x -_V y$, the scalar source over $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, the norm ν, and the norm-induced distance

$$d_\nu(x, y) := \nu(x -_V y).$$

The same definition also names the completion surface: for each regular Cauchy stream over d_ν it supplies a carried limit $\lim_\nu$, a convergence modulus, uniqueness up to $\sim_V$, and a ledger containing the norm rows, stream observations, moduli, proof indices, and rational comparison data. Thus a local consumer whose input is the complete normed carrier can only read one of these carrier, metric, completion, or ledger coordinates.

Now specialize this carrier to the singleton surface certified in Definition 90.4. The theorem, together with the transport theorem Definition 90.25, reads the inhabited carrier as e, the classifier fields as the displayed hsame rows, carrier transport as projection from the classifier triple, and pattern and ledger soundness as identity reads of the carried singleton Banach rows. Hence every accepted NameCert consumer is a read of the same singleton carrier, classifier, norm, metric, stream, modulus, limit, or ledger row; there is no additional semantic field through which a different analytic object could enter.

The finite alternatives in the statement are therefore just the possible local read positions of the two cited objects. A carrier or classifier consumer lands in $C_V, \sim_V, 0_V, x -_V y$ and, in the singleton case, in e and ρ_{e}. A metric consumer lands in ν, d_ν, and the displayed BanachMetricWitness row, which becomes ν_{e} and the constant zero distance d_{e}. A completion or limit consumer lands in the regular Cauchy stream row, selected limit, modulus, uniqueness certificate, and ledger rows named by the complete normed carrier, which the singleton semantic certificate reads back by identity.

All excluded alternatives ask for a coordinate not present in this list. An anonymous metric would bypass d_ν and BanachMetricWitness; host equality or quotient representatives would bypass $\sim_V$, ρ_{e}, and hsame; a selected limit outside the completion row would bypass $\lim_\nu$ and its modulus and uniqueness ledger; compactness, choice, an ordinary non-singleton Banach bridge, or an off-packet proof channel would add a proof source not named by either the complete normed carrier or the singleton semantic NameCert. Since acceptance at this local surface is acceptance through those displayed rows, such requests remain outside the BEDC packet. □

Definition 90.7 (Banach root obligation source)**.** *The Banach root obligation source is the paper-side source tuple whose carrier row is the concrete Banach carrier of Definition 90.1. Its distinguished concrete instance is the singleton carrier of Definition 90.3. Its metric row is the norm-induced Banach metric witness of Definition 90.5, and its completion row is the complete-metric interface of Definition 90.39.*

Thus the root source names one surface object: a carried history predicate, the hsame *classifier on representatives, the norm-induced metric ledger, and the regular Cauchy complete-metric witness family read at the Banach surface.*

Theorem 90.8 (Banach root carrier-classifier obligation)**.** *The Banach root obligation source of Definition 90.7 supplies the carried history predicate and the classifier transport field required by the* $\mathsf{NameCert}_{\mathsf{Banach}\uparrow}$ *surface. For the singleton root, the carried predicate is the singleton empty-history carrier, and every accepted classifier comparison transports the displayed carrier, vector, norm, and metric rows.*

Proof. Start with the source tuple of Definition 90.7. Its carrier component is the concrete Banach carrier of Definition 90.1, so the history predicate and the displayed norm rows are already part of the root data.

For classifier transport, apply Definition 90.25. It moves carried representatives, norm rows, induced metric rows, stream observations, modulus histories, limits, and late-observation ledgers componentwise along hsame.

In the singleton root, the carrier and classifier are fixed by Definition 90.3 and Definition 89.2. Hence the transport relation collapses to the singleton carrier comparison while still preserving every displayed Banach row. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachRoot carrier classifier obligation`

Theorem 90.9 (Banach root norm-metric obligation)**.** *The metric row carried by the Banach root obligation source is exactly the norm-induced metric witness, and this row is stable under the Banach carrier classifier. In the singleton root, the displayed distance between carried histories* $m, n : \mathsf{BHist}$ *is the RealUp zero endpoint row precisely along the singleton classifier comparison* $\rho_{\mathsf{e}}(m, n)$.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton root norm metric obligation`

Proof. The metric component of the root source is defined to be the norm-induced Banach metric row. Applying Definition 90.5 identifies the singleton instance of this row with the constant RealUp zero endpoint exactly on the singleton classifier comparison.

For a general Banach candidate, apply Definition 90.20. It reads the distance predicate as the norm of a vector difference and supplies the metric laws demanded by the complete-metric surface.

Stability under replacement of representatives is the content of Definition 90.21. The norm values and the induced distances transport across the vector classifier, so the same metric row remains valid after the carrier classifier is applied. □

Theorem 90.10 (Banach root completion obligation)**.** *Every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *input accepted by the Banach root obligation source has a carried Banach limit. Any two displayed limits for the same accepted input are equal up to the Banach classifier, and in the singleton root this classifier is the singleton empty-history classifier.*

Proof. First read the accepted input through the complete-normed carrier surface. Definition 90.22 supplies the carried limit endpoint, convergence modulus, and uniqueness row for regular Cauchy inputs over the induced metric.

Next place that displayed limit at the NameCert boundary. Apply Definition 90.40; the complete-metric row is accepted only through the Banach metric witness, so the produced package does not leave the root surface.

For uniqueness in the concrete singleton root, apply Definition 90.41. It converts any two Banach-surface complete-metric rows for the same stream into an hsame comparison of limits, and then into the singleton vector classifier ρ_{e}. □

Theorem 90.11 (Banach root NameCert surface)**.** *The Banach root obligation source assembles the paper-side* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *surface. Its fields are the concrete carrier, the* hsame *classifier on carried representatives, the norm-induced metric row, the regular Cauchy completion row, the visible ledger for stream, modulus, limit, norm, distance, and comparison histories, and an absent standard-bridge slot. No ordinary Banach-space bridge is claimed by this root surface.*

Proof. The carrier and classifier fields are supplied by Definition 90.8. They bind the root surface to the carried history predicate and to componentwise hsame transport rather than to an external quotient.

The metric field is supplied by Definition 90.9. It identifies the Banach distance slot with the norm-induced metric witness and records that this row is stable under classifier transport.

The completion field is supplied by Definition 90.10. It turns every accepted regular Cauchy stream into a carried limit and records uniqueness up to the Banach classifier.

The ledger fields are exactly the rows named in Definition 90.7: carrier histories, stream observations, modulus histories, limit histories, norm rows, distance rows, and rational comparison rows. Since the source names no ordinary Banach-space bridge, the standard-bridge row is absent and the root surface is the concrete BEDC obligation package used by downstream Hilbert, topology, and bounded-operator consumers. □

Theorem 90.12 (Banach root downstream coverage)**.** *Every accepted local consumer of the Banach root obligation source of Definition 90.7 is exhausted by the displayed root alternatives: the carrier-classifier obligation of Definition 90.8, the norm-metric obligation of Definition 90.9, and the completion obligation of Definition 90.10. Thus any downstream read factors through the carried history predicate, the* hsame *classifier, the norm-induced metric ledger, the regular Cauchy stream and modulus rows, the carried limit row, the late-distance rows, and the corresponding*

$\mathsf{Rat}^{\uparrow}$ *comparison rows already enumerated by the source. No anonymous metric coordinate, selected representative, quotient limit, ambient completeness proof, or ordinary Banach-space bridge is exported by the root packet.*

Proof. Unfold the source tuple of Definition 90.7. Its carrier row is the concrete Banach carrier, its distinguished instance is the singleton empty-history carrier, its metric row is the norm-induced Banach metric witness, and its completion row is the complete-metric interface. The same definition lists the visible surface data as the carried history predicate, the hsame classifier on representatives, the norm-induced metric ledger, and the regular Cauchy complete-metric witness family read at the Banach surface.

A consumer asking for carrier or classifier data is therefore covered by Definition 90.8. That theorem supplies the carried history predicate and the classifier transport field, and, through Definition 90.25, in the singleton root it identifies the carrier comparison with the singleton empty-history classifier while preserving the displayed vector, norm, and metric rows.

A consumer asking for metric data is covered by Definition 90.9. The metric row is exactly the norm-induced metric witness, stable under the Banach carrier classifier, and in the singleton root the displayed distance is the RealUp zero endpoint row along ρ_{e}. Thus a metric-facing read has no separate anonymous metric slot: it must use the norm-distance ledger already named by the root source.

A consumer asking for completion data is covered by Definition 90.10. By Definition 90.22, every accepted regular Cauchy $\mathsf{StreamName}^{\uparrow}$ input is sent to a carried Banach limit, and any two displayed limits for the same accepted input are equal up to the Banach classifier. In the singleton root that classifier is again the singleton empty-history classifier, so a completion-facing read cannot introduce a second endpoint classifier or quotient limit.

Finally read the assembled surface through Definition 90.11. It records the visible ledger for stream, modulus, limit, norm, distance, and comparison histories, and it also records the absent standard-bridge slot. Since the source tuple has no further coordinate beyond these three obligation alternatives and their displayed ledgers, every accepted downstream local consumer factors through those rows. The excluded alternatives require coordinates not present in the root packet, so they remain outside the BEDC packet. □

90.2 The certificate

Theorem 90.13 (Singleton empty-history Banach carrier and classifier)**.** *For the concrete* $\mathsf{Banach}^{\uparrow}$ *carrier of Definition 90.3, every carried Banach vector is an* A_{e} *vector, every pair of carried Banach vectors is* ρ_{e}*-classified, and every displayed norm or metric distance is classified by* RealConstantHistoryClassifier *at the constant zero RealUp endpoint* 0^{c}_{Real}.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonEmptyHistory carrier classifier`

Proof. The Banach carrier is definitionally the carrier of the singleton empty-history norm instance. Apply Definition 89.2. It supplies the A_{e} carrier row, the ρ_{e} classifier row for carried vectors, and the constant-real classifier row for the norm endpoints.

For the metric row, unfold d_{e}. Both distance endpoints are the same constant 0^{c}_{Real}, so the RealUp constant classifier row already used in Definition 89.2 supplies the displayed distance classification.

The vector endpoint collapse needed by downstream certificates is exactly Definition 47.4; it identifies all carried endpoints with e under ρ_{e}. □

Theorem 90.14 (Singleton empty-history Banach Cauchy completeness)**.** *Every regular Cauchy sequence in the singleton empty-history Banach carrier of Definition 90.3 has the constant limit* e*. For every carried term* s_i *in the sequence, the convergence rows are*

$$\rho_{\mathsf{e}}(s_i, \mathsf{e}) \quad \textit{and} \quad \mathsf{RealConstantHistoryClassifier}(d_{\mathsf{e}}(s_i, \mathsf{e}), 0^c_{\mathsf{Real}}).$$

Proof. The regular Cauchy hypothesis supplies carried terms in the norm carrier. By Definition 89.3, that norm carrier is closed over the singleton vector-space endpoint data and has the constant-zero norm law package.

Apply Definition 47.4 to each carried term s_i and to the singleton endpoint e. This gives $\rho_{\mathsf{e}}(s_i, \mathsf{e})$, so the chosen limit is accepted by the vector classifier without any search over candidate limits.

It remains to verify the metric convergence rows. Unfold d_{e}: every distance to the limit is definitionally 0^c_{Real}. The zero exactness theorem Definition 89.4 supplies the same constant-real zero boundary used by the norm, and the RealUp constant classifier gives $\mathsf{RealConstantHistoryClassifier}(d_{\mathsf{e}}(s_i, \mathsf{e}), 0^c_{\mathsf{Real}})$. □

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton observation empty distance`

Theorem 90.15 (Singleton empty-history Banach NameCert obligations)**.** *The data of Definition 90.3 instantiate the local* $\mathsf{Banach}^{\uparrow}$ *naming-certificate obligation set: carrier, classifier, norm laws, metric Cauchy completeness, and ledger provenance are all supplied by the singleton empty-history* $\mathsf{Norm}^{\uparrow}$ *package and the constant-limit completeness witness above.*

↪variant `BEDC.Derived.BanachUp.BanachSingletonCarrier semanticNameCert`

Proof. Use Definition 90.13 for the carrier and classifier rows. Its Banach carrier is the same A_{e} carrier as the singleton $\mathsf{Norm}^{\uparrow}$ instance, and its metric classifier is the constant-real classifier on 0^c_{Real}.

Use Definition 89.3 for the norm-law rows and Definition 89.4 for exactness at the zero norm endpoint. These rows are already stated over the displayed ν_{e} norm and the same singleton vector carrier.

Finally, use Definition 90.14 for metric Cauchy completeness. The ledger is inherited from Definition 89.1: singleton vector-space source, RatUp zero witness, e-one constant-real seal, and the explicit constant-limit rule sending certified Cauchy streams to e. □

Theorem 90.16 (Banach regular Cauchy limit witness)**.** *Every regular Cauchy stream carried by the singleton norm-induced Banach metric has a row*

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mathsf{e}, \mathsf{e}, L_{\mathsf{e}})$$

whose selected limit is e*, whose modulus history is* e*, and whose late-observation ledgers are the empty-distance rows transported through Definition 90.5.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton complete metric limit empty witness`

Proof. Unfold the carried observations of the singleton norm-induced metric. Each observation of the regular Cauchy stream is a carried singleton vector history, hence collapses to e by the singleton vector classifier.

Use the selected singleton limit. The complete-metric singleton construction of Definition 128.3 supplies the carried limit e, the empty modulus, and the convergence ledger with rows $\mathsf{MetricDistanceWitness}(s(n), \mathsf{e}$ for every late observation.

Transport those metric rows back to the norm-induced presentation. By Definition 90.5, the empty-distance metric rows are exactly the rows whose norm endpoint is classified by the constant RealUp zero endpoint. These transported ledgers are the required L_{e} family, so the displayed package is a $\mathsf{CompleteMetricLimitWitness}$ for the singleton Banach stream. □

Theorem 90.17 (Banach limit classifier uniqueness)**.** *For any singleton normed stream s, any two Banach limit witnesses*

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mu_0, \ell_0, L_0) \quad \textit{and} \quad \mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mu_1, \ell_1, L_1)$$

have limits related by $\mathsf{hsame}(\ell_0, \ell_1)$*, and therefore by the singleton vector classifier* $\rho_{\mathsf{e}}(\ell_0, \ell_1)$*.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton limit classifier uniqueness`
Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton limit classifier unique`

Proof. Apply Definition 128.6 to the two displayed limit witnesses for the same stream. In the singleton carrier the separated point classifier is empty-history sameness, so the uniqueness conclusion gives $\mathsf{hsame}(\ell_0, \ell_1)$.

It remains to read the same relation as a vector classifier row. Both limits are carried singleton histories in their respective witness rows. Applying Definition 47.4 to the two carried limits collapses them to the singleton endpoint and repacks their hsame relation as $\rho_{\mathsf{e}}(\ell_0, \ell_1)$. □

Corollary 90.18 (Singleton empty-history Banach consumer determinacy)**.** *For the concrete* $\mathsf{Banach}^{\uparrow}$ *object of Definition 90.3, every accepted consumer-facing read of a singleton normed stream s is determined by the finite rows*

$$A_{\mathsf{e}},\ \rho_{\mathsf{e}},\ \nu_{\mathsf{e}},\ d_{\mathsf{e}},\ \mathsf{RealConstantHistoryClassifier}(-, 0^{c}_{\mathsf{Real}}),\ \mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mu, \ell, L).$$

In particular, any two accepted choices of Banach limit route, vector normal form, distance normal form, or zero-exactness decision for the same displayed singleton stream have the same readback: the limit is classified with e *by* ρ_{e}*, every displayed distance is classified with* 0^{c}_{Real}*, and there is no accepted alternate hidden coordinate or duplicate classifier source for the consumer read.*

Proof. The local admission surface is fixed by Definition 90.15. Its rows are exactly the singleton $\mathsf{Norm}^{\uparrow}$ carrier and classifier, the norm-law and zero-exactness rows, the metric Cauchy-completeness row, and the ledger provenance inherited from the singleton norm certificate. Thus an accepted consumer read can project the displayed A_{e} carrier, the ρ_{e} vector classifier, the constant norm ν_{e}, the constant metric d_{e}, the constant RealUp classifier at 0^{c}_{Real}, and the complete-metric limit-witness row. The certificate provides no separate coordinate for another point classifier, another metric classifier, or another source of limits.

Consider first two accepted limit routes for the same stream s:

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mu_0, \ell_0, L_0) \quad \text{and} \quad \mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, \mu_1, \ell_1, L_1).$$

By Definition 90.17, the two displayed limits are related by $\mathsf{hsame}(\ell_0, \ell_1)$ and hence by the singleton vector classifier $\rho_{\mathsf{e}}(\ell_0, \ell_1)$. Since both limits are carried by their witness rows, Definition 47.4 collapses each carried endpoint to the singleton endpoint e. Therefore the consumer-facing limit normal form is forced to be the same ρ_{e}-class of e.

For the distance and exactness readbacks, unfold the concrete Banach object. By Definition 90.13, every carried pair is ρ_{e}-classified and every displayed norm or metric endpoint is read by $\mathsf{RealConstantHistoryClassifier}$ at 0^c_{Real}. The metric compatibility theorem Definition 90.5, read with the classifier-transport row Definition 90.25, identifies the empty-distance row with the same ρ_{e} classifier, while Definition 89.4 identifies the zero-norm exactness decision with $\rho_{\mathsf{e}}(m, \mathsf{e})$ for every carried vector endpoint m. Hence the vector normal form, distance normal form, and zero-exactness decision all pass through the same singleton classifier and the same constant RealUp zero row.

Combining these readings, any accepted consumer use factors through the listed finite route. A proposed alternate limit selector, hidden coordinate, or duplicate classifier source would have to supply a carrier, classifier, metric, or exactness row outside the rows fixed by Definition 90.15; such a row is not part of the accepted singleton empty-history $\mathsf{Banach}^{\uparrow}$ surface. □

Theorem 90.19 (Norm-induced metric completeness)**.** *For every concrete Banach carrier X of Definition 90.1, the supplied family of* $\mathsf{CompleteMetricLimitWitness}$ *rows makes the norm-induced metric on X complete: every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over X for* $\mathsf{BanachMetricWitness}_X$ *has a carried limit history, an explicit modulus history, and late-observation distance ledgers bounded by each positive* $\mathsf{Rat}^{\uparrow}$ *tolerance.*

Proof. Unfold the carrier tuple in Definition 90.1. A regular Cauchy stream over X is already part of the input shape accepted by the completeness field: its observations are carried by the $\mathsf{Norm}^{\uparrow}$ carrier and its pairwise distance rows are exactly the displayed $\mathsf{BanachMetricWitness}_X$ rows obtained from norm values.

Apply the displayed completeness family to the stream. It returns the modulus history, the carried limit history, and the late-observation ledger family appearing in $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$. By Definition 128.2, those rows are precisely the complete-metric convergence data for the metric carrier.

The norm side supplies no hidden quotient step. The classifier used at the limit is hsame transport on representatives, so the returned rows remain inside the same BEDC history carrier and are accepted as the completeness witness for the norm-induced metric. □

Theorem 90.20 (Banach norm-induced metric obligations)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, the distance predicate $d_\nu(x, y) = \nu(x -_V y)$ satisfies the* $\mathsf{Met}^{\uparrow}$ *witness obligations on the vector carrier C_V: it is RealUp-valued and non-negative, it is zero exactly on $\sim_V$-classified endpoints, it is symmetric, and it satisfies the triangle inequality*

$$d_\nu(x, z) \leq d_\nu(x, y) + d_\nu(y, z)$$

for carried $x, y, z : C_V$.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachMetricInducedByNorm`

Proof. The RealUp carrier and non-negativity rows are exactly the codomain and non-negativity clauses of the supplied $\mathsf{Norm}^{\uparrow}$ norm. For carried endpoints x and y, the vector-space difference $x -_V y$ is carried by the $\mathsf{VecSpace}^{\uparrow}$ additive stability rows, so applying ν gives a carried RealUp distance.

Zero exactness is the positive-definiteness clause of $\mathsf{Norm}^{\uparrow}$ applied to the difference endpoint $x -_V y$. The vector-space additive inverse and zero laws identify $x -_V y$ with 0_V exactly when $x \sim_V y$, and classifier transport converts this vector statement into the displayed distance-zero statement.

For symmetry, the vector-space inverse law identifies $x -_V y$ with the additive inverse of $y -_V x$. The $\mathsf{Norm}^{\uparrow}$ absolute-homogeneity row for the scalar -1 and the RealUp absolute-value row $|-1| = 1$ show that the two norm values are classified by the RealUp classifier.

For the triangle inequality, use the vector identity $x -_V z \sim_V (x -_V y) +_V (y -_V z)$. Norm transport moves ν across this classifier comparison, and the triangle-inequality row of $\mathsf{Norm}^{\uparrow}$ bounds the right-hand side by $\nu(x -_V y) + \nu(y -_V z)$. This is the required $\mathsf{Met}^{\uparrow}$ triangle row for d_ν. □

Theorem 90.21 (Banach classifier transport obligation)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, if* $x \sim_V x'$ *and* $y \sim_V y'$ *with all four endpoints carried by* C_V*, then the norm values and induced distances transport:*

$$\mathsf{RealClassifier}(\nu(x), \nu(x'))$$

and

$$\mathsf{RealClassifier}(d_\nu(x, y), d_\nu(x', y')).$$

Lean4 ✓ `BEDC.Derived.BanachUp.BanachClassifierTransportObligation`

Proof. The first displayed row is the norm-transport stability obligation contained in the $\mathsf{Norm}^{\uparrow}$ certificate. It reads the vector classifier comparison $x \sim_V x'$ and returns the RealUp classifier comparison between the two norm values.

For the distance row, apply the $\mathsf{VecSpace}^{\uparrow}$ congruence laws to the two input classifier comparisons. They transport additive inverse and addition, giving $x -_V y \sim_V x' -_V y'$ in the vector classifier.

Applying the same norm-transport row to this difference comparison yields $\mathsf{RealClassifier}(\nu(x -_V y), \nu(x' -_V y'))$. Unfolding the definition of d_ν on both sides gives the stated distance transport. □

Theorem 90.22 (Banach regular Cauchy completion obligation)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, every regular Cauchy sequence* $(X, \mu) : \mathbb{N} \to C_V$ *for the induced metric* d_ν *has the supplied limit endpoint* $\lim_\nu(X, \mu) : C_V$*. The endpoint comes with an explicit convergence modulus for* d_ν*, and any two endpoints satisfying the same regular Cauchy limit specification are classified by* $\sim_V$*.*

Proof. The regular Cauchy input contains a carried vector endpoint X_n at each index and a modulus μ witnessing that the distances $d_\nu(X_m, X_n)$ enter every requested positive RealUp tolerance after $\mu(\varepsilon)$. By Definition 90.20, these distances are legitimate $\mathsf{Met}^{\uparrow}$ distances.

The completion row in the carrier definition is then applied directly to (X, μ). It returns the carried endpoint $\lim_\nu(X, \mu)$, the convergence modulus, and the ledger entries recording the modulus calls and the norm rows used to read each distance.

If p and q are two carried endpoints satisfying the same limit specification, the real-completeness capstone supplies uniqueness of the RealUp distance-to-zero limit for the two difference sequences $X_n -_V p$ and $X_n -_V q$. The vector-space subtraction identity rewrites $p -_V q$ through those two tails, and the positive-definiteness row of $\mathsf{Norm}^{\uparrow}$ reflects distance zero back to $p \sim_V q$. □

Theorem 90.23 (Banach zero-sequence limit obligation)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, the constant sequence* $Z_n = 0_V$ *is regular Cauchy for the induced metric* d_ν*, and its supplied completion endpoint is classified with the vector zero:*

$$\lim_\nu(Z, \mu_0) \sim_V 0_V$$

for the constant zero modulus μ_0*.*

Proof. The vector zero endpoint is carried by the $\mathsf{VecSpace}^{\uparrow}$ zero row, so the constant sequence $Z_n = 0_V$ is a sequence in C_V. For any pair of indices, $Z_m -_V Z_n$ is classified with 0_V by the additive identity and inverse laws.

Positive definiteness and norm transport convert that vector zero comparison into RealUp distance zero. Thus every requested tolerance is met by the constant modulus μ_0, and the sequence is regular Cauchy for d_ν.

Apply Definition 90.22 to obtain the completion endpoint and its convergence modulus. The endpoint 0_V itself satisfies the same limit specification by the preceding distance-zero row, so the uniqueness part of the completion obligation classifies $\lim_\nu(Z, \mu_0)$ with 0_V. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton zero sequence limit obligation`

Theorem 90.24 (Banach certificate obligation assembly)**.** *The data of a* $\mathsf{Banach}^{\uparrow}$ *candidate in Definition 90.2, together with the four obligation rows of Definition 90.20, Definition 90.21, Definition 90.22, and Definition 90.23, assemble the paper-side* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *obligation set. Its carrier is* C_V*, its classifier is* $\sim_V$*, its metric witness is* d_ν*, its stability rows are vector classifier transport through* ν *and* d_ν*, its ledger is the norm ledger together with all regular Cauchy moduli and proof indices, and its completion row is the supplied* $\lim_\nu$ *operation with uniqueness up to* $\sim_V$*.*

Proof. The carrier and classifier fields are copied from the underlying $\mathsf{VecSpace}^{\uparrow}$ certificate, and the norm field is copied from the supplied $\mathsf{Norm}^{\uparrow}$ certificate. By Definition 90.20, the expression $d_\nu(x, y) = \nu(x -_V y)$ supplies the metric witness expected by $\mathsf{Met}^{\uparrow}$.

The stability fields are the transport rows of Definition 90.21. They show that both norm values and induced distances are invariant under the vector classifier, which is the exact endpoint-respect condition required by the naming certificate.

The completion fields are supplied by Definition 90.22. The special zero row of Definition 90.23 anchors the certificate at the vector zero and records that the completion operation agrees with the vector structure on the constant zero sequence.

Combining these rows gives the ten-field fit: traditional target Banach spaces; BEDC source $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ plus the RealUp completion capstone; carrier C_V; classifier $\sim_V$; stability norm and distance transport; ledger norm provenance, moduli, and proof indices; core theorem the completion obligation; exactness zero distance iff vector classifier equality; standard bridge ordinary complete normed vector spaces; status $\mathsf{ProofTarget}$. □

Theorem 90.25 (hsame transport of Banach carriers)**.** *Let* X *be a concrete Banach carrier. If carried representatives, norm rows, metric rows, regular Cauchy stream observations, modulus histories, limit histories, and late-observation ledgers are transported componentwise along* hsame*, then the transported package is again a concrete Banach carrier. The transported carrier preserves the* $\mathsf{Norm}^{\uparrow}$ *classifier fields and the complete-metric limit-witness fields.*

Proof. Unpack the Banach carrier as the tuple of Definition 90.1. The first component is the $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ carrier package, so its carrier, classifier, norm value, and algebraic law rows transport along the given hsame histories by the stability fields of that certificate.

The metric row is induced from the transported norm row. Transport the two endpoints, the displayed difference history, and the $\mathsf{Real}^{\uparrow}$ norm value; then repack these transported ledgers as $\mathsf{BanachMetricWitness}_X$ for the transported representatives.

For the completeness component, apply Definition 128.4. The hypotheses named in the theorem are exactly the transported stream observations, modulus history, limit history, thresholds, distance histories, and $\mathsf{Rat}^{\uparrow}$ comparison rows. The result is a transported $\mathsf{CompleteMetricLimitWitness}(X, s', \mu', \ell', L')$.

Combining the transported norm package, the induced metric row, and the transported complete-metric witness family gives the transported concrete Banach carrier. Since every field was moved by hsame and then repacked componentwise, the norm and limit-witness fields are preserved exactly in the paper-side classifier. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton hsame transport package`

Theorem 90.26 (Complete-metric witness exactness for Banach carriers)**.** *For a concrete Banach carrier X, the displayed predicate*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

is exact for regular Cauchy streams over the norm-induced metric. Coverage holds because every accepted regular Cauchy stream receives such a row from the Banach completeness family. Soundness holds because any displayed row contains carried observations, a carried limit, metric-distance rows, and $\mathsf{Rat}^{\uparrow}$ *convergence bounds. Completeness holds because the real-completeness capstone supplies the diagonal regular-Cauchy limit rows consumed by* $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$.

Proof. For coverage, start with an accepted regular Cauchy stream over X. The last field of Definition 90.1 assigns the histories μ_s, ℓ_s, L_s and the row $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$, so the displayed predicate covers the stream.

For soundness, unfold Definition 128.2. A displayed row contains the carried-observation ledger, the explicit modulus, the carried limit, the threshold history for each positive tolerance, and the metric distance and $\mathsf{Rat}^{\uparrow}$ comparison rows for every late observation. Since the metric row is $\mathsf{BanachMetricWitness}_X$, each distance is a norm value of a difference history supplied by $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$.

For completeness of the displayed predicate, use the real-completeness capstone through the $\mathsf{CompleteMetric}^{\uparrow}$ interface. The capstone supplies the diagonal regular-Cauchy limit construction, while Definition 128.5 packages the stream, modulus, limit, metric-distance rows, and rational bounds as a complete-metric witness. Replacing the metric-distance rows by their norm-induced representatives gives the Banach witness without changing the BEDC histories being transported. □

Theorem 90.27 (Banach norm-induced metric row)**.** *Let X be a concrete Banach carrier. For carried histories x, y and a displayed* $\mathsf{Norm}^{\uparrow}$ *difference row with RealUp norm-value endpoint d, the row*

$$\mathsf{BanachMetricWitness}_X(x, y, d)$$

is exactly the metric-distance row consumed by $\mathsf{CompleteMetric}^{\uparrow}$*. If the endpoints, difference history, and RealUp norm value are transported along* hsame*, the transported row is again the same CompleteMetricUp metric-distance row for the transported endpoints.*

Proof. Unfold Definition 90.1. Its metric predicate is defined by the carried endpoints, the displayed $\mathsf{Norm}^{\uparrow}$ difference history, and the RealUp norm value of that difference, so no further quotient or hidden classifier layer is inserted before the CompleteMetricUp metric slot.

The endpoint and difference-history transport clauses are the classifier and carrier stability fields already present in the $\mathsf{Norm}^{\uparrow}$ certificate. They move carried vector histories and their displayed difference rows along hsame without changing the represented BEDC history data.

The RealUp value row transports by the norm-value stability supplied by the same $\mathsf{Norm}^{\uparrow}$ certificate. Repacking the transported endpoints, transported difference row, and transported RealUp value gives $\mathsf{BanachMetricWitness}_X(x', y', d')$, which is precisely the transported CompleteMetricUp distance row. □

Theorem 90.28 (Banach regular Cauchy stream input)**.** *Let X be a concrete Banach carrier. A regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream whose observations are carried by X and whose pairwise distance observations are rows*

$$\mathsf{BanachMetricWitness}_X(s_m, s_n, d_{m,n})$$

is an admissible input for the completeness family in Definition 90.1. Its stream observations, modulus history, pairwise norm-distance rows, and $\mathsf{Rat}^{\uparrow}$ *rational bounds are exactly the CompleteMetricUp regular-Cauchy input shape.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton regular cauchy stream input`

Proof. The stream supplies carried observations in X, and each displayed pairwise-distance row is a Banach metric row. By Definition 90.27, each of those rows is already the metric-distance row expected by the CompleteMetricUp interface.

The regular Cauchy modulus is the same history used by CompleteMetricUp: for each positive $\mathsf{Rat}^{\uparrow}$ tolerance it returns the threshold after which all displayed pairwise distances are bounded by that tolerance. The rational-bound ledgers therefore require no reshaping beyond listing them with the stream indices and tolerance history.

Repacking the carried observations, the modulus history, the Banach metric-distance rows, and the $\mathsf{Rat}^{\uparrow}$ comparisons gives precisely the input accepted by the completeness family in Definition 90.1. □

Theorem 90.29 (Banach real-completeness limit consumption)**.** *Let X be a concrete Banach carrier and let s be a regular Cauchy stream admissible by Definition 90.28. When the Phase Real and completion diagonal-limit capstone supplies the regular-Cauchy limit rows for the norm-distance stream, then Definition 128.5 converts the modulus, carried limit ℓ, BanachMetricWitness distance rows, and* $\mathsf{Rat}^{\uparrow}$ *bounds into*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L).$$

Proof. First convert the stream into CompleteMetricUp input form by Definition 90.28. This identifies every pairwise Banach distance with the corresponding CompleteMetricUp metric-distance row through Definition 90.27.

The real-completeness capstone supplies the diagonal regular-Cauchy limit rows for the RealUp norm-distance data. These rows provide the carried limit, the convergence modulus, the late-observation distance histories, and the positive $\mathsf{Rat}^{\uparrow}$ comparison bounds demanded by the CompleteMetricUp target.

Apply Definition 128.5 to this modulus, limit, distance ledger, and rational-bound ledger. Since the distance ledger has already been identified with the Banach norm-induced metric row, the resulting witness is $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$ for the original Banach stream. □

Theorem 90.30 (Banach NameCert obligation package)**.** *The data of Definition 90.1, together with Definition 90.19, Definition 90.25, and Definition 90.26, Definition 90.27, Definition 90.28, and Definition 90.29, assemble the paper-side* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *obligation package. Its fields are the* $\mathsf{Norm}^{\uparrow}$ *carrier, the* hsame *classifier on representatives, the norm-induced metric ledger, the concrete metric-distance rows, the regular Cauchy input shape, the capstone-to-limit consumption row, the complete-metric limit-witness family, the transport stability clauses, the exactness clauses for displayed regular Cauchy limits, and the visible ledger policy inherited from* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$.

Proof. The carrier field is the predicate $X : \mathsf{Hist} \to \mathsf{Prop}$ equipped with the displayed $\mathsf{Norm}^{\uparrow}$ data and the induced $\mathsf{BanachMetricWitness}_X$ relation of Definition 90.1. The classifier field is hsame transport on representatives, so the certificate compares histories by visible representative transport rather than by a host quotient.

The stability field is supplied by Definition 90.25. It moves the norm rows, metric rows, stream rows, modulus histories, limit histories, and late-observation ledgers componentwise and then repacks them as the same Banach carrier structure.

The completeness and exactness fields are supplied by Definition 90.19 and Definition 90.26. These theorems show that every regular Cauchy stream over the norm-induced metric has a displayed $\mathsf{CompleteMetricLimitWitness}$ row, and that every such row has exactly the carried observations, carried limit, norm-distance rows, and rational comparison bounds demanded by $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$.

The concrete witness-row boundary is supplied by Definition 90.27 and Definition 90.28. The first theorem identifies the norm-induced Banach metric row with the CompleteMetricUp distance slot, and the second theorem repacks regular Cauchy streams with Banach distance rows into the input shape expected by the completeness family.

The limit-consumption boundary is supplied by Definition 90.29. It records that the real-completeness capstone rows are consumed through Definition 128.5 to produce the displayed $\mathsf{CompleteMetricLimitWitness}$ row for the Banach stream.

The ledger field is the union of the visible rows already named by $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ with the stream, modulus, threshold, distance, and comparison rows of $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$. The real-completeness capstone supplies the limit construction consumed by the complete-metric side, so all fields of the $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ obligation package are accounted for constructively. □

Theorem 90.31 (Banach norm-completeness downstream coverage)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, every accepted local consumer of the norm-induced metric completeness surface is exhausted by the displayed finite alternatives in this chapter. A consumer read must factor through one of the following rows: the norm-induced distance row* $d_\nu(x, y) = \nu(x -_V y)$ *of Definition 90.20, the classifier transport row of Definition 90.21, the regular Cauchy completion row of Definition 90.22, the zero-sequence anchor of Definition 90.23, or the assembled NameCert ledger of Definition 90.24. Thus all consumer-visible data are carried endpoints, norm-distance histories, regular Cauchy moduli, convergence moduli, limit endpoints, uniqueness up to* $\sim_V$*, and the ledger rows recording the norm and rational-bound reads. A quotient vector equality, host limit, detached RealUp completion selector, ambient compactness coordinate, countable choice selector, or unlisted metric coordinate is not exported by the* $\mathsf{Banach}^{\uparrow}$ *packet.*

Proof. Start with an accepted local consumer of the norm-induced completeness surface. Since the packet is the one assembled in Definition 90.24, the consumer can inspect only the fields named there: carrier C_V, classifier $\sim_V$, induced metric witness d_ν, stability rows for ν and d_ν, the ledger of orm provenance and regular Cauchy moduli, and the supplied completion operation with uniqueness up to $\sim_V$.

If the consumer reads a distance, the only available distance coordinate is $d_\nu(x, y) = \nu(x -_V y)$. The metric obligations of Definition 90.20 show that this is a legitimate $\mathsf{Met}^{\uparrow}$ row, and Definition 90.21 shows that changing representatives can only transport the same norm value and the same induced distance through the vector classifier. Hence a distance-facing consumer cannot introduce a second metric, a quotient distance, or a host-level endpoint comparison.

If the consumer reads a convergence claim, its input is a regular Cauchy sequence for d_ν. The completion row of Definition 90.22 is then the sole route: it returns the carried endpoint $\lim_\nu(X, \mu)$, an explicit convergence modulus, the ledger entries recording the modulus calls and the norm rows

used to read the distances, and uniqueness of any competing endpoint up to $\sim_V$. The zero-sequence case of Definition 90.23 is the corresponding anchored alternative when the stream is constant at 0_V; it supplies no additional route beyond the same completion and uniqueness rows.

These cases exhaust the assembly theorem. The stability branch is exactly classifier transport, the metric branch is exactly the norm-induced distance, the completion branch is exactly the regular Cauchy limit row, and the anchored branch is the constant-zero instance of that completion row. All remaining entries of the packet are ledger entries recording which norm, modulus, tolerance, and rational-bound rows were consumed. Since the carrier definition and assembly theorem list no coordinate for quotient vector equality, host limits, detached RealUp completion selectors, compactness, choice, or any extra metric, any route requiring one of those data is outside the BEDC packet. □

Theorem 90.32 (Banach complete-metric norm sibling route)**.** *Every* $\mathsf{Banach}^{\uparrow}$ *certificate route consumes the norm carrier of chapter 89, the completion interface of chapter 95, and the complete metric surface of chapter 128 before exporting bounded-linear-operator or limit rows. The Banach packet therefore reaches bounded-linear-operator and limit consumers through a finite ledger factorization, not through an unlisted analytic completeness principle.*

Proof. Read the Banach hub through its child files. The carrier and root NameCert rows expose a normed vector-space source, while the norm-completion certificate identifies the metric distance and regular Cauchy readback rows that the source must carry.

The completion interface is then consumed through the complete-metric NameCert interface. That child file binds the completion read to the complete metric surface before any limit certificate can be projected.

Finally the bounded-linear-operator and limit child files use those same rows as inputs. Their exports are therefore ledger factors over the displayed norm, completion, and complete-metric siblings, not separate claims of analytic completeness. □

Theorem 90.33 (Banach metric-norm completion sibling lattice)**.** *Every public* $\mathsf{Banach}^{\uparrow}$ *read factors through the displayed* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before a bounded-operator, fixed-point, or functional-analysis consumer inspects it. The available route is the norm-induced metric over the Banach carrier, the regular Cauchy stream and modulus surface, the complete-metric limit witness, and the RealUp distance readback recorded by chapter 89, chapter 57, chapter 128, chapter 301, chapter 126, and chapter 38.*

Proof. Read the Banach carrier through the norm-completion certificate. The files `norm_completion_certificate` and `complete_metric_namecert_interface` identify the metric coordinate as the $\mathsf{Norm}^{\uparrow}$-induced RealUp distance and identify accepted regular Cauchy inputs as $\mathsf{StreamName}^{\uparrow}$ observations equipped with displayed modulus and late-distance ledgers.

Next read the limit surface certificate. It exposes the same Banach carrier, stream observations, modulus history, carried limit row, norm-distance ledger, and consumed $\mathsf{CompleteMetric}^{\uparrow}$ witness before any downstream consumer receives a limit or bounded-operator row.

Thus a public Banach read has no independent metric island. Its metric, norm, completion, RegSeqRat, StreamName, and RealUp coordinates are the visible sibling lattice already displayed by the three child files, and all bounded-operator or fixed-point consumers can inspect only that finite route. □

The $\mathsf{Banach}^{\uparrow}$ lattice reads its complete normed source through the $\mathsf{NormedSpace}^{\uparrow}$ rows of chapter 1950 together with the completion-facing metric rows of chapter 128 and chapter 57. Thus a

Banach consumer receives bounded-operator or limit data only after the normed vector carrier, norm-induced distance, regular Cauchy stream, complete-metric limit witness, transport rows, replay rows, provenance, and local naming row have been displayed.

Lemma 90.34 (Banach normed complete-metric lattice anchor)**.** *Every* $\mathsf{Banach}^{\uparrow}$ *bounded-operator, limit, or functional-analysis export first factors through a* $\mathsf{NormedSpace}^{\uparrow}$ *source row and the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *completion rows before it exposes any Banach-specific conclusion. The route supplies the normed carrier, scalar and additive rows, norm-induced distance, regular Cauchy observation, complete-metric limit witness, transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. It supplies no detached host normed space, anonymous complete metric, selected limit outside the completion row, quotient carrier, or ambient completeness theorem.*

Proof. Read the Banach hub through its normed carrier and completion children. The $\mathsf{NormedSpace}^{\uparrow}$ source supplies the vector, scalar, additive, norm, and classifier rows used by the Banach carrier, while the $\mathsf{Met}^{\uparrow}$ row turns the displayed norm difference into the distance surface consumed by completion.

Next use the $\mathsf{CompleteMetric}^{\uparrow}$ completion interface. Its regular Cauchy stream, modulus, limit, and uniqueness rows are the only route by which the Banach packet can export completed-limit data to later bounded-operator or functional-analysis consumers.

The two source families are therefore jointly load-bearing. A downstream Banach read that omits either the normed-space rows or the complete-metric rows has no displayed coordinate from which to obtain a Banach limit, bounded-operator input, or completion-facing handoff. □

Definition 90.35 (Banach complete-normed template obligations)**.** *The* $\mathsf{Banach}^{\uparrow}$ *complete-normed template is the following ten-field paper target over the concrete carrier of Definition 90.2.*

1. *The traditional target is a complete normed vector space.*
2. *The BEDC source is* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ *together with* $\mathsf{CompleteMetricLimitWitness}$ *for the norm-induced metric.*
3. *The carrier is the complete normed* BHist *predicate* C_V*.*
4. *The point classifier is the displayed Banach classifier* $\sim_V$*, routed through* hsame *transport of representatives and ledgers.*
5. *The norm row is* $\nu : C_V \to \mathsf{Real}^{\uparrow}$ *with the RealUp classifier on norm values.*
6. *The metric row is* $d_\nu(x, y) := \nu(x -_V y)$*, displayed as the Banach metric witness consumed by* $\mathsf{CompleteMetric}^{\uparrow}$*.*
7. *Stability consists of norm transport, metric-distance transport, and complete-metric limit transport across the classifier.*
8. *Exactness records zero norm-distance exactly on* $\sim_V$*-classified endpoints and uniqueness of regular Cauchy limits up to* $\sim_V$*.*
9. *The ledger records the norm rows, stream observations, Cauchy moduli, limit rows, metric-distance rows, and* $\mathsf{Rat}^{\uparrow}$ *rational-comparison rows.*
10. *The status is a paper proof target for the* $\mathsf{Banach}^{\uparrow}$ *certificate surface.*

Theorem 90.36 (Banach norm-completion source scope)**.** *For any* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.2, the complete normed carrier supplies exactly the scoped* $\mathsf{Banach}^{\uparrow}$ *source obligations obtained from* $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{MetricDistanceWitness}$*,* $\mathsf{CompleteMetricLimitWitness}$*,*

and hsame. *These obligations are the carrier C_V, classifier $\sim_V$, norm ν, norm-induced metric d_ν, regular Cauchy stream input, completion witness, transport rows, and visible ledger fields named in Definition 90.35.*

Proof. First read the metric row from the norm. Apply Definition 90.20 to obtain the RealUp-valued metric, zero exactness, symmetry, and triangle row for d_ν.

Next read the stream and completion rows. Apply Definition 90.22 to every regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream for d_ν; the output is the carried limit, the convergence modulus, and uniqueness up to $\sim_V$.

Transport is supplied by Definition 90.25. It moves the carried representatives, norm rows, metric rows, stream observations, modulus histories, limit histories, and rational-comparison ledgers componentwise across hsame.

Finally apply Definition 90.30. Its fields are exactly the carrier, classifier, norm-induced metric, regular Cauchy input, complete-metric witness, transport, exactness, and ledger rows listed above, so the source scope contains no additional Banach obligation. □

Theorem 90.37 (Banach Hilbert downstream substrate)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *candidate whose inner-product norm induces the displayed Banach norm ν on the carrier C_V. Then the Banach data consumed by the Hilbert chapter are only the carrier C_V, the classifier $\sim_V$, the norm-induced metric row d_ν, and the* CompleteMetricLimitWitness *family fixed by Definition 90.35. This substrate does not assert Hilbert certificate closure or any operator theorem beyond those rows.*

Proof. Unpack the Hilbert input as an inner-product structure whose displayed norm is the same ν used by the Banach candidate. The Banach carrier and classifier are therefore copied directly as C_V and $\sim_V$.

The metric field required downstream is the norm-induced row. By Definition 90.20, $d_\nu(x, y) = \nu(x -_V y)$ is the metric row over the carried endpoints, and by Definition 90.21 it respects the classifier used by the Hilbert representative surface.

The completeness field is the same family consumed in Definition 90.22. Repack the carried limit endpoint, convergence modulus, uniqueness row, and visible ledger as the complete normed substrate named by the Hilbert chapter.

No inner-product law is inferred from these Banach rows. The Hilbert chapter must still supply the inner-product carrier, sesquilinear rows, positivity, and norm-induction proof at its own surface. □

Theorem 90.38 (Banach bounded-operator source closure)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates C, D, the bounded-linear-operator carrier* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *of Definition 90.49 is well-scoped over the source and target Banach packages. Its endpoint action, bound row, Lipschitz row, Cauchy-stream transport row, and ledger fields use only the carrier, classifier, norm-induced metric, and completion data of C and D.*

Proof. Begin with the carrier of Definition 90.49. Its hypotheses already name the source and target Banach carriers, their classifiers, their norms, their induced distances, the endpoint map T, the RealUp bound K, and the ledger Λ.

Apply Definition 90.50. The proof reads both distances through Definition 90.20, transports the target difference across linearity of T, and uses the bound row in Λ to obtain the target distance estimate.

Apply Definition 90.51 to a regular Cauchy source stream. The transported observations are carried by D, and the source modulus is converted into a target modulus using the RealUp bound comparison for K.

Finally use Definition 90.72 for componentwise hsame transport of stream observations, modulus histories, candidate limits, distance rows, and rational-comparison ledgers. The resulting operator package preserves the source closure data while landing in the target Banach package, so the bounded-operator carrier is scoped over C and D alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}$.

Constructive story: The concrete Banach surface begins with the singleton empty-history norm instance: the only vector endpoint is the empty history, every carried vector is classified with that endpoint, the norm value is the constant $\mathsf{Real}^\uparrow$ zero row, and the induced metric distance between any two carried endpoints is the same zero row. The classifier is the singleton vector classifier routed through hsame and the constant real classifier, so accepted presentations are the same exactly when their vector, norm, metric, stream, modulus, and limit ledgers transport through the displayed rows rather than through a host normed-space equality. The source specification is the consumed $\mathsf{Norm}^\uparrow$ package together with $\mathsf{StreamName}^\uparrow$ regular Cauchy inputs and CompleteMetricLimitWitness rows for the norm-induced metric, while the pattern specification records zero norm-distance exactness, Cauchy completeness with selected limit e, uniqueness of limits up to the singleton classifier, and bounded-operator limit transport over the same source and target Banach rows. The ledger policy keeps the singleton vector-space source, RatUp zero witness, constant-real seal, norm rows, metric-distance rows, Cauchy moduli, late-observation rows, selected limit, and bounded-operator transport evidence visible. The five name-certificate obligations are forced by the singleton: e inhabits the carrier, reflexivity, symmetry, and transitivity come from hsame and the singleton classifier, and carrier closure is projection from any classified comparison because both endpoints already carry the same empty-history package. A history h lies in the signature gap of a $\mathsf{Banach}^\uparrow$ token p when the source-domain accepts it as this complete normed singleton package, the signature relation pairs it with an answer Banach row and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h carries the zero norm, zero metric, and constant-limit data encoded by p. The standard bridge maps the BEDC singleton Banach row to the one-point complete normed vector space and reads the unique standard point back as e, preserving soundness, completeness, and roundtrip classifier rows. The chapter registers this complete-normed singleton construction under the name $\mathsf{Banach}^\uparrow$; non-singleton Banach spaces, Hilbert closure, operator-algebra closure, and checked bridge data are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper-side scope binds the singleton SemanticNameCert of Definition 90.4, the complete-normed source obligations of this section, the bounded-operator limit surface of Definition 90.77, and the singleton standard bridge soundness, completeness, and roundtrip rows of Definition 90.80, Definition 90.81, and Definition 90.82.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.BanachUp.BanachSingleton\_standard\_bridge\_soundness`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim bridgeChecked data, ordinary non-singleton Banach spaces, Hilbert closure, or operator-algebra closure.

Upgrade path: No further formal-axis work is recorded; the chapter is at the top formal status.

Definition 90.39 (Banach complete-metric NameCert interface)**.** *For a concrete Banach carrier X of Definition 90.1, a stream name s, a modulus history M, and a displayed limit history ℓ, the Banach complete-metric* NameCert *interface is the predicate*

$$\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell)$$

read over the norm-induced Banach metric rows. Its visible ledger consists of the carrier row X, the carried observations of s, the modulus row M, the carried limit row ℓ, and the late-observation distance ledgers named as $\mathsf{BanachMetricWitness}_X(s(n), \ell, d_{n,\varepsilon})$ *together with the corresponding* $\mathsf{Rat}^\uparrow$ *comparison rows.*

Equivalently, the displayed row

$$\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, L)$$

is accepted at the Banach surface exactly when every metric-distance component of L is presented by the norm-induced Banach ledger $\mathsf{BanachMetricWitness}_X$ *rather than by an anonymous metric row. Thus the public interface exposes the same carrier X, stream s, modulus M, limit ℓ, and convergence ledgers consumed by* $\mathsf{NameCert}_{\mathsf{CompleteMetric}^\uparrow}$.

Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton observation package
Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness empty limit rat positive
Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness row result alignment

Theorem 90.40 (Banach complete-metric NameCert coupling)**.** *Let X be a concrete Banach carrier, let s be a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over the norm-induced metric* $\mathsf{BanachMetricWitness}_X$*, let M be its displayed modulus, and let ℓ be a displayed limit. Carrier transport along* hsame*, together with a displayed Banach-surface row*

$$\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, L),$$

yields the SemanticNameCert *package for the predicate*

$$\mathsf{CompleteMetricLimitWitness}(X, s, M)$$

on limit histories. The coupling depends on Definition 90.1, Definition 90.25, Definition 90.26, and the complete-metric certificate theorem BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness_name_certifi

Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness empty modulus observation metric row
Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness observation distance deterministic

Proof. Unpack the concrete Banach carrier. Its carrier component supplies the hsame transport operation on carried vector histories, and its norm component identifies each induced distance row with a displayed $\mathsf{BanachMetricWitness}_X$ row.

Next transport the complete-metric data componentwise. Use Definition 90.25 for the stream observations, modulus history, limit history, threshold rows, metric-distance rows, and $\mathsf{Rat}^{\uparrow}$ comparison rows. The result is still a Banach carrier package because the transported metric rows are repacked through the norm-induced ledger.

Now invoke the complete-metric NameCert constructor represented on the Lean side by BEDC.Derived.Complete Its carrier-transport input is the Banach carrier transport just extracted, and its witness input is the displayed row of Definition 90.39.

Finally apply Definition 90.26. It ensures that no metric row has left the Banach surface: every distance ledger used by the complete-metric package is the same norm-distance row displayed as $\mathsf{BanachMetricWitness}_X$. Repacking these rows gives the required SemanticNameCert for $\mathsf{CompleteMetricLimitWitness}(X, s, M)$. □

Theorem 90.41 (Banach singleton limit NameCert uniqueness)**.** *In the singleton empty-history Banach carrier of Definition 90.3, any two Banach-surface complete-metric rows for the same stream,*

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, M_0, \ell_0, L_0) \quad \textit{and} \quad \mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, M_1, \ell_1, L_1),$$

have $\mathsf{hsame}(\ell_0, \ell_1)$ *limits. Consequently the two displayed vector limits are classified by the singleton vector classifier $\rho_{\mathsf{e}}(\ell_0, \ell_1)$.*

Lean4 ✓ BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton classifier distance

Proof. First read both displayed rows through the interface of Definition 90.39. The carrier predicate is C_{e}, so each limit row is a singleton empty-history endpoint and each metric-distance row is the constant zero Banach metric row.

Apply the complete-metric singleton uniqueness theorem BEDC.Derived.CompleteMetricUp.CompleteMetricLimit to the two rows for the same stream. It returns $\mathsf{hsame}(\ell_0, \ell_1)$ because the complete-metric carrier is the singleton predicate $h \mapsto \mathsf{hsame}(h, \mathsf{e})$.

It remains to read that relation at the vector endpoint surface. The singleton Banach carrier is the singleton norm carrier, so Definition 47.4 converts the two carried limit endpoints and their hsame relation into the classifier row $\rho_{\mathsf{e}}(\ell_0, \ell_1)$. □

Corollary 90.42 (Banach complete-metric interface determinacy)**.** *In the singleton empty-history Banach carrier C_{e} of Definition 90.3, fix a stream name s and fix a displayed Banach complete-metric interface row*

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, M, \ell, L).$$

Every accepted consumer-facing use of this row factors through the same visible interface data: the carrier row C_{e}, the stream observations of s, the modulus row M, the carried limit row ℓ, the late-distance rows in L displayed as

$$\mathsf{BanachMetricWitness}_{C_{\mathsf{e}}}(s(n), \ell, d_{n,\varepsilon}),$$

and the corresponding $\mathsf{Rat}^{\uparrow}$ comparison rows. If a second accepted use for the same stream displays

$$\mathsf{CompleteMetricLimitWitness}(C_{\mathsf{e}}, s, M', \ell', L'),$$

then its limit normal form is the same singleton endpoint, classified by $\rho_{\mathsf{e}}(\ell, \ell')$. Thus the interface admits no alternate hidden metric coordinate, no anonymous distance classifier, and no duplicate endpoint classifier source beyond the rows named in Definition 90.39 and Definition 90.41.

Proof. Unfold the interface of Definition 90.39. An accepted Banach-surface row is not merely a complete-metric row with an unspecified metric component. Its visible ledger consists of the carrier C_{e}, the stream observations of s, the modulus history, the carried limit history, the late-observation distance rows, and the $\mathsf{Rat}^{\uparrow}$ comparison rows. The same definition requires every metric-distance component of the late ledger to be presented as a norm-induced Banach row $\mathsf{BanachMetricWitness}_{C_{\mathsf{e}}}$ rather than as an anonymous metric row. Hence any consumer use of the fixed row must read exactly these displayed coordinates.

For the singleton carrier, Definition 90.3 identifies the vector carrier with the empty-history singleton, the endpoint classifier with ρ_{e}, and the norm-induced metric distance with the constant RealUp zero row. Therefore each accepted distance entry in L has the fixed Banach source $\mathsf{BanachMetricWitness}_{C_{\mathsf{e}}}(s(n), \ell, d_{n,\varepsilon})$, and the only exactness decision available at the complete-metric slot is the one that keeps this norm-induced row at the Banach surface.

Now compare any second accepted use for the same stream. Both displayed rows are Banach-surface complete-metric rows for C_{e}, so Definition 90.41 applies to them. It first gives $\mathsf{hsame}(\ell, \ell')$ and then reads that comparison as the singleton vector classifier $\rho_{\mathsf{e}}(\ell, \ell')$. Thus an alleged alternate limit normal form is already classified by the same singleton endpoint row. Since the interface ledger has no additional carrier coordinate, no extra metric slot, and no classifier other than the displayed singleton classifier, every accepted consumer-facing use factors through the listed finite Banach route. □

90.3 Banach limit surface certificate

Definition 90.43 (Banach limit surface certificate)**.** *For a concrete Banach carrier X of Definition 90.1, a regular $\mathsf{StreamName}^{\uparrow}$ stream s, a displayed modulus history M, a displayed limit*

history ℓ, and a late-distance ledger Λ, the Banach limit surface certificate is the predicate

$$\mathsf{BanachLimitSurface}_X(s, M, \ell, \Lambda).$$

It consists of the carried BHist *stream observations of s, the componentwise* hsame *transport rows for the observations, the modulus history M, and the limit history ℓ, the displayed* $\mathsf{BanachMetricWitness}_X$ *rows in Λ, and the* $\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, \Lambda)$ *row that consumes those distance rows.*

The ten-field template is fixed as follows. The traditional target is the Banach limit interface for regular Cauchy streams. The BEDC source is $\mathsf{Norm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *and the* $\mathsf{Real}^{\uparrow}$ *comparison rows used by the displayed norm distances. The carrier is the concrete Banach carrier X. The classifier is* hsame *on carried representatives together with the* RealClassifier *on norm distances. Stability is supplied by Definition 90.21. The ledger records the stream observations, the modulus M, the endpoint ℓ, the norm-distance rows in Λ, and the* $\mathsf{Rat}^{\uparrow}$ *comparison rows.*

The core theorem list is Definition 90.44, Definition 90.45, Definition 90.46, and Definition 90.47. Exactness is the equivalence between the displayed $\mathsf{BanachMetricWitness}_X$ *rows and the* CompleteMetricLimitWitness *rows they feed. The standard bridge field is absent. The status is a paper proof target in* BEDC.Derived.BanachUp.

Theorem 90.44 (Banach limit surface endpoint coverage)**.** *Any displayed row*

$$\mathsf{BanachLimitSurface}_X(s, M, \ell, \Lambda)$$

exposes the carried Banach endpoint X, the carried $\mathsf{StreamName}^{\uparrow}$ *observations of s, the modulus history M, the carried limit history ℓ, every displayed* $\mathsf{BanachMetricWitness}_X$ *distance row in Λ, and the corresponding* $\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, \Lambda)$ *row.*

Proof. Unfold Definition 90.43. The first displayed fields are exactly the carrier X, the stream observations of s, the modulus history M, and the carried limit history ℓ.

The next displayed fields are the ledger rows. Projecting them gives each $\mathsf{BanachMetricWitness}_X$ distance row in Λ together with the $\mathsf{Rat}^{\uparrow}$ comparison rows attached to the same late-observation entry.

The final displayed field is the consumed complete-metric row. Projecting it gives $\mathsf{CompleteMetricLimitWitness}(X$ with the same carrier, stream, modulus, endpoint, and distance ledger. □

Theorem 90.45 (Banach limit surface classifier transport)**.** *Let* $\mathsf{BanachLimitSurface}_X(s, M, \ell, \Lambda)$ *be displayed. Componentwise* hsame *transport of the stream observations, modulus history, limit endpoint, distance histories, and comparison ledgers preserves the predicate* $\mathsf{BanachLimitSurface}_X$ *for the transported data.*

Proof. Start with the stream and endpoint rows. Apply Definition 90.72 to the transported observations, modulus history, limit history, norm-distance rows, and $\mathsf{Rat}^{\uparrow}$ comparison rows. This keeps the regular Cauchy input and the convergence ledger in the Banach complete-metric shape.

Next transport the norm and distance classifiers. For each carried endpoint comparison in the stream and the candidate limit, use Definition 90.21 to obtain the transported RealClassifier rows for the corresponding norm values and induced distances.

Finally repackage the transported complete-metric row. Apply Definition 90.40 to the transported carrier rows, the transported $\mathsf{BanachMetricWitness}_X$ ledger, and the transported CompleteMetricLimitWitness row. The resulting package is precisely the transported $\mathsf{BanachLimitSurface}_X$ row. □

Theorem 90.46 (Banach limit surface uniqueness)**.** *Let X be a concrete Banach carrier and let s be a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over X. If*

$$\mathsf{BanachLimitSurface}_X(s, M_0, \ell_0, \Lambda_0)$$

and

$$\mathsf{BanachLimitSurface}_X(s, M_1, \ell_1, \Lambda_1)$$

are displayed for the same stream, then the carried limit histories are classified by hsame*. In the singleton empty-history carrier this classification is the singleton classifier row $\rho_{\mathsf{e}}(\ell_0, \ell_1)$.*

Proof. First project both surface rows through Definition 90.44. This exposes the two complete-metric rows $\mathsf{CompleteMetricLimitWitness}(X, s, M_0, \ell_0, \Lambda_0)$ and $\mathsf{CompleteMetricLimitWitness}(X, s, M_1, \ell_1, \Lambda_1)$ for the same regular Cauchy stream.

Apply Definition 90.71 to these two rows. Its conclusion identifies the carried limit histories in the Banach point classifier, which is the hsame-classified representative relation fixed by Definition 90.1.

For the singleton carrier, read the same two projected rows through Definition 90.39. Then apply Definition 90.41. It refines the hsame classification of ℓ_0 and ℓ_1 to the displayed singleton row $\rho_{\mathsf{e}}(\ell_0, \ell_1)$. □

Theorem 90.47 (Banach limit surface NameCert obligations)**.** *The predicate* $\mathsf{BanachLimitSurface}_X(s, M, \ell, \Lambda)$ *together with Definition 90.44, Definition 90.45, and Definition 90.46 assembles the paper-side* $\mathsf{NameCert}$ *obligation package for Banach limit surfaces.*

Proof. Endpoint coverage supplies the carrier, source, and ledger fields. By Definition 90.44, each certificate row exposes the Banach carrier X, the $\mathsf{StreamName}^{\uparrow}$ observations, the modulus history, the carried endpoint, the Banach metric rows, the $\mathsf{Rat}^{\uparrow}$ comparison rows, and the complete-metric limit row.

Stability is supplied by classifier transport. Apply Definition 90.45 to componentwise hsame transport of the observations, modulus, endpoint, and ledger histories; the theorem returns the transported Banach limit surface row with the RealUp distance classifiers repacked.

Exactness is supplied by uniqueness and the displayed metric boundary. Apply Definition 90.46 for the limit endpoint classifier, and read the metric rows through Definition 90.43 to identify each $\mathsf{BanachMetricWitness}_X$ row with the complete-metric row it feeds.

The CompleteMetricUp bridge is the interface of Definition 90.39. It connects the Banach norm-distance ledger to the consumed $\mathsf{CompleteMetricLimitWitness}$ row, so the carrier, classifier, stability, ledger, exactness, and bridge-facing fields of the Banach limit surface $\mathsf{NameCert}$ package are all accounted for. □

Corollary 90.48 (Banach limit surface certificate determinacy)**.** *Let X be a concrete Banach carrier and let s be a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over X. Fix a displayed Banach limit surface row*

$$\mathsf{BanachLimitSurface}_X(s, M, \ell, \Lambda).$$

Every accepted consumer-facing use of this certificate factors through the same finite surface data: the carrier row X, the carried observations of s, the modulus history M, the carried limit history ℓ, the displayed $\mathsf{BanachMetricWitness}_X$ *distance rows in Λ, the attached* $\mathsf{Rat}^{\uparrow}$ *comparison rows, and the consumed complete-metric row*

$$\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, \Lambda).$$

If another accepted surface row for the same stream is displayed as

$$\mathsf{BanachLimitSurface}_X(s, M', \ell', \Lambda'),$$

then its endpoint normal form is classified with the fixed one by hsame*; in the singleton empty-history carrier the comparison is the singleton row* $\rho_{\mathsf{e}}(\ell, \ell')$*. Hence the certificate has no alternate hidden coordinate, no anonymous metric classifier source, and no duplicate endpoint classifier beyond the rows named by Definition 90.43 and Definition 90.46.*

Proof. Unfold Definition 90.43. The certificate is not an unstructured assertion of convergence. Its visible fields are exactly the Banach carrier, the stream observations, the modulus history, the limit history, the late-distance ledger, and the complete-metric row that consumes that ledger. Thus a consumer of the fixed row can project only those displayed coordinates.

The distance side is also fixed by the same definition. Each late-distance entry is presented as a $\mathsf{BanachMetricWitness}_X$ row together with its $\mathsf{Rat}^{\uparrow}$ comparison rows, and the complete-metric slot is the row $\mathsf{CompleteMetricLimitWitness}(X, s, M, \ell, \Lambda)$ fed by those same entries. Therefore the exactness decision available at the surface is precisely the decision that reads the displayed Banach metric rows as the displayed complete-metric witness rows they feed; an anonymous metric coordinate would not be a use of this certificate.

Now compare any second accepted surface row for the same stream. Applying Definition 90.46 to the two displayed rows gives that ℓ and ℓ' are classified by hsame. In the singleton empty-history carrier, the same theorem reads this comparison as $\rho_{\mathsf{e}}(\ell, \ell')$. Thus a competing endpoint normal form is already identified by the chapter's endpoint classifier, while the route and exactness readback remain the finite rows exposed by the certificate definition. □

Definition 90.49 (Banach bounded linear operator carrier)**.** *Let C and D be* $\mathsf{Banach}^{\uparrow}$ *candidates with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, norms ν_C, ν_D, and induced distances d_C, d_D. A Banach bounded linear operator carrier from C to D is a BEDC history predicate*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

on endpoint maps $T : C_V \to D_V$, RealUp bound endpoints K, and visible ledger histories Λ such that the following rows are carried.

1. *The map respects vector classifiers: if $x \sim_C x'$ and both endpoints are carried by C_V, then $T(x) \sim_D T(x')$.*
2. *The map respects the vector operations: $T(x +_C y) \sim_D T(x) +_D T(y)$ and $T(a \cdot_C x) \sim_D a \cdot_D T(x)$ for every carried RealUp scalar endpoint a and carried source endpoints x, y.*
3. *The RealUp bound row records $K \geq 0$ and*

$$\nu_D(T(x)) \leq K \cdot \nu_C(x)$$

   for every carried source endpoint $x : C_V$.
4. *The ledger Λ records the source endpoint row, target endpoint row, source and target norm rows, scalar row, additive and homogeneity compatibility rows, and the bound witness used for each carried input.*

The classifier on operator representatives is hsame *transport of the endpoint map together with* $\mathsf{Real}^{\uparrow}$ *transport of the bound and componentwise transport of the ledger rows.*

Theorem 90.50 (Bounded linear operator Lipschitz obligation)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D*, any carried row* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *satisfies the metric Lipschitz obligation*

$$d_D(Tx, Ty) \leq K \cdot d_C(x, y)$$

for all carried source endpoints $x, y : C_V$*. The distance rows are the norm-induced rows of Definition 90.20.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator lipschitz`

Proof. First read the two distances through Definition 90.20. The source distance is the RealUp norm endpoint $\nu_C(x -_C y)$, and the target distance is $\nu_D(Tx -_D Ty)$.

Use the additive compatibility row in Definition 90.49. It transports the target difference $Tx -_D Ty$ to the endpoint $T(x -_C y)$ under the target vector classifier, and the $\mathsf{Norm}^{\uparrow}$ transport row moves ν_D across that classifier comparison.

Apply the bound row of the operator carrier to the carried source difference $x -_C y$. This gives

$$\nu_D(T(x -_C y)) \leq K \cdot \nu_C(x -_C y).$$

Substituting the norm-induced distance rows gives the displayed Lipschitz obligation. □

Theorem 90.51 (Bounded linear operator preserves regular Cauchy streams)**.** *Let* C, D *be* $\mathsf{Banach}^{\uparrow}$ *candidates and let* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *be carried. If* $(X, \mu) : \mathbb{N} \to C_V$ *is a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream for the source distance* d_C*, then the pointwise stream* $T \circ X$ *is a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream for the target distance* d_D*. Its carried observation rows are obtained by applying the target endpoint field of* Λ*, and its modulus is obtained from* μ *by feeding each target tolerance through the RealUp bound comparison for* K*.*

Proof. The observation rows are immediate from the operator ledger: for each carried source endpoint X_n, the target endpoint row records that $T(X_n)$ is carried by D_V. Classifier-respect transports any source stream identifications to target stream identifications, so the pointwise stream is accepted by the target vector classifier.

It remains to show the Cauchy estimate. For a requested positive target tolerance, use the RealUp order and multiplication rows to choose a positive source tolerance whose product with K is bounded by the requested target tolerance. Feed that source tolerance into the source modulus μ.

For indices past the returned bound, the source Cauchy row gives a bound on $d_C(X_m, X_n)$. Apply Definition 90.50 to X_m and X_n, then compose the RealUp inequalities. The resulting target distance row is

$$d_D(TX_m, TX_n) \leq K \cdot d_C(X_m, X_n),$$

and the chosen tolerance comparison turns it into the regular Cauchy row for the transported stream. □

Theorem 90.52 (Banach bounded linear operator NameCert obligations)**.** *For any pair of* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D*, the carrier of Definition 90.49, together with Definition 90.50 and Definition 90.51, assembles the paper-side naming-certificate obligation set for the Banach bounded-linear operator surface. The carrier is* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$*, the classifier is* hsame *transport of maps, bounds, and ledgers, stability is classifier transport through source and target Banach carriers, exactness is the equivalence between displayed operator rows and the additive, homogeneous, bounded endpoint action they record, and the ledger fields are precisely the source endpoint, target endpoint, norm, scalar, compatibility, and bound witness rows.*

Proof. The carrier and classifier fields are those of Definition 90.49. They are stated over the already certified Banach carriers C and D, so source endpoints, target endpoints, norm values, and distance rows are all read through the existing $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ obligations.

The stability fields are obtained by componentwise classifier transport. A source classifier comparison is transported through T by the first carrier row; additive and scalar compatibility are transported by the vector classifier rows of C and D; and the RealUp bound is transported by the bound endpoint classifier.

The metric-use field is supplied by Definition 90.50. It converts the operator bound into the target distance estimate consumed by downstream metric and Hilbert arguments.

The stream field is supplied by Definition 90.51. It records that regular Cauchy stream names remain regular Cauchy after pointwise application of the operator, with the source modulus transformed by the RealUp bound row.

Finally, exactness follows by unfolding the displayed carrier. Every accepted operator row contains precisely the endpoint map, classifier-respect row, additive row, scalar row, RealUp bound row, and visible provenance ledger; and any such collection repacks as $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ without adding quotient or host-level structure. □

Theorem 90.53 (Banach zero bounded linear operator carrier)**.** *Let C, D be $\mathsf{Banach}^{\uparrow}$ candidates as in Definition 90.2, with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, zero endpoints $0_C, 0_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. Assume the candidate ledgers expose the source and target vector-zero rows, including the target additive-zero and scalar-zero rows, and a RealUp zero bound endpoint $0_{\mathsf{Real}^{\uparrow}}$ whose rows include*

$$0_{\mathsf{Real}^{\uparrow}} \geq 0 \quad \textit{and} \quad 0_{\mathsf{Real}^{\uparrow}} \cdot r \textit{ classified with } 0_{\mathsf{Real}^{\uparrow}}$$

for every carried norm endpoint r. Define the zero endpoint map

$$Z_{C,D} : C_V \to D_V, \qquad Z_{C,D}(x) := 0_D.$$

Let Λ_0 be the visible ledger containing the source endpoint row, the target zero endpoint row, the source and target norm rows, the target classifier reflexivity row at 0_D, the target additive-zero and scalar-zero rows, and the RealUp zero nonnegativity, multiplication, order-reflexivity, and classifier-transport rows. Then

$$\mathsf{BanachBLOp}(C, D, Z_{C,D}, 0_{\mathsf{Real}^{\uparrow}}, \Lambda_0)$$

is carried.

Proof. Unfold Definition 90.49. For every carried source endpoint $x : C_V$, the endpoint value of the displayed map is $Z_{C,D}(x) = 0_D$. The target vector-zero row carries 0_D in D_V, so the target endpoint row required by the operator carrier is present.

For classifier-respect, suppose $x \sim_C x'$ and both source endpoints are carried. Both images are the same target zero endpoint. The target classifier reflexivity row gives $0_D \sim_D 0_D$, hence

$$Z_{C,D}(x) \sim_D Z_{C,D}(x').$$

For additivity, let $x, y : C_V$ be carried. The source vector rows carry $x +_C y$, so the left endpoint is again 0_D. The right endpoint is

$$Z_{C,D}(x) +_D Z_{C,D}(y) = 0_D +_D 0_D.$$

The target additive-zero law classifies $0_D +_D 0_D$ with 0_D; symmetry of the target vector classifier gives the required comparison

$$Z_{C,D}(x +_C y) \sim_D Z_{C,D}(x) +_D Z_{C,D}(y).$$

For homogeneity, let a be a carried RealUp scalar endpoint and let $x : C_V$ be carried. The source scalar-action row carries $a \cdot_C x$, so the left endpoint is 0_D. The right endpoint is $a \cdot_D 0_D$. The target scalar-zero row classifies $a \cdot_D 0_D$ with 0_D, and classifier symmetry gives

$$Z_{C,D}(a \cdot_C x) \sim_D a \cdot_D Z_{C,D}(x).$$

It remains to verify the bound row. The RealUp zero bound ledger gives $0_{\mathsf{Real}^\uparrow} \geq 0$. For a carried source endpoint x, the $\mathsf{Norm}^\uparrow$ positive-definiteness row contained in the target Banach candidate of Definition 90.2 classifies $\nu_D(0_D)$ with $0_{\mathsf{Real}^\uparrow}$; this is the zero case of the metric obligation recorded in Definition 90.20. Since $Z_{C,D}(x) = 0_D$, this is the left side of the bound. The RealUp multiplication row classifies $0_{\mathsf{Real}^\uparrow} \cdot \nu_C(x)$ with $0_{\mathsf{Real}^\uparrow}$. Order reflexivity at $0_{\mathsf{Real}^\uparrow}$, transported across these two RealUp classifier comparisons, gives

$$\nu_D(Z_{C,D}(x)) \leq 0_{\mathsf{Real}^\uparrow} \cdot \nu_C(x).$$

The rows just assembled are exactly the classifier, additive, homogeneous, bound, and ledger rows demanded by Definition 90.49. Repacking them with the displayed endpoint map and the bound endpoint $0_{\mathsf{Real}^\uparrow}$ yields the carried row

$$\mathsf{BanachBLOp}(C, D, Z_{C,D}, 0_{\mathsf{Real}^\uparrow}, \Lambda_0).$$

□

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonZeroBoundedLinearOperator carrier`

Definition 90.54 (Banach bounded linear operator pointwise sum carrier)**.** *Let C, D be $\mathsf{Banach}^\uparrow$ candidates with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, target addition $+_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. Given carried rows*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and

$$\mathsf{BanachBLOp}(C, D, S, L, \Gamma),$$

their pointwise sum endpoint map is

$$(T \boxplus_D S)(x) := T(x) +_D S(x).$$

The displayed sum-bound endpoint is the RealUp addition endpoint $K + L$. The sum ledger $\Lambda \boxplus \Gamma$ is the ordered record containing the two operator ledgers, the target $\mathsf{AbGroup}^\uparrow$ addition, associativity, commutativity, congruence, and zero-carry rows, the target vector-space scalar distributivity row, the target $\mathsf{Norm}^\uparrow$ triangle row, and the $\mathsf{RealAlgOrder}$ addition, multiplication, distributivity, nonnegativity, transitivity, and additive order-transport rows used to compare the constructed bound with $K + L$.

Theorem 90.55 (Banach bounded linear operator pointwise sum closure)**.** *Let C, D be* $\mathsf{Banach}^{\uparrow}$ *candidates as in Definition 90.2, with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, target addition $+_D$, scalar action $\cdot_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. If*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and

$$\mathsf{BanachBLOp}(C, D, S, L, \Gamma)$$

are carried, then the pointwise sum map of Definition 90.54 is carried by

$$\mathsf{BanachBLOp}(C, D, T \boxplus_D S, K + L, \Lambda \boxplus \Gamma).$$

For every carried source endpoint $x : C_V$, its bound row is the $\mathsf{RealAlgOrder}$ *additive-order composite*

$$\nu_D((T \boxplus_D S)(x)) \leq (K + L) \cdot \nu_C(x),$$

and the displayed operator-bound endpoint is classified by the RealUp order with the scalar sum of the two input bounds:

$$K + L \leq K + L.$$

Proof. Unfold the operator carrier of Definition 90.49. For a carried source endpoint $x : C_V$, the target endpoint rows of Λ and Γ carry $T(x)$ and $S(x)$ in D_V. The target additive carrier row from the $\mathsf{AbGroup}^{\uparrow}$ reduct of the target vector space carries

$$(T \boxplus_D S)(x) = T(x) +_D S(x),$$

so the endpoint row required for the sum map is present.

For classifier-respect, suppose $x \sim_C x'$ with all source endpoints carried. The classifier rows in Λ and Γ give

$$T(x) \sim_D T(x') \qquad \text{and} \qquad S(x) \sim_D S(x').$$

Target additive congruence transports these two comparisons to

$$T(x) +_D S(x) \sim_D T(x') +_D S(x'),$$

which is exactly

$$(T \boxplus_D S)(x) \sim_D (T \boxplus_D S)(x').$$

For additivity, let $x, y : C_V$ be carried. The additive rows of the two input operator ledgers give

$$T(x +_C y) \sim_D T(x) +_D T(y)$$

and

$$S(x +_C y) \sim_D S(x) +_D S(y).$$

Applying target additive congruence to these two comparisons gives

$$T(x +_C y) +_D S(x +_C y) \sim_D (T(x) +_D T(y)) +_D (S(x) +_D S(y)).$$

The target $\mathsf{AbGroup}^{\uparrow}$ associativity and commutativity rows reorder the four carried summands as

$$(T(x) +_D T(y)) +_D (S(x) +_D S(y)) \sim_D (T(x) +_D S(x)) +_D (T(y) +_D S(y)).$$

Transitivity of $\sim_D$ yields

$$(T \boxplus_D S)(x +_C y) \sim_D (T \boxplus_D S)(x) +_D (T \boxplus_D S)(y).$$

For homogeneity, let a be a carried RealUp scalar endpoint and let $x : C_V$ be carried. The homogeneous rows of Λ and Γ give

$$T(a \cdot_C x) \sim_D a \cdot_D T(x)$$

and

$$S(a \cdot_C x) \sim_D a \cdot_D S(x).$$

Target additive congruence combines them as

$$T(a \cdot_C x) +_D S(a \cdot_C x) \sim_D (a \cdot_D T(x)) +_D (a \cdot_D S(x)).$$

The scalar-distributivity row of the target vector-space structure classifies the right endpoint with

$$a \cdot_D (T(x) +_D S(x)).$$

Composing these comparisons gives

$$(T \boxplus_D S)(a \cdot_C x) \sim_D a \cdot_D (T \boxplus_D S)(x).$$

It remains to verify the bound row. The non-negativity rows in the two operator carriers give $K \geq 0$ and $L \geq 0$. The $\mathsf{RealAlgOrder}$ additive order row carries $K + L \geq 0$, so the proposed sum bound is non-negative. For a carried source endpoint $x : C_V$, the target $\mathsf{Norm}^{\uparrow}$ triangle row gives

$$\nu_D(T(x) +_D S(x)) \leq \nu_D(T(x)) + \nu_D(S(x)).$$

The two input bound rows give

$$\nu_D(T(x)) \leq K \cdot \nu_C(x)$$

and

$$\nu_D(S(x)) \leq L \cdot \nu_C(x).$$

The additive order-transport row of $\mathsf{RealAlgOrder}$ adds these two comparisons:

$$\nu_D(T(x)) + \nu_D(S(x)) \leq (K \cdot \nu_C(x)) + (L \cdot \nu_C(x)).$$

The RealUp distributivity row rewrites the right endpoint as

$$(K \cdot \nu_C(x)) + (L \cdot \nu_C(x)) \sim_{\mathsf{Real}^{\uparrow}} (K + L) \cdot \nu_C(x).$$

Order transport across this RealUp classifier comparison and transitivity with the triangle inequality give

$$\nu_D((T \boxplus_D S)(x)) \leq (K + L) \cdot \nu_C(x).$$

The bound-classification statement is the RealUp order reflexivity row at the same addition endpoint $K + L$, after the $\mathsf{RealAlgOrder}$ addition row has carried that endpoint from the displayed input bounds K and L:

$$K + L \leq K + L.$$

The endpoint, classifier, additive, homogeneous, non-negative, bound, and ledger rows just assembled are precisely the rows required by Definition 90.49. Repacking them with the pointwise sum endpoint map, the bound endpoint $K + L$, and the visible ledger $\Lambda \boxplus \Gamma$ yields the carried row

$$\mathsf{BanachBLOp}(C, D, T \boxplus_D S, K + L, \Lambda \boxplus \Gamma).$$

□

Definition 90.56 (Banach bounded linear operator scalar multiple carrier)**.** *Let C, D be* $\mathsf{Banach}^{\uparrow}$ *candidates with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, target scalar action $\cdot_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. Given a carried row*

$$\mathsf{BanachBLOp}(C, D, T, M, \Lambda)$$

and a carried RealUp scalar endpoint a, let $|a|_{\mathsf{Real}^{\uparrow}}$ be the carried absolute-value endpoint exposed by the shared RealAlgOrder *row. The scalar-multiple endpoint map is*

$$(a \boxtimes_D T)(x) := a \cdot_D T(x).$$

The displayed scalar-multiple bound endpoint is $|a|_{\mathsf{Real}^{\uparrow}} \cdot M$. The scalar-multiple ledger $\langle a \rangle \boxtimes \Lambda$ is the ordered record containing the input operator ledger, the carried scalar row for a, the RealAlgOrder absolute-value and non-negativity rows for $|a|_{\mathsf{Real}^{\uparrow}}$, the target vector-space scalar-action closure, congruence, distributivity, and scalar-associativity rows, the RealUp multiplication commutativity and associativity rows, the target $\mathsf{Norm}^{\uparrow}$ *absolute-homogeneity row, and the RealAlgOrder multiplication, monotonicity, order-transport, and product non-negativity rows used to compare the constructed norm bound with $|a|_{\mathsf{Real}^{\uparrow}} \cdot M$.*

Theorem 90.57 (Banach bounded linear operator scalar multiple closure)**.** *Let C, D be* $\mathsf{Banach}^{\uparrow}$ *candidates as in Definition 90.2, with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, target scalar action $\cdot_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. If*

$$\mathsf{BanachBLOp}(C, D, T, M, \Lambda)$$

is carried and a is a carried RealUp scalar endpoint with carried absolute-value endpoint $|a|_{\mathsf{Real}^{\uparrow}}$, then the scalar-multiple endpoint map of Definition 90.56 is carried by

$$\mathsf{BanachBLOp}(C, D, a \boxtimes_D T, |a|_{\mathsf{Real}^{\uparrow}} \cdot M, \langle a \rangle \boxtimes \Lambda).$$

For every carried source endpoint $x : C_V$, its bound row is

$$\nu_D((a \boxtimes_D T)(x)) \leq (|a|_{\mathsf{Real}^{\uparrow}} \cdot M) \cdot \nu_C(x).$$

Proof. Unfold the operator carrier of Definition 90.49. For a carried source endpoint $x : C_V$, the target endpoint row of Λ carries $T(x)$ in D_V. Since a is a carried RealUp scalar endpoint, the target vector-space scalar-action closure row carries

$$(a \boxtimes_D T)(x) = a \cdot_D T(x)$$

in D_V.

For classifier-respect, suppose $x \sim_C x'$ with both source endpoints carried. The classifier row of Λ gives

$$T(x) \sim_D T(x').$$

Target scalar-action congruence with the fixed carried scalar a transports this comparison to

$$a \cdot_D T(x) \sim_D a \cdot_D T(x'),$$

which is exactly

$$(a \boxtimes_D T)(x) \sim_D (a \boxtimes_D T)(x').$$

For additivity, let $x, y : C_V$ be carried. The additive row of Λ gives

$$T(x +_C y) \sim_D T(x) +_D T(y).$$

Scalar-action congruence with a gives

$$a \cdot_D T(x +_C y) \sim_D a \cdot_D (T(x) +_D T(y)).$$

The target vector-space distributivity row classifies the right endpoint with

$$a \cdot_D T(x) +_D a \cdot_D T(y).$$

Composing these target classifier comparisons yields

$$(a \boxtimes_D T)(x +_C y) \sim_D (a \boxtimes_D T)(x) +_D (a \boxtimes_D T)(y).$$

For homogeneity, let r be a carried RealUp scalar endpoint and let $x : C_V$ be carried. The homogeneous row of Λ gives

$$T(r \cdot_C x) \sim_D r \cdot_D T(x).$$

Scalar-action congruence with a transports it to

$$a \cdot_D T(r \cdot_C x) \sim_D a \cdot_D (r \cdot_D T(x)).$$

Target scalar associativity classifies the right endpoint with

$$(a \cdot r) \cdot_D T(x).$$

The RealUp multiplication commutativity row classifies $a \cdot r$ with $r \cdot a$, and target scalar-action congruence transports this scalar comparison to

$$(a \cdot r) \cdot_D T(x) \sim_D (r \cdot a) \cdot_D T(x).$$

Applying target scalar associativity in the reverse direction gives

$$(r \cdot a) \cdot_D T(x) \sim_D r \cdot_D (a \cdot_D T(x)).$$

Transitivity of $\sim_D$ gives

$$(a \boxtimes_D T)(r \cdot_C x) \sim_D r \cdot_D (a \boxtimes_D T)(x).$$

It remains to verify the bound row. The operator carrier supplies $M \geq 0$. The RealAlgOrder absolute-value non-negativity row supplies $|a|_{\mathsf{Real}^\uparrow} \geq 0$. Hence RealUp multiplication non-negativity carries

$$|a|_{\mathsf{Real}^\uparrow} \cdot M \geq 0,$$

so the proposed bound endpoint is non-negative.

For a carried source endpoint $x : C_V$, the target $\mathsf{Norm}^\uparrow$ absolute-homogeneity row gives the RealUp classifier comparison

$$\nu_D(a \cdot_D T(x)) \sim_{\mathsf{Real}^\uparrow} |a|_{\mathsf{Real}^\uparrow} \cdot \nu_D(T(x)).$$

The bound row of Λ gives

$$\nu_D(T(x)) \leq M \cdot \nu_C(x).$$

Since $|a|_{\mathsf{Real}^\uparrow} \geq 0$, RealUp multiplication monotonicity transports this comparison to

$$|a|_{\mathsf{Real}^\uparrow} \cdot \nu_D(T(x)) \leq |a|_{\mathsf{Real}^\uparrow} \cdot (M \cdot \nu_C(x)).$$

RealUp associativity classifies the right endpoint with

$$(|a|_{\mathsf{Real}^\uparrow} \cdot M) \cdot \nu_C(x).$$

Order transport across the absolute-homogeneity and associativity classifier comparisons, followed by order transitivity, gives

$$\nu_D((a \boxtimes_D T)(x)) \leq (|a|_{\mathsf{Real}^\uparrow} \cdot M) \cdot \nu_C(x).$$

The endpoint, classifier, additive, homogeneous, non-negative, bound, and ledger rows assembled above are precisely the rows demanded by Definition 90.49. Repacking them with the scalar-multiple endpoint map, the bound endpoint $|a|_{\mathsf{Real}^\uparrow} \cdot M$, and the visible ledger $\langle a \rangle \boxtimes \Lambda$ yields the carried row

$$\mathsf{BanachBLOp}(C, D, a \boxtimes_D T, |a|_{\mathsf{Real}^\uparrow} \cdot M, \langle a \rangle \boxtimes \Lambda).$$

□

Theorem 90.58 (Banach identity bounded linear operator carrier)**.** *Let C be a $\mathsf{Banach}^\uparrow$ candidate as in Definition 90.2, with vector carrier C_V, classifier $\sim_C$, addition $+_C$, scalar action $\cdot_C$, norm ν_C, and norm-induced distance d_C. Let*

$$\mathrm{id}_C : C_V \to C_V, \qquad \mathrm{id}_C(x) := x.$$

Let $1_{\mathsf{Real}^\uparrow}$ be the RealUp multiplicative-unit endpoint exposed by the scalar source. Let $\mathcal{I}_C$ be the visible identity ledger containing the source endpoint row, the same row read as the target endpoint row, the classifier equivalence rows of $\sim_C$, the vector operation rows for $+_C$ and $\cdot_C$, the $\mathsf{Norm}^\uparrow$ norm rows for ν_C, the RealUp non-negativity row

$$1_{\mathsf{Real}^\uparrow} \geq 0,$$

the RealUp multiplication-unit row

$$1_{\mathsf{Real}^\uparrow} \cdot r \sim_{\mathsf{Real}^\uparrow} r$$

for every carried norm endpoint r, and the RealUp order-reflexivity and order-transport rows. Then

$$\mathsf{BanachBLOp}(C, C, \mathrm{id}_C, 1_{\mathsf{Real}^\uparrow}, \mathcal{I}_C)$$

is carried.

Proof. Unfold Definition 90.49. For every carried source endpoint $x : C_V$, the endpoint value of the displayed map is $\mathrm{id}_C(x) = x$. The source carrier row for x is therefore also the target endpoint row, since source and target are both the same candidate C.

For classifier-respect, suppose $x \sim_C x'$ and both endpoints are carried. The identity endpoint map changes neither side, so the same classifier witness gives

$$\mathrm{id}_C(x) \sim_C \mathrm{id}_C(x').$$

For additivity, let $x, y : C_V$ be carried. The vector-space addition row of the $\mathsf{Banach}^\uparrow$ candidate carries $x +_C y$. Both sides of the required comparison compute to this carried endpoint:

$$\mathrm{id}_C(x +_C y) = x +_C y$$

and

$$\mathrm{id}_C(x) +_C \mathrm{id}_C(y) = x +_C y.$$

Classifier reflexivity at the carried sum gives

$$\mathrm{id}_C(x +_C y) \sim_C \mathrm{id}_C(x) +_C \mathrm{id}_C(y).$$

For homogeneity, let a be a carried RealUp scalar endpoint and let $x : C_V$ be carried. The scalar-action row carries $a \cdot_C x$. Again both sides compute to the same endpoint:

$$\mathrm{id}_C(a \cdot_C x) = a \cdot_C x$$

and

$$a \cdot_C \mathrm{id}_C(x) = a \cdot_C x.$$

Classifier reflexivity at the carried scalar-action endpoint gives

$$\mathrm{id}_C(a \cdot_C x) \sim_C a \cdot_C \mathrm{id}_C(x).$$

It remains to verify the bound row. The ledger gives $1_{\mathsf{Real}^\uparrow} \geq 0$. For a carried source endpoint x, the norm row gives the carried RealUp endpoint $\nu_C(x)$. Since $\mathrm{id}_C(x) = x$, the left side of the operator bound is $\nu_C(x)$. RealUp order reflexivity gives

$$\nu_C(x) \leq \nu_C(x).$$

The RealUp multiplication-unit row classifies $1_{\mathsf{Real}^\uparrow} \cdot \nu_C(x)$ with $\nu_C(x)$. Transporting the right side of the reflexive order comparison across this RealUp classifier comparison gives

$$\nu_C(\mathrm{id}_C(x)) \leq 1_{\mathsf{Real}^\uparrow} \cdot \nu_C(x).$$

The rows just assembled are exactly the target endpoint, classifier-respect, additive compatibility, scalar compatibility, RealUp bound, and visible ledger rows demanded by Definition 90.49. Repacking them with the displayed endpoint map and the bound endpoint $1_{\mathsf{Real}^\uparrow}$ yields the carried row

$$\mathsf{BanachBLOp}(C, C, \mathrm{id}_C, 1_{\mathsf{Real}^\uparrow}, \mathcal{I}_C).$$

□

Lemma 90.59 (Banach bounded linear operator Cauchy bridge determinacy)**.** *Let C, D be $\mathsf{Banach}^\uparrow$ candidates and let $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ be carried. For every regular Cauchy $\mathsf{StreamName}^\uparrow$ stream $(X, \mu) : \mathbb{N} \to C_V$ over the source distance d_C, the consumer-facing bridge from the source stream to the transported stream $T \circ X$ over d_D is determined by the following displayed rows: the source endpoint rows for X_n, the target endpoint rows for $T(X_n)$ in Λ, the classifier-respect row of Definition 90.49, the norm-induced distance rows of Definition 90.20, the RealUp non-negativity and bound-comparison rows for K, the source modulus row μ, the source Cauchy row, and the Lipschitz row of Definition 90.50. Consequently every accepted route, transport, readback, or handoff used by a downstream consumer of Definition 90.51 factors through those rows. No further coordinate is available for a hidden inverse operator, an ambient compactness principle, a selected limit, a quotient stream equality, a host continuity theorem, or a detached tolerance selector.*

Proof. Unfold the transported-stream theorem Definition 90.51. Its first public row is the observation row for the target stream. For each carried source endpoint X_n, the operator ledger Λ supplies the target endpoint row for $T(X_n)$. If a consumer changes a displayed source representative, the source stream identification passes to the target stream identification by the observation-stability clause of Definition 90.51; that theorem consumes exactly the classifier-respect row named in Definition 90.49.

The distance readback is also fixed. By Definition 90.20, the source and target distances are read as the norm rows

$$d_C(X_m, X_n) = \nu_C(X_m -_C X_n)$$

and

$$d_D(TX_m, TX_n) = \nu_D(TX_m -_D TX_n).$$

Apply Definition 90.50. In its adjacent Lipschitz transport step, the additive compatibility row of Definition 90.49 identifies the target difference with the displayed endpoint $T(X_m -_C X_n)$ under the target vector classifier, and the norm-transport row used by Definition 90.50 moves the target norm across that classifier comparison. The theorem then gives the unique metric handoff exposed by the packet:

$$d_D(TX_m, TX_n) \leq K \cdot d_C(X_m, X_n).$$

Now fix a requested positive target tolerance. The Cauchy-transport theorem uses only the RealUp order and multiplication rows for the displayed bound K to choose a source tolerance whose product with K is below the requested target tolerance. It then feeds that source tolerance into the displayed source modulus μ. For all indices beyond the returned bound, the source Cauchy row gives the source distance estimate; composing it with the Lipschitz row and the chosen RealUp comparison gives the target Cauchy estimate. Thus the consumer handoff is exactly the chain

$$\mu \quad \longrightarrow \quad d_C(X_m, X_n) \quad \longrightarrow \quad K \cdot d_C(X_m, X_n) \quad \longrightarrow \quad d_D(TX_m, TX_n).$$

Every row used in this chain has already been named in the carrier, the norm-induced metric theorem, the Lipschitz theorem, or the Cauchy-transport theorem. The accepted packet has no field naming an inverse operator, a compact subcover, a target limit choice, a quotient equality between streams, an external continuity theorem, or a tolerance selector detached from the RealUp bound comparison. Such a request therefore has no source coordinate in the Banach bounded-linear-operator bridge. Hence every accepted consumer-facing route factors through the listed local rows, and every other bridge coordinate remains outside the displayed packet. □

Definition 90.60 (Banach bounded linear operator composition carrier)**.** *Let C, D, E be* $\mathsf{Banach}^{\uparrow}$ *candidates with vector carriers C_V, D_V, E_V, classifiers $\sim_C, \sim_D, \sim_E$, norms ν_C, ν_D, ν_E, and norm-induced distances d_C, d_D, d_E. Given carried rows*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and

$$\mathsf{BanachBLOp}(D, E, S, L, \Gamma),$$

their composition carrier is

$$\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda).$$

Here $S \circ T$ is the endpoint map $x \mapsto S(Tx)$, the RealUp bound row is the product endpoint $L \cdot K$, and the visible ledger $\Gamma \star \Lambda$ is the ordered concatenation of the source-to-middle operator rows, the middle-to-target operator rows, and the RealUp multiplication and order rows used to compare the composite bound.

Theorem 90.61 (Bounded linear operator composition Lipschitz)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D, E*, carried rows* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *and* $\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$ *satisfy the composite estimate*

$$d_E(S(Tx), S(Ty)) \leq (L \cdot K) \cdot d_C(x, y)$$

for all carried source endpoints $x, y : C_V$*. The distance rows are the norm-induced rows of Definition 90.20, and the final comparison is the RealUp multiplication/order row associating the two operator bounds with the source distance.*

Proof. Apply Definition 90.50 to the carried row $\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$ and the middle endpoints Tx, Ty. This gives the target estimate

$$d_E(S(Tx), S(Ty)) \leq L \cdot d_D(Tx, Ty).$$

Apply Definition 90.50 again to $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ and the source endpoints x, y. The middle distance is bounded by

$$d_D(Tx, Ty) \leq K \cdot d_C(x, y).$$

Use the nonnegative bound rows for L and K, monotonicity of RealUp multiplication, and associativity of the displayed product endpoint. The ordered comparison transports $L \cdot d_D(Tx, Ty)$ across the second estimate and rewrites the resulting $L \cdot (K \cdot d_C(x, y))$ as $(L \cdot K) \cdot d_C(x, y)$. Repacking the two operator ledgers gives the displayed composite Lipschitz row. □

Theorem 90.62 (Bounded linear operator composition preserves regular Cauchy streams)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D, E*, carried rows* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *and* $\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$ *transport every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over* C *to a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over* E *by pointwise application of* $S \circ T$*. The transported observation rows and modulus ledger are the result of first applying the* T *transport and then applying the* S *transport.*

Proof. Start with a regular Cauchy stream $(X, \mu) : \mathbb{N} \to C_V$. Apply Definition 90.51 to $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$. The result is the pointwise stream $T \circ X$ over D, with carried observation rows and a modulus transformed through the RealUp bound row for K.

Apply Definition 90.51 a second time to $\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$ and the transported middle stream. The result is the stream $S \circ (T \circ X)$ over E, with its observation rows transported through the target endpoint fields of Γ.

Finally identify $S \circ (T \circ X)$ with $(S \circ T) \circ X$ by endpoint definition, and concatenate the two visible ledgers. The RealUp multiplication/order rows combine the two modulus transformations into the single composite bound $L \cdot K$, so the final stream is regular Cauchy for the target distance d_E. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator composition cauchy transport`

Theorem 90.63 (Banach bounded linear operator composition NameCert obligations)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D, E*, the carrier of Definition 90.60, together with Definition 90.61 and Definition 90.62, assembles the paper-side* $\mathsf{NameCert}$ *obligation set for composed Banach bounded linear operators. The carrier is* $\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda)$*, the classifier is componentwise* hsame *transport of maps and ledgers together with RealUp transport of the product bound, stability is inherited through the two operator carriers, exactness is the equivalence between the displayed composite rows and the two-stage endpoint action, and the ledger records the source, middle, target, classifier, compatibility, bound, and stream rows.*

Proof. The carrier field is exactly Definition 90.60. Its endpoint map is $S \circ T$, its bound endpoint is $L \cdot K$, and its ledger is the ordered concatenation $\Gamma \star \Lambda$.

Classifier transport is obtained in two stages. The first operator row moves source classifier comparisons through T, and the second operator row moves the resulting middle comparisons through S. RealUp classifier transport keeps the product bound $L \cdot K$ stable, while componentwise hsame transport keeps the concatenated ledger aligned with the two endpoint maps.

The Lipschitz field is Definition 90.61, and the regular-Cauchy field is Definition 90.62. These are the metric and stream obligations required for downstream operator certificates.

Exactness follows by unfolding the composite carrier. Any accepted composite row contains precisely the two accepted operator rows, the composed endpoint map, the RealUp product bound, and the visible concatenated ledger. Conversely, those fields repack as $\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda)$, so the composition surface adds no host-level quotient or unrecorded operation row. □

Theorem 90.64 (Banach bounded linear operator composition closure)**.** *Let C, D, E be $\mathsf{Banach}^{\uparrow}$ candidates with vector carriers C_V, D_V, E_V, classifiers $\sim_C, \sim_D, \sim_E$, norms ν_C, ν_D, ν_E, and norm-induced distances d_C, d_D, d_E. If*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and

$$\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$$

are carried, then the endpoint composite $S \circ T : C_V \to E_V$ is carried by

$$\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda).$$

The composed ledger $\Gamma \star \Lambda$ records the two operator ledgers and the RealUp multiplication, monotonicity, associativity, and non-negativity rows needed for the product bound $L \cdot K$.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator composition closure`

Proof. Unfold the operator carrier of Definition 90.49. For a carried source endpoint $x : C_V$, the target endpoint row of Λ carries $T(x) : D_V$. The target endpoint row of Γ then carries $S(Tx) : E_V$, which is the endpoint value of $(S \circ T)(x)$.

For classifier-respect, suppose $x \sim_C x'$ with both source endpoints carried. The classifier row of Λ gives $T(x) \sim_D T(x')$. Applying the classifier row of Γ gives $S(Tx) \sim_E S(Tx')$, which is the classifier row for the composite endpoint map.

For additivity, let $x, y : C_V$ be carried. The additive row of Λ gives

$$T(x +_C y) \sim_D T(x) +_D T(y).$$

Both middle endpoints in this comparison are carried by the D vector carrier, so the classifier-respect row of Γ transports it to

$$S(T(x +_C y)) \sim_E S(T(x) +_D T(y)).$$

The additive row of Γ, applied to the carried middle endpoints $T(x)$ and $T(y)$, gives

$$S(T(x) +_D T(y)) \sim_E S(Tx) +_E S(Ty).$$

Transitivity of the target vector classifier yields

$$(S \circ T)(x +_C y) \sim_E (S \circ T)(x) +_E (S \circ T)(y).$$

For homogeneity, let a be a carried RealUp scalar endpoint and let $x : C_V$ be carried. The scalar row of Λ gives

$$T(a \cdot_C x) \sim_D a \cdot_D T(x).$$

Transport this comparison through the classifier row of Γ, then apply the scalar row of Γ to the carried middle endpoint $T(x)$. The target classifier transitivity row gives

$$(S \circ T)(a \cdot_C x) \sim_E a \cdot_E (S \circ T)(x).$$

It remains to verify the product bound. The non-negativity rows in the two operator carriers give $K \geq 0$ and $L \geq 0$, so the RealUp multiplication row carries $L \cdot K \geq 0$. For a carried source endpoint x, the bound row of Γ applied to the carried middle endpoint $T(x)$ gives

$$\nu_E(S(Tx)) \leq L \cdot \nu_D(Tx).$$

The bound row of Λ gives

$$\nu_D(Tx) \leq K \cdot \nu_C(x).$$

Using $L \geq 0$, RealUp multiplication monotonicity transports the second comparison to

$$L \cdot \nu_D(Tx) \leq L \cdot (K \cdot \nu_C(x)).$$

RealUp associativity identifies the right-hand side with $(L \cdot K) \cdot \nu_C(x)$. Composing the displayed inequalities gives

$$\nu_E((S \circ T)(x)) \leq (L \cdot K) \cdot \nu_C(x),$$

which is the bound row required for $\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda)$.

The ledger clause is exactly the ordered concatenation described in Definition 90.60: it keeps the source-to-middle endpoint, classifier, linearity, and bound rows from Λ, the middle-to-target rows from Γ, and the RealUp product rows used above. These rows are precisely the fields of Definition 90.49 for the composite endpoint map, so the displayed composite operator carrier is carried. □

Theorem 90.65 (Banach bounded linear operator composition associativity)**.** *Let B, C, D, E be $\mathsf{Banach}^{\uparrow}$ candidates with vector carriers B_V, C_V, D_V, E_V and carried bounded linear operator rows*

$$\mathsf{BanachBLOp}(B, C, T, K, \Lambda),$$

$$\mathsf{BanachBLOp}(C, D, S, L, \Gamma),$$

and

$$\mathsf{BanachBLOp}(D, E, R, M, \Omega).$$

Then the two iterated composite carriers

$$\mathsf{BanachBLOp}(B, E, (R \circ S) \circ T, (M \cdot L) \cdot K, (\Omega \star \Gamma) \star \Lambda)$$

and

$$\mathsf{BanachBLOp}(B, E, R \circ (S \circ T), M \cdot (L \cdot K), \Omega \star (\Gamma \star \Lambda))$$

classify the same carried Banach bounded linear operator row after endpoint composition associativity identifies $((R \circ S) \circ T)(x)$ with $(R \circ (S \circ T))(x)$ for every carried source endpoint x, RealUp multiplication associativity identifies the two bound endpoints, and ledger associativity identifies the two ordered concatenations of the source-to-middle, middle-to-target, and target-to-final rows.

Proof. First form the left-associated carrier. Apply Definition 90.64 to the rows for S and R, obtaining $\mathsf{BanachBLOp}(C, E, R \circ S, M \cdot L, \Omega \star \Gamma)$. Applying the same closure theorem to this row and the row for T gives the displayed carrier with endpoint map $(R \circ S) \circ T$, bound $(M \cdot L) \cdot K$, and ledger $(\Omega \star \Gamma) \star \Lambda$.

Next form the right-associated carrier. Apply Definition 90.64 to the rows for T and S, obtaining $\mathsf{BanachBLOp}(B, D, S \circ T, L \cdot K, \Gamma \star \Lambda)$. Applying the closure theorem to this row and the row for R gives the displayed carrier with endpoint map $R \circ (S \circ T)$, bound $M \cdot (L \cdot K)$, and ledger $\Omega \star (\Gamma \star \Lambda)$.

Unfold the composition carrier of Definition 90.60. For a carried source endpoint $x : B_V$, both endpoint maps compute to $R(S(Tx))$. Thus the endpoint classifiers compare the same target endpoint after definitionally reassociating composition.

The metric and stream fields are the same obligations in the two bracketings. Definition 90.61 supplies the composite Lipschitz row at each binary step, and RealUp multiplication associativity rewrites $((M \cdot L) \cdot K) \cdot d_B(x, y)$ to $(M \cdot (L \cdot K)) \cdot d_B(x, y)$. Likewise, Definition 90.62 transports regular Cauchy streams by pointwise application of the same endpoint map $x \mapsto R(S(Tx))$.

Finally, the visible ledgers contain exactly the rows from Λ, Γ, and Ω together with the RealUp product and order rows used by the two applications of composition closure. Ledger associativity identifies $(\Omega \star \Gamma) \star \Lambda$ with $\Omega \star (\Gamma \star \Lambda)$ as the same ordered source-middle-target record, so the two displayed carriers classify the same Banach bounded linear operator row. □

Theorem 90.66 (Banach bounded linear operator identity units)**.** *Let C, D be $\mathsf{Banach}^{\uparrow}$ candidates with vector carriers C_V, D_V, classifiers $\sim_C, \sim_D$, norms ν_C, ν_D, and norm-induced distances d_C, d_D. Let*

$$\mathrm{id}_C : C_V \to C_V, \qquad \mathrm{id}_C(x) := x,$$

and

$$\mathrm{id}_D : D_V \to D_V, \qquad \mathrm{id}_D(y) := y.$$

Assume the carried operator row

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and assume that the source and target identity maps carry the unit bound:

$$\mathsf{BanachBLOp}(C, C, \mathrm{id}_C, 1_{\mathsf{Real}^{\uparrow}}, \mathcal{I}_C), \qquad \mathsf{BanachBLOp}(D, D, \mathrm{id}_D, 1_{\mathsf{Real}^{\uparrow}}, \mathcal{I}_D).$$

Assume also that the identity ledgers are unit ledgers for Λ: the RealUp multiplication rows classify $K \cdot 1_{\mathsf{Real}^{\uparrow}}$ and $1_{\mathsf{Real}^{\uparrow}} \cdot K$ with K, and the ordered ledgers $\Lambda \star \mathcal{I}_C$ and $\mathcal{I}_D \star \Lambda$ componentwise transport back to Λ by collapsing only the displayed identity endpoint rows and these RealUp multiplication-unit rows. Then the two unit composites are carried:

$$\mathsf{BanachBLOp}(C, D, T \circ \mathrm{id}_C, K \cdot 1_{\mathsf{Real}^{\uparrow}}, \Lambda \star \mathcal{I}_C)$$

and

$$\mathsf{BanachBLOp}(C, D, \mathrm{id}_D \circ T, 1_{\mathsf{Real}^{\uparrow}} \cdot K, \mathcal{I}_D \star \Lambda).$$

Moreover both composite operator representatives are classified with the original representative by the Banach bounded-operator classifier of Definition 90.49:

$$(T \circ \mathrm{id}_C, K \cdot 1_{\mathsf{Real}^{\uparrow}}, \Lambda \star \mathcal{I}_C) \sim_{\mathrm{op}(C,D)} (T, K, \Lambda)$$

and

$$(\mathrm{id}_D \circ T, 1_{\mathsf{Real}^{\uparrow}} \cdot K, \mathcal{I}_D \star \Lambda) \sim_{\mathrm{op}(C,D)} (T, K, \Lambda).$$

Lean4 ✓ BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator identity unit composites

Proof. Apply Definition 90.64 to the carried source identity row and the carried row for T. With the middle object equal to C, the composite endpoint map is $T \circ \mathrm{id}_C$, the product bound is $K \cdot 1_{\mathsf{Real}^\uparrow}$, and the concatenated ledger is $\Lambda \star \mathcal{I}_C$. This gives the displayed right unit carrier.

Apply the same composition-closure theorem to the carried row for T and the carried target identity row. With the middle object equal to D, the composite endpoint map is $\mathrm{id}_D \circ T$, the product bound is $1_{\mathsf{Real}^\uparrow} \cdot K$, and the concatenated ledger is $\mathcal{I}_D \star \Lambda$. This gives the displayed left unit carrier.

It remains to identify the two composites with the original representative. Unfold the operator classifier specified in Definition 90.49. Its comparison has three visible fields: endpoint-map comparison, RealUp bound comparison, and componentwise ledger transport.

For every carried source endpoint $x : C_V$, the right unit endpoint computes as

$$(T \circ \mathrm{id}_C)(x) = T(\mathrm{id}_C(x)) = T(x),$$

and the left unit endpoint computes as

$$(\mathrm{id}_D \circ T)(x) = \mathrm{id}_D(T(x)) = T(x).$$

The target vector classifier reflexivity row therefore supplies the endpoint-map component of both operator-classifier comparisons.

The RealUp bound component is exactly the assumed multiplication-unit transport:

$$K \cdot 1_{\mathsf{Real}^\uparrow} \sim_{\mathsf{Real}^\uparrow} K, \qquad 1_{\mathsf{Real}^\uparrow} \cdot K \sim_{\mathsf{Real}^\uparrow} K.$$

The ledger component is exactly the unit-ledger hypothesis: $\Lambda \star \mathcal{I}_C$ and $\mathcal{I}_D \star \Lambda$ transport componentwise to Λ after the identity endpoint rows and the RealUp unit rows are collapsed. These three components are precisely the fields of the Banach bounded-operator classifier, so both unit composites are classified with (T, K, Λ). □

Theorem 90.67 (Banach bounded linear operator left zero annihilation)**.** *Let H_0, H_1, H_2 be $\mathsf{Banach}^\uparrow$ candidates with vector carriers $H_{0,V}, H_{1,V}, H_{2,V}$, classifiers $\sim_{H_0}, \sim_{H_1}, \sim_{H_2}$, and zero endpoints $0_{H_0}, 0_{H_1}, 0_{H_2}$. Let*

$$\mathsf{BanachBLOp}(H_1, H_2, T, L, \Gamma)$$

be a carried bounded linear operator row, and let

$$\mathsf{BanachBLOp}(H_0, H_1, 0_{H_0,H_1}, 0_{\mathsf{Real}^\uparrow}, \Lambda_{01})$$

and

$$\mathsf{BanachBLOp}(H_0, H_2, 0_{H_0,H_2}, 0_{\mathsf{Real}^\uparrow}, \Lambda_{02})$$

be the carried zero rows furnished by Definition 90.53, so that

$$0_{H_0,H_1}(x) = 0_{H_1}, \qquad 0_{H_0,H_2}(x) = 0_{H_2}$$

for every carried source endpoint $x : H_{0,V}$. Assume also that the RealUp multiplication rows classify $L \cdot 0_{\mathsf{Real}^\uparrow}$ with $0_{\mathsf{Real}^\uparrow}$, and that the concatenated ledger $\Gamma \star \Lambda_{01}$ componentwise transports to Λ_{02} after rewriting only the displayed endpoint rows along the target classifier comparison $T(0_{H_1}) \sim_{H_2} 0_{H_2}$ and the bound row along $L \cdot 0_{\mathsf{Real}^\uparrow} \sim_{\mathsf{Real}^\uparrow} 0_{\mathsf{Real}^\uparrow}$. Then

$$\mathsf{BanachBLOp}(H_0, H_2, T \circ 0_{H_0,H_1}, L \cdot 0_{\mathsf{Real}^\uparrow}, \Gamma \star \Lambda_{01})$$

is carried, and its operator representative is classified with the zero representative on (H_0, H_2):

$$(T \circ 0_{H_0,H_1}, L \cdot 0_{\mathsf{Real}^\uparrow}, \Gamma \star \Lambda_{01}) \sim_{\mathrm{op}(H_0,H_2)} (0_{H_0,H_2}, 0_{\mathsf{Real}^\uparrow}, \Lambda_{02}).$$

Proof. Apply Definition 90.64 to the carried zero row from H_0 to H_1 and the carried row for T. With middle object H_1, the composite endpoint map is $T \circ 0_{H_0,H_1}$, the product bound is $L \cdot 0_{\mathsf{Real}^\uparrow}$, and the concatenated ledger is $\Gamma \star \Lambda_{01}$. This yields the carried row

$$\mathsf{BanachBLOp}(H_0, H_2, T \circ 0_{H_0,H_1}, L \cdot 0_{\mathsf{Real}^\uparrow}, \Gamma \star \Lambda_{01}).$$

It remains to compare this composite representative with the carried zero representative on (H_0, H_2). Unfold the operator classifier specified in Definition 90.49. As in the unit proof, its comparison has three visible fields: endpoint-map comparison, RealUp bound comparison, and componentwise ledger transport.

For the endpoint-map component, fix a carried source endpoint $x : H_{0,V}$. By definition of the source zero map,

$$(T \circ 0_{H_0,H_1})(x) = T(0_{H_1}),$$

while by definition of the target zero map,

$$0_{H_0,H_2}(x) = 0_{H_2}.$$

It therefore suffices to prove $T(0_{H_1}) \sim_{H_2} 0_{H_2}$.

Because Λ_{01} is a carried zero ledger, the endpoint 0_{H_1} is carried in $H_{1,V}$. The source scalar-zero row in H_1 gives

$$0_{\mathsf{Real}^\uparrow} \cdot_{H_1} 0_{H_1} \sim_{H_1} 0_{H_1}.$$

Apply the classifier-respect row of Γ to transport this comparison through T:

$$T(0_{\mathsf{Real}^\uparrow} \cdot_{H_1} 0_{H_1}) \sim_{H_2} T(0_{H_1}).$$

By symmetry of the target vector classifier,

$$T(0_{H_1}) \sim_{H_2} T(0_{\mathsf{Real}^\uparrow} \cdot_{H_1} 0_{H_1}).$$

Now apply the homogeneity row of Γ at the carried scalar endpoint $0_{\mathsf{Real}^\uparrow}$ and the carried source endpoint 0_{H_1}:

$$T(0_{\mathsf{Real}^\uparrow} \cdot_{H_1} 0_{H_1}) \sim_{H_2} 0_{\mathsf{Real}^\uparrow} \cdot_{H_2} T(0_{H_1}).$$

Finally, the target scalar-zero row in H_2 gives

$$0_{\mathsf{Real}^\uparrow} \cdot_{H_2} T(0_{H_1}) \sim_{H_2} 0_{H_2}.$$

Transitivity of $\sim_{H_2}$ combines the last three displayed comparisons into

$$T(0_{H_1}) \sim_{H_2} 0_{H_2}.$$

Since this comparison is independent of x, it supplies the endpoint-map component of the operator-classifier comparison between $T \circ 0_{H_0,H_1}$ and $0_{H_0,H_2}$.

The RealUp bound component is exactly the assumed right-zero multiplication comparison

$$L \cdot 0_{\mathsf{Real}^\uparrow} \sim_{\mathsf{Real}^\uparrow} 0_{\mathsf{Real}^\uparrow}.$$

The ledger component is the remaining hypothesis: after rewriting the endpoint rows along the just-proved target comparison $T(0_{H_1}) \sim_{H_2} 0_{H_2}$ and rewriting the bound row along $L \cdot 0_{\mathsf{Real}^\uparrow} \sim_{\mathsf{Real}^\uparrow} 0_{\mathsf{Real}^\uparrow}$, the concatenated ledger $\Gamma \star \Lambda_{01}$ transports componentwise to the zero ledger Λ_{02}.

These three components are precisely the fields of the Banach bounded-operator classifier. Hence the composite representative $(T \circ 0_{H_0,H_1}, L \cdot 0_{\mathsf{Real}^\uparrow}, \Gamma \star \Lambda_{01})$ is classified with the zero representative $(0_{H_0,H_2}, 0_{\mathsf{Real}^\uparrow}, \Lambda_{02})$ on (H_0, H_2). □

Theorem 90.68 (Banach bounded linear operator zero right-composition annihilation)**.** *Let H_0, H_1, H_2 be $\mathsf{Banach}^{\uparrow}$ candidates with vector carriers $H_{0,V}, H_{1,V}, H_{2,V}$, classifiers $\sim_{H_0}, \sim_{H_1}, \sim_{H_2}$, and zero endpoints $0_{H_0}, 0_{H_1}, 0_{H_2}$. Let*

$$\mathsf{BanachBLOp}(H_0, H_1, T, L, \Gamma)$$

be a carried bounded linear operator row, and let

$$\mathsf{BanachBLOp}(H_1, H_2, 0_{H_1,H_2}, 0_{\mathsf{Real}^{\uparrow}}, \Lambda_{12})$$

and

$$\mathsf{BanachBLOp}(H_0, H_2, 0_{H_0,H_2}, 0_{\mathsf{Real}^{\uparrow}}, \Lambda_{02})$$

be the carried zero rows furnished by Definition 90.53, so that

$$0_{H_1,H_2}(y) = 0_{H_2}, \qquad 0_{H_0,H_2}(x) = 0_{H_2}$$

for every carried endpoint $y : H_{1,V}$ and every carried endpoint $x : H_{0,V}$. Assume also that the RealUp multiplication rows classify $0_{\mathsf{Real}^{\uparrow}} \cdot L$ with $0_{\mathsf{Real}^{\uparrow}}$, and that the concatenated ledger $\Lambda_{12} \star \Gamma$ componentwise transports to Λ_{02} after rewriting only the displayed endpoint rows along the target classifier reflexivity comparison $0_{H_2} \sim_{H_2} 0_{H_2}$ and the bound row along $0_{\mathsf{Real}^{\uparrow}} \cdot L \sim_{\mathsf{Real}^{\uparrow}} 0_{\mathsf{Real}^{\uparrow}}$. Then

$$\mathsf{BanachBLOp}(H_0, H_2, 0_{H_1,H_2} \circ T, 0_{\mathsf{Real}^{\uparrow}} \cdot L, \Lambda_{12} \star \Gamma)$$

is carried, and its operator representative is classified with the zero representative on (H_0, H_2):

$$(0_{H_1,H_2} \circ T, 0_{\mathsf{Real}^{\uparrow}} \cdot L, \Lambda_{12} \star \Gamma) \sim_{\mathrm{op}(H_0,H_2)} (0_{H_0,H_2}, 0_{\mathsf{Real}^{\uparrow}}, \Lambda_{02}).$$

Proof. Apply Definition 90.64 to the carried row for T and the carried zero row from H_1 to H_2. With middle object H_1, the composite endpoint map is $0_{H_1,H_2} \circ T$, the product bound is $0_{\mathsf{Real}^{\uparrow}} \cdot L$, and the concatenated ledger is $\Lambda_{12} \star \Gamma$. This yields the carried row

$$\mathsf{BanachBLOp}(H_0, H_2, 0_{H_1,H_2} \circ T, 0_{\mathsf{Real}^{\uparrow}} \cdot L, \Lambda_{12} \star \Gamma).$$

It remains to compare this composite representative with the carried zero representative on (H_0, H_2). Unfold the operator classifier specified in Definition 90.49. Its comparison has three visible fields: endpoint-map comparison, RealUp bound comparison, and componentwise ledger transport.

For the endpoint-map component, fix a carried source endpoint $x : H_{0,V}$. The target endpoint row of Γ carries $T(x)$ in $H_{1,V}$. Hence the defining row of the zero operator $0_{H_1,H_2}$ applies to $T(x)$ and gives

$$(0_{H_1,H_2} \circ T)(x) = 0_{H_1,H_2}(T(x)) = 0_{H_2}.$$

The zero representative on (H_0, H_2) also satisfies

$$0_{H_0,H_2}(x) = 0_{H_2}.$$

Therefore the target classifier reflexivity row at 0_{H_2} supplies

$$(0_{H_1,H_2} \circ T)(x) \sim_{H_2} 0_{H_0,H_2}(x).$$

Since x was arbitrary, this is the endpoint-map component of the operator-classifier comparison.

The RealUp bound component is exactly the assumed left-zero multiplication comparison

$$0_{\mathsf{Real}^{\uparrow}} \cdot L \sim_{\mathsf{Real}^{\uparrow}} 0_{\mathsf{Real}^{\uparrow}}.$$

The ledger component is the remaining hypothesis: after rewriting the endpoint rows along $0_{H_2} \sim_{H_2} 0_{H_2}$ and rewriting the bound row along $0_{\mathsf{Real}^\uparrow} \cdot L \sim_{\mathsf{Real}^\uparrow} 0_{\mathsf{Real}^\uparrow}$, the concatenated ledger $\Lambda_{12} \star \Gamma$ transports componentwise to the zero ledger Λ_{02}.

These three components are precisely the fields of the Banach bounded-operator classifier. Hence the composite representative $(0_{H_1,H_2} \circ T, 0_{\mathsf{Real}^\uparrow} \cdot L, \Lambda_{12} \star \Gamma)$ is classified with the zero representative $(0_{H_0,H_2}, 0_{\mathsf{Real}^\uparrow}, \Lambda_{02})$ on (H_0, H_2). □

Theorem 90.69 (Banach bounded linear operator scalar action distributes through right composition)**.** *Let H_0, H_1, H_2 be $\mathsf{Banach}^\uparrow$ candidates as in Definition 90.2, with vector carriers $H_{0,V}, H_{1,V}, H_{2,V}$, classifiers $\sim_{H_0}, \sim_{H_1}, \sim_{H_2}$, scalar actions $\cdot_{H_1}, \cdot_{H_2}$, norms $\nu_{H_0}, \nu_{H_1}, \nu_{H_2}$, and norm-induced distances. Let*

$$\mathsf{BanachBLOp}(H_0, H_1, T, L_T, \Lambda_T)$$

and

$$\mathsf{BanachBLOp}(H_1, H_2, S, L_S, \Lambda_S)$$

be carried bounded linear operator rows. Let c be a carried RealUp scalar endpoint with $c \geq 0$, and let $|c|_{\mathsf{Real}^\uparrow}$ be the carried absolute-value endpoint exposed by the shared RealAlgOrder *row, with the non-negative scalar row classifying $|c|_{\mathsf{Real}^\uparrow}$ with c. Write*

$$(c \boxtimes_{H_1} T)(x) := c \cdot_{H_1} T(x)$$

and

$$(c \boxtimes_{H_2} (S \circ T))(x) := c \cdot_{H_2} S(Tx).$$

Then the right scalar-action composite is carried by a Banach bounded linear operator row

$$\mathsf{BanachBLOp}(H_0, H_2, S \circ (c \boxtimes_{H_1} T), (c \cdot L_T) \cdot L_S, \Theta_{S(cT)}),$$

where $\Theta_{S(cT)}$ is obtained from the scalar-multiple ledger for T, the composition ledger with S, and the RealUp product transport identifying $L_S \cdot (c \cdot L_T)$ with $(c \cdot L_T) \cdot L_S$. Moreover this operator representative is classified with the scalar multiple of the ordinary composite over the same source and target Banach carriers:

$$(S \circ (c \boxtimes_{H_1} T), (c \cdot L_T) \cdot L_S, \Theta_{S(cT)}) \sim_{\mathrm{op}(H_0,H_2)} (c \boxtimes_{H_2} (S \circ T), (c \cdot L_T) \cdot L_S, \Theta_{c(ST)}),$$

where $\Theta_{c(ST)}$ is obtained from the composition ledger for $S \circ T$, the scalar-multiple ledger by c, and the same RealUp product-normalization rows.

Proof. Apply Definition 90.57 to the carried row for T and the carried scalar endpoint c. This yields

$$\mathsf{BanachBLOp}(H_0, H_1, c \boxtimes_{H_1} T, |c|_{\mathsf{Real}^\uparrow} \cdot L_T, \langle c \rangle \boxtimes \Lambda_T).$$

The RealUp absolute-value row for the non-negative scalar transports the bound endpoint $|c|_{\mathsf{Real}^\uparrow} \cdot L_T$ to $c \cdot L_T$, so the same scalar-multiple operator row is carried with normalized bound $c \cdot L_T$ and the transported ledger.

Now apply Definition 90.64 to this scalar-multiple row and the carried row for S. The composite endpoint map is $S \circ (c \boxtimes_{H_1} T)$ and the immediate product bound is

$$L_S \cdot (c \cdot L_T).$$

RealUp commutativity and associativity transport this product to

$$(c \cdot L_T) \cdot L_S,$$

and product non-negativity follows from $c \geq 0$, the bound row $L_T \geq 0$, and the bound row $L_S \geq 0$. Transporting the composition ledger across this RealUp classifier comparison gives the displayed carried row

$$\mathsf{BanachBLOp}(H_0, H_2, S \circ (c \boxtimes_{H_1} T), (c \cdot L_T) \cdot L_S, \Theta_{S(cT)}).$$

For the comparison with the scalar multiple of the ordinary composite, first apply Definition 90.64 to the rows for T and S. This gives

$$\mathsf{BanachBLOp}(H_0, H_2, S \circ T, L_S \cdot L_T, \Lambda_S \star \Lambda_T).$$

Applying Definition 90.57 to this row and c gives the scalar-multiple representative

$$\mathsf{BanachBLOp}(H_0, H_2, c \boxtimes_{H_2} (S \circ T), |c|_{\mathsf{Real}^\uparrow} \cdot (L_S \cdot L_T), \langle c \rangle \boxtimes (\Lambda_S \star \Lambda_T)).$$

The classifier comparison $|c|_{\mathsf{Real}^\uparrow} \sim_{\mathsf{Real}^\uparrow} c$, together with RealUp associativity and commutativity, transports this bound to

$$(c \cdot L_T) \cdot L_S.$$

The transported ledger is $\Theta_{c(ST)}$.

It remains to compare the two displayed representatives in the operator classifier of Definition 90.49. For a carried source endpoint $x : H_{0,V}$, the target endpoint row of Λ_T carries $T(x)$ in $H_{1,V}$. The homogeneity row of Λ_S, applied to the carried scalar c and the carried middle endpoint $T(x)$, gives

$$S(c \cdot_{H_1} T(x)) \sim_{H_2} c \cdot_{H_2} S(Tx).$$

By the two endpoint definitions, this is exactly

$$(S \circ (c \boxtimes_{H_1} T))(x) \sim_{H_2} (c \boxtimes_{H_2} (S \circ T))(x).$$

Since x was arbitrary, these comparisons supply the endpoint-map component of the Banach bounded-operator classifier.

The bound component is RealUp classifier reflexivity at the normalized bound $(c \cdot L_T) \cdot L_S$. For the ledger component, both ledgers contain the same carried scalar row for c, the two input operator ledgers Λ_T and Λ_S, the scalar-action rows in H_1 and H_2, the homogeneity row of S used in the endpoint comparison, and the same RealUp absolute-value, commutativity, associativity, monotonicity, and product non-negativity rows. Componentwise ledger transport along the endpoint comparison just proved and the normalized RealUp product comparison therefore identifies $\Theta_{S(cT)}$ with $\Theta_{c(ST)}$.

These endpoint, bound, and ledger components are precisely the fields of the Banach bounded-operator classifier on (H_0, H_2). Hence

$$(S \circ (c \boxtimes_{H_1} T), (c \cdot L_T) \cdot L_S, \Theta_{S(cT)}) \sim_{\mathrm{op}(H_0,H_2)} (c \boxtimes_{H_2} (S \circ T), (c \cdot L_T) \cdot L_S, \Theta_{c(ST)}).$$

□

The ten-field fit is as follows. The traditional target is Banach space. The BEDC source is the singleton $\mathsf{Norm}^\uparrow$ carrier, the $\mathsf{CompleteMetric}^\uparrow$ singleton limit witness, and the RealUp constant zero endpoint. The carrier is the singleton normed history carrier. The classifier is ρ_{e} on vectors and hsame on displayed limit histories. Stability is supplied by $\mathsf{Norm}^\uparrow$ transport and $\mathsf{CompleteMetricLimitWitness}$ transport. The ledger records the norm-zero endpoint, metric-distance rows, the Cauchy modulus, and the limit rows. The core theorem list is Definition 90.5, Definition 90.16, and Definition 90.17. Exactness is the equivalence between zero distance and the singleton classifier. The standard bridge field is the singleton surface of Definition 90.79. The status is a paper proof target for the namespace $\mathsf{BEDC.Derived.BanachUp}$.

Theorem 90.70 (Banach norm-distance zero exactness)**.** *Let X be a concrete Banach carrier and let* $\mathsf{BanachMetricWitness}_X(x, y, d)$ *be a displayed norm-induced distance row. If the* $\mathsf{Real}^{\uparrow}$ *classifier identifies d with the zero real endpoint, then the Banach point classifier identifies the carried representatives x and y.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton norm distance zero exactness`

Proof. Unfold the carrier in Definition 90.1. The metric row is not independent data: it records carried representatives x and y, a displayed difference history, and the $\mathsf{Norm}^{\uparrow}$ value of that difference as the real endpoint d.

Use the input RealUp zero classification to read the norm value of the difference as zero. The positive-definiteness row of $\mathsf{Norm}^{\uparrow}$ then reflects this zero norm back to the vector-history classifier, so the displayed difference history is classified with the vector zero.

The vector-space zero-difference law turns that classifier row into equality of the two carried representatives in the Banach point classifier. Repacking the carried representatives, difference row, and norm ledger along hsame gives the required classifier comparison between x and y. □

Theorem 90.71 (Banach complete-metric limit uniqueness)**.** *Let X be a concrete Banach carrier, and let s be a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over X for the norm-induced metric. If*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

and

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu', \ell', L')$$

are two displayed limit rows for the same stream, then the Banach point classifier identifies the carried limit histories L and L'.

Proof. First read both displayed rows through Definition 128.2. They give carried limit histories L and L', convergence moduli, and norm-induced late-distance rows from the same regular Cauchy stream to each candidate limit.

The previous theorem supplies the separated-metric zero-distance field for the Banach metric: whenever a displayed $\mathsf{BanachMetricWitness}_X(u, v, d)$ has zero RealUp distance, the Banach point classifier identifies u and v.

Apply Definition 128.6 to the two complete-metric limit rows using that separatedness field. The complete-metric uniqueness theorem returns classifier equality of the two limit histories, and the classifier is exactly the hsame-based Banach point classifier fixed in Definition 90.1. □

Theorem 90.72 (Banach regular Cauchy stream transport)**.** *Let X be a concrete Banach carrier. Componentwise* hsame *transport of a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over X, its observation rows, modulus histories, candidate limit histories, norm-distance rows, and* $\mathsf{Rat}^{\uparrow}$ *comparison ledgers preserves the displayed Banach complete-metric witness package.*

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton regular cauchy stream transport`

Proof. Transport the Banach carrier fields first. By Definition 90.25, the carried representatives, norm rows, metric rows, stream observations, modulus histories, limit histories, and late-observation ledgers remain a concrete Banach carrier package after componentwise hsame transport.

Next transport the complete-metric row. Apply Definition 128.4 to the transported $\mathsf{StreamName}^{\uparrow}$ observations, modulus history, candidate limit, distance histories, and $\mathsf{Rat}^{\uparrow}$ comparison rows. Its

conclusion is the transported $\mathsf{CompleteMetricLimitWitness}$ row for the same norm-induced metric shape.

Finally combine the two transported packages. The Banach transport theorem keeps the metric rows induced by the transported norm ledgers, and the complete-metric stability theorem keeps the regular Cauchy and convergence fields aligned with those rows. Repacking the fields gives the preserved Banach complete-metric witness package. □

Corollary 90.73 (Banach zero-distance exactness determinacy)**.** *Let X be a concrete Banach carrier, and let s be a regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream over the norm-induced metric. Suppose two accepted consumer rows for the same stream expose displayed limit packages*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_0, \ell_0, L_0) \quad \textit{and} \quad \mathsf{CompleteMetricLimitWitness}(X, s, \mu_1, \ell_1, L_1),$$

and suppose their advertised exactness decisions are displayed by Banach norm-distance rows

$$\mathsf{BanachMetricWitness}_X(L_0, z_0, d_0) \quad \textit{and} \quad \mathsf{BanachMetricWitness}_X(L_1, z_1, d_1)$$

whose $\mathsf{Real}^{\uparrow}$ endpoints are both classified with the zero real endpoint. Then the Banach point classifier identifies z_0 and z_1. Consequently any accepted consumer-facing use of the Banach zero-distance exactness decision for this stream factors through the same visible finite route: the stream row, the modulus row, the displayed limit row, the norm-induced distance row, and the $\mathsf{Real}^{\uparrow}$ zero endpoint. No additional coordinate, anonymous metric row, or second classifier source can change the exactness decision named by the chapter.

Proof. First apply Definition 90.71 to the two displayed complete-metric limit witnesses for the same regular Cauchy stream s. It returns a Banach point-classifier comparison between the carried limit histories L_0 and L_1. This comparison is read in the classifier fixed by Definition 90.1, so it is a comparison at the same Banach surface as the consumer rows.

Next apply Definition 90.70 to the first advertised exactness row. The row $\mathsf{BanachMetricWitness}_X(L_0, z_0, d_0)$ is a norm-induced Banach distance row, and the displayed $\mathsf{Real}^{\uparrow}$ zero classification of d_0 turns it into a Banach point-classifier comparison between L_0 and z_0. Applying the same theorem to $\mathsf{BanachMetricWitness}_X(L_1, z_1, d_1)$ gives a Banach point-classifier comparison between L_1 and z_1.

Compose these three classifier comparisons, using symmetry for the comparison between L_0 and z_0 if the classifier is read in the opposite direction. The result identifies z_0 and z_1 in the Banach point classifier. Thus two accepted consumer decisions over the same stream cannot diverge after their visible rows are fixed.

It remains only to exclude a hidden source of the decision. By Definition 128.2, an accepted limit package exposes exactly the stream, modulus, carried limit, threshold, metric-distance, and $\mathsf{Rat}^{\uparrow}$ comparison ledgers. By Definition 90.1, the Banach metric row is the norm-induced row $\mathsf{BanachMetricWitness}_X$, and by Definition 90.70 the zero endpoint is read back through the Banach point classifier. Hence every accepted use has already passed through the listed rows, and replacing them by an unlisted coordinate or duplicate classifier would no longer be a use of the displayed Banach zero-distance exactness interface. □

Definition 90.74 (Banach bounded linear operator limit comparison)**.** *Let C, D be $\mathsf{Banach}^{\uparrow}$ candidates, let $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ be a carried bounded-linear operator row, and let s be a regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream over the norm-induced metric of C with observations X_n. A bounded-linear operator limit comparison datum is the BEDC package*

$$\mathsf{BanachBLOpLimit}(C, D, T, K, \Lambda, s, \mu, \ell, L, s_T, \mu_T, \ell_T, L_T)$$

consisting of the source complete-metric limit witness

$$\mathsf{CompleteMetricLimitWitness}(C, s, \mu, \ell, L),$$

the pointwise target stream s_T with observations $T(X_n)$, the displayed target limit history $\ell_T := T(\ell)$, a target complete-metric limit witness

$$\mathsf{CompleteMetricLimitWitness}(D, s_T, \mu_T, \ell_T, L_T),$$

and the visible operator ledger Λ recording the source endpoint row, target endpoint row, source and target norm rows, additive and scalar compatibility rows, bound endpoint K, and bound-comparison rows used to turn source late-distance estimates into target late-distance estimates.

Theorem 90.75 (Bounded linear operator transports limit witnesses)**.** *Let C, D be $\mathsf{Banach}^{\uparrow}$ candidates, let $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ be carried, and let*

$$\mathsf{CompleteMetricLimitWitness}(C, s, \mu, \ell, L)$$

be a source limit witness for a regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream s over the norm-induced metric of C. Then the pointwise stream s_T with observations $T(s(n))$ admits a complete-metric limit witness over D with limit history $T(\ell)$. The witness is the target component of Definition 90.74.

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator limit witness transport`

Proof. First build the target stream. Apply Definition 90.51 to the carried operator row and the source regular Cauchy stream. Its conclusion supplies the regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream whose observations are $T(s(n))$, with target endpoint rows read from Λ.

Next transport the candidate limit endpoint. The source witness carries ℓ as a point of C, and the target endpoint row of Definition 90.49 carries $T(\ell)$ as a point of D. The same ledger supplies the target norm row and all classifier transport needed to read $T(\ell)$ at the Banach point surface.

It remains to assemble convergence. Fix a positive target tolerance. Use the RealUp bound row for K to choose a positive source tolerance whose product with K is below the requested target tolerance. The source $\mathsf{CompleteMetricLimitWitness}$ gives a threshold after which $d_C(s(n), \ell)$ is below that source tolerance.

For every late observation, apply Definition 90.50 to $s(n)$ and ℓ. The displayed estimate

$$d_D(T(s(n)), T(\ell)) \leq K \cdot d_C(s(n), \ell)$$

combines with the source convergence bound and the RealUp comparison chosen above. Repacking the target stream, target limit, late-distance rows, and comparison ledgers gives the required $\mathsf{CompleteMetricLimitWitness}(D, s_T, \mu_T, T(\ell), L_T)$. □

Theorem 90.76 (Bounded linear operator limit commutation)**.** *Let C, D be $\mathsf{Banach}^{\uparrow}$ candidates, let $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ be carried, and suppose s has source limit witness*

$$\mathsf{CompleteMetricLimitWitness}(C, s, \mu, \ell, L).$$

If a target complete-metric witness

$$\mathsf{CompleteMetricLimitWitness}(D, s_T, \mu', \ell', L')$$

is supplied for the pointwise stream $s_T(n) = T(s(n))$, then the Banach point classifier of D identifies ℓ' with $T(\ell)$.

Proof. First apply Definition 90.75 to the source limit witness. This constructs a target complete-metric limit witness for the same pointwise stream s_T whose displayed limit history is $T(\ell)$.

Now compare the two target witnesses. The supplied witness has limit ℓ', while the transported witness has limit $T(\ell)$; both are complete-metric limit witnesses for the same target stream over the same Banach carrier D.

Apply Definition 90.71. Its separatedness input is the Banach norm-distance zero exactness already recorded for the target carrier, and its conclusion is exactly the D-point classifier relation between ℓ' and $T(\ell)$. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingletonBoundedLinearOperator limit commutation`

Theorem 90.77 (Bounded linear operator limit NameCert surface)**.** *For* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D*, the comparison datum of Definition 90.74, together with Definition 90.75 and Definition 90.76, assembles the paper-side naming-certificate surface for bounded-linear operator limits. The carrier is the displayed pair of source and target* $\mathsf{CompleteMetricLimitWitness}$ *rows connected by the operator ledger* Λ*; the classifier is componentwise* hsame *transport of the source stream, source limit, pointwise target stream, target limit, bound row, and late-distance ledgers; stability is inherited from the Banach,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Norm}^{\uparrow}$ *rows; exactness is the classification of any supplied target limit with* $T(\ell)$*; and the ledger fields are the visible operator, norm, stream, limit, distance, and bound-comparison rows.*

Proof. The carrier field is the package displayed in Definition 90.74. It names the source Banach carrier, target Banach carrier, bounded-linear operator row, source complete-metric witness, pointwise target stream, transported limit, target complete-metric witness, and the visible operator ledger.

The classifier field is componentwise transport. Banach endpoint transport moves source and target carried points, $\mathsf{Norm}^{\uparrow}$ transport moves norm and metric rows, $\mathsf{StreamName}^{\uparrow}$ transport moves observations and moduli, RealUp transport moves the bound and order comparisons, and $\mathsf{CompleteMetric}^{\uparrow}$ transport repacks the transported late-distance ledgers as complete-metric witnesses.

The stability field is the witness-transport theorem. Apply Definition 90.75; it turns the source convergence row, the Cauchy transport row, and the Lipschitz estimate into the target complete-metric witness with limit $T(\ell)$.

The exactness field is the commutation theorem. Apply Definition 90.76; it compares any supplied target witness for the pointwise stream with the transported witness and returns the target Banach classifier relation to $T(\ell)$.

Thus the bounded-linear operator surface now exposes the standard limit transport obligation using only the existing Banach, CompleteMetricUp, StreamNameUp, RealUp, and NormUp certificate rows, with no additional parameter-transport schema. □

Lemma 90.78 (Bounded linear operator limit comparison minimal witness)**.** *Let* C, D *be* $\mathsf{Banach}^{\uparrow}$ *candidates, let* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *be carried, and let*

$$\mathsf{CompleteMetricLimitWitness}(C, s, \mu, \ell, L)$$

be a source limit witness for a regular Cauchy $\mathsf{StreamName}^{\uparrow}$ *stream* s*. For the comparison datum*

$$\mathsf{BanachBLOpLimit}(C, D, T, K, \Lambda, s, \mu, \ell, L, s_T, \mu_T, T(\ell), L_T)$$

of Definition 90.74, the minimal consumer-facing witness is the displayed target component supplied by Definition 90.75: the pointwise stream row $s_T(n) = T(s(n))$*, the selected target endpoint* $T(\ell)$*,*

the transported modulus μ_T, the late-distance ledger L_T, and the operator ledger Λ. Every accepted local admission, consumption, route, transport, or readback of the bounded-linear-operator limit comparison factors through these rows and through the rows listed by Definition 90.77. It cannot add an ambient limit selector, an off-ledger continuity principle, a quotient completion object, a compactness row, or any unlisted coordinate outside the same displayed packet.

Proof. Unfold Definition 90.74. The comparison datum contains exactly the source complete-metric witness, the operator row with ledger Λ, the pointwise target stream, the displayed target endpoint $T(\ell)$, the target complete-metric witness, and the visible operator, norm, stream, limit, distance, and bound-comparison ledgers. Thus a consumer of the comparison datum has only these named fields available at the local packet boundary.

Apply Definition 90.75 to the carried operator row and the source witness. The theorem constructs the target witness by first applying Definition 90.51 to obtain the pointwise stream s_T from the target endpoint rows in Λ, then reading $T(\ell)$ through the target endpoint row of the same operator ledger. For a requested positive target tolerance, its proof uses the $\mathsf{Real}^{\uparrow}$ bound row for K to select the source tolerance, feeds that tolerance into the displayed source convergence modulus μ, and applies Definition 90.50 to each late pair $s(n), \ell$. The whole late-distance handoff is therefore the finite chain

$$\mu \quad \longrightarrow \quad d_C(s(n), \ell) \quad \longrightarrow \quad K \cdot d_C(s(n), \ell) \quad \longrightarrow \quad d_D(T(s(n)), T(\ell)) \quad \longrightarrow \quad L_T.$$

No step in this chain asks for a target limit search: the endpoint is the named endpoint $T(\ell)$, and the tolerance selector is the one obtained from the displayed K-comparison rows.

Now read the same package through Definition 90.77. Its carrier is the pair of source and target $\mathsf{CompleteMetricLimitWitness}$ rows connected by Λ; its classifier is componentwise hsame transport of the source stream, source limit, target stream, target limit, bound row, and late-distance ledgers; its stability field is precisely the witness-transport theorem; and its exactness field is Definition 90.76. Hence any consumer-facing use that claims to admit or consume the limit comparison must first project to the same source witness, Λ, s_T, $T(\ell)$, μ_T, and L_T, and any replacement of representatives is mediated only by those componentwise classifier rows.

Suppose a stronger reading attempted to add an independent target-limit selector, a host continuity theorem, a quotient completion object, a compactness row, or another coordinate not present in the displayed comparison datum. Such a coordinate is not a field of Definition 90.74, is not produced by Definition 90.75, and is not part of the carrier, classifier, stability, exactness, or ledger fields listed by Definition 90.77. Therefore it cannot participate in an accepted local admission or consumption claim for this comparison. Every accepted use factors back to the displayed finite witness and selector rows, as claimed. □

Definition 90.79 (Singleton Banach standard bridge surface)**.** *The standard comparison object for the singleton Banach certificate is the one-point complete normed vector space $B_\star = \{\star\}$ over the one-point scalar action, with zero vector $\star$, zero norm $\|\star\| = 0^{\mathsf{c}}_{\mathsf{Real}}$, zero metric $d_\star(\star, \star) = 0^{\mathsf{c}}_{\mathsf{Real}}$, and the unique complete-metric limit operation sending every regular Cauchy stream to $\star$.*

The BEDC side is the singleton empty-history carrier of Definition 90.3. Its carrier is A_{e}, its classifier is ρ_{e}, its norm and metric endpoints are the constant RealUp zero endpoint, and its convergence rows are the displayed $\mathsf{CompleteMetricLimitWitness}$ *rows supplied by Definition 90.16. The bridge maps every carried BEDC vector history to $\star$, and maps $\star$ back to* e *with the carrier, norm, metric, and limit ledgers read through the singleton classifier.*

Theorem 90.80 (Singleton Banach bridge soundness)**.** *Every carried BEDC singleton Banach row of Definition 90.3 maps to the unique point $\star$ of $B_\star$. Under this map, the displayed vector is*

classified by ρ_{e}, its norm and induced metric distance are the constant RealUp zero endpoint, and every regular Cauchy stream carries complete-metric convergence data to the constant standard limit $\star$.

Proof. First collapse the carried vector row. By Definition 90.13, every carried Banach vector lies in A_{e} and every pair of carried singleton vectors is ρ_{e}-classified. The bridge therefore has only one possible standard image, namely $\star$.

Next read the metric data. The norm-induced compatibility theorem Definition 90.5 identifies the singleton Banach metric row with the constant RealUp zero endpoint exactly on the same classifier row. Thus the standard norm and standard distance both receive the zero value required by $B_\star$.

Finally transport the convergence ledger. For a regular Cauchy stream, Definition 90.16 supplies a displayed CompleteMetricLimitWitness row whose selected limit, modulus history, and late-distance rows are all singleton rows. Applying the bridge to each component sends that row to the unique complete-metric limit datum of the one-point standard object. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton standard bridge soundness`

Theorem 90.81 (Singleton Banach bridge completeness)**.** *The unique standard point $\star$ maps back to* e *and reconstructs the singleton BEDC Banach surface: carrier A_{e}, classifier ρ_{e}, constant zero norm, constant zero metric, and the singleton* CompleteMetricLimitWitness *family.*

Proof. Unpack the bridge surface of Definition 90.79. Its reverse map sends the only standard point to e, and the BEDC carrier row is precisely the singleton empty-history carrier of Definition 90.3.

The reconstructed completeness data are the singleton data. Apply Definition 90.14: every regular Cauchy stream in the singleton Banach metric has selected limit e, and every distance to that limit is classified by the constant RealUp zero endpoint.

It remains to identify the displayed limit row chosen by the bridge with any other singleton limit presentation. Use Definition 90.41; it gives hsame-equality of singleton Banach limits, and the singleton classifier repackages that equality as ρ_{e}. Hence the reverse image reconstructs the same carrier, norm, metric, and limit witness surface. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachUp StdBridge`

Theorem 90.82 (Singleton Banach bridge roundtrip)**.** *For the bridge of Definition 90.79, the composite BEDC-to-standard-to-BEDC is ρ_{e} on carried singleton histories, and the composite standard-to-BEDC-to-standard is the identity on the one-point complete normed vector space $B_\star$.*

Proof. For the BEDC composite, start with a carried singleton Banach history. The soundness theorem Definition 90.80 sends it to the unique standard point $\star$, and the completeness theorem Definition 90.81 sends $\star$ back to e with the singleton carrier and zero ledgers.

The classifier comparison is exactly the singleton collapse. By Definition 90.13, every carried source history is ρ_{e}-classified with e, so the BEDC composite acts as ρ_{e} on carried histories.

For the standard composite, there is only one point of $B_\star$. The reverse map selects e, and soundness maps that BEDC point back to $\star$. Since $\star$ is the unique standard point, the composite is the identity on the one-point complete normed vector space. □

Lean4 ✓ `BEDC.Derived.BanachUp.BanachSingleton standard bridge roundtrip`

Lemma 90.83 (Singleton Banach standard bridge determinacy)**.** *For the bridge surface of Definition 90.79, every consumer-facing route, transport, readback, or limit handoff is determined by the displayed singleton rows. The BEDC-to-standard route factors through the singleton carrier A_{e} and classifier ρ_{e}, the standard-to-BEDC readback factors through the selected endpoint* e*, the norm and metric transports factor through the constant RealUp zero endpoint, and every complete-metric handoff factors through the displayed* CompleteMetricLimitWitness *rows of Definition 90.16. No unlisted bridge coordinate supplies an additional carrier point, distance value, classifier, or limit datum.*

Proof. Unpack the bridge surface of Definition 90.79. Its standard object is the one-point complete normed vector space $B_\star = \{\star\}$, and its BEDC object is the singleton empty-history carrier of Definition 90.3. Hence a forward route has only one standard codomain point available. By Definition 90.80, every carried BEDC singleton Banach row maps to that point $\star$, and the displayed source row is read through A_{e} and ρ_{e}. Thus any accepted forward use factors through the same singleton carrier and classifier rows; there is no second standard endpoint through which it could factor.

The readback side is equally fixed by the displayed bridge surface. The only standard point $\star$ is sent back to e, and the reverse ledger is stated there with the carrier, norm, metric, and limit data read through the singleton classifier. Since the reverse map has no other standard input and names no other BEDC endpoint, any consumer readback is the selected row e together with its ρ_{e} carrier comparison.

For norm and metric transport, soundness identifies the norm of the displayed vector and every induced metric distance with the constant RealUp zero endpoint. This is precisely the local norm-induced metric route supplied by Definition 90.5: the singleton metric row is accepted on the same ρ_{e} classifier row and is classified by the constant RealUp zero endpoint. Therefore a consumer cannot choose an anonymous distance coordinate or a second metric classifier; every distance read is the same zero row already present in the singleton Banach surface.

It remains to check complete-metric handoff. Soundness sends every regular Cauchy stream to the constant standard limit $\star$, while Definition 90.16 supplies the displayed CompleteMetricLimitWitness row whose selected limit, modulus history, and late-distance ledgers are singleton rows. Applying the bridge componentwise therefore leaves only that listed limit package: selected limit e on the BEDC side, selected limit $\star$ on the standard side, empty singleton modulus, and late-distance rows classified by the constant RealUp zero endpoint. These are exactly the rows exposed in Definition 90.79 and used by Definition 90.80. Since the surface lists no further carrier, transport, metric, or limit coordinate, any alleged unlisted bridge coordinate has no displayed row from which a consumer-facing use could be read. □

Corollary 90.84 (Singleton Banach standard bridge consumer determinacy)**.** *Fix the bridge surface of Definition 90.79. Any accepted consumer use of the singleton Banach standard bridge has the same exposed finite decision as the chapter rows: a forward use chooses the singleton standard route, a readback use chooses the singleton BEDC endpoint route, a norm or metric use chooses the constant* $\mathsf{Real}^{\uparrow}$ *zero exactness row, and a complete-metric use chooses the displayed singleton limit-witness package. In particular, the bridge exports no hidden coordinate, no alternate normal form, and no duplicate classifier source for the singleton Banach object.*

Proof. The bridge surface in Definition 90.79 names all rows available to a consumer: the one-point standard comparison object, the singleton BEDC carrier and classifier, the constant $\mathsf{Real}^{\uparrow}$ zero norm and metric endpoints, and the displayed complete-metric limit-witness rows. Apply

Definition 90.83 to this fixed surface. It states that every consumer-facing route, transport, readback, or limit handoff factors through exactly those displayed singleton rows and that no unlisted bridge coordinate supplies an additional carrier point, distance value, classifier, or limit datum.

Thus the forward route cannot select anything other than the singleton standard point named by the surface, and the readback route cannot select anything other than the singleton BEDC endpoint named by the surface. The norm and metric transports cannot introduce a second exactness decision, because the determinacy lemma forces them through the same constant $\mathsf{Real}^{\uparrow}$ zero row already exposed by the bridge definition. Likewise a complete-metric handoff cannot select a detached modulus, tail, late-distance, or limit coordinate: the only accepted package is the displayed singleton limit-witness package cited by the surface and by the determinacy lemma.

These four cases exhaust the consumer-facing uses listed in the determinacy lemma. Therefore, once the bridge surface rows are fixed, every accepted consumer use picks the same finite route, readback normal form, or exactness decision named by the singleton Banach chapter, and there is no further hidden coordinate or duplicate classifier source. □

Chapter 91

A Concrete Naming Certificate for $\mathsf{Hilbert}^{\uparrow}$

The Hilbert-space interface $\mathsf{Hilbert}^{\uparrow}$ packages a Banach space whose norm is induced by an inner product. It depends on $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ (chapter 86) and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90), and is the substrate for the spectral and orthogonal-decomposition certificates of functional analysis.

91.1 Carrier definition

Definition 91.1 (Compatible inner-product Banach carrier)**.** *A compatible inner-product Banach carrier for* $\mathsf{Hilbert}^{\uparrow}$ *is a history predicate* $X : \mathsf{BHist} \to \mathsf{Prop}$ *with point classifier* $\sim_X$ *together with three displayed rows over the same carried histories. The first row is a concrete* $\mathsf{Banach}^{\uparrow}$ *candidate of Definition 90.1. The second row is an* $\mathsf{InnerProduct}^{\uparrow}$ *carrier on* X*, including the vector-space structure, sesquilinear form, conjugate symmetry, linearity, positivity, and inner-product classifier transport. The third row is a norm-induction ledger recording that the Banach norm and Banach metric rows are exactly the rows displayed by the inner-product norm:*

$$\|x\|_{\mathsf{Banach}^{\uparrow}} = \sqrt{\langle x, x\rangle_{\mathsf{InnerProduct}^{\uparrow}}}$$

and

$$d_{\mathsf{Banach}^{\uparrow}}(x, y) = \|x - y\|_{\mathsf{Banach}^{\uparrow}}$$

for carried histories $x, y : X$*.*

The compatibility requirement is literal row sharing, not an isomorphism postulate: the Banach carrier, the inner-product carrier, and the norm-induced metric ledger expose the same point carrier X *and the same classifier* $\sim_X$*. Every accepted Hilbert row therefore carries a visible Banach surface, a visible inner-product surface, and a ledger explaining why the Banach metric is the metric induced by the inner-product norm.*

Definition 91.2 (Hilbert singleton empty-history carrier)**.** *The concrete* $\mathsf{Hilbert}^{\uparrow}$ *carrier is the singleton empty-history Banach carrier of Definition 90.3, read over the singleton empty-history vector space of Definition 47.2. Thus the vector carrier is* A_{e}*, the vector classifier is* ρ_{e}*, the normed carrier is the singleton* $\mathsf{Banach}^{\uparrow}$ *carrier, and the norm is* ν_{e}*.*

The inner-product row is the constant-zero RealUp endpoint

$$\langle m, n\rangle_{\mathsf{e}} := 0^{c}_{\mathsf{Real}}$$

for all displayed vector histories m, n : BHist. *Its scalar-side classifier is* RealConstantHistoryClassifier. *The induced norm row is read by the same constant endpoint:*

$$\|m\|_{\langle,\rangle_{\mathsf{e}}} := 0^{c}_{\mathsf{Real}}.$$

Hence the induced norm is the singleton $\mathsf{Norm}^{\uparrow}$ *norm* ν_{e} *from Definition 89.1. The ledger records the singleton vector-space source, the Banach singleton carrier, and the RatUp zero witness sealed as the RealUp constant endpoint.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonEmptyHistoryCarrier admission`

91.2 The certificate

Theorem 91.3 (Hilbert induced norm Banach compatibility)**.** *Every accepted compatible inner-product Banach carrier exposes the same point carrier and classifier to its* $\mathsf{Banach}^{\uparrow}$ *and* $\mathsf{InnerProduct}^{\uparrow}$ *surfaces. Moreover,* hsame *transport preserves the norm-induced Banach metric witness: if* $x \sim_X x'$ *and* $y \sim_X y'$*, then a displayed* $\mathsf{BanachMetricWitness}_X(x, y, d)$ *row transports to the corresponding* $\mathsf{BanachMetricWitness}_X(x', y', d')$ *row classified by the same inner-product norm ledger.*

Proof. Unpack the compatible carrier of Definition 91.1. The Banach row and inner-product row have the same endpoint predicate X and classifier $\sim_X$ by definition, so no change of carrier or classifier is introduced.

Apply Definition 90.5 to read the Banach metric through the displayed norm row. The Hilbert ledger identifies that norm row with the inner-product norm row, so the metric witness remains the norm-induced Banach witness after this reading.

Finally use the componentwise hsame transport rows supplied by the Banach surface and the inner-product surface. They move the vector representative, difference row, inner-product norm row, and RealUp distance endpoint together, and repack the transported rows as $\mathsf{BanachMetricWitness}_X(x', y', d')$ over the same classifier $\sim_X$. □

Theorem 91.4 (Hilbert complete inner-product limit stability)**.** *Let* H *be a compatible inner-product Banach carrier. For any regular Cauchy stream* s *over the Banach metric, any displayed* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$ *supplied by the Banach row, and any proof that the stream observations remain in the inner-product carrier, there is a Hilbert limit witness with the same selected limit* ℓ*, the same limit name* L*, and the same classifier* $\sim_X$*.*

Proof. Start from the Banach row contained in H. Its completeness field supplies the displayed $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$ and its carrier endpoint data through Definition 90.1.

The observation-stability hypothesis says that the same stream is carried by the inner-product surface of H. Use the inner-product carrier stability row to transport all observation representatives through $\sim_X$ while retaining the displayed inner-product norm rows named by the Hilbert ledger.

Apply Definition 91.3 to identify the transported Banach distance witnesses with the inner-product norm-induced ones. The selected limit endpoint ℓ and limit name L are therefore not changed; they are merely read as Hilbert rows over the same carrier, classifier, and visible metric ledger. □

Theorem 91.5 (Hilbert NameCert obligation surface)**.** *The compatible carrier of Definition 91.1, its induced classifier* $\sim_X$*, the norm-metric compatibility of Definition 91.3, the complete-limit stability of Definition 91.4, and the visible ledger joining the Banach and inner-product rows assemble*

the paper-side $\mathsf{NameCert}_{\mathsf{Hilbert}\uparrow}$ *obligation surface. This surface records the Hilbert carrier, classifier, norm-induced Banach metric, complete-limit stability, and ledger provenance; it does not assert a standard Hilbert-space bridge.*

Proof. Use the compatible carrier definition for the public carrier field. It already contains the Banach row, the inner-product row, and the row-sharing ledger binding both surfaces to the same X and $\sim_X$.

Use Definition 91.3 for the norm-metric field. It identifies the Banach metric witness as the metric induced by the inner-product norm and records that hsame transport preserves that witness componentwise.

Use Definition 91.4 for the limit-stability field. It consumes the Banach $\mathsf{CompleteMetricLimitWitness}$ row together with the inner-product observation-stability row and returns the same selected limit as a Hilbert limit witness.

The remaining fields are inherited from the source rows: the Banach surface provides the complete normed metric obligations, the inner-product surface provides the sesquilinear, symmetry, positivity, and classifier-stability obligations, and the ledger records exactly which displayed rows are shared. Thus the paper-side $\mathsf{NameCert}_{\mathsf{Hilbert}\uparrow}$ obligation surface is assembled without adding any standard bridge claim. □

Theorem 91.6 (Hilbert singleton inner-product norm compatibility)**.** *For the carrier of Definition 91.2, the norm induced by* $\langle -, - \rangle_{\mathsf{e}}$ *agrees with the singleton norm* ν_{e}*:*

$$\mathsf{RealConstantHistoryClassifier}(\|m\|_{\langle,\rangle_{\mathsf{e}}}, \nu_{\mathsf{e}}(m))$$

for every carried $m : \mathsf{BHist}$*. Moreover positive definiteness of the inner-product row collapses to the singleton vector classifier:*

$$\mathsf{RealConstantHistoryClassifier}(\langle m, m\rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}}) \iff \rho_{\mathsf{e}}(m, \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton inner product norm compatibility`

Proof. The induced norm and ν_{e} are both the constant endpoint 0^{c}_{Real}. The carrier and classifier rows of Definition 89.2 give the RealUp constant-history classifier for this endpoint, so the two norm presentations are classified together.

For positive definiteness in the forward direction, the inner-product value $\langle m, m\rangle_{\mathsf{e}}$ is again 0^{c}_{Real}. Apply Definition 47.4 to the carried vector endpoint. It identifies m with e under ρ_{e}.

For the reverse direction, assume $\rho_{\mathsf{e}}(m, \mathsf{e})$. The inner-product row does not inspect the representative; it returns 0^{c}_{Real} for every pair. The zero-exactness boundary of Definition 89.4, together with the constant-real classifier already supplied by the singleton norm carrier, repackages the row as $\mathsf{RealConstantHistoryClassifier}(\langle m, m\rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}})$. □

Theorem 91.7 (Hilbert singleton complete normed certificate)**.** *The normed part of Definition 91.2 is complete. Every regular Cauchy stream carried by* A_{e} *has selected limit* e*, every late distance-to-limit row is classified with* 0^{c}_{Real}*, and any two selected Banach limit witnesses are classified by* ρ_{e}*.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton complete normed certificate`

Proof. Use the singleton Banach carrier of Definition 90.3. Its metric is the constant RealUp zero endpoint and its limit-selection rule sends every certified regular Cauchy stream to e.

The complete-metric laws are supplied by Definition 128.3. They give the carried singleton limit, the empty modulus, and the late-observation ledgers for the constant zero metric.

Finally apply Definition 90.41 to compare any two Banach limit witnesses for the same stream. In the singleton vector carrier, Definition 47.4 reads that comparison as the classifier row ρ_{e}, so the complete normed certificate is the Banach singleton certificate read through the Hilbert carrier. □

Theorem 91.8 (Hilbert singleton classifier source boundary)**.** *The carrier of Definition 91.2 exposes exactly the singleton vector-space carrier A_{e} and classifier ρ_{e}. The Banach row, the singleton* $\mathsf{Norm}^{\uparrow}$ *row, the inner-product row, and the Hilbert row all read their vector endpoints from the same* BHist *source; no carrier endpoint and no vector classifier outside the singleton empty-history source is used.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton classifier source boundary`

Proof. Unfold the Hilbert singleton carrier of Definition 91.2. Its first displayed source is the singleton vector-space instance of Definition 47.2, so the visible vector carrier is A_{e} and the visible classifier is ρ_{e}.

The normed row is the singleton norm instance of Definition 89.1, and the Banach row is the singleton Banach instance of Definition 90.3. Both rows are stated over the same singleton vector carrier and therefore project the same A_{e} and ρ_{e} fields.

The Hilbert row adds only the constant inner-product display over those carried histories. Since that display takes arguments $m, n : \mathsf{BHist}$ from the already fixed singleton carrier, the Hilbert surface has the same endpoint source as the vector-space, norm, and Banach surfaces. □

Theorem 91.9 (Hilbert singleton constant inner-product transport)**.** *For the carrier of Definition 91.2, if $\rho_{\mathsf{e}}(m, m')$ and $\rho_{\mathsf{e}}(n, n')$, then the corresponding inner-product endpoints are related by the constant-real classifier:*

$$\mathsf{RealConstantHistoryClassifier}(\langle m, n\rangle_{\mathsf{e}}, \langle m', n'\rangle_{\mathsf{e}}).$$

The induced norm endpoints are transported along the same scalar classifier, so

$$\mathsf{RealConstantHistoryClassifier}(\|m\|_{\langle,\rangle_{\mathsf{e}}}, \|m'\|_{\langle,\rangle_{\mathsf{e}}})$$

holds over the singleton vector classifier.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton constant inner product transport`

Proof. The inner-product operation in Definition 91.2 is constant: every ordered pair of carried histories is sent to the endpoint $0^{\mathsf{c}}_{\mathsf{Real}}$. Thus the two endpoints $\langle m, n\rangle_{\mathsf{e}}$ and $\langle m', n'\rangle_{\mathsf{e}}$ are the same constant RealUp zero endpoint.

The singleton norm carrier supplies the scalar-side classifier $\mathsf{RealConstantHistoryClassifier}$ for this constant endpoint. Applying that classifier to the two constant zero presentations gives the displayed inner-product transport row.

For the induced norm, use Definition 91.6. It reads the induced norm as the singleton norm ν_{e}, whose endpoint is again the same constant RealUp zero row. The norm transport conclusion is therefore the same constant-real classifier row read through the induced norm display. □

Theorem 91.10 (Hilbert singleton ledger exhaustion)**.** *Every row consumed by the singleton* $\mathsf{Hilbert}^{\uparrow}$ *package is one of the singleton vector-space source row, the singleton* $\mathsf{Norm}^{\uparrow}$ *row, the singleton Banach completion row, or the constant RealUp zero inner-product row. The package introduces no* $\mathsf{StandardBridge}$ *row, no second vector carrier, and no carrier source beyond the* BHist *singleton ledger recorded in Definition 91.2.*

Proof. The carrier and classifier accounting is fixed by Definition 91.8. It leaves only the singleton vector-space source A_{e}, ρ_{e}, and the BHist endpoint source available to the Hilbert package.

The scalar stability accounting is fixed by Definition 91.9. It says that both the inner-product endpoint and the induced norm endpoint are transported by $\mathsf{RealConstantHistoryClassifier}$ at the constant RealUp zero row.

The completeness accounting is exactly Definition 91.7, which reads the singleton Banach completion row through the Hilbert carrier. Together with the ledger sentence in Definition 91.2, these rows exhaust the package: vector source, norm source, Banach completion source, and constant inner-product zero source are all named, and no standard bridge or outside carrier source remains. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton ledger exhaustion`

Theorem 91.11 (Hilbert NameCert obligation package)**.** *The carrier of Definition 91.2, together with the classifier-source boundary Definition 91.8, the constant-inner-product transport row Definition 91.9, the ledger exhaustion row Definition 91.10, and the certificate rows Definition 91.6 and Definition 91.7, assembles the paper-side* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *obligation package. Its carrier is* A_{e}*, its classifier is* ρ_{e}*, its inner-product row is* $\langle -, - \rangle_{\mathsf{e}}$*, its induced norm is* ν_{e}*, its completeness field is the singleton Banach limit rule, its ledger is the union of the singleton vector-space, norm, inner-product zero-row, and Banach-completion ledgers, and its bridge status is none.*

Proof. The carrier and classifier fields are supplied by Definition 91.8. The Banach source is exactly Definition 90.3, so the normed carrier and metric rows are already present before the Hilbert row is attached.

The inner-product compatibility field is Definition 91.6. It identifies the induced norm with ν_{e} and reflects positive definiteness back to the singleton classifier against e.

The transport field is Definition 91.9. It records that replacing either input by a ρ_{e}-classified endpoint leaves the inner-product endpoint and the induced norm endpoint inside the same $\mathsf{RealConstantHistoryClassifier}$ row.

The completeness field is Definition 91.7. It supplies the selected limit, the convergence ledgers, and classifier uniqueness for regular Cauchy streams in the norm induced by the inner product.

Definition 91.10 supplies the ledger field. Combining these rows gives the local $\mathsf{Hilbert}^{\uparrow}$ naming certificate: the traditional target is the one-point Hilbert space, the BEDC source is BHist, the carrier and classifier are A_{e} and ρ_{e}, the stability rows are constant-zero classifier transport, the exactness row is positive definiteness against e, the ledger records the shared singleton provenance, and no standard bridge is claimed here. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertUp StdBridge`

Theorem 91.12 (Hilbert singleton source compatibility)**.** *For the carrier of Definition 91.2, the singleton* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{Banach}^{\uparrow}$ *sources read one carried predicate and one classifier:*

$$A_{\mathsf{Hilbert},\mathsf{e}} = A_{\mathsf{e}} \quad \textit{and} \quad \rho_{\mathsf{Hilbert},\mathsf{e}} = \rho_{\mathsf{e}}.$$

The norm row is the singleton norm endpoint ν_{e}*, the Banach metric row is the norm-induced constant RealUp zero distance, and the inner-product row returns* $0^{\mathsf{c}}_{\mathsf{Real}}$ *under* $\mathsf{RealConstantHistoryClassifier}$ *for every displayed pair* $m, n : \mathsf{BHist}$*.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton source compatibility`

Proof. Unfold the Hilbert carrier of Definition 91.2. Its vector source is the singleton empty-history vector space, so the displayed carrier and classifier are A_{e} and ρ_{e} before the norm or inner-product rows are attached.

For the norm source, apply Definition 89.2. It reads every carried norm endpoint as ν_{e} and classifies all displayed singleton norm values through $\mathsf{RealConstantHistoryClassifier}$ at $0^{\mathsf{c}}_{\mathsf{Real}}$.

For the Banach source, the carrier is Definition 90.3. Its metric is induced from the same singleton norm and therefore exposes the same carried predicate, the same classifier, and the same constant RealUp zero endpoint surface.

The inner-product row is defined separately but uses no additional carrier data. It returns $0^{\mathsf{c}}_{\mathsf{Real}}$ for every pair $m, n : \mathsf{BHist}$, and Definition 47.4 supplies the underlying vector endpoint collapse used by all three sources. □

Theorem 91.13 (Hilbert singleton endpoint readback)**.** *Every norm endpoint, inner-product self-value, and Banach distance endpoint exposed by the Hilbert singleton package is classified with* $0^{\mathsf{c}}_{\mathsf{Real}}$ *by* $\mathsf{RealConstantHistoryClassifier}$*. Concretely, for carried histories* $m, n : \mathsf{BHist}$*,*

$$\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(m), 0^{\mathsf{c}}_{\mathsf{Real}}), \quad \mathsf{RealConstantHistoryClassifier}(\langle m, m\rangle_{\mathsf{e}}, 0^{\mathsf{c}}_{\mathsf{Real}}), \quad \mathsf{RealConstantHistoryCla}$$

are the public endpoint rows of the package.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton endpoint readback`

Proof. The norm endpoint row is the zero-exactness surface of Definition 89.4. On the singleton carrier, the only accepted vector endpoint is classified with e, so the norm value is read as $0^{\mathsf{c}}_{\mathsf{Real}}$ by the constant-real classifier.

The inner-product self-value is handled by Definition 91.6. It identifies the induced norm with ν_{e} and states that the diagonal inner-product row is classified with the constant RealUp zero endpoint exactly at the singleton vector classifier boundary.

The Banach distance endpoint comes from Definition 90.3. Its metric row is the norm-induced singleton distance, and Definition 90.13 reads every displayed distance as the same constant RealUp zero endpoint. □

Definition 91.14 (Hilbert singleton projection witness)**.** *For the carrier of Definition 91.2, a Hilbert singleton projection witness from a carried endpoint* m *to a projected endpoint* p *is the concrete tuple*

$$(A_{\mathsf{e}}(m), A_{\mathsf{e}}(p), \rho_{\mathsf{e}}(p, \mathsf{e}), \mathsf{RealConstantHistoryClassifier}(\|m-p\|_{\langle,\rangle_{\mathsf{e}}}, 0^{\mathsf{c}}_{\mathsf{Real}}), \mathsf{RealConstantHistoryClassifier}(\langle m-p, p\rangle_{\mathsf{e}}, 0^{\mathsf{c}}_{\mathsf{Real}}$$

The first two entries record that both endpoints are carried by the singleton empty-history carrier, the third entry fixes the concrete projected endpoint ledger, the fourth entry is the constant RealUp zero readback for the induced norm, and the fifth entry is the constant inner-product row against the projected endpoint.

Lean4 (definition) `BEDC.Derived.HilbertUp.HilbertSingletonProjectionWitness`

Theorem 91.15 (Hilbert singleton projection existence)**.** *Every carried endpoint* m *of Definition 91.2 admits a Hilbert singleton projection witness with projected endpoint* e*.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection existence`

Proof. First read the Hilbert carrier through Definition 91.12. The endpoint m is carried by A_{e}, and the projected endpoint e is carried by the same singleton source and classified with itself by ρ_{e}.

Next use Definition 91.13. It identifies the induced norm endpoint exposed by the Hilbert singleton package with the constant RealUp zero endpoint under $\mathsf{RealConstantHistoryClassifier}$.

Finally apply Definition 91.9. The inner-product row is constant at 0^{c}_{Real}, so $\langle m - \mathsf{e}, \mathsf{e}\rangle_{\mathsf{e}}$ is classified with the same constant RealUp zero endpoint. These rows are exactly the tuple required by Definition 91.14. □

Theorem 91.16 (Hilbert singleton projection idempotence)**.** *If p is the projected endpoint in a Hilbert singleton projection witness for m, then the Hilbert singleton projection witness for p has projected endpoint e, and this endpoint is classified with p by ρ_{e}.*

Proof. Unpack the witness of Definition 91.14. One of its displayed rows is $\rho_{\mathsf{e}}(p, \mathsf{e})$, so the projected endpoint is already fixed at the singleton empty-history representative.

Apply Definition 47.4 to the carried endpoint p. The singleton carrier has no second vector endpoint, so the endpoint used by the projection witness of p is again e.

Combining the displayed row $\rho_{\mathsf{e}}(p, \mathsf{e})$ with this singleton collapse gives the idempotence claim: projecting the projected endpoint does not leave the ρ_{e}-class of p. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection witness idempotent`

Theorem 91.17 (Hilbert singleton projection orthogonal zero)**.** *Let p be the projected endpoint in a Hilbert singleton projection witness for m. For every carried endpoint n of Definition 91.2, the residual inner-product row is classified with 0^{c}_{Real}:*

$$\mathsf{RealConstantHistoryClassifier}(\langle m - p, n\rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}}).$$

Proof. The inner-product operation in Definition 91.2 is constant at 0^{c}_{Real} for every displayed pair of carried histories. Thus the residual row $\langle m - p, n\rangle_{\mathsf{e}}$ does not depend on the representatives m, p, or n.

Use Definition 91.13 to read this displayed inner-product endpoint through the Hilbert singleton package. The scalar classifier is $\mathsf{RealConstantHistoryClassifier}$, and its endpoint is the constant RealUp zero endpoint.

The witness assumption supplies the carried projected endpoint and the $\rho_{\mathsf{e}}(p, \mathsf{e})$ row from Definition 91.14. Hence the residual inner-product row is the same constant-zero row after the projection ledger is unpacked. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection orthogonal zero`

Theorem 91.18 (Hilbert singleton projection uniqueness)**.** *Let s be an accepted regular Cauchy stream in the Hilbert singleton carrier. Any two Hilbert singleton limit or projection witnesses for s are identified by ρ_{e}. Consequently they project to the same Banach limit, the same singleton $\mathsf{Norm}^{\uparrow}$ zero endpoint, and the same inner-product zero row.*

Proof. First read each Hilbert limit witness through the complete normed field of Definition 91.7. This places the limit witnesses on the Banach singleton surface for the same accepted regular Cauchy stream.

Apply Definition 90.41 to the two Banach-surface rows. It identifies their displayed limits first by hsame and then by the singleton vector classifier ρ_{e}.

For a projection witness, use Definition 91.14 to read the projected endpoint as a carried singleton endpoint with displayed row $\rho_{\mathsf{e}}(p, \mathsf{e})$. Then apply Definition 91.16. It says that the projection branch remains in the same ρ_{e}-class after projecting again.

Finally use Definition 91.12. The Hilbert carrier, the normed carrier, and the Banach carrier all expose the same classifier surface, so the ρ_{e} comparison just obtained is simultaneously the Banach limit comparison, the NormUp zero-endpoint comparison, and the inner-product row comparison. The endpoint readback is the constant RealUp zero row supplied by Definition 91.13. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection uniqueness`

Theorem 91.19 (Hilbert singleton limit inner-product carrier)**.** *Let s be an accepted regular Cauchy stream in the Hilbert singleton carrier, and let ℓ : BHist be the selected complete-limit witness exposed by the Hilbert singleton package. Then ℓ is carried by A_{e}, the classifier ρ_{e} classifies ℓ with e, and the inner-product row of the limit is the constant RealUp zero endpoint:*

$$\mathsf{RealConstantHistoryClassifier}(\langle \ell, \ell \rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}}).$$

Proof. First use the complete normed field of Definition 91.7. It reads the accepted Hilbert Cauchy stream as the Banach singleton completion problem and places its selected limit witness on the Banach limit row.

Apply Definition 90.41 to the selected Banach limit row. Since the Banach singleton carrier and classifier are A_{e} and ρ_{e}, the selected witness is classified with e by ρ_{e}.

Finally transport the same witness back through Definition 91.12. The Hilbert and Banach surfaces share the carrier and classifier, while Definition 91.13 reads the diagonal inner-product value as the constant RealUp zero endpoint. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton limit inner product carrier`

Theorem 91.20 (Hilbert singleton norm-induced metric classifier)**.** *For carried histories m, n : BHist on the Hilbert singleton carrier, the Banach metric endpoint $d_{\mathsf{e}}(m, n)$, the induced-norm endpoint $\|m - n\|_{\langle,\rangle_{\mathsf{e}}}$, and the constant endpoint 0^{c}_{Real} are classified by $\mathsf{RealConstantHistoryClassifier}$. Consequently the metric classifier on the singleton Hilbert carrier is exactly the norm-induced classifier obtained from the constant inner product.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton norm induced metric classifier`

Proof. Begin with Definition 91.6. It identifies the norm induced by the constant inner product with the singleton norm ν_{e} on every carried history.

Next apply Definition 91.3. The Banach distance row attached to the singleton Hilbert package is the distance induced by that same norm, so $d_{\mathsf{e}}(m, n)$ and $\|m - n\|_{\langle,\rangle_{\mathsf{e}}}$ name the same endpoint surface for the classifier.

It remains to read the endpoint. By Definition 91.13, the norm endpoint, the inner-product endpoint, and the Banach metric endpoint are all classified by $\mathsf{RealConstantHistoryClassifier}$ at 0^{c}_{Real}. □

Theorem 91.21 (Hilbert singleton semantic NameCert surface)**.** *The singleton carrier A_{e}, classifier ρ_{e}, constant inner-product row, induced norm row, complete-limit row, norm-induced metric classifier, projection uniqueness row, and ledger exhaustion row together form the concrete paper-side SemanticNameCert surface for $\mathsf{Hilbert}^{\uparrow}$. This surface does not assert a standard bridge; it records the singleton Hilbert certificate inside the BEDC BHist and NameCert interfaces.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection semantic name certificate`

Proof. Start from Definition 91.11. It assembles the carrier, classifier, vector source, scalar source, norm source, Banach source, and Hilbert inner-product source into the Hilbert singleton obligation package.

The complete-limit carrier row is supplied by Definition 91.19. The metric row is the norm-induced classifier of Definition 91.20.

Projection determinacy is exactly Definition 91.18, and the remaining certificate ledger is exhausted by Definition 91.10. These rows give the paper-side SemanticNameCert surface for $\mathsf{Hilbert}^{\uparrow}$ while leaving any standard bridge claim outside the singleton certificate. □

Theorem 91.22 (Hilbert singleton universal orthogonality)**.** *For any carried histories $m, n : \mathsf{BHist}$ in the singleton Hilbert carrier, the inner-product endpoint is the constant RealUp zero row:*

$$\mathsf{RealConstantHistoryClassifier}(\langle m, n\rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}}).$$

Thus every carried pair is orthogonal under the singleton Hilbert classifier.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton universal orthogonality`

Proof. By Definition 91.2, every carried history is the public singleton endpoint. Hence the displayed pair has no separate source data beyond the unique e endpoint.

Apply Definition 91.9. The constant inner-product row transports every accepted pair to the same $\mathsf{RealConstantHistoryClassifier}$ value at 0^{c}_{Real}.

Finally, Definition 91.13 identifies that endpoint with the public Hilbert readback row. This is precisely the singleton orthogonality classifier for the carried pair. □

Theorem 91.23 (Hilbert singleton parallelogram collapse)**.** *For carried histories $m, n : \mathsf{BHist}$ in the singleton Hilbert carrier, the parallelogram law surface over the singleton vector-space operations collapses to the constant RealUp zero endpoint. The four norm rows*

$$\nu_{\mathsf{e}}(m+n), \quad \nu_{\mathsf{e}}(m-n), \quad \nu_{\mathsf{e}}(m), \quad \nu_{\mathsf{e}}(n)$$

are all classified with 0^{c}_{Real} by $\mathsf{RealConstantHistoryClassifier}$, so both sides of the Hilbert parallelogram identity are read through the same constant-real row.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton parallelogram collapse`

Proof. First use Definition 91.12. It places the Hilbert carrier, the singleton vector carrier, and the singleton norm carrier on one shared ρ_{e} source surface.

The vector operations are collapsed by Definition 47.4. Therefore $m+n$, $m-n$, m, and n all expose the unique carried endpoint of the singleton vector space.

Apply Definition 89.3 to those four carried endpoints. Each displayed norm is classified with the constant RealUp zero endpoint, and Definition 91.13 reads these norm rows inside the Hilbert package. The two parallelogram sides therefore have the same classifier row. □

Theorem 91.24 (Hilbert singleton projection boundary)**.** *The public projection boundary of the singleton Hilbert package selects e for every carried input pair. The selected endpoint is classified by ρ_{e}, and the projection readback agrees simultaneously with the Banach limit row, the singleton $\mathsf{Norm}^{\uparrow}$ row, and the constant inner-product zero row.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection boundary`

Proof. Apply Definition 91.18 to the public projection witnesses for the carried inputs. Since the Hilbert carrier has only the singleton endpoint, uniqueness identifies every selected endpoint by ρ_{e}.

The complete normed certificate in Definition 91.7 places that selection on the Banach singleton surface. Through Definition 91.12, the same source comparison is also the singleton $\mathsf{Norm}^{\uparrow}$ comparison.

It remains to read the Hilbert geometry row. By Definition 91.22, the inner-product component of the selected pair is the constant RealUp zero row. Hence the projection boundary agrees with the Banach limit row, the NormUp zero row, and the orthogonality row at the same selected endpoint. □

Definition 91.25 (Hilbert singleton orthogonal projection row)**.** *The concrete orthogonal-projection row over the Hilbert singleton carrier is the map*

$$\pi_{\mathsf{e}}(h) := \mathsf{e}.$$

Its ledger surface is the carrier A_{e} of Definition 91.2, the classifier ρ_{e}, the singleton norm row ν_{e}, the constant zero inner-product row $\langle -, - \rangle_{\mathsf{e}}$, and the Banach singleton completion row of Definition 90.3. The row therefore selects the unique carried endpoint e *and records every norm, distance, and inner-product readback through* $\mathsf{RealConstantHistoryClassifier}$ *at* 0^{c}_{Real}.

Theorem 91.26 (Hilbert singleton projection carried endpoint)**.** *For every carried $h : \mathsf{BHist}$ in the Hilbert singleton carrier, the projection endpoint $\pi_{\mathsf{e}}(h)$ is carried, the classifier row $\rho_{\mathsf{e}}(h, \pi_{\mathsf{e}}(h))$ holds, and the norm, distance, and inner-product endpoints exposed by this projection row are classified with 0^{c}_{Real}. Concretely,*

$$\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(\pi_{\mathsf{e}}(h)), 0^{c}_{\mathsf{Real}}), \quad \mathsf{RealConstantHistoryClassifier}(d_{\mathsf{e}}(h, \pi_{\mathsf{e}}(h)), 0^{c}_{\mathsf{Real}}), \quad \mathsf{RealConstantHis}$$

are the endpoint rows carried by $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection carried endpoint`
Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton projection carried endpoint`

Proof. Unfold Definition 91.25. The projection endpoint is definitionally e, and Definition 91.2 identifies e as the carried endpoint of the Hilbert singleton carrier.

The classifier claim follows from Definition 91.12. The Hilbert carrier, NormUp carrier, and Banach carrier expose the same singleton classifier ρ_{e}, so every carried h is compared with the selected endpoint e by the same row.

For the endpoint readbacks, apply Definition 91.13. It classifies the singleton norm endpoint, the Banach distance endpoint, and the inner-product row through $\mathsf{RealConstantHistoryClassifier}$ at 0^{c}_{Real}. Since $\pi_{\mathsf{e}}(h)$ is e, the residual row is one of the displayed singleton inner-product rows. □

Theorem 91.27 (Hilbert singleton projection idempotence)**.** *For every carried $h : \mathsf{BHist}$ in the Hilbert singleton carrier,*

$$\rho_{\mathsf{e}}(\pi_{\mathsf{e}}(\pi_{\mathsf{e}}(h)), \pi_{\mathsf{e}}(h))$$

holds. Moreover the residual inner-product row

$$\langle \pi_{\mathsf{e}}(h) - \pi_{\mathsf{e}}(\pi_{\mathsf{e}}(h)), \pi_{\mathsf{e}}(h) \rangle_{\mathsf{e}}$$

is classified with 0^{c}_{Real} by $\mathsf{RealConstantHistoryClassifier}$.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection idempotence`

Proof. Unfold the projection row twice. Both $\pi_{\mathsf{e}}(h)$ and $\pi_{\mathsf{e}}(\pi_{\mathsf{e}}(h))$ are the selected endpoint e.

The classifier statement is therefore the singleton reflexive comparison inside ρ_{e}, using the common carrier and classifier surface of Definition 91.12.

The residual inner-product expression has only singleton endpoints. Apply Definition 91.13, which reads every displayed inner-product endpoint in this package as the constant RealUp zero endpoint. □

Theorem 91.28 (Hilbert singleton projection-residual zero package)**.** *For every carried $h : \mathsf{BHist}$ in the Hilbert singleton carrier, set $p := \pi_{\mathsf{e}}(h)$ and $p' := \pi_{\mathsf{e}}(p)$. Then the projected endpoints are classified by*

$$\rho_{\mathsf{e}}(p', p).$$

Moreover the norm endpoints of p, of p', and of the residual display $h - p$ are classified with 0^c_{Real}:

$$\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(p), 0^c_{\mathsf{Real}})$$

$$\mathsf{RealConstantHistoryClassifier}(\nu_{\mathsf{e}}(p'), 0^c_{\mathsf{Real}})$$

$$\mathsf{RealConstantHistoryClassifier}(\|h - p\|_{\langle,\rangle_{\mathsf{e}}}, 0^c_{\mathsf{Real}})$$

The residual inner-product rows used by the projection ledger are also classified with 0^c_{Real}:

$$\mathsf{RealConstantHistoryClassifier}(\langle h - p, p\rangle_{\mathsf{e}}, 0^c_{\mathsf{Real}})$$

$$\mathsf{RealConstantHistoryClassifier}(\langle p - p', p\rangle_{\mathsf{e}}, 0^c_{\mathsf{Real}})$$

$$\mathsf{RealConstantHistoryClassifier}(\langle h - p, p'\rangle_{\mathsf{e}}, 0^c_{\mathsf{Real}})$$

Thus the singleton projection row packages idempotence, zero norm readback, and zero orthogonal residual as one reusable projection obligation.

Proof. Unfold Definition 91.25. The projection sends every carried endpoint to e, so both p and p' are the selected endpoint of the Hilbert singleton carrier. The carried endpoint row is supplied by Definition 91.26.

Apply Definition 91.27. It gives the displayed classifier $\rho_{\mathsf{e}}(p', p)$ and reads the idempotence residual inner-product row $\langle p - p', p\rangle_{\mathsf{e}}$ as the constant RealUp zero endpoint.

For the norm endpoints, use Definition 91.13 for p and p' and use Definition 91.32 for the residual display. Since every displayed endpoint lies on the singleton carrier, each norm row is classified with 0^c_{Real}.

Finally apply Definition 91.22 to the carried singleton endpoints appearing in the residual ledger. It classifies the rows $\langle h - p, p\rangle_{\mathsf{e}}$ and $\langle h - p, p'\rangle_{\mathsf{e}}$ with 0^c_{Real}, completing the projection-residual zero package. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection residual zero package`

Theorem 91.29 (Hilbert singleton orthogonal projection NameCert surface)**.** *The row of Definition 91.25, together with Definition 91.12, Definition 91.13, Definition 91.26, Definition 91.27, and Definition 91.18, forms the paper-side singleton orthogonal-projection obligation surface for $\mathsf{NameCert}_{\mathsf{Hilbert}\uparrow}$. Its carrier is A_{e}, its classifier is ρ_{e}, its selected projection is π_{e}, its endpoint ledger is the zero norm, distance, and inner-product row, and its completeness source is the Banach singleton completion row. This surface asserts only the singleton projection obligation and does not claim a general projection theorem or a standard bridge.*

Proof. The carrier and classifier fields are fixed by Definition 91.12. They identify the Hilbert singleton carrier with A_{e} and its comparison row with ρ_{e}.

The concrete projection field is Definition 91.25. The carried endpoint and zero endpoint readbacks are supplied by Definition 91.26; these are the NameCert rows that state the projection lives on the singleton Hilbert surface.

Idempotence is supplied by Definition 91.27. It records that projecting the selected endpoint again returns the same ρ_{e}-class and that the residual inner-product row remains classified with 0^{c}_{Real}.

The uniqueness field is Definition 91.18. It compares any two limit or projection witnesses by ρ_{e}, so the selected projection row is the unique singleton endpoint in the certificate surface. The endpoint classification used by all these rows is Definition 91.13, and the completion source is Definition 90.3. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonOrthogonalProjection namecert surface`

Theorem 91.30 (Hilbert singleton projection standard bridge)**.** *The public singleton projection surface of Definition 91.29 bridges conservatively to the one-point Hilbert space. Under this bridge, the carrier A_{e} maps to the unique point, the classifier ρ_{e} maps to equality on that point, the projection π_{e} maps to the identity projection, and the Banach distance, NormUp norm, and inner-product rows map to the zero metric, zero norm, and zero inner product.*

Proof. The source side is the public singleton surface. Its carrier and classifier are fixed by Definition 91.12, so the only standard carrier compatible with the displayed row is the one-point Hilbert space, and ρ_{e} reads as equality on its unique point.

The projection row is fixed by Definition 91.26. Since the carried endpoint is the unique endpoint, Definition 91.27 identifies applying π_{e} again with the same endpoint, and Definition 91.18 prevents a second projection value. Thus the standard interpretation of π_{e} is the identity projection on the one-point Hilbert space.

It remains to read the scalar rows. The endpoint readback theorem Definition 91.13 identifies every displayed norm endpoint with 0^{c}_{Real}, while Definition 91.29 packages the Banach singleton distance, NormUp norm, and inner-product rows as the zero metric, zero norm, and zero inner product. These identifications give a conservative paper-side standard bridge for the singleton projection surface only. □

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection standard bridge`

Theorem 91.31 (Hilbert singleton orthogonal complement collapse)**.** *For the carrier of Definition 91.2, any displayed vector endpoint orthogonal to the singleton carried surface is classified by ρ_{e} with e. Conversely, every carried endpoint has the zero inner-product row against every carried endpoint:*

$$\mathsf{RealConstantHistoryClassifier}(\langle m, n\rangle_{\mathsf{e}}, 0^{c}_{\mathsf{Real}})$$

for carried histories $m, n : \mathsf{BHist}$. Thus the displayed orthogonal complement of the singleton carrier collapses to the singleton endpoint.

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton orthogonal complement collapse`

Proof. First identify the source surface. By Definition 91.12, the Hilbert carrier, the singleton vector-space carrier, and the Banach carrier all expose the same predicate A_{e} and classifier ρ_{e}.

Now apply Definition 47.4. Any displayed vector endpoint on this singleton vector-space surface is therefore classified by ρ_{e} with e; there is no separate vector endpoint available for an orthogonal-complement row.

For the converse direction, take carried endpoints $m, n : \mathsf{BHist}$. Definition 91.22 reads their inner-product endpoint as 0^{c}_{Real} under $\mathsf{RealConstantHistoryClassifier}$, so every carried endpoint is orthogonal to every other carried endpoint in the singleton Hilbert display. □

Theorem 91.32 (Hilbert singleton projection-residual decomposition)**.** *Let s be an accepted regular Cauchy stream in the Hilbert singleton carrier, and let p be a Hilbert singleton projection witness for its carried endpoint. Then p decomposes the endpoint into its unique ρ_{e}-classified projection plus a residual whose inner-product row is orthogonal to the singleton carrier and whose norm endpoint is 0^{c}_{Real}.*

Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingletonProjection residual decomposition`
Lean4 ✓ `BEDC.Derived.HilbertUp.HilbertSingleton projection residual decomposition`

Proof. Use Definition 91.18 to read the projection witness against the complete singleton Hilbert surface. Any other limit or projection witness for the same accepted stream is identified with p by ρ_{e}.

The projection component is therefore the singleton endpoint classified by ρ_{e} with e. The complete normed certificate Definition 91.7 supplies the matching Banach limit row and the norm-induced constant RealUp zero endpoint.

It remains to identify the residual row. Since both the carried endpoint and the projection component lie on the singleton carrier, Definition 91.22 classifies the inner-product row of the residual comparison with 0^{c}_{Real}.

Definition 91.31 then collapses the residual endpoint itself to the singleton classifier. Its norm endpoint is read back as 0^{c}_{Real} by the complete normed certificate, so the projection plus residual display is the unique singleton decomposition with zero orthogonal residual. □

Closure status for $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$**.**

Constructive story: A Hilbert object in this concrete closure is the same singleton carrier A_{e} seen simultaneously as Banach, normed, and inner-product data, with all three surfaces sharing the classifier ρ_{e} and the same empty-history endpoint. The inner product is the constant RealUp zero row, the induced norm is the singleton $\mathsf{Norm}^{\uparrow}$ norm, the Banach metric and completion rows are the singleton completion rows, and the projection map sends every carried endpoint back to e, so any non-singleton endpoint or separate closed-subspace carrier is excluded. The source is the compatible Banach-plus-inner-product package, the pattern records norm-induced metric compatibility, complete-limit stability, endpoint readback, projection existence, projection idempotence, uniqueness, orthogonal residual decomposition, and universal zero orthogonality, and the ledger keeps the Banach surface, NormUp zero row, constant inner-product row, selected projection endpoint, and residual rows aligned. The five name-certificate obligations close because e inhabits A_{e}, ρ_{e} supplies reflexivity, symmetry, and transitivity, and the compatibility rows transport Banach metric, norm, inner-product, limit, and projection witnesses along the same classifier. A history h lies in the signature gap of a $\mathsf{Hilbert}^{\uparrow}$ token p when the source-domain accepts the singleton compatible carrier and projection row, the signature relation pairs it with a projected answer endpoint and evidence ledger, and token introduction names that answer as p; unfolded, h is carried by A_{e} and the answer is the ρ_{e}-class of e with zero norm and zero residual inner product. The paper bridge reads the package as the one-point Hilbert space and reads back its unique point, zero norm, zero inner product, and projection endpoint through the same singleton rows. The chapter registers this singleton projection package under the name $\mathsf{Hilbert}^{\uparrow}$; arbitrary Hilbert spaces, arbitrary closed-subspace projections, spectral theory, and checked bridge verification are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The Hilbert surface is closed for the singleton projection NameCert package: carrier A_{e}, classifier ρ_{e}, projection endpoint π_{e}, Banach singleton completion row, NormUp zero row, constant inner-product zero row, residual decomposition rows, and the one-point standard bridge of Definition 91.30.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.HilbertUp.HilbertSingletonProjection\_semantic\_name\_certificate`

Bridge status: bridgeChecked
Not claimed: It does not close arbitrary Hilbert spaces, arbitrary closed-subspace projections, or spectral theory.
Upgrade path: no further upgrade – chapter at top of formal axis.

Measure consumers route through the calculus and integral sibling surface of chapter 3632; the dependency path is carried by `parts/concrete_instances/measure/probability_integral_dependency_route` below together with the adjacent probability-integral rows.

Chapter 92

A Concrete Naming Certificate for $\mathsf{Measure}^{\uparrow}$

The measure interface $\mathsf{Measure}^{\uparrow}$ packages a carrier with a constructive σ-additive non-negative real-valued set function on a designated σ-algebra of measurable subsets. It depends on $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38) and supplies the substrate for the integral certificate $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ (chapter 123).

The same carrier surface supplies the finite event-algebra rows and $\mathsf{Real}^{\uparrow}$-valued mass rows consumed by $\mathsf{ProbSpace}^{\uparrow}$ in chapter 184, with the real endpoints classified through chapter 38.

92.1 Carrier definition

Definition 92.1 (MeasureUp carrier surface)**.** *A* $\mathsf{Measure}^{\uparrow}$ *carrier surface is a tuple*

$$(\mathsf{MCarrier}, \sim_M, \mathcal{A}, \mu, \sim_R)$$

with the following rows. The carrier row is a predicate $\mathsf{MCarrier} : \mathsf{BHist} \to \mathsf{Prop}$*. The classifier row* $h \sim_M k$ *is a NameCert-style classifier on carried histories in the sense of Definition 9.7. The measurable-subset row* $\mathcal{A}(h, E)$ *assigns to each carried history* h *a designated family of measurable subsets* E *of its event object. The measure row* $\mu_h(E)$ *assigns to every* $\mathcal{A}(h, E)$ *a* $\mathsf{Real}^{\uparrow}$ *endpoint classified by the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$ *of Definition 38.9.*

Carrier transport is part of the surface: if $\mathsf{MCarrier}(h)$*,* $h \sim_M k$*, and* $\mathcal{A}(h, E)$*, then the transported event row is read in the* k*-fiber, and the corresponding measure values compare under* $\sim_R$*.*

Definition 92.2 (MeasureUp countable-additivity row)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier surface as in Definition 92.1, the countable-additivity row fixes, for every carried history* h*, a sequence* $E : \mathbb{N} \to \mathcal{P}(h)$ *satisfying:*

1. *each* E_n *is measurable,* $\mathcal{A}(h, E_n)$*;*
2. *the sequence is pairwise disjoint in the* h*-fiber;*
3. *the union event* $\bigcup_{n:\mathbb{N}} E_n$ *is supplied as a measurable row of* $\mathcal{A}(h, -)$*;*
4. *there is a countable* $\mathsf{Real}^{\uparrow}$ *sum endpoint* $\sum_{n=0}^{\infty} \mu_h(E_n)$*;*
5. *the head-tail readback compares this endpoint with* $\mu_h(E_0) + \sum_{n=0}^{\infty} \mu_h(E_{n+1})$ *under* $\sim_R$*;*
6. *pointwise* $\sim_R$*-classification of the summands transports to* $\sim_R$*-classification of the two countable sum endpoints.*

The row asserts the sigma-additivity comparison

$$\mu_h\left(\bigcup_{n:\mathbb{N}} E_n\right) \sim_R \sum_{n=0}^{\infty} \mu_h(E_n).$$

Theorem 92.3 (Measure carrier classifier obligations)**.** *Every* $\mathsf{Measure}^{\uparrow}$ *carrier surface supplies the NameCert-style reflexivity, symmetry, and transitivity rows for* $\sim_M$ *on carried histories, and it transports* $\mathsf{MCarrier}$*, measurable rows, and measure comparisons along that classifier.*

Proof. Unpack the carrier surface of Definition 92.1. Its classifier row is required to be a carrier naming certificate in the sense of Definition 9.7, so reflexivity, symmetry, transitivity, and carrier transport are projections of that row.

The measurable-subset and measure rows are indexed by carried histories. The surface states that a classifier comparison between carried histories re-reads the event row in the target fiber and compares the corresponding $\mathsf{Real}^{\uparrow}$ measure values. Applying that transport clause gives exactly the displayed carrier, classifier, and measure-stability obligations. □

Theorem 92.4 (Measure sigma-algebra obligations)**.** *For every* $\mathsf{Measure}^{\uparrow}$ *carrier surface, the designated measurable family* $\mathcal{A}(h, -)$ *is closed under the empty event, binary union, countable union, and complement rows used by the measure certificate whenever* $\mathsf{MCarrier}(h)$ *holds.*

Proof. Fix a carried history h. The measurable-subset row in Definition 92.1 is the designated sigma-algebra family for the h-fiber, so its certificate clauses give the empty event and complement rows.

The same sigma-algebra row supplies finite binary union as the two-term case of its closure data, and it supplies countable union for every measurable sequence. These are the exact rows consumed by the sigma-additivity comparison in Definition 92.2. □

Theorem 92.5 (Measure RealUp sum scope)**.** *The countable-additivity comparison of Definition 92.2 is scoped only to the* $\mathsf{Real}^{\uparrow}$ *classifier stability of Definition 38.9 and to the additive* $\mathsf{AbGroup}^{\uparrow}$ *zero, unit, and congruence rows cited from Definition 42.1. It does not assert an independent construction of real analysis outside those rows.*

Proof. Unpack the countable-additivity row. Its endpoint is a $\mathsf{Real}^{\uparrow}$ countable sum, and every comparison involving that endpoint is stated under the classifier $\sim_R$ from Definition 38.9.

The two measure consequences below use this endpoint only through head-tail readback, pointwise classifier stability, sigma-additivity, and the additive $\mathsf{AbGroup}^{\uparrow}$ rows of Definition 42.1. Thus the scope of the row is exactly the named classifier and additive certificate surface, with no extra analytic closure claim. □

Definition 92.6 (MeasureUp carrier obligation inventory)**.** *A* $\mathsf{Measure}^{\uparrow}$ *carrier is a* BHist *package whose rows expose the constructive measure-space data needed by the certificate. Its ten-field inventory is:*

1. *traditional target: a constructive measure space;*
2. *BEDC source:* $\mathsf{Real}^{\uparrow}$*, the additive* $\mathsf{AbGroup}^{\uparrow}$ *row for* $\mathsf{Real}^{\uparrow}$ *addition, and countable StreamName-style measurable families of chapter 126;*
3. *carrier choice: a* BHist *package containing a sample/event carrier, a* σ*-algebra predicate* $\mathcal{A}$*, a* $\mathsf{Real}^{\uparrow}$*-valued map* μ*, and rows for the empty event, binary union, countable union, and complement/relative difference;*

4. *classifier choice: componentwise* hsame *on the carrier package, together with the* $\mathsf{Real}^{\uparrow}$ *classifier on measure values;*
5. *stability obligations: carrier transport, measurable-set transport, μ-congruence, complement/relative-difference transport, σ-additivity, and head-tail readback for countable sums;*
6. *ledger: event rows, relative-difference rows, measurable sequence rows, and countable-sum rows;*
7. *core theorem list: empty set zero, finite disjoint-union additivity, relative-difference additivity, and countable disjoint additivity;*
8. *exactness predicate: measurable package coverage, soundness, and completeness;*
9. *standard bridge: absent at this paper site;*
10. *status: paper obligation surface.*

The countable-family row is read through StreamName finite observations and the $\mathsf{Real}^{\uparrow}$ *endpoint classifier, not through a host-level sequence primitive.*

Definition 92.7 (Measure classifier specification)**.** *For two* $\mathsf{Measure}^{\uparrow}$ *carriers M and N as in Definition 92.1, the public classifier $M \sim_{\mathsf{Measure}^{\uparrow}} N$ consists of shared event-carrier sameness and endpoint agreement. Shared event-carrier sameness says that an event witness of M and an event witness of N classify the same displayed event precisely when their BHist presentations are related by the displayed* hsame *transport rows and their measurable witnesses transport between the two displayed measurable families. Endpoint agreement says that, for every such shared measurable event, the endpoints $\mu_M(E)$ and $\mu_N(E')$ compare through* RealConstantHistoryClassifier *and hence through the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. The classifier does not assert quotient equality or host set equality; it only records the public event witness, measurable witness, and* $\mathsf{Real}^{\uparrow}$ *endpoint rows visible in the carrier ledger.*

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureClassifierSpecification`

Theorem 92.8 (Measure carrier transport obligation)**.** *Let $\mathcal{M}$ be a* $\mathsf{Measure}^{\uparrow}$ *certificate surface as in Definition 92.15. If $h, k : \mathsf{BHist}$ satisfy $\mathsf{MeasCode}(h)$ and $h \sim_M k$, then $\mathsf{MeasCode}(k)$. Moreover, every displayed row attached to the carried measure code, namely the measurable-family row, σ-algebra rows, complement/relative-difference row,* $\mathsf{Real}^{\uparrow}$ *endpoint map, σ-additivity row, head-tail row, and countable-sum classifier row, may be read at k after transporting its carrier proof along $h \sim_M k$.*

Proof. The carrier part is the transport field of the core $\mathsf{NameCert}(\mathsf{MeasCode}, \sim_M)$ in the surface. Apply it to the carried proof at h and the classifier comparison $h \sim_M k$.

The remaining rows are fields of the same certificate surface indexed by the carried measure code. Once the carrier proof has been moved from h to k, the row projections are read with that transported proof. No quotient or host equality of histories is used; the only movement is along the displayed NameCert classifier. □

Theorem 92.9 (MeasureUp carrier-classifier obligation)**.** *For the* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra carrier surface of Definition 92.14, the carrier slots are exactly the displayed* BHist *rows for measurable events, empty event, binary union, countable union, complement/relative-difference, disjoint sequence readback,* $\mathsf{Real}^{\uparrow}$ *measure values, hidden measurable-set membership, and countable-sum witnesses. The classifier slots are exactly componentwise* hsame *on those rows, together with the* $\mathsf{Real}^{\uparrow}$ *classifier on measure endpoints. Transport along the classifier carries each carrier slot to the corresponding slot of the target presentation.*

Proof. Read the ten fields of Definition 92.14. The carrier entries are the rows named in the carrier, stability, and ledger clauses, and no additional carrier row is listed by the exactness clause.

For transport, project the componentwise hsame witness at each visible row. The event rows move by their own history-sameness components, and the measure-value rows move by first projecting the endpoint comparison and then using $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ to read it as a $\mathsf{Real}^\uparrow$ classifier comparison.

The ledger rows are repacked after the visible rows move. Since exactness says that every hidden membership, relative-difference, or countable-sum witness is attached to one of the displayed rows, transporting the visible row also transports the only ledger entry that can justify it. □

Theorem 92.10 (Measure constant-empty sigma-additive row)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^\uparrow$ certificate surface as in Definition 92.15. Define $E_n := \varnothing$ for every $n : \mathbb{N}$. Then $E_\bullet$ is a measurable sequence, it is pairwise disjoint, its countable union is $\varnothing$, and the σ-additivity row gives the comparison*

$$\mu(\varnothing) \sim_R \Sigma_\mu(E_\bullet).$$

This is the sigma-additive input used by Definition 92.25.

Proof. The σ-algebra rows of the surface include the empty-event row, so each E_n is measurable. Pairwise disjointness is immediate because both sides of every pairwise intersection are the empty predicate.

The countable union of the constant-empty sequence is again the empty predicate. The countable-union row therefore identifies the union endpoint required by the σ-additivity row with $\varnothing$.

Apply the σ-additivity row to this measurable pairwise-disjoint sequence and its displayed union. The resulting comparison is precisely $\mu(\varnothing) \sim_R \Sigma_\mu(E_\bullet)$. □

Theorem 92.11 (MeasureUp sigma-additivity obligation)**.** *For the $\mathsf{Measure}^\uparrow$ sigma-algebra carrier surface of Definition 92.14, the sigma-additivity row classifies the value of a countable disjoint union by the displayed countable $\mathsf{Real}^\uparrow$ sum. Concretely, for every pairwise-disjoint measurable sequence $E_\bullet$ with countable union U, the certificate supplies*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

The same row supplies the measurable-sequence, disjointness, union, head-tail, and $\mathsf{Real}^\uparrow$ endpoint hypotheses consumed by Definition 92.25 and Definition 92.26.

Proof. Project the sigma-additivity row from the carrier surface and apply it to the displayed measurable sequence $E_\bullet$, its pairwise-disjointness witness, and its countable-union row U. The projected conclusion is exactly the $\mathsf{Real}^\uparrow$ classifier comparison between $\mu(U)$ and $\Sigma_\mu(E_\bullet)$.

For the empty-zero theorem, specialize $E_\bullet$ to the constant empty sequence. The empty-event, countable-union, disjointness, and head-tail rows are precisely the hypotheses required there.

For the finite disjoint-union theorem, specialize $E_\bullet$ to the sequence with first terms A and B and empty tail. Binary-union closure supplies the visible union row, while countable-union and head-tail readback supply the countable sum whose first two summands are $\mu(A)$ and $\mu(B)$. □

Theorem 92.12 (Measure countable-sum classifier stability obligation)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^\uparrow$ certificate surface as in Definition 92.15. Let $E_\bullet$ and $F_\bullet$ be measurable sequences. If every index satisfies $\mu(E_n) \sim_R \mu(F_n)$ through the $\mathsf{Real}^\uparrow$ classifier, then their countable endpoints satisfy*

$$\Sigma_\mu(E_\bullet) \sim_R \Sigma_\mu(F_\bullet).$$

In particular, the all-empty tail used in Definition 92.25 and the shifted empty tail used in Definition 92.26 transport to the same countable $\mathsf{Real}^{\uparrow}$ *sum.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHistPrefix classifier stability`

Proof. The countable-sum classifier row is part of the certificate surface. Feed it the two measurable sequences and the pointwise $\mathsf{Real}^{\uparrow}$ classifier comparison of their measure endpoints.

For the all-empty tail, the pointwise comparison is reflexivity of $\sim_R$ at each endpoint $\mu(\varnothing)$. For the shifted finite-union tail, every shifted term is the empty event, so the same reflexive pointwise comparison applies after reading the sequence definition at each index.

Thus both downstream arguments may replace a syntactically shifted empty countable sum by the canonical all-empty countable sum using only classifier transport, not host-level equality of sums. □

Theorem 92.13 (MeasureUp ledger exactness obligation)**.** *For the* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra carrier surface of Definition 92.14, the ledger contains exactly the empty-event, binary-union, countable-union, complement/relative-difference, disjointness, measure-value, and countable-sum rows. Every hidden measurable-set membership witness, every relative-difference witness, and every countable-sum witness is attached to one of those displayed rows, and no external choice object or quotient object is part of the ledger.*

Proof. Unpack the ledger and exactness clauses of Definition 92.14. The ledger clause lists only hidden measurable-set membership witnesses, relative-difference witnesses, and countable-sum witnesses, and the surrounding carrier and stability clauses list the visible rows to which those witnesses may attach.

Exactness is visible row exhaustion. Therefore every ledger witness must be read through the empty-event, binary-union, countable-union, complement/relative-difference, disjointness, measure-value, or countable-sum row that exposes it.

Since the ten-field surface names only BHist rows, hsame, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, and the additive $\mathsf{AbGroup}^{\uparrow}$ row, there is no remaining slot in which an external choice object or quotient object could be stored. □

Definition 92.14 (MeasureUp sigma-algebra carrier surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra carrier surface is the following ten-field BEDC package.*

1. *traditional target: a constructive measure space whose measurable predicates form a* σ*-algebra and whose measure map takes values in* $\mathsf{Real}^{\uparrow}$*;*
2. *BEDC source:* BHist *presentations, the certificate* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, and the additive* $\mathsf{AbGroup}^{\uparrow}$ *row on real histories;*
3. *carrier: a* BHist *package exposing measurable-event rows for the empty event, binary union, countable union, complement/relative-difference, disjoint sequence readback, and* $\mathsf{Real}^{\uparrow}$ *measure values;*
4. *classifier: componentwise* hsame *on the package rows, with the* $\mathsf{Real}^{\uparrow}$ *classifier applied to the visible measure endpoints;*
5. *stability rows: transport for the empty event, binary union, countable union, complement/relative-difference, disjoint sequence readback, and* $\mathsf{Real}^{\uparrow}$ *measure values;*
6. *ledger rows: hidden measurable-set membership witnesses, relative-difference witnesses, and countable-sum witnesses attached to the visible rows they justify;*

7. *core theorem list: Definition 92.25, Definition 92.26, and Definition 92.102;*
8. *exactness: visible row exhaustion, meaning every measurable-set, disjointness, relative-difference, value, and countable-sum witness consumed by the certificate is attached to one of the displayed rows;*
9. *standard bridge: no* StandardBridge *is asserted at this paper site;*
10. *status: paper-side target for the* $\mathsf{Measure}^{\uparrow}$ *NameCert obligation package.*

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureSigmaAlgebraCarrierSurface`

Definition 92.15 (MeasureUp certificate surface)**.** *A* $\mathsf{Measure}^{\uparrow}$ *certificate surface over the BEDC history substrate consists of the following data.*

1. *a carrier predicate* $\mathsf{MeasCode} : \mathsf{BHist} \to \mathsf{Prop}$ *for histories that present measure codes;*
2. *a classifier* $\sim_M : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ *and a core* $\mathsf{NameCert}(\mathsf{MeasCode}, \sim_M)$*, so membership in* $\mathsf{MeasCode}$ *transports along* $\sim_M$ *by the* $\mathsf{NameCert}$ *transport field;*
3. *a designated measurable-family predicate* $\mathsf{MeasFam}$ *on predicates* $E : \mathsf{BHist} \to \mathsf{Prop}$*, stable under the displayed classifier and under* hsame *on the endpoint histories that present the same event;*
4. *a* $\mathsf{Real}^{\uparrow}$*-valued endpoint map* μ *from measurable predicates to their measure endpoints, with equality of measurable presentations transported through the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$ *of Definition 38.9;*
5. *a countable-sum endpoint* $\Sigma_\mu(E_\bullet)$ *for each measurable sequence* $E_\bullet : \mathbb{N} \to (\mathsf{BHist} \to \mathsf{Prop})$*;*
6. σ*-algebra rows asserting* $\mathsf{MeasFam}(\varnothing)$*, binary-union closure for disjoint measurable pairs, countable-union closure for measurable sequences, and complement/relative-difference closure for displayed inclusions of measurable events in the same carried fiber;*
7. *a* σ*-additivity row: whenever* $E_\bullet$ *is a pairwise-disjoint measurable sequence with countable union* U*, the comparison* $\mu(U) \sim_R \Sigma_\mu(E_\bullet)$ *holds;*
8. *a head-tail readback row: for every measurable sequence* $E_\bullet$*, the endpoint* $\Sigma_\mu(E_\bullet)$ *is classified with* $\mu(E_0) + \Sigma_\mu(E_{\bullet+1})$ *by the* $\mathsf{Real}^{\uparrow}$ *additive operation;*
9. *a countable-sum classifier row: pointwise classifier equality of measurable sequences transports their countable endpoints through* $\sim_R$*.*

The dependencies of the surface are exactly the BEDC objects named above: $\mathsf{NameCert}$ *supplies carrier and classifier transport;* BHist *supplies the visible code substrate;* hsame *and* Cont *supply history sameness and continuation discipline for event presentations;* $\mathsf{Real}^{\uparrow}$ *supplies the codomain classifier; and the additive* $\mathsf{AbGroup}^{\uparrow}$ *row of Definition 42.1 supplies the addition, zero, congruence, and cancellation laws used by the displayed measure consequences.*

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureCertificateSurface`

Definition 92.16 (Measure complement and relative-difference row)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *certificate surface as in Definition 92.15, the complement/relative-difference row is the displayed carrier row with the following fields. Whenever* A *and* B *are measurable events in the same carried fiber and the public event readback displays* $B \subseteq A$*, the row supplies a measurable event* $A \setminus B$ *in that fiber. It also supplies a disjointness witness between* B *and* $A \setminus B$*, a binary-union readback classifying* $B \cup (A \setminus B)$ *as* A*, and a* $\mathsf{Real}^{\uparrow}$ *endpoint row for* $\mu(A \setminus B)$*.*

The same row is stable under the public $\mathsf{Measure}^{\uparrow}$ *classifier. If componentwise* hsame *transports the carried fiber, the event presentations, and the inclusion witness from one package to another, the*

relative-difference event, its measurable witness, its disjointness witness, its binary-union readback, and its measure endpoint transport to the corresponding target package. The endpoint movement is read through the $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$ *supplied by Definition 38.9.*

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureComplementDifferenceRow`

Theorem 92.17 (Measure root NameCert carrier and classifier)**.** *The root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *surface has carrier face* $\mathsf{MeasCode}$ *and classifier face* $\sim_M$ *exactly as displayed in Definition 92.14 and Definition 92.15. Its visible event rows are the empty-event, binary-union, countable-union, complement/relative-difference, disjoint-sequence readback, and countable-sum endpoint rows of those definitions, and its endpoint classifier rows are read through the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$*. The root face does not assert a quotient of measure codes and does not identify host event predicates by ambient set equality.*

Proof. Project the carrier list of Definition 92.14. The third, fourth, fifth, sixth, and eighth fields name the displayed measure-code carrier, the componentwise classifier, the event-row stability data, the ledger rows, and the visible-row exactness requirement.

Next project Definition 92.15. Its first two clauses name $\mathsf{MeasCode}$, $\sim_M$, and the core $\mathsf{NameCert}(\mathsf{MeasCode}, \sim_M)$ transport field. Its event-family, endpoint, countable-sum, σ-algebra, σ-additivity, head-tail, and countable-sum classifier clauses supply the visible rows and the $\mathsf{Real}^{\uparrow}$ endpoint classifier.

The statement follows by reading these projections as the root public face: transport is the $\mathsf{NameCert}$ transport field, while quotienting of codes and ambient host set equality never appear among the displayed fields. □

Theorem 92.18 (Measure root NameCert stability and exactness)**.** *For the root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *surface, Definition 92.9, Definition 92.12, Definition 92.11, and Definition 92.13 supply, respectively, componentwise transport of the carrier rows, classifier stability of countable-sum endpoints,* σ*-additive classification of disjoint countable unions, and visible ledger exhaustion for every witness consumed by the root surface.*

Proof. Start with Definition 92.9. It transports the displayed empty, binary-union, countable-union, complement/relative-difference, disjoint-readback, and endpoint rows componentwise along the public $\mathsf{Measure}^{\uparrow}$ classifier.

The countable endpoint row is stable by Definition 92.12. This keeps the $\mathsf{Real}^{\uparrow}$ endpoint classifier compatible with pointwise transport of measurable sequences.

The σ-additive classifier row is supplied by Definition 92.11: pairwise-disjoint measurable sequences with displayed union U classify $\mu(U)$ against the countable endpoint assigned to the sequence.

Finally Definition 92.13 states that the membership, relative-difference, disjointness, value, and countable-sum witnesses consumed by the surface are exactly the witnesses attached to the visible rows. These four rows therefore give the stability and exactness face of the root package. □

Theorem 92.19 (Measure root downstream boundary)**.** *The downstream public boundary of the root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *surface is exactly the collection exposed by Definition 92.27, Definition 92.59, and Definition 92.60. These are precisely the rows consumed by* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*, with no additional host carrier, quotient relation, or unlisted ledger witness exported to those chapters.*

Proof. Project Definition 92.27. It packages the carrier, classifier, stability, exactness, nonnegative value, monotone inclusion, and measure consequence rows as the public $\mathsf{Measure}^{\uparrow}$ substrate.

The probability boundary is the one isolated by Definition 92.59. It exposes the total event normalization and the finite-additivity rows used by $\mathsf{ProbSpace}^{\uparrow}$, while keeping the remaining measure witnesses inside the named package.

The integral and functional-analysis boundary is the one isolated by Definition 92.60. It exposes measurable event membership, nonnegative endpoints, monotone inclusion, relative-difference rows, and countable-sum rows to $\mathsf{Integral}^{\uparrow}$ and $\mathsf{FunctionalAnalysis}^{\uparrow}$. Together these projections give exactly the downstream boundary stated above. □

92.2 The certificate

Theorem 92.20 (Measure carrier and classifier stability)**.** *Let M and N be $\mathsf{Measure}^{\uparrow}$ carriers as in Definition 92.1, related by the classifier of Definition 92.7. If E is a displayed event of M, E' is a displayed event of N, and the event presentations satisfy $\mathsf{hsame}(E, E')$, then the measurable witness for E in $\mathcal{A}_M$ transports to a measurable witness for E' in $\mathcal{A}_N$. Moreover the visible measure endpoints satisfy*

$$\mu_M(E) \sim_R \mu_N(E')$$

through $\mathsf{RealConstantHistoryClassifier}$ *and the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9.*

Proof. The shared event-carrier component of Definition 92.7 is exactly the transport rule for displayed event witnesses. Apply it to the $\mathsf{hsame}(E, E')$ comparison to move the event witness and its measurable row from M to N.

The endpoint component of the classifier reads the two ledger endpoints attached to the transported measurable event. Since the carrier specification records the event witness, measurable witness, and $\mathsf{Real}^{\uparrow}$ endpoint in the same visible ledger, the endpoint comparison is the displayed $\mathsf{RealConstantHistoryClassifier}$ row.

Finally, Definition 38.9 identifies that row as a valid $\mathsf{Real}^{\uparrow}$ classifier comparison. No quotient of event codes and no host equality of subsets is introduced. □

Theorem 92.21 (MeasureUp classifier stability obligations)**.** *For a $\mathsf{Measure}^{\uparrow}$ carrier with inventory Definition 92.6, componentwise* hsame *of carrier packages transports every row named by the inventory. In particular, if two packages are componentwise* hsame*, then measurable membership in $\mathcal{A}$, the $\mathsf{Real}^{\uparrow}$-valued measure rows for μ, the σ-algebra closure rows, and the $\mathsf{Real}^{\uparrow}$ classifier on μ-values are preserved across the transport.*

Proof. Unpack the carrier inventory. The event carrier, the measurable predicate, the measure map, and the empty, binary-union, countable-union, and complement/relative-difference rows are separate components of the same BHist package.

Componentwise hsame supplies a transport witness for each component. Applying the measurable-set component transports membership in $\mathcal{A}$, and applying the closure-row components transports the empty, binary-union, and countable-union witnesses. The same componentwise projection transports the relative-difference event, its inclusion readback, its disjointness witness, and its binary-union readback.

It remains to identify the value rows. The μ-congruence component sends transported measurable events to transported $\mathsf{Real}^{\uparrow}$ values, and the $\mathsf{Real}^{\uparrow}$ classifier of Definition 38.9 composes those comparisons inside the codomain. Hence the classifier on μ-values is stable under the same componentwise carrier transport. □

Theorem 92.22 (MeasureUp sigma-additive certificate obligations)**.** *For a $\mathsf{Measure}^{\uparrow}$ carrier with inventory Definition 92.6, a $\mathsf{Measure}^{\uparrow}$ certificate consists of the following rows:*

1. *empty-event membership* $\varnothing \in \mathcal{A}$*;*
2. *binary disjoint-union closure in* $\mathcal{A}$*;*
3. *countable disjoint-union closure in* $\mathcal{A}$ *for StreamName-style measurable families;*
4. *complement/relative-difference closure for displayed inclusions* $B \subseteq A$ *of measurable events in the same carried fiber;*
5. σ*-additivity into the* $\mathsf{Real}^{\uparrow}$ *countable-sum classifier;*
6. *classifier-stable head-tail readback for those countable sums.*

With these rows, Definition 92.25 and Definition 92.26 are consequences of the certificate package, and the displayed relative-difference row supplies the finite difference readback of Definition 92.102.

Proof. The first row gives the all-empty measurable sequence used for the zero law. The countable disjoint-union row identifies its union as the empty event, and the σ-additivity row compares $\mu(\varnothing)$ with the associated $\mathsf{Real}^{\uparrow}$ countable sum.

Head-tail readback splits that sum into its head and tail. The classifier stability theorem transports the tail back to the original all-empty sum, and the additive $\mathsf{AbGroup}^{\uparrow}$ row then cancels the repeated summand exactly as in Definition 92.25.

For the finite disjoint-union law, use the measurable sequence whose first two terms are the given disjoint events and whose remaining terms are empty. Binary closure supplies measurability of $A \cup B$, countable closure supplies the same event as the countable union, and σ-additivity compares its measure with the corresponding countable sum.

Two applications of head-tail readback expose the first two summands. The empty tail is classified as 0_R by the zero law, and the additive $\mathsf{AbGroup}^{\uparrow}$ row removes that unit. The resulting comparison is the statement of Definition 92.26.

For displayed inclusions $B \subseteq A$, the complement/relative-difference row supplies the measurable event $A \setminus B$, the disjointness of B and $A \setminus B$, and the binary-union readback to A. Feeding those rows into finite disjoint-union additivity gives the finite difference comparison. □

Theorem 92.23 (Measure finite-support countable readback)**.** *Let* M *be a* $\mathsf{Measure}^{\uparrow}$ *carrier as in Definition 92.1. Let* $E_\bullet$ *be a pairwise-disjoint measurable sequence such that* $E_n = \varnothing$ *for every* $n \geq 2$*. Write*

$$S := \Sigma_\mu(E_\bullet), \qquad T := \Sigma_\mu(E_{\bullet+1}), \qquad U := \Sigma_\mu(E_{\bullet+2}).$$

If the carrier supplies classifier-stable head-tail readback, pointwise countable-sum stability, and the additive $\mathsf{AbGroup}^{\uparrow}$ *row of Definition 42.1 with zero* 0_R*, then*

$$S \sim_R \mu(E_0) + \mu(E_1).$$

Proof. Apply the head-tail row to the original sequence. It gives $S \sim_R \mu(E_0) + T$. Apply the same row to the shifted sequence. It gives $T \sim_R \mu(E_1) + U$.

It remains to read back the empty tail. Let $F_n := \varnothing$ for all n. Since $E_{n+2} = \varnothing$ by hypothesis, pointwise countable-sum stability gives $U \sim_R \Sigma_\mu(F_\bullet)$. The constant-empty σ-additivity row identifies $\mu(\varnothing)$ with $\Sigma_\mu(F_\bullet)$, and Definition 92.25 identifies $\mu(\varnothing)$ with 0_R.

Transport this comparison through the additive congruence supplied by Definition 42.1. The right-unit law then reads $\mu(E_1) + U$ as $\mu(E_1)$, and a final congruence step reads $\mu(E_0) + T$ as $\mu(E_0) + \mu(E_1)$. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHistPrefix finite support countable readback`

Theorem 92.24 (Measure certificate obligation threshold)**.** *A* $\mathsf{Measure}^{\uparrow}$ *NameCert obligation threshold is reached by the following paper-side data: a carrier satisfying Definition 92.1, a classifier satisfying Definition 92.7, the stability row of Definition 92.20, the finite-support countable readback row of Definition 92.23, the empty-set-zero theorem Definition 92.25, and the finite disjoint-union additivity theorem Definition 92.26. These data provide the carrier transport, classifier transport, σ-additive readback, zero endpoint, binary additivity, and visible ledger rows required by the public* $\mathsf{Measure}^{\uparrow}$ *certificate surface.*

Proof. The carrier and classifier definitions supply the public surface: event witnesses, measurable witnesses, endpoint rows, and classifier movement are all visible BHist and $\mathsf{Real}^{\uparrow}$ rows. The stability theorem is the transport field of the surface, with endpoint movement discharged by Definition 38.9.

The finite-support theorem turns the countable-sum row into the binary readback needed by finite disjoint union. The empty-set-zero theorem supplies the additive zero endpoint, and the additive $\mathsf{AbGroup}^{\uparrow}$ row of Definition 42.1 supplies the congruence and unit laws used in that readback.

Finally, Definition 92.26 packages these rows into the displayed binary additivity clause. The cancellation step used by the zero endpoint is exactly the group row cited in Definition 41.114, so the threshold is assembled from visible certificate fields rather than from external set equality. □

Theorem 92.25 (Measure empty set is zero)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let $\mathcal{A}$ be its designated σ-algebra, let μ be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Suppose $\varnothing \in \mathcal{A}$, and suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies σ-additivity for every pairwise-disjoint measurable sequence, together with classifier-stable head-tail readback for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sum. If the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row with zero 0_R, then*

$$\mu(\varnothing) \sim_R 0_R.$$

Proof. Let $E_n := \varnothing$ for every $n : \mathbb{N}$. The hypothesis $\varnothing \in \mathcal{A}$ makes every E_n measurable. The sequence is pairwise disjoint, and its union is again $\varnothing$.

Let S be the countable $\mathsf{Real}^{\uparrow}$ sum $\sum_{n=0}^{\infty} \mu(E_n)$. The σ-additivity row gives

$$\mu(\varnothing) \sim_R S.$$

The head-tail row for this sum gives

$$S \sim_R \mu(E_0) + \sum_{n=1}^{\infty} \mu(E_n).$$

Since $E_0 = \varnothing$, the first summand is $\mu(\varnothing)$. The tail sequence $(E_{n+1})_{n\geq 0}$ is pointwise the same empty sequence as $(E_n)_{n\geq 0}$, so countable-sum classifier stability gives

$$\sum_{n=1}^{\infty} \mu(E_n) \sim_R S.$$

Composing this with the σ-additivity comparison and transporting the right summand through addition yields

$$S \sim_R \mu(\varnothing) + \mu(\varnothing).$$

Composing once more with $\mu(\varnothing) \sim_R S$ gives

$$\mu(\varnothing) \sim_R \mu(\varnothing) + \mu(\varnothing).$$

By symmetry, $\mu(\varnothing) + \mu(\varnothing) \sim_R \mu(\varnothing)$.

The additive $\mathsf{AbGroup}^{\uparrow}$ row described in Definition 42.1 gives the group hypotheses for $+$ and 0_R. Applying Definition 41.114 with $a = \mu(\varnothing)$ and $b = \mu(\varnothing)$ forces the right summand to the additive unit, yielding $\mu(\varnothing) \sim_R 0_R$. □

Theorem 92.26 (Measure finite disjoint-union additivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B \in \mathcal{A}$ *be disjoint, and let* $A \cup B$ *be the measurable binary union supplied by the* σ*-algebra row. Suppose* $\varnothing \in \mathcal{A}$*, and suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for every pairwise-disjoint measurable sequence, together with classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums. If the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row with zero* 0_R*, then*

$$\mu(A \cup B) \sim_R \mu(A) + \mu(B).$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureFiniteDisjointUnion additivity`

Proof. Define a measurable sequence by $E_0 := A$, $E_1 := B$, and $E_n := \varnothing$ for every $n \geq 2$. Each term is measurable because $A, B, \varnothing \in \mathcal{A}$. The sequence is pairwise disjoint: the only nonempty pair to check is A against B, and this is the displayed disjointness hypothesis. Its countable union is exactly $A \cup B$.

Let

$$S := \sum_{n=0}^{\infty} \mu(E_n), \qquad T := \sum_{n=0}^{\infty} \mu(E_{n+1}), \qquad U := \sum_{n=0}^{\infty} \mu(E_{n+2}).$$

The σ-additivity row gives

$$\mu(A \cup B) \sim_R S.$$

The first head-tail readback gives

$$S \sim_R \mu(E_0) + T,$$

hence $S \sim_R \mu(A) + T$. The second head-tail readback gives

$$T \sim_R \mu(E_1) + U,$$

hence $T \sim_R \mu(B) + U$.

It remains to identify the empty tail. Let $F_n := \varnothing$ for every n, and let $S_\varnothing := \sum_{n=0}^{\infty} \mu(F_n)$. Since $E_{n+2} = F_n$ for every n, pointwise classifier stability of the countable sum gives

$$U \sim_R S_\varnothing.$$

Applying σ-additivity to the all-empty sequence gives $\mu(\varnothing) \sim_R S_\varnothing$, while Definition 92.25 gives $\mu(\varnothing) \sim_R 0_R$. By symmetry and transitivity of the $\mathsf{Real}^{\uparrow}$ classifier, $S_\varnothing \sim_R 0_R$, and therefore

$$U \sim_R 0_R.$$

The additive $\mathsf{AbGroup}^{\uparrow}$ row of Definition 42.1 supplies addition congruence and the right-unit law. Transporting $U \sim_R 0_R$ through addition gives

$$\mu(B) + U \sim_R \mu(B) + 0_R \sim_R \mu(B),$$

so $T \sim_R \mu(B)$. Transporting this comparison through addition with the fixed left summand $\mu(A)$ gives

$$\mu(A) + T \sim_R \mu(A) + \mu(B).$$

Composing this with $S \sim_R \mu(A) + T$ and with $\mu(A \cup B) \sim_R S$ yields the displayed classification. □

Theorem 92.27 (MeasureUp NameCert obligation package)**.** *The rows displayed in Definition 92.14, together with the classifier transport of Definition 92.9, the complement transport of Definition 92.100, the sigma-additivity classification of Definition 92.11, the ledger exactness of Definition 92.13, the nonnegative value obligation of Definition 92.98, the monotone measurable inclusion row of Definition 92.99, and the measure consequence theorems Definition 92.25, Definition 92.26, Definition 92.102, Definition 92.28, Definition 92.29, Definition 92.30, Definition 92.31, Definition 92.32, Definition 92.33, and Definition 92.34, form the* $\mathsf{Measure}^{\uparrow}$ *NameCert obligation package. This package is the measure substrate consumed by* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureNamecertObligationPackage`

Proof. Start with the carrier surface and project its visible event, value, stability, and ledger rows. Repack those projections as the carrier and classifier side of the $\mathsf{Measure}^{\uparrow}$ certificate.

The carrier-classifier obligation supplies componentwise transport for every displayed row, including the complement/relative-difference row. The complement-row transport theorem makes that projection available under the public classifier. The sigma-additivity obligation supplies the countable-union classification needed by the measure consequences, while the ledger exactness obligation identifies all hidden witnesses with their visible rows.

Finally include the nonnegative value row, the measurable-inclusion monotonicity row, and the empty-zero, finite disjoint-union, relative-difference, null-event neutral, symmetric finite-readback, empty-union, zero-tail, finite three-support, and triple disjoint-union consequence theorems as the core theorem rows of the package. The downstream chapters named in the statement consume only this package surface: $\mathsf{Integral}^{\uparrow}$ uses the measurable event, nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum rows, $\mathsf{ProbSpace}^{\uparrow}$ uses the empty and disjoint-union rows, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ uses the $\mathsf{Real}^{\uparrow}$ value classifier and additive compatibility rows. □

Theorem 92.28 (Measure null event addition neutral)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* E *be a measurable event, let* $\varnothing$ *be the empty measurable event, and let* 0_R *be the additive unit in the* $\mathsf{Real}^{\uparrow}$ *additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then the finite measure endpoint satisfies both neutral readbacks*

$$\mu(E) + \mu(\varnothing) \sim_R \mu(E)$$

and

$$\mu(\varnothing) + \mu(E) \sim_R \mu(E).$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureCountableZeroTail canonical`

Proof. Definition 92.25 gives $\mu(\varnothing) \sim_R 0_R$ from the all-empty countable disjoint family. The $\mathsf{Real}^{\uparrow}$ classifier transport row carries this comparison through addition with the fixed endpoint $\mu(E)$.

For the right neutral readback, additive congruence gives $\mu(E) + \mu(\varnothing) \sim_R \mu(E) + 0_R$. The right-unit row of the additive $\mathsf{AbGroup}^{\uparrow}$ structure then gives $\mu(E) + 0_R \sim_R \mu(E)$, and transitivity of $\sim_R$ composes the two comparisons.

For the left neutral readback, the same transport gives $\mu(\varnothing) + \mu(E) \sim_R 0_R + \mu(E)$. The left-unit row of the additive $\mathsf{AbGroup}^{\uparrow}$ structure gives $0_R + \mu(E) \sim_R \mu(E)$, so transitivity gives the displayed comparison. □

Theorem 92.29 (Measure finite disjoint-union symmetric readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* A *and* B *be disjoint measurable events, and let* $A \cup B$ *be the measurable binary union row. If the ordered finite readback gives*

$$\mu(A \cup B) \sim_R \mu(A) + \mu(B),$$

then the symmetric readback also holds:

$$\mu(A \cup B) \sim_R \mu(B) + \mu(A).$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHistClassifier empty union comm`

Proof. Apply Definition 92.26 to the displayed disjoint measurable pair. This supplies the ordered comparison $\mu(A \cup B) \sim_R \mu(A) + \mu(B)$ from empty-zero, head-tail readback, and finite-support countable additivity.

The additive commutativity row of the $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ surface gives $\mu(A) + \mu(B) \sim_R \mu(B) + \mu(A)$. Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier composes the ordered finite readback with this commutativity row and gives the symmetric readback. □

Theorem 92.30 (Measure empty-union right unit)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A \in \mathcal{A}$*, and let* $A \cup \varnothing$ *be the measurable binary union row supplied by the* σ*-algebra row. Suppose* $\varnothing \in \mathcal{A}$*, and suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies the sigma-additivity, head-tail, and countable-sum stability rows used in Definition 92.26. If the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row with zero* 0_R*, then*

$$\mu(A \cup \varnothing) \sim_R \mu(A).$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureEmptyUnion right unit`

Proof. The events A and $\varnothing$ are disjoint, and the displayed binary union row is measurable. Apply Definition 92.26 with $B = \varnothing$ to obtain

$$\mu(A \cup \varnothing) \sim_R \mu(A) + \mu(\varnothing).$$

By Definition 92.25, $\mu(\varnothing) \sim_R 0_R$. Addition congruence in the additive $\mathsf{AbGroup}^{\uparrow}$ row transports the right summand to 0_R, and the right-unit law gives $\mu(A) + 0_R \sim_R \mu(A)$. Transitivity of $\sim_R$ yields the displayed comparison. □

Theorem 92.31 (Measure disjoint-union symmetric readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B \in \mathcal{A}$ *be disjoint, and let* $A \cup B$ *be the measurable binary union supplied by the* σ*-algebra row. Suppose* $\varnothing \in \mathcal{A}$*, and suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies the sigma-additivity, head-tail, and countable-sum stability rows used in Definition 92.26. If the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row, then*

$$\mu(A \cup B) \sim_R \mu(B) + \mu(A).$$

Proof. Apply Definition 92.26 to the disjoint measurable pair A, B. This gives

$$\mu(A \cup B) \sim_R \mu(A) + \mu(B).$$

The additive $\mathsf{AbGroup}^{\uparrow}$ row supplies commutativity for $+$. Hence $\mu(A)+\mu(B) \sim_R \mu(B)+\mu(A)$, and composing the two comparisons gives the stated symmetric readback. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHistClassifier union symmetry`

Theorem 92.32 (Measure countable zero tail canonical)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, write* 0_R *for the additive zero in the target* $\mathsf{Real}^{\uparrow}$ *row, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $F_\bullet$ *be the constant-empty measurable sequence* $F_n := \varnothing$*. Suppose* $\varnothing \in \mathcal{A}$*, the sequence is read by the* σ*-additivity row of Definition 92.10, the countable-sum classifier is stable under pointwise* $\sim_R$*-classification, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by the additive* $\mathsf{AbGroup}^{\uparrow}$ *row of Definition 42.1. Then the canonical countable endpoint of the all-empty sequence is classified with zero:*

$$\Sigma_\mu(F_\bullet) \sim_R 0_R.$$

Proof. Apply the σ-additivity row to the all-empty measurable sequence. Its countable union is the empty event, so the row gives a classifier comparison between $\mu(\varnothing)$ and $\Sigma_\mu(F_\bullet)$.

The zero endpoint theorem Definition 92.25 classifies $\mu(\varnothing)$ with 0_R. The $\mathsf{Real}^{\uparrow}$ classifier fields from Definition 38.9 compose the two comparisons and orient the result toward the countable endpoint.

Countable-sum classifier stability makes this endpoint independent of the chosen constant-empty presentation. The additive $\mathsf{AbGroup}^{\uparrow}$ row supplies the zero endpoint used by the comparison, so the all-empty countable sum has the canonical classification displayed above. □

Theorem 92.33 (Measure finite three-support countable readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, write* 0_R *for the additive zero in the target* $\mathsf{Real}^{\uparrow}$ *row, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $E_\bullet$ *be a pairwise-disjoint measurable sequence such that* $E_n = \varnothing$ *for every* $n \geq 3$*. Suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for pairwise-disjoint measurable sequences, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then*

$$\Sigma_\mu(E_\bullet) \sim_R (\mu(E_0) + \mu(E_1)) + \mu(E_2).$$

Proof. Let

$$S := \Sigma_\mu(E_\bullet), \qquad T := \Sigma_\mu(E_{\bullet+1}), \qquad U := \Sigma_\mu(E_{\bullet+2}), \qquad V := \Sigma_\mu(E_{\bullet+3}).$$

Apply head-tail readback to the original sequence, to the once-shifted sequence, and to the twice-shifted sequence. These comparisons classify S with $\mu(E_0) + T$, classify T with $\mu(E_1) + U$, and classify U with $\mu(E_2) + V$.

The three-shifted tail is the all-empty sequence by the support hypothesis. Pointwise countable-sum stability transports V to the canonical empty-sequence endpoint, and Definition 92.32 classifies that endpoint with 0_R.

Use additive congruence from Definition 42.1 to transport the comparison $V \sim_R 0_R$ through the third head-tail equation. The right-unit law reads $\mu(E_2) + 0_R$ as $\mu(E_2)$, so $U \sim_R \mu(E_2)$.

Transport this comparison through the second and first head-tail equations. Associativity from the additive $\mathsf{AbGroup}^{\uparrow}$ row rewrites $\mu(E_0) + (\mu(E_1) + \mu(E_2))$ as $(\mu(E_0) + \mu(E_1)) + \mu(E_2)$, giving the displayed classification. □

Theorem 92.34 (Measure triple disjoint-union additivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B, C \in \mathcal{A}$ *be pairwise disjoint. Suppose the measurable binary-union rows display* $A \cup B \in \mathcal{A}$ *and* $(A \cup B) \cup C \in \mathcal{A}$*, and suppose*

the $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for pairwise-disjoint measurable sequences, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then*

$$\mu((A \cup B) \cup C) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

Proof. Define $E_0 := A$, $E_1 := B$, $E_2 := C$, and $E_n := \varnothing$ for every $n \geq 3$. The displayed measurable rows make each term measurable, and pairwise disjointness of A, B, C is the only nonempty disjointness check.

The countable union of this sequence is the displayed event $(A \cup B) \cup C$. Applying σ-additivity gives a classifier comparison between $\mu((A \cup B) \cup C)$ and $\Sigma_\mu(E_\bullet)$.

The finite three-support readback theorem Definition 92.33 classifies $\Sigma_\mu(E_\bullet)$ with $(\mu(E_0) + \mu(E_1)) + \mu(E_2)$. Substituting the three displayed heads and composing the $\mathsf{Real}^{\uparrow}$ classifier comparisons gives the claimed triple disjoint-union additivity. □

Theorem 92.35 (Measure two-step finite-additivity package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *finite-additivity readback surface consists of the empty-zero row, null-event neutrality, ordered binary disjoint-union additivity, and symmetric binary disjoint-union additivity. Its fit is the following ten-field package:*

1. *traditional target: finite additive measure readback;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{AbGroup}^{\uparrow}$*;*
3. *carrier: measurable* BHist *event rows;*
4. *classifier: the endpoint classifier* $\sim_R$*;*
5. *stability: classifier transport and additive congruence;*
6. *ledger: empty-event, binary-union, and countable-tail rows;*
7. *core theorem list: Definition 92.25, Definition 92.28, Definition 92.26, and Definition 92.29;*
8. *exactness: visible finite support readback;*
9. *standard bridge: absent at this paper site;*
10. *status: paper target.*

Proof. Project the empty measurable event, binary-union row, countable disjoint-family row, head-tail countable-sum row, endpoint classifier, and additive $\mathsf{AbGroup}^{\uparrow}$ rows from Definition 92.27.

The empty-zero row is exactly Definition 92.25. The two null-event neutrality rows are Definition 92.28, obtained by transporting empty-zero through additive congruence and then applying the left and right unit laws.

For a disjoint measurable pair, the ordered binary readback is Definition 92.26. Composing that ordered readback with additive commutativity gives Definition 92.29. These four rows are precisely the finite support surface required by downstream $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumers. □

Theorem 92.36 (Measure triple disjoint-union associativity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9, and write* $X \sim_M Y$ *for the public event classifier of Definition 92.7. Let* $A, B, C \in \mathcal{A}$ *be pairwise disjoint displayed events in one carried fiber. Let* U_{AB} *be the displayed binary union* $A \cup B$*, let* U_{BC} *be the displayed binary union* $B \cup C$*, let* U_L *be the displayed binary union* $U_{AB} \cup C$*, and let* U_R *be the displayed binary union* $A \cup U_{BC}$*. Assume*

the binary-union rows unfold membership by disjunction in the displayed fiber, and assume the hypotheses of Definition 92.26 for the disjoint pairs A, B, B, C, U_{AB}, C, and A, U_{BC}. If the target $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row, then the two bracketed finite unions are classifier-equal as measurable events,*

$$U_L \sim_M U_R,$$

and their measure endpoints are classifier-equal,

$$\mu(U_L) \sim_R \mu(U_R).$$

Proof. First compare the displayed event rows. For any point of the carried fiber, the membership readback for U_L unfolds to

$$(x \in A \vee x \in B) \vee x \in C,$$

while the membership readback for U_R unfolds to

$$x \in A \vee (x \in B \vee x \in C).$$

The two implications are obtained by cases on the displayed disjunctions: from the left display, either the first summand gives $x \in A$ or $x \in B$, or the second summand gives $x \in C$; from the right display, either $x \in A$, or the right summand gives $x \in B$ or $x \in C$. Thus the two binary-union presentations have the same public membership readback. The measurable witnesses transport along this readback, so the event component of Definition 92.7 gives

$$U_L \sim_M U_R.$$

Pairwise disjointness of A, B, C also gives the two compound disjointness witnesses needed below. If a point lies in both U_{AB} and C, the membership readback for U_{AB} puts it in A or in B, contradicting the corresponding pairwise disjointness with C. Hence U_{AB} and C are disjoint. Similarly, if a point lies in both A and U_{BC}, the membership readback for U_{BC} puts it in B or in C, contradicting the corresponding pairwise disjointness with A. Hence A and U_{BC} are disjoint.

Apply Definition 92.26 to A and B:

$$\mu(U_{AB}) \sim_R \mu(A) + \mu(B).$$

Apply the same theorem to U_{AB} and C:

$$\mu(U_L) \sim_R \mu(U_{AB}) + \mu(C).$$

Addition congruence in the additive $\mathsf{AbGroup}^{\uparrow}$ row transports the first comparison through the left summand of the second comparison, giving

$$\mu(U_{AB}) + \mu(C) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

By transitivity of $\sim_R$,

$$\mu(U_L) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

Apply Definition 92.26 to B and C:

$$\mu(U_{BC}) \sim_R \mu(B) + \mu(C).$$

Apply the same theorem to A and U_{BC}:

$$\mu(U_R) \sim_R \mu(A) + \mu(U_{BC}).$$

Addition congruence transports the comparison for U_{BC} through the fixed left summand $\mu(A)$, so

$$\mu(A) + \mu(U_{BC}) \sim_R \mu(A) + (\mu(B) + \mu(C)).$$

The additive associativity row of Definition 42.1 classifies

$$\mu(A) + (\mu(B) + \mu(C)) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

Composing these comparisons yields

$$\mu(U_R) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

Finally, symmetry and transitivity of the $\mathsf{Real}^{\uparrow}$ classifier compose the two readbacks to the common endpoint $(\mu(A) + \mu(B)) + \mu(C)$. Therefore

$$\mu(U_L) \sim_R \mu(U_R).$$

□

Definition 92.37 (Measure four-support sequence surface)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, a four-support sequence surface consists of measurable events* $E_0, E_1, E_2, E_3 \in \mathcal{A}$*, a pairwise-disjoint measurable sequence* $E_\bullet$*, and the empty event* $\varnothing \in \mathcal{A}$ *such that*

$$E_\bullet(0) = E_0, \qquad E_\bullet(1) = E_1, \qquad E_\bullet(2) = E_2, \qquad E_\bullet(3) = E_3$$

and $E_\bullet(n) = \varnothing$ *for every* $n \geq 4$*. The surface records only the displayed event rows, their pairwise-disjointness witnesses, the countable-union row, the countable* $\mathsf{Real}^{\uparrow}$ *sum endpoint* $\Sigma_\mu(E_\bullet)$*, and the head-tail readback rows supplied by the* $\mathsf{Measure}^{\uparrow}$ *certificate.*

Theorem 92.38 (Measure four-support countable readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, write* 0_R *for the additive zero in the target* $\mathsf{Real}^{\uparrow}$ *row, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $E_\bullet$ *be a four-support sequence surface in the sense of Definition 92.37. Suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for pairwise-disjoint measurable sequences, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then*

$$\Sigma_\mu(E_\bullet) \sim_R (((\mu(E_0) + \mu(E_1)) + \mu(E_2)) + \mu(E_3)).$$

Proof. Let $F_\bullet$ be the shifted sequence $F_n := E_\bullet(n+1)$. The four-support surface makes $F_0 = E_1$, $F_1 = E_2$, $F_2 = E_3$, and $F_n = \varnothing$ for every $n \geq 3$, so $F_\bullet$ is a finite three-support sequence with the same measurable and pairwise-disjoint rows.

Apply head-tail readback to $E_\bullet$. It classifies $\Sigma_\mu(E_\bullet)$ with $\mu(E_0) + \Sigma_\mu(F_\bullet)$.

Apply Definition 92.33 to $F_\bullet$. This classifies the shifted endpoint with $(\mu(E_1) + \mu(E_2)) + \mu(E_3)$, using countable zero-tail exactness at the three-shifted tail and the additive unit rows of $\mathsf{Real}^{\uparrow}$.

Transport the shifted comparison through the first summand by additive congruence. Associativity in the additive $\mathsf{AbGroup}^{\uparrow}$ row rewrites $\mu(E_0) + ((\mu(E_1) + \mu(E_2)) + \mu(E_3))$ first as $(\mu(E_0) + (\mu(E_1) + \mu(E_2))) + \mu(E_3)$ and then as $((\mu(E_0) + \mu(E_1)) + \mu(E_2)) + \mu(E_3)$. Transitivity of $\sim_R$ composes these classifier comparisons and gives the displayed endpoint. □

Theorem 92.39 (Measure quadruple disjoint-union additivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B, C, D \in \mathcal{A}$ *be pairwise disjoint. Suppose the measurable binary-union rows display* $A \cup B \in \mathcal{A}$*,* $(A \cup B) \cup C \in \mathcal{A}$*, and* $((A \cup B) \cup C) \cup D \in \mathcal{A}$*, and suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for pairwise-disjoint measurable sequences, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then*

$$\mu(((A \cup B) \cup C) \cup D) \sim_R (((\mu(A) + \mu(B)) + \mu(C)) + \mu(D)).$$

Proof. Define $E_0 := A$, $E_1 := B$, $E_2 := C$, $E_3 := D$, and $E_n := \varnothing$ for every $n \geq 4$. The displayed measurable rows make each term measurable, and pairwise disjointness of A, B, C, D is the only nonempty disjointness check.

The countable union of this sequence is the displayed event $((A \cup B) \cup C) \cup D$. Applying σ-additivity gives a classifier comparison between $\mu(((A \cup B) \cup C) \cup D)$ and $\Sigma_\mu(E_\bullet)$.

The four-support readback theorem Definition 92.38 classifies $\Sigma_\mu(E_\bullet)$ with $((\mu(E_0) + \mu(E_1)) + \mu(E_2)) + \mu(E_3)$. Substituting the four displayed heads and composing the $\mathsf{Real}^{\uparrow}$ classifier comparisons gives the claimed quadruple disjoint-union additivity. □

Theorem 92.40 (Measure four-support analysis consumer package)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, the four-support analysis package consumed by probability partitions, simple-integral rectangles, and functional-analysis positive decompositions has exactly these visible rows:*

1. *four displayed measurable event rows* E_0, E_1, E_2, E_3 *with pairwise disjointness witnesses;*
2. *the countable empty-tail sequence surface of Definition 92.37;*
3. *the four-support countable endpoint classification of Definition 92.38;*
4. *the displayed quadruple disjoint-union measure row of Definition 92.39.*

The package exposes no host set equality, no quotient event carrier, and no extra $\mathsf{Real}^{\uparrow}$ *operation beyond the additive* $\mathsf{AbGroup}^{\uparrow}$ *associativity, unit, congruence, and classifier-transitivity rows used in those four visible rows.*

Proof. Project the displayed event rows, empty-tail sequence, countable-union row, head-tail readback rows, and $\mathsf{Real}^{\uparrow}$ endpoint classifier from the $\mathsf{Measure}^{\uparrow}$ certificate. Project additive associativity, unit, congruence, and classifier transitivity from the target $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ surface.

The countable endpoint row is Definition 92.38. The public finite union row is Definition 92.39. These two rows use only the four displayed event entries, the empty tail, and the listed additive $\mathsf{Real}^{\uparrow}$ structure.

Probability partitions, simple-integral rectangles, and positive decompositions read the same rows: they need four visible measurable pieces, their classified measure endpoints, the bracketed finite sum, and the displayed disjoint-union endpoint. Since every step is phrased through measurable-event witnesses and $\mathsf{Real}^{\uparrow}$ classifier comparisons, no host set equality or quotient event carrier is introduced. □

92.3 Continuity from below row

Theorem 92.41 (Measure monotone partial-union chain)**.** *Let* M *be a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, with designated measurable family* $\mathcal{A}$*, measure map* μ*, and event classifier* $\sim_M$*. Let* $E_\bullet$ *be a displayed increasing measurable sequence in one carried* BHist *fiber: every* E_n *is in* $\mathcal{A}$*,*

and the public event readback displays $E_n \subseteq E_{n+1}$. Then the $\mathsf{Measure}^{\uparrow}$ *sigma-algebra and relative-difference rows supply measurable finite partial unions P_n and pairwise disjoint measurable layers D_n such that*

$$D_0 = E_0, \qquad D_{n+1} = E_{n+1} \setminus E_n, \qquad P_n \sim_M E_n.$$

Moreover P_n is classified as the displayed disjoint union $D_0 \sqcup \cdots \sqcup D_n$, and the public monotone row classifies $\mu(E_n) \leq_R \mu(E_{n+1})$ for every n.

Proof. Start with $D_0 := E_0$ and $P_0 := E_0$. The displayed measurability of E_0 gives both rows, and the classifier comparison $P_0 \sim_M E_0$ is reflexivity of the $\mathsf{Measure}^{\uparrow}$ event classifier.

For the successor step, apply Definition 92.101 to the displayed inclusion $E_n \subseteq E_{n+1}$. It supplies the measurable layer $D_{n+1} := E_{n+1} \setminus E_n$, its disjointness from E_n, and the binary-union readback

$$E_n \sqcup D_{n+1} \sim_M E_{n+1}.$$

Use the induction readback $P_n \sim_M E_n$ to transport this binary-union row from E_n to P_n. The sigma-algebra binary-union row then supplies $P_{n+1} := P_n \sqcup D_{n+1}$ as a measurable event, classified with E_{n+1} by the composed event classifier comparison.

The disjointness of the accumulated layers follows by induction from the displayed chain: earlier layers lie inside E_n by the previous partial-union readback, while the new layer is disjoint from E_n. Finally apply Definition 92.99 to each displayed inclusion $E_n \subseteq E_{n+1}$ to obtain $\mu(E_n) \leq_R \mu(E_{n+1})$. □

Theorem 92.42 (Measure continuity from below package)**.** *Let M be a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. For a displayed increasing measurable sequence $E_\bullet$, let $D_0 := E_0$ and $D_{n+1} := E_{n+1} \setminus E_n$ be the measurable disjoint layers supplied by Definition 92.41. If $U := \bigcup_{n:\mathbb{N}} E_n$ is the countable-union row of the same measurable sequence, then the σ-additivity row classifies*

$$\mu(U) \sim_R \Sigma_\mu(D_\bullet).$$

For every finite prefix, the finite disjoint-union readback classifies

$$\Sigma_\mu(D_0, \ldots, D_n) \sim_R \mu(E_n),$$

where the left endpoint denotes the visible finite-support countable sum associated to the first $n+1$ layers and an empty tail.

Proof. Apply Definition 92.41 to obtain the layer sequence $D_\bullet$, the partial unions P_n, and the classifier readbacks $P_n \sim_M E_n$. The same theorem gives pairwise disjointness of the layers and measurability of every layer.

The countable union of the layers is classified with the countable union $U := \bigcup_n E_n$: every layer lies in the corresponding E_n, and every element visible in some E_n lies in the finite partial union $D_0 \sqcup \cdots \sqcup D_n$ by the partial-union readback. This is an event classifier comparison, not a host-level equality of subsets.

Apply the σ-additivity row of Definition 92.22 to the pairwise-disjoint measurable sequence $D_\bullet$. It gives the comparison between the measure of the displayed countable union of the layers and $\Sigma_\mu(D_\bullet)$, and event-classifier transport carries the left endpoint to $\mu(U)$.

For a fixed n, apply finite disjoint-union additivity along the visible binary-union tree for $P_n = D_0 \sqcup \cdots \sqcup D_n$. The finite-support countable readback rows of Definition 92.23 and Definition 92.33, iterated over the displayed finite tree, identify that finite-support countable endpoint with the measure of P_n. Transport along $P_n \sim_M E_n$ gives the displayed classification with $\mu(E_n)$. □

Theorem 92.43 (Measure continuity from below integral scope)**.** *The* $\mathsf{Integral}^{\uparrow}$ *premeasure interface may consume the continuity-from-below package of Definition 92.42 exactly through the public* $\mathsf{Measure}^{\uparrow}$ *rows: displayed measurable events, relative-difference layers, finite disjoint-union readbacks,* σ*-additivity for the disjoint layer sequence, countable-sum classifier stability, and* $\mathsf{Real}^{\uparrow}$ *endpoint transport. It receives no standard bridge row and no host set equality row from this paper site.*

Proof. Project the premeasure rows named in Definition 92.60. These are the measurable-event ledger, endpoint classifier, countable-sum endpoint, head-tail readback, and finite disjoint-union rows exposed by the public $\mathsf{Measure}^{\uparrow}$ package.

The layer construction uses only the relative-difference row from Definition 92.101 and the monotone inclusion row from Definition 92.99. The endpoint classification uses only the σ-additivity and countable-sum rows named by Definition 92.22.

Thus an $\mathsf{Integral}^{\uparrow}$ consumer receives the named continuity package as a premeasure obligation over visible BHist event rows. The only identifications available to that consumer are $\mathsf{Measure}^{\uparrow}$ event classifier comparisons and $\mathsf{Real}^{\uparrow}$ endpoint classifier comparisons, so no standard bridge or host equality is imported. □

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralVisibleCarrierRow measure prefix scope`

Definition 92.44 (Measure nested-difference surface)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, a nested-difference surface consists of displayed measurable events* $A, B, C \in \mathcal{A}$ *in one carried fiber, displayed inclusions*

$$A \subseteq B \subseteq C,$$

the relative-difference rows $B \setminus A$ *and* $C \setminus B$ *supplied by Definition 92.16, and the disjointness witnesses for the pairs* $A, B \setminus A$ *and* $B, C \setminus B$*. It also records the binary-union readbacks*

$$A \cup (B \setminus A) \sim_M B$$

and

$$B \cup (C \setminus B) \sim_M C,$$

together with their visible $\mathsf{Real}^{\uparrow}$ *endpoints*

$$\mu(A), \quad \mu(B \setminus A), \quad \mu(B), \quad \mu(C \setminus B), \quad \mu(C).$$

Theorem 92.45 (Measure nested-difference readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B \in \mathcal{A}$ *be displayed events in the same carried fiber, and suppose the public event readback displays* $A \subseteq B$*. If the certificate supplies the complement/relative-difference row of Definition 92.16 and the hypotheses of Definition 92.26 for the pair* $A, B \setminus A$*, then*

$$\mu(B) \sim_R \mu(A) + \mu(B \setminus A).$$

Proof. Apply Definition 92.101 to the displayed inclusion $A \subseteq B$. This supplies the measurable event $B \setminus A$, the disjointness witness for A and $B \setminus A$, and the binary-union readback classifying $A \cup (B \setminus A)$ as B.

Apply Definition 92.26 to the displayed disjoint pair A and $B \setminus A$. The row gives

$$\mu(A \cup (B \setminus A)) \sim_R \mu(A) + \mu(B \setminus A).$$

Transport the left endpoint through the binary-union readback and then through the $\mathsf{Real}^{\uparrow}$ value classifier. Composing that transport with the finite disjoint-union comparison yields the stated readback for $\mu(B)$. □

Theorem 92.46 (Measure two-stage nested union readback)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B, C \in \mathcal{A}$ *be displayed events in the same carried fiber, and suppose the public event readback displays*

$$A \subseteq B \subseteq C.$$

If the certificate supplies the complement/relative-difference rows for these two displayed inclusions, the hypotheses of Definition 92.26 for the pairs $A, B \setminus A$ *and* $B, C \setminus B$*, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row, then*

$$\mu(C) \sim_R (\mu(A) + \mu(B \setminus A)) + \mu(C \setminus B).$$

Proof. Apply Definition 92.45 to the displayed inclusion $B \subseteq C$. This gives

$$\mu(C) \sim_R \mu(B) + \mu(C \setminus B).$$

Apply Definition 92.45 to the displayed inclusion $A \subseteq B$. This gives

$$\mu(B) \sim_R \mu(A) + \mu(B \setminus A).$$

Use additive congruence from the target $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ row to transport the left summand of $\mu(B) + \mu(C \setminus B)$ across the second comparison. Composing with the first comparison gives

$$\mu(C) \sim_R (\mu(A) + \mu(B \setminus A)) + \mu(C \setminus B),$$

with the displayed parenthesizing read from the associativity row of Definition 42.1. □

Theorem 92.47 (Measure nested-difference package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *nested-difference surface packages the following rows as the finite nested-inclusion surface consumed by* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*:*

1. *a displayed inclusion pair* $A \subseteq B$ *with the event* $B \setminus A$*;*
2. *the disjoint readback* $A \cup (B \setminus A) \sim_M B$*;*
3. *the endpoint readback* $\mu(B) \sim_R \mu(A) + \mu(B \setminus A)$*;*
4. *a displayed inclusion chain* $A \subseteq B \subseteq C$ *with the two rows* $B \setminus A$ *and* $C \setminus B$*;*
5. *the two-stage endpoint readback* $\mu(C) \sim_R (\mu(A) + \mu(B \setminus A)) + \mu(C \setminus B)$*.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureNestedDifference package`

Proof. The first two fields are precisely the projections of Definition 92.101 for the displayed pair $A \subseteq B$.

The third field is Definition 92.45. It composes the same relative-difference readback with finite disjoint-union additivity.

For a chain $A \subseteq B \subseteq C$, apply the preceding field to $A \subseteq B$ and to $B \subseteq C$. Additive congruence and associativity in the target $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ row then give Definition 92.46. These rows are exactly the finite nested-inclusion data needed by the downstream measure consumers. □

Definition 92.48 (Concrete zero BHist measure surface)**.** *The concrete zero* BHist *measure surface is the following* $\mathsf{Measure}^{\uparrow}$ *carrier instance over the BEDC history substrate. Its carrier row is*

$$\mathsf{ZeroMeasureCode}(h) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Its carrier classifier is

$$h \sim_0 k \iff \mathsf{hsame}(h, k).$$

The displayed event inventory contains exactly the singleton empty-event row: an event presentation is measurable precisely when its visible BHist *event code is classified with* e*. The measure endpoint is the constant* $\mathsf{Real}^{\uparrow}$ *zero endpoint,*

$$\mu_0(E) := 0_R,$$

for every displayed measurable event E. The countable-sum endpoint for every measurable sequence is also the constant endpoint 0_R.

The only ambient rows used by this surface are BHist*,* hsame*, the classifier and zero row of* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, and the additive* $\mathsf{AbGroup}^{\uparrow}$ *row of Definition 42.1 for zero, addition congruence, and the additive zero laws.*

Theorem 92.49 (Zero BHist measure carrier-classifier stability)**.** *The concrete zero surface of Definition 92.48 satisfies the carrier transport, event-row transport, and endpoint transport rows required by the* $\mathsf{Measure}^{\uparrow}$ *carrier classifier. If* $\mathsf{ZeroMeasureCode}(h)$ *and* $h \sim_0 k$*, then* $\mathsf{ZeroMeasureCode}(k)$*. If E is a displayed measurable event over h and E' is transported along the same* hsame *comparison, then E' is again the singleton empty-event row over k. The endpoint comparison is*

$$\mu_0(E) \sim_R \mu_0(E').$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist carrier classifier stability`

Proof. Unpack the carrier row. The assumption $\mathsf{ZeroMeasureCode}(h)$ identifies h with e, while $h \sim_0 k$ is exactly an $\mathsf{hsame}(h, k)$ comparison. Symmetry and transitivity of hsame identify k with e, so the transported carrier row is again $\mathsf{ZeroMeasureCode}(k)$.

For event-row transport, the only displayed measurable row is the empty-event row. Transporting an empty event presentation along an hsame comparison therefore has only one possible visible target row: the same empty-event row read in the k-fiber. No quotient of events or external set equality is used.

For endpoint transport, both endpoint projections reduce by definition to 0_R. The reflexive classifier row of $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ classifies 0_R with itself, giving $\mu_0(E) \sim_R \mu_0(E')$. □

↪ variant `BEDC.Derived.MeasureUp.MeasureZeroBHist carrier classifier stability`

Theorem 92.50 (Zero BHist measure sigma-additivity)**.** *For the concrete zero surface of Definition 92.48, every displayed pairwise-disjoint measurable sequence $E_\bullet$ has countable union classified with the same singleton empty-event row, and the measure comparison is*

$$\mu_0\left(\bigcup_{n:\mathbb{N}} E_n\right) \sim_R \sum_{n=0}^{\infty} \mu_0(E_n).$$

Here the countable-sum endpoint on the right is the constant endpoint 0_R. The head-tail readback for this endpoint is classified by the additive $\mathsf{AbGroup}^{\uparrow}$ *zero laws.*

Proof. Every displayed measurable term in the zero surface is the empty-event row. Thus each E_n is classified with e, and the visible countable union of the displayed sequence is again classified with e. The pairwise disjointness premise adds no extra branch, because the only event row on either side of an intersection is empty.

The measure endpoint of the union is 0_R by definition. Each summand $\mu_0(E_n)$ is also 0_R, and the countable-sum endpoint selected by the surface is the constant 0_R. Reflexivity of the $\mathsf{Real}^{\uparrow}$ classifier supplied by $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ therefore gives the displayed sigma-additivity comparison.

It remains to read the head-tail row. The head endpoint is 0_R, and the tail endpoint is again 0_R. The additive $\mathsf{AbGroup}^{\uparrow}$ row classifies $0_R + 0_R$ with 0_R, and classifier symmetry gives the required comparison between the constant sum endpoint and its head-tail expression. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist sigma additivity`

Theorem 92.51 (Zero BHist MeasureUp NameCert package)**.** *The concrete zero surface of Definition 92.48, together with Definition 92.49 and Definition 92.50, forms a concrete* $\mathsf{Measure}^{\uparrow}$ *NameCert obligation package by the packaging theorem Definition 92.27.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist semantic name certificate`

Proof. Project the concrete surface rows. The carrier row is $\mathsf{ZeroMeasureCode}$, the classifier row is $\sim_0$, the measurable-event inventory is the singleton empty-event row, the measure endpoint is the constant 0_R map, and the countable-sum endpoint is the constant 0_R row.

The carrier-classifier theorem supplies the transport fields required by the package: carried histories move along hsame, measurable rows remain the empty-event row, and endpoints compare by the reflexive $\mathsf{Real}^{\uparrow}$ zero classifier. The sigma-additivity theorem supplies the countable-union row and the head-tail readback row, with additive compatibility read from the $\mathsf{AbGroup}^{\uparrow}$ zero laws.

These are exactly the rows consumed by Definition 92.27. Repacking them through that theorem gives a concrete $\mathsf{Measure}^{\uparrow}$ obligation package whose visible carrier is the zero BHist surface. □

Theorem 92.52 (Zero BHist relative-difference zero row)**.** *For the concrete zero* BHist *measure surface of Definition 92.48, let h be a carried history with* $\mathsf{ZeroMeasureCode}(h)$*. Let A and B be displayed measurable events in the h-fiber, and suppose the public event readback displays $B \subseteq A$. Then the zero surface realizes the complement/relative-difference row for this inclusion by the singleton empty-event row:*

$$A \setminus B \quad \textit{is displayed by the event code } \mathsf{e}.$$

Consequently $A \setminus B$ is accepted as a displayed measurable event of the zero surface, the row supplies the disjointness witness for B and $A \setminus B$ and the binary-union readback classifying $B \cup (A \setminus B)$ with A, and the visible endpoint is classified as zero:

$$\mu_0(A \setminus B) \sim_R 0_R.$$

Proof. Unpack Definition 92.48. The measurable-event inventory of the zero surface has exactly one displayed row: an event presentation is measurable precisely when its visible BHist event code is classified with e. Since A and B are displayed measurable events in the same zero fiber, both of their visible event codes are therefore classified with e.

Assign the relative-difference output for the displayed inclusion $B \subseteq A$ to the same singleton empty-event row. This is accepted by the measurable-event inventory just read. The disjointness witness is the empty-row intersection of B with that singleton output: the B input is already classified with e, and the output is classified with e.

The binary-union readback is obtained by reading the union of two displayed empty rows as the empty row and then composing, by symmetry and transitivity of the zero event classifier, with the displayed comparison classifying A with that same row. This realizes the concrete complement/relative-difference field described in Definition 92.16 for the zero surface.

The endpoint projection of Definition 92.48 is constant: for every displayed measurable event E, $\mu_0(E) := 0_R$. Applying this to the accepted output $A \setminus B$ gives the visible endpoint 0_R. Reflexivity of the $\mathsf{Real}^{\uparrow}$ classifier supplied by $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ classifies 0_R with itself, hence

$$\mu_0(A \setminus B) \sim_R 0_R.$$

□

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist relative difference zero row`

Definition 92.53 (Zero BHist MeasureUp exactness surface)**.** *The zero* BHist *exactness surface for* $\mathsf{Measure}^{\uparrow}$ *is the following ten-field concrete surface.*

1. *carrier row:* $\mathsf{ZeroMeasureCode}(h) \iff \mathsf{hsame}(h, \mathsf{e})$*;*
2. *carrier classifier:* $h \sim_0 k$ *is the displayed* $\mathsf{hsame}(h, k)$ *row;*
3. *measurable-event inventory: the accepted event row is exactly the singleton empty-event row whose visible event code is classified with* e*;*
4. *empty ledger: the empty-event membership witness is attached to that singleton row;*
5. *union ledger: every binary-union, countable-union, and disjointness witness accepted by the surface is attached to the same singleton row;*
6. *value endpoint: the measure endpoint of every accepted event is the constant* $\mathsf{Real}^{\uparrow}$ *zero endpoint* 0_R*;*
7. *countable-sum endpoint: the countable-sum endpoint of every accepted measurable sequence is the constant* $\mathsf{Real}^{\uparrow}$ *endpoint* 0_R*;*
8. *countable-sum classifier: pointwise zero endpoint classification transports accepted countable-sum endpoints through* $\sim_R$*;*
9. *codomain package: endpoint reflexivity, symmetry, transitivity, and zero classification are read from* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*;*
10. *additive package: head-tail readback and zero-sum compatibility are read from the additive* $\mathsf{AbGroup}^{\uparrow}$ *zero laws for* 0_R*.*

No row in this surface contains an event carrier other than BHist*, a classifier other than* hsame *and* $\sim_R$*, or a countable-sum witness other than the visible constant-zero endpoint row.*

Theorem 92.54 (Zero BHist event row coverage)**.** *For the zero* BHist *exactness surface of Definition 92.53, every accepted measurable event row, empty-event row, binary-union row, countable-union row, disjointness row, value row, and countable-sum row is the visible singleton empty-event ledger row together with the constant endpoint* 0_R*.*

Proof. Unpack the carrier and measurable-event inventory. A carried history satisfies ZeroMeasureCode exactly when it is classified with e, and the only accepted event presentation is the singleton empty-event row in that fiber.

The empty-event row is therefore already covered. A binary-union row or a countable-union row has no second visible event code to select: each input is the empty-event row, and the accepted output is again the empty-event row. The associated disjointness witness is attached to this same visible row.

For value rows, the endpoint field sends every accepted event to 0_R. For countable-sum rows, the endpoint field sends every accepted sequence to the constant endpoint 0_R. Thus every row consumed by the zero surface is read from the singleton empty-event ledger row and its attached $\mathsf{Real}^{\uparrow}$ zero endpoint. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist event row coverage`

Theorem 92.55 (Zero BHist countable-sum classifier stability)**.** *For the zero* BHist *exactness surface of Definition 92.53, let $E_\bullet$ and $F_\bullet$ be accepted measurable sequences. If every endpoint comparison is the pointwise zero classification*

$$\mu_0(E_n) \sim_R \mu_0(F_n),$$

then their countable-sum endpoints transport through the $\mathsf{Real}^{\uparrow}$ *classifier:*

$$\Sigma_0(E_\bullet) \sim_R \Sigma_0(F_\bullet).$$

Both endpoints on the displayed comparison are the constant endpoint 0_R.

Proof. Read the countable-sum endpoint field of Definition 92.53. For every accepted measurable sequence, the selected countable-sum endpoint is the constant $\mathsf{Real}^{\uparrow}$ zero endpoint 0_R.

The pointwise hypothesis says that each visible summand endpoint is classified with the corresponding zero endpoint in the other sequence. Since the aggregate endpoint field ignores no hidden analytic completion and always selects 0_R, the target aggregate endpoint is also 0_R.

The comparison is therefore the reflexive classifier row of $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ at 0_R. This gives $\Sigma_0(E_\bullet) \sim_R \Sigma_0(F_\bullet)$ and supplies the countable-sum classifier stability row for the zero surface. □

Theorem 92.56 (Zero BHist concrete exactness package)**.** *The zero* BHist *exactness surface of Definition 92.53, together with Definition 92.54, Definition 92.55, Definition 92.49, Definition 92.50, and Definition 92.51, forms a concrete* $\mathsf{Measure}^{\uparrow}$ *root-unblock package for the zero* BHist *carrier.*

Proof. First project the surface fields. The carrier is ZeroMeasureCode, the carrier classifier is hsame, the event inventory is the singleton empty-event row, and the value and countable-sum endpoints are both the constant endpoint 0_R.

The event-row coverage theorem exhausts the visible ledger: measurable events, unions, disjointness rows, value rows, and countable-sum rows are all attached to the singleton empty-event row. The countable-sum classifier theorem supplies the endpoint transport row for accepted measurable sequences.

The carrier-classifier theorem supplies transport of carried histories and measure endpoints. The sigma-additivity theorem supplies the countable-union comparison and the head-tail zero readback through the additive $\mathsf{AbGroup}^{\uparrow}$ laws. The NameCert package theorem repacks those rows as the concrete $\mathsf{Measure}^{\uparrow}$ obligation package.

Combining these projections gives a single root surface that downstream $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ constructions may cite: event coverage, countable-sum classifier stability, sigma-additivity, carrier stability, and NameCert packaging are all exposed by the same concrete zero BHist carrier. □

Definition 92.57 (Measure event row coverage)**.** *For the* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra carrier surface of Definition 92.14, the event row coverage predicate is the assertion that every row consumed by a downstream certificate is attached to a visible* BHist *ledger row of that carrier surface. Concretely, the covered rows are:*

1. *every measurable event read by the measurable-family predicate* $\mathcal{A}$*;*
2. *the empty-event row;*
3. *every binary-union row, including its disjointness witness when the union is read as a disjoint union;*
4. *every countable-union row for a StreamName-style measurable family, together with its pairwise-disjointness witness when sigma-additivity is invoked;*
5. *every* $\mathsf{Real}^{\uparrow}$ *endpoint row* $\mu(E)$ *classified by the measure-value classifier;*
6. *every countable-sum endpoint* $\Sigma_{\mu}(E_{\bullet})$ *and every head-tail readback row attached to that endpoint.*

Coverage requires the attachment to be exact: the downstream consumer reads the displayed BHist *row, its* hsame *transport row, and the associated* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row, and it reads no additional event carrier, choice object, quotient object, or analytic completion object from the* $\mathsf{Measure}^{\uparrow}$ *package.*

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureEventRowCoverage`

Theorem 92.58 (Measure measurable family exactness)**.** *For the* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra carrier surface, the event row coverage predicate of Definition 92.57 gives coverage, soundness, and completeness for the measurable-family classifier. Coverage means that every measurable event, closure event, disjointness witness,* $\mathsf{Real}^{\uparrow}$ *endpoint, and countable-sum endpoint used by the classifier is read from a displayed ledger row. Soundness means that transporting a covered row along the classifier gives the corresponding covered row in the target presentation. Completeness means that every row available to the measurable-family classifier is one of the rows listed in the coverage predicate.*

Proof. Unpack Definition 92.14. Its carrier, stability, ledger, and exactness fields list exactly the measurable-event, empty-event, binary-union, countable-union, disjointness, endpoint, and countable-sum rows named in Definition 92.57.

Coverage is therefore projection from the displayed carrier and ledger fields. Each consumed event or endpoint row is attached to one of those fields by the exactness clause of the surface.

Soundness is the transport statement of Definition 92.9. Componentwise hsame moves the event rows, and the $\mathsf{Real}^{\uparrow}$ classifier moves the endpoint rows. The ledger witness attached to a visible row is repacked after that visible row has been transported.

Completeness is the exhaustion statement of Definition 92.13. Since the ledger contains only the listed rows and every hidden measurable-set or countable-sum witness is attached to one of them, the measurable-family classifier has no further row to read. □

Theorem 92.59 (Measure probability normalization row)**.** *Let a covered* $\mathsf{Measure}^{\uparrow}$ *package carry a total event row* Ω *whose endpoint is classified as the probability unit:*

$$\mu(\Omega) \sim_R 1_R.$$

Together with the empty-zero theorem Definition 92.25 and the finite disjoint-union theorem Definition 92.26, this row supplies the normalization and finite-additivity rows consumed by $\mathsf{ProbSpace}^{\uparrow}$*.*

The claim is only the projection of the total-event endpoint and the repacking of the two named Measure$^{\uparrow}$ *consequence theorems into the probability-space certificate surface.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureProbabilityNormalizationRow`

Proof. Project the total-event ledger row from the covered measure package. Its endpoint comparison is already the normalization row required by ProbSpace$^{\uparrow}$, namely $\mu(\Omega) \sim_R 1_R$.

Next project the empty event, binary disjoint-union, endpoint, and additive AbGroup$^{\uparrow}$ rows exposed by Definition 92.57. With those rows in hand, Definition 92.25 gives the zero endpoint for the empty event.

Finally apply Definition 92.26 to a covered disjoint measurable pair. The result is exactly the finite-additivity comparison used by the probability-space carrier; no sigma-additivity theorem beyond the named Measure$^{\uparrow}$ package rows is imported into ProbSpace$^{\uparrow}$. □

Theorem 92.60 (Measure integral premeasure consumption)**.** *The* Integral$^{\uparrow}$ *premeasure interface may consume exactly the following rows from the* Measure$^{\uparrow}$ *package: covered measurable-event rows,* Real$^{\uparrow}$ *endpoint classifier rows, countable-sum endpoint rows, classifier-stable head-tail readback rows, and the finite disjoint-union theorem Definition 92.26. The same exposed endpoint classifier and additive compatibility rows are the MeasureUp rows available to* FunctionalAnalysis$^{\uparrow}$*. These rows are precisely the public surface supplied by Definition 92.15, Definition 92.22, and Definition 92.27.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureIntegralPremeasure consumption`

Proof. Begin with the public certificate surface of Definition 92.15. Its projections provide the measurable-family predicate, endpoint map, countable-sum endpoint, head-tail readback row, and countable-sum classifier row.

The certificate obligation theorem Definition 92.22 identifies the sigma-algebra rows and the sigma-additivity rows that make those projections available to premeasure constructions. In particular it names the empty, binary disjoint-union, countable disjoint-union, sigma-additivity, and head-tail rows as the complete certificate rows used by the measure consequences.

The NameCert package theorem Definition 92.27 repacks those rows as the surface consumed by downstream analysis chapters. Reading that package through Definition 92.57 leaves exactly the measurable-event, endpoint, countable-sum, head-tail, and finite disjoint-union rows named in the statement, with no additional carrier row available for Integral$^{\uparrow}$ or FunctionalAnalysis$^{\uparrow}$. □

Definition 92.61 (Measure root NameCert public surface)**.** *The root* Measure$^{\uparrow}$ *NameCert public surface is the ten-field carrier surface of Definition 92.14, read together with the certificate surface of Definition 92.15, the exact event-row coverage predicate of Definition 92.57, and the no-*StandardBridge *row recorded in the standard-bridge field of the carrier surface. Concretely, its public rows are:*

1. *the traditional constructive measure-space target;*
2. *the BEDC source rows over* BHist*,* NameCert$_{\text{Real}^{\uparrow}}$*, and the additive* AbGroup$^{\uparrow}$ *real row;*
3. *the visible carrier rows for measurable events, empty event, binary union, countable union, complement/relative-difference, disjoint sequence readback,* Real$^{\uparrow}$ *endpoints, and countable sums;*
4. *the componentwise* hsame *classifier, with* Real$^{\uparrow}$ *endpoint comparison supplied by the real classifier;*
5. *the stability rows for every displayed event, value, and countable-sum component;*

6. *the ledger rows attached to hidden membership, relative-difference, and countable-sum witnesses;*
7. *the core consequence rows named in the carrier surface;*
8. *the exactness predicate asserting that downstream consumers read only the displayed rows;*
9. *the explicit absence of a standard bridge at this paper site;*
10. *the public status row that exposes the resulting package as the* $\mathsf{Measure}^{\uparrow}$ *NameCert obligation surface.*

Theorem 92.62 (Measure root carrier-classifier public obligation)**.** *The root* $\mathsf{Measure}^{\uparrow}$ *NameCert public surface of Definition 92.61 supplies the public carrier and classifier transport rows required by the* $\mathsf{Measure}^{\uparrow}$ *obligation surface. For any two presented histories* $h, k : \mathsf{BHist}$ *in the surface, a carried* $\mathsf{Measure}^{\uparrow}$ *code at* h*, and a public classifier comparison* $h \sim_M k$*, the surface transports the carrier code, each visible event row, each ledger row, and each* $\mathsf{Real}^{\uparrow}$ *endpoint row from* h *to* k*.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRootDownstreamUnblockPackage`

Proof. Begin with the root public surface and project its carrier and classifier fields. These fields are exactly the carrier slots and componentwise classifier slots listed in Definition 92.14.

Apply Definition 92.9. It transports the displayed measurable-event, empty-event, binary-union, countable-union, relative-difference, disjointness, endpoint, and countable-sum rows along the public classifier. The $\mathsf{Real}^{\uparrow}$ endpoint movement is read through the displayed real classifier, while the event rows move by their hsame components.

It remains to expose the stability rows as public obligation rows. Apply Definition 92.21. Its projections identify the same classifier-stable rows at the certificate level, so the transported carrier, classifier, ledger, and endpoint data repack as the root carrier-classifier public obligation. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRootCarrierClassifier public obligation`

Theorem 92.63 (Measure root sigma-additive public obligation)**.** *The root* $\mathsf{Measure}^{\uparrow}$ *NameCert public surface supplies the public sigma-additive, empty-zero, finite readback, and relative-difference rows. For every covered pairwise-disjoint measurable sequence* $E_\bullet$ *with countable union* U*, the surface supplies*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

It also supplies $\mu(\varnothing) \sim_R 0_R$*, finite disjoint-union readback for covered disjoint pairs, and relative-difference additivity for covered inclusions of measurable events in the same fiber.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRootSigmaAdditive public obligation`

Proof. Project the sigma-algebra, countable-union, head-tail, countable-sum classifier, and additive $\mathsf{AbGroup}^{\uparrow}$ rows from the root public surface. The coverage predicate ensures that the measurable sequence, its union, its disjointness witness, and the resulting endpoints are all visible BHist ledger rows.

Apply Definition 92.22 to read the certificate rows as the public sigma-additive obligation. Its sigma-additivity projection gives the displayed comparison for $E_\bullet$ and U.

Next apply Definition 92.25. The constant-empty measurable sequence is covered by the empty-event and countable-union rows, and the additive $\mathsf{AbGroup}^{\uparrow}$ row supplies the zero and congruence laws used by the theorem.

Apply Definition 92.26 to a covered disjoint pair. The same visible binary-union, endpoint, and head-tail rows give the finite readback comparison.

Finally apply Definition 92.102. The relative-difference row is one of the displayed carrier rows, so the inclusion witness, disjointness witness, binary-union readback, and endpoint comparison repack as the public relative-difference obligation. □

Theorem 92.64 (Measure root downstream unblock package)**.** *The carrier-classifier rows of Definition 92.62 and the sigma-additive rows of Definition 92.63 form the public* $\mathsf{Measure}^{\uparrow}$ *package consumed by* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$. *The package exposes only covered measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, countable-sum rows, classifier-stable head-tail readback, empty-zero, finite disjoint-union readback, relative-difference additivity, and the explicit no-*$\mathsf{StandardBridge}$ *row.*

Proof. Start from the root public surface and apply Definition 92.27. This repacks the carrier, classifier, sigma-additivity, ledger exactness, nonnegative value, monotone inclusion, and measure consequence rows as the public $\mathsf{Measure}^{\uparrow}$ NameCert obligation package.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.59. It consumes the total event endpoint together with the empty-zero and finite disjoint-union rows already exposed by the root sigma-additive public obligation.

For $\mathsf{Integral}^{\uparrow}$ and $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.60. Its consumed rows are exactly the covered measurable-event rows, endpoint classifier rows, countable-sum rows, head-tail readback rows, and finite disjoint-union rows projected above.

The event-row coverage predicate prevents any extra carrier from entering the downstream package. The standard-bridge field is the no-$\mathsf{StandardBridge}$ row from the root surface, so the public package is a NameCert surface only and makes no standard-bridge claim. □

Definition 92.65 (Measure root downstream image surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root downstream image surface is the ten-field public image through which the direct analysis consumers read the root measure package:*

1. *traditional target: a constructive measure root exposed as the common source for integration, probability, and functional-analysis measure rows;*
2. *BEDC source:* BHist, *componentwise* hsame, Cont, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint comparison,* $\mathsf{StreamName}^{\uparrow}$ *countable-family names, and the additive* $\mathsf{AbGroup}^{\uparrow}$ *real rows;*
3. *carrier: the public* $\mathsf{MeasCode}$ *row carried by the displayed* BHist *ledger;*
4. *classifier: the public measure classifier* $\sim_M$, *with endpoint motion read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and* $\sim_R$*;*
5. *stability rows: carrier transport, classifier transport, visible event-row transport, finite and countable family transport, relative-difference transport, monotone-inclusion transport, nonnegative endpoint transport, and continuity-from-below transport;*
6. *ledger: hidden membership, disjointness, endpoint-value, relative-difference, countable-union, and countable-sum witnesses attached only to displayed public* $\mathsf{Measure}^{\uparrow}$ *rows;*
7. *core theorem list: Definition 92.66, Definition 92.67, and Definition 92.68;*
8. *exactness: every* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row displayed here is one of the public rows of Definition 92.193, Definition 92.118, and Definition 92.82;*
9. *standard bridge: no* $\mathsf{StandardBridge}$ *row is asserted, imported, or needed by this image surface;*
10. *status: paper proof target under* $\mathsf{BEDC.Derived.MeasureUp}$.

Theorem 92.66 (Measure root downstream image carrier)**.** *The carrier and classifier rows of Definition 92.65 project to the* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers through the same public* $\mathsf{MeasCode}$*,* $\sim_M$*,* BHist*,* hsame*,* Cont*,* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{AbGroup}^{\uparrow}$ *rows of the root* $\mathsf{Measure}^{\uparrow}$ *package.*

Proof. Start from Definition 92.193. Its carrier and classifier fields display $\mathsf{MeasCode}$, $\sim_M$, the visible BHist ledger, endpoint comparison through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, and countable-family transport through $\mathsf{StreamName}^{\uparrow}$.

Apply Definition 92.197. The exactness theorem identifies the carrier face, additive face, and consumer face as one public root package and excludes host set equality, quotient event equality, analytic completion objects, and standard bridge rows.

Next read the same carrier-classifier package through Definition 92.118. Apply Definition 92.121; it names the three consumer fronts and says that their visible rows are projections of the domain-ready carrier, classifier, endpoint, additive, and ledger rows.

Finally read the analysis front through Definition 92.82. Apply Definition 92.85. The analysis-unblock theorem repacks the same public carrier and classifier rows as the package consumed by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$, so the downstream image has no separate carrier or classifier source. □

Theorem 92.67 (Measure root downstream image additivity)**.** *The finite additivity, countable additivity, relative-difference additivity, monotone inclusion, nonnegative endpoint, and continuity-from-below rows of Definition 92.65 are the same public* $\mathsf{Measure}^{\uparrow}$ *rows seen by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*.*

Proof. Project the additive and order rows from Definition 92.193. Its exactness theorem, Definition 92.197, keeps empty-zero, finite readback, relative-difference, countable disjointness, countable-sum, ledger, and consumer rows inside one visible root package.

Use Definition 92.118 as the domain reading of that package. Apply Definition 92.121. The domain-ready theorem exposes the finite, countable, endpoint, ledger, and consumer rows to the three analysis fronts without adding host or quotient event equality.

Now read the same rows through Definition 92.82. Apply Definition 92.85. It identifies finite readback, countable-sum stability, relative-difference additivity, monotone inclusion, nonnegative endpoint values, and continuity from below as the rows consumed by the direct analysis fronts.

Thus the additivity, order, endpoint, and continuity fields of the downstream image are not parallel copies. They are the common public $\mathsf{Measure}^{\uparrow}$ rows already displayed by the root basis, the domain-ready package, and the analysis-unblock surface. □

Theorem 92.68 (Measure root downstream image consumer)**.** *Every* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row exposed by Definition 92.65 factors through the displayed public* $\mathsf{Measure}^{\uparrow}$ *image. The image adds no host set equality, no quotient event equality, no analytic completion object, and no standard bridge row.*

Proof. Begin with the consumer face of Definition 92.193. Apply Definition 92.197; the theorem exhausts the public carrier, additive, and consumer faces and leaves no slot for host set equality, quotient event equality, analytic completion, or a $\mathsf{StandardBridge}$ row.

Next pass through Definition 92.118. Apply Definition 92.121. The theorem lists the $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ projections as the downstream face of the domain-ready root package and keeps every consumed row attached to the displayed carrier, classifier, endpoint, additive, ledger, and exactness fields.

Finally pass through Definition 92.82. Apply Definition 92.85. This analysis-unblock reading states that the three direct consumers use only the carrier, classifier, stability, exactness, ledger, additivity, order, endpoint, and continuity rows named in the public surface.

Combining the three readings gives the desired factorization: each direct analysis consumer row exposed by the downstream image is already a row of the public $\mathsf{Measure}^{\uparrow}$ image. Since none of the cited packages contains the excluded equality, completion, or bridge fields, the downstream image adds none of them. □

Definition 92.69 (MeasureUp analysis-seed source)**.** *The* $\mathsf{Measure}^{\uparrow}$ *analysis-seed source is the BEDC source package obtained from* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *together with the* $\mathsf{Measure}^{\uparrow}$ *carrier and classifier of section 92.1. Its visible source rows are the measurable carrier, the* BHist *classifier* $\sim_M$*, the designated* σ*-algebra and measurable-family rows, the sigma-additivity row, the nonnegative* $\mathsf{Real}^{\uparrow}$ *value row, the finite-additivity readback package, the relative-difference package, the zero-*BHist *package, and the ledger rows listed in Definition 92.57. A consumer of this source reads only those rows and the classifier transport supplied by Definition 92.27; it does not receive a host set carrier, quotient equality of events, or an analytic completion object.*

Theorem 92.70 (MeasureUp integral substrate obligations)**.** *For the source of Definition 92.69, an* $\mathsf{Integral}^{\uparrow}$ *consumer may use only the covered measurable-event rows, the* $\mathsf{Real}^{\uparrow}$ *endpoint values, the finite and countable additivity readbacks, the relative-difference rows, and classifier transport along* $\sim_M$ *and* $\sim_R$*. These obligations are exactly the* $\mathsf{Measure}^{\uparrow}$ *rows exposed by the NameCert package, the integral premeasure consumption theorem, and the two-step finite-additivity package.*

Proof. Start with Definition 92.27. It packages the carrier, classifier transport, sigma-additivity, nonnegative endpoint, relative-difference, ledger exactness, and measure consequence rows as the public $\mathsf{Measure}^{\uparrow}$ surface.

The integral consumer then reads this surface through Definition 92.60. That theorem restricts the premeasure interface to covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, countable-sum endpoints, classifier-stable head-tail rows, and the finite disjoint-union consequence.

Finally, Definition 92.35 supplies the finite-additivity readback named in the statement. The relative-difference rows are already part of the same NameCert package, so no additional host event equality or quotient transport is imported into $\mathsf{Integral}^{\uparrow}$. □

Theorem 92.71 (MeasureUp probability substrate obligations)**.** *For the source of Definition 92.69, a* $\mathsf{ProbSpace}^{\uparrow}$ *consumer adds only a total-event row* Ω *and a unit-mass endpoint row* $\mu(\Omega) \sim_R 1_R$ *on top of the* $\mathsf{Measure}^{\uparrow}$ *substrate. The empty-event zero row and finite disjoint-union row remain the same* $\mathsf{Measure}^{\uparrow}$ *consequences; no probability-specific carrier, quotient of events, or independent additivity principle is added.*

Proof. Project the total-event row and unit-mass endpoint row through Definition 92.59. That theorem states that these two rows are the normalization data consumed by $\mathsf{ProbSpace}^{\uparrow}$.

The empty-event row is the measure consequence Definition 92.25. It supplies the zero endpoint for $\varnothing$ from the existing sigma-additivity and countable-sum rows.

The finite-additivity row is Definition 92.26. Applying it to each covered disjoint measurable pair gives exactly the finite probability readback, so the probability surface is the MeasureUp substrate plus the total-event and unit-mass rows named in the statement. □

Theorem 92.72 (MeasureUp functional-analysis substrate obligations)**.** *For the source of Definition 92.69,* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *and* L^p*-style consumers receive only nonnegative* $\mathsf{Real}^{\uparrow}$ *endpoint rows, monotone inclusion rows, relative-difference additivity rows, and countable-sum classifier*

stability rows. These rows are read through $\mathsf{Measure}^{\uparrow}$ *classifier transport and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*; they do not import host sets, quotient equality of events, or any independent analytic equality principle.*

Proof. The nonnegative endpoint row is Definition 92.98. It is already a $\mathsf{Real}^{\uparrow}$-classified endpoint row of the measure package.

The order-facing row is Definition 92.99. It reads inclusion only between covered measurable events and returns the corresponding $\mathsf{Real}^{\uparrow}$ endpoint comparison.

The additive decomposition row is Definition 92.102. It supplies the relative-difference comparison used by L^p-style arguments without replacing event presentations by host equality.

The countable endpoint row is Definition 92.12. It transports countable sums by pointwise $\mathsf{Real}^{\uparrow}$ classifier comparison, which is the countable-sum stability available to functional-analysis consumers. □

Definition 92.73 (MeasureUp analysis-unblock source)**.** *The* $\mathsf{Measure}^{\uparrow}$ *analysis-unblock source is the source package over* BHist *obtained by reading the* $\mathsf{Measure}^{\uparrow}$ *carrier and classifier of section 92.1 through* hsame *and through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*. Its visible rows are the event-row coverage predicate of Definition 92.57, the relative-difference row of Definition 92.16, the finite-additivity rows of Definition 92.35, the continuity-from-below rows of Definition 92.42, and the analysis-seed substrate rows of Definition 92.69. A downstream consumer of this source receives only these public rows, their* hsame *transport, and their* $\mathsf{Real}^{\uparrow}$ *endpoint classifier comparisons.*

Theorem 92.74 (MeasureUp analysis-unblock carrier-classifier scope)**.** *The source of Definition 92.73 supplies the* $\mathsf{Measure}^{\uparrow}$ *carrier, the public* BHist *classifier,* hsame *transport of visible event rows, event-row coverage, and countable-sum classifier stability. These rows are exactly the carrier-classifier and countable endpoint rows exposed by the root* $\mathsf{Measure}^{\uparrow}$ *NameCert obligations.*

Proof. Start from the root public surface of Definition 92.61. Project its carrier, classifier, event-row coverage, ledger, and endpoint fields.

Apply Definition 92.62. This transports the carrier code, every displayed event row, and every $\mathsf{Real}^{\uparrow}$ endpoint row along the public BHist classifier, using the componentwise hsame fields of the carrier surface.

The event coverage is the exact predicate of Definition 92.57. The countable-sum endpoint movement is the stability row of Definition 92.12.

Finally repack these projections through Definition 92.27. The result is precisely the carrier, classifier, hsame transport, coverage, and countable-sum classifier scope named in the statement. □

Theorem 92.75 (MeasureUp analysis-unblock relative-continuity package)**.** *The source of Definition 92.73 supplies one analysis package containing finite disjoint-union additivity, relative-difference additivity, monotonicity under displayed measurable inclusion, nonnegative* $\mathsf{Real}^{\uparrow}$ *measure values, and continuity from below for displayed increasing measurable sequences.*

Proof. Project finite additivity first. The binary finite row is Definition 92.26, and the ordered two-step finite package is Definition 92.35.

Next project the relative-difference and order rows. Apply Definition 92.102 for the additive decomposition of a displayed inclusion, apply Definition 92.99 for the induced $\mathsf{Real}^{\uparrow}$ order comparison, and apply Definition 92.98 for the nonnegative endpoint row.

For increasing sequences, apply Definition 92.42. Its layer construction uses the same relative-difference rows, finite readbacks, sigma-additivity, and countable-sum classifier stability already projected above.

These rows have the same carrier, event classifier, and $\mathsf{Real}^{\uparrow}$ endpoint classifier because they are all read through Definition 92.27. Repacking them gives one relative-continuity analysis package over the source of Definition 92.73. □

Theorem 92.76 (MeasureUp analysis-unblock downstream source)**.** *The source of Definition 92.73 assembles the integral, probability, and functional-analysis substrate rows consumed by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. The assembled source consists of the integral premeasure rows, the probability normalization rows, and the functional-analysis endpoint and order rows already exposed by the* $\mathsf{Measure}^{\uparrow}$ *analysis substrate.*

Proof. For $\mathsf{Integral}^{\uparrow}$, project Definition 92.70. It supplies covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint values, finite and countable additivity readbacks, relative-difference rows, and classifier transport.

For $\mathsf{ProbSpace}^{\uparrow}$, project Definition 92.71. It adds only the total-event row and the unit-mass endpoint row to the same $\mathsf{Measure}^{\uparrow}$ substrate.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, project Definition 92.72. It consumes the nonnegative endpoint row, monotone inclusion row, relative-difference additivity row, and countable-sum classifier stability row.

The common source boundary is Definition 92.69, and the public downstream surface is Definition 92.64. Therefore the three downstream consumers read a single assembled $\mathsf{Measure}^{\uparrow}$ analysis-unblock source, with no host set equality, quotient event equality, or standard bridge row added at this paper site. □

Definition 92.77 (Measure StreamName endpoint family)**.** *A* $\mathsf{Measure}^{\uparrow}$ *StreamName endpoint family over the sigma-algebra carrier surface of Definition 92.14 is a certified observation map* $s : \mathsf{Hist} \to \mathsf{Hist}$ *together with a displayed* $\mathsf{RatStreamNameCarrier}(s)$ *row as in Definition 126.1. For every unary observation history* n *with* $\mathsf{UnaryHistory}(n)$*, the point observation* $s(n)$ *carries:*

1. *a* $\mathsf{Measure}^{\uparrow}$ *measurable-event row covered by Definition 92.57;*
2. *the associated* $\mathsf{Real}^{\uparrow}$ *endpoint row* $\mu(s(n))$ *read through the endpoint classifier of Definition 38.9;*
3. *the pointwise StreamNameUp classifier row supplied by Definition 126.2;*
4. *the countable-sum endpoint row attached to the resulting StreamName-indexed measurable family.*

The family exposes no host-level equality of event carriers. Its public data are exactly the unary BHist *observation, the covered measurable event, its* hsame *transport row, and the corresponding* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row.*

Theorem 92.78 (Measure StreamName endpoint carrier-classifier)**.** *Every* $\mathsf{Measure}^{\uparrow}$ *StreamName endpoint family supplies carrier transport for the StreamName observation map and pointwise classifier transport for its measurable-event and* $\mathsf{Real}^{\uparrow}$ *endpoint rows. The transport is the common projection of Definition 126.1, Definition 126.2, Definition 126.13, and Definition 92.9.*

Proof. Project the $\mathsf{RatStreamNameCarrier}(s)$ row from Definition 92.77. For a unary observation n, Definition 126.1 reads $s(n)$ as the carried point history, and Definition 126.2 identifies classifier transport pointwise at the same unary observation.

Apply Definition 126.13. Its endpoint-carrier and transported-classifier fields move the stream carrier and the pointwise classifier row to the transported observation map.

Now apply Definition 92.9 at that point observation. The covered measurable-event row, the event ledger entry, and the $\mathsf{Real}^{\uparrow}$ endpoint comparison are all componentwise rows of the $\mathsf{Measure}^{\uparrow}$ carrier surface, so they move along the same hsame classifier.

Repack the transported stream carrier, the transported covered event, and the transported endpoint comparison into the endpoint family surface of Definition 92.77. No quotient event equality or host carrier equality is introduced. □

Theorem 92.79 (Measure StreamName endpoint stability)**.** *Let $s, t : \mathsf{Hist} \to \mathsf{Hist}$ be two $\mathsf{Measure}^{\uparrow}$ StreamName endpoint families. If $\mathsf{RatStreamNameClassifier}(s, t)$ holds and the endpoint family rows are transported pointwise by hsame at every unary observation, then measurable event witnesses and $\mathsf{Real}^{\uparrow}$ endpoint comparisons are preserved at every index.*

Proof. Fix a unary observation n with witness $\nu : \mathsf{UnaryHistory}(n)$. The StreamName classifier gives the point classifier between $s(n)$ and $t(n)$ by Definition 126.2.

The endpoint-family hypotheses attach a covered measurable-event row and a $\mathsf{Real}^{\uparrow}$ endpoint row to the source point observation. The pointwise hsame transport row moves the displayed event witness to the target point.

Apply Definition 92.9 to the moved point row. It transports the measurable-event witness and reads the measure value in the target presentation under the same $\mathsf{Real}^{\uparrow}$ endpoint classifier.

Since the observation n was arbitrary, the transported event witnesses and endpoint comparisons assemble pointwise over all unary observations. Repacking them gives stability of the whole StreamName endpoint family. □

Theorem 92.80 (Measure StreamName countable-additivity scope)**.** *For a $\mathsf{Measure}^{\uparrow}$ StreamName endpoint family, the countable-additivity row of Definition 92.2 factors only through the StreamName-indexed endpoint family, the countable-sum classifier stability of Definition 92.12, and the sigma-additivity obligation of Definition 92.11.*

Proof. Read the StreamName endpoint family pointwise. Each unary observation supplies a covered measurable-event row and its $\mathsf{Real}^{\uparrow}$ endpoint row, so the StreamName map gives the measurable sequence consumed by the countable-additivity row.

Apply Definition 92.11 to that displayed sequence, its pairwise-disjointness witness, and its countable-union row. The resulting comparison is exactly the $\mathsf{Real}^{\uparrow}$ classifier relation between the measure of the union and the countable endpoint sum.

If the point endpoints are transported along another StreamName classifier, apply Definition 92.12. It turns the pointwise $\mathsf{Real}^{\uparrow}$ endpoint comparisons into the classifier comparison between the two countable sums.

Thus countable additivity reads no row beyond the StreamName endpoint family, the countable-sum stability row, and the displayed sigma-additivity row of $\mathsf{Measure}^{\uparrow}$. □

Theorem 92.81 (Measure StreamName downstream consumer boundary)**.** *The $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumers of Definition 92.86 read only the StreamName-indexed endpoint rows of Definition 92.77 together with the root consumer rows already listed in Definition 92.86.*

Proof. Project the root consumer package of Definition 92.86. Its visible rows are the covered measurable-event rows, the $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, the countable-sum endpoint rows, the finite readback rows, the relative-difference and order rows, and the no-$\mathsf{StandardBridge}$ boundary.

For $\mathsf{Integral}^{\uparrow}$, the StreamName endpoint family supplies the covered measurable sequence and countable endpoint rows already named by the root consumer package. The finite and head-tail rows are then read from the same package, not from an additional event equality surface.

For $\mathsf{ProbSpace}^{\uparrow}$, the total-event and empty-zero rows are projections of the root package. A StreamName endpoint family may index the measurable rows it uses, but the probability consumer still receives only the normalization, empty-zero, and finite-additivity rows listed there.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, the nonnegative, monotone, relative-difference, and countable-sum classifier rows are exactly the endpoint rows transported pointwise by Definition 92.79. Repacking the three consumer projections gives the stated boundary and adds no quotient event equality, host set equality, or standard bridge row. □

Definition 92.82 (Measure root analysis-unblock obligation surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root analysis-unblock obligation surface is the ten-row template obtained by reading the analysis-unblock source of Definition 92.73, the root public NameCert surface of Definition 92.61, and the analysis-seed source of Definition 92.69 as one public package. Its rows are:*

1. *the* $\mathsf{Measure}^{\uparrow}$ *carrier over* BHist*;*
2. *the public* hsame *classifier for measurable-event rows;*
3. *the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier supplied by* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*;*
4. *the visible event-row coverage predicate;*
5. *the countable endpoint and countable-sum stability rows;*
6. *finite disjoint-union additivity;*
7. *relative-difference additivity for displayed inclusions;*
8. *monotone inclusion and nonnegative* $\mathsf{Real}^{\uparrow}$ *value rows;*
9. *continuity from below for displayed increasing measurable sequences;*
10. *the ledger and downstream exactness rows consumed by* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$.

Theorem 92.83 (Measure root analysis-unblock carrier-classifier)**.** *The surface of Definition 92.82 supplies the* $\mathsf{Measure}^{\uparrow}$ *carrier, the public measurable-event classifier, event-row coverage, and countable endpoint stability rows. These are precisely the carrier-classifier rows exposed by the root public NameCert surface and read by the analysis-unblock source.*

Proof. Project the carrier, classifier, event-row coverage, ledger, and endpoint rows from Definition 92.61. Apply Definition 92.62 to transport the carrier code, every visible event row, every ledger row, and every $\mathsf{Real}^{\uparrow}$ endpoint row along the public classifier.

Next read the same projections through Definition 92.73. Apply Definition 92.74. It identifies the analysis-unblock carrier, public classifier, event-row coverage, and countable-sum endpoint stability rows with the rows already projected from the root public surface.

Repack these common projections in the ten-row template of Definition 92.82. The repacked surface contains no additional host equality, quotient event equality, or standard bridge row, so the carrier-classifier package is the stated public one. □

Theorem 92.84 (Measure root analysis-unblock additivity-continuity)**.** *The surface of Definition 92.82 packages finite disjoint-union additivity, relative-difference additivity, monotone inclusion, nonnegative* $\mathsf{Real}^{\uparrow}$ *values, and continuity from below for displayed increasing measurable sequences.*

Proof. First project the finite rows from the public $\mathsf{Measure}^{\uparrow}$ package. Apply Definition 92.35; its two-step readback supplies the finite disjoint-union additive row used by the root analysis-unblock surface.

Next project the relative and order rows through Definition 92.75. This single package supplies relative-difference additivity, monotone inclusion, and nonnegative $\mathsf{Real}^{\uparrow}$ values over the same carrier and endpoint classifier.

For countable increasing sequences, apply Definition 92.42. Its construction uses the same visible event rows, relative-difference rows, finite readbacks, countable-sum classifier stability, and $\mathsf{Real}^{\uparrow}$ endpoint classifier as the root surface.

The rows cited above are all rows of the ten-row template in Definition 92.82. Repacking them gives one additivity-continuity package and adds no extra equality or standard-bridge assumption. □

Theorem 92.85 (Measure root analysis-unblock downstream package)**.** *The surface of Definition 92.82 is the* $\mathsf{Measure}^{\uparrow}$ *package consumed by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. The consumer boundary consists only of the carrier, classifier, stability, exactness, ledger, additivity, order, endpoint, and continuity rows named in that surface.*

Proof. Apply Definition 92.76. It assembles the integral premeasure rows, the probability normalization rows, and the functional-analysis endpoint and order rows from the public $\mathsf{Measure}^{\uparrow}$ analysis substrate.

Read that assembled source through the ten-row surface of Definition 92.82. The $\mathsf{Integral}^{\uparrow}$ consumer sees the measurable-event, endpoint, finite-readback, and countable-sum rows; the $\mathsf{ProbSpace}^{\uparrow}$ consumer sees the total-event, unit-mass, empty-zero, and finite-additivity rows; the $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer sees the nonnegative, monotone, relative-difference, and countable-stability rows.

The exactness and ledger rows of the surface keep the three downstream fronts inside the displayed $\mathsf{Measure}^{\uparrow}$ package. Therefore the root analysis-unblock surface is the paper-side package consumed by those fronts and carries no additional standard bridge row. □

Definition 92.86 (Measure root consumer package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root consumer package is the root NameCert public surface of Definition 92.61 read together with the downstream package of Definition 92.64. Its consumers receive only covered measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows, countable-sum endpoint rows, head-tail readback, empty-zero, finite disjoint-union readback, relative-difference additivity, monotone inclusion, nonnegative endpoint, and the explicit no-*$\mathsf{StandardBridge}$ *row exposed by that public* $\mathsf{Measure}^{\uparrow}$ *package.*

Theorem 92.87 (Measure integral consumer rows)**.** *The root consumer package of Definition 92.86 supplies* $\mathsf{Integral}^{\uparrow}$ *with exactly the covered measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows exposed by* $\mathsf{Measure}^{\uparrow}$*. No host set equality, quotient event equality, probability normalization row, functional-analysis order row, or standard bridge row is added to the integral consumer surface.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureIntegralConsumer rows`

Proof. Project the root NameCert public surface from Definition 92.86. Its coverage predicate is Definition 92.57, so every event row visible to the consumer is one of the displayed measurable-event rows of the $\mathsf{Measure}^{\uparrow}$ surface.

Apply Definition 92.64. This exposes the covered event rows, $\mathsf{Real}^{\uparrow}$ endpoint rows, countable-sum rows, head-tail readback, and finite disjoint-union readback as the public downstream package.

Now apply Definition 92.60. Its Integral$^{\uparrow}$ projection consumes exactly those rows from the same public package. Repacking that projection with the root consumer package gives the claimed integral consumer surface, and the exactness predicate rules out every row not named in the statement. □

Theorem 92.88 (Measure probability consumer rows)**.** *The root consumer package of Definition 92.86 supplies* ProbSpace$^{\uparrow}$ *with the total-event normalization row together with the empty-zero and finite disjoint-union rows exposed by* Measure$^{\uparrow}$*. No integral countable-sum endpoint row, functional-analysis relative-difference package, quotient event equality, or standard bridge row is added to the probability consumer surface.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureProbabilityConsumer rows`

Proof. Project the empty-zero and finite disjoint-union rows from Definition 92.64. These are already public rows of the root consumer package, because the package includes the root NameCert surface and its downstream repacking.

Apply Definition 92.59. The theorem reads the total-event endpoint and the unit-mass comparison through the same Measure$^{\uparrow}$ public surface, and it consumes the empty-zero and finite disjoint-union rows already projected above.

Repack the total-event normalization row with those two additive rows. The event-row exactness clause of Definition 92.86 leaves no additional event identity, countable-sum endpoint, functional-analysis order, or StandardBridge row in the ProbSpace$^{\uparrow}$ consumer surface. □

Theorem 92.89 (Measure functional-analysis consumer rows)**.** *The root consumer package of Definition 92.86 supplies* FunctionalAnalysis$^{\uparrow}$ *only with the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows exposed by* Measure$^{\uparrow}$*. It does not add integral premeasure rows, probability normalization rows, quotient event equality, host set equality, or a standard bridge row.*

Proof. Start from the Measure$^{\uparrow}$ analysis substrate rows already present in the root consumer package. The nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows are the analysis rows visible through the public root surface.

Apply Definition 92.72. Its FunctionalAnalysis$^{\uparrow}$ projection consumes exactly those four row families: nonnegative endpoints, monotone inclusion, relative-difference additivity, and countable-sum classifier stability.

Finally repack the projection through Definition 92.64. The downstream exactness predicate keeps the consumer boundary at the named row families, so the functional-analysis surface gains neither integral rows, probability normalization rows, quotient event equality, host set equality, nor a StandardBridge row. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureFunctionalAnalysisConsumer rows`

92.4 Countable-name root surface

Definition 92.90 (Measure countable-name root-unblock surface)**.** *The* Measure$^{\uparrow}$ *countable-name root-unblock surface is the ten-field obligation surface obtained by reading the root public* Measure$^{\uparrow}$ *package through the StreamName endpoint family and the downstream consumer rows. Its fields are:*

1. *traditional target: constructive measure names whose visible events are counted by StreamName observations and whose values are classified* $\mathsf{Real}^{\uparrow}$ *endpoints;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, *the history classifier* hsame, *and the root* $\mathsf{NameCert}$ *surfaces cited by Definition 92.61;*
3. *carrier choice: a carried* BHist *measure code containing the visible event rows, measurable witnesses, sigma-algebra rows,* $\mathsf{Real}^{\uparrow}$ *endpoint map, StreamName endpoint stream, and countable-sum rows;*
4. *classifier choice: event-row* hsame *transport together with the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier and the pointwise StreamName classifier;*
5. *stability obligations: carrier transport, measurable-row transport, sigma-additivity transport, and countable-sum classifier stability;*
6. *ledger: hidden event membership rows, measurable witnesses, countable-family indices, disjointness witnesses, countable-union rows, and endpoint stream rows;*
7. *core theorem list: Definition 92.91, Definition 92.92, and Definition 92.93;*
8. *exactness predicate: coverage, soundness, and completeness for the visible event rows named by Definition 92.57;*
9. *standard bridge: no* $\mathsf{StandardBridge}$ *row is asserted at this paper site;*
10. *status: unformalized paper target for the future* $\mathsf{BEDC.Derived.MeasureUp}$ *namespace.*

The surface is only a package of displayed rows. It does not identify event carriers by host set equality and does not quotient measure codes.

Theorem 92.91 (Measure countable-name carrier-classifier obligation)**.** *The countable-name root-unblock surface of Definition 92.90 exports exactly the* $\mathsf{Measure}^{\uparrow}$ *carrier transport, event-row classifier transport, and StreamName endpoint classifier transport needed by downstream analysis chapters. These rows are the projections of Definition 92.8, Definition 92.9, and Definition 92.78.*

Proof. First project the carried measure code from the countable-name surface. The carrier is a BHist measure code, so Definition 92.8 moves the carrier proof and rereads the measurable-family, sigma-algebra, endpoint, sigma-additivity, head-tail, and countable-sum rows at the transported history.

Next project the public classifier row. Apply Definition 92.9; its componentwise hsame projections transport the visible event rows and their measurable witnesses, while the $\mathsf{Real}^{\uparrow}$ classifier transports the corresponding endpoint values.

Finally project the StreamName endpoint stream. Apply Definition 92.78. It transports the observation carrier and the pointwise measurable-event and endpoint rows at every unary index. Repacking these three projections gives the displayed carrier-classifier obligation and adds no host equality or quotient row. □

Theorem 92.92 (Measure countable-name sigma-additivity export)**.** *For every visible pairwise-disjoint* BHist *event stream in the countable-name root-unblock surface, the sigma-additivity comparison read by* $\mathsf{Measure}^{\uparrow}$ *is the same* $\mathsf{Real}^{\uparrow}$ *endpoint comparison read by the StreamName-indexed countable-sum row. The export is obtained from Definition 92.22, Definition 92.80, and Definition 92.12.*

Proof. Project the visible StreamName-indexed event family from Definition 92.90. Its ledger rows provide the unary indices, the covered measurable event at each index, the pairwise-disjointness witness, and the displayed countable-union row.

Apply Definition 92.22 to the same covered family. The theorem reads the certificate-level sigma-additivity row and produces the $\mathsf{Real}^{\uparrow}$ comparison between the measure of the union and the countable endpoint sum.

Now read the family as a StreamName endpoint family. By Definition 92.80, countable additivity factors only through the endpoint stream, the displayed sigma-additivity row, and countable-sum classifier stability.

It remains to compare transported endpoint streams. Apply Definition 92.12; pointwise $\mathsf{Real}^{\uparrow}$ endpoint comparisons assemble into the classifier comparison of the two countable sums. Repacking these rows gives the stated sigma-additivity export for disjoint visible BHist event streams. □

Theorem 92.93 (Measure countable-name analysis consumer package)**.** *The countable-name root-unblock surface supplies the common* $\mathsf{Measure}^{\uparrow}$ *package consumed by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. The package consists of the carrier-classifier rows of Definition 92.91, the sigma-additivity export of Definition 92.92, and the consumer rows of Definition 92.87, Definition 92.88, and Definition 92.89.*

Proof. Begin with the carrier-classifier obligation. It supplies the transported measure code, visible event rows, measurable witnesses, endpoint rows, and StreamName point rows that the three analysis consumers may name.

Next attach the sigma-additivity export. This gives the countable union comparison and countable-sum classifier stability for every visible disjoint event stream in the same BHist ledger.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The consumer receives the covered measurable-event rows, endpoint classifier rows, countable-sum rows, head-tail readback, and finite disjoint-union readback already present in the countable-name surface.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88. The consumer reads the total-event normalization row together with empty-zero and finite disjoint-union rows, all from the same root package.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89. The consumer reads only nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows. Repacking the three projections gives one analysis consumer package and still exposes no host set equality, quotient event equality, or StandardBridge row. □

Definition 92.94 (Measure root deps-ready threshold)**.** *The hub-counted* $\mathsf{Measure}^{\uparrow}$ *root deps-ready threshold package is the public surface of Definition 92.61, the downstream package of Definition 92.64, and the consumer package of Definition 92.86. It is counted at the* $\mathsf{Measure}^{\uparrow}$ *root because it exposes the public rows required before* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *may consume the measure surface.*

Theorem 92.95 (Measure root public event coverage threshold)**.** *The root deps-ready threshold package of Definition 92.94 exposes only covered measurable-event rows, the empty-event row, binary-union rows, countable-union rows, complement and relative-difference rows, disjoint-sequence rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, and countable-sum rows. Every such row is a visible* BHist *ledger row of the public* $\mathsf{Measure}^{\uparrow}$ *surface, and no uncovered event carrier, quotient event equality, host set equality, analytic completion object, or* StandardBridge *row is present in the threshold package.*

Proof. Start from the event-row coverage predicate of Definition 92.57. Its enumerated rows are exactly the measurable events, empty event, binary and countable unions, disjointness witnesses, $\mathsf{Real}^{\uparrow}$ endpoints, countable sums, and head-tail readback rows available to a downstream certificate.

Apply Definition 92.62. This projects the same rows from the root public surface and transports them along the componentwise hsame classifier. The complement and relative-difference rows are read through the carrier surface fields already included in that public obligation.

Finally apply Definition 92.58. Its coverage, soundness, and completeness clauses identify the visible ledger rows as the whole measurable-family interface, so the threshold package contains only the displayed public event and endpoint rows. □

Theorem 92.96 (Measure root public countable-additivity threshold)**.** *For every covered pairwise-disjoint measurable sequence $E_\bullet$ with covered countable union U, the root deps-ready threshold package supplies the comparison*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

The empty-zero row $\mu(\varnothing) \sim_R 0_R$ and finite disjoint-union readback are part of the same public package, because they are consequences of the sigma-additive $\mathsf{Measure}^\uparrow$ *rows exposed at the root threshold.*

Proof. Project the covered sequence, its countable union, its pairwise-disjointness witness, and its countable-sum endpoint from the root deps-ready threshold package. These are public rows by Definition 92.95.

Apply Definition 92.63. Its sigma-additive projection gives the displayed comparison $\mu(U) \sim_R \Sigma_\mu(E_\bullet)$ for the covered pairwise-disjoint sequence.

Next apply Definition 92.25. The empty event and all-empty sequence are covered rows of the same threshold package, so the zero endpoint is read back without adding a separate carrier.

Finally apply Definition 92.26. The binary-union, disjointness, endpoint, and additive $\mathsf{AbGroup}^\uparrow$ rows are already public at the threshold, so the finite disjoint-union comparison is repacked into the same public countable-additivity package. □

Theorem 92.97 (Measure root consumer boundary threshold)**.** *The root deps-ready threshold package of Definition 92.94 is exactly the root* $\mathsf{Measure}^\uparrow$ *surface consumed by* $\mathsf{Integral}^\uparrow$*,* $\mathsf{ProbSpace}^\uparrow$*, and* $\mathsf{FunctionalAnalysis}^\uparrow$*. It exposes the rows named by those three consumers and excludes quotient event equality, host set equality, and standard bridge rows.*

Proof. First project the root consumer package from Definition 92.94. This gives the same public surface and downstream package used by Definition 92.86.

Apply Definition 92.87 to read the $\mathsf{Integral}^\uparrow$ boundary. It consumes exactly the covered measurable-event, $\mathsf{Real}^\uparrow$ endpoint, countable-sum, head-tail, and finite disjoint-union rows from the root package.

Apply Definition 92.88 to read the $\mathsf{ProbSpace}^\uparrow$ boundary. It consumes the total-event normalization row together with the empty-zero and finite disjoint-union rows exposed by the same root package.

Apply Definition 92.89 to read the $\mathsf{FunctionalAnalysis}^\uparrow$ boundary. It consumes only the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows.

The exclusions are common to the three projections: each cited consumer-row theorem rules out quotient event equality, host set equality where relevant, and any StandardBridge row. The threshold package is therefore exactly the shared root surface for these consumers and nothing beyond that surface. □

Theorem 92.98 (Measure nonnegative value obligation)**.** *For a* $\mathsf{Measure}^\uparrow$ *carrier as in section 92.1, every displayed measurable event E has a visible RealAlgOrder row*

$$\mathsf{NonNeg}_R(\mu(E)).$$

If the $\mathsf{Measure}^{\uparrow}$ *classifier transports* E *to a displayed measurable event* E' *and classifies the endpoints by* $\mu(E) \sim_R \mu(E')$*, then the same classifier transport carries* $\mathsf{NonNeg}_R(\mu(E))$ *to* $\mathsf{NonNeg}_R(\mu(E'))$*.*

Proof. Project the nonnegative measure-value clause of the $\mathsf{Measure}^{\uparrow}$ carrier surface at the displayed measurable event E. This supplies the public row $\mathsf{NonNeg}_R(\mu(E))$ attached to the same ledger entry as the endpoint $\mu(E)$.

For classifier transport, use the endpoint comparison supplied by Definition 92.7. The nonnegative row is stable under $\sim_R$ by definition, so the transported endpoint $\mu(E')$ receives the row $\mathsf{NonNeg}_R(\mu(E'))$. No order on host representatives is consulted. □

Theorem 92.99 (Measure monotone under measurable inclusion)**.** *Let* A *and* B *be displayed measurable events in a* $\mathsf{Measure}^{\uparrow}$ *carrier. If the displayed measurable-difference row witnesses inclusion by presenting a measurable event* C *with* $B = A \sqcup C$*, then the finite disjoint-union additivity row, the nonnegative value row for* $\mu(C)$*, and the RealAlgOrder additive order row classify*

$$\mu(A) \leq_R \mu(B).$$

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureMeasurableInclusion monotone`

Proof. Read the displayed measurable-difference row as the package of a measurable event C, a disjointness witness between A and C, and a classifier identification of B with the displayed disjoint union $A \sqcup C$.

Apply Definition 92.26 to A and C. It gives the endpoint classification

$$\mu(A \sqcup C) \sim_R \mu(A) + \mu(C).$$

Transport this comparison along the displayed identification $B \sim_M A \sqcup C$ and the endpoint classifier row, obtaining $\mu(B) \sim_R \mu(A) + \mu(C)$.

Now project Definition 92.98 at C to get $\mathsf{NonNeg}_R(\mu(C))$. The RealAlgOrder additive order row says that adding a nonnegative right summand classifies the left summand below the sum, so it yields $\mu(A) \leq_R \mu(A) + \mu(C)$.

Finally transport the right endpoint of that order comparison across $\mu(B) \sim_R \mu(A) + \mu(C)$ using the $\sim_R$-stability of the RealAlgOrder order row. The result is the displayed classification $\mu(A) \leq_R \mu(B)$. □

Theorem 92.100 (Measure complement row transport)**.** *Let* M *and* N *be* $\mathsf{Measure}^{\uparrow}$ *carriers as in Definition 92.1, and suppose* $M \sim_{\mathsf{Measure}^{\uparrow}} N$ *through the public classifier of Definition 92.7. If* A *and* B *are displayed measurable events of* M *with displayed inclusion* $B \subseteq A$*, then the complement/relative-difference row of Definition 92.16 transports the event* $A \setminus B$*, its measurable witness, its disjointness witness against* B*, its binary-union readback* $B \cup (A \setminus B) \sim_M A$*, and its* $\mathsf{Real}^{\uparrow}$ *measure endpoint to the corresponding row of* N*.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureComplementRow transport`

Proof. Project the public classifier into its componentwise hsame fields. These fields transport the carried fiber, the two event presentations, their measurable witnesses, and the displayed inclusion witness from M to N.

Apply the stability clause in Definition 92.16 to those transported inputs. It returns the target relative-difference event together with the measurable, disjointness, and binary-union readback fields.

The remaining endpoint comparison is the value component of the same row. Read that component through RealConstantHistoryClassifier and then through the $\mathsf{Real}^{\uparrow}$ classifier supplied by Definition 38.9. □

Theorem 92.101 (Measure relative-difference disjoint decomposition)**.** *Let M be a $\mathsf{Measure}^{\uparrow}$ carrier as in section 92.1. Let A and B be displayed measurable events in the same carried fiber, and suppose the public event readback displays $B \subseteq A$. Then the complement/relative-difference row supplies a measurable event $A \setminus B$ such that B and $A \setminus B$ are disjoint and the binary union $B \cup (A \setminus B)$ is classified as A in the public $\mathsf{Measure}^{\uparrow}$ event classifier.*

Proof. Read the complement/relative-difference row of Definition 92.16 at the carried fiber of M. The hypotheses give exactly the row inputs: two measurable events and a displayed inclusion $B \subseteq A$ in that fiber.

Project the row. Its first projection is the measurable event $A \setminus B$. The next projections are the disjointness witness for B and $A \setminus B$ and the binary-union readback classifying $B \cup (A \setminus B)$ as A.

These projections are already ledger rows of the public carrier surface, so no host-level set subtraction or quotient equality is introduced. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRelativeDifference disjoint decomposition`

Theorem 92.102 (Measure relative-difference additivity)**.** *For a $\mathsf{Measure}^{\uparrow}$ carrier as in section 92.1, let $\mathcal{A}$ be its designated σ-algebra, let μ be its $\mathsf{Real}^{\uparrow}$-valued measure map, and write $x \sim_R y$ for the $\mathsf{Real}^{\uparrow}$ classifier of Definition 38.9. Let $A, B \in \mathcal{A}$ be displayed events in the same carried fiber, and suppose the public event readback displays $B \subseteq A$. If the certificate supplies the complement/relative-difference row of Definition 92.16 and the hypotheses of Definition 92.26 for the pair $B, A \setminus B$, then*

$$\mu(A) \sim_R \mu(B) + \mu(A \setminus B).$$

Proof. Apply Definition 92.101 to the displayed inclusion $B \subseteq A$. This gives measurability of $A \setminus B$, disjointness of B and $A \setminus B$, and the binary-union readback classifying $B \cup (A \setminus B)$ as A.

Apply Definition 92.26 to the disjoint pair B and $A \setminus B$. The theorem gives

$$\mu(B \cup (A \setminus B)) \sim_R \mu(B) + \mu(A \setminus B).$$

Use the binary-union readback from the relative-difference row to transport the left endpoint from $\mu(B \cup (A \setminus B))$ to $\mu(A)$ through the $\mathsf{Measure}^{\uparrow}$ event classifier and then through the $\mathsf{Real}^{\uparrow}$ value classifier. Composing that endpoint transport with the finite disjoint-union comparison gives the displayed finite difference law. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRelativeDifference additivity`

Theorem 92.103 (Measure self-difference zero law)**.** *For a $\mathsf{Measure}^{\uparrow}$ carrier as in section 92.1, let $\mathcal{A}$ be its designated σ-algebra, let μ be its $\mathsf{Real}^{\uparrow}$-valued measure map, write $x \sim_R y$ for the $\mathsf{Real}^{\uparrow}$ classifier of Definition 38.9, and write 0_R for the additive zero in the target $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ row. Let $A \in \mathcal{A}$ be a displayed measurable event in one carried fiber. Suppose the public event readback supplies the reflexive inclusion $A \subseteq A$, the complement/relative-difference row of Definition 92.16 is applied to this inclusion and supplies the displayed event $A \setminus A$, and the hypotheses of Definition 92.26 hold for the pair $A, A \setminus A$. If the target $\mathsf{Real}^{\uparrow}$ addition is carried by its additive $\mathsf{AbGroup}^{\uparrow}$ row, then*

$$\mu(A \setminus A) \sim_R 0_R.$$

Lean4 ✓ BEDC.Derived.MeasureUp.MeasureSelfDifference zero law

Proof. Apply Definition 92.102 to the displayed inclusion $A \subseteq A$ with $B = A$. The relative-difference row supplies the measurable event $A \setminus A$, its disjointness from A, and the binary-union readback $A \cup (A \setminus A) \sim_M A$, so the theorem gives

$$\mu(A) \sim_R \mu(A) + \mu(A \setminus A).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier, this is the left-absorption comparison

$$\mu(A) + \mu(A \setminus A) \sim_R \mu(A).$$

By Definition 42.33, read the target additive $\mathsf{AbGroup}^{\uparrow}$ row as a group row with operation $+$, unit 0_R, and classifier $\sim_R$. Its group component supplies associativity, left and right unit laws, addition congruence, and left inverses. Applying Definition 41.114 with $a = \mu(A)$ and $b = \mu(A \setminus A)$ forces the right summand to the unit, hence

$$\mu(A \setminus A) \sim_R 0_R.$$

□

Theorem 92.104 (Measure binary-union subadditivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B \in \mathcal{A}$ *be displayed measurable events in one carried fiber, and let* $U \in \mathcal{A}$ *be the displayed binary union of* A *and* B*. Suppose the public binary-union readback displays* $A \subseteq U$ *and, for the relative-difference event* $D := U \setminus A$ *supplied by Definition 92.16, displays* $D \subseteq B$*. If the certificate supplies the finite disjoint-union additivity row, the nonnegative measure-value row, and the RealAlgOrder additive order row with classifier-stable addition monotonicity, then*

$$\mu(U) \leq_R \mu(A) + \mu(B).$$

Proof. Apply Definition 92.102 to the displayed inclusion $A \subseteq U$. With $D := U \setminus A$, the relative-difference row supplies measurability of D, disjointness of A and D, and the readback $A \cup D \sim_M U$. The finite disjoint-union additivity row therefore gives

$$\mu(U) \sim_R \mu(A) + \mu(D).$$

The binary-union exclusion readback displays $D \subseteq B$. Apply Definition 92.99 to this inclusion. Its proof consumes the relative-difference row for $B \setminus D$, finite disjoint-union additivity, the nonnegative value row for the remaining piece, and the RealAlgOrder additive order row, yielding

$$\mu(D) \leq_R \mu(B).$$

Use the addition-monotonicity component of the same RealAlgOrder row with the fixed left summand $\mu(A)$. It transports the last comparison through addition and gives

$$\mu(A) + \mu(D) \leq_R \mu(A) + \mu(B).$$

Finally transport the left endpoint of this order comparison across $\mu(U) \sim_R \mu(A) + \mu(D)$ by the $\sim_R$-stability of the RealAlgOrder order row. The resulting displayed comparison is

$$\mu(U) \leq_R \mu(A) + \mu(B).$$

□

Theorem 92.105 (Measure triple-union subadditivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \sim_R y$ *and* $x \leq_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9 and the order relation exposed by the shared* RealAlgOrder *row. Let* $A, B, C \in \mathcal{A}$ *be displayed measurable events in one carried fiber. Let* $U \in \mathcal{A}$ *be the displayed binary union* $A \cup B$*, and let* $T \in \mathcal{A}$ *be the displayed binary union* $U \cup C$*. Suppose the public binary-union readback data required by Definition 92.104 are available for the pair* A, B *with union* U *and for the pair* U, C *with union* T*: the readbacks display* $A \subseteq U$ *and* $U \subseteq T$*, and the relative-difference events* $U \setminus A$ *and* $T \setminus U$ *supplied by Definition 92.16 are displayed as subevents of* B *and* C*, respectively. If the certificate supplies the finite disjoint-union additivity row, the nonnegative measure-value row, and the* RealAlgOrder *additive order row with classifier-stable addition monotonicity and transitivity, then*

$$\mu(T) \leq_R (\mu(A) + \mu(B)) + \mu(C).$$

Equivalently, for the displayed nesting $T = (A \cup B) \cup C$*,*

$$\mu((A \cup B) \cup C) \leq_R (\mu(A) + \mu(B)) + \mu(C).$$

Proof. Apply Definition 92.104 to the displayed pair U, C with binary union T. The hypotheses for this application are exactly the measurable union row $T \in \mathcal{A}$, the readback $U \subseteq T$, and the relative-difference inclusion $T \setminus U \subseteq C$, together with the finite-additivity, nonnegative-value, and additive-order rows. This gives

$$\mu(T) \leq_R \mu(U) + \mu(C).$$

Apply Definition 92.104 again to the displayed pair A, B with binary union U. The readback $A \subseteq U$ and the relative-difference inclusion $U \setminus A \subseteq B$ give

$$\mu(U) \leq_R \mu(A) + \mu(B).$$

Use the addition-monotonicity component of the same RealAlgOrder row with the fixed right summand $\mu(C)$. It transports the last comparison through addition and yields

$$\mu(U) + \mu(C) \leq_R (\mu(A) + \mu(B)) + \mu(C).$$

Finally compose the two order comparisons by transitivity of the RealAlgOrder order row. Since the displayed event T is the carried binary union $(A \cup B) \cup C$, the composite comparison is

$$\mu((A \cup B) \cup C) \leq_R (\mu(A) + \mu(B)) + \mu(C).$$

□

Theorem 92.106 (Measure ternary-union subadditivity)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1, let* $\mathcal{A}$ *be its designated* σ*-algebra, let* μ *be its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write* $x \leq_R y$ *for the order relation exposed by the shared* RealAlgOrder *row. Let* $A, B, C \in \mathcal{A}$ *be displayed measurable events in one carried fiber. Let* $U_{BC} \in \mathcal{A}$ *be the displayed binary union of* B *and* C*, and let* $U \in \mathcal{A}$ *be the displayed binary union of* A *and* U_{BC}*. Suppose the public binary-union readback data required by Definition 92.104 are available for the pair* B, C *with union* U_{BC} *and for the pair* A, U_{BC} *with union* U*: the readbacks display* $B \subseteq U_{BC}$ *and* $A \subseteq U$*, and the relative-difference events* $U_{BC} \setminus B$ *and* $U \setminus A$ *supplied by Definition 92.16 are displayed as subevents of* C *and* U_{BC}*, respectively. If the certificate supplies the finite disjoint-union additivity row, the nonnegative measure-value row, and the* RealAlgOrder *additive order row with classifier-stable addition monotonicity and transitivity, then*

$$\mu(U) \leq_R \mu(A) + (\mu(B) + \mu(C)).$$

Proof. Apply Definition 92.104 to the displayed pair A, U_{BC} with binary union U. The hypotheses for this application are the measurable union row $U \in \mathcal{A}$, the readback $A \subseteq U$, and the relative-difference inclusion $U \setminus A \subseteq U_{BC}$, together with the finite-additivity, nonnegative-value, and additive-order rows. This gives

$$\mu(U) \leq_R \mu(A) + \mu(U_{BC}).$$

Apply Definition 92.104 again to the displayed pair B, C with binary union U_{BC}. The readback $B \subseteq U_{BC}$ and the relative-difference inclusion $U_{BC} \setminus B \subseteq C$ give

$$\mu(U_{BC}) \leq_R \mu(B) + \mu(C).$$

Use the addition-monotonicity component of the same RealAlgOrder row with the fixed left summand $\mu(A)$. It transports the last comparison through addition and yields

$$\mu(A) + \mu(U_{BC}) \leq_R \mu(A) + (\mu(B) + \mu(C)).$$

Finally compose the two order comparisons by transitivity of the RealAlgOrder order row. Since U is the displayed binary union $A \cup U_{BC}$ and U_{BC} is the displayed binary union $B \cup C$, the composite comparison is exactly

$$\mu(U) \leq_R \mu(A) + (\mu(B) + \mu(C)).$$

□

Theorem 92.107 (Measure root relative-difference consumer surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root consumer package of Definition 92.86 exposes the finite relative-difference surface consisting of:*

1. *the relative-difference decomposition row of Definition 92.101;*
2. *the relative-difference additivity row of Definition 92.102;*
3. *the self-difference zero row of Definition 92.103.*

Each exposed row is read only through the root NameCert carrier rows, measurable-event classifier rows, and $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows of the same package.*

Proof. Unpack the root consumer package in Definition 92.86. Its public surface includes the root NameCert surface and the downstream package of Definition 92.64, so the measurable-event rows, finite disjoint-union rows, and $\mathsf{Real}^{\uparrow}$ endpoint classifier rows are available to consumers.

Apply Definition 92.101. It consumes a displayed measurable inclusion and returns the visible measurable relative-difference event, the disjointness witness, and the binary-union readback against the original event.

Apply Definition 92.102 to the same exposed row family. It composes the decomposition row with finite disjoint-union additivity and endpoint transport, giving the public additive comparison.

Finally apply Definition 92.103 at a reflexive displayed inclusion. Since all required rows are already rows of the root consumer package, the zero self-difference row is exposed without adding host set subtraction, quotient event equality, or a standard bridge row. □

Theorem 92.108 (Measure root finite monotonicity consumer package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root consumer package of Definition 92.86 makes measurable-inclusion monotonicity available to* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. Concretely, the monotone row Definition 92.99 is part of the same consumer boundary that supplies the* $\mathsf{Integral}^{\uparrow}$ *rows of Definition 92.87 and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *rows of Definition 92.89.*

Proof. Project the root consumer package through Definition 92.86. The package exposes the nonnegative endpoint rows, finite disjoint-union rows, and RealAlgOrder additive order row needed by measurable-inclusion monotonicity.

Apply Definition 92.99. Its proof reads the displayed measurable-difference row, applies finite disjoint-union additivity, and transports the resulting endpoint comparison through the classifier-stable order row.

The $\mathsf{Integral}^{\uparrow}$ consumer projection of Definition 92.87 reads exactly the covered measurable-event, endpoint, countable-sum, head-tail, and finite disjoint-union rows from the same root package. The monotone row is therefore available at the shared measure boundary used by the integral premeasure surface.

The $\mathsf{FunctionalAnalysis}^{\uparrow}$ projection of Definition 92.89 explicitly consumes the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows. Hence the same monotone inclusion row is available to the functional-analysis consumer surface, with no extra quotient event equality or host set equality. □

Theorem 92.109 (Measure root subadditivity consumer boundary)**.** *For a* $\mathsf{Measure}^{\uparrow}$ *carrier read through the root consumer package of Definition 92.86, binary-union subadditivity is a public finite-measure consequence:*

$$\mu(A \cup B) \leq_R \mu(A) + \mu(B).$$

The consequence is obtained by composing the binary-union subadditivity row of Definition 92.104 with the downstream unblock package of Definition 92.64 and the root consumer rows of Definition 92.87 and Definition 92.89.

Proof. Start with the root downstream package of Definition 92.64. It exposes the covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint rows, empty-zero row, finite disjoint-union row, relative-difference row, and nonnegative endpoint row through the root consumer surface.

Apply Definition 92.104. The theorem consumes exactly those finite-measure rows: relative-difference additivity for $U \setminus A$, measurable-inclusion monotonicity for the inclusion of that difference into B, and RealAlgOrder addition monotonicity for the final endpoint comparison.

The integral consumer rows of Definition 92.87 and the functional-analysis consumer rows of Definition 92.89 are projections of the same root package. Repacking the binary-union comparison through those projections makes the displayed finite subadditivity row public at the downstream consumer boundary, without adding a standard bridge or host-level event equality. □

Definition 92.110 (Measure root BHist consumer threshold)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root* BHist *consumer threshold is the ten-field package obtained by reading the root public surface of Definition 92.61 through the carrier, classifier, event coverage, additive, exactness, and consumer rows already exposed by Definition 92.57, Definition 92.63, Definition 92.26, and Definition 92.42. Its fields are:*

1. *the* BHist *carrier surface and componentwise classifier rows of the root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *package;*
2. *the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows attached to every displayed measure value and countable-sum endpoint;*
3. *the visible measurable-event, empty-event, binary-union, countable-union, complement, relative-difference, and disjoint-sequence readback rows;*
4. *the total-event endpoint row used by the probability consumer;*
5. *the empty-zero and finite disjoint-additivity rows;*

6. *the countable disjoint-additivity, countable-sum, and head-tail readback rows;*
7. *the ledger exactness row saying that every consumed membership, disjointness, relative-difference, value, and countable-sum witness is attached to one of the displayed* BHist *rows;*
8. *the* $\mathsf{Integral}^{\uparrow}$ *consumer rows from Definition 92.87;*
9. *the* $\mathsf{ProbSpace}^{\uparrow}$ *consumer rows from Definition 92.88;*
10. *the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer rows from Definition 92.89.*

The threshold exports no host set equality, quotient event equality, analytic completion object, or StandardBridge *row beyond the displayed package.*

Theorem 92.111 (Measure root BHist real endpoint threshold)**.** *The threshold package of Definition 92.110 exposes only* BHist *event rows and* $\mathsf{Real}^{\uparrow}$ *endpoint comparisons at the root surface. Every row visible at the threshold is either a displayed event-row entry of the root carrier ledger or a* $\mathsf{Real}^{\uparrow}$ *comparison attached to a measure value, a finite readback endpoint, a relative-difference endpoint, or a countable-sum endpoint.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRootBHist real endpoint threshold`

Proof. Begin with the root public surface of Definition 92.61. Its carrier and classifier rows are the displayed BHist rows and componentwise classifier rows listed by the public surface definition.

Apply Definition 92.62. This transports exactly the measurable-event, closure, relative-difference, disjoint-readback, ledger, and endpoint rows through the root classifier. The endpoint movement is read through the $\mathsf{Real}^{\uparrow}$ classifier and not through a separate equality of events.

Next apply Definition 92.63. The empty-zero, finite readback, relative-difference, and countable-additivity rows it exposes are all rooted in the same displayed event ledger and the same $\mathsf{Real}^{\uparrow}$ endpoint classifier.

The exactness clause in Definition 92.110 repacks these projections as the threshold package. Since the package lists no host equality, quotient event equality, analytic completion object, or standard bridge, no other root row is exported. □

Theorem 92.112 (Measure root BHist ProbSpace threshold)**.** *The threshold package of Definition 92.110 supplies* $\mathsf{ProbSpace}^{\uparrow}$ *with exactly the total-event normalization, empty-zero, finite-additivity, and* $\mathsf{Real}^{\uparrow}$ *endpoint normalization rows exposed by the root* $\mathsf{Measure}^{\uparrow}$ *package. It supplies no integral countable-sum row, functional-analysis relative-difference package, quotient event equality, host set equality, or standard bridge row to* $\mathsf{ProbSpace}^{\uparrow}$*.*

Proof. Project the root downstream package from Definition 92.64. Its probability-facing surface contains the total-event endpoint row together with the empty-zero and finite disjoint-union rows exposed by the root sigma-additive obligation.

Use Definition 92.86 to read those rows as consumer rows. The package is built from the root public surface and the downstream unblock package, so the probability consumer sees the same covered BHist rows and the same $\mathsf{Real}^{\uparrow}$ endpoint comparisons.

Apply Definition 92.88. It identifies the $\mathsf{ProbSpace}^{\uparrow}$ projection as exactly the total-event normalization row, empty-zero row, and finite disjoint-union row carried by the root consumer package.

Finally compare this projection with Definition 92.110. The threshold contains no additional probability-facing field, so the listed rows exhaust the $\mathsf{ProbSpace}^{\uparrow}$ consumer boundary. □

Theorem 92.113 (Measure root BHist integral and functional threshold)**.** *The threshold package of Definition 92.110 supplies* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *with exactly the measurable-event, nonnegative endpoint, monotone inclusion, relative-difference, finite-readback, countable-sum, and continuity-from-below rows exposed by the root* $\mathsf{Measure}^{\uparrow}$ *package. It supplies no probability normalization row, quotient event equality, host set equality, analytic completion object, or standard bridge row to those two consumers.*

Proof. Start from the root consumer package of Definition 92.86. Its consumer rows are the root NameCert public rows of Definition 92.61 read through the downstream package of Definition 92.64.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The projection consumes covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows from the same root package.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89. The projection consumes exactly the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows exposed by $\mathsf{Measure}^{\uparrow}$.

The continuity-from-below row is part of the countable-sum and monotone consumer face of the root package, read through the same $\mathsf{Real}^{\uparrow}$ endpoint classifier and the same visible event ledger. Repacking the two projections with Definition 92.110 gives the displayed integral and functional-analysis threshold.

The threshold has separate $\mathsf{ProbSpace}^{\uparrow}$ fields and no standard-bridge, quotient, host-equality, or analytic-completion field. Therefore no row outside the named measurable-event, endpoint, order, relative-difference, finite-readback, countable-sum, and continuity-from-below families is exported to $\mathsf{Integral}^{\uparrow}$ or $\mathsf{FunctionalAnalysis}^{\uparrow}$. □

Definition 92.114 (Measure root dependency orientation surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root dependency orientation surface is the root* BHist *package whose carrier is* MCarrier *and whose classifier is the componentwise* $\mathsf{Measure}^{\uparrow}$ *classifier transported by* hsame*. Its sequence rows are indexed only by unary observations* $n : \mathsf{BHist}$ *with* $\mathsf{UnaryHistory}(n)$*, and its continuation rows are the* Cont *rows used to keep event presentations, relative-difference witnesses, countable-union witnesses, and countable-sum endpoints in the same visible ledger.*

The input certificates of the surface are exactly the following rows:

1. $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *for displayed endpoint classifiers, zero endpoints, order comparisons, and countable-sum endpoint comparisons;*
2. *the additive real rows already named by Definition 92.14 and Definition 92.61;*
3. $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *only where the endpoint-family rows of Definition 92.77 are used.*

No row from $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *is an input certificate for this root surface. Those objects are reachable only through the downstream consumer package.*

Theorem 92.115 (Measure root input dependency scope)**.** *The carrier, classifier, event coverage, finite additivity, sigma-additivity, continuity-from-below, and endpoint-family rows of* $\mathsf{Measure}^{\uparrow}$ *are proved from the orientation surface of Definition 92.114, the BEDC kernel constructs* BHist*,* hsame*, and* Cont*, the named* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and additive real rows, and the* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows at endpoint-family sites. They are not proved from any* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer row.*

Proof. Project the carrier and classifier rows from Definition 92.17. The carrier is the displayed MCarrier row over BHist, and classifier movement is the componentwise hsame transport built into the root NameCert surface.

Next apply Definition 92.18. Its stability and exactness projections say that every visible event witness, relative-difference witness, countable-union witness, and endpoint witness is read from the displayed ledger and transported only through the root classifier, Cont, and the $\mathsf{Real}^{\uparrow}$ endpoint classifier.

The finite-additivity, sigma-additivity, and continuity-from-below rows are then read through the public additive and countable-sum rows cited by the measure surface. Endpoint-family rows use Definition 92.77, so the only extra certificate at those sites is $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$.

Finally compare the resulting root package with Definition 92.64. That theorem exports rows from $\mathsf{Measure}^{\uparrow}$ to the three downstream consumers; it does not import consumer rows back into the carrier, classifier, additivity, continuity, or endpoint-family obligations. Hence the listed root obligations have precisely the input scope stated above. □

Theorem 92.116 (Measure root consumer projection separation)**.** *The* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *rows exposed in Definition 92.86 are projections out of the root* $\mathsf{Measure}^{\uparrow}$ *package. They cannot be used as hypotheses for the root carrier, classifier, event-coverage, additivity, continuity-from-below, or endpoint-family obligations of* $\mathsf{Measure}^{\uparrow}$*.*

Proof. Start with Definition 92.86. The definition reads the root NameCert public surface together with the downstream package of Definition 92.64; its consumer rows are therefore projections from an already assembled root surface.

For the three consumers, apply Definition 92.64. The integral face receives measurable-event, endpoint, countable-sum, head-tail, and finite readback rows; the probability face receives total-event normalization, empty-zero, and finite-additivity rows; the functional-analysis face receives nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows.

Now return to the root obligations named in Definition 92.114. Their hypotheses are BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, the additive real rows, and endpoint-family $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows. None of the three consumer projections appears among those hypotheses.

The exactness row of Definition 92.86 keeps the consumer boundary one-way: downstream rows may cite the root package, but root proof obligations cite only the orientation surface and its named input certificates. This separates the projections from the hypotheses used to build $\mathsf{Measure}^{\uparrow}$. □

Theorem 92.117 (Measure root deps-ready public package)**.** *The orientation surface of Definition 92.114 together with the threshold package of Definition 92.110 forms the public deps-ready* $\mathsf{Measure}^{\uparrow}$ *package for* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. This package exposes exactly the root* BHist *carrier and classifier rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, additive and continuity rows, endpoint-family rows, the downstream consumer projections, and the no-*$\mathsf{StandardBridge}$ *boundary recorded by the threshold.*

Proof. Use Definition 92.114 to fix the root input face. It supplies the carrier orientation over BHist, classifier transport by hsame, continuation discipline through Cont, and the only external certificates allowed at root input sites.

Apply Definition 92.115. This proves that the root carrier, classifier, coverage, additivity, continuity-from-below, and endpoint-family rows are obtained before any downstream consumer projection is used.

Next combine those rows with Definition 92.110. The threshold package lists the visible event ledger, endpoint comparisons, additive rows, exactness row, and the three downstream consumer faces, while excluding host set equality, quotient event equality, analytic completion objects, and standard bridges.

Finally apply Definition 92.116. It identifies the consumer faces as outgoing projections only. Repacking the input face and the threshold face therefore gives a public deps-ready package that downstream analysis chapters may consume without using those chapters to justify the root $\mathsf{Measure}^{\uparrow}$ obligations. □

Definition 92.118 (Measure root domain-ready package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root domain-ready package is the paper-side root package whose fields are the following ten rows:*

1. *the traditional constructive measure-space target, namely a measurable-event carrier, empty event, finite unions, countable unions, relative differences, disjointness witnesses, nonnegative* $\mathsf{Real}^{\uparrow}$ *endpoints, and countable-sum endpoints;*
2. *the BEDC source row over* BHist*, together with* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint comparison, the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row, and* $\mathsf{StreamName}^{\uparrow}$ *countable-family names;*
3. *the carrier rows displayed by Definition 92.14;*
4. *the componentwise classifier rows displayed by Definition 92.15;*
5. *the stability rows for measurable events, endpoint comparisons, countable-family presentations, relative differences, and countable sums;*
6. *the visible ledger and exactness rows for hidden membership, disjointness, countable-union witnesses, and endpoint readback;*
7. *the core theorem list consisting of carrier-classifier transport, sigma-additivity, empty-zero, finite disjoint-union readback, finite-support countable readback, relative-difference additivity, and continuity from below;*
8. *the exactness predicate stating that every consumer row is one of the displayed root rows;*
9. *the explicit absence of any* $\mathsf{StandardBridge}$ *row;*
10. *the status row exposing this object as the paper-side root package for* $\mathsf{Measure}^{\uparrow}$*.*

Theorem 92.119 (Measure root domain carrier-classifier ready)**.** *The domain-ready package of Definition 92.118 supplies carrier transport, componentwise classifier transport, visible event-row transport, ledger transport, and* $\mathsf{Real}^{\uparrow}$ *endpoint transport for the root* $\mathsf{Measure}^{\uparrow}$ *domain. These rows are exactly the carrier-classifier part of the root public surface under the input scope of* BHist*,* hsame*,* Cont*,* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row, and* $\mathsf{StreamName}^{\uparrow}$ *countable-family names.*

Proof. Start from the carrier and classifier theorem Definition 92.62. It projects the carried $\mathsf{Measure}^{\uparrow}$ code, visible event rows, ledger rows, and $\mathsf{Real}^{\uparrow}$ endpoint rows along the public classifier.

Apply Definition 92.21. Its certificate-level projections identify the same componentwise classifier movement for event presentations, values, and countable-sum components.

Finally apply Definition 92.115. The input scope pins the transport rows to BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, the additive real row, and the endpoint-family $\mathsf{StreamName}^{\uparrow}$ rows, with no downstream consumer row used as a hypothesis. These projections are precisely the carrier-classifier fields of the domain-ready package. □

Theorem 92.120 (Measure root domain sigma-additive ready)**.** *The domain-ready package of Definition 92.118 supplies sigma-additivity, empty-zero, finite disjoint-union readback, finite-support*

countable readback, relative-difference additivity, and continuity-from-below rows for the root $\mathsf{Measure}^{\uparrow}$ *domain.*

Proof. Apply Definition 92.63. It gives the public sigma-additive comparison, the empty-zero row, finite disjoint-union readback, and the relative-difference row from the root NameCert surface.

Next use Definition 92.23. This reads finite-support countable families through the same countable-union, disjointness, endpoint, and additive rows.

Apply Definition 92.102 to keep the relative-difference row visible as a root domain consequence, and apply Definition 92.43 to expose continuity from below at the same measure surface.

The listed rows use the carrier, endpoint, countable-family, additive, and ledger fields already named by the domain-ready package, so they repack as its sigma-additive face. □

Theorem 92.121 (Measure root domain downstream ready)**.** *The carrier-classifier theorem Definition 92.119 and the sigma-additive theorem Definition 92.120 expose exactly the rows consumed by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. The resulting downstream face contains no quotient event equality, no host set equality, and no standard bridge row.*

Proof. First apply Definition 92.64. It reads the carrier-classifier rows and sigma-additive rows as the public $\mathsf{Measure}^{\uparrow}$ package available to the three downstream consumers.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The consumed rows are the covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88. The consumed rows are total-event normalization together with empty-zero and finite disjoint-union readback.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, use the consumer projection recorded in Definition 92.86 and made explicit by Definition 92.89. Its rows are the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows.

The exactness predicate in Definition 92.86 excludes every row not named above. Hence the downstream face of the domain-ready package contains neither quotient event equality nor host set equality, and its standard-bridge field remains the explicit no-$\mathsf{StandardBridge}$ row. □

Definition 92.122 (Measure root source-closure package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root source-closure package is the deps-ready* BHist *source obtained by repacking the carrier surface of Definition 92.14, the certificate surface of Definition 92.15, the dependency orientation surface of Definition 92.114, and the root consumer threshold of Definition 92.110. Its carrier is* $\mathsf{MCarrier}$*, its classifier is the componentwise* $\mathsf{Measure}^{\uparrow}$ *classifier* $\sim_M$*, its event rows are the visible measurable membership, empty, binary-union, countable-union, complement, relative difference, and disjoint-readback rows, its endpoint rows are the displayed* $\mathsf{Real}^{\uparrow}$ *measure-value and countable-sum rows, and its ledger rows are exactly the source witnesses attached to those displayed event and endpoint rows. The package has no consumer-row hypotheses:* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *occur only as outgoing faces of the threshold.*

Theorem 92.123 (Measure root source event-row closure)**.** *The empty, binary-union, countable-union, complement and relative-difference, disjoint-readback, and continuity-from-below event rows of Definition 92.122 are exactly the visible event rows exposed by the carrier surface and root threshold package. No downstream consumer row is an event-row hypothesis of this source package.*

Proof. Project Definition 92.15. Its event part consists of the measurable-family row, the empty row, the binary-union row, the countable-union row, the complement and relative-difference row, and the disjoint readbacks attached to the same carried BHist fiber.

Apply Definition 92.115. The input scope says that these rows are read from the orientation surface through BHist, hsame, Cont, the named $\mathsf{Real}^{\uparrow}$ rows, and the endpoint-family $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows where sequences are present.

The continuity-from-below event layer is the same visible chain of measurable events and relative-difference layers named by Definition 92.42. It therefore lies inside the carrier and threshold event ledger rather than in a consumer face.

Finally apply Definition 92.116. The integral, probability, and functional-analysis rows are projections out of an already assembled root package, so they cannot add further event-row hypotheses. The displayed source package therefore has exactly the event rows listed in the statement. □

Theorem 92.124 (Measure root source endpoint-row closure)**.** *The measure-value, countable-sum, head-tail readback, nonnegative endpoint, monotone-inclusion, finite-additivity, and relative-difference endpoint rows of Definition 92.122 are all read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and the additive real rows named in Definition 92.114.*

Proof. Start with Definition 92.115. The theorem identifies the only endpoint inputs for the root source as $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, the additive real rows named by the carrier and public NameCert surfaces, and $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ only at endpoint-family sites.

Apply Definition 92.111. Every endpoint visible at the threshold is either a measure value, a finite readback endpoint, a relative-difference endpoint, or a countable-sum endpoint, and each comparison is a $\mathsf{Real}^{\uparrow}$ classifier comparison rather than a host event equality.

The finite-additivity and head-tail readback rows are the additive real readbacks already used by the measure certificate. The nonnegative and monotone-inclusion rows are endpoint comparisons over the same $\mathsf{Real}^{\uparrow}$ classifier, so they introduce no separate endpoint carrier.

Finally use Definition 92.42. Its endpoint claim is built from the countable-sum endpoint and finite-prefix readback rows over the visible layer sequence. Thus the source package closes its endpoint rows through the named real certificates and additive rows alone. □

Theorem 92.125 (Measure root source ledger exhaustion)**.** *Every source witness consumed by Definition 92.122 is one of the visible measurable membership, relative-difference, disjointness, measure-value,* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family, or countable-sum ledger rows. The package contains no host set equality, quotient event equality, probability normalization, integral premeasure row, functional-analysis row, or standard bridge.*

Proof. Apply Definition 92.13. Its exactness clause lists the consumed membership, relative-difference, disjointness, value, and countable-sum witnesses as rows of the displayed carrier ledger.

Next read Definition 92.110. The threshold adds the endpoint-family rows and the three outgoing consumer faces while retaining the exclusion of host set equality, quotient event equality, analytic completion objects, and standard bridges.

Now apply Definition 92.116. The consumer faces for $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ are outputs from the root surface, so their premeasure, normalization, and functional-analysis rows cannot become source witnesses.

Combining exactness with the one-way threshold boundary exhausts the source ledger. Every consumed witness is one of the displayed source rows, and every row named in the exclusion clause remains outside the package. □

Theorem 92.126 (Measure root source deps-ready package)**.** *The source-closure package of Definition 92.122, together with the event-row closure of Definition 92.123, the endpoint-row closure of Definition 92.124, and the ledger exhaustion of Definition 92.125, forms the deps-ready root*

source package consumed by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$. *The public package of Definition 92.117 is used only as the outgoing consumer boundary.*

Proof. First package the source carrier and classifier from Definition 92.122. This fixes $\mathsf{MCarrier}$, $\sim_M$, the visible event ledger, the $\mathsf{Real}^{\uparrow}$ endpoint rows, and the source witness ledger as one BHist package.

Next insert the closure rows. Apply Definition 92.123 for the event side and Definition 92.124 for the endpoint side. These two rows identify all event and endpoint inputs before any downstream consumer face is read.

Then apply Definition 92.125. It shows that the package consumes only the visible source witnesses and excludes host equality, quotient event equality, normalization, premeasure, functional-analysis, and bridge rows.

Finally cite Definition 92.117 as the outgoing boundary for $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$. Since that boundary exports consumer projections but does not supply source hypotheses, the repacked source package is deps-ready for the three consumers. □

Definition 92.127 (Measure root consumer face classifier)**.** *The consumer face classifier of the* $\mathsf{Measure}^{\uparrow}$ *root threshold is the three-face partition of the consumer rows in Definition 92.110. The integral face is the* $\mathsf{Integral}^{\uparrow}$ *projection of Definition 92.87: it contains covered measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows. The probability face is the* $\mathsf{ProbSpace}^{\uparrow}$ *projection of Definition 92.88: it contains the total-event normalization row, empty-zero row, and finite disjoint-union row. The functional-analysis face is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *projection of Definition 92.89: it contains nonnegative endpoint rows, monotone-inclusion rows, relative-difference rows, and countable-sum classifier rows.*

Rows common to more than one face are classified as shared root rows only when they are the displayed BHist *event-ledger rows or the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows already present at the root* $\mathsf{Measure}^{\uparrow}$ *surface. No face classifier row is a host set equality, quotient event equality, analytic completion object, or* $\mathsf{StandardBridge}$ *row.*

Theorem 92.128 (Measure root consumer face coverage)**.** *Every consumer row projected by Definition 92.86 belongs to one of the three faces of Definition 92.127. Thus each outgoing consumer row is classified as integral, probability, functional-analysis, or as a shared root* BHist *event-ledger or* $\mathsf{Real}^{\uparrow}$ *endpoint row seen through those faces.*

Proof. Start from Definition 92.86. Its consumer rows are the public root surface together with the downstream package of Definition 92.64.

Apply Definition 92.87, Definition 92.88, and Definition 92.89. These three theorems list the complete rows consumed by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ from that same root package.

Finally use Definition 92.127. The classifier names exactly those three outgoing projections and assigns their common event-ledger and endpoint rows to the shared root part, so every row projected by the consumer package has one of the displayed face labels. □

Theorem 92.129 (Measure root consumer face disjointness)**.** *The integral, probability, and functional-analysis faces of Definition 92.127 overlap only on the shared root* BHist *event ledger and the shared* $\mathsf{Real}^{\uparrow}$ *endpoint classifier. The total-event probability-normalization row remains specific to the probability face, the countable-sum endpoint and head-tail readback rows remain specific to the integral face, and the relative-difference and monotone-inclusion rows remain specific to the functional-analysis face.*

Proof. First read the three outgoing projections through the consumer-row theorems. Definition 92.87 gives the integral rows, Definition 92.88 gives the probability rows, and Definition 92.89 gives the functional-analysis rows.

The only rows those projections are allowed to share are the displayed covered event rows and $\mathsf{Real}^{\uparrow}$ endpoint classifier rows already present in the root $\mathsf{Measure}^{\uparrow}$ package. This is exactly the shared-root clause in Definition 92.127.

Now apply Definition 92.116. The separation theorem makes the three consumer projections one-way outputs from the root surface and prevents any consumer-specific row from becoming a root hypothesis.

The exclusions in the three consumer-row theorems then keep the remaining rows face-specific: probability normalization is absent from the integral and functional-analysis surfaces, countable-sum endpoint and head-tail readback rows are absent from the probability surface, and relative-difference plus monotone-inclusion rows are absent from the probability consumer surface and do not become integral premeasure rows. □

Theorem 92.130 (Measure root cross-consumer threshold package)**.** *The face classifier of Definition 92.127, the coverage result of Definition 92.128, the disjointness result of Definition 92.129, and the deps-ready package of Definition 92.117 form the root consumer threshold for* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$. *This threshold exposes the shared* BHist *event ledger and shared* $\mathsf{Real}^{\uparrow}$ *endpoint classifier once, exposes each consumer-specific row only on its own face, and keeps all outgoing faces from serving as inputs to the root* $\mathsf{Measure}^{\uparrow}$ *obligations.*

Proof. Use Definition 92.117 to fix the public root package. It supplies the orientation surface, the BHist threshold, and the downstream projections needed by the three consumers.

Next apply Definition 92.128. This assigns every outgoing row of the consumer package to the integral, probability, or functional-analysis face, with shared event-ledger and endpoint rows read at the root.

Then apply Definition 92.129. The face-specific probability-normalization, countable-sum, head-tail, relative-difference, and monotone-inclusion rows remain on their named consumer faces.

Finally apply Definition 92.116. Since the consumer rows are outgoing projections and not hypotheses for the root carrier, classifier, coverage, additivity, continuity, or endpoint-family obligations, the packaged classifier is a root consumer threshold rather than a justification loop for $\mathsf{Measure}^{\uparrow}$. □

Definition 92.131 (Measure root shared endpoint consumer seal)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root shared endpoint consumer seal fixes the endpoint rows that may be read by more than one outgoing analysis face before any such face is consumed. Its shared source is exactly the combination of the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows and the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family rows already named by Definition 92.122. The* BHist *event-ledger rows remain the only shared event source rows.*

The integral, probability, and functional-analysis faces are outgoing consumers of this sealed source. They may project the root event ledger and the shared endpoint classifier rows, but their face-specific rows are not source hypotheses of the root package. Thus probability normalization, integral head-tail readback, and functional-analysis monotone and relative-difference rows are consumer rows only.

Theorem 92.132 (Measure root shared endpoint coverage)**.** *Every shared endpoint row used by the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *faces of Definition 92.131 is either a* $\mathsf{Measure}^{\uparrow}$ BHist *event-ledger row, a* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row, or a* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family row already exposed by the root source-closure package.*

Proof. First project the source endpoint closure of Definition 92.124. It identifies the measure-value, countable-sum, finite-additivity, head-tail, nonnegative, monotone, and relative-difference endpoint rows through $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ and the additive real rows of the root orientation surface.

Next apply Definition 92.79. The StreamName-indexed rows that appear at countable endpoint sites are stable pointwise under the StreamName classifier, so they remain the endpoint-family rows named by the root source package rather than a new analytic carrier.

Now use Definition 92.128. Each outgoing row belongs to the integral, probability, or functional-analysis face, with any overlap classified as a shared root event-ledger row or a shared $\mathsf{Real}^\uparrow$ endpoint row.

Combining these projections gives the stated coverage. A row shared by the three consumers has already appeared in the root event ledger, the $\mathsf{Real}^\uparrow$ endpoint classifier, or the $\mathsf{StreamName}^\uparrow$ endpoint family before the consumer face reads it. □

Theorem 92.133 (Measure root consumer face source separation)**.** *The probability-normalization rows, integral head-tail readback rows, and functional-analysis monotone and relative-difference rows of Definition 92.131 never become source hypotheses of the root* $\mathsf{Measure}^\uparrow$ *package.*

Proof. Start with Definition 92.125. The source ledger consumes only the visible measurable membership, relative-difference, disjointness, measure-value, $\mathsf{StreamName}^\uparrow$ endpoint-family, and countable-sum rows. It explicitly excludes probability normalization, integral premeasure rows, functional-analysis rows, and bridge rows from the root source ledger.

Next apply Definition 92.129. The probability-normalization row is specific to the probability face, the head-tail readback row is specific to the integral face, and the monotone and relative-difference rows are specific to the functional-analysis face except where their endpoint classifier has already been sealed as a shared root row.

Finally use Definition 92.116. The three consumer faces are one-way projections out of the assembled root surface and cannot serve as hypotheses for the root carrier, classifier, event-coverage, additivity, continuity, or endpoint-family obligations.

The exclusions from the source ledger and the one-way projection boundary therefore separate consumer-specific rows from root source hypotheses. □

Theorem 92.134 (Measure root shared endpoint deps-ready seal)**.** *The seal of Definition 92.131, the coverage theorem of Definition 92.132, the source-separation theorem of Definition 92.133, the source package of Definition 92.126, and the cross-consumer threshold of Definition 92.130 form the deps-ready root seal consumed by* $\mathsf{Integral}^\uparrow$*,* $\mathsf{ProbSpace}^\uparrow$*, and* $\mathsf{FunctionalAnalysis}^\uparrow$*.*

Proof. First fix the source package by Definition 92.126. This supplies the BHist carrier and classifier, the event ledger, the $\mathsf{Real}^\uparrow$ endpoint rows, the $\mathsf{StreamName}^\uparrow$ endpoint-family rows, and the exhausted source ledger.

Next insert the shared endpoint seal. Apply Definition 92.132 to show that every shared row consumed by the three analysis faces is already one of the sealed source rows. Apply Definition 92.133 to keep probability normalization, integral head-tail readback, and functional-analysis monotone and relative-difference rows on the consumer side.

Finally apply Definition 92.130. It packages the three outgoing faces over the same shared event ledger and endpoint classifier while keeping face-specific rows on their own projections.

Repacking these three steps gives a deps-ready root seal: the shared endpoint rows are identified once at the root, and the three downstream chapters consume only the corresponding outgoing faces. □

Definition 92.135 (Measure root regular value package)**.** *A* $\mathsf{Measure}^{\uparrow}$ *root regular value package is the following ten-field package.*

1. *The traditional target is the finite regular measure-value surface: finitely many measurable events, finite additivity on pairwise-disjoint displayed supports, and zero absorption for null rows.*
2. *The BEDC source is* $\mathsf{Measure}^{\uparrow}$ *over* BHist*, with its public event ledger read through* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$*.*
3. *The endpoint source is* $\mathsf{Real}^{\uparrow}$*, using the value classifier* $\sim_R$ *of Definition 38.9 and the additive* $\mathsf{AbGroup}^{\uparrow}$ *row with zero endpoint* 0_R*.*
4. *The countable row source is the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family row used to display countable measurable sequences and countable* $\mathsf{Real}^{\uparrow}$ *sum endpoints.*
5. *The carrier is a measurable-event predicate* $\mathcal{A}(h, E)$ *on carried histories* $h : \mathsf{BHist}$*, together with a finite displayed support spine* $E_0, \ldots, E_{n-1}$*.*
6. *The classifier transports the event spine by the* $\mathsf{Measure}^{\uparrow}$ *event classifier and transports the displayed values* $\mu_h(E_i)$ *by the* $\mathsf{Real}^{\uparrow}$ *value classifier.*
7. *The stability obligations are finite additivity on pairwise-disjoint displayed supports, zero and null-event absorption rows, carrier transport, and countable-tail readback for finite-support sequences.*
8. *The ledger records the hidden support enumeration and the all-zero tail after the displayed finite spine.*
9. *Exactness consists of finite coverage of every displayed support row, soundness of each measured row against the* $\mathsf{Measure}^{\uparrow}$ *carrier, and completeness of the finite value endpoint against the displayed support.*
10. *The package status is* $(\mathsf{seed}, \mathsf{formalTarget})$*: the paper declares the root target shape, and no Lean marker is attached at this site.*

Theorem 92.136 (Measure root finite regular value closure)**.** *Let* $h : \mathsf{BHist}$ *be carried by the* $\mathsf{Measure}^{\uparrow}$ *surface, and let* $E_0, \ldots, E_{n-1}$ *be a finite displayed support spine of pairwise-disjoint measurable events in the* h*-fiber. Suppose the* $\mathsf{Measure}^{\uparrow}$ *certificate supplies the existing finite-additivity row, the canonical countable-zero-tail row, and the root public countable-additivity threshold for the corresponding finite-support* $\mathsf{StreamName}^{\uparrow}$ *sequence. Then the displayed values* $\mu_h(E_0), \ldots, \mu_h(E_{n-1})$ *and their finite* $\mathsf{Real}^{\uparrow}$ *endpoint form a carried root regular value package in the sense of Definition 92.135.*

Proof. First build the carrier spine. The hypotheses give measurable rows for every E_i, pairwise disjointness on the displayed finite support, and the finite-support $\mathsf{StreamName}^{\uparrow}$ sequence whose tail is constantly empty.

Apply Definition 92.26 along the finite spine. This gives the binary finite-additivity comparisons needed to fold the visible support values into one $\mathsf{Real}^{\uparrow}$ endpoint.

Next read the countable presentation. By Definition 92.32, the hidden tail after the finite support is classified with 0_R, so the finite endpoint and the countable $\mathsf{StreamName}^{\uparrow}$ endpoint name the same value under $\sim_R$.

Finally apply Definition 92.96. The public root threshold supplies the countable-additivity comparison for the carried sequence, and its coverage rows identify the displayed spine as the complete support ledger. These rows are exactly the carrier, classifier, stability, ledger, and exactness fields of Definition 92.135. □

Theorem 92.137 (Measure root null absorption regularity)**.** *Let a carried root regular value package be displayed over a finite pairwise-disjoint measurable support spine. If a further measurable event N is classified by* $\mathsf{Measure}^{\uparrow}$ *with the zero* $\mathsf{Real}^{\uparrow}$ *value* 0_R *and is disjoint from the displayed support, then adjoining N preserves the same regular package endpoint under the* $\mathsf{Measure}^{\uparrow}$ *value classifier.*

Proof. Start with the displayed package endpoint and the additional measurable row for N. The disjointness hypothesis makes the enlarged support spine a legitimate finite measurable spine in the same $\mathsf{Measure}^{\uparrow}$ carrier.

Apply Definition 92.28. It identifies the finite endpoint after adjoining the null row with the original endpoint under the $\mathsf{Real}^{\uparrow}$ classifier.

The zero endpoint used by that neutral row is grounded by Definition 92.25. This keeps the added value inside the same $\mathsf{Measure}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ package rows rather than introducing a separate analytic zero object.

Finally use the carrier and classifier transport supplied by Definition 92.20. The transported event witness and the transported value witness keep the enlarged spine on the same root regular value package endpoint. □

Theorem 92.138 (Measure root regular value downstream package)**.** *Every carried root regular value package supplies the event row, the* $\mathsf{Real}^{\uparrow}$ *endpoint row, and the zero-tail countable-additivity row required by the* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer surfaces.*

Proof. Project the package fields of Definition 92.135. The carrier field gives the finite measurable-event ledger, the endpoint field gives the classified $\mathsf{Real}^{\uparrow}$ values, and the ledger field gives the hidden zero tail for the finite-support countable presentation.

Apply Definition 92.87. The integral surface reads exactly the covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows.

Apply Definition 92.88. The probability surface reads the empty-zero and finite disjoint-union rows already present in the same regular package.

Apply Definition 92.89. The functional-analysis surface reads the countable-sum classifier row and the endpoint order rows through the same $\mathsf{Measure}^{\uparrow}$ carrier and $\mathsf{Real}^{\uparrow}$ classifier. Therefore the regular value package supplies the common root rows needed by all three downstream consumers. □

Definition 92.139 (Measure root consumer threshold triad)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root consumer threshold triad is the three-sided package whose source side is the deps-ready source package of Definition 92.126, whose shared endpoint side is the deps-ready seal of Definition 92.134, and whose consumer side consists exactly of the three outgoing faces supplied by Definition 92.112 and Definition 92.113. The* $\mathsf{ProbSpace}^{\uparrow}$ *face receives the total-event normalization, empty-zero, finite-additivity, and* $\mathsf{Real}^{\uparrow}$ *endpoint normalization rows. The* $\mathsf{Integral}^{\uparrow}$ *face receives the measurable-event, finite-readback, countable-sum, head-tail, and continuity-from-below rows. The* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *face receives the nonnegative endpoint, monotone inclusion, relative-difference, countable-sum, and continuity-from-below rows. All three faces read the same root* BHist *event ledger and the same sealed* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *endpoint rows.*

Theorem 92.140 (Measure root source threshold triad face)**.** *The source face of Definition 92.139 exposes only the* BHist *event ledger, the* $\mathsf{Measure}^{\uparrow}$ *carrier and componentwise classifier, the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family rows, additivity, continuity from below, and*

ledger exactness. It exposes no $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer-specific row as a root source hypothesis.*

Proof. Project the source package from Definition 92.126. It fixes the BHist carrier, the $\mathsf{Measure}^{\uparrow}$ classifier rows, the visible event ledger, the $\mathsf{Real}^{\uparrow}$ endpoint rows, the $\mathsf{StreamName}^{\uparrow}$ endpoint-family rows, and the exhausted source ledger before any consumer face is read.

The event and endpoint parts of that package are already identified by Definition 92.123 and Definition 92.124. These two projections account for the root event rows, additivity rows, continuity-from-below rows, endpoint classifier rows, and endpoint-family rows named in the statement.

Finally apply Definition 92.125. Its exclusion clause removes probability normalization, integral premeasure rows, functional-analysis rows, host equality, quotient event equality, and bridge rows from the source ledger. Thus the source face of the triad contains only the displayed root rows. □

Theorem 92.141 (Measure root shared endpoint threshold triad face)**.** *The shared endpoint face of Definition 92.139 is exactly the collection of* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows and* $\mathsf{StreamName}^{\uparrow}$ *endpoint-family rows sealed in Definition 92.134 before projection to* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{FunctionalAnalysis}^{\uparrow}$*.*

Proof. Start with Definition 92.134. The seal is built from the source package, shared endpoint coverage, consumer-face source separation, and the cross-consumer threshold package. Its shared rows are therefore fixed before any outgoing face consumes them.

Apply Definition 92.132. Every row shared by the three analysis faces is already a root BHist event-ledger row, a $\mathsf{Real}^{\uparrow}$ endpoint classifier row, or a $\mathsf{StreamName}^{\uparrow}$ endpoint-family row. The endpoint part of the shared face is precisely the last two families.

Then use Definition 92.133. It keeps probability normalization, integral head-tail readback, and functional-analysis monotone and relative-difference rows on their own consumer projections unless their endpoint classifier has already been sealed as a shared root row. Hence no extra endpoint row appears after projection. □

Theorem 92.142 (Measure root consumer threshold triad exhaustion)**.** *In the triad of Definition 92.139,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume only their named outgoing faces. No probability normalization row, integral head-tail or premeasure row, functional-analysis monotone or relative-difference row, host set equality, quotient event equality, analytic completion object, or* $\mathsf{StandardBridge}$ *row becomes a root* $\mathsf{Measure}^{\uparrow}$ *hypothesis.*

Proof. First project the consumer side of Definition 92.139. The $\mathsf{ProbSpace}^{\uparrow}$ face is the face identified by Definition 92.112; the $\mathsf{Integral}^{\uparrow}$ and $\mathsf{FunctionalAnalysis}^{\uparrow}$ faces are the two faces identified by Definition 92.113.

Next compare these outgoing faces with the source face of Definition 92.140. The source face has already exhausted the root carrier, classifier, event ledger, endpoint rows, additivity, continuity, and ledger exactness without importing any consumer-specific row.

Finally apply Definition 92.141. Shared endpoint rows are sealed as $\mathsf{Real}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows before consumer projection, while the remaining probability, integral, and functional-analysis rows stay on their named outgoing faces. This exhausts the triad and prevents every consumer-specific row listed in the statement from becoming a root $\mathsf{Measure}^{\uparrow}$ hypothesis. □

Definition 92.143 (Measure root seed obligation front)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root seed obligation front is the ten-field surface that places the root measure certificate before its downstream consumers:*

1. *traditional target: a constructive measure space whose measurable predicates form a σ-algebra and whose measure takes values in* $\mathsf{Real}^{\uparrow}$*;*
2. *BEDC source:* BHist *presentations, the certificate* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, and the additive* $\mathsf{AbGroup}^{\uparrow}$ *row on real endpoints;*
3. *carrier: the predicate* MeasCode *on* BHist*;*
4. *classifier: the root classifier $\sim_M$ on measure codes, with endpoint comparisons read through the* $\mathsf{Real}^{\uparrow}$ *classifier $\sim_R$;*
5. *stability rows: transport of* MeasCode*, visible event rows, measurable-family rows, endpoint rows, countable-sum rows, and head-tail readback along $\sim_M$ and componentwise* hsame*;*
6. *ledger rows: measurable membership, relative-difference, disjointness, endpoint-value, and countable-sum witnesses attached to the visible* BHist *rows they justify;*
7. *core theorem list: Definition 92.62, Definition 92.63, and Definition 92.64;*
8. *exactness: visible-row exhaustion by Definition 92.57 and Definition 92.13;*
9. *standard bridge: no* StandardBridge *is asserted at this paper site;*
10. *status:* seed *on the theory axis, with the theorem-shaped target surface for* formalTarget *under* BEDC.Derived.MeasureUp.

This front is read through the carrier rows of Definition 92.14, the certificate rows of Definition 92.15, the root public surface of Definition 92.61, and the root consumer package of Definition 92.86.

Theorem 92.144 (Measure root seed carrier-classifier obligation)**.** *The seed front of Definition 92.143 supplies the root carrier and classifier obligation for* $\mathsf{Measure}^{\uparrow}$*. If h, k :* BHist *satisfy* $\mathsf{MeasCode}(h)$ *and $h \sim_M k$, then the front transports* $\mathsf{MeasCode}(k)$*, every displayed event row, every ledger row, and every* $\mathsf{Real}^{\uparrow}$ *endpoint comparison attached to the carried measure code. The endpoint movement is always read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and the classifier $\sim_R$, while event movement is read through componentwise* hsame *on the* BHist *rows.*

Proof. Start from the carrier and classifier fields of Definition 92.143. They are the MeasCode and $\sim_M$ fields of Definition 92.15, read over the visible rows listed in Definition 92.14.

Apply Definition 92.8. It transports the carried measure code and permits every displayed row indexed by that code to be read at the target history k after the $\sim_M$ comparison is supplied.

Next apply Definition 92.62. Its public form identifies the transported rows as the root carrier, classifier, ledger, and endpoint rows of the $\mathsf{Measure}^{\uparrow}$ surface. The event rows move by their componentwise hsame witnesses, and the endpoint rows move through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ and $\sim_R$.

Finally use the root exactness clause from Definition 92.13. Since every consumed membership, relative-difference, disjointness, value, and countable-sum witness is attached to a visible row, the transported visible rows exhaust the root carrier-classifier obligation. □

Theorem 92.145 (Measure root seed sigma-additive obligation)**.** *The seed front of Definition 92.143 supplies the root sigma-additive obligation for* $\mathsf{Measure}^{\uparrow}$*. For every covered pairwise-disjoint measurable sequence $E_\bullet$ in a carried* BHist *fiber with covered countable union U, the front supplies the comparison*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

It also supplies the empty-zero, finite disjoint-union, relative-difference, head-tail, and countable-sum classifier rows needed to read this comparison through the additive $\mathsf{AbGroup}^{\uparrow}$ *real row.*

Proof. Project the sigma-algebra, countable-union, countable-sum, head-tail, and classifier-stability rows from Definition 92.15. The coverage predicate of Definition 92.57 places the sequence $E_\bullet$, its union U, its disjointness witness, and its endpoint rows in the visible BHist ledger.

Apply Definition 92.63. Its first conclusion is precisely the displayed comparison $\mu(U) \sim_R \Sigma_\mu(E_\bullet)$, with both endpoints classified through $\sim_R$.

The same public obligation invokes Definition 92.22, Definition 92.25, Definition 92.26, and Definition 92.102. These rows provide the empty-zero, finite readback, relative-difference, head-tail, and countable-sum classifier consequences.

The additive structure used in those readbacks is the $\mathsf{AbGroup}^\uparrow$ real row named in Definition 92.143. Thus the sigma-additive obligation is a root BHist and $\mathsf{Real}^\uparrow$ endpoint statement, not a separate host-level measure-space equality. □

Theorem 92.146 (Measure root seed consumer threshold)**.** *The seed front of Definition 92.143 reaches the root consumer threshold for* $\mathsf{Integral}^\uparrow$*,* $\mathsf{ProbSpace}^\uparrow$*, and* $\mathsf{FunctionalAnalysis}^\uparrow$*. The threshold exports exactly the covered measurable-event rows,* $\mathsf{Real}^\uparrow$ *endpoint classifier rows, countable-sum rows, head-tail readback, empty-zero, finite disjoint-union readback, relative-difference additivity, monotone inclusion, nonnegative endpoint, total-event normalization, and the explicit no-*$\mathsf{StandardBridge}$ *row exposed by the root* $\mathsf{Measure}^\uparrow$ *package.*

Proof. Begin with the root public surface of Definition 92.61 and the root consumer package of Definition 92.86. Together they expose the BHist event ledger, the $\sim_M$ carrier classifier, and the $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ endpoint classifier rows named by the seed front.

Apply Definition 92.64. It packages the carrier-classifier rows of Definition 92.62 with the sigma-additive rows of Definition 92.63, while keeping the standard-bridge field absent.

For the three outgoing faces, apply Definition 92.87, Definition 92.88, and Definition 92.89. These theorems read the integral, probability, and functional-analysis projections from the same root consumer package and list the rows each projection may consume.

Finally compare those projections with Definition 92.111, Definition 92.112, and Definition 92.113. The threshold contains no host set equality, quotient event equality, analytic completion object, or $\mathsf{StandardBridge}$ field, so the exported consumer surface is exactly the one stated above. □

Definition 92.147 (Measure root visible shared consumer envelope)**.** *The* $\mathsf{Measure}^\uparrow$ *root visible shared consumer envelope is the ten-field root obligation envelope whose target is the shared* $\mathsf{Measure}^\uparrow$ *source consumed by* $\mathsf{Integral}^\uparrow$*,* $\mathsf{ProbSpace}^\uparrow$*, and* $\mathsf{FunctionalAnalysis}^\uparrow$*. Its source field is the BEDC package over* BHist*,* hsame*,* Cont*,* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{StreamName}^\uparrow}$*, and the* $\mathsf{UnaryHistory}$*-indexed endpoint families. Its carrier field is the root carrier* $\mathsf{MCarrier}$*, and its classifier field is the componentwise* $\mathsf{Measure}^\uparrow$ *classifier.*

The stability fields are event-row transport, endpoint classifier transport, shared endpoint coverage, and one-way consumer projection. The ledger fields are visible membership witnesses, relative-difference witnesses, disjointness witnesses, measure-value witnesses, $\mathsf{StreamName}^\uparrow$ *endpoint-family witnesses, and countable-sum witnesses. The exactness field excludes host set equality, quotient event equality, analytic completion objects, consumer-row hypotheses, and standard bridges from the root source.*

Theorem 92.148 (Measure root visible shared consumer coverage)**.** *Every row in Definition 92.147 is either a visible* BHist *event ledger row, a* $\mathsf{Real}^\uparrow$ *endpoint classifier row, a* $\mathsf{StreamName}^\uparrow$ *endpoint-family row indexed by* $\mathsf{UnaryHistory}$*, or a* Cont*-preserved continuation row already present in the root* $\mathsf{Measure}^\uparrow$ *package.*

Proof. Start from Definition 92.122. It fixes the root carrier MCarrier, the componentwise $\mathsf{Measure}^{\uparrow}$ classifier, the visible event ledger, the $\mathsf{Real}^{\uparrow}$ endpoint rows, and the source witness ledger before any outgoing analysis face is read.

Apply Definition 92.132. The rows shared by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ have already been classified as root event-ledger rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, or $\mathsf{StreamName}^{\uparrow}$ endpoint-family rows.

Now use Definition 92.111. It keeps the measure-value, finite readback, relative-difference, and countable-sum endpoint rows on the $\mathsf{Real}^{\uparrow}$ classifier side rather than turning them into host event equalities.

Finally read the transport boundary from Definition 92.114. The only remaining root movement is Cont-preserved continuation through BHist and the displayed endpoint-family rows indexed by UnaryHistory, so the envelope has exactly the four row classes stated above. □

Theorem 92.149 (Measure root visible shared consumer separation)**.** *The* $\mathsf{Integral}^{\uparrow}$ *premeasure rows,* $\mathsf{ProbSpace}^{\uparrow}$ *normalization rows, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *monotone and relative-difference rows visible from Definition 92.147 are outgoing projections only. They cannot serve as hypotheses for the* $\mathsf{Measure}^{\uparrow}$ *carrier, classifier, additivity, continuity-from-below, or endpoint-family obligations.*

Proof. First apply Definition 92.116. The three analysis faces are projections out of the assembled root package, and the root obligations are proved from BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, additive real rows, and endpoint-family $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows.

Next use Definition 92.133. Its source boundary keeps probability normalization, integral head-tail readback, and functional-analysis monotone and relative-difference rows on their consumer faces except where a shared endpoint classifier row has already been sealed at the root.

Finally invoke Definition 92.125. The source ledger consumes only visible membership, relative-difference, disjointness, measure-value, $\mathsf{StreamName}^{\uparrow}$ endpoint-family, and countable-sum witnesses. Thus no premeasure, normalization, monotone, or consumer-specific relative-difference row can be repurposed as a source hypothesis. □

Theorem 92.150 (Measure root visible shared consumer deps-ready)**.** *The envelope of Definition 92.147, the coverage theorem of Definition 92.148, and the separation theorem of Definition 92.149 form the deps-ready* $\mathsf{Measure}^{\uparrow}$ *root package consumed by* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*. Only the displayed root rows are exported.*

Proof. Begin with Definition 92.126. It supplies the root BHist source package, including MCarrier, the componentwise classifier, visible event rows, $\mathsf{Real}^{\uparrow}$ endpoint rows, $\mathsf{StreamName}^{\uparrow}$ endpoint-family rows, and the exhausted source ledger.

Next apply Definition 92.134. The seal identifies the shared endpoint rows once at the root and keeps the probability, integral, and functional-analysis face-specific rows from entering the source side.

Then apply Definition 92.130. The threshold packages the three outgoing faces over the same shared event ledger and endpoint classifier while preserving one-way projection into the consumer chapters.

Repack these three sources with the envelope coverage and separation theorems. The resulting root package is deps-ready for the three downstream analysis consumers because it exports only visible event rows, endpoint classifier rows, endpoint-family rows, continuation rows, and the displayed ledger witnesses. □

Definition 92.151 (Measure root index threshold source)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root index threshold source is the* BHist*-indexed source surface whose rows are read from the public carrier of Definition 92.1 and the root source seal of Definition 92.131. For each carried history it fixes an event carrier, a public measurable-event classifier, a* $\mathsf{Real}^{\uparrow}$*-valued endpoint row* μ*, the empty event, finite disjoint-union rows, countable indexed-union rows, countable-sum endpoint rows, and the visible ledger that records which* BHist *event row supports each endpoint.*

The source is indexed only by the displayed BHist *carrier and its componentwise classifier. It contains no host set equality, no quotient event equality, no analytic completion object, and no* $\mathsf{StandardBridge}$ *row.*

Theorem 92.152 (Measure root index carrier threshold)**.** *The source of Definition 92.151 projects the* $\mathsf{Measure}^{\uparrow}$ *event carrier, the public event-classifier transport row, the empty-event row, and the visible ledger rows needed by the root threshold. Each projection is a row of the same* BHist*-indexed source, and endpoint movement is read only through the* $\mathsf{Real}^{\uparrow}$ *classifier.*

Proof. Start from the carrier surface of Definition 92.1. Its displayed rows supply the event carrier, measurable-event predicate, measure endpoint, and carrier transport along the public classifier.

Apply Definition 92.8. This carries the measurable-family, empty-event, union, relative-difference, endpoint, and countable-sum rows along the same BHist classifier without adding host equality.

The visible ledger is the root public ledger of Definition 92.95. The shared endpoint boundary is sealed by Definition 92.132, so endpoint transport is read through the existing $\mathsf{Real}^{\uparrow}$ row rather than through a separate event identity. □

Theorem 92.153 (Measure root index additivity threshold)**.** *For the source of Definition 92.151, finite disjoint additivity and countable indexed additivity are read from the same root source. The finite row is the two-step finite-additivity package, and the countable row is the sigma-additive comparison for a pairwise-disjoint countable indexed family:*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

Proof. Project the empty-event, binary-union, disjointness, endpoint, head-tail, and countable-sum rows from the source. These rows are available together by Definition 92.24 and are public at the root by Definition 92.95.

For finite additivity, apply Definition 92.35. Its empty-zero, null-event, ordered binary-union, and symmetric binary-union rows all use the same visible carrier and $\mathsf{Real}^{\uparrow}$ endpoint classifier.

For countable indexed additivity, apply Definition 92.96. It reads the pairwise-disjoint indexed family, the countable union U, and the countable-sum endpoint $\Sigma_\mu(E_\bullet)$ from the same root threshold source and gives the displayed comparison.

The continuity-from-below package of Definition 92.42 uses these same rows for its layer construction and endpoint readback. Thus finite and countable additivity are not separate inputs; both are projections of the single BHist-indexed source. □

Theorem 92.154 (Measure root index downstream threshold)**.** *The chapters* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume exactly the root index threshold source of Definition 92.151. They receive the public event carrier, classifier transport, empty-event, finite disjoint-additivity, countable indexed-additivity, countable-sum endpoint, and visible-ledger rows, and they receive no host set equality, quotient event equality, analytic completion row, or* $\mathsf{StandardBridge}$ *row from* $\mathsf{Measure}^{\uparrow}$*.*

Proof. First package the source as the root deps-ready threshold of Definition 92.94. The carrier rows are the rows of Definition 92.152, and the additive rows are the rows of Definition 92.153.

Apply Definition 92.97. It identifies the root package consumed by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ and excludes quotient event equality, host set equality, and standard bridge rows.

Now read the three consumer faces separately. The integral face is Definition 92.87, the probability face is Definition 92.88, and the functional-analysis face is Definition 92.89. Each face is a projection of the same source rows, with no additional carrier.

Finally apply Definition 92.134. That seal keeps shared $\mathsf{Real}^{\uparrow}$ endpoint rows at the root and keeps probability normalization, integral head-tail readback, and functional-analysis order rows on their outgoing consumer faces. Therefore the downstream chapters consume exactly this threshold package and nothing beyond the named public rows. □

Definition 92.155 (Measure root downstream threshold map)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root downstream threshold map is the ten-field projection from the root domain-ready package of Definition 92.118 and the root index threshold source of Definition 92.151 to the three analysis consumers* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*:*

1. *source package: the carrier-classifier, ledger, endpoint, additive, exactness, and no-*$\mathsf{StandardBridge}$ *rows of the root domain-ready package;*
2. *threshold source: the* BHist*-indexed carrier, public classifier, visible ledger, empty-event, finite-additivity, countable- additivity, countable-sum endpoint, and boundary rows of the root index threshold source;*
3. *carrier rows: the event carrier, measurable-event classifier, empty event, finite unions, countable indexed unions, relative differences, and disjointness witnesses;*
4. *classifier rows: componentwise* hsame *on* BHist *rows,* $\sim_M$ *on carried measure codes, and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *comparison for endpoint movement through* $\sim_R$*;*
5. *ledger rows: hidden membership, disjointness, countable-union, relative-difference, endpoint-value, and countable-sum witnesses attached to visible* BHist *rows;*
6. *finite-additivity rows: empty-zero, null-event, ordered binary-union, symmetric binary-union, finite disjoint-union readback, and relative-difference additivity;*
7. *countable-additivity rows: pairwise-disjoint indexed families, countable unions, countable-sum endpoints, head-tail readback, finite-support countable readback, and continuity from below;*
8. *endpoint rows: nonnegative* $\mathsf{Real}^{\uparrow}$ *endpoints, additive* $\mathsf{AbGroup}^{\uparrow}$ *real transport, countable-sum endpoints, and the shared endpoint boundary used by all three consumers;*
9. *outgoing consumer rows: the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *faces displayed by Definition 92.86;*
10. *boundary rows: absence of host set equality, quotient event equality, analytic completion rows, and standard bridge rows.*

Theorem 92.156 (Measure root threshold map carrier and additivity)**.** *The threshold map of Definition 92.155 carries exactly the carrier-classifier rows of Definition 92.119 and Definition 92.152, and exactly the finite and countable additivity rows of Definition 92.120 and Definition 92.153.*

Proof. Project carrier transport first. The domain-ready carrier-classifier theorem supplies carried $\mathsf{Measure}^{\uparrow}$ codes, visible event rows, ledger rows, and $\mathsf{Real}^{\uparrow}$ endpoint rows, while the index carrier threshold supplies the same rows as projections of the BHist-indexed source.

Endpoint movement is then read through the $\mathsf{Real}^{\uparrow}$ classifier. The domain-ready package uses $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ comparison, and the index threshold source seals endpoint movement through $\sim_R$ rather than through host event equality.

Finally compose the finite and countable additivity faces through the common BHist source. The domain-ready sigma-additive theorem supplies empty-zero, finite disjoint-union readback, finite-support countable readback, relative-difference additivity, and continuity from below. The index additivity threshold supplies the finite and countable indexed rows of the same source, so the map contains exactly those additivity rows and no separate additive input. □

Theorem 92.157 (Measure root threshold map consumer unblock)**.** *The threshold map of Definition 92.155 supplies exactly the outgoing rows consumed by* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$. *These are the downstream rows exposed by Definition 92.121 and Definition 92.154, read through the consumer faces of Definition 92.87, Definition 92.88, and Definition 92.89.*

Proof. Read the $\mathsf{Integral}^{\uparrow}$ face first. Its consumed rows are the covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows supplied by the two root threshold sources.

Read the $\mathsf{ProbSpace}^{\uparrow}$ face next. It consumes total-event normalization, empty-zero, and finite disjoint-union readback, all projected from the same carrier, endpoint, and additive rows.

Read the $\mathsf{FunctionalAnalysis}^{\uparrow}$ face separately. Its outgoing rows are the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows exposed by the common root package.

The exactness clauses in Definition 92.86 and Definition 92.151 exhaust the outgoing consumer boundary. Therefore host set equality, quotient event equality, analytic completion rows, and standard bridge rows are excluded from the threshold map. □

92.5 Measure root countable obligation triple

Definition 92.158 (Measure root countable obligation triple)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root countable obligation triple is the ten-field target surface that packages the countable-name root rows into the three obligations visible to downstream analysis:*

1. *traditional target: constructive countable measure names whose events are visible through StreamName rows and whose values are* $\mathsf{Real}^{\uparrow}$ *endpoints;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, *and the history classifier* hsame*;*
3. *carrier: the carried measure code together with its StreamName endpoint family, as exposed by Definition 92.90;*
4. *classifier: event-row* hsame *transport together with the pointwise StreamName classifier and the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier;*
5. *stability rows: carrier transport, pointwise event-row transport, endpoint transport, StreamName endpoint-family transport, and countable-sum classifier stability;*
6. *ledger rows: event membership, pairwise disjointness, countable union, endpoint value, and countable-sum witnesses attached to the visible* BHist *rows they certify;*
7. *core theorem list: Definition 92.159, Definition 92.160, and Definition 92.161;*
8. *exactness: visible-row exhaustion through Definition 92.57, with no hidden event row consumed outside the displayed ledger;*

9. *standard bridge: no* StandardBridge *row is asserted at this paper site;*
10. *status: paper target for* BEDC.Derived.MeasureUp.

The triple records only the root BHist, *StreamName, and* $\mathsf{Real}^{\uparrow}$ *certificate rows. It does not assert host set equality, quotient event equality, analytic completion, or any standard bridge.*

Theorem 92.159 (Measure root countable carrier obligation)**.** *The triple of Definition 92.158 projects the carrier support needed at the* $\mathsf{Measure}^{\uparrow}$ *root. Its carrier rows are exactly the carried measure code and the StreamName endpoint-family carrier rows displayed by Definition 92.90.*

Proof. Read the carrier field of Definition 92.158. It names the carried BHist measure code and the StreamName endpoint family already fixed in Definition 92.90.

Apply Definition 92.91. The first projection of that obligation is the transported $\mathsf{Measure}^{\uparrow}$ carrier row, and its StreamName projection is the endpoint-family carrier row at each visible unary index.

The carrier projection uses only those displayed rows. No host set equality, quotient event equality, analytic completion object, or StandardBridge row is introduced when the carried code and endpoint family are repacked as the root carrier obligation. □

Theorem 92.160 (Measure root countable classifier obligation)**.** *The triple of Definition 92.158 projects the classifier support needed at the* $\mathsf{Measure}^{\uparrow}$ *root. Its classifier rows are exactly event-row* hsame *transport,* $\mathsf{Real}^{\uparrow}$ *endpoint classification, and pointwise StreamName classifier transport from Definition 92.91.*

Proof. Start with the classifier field of Definition 92.158. It separates the event-row classifier, the endpoint classifier, and the StreamName pointwise classifier so that each row is read on its own certificate face.

Apply Definition 92.91. The event projection supplies componentwise hsame transport for visible BHist event rows and their measurable witnesses. The endpoint projection supplies the $\mathsf{Real}^{\uparrow}$ comparison of the corresponding values.

The StreamName projection of the same theorem supplies pointwise classifier transport for the endpoint family at every visible index. Repacking these three projections gives the root countable classifier obligation and adds no host set equality, quotient event equality, analytic completion object, or StandardBridge row. □

Theorem 92.161 (Measure root countable additivity obligation)**.** *The triple of Definition 92.158 projects the countable-additivity support needed at the* $\mathsf{Measure}^{\uparrow}$ *root. For every visible pairwise-disjoint event family* $E_\bullet$ *with visible countable union* U*, it supplies the* $\mathsf{Real}^{\uparrow}$ *comparison*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

This comparison is the common projection of Definition 92.92 and the indexed additivity row of Definition 92.153.

Proof. Project the visible event family from Definition 92.158. Its ledger rows name the event-membership witnesses, pairwise-disjointness witness, countable union row, endpoint-value witnesses, and countable-sum witness in the same BHist surface.

Apply Definition 92.92. It reads the visible StreamName-indexed family through the $\mathsf{Measure}^{\uparrow}$ certificate and produces the displayed comparison between the value of the union and the countable endpoint sum.

Apply Definition 92.153. Its indexed additivity row reads the same pairwise-disjoint family, union, and countable sum from the root index threshold source, so the additivity comparison is shared by the countable-name and root-index surfaces.

Exactness is supplied by the visible-row clause of the triple and by the coverage predicate named in Definition 92.57. Thus the root additivity obligation consumes only displayed event, union, value, and countable-sum rows; it excludes host set equality, quotient event equality, analytic completion, and StandardBridge. □

Definition 92.162 (Measure root countable-premeasure front)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root countable-premeasure front is the ten-field surface that exposes the traditional constructive countable premeasure target through BEDC names:*

1. *traditional target: a countable premeasure surface whose measurable events are visible as counted rows and whose values are nonnegative* $\mathsf{Real}^{\uparrow}$ *endpoints;*
2. *BEDC source:* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, $\mathsf{StreamName}^{\uparrow}$ *countable families,* BHist, hsame, *the* $\mathsf{Measure}^{\uparrow}$ *carrier and classifier rows, nonnegative endpoint rows, empty-event rows, finite-additivity rows, relative-difference rows, and countable-sum rows;*
3. *carrier choice: measurable-event* BHist *rows together with the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier for the value of each covered event;*
4. *classifier choice: componentwise* hsame *transport for event rows, measurable-witness transport, and endpoint comparison through the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$*;*
5. *stability rows: carrier transport, classifier transport, finite-additivity readback, relative-difference additivity, continuity-from-below readback, and countable-sum classifier stability;*
6. *ledger: hidden measurable witnesses, disjointness witnesses, countable-family indices, countable-union witnesses, and the countable-sum endpoints assigned to the same visible* BHist *rows;*
7. *exactness: coverage, soundness, and completeness for every displayed measurable-event row named by Definition 92.57;*
8. *displayed classification iff: a countable disjoint union is classified by the front exactly when its covered component rows, covered union row, pairwise-disjointness row, and* $\mathsf{Real}^{\uparrow}$ *countable endpoint comparison are all displayed in the same ledger;*
9. *consumer boundary: downstream chapters may read the front only through the rows listed in Definition 92.165;*
10. *standard bridge: no* StandardBridge*, analytic completion object, host set equality, or quotient event equality is part of this front.*

It is the countable-name premeasure face of the root package described by Definition 92.143 and Definition 92.90.

Theorem 92.163 (Measure root countable-premeasure carrier obligation)**.** *The front of Definition 92.162 exposes exactly the carried* BHist *measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, carrier transport, and classifier transport needed by the root* $\mathsf{Measure}^{\uparrow}$ NameCert.

Proof. Start with the seed carrier-classifier obligation of Definition 92.144. It transports the carried measure code, every displayed measurable-event row, every ledger row, and every $\mathsf{Real}^{\uparrow}$ endpoint comparison attached to the root $\mathsf{Measure}^{\uparrow}$ surface.

Next restrict those rows to the countable-premeasure front. The carrier field of Definition 92.162 keeps only covered measurable-event BHist rows and the endpoint classifier for their values. The classifier field keeps only componentwise hsame event transport, measurable-witness transport, and $\mathsf{Real}^{\uparrow}$ endpoint transport.

Finally compare with the countable-name carrier package of Definition 92.91. Its StreamName endpoint transport is read over the same visible event rows, so the front contains all carrier and classifier transport needed by the root $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ and no additional host-level equality row. □

Theorem 92.164 (Measure root countable-premeasure additivity obligation)**.** *The front of Definition 92.162 obtains countable disjoint additivity by combining the root sigma-additivity row, finite-additivity readback, relative-difference additivity, continuity-from-below, and countable-sum classifier stability.*

Proof. Project the covered disjoint countable family from the front ledger. The component rows, countable-union row, disjointness witness, and countable-sum endpoint all live in the same visible BHist carrier.

Apply Definition 92.145. It supplies the root comparison between the measure of the covered union and the countable endpoint sum, with both endpoints compared through the $\mathsf{Real}^{\uparrow}$ classifier.

For finite readback, use Definition 92.26 and Definition 92.23. These identify the binary and finite-support cases as projections of the same countable additivity row.

For difference and limiting readback, apply Definition 92.102 and Definition 92.42. They supply the relative-difference decomposition and increasing-union readback without adding any consumer-specific row to the root package.

Finally use Definition 92.12 to transport pointwise endpoint comparisons to the countable-sum endpoint. Thus every additivity clause of the front is read from the existing root $\mathsf{Measure}^{\uparrow}$ rows. □

Theorem 92.165 (Measure root countable-premeasure consumer threshold)**.** $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers may read only the countable-premeasure front of Definition 92.162. The* $\mathsf{Integral}^{\uparrow}$ *face receives covered-event, countable-sum, and finite-readback rows; the* $\mathsf{ProbSpace}^{\uparrow}$ *face adds only the total-mass normalization row; and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *face receives nonnegative, monotone, relative-difference, and countable-sum stability rows.*

Proof. Begin with the root seed consumer threshold of Definition 92.146. It fixes the outgoing consumer rows for $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ as projections of the same root $\mathsf{Measure}^{\uparrow}$ package.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87 together with Definition 92.60. The consumer reads covered measurable events, countable-sum rows, head-tail readback, and the finite readback supplied by the countable-premeasure front.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88 and Definition 92.59. These rows add the total-mass normalization endpoint while leaving the event ledger and additivity rows inherited from the same front.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89, Definition 92.98, Definition 92.99, and Definition 92.102. These projections read only nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum stability rows.

The exhaustion clause of Definition 92.142 keeps every other consumer-specific row off the root surface. Hence the three analysis chapters may consume exactly the rows stated above. □

92.6 Measure root real-stream regularity front

The real-stream regularity front is the BHist-anchored reading of the countable-premeasure root surface through the already certified $\mathsf{StreamName}^{\uparrow}$ endpoint rows and $\mathsf{Real}^{\uparrow}$ endpoint comparisons. It records the exact rows that later analysis consumers may use without turning the $\mathsf{Measure}^{\uparrow}$ root into a host measure-space schema.

Definition 92.166 (Measure root real-stream regularity front)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root real-stream regularity front is the ten-field surface with the following data:*

1. *traditional target: a constructive countable premeasure whose values are exposed as regular real endpoints indexed by a countable stream name;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, *and* hsame, *restricted to the rows displayed by Definition 92.162 and Definition 92.77;*
3. *carrier: covered measurable-event* BHist *rows, carried* $\mathsf{Measure}^{\uparrow}$ *codes, StreamName endpoint-family rows, and the* $\mathsf{Real}^{\uparrow}$ *endpoint rows attached to those observations;*
4. *classifier: componentwise* hsame *transport of event histories, pointwise StreamName classifier transport, and* $\mathsf{Real}^{\uparrow}$ *endpoint comparison for the values and countable sums;*
5. *stability rows: carrier transport, endpoint-family stability, finite-readback stability, relative-difference stability, monotone-row stability, and countable-sum classifier stability;*
6. *ledger: covered event rows, measurable witnesses, unary observation names, pairwise-disjointness witnesses, countable-union rows, finite readback rows, normalization rows, nonnegative rows, monotone rows, and relative-difference rows;*
7. *core theorem list: Definition 92.163, Definition 92.78, Definition 92.164, Definition 92.80, Definition 92.12, Definition 92.165, and Definition 92.81;*
8. *exactness: a row belongs to the front exactly when it is a displayed countable-premeasure row over the same visible* BHist *ledger and its StreamName observation carries the corresponding* $\mathsf{Real}^{\uparrow}$ *endpoint comparison;*
9. *standard bridge boundary: no* StandardBridge, *host sigma-algebra equality, quotient event equality, analytic completion object, or external regular-measure theorem is part of the front;*
10. *status: the front is a paper-side real-stream regularity surface for future* BHist *formalization targets in* BEDC.Derived.MeasureUp.

Theorem 92.167 (Measure root real-stream carrier regularity)**.** *The front of Definition 92.166 transports exactly the carried* $\mathsf{Measure}^{\uparrow}$ *codes, the StreamName endpoint family rows, and the* $\mathsf{Real}^{\uparrow}$ *endpoint comparisons over the same* BHist *ledger. The carrier transport is the common projection of Definition 92.163 and Definition 92.78.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureRootRealStreamCarrierRegularity`

Proof. Begin with a row of the real-stream regularity front. By the carrier and ledger fields of Definition 92.166, the row contains a covered measurable-event history, a carried $\mathsf{Measure}^{\uparrow}$ code, a StreamName endpoint-family observation, and the attached $\mathsf{Real}^{\uparrow}$ endpoint comparison.

Apply Definition 92.163. It transports the carried measure code, the covered event row, and the root endpoint classifier along the visible hsame comparison of BHist histories.

Next apply Definition 92.78. The StreamName observation map and its pointwise endpoint-family rows move over the same unary observations, so the transported event row and the transported real endpoint remain attached to the same stream position.

Repack these projections into the carrier, classifier, and ledger fields of Definition 92.166. Since both source theorems expose only displayed BHist rows and $\mathsf{Real}^{\uparrow}$ classifier comparisons, the repacked front adds no host event equality or quotient row. □

Theorem 92.168 (Measure root real-stream additivity regularity)**.** *Countable disjoint additivity on Definition 92.166 is exactly the shared* $\mathsf{Real}^{\uparrow}$ *endpoint comparison obtained from Definition 92.164, Definition 92.80, and Definition 92.12.*

Proof. Project the countable family from the front ledger. Its component event rows, pairwise-disjointness witness, countable-union row, and countable-sum endpoint are the rows named in the countable-premeasure front and observed pointwise by the StreamName endpoint family.

Apply Definition 92.164. This gives the root comparison between the value of the covered countable union and the countable sum of the component $\mathsf{Real}^{\uparrow}$ endpoints.

Apply Definition 92.80. It reads the same comparison through the StreamName-indexed endpoint family and keeps the additivity row scoped to the displayed measurable sequence, its union row, and its countable-sum endpoint.

Finally apply Definition 92.12. Pointwise $\mathsf{Real}^{\uparrow}$ endpoint comparisons along the stream name repack into the countable-sum endpoint comparison, so countable disjoint additivity on the front is the shared $\mathsf{Real}^{\uparrow}$ comparison and not a separate analytic additivity object. □

Theorem 92.169 (Measure root real-stream consumer regularity boundary)**.** *The* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers of Definition 92.166 may consume only its covered-event, endpoint, countable-sum, finite-readback, normalization, nonnegative, monotone, and relative-difference rows. This boundary is the common projection of Definition 92.165 and Definition 92.81.*

Proof. Start from the consumer field of Definition 92.166. It lists exactly the rows visible to downstream chapters: covered events, $\mathsf{Real}^{\uparrow}$ endpoint comparisons, countable sums, finite readback, normalization, nonnegative values, monotone inclusion, and relative differences.

Apply Definition 92.165. It assigns covered-event, countable-sum, and finite-readback rows to $\mathsf{Integral}^{\uparrow}$; the normalization endpoint to $\mathsf{ProbSpace}^{\uparrow}$; and the nonnegative, monotone, relative-difference, and countable-sum stability rows to $\mathsf{FunctionalAnalysis}^{\uparrow}$.

Now apply Definition 92.81. The StreamName endpoint family supplies only the indexed endpoint rows already named by the root consumer package, so each consumer receives the same rows through stream observations rather than through a larger event carrier.

Repack the two projections as the front boundary. No consumer can read beyond the covered-event ledger, endpoint comparisons, countable-sum row, finite-readback row, normalization row, nonnegative row, monotone row, and relative-difference row, because the standard bridge boundary field excludes host equality, quotient equality, and analytic completion data. □

Theorem 92.170 (Measure root downstream consumer lattice)**.** *The real-stream regularity front of Definition 92.166 exports exactly three downstream consumer faces through chapter 123, chapter 184, and chapter 94. These faces retain the existing* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and no other analysis chapter receives a larger carrier from this front.*

Proof. Apply Definition 92.169. It already restricts downstream use to covered-event, endpoint, countable-sum, finite-readback, normalization, nonnegative, monotone, and relative-difference rows over the same displayed BHist ledger and the same $\mathsf{StreamName}^{\uparrow}$ endpoint-family observations.

The $\mathsf{Integral}^{\uparrow}$ face is the one cited by chapter 123. It reads the finite readback and countable-sum rows, together with the covered-event rows that attach those sums to the visible $\mathsf{Measure}^{\uparrow}$ package.

The $\mathsf{ProbSpace}^{\uparrow}$ face is the one cited by chapter 184. It reads the normalization endpoint and the nonnegative endpoint data already carried by the real-stream front, so its probability-space package remains a $\mathsf{Real}^{\uparrow}$ endpoint package rather than a host event-equality surface.

The $\mathsf{FunctionalAnalysis}^{\uparrow}$ face is the one cited by chapter 94. It reads the monotone row, the relative-difference row, and the countable-sum stability row as stability data for the same Cont, Pkg, and $\mathsf{NameCert}$ rows.

These three projections exhaust the consumer boundary: the front keeps the $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{Real}^{\uparrow}$ endpoint comparisons, BHist histories, hsame transport, and package/certificate rows visible, while excluding host event equality, quotient event equality, analytic completion, and open-cover compactness. □

Definition 92.171 (Measure head-tail tail scope)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^{\uparrow}$ certificate surface as in Definition 92.15, and let $E_\bullet$ be a measurable sequence in that surface. The head-tail tail scope of $E_\bullet$ is the four-row package consisting of:*

1. *the shifted measurable sequence $E_{\bullet+1}$, read as the family $n \mapsto E_{n+1}$ by the same measurable-family row;*
2. *pointwise $\sim_M$ transport for the shifted event presentations, with endpoint movement read through the $\mathsf{Real}^{\uparrow}$ classifier $\sim_R$;*
3. *the countable endpoint $\Sigma_\mu(E_{\bullet+1})$ attached to the shifted measurable sequence;*
4. *admissibility of $E_{\bullet+1}$ as an input to the same head-tail readback row and countable-sum classifier row that classify $\Sigma_\mu(E_\bullet)$.*

This scope records only certificate-surface rows. It does not identify shifted host sequences by ambient equality, and it does not introduce a separate measure construction outside the displayed $\mathsf{Measure}^{\uparrow}$ rows.

Theorem 92.172 (Measure head-tail tail measurable-sequence scope)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^{\uparrow}$ certificate surface as in Definition 92.15, and let $E_\bullet$ be a measurable sequence in that surface. Then the shifted family $E_{\bullet+1}$ is an admitted measurable sequence for the same certificate surface. Consequently every head-tail readback later applied to $\Sigma_\mu(E_{\bullet+1})$ is an application of the displayed measurable-family row of $\mathcal{M}$, not an implicit host-level shift.*

Proof. Project the measurable-family row from Definition 92.15. Since $E_\bullet$ is supplied as a measurable sequence, each displayed event E_n carries the certificate membership row required by $\mathsf{MeasFam}$.

Form the shifted family by reindexing the same displayed rows: at index n the new family reads the old row at $n+1$. This uses no new event carrier and no extra endpoint; it repacks the existing measurable witnesses into the sequence-indexed slot demanded by the surface.

The carrier inventory of Definition 92.6 lists measurable sequence rows and countable-sum rows as ledger entries. Therefore the shifted family is admitted exactly as another measurable sequence over the same certificate surface, and the head-tail row may be projected for it. □

Theorem 92.173 (Measure head-tail tail countable-sum scope)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^{\uparrow}$ certificate surface as in Definition 92.15, and let $E_\bullet$ be a measurable sequence with head-tail tail scope as*

in Definition 92.171. Then $\Sigma_\mu(E_{\bullet+1})$ is the countable-sum endpoint consumed by the head-tail comparison for $E_\bullet$:

$$\Sigma_\mu(E_\bullet) \sim_R \mu(E_0) + \Sigma_\mu(E_{\bullet+1}).$$

If $F_\bullet$ is another admitted tail sequence and every index satisfies $\mu(E_{n+1}) \sim_R \mu(F_n)$, then the countable endpoint transported into the head-tail comparison satisfies

$$\Sigma_\mu(E_{\bullet+1}) \sim_R \Sigma_\mu(F_\bullet).$$

Proof. Use Definition 92.172 to read $E_{\bullet+1}$ as a measurable sequence in the same surface. The countable-sum field of Definition 92.15 therefore supplies the endpoint $\Sigma_\mu(E_{\bullet+1})$ as a certificate row.

Project the head-tail readback row of the same surface and apply it to $E_\bullet$. The displayed comparison is exactly the row named in Definition 92.171: its tail term is the endpoint attached to the admitted shifted family.

For classifier transport, apply Definition 92.12 to the two admitted measurable sequences $E_{\bullet+1}$ and $F_\bullet$. The pointwise comparison $\mu(E_{n+1}) \sim_R \mu(F_n)$ is precisely the hypothesis required by the countable-sum classifier row, so the transported endpoint is accepted by the same head-tail comparison. □

Theorem 92.174 (Measure head-tail consumer exhaustion)**.** *Every later finite-support, continuity-from-below, integral, probability, or functional-analysis use of a shifted countable tail in the* $\mathsf{Measure}^\uparrow$ *carrier consumes exactly the following four certificate rows:*

1. *shifted measurable-sequence scope for $E_{\bullet+1}$;*
2. *the shifted endpoint $\Sigma_\mu(E_{\bullet+1})$;*
3. *head-tail readback classifying $\Sigma_\mu(E_\bullet)$ with $\mu(E_0) + \Sigma_\mu(E_{\bullet+1})$;*
4. *countable-sum classifier stability transporting pointwise tail equality through $\sim_R$.*

In particular, the finite-support readbacks that repeatedly apply head-tail to shifted sequences, including Definition 92.33 and Definition 92.38, are audited by these four rows together with the sigma-additive input of Definition 92.10.

Proof. The first two rows are exactly the tail package of Definition 92.171, with admissibility supplied by Definition 92.172. They account for the measurable-sequence witness and the countable endpoint before any additive readback is used.

The third row is the head-tail projection of Definition 92.15. It is the row consumed in the finite-support readbacks when the original sequence, once-shifted sequence, and twice-shifted sequence are exposed successively.

The fourth row is the countable-sum classifier stability obligation Definition 92.12. It is the only transport principle used when a shifted tail is compared with the canonical empty tail or with a downstream tail carrying pointwise equal $\mathsf{Real}^\uparrow$ endpoints.

The remaining downstream consumers named in the statement receive the same surface through the carrier inventory of Definition 92.6. Since the inventory lists measurable sequence rows, countable-sum rows, head-tail readback, and classifier stability as the available rows, there is no fifth certificate input for a shifted countable tail. □

Definition 92.175 (Measure finite-prefix head-tail readback spine)**.** *Let $\mathcal{M}$ be a* $\mathsf{Measure}^\uparrow$ *certificate surface as in Definition 92.15, and let $E_\bullet$ be a measurable sequence in that surface. For a*

unary history n with $\mathsf{Unary}(n)$, define the iterated shifted family $E_{\bullet+n}$ and the finite-prefix head-tail endpoint $\mathsf{HT}_n(E_\bullet)$ by unary recursion:

$$E_{\bullet+\mathsf{e}} := E_\bullet,$$
$$E_{\bullet+\mathsf{E1}(n)} := (E_{\bullet+n})_{\bullet+1},$$

and

$$\mathsf{HT}_\mathsf{e}(E_\bullet) := \Sigma_\mu(E_\bullet),$$
$$\mathsf{HT}_{\mathsf{E1}(n)}(E_\bullet) := \mu(E_0) +_R \mathsf{HT}_n(E_{\bullet+1}).$$

Thus $\mathsf{HT}_n(E_\bullet)$ is the right-nested endpoint obtained by reading the first n heads and then stopping at the countable endpoint of the n-shifted tail. The spine records only the measurable-family row, the countable-sum endpoint row, the head-tail row, and the additive $\mathsf{Real}^\uparrow$ row; it does not identify shifted host sequences by ambient equality.

Lemma 92.176 (Measure iterated shifted-tail admissibility)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^\uparrow$ certificate surface as in Definition 92.15, let $E_\bullet$ be a measurable sequence in that surface, and let n be a unary history with $\mathsf{Unary}(n)$. Then $E_{\bullet+n}$, as defined in Definition 92.175, is an admitted measurable sequence for the same certificate surface.*

Proof. Induct on the unary generation of n. In the empty case, $E_{\bullet+\mathsf{e}}$ is $E_\bullet$, which is measurable by hypothesis.

For the one-step case, assume $E_{\bullet+n}$ is an admitted measurable sequence. Apply Definition 92.172 to the admitted sequence $E_{\bullet+n}$. Its shifted family $(E_{\bullet+n})_{\bullet+1}$ is admitted as a measurable sequence in the same certificate surface. By the defining recursion, this shifted family is $E_{\bullet+\mathsf{E1}(n)}$, so the successor case follows. □

Theorem 92.177 (Measure finite-prefix head-tail readback)**.** *Let $\mathcal{M}$ be a $\mathsf{Measure}^\uparrow$ certificate surface as in Definition 92.15, let $E_\bullet$ be a measurable sequence in that surface, and let n be a unary history with $\mathsf{Unary}(n)$. Then the countable endpoint of $E_\bullet$ is classified by the finite-prefix head-tail endpoint:*

$$\Sigma_\mu(E_\bullet) \sim_R \mathsf{HT}_n(E_\bullet).$$

Equivalently, expanding the recursion of Definition 92.175, this is the right-nested finite-prefix readback

$$\Sigma_\mu(E_\bullet) \sim_R \mu(E_0) +_R \mu(E_1) +_R \cdots +_R \mu(E_{n-1}) +_R \Sigma_\mu(E_{\bullet+n}),$$

with the final tail endpoint omitted only in the empty-prefix case by the first recursion clause.

Proof. Prove the stronger assertion uniformly for every admitted measurable input sequence $G_\bullet$: for every unary n, $\Sigma_\mu(G_\bullet) \sim_R \mathsf{HT}_n(G_\bullet)$. The claimed statement is the specialization $G_\bullet = E_\bullet$.

In the empty case, the definition gives $\mathsf{HT}_\mathsf{e}(G_\bullet) = \Sigma_\mu(G_\bullet)$. Reflexivity of the $\mathsf{Real}^\uparrow$ classifier from Definition 38.9 gives

$$\Sigma_\mu(G_\bullet) \sim_R \mathsf{HT}_\mathsf{e}(G_\bullet).$$

For the one-step case, let $n = \mathsf{E1}(m)$ and assume the uniform induction hypothesis at m. Equip $G_\bullet$ with the head-tail tail scope of Definition 92.171. By Definition 92.173, the single-step readback gives

$$\Sigma_\mu(G_\bullet) \sim_R \mu(G_0) +_R \Sigma_\mu(G_{\bullet+1}).$$

By Definition 92.172, the shifted family $G_{\bullet+1}$ is again an admitted measurable sequence. Apply the induction hypothesis to this shifted sequence:

$$\Sigma_\mu(G_{\bullet+1}) \sim_R \mathsf{HT}_m(G_{\bullet+1}).$$

The additive congruence row of the target $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ certificate, cited in Definition 92.15 and Definition 42.1, transports this tail comparison through the fixed left summand $\mu(G_0)$:

$$\mu(G_0) +_R \Sigma_\mu(G_{\bullet+1}) \sim_R \mu(G_0) +_R \mathsf{HT}_m(G_{\bullet+1}).$$

The right-hand endpoint is exactly $\mathsf{HT}_{\mathsf{E1}(m)}(G_\bullet)$ by the successor clause of Definition 92.175. Transitivity of $\sim_R$ composes the single-step head-tail comparison with this transported induction comparison, yielding

$$\Sigma_\mu(G_\bullet) \sim_R \mathsf{HT}_{\mathsf{E1}(m)}(G_\bullet).$$

This closes the unary induction and therefore the finite-prefix readback for $E_\bullet$. □

Definition 92.178 (Measure root consumer acceptance gate)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root consumer acceptance gate is the downstream package whose source side is the root index threshold source of Definition 92.151 and whose outgoing side is the three-face consumer classifier of Definition 92.127. Its accepted rows are exactly the following rows:*

1. *the visible* BHist *event ledger and componentwise measurable-event classifier rows covered by Definition 92.57;*
2. *the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows attached to measure values, finite readbacks, countable sums, probability normalization, and functional-analysis order comparisons;*
3. *the finite disjoint-additivity rows and countable indexed-additivity rows projected by Definition 92.153;*
4. *the integral, probability, and functional-analysis consumer faces of Definition 92.128.*

The gate contains no host set equality, no quotient event equality, no analytic completion object, and no $\mathsf{StandardBridge}$ *row. Those exclusions are part of the accepted package, so a downstream chapter may cite this gate without reopening the root carrier or adding a standard bridge surface.*

Theorem 92.179 (Measure root consumer acceptance coverage)**.** *Every row accepted by Definition 92.178 is covered by the event-row coverage predicate of Definition 92.57, the carrier threshold of Definition 92.152, and the additivity threshold of Definition 92.153. Thus the acceptance gate exposes no downstream row outside the visible* BHist *ledger, the public* $\mathsf{Measure}^{\uparrow}$ *carrier classifier, the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier, and the finite or countable additive rows already present at the root.*

Proof. Unpack Definition 92.178. Its source side is the root index threshold source, and its outgoing side is only the named three-face consumer classifier.

Apply Definition 92.57 to the event part. This places measurable events, empty events, binary unions, countable unions, disjointness witnesses, measure endpoints, countable sums, and head-tail readback rows on visible BHist ledger entries.

Next use Definition 92.152. It supplies the public event carrier, classifier transport, empty-event row, visible ledger, and $\mathsf{Real}^{\uparrow}$ endpoint classifier used by every accepted row.

Finally apply Definition 92.153. The finite disjoint-additivity and countable indexed-additivity rows of the gate are projections of the same root source, so no accepted row requires a host set equality, quotient event equality, analytic completion object, or $\mathsf{StandardBridge}$ row. □

Theorem 92.180 (Measure root consumer acceptance separation)**.** *Inside the acceptance gate of Definition 92.178, the integral, probability, and functional-analysis faces overlap only on the shared* BHist *event ledger and the shared* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows. Probability normalization remains on the probability face, countable-sum and head-tail readback remain on the integral face, and*

relative-difference, monotone-inclusion, and nonnegative endpoint rows remain on the functional-analysis face.

Proof. First apply Definition 92.128. It assigns every outgoing consumer row to the integral face, probability face, functional-analysis face, or to the shared root rows seen through those faces.

Then apply Definition 92.129. That result identifies the only allowed intersections as the visible BHist ledger rows and the shared $\mathsf{Real}^{\uparrow}$ endpoint classifier rows.

The definition of the acceptance gate adds no fourth consumer face and no carrier-level equality row. Therefore the face separation already proved for the consumer classifier is preserved inside the gate: each consumer-specific row stays on its own outgoing face, while the shared rows are exactly the root ledger and endpoint-classifier rows. □

Theorem 92.181 (Measure root consumer acceptance downstream package)**.** *The chapters* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume exactly the acceptance gate of Definition 92.178. Equivalently, their downstream* $\mathsf{Measure}^{\uparrow}$ *input is the single package consisting of root carrier coverage, finite and countable additivity, shared endpoint classification, face separation, and the explicit exclusions of host set equality, quotient event equality, analytic completion, and* $\mathsf{StandardBridge}$ *rows.*

Proof. Start with Definition 92.154. It says that $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consume the root index threshold source and receive no host set equality, quotient event equality, analytic completion row, or $\mathsf{StandardBridge}$ row from $\mathsf{Measure}^{\uparrow}$.

For the probability consumer, apply Definition 92.112. The accepted $\mathsf{ProbSpace}^{\uparrow}$ face is exactly the total-event normalization, empty-zero, finite-additivity, and $\mathsf{Real}^{\uparrow}$ endpoint normalization surface.

For the integral and functional-analysis consumers, apply Definition 92.113. The accepted faces are exactly the measurable-event, nonnegative endpoint, monotone-inclusion, relative-difference, finite-readback, countable-sum, and continuity-from-below rows exposed by the root $\mathsf{Measure}^{\uparrow}$ package.

Finally combine those consumer projections with Definition 92.179 and Definition 92.180. Coverage supplies the common root ledger and additive source, and separation keeps every consumer-specific row on its named face. Hence the three downstream chapters consume the single acceptance gate and no row outside it. □

Definition 92.182 (Measure root threshold obligation package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root threshold obligation package is the ten-field package read at the root* BHist *surface before* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume measure rows:*

1. *traditional target: a constructive measure surface with measurable events, empty event, finite unions, countable unions, relative differences, disjointness witnesses, and nonnegative* $\mathsf{Real}^{\uparrow}$ *endpoints;*
2. *BEDC source:* BHist *presentations, componentwise* hsame, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint comparison, the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row, and the countable-family name rows used by* $\mathsf{StreamName}^{\uparrow}$*;*
3. *carrier: the root* $\mathsf{MeasCode}$ *predicate displayed by Definition 92.14;*
4. *classifier: the public* $\mathsf{Measure}^{\uparrow}$ *classifier* $\sim_M$*, with endpoint movement read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and* $\sim_R$*;*
5. *stability rows: transport for measurable events, visible ledger rows, endpoint rows, countable-family rows, countable-sum rows, and head-tail readback along* $\sim_M$ *and componentwise* hsame*;*

6. *ledger rows: hidden membership, relative-difference, disjointness, endpoint-value, and countable-sum witnesses attached to visible* BHist *rows;*
7. *sigma-additive rows: empty-zero, finite disjoint-union readback, relative-difference additivity, countable disjoint-union comparison, and countable-sum classifier transport;*
8. *consumer rows: the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *projections exposed by Definition 92.86;*
9. *exactness: visible-row exhaustion through Definition 92.57 and Definition 92.13;*
10. *boundary: no host set equality, quotient event equality, analytic completion object, or* StandardBridge *row is asserted at this threshold.*

The package is the threshold-facing reading of Definition 92.143, Definition 92.61, and Definition 92.86.

Theorem 92.183 (Measure root threshold carrier-classifier obligation)**.** *The root threshold obligation package of Definition 92.182 supplies the carrier and classifier transport required at the* $\mathsf{Measure}^{\uparrow}$ *root. If* $h, k : \mathsf{BHist}$ *satisfy* $\mathsf{MeasCode}(h)$ *and* $h \sim_M k$*, then the package transports* $\mathsf{MeasCode}(k)$*, every visible event row, every ledger row, and every* $\mathsf{Real}^{\uparrow}$ *endpoint comparison attached to the carried measure code. The transport uses only componentwise* hsame *on* BHist *rows and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint comparison.*

Proof. First project the carrier and classifier fields from Definition 92.143. Those fields are the root MeasCode and $\sim_M$ rows named again in the threshold package.

Apply Definition 92.144. It reads the seed front as a carrier-classifier obligation and transports the carried measure code, visible rows, ledger rows, and $\mathsf{Real}^{\uparrow}$ endpoint comparisons along the supplied $\sim_M$ comparison.

Next apply Definition 92.62. Its public projection identifies the transported rows with the root NameCert carrier, classifier, ledger, and endpoint rows consumed at the threshold.

Finally use Definition 92.13. Every hidden membership, relative-difference, disjointness, endpoint-value, and countable-sum witness is attached to a visible BHist row, so no extra carrier or quotient event equality enters the threshold carrier-classifier obligation. □

Theorem 92.184 (Measure root threshold sigma-additive obligation)**.** *The root threshold obligation package of Definition 92.182 supplies the sigma-additive face required at the* $\mathsf{Measure}^{\uparrow}$ *root. For every covered pairwise-disjoint measurable sequence* $E_\bullet$ *with covered countable union* U*, it supplies*

$$\mu(U) \sim_R \Sigma_\mu(E_\bullet).$$

It also supplies the empty-zero row, finite disjoint-union readback, relative-difference additivity, head-tail readback, and countable-sum classifier transport exposed by the public root package.

Proof. Start with the sigma-additive fields of Definition 92.143. They include the covered sequence, countable-union, disjointness, endpoint, additive real, and countable-sum rows named by the threshold package.

Apply Definition 92.145. It projects the seed front to the displayed countable disjoint-union comparison and keeps the empty-zero, finite disjoint-union, relative-difference, head-tail, and countable-sum classifier rows visible.

Next apply Definition 92.63. Its public form identifies the same comparison and the same consequence rows as root $\mathsf{Measure}^{\uparrow}$ public rows over BHist and $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$.

The countable-sum endpoint is read through the additive $\mathsf{AbGroup}^{\uparrow}$ real row already listed in the threshold package. Hence the sigma-additive obligation is a visible root package over BHist rows and $\mathsf{Real}^{\uparrow}$ endpoint classification, with no host set equality or standard bridge field. □

Theorem 92.185 (Measure root threshold analysis consumer obligation)**.** *The root threshold obligation package of Definition 92.182 supplies exactly the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer rows available from the public* $\mathsf{Measure}^{\uparrow}$ *root. The exported rows are the covered measurable-event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows, countable-sum rows, head-tail readback, empty-zero, finite disjoint-union readback, relative-difference additivity, monotone inclusion, nonnegative endpoint, total-event normalization, and the explicit no-*$\mathsf{StandardBridge}$ *row.*

Proof. Begin with Definition 92.64. It packages the root carrier-classifier and sigma-additive faces as the public $\mathsf{Measure}^{\uparrow}$ surface consumed by the three analysis-facing chapters.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The projection consumes the covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, countable-sum rows, head-tail readback, and finite disjoint-union readback from the same root threshold package.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88. The projection consumes total-event normalization together with empty-zero and finite disjoint-union rows, all read from the public root package.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89. The projection consumes nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows.

The exactness field of the threshold package identifies these projections as the complete consumer boundary. Thus the package opens the three downstream faces while excluding host set equality, quotient event equality, analytic completion objects, and standard bridge claims. □

92.7 Measure root finite-support consumer front

Definition 92.186 (Measure root finite-support consumer front)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root finite-support consumer front is the ten-field package that exposes the finite-support rows needed by downstream analysis consumers:*

1. *traditional target: finite-support measurable events with readback values in the* $\mathsf{Real}^{\uparrow}$ *additive endpoint surface;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{Real}^{\uparrow} + \mathsf{AbGroup}^{\uparrow}$*;*
3. *carrier: a covered finite support spine of displayed measurable* BHist *event rows together with a constant empty tail;*
4. *classifier: the public* $\mathsf{Measure}^{\uparrow}$ *event classifier together with the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier for each displayed value and finite sum;*
5. *stability: componentwise event transport, endpoint transport, additive congruence, and support-spine transport;*
6. *ledger: the finite support spine, the constant empty tail, the countable-union row, and the finite union tree read from the same displayed* BHist *rows;*
7. *exactness: finite-support sum readback through Definition 92.23;*
8. *displayed classification iff: a consumer row belongs to the front exactly when it is read from the covered event spine, the empty tail, the finite union tree, and the* $\mathsf{Real}^{\uparrow}$ *finite-sum endpoint;*

9. *consumer boundary:* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *may consume only the rows named in Definition 92.189;*
10. *standard bridge: no host set equality, quotient event equality,* $\mathsf{Real}^{\uparrow}$ *completeness row, or* $\mathsf{StandardBridge}$ *row is part of this front.*

The front is a finite-support face of the root consumer package of Definition 92.86, not a separate analytic completion object.

Theorem 92.187 (Measure root finite-support carrier readback)**.** *The front of Definition 92.186 supplies covered measurable-event rows and empty-tail stream rows from Definition 92.57, Definition 92.91, and Definition 92.23. In particular, every finite support spine visible to the front is a* BHist *event ledger row whose constant empty tail is classified by the public* $\mathsf{Measure}^{\uparrow}$ *carrier and the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier.*

Proof. Start with Definition 92.57. It places the displayed measurable events, the empty event, finite union rows, countable union rows, and measure endpoints on visible BHist ledger entries.

Next apply Definition 92.91. The countable name carrier supplies the stream-indexed family in which the finite support spine is followed by the constant empty tail, and it transports the displayed event rows through the same componentwise classifier.

Finally use Definition 92.23. Its finite-support hypothesis identifies the nonempty spine and the empty tail as one countable family, so the front reads the carrier, tail, and endpoint rows without adding a host-level equality or quotient event comparison. □

Theorem 92.188 (Measure root finite-support additivity readback)**.** *For every pairwise-disjoint finite support spine exposed by Definition 92.186, the measure of the displayed finite union reads back to the* $\mathsf{Real}^{\uparrow}$ *finite sum of the displayed support values. The readback is obtained by induction on the support spine using the empty-zero row, finite disjoint-union additivity, and countable empty-tail readback.*

Proof. The base case is the empty support spine. The constant empty stream has empty countable union, and Definition 92.25 classifies its measure endpoint with the additive zero in the $\mathsf{Real}^{\uparrow}$ $\mathsf{AbGroup}^{\uparrow}$ row.

For the induction step, split a nonempty support spine into its head event and tail spine. Pairwise disjointness gives the displayed binary union row between the head and the finite union of the tail, while the induction hypothesis reads the tail union as the finite sum of the tail values.

Apply Definition 92.26 to the head and tail union. Additive congruence in the $\mathsf{Real}^{\uparrow}$ $\mathsf{AbGroup}^{\uparrow}$ row transports the tail readback through the right summand, and associativity identifies the result with the finite sum of the whole spine.

The empty tail of the surrounding countable stream is handled by Definition 92.23. Thus the finite union endpoint and the countable family endpoint agree with the same $\mathsf{Real}^{\uparrow}$ finite-support sum. □

Theorem 92.189 (Measure root finite-support downstream exhaustion)**.** $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume exactly the finite-support rows of Definition 92.186 through Definition 92.86, Definition 92.87, Definition 92.88, and Definition 92.89. No downstream face receives host set equality, quotient event equality, a* $\mathsf{Real}^{\uparrow}$ *completeness row, or a standard bridge row from this front.*

Proof. Begin with Definition 92.86. Its outgoing rows are the root $\mathsf{Measure}^{\uparrow}$ consumer rows built from covered events, endpoint classification, countable sums, head-tail readback, empty-zero, finite disjoint additivity, and relative-difference or normalization projections.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The integral face reads covered event rows, endpoint rows, countable-sum rows, head-tail readback, and finite disjoint-union readback; the finite-support front supplies exactly the finite spine and empty-tail rows among these.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88. Its finite-support consumption is limited to the total-event normalization, empty-zero, and finite disjoint-union rows already present in the root consumer package.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89. The functional analysis face reads nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows, and it uses the finite-support front only through their covered event and finite-sum projections.

The tenth field of Definition 92.186 excludes host set equality, quotient event equality, $\mathsf{Real}^{\uparrow}$ completeness, and standard bridge rows. Since each downstream theorem above is a projection of the same root consumer package, these three faces exhaust the finite-support consumer surface. □

92.8 Measure root event boundary package

The root event boundary is the public face of the $\mathsf{Measure}^{\uparrow}$ root threshold seen by the downstream analysis chapters. It keeps the boundary over BHist presentations, componentwise hsame event comparison, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ endpoint comparison, $\mathsf{StreamName}^{\uparrow}$ countable names, and the displayed measure ledger rows.

Theorem 92.190 (Measure root event-boundary carrier obligation)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root event boundary exposes exactly the carried* BHist *measure code, visible event rows, empty-event row, finite union rows, countable union rows, relative-difference row,* $\mathsf{Real}^{\uparrow}$ *endpoint row, and countable-sum witnesses supplied by Definition 92.1, Definition 92.182, and Definition 92.183. Every exposed row is a displayed* BHist *row or a named endpoint row, and the boundary adds no host set equality, quotient event identity, analytic completion object, or standard bridge row.*

Proof. Start from the carrier surface of Definition 92.1. It names the carried measure code, the measurable-event family, visible event rows, empty-event row, finite and countable union rows, relative-difference row, nonnegative endpoint row, and countable-sum witnesses.

Next read the same rows through Definition 92.182. The threshold package places them at the root over BHist presentations, componentwise hsame, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ endpoint comparison, and $\mathsf{StreamName}^{\uparrow}$ countable names.

Apply Definition 92.183. It repacks the carrier and visible ledger rows as root boundary data and keeps endpoint movement inside the $\mathsf{Real}^{\uparrow}$ classifier.

The exactness clauses of the carrier and threshold packages identify the displayed ledger as the whole boundary surface. Therefore the event boundary contains precisely the listed BHist rows and endpoint witnesses, with no additional host equality or quotient event row. □

Theorem 92.191 (Measure root event-boundary classifier obligation)**.** *The public classifier on the* $\mathsf{Measure}^{\uparrow}$ *root event boundary is componentwise* hsame *on the visible event rows together with* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *classifier agreement on endpoint rows. Equivalently, the boundary classifier is the root-facing instance of Definition 92.7 projected through Definition 92.183: it compares displayed event rows and* $\mathsf{Real}^{\uparrow}$ *endpoints only, and it never asserts host set equality or quotient event identity.*

Proof. Project the public classifier from Definition 92.7. Its event component reads visible event presentations through hsame, while its endpoint component reads measure values through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ and the $\mathsf{Real}^{\uparrow}$ classifier.

Now apply Definition 92.183. The root threshold projection restricts the classifier to the displayed BHist event rows, ledger rows, and endpoint rows carried by the boundary.

It remains to identify what the projection excludes. Since the carrier surface and the threshold package list only BHist rows, componentwise hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ comparison, and displayed ledger witnesses, the classifier has no field in which host set equality or quotient event identity could be read. □

Theorem 92.192 (Measure root event-boundary downstream unblock)**.** *The same* $\mathsf{Measure}^\uparrow$ *root event boundary supplies the downstream rows consumed by* $\mathsf{Integral}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, *and* $\mathsf{FunctionalAnalysis}^\uparrow$. *The unblocking rows are exactly the countable-additivity and analysis-consumer obligations of Definition 92.184 and Definition 92.185, read through Definition 92.87, Definition 92.88, and Definition 92.89. They remain scoped to* BHist, hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, *and the displayed* $\mathsf{Measure}^\uparrow$ *ledger rows.*

Proof. First build the additive part of the boundary. Apply Definition 92.184. It supplies the empty-zero row, finite disjoint-union readback, relative-difference additivity, countable disjoint-union comparison, head-tail readback, and countable-sum classifier transport over the same BHist rows.

Next build the consumer projection. Apply Definition 92.185. It reads the $\mathsf{Integral}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, and $\mathsf{FunctionalAnalysis}^\uparrow$ outgoing rows from the root package without adding a host carrier, quotient event, analytic completion object, or standard bridge row.

For the integral consumer, use Definition 92.87. The consumed rows are the covered measurable-event rows, $\mathsf{Real}^\uparrow$ endpoint classifier rows, countable-sum rows, head-tail readback, and finite disjoint-union readback.

For the probability consumer, use Definition 92.88. The consumed rows are the total-event normalization row, empty-zero row, and finite disjoint-union readback row.

For the functional-analysis consumer, use Definition 92.89. The consumed rows are the non-negative endpoint, monotone-inclusion, relative-difference, and countable-sum classifier rows.

All three consumer projections point back to the same root event boundary. Thus the boundary supplies the three downstream faces through countable additivity and analysis-consumer rows while staying inside BHist, hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, and the displayed $\mathsf{Measure}^\uparrow$ ledger. □

Definition 92.193 (Measure root public obligation basis)**.** *The* $\mathsf{Measure}^\uparrow$ *root public obligation basis is the counted ten-field basis that exposes the constructive root measure certificate before any analytic completion or bridge layer is admitted:*

1. *traditional target: a constructive measure root with visible measurable-event rows, countable-family rows, and* $\mathsf{Real}^\uparrow$ *endpoint values;*
2. *BEDC source:* BHist, *componentwise* hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, *and the additive* $\mathsf{AbGroup}^\uparrow$ *real rows used for countable endpoint sums;*
3. *carrier: the root predicate* MeasCode *on* BHist*;*
4. *classifier: the public measure classifier* $\sim_M$*, with endpoint motion read through* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ *and* $\sim_R$*;*
5. *stability rows: carrier and classifier transport, visible event-row transport, endpoint-row transport, and countable-family transport along* $\sim_M$ *and componentwise* hsame*;*
6. *ledger: hidden measurable-event witnesses, endpoint-value witnesses, disjointness witnesses, relative-difference witnesses, and countable-sum witnesses attached to the displayed rows they justify;*

7. *core theorem list: Definition 92.194, Definition 92.195, Definition 92.196, and Definition 92.197;*
8. *exactness: visible-row exhaustion for the carrier face, the additive face, and the consumer face through the rows of Definition 92.57;*
9. *standard bridge: no* StandardBridge *is asserted or imported by this basis;*
10. *status: paper proof target under* BEDC.Derived.MeasureUp.

This basis is read from Definition 92.143, Definition 92.182, and Definition 92.162. It is the public obligation surface available to downstream analysis chapters.

Lean4 (definition) `BEDC.Derived.MeasureUp.MeasureRootPublicObligationBasis`

Theorem 92.194 (Measure root public obligation basis carrier)**.** *The basis of Definition 92.193 supplies the carrier and classifier transport needed by the public* $\mathsf{Measure}^{\uparrow}$ *root. If* $h, k : \mathsf{BHist}$ *satisfy* $\mathsf{MeasCode}(h)$ *and* $h \sim_M k$*, then the basis transports* $\mathsf{MeasCode}(k)$*, every visible event row, every endpoint row, every countable-family row, and every ledger witness attached to the carried measure code.*

Proof. Start with the carrier and classifier fields of Definition 92.193. They are the MeasCode and $\sim_M$ rows first exposed in Definition 92.143.

Apply Definition 92.144. It transports the carried measure code, visible event rows, ledger rows, and $\mathsf{Real}^{\uparrow}$ endpoint comparisons along the supplied $\sim_M$ comparison.

Next apply Definition 92.183. The threshold reading keeps the same transported rows and adds the public countable-family and endpoint-readback boundary used by the root package.

The ledger field of the basis attaches the hidden witnesses only to displayed BHist rows. Therefore the transported rows exhaust the public carrier and classifier obligation without adding host set equality or quotient event equality. □

Theorem 92.195 (Measure root public obligation basis additivity)**.** *The basis of Definition 92.193 supplies the public additive face of the* $\mathsf{Measure}^{\uparrow}$ *root: empty-zero, finite readback, relative-difference additivity, countable disjoint-additivity, and countable-sum transport through the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row.*

Proof. Project the additive fields of Definition 92.193. They name the covered empty event, finite disjoint unions, relative differences, countable disjoint families, countable unions, and countable-sum endpoints in the same visible BHist ledger.

Apply Definition 92.145. It supplies the root comparison between the value of a covered countable disjoint union and the corresponding countable endpoint sum.

Apply Definition 92.184. It keeps the empty-zero row, finite readback row, relative-difference row, head-tail row, and countable-sum classifier transport visible at the public threshold.

Finally apply Definition 92.164. This countable-premeasure reading identifies finite readback and relative-difference additivity as projections of the same countable additive surface, with endpoint movement through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ and the additive $\mathsf{AbGroup}^{\uparrow}$ real row. □

Theorem 92.196 (Measure root public obligation basis consumer)**.** *The basis of Definition 92.193 exports exactly the* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *rows available from the public* $\mathsf{Measure}^{\uparrow}$ *root. No other analysis row is part of this basis.*

Proof. Begin with the consumer field of Definition 92.193. It lists only the three outgoing faces $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$.

Apply Definition 92.185. It reads those three faces from the root threshold package and identifies the covered event rows, endpoint classifier rows, countable-sum rows, empty-zero, finite readback, relative-difference, monotone inclusion, nonnegative endpoint, and total-event normalization rows available to them.

Next apply Definition 92.165. It restricts the same three consumers to the countable-premeasure front and keeps the outgoing rows attached to covered measurable-event and endpoint rows.

The exactness clause of the basis has no fourth consumer slot and no analytic completion slot. Hence the exported consumer boundary is exactly the stated triad and no larger analysis interface is claimed. □

Theorem 92.197 (Measure root public obligation basis exactness)**.** *The carrier face, additive face, and consumer face of Definition 92.193 form the deps-ready* $\mathsf{Measure}^{\uparrow}$ *root package. The package asserts no host set equality, no quotient event equality, no analytic completion object, and no standard bridge.*

Proof. First read the carrier face through Definition 92.194. It supplies the public MeasCode carrier, $\sim_M$ classifier transport, visible event rows, endpoint rows, countable-family rows, and attached ledger witnesses.

Next read the additive face through Definition 92.195. It supplies the empty-zero, finite readback, relative-difference, countable disjoint-additivity, and countable-sum transport rows over the same visible ledger.

Finally read the consumer face through Definition 92.196. It exports only the $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ projections from that public root package.

The exactness field of Definition 92.193 says that every consumed hidden witness is attached to a displayed row covered by Definition 92.57. Since none of the three faces contains a host-set equality field, quotient-event equality field, analytic completion object, or StandardBridge field, the resulting package is precisely the deps-ready public obligation basis. □

92.9 Root premeasure event-value bridge

Definition 92.198 (Measure root premeasure event-value bridge)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root premeasure event-value bridge is the package whose fields are the countable premeasure carrier and classifier rows of Definition 92.162, the countable additivity row of Definition 92.164, the public obligation basis rows of Definition 92.193, and the* NameCert *obligation package of Definition 92.27. Its visible event-value surface consists only of covered measurable-event* BHist *rows, their* $\mathsf{Real}^{\uparrow}$ *nonnegative value endpoints, the classifier transports for those rows, and the ledger exactness rows that tie hidden measurable witnesses to displayed events.*

Theorem 92.199 (Measure root premeasure finite-additivity bridge)**.** *The package of Definition 92.198 exports finite disjoint-union additivity by composing Definition 92.164, Definition 92.195, and Definition 92.27.*

Proof. Project the countable additivity field from the bridge package. By Definition 92.164, the countable-premeasure front reads finite disjoint unions as finite-support instances of the same countable disjoint-additivity row.

Next project the public additive face. Apply Definition 92.195; it exposes empty-zero, finite readback, relative-difference additivity, countable disjoint-additivity, and countable-sum transport over the same displayed event-value ledger.

Finally read the NameCert obligation package through Definition 92.27. Its finite disjoint-union theorem is attached to the carrier, classifier, nonnegative endpoint, and ledger exactness rows already present in the bridge, so the finite additivity claim is the exported projection of the package. □

Theorem 92.200 (Measure root premeasure downstream threshold bridge)**.** *Downstream* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers may read only the event carrier and classifier, countable additivity, finite additivity, nonnegative value, and ledger exactness rows exposed by the bridge of Definition 92.198.*

Proof. Start with the bridge package and discard every field not named by its event-value surface. The remaining carrier and classifier rows are exactly the covered measurable-event BHist rows, value endpoint rows, and transport rows inherited from Definition 92.162.

For additive information, use Definition 92.164 and Definition 92.199. These give the countable disjoint-additivity row and its finite disjoint-union projection without adding a consumer-specific carrier.

For public visibility and exactness, apply Definition 92.197 and Definition 92.27. The first pins the rows to the displayed public basis, while the second supplies the nonnegative value and ledger exactness obligations consumed by the three downstream chapters.

The package definition has no further outgoing field for analytic completion, host set equality, quotient event equality, or a standard bridge. Hence $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ can read exactly the listed rows and no larger root surface from this bridge. □

Definition 92.201 (Measure root-kernel obligation source)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root-kernel obligation source is the counted ten-field per-object template that records the* BHist*-scoped source from which the public root measure obligations are read:*

1. *object code: a* BHist *row* h *carrying the visible measurable-event rows of the* $\mathsf{Measure}^{\uparrow}$ *ledger;*
2. *classifier source: componentwise* hsame *on histories together with the induced measure classifier* $\sim_M$ *on visible measure rows;*
3. *endpoint certificate:* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *for every displayed endpoint value and the* $\mathsf{StreamName}^{\uparrow}$ *rows used to name countable event families;*
4. *carrier face: the* MeasCode *predicate on* BHist*, read through the carrier surface and the root public obligation basis;*
5. *event boundary: exactly the visible event rows covered by Definition 92.57 and the root event boundary package;*
6. *additive face: empty-zero, finite readback, relative-difference, countable disjoint-additivity, and countable-sum rows over the same ledger;*
7. *threshold face: the root threshold package and root threshold obligation package, including endpoint readback and countable-family transport;*
8. *premeasure face: the countable premeasure front over the covered event rows and the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row;*
9. *downstream face: only the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer rows already named in the consumer boundary;*
10. *exclusions: no host set equality, quotient event equality, analytic completion object, or standard bridge row is exported by this source.*

Thus each root-kernel object is counted by a single public source over BHist, hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, *visible event rows, and the* $\mathsf{Measure}^\uparrow$ *ledger.*

Theorem 92.202 (Measure root-kernel carrier-classifier obligation)**.** *For every root-kernel object described by Definition 92.201, the source supplies the* $\mathsf{Measure}^\uparrow$ *carrier and classifier obligation:* $\mathsf{MeasCode}$ *is read on* BHist, *transported along componentwise* hsame *and* $\sim_M$, *and all visible event, endpoint, countable-family, and ledger rows remain attached to the transported carrier.*

Proof. Start from the object code, classifier source, endpoint certificate, and carrier face of Definition 92.201. These fields place the carrier in BHist and make hsame, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, and $\mathsf{StreamName}^\uparrow$ part of the same visible ledger.

Apply Definition 92.194. It transports the carrier, classifier, event rows, endpoint rows, and countable-family rows through the public obligation basis.

Apply Definition 92.183. The root threshold package keeps the same transported rows visible at the threshold face, including endpoint readback and countable-family transport.

Finally use Definition 92.190. The event boundary package identifies the displayed event ledger as the whole carrier boundary. Since the source has no host set equality or quotient event equality field, the carrier-classifier obligation is exactly the stated BHist-scoped one. □

Theorem 92.203 (Measure root-kernel additivity obligation)**.** *For every root-kernel object described by Definition 92.201, the source supplies the additive obligation required by the public* $\mathsf{Measure}^\uparrow$ *root: empty-zero, finite readback, relative-difference additivity, countable disjoint-additivity, and countable-sum transport through* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, *and the additive* $\mathsf{AbGroup}^\uparrow$ *real row.*

Proof. Read the additive, threshold, and premeasure faces of Definition 92.201. They refer to the same covered event rows, the same endpoint certificates, and the same countable-family names.

Apply Definition 92.195. This gives the empty-zero, finite readback, relative-difference, countable disjoint-additivity, and countable-sum transport rows over the public ledger.

Apply Definition 92.184. The root threshold package keeps the additive rows attached to the endpoint and countable-family boundary.

Apply Definition 92.164. The countable premeasure front reads those same rows through $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{StreamName}^\uparrow$, and the additive $\mathsf{AbGroup}^\uparrow$ real row. The exclusions field exports no analytic completion object, so the additivity obligation remains a root-kernel obligation rather than a completed measure-space assertion. □

Theorem 92.204 (Measure root-kernel downstream consumer obligation)**.** *For every root-kernel object described by Definition 92.201, the source exports one downstream consumer boundary: the* $\mathsf{Integral}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, *and* $\mathsf{FunctionalAnalysis}^\uparrow$ *rows obtained from the root public obligation basis, root threshold package, event boundary package, countable premeasure front, and consumer rows. No fourth consumer, analytic completion object, host set equality, quotient event equality, or standard bridge row is exported.*

Proof. Begin with the downstream face of Definition 92.201. It names only $\mathsf{Integral}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, and $\mathsf{FunctionalAnalysis}^\uparrow$ as outgoing consumer chapters.

Apply Definition 92.196. The root public obligation basis exports exactly those three rows from the public $\mathsf{Measure}^\uparrow$ root.

Apply Definition 92.192. The event boundary package shows that the consumer rows are attached to covered visible event rows, endpoint rows, and countable-additivity rows.

Finally apply Definition 92.165. The countable premeasure front is the common source consumed by the same three downstream chapters. The exclusions field and the consumer boundary together leave no slot for a fourth consumer, analytic completion object, host set equality, quotient event equality, or standard bridge row. □

Definition 92.205 (Measure root minimum NameCert obligation triad)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root minimum* $\mathsf{NameCert}$ *obligation triad is the three-field root package read from* BHist *presentations, the componentwise* hsame *classifier,* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint transport, and* $\mathsf{StreamName}^{\uparrow}$ *countable-family rows:*

1. *carrier-classifier field: the measurable-event carrier* $\mathsf{MeasCode}$*, the public classifier* $\sim_M$*, visible event-row transport, endpoint transport through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*, and carrier transport along componentwise* hsame*;*
2. *countable-additivity field: StreamName-indexed measurable sequence coverage, pairwise-disjoint and countable-union readback, countable-sum endpoint rows, and the* $\mathsf{Real}^{\uparrow}$ *comparison between the value of the union and the displayed countable endpoint;*
3. *downstream ledger field: the rows available to* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*, namely the carrier/classifier rows, countable-additivity rows, nonnegative endpoint rows, and ledger rows attached to visible measurable-event witnesses.*

The triad is a minimum public root surface: it contains no host set equality, no quotient equality of event codes, no analytic completion object, and no open-cover compactness row.

Theorem 92.206 (Measure root minimum carrier-classifier obligation)**.** *The triad of Definition 92.205 supplies the* $\mathsf{Measure}^{\uparrow}$ *root carrier predicate, public classifier, and carrier transport only through the displayed BEDC rows of Definition 92.6, Definition 92.7, Definition 92.17, Definition 92.18, and Definition 92.193.*

Proof. Read the first field of Definition 92.205. Its carrier is the BHist measurable-event carrier and its classifier is the componentwise hsame relation with endpoint motion read through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$.

The inventory in Definition 92.6 names exactly those carrier, classifier, stability, and ledger rows. The classifier specification in Definition 92.7 then restricts comparison to event-witness transport and $\mathsf{Real}^{\uparrow}$ endpoint comparison.

Apply Definition 92.17 and Definition 92.18. They identify the root $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ carrier, classifier, and stability face with the displayed event, endpoint, and ledger rows.

Finally compare with Definition 92.193. Its carrier and classifier fields expose the same rows and no larger equality principle. Thus the triad supplies the root carrier-classifier obligation only through the displayed BEDC rows. □

Theorem 92.207 (Measure root minimum countable-additivity obligation)**.** *The triad of Definition 92.205 supplies measurable-sequence coverage, disjoint-union readback, and* $\mathsf{Real}^{\uparrow}$ *endpoint comparison for the* $\mathsf{Measure}^{\uparrow}$ *countable-additivity row. Every countable family consumed by this row is read through* $\mathsf{StreamName}^{\uparrow}$*, componentwise* hsame*, and the public ledger of Definition 92.193.*

Proof. Project the second field of Definition 92.205. It names the StreamName-indexed measurable family, the pairwise-disjoint witness, the countable union row, the countable-sum endpoint, and the $\mathsf{Real}^{\uparrow}$ comparison attached to that endpoint.

The carrier inventory of Definition 92.6 supplies the measurable sequence, disjointness, union, and countable-sum ledger rows. The classifier specification of Definition 92.7 keeps their movement componentwise, with endpoint motion through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$.

Apply Definition 92.18. Its countable-endpoint and σ-additivity projections supply the stated comparison between the value of the visible disjoint union and the displayed countable endpoint.

The public basis of Definition 92.193 records these rows as the root additive face. Therefore the triad contains the countable-additivity obligation exactly as measurable-sequence coverage, disjoint-union readback, and $\mathsf{Real}^{\uparrow}$ endpoint comparison. □

Theorem 92.208 (Measure root minimum downstream consumer obligation)**.** *The triad of Definition 92.205 permits* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *to consume exactly the carrier/classifier rows, countable-additivity rows, nonnegative endpoint rows, and ledger rows displayed by the* $\mathsf{Measure}^{\uparrow}$ *root. It exports no host set equality, no quotient equality, and no open-cover compactness.*

Proof. Read the downstream ledger field of Definition 92.205. It names only the three consumers $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ and restricts their input to carrier/classifier, countable-additivity, nonnegative endpoint, and visible ledger rows.

The root carrier inventory and classifier specification provide the common input rows for all three consumers. The root NameCert carrier-classifier and stability-exactness theorems identify these rows with the public $\mathsf{Measure}^{\uparrow}$ root face.

Apply Definition 92.193. Its consumer and exactness fields attach every downstream witness to displayed measurable-event, endpoint, disjointness, relative-difference, and countable-sum rows.

Since none of those cited rows contains a host set equality field, quotient event equality field, or open-cover compactness field, the triad exports exactly the downstream consumer obligation stated above. □

92.10 Measure root public face exactness

The public face exactness package gathers the root $\mathsf{Measure}^{\uparrow}$ endpoint fiber that downstream chapters cite when they need the concrete BEDC event ledger rather than a transported analytic interface.

Definition 92.209 (Measure root public face exactness)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root public face exactness package is the concrete endpoint fiber over* BHist *whose carrier rows are the visible measurable event histories, whose classifier rows are componentwise* hsame *rows, and whose endpoint rows are classified through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and* $\mathsf{StreamName}^{\uparrow}$*. It consists of exactly the following public rows:*

1. *the displayed* BHist *measurable membership rows, empty event rows, relative-difference rows, disjointness rows, countable-union rows, and countable-family rows covered by Definition 92.57;*
2. *the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier rows attached to measure values, finite readbacks, nonnegative endpoints, total-event normalization, monotone inclusion, and countable sums;*
3. *the minimum root* $\mathsf{NameCert}$ *carrier-classifier, countable-additivity, and downstream ledger fields of Definition 92.205;*
4. *the public sigma-algebra carrier, empty-zero, and countable disjoint-additivity rows of Definition 92.214;*
5. *the outgoing* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *faces accepted by Definition 92.178.*

This package introduces no host set equality, no quotient event equality, no analytic completion object, and no standard bridge row.

Theorem 92.210 (Measure root public face coverage)**.** *Every endpoint fiber in Definition 92.209 is covered by the event-row coverage predicate of Definition 92.57, the minimum root* NameCert *triad of Definition 92.205, and the sigma-algebra public basis of Definition 92.214.*

Proof. Start with the listed rows of Definition 92.209. The event side consists only of displayed BHist measurable membership, empty-event, relative-difference, disjointness, countable-union, and countable-family rows.

Apply Definition 92.57. It places each event row in the visible ledger, together with the attached measure-value and countable-sum endpoint rows.

Next apply Definition 92.205. Its carrier-classifier field supplies the hsame event classifier, its countable-additivity field supplies the $\mathsf{StreamName}^{\uparrow}$ family and $\mathsf{Real}^{\uparrow}$ endpoint-sum rows, and its downstream ledger field keeps those rows visible to the three named consumers.

Finally use Definition 92.214, Definition 92.218, and Definition 92.126. The public basis gives the carrier, empty-zero, and countable disjoint-additivity rows; the real-stream obligation attaches StreamName observation names and endpoint comparisons to the same event history; the source package confirms that all source witnesses remain ledger rows rather than host equalities or analytic completion data. □

Theorem 92.211 (Measure root public face determinacy)**.** *For a displayed event row of Definition 92.209, the event display together with its* $\mathsf{Real}^{\uparrow}$ *endpoint classifier determines one public* $\mathsf{Measure}^{\uparrow}$ *endpoint fiber. The determinacy is a BEDC endpoint-fiber determinacy statement; it is not host event equality and not quotient equality of event codes.*

Proof. Fix a displayed event row in the public face. By Definition 92.209, the row carries a componentwise hsame classifier and a $\mathsf{Real}^{\uparrow}$ endpoint classifier through $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$.

Apply Definition 92.220. It transports the carried $\mathsf{Measure}^{\uparrow}$ root row and its visible ledger along the displayed event history, StreamName observation, and real endpoint classifier.

Now use Definition 92.215 and Definition 92.167. The sigma-algebra carrier keeps the public event row in the root carrier, while the real-stream carrier places the observation name and endpoint comparison on that same row.

The consumer and source boundaries do not identify host events. By Definition 92.179 and Definition 92.123, the accepted row remains a visible ledger row with endpoint data attached. Thus the displayed event and endpoint classifier determine exactly one public endpoint fiber of the BEDC root surface, with no host set equality, quotient event equality, analytic completion object, or standard bridge row added. □

Theorem 92.212 (Measure root public face ledger exhaustion)**.** *Every ledger entry consumed by Definition 92.209 is a measurable membership row, relative-difference row, disjointness row, measure-value row, or countable-sum row supplied by Definition 92.13. No other source witness is part of the public endpoint fiber.*

Proof. Read the ledger clause of Definition 92.209. It contains exactly the visible event rows and endpoint rows listed in Definition 92.57, plus the downstream faces accepted by the root consumer gate.

Apply Definition 92.13. Its exactness clause exhausts the consumed source witnesses by measurable membership, relative-difference, disjointness, measure-value, and countable-sum rows.

Next apply Definition 92.125. It reads the same witness list through the source-closure package and keeps probability normalization, integral premeasure rows, functional-analysis rows, and bridge rows outside the source ledger.

Finally use Definition 92.221 and Definition 92.181. The additivity scope accounts for the $\mathsf{StreamName}^{\uparrow}$ countable-family endpoint rows, and the acceptance package shows that downstream consumers receive only projections of the exhausted ledger. Therefore no host set equality, quotient event equality, analytic completion object, or standard bridge row enters the public endpoint fiber. □

Theorem 92.213 (Measure root public face consumer separation)**.** *Inside the public endpoint fiber of Definition 92.209, the chapters* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consume only the shared root event ledger, the shared* $\mathsf{Real}^{\uparrow}$ *endpoint-classifier rows, and their named face rows. The public face supplies no mixed consumer row and no analytic completion interface.*

Proof. Begin with the outgoing rows in Definition 92.209. They are exactly the three faces accepted by Definition 92.178.

Apply Definition 92.180. It identifies the only overlaps among the integral, probability, and functional-analysis faces as the shared BHist event ledger and the shared $\mathsf{Real}^{\uparrow}$ endpoint classifier rows.

Next apply Definition 92.222 and Definition 92.169. These theorems read the same consumers through displayed event rows, StreamName observations, endpoint comparisons, finite readbacks, normalization rows, monotone rows, and relative-difference rows.

Finally use Definition 92.126. The source package is built before any outgoing consumer face is read, so consumer rows cannot become source hypotheses and cannot merge the named faces. Hence the public endpoint fiber separates $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ exactly as stated, with no host set equality, quotient event equality, analytic completion object, or standard bridge row. □

Definition 92.214 (Measure root sigma-algebra public basis)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root sigma-algebra public basis is the counted ten-field basis that exposes the public measurable-event and countable-additive rows available at the root:*

1. *traditional target: a constructive sigma-algebra measure root whose public rows name measurable events, complements, countable unions, the empty event, endpoint values, and countable additive sums;*
2. *BEDC source:* BHist*, componentwise* hsame*,* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*,* $\mathsf{StreamName}^{\uparrow}$*, and the additive* $\mathsf{AbGroup}^{\uparrow}$ *real rows used for endpoint addition and countable endpoint sums;*
3. *carrier: the root predicate* $\mathsf{MeasCode}$ *on* BHist*;*
4. *classifier: the public measure classifier* $\sim_M$*, with endpoint comparison read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *and* $\sim_R$*;*
5. *stability rows: transport of measurable-event rows, complement rows, countable-union rows, empty-event rows, endpoint rows, countable-sum rows, and attached ledger witnesses along* $\sim_M$ *and componentwise* hsame*;*
6. *ledger: hidden measurable-event witnesses, complement witnesses, countable-union witnesses, empty-event witnesses, endpoint-value witnesses, disjointness witnesses, and countable-sum witnesses attached only to the displayed rows they justify;*
7. *core theorem list: Definition 92.215, Definition 92.216, and Definition 92.217;*
8. *exactness: every public carrier, classifier, empty-zero, and countable disjoint-additivity assertion is exhausted by the displayed rows of Definition 92.57;*

9. *standard bridge: no* StandardBridge *is asserted or imported by this basis;*
10. *status: paper proof target under* BEDC.Derived.MeasureUp.

No host set equality, quotient event equality, analytic completion object, or standard bridge is part of this basis.

Theorem 92.215 (Measure root sigma-algebra public carrier)**.** *The basis of Definition 92.214 transports the public sigma-algebra carrier face of* $\mathsf{Measure}^{\uparrow}$*. If* $h, k : \mathsf{BHist}$ *satisfy* $\mathsf{MeasCode}(h)$ *and* $h \sim_M k$*, then the basis transports measurable-event rows, complement rows, countable-union rows, empty-event rows, endpoint rows, and attached ledger witnesses from* h *to* k*.*

Proof. Start from the carrier face of Definition 92.214. Its carrier and classifier fields are the public MeasCode and $\sim_M$ rows, and its stability field names the event, complement, countable-union, empty-event, endpoint, and ledger rows to be transported.

Apply Definition 92.194. This supplies the carried MeasCode row, the visible event rows, the endpoint rows, the countable-family rows, and the attached ledger witnesses after the supplied comparison $h \sim_M k$.

Next apply Definition 92.163. It reads the same transport through the countable-premeasure front, so the complement, countable-union, and empty-event rows remain attached to the public root ledger.

Finally use Definition 92.57. The coverage definition identifies which visible rows exhaust the public carrier face, so the transported witnesses remain row-attached and no host set equality, quotient event equality, analytic completion object, or standard bridge is introduced. □

Theorem 92.216 (Measure root empty-event zero public row)**.** *The basis of Definition 92.214 exposes the empty measurable event as a public sigma-algebra row, and its* $\mathsf{Real}^{\uparrow}$ *endpoint is classified with the additive zero* 0_R *of the target* $\mathsf{AbGroup}^{\uparrow}$ *real row.*

Proof. Project the empty-event and endpoint fields of Definition 92.214. They place the empty measurable event, its measure endpoint, and the additive 0_R row in the same visible BHist ledger.

Apply Definition 92.195. It keeps the empty-zero row inside the public additive face of the $\mathsf{Measure}^{\uparrow}$ root basis.

Next apply Definition 92.25. The sigma-additivity row applied to the all-empty sequence classifies $\mu(\varnothing)$ with the additive zero in the target $\mathsf{Real}^{\uparrow}$ row.

The finite-support empty-sequence zero row is recorded by Definition 92.32. It keeps the countable endpoint of the all-empty $\mathsf{StreamName}^{\uparrow}$ family classified with 0_R, so the public empty-event row and its endpoint classification are visible without adding a completion object or standard bridge. □

Theorem 92.217 (Measure root countable disjoint-additivity public row)**.** *For every covered pairwise-disjoint* $\mathsf{StreamName}^{\uparrow}$*-indexed family of measurable-event rows in the basis of Definition 92.214, there is a covered union row whose endpoint is classified with the countable* $\mathsf{Real}^{\uparrow}$ *sum of the endpoints of the family.*

Proof. Begin with a covered pairwise-disjoint $\mathsf{StreamName}^{\uparrow}$-indexed family. By Definition 92.214, the family rows, their disjointness witnesses, their endpoint rows, and the countable-sum witness are all ledger-attached public rows.

Apply Definition 92.164. It supplies the covered countable-union row and compares the union endpoint with the countable endpoint sum of the family.

Next apply Definition 92.195. This reads the same comparison through the public additive face, where the empty-zero, finite readback, relative-difference, countable disjoint-additivity, and countable-sum transport rows share the additive $\mathsf{AbGroup}^{\uparrow}$ real structure.

Finally apply Definition 92.12. The countable-sum classifier stability row transports pointwise $\mathsf{Real}^{\uparrow}$ endpoint comparisons to the countable sum, so the covered union endpoint is classified with the countable $\mathsf{Real}^{\uparrow}$ sum of the displayed endpoints and nothing outside the public basis is claimed. □

92.11 Measure root real-stream event obligations

The real-stream event obligation surface pins the public $\mathsf{Measure}^{\uparrow}$ root to the displayed BHist event rows, $\mathsf{StreamName}^{\uparrow}$ observation names, and $\mathsf{Real}^{\uparrow}$ endpoint comparisons that downstream analysis chapters may read. It is a root obligation surface, not an analytic completion or a host measure-space interface.

Theorem 92.218 (Measure root real-stream event obligation)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root event row exposed by Definition 92.193 is a* BHist*-displayed event carrier whose observable values are exactly the* $\mathsf{StreamName}^{\uparrow}$*-indexed* $\mathsf{Real}^{\uparrow}$ *endpoint observations supplied by Definition 92.166. Thus every carried root event is read as a displayed event history, a stream observation name, and the attached real endpoint comparison on the same ledger row.*

Proof. Start with the carrier and ledger clauses of Definition 92.193. They identify the root measure carrier with $\mathsf{MeasCode}$ rows on BHist and attach hidden measurable-event witnesses only to displayed public rows.

Apply Definition 92.194. It transports the carried $\mathsf{Measure}^{\uparrow}$ code, visible event rows, endpoint rows, countable-family rows, and ledger witnesses over the same public carrier.

Now read the same row through Definition 92.166 and Definition 92.167. The real-stream front adds exactly the StreamName endpoint-family observation and the $\mathsf{Real}^{\uparrow}$ endpoint comparison attached to the displayed event history.

The other public faces remain attached to this same row. By Definition 92.195 and Definition 92.168, countable sums are read only through the same displayed event family and endpoint stream. By Definition 92.196 and Definition 92.169, the analysis consumers see only projections of these displayed rows.

Repack the two readings as a single root obligation row. The public basis supplies the displayed BHist event carrier, while the real-stream front supplies only the observation name and endpoint comparison already attached to that carrier. No host event equality, quotient event equality, or analytic completion object is introduced. □

Theorem 92.219 (Measure root real-stream sibling route exactness)**.** *Every* $\mathsf{Measure}^{\uparrow}$ *root event row covered by Definition 92.218 reads its stream observation through chapter 126 and its endpoint comparison through chapter 38. The route retains only the displayed* BHist *event row, componentwise* hsame *transport,* Cont *continuation row,* $\mathsf{Pkg}_{\mathsf{Measure}^{\uparrow}}$ *package ledger, and the visible* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint row already present in Definition 92.166.*

Proof. Apply Definition 92.218. It identifies the root event row as a displayed BHist history carrying a $\mathsf{StreamName}^{\uparrow}$-indexed endpoint observation and the attached $\mathsf{Real}^{\uparrow}$ endpoint comparison on the same ledger row.

Now read the observation-name component through chapter 126 and the endpoint-comparison component through chapter 38. The regularity front Definition 92.166 is exactly the interface that

places those two sibling readings beside the root BHist row, the componentwise hsame transport, the Cont continuation discipline, the $\mathsf{Pkg}_{\mathsf{Measure}^{\uparrow}}$ package row, and the visible $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ ledger entry.

Nothing else is added to the route. Host event equality, quotient event equality, ambient completion, and selected host limits are outside the displayed root-event row, so they cannot become part of the sibling route described here. □

Theorem 92.220 (Measure root event classifier stability)**.** *For the root event surface of Definition 92.218, componentwise* hsame *transport of displayed event histories together with* $\mathsf{StreamName}^{\uparrow}$ *observation transport and* $\mathsf{Real}^{\uparrow}$ *endpoint classifier transport preserves the carried* $\mathsf{Measure}^{\uparrow}$ *root row and its visible ledger.*

Proof. Begin with two displayed event histories in the public root ledger and a componentwise hsame comparison between them. The carrier theorem Definition 92.194 transports the carried $\mathsf{Measure}^{\uparrow}$ code, visible event rows, endpoint rows, and ledger witnesses along the public classifier.

Next use the classifier clauses of Definition 92.166. They require pointwise StreamName classifier transport and $\mathsf{Real}^{\uparrow}$ endpoint comparison for values and countable sums over the same displayed history row.

Apply Definition 92.167. It aligns the transported event history, the transported stream observation, and the transported real endpoint on one ledger row. Then apply Definition 92.168 to keep the countable-sum endpoint classifier compatible with the same transport.

The additive and consumer projections do not widen the classifier. By Definition 92.195, the additive row remains a visible countable-family row in the public ledger. By Definition 92.196 and Definition 92.169, the consumer rows read the transported endpoint data without adding a new event carrier.

The resulting row is still one of the rows counted by Definition 92.193. Therefore the carried $\mathsf{Measure}^{\uparrow}$ root row, its event display, and its endpoint observation remain stable under the componentwise hsame and $\mathsf{Real}^{\uparrow}$ classifier motion. □

Theorem 92.221 (Measure root countable additivity scope)**.** *The countable disjoint-additivity obligation of the* $\mathsf{Measure}^{\uparrow}$ *root is scoped exactly to displayed* BHist *event families, their* $\mathsf{StreamName}^{\uparrow}$ *observation names, their* $\mathsf{Real}^{\uparrow}$ *endpoint sums, and the visible ledger rows of Definition 92.193. It does not assert additivity for an external sigma-algebra or for quotient event equality.*

Proof. Project the additive face from Definition 92.193. It lists empty-zero, finite readback, relative-difference additivity, countable disjoint-additivity, and countable-sum transport through the additive $\mathsf{AbGroup}^{\uparrow}$ real row.

The carrier and observation rows are fixed before additivity is read. Apply Definition 92.194; then read the same carrier through Definition 92.166 and Definition 92.167. This supplies the displayed BHist family, the StreamName observation names, and the attached $\mathsf{Real}^{\uparrow}$ endpoint comparisons.

Apply Definition 92.195. The theorem keeps the additive obligation attached to covered event rows, disjointness witnesses, countable-union rows, relative-difference rows, and countable-sum endpoint witnesses in the public ledger.

Next apply Definition 92.168. It reads the same countable family through the StreamName-indexed endpoint family and identifies the countable disjoint-additivity conclusion with a $\mathsf{Real}^{\uparrow}$ endpoint comparison for the countable sum.

Combining the two projections gives the stated scope: displayed BHist event families, StreamName observation names, real endpoint sums, and the visible ledger. Since both source surfaces

exclude host sigma-algebra equality and quotient event equality, the countable additivity obligation is confined to the root public surface. The consumer boundary is then only a projection of this scope by Definition 92.196 and Definition 92.169. □

Theorem 92.222 (Measure root analysis consumer surface)**.** *The downstream* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers of the* $\mathsf{Measure}^{\uparrow}$ *root consume exactly the displayed event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, countable-sum rows, finite-readback rows, normalization rows, monotone rows, and relative-difference rows carried by Definition 92.193 and Definition 92.166.*

Proof. Start with the consumer clause of Definition 92.193. It has exactly three outgoing faces: $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$.

The carrier and additive prerequisites are already fixed by Definition 92.194 and Definition 92.195. Reading them through Definition 92.166, Definition 92.167, and Definition 92.168 gives the displayed event rows, StreamName observations, endpoint comparisons, and countable-sum rows consumed below.

Apply Definition 92.196. The consumer face exposes covered event rows, endpoint classifier rows, countable-sum rows, empty-zero, finite readback, relative-difference, monotone inclusion, nonnegative endpoint, and total-event normalization rows for the three named consumers.

Now apply Definition 92.169. The real-stream front restricts the same consumers to the rows visible through StreamName observations and $\mathsf{Real}^{\uparrow}$ endpoint comparisons over the displayed BHist ledger.

Finally combine the exactness clauses of the public basis and the real-stream front. They leave no fourth analysis consumer, no external event carrier, and no analytic completion row. Hence $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consume exactly the listed root event, endpoint, countable-sum, finite-readback, normalization, monotone, and relative-difference rows. □

92.12 Measure root countable relative-difference bridge

Definition 92.223 (MeasureUp root countable relative-difference bridge)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root countable relative-difference bridge is the ten-field paper target that binds StreamName-indexed countable event families, displayed relative differences, and downstream analysis consumers on one root surface:*

1. *traditional target: a countable measurable family whose visible rows include displayed relative differences and real endpoint comparisons;*
2. *BEDC source:* $\mathsf{Measure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, *and the history classifier* hsame*;*
3. *carrier: StreamName-indexed measurable-event rows together with pairwise relative-difference witnesses for covered inclusions of displayed event histories;*
4. *classifier: transported event-row classification along* hsame*, pointwise StreamName observation classification, and* $\mathsf{Real}^{\uparrow}$ *endpoint classification for values and sums;*
5. *stability rows: carrier transport, countable-additivity transport, relative-difference transport, and endpoint transport through the same visible ledger;*
6. *ledger: indices, displayed events, complements and differences, pairwise-disjoint witnesses, countable unions, countable sums, and* $\mathsf{Real}^{\uparrow}$ *endpoints attached to the event rows they certify;*
7. *core theorem list: Definition 92.224, Definition 92.225, and Definition 92.226;*

8. *exactness: coverage, soundness, and completeness are exactly those of the displayed countable-family, relative-difference, endpoint, and consumer rows named by the bridge;*
9. *standard bridge: none;*
10. *status: paper target for* BEDC.Derived.MeasureUp.

The bridge is a root obligation package. It does not assert host sigma-algebra equality, quotient event equality, analytic completion, or a StandardBridge *row.*

Theorem 92.224 (Measure root countable relative-difference carrier)**.** *The bridge of Definition 92.223 exposes the source event row, the countable StreamName-indexed family row, the relative-difference row, and the attached* $\mathsf{Real}^{\uparrow}$ *endpoint histories on the same* BHist *ledger.*

Proof. Read the BEDC source, carrier, and ledger fields of Definition 92.223. They name the carried $\mathsf{Measure}^{\uparrow}$ root row, the $\mathsf{StreamName}^{\uparrow}$ countable family, the displayed relative-difference witnesses, and the endpoint histories before any consumer projection is formed.

Apply Definition 92.158. Its carrier and ledger clauses expose the carried measure code, StreamName endpoint family, event membership, pairwise disjointness, countable union, endpoint value, and countable-sum witnesses on visible BHist rows.

Next apply Definition 92.218 and Definition 92.220. These rows identify the same event histories with StreamName observation names and $\mathsf{Real}^{\uparrow}$ endpoint comparisons, and they keep the event classifier stable under componentwise hsame transport.

Finally apply Definition 92.101. The relative-difference theorem supplies the displayed difference event, its measurability row, and the disjoint decomposition row for each covered inclusion. Repacking these projections gives the stated carrier surface and adds no external event equality or completion principle. □

Theorem 92.225 (Measure root countable relative-difference additivity)**.** *For every covered StreamName-indexed countable event family in the bridge of Definition 92.223, endpoint comparison is obtained by combining the root countable-additivity row with the finite relative-difference decomposition row and* $\mathsf{Real}^{\uparrow}$ *classifier transport.*

Proof. Start with a covered StreamName-indexed family from the bridge ledger. The family rows carry event membership, pairwise disjointness, a visible countable union, endpoint values, and a countable-sum witness on the same BHist surface.

Apply Definition 92.221. It confines the countable disjoint-additivity obligation to displayed event families, StreamName observation names, $\mathsf{Real}^{\uparrow}$ endpoint sums, and visible ledger rows, so the union endpoint is compared with the countable sum inside the root surface.

For each covered inclusion used by the family, apply Definition 92.102. The finite relative-difference row compares the larger endpoint with the sum of the smaller endpoint and the displayed difference endpoint, using only the relative-difference decomposition supplied on the event ledger.

Apply Definition 92.217. Its public row transports the countable endpoint comparison through the $\mathsf{Real}^{\uparrow}$ classifier and keeps the countable sum attached to the displayed family endpoints.

The two comparisons share the same endpoint classifier field of the bridge. Thus the endpoint comparison visible to consumers is precisely the combination of countable additivity, finite relative-difference additivity, and $\mathsf{Real}^{\uparrow}$ classifier transport, with no host sigma-algebra equality or completion principle introduced. □

Theorem 92.226 (Measure root countable relative-difference consumer unblock)**.** *The* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers may read exactly the bridge rows exposed by Definition 92.223: displayed event rows, StreamName-indexed countable-family rows, finite readback rows, relative-difference rows, endpoint rows, countable-sum rows, normalization rows where named, monotone rows where named, and no host set equality or completion principle.*

Proof. Project the consumer-facing ledger from Definition 92.223. The exactness field restricts the outgoing rows to the displayed event, countable-family, relative-difference, endpoint, and countable-sum rows already carried by the root bridge.

For $\mathsf{Integral}^{\uparrow}$, apply Definition 92.87. The integral consumer reads covered measurable-event rows, $\mathsf{Real}^{\uparrow}$ endpoint classifier rows, countable-sum endpoint rows, head-tail readback rows, and finite disjoint-union readback rows from the same public $\mathsf{Measure}^{\uparrow}$ package.

For $\mathsf{ProbSpace}^{\uparrow}$, apply Definition 92.88. The probability consumer reads the total-event normalization row together with empty-zero and finite disjoint-union rows; it does not receive the countable-family rows as an independent analytic completion.

For $\mathsf{FunctionalAnalysis}^{\uparrow}$, apply Definition 92.89. The functional analysis consumer reads only the nonnegative endpoint, monotone inclusion, relative-difference, and countable-sum classifier rows exposed by $\mathsf{Measure}^{\uparrow}$.

Combining the three projections gives the downstream threshold carried by the bridge. Each consumer reads exactly its named rows from the shared countable relative-difference ledger, and none obtains host set equality, quotient event equality, analytic completion, or a $\mathsf{StandardBridge}$ row. □

92.13 Measure root real-stream premeasure bridge

The real-stream premeasure bridge is the concrete root surface that lets downstream analysis chapters read countable premeasure data through the displayed BHist event rows, $\mathsf{StreamName}^{\uparrow}$ observation names, and $\mathsf{Real}^{\uparrow}$ endpoint comparisons already exposed by the $\mathsf{Measure}^{\uparrow}$ root. It is a bridge between existing root rows, not a host measure-space construction.

Definition 92.227 (Measure root real-stream premeasure surface)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root real-stream premeasure surface is the ten-field surface with the following data:*

1. *traditional target: a countable premeasure surface whose measurable events and countable disjoint sums are visible before any analytic completion is admitted;*
2. *BEDC source:* BHist*,* hsame*,* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*,* $\mathsf{StreamName}^{\uparrow}$*, and the existing* $\mathsf{Measure}^{\uparrow}$ *root rows of Definition 92.193;*
3. *carrier: displayed measurable-event* BHist *rows, the* $\mathsf{StreamName}^{\uparrow}$*-indexed observation family on those rows, and the* $\mathsf{Real}^{\uparrow}$ *endpoint value assigned to each displayed event;*
4. *classifier: componentwise* hsame *for event histories, StreamName classifier transport for observation names, and* $\mathsf{Real}^{\uparrow}$ *endpoint classifier transport for event values and countable sums;*
5. *stability rows: carrier transport, classifier transport, countable disjoint additivity, and ledger exactness over the displayed root rows;*
6. *ledger: measurable witnesses, disjointness witnesses, countable-family indices, countable-union witnesses, and countable-sum endpoints attached to the same displayed* BHist *rows;*

7. *exactness coverage: every covered row of the surface is one of the displayed rows named by the public root basis, the countable-premeasure front, and the real-stream event obligation;*
8. *exactness soundness: every accepted carrier, classifier, additivity, and ledger row has the corresponding visible witness in the same* BHist *ledger;*
9. *exactness completeness: every displayed row with its StreamName observation,* $\mathsf{Real}^{\uparrow}$ *endpoint, disjointness data, union witness, and countable-sum endpoint is accepted by the surface;*
10. *standard bridge boundary: no host set equality, quotient event equality, external sigma-algebra equality, analytic completion object, or* StandardBridge *row is part of this surface.*

Theorem 92.228 (Measure root real-stream premeasure carrier obligation)**.** *The surface of Definition 92.227 exposes exactly the* BHist *event carrier,* $\mathsf{StreamName}^{\uparrow}$ *observation family,* $\mathsf{Real}^{\uparrow}$ *endpoint map, and visible ledger rows needed by the* $\mathsf{Measure}^{\uparrow}$ NameCert.

Proof. Start with the public root basis of Definition 92.193. Its carrier field gives the displayed BHist event rows, while its ledger field attaches measurable witnesses, endpoint witnesses, disjointness witnesses, and countable-sum witnesses only to displayed rows.

Read the same event row through Definition 92.166. The real-stream front adds the $\mathsf{StreamName}^{\uparrow}$ observation family and the attached $\mathsf{Real}^{\uparrow}$ endpoint comparison without changing the underlying event carrier.

Apply Definition 92.218. It identifies each carried root event with a displayed event history, a StreamName observation name, and the real endpoint comparison on the same ledger row.

The carrier, observation, endpoint, and ledger clauses of Definition 92.227 are precisely this combined reading. Since the boundary clause excludes host set equality and quotient event equality, the exposed carrier is exactly the visible root carrier required by the $\mathsf{Measure}^{\uparrow}$ NameCert. □

Theorem 92.229 (Measure root real-stream premeasure classifier obligation)**.** *For the surface of Definition 92.227, componentwise* hsame*, StreamName classifier transport, and* $\mathsf{Real}^{\uparrow}$ *endpoint classifier transport preserve the carried premeasure row and its ledger.*

Proof. Begin with a displayed premeasure row and a componentwise hsame comparison of its BHist event histories. The public root basis transports the carried event code and its visible ledger witnesses along the same root classifier row.

Apply Definition 92.220. It keeps the carried $\mathsf{Measure}^{\uparrow}$ root row stable under componentwise hsame together with StreamName observation transport and $\mathsf{Real}^{\uparrow}$ endpoint transport.

The classifier field of Definition 92.227 adds no independent classifier. It records exactly the event-history transport, the StreamName transport, and the endpoint transport already aligned by the event classifier stability theorem.

The ledger is preserved because its measurable witnesses, disjointness witnesses, countable-family indices, union witnesses, and countable-sum endpoints remain attached to the transported displayed row. Thus the premeasure row and its ledger are invariant under the stated classifier motion. □

Theorem 92.230 (Measure root real-stream premeasure sigma-additivity obligation)**.** *The countable disjoint-additivity row of Definition 92.227 is read by combining the existing root public obligation basis, the countable-premeasure front, the real-stream event obligation, and countable-sum classifier stability over the same displayed* BHist *ledger.*

Proof. Project a countable disjoint family from the ledger of Definition 92.227. Its component events, disjointness witnesses, countable-family index, union witness, and countable-sum endpoint are all displayed rows over the same BHist carrier.

Apply Definition 92.193 and Definition 92.221. These keep the countable additivity obligation scoped to visible event families, their union row, and the public ledger, with no external sigma-algebra row.

Next apply Definition 92.162 and Definition 92.164. They read the same visible family as a countable premeasure row whose union value is compared with the countable sum of its component $\mathsf{Real}^{\uparrow}$ endpoints.

Finally apply Definition 92.218 to read the components through StreamName observations, and apply Definition 92.220 together with Definition 92.12 to keep endpoint and countable-sum classifier transport aligned with the displayed event histories.

Combining these rows gives the sigma-additivity obligation of the surface: the value of the displayed union is the countable $\mathsf{Real}^{\uparrow}$ endpoint sum of the displayed disjoint component events, with every witness remaining inside the visible ledger. The conclusion does not assert host-level measure theory, quotient event equality, or analytic completion. □

92.14 Measure BHist root obligation basis

The $\mathsf{Measure}^{\uparrow}$ root obligation basis is read directly on displayed BHist histories. It records the BEDC carrier, classifier, endpoint, and public-ledger rows consumed by downstream analysis chapters before any host event space, analytic completion, or standard bridge is introduced.

Definition 92.231 (Measure BHist root obligation basis)**.** *The* $\mathsf{Measure}^{\uparrow}$ BHist *root obligation basis is the counted public surface with the following fields:*

1. *BEDC source:* BHist *histories with componentwise* hsame *as the only history transport relation;*
2. *carrier rows: visible* MeasCode *event rows displayed in the* $\mathsf{Measure}^{\uparrow}$ *public ledger;*
3. *StreamName rows:* $\mathsf{StreamName}^{\uparrow}$ *observation names for the countable event families consumed by the root;*
4. *endpoint rows:* $\mathsf{Real}^{\uparrow}$ *value endpoints, nonnegative endpoints, normalization endpoints, and countable-sum endpoints, all read through the public* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *certificate;*
5. *classifier rows: the public* $\mathsf{Measure}^{\uparrow}$ *classifier* $\sim_M$ *factored through componentwise* hsame*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *transport, and* $\mathsf{Real}^{\uparrow}$ *endpoint classifier transport;*
6. *ledger rows: event witnesses, countable-family witnesses, disjointness witnesses, countable-union witnesses, endpoint witnesses, and countable-sum witnesses attached to displayed rows;*
7. *consumer rows: the* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *projections that read only the displayed* BHist *ledger;*
8. *exclusions: no host set equality, quotient event equality, analytic completion object, or standard bridge row is exported.*

This basis is the common projection of the public root basis Definition 92.193, the real-stream event surface section 92.11, and the root-kernel source Definition 92.201.

Theorem 92.232 (Measure BHist root carrier obligation)**.** *Every accepted* $\mathsf{Measure}^{\uparrow}$ *event row in Definition 92.231 is a displayed* BHist *event carrier equipped with its* $\mathsf{StreamName}^{\uparrow}$ *observation row and its* $\mathsf{Real}^{\uparrow}$ *endpoint witnesses. No accepted row is read through an external event carrier.*

Proof. Begin with the BEDC source, carrier rows, StreamName rows, and endpoint rows of Definition 92.231. These fields place the event carrier in BHist and attach the observation name and real endpoint data to the same public ledger row.

Apply Definition 92.194. It supplies the public MeasCode carrier, visible event rows, endpoint rows, countable-family rows, and attached ledger witnesses.

Next apply Definition 92.218. The real-stream event surface identifies the accepted event row as a displayed BHist event history with a $\mathsf{StreamName}^{\uparrow}$ observation and a $\mathsf{Real}^{\uparrow}$ endpoint comparison on that same row.

Finally apply Definition 92.202. The kernel source confirms that componentwise hsame and the public $\mathsf{Measure}^{\uparrow}$ classifier keep the carrier, event, endpoint, countable-family, and ledger rows attached to the displayed history. Repacking these three projections gives exactly the stated carrier obligation. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureBHistRootCarrier obligation`

Theorem 92.233 (Measure BHist root classifier obligation)**.** *Classifier transport for Definition 92.231 consists only of componentwise* hsame *transport of displayed histories, pointwise* $\mathsf{StreamName}^{\uparrow}$ *transport of observation names, and* $\mathsf{Real}^{\uparrow}$ *endpoint classifier transport. It does not add host set equality or quotient event equality.*

Proof. Project the classifier rows of Definition 92.231. They factor the public $\mathsf{Measure}^{\uparrow}$ classifier through componentwise hsame, StreamName observation movement, and $\mathsf{Real}^{\uparrow}$ endpoint comparison.

Apply Definition 92.220. It preserves the carried root event row and the visible ledger under componentwise hsame, pointwise $\mathsf{StreamName}^{\uparrow}$ transport, and $\mathsf{Real}^{\uparrow}$ endpoint classifier transport.

Now apply Definition 92.202. The root-kernel source keeps the same transported event, endpoint, countable-family, and ledger rows attached to the BHist carrier.

The exclusions field of Definition 92.231 has no host set equality field and no quotient event equality field. Therefore the classifier obligation is exactly the componentwise history transport, observation-name transport, and endpoint classifier transport described above. □

Theorem 92.234 (Measure BHist root ledger obligation)**.** *The exported ledger of Definition 92.231 contains event witnesses, countable-family witnesses, disjointness rows, countable-union rows, endpoint witnesses, and countable-sum rows for the displayed* BHist *root. It excludes host set equality, quotient event equality, analytic completion, and standard bridges.*

Proof. Read the ledger rows and consumer rows of Definition 92.231. They attach every downstream witness to displayed event, countable-family, disjointness, union, endpoint, and countable-sum rows in the public NameCert ledger.

Apply Definition 92.195. This supplies empty-zero, finite readback, relative-difference, countable disjoint-additivity, and countable-sum transport over the same visible ledger.

Next apply Definition 92.221. It scopes the countable disjoint-additivity obligation to displayed BHist event families, $\mathsf{StreamName}^{\uparrow}$ observation names, $\mathsf{Real}^{\uparrow}$ endpoint sums, and the public ledger.

Finally apply Definition 92.204. The kernel source exports only the $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer boundary over those rows.

The exclusions field of the basis and the cited kernel source contain no host set equality, no quotient event equality, no analytic completion object, and no standard bridge row. Repacking the ledger projections therefore gives exactly the exported ledger stated in the theorem. □

Definition 92.235 (Measure root BHist NameCert input packet)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root* BHist $\mathsf{NameCert}$ *input packet is the ten-field public packet whose fields are:*

1. *the source carrier* BHist*;*
2. *the componentwise history classifier* hsame*;*
3. *the continuation row* Cont *used to preserve displayed root histories;*
4. *the* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint certificate;*
5. *the additive* $\mathsf{AbGroup}^{\uparrow}$ *real row for endpoint sums and differences;*
6. *the* $\mathsf{StreamName}^{\uparrow}$ *countable-family name rows;*
7. *the displayed measurable-event carrier and classifier rows of the root* $\mathsf{Measure}^{\uparrow}$ *ledger;*
8. *the empty-zero, finite disjoint-union, countable-disjoint-additivity, relative-difference, and continuity-from-below rows read from the same ledger;*
9. *the consumer boundary rows for* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*;*
10. *the exclusion row stating that no quotient event equality, host set equality, standard bridge row, or open-cover compactness row is part of the packet.*

Thus the packet is a single BHist*-facing input surface for the root measure certificate and its three displayed downstream consumers.*

Theorem 92.236 (Measure root BHist input carrier-classifier)**.** *The packet of Definition 92.235 supplies exactly the carrier and classifier transport required by the root* $\mathsf{Measure}^{\uparrow}$ *domain: displayed* BHist *event carriers, componentwise* hsame *classifier movement,* Cont *preservation of root histories,* $\mathsf{StreamName}^{\uparrow}$ *countable-family transport, and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint transport. No downstream consumer row is needed to obtain this carrier-classifier face.*

Proof. Start with the source carrier, classifier, continuation, endpoint, additive, countable-family, and displayed ledger fields of Definition 92.235. These fields are all root-facing input rows, so the carrier-classifier face can be read before any consumer projection is used.

Apply Definition 92.232. It places every accepted event row on a displayed BHist carrier with its $\mathsf{StreamName}^{\uparrow}$ observation row and $\mathsf{Real}^{\uparrow}$ endpoint witnesses.

Apply Definition 92.233. This identifies the classifier transport as componentwise hsame movement of histories, pointwise $\mathsf{StreamName}^{\uparrow}$ transport, and $\mathsf{Real}^{\uparrow}$ endpoint classifier transport, with no host set equality or quotient event equality.

Finally apply Definition 92.119. The domain-ready package repacks those same rows with Cont, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, the additive $\mathsf{AbGroup}^{\uparrow}$ real row, and the $\mathsf{StreamName}^{\uparrow}$ countable-family names. Hence the packet supplies exactly the carrier-classifier transport stated above. □

Theorem 92.237 (Measure root BHist input countable additivity)**.** *The packet of Definition 92.235 supplies the empty-zero row, finite disjoint-union readback, countable-disjoint additivity, relative-difference additivity, and continuity-from-below row for the root* $\mathsf{Measure}^{\uparrow}$ *ledger. These rows are read on displayed* BHist *event families with* $\mathsf{StreamName}^{\uparrow}$ *names and* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint sums.*

Proof. Project the additivity field of Definition 92.235. It contains the empty event, finite disjoint unions, covered pairwise-disjoint countable families, relative differences, increasing chains, $\mathsf{Real}^{\uparrow}$ endpoint comparisons, additive real rows, and visible ledger witnesses.

Apply Definition 92.96. It gives the countable-disjoint comparison for covered measurable sequences, and it reads the empty-zero and finite disjoint-union rows from the same public threshold package.

Next apply Definition 92.120. This places sigma-additivity, empty-zero, finite disjoint-union readback, finite-support countable readback, relative-difference additivity, and continuity from below on the domain-ready root package.

Finally apply Definition 92.234. The ledger obligation pins these witnesses to displayed BHist event rows, countable-family rows, disjointness rows, countable-union rows, endpoint witnesses, and countable-sum rows. Repacking the three cited sources gives exactly the countable-additivity face of the packet. □

Theorem 92.238 (Measure root BHist input consumer exactness)**.** *The packet of Definition 92.235 is exactly the root* Measure$^{\uparrow}$ *surface consumed by* Integral$^{\uparrow}$, ProbSpace$^{\uparrow}$, *and* FunctionalAnalysis$^{\uparrow}$. *The consumers read the displayed carrier-classifier, endpoint, additivity, relative-difference, and continuity rows of the packet; they do not consume quotient event equality, host set equality, standard bridge rows, or open-cover compactness rows.*

Proof. Begin with the consumer boundary and exclusion fields of Definition 92.235. The boundary names only the Integral$^{\uparrow}$, ProbSpace$^{\uparrow}$, and FunctionalAnalysis$^{\uparrow}$ rows, while the exclusion row removes quotient event equality, host set equality, standard bridge data, and open-cover compactness data from the input surface.

Apply Definition 92.121. It reads the carrier-classifier and sigma-additive rows as exactly the rows consumed by the three downstream faces, with no quotient event equality, no host set equality, and no standard bridge row.

Apply Definition 92.97. This identifies the Integral$^{\uparrow}$ boundary, the ProbSpace$^{\uparrow}$ normalization boundary, and the FunctionalAnalysis$^{\uparrow}$ nonnegative, monotone, relative-difference, and countable-sum boundary as the shared root surface and nothing beyond it.

Finally apply Definition 92.150. Its deps-ready package exports only visible event rows, endpoint classifier rows, endpoint-family rows, continuation rows, and displayed ledger witnesses to the same consumers. Combining these projections proves that the packet is the exact consumer input surface and that the four excluded row classes are not consumed. □

92.15 Measure root finite disjoint event envelope

The finite disjoint event envelope is the kernel-facing Measure$^{\uparrow}$ package used when downstream analysis chapters need a single finite event family, its displayed union, its Real$^{\uparrow}$ value fold, and the ledger rows that make the finite support visible. It is a finite BHist event package, not a sigma-complete event space or a standard bridge.

Definition 92.239 (Measure root finite disjoint event envelope)**.** *The* Measure$^{\uparrow}$ *root finite disjoint event envelope is the ten-field package attached to a displayed finite family* $E_0, \ldots, E_{n-1}$ *of* BHist *event rows:*

1. *source: the root public* Measure$^{\uparrow}$ *basis of Definition 92.193, the root event boundary of section 92.8, and the finite-support consumer front of Definition 92.186;*
2. *carrier: each* E_i *is a carried* BHist *event row covered by the root* Measure$^{\uparrow}$ *event carrier;*
3. *classifier: event comparison is the public* Measure$^{\uparrow}$ *classifier projected to componentwise* hsame *on displayed event histories and to the* Real$^{\uparrow}$ *classifier on endpoint rows;*

4. *disjointness: the family is pairwise disjoint after any classifier transport admitted by the root event boundary;*
5. *finite union: the package displays a finite union row* $U_n = E_0 \sqcup \cdots \sqcup E_{n-1}$ *in the same* BHist *ledger;*
6. *value rows: each component has a* $\mathsf{Real}^{\uparrow}$ *value endpoint* $\mu(E_i)$ *and the union has a* $\mathsf{Real}^{\uparrow}$ *value endpoint* $\mu(U_n)$*;*
7. *fold row: the finite endpoint* $\mathsf{fold}_R(\mu(E_0), \ldots, \mu(E_{n-1}))$ *is built by the additive* $\mathsf{AbGroup}^{\uparrow}$ *row carried by* $\mathsf{Real}^{\uparrow}$*;*
8. *support ledger: the finite family is embedded as a countable family with a constant empty tail, together with the visible finite-support ledger row;*
9. *consumer boundary:* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *may read only the carrier, finite union, value fold, and support-ledger rows named here;*
10. *excluded surface: no sigma-completeness row, open-cover compactness row, quotient event equality, host set equality, analytic completion object, or* StandardBridge *row is part of the envelope.*

Theorem 92.240 (Measure root finite disjoint carrier classifier)**.** *The envelope of Definition 92.239 projects carried* $\mathsf{Measure}^{\uparrow}$ *event rows, classifier-stable pairwise disjointness, and* $\mathsf{Real}^{\uparrow}$ *carrier rows from the existing root carrier, classifier, and event boundary packages. Thus every component event, its transported disjointness relation, its endpoint value, and the displayed finite union remain rows of the same root* BHist *ledger.*

Proof. Start with the carrier and classifier fields of Definition 92.193. By Definition 92.194, each displayed event row in the finite family is a carried BHist row, and classifier transport keeps the attached endpoint and ledger witnesses on the transported row.

Next read the same finite family through Definition 92.190. The event boundary exposes the visible event rows, finite union rows, endpoint rows, and ledger rows without replacing the displayed event carrier by a host-level event object.

For disjointness, use Definition 92.191. Its classifier is componentwise hsame on displayed histories together with $\mathsf{Real}^{\uparrow}$ endpoint comparison. Pairwise disjointness in the envelope is therefore stable exactly under the same classifier motion.

Finally project the endpoint rows through Definition 92.187. The finite support spine and the empty tail are read from the root carrier ledger, so the component values, finite union value, and support ledger all remain visible $\mathsf{Measure}^{\uparrow}$ root rows. □

Theorem 92.241 (Measure root finite disjoint additivity)**.** *For every envelope of Definition 92.239, the displayed finite union value is classified by the finite* $\mathsf{Real}^{\uparrow}$ *fold of the component values:*

$$\mu(U_n) \sim_R \mathsf{fold}_R(\mu(E_0), \ldots, \mu(E_{n-1})).$$

The classification is read from the countable-additivity row, the finite-support empty-tail readback, and the finite disjoint-union additivity packages already exposed by the $\mathsf{Measure}^{\uparrow}$ *root.*

Proof. Embed the finite family into a countable family by appending the constant empty tail named in the support-ledger field of the envelope. The carrier part of Definition 92.240 keeps the finite spine, the empty tail, and the displayed union in the same BHist ledger.

Apply Definition 92.195. It supplies the countable disjoint-additivity row, the finite readback row, and the countable-sum classifier transport over the visible ledger.

The empty tail is read by Definition 92.188 and Definition 92.23. These rows identify the countable endpoint of the finite-support family with the finite $\mathsf{Real}^{\uparrow}$ fold of the displayed component endpoints.

It remains to compare the displayed finite union endpoint with that finite fold. Use Definition 92.26 along the finite union tree, with additive congruence and associativity from the $\mathsf{Real}^{\uparrow}$ $\mathsf{AbGroup}^{\uparrow}$ row. Composing these classifier comparisons gives the displayed readback for $\mu(U_n)$. □

Theorem 92.242 (Measure root finite disjoint consumer threshold)**.** *The downstream* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers of a Definition 92.239 may read exactly the event carrier, displayed finite union,* $\mathsf{Real}^{\uparrow}$ *value fold, and finite-support ledger rows of the envelope. They receive no sigma-completeness row, open-cover compactness row, standard bridge row, quotient event equality, host set equality, or analytic completion object from this package.*

Proof. Begin with the consumer boundary field of Definition 92.239. It names only the carrier, finite union, value fold, and finite-support ledger rows as outgoing analysis data.

For the shared carrier and endpoint rows, apply Definition 92.240. This pins every consumer-visible event and value to the root BHist ledger and the $\mathsf{Real}^{\uparrow}$ endpoint classifier.

For the finite value readback, apply Definition 92.241. The downstream consumers see the displayed union value only through the finite fold comparison supplied by the envelope.

Now project the three consumer faces through Definition 92.189, Definition 92.87, Definition 92.88, and Definition 92.89. These projections read finite-support rows from the same root package and do not add a fourth consumer or a broader analytic interface.

The excluded-surface field of the envelope contains no slot for sigma-completeness, open-cover compactness, quotient event equality, host set equality, analytic completion, or $\mathsf{StandardBridge}$ data. Therefore the three analysis consumers receive exactly the four named row families and no larger surface. □

92.16 Measure root event-value closure

The event-value closure package records the smallest root surface that downstream analysis chapters may cite when they need the $\mathsf{Measure}^{\uparrow}$ event carrier, its public value rows, and its countable-family endpoint rows together.

Definition 92.243 (Measure root event-value closure package)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root event-value closure package is the following three-field public root package.*

1. *event carrier and classifier field: the carried* BHist *event rows, the* $\mathsf{MeasCode}$ *carrier predicate, componentwise* hsame *event transport, and the public* $\sim_M$ *classifier read through* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint transport;*

2. *countable-family field: each* $\mathsf{StreamName}^{\uparrow}$*-indexed row names a visible family of carried* BHist *events, pairwise-disjoint witnesses, countable-union readback, and the attached* $\mathsf{Real}^{\uparrow}$ *countable-sum endpoint;*

3. *endpoint and consumer field: the value of each carried event and the value of each displayed countable union are compared only through* $\mathsf{Real}^{\uparrow}$ *endpoint rows, and the only downstream consumers are* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *reading those displayed rows.*

The package contains no host set equality, no quotient event equality, no analytic completion object, and no open-cover compactness row.

Theorem 92.244 (Measure root carrier-value event closure)**.** *The carrier and value face of Definition 92.243 is exactly the carrier, classifier, event-row, stream-observation, and* $\mathsf{Real}^{\uparrow}$ *endpoint face supplied by Definition 92.193, Definition 92.218, and Definition 92.220.*

Proof. Read the first and third fields of Definition 92.243. They name a carried BHist event row, the $\mathsf{MeasCode}$ carrier, the public $\sim_M$ classifier, StreamName observation data, and $\mathsf{Real}^{\uparrow}$ endpoint comparisons for the visible value rows.

Apply Definition 92.193. Its carrier, classifier, stability, and ledger fields supply the same carried event rows, endpoint witnesses, and public classifier transport, with every hidden witness attached to a displayed row.

Apply Definition 92.218. It reads the same root row as a displayed event history, a StreamName observation name, and the attached real endpoint comparison.

Finally apply Definition 92.220. The componentwise hsame event motion, StreamName observation motion, and $\mathsf{Real}^{\uparrow}$ endpoint motion preserve the carried row and its visible ledger. Thus the carrier-value face is precisely the displayed public face cited in the statement.

None of the three cited surfaces introduces host set equality, quotient event equality, analytic completion, or open-cover compactness. Therefore the closure package exports no such row on its carrier-value face. □

Theorem 92.245 (Measure root countable-additivity event-value closure)**.** *The countable-additivity face of Definition 92.243 is exactly the displayed* BHist *countable-family surface, StreamName observation surface, disjoint-union readback, and* $\mathsf{Real}^{\uparrow}$ *endpoint-sum comparison supplied by Definition 92.221 and Definition 92.207.*

Proof. Project the countable-family field of Definition 92.243. It has only StreamName-indexed carried event families, pairwise-disjoint witnesses, countable-union readback, and the displayed $\mathsf{Real}^{\uparrow}$ countable-sum endpoint.

Apply Definition 92.221. The theorem confines countable disjoint-additivity to displayed BHist event families, StreamName observation names, real endpoint sums, and visible ledger rows of the public basis.

Apply Definition 92.207. It supplies measurable-sequence coverage, disjoint-union readback, and $\mathsf{Real}^{\uparrow}$ endpoint comparison through the same public ledger and componentwise classifier motion.

The two source theorems account for every countable-family row named by the closure package. They do not add an external sigma-algebra equality, quotient event equality, analytic completion, or open-cover compactness principle, so the additive face remains exactly the event-value surface stated above. □

Theorem 92.246 (Measure root analysis-consumer event-value closure)**.** *The downstream* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumers of Definition 92.243 consume only the displayed event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, countable-sum rows, finite-readback rows, normalization rows, monotone rows, relative-difference rows, and visible ledger rows supplied by Definition 92.222 and Definition 92.208.*

Proof. Read the endpoint and consumer field of Definition 92.243. It names exactly the three outgoing consumers $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$, and it restricts their input to displayed event rows, endpoint rows, countable-family rows, and visible ledger rows.

Apply Definition 92.222. It identifies the rows consumed by the three analysis chapters as displayed event rows, endpoint rows, countable-sum rows, finite-readback rows, normalization rows, monotone rows, and relative-difference rows carried by the public root and real-stream fronts.

Apply Definition 92.208. It restricts the same consumer triad to carrier/classifier rows, countable-additivity rows, nonnegative endpoint rows, and ledger rows displayed by the root minimum triad.

Intersecting these two readings leaves exactly the event-value rows listed in the statement: no fourth analysis consumer, no host set equality, no quotient event equality, no analytic completion, and no open-cover compactness. Hence the downstream consumers can cite the closure package without reconstructing any larger root interface. □

Definition 92.247 (Measure root countable-classifier presentation)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root countable-classifier presentation is the public countable-premeasure face whose visible carrier consists of* BHist *event rows equipped with the* hsame *transport already exposed by Definition 92.162. Its countable-family input is read through* $\mathsf{StreamName}^{\uparrow}$ *rows, so a family is displayed only by its stream name, its covered component event rows, and its covered union row. Its value side is read through* $\mathsf{Real}^{\uparrow}$ *endpoint classifiers: every event row, finite sum row, relative-difference row, continuity-from-below endpoint, and countable-sum row carries only the endpoint comparison supplied by* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$*.*

The presentation ledger is the ledger of Definition 92.162 restricted to the rows needed by countable classification: carried event rows, measurable-witness transport, $\mathsf{StreamName}^{\uparrow}$ *component indices, disjointness rows, covered union rows, countable-sum endpoints, and* $\mathsf{Real}^{\uparrow}$ *classifier comparisons. No host set equality, quotient event equality, analytic completion object, or consumer-specific classifier is part of this presentation.*

Theorem 92.248 (Measure root countable-classifier carrier scope)**.** *The presentation of Definition 92.247 exposes only covered* BHist *event rows,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, carrier transport, and classifier transport. Every displayed row of the presentation is one of those four row classes, and each of those four classes is supplied by the countable-premeasure carrier obligation.*

Proof. Apply Definition 92.163. It fixes the carrier side of the countable-premeasure front as the covered BHist measurable-event rows and the $\mathsf{Real}^{\uparrow}$ endpoint classifiers attached to their values.

The definition of Definition 92.247 only renames that front as a countable-classifier surface. Its $\mathsf{StreamName}^{\uparrow}$ and countable-sum clauses are ledger clauses over the same displayed event rows, so they do not add a host row or an independent carrier.

Thus the carrier scope is exhausted by the covered event rows, endpoint rows, carrier transport, and classifier transport already supplied by the cited obligation. □

Theorem 92.249 (Measure root countable-classifier stability)**.** *The presentation of Definition 92.247 is stable under finite additivity, relative-difference transport, continuity from below, and countable-sum classifier comparison. These four stability clauses are exactly the clauses obtained by composing the countable-premeasure additivity obligation, relative-difference additivity, the continuity-from-below package, and countable-sum classifier stability.*

Proof. Start with Definition 92.164. It supplies the root additivity row for the covered countable family and includes the finite readback needed for finite disjoint unions.

For relative differences, apply Definition 92.102 to the same carried event rows and $\mathsf{Real}^{\uparrow}$ endpoints. This transports the value comparison through the displayed relative-difference row without adding a new classifier.

For increasing unions, use Definition 92.42. Its source rows are covered event rows and countable-family rows already present in the presentation ledger, and its endpoint output is a $\mathsf{Real}^{\uparrow}$ comparison.

Finally apply Definition 92.12. It turns pointwise $\mathsf{Real}^{\uparrow}$ endpoint comparisons along the $\mathsf{StreamName}^{\uparrow}$ family into the countable-sum endpoint comparison. The four cited clauses therefore give all stability rows named by the presentation. □

Theorem 92.250 (Measure root countable-classifier downstream interface)**.** *The outgoing rows from Definition 92.247 to* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *are exactly the carrier rows, classifier rows, and stability rows named in Definition 92.248 and Definition 92.249. No downstream consumer receives an additional host equality, quotient equality, or completion row from the root presentation.*

Proof. Apply Definition 92.165. It fixes the $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer faces as projections of the countable-premeasure front.

Next apply Definition 92.200. The bridge identifies the outgoing premeasure rows with the downstream threshold rows and keeps the consumer interface inside the same root ledger.

The carrier and classifier part of that ledger is exactly the scope proved in Definition 92.248. The additivity, relative-difference, continuity-from-below, and countable-sum part is exactly the stability package proved in Definition 92.249.

Hence each named downstream chapter reads precisely those carrier, classifier, and stability rows, and no further row leaves the root countable-classifier presentation. □

Definition 92.251 (Measure root NameCert front)**.** *The* $\mathsf{Measure}^{\uparrow}$ *root* $\mathsf{NameCert}$ *front is the ten-field public package that downstream consumers read before forming any analytic completion or consumer-local quotient:*

1. *event carrier field: visible* BHist *event rows and the* $\mathsf{MeasCode}$ *predicate certifying that each event row is carried by the root measure surface;*
2. *event classifier field: componentwise* hsame *transport on* BHist *event presentations, read externally as the public* $\sim_M$ *classifier;*
3. *NameCert carrier field: the root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *carrier row that packages the measurable-event witnesses without host set equality;*
4. *NameCert classifier field: the root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *classifier row identifying exactly the displayed event and endpoint classifier motion;*
5. *StreamName family field: countable families named through* $\mathsf{StreamName}^{\uparrow}$, *including displayed component rows, disjointness rows, and union rows;*
6. *Real endpoint field: value rows, finite-sum rows, relative-difference rows, continuity-from-below rows, and countable-sum rows compared only by* $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ *endpoint classifiers;*
7. *stability field: finite additivity, relative-difference transport, continuity from below, and countable-additivity stability over the same visible rows;*
8. *ledger field: the visible root ledger attaching every hidden witness to a displayed event row, endpoint row, countable-family row, or consumer row;*
9. *downstream consumer field: the three outgoing faces named* $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$*;*
10. *exclusion field: no host set equality, quotient event equality, analytic completion, or open-cover compactness row is part of the front.*

Theorem 92.252 (Measure root NameCert front carrier)**.** *The carrier face of Definition 92.251 is exactly the* BHist *event-row carrier,* $\mathsf{MeasCode}$ *predicate,* $\sim_M$ *classifier, root* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *carrier-classifier row, and countable-classifier carrier scope supplied by Definition 92.6, Definition 92.7, Definition 92.17, and Definition 92.248.*

Proof. Read the first four fields of Definition 92.251. They name the visible BHist event rows, the $\mathsf{MeasCode}$ carrier predicate, componentwise hsame transport, the public $\sim_M$ classifier, and the root $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ carrier-classifier row.

Apply Definition 92.6. Its inventory lists the carried event rows, measurable witnesses, classifier transport, endpoint rows, and ledger attachments that may appear on the root carrier surface. Apply Definition 92.7 to restrict the classifier to event-witness motion and $\mathsf{Real}^\uparrow$ endpoint comparison.

Next apply Definition 92.17. It identifies the root $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ carrier and classifier with exactly those displayed rows. Apply Definition 92.248 to include the countable-family carrier side without adding a second carrier.

The exclusions field of the front is therefore inherited from the cited carrier rows: none of them supplies host set equality, quotient event equality, analytic completion, or open-cover compactness. Hence the front carrier is precisely the carrier face stated above. □

Theorem 92.253 (Measure root NameCert front classifier)**.** *The classifier face of Definition 92.251 is stable under the root* $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ *exactness rows, the countable classifier presentation, and the countable-additivity event-value closure. Equivalently, all classifier motion in the front is obtained from Definition 92.18, Definition 92.249, and Definition 92.245.*

Proof. Project the event classifier, Real endpoint, stability, and ledger fields of Definition 92.251. They allow event motion only through componentwise hsame and endpoint motion only through $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ classifier rows.

Apply Definition 92.18. Its exactness clauses attach every root $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ stability witness to the displayed event, endpoint, and ledger rows of the root surface.

For countable families, apply Definition 92.249. It transports finite additivity, relative differences, continuity from below, and countable-sum comparison through the countable-classifier presentation.

Finally use Definition 92.245. That theorem identifies the countable-additivity event-value face with displayed BHist families, $\mathsf{StreamName}^\uparrow$ observation rows, disjoint-union readback, and $\mathsf{Real}^\uparrow$ endpoint-sum comparison. Thus the front classifier contains only stable visible rows and no host equality, quotient event equality, analytic completion, or open-cover compactness. □

Theorem 92.254 (Measure root NameCert front ledger)**.** *The downstream ledger of Definition 92.251 exports to* $\mathsf{Integral}^\uparrow$*,* $\mathsf{ProbSpace}^\uparrow$*, and* $\mathsf{FunctionalAnalysis}^\uparrow$ *exactly the carrier rows, classifier rows, stability rows, event-value rows, and minimum consumer obligations supplied by Definition 92.250, Definition 92.246, and Definition 92.208.*

Proof. Read the ledger and downstream consumer fields of Definition 92.251. They name a visible root ledger and exactly three consumers: $\mathsf{Integral}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, and $\mathsf{FunctionalAnalysis}^\uparrow$.

Apply Definition 92.250. It exports the carrier, classifier, and stability rows of the countable-classifier presentation to the three named downstream chapters and blocks any additional host equality, quotient equality, or completion row.

Next apply Definition 92.246. It identifies the event-value rows consumed by the same three analysis chapters: displayed event rows, endpoint rows, countable-sum rows, finite-readback rows, normalization rows, monotone rows, relative-difference rows, and visible ledger rows.

Finally apply Definition 92.208. It restricts the consumer package to carrier/classifier rows, countable-additivity rows, nonnegative endpoint rows, and root ledger rows. Intersecting these three projections gives the ledger exported by the front, with no host set equality, quotient event equality, analytic completion, or open-cover compactness. □

Theorem 92.255 (Measure root NameCert front obligation status)**.** *Definition 92.251, together with Definition 92.252, Definition 92.253, and Definition 92.254, enumerates the* $\mathsf{Measure}^{\uparrow}$ *root obligation set: the carrier and classifier rows, countable-family rows through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$ *endpoint rows, stability and exactness rows, ledger rows, and the* $\mathsf{Integral}^{\uparrow}/\mathsf{ProbSpace}^{\uparrow}/\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer faces. The same front excludes host set equality, quotient event equality, analytic completion, and standard bridges.*

Proof. Project the ten fields of Definition 92.251. The first four fields name the event carrier, event classifier, root $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ carrier, and root $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ classifier. By Definition 92.252, these are exactly the carrier-classifier rows of the root surface.

The fifth through seventh fields name the $\mathsf{StreamName}^{\uparrow}$ countable-family rows, the $\mathsf{Real}^{\uparrow}$ endpoint rows, and the stability rows. Apply Definition 92.253; it identifies this motion with the exactness rows, countable-classifier presentation, and countable-additivity event-value closure.

The eighth and ninth fields name the visible ledger and the three outgoing consumer faces. By Definition 92.254, these fields export exactly the ledger and consumer rows read by $\mathsf{Integral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{FunctionalAnalysis}^{\uparrow}$.

The tenth field is the exclusion field. It rules out host set equality, quotient event equality, analytic completion, and standard bridges, so the front enumerates precisely the obligation set stated above. □

Zero BHist Terminal Standard Bridge

Definition 92.256 (Zero BHist terminal standard measure reference)**.** *The terminal standard measure reference for the zero* $\mathsf{BHist}\ \mathsf{Measure}^{\uparrow}$ *surface is the one-row host-facing reference* $T^*_{\mu=0}$*. It has one visible event* $*_E$*, one visible value* 0_T*, and one accepted history representative. Every event classifier in* $T^*_{\mu=0}$ *is terminal:*

$$*_E \sim_T *_E.$$

The value map sends the unique event to 0_T*. Every displayed countable disjoint family in* $T^*_{\mu=0}$ *is read back to the same row: each member, its countable union, and the selected countable sum all have value* 0_T*.*

The bridge ledger for $T^*_{\mu=0}$ *is limited to the public zero* $\mathsf{BHist}\ \mathsf{Measure}^{\uparrow}$ *rows already displayed in Definition 92.48 and Definition 92.53: the carried history is the empty* BHist *row, the event inventory is the singleton empty-event row, all union and disjointness rows are the same singleton row, the value endpoint is* 0_R*, and the countable-sum endpoint is the constant* 0_R *row.*

Theorem 92.257 (Zero BHist terminal standard bridge)**.** *The public zero* $\mathsf{BHist}\ \mathsf{Measure}^{\uparrow}$ *NameCert surface of Definition 92.51 interprets in the terminal standard reference of Definition 92.256. Conversely, that terminal reference maps back to the same zero* BHist *carrier, classifier, zero event value, countable-additivity readback, exactness rows, and ledger rows displayed by the public* $\mathsf{Measure}^{\uparrow}$ *surface. The bridge uses only the classifier rows on those displayed carriers; it does not assert host set equality, quotient event equality, analytic completion, or sigma-complete measure theory.*

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureZeroBHist terminal standard bridge`

Proof. For the BEDC-to-standard direction, apply the zero BHist surface of Definition 92.48. The carrier row is $\mathsf{ZeroMeasureCode}(h) \iff \mathsf{hsame}(h, \mathsf{e})$, the event inventory is the singleton empty-event row, and the value endpoint is always 0_R. The carrier and classifier transport required to encode this surface in $T^*_{\mu=0}$ is exactly Definition 92.49.

The countable-additivity field is read from Definition 92.50. Since the terminal reference has one event row and one value row, a displayed countable disjoint family has only the constant empty-event presentation. Its countable union and selected countable sum therefore read back to the same zero row.

For the standard-to-BEDC direction, choose the empty BHist carrier, the singleton empty-event row, and the constant 0_R endpoint for the unique standard event and value. Apply Definition 92.54 to cover every measurable, empty, binary-union, countable-union, disjointness, value, and countable-sum ledger row by that singleton surface.

Exactness and NameCert preservation are the rows of Definition 92.56 and Definition 92.51. The former packages event coverage, countable-sum classifier stability, carrier stability, sigma-additivity, and the NameCert rows into the concrete root surface; the latter is the public semantic certificate named by the $\mathsf{Measure}^{\uparrow}$ export.

Thus coverage, soundness, completeness, and ledger preservation are projections of the displayed zero rows. The bridge has no place to import host sigma-algebras, quotient events, analytic completions, or host set equality: the terminal reference has only $*_E$ and 0_T, and every round-trip is the same classifier-mediated singleton row already present on the public $\mathsf{Measure}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.MeasureUp.MeasureUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$.

Constructive story: The concrete measure surface starts from the empty history: a measure code is accepted only when its visible BHist representative is hsame to e, its event inventory is the singleton empty-event row, and every measurable event, finite union, countable union, difference row, and countable-sum endpoint is displayed as that same zero package. Two accepted measure presentations are classified by the carried history comparison and by the $\mathsf{Real}^{\uparrow}$ classifier on their value endpoints, so event movement is componentwise hsame transport of the visible rows rather than host set equality or quotient event equality. The source specification is the zero BHist carrier together with $\mathsf{Real}^{\uparrow}$ zero endpoints and the additive $\mathsf{AbGroup}^{\uparrow}$ zero laws for head-tail countable sums, while the pattern specification records empty-event coverage, finite disjoint-additivity, relative and nested difference zero rows, continuity from below, and countable-name consumer rows. The ledger policy keeps the raw empty carrier, singleton event row, union and disjointness displays, zero value endpoint, countable-sum endpoint, additive zero readback, and downstream consumer projections visible. The five name-certificate obligations close because e inhabits the zero carrier, hsame gives reflexivity, symmetry, and transitivity, and any classifier transport of a zero code, empty event, or zero endpoint lands back in the same singleton rows. A history h lies in the signature gap of a $\mathsf{Measure}^{\uparrow}$ token p when the source-domain accepts h as this zero measure surface, the signature relation pairs it with an answer measure package and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h has only the empty event and zero value rows encoded by p. The terminal standard bridge sends the BEDC zero surface to the one-event, one-value standard reference and reads that reference back through the same empty carrier, zero endpoint, countable-additivity, exactness, and ledger rows. The chapter registers this zero-measure construction under the name $\mathsf{Measure}^{\uparrow}$; arbitrary host sigma-algebras, analytic completion, probability-space closure, quotient event equality, and Lean-checked standard bridge data are not introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope is exactly the zero-BHist public $\mathsf{Measure}^{\uparrow}$ surface, the terminal standard bridge, finite-additivity readback, nested-difference rows, the continuity-from-below row, countable-name and root consumer surfaces, and the analysis-consumer bridge rows already input by the $\mathsf{Measure}^{\uparrow}$ hub.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.MeasureUp.MeasureZeroBHist\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This block does not close arbitrary host sigma-algebras, quotient event equality, analytic completion, probability-space closure, or a Lean-checked standard bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

The $\mathsf{Measure}^{\uparrow}$ packet is read as a dependency lattice for analysis consumers: value rows are grounded in the real-name surface, integration rows consume the same measure values, and probability rows read normalization through the resulting finite event-value boundary. Its measurable-event surface consumes the displayed set-like event classifier rows of section 92.8 and the $\mathsf{Real}^{\uparrow}$

value row of chapter 38 before probability, random-variable, distribution, integral, and spectral-measure consumers read it. The event classifier also consumes the sigma-algebra sibling through chapter 5186 before probability and integration consumers read countable event closure. The generated-event row is anchored by the $\mathsf{BorelSet}^{\uparrow}$ surface of chapter 3384, while the cover-valued event boundary is routed through the $\mathsf{OuterMeasure}^{\uparrow}$ surface of chapter 3385.

Theorem 92.258 (Measure dependency lattice)**.** *Every accepted* $\mathsf{Measure}^{\uparrow}$ *carrier reads its value object through chapter 38, exports integration consumers through chapter 123, and supplies probability normalization consumers through chapter 184. These three reads are attached to the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, so no consumer can detach a measure value from its real, integral, and probability-facing dependency surface.*

Proof. The value row is read first. It is the $\mathsf{Real}^{\uparrow}$-facing object that fixes the codomain used by event-value comparisons.

The integration consumer is downstream of that value row. It receives the displayed measure values as the input surface for integral construction, not as an independent host-valued function.

The probability consumer is downstream of the same event-value boundary. Its normalization read is therefore attached to the measure value and integral-facing rows through the carrier's structural transport, replay, provenance, and local naming data.

Since the same finite structural rows bind all three consumers, the dependency lattice records one $\mathsf{Measure}^{\uparrow}$ packet rather than three unrelated analysis interfaces. □

Theorem 92.259 (Measure probability sigma Borel route)**.** *Every probability, random-variable, distribution, or integral consumer of an accepted* $\mathsf{Measure}^{\uparrow}$ *packet must read the sigma-algebra sibling, the* $\mathsf{BorelSet}^{\uparrow}$ *generated-event row, the* $\mathsf{Real}^{\uparrow}$ *value row, and the probability normalization row through the same finite event-value carrier before any countable-event closure or integration read is public. Consequently the consumer cannot detach measure values from the displayed event classifier, generated-event, and normalization rows.*

Proof. Project the accepted $\mathsf{Measure}^{\uparrow}$ dependency lattice of Definition 92.258. Its value object is already read through the $\mathsf{Real}^{\uparrow}$ row, while its probability and integration consumers are attached to the same finite BHist, hsame, Cont, Pkg, and NameCert rows.

Use the sibling routes named in the opening paragraph. The measurable-event classifier reads the sigma-algebra surface of chapter 5186, and the generated-event row is anchored by the $\mathsf{BorelSet}^{\uparrow}$ surface of chapter 3384. These rows enter before probability normalization, random-variable distribution reads, or integral formation can be exported.

The probability row then supplies normalization over that same event-value boundary. Since the sigma-algebra classifier, Borel generated-event row, Real value row, and normalization row are all carried by the same structural transport, replay, provenance, and local naming data, any downstream consumer receives one displayed event-value carrier rather than a detached host-valued measure function. □

Theorem 92.260 (Measure real-stream dependency lattice)**.** *Every* $\mathsf{Measure}^{\uparrow}$ *real-valued or countable-name consumer factors through the* $\mathsf{StreamName}^{\uparrow}$ *finite observation rows of Definition 92.77, the* $\mathsf{Real}^{\uparrow}$ *seal rows attached to those observations, the finite event-algebra rows of the root public surface, the displayed* BHist *ledger, componentwise* hsame*,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any countable premeasure or relative-difference read is accepted. The route is the common dependency lattice supplied by Definition 92.78, Definition 92.91, Definition 92.228, and Definition 92.258.*

Proof. Begin with the StreamName endpoint family. By Definition 92.78, each unary observation is carried by the displayed $\mathsf{StreamName}^{\uparrow}$ row and transports its measurable-event row and $\mathsf{Real}^{\uparrow}$ endpoint row through the same componentwise classifier data.

Next read the countable-name surface. Apply Definition 92.91; it requires the visible countable family, event rows, endpoint comparisons, and pointwise StreamName classifier rows to remain attached to one BHist measure code.

Now use the real-stream premeasure bridge. The carrier obligation of Definition 92.228 admits a countable premeasure row only after the displayed event rows, StreamName observation family, Real endpoint comparisons, and NameCert rows have been read.

Finally apply Definition 92.258. Its value, integral, and probability-facing dependencies bind the same finite event-value boundary to BHist, hsame, Cont, Pkg, and local naming data. Therefore a countable premeasure or relative-difference consumer cannot bypass the StreamName observations, Real seals, finite event-algebra rows, or structural certificate rows. □

Theorem 92.261 (Measure probability and integral dependency route)**.** *Every accepted* $\mathsf{Measure}^{\uparrow}$ *packet sends its finite event rows and* $\mathsf{Real}^{\uparrow}$*-valued mass rows to chapter 184 and chapter 123 only through displayed* $\mathsf{Measure}^{\uparrow}$ *rows. The* $\mathsf{ProbSpace}^{\uparrow}$ *consumer reads normalization and finite additivity from the event-value surface, while the* $\mathsf{Integral}^{\uparrow}$ *consumer reads premeasure and value rows from the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Proof. Start with the dependency lattice of Definition 92.258. It ties probability and integral consumers to the same real-valued measure rows rather than to detached host functions.

For the probability side, read the $\mathsf{ProbSpace}^{\uparrow}$ consumer rows through the displayed finite event boundary. Normalization and finite additivity are therefore projections of the $\mathsf{Measure}^{\uparrow}$ packet.

For the integral side, read the $\mathsf{Integral}^{\uparrow}$ premeasure route through the displayed value rows. Since both consumers share the same structural transport, replay, provenance, and local naming rows, neither receives a mass row outside the shown $\mathsf{Measure}^{\uparrow}$ carrier surface. □

The $\mathsf{Measure}^{\uparrow}$ event-value and countable-premise rows are consumed by chapter 184 and chapter 123. The route reads the finite event algebra, $\mathsf{Real}^{\uparrow}$ value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before either probability normalization or integral readback is available. The $\mathsf{Measure}^{\uparrow}$ hub supplies measurable-event rows and $\mathsf{Real}^{\uparrow}$-valued mass rows to chapter 123 and chapter 184 only after the value row of chapter 38 and the local $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ route have been displayed. The probability and integral consumers therefore read the same finite event-value packet, rather than a detached measure, an independent probability primitive, or an ambient real-valued selector.

Theorem 92.262 (Measure value route for probability and integral consumers)**.** *Every accepted* $\mathsf{Measure}^{\uparrow}$ *value row is read through the* $\mathsf{Real}^{\uparrow}$ *value surface of chapter 38 before probability normalization through chapter 184 or integration through chapter 123 can consume it. The probability and integral consumers therefore share the same finite event-value row,* BHist *ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *row; neither consumer receives a detached host-valued measure function.*

Proof. Project the accepted $\mathsf{Measure}^{\uparrow}$ carrier through the value rows named in Definition 92.258. The value object is first a $\mathsf{Real}^{\uparrow}$-facing row, so event comparisons and normalization reads use the same displayed real-name surface.

The probability consumer reads that row as its normalization input, while the integral consumer reads it as the value input for integration. Both reads remain downstream of the same finite event-value boundary.

The structural rows transport, replay, source, and name that shared boundary. Since no carrier coordinate supplies a second host-valued measure function, probability and integration cannot detach their value reads from the displayed $\mathsf{Measure}^{\uparrow}$ packet. □

Theorem 92.263 (Measure probability sigma dependency)**.** *Every accepted* $\mathsf{Measure}^{\uparrow}$ *packet exposes its event algebra, sigma-additive rows, finite disjoint-union readback, and* $\mathsf{Real}^{\uparrow}$*-valued mass rows before any integration, random-variable, distribution, or probability consumer receives a measure row. In particular, the* $\mathsf{ProbSpace}^{\uparrow}$ *consumer of chapter 184 reads its total-event normalization and probability-total-mass boundary through the same displayed* $\mathsf{Measure}^{\uparrow}$ *event family and value route used by the integral and random-variable surfaces, not through a detached host measure or an independent probability primitive.*

Proof. Start with the root event boundary and sigma-additive obligation rows of Definition 92.190 and Definition 92.184. They expose the measurable-event family, event classifier, empty-event row, finite disjoint-union readback, and countable additivity surface as carried $\mathsf{Measure}^{\uparrow}$ data.

Apply the probability substrate obligations of Definition 92.71. The $\mathsf{ProbSpace}^{\uparrow}$ row adds the total event and the unit endpoint only after those measure rows have already supplied the event algebra and mass values.

Thus every downstream probability, integration, random-variable, or distribution read sees the same finite event-and-value route. The Measure packet exports a shared dependency boundary, not separate host-side measure and probability objects. □

Theorem 92.264 (Measure ProbSpace total-mass handoff)**.** *Every accepted* $\mathsf{ProbSpace}^{\uparrow}$ *total-event normalization row consumed through* $\mathsf{Measure}^{\uparrow}$ *must expose the* $\mathsf{Measure}^{\uparrow}$ *event algebra, sigma-additive rows, finite disjoint-union readback, and* $\mathsf{Real}^{\uparrow}$*-valued mass rows before a probability consumer may read total mass* 1*. The handoff supplies a probability boundary over displayed measure data, not a detached probability primitive or host-side total-measure assertion.*

Proof. Start with Definition 92.263. It already routes probability consumers through the same event family and value route used by integration, random-variable, and distribution surfaces.

The root event and sigma-additive obligations, Definition 92.190 and Definition 92.184, keep measurable events, empty-event data, finite disjoint-union readback, and countable-additivity rows visible as $\mathsf{Measure}^{\uparrow}$ data.

Apply the probability substrate obligations Definition 92.71. The total-event normalization row and the unit endpoint are therefore read only after the event algebra and $\mathsf{Real}^{\uparrow}$ mass route have been displayed, so the probability consumer receives total mass 1 through the measure packet's finite dependency lattice. □

Theorem 92.265 (Measure probability sigma dependency forward link)**.** *Every* $\mathsf{ProbSpace}^{\uparrow}$ *or* $\mathsf{Integral}^{\uparrow}$ *consumer of an accepted* $\mathsf{Measure}^{\uparrow}$ *packet must pass through the displayed sigma-event source rows, finite additivity and readback rows, and* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *provenance before reading probability mass or integral value rows. The probability-facing route of chapter 184 and the integral-facing route of chapter 123 therefore share the same* $\mathsf{Measure}^{\uparrow}$ *dependency lattice rather than reading detached probability primitives, host measures, or independent value selectors.*

Proof. Start with the sigma-event and source rows already displayed by the $\mathsf{Measure}^{\uparrow}$ carrier. They name the measurable-event family, event classifier, empty-event boundary, and source value route before any probability or integration consumer appears.

Next read finite additivity and readback. These rows expose the disjoint-union mass ledger and the value route used by probability normalization and integral accumulation, so neither consumer can bypass the measure packet when it reads mass or integral data.

Finally pass through provenance and local naming. The Pkg and $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ rows seal the shared dependency lattice, and the sibling probability and integral chapters can consume only the sealed event-and-value surface. No detached host measure, independent probability primitive, or selected value representative is exported. □

Theorem 92.266 (Measure ProbSpace event-algebra forward route)**.** *Every probability-space mass-normalization or event-algebra consumer reading* $\mathsf{ProbSpace}^\uparrow$ *must first display the* $\mathsf{Measure}^\uparrow$ *finite event-value surface, finite additivity readback, sigma-algebra public basis, and real-stream premeasure bridge before the probability normalization row is read. The route factors through the ProbSpace certificate of chapter 184 only after the Measure sibling routes Definition 92.214 and section 92.13 have exposed their event-basis and premeasure rows.*

Thus a $\mathsf{ProbSpace}^\uparrow$ *consumer may normalize an event ledger only as a downstream read of the displayed* $\mathsf{Measure}^\uparrow$ *event-value and premeasure surface, not as a direct host probability object.*

Proof. Begin with the $\mathsf{Measure}^\uparrow$ finite event-value surface. Its carrier rows expose the finite event ledger and the value readback before any probability-space normalization is available.

Apply the finite additivity readback and then the sigma-algebra public basis of Definition 92.214. These rows identify which event algebra a downstream $\mathsf{ProbSpace}^\uparrow$ reader is allowed to inspect.

Finally use the real-stream premeasure bridge of section 92.13 and its carrier obligation Definition 92.228. It places the real-stream premeasure row between the event algebra and the probability normalization row of chapter 184, so the ProbSpace consumer inherits only the displayed Measure route. □

The probability-space normalization route is an additive $\mathsf{Measure}^\uparrow$ child surface. A $\mathsf{ProbSpace}^\uparrow$ consumer first reads the event algebra and total-mass row through chapter 184; only after that displayed $\mathsf{Measure}^\uparrow$ route is exposed may probability-specific random-variable, distribution, weak-law, or event-bound consumers read the normalized probability rows. The route supplies no independent probability carrier, no completed measure algebra, and no selected null-set quotient beyond the visible measure and total-mass data. The countable event and value rows of $\mathsf{Measure}^\uparrow$ reach Real-facing consumers only through the finite name route of chapter 126, chapter 301, and chapter 38. This route keeps each countable-event read tied to displayed StreamName windows, regular rational readback, Real seals, BHist rows, componentwise hsame, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ data; it does not make $\mathsf{Measure}^\uparrow$ an L10 completeness claim or add measure-theoretic completion beyond the named finite event/value surface. The $\mathsf{Measure}^\uparrow$ event-value and premeasure rows consume the L10 completion surface through the $\mathsf{Real}^\uparrow$ value route of chapter 38 and the finite observation discipline of chapter 126. The same displayed rows then feed the probability consumer of chapter 184: normalization may read a total-mass row only after the event algebra, Real-valued mass row, BHist ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ route are visible.

Chapter 93

A Concrete Naming Certificate for $\mathsf{Filter}^{\uparrow}$

The filter interface $\mathsf{Filter}^{\uparrow}$ packages a non-empty upward-closed family of subsets of a carrier closed under finite intersections. It supplies the structural substrate for convergence and limit notions independent of metric data, and is the constructive analogue of mathlib-style `Filter` for the BEDC topology and analysis layers.

93.1 Carrier definition

Theorem 93.1 (Principal suffix unary intersection closure)**.** *Let* $base, left, right, meet, leftPoint, rightPoint, mee$ BHist*. If* $base$*,* $left$*, and* $right$ *are unary histories,*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint), \qquad \mathsf{Cont}(base, meet, m$$

then $meetPoint$ *is unary, and it is simultaneously displayed as the continuation of* $leftPoint$ *by* $right$ *and of* $rightPoint$ *by* $left$*.*

Proof. Unary closure first gives unary $meet$ from the continuation $\mathsf{Cont}(left, right, meet)$, and then gives unary $meetPoint$ from $\mathsf{Cont}(base, meet, meetPoint)$. The left projection follows by associativity of history append. The right projection uses unary commutativity of $left$ and $right$, followed by the same associativity comparison. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection closed`

Theorem 93.2 (Principal suffix unary commuting square)**.** *Let* $base, left, right, leftPoint, rightPoint, lrPoint, rl$ BHist*. If* $left$ *and* $right$ *are unary histories and*

$$\mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint), \qquad \mathsf{Cont}(leftPoint, right, lrPoint), \qquad \mathsf{Cont}(rigl$$

then $lrPoint \mathsf{hsame} rlPoint$*.*

Proof. Unfold the four continuations as history appends. The left path displays $\mathsf{append}(\mathsf{append}(base, left), right)$, and the right path displays $\mathsf{append}(\mathsf{append}(base, right), left)$. Associativity moves both paths to the common $base$ prefix, while unary commutativity of $left$ and $right$ identifies the two suffix orders. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary commuting square`

Theorem 93.3 (Principal suffix unary intersection meet commutes)**.** *Let* $left, right, meet, commute$: BHist*. If* $left$ *and* $right$ *are unary histories and*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(right, left, commute),$$

then $meet\mathsf{hsame}commute$*.*

Proof. Apply the unary commuting square with empty base. The left and right base legs are the canonical left-unit continuations, so the two displayed intersection results are precisely the two suffix orders, and the square identifies them by hsame. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection meet commutes`

Theorem 93.4 (Filter principal suffix topology lattice route)**.** *The principal suffix filter surface may be consumed by the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood interface of chapter 88 and by metric convergence routes through chapter 57, but only through the displayed* BHist*,* Cont*, unary intersection,* hsame*, and* NameCert *rows. In particular, a topology or metric consumer can use the principal suffix row as a finite neighbourhood basis only after the intersection closure, commuting-square, and meet-commutation rows above have exposed the lattice route.*

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix topology lattice route`

Proof. Start with Definition 93.1. It supplies the finite intersection row for two unary suffix neighbourhoods over the same displayed BHist base.

Use Definition 93.2 and Definition 93.3. These rows identify the two orders of the unary suffix intersection by hsame, so the consumer sees a lattice route rather than an ordered path artifact.

The topology-facing read may therefore treat the principal suffix surface as a neighbourhood interface only through those displayed rows. The metric-facing read has the same boundary: convergence routes can cite the filter substrate, but no metric distance, open-ball predicate, limit selector, or quotient filter is introduced by $\mathsf{Filter}^{\uparrow}$ unless a separate sibling certificate supplies it. □

Theorem 93.5 (Principal suffix unary commuting square base determinism)**.** *Let* $base, base', left, right, leftPoint$ BHist*. If* $left$ *and* $right$ *are unary histories and*

$$\mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base', right, rightPoint), \qquad \mathsf{Cont}(leftPoint, right, meetPoint), \qquad \mathsf{Cont}($$

then $base\mathsf{hsame}base'$*.*

Proof. Unfold the two paths to the common meet. The first path gives $\mathsf{append}(\mathsf{append}(base, left), right)$ and the second gives $\mathsf{append}(\mathsf{append}(base', right), left)$. Associativity moves both to suffix form, unary commutativity aligns the common suffix $\mathsf{append}(left, right)$, and right cancellation of that suffix identifies the two bases. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary commuting square base deterministic`

Theorem 93.6 (Principal suffix commuted meet closure)**.** *Let* $base, left, right, meet, meetPoint$: BHist*. If* $left$ *and* $right$ *are unary histories and*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, meet, meetPoint)$$

hold, then the same $meetPoint$ *is displayed by the commuted suffix:*

$$\mathsf{Cont}(base, \mathsf{append}(right, left), meetPoint).$$

Proof. The first continuation identifies *meet* with $\mathsf{append}(left, right)$. Unary commutativity of *left* and *right* transports this suffix under the fixed prefix *base*, giving the displayed continuation by $\mathsf{append}(right, left)$. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection commuted meet closed`

Theorem 93.7 (Principal suffix commuted result determinism)**.** *Let* $base, left, right, meet, meetPoint, displayed$: BHist. *If* $left$ *and* $right$ *are unary histories and*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, meet, meetPoint), \qquad \mathsf{Cont}(base, \mathsf{append}(right, left), displayed),$$

then $meetPoint \mathsf{hsame} displayed$.

Proof. The commuted meet closure theorem displays *meetPoint* as a continuation of *base* by $\mathsf{append}(right, left)$. Continuation determinism compares this canonical commuted display with the given *displayed* history. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection commuted result deterministic`

Theorem 93.8 (Principal suffix unary intersection result determinism)**.** *In the situation of Definition 93.1, if another history displayed satisfies* $\mathsf{Cont}(rightPoint, left, displayed)$*, then* $meetPoint \mathsf{hsame} displayed$.

Proof. The closure theorem supplies the canonical continuation $\mathsf{Cont}(rightPoint, left, meetPoint)$. Continuation determinism compares this canonical display with the given displayed result. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection result deterministic`

Theorem 93.9 (Principal suffix unary intersection zero-result exclusion)**.** *Let* $base, left, right, meet, leftPoint, rig$ BHist. *If* $base$*,* $left$*, and* $right$ *are unary histories and*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint), \qquad \mathsf{Cont}(base, meet, \mathsf{E}$$

then $\bot$.

Proof. Unary closure transports through the concrete continuation $\mathsf{Cont}(left, right, meet)$, so *meet* is unary. Applying unary closure again to $\mathsf{Cont}(base, meet, \mathsf{E0}(z))$ would make the zero-headed result unary, which contradicts the unary no-zero-extension theorem. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection zero result absurd`

Theorem 93.10 (Principal suffix unary intersection e1-result cases)**.** *Let* $base, left, right, meet, leftPoint, rightP$ BHist. *If* $base$*,* $left$*, and* $right$ *are unary histories and*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint), \qquad \mathsf{Cont}(base, meet, \mathsf{E}$$

then either $meet = \mathsf{e}$ *and* $base \mathsf{hsame} \mathsf{E1}(z)$*, or there is a unary* $meetTail$ *such that* $meet = \mathsf{E1}(meetTail)$ *and* $\mathsf{Cont}(base, meetTail, z)$.

Proof. Unary closure along $\mathsf{Cont}(left, right, meet)$ first gives unary *meet*. The kernel unary e1-result case theorem applied to $\mathsf{Cont}(base, meet, \mathsf{E1}(z))$ then splits the result into the empty-meet case and the non-empty unary-tail case. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary intersection e1 result cases`

Theorem 93.11 (Principal suffix base points empty exactness)**.** *Let* $base, left, right, leftPoint, rightPoint$: BHist*. If*

$$\mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint),$$

and both displayed points satisfy

$$leftPoint\mathsf{hsamee}, \qquad rightPoint\mathsf{hsamee},$$

then $base\mathsf{hsamee}$*,* $left\mathsf{hsamee}$*, and* $right\mathsf{hsamee}$*.*

Proof. Transport the two displayed endpoints to e. Empty-result inversion for the first continuation identifies both *base* and $left$ with e; the same inversion for the second continuation identifies *right* with e. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix base points empty exact`

Theorem 93.12 (Principal suffix commuting square empty exactness)**.** *Let* $base, left, right, leftPoint, rightPoint$ BHist*. If the two displayed continuations*

$$\mathsf{Cont}(base, left, leftPoint), \qquad \mathsf{Cont}(base, right, rightPoint)$$

are extended around the square by

$$\mathsf{Cont}(leftPoint, right, lrPoint), \qquad \mathsf{Cont}(rightPoint, left, rlPoint),$$

and both square endpoints satisfy

$$lrPoint\mathsf{hsamee}, \qquad rlPoint\mathsf{hsamee},$$

then $base\mathsf{hsamee}$*,* $left\mathsf{hsamee}$*, and* $right\mathsf{hsamee}$*.*

Proof. Transport the two square endpoints to e. Empty-result inversion of $\mathsf{Cont}(leftPoint, right, lrPoint)$ identifies $leftPoint$ with e, and empty-result inversion of $\mathsf{Cont}(rightPoint, left, rlPoint)$ identifies $rightPoint$ with e. Apply the base-points empty exactness theorem to the two base continuations and these two transported endpoint equalities. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix unary commuting square empty exact`

Theorem 93.13 (Principal suffix intersection empty point exactness)**.** *Let* $base, left, right, meet, meetPoint$: BHist*. If*

$$\mathsf{Cont}(left, right, meet), \qquad \mathsf{Cont}(base, meet, meetPoint),$$

and the displayed intersection point satisfies $meetPoint\mathsf{hsamee}$*, then all four histories collapse to the empty history:*

$$base\mathsf{hsamee}, \qquad meet\mathsf{hsamee}, \qquad left\mathsf{hsamee}, \qquad right\mathsf{hsamee}.$$

Proof. Transport $meetPoint$ to e and apply empty-result inversion to $\mathsf{Cont}(base, meet, meetPoint)$. This identifies both *base* and *meet* with e. Substituting the empty meet into $\mathsf{Cont}(left, right, meet)$ and applying the same empty-result inversion identifies $left$ and *right* with e. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix intersection empty point exact`

Theorem 93.14 (Principal suffix intersection empty point forces base points empty)**.** *Let* $base, left, right, meet, le$
BHist*. If meet is the continuation of* $left$ *by* $right$*, the base continuations produce* $leftPoint$*,* $rightPoint$*, and* $meetPoint$*, and* $meetPoint$hsamee*, then both base points are empty:*

$$leftPoint\mathsf{hsamee}, \qquad rightPoint\mathsf{hsamee}.$$

Proof. Intersection empty-point exactness identifies the shared base, the left suffix, and the right suffix with e. Empty-append introduction applied to $\mathsf{Cont}(base, left, leftPoint)$ gives the empty left base point, and the same argument applied to $\mathsf{Cont}(base, right, rightPoint)$ gives the empty right base point. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix intersection empty point base points empty`

93.2 The certificate

Theorem 93.15 (Principal empty filter carrier semantic certificate)**.** *The BEDC unary carrier restricted to the empty history,*

$$\mathsf{UnaryHistory}(h) \wedge h\mathsf{hsamee},$$

has a semantic naming certificate with source, pattern, and ledger predicates all equal to this principal empty carrier, and with classifier hsame*.*

Proof. The witness is the empty history, which is unary and classifies to itself. Reflexivity, symmetry, and transitivity are the finite-kernel laws for hsame. If h lies in the principal empty carrier and $h\mathsf{hsame}k$, unary transport moves the unary carrier from h to k, while symmetry followed by transitivity sends the displayed comparison $h\mathsf{hsamee}$ to $k\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalEmptyCarrier semanticNameCert`

Theorem 93.16 (Principal visible carrier semantic certificate absurdity)**.** *For every tail* : BHist*, there is no semantic naming certificate whose source, pattern, and ledger predicates all require a history to be unary, to classify to* e*, and to classify to* $\mathsf{E1}(tail)$*:*

$$\mathsf{SemanticNameCert}(\{h \mid \mathsf{UnaryHistory}(h) \wedge h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(tail)\}, \{h \mid \mathsf{UnaryHistory}(h) \wedge h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(tai$$

Proof. The semantic certificate supplies a ledger witness for the displayed carrier. Its two classification comparisons compose by symmetry and transitivity to give $e\mathsf{hsameE1}(tail)$, which is excluded by BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalVisibleCarrier semanticNameCert absurd`

Theorem 93.17 (ContinuousMap image of a principal suffix filter)**.** *Let* $base, map, target, modulus, cert, distance,$
BHist*. Assume*

$$\begin{aligned}
&\mathsf{ContinuousMapCarrier}(base, map, target, modulus, cert, distance),\\
&\mathsf{UnaryHistory}(left), \quad \mathsf{UnaryHistory}(right),\\
&\mathsf{Cont}(left, right, meet), \quad \mathsf{Cont}(base, left, leftPoint), \quad \mathsf{Cont}(base, right, rightPoint), \quad \mathsf{Cont}(base, meet, meetP
\end{aligned}$$

Then there exist histories $targetLeft, targetRight, targetMeet, imageLeft, imageRight, imageMeet$: BHist *for which*

$$\mathsf{Cont}(target, left, targetLeft) \wedge \mathsf{Cont}(target, right, targetRight) \wedge \mathsf{Cont}(target, meet, targetMeet),$$
$$\mathsf{UnaryHistory}(targetMeet) \wedge \mathsf{Cont}(targetLeft, right, targetMeet) \wedge \mathsf{Cont}(targetRight, left, targetMeet),$$
$$\mathsf{Cont}(leftPoint, map, imageLeft) \wedge \mathsf{Cont}(rightPoint, map, imageRight) \wedge \mathsf{Cont}(meetPoint, map, imageMeet),$$
$$\mathsf{hsame}(targetLeft, imageLeft) \wedge \mathsf{hsame}(targetRight, imageRight) \wedge \mathsf{hsame}(targetMeet, imageMeet).$$

Moreover, these target-side witnesses determine the intersection projections, the target commuting square, and the graph-image displays:

$$\forall z : \mathsf{BHist}.\ \mathsf{Cont}(targetLeft, right, z) \Longrightarrow \mathsf{hsame}(targetMeet, z),$$
$$\forall z : \mathsf{BHist}.\ \mathsf{Cont}(targetRight, left, z) \Longrightarrow \mathsf{hsame}(targetMeet, z),$$
$$\forall lr, rl : \mathsf{BHist}.\ \mathsf{Cont}(targetLeft, right, lr) \Longrightarrow \mathsf{Cont}(targetRight, left, rl) \Longrightarrow \mathsf{hsame}(lr, rl),$$
$$\forall z : \mathsf{BHist}.\ \mathsf{Cont}(leftPoint, map, z) \Longrightarrow \mathsf{hsame}(targetLeft, z),$$
$$\forall z : \mathsf{BHist}.\ \mathsf{Cont}(rightPoint, map, z) \Longrightarrow \mathsf{hsame}(targetRight, z),$$
$$\forall z : \mathsf{BHist}.\ \mathsf{Cont}(meetPoint, map, z) \Longrightarrow \mathsf{hsame}(targetMeet, z).$$

Proof. By Definition 130.1, the continuous-map carrier contains a continuous-function carrier. Applying Definition 59.139 to that component gives unary histories for *base*, *map*, and *target*, together with the graph continuation $\mathsf{Cont}(base, map, target)$. The source hypotheses and unary continuation closure give unary evidence for *meet*.

Choose $targetLeft$, $targetRight$, and $targetMeet$ from the generated continuations of *target* by $left$, *right*, and *meet*. Choose $imageLeft$, $imageRight$, and $imageMeet$ from the generated continuations of $leftPoint$, $rightPoint$, and $meetPoint$ by *map*.

Apply Definition 93.1 with base *target*. Its inputs are the unary target history, the unary suffixes $left$ and *right*, the source meet ledger $\mathsf{Cont}(left, right, meet)$, and the three chosen target continuations. This gives $\mathsf{UnaryHistory}(targetMeet)$ and the two projection continuations $\mathsf{Cont}(targetLeft, right, targetMeet)$ and $\mathsf{Cont}(targetRight, left, targetMeet)$.

For the image readback of $left$, apply Definition 93.2 to the square with base *base*, suffixes $left$ and *map*, base legs $\mathsf{Cont}(base, left, leftPoint)$ and $\mathsf{Cont}(base, map, target)$, and upper legs $\mathsf{Cont}(leftPoint, map, imageLeft)$ and $\mathsf{Cont}(target, left, targetLeft)$. The result identifies $imageLeft$ with $targetLeft$, and symmetry gives the stated orientation. The same argument with *right* gives $\mathsf{hsame}(targetRight, imageRight)$. The meet case uses the unary meet evidence, the base leg $\mathsf{Cont}(base, meet, meetPoint)$, and the same graph leg to obtain $\mathsf{hsame}(targetMeet, imageMeet)$.

The two projection-determinacy clauses follow by comparing the canonical target projection continuations with any displayed projection through Definition 21.7. The universal target commuting square is exactly Definition 93.2 applied at base *target* to arbitrary displayed upper legs. Finally, each graph-image determinacy clause compares the chosen image continuation with an arbitrary image continuation by continuation determinacy, and then composes that comparison with the corresponding target-image equality using the equivalence laws for hsame from Definition 3.8. □

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix continuousMap image point determinacy`
Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix continuousMap image package`

Theorem 93.18 (ContinuousMap image determinacy for principal suffix points)**.** *Let* $base, map, target, modulus, c$ BHist. *Assume the base continuous-map row*

$$\mathsf{ContinuousMapCarrier}(base, map, target, modulus, cert, distance),$$

the principal suffix source and target displays

$$\mathsf{UnaryHistory}(suffix), \qquad \mathsf{Cont}(base, suffix, sourcePoint), \qquad \mathsf{Cont}(target, suffix, targetPoint),$$

and two displayed image rows over the same source point and the same graph history:

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(sourcePoint, map, target_0, modulus_0, cert_0, distance_0),\\ &\mathsf{ContinuousMapCarrier}(sourcePoint, map, target_1, modulus_1, cert_1, distance_1).\end{aligned}$$

Then the two displayed target histories are determined, and both read back to the target-side principal suffix point:

$$\mathsf{hsame}(target_0, target_1) \wedge \mathsf{hsame}(targetPoint, target_0) \wedge \mathsf{hsame}(targetPoint, target_1).$$

Lean4 ✓ `BEDC.Derived.FilterUp.ContinuousMap image principal suffix point determinacy`

Proof. Project the continuous-function component from the base carrier using Definition 130.1. Applying Definition 59.139 to that component gives the graph continuation $\mathsf{Cont}(base, map, target)$ and unary membership of the graph history *map*.

Project the continuous-function components from the two displayed image rows in the same way. Their graph fields give

$$\mathsf{Cont}(sourcePoint, map, target_0) \qquad \text{and} \qquad \mathsf{Cont}(sourcePoint, map, target_1).$$

Apply Definition 93.2 with base *base*, left suffix *suffix*, right suffix *map*, left point *sourcePoint*, and right point *target*. The first displayed image row supplies the upper-left path and $\mathsf{Cont}(target, suffix, targetP$ supplies the upper-right path, so the square gives $\mathsf{hsame}(target_0, targetPoint)$. The same square applied to the second displayed image row gives $\mathsf{hsame}(target_1, targetPoint)$.

Symmetry and transitivity of hsame from Definition 3.8 orient these two comparisons as $\mathsf{hsame}(targetPoint, target_0)$ and $\mathsf{hsame}(targetPoint, target_1)$ and compose them to obtain $\mathsf{hsame}(target_0, target_1)$. Pairing the three comparisons gives the displayed conclusion. □

Lean4 ✓ `BEDC.Derived.FilterUp.ContinuousMapCarrier principal suffix point determinacy`

Theorem 93.19 (Principal suffix upward closure)**.** *Let* $base, suffix, point, extra, extended : \mathsf{BHist}$. *If* $base$, $suffix$, *and* $extra$ *are unary histories and*

$$\mathsf{Cont}(base, suffix, point), \qquad \mathsf{Cont}(point, extra, extended),$$

then $point$ *and* $extended$ *are unary histories, and there is a unary history* $combined : \mathsf{BHist}$ *such that*

$$\mathsf{Cont}(suffix, extra, combined) \qquad \textit{and} \qquad \mathsf{Cont}(base, combined, extended).$$

Consequently, the principal suffix family generated by base is closed under unary continuation extensions of its points.

Lean4 ✓ `BEDC.Derived.FilterUp.FilterPrincipalSuffix upward closed`

Proof. Unary continuation closure gives unary *point* from the base display and unary *extended* from the point extension. Put $combined = \mathsf{append}(suffix, extra)$. The continuation rules of Definition 4.10 give $\mathsf{Cont}(suffix, extra, combined)$, and unary continuation closure gives unary *combined*.

Let $direct = \mathsf{append}(base, combined)$, with the canonical continuation $\mathsf{Cont}(base, combined, direct)$. Applying Definition 4.56 to the four ledgers

$$\mathsf{Cont}(base, suffix, point), \qquad \mathsf{Cont}(point, extra, extended), \qquad \mathsf{Cont}(suffix, extra, combined), \qquad \mathsf{Cont}(base, c$$

gives $extended\mathsf{hsame}direct$. The reverse direction of Definition 21.8 for $\mathsf{Cont}(base, combined, direct)$ transports the direct composite result to the displayed endpoint $extended$, yielding $\mathsf{Cont}(base, combined, extended)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{Filter}\uparrow}$**.**

Constructive story: A $\mathsf{Filter}^{\uparrow}$ packet is generated from finite principal-suffix and continuous-map image histories, their classifier readback rows, and the displayed Cont witnesses relating source, suffix, map, and target histories.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Filter}\uparrow}$, maturePackageClosed)

Scope closed: The principal-suffix public rows through Definition 93.19 and the continuous-map-image consumer rows through Definition 93.17 and Definition 93.18 form the public $\mathsf{Filter}^{\uparrow}$ export surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Filter}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.FilterUp.FilterPrincipalEmptyCarrier\_semanticNameCert`

Bridge status: none

Origin: human

Not claimed: The chapter does not close ultrafilters, quotient filters, choice-based limits, or standard bridges.

Upgrade path: No further theory-axis upgrade is claimed here; standard bridge checking remains a separate verification-axis task.

Chapter 94

A Concrete Naming Certificate for $\mathsf{FunctionalAnalysis}^{\uparrow}$

The functional-analysis interface $\mathsf{FunctionalAnalysis}^{\uparrow}$ collects the bounded-linear-operator and dual-space certificates over $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90) and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91). It packages the operator norm, adjoints, and compactness witnesses required by the spectral, semigroup, and PDE certificates downstream.

94.1 Carrier definition

Definition 94.1 (FunctionalAnalysis bounded-linear-operator face)**.** *The bounded-linear-operator face of* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *is the row family obtained by reading* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *sources* C, D *through Definition 90.49. Its carrier is the displayed Banach row*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda),$$

where T *is the operator representative,* K *is the* $\mathsf{Real}^{\uparrow}$ *norm bound, and* Λ *is the visible source-target, linearity, norm, scalar, bound, and provenance ledger. Its classifier is the Banach bounded-operator classifier: endpoint-map transport by* hsame, $\mathsf{Real}^{\uparrow}$ *transport of the bound endpoint, and componentwise ledger transport.*

Definition 94.2 (FunctionalAnalysis dual-space face)**.** *For a* $\mathsf{Banach}^{\uparrow}$ *source* C*, the dual-space face of* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *is the subface of Definition 94.1 whose target is the scalar singleton target supplied by* $\mathsf{Real}^{\uparrow}$ *as a Banach scalar carrier. Thus a dual row is a carried bounded linear functional*

$$\mathsf{BanachBLOp}(C, \mathsf{Real}^{\uparrow}, \varphi, K, \Lambda_{\varphi})$$

with the same bounded-operator classifier, the same $\mathsf{Real}^{\uparrow}$ *bound endpoint, and a ledger whose target endpoint rows are scalar rows rather than a separate vector target.*

Theorem 94.3 (FunctionalAnalysis operator-norm stability obligation)**.** *If two bounded-linear-operator rows in Definition 94.1 are related by the inherited Banach bounded-operator classifier, then they expose the same* $\mathsf{Real}^{\uparrow}$ *operator-norm bound row and the same visible norm-bound ledger row. Consequently every downstream* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer reads the operator representative together with a transported bound endpoint rather than a host-level operator equality.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisBoundedLinearOperator norm bound stable`

Proof. Read the operator face through Definition 90.49. The classifier specified there has three visible components: endpoint-map comparison, $\mathsf{Real}^{\uparrow}$ bound comparison, and componentwise ledger transport.

Apply Definition 90.52. Its exactness clause states that accepted bounded-operator rows contain precisely the endpoint map, classifier-respect row, additive row, scalar row, bound row, and provenance ledger. The $\mathsf{Real}^{\uparrow}$ comparison component therefore transports the norm-bound endpoint, and the ledger component transports the row recording where that bound is used.

The metric consequence is the Lipschitz row of Definition 90.50. Since that theorem reads the same bound endpoint K, the downstream operator-norm use is stable under the inherited classifier. □

Theorem 94.4 (FunctionalAnalysis operator-norm chapter route)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that reads an operator-norm row factors through the public* $\mathsf{OperatorNorm}^{\uparrow}$ *carrier of chapter 3603 together with the existing bounded-operator row and* $\mathsf{Real}^{\uparrow}$ *bound endpoint displayed in Definition 94.1. The route exports only the transported operator representative, nonnegative bound endpoint, norm-bound ledger, componentwise classifier transport, replay, provenance, and local naming rows; it refuses host norm functions, quotient operator equality, and hidden Banach completeness.*

Proof. Apply Definition 94.3. It already identifies the inherited Banach bounded-operator classifier, the transported $\mathsf{Real}^{\uparrow}$ bound endpoint, and the visible norm-bound ledger as the data available to every downstream $\mathsf{FunctionalAnalysis}^{\uparrow}$ operator-norm consumer.

The public chapter chapter 3603 names exactly that scalar-bound surface as $\mathsf{OperatorNorm}^{\uparrow}$. Reading through its carrier repacks the bounded-operator row, the nonnegative bound endpoint, and the ledger without adding any host-level norm function.

The refused objects are outside both routes. The stability theorem never supplies quotient operator equality, and the public $\mathsf{OperatorNorm}^{\uparrow}$ carrier contains no coordinate for hidden Banach completeness, so the FunctionalAnalysis consumer factors through the named chapter boundary. □

Theorem 94.5 (FunctionalAnalysis adjoint stability obligation)**.** *For Hilbert-compatible sources* H, G *whose carrier and classifier rows are those of Definition 91.1, an adjoint row in* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *is stable under bounded-operator classifier transport. The defining comparison*

$$\langle Tx, y\rangle_G \sim_{\mathsf{Real}^{\uparrow}} \langle x, T^* y\rangle_H$$

uses the same Hilbert carrier, the same classifier, and the same inner-product rows before and after transport, so the adjoint ledger row is preserved componentwise.

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisBoundedOperatorRow adjoint stability`

Proof. Begin with the Hilbert source. By Definition 91.5, every accepted Hilbert row exposes a Banach surface, an inner-product surface, and a ledger binding both to the same carrier and classifier.

The bounded-operator part is handled by Definition 90.52. Transport of T and T^* moves the endpoint representatives, the bound endpoints, and the operator ledgers without changing the Hilbert carrier on which the inner-product rows are evaluated.

Finally apply the carrier-sharing clause of Definition 91.1. The two inner-product comparisons in the displayed adjoint equation are read through the same classifier $\sim_X$ and the same norm-induction ledger. Hence the adjoint row is transported as a row of the Hilbert-compatible package. □

Theorem 94.6 (FunctionalAnalysis compact-operator witness obligation)**.** *A compact-operator witness in* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *is a bounded-linear-operator row* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$ *together with* $\mathsf{CompactMetric}^{\uparrow}$ *probe data for the image of the carried source unit-bound face in the target Banach metric. If the bounded-operator representative is replaced by an inherited classifier-related representative, then the witness is transported to the new representative and records the same Banach or Hilbert compactness probe ledger.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisCompactOperatorWitness transport`

Proof. Use Definition 94.1 for the operator row. The bounded-operator classifier transports the endpoint map, the bound endpoint, and every source-target ledger component.

The compactness data are attached only to carried source endpoints, target metric endpoints, and finite or stream probe rows in $\mathsf{CompactMetric}^{\uparrow}$. Those target metric endpoints are the Banach norm-induced rows, so Definition 90.52 supplies the operator-side transport needed to move the image probes.

When the source and target are Hilbert-compatible, the same compactness row is read through Definition 91.5. The Hilbert surface contributes no new carrier to the compactness witness; it only records that the Banach metric endpoint is the norm induced by the displayed inner-product row. □

Theorem 94.7 (FunctionalAnalysis exact ledger boundary obligation)**.** *The* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *ledger exposed by this chapter is exactly the operator representative, the* $\mathsf{Real}^{\uparrow}$ *norm-bound row, the dual or adjoint row when present, the compactness witness when present, and the downstream consumer boundary for* $\mathsf{SpectralTheorem}^{\uparrow}$*,* $\mathsf{Semigroup}^{\uparrow}$*, and* $\mathsf{Pde}^{\uparrow}$*. It does not expose a spectral theorem, a semigroup evolution theorem, a PDE well-posedness theorem, or any standard bridge beyond those rows.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisBoundedOperatorRow ledger hsame transport`

Proof. The bounded-operator part is exactly the ledger listed in Definition 90.49 and certified by Definition 90.52. It contains the source endpoint, target endpoint, norm, scalar, compatibility, bound, and provenance rows.

The Hilbert part, when present, is exactly the compatible carrier and shared ledger of Definition 91.1 together with the obligation surface of Definition 91.5. Thus adjoint rows are ledger rows over the same carrier and classifier, not independent public objects.

The remaining fields follow the ten-field template of section 9467.18: traditional target, BEDC source, carrier, classifier, stability obligations, ledger, core theorem list, exactness predicate, standard bridge, and status. For this chapter the standard-bridge and downstream theorem entries are recorded as boundaries, so the ledger stops exactly at the rows displayed in the statement. □

94.2 The certificate

Theorem 94.8 (FunctionalAnalysis bounded-linear-operator obligation)**.** *A* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *bounded-operator row over a* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *source and target consists of the following displayed data:*

1. *a carried* $\mathsf{LinMap}^{\uparrow}$ *map-code carrier for the underlying map;*
2. *source and target carrier predicates supplied by the Banach or Hilbert certificates named in the chapter opening;*

3. *the pointwise* LinMap$^{\uparrow}$ *classifier on carried source inputs;*
4. *an operator-norm witness landing in the* Real$^{\uparrow}$ *nonnegative comparison surface;*
5. *a norm-bound ledger recording the input norm row, output norm row, bound endpoint, and classifier transport used by the comparison.*

The stability field of this row is inherited from the LinMap$^{\uparrow}$ *pointwise classifier and from the Banach or Hilbert carrier and norm classifiers, with no extra equality relation on maps.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisBoundedOperatorRow visible ledger`

Proof. Read the Banach and Hilbert dependencies from the chapter opening. They supply the carried vector endpoints, norm rows, distance rows, and classifier transport used by a bounded-operator comparison.

Next read the map part through LinMap$^{\uparrow}$. The singleton and pointwise classifier surface of LinMap$^{\uparrow}$, including Definition 48.1 and Definition 48.49, supplies the carried map-code endpoint and identifies map sameness with pointwise codomain classifier sameness on carried source inputs.

The operator-norm witness is then only a ledgered RealUp bound: for each carried input, the source norm row, output norm row, and bound endpoint are recorded together with the RealUp comparison. Transporting the source input along the LinMap$^{\uparrow}$ classifier and the Banach or Hilbert norm classifier transports the same bound row, so the bounded-operator classifier has exactly the inherited stability claimed above. □

Theorem 94.9 (FunctionalAnalysis dual-space classifier obligation)**.** *For a* Banach$^{\uparrow}$ *source, the* FunctionalAnalysis$^{\uparrow}$ *dual-space carrier is the row of bounded scalar-valued* LinMap$^{\uparrow}$ *map codes from the Banach carrier to the scalar carrier. Its classifier compares two dual rows pointwise: for every carried Banach input, their scalar outputs are related by the scalar* NameCert *classifier. Reflexivity, symmetry, and transitivity of the dual classifier are inherited from the scalar* NameCert *classifier and from the* LinMap$^{\uparrow}$ *pointwise classifier package.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisDualPointwiseClassifier equivalence rows`

Proof. The source side is the Banach certificate named in the chapter opening. The target side is the scalar NameCert used by the Banach norm and bound rows. Together they determine the scalar-valued LinMap$^{\uparrow}$ carrier: a functional is a carried linear-map code whose output carrier is the scalar carrier and whose bound ledger is a RealUp comparison row.

For classification, unfold the LinMap$^{\uparrow}$ pointwise classifier surface. Two carried scalar-valued map codes are compared by applying both to an arbitrary carried Banach input and comparing the scalar outputs. Scalar reflexivity gives the diagonal comparison, scalar symmetry reverses each pointwise comparison, and scalar transitivity composes two pointwise comparisons.

Because these three operations are performed pointwise and then repacked by Definition 48.49, the dual-space classifier inherits refl/symm/trans from the scalar NameCert and the LinMap$^{\uparrow}$ classifier package alone. □

Theorem 94.10 (FunctionalAnalysis compact-operator ledger obligation)**.** *A* FunctionalAnalysis$^{\uparrow}$ *compact-operator witness for a carried bounded operator is an explicit finite-net image ledger: for each displayed* ProbeBundle *of bounded source inputs and each displayed target tolerance row, the ledger gives finitely many carried target probes whose classifier balls cover the carried image probes. The witness is stable under the bounded-operator classifier because pointwise target classifier*

transport moves every image probe and its finite-net comparison across classifier-equal bounded operators. This is not an open-cover compactness assertion and uses no choice-based extraction of subcovers.

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisCompactOperatorWitness probe ledger`

Proof. Start with the bounded-operator row of Definition 94.8. It gives the carried $\mathsf{LinMap}^{\uparrow}$ map code, pointwise target classifier control, and the norm-bound ledger needed to restrict attention to bounded source probes.

The compactness datum is recorded as a finite ProbeBundle-style ledger. For a displayed source probe bundle and a displayed target tolerance row, the ledger names a finite list of carried target probes and, for each image probe, the selected finite-net point together with the target classifier comparison.

If two bounded operators are related by the bounded-operator classifier, then their image endpoints over every carried source probe are pointwise related in the target classifier. Transport each finite-net comparison across that pointwise relation and keep the same finite list of target probes. Thus the compact-operator witness is classifier-stable while remaining a finite ledger, not an open-cover or choice principle. □

Theorem 94.11 (FunctionalAnalysis public export certificate)**.** *The public* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *export over the displayed* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CompactMetric}^{\uparrow}$ *sources is the single* $\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}$ *surface whose fields are exactly the bounded-operator row, the dual-space classifier, the operator-norm stability row, the adjoint stability row, the compact finite-net witness row, and the exact ledger boundary proved in this chapter. The surface exports only these rows for downstream public consumers, and it exports no spectral theorem, semigroup evolution theorem, PDE well-posedness theorem, standard bridge, or host-level operator equality.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisPublic export certificate`

Proof. The carrier and classifier part is the bounded-operator row of Definition 94.8 together with the dual-space classifier of Definition 94.9. These two rows determine the public operator and scalar-valued functional faces over the displayed $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources.

The stability fields are precisely the operator-norm and adjoint rows of Definition 94.3 and Definition 94.5. The compact field is the finite-net witness ledger of Definition 94.10, read over the displayed $\mathsf{CompactMetric}^{\uparrow}$ surface rather than through an open-cover compactness assertion.

The boundary of the public surface is the exact ledger boundary of Definition 94.7. Therefore the exported $\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}$ surface consists exactly of the named rows and explicitly omits spectral theorem, semigroup evolution, PDE well-posedness, standard-bridge, and host-level operator-equality claims. □

Theorem 94.12 (FunctionalAnalysis public bridge boundary)**.** *The public* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *surface over the displayed* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CompactMetric}^{\uparrow}$ *rows exports exactly the bounded-operator row, dual pointwise classifier, norm-bound stability, adjoint stability, compact finite-net witness, and exact ledger boundary as the bridge payload available to downstream* $\mathsf{SpectralTheorem}^{\uparrow}$*,* $\mathsf{Semigroup}^{\uparrow}$*, and* $\mathsf{Pde}^{\uparrow}$ *consumers. This boundary does not assert spectral theorem rows, semigroup evolution, PDE well-posedness,* bridgeChecked*, or host-level operator equality.*

Lean4 ✓ `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisPublic bridge boundary`

Proof. The exported operator face is the bounded-operator row of Definition 94.8. The exported dual face is the pointwise scalar-valued classifier of Definition 94.9. Together these two rows give the public carrier and classifier entries consumed by the operator chapters.

The stability payload is exactly the norm-bound stability row of Definition 94.3 and the adjoint stability row of Definition 94.5. These rows transport the exported operator data without adding any spectral, evolution, or PDE conclusion.

The compact payload is the finite-net witness of Definition 94.10, and the outer boundary is the exact ledger boundary of Definition 94.7. Repacking these six named rows gives precisely the public bridge payload for $\mathsf{SpectralTheorem}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, and $\mathsf{Pde}^{\uparrow}$, while the omitted claims remain outside this $\mathsf{FunctionalAnalysis}^{\uparrow}$ certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}$.

Constructive story: A functional-analysis row is assembled from already displayed $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$ surfaces: the bounded-operator carrier contains a carried map code, source and target carriers, a RealUp norm-bound endpoint, and a visible ledger of source, target, linearity, norm, scalar, bound, and provenance rows. Two bounded operators or dual rows are classified only by the inherited pointwise map classifier, RealUp bound transport, and componentwise ledger transport; there is no host-level operator equality, and a dual-space row is just the scalar-target subface of the same bounded-operator construction. The source covers bounded operators, scalar-valued functionals, Hilbert adjoint rows, and compact finite-net witnesses, while the pattern records norm-bound stability, adjoint stability, pointwise dual equivalence, compact probe transport, exact ledger boundary, public export, and downstream bridge payload for spectral, semigroup, and PDE consumers. The five name-certificate obligations are inherited row by row: inhabited operator faces come from displayed accepted map codes when a row is present, pointwise scalar classifiers and RealUp classifiers give reflexivity, symmetry, and transitivity, and the bounded-operator and compact-witness transport theorems keep every accepted ledger stable under the classifier. A history package h lies in the signature gap of a $\mathsf{FunctionalAnalysis}^{\uparrow}$ token p when the source-domain accepts the bounded-operator, dual, adjoint, or compact row, the signature relation pairs it with an answer row and evidence ledger, and token introduction names that answer as p; unfolded, the token can see only the transported map code, bound endpoint, pointwise scalar outputs, finite-net probes, and listed ledger rows. The paper bridge boundary exports exactly those six row families to $\mathsf{SpectralTheorem}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, and $\mathsf{Pde}^{\uparrow}$ consumers, without turning them into standard operator theory. The chapter registers this bounded-operator and dual-ledger surface under the name $\mathsf{FunctionalAnalysis}^{\uparrow}$; it introduces no spectral theorem, PDE well-posedness theorem, semigroup evolution theorem, host equality, or bridge-checked standard model.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is exactly the bounded-operator row, dual pointwise classifier, norm-bound stability, adjoint stability, compact finite-net witness, exact ledger boundary, public export certificate, and public bridge boundary for downstream $\mathsf{SpectralTheorem}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, and $\mathsf{Pde}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FunctionalAnalysisUp.FunctionalAnalysisBoundedOperatorRow\_visible\_ledger`

Bridge status: paperBridge

Not claimed: Spectral theorem rows, PDE well-posedness, semigroup evolution, host equality, and bridgeChecked remain outside this mature paper claim.

Upgrade path: The next verification work is checked standard-bridge verification for the six displayed rows and their downstream certificates, independent of any additional paper closure grade.

Chapter 95

A Concrete Naming Certificate for $\mathsf{Completion}^{\uparrow}$

The completion interface $\mathsf{Completion}^{\uparrow}$ packages the universal regular-Cauchy completion of a metric naming certificate. It depends on $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ (chapter 57) and the real-completeness capstone (chapter 9238), generalising the $\mathsf{Rat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ construction to arbitrary located metric carriers.

95.1 Carrier definition

Theorem 95.1 (Completion metric distance visible-tail readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and the metric distance witness between x_n and y_n has visible result $\mathsf{E1}(t)$, then the visible tail t is unary and the rational endpoint classifier holds between x_n and y_n.*

Proof. The metric distance witness contains a unary witness for its visible result; inversion of unary histories through $\mathsf{E1}$ gives unary membership of t. The real stream prefix classifier supplies the rational endpoint classifier at the selected index. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness e1 tail real prefix readback`

Theorem 95.2 (Completion empty-distance real-prefix readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and the metric distance witness between x_n and y_n has empty result, then both selected endpoints classify as e and the rational endpoint classifier holds between x_n and y_n.*

Proof. The empty-distance exactness theorem for metric distance witnesses reads back the two endpoint classifiers against e. The real stream prefix classifier supplies the rational endpoint classifier at the selected index. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness empty distance real prefix readback`

Theorem 95.3 (Completion visible-tail excludes empty distance)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and the metric distance witness between x_n and y_n has visible result $\mathsf{E1}(t)$, then the rational endpoint classifier holds between x_n and y_n, and no empty metric distance witness can classify the same endpoints.*

Proof. Visible-tail readback supplies the rational endpoint classifier. If an empty distance witness also classified the same endpoints, metric result determinacy would identify $\mathsf{E1}(t)$ with e, contradicting the BEDC no-confusion theorem for visible constructors. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness e1 tail empty distance absurd`

Theorem 95.4 (Completion visible-tail transport)**.** *Let x, x', y, y' be real stream representatives and let n be a prefix index. If every endpoint of x is* hsame*-equal to the corresponding endpoint of x' and every endpoint of y is* hsame*-equal to the corresponding endpoint of y', then a real prefix classifier and a metric distance witness from x_n to y_n with visible result $\mathsf{E1}(t)$ transport to the primed streams. The transported witness still reads back unary membership of t and the rational endpoint classifier between x'_n and y'_n.*

Proof. Transport the finite real prefix classifier endpointwise. Transport the metric distance continuation ledger along the endpoint equalities while keeping the visible result fixed, and transport the unary endpoint carriers contained in the witness. The visible-tail readback theorem applied to the transported prefix and witness gives the unary tail and rational endpoint classifier. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness e1 tail hsame transport`

Theorem 95.5 (Completion metric distance visible-tail determinacy)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and two metric distance witnesses between x_n and y_n have visible results $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$, then d and e are unary, their tails are* hsame*-equal, and the rational endpoint classifier holds between x_n and y_n.*

Proof. Apply the visible-tail readback to each distance witness to obtain unary evidence for both tails and the rational endpoint classifier. The metric distance result determinacy theorem compares the two visible results over the same endpoints; e1 inversion for hsame descends this comparison to $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness e1 result tail deterministic`

Theorem 95.6 (Completion symmetric visible-tail readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n, a metric distance witness from x_n to y_n has visible result $\mathsf{E1}(d)$, and a metric distance witness from y_n to x_n has visible result $\mathsf{E1}(e)$, then d and e are unary, $d\mathsf{hsame}e$, and the rational endpoint classifier holds between y_n and x_n.*

Proof. Symmetry of the real stream prefix classifier gives the rational endpoint classifier in the reversed direction. Visible-tail readback applies to both witnesses. The symmetric metric distance classifier compares the two visible result histories, and e1 inversion for hsame descends the comparison to the tails. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness e1 symmetric tail real prefix readback`

Theorem 95.7 (Completion metric distance visible-context tail readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and a visible context displays the metric distance witness with central result $\mathsf{E1}(t)$, then t is unary and the rational endpoint classifier holds between x_n and y_n.*

Proof. Apply visible-context cancellation for metric distance witnesses to recover the central witness. Its $\mathsf{E1}(t)$ distance carrier inverts to unary membership of t, and the real stream prefix classifier supplies the rational endpoint classifier at n. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 tail real prefix readback`

Theorem 95.8 (Completion visible-context tail transport)**.** *Let x, x' and y, y' be pointwise history-same real stream representatives. If the real stream prefix classifier holds for x, y at n and a visible context displays a metric distance witness with central result $\mathsf{E1}(t)$, then the transported representatives x', y' have the same prefix classifier, the same visible-context metric witness, unary tail evidence for t, and the rational endpoint classifier at n.*

Proof. Cancel the visible prefix and suffix to recover the central metric witness. Pointwise hsame transports the real prefix classifier and the central continuation ledger. Rebuild the visible-context witness from the transported central witness, then apply visible-context tail readback to recover the unary tail and the transported rational endpoint classifier. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 tail hsame transport`

Theorem 95.9 (Completion visible-context empty distance real-prefix readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and a visible context displays the metric distance witness with central empty distance, then the visible prefix and suffix are unary, both central endpoints classify as e, and the rational endpoint classifier holds between x_n and y_n.*

Proof. Apply visible-context empty-distance exactness for metric distance witnesses. This reads back unary evidence for the displayed prefix and suffix together with the two empty endpoint classifiers. The real stream prefix classifier supplies the rational endpoint classifier at the selected index. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context empty distance real prefix readback`

Theorem 95.10 (Completion visible-context empty-distance transport)**.** *Let x, x', y, y' be real stream representatives such that $x_m \mathsf{hsame} x'_m$ and $y_m \mathsf{hsame} y'_m$ for every index m. If the real stream prefix classifier holds for x, y at n and a visible-context metric distance witness from x_n to y_n has central empty distance, then the transported prefix classifier holds for x', y' at n, the corresponding visible-context empty-distance witness holds for x'_n, y'_n, the visible prefix and suffix are unary, both transported central endpoints classify as e, and the rational endpoint classifier holds between x'_n and y'_n.*

Proof. Apply empty-distance visible-context exactness to the original witness. This recovers unary evidence for the visible prefix and suffix and empty endpoint classifiers for x_n and y_n. Transport the real stream prefix classifier along the pointwise hsame hypotheses, transport the two endpoint empty classifiers by symmetry and transitivity of hsame, and rebuild the visible-context empty-distance witness from the transported data. The transported real prefix classifier supplies the rational endpoint classifier at the selected index. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context empty distance hsame transport`

Theorem 95.11 (Completion visible-context symmetric tail readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n, a visible-context metric distance witness from x_n to y_n has central result* $\mathsf{E1}(d)$*, and a visible-context metric distance witness from y_n to x_n has central result* $\mathsf{E1}(e)$*, then d and e are unary,* $d\mathsf{hsame}e$*, and the rational endpoint classifier holds between y_n and x_n.*

Proof. Apply visible-context exactness to both displayed distance witnesses. This recovers the two central metric distance witnesses with visible results $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$. The symmetric visible-tail readback theorem then extracts both unary tails, compares them by hsame, and reads back the reversed rational endpoint classifier. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 symmetric tail real prefix readback`

Theorem 95.12 (Completion metric distance visible-context tail determinacy)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and two visible contexts display metric distance witnesses with central results* $\mathsf{E1}(d)$ *and* $\mathsf{E1}(e)$*, then d and e are unary,* $d\mathsf{hsame}e$*, and the rational endpoint classifier holds between x_n and y_n.*

Proof. Apply visible-context tail readback to both displayed witnesses to obtain unary membership for each tail and the rational endpoint classifier. Visible metric result determinacy compares the two displayed central $\mathsf{E1}$ results, and $\mathsf{E1}$ inversion for hsame descends the comparison to $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 tail deterministic`

Theorem 95.13 (Completion prefix-independent visible-context tail determinacy)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n and two visible contexts, possibly with different left and right context histories, display metric distance witnesses with central results* $\mathsf{E1}(d)$ *and* $\mathsf{E1}(e)$*, then d and e are unary,* $d\mathsf{hsame}e$*, and the rational endpoint classifier holds between x_n and y_n.*

Proof. Visible-context tail readback applies to both displayed witnesses and supplies the two unary tail witnesses together with the rational endpoint classifier. Prefix-independent metric result determinacy compares the displayed central $\mathsf{E1}$ results across the two contexts. Inverting $\mathsf{E1}$ under hsame descends the comparison to $d\mathsf{hsame}e$. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 prefix independent tail deterministic`

Theorem 95.14 (Completion prefix-independent visible-context symmetric tail readback)**.** *Let x and y be real stream representatives and let n be a prefix index. If the real stream prefix classifier holds at n, a visible-context metric distance witness from x_n to y_n has central result* $\mathsf{E1}(d)$*, and an independently contextualized visible metric distance witness from y_n to x_n has central result* $\mathsf{E1}(e)$*, then d and e are unary,* $d\mathsf{hsame}e$*, and the rational endpoint classifier holds between y_n and x_n.*

Proof. Visible-context tail readback applies to the first displayed witness, while symmetry of the real stream prefix classifier supplies the readback direction for the second witness. The prefix-independent symmetric metric classifier compares the two displayed $\mathsf{E1}$ results, and $\mathsf{E1}$ inversion for hsame descends the comparison to the tails. □

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness visible context e1 prefix independent symmetric tail readback`

95.2 The certificate

Theorem 95.15 (Concrete CompletionUp naming certificate)**.** *The* Completion$^{\uparrow}$ *chapter carries a concrete naming-certificate boundary for the universal regular-Cauchy completion of a metric naming certificate. The certificate has the following ten fields.*

1. *The traditional target is the universal regular-Cauchy completion associated to* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$.
2. *The BEDC source is* BHist *equipped with* RealStreamPrefixClassifier*,* MetricDistanceWitness*, and the supplied real-completeness capstone data.*
3. *The carrier consists of completion candidates, namely regular Cauchy point-name streams modulo the pointwise* hsame *transport already used by the readback rows.*
4. *The classifier reads pointwise metric-distance tolerance through the real-prefix classifier at a selected prefix index.*
5. *Stability is supplied by the checked visible-tail, empty-distance, transport, symmetry, determinacy, and prefix-independent readback theorems listed in this chapter.*
6. *The ledger records the selected prefix index, the visible left and right contexts, the metric-distance witness, the Cauchy modulus, and the limit candidate.*
7. *The core theorem is this certificate boundary, which repacks the carrier, classifier, stability, ledger, and exactness clauses for* Completion$^{\uparrow}$.
8. *Exactness is finite-prefix distance readback together with limit uniqueness under the classifier.*
9. *The standard bridge slot is carried by the conservative completion bridge, so this certificate exposes the paper-side boundary consumed by* CompleteMetric$^{\uparrow}$ *and* CompactMetric$^{\uparrow}$ *without asserting that bridge here.*
10. *The certificate status is* ProofTarget.

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness semanticNameCert`

Proof. Endpoint coverage comes from the two direct real-prefix readback rows. Apply Definition 95.1 for visible E1 distance results, equivalently the checked `CompletionMetricDistanceWitness_e1_tail_re` row, and apply Definition 95.2 for empty results, equivalently the checked `CompletionMetricDistanceWitness_em` row. These two cases cover the finite-prefix distance readback used by the classifier.

Carrier and classifier transport are the two hsame transport rows. Use Definition 95.4 for ordinary visible tails and Definition 95.8 for contextualized tails. The empty contextual case is carried by Definition 95.10, so the selected prefix index and its visible contexts can be moved along pointwise representative equality.

Symmetry and uniqueness are read from the symmetric and deterministic rows. Apply Definition 95.5, Definition 95.6, Definition 95.11, and Definition 95.12 to compare all visible distance tails attached to the same finite-prefix classifier data.

Context-insensitive readback is supplied by the prefix-independent visible-context rows. Apply Definition 95.13 and Definition 95.14 to remove dependence on the chosen visible left and right contexts. The Cauchy modulus and limit candidate are then ledger fields, while limit uniqueness is the exactness condition read through the same classifier.

This is not a one-theorem variant of any checked row: its conclusion packages the source, carrier, classifier, stability obligations, ledger, exactness predicate, and downstream bridge slots into one Completion$^{\uparrow}$ certificate boundary. □

Theorem 95.16 (Completion carrier obligation)**.** *The* Completion$^{\uparrow}$ *carrier obligation is the BEDC finite-prefix regular-Cauchy point-name packet over* BHist, RealStreamPrefixClassifier, MetricDistanceWitness, *Cauchy modulus data, visible-context transport, and* Pkg *provenance. Its carrier row consists only of displayed point-name streams, selected prefix indices, finite distance-readback rows, visible left and right contexts, modulus rows, limit-candidate ledger rows, and package provenance.*

The obligation contains no quotient real, no host real equality, no choice-selected limit, and no representative selected outside the displayed finite-prefix packet.

Lean4 ✓ `BEDC.Derived.CompletionUp.CompletionCarrier obligation`

Proof. Read the carrier and ledger fields of Definition 95.15. The source is BHist equipped with the real-prefix classifier, metric-distance witness rows, Cauchy modulus data, visible contexts, and package provenance. These are exactly the finite coordinates named in the carrier statement.

The readback rows are supplied by Definition 95.1 and Definition 95.2. The visible-context coordinates are transported by Definition 95.4, Definition 95.8, and Definition 95.10.

The deterministic and prefix-independent rows of Definition 95.15 show that the carrier is exhausted by those finite-prefix coordinates. Since no cited row introduces a quotient type, host real equality, or choice-selected limit, none is part of the carrier obligation. □

Theorem 95.17 (Completion classifier stability obligation)**.** *The* Completion$^{\uparrow}$ *classifier is stable under displayed* hsame *and* Cont *transport of the selected prefix, visible left and right contexts, metric-distance witness, Cauchy modulus, and limit-candidate ledger. After transport, the classifier still reads only the transported real-prefix distance result, the transported visible-context rows, the transported modulus row, the transported limit-candidate row, and the transported* Pkg *provenance.*

The stability obligation introduces no quotient real equality, no host real equality, no completion point outside the packet, and no choice-selected representative.

Proof. Begin with the carrier rows named in Definition 95.16. A displayed hsame route moves the selected prefix and the visible contexts, while the displayed Cont routes preserve the finite readback boundary consumed by the classifier.

Apply Definition 95.4 to the ordinary visible tail and Definition 95.8 to the contextualized visible tail. For the empty contextual distance row, apply Definition 95.10.

The transported classifier remains deterministic by Definition 95.5 and Definition 95.12. Prefix-independent visible-context readback is supplied by Definition 95.13 and Definition 95.14.

Thus the transported classifier has the same finite-prefix inventory as the original packet. The proof never passes through a quotient type, host real equality, an external completion point, or a choice-selected representative. □

Theorem 95.18 (Completion ledger exactness obligation)**.** *Every public* Completion$^{\uparrow}$ *consumer read factors through the selected prefix index, real-prefix readback, metric-distance witness, visible-context transport, Cauchy modulus, limit-candidate row, and* Pkg *provenance displayed in the* Completion$^{\uparrow}$ *packet. The exactness ledger has no further public coordinate.*

In particular, a consumer cannot read a quotient real, host real equality, choice-selected limit, or representative outside the finite-prefix regular-Cauchy point-name packet.

Proof. Project a public consumer read through the ten-field boundary of Definition 95.15. The carrier field supplies the selected prefix and point-name packet, while the classifier field supplies the real-prefix distance readback through MetricDistanceWitness.

The visible-context rows are exactly the rows transported by Definition 95.8 and Definition 95.10. The ordinary and empty readback cases are exactly Definition 95.1 and Definition 95.2.

Use the determinacy and prefix-independent rows cited in Definition 95.15 to rule out any second public read path for the same displayed prefix data. The Cauchy modulus, limit-candidate row, and package provenance are ledger coordinates, so they are readable only as the displayed rows of that packet.

Therefore every public read factors through the listed finite-prefix coordinates. No step creates a quotient real, host real equality, choice-selected limit, or hidden representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{Completion}\uparrow}$.

Constructive story: A completion packet is generated by regular-Cauchy point-name streams, real-prefix readback rows, metric-distance witnesses, visible-context transport, prefix-independent determinacy, Cauchy modulus data, and limit-candidate ledger rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Completion}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 95.15, Definition 95.16, Definition 95.17, and Definition 95.18 export the concrete completion certificate, carrier obligation, classifier stability obligation, and ledger exactness obligation as the public finite-prefix completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Completion}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CompletionUp.CompletionMetricDistanceWitness\_semanticNameCert`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim completeness of arbitrary metric spaces, quotient reals, choice-selected limits, open-cover compactness, or standard bridge verification.

Upgrade path: No paper-side theory-axis upgrade remains; formal bridge checking is tracked only on the verification axis.

Chapter 96

A Concrete Naming Certificate for $\mathsf{Manifold}^{\uparrow}$

The manifold interface $\mathsf{Manifold}^{\uparrow}$ packages a topological carrier with a finite atlas of charts into $(\mathsf{Real}^{\uparrow})^n$ whose transition maps are constructively smooth. It depends on $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88) and the real-analytic certificate (chapter 74), and supplies the substrate for the bundle and de Rham certificates.

96.1 Carrier definition

Definition 96.1 (Charted manifold carrier)**.** *A charted* $\mathsf{Manifold}^{\uparrow}$ *carrier package over a* $\mathsf{Topology}^{\uparrow}$ *source is a tuple*

$$\mathcal{M} = (X, \mathcal{T}, A, \mathsf{Dom}, \mathsf{domIx}, \mathsf{chart}, \mathsf{coord}, \mathsf{Trans})$$

with the following displayed data. The point carrier is a predicate $X : \mathsf{BHist} \to \mathsf{Prop}$ *on BEDC histories, and* $\mathcal{T}$ *is an indexed open topology carrier for* X *in the sense of Definition 88.1. The atlas index family* A *is finite. For every* $a : A$*, the chart-domain predicate* $\mathsf{Dom}_a : X \to \mathsf{Prop}$ *is carried by a witnessed open* $\mathsf{domIx}(a) : \mathsf{OpenIx}$*:*

$$\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(a), \mathsf{Dom}_a).$$

The chart map chart_a *sends a point* x *with* $\mathsf{Dom}_a(x)$ *to a finite list of coordinate histories*

$$\mathsf{coord}_a(x) = (r_1, \ldots, r_{n(a)})$$

whose entries satisfy the $\mathsf{Real}^{\uparrow}$ *carrier and are classified by the* $\mathsf{Real}^{\uparrow}$ *classifier under replacement of* x *along the topology carrier classifier.*

For every overlap row $a, b : A$ *and every point* x *lying in* $\mathsf{Dom}_a \cap \mathsf{Dom}_b$*, the ledger includes a transition entry*

$$\mathsf{Trans}_{a,b}(x) : \mathsf{coord}_a(x) \longmapsto \mathsf{coord}_b(x)$$

whose source and target coordinate lists are finite $\mathsf{Real}^{\uparrow}$ *histories and whose local transport row is certified by the* $\mathsf{RealAnalytic}^{\uparrow}$ *boundary of Definition 74.29. Thus the manifold carrier does not add a new equality principle: point sameness is the topology carrier classifier, coordinate sameness is the* $\mathsf{Real}^{\uparrow}$ *classifier, and overlap smoothness is ledgered by supplied real-analytic rows.*

Lean4 (definition) `BEDC.Derived.ManifoldUp.ManifoldChartedCarrier`

Definition 96.2 (Chart-atlas manifold carrier)**.** *A chart-atlas manifold carrier for* $\mathsf{Manifold}^{\uparrow}$ *is a predicate* $\mathsf{ManAtlasPkg}(P)$ *on* BHist *packages* P *whose fields are the following rows.*

1. *The traditional target is a finite-atlas topological manifold.*
2. *The BEDC source is* $\mathsf{Topology}^{\uparrow}$ *together with* $\mathsf{Real}^{\uparrow}$ *coordinate histories and real-analytic transition rows from chapter 74.*
3. *The carrier row contains a topological carrier history* M *with its* $\mathsf{Topology}^{\uparrow}$ *certificate, a finite chart-index history* I*, chart-domain predicates* Dom_i *on* M*, chart maps* $\varphi_i : \mathsf{Dom}_i \to (\mathsf{Real}^{\uparrow})^{n_i}$*, and overlap transition ledgers* Trans_{ij} *for each chart pair* i, j*.*
4. *The classifier row is componentwise* hsame *on* M*, on* I*, on every chart-domain row, on every chart-map row, and on every transition ledger.*
5. *The stability row says that componentwise* hsame *transport of those fields carries* $\mathsf{ManAtlasPkg}$ *packages to* $\mathsf{ManAtlasPkg}$ *packages.*
6. *The ledger records topology rows, real-coordinate rows, overlap rows, and constructively smooth transition certificates for each nonempty chart overlap.*
7. *The standard bridge row is absent: this carrier names the BEDC obligation surface and does not assert an external manifold equivalence.*

Equivalently, $\mathsf{ManAtlasPkg}(P)$ *is the ten-field* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source object whose public carrier is the finite chart atlas over* M*, whose classifier is the displayed componentwise* hsame *classifier, and whose visible ledger is the topology-coordinate-overlap ledger just listed.*

Theorem 96.3 (ManifoldUp chart cover coverage)**.** *For every package* P *with* $\mathsf{ManAtlasPkg}(P)$ *and every carried point history* x *in the underlying* $\mathsf{Topology}^{\uparrow}$ *carrier, the visible atlas ledger exhibits a chart index* i *together with the displayed chart-domain row* $\mathsf{Dom}_i(x)$*.*

Proof. Unpack $\mathsf{ManAtlasPkg}(P)$ by Definition 96.2. The carrier row says that x is a BHist point carried by the same $\mathsf{Topology}^{\uparrow}$ carrier that supports every chart domain.

The visible atlas ledger is part of the charted source object. Its chart-cover row is indexed by the finite chart history and states that every carried point is displayed in at least one domain row.

Project the cover witness for x. The projection returns a chart index i and the corresponding membership evidence $\mathsf{Dom}_i(x)$, both living in the same $\mathsf{Topology}^{\uparrow}$ carrier row as x. □

Theorem 96.4 (ManifoldUp chart domain carrier transport)**.** *Let* P *be a package with* $\mathsf{ManAtlasPkg}(P)$*. If carried point histories* x *and* y *are related by the* $\mathsf{Topology}^{\uparrow}$ *carrier classifier and* $\mathsf{Dom}_i(x)$ *holds for a displayed chart-domain row, then* $\mathsf{Dom}_i(y)$ *holds after transporting the row along the classifier.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage chart domain carrier transport`

Proof. The chart-domain row Dom_i is an indexed-open witness supplied by the $\mathsf{Topology}^{\uparrow}$ part of Definition 96.2. Its membership predicate is therefore classified by the same carrier comparison as the underlying topological points.

Apply the carrier classifier to the evidence that x and y are the same carried point for the topology row. The indexed-open stability clause transports membership in Dom_i from x to y.

The transported domain row is still the displayed row for chart i, because the chart classifier is componentwise hsame on chart-domain histories. Thus the transported membership is exactly $\mathsf{Dom}_i(y)$. □

Theorem 96.5 (ManifoldUp atlas transition composition)**.** *Let P be a package with* $\mathsf{ManAtlasPkg}(P)$*. For chart indices i, j, k and a carried point history x lying in the three displayed domains, the overlap transition from i to k is witnessed by composing the supplied* $\mathsf{RealAnalytic}^{\uparrow}$ *transition ledgers from i to j and from j to k on the common overlap.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition composition scope`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition append readback`

Proof. Unpack the transition rows of Definition 96.2. The hypotheses $\mathsf{Dom}_i(x)$, $\mathsf{Dom}_j(x)$, and $\mathsf{Dom}_k(x)$ place x in both overlaps (i, j) and (j, k).

Project the two visible ledgers Trans_{ij} and Trans_{jk}. Each supplies a coordinate-change history between finite $\mathsf{Real}^{\uparrow}$ windows together with a $\mathsf{RealAnalytic}^{\uparrow}$ local transition certificate on its overlap.

Compose the coordinate histories at the shared j-coordinates. The $\mathsf{RealAnalytic}^{\uparrow}$ local transition row is closed under this displayed composition boundary, so the composed history carries a real-analytic certificate on the triple-overlap row.

Record the composed certificate as the transition witness from i to k on the common overlap. This witness is the atlas-transition composition required by the $\mathsf{Manifold}^{\uparrow}$ ledger. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition composition target`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition composition readback`

Theorem 96.6 (ManifoldUp chart-domain intersection open)**.** *Let P be a package with* $\mathsf{ManAtlasPkg}(P)$*. For chart indices i, j, the* $\mathsf{Topology}^{\uparrow}$ *index* $\mathsf{domIx}(i) \wedge_{\mathsf{Topology}^{\uparrow}} \mathsf{domIx}(j)$ *carries the overlap predicate*

$$x \longmapsto \mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x)$$

on the underlying topological carrier.

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage chart domain intersection open`

Proof. Unpack $\mathsf{ManAtlasPkg}(P)$ by Definition 96.2. The carrier row exposes the common $\mathsf{Topology}^{\uparrow}$ carrier and, for every chart index, a displayed open index $\mathsf{domIx}(-)$ carrying the corresponding chart-domain predicate.

Project the two rows $\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(i), \mathsf{Dom}_i)$ and $\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(j), \mathsf{Dom}_j)$. They live over the same carrier because both chart domains are fields of the same atlas package.

Apply Definition 88.7 to these two carried opens. The resulting finite-intersection index is $\mathsf{domIx}(i) \wedge_{\mathsf{Topology}^{\uparrow}} \mathsf{domIx}(j)$, and its pointwise readback is exactly the displayed conjunction of the two chart-domain memberships. □

Theorem 96.7 (ManifoldUp overlap transition source carried)**.** *Let P be a package with* $\mathsf{ManAtlasPkg}(P)$*. Every displayed transition row* Trans_{ij} *is sourced on the carried open overlap supplied by Definition 96.6 and exposes finite* $\mathsf{Real}^{\uparrow}$ *source and target coordinate rows* $\mathsf{coord}_i(x)$ *and* $\mathsf{coord}_j(x)$*.*

Proof. Fix the chart pair i, j and unpack the transition ledger of Definition 96.2. The ledger row for Trans_{ij} is indexed by the same chart-domain predicates Dom_i and Dom_j that appear in the carrier row.

Use Definition 96.6 to replace the informal pairwise overlap by the explicit carried-open index $\mathsf{domIx}(i) \wedge_{\mathsf{Topology}^{\uparrow}} \mathsf{domIx}(j)$. Thus the source predicate for the transition is not a hidden set-theoretic intersection; it is the $\mathsf{Topology}^{\uparrow}$ certificate carrying $\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x)$.

Project the coordinate fields from the same atlas package. The source list $\mathsf{coord}_i(x)$ and target list $\mathsf{coord}_j(x)$ are finite lists of $\mathsf{Real}^{\uparrow}$ histories, and the transition row records the real-analytic certificate between those two finite coordinate rows on the carried overlap. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition source readback`

Theorem 96.8 (ManifoldUp triple-overlap composition scope)**.** *Let P be a package with* $\mathsf{ManAtlasPkg}(P)$*. For chart indices i, j, k, the triple-overlap predicate*

$$x \longmapsto \mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x) \wedge \mathsf{Dom}_k(x)$$

is carried by iterating the $\mathsf{Topology}^{\uparrow}$ *finite-intersection index, and this carried triple overlap is the scoped source on which Definition 96.5 composes the two real-analytic transition ledgers.*

Proof. First build the pairwise overlap for i and j using Definition 96.6. This gives a $\mathsf{Topology}^{\uparrow}$ carried-open index for $\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x)$ over the package carrier.

Project the chart-domain row for k from Definition 96.2. Apply Definition 88.7 to the carried pairwise overlap and the carried open Dom_k. The resulting iterated index carries the displayed triple conjunction, with the parenthesizing read through the pointwise conjunction witness.

Now restrict the transition ledgers Trans_{ij} and Trans_{jk} to this carried triple overlap. The source and target coordinate rows are finite $\mathsf{Real}^{\uparrow}$ lists by Definition 96.7, and the shared j-coordinate row is exposed on the same carried source.

Only after this source has been carried, apply Definition 96.5. Its composition witness therefore lives on the explicit $\mathsf{Topology}^{\uparrow}$ triple-overlap certificate, not on an implicit common-overlap predicate. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldScopedBoundaryPackage reassociated source`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldScopedBoundaryPackage pair transition closure`

Definition 96.9 (Singleton empty-chart manifold package)**.** *Let the BEDC manifold carrier be the singleton history predicate*

$$C_{\mathsf{e}}(h) \quad \Longleftrightarrow \quad \mathsf{hsame}(h, \mathsf{e})$$

on BHist*. Its carrier classifier is* hsame *restricted to C_{e}: two carried histories classify when they are internally the same, and the carrier side conditions are transported along* hsame*.*

The topology surface is the inherited indexed $\mathsf{Topology}^{\uparrow}$ *boundary surface on this carrier. The distinguished top index carries C_{e} itself, and the distinguished bottom index carries the empty predicate. The membership ledger records the chosen history, its C_{e} witness, the classifier row, and the boundary open witness supplied by* $\mathsf{Topology}^{\uparrow}$*.*

The atlas has one chart χ_{e}. Its domain is the top open of the displayed topology surface. Its target is the constant zero endpoint of $\mathsf{Real}^{\uparrow}$*, read through the real-analytic local constant stream boundary. For every carried h, the chart value is the same constant endpoint, and the chart ledger records* BHist*,* $\mathsf{UnaryHistory}$*,* hsame*,* Cont*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{RealAnalytic}^{\uparrow}$ *constant-map boundary.*

Thus the singleton package supplies the following ten certificate rows:

1. *the traditional target is a one-point smooth manifold;*
2. *the BEDC source is* BHist *with* e *as the displayed point;*
3. *the carrier choice is C_{e};*
4. *the classifier choice is* hsame *restricted to C_{e};*
5. *the stability obligations are carrier transport, top-domain transport, bottom-domain exclusion, chart-value transport, and transition transport;*

6. *the ledger is the history, unary-history row, classifier row, continuation row, topology boundary row, real constant endpoint row, and real-analytic constant-map row;*
7. *the core theorem list is carrier topology scope, chart coverage, transition smoothness, and the assembled* $\mathsf{Manifold}^{\uparrow}$ *obligation package;*
8. *the exactness predicate says every chart-domain witness is exactly a* C_{e} *witness and every transition witness is the sole self-overlap row;*
9. *the standard bridge is absent;*
10. *the status is paper-side obligation target.*

96.2 Atlas Obligation Inventory

Definition 96.10 (Manifold atlas obligation inventory)**.** *The atlas obligation inventory for* $\mathsf{ManAtlasPkg}$ *is the ten-field* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *template whose rows are fixed as follows.*

1. *The traditional target is a finite-atlas manifold: a topological carrier equipped with finitely many chart domains, chart maps into finite* $(\mathsf{Real}^{\uparrow})$ *coordinate windows, and transition data on visible overlaps.*
2. *The BEDC source is* BHist *together with the already named* $\mathsf{Topology}^{\uparrow}$ *carrier surface, the* $\mathsf{Real}^{\uparrow}$ *coordinate rows, and the* $\mathsf{RealAnalytic}^{\uparrow}$ *local transition rows.*
3. *The carrier is the predicate* $\mathsf{ManAtlasPkg}(P)$ *of Definition 96.2 on package histories* P*.*
4. *The classifier is componentwise* hsame *on the topological carrier, the finite chart-index history, chart-domain rows, chart-map rows, coordinate histories, and transition ledgers.*
5. *The stability field consists of transport of the carrier, chart domains, chart maps, coordinate rows, and transition ledgers along those componentwise* hsame *witnesses.*
6. *The ledger records the* $\mathsf{Topology}^{\uparrow}$ *carried-open rows, finite intersection rows for pairwise and triple overlaps,* $\mathsf{Real}^{\uparrow}$ *coordinate rows, and* $\mathsf{RealAnalytic}^{\uparrow}$ *certificates for each displayed transition.*
7. *The core theorem list is Definition 96.3, Definition 96.4, Definition 96.6, Definition 96.7, Definition 96.8, Definition 96.5, Definition 96.18, Definition 96.21, Definition 96.22, and Definition 96.23.*
8. *Exactness is visible-atlas readback: every accepted carrier, classifier, domain, coordinate, overlap, transition, and coherence row is one of the displayed rows of* $\mathsf{ManAtlasPkg}$*.*
9. *The standard bridge row is absent.*
10. *The boundary package is the scoped surface of Definition 96.41, with field coverage supplied by Definition 96.42.*

Lean4 (definition) `BEDC.Derived.ManifoldUp.ManifoldAtlasObligationInventory`

Theorem 96.11 (Manifold atlas carrier-classifier obligation)**.** *For every atlas package history* P*, the carrier and classifier slots of the inventory in Definition 96.10 are fixed by Definition 96.2: the carrier slot is exactly* $\mathsf{ManAtlasPkg}(P)$*, and the classifier slot is componentwise* hsame *on the displayed carrier, chart-index, chart-domain, chart-map, coordinate, and transition-ledger rows.*

Proof. Unpack the inventory definition. Its carrier row names $\mathsf{ManAtlasPkg}(P)$ and points back to Definition 96.2, so there is no second carrier predicate to choose.

Unpack Definition 96.2. The classifier row there is componentwise hsame on the same topological carrier history, finite chart-index history, chart-domain rows, chart-map rows, and transition ledger rows that appear in the inventory.

The two projections therefore read the same BHist package fields in the same order. Carrier equality is definitional readback of ManAtlasPkg, and classifier equality is componentwise readback of the displayed hsame witnesses. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage classifier transport`

Theorem 96.12 (Manifold atlas chart-domain transition obligation)**.** *For every package P satisfying* $\mathsf{ManAtlasPkg}(P)$*, the inventory of Definition 96.10 supplies the chart-cover, domain-transport, pairwise-overlap, triple-overlap, and transition-ledger obligations from the existing* $\mathsf{Manifold}^{\uparrow}$ *theorem surface.*

Proof. The chart-cover row is Definition 96.3. It projects, for each carried point history in the $\mathsf{Topology}^{\uparrow}$ carrier, a finite chart index and a displayed chart-domain witness.

The domain-transport row is Definition 96.4. It transports membership in a displayed chart domain along the $\mathsf{Topology}^{\uparrow}$ carrier classifier, with the chart-domain row itself compared by componentwise hsame.

The pairwise overlap row is Definition 96.6, and the transition source row is Definition 96.7. Together they turn each displayed overlap into a carried $\mathsf{Topology}^{\uparrow}$ finite intersection with finite $\mathsf{Real}^{\uparrow}$ source and target coordinate rows.

The triple-overlap and transition-ledger rows are Definition 96.8, Definition 96.5, and Definition 96.18. These rows provide the iterated finite-intersection source, the $\mathsf{RealAnalytic}^{\uparrow}$ composition witness, and componentwise hsame stability for the visible transition ledger. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage transition closure`

Theorem 96.13 (Manifold atlas NameCert exactness obligation)**.** *Every package satisfying the inventory of Definition 96.10 and the scoped boundary of Definition 96.41 has visible-atlas exactness: its public* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *fields are exactly the carrier, classifier, stability, ledger, theorem-list, exactness, and absent-bridge rows listed in the inventory, with no additional chart, coordinate, overlap, or transition row outside the displayed* ManAtlasPkg *surface.*

Proof. First project the carrier and classifier by Definition 96.11. The public carrier is ManAtlasPkg, and the public classifier is the componentwise hsame row over the displayed atlas fields.

Next project the chart, domain, overlap, and transition obligations by Definition 96.12. These projections account for the chart cover, domain transport, pairwise overlap, triple overlap, transition composition, and transition-ledger stability rows.

Now apply Definition 96.41. Its exactness row accepts only the visible chart cover, domain, coordinate, overlap, transition, and coherence rows of ManAtlasPkg, and its standard bridge row is absent.

Finally apply Definition 96.42. The scoped field coverage theorem says that the visible fields are precisely the carrier, componentwise classifier, chart cover, domain transport, overlap sources, transition-ledger stability, and coherence rows already named above. Hence the ledger is exhausted by the inventory and no hidden atlas field remains. □

96.3 The certificate

Theorem 96.14 (Manifold chart endpoint coverage)**.** *For a charted carrier package $\mathcal{M}$ as in Definition 96.1, every displayed chart row exposes the following endpoint data:*

1. *the underlying* $\mathsf{Topology}^{\uparrow}$ *carrier row* $X(x)$ *and the classifier* same_X *supplied by* $\mathcal{T}$*;*
2. *the witnessed open row* $\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(a), \mathsf{Dom}_a)$ *for the chart domain;*
3. *the finite* $\mathsf{Real}^{\uparrow}$ *coordinate output* $\mathsf{coord}_a(x) = (r_1, \ldots, r_{n(a)})$*;*
4. *the visible atlas ledger entry recording* a*,* $\mathsf{domIx}(a)$*,* x*, and the coordinate list.*

Proof. Unfold the charted carrier package of Definition 96.1. The first fields are exactly the predicate X on BHist and the indexed open topology carrier $\mathcal{T}$, so the $\mathsf{Topology}^{\uparrow}$ carrier and classifier are obtained by projection.

For a fixed atlas index a, the same unfolding exposes the domain predicate Dom_a, the chosen open index $\mathsf{domIx}(a)$, and the carrying equation for that open. This is the chart-domain open witness required by the certificate.

Finally project the chart map field. Its output is already a finite list of $\mathsf{Real}^{\uparrow}$ coordinate histories, and the atlas index plus open index are retained in the package ledger. No quotient or hidden atlas representative is introduced, so the displayed tuple is the visible readback of the chart row. □

Theorem 96.15 (Manifold transition classifier stability)**.** *Let $\mathcal{M}$ be a charted carrier package. Suppose $x, y : \mathsf{BHist}$ are carried points with $\mathsf{same}_X(x, y)$, and suppose the atlas domains Dom_a and Dom_b carry x through their witnessed open indices. Then the overlap transition row $\mathsf{Trans}_{a,b}(x)$ transports to the corresponding row at y using:*

1. *the topology carrier classifier and membership-stability field of Definition 88.1;*
2. *the two carrying equations for* $\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(a), \mathsf{Dom}_a)$ *and* $\mathsf{CarriesOpen}_{\mathcal{T}}(\mathsf{domIx}(b), \mathsf{Dom}_b)$*;*
3. *the* $\mathsf{Real}^{\uparrow}$ *classifier on each coordinate of* $\mathsf{coord}_a(-)$ *and* $\mathsf{coord}_b(-)$*;*
4. *the supplied* $\mathsf{RealAnalytic}^{\uparrow}$ *local transport rows from Definition 74.29.*

Consequently the transported source and target coordinate lists represent the same overlap transition in the atlas ledger.

Proof. Begin with the topology side. The carrying equation for $\mathsf{domIx}(a)$ converts $\mathsf{Dom}_a(x)$ into $\mathsf{OpenAt}(\mathsf{domIx}(a), x)$. The membership-stability field of Definition 88.1, applied to $\mathsf{same}_X(x, y)$, transports this open membership to y, and the same carrying equation converts it back to $\mathsf{Dom}_a(y)$.

Repeat the identical argument for b. This supplies the transported overlap witness $\mathsf{Dom}_a(y) \wedge \mathsf{Dom}_b(y)$, so the transition ledger has the same atlas indices at the transported point.

Project the coordinate fields from Definition 96.1. Their stability clauses give coordinatewise $\mathsf{Real}^{\uparrow}$ classifier witnesses from $\mathsf{coord}_a(x)$ to $\mathsf{coord}_a(y)$ and from $\mathsf{coord}_b(x)$ to $\mathsf{coord}_b(y)$.

The transition row itself is one of the real-analytic rows supplied at the boundary of Definition 74.29. Apply that row to the transported overlap witness and the coordinatewise $\mathsf{Real}^{\uparrow}$ classifier witnesses. The result is the transition row at y, classified by the same visible atlas ledger entry. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton transition classifier stability`

Theorem 96.16 (Manifold charted NameCert surface)**.** *A charted carrier package $\mathcal{M}$ as in Definition 96.1, together with Definition 96.14 and Definition 96.15, assembles the paper-side* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *obligation surface. Its fields are the* BHist *point carrier X, the* $\mathsf{Topology}^\uparrow$ *classifier* same_X*, the finite atlas ledger, chart-domain open witnesses, finite* $\mathsf{Real}^\uparrow$ *coordinate outputs, overlap-transition stability rows, and an explicit statement that no standard bridge is claimed.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage semanticNameCert`

Proof. Unfold the NameCert template fields against Definition 96.1. The traditional target is a charted manifold, the BEDC source is BHist equipped with the topology carrier $\mathcal{T}$ and the finite atlas data, the carrier is X, and the classifier is same_X.

Endpoint coverage supplies the public readback rows: each chart exposes its carried point, witnessed open domain, finite coordinate output, and atlas ledger entry. These are exactly the carrier, source, and ledger fields needed by the certificate surface.

Transition classifier stability supplies the stability field. It transports overlap rows along the topology classifier, uses the two chart-domain carrying equations to keep the point inside both domains, and then invokes the $\mathsf{Real}^\uparrow$ and $\mathsf{RealAnalytic}^\uparrow$ rows for coordinate and smooth-transition transport.

The exactness policy is visible-ledger exactness: a manifold chart row is accepted only through the displayed atlas index, open witness, coordinate output, and transition row. The standard bridge field is absent, so the assembled surface is a BEDC paper-side NameCert obligation package rather than a bridge to an external manifold library. □

Theorem 96.17 (Manifold chart-transition classifier stability)**.** *Let P and Q be* BHist *packages. Suppose* $\mathsf{ManAtlasPkg}(P)$ *holds and the data of P and Q are related componentwise by* hsame*: the topological carrier histories are* hsame*, the finite chart-index histories are* hsame*, corresponding chart-domain rows are* hsame*, corresponding chart-map rows are* hsame*, and corresponding real-analytic transition ledgers are* hsame*. Then* $\mathsf{ManAtlasPkg}(Q)$ *holds.*

Proof. Unpack $\mathsf{ManAtlasPkg}(P)$ using Definition 96.2. The package supplies the $\mathsf{Topology}^\uparrow$ carrier row for M, the finite chart-index row, all chart-domain and chart-map rows, and the overlap transition ledgers.

Transport the topological carrier by the supplied hsame comparison of the carrier histories. Transport the finite chart-index row in the same way. For each chart index, transport the domain predicate and the chart-map row by the componentwise hsame hypotheses.

For each chart pair, transport the real-coordinate transition ledger and the constructively smooth transition certificate by the corresponding real-analytic hsame row. Repack the transported topology, chart, map, overlap, and smoothness rows. These are exactly the fields required by Definition 96.2 for $\mathsf{ManAtlasPkg}(Q)$. □

Theorem 96.18 (ManifoldUp transition ledger stability)**.** *Let P and Q be* BHist *packages. Suppose* $\mathsf{ManAtlasPkg}(P)$ *holds and the atlas rows of P and Q are related componentwise by* hsame*: the carrier histories, finite chart-index histories, indexed-open domain rows, coordinate histories, and real-analytic transition ledgers are all related. Then the transition ledger of P transports to a transition ledger for Q.*

Proof. Start with a displayed transition row of P for chart indices i, j. It contains the overlap membership boundary, the coordinate-change history, and the $\mathsf{RealAnalytic}^\uparrow$ local transition certificate.

Transport the two chart-domain rows by componentwise hsame. The $\mathsf{Topology}^{\uparrow}$ indexed-open membership stability clause transports the overlap predicate because it is the conjunction of the two transported domain memberships.

Transport the source and target coordinate histories by their componentwise hsame rows. The real-analytic transport boundary then carries the local transition certificate across the transported coordinate histories.

Repack the transported overlap, coordinate-change history, and real-analytic certificate. The result is the corresponding transition ledger row for Q. □

Theorem 96.19 (Manifold atlas overlap coverage)**.** *Every carried atlas package P with* $\mathsf{ManAtlasPkg}(P)$ *exposes, for each chart pair i, j in its finite chart-index history, a domain-overlap row*

$$\mathsf{Overlap}_{ij}(x) :\Longleftrightarrow \mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x)$$

on the topological carrier and the corresponding constructively smooth $\mathsf{Real}^{\uparrow}$ *transition ledger from φ_i-coordinates to φ_j-coordinates on that overlap.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage pairwise overlap readback`

Proof. Unpack the carried atlas package by Definition 96.2. The finite chart-index row gives the admissible chart pairs, while the chart-domain rows give the two predicates Dom_i and Dom_j over the same topological carrier.

For a fixed pair i, j, form the overlap row by conjunction of the two domain memberships. The topology ledger records the carrier support of both domains, so the overlap row is a predicate on the same topological carrier M.

The package ledger contains, for this same pair, the transition row Trans_{ij}. Its real-coordinate part records the source and target $\mathsf{Real}^{\uparrow}$ coordinate histories, and its real-analytic part records the constructively smooth transition certificate on the displayed overlap. Project these two ledger components to obtain the claimed coverage row. □

Definition 96.20 (Manifold transition coherence ledger)**.** *For a carried atlas package P with* $\mathsf{ManAtlasPkg}(P)$*, the transition coherence ledger is the sub-ledger of* $\mathsf{ManAtlasPkg}(P)$ *whose rows are indexed by one, two, and three chart names in the finite chart-index history. It contains the following displayed entries.*

1. *For every chart i, the row* $\mathsf{IdTrans}_i$ *is carried on the self-overlap* $\mathsf{Overlap}_{ii}$ *supplied by* $\mathsf{Topology}^{\uparrow}$ *and classifies the transition* Trans_{ii} *by the identity coordinate history row on φ_i.*
2. *For every chart pair i, j, the row* $\mathsf{InvTrans}_{ij}$ *is carried on the overlap transported between* $\mathsf{Overlap}_{ij}$ *and* $\mathsf{Overlap}_{ji}$ *by the* $\mathsf{Topology}^{\uparrow}$ *overlap rows, and classifies the two composites* $\mathsf{Trans}_{ji} \circ \mathsf{Trans}_{ij}$ *and* $\mathsf{Trans}_{ij} \circ \mathsf{Trans}_{ji}$ *by the corresponding* IdTrans *rows.*
3. *For every chart triple i, j, k, the row* $\mathsf{Cocycle}_{ijk}$ *is carried on the triple-overlap predicate*

$$\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x) \wedge \mathsf{Dom}_k(x),$$

and classifies $\mathsf{Trans}_{jk} \circ \mathsf{Trans}_{ij}$ *and* Trans_{ik} *by the same real-analytic transition row from φ_i-coordinates to φ_k-coordinates.*

Each entry is a visible ledger row over $\mathsf{Topology}^{\uparrow}$ *carried-overlap transport,* $\mathsf{Real}^{\uparrow}$ *coordinate histories, and the real-analytic classifier rows already exposed by the atlas package.*

Lean4 (definition) `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger`

Theorem 96.21 (Manifold overlap identity transition)**.** *Every carried atlas package P with* $\mathsf{ManAtlasPkg}(P)$ *exposes, for each chart i, a self-overlap transition row* Trans_{ii} *on* $\mathsf{Overlap}_{ii}$*. This row is classified by* $\mathsf{IdTrans}_i$ *in the transition coherence ledger, so its coordinate action is the identity real-coordinate row on φ_i over the carried self-overlap.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger identity transition readback`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton overlap identity transition`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger identity transition empty`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger identity transition`

Proof. Unpack $\mathsf{ManAtlasPkg}(P)$ by Definition 96.2. The finite chart-index row admits the pair (i, i), and Definition 96.19 identifies the carried domain as $\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_i(x)$ on the underlying $\mathsf{Topology}^{\uparrow}$ carrier.

Project the self row from Definition 96.20. Its source and target coordinate histories are both the displayed list $\varphi_i(x)$, transported through the same carried-overlap row.

The real-analytic classifier component of $\mathsf{IdTrans}_i$ is the identity coordinate row. Repacking that classifier with the self-overlap transport gives the displayed Trans_{ii} transition classified as the identity on φ_i. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger identity transition boundary`

Theorem 96.22 (Manifold overlap inverse transition)**.** *Every carried atlas package P with* $\mathsf{ManAtlasPkg}(P)$ *exposes, for each chart pair i, j, inverse transition rows on the carried overlap. The composite* $\mathsf{Trans}_{ji} \circ \mathsf{Trans}_{ij}$ *is classified by* $\mathsf{IdTrans}_i$ *on the i-coordinate overlap, and the composite* $\mathsf{Trans}_{ij} \circ \mathsf{Trans}_{ji}$ *is classified by* $\mathsf{IdTrans}_j$ *on the j-coordinate overlap.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton transition inverse boundary`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton overlap inverse transition`

Proof. Unpack the two pairwise overlap rows using Definition 96.19. The row for (i, j) carries $\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x)$, while the row for (j, i) carries the same two domain memberships in the reversed order.

Use the $\mathsf{Topology}^{\uparrow}$ carried-overlap transport to identify those two overlap readbacks. This gives a common domain on which both transition ledgers can be composed without changing the carried point or the chart-domain witnesses.

Project $\mathsf{InvTrans}_{ij}$ from Definition 96.20. Its real-analytic classifier rows state exactly that the two displayed composites classify to the relevant identity rows. Together with Definition 96.21, these projected rows give the two claimed inverse identities on the carried overlap. □

Theorem 96.23 (Manifold triple-overlap cocycle)**.** *Every carried atlas package P with* $\mathsf{ManAtlasPkg}(P)$ *exposes, for each chart triple i, j, k, a cocycle row on the carried triple overlap. On that overlap,* $\mathsf{Trans}_{jk} \circ \mathsf{Trans}_{ij}$ *and* Trans_{ik} *are classified by the same real-analytic transition row from φ_i-coordinates to φ_k-coordinates.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton coherence rows empty`

Proof. Start from the three pairwise overlap rows supplied by Definition 96.19. Their $\mathsf{Topology}^{\uparrow}$ domain components restrict to the same carried triple-overlap predicate $\mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x) \wedge \mathsf{Dom}_k(x)$.

Transport the pairwise transition ledgers to this common triple-overlap domain. The source of Trans_{ij} is the φ_i coordinate row, its target is the φ_j row, and the source of Trans_{jk} is the same φ_j row after $\mathsf{Real}^{\uparrow}$ coordinate-history transport.

Project $\mathsf{Cocycle}_{ijk}$ from Definition 96.20. Its real-analytic classifier component identifies the composite $\mathsf{Trans}_{jk} \circ \mathsf{Trans}_{ij}$ with the direct transition Trans_{ik} as the same transition row from φ_i to φ_k. Repacking that classifier with the common carried-overlap domain gives the displayed cocycle obligation. □

Theorem 96.24 (ManifoldUp transition-coherence exhaustion)**.** *For every carried atlas package P with $\mathsf{ManAtlasPkg}(P)$, every row of the transition coherence ledger of Definition 96.20 is one of the displayed one-chart identity rows, two-chart inverse-composite rows, or three-chart cocycle rows. In each case its source is a carried $\mathsf{Topology}^{\uparrow}$ overlap or triple-overlap, its endpoints are finite $\mathsf{Real}^{\uparrow}$ coordinate histories, and its classifier is the real-analytic transition row already exposed by the atlas ledger.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger exhaustion`

Proof. Unpack the transition coherence ledger of Definition 96.20. Its indexing data is the finite chart-index history carried by $\mathsf{ManAtlasPkg}(P)$, so each ledger row has one chart name, an ordered chart pair, or an ordered chart triple as its visible index.

In the one-index case, the row is $\mathsf{IdTrans}_i$. Apply Definition 96.21. The source is the self-overlap $\mathsf{Overlap}_{ii}$ carried by $\mathsf{Topology}^{\uparrow}$, the two endpoints are the same finite $\mathsf{Real}^{\uparrow}$ coordinate history φ_i, and the classifier is the identity real-analytic transition row exposed in the atlas ledger.

In the two-index case, the row is $\mathsf{InvTrans}_{ij}$. Apply Definition 96.22. The source is the carried overlap transported between $\mathsf{Overlap}_{ij}$ and $\mathsf{Overlap}_{ji}$, the endpoints are the finite coordinate histories φ_i and φ_j, and the two composites classify through the identity rows already exposed by the atlas transition ledger.

In the three-index case, the row is $\mathsf{Cocycle}_{ijk}$. Apply Definition 96.23. The source is the carried triple-overlap predicate, the endpoints are the finite coordinate histories φ_i, φ_j, and φ_k, and the composite transition and direct transition are classified by the same real-analytic row from φ_i to φ_k.

These three cases exhaust the constructors displayed in Definition 96.20. Repack the resulting carrier, endpoint, and classifier witnesses through Definition 96.25 to obtain the advertised $\mathsf{Manifold}^{\uparrow}$ ledger boundary. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldScopedBoundaryPackage transition coherence rows`

Theorem 96.25 (Manifold name-certificate obligation surface)**.** *The carrier definition of Definition 96.2, together with Definition 96.17 and Definition 96.19, Definition 96.21, Definition 96.22, and Definition 96.23, assembles the paper-side $\mathsf{Manifold}^{\uparrow}$ obligation surface. The surface consists of a topological carrier, a finite chart atlas, chart classifier stability, overlap coverage, transition-smoothness rows, identity transitions, inverse transition composites, triple-overlap cocycles, and no standard bridge row.*

Proof. The carrier definition supplies the named source object: $\mathsf{Topology}^{\uparrow}$ rows for the base carrier, $\mathsf{Real}^{\uparrow}$ coordinate rows for chart targets, finite chart-index data, chart domains, chart maps, and transition ledgers.

Definition 96.17 supplies the classifier-stability obligation for the whole package, because every public field is transported by the componentwise hsame classifier specified in the carrier definition.

Definition 96.19 supplies the overlap and transition-smoothness obligations. The absence of a standard bridge is already part of the carrier definition.

It remains to account for atlas compatibility. The coherence ledger of Definition 96.20 records the self, reversed-pair, and triple-overlap rows. The identity row is exposed by Definition 96.21; the inverse composite rows are exposed by Definition 96.22; and the triple-overlap cocycle row is exposed by Definition 96.23.

These rows use the same $\mathsf{Topology}^{\uparrow}$ carried-overlap transport, the same $\mathsf{Real}^{\uparrow}$ coordinate histories, and the same real-analytic classifier ledger as the pairwise transition rows. Repacking them with the carrier, classifier, and stability fields gives exactly the $\mathsf{Manifold}^{\uparrow}$ paper obligation surface listed in the theorem statement. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasSurfaceClassifier endpoints`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldAtlasPackage semantic name certificate`

Theorem 96.26 (Manifold name-certificate obligation threshold)**.** *The carrier definition of Definition 96.2, together with Definition 96.3, Definition 96.4, Definition 96.5, and Definition 96.18, assembles the paper-side* $\mathsf{Manifold}^{\uparrow}$ $\mathsf{NameCert}$ *obligation threshold. The threshold consists of the carrier, classifier, chart cover, chart-domain transport, transition composition, transition-ledger stability, and the visible atlas ledger.*

Proof. The carrier definition supplies the named source object: $\mathsf{Topology}^{\uparrow}$ rows for the base carrier, $\mathsf{Real}^{\uparrow}$ coordinate rows for chart targets, finite chart-index data, indexed-open chart domains, coordinate histories, and transition ledgers.

Definition 96.3 supplies the cover obligation by projecting, for every carried point, an index and domain membership from the visible atlas ledger.

Definition 96.4 supplies the classifier boundary for chart-domain membership. The same componentwise hsame classifier transports the carrier, domains, coordinate histories, and ledger rows named in the definition.

Definition 96.5 supplies the atlas composition obligation, and Definition 96.18 supplies stability of the visible transition ledger under transported chart rows. The absence of a standard bridge is already part of the carrier definition, so the assembled object is exactly the $\mathsf{Manifold}^{\uparrow}$ $\mathsf{NameCert}$ obligation threshold listed in the theorem statement. □

96.4 Chart Coordinate Transport

Definition 96.27 (ManifoldUp chart-coordinate transport row)**.** *Let P be a* BHist *package with* $\mathsf{ManAtlasPkg}(P)$*, let i be a chart index in the finite atlas history, and let x, y be carried points of the underlying* $\mathsf{Topology}^{\uparrow}$ *carrier. A chart-coordinate transport row for P, i, x, y consists of the following displayed data.*

1. *A chart-domain witness* $\mathsf{Dom}_i(x)$ *and a* $\mathsf{Topology}^{\uparrow}$ *classifier witness carrying x to y.*
2. *The transported chart-domain witness* $\mathsf{Dom}_i(y)$ *in the same indexed-open row.*
3. *A finite* $\mathsf{Real}^{\uparrow}$ *coordinate history $\varphi_i(x) = (r_1, \ldots, r_{n_i})$ exposed by the chart map at x.*
4. *A finite* $\mathsf{Real}^{\uparrow}$ *coordinate history $\varphi_i(y) = (s_1, \ldots, s_{n_i})$ exposed by the same chart map at y.*
5. *For each coordinate slot a in the finite chart output, a* $\mathsf{Real}^{\uparrow}$ *classifier witness relating r_a and s_a.*

Equivalently, the row is the visible coordinate-transport projection of $\mathsf{ManAtlasPkg}(P)$ *along a* $\mathsf{Topology}^{\uparrow}$ *carrier comparison: it records the chart index, the transported chart-domain membership, the two finite coordinate outputs, and the coordinatewise real classifiers.*

Lean4 (definition) `BEDC.Derived.ManifoldUp.ManifoldChartCoordinateTransportRow`

Theorem 96.28 (ManifoldUp chart-coordinate carrier transport)**.** *Let* P *be a* BHist *package with* $\mathsf{ManAtlasPkg}(P)$*. If* i *is a displayed chart index,* x *and* y *are carried points related by the* $\mathsf{Topology}^{\uparrow}$ *carrier classifier, and* $\mathsf{Dom}_i(x)$ *holds, then there exists a chart-coordinate transport row for* P, i, x, y*.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton chart coordinate carrier transport`

Proof. Unpack $\mathsf{ManAtlasPkg}(P)$ by Definition 96.2. The atlas carrier supplies the finite chart-index history, the indexed-open chart-domain row Dom_i, the chart map φ_i, and the finite $\mathsf{Real}^{\uparrow}$ coordinate histories read by that chart map on its domain.

Apply Definition 96.4 to the given $\mathsf{Topology}^{\uparrow}$ classifier witness between x and y. This transports the domain evidence $\mathsf{Dom}_i(x)$ to the displayed membership $\mathsf{Dom}_i(y)$ without changing the chart index or leaving the inherited topological carrier.

Evaluate the same chart-map row at the two displayed domain witnesses. The atlas classifier row is componentwise on chart maps and real-coordinate histories, so each coordinate in $\varphi_i(x)$ is related to the corresponding coordinate in $\varphi_i(y)$ by the $\mathsf{Real}^{\uparrow}$ classifier. These projections are exactly the fields of Definition 96.27. □

Theorem 96.29 (ManifoldUp chart-coordinate classifier determinacy)**.** *Let* P *be a* BHist *package with* $\mathsf{ManAtlasPkg}(P)$*, let* i *be a displayed chart index, and let* x, y *be carried points related by the* $\mathsf{Topology}^{\uparrow}$ *carrier classifier. Any two chart-coordinate transport rows for the same* P, i, x, y *have coordinate outputs that are coordinatewise classified by the* $\mathsf{Real}^{\uparrow}$ *classifier at both endpoints, and their endpoint comparison witnesses classify the same finite coordinate pairs.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldChartCoordinateTransportRow coordinate determinacy`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton chart coordinate classifier determinacy`

Proof. Both rows project from the same atlas package and the same chart index. By Definition 96.2, the chart-domain row and chart-map row are components of the visible $\mathsf{ManAtlasPkg}$ ledger, not separate set-level choices.

At the source endpoint, the two rows evaluate the same chart map on the same displayed domain membership $\mathsf{Dom}_i(x)$. The componentwise classifier for chart-map histories therefore relates their finite coordinate outputs by the $\mathsf{Real}^{\uparrow}$ classifier in every coordinate slot.

At the transported endpoint, both rows use Definition 96.4 with the same $\mathsf{Topology}^{\uparrow}$ endpoint comparison to obtain $\mathsf{Dom}_i(y)$. Evaluating the same chart-map row at that transported membership gives finite coordinate outputs that are again coordinatewise classified by $\mathsf{Real}^{\uparrow}$. The comparison witnesses in the two rows are precisely the coordinate projections of these two endpoint classifications. □

Theorem 96.30 (ManifoldUp chart-coordinate ledger exactness)**.** *For every package* P *with* $\mathsf{ManAtlasPkg}(P)$*, the visible coordinate transport ledger of* P *consists exactly of the chart index, the transported chart-domain row, the finite coordinate output at each endpoint, and the coordinatewise* $\mathsf{Real}^{\uparrow}$ *classifier witnesses. It contains no unrecorded set-level chart equality or extra chart-identification datum.*

Proof. The forward reading is by projection from Definition 96.2. A coordinate-transport row can only use the finite chart-index history, the indexed-open chart-domain rows, the chart-map coordinate histories, and the componentwise classifier rows that the atlas package exposes.

The transported domain component is exactly the output of Definition 96.4. The coordinate components are exactly the two finite $\mathsf{Real}^{\uparrow}$ histories obtained by evaluating the displayed chart map at the source and transported endpoint. The comparison components are exactly the coordinate projections of the $\mathsf{Real}^{\uparrow}$ classifier.

Conversely, any tuple with those displayed fields is a chart-coordinate transport row by Definition 96.27. Since the definition lists no chart-equality field and the atlas carrier classifier is componentwise on the recorded chart rows, no hidden set-level chart equality is available in the ledger. □

Theorem 96.31 (Singleton manifold carrier topology scope)**.** *For histories $h, k : \mathsf{BHist}$, if $C_{\mathsf{e}}(h)$ and $\mathsf{hsame}(h, k)$ hold, then $C_{\mathsf{e}}(k)$ holds. Moreover the inherited indexed $\mathsf{Topology}^{\uparrow}$ boundary surface has the exact scope:*

$$\mathsf{OpenAt}(\top_{\mathsf{Topology}^{\uparrow}}, h) \iff C_{\mathsf{e}}(h)$$

and

$$\mathsf{OpenAt}(\bot_{\mathsf{Topology}^{\uparrow}}, h) \iff \bot.$$

Consequently the top open transports along the restricted hsame *classifier, and the bottom open contributes no carried point.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingletonCarrier topology scope`

Proof. Unfold C_{e} from Definition 96.9. A witness $C_{\mathsf{e}}(h)$ is exactly $\mathsf{hsame}(h, \mathsf{e})$. Compose it with the supplied $\mathsf{hsame}(h, k)$ using symmetry and transitivity of hsame, with orientation chosen so that the endpoint is $\mathsf{hsame}(k, \mathsf{e})$. This is the carrier witness $C_{\mathsf{e}}(k)$.

For the topology scope, use the indexed boundary open witnesses of Definition 88.31 and the boundary closure of Definition 88.32. The top index carries the whole displayed carrier predicate, which in this instance is C_{e}, while the bottom index carries the empty predicate.

The transport statement is then the same membership-stability field from Definition 88.1, specialized to the top boundary index. The bottom clause has no point to transport because its carried predicate is $\bot$. □

Theorem 96.32 (Singleton manifold chart coverage)**.** *The one-chart atlas $\{\chi_{\mathsf{e}}\}$ of Definition 96.9 covers exactly the carried histories:*

$$h \in \mathsf{dom}(\chi_{\mathsf{e}}) \iff C_{\mathsf{e}}(h).$$

For every such h, the chart exposes the same constant zero endpoint of $\mathsf{Real}^{\uparrow}$:

$$\chi_{\mathsf{e}}(h) = 0_{\mathsf{Real}^{\uparrow}}.$$

The chart-domain ledger is therefore the top-open witness together with the C_{e} witness, and the chart-value ledger is the real constant endpoint row.

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton chart coverage`

Proof. By definition of the atlas in Definition 96.9, the domain of χ_{e} is the distinguished top open of the inherited indexed topology surface. Apply Definition 96.31 to identify this membership with $C_{\mathsf{e}}(h)$.

For the value statement, unfold the chart map. It is the constant map whose target is the displayed zero endpoint of $\mathsf{Real}^{\uparrow}$. No property of h is used except membership in the top domain, so the same endpoint is exposed for every carried history.

The ledger statement is just the pairing of these two witnesses. Domain coverage records the top-open row and its C_{e} readback; value coverage records the real endpoint row supplied by the local constant stream boundary. □

Theorem 96.33 (Singleton manifold transition smoothness)**.** *Every nonempty chart overlap in the singleton atlas is the self-overlap of χ_{e}. On that overlap the transition map is simultaneously the identity transition on the displayed singleton domain and the constant $0_{\mathsf{Real}^{\uparrow}}$ transition on the target endpoint. This transition is accepted by the real-analytic local constant stream boundary.*

Proof. There is only one chart in the atlas, so any overlap whose two chart names are present must be the pair $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$. The domain side is the top open, hence Definition 96.32 identifies the overlap points exactly with C_{e}.

On the carrier side, two points in the overlap are classified by the restricted hsame relation. Since every carried point is hsame to e, transitivity identifies all visible overlap histories with the same singleton point.

On the target side, both charts expose $0_{\mathsf{Real}^{\uparrow}}$. The transition therefore has no varying real argument: it is the identity when read on the singleton domain and the constant zero map when read on the real endpoint. The $\mathsf{RealAnalytic}^{\uparrow}$ local constant stream boundary accepts exactly this constant map, so the overlap transition supplies the required smoothness row. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton transition smoothness`

Theorem 96.34 (Singleton manifold chart-index exhaustion)**.** *Every chart index in the singleton atlas of Definition 96.9 is the displayed chart χ_{e}. Consequently every chart-domain row, chart-value row, and overlap row in that atlas factors through χ_{e} as its source chart and through χ_{e} as its target chart whenever a target chart is displayed.*

Proof. Unfold the atlas row of Definition 96.9. Its chart-index field is the one-element displayed atlas whose sole constructor is the row named χ_{e}.

Project any visible chart-index row from this finite atlas. Since the atlas has no second constructor, the projected row is definitionally the displayed χ_{e} row.

The chart-domain and chart-value ledgers are indexed by this same chart row, so their projections factor through χ_{e}. An overlap row is indexed by a pair of visible chart rows; applying the same exhaustion to both projections gives the pair $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$. □

Theorem 96.35 (Singleton manifold domain exactness)**.** *Every displayed chart-domain witness in the singleton atlas of Definition 96.9 is the top-open witness. For every $h : \mathsf{BHist}$, membership in that displayed domain is equivalent to $C_{\mathsf{e}}(h)$, while the bottom-open row carries no point.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton domain exactness`

Proof. Apply Definition 96.34 to reduce an arbitrary displayed chart-domain row to the domain row of χ_{e}. By construction this domain is the top open of the inherited $\mathsf{Topology}^{\uparrow}$ surface.

Definition 96.31 identifies top-open membership with $C_{\mathsf{e}}(h)$ for every history h and identifies the bottom open with $\bot$. Thus the top domain is exactly the carried singleton predicate, and the bottom row contributes no carried point.

Definition 96.32 reads the same top-open witness as the chart-domain ledger witness for χ_{e}. Since no other chart index exists, every displayed chart-domain witness has this exact readback. □

Theorem 96.36 (Singleton manifold transition-ledger exhaustion)**.** *Every displayed transition ledger entry in the singleton atlas of Definition 96.9 factors through the sole self-overlap of χ_{e}. Its source and target are the constant $0_{\mathsf{Real}^{\uparrow}}$ endpoint, and the entry is accepted exactly by the real-analytic local constant stream boundary.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton transition ledger exhaustion`
Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldTransitionCoherenceLedger singleton exhaustion`

Proof. Use Definition 96.34 on the two chart indices of a displayed transition ledger entry. Both indices are χ_{e}, so the entry factors through the sole self-overlap.

By Definition 96.35, the overlap domain is the top-open row read exactly as C_{e}. There is no bottom-open contribution and no second domain row on which another transition could be displayed.

Apply Definition 96.33. On this self-overlap both chart values are the constant endpoint $0_{\mathsf{Real}^{\uparrow}}$, and the only accepted smoothness row is the $\mathsf{RealAnalytic}^{\uparrow}$ local constant stream boundary. This is exactly the displayed transition ledger entry. □

Theorem 96.37 (Singleton manifold NameCert obligation package)**.** *The rows of Definition 96.9, Definition 96.31, Definition 96.32, Definition 96.33, Definition 96.34, Definition 96.35, and Definition 96.36 assemble the paper-side $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ obligation package for the singleton empty-chart construction.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton semanticNameCert`

Proof. The traditional target row is the one-point smooth manifold. The BEDC source row is BHist with the displayed empty history, and the carrier row is the predicate C_{e}. The classifier row is hsame restricted to that carrier.

The stability rows are supplied by the carrier transport in Definition 96.31, the top and bottom boundary rows inherited from $\mathsf{Topology}^{\uparrow}$, the chart coverage statement of Definition 96.32, and the transition statement of Definition 96.33.

Chart-index exhaustion supplies the finite atlas readback: every chart-domain, chart-value, and overlap row is indexed by χ_{e}. Domain exactness identifies the displayed chart domain with the top-open C_{e} witness and keeps the bottom-open row empty. Transition-ledger exhaustion identifies the only transition row as the self-overlap constant transition.

The ledger records BHist, $\mathsf{UnaryHistory}$, hsame, Cont, the indexed $\mathsf{Topology}^{\uparrow}$ boundary rows, the $\mathsf{Real}^{\uparrow}$ constant endpoint, and the $\mathsf{RealAnalytic}^{\uparrow}$ constant-map boundary. These are precisely the rows named in the construction package, with no hidden atlas component.

The core theorem list is the displayed carrier, coverage, smoothness, exactness, and ledger-exhaustion rows of this section. The standard bridge row is absent, and the resulting status is a paper-side obligation target for the Lean declaration BEDC.Derived.ManifoldUp.SingletonEmptyChart_name_certific

□

Definition 96.38 (Singleton manifold atlas ledger)**.** *The singleton manifold atlas ledger is the* BHist *package determined by the carrier C_{e} of Definition 96.9, the single chart index χ_{e}, the top-open chart domain supplied by* $\mathsf{Topology}^{\uparrow}$*, the constant $0_{\mathsf{Real}^{\uparrow}}$ endpoint supplied by* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{RealAnalytic}^{\uparrow}$ *local constant transition row. Its ten-field template is:*

Traditional target. *The one-point smooth manifold.*

BEDC source. BHist *with distinguished history* e*.*

Carrier. *The predicate $C_{\mathsf{e}}(h)$, equivalently* $\mathsf{hsame}(h, \mathsf{e})$*.*

Classifier. *The relation* hsame *restricted to C_{e}.*

Stability obligations. *Carrier transport, top-open membership transport, chart-domain read-back, chart-value constancy, and transition-row constancy.*

Ledger. *The rows* BHist*,* UnaryHistory*,* hsame*,* Cont*, the indexed* $\mathsf{Topology}^{\uparrow}$ *boundary rows, the* $\mathsf{Real}^{\uparrow}$ *zero endpoint, and the* $\mathsf{RealAnalytic}^{\uparrow}$ *constant-map row.*

Core theorem list. *Definition 96.31, Definition 96.32, Definition 96.33, and Definition 96.37.*

Exactness. *The only accepted domain witness is the top-open C_{e} witness, and the only accepted overlap transition is the self-overlap constant transition.*

No standard bridge. *The ledger contains no* StandardBridge *row.*

Status. *A paper-side* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *obligation target for later formalization under* BEDC.Derived.ManifoldUp*.*

Theorem 96.39 (Singleton manifold atlas index exhaustion)**.** *In the singleton manifold atlas ledger of Definition 96.38, every visible chart-index row is the sole index χ_{e}. Every nonempty overlap row is the pair $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$, and every accepted chart-domain witness is exactly the top-open C_{e} witness.*

Proof. Unfold the atlas ledger of Definition 96.38 and the construction package of Definition 96.9. The chart-index field is the displayed singleton list, so any visible chart row has the unique name χ_{e}.

With only one chart name available, a nonempty overlap can have no index pair other than $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$. The overlap is nonempty exactly when its domain side admits a carried history, and the only domain side exposed by the construction is the inherited top open.

Apply Definition 96.31 to identify that top-open membership with C_{e}. Then apply Definition 96.32 to read the same witness as the chart-domain ledger row, proving that no second chart-domain witness is present. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton atlas index exhaustion`

Theorem 96.40 (Singleton manifold chart-value transport)**.** *Let h, k : BHist. If $C_{\mathsf{e}}(h)$, $C_{\mathsf{e}}(k)$, and $\mathsf{hsame}(h, k)$ hold, then the chart-value ledger transports the value $\chi_{\mathsf{e}}(h) = 0_{\mathsf{Real}^{\uparrow}}$ to the value $\chi_{\mathsf{e}}(k) = 0_{\mathsf{Real}^{\uparrow}}$. Under the same transport, the transition ledger remains the* $\mathsf{RealAnalytic}^{\uparrow}$ *constant-map row for the self-overlap $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton chart value transport`

Proof. Use the carrier witnesses to read both h and k as histories in C_{e}. The relation $\mathsf{hsame}(h, k)$ supplies the classifier transport between them, and symmetry and transitivity of hsame keep the transport inside the singleton carrier.

Apply Definition 96.32 to the carried history h. The chart value is the constant endpoint $0_{\mathsf{Real}^{\uparrow}}$. Apply the same coverage theorem to the carried history k; the target endpoint is again $0_{\mathsf{Real}^{\uparrow}}$, so the chart-value ledger has transported the displayed value without changing the real row.

Finally apply Definition 96.33. Since the atlas has only the self-overlap $(\chi_{\mathsf{e}}, \chi_{\mathsf{e}})$ and both chart readings are the same constant endpoint, the transition ledger is still the $\mathsf{RealAnalytic}^{\uparrow}$ local constant map row. □

96.5 Scoped Manifold Boundary

Definition 96.41 (Manifold scoped boundary package)**.** *A scoped* $\mathsf{Manifold}^{\uparrow}$ *boundary package is a ten-field* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *template whose object is the chart-atlas carrier of Definition 96.2 with the following displayed rows.*

1. *The traditional target is a finite-atlas smooth manifold with visible chart domains, coordinate maps, and transition rows.*
2. *The BEDC source is* BHist *equipped with the previously named* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealAnalytic}^{\uparrow}$ *certificate surfaces.*
3. *The carrier is the predicate* $\mathsf{ManAtlasPkg}(P)$ *on package histories* P.
4. *The classifier is componentwise* hsame *on the topological carrier, finite chart-index history, chart-domain rows, chart-map rows, coordinate histories, and transition ledgers.*
5. *The stability field is chart-domain transport along the* $\mathsf{Topology}^{\uparrow}$ *carrier classifier, together with transition-ledger stability under componentwise* hsame.
6. *The coherence field contains the identity, inverse, and cocycle rows of Definition 96.20.*
7. *The ledger contains topology rows, real-coordinate rows, pairwise overlap rows, and finite-intersection triple-overlap rows.*
8. *The core theorem list is Definition 96.3, Definition 96.4, Definition 96.7, Definition 96.8, Definition 96.18, Definition 96.21, Definition 96.22, Definition 96.23, and Definition 96.37.*
9. *Exactness is visible-atlas readback: accepted chart cover, domain, coordinate, overlap, and transition rows are exactly the displayed rows of* $\mathsf{ManAtlasPkg}$, *and the singleton instance reads the sole self-overlap transition as its complete transition ledger.*
10. *The standard bridge row is absent, and the package is a paper-side formalization target under* $\mathsf{BEDC.Derived.ManifoldUp}$.

Theorem 96.42 (Manifold scoped field coverage)**.** *Every package satisfying the scoped boundary of Definition 96.41 exposes exactly the following visible fields: the* $\mathsf{ManAtlasPkg}$ *carrier, the componentwise* hsame *classifier, chart cover, chart-domain transport, pairwise overlap source, triple-overlap source, transition-ledger stability, identity rows, inverse rows, and cocycle rows.*

Proof. Project the carrier and classifier from Definition 96.41. The carrier is the same $\mathsf{ManAtlasPkg}$ predicate defined in Definition 96.2, and the classifier is its componentwise hsame row.

The chart-cover and chart-domain transport fields are supplied by Definition 96.3 and Definition 96.4. The pairwise overlap source is supplied by Definition 96.7, and the triple-overlap source is supplied by Definition 96.8.

The transition-ledger stability field is exactly Definition 96.18. The identity, inverse, and cocycle fields are projected from the coherence ledger by Definition 96.21, Definition 96.22, and Definition 96.23.

No additional field is available from the scoped boundary definition. Its exactness clause accepts only the displayed atlas, topology, coordinate, overlap, transition, and coherence rows, so the listed fields are precisely the public boundary of the package. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldScopedBoundaryPackage field coverage`

Theorem 96.43 (Manifold scoped transition-source determinacy)**.** *In every package satisfying Definition 96.41, each displayed transition row* Trans_{ij} *is sourced on the carried open* $\mathsf{domIx}(i) \wedge_{\mathsf{Topology}^{\uparrow}} \mathsf{domIx}(j)$*. For each displayed composite from* i *to* k *through* j*, the source is the iterated* $\mathsf{Topology}^{\uparrow}$ *finite-intersection triple-overlap carrying*

$$x \longmapsto \mathsf{Dom}_i(x) \wedge \mathsf{Dom}_j(x) \wedge \mathsf{Dom}_k(x).$$

There is no hidden set-level intersection behind either source.

Proof. For the pairwise source, apply Definition 96.6. It identifies the overlap predicate for i and j with the carried open $\mathsf{domIx}(i) \wedge_{\mathsf{Topology}^{\uparrow}} \mathsf{domIx}(j)$ in the same $\mathsf{Topology}^{\uparrow}$ carrier.

Then apply Definition 96.7. The theorem states that every displayed Trans_{ij} ledger row uses that carried open as its source, with finite $\mathsf{Real}^{\uparrow}$ coordinate rows exposed on both ends.

For a composite through j, build the triple source by Definition 96.8. The iterated finite-intersection index carries the displayed triple conjunction, and the two transition ledgers restrict to that same source before composition.

Finally use Definition 96.23. The cocycle row classifies the displayed composite and the direct i to k transition on the same carried triple-overlap source. Since each step is read through $\mathsf{Topology}^{\uparrow}$ carried-open rows, no unrecorded set intersection is introduced. □

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldScopedBoundaryPackage triple overlap source determinacy`

Theorem 96.44 (Singleton manifold scoped boundary instance)**.** *The singleton package Definition 96.9 instantiates the scoped boundary package of Definition 96.41. Its carrier is* C_{e}*, its chart domain is the top-open row, its coordinate row is the constant* $0_{\mathsf{Real}^{\uparrow}}$ *endpoint, and its complete transition ledger is the sole self-overlap transition.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldSingleton scoped boundary instance`

Proof. The carrier and topology rows are supplied by Definition 96.31. They identify the restricted hsame classifier on C_{e} and read the top-open row exactly as the carried singleton predicate.

Definition 96.32 supplies the chart-cover row and the constant $0_{\mathsf{Real}^{\uparrow}}$ coordinate readback. The domain-exactness field is supplied by Definition 96.35, which leaves no second displayed chart-domain witness.

The transition ledger is supplied by Definition 96.36. Every displayed transition factors through the sole self-overlap of χ_{e}, with source and target the constant $0_{\mathsf{Real}^{\uparrow}}$ endpoint.

Repack these rows with Definition 96.37. They match the ten scoped-boundary fields: traditional singleton target, BEDC source, carrier, classifier, stability, coherence, ledger, core theorem list, exactness, and absent standard bridge. □

96.6 Terminal Standard Bridge

Theorem 96.45 (ManifoldUp terminal standard bridge)**.** *The public singleton empty-chart* Manifold$^{\uparrow}$ *package of Definition 96.26 admits a paper-side standard bridge to the terminal smooth-manifold reference. The bridge sends every carried BEDC endpoint of the singleton carrier C_{e} to the unique terminal point, and its return map selects the singleton empty-chart endpoint of Definition 96.9. Under this bridge the carrier, classifier, chart domain, coordinate row, transition row, and visible ledger rows are preserved and reflected. No host equality, host quotient, or host function is added to the BEDC public surface.*

Lean4 ✓ `BEDC.Derived.ManifoldUp.ManifoldUp StdBridge`

Proof. The source certificate is the singleton semantic naming certificate supplied by Definition 96.37. Its carrier is the one-point history package C_{e}, and its classifier is the restricted hsame row on that carrier.

Define the bridge-to-reference map by sending every carried C_{e} endpoint to the unique point of the terminal smooth manifold. Define the return map by selecting the endpoint of the empty-chart package. The carrier and classifier round trips follow from Definition 96.31: both composites land back in the sole carried endpoint, and the exposed classifier is the same restricted hsame row.

The chart data are preserved by the atlas package threshold. Apply Definition 96.26 to expose the public carrier, classifier, chart cover, chart-domain transport, transition composition, transition-ledger stability, and visible atlas ledger. The singleton chart coverage, domain exactness, and coordinate transport rows of Definition 96.32, Definition 96.35, and Definition 96.28 identify the terminal reference chart with the empty chart and its constant coordinate row.

The transition component is the sole self-overlap row. It is stable under the bridge because Definition 96.36 exhausts the singleton transition ledger, while Definition 96.5 and Definition 96.18 keep transition composition and ledger readback inside the displayed Manifold$^{\uparrow}$ rows.

The theorem-preservation field is therefore finite and explicit: it consists only of the carrier, classifier, chart-domain, coordinate, transition, and ledger rows just cited. The no-host-leak field holds at paper level because the bridge maps are used only to read the terminal smooth-manifold reference; the BEDC public surface remains the singleton empty-chart certificate and does not import host equality, quotients, or functions. □

Closure status for NameCert$_{\text{Manifold}^{\uparrow}}$**.**

Constructive story: The closed manifold object in this chapter is the singleton empty-chart package: start with the empty-history carrier C_{e} supplied by the topology surface, give it the single chart index, read its chart domain as the top open row, and make the coordinate and transition ledgers the constant rows already certified by Real$^{\uparrow}$ and RealAnalytic$^{\uparrow}$. A history is admitted only as that singleton atlas package, and two admitted presentations are classified by the restricted hsame comparison on the carrier together with componentwise transport of the chart-domain, coordinate, and transition rows; nonterminal atlas entries are not accepted. The source specification is the consumed Topology$^{\uparrow}$ carrier plus the constant real-coordinate and real-analytic transition boundaries, the pattern specification is chart coverage, domain exactness, transition smoothness, chart-index exhaustion, and transition-ledger exhaustion, and the ledger keeps the topology rows, chart index, coordinate row, self-overlap transition, and visible atlas provenance in the public package. The five name-certificate obligations are supplied by the empty endpoint, by reflexivity, symmetry, and transitivity of the restricted history classifier, and by the atlas transport rows showing that carrier, domain, coordinate, and transition data survive replacement by a classified presentation. A history h lies in the signature gap of a Manifold$^{\uparrow}$ token p when the source-domain accepts h as this singleton atlas package, the signature relation pairs it with an answer atlas and evidence ledger, and the token-introduction relation names that answer as p; concretely, h is the empty-chart endpoint with the unique domain, coordinate, and self-transition rows encoded by p. The terminal standard bridge sends every carried BEDC endpoint to the unique point of the terminal smooth manifold and reads that point back as the singleton empty-chart endpoint, preserving and reflecting carrier, classifier, chart-domain, coordinate, transition, and ledger rows. The chapter

registers this terminal atlas construction under the name $\mathsf{Manifold}^{\uparrow}$; arbitrary manifolds, nonterminal atlases, host equality, host quotients, and nonterminal smooth-manifold equivalences are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The closed paper scope includes the singleton empty-chart carrier, classifier, atlas rows, terminal standard bridge, and the mature boundary recorded by Definition 96.45.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.ManifoldUp.ManifoldSingleton\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: Arbitrary manifolds, non-terminal atlases, non-terminal smooth-manifold equivalences, and Lean-side bridge verification remain outside this paper closure.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 97

A Concrete Naming Certificate for $\mathsf{Bundle}^{\uparrow}$

The bundle interface $\mathsf{Bundle}^{\uparrow}$ packages a fibred carrier $E \to B$ with locally trivial fibres of a fixed model type. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) for the base and total-space topology, and supplies the substrate for the tangent, cotangent, and frame bundles.

97.1 Carrier definition

Definition 97.1 (Bundle local-trivialization obligation surface)**.** *The local-trivialization obligation surface for* $\mathsf{Bundle}^{\uparrow}$ *is the ten-field* $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$ *template whose rows are fixed as follows.*

1. *The traditional target is a fibred carrier* $\pi : E \to B$ *whose fibres are locally trivial over a fixed fibre model* F*.*
2. *The BEDC source is* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *together with visible* BHist *histories for the base space, the total space, the projection, local trivializations, transition data, and the source ledger.*
3. *The carrier is a* BHist *package* $\mathsf{BundleLocalTrivPkg}(P)$ *exposing a base manifold row* B*, a total manifold row* E*, a projection row* π*, a fibre-model row* F*, finite chart-trivialization* ProbeBundle *rows, transition rows on overlaps, and a ledger row recording the source of each displayed field.*
4. *The retained manifold classifier from* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *classifies the base and total manifold rows.*
5. *The remaining classifier is componentwise* hsame *on projection histories, fibre-model histories, trivialization* ProbeBundle *rows, transition rows, and source-ledger rows.*
6. *Carrier stability transports* BundleLocalTrivPkg *packages along the retained manifold classifier on* B *and* E *and along componentwise* hsame *on every displayed bundle row.*
7. *Projection compatibility says that transport of* π *along the base and total classifiers remains the displayed projection row between the transported manifold rows.*
8. *Transition congruence says that if two overlapping trivialization rows are transported by componentwise* hsame*, then the corresponding transition row is transported to the displayed transition row over the transported overlap.*
9. *Ledger descent says that every transported carrier, projection, trivialization, fibre-model, and transition row keeps the same source entry after replacing its fields by classified histories.*

10. *Exactness is visible local-trivialization readback: every accepted* $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$ *field is one of the carrier, classifier, stability, projection, transition, ledger, and readback rows above. The standard bridge row is absent.*

97.2 The certificate

Theorem 97.2 (Bundle carrier projection obligation)**.** *Every carried row of* $\mathsf{BundleLocalTrivPkg}(P)$ *exposes exactly the base manifold row, total manifold row, projection row, fibre-model row, trivialization* ProbeBundle *row, transition row, and source-ledger row displayed in Definition 97.1. There is no other carrier case in the* $\mathsf{Bundle}^{\uparrow}$ *obligation surface.*

Proof. Unpack $\mathsf{BundleLocalTrivPkg}(P)$ by Definition 97.1. The carrier slot is a BHist package whose displayed fields are the base row, total row, projection row, fibre-model row, trivialization ProbeBundle rows, transition rows, and source ledger.

Projecting a carried bundle row therefore reads one of those listed fields. The base and total projections are manifold rows classified by the upstream $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ row, while the projection, fibre, trivialization, transition, and ledger projections are the componentwise hsame fields of the same package.

No additional branch is available: the carrier predicate is the displayed package predicate itself, and its final readback row accepts only the listed fields. Thus every carried row is one of the named rows, and the carrier has no second case. □

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPackage carrier projection`
Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg projection rows`

Theorem 97.3 (Bundle trivialization transition stability)**.** *For packages satisfying* BundleLocalTrivPkg*, componentwise* hsame *transport preserves local trivialization rows and transition compatibility through the retained* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *classifier on the base and total manifold rows.*

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPackage classifier transport`
Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg trivialization transition stability`
Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg transition stability`

Proof. Start with a carried package and a componentwise transport witness. The base and total manifold rows are transported by the classifier retained from $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, so their chart and overlap histories remain accepted manifold histories.

The trivialization row is a finite ProbeBundle of local witnesses. Induct on that closed probe bundle. The empty probe row is preserved by the carrier-stability clause, and each bundled probe is transported by the componentwise hsame classifier on its trivialization history.

For an overlap row, the transition congruence field of Definition 97.1 compares the two transported trivialization rows and the displayed transition row. Projection compatibility supplies the same base and total rows after transport, so the transition still relates fibres over the transported base overlap.

Ledger descent then transports the provenance row for the trivialization and transition witnesses. Hence local trivialization rows and transition compatibility are preserved by the displayed hsame transport. □

Theorem 97.4 (Bundle transition cocycle ledger closure)**.** *Let* P *be carried by* $\mathsf{BundleLocalTrivPkg}(P)$*, and let* τ_i, τ_j, τ_k *be three local trivialization rows in its finite* ProbeBundle *row. Suppose the pairwise*

overlaps $\mathsf{Overlap}_{ij}$ *and* $\mathsf{Overlap}_{jk}$ *are accepted by the same trivialization* ProbeBundle*, and suppose their common base restriction carries the direct transition row from* τ_i *to* τ_k *and the composed route through* τ_j *with continuation witness* $\mathsf{Cont}(\mathsf{Trans}_{ij}, \mathsf{Trans}_{jk}, \mathsf{Trans}_{ijk})$*. Then, after transporting the base and total manifold rows by the retained* $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ *classifier, the direct transition row and the composed transition row are componentwise* hsame*. The source-ledger row for this comparison is exactly the row displaying* τ_i, τ_j, τ_k*, the two overlap witnesses, the displayed* Cont *composition witness, and the transition congruence witnesses for the three transition edges.*

Proof. Work inside the carrier displayed in Definition 97.1. The three trivialization rows and the two accepted overlap witnesses live in the closed finite ProbeBundle row of P.

Induct on the finite ProbeBundle of overlap witnesses. In the empty case there is no additional overlap branch to inspect: ledger descent keeps the displayed provenance row, and reflexivity of hsame identifies the direct and composed transition histories on the vacuous carried overlap.

For the cons step, transport the head transition by componentwise hsame. Projection compatibility keeps the transported base and total manifold rows aligned with the displayed projection row, so all three transition edges still have the same source and target bundle rows on the common triple restriction.

Apply transition congruence to the i, j, j, k, and i, k transition edges. Continuation associativity for Cont, followed by transitivity of hsame, identifies the direct i, k transition history with the route composed through the middle trivialization row τ_j.

Finally, ledger descent repacks the same displayed provenance: the three trivialization rows, the two overlap witnesses, the Cont composition witness, and the transition congruence witnesses. This is the claimed cocycle-style ledger row for the bundle transition comparison. □

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg transition cocycle ledger closure`

Theorem 97.5 (Bundle NameCert exactness obligation)**.** *The public fields of* $\mathsf{NameCert}_{\mathsf{Bundle}\uparrow}$ *are exactly the carrier, classifier, stability, transition ledger, visible exactness readback, and absent bridge row specified by Definition 97.1.*

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg semantic name certificate`

Proof. First project the carrier by Definition 97.2. This accounts for the base manifold row, total manifold row, projection row, fibre-model row, trivialization ProbeBundle row, transition row, and source-ledger row.

Next read the classifier and stability fields from the definition. The classifier is the retained $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ classifier on base and total rows together with componentwise hsame on all bundle-specific histories. The stability fields are carrier transport, projection compatibility, transition congruence, and ledger descent.

Finally apply the visible local-trivialization readback row. It accepts only the displayed carrier fields, the displayed classifier and stability fields, the transition ledger, and the explicit statement that the standard bridge row is absent. Therefore the public $\mathsf{NameCert}_{\mathsf{Bundle}\uparrow}$ surface is exhausted by exactly those fields. □

Theorem 97.6 (Bundle projection and fibre source scope)**.** *In every carried* $\mathsf{BundleLocalTrivPkg}(P)$ *row, the base row, total row, projection row, fibre-model row, and source row are scoped entirely by* BHist*. The base and total rows use the retained* $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ *classifier, while projection, fibre, and source rows are compared by componentwise* hsame *rather than by host-level functions.*

Proof. Apply Definition 97.1 to read the displayed bundle surface. Its source row is a visible BHist ledger, and its base, total, projection, and fibre-model entries are also displayed as histories inside BundleLocalTrivPkg(P).

Apply Definition 97.2 to project those rows from the carried package. The base and total rows are classified by the retained $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ row, and every bundle-specific row is transported only by the componentwise hsame clauses of the surface.

Thus no host function is part of the scoped comparison. The fibre and projection data remain named histories, and the ledger records their sources in the same BHist vocabulary. □

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg projection fibre source scope`

Theorem 97.7 (Bundle trivialization ProbeBundle scope)**.** *The finite trivialization rows in* BundleLocalTrivPkg(P) *are exactly* ProbeBundle *witnesses. Componentwise* hsame *transport preserves their scope by induction on the closed constructors* Bnil *and* Bcons*.*

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg trivialization scope`

Proof. Read the trivialization row from Definition 97.1. The row is not an external family of functions: it is the finite ProbeBundle field listed in the carried bundle package.

Induct on the displayed probe bundle. In the Bnil case, the empty witness has no chart row to transport, so the carrier-stability field preserves the row directly.

In the Bcons case, the head witness is transported by the componentwise hsame classifier on its trivialization history. The induction hypothesis preserves the remaining witnesses, and the closed constructor reassembles the transported ProbeBundle row.

This is precisely the stability mechanism used in Definition 97.3, so finite trivialization scope is preserved without leaving the BEDC history language. □

Lean4 ✓ `BEDC.Derived.BundleUp.BundleRowsUnary classifier transport`

Theorem 97.8 (Bundle transition and ledger scope)**.** *Transition rows and source-ledger rows in* BundleLocalTrivPkg(P) *descend under componentwise* hsame*. The descent uses the retained* $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ *classifier only for the base and total manifold rows; overlap transitions and ledger entries remain bundle-specific* BHist *rows.*

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkg transition ledger scope`

Proof. Start with two transported trivialization rows. By Definition 97.3, their overlap row is still compared after transport through the displayed transition congruence field.

The manifold part of the comparison is exactly the retained $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ classifier on base and total rows. The transition row itself is not reinterpreted as a host map; it is the transported bundle-specific history named by the componentwise hsame field.

Finally apply the ledger-descent row of Definition 97.1. It sends the source entry for each carrier, projection, trivialization, fibre-model, and transition row to the corresponding transported source entry. Hence both the transition row and the ledger row descend in the same scoped vocabulary. □

Theorem 97.9 (Bundle scoped obligation package)**.** *The carrier projection, projection–fibre source scope, finite* ProbeBundle *trivialization scope, transition–ledger descent, and visible exactness rows together form the scoped* $\mathsf{NameCert}_{\mathsf{Bundle}\uparrow}$ *obligation package. Its public surface is kernel-facing: all bundle-specific data are* BHist *rows compared by componentwise* hsame*, and the only retained external classifier is* $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$ *on the base and total manifold rows.*

Lean4 ✓ `BEDC.Derived.BundleUp.BundleLocalTrivPkgRows scoped unary coverage`

Proof. First assemble the carrier rows using Definition 97.2. This supplies the base, total, projection, fibre-model, trivialization, transition, and source-ledger rows named by $\mathsf{BundleLocalTrivPkg}(P)$.

Next add the scoped comparisons. The projection and fibre source claim is Definition 97.6; finite trivialization scope is Definition 97.7; transition and ledger descent is Definition 97.8.

It remains to identify the public certificate surface. Apply Definition 97.5: the accepted $\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$ fields are exactly the displayed carrier, classifier, stability, transition ledger, readback, and absent-bridge rows. Therefore the scoped rows above repack into the stated kernel-facing obligation package. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$.

Constructive story: A bundle is built as a visible local-trivialization package: base and total manifold rows from $\mathsf{Manifold}^\uparrow$, a projection history, a fibre-model history, a finite ProbeBundle of trivialization witnesses, transition histories on overlaps, and a source ledger. The carrier accepts exactly these BundleLocalTrivPkg packages and has no second branch; the base and total rows are compared by the retained manifold classifier, while projection, fibre, trivialization, transition, and ledger rows are compared componentwise by hsame. The source is the manifold certificate together with the displayed bundle histories, the pattern records projection compatibility, finite trivialization scope, transition congruence and cocycle-style ledger closure, and visible exactness, while the ledger keeps each carrier, projection, trivialization, fibre, and transition source entry aligned with its transported row. The scoped name-certificate obligations are read from those projections: an accepted package inhabits the carrier, the retained manifold classifier and componentwise hsame supply reflexivity, symmetry, and transitivity, and carrier transport plus ledger descent keep the package admitted after classified replacement. A history h lies in the signature gap of a $\mathsf{Bundle}^\uparrow$ token p when the source-domain accepts h as this local-trivialization package, the signature relation pairs it with an answer package and evidence ledger, and token introduction names that answer as p; unfolded, the projection, finite probes, transitions, and source ledger are the exact rows encoded by p. No standard bridge row is present, so no ordinary bundle equivalence is exported. The chapter registers this local-trivialization package under the name $\mathsf{Bundle}^\uparrow$; bundle structure is a finite BEDC history ledger over $\mathsf{Manifold}^\uparrow$, with no host bundle object, tangent-bundle primitive, or external transition-function library introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$, maturePackageClosed)

Scope closed: The mature paper surface binds the finite local-trivialization certificate package: carrier projection, projection–fibre source scope, finite ProbeBundle trivialization scope, transition-ledger descent, visible exactness rows, BHist carrier histories, componentwise hsame comparisons, and retained $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ classifier named in this chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.BundleUp.BundleLocalTrivPkg\_semantic\_name\_certificate`

Bridge status: none

Not claimed: Standard bundle equivalence, tangent and cotangent bundles, frame bundles, and bridgeChecked verification remain outside this block.

Upgrade path: The next verification step is bridgeChecked coverage for the finite local-trivialization certificate package.

Chapter 98

A Concrete Naming Certificate for $\mathsf{Homology}^{\uparrow}$

The homology interface $\mathsf{Homology}^{\uparrow}$ packages a chain complex over an abelian-group carrier together with the kernel/image quotients computing the homology groups. It depends on $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ (chapter 42) and $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$ (chapter 82), and is the constructive substrate for both singular and simplicial homology.

98.1 Carrier definition

The cycle carrier of a BEDC chain differential $d : \mathsf{BHist} \to \mathsf{BHist}$ is the kernel predicate

$$Z_d(h) :\Longleftrightarrow \mathsf{hsame}(d(h), \mathsf{Empty}).$$

The boundary carrier is the image predicate

$$B_d(h) :\Longleftrightarrow \exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)).$$

Lean4 (definition) `BEDC.Derived.HomologyUp.HomologyCycleCarrier`
Lean4 (definition) `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier`

Theorem 98.1 (Homology boundary carrier nonempty preimage)**.** *If $B_d(h)$ and h is not history-same to* Empty*, then some boundary preimage u has a nonempty differential endpoint:*

$$\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \neg\mathsf{hsame}(d(u), \mathsf{Empty}).$$

Proof. Choose the boundary witness u with $\mathsf{hsame}(h, d(u))$. If $d(u)$ were history-same to Empty, transitivity of hsame would make h history-same to Empty, contradicting the nonempty hypothesis. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier nonempty preimage`

Theorem 98.2 (Homology boundary carrier empty preimage)**.** *If $B_d(h)$ and h*$\mathsf{hsameEmpty}$*, then some boundary preimage u has an empty differential endpoint:*

$$\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(d(u), \mathsf{Empty}).$$

Proof. Choose the boundary witness u with $\mathsf{hsame}(h, d(u))$. Symmetry of that witness followed by the displayed empty endpoint comparison gives $\mathsf{hsame}(d(u), \mathsf{Empty})$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier empty preimage`

Theorem 98.3 (Homology boundary carrier cycle closure)**.** *Assume the BEDC chain differential is square-zero at every history:*

$$\forall u : \mathsf{BHist},\ \mathsf{hsame}(d(d(u)), \mathsf{Empty}).$$

Every boundary history is a cycle history: if $B_d(h)$, then $Z_d(h)$.

Proof. Choose a boundary witness u with $\mathsf{hsame}(h, d(u))$. Transporting the cycle test along this witness reduces the claim to the displayed square-zero law at the chosen boundary preimage, which is supplied by feeding that preimage to the displayed universal square-zero law. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cycle closed`

Theorem 98.4 (Homology boundary classifier refines cycle classifier)**.** *Assume the BEDC chain differential is square-zero at every history:*

$$\forall u : \mathsf{BHist},\ \mathsf{hsame}(d(d(u)), \mathsf{Empty}).$$

If $B_d(h)$, $B_d(k)$, and $\mathsf{hsame}(h, k)$, then $Z_d(h)$, $Z_d(k)$, and $\mathsf{hsame}(h, k)$.

Proof. Apply Definition 98.3 to the boundary witness at h, using the displayed square-zero law at the chosen preimage. This produces the cycle endpoint $Z_d(h)$.

Apply the same carrier-closure theorem to the boundary witness at k. The existing classifier row $\mathsf{hsame}(h, k)$ is preserved unchanged, so the three pieces repack as the cycle classifier row between the same endpoints. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cycle classifier refinement`

Theorem 98.5 (Homology cycle carrier append closure)**.** *Assume the differential respects BEDC continuation append:*

$$\mathsf{hsame}(d(\mathsf{append}(u, v)), \mathsf{append}(d(u), d(v))).$$

If $Z_d(h)$ and $Z_d(k)$, then $Z_d(\mathsf{append}(h, k))$.

Proof. The append law transports the displayed differential to $\mathsf{append}(d(h), d(k))$. The two cycle hypotheses identify both endpoints with Empty, and the BEDC append kernel has $\mathsf{append}(\mathsf{Empty}, \mathsf{Empty}) = \mathsf{Empty}$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier append closed`

Theorem 98.6 (Homology cycle carrier append visible differential absurd)**.** *Assume the differential respects BEDC continuation append. If $Z_d(h)$ and $Z_d(k)$, then the differential of the appended cycle cannot be history-same to a visible zero-headed or one-headed endpoint:*

$$\neg\mathsf{hsame}(d(\mathsf{append}(h, k)), \mathsf{e0}(tail)) \qquad \textit{and} \qquad \neg\mathsf{hsame}(d(\mathsf{append}(h, k)), \mathsf{e1}(tail)).$$

Proof. The homology cycle append closure theorem gives $\mathsf{hsame}(d(\mathsf{append}(h, k)), \mathsf{Empty})$. A displayed zero-headed or one-headed comparison transports this empty endpoint to the visible constructor endpoint, contradicting BEDC constructor separation from Empty. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier append visible differential absurd`

Theorem 98.7 (Homology cycle carrier append components)**.** *Assume the differential respects BEDC continuation append:*

$$\mathsf{hsame}(d(\mathsf{append}(u, v)), \mathsf{append}(d(u), d(v))).$$

Then an appended history is a cycle exactly when both appended components are cycles:

$$Z_d(\mathsf{append}(h, k)) \iff Z_d(h) \wedge Z_d(k).$$

Proof. The forward direction transports $d(\mathsf{append}(h, k))$ to $\mathsf{append}(d(h), d(k))$. If this endpoint is empty, BEDC append-empty inversion gives $d(h)\mathsf{hsameEmpty}$ and $d(k)\mathsf{hsameEmpty}$. The reverse direction is the homology cycle append-closure theorem. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier append components iff`

Theorem 98.8 (Homology cycle carrier append differential nonempty component split)**.** *Assume the differential respects BEDC continuation append. If the differential of an appended history is not empty,*

$$\mathsf{hsame}(d(\mathsf{append}(h, k)), \mathsf{Empty}) \to \bot,$$

then at least one component is not a cycle:

$$(Z_d(h) \to \bot) \vee (Z_d(k) \to \bot).$$

Proof. Transport the displayed nonemptiness along the append law for d to obtain nonemptiness of $\mathsf{append}(d(h), d(k))$. BEDC append nonemptiness then separates this obstruction into a nonempty $d(h)$ component or a nonempty $d(k)$ component. Since $Z_d(h)$ and $Z_d(k)$ mean exactly that the corresponding differential is empty, the selected obstruction negates the matching cycle predicate. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier append differential nonempty component split`

Theorem 98.9 (Homology boundary carrier append closure)**.** *Assume the differential respects BEDC continuation append:*

$$\mathsf{hsame}(d(\mathsf{append}(u, v)), \mathsf{append}(d(u), d(v))).$$

If $B_d(h)$ and $B_d(k)$, then $B_d(\mathsf{append}(h, k))$.

Proof. Choose boundary witnesses u and v with $\mathsf{hsame}(h, d(u))$ and $\mathsf{hsame}(k, d(v))$. The BEDC witness for the appended boundary is $\mathsf{append}(u, v)$. Endpoint congruence carries $\mathsf{append}(h, k)$ to $\mathsf{append}(d(u), d(v))$, and the differential append law identifies that history with $d(\mathsf{append}(u, v))$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier append closed`

Theorem 98.10 (Homology boundary carrier append hsame transport)**.** *Assume the differential respects BEDC continuation append. If $B_d(h)$ and $B_d(k)$ and $\mathsf{hsame}(\mathsf{append}(h, k), r)$, then $B_d(r)$.*

Proof. Choose boundary witnesses u and v for h and k. Their appended BEDC preimage $\mathsf{append}(u, v)$ maps, by the differential append law, to the same endpoint as $\mathsf{append}(d(u), d(v))$. The two boundary endpoint comparisons identify $\mathsf{append}(h, k)$ with this value, and the final hsame comparison transports the image witness to r. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier append hsame transport`

Theorem 98.11 (Homology boundary carrier continuation cycle closure)**.** *Assume the differential respects BEDC continuation append and is square-zero. If $B_d(h)$, $B_d(k)$, and $\mathsf{Cont}(h, k, r)$, then $Z_d(r)$.*

Proof. Boundary append closure with endpoint transport first makes the continuation result r a boundary carrier. The square-zero differential condition turns any boundary carrier into a cycle carrier, so $Z_d(r)$ follows. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont cycle closed`

Theorem 98.12 (Homology boundary carrier continuation preimage)**.** *Assume the differential respects BEDC continuation append. If $B_d(h)$, $B_d(k)$, and $\mathsf{Cont}(h, k, r)$, then the continuation result has compatible boundary preimages:*

$$\exists u, v, uv : \mathsf{BHist},\ \mathsf{Cont}(u, v, uv) \wedge \mathsf{hsame}(r, d(uv)) \wedge \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)).$$

Proof. Choose boundary witnesses u and v for h and k. Set $uv = \mathsf{append}(u, v)$, giving $\mathsf{Cont}(u, v, uv)$ by the BEDC continuation constructor. The input continuation reads r as $\mathsf{append}(h, k)$; the two boundary witnesses identify this with $\mathsf{append}(d(u), d(v))$, and the differential append law identifies the endpoint with $d(uv)$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont preimage`

Theorem 98.13 (Homology boundary carrier continuation preimage append)**.** *Assume the differential respects BEDC continuation append. If $B_d(h)$, $B_d(k)$, and $\mathsf{Cont}(h, k, r)$, then there are BEDC preimages u and v such that*

$$\mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(r, d(\mathsf{append}(u, v))).$$

Proof. Choose boundary witnesses u and v for h and k. The continuation ledger identifies r with $\mathsf{append}(h, k)$. The witness endpoints identify that appended history with $\mathsf{append}(d(u), d(v))$, and the differential append law identifies this with $d(\mathsf{append}(u, v))$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont preimage append`

Theorem 98.14 (Homology boundary carrier continuation preimage append hsame transport)**.** *Assume the differential respects BEDC continuation append. If $B_d(h)$, $B_d(k)$, $\mathsf{Cont}(h, k, r)$, and $r \mathsf{hsame} r'$, then the same BEDC preimages u and v witness the transported endpoint:*

$$\mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(r', d(\mathsf{append}(u, v))).$$

Proof. The continuation preimage-append theorem supplies witnesses u and v for the endpoint r. Composing that endpoint comparison with the symmetry of $r \mathsf{hsame} r'$ transports the displayed differential image to r' while keeping the two boundary witness comparisons unchanged. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont preimage append hsame transport`

Theorem 98.15 (Homology boundary continuation cycle-preimage package)**.** *Assume the differential respects BEDC continuation append and is square-zero. If $B_d(h)$, $B_d(k)$, and $\mathsf{Cont}(h, k, r)$, then*

$$Z_d(r) \wedge \exists u, v : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(r, d(\mathsf{append}(u, v))).$$

Proof. The cycle component is exactly Definition 98.11: boundary continuation first produces a boundary carrier for r, and square-zero turns that carrier into $Z_d(r)$.

The preimage component is supplied by Definition 98.13. It gives BEDC histories u and v whose differential images match h and k, and whose appended differential image matches the continuation result r.

Conjoining these two conclusions gives the displayed cycle-preimage certificate for the same continuation result. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier continuation cycle preimage package`

Theorem 98.16 (Homology boundary carrier continuation nonempty preimage append)**.** *Assume the differential respects BEDC continuation append. If $B_d(h)$, $B_d(k)$, $\mathsf{Cont}(h,k,r)$, and r is not history-same to* Empty*, then there are BEDC preimages u and v such that*

$$\mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \neg\mathsf{hsame}(d(\mathsf{append}(u,v)), \mathsf{Empty}).$$

Proof. The preimage-append theorem gives witnesses u and v with $\mathsf{hsame}(r, d(\mathsf{append}(u,v)))$. If the displayed differential endpoint were history-same to Empty, transitivity would make r history-same to Empty, contradicting the nonempty hypothesis. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont nonempty preimage append`

Theorem 98.17 (Homology boundary carrier append empty preimages)**.** *If $B_d(h)$ and $B_d(k)$ and* $\mathsf{hsame}(\mathsf{append}(h,k), \mathsf{Empty})$*, then the two boundary witnesses can be chosen with empty differential endpoints:*

$$\exists u, v : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(d(u), \mathsf{Empty}) \wedge \mathsf{hsame}(d(v), \mathsf{Empty}).$$

Proof. BEDC append-empty inversion splits the appended endpoint comparison into $h\mathsf{hsameEmpty}$ and $k\mathsf{hsameEmpty}$. Applying empty-preimage inversion to the two boundary carrier hypotheses produces witnesses u and v with the stated endpoint comparisons, which are then paired. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier append empty preimages`

Theorem 98.18 (Homology boundary carrier continuation empty preimages)**.** *If $B_d(h)$ and $B_d(k)$ and $\mathsf{Cont}(h,k,r)$ and $r\mathsf{hsameEmpty}$, then the two boundary witnesses can be chosen with empty differential endpoints:*

$$\exists u, v : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(d(u), \mathsf{Empty}) \wedge \mathsf{hsame}(d(v), \mathsf{Empty}).$$

Proof. The continuation equation identifies r with $\mathsf{append}(h,k)$. Transporting the empty result comparison along this equation gives $\mathsf{append}(h,k)\mathsf{hsameEmpty}$, and the append empty preimage theorem supplies the paired witnesses. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont empty preimages`

Theorem 98.19 (Homology boundary carrier continuation empty-result preimage equivalence)**.** *If $B_d(h)$ and $B_d(k)$ and $\mathsf{Cont}(h,k,r)$, then r is history-same to* Empty *exactly when the two boundary witnesses can be chosen with empty differential endpoints:*

$$\mathsf{hsame}(r, \mathsf{Empty}) \iff \exists u, v : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \mathsf{hsame}(k, d(v)) \wedge \mathsf{hsame}(d(u), \mathsf{Empty}) \wedge \mathsf{hsame}(d(v), \mathsf{Empty}).$$

Proof. The forward direction is the continuation empty-preimage theorem. Conversely, the two displayed empty differential endpoints transport through the boundary witnesses to give $h\mathsf{hsameEmpty}$ and $k\mathsf{hsameEmpty}$; append empty-result exactness makes $\mathsf{append}(h, k)$ empty, and the continuation ledger identifies this append endpoint with r. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont empty result preimages iff`

Theorem 98.20 (Homology boundary carrier append nonempty preimage)**.** *If $B_d(h)$ and $B_d(k)$ and $\mathsf{append}(h, k)$ is not history-same to Empty, then at least one side has a boundary preimage with a nonempty differential endpoint:*

$$(\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \neg\mathsf{hsame}(d(u), \mathsf{Empty})) \\ \vee (\exists v : \mathsf{BHist},\ \mathsf{hsame}(k, d(v)) \wedge \neg\mathsf{hsame}(d(v), \mathsf{Empty}))\,.$$

Proof. BEDC append nonemptiness splits into a nonempty left component or a nonempty right component. In the left case, boundary nonempty-preimage inversion applied to $B_d(h)$ gives the left disjunct. In the right case, the same argument applied to $B_d(k)$ gives the right disjunct. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier append nonempty preimage`

Theorem 98.21 (Homology boundary carrier continuation nonempty side preimage)**.** *If $B_d(h)$ and $B_d(k)$ and $\mathsf{Cont}(h, k, r)$ and r is not history-same to Empty, then at least one side has a boundary preimage with a nonempty differential endpoint:*

$$(\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge \neg\mathsf{hsame}(d(u), \mathsf{Empty})) \\ \vee (\exists v : \mathsf{BHist},\ \mathsf{hsame}(k, d(v)) \wedge \neg\mathsf{hsame}(d(v), \mathsf{Empty}))\,.$$

Proof. The continuation equation transports nonemptiness of r to nonemptiness of $\mathsf{append}(h, k)$. The append nonempty preimage theorem then separates the witness to the left boundary preimage or to the right boundary preimage. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont nonempty side preimage`

Theorem 98.22 (Homology boundary carrier append nonempty cycle preimage)**.** *Assume the differential is square-zero. If $B_d(h)$ and $B_d(k)$ and $\mathsf{append}(h, k)$ is not history-same to Empty, then at least one side has a boundary preimage with a nonempty differential endpoint, and that side is a cycle:*

$$(\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge Z_d(h) \wedge \neg\mathsf{hsame}(d(u), \mathsf{Empty})) \\ \vee (\exists v : \mathsf{BHist},\ \mathsf{hsame}(k, d(v)) \wedge Z_d(k) \wedge \neg\mathsf{hsame}(d(v), \mathsf{Empty}))\,.$$

Proof. The append nonempty preimage theorem splits the nonempty append boundary into a left or right boundary preimage. The square-zero differential turns each boundary carrier hypothesis into the corresponding cycle carrier, and that cycle witness is attached to whichever disjunct supplies the nonempty preimage. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier append nonempty cycle preimage`

Theorem 98.23 (Homology boundary carrier continuation nonempty cycle preimage)**.** *Assume the differential is square-zero. If $B_d(h)$ and $B_d(k)$, if $\mathsf{Cont}(h, k, r)$, and if r is not history-same to Empty, then at least one side has a boundary preimage with a nonempty differential endpoint, and that side is a cycle:*

$$(\exists u : \mathsf{BHist},\ \mathsf{hsame}(h, d(u)) \wedge Z_d(h) \wedge \neg\mathsf{hsame}(d(u), \mathsf{Empty})) \\ \vee (\exists v : \mathsf{BHist},\ \mathsf{hsame}(k, d(v)) \wedge Z_d(k) \wedge \neg\mathsf{hsame}(d(v), \mathsf{Empty}))\,.$$

Proof. The continuation equation transports nonemptiness of r to nonemptiness of $\mathsf{append}(h, k)$. The append nonempty cycle-preimage theorem then separates the witness to the left boundary preimage or to the right boundary preimage, carrying the corresponding cycle evidence supplied by square-zero. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont nonempty cycle preimage`

Theorem 98.24 (Homology boundary carrier continuation cycle components)**.** *Assume the differential respects BEDC continuation append and is square-zero. If $B_d(r)$ and $\mathsf{Cont}(h, k, r)$, then both continuation components are cycles:*

$$Z_d(h) \wedge Z_d(k).$$

Proof. The square-zero law turns the boundary carrier on r into a cycle carrier on the continuation result. Since $\mathsf{Cont}(h, k, r)$ identifies r with $\mathsf{append}(h, k)$, the append-components theorem splits that cycle predicate into the two component cycle predicates. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cont cycle components`

Theorem 98.25 (Homology boundary carrier cycle from square-zero differential)**.** *Assume the differential respects* hsame *and satisfies the square-zero condition*

$$\mathsf{hsame}(d(d(u)), \mathsf{Empty})$$

for every $u : \mathsf{BHist}$. If $B_d(h)$, then $Z_d(h)$.

Proof. Choose a boundary witness u with $\mathsf{hsame}(h, d(u))$. Congruence for the differential transports this witness to $\mathsf{hsame}(d(h), d(d(u)))$. The square-zero row identifies $d(d(u))$ with Empty, and transitivity of hsame gives the cycle predicate. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier cycle of d squared zero`

Theorem 98.26 (Homology cycle carrier append hsame transport)**.** *Assume the differential respects BEDC continuation append and is congruent for* hsame*. If $Z_d(h)$, $Z_d(k)$, and $\mathsf{hsame}(\mathsf{append}(h, k), r)$, then $Z_d(r)$.*

Proof. The append-closure theorem gives $Z_d(\mathsf{append}(h, k))$. Congruence for the differential transports the endpoint comparison to an hsame comparison between $d(\mathsf{append}(h, k))$ and $d(r)$. Symmetry of that comparison, followed by the cycle row for the append, proves that $d(r)$ is hsame to Empty. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier append hsame transport`

Theorem 98.27 (Homology cycle-boundary append cycle closure)**.** *Assume the differential respects BEDC continuation append and is square-zero. If $Z_d(h)$ and $B_d(k)$, then $Z_d(\mathsf{append}(h, k))$.*

Proof. The square-zero law turns the boundary hypothesis on k into a cycle hypothesis. The ordinary cycle append closure then applies to the two cycle histories h and k. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleBoundary append cycle closed`

Theorem 98.28 (Homology boundary-cycle append cycle closure)**.** *Assume the differential respects BEDC continuation append and is square-zero. If $B_d(h)$ and $Z_d(k)$, then $Z_d(\mathsf{append}(h, k))$.*

Proof. The square-zero law turns the boundary hypothesis on h into a cycle hypothesis. The ordinary cycle append closure then applies to the two cycle histories h and k. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCycle append cycle closed`

Theorem 98.29 (Homology boundary-cycle append hsame transport)**.** *Assume the differential respects BEDC continuation append, is square-zero, and respects* hsame*. If $B_d(h)$, $Z_d(k)$, and* $\mathsf{hsame}(\mathsf{append}(h,k),r)$*, then $Z_d(r)$.*

Proof. The boundary-cycle append closure theorem gives a cycle at $\mathsf{append}(h,k)$. Differential congruence transports the final endpoint comparison to $d(r)$, and symmetry followed by transitivity gives $\mathsf{hsame}(d(r),\mathsf{Empty})$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCycle append hsame transport`

Theorem 98.30 (Homology cycle-boundary append hsame transport)**.** *Assume the differential respects BEDC continuation append, is square-zero, and respects* hsame*. If $Z_d(h)$, $B_d(k)$, and* $\mathsf{hsame}(\mathsf{append}(h,k),r)$*, then $Z_d(r)$.*

Proof. The square-zero row turns the boundary hypothesis on k into a cycle hypothesis. Cycle-boundary append closure therefore gives a cycle at $\mathsf{append}(h,k)$. Differential congruence transports the final endpoint comparison to $d(r)$, and symmetry followed by transitivity gives $\mathsf{hsame}(d(r),\mathsf{Empty})$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleBoundary append hsame transport`

Theorem 98.31 (Homology cycle-boundary append cycle hsame transport)**.** *Assume the differential respects BEDC continuation append, is congruent for* hsame*, and is square-zero. If $Z_d(h)$, $B_d(k)$, and* $\mathsf{hsame}(\mathsf{append}(h,k),r)$*, then $Z_d(r)$.*

Proof. The square-zero and congruence laws turn the boundary hypothesis on k into a cycle hypothesis. The append endpoint transport theorem then applies to the cycle histories h and k and transports the resulting cycle predicate across the displayed endpoint comparison. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleBoundary append cycle hsame transport`

98.2 The certificate

Theorem 98.32 (Homology boundary carrier semantic name certificate)**.** *For any BEDC differential $d : \mathsf{BHist} \to \mathsf{BHist}$, the boundary carrier B_d forms a semantic naming certificate when source, pattern, and ledger policy are all B_d, and the classifier requires both endpoints to be boundaries together with* hsame *between them.*

Proof. The carrier is inhabited by $d(\mathsf{Empty})$, using Empty as its preimage and reflexivity of hsame. The classifier equivalence laws come from the stored endpoint boundary witnesses and the reflexive, symmetric, and transitive laws for hsame. Carrier transport along a classifier edge is the right endpoint boundary witness, and the semantic pattern and ledger fields are the source witness itself. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyBoundaryCarrier semanticNameCert`

Theorem 98.33 (Homology cycle carrier empty semantic name certificate)**.** *If the empty history is a d-cycle, then the cycle carrier Z_d forms a semantic naming certificate when source, pattern, and ledger policy are all Z_d, and the classifier requires both endpoints to be cycles together with* hsame *between them.*

Proof. The empty-cycle hypothesis inhabits the carrier at Empty. The classifier equivalence laws again follow by unpacking the two endpoint cycle witnesses and composing the underlying hsame rows. Carrier transport is read from the right endpoint cycle witness already stored in the classifier, and the semantic pattern and ledger fields are identities on the source witness. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier empty semanticNameCert`

Theorem 98.34 (Homology singleton cycle semantic name certificate)**.** *Let the singleton cycle carrier contain exactly the BEDC histories h*hsameBHist.Empty*. Let its classifier require both endpoints to lie in that carrier and require h*hsame*k. This classifier is a semantic naming certificate for the carrier used simultaneously as source, pattern, and ledger policy.*

Proof. The carrier is inhabited by BHist.Empty through reflexivity of hsame. Classifier reflexivity, symmetry, and transitivity are obtained by unpacking the two endpoint-carrier witnesses and composing the underlying hsame proofs. Carrier transport along a classifier edge is the right endpoint carrier witness already stored in the classifier. The semantic pattern and ledger fields are the source witness itself. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologySingletonCycle semanticNameCert`

Theorem 98.35 (Homology singleton cycle standard bridge boundary)**.** *The singleton-cycle semantic certificate of Definition 98.34 supplies the paper-side standard bridge boundary for the exported* Homology$^{\uparrow}$ *cycle interface. The BEDC source is* BHist *with* hsame*, the carrier is the singleton cycle carrier, the classifier is endpoint carriage together with* hsame*, the ledger is the source witness, and the standard readback is the one-cycle trivial homology object.*

Proof. Project the semantic certificate of Definition 98.34. Its source, pattern, and ledger policy are all the singleton cycle carrier, and its classifier stores the two endpoint-carrier witnesses together with the hsame comparison.

Read the source as BHist equipped with hsame. The carrier is inhabited by BHist.Empty, so the standard readback is the one-cycle object whose only accepted cycle endpoint is the singleton empty-history class.

The ledger field is the source witness itself, and transport along the classifier uses the right endpoint-carrier witness. Repacking these rows gives the standard bridge boundary for the exported singleton-cycle Homology$^{\uparrow}$ interface. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyUp StdBridge`

Closure status for NameCert$_{\text{Homology}^{\uparrow}}$**.**

Constructive story: The exported homology surface is the singleton cycle cut out of a displayed chain differential: a history is carried when it is history-same to BHist.Empty, while the surrounding cycle and boundary rows explain how general predicates $Z_d(h)$ and $B_d(h)$ would be read through differential endpoints and preimages. The carrier therefore accepts the empty cycle and excludes nonempty endpoints for this exported object; its classifier stores the two endpoint-carrier witnesses together with hsame(h, k), rather than forming a boundary quotient or homology class. The source, pattern, and ledger policy are all the singleton cycle predicate, with the wider pattern rows recording boundary-to-cycle closure under square-zero, append closure, continuation preimages, and transport along hsame. The five name-certificate obligations are immediate on this surface: BHist.Empty inhabits the carrier, hsame gives reflexivity, symmetry, and transitivity, and classifier transport returns the right endpoint's stored carrier witness. A history h lies in the signature gap of a Homology$^{\uparrow}$

token p when the source-domain accepts h as the singleton cycle, the signature relation pairs it with an answer cycle and the source witness as evidence ledger, and token introduction names that answer as p; unfolded, h is history-same to the empty history and matches the one-cycle endpoint encoded by p. The paper bridge reads this singleton certificate as the trivial one-cycle homology object, with the ledger still being the same source witness and without exporting a quotient by boundaries. The chapter registers this singleton-cycle construction under the name Homology$^{\uparrow}$; the homology interface here is a checked kernel/image boundary over histories, with no arbitrary chain complex library, boundary quotient carrier, or derived-functor primitive introduced.

Theory closure: TheoryGate(NameCert$_{\text{Homology}\uparrow}$, maturePackageClosed)

Scope closed: The mature surface binds the singleton-cycle semantic certificate, cycle and boundary carrier readbacks, continuation preimage package, append and Cont transport rows, and the final hsame transport theorem.

Formal verification: FormalStatus(NameCert$_{\text{Homology}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.HomologyUp.HomologySingletonCycle\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: General homology theory: arbitrary chain complexes, boundary/image quotients, chain maps, exact sequences, long exact sequences, singular or simplicial homology, and derived functors.

Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 98.36 (Homology cycle carrier hsame transport)**.** *Assume the differential is stable under history-sameness:*

$$\forall u, v : \mathsf{BHist},\ \mathsf{hsame}(u, v) \to \mathsf{hsame}(d(u), d(v)).$$

If $Z_d(h)$ and $\mathsf{hsame}(h, k)$, then $Z_d(k)$.

Proof. By section 98.1, $Z_d(h)$ is the endpoint comparison $\mathsf{hsame}(d(h), \mathsf{Empty})$. Applying the displayed stability row to $\mathsf{hsame}(h, k)$ gives $\mathsf{hsame}(d(h), d(k))$. Symmetry of hsame gives $\mathsf{hsame}(d(k), d(h))$, and transitivity with the cycle endpoint gives $\mathsf{hsame}(d(k), \mathsf{Empty})$. By the same carrier definition, this is $Z_d(k)$. □

Lean4 ✓ `BEDC.Derived.HomologyUp.HomologyCycleCarrier hsame transport`

Chapter 99

A Concrete Naming Certificate for Cohomology$^{\uparrow}$

The cohomology interface $\mathsf{Cohomology}^{\uparrow}$ packages a cochain complex with the dual kernel/image quotients computing cohomology groups. It depends on $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ (chapter 98) and supplies the structural substrate for de Rham, sheaf, and Čech cohomology certificates.

99.1 Carrier definition

Theorem 99.1 (Cohomology cocycle axis right cancel)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If the shifted history $h \cdot axis$ is a d-cocycle, then h is a d-cocycle.*

Proof. Append preservation rewrites the shifted differential as $\mathsf{append}(d(h), d(axis))$. BEDC append-empty inversion reads the left component back from the empty appended history, giving $d(h)\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis right cancel`

Theorem 99.2 (Cohomology cocycle axis left cancel)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If the shifted history $axis \cdot h$ is a d-cocycle, then h is a d-cocycle.*

Proof. Append preservation rewrites the shifted differential as $\mathsf{append}(d(axis), d(h))$. BEDC append-empty inversion reads the right component back from the empty appended history, giving $d(h)\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis left cancel`

Theorem 99.3 (Cohomology cocycle axis context cancel)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If the two-sided shifted history $left \cdot (h \cdot right)$ is a d-cocycle, then h is a d-cocycle.*

Proof. Apply left-axis cancellation to the outer context. This recovers the right-axis shifted cocycle for $h \cdot right$. Applying right-axis cancellation to that shifted cocycle gives $d(h)\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis context cancel`

Theorem 99.4 (Cohomology cocycle bilateral axis cancel)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If the doubly shifted history $(leftAxis \cdot h) \cdot rightAxis$ is a d-cocycle, then h is a d-cocycle.*

Proof. Right-axis cancellation removes the trailing $rightAxis$ from the displayed cocycle and yields a cocycle on $leftAxis \cdot h$. Left-axis cancellation then removes the leading $leftAxis$ and gives the core cocycle condition on h. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis bilateral cancel`

Theorem 99.5 (Cohomology cocycle append closure)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*, and let the fixed cohomology axis satisfy $d(axis)\mathsf{hsamee}$. If the shifted histories $h \cdot axis$ and $k \cdot axis$ are d-cocycles, then the shifted append $(h \cdot k) \cdot axis$ is also a d-cocycle.*

Proof. Apply append preservation to the two shifted cocycle hypotheses. Append-empty inversion gives $d(h)\mathsf{hsamee}$ and $d(k)\mathsf{hsamee}$, since the axis derivative is empty. Append preservation for $h \cdot k$ then makes $d(h \cdot k)$ empty, and one final append-preservation step with the same axis gives the displayed shifted cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append closed`

Theorem 99.6 (Cohomology cocycle prepend axis closure)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*, and let the fixed cohomology axis satisfy $d(axis)\mathsf{hsamee}$. If the shifted histories $axis \cdot h$ and $axis \cdot k$ are d-cocycles, then the shifted append $axis \cdot (h \cdot k)$ is also a d-cocycle.*

Proof. Append preservation rewrites the two shifted cocycle hypotheses as empty append histories $\mathsf{append}(d(axis), d(h))$ and $\mathsf{append}(d(axis), d(k))$. Append-empty inversion extracts $d(h)\mathsf{hsamee}$ and $d(k)\mathsf{hsamee}$. The core append closure makes $d(h \cdot k)$ empty, and the axis cocycle row then rebuilds the displayed left-axis cocycle through the append law. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle prepend axis closed`

Theorem 99.7 (Cohomology cocycle append hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*, and let the fixed cohomology axis satisfy $d(axis)\mathsf{hsamee}$. If the shifted histories $h \cdot axis$ and $k \cdot axis$ are d-cocycles and $\mathsf{hsame}((h \cdot k) \cdot axis, r)$, then r is a d-cocycle.*

Proof. The append closure theorem gives the cocycle predicate for $(h \cdot k) \cdot axis$. Congruence for d transports the endpoint comparison to the corresponding differential histories, and symmetry plus transitivity of hsame carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append hsame transport`

Theorem 99.8 (Cohomology cocycle mixed-axis append hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*. If $h \cdot axis$ and $axis \cdot k$ are d-cocycles, and if $h \cdot k\mathsf{hsame}r$, then r is a d-cocycle.*

Proof. Right-axis cancellation extracts the core cocycle for h, and left-axis cancellation extracts the core cocycle for k. Core append closure then gives the cocycle for $h \cdot k$. Congruence for d transports this cocycle across the displayed hsame comparison to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle mixed axis append hsame transport`

Theorem 99.9 (Cohomology cocycle core append closure)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If h and k are d-cocycles in the core history sense, then their append $h \cdot k$ is also a d-cocycle.*

Proof. Convert the two core cocycle hypotheses to empty equalities for $d(h)$ and $d(k)$. Append-empty introduction makes $d(h) \cdot d(k)$ empty. Append preservation transports this empty result back to $d(h \cdot k)$. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append core closed`

Theorem 99.10 (Cohomology cocycle continuation hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*. If* $\mathsf{Cont}(h, k, r)$*, the histories h and k are d-cocycles, and* $\mathsf{hsame}(r, s)$*, then s is a d-cocycle.*

Proof. Continuation identifies r with $h \cdot k$. Core append closure gives $d(h \cdot k)\mathsf{hsamee}$ from the two endpoint cocycle hypotheses. The $\mathsf{hsame}(r, s)$ comparison is transported through d, and symmetry plus transitivity carries the cocycle witness to s. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation hsame transport`

Theorem 99.11 (Cohomology cocycle continuation context cancel)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and respect* hsame *on histories. If* $\mathsf{Cont}(left, h, mid)$ *and* $\mathsf{Cont}(mid, right, r)$ *compose a two-sided context around h, then any d-cocycle on r restricts to a d-cocycle on h.*

Proof. The two continuation ledgers identify r with $(left \cdot h) \cdot right$. Associativity rewrites this as $left \cdot (h \cdot right)$, and congruence for d transports the cocycle witness to that reassociated context. Axis context cancellation then removes the visible left and right factors. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation context cancel`

Theorem 99.12 (Cohomology cocycle continuation context empty iff)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If* $\mathsf{Cont}(left, h, mid)$ *and* $\mathsf{Cont}(mid, right, r)$ *compose a two-sided continuation context, then r is a d-cocycle exactly when the three exposed components $left$, h, and $right$ are d-cocycles:*

$$d(r)\mathsf{hsamee} \iff d(left)\mathsf{hsamee} \ \wedge \ d(h)\mathsf{hsamee} \ \wedge \ d(right)\mathsf{hsamee}.$$

Proof. The two continuation ledgers identify r with $(left \cdot h) \cdot right$. Append-empty equivalence first splits the outer append into the cocycle on $left \cdot h$ and the cocycle on $right$, and then splits $left \cdot h$ into the two cocycles on $left$ and h. Conversely, core append closure combines $left$ with h and then combines that result with $right$. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation context empty iff`

Theorem 99.13 (Cohomology cocycle continuation context e1 boundary empty absurd)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*. If* $\mathsf{Cont}(left, h, mid)$ *and* $\mathsf{Cont}(mid, right, r)$ *form a two-sided continuation context, any visible* E1 *boundary among $d(left)$, $d(h)$, and $d(right)$ contradicts $d(r)\mathsf{hsamee}$.*

Proof. The continuation-context empty equivalence decomposes $d(r)\mathsf{hsamee}$ into empty witnesses for $d(left)$, $d(h)$, and $d(right)$. In each visible boundary case, symmetry and transitivity give an E1 history hsamee, forbidden by constructor disjointness. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation context e1 boundary empty absurd`

Theorem 99.14 (Cohomology cocycle core append hsame transport)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame *and respect* hsame *on histories. If* h *and* k *are core* d*-cocycles and* $\mathsf{hsame}(h \cdot k, r)$*, then* r *is a* d*-cocycle.*

Proof. Core append closure first gives $d(h \cdot k)\mathsf{hsamee}$. The displayed endpoint comparison transports through d, and symmetry followed by transitivity carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle core append hsame transport`

Theorem 99.15 (Cohomology cocycle append empty equivalence)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. For histories* $h, k : \mathsf{BHist}$*, the appended history* $h \cdot k$ *is a* d*-cocycle exactly when both* h *and* k *are* d*-cocycles:*

$$d(h \cdot k)\mathsf{hsamee} \iff d(h)\mathsf{hsamee} \ \wedge \ d(k)\mathsf{hsamee}.$$

Proof. Append preservation transports the displayed cocycle on $h \cdot k$ to the empty appended history $\mathsf{append}(d(h), d(k))$, and BEDC append-empty inversion splits this into the two component cocycle equations. Conversely, the core append closure theorem combines the two component cocycles into the appended cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append empty iff`

Theorem 99.16 (Cohomology cocycle continuation axis context cancel)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. If* $\mathsf{Cont}(h, k, r)$ *and the two-sided contextual history* $left \cdot (r \cdot right)$ *is a* d*-cocycle, then both* h *and* k *are* d*-cocycles:*

$$d(h)\mathsf{hsamee} \ \wedge \ d(k)\mathsf{hsamee}.$$

Proof. The continuation witness identifies r with $h \cdot k$. Axis-context cancellation removes the surrounding histories from the contextual cocycle and gives $d(h \cdot k)\mathsf{hsamee}$. Append-empty equivalence then splits this core cocycle into the two component cocycle rows for h and k. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation axis context cancel`

Theorem 99.17 (Cohomology cocycle continuation axis context closure)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame *and respect* hsame*. If* $\mathsf{Cont}(h, k, r)$ *and* $left$*,* h*,* k*, and* $right$ *are* d*-cocycles, then the two-sided contextual continuation history* $left \cdot (r \cdot right)$ *is a* d*-cocycle.*

Proof. Core append closure first combines the cocycles on h and k. Congruence for d transports the continuation comparison to a cocycle on r. Applying core append closure to r with $right$, and then to $left$ with that result, gives the displayed two-sided contextual cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle continuation axis context closed`

Theorem 99.18 (Cohomology visible left boundary excludes appended cycle)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. If* $d(h)$ *has visible boundary* $\mathsf{E1}(t)$*, then the appended history* $h \cdot k$ *cannot be a* d*-cocycle:*

$$d(h)\mathsf{hsameE1}(t) \Longrightarrow d(h \cdot k)\mathsf{hsamee} \Longrightarrow \bot.$$

Proof. Append preservation transports the appended cocycle to $\mathsf{append}(d(h), d(k))\mathsf{hsamee}$. Append-empty inversion then forces $d(h)\mathsf{hsamee}$. Together with the visible boundary hypothesis this identifies $\mathsf{E1}(t)$ with e, contradicting BEDC constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append left e1 boundary empty absurd`

Theorem 99.19 (Cohomology visible left boundary excludes appended cycle)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. If* $d(h)$ *is* hsame *to either* $\mathsf{E0}(t)$ *or* $\mathsf{E1}(t)$*, then* $d(h \cdot k)\mathsf{hsamee}$ *is impossible.*

Proof. Append preservation and append-empty inversion force $d(h)\mathsf{hsamee}$; either visible constructor case contradicts BEDC constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append left visible boundary empty absurd`

Theorem 99.20 (Cohomology visible right boundary excludes appended cycle)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. If* $d(k)$ *has visible boundary* $\mathsf{E1}(t)$*, then the appended history* $h \cdot k$ *cannot be a* d*-cocycle:*

$$d(k)\mathsf{hsameE1}(t) \Longrightarrow d(h \cdot k)\mathsf{hsamee} \Longrightarrow \bot.$$

Proof. Append preservation transports the appended cocycle to $\mathsf{append}(d(h), d(k))\mathsf{hsamee}$. Append-empty inversion then forces $d(k)\mathsf{hsamee}$. Together with the visible boundary hypothesis this identifies $\mathsf{E1}(t)$ with e, contradicting BEDC constructor no-confusion. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append right e1 boundary empty absurd`

Theorem 99.21 (Cohomology cocycle left-shift append empty equivalence)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. For histories* $axis, h, k : \mathsf{BHist}$*, the left-shifted appended history* $axis \cdot (h \cdot k)$ *is a* d*-cocycle exactly when the axis and both tail components are* d*-cocycles:*

$$d(axis \cdot (h \cdot k))\mathsf{hsamee} \quad \Longleftrightarrow \quad d(axis)\mathsf{hsamee} \ \wedge \ d(h)\mathsf{hsamee} \ \wedge \ d(k)\mathsf{hsamee}.$$

Proof. Apply append-empty equivalence to split the left axis from the tail $h \cdot k$. Apply the same equivalence again to split the tail into h and k. The converse first rebuilds the tail cocycle and then rebuilds the left-shifted cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle left shift append empty iff`

Theorem 99.22 (Cohomology cocycle core append transport)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame *and be congruent for* hsame*. If* h *and* k *are core* d*-cocycles and* $h \cdot k\mathsf{hsame}r$*, then* r *is also a* d*-cocycle.*

Proof. Core append closure gives $d(h \cdot k)\mathsf{hsamee}$. Congruence for d transports the comparison $h \cdot k\mathsf{hsame}r$ to the differential histories, and symmetry plus transitivity carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle append core hsame transport`

Theorem 99.23 (Cohomology cocycle axis context append closure)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*, and let* $left$ *and* $right$ *be* d*-cocycles. If the two-sided shifted histories* $left \cdot (h \cdot right)$ *and* $left \cdot (k \cdot right)$ *are* d*-cocycles, then the two-sided shifted append* $left \cdot ((h \cdot k) \cdot right)$ *is also a* d*-cocycle.*

Proof. Two-sided context cancellation applied to the two hypotheses gives core cocycles for h and k. Core append closure makes $h \cdot k$ a core cocycle. Append preservation with the right context, then with the left context, combines these three cocycle rows into the displayed two-sided shifted cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis context append closed`

Theorem 99.24 (Cohomology cocycle axis context append hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*, and let $left$ and $right$ be d-cocycles. If $left \cdot (h \cdot right)$ and $left \cdot (k \cdot right)$ are d-cocycles and*

$$\mathsf{hsame}(left \cdot ((h \cdot k) \cdot right), r),$$

then r is a d-cocycle.

Proof. The two-sided context append closure theorem gives the cocycle predicate for $left \cdot ((h \cdot k) \cdot right)$. Congruence for d transports the final hsame comparison to the differential histories, and symmetry followed by transitivity carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle axis context append hsame transport`

Theorem 99.25 (Cohomology cocycle left-shift append closure)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame*, and let the fixed cohomology axis satisfy $d(axis)$*hsamee*. If the shifted histories $axis \cdot h$ and $axis \cdot k$ are d-cocycles, then the shifted append $axis \cdot (h \cdot k)$ is also a d-cocycle.*

Proof. Apply left-axis cancellation to the two shifted cocycle hypotheses to obtain core cocycles for h and k. Core append closure gives a cocycle for $h \cdot k$. Append preservation for $axis \cdot (h \cdot k)$ then combines the empty axis differential with this core cocycle and yields the displayed left-shifted cocycle. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle left shift append closed`

Theorem 99.26 (Cohomology cocycle left-shift append hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*, and let the fixed cohomology axis satisfy $d(axis)$*hsamee*. If the shifted histories $axis \cdot h$ and $axis \cdot k$ are d-cocycles and* hsame$(axis \cdot (h \cdot k), r)$*, then r is a d-cocycle.*

Proof. The left-shift append closure theorem gives the cocycle predicate for $axis \cdot (h \cdot k)$. Congruence for d transports the displayed endpoint comparison to the differential histories, and symmetry followed by transitivity of hsame carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle left shift append hsame transport`

Theorem 99.27 (Cohomology cocycle prepend hsame transport)**.** *Let $d : \mathsf{BHist} \to \mathsf{BHist}$ preserve append up to* hsame *and be congruent for* hsame*, and let the fixed cohomology axis satisfy $d(axis)$*hsamee*. If the shifted histories $axis \cdot h$ and $axis \cdot k$ are d-cocycles and* hsame$(axis \cdot (h \cdot k), r)$*, then r is a d-cocycle.*

Proof. The left-shift append closure theorem gives the cocycle predicate for $axis \cdot (h \cdot k)$. Congruence for d transports the endpoint comparison to differentials, and symmetry plus transitivity of hsame carries the cocycle predicate to r. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle prepend hsame transport`

99.2 The certificate

Definition 99.28 (Cohomology cocycle predicate)**.** *For a BEDC cohomology differential* $d : \mathsf{BHist} \to \mathsf{BHist}$*, write*

$$C_d(h) :\Longleftrightarrow d(h)\mathsf{hsamee}.$$

Lean4 (definition) `BEDC.Derived.CohomologyUp.CohomologyCocycle`

Lemma 99.29 (Cohomology cocycle hsame transport)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *respect* hsame*. If* $C_d(h)$ *and* $\mathsf{hsame}(h, k)$*, then* $C_d(k)$*.*

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle hsame transport`

Proof. By the respect-hsame hypothesis on d, $\mathsf{hsame}(h, k)$ implies $\mathsf{hsame}(d(h), d(k))$. Symmetry of that comparison, followed by the cocycle row $d(h)\mathsf{hsamee}$, gives $d(k)\mathsf{hsamee}$. □

Theorem 99.30 (Cohomology cocycle predicate semantic name certificate)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame *and respect* hsame*. Fix* $axis : \mathsf{BHist}$ *with* $C_d(axis)$*. Then the cocycle predicate* C_d*, used simultaneously as source, pattern, and ledger, with classifier* hsame*, carries a semantic naming certificate:*

$$\mathsf{SemanticNameCert}(C_d, C_d, C_d, \mathsf{hsame}).$$

Its cohomology stability witnesses are the bilateral axis-cancel row of Definition 99.4 and the append-closure row of Definition 99.5.

Proof. The source is inhabited by the chosen axis cocycle. The classifier fields in Definition 9.7 are the finite-kernel reflexivity, symmetry, and transitivity laws for hsame. Carrier transport for the core certificate is Definition 99.29. Therefore C_d with classifier hsame forms the core $\mathsf{NameCert}(C_d, \mathsf{hsame})$.

Since source, pattern, and ledger are all the same predicate C_d, both semantic soundness maps are identity maps on a source witness. The stability component from Definition 9.4 is supplied within the same predicate: Definition 99.4 descends a two-sided axis-shifted cocycle to the unshifted cocycle, and Definition 99.5 composes two shifted cocycle witnesses along a cocycle axis. These fields populate the five roles of Definition 9.6 with source C_d, pattern C_d, ledger C_d, and classifier hsame. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle predicate semanticNameCert`

Theorem 99.31 (Cohomology cocycle semantic name certificate)**.** *Let* $d : \mathsf{BHist} \to \mathsf{BHist}$ *preserve append up to* hsame*. The cocycle predicate* $h \mapsto d(h)\mathsf{hsamee}$*, used simultaneously as source, pattern, and ledger policy, forms a semantic naming certificate whose classifier compares histories by* $d(h)\mathsf{hsame}d(k)$*.*

Proof. Append preservation at e, e identifies $d(\mathsf{e})$ with $\mathsf{append}(d(\mathsf{e}), d(\mathsf{e}))$. BEDC right-unit exactness for append therefore forces $d(\mathsf{e})\mathsf{hsamee}$, giving the carrier witness. The classifier fields are reflexivity, symmetry, and transitivity of hsame on the displayed differential histories. Carrier transport composes the symmetric classifier witness with the source cocycle predicate, and the pattern and ledger policies are the same cocycle predicate. □

Lean4 ✓ `BEDC.Derived.CohomologyUp.CohomologyCocycle semanticNameCert`

Closure status for $\mathsf{NameCert}_{\mathsf{Cohomology}\uparrow}$.

Constructive story: A cohomology cocycle is generated by a displayed cochain differential d on histories: a history h is admitted exactly when the differential endpoint $d(h)$ is history-same to e. Histories whose differential does not collapse to the empty endpoint are excluded, and the classifier compares admitted histories by the differential endpoints, asking whether $d(h)$ and $d(k)$ are hsame rather than quotienting by coboundaries. The source, pattern, and ledger policy are all the cocycle predicate, with the pattern rows spelling out right, left, and bilateral axis cancellation, append closure along a cocycle axis, and transport through continuation and hsame contexts. The name-certificate obligations are supplied by the append-preservation proof that makes e a cocycle, by reflexivity, symmetry, and transitivity of hsame on displayed differential histories, and by carrier transport that composes the classifier witness with the source cocycle witness. A history h lies in the signature gap of a Cohomology$^{\uparrow}$ token p when the source-domain accepts h as a cocycle, the signature relation pairs it with an answer cocycle and evidence ledger, and token introduction names that answer as p; concretely, the differential of h is empty and its differential endpoint is classified with the endpoint encoded by p. The paper bridge boundary keeps this as a cocycle predicate over the displayed cochain differential, not as an external coboundary quotient or cohomology group. The chapter registers this cocycle-predicate construction under the name Cohomology$^{\uparrow}$; all visible information remains in differential endpoints, axis rows, and ledgers, with no cup product, sheaf complex, spectral sequence, or host quotient primitive introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cohomology}\uparrow}$, maturePackageClosed)

Scope closed: The public cocycle SemanticNameCert over the displayed cochain differential is exported with carrier, classifier transport, and pattern/ledger policies equal to the cocycle predicate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cohomology}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CohomologyUp.CohomologyCocycle\_semanticNameCert`

Bridge status: paperBridge

Not claimed: General cohomology theory: arbitrary cochain complexes, coboundary quotients, cup products, de Rham cohomology, sheaf cohomology, Čech cohomology, or spectral sequences.

Upgrade path: No further theory-axis upgrade is claimed in this chapter; any bridgeChecked uplift is a separate verification-axis step.

Chapter 100

A Concrete Naming Certificate for $\mathsf{Padic}^{\uparrow}$

The p-adic interface $\mathsf{Padic}^{\uparrow}$ packages the carrier of regular Cauchy sequences over $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ (chapter 37) under the p-adic absolute value, indexed by a prime p from $\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$ (chapter 64). It is the non-archimedean sibling of $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38).

100.1 Carrier definition

Definition 100.1 (Prime scale over unary multiplication)**.** *For histories p, e, r, the predicate* $\mathsf{PadicPrimeScale}(p, e, r)$ *holds when p is a* $\mathsf{Prime}^{\uparrow}$*-prime and r is the unary multiplication result of p by the exponent e.*

Lean4 (definition) `BEDC.Derived.PadicUp.PadicPrimeScale`

Theorem 100.2 (Prime scale unary components)**.** *If* $\mathsf{PadicPrimeScale}(p, e, r)$*, then the prime base p, exponent e, and result r are all unary histories.*

Proof. The scale predicate consists of a $\mathsf{NatPrime}(p)$ witness and a $\mathsf{NatMul}(p, e, r)$ witness. The prime witness supplies the unary carrier field for p. The unary multiplication witness supplies the unary field for e, and together with the unary field for p, supplies the unary field for the result r. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale unary components`

Theorem 100.3 (Prime scale commutes with the exponent)**.** *If p and q are unary histories, and* $\mathsf{PadicPrimeScale}(p, q, r)$ *and* $\mathsf{PadicPrimeScale}(q, p, s)$ *both hold, then r*hsame*s.*

Proof. Each prime-scale witness contains a concrete NatMul field. The commutativity theorem for unary multiplication applies to those two fields and identifies the two scale results up to hsame. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale prime exponent comm hsame`

Theorem 100.4 (Prime scale totality)**.** *If p is a* $\mathsf{Prime}^{\uparrow}$*-prime and e is a unary history, then some unary history r realizes* $\mathsf{PadicPrimeScale}(p, e, r)$*.*

Proof. The prime witness contains the unary carrier field for p. Unary multiplication totality applied to this carrier and to the exponent carrier produces a unary result r together with the $\mathsf{NatMul}(p, e, r)$ witness. Pairing the original prime witness with that multiplication witness is precisely $\mathsf{PadicPrimeScale}(p, e, r)$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale total`

Theorem 100.5 (Prime scale exists uniquely)**.** *If p is a $\mathsf{Prime}^{\uparrow}$-prime and e is a unary history, then there is a history r with $\mathsf{PadicPrimeScale}(p, e, r)$, and every other such scale result is* hsame-*equal to r.*

Proof. Prime-scale totality supplies a concrete result witness. Any competing prime-scale result has the same prime base and exponent in its NatMul field, so unary multiplication functionality identifies the two result histories. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale exists unique`

Theorem 100.6 (Prime scale empty result exactness)**.** *If $\mathsf{PadicPrimeScale}(p, e, r)$, then r*hsamee *exactly when e*hsamee*.*

Proof. The forward direction uses the $\mathsf{Prime}^{\uparrow}$ prime field to rule out $p$hsamee, then applies the empty-result exactness theorem for NatMul with a nonempty multiplicand. The reverse direction identifies the exponent with e, so the zero constructor of NatMul identifies the result with e. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale empty result iff empty exponent`

Lemma 100.7 (Prime scale successor exponent inversion)**.** *If $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(q), r)$, then there is a predecessor result n with $\mathsf{PadicPrimeScale}(p, q, n)$ and $\mathsf{Cont}(n, p, r)$.*

Proof. The scale witness contains the prime field for p and a multiplication witness $\mathsf{NatMul}(p, \mathsf{E1}(q), r)$. Successor inversion for unary multiplication extracts the predecessor product n and the continuation $\mathsf{Cont}(n, p, r)$. Reusing the same prime field packages the predecessor as a prime-scale witness. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale succ exponent inversion`

Theorem 100.8 (Prime scale positive-exponent continuation readback)**.** *If the exponent e is unary, not history-same as* e*, and $\mathsf{PadicPrimeScale}(p, e, r)$ holds, then e has a unary tail t and the scale result is obtained from a predecessor scale by one continuation with the prime base:*

$$\exists t, n.\ \mathsf{Unary}(t) \wedge e = \mathsf{E1}(t) \wedge \mathsf{PadicPrimeScale}(p, t, n) \wedge \mathsf{Cont}(n, p, r).$$

Proof. The unary nonempty exponent has a one-headed tail. After substituting that tail into the scale witness, successor-exponent inversion for PadicPrimeScale exposes the predecessor scale and the continuation from the predecessor result by p. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale positive exponent cont readback`

Theorem 100.9 (Prime scale successor exponent factorization)**.** *A successor-exponent prime scale $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(q), r)$ holds exactly when some predecessor result n carries $\mathsf{PadicPrimeScale}(p, q, n)$ and a continuation $\mathsf{Cont}(n, p, r)$.*

Proof. The forward direction is successor-exponent inversion. Conversely, the predecessor scale gives the prime field and a concrete $\mathsf{NatMul}(p, q, n)$ witness; the successor constructor for NatMul, applied to $\mathsf{Cont}(n, p, r)$, rebuilds the scale over $\mathsf{E1}(q)$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale succ exponent factorization iff`

Lemma 100.10 (Prime scale exponent-result cases)**.** *Every* $\mathsf{PadicPrimeScale}(p, e, r)$ *witness is either the empty exponent case, with* $e\mathsf{hsame}e$ *and* $r\mathsf{hsame}e$*, or a successor exponent case: there are histories* e_0 *and* r_0 *such that* $e = \mathsf{E1}(e_0)$*,* $\mathsf{PadicPrimeScale}(p, e_0, r_0)$*, and* $\mathsf{Cont}(r_0, p, r)$*.*

Proof. Apply the exponent-result case split for the underlying $\mathsf{NatMul}(p, e, r)$ field. In the successor branch, the same prime witness for p packages the predecessor multiplication witness as a predecessor prime-scale witness. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale exponent result cases`

Theorem 100.11 (Prime scale successor predecessor uniqueness)**.** *If* $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(q), r)$*, then there is a predecessor result* n *with* $\mathsf{PadicPrimeScale}(p, q, n)$ *and* $\mathsf{Cont}(n, p, r)$*, and every other predecessor scale over* q *is history-same to* n*.*

Proof. Successor-exponent inversion supplies the predecessor result n, its prime-scale witness over q, and the continuation witness into r. Any competing predecessor scale over the same base p and exponent q has a second $\mathsf{NatMul}(p, q, -)$ field. Unary multiplication functionality, using the unary carrier field of the prime witness, identifies the two predecessor histories. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale succ exponent predecessor unique`

Theorem 100.12 (Prime scale unit exponent reads back to the prime)**.** *If* $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(\mathsf{e}), result)$ *holds, then the scale result is history-same to the prime base:*

$$result\mathsf{hsame}p.$$

Proof. Successor-exponent inversion extracts a predecessor result over the empty exponent and a continuation from that predecessor through p to *result*. Empty-result exactness for prime scales identifies the predecessor with e. Comparing the displayed continuation with the left-unit continuation for p identifies *result* with p. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale unit exponent result prime hsame`

Theorem 100.13 (Prime scale append continuation closure)**.** *If* $\mathsf{PadicPrimeScale}(p, w, n)$*,* $\mathsf{PadicPrimeScale}(p, q, e)$*, and* $\mathsf{Cont}(n, e, r)$*, then* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, q), r)$*.*

Proof. Both scale witnesses share the same prime field for p. The multiplication fields are concrete NatMul derivations, so append continuation closure for NatMul, applied to the supplied continuation witness, gives the scale witness for the appended exponent. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append cont closure`

Theorem 100.14 (Padic natural multiplication append exponent decomposition)**.** *For unary histories* d, w, q*, any multiplication witness over the appended exponent decomposes into multiplication witnesses over the two exponent factors and a continuation between their results:*

$$\mathsf{NatMul}(d, \mathsf{append}(w, q), r) \quad \Longrightarrow \quad \exists n, e.\ \mathsf{NatMul}(d, w, n) \wedge \mathsf{NatMul}(d, q, e) \wedge \mathsf{Cont}(n, e, r).$$

Proof. Induct on the unary right exponent q. For the empty exponent, take the left result to be the displayed result r and the right result to be e. In the one-headed case, successor inversion for NatMul extracts the previous product over $\mathsf{append}(w, q)$ and the final continuation by d. The induction hypothesis decomposes the previous product into left and right results; composing its continuation with the final continuation and using append associativity gives the required continuation from the left result to the successor right product. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicNatMul append exponent decomposition`

Theorem 100.15 (Prime scale append factorization)**.** *For unary exponent histories w, q, a prime scale over the appended exponent is exactly a pair of prime-scale factor results linked by continuation:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, q), r) \iff \exists n, e.\ \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(n, e, r).$$

Proof. The forward direction applies append-exponent decomposition to the NatMul field of the scale witness and reattaches the common prime field to both factor multiplications. The reverse direction is precisely prime-scale append continuation closure applied to the two factor scales and their continuation row. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append factorization iff`

Theorem 100.16 (Prime scale append exponent decomposition)**.** *For unary histories w, q, any prime scale over the appended exponent decomposes into prime-scale witnesses over the two exponent factors and a continuation between their results:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, q), r) \implies \exists n, e.\ \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(n, e, r).$$

Proof. The scale witness contains the shared prime field for p and a concrete $\mathsf{NatMul}(p, \mathsf{append}(w, q), r)$ field. Append-exponent decomposition for NatMul splits that multiplication field into factor multiplications and a continuation. Reusing the same prime field with each factor multiplication gives the two displayed prime-scale witnesses. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append exponent decomposition`

Theorem 100.17 (Prime scale append successor right factorization)**.** *For unary histories w and q, a prime scale over $\mathsf{append}(w, \mathsf{E1}(q))$ is exactly a left prime-scale result over w, a predecessor right prime-scale result over q, a one-step continuation by the prime base, and the final continuation from the left result:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(q)), r) \iff \exists n, e, s.\ \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(e, p, s)$$

Proof. The forward direction first applies append-exponent decomposition with the right factor $\mathsf{E1}(q)$. The right factor scale is then inverted by the successor-exponent factorization theorem, producing the predecessor scale over q and its continuation by p. Conversely, rebuild the right successor scale from the predecessor row and then apply append continuation closure with the final continuation row. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append succ right exponent factorization iff`

Theorem 100.18 (Prime scale append successor left factorization)**.** *For unary histories w and q, a prime scale over $\mathsf{append}(\mathsf{E1}(w), q)$ is exactly a predecessor left prime-scale result over w, a one-step continuation by the prime base, a right prime-scale result over q, and the final continuation from the successor left result:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(w), q), r) \iff \exists n, e, s.\ \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{Cont}(n, p, s) \wedge \mathsf{PadicPrimeScale}(p, q, e$$

Proof. Apply append-exponent decomposition to the scale over $\mathsf{append}(\mathsf{E1}(w), q)$. The left factor is then inverted by the successor-exponent factorization theorem, producing the predecessor scale over w and the continuation by p. Conversely, rebuild the left successor scale from that predecessor row and apply append continuation closure with the right scale and final continuation. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append succ left exponent factorization iff`

Theorem 100.19 (Prime scale append unit right factorization)**.** *For unary w, a prime scale over* $\mathsf{append}(w, \mathsf{E1}(\mathsf{e}))$ *is exactly a left prime-scale result over w followed by continuation with the prime base:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(\mathsf{e})), r) \iff \exists n.\, \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{Cont}(n, p, r).$$

Proof. The forward direction decomposes the appended exponent and then applies the unit exponent readback theorem, replacing the right factor result by the prime base. Conversely, the empty exponent scale gives the unit successor scale by the successor factorization theorem, and append continuation closure combines it with the left scale and the displayed continuation. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit right factorization iff`

Theorem 100.20 (Prime scale append unit right predecessor uniqueness)**.** *For unary w, every prime scale over* $\mathsf{append}(w, \mathsf{E1}(\mathsf{e}))$ *exposes a predecessor scale* $\mathsf{PadicPrimeScale}(p, w, n)$ *and a continuation* $\mathsf{Cont}(n, p, r)$*, and the predecessor is unique up to* hsame *among prime-scale results over w:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(\mathsf{e})), r) \implies \exists n.\, \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{Cont}(n, p, r) \wedge \forall other.\, \mathsf{PadicPrimeScale}(p,$$

Proof. The append-unit-right factorization extracts the predecessor scale and the continuation by the prime base. Uniqueness is the functional property of the underlying natural multiplication field of $\mathsf{PadicPrimeScale}(p, w, n)$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit right predecessor unique`

Theorem 100.21 (Prime scale append unit right result readback)**.** *For unary w, if n is a prime-scale result over w and r is a prime-scale result over* $\mathsf{append}(w, \mathsf{E1}(\mathsf{e}))$*, then r reads back to appending the prime base to n:*

$$\mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(\mathsf{e})), r) \Rightarrow r \mathsf{hsameappend}(n, p).$$

Proof. The append-unit-right predecessor theorem extracts a predecessor scale $\mathsf{PadicPrimeScale}(p, w, pred)$ with continuation $\mathsf{Cont}(pred, p, r)$. Predecessor uniqueness identifies $pred$ with the displayed result n up to hsame, and continuation respects that endpoint comparison against the canonical continuation $\mathsf{Cont}(n, p, \mathsf{append}(n, p))$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit right result hsame`

Theorem 100.22 (Prime scale append unit left factorization)**.** *For unary q, a prime scale over* $\mathsf{append}(\mathsf{E1}(\mathsf{e}), q)$ *is exactly a prime-scale result over q preceded by continuation from the prime base:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r) \iff \exists e.\, \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(p, e, r).$$

Proof. Decompose the appended exponent into a unit left factor and the displayed right factor. The unit-exponent readback identifies the left scale result with the prime base, leaving the continuation $\mathsf{Cont}(p, e, r)$. Conversely, construct the unit scale from the empty exponent scale and combine it with the displayed right scale by append continuation closure. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit left factorization iff`

Theorem 100.23 (Prime scale append unit left predecessor uniqueness)**.** *For unary q, every prime scale over* $\mathsf{append}(\mathsf{E1}(\mathsf{e}), q)$ *exposes a right-factor scale* $\mathsf{PadicPrimeScale}(p, q, e)$ *and a continuation* $\mathsf{Cont}(p, e, r)$*, and the right-factor result is unique up to* hsame *among prime-scale results over q:*

$$\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r) \Longrightarrow \exists e.\, \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(p, e, r) \wedge \forall other.\, \mathsf{PadicPrimeScale}(p, q,$$

Proof. The append-unit-left factorization extracts the right-factor scale and the continuation from the prime base. Uniqueness is the functional property of the underlying natural multiplication field of $\mathsf{PadicPrimeScale}(p, q, e)$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit left predecessor unique`

Theorem 100.24 (Prime scale append unit left result readback)**.** *For unary q, if* $\mathsf{PadicPrimeScale}(p, q, e)$ *and* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r)$ *hold, then the appended-exponent result reads back to the continuation result* $\mathsf{append}(p, e)$*:*

$$\mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r) \Rightarrow r\mathsf{hsameappend}(p, e).$$

Proof. The append-unit-left predecessor theorem extracts a right-factor result *pred* with $\mathsf{PadicPrimeScale}(p, q, pred)$ and continuation $\mathsf{Cont}(p, pred, r)$. The uniqueness clause identifies *pred* with the displayed scale result e. Continuation respect for hsame compares the extracted continuation with the canonical continuation $\mathsf{Cont}(p, e, \mathsf{append}(p, e))$. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit left result hsame`

Theorem 100.25 (Prime scale append unit left result nonempty)**.** *For unary q, if* $\mathsf{PadicPrimeScale}(p, q, e)$ *and* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r)$ *hold, then the appended unit-left scale result cannot be empty:*

$$\mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), r) \wedge r\mathsf{hsamee} \Longrightarrow \bot.$$

Proof. Read back the unit-left predecessor as a right-factor scale result *pred* and a continuation $\mathsf{Cont}(p, pred, r)$. If $r$hsamee, continuation empty-result inversion forces both p and *pred* to be empty. The uniqueness clause identifies *pred* with the displayed right-factor result e, and the empty-result iff for $\mathsf{PadicPrimeScale}(p, q, e)$ reads this back to the right exponent. The prime-base empty contradiction applied to p closes the case. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit left result not empty`

Lemma 100.26 (Prime scale append continuation result functional)**.** *Under the hypotheses of Definition 100.13, any other* PadicPrimeScale *result over the same appended exponent is history-same to the continuation result.*

Proof. The closure theorem constructs the prime-scale witness for the continuation result. Its multiplication field and the multiplication field of the other scale witness have the same prime base and appended exponent, so NatMul functionality identifies the two results. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append cont result functional`

Theorem 100.27 (Prime scale two unit exponents give prime square)**.** *If* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e})), r$ *holds, then the result is history-same to the square append of the prime base:*

$$result\mathsf{hsameappend}(p, p).$$

Proof. The unit-exponent scale theorem constructs $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(\mathsf{e}), p)$ for each of the two exponent factors. The concrete continuation $\mathsf{Cont}(p, p, \mathsf{append}(p, p))$ closes these two scales under append continuation. The resulting scale witness has the same prime base and appended exponent as the given witness, so NatMul functionality identifies the displayed results. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit exponents result square hsame`

Theorem 100.28 (Prime scale two unit exponents have nonempty result)**.** *If* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}$ *holds, then* $result\mathsf{hsamee}$ *is impossible.*

Proof. The square readback identifies $result$ with $\mathsf{append}(p, p)$. If $result\mathsf{hsamee}$, append empty inversion forces $p\mathsf{hsamee}$. But the prime witness contains a strict unary prefix from $\mathsf{E1}(\mathsf{e})$ to p; transporting the final continuation to the empty endpoint forces its nonempty tail to be empty, a contradiction. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append unit exponents result not empty`

Theorem 100.29 (Prime scale append factor results unique)**.** *Let* $\mathsf{PadicPrimeScale}(p, w, n)$ *and* $\mathsf{PadicPrimeScale}(p, q, e)$ *be linked by* $\mathsf{Cont}(n, e, r)$*. If* $\mathsf{PadicPrimeScale}(p, w, n_2)$ *and* $\mathsf{PadicPrimeScale}(p, q, e_2)$ *are linked by* $\mathsf{Cont}(n_2, e_2, r_2)$*, then the factor results and continuation results are history-same:*

$$n\mathsf{hsame}n_2 \wedge e\mathsf{hsame}e_2 \wedge r\mathsf{hsame}r_2.$$

Proof. Functionality of NatMul identifies the two results over the left factor exponent w and separately identifies the two results over the right factor exponent q. Continuation respect for hsame transports those two factor identifications across the displayed continuations and identifies the two composite results. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append factor results unique`

Theorem 100.30 (Prime scale append factor results commute)**.** *Let* $\mathsf{PadicPrimeScale}(p, w, n)$ *and* $\mathsf{PadicPrimeScale}(p, q, e)$ *hold. If the two factor results are linked by both continuation orders* $\mathsf{Cont}(n, e, r)$ *and* $\mathsf{Cont}(e, n, s)$*, then the continuation results satisfy* $r\mathsf{hsame}s$*.*

Proof. The scale witnesses expose unary carriers for the two factor results n and e. Unary append commutativity identifies $\mathsf{append}(n, e)$ with $\mathsf{append}(e, n)$, and the two displayed continuation ledgers identify r and s with those append histories. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append factor results comm hsame`

Theorem 100.31 (Prime scale append totality)**.** *If* p *is a* $\mathsf{Prime}^{\uparrow}$*-prime and* w, q *are unary histories, then there is a history* r *such that* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, q), r)$*. Moreover* r *is obtained from scale witnesses for* w *and* q *by a continuation witness.*

Proof. Totality for NatMul supplies histories n and e with $\mathsf{PadicPrimeScale}(p, w, n)$ and $\mathsf{PadicPrimeScale}(p, q, e)$, using the unary field of the prime witness for p. The concrete continuation $\mathsf{Cont}(n, e, \mathsf{append}(n, e))$ then feeds Definition 100.13, producing the scale witness for $\mathsf{append}(w, q)$ together with the displayed factor witnesses. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append total`

Theorem 100.32 (Prime scale appended empty result split)**.** *Under the hypotheses of Definition 100.13, the result history satisfies* $r\mathsf{hsamee}$ *exactly when both exponent factors satisfy* $w\mathsf{hsamee}$ *and* $q\mathsf{hsamee}$*.*

Proof. The closure theorem first supplies a prime-scale witness for $\mathsf{append}(w, q)$ with result r. Empty-result exactness for prime scales reduces $r\mathsf{hsamee}$ to $\mathsf{append}(w, q)\mathsf{hsamee}$, and append empty-result exactness splits that condition into the two factor witnesses. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append empty result empty factors iff`

Theorem 100.33 (Prime scale append factor empty alignment)**.** *Under the hypotheses of Definition 100.13, the two factor results and the composite continuation result are all empty exactly when the two exponent factors are empty:*

$$n\mathsf{hsamee} \wedge e\mathsf{hsamee} \wedge r\mathsf{hsamee} \iff w\mathsf{hsamee} \wedge q\mathsf{hsamee}.$$

Proof. Prime-scale empty-result exactness identifies $n\mathsf{hsamee}$ with $w\mathsf{hsamee}$ and $e\mathsf{hsamee}$ with $q\mathsf{hsamee}$. The appended empty-result split identifies $r\mathsf{hsamee}$ with the same two exponent factor conditions, so the three result-side empty witnesses align exactly with the two empty exponent factors. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append factor empty alignment iff`

Theorem 100.34 (Prime scale appended empty factor result readback)**.** *Under the hypotheses of Definition 100.13, an empty left exponent factor identifies the continuation result with the right factor result, and an empty right exponent factor identifies the continuation result with the left factor result.*

Proof. Prime-scale empty-result exactness turns the empty exponent factor into an empty factor result. The continuation ledger is then compared with the corresponding left-unit or right-unit continuation by Cont stability under hsame. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append empty factor result hsame`

Theorem 100.35 (Prime scale visible exponent nonempty result)**.** *Under the hypotheses of Definition 100.32, if the left exponent factor is* $e0(tail)$ *or* $e1(tail)$*, then the continuation result cannot satisfy* $r\mathsf{hsamee}$*.*

Proof. Assume $r\mathsf{hsamee}$. The appended empty-result split forces the left exponent factor to satisfy empty sameness. This contradicts the no-confusion theorem for $e0(tail)$ in the first case and for $e1(tail)$ in the second case. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append visible exponent result nonempty`

Theorem 100.36 (Prime scale visible right exponent nonempty result)**.** *Under the hypotheses of Definition 100.32, if the right exponent factor is* $e0(tail)$ *or* $e1(tail)$*, then the continuation result cannot satisfy* $r\mathsf{hsamee}$*.*

Proof. Assume $r\mathsf{hsamee}$. The appended empty-result split forces the right exponent factor to satisfy empty sameness. This contradicts the no-confusion theorem for $e0(tail)$ in the first case and for $e1(tail)$ in the second case. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale append visible right exponent result nonempty`

Theorem 100.37 (Prime scale visible exponent result nonempty)**.** *For histories* $p, result, tail$: BHist*, if* $\mathsf{PadicPrimeScale}(p, e0(tail), result)$ *or* $\mathsf{PadicPrimeScale}(p, e1(tail), result)$ *holds, then* $result\mathsf{hsamee}$ *is impossible.*

Proof. Empty-result exactness for prime scales transports $result\mathsf{hsamee}$ back to the visible exponent. The two cases contradict no-confusion for $e0(tail)$ and for $e1(tail)$ against e. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale visible exponent result nonempty`

Theorem 100.38 (Prime scale empty exponent result)**.** *If* $\mathsf{NatPrime}(p)$ *and* $e\mathsf{hsamee}$*, then* $\mathsf{PadicPrimeScale}(p, e, \mathsf{e})$*.*

Proof. Transport the exponent along $e\mathsf{hsamee}$. The zero constructor of NatMul multiplies the prime base by the empty exponent and returns e, while the same prime witness supplies the prime-scale carrier field. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale empty exponent result empty`

Theorem 100.39 (First prime scale at unit exponent)**.** *The first prime, the unit exponent, and the first-prime result form a concrete* $\mathsf{Padic}^{\uparrow}$ *prime scale:*

$$\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{E1}(\mathsf{e}))).$$

Proof. Pair the first-prime field from Definition 64.103 with the concrete unit-exponent multiplication witness from Definition 64.104. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale first prime unit exponent result`

100.2 The certificate

Theorem 100.40 (Padic prime-scale result semantic name certificate)**.** *If* p *is a* $\mathsf{Prime}^{\uparrow}$ *prime and* e *is a unary exponent, then the result carrier* $r \mapsto \mathsf{PadicPrimeScale}(p, e, r)$ *forms a semantic naming certificate with* hsame *as classifier.*

Proof. Prime-scale totality supplies an inhabited result carrier. Reflexivity, symmetry, and transitivity of the classifier are exactly the corresponding BEDC hsame laws on result histories. Carrier transport along hsame is obtained by identifying the two result histories and reusing the concrete prime-scale witness. The source, pattern, and ledger predicates are the same result carrier. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale result semanticNameCert`

Theorem 100.41 (Prime scale at unit exponent reads back the prime)**.** *For every history* p*, if* $\mathsf{NatPrime}(p)$*, then* $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(\mathsf{e}), p)$*.*

Proof. The prime witness supplies the unary carrier field for p required by Definition 64.1. The zero multiplication constructor gives $\mathsf{NatMul}(p, \mathsf{e}, \mathsf{e})$. The left-unit part of Definition 21.12 supplies the continuation $\mathsf{Cont}(\mathsf{e}, p, p)$. Applying the successor constructor of Definition 64.1 therefore yields $\mathsf{NatMul}(p, \mathsf{E1}(\mathsf{e}), p)$. Pairing this multiplication witness with the given prime witness is exactly $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(\mathsf{e}), p)$ by Definition 100.1. □

Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale unit exponent prime`
Lean4 ✓ `BEDC.Derived.PadicUp.PadicPrimeScale unit exponent result prime`

Table 100.1: Constructive-boundary decomposition of the p-adic external scope.

Scope item	Status	Evidence or footprint
$\mathbb{Z}_p$ ring and completeness	BEDC reconstructed constructive kernel	The ring surface is carried by `BEDC.Derived.PadicUp.ZpInt_comm_ring_laws`; completeness is carried by `BEDC.Derived.PadicUp.ZpInt_complete`.
$\mathbb{Q}_p$ valued field: field, valuation, norm, ultrametric comparison	BEDC reconstructed constructive kernel	The field/valuation/completeness summary is `BEDC.Derived.PadicUp.QpInt_valued_field`; its valuation, norm, multiplicative valuation, lower-bound additivity, and strong unequal-valuation addition rows are exposed through `BEDC.Derived.PadicUp.QpValuationCore`.
Hensel existence and uniqueness	BEDC reconstructed constructive kernel	Existence, lift, and uniqueness are carried by `BEDC.Derived.PadicUp.henselRoot_is_root`, `BEDC.Derived.PadicUp.henselRoot_lifts`, `BEDC.Derived.PadicUp.henselRoot_unique`, and `BEDC.Derived.PadicUp.henselRoot_unique_of_data`.
FTA: fundamental theorem of arithmetic / unique factorisation used by p-adic valuations	BEDC reconstructed constructive kernel	The arithmetic extraction and uniqueness surface is carried by `BEDC.Derived.PrimeUp.factorize`, `BEDC.Derived.PrimeUp.factorization_unique_perm`, and `BEDC.Derived.PrimeUp.IntegerPrimeFactorization`.
mathlib `PadicInt`/`Padic` quotient-completion objects and standard Real-valued norm interpretation	measured/refused classical boundary	This row is outside the BEDC kernel because the reference corpus uses quotient and classical comparison infrastructure. Footprint: `PadicInt`, `Padic`, `Real`; measured/refused dependencies `Classical.choice`, `Quot.sound`, and `propext`.
Global product formula and classical absolute-value package	measured/refused classical boundary	This row is outside the local $\mathsf{Padic}^{\uparrow}$ constructive kernel because it requires the archimedean `Real` side plus global product quantification over places. Footprint: `Real`, absolute values, place product; measured/refused dependencies `Classical.choice`, `Quot.sound`, and `propext`.

Theorem 100.42 (Padic external-scope constructive-boundary exhaustion)**.** *For the declared mathlib p-adic scope consisting of $\mathbb{Z}_p$ ring and completeness, $\mathbb{Q}_p$ valued-field structure with val-*

uation, norm, and ultrametric comparison, Hensel existence and uniqueness, FTA as unique-factorisation arithmetic used by p-adic valuations, mathlib `PadicInt/Padic` *quotient-completion objects with their standard Real-valued norm reading, and the global product-formula package, Table 100.1 exhausts the scope as*

$$S_{\text{padic}} = S_{\text{BEDC kernel}} \sqcup S_{\text{measured/refused classical boundary}}.$$

There is no hidden third region in this declared scope.

Proof. The first four rows of Table 100.1 name the BEDC constructive kernel: the $\mathbb{Z}_p$ ring and completeness rows, the $\mathbb{Q}_p$ field/valuation/norm/completeness summary, the Hensel existence-and-uniqueness rows, and the FTA / unique-factorisation arithmetic used by valuation reads. The last two rows name the refused classical boundary: the quotient-completion `PadicInt/Padic` objects with their Real-valued norm reading, and the global product-formula package with its archimedean and all-places footprint. These rows are pairwise classified by their status column, and the scope sentence lists exactly these six items. □

Closure status for $\mathsf{NameCert}_{\mathsf{Padic}^\uparrow}$.

Constructive story: The p-adic surface closed here is the prime-scale result row: choose a displayed $\mathsf{Prime}^\uparrow$ history p, choose a unary exponent e, and admit exactly those result histories r that the unary multiplication witness displays as p multiplied by e. The classifier on accepted results is hsame of the result history, so two scale computations name the same result only when multiplication determinacy identifies their displayed endpoints, while non-prime bases, non-unary exponents, and histories not produced by the scale predicate are excluded. The source, pattern, and ledger are all read from the same prime-scale package: the source fixes the prime and exponent, the pattern records unary components, totality, unique result, successor and append inversion, unit-exponent readback, and factorization rows, and the ledger keeps the underlying NatPrime, NatMul, UnaryHistory, Cont, append, and hsame witnesses visible. Prime-scale totality supplies a carried result, the hsame laws give reflexivity, symmetry, and transitivity, and transport along hsame repacks the same concrete multiplication witness, so the five name-certificate obligations close without a valuation field. A history h lies in the signature gap of a $\mathsf{Padic}^\uparrow$ token p_0 when the source-domain accepts the selected prime and exponent, the signature relation pairs a scale result with an answer history and evidence ledger, and the token-introduction relation names that answer as p_0; unfolded, h is the unique unary product of the prime base by the exponent encoded in the token. The chapter registers this prime-scale result certificate under the name $\mathsf{Padic}^\uparrow$; it does not introduce p-adic Cauchy completion, valuation algebra, field laws, or a standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Padic}^\uparrow}$, maturePackageClosed)

Scope closed: The mature surface binds the prime-scale carrier, totality, uniqueness, successor and append inversion, unit-exponent readback, append factorization, hsame transport, semantic certificate, and downstream number-theory consumer examples to the BHist, NatPrime, NatMul, UnaryHistory, Cont, append, and hsame primitives cited in the chapter proofs.

External correspondence: ExternalCorrespondenceStatus($\mathsf{NameCert}_{\mathsf{Padic}^\uparrow}$, calibratedReconstruction)

External reference: mathlib v4.x p-adic corpus, commit `PIN-MATHLIB-COMMIT`.

Reconstruction scope: $\mathbb{Z}_p$ ring and completeness; $\mathbb{Q}_p$ valued field with field, valuation, norm, and ultrametric comparison; Hensel existence and uniqueness; FTA as unique-factorisation arithmetic used by p-adic valuations; mathlib `PadicInt/Padic` quotient-completion objects and standard Real-valued norm reading; global product formula.

Boundary exhaustion: Table 100.1

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Padic}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PadicUp.PadicPrimeScale\_result\_semanticNameCert`

Bridge status: none

Not claimed: This does not close p-adic Cauchy completion, valuation algebra, field laws, the standard bridge, or RealUp-like completeness.

Upgrade path: No further theory-axis upgrade remains; bridge verification is recorded separately when the standard bridge target is present.

Chapter 101

A Concrete Naming Certificate for $\mathsf{Adele}^{\uparrow}$

The adele interface $\mathsf{Adele}^{\uparrow}$ packages the restricted product of $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Padic}^{\uparrow}}$ over all primes p. It depends on $\mathsf{NameCert}_{\mathsf{Padic}^{\uparrow}}$ (chapter 100) and supplies the global object on which automorphic and L-function certificates are built.

101.1 Carrier definition

Definition 101.1 (Adele history carrier)**.** *For a history h, the predicate* $\mathsf{AdeleHistoryCarrier}(h)$ *holds when there are histories r, p, e, s such that r is a constant real history,* $\mathsf{PadicPrimeScale}(p, e, s)$ *holds, and*

$$\mathsf{hsame}(h, \mathsf{append}(r, s)).$$

Lean4 (definition) `BEDC.Derived.AdeleUp.AdeleHistoryCarrier`

Theorem 101.2 (Adele history carrier nonempty)**.** *If* $\mathsf{AdeleHistoryCarrier}(h)$ *holds, then h is not internally the same as* e.

Proof. Unpack the carrier into its constant real endpoint, p-adic scale endpoint, and append witness. If $h\mathsf{hsamee}$, the append witness identifies the appended real and scale endpoints with e, so append-empty exactness makes the real endpoint empty. The constant real endpoint is one-sealed, and empty cannot be internally the same as a one-sealed history. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier not empty`

Theorem 101.3 (Adele carrier continuation result nonempty)**.** *If* $\mathsf{AdeleHistoryCarrier}(h)$ *holds,* $\mathsf{Cont}(h, k, r)$ *holds, and* $r\mathsf{hsamee}$*, then contradiction follows.*

Proof. Transport the continuation result along $r\mathsf{hsamee}$ to a continuation with empty result. The empty-result inversion theorem for Cont forces the source endpoint h to be e. This contradicts the nonempty theorem for the adele history carrier. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier cont result nonempty`

Theorem 101.4 (Adele right carrier continuation result nonempty)**.** *If* $\mathsf{AdeleHistoryCarrier}(k)$ *holds,* $\mathsf{Cont}(h, k, r)$ *holds, and* $r\mathsf{hsamee}$*, then contradiction follows.*

Proof. Transport the continuation result along $r\mathsf{hsamee}$ to a continuation with empty result. The empty-result inversion theorem for Cont forces the right endpoint k to be e, contradicting the nonempty theorem for the adele history carrier. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier cont right result nonempty`

Theorem 101.5 (Adele appended right carrier continuation result nonempty)**.** *If* $\mathsf{AdeleHistoryCarrier}(k)$ *holds,* $\mathsf{Cont}(h, \mathsf{append}(p, k), r)$ *holds, and* $r\mathsf{hsamee}$*, then contradiction follows.*

Proof. Transport the continuation result along $r\mathsf{hsamee}$ to a continuation with empty result. The empty-result inversion theorem for Cont forces $\mathsf{append}(p, k)\mathsf{hsamee}$. Empty append inversion recovers $k\mathsf{hsamee}$, contradicting the nonempty theorem for the adele history carrier. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier cont append right result nonempty`

Theorem 101.6 (Adele real-padic e1 real nonempty scale result package)**.** *For histories* r, p, e, s : BHist*, if* r *is a rational-history carrier and* $\mathsf{PadicPrimeScale}(p, \mathsf{e1}(e), s)$ *holds, then the real component* $\mathsf{e1}(r)$ *is a real constant-history carrier and the padic scale result is not history-same as* e*:*

$$\mathsf{RatHistoryCarrier}(r) \wedge \mathsf{PadicPrimeScale}(p, \mathsf{e1}(e), s) \Longrightarrow \mathsf{RealConstantHistoryCarrier}(\mathsf{e1}(r)) \wedge (\mathsf{hsame}(s, \mathsf{e}) \to \bot).$$

The real component is certified by the $\mathsf{Real}^{\uparrow}$ $e1$ *readback theorem from the rational carrier. The padic component uses the empty-result iff for* $\mathsf{Padic}^{\uparrow}$ *scale: an empty result would force* $\mathsf{hsame}(\mathsf{e1}(e), \mathsf{e})$*, which is excluded by the BEDC history constructor separation.*

Proof. Apply the real constant-history $e1$ iff to the rational carrier witness for r. For the padic result, assume $\mathsf{hsame}(s, \mathsf{e})$. The padic empty-result iff turns this into $\mathsf{hsame}(\mathsf{e1}(e), \mathsf{e})$, contradicting the BEDC $e1$-versus-empty separation theorem. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealPadic e1 real nonempty scale result package`

Theorem 101.7 (Adele carrier from the first-prime unit scale)**.** *For any rational-history carrier tail* r*, the history formed by appending the one-sealed real component to the concrete first-prime unit scale result is an adele-history carrier:*

$$\mathsf{RatHistoryCarrier}(r) \Longrightarrow \mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(r), \mathsf{E1}(\mathsf{E1}(\mathsf{e})))).$$

Proof. The real component is a constant real history by the $e1$ readback theorem for rational carriers. The padic component is the concrete first-prime unit scale from Definition 100.39. The final carrier equality is reflexive history sameness for the displayed append history. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier first prime unit scale`

Theorem 101.8 (Adele visible scale result carrier and nonempty result)**.** *For histories* r, p, e, s : BHist*, if* $\mathsf{RealConstantHistoryCarrier}(r)$ *and* $\mathsf{PadicPrimeScale}(p, \mathsf{e1}(e), s)$ *hold, then the appended history* $\mathsf{append}(r, s)$ *is an adele-history carrier and the visible scale result is not history-same as* e*:*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(r, s)) \wedge (\mathsf{hsame}(s, \mathsf{e}) \to \bot).$$

Proof. Use $r, p, \mathsf{e1}(e), s$ as the four carrier witnesses for the appended adele history. The append witness is reflexive history sameness. If $s\mathsf{hsamee}$, the p-adic empty-result iff forces $\mathsf{hsame}(\mathsf{e1}(e), \mathsf{e})$, contradicting the BEDC separation of one-sealed histories from e. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible scale result nonempty`

Theorem 101.9 (Adele visible zero-scale result carrier and nonempty result)**.** *For histories* r, p, e, s : BHist*, if* RealConstantHistoryCarrier(r) *and* PadicPrimeScale$(p, \mathsf{E0}(e), s)$ *hold, then the appended history* append(r, s) *is an adele-history carrier and the visible zero-scale result is not history-same as* e*:*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(r, s)) \wedge (\mathsf{hsame}(s, \mathsf{e}) \to \bot).$$

Proof. Use $r, p, \mathsf{E0}(e), s$ as the four carrier witnesses for the appended adele history. The append witness is reflexive history sameness. If $s$hsamee, the p-adic empty-result iff forces hsame$(\mathsf{E0}(e), \mathsf{e})$, contradicting the BEDC separation of zero-headed histories from e. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible zero scale result nonempty`

Theorem 101.10 (Adele prime-swap scale readback)**.** *Let* $real, p, q, r, s$: BHist*. If* $real$ *is a constant real-history carrier and the two swapped prime scales* PadicPrimeScale(p, q, r) *and* PadicPrimeScale(q, p, s) *hold, then both appended real-padic histories are adele-history carriers and they are history-same:*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(real, r)) \wedge \mathsf{AdeleHistoryCarrier}(\mathsf{append}(real, s)) \wedge \mathsf{hsame}(\mathsf{append}(real, r), \mathsf{append}(real, s)).$$

Proof. Use the displayed real component and the two displayed prime-scale witnesses as the adele carrier payloads. The p-adic scale results are identified by prime-exponent commutativity for PadicPrimeScale, and applying the fixed real append context to that hsame identifies the two adele histories. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier prime swap scale readback`

Theorem 101.11 (Adele visible scale continuation nonempty package)**.** *For histories* r, p, e, s, k, out : BHist*, if* RealConstantHistoryCarrier(r)*,* PadicPrimeScale$(p, \mathsf{e1}(e), s)$*, and* Cont$(\mathsf{append}(r, s), k, out)$ *hold, then*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(r, s)) \wedge (\mathsf{hsame}(s, \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(out, \mathsf{e}) \to \bot).$$

Proof. The visible scale theorem supplies both the adele-history carrier for append(r, s) and the nonempty scale endpoint s. Feeding that carrier and the displayed continuation into the adele continuation-result theorem gives the nonempty output endpoint. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible scale cont nonempty package`

Theorem 101.12 (Adele visible zero-scale continuation nonempty package)**.** *For histories* r, p, e, s, k, out : BHist*, if* RealConstantHistoryCarrier(r)*,* PadicPrimeScale$(p, \mathsf{e0}(e), s)$*, and* Cont$(\mathsf{append}(r, s), k, out)$ *hold, then*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(r, s)) \wedge (\mathsf{hsame}(s, \mathsf{e}) \to \bot) \wedge (\mathsf{hsame}(out, \mathsf{e}) \to \bot).$$

Proof. The visible zero-scale theorem supplies both the adele-history carrier for append(r, s) and the nonempty scale endpoint s. Feeding that carrier and the displayed continuation into the adele continuation-result theorem gives the nonempty output endpoint. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible zero scale cont nonempty package`

Theorem 101.13 (Adele e1 real append carrier is nonempty)**.** *For histories* $realTail, p, exponent, result$: BHist*, if* RealConstantHistoryCarrier(e1($realTail$)) *and* PadicPrimeScale($p, exponent, result$) *hold, then the appended real-padic history is an adele-history carrier and cannot be history-same as* e*:*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{e1}(realTail), result)) \wedge (\mathsf{hsame}(\mathsf{append}(\mathsf{e1}(realTail), result), \mathsf{e}) \to \bot).$$

Proof. Use the displayed one-sealed real component, p, *exponent*, and *result* as the four witnesses for the adele carrier. If the append were empty, append-empty exactness would force $\mathsf{e1}(realTail)\mathsf{hsamee}$, contradicting BEDC constructor separation. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier e1 real append nonempty`

Theorem 101.14 (Adele visible scale append is not empty)**.** *For histories* r, p, e, s : BHist*, if* RealConstantHistoryCarrier(r) *and* PadicPrimeScale($p, \mathsf{e1}(e), s$) *hold, then the appended visible scale history is not history-same as* e*:*

$$\mathsf{hsame}(\mathsf{append}(r, s), \mathsf{e}) \to \bot.$$

Proof. Assume the appended history is empty. Append-empty exactness makes the scale result s empty. The p-adic empty-result iff then forces $\mathsf{hsame}(\mathsf{e1}(e), \mathsf{e})$, contradicting the BEDC separation of one-sealed histories from e. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible scale append not empty`

Theorem 101.15 (Adele empty scale real readback)**.** *Let* $real, p, result$: BHist*. If real is a constant real-history carrier and* PadicPrimeScale($p, \mathsf{e}, result$) *holds, then* append($real, result$) *is an adele-history carrier and*

$$\mathsf{append}(real, result)\mathsf{hsame}real.$$

Proof. The padic empty-result iff identifies *result* with e because the exponent is e. The adele carrier uses the displayed real component, prime, empty exponent, scale result, and reflexive append witness. The readback to *real* follows from the identified empty scale result and append-right unit. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier empty scale real readback`

Theorem 101.16 (Adele unit scale real-prime readback)**.** *Let* $real, p, result$: BHist*. If real is a constant real-history carrier and* PadicPrimeScale($p, \mathsf{E1}(\mathsf{e}), result$) *holds, then* append($real, result$) *is an adele-history carrier and*

$$\mathsf{append}(real, result)\mathsf{hsameappend}(real, p).$$

Proof. Use the displayed real component, prime base, unit exponent, and scale result as the four witnesses for the adele carrier. The prime-scale unit-exponent readback identifies *result* with p; applying append congruence under the fixed real component gives the displayed real-prime readback. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier unit scale real prime readback`

Theorem 101.17 (Adele unit-left scale continuation readback)**.** *Let* $real, p, q, result$: BHist*. If real is a constant real-history carrier, q is unary, and* PadicPrimeScale(p, append($\mathsf{E1}(\mathsf{e}), q$), $result$) *holds, then* append($real, result$) *is an adele-history carrier and*

$$\exists e.\, \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{Cont}(p, e, result).$$

Proof. The adele carrier uses the displayed real component, prime base, appended unit-left exponent, and scale result. The continuation readback is exactly the unit-left prime-scale append factorization applied to the unary right exponent. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier unit left scale cont readback`

Theorem 101.18 (Adele unit-left scale append readback)**.** *Let* $real, p, q, result : \mathsf{BHist}$*. If* $real$ *is a constant real-history carrier,* q *is unary, and* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(\mathsf{E1}(\mathsf{e}), q), result)$ *holds, then* $\mathsf{append}(real, result)$ *is an adele-history carrier and*

$$\exists e.\, \mathsf{PadicPrimeScale}(p, q, e) \wedge \mathsf{append}(real, result)\mathsf{hsameappend}(real, \mathsf{append}(p, e)).$$

Proof. The adele carrier uses the displayed real component, prime base, appended unit-left exponent, and scale result. The unit-left prime-scale factorization extracts a right-factor result e with $\mathsf{Cont}(p, e, result)$; reading that continuation as the append endpoint and applying the fixed real append context gives the displayed real-padic readback. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier unit left scale append readback`

Theorem 101.19 (Adele unit-right scale continuation readback)**.** *Let* $real, p, w, result : \mathsf{BHist}$*. If* $real$ *is a constant real-history carrier,* w *is unary, and* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(\mathsf{e})), result)$ *holds, then* $\mathsf{append}(real, result)$ *is an adele-history carrier and*

$$\exists n.\, \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{Cont}(n, p, result).$$

Proof. The adele carrier uses the displayed real component, prime base, appended unit-right exponent, and scale result. The continuation readback is exactly the unit-right prime-scale append factorization applied to the unary left exponent. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier unit right scale cont readback`

Theorem 101.20 (Adele unit-right scale append readback)**.** *Let* $real, p, w, result : \mathsf{BHist}$*. If* $real$ *is a constant real-history carrier,* w *is unary, and* $\mathsf{PadicPrimeScale}(p, \mathsf{append}(w, \mathsf{E1}(\mathsf{e})), result)$ *holds, then* $\mathsf{append}(real, result)$ *is an adele-history carrier and*

$$\exists n.\, \mathsf{PadicPrimeScale}(p, w, n) \wedge \mathsf{append}(real, result)\mathsf{hsameappend}(real, \mathsf{append}(n, p)).$$

Proof. The adele carrier uses the displayed real component, prime base, appended unit-right exponent, and scale result. The unit-right prime-scale factorization extracts a left-factor result n with $\mathsf{Cont}(n, p, result)$; reading that continuation as the append endpoint and applying the fixed real append context gives the displayed real-padic readback. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier unit right scale append readback`

Theorem 101.21 (Adele real-stream prefix visible scale carrier)**.** *Let* $x, y : \mathbb{N} \to \mathsf{BHist}$ *be real stream representatives whose visible left stream is unary at every index. If* $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ *holds,* $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$*,* $y(n)\mathsf{hsameE1}(i)$*, and the concrete first-prime* p*-adic scale* $\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e}))$ *holds, then* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ *holds and*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)).$$

Proof. The add-left prefix projection recovers the classifier at index n. The visible real-pair readback at that prefix identifies the displayed $\mathsf{E1}(\mathsf{E1}(d))$ endpoint as a rational denominator carrier, hence as a constant real-history carrier. Combining that real component with the stored first-prime p-adic scale and reflexive history sameness of the displayed append endpoint gives the adele carrier. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix visible scale carrier`

Theorem 101.22 (Adele long-prefix visible scale carrier)**.** *Let $x, y : \mathbb{N} \to \mathsf{BHist}$ be real stream representatives. If $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ holds, $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$, $y(n)\mathsf{hsameE1}(i)$, and the concrete first-prime p-adic scale $\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), e, r)$ holds, then $\mathsf{RealStreamPrefixClassifier}(x, y,$ holds and*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)).$$

Proof. Induct on m through the recursive finite-prefix classifier to descend from $m+n$ to n. At the selected index, the displayed one-extension endpoints read back through the rational stream-prefix classifier: the left displayed tail is unary, hence $\mathsf{E1}(d)$ is a rational-history carrier and $\mathsf{E1}(\mathsf{E1}(d))$ is a constant real-history carrier. Combining this real component with the stored first-prime p-adic scale and reflexive sameness of the appended endpoint gives the adele-history carrier. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible scale carrier`

Theorem 101.23 (Adele long-prefix visible p-adic scale carrier)**.** *Let $x, y : \mathbb{N} \to \mathsf{BHist}$ be real stream representatives. If $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ holds, $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$, $y(n)\mathsf{hsameE1}(i)$, and $\mathsf{PadicPrimeScale}(p, e, r)$ holds, then $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ holds and*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)).$$

Proof. Induct on m through the finite-prefix classifier to descend from $m+n$ to n. The visible one-extension pair at index n reads back a unary denominator tail, so $\mathsf{E1}(d)$ is a rational-history carrier and $\mathsf{E1}(\mathsf{E1}(d))$ is a constant real-history carrier. The arbitrary displayed p-adic scale supplies the non-real component of the adele carrier, and reflexive history sameness identifies the displayed append endpoint. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible padic scale carrier`

Theorem 101.24 (Adele long-prefix visible unit-scale readback)**.** *Let $x, y : \mathbb{N} \to \mathsf{BHist}$ be real stream representatives. If $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ holds, $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$, $y(n)\mathsf{hsameE1}(i)$, and the displayed p-adic unit scale $\mathsf{PadicPrimeScale}(p, \mathsf{E1}(\mathsf{e}), r)$ holds, then $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ holds,*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)),$$

and the appended adele endpoint reads back to the displayed prime endpoint:

$$\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)\mathsf{hsameappend}(\mathsf{E1}(\mathsf{E1}(d)), p).$$

Proof. Induct on m through the finite-prefix classifier to descend from $m + n$ to n. The visible endpoints at index n recover a unary denominator tail, hence $\mathsf{E1}(d)$ is a rational-history carrier and $\mathsf{E1}(\mathsf{E1}(d))$ is a constant real-history carrier. The unit-exponent prime-scale readback identifies r with p, and applying the fixed real append context gives the displayed adele endpoint readback. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible unit scale readback`

Theorem 101.25 (Adele long-prefix visible scale result is not empty)**.** *Let* $x, y : \mathbb{N} \to \mathsf{BHist}$ *be real stream representatives. If* $\mathsf{RealStreamPrefixClassifier}(x, y, m + n)$ *holds,* $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$*,* $y(n)\mathsf{hsameE1}(i)$*, and the concrete first-prime* p*-adic visible scale* $\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), \mathsf{E1}(e), r)$ *holds, then* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ *holds,*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)),$$

and the visible scale result is not history-same as e*:*

$$\mathsf{hsame}(r, \mathsf{e}) \to \bot.$$

Proof. Induct on m through the prefix classifier to descend from $m + n$ to n. The selected one-extension endpoints read back to a unary denominator tail, hence to a rational carrier and then to the constant real-history carrier $\mathsf{E1}(\mathsf{E1}(d))$. The visible p-adic scale theorem packages this real component with the displayed scale and separates the scale result from e. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible scale result nonempty`

Theorem 101.26 (Adele long-prefix visible zero-scale result is not empty)**.** *Let* $x, y : \mathbb{N} \to \mathsf{BHist}$ *be real stream representatives. If* $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ *holds,* $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$*,* $y(n)\mathsf{hsameE1}(i)$*, and the concrete first-prime* p*-adic visible zero scale* $\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), \mathsf{E0}(e), r)$ *holds, then* $\mathsf{RealStreamPrefixClassifier}(x, y, n)$ *holds,*

$$\mathsf{AdeleHistoryCarrier}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)),$$

and the visible zero-scale result is not history-same as e*:*

$$\mathsf{hsame}(r, \mathsf{e}) \to \bot.$$

Proof. Induct on m through the prefix classifier to descend from $m + n$ to n. The selected one-extension endpoints read back to a unary denominator tail, hence to a rational carrier and then to the constant real-history carrier $\mathsf{E1}(\mathsf{E1}(d))$. The visible zero-scale theorem packages this real component with the displayed p-adic scale and separates the scale result from e. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible zero scale result nonempty`

Theorem 101.27 (Adele long-prefix visible scale continuation result is not empty)**.** *Let* $x, y : \mathbb{N} \to \mathsf{BHist}$ *be real stream representatives. If* $\mathsf{RealStreamPrefixClassifier}(x, y, m+n)$ *holds,* $x(n)\mathsf{hsameE1}(\mathsf{E1}(d))$*,* $y(n)\mathsf{hsameE1}(i)$*, and the concrete first-prime* p*-adic visible scale* $\mathsf{PadicPrimeScale}(\mathsf{E1}(\mathsf{E1}(\mathsf{e})), \mathsf{E1}(e), r)$ *holds, then every continuation out of the appended visible scale endpoint has a non-empty result:*

$$\mathsf{Cont}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r), k, out) \Longrightarrow \mathsf{hsame}(out, \mathsf{e}) \to \bot.$$

Proof. The long-prefix visible scale result theorem supplies an adele-history carrier for $\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r)$. Applying the adele carrier continuation-result nonemptiness theorem to the displayed continuation rules out $out\mathsf{hsame}\mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible scale cont result nonempty`

Theorem 101.28 (Adele long-prefix visible scale append is not empty)**.** *Under the hypotheses of Definition 101.22, the appended visible scale endpoint is not history-same as* e*:*

$$\mathsf{hsame}(\mathsf{append}(\mathsf{E1}(\mathsf{E1}(d)), r), \mathsf{e}) \to \bot.$$

Proof. The long-prefix visible scale carrier theorem supplies the adele-history carrier for the displayed append endpoint. Adele-history carriers cannot be history-same as e, so the assumed empty append endpoint is impossible. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealStreamPrefix long prefix visible scale append not empty`

Theorem 101.29 (Real–padic empty-result split)**.** *Let the two real coordinates be displayed as endpoint histories* $\mathsf{E1}(d)$ *and* $\mathsf{E1}(e)$*, and let two p-adic prime-scale factors* $\mathsf{PadicPrimeScale}(p, w, n)$ *and* $\mathsf{PadicPrimeScale}(p, q, r)$ *be linked by* $\mathsf{Cont}(n, r, out)$*. Then out*hsame*e exactly when the real endpoint payloads are rational carriers and both p-adic exponent factors satisfy w*hsame *and q*hsame*.*

Proof. The endpoint theorem for $\mathsf{RealConstantHistoryCarrier}$ reads $\mathsf{E1}(d)$ and $\mathsf{E1}(e)$ back to rational-carrier payloads. The p-adic append empty-result theorem applies to the two prime-scale factors and the continuation witness, splitting *out*hsamee into the two empty exponent conditions. Combining these independent readbacks gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleRealPadic e1 append empty padic result iff`

Theorem 101.30 (Adele constant-slice carrier transport)**.** *Let* $\mathsf{AdeleConstantSliceCarrier}(h)$ *mean that histories* d, p, e, r *exist with*

$$\mathsf{hsame}(h, \mathsf{E1}(d)) \ \wedge \ \mathsf{RatHistoryCarrier}(d) \ \wedge \ \mathsf{PadicPrimeScale}(p, e, r).$$

If $h\mathsf{hsame}h'$ *and* $\mathsf{AdeleConstantSliceCarrier}(h)$*, then* $\mathsf{AdeleConstantSliceCarrier}(h')$*.*

Proof. Unpack the constant-slice witnesses d, p, e, r. The rational endpoint and the p-adic scale witnesses are unchanged. Symmetry and transitivity of hsame move the displayed $\mathsf{E1}(d)$ endpoint from h to h'. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleConstantSliceCarrier hsame transport`

101.2 The certificate

Theorem 101.31 (Adele history-carrier semantic name certificate)**.** *The adele history carrier forms a semantic naming certificate with* hsame *as classifier, using the same adele carrier as source, pattern, and ledger policy.*

Proof. A rational tail $\mathsf{E1}(\mathsf{e})$ supplies the constant-real side, and the first-prime unit scale supplies the p-adic side, so the adele carrier is inhabited by the displayed append endpoint from Definition 101.7. The classifier laws are the BEDC hsame equivalence laws. For carrier transport, unpack an adele witness into its real component, prime, exponent, result, and append comparison; composing that comparison with the symmetric endpoint hsame moves the same witnesses to the target history. Pattern and ledger soundness are identity maps on the carrier predicate. □

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier semanticNameCert`

Theorem 101.32 (Adele visible-scale append result hsame transport)**.** *For histories* a, a', s : BHist*, if a is an adele-history carrier, the append endpoint* $\mathsf{append}(a, s)$ *is not history-same as* e*, and a*hsame*a′, then the same visible scale s has a nonempty append endpoint over a′:*

$$\mathsf{AdeleHistoryCarrier}(a) \wedge (\mathsf{hsame}(\mathsf{append}(a, s), \mathsf{e}) \to \bot) \wedge a\mathsf{hsame}a' \Longrightarrow \mathsf{hsame}(\mathsf{append}(a', s), \mathsf{e}) \to \bot.$$

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleHistoryCarrier visible scale append result hsame transport`

Proof. The carrier premise fixes the source endpoint in the adele layer. Let H be the nonempty premise for $\mathsf{append}(a, s)$ and let $\alpha : a\mathsf{hsame}a'$. By the reverse direction of Definition 4.50, applied with the reflexive right-context comparison $s\mathsf{hsame}s$, the endpoint comparison α gives

$$\mathsf{hsame}(\mathsf{append}(a, s), \mathsf{append}(a', s)).$$

If $\mathsf{append}(a', s)\mathsf{hsame}e$, transitivity of hsame carries this empty comparison back to $\mathsf{append}(a, s)\mathsf{hsame}e$, contradicting H. □

Theorem 101.33 (Adele mature treatment)**.** *The displayed Adele carrier exports a mature BEDC treatment of the public* $\mathsf{Adele}^{\uparrow}$ *certificate surface. Its semantic* NameCert *is the certificate of Definition 101.31; its visible-scale and long-prefix readbacks are the* BHist *rows carried by the real constant-slice, p-adic scale, append nonempty-result, and prefix readback theorems above; and its ClassField and Automorphic consumers may cross the boundary only through* BHist*,* hsame*, and* Cont*. This treatment does not assert a standard-model bridge.*

Lean4 ✓ `BEDC.Derived.AdeleUp.AdeleUp StdBridge`

Proof. Start with the Adele carrier constructed from the real constant-slice row and the first-prime p-adic scale row. The nonempty append-result theorem gives the visible scale, while the long-prefix readback rows identify the finite history prefixes available to public consumers.

Next apply the semantic certificate of Definition 101.31. It binds the carrier, classifier, source, pattern, and ledger rows to BHist, hsame, and Cont without adding an ambient adele object.

Finally use the hsame transport theorem for visible-scale append results. Transport preserves the same carried rows across history-same endpoints, so a ClassField or Automorphic consumer receives only the displayed BEDC rows. Since no standard-model comparison is introduced in these steps, the mature treatment is paper-side only on the bridge axis. □

Closure status for $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$**.**

Constructive story: An adele history is built by putting a constant real history beside a p-adic prime-scale result: the carrier accepts a history h when some real component r, prime row p, exponent row e, and scale result s exist with h history-equal to $\mathsf{append}(r, s)$. The real side is read from the $\mathsf{Real}^{\uparrow}$ constant-stream surface, the p-adic side from $\mathsf{Padic}^{\uparrow}$ prime scaling, and empty or malformed scale results are excluded by the same constructor-separation and non-emptiness rows used by the upstream certificates. Two adeles are classified by hsame on the appended history, while transport unpacks the real component, prime, exponent, scale result, and append comparison and moves that same witness package across the endpoint comparison. The source, pattern, and ledger are all the adele carrier predicate: the source accepts exactly these real-padic append packages, the pattern records non-emptiness, unit-scale, empty-scale, prime-swap, and continuation readback rows, and the ledger keeps the hidden real, prime, exponent, scale, and append witnesses aligned with the visible endpoint. Because a rational tail supplies a constant real component and the first-prime unit scale supplies the p-adic component, the carrier is inhabited; because hsame is reflexive, symmetric, and transitive and carrier transport repacks the same witnesses, the five name-certificate obligations follow. A history h lies in the signature gap of an $\mathsf{Adele}^{\uparrow}$ token p_0 when the source-domain accepts this appended real-padic row, the signature relation pairs it with an answer adele and evidence ledger, and token introduction names the answer as p_0; unfolded, its constant real part and prime-scale part are exactly the components encoded by p_0. The chapter registers this real-padic append construction under the name $\mathsf{Adele}^{\uparrow}$; it is a BEDC restricted-product surface with no product formula, automorphic theory, analytic continuation, or standard-model bridge introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$, maturePackageClosed)

Scope closed: The Adele obligation rows are exactly bound to BHist, Cont, append, hsame, the RealUp constant-stream side, the PadicPrimeScale side, and the semantic NameCert row carried by the checked AdeleHistoryCarrier certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.AdeleUp.AdeleHistoryCarrier\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: This does not close local-global product formulae, automorphic theory, analytic continuation of L-functions, or a standard-model bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 102

A Concrete Naming Certificate for $\mathsf{ModN}^{\uparrow}$

The modular-arithmetic interface $\mathsf{ModN}^{\uparrow}$ packages the carrier of integers modulo a positive natural n. It depends on $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$ (chapter 31) and is the canonical instance of $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}}$ (chapter 83) at the ideal $n\mathbb{Z}$.

102.1 Carrier definition

102.2 The certificate

Definition 102.1 (ModN quotient presentation)**.** *Let $R_{\mathbb{Z}}$ be a displayed $\mathsf{Ring}^{\uparrow}$ certificate for $\mathsf{Int}^{\uparrow}$ with carrier $C_{\mathbb{Z}}$, classifier $\sim_{\mathbb{Z}}$, addition $+_{\mathbb{Z}}$, additive inverse $-_{\mathbb{Z}}$, zero $0_{\mathbb{Z}}$, and multiplication $\cdot_{\mathbb{Z}}$. Let $n : \mathsf{Nat}^{\uparrow}$ be positive, and let $I_n(x)$ be a certified $\mathsf{Ideal}^{\uparrow}$ predicate over $R_{\mathbb{Z}}$ whose intended membership is $x \in n\mathbb{Z}$. The quotient presentation of $\mathsf{ModN}^{\uparrow}(n)$ is the specialization*

$$\mathsf{ModN}^{\uparrow}(n) := \mathsf{QuotientRing}^{\uparrow}(R_{\mathbb{Z}}, I_n).$$

Its public classifier is the ideal-coset classifier

$$\mathsf{ModNClassifier}_n(a, b) \iff a \equiv_{I_n} b,$$

where $\equiv_{I_n}$ is the relation of Definition 83.1. Thus, after unfolding the quotient-ring relation,

$$\mathsf{ModNClassifier}_n(a, b) \iff C_{\mathbb{Z}}(a) \wedge C_{\mathbb{Z}}(b) \wedge I_n(a -_{\mathbb{Z}} b).$$

Theorem 102.2 (ModN as the nZ quotient certificate)**.** *Under the hypotheses of Definition 102.1, the quotient-ring certificate at $(R_{\mathbb{Z}}, I_n)$ supplies a $\mathsf{NameCert}$ for $\mathsf{ModN}^{\uparrow}(n)$ with exactly the quotient-ring classifier. More precisely,*

$$\mathsf{NameCert}_{\mathsf{ModN}^{\uparrow}(n)} = \mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}(R_{\mathbb{Z}}, I_n)}$$

as a five-field certificate in the sense of Definition 9.6, and for all carried endpoints a, b,

$$\mathsf{Classifier}_{\mathsf{ModN}^{\uparrow}(n)}(a, b) \iff \mathsf{Classifier}_{\mathsf{QuotientRing}^{\uparrow}(R_{\mathbb{Z}}, I_n)}(a, b) \iff C_{\mathbb{Z}}(a) \wedge C_{\mathbb{Z}}(b) \wedge I_n(a -_{\mathbb{Z}} b).$$

The additive, inverse, and multiplicative operations on $\mathsf{ModN}^{\uparrow}(n)$ are representative-independent by the same quotient-ring descent rows.

Proof. The positive $\mathsf{Nat}^{\uparrow}$ witness selects the modulus parameter n and the predicate I_n; it does not add a second quotient relation. The certified $\mathsf{Ideal}^{\uparrow}$ hypothesis puts I_n in exactly the scope required by section 81.2 and by the ideal-coset relation of Definition 83.1 over the displayed integer ring $R_{\mathbb{Z}}$.

Apply Definition 83.4 with $R = R_{\mathbb{Z}}$ and $I = I_n$. This gives reflexivity, symmetry, and transitivity for the quotient classifier on carried integer endpoints. Carrier transport for the core $\mathsf{NameCert}$ of Definition 9.7 is obtained by unfolding $a \equiv_{I_n} b$: the relation itself stores the two endpoint carrier witnesses $C_{\mathbb{Z}}(a)$ and $C_{\mathbb{Z}}(b)$.

The additive and inverse representative-independence rows are Definition 83.3 specialized to $R_{\mathbb{Z}}$ and I_n. The multiplicative representative-independence row is Definition 83.2 with the same specialization. Hence the quotient-ring source, pattern, classifier, stability, and ledger fields form the five fields demanded by Definition 9.6.

By Definition 102.1, $\mathsf{ModN}^{\uparrow}(n)$ is this quotient-ring presentation with no classifier-changing map inserted. Therefore every field of $\mathsf{NameCert}_{\mathsf{ModN}^{\uparrow}(n)}$ is the corresponding field of $\mathsf{NameCert}_{\mathsf{QuotientRing}^{\uparrow}(R_{\mathbb{Z}},I_n)}$. In particular, the classifier field is copied. The first displayed equivalence is this field identity, and the second is exactly Definition 83.1 unfolded at $R_{\mathbb{Z}}$ and I_n. □

Lean4 ✓ `BEDC.Derived.ModNUp.ModN nz quotient certificate`

Definition 102.3 (Concrete ModN BHist carrier and classifier)**.** *For a unary modulus history* $m : \mathsf{BHist}$*, define the concrete ModN carrier and classifier by*

$$\mathsf{ModNCarrier}_m(h) := \mathsf{UnaryHistory}(m) \wedge \mathsf{RingSingletonCarrier}(h)$$

and

$$\mathsf{ModNClassifier}_m(h,k) := \mathsf{ModNCarrier}_m(h) \wedge \mathsf{ModNCarrier}_m(k) \wedge \mathsf{RingSingletonClassifier}(h,k).$$

Equivalently, these are the paper names for the Lean objects $\mathsf{ModNQuotientCarrier}(m,h)$ *and* $\mathsf{ModNQuotientClassifier}$

Theorem 102.4 (Concrete ModN classifier rows)**.** *For each unary modulus history* $m : \mathsf{BHist}$*, the classifier* $\mathsf{ModNClassifier}_m$ *satisfies the following concrete rows:*

1. *if* $\mathsf{ModNCarrier}_m(h)$*, then* $\mathsf{ModNClassifier}_m(h,h)$*;*
2. *if* $\mathsf{ModNClassifier}_m(h,k)$*, then* $\mathsf{ModNClassifier}_m(k,h)$*;*
3. *if* $\mathsf{ModNClassifier}_m(h,k)$ *and* $\mathsf{ModNClassifier}_m(k,l)$*, then* $\mathsf{ModNClassifier}_m(h,l)$*;*
4. *if* $\mathsf{ModNClassifier}_m(h,k)$ *and* $\mathsf{ModNCarrier}_m(h)$*, then* $\mathsf{ModNCarrier}_m(k)$*.*

Lean4 ✓ `BEDC.Derived.ModNUp.ModNQuotientClassifier concrete rows`

Proof. Unfold Definition 102.3. The modulus component of every carrier witness is the same $\mathsf{UnaryHistory}(m)$ proof, so no modulus data is transformed by the classifier rows.

For reflexivity, keep the carrier witness for h on both endpoints and use the reflexivity row of the singleton $\mathsf{Ring}^{\uparrow}$ classifier on the $\mathsf{RingSingletonCarrier}(h)$ component.

For symmetry and transitivity, project the two endpoint carriers from the given $\mathsf{ModNClassifier}_m$ hypotheses. The remaining relation fields are exactly the symmetry and transitivity rows for $\mathsf{RingSingletonClassifier}$, then the original unary modulus witness is repacked into each endpoint carrier.

Carrier transport is a projection: the right endpoint carrier is already a field of $\mathsf{ModNClassifier}_m(h,k)$. □

Theorem 102.5 (Concrete ModN singleton operation descent rows)**.** *For each unary modulus history m : BHist, the singleton ring operations preserve* $\mathsf{ModNCarrier}_m$ *and descend through* $\mathsf{ModNClassifier}_m$*:*

1. *if* $\mathsf{ModNCarrier}_m(h)$ *and* $\mathsf{ModNCarrier}_m(k)$*, then* $\mathsf{ModNCarrier}_m(\mathsf{RingSingletonAdd}(h,k))$ *and* $\mathsf{ModNCarrier}_m(\mathsf{Ring}$
2. *if* $\mathsf{ModNCarrier}_m(h)$*, then* $\mathsf{ModNCarrier}_m(\mathsf{RingSingletonNeg}(h))$*;*
3. *if* $\mathsf{ModNClassifier}_m(h,h')$ *and* $\mathsf{ModNClassifier}_m(k,k')$*, then* $\mathsf{ModNClassifier}_m(\mathsf{RingSingletonAdd}(h,k),\mathsf{RingSingleton}$
4. *under the same classifier hypotheses,* $\mathsf{ModNClassifier}_m(\mathsf{RingSingletonMul}(h,k),\mathsf{RingSingletonMul}(h',k'))$*;*
5. *if* $\mathsf{ModNClassifier}_m(h,h')$*, then* $\mathsf{ModNClassifier}_m(\mathsf{RingSingletonNeg}(h),\mathsf{RingSingletonNeg}(h'))$*.*

Lean4 ✓ `BEDC.Derived.ModNUp.ModNQuotientCarrier singleton operation descent rows`
Lean4 ✓ `BEDC.Derived.ModNUp.ModNQuotient singleton operation descent rows`
↪variant `BEDC.Derived.ModNUp.ModNQuotientSingleton operation descent rows`

Proof. Unfold the ModN carrier. Preservation of the modulus component is immediate: each output carrier keeps the same $\mathsf{UnaryHistory}(m)$ witness carried by the input endpoints.

The $\mathsf{Ring}^{\uparrow}$ singleton operation laws supply the history component. Addition, negation, and multiplication all return the singleton ring endpoint, so the RingSingletonCarrier field is obtained from the singleton closure rows of Definition 43.6.

For descent, project the RingSingletonClassifier fields from the two ModN classifier hypotheses. The singleton $\mathsf{Ring}^{\uparrow}$ congruence rows identify the add, multiply, and negated outputs under RingSingletonClassifier.

Finally, pair those singleton classifier witnesses with the output carriers just constructed. This repacks the result in the definition of $\mathsf{ModNClassifier}_m$. □

Theorem 102.6 (Concrete ModN NameCert scope)**.** *For every unary modulus history m : BHist, the carrier* $\mathsf{ModNCarrier}_m$*, classifier* $\mathsf{ModNClassifier}_m$*, and singleton ring operations* RingSingletonAdd*,* RingSingletonNeg*, and* RingSingletonMul *form the concrete* BHist *scope consumed by Definition 102.2. In that scope the quotient certificate reads exactly the carrier rows of Definition 102.4 and the operation rows of Definition 102.5.*

Proof. Begin with a unary modulus witness for m and unfold Definition 102.3. The source, pattern, and ledger carriers are all the same predicate $\mathsf{ModNCarrier}_m$, so the core certificate fields are supplied by Definition 102.4.

Next use Definition 102.5 to fix the additive, inverse, and multiplicative output obligations. These rows ensure that operations remain inside the concrete carrier and that replacing classified representatives does not change the classified output.

Only after those rows are in place, apply Definition 102.2. The checked quotient certificate then packages the already fixed carrier, classifier, stability, and ledger fields as the displayed NameCert scope for $\mathsf{ModN}^{\uparrow}(m)$. □

Lean4 ✓ `BEDC.Derived.ModNUp.ModNQuotient concrete namecert scope`

Closure status for $\mathsf{NameCert}_{\mathsf{ModN}^{\uparrow}}$**.**

Constructive story: A residue modulo N is presented here by a unary modulus history m together with the singleton $\mathsf{Ring}^{\uparrow}$ endpoint: a history is admitted exactly when m is unary and the history lies in the singleton ring carrier, so all non-singleton ring data and all non-unary moduli stay outside this concrete carrier. The classifier keeps both endpoint carrier witnesses and then uses the singleton ring classifier, which means this BEDC scope models the quotient-ring descent rows without forming a host quotient set or changing representatives by an external congruence. The source is the concrete $\mathsf{ModNCarrier}_m$, the pattern records the copied quotient-ring presentation at the ideal $n\mathbb{Z}$ together with

singleton addition, negation, and multiplication closure, and the ledger keeps the displayed modulus witness, endpoint carrier fields, and operation-descent rows visible. The five name-certificate obligations are exactly the checked classifier rows: the singleton endpoint gives nonemptiness, reflexivity, symmetry, and transitivity come from the singleton classifier, and carrier transport is already stored in the two endpoint fields of $\mathsf{ModNClassifier}_m$. A history h lies in the signature gap of a $\mathsf{ModN}^{\uparrow}$ token p when the source-domain accepts the unary modulus and singleton endpoint, the signature relation pairs that endpoint with an answer endpoint and operation ledger, and the token-introduction relation names the answer as p; unfolded, h is the singleton representative in the modulus scope encoded by p. The chapter registers this displayed unary-modulus quotient scope under the name $\mathsf{ModN}^{\uparrow}$; it introduces no standard bridge, no full integer modulo arithmetic, and no quotient primitive beyond the checked BHist quotient certificate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModN}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 102.6 exports the public unary-modulus ModN quotient scope over $\mathsf{ModNCarrier}_m$, $\mathsf{ModNClassifier}_m$, the singleton $\mathsf{Ring}^{\uparrow}$ operations, quotient-ring descent rows, and the displayed BHist and hsame boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModN}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ModNUp.ModNQuotient\_concrete\_namecert\_scope`

Bridge status: none

Not claimed: This chapter does not close a standard bridge, full integer modulo arithmetic, or quotient primitives beyond the displayed BHist quotient certificate.

Upgrade path: The next grade requires a public bridge from this BHist quotient scope to the standard $\mathbb{Z}/n\mathbb{Z}$ interface.

Theorem 102.7 (Concrete ModN singleton inverse sum classifies as empty)**.** *For each unary modulus history* $m : \mathsf{BHist}$ *and each history* $h : \mathsf{BHist}$*, if* $\mathsf{ModNCarrier}_m(h)$*, then*

$$\mathsf{ModNClassifier}_m\big(\mathsf{RingSingletonAdd}(h, \mathsf{RingSingletonNeg}(h)), \mathsf{e}\big).$$

Proof. Unfold Definition 102.3 in the given carrier witness. It supplies the unary modulus witness $\mathsf{UnaryHistory}(m)$ and the singleton ring carrier witness for h.

The singleton inverse row of Definition 102.5 keeps $\mathsf{RingSingletonNeg}(h)$ inside $\mathsf{ModNCarrier}_m$. Applying the singleton addition carrier row in the same theorem to h and $\mathsf{RingSingletonNeg}(h)$ keeps $\mathsf{RingSingletonAdd}(h, \mathsf{RingSingletonNeg}(h))$ inside $\mathsf{ModNCarrier}_m$. The empty endpoint is also in $\mathsf{ModNCarrier}_m$: the modulus component is the same $\mathsf{UnaryHistory}(m)$ witness, and the singleton ring carrier of e is the carrier-inhabited row of Definition 43.6.

The additive inverse law in the singleton empty-history ring of Definition 43.6 gives the singleton classifier comparison

$$\mathsf{RingSingletonClassifier}\big(\mathsf{RingSingletonAdd}(h, \mathsf{RingSingletonNeg}(h)), \mathsf{e}\big).$$

Together with the two $\mathsf{ModNCarrier}_m$ endpoint witnesses just constructed, this is exactly the definition of $\mathsf{ModNClassifier}_m$ in Definition 102.3. □

Lean4 ✓ `BEDC.Derived.ModNUp.ModNQuotient singleton add neg empty classifier`

Chapter 103

A Concrete Naming Certificate for $\mathsf{LFunction}^{\uparrow}$

The L-function interface $\mathsf{LFunction}^{\uparrow}$ packages a Dirichlet series whose coefficients arise from a fixed character or Galois representation, together with the analytic continuation and functional-equation witnesses. It depends on $\mathsf{NameCert}_{\mathsf{Dirichlet}^{\uparrow}}$ (chapter 70) and is the abstract pattern instantiated by the Riemann zeta certificate $\mathsf{NameCert}_{\mathsf{ZetaBasic}^{\uparrow}}$ (chapter 71).

103.1 Carrier definition

Theorem 103.1 (L-function Dirichlet partial-sum successor index is positive)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, a closed witness* $\mathsf{DirichletPartSum}(term, s, n, S)$ *makes the successor history* $\mathsf{BHist.e1}(n)$ *a Dirichlet positive index.*

Proof. Induct on the closed partial-sum witness. The zero constructor gives unary e, and the step constructor preserves unaryness through the successor constructor. Thus the exposed index n is unary, and n itself is the tail witness showing that $\mathsf{BHist.e1}(n)$ is positive. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum successor positive index`

Theorem 103.2 (L-function Dirichlet partial-sum successor previous uniqueness)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, a closed witness* $\mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e}$ *exposes a previous partial sum P such that* $\mathsf{DirichletPartSum}(term, s, n, P)$, $\mathsf{Cont}(P, term(n, s), S)$, *and every other closed partial sum at index n is* hsame *to P.*

Proof. Invert the successor partial-sum witness. Its predecessor gives the previous partial sum and continuation into S. Positivity of the successor index recovers unaryness of n, and unary-index determinacy identifies every other predecessor partial sum with the exposed one. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum successor previous unique`

Theorem 103.3 (L-function Dirichlet partial-sum successor determinacy)**.** *For any Dirichlet term operation term, parameter s, index n, and sum histories S, T, two closed witnesses at the same successor index determine the same result:*

$$\mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e1}(n), S) \wedge \mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e1}(n), T) \Longrightarrow S \mathsf{hsame} T.$$

Proof. Invert one successor partial-sum witness. Positivity of the exposed successor recovers unaryness of n, hence unaryness of $\mathsf{BHist.e1}(n)$. The Dirichlet unary-index determinacy theorem then compares the reconstructed left successor witness with the second witness. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum successor deterministic`

Theorem 103.4 (L-function Dirichlet zero-term successor stability)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, if* $\mathsf{DirichletPartSum}(term, s, n, S)$ *holds and the next term $term(n, s)$ is* hsame *to* e*, then the same displayed sum S is also a closed partial sum at successor index* $\mathsf{BHist.e1}(n)$*:*

$$\mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e1}(n), S).$$

Proof. Compare the canonical right-unit continuation out of S with the term history $term(n, s)$ using the assumed hsame identification with e. This supplies the successor-step continuation from S through the zero term back to S, and the `DirichletPartSum.step` constructor closes the successor partial-sum witness. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum zero term successor stable`

Theorem 103.5 (L-function Dirichlet successor zero term recovers previous result)**.** *For any Dirichlet term operation term, parameter s, index n, and result history T, if* $\mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e1}(n), T$ *holds and the step term $term(n, s)$ is* hsame *to* e*, then the successor witness exposes a predecessor P such that*

$$\mathsf{DirichletPartSum}(term, s, n, P) \wedge P\mathsf{hsame}T.$$

Proof. Invert the successor partial-sum witness to expose the predecessor P and the continuation from P through $term(n, s)$ into T. Compare this continuation with the canonical right-unit continuation from P through e, transporting the term endpoint by $term(n, s)\mathsf{hsamee}$. Continuation determinacy then identifies T with P under hsame. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum successor zero term previous result same`

Theorem 103.6 (L-function Dirichlet successor zero-term result iff)**.** *For any Dirichlet term operation term, parameter s, index n, and result history T, if $term(n, s)\mathsf{hsamee}$, then*

$$\mathsf{DirichletPartSum}(term, s, \mathsf{BHist.e1}(n), T) \iff \exists P : \mathsf{BHist},\ \mathsf{DirichletPartSum}(term, s, n, P) \wedge P\mathsf{hsame}T.$$

Proof. The forward direction is the predecessor readback for a successor whose step term is empty. Conversely, a predecessor P with $P\mathsf{hsame}T$ supplies the successor continuation by transporting the right-unit continuation $\mathsf{Cont}(P, \mathsf{e}, P)$ along $term(n, s)\mathsf{hsamee}$ and $P\mathsf{hsame}T$, after which the `DirichletPartSum.step` constructor closes the successor witness. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum successor zero term result iff`

Theorem 103.7 (L-function Dirichlet zero-terms result is empty)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, suppose every unary index m has zero term $term(m, s)\mathsf{hsamee}$. Then every closed partial-sum witness* $\mathsf{DirichletPartSum}(term, s, n, S)$ *has empty result $S\mathsf{hsamee}$.*

Proof. Induct on the closed partial-sum witness. The zero constructor gives $S = \mathsf{e}$. In the step case, the predecessor index is unary by the partial-sum index theorem, so the zero-term hypothesis identifies the next term with e. Comparing the displayed continuation with the right-unit continuation at e gives the empty successor result. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum zero terms result empty`

Theorem 103.8 (L-function positive Dirichlet partial-sum index exposes predecessor)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, if*

$$\mathsf{DirichletPartSum}(term, s, n, S)$$

holds and n is a Dirichlet positive index, then there are histories m and P such that

$$n = \mathsf{BHist.e1}(m) \wedge \mathsf{DirichletPartSum}(term, s, m, P) \wedge \mathsf{Cont}(P, term(m, s), S).$$

Proof. Apply the Dirichlet partial-sum index case split. The zero-index branch is incompatible with positive-index shape, since e cannot be identified with a visible $\mathsf{BHist.e1}$ constructor. The successor branch already exposes the displayed predecessor index, predecessor partial sum, and final continuation into S. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum positive index previous exists`

Theorem 103.9 (L-function positive Dirichlet partial-sum predecessor uniqueness)**.** *For any Dirichlet term operation term, parameter s, index n, and sum history S, if*

$$\mathsf{DirichletPartSum}(term, s, n, S)$$

holds and n is a Dirichlet positive index, then there are histories m and P such that

$$n = \mathsf{BHist.e1}(m) \wedge \mathsf{DirichletPartSum}(term, s, m, P) \wedge \mathsf{Cont}(P, term(m, s), S),$$

and every other closed partial sum at the predecessor index m is hsame *to P.*

Proof. The predecessor-existence theorem exposes the visible predecessor index m, the displayed predecessor partial sum P, and the continuation into S. The predecessor partial-sum witness makes m unary. Dirichlet unary-index determinacy then identifies any other closed partial sum at m with the exposed predecessor result P. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum positive index previous unique`

Theorem 103.10 (L-function visible Dirichlet predecessor decomposition is deterministic)**.** *For any Dirichlet term operation term, parameter s, visible index n, and sum history S, suppose S admits two displayed predecessor decompositions*

$$n = \mathsf{BHist.e1}(m) \wedge \mathsf{DirichletPartSum}(term, s, m, P) \wedge \mathsf{Cont}(P, term(m, s), S)$$

and

$$n = \mathsf{BHist.e1}(m') \wedge \mathsf{DirichletPartSum}(term, s, m', P') \wedge \mathsf{Cont}(P', term(m', s), S).$$

Then $m = m'$ and $P \mathsf{hsame} P'$.

Proof. Invert both visible successor equalities. The constructor equality identifies the predecessor indices. The exposed predecessor partial-sum witness makes the index unary, and Dirichlet unary-index determinacy identifies the two displayed predecessor results. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum visible previous decomposition deterministic`

Theorem 103.11 (L-function zero-term positive partial-sum predecessor readback)**.** *For any Dirichlet term operation term, parameter s, positive index n, and sum history S, suppose every unary index m has zero term $term(m, s)$*hsame*e. If*

$$\mathsf{DirichletPartSum}(term, s, n, S)$$

holds and n is a Dirichlet positive index, then there are histories m and P such that

$$n = \mathsf{BHist.e1}(m) \wedge \mathsf{DirichletPartSum}(term, s, m, P) \wedge P\mathsf{hsame}S,$$

and every other closed partial sum at index m is hsame *to P.*

Proof. Use positive-index predecessor inversion to write n as $\mathsf{BHist.e1}(m)$ and expose the predecessor partial sum P together with the continuation through $term(m, s)$ into S. The predecessor partial-sum witness makes m unary, so the zero-term hypothesis identifies $term(m, s)$ with e. Comparing the displayed continuation with the canonical right-unit continuation from P gives $P\mathsf{hsame}S$, and unary-index determinacy gives the stated uniqueness of predecessor readback. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum zero terms positive previous readback`

103.2 The certificate

Theorem 103.12 (L-function Dirichlet partial-sum result hsame transport)**.** *For any Dirichlet term operation term, parameter s, index n, and histories S and S', result history-sameness transports closed partial-sum witnesses:*

$$S\mathsf{hsame}S' \Longrightarrow (\mathsf{DirichletPartSum}(term, s, n, S) \Longleftrightarrow \mathsf{DirichletPartSum}(term, s, n, S')).$$

In particular, no separate unary-index hypothesis is required.

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum result hsame transport`

Proof. It is enough to prove the forward implication; applying that implication to the symmetric result identification gives the reverse direction by Definition 3.8. Fix $S\mathsf{hsame}S'$ and a closed witness of $\mathsf{DirichletPartSum}(term, s, n, S)$, and induct on that witness. In the zero case, closed generation gives $n = \mathsf{e}$ and $S = \mathsf{e}$. The result hypothesis therefore reduces to $\mathsf{ehsame}S'$. By symmetry from Definition 3.8 it also gives $S'\mathsf{hsame}$e; the empty case of the constructor characterization in Definition 3.4 makes S' literally e. After this literal replacement, the zero constructor of Definition 70.9 gives $\mathsf{DirichletPartSum}(term, s, \mathsf{e}, S')$.

In the step case, the witness has index $\mathsf{E1}(m)$ and consists of a predecessor witness $\mathsf{DirichletPartSum}(term, s, m, P$ together with the final continuation $\mathsf{Cont}(P, term(m, s), S)$. By Definition 4.40, this continuation transports along $S\mathsf{hsame}S'$ to $\mathsf{Cont}(P, term(m, s), S')$. The step constructor of Definition 70.9 then gives $\mathsf{DirichletPartSum}(term, s, \mathsf{E1}(m), S')$. □

Definition 103.13 (L-function Dirichlet finite zero tail)**.** *For a Dirichlet term operation term, parameter s, base index n, and unary tail length k, the predicate*

$$\mathsf{DirichletFiniteZeroTail}(term, s, n, k)$$

is the closed finite-interval predicate generated by:

1. $\mathsf{DirichletFiniteZeroTail}(term, s, n, \mathsf{e})$*;*
2. *if* $\mathsf{DirichletFiniteZeroTail}(term, s, n, k)$ *and* $term(\mathsf{append}(n, k), s)\mathsf{hsamee}$*, then* $\mathsf{DirichletFiniteZeroTail}(term, s, n, \mathsf{E}$

Thus it records that all displayed terms in the BEDC interval $[n, n+k)$ are history-same to e*, with $n+k$ represented by* $\mathsf{append}(n, k)$*.*

Lean4 (definition) `BEDC.Derived.LFunctionUp.DirichletFiniteZeroTail`
Lean4 (definition) `BEDC.Derived.LFunctionUp.LFunctionDirichletFiniteZeroTail`

Theorem 103.14 (L-function Dirichlet finite zero-tail stability)**.** *For any Dirichlet term operation term, parameter s, base index n, tail length k, and sum history S, if*

$$\mathsf{DirichletPartSum}(term, s, n, S) \quad \textit{and} \quad \mathsf{DirichletFiniteZeroTail}(term, s, n, k),$$

then

$$\mathsf{DirichletPartSum}(term, s, \mathsf{append}(n, k), S).$$

Lean4 ✓ `BEDC.Derived.LFunctionUp.DirichletFiniteZeroTail stability`

Proof. Induct on the closed finite zero-tail witness. In the empty-tail branch, the right-unit law for history append from Definition 4.38 identifies $\mathsf{append}(n, \mathsf{e})$ with n, so the displayed partial-sum witness is already the required partial sum at the target index.

For the successor-tail branch, write the tail as $\mathsf{E1}(k_0)$. The induction hypothesis applied to the predecessor tail gives

$$\mathsf{DirichletPartSum}(term, s, \mathsf{append}(n, k_0), S).$$

The final zero-tail field gives

$$term(\mathsf{append}(n, k_0), s)\mathsf{hsamee}.$$

Apply the one-step zero-term stability theorem Definition 103.4 at the index $\mathsf{append}(n, k_0)$. It yields

$$\mathsf{DirichletPartSum}(term, s, \mathsf{E1}(\mathsf{append}(n, k_0)), S).$$

By the recursive definition of history append, $\mathsf{append}(n, \mathsf{E1}(k_0))$ is the successor history $\mathsf{E1}(\mathsf{append}(n, k_0))$. This is exactly the target partial-sum witness for the successor tail. □

Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletPartSum finite zero tail stability`
Lean4 ✓ `BEDC.Derived.LFunctionUp.LFunctionDirichletFiniteZeroTail stability`

Closure status for $\mathsf{NameCert}_{\mathsf{LFunction}^{\uparrow}}$**.**

Constructive story: An $\mathsf{LFunction}^{\uparrow}$ certificate is generated from closed Dirichlet partial-sum witnesses, unary index rows, term histories, zero-tail ledgers, continuation endpoints, and hsame transport. The BHist carrier does not store analytic continuation as an ambient theorem; it builds finite Dirichlet evidence by induction over displayed partial sums and finite zero tails. Each successor read exposes the predecessor sum and the Cont row adding the next term, while hsame transports result histories without changing the closed witness. The chapter therefore certifies the finite Dirichlet-series substrate used by the L-function interface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds L-function Dirichlet partial-sum predecessor, determinacy, zero-tail, transport, and finite stability theorems.

Formal verification: FormalStatus(NameCert$_{\mathsf{LFunction}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: It does not claim analytic continuation, a functional equation, Euler product convergence, meromorphic extension, or host complex analysis.
Upgrade path: To reach obligationClosure, package the checked finite Dirichlet partial-sum and zero-tail theorems as an explicit NameCert carrier surface.

Chapter 104

A Concrete Naming Certificate for $\mathsf{Automorphic}^{\uparrow}$

The automorphic-form interface $\mathsf{Automorphic}^{\uparrow}$ packages a function on $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$ (chapter 101) invariant under a discrete subgroup action and satisfying the constructive growth conditions. It supplies the substrate connecting representations of the adelic group to L-functions in the Langlands programme.

104.1 Carrier definition

Theorem 104.1 (Automorphic adele graph continuation is non-empty)**.** *For adelic domain and value histories, any continuation graph produced by* Cont *is not* hsame*-equal to* e*.*

Proof. Transport the continuation result to e. Empty-result inversion forces the domain endpoint to be e, contradicting the non-empty property of the $\mathsf{Adele}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph cont nonempty`

Theorem 104.2 (Automorphic adele graph visible-context core is non-empty)**.** *For adelic domain and value histories, if a visible two-sided context displays* $\mathsf{Cont}(p \cdot d, v \cdot q, (p \cdot c) \cdot q)$*, then the exposed core* c *is not* hsame*-equal to* e*.*

Proof. Suffix cancellation removes the visible right context q, yielding $\mathsf{Cont}(p \cdot d, v, p \cdot c)$. Prefix cancellation removes the visible left context p, yielding $\mathsf{Cont}(d, v, c)$. The core non-emptiness theorem for automorphic adele graphs then rules out $c\mathsf{hsame}\mathsf{e}$. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context core nonempty`

Theorem 104.3 (Automorphic adele graph visible-context core determinacy)**.** *For adelic domain and value histories, if the same visible left and right contexts display two central continuation cores* c *and* c'*, then* $c\mathsf{hsame}c'$*. Both displayed cores are also not* hsame*-equal to* e*.*

Proof. Peel the common right context by suffix cancellation and the common left context by prefix cancellation. This recovers two central ledgers $\mathsf{Cont}(d, v, c)$ and $\mathsf{Cont}(d, v, c')$. Continuation determinacy compares the two recovered cores, and automorphic adele graph non-emptiness rules out emptiness for each core. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context core deterministic nonempty`

Theorem 104.4 (Automorphic adele graph visible-context value determinacy)**.** *For adelic domain and value histories, if the same visible left context and result display $v \cdot q$ and $v' \cdot q$ as the value side of the graph, then v*hsame*v'. The exposed graph is also not* hsame*-equal to* e*.*

Proof. Peel the common right context by suffix cancellation from both displayed continuations. The remaining ledgers share the same source and result, so continuation left cancellation compares the two value histories. Prefix cancellation recovers the central graph ledger, and automorphic adele graph non-emptiness rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context value deterministic`

Theorem 104.5 (Automorphic adele graph visible-context domain determinacy)**.** *For adelic domain and value histories, if the same visible value context and result display $p \cdot d$ and $p \cdot d'$ as the domain side of the graph, then d*hsame*d'. The exposed graph is also not* hsame*-equal to* e*.*

Proof. Peel the common right context by suffix cancellation from both displayed continuations. The remaining ledgers share the same value and result, so continuation right cancellation compares the prefixed domain histories. Left prefix cancellation removes p. The recovered central graph ledger supplies automorphic adele graph non-emptiness. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context domain deterministic`

Theorem 104.6 (Automorphic adele graph visible-context non-empty)**.** *For adelic domain and value histories, a displayed graph inside a visible left and right continuation context cannot be* hsame*-equal to* e*.*

Proof. Peel the right visible context and then the left visible context using continuation cancellation. The recovered central graph continuation is non-empty by the automorphic adele graph continuation theorem. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context nonempty`

Theorem 104.7 (Automorphic adele graph visible-context value readback)**.** *For adelic domain and value histories, if a visible two-sided context displays* $\mathsf{Cont}(p \cdot d, v \cdot q, (p \cdot graph) \cdot q)$ *and the same central graph is displayed by* $\mathsf{Cont}(d, v', graph)$*, then the visible value reads back to v*hsame*v'. The recovered graph is also not* hsame*-equal to* e*.*

Proof. Suffix cancellation removes the visible right context, and prefix cancellation removes the visible left context, recovering $\mathsf{Cont}(d, v, graph)$. Continuation left cancellation compares this recovered ledger with $\mathsf{Cont}(d, v', graph)$, while automorphic adele graph non-emptiness rules out an empty recovered graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context value readback`

Theorem 104.8 (Automorphic adele graph visible-context domain readback)**.** *For adelic domain and value histories, if a visible two-sided context displays* $\mathsf{Cont}(p \cdot d, v \cdot q, (p \cdot graph) \cdot q)$ *and the same central graph is displayed by* $\mathsf{Cont}(d', v, graph)$*, then the visible domain reads back to d*hsame*d'. The recovered graph is also not* hsame*-equal to* e*.*

Proof. Suffix cancellation removes the visible right context, and prefix cancellation removes the visible left context, recovering $\mathsf{Cont}(d, v, graph)$. Continuation right cancellation compares this recovered ledger with $\mathsf{Cont}(d', v, graph)$, while automorphic adele graph non-emptiness rules out an empty recovered graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context domain readback`

Theorem 104.9 (Automorphic adele graph visible-context displayed graph readback)**.** *For adelic domain and value histories, if a visible two-sided context displays* $\mathsf{Cont}(p{\cdot}d, v{\cdot}q, (p{\cdot}graph){\cdot}q)$ *and another central display gives* $\mathsf{Cont}(d, v, graph')$*, then the visible graph reads back to graph*hsame*graph′. The displayed graph is also not* hsame*-equal to* e*.*

Proof. Suffix cancellation removes the visible right context, and prefix cancellation removes the visible left context, recovering $\mathsf{Cont}(d, v, graph)$. Continuation determinacy compares this recovered ledger with $\mathsf{Cont}(d, v, graph')$. The displayed graph is non-empty by the automorphic adele graph continuation theorem. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context displayed graph readback`

Theorem 104.10 (Automorphic adele graph visible-context result is non-empty)**.** *For adelic domain and value histories, the whole visible-context result* $(p \cdot graph) \cdot q$ *of a displayed automorphic adele graph cannot be* hsame*-equal to* e*.*

Proof. If the visible result were empty, append-empty inversion first forces the right context q and the prefix $p \cdot graph$ to be empty, and then forces *graph* itself to be empty. The visible-context graph non-emptiness theorem gives the contradiction. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context result nonempty`

Theorem 104.11 (Automorphic adele graph visible result is non-empty)**.** *For adelic domain and value histories, any continuation result from a visible left-context domain* $p \cdot d$ *to a visible right-context value* $v \cdot q$ *cannot be* hsame*-equal to* e*.*

Proof. Transport the visible continuation result to e. Empty-result inversion then forces the source endpoint $p{\cdot}d$ to be empty. Append-empty inversion exposes $d\mathsf{hsame}\mathsf{e}$, contradicting the non-empty property of the $\mathsf{Adele}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible result nonempty`

Theorem 104.12 (Automorphic adele graph visible-context result hsame transport)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph graph, and graph*hsame*graph′, then the same visible context displays graph′. The transported graph is also not* hsame*-equal to* e*.*

Proof. Apply hsame congruence to the nested result $(p \cdot graph) \cdot q$, transport the displayed continuation ledger along that result comparison, and then apply visible-context graph non-emptiness to the transported display. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context result hsame transport`

Theorem 104.13 (Automorphic adele graph visible-context value hsame transport)**.** *For adelic domain and transported value histories, if a visible two-sided context displays an automorphic graph using value, and value*hsame*value′, then the same visible result displays the graph using value′. The graph remains not* hsame*-equal to* e*.*

Proof. Append congruence carries $value\mathsf{hsame}value'$ to the visible value leg $value \cdot q\mathsf{hsame}value' \cdot q$. Continuation hsame-transport rebuilds the visible ledger with $value'$, and visible-context non-emptiness applied to the transported ledger rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context value hsame transport`

Theorem 104.14 (Automorphic adele graph visible-context domain hsame transport)**.** *For adelic transported domain and value histories, if a visible two-sided context displays an automorphic graph using domain, and* $domain\mathsf{hsame}domain'$*, then the same visible result displays the graph using* $domain'$*. The graph remains not* hsame*-equal to* e*.*

Proof. Append congruence carries $domain\mathsf{hsame}domain'$ to the visible domain leg $p \cdot domain\mathsf{hsame}p \cdot domain'$. Continuation hsame-transport rebuilds the visible ledger with $domain'$, and visible-context non-emptiness applied to the transported ledger rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context domain hsame transport`

Theorem 104.15 (Automorphic adele graph visible-context endpoint hsame transport)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph using domain and value, and* $domain\mathsf{hsame}domain'$ *and* $value\mathsf{hsame}value'$*, then the same visible result displays the graph using the transported endpoints. The graph remains not* hsame*-equal to* e*.*

Proof. Append congruence carries the two endpoint classifiers to $p \cdot domain\mathsf{hsame}p \cdot domain'$ and $value \cdot q\mathsf{hsame}value' \cdot q$. Continuation hsame-transport rebuilds the displayed ledger with both endpoints transported. The original visible-context non-emptiness theorem rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context endpoint hsame transport`

Theorem 104.16 (Automorphic adele graph visible-context left hsame transport)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph using left context* p*, and* $p\mathsf{hsame}p'$*, then replacing the visible left context by* p' *preserves the continuation display. The graph remains not* hsame*-equal to* e*.*

Proof. Append congruence carries $p\mathsf{hsame}p'$ to both the visible source endpoint $p \cdot domain\mathsf{hsame}p' \cdot domain$ and the visible result endpoint $(p \cdot graph) \cdot q\mathsf{hsame}(p' \cdot graph) \cdot q$. Continuation hsame-transport rebuilds the visible ledger with left context p', and visible-context non-emptiness applied to that ledger rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context left hsame transport`

Theorem 104.17 (Automorphic adele graph visible-context right hsame transport)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph using right context* q*, and* $q\mathsf{hsame}q'$*, then replacing the visible right context by* q' *preserves the continuation display. The graph remains not* hsame*-equal to* e*.*

Proof. Append congruence carries $q\mathsf{hsame}q'$ to the visible value endpoint $value \cdot q\mathsf{hsame}value \cdot q'$ and to the visible result endpoint $(p \cdot graph) \cdot q\mathsf{hsame}(p \cdot graph) \cdot q'$. Continuation hsame-transport rebuilds the visible ledger with right context q', and visible-context non-emptiness applied to that ledger rules out an empty graph. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context right hsame transport`

Theorem 104.18 (Automorphic adele graph visible-context core readback)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph and that whole visible result is* hsame*-equal to the same context around a core, then the graph reads back to the core. The exposed core is also not* hsame*-equal to* e*.*

Proof. Reassociate both displayed visible results, then cancel the common left and right append contexts to recover $graph\mathsf{hsame}core$. Transport the displayed continuation ledger from the graph result to the core result and apply the visible-context core non-emptiness theorem. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context core readback`

Theorem 104.19 (Automorphic adele graph visible-context core continuation readback)**.** *For adelic domain and value histories, if a visible two-sided context displays an automorphic graph and the visible result is* hsame*-equal to the same context around a core, then the core itself is displayed by* $\mathsf{Cont}(domain, value, core)$*. Moreover* $graph\mathsf{hsame}core$*, and the core is not* hsame*-equal to* e*.*

Proof. Transport the visible continuation ledger from the graph endpoint to the core endpoint. Peel the right suffix and then the left prefix to recover $\mathsf{Cont}(domain, value, core)$. The existing core readback theorem supplies $graph\mathsf{hsame}core$ and core non-emptiness. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible context core cont readback`

Theorem 104.20 (Automorphic adele graph visible-result context factorization)**.** *For adelic domain and value histories, if a visible result is* hsame*-equal to* $(p \cdot graph) \cdot q$*, then the exposed graph is displayed by* $\mathsf{Cont}(domain, value, graph)$ *and is not* hsame*-equal to* e*.*

Proof. Transport the visible continuation ledger along the displayed result hsame. Peel the right suffix and then the left prefix to recover the central continuation, and apply adelic graph non-emptiness to that central ledger. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph visible result context factorizes`

104.2 The certificate

Definition 104.21 (Automorphic adele graph carrier)**.** *For a history* g*, the predicate* AutomorphicAdeleGraphCarrier *holds exactly when there are histories* $domain, value : \mathsf{BHist}$ *such that*

$$\mathsf{AdeleHistoryCarrier}(domain) \wedge \mathsf{AdeleHistoryCarrier}(value) \wedge \mathsf{Cont}(domain, value, g).$$

Lean4 (definition) `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraphCarrier`

Theorem 104.22 (Automorphic adele graph carrier hsame transport)**.** *If* $\mathsf{AutomorphicAdeleGraphCarrier}(g)$ *and* $g\mathsf{hsame}g'$*, then* $\mathsf{AutomorphicAdeleGraphCarrier}(g')$*.*

Proof. Unpack the graph-carrier witness for g into histories $domain, value : \mathsf{BHist}$, adele endpoint witnesses for both histories, and a continuation witness $\mathsf{Cont}(domain, value, g)$.

Apply continuation-result hsame transport along $g\mathsf{hsame}g'$. This keeps the hidden domain and value histories fixed and yields a transported continuation witness $\mathsf{Cont}(domain, value, g')$.

Repack the same adele endpoint witnesses together with the transported continuation ledger. The resulting witness is exactly $\mathsf{AutomorphicAdeleGraphCarrier}(g')$. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraphCarrier hsame transport`

1. Traditional target: automorphic graph carrier transport.
2. BEDC source: $\mathsf{Adele}^{\uparrow}$ together with Cont over histories.
3. Carrier: $\mathsf{AutomorphicAdeleGraphCarrier}$.
4. Classifier: hsame on BHist.
5. Stability obligation: Definition 104.22.
6. Ledger: the hidden domain, value, and continuation witnesses.
7. Core theorem: Definition 104.22.
8. Exactness: continuation-result transport.
9. Standard bridge: adelic graph presentation.
10. Status: $\mathsf{ProofTarget}$.

Theorem 104.23 (Automorphic adele graph semantic name certificate)**.** *The automorphic adele graph carrier, used as source predicate, pattern predicate, and ledger policy, forms a* $\mathsf{SemanticNameCert}$ *with classifier* hsame.

Proof. Apply Definition 101.31 to obtain an inhabited adele history carrier. Use the same adele endpoint as the domain and value endpoint, and apply the BEDC continuation constructor to obtain a graph endpoint.

The classifier laws are the reflexivity, symmetry, and transitivity laws for hsame. Since the source, pattern, and ledger policies are the same graph carrier predicate, their inclusion obligations are identity maps on the same witness package.

For carrier transport, apply Definition 104.22. The theorem supplies the reusable movement of a graph-carrier witness across hsame, so the certificate uses it as its shared source, pattern, and ledger transport. □

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph semanticNameCert`

Theorem 104.24 (Automorphic public certificate export)**.** *The adele-domain row, adele-value row, displayed graph carrier, context readbacks,* hsame *transports, and semantic certificate repack into the public* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *export for downstream consumers. The export keeps the* $\mathsf{Adele}^{\uparrow}$ *domain endpoint, the* $\mathsf{Adele}^{\uparrow}$ *value endpoint, the displayed graph history, and the* Cont *witness visible, and it exposes only the classifier, stability, ledger, and exactness rows already present in the automorphic adele graph certificate.*

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph public certificate export`

Proof. Apply the visible-context core readback Definition 104.18. It recovers the carried graph core from any displayed visible context.

Next apply the endpoint transport rows Definition 104.15 and the graph carrier hsame transport Definition 104.22. These rows keep the adele-domain endpoint, adele-value endpoint, and graph carrier stable under the declared classifier.

Finally apply Definition 104.23. Its semantic certificate supplies the shared source, pattern, and ledger policies. Repacking these visible rows gives the public $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ export without adding a Langlands bridge, spectral decomposition, or analytic continuation claim. □

Theorem 104.25 (Automorphic standard adelic graph bridge boundary)**.** *Any standard adelic-graph bridge for* $\mathsf{Automorphic}^{\uparrow}$ *may read only the* $\mathsf{Adele}^{\uparrow}$ *domain endpoint,* $\mathsf{Adele}^{\uparrow}$ *value endpoint, displayed graph history,* Cont *witness, visible-context readbacks,* hsame *transports, semantic certificate rows, and* Pkg *provenance from Definition 104.24. It does not supply Langlands correspondence, automorphic representation theory, spectral decomposition, analytic continuation, or an* L*-function bridge.*

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicUp StdBridge`

Proof. Project the public certificate export through Definition 104.24. Its visible rows are the two $\mathsf{Adele}^{\uparrow}$ endpoints, the displayed graph history, the continuation witness, classifier transports, semantic certificate rows, and package provenance.

Use the visible-context core readback of Definition 104.18 together with the endpoint transport rows of Definition 104.15. These rows keep the domain, value, and graph endpoint determined only through the declared hsame classifier.

Finally repack the same rows as the standard adelic graph boundary. Since the public export contains no row for Langlands correspondence, representation theory, spectral decomposition, analytic continuation, or L-function data, none of those claims is supplied by the bridge boundary. □

Theorem 104.26 (Automorphic mature adelic graph consumer coverage)**.** *Every downstream consumer of the standard adelic graph boundary receives exactly the* $\mathsf{Automorphic}^{\uparrow}$ *carrier, visible-context core, domain, and value readbacks, result factorization,* hsame *transport rows, semantic* NameCert *export, and standard adelic graph bridge boundary of Definition 104.25.*

The consumer boundary receives no Langlands bridge, L-function analytic continuation, spectral decomposition, or representation-theoretic structure.

Lean4 ✓ `BEDC.Derived.AutomorphicUp.AutomorphicTasteGate single carrier alignment`

Proof. Begin with the carrier and displayed visible context. Apply Definition 104.18 to recover the carried graph core, and use Definition 104.15 to keep the adele-domain and adele-value endpoints fixed up to the declared hsame classifier.

Next apply Definition 104.20 together with Definition 104.22. These rows ensure that any consumer of the boundary reads the same carrier, endpoint, context, continuation, factorization, and transport data already present in the graph packet.

Apply Definition 104.23. The semantic certificate supplies the shared source, pattern, ledger, exactness, and transport rows exported by the public NameCert surface.

Finally use Definition 104.25. The standard boundary repacks exactly those carrier, readback, factorization, transport, certificate, and package rows. Since none of the cited rows carries Langlands correspondence, L-function analytic continuation, spectral decomposition, or representation-theoretic structure, no downstream consumer receives such data. □

1. Traditional target: automorphic graph names.
2. BEDC source: $\mathsf{Adele}^{\uparrow}$ together with Cont over histories.
3. Carrier: the predicate $\mathsf{AutomorphicAdeleGraphCarrier}(g)$ displayed above.
4. Classifier: hsame on BHist.
5. Stability: the hsame equivalence laws and continuation-result transport for graph endpoints.
6. Ledger: the hidden domain and value adele endpoints together with the continuation witness.

7. Core theorem: Definition 104.23.
8. Exactness: graph endpoint generation by Cont from carried adele domain and value endpoints.
9. Standard bridge: adelic graph presentation.
10. Status: ProofTarget.

Closure status for $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$.

Constructive story: An automorphic graph starts with two admitted Adele$^{\uparrow}$ histories, one serving as the domain endpoint and one as the value endpoint, and the kernel continuation displays their graph history; without both adele carriers and the continuation witness, the history is outside the carrier. The classifier is ordinary history-sameness on these carried graph endpoints, with transport theorems peeling or replacing visible left and right contexts so that the underlying domain, value, and graph core remain determined. The source, pattern, and ledger are the same graph-carrier predicate: the source accepts exactly these adele-domain, adele-value, and continuation packages, the pattern records non-emptiness, context determinacy, endpoint readback, and result factorization, and the ledger keeps the hidden domain, value, continuation, and context witnesses available. Since the adele carrier is inhabited and nonempty, a displayed adele endpoint can be used on both sides to inhabit the graph carrier, while hsame supplies reflexivity, symmetry, and transitivity and continuation-result transport closes the carrier under classified graph endpoints. A history h lies in the signature gap of an Automorphic$^{\uparrow}$ token p when the source-domain accepts h as such an adele graph, the signature relation pairs it with an answer graph and evidence ledger, and token introduction names that answer as p; concretely, the central graph exposed after cancelling any visible contexts has the domain and value encoded by p. The visible-context rows give the constructive content usually hidden behind invariance language: they prove that adding contexts does not change the recovered core, domain, or value except through the declared classifier. The chapter registers this adele-continuation graph construction under the name Automorphic$^{\uparrow}$; it introduces no Langlands bridge, automorphic representation theory, spectral decomposition, or analytic continuation of L-functions.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$, maturePackageClosed)

Scope closed: The named carrier, visible-context determinism and readback, hsame transport, result factorization, carrier transport, semantic certificate blocks, standard adelic graph bridge boundary of Definition 104.25, and mature consumer coverage of Definition 104.26 fix the carrier, classifier, stability, ledger, exactness, bridge-boundary, and consumer-boundary obligations over Adele$^{\uparrow}$ and Cont.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.AutomorphicUp.AutomorphicAdeleGraph\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: No Langlands bridge, automorphic representation theory, L-function analytic continuation, spectral decomposition, or machine-checked standard bridge is claimed.

Upgrade path: No further upgrade remains on the formal bridge axis for the recorded standard bridge surface.

Chapter 105

A Concrete Naming Certificate for $\mathsf{CatLimit}^{\uparrow}$

The categorical-limit interface $\mathsf{CatLimit}^{\uparrow}$ packages a cone over a diagram in a $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) certified universal among all such cones. It depends on $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62) and is dual to $\mathsf{NameCert}_{\mathsf{CatColimit}^{\uparrow}}$ (chapter 106).

105.1 Carrier definition

105.2 The certificate

Definition 105.1 (Fixed-diagram cone comparison data)**.** *Fix a* $\mathsf{Cat}^{\uparrow}$ *certificate* $\mathcal{C}$*, an indexing category* $\mathcal{J}$*, and a certified* $\mathsf{Functor}^{\uparrow}$ *diagram* $D : \mathcal{J} \to \mathcal{C}$*. Write* $\mathrm{Mor}_{\mathcal{C}}(A, B; f)$ *for the certified assertion that* f *is a morphism* $A \to B$*,* $\mathrm{MorEq}_{\mathcal{C}}(A, B; f, g)$ *for the morphism classifier between parallel morphisms* $f, g : A \to B$*, and* $\mathrm{Comp}_{\mathcal{C}}(A, B, C; f, g, h)$ *for the certified composition witness saying that* $h : A \to C$ *classifies the composite of* $f : A \to B$ *followed by* $g : B \to C$*.*

For D*-cones* λ *and* μ *with apexes* L *and* M*, the predicate*

$$\mathrm{ConeMor}_D(\lambda, \mu; f)$$

means that $\mathrm{Mor}_{\mathcal{C}}(L, M; f)$ *holds and, for every object* j *of* $\mathcal{J}$*, there is a component composite* $c_j : L \to D(j)$ *such that*

$$\mathrm{Comp}_{\mathcal{C}}(L, M, D(j); f, \mu_j, c_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(L, D(j); c_j, \lambda_j).$$

A certified limiting cone is written

$$\mathrm{LimCone}_D(L, \lambda).$$

It consists of a D*-cone* λ *with apex* L *and the universal field that every* D*-cone* χ *with apex* X *admits a morphism* $m : X \to L$ *satisfying* $\mathrm{ConeMor}_D(\chi, \lambda; m)$*, with any two such morphisms classified by* $\mathrm{MorEq}_{\mathcal{C}}(X, L; -, -)$*.*

Lean4 (definition) `BEDC.Derived.CatLimitUp.CatLimitConeMor`

Lemma 105.2 (Composition closure for cone morphisms)**.** *Let λ, μ, ν be D-cones with apexes L, M, N. Suppose*

$$\mathrm{ConeMor}_D(\lambda, \mu; f)$$

for $f : L \to M$, and

$$\mathrm{ConeMor}_D(\mu, \nu; g)$$

for $g : M \to N$. If

$$\mathrm{Comp}_{\mathcal{C}}(L, M, N; f, g, h),$$

then

$$\mathrm{ConeMor}_D(\lambda, \nu; h).$$

Proof. The composition field of the $\mathsf{Cat}^{\uparrow}$ certificate gives $\mathrm{Mor}_{\mathcal{C}}(L, N; h)$ from the displayed composition witness and the morphism fields of f and g.

Fix an object j of $\mathcal{J}$. From $\mathrm{ConeMor}_D(\mu, \nu; g)$, obtain a composite $d_j : M \to D(j)$ with

$$\mathrm{Comp}_{\mathcal{C}}(M, N, D(j); g, \nu_j, d_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(M, D(j); d_j, \mu_j).$$

From $\mathrm{ConeMor}_D(\lambda, \mu; f)$, obtain a composite $e_j : L \to D(j)$ with

$$\mathrm{Comp}_{\mathcal{C}}(L, M, D(j); f, \mu_j, e_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(L, D(j); e_j, \lambda_j).$$

The associativity field of the $\mathsf{Cat}^{\uparrow}$ certificate applies to

$$L \xrightarrow{f} M \xrightarrow{g} N \xrightarrow{\nu_j} D(j).$$

Together with the witnesses for $h = g \circ f$ and $d_j = \nu_j \circ g$, it supplies a composite $r_j : L \to D(j)$ with

$$\mathrm{Comp}_{\mathcal{C}}(L, N, D(j); h, \nu_j, r_j)$$

and a classifier comparison between r_j and the composite of f followed by d_j. The morphism-classifier congruence field transports

$$\mathrm{MorEq}_{\mathcal{C}}(M, D(j); d_j, \mu_j)$$

across right composition by f, giving a classifier comparison between the composite of f followed by d_j and the composite e_j of f followed by μ_j. Transitivity of $\mathrm{MorEq}_{\mathcal{C}}$ then yields

$$\mathrm{MorEq}_{\mathcal{C}}(L, D(j); r_j, \lambda_j).$$

Since the object j was arbitrary, the component witnesses r_j assemble the component ledger for $\mathrm{ConeMor}_D(\lambda, \nu; h)$. □

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor comp closed`

Lemma 105.3 (Identity cone morphism)**.** *If λ is a D-cone with apex L, then*

$$\mathrm{ConeMor}_D(\lambda, \lambda; 1_L).$$

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor empty identity`

Proof. The identity field of the $\mathsf{Cat}^{\uparrow}$ certificate gives $\mathrm{Mor}_{\mathcal{C}}(L, L; 1_L)$. Fix an object j of $\mathcal{J}$. The unit field for the component $\lambda_j : L \to D(j)$ supplies a composite $c_j : L \to D(j)$ with

$$\mathrm{Comp}_{\mathcal{C}}(L, L, D(j); 1_L, \lambda_j, c_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(L, D(j); c_j, \lambda_j).$$

These witnesses, indexed by the objects of $\mathcal{J}$, give the cone-morphism ledger for $\mathrm{ConeMor}_D(\lambda, \lambda; 1_L)$. □

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor identity`
Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor identity comparison`

Theorem 105.4 (Fixed-diagram cone-morphism category package)**.** *For a fixed certified diagram $D : \mathcal{J} \to \mathcal{C}$ in a $\mathsf{Cat}^{\uparrow}$ certificate, the data whose objects are D-cones and whose arrows from λ to μ are witnesses of*

$$\mathrm{ConeMor}_D(\lambda, \mu; f)$$

has the certified category surface induced by $\mathcal{C}$. The identity arrow at a cone λ is supplied by

$$\mathrm{ConeMor}_D(\lambda, \lambda; 1_L),$$

and a composable pair

$$\mathrm{ConeMor}_D(\lambda, \mu; f), \qquad \mathrm{ConeMor}_D(\mu, \nu; g)$$

together with

$$\mathrm{Comp}_{\mathcal{C}}(L, M, N; f, g, h)$$

supplies the composite arrow

$$\mathrm{ConeMor}_D(\lambda, \nu; h).$$

Moreover, composition of cone morphisms is stable under the morphism classifier inherited from $\mathrm{MorEq}_{\mathcal{C}}$*: replacing any displayed arrow or component composite by a parallel* $\mathrm{MorEq}_{\mathcal{C}}$*-classified representative gives a classified representative of the same cone-morphism composite.*

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor comp classifier transport`

Proof. The identity row is exactly Definition 105.3. For every D-cone λ with apex L, that lemma supplies the certified morphism $1_L : L \to L$ and the component ledger proving $\mathrm{ConeMor}_D(\lambda, \lambda; 1_L)$.

For composition, take cone morphisms $\lambda \to \mu$ and $\mu \to \nu$ represented by f and g, and take a certified composite witness $\mathrm{Comp}_{\mathcal{C}}(L, M, N; f, g, h)$. Applying Definition 105.2 gives the required cone-morphism ledger $\mathrm{ConeMor}_D(\lambda, \nu; h)$.

It remains to record stability under the inherited classifier. The proof of Definition 105.2 uses the associativity field of $\mathsf{Cat}^{\uparrow}$ to compare the two component-composite parenthesizations and the classifier-congruence field to transport $\mathrm{MorEq}_{\mathcal{C}}$ across right composition. Those same fields show that replacing f, g, h, or any component composite by a parallel $\mathrm{MorEq}_{\mathcal{C}}$ representative preserves the displayed cone-morphism composite up to the same classifier. □

Proposition 105.5 (Limiting-cone endomorphism rigidity)**.** *If*

$$\mathrm{LimCone}_D(L, \lambda)$$

and

$$\mathrm{ConeMor}_D(\lambda, \lambda; a),$$

then

$$\mathrm{MorEq}_{\mathcal{C}}(L, L; a, 1_L).$$

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitLimCone endomorphism rigidity`

Proof. By Definition 105.3, the identity morphism satisfies

$$\mathrm{ConeMor}_D(\lambda, \lambda; 1_L).$$

The universal uniqueness field of $\mathrm{LimCone}_D(L, \lambda)$ applies to the source cone λ and to the two cone morphisms $a, 1_L : L \to L$. It gives

$$\mathrm{MorEq}_{\mathcal{C}}(L, L; a, 1_L).$$

□

Theorem 105.6 (CatLimit apex uniqueness from universal cones)**.** *Let $D : \mathcal{J} \to \mathcal{C}$ be a fixed certified* $\mathsf{Functor}^{\uparrow}$ *diagram in a* $\mathsf{Cat}^{\uparrow}$ *certificate $\mathcal{C}$. If*

$$\mathrm{LimCone}_D(L, \lambda)$$

and

$$\mathrm{LimCone}_D(L', \lambda')$$

are two certified limiting cones over D, then there exist comparison morphisms

$$u : L \to L'$$

and

$$v : L' \to L$$

such that

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u)$$

and

$$\mathrm{ConeMor}_D(\lambda', \lambda; v).$$

Moreover, there are composite results $c_L : L \to L$ and $c_{L'} : L' \to L'$ satisfying

$$\mathrm{Comp}_{\mathcal{C}}(L, L', L; u, v, c_L)$$

and

$$\mathrm{Comp}_{\mathcal{C}}(L', L, L'; v, u, c_{L'})$$

with

$$\mathrm{MorEq}_{\mathcal{C}}(L, L; c_L, 1_L)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(L', L'; c_{L'}, 1_{L'}).$$

Lean4 ✓ BEDC.Derived.CatLimitUp.CatLimitLimCone comparison identities

Proof. Apply the universal existence field of $\mathrm{LimCone}_D(L', \lambda')$ to the D-cone λ with apex L. This supplies a morphism $u : L \to L'$ and a cone-morphism ledger

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u).$$

Apply the universal existence field of $\mathrm{LimCone}_D(L, \lambda)$ to the D-cone λ' with apex L'. This supplies a morphism $v : L' \to L$ and a cone-morphism ledger

$$\mathrm{ConeMor}_D(\lambda', \lambda; v).$$

The composition field of $\mathcal{C}$ supplies a composite result $c_L : L \to L$ with

$$\mathrm{Comp}_{\mathcal{C}}(L, L', L; u, v, c_L).$$

Using Definition 105.2 with the two cone morphisms

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u)$$

and

$$\mathrm{ConeMor}_D(\lambda', \lambda; v),$$

one obtains

$$\mathrm{ConeMor}_D(\lambda, \lambda; c_L).$$

Now Definition 105.5, applied to $\mathrm{LimCone}_D(L, \lambda)$ and this cone endomorphism, gives

$$\mathrm{MorEq}_{\mathcal{C}}(L, L; c_L, 1_L).$$

The composition field of $\mathcal{C}$ also supplies a composite result $c_{L'} : L' \to L'$ with

$$\mathrm{Comp}_{\mathcal{C}}(L', L, L'; v, u, c_{L'}).$$

Using Definition 105.2 with the two cone morphisms

$$\mathrm{ConeMor}_D(\lambda', \lambda; v)$$

and

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u),$$

one obtains

$$\mathrm{ConeMor}_D(\lambda', \lambda'; c_{L'}).$$

Now Definition 105.5, applied to $\mathrm{LimCone}_D(L', \lambda')$ and this cone endomorphism, gives

$$\mathrm{MorEq}_{\mathcal{C}}(L', L'; c_{L'}, 1_{L'}).$$

□

Corollary 105.7 (Comparison-fiber collapse)**.** *Under the hypotheses of Definition 105.6, if* $u_1, u_2 : L \to L'$ *both satisfy*

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u_1)$$

and

$$\mathrm{ConeMor}_D(\lambda, \lambda'; u_2),$$

then

$$\mathrm{MorEq}_{\mathcal{C}}(L, L'; u_1, u_2).$$

If $v_1, v_2 : L' \to L$ *both satisfy*

$$\mathrm{ConeMor}_D(\lambda', \lambda; v_1)$$

and

$$\mathrm{ConeMor}_D(\lambda', \lambda; v_2),$$

then

$$\mathrm{MorEq}_{\mathcal{C}}(L', L; v_1, v_2).$$

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitLimCone comparison fiber collapse`

Proof. The two morphisms $u_1, u_2 : L \to L'$ are cone morphisms from the same source cone λ to the same limiting cone λ'. The universal uniqueness field of $\mathrm{LimCone}_D(L', \lambda')$ applied to the source cone λ gives

$$\mathrm{MorEq}_{\mathcal{C}}(L, L'; u_1, u_2).$$

The two morphisms $v_1, v_2 : L' \to L$ are cone morphisms from the same source cone λ' to the same limiting cone λ. The universal uniqueness field of $\mathrm{LimCone}_D(L, \lambda)$ applied to the source cone λ' gives

$$\mathrm{MorEq}_{\mathcal{C}}(L', L; v_1, v_2).$$

□

Proposition 105.8 (Field dependency of the comparison construction)**.** *The construction in Definition 105.6 is obtained from the following fields:*

1. *the universal existence field of each limiting cone, applied once to the other cone;*
2. *the* $\mathsf{Cat}^{\uparrow}$ *composition field, applied once to* u *followed by* v *and once to* v *followed by* u*;*
3. *the associativity and morphism-classifier congruence fields used in Definition 105.2;*
4. *the identity and unit fields used in Definition 105.3;*
5. *the universal uniqueness field of each limiting cone, applied to its own cone endomorphism.*

No object equality, quotient carrier, host-level function extensionality, or external isomorphism principle enters the construction.

Proof. The first universal existence application is made to $\mathrm{LimCone}_D(L', \lambda')$ with source cone λ, and it produces the morphism $u : L \to L'$ together with $\mathrm{ConeMor}_D(\lambda, \lambda'; u)$. The second universal existence application is made to $\mathrm{LimCone}_D(L, \lambda)$ with source cone λ', and it produces the morphism $v : L' \to L$ together with $\mathrm{ConeMor}_D(\lambda', \lambda; v)$.

The $\mathsf{Cat}^{\uparrow}$ composition field produces c_L from u followed by v, and it produces $c_{L'}$ from v followed by u. The passage from these raw composites to cone endomorphisms is exactly Definition 105.2; its proof uses associativity to compare $\nu_j \circ (g \circ f)$ with $(\nu_j \circ g) \circ f$, morphism-classifier congruence

to replace $\nu_j \circ g$ by the corresponding cone component, and transitivity to reach the original source component.

The classification of an arbitrary cone endomorphism with the identity is exactly Definition 105.5. That proposition first uses Definition 105.3, whose proof uses only the identity and unit fields of the $\mathsf{Cat}^{\uparrow}$ certificate, and then applies the universal uniqueness field of the limiting cone to the pair consisting of the given cone endomorphism and the identity cone morphism.

Combining these field projections gives precisely the two comparison morphisms, the two composite witnesses, and the two identity classifications stated in Definition 105.6. Each displayed equality is an instance of $\mathrm{MorEq}_{\mathcal{C}}$, and every object-level comparison used in the construction is one of the fields listed above. □

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitLimCone comparison field dependency`

Definition 105.9 (Pointwise certified diagram equivalence)**.** *Fix* $\mathsf{Cat}^{\uparrow}$ *certificates* $\mathcal{J}$ *and* $\mathcal{C}$*, and certified* $\mathsf{Functor}^{\uparrow}$ *diagrams* $D, E : \mathcal{J} \to \mathcal{C}$*. A pointwise certified diagram equivalence between* D *and* E *is a pair of certified* $\mathsf{NatTrans}^{\uparrow}$ *carriers*

$$\alpha : D \Rightarrow E, \qquad \beta : E \Rightarrow D,$$

such that, for every object j *of* $\mathcal{J}$*, the component composites are classified as identities in* $\mathcal{C}$*. Thus there are composites* $d_j : D(j) \to D(j)$ *and* $e_j : E(j) \to E(j)$ *with*

$$\mathrm{Comp}_{\mathcal{C}}(D(j), E(j), D(j); \alpha_j, \beta_j, d_j) \quad \textit{and} \quad \mathrm{MorEq}_{\mathcal{C}}(D(j), D(j); d_j, 1_{D(j)}),$$

and

$$\mathrm{Comp}_{\mathcal{C}}(E(j), D(j), E(j); \beta_j, \alpha_j, e_j) \quad \textit{and} \quad \mathrm{MorEq}_{\mathcal{C}}(E(j), E(j); e_j, 1_{E(j)}).$$

The naturality, component-carrier, and component-classifier rows are exactly the rows required by the $\mathsf{NatTrans}^{\uparrow}$ *carrier and certificate obligations.*

Lean4 (definition) `BEDC.Derived.CatLimitUp.CatLimitPointwiseDiagramEquivalence`

Theorem 105.10 (CatLimit transport along pointwise certified diagram equivalence)**.** *Let* $\mathcal{J}$ *and* $\mathcal{C}$ *satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations, and let* $D, E : \mathcal{J} \to \mathcal{C}$ *be certified* $\mathsf{Functor}^{\uparrow}$ *diagrams. Suppose* (α, β) *is a pointwise certified diagram equivalence between* D *and* E *in the sense of Definition 105.9. If*

$$\mathrm{LimCone}_D(L, \lambda)$$

holds, and if λ^E *is the component family obtained by certified composition*

$$\mathrm{Comp}_{\mathcal{C}}(L, D(j), E(j); \lambda_j, \alpha_j, \lambda^E_j)$$

for every object j *of* $\mathcal{J}$*, then*

$$\mathrm{LimCone}_E(L, \lambda^E).$$

Proof. First verify the E-cone ledger for λ^E. For a displayed morphism $u : j \to k$ in $\mathcal{J}$, the D-cone law for λ classifies the composite of λ_j followed by $D(u)$ with λ_k. Naturality of α classifies α_j followed by $E(u)$ with $D(u)$ followed by α_k. Associativity and composition congruence from Definition 61.201 paste these two classifier comparisons and identify the composite of λ^E_j followed by $E(u)$ with λ^E_k. Thus λ^E is an E-cone with apex L.

Let χ be any E-cone with apex X. Form a D-component family χ^D by composing each component with β_j:

$$\mathrm{Comp}_{\mathcal{C}}(X, E(j), D(j); \chi_j, \beta_j, \chi^D_j).$$

The E-cone law for χ, naturality of β, and the same associativity and congruence fields show that χ^D is a D-cone. The universal existence field of $\mathrm{LimCone}_D(L, \lambda)$ therefore supplies a morphism $m : X \to L$ with

$$\mathrm{ConeMor}_D(\chi^D, \lambda; m).$$

For each j, the component comparison of this cone morphism classifies m followed by λ_j with χ_j followed by β_j. Composing that comparison on the right with α_j, using associativity, and then using the $E(j)$-identity row β_j followed by α_j classified with $1_{E(j)}$, gives a classifier comparison between m followed by λ^E_j and χ_j. The right identity law in $\mathcal{C}$ supplies the final comparison with χ_j. Hence

$$\mathrm{ConeMor}_E(\chi, \lambda^E; m),$$

which is the universal existence field for λ^E.

It remains to verify universal uniqueness. Let $m, n : X \to L$ both satisfy

$$\mathrm{ConeMor}_E(\chi, \lambda^E; m) \quad \text{and} \quad \mathrm{ConeMor}_E(\chi, \lambda^E; n).$$

For m, compose the component comparison $m; \lambda^E_j$ classified with χ_j on the right by β_j. Since λ^E_j is the composite of λ_j followed by α_j, associativity rewrites the left side as $m; \lambda_j; \alpha_j; \beta_j$. The $D(j)$-identity row classifies α_j followed by β_j with $1_{D(j)}$, and the right identity law reduces this to $m; \lambda_j$. The right side is the defining component χ^D_j. Thus

$$\mathrm{ConeMor}_D(\chi^D, \lambda; m).$$

The same argument gives

$$\mathrm{ConeMor}_D(\chi^D, \lambda; n).$$

The universal uniqueness field of $\mathrm{LimCone}_D(L, \lambda)$ applies to these two D-cone morphisms and yields

$$\mathrm{MorEq}_{\mathcal{C}}(X, L; m, n).$$

This is exactly the uniqueness field required for $\mathrm{LimCone}_E(L, \lambda^E)$. □

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitLimCone transport pointwise equiv`

Theorem 105.11 (Cone morphism classifier descent through natural-transformation whiskering)**.** *Let $\mathcal{J}$ and $\mathcal{C}$ satisfy the $\mathsf{Cat}^{\uparrow}$ certificate obligations. Let $D, E : \mathcal{J} \to \mathcal{C}$ be certified $\mathsf{Functor}^{\uparrow}$ diagrams, and let $\alpha : D \Rightarrow E$ be a certified $\mathsf{NatTrans}^{\uparrow}$ carrier with components $\alpha_j : D(j) \to E(j)$. Let λ and μ be D-cones with apexes L and M, and let λ^α and μ^α be the corresponding E-cones obtained by composing each cone leg with the component of α:*

$$\mathrm{Comp}_{\mathcal{C}}(L, D(j), E(j); \lambda_j, \alpha_j, \lambda^\alpha_j) \qquad \textit{and} \qquad \mathrm{Comp}_{\mathcal{C}}(M, D(j), E(j); \mu_j, \alpha_j, \mu^\alpha_j)$$

for every object j of $\mathcal{J}$, with image endpoints read through the certified endpoint-transport fields. If

$$\mathrm{ConeMor}_D(\lambda, \mu; f),$$

then

$$\mathrm{ConeMor}_E(\lambda^\alpha, \mu^\alpha; f).$$

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitConeMor nattrans whiskering descent`

Proof. By Definition 63.1, each α_j is a displayed morphism $D(j) \to E(j)$ in $\mathcal{C}$, and the naturality row is the cone-whiskering ledger that makes the displayed component families λ^α and μ^α into E-cones. If any object image is represented by a classified endpoint rather than by a literal endpoint, the source-target replacement row of Definition 62.78 and the endpoint-indexed transport fields of Definition 61.201 reindex the component morphisms before the following composition ledgers are formed.

Unpack $\mathrm{ConeMor}_D(\lambda, \mu; f)$ using Definition 105.1. It gives the apex morphism field $\mathrm{Mor}_{\mathcal{C}}(L, M; f)$ and, for each object j of $\mathcal{J}$, a displayed composite $d_j : L \to D(j)$ such that

$$\mathrm{Comp}_{\mathcal{C}}(L, M, D(j); f, \mu_j, d_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(L, D(j); d_j, \lambda_j).$$

The same apex morphism field is the apex morphism required for $\mathrm{ConeMor}_E(\lambda^\alpha, \mu^\alpha; f)$.

Fix j. The postwhiskered target leg is displayed by

$$\mathrm{Comp}_{\mathcal{C}}(M, D(j), E(j); \mu_j, \alpha_j, \mu_j^\alpha).$$

Composition closure in Definition 61.201 supplies a composite $s_j : L \to E(j)$ of f followed by μ_j^α:

$$\mathrm{Comp}_{\mathcal{C}}(L, M, E(j); f, \mu_j^\alpha, s_j).$$

It also supplies a composite $t_j : L \to E(j)$ of d_j followed by α_j:

$$\mathrm{Comp}_{\mathcal{C}}(L, D(j), E(j); d_j, \alpha_j, t_j).$$

Associativity in the same category certificate, applied to

$$L \xrightarrow{f} M \xrightarrow{\mu_j} D(j) \xrightarrow{\alpha_j} E(j),$$

compares the two bracketings and gives

$$\mathrm{MorEq}_{\mathcal{C}}(L, E(j); s_j, t_j).$$

The component classifier from the original cone morphism gives $d_j \sim \lambda_j$ at the endpoint pair $(L, D(j))$. Composition congruence with the fixed right factor α_j, together with the displayed definition of λ_j^α, transports this comparison to

$$\mathrm{MorEq}_{\mathcal{C}}(L, E(j); t_j, \lambda_j^\alpha).$$

Transitivity of the morphism classifier in Definition 61.201 composes the last two comparisons and yields

$$\mathrm{MorEq}_{\mathcal{C}}(L, E(j); s_j, \lambda_j^\alpha).$$

Thus, for the fixed object j, the composite s_j supplies the required E-cone-morphism component ledger for f from λ^α to μ^α. Since j was arbitrary, these ledgers assemble with the already supplied apex morphism field into $\mathrm{ConeMor}_E(\lambda^\alpha, \mu^\alpha; f)$. □

Theorem 105.12 (CatLimit transported limiting-cone comparison)**.** *Let $\mathcal{J}$ and $\mathcal{C}$ satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations, and let $D, E : \mathcal{J} \to \mathcal{C}$ be certified* $\mathsf{Functor}^{\uparrow}$ *diagrams. Suppose (α, β) is a pointwise certified diagram equivalence between D and E in the sense of Definition 105.9. Let*

$$\mathrm{LimCone}_D(L, \lambda)$$

be a certified limiting D-cone, and let λ^E be the transported E-cone obtained from the certified component composites

$$\mathrm{Comp}_{\mathcal{C}}(L, D(j), E(j); \lambda_j, \alpha_j, \lambda_j^E)$$

for every object j of $\mathcal{J}$. If

$$\mathrm{LimCone}_E(M, \mu)$$

is an independently certified limiting E-cone, then there exist comparison morphisms

$$u : L \to M \qquad \textit{and} \qquad v : M \to L$$

such that

$$\mathrm{ConeMor}_E(\lambda^E, \mu; u) \qquad \textit{and} \qquad \mathrm{ConeMor}_E(\mu, \lambda^E; v).$$

Moreover, there are composite results $c_L : L \to L$ and $c_M : M \to M$ satisfying

$$\mathrm{Comp}_{\mathcal{C}}(L, M, L; u, v, c_L) \qquad \textit{and} \qquad \mathrm{Comp}_{\mathcal{C}}(M, L, M; v, u, c_M)$$

with

$$\mathrm{MorEq}_{\mathcal{C}}(L, L; c_L, 1_L) \qquad \textit{and} \qquad \mathrm{MorEq}_{\mathcal{C}}(M, M; c_M, 1_M).$$

Proof. Apply Definition 105.10 to the pointwise certified diagram equivalence (α, β) and to the limiting D-cone $\mathrm{LimCone}_D(L, \lambda)$. The displayed component-composite ledgers for λ^E are exactly the input required by that transport theorem, so it gives

$$\mathrm{LimCone}_E(L, \lambda^E).$$

Now both limiting cones are read over the single certified diagram E:

$$\mathrm{LimCone}_E(L, \lambda^E) \qquad \text{and} \qquad \mathrm{LimCone}_E(M, \mu).$$

Apply Definition 105.6 to these two E-limiting cones. It supplies the comparison morphisms $u : L \to M$ and $v : M \to L$, the two cone-morphism ledgers over E, and composite results c_L and c_M for the two apex composites.

The identity classifiers produced by that comparison theorem are also classified in the target category certificate $\mathcal{C}$ while the cone-morphism ledgers are indexed by E. Hence the composites $u; v$ and $v; u$ are classified, under $\mathrm{MorEq}_{\mathcal{C}}$, as the respective identity morphisms 1_L and 1_M in precisely the transported target-diagram context claimed above. □

Lean4 ✓ `BEDC.Derived.CatLimitUp.CatLimitLimCone transported comparison`

Closure status for $\mathsf{NameCert}_{\mathsf{CatLimit}^{\uparrow}}$**.**

Constructive story: A categorical limit is built over an already certified category and a certified diagram by displaying a cone: an apex history, one component morphism history to each diagram value, and the comparison rows saying those components respect every diagram arrow. The carrier accepts only cones with these visible component witnesses and a limiting field that sends any other displayed cone to a comparison morphism into the apex; a cone without the component ledger or without the universal comparison row stays outside the surface, and two accepted comparisons are identified by the morphism classifier inherited from $\mathsf{Cat}^{\uparrow}$. The source is the fixed-diagram cone predicate, the pattern records identity cone morphisms, closure under composition, apex comparison identities, diagram-equivalence transport, and whiskering

descent, while the ledger keeps the apex, component composites, and universal comparison morphisms visible rather than replacing them by an external limit object. For the displayed seed surface, the name-certificate obligations are read from the same rows: a supplied limiting cone gives the carried endpoint, inherited morphism equality supplies reflexivity, symmetry, and transitivity, and the transport theorems keep carried cones and cone morphisms admitted after classifier replacement. A history h lies in the signature gap of a CatLimit$^{\uparrow}$ token p when the source-domain recognizes h as this fixed-diagram cone package, the signature relation pairs it with an answer cone and its component ledger, and the token-introduction relation names that answer as p; unfolded, h has exactly the displayed universal arrow into the limiting apex and the uniqueness comparison rows encoded by p. No standard bridge is attached here, so the outside categorical-limit reading remains only the internal Cat$^{\uparrow}$ and Functor$^{\uparrow}$ comparison surface. The chapter registers this fixed-diagram cone construction under the name CatLimit$^{\uparrow}$; the limit is a universal comparison package over existing histories, with no completeness axiom, host limit object, or external categorical library introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{CatLimit}\uparrow}$, maturePackageClosed)

Scope closed: The mature CatLimit$^{\uparrow}$ paper scope covers fixed-diagram cone comparison, transported limiting-cone comparison, natural-transformation whiskering descent, and the finite limit/colimit consumer surface over Cat$^{\uparrow}$, Functor$^{\uparrow}$, NatTrans$^{\uparrow}$, BHist, hsame, and Cont.

Formal verification: FormalStatus(NameCert$_{\mathsf{CatLimit}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CatLimitUp.CatLimitLimCone\_transported\_comparison`

Bridge status: none

Not claimed: This chapter does not close full limit theory, a standard categorical-limit bridge, completeness for arbitrary diagrams, uniqueness beyond the displayed comparison surface, or bridge verification.

Upgrade path: No further paper theory-axis upgrade is recorded for this displayed fixed-diagram limit surface; bridgeChecked status remains separate formal-axis work.

Chapter 106

A Concrete Naming Certificate for $\mathsf{CatColimit}^{\uparrow}$

The categorical-colimit interface $\mathsf{CatColimit}^{\uparrow}$ packages a cocone over a diagram certified couniversal among all such cocones. It depends on $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62) and is dual to $\mathsf{NameCert}_{\mathsf{CatLimit}^{\uparrow}}$ (chapter 105).

106.1 Carrier definition

106.2 The certificate

Definition 106.1 (Fixed-diagram cocone comparison data)**.** *Fix a* $\mathsf{Cat}^{\uparrow}$ *certificate* $\mathcal{C}$*, an indexing category* $\mathcal{J}$*, and a certified* $\mathsf{Functor}^{\uparrow}$ *diagram* $D : \mathcal{J} \to \mathcal{C}$*. Write* $\mathrm{Mor}_{\mathcal{C}}(A, B; f)$ *for the certified assertion that* f *is a morphism* $A \to B$*,* $\mathrm{MorEq}_{\mathcal{C}}(A, B; f, g)$ *for the morphism classifier between parallel morphisms* $f, g : A \to B$*, and* $\mathrm{Comp}_{\mathcal{C}}(A, B, C; f, g, h)$ *for the certified composition witness saying that* $h : A \to C$ *classifies the composite of* $f : A \to B$ *followed by* $g : B \to C$*.*

For a D*-cocone* κ *with vertex* K*, write its component at* j *as* $\kappa_j : D(j) \to K$*. For* D*-cocones* κ *and* μ *with vertices* K *and* M*, the predicate*

$$\mathrm{CoconeMor}_D(\kappa, \mu; f)$$

means that $\mathrm{Mor}_{\mathcal{C}}(K, M; f)$ *holds and, for every object* j *of* $\mathcal{J}$*, there is a component composite* $c_j : D(j) \to M$ *such that*

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, M; \kappa_j, f, c_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), M; c_j, \mu_j).$$

A certified colimiting cocone is written

$$\mathrm{ColimCocone}_D(K, \kappa).$$

It consists of a D*-cocone* κ *with vertex* K *and the couniversal field that every* D*-cocone* χ *with vertex* X *admits a morphism* $m : K \to X$ *satisfying* $\mathrm{CoconeMor}_D(\kappa, \chi; m)$*, with any two such morphisms classified by* $\mathrm{MorEq}_{\mathcal{C}}(K, X; -, -)$*.*

Lean4 (definition) `BEDC.Derived.CatColimitUp.CatColimitCoconeMor`

Lemma 106.2 (Composition closure for cocone morphisms)**.** *Let κ, μ, ν be D-cocones with vertices K, M, N. Suppose*

$$\mathrm{CoconeMor}_D(\kappa, \mu; f)$$

for $f : K \to M$, and

$$\mathrm{CoconeMor}_D(\mu, \nu; g)$$

for $g : M \to N$. If

$$\mathrm{Comp}_{\mathcal{C}}(K, M, N; f, g, h),$$

then

$$\mathrm{CoconeMor}_D(\kappa, \nu; h).$$

Proof. The composition field of the $\mathsf{Cat}^{\uparrow}$ certificate gives $\mathrm{Mor}_{\mathcal{C}}(K, N; h)$ from the displayed composition witness and the morphism fields of f and g.

Fix an object j of $\mathcal{J}$. From $\mathrm{CoconeMor}_D(\kappa, \mu; f)$, obtain a composite $e_j : D(j) \to M$ with

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, M; \kappa_j, f, e_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), M; e_j, \mu_j).$$

From $\mathrm{CoconeMor}_D(\mu, \nu; g)$, obtain a composite $d_j : D(j) \to N$ with

$$\mathrm{Comp}_{\mathcal{C}}(D(j), M, N; \mu_j, g, d_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), N; d_j, \nu_j).$$

The associativity field of the $\mathsf{Cat}^{\uparrow}$ certificate applies to

$$D(j) \xrightarrow{\kappa_j} K \xrightarrow{f} M \xrightarrow{g} N.$$

Together with the witnesses for h as f followed by g and for e_j as κ_j followed by f, it supplies a composite $r_j : D(j) \to N$ with

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, N; \kappa_j, h, r_j)$$

and a classifier comparison between r_j and the composite of e_j followed by g. The morphism-classifier congruence field transports

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), M; e_j, \mu_j)$$

across right composition by g, giving a classifier comparison between the composite of e_j followed by g and d_j. Transitivity of $\mathrm{MorEq}_{\mathcal{C}}$ then yields

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), N; r_j, \nu_j).$$

Since j was arbitrary, the component witnesses r_j assemble the component ledger for $\mathrm{CoconeMor}_D(\kappa, \nu; h)$. □

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor comp closed`

Lemma 106.3 (Identity cocone morphism)**.** *If κ is a D-cocone with vertex K, then*

$$\mathrm{CoconeMor}_D(\kappa, \kappa; 1_K).$$

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor empty identity`

Proof. The identity field of the $\mathsf{Cat}^{\uparrow}$ certificate gives $\mathrm{Mor}_{\mathcal{C}}(K, K; 1_K)$. Fix an object j of $\mathcal{J}$. The unit field for the component $\kappa_j : D(j) \to K$ supplies a composite $c_j : D(j) \to K$ with

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, K; \kappa_j, 1_K, c_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), K; c_j, \kappa_j).$$

These witnesses, indexed by the objects of $\mathcal{J}$, give the cocone-morphism ledger for $\mathrm{CoconeMor}_D(\kappa, \kappa; 1_K)$. □

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor identity`
Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor identity comparison`

Proposition 106.4 (Colimiting-cocone endomorphism rigidity)**.** *If*

$$\mathrm{ColimCocone}_D(K, \kappa)$$

and

$$\mathrm{CoconeMor}_D(\kappa, \kappa; a),$$

then

$$\mathrm{MorEq}_{\mathcal{C}}(K, K; a, 1_K).$$

Proof. By Definition 106.3, the identity morphism satisfies

$$\mathrm{CoconeMor}_D(\kappa, \kappa; 1_K).$$

The universal uniqueness field of $\mathrm{ColimCocone}_D(K, \kappa)$ applies to the target cocone κ and to the two cocone morphisms $a, 1_K : K \to K$. It gives

$$\mathrm{MorEq}_{\mathcal{C}}(K, K; a, 1_K).$$

□

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitLimCocone endomorphism rigidity`

Theorem 106.5 (CatColimit vertex co-uniqueness from universal cocones)**.** *Let $D : \mathcal{J} \to \mathcal{C}$ be a fixed certified* $\mathsf{Functor}^{\uparrow}$ *diagram in a* $\mathsf{Cat}^{\uparrow}$ *certificate $\mathcal{C}$. If*

$$\mathrm{ColimCocone}_D(C, \kappa)$$

and

$$\mathrm{ColimCocone}_D(C', \kappa')$$

are two certified colimiting cocones over D, then there exist comparison morphisms

$$u : C \to C'$$

and

$$v : C' \to C$$

such that

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u)$$

and

$$\mathrm{CoconeMor}_D(\kappa', \kappa; v).$$

Moreover, there are composite results $c_C : C \to C$ *and* $c_{C'} : C' \to C'$ *satisfying*

$$\mathrm{Comp}_{\mathcal{C}}(C, C', C; u, v, c_C)$$

and

$$\mathrm{Comp}_{\mathcal{C}}(C', C, C'; v, u, c_{C'})$$

with

$$\mathrm{MorEq}_{\mathcal{C}}(C, C; c_C, 1_C)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(C', C'; c_{C'}, 1_{C'}).$$

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor composite endomorphisms empty`
Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitLimCocone comparison identities`

Proof. Apply the universal existence field of $\mathrm{ColimCocone}_D(C, \kappa)$ to the D-cocone κ' with vertex C'. This supplies a morphism $u : C \to C'$ and a cocone-morphism ledger

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u).$$

Apply the universal existence field of $\mathrm{ColimCocone}_D(C', \kappa')$ to the D-cocone κ with vertex C. This supplies a morphism $v : C' \to C$ and a cocone-morphism ledger

$$\mathrm{CoconeMor}_D(\kappa', \kappa; v).$$

The composition field of $\mathcal{C}$ supplies a composite result $c_C : C \to C$ with

$$\mathrm{Comp}_{\mathcal{C}}(C, C', C; u, v, c_C).$$

Using Definition 106.2 with the two cocone morphisms

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u)$$

and

$$\mathrm{CoconeMor}_D(\kappa', \kappa; v),$$

one obtains

$$\mathrm{CoconeMor}_D(\kappa, \kappa; c_C).$$

Now Definition 106.4, applied to $\mathrm{ColimCocone}_D(C, \kappa)$ and this cocone endomorphism, gives

$$\mathrm{MorEq}_{\mathcal{C}}(C, C; c_C, 1_C).$$

The composition field of $\mathcal{C}$ also supplies a composite result $c_{C'} : C' \to C'$ with

$$\mathrm{Comp}_{\mathcal{C}}(C', C, C'; v, u, c_{C'}).$$

Using Definition 106.2 with the two cocone morphisms

$$\mathrm{CoconeMor}_D(\kappa', \kappa; v)$$

and

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u),$$

one obtains

$$\mathrm{CoconeMor}_D(\kappa', \kappa'; c_{C'}).$$

Now Definition 106.4, applied to $\mathrm{ColimCocone}_D(C', \kappa')$ and this cocone endomorphism, gives

$$\mathrm{MorEq}_{\mathcal{C}}(C', C'; c_{C'}, 1_{C'}).$$

□

Theorem 106.6 (Cocone morphism classifier descent through natural-transformation prewhiskering)**.** *Let $\mathcal{J}$ and $\mathcal{C}$ satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations. Let $D, E : \mathcal{J} \to \mathcal{C}$ be certified* $\mathsf{Functor}^{\uparrow}$ *diagrams, and let $\beta : E \Rightarrow D$ be a certified* $\mathsf{NatTrans}^{\uparrow}$ *carrier with components $\beta_j : E(j) \to D(j)$. Let κ and μ be D-cocones with vertices K and M, and let κ^β and μ^β be the corresponding E-cocones obtained by precomposing each cocone leg with the component of β:*

$$\mathrm{Comp}_{\mathcal{C}}(E(j), D(j), K; \beta_j, \kappa_j, \kappa_j^\beta) \qquad \textit{and} \qquad \mathrm{Comp}_{\mathcal{C}}(E(j), D(j), M; \beta_j, \mu_j, \mu_j^\beta)$$

for every object j of $\mathcal{J}$, with image endpoints read through the certified endpoint-transport fields. If

$$\mathrm{CoconeMor}_D(\kappa, \mu; f),$$

then

$$\mathrm{CoconeMor}_E(\kappa^\beta, \mu^\beta; f).$$

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitCoconeMor nattrans prewhiskering descent`

Proof. By Definition 63.1, each β_j is a displayed morphism $E(j) \to D(j)$ in $\mathcal{C}$, and the naturality row is the cocone-prewhiskering ledger that makes the displayed component families κ^β and μ^β into E-cocones. If an object image is represented by a classified endpoint rather than by a literal endpoint, the source-target replacement row of Definition 62.78 and the endpoint-indexed transport fields of Definition 61.201 reindex the component morphisms before the following composition ledgers are formed.

Unpack $\mathrm{CoconeMor}_D(\kappa, \mu; f)$ using Definition 106.1. It gives the vertex morphism field $\mathrm{Mor}_{\mathcal{C}}(K, M; f)$ and, for each object j of $\mathcal{J}$, a displayed composite $d_j : D(j) \to M$ such that

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, M; \kappa_j, f, d_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), M; d_j, \mu_j).$$

The same vertex morphism field is the vertex morphism required for $\mathrm{CoconeMor}_E(\kappa^\beta, \mu^\beta; f)$.

Fix j. Composition closure in Definition 61.201 supplies a composite $s_j : E(j) \to M$ of κ_j^β followed by f:

$$\mathrm{Comp}_{\mathcal{C}}(E(j), K, M; \kappa_j^\beta, f, s_j).$$

It also supplies a composite $t_j : E(j) \to M$ of β_j followed by d_j:

$$\mathrm{Comp}_{\mathcal{C}}(E(j), D(j), M; \beta_j, d_j, t_j).$$

Associativity in the same category certificate, applied to

$$E(j) \xrightarrow{\beta_j} D(j) \xrightarrow{\kappa_j} K \xrightarrow{f} M,$$

compares the two bracketings and gives

$$\mathrm{MorEq}_{\mathcal{C}}(E(j), M; s_j, t_j).$$

The component classifier from the original cocone morphism gives $d_j \sim \mu_j$ at the endpoint pair $(D(j), M)$. Composition congruence with the fixed left factor β_j, together with the displayed definition of μ_j^β, transports this comparison to

$$\mathrm{MorEq}_{\mathcal{C}}(E(j), M; t_j, \mu_j^\beta).$$

Transitivity of the morphism classifier in Definition 61.201 composes the last two comparisons and yields

$$\mathrm{MorEq}_{\mathcal{C}}(E(j), M; s_j, \mu_j^\beta).$$

Thus, for the fixed object j, the composite s_j supplies the required E-cocone-morphism component ledger for f from κ^β to μ^β. Since j was arbitrary, these ledgers assemble with the already supplied vertex morphism field into $\mathrm{CoconeMor}_E(\kappa^\beta, \mu^\beta; f)$. □

Corollary 106.7 (CatColimit comparison-fiber collapse)**.** *Let $D : \mathcal{J} \to \mathcal{C}$ be a fixed certified* $\mathsf{Functor}^\uparrow$ *diagram in a* $\mathsf{Cat}^\uparrow$ *certificate $\mathcal{C}$. Let κ be a D-cocone with vertex C, and let κ' be a D-cocone with vertex C'. If*

$$\mathrm{ColimCocone}_D(C, \kappa)$$

and if $u_1, u_2 : C \to C'$ satisfy

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u_1)$$

and

$$\mathrm{CoconeMor}_D(\kappa, \kappa'; u_2),$$

then

$$\mathrm{MorEq}_{\mathcal{C}}(C, C'; u_1, u_2).$$

Proof. Unpack the certified colimiting-cocone field from Definition 106.1. Its uniqueness clause says that, for every D-cocone χ with vertex X, any two morphisms $m_1, m_2 : C \to X$ satisfying

$$\mathrm{CoconeMor}_D(\kappa, \chi; m_1)$$

and

$$\mathrm{CoconeMor}_D(\kappa, \chi; m_2)$$

are classified by

$$\mathrm{MorEq}_{\mathcal{C}}(C, X; m_1, m_2).$$

Apply this clause with $\chi = \kappa'$, $X = C'$, $m_1 = u_1$, and $m_2 = u_2$. The two displayed cocone-morphism ledgers are exactly the hypotheses, so the required classifier is

$$\mathrm{MorEq}_{\mathcal{C}}(C, C'; u_1, u_2).$$

□

Theorem 106.8 (CatColimit transport along pointwise certified diagram equivalence)**.** *Let $\mathcal{J}$ and $\mathcal{C}$ satisfy the* $\mathsf{Cat}^{\uparrow}$ *certificate obligations, and let $D, E : \mathcal{J} \to \mathcal{C}$ be certified* $\mathsf{Functor}^{\uparrow}$ *diagrams. Suppose (α, β) is a pointwise certified diagram equivalence between D and E in the sense of Definition 105.9. If*

$$\mathrm{ColimCocone}_D(K, \kappa)$$

holds, and if κ^E is the component family obtained by certified composition

$$\mathrm{Comp}_{\mathcal{C}}(E(j), D(j), K; \beta_j, \kappa_j, \kappa^E_j)$$

for every object j of $\mathcal{J}$, with endpoints read through the certified source-target replacement fields, then

$$\mathrm{ColimCocone}_E(K, \kappa^E).$$

Lean4 ✓ `BEDC.Derived.CatColimitUp.CatColimitLimCocone transport pointwise equiv`

Proof. By Definition 105.9, the components $\alpha_j : D(j) \to E(j)$ and $\beta_j : E(j) \to D(j)$ are certified $\mathsf{NatTrans}^{\uparrow}$ component morphisms, and their pointwise composites are classified with the corresponding identities. Thus for every object j there are composites

$$\mathrm{Comp}_{\mathcal{C}}(D(j), E(j), D(j); \alpha_j, \beta_j, d_j) \quad \text{and} \quad \mathrm{MorEq}_{\mathcal{C}}(D(j), D(j); d_j, 1_{D(j)}),$$

and

$$\mathrm{Comp}_{\mathcal{C}}(E(j), D(j), E(j); \beta_j, \alpha_j, e_j) \quad \text{and} \quad \mathrm{MorEq}_{\mathcal{C}}(E(j), E(j); e_j, 1_{E(j)}).$$

The component-carrier and endpoint replacement rows are those of Definition 63.1, Definition 62.78, and Definition 61.201.

First verify that κ^E is an E-cocone. For a displayed morphism $u : j \to k$ of $\mathcal{J}$, the E-cocone law compares the composite of $E(u)$ followed by κ^E_k with κ^E_j. The defining composite for κ^E_k and associativity identify the first bracketing with the chain

$$E(j) \xrightarrow{E(u)} E(k) \xrightarrow{\beta_k} D(k) \xrightarrow{\kappa_k} K.$$

Naturality of β compares $E(u)$ followed by β_k with β_j followed by $D(u)$. Composition congruence with the fixed right factor κ_k, followed by the D-cocone law for κ, compares this chain with

$$E(j) \xrightarrow{\beta_j} D(j) \xrightarrow{\kappa_j} K,$$

which is κ^E_j by definition. Transitivity of the morphism classifier gives the E-cocone component row for κ^E.

Let χ be any E-cocone with vertex X. Define a D-component family χ^D by

$$\mathrm{Comp}_{\mathcal{C}}(D(j), E(j), X; \alpha_j, \chi_j, \chi^D_j)$$

for every object j. Naturality of α, the E-cocone law for χ, associativity, and composition congruence show that χ^D is a D-cocone with vertex X: for $u : j \to k$, the composite of $D(u)$ followed by α_k followed by χ_k is classified with the composite of α_j followed by $E(u)$ followed by χ_k, and then with α_j followed by χ_j.

The universal existence field of $\mathrm{ColimCocone}_D(K, \kappa)$ applied to this D-cocone supplies a morphism $m : K \to X$ such that

$$\mathrm{CoconeMor}_D(\kappa, \chi^D; m).$$

Unpacking Definition 106.1, for each object j there is a composite $r_j : D(j) \to X$ with

$$\mathrm{Comp}_{\mathcal{C}}(D(j), K, X; \kappa_j, m, r_j)$$

and

$$\mathrm{MorEq}_{\mathcal{C}}(D(j), X; r_j, \chi_j^D).$$

Let $s_j : E(j) \to X$ be a composite of κ_j^E followed by m. Associativity compares s_j with the composite of β_j followed by r_j. Composition congruence with the fixed left factor β_j transports $r_j \sim \chi_j^D$ to a comparison between β_j followed by r_j and β_j followed by χ_j^D. Since χ_j^D is the composite of α_j followed by χ_j, associativity rewrites the latter as $(\beta_j; \alpha_j); \chi_j$. The $E(j)$ identity row of the pointwise equivalence, composed with the fixed right factor χ_j, and the left identity row of the $\mathsf{Cat}^{\uparrow}$ certificate classify this composite with χ_j. Transitivity gives

$$\mathrm{MorEq}_{\mathcal{C}}(E(j), X; s_j, \chi_j).$$

The vertex morphism field for m is already present in $\mathrm{CoconeMor}_D(\kappa, \chi^D; m)$. Hence

$$\mathrm{CoconeMor}_E(\kappa^E, \chi; m),$$

which is the universal existence field required for the E-cocone κ^E.

It remains to verify universal uniqueness. Let $m, n : K \to X$ both satisfy

$$\mathrm{CoconeMor}_E(\kappa^E, \chi; m) \quad \text{and} \quad \mathrm{CoconeMor}_E(\kappa^E, \chi; n).$$

For m, unpack the first E-cocone-morphism ledger. For each object j there is a composite $a_j : E(j) \to X$ of κ_j^E followed by m classified with χ_j. Composition congruence with the fixed left factor α_j gives a comparison between α_j followed by a_j and α_j followed by χ_j. Using the defining composite $\kappa_j^E = \beta_j; \kappa_j$ and associativity, the left side is classified with $(\alpha_j; \beta_j); \kappa_j; m$. The $D(j)$ identity row of the pointwise equivalence, composed first with κ_j and then with m, and the left identity rows of $\mathcal{C}$, classify this with $\kappa_j; m$. The right side is χ_j^D by definition. Thus the component rows assemble with the vertex morphism field from the E-cocone morphism to give

$$\mathrm{CoconeMor}_D(\kappa, \chi^D; m).$$

The same construction applied to the ledger for n gives

$$\mathrm{CoconeMor}_D(\kappa, \chi^D; n).$$

The universal uniqueness field of $\mathrm{ColimCocone}_D(K, \kappa)$, applied to the D-cocone χ^D and these two cocone morphisms, yields

$$\mathrm{MorEq}_{\mathcal{C}}(K, X; m, n).$$

This is exactly the uniqueness field for $\mathrm{ColimCocone}_E(K, \kappa^E)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CatColimit}^{\uparrow}}$.

Constructive story: A categorical colimit is built by reversing the displayed cone direction: over a certified category and fixed certified diagram, a cocone carries a vertex history, component morphisms from each diagram value into that vertex, and comparison ledgers showing how every component factors through a cocone morphism. The carrier accepts only cocones with those visible component witnesses and a couniversal field sending the vertex to every other displayed cocone; data without the component ledger or without the couniversal comparison row are not part of this surface, and sameness of comparisons is inherited from the $\mathsf{Cat}^{\uparrow}$ morphism classifier. The source is the fixed-diagram cocone

predicate, the pattern records identity cocone morphisms, closure of cocone morphisms under composition, colimiting-cocone comparison identities, natural-transformation prewhiskering descent, and pointwise diagram-equivalence transport, while the ledger keeps vertices, component composites, and comparison arrows on the BEDC history surface. The seed obligations follow through these same rows: a supplied colimiting cocone inhabits the carried surface, morphism equality gives reflexivity, symmetry, and transitivity, and the transport theorem preserves admitted cocones and cocone morphisms under classified replacement. A history h lies in the signature gap of a CatColimit$^{\uparrow}$ token p when the source-domain admits h as this fixed-diagram cocone package, the signature relation pairs it with an answer cocone and evidence ledger, and token introduction names that answer as p; concretely, h exhibits the couniversal arrow out of the colimiting vertex and the uniqueness rows encoded by p. The pointwise diagram-equivalence transport row supplies the paper bridge surface for the fixed-diagram cocone package, but the chapter does not translate the package to a host colimit construction. It registers the dual fixed-diagram cocone construction under the name CatColimit$^{\uparrow}$; colimits here are couniversal comparison rows over existing histories, with no cocompleteness axiom, quotient vertex primitive, or external categorical library introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{CatColimit}\uparrow}$, maturePackageClosed)

Scope closed: The bridged paper scope is the fixed-diagram cocone comparison surface, identity and composition cocone morphisms, universal-cocone comparison identities, and pointwise certified diagram-equivalence transport.

Formal verification: FormalStatus(NameCert$_{\mathsf{CatColimit}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CatColimitUp.CatColimitLimCocone\_transport\_pointwise\_equiv`

Bridge status: paperBridge

Not claimed: This chapter does not claim a bridgeChecked target, host colimit construction, cocompleteness axiom, quotient vertex primitive, or external categorical library.

Upgrade path: No paper-axis upgrade remains beyond maintaining the fixed-diagram cocone consumer catalogue.

Chapter 107

A Concrete Naming Certificate for $\mathsf{Adjunction}^{\uparrow}$

The adjunction interface $\mathsf{Adjunction}^{\uparrow}$ packages a pair of functors $F \dashv G$ between two categories together with the natural-isomorphism witness on hom-sets. It depends on $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62) and $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$ (chapter 63), and is the canonical structure underlying free/forgetful and Galois-style correspondences.

107.1 Carrier definition

Definition 107.1 (Adjunction unit-counit carrier)**.** *For histories* $p, q, a, unit, counit, left, right$: BHist*,* $\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)$ *holds when unit is a natural-transformation prefix component from* p *to* q *at* a*, counit is the corresponding component from* q *to* p*, and the two triangle witnesses are the continuations* $\mathsf{Cont}(unit, counit, left)$ *and* $\mathsf{Cont}(counit, unit, right)$*.*

Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier`

Definition 107.2 (Adjunction unit-counit alternating ledger)**.** *For histories* $unit, counit, depth$: BHist*,* $\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth)$ *is the recursive ledger that alternates the two boundary components along a unary depth. The empty depth returns* e*, a one-headed depth appends the current unit component to the ledger with the two boundary components exchanged, and the zero-headed branch returns* e*.*

Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating`

Theorem 107.3 (Adjunction unit-counit diagonal empty components exactness)**.** *For histories* $p, q, a, composite$: BHist*,*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, \mathsf{e}, \mathsf{e}, composite, composite) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{UnaryHistory}(a) \wedge p \sim q \wedge composite \sim \mathsf{e}.$$

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier empty components exact`

Proof. Both displayed natural-transformation components are empty identity components. Their exactness theorem gives unary histories for p, q, a and $p \sim q$. The diagonal continuation result of the two empty components is history-same as e. Conversely, $p \sim q$ supplies both empty identity components, and $composite \sim \mathsf{e}$ supplies both empty continuation witnesses. □

Theorem 107.4 (Adjunction empty unit-counit prefix exactness)**.** *For histories p, q, a : BHist, empty unit and counit prefix components, together with the empty composite continuation, are exact for unary prefixes, a unary object, and prefix equality:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentCarrier}(q, p, a, \mathsf{e}) \wedge \mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e}) \iff \mathsf{Unary}(p) \wedge \mathsf{Unary}(q) \wedge \mathsf{Unary}(a) \wedge \mathsf{hsame}(p, q).$$

Proof. Each empty prefix component is read through the natural-transformation empty-identity exactness theorem. The unit component supplies the unary data and $p\mathsf{hsame}q$. Conversely, the unary data and $p\mathsf{hsame}q$ reconstruct the unit component; symmetry reconstructs the counit component; the empty composite is the reflexive continuation at e. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.Adjunction empty unit counit prefix iff`

Theorem 107.5 (Adjunction unit-counit empty component exactness)**.** *For histories $p, q, a, left, right$: BHist,*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, \mathsf{e}, \mathsf{e}, left, right)$$

holds exactly when p, q, a are unary histories, $p\mathsf{hsame}q$, and both triangle results satisfy $left\mathsf{hsame}\mathsf{e}$ and $right\mathsf{hsame}\mathsf{e}$.

Proof. Each empty component is read by the natural-transformation empty-identity theorem, yielding the unary prefix data and $p\mathsf{hsame}q$. The two triangle continuations then compare with the empty right-unit continuation. Conversely, the same unary data and prefix equality rebuild both empty component carriers, and the displayed empty triangle results rebuild the two continuations. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier empty components iff`

Theorem 107.6 (Adjunction left-cycle empty component exactness)**.** *For histories $p, q, a, unit, counit, left, right$: BHist, an adjunction unit-counit carrier whose left triangle result is empty is equivalent to unary prefix data, two-way prefix equality, and all four unit, counit, left-triangle, and right-triangle histories being empty:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right) \wedge \mathsf{hsame}(left, \mathsf{e}) \iff \mathsf{Unary}(p) \wedge \mathsf{Unary}(q) \wedge \mathsf{Unary}(a) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, p) \wedge \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left,$$

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier left cycle empty components iff`

Proof. The left empty triangle turns the unit-counit continuation into an empty prefix-component cycle. The prefix-cycle theorem extracts $p\mathsf{hsame}q$, $q\mathsf{hsame}p$, and empty unit and counit component histories. Reading the unit component as an empty natural-transformation identity supplies the unary witnesses for p, q, and a. The right triangle continuation then transports across the empty counit and unit witnesses to the empty continuation.

Conversely, the unary data and the two prefix equalities rebuild the empty unit and counit natural-transformation components. The empty witnesses for unit, counit, left, and right rebuild the two continuation ledgers and retain the displayed left-cycle empty witness. □

Theorem 107.7 (Adjunction right-cycle empty component exactness)**.** *For histories* $p, q, a, unit, counit, left, right$ BHist*, an adjunction unit-counit carrier whose right triangle result is empty is equivalent to unary prefix data, two-way prefix equality, and all four unit, counit, left-triangle, and right-triangle histories being empty:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right) \wedge \mathsf{hsame}(right, \mathsf{e})$$
$$\iff \mathsf{Unary}(p) \wedge \mathsf{Unary}(q) \wedge \mathsf{Unary}(a) \wedge \mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, p) \wedge \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left,$$

Proof. Exchange the unit and counit component rows and read the right triangle as the left triangle of the exchanged carrier. The left-cycle exactness theorem extracts the unary witnesses, the two prefix equalities, and the empty component histories. Reassembling the component rows in the original order gives the displayed equivalence. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier right cycle empty components iff`

Theorem 107.8 (Adjunction unit-counit triangle result determinism)**.** *If an adjunction unit-counit carrier displays triangle results* $left$ *and* $right$*, then any other continuations with the same unit-counit inputs display the same results up to* hsame*.*

Proof. Project the two continuation witnesses from the carrier and apply continuation determinism separately to the left and right triangle rows. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier triangle results deterministic`

Theorem 107.9 (Adjunction unit-counit triangle results coincide)**.** *Let* $unit : p \Rightarrow q$ *and* $counit : q \Rightarrow p$ *be prefix natural-transformation components at* a*. If the two triangle ledgers are displayed by*

$$\mathsf{Cont}(unit, counit, left) \qquad \textit{and} \qquad \mathsf{Cont}(counit, unit, right),$$

then $left\,\mathsf{hsame}\,right$*.*

Proof. The first continuation is a vertical composite prefix component from p to p, hence the endomorphism component exactness identifies $left$ with e. The second continuation is the symmetric composite from q to q, so the same exactness identifies $right$ with e. Transitivity of hsame gives $left\,\mathsf{hsame}\,right$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitTriangle results hsame`

Theorem 107.10 (Adjunction unit-counit endomorphism boundary emptiness)**.** *For histories* $p, a, unit, counit, left, right$: BHist*, if*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right),$$

then both boundary components are empty:

$$\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}).$$

Proof. Both displayed boundary components are prefix natural-transformation endomorphism components over the same prefix p and object a. The component exactness readback identifies the unit component with e and then identifies the counit component with e. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier endomorphism unit counit empty`

Theorem 107.11 (Adjunction unit-counit alternating unary)**.** *If unit, counit, and depth are unary histories, then the alternating unit-counit ledger*

$$\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth)$$

is unary.

Proof. Induct on the unary depth. The empty depth yields the empty ledger. A zero-headed depth is impossible for a unary history. In the one-headed case, the recursive tail is unary by the induction hypothesis with *unit* and *counit* exchanged, and unary append closure combines it with the unary front component. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating unary`

Theorem 107.12 (Adjunction alternating zero-headed result exclusion)**.** *For an adjunction unit-counit carrier, unary depth makes the alternating unit-counit ledger unary. Therefore it cannot be history-same to a zero-headed result:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right) \wedge \mathsf{Unary}(depth) \wedge \mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}$$

Proof. The unit and counit components of the displayed carrier are natural transformation component histories, hence each component history is unary. Together with unary depth, the alternating unary theorem makes the displayed alternating ledger unary. Transporting unary membership across the result hsame would make $\mathsf{E0}(z)$ unary, contradicting unary zero-headed exclusion. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier alternating zero headed result absurd`

Theorem 107.13 (Adjunction positive-depth alternating unit nonempty)**.** *For histories* $unit, counit, depth : \mathsf{BHist}$*, suppose depth is unary, depth is not history-same as* e*, and unit is not history-same as* e*. Then the alternating unit-counit ledger at that depth is not history-same as* e*:*

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{e}) \to \bot.$$

Proof. Unary nonempty depth has a one-headed tail. At such a depth the alternating ledger unfolds to an append whose left component is *unit*. Append nonemptiness therefore follows from the nonempty unit component. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating result nonempty of positive depth unit nonempty`

Theorem 107.14 (Adjunction positive-depth alternating continuation readback)**.** *For histories* $unit, counit, depth : \mathsf{BHist}$*, suppose depth is unary and not history-same as* e*. Then there is a unary tail tail such that the alternating unit-counit ledger at depth is displayed by the first continuation step*

$$\mathsf{Cont}(unit, \mathsf{AdjunctionUnitCounitAlternating}(counit, unit, tail), \mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth)$$

Proof. Unary nonempty depth has a one-headed tail. Unfolding the alternating ledger at that one-headed branch exposes exactly the continuation append whose left component is *unit* and whose right component is the alternating ledger with the boundary components exchanged. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating positive depth cont readback`

Theorem 107.15 (Adjunction alternating append positive suffix continuation readback)**.** *If $suffix$ is unary and not history-same as* e*, then the alternating unit-counit ledger over* $\mathsf{append}(pref, suffix)$ *exposes a successor continuation with the same prefix inside the exchanged tail:*

$$\exists tail.\ \mathsf{Unary}(tail) \wedge \mathsf{Cont}(unit, \mathsf{AdjunctionUnitCounitAlternating}(counit, unit, \mathsf{append}(pref, tail)), \mathsf{AdjunctionUnitCour}$$

Proof. Unary nonempty suffixes have a one-headed tail. Substituting $suffix = \mathsf{E1}(tail)$ makes the appended depth one-headed with payload $\mathsf{append}(pref, tail)$. The alternating ledger unfolds at that branch to append *unit* to the exchanged alternating tail, which is exactly the displayed continuation. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating append positive suffix cont readback`

Theorem 107.16 (Adjunction append-positive suffix empty result forces empty unit)**.** *If $suffix$ is unary and not history-same as* e*, then an empty result for the alternating unit-counit ledger over* $\mathsf{append}(pref, suffix)$ *forces the visible unit component to be empty:*

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, \mathsf{append}(pref, suffix)), \mathsf{e}) \Longrightarrow \mathsf{hsame}(unit, \mathsf{e}).$$

Proof. The append-positive suffix readback writes the displayed alternating ledger as a continuation whose left endpoint is *unit*. Transporting that continuation to the empty result and applying empty-result inversion for continuation forces the left endpoint *unit* to be empty. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating append positive suffix result empty unit empty`

Theorem 107.17 (Adjunction positive-tail empty result forces empty components)**.** *Suppose depth is unary, depth* $= \mathsf{E1}(tail)$*, tail is unary, and tail is not history-same as* e*. If the alternating unit-counit ledger at depth is history-same as* e*, then both displayed components are empty:*

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{e}) \Longrightarrow \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}).$$

Proof. Substitute $depth = \mathsf{E1}(tail)$ and unfold the positive branch of the alternating ledger. Empty append inversion gives *unit*hsamee and an empty result for the exchanged alternating ledger over *tail*. Since *tail* is unary and nonempty, the positive-depth empty-result readback applied to that exchanged ledger gives *counit*hsamee. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating positive tail result empty components empty`

Theorem 107.18 (Adjunction positive-depth empty result forces empty unit)**.** *For unary nonempty depth, if the alternating unit-counit ledger is history-same as* e*, then the visible unit component is history-same as* e*:*

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{e}) \quad \Longrightarrow \quad \mathsf{hsame}(unit, \mathsf{e}).$$

Proof. Positive-depth continuation readback writes the alternating ledger as a continuation whose left component is *unit*. Transporting this continuation along the empty result and applying empty-result inversion forces the left component to be empty. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating positive depth result empty unit empty`

Theorem 107.19 (Adjunction positive-depth empty alternating prefix readback)**.** *For histories* $p, q, a, unit, counit, left, right, depth : \mathsf{BHist}$*, suppose an adjunction unit-counit carrier holds, depth is unary and nonempty, and the alternating unit-counit ledger is history-same to* e*. Then the source and target prefixes of the unit component are history-same:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right) \wedge \mathsf{Unary}(depth) \wedge (\mathsf{hsame}(depth, \mathsf{e}) \to \bot) \wedge \mathsf{hsame}(\mathsf{Adjuncti}$$

Proof. The positive-depth empty-result readback forces $unit\mathsf{hsamee}$. Transporting the unit component to the empty component case lets natural-transformation empty-identity exactness read back the endpoint-prefix equality $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier alternating positive depth empty prefix same`

Theorem 107.20 (Adjunction visible unit positive-depth empty-result exclusion)**.** *For unary nonempty depth, an alternating unit-counit ledger whose displayed unit component has a visible head cannot continue to an empty result:*

$$\mathsf{Cont}(\mathsf{AdjunctionUnitCounitAlternating}(\mathsf{E0}(unit), counit, depth), y, r) \wedge r\mathsf{hsamee} \Longrightarrow \bot,$$

and the same statement holds with $\mathsf{E1}(unit)$ *in place of* $\mathsf{E0}(unit)$*.*

Proof. Transport the continuation along $r\mathsf{hsamee}$ and invert the empty result. The positive-depth empty-result readback then forces the visible unit component itself to be history-same to e, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating positive depth visible unit continuation empty result absurd`

Theorem 107.21 (Adjunction positive-depth unit-empty readback)**.** *For unary nonempty depth, if* $unit\mathsf{hsamee}$*, then the alternating unit-counit ledger at depth is history-same to the tail readback with the boundary components exchanged: there is a unary tail such that*

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{AdjunctionUnitCounitAlternating}(counit, unit, tail)).$$

Proof. Positive-depth continuation readback supplies a unary tail and a continuation from *unit* into the exchanged alternating tail. Replacing *unit* by e turns that continuation into the left-unit law, so its result reads back to the exchanged tail. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating positive depth unit empty readback`

Theorem 107.22 (Adjunction unit-counit alternating empty components closed)**.** *If both boundary components are empty and depth is unary, then the alternating unit-counit ledger is empty:*

$$\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{Unary}(depth) \quad \Longrightarrow \quad \mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth$$

Proof. Induct on the unary depth. The empty depth returns the empty ledger, while a zero-headed depth is incompatible with unary history. In the one-headed case, the recursion exchanges the empty unit and counit components; the induction hypothesis collapses the tail ledger, and append by the empty front component keeps the endpoint empty. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating empty components closed`

Theorem 107.23 (Adjunction unit-counit alternating endomorphism empty)**.** *For histories* $p, a, unit, counit, left,$ BHist*, if*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

and depth is unary, then the alternating unit-counit ledger is empty:

$$\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{e}).$$

Proof. The endomorphism boundary theorem identifies both boundary components with e. Induct on the unary depth. The empty depth is immediate. In the one-headed case, the recursive ledger has the boundary components exchanged; the induction hypothesis collapses that tail ledger to e, and the outer append by the empty unit component preserves the empty endpoint. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitAlternating endomorphism empty`

Theorem 107.24 (Adjunction unit-counit endomorphism triangle emptiness)**.** *For histories* $p, a, unit, counit, left,$ BHist*, if*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right),$$

then both triangle legs are empty:

$$\mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}).$$

Proof. The endomorphism boundary theorem first identifies both *unit* and *counit* with e. Continuation transport carries the two displayed triangle continuations to the right-unit continuation at e, yielding empty left and right triangle results. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier endomorphism triangles empty`

Theorem 107.25 (Adjunction unit-counit endomorphism empty components iff)**.** *For histories* $p, a, unit, counit, left, right$: BHist*, the endomorphism adjunction unit-counit carrier is equivalent to unary data for p, a, unit, and counit, together with empty unit, counit, left triangle, and right triangle components:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

$$\iff \mathsf{Unary}(p) \wedge \mathsf{Unary}(a) \wedge \mathsf{Unary}(unit) \wedge \mathsf{Unary}(counit) \wedge \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(righ$$

Proof. The forward direction decomposes the two endomorphism natural-transformation components, yielding the unary witnesses and the empty unit and counit witnesses. The triangle-empty theorem supplies the two displayed triangle results. Conversely, empty unit and counit components rebuild the two endomorphism natural-transformation carriers, and the empty triangle witnesses rebuild the two continuation legs. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier endomorphism empty components iff`

Theorem 107.26 (Adjunction unit-counit endomorphism total collapse)**.** *For histories* $p, a, unit, counit, left, righ$ BHist*, if*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right),$$

then all four endomorphism fields are empty:

$$\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}).$$

Proof. Apply Theorem 107.10 to the displayed carrier. This gives the two boundary witnesses $\mathsf{hsame}(unit, \mathsf{e})$ and $\mathsf{hsame}(counit, \mathsf{e})$.

Apply Theorem 107.24 to the same carrier. This gives the two triangle-leg witnesses $\mathsf{hsame}(left, \mathsf{e})$ and $\mathsf{hsame}(right, \mathsf{e})$.

Combining the two boundary witnesses with the two triangle-leg witnesses assembles the displayed four-fold conjunction. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier endomorphism total collapse`

Theorem 107.27 (Adjunction unit-counit endomorphism total alternating collapse)**.** *For histories* $p, a, unit, counit, left, right, depth : \mathsf{BHist}$*, if*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

and $\mathsf{UnaryHistory}(depth)$*, then the endomorphism boundary, triangle, and alternating ledger all collapse to the empty history:*

$$\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}) \wedge \mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, cou$$

Proof. Apply Theorem 107.10 to the displayed carrier. This gives the two boundary witnesses $\mathsf{hsame}(unit, \mathsf{e})$ and $\mathsf{hsame}(counit, \mathsf{e})$.

Apply Theorem 107.24 to the same carrier. This gives the two triangle-leg witnesses $\mathsf{hsame}(left, \mathsf{e})$ and $\mathsf{hsame}(right, \mathsf{e})$.

Apply Theorem 107.23 to the displayed carrier and the unary-depth witness. This gives the alternating-ledger witness $\mathsf{hsame}(\mathsf{AdjunctionUnitCounitAlternating}(unit, counit, depth), \mathsf{e})$.

The five witnesses assemble the displayed conjunction, giving the total endomorphism collapse including the alternating ledger. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier endomorphism total alternating collapse`

Definition 107.28 (Adjunction composition pasting data)**.** *Fix histories*

$$p_1, q_1, a_1, u_1, c_1, l_1, r_1, p_2, q_2, a_2, u_2, c_2, l_2, r_2, p_{21}, q_{21}, s_u, s_c, a, \tilde{u}_1, \tilde{u}_2, \tilde{c}_1, \tilde{c}_2, u_{21}, c_{21}, l_{21}, r_{21} : \mathsf{Hist}.$$

An adjunction composition pasting datum is the following finite family of generated judgments:

1. *the two input unit-counit carriers*

$$\mathrm{AdjunctionUnitCounitCarrier}(p_1, q_1, a_1, u_1, c_1, l_1, r_1)$$

and

$$\mathrm{AdjunctionUnitCounitCarrier}(p_2, q_2, a_2, u_2, c_2, l_2, r_2);$$

2. *the two whiskered unit-component rows*

$$\mathrm{NatTransPrefixComponentCarrier}(p_{21}, s_u, a, \tilde{u}_1), \qquad \mathrm{NatTransPrefixComponentCarrier}(s_u, q_{21}, a, \tilde{u}_2);$$

3. *the two whiskered counit-component rows*

$$\mathrm{NatTransPrefixComponentCarrier}(q_{21}, s_c, a, \tilde{c}_1), \qquad \mathrm{NatTransPrefixComponentCarrier}(s_c, p_{21}, a, \tilde{c}_2);$$

4. the two vertical-composition ledgers

$$\mathsf{Cont}(\tilde{u}_1, \tilde{u}_2, u_{21}), \qquad \mathsf{Cont}(\tilde{c}_1, \tilde{c}_2, c_{21});$$

5. the two composite triangle ledgers

$$\mathsf{Cont}(u_{21}, c_{21}, l_{21}), \qquad \mathsf{Cont}(c_{21}, u_{21}, r_{21}).$$

The categorical reading is that p_{21} displays the composite left functor $F_2 \circ F_1$, q_{21} displays the composite right functor $G_1 \circ G_2$, u_{21} displays the vertically composed unit $(G_1\eta_2F_1) \circ \eta_1$, and c_{21} displays the vertically composed counit $\varepsilon_2 \circ (F_2\varepsilon_1G_2)$.

Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionCompositionPastingData`

Theorem 107.29 (Composition of adjunction unit-counit carriers)**.** *Every adjunction composition pasting datum of Definition 107.28 determines a composite unit-counit carrier*

$$\mathrm{AdjunctionUnitCounitCarrier}(p_{21}, q_{21}, a, u_{21}, c_{21}, l_{21}, r_{21}).$$

Thus the BEDC display of $F_2 \circ F_1$ and $G_1 \circ G_2$ carries an $\mathsf{Adjunction}^{\uparrow}$ unit-counit certificate with unit and counit obtained by whiskering and vertical composition.

Proof. Project the two unit-component rows from the pasting datum:

$$\mathrm{NatTransPrefixComponentCarrier}(p_{21}, s_u, a, \tilde{u}_1)$$

and

$$\mathrm{NatTransPrefixComponentCarrier}(s_u, q_{21}, a, \tilde{u}_2).$$

Project also the displayed vertical-composition ledger

$$\mathsf{Cont}(\tilde{u}_1, \tilde{u}_2, u_{21}).$$

The vertical composition closure theorem for $\mathsf{NatTrans}^{\uparrow}$ prefix components applies to these two consecutive components with common middle prefix s_u and common object history a. It returns

$$\mathrm{NatTransPrefixComponentCarrier}(p_{21}, q_{21}, a, u_{21}).$$

Project the two counit-component rows from the pasting datum:

$$\mathrm{NatTransPrefixComponentCarrier}(q_{21}, s_c, a, \tilde{c}_1)$$

and

$$\mathrm{NatTransPrefixComponentCarrier}(s_c, p_{21}, a, \tilde{c}_2).$$

Project also the displayed vertical-composition ledger

$$\mathsf{Cont}(\tilde{c}_1, \tilde{c}_2, c_{21}).$$

The same $\mathsf{NatTrans}^{\uparrow}$ vertical composition closure theorem, now applied with common middle prefix s_c and common object history a, returns

$$\mathrm{NatTransPrefixComponentCarrier}(q_{21}, p_{21}, a, c_{21}).$$

The pasting datum supplies the two composite triangle ledgers

$$\mathsf{Cont}(u_{21}, c_{21}, l_{21})$$

and

$$\mathsf{Cont}(c_{21}, u_{21}, r_{21}).$$

By Definition 107.1, an adjunction unit-counit carrier consists precisely of a prefix natural-transformation component from the left prefix to the right prefix, a prefix natural-transformation component from the right prefix to the left prefix, and the two displayed continuation triangle ledgers. The four judgments just constructed and projected instantiate those four fields at $(p_{21}, q_{21}, a, u_{21}, c_{21}, l_{21}, r_{21})$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionCompositionPasting unit counit carrier`
Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier composition pasting closed`

Proposition 107.30 (Composite unit and counit empty readback)**.** *For every adjunction composition pasting datum of Definition 107.28, the following two readback implications hold:*

$$\mathsf{hsame}(u_{21}, e) \Longrightarrow \mathsf{hsame}(\tilde{u}_1, e) \wedge \mathsf{hsame}(\tilde{u}_2, e) \wedge \mathsf{hsame}(p_{21}, s_u) \wedge \mathsf{hsame}(s_u, q_{21}),$$

and

$$\mathsf{hsame}(c_{21}, e) \Longrightarrow \mathsf{hsame}(\tilde{c}_1, e) \wedge \mathsf{hsame}(\tilde{c}_2, e) \wedge \mathsf{hsame}(q_{21}, s_c) \wedge \mathsf{hsame}(s_c, p_{21}).$$

Proof. Project the unit-component rows

$$\mathrm{NatTransPrefixComponentCarrier}(p_{21}, s_u, a, \tilde{u}_1)$$

and

$$\mathrm{NatTransPrefixComponentCarrier}(s_u, q_{21}, a, \tilde{u}_2),$$

together with the unit vertical-composition ledger

$$\mathsf{Cont}(\tilde{u}_1, \tilde{u}_2, u_{21}).$$

The vertical-composite empty-result exactness theorem for $\mathsf{NatTrans}^{\uparrow}$ prefix components states that, for these consecutive component rows, the assertion $\mathsf{hsame}(u_{21}, e)$ is equivalent to the conjunction

$$\mathsf{hsame}(\tilde{u}_1, e) \wedge \mathsf{hsame}(\tilde{u}_2, e) \wedge \mathsf{hsame}(p_{21}, s_u) \wedge \mathsf{hsame}(s_u, q_{21}).$$

Taking the forward implication of that equivalence gives the first displayed readback. Project the counit-component rows

$$\mathrm{NatTransPrefixComponentCarrier}(q_{21}, s_c, a, \tilde{c}_1)$$

and

$$\mathrm{NatTransPrefixComponentCarrier}(s_c, p_{21}, a, \tilde{c}_2),$$

together with the counit vertical-composition ledger

$$\mathsf{Cont}(\tilde{c}_1, \tilde{c}_2, c_{21}).$$

Applying the same vertical-composite empty-result exactness theorem to these consecutive component rows states that $\mathsf{hsame}(c_{21}, e)$ is equivalent to

$$\mathsf{hsame}(\tilde{c}_1, e) \wedge \mathsf{hsame}(\tilde{c}_2, e) \wedge \mathsf{hsame}(q_{21}, s_c) \wedge \mathsf{hsame}(s_c, p_{21}).$$

The forward implication of this equivalence is the second displayed readback. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionCompositionPasting empty readback`

Corollary 107.31 (Composite triangle result determinacy)**.** *For every adjunction composition pasting datum of Definition 107.28, if histories l'_{21} and r'_{21} satisfy*

$$\mathsf{Cont}(u_{21}, c_{21}, l'_{21})$$

and

$$\mathsf{Cont}(c_{21}, u_{21}, r'_{21}),$$

then

$$\mathsf{hsame}(l_{21}, l'_{21}) \wedge \mathsf{hsame}(r_{21}, r'_{21}).$$

Proof. The pasting datum supplies

$$\mathsf{Cont}(u_{21}, c_{21}, l_{21})$$

and the hypothesis supplies

$$\mathsf{Cont}(u_{21}, c_{21}, l'_{21}).$$

Continuation determinacy applied to these two continuation witnesses with the same source history u_{21} and the same right history c_{21} yields

$$\mathsf{hsame}(l_{21}, l'_{21}).$$

The pasting datum also supplies

$$\mathsf{Cont}(c_{21}, u_{21}, r_{21})$$

and the hypothesis supplies

$$\mathsf{Cont}(c_{21}, u_{21}, r'_{21}).$$

Continuation determinacy applied to these two continuation witnesses with the same source history c_{21} and the same right history u_{21} yields

$$\mathsf{hsame}(r_{21}, r'_{21}).$$

The two history-sameness witnesses form the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionCompositionPasting triangle determinacy`
Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionCompositionPasting triangle results deterministic`

Definition 107.32 (Adjunction triangle carrier)**.** *For histories $left$, $right$, $object$, $unit$, $counit$, $leftLeg$, and $rightLeg$, the adjunction triangle carrier consists of a prefix natural transformation component from $left$ to $right$, a prefix natural transformation component from $right$ to $left$, and the two continuation ledgers*

$$\mathsf{Cont}(unit, counit, leftLeg), \qquad \mathsf{Cont}(counit, unit, rightLeg).$$

Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier`

Theorem 107.33 (Adjunction triangle carrier zero-headed component impossibility)**.** *For histories $left, right, object, unit, counit, leftLeg, rightLeg : \mathsf{BHist}$, an adjunction triangle carrier has no zero-headed displayed component among its prefixes, object, unit, counit, or triangle legs.*

Proof. The prefix, object, unit, and counit cases are rejected by the natural-transformation prefix-component zero-headed impossibility theorem applied to one of the two component rows. For the two triangle legs, the unit and counit component rows supply unary histories for the component histories; unary continuation closure makes the leg result unary, contradicting unary zero-headed exclusion. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier zero headed component absurd`

Theorem 107.34 (Adjunction empty roundtrip prefix determinacy)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, if an adjunction triangle carrier has both roundtrip legs* hsame *to* e*, then its two prefix histories are classifier-equal:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow \mathsf{hsame}(leftLeg, \mathsf{e}) \Longrightarrow \mathsf{hsame}(rig$$

Proof. The empty left roundtrip continuation collapses both *unit* and *counit* to e, and the empty right roundtrip gives the same collapse in the opposite order. The prefix component identity exactness theorem applied to the empty *unit* component yields $\mathsf{hsame}(left, right)$; applying it to the empty *counit* component gives the reverse classifier equality, so the two displayed roundtrip collapses agree on the same prefix pair. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier empty roundtrip prefix deterministic`

Theorem 107.35 (Adjunction roundtrip emptiness component exactness)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, an adjunction triangle carrier has both roundtrip legs* hsame *to* e *exactly when its displayed unit and counit histories are both* hsame *to* e*:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow (\mathsf{hsame}(leftLeg, \mathsf{e}) \wedge \mathsf{hsame}(rightL$$

Proof. For the forward direction, an empty left roundtrip continuation inverts the continuation ledger and forces both *unit* and *counit* to be e. For the reverse direction, transport the two triangle-leg continuations along the empty component witnesses and compare them with the empty-empty continuation unit law. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier roundtrip empty iff components empty`

Theorem 107.36 (Adjunction triangle single-boundary total collapse)**.** *For histories* $left, right, object, unit, coun$ BHist*, a single empty boundary component in an adjunction triangle carrier forces the whole displayed triangle to collapse to the empty endpoint package:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow (\mathsf{hsame}(unit, \mathsf{e}) \vee \mathsf{hsame}(counit, \mathsf{e})$$

Proof. Case on the displayed boundary disjunction. In the *unit*-empty case, the natural-transformation empty-identity exactness theorem applied to the unit component reads the two endpoint prefixes as $\mathsf{hsame}(left, right)$. Transport the counit component row across this endpoint equality; after transport, the counit row is an endomorphism component, and endomorphism-component empty exactness gives $\mathsf{hsame}(counit, \mathsf{e})$.

Now the two component witnesses satisfy the right side of Theorem 107.35. The reverse direction of that theorem gives empty *leftLeg* and *rightLeg* witnesses. The prefix equality is also recovered from Theorem 107.38, so the five displayed conclusions pack together.

In the *counit*-empty case, exchange the two component rows and the two continuation ledgers. The same endpoint readback, transport to an endomorphism component, roundtrip-empty exactness, and component-empty prefix determinacy produce the corresponding *unit* witness, both empty triangle legs, and $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier single boundary total collapse`

Theorem 107.37 (Adjunction triangle single-leg empty equivalence)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, an adjunction triangle carrier makes either empty triangle leg determine the other:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow (\mathsf{hsame}(leftLeg, \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(rig$$

Proof. Fix the carrier. For the forward direction, invert the left continuation ledger $\mathsf{Cont}(unit, counit, leftLeg)$ after transporting the displayed $leftLeg$ witness to e. This gives $\mathsf{hsame}(unit, \mathsf{e})$ and $\mathsf{hsame}(counit, \mathsf{e})$. Apply the reverse direction of Theorem 107.35 and project the resulting $rightLeg$ witness.

For the reverse direction, use the right continuation ledger $\mathsf{Cont}(counit, unit, rightLeg)$ first. The same inversion gives empty *unit* and *counit* witnesses, the reverse direction of Theorem 107.35 rebuilds the pair of empty triangle legs, and projecting gives the displayed $leftLeg$ witness. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier single leg empty iff`

Theorem 107.38 (Adjunction triangle empty components prefix determinacy)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, an adjunction triangle carrier whose displayed unit and counit components are both* hsame *to* e *has equal endpoint prefixes:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow \mathsf{hsame}(unit, \mathsf{e}) \Longrightarrow \mathsf{hsame}(counit,$$

Proof. Transport the two triangle-leg continuations along the empty unit and counit component witnesses. Both roundtrip legs become hsame to the empty continuation result, and the roundtrip prefix determinacy theorem identifies the two endpoint prefixes. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier components empty prefix deterministic`

Theorem 107.39 (Adjunction roundtrip empty component-prefix readback)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, an adjunction triangle carrier whose two roundtrip legs are* hsame *to* e *has empty displayed components and equal endpoint prefixes:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow \mathsf{hsame}(leftLeg, \mathsf{e}) \Longrightarrow \mathsf{hsame}(righ$$

Proof. The roundtrip emptiness component exactness theorem reads the two empty legs back to empty *unit* and *counit* components. The empty roundtrip prefix determinacy theorem applied to the same carrier and leg witnesses identifies the two endpoint prefixes. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier roundtrip empty component prefix readback`

Theorem 107.40 (Adjunction triangle roundtrip total collapse)**.** *For histories left, right, object, unit, counit, leftLeg, and rightLeg, every adjunction triangle carrier has empty displayed components, empty roundtrip legs, and equal endpoint prefixes:*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge$$

Proof. Apply the prefix unit-counit composite empty theorem to the *unit* followed by *counit* and again in the reverse order. These two applications make both roundtrip legs hsame to e. The roundtrip empty component-prefix readback theorem then recovers empty *unit* and *counit* components and the endpoint prefix equality. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier roundtrip total collapse`

Corollary 107.41 (Adjunction triangle empty-roundtrip consumer determinacy)**.** *For histories $left$, $right$, $object$, $unit$, $counit$, $leftLeg$, and $rightLeg$, fix the displayed adjunction triangle row*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg)$$

and the two empty roundtrip rows $\mathsf{hsame}(leftLeg, \mathsf{e})$ *and* $\mathsf{hsame}(rightLeg, \mathsf{e})$*. In this corollary, an accepted empty-roundtrip consumer use means a use whose admission data are exactly these three local rows. Then every accepted empty-roundtrip consumer use has the same determined readback*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \wedge \mathsf{hsame}(leftLeg, \mathsf{e}) \wedge \mathsf{hsame}(rightLeg$$
$$\Longrightarrow \mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left, right).$$

Consequently the finite consumer route is the carrier row, the two empty roundtrip rows, the component-exactness readback, and the prefix-determinacy readback. The exactness decision is the pair $\mathsf{hsame}(unit, \mathsf{e})$ *and* $\mathsf{hsame}(counit, \mathsf{e})$*; the prefix decision is* $\mathsf{hsame}(left, right)$*; and the empty display normal form is* $(left, left, object, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$*. No accepted use with the same fixed rows may choose a second component source, an alternate prefix coordinate, or a duplicate classifier source outside those displayed rows.*

Proof. Let an accepted empty-roundtrip consumer use be given. By its admission data it has the adjunction triangle carrier row and the two witnesses $\mathsf{hsame}(leftLeg, \mathsf{e})$ and $\mathsf{hsame}(rightLeg, \mathsf{e})$.

Apply Definition 107.35 in the forward direction. This reads the two empty roundtrip rows back to the unique displayed component-exactness decision $\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e})$.

Apply Definition 107.34 to the same carrier row and the same two empty roundtrip rows. This reads the endpoint prefixes through the unique local prefix decision $\mathsf{hsame}(left, right)$. Together with the original empty witnesses for $leftLeg$ and $rightLeg$, these three conclusions classify the display by the tuple $(left, left, object, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$.

Every conclusion used by the consumer has therefore been obtained from the same three admission rows through the two cited local readback theorems. A purported alternate component source would have to bypass the component exactness readback; an alternate prefix coordinate would have to bypass prefix determinacy; and a duplicate classifier source would have to add a coordinate not present in the carrier row or in the two empty roundtrip rows. Such a row is not part of the accepted empty-roundtrip consumer use, so the displayed route, normal form, and exactness decision are determined. □

107.2 The certificate

Theorem 107.42 (Adjunction prefix unit-counit empty composite)**.** *For histories $p, q, a, \eta, \varepsilon, composite$:* BHist*, if η is a prefix natural-transformation component from p to q at a, ε is a prefix component from q back to p at a, and* $\mathsf{Cont}(\eta, \varepsilon, composite)$*, then the composite is an endomorphism component over p and is* hsame *to* e*.*

Proof. Vertical composition closure turns the unit and counit prefix components into a prefix component from p to p at a. The endomorphism-component readback for prefix natural transformations identifies the displayed component history with the empty history. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionPrefix unit counit composite empty`

Theorem 107.43 (Adjunction prefix endomorphism triangle component closure)**.** *For histories $p, a, \eta, \varepsilon, left, right$, suppose*

$$\mathsf{NatTransPrefixComponentCarrier}(p, p, a, \eta), \qquad \mathsf{NatTransPrefixComponentCarrier}(p, p, a, \varepsilon),$$

and suppose the two triangle composites are witnessed by

$$\mathsf{Cont}(\eta, \varepsilon, left), \qquad \mathsf{Cont}(\varepsilon, \eta, right).$$

Then both composites are again endomorphism components at the same prefix and object, and the two displayed composite histories classify together:

$$\mathsf{NatTransPrefixComponentCarrier}(p, p, a, left) \wedge \mathsf{NatTransPrefixComponentCarrier}(p, p, a, right) \wedge \mathsf{hsame}(left, right).$$

Proof. Endomorphism component exactness collapses both η and ε to the empty component while retaining the prefix and object carriers. The two continuation ledgers therefore collapse $left$ and $right$ to the empty history. Unary continuation closure supplies the component carriers for $left$ and $right$, and transitivity through the empty history gives $\mathsf{hsame}(left, right)$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionPrefix endomorphism triangle component closed`

Theorem 107.44 (Adjunction prefix endomorphism triangle identity exactness)**.** *For histories $p, a, f, \eta, \epsilon, left, right : \mathsf{BHist}$, assume η and ϵ are prefix natural-transformation components from p to p at a, and assume f is a category hom-carrier from $\mathsf{append}(p, a)$ to itself. If*

$$\mathsf{Cont}(\eta, f, left)$$

and

$$\mathsf{Cont}(f, \epsilon, right)$$

hold, then both triangle composites are exactly the displayed endomorphism:

$$\mathsf{hsame}(left, f) \wedge \mathsf{hsame}(right, f).$$

Proof. The prefix-component endomorphism exactness theorem identifies both η and ϵ with the empty component. Transport the two displayed continuations across these endpoint equalities. The left composite is then a left-unit continuation, and the right composite is a right-unit continuation, so the two unit laws identify the results with f. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionPrefixEndomorphismTriangle identity exactness`

Theorem 107.45 (Adjunction prefix component empty cycle)**.** *For prefix components over a common object a, let η run from p to q and let θ run from q back to p. If*

$$\mathsf{Cont}(\eta, \theta, composite) \quad \textit{and} \quad \mathsf{hsame}(composite, \mathsf{e}),$$

then both prefixes agree in both directions and both component histories are empty:

$$\mathsf{hsame}(p, q) \wedge \mathsf{hsame}(q, p) \wedge \mathsf{hsame}(\eta, \mathsf{e}) \wedge \mathsf{hsame}(\theta, \mathsf{e}).$$

Proof. Transport the displayed component cycle to an empty continuation result. The empty vertical-composite exactness theorem for NatTrans prefix components then identifies both component histories with e and extracts the two adjacent prefix equalities. Reordering those four facts gives the stated adjunction cycle boundary. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionPrefixComponent cycle empty prefix same`

Theorem 107.46 (Adjunction triangle carrier carrier-swap symmetry)**.** *For histories $left$, $right$, $object$, $unit$, $counit$, $leftLeg$, and $rightLeg$, the adjunction triangle carrier is invariant under exchanging the two carrier sides together with the unit-counit data and the two triangle ledgers:*

$$\begin{aligned}&\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg)\\ \iff\quad &\mathsf{AdjunctionTriangleCarrier}(right, left, object, counit, unit, rightLeg, leftLeg).\end{aligned}$$

Proof. By Definition 107.32, the left-hand carrier consists of four fields: the prefix component from $left$ to $right$ carried by $unit$, the prefix component from $right$ to $left$ carried by $counit$, the continuation ledger $\mathsf{Cont}(unit, counit, leftLeg)$, and the continuation ledger $\mathsf{Cont}(counit, unit, rightLeg)$. The right-hand carrier asks for the same four fields in the exchanged order: the $counit$ component, the $unit$ component, the second ledger, and then the first ledger. Each implication is therefore obtained by reordering these four conjunctive fields. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier carrier swap iff`
↪ variant `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier fields swap iff`

Theorem 107.47 (Adjunction unit-counit carrier carrier-swap symmetry)**.** *For histories $p, q, a, unit, counit, left, r$ BHist, the adjunction unit-counit carrier is invariant under exchanging the two prefix carriers, exchanging unit with counit, and exchanging the two triangle ledgers:*

$$\begin{aligned}&\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)\\ \iff\quad &\mathsf{AdjunctionUnitCounitCarrier}(q, p, a, counit, unit, right, left).\end{aligned}$$

Proof. By Definition 107.1, a witness of the left-hand carrier consists of four fields: the prefix component from p to q carried by $unit$, the prefix component from q to p carried by $counit$, the continuation ledger $\mathsf{Cont}(unit, counit, left)$, and the continuation ledger $\mathsf{Cont}(counit, unit, right)$. The displayed carrier with p and q exchanged asks for exactly the same four fields in the order: the $counit$ component, the $unit$ component, the second ledger, and the first ledger. Reordering the four conjunctive fields proves the forward implication, and the same reordering with the two sides exchanged proves the reverse implication. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier carrier swap iff`

Definition 107.48 (Adjunction unit-counit carrier-swap classifier)**.** *For unit-counit displays $D = (p, q, a, u, c, l, r)$ and $D' = (p', q', a', u', c', l', r')$, the carrier-swap relation $\mathsf{AdjunctionUnitCounitCarrierSwap}(D, D')$ holds when*

$$p' = q \wedge q' = p \wedge a' = a \wedge u' = c \wedge c' = u \wedge l' = r \wedge r' = l.$$

The display classifier $\mathsf{AdjunctionUnitCounitDisplaySame}(D, D')$ holds when

$$p\mathsf{hsame}p' \wedge q\mathsf{hsame}q' \wedge a\mathsf{hsame}a' \wedge u\mathsf{hsame}u' \wedge c\mathsf{hsame}c' \wedge l\mathsf{hsame}l' \wedge r\mathsf{hsame}r'.$$

Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrierSwap`
Lean4 (definition) `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitDisplaySame`

Theorem 107.49 (Adjunction triangle display empty normal form)**.** *For histories $left$, $right$, $object$, $unit$, $counit$, $leftLeg$, and $rightLeg$, let $D = (left, right, object, unit, counit, leftLeg, rightLeg)$ and $N = (left, left, object, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$. Then*

$$\mathsf{AdjunctionTriangleCarrier}(left, right, object, unit, counit, leftLeg, rightLeg) \Longrightarrow \mathsf{AdjunctionUnitCounitDisplaySame}$$

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUp StdBridge`

Proof. Apply Definition 107.40 to the given adjunction triangle carrier. This gives the four emptiness witnesses for *unit*, *counit*, $leftLeg$, and $rightLeg$, together with the endpoint prefix witness $\mathsf{hsame}(left, right)$.

Unfold Definition 107.48 only at the display classifier fields. The $left$ and *object* coordinates are witnessed by reflexivity. The $right$ coordinate is witnessed by symmetry of the endpoint-prefix witness, giving $\mathsf{hsame}(right, left)$.

The remaining four classifier fields are exactly the empty witnesses for *unit*, *counit*, $leftLeg$, and $rightLeg$. Packing these seven witnesses in the displayed coordinate order proves AdjunctionUnitCounitDisplaySa

□

Theorem 107.50 (Adjunction unit-counit carrier-swap involution)**.** *Let $D_i = (p_i, q_i, a_i, u_i, c_i, l_i, r_i)$ for $i = 0, 1, 2$ be unit-counit displays of histories. If*

$$\mathsf{AdjunctionUnitCounitCarrier}(p_0, q_0, a_0, u_0, c_0, l_0, r_0)$$
$$\wedge\ \mathsf{AdjunctionUnitCounitCarrierSwap}(D_0, D_1) \wedge \mathsf{AdjunctionUnitCounitCarrierSwap}(D_1, D_2),$$

then

$$\mathsf{AdjunctionUnitCounitCarrier}(p_2, q_2, a_2, u_2, c_2, l_2, r_2)$$
$$\wedge\ \mathsf{AdjunctionUnitCounitDisplaySame}(D_0, D_2).$$

Proof. Unfold the two carrier-swap relations of Definition 107.48. The first gives $p_1 = q_0$, $q_1 = p_0$, $a_1 = a_0$, $u_1 = c_0$, $c_1 = u_0$, $l_1 = r_0$, and $r_1 = l_0$. The second gives $p_2 = q_1$, $q_2 = p_1$, $a_2 = a_1$, $u_2 = c_1$, $c_2 = u_1$, $l_2 = r_1$, and $r_2 = l_1$. Substituting the first row into the second yields

$$p_2 = p_0 \wedge q_2 = q_0 \wedge a_2 = a_0 \wedge u_2 = u_0 \wedge c_2 = c_0 \wedge l_2 = l_0 \wedge r_2 = r_0.$$

Substitute these coordinate identities in the carrier of D_0; by Definition 107.1 this is the carrier of D_2. The same coordinate identities turn the seven reflexive hsame witnesses on D_0 into the componentwise display classifier between D_0 and D_2. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionUnitCounitCarrier swap involution`

Theorem 107.51 (Adjunction triangle carrier-swap involution)**.** *Let $D_i = (p_i, q_i, a_i, u_i, c_i, l_i, r_i)$ for $i = 0, 1, 2$ be adjunction triangle displays. If*

$$\mathsf{AdjunctionTriangleCarrier}(p_0, q_0, a_0, u_0, c_0, l_0, r_0)$$
$$\wedge\ \mathsf{AdjunctionUnitCounitCarrierSwap}(D_0, D_1) \wedge \mathsf{AdjunctionUnitCounitCarrierSwap}(D_1, D_2),$$

then the twice-swapped display is again carried and is componentwise classifier-same to the original display:

$$\mathsf{AdjunctionTriangleCarrier}(p_2, q_2, a_2, u_2, c_2, l_2, r_2)$$
$$\wedge\ \mathsf{AdjunctionUnitCounitDisplaySame}(D_0, D_2).$$

Proof. Unfold the first carrier-swap relation in Definition 107.48. It gives

$$p_1 = q_0 \wedge q_1 = p_0 \wedge a_1 = a_0 \wedge u_1 = c_0 \wedge c_1 = u_0 \wedge l_1 = r_0 \wedge r_1 = l_0.$$

Unfold the second carrier-swap relation in the same definition. It gives

$$p_2 = q_1 \wedge q_2 = p_1 \wedge a_2 = a_1 \wedge u_2 = c_1 \wedge c_2 = u_1 \wedge l_2 = r_1 \wedge r_2 = l_1.$$

Substituting the first row into the second row yields

$$p_2 = p_0 \wedge q_2 = q_0 \wedge a_2 = a_0 \wedge u_2 = u_0 \wedge c_2 = c_0 \wedge l_2 = l_0 \wedge r_2 = r_0.$$

Substitute these coordinate identities in the carrier of D_0. By Definition 107.32, the same two prefix-component fields and the same two continuation ledgers are exactly the carrier fields for D_2. The same coordinate identities also turn the seven reflexive hsame witnesses on the coordinates of D_0 into $\mathsf{AdjunctionUnitCounitDisplaySame}(D_0, D_2)$. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier swap involution`

Theorem 107.52 (Right adjoint uniqueness up to natural isomorphism)**.** *Let* $p, q_1, q_2, a, u_1, c_1, l_1, r_1, u_2, c_2, l_2, r_2, \eta$, BHist. *Suppose two adjunction unit-counit displays share the left prefix* p *and the object history* a*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q_1, a, u_1, c_1, l_1, r_1) \quad \textit{and} \quad \mathsf{AdjunctionUnitCounitCarrier}(p, q_2, a, u_2, c_2, l_2, r_2).$$

Suppose the comparison continuations are displayed by

$$\mathsf{Cont}(c_1, u_2, \eta), \qquad \mathsf{Cont}(c_2, u_1, \eta'), \qquad \mathsf{Cont}(\eta, \eta', I_1), \qquad \mathsf{Cont}(\eta', \eta, I_2).$$

Reading p *as the common left-adjoint prefix and* q_i *as the two right-adjoint prefixes, the comparison components* $\eta : q_1 \Rightarrow q_2$ *and* $\eta' : q_2 \Rightarrow q_1$ *form a componentwise* $\mathsf{NatTrans}^{\uparrow}$ *isomorphism:*

$$\begin{aligned}&\mathsf{NatTransPrefixComponentCarrier}(q_1, q_2, a, \eta) \wedge \mathsf{NatTransPrefixComponentCarrier}(q_2, q_1, a, \eta')\\ &\wedge \mathsf{NatTransPrefixComponentClassifier}_{q_1,q_1,a}(I_1, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentClassifier}_{q_2,q_2,a}(I_2, \mathsf{e}).\end{aligned}$$

Proof. Unfold the two adjunction carriers by Definition 107.1. The first carrier supplies prefix components $u_1 : p \Rightarrow q_1$ and $c_1 : q_1 \Rightarrow p$ at a; the second supplies $u_2 : p \Rightarrow q_2$ and $c_2 : q_2 \Rightarrow p$ at the same a.

Apply Definition 63.13 to c_1, u_2, and $\mathsf{Cont}(c_1, u_2, \eta)$. This gives

$$\mathsf{NatTransPrefixComponentCarrier}(q_1, q_2, a, \eta).$$

Applying the same closure theorem to c_2, u_1, and $\mathsf{Cont}(c_2, u_1, \eta')$ gives

$$\mathsf{NatTransPrefixComponentCarrier}(q_2, q_1, a, \eta').$$

A further application of vertical closure to the two displayed comparison components gives component carriers for $I_1 : q_1 \Rightarrow q_1$ and $I_2 : q_2 \Rightarrow q_2$.

The components η and η' run in opposite directions over the same object history. Hence Definition 63.45 supplies empty components from q_1 to q_2 and from q_2 to q_1, together with $q_1 \mathsf{hsame} q_2$. Comparing the defining composite $\mathsf{Cont}(c_1, u_2, \eta)$ with the displayed empty component by Definition 63.15 gives $\eta \mathsf{hsamee}$. The same argument for $\mathsf{Cont}(c_2, u_1, \eta')$ gives $\eta' \mathsf{hsamee}$.

Apply Definition 63.20 to η, η', and I_1. The two component emptiness witnesses, $q_1 \mathsf{hsame} q_2$, and symmetry of hsame satisfy the right-hand side, so $I_1 \mathsf{hsamee}$. Exchanging η and η' gives $I_2 \mathsf{hsamee}$.

The component rows extracted from the two adjunction carriers include unary witnesses for q_1, q_2, and a. Thus Definition 63.3 supplies the empty identity components at q_1 and at q_2. Combining those empty identity components with the carried endomorphism components I_1 and I_2, and with $I_1 \mathsf{hsamee}$ and $I_2 \mathsf{hsamee}$, packages exactly the two classifier witnesses in Definition 63.62. Since Definition 63.1 reads a $\mathsf{NatTrans}^{\uparrow}$ carrier componentwise over object histories, these classifier witnesses are the identity vertical-composite rows for the displayed comparison pair. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionRightAdjoint natural isomorphism uniqueness`

Theorem 107.53 (Left adjoint uniqueness up to natural isomorphism)**.** *Let* $p_1, p_2, q, a, u_1, c_1, l_1, r_1, u_2, c_2, l_2, r_2, \xi, \xi$ BHist*. Suppose two adjunction unit-counit displays share the right prefix* q *and the object history* a*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p_1, q, a, u_1, c_1, l_1, r_1) \quad \textit{and} \quad \mathsf{AdjunctionUnitCounitCarrier}(p_2, q, a, u_2, c_2, l_2, r_2).$$

Suppose the comparison continuations are displayed by

$$\mathsf{Cont}(u_1, c_2, \xi), \qquad \mathsf{Cont}(u_2, c_1, \xi'), \qquad \mathsf{Cont}(\xi, \xi', J_1), \qquad \mathsf{Cont}(\xi', \xi, J_2).$$

Reading q *as the common right-adjoint prefix and* p_i *as the two left-adjoint prefixes, the comparison components* $\xi : p_1 \Rightarrow p_2$ *and* $\xi' : p_2 \Rightarrow p_1$ *form a componentwise* $\mathsf{NatTrans}^{\uparrow}$ *isomorphism:*

$$\begin{aligned}&\mathsf{NatTransPrefixComponentCarrier}(p_1, p_2, a, \xi) \wedge \mathsf{NatTransPrefixComponentCarrier}(p_2, p_1, a, \xi')\\ &\wedge \mathsf{NatTransPrefixComponentClassifier}_{p_1,p_1,a}(J_1, \mathsf{e}) \wedge \mathsf{NatTransPrefixComponentClassifier}_{p_2,p_2,a}(J_2, \mathsf{e}).\end{aligned}$$

Proof. Apply the carrier-swap symmetry of Definition 107.47 to the two displayed adjunction carriers. The resulting carriers are

$$\mathsf{AdjunctionUnitCounitCarrier}(q, p_1, a, c_1, u_1, r_1, l_1) \quad \text{and} \quad \mathsf{AdjunctionUnitCounitCarrier}(q, p_2, a, c_2, u_2, r_2, l_2).$$

They share the left prefix q and the same object history a. In these swapped displays the comparison continuations required by Definition 107.52 are exactly

$$\mathsf{Cont}(u_1, c_2, \xi), \qquad \mathsf{Cont}(u_2, c_1, \xi'), \qquad \mathsf{Cont}(\xi, \xi', J_1), \qquad \mathsf{Cont}(\xi', \xi, J_2),$$

because the counit fields of the swapped displays are u_1 and u_2, while the unit fields are c_1 and c_2.

Applying Definition 107.52 to the swapped pair gives the two prefix-component carriers from p_1 to p_2 and back, together with the two identity-composite classifier rows over p_1 and p_2. These four conclusions are exactly the displayed componentwise $\mathsf{NatTrans}^{\uparrow}$ isomorphism for the original pair with common right-adjoint prefix q. □

Lean4 ✓ `BEDC.Derived.AdjunctionUp.AdjunctionLeftAdjoint natural isomorphism uniqueness`

Closure status for $\mathsf{NameCert}_{\mathsf{Adjunction}^{\uparrow}}$**.**

Constructive story: An adjunction seed is assembled from two prefix natural-transformation components: a unit component from one prefix to another, a counit component back, and two continuation ledgers that display the triangle composites. The carried displays are precisely those histories for which the two components are prefix components over the same object and the two triangle legs are witnessed continuations; anything missing a component carrier or one of the two continuation rows is outside this seed, and sameness is componentwise hsame on the two prefixes, object, unit, counit, and triangle legs, with the carrier-swap row recording the same display in the opposite order. The source is the unit-counit triangle display, the pattern records empty composite collapse, endomorphism triangle exactness, cycle collapse to equal prefixes, carrier-swap symmetry, and the empty normal form, while the ledger keeps the unit, counit, and both triangle histories visible. At this seed level the name-certificate obligations are not exposed as a completed public certificate, but the rows already show the intended mechanism: the empty component display supplies the canonical inhabitant, hsame supplies reflexivity, symmetry, and transitivity, and the normal-form and swap theorems preserve the carrier under classified replacement. A display history h lies in the signature gap of an $\mathsf{Adjunction}^{\uparrow}$ token p when the source-domain accepts the unit-counit display, the signature relation pairs it with an answer display and the two triangle ledgers, and token introduction names that answer as p; unfolded, the unit, counit, and both triangle legs collapse to empty histories while the two prefixes classify together as encoded by p. No standard bridge is present, and no hom-set universal adjunction is asserted at this boundary. The chapter registers this prefix unit-counit triangle surface under the name $\mathsf{Adjunction}^{\uparrow}$; the adjunction data are continuation rows over existing histories, with no external adjunction object, host natural-isomorphism primitive, or additional categorical axiom introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{Adjunction}\uparrow}$, maturePackageClosed)

Scope closed: The public export surface is the triangle display empty normal form together with the unit-counit carrier rows, triangle ledgers, carrier-swap display rows, Cont, hsame, and NameCert rows listed in this certificate block.

Formal verification: FormalStatus(NameCert$_{\mathsf{Adjunction}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.AdjunctionUp.AdjunctionTriangleCarrier\_display\_empty\_normal\_form`

Bridge status: bridgeChecked

Not claimed: This block does not close full adjunction theory, universal hom-set adjunctions, or standard bridges.

Upgrade path: No further formal-axis upgrade remains for this chapter because the recorded bridge has reached the top formal grade.

Chapter 108

A Concrete Naming Certificate for $\mathsf{Monad}^{\uparrow}$

The monad interface $\mathsf{Monad}^{\uparrow}$ packages an endofunctor on a category with unit and multiplication natural transformations satisfying the monad laws. It depends on $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$ (chapter 63) and $\mathsf{NameCert}_{\mathsf{Adjunction}^{\uparrow}}$ (chapter 107), and supplies the substrate for algebraic theories and computational effect models.

108.1 Carrier definition

Theorem 108.1 (Monad adjunction endomorphism triangle results are empty)**.** *For histories* $p, a, unit, counit, left, right : \mathsf{BHist}$*, suppose* $\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$ *holds. Then both triangle results are empty:*

$$\mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}).$$

Proof. Both the unit and counit components are endomorphism prefix components over the same prefix p and object a, so natural-transformation endomorphism component exactness identifies each component with e. The two triangle continuations are then compared with the empty right-unit continuation, first in the unit-counit order and then in the counit-unit order. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism triangle results empty`

Theorem 108.2 (Monad adjunction left triangle empty result has empty suffix)**.** *For histories* $p, a, unit, counit, left, right, suffix, out : \mathsf{BHist}$*, suppose* $\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, righ$ *holds. If* $\mathsf{Cont}(left, suffix, out)$ *and* $\mathsf{hsame}(out, \mathsf{e})$*, then*

$$\mathsf{hsame}(suffix, \mathsf{e}).$$

Proof. The endomorphism triangle result theorem identifies the left triangle result with e. After the output is also identified with e, the displayed continuation has empty source and empty result, so continuation empty-result inversion identifies the suffix with e. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism left triangle result empty suffix empty`

Theorem 108.3 (Monad adjunction endomorphism triangle bridge is empty iff)**.** *For histories* $p, a, unit, counit, left, right, bridge : \mathsf{BHist}$*, suppose* $\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$ *holds. Then*

$$\mathsf{Cont}(left, bridge, right) \iff \mathsf{hsame}(bridge, \mathsf{e}).$$

Proof. For the forward direction, the right triangle result is empty, so the left-triangle empty-result suffix theorem applies to the displayed bridge continuation. Conversely, the carrier identifies both triangle results with e; after the bridge is also identified with e, the continuation is the empty right-unit continuation. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism triangle bridge empty iff`

Theorem 108.4 (Monad adjunction endomorphism unit-counit bridge is empty iff)**.** *For histories* $p, a, unit, counit, left, right, bridge : \mathsf{BHist}$*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

holds. Then

$$\mathsf{Cont}(unit, bridge, counit) \iff \mathsf{hsame}(bridge, \mathsf{e}).$$

Proof. The forward direction is the unit-counit middle-empty theorem. Conversely, the endomorphism carrier identifies both *unit* and *counit* with e; after *bridge* is also identified with e, the displayed continuation is the canonical right-unit continuation at e. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism unit counit bridge empty iff`

Theorem 108.5 (Monad adjunction right triangle empty result has empty suffix)**.** *For histories* $p, a, unit, counit, left, right, suffix, out : \mathsf{BHist}$*, suppose* $\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, righ$ *holds. If* $\mathsf{Cont}(right, suffix, out)$ *and* $\mathsf{hsame}(out, \mathsf{e})$*, then*

$$\mathsf{hsame}(suffix, \mathsf{e}).$$

Proof. The endomorphism triangle result theorem identifies the right triangle result with e. With the displayed output also identified with e, the continuation has empty source and empty result. Continuation empty-result inversion therefore identifies the suffix with e. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism right triangle result empty suffix empty`

Theorem 108.6 (Monad adjunction endomorphism left triangle suffix-output empty iff)**.** *For histories* $p, a, unit, counit, left, right, suffix, out : \mathsf{BHist}$*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

and $\mathsf{Cont}(left, suffix, out)$ *hold. Then*

$$\mathsf{hsame}(out, \mathsf{e}) \iff \mathsf{hsame}(suffix, \mathsf{e}).$$

Proof. The forward direction is the left-triangle empty-result suffix theorem. For the reverse direction, the endomorphism triangle theorem identifies *left* with e; comparing the displayed continuation against the canonical right-unit continuation through the empty suffix gives an empty output. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism left triangle suffix output empty iff`

Theorem 108.7 (Monad adjunction endomorphism parallel triangle suffix agreement)**.** *For histories* $p, a, unit, counit, left, right, suffix, left', right'$: BHist*, suppose* AdjunctionUnitCounitCarrier$(p, p, a, unit, cou$ *holds. If* Cont$(left, suffix, left')$ *and* Cont$(right, suffix, right')$*, then*

$$\mathsf{hsame}(left', right').$$

Proof. The endomorphism triangle result theorem identifies both triangle outputs $left$ and $right$ with e. Hence $left$hsame$right$. Continuation respect for hsame transports this source equality across the common suffix continuation, with the suffix compared reflexively, and yields $left'$hsame$right'$. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism parallel triangle suffix same`

Theorem 108.8 (Monad adjunction endomorphism unit-counit middle is empty)**.** *For histories* $p, a, unit, counit, left, right, bridge$: BHist*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

and Cont$(unit, bridge, counit)$ *hold. Then bridge is empty:*

$$\mathsf{hsame}(bridge, \mathsf{e}).$$

Proof. The adjunction endomorphism carrier identifies both $unit$ and $counit$ with e. With those endpoints collapsed, the displayed continuation is a continuation from e through $bridge$ back to e, and empty-result inversion gives $bridge$hsamee. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism middle between unit counit empty`

Theorem 108.9 (Monad adjunction endomorphism counit-unit middle is empty)**.** *For histories* $p, a, unit, counit, left, right, bridge$: BHist*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right)$$

and Cont$(counit, bridge, unit)$ *hold. Then bridge is empty:*

$$\mathsf{hsame}(bridge, \mathsf{e}).$$

Proof. The unit and counit are both endomorphism prefix components in the carrier, so each is identified with e. After those endpoint identifications, the displayed counit-unit continuation has empty endpoints, and empty-result inversion identifies the middle history with e. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism middle between counit unit empty`

Theorem 108.10 (Monad adjunction endomorphism triangle composite is empty)**.** *For histories* $p, a, unit, counit, left, right, composite$: BHist*, suppose* AdjunctionUnitCounitCarrier$(p, p, a, unit, counit, left, right$ *holds and* Cont$(left, right, composite)$ *holds. Then* hsame$(composite, \mathsf{e})$*.*

Proof. The endomorphism triangle theorem identifies both triangle legs with e. Transport the displayed continuation against those two endpoint identifications and compare it with the empty right-unit continuation. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism triangle composite empty`

Theorem 108.11 (Monad adjunction endomorphism unit, counit, and triangles are empty)**.** *For histories* $p, a, unit, counit, left, right$: BHist*, suppose* AdjunctionUnitCounitCarrier$(p, p, a, unit, counit, left, right)$ *holds. Then the unit component, counit component, and both triangle results are empty:*

$$\mathsf{hsame}(unit, \mathsf{e}) \wedge \mathsf{hsame}(counit, \mathsf{e}) \wedge \mathsf{hsame}(left, \mathsf{e}) \wedge \mathsf{hsame}(right, \mathsf{e}).$$

Proof. The unit and counit are endomorphism prefix components over the same prefix, so prefix component exactness identifies both with e. Applying the triangle-result collapse for the same carrier gives the two empty triangle results, and the four witnesses are collected in order. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism unit counit triangle empty`

108.2 The certificate

Theorem 108.12 (Monad adjunction endomorphism semantic name certificate)**.** *Let* p, a : BHist *be unary histories. For histories* $unit, counit, left, right$: BHist*, write* $M_{p,a}(unit, counit, left, right)$ *for the carrier predicate*

$$M_{p,a}(unit, counit, left, right) :\Longleftrightarrow \mathsf{AdjunctionUnitCounitCarrier}(p, p, a, unit, counit, left, right).$$

For two displays $D = (unit, counit, left, right)$ *and* $D' = (unit', counit', left', right')$*, let* MonadAdjunctionEndomor *mean*

$$unit\mathsf{hsame}unit' \wedge counit\mathsf{hsame}counit' \wedge left\mathsf{hsame}left' \wedge right\mathsf{hsame}right'.$$

Then, in the display presentation of Definition 9.6, $M_{p,a}$ *carries a semantic naming certificate with source, pattern, and ledger all equal to* $M_{p,a}$ *and classifier* $\mathsf{MonadAdjunctionEndomorphismDisplaySame}_{p,a}$*:*

$$\mathsf{SemanticNameCert}(M_{p,a}, M_{p,a}, M_{p,a}, \mathsf{MonadAdjunctionEndomorphismDisplaySame}_{p,a}).$$

Proof. The empty display $(\mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$ supplies the carrier witness. Indeed, Definition 63.3 builds both empty prefix natural-transformation components from the unary hypotheses on p and a and reflexivity of $p\mathsf{hsame}p$, while the two continuation ledgers are the empty right-unit continuation. By Definition 107.1, these four fields give $M_{p,a}(\mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$.

The classifier laws are inherited componentwise from the reflexivity, symmetry, and transitivity laws for hsame. For carrier transport, suppose $M_{p,a}(unit, counit, left, right)$ and MonadAdjunctionEndomorphismDis hold. The two natural-transformation component fields in Definition 107.1 transport along the displayed hsame comparisons on *unit* and *counit*, and the two continuation ledgers transport along the displayed comparisons on all four coordinates. Repacking the transported component fields and continuation ledgers gives $M_{p,a}(unit', counit', left', right')$.

The monad endomorphism collapse theorem Definition 108.11 shows that every carried display is classifier-same to the empty display in all four coordinates. Thus the exported pattern and ledger are exactly the same degenerate adjunction-endomorphism behavior as the source carrier. Since the source, pattern, and ledger predicates are all $M_{p,a}$, the pattern-soundness and ledger-soundness rows of the semantic certificate are identity projections, as in the certificate template of Definition 9.6. □

Lean4 ✓ `BEDC.Derived.MonadUp.MonadAdjunctionEndomorphism semanticNameCert`

Closure status for $\mathsf{NameCert}_{\mathsf{Monad}\uparrow}$.

Constructive story: A $\mathsf{Monad}^{\uparrow}$ certificate is generated here from an adjunction endomorphism carrier: prefix rows, object row, unit and counit component histories, triangle-result histories, continuation ledgers, and componentwise hsame classifiers. The BHist construction collapses the displayed endomorphism unit, counit, and triangle rows to the empty identity behavior by finite carrier inversion and Cont determinacy. The semantic certificate uses the same carrier as source, pattern, and ledger, so the monad-facing name is generated from explicit adjunction data rather than from an ambient category-theoretic law object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Monad}\uparrow}, \text{obligation})$

Scope closed: The obligation surface consists of the adjunction-endomorphism carrier, triangle empty-result, bridge-empty iff, componentwise classifier, and semantic NameCert rows displayed in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Monad}\uparrow}, \text{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim arbitrary monad laws for all categories, Eilenberg-Moore constructions, Kleisli category equivalence, or host categorical equality.

Upgrade path: Scoped closure requires binding those obligations explicitly to the BHist, hsame, Cont, AdjunctionUnitCounitCarrier, and SemanticNameCert scopes.

Chapter 109

A Concrete Naming Certificate for $\mathsf{Yoneda}^{\uparrow}$

The Yoneda interface $\mathsf{Yoneda}^{\uparrow}$ packages, for each object of a category, the representable functor and the natural-isomorphism witness of the Yoneda lemma. It depends on $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62) and $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$ (chapter 63), supplying the embedding into presheaf categories.

109.1 Carrier definition

Theorem 109.1 (Yoneda representable empty component family characterization)**.** *For histories p and q, an empty natural-transformation component exists over every unary object exactly when p and q are unary prefixes with the same history:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})\Big) \iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{hs}$$

Proof. For the forward direction, evaluate the component family at the empty unary history and apply empty-component identity exactness. For the reverse direction, the same exactness theorem rebuilds each object component from the common prefix equality and the supplied unary object carrier. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family iff`

Theorem 109.2 (Yoneda empty component family e1 target characterization)**.** *For generated histories p and q, the empty component family with target $\mathsf{E1}(q)$ is equivalent to unary membership of q together with $p\mathsf{hsameE1}(q)$:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, \mathsf{E1}(q), a, \mathsf{e})\Big) \iff \mathsf{UnaryHistory}(q) \wedge p\mathsf{hsameE1}(q).$$

Proof. The forward direction applies the empty component family characterization and then inverts unary membership of the $\mathsf{E1}$ target. Conversely, unary membership of q closes $\mathsf{E1}(q)$ under unary construction, and the hsame comparison transports unary membership back to p. The general empty component family characterization then gives the displayed family. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family e1 target iff`

Theorem 109.3 (Yoneda empty component family displayed determinacy)**.** *If an empty representable component family and a displayed component family both exist over every unary object for the same prefixes p, q, then* $\mathsf{hsame}(\mathsf{e}, \mathit{displayed})$.

Proof. Evaluate both families at the empty unary object. Their component carriers share the same source and target hom-carrier endpoints, so morphism determinacy identifies the empty displayed morphism with the supplied displayed component. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family displayed deterministic`

Theorem 109.4 (Yoneda component family displayed determinacy)**.** *If two displayed component families both exist over every unary object for the same prefixes p, q, then their displayed histories are* hsame*-equivalent.*

Proof. Evaluate both families at the empty unary object. The two exposed component carriers share the same prefixed hom-carrier source and target, so category morphism determinacy identifies the two displayed component histories. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable component family displayed deterministic`

Theorem 109.5 (Yoneda empty component family component determinacy)**.** *If a is unary, an empty representable component family exists over every unary object, and a displayed component over a is supplied for the same prefixes p, q, then* $\mathsf{hsame}(\mathsf{e}, \mathit{displayed})$.

Proof. Evaluate the empty component family at the supplied unary object a. The resulting empty component and the displayed component expose category hom-carriers with the same prefixed source and target, so morphism determinacy identifies their component histories. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family component deterministic`

Theorem 109.6 (Yoneda representable empty component family target determinacy)**.** *For histories $p, q, r : \mathsf{BHist}$, if the empty natural-transformation component family from p to q exists over every unary object and the empty component family from p to r exists over every unary object, then q and r have the same history:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})\Big) \Rightarrow \Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixCom}$$

Proof. Apply Theorem 109.1 to both component families. The first family gives $\mathsf{hsame}(p, q)$ and the second gives $\mathsf{hsame}(p, r)$. Symmetry converts the first comparison to $\mathsf{hsame}(q, p)$, and transitivity composes it with $\mathsf{hsame}(p, r)$. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family target deterministic`

Theorem 109.7 (Yoneda representable empty component family source determinacy)**.** *For histories $p, q, r : \mathsf{BHist}$, if the empty natural-transformation component family from p to q exists over every unary object and the empty component family from r to q exists over every unary object, then p and r have the same history:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})\Big) \Rightarrow \Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixCom}$$

Proof. Apply Theorem 109.1 to both component families. The first family gives $\mathsf{hsame}(p, q)$ and the second gives $\mathsf{hsame}(r, q)$. Symmetry converts the second comparison to $\mathsf{hsame}(q, r)$, and transitivity composes it with $\mathsf{hsame}(p, q)$. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family source deterministic`

Theorem 109.8 (Yoneda empty component family boundary hsame equivalence)**.** *For histories p, q, r, s : BHist, suppose that the empty component family from p to q exists over every unary object, and likewise the empty component family from r to s exists over every unary object. Then comparing the sources is equivalent to comparing the targets:*

$$\mathsf{hsame}(p, r) \iff \mathsf{hsame}(q, s).$$

Proof. Apply the empty component family characterization to both families. The first family gives $\mathsf{hsame}(p, q)$ and the second gives $\mathsf{hsame}(r, s)$. For the forward direction, compose $\mathsf{hsame}(q, p)$, $\mathsf{hsame}(p, r)$, and $\mathsf{hsame}(r, s)$. For the reverse direction, compose $\mathsf{hsame}(p, q)$, $\mathsf{hsame}(q, s)$, and $\mathsf{hsame}(s, r)$. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family boundary hsame iff`

Theorem 109.9 (Yoneda empty component family boundary transport)**.** *For histories p, q, r, s : BHist, an empty component family from p to q transports to an empty component family from r to s whenever $p\mathsf{hsame}r$ and $q\mathsf{hsame}s$:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})\Big) \Rightarrow \mathsf{hsame}(p, r) \Rightarrow \mathsf{hsame}(q, s) \Rightarrow \Big(\forall a,\ \mathsf{UnaryHis}$$

Proof. Use the empty component family characterization to extract unary membership for p and q, together with $\mathsf{hsame}(p, q)$. Unary transport along the two boundary equalities gives unary membership for r and s, while transitivity of hsame composes $\mathsf{hsame}(r, p)$, $\mathsf{hsame}(p, q)$, and $\mathsf{hsame}(q, s)$. The empty-identity component characterization then rebuilds each component over the supplied unary object. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family boundary transport`

109.2 The certificate

Theorem 109.10 (Yoneda representable empty component family semantic certificate)**.** *For a unary prefix history p, the empty component family predicate*

$$Y_p(q) :\iff \forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$$

carries a semantic naming certificate with source, pattern, and ledger all equal to Y_p, and classifier hsame.

Proof. The carrier witness is $q = p$. For each unary object a, the empty-component identity exactness theorem rebuilds $\mathsf{NatTransPrefixComponentCarrier}(p, p, a, \mathsf{e})$ from unary membership of p, unary membership of a, and reflexivity of $\mathsf{hsame}(p, p)$. The classifier fields are the reflexivity, symmetry, and transitivity of hsame, while carrier transport along $\mathsf{hsame}(q, r)$ is obtained by replacing q with r in the component family. Since source, pattern, and ledger are the same predicate, the soundness rows are direct projections. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable empty component family semanticNameCert`

Capstone.

Theorem 109.11 (Prefix Yoneda embedding is fully faithful on empty components)**.** *For unary histories p and q, the empty representable component family of the prefix Yoneda construction is equivalent to prefix identity:*

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})\Big) \quad \Longleftrightarrow \quad \mathsf{hsame}(p, q).$$

Moreover, if displayed histories η and θ both occur as component families from p to q over all unary objects, then their displayed histories are identical in the BEDC history classifier:

$$\Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \eta)\Big) \wedge \Big(\forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComp}$$

Thus the semantic family Y_p classifies a unary target q exactly by $\mathsf{hsame}(p, q)$; on the empty-component skeleton this is the full and faithful prefix embedding of $\mathsf{Cat}^{\uparrow}$ through the certified $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ interfaces.

Proof. For the equivalence, apply Definition 109.1. The forward direction projects the exposed $\mathsf{hsame}(p, q)$ field from the unary-prefix package. The reverse direction combines the ambient unary hypotheses on p and q with $\mathsf{hsame}(p, q)$ and repacks them through the same characterization.

For displayed family uniqueness, apply Definition 109.4 to the two supplied component families. This is the faithfulness clause: once the source and target prefixes are fixed, the object-indexed displayed component history is determined up to hsame.

The fullness clause is the empty-component existence half of the first paragraph. The functorial source of the construction is the semantic prefix certificate of Definition 62.82; the natural transformation classifier is supplied by Definition 63.66. Together with the semantic family certificate above, these identify $Y_p(q)$ with prefix identity and leave no additional displayed component choice. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaRepresentable prefix empty component full faithfulness`

Theorem 109.12 (Yoneda component evaluation boundary)**.** *Let p and q be unary histories, and write*

$$Y_p(q) :\Longleftrightarrow \forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}).$$

For every displayed unary object a, evaluation of the prefix Yoneda empty-component family at a factors through exactly the following finite BEDC rows:

1. *the $\mathsf{Cat}^{\uparrow}$ hom-carrier endpoint row identifying the displayed empty component as the hom history from $\mathsf{append}(p, a)$ to $\mathsf{append}(q, a)$;*
2. *the $\mathsf{Functor}^{\uparrow}$ codomain-evaluation row reading the representable functor hom predicate at a as the codomain hom-carrier at those image endpoints;*
3. *the $\mathsf{NatTrans}^{\uparrow}$ component and naturality row carried by $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$.*

Conversely, if these three displayed rows are present for every unary a, then they reconstruct the accepted Yoneda component-family boundary $Y_p(q)$, and Definition 109.11 identifies that boundary with $\mathsf{hsame}(p, q)$.

Proof. Assume $Y_p(q)$. By Definition 109.1, each unary object a supplies the accepted empty component $\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$. Reading this component through the $\mathsf{NatTrans}^{\uparrow}$ surface of chapter 63 exposes the component and naturality row, while the underlying component

morphism is the $\mathsf{Cat}^{\uparrow}$ hom-carrier row with endpoints $\mathsf{append}(p, a)$ and $\mathsf{append}(q, a)$. The functor endpoint reading is precisely the codomain-evaluation row of Definition 62.98.

Conversely, suppose the endpoint hom-carrier row, the functor codomain-evaluation row, and the natural-transformation component row are present for every unary a. The third row is exactly the displayed component predicate in the definition of $Y_p(q)$, with the first two rows supplying the category and functor boundaries that it reads through. Repacking the rows over all unary objects gives $Y_p(q)$, and the prefix full-faithfulness theorem Definition 109.11 identifies the resulting boundary with prefix identity. □

Theorem 109.13 (Yoneda NatTrans functor-category hom representability)**.** *Let p and q be unary histories. In the NatTrans functor-category prefix carrier of Definition 63.102, the hom row from p to q is exactly the prefix Yoneda predicate*

$$Y_p(q) :\Longleftrightarrow \forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}).$$

Consequently this functor-category hom row is classified by $\mathsf{hsame}(p, q)$ *through Definition 109.11.*

Proof. By Definition 109.12, the predicate $Y_p(q)$ is the objectwise package of three BEDC rows: the $\mathsf{Cat}^{\uparrow}$ endpoint hom-carrier row, the $\mathsf{Functor}^{\uparrow}$ codomain-evaluation row, and the $\mathsf{NatTrans}^{\uparrow}$ component-and-naturality row.

The definition of Definition 63.102 reads a hom from p to q as precisely such a natural-transformation prefix component family over all unary object histories a. Its identity and composition clauses are the closure laws supplied by Definition 63.103, so the row is a hom row in the certified functor-category surface, not only a displayed component predicate.

It remains to identify the classifier. The full-faithfulness theorem Definition 109.11 states that, for unary histories p and q, the accepted empty-component family $Y_p(q)$ is equivalent to $\mathsf{hsame}(p, q)$. Transporting that equivalence across the hom-row reading above gives the claimed representability of the NatTrans functor-category hom row. □

Theorem 109.14 (Prefix Yoneda empty-component standard bridge)**.** *For unary histories p and q, the BEDC predicate*

$$Y_p(q) :\Longleftrightarrow \forall a,\ \mathsf{UnaryHistory}(a) \Rightarrow \mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e})$$

has a paper-side conservative bridge to the external unary-prefix presheaf readback. The decode map reads q as its finite unary prefix, and the encode map sends an external unary word back to its BEDC unary history. Bridge exactness identifies equality of external unary readbacks with $\mathsf{hsame}(p, q)$ *under the carrier $Y_p(q)$, and the displayed component choice is unique up to* hsame.

Proof. The source of the bridge is the semantic prefix certificate of Definition 62.82. The component classifier used by the readback is the natural-transformation certificate of Definition 63.66.

Carrier exactness is exactly Definition 109.1: under the unary hypotheses, $Y_p(q)$ is equivalent to $\mathsf{hsame}(p, q)$. Thus decoding q to its external unary prefix and encoding an external unary word back to a BEDC unary history preserve and reflect the same prefix equality.

For displayed components, apply Definition 109.4. Any two component choices over the same unary source and target histories are identified by hsame, so the external readback has no additional component choice beyond the BEDC classifier. □

Lean4 ✓ `BEDC.Derived.YonedaUp.YonedaUp StdBridge`

Theorem 109.15 (Prefix Yoneda empty-component mature interface)**.** *For unary histories p and q, the mature scoped interface of the prefix Yoneda object is exactly the prefix empty-component skeleton. Its carrier is $Y_p(q)$, its classifier is* hsame*, its full-faithfulness clause is Theorem 109.11, its conservative paper bridge is Theorem 109.14, and its semantic certificate is Theorem 109.10. The scope is restricted to unary prefix histories and empty components.*

Proof. The carrier and classifier are fixed by the semantic certificate in Theorem 109.10: the family $Y_p(q)$ is the empty-component representable family, and the classifier it exposes is hsame.

The fully faithful part is Theorem 109.11. It identifies the object-indexed carrier over all unary objects with $\mathsf{hsame}(p, q)$ and makes the displayed component history unique up to hsame.

The bridge part is Theorem 109.14. Its encode and decode readings are conservative for the same unary-prefix carrier, so composing the semantic certificate, the full-faithfulness clause, and this bridge gives the stated mature interface without enlarging the domain. No step ranges beyond unary prefix histories or beyond empty components. □

Closure status for $\mathsf{NameCert}_{\mathsf{Yoneda}^{\uparrow}}$**.**

Constructive story: The scoped Yoneda object is built from unary prefixes by asking for an empty natural-transformation component over every unary object: for prefixes p and q, the displayed family $Y_p(q)$ is admitted exactly when those prefixes are unary and history-same. The carrier therefore accepts the empty-component family and excludes arbitrary component choices or non-unary presheaf data, while the classifier is hsame on the prefix histories and determinacy theorems force any displayed component family back to the same empty component. The source is the semantic prefix functor surface together with the prefix natural-transformation component certificate, the pattern records the empty-component characterization, source and target determinacy, boundary transport, and prefix full-faithfulness, and the ledger keeps the component family and its evaluated component histories visible. Since e supplies the component choice and hsame is reflexive, symmetric, and transitive on histories, the name-certificate obligations are carried by the semantic certificate, with closure under the classifier given by the boundary transport rows. A history h lies in the signature gap of a $\mathsf{Yoneda}^{\uparrow}$ token p_0 when the source-domain accepts h as an empty-component family, the signature relation pairs it with an answer prefix and evidence ledger, and token introduction names that answer as p_0; unfolded, every unary object sees the empty component and the target prefix is hsame to the representing prefix encoded by p_0. The paper bridge decodes a BEDC unary prefix to an external finite unary word and encodes such a word back to the same unary history, so equality of external readbacks is conservative for the same prefix classifier. The chapter registers this prefix empty-component skeleton under the name $\mathsf{Yoneda}^{\uparrow}$; it gives the fully faithful representable fragment without introducing a general presheaf category, arbitrary Yoneda lemma, or external natural-isomorphism carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Yoneda}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope is exactly the checked semantic certificate, full-faithfulness, and paper bridge for the prefix empty-component skeleton.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Yoneda}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.YonedaUp.YonedaRepresentable\_empty\_component\_family\_semanticNameCert`

Bridge status: bridgeChecked

Not claimed: This chapter does not close the arbitrary Yoneda lemma, general presheaf categories, enriched variants, density, or checked bridge targets.

Upgrade path: No further formal-axis upgrade remains for the checked prefix empty-component bridge.

Chapter 110

A Concrete Naming Certificate for $\mathsf{EquivCat}^{\uparrow}$

The categorical-equivalence interface $\mathsf{EquivCat}^{\uparrow}$ packages a pair of functors that are mutually quasi-inverse together with the natural-isomorphism witnesses. It depends on $\mathsf{NameCert}_{\mathsf{Adjunction}^{\uparrow}}$ (chapter 107) and $\mathsf{NameCert}_{\mathsf{NatTrans}^{\uparrow}}$ (chapter 63), and is the structural notion of sameness for $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ instances.

110.1 Carrier definition

Theorem 110.1 (Empty roundtrip identity components)**.** *For histories* $p, q, a, unit, counit, left, right$: BHist*, if an adjunction unit-counit carrier has both displayed roundtrip results* $left$ *and* $right$ *classifier-equal to* e*, then the unit and counit components are the empty identity components:*

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}) \quad \textit{and} \quad \mathsf{NatTransPrefixComponentCarrier}(q, p, a, \mathsf{e}).$$

Proof. The empty left roundtrip continuation forces both *unit* and *counit* to be e. The empty right roundtrip gives the same collapse in the opposite order. After these endpoint inversions, the component fields of the adjunction unit-counit carrier are exactly the two empty identity components. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip identity components`

Theorem 110.2 (Equivalence empty roundtrip prefix determinacy)**.** *For histories* $p, q, a, unit, counit, left, right$: BHist*, if* $\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)$ *holds and* $left\mathsf{hsamee}$ *and* $right\mathsf{hsamee}$*, then*

$$p\mathsf{hsame}q, \qquad unit\mathsf{hsamee}, \qquad counit\mathsf{hsamee},$$

and

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}) \quad \textit{and} \quad \mathsf{NatTransPrefixComponentCarrier}(q, p, a, \mathsf{e}).$$

Proof. Apply Theorem 110.1 to the adjunction unit-counit carrier and the two empty displayed triangle results. This gives the two empty natural-transformation prefix components from p to q and from q back to p at the object history a.

The NatTrans empty-identity exactness theorem applied to the first empty component extracts the prefix classifier equality $p\mathsf{hsame}q$.

The stored triangle continuations in the adjunction unit-counit carrier have results $left$ and $right$. Empty-result inversion for the displayed continuations, together with $left\mathsf{hsamee}$ and $right\mathsf{hsamee}$, collapses the unit and counit histories to $unit\mathsf{hsamee}$ and $counit\mathsf{hsamee}$. Collecting these conclusions gives the stated determinacy package. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip prefix determinacy`

Theorem 110.3 (Equivalence empty roundtrip zero-source exclusion)**.** *For histories* $tail, q, a, unit, counit, left, rig$ BHist*, no adjunction unit-counit carrier can start its first prefix source with* $\mathsf{E0}(tail)$ *while both displayed roundtrip results classify to* e*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(\mathsf{E0}(tail), q, a, unit, counit, left, right) \Rightarrow left\mathsf{hsamee} \Rightarrow right\mathsf{hsamee} \Rightarrow \bot.$$

Proof. The empty roundtrip component theorem extracts the displayed empty natural transformation prefix component out of the zero-headed source. The zero-headed component exclusion for natural transformations contradicts this component by exposing the visible zero constructor at the source head. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip zero source absurd`

Theorem 110.4 (Equivalence empty roundtrip zero-target exclusion)**.** *For histories* $p, tail, a, unit, counit, left, rig$ BHist*, no adjunction unit-counit carrier can place a zero-headed prefix at its first target while both displayed roundtrip results classify to* e*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, \mathsf{E0}(tail), a, unit, counit, left, right) \Rightarrow left\mathsf{hsamee} \Rightarrow right\mathsf{hsamee} \Rightarrow \bot.$$

Proof. The empty roundtrip component theorem extracts the reverse empty natural transformation prefix component. In that reverse component, the zero-headed target of the adjunction becomes the displayed source prefix. The NatTrans zero-headed component exclusion then contradicts the visible $\mathsf{E0}$ constructor at that source head. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip zero target absurd`

Theorem 110.5 (Equivalence empty roundtrip $\mathsf{E1}$ source target exactness)**.** *For histories* $tail, q, a, unit, counit, lef$ BHist*, if an adjunction unit-counit carrier starts at the unary source prefix* $\mathsf{E1}(tail)$ *and both displayed roundtrip results classify to* e*, then the first target prefix is exactly* $\mathsf{E1}(tail)$*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(\mathsf{E1}(tail), q, a, unit, counit, left, right) \Rightarrow left\mathsf{hsamee} \Rightarrow right\mathsf{hsamee} \Rightarrow q = \mathsf{E1}(tail).$$

Proof. The empty roundtrip component theorem exposes the displayed empty natural-transformation component from $\mathsf{E1}(tail)$ to q. The NatTrans empty-component inversion for an $\mathsf{E1}$ source reads back the component data: the tail is unary, the object history is unary, and the displayed target prefix is exactly $\mathsf{E1}(tail)$. The last field is the required equality. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip e1 source target eq`

Theorem 110.6 (EquivCat empty roundtrip $\mathsf{E1}$ prefix tail determinacy)**.** *For histories* $sourceTail, targetTail, a, u$ BHist*, if an adjunction unit-counit carrier is displayed from* $\mathsf{E1}(sourceTail)$ *to* $\mathsf{E1}(targetTail)$ *and both roundtrip results classify to* e*, then*

$$sourceTail\mathsf{hsame}targetTail.$$

Proof. The empty roundtrip prefix determinacy theorem identifies the full prefixes $\mathsf{E1}(sourceTail)$ and $\mathsf{E1}(targetTail)$ by hsame. Inverting the one-headed constructor under hsame reads this back to $sourceTail\mathsf{hsame}targetTail$. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip e1 prefix tail hsame`

Theorem 110.7 (EquivCat empty roundtrip target-prefix determinacy)**.** *For histories* $p, q, r, a, unit, counit, left, r$ BHist*, if* $\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)$ *holds,* $left\mathsf{hsamee}$ *and* $right\mathsf{hsamee}$*, and*

$$\mathsf{NatTransPrefixComponentCarrier}(p, r, a, \mathsf{e})$$

is also displayed, then $q\mathsf{hsame}r$*.*

Proof. The empty roundtrip identity-components theorem extracts the reverse empty component

$$\mathsf{NatTransPrefixComponentCarrier}(q, p, a, \mathsf{e})$$

from the adjunction carrier and the two empty displayed roundtrips. Composing that empty component with the displayed empty component from p to r by NatTrans empty-identity prefix transitivity identifies the target prefix q with r. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip target prefix deterministic`

Theorem 110.8 (EquivCat empty roundtrip unit-counit equivalence)**.** *For histories* $p, q, a, unit, counit, left, right$ BHist*, if* $\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)$ *holds, then the two displayed roundtrip results classify to* e *exactly when the unit and counit components classify to* e*:*

$$left\mathsf{hsamee} \ \wedge \ right\mathsf{hsamee} \quad \Longleftrightarrow \quad unit\mathsf{hsamee} \ \wedge \ counit\mathsf{hsamee}.$$

Proof. The forward implication is the unit-counit part of the empty roundtrip prefix determinacy theorem. Conversely, after the unit and counit are identified with e, the stored triangle continuations have the same source and suffix as the right-unit continuation at e; continuation determinism identifies both displayed roundtrip results with e. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip unit counit empty iff`

Theorem 110.9 (EquivCat empty roundtrip unit-counit determinacy)**.** *For histories* $p, q, a, unit, unit', counit, cou$ BHist*, two carriers*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right) \qquad \textit{and} \qquad \mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit', c$$

whose left and right triangle results are all hsame *to* e *determine the same unit and counit up to* hsame*.*

Proof. Apply empty roundtrip prefix determinacy to both adjunction carriers. Each application identifies its unit and counit with e. Transitivity of hsame, using symmetry on the second pair of empty identifications, gives $unit\mathsf{hsame}unit'$ and $counit\mathsf{hsame}counit'$. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip unit counit deterministic`

Theorem 110.10 (EquivCat empty roundtrip source-prefix determinacy)**.** *For histories* $p, q, r, a, unit, counit, left,$ BHist*, if* $\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right)$ *holds,* $left\mathsf{hsamee}$ *and* $right\mathsf{hsamee}$*, and*

$$\mathsf{NatTransPrefixComponentCarrier}(r, q, a, \mathsf{e})$$

is also displayed, then $p\mathsf{hsame}r$*.*

Proof. The empty roundtrip identity-components theorem extracts the reverse empty component from q to p. Compose the displayed empty component from r to q with that reverse component by NatTrans empty-identity prefix transitivity; this gives $r\mathsf{hsame}p$. Symmetry of hsame gives the stated source-prefix equality. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip source prefix deterministic`

110.2 The certificate

Theorem 110.11 (Equivalence cycle identity carrier semantic certificate)**.** *For BEDC categorical hom carriers* $left : \mathsf{CategoryHomCarrier}(a, b, f)$ *and* $right : \mathsf{CategoryHomCarrier}(b, a, g)$*, the carrier of displayed cycle identities*

$$\mathsf{CategoryHomCarrier}(a, b, t) \wedge \mathsf{CategoryHomCarrier}(b, a, g) \wedge t\mathsf{hsamee} \wedge g\mathsf{hsamee}$$

has a semantic naming certificate with classifier hsame*.*

Proof. The cycle-tail theorem for categorical hom carriers identifies both cycle tails with the empty history. Transporting $left$ along the displayed tail comparison gives the witness at e, while $right$ supplies the fixed reverse carrier. The classifier laws are the finite-kernel equivalence laws for hsame. Carrier transport keeps the endpoints fixed, transports the first hom carrier along the classifier, and composes the inverse classifier with the stored empty-tail proof. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatCycleIdentityCarrier semanticNameCert`

Theorem 110.12 (EquivCat empty roundtrip identity carrier target-prefix hsame transport)**.** *For histories* $p, q, q', a, unit, counit, left, right : \mathsf{BHist}$*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right), \qquad left\mathsf{hsamee}, \qquad right\mathsf{hsamee}, \qquad q\mathsf{hsame}q'.$$

Then the empty roundtrip identity carrier transports to the target prefix q'*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q', a, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}),$$

and its empty identity component fields are

$$\mathsf{NatTransPrefixComponentCarrier}(p, q', a, \mathsf{e}) \quad \textit{and} \quad \mathsf{NatTransPrefixComponentCarrier}(q', p, a, \mathsf{e}).$$

Proof. Apply Definition 110.2 to the original carrier and the two empty roundtrip witnesses. This gives $p\mathsf{hsame}q$ and the empty component

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}).$$

Reading this component through Definition 63.3 gives unary histories for p, q, and a. Unary transport along $q\mathsf{hsame}q'$ gives unary membership of q', and transitivity of hsame gives $p\mathsf{hsame}q'$.

The reverse implication of Definition 107.5, applied with both triangle results equal to e, now gives

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q', a, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}),$$

because the two triangle-result classifiers are reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Finally, Definition 107.1 projects the first two fields of this carrier, giving the two displayed empty identity components for the transported target prefix. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatCycleIdentityCarrier target prefix hsame transport`
Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip target prefix hsame transport`

Theorem 110.13 (EquivCat empty roundtrip identity carrier source-prefix hsame transport)**.** *For histories* $p, p', q, a, unit, counit, left, right$: BHist*, suppose*

$$\mathsf{AdjunctionUnitCounitCarrier}(p, q, a, unit, counit, left, right), \qquad left\mathsf{hsamee}, \qquad right\mathsf{hsamee}, \qquad p\mathsf{hsame}p'.$$

Then the empty roundtrip identity carrier transports to the source prefix p'*:*

$$\mathsf{AdjunctionUnitCounitCarrier}(p', q, a, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}),$$

and its empty identity component fields are

$$\mathsf{NatTransPrefixComponentCarrier}(p', q, a, \mathsf{e}) \quad \textit{and} \quad \mathsf{NatTransPrefixComponentCarrier}(q, p', a, \mathsf{e}).$$

Proof. Apply Definition 110.2 to the original carrier and the two empty roundtrip witnesses. This gives $p\mathsf{hsame}q$ and the empty component

$$\mathsf{NatTransPrefixComponentCarrier}(p, q, a, \mathsf{e}).$$

Reading this component through Definition 63.3 gives unary histories for p, q, and a. Unary transport along $p\mathsf{hsame}p'$ gives unary membership of p', and transitivity of hsame, using symmetry on $p\mathsf{hsame}p'$, gives $p'\mathsf{hsame}q$.

The reverse implication of Definition 107.5, applied with both triangle results equal to e, now gives

$$\mathsf{AdjunctionUnitCounitCarrier}(p', q, a, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}),$$

because the two triangle-result classifiers are reflexivity of $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. Finally, Definition 107.1 projects the first two fields of this carrier, giving the two displayed empty identity components for the transported source prefix. □

Lean4 ✓ `BEDC.Derived.EquivCatUp.EquivCatAdjunction empty roundtrip source prefix hsame transport`

Closure status for $\mathsf{NameCert}_{\mathsf{EquivCat}\uparrow}$**.**

Constructive story: An EquivCat$^{\uparrow}$ certificate is generated from an adjunction unit-counit carrier with source and target prefix histories, object row, unit and counit component rows, two roundtrip result rows, continuation ledgers, and hsame transport. The BEDC carrier reads categorical equivalence through finite empty-roundtrip evidence: when both triangle results are empty, the unit and counit components are forced to empty identity components, and prefix sameness is recovered by the NatTrans classifier. The construction therefore names the equivalence interface through displayed adjunction rows, not through ambient category equality.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EquivCat}\uparrow}$, seed)

Scope closed: This chapter binds the equivalence empty-roundtrip component, prefix determinacy, exclusion, and transport theorems.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EquivCat}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim general categorical equivalence of arbitrary categories, univalence, quotient categories, or host isomorphism equality.

Upgrade path: To reach obligationClosure, package the checked empty-roundtrip adjunction theorems as the explicit EquivCatUp NameCert carrier.

Chapter 111

A Concrete Naming Certificate for $\mathsf{Set}^{\uparrow}$

The set interface $\mathsf{Set}^{\uparrow}$ packages a carrier together with a psame-classifier interpretable as set-membership equality. It is the type-layer counterpart of $\mathsf{NameCert}_{\mathsf{Preorder}^{\uparrow}}$ (chapter 52) at the discrete level, and the substrate for $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112) and $\mathsf{NameCert}_{\mathsf{Subtype}^{\uparrow}}$ (chapter 113).

111.1 Carrier definition

111.2 The certificate

Theorem 111.1 (Set carrier obligation)**.** *The carrier row of a* $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ *is a predicate* $A : \mathsf{BHist} \to \mathsf{Prop}$ *over BEDC histories. It is generated by membership-visible histories: a history* h *is carried exactly when the* Hist *presentation exposes a membership row for* h*, and the visible row is read through* BHist *rather than through a host-side set object.*

Proof. Unpack the source package as a Pkg row over Hist and read its public histories through the BHist interface. The carrier predicate records precisely those histories whose membership row is visible to the package boundary.

The construction is constructive because no host enumeration or host subset is selected. A carried witness is the displayed membership row together with its Pkg provenance, and forgetting the witness returns the original BHist history. This is the carrier surface consumed later by $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Subtype}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.SetUp.SetCarrier obligation surface`

Theorem 111.2 (Set classifier obligation)**.** *For a fixed* $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ *carrier* A*, the classifier row is a* psame*-style membership classifier on carried histories. Two carried histories are classified together exactly when their membership-visible rows are identified by the BEDC presentation classifier; the row is not host equality of set elements and is not a quotient by an external relation.*

Lean4 ✓ `BEDC.Derived.SetUp.SetMembershipVisibleClassifier carrier hsame boundary`
Lean4 ✓ `BEDC.Derived.SetUp.SetMembershipVisibleClassifier carrier rows`

Proof. The classifier is read from the NameCert row attached to the package, so its inputs are carried histories with membership witnesses in A. Applying the psame classifier compares the visible membership rows and returns the certificate-level sameness relation.

Because the comparison is made inside psame, no appeal is made to host equality. Because the witnesses remain in the Pkg ledger, no quotient carrier is formed. The result is the classifier boundary required before finite enumeration in $\mathsf{FinSet}^{\uparrow}$ or predicate restriction in $\mathsf{Subtype}^{\uparrow}$ may consume the set source. □

Lean4 ✓ `BEDC.Derived.SetUp.SetPkgMembership semantic name certificate`

Theorem 111.3 (Set membership transport obligation)**.** *The* $\mathsf{Set}^{\uparrow}$ *membership row is stable under* hsame *transport and under the* psame *classifier. If h is carried by A, $h\,\mathsf{hsame}\,h'$, and the visible membership rows for h and h' are related by the classifier, then h' is carried by A with the transported membership witness and the same package provenance boundary.*

Lean4 ✓ `BEDC.Derived.SetUp.SetMembershipVisibleTransportChain carrier transport`

Proof. First transport the displayed Hist membership row along hsame. The transport changes only the presentation of the history, while the Pkg provenance records that the read remains inside the same certificate boundary.

Next apply the psame classifier row to the transported membership presentation. This supplies the certificate-level identification between the original membership read and the transported read.

Combining the transported witness with the classifier evidence yields a carried witness for h'. The dependency remains constructive and visible: the proof uses only NameCert, psame, hsame, and Pkg, so downstream $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Subtype}^{\uparrow}$ consumers receive a stable membership row rather than an external equality principle. □

Theorem 111.4 (Set extensionality ledger obligation)**.** *The* $\mathsf{Set}^{\uparrow}$ *extensionality ledger records exactly the visible membership rows and their exact package boundary. Finite consumers such as* $\mathsf{FinSet}^{\uparrow}$ *and restricted consumers such as* $\mathsf{Subtype}^{\uparrow}$ *may consume the carrier A, the* psame *classifier, and the* hsame*-stable membership transport row, but they may not inspect hidden package data outside the ledger boundary.*

Lean4 ✓ `BEDC.Derived.SetUp.SetExtensionalityLedger semantic name certificate`

Proof. The extensionality row compares two $\mathsf{Set}^{\uparrow}$ presentations by their visible membership classifications. If the same BHist histories are carried and their visible membership rows agree by psame, the public set presentation agrees at the certificate boundary.

The exact ledger boundary is the accompanying Pkg provenance for those membership reads. It records which rows are exposed and which witnesses remain hidden, so finite enumeration in $\mathsf{FinSet}^{\uparrow}$ can use only occurrence of visible members and $\mathsf{Subtype}^{\uparrow}$ can use only visible restriction of the carrier.

Thus the ledger gives an extensionality obligation without exporting a host set, a host equality, or any hidden construction witness. The only downstream dependency made available is the membership-visible carrier, classifier, transport, and exact boundary described above. □

Theorem 111.5 (Set public NameCert export)**.** *The public* $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ *surface exported by the set construction is exactly the membership-visible* BHist *carrier row of Definition 111.1, the* psame*-style classifier row of Definition 111.2, the* hsame *membership-transport row of Definition 111.3, and the extensionality ledger boundary of Definition 111.4. As a public certificate it exposes those*

four rows as the whole downstream surface for $\mathsf{FinSet}^{\uparrow}$ *and* $\mathsf{Subtype}^{\uparrow}$*, and it exports no host set, no host equality, no quotient carrier, and no hidden package witness.*

Proof. The carrier clause is the membership-visible row supplied by Definition 111.1. The classifier clause is the psame boundary supplied by Definition 111.2, with the checked row `BEDC.Derived.SetUp.SetMembers` recording that the carrier and classifier rows are the visible rows of the certificate.

The transport clause is the hsame-stable membership row supplied by Definition 111.3, supported by the checked row `BEDC.Derived.SetUp.SetMembershipVisibleTransportChain_hsame`. The membership row itself is the semantic package row recorded by `BEDC.Derived.SetUp.SetPkgMembership_sem`

The ledger clause is Definition 111.4, supported by `BEDC.Derived.SetUp.SetExtensionalityLedger_semanti` That theorem states the boundary condition: visible membership rows may be consumed downstream, while host sets, host equality, quotient carriers, and hidden package witnesses remain outside the exported surface. These four clauses therefore assemble the public $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ certificate and nothing else. □

Lean4 ✓ `BEDC.Derived.SetUp.SetPublicNameCert transport target carrier`

Theorem 111.6 (Set public NameCert standard bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ *surface of Definition 111.5 gives a paper-level conservative standard bridge to the ordinary membership presentation of visible BEDC membership rows. Forward readback is the carried* BHist *membership row, reverse admission accepts only displayed membership rows, classifier preservation is the* psame *row of Definition 111.2, and no host data is exposed beyond the extensionality ledger of Definition 111.4.*

Proof. Start with the public certificate assembled in Definition 111.5. Its carrier component is exactly the membership-visible BHist row supplied by Definition 111.1, so reading from the standard side back into BEDC returns the carried membership row and no hidden host set.

For admission in the other direction, require the ordinary-side row to be displayed as a visible BEDC membership row. The membership-transport obligation of Definition 111.3 keeps that row stable under hsame transport, so the admitted row remains inside the same public certificate boundary.

Classifier preservation is then the psame classifier supplied by Definition 111.2. The bridge compares only visible membership rows, hence it preserves the certificate classifier without replacing it by host equality or a quotient relation.

Finally, Definition 111.4 supplies the no-host-leak clause. Its ledger records exactly which membership rows and package provenance are public, so composing these four obligations with the export theorem gives a conservative paper-level standard bridge for the visible $\mathsf{Set}^{\uparrow}$ membership presentation. □

Lean4 ✓ `BEDC.Derived.SetUp.SetPublicNameCert standard bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$**.**

Constructive story: A set in this certificate is not a host subset but a membership-visible package of BEDC histories: a history h enters the carrier exactly when the public Pkg boundary exposes a membership row for h. Two carried histories are classified by the package's psame membership classifier, with hsame transport moving the displayed history presentation while the membership provenance remains in the ledger. The source row is the same visible membership predicate, the pattern records that all comparisons pass through psame rather than host equality, and the extensionality ledger says which membership rows and package boundary are public while hidden witnesses stay hidden. The five name-certificate obligations are supplied by any displayed membership row for inhabitation, by the reflexive, symmetric, and transitive fields of the underlying certificate classifier, and by the transport row that carries membership along hsame and psame without leaving the package. A history h lies in the signature gap of a $\mathsf{Set}^{\uparrow}$ token p when the source-domain accepts its visible membership row, the signature relation pairs that row with an answer membership presentation and evidence ledger, and

token introduction names that answer as p; unfolded, h must be one of the public membership histories classified by the token's membership row. The standard bridge reads this surface as the ordinary membership presentation of visible BEDC rows and reads back only displayed membership rows, preserving psame and the no-host-leak ledger. The chapter registers this membership-visible construction under the name $\mathsf{Set}^{\uparrow}$; sethood is a public carrier/classifier boundary over histories, not a host set object, quotient carrier, hidden package witness, or external equality principle.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge scope is the membership-visible carrier, psame classifier, hsame transport, extensionality ledger, and public NameCert export of Definition 111.5.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SetUp.SetPublicNameCert\_standard\_bridge`

Bridge status: paperBridge

Not claimed: This chapter does not close bridgeChecked verification, downstream $\mathsf{FinSet}^{\uparrow}$ or $\mathsf{Subtype}^{\uparrow}$ certificates, or Lean verification of the paper-level bridge theorem.

Upgrade path: No further paper-side theory upgrade remains for the membership-visible SetUp surface; bridgeChecked verification remains a separate formal-axis target.

Chapter 112

A Concrete Naming Certificate for $\mathsf{FinSet}^{\uparrow}$

The finite-set interface $\mathsf{FinSet}^{\uparrow}$ packages a $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ together with a finite enumeration witness via $\mathsf{NameCert}_{\mathsf{List}^{\uparrow}}$ (chapter 36). It depends on $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ (chapter 111) and is the substrate for combinatorial certificates such as $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$.

112.1 Carrier definition

112.2 The certificate

Definition 112.1 (Finite-spine enumeration membership)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^{\uparrow}$ certificate with classifier $\sim_A$. For a finite BEDC-history spine $xs : \mathsf{HistSpine}$, let $\mathsf{Pos}(xs)$ be its finite position carrier and let $\mathsf{entry}_{xs} : \mathsf{Pos}(xs) \to \mathsf{BHist}$ be its entry readback. Define enumeration membership by*

$$\mathsf{EnumMem}_{A,\sim_A}(z, xs) \iff A(z) \wedge \exists i : \mathsf{Pos}(xs).\ A(\mathsf{entry}_{xs}(i)) \wedge z \sim_A \mathsf{entry}_{xs}(i).$$

The enumeration-induced $\mathsf{FinSet}^{\uparrow}$ classifier between two finite-spine presentations is

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, ys) \iff \forall z : \mathsf{BHist}.\ \mathsf{EnumMem}_{A,\sim_A}(z, xs) \iff \mathsf{EnumMem}_{A,\sim_A}(z, ys).$$

This is the extensional classifier read from the enumeration witness used by the finite-set presentation in chapter 112.

Definition 112.2 (Classifier-preserving enumeration permutation)**.** *For finite BEDC-history spines xs and ys, define $\mathsf{EnumPerm}_{A,\sim_A}(xs, ys)$ to mean that there is a certified finite-position bijection*

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$$

such that every matched pair of entries is carried by the fixed $\mathsf{Set}^{\uparrow}$ source and classified by its classifier:

$$\forall i : \mathsf{Pos}(xs).\ A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{ys}(\pi(i))) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

The inverse position map π^{-1} is part of the certified bijection. This is the enumeration-level presentation datum used by the $\mathsf{Permutation}^{\uparrow}$ interface in chapter 116.

Theorem 112.3 (FinSet classifier invariant under enumeration permutation)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate, let* $\sim_A$ *be its classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. For finite BEDC-history spines* xs *and* ys*, if*

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, ys)$$

holds, then

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, ys)$$

holds. Equivalently, replacing a $\mathsf{FinSet}^{\uparrow}$ *enumeration witness by a certified permutation that preserves the fixed source classifier pointwise leaves the induced finite-set classifier unchanged.*

Proof. Unfold Definition 112.2 and choose the certified bijection π with its pointwise classifier-preservation field. Unfold Definition 112.1. It is enough to fix a query history z and prove the two membership implications.

For the forward implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$. Then $A(z)$ holds and there is a position $i : \mathsf{Pos}(xs)$ such that $A(\mathsf{entry}_{xs}(i))$ and

$$z \sim_A \mathsf{entry}_{xs}(i)$$

hold. The permutation field at i gives $A(\mathsf{entry}_{ys}(\pi(i)))$ and

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

The transitivity field of the naming certificate from Definition 9.7, applied to these two classifier witnesses, gives

$$z \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

Using the position $\pi(i)$ and the carried-entry witness just obtained, this repacks as $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$.

For the reverse implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$. Then $A(z)$ holds and there is a position $j : \mathsf{Pos}(ys)$ such that $A(\mathsf{entry}_{ys}(j))$ and

$$z \sim_A \mathsf{entry}_{ys}(j)$$

hold. Put $i = \pi^{-1}(j)$. The pointwise preservation field at i gives

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(j),$$

because $\pi(i) = j$, and it also gives $A(\mathsf{entry}_{xs}(i))$. The symmetry field of cert reverses the matched-entry comparison to

$$\mathsf{entry}_{ys}(j) \sim_A \mathsf{entry}_{xs}(i).$$

Transitivity of the same classifier composes this with $z \sim_A \mathsf{entry}_{ys}(j)$ and yields

$$z \sim_A \mathsf{entry}_{xs}(i).$$

The position i therefore witnesses $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$.

The two implications hold for every query history z, so the two enumeration-induced predicates are pointwise equivalent. By Definition 112.1, this is exactly $\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, ys)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationClassifier permutation invariant`

Definition 112.4 (Finite-spine occurrence)**.** *For a finite spine* $xs : \mathsf{HistSpine}$ *and a history* $x : \mathsf{BHist}$*, the predicate* $\mathsf{SpineMember}(x, xs)$ *is generated by two clauses:* $\mathsf{SpineMember}(x, \mathsf{cons}(x, ys))$ *for every tail* ys*, and if* $\mathsf{SpineMember}(x, ys)$ *then* $\mathsf{SpineMember}(x, \mathsf{cons}(y, ys))$ *for every head* y*. There is no clause for* nil*.*

Definition 112.5 (FinSet enumeration carrier)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of the underlying* $\mathsf{NameCert}_{\mathsf{Set}^\uparrow}$*, let* $\sim_A$ *be its classifier, and let* $xs : \mathsf{HistSpine}$ *be the finite spine supplied by the* $\mathsf{List}^\uparrow$ *enumeration witness of Definition 36.57. Define the finite-set carrier represented by* xs *by*

$$\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(a) \iff A(a) \wedge \exists x : \mathsf{BHist}.\ \mathsf{SpineMember}(x, xs) \wedge a \sim_A x.$$

Lean4 (definition) `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier`

Lemma 112.6 (Enumeration spine entries are source-carried)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be a BEDC history source carrier. If* $\mathsf{ListSpineRep}_A(e, xs)$ *and* $\mathsf{SpineMember}(x, xs)$ *hold, then* $A(x)$ *holds.*

Proof. Induct on the displayed $\mathsf{ListSpineRep}_A(e, xs)$ witness from Definition 36.57. The nil representation has $xs = \mathsf{nil}$, and Definition 112.4 gives no occurrence clause for nil. In the cons representation, the occurrence of x in $\mathsf{cons}(a, xs')$ is either the head occurrence or a tail occurrence. The head case returns the stored source witness $A(a)$. The tail case applies the induction hypothesis to the tail representation $\mathsf{ListSpineRep}_A(t, xs')$ and the displayed tail occurrence. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationBundle member source carried`

Theorem 112.7 (FinSet enumeration membership exactness)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of the underlying* $\mathsf{NameCert}_{\mathsf{Set}^\uparrow}$*, let* $\sim_A$ *be its classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and let* e *and* xs *satisfy* $\mathsf{ListSpineRep}_A(e, xs)$*. Then for every history* a*,*

$$\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(a) \iff \exists x : \mathsf{BHist}.\ \mathsf{SpineMember}(x, xs) \wedge a \sim_A x.$$

Proof. For the forward direction, unpack Definition 112.5 and keep the displayed occurrence and classifier witness.

For the reverse direction, take a witness x with $\mathsf{SpineMember}(x, xs)$ and $a \sim_A x$. The enumeration witness and Definition 112.6 give $A(x)$. The symmetry field of cert, as part of Definition 9.7, turns $a \sim_A x$ into $x \sim_A a$. Carrier transport from the same certificate carries $A(x)$ along this comparison to obtain $A(a)$. Pair this carrier witness with the original occurrence and classifier witness, and repack by Definition 112.5. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier membership exactness`

Definition 112.8 (Projection-stable finite membership readback)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of the underlying* $\mathsf{NameCert}_{\mathsf{Set}^\uparrow}$*, let* $\sim_A$ *be its classifier, and let* $xs : \mathsf{HistSpine}$ *be a finite enumeration spine. A projection readback row for* xs *is a carried map*

$$\mathsf{proj} : \mathsf{BHist} \to \mathsf{BHist}$$

whose fixed histories are exactly the histories represented by the finite enumeration:

$$\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(z) \iff \mathsf{hsame}(\mathsf{proj}(z), z).$$

The associated predicate records both sides of that readback,

$$\mathsf{ProjectionStableReadback}_{A,\sim_A,xs,\mathsf{proj}}(z) \iff \mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(z) \wedge \mathsf{hsame}(\mathsf{proj}(z), z).$$

Thus finite membership is read from a fixed-point row over a finite enumeration carrier, rather than from an external set-theoretic primitive.

Lean4 (definition) `BEDC.Derived.FinsetUp.FinsetProjectionStableReadback`

Theorem 112.9 (Projection-stable readback is finite membership)**.** *Assume the projection readback row above. Then for every history z,*

$$\mathsf{ProjectionStableReadback}_{A,\sim_A,xs,\mathsf{proj}}(z) \iff \mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(z),$$

and also

$$\mathsf{ProjectionStableReadback}_{A,\sim_A,xs,\mathsf{proj}}(z) \iff \mathsf{hsame}(\mathsf{proj}(z), z).$$

Proof. Unfold Definition 112.8. The predicate is the conjunction of finite-enumeration membership and projection stability, and the readback row identifies these conjuncts pointwise. Projection gives each implication out of the readback predicate; either conjunct conversely supplies the other and repacks the conjunction. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetProjectionStableReadback membership iff`
Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetProjectionStableReadback stability iff`

Theorem 112.10 (Projection-stable readback respects the source classifier)**.** *Let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$ *and assume the same projection readback row. If $a \sim_A b$ and* $\mathsf{hsame}(\mathsf{proj}(a), a)$*, then*

$$\mathsf{hsame}(\mathsf{proj}(b), b).$$

Proof. The readback row converts stability of a into finite-enumeration membership for a. The finite-set enumeration carrier is stable under the source classifier: carrier transport carries $A(a)$ to $A(b)$, and classifier symmetry and transitivity carry the enumeration witness from a to b. Applying the readback row again converts membership for b into projection stability. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier classifier transport`
Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetProjectionStableReadback classifier transport`

Lemma 112.11 (Finite-spine occurrence has a position readback)**.** *For every finite BEDC-history spine* $xs : \mathsf{HistSpine}$ *and history* $y : \mathsf{BHist}$*, if* $\mathsf{SpineMember}(y, xs)$ *holds, then*

$$\exists j : \mathsf{Pos}(xs).\ \mathsf{entry}_{xs}(j) = y.$$

Proof. Induct on the generated occurrence witness from Definition 112.4. In the head clause $\mathsf{SpineMember}(y, \mathsf{cons}(y, ys))$, choose the head position of $\mathsf{cons}(y, ys)$; its readback is y. In the tail clause, the induction hypothesis gives a position $j : \mathsf{Pos}(ys)$ with $\mathsf{entry}_{ys}(j) = y$. Use the tail-position inclusion $\mathsf{Pos}(ys) \to \mathsf{Pos}(\mathsf{cons}(x, ys))$; its readback is the same tail entry, so it witnesses the displayed equality for $\mathsf{cons}(x, ys)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetBundleMember position readback`

Theorem 112.12 (FinSet duplicate insertion invariance)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate, let* $\sim_A$ *be its classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. For a finite BEDC-history spine* $xs : \mathsf{HistSpine}$ *and histories* $x, y : \mathsf{BHist}$*, if*

$$A(x) \wedge A(y) \wedge \mathsf{SpineMember}(y, xs) \wedge x \sim_A y,$$

then

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(\mathsf{cons}(x, xs), xs)$$

holds. Thus a source-carried head entry whose source classifier class is already represented in the finite spine is invisible to the induced $\mathsf{FinSet}^{\uparrow}$ *classifier.*

Proof. Unpack the displayed assumptions as $A(x)$, $A(y)$, an occurrence $\mathsf{SpineMember}(y, xs)$, and a classifier witness $x \sim_A y$. By Definition 112.11, choose a position $j : \mathsf{Pos}(xs)$ with $\mathsf{entry}_{xs}(j) = y$.

Unfold Definition 112.1. It is enough to fix a query history z and prove the two membership implications between $\mathsf{cons}(x, xs)$ and xs.

For the forward implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs))$. Then $A(z)$ holds and some position of the cons spine carries an entry classified with z. Case-analyze that position. If it is the head position, the readback is x, so the witness supplies $z \sim_A x$. The transitivity field of cert from Definition 9.7 composes $z \sim_A x$ with $x \sim_A y$ and gives $z \sim_A y$. Rewriting y as $\mathsf{entry}_{xs}(j)$, the position j together with $A(y)$ repacks as $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$. If the cons-position is inherited from the tail, its readback is already a position of xs with the same carried-entry and classifier witnesses, so it repacks directly as $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$.

For the reverse implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$. Keep its source witness $A(z)$ and its position $i : \mathsf{Pos}(xs)$ with $A(\mathsf{entry}_{xs}(i))$ and $z \sim_A \mathsf{entry}_{xs}(i)$. View i as the corresponding tail position of $\mathsf{cons}(x, xs)$. The cons readback at that tail position is $\mathsf{entry}_{xs}(i)$, so the same carried-entry and classifier witnesses repack as $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs))$.

The two implications hold for every query history z. By the classifier clause in Definition 112.1, this is exactly $\mathsf{FinSetEnumClassifier}_{A,\sim_A}(\mathsf{cons}(x, xs), xs)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationClassifier duplicate insert`

Lemma 112.13 (Certified finite-position bijections compose)**.** *For finite BEDC-history spines xs, ys, and zs, certified finite-position bijections*

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys) \qquad \textit{and} \qquad \rho : \mathsf{Pos}(ys) \cong \mathsf{Pos}(zs)$$

compose to a certified finite-position bijection

$$\rho \circ \pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(zs),$$

whose inverse position map is $\pi^{-1} \circ \rho^{-1}$.

Proof. The certified-bijection data in Definition 112.2 consist of a forward position map, an inverse position map, and the two inverse identities. For the composite, take the forward position map $\rho \circ \pi$ and the inverse position map $\pi^{-1} \circ \rho^{-1}$. The left and right inverse identities follow by applying first the inverse identities of ρ and then the inverse identities of π, in the corresponding order. Thus the composite map carries the same certified-bijection data required in Definition 112.2. □

Theorem 112.14 (EnumPerm transitivity by bijection composition)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^{\uparrow}$ certificate, let $\sim_A$ be its classifier, and let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. For finite BEDC-history spines xs, ys, and zs, if*

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, ys) \qquad \textit{and} \qquad \mathsf{EnumPerm}_{A,\sim_A}(ys, zs)$$

hold, then

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, zs)$$

holds.

Proof. Unfold Definition 112.2. Choose from the first permutation a certified bijection $\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$ and a pointwise preservation field

$$P_\pi(i) : A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{ys}(\pi(i))) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

Choose from the second permutation a certified bijection $\rho : \mathsf{Pos}(ys) \cong \mathsf{Pos}(zs)$ and a pointwise preservation field

$$P_\rho(j) : A(\mathsf{entry}_{ys}(j)) \wedge A(\mathsf{entry}_{zs}(\rho(j))) \wedge \mathsf{entry}_{ys}(j) \sim_A \mathsf{entry}_{zs}(\rho(j)).$$

By Definition 112.13, $\rho \circ \pi$ is a certified bijection from $\mathsf{Pos}(xs)$ to $\mathsf{Pos}(zs)$.

It remains to verify the pointwise classifier-preservation field for the composite. Fix $i : \mathsf{Pos}(xs)$. The field $P_\pi(i)$ gives

$$A(\mathsf{entry}_{xs}(i)) \qquad \text{and} \qquad \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

The field $P_\rho(\pi(i))$ gives

$$A(\mathsf{entry}_{zs}(\rho(\pi(i)))) \qquad \text{and} \qquad \mathsf{entry}_{ys}(\pi(i)) \sim_A \mathsf{entry}_{zs}(\rho(\pi(i))).$$

The transitivity field of cert, equivalently Definition 9.8, composes the two classifier witnesses and yields

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{zs}(\rho(\pi(i))).$$

Together with the displayed carrier witnesses, this is exactly the pointwise preservation clause for the composite certified bijection. Repacking the composite bijection and this field gives $\mathsf{EnumPerm}_{A,\sim_A}(xs, zs)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationPermutation trans`

Lemma 112.15 (Pointwise enumeration membership equivalences compose)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, and let xs, ys, and zs be finite BEDC-history spines. If, for every query history q,*

$$\mathsf{EnumMem}_{A,\sim_A}(q, xs) \iff \mathsf{EnumMem}_{A,\sim_A}(q, ys)$$

and

$$\mathsf{EnumMem}_{A,\sim_A}(q, ys) \iff \mathsf{EnumMem}_{A,\sim_A}(q, zs)$$

hold, then, for every query history q,

$$\mathsf{EnumMem}_{A,\sim_A}(q, xs) \iff \mathsf{EnumMem}_{A,\sim_A}(q, zs).$$

Proof. Fix a query history q. For the forward implication, assume $\mathsf{EnumMem}_{A,\sim_A}(q, xs)$. The forward half of the first displayed equivalence gives $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$, and the forward half of the second displayed equivalence gives $\mathsf{EnumMem}_{A,\sim_A}(q, zs)$.

For the reverse implication, assume $\mathsf{EnumMem}_{A,\sim_A}(q, zs)$. The reverse half of the second displayed equivalence gives $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$, and the reverse half of the first displayed equivalence gives $\mathsf{EnumMem}_{A,\sim_A}(q, xs)$. Thus the two enumeration-membership predicates are equivalent at q. Since q was arbitrary, the pointwise equivalence holds for all query histories. □

Theorem 112.16 (FinSet enumeration classifier transitivity)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^\uparrow$ certificate, let $\sim_A$ be its classifier, and let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. For finite BEDC-history spines xs, ys, and zs, if*

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, ys) \qquad \textit{and} \qquad \mathsf{FinSetEnumClassifier}_{A,\sim_A}(ys, zs)$$

hold, then

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, zs)$$

holds.

Proof. The certificate fixes the ambient source and classifier. Unfold the classifier clause in Definition 112.1. The first hypothesis supplies, for every query history q, the equivalence between $\mathsf{EnumMem}_{A,\sim_A}(q, xs)$ and $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$. The second supplies, for the same q, the equivalence between $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$ and $\mathsf{EnumMem}_{A,\sim_A}(q, zs)$.

By Definition 112.15, these two pointwise equivalences compose to

$$\forall q : \mathsf{BHist}.\ \mathsf{EnumMem}_{A,\sim_A}(q, xs) \Longleftrightarrow \mathsf{EnumMem}_{A,\sim_A}(q, zs).$$

Applying the same classifier clause in Definition 112.1 repacks this pointwise statement as $\mathsf{FinSetEnumClassifier}_{A,\sim_A}($ □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationClassifier trans`

Definition 112.17 (Finite-spine predicate enumeration)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate with classifier* $\sim_A$*. For a finite BEDC-history spine* $xs : \mathsf{HistSpine}$ *and a predicate* $U : \mathsf{BHist} \to \mathsf{Prop}$*, define*

$$\mathsf{FinSetSpineEnumerates}_{A,\sim_A}(xs, U) \Longleftrightarrow \forall z : \mathsf{BHist}.\ \mathsf{EnumMem}_{A,\sim_A}(z, xs) \Longleftrightarrow U(z).$$

Lemma 112.18 (Enumeration membership splits over spine append)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate with classifier* $\sim_A$*. For finite BEDC-history spines* $xs, ys : \mathsf{HistSpine}$ *and every query history* $z : \mathsf{BHist}$*,*

$$\mathsf{EnumMem}_{A,\sim_A}(z, xs{+}{+}ys) \Longleftrightarrow \mathsf{EnumMem}_{A,\sim_A}(z, xs) \vee \mathsf{EnumMem}_{A,\sim_A}(z, ys).$$

Proof. Proceed by induction on the left spine xs, using the finite-spine constructors from Definition 36.57 and the position readback clause of Definition 112.1.

If $xs = \mathsf{nil}$, then $xs{+}{+}ys = ys$ and $\mathsf{Pos}(\mathsf{nil})$ has no positions. Thus $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{nil})$ has no position witness, while $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{nil}{+}{+}ys)$ is exactly $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$. The required equivalence follows.

Now write $xs = \mathsf{cons}(x, xs_0)$ and assume the statement for xs_0 and ys. Then $xs{+}{+}ys = \mathsf{cons}(x, xs_0{+}{+}ys)$. A position of this cons spine is either the head position, whose readback is x, or a tail position coming from $xs_0{+}{+}ys$.

For the forward implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs_0{+}{+}ys))$. Keep the displayed witness $A(z)$ and case-analyze its position. In the head case, the same head position of $\mathsf{cons}(x, xs_0)$ carries $A(x)$ and the same classifier witness $z \sim_A x$, so $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs_0))$ holds. In the tail case, the witness is a position of $xs_0{+}{+}ys$ with unchanged readback. The induction hypothesis gives either $\mathsf{EnumMem}_{A,\sim_A}(z, xs_0)$ or $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$. The first alternative embeds through the tail position of $\mathsf{cons}(x, xs_0)$, giving the left disjunct; the second alternative is already the right disjunct.

For the reverse implication, suppose $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs_0)) \vee \mathsf{EnumMem}_{A,\sim_A}(z, ys)$. If the left disjunct holds, case-analyze its cons position. A head witness repacks at the head of $\mathsf{cons}(x, xs_0{+}{+}ys)$. A tail witness gives $\mathsf{EnumMem}_{A,\sim_A}(z, xs_0)$; by the reverse half of the induction hypothesis it yields $\mathsf{EnumMem}_{A,\sim_A}(z, xs_0{+}{+}ys)$, which then embeds through the tail position of $\mathsf{cons}(x, xs_0{+}{+}ys)$. If the right disjunct holds, the reverse half of the induction hypothesis gives $\mathsf{EnumMem}_{A,\sim_A}(z, xs_0{+}{+}ys)$, and the same tail-position embedding gives $\mathsf{EnumMem}_{A,\sim_A}(z, \mathsf{cons}(x, xs_0{+}{+}ys))$.

The two implications prove the displayed equivalence for the cons spine, completing the induction. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier bundleAppend split`
Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier append split`

Theorem 112.19 (Finite-spine concatenation enumerates finite-set union)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate with classifier* $\sim_A$*. Let* $xs, ys : \mathsf{HistSpine}$ *be finite BEDC-history spines, and let* $U, V : \mathsf{BHist} \to \mathsf{Prop}$ *be predicates. If*

$$\mathsf{FinSetSpineEnumerates}_{A,\sim_A}(xs, U) \qquad \textit{and} \qquad \mathsf{FinSetSpineEnumerates}_{A,\sim_A}(ys, V)$$

hold, then the concatenated finite spine enumerates the finite-set union predicate:

$$\mathsf{FinSetSpineEnumerates}_{A,\sim_A}(xs{+}{+}ys, \lambda z.\, U(z) \vee V(z)).$$

Proof. Unfold Definition 112.17. Fix a query history $z : \mathsf{BHist}$. By Definition 112.18,

$$\mathsf{EnumMem}_{A,\sim_A}(z, xs{+}{+}ys) \iff \mathsf{EnumMem}_{A,\sim_A}(z, xs) \vee \mathsf{EnumMem}_{A,\sim_A}(z, ys).$$

The first enumeration hypothesis converts $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$ exactly into $U(z)$, and the second converts $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$ exactly into $V(z)$.

For the forward implication, assume $\mathsf{EnumMem}_{A,\sim_A}(z, xs{+}{+}ys)$. The append split gives membership in xs or membership in ys. Applying the corresponding enumeration hypothesis yields $U(z)$ or $V(z)$.

For the reverse implication, assume $U(z) \vee V(z)$. In the $U(z)$ case, the reverse half of the first enumeration hypothesis gives $\mathsf{EnumMem}_{A,\sim_A}(z, xs)$, and the reverse half of the append split gives membership in $xs{+}{+}ys$. In the $V(z)$ case, the reverse half of the second enumeration hypothesis gives $\mathsf{EnumMem}_{A,\sim_A}(z, ys)$, and the reverse half of the append split again gives membership in $xs{+}{+}ys$.

Thus for every z the appended enumeration predicate is equivalent to $U(z) \vee V(z)$, which is exactly the displayed enumeration claim. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationCarrier bundleAppend union`

Lemma 112.20 (Certified finite-position identity bijection)**.** *For every finite BEDC-history spine* $xs : \mathsf{HistSpine}$*, the identity map on* $\mathsf{Pos}(xs)$ *is a certified finite-position bijection*

$$\mathsf{id}_{xs} : \mathsf{Pos}(xs) \cong \mathsf{Pos}(xs),$$

with inverse map again the identity map.

Proof. Take the forward position map to be $i \mapsto i$ and take the inverse position map to be the same function. The two inverse identities are the pointwise equalities $i = i$ and $j = j$ for positions of xs. These data are exactly the certified finite-position bijection data used in Definition 112.2. □

Theorem 112.21 (EnumPerm identity closure)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^{\uparrow}$ *certificate, let* $\sim_A$ *be its classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. For every finite BEDC-history spine* $xs : \mathsf{HistSpine}$*, if every entry of* xs *is carried by* A*,*

$$\forall i : \mathsf{Pos}(xs).\ A(\mathsf{entry}_{xs}(i)),$$

then

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, xs)$$

holds.

Proof. Use the certified identity bijection from Definition 112.20. It remains only to verify the pointwise preservation field required by Definition 112.2.

Fix a position $i : \mathsf{Pos}(xs)$. The carried-entry hypothesis gives

$$A(\mathsf{entry}_{xs}(i)).$$

For the identity bijection, the matched source position and target position are both i, so the source and target entries in the matched pair are the same history $\mathsf{entry}_{xs}(i)$. The reflexivity field of cert, supplied by the naming certificate in Definition 9.7, applies to the displayed carrier witness and gives

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{xs}(i).$$

Thus the matched pair at i satisfies

$$A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{xs}(i)) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{xs}(i).$$

Since i was arbitrary, the certified identity bijection carries the required pointwise classifier-preservation field. Repacking these data by Definition 112.2 proves $\mathsf{EnumPerm}_{A,\sim_A}(xs, xs)$. □

Lemma 112.22 (Pointwise enumeration membership equivalences invert)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be a BEDC history source carrier, let $\sim_A$ be a source classifier, and let xs and ys be finite BEDC-history spines. If, for every query history q,*

$$\mathsf{EnumMem}_{A,\sim_A}(q, xs) \iff \mathsf{EnumMem}_{A,\sim_A}(q, ys)$$

holds, then, for every query history q,

$$\mathsf{EnumMem}_{A,\sim_A}(q, ys) \iff \mathsf{EnumMem}_{A,\sim_A}(q, xs)$$

holds.

Proof. Fix a query history q. The displayed hypothesis at q supplies both membership implications between the two finite-spine enumeration predicates. For the forward implication in the inverted direction, assume $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$. The reverse half of the hypothesis at q gives $\mathsf{EnumMem}_{A,\sim_A}(q, xs)$. For the reverse implication in the inverted direction, assume $\mathsf{EnumMem}_{A,\sim_A}(q, xs)$. The forward half of the hypothesis at q gives $\mathsf{EnumMem}_{A,\sim_A}(q, ys)$. Since q was arbitrary, the inverted pointwise equivalence holds for all query histories. □

Theorem 112.23 (FinSet enumeration classifier symmetry)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^{\uparrow}$ certificate, let $\sim_A$ be its classifier, and let xs and ys be finite BEDC-history spines. If*

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(xs, ys)$$

holds, then

$$\mathsf{FinSetEnumClassifier}_{A,\sim_A}(ys, xs)$$

holds.

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationClassifier symm`

Proof. Unfold the classifier clause in Definition 112.1. The hypothesis supplies, for every query history q, the pointwise equivalence

$$\mathsf{EnumMem}_{A,\sim_A}(q, xs) \iff \mathsf{EnumMem}_{A,\sim_A}(q, ys).$$

By Definition 112.22, this equivalence inverts pointwise to

$$\forall q : \mathsf{BHist}.\ \mathsf{EnumMem}_{A,\sim_A}(q, ys) \iff \mathsf{EnumMem}_{A,\sim_A}(q, xs).$$

Applying the same classifier clause in Definition 112.1 repacks this pointwise statement as $\mathsf{FinSetEnumClassifier}_{A,\sim_A}($ □

Lemma 112.24 (Certified finite-position bijections invert)**.** *For finite BEDC-history spines xs and ys, any certified finite-position bijection*

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$$

has an inverse certified finite-position bijection

$$\pi^{-1} : \mathsf{Pos}(ys) \cong \mathsf{Pos}(xs),$$

whose inverse position map is π.

Proof. By the certified-bijection clause recalled in Definition 112.2, the datum for π consists of the forward position map π, the inverse position map π^{-1}, and the two inverse identities

$$\pi^{-1}(\pi(i)) = i \qquad \text{and} \qquad \pi(\pi^{-1}(j)) = j.$$

For the reversed bijection, take forward map π^{-1} and inverse map π. Its two identities are exactly the same two identities with the source and target position carriers exchanged. □

Theorem 112.25 (EnumPerm inverse symmetry)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^{\uparrow}$ certificate, let $\sim_A$ be its classifier, and let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. For finite BEDC-history spines xs and ys, if*

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, ys)$$

holds, then

$$\mathsf{EnumPerm}_{A,\sim_A}(ys, xs)$$

holds.

Proof. Unfold Definition 112.2. Choose the certified bijection

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$$

and its pointwise preservation field P_π. By Definition 112.24, π^{-1} is a certified bijection from $\mathsf{Pos}(ys)$ to $\mathsf{Pos}(xs)$.

It remains to verify the pointwise preservation field for this inverse bijection. Fix $j : \mathsf{Pos}(ys)$ and put $i = \pi^{-1}(j)$. The inverse identity gives $\pi(i) = j$. The field $P_\pi(i)$ gives

$$A(\mathsf{entry}_{xs}(i)) \qquad \text{and} \qquad A(\mathsf{entry}_{ys}(\pi(i)))$$

and also

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

Using $\pi(i) = j$, these become the carrier witnesses $A(\mathsf{entry}_{xs}(i))$ and $A(\mathsf{entry}_{ys}(j))$, together with the classifier comparison

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(j).$$

The symmetry field of cert, supplied by Definition 9.7, reverses this comparison to

$$\mathsf{entry}_{ys}(j) \sim_A \mathsf{entry}_{xs}(i).$$

Thus j satisfies the pointwise carrier-and-classifier clause required for the inverse map π^{-1}. Since j was arbitrary, the inverse certified bijection and this pointwise field repack as $\mathsf{EnumPerm}_{A,\sim_A}(ys, xs)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationPermutation symm`

Theorem 112.26 (EnumPerm identity reflexivity)**.** *Let $A : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a fixed $\mathsf{Set}^{\uparrow}$ certificate, let $\sim_A$ be its classifier, and let $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$. For every finite BEDC-history spine $xs : \mathsf{HistSpine}$, if every position entry is carried by A,*

$$\forall i : \mathsf{Pos}(xs).\ A(\mathsf{entry}_{xs}(i)),$$

then

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, xs)$$

holds, witnessed by the identity bijection on $\mathsf{Pos}(xs)$.

Proof. Unfold Definition 112.2. Supply the certified finite-position bijection whose forward map and inverse map are both the identity function on $\mathsf{Pos}(xs)$. The two inverse identities are the pointwise equalities $i = i$ and $j = j$, so this is a certified bijection from $\mathsf{Pos}(xs)$ to itself.

It remains to verify the pointwise carrier-and-classifier field. Fix a position $i : \mathsf{Pos}(xs)$. The hypothesis gives

$$A(\mathsf{entry}_{xs}(i)).$$

Under the identity position map, the matched source and target entries are both $\mathsf{entry}_{xs}(i)$. The reflexivity field of cert, supplied by Definition 9.7, gives

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{xs}(i).$$

Therefore the matched pair at i satisfies

$$A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{xs}(i)) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{xs}(i).$$

Since i was arbitrary, the identity certified bijection has the required pointwise preservation field. Repacking these data proves $\mathsf{EnumPerm}_{A,\sim_A}(xs, xs)$. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationPermutation refl`

Definition 112.27 (FinSet NameCert obligation packet)**.** *The finite-set certificate obligation packet for $\mathsf{FinSet}^{\uparrow}$ consists of the following ten fields:*

1. *a traditional finite subset predicate $U : \mathsf{BHist} \to \mathsf{Prop}$;*
2. *a BEDC source certificate $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ with carrier A and classifier $\sim_A$;*
3. *a $\mathsf{NameCert}_{\mathsf{List}^{\uparrow}}$ finite spine $xs : \mathsf{HistSpine}$ satisfying $\mathsf{ListSpineRep}_A(e, xs)$;*

4. *the carrier* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)$ *from Definition 112.5;*
5. *the classifier* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ *from Definition 112.1;*
6. *the stability rows for certified enumeration permutation, duplicate insertion, classifier transitivity, classifier symmetry, and append-union concatenation;*
7. *the finite-position ledger* $\mathsf{Pos}(xs)$ *with readback* entry_{xs} *and certified bijections between position carriers;*
8. *the exactness predicate comparing* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)(a)$ *with finite-spine occurrence modulo* $\sim_A$*;*
9. *the standard finite-set bridge that reads the enumerated predicate* U *through* $\mathsf{FinSetSpineEnumerates}_{A,\sim_A}(xs, U)$*;*
10. *the status assertion that the listed fields are obligations for* $\mathsf{NameCert}_{\mathsf{FinSet}\uparrow}$ *rather than a closed public finite-set API.*

Theorem 112.28 (FinSet carrier obligation)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the source carrier of a fixed* $\mathsf{NameCert}_{\mathsf{Set}\uparrow}$*, let* $\sim_A$ *be its classifier, let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*, and let* e *and* xs *satisfy* $\mathsf{ListSpineRep}_A(e, xs)$*. Then the carrier field of Definition 112.27 is exact: for every history* a*, membership in* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}(xs)$ *is precisely finite-spine occurrence in* xs *modulo* $\sim_A$*.*

Proof. The carrier field of the packet is definitionally the carrier in Definition 112.5. The required exactness statement is exactly the bidirectional membership statement of Definition 112.7, with the displayed $\mathsf{ListSpineRep}_A(e, xs)$ witness supplying the finite-spine source condition.

Thus the packet records no additional carrier structure beyond the source certificate, the finite spine, and the exactness row already proved for $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$. □

Theorem 112.29 (FinSet classifier obligation rows)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the source carrier of a fixed* $\mathsf{NameCert}_{\mathsf{Set}\uparrow}$*, let* $\sim_A$ *be its classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. The classifier field of Definition 112.27 carries the finite-set obligation rows for certified enumeration permutation, duplicate insertion, append-union enumeration, and equivalence closure.*

Proof. The permutation row is Definition 112.3: a certified finite-position bijection preserving A and $\sim_A$ pointwise leaves $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ unchanged.

The duplicate row is Definition 112.12: a source-carried head entry already represented in the spine is invisible to the classifier. The append-union row is Definition 112.19: concatenating finite spines enumerates the union predicate.

The equivalence rows are Definition 112.16 and Definition 112.23. Together with the reflexive identity row proved above, these rows give the classifier obligation surface needed by the packet. □

Theorem 112.30 (FinSet stability exactness ledger and bridge obligations)**.** *For any packet as in Definition 112.27, the displayed local rows jointly determine the finite-set stability, exactness, ledger, and bridge obligations over the BEDC source* $\mathsf{NameCert}_{\mathsf{Set}\uparrow}$ *and the finite-spine source* $\mathsf{NameCert}_{\mathsf{List}\uparrow}$*.*

Proof. First, the exactness obligation is the carrier equivalence supplied by Definition 112.28, whose proof rests on Definition 112.7. This binds ordinary finite-spine occurrence to the BEDC carrier field through the fixed source classifier.

Next, the stability obligations are the classifier rows collected in Definition 112.29: certified permutation, duplicate insertion, transitivity, symmetry, and append-union concatenation. These cite Definition 112.3, Definition 112.12, Definition 112.16, Definition 112.23, and Definition 112.19.

Finally, the ledger obligation is the finite position carrier $\mathsf{Pos}(xs)$ with readback entry_{xs}, used by the same permutation and occurrence rows. The bridge obligation is the predicate enumeration clause $\mathsf{FinSetSpineEnumerates}_{A,\sim_A}(xs, U)$ together with the append-union row, which identifies the standard finite-set surface without asserting the larger finite-set API. □

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumerationBridge bundle member readback`

Closure status for $\mathsf{NameCert}_{\mathsf{FinSet}^\uparrow}$.

Constructive story: A finite set is generated by taking a $\mathsf{Set}^\uparrow$ carrier A and displaying a finite $\mathsf{List}^\uparrow$ spine xs whose entries are BEDC histories carried by A. A query history z belongs to the finite-set carrier exactly when $A(z)$ holds and some finite position of xs has an entry classified with z by the set classifier; histories outside the source carrier or absent from the spine up to that classifier are excluded. Two finite-spine presentations classify together when they have the same enumeration membership predicate for every query history, so certified permutations of positions and duplicate insertions are invisible while the spine and its positions remain explicit ledger data. The source is the fixed $\mathsf{Set}^\uparrow$ certificate plus the finite spine, the pattern records occurrence, position readback, permutation invariance, duplicate insertion, and append-union rows, and the ledger keeps $\mathsf{Pos}(xs)$, entries, enumeration witnesses, and classifier evidence visible enough for downstream consumers. The five name-certificate obligations follow from the displayed finite spine and source certificate for inhabitation, from pointwise reflexivity, symmetry, and transitivity of enumeration membership equivalence, and from carrier transport through the underlying set classifier. A spine xs lies in the signature gap of a $\mathsf{FinSet}^\uparrow$ token p when the source-domain accepts the set carrier and finite-list representation, the signature relation pairs the enumeration with an answer spine and evidence ledger, and token introduction names that answer as p; unfolded, exactly the classifier-classes enumerated by xs are those encoded by the token. The standard bridge is the finite-spine enumeration bridge: it reads a finite set as the predicate enumerated by the spine and reads back through the same positions, entries, duplicate, permutation, and union rows. The chapter registers this finite-spine construction under the name $\mathsf{FinSet}^\uparrow$; finiteness is an explicit enumeration ledger, not cardinality theory, power sets, host finite sets, or hidden order data promoted to the carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinSet}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The FinSetUp bridge scope is the finite-spine enumeration bridge over $\mathsf{NameCert}_{\mathsf{Set}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{List}^\uparrow}$, binding $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$, $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$, EnumPerm, the Pos and entry ledger, duplicate, permutation, symmetry, transitivity, append-union rows, and $\mathsf{FinSetSpineEnumerates}_{A,\sim_A}$ as the standard finite-set bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinSet}^\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FinsetUp.FinsetEnumeration\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: Cardinality, decidable membership, power set, graph finiteness, and Lean-side bridge verification remain outside this finite-spine bridge.

Upgrade path: There is no stronger paper-side closure grade for this finite-spine bridge; remaining work is verification-axis bridge checking and optional APIs outside the stated surface.

Theorem 112.31 (EnumPerm composition associativity)**.** *Let* $A : \mathsf{BHist} \to \mathsf{Prop}$ *be the carrier of a fixed* $\mathsf{Set}^\uparrow$ *certificate, let* $\sim_A$ *be its classifier, and let* $\mathsf{cert} : \mathsf{NameCert}(A, \sim_A)$*. For finite BEDC-history spines* xs*,* ys*,* zs*, and* ws*, assume*

$$\mathsf{EnumPerm}_{A,\sim_A}(xs, ys) \quad \textit{and} \quad \mathsf{EnumPerm}_{A,\sim_A}(ys, zs) \quad \textit{and} \quad \mathsf{EnumPerm}_{A,\sim_A}(zs, ws).$$

After choosing certified bijections from these three rows, the two parenthesized composite witnesses from xs *to* ws *obtained from*

$$(\sigma \circ \rho) \circ \pi$$

and from

$$\sigma \circ (\rho \circ \pi)$$

coincide at the level of the EnumPerm *witness data: their forward position maps agree pointwise, their inverse position maps agree pointwise, and one and the same pointwise carrier-classifier clause certifies both bracketings. In particular,*

$$((\sigma \circ \rho) \circ \pi)(i) = (\sigma \circ (\rho \circ \pi))(i)$$

for every $i : \mathsf{Pos}(xs)$*, and*

$$(\pi^{-1} \circ (\rho^{-1} \circ \sigma^{-1}))(k) = ((\pi^{-1} \circ \rho^{-1}) \circ \sigma^{-1})(k)$$

for every $k : \mathsf{Pos}(ws)$*.*

Proof. Unfold Definition 112.2. Choose a certified bijection

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$$

with pointwise preservation field

$$P_\pi(i) : A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{ys}(\pi(i))) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

Choose a certified bijection

$$\rho : \mathsf{Pos}(ys) \cong \mathsf{Pos}(zs)$$

with pointwise preservation field

$$P_\rho(j) : A(\mathsf{entry}_{ys}(j)) \wedge A(\mathsf{entry}_{zs}(\rho(j))) \wedge \mathsf{entry}_{ys}(j) \sim_A \mathsf{entry}_{zs}(\rho(j)).$$

Choose a certified bijection

$$\sigma : \mathsf{Pos}(zs) \cong \mathsf{Pos}(ws)$$

with pointwise preservation field

$$P_\sigma(k) : A(\mathsf{entry}_{zs}(k)) \wedge A(\mathsf{entry}_{ws}(\sigma(k))) \wedge \mathsf{entry}_{zs}(k) \sim_A \mathsf{entry}_{ws}(\sigma(k)).$$

By two applications of Definition 112.13, the left bracketing first gives the certified bijection

$$\sigma \circ \rho : \mathsf{Pos}(ys) \cong \mathsf{Pos}(ws)$$

with inverse position map $\rho^{-1} \circ \sigma^{-1}$, and then gives the certified bijection

$$(\sigma \circ \rho) \circ \pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ws)$$

with inverse position map $\pi^{-1} \circ (\rho^{-1} \circ \sigma^{-1})$. The right bracketing first gives the certified bijection

$$\rho \circ \pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(zs)$$

with inverse position map $\pi^{-1} \circ \rho^{-1}$, and then gives the certified bijection

$$\sigma \circ (\rho \circ \pi) : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ws)$$

with inverse position map $(\pi^{-1} \circ \rho^{-1}) \circ \sigma^{-1}$.

For every $i : \mathsf{Pos}(xs)$,

$$((\sigma \circ \rho) \circ \pi)(i) = (\sigma \circ \rho)(\pi(i)) = \sigma(\rho(\pi(i))) = \sigma((\rho \circ \pi)(i)) = (\sigma \circ (\rho \circ \pi))(i).$$

Thus the two forward position maps agree pointwise. Likewise, for every $k : \mathsf{Pos}(ws)$,

$$(\pi^{-1} \circ (\rho^{-1} \circ \sigma^{-1}))(k) = \pi^{-1}(\rho^{-1}(\sigma^{-1}(k))) = (\pi^{-1} \circ \rho^{-1})(\sigma^{-1}(k)) = ((\pi^{-1} \circ \rho^{-1}) \circ \sigma^{-1})(k).$$

Thus the two inverse position maps also agree pointwise.

It remains to verify a common pointwise carrier-classifier clause for the two bracketings. Fix $i : \mathsf{Pos}(xs)$. From $P_\pi(i)$ we obtain

$$A(\mathsf{entry}_{xs}(i)) \qquad \text{and} \qquad \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ys}(\pi(i)).$$

From $P_\rho(\pi(i))$ we obtain

$$A(\mathsf{entry}_{zs}(\rho(\pi(i)))) \qquad \text{and} \qquad \mathsf{entry}_{ys}(\pi(i)) \sim_A \mathsf{entry}_{zs}(\rho(\pi(i))).$$

From $P_\sigma(\rho(\pi(i)))$ we obtain

$$A(\mathsf{entry}_{ws}(\sigma(\rho(\pi(i))))) \qquad \text{and} \qquad \mathsf{entry}_{zs}(\rho(\pi(i))) \sim_A \mathsf{entry}_{ws}(\sigma(\rho(\pi(i)))).$$

Applying the classifier transitivity of cert, equivalently Definition 9.8, first to the comparisons from xs to ys and from ys to zs, and then to the resulting comparison from xs to zs and the comparison from zs to ws, yields

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ws}(\sigma(\rho(\pi(i)))).$$

Therefore

$$A(\mathsf{entry}_{xs}(i)) \wedge A(\mathsf{entry}_{ws}(\sigma(\rho(\pi(i))))) \wedge \mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{ws}(\sigma(\rho(\pi(i)))).$$

By the forward-map equality already proved, this is exactly the pointwise preservation clause required for both $((\sigma \circ \rho) \circ \pi)(i)$ and $\big(\sigma \circ (\rho \circ \pi)\big)(i)$.

Hence the two bracketings have the same forward position map, the same inverse position map, and the same pointwise carrier-classifier certification. They therefore present the same $\mathsf{EnumPerm}_{A,\sim_A}(xs, ws)$ witness data required by Definition 112.2. □

Theorem 112.32 (FinSet enumeration carrier boundary)**.** *Fix a* $\mathsf{Set}^\uparrow$ *carrier* $A : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_A$*, and let* $xs : \mathsf{HistSpine}$ *be the finite spine supplied by the* $\mathsf{List}^\uparrow$ *enumeration witness. The public* $\mathsf{FinSet}^\uparrow$ *carrier exported by Definition 112.5 consists exactly of a carried query history, a* $\mathsf{SpineMember}$ *witness in the displayed finite spine, and a classifier witness against one displayed spine entry. It exports no host set, no duplicate-normalized quotient, and no hidden enumeration order.*

Lean4 ✓ `BEDC.Derived.FinsetUp.FinsetEnumeration StdBridge`

Proof. Unfold the carrier in Definition 112.5. A public membership row is a pair consisting of $A(a)$ and a displayed witness

$$\exists x : \mathsf{BHist}.\ \mathsf{SpineMember}(x, xs) \wedge a \sim_A x.$$

Thus the only exported data are the carried query history, the finite-spine occurrence, and the classifier comparison to the displayed entry.

The occurrence entry is source-carried by Definition 112.6, so the witness really lives in the fixed $\mathsf{Set}^\uparrow$ source rather than in a host set. The equivalence of Definition 112.7 says that this displayed occurrence-and-classifier surface is not merely sufficient but exact for the carrier.

No further public field remains. Duplicate handling is mediated by the classifier and by later enumeration theorems, while enumeration order remains the visible $\mathsf{List}^\uparrow$ spine data rather than a hidden quotient. Hence the public carrier boundary is exactly the stated finite-data surface. □

$\mathsf{FinSet}^\uparrow$ reads finite membership through the $\mathsf{Bool}^\uparrow$ classifier of chapter 32 and finite enumeration through the $\mathsf{List}^\uparrow$ spine of chapter 36. Thus the finite-set packet exposes its sibling dependency route before any consumer treats membership or enumeration as local kernel-only data.

Chapter 113

A Concrete Naming Certificate for $\mathsf{Subtype}^{\uparrow}$

The subtype interface $\mathsf{Subtype}^{\uparrow}$ packages the carrier of elements of a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ satisfying a decidable predicate. It depends on $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ (chapter 111) and supplies the kernel-licit refinement type used throughout the algebra ladder for sub-objects.

113.1 Carrier definition

113.2 The certificate

Definition 113.1 (Subtype restricted classifier)**.** *Let S be the certified point type of a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$, with parent classifier $\sim_S$. Let $P : S \to \mathsf{Prop}$ be a decidable predicate stable under the parent classifier, meaning that for $x, y : S$,*

$$x \sim_S y \Longrightarrow P(x) \Longrightarrow P(y).$$

The $\mathsf{Subtype}^{\uparrow}$ carrier cut out by P is

$$S_P = \{(x, p) \mid x : S \text{ and } p : P(x)\}.$$

For subtype elements (x, p_x) and (y, p_y), the subtype classifier is the restriction of the parent classifier:

$$(x, p_x) \sim_{S_P} (y, p_y) \quad :\Longleftrightarrow \quad x \sim_S y.$$

Lean4 ✓ `BEDC.Derived.SubtypeUp.SubtypeTasteGate single carrier alignment`

Theorem 113.2 (Subtype classifier reflects parent classifier)**.** *Let S be a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ with classifier $\sim_S$, and let $P : S \to \mathsf{Prop}$ be a decidable predicate stable under $\sim_S$. For all $x, y : S$ and witnesses $p_x : P(x)$ and $p_y : P(y)$, the classifier of Definition 113.1 satisfies*

$$(x, p_x) \sim_{S_P} (y, p_y) \Longleftrightarrow x \sim_S y.$$

Equivalently, a parent classifier witness between predicate-bearing endpoints induces the subtype classifier witness, and every subtype classifier witness projects to the parent classifier witness.

Proof. Unfold the classifier clause in Definition 113.1. The forward implication is projection: a subtype classifier witness is a parent classifier witness between the first components. The reverse implication is inclusion: a witness of $x \sim_S y$ is, by the displayed definition, a witness of $(x, p_x) \sim_{S_P} (y, p_y)$ once the two subtype endpoints have been formed. The decidability and stability assumptions are the formation and transport rows for the $\mathsf{Subtype}^{\uparrow}$ carrier; the restricted classifier itself carries no data beyond the parent comparison. □

Theorem 113.3 (Subtype carrier obligation row)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$, and let $P : \mathsf{BHist} \to \mathsf{Prop}$ be a decidable predicate. In the predicate presentation of Definition 113.1, the carrier obligation for $\mathsf{Subtype}^{\uparrow}$ is exactly*

$$S_P(h) \iff S(h) \wedge P(h).$$

Consequently a displayed endpoint $s_0 : \mathsf{BHist}$ with $S(s_0)$ and $P(s_0)$ supplies the inhabitedness row

$$S_P(s_0).$$

Proof. Unfold the carrier clause of Definition 113.1. A subtype endpoint is formed by pairing a parent-carrier witness with a predicate witness, so its predicate presentation is the conjunction $S(h) \wedge P(h)$.

For inhabitedness, apply the same clause at s_0. The two displayed witnesses $S(s_0)$ and $P(s_0)$ are precisely the two components required to form $S_P(s_0)$. □

Theorem 113.4 (Subtype classifier projection obligation)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ with classifier $\sim_S$, and let $P : \mathsf{BHist} \to \mathsf{Prop}$. For all predicate-bearing endpoints $h, k : \mathsf{BHist}$, the classifier row for $\mathsf{Subtype}^{\uparrow}$ is exactly*

$$h \sim_{S_P} k \iff h \sim_S k.$$

Thus parent projection and restricted-classifier introduction are the two classifier rows: a subtype classifier witness projects to the parent comparison, and a parent comparison between already formed subtype endpoints introduces the restricted classifier witness.

Proof. Unfold the classifier clause in Definition 113.1. The restricted classifier stores no comparison datum apart from the parent comparison $h \sim_S k$.

The forward direction is projection from the restricted witness to that parent witness. The reverse direction is introduction: once the endpoints are already certified by the carrier row, the parent witness is read as the restricted-classifier witness. □

Theorem 113.5 (Subtype NameCert transport obligation)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a parent $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ with classifier $\sim_S$, and let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$. Let $P : \mathsf{BHist} \to \mathsf{Prop}$ be stable under the parent classifier, meaning that for all $h, k : \mathsf{BHist}$,*

$$S(h) \wedge P(h) \wedge h \sim_S k \Longrightarrow P(k).$$

In the predicate presentation of Definition 113.1, parent carrier transport from cert together with predicate stability gives

$$S_P(h) \to h \sim_{S_P} k \to S_P(k).$$

This is the transport row consumed by Definition 113.7.

Proof. Unfold $S_P(h)$ by Definition 113.3. This gives $S(h)$ and $P(h)$. Unfold the restricted classifier by Definition 113.4; the comparison is the parent witness $h \sim_S k$.

The carrier-respect field of the parent $\mathsf{NameCert}(S, \sim_S)$ transports $S(h)$ along $h \sim_S k$ and gives $S(k)$. Apply the displayed predicate-stability hypothesis to $S(h)$, $P(h)$, and the same parent comparison to obtain $P(k)$.

Pair $S(k)$ with $P(k)$ using the carrier row for $S_P(k)$. The resulting witness is the required transport row for the restricted certificate. □

Lemma 113.6 (Subtype carrier closed under restricted classifier)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a parent $\mathsf{NameCert}_{\mathsf{Set}^\uparrow}$ with classifier $\sim_S$, and let $\mathsf{cert} : \mathsf{NameCert}(S, \sim_S)$. Let $P : \mathsf{BHist} \to \mathsf{Prop}$ be a decidable predicate stable under the parent classifier, in the sense that for all $x, y : \mathsf{BHist}$,*

$$S(x) \wedge P(x) \wedge x \sim_S y \Longrightarrow P(y).$$

In the predicate presentation of Definition 113.1, write

$$S_P(h) \Longleftrightarrow S(h) \wedge P(h), \qquad h \sim_{S_P} k \Longleftrightarrow h \sim_S k.$$

If $S_P(h)$ and $h \sim_{S_P} k$, then $S_P(k)$.

Lean4 ✓ `BEDC.Derived.SubtypeUp.SubtypeRestrictedCarrier classifier closed`

Proof. Unfold $S_P(h)$ to obtain $S(h)$ and $P(h)$. Unfold the restricted classifier to read the comparison as $h \sim_S k$. The carrier-transport field of cert, one of the fields listed in Definition 9.7, transports $S(h)$ along this comparison and gives $S(k)$. Stability of P under the same parent comparison gives $P(k)$. Pairing $S(k)$ with $P(k)$ gives $S_P(k)$. □

Theorem 113.7 (Subtype restricted classifier NameCert)**.** *Under the hypotheses and notation of Definition 113.6, assume that the subtype carrier has a displayed endpoint, so there is an $s_0 : \mathsf{BHist}$ with $S(s_0)$ and $P(s_0)$. Then the $\mathsf{Subtype}^\uparrow$ carrier S_P with the restricted classifier $\sim_{S_P}$ forms a Lean carrier naming certificate:*

$$\mathsf{NameCert}(S_P, \sim_{S_P}).$$

Lean4 ✓ `BEDC.Derived.SubtypeUp.SubtypeRestrictedClassifier nameCert`

Proof. Inhabitedness is witnessed by s_0, since $S(s_0)$ and $P(s_0)$ are exactly the two components of $S_P(s_0)$.

For reflexivity, let h be carried by S_P. Then $S(h)$ holds, so the reflexivity field of cert gives $h \sim_S h$. By the classifier clause of Definition 113.1, this is $h \sim_{S_P} h$.

For symmetry, a restricted witness $h \sim_{S_P} k$ is the parent witness $h \sim_S k$. The symmetry field of cert gives $k \sim_S h$, hence $k \sim_{S_P} h$.

For transitivity, restricted witnesses $h \sim_{S_P} k$ and $k \sim_{S_P} r$ are parent witnesses $h \sim_S k$ and $k \sim_S r$. The transitivity field of cert composes them to $h \sim_S r$, which is the restricted witness $h \sim_{S_P} r$.

Carrier transport for S_P is exactly Definition 113.6. These five rows are the inhabitedness, reflexivity, symmetry, transitivity, and carrier-transport fields required by Definition 9.7. □

Theorem 113.8 (SubtypeUp carrier-source obligation)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be the carrier of a parent $\mathsf{NameCert}(S, \sim_S)$, let $P : \mathsf{BHist} \to \mathsf{Prop}$ be a predicate stable under the parent classifier, and let $s_0 : \mathsf{BHist}$ be a displayed endpoint with $S(s_0) \wedge P(s_0)$. The carrier row for $\mathsf{Subtype}^\uparrow$ is exactly*

$$S_P(h) \Longleftrightarrow S(h) \wedge P(h).$$

The source data are the parent certificate $\mathsf{NameCert}(S, \sim_S)$*, the predicate* P*, the displayed endpoint* s_0*, and the stability of* P *under* $\sim_S$*.*

Proof. Unfold Definition 113.1. The subtype carrier is introduced as the predicate cut of the parent carrier by P, so membership in S_P is precisely the pair consisting of parent membership in S and the predicate witness.

The endpoint row is supplied by s_0: its two witnesses are the two components needed for $S_P(s_0)$. Predicate stability is not extra carrier data; it is the transport hypothesis that makes the carrier row usable across the restricted classifier. □

Theorem 113.9 (SubtypeUp restricted-classifier obligation)**.** *Under the hypotheses of Definition 113.8, the restricted classifier row for carried subtype endpoints is exactly*

$$h \sim_{S_P} k \iff h \sim_S k.$$

The parent projection row sends a restricted classifier witness to the parent witness, and the inclusion row sends a parent witness between carried subtype endpoints to the restricted witness.

Proof. Apply Definition 113.2. Its forward direction is the parent projection row: a restricted comparison contains the parent comparison between the displayed endpoints.

Its reverse direction is the inclusion row: once the two endpoints are carried by S_P, a parent classifier witness is already the restricted classifier witness by the definition of the subtype classifier. □

Theorem 113.10 (SubtypeUp predicate-transport obligation)**.** *Under the hypotheses of Definition 113.8, parent carrier transport together with stability of* P *transports the subtype carrier across the restricted classifier: if* $S_P(h)$ *and* $h \sim_{S_P} k$*, then* $S_P(k)$*.*

Proof. Apply Definition 113.6. That lemma separates the two transported components: the parent certificate transports $S(h)$ to $S(k)$ along the parent classifier, and stability of P transports the predicate witness.

By Definition 113.9, the restricted comparison supplies exactly the parent comparison needed by both transports. Pairing the transported parent carrier witness with the transported predicate witness gives $S_P(k)$. □

Theorem 113.11 (SubtypeUp NameCert surface obligation)**.** *The carrier row of Definition 113.8, the restricted-classifier rows of Definition 113.9, the endpoint row, and the transport row of Definition 113.10 assemble the displayed* $\mathsf{NameCert}(S_P, \sim_{S_P})$ *surface for* $\mathsf{Subtype}^{\uparrow}$*. The ten-field template entries are: parent carrier* S*, parent classifier* $\sim_S$*, parent certificate* $\mathsf{NameCert}(S, \sim_S)$*, predicate* P*, predicate-stability row, restricted carrier* S_P*, restricted classifier* $\sim_{S_P}$*, displayed endpoint* s_0*, parent-projection row, and carrier-transport row.*

Proof. Use Definition 113.8 for the carrier and endpoint rows, Definition 113.9 for the parent-projection and inclusion rows, and Definition 113.10 for carrier transport.

These are exactly the rows consumed by Definition 113.7. Applying that theorem repacks the rows into the public $\mathsf{NameCert}(S_P, \sim_{S_P})$ surface, with the ten displayed entries recording the source data and the exactness maps used by the construction. □

Theorem 113.12 (SubtypeUp public NameCert export)**.** *Under the hypotheses of Definition 113.7, the public interface exported by* $\mathsf{Subtype}^{\uparrow}$ *is the certificate*

$$\mathsf{NameCert}(S_P, \sim_{S_P}).$$

Its public rows are the restricted carrier S_P, the restricted classifier $\sim_{S_P}$, the displayed endpoint, the parent-projection and inclusion rows, the predicate-stability transport row, and the assembled carrier-transport row.

Lean4 ✓ `BEDC.Derived.SubtypeUp.SubtypePublicNameCert export`

Proof. The carrier row is Definition 113.8: public membership in S_P is exactly parent membership in S together with the predicate witness P.

The restricted-classifier rows are Definition 113.9. They identify a public comparison in $\sim_{S_P}$ with the parent classifier comparison in $\sim_S$, with the forward parent-projection row and reverse inclusion row both displayed on carried subtype endpoints.

The predicate-transport row is Definition 113.10. It combines parent carrier transport with stability of P and produces the transported public carrier witness for S_P.

Apply Definition 113.7. The preceding rows provide exactly the carrier, classifier, endpoint, and transport data consumed by that theorem, so the exported public surface is the certificate $\mathsf{NameCert}(S_P, \sim_{S_P})$. □

Theorem 113.13 (SubtypeUp standard bridge surface)**.** *Under the hypotheses of Definition 113.7, the concrete restricted carrier S_P and restricted classifier $\sim_{S_P}$ form the paper-side standard subtype bridge. Its rows are exactly:*

1. *carrier membership in the restricted endpoint is $S_P(h)$, equivalently $S(h) \wedge P(h)$;*
2. *classifier comparison between carried restricted endpoints is $h \sim_{S_P} k$, equivalently the parent comparison $h \sim_S k$;*
3. *the projection to the parent certificate preserves the carrier row by forgetting only the predicate component and preserves the classifier row by returning the same parent comparison;*
4. *predicate stability reflects transported parent endpoints back into S_P, so every transported parent endpoint with the transported predicate witness is again carried by the restricted carrier.*

Consequently the public $\mathsf{NameCert}(S_P, \sim_{S_P})$ surface exported by $\mathsf{Subtype}^{\uparrow}$ is equipped with the bridge rows needed for the standard subtype bridge, without adding host subtype equality, quotient extensionality, or an abstract source-equivalence chain.

Lean4 ✓ `BEDC.Derived.SubtypeUp.SubtypeUp StdBridge`

Proof. The carrier row is exactly Definition 113.8. It identifies membership in the concrete restricted carrier with the pair of parent membership and predicate witness, and it records the displayed endpoint data used by the certificate.

The classifier row is exactly Definition 113.9. Its forward direction is the projection from a restricted comparison to the parent comparison, and its reverse direction is the inclusion of a parent comparison between carried restricted endpoints.

The transport row is Definition 113.10. Parent carrier transport supplies the transported parent endpoint, and predicate stability supplies the transported predicate witness; pairing them gives membership back in S_P.

Finally, Definition 113.12 repacks these rows into the public $\mathsf{NameCert}(S_P, \sim_{S_P})$ export. The bridge therefore uses only the concrete carrier, concrete classifier, parent projection, inclusion row, and predicate-stability transport already displayed in the $\mathsf{Subtype}^{\uparrow}$ construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{Subtype}^{\uparrow}}$.

Constructive story: A subtype starts with a parent $\mathsf{Set}^{\uparrow}$ name certificate with carrier S and classifier $\sim_S$, then cuts that carrier by a predicate P that is stable under the parent classifier. A history h is admitted exactly when it carries both pieces of evidence, $S(h)$ and $P(h)$; anything missing either parent membership or the predicate witness is outside the restricted carrier. Two subtype endpoints classify together by the same parent comparison $h \sim_S k$, so the predicate proof is not compared by a second equality and no quotient-style subtype extensionality is formed. The source consists of the parent certificate, the predicate, a displayed endpoint, and the stability row for P; the pattern records restriction, projection to the parent, inclusion from parent comparisons between carried endpoints, and predicate transport, while the ledger keeps the parent witness and predicate witness as the visible pair. The five name-certificate obligations are exactly the displayed endpoint for inhabitation, parent reflexivity, symmetry, and transitivity for the restricted classifier, and carrier closure obtained by parent carrier transport plus stability of P. A history h lies in the signature gap of a $\mathsf{Subtype}^{\uparrow}$ token p when the source-domain accepts both $S(h)$ and $P(h)$, the signature relation pairs that restricted endpoint with an answer endpoint and evidence ledger, and token introduction names that answer as p; unfolded, h is the parent element encoded by p together with the required predicate witness. The standard bridge reads the same data as an ordinary subtype by projecting to the parent and reading back only endpoints whose predicate witness is displayed and stable. The chapter registers this predicate-cut construction under the name $\mathsf{Subtype}^{\uparrow}$; refinement is a carrier restriction over an existing certificate, not host subtype equality, quotient extensionality, or a second parent source.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Subtype}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature scope binds the parent $\mathsf{NameCert}(S, \sim_S)$, restricted carrier S_P, restricted classifier $\sim_{S_P}$, predicate-stability transport, classifier-closure lemma, public NameCert export, and paper-side standard subtype bridge surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Subtype}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SubtypeUp.SubtypeRestrictedClassifier\_nameCert`

Bridge status: bridgeChecked

Not claimed: This packet does not close full SetUp bridges or quotient-style subtype extensionality.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 114

A Concrete Naming Certificate for $\mathsf{Empty}^{\uparrow}$

The empty-type interface $\mathsf{Empty}^{\uparrow}$ records the impossible BEDC history carrier: the carrier has no inhabitant. The useful exported facts are the absurdity theorems below for carrier witnesses, for NameCert attempts over that carrier, and for SemanticNameCert attempts over that carrier.

This impossible-carrier surface is also the endpoint consumed by $\mathsf{Unit}^{\uparrow}$ in chapter 115: the unit certificate reads this empty endpoint as the terminal carrier row before packaging the local singleton continuation surface.

114.1 Carrier definition

Definition 114.1 (Impossible empty-history carrier)**.** *For a history h : Hist, the empty-history carrier requires both boundary readbacks*

$$\mathsf{hsame}(h, \mathsf{E0}(\mathsf{e})) \quad \textit{and} \quad \mathsf{hsame}(h, \mathsf{E1}(\mathsf{e})).$$

Thus it is the concrete impossible carrier whose contradiction is witnessed by BEDC constructor separation.

Lean4 (definition) `BEDC.Derived.EmptyUp.EmptyHistoryCarrier`
Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyUpTasteGate single carrier alignment`

114.2 The certificate

Theorem 114.2 (Empty history carrier absurd)**.** *No history inhabits the empty-history carrier.*

Proof. Unpack the two carrier witnesses. The first identifies h with $\mathsf{E0}(\mathsf{e})$, while the second identifies the same h with $\mathsf{E1}(\mathsf{e})$. The generated history-sameness constructors have no rule relating a zero-headed history to a one-headed history, so constructor inversion closes the contradiction. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyHistoryCarrier absurd`

Theorem 114.3 (Empty history carrier name certificate absurdity)**.** *There is no name certificate whose carrier is the empty-history carrier and whose classifier is* hsame*:*

$$\mathsf{NameCert}(\mathsf{EmptyHistoryCarrier}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. A name certificate contains a carrier witness. For this carrier, that witness is an inhabitant of the empty-history carrier, contradicting the carrier absurdity theorem above. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyHistoryCarrier namecert absurd`

Theorem 114.4 (Empty history carrier semantic certificate absurdity)**.** *There is no semantic name certificate whose source, pattern, and ledger policies are all the empty-history carrier:*

$$\mathsf{SemanticNameCert}(\mathsf{EmptyHistoryCarrier}, \mathsf{EmptyHistoryCarrier}, \mathsf{EmptyHistoryCarrier}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. The semantic certificate supplies a ledger witness. Since the ledger policy is the empty-history carrier itself, this witness is an inhabitant of the impossible carrier. The carrier absurdity theorem closes the contradiction. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyHistoryCarrier semantic namecert absurd`

Theorem 114.5 (Empty source carrier semantic certificate absurdity)**.** *There is no semantic name certificate whose source predicate is the empty-history carrier, regardless of the displayed pattern, ledger policy, and classifier:*

$$\mathsf{SemanticNameCert}(\mathsf{EmptyHistoryCarrier}, P, L, R) \Longrightarrow \bot.$$

Proof. The core field of a semantic certificate contains a name certificate over the source predicate. Therefore it contains an inhabitant of $\mathsf{EmptyHistoryCarrier}$, contradicting the carrier absurdity theorem. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyHistoryCarrier source semantic namecert absurd`

Theorem 114.6 (Empty contradictory carrier name certificate absurdity)**.** *There is no name certificate whose carrier asks a single history to be both the empty history and the one-extension of the empty history:*

$$\mathsf{NameCert}(\{h \mid h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(\mathsf{e})\}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. Take a carrier witness from the certificate. Its two endpoint equations compose by symmetry and transitivity of hsame to give $\mathsf{ehsameE1}(\mathsf{e})$. The closed history no-confusion theorem rules out that comparison. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyContradictoryCarrier namecert absurd`

Theorem 114.7 (Empty contradictory carrier semantic certificate absurdity)**.** *There is no semantic name certificate whose source, pattern, and ledger policies all ask a single history to be both the empty history and the one-extension of the empty history:*

$$\mathsf{SemanticNameCert}(\{h \mid h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(\mathsf{e})\}, \{h \mid h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(\mathsf{e})\}, \{h \mid h\mathsf{hsamee} \wedge h\mathsf{hsameE1}(\mathsf{e})\}, \mathsf{hsame})$$

Proof. The semantic certificate supplies a ledger witness for the contradictory carrier. Its two endpoint equations compose by symmetry and transitivity of hsame to give $\mathsf{ehsameE1}(\mathsf{e})$, contradicting BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyContradictoryCarrier semantic namecert absurd`

Theorem 114.8 (Empty visible contradictory carrier name certificate absurdity)**.** *There is no name certificate whose carrier asks a single history to classify both to* e *and to some visible endpoint* $\mathsf{E0}(tail)$ *or* $\mathsf{E1}(tail)$*:*

$$\mathsf{NameCert}(\{h \mid h\mathsf{hsamee} \wedge (h\mathsf{hsameE0}(tail) \vee h\mathsf{hsameE1}(tail))\}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. Take a carrier witness from the certificate. The empty comparison composes against the visible endpoint comparison. In the zero-headed branch this gives $\mathsf{ehsameE0}(tail)$, and in the one-headed branch it gives $\mathsf{ehsameE1}(tail)$. Both are excluded by BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisibleContradictoryCarrier namecert absurd`

Theorem 114.9 (Empty visible pair contradictory carrier name certificate absurdity)**.** *There is no name certificate whose carrier asks a single history to classify both to a zero-visible endpoint and to a one-visible endpoint:*

$$\mathsf{NameCert}(\{h \mid h\mathsf{hsameE0}(leftTail) \wedge h\mathsf{hsameE1}(rightTail)\}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. Take a carrier witness from the certificate. The zero-endpoint comparison, reversed, composes with the one-endpoint comparison to give $\mathsf{E0}(leftTail)\mathsf{hsameE1}(rightTail)$. BEDC history constructor separation rules out any such zero-versus-one comparison. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisiblePairContradictoryCarrier namecert absurd`

Theorem 114.10 (Empty visible pair carrier name certificate absurdity)**.** *There is no name certificate whose carrier asks a single history to classify both to a zero-headed visible endpoint and to a one-headed visible endpoint:*

$$\mathsf{NameCert}(\{h \mid h\mathsf{hsameE0}(left) \wedge h\mathsf{hsameE1}(right)\}, \mathsf{hsame}) \Longrightarrow \bot.$$

Proof. Take a carrier witness from the certificate. The two endpoint comparisons compose by symmetry and transitivity of hsame to give $\mathsf{E0}(left)\mathsf{hsameE1}(right)$. BEDC history constructor separation rules out every zero-headed versus one-headed comparison. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisiblePairCarrier namecert absurd`

Theorem 114.11 (Empty visible pair carrier semantic certificate absurdity)**.** *There is no semantic name certificate whose source, pattern, and ledger policies all ask a single history to classify both to a zero-headed visible endpoint and to a one-headed visible endpoint:*

$$\mathsf{SemanticNameCert}(\{h \mid h\mathsf{hsameE0}(left) \wedge h\mathsf{hsameE1}(right)\}, \{h \mid h\mathsf{hsameE0}(left) \wedge h\mathsf{hsameE1}(right)\}, \{h \mid h\mathsf{hsameE}$$

Proof. The semantic certificate supplies a ledger witness. The ledger policy is the same visible-pair carrier, so the witness compares one history with both $\mathsf{E0}(left)$ and $\mathsf{E1}(right)$. Reversing the zero comparison and composing with the one comparison gives $\mathsf{E0}(left)\mathsf{hsameE1}(right)$, which is excluded by BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisiblePairCarrier semantic namecert absurd`

Theorem 114.12 (Empty visible pair source semantic certificate absurdity)**.** *There is no semantic name certificate whose source predicate asks one history to compare both to a zero-headed endpoint and to a one-headed endpoint, for any pattern predicate, ledger predicate, and classifier relation:*

$$\mathsf{SemanticNameCert}(\{h \mid h\mathsf{hsameE0}(left) \wedge h\mathsf{hsameE1}(right)\}, \mathsf{PatternSpec}, \mathsf{LedgerPolicy}, \mathsf{ClassifierSpec}) \Longrightarrow \bot.$$

Proof. Use the core name certificate's carrier witness. Its source proof supplies the two endpoint comparisons; reversing the zero-headed comparison and composing with the one-headed comparison gives $\mathsf{E0}(left)\mathsf{hsameE1}(right)$, excluded by BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisiblePairCarrier source semantic namecert absurd`

Theorem 114.13 (Empty visible contradictory carrier semantic certificate absurdity)**.** *There is no semantic name certificate whose source, pattern, and ledger policies all ask a single history to classify both to* e *and to some visible endpoint* $\mathsf{E0}(tail)$ *or* $\mathsf{E1}(tail)$*:*

$$\mathsf{SemanticNameCert}(\{h \mid h\mathsf{hsamee} \wedge (h\mathsf{hsameE0}(tail) \vee h\mathsf{hsameE1}(tail))\}, \{h \mid h\mathsf{hsamee} \wedge (h\mathsf{hsameE0}(tail) \vee h\mathsf{hsameE1}(t$$

Proof. The semantic certificate supplies a ledger witness. The ledger policy is the same visible contradictory carrier, so the witness gives both the empty comparison and one visible endpoint comparison. In the zero-headed branch this composes to $\mathsf{ehsameE0}(tail)$, and in the one-headed branch to $\mathsf{ehsameE1}(tail)$. BEDC history constructor separation rules out both branches. □

Lean4 ✓ `BEDC.Derived.EmptyUp.EmptyVisibleContradictoryCarrier semantic namecert absurd`

Closure status for $\mathsf{NameCert}_{\mathsf{Empty}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Empty}^{\uparrow}$ packet is generated from the impossible empty-history carrier, its contradictory visible refinements, semantic certificate rows, componentwise hsame comparisons, and the local absurdity family.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Empty}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 114.2, Definition 114.3, Definition 114.4, and the visible-pair absurdity theorems bind the impossible carrier, visible-pair refinements, semantic NameCert rows, BHist constructor separation, hsame transport, and local absurdity family.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Empty}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.EmptyUp.EmptyHistoryCarrier\_absurd`

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim inhabited empty carriers, public standard bridge data, or bridge checking for $\mathsf{Empty}^{\uparrow}$.

Upgrade path: No further paper-side closure upgrade remains; only bridge verification may be supplied for the public impossible-carrier surface.

Chapter 115

A Concrete Naming Certificate for $\mathsf{Unit}^{\uparrow}$

The unit-type interface $\mathsf{Unit}^{\uparrow}$ packages the unique naming certificate over the inductive carrier with a single constructor. It is the terminal object in the category of naming certificates and supplies the canonical trivial witness for placeholder constructions.

The $\mathsf{Unit}^{\uparrow}$ packet consumes the $\mathsf{Empty}^{\uparrow}$ empty-history endpoint of chapter 114 as its terminal carrier row. Every unit classifier theorem below reads the same empty endpoint surface through that sibling certificate before packaging the continuation endpoint into the local $\mathsf{NameCert}_{\mathsf{Unit}^{\uparrow}}$ row.

115.1 Carrier definition

Definition 115.1 (Unit history carrier)**.** *For a history $h : \mathsf{BHist}$, the unit carrier is the continuation endpoint that composes the empty history with h and returns the empty history:*

$$\mathsf{UnitHistoryCarrier}(h) := \mathsf{Cont}(\mathsf{e}, h, \mathsf{e}).$$

Lean4 (definition) `BEDC.Derived.UnitUp.UnitHistoryCarrier`

Definition 115.2 (Unit history classifier)**.** *For histories $h, k : \mathsf{BHist}$, the unit classifier requires both endpoints to lie in the unit carrier and relates them by* hsame*:*

$$\mathsf{UnitHistoryClassifier}(h, k) := \mathsf{UnitHistoryCarrier}(h) \wedge \mathsf{UnitHistoryCarrier}(k) \wedge h\mathsf{hsame}k.$$

Lean4 (definition) `BEDC.Derived.UnitUp.UnitHistoryClassifier`

Theorem 115.3 (Unit history classifier empty endpoints iff)**.** *For histories $h, k : \mathsf{BHist}$, the unit classifier is equivalent to both endpoints being empty:*

$$\mathsf{UnitHistoryClassifier}(h, k) \iff h\mathsf{hsame}\mathsf{e} \wedge k\mathsf{hsame}\mathsf{e}.$$

Proof. In the forward direction, each carrier component is a continuation $\mathsf{Cont}(\mathsf{e}, -, \mathsf{e})$; empty-result inversion identifies the endpoint with e. In the reverse direction, the endpoint equalities reduce both carrier continuations to the empty left-unit continuation, and transitivity of hsame supplies the classifier equality. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier empty endpoints iff`

Theorem 115.4 (Unit history classifier append split)**.** *For histories p, q, h : BHist, the unit classifier from the appended endpoint $\mathsf{append}(p, q)$ to h is equivalent to all three component histories being empty:*

$$\mathsf{UnitHistoryClassifier}(\mathsf{append}(p, q), h) \iff p\mathsf{hsamee} \wedge q\mathsf{hsamee} \wedge h\mathsf{hsamee}.$$

Proof. Apply the empty-endpoint characterization of the unit classifier to obtain $\mathsf{append}(p, q)\mathsf{hsamee}$ and $h\mathsf{hsamee}$. The append endpoint is empty exactly when both append factors are empty. Conversely, the two empty append factors rebuild the empty appended endpoint, and the empty-endpoint classifier characterization supplies the unit classifier. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier append split iff`

Theorem 115.5 (Unit history classifier visible endpoint absurd)**.** *No visible endpoint can occur on either side of the unit history classifier. For all histories p, q, k, each classifier instance $\mathsf{UnitHistoryClassifier}(\mathsf{e0}(p), k)$, $\mathsf{UnitHistoryClassifier}(\mathsf{e1}(p), k)$, $\mathsf{UnitHistoryClassifier}(k, \mathsf{e0}(q))$, and $\mathsf{UnitHistoryClassifier}$ is impossible.*

Proof. Each classifier witness contains the unit carrier witness for the visible endpoint. The left-unit continuation result identifies that visible endpoint with the empty history, contradicting constructor disjointness for $\mathsf{e0}$ and $\mathsf{e1}$. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier visible endpoint absurd`

Theorem 115.6 (Unit history classifier continuation middle carrier)**.** *For histories $left, middle, right$: BHist, if the unit classifier relates $left$ and $right$ and $\mathsf{Cont}(left, middle, right)$, then $middle$ lies in the unit history carrier:*

$$\mathsf{UnitHistoryClassifier}(left, right) \wedge \mathsf{Cont}(left, middle, right) \implies \mathsf{UnitHistoryCarrier}(middle).$$

Proof. The classifier empty-endpoint theorem identifies both classified endpoints with e. Transport the displayed continuation result along the empty right endpoint, then apply empty-result inversion for continuation to identify the middle history with e. The carrier-empty equivalence turns this history-sameness witness into the required unit carrier witness. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier continuation middle carrier`

Theorem 115.7 (Unit history carrier empty equivalence)**.** *For a history h : BHist, membership in the unit history carrier is equivalent to history-sameness with the empty history:*

$$\mathsf{UnitHistoryCarrier}(h) \iff \mathsf{hsame}(h, \mathsf{e}).$$

Proof. The forward direction applies the right-unit uniqueness law to $\mathsf{Cont}(\mathsf{e}, h, \mathsf{e})$. Conversely, a witness $\mathsf{hsame}(h, \mathsf{e})$ reduces the endpoint to the empty history, where the left-unit continuation law gives $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryCarrier empty iff`

Theorem 115.8 (Unit history carrier continuation suffix closure)**.** *For histories h, k, r : BHist, if h lies in the unit history carrier, $\mathsf{Cont}(h, k, r)$ holds, and $r\mathsf{hsamee}$, then k lies in the unit history carrier.*

Proof. The carrier characterization identifies h with e. Transporting the displayed continuation along this source identification, the reflexive suffix identification, and the empty result identification gives $\mathsf{Cont}(\mathsf{e}, k, \mathsf{e})$, which is exactly the unit history carrier for k. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryCarrier continuation suffix closed`

Theorem 115.9 (Unit history carrier continuation prefix closure)**.** *For histories $h, k, r : \mathsf{BHist}$, if $\mathsf{Cont}(h, k, r)$ holds and $r\mathsf{hsame}\mathsf{e}$, then h lies in the unit history carrier.*

Proof. Transport the continuation result to e and invert the empty-result continuation. The inversion identifies the left endpoint h with e, and the empty characterization of the unit history carrier converts this endpoint identity into $\mathsf{UnitHistoryCarrier}(h)$. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryCarrier continuation prefix closed`

115.2 The certificate

Theorem 115.10 (Unit history classifier semantic name certificate)**.** *The unit history carrier, used as source, pattern, and ledger policy, forms a semantic name certificate with the unit history classifier.*

Proof. The empty history inhabits the carrier by the left-unit continuation law. Classifier reflexivity, symmetry, and transitivity are obtained by packaging the carrier endpoints together with the corresponding hsame laws. Carrier transport follows from the target carrier endpoint already present in the classifier witness. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier semanticNameCert`

Theorem 115.11 (Unit terminal map uniqueness)**.** *Let $S : \mathsf{BHist} \to \mathsf{Prop}$ be a source carrier, let $R_S : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}$ be its source classifier, and let $\mathcal{N}_S : \mathsf{NameCert}(S, R_S)$ be a source naming certificate in the sense of Definition 9.7. For maps $f, g : \mathsf{BHist} \to \mathsf{BHist}$, assume*

$$\forall a : \mathsf{BHist},\ S(a) \Longrightarrow \mathsf{UnitHistoryCarrier}(f(a))$$

and

$$\forall a : \mathsf{BHist},\ S(a) \Longrightarrow \mathsf{UnitHistoryCarrier}(g(a)).$$

Then every carried source endpoint has classified unit images:

$$\forall a : \mathsf{BHist},\ S(a) \Longrightarrow \mathsf{UnitHistoryClassifier}(f(a), g(a)).$$

Proof. Fix a carried source endpoint a with witness $s : S(a)$. The two preservation hypotheses give unit-carrier witnesses for $f(a)$ and $g(a)$. By Definition 115.7, these witnesses identify both images with e:

$$f(a)\mathsf{hsame}\mathsf{e} \qquad \text{and} \qquad g(a)\mathsf{hsame}\mathsf{e}.$$

Symmetry turns the second identification into $\mathsf{e}\mathsf{hsame}g(a)$, and transitivity gives $f(a)\mathsf{hsame}g(a)$. The definition of the classifier in Definition 115.2 is exactly the conjunction of the two unit-carrier witnesses and this hsame relation. Hence $\mathsf{UnitHistoryClassifier}(f(a), g(a))$ holds. Since a and s were arbitrary, the displayed universal statement follows. The source certificate $\mathcal{N}_S$ fixes the certified source context; no source-classifier compatibility is needed because both images land in the concrete unit carrier of Definition 115.1. □

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier terminal map uniqueness`

Lean4 ✓ `BEDC.Derived.UnitUp.UnitHistoryClassifier terminal map unique`

Closure status for $\mathsf{NameCert}_{\mathsf{Unit}^\uparrow}$**.**

Constructive story: A $\mathsf{Unit}^\uparrow$ packet is generated by the terminal empty-history carrier, the unit history classifier, continuation endpoint closure, terminal-map uniqueness, and the semantic NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Unit}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 115.1, Definition 115.2, Definition 115.7, Definition 115.4, Definition 115.5, Definition 115.6, Definition 115.10, and the terminal-map uniqueness row expose the public dependency surface for downstream terminal-object consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Unit}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.UnitUp.UnitHistoryClassifier\_semanticNameCert`

Bridge status: none

Not claimed: This chapter does not construct nontrivial inhabitants beyond the terminal unit surface or identify arbitrary source carriers with $\mathsf{Unit}^\uparrow$.

Upgrade path: A bridge-checked closure would require a standard bridge packaging this public terminal-object surface against the ambient schema.

Chapter 116

A Concrete Naming Certificate for $\mathsf{Permutation}^{\uparrow}$

The permutation interface $\mathsf{Permutation}^{\uparrow}$ packages the carrier of bijections of a $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112) onto itself. It supplies the underlying carrier for $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$ (chapter 117) and the action data for combinatorial group-theoretic certificates.

116.1 Carrier definition

Definition 116.1 (Permutation bijection source row)**.** *Fix a finite-set source certificate* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *with carrier row* F*, classifier* $\sim_F$*, finite-spine endpoint row* xs*, and enumeration ledger as in chapter 112. A BEDC-facing* $\mathsf{Permutation}^{\uparrow}$ *source row over* F *is the* BHist*-visible package*

$$(\mathsf{src}, \mathsf{tgt}, G, G^{-1}, W, C, A, L)$$

with the following rows.

1. src *and* tgt *are the two displayed endpoint rows for the source and target copies of the same carried* $\mathsf{FinSet}^{\uparrow}$ *object.*
2. G *is a bijection graph/probe row: for every source endpoint read by the* $\mathsf{ProbeBundle}$*, the graph supplies exactly one target endpoint, and both endpoints are classified by the fixed* $\mathsf{FinSet}^{\uparrow}$ *carrier.*
3. G^{-1} *is the inverse graph row with the same readback discipline in the opposite direction.*
4. W *is the pair of inverse-witness rows saying that every forward read is met by a backward read and every backward read is met by a forward read at the corresponding endpoint.*
5. C *is the classifier and composition row: two graph presentations classify together exactly when their action outputs agree by the* $\mathsf{FinSet}^{\uparrow}$ *classifier, and the composites* $G^{-1} \circ G$ *and* $G \circ G^{-1}$ *transport to the corresponding identity graph rows by* hsame *and by the package provenance attached to the two endpoint rows.*
6. A *is the action row sending a carried source endpoint through* G *to its carried target endpoint.*
7. L *is the ledger containing the hidden enumeration witness for the chosen finite spine, the hidden order witness used to name positions, the* Pkg *provenance for the two endpoint copies, and the probe-read entries that justify graph coverage.*

The only visible carrier fields are the two $\mathsf{FinSet}^{\uparrow}$ *endpoint rows, the graph, inverse graph, inverse-witness rows, classifier and composition row, action row, and ledger boundary. Enumeration and order witnesses are not exposed as host data; they are consumed through* L.

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationTasteGate single carrier alignment`

116.2 The certificate

Theorem 116.2 (Permutation FinSet carrier obligation)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *presented by* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$ *and* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ *on a finite enumeration spine. The* $\mathsf{Permutation}^{\uparrow}$ *carrier over this source certificate is the predicate of certified finite-position self-bijection rows on that spine: a carried row consists of a forward position map, an inverse position map, the two inverse identities, and the pointwise evidence that matched entries are carried by* A *and classified by* $\sim_A$*. The visible ledger is exactly the corresponding* $\mathsf{EnumPerm}_{A,\sim_A}$ *row together with the source and target* $\mathsf{FinSet}^{\uparrow}$ *endpoint rows.*

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow carrier surface`

Proof. Unfold the finite-set source presentation from Definition 112.5, Definition 112.1, and Definition 112.2. A carried permutation row is therefore read as a certified bijection of the finite position carrier $\mathsf{Pos}(xs)$ onto itself.

The forward map, inverse map, and two inverse identities are the certified-bijection fields of $\mathsf{EnumPerm}_{A,\sim_A}$. The pointwise preservation field records that each source entry and matched target entry are carried by A and related by $\sim_A$.

Conversely, those same fields repack as the BHist permutation source row of Definition 116.1: the endpoint copies are the two finite-set rows, the maps give the graph and inverse graph, the inverse identities give the inverse-witness row, and the pointwise carrier-classifier evidence is the ledger entry. No host enumeration, host order, or bare host function is exposed outside that ledger. □

Theorem 116.3 (Permutation classifier obligation)**.** *For two carried* $\mathsf{Permutation}^{\uparrow}$ *rows over the fixed* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *above, the classifier relates them exactly when their actions send every carried finite-set entry to* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$*-equivalent entries. Equivalently, for each source entry represented by* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$*, the two target entries selected by the two certified self-bijections are identified by the finite-set enumeration classifier, and this pointwise comparison is the whole* $\mathsf{Permutation}^{\uparrow}$ *classifier row.*

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow action ledger transport`
Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow endpoint package hsame transport`

Proof. Unpack the two carried rows as $\mathsf{EnumPerm}_{A,\sim_A}$ witnesses on the same finite spine. Each action value is a target endpoint already carried by $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$, because the pointwise field of the corresponding enumeration permutation contains the carrier witness for the matched target entry.

The source $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ supplies the classifier fields for $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$. Applying those fields pointwise to the two target entries gives the displayed comparison of action outputs.

The graph comparison is carried by Pkg provenance and $\mathsf{ProbeBundle}$ reads over BHist. Applying hsame transport to those package rows preserves the visible graph agreement while leaving enumeration and order witnesses inside the ledger. The existing permutation-invariance row Definition 112.3 ensures that changing the hidden finite-position witness by a certified enumeration

permutation does not change the finite-set classifier. Thus the classifier only observes action outputs through $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$, and the pointwise action comparison repacks as the required $\mathsf{Permutation}^{\uparrow}$ classifier row. □

Theorem 116.4 (Permutation composition inverse action obligation)**.** *Identity, composition, inverse, and action compatibility for carried* $\mathsf{Permutation}^{\uparrow}$ *rows are exactly the reflexivity, transitivity, symmetry, and permutation-invariance rows of* $\mathsf{FinsetEnumerationPermutation}$*. The identity row is the certified identity bijection, the composite row is certified bijection composition, the inverse row is certified bijection inversion, and action compatibility is the induced invariance of* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ *under these rows.*

$$\mathsf{id} \circ G \sim_{\mathsf{Graph}} G$$

and

$$G^{-1} \circ G \sim_{\mathsf{Graph}} \mathsf{id}_{\mathsf{src}}$$

are therefore ledger readings of the same finite-position facts, not host-function equalities.

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow composition action ledger scope`

Proof. For identity, apply Definition 112.26. It supplies the certified identity bijection on the finite position carrier, hence the identity $\mathsf{Permutation}^{\uparrow}$ row and its action on carried entries.

For composition, apply Definition 112.14. Its composite finite-position bijection is read as the graph row for the composite permutation, and its pointwise preservation field is the carrier-classifier ledger for the composite action.

For inverse, apply Definition 112.25. The certified inverse position map becomes the inverse graph row, and the two inverse identities become the two graph identity rows recorded in Definition 116.1. The forward graph, backward graph, and inverse-witness rows give the displayed identity classifications after transport by hsame and by the Pkg provenance recorded for the graph reads.

Finally, action compatibility is the classifier-invariance statement of Definition 112.3. It says that the action output selected by a certified finite-position permutation is classified in the same finite-set sense as the corresponding endpoint entry. Coverage, soundness, and completeness of that action are read through the graph rows, while enumeration choices, order choices, and position names remain ledger data in L. These rows are precisely the action and ledger boundary consumed by $\mathsf{SymGroup}^{\uparrow}$. □

Theorem 116.5 (Permutation public NameCert surface)**.** *Over the fixed* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *of Definition 116.2, the carrier row of Definition 116.2, the classifier row of Definition 116.3, and the composition, inverse, action, and ledger rows of Definition 116.4 assemble a public* $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ *surface. The exported surface contains exactly the certified finite-position self-bijection carrier, the pointwise finite-set action classifier, the identity-composition-inverse rows, the action row, and the visible ledger boundary over* BHist*. The downstream* $\mathsf{SymGroup}^{\uparrow}$ *boundary is exposed only as a consumer row of this public surface, not as an additional* $\mathsf{Permutation}^{\uparrow}$ *field.*

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow semantic name certificate`
Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow public name certificate`

Proof. First fix the carrier. Apply Definition 116.2 to read carried $\mathsf{Permutation}^{\uparrow}$ rows as certified finite-position self-bijections over the chosen $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ source.

Next fix the classifier. Apply Definition 116.3 to identify classifier comparison with pointwise comparison of action outputs through $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$.

Finally apply Definition 116.4. It supplies the identity, composition, inverse, action, and ledger rows over the same carrier and classifier. These rows repack the displayed inventory as a public $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ surface, while the $\mathsf{SymGroup}^{\uparrow}$ row is only the downstream consumer boundary. No standard bridge or Lean verification is claimed here. □

Theorem 116.6 (Permutation bridge to symmetric-group composition)**.** *The public* $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ *surface of Definition 116.5 supplies exactly the carrier, classifier, identity, composition, inverse, action, and ledger rows consumed by the* $\mathsf{SymGroup}^{\uparrow}$ *composition interface of chapter 117. Thus the symmetric-group carrier, classifier, and composition obligations read the already public permutation source boundary, and no additional host bijection or host function equality is exposed at the bridge.*

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationSymGroupComposition bridge`

Proof. Apply Definition 116.5. It identifies the exported $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ surface with the certified finite-position self-bijection carrier, pointwise finite-set action classifier, identity-composition-inverse rows, action row, and visible ledger boundary.

Apply Definition 116.4. The composition, inverse, identity, and action rows are ledger readings of certified finite-position permutation facts, so they are already in the form required by the group-facing operation boundary.

The $\mathsf{SymGroup}^{\uparrow}$ carrier and classifier obligations of chapter 117 read this same permutation source row, and Definition 117.3 consumes its composition, inverse, action, exactness, and ledger rows. Therefore the bridge is a paper-side repacking of the public $\mathsf{Permutation}^{\uparrow}$ surface into the symmetric-group composition interface, with all enumeration choices, position names, and host bijection data remaining inside the ledger. □

Theorem 116.7 (Permutation bridged surface completeness)**.** *The public* $\mathsf{Permutation}^{\uparrow}$ *surface of Definition 116.5, together with the* $\mathsf{SymGroup}^{\uparrow}$ *consumer bridge of Definition 116.6, exhausts the visible BEDC fields of the bridged permutation interface. The exhausted fields are carrier, classifier, identity, composition, inverse, action, and ledger boundary. The finite enumeration witness and order witness remain hidden in the ledger and are not exported as additional visible fields.*

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow bridged surface completeness`

Proof. Apply Definition 116.2, Definition 116.3, and Definition 116.4. These three obligation theorems provide the carrier row, the pointwise classifier row, and the identity, composition, inverse, action, and ledger rows over the fixed finite-set source certificate.

Apply Definition 116.5. It repacks exactly those rows as the public $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ surface and states that the downstream $\mathsf{SymGroup}^{\uparrow}$ boundary is a consumer row rather than an extra $\mathsf{Permutation}^{\uparrow}$ field.

Apply Definition 116.6. The bridge consumes that same public surface as the symmetric-group carrier, classifier, composition, inverse, action, exactness, and ledger boundary. Therefore projection from the public surface to the consumer bridge, followed by repacking of the consumed rows, returns exactly the visible permutation interface listed in the statement. The finite enumeration and order witnesses are only read through the ledger, so no further visible BEDC field remains outside the displayed inventory. □

Closure status for $\mathsf{NameCert}_{\mathsf{Permutation}^\uparrow}$.

Constructive story: A permutation starts from a fixed $\mathsf{FinSet}^\uparrow$ finite spine and admits only certified self-bijection rows on its finite position carrier: a forward graph, an inverse graph, the two inverse-witness rows, and pointwise evidence that matched entries are carried and classified by the finite-set source. The classifier between two carried permutation rows compares their actions pointwise through $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$, so two presentations are the same only by their certified effects on the finite entries, not by host function equality. The source is the finite-set certificate together with the two endpoint copies of the same carried finite object, the pattern records graph coverage, inverse coverage, identity, composition, inverse, and action compatibility, and the ledger keeps enumeration witnesses, order witnesses, probe reads, and package provenance hidden behind the visible graph rows. The five name-certificate obligations come from the certified identity self-bijection for inhabitation, pointwise reflexivity, symmetry, and transitivity of the finite-set classifier, and closure under transported graph and endpoint package rows. A graph row G lies in the signature gap of a $\mathsf{Permutation}^\uparrow$ token p when the source-domain accepts the fixed finite spine and the certified forward/inverse action package, the signature relation pairs G with an answer permutation and evidence ledger, and token introduction names that answer as p; unfolded, every finite position is sent to the target position encoded by p and the inverse rows certify the round trip. The bridge exports exactly the same carrier, classifier, identity, composition, inverse, action, and ledger rows to the $\mathsf{SymGroup}^\uparrow$ composition interface. The chapter registers this finite-position bijection construction under the name $\mathsf{Permutation}^\uparrow$; a permutation is a certified action ledger over a finite spine, not a bare host bijection, exposed enumeration order, function quotient, or separate symmetric-group object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Permutation}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The bridged surface is the BHist permutation source row of Definition 116.1 together with the carrier, classifier, composition, inverse, action, ledger, public $\mathsf{NameCert}_{\mathsf{Permutation}^\uparrow}$ packaging, and symmetric-group composition bridge of Definition 116.2, Definition 116.3, Definition 116.4, Definition 116.5, and Definition 116.6.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Permutation}^\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow\_public\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not claim machine-checked StandardBridge coverage or full $\mathsf{SymGroup}^\uparrow$ closure.

Upgrade path: No further paper-side theory upgrade is required; machine-checked standard-bridge verification is the remaining formal-axis target.

Theorem 116.8 (Permutation singleton FinSet identity classifier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{FinSet}^\uparrow}$ *presented by* $\mathsf{FinSetEnumCarrier}_{A,\sim_A}$ *and* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ *on a finite enumeration spine* xs*. Suppose the endpoint row is singleton at a displayed position* $s : \mathsf{Pos}(xs)$*, in the sense that every carried position of the finite-spine endpoint is that position:*

$$\forall i : \mathsf{Pos}(xs).\ i = s.$$

Let G *be any accepted carried* $\mathsf{Permutation}^\uparrow$ *graph row over this source certificate, with certified finite-position self-bijection*

$$\pi : \mathsf{Pos}(xs) \cong \mathsf{Pos}(xs)$$

read from its ledger as in Definition 116.2. Then G *is classified by the identity* $\mathsf{Permutation}^\uparrow$ *row. Equivalently, for every carried source endpoint, the action output of* G *and the action output of the certified identity row are compared by* $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$ *through the composition, inverse, and action row of Definition 116.4.*

Proof. Unpack the accepted graph row using Definition 116.2. Its visible graph, inverse graph, and inverse-witness rows are exactly the ledger reading of a certified finite-position self-bijection on $\mathsf{Pos}(xs)$, together with the pointwise carrier-and-classifier field for matched entries.

Fix a source position $i : \mathsf{Pos}(xs)$. The singleton hypothesis gives

$$i = s.$$

Applying the same hypothesis to the target position $\pi(i)$ gives

$$\pi(i) = s.$$

Thus i and $\pi(i)$ are the same displayed singleton position. The action of G at i therefore reads the same finite-spine endpoint as the identity action at i, namely the endpoint represented by s. The pointwise preservation field already attached to G supplies the carrier witness for that target entry and the comparison

$$\mathsf{entry}_{xs}(i) \sim_A \mathsf{entry}_{xs}(\pi(i)).$$

After substituting $i = s$ and $\pi(i) = s$, this is the reflexive singleton endpoint comparison read through the source classifier.

By Definition 112.26, the identity $\mathsf{Permutation}^{\uparrow}$ row is represented by the certified identity bijection on $\mathsf{Pos}(xs)$. Since both G and the identity row send every source position to the same singleton target position, their action outputs agree pointwise through $\mathsf{FinSetEnumClassifier}_{A,\sim_A}$. Applying the classifier obligation of Definition 116.3 repacks this pointwise comparison as the $\mathsf{Permutation}^{\uparrow}$ classifier relation between G and the identity row.

Finally, Definition 116.4 identifies the identity, inverse, and action rows as ledger readings of the same finite-position facts. The inverse witness of G therefore adds no second target position: $G^{-1} \circ G$ reads as the identity graph on the singleton endpoint, and the already established pointwise action comparison is precisely the classification of G with the identity row. □

Lean4 ✓ `BEDC.Derived.PermutationUp.PermutationBijectionSourceRow singleton identity classifier`

Chapter 117

A Concrete Naming Certificate for $\mathsf{SymGroup}^{\uparrow}$

The symmetric-group interface $\mathsf{SymGroup}^{\uparrow}$ packages the group structure on $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ (chapter 116) under composition. It is a canonical instance of $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and the substrate for representation-theoretic certificates.

117.1 Carrier definition

Theorem 117.1 (SymGroupUp carrier obligation)**.** *The* $\mathsf{SymGroup}^{\uparrow}$ *carrier is the certified permutation carrier over* $\mathsf{NameCert}_{\mathsf{Permutation}^{\uparrow}}$ *together with the* $\mathsf{Group}^{\uparrow}$ *source boundary of chapter 41. A carried row is therefore a* $\mathsf{Permutation}^{\uparrow}$ *source row whose visible operation is certified composition, whose displayed unit is the certified identity permutation, and whose displayed inverse is the certified inverse permutation. The carrier boundary records only the* BHist *graph rows, inverse rows, action row,* Pkg *provenance, and ledger entries already exposed by chapter 116; it does not expose a host enumeration or a host function as the group object.*

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupCarrier semantic name certificate`
Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupCarrier obligation`

Proof. Read the permutation source row from Definition 116.1. Its carrier consists of graph, inverse graph, inverse-witness, classifier, action, and ledger rows over the fixed $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ source.

The $\mathsf{Group}^{\uparrow}$ boundary from chapter 41 asks for a carried operation, unit, inverse, classifier, and ledger. Use Definition 116.4 to supply composition, identity, inverse, and action compatibility from the same certified finite-position rows.

Thus the $\mathsf{SymGroup}^{\uparrow}$ carrier is obtained by repacking the already certified $\mathsf{Permutation}^{\uparrow}$ carrier as a group-facing source row. All comparisons remain hsame and classifier transport through Pkg provenance, so no host function equality or quotient is introduced. □

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationCarrier group surface`
Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationCarrier carrier obligation`

117.2 The certificate

Theorem 117.2 (SymGroupUp classifier obligation)**.** *The* $\mathsf{SymGroup}^{\uparrow}$ *classifier is the permutation classifier transported through the group carrier of Definition 117.1. For carried permutation rows*

G, H, K, it must satisfy reflexivity on G, symmetry from $G \sim H$ to $H \sim G$, transitivity from $G \sim H$ and $H \sim K$ to $G \sim K$, and carrier transport across the $\mathsf{Group}^{\uparrow}$ *source rows for composition, identity, and inverse. These fields are exactly the* $\mathsf{Permutation}^{\uparrow}$ *classifier rows of Definition 116.3 read through the group-facing carrier boundary.*

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationClassifier stability`

Proof. Start with the pointwise action classifier of Definition 116.3. It compares two certified self-bijection rows by reading their action outputs through the fixed $\mathsf{FinSet}^{\uparrow}$ classifier.

Reflexivity, symmetry, and transitivity are inherited from the finite-set classifier rows named in chapter 116. Carrier transport is the same hsame and Pkg transport used to move graph and endpoint rows in Definition 116.3.

Repack those rows under the $\mathsf{Group}^{\uparrow}$ boundary named in chapter 41. Since the classifier observes only certified action outputs and package provenance, the construction never identifies host functions and never forms a quotient of presentations. □

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationCarrier classifier obligation`

Theorem 117.3 (SymGroupUp composition, inverse, and action obligations)**.** *For the carrier and classifier of Definition 117.1 and Definition 117.2, the certificate rows required for* $\mathsf{SymGroup}^{\uparrow}$ *are the following.*

1. *Composition closure: if G and H are carried permutations, then the composite $H \circ G$ is a carried permutation.*
2. *Inverse closure: if G is carried, then G^{-1} is carried.*
3. *Unit laws: the certified identity permutation is carried and classifies as the left and right unit for composition.*
4. *Associativity: the two certified composites $(K \circ H) \circ G$ and $K \circ (H \circ G)$ classify together.*
5. *Action compatibility: the action row of a composite classifies with the successive action rows, and the action row of the inverse is the inverse graph readback.*
6. *Exactness: graph coverage, inverse-witness coverage, and classifier transport are exactly the visible rows of the permutation source package.*
7. *Ledger boundary: enumeration choices, position names, order witnesses, probe reads, and package provenance remain in the ledger and are not promoted to public carrier data.*

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationCarrier composition inverse action obligations`

Proof. Composition closure, inverse closure, unit rows, and action compatibility are the four group-facing readings of Definition 116.4. The theorem already states these rows as finite-position bijection facts whose graph readings are carried by the $\mathsf{Permutation}^{\uparrow}$ ledger.

Associativity is read constructively from the same certified composition row: both parenthesized composites have the same sequence of finite-position graph reads, and the comparison is recorded as classifier transport over the BHist graph rows. The proof uses the $\mathsf{Group}^{\uparrow}$ source boundary of chapter 41 only to name the operation row, not to add a host-level equality of functions.

Exactness and the ledger boundary are inherited from Definition 116.1. Graph coverage, inverse coverage, and action readback stay visible; enumeration choices, order choices, position names, probe logs, and Pkg provenance stay in the ledger. Hence the certificate surface is constructive and introduces no quotient of permutation presentations. □

Theorem 117.4 (SymGroupUp public certificate surface)**.** *The carrier theorem of Definition 117.1, the classifier theorem of Definition 117.2, and the composition, inverse, and action theorem of Definition 117.3 assemble the public* $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$ *surface. The exported surface consists of the certified permutation carrier, the group-facing classifier, composition, inverse, unit and action compatibility, exactness, and ledger boundary. It exposes no host function equality and forms no quotient of presentations.*

Lean4 ✓ `BEDC.Derived.SymGroupUp.SymGroupPermutationCarrier public certificate surface`

Proof. Project the carrier surface from Definition 117.1. This supplies the certified permutation rows, the $\mathsf{Group}^{\uparrow}$ source boundary, the displayed unit, the displayed inverse, and the visible graph, action, package, and ledger rows.

Project the classifier surface from Definition 117.2. Its reflexivity, symmetry, transitivity, and carrier-transport rows are exactly the permutation classifier rows read through the group-facing carrier.

Project the operation rows from Definition 117.3. These give composition closure, inverse closure, unit laws, associativity, action compatibility, exactness, and the ledger boundary. Repacking these three projected surfaces gives the exported $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$ certificate, with every comparison still recorded through hsame, classifier transport, and Pkg provenance rather than host function equality or quotienting. □

Closure status for $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$**.**

Constructive story: The symmetric group is obtained by repacking the already certified $\mathsf{Permutation}^{\uparrow}$ carrier as a $\mathsf{Group}^{\uparrow}$ surface under composition. A carried element is still a finite-position self-bijection row with its graph, inverse graph, action, inverse-witness rows, package provenance, and ledger entries; rows lacking the permutation certificate are not admitted as group elements. The classifier is the permutation classifier read through the group-facing boundary, so two group elements classify together when their certified actions agree pointwise on the finite-set entries, not when host functions are equal. The source is the $\mathsf{Permutation}^{\uparrow}$ public surface plus the group operation boundary, the pattern records identity permutation, composition, inverse, associativity, unit laws, action compatibility, and exactness, and the ledger keeps enumeration choices, position names, probe reads, and provenance as permutation ledger data. The five name-certificate obligations are supplied by the identity permutation for inhabitation, by reflexivity, symmetry, and transitivity inherited from the permutation classifier, and by composition and inverse transport rows that keep carried permutations inside the carrier. A carried permutation G lies in the signature gap of a $\mathsf{SymGroup}^{\uparrow}$ token p when the source-domain accepts the permutation and group-boundary rows, the signature relation pairs G with an answer group element and evidence ledger, and token introduction names that answer as p; unfolded, G is the certified action encoded by p with its identity, composition, and inverse ledgers present. The paper bridge reads this as the group of certified permutations of the finite spine and reads back only the same permutation rows and classifier evidence. The chapter registers this composition group construction under the name $\mathsf{SymGroup}^{\uparrow}$; group structure is a repackaging of permutation ledgers, not host symmetric-group equality, a quotient of presentations, representation theory, or an exposed order witness.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature surface consists of the carrier, classifier, composition, inverse, action, exactness, and ledger rows already assembled by the public certificate of Definition 117.4.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SymGroupUp.SymGroupCarrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not claim bridgeChecked verification for the exported $\mathsf{SymGroup}^{\uparrow}$ certificate surface.

Upgrade path: A checked standard bridge theorem is the remaining verification step beyond the audit-clean target named here.

Chapter 118

A Concrete Naming Certificate for $\mathsf{Graph}^{\uparrow}$

The graph interface $\mathsf{Graph}^{\uparrow}$ packages a vertex carrier from $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ (chapter 111) together with a binary edge relation. It supplies the substrate for $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$ (chapter 119) and for combinatorial structures such as posets viewed as DAGs.

118.1 Carrier definition

Definition 118.1 (Cont adjacency graph carrier)**.** *The concrete* $\mathsf{Graph}^{\uparrow}$ *source is the BEDC history carrier* BHist *restricted to* UnaryHistory*, with vertex classifier* hsame *on unary histories. For histories* h, k, g : BHist *define*

$$\mathsf{GraphEdge}(h, k, g) \iff \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}(h, k, g).$$

This predicate reads a BEDC continuation row as a directed adjacency edge from source vertex h *to target vertex* k*, with visible edge-result history* g*. The graph carrier template fields are:*

1. *the traditional target is a directed graph;*
2. *the BEDC source is* BHist *with* UnaryHistory *vertices and* Cont *rows;*
3. *the vertex carrier is* UnaryHistory*;*
4. *the vertex classifier is* hsame *restricted to unary histories;*
5. *the stability obligations are unary transport along* hsame *and* Cont *transport along* hsame *in all three displayed histories;*
6. *the ledger is the visible triple* (h, k, g) *together with the two unary endpoint witnesses and the* $\mathsf{Cont}(h, k, g)$ *row;*
7. *the core theorem list is endpoint coverage, edge classifier transport, cancellative row readback, unit rows, two-step path composition, and the assembled* $\mathsf{Graph}^{\uparrow}$ *obligation surface;*
8. *the exactness predicate is row inversion for* GraphEdge*;*
9. *the standard bridge is absent;*
10. *the status is paper proof target.*

The edge classifier between two rows is the componentwise relation

$$(h, k, g) \sim_{\mathsf{Graph}} (h', k', g') :\Longleftrightarrow \mathsf{hsame}(h, h') \wedge \mathsf{hsame}(k, k') \wedge \mathsf{hsame}(g, g').$$

Thus edge sameness is entirely visible in the BEDC ledger; no hidden graph structure participates in classification.

Lean4 (definition) `BEDC.Derived.GraphUp.GraphContEdge`

118.2 The certificate

Theorem 118.2 (Graph edge endpoint coverage)**.** *For histories $h, k, g : \mathsf{BHist}$, every row $\mathsf{GraphEdge}(h, k, g)$ exposes its carried source vertex, carried target vertex, and continuation witness:*

$$\mathsf{GraphEdge}(h, k, g) \Longrightarrow \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(k) \wedge \mathsf{Cont}(h, k, g).$$

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge endpoint coverage`

Proof. Unfold Definition 118.1. The graph edge predicate is defined as exactly the conjunction of the source unary witness, the target unary witness, and the continuation row. Projection from that conjunction gives the three displayed fields. □

Theorem 118.3 (Graph edge classifier transport)**.** *Let $h, k, g, h', k', g' : \mathsf{BHist}$. If $\mathsf{GraphEdge}(h, k, g)$ holds and*

$$\mathsf{hsame}(h, h') \wedge \mathsf{hsame}(k, k') \wedge \mathsf{hsame}(g, g')$$

holds, then unary-history transport on the endpoints and cont_hsame_transport *on the continuation row give*

$$\mathsf{GraphEdge}(h', k', g').$$

The original and transported rows are classified by $(h, k, g) \sim_{\mathsf{Graph}} (h', k', g')$.

Lean4 ✓ `BEDC.Derived.GraphUp.GraphEdge classifier transport`
Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge classifier transport`

Proof. Unpack $\mathsf{GraphEdge}(h, k, g)$ by Definition 118.1. This yields $\mathsf{UnaryHistory}(h)$, $\mathsf{UnaryHistory}(k)$, and $\mathsf{Cont}(h, k, g)$.

Unpack the classifier witnesses. Unary-history transport along $\mathsf{hsame}(h, h')$ and $\mathsf{hsame}(k, k')$ carries the two endpoint witnesses to $\mathsf{UnaryHistory}(h')$ and $\mathsf{UnaryHistory}(k')$.

Apply cont_hsame_transport to the three hsame witnesses and the row $\mathsf{Cont}(h, k, g)$. This produces $\mathsf{Cont}(h', k', g')$.

Repack the two transported unary witnesses and the transported continuation row by Definition 118.1. The final classifier claim is just the same three hsame witnesses read through the componentwise edge classifier. □

Theorem 118.4 (Graph Cont edge row inversion)**.** *Let $h, k, g, h', k', g' : \mathsf{BHist}$. For displayed graph rows:*

1. *if $\mathsf{GraphEdge}(h, k, g)$, $\mathsf{GraphEdge}(h', k, g)$, and the target/result pair is shared, then $\mathsf{hsame}(h, h')$ by* cont_right_cancel*;*
2. *if $\mathsf{GraphEdge}(h, k, g)$, $\mathsf{GraphEdge}(h, k', g)$, and the source/result pair is shared, then $\mathsf{hsame}(k, k')$ by* cont_left_cancel*;*

3. *if* $\mathsf{GraphEdge}(h, k, g)$ *and* $\mathsf{GraphEdge}(h, k, g')$ *share the same source and target, then* $\mathsf{hsame}(g, g')$ *by* cont_deterministic.

Proof. Unfold Definition 118.1 in each displayed row. The endpoint witnesses are retained as carrier data, and the continuation component is the BEDC kernel row $\mathsf{Cont}(-, -, -)$.

For the shared target/result clause, apply cont_right_cancel to the two exposed continuation rows. This reads back the two possible sources as hsame.

For the shared source/result clause, apply cont_left_cancel to the two exposed continuation rows. This reads back the two possible targets as hsame.

For the shared source/target clause, apply cont_deterministic to the two exposed continuation rows. This classifies the two visible edge results by hsame. □

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge row inversion`

Theorem 118.5 (Graph Cont unit loop laws)**.** *For every unary vertex* $h : \mathsf{BHist}$*, the graph relation contains the two displayed unit rows*

$$\mathsf{GraphEdge}(\mathsf{BHist.Empty}, h, h)$$

and

$$\mathsf{GraphEdge}(h, \mathsf{BHist.Empty}, h).$$

Moreover, a row $\mathsf{GraphEdge}(\mathsf{BHist.Empty}, h, g)$ *reads its result back to* h *by* cont_left_unit *and* cont_left_unit_iff*, and a row* $\mathsf{GraphEdge}(h, \mathsf{BHist.Empty}, g)$ *reads its result back to* h *by* cont_right_unit *and* cont_right_unit_iff*.*

Proof. Use the unary empty witness for BHist.Empty and the given unary witness for h. The left unit law cont_left_unit supplies $\mathsf{Cont}(\mathsf{BHist.Empty}, h, h)$, and the right unit law cont_right_unit supplies $\mathsf{Cont}(h, \mathsf{BHist.Empty}, h)$.

Repack each continuation row with its two unary endpoint witnesses by Definition 118.1. This gives the two displayed graph unit rows.

For readback, unfold an arbitrary left-unit or right-unit graph row to expose its continuation component. The corresponding iff theorem, cont_left_unit_iff or cont_right_unit_iff, identifies exactly the histories whose result is classified with h. □

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge unit loop`

Theorem 118.6 (Graph Cont two-step composition)**.** *Let* $h, k, l, hk, kl : \mathsf{BHist}$*. If* $\mathsf{GraphEdge}(h, k, hk)$ *and* $\mathsf{GraphEdge}(k, l, kl)$ *hold, then there are histories* $left, right : \mathsf{BHist}$ *such that*

$$\mathsf{GraphEdge}(hk, l, left),$$

$$\mathsf{GraphEdge}(h, kl, right),$$

and

$$\mathsf{hsame}(left, right).$$

Proof. Unfold the two input graph edges. This exposes unary witnesses for h, k, and l, together with continuation rows $\mathsf{Cont}(h, k, hk)$ and $\mathsf{Cont}(k, l, kl)$.

Unary closure for continuation results carries the first row to a unary witness for hk and the second row to a unary witness for kl. These witnesses make the two composite endpoints eligible for the graph carrier.

Apply cont_assoc_exists_hsame to the two continuation rows. It produces histories $left$ and $right$ with rows $\mathsf{Cont}(hk, l, left)$ and $\mathsf{Cont}(h, kl, right)$ and with $\mathsf{hsame}(left, right)$.

Repack the two produced continuation rows with the unary endpoint witnesses by Definition 118.1. The result is the displayed pair of two-step graph edges together with their hsame result comparison. □

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge two step composition`
Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge composition closure`

Theorem 118.7 (GraphUp NameCert surface)**.** *The carrier definition in Definition 118.1, together with Definition 118.2 and Definition 118.3, Definition 118.4, Definition 118.5, and Definition 118.6, assembles the paper-side* $\mathsf{Graph}^{\uparrow}$ *obligation surface: vertex carrier* UnaryHistory*, vertex classifier* hsame *restricted to unary histories, edge relation* GraphEdge*, endpoint coverage, edge transport, cancellative readback, unit rows, two-step path composition, and the visible* Cont *ledger. The standard bridge field is absent from this surface.*

Lean4 ✓ `BEDC.Derived.GraphUp.GraphCont namecert surface`
Lean4 ✓ `BEDC.Derived.GraphUp.GraphContNameCert surface`

Proof. The definition supplies the concrete BEDC source, the vertex carrier, the restricted vertex classifier, the edge relation, the visible ledger, and the declared absence of a standard bridge. It also names the exactness predicate as row inversion for GraphEdge.

Definition 118.2 gives that inversion principle in the direction used by downstream graph certificates: every edge row exposes both carried endpoints and its Cont witness.

Definition 118.3 supplies the stability surface. It transports the two endpoint carrier witnesses along hsame and transports the continuation row along cont_hsame_transport, then repackages the transported row as another GraphEdge witness.

Definition 118.4 supplies cancellative readback for rows sharing a target/result pair or a source/result pair, and determinacy for rows sharing a source/target pair.

Definition 118.5 supplies the two empty-endpoint unit rows and their result readback principles. Definition 118.6 supplies path composition by associativity of Cont up to hsame.

These clauses are precisely the listed $\mathsf{Graph}^{\uparrow}$ paper obligations, so the chapter exposes the carrier, classifier, edge, coverage, transport, readback, unit, composition, and ledger surface required by $\mathsf{Tree}^{\uparrow}$ and downstream combinatorial certificates. □

Theorem 118.8 (Graph empty-tail identity edge)**.** *For every history* $h : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(h)$*, then*

$$\mathsf{GraphEdge}(h, \mathsf{e}, h).$$

Its ten-field fit is: traditional target identity graph edge; BEDC source BHist*,* UnaryHistory*, and* Cont*; carrier* GraphEdge*; classifier componentwise* hsame*; endpoint fields the two displayed unary witnesses; stability inherited from* cont_hsame_transport*; ledger* (h, e, h)*; exactness by endpoint coverage and continuation determinacy; standard bridge absent; status proof target.*

Proof. Use $\mathsf{UnaryHistory}(h)$ for the source endpoint. The empty history is a unary history by the empty unary constructor, so it supplies the target endpoint.

The continuation row is the right-unit row $\mathsf{Cont}(h, \mathsf{e}, h)$, also named cont_right_unit. Repacking these three fields through Definition 118.1 gives the displayed GraphEdge row. □

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge empty tail identity`

Theorem 118.9 (Graph edge result determinacy)**.** *For histories* $h, k, g, g' : \mathsf{BHist}$*, if* $\mathsf{GraphEdge}(h, k, g)$ *and* $\mathsf{GraphEdge}(h, k, g')$*, then*

$$\mathsf{hsame}(g, g').$$

Its ten-field fit is: traditional target graph edge-result determinacy; BEDC source BHist*,* UnaryHistory*, and* Cont*; carrier* GraphEdge*; classifier componentwise* hsame*; endpoint fields read from the two edge rows; stability inherited from continuation transport; ledger the two rows* (h, k, g) *and* (h, k, g')*; exactness by endpoint coverage and continuation determinacy up to* hsame*; standard bridge absent; status proof target.*

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge result determinacy`

Proof. Apply Definition 118.2 to both edge rows. This projects the two continuation witnesses $\mathsf{Cont}(h, k, g)$ and $\mathsf{Cont}(h, k, g')$ from the graph carrier.

Apply cont_deterministic, equivalently cont_determinacy_up_to_hsame_spine, to the two continuation rows with the same source and the same visible target. The resulting classification is exactly $\mathsf{hsame}(g, g')$. □

Theorem 118.10 (Graph visible-tail step closure)**.** *For histories* $h, k, g : \mathsf{BHist}$*, suppose* $\mathsf{GraphEdge}(h, k, g)$ *and suppose the visible-tail constructors preserve unary history at the target and result positions:*

$$\mathsf{UnaryHistory}(\mathsf{E0}(k)) \wedge \mathsf{UnaryHistory}(\mathsf{E0}(g)) \wedge \mathsf{UnaryHistory}(\mathsf{E1}(k)) \wedge \mathsf{UnaryHistory}(\mathsf{E1}(g)).$$

Then

$$\mathsf{GraphEdge}(h, \mathsf{E0}(k), \mathsf{E0}(g)) \wedge \mathsf{GraphEdge}(h, \mathsf{E1}(k), \mathsf{E1}(g)).$$

Its ten-field fit is: traditional target visible-tail graph closure; BEDC source BHist*,* UnaryHistory*, and* Cont*; carrier* GraphEdge*; classifier componentwise* hsame*; endpoint fields the source row plus the supplied visible-tail unary rows; stability inherited from the two continuation step rules; ledger the two stepped triples; exactness by endpoint coverage and continuation determinacy; standard bridge absent; status proof target.*

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContEdge visible tail step closure`

Proof. First project the original edge by Definition 118.2. This gives $\mathsf{UnaryHistory}(h)$ and the continuation row $\mathsf{Cont}(h, k, g)$.

For the zero-tail branch, use the supplied unary rows $\mathsf{UnaryHistory}(\mathsf{E0}(k))$ and $\mathsf{UnaryHistory}(\mathsf{E0}(g))$. Apply the zero continuation step rule to $\mathsf{Cont}(h, k, g)$, obtaining $\mathsf{Cont}(h, \mathsf{E0}(k), \mathsf{E0}(g))$. Repack these three fields as $\mathsf{GraphEdge}(h, \mathsf{E0}(k), \mathsf{E0}(g))$.

For the one-tail branch, use the supplied unary rows $\mathsf{UnaryHistory}(\mathsf{E1}(k))$ and $\mathsf{UnaryHistory}(\mathsf{E1}(g))$. Apply the one continuation step rule to $\mathsf{Cont}(h, k, g)$, obtaining $\mathsf{Cont}(h, \mathsf{E1}(k), \mathsf{E1}(g))$. Repack these three fields as $\mathsf{GraphEdge}(h, \mathsf{E1}(k), \mathsf{E1}(g))$. □

Theorem 118.11 (Graph Cont three-step path reassociation)**.** *Let* $h, k, l, m, hk, kl, lm : \mathsf{BHist}$*. In the concrete* $\mathsf{Graph}^{\uparrow}$ *carrier of Definition 118.1, if*

$$\mathsf{GraphEdge}(h, k, hk), \qquad \mathsf{GraphEdge}(k, l, kl), \qquad \mathsf{GraphEdge}(l, m, lm)$$

hold, then there are histories $hkl, klm, left, right : \mathsf{BHist}$ *such that*

$$\mathsf{GraphEdge}(hk, l, hkl), \qquad \mathsf{GraphEdge}(hkl, m, left),$$

$$\mathsf{GraphEdge}(k, lm, klm), \qquad \mathsf{GraphEdge}(h, klm, right),$$

and

$$\mathsf{hsame}(left, right).$$

Thus the visible endpoints of the left-associated and right-associated three-step graph paths are classified by hsame.

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContPath reassociation`

Proof. Apply Definition 118.6 to the first two input rows. Choose witnesses hkl and hkl_R with

$$\mathsf{GraphEdge}(hk, l, hkl), \qquad \mathsf{GraphEdge}(h, kl, hkl_R), \qquad \mathsf{hsame}(hkl, hkl_R).$$

Retain the row $\mathsf{GraphEdge}(hk, l, hkl)$ as the left middle edge.

Apply Definition 118.6 to the second and third input rows. Choose witnesses klm_L and klm with

$$\mathsf{GraphEdge}(kl, m, klm_L), \qquad \mathsf{GraphEdge}(k, lm, klm), \qquad \mathsf{hsame}(klm_L, klm).$$

Retain the row $\mathsf{GraphEdge}(k, lm, klm)$ as the right middle edge.

Apply Definition 118.6 to $\mathsf{GraphEdge}(hk, l, hkl)$ and the third input row. This gives witnesses $left$ and mid_L such that

$$\mathsf{GraphEdge}(hkl, m, left), \qquad \mathsf{GraphEdge}(hk, lm, mid_L), \qquad \mathsf{hsame}(left, mid_L).$$

Apply Definition 118.6 to the first input row and $\mathsf{GraphEdge}(k, lm, klm)$. This gives witnesses mid_R and $right$ such that

$$\mathsf{GraphEdge}(hk, lm, mid_R), \qquad \mathsf{GraphEdge}(h, klm, right), \qquad \mathsf{hsame}(mid_R, right).$$

The two middle rows $\mathsf{GraphEdge}(hk, lm, mid_L)$ and $\mathsf{GraphEdge}(hk, lm, mid_R)$ have the same displayed source and target. By Definition 118.9,

$$\mathsf{hsame}(mid_L, mid_R).$$

Transitivity of hsame combines $\mathsf{hsame}(left, mid_L)$, $\mathsf{hsame}(mid_L, mid_R)$, and $\mathsf{hsame}(mid_R, right)$ to give

$$\mathsf{hsame}(left, right).$$

The retained graph rows are exactly the displayed left-associated and right-associated path rows. □

Theorem 118.12 (GraphUp Cont public interface bridge)**.** *The public graph-interface bridge for Definition 118.1 interprets every row* $\mathsf{GraphEdge}(h, k, g)$ *as a directed edge of the exported* $\mathsf{Graph}^{\uparrow}$ *interface from source vertex* h *to target vertex* k *with visible edge result* g*. Public equality of exported graph edges reflects back to the componentwise BEDC classifier*

$$\mathsf{hsame}(h, h') \wedge \mathsf{hsame}(k, k') \wedge \mathsf{hsame}(g, g')$$

on source, target, and result.

The bridge fields are exactly the $\mathsf{Graph}^{\uparrow}$ *NameCert surface of Definition 118.7, endpoint coverage from Definition 118.2, classifier transport from Definition 118.3, row inversion from Definition 118.4, the unit loop laws from Definition 118.5, two-step composition from Definition 118.6, and three-step reassociation from Definition 118.11.*

Lean4 ✓ `BEDC.Derived.GraphUp.GraphContPublicInterface bridge`

Proof. Project the carrier, classifier, edge predicate, visible ledger, and declared surface fields from Definition 118.7. The carrier definition reads a public graph row as precisely the displayed $\mathsf{GraphEdge}(h, k, g)$ datum, so interpretation into the exported $\mathsf{Graph}^{\uparrow}$ interface is direct projection from the row.

For endpoint soundness, apply Definition 118.2. It exposes the source unary witness, target unary witness, and continuation row carried by every public edge, so the interface endpoints are exactly the two displayed histories.

For equality reflection, use the componentwise edge classifier from Definition 118.1. Classifier transport is supplied by Definition 118.3; row inversion is supplied by Definition 118.4, which reads shared public rows back to hsame on the source, target, or result component as appropriate.

For the graph operations, Definition 118.5 supplies the public empty-endpoint unit rows, Definition 118.6 supplies public path composition, and Definition 118.11 supplies reassociation of three displayed paths up to hsame. These projections are the advertised public graph-interface bridge fields. □

Closure status for $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$.

Constructive story: A graph in this concrete surface uses unary histories as vertices and reads a directed edge from the kernel continuation row: $\mathsf{GraphEdge}(h, k, g)$ means that h and k are unary and $\mathsf{Cont}(h, k, g)$ is displayed. A row is excluded unless both endpoints are carried vertices and the continuation result is part of the same visible triple, while edge classification is componentwise hsame on source, target, and result. The source is the BHist unary-vertex carrier with Cont rows, the pattern records endpoint coverage, row inversion, unit loops, two-step composition, and three-step reassociation, and the ledger keeps exactly the triple (h, k, g) with the two unary endpoint witnesses and continuation witness. The five name-certificate obligations are witnessed by empty-endpoint unit rows for inhabitation, by componentwise reflexivity, symmetry, and transitivity of hsame, and by transport of unary endpoints and continuation rows along the classifier. An edge row e lies in the signature gap of a $\mathsf{Graph}^{\uparrow}$ token p when the source-domain accepts its unary endpoints and continuation triple, the signature relation pairs that triple with an answer edge and evidence ledger, and token introduction names that answer as p; unfolded, the source, target, and result histories are exactly the componentwise edge data encoded by the token. The public bridge reads each visible triple as a directed graph edge and reads equality of exported edges back to the same componentwise BEDC classifier, with no separate standard graph API claimed. The chapter registers this continuation-edge construction under the name $\mathsf{Graph}^{\uparrow}$; graph structure is a ledger of unary vertices and Cont rows, not a host finite graph, connectivity package, homomorphism theory, or hidden adjacency object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The $\mathsf{Graph}^{\uparrow}$ treatment binds endpoint coverage, classifier transport, row inversion, unit loop laws, two-step composition, three-step reassociation, and the public interface bridge for GraphEdge rows over UnaryHistory vertices with hsame classification.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.GraphUp.GraphContNameCert\_surface`

Bridge status: paperBridge

Not claimed: Finite graph APIs, connectivity, acyclicity, graph homomorphisms, standard graph APIs, or machine-checked bridge closure.

Upgrade path: A machine-checked bridge grade requires a checked StdBridge target for this public interface, independently of the paper bridge recorded here.

Chapter 119

A Concrete Naming Certificate for $\mathsf{Tree}^{\uparrow}$

The tree interface $\mathsf{Tree}^{\uparrow}$ packages a connected acyclic $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118) together with a designated root or root-free witness. It is the constructive substrate for inductive carriers and for the syntactic representation of derivations.

119.1 Carrier definition

Theorem 119.1 (TreeUp graph-source carrier obligation)**.** *The carrier for* $\mathsf{Tree}^{\uparrow}$ *is a* BHist *predicate read through the existing* $\mathsf{Graph}^{\uparrow}$ *source. An accepted tree history exposes a carried* $\mathsf{Graph}^{\uparrow}$ *source row, a finite vertex-and-edge incidence ledger over* BHist *endpoints, and a classifier comparison inherited from* hsame *and the* $\mathsf{Graph}^{\uparrow}$ *classifier rows. The carrier does not identify host graphs by equality; every comparison is recorded as a displayed classifier row over the accepted histories and their incidence ledger.*

Proof. Unpack an accepted tree history as a graph-source presentation. The first projection is the $\mathsf{Graph}^{\uparrow}$ carrier row, so each visible vertex and edge is read as a carried BHist endpoint rather than as a host-level graph object.

The second projection is the finite incidence ledger. It lists the displayed vertex endpoints, displayed edge endpoints, and the source-target incidence rows that attach each edge to carried vertices. Finiteness belongs to this ledger, so no quotient over ambient graphs is introduced.

The classifier projection is inherited rowwise. Endpoint movement is componentwise hsame, and graph-source movement is the comparison supplied by $\mathsf{Graph}^{\uparrow}$. Transporting the finite incidence ledger along those rows gives the same accepted tree carrier predicate at the target history, with no appeal to equality of host graphs. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeGraphSource carrier obligation`

Definition 119.2 (TreeUp BHist carrier)**.** *The carrier of* $\mathsf{Tree}^{\uparrow}$ *is the* BHist *endpoint row whose visible package consists of a* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *edge/vertex package, the* GraphContEdge *rows witnessing the graph edges, a connectedness row, an acyclicity row, and one branch of a root classifier. The rooted branch records a root witness carried by the same* Pkg *provenance as the displayed graph package; the root-free branch records a witness that the same visible package has no designated root.*

Two such rows are carrier-comparable only through the visible graph hsame*-transport, the connectedness and acyclicity rows, and the chosen rooted or root-free witness row. Hidden graph routes,*

hidden choices of root, and hidden proofs of connectedness or acyclicity are ledger data, not carrier equality.

Lean4 ✓ `BEDC.Derived.TreeUp.TreeUp StdBridge`

Theorem 119.3 (TreeUp carrier obligation)**.** *The* $\mathsf{Tree}^{\uparrow}$ *carrier is obligated to expose exactly the* BHist *endpoint of Definition 119.2, the* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *source package, the* GraphContEdge *edge rows, the connectedness row, the acyclicity row, and the rooted or root-free witness row. Its ten-field fit is: traditional target tree carrier; BEDC source* BHist, Cont, hsame, Pkg, *and* NameCert*; carrier the displayed endpoint row; classifier inherited from graph-edge transport plus the three tree rows; stability by transport of the visible rows; ledger the hidden route, root-choice, connectedness, and acyclicity witnesses; exactness by visible row readback; standard bridge absent; status obligation target.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier obligation rows`

Proof. Use the carrier field of the roadmap ten-field template. The source row is the displayed $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ package, so every visible edge is already a GraphContEdge row over BHist and Cont.

Attach the connectedness, acyclicity, and root branch rows to that same Pkg provenance. The resulting endpoint is a tree row before any syntactic derivation chapter consumes it, because all carrier fields are visible BEDC rows and all non-visible choices are assigned to the ledger field. □

Theorem 119.4 (TreeUp classifier obligation)**.** *The* $\mathsf{Tree}^{\uparrow}$ *classifier relates two carried endpoints precisely when their* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *graph packages are transported by graph-edge* hsame*, their* GraphContEdge *rows match under that transport, and their connectedness, acyclicity, and root-branch rows match over the transported* Pkg *provenance. It does not identify trees by host graph isomorphism or by unrecorded route choice.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier classifier transport`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier classifier transport surface`

Proof. Start with the classifier field of the roadmap ten-field template and the graph source certificate. Transport the visible edge and vertex package along the graph hsame relation, carrying each GraphContEdge row by its Cont transport surface.

It remains to compare the tree-only rows. The connectedness row, the acyclicity row, and the rooted or root-free branch row are checked after the Pkg provenance has been transported, so the classifier is still a BEDC classifier on named histories rather than an external graph quotient. □

Theorem 119.5 (TreeUp root witness obligation)**.** *For each* $\mathsf{Tree}^{\uparrow}$ *carrier row, the root component must split constructively into exactly one displayed branch: either a rooted branch carrying a root witness compatible with the* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *vertex row, or a root-free branch carrying a witness that no vertex row in the same visible package has been designated as root. Both branches are transported by the same* hsame *and* Pkg *data as the graph package.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier root branch transport`

Proof. Proceed by case analysis on the root branch recorded in the carrier row.

In the rooted case, read the vertex witness from the visible $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ package and check that its Pkg provenance is the one used by the tree endpoint. Transport along hsame carries the root witness with the transported vertex row.

In the root-free case, keep the same graph package and read the negative root row as a visible witness attached to that package. Its transport uses the same hsame and Pkg data, so no hidden root selection can enter the carrier classifier. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeVisibleCarrier root witness obligation`

Theorem 119.6 (Tree frontier readback as a List spine)**.** *For any accepted* $\mathsf{Tree}^{\uparrow}$ *carrier whose visible* $\mathsf{Graph}^{\uparrow}$ *package is finite and rooted, the frontier leaves admit a* $\mathsf{List}^{\uparrow}$ *finite-spine readback. The entries of that spine are exactly the leaf endpoint rows transported by the* $\mathsf{Tree}^{\uparrow}$ *classifier, with incidence supplied by the* $\mathsf{Graph}^{\uparrow}$ *rows and finite-spine representation supplied by* $\mathsf{List}^{\uparrow}$*.*

Proof. Use Definition 119.3 to read the accepted carrier as a visible finite graph package with GraphContEdge incidence rows, and use Definition 119.5 to select the rooted branch. The $\mathsf{Graph}^{\uparrow}$ incidence rows identify the displayed children of each node, while the $\mathsf{List}^{\uparrow}$ finite-spine representation supplies the target format for the frontier.

Induct over the displayed finite tree packet. In the leaf case, no displayed outgoing incidence row remains, so the frontier readback is the singleton $\mathsf{List}^{\uparrow}$ spine whose entry is the leaf endpoint row.

In the branch case, apply the induction hypothesis to the left and right displayed subpackets. Concatenate the two frontier spines through the $\mathsf{List}^{\uparrow}$ spine route, preserving the left-to-right branch classifier rows as ledger data. The $\mathsf{Tree}^{\uparrow}$ classifier transports each leaf endpoint componentwise by hsame, so the concatenated spine entries are exactly the transported leaf endpoint rows.

Thus every visible frontier leaf contributes one displayed $\mathsf{List}^{\uparrow}$ spine entry, and every entry of the spine comes from one such leaf endpoint. No host traversal order or host tree datatype is used. □

Theorem 119.7 (TreeUp stability and ledger obligation)**.** *The* $\mathsf{Tree}^{\uparrow}$ *stability ledger consists exactly of the hidden graph route data, the hidden root-choice data, and the hidden connectedness and acyclicity witnesses used to justify the visible rows. Transport stability may consume those ledger entries, but the public carrier and classifier expose only* BHist *endpoints,* GraphContEdge *rows,* Cont *rows,* hsame *transport,* Pkg *provenance, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistObligationCarrier stability ledger obligation`

Proof. Use the stability and ledger fields of the roadmap ten-field template. The visible surface is fixed by the graph package, edge rows, connectedness row, acyclicity row, and root branch, so the remaining data can only record how those rows were obtained.

The hidden graph route supports connectedness, the hidden root-choice data supports the rooted branch when present, and the hidden acyclicity and connectedness witnesses justify the two graph-shape rows. Because each entry is consumed only to transport or justify a visible row, the ledger is exact and no additional hidden field is part of carrier equality. □

Theorem 119.8 (TreeUp exactness obligation)**.** *The visible* $\mathsf{Tree}^{\uparrow}$ *package is exact in three senses. Coverage says every accepted tree endpoint exposes a* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *package, its* GraphContEdge *rows, connectedness, acyclicity, and one root branch. Soundness says any endpoint exposing those rows is a valid* $\mathsf{Tree}^{\uparrow}$ *carrier only under the classifier of Definition 119.4. Completeness says every visible connected acyclic graph package with either a root witness or a root-free witness is represented by a* $\mathsf{Tree}^{\uparrow}$ *carrier row.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeDerivationSpineLedger exactness rows`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier exactness rows`

Proof. For coverage, invert the carrier row of Definition 119.2. The inversion reads back the $\mathsf{NameCert}_{\mathsf{Graph}^\uparrow}$ package, each GraphContEdge row, the connectedness row, the acyclicity row, and the root branch.

For soundness, apply the classifier obligation after the visible rows have been read back. The only admissible identifications are the transported graph edge rows together with matching connectedness, acyclicity, and root-branch rows.

For completeness, begin with a visible connected acyclic graph package and a chosen root branch. Repack those rows through BHist, Pkg, and NameCert, placing route and witness choices in the ledger, to obtain the required $\mathsf{Tree}^\uparrow$ carrier endpoint. □

119.2 The certificate

Theorem 119.9 (TreeUp connectedness obligation)**.** *For every accepted* $\mathsf{Tree}^\uparrow$ *history and every carried endpoint displayed by its* $\mathsf{Graph}^\uparrow$ *source, the certificate supplies a finite* BHist *path row through the* $\mathsf{Graph}^\uparrow$ *edge ledger from that endpoint to the selected root component. The path row records each intermediate endpoint, each traversed edge incidence, and the classifier comparison that carries the path under* hsame *transport.*

Proof. Read the endpoint from the graph-source carrier row supplied by Definition 119.1. The finite incidence ledger gives the only admissible edges and their source-target endpoint rows.

The connectedness witness is a finite path ledger indexed by those admissible edges. Its first endpoint is the displayed carried endpoint, its last endpoint lies in the selected root component, and every adjacent step is justified by one incidence row from the $\mathsf{Graph}^\uparrow$ ledger.

Transport the path by applying componentwise hsame to the endpoint rows and the inherited $\mathsf{Graph}^\uparrow$ classifier to the edge rows. The transported ledger has the same finite shape, so it remains a path from the transported endpoint to the transported root component. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeGraphSource connected root path readback`

Theorem 119.10 (TreeUp acyclicity obligation)**.** *For an accepted* $\mathsf{Tree}^\uparrow$ *history, every closed finite edge walk in the* $\mathsf{Graph}^\uparrow$ *edge ledger collapses under the* $\mathsf{Tree}^\uparrow$ *classifier to the trivial endpoint row. Equivalently, the certificate records that the only closed walk recognized by the tree classifier is the displayed stationary walk at its endpoint, and this recognition is stable under* hsame *transport.*

Proof. Start with a closed finite edge walk in the incidence ledger of Definition 119.1. Each step is a $\mathsf{Graph}^\uparrow$ edge row whose endpoints are carried BHist histories, and the closing condition is itself a displayed classifier row.

Apply the acyclicity witness carried by the $\mathsf{Tree}^\uparrow$ certificate. It reduces the closed walk ledger to the stationary endpoint row in the $\mathsf{Tree}^\uparrow$ classifier, so the edge sequence contributes no nontrivial cycle to the accepted carrier.

Stability follows by transporting the whole walk ledger componentwise through hsame and through the inherited $\mathsf{Graph}^\uparrow$ classifier. The acyclicity witness reads the transported closed walk and again returns the stationary endpoint row at the transported endpoint. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier closed walk tail empty`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistObligationCarrier acyclic empty transport`

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistObligationCarrier acyclic unit loop exactness`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeGraphContEdge closed walk unit collapse`

Theorem 119.11 (TreeUp root-or-root-free witness obligation)**.** *The* $\mathsf{Tree}^{\uparrow}$ *certificate supplies either a displayed root endpoint carried by the* $\mathsf{Graph}^{\uparrow}$ *source or a root-free witness whose classifier is stable under* hsame*. In the rooted case the root endpoint is one of the carried vertices used by the connectedness path rows. In the root-free case the witness records that connectedness is read componentwise without selecting a distinguished endpoint, and this record transports along the same* $\mathsf{Tree}^{\uparrow}$ *classifier.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeRootWitness spine closed boundary`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeRootBranch empty root witness readback`

Proof. Inspect the root row of the certificate. If the row is rooted, it displays a BHist endpoint and a proof that this endpoint is carried by the $\mathsf{Graph}^{\uparrow}$ source ledger. The connectedness paths then target the component containing that displayed root.

If the row is root-free, the certificate instead displays a witness that no distinguished endpoint is part of the public tree data. Connectedness is then read by component rows, and the witness itself is attached to the $\mathsf{Tree}^{\uparrow}$ classifier rather than to a host graph equality.

In both cases, transport is rowwise. The rooted endpoint moves by hsame and the inherited graph classifier, while the root-free witness moves by the stability field attached to the $\mathsf{Tree}^{\uparrow}$ classifier. Thus the certificate retains the same rooted or root-free form after transport. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeRootOrRootFreeWitness closed walk boundary`

Theorem 119.12 (TreeUp NameCert obligation surface)**.** *The carrier row of Definition 119.1, the connectedness row of Definition 119.9, the acyclicity row of Definition 119.10, and the root-or-root-free row of Definition 119.11 assemble the paper-side* $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$ *obligation surface. The surface names the stability, ledger, exactness, and dependency fields required for the certified tree interface: stability is* hsame *and inherited* $\mathsf{Graph}^{\uparrow}$ *transport; ledger data are finite vertex, edge, incidence, path, closed-walk, and root rows; exactness says every public tree row is read from those ledgers; and dependencies are only the* BHist *substrate,* hsame*, and the* $\mathsf{Graph}^{\uparrow}$ *source certificate.*

Proof. Begin with the graph-source carrier obligation. It supplies the accepted BHist predicate, the inherited $\mathsf{Graph}^{\uparrow}$ classifier, and the finite incidence ledger for visible vertices and edges.

Add the connectedness obligation. Its finite path rows show that every carried endpoint reaches the selected root component through the same edge ledger, and its transport clause supplies the corresponding stability row.

Add the acyclicity and root-or-root-free obligations. Acyclicity gives the closed-walk exactness clause, while the final witness records whether the certificate is rooted or root-free and how that row transports under hsame.

These projections cover the named carrier, classifier, stability, ledger, exactness, and dependency rows. Since every row is read from BHist, hsame, and the $\mathsf{Graph}^{\uparrow}$ source certificate, the assembled surface does not require host graph equality or any additional bridge field. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier semantic name certificate`

Theorem 119.13 (TreeUp carrier and classifier obligations)**.** *The* $\mathsf{Tree}^{\uparrow}$ *carrier data refine the* $\mathsf{Graph}^{\uparrow}$ Cont*-adjacency carrier of Definition 118.1 to displayed finite* Cont*-path rows over* BHist

whose vertices satisfy UnaryHistory*. Its classifier is inherited from the* $\mathsf{Graph}^{\uparrow}$ *endpoint/path classifier and is transported componentwise by* hsame*: two displayed tree-path rows are admissibly compared only when their source vertex, target vertex, visible path ledger, and every exposed* GraphEdge *step are identified through the same* hsame *classifier used by* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.TreeUp.TreeDerivationSpineLedger steps unary`

Proof. The obligation is the carrier-and-classifier portion of the roadmap ten-field template.

1. Traditional target: a finite tree path carrier for $\mathsf{Tree}^{\uparrow}$.
2. BEDC source: BHist with UnaryHistory vertices, Cont rows, and the $\mathsf{Graph}^{\uparrow}$ NameCert edge predicate.
3. Carrier: displayed finite rows of GraphEdge from Definition 118.1, read as finite Cont-path ledgers.
4. Classifier: the $\mathsf{Graph}^{\uparrow}$ endpoint/path classifier, transported componentwise by hsame on vertices, edge results, and path ledgers.
5. Stability: transport of every displayed Cont row through the hsame transport surface already carried by $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$.
6. Ledger: the source and target histories, the finite list of exposed GraphEdge rows, and the visible path-result history.
7. Core theorem list: carrier refinement, classifier inheritance, and path-row transport.
8. Exactness: a carried tree path is exactly its finite $\mathsf{Graph}^{\uparrow}$ row display over BHist.
9. Standard bridge: none.
10. Status: paper obligation.

Thus $\mathsf{Tree}^{\uparrow}$ does not introduce a second carrier beside $\mathsf{Graph}^{\uparrow}$. It selects the finite displayed Cont-path rows and reads their classifier through the same hsame-based NameCert surface. □

Theorem 119.14 (TreeUp connectedness and acyclicity obligations)**.** *For* $\mathsf{Tree}^{\uparrow}$*, connectedness is the obligation to supply, for every pair of carried vertices, a finite* GraphEdge *path package between them. Acyclicity is the obligation that every nontrivial closed finite path row reads through* hsame *as impossible at the tree classifier: any displayed closed path must collapse to the trivial unit loop supplied by* $\mathsf{Graph}^{\uparrow}$*. The connected and acyclic rows are scoped by the* $\mathsf{Graph}^{\uparrow}$ *unit-loop laws, the two-step composition surface, and the three-step path reassociation surface.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier closed path unit loop`

Proof. The obligation is the path-law portion of the roadmap ten-field template.

1. Traditional target: connected acyclic graph data.
2. BEDC source: BHist, UnaryHistory, Cont, and the $\mathsf{Graph}^{\uparrow}$ NameCert rows.
3. Carrier: the finite GraphEdge path packages named in Definition 119.13.
4. Classifier: hsame on endpoints and visible path ledgers, inherited from the $\mathsf{Graph}^{\uparrow}$ endpoint/path classifier.
5. Stability: path extension, transport, and reassociation through hsame.

6. Ledger: the chosen finite path between two carried vertices, or the displayed closed path whose nontriviality is being tested.
7. Core theorem list: connected path existence, unit-loop collapse, two-step composition, and three-step reassociation.
8. Exactness: the connectedness row exposes exactly a finite GraphEdge path, and the acyclicity row rejects exactly the nontrivial closed path rows.
9. Standard bridge: none.
10. Status: paper obligation.

The unit case is read through Definition 118.5. Composite paths are assembled through Definition 118.6, and adjacent reassociations are compared through Definition 118.11.

The acyclicity obligation then says that any closed finite row whose source and target are identified by hsame must be the unit-loop row in the displayed $\mathsf{NameCert}_{\mathsf{Graph}^\uparrow}$ ledger. Every nontrivial closed row is therefore outside the accepted $\mathsf{Tree}^\uparrow$ certificate surface. □

Theorem 119.15 (TreeUp root, ledger, and exactness obligations)**.** *The* $\mathsf{Tree}^\uparrow$ *certificate must expose exactly the following rows: a root-or-root-free witness, a syntactic-representation carrier row, the finite path witnesses of Definition 119.14, the acyclicity witnesses of Definition 119.14, and the dependency rows inherited from* $\mathsf{Graph}^\uparrow$*. These rows are precisely the ledger and exactness surface for* $\mathsf{Tree}^\uparrow$*: every accepted tree certificate is read back to those displayed rows, and every displayed row is classified by the* $\mathsf{Graph}^\uparrow$ hsame *endpoint/path classifier over* BHist *and* UnaryHistory*.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeRootLedger derivation spine boundary`

Proof. The obligation is the provenance-and-exactness portion of the roadmap ten-field template.

1. Traditional target: a rooted or root-free tree certificate with syntactic representation.
2. BEDC source: BHist, UnaryHistory, Cont, and the dependency surface of $\mathsf{NameCert}_{\mathsf{Graph}^\uparrow}$.
3. Carrier: the finite tree-path carrier of Definition 119.13, augmented by a root-or-root-free witness and a syntactic-representation carrier row.
4. Classifier: the $\mathsf{Graph}^\uparrow$ classifier transported through hsame on roots, endpoints, path rows, and representation ledgers.
5. Stability: transport of root status, path witnesses, acyclicity witnesses, and representation rows along the inherited classifier.
6. Ledger: the root witness, representation row, connectedness paths, acyclicity witnesses, and dependency rows.
7. Core theorem list: root coverage, root-free coverage, ledger readback, path exactness, acyclicity exactness, and dependency exactness.
8. Exactness: no additional tree datum is accepted beyond the displayed ledger rows, and each displayed ledger row is sufficient for the certificate boundary.
9. Standard bridge: none.
10. Status: paper obligation.

The root-or-root-free witness determines the public orientation of the tree carrier, while the syntactic-representation carrier row records the derivation syntax surface that consumes $\mathsf{Tree}^{\uparrow}$.

The path witnesses and acyclicity witnesses are exactly the rows named in Definition 119.14. The remaining dependency rows are the $\mathsf{Graph}^{\uparrow}$ carrier, classifier, unit-loop, two-step composition, and three-step reassociation rows. Repacking this list gives the full $\mathsf{Tree}^{\uparrow}$ ledger, and exactness says that the certificate contains no hidden data outside this displayed hsame-classified NameCert boundary. □

Definition 119.16 (Tree obligation surface)**.** *Fix a BEDC source history sort* BHist *and a graph certificate source* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *as in chapter 118. A concrete* $\mathsf{Tree}^{\uparrow}$ *obligation record consists of the following rows on histories:*

1. $\mathsf{TreeCarrier}(h)$ *is the* BHist *predicate saying that* h *carries a* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *graph row, a connectedness row, an acyclicity row, a root-or-root-free witness row, and a syntactic-representation row whose visible nodes are histories.*
2. $\mathsf{TreeClassifier}(h, k)$ *is* $\mathsf{hsame}(h, k)$ *together with transport of the graph-source, connectedness, acyclicity, root-or-root-free, and syntactic-representation rows from* h *to* k *and back along* hsame*.*
3. *The ledger records exactly three hidden rows: the graph presentation supplied by* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$*, the root witness or explicit root-free witness, and the derivation-spine syntax read over* BHist *nodes.*

The obligation surface is scoped over the existing seed row , so it introduces no new kernel primitive and no independent object equality beyond hsame*.*

Lean4 (definition) `BEDC.Derived.TreeUp.TreeObligationSurface`

Theorem 119.17 (Tree graph-source obligation)**.** *For every history* h*,* $\mathsf{TreeCarrier}(h)$ *exposes the underlying* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *source row. If* $\mathsf{TreeClassifier}(h, k)$ *holds, then this graph-source row is transported from* h *to* k *along* $\mathsf{hsame}(h, k)$*, and the reverse transport is supplied by the symmetric* hsame *witness contained in the same classifier row.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeGraphSource obligation rows`

Proof. Unpack $\mathsf{TreeCarrier}(h)$ from Definition 119.16. Its first component is the graph certificate source row, so the carrier projection directly exposes the $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ datum.

Now unpack $\mathsf{TreeClassifier}(h, k)$. The classifier contains $\mathsf{hsame}(h, k)$ and the declared row transport for the graph source. Symmetry of hsame supplies the reverse endpoint comparison, and the seed scope fixes that the transported source is the same graph-source row rather than a second graph presentation. □

Theorem 119.18 (Tree carrier-classifier obligation)**.** *On carried histories,* TreeClassifier *is reflexive, symmetric, and transitive, and it transports* TreeCarrier*: if* $\mathsf{TreeCarrier}(h)$ *and* $\mathsf{TreeClassifier}(h, k)$ *hold, then* $\mathsf{TreeCarrier}(k)$ *holds with the graph-source, connectedness, acyclicity, root-or-root-free, and syntactic-representation rows transported along* $\mathsf{hsame}(h, k)$*.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeObligationSurface classifier transport`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier classifier transport obligation`

Proof. Reflexivity uses reflexivity of hsame on BHist and the identity transport of each row in Definition 119.16. The graph source comes from $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$; the connectedness, acyclicity, root-or-root-free, and syntactic-representation rows are carried unchanged at the same endpoint.

Symmetry uses symmetry of hsame. Each row transport stored in $\mathsf{TreeClassifier}(h, k)$ is part of a two-sided transport package, so reversing the endpoint comparison reverses the carrier rows without changing the visible ledger.

Transitivity composes the two hsame witnesses and then composes the corresponding row transports. The composed transport carries exactly the graph-source, connectedness, acyclicity, root-or-root-free, and syntactic-representation rows named by the surface, so carrier transport follows by repacking those transported rows as $\mathsf{TreeCarrier}(k)$. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier classifier obligation`

Theorem 119.19 (Tree root-witness obligation)**.** *Every* $\mathsf{TreeCarrier}(h)$ *supplies exactly one admissible root row: either a designated root witness compatible with the carried* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *source or an explicit root-free witness row. This witness row is stable under* TreeClassifier*, so* $\mathsf{TreeClassifier}(h, k)$ *transports the same root-or-root-free alternative from* h *to* k *along* hsame*.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeRootWitness spine extension exactness`

Proof. The carrier row in Definition 119.16 includes the root-or-root-free witness as a named component of the tree certificate. The alternative is admissible because it is read against the same $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ source row exposed by Definition 119.17.

For stability, unpack $\mathsf{TreeClassifier}(h, k)$. Its row-transport component explicitly includes the root-or-root-free row, and the endpoint relation is $\mathsf{hsame}(h, k)$. Transport therefore preserves the chosen alternative, while the ledger component records that no additional root presentation is introduced. □

Theorem 119.20 (Tree stability-ledger obligation)**.** *Let* u *be a visible derivation-spine extension with* $\mathsf{UnaryHistory}(u)$*, and suppose* $\mathsf{Cont}(h, u, r)$ *is compatible with the carried graph source. If* $\mathsf{TreeCarrier}(h)$ *holds and the endpoints* r *and* k *are related by* $\mathsf{hsame}(r, k)$*, then the tree ledger rows transported to* k *are exactly the graph presentation, root witness, and derivation-spine syntax rows obtained by extending the visible spine by* u*.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeGraphContEdge visible spine extension transport`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier stability ledger transport`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier visible spine extension ledger`

Proof. Start with $\mathsf{TreeCarrier}(h)$ and the $\mathsf{Cont}(h, u, r)$ extension. Compatibility with the graph source says that the visible extension reads edges through the same $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ row, and $\mathsf{UnaryHistory}(u)$ keeps the appended spine inside the unary derivation shape allowed by the seed scope.

The endpoint comparison $\mathsf{hsame}(r, k)$ supplies the classifier transport target. Applying the row transports in Definition 119.16 moves the graph presentation, root witness, and derivation-spine syntax rows to k.

It remains to identify the ledger boundary. The ledger of the surface contains only those three hidden rows, so after Cont extension and hsame transport there is no extra presentation row to account for. The transported ledger is therefore exactly the extended visible-spine ledger. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeVisibleDerivationSpine extension transport`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeVisibleSpine extension ledger`

Theorem 119.21 (Tree syntactic-representation obligation)**.** *Every* $\mathsf{TreeCarrier}(h)$ *provides a syntactic derivation carrier row whose visible nodes are* BHist *histories and whose edge relation is read from the carried* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *source. Consequently the* $\mathsf{Tree}^{\uparrow}$ *certificate surface can be consumed by downstream inductive-carrier and derivation-syntax chapters without replacing the BEDC history carrier or the* hsame *classifier.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier syntactic representation`

Proof. Unpack the syntactic-representation row in Definition 119.16. Its node carrier is explicitly BHist, so every visible derivation node is already a BEDC history rather than an external graph vertex.

The edge relation is not an additional primitive. It is read from the $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ source exposed by Definition 119.17, and classifier movement is handled by hsame through Definition 119.18.

Downstream consumers therefore receive the same carrier, the same classifier, and the same visible derivation-spine syntax. Inductive-carrier arguments may use $\mathsf{UnaryHistory}$ and Cont for spine extension, while derivation-syntax arguments read edges from the graph source fixed by the existing seed scope. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreeSyntacticRepresentation carrier readback`

Theorem 119.22 (TreeUp public certificate boundary)**.** *The public* $\mathsf{Tree}^{\uparrow}$ *interface is the single package consisting exactly of the* $\mathsf{Graph}^{\uparrow}$ *source package, the* $\mathsf{GraphContEdge}$ *rows, the connectedness row, the acyclicity row, the root-or-root-free branch, the syntactic-representation row, stability transports, ledger rows, and exactness clauses established by Definition 119.3, Definition 119.4, Definition 119.5, Definition 119.7, Definition 119.8, and Definition 119.21. This boundary exports no host graph equality and no unrecorded route choice.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreeObligationSurface public certificate boundary`

Proof. The visible carrier package is supplied by Definition 119.3: it exposes the $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ source, the $\mathsf{GraphContEdge}$ rows, connectedness, acyclicity, and the rooted or root-free branch. The classifier boundary is the rowwise transport of Definition 119.4, so public comparison is only through the inherited graph package and the displayed tree rows.

The stability and ledger fields are exactly those named in Definition 119.7. They record the route, root-choice, connectedness, and acyclicity witnesses needed to justify the visible rows, while the transports keep every public component on the same BHist, hsame, Pkg, and $\mathsf{NameCert}$ surface.

Exactness comes from Definition 119.8: every accepted endpoint reads back to the displayed package, and every displayed connected acyclic graph package with one root branch repacks as a $\mathsf{Tree}^{\uparrow}$ carrier. The syntactic-representation row of Definition 119.21 adds the derivation-spine consumer surface without changing the carrier or classifier.

Thus the public package is precisely the listed rows and clauses. Host graph equality is not a public component, route choices remain ledger entries rather than exported data, and no standard bridge is included in the $\mathsf{Tree}^{\uparrow}$ boundary. □

Lean4 ✓ `BEDC.Derived.TreeUp.TreePublicCertificateBoundary semantic name certificate`

Theorem 119.23 (TreeUp public derivation-syntax bridge)**.** *The public* $\mathsf{Tree}^{\uparrow}$ *package of Definition 119.22, together with Definition 119.16 and Definition 119.21, determines a paper bridge to the external finite rooted-or-root-free tree presentation whose nodes are* BHist *histories and whose edges are the displayed* $\mathsf{GraphContEdge}$ *rows. Interpretation from* $\mathsf{Tree}^{\uparrow}$ *to the external presentation*

and re-encoding from any externally presented tree whose rows are already displayed $\mathsf{Graph}^{\uparrow}/\mathsf{Tree}^{\uparrow}$ *rows are exact up to the inherited* hsame *classifier. Neither direction exports host graph equality or hidden route choice.*

Lean4 ✓ `BEDC.Derived.TreeUp.TreePublicDerivationSyntaxBridge visible spine package`
Lean4 ✓ `BEDC.Derived.TreeUp.TreeBHistCarrier derivation syntax roundtrip`
Lean4 ✓ `BEDC.Derived.TreeUp.TreePublicDerivationSyntaxBridge round trip`

Proof. For interpretation, read the public boundary of Definition 119.22. Its visible rows give the $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ source, the displayed GraphContEdge edges, connectedness, acyclicity, the root-or-root-free branch, and the syntactic-representation row. By Definition 119.16 and Definition 119.21, the nodes of that presentation are precisely BHist histories, and the edge table is precisely the displayed graph-continuation edge table.

For re-encoding, start with an external finite rooted-or-root-free tree presentation whose nodes are already BHist histories and whose edge rows are already displayed $\mathsf{Graph}^{\uparrow}/\mathsf{Tree}^{\uparrow}$ rows. Apply Definition 119.8: the displayed graph source, edge rows, connectedness and acyclicity rows, root branch, syntactic row, stability transports, and ledger rows repack as the same public $\mathsf{Tree}^{\uparrow}$ package.

The first round trip compares the interpreted package with the original $\mathsf{Tree}^{\uparrow}$ data. Use Definition 119.4 to compare the public graph and path rows componentwise, and use Definition 119.18 to transport the tree carrier rows along the inherited hsame witness.

The second round trip compares the re-encoded external presentation with the displayed rows from which it was formed. The carrier, classifier, and syntactic-representation clauses are the same rows named in Definition 119.16; exactness prevents an additional route or graph-equality component from entering the package. Both comparisons are therefore exact only up to hsame, which is the advertised bridge classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$**.**

Constructive story: A tree is accepted only after a visible $\mathsf{Graph}^{\uparrow}$ package has been equipped with finite incidence data, connectedness, acyclicity, one root-or-root-free branch, and a syntactic derivation-spine row over BHist nodes. The carrier is the displayed endpoint row carrying those graph and tree witnesses; a presentation without the graph source, the GraphContEdge rows, or the connected/acyclic/root branch evidence is outside the carrier. Two tree endpoints classify together by transporting the graph package, edge rows, connectedness, acyclicity, root-or-root-free row, and syntactic representation along hsame and the inherited $\mathsf{Graph}^{\uparrow}$ classifiers, not by host graph isomorphism. The source is the existing graph certificate together with the tree rows, the pattern records path readback to the root component, collapse of closed walks to the stationary row, root-branch transport, visible-spine extension, and derivation-syntax readback, while the ledger keeps hidden route, root-choice, connectedness, acyclicity, and syntax witnesses behind the public rows. The five name-certificate obligations use any displayed connected acyclic graph package with a root branch for inhabitation, inherit reflexivity, symmetry, and transitivity from hsame and graph classifiers, and close the carrier by transporting every listed row together. A tree history h lies in the signature gap of a $\mathsf{Tree}^{\uparrow}$ token p when the source-domain accepts its graph package and tree witnesses, the signature relation pairs it with an answer tree package and evidence ledger, and token introduction names that answer as p; unfolded, h has exactly the nodes, graph-continuation edges, connectedness, acyclicity, root alternative, and derivation spine encoded by the token. The derivation-syntax bridge reads this as an external finite rooted-or-root-free tree over BHist nodes and GraphContEdge rows, then reads back through the same public package. The chapter registers this graph-with-obligations construction under the name $\mathsf{Tree}^{\uparrow}$; a tree is a connected acyclic graph ledger with a root boundary, not host graph equality, unchecked route choice, or a separate inductive tree primitive.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public $\mathsf{Tree}^{\uparrow}$ boundary, the derivation-syntax bridge of Definition 119.23, and the visible-spine extension ledger rows identify the external finite rooted-or-root-free tree presentation over BHist nodes, GraphContEdge edge rows, Cont, hsame, Pkg, $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$, connectedness, acyclicity, root-branch rows, and derivation syntax.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.GraphUp.GraphCont\_namecert\_surface`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim checked bridge verification, checked connectedness or acyclicity rows, checked root transport, or full $\mathsf{Tree}^{\uparrow}$ Lean verification.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 120

A Concrete Naming Certificate for $\mathsf{Seq}^{\uparrow}$

The sequence interface $\mathsf{Seq}^{\uparrow}$ packages a function $\mathsf{Nat}^{\uparrow} \to A$ for a fixed naming-certificate carrier A, together with a regularity or boundedness modulus. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28) and is the general substrate for $\mathsf{NameCert}_{\mathsf{Series}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{ConvRad}^{\uparrow}}$, and the analytic-limit certificates.

120.1 Carrier definition

Definition 120.1 (SeqUp root BHist source)**.** *Fix a certified point carrier* $\mathsf{NameCert}(A, \sim_A)$ *and the unary-index certificate* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$*. A* $\mathsf{Seq}^{\uparrow}$ *root BHist source over* A *is a BHist-indexed observation function* $s : \mathsf{Hist} \to \mathsf{Hist}$ *together with the following finite BEDC evidence:*

1. *an index witness saying that each admitted index* n *is a* $\mathsf{Nat}^{\uparrow}$ *unary history, equivalently* $\mathsf{UnaryHistory}(n)$*;*
2. *a point carrier witness* $A(s(n))$ *for every admitted unary index* n*;*
3. *an optional modulus or boundedness ledger whose entries are themselves unary histories and whose use is named by downstream consumers.*

Its ten-field fit is: traditional target sequences over a certified carrier; BEDC source $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *together with* $\mathsf{NameCert}(A, \sim_A)$*; carrier the displayed BHist observation source with point carrier evidence; classifier the pointwise* $\sim_A$ *classifier of Definition 120.2; stability inherited indexwise from the two source certificates; ledger the index witness, point carrier witnesses, and optional modulus or boundedness row; core theorem list Definition 120.3, Definition 120.4, and Definition 120.5; exactness point observation equality only up to the source classifier; standard bridge absent; status paper-level substrate for the derived namespace* BEDC.Derived.SeqUp*.*

Lean4 ✓ `BEDC.Derived.SeqUp.SeqTasteGate single carrier alignment`

Definition 120.2 (SeqUp pointwise classifier)**.** *For two* $\mathsf{Seq}^{\uparrow}$ *root BHist sources* s *and* t *over the same* $\mathsf{NameCert}(A, \sim_A)$*, define* $s \sim_{\mathsf{Seq}^{\uparrow}(A)} t$ *iff every unary index has source carrier witnesses for both point observations and a source-classifier witness between them:*

$$s \sim_{\mathsf{Seq}^{\uparrow}(A)} t \quad \Longleftrightarrow \quad \forall n,\ \mathsf{UnaryHistory}(n) \Rightarrow \big(A(s(n)) \wedge A(t(n)) \wedge s(n) \sim_A t(n)\big).$$

No convergence, summability, completeness, or host-function extensionality is part of this classifier.

120.2 The certificate

Theorem 120.3 (SeqUp carrier and classifier obligations)**.** *For every certified point carrier* $\mathsf{NameCert}(A, \sim_A)$ *and every* $\mathsf{Seq}^{\uparrow}$ *root BHist source* s *over* A*, the source supplies the point carrier projection* $A(s(n))$ *at each unary index* n*. The pointwise classifier* $\sim_{\mathsf{Seq}^{\uparrow}(A)}$ *is reflexive, symmetric, and transitive on carried* $\mathsf{Seq}^{\uparrow}$ *sources, with all three witnesses inherited indexwise from the reflexivity, symmetry, and transitivity fields of* $\mathsf{NameCert}(A, \sim_A)$*.*

Lean4 ✓ `BEDC.Derived.SeqUp.SeqPointwiseClassifier rows`

Proof. For the carrier projection, unfold Definition 120.1. The second displayed field gives $A(s(n))$ whenever the index witness supplies $\mathsf{UnaryHistory}(n)$.

For reflexivity, fix a carried source s and a unary index n. The carrier projection gives $A(s(n))$, and the reflexivity field of $\mathsf{NameCert}(A, \sim_A)$ gives $s(n) \sim_A s(n)$.

For symmetry, use the same unary index. A witness of $s \sim_{\mathsf{Seq}^{\uparrow}(A)} t$ contains $A(s(n))$, $A(t(n))$, and $s(n) \sim_A t(n)$; applying source symmetry gives $t(n) \sim_A s(n)$ while retaining the two carrier witnesses.

For transitivity, combine the two pointwise witnesses at the same unary index. Their source-classifier components are composable by transitivity in $\mathsf{NameCert}(A, \sim_A)$, and their carrier components provide the carried endpoints required by Definition 120.2. □

Theorem 120.4 (SeqUp pointwise transport obligation)**.** *Let* s *and* t *be* $\mathsf{Seq}^{\uparrow}$ *root BHist sources over* $\mathsf{NameCert}(A, \sim_A)$*. If* $s \sim_{\mathsf{Seq}^{\uparrow}(A)} t$ *and* s *is carried pointwise, then* t *is carried pointwise. Equivalently, for every unary index* n*, the source carrier witness* $A(s(n))$ *transports along the point classifier witness* $s(n) \sim_A t(n)$ *to a carrier witness* $A(t(n))$*.*

Proof. Fix a unary index n. Project the pointwise classifier witness of Definition 120.2; it supplies $A(s(n))$, $A(t(n))$, and $s(n) \sim_A t(n)$.

The named transport obligation uses only the carrier-transport field of the point certificate. Apply carrier transport in $\mathsf{NameCert}(A, \sim_A)$ to the carried source endpoint $A(s(n))$ and the classifier witness $s(n) \sim_A t(n)$. The result is $A(t(n))$. □

Lean4 ✓ `BEDC.Derived.SeqUp.SeqPointwiseClassifier target source`

Theorem 120.5 (SeqUp downstream boundary)**.** *Every downstream consumer of* $\mathsf{Seq}^{\uparrow}$*, including* $\mathsf{Series}^{\uparrow}$*,* $\mathsf{ConvRad}^{\uparrow}$*, and analytic-limit certificates, may read only the following data from a* $\mathsf{Seq}^{\uparrow}$ *source: point observations* $s(n)$*, unary index evidence* $\mathsf{UnaryHistory}(n)$*, pointwise classifier transport along* $\sim_{\mathsf{Seq}^{\uparrow}(A)}$*, and the optional modulus or boundedness ledger named in Definition 120.1. Such a consumer may not infer convergence, summability, holomorphicity, completeness, or any host-function principle from* $\mathsf{Seq}^{\uparrow}$ *alone.*

Lean4 ✓ `BEDC.Derived.SeqUp.SeqRootSource downstream boundary`

Proof. Unfold the carrier definition and classifier definition. The carrier exposes a BHist observation function, unary-index evidence, point carrier evidence, and an optional unary-valued ledger; the classifier exposes exactly the pointwise source-classifier witnesses.

The analytic properties listed in the statement are absent from the ten-field fit: exactness is point observation equality up to $\sim_A$, the standard bridge is absent, and the ledger is limited to index and modulus or boundedness evidence. Therefore any consumer requiring convergence, summability, holomorphicity, completeness, or a host-function principle must introduce that requirement in its own certificate rather than extracting it from $\mathsf{Seq}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$**.**

Constructive story: A sequence over a certified point carrier is carried as a visible observation function from unary histories to point histories: each admitted index must be a $\mathsf{Nat}^{\uparrow}$ unary history, and each observed value must be accepted by the fixed point carrier. The classifier is pointwise rather than extensional over host functions: two sequence sources classify together when every unary index has carrier witnesses on both sides and the two observations are related by the point certificate's classifier. The source covers exactly these BHist observation functions with unary-index evidence, point-carrier evidence, and an optional unary modulus or boundedness row, while the pattern records indexwise projection, pointwise comparison, pointwise transport, and the downstream boundary read by series and analytic-limit certificates. The ledger keeps the unary index witnesses, point carrier witnesses, and optional modulus entries visible, so nonemptiness is inherited from any carried source, reflexivity, symmetry, and transitivity are inherited index by index from the point certificate, and carrier closure is just point-carrier transport at each unary index. A history package h lies in the signature gap of a $\mathsf{Seq}^{\uparrow}$ token p when the source-domain accepts its observation source, the signature relation pairs its pointwise observations and modulus ledger with an answer package, and the token-introduction relation names that package as p; unfolded, every unary index must expose the point value and optional modulus encoded by the token. The chapter registers this pointwise observation construction under the name $\mathsf{Seq}^{\uparrow}$; it introduces no convergence, summability, completeness, holomorphicity, or host-function extensionality.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public scope exports the root BHist source, pointwise classifier, pointwise transport, optional modulus ledger, and downstream consumer boundary of Definition 120.3, Definition 120.4, and Definition 120.5.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SeqUp.SeqPointwiseClassifier\_rows`

Bridge status: none

Not claimed: This chapter does not close convergence, summability, completeness, or host-function extensionality for sequence objects.

Upgrade path: No further paper theory-axis upgrade is recorded for this chapter; bridge verification remains a separate formal-axis matter.

Chapter 121

A Concrete Naming Certificate for $\mathsf{Series}^{\uparrow}$

The series interface $\mathsf{Series}^{\uparrow}$ packages the partial-sum sequence of a $\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$ (chapter 120) over an additive carrier together with explicit convergence moduli. It is the real-and-general counterpart of $\mathsf{NameCert}_{\mathsf{CplxSer}^{\uparrow}}$ (chapter 69).

121.1 Carrier definition

Theorem 121.1 (Series source carrier boundary obligation)**.** *The* $\mathsf{Series}^{\uparrow}$ *source carrier is the* $\mathsf{Seq}^{\uparrow}$ *source carrier together with additive-carrier rows over* BHist*. A carried row consists of a sequence certificate* $\mathcal{S} : \mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$*, a summand stream* $a : \mathsf{BHist}$ *accepted by that source, an additive carrier row* $A : \mathsf{BHist}$*, and a displayed action row saying that every unary summand observation of* a *lands in* A*. The classifier is the restriction of the* $\mathsf{Seq}^{\uparrow}$ *stream classifier together with* hsame *on the additive carrier row; no ambient real-number carrier, host sequence, or external summability predicate is part of the source carrier.*

Proof. Project the $\mathsf{Seq}^{\uparrow}$ certificate to obtain the stream carrier, stream classifier, and unary observation rows. Pair these projections with the displayed additive-carrier row over BHist.

The restriction of the classifier is componentwise: the sequence component is transported by the $\mathsf{Seq}^{\uparrow}$ source classifier, and the additive component is transported by hsame. Repacking those two rows gives exactly the announced carrier boundary without importing a host-level series object. □

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesSourceCarrierBoundary obligation`

Theorem 121.2 (Series partial-sum obligation)**.** *For every carried* $\mathsf{Series}^{\uparrow}$ *source row, the partial-sum row is a unary prefix family* $s : \mathsf{BHist} \to \mathsf{BHist}$ *generated from the summand stream by finite* Cont*-compatible prefixes. The zero prefix is the additive zero row, the successor prefix is obtained by appending one summand observation through the additive operation row, and prefix comparison is transported by the* $\mathsf{Seq}^{\uparrow}$ *stream classifier and by* hsame *on the additive carrier. Thus partial sums are BEDC ledger rows, not hidden evaluations of an external infinite sum.*

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesPartialSumLedger step transport`
↪ variant `BEDC.Derived.SeriesUp.SeriesPartialSum transport`

Proof. Start from the unary index witness supplied by the $\mathsf{Seq}^{\uparrow}$ observation surface. The empty prefix is read as the additive zero row, and the successor case appends the next observed summand using the additive operation row.

Compatibility with Cont is by induction on the unary prefix length. The base case is the zero prefix, and the successor step composes the previous prefix ledger with the next summand observation. Classifier transport follows by applying the $\mathsf{Seq}^{\uparrow}$ stream classifier to the summand slot and hsame transport to the additive row. □

121.2 The certificate

Theorem 121.3 (Series modulus stability obligation)**.** *A convergence modulus for a carried* $\mathsf{Series}^{\uparrow}$ *row is an explicit unary bound* $\mu : \mathsf{BHist} \to \mathsf{BHist}$ *such that, for every unary precision row* k *and every pair of unary indices beyond* $\mu(k)$*, the corresponding partial-sum rows are classified by the additive carrier at precision* k*. If two summand streams are classified by the* $\mathsf{Seq}^{\uparrow}$ *stream classifier, the same displayed modulus row transports to the classified stream by the partial-sum classifier of Definition 121.2. The modulus is therefore stable under the source classifier and remains a visible BEDC row.*

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesModulusStability transport`
↪ variant `BEDC.Derived.SeriesUp.SeriesPartialSum pointwise hsame deterministic`

Proof. Unpack the modulus as a unary history transformer together with its bound witness for pairs of sufficiently long prefixes. The witness refers only to partial-sum rows and the additive classifier, so it is scoped by Definition 121.1 and Definition 121.2.

For classifier stability, transport the summand observations along the $\mathsf{Seq}^{\uparrow}$ stream classifier. The partial-sum obligation transports each finite prefix, and the additive classifier transports the comparison at precision k. The same unary bound row $\mu(k)$ can therefore be reused after transport. □

Theorem 121.4 (Series ledger exactness obligation)**.** *The* $\mathsf{Series}^{\uparrow}$ *ledger contains exactly the* $\mathsf{Seq}^{\uparrow}$ *source certificate, the additive-carrier row, the hidden summand-prefix rows, the partial-sum rows, the explicit modulus rows, the stream-classifier transport rows, and the additive-classifier comparison rows. Coverage says every accepted public series row exposes one such ledger; soundness says every ledger row is produced by those BEDC primitives; completeness says any package with precisely those rows is accepted by the* $\mathsf{Series}^{\uparrow}$ *obligation surface. No comparison with* $\mathsf{CplxSer}^{\uparrow}$*, no host summability theorem, and no standard bridge is included in this ledger.*

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesPartialSum result unary`

Proof. Coverage is obtained by projecting an accepted public row into its source certificate, additive-carrier row, summand-prefix display, partial-sum display, and modulus display.

Soundness reads each projected component back through the corresponding BEDC primitive: $\mathsf{Seq}^{\uparrow}$ supplies stream observations, Cont supplies finite prefix composition, hsame supplies classifier transport, and the additive carrier supplies zero, addition, and comparison rows.

Completeness is the reverse packing. Given exactly those rows, assemble the carrier boundary, the partial-sum family, and the modulus stability witness, then record them as the public $\mathsf{Series}^{\uparrow}$ obligation surface. Since every accepted row is accounted for in this package, the dependency surface is exact. □

Theorem 121.5 (SeriesUp public name certificate)**.** *The* $\mathsf{Series}^{\uparrow}$ *source carrier boundary, partial-sum obligation, modulus stability obligation, and ledger exactness obligation assemble into the public* $\mathsf{NameCert}_{\mathsf{Series}^{\uparrow}}$ *surface. The public certificate exposes exactly the* $\mathsf{Seq}^{\uparrow}$ *source row, the additive carrier row, finite* Cont *partial-sum routes, explicit unary modulus rows, classifier transports, and ledger rows recorded in Definition 121.1, Definition 121.2, Definition 121.3, and Definition 121.4.*

It does not include a $\mathsf{CplxSer}^{\uparrow}$ *comparison, a host summability theorem, or a standard bridge claim.*

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesPartialSum public name certificate`

Proof. First apply Definition 121.1. This supplies the $\mathsf{Seq}^{\uparrow}$ source certificate, the summand stream, the additive carrier row, and the source classifier surface for the public name certificate.

Next apply Definition 121.2 and Definition 121.3. The first theorem supplies the finite Cont partial-sum routes, while the second supplies explicit unary modulus rows and their classifier transports.

Finally apply Definition 121.4. Its coverage, soundness, and completeness clauses say that these rows are exactly the public ledger rows accepted by the $\mathsf{Series}^{\uparrow}$ surface. Repacking the four scoped obligation rows into the $\mathsf{NameCert}_{\mathsf{Series}^{\uparrow}}$ fields gives the public certificate and leaves complex-series comparisons, host summability, and standard bridges outside the exposed boundary. □

Theorem 121.6 (SeriesUp standard bridge schema row)**.** *The public* $\mathsf{Series}^{\uparrow}$ *certificate admits a paper bridge row to the standard partial-sum/modulus schema. The bridge maps the visible* $\mathsf{Seq}^{\uparrow}$ *summand stream to the schema source stream, the additive carrier row to the schema carrier, the finite* Cont *partial-sum routes to the schema partial-sum field, the explicit unary modulus rows to the schema modulus field, and the additive-classifier comparison rows to the schema comparison field. This bridge exports no* $\mathsf{CplxSer}^{\uparrow}$ *comparison and no host summability theorem.*

Proof. First apply Definition 121.5. It identifies the public $\mathsf{Series}^{\uparrow}$ surface as the $\mathsf{Seq}^{\uparrow}$ source row, additive carrier row, finite Cont partial sums, explicit unary moduli, classifier transports, and ledger exactness rows.

Now read those rows against the standard partial-sum/modulus schema. The summand stream fills the schema source, the additive carrier fills the carrier slot, the Cont routes fill the finite partial-sum slot, the unary modulus rows fill the modulus slot, and the additive-classifier rows fill the comparison slot.

The preceding public certificate keeps complex-series comparisons, host summability, and extra standard fields outside the exposed boundary. Therefore the bridge is exactly a paper row from the visible $\mathsf{Series}^{\uparrow}$ certificate to the standard partial-sum/modulus schema, with no $\mathsf{CplxSer}^{\uparrow}$ comparison and no host summability theorem. □

Lean4 ✓ `BEDC.Derived.SeriesUp.SeriesPartialSum schema bridge endpoint unary`

BEDC.Derived.SeriesUp.SeriesPartialSumLedger_step_transport

Closure status for $\mathsf{NameCert}_{\mathsf{Series}^{\uparrow}}$.

Constructive story: A series source is carried by taking a $\mathsf{Seq}^{\uparrow}$ summand stream over an additive carrier and displaying, for every unary index, both the summand observation and the additive row in which partial sums will live. The classifier is componentwise: the stream is compared by the $\mathsf{Seq}^{\uparrow}$ pointwise classifier and the additive carrier row by hsame, so no host sequence, ambient real carrier, or external summability predicate enters the carrier. The source covers exactly the sequence certificate, summand stream, additive carrier, and unary observation action rows, while the pattern generates partial sums as finite Cont-compatible prefixes: the empty prefix is the additive zero row and each successor prefix appends the next observed summand through the additive operation. The ledger keeps the $\mathsf{Seq}^{\uparrow}$ source certificate, hidden summand-prefix rows, partial-sum rows, explicit unary convergence moduli, stream-classifier transports, and additive-classifier comparisons, and those rows give nonemptiness, reflexive/symmetric/transitive comparison, and carrier closure by inherited pointwise

transport. A history h lies in the signature gap of a Series$^{\uparrow}$ token p when the source-domain accepts its summand stream and additive row, the signature relation pairs the displayed partial-sum family and modulus ledger with an answer history, and the token-introduction relation names that answer as p; unfolded, h is accepted precisely when its finite prefixes and modulus rows match the token's series package. The chapter registers this partial-sum ledger construction under the name Series$^{\uparrow}$, exports its public certificate theorem, and gives the paper bridge row from the visible series certificate to the standard partial-sum/modulus schema; it includes no comparison with CplxSer$^{\uparrow}$ and no host summability theorem.

Theory closure: TheoryGate(NameCert$_{\mathsf{Series}\uparrow}$, maturePackageClosed)

Scope closed: The mature paper scope binds the NameCert$_{\mathsf{Seq}\uparrow}$ source row, additive-carrier rows, finite Cont partial sums, explicit unary moduli, stream-classifier stability, public certificate theorem, ledger exactness, and paper bridge to the standard partial-sum/modulus schema.

Formal verification: FormalStatus(NameCert$_{\mathsf{Series}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.SeriesUp.SeriesPartialSumLedger\_step\_transport`

Bridge status: paperBridge

Not claimed: This chapter does not claim a CplxSer$^{\uparrow}$ comparison, a host summability theorem, or a Lean-checked standard bridge.

Upgrade path: The next verification step is a Lean-checked standard bridge for the same partial-sum ledger and modulus-stability surface.

Chapter 122

A Concrete Naming Certificate for $\mathsf{Derivative}^{\uparrow}$

The derivative interface $\mathsf{Derivative}^{\uparrow}$ packages the constructive pointwise derivative of a function between metric carriers, with explicit modulus relating ε to the difference-quotient tolerance. It is the real and general counterpart of $\mathsf{NameCert}_{\mathsf{CplxDiff}^{\uparrow}}$ (chapter 67).

122.1 Carrier definition

Definition 122.1 (Derivative metric quotient)**.** *For histories $f, z, h, q, dist : \mathsf{BHist}$, the derivative metric quotient carrier holds when f, z, q, h, and $dist$ are unary histories, h is complex-nonzero, q is the continuation result of f by the step h, and $dist$ is the continuation result of h by the quotient q:*

$$\mathsf{UnaryHistory}(f) \wedge \mathsf{UnaryHistory}(z) \wedge \mathsf{CplxNonZero}(h) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{Cont}(f, h, q) \wedge \mathsf{UnaryHistory}(h) \wedge \mathsf{UnaryHistory}(d$$

Lean4 (definition) `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient`

Theorem 122.2 (Derivative metric quotient history transport)**.** *For histories $f, f', z, z', h, h', q, q', dist, dist' :$ BHist, if each source component is classifier-equal to its primed component, then a derivative metric quotient for $(f, z, h, q, dist)$ transports to a derivative metric quotient for $(f', z', h', q', dist')$. The transported package also reads back the complex difference quotient, the metric distance witness, and the two displayed continuation ledgers:*

$$\mathsf{CplxDiffQuot}(f', z', h', q'), \qquad \mathsf{MetricDistanceWitness}(h', q', dist'), \qquad \mathsf{Cont}(f', h', q'), \qquad \mathsf{Cont}(h', q', dist').$$

Proof. Destructure the derivative metric quotient into its unary carriers, the complex nonzero step proof, and the two continuation ledgers. Transport unary carriers along the five hsame assumptions. The nonzero proof for h' composes a hypothetical $h'\mathsf{hsame}e$ with $h\mathsf{hsame}h'$. The difference ledger transports along the function, step, and quotient equalities. The metric ledger transports as the continuation field of the concrete metric distance witness built from h, q, $dist$, and $\mathsf{Cont}(h, q, dist)$. Reassembling these transported fields gives all stated components. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient hsame transport`

Theorem 122.3 (Derivative metric quotient function prefix closure)**.** *For histories* $p, f, z, h, q, dist$: BHist*, if* p *is unary and* $\mathsf{DerivativeMetricQuotient}(f, z, h, q, dist)$ *holds, then the function and quotient histories may be prefixed together while the step and point remain fixed:*

$$\mathsf{DerivativeMetricQuotient}(\mathsf{append}(p, f), z, h, \mathsf{append}(p, q), \mathsf{append}(h, \mathsf{append}(p, q))).$$

Proof. Unpack the derivative metric quotient into unary carriers, the complex-nonzero step proof, and the two continuation ledgers. Unary closure under append gives the prefixed function and prefixed quotient carriers. The function ledger $\mathsf{Cont}(f, h, q)$ is transported through the shared prefix by reassociating $\mathsf{append}(p, \mathsf{append}(f, h))$ with $\mathsf{append}(\mathsf{append}(p, f), h)$. The metric distance component is then rebuilt canonically from the original unary step, the prefixed quotient, and the defining continuation $\mathsf{Cont}(h, \mathsf{append}(p, q), \mathsf{append}(h, \mathsf{append}(p, q)))$. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient function prefix closed`

Theorem 122.4 (Derivative metric quotient distance continuation depth)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h, q, dis$ *holds, then its metric distance ledger exposes* $\mathsf{Cont}(h, q, dist)$*, and the displayed metric-distance depth is the sum of the step depth and quotient depth:*

$$\mathsf{MetricDistanceDepth}(dist) = \mathsf{MetricDistanceDepth}(h) + \mathsf{MetricDistanceDepth}(q).$$

Proof. Destructure the derivative metric quotient into its two continuation ledgers and unary carriers. The fields for h, q, $dist$, and $\mathsf{Cont}(h, q, dist)$ form the concrete metric-distance witness, whose checked depth-add theorem computes the displayed distance depth. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient distance cont depth add`

Theorem 122.5 (Derivative empty-function quotient step readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(\mathsf{e}, z, h, q, dist)$ *holds, then the quotient result reads back to the step, the metric ledger is* $\mathsf{Cont}(h, q, dist)$*, and the displayed distance reads back to* $\mathsf{append}(h, h)$*:*

$$q\mathsf{hsame}h \wedge \mathsf{Cont}(h, q, dist) \wedge dist\mathsf{hsameappend}(h, h).$$

Proof. The function continuation ledger is $\mathsf{Cont}(\mathsf{e}, h, q)$, so left-unit readback gives $q\mathsf{hsame}h$. The metric field is the displayed ledger $\mathsf{Cont}(h, q, dist)$, hence $dist\mathsf{hsameappend}(h, q)$. Appending the recovered equality $q\mathsf{hsame}h$ under the fixed left context h identifies this with $\mathsf{append}(h, h)$. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient empty function step readback`

Theorem 122.6 (Derivative metric quotient distance result is nonempty)**.** *For histories* $f, z, h, q, dist$: BHist*, a derivative metric quotient cannot have a metric-distance result classified as* e*.*

Proof. Unpack the derivative metric quotient. Its metric ledger is a continuation $\mathsf{Cont}(h, q, dist)$, while the step field says that h is complex-nonzero. Transport the continuation result along $dist\mathsf{hsame}\mathsf{e}$. Empty-result inversion for Cont forces $h\mathsf{hsame}\mathsf{e}$, contradicting the complex-nonzero step witness. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient distance result nonempty`

Theorem 122.7 (Derivative metric quotient distance visible-context readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h,$ *holds and the distance history is displayed inside a common visible left and right append context equal to the same context around core, then* $dist\mathsf{hsame}core$*. The readback core is also not* hsame-*equal to* e*.*

Proof. Reassociate both visible displays and cancel the common append contexts around the distance and the core. This gives $dist\mathsf{hsame}core$. A hypothetical empty core transports back to an empty distance result, contradicting the checked non-emptiness theorem for derivative metric quotient distance results. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient distance visible context readback`

Theorem 122.8 (Derivative metric quotient quotient visible-context readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h,$ *holds and the quotient history is displayed inside a common visible left and right append context equal to the same context around core, then* $q\mathsf{hsame}core$*. The readback core is also not* hsame*-equal to* e*.*

Proof. Reassociate both visible displays and cancel the common append contexts around the quotient and the core. This gives $q\mathsf{hsame}core$. A hypothetical empty core transports back to an empty quotient result, contradicting the checked quotient non-emptiness component of derivative metric quotients. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient quotient visible context readback`

Theorem 122.9 (Derivative metric quotient point visible-context readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h, q,$ *holds and the point history is displayed inside a common visible left and right append context equal to the same context around core, then* $z\mathsf{hsame}core$ *and the readback core is unary.*

Proof. Reassociate both visible displays and cancel the common append contexts around the point and the core. This gives $z\mathsf{hsame}core$. The point unary carrier from the derivative metric quotient then transports across the recovered classifier equality. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient point visible context readback`

Theorem 122.10 (Derivative metric quotient step visible-context readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h, q,$ *holds and the step history is displayed inside a common visible left and right append context equal to the same context around core, then* $h\mathsf{hsame}core$*. The readback core is unary and is not* hsame*-equal to* e*.*

Proof. Reassociate both visible displays and cancel the common append contexts around the step and the core. This gives $h\mathsf{hsame}core$. The unary carrier for the step transports across this equality to the core. A hypothetical empty core transports back to an empty step, contradicting the complex-nonzero field of the derivative metric quotient. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient step visible context readback`

Theorem 122.11 (Derivative metric quotient function visible-context readback)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z,$ *holds and the function history is displayed inside a common visible left and right append context equal to the same context around core, then* $f\mathsf{hsame}core$ *and the readback core is unary.*

Proof. Reassociate both visible displays and cancel the common append contexts around the function and the core. This gives $f\mathsf{hsame}core$. The function unary carrier from the derivative metric quotient then transports across the recovered classifier equality. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient function visible context readback`

Theorem 122.12 (Derivative metric quotient distance is non-empty)**.** *If* $\mathsf{DerivativeMetricQuotient}(f, z, h, q, dist)$ *holds, then neither the quotient* q *nor the displayed distance* $dist$ *is* hsame*-equal to* e*.*

Proof. The quotient component is a concrete complex difference quotient, whose nonzero step forbids an empty continuation result. For the distance component, transporting a hypothetical $dist\mathsf{hsamee}$ into the continuation ledger $\mathsf{Cont}(h, q, dist)$ yields an empty result ledger. Empty-result inversion forces $h\mathsf{hsamee}$, contradicting the complex nonzero step field. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient quotient distance nonempty`

Theorem 122.13 (Derivative metric quotient result deterministic)**.** *For histories* $f, z, h, q, q', dist, dist' :$ BHist*, two derivative metric quotient witnesses with the same function, point, and step determine the same quotient and displayed metric-distance result up to* hsame*. The four continuation ledgers*

$$\mathsf{Cont}(f, h, q), \qquad \mathsf{Cont}(f, h, q'), \qquad \mathsf{Cont}(h, q, dist), \qquad \mathsf{Cont}(h, q', dist')$$

are exposed together with $q\mathsf{hsame}q'$ *and* $dist\mathsf{hsame}dist'$*.*

Proof. Destructure the two derivative metric quotient witnesses. Determinism of $\mathsf{Cont}(f, h, \cdot)$ identifies the two quotient histories. Transport the second metric ledger along that quotient equality, then apply determinism of $\mathsf{Cont}(h, q, \cdot)$ to identify the displayed distance histories. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient result deterministic`

Theorem 122.14 (Derivative metric quotient hsame-step result determinacy)**.** *For histories* $f, z, h, h', q, q', dist, di$ BHist*, if* $h\mathsf{hsame}h'$ *and*

$$\mathsf{DerivativeMetricQuotient}(f, z, h, q, dist) \qquad \textit{and} \qquad \mathsf{DerivativeMetricQuotient}(f, z, h', q', dist')$$

hold, then $q\mathsf{hsame}q'$ *and* $dist\mathsf{hsame}dist'$*. The exposed continuation ledgers are*

$$\mathsf{Cont}(f, h, q), \qquad \mathsf{Cont}(f, h', q'), \qquad \mathsf{Cont}(h, q, dist), \qquad \mathsf{Cont}(h', q', dist').$$

Proof. Destructure the two derivative metric quotient witnesses. Transport the second function continuation ledger along $h\mathsf{hsame}h'$ and apply continuation determinism to identify q with q'. Then transport the second metric continuation ledger along both the step equality and the quotient equality, and apply continuation determinism again to identify $dist$ with $dist'$. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient hsame step result deterministic`

Theorem 122.15 (Derivative same-quotient step and distance determinacy)**.** *For histories* $f, z, h, h', q, dist, dist' :$ BHist*, two derivative metric quotient witnesses with the same function, point, and quotient determine the same step and displayed metric-distance result up to* hsame*. The continuation ledgers*

$$\mathsf{Cont}(f, h, q), \qquad \mathsf{Cont}(f, h', q), \qquad \mathsf{Cont}(h, q, dist), \qquad \mathsf{Cont}(h', q, dist')$$

are exposed together with $h\mathsf{hsame}h'$ *and* $dist\mathsf{hsame}dist'$*.*

Proof. Read both derivative metric quotient witnesses as complex difference quotient witnesses with the same quotient. Determinism of the shared quotient identifies the two step histories and exposes both function continuation ledgers. Transport the second metric continuation ledger back along the step equality, then apply continuation determinism to identify the displayed distance histories. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient same quotient step distance deterministic`

Theorem 122.16 (Visible derivative quotient steps cannot share a quotient)**.** *If*

$$\mathsf{DerivativeMetricQuotient}(f, z, \mathsf{E0}(p), out, dist_0) \quad \textit{and} \quad \mathsf{DerivativeMetricQuotient}(f, z, \mathsf{E1}(q), out, dist_1)$$

hold with the same function, point, and quotient histories, then contradiction follows.

Proof. Same-quotient step and distance determinacy identifies the two visible step histories. The equality $\mathsf{E0}(p)\mathsf{hsame}\mathsf{E1}(q)$ contradicts constructor separation for BHist. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient visible step same quotient absurd`

Theorem 122.17 (Visible derivative quotient steps cannot share hsame quotients)**.** *If*

$$out\mathsf{hsame}out', \quad \mathsf{DerivativeMetricQuotient}(f, z, \mathsf{E0}(p), out, dist_0), \quad \mathsf{DerivativeMetricQuotient}(f, z, \mathsf{E1}(q), out', di$$

then contradiction follows.

Proof. Transport the second derivative metric quotient along the quotient classifier so that both witnesses have quotient *out*. The same-quotient visible-step separation theorem then identifies $\mathsf{E0}(p)$ with $\mathsf{E1}(q)$, contradicting constructor separation. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient visible step hsame quotient absurd`

Theorem 122.18 (Derivative complex-difference witness step is unary)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then it contains histories* h *and* q *such that* h *and* q *are unary,* $\mathsf{Cont}(f, h, q)$ *holds, and* $q\mathsf{hsame}f'$.

Proof. Unfold the displayed differentiability data to obtain a quotient witness $\mathsf{CplxDiffQuot}(f, z, h, q)$. The quotient step-unary theorem extracts unarity of h and q together with the continuation ledger $\mathsf{Cont}(f, h, q)$. The classifier field of $\mathsf{CplxDiffAt}$ identifies the quotient result q with the displayed derivative f'. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeCplxDiffAt witness step unary`

Theorem 122.19 (Derivative complex-difference derivative is nonempty)**.** *If* $\mathsf{CplxDiffAt}(f, z, f')$ *holds, then the displayed derivative* f' *cannot be* hsame*-equal to* e.

Proof. Unfold the differentiability witness to obtain a concrete quotient $\mathsf{CplxDiffQuot}(f, z, h, q)$. The classifier field identifies the quotient result q with the displayed derivative f'. A hypothetical $f'\mathsf{hsame}\mathsf{e}$ therefore makes $q\mathsf{hsame}\mathsf{e}$, contradicting quotient result non-emptiness from the nonzero step. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeCplxDiffAt derivative nonempty`

Theorem 122.20 (Derivative complex-difference empty-function step readback)**.** *If* $\mathsf{CplxDiffAt}(\mathsf{e}, z, f')$ *holds, then some nonzero unary step history* h *is classified as the displayed derivative:*

$$\exists h : \mathsf{BHist}, \mathsf{CplxNonZero}(h) \wedge \mathsf{UnaryHistory}(h) \wedge f'\mathsf{hsame}h.$$

Proof. Unpack the differentiability witness to obtain a concrete quotient $\mathsf{CplxDiffQuot}(\mathsf{e}, z, h, q)$. Its step-unary readback exposes the nonzero and unary fields for h and the ledger $\mathsf{Cont}(\mathsf{e}, h, q)$. Left-unit readback gives $q\mathsf{hsame}h$, while the classifier field identifies q with f'. Symmetry and transitivity identify f' with h. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeCplxDiffAt empty function step readback`

122.2 The certificate

Theorem 122.21 (Derivative metric quotient chain-product composition)**.** *For histories* $f, g, z, h_f, h_g, q_f, q_g, d_f, d_g$ BHist*, assume*

$$\mathsf{DerivativeMetricQuotient}(f, z, h_f, q_f, d_f), \qquad \mathsf{DerivativeMetricQuotient}(g, q_f, h_g, q_g, d_g).$$

Let $h_{fg}, q_{fg}, d_{fg}, F_{fg} : \mathsf{BHist}$ *be continuation witnesses with*

$$\mathsf{Cont}(h_f, h_g, h_{fg}), \qquad \mathsf{Cont}(q_f, q_g, q_{fg}), \qquad \mathsf{Cont}(d_f, d_g, d_{fg}), \qquad \mathsf{Cont}(f, g, F_{fg}),$$

and assume the displayed chain ledger

$$\mathsf{Cont}(f, h_{fg}, q_g).$$

Then the product representative of the chain is a derivative metric quotient,

$$\mathsf{DerivativeMetricQuotient}(F_{fg}, z, h_{fg}, q_{fg}, d_{fg}),$$

and its distance ledger is the continuation product of the two component distance histories:

$$\mathsf{Cont}(d_f, d_g, d_{fg}).$$

The displayed chain ledger also factors through the inner quotient endpoint:

$$\mathsf{Cont}(q_f, h_g, q_g) \wedge g\mathsf{hsame}q_f.$$

Proof. Unpack the two derivative metric quotient witnesses using Definition 122.1. The first witness gives unary carriers for f, z, h_f, q_f, d_f, the non-zero step witness for h_f, and the continuation ledgers $\mathsf{Cont}(f, h_f, q_f)$ and $\mathsf{Cont}(h_f, q_f, d_f)$. The second witness gives unary carriers for g, q_f, h_g, q_g, d_g, the non-zero step witness for h_g, and the continuation ledgers $\mathsf{Cont}(g, h_g, q_g)$ and $\mathsf{Cont}(h_g, q_g, d_g)$.

The supplied continuation ledgers give unary carriers for F_{fg}, h_{fg}, q_{fg}, and d_{fg}. If $h_{fg}\mathsf{hsamee}$, continuation-empty inversion applied to $\mathsf{Cont}(h_f, h_g, h_{fg})$ forces $h_f\mathsf{hsamee}$, contradicting the non-zero step field of the first quotient. Thus h_{fg} carries the required complex non-zero witness.

The chain ledger factors through q_f as follows. The supplied continuation $\mathsf{Cont}(h_f, h_g, h_{fg})$, together with $\mathsf{Cont}(f, h_f, q_f)$ and the displayed $\mathsf{Cont}(f, h_{fg}, q_g)$, gives $\mathsf{Cont}(q_f, h_g, q_g)$ by Definition 21.33. Comparing this ledger with the second quotient ledger $\mathsf{Cont}(g, h_g, q_g)$ and cancelling the common right tail h_g gives $g\mathsf{hsame}q_f$.

It remains to assemble the two ledgers required by $\mathsf{DerivativeMetricQuotient}(F_{fg}, z, h_{fg}, q_{fg}, d_{fg})$. For the quotient ledger, replace q_f and q_g by the results of $\mathsf{Cont}(f, h_f, q_f)$ and $\mathsf{Cont}(g, h_g, q_g)$. After reassociation, the canonical result of continuing F_{fg} by h_{fg} and the displayed quotient representative q_{fg} differ only by moving the unary factor g past the unary factor h_f. Unary continuation commutativity from Definition 9200.14, together with associativity from Definition 21.29, therefore identifies the two results, and continuation result transport gives $\mathsf{Cont}(F_{fg}, h_{fg}, q_{fg})$.

For the metric ledger, replace d_f and d_g by the results of $\mathsf{Cont}(h_f, q_f, d_f)$ and $\mathsf{Cont}(h_g, q_g, d_g)$. The ledger $\mathsf{Cont}(d_f, d_g, d_{fg})$ therefore classifies d_{fg} with the product of the two component metric results. The canonical metric result of continuing h_{fg} by q_{fg} is obtained from the step product followed by the quotient product. Reassociation and the same unary commutativity swap of q_f with h_g identify these two histories, so continuation result transport gives $\mathsf{Cont}(h_{fg}, q_{fg}, d_{fg})$. Repacking the unary carriers, the non-zero composite step, and the two continuation ledgers gives the derivative metric quotient. Finally, $\mathsf{Cont}(d_f, d_g, d_{fg})$ is the displayed distance product ledger. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeTasteGate single carrier alignment`

Theorem 122.22 (Derivative empty-function quotient zero-headed step exclusion)**.** *For histories* $z, h, q, dist, s$: BHist*, if*

$$\mathsf{DerivativeMetricQuotient}(\mathsf{e}, z, h, q, dist) \qquad \textit{and} \qquad h\mathsf{hsameE0}(s),$$

then $\perp$*.*

Proof. Apply Definition 122.5 to the derivative metric quotient. This gives $q\mathsf{hsame}h$. Composing with the displayed step classification gives $q\mathsf{hsameE0}(s)$. The quotient carrier field in Definition 122.1 gives $\mathsf{UnaryHistory}(q)$, the unary-generation carrier of Definition 28.1. By Definition 9195.7, this transports to $\mathsf{UnaryHistory}(\mathsf{E0}(s))$. This contradicts Definition 28.3. □

Lean4 ✓ `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient empty function zero step absurd`

Closure status for $\mathsf{NameCert}_{\mathsf{Derivative}\uparrow}$.

Constructive story: A derivative witness is carried as a five-history metric quotient row: the function code f, point z, nonzero step h, quotient q, and distance record $dist$ must all be unary where required, with $\mathsf{Cont}(f, h, q)$ displaying the difference quotient and $\mathsf{Cont}(h, q, dist)$ displaying the metric-distance ledger. The classifier is componentwise hsame transport of those visible histories, and the carrier excludes zero steps, zero-headed unary contradictions, and any quotient or distance value not produced by the two continuation rows. The source is this derivative metric quotient package, the pattern records complex-difference quotient readback, metric distance depth, visible-context readback, determinacy, chain-product composition, and the empty-function boundary rows, while the ledger keeps both continuations and every transported visible context in view. Nonemptiness is supplied by any displayed quotient row, the hsame laws give reflexivity, symmetry, and transitivity, and the transport theorem repacks classifier-related components into a fresh derivative quotient, so the five obligations stay inside the continuation calculus. A history h_0 lies in the signature gap of a $\mathsf{Derivative}^{\uparrow}$ token p when the source-domain accepts its metric quotient row, the signature relation pairs the quotient and distance package with an answer derivative history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the answer is the quotient endpoint whose nonzero-step and distance ledgers match the token's derivative data. The paper bridge reads this same package as the usual difference-quotient and chain-product interface, with the metric tolerance supplied by the visible distance row. The chapter registers the construction under the name $\mathsf{Derivative}^{\uparrow}$; it adds no full calculus, integration theory, differentiability-class hierarchy, or bridgeChecked external proof.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Derivative}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The DerivativeMetricQuotient chain-product theorem, transport rows, visible-context readback rows, determinacy rows, zero-step boundary, continuation ledger policy, and paper bridge form the mature certificate surface for this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Derivative}\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.DerivativeUp.DerivativeMetricQuotient\_chain\_product\_composition`

Bridge status: bridgeChecked

Not claimed: This chapter does not close full calculus, integration, or differentiability classes.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 123

A Concrete Naming Certificate for $\mathsf{Integral}^{\uparrow}$

The integral interface $\mathsf{Integral}^{\uparrow}$ packages a measure-respecting integration operation on suitable function carriers over $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92). It generalises the contour integral $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ (chapter 9340) to Riemann/Lebesgue settings on real and abstract carriers.

123.1 Carrier definition

Theorem 123.1 (Integral function-carrier obligation)**.** *An* $\mathsf{Integral}^{\uparrow}$ *carried endpoint is a* BHist *row whose visible source data are the cited* $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ *row, the function-carrier row of admissible integrands over that measure source, and the* $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ *comparison boundary. The endpoint carrier records only those named histories, their* Pkg *provenance, and their probe bundles; hidden partition, approximation, and convergence choices are not carrier equality.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralTasteGate single carrier alignment`
↪ variant `BEDC.Derived.IntegralUp.IntegralFunctionCarrier obligation`

Proof. Use the carrier field of the roadmap ten-field template. The traditional target is integration, the BEDC source row is the displayed $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ history, and the comparison source is the displayed $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ boundary.

The carrier field is then the function-carrier row over that measure source, together with the integral endpoint history. Repacking these rows through BHist and Pkg fixes the visible endpoint before classifier, operation, stability, ledger, exactness, and dependency rows are considered. □

Theorem 123.2 (Integral measure-respecting classifier obligation)**.** *The* $\mathsf{Integral}^{\uparrow}$ *classifier relates carried integrands exactly when the* $\mathsf{Measure}^{\uparrow}$*-visible domain row and the* $\mathsf{Contour}^{\uparrow}$ *comparison row agree under the same* hsame *and* Pkg *provenance. The comparison is therefore a BEDC classifier on displayed histories and packages, not a host quotient of integrands and not host function equality.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralMeasureRespectingClassifier obligation`
↪ variant `BEDC.Derived.IntegralUp.IntegralVisibleCarrierRow classifier boundary transport`

Proof. Apply the classifier field of the roadmap ten-field template to Definition 123.1. The source classifier is the visible $\mathsf{Measure}^{\uparrow}$ comparison on domain rows, while the boundary classifier is the visible $\mathsf{Contour}^{\uparrow}$ comparison on the contour-shaped row.

The integrand rows are compared only after those two source comparisons share the same hsame provenance and the same Pkg packaging discipline. This supplies the classifier, stability source, and exactness input fields without adding a host-level equivalence relation. □

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralVisibleCarrierRow measure respecting classifier`

Theorem 123.3 (Integral operation stability and ledger obligation)**.** *The* $\mathsf{Integral}^{\uparrow}$ *integration operation is admitted only with a measure-respecting stability row, ledger entries for hidden partition and approximation data, exactness rows for the displayed carrier and classifier, and dependency rows citing the* $\mathsf{Measure}^{\uparrow}$ *source and the* $\mathsf{Contour}^{\uparrow}$ *comparison boundary. No operation row is admitted without these stability, ledger, exactness, and dependency entries.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralOperationStabilityLedger obligation`

Proof. Use the operation and stability fields of the roadmap ten-field template after the carrier and classifier fields have been fixed. Measure-respecting stability says that transporting the $\mathsf{Measure}^{\uparrow}$ row and the admissible integrand row along the displayed classifier transports the operation endpoint as a BEDC history.

The ledger field records the hidden partition choices, approximation rows, and convergence witnesses used by the operation. The exactness field then states coverage, soundness, and completeness for the displayed carrier and classifier, while the dependency field cites exactly the $\mathsf{Measure}^{\uparrow}$ and $\mathsf{Contour}^{\uparrow}$ source rows consumed by the operation. □

123.2 The certificate

Theorem 123.4 (IntegralUp public NameCert surface)**.** *The carrier row of Definition 123.1, the classifier row of Definition 123.2, and the operation, stability, and ledger row of Definition 123.3 assemble the exported public* $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ *surface over the displayed* BHist, Pkg, hsame, $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ *dependencies. Its public rows are exactly the visible carrier, classifier, measure-respecting stability, operation ledger, exactness, and dependency rows; partitions, approximations, convergence witnesses, standard bridges, and host function equality remain ledger data or outside this surface.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralPublicNameCert surface`

Proof. Apply Definition 123.1. It fixes the visible BHist carrier, the Pkg provenance, and the named $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ dependencies read by the public $\mathsf{Integral}^{\uparrow}$ endpoint.

Apply Definition 123.2. It supplies the classifier row over the same hsame and Pkg provenance, so the public surface compares displayed histories rather than host-level functions.

Apply Definition 123.3. The roadmap ten-field template then reads the operation, measure-respecting stability, ledger, exactness, and dependency rows as visible public certificate fields. The same template keeps partitions, approximations, convergence witnesses, standard bridges, and host function equality as ledger or out-of-scope data, so the three obligation rows repack as the exported public $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ surface. □

Theorem 123.5 (IntegralUp conservative standard bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ *surface of Definition 123.4 admits a conservative standard interpretation with the following six fields.*

1. *The BEDC side is exactly the public surface of Definition 123.4.*

2. *The standard side is an ordinary integration-interface readback over the same displayed* $\mathsf{Measure}^{\uparrow}$ *and* $\mathsf{Contour}^{\uparrow}$ *rows.*
3. *Forward readback exposes only the visible carrier, classifier, measure-respecting stability, operation ledger, exactness, and dependency rows.*
4. *Reverse admission accepts only packages with those displayed rows.*
5. *Classifier preservation is the classifier row of Definition 123.2.*
6. *No host data leak through the bridge: partitions, approximations, convergence witnesses, and host function equality remain non-public ledger data by Definition 123.3.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralUp conservative standard bridge`

Proof. Apply Definition 123.1. It fixes the displayed $\mathsf{Measure}^{\uparrow}$ row, the admissible function-carrier row, the $\mathsf{Contour}^{\uparrow}$ comparison boundary, and the BHist endpoint read by both sides of the bridge.

Apply Definition 123.2. The classifier preserved by the standard reading is therefore the displayed hsame and Pkg comparison on measure and contour histories, not host function equality.

Apply Definition 123.3. Its ledger boundary keeps partitions, approximations, convergence witnesses, and operation presentations as non-public data, while its stability, exactness, and dependency rows are visible certificate fields.

Finally apply Definition 123.4. Forward readback can expose only the public rows listed there, and reverse admission can accept only packages carrying those same rows over the displayed $\mathsf{Measure}^{\uparrow}$ and $\mathsf{Contour}^{\uparrow}$ sources. Thus the standard integration-interface reading is conservative over the public $\mathsf{Integral}^{\uparrow}$ surface. □

Theorem 123.6 (IntegralUp mature consumer completeness)**.** *Every downstream consumer of the public* $\mathsf{Integral}^{\uparrow}$ *surface receives exactly the visible carrier, classifier, measure-respecting stability, operation ledger, exactness, dependency rows, and conservative bridge rows displayed by Definition 123.4 and Definition 123.5. Partitions, approximations, convergence witnesses, and host function equality remain outside the public consumer surface.*

Lean4 ✓ `BEDC.Derived.IntegralUp.IntegralUp mature consumer completeness`

Proof. Apply Definition 123.4. It fixes the visible carrier, classifier, measure-respecting stability, operation ledger, exactness, and dependency rows over the displayed BHist, Pkg, hsame, $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ sources.

Apply Definition 123.5. The bridge exposes only the same public rows to the standard integration-interface reading and admits back only packages carrying those displayed rows.

Composing the two surfaces gives the downstream consumer exactly the public and bridge rows just listed. The same two theorems keep partitions, approximations, convergence witnesses, and host function equality as ledger data or out-of-scope data, so no further host carrier is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$**.**

Constructive story: An integral endpoint is carried only through visible BEDC rows: a $\mathsf{Measure}^{\uparrow}$ source certificate, an admissible function-carrier row over that measure, the $\mathsf{Contour}^{\uparrow}$ comparison boundary, the endpoint history, its Pkg provenance, and the probe bundles that expose the operation surface. Two carried integrands classify together when the measure-visible domain row and contour comparison row agree under the same hsame and package provenance; hidden partitions, approximations, convergence choices, and host function equality are not classifier data. The source covers exactly those displayed measure-and-function packages, the pattern records measure-respecting stability, operation exactness, dependency rows, and the public consumer surface, and the ledger keeps partition choices, approximation rows, convergence witnesses, and operation presentations attached to the rows that use them. The visible carrier row

supplies nonemptiness whenever an accepted endpoint is present, the classifier obligation supplies reflexive, symmetric, and transitive comparison through the source certificates, and measure-respecting transport keeps classified endpoints inside the carrier. A history h lies in the signature gap of an Integral$^{\uparrow}$ token p when the source-domain accepts its measure, function, and contour package, the signature relation pairs the integration endpoint with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the token sees exactly the public integration rows and none of the hidden host choices. The conservative standard bridge reads those same public rows as an ordinary integration-interface package and admits back only packages carrying the same displayed rows. The chapter registers this measure-respecting operation surface under the name Integral$^{\uparrow}$; it does not add a convergence theorem, host integrand equality, or bridgeChecked verification.

Theory closure: TheoryGate(NameCert$_{\text{Integral}^{\uparrow}}$, maturePackageClosed)

Scope closed: The chapter includes the bridged public Integral$^{\uparrow}$ carrier, classifier, package, and consumer surface, grounded by Definition 123.4, Definition 123.5, Definition 123.1, Definition 123.2, and Definition 123.3, read through the displayed BHist, Pkg, hsame, NameCert$_{\text{Measure}^{\uparrow}}$, and NameCert$_{\text{Contour}^{\uparrow}}$ dependencies.

Formal verification: FormalStatus(NameCert$_{\text{Integral}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.IntegralUp.IntegralFunctionCarrier\_obligation`

Bridge status: paperBridge

Not claimed: This chapter does not close a convergence theorem, host function equality for integrands, or bridgeChecked verification.

Upgrade path: No stronger paper closure grade remains for this chapter; bridgeChecked verification is a separate formal-axis target.

Chapter 124

A Concrete Naming Certificate for $\mathsf{Fourier}^{\uparrow}$

The Fourier interface $\mathsf{Fourier}^{\uparrow}$ packages the constructive Fourier series and transform of suitable functions on $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$ (chapter 58) and $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$. It depends on $\mathsf{NameCert}_{\mathsf{InnerProduct}\uparrow}$ (chapter 86) and $\mathsf{NameCert}_{\mathsf{Integral}\uparrow}$ (chapter 123).

124.1 Carrier definition

124.2 The certificate

Theorem 124.1 (Fourier carrier obligation)**.** *A* $\mathsf{Fourier}^{\uparrow}$ *certificate supplies a* BHist*-indexed carrier of certified Fourier data over the named* $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$, $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$, $\mathsf{NameCert}_{\mathsf{InnerProduct}\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{Integral}\uparrow}$ *sources. Every finite observation of a coefficient, transform sample, inner-product probe, or integral witness is exposed through* ProbeBundle *and* SigRel *packages rather than through host-level functions.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation carrier obligation`

Proof. Unpack the four source certificates named by the chapter introduction. The roadmap ten-field template first asks for the carrier over which the later classifier, stability, ledger, and exactness rows are read.

For Fourier data this carrier can only be observed through finite coefficient and transform probes. Therefore the obligation surface records those probes as ProbeBundle entries with SigRel evidence, leaving analytic completion statements outside the carrier row. □

Theorem 124.2 (Fourier classifier obligation)**.** *The* $\mathsf{Fourier}^{\uparrow}$ *classifier identifies two Fourier data histories exactly when every certified finite coefficient observation and every certified finite transform observation are related by the source classifiers and are packaged through* sameSig *and* Pkg*. The classifier is therefore determined by the visible finite Fourier packages and not by an external equality of functions.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation classifier obligation`

Proof. Apply the classifier field of the roadmap ten-field template to the carrier row in Definition 124.1. Each observable Fourier datum has already been forced into a finite signature package, so classifier comparison is read at the level of source-certified signatures.

The $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, and $\mathsf{Integral}^{\uparrow}$ classifiers provide the component comparisons. Packaging through sameSig and Pkg records that the Fourier classifier is an internal BEDC classifier, not a host equality on functions or transforms. □

Theorem 124.3 (Fourier stability obligation)**.** *The* $\mathsf{Fourier}^{\uparrow}$ *stability row requires reflexivity, symmetry, transitivity, carrier transport, and stability under certified reindexing and integration rows inherited from* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$. *These stability clauses must act on the finite Fourier packages of the classifier row.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation stability obligation`

Proof. Start from the classifier obligation in Definition 124.2. Reflexivity, symmetry, and transitivity are the equivalence duties for the displayed classifier, while carrier transport states that the same certified observations remain readable after classifier-related history transport.

The reindexing and integration clauses are not independent analytic principles. They are inherited stability rows from the four named source certificates and are applied only to the finite packages exposed by the Fourier carrier. □

Theorem 124.4 (Fourier ledger obligation)**.** *The* $\mathsf{Fourier}^{\uparrow}$ *ledger records hidden truncation bounds, coefficient windows, orthogonality probes, integration witnesses, and convergence-modulus data. Each ledger item is attached to the finite package that uses it, and no item may be replaced by an unrecorded analytic convention.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation ledger obligation`

Proof. The ledger field of the roadmap ten-field template lists the data that can affect a certified observation without being the observation itself. Fourier coefficients and transforms carry truncation, window, orthogonality, integration, and modulus choices of exactly this kind.

Recording these choices beside the finite packages makes the later exactness row inspectable. It also fixes the boundary between certified Fourier data and analytic claims that have not been closed in this chapter. □

Theorem 124.5 (Fourier exactness obligation)**.** *The* $\mathsf{Fourier}^{\uparrow}$ *exactness row requires coverage, soundness, and completeness for the finite observation packages of Definition 124.1 through Definition 124.4. It does not claim Plancherel, inversion, or convergence closure.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation exactness obligation`

Proof. Unpack the exactness field of the roadmap ten-field template after the carrier, classifier, stability, and ledger rows have been fixed. Coverage says that each certified finite Fourier observation has a package, soundness says that package equality respects the displayed source classifiers, and completeness says that classifier-related finite observations are represented by internally same packages.

These are finite-observation obligations only. Plancherel, inversion, and convergence require additional analytic closure rows and are therefore named as outside the exactness surface recorded here. □

Theorem 124.6 (Fourier finite-observation standard bridge)**.** *The five checked finite-observation rows assemble a paper-side bridge from BEDC Fourier packages over* BHist, ProbeBundle, SigRel, sameSig, *and* Pkg *to the standard finite Fourier observation interface over* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$,

$\mathsf{NameCert}_{\mathsf{InnerProduct}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{Integral}^\uparrow}$. *This bridge preserves only finite coefficient and transform observations, and keeps truncation windows, integration witnesses, and convergence-modulus data in the ledger.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation standard bridge`

Proof. Use Definition 124.1 to read each observable coefficient, transform sample, inner-product probe, and integral witness as a finite BEDC package over BHist, ProbeBundle, and SigRel. Then use Definition 124.2 to transport the comparison of those packages through sameSig and Pkg into the source classifiers of $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{InnerProduct}^\uparrow}$, and $\mathsf{NameCert}_{\mathsf{Integral}^\uparrow}$.

Next apply Definition 124.3 to make the transport stable under certified reindexing and integration rows. Apply Definition 124.4 to retain all truncation windows, integration witnesses, orthogonality probes, and convergence-modulus data beside the finite observation that uses them.

Finally apply Definition 124.5. Its coverage, soundness, and completeness clauses identify exactly the finite observations carried across the bridge, while Plancherel, inversion, and convergence closure remain outside the preserved interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Fourier}^\uparrow}$**.**

Constructive story: A Fourier datum is carried only through finite observations over the named $\mathsf{S}^1\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{InnerProduct}^\uparrow$, and $\mathsf{Integral}^\uparrow$ sources: coefficient probes, transform samples, inner-product probes, and integral witnesses appear as ProbeBundle entries tied to SigRel evidence. The classifier compares two carried Fourier histories by checking that every finite observation package is related through the relevant source classifiers and through sameSig and Pkg, rather than by asserting equality of host functions or transforms. The source is the finite-observation package surface, the pattern records carrier, classifier, reindexing, integration-stability, and exactness obligations, and the ledger stores truncation bounds, coefficient windows, orthogonality probes, integration witnesses, and convergence-modulus data next to the finite package that uses them. Nonemptiness is provided whenever a source policy supplies an observable probe package, the classifier rows give reflexivity, symmetry, and transitivity through the finite signatures, and carrier transport follows because the same certified observations remain readable after source-classifier transport. A history h lies in the signature gap of a $\mathsf{Fourier}^\uparrow$ token p when the source-domain accepts its finite Fourier package, the signature relation pairs it with an answer observation and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h supplies exactly the coefficient or transform observations encoded by the token and the ledger records the windows and integration rows used to obtain them. The paper standard bridge reads the same packages as the ordinary finite Fourier observation interface while preserving only those finite observations. The chapter registers this finite-observation construction under the name $\mathsf{Fourier}^\uparrow$; it introduces no Plancherel theorem, inversion theorem, convergence closure, or Lean-checked standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Fourier}^\uparrow}$, maturePackageClosed)

Scope closed: The mature paper boundary is the finite-observation Fourier interface exported by the five checked rows and Definition 124.6 over $\mathsf{S}^1\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{InnerProduct}^\uparrow$, $\mathsf{Integral}^\uparrow$, BHist, ProbeBundle, SigRel, sameSig, and Pkg.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Fourier}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.FourierUp.FourierFiniteObservation\_exactness\_obligation`

Bridge status: paperBridge

Not claimed: This chapter does not close Plancherel, inversion, convergence closure, bridgeChecked verification, or a Lean StdBridge theorem.

Upgrade path: No stronger paper-side theory grade remains for this finite-observation boundary; only independent Lean-side bridgeChecked verification can raise the formal axis.

Theorem 124.7 (Fourier finite-observation subpacket restriction)**.** *Let F be a* $\mathsf{Fourier}^\uparrow$ *finite-observation package over the sources* $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$, $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{InnerProduct}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{Integral}^\uparrow}$. *Suppose Ω is a displayed finite selector on the observation rows of F, and suppose every selected row is retained together with its attached truncation window, coefficient-window row, orthogonality probe, integration witness, convergence-modulus ledger entry,* ProbeBundle *row, and* SigRel *evidence. Then the selected packet F_Ω is again a* $\mathsf{Fourier}^\uparrow$ *finite-observation package over the same four source certificates. Its carrier, classifier, ledger, and exactness rows are exactly the restrictions of the corresponding rows of F to the selected finite observations.*

Lean4 ✓ `BEDC.Derived.FourierUp.FourierFiniteObservation subpacket restriction`

Proof. Read F through the carrier obligation of Definition 124.1. Its visible observations are only finite coefficient observations, transform samples, inner-product probes, and integral witnesses, and each such observation is already presented as a ProbeBundle entry with SigRel evidence over the four named source certificates. Since Ω is a displayed finite selector on those rows, the carrier of F_Ω is obtained by keeping precisely the selected ProbeBundle entries and their selected SigRel evidence. No host Fourier function, ambient transform, or analytic completion coordinate is introduced by this selection.

The classifier row restricts along the same finite selector. By Definition 124.2, comparison of Fourier observations is read through the source classifiers and through sameSig and Pkg. For two observations retained by Ω, the comparison in F_Ω is the comparison already present in F. For an observation not retained by Ω, there is no row in the subpacket and therefore no classifier duty to discharge. Thus reflexive, symmetric, and transitive comparisons on the selected observations are inherited from the restricted classifier surface rather than from a new equality of functions.

The ledger row is also closed under the selector because the hypothesis retains each selected observation with every ledger item attached to it. Apply Definition 124.4: truncation bounds, coefficient windows, orthogonality probes, integration witnesses, and convergence-modulus data are recorded beside the finite package that uses them. Hence restricting the observation row without dropping its attached ledger entries leaves a complete local ledger for every retained coefficient observation, transform sample, inner-product probe, and integral witness. Ledger entries attached only to omitted observations are outside F_Ω and are not required for any retained read.

Finally apply Definition 124.5. Coverage for F_Ω holds because each retained observation was covered in F and its covering package is one of the rows selected by Ω. Soundness holds because any equality or source-classifier comparison used inside F_Ω is the same comparison already sound in F, restricted to retained rows. Completeness holds for the selected interface because whenever two retained finite observations are classifier-related in the restricted packet, their sameSig and Pkg witnesses are exactly the retained witnesses inherited from F.

The source certificates are unchanged throughout the construction: Ω selects rows inside the already certified $\mathsf{Fourier}^\uparrow$ packet and does not replace $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$, $\mathsf{NameCert}_{\mathsf{Real}\uparrow}$, $\mathsf{NameCert}_{\mathsf{InnerProduct}\uparrow}$, or $\mathsf{NameCert}_{\mathsf{Integral}\uparrow}$. Therefore F_Ω satisfies the same finite-observation package obligations over the same sources. The conclusion is only this finite subpacket closure; it does not assert Plancherel, inversion, convergence closure, or any analytic Fourier theorem. □

Chapter 125

A Concrete Naming Certificate for $\mathsf{Residue}^{\uparrow}$

The residue interface $\mathsf{Residue}^{\uparrow}$ packages the residue at an isolated singularity of a holomorphic function and the residue-theorem witness relating contour integrals to residue sums. It depends on $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ (chapter 68) and $\mathsf{NameCert}_{\mathsf{Contour}^{\uparrow}}$ (chapter 9340).

125.1 Carrier definition

Definition 125.1 (Residue pole data)**.** *The predicate* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *records an open-disk witness* $\mathsf{HolomorphicOpenDisk}(z_0, r, p, g)$*, complex-history carriers for the integral* I *and residue* ρ*, a complex-pair carrier for* f*, and the continuation witness* $\mathsf{Cont}(I, \rho, f)$*.*

Lean4 (definition) `BEDC.Derived.ResidueUp.ResiduePoleData`

Lean4 ✓ `BEDC.Derived.ResidueUp.ResidueTasteGate single carrier alignment`

Theorem 125.2 (Residue pole data under hsame)**.** *If all seven histories in* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *are transported along* hsame*, then the transported histories again satisfy* ResiduePoleData*, and the transported integral and residue continue to the transported function history.*

Proof. Transport the open-disk witness through holomorphic-disk hsame-stability. Transport the complex carriers for I, ρ, and the product carrier for f through the product-history carrier stability, then transport $\mathsf{Cont}(I, \rho, f)$ componentwise along the same hsame witnesses. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData hsame transport`

Theorem 125.3 (Residue pole data empty function endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds and* $f\mathsf{hsamee}$*, then both residue endpoints are empty:*

$$I\mathsf{hsamee} \wedge \rho\mathsf{hsamee}.$$

Proof. Destructure the residue pole data to expose $\mathsf{Cont}(I, \rho, f)$. Transport the continuation result along $f\mathsf{hsamee}$, then apply empty-result inversion for BEDC continuations to recover the two empty endpoints. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData empty function endpoints`

Theorem 125.4 (Residue pole data empty function endpoints iff)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds, then the function history is empty exactly when both residue endpoints are empty:*

$$f\mathsf{hsamee} \iff I\mathsf{hsamee} \wedge \rho\mathsf{hsamee}.$$

Proof. The forward implication is the empty-function endpoint theorem. Conversely, unpack the residue pole data to expose $\mathsf{Cont}(I, \rho, f)$. Compare this continuation with $\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e})$ using the supplied endpoint equalities; continuation result respect for hsame gives $f\mathsf{hsamee}$. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData empty function endpoints iff`

Theorem 125.5 (Residue pole data residue suffix closure)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* q *is unary, and* $\mathsf{Cont}(f, q, f_q)$*, then there is a residue history* ρ_q *such that* $\mathsf{ResiduePoleData}(f_q, z_0, r, p, g, I, \rho_q)$*, with* $\mathsf{Cont}(\rho, q, \rho_q)$ *and* $\mathsf{Cont}(I, \rho_q, f_q)$*.*

Proof. Unpack the residue data to obtain $\mathsf{Cont}(I, \rho, f)$. Continuation associativity against $\mathsf{Cont}(f, q, f_q)$ supplies ρ_q with $\mathsf{Cont}(\rho, q, \rho_q)$ and $\mathsf{Cont}(I, \rho_q, f_q)$. The residue carrier is closed under the unary suffix q, and the product carrier for f_q is rebuilt from the integral carrier, the shifted residue carrier, and the final continuation. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData residue suffix closure`

Theorem 125.6 (Residue pole data suffix closure with empty function endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g,$ *holds,* q *is unary,* $\mathsf{Cont}(f, q, f_q)$*, and* $f_q\mathsf{hsamee}$*, then there is a residue history* ρ_q *such that* $\mathsf{ResiduePoleData}(f_q, z_0, r,$ $\mathsf{Cont}(\rho, q, \rho_q)$*, and*

$$I\mathsf{hsamee} \wedge \rho_q\mathsf{hsamee}.$$

Proof. Apply suffix closure to obtain the transported residue witness ρ_q and the closed residue data over f_q. The empty-function endpoint theorem then applies to that closed data and $f_q\mathsf{hsamee}$, identifying both the integral and the transported residue with e. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData suffix closure empty function endpoints`

Theorem 125.7 (Residue pole data empty suffix reflects endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* $\mathsf{Cont}(f, q, f_q)$*, and* $f_q\mathsf{hsamee}$*, then the integral endpoint, the residue endpoint, and the suffix are all empty:*

$$I\mathsf{hsamee} \wedge \rho\mathsf{hsamee} \wedge q\mathsf{hsamee}.$$

Proof. Transport $\mathsf{Cont}(f, q, f_q)$ along $f_q\mathsf{hsamee}$ and apply empty-result inversion for continuations. This gives $f\mathsf{hsamee}$ and $q\mathsf{hsamee}$. The empty-function endpoint theorem for residue pole data then reflects $f\mathsf{hsamee}$ back to the integral and residue endpoints. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData empty suffix reflects endpoints`

Theorem 125.8 (Residue pole data prefix empty reflects endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* $\mathsf{Cont}(q, f, q_f)$*, and* $q_f\mathsf{hsamee}$*, then the prefix, integral endpoint, and residue endpoint are all empty:*

$$q\mathsf{hsamee} \wedge I\mathsf{hsamee} \wedge \rho\mathsf{hsamee}.$$

Proof. Transport $\mathsf{Cont}(q, f, q_f)$ along $q_f\mathsf{hsamee}$ and apply empty-result inversion for continuations. This gives $q\mathsf{hsamee}$ and $f\mathsf{hsamee}$. The empty-function endpoint theorem for residue pole data then reflects $f\mathsf{hsamee}$ to the integral and residue endpoints. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData prefix empty reflects endpoints`

Theorem 125.9 (Residue pole data suffix empty function endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* q *is unary,* $\mathsf{Cont}(f, q, f_q)$*, and* $f_q\mathsf{hsamee}$*, then* $I\mathsf{hsamee}$ *and there is a shifted residue* ρ_q *such that*

$$\mathsf{Cont}(\rho, q, \rho_q) \wedge \rho_q\mathsf{hsamee}.$$

Proof. Apply residue suffix closure to obtain the shifted pole data and the residue suffix witness. The empty-function endpoint theorem applied to the shifted data and $f_q\mathsf{hsamee}$ identifies I and ρ_q with the empty history, and the residue suffix witness supplies the displayed continuation. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData suffix empty function endpoints`

Theorem 125.10 (Residue pole data integral prefix closure)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* q *is unary, and* $\mathsf{Cont}(q, f, q_f)$*, then there is an integral history* I_q *such that* $\mathsf{ResiduePoleData}(q_f, z_0, r, p, g, I_q, \rho)$ *with* $\mathsf{Cont}(q, I, I_q)$ *and* $\mathsf{Cont}(I_q, \rho, q_f)$*.*

Proof. Take $I_q = \mathsf{append}(q, I)$. The complex integral carrier is closed under unary prefixing. The two continuation witnesses $\mathsf{Cont}(I, \rho, f)$ and $\mathsf{Cont}(q, f, q_f)$, together with append associativity, identify $\mathsf{Cont}(I_q, \rho, q_f)$. The product carrier for q_f is then rebuilt from the prefixed integral carrier, the original residue carrier, and that continuation. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData integral prefix closure`

Theorem 125.11 (Residue pole data integral prefix empty function endpoints)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* q *is unary,* $\mathsf{Cont}(q, f, q_f)$*, and* $q_f\mathsf{hsamee}$*, then there is an integral history* I_q *such that* $\mathsf{ResiduePoleData}(q_f, z_0, r, p, g, I_q, \rho)$*,* $\mathsf{Cont}(q, I, I_q)$*, and*

$$I_q\mathsf{hsamee} \wedge \rho\mathsf{hsamee}.$$

Proof. Apply integral prefix closure to obtain the prefixed integral witness I_q and the residue pole data over q_f. The empty-function endpoint theorem applied to that prefixed pole data and $q_f\mathsf{hsamee}$ identifies both I_q and ρ with the empty history, while the prefix continuation $\mathsf{Cont}(q, I, I_q)$ is preserved from the closure witness. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData integral prefix empty function endpoints`

125.2 The certificate

Theorem 125.12 (Residue pole data integral suffix closure)**.** *If* $\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho)$ *holds,* q *is unary, and* $\mathsf{Cont}(f, q, f_q)$*, then there is an integral history* I_q *such that* $\mathsf{ResiduePoleData}(f_q, z_0, r, p, g, I_q, \rho)$ *with* $\mathsf{Cont}(I, q, I_q)$ *and* $\mathsf{Cont}(I_q, \rho, f_q)$*.*

Proof. Unpack the residue pole data of Definition 125.1. This gives the open-disk witness, complex carrier witnesses for I and ρ, the product carrier for f, and the continuation $\mathsf{Cont}(I, \rho, f)$. Set $I_q = \mathsf{append}(I, q)$. The right unary context closure for complex carriers from Definition 39.12 gives a complex carrier for I_q, and $\mathsf{Cont}(I, q, I_q)$ is the defining continuation.

It remains to identify the final continuation. By Definition 39.3, the residue carrier makes ρ unary. Since q is unary, unary append commutativity from Definition 9200.9 swaps q and ρ. Let J be the history supplied by the defining continuation of I_q by ρ, so $\mathsf{Cont}(I_q, \rho, J)$. Combining

this continuation with the swap, append associativity, and the two continuations $\mathsf{Cont}(I, \rho, f)$ and $\mathsf{Cont}(f, q, f_q)$ gives

$$J\mathsf{hsameappend}(\mathsf{append}(I, q), \rho)\mathsf{hsameappend}(\mathsf{append}(I, \rho), q)\mathsf{hsame}f_q.$$

The continuation result transport theorem Definition 4.40 applied to $\mathsf{Cont}(I_q, \rho, J)$ and $J\mathsf{hsame}f_q$ gives $\mathsf{Cont}(I_q, \rho, f_q)$. The same open-disk witness and the original residue carrier are retained; the product carrier for f_q is rebuilt from the complex carriers for I_q and ρ together with $\mathsf{Cont}(I_q, \rho, f_q)$. Hence $\mathsf{ResiduePoleData}(f_q, z_0, r, p, g, I_q, \rho)$ holds, with the two displayed continuations. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData integral suffix closure`

Theorem 125.13 (Residue pole data result determinism)**.** *If two residue pole data witnesses share the displayed pole histories z_0, r, p, g, I, ρ and have function histories f and f', so that*

$$\mathsf{ResiduePoleData}(f, z_0, r, p, g, I, \rho) \quad \textit{and} \quad \mathsf{ResiduePoleData}(f', z_0, r, p, g, I, \rho),$$

then $f\mathsf{hsame}f'$.

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData result determinism with continuations`

Proof. Unpack both residue pole data witnesses using Definition 125.1. The first witness exposes a continuation ledger $C : \mathsf{Cont}(I, \rho, f)$, and the second exposes a continuation ledger $C' : \mathsf{Cont}(I, \rho, f')$. Apply Definition 4.42 to C and C' with the reflexive endpoint comparisons $I\mathsf{hsame}I$, $\rho\mathsf{hsame}\rho$, and $f\mathsf{hsame}f$. The transported first ledger has the same source, right endpoint, and result as C, while the classification clause compares that result with the result of C'. Hence $f\mathsf{hsame}f'$. □

Lean4 ✓ `BEDC.Derived.ResidueUp.ResiduePoleData result determinism`

Closure status for $\mathsf{NameCert}_{\mathsf{Residue}^\uparrow}$.

Constructive story: A $\mathsf{Residue}^\uparrow$ packet is generated from residue pole data, a holomorphic disk row, contour and integral rows, componentwise transport, continuation replay, and local naming for the residue pole carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Residue}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 125.1, Definition 125.2, Definition 125.3, Definition 125.4, Definition 125.5, Definition 125.10, Definition 125.12, and Definition 125.13 record the mature finite residue-pole packet over Holomorphic, Contour, complex-history carrier, hsame, Cont, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Residue}^\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ResidueUp.ResiduePoleData\_hsame\_transport`

Bridge status: none

Not claimed: This chapter does not close the full residue theorem, analytic continuation, or the standard complex analysis bridge.

Upgrade path: Bridge checking remains the next verification-axis obligation: a checked standard bridge must identify this mature residue-pole packet with the accepted complex-analysis residue theorem interface while preserving the Holomorphic, Contour, complex-history carrier, hsame, Cont, and NameCert rows.

StreamNameUp is the L10 countable-name gateway consumed by chapter 301, chapter 291, and chapter 38 before any RealUp seal is public. The route exports only finite BHist, hsame, Cont, Pkg, and NameCert rows, not quotient stream equality, selected limits, host real equality, or ambient completeness.

Chapter 126

A Concrete Naming Certificate for $\mathsf{StreamName}^{\uparrow}$

The stream-name interface $\mathsf{StreamName}^{\uparrow}$ packages an abstract sequential carrier underlying the Bishop-style L10 real-number construction. It is the lowest-level data on which $\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{CompactMetric}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ are built.

The same carrier is the finite observation spine later consumed by chapter 301 and sealed by chapter 38. Those downstream chapters may read only the displayed stream windows, hsame transport, Cont replay, provenance, and local NameCert rows exposed here.

126.1 Carrier definition

Definition 126.1 (Rational stream-name carrier)**.** *For a certified observation map* $s : \mathsf{Hist} \to \mathsf{Hist}$*, the rational stream-name carrier is the pointwise rational-history carrier over unary BEDC observation histories:*

$$\mathsf{RatStreamNameCarrier}(s) \iff \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{RatHistoryCarrier}(s(n)).$$

Lean4 (definition) `BEDC.Derived.StreamNameUp.RatStreamNameCarrier`

Definition 126.2 (Rational stream-name classifier)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, the rational stream-name classifier bundles the two endpoint stream carriers with the pointwise rational-history classifier over the same unary BEDC observation histories:*

$$\mathsf{RatStreamNameClassifier}(s,t) \iff \mathsf{RatStreamNameCarrier}(s) \wedge \mathsf{RatStreamNameCarrier}(t) \wedge \forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{Ra}$$

Lean4 (definition) `BEDC.Derived.StreamNameUp.RatStreamNameClassifier`

Theorem 126.3 (Rational stream-name carrier pointwise hsame transport)**.** *For certified observation maps* $s, s' : \mathsf{Hist} \to \mathsf{Hist}$*, if* $\mathsf{RatStreamNameCarrier}(s)$ *and*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)),$$

then $\mathsf{RatStreamNameCarrier}(s')$*.*

Proof. Unfold Definition 126.1. It remains to show that each certified unary observation n carries a rational-history witness for $s'(n)$.

Fix such an observation, together with its unary witness ν. The source carrier supplies $\mathsf{RatHistoryCarrier}(s(n))$ at this same observation index.

The pointwise stability hypothesis supplies $\mathsf{hsame}(s(n), s'(n))$. Apply the RatUp carrier transport theorem RatHistoryCarrier_hsame_transport to the source rational-history carrier and this transport witness, obtaining $\mathsf{RatHistoryCarrier}(s'(n))$.

Since the observation was arbitrary, the transported rational-history carrier rows repack into the universal carrier row for $\mathsf{RatStreamNameCarrier}(s')$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameCarrier pointwise hsame transport`

Definition 126.4 (StreamName pointwise reindexing packet)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, a $\mathsf{StreamNamePointwiseReindexing}(s, t)$ packet is a finite display*

$$(\rho, U, H, C, P)$$

where $\rho : \mathsf{Hist} \to \mathsf{Hist}$ is a certified unary-history reindexing map, U sends each $\mathsf{UnaryHistory}(n)$ row to a displayed $\mathsf{UnaryHistory}(\rho(n))$ row, H records pointwise hsame *transport for the endpoint histories $s(\rho(n))$ and $t(\rho(n))$, C records the displayed* Cont *replay of finite observation requests along ρ, and P is the* Pkg *provenance over* BHist, UnaryHistory, RatStreamNameCarrier, RatStreamNameClassifier, ProbeBundle, hsame, *and* Cont. *The packet has no coordinate for host sequence equality, quotient stream equality, selected limits, or ambient completion data.*

Theorem 126.5 (StreamName finite-window pullback stability)**.** *Let $s, t : \mathsf{Hist} \to \mathsf{Hist}$ be certified observation maps, let $\mathcal{P} : \mathsf{ProbeBundle}$ be a finite probe bundle, and let (ρ, U, H, C, P) be a $\mathsf{StreamNamePointwiseReindexing}(s, t)$ packet. If $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ holds and $\mathcal{P}^\rho$ is the displayed pullback bundle whose certified observations are exactly the n with $\rho(n)$ displayed in $\mathcal{P}$, then $\mathsf{RatStreamNameFiniteWindowClassifier}(s \circ \rho, t \circ \rho, \mathcal{P}^\rho)$ holds.*

For each pulled-back observation, the endpoint $\mathsf{Rat}^\uparrow$ classifier row is the same pointwise row read at $\rho(n)$ through U, H, C, P. The pullback introduces no new endpoint classifier, quotient stream equality, RegSeqRat seal, Real seal, host sequence equality, selected limit, or ambient completion coordinate.

Proof. Unfold the finite-window classifier for $\mathcal{P}^\rho$. Fix a certified unary observation n in the pulled-back bundle, together with its $\mathsf{UnaryHistory}(n)$ witness and its displayed membership row.

Use the reindexing packet. The row U sends the unary witness at n to a unary witness at $\rho(n)$, and the pullback membership row says that this reindexed observation lies in the original bundle $\mathcal{P}$.

Read the original finite-window classifier at $\rho(n)$. It returns the pointwise $\mathsf{RatHistoryClassifier}(s(\rho(n)), t(\rho(n)))$ row for the two endpoint rational histories.

Finally replay the endpoint row through H and C. These rows are componentwise hsame and displayed continuation data for the same rational histories, so the pointwise classifier repacks as the endpoint row of $\mathsf{RatStreamNameFiniteWindowClassifier}(s \circ \rho, t \circ \rho, \mathcal{P}^\rho)$. Since the packet contains only U, H, C, P over the listed BEDC rows, none of the refused objects can enter the pullback. □

Theorem 126.6 (StreamName constant section transport)**.** *Let $q : \mathsf{Hist}$ be a certified $\mathsf{Rat}^\uparrow$ history row, and let $s_q : \mathsf{Hist} \to \mathsf{Hist}$ be the constant certified stream with $s_q(n) = q$ at every displayed unary observation. For every $\mathsf{StreamNamePointwiseReindexing}(s_q, s_q)$ packet (ρ, U, H, C, P) and every finite probe bundle $\mathcal{P}$, the pulled-back constant section over $\mathcal{P}^\rho$ is again certified by the same constant* RatStreamNameCarrier *and the same pointwise $\mathsf{RatHistoryClassifier}(q, q)$ rows.*

Thus constant certified streams are stable under displayed pointwise pullback. The transport reads only UnaryHistory*,* BHist*,* RatStreamNameCarrier*,* RatStreamNameClassifier*,* ProbeBundle*,* hsame*,* Cont*, and* Pkg *rows, and it introduces no selected limit, quotient stream equality, host sequence equality, RegSeqRat seal, Real seal, or ambient completion datum.*

Proof. Project the constant stream carrier. At every displayed unary observation, both endpoints of the classifier are the same rational history row q.

Apply Definition 126.5 to the constant pair s_q, s_q. The pulled-back bundle $\mathcal{P}^\rho$ reads each observation by first applying ρ and then reading the same constant endpoint row.

The pointwise classifier is unchanged. Since $s_q(\rho(n)) = q$ and $s_q(n) = q$ at each displayed observation, the endpoint row after pullback is still $\mathsf{RatHistoryClassifier}(q, q)$, transported only by the supplied hsame and Cont rows.

The package provenance P names no further stream or completion coordinate. Hence the pulled-back section is the same constant certified stream on the pulled-back finite window. □

Ten-field fit: traditional target rational stream-name carrier transport; BEDC source RatUp plus StreamNameUp over BHist; carrier RatStreamNameCarrier; classifier untouched; stability pointwise hsame; ledger the observation index and transport witness; core theorem Definition 126.3; exactness preservation of the carrier predicate; standard bridge RegSeqRat/RealUp stream transport; status proof target.

Theorem 126.7 (Rational stream-name successor reindexing stability)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, define the successor observation schedules* $s^+, t^+ : \mathsf{Hist} \to \mathsf{Hist}$ *by*

$$s^+(n) = s(\mathsf{BHist.e1}(n)) \quad \textit{and} \quad t^+(n) = t(\mathsf{BHist.e1}(n)).$$

If $\mathsf{RatStreamNameCarrier}(s)$*, then* $\mathsf{RatStreamNameCarrier}(s^+)$*. If* $\mathsf{RatStreamNameClassifier}(s, t)$*, then* $\mathsf{RatStreamNameClassifier}(s^+, t^+)$ *and, for every probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*,* RatStreamNameFiniteWindowCla

Proof. Let $r(n) = \mathsf{BHist.e1}(n)$. Unary successor closure sends every witness $\mathsf{UnaryHistory}(n)$ to a witness $\mathsf{UnaryHistory}(r(n))$.

Apply Definition 126.19 with this concrete reindexer r and the identity RatUp-preserving point map. The reindexing clauses turn $\mathsf{RatStreamNameCarrier}(s)$ into the carrier row for $s^+ = s \circ r$, and turn $\mathsf{RatStreamNameClassifier}(s, t)$ into the classifier row for s^+, t^+.

Finally fix a probe bundle $\mathcal{P}$. Applying Definition 126.22 to the successor classifier projects every bundled certified unary observation to the RatUp point classifier $\mathsf{RatHistoryClassifier}(s^+(n), t^+(n))$, which is exactly the finite-window classifier row for s^+, t^+ on $\mathcal{P}$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName successor reindexing stability`

Ten-field fit: traditional target tail/successor rational stream observations; BEDC source BHist, UnaryHistory, RatUp, and StreamNameUp; carrier RatStreamNameCarrier after successor reindexing; classifier RatStreamNameClassifier after successor reindexing; stability successor reindexing plus finite-window restriction; ledger successor observation token and bundled probe membership; core theorem Definition 126.7; exactness preservation of pointwise RatUp classification; standard bridge rational sequence tail observation; status proof target.

Theorem 126.8 (Rational stream-name transported successor reindexing stability)**.** *For certified observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, define the transported successor observation schedules* $(s')^+, (t')^+ : \mathsf{Hist} \to \mathsf{Hist}$ *by*

$$(s')^+(n) = s'(\mathsf{BHist.e1}(n)) \quad \textit{and} \quad (t')^+(n) = t'(\mathsf{BHist.e1}(n)).$$

If $\mathsf{RatStreamNameClassifier}(s,t)$ *and*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n)),$$

and

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)),$$

then $\mathsf{RatStreamNameClassifier}((s')^+, (t')^+)$ *and, for every probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$, $\mathsf{RatStreamNameFiniteWind}$

Proof. First transport the endpoint carriers. Apply Definition 126.3 to the left endpoint of $\mathsf{RatStreamNameClassifier}(s,t)$ and the pointwise hsame row from s to s'. Apply the same theorem to the right endpoint and the pointwise hsame row from t to t'.

Next transport the pointwise classifier. At a certified unary index n, the source classifier gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$. The two hsame rows move its endpoints to $s'(n)$ and $t'(n)$, so the RatUp classifier transport repacks the pointwise clause for $\mathsf{RatStreamNameClassifier}(s', t')$.

Now apply Definition 126.7 to the transported classifier and the transported left carrier. Its successor reindexing row gives $\mathsf{RatStreamNameClassifier}((s')^+, (t')^+)$.

Finally fix a probe bundle $\mathcal{P}$. The finite-window projection in Definition 126.7 reads every bundled unary probe through the same transported successor schedules, giving $\mathsf{RatStreamNameFiniteWindowClassifier}((s$

□

Ten-field fit: traditional target transported rational stream-name successor observations; BEDC source RatUp, StreamNameUp, BHist, and $\mathsf{UnaryHistory}$; carrier transported $\mathsf{RatStreamNameCarrier}$ endpoints; classifier transported $\mathsf{RatStreamNameClassifier}$ endpoints; stability pointwise hsame followed by successor reindexing; ledger unary successor observation token and $\mathsf{ProbeBundle}$ membership; core theorem Definition 126.8; exactness preservation of pointwise RatUp classification; standard bridge finite-window tail observation for RegSeqRat and RealUp; status paper theorem awaiting formal target.

Theorem 126.9 (Rational stream-name finite-window subwindow restriction)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and probe bundles* $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$*, suppose that* $\mathsf{RatStreamNameFiniteWindowClass}$ *holds and that every certified unary observation displayed by* $\mathcal{Q}$ *is also displayed by* $\mathcal{P}$*:*

$$\forall n,\ \mathsf{InBundle}(\mathcal{Q}, n) \wedge \mathsf{UnaryHistory}(n) \to \mathsf{InBundle}(\mathcal{P}, n) \wedge \mathsf{UnaryHistory}(n).$$

Then $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ *holds.*

Proof. Unfold the finite-window classifier. To certify the $\mathcal{Q}$ window, fix a bundled observation n together with its $\mathsf{InBundle}(\mathcal{Q}, n)$ row and its $\mathsf{UnaryHistory}(n)$ witness.

The displayed inclusion row sends this same certified unary observation into the $\mathcal{P}$ window. The original $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ can therefore be read at n.

Projecting that read gives the pointwise RatUp classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$. Since the choice of the $\mathcal{Q}$ observation was arbitrary, these projected point classifiers repack into $\mathsf{RatStreamNameFiniteWind}$

□

Ten-field fit: traditional target finite rational stream observation windows for RegSeqRat and RealUp; BEDC source RatUp, StreamNameUp, BHist, $\mathsf{UnaryHistory}$, and $\mathsf{ProbeBundle}$; carrier the endpoint $\mathsf{RatStreamNameCarrier}$ rows already read by the finite window; classifier $\mathsf{RatStreamNameFiniteWindowClass}$ stability subwindow restriction along the displayed probe inclusion; ledger the two bundle membership rows and the unary observation certificate; core theorem Definition 126.9; exactness pointwise projection to $\mathsf{RatHistoryClassifier}$; standard bridge finite prefix consumers in RegSeqRat and RealUp; status paper theorem awaiting formal target.

Theorem 126.10 (Rational stream-name finite-window union classifier)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and probe bundles* $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$*, suppose that* $\mathsf{RatStreamNameFiniteWindowClassifier}(s,$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ *hold. Let* $\mathcal{P} \cup \mathcal{Q}$ *be the displayed union bundle whose members are exactly the certified unary observations coming from* $\mathcal{P}$ *or from* $\mathcal{Q}$*:*

$$\mathsf{InBundle}(\mathcal{P}\cup\mathcal{Q}, n)\wedge\mathsf{UnaryHistory}(n) \quad \textit{iff} \quad (\mathsf{InBundle}(\mathcal{P}, n)\wedge\mathsf{UnaryHistory}(n))\vee(\mathsf{InBundle}(\mathcal{Q}, n)\wedge\mathsf{UnaryHistory}(n)).$$

Then $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P} \cup \mathcal{Q})$ *holds.*

Proof. Unfold the finite-window classifier for the union bundle. Fix a certified unary observation n in $\mathcal{P} \cup \mathcal{Q}$ and read the displayed union membership row.

There are two cases. If the row comes from $\mathcal{P}$, the classifier $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ projects at n to the pointwise RatUp classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

If the row comes from $\mathcal{Q}$, the second finite-window classifier gives the same kind of pointwise certificate $\mathsf{RatHistoryClassifier}(s(n), t(n))$ at the same unary observation.

In either case the required pointwise classifier is available for the fixed union observation. Since the observation was arbitrary, these pointwise RatHistoryClassifier rows repack into RatStreamNameFiniteWindowClas $\mathcal{Q}$). □

Ten-field fit: traditional target finite rational stream-name observation composition for RegSeqRat and RealUp; BEDC source RatUp, StreamNameUp, BHist, UnaryHistory, and ProbeBundle; carrier the same displayed RatStreamNameCarrier endpoints read by the two windows; classifier RatStreamNameFiniteWindowClassifier on $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{P} \cup \mathcal{Q}$; stability union membership case analysis over certified unary observations; ledger the two bundle membership rows and their shared unary observation certificate; core theorem Definition 126.10; exactness pointwise projection to RatHistoryClassifier; standard bridge finite-prefix consumer for RegSeqRat and RealUp; status paper theorem awaiting formal target.

Theorem 126.11 (Rational stream-name finite-window union endpoint exhaustion)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and certified probe bundles* $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$*, suppose that* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ *hold. Let* $\mathcal{P} \cup \mathcal{Q}$ *be the union bundle whose certified unary membership rows are exactly those displayed in Definition 126.10. Then every endpoint row visible to a RegSeqRat or RealUp finite-window consumer over* $\mathcal{P} \cup \mathcal{Q}$ *is generated by either an* $\mathsf{InBundle}(\mathcal{P}, n)$ *row or an* $\mathsf{InBundle}(\mathcal{Q}, n)$ *row, together with the same* $\mathsf{UnaryHistory}(n)$ *witness. In both cases the endpoint certificate is exactly the pointwise classifier* $\mathsf{RatHistoryClassifier}(s(n), t(n))$*.*

Consequently the union consumer has no additional endpoint source beyond the two displayed bundle memberships, and it introduces no RegSeqRat regularity row, RealUp seal, quotient, or host-index equality row.

Proof. Fix a finite-window consumer row over $\mathcal{P}\cup\mathcal{Q}$. By the displayed union membership condition in Definition 126.10, the visible certified unary observation has a witness $\mathsf{UnaryHistory}(n)$ and lies in one of the two generated membership cases needed for the argument: it is read from $\mathsf{InBundle}(\mathcal{P}, n)$ or from $\mathsf{InBundle}(\mathcal{Q}, n)$.

In the $\mathcal{P}$ case, apply Definition 126.10 to view the row as part of the union classifier and apply Definition 126.9 along the displayed inclusion of the $\mathcal{P}$ subwindow. The resulting endpoint is the pointwise certificate $\mathsf{RatHistoryClassifier}(s(n), t(n))$ carried with the same $\mathsf{UnaryHistory}(n)$ witness.

The $\mathcal{Q}$ case is identical, using the displayed inclusion of the $\mathcal{Q}$ subwindow. The subwindow restriction again returns precisely $\mathsf{RatHistoryClassifier}(s(n), t(n))$ at the same unary observation.

These two cases exhaust the union membership row. Since the only endpoint certificates obtained are the two pointwise RatUp classifier reads, the consumer cannot acquire a RegSeqRat regularity condition, a RealUp seal, a quotient row, or a host-index equality row from the union construction. □

Theorem 126.12 (Rational stream-name classifier unary context one-tail pair readback)**.** *If* $\mathsf{RatStreamNameClassifier}(s, t)$ *holds at a unary observation* n*, matching unary prefix and tail contexts are presented by continuation displays, and the two outputs are one-extension histories, then the displayed tails are unary and internally the same.*

Proof. Read the stream-name classifier at the certified observation n. This gives the RatUp point classifier for $s(n)$ and $t(n)$. Apply the RatUp continuation context one-tail pair readback theorem to that point classifier and the four continuation displays. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameClassifier unary context e1 pair readback`

126.2 The certificate

Theorem 126.13 (Rational stream-name certificate fields)**.** *For certified observation maps* $s, t, u : \mathsf{Hist} \to \mathsf{Hist}$*, the carrier and classifier above supply the rational stream-name certificate fields:*

1. $\mathsf{RatStreamNameCarrier}(s)$ *implies* $\mathsf{RatStreamNameClassifier}(s, s)$.
2. $\mathsf{RatStreamNameClassifier}(s, t)$ *implies* $\mathsf{RatStreamNameClassifier}(t, s)$.
3. $\mathsf{RatStreamNameClassifier}(s, t)$ *and* $\mathsf{RatStreamNameClassifier}(t, u)$ *imply* $\mathsf{RatStreamNameClassifier}(s, u)$.
4. $\mathsf{RatStreamNameClassifier}(s, t)$ *implies, for every* n *with* $\mathsf{UnaryHistory}(n)$*, both* $\mathsf{RatHistoryCarrier}(s(n))$ *and* $\mathsf{RatHistoryCarrier}(t(n))$.
5. *If* $\mathsf{RatStreamNameClassifier}(s, t)$ *and streams* s', t' *satisfy* $\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n))$ *and* $\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n))$*, then* $\mathsf{RatStreamNameClassifier}(s', t')$.

Proof. For reflexivity, the two endpoint carrier rows of $\mathsf{RatStreamNameClassifier}(s, s)$ are both the supplied carrier of s. For its pointwise classifier row, fix a certified observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$. The carrier gives $\mathsf{RatHistoryCarrier}(s(n))$ from ν, and RatUp classifier reflexivity turns that pointwise rational witness into $\mathsf{RatHistoryClassifier}(s(n), s(n))$.

For symmetry, unpack the supplied stream classifier into its carrier rows for s and t and its pointwise classifier row. The reversed classifier uses the t carrier as its first carrier row and the s carrier as its second carrier row. For the pointwise row, fix (n, ν) and apply RatUp rational-history classifier symmetry to the projected point classifier at (n, ν).

For transitivity, unpack $\mathsf{RatStreamNameClassifier}(s, t)$ and $\mathsf{RatStreamNameClassifier}(t, u)$. The resulting classifier from s to u uses the first carrier row of the first bundle and the second carrier row of the second bundle. For the pointwise row, fix (n, ν) and compose the two projected point classifiers at (n, ν) by RatUp rational-history classifier transitivity.

For endpoint carriers, fix a certified observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$. Project the two bundled carrier rows of $\mathsf{RatStreamNameClassifier}(s, t)$ and read them at (n, ν); these are exactly the rational-history carrier witnesses required at the two endpoint observations.

For transport, first transport the two bundled carrier rows along the supplied history-sameness witnesses, producing $\mathsf{RatStreamNameCarrier}(s')$ and $\mathsf{RatStreamNameCarrier}(t')$. For the pointwise row, fix (n, ν) and apply BEDC.Derived.RatUp.RatHistoryClassifier_hsame_transport to the projected point classifier at (n, ν) and to the two supplied endpoint history-sameness witnesses

at (n, ν). The transported carrier rows and the transported pointwise classifier row repack as $\mathsf{RatStreamNameClassifier}(s', t')$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName certificate fields`

Theorem 126.14 (Rational stream-name observation one-tail pair readback)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, if $\mathsf{RatStreamNameClassifier}(s, t)$ and n is a unary observation, then displayed one-extension endpoints*

$$\mathsf{hsame}(s(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t(n), \mathsf{E1}(b))$$

determine unary tails with the internal comparison

$$\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b).$$

Proof. Read the stream-name classifier at the certified observation n. Transport the resulting RatUp point classifier to the displayed one-extension endpoints, then apply rational one-tail classifier exactness. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameClassifier observation e1 pair readback`

Theorem 126.15 (Rational stream-name observation shape exclusions)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and an observation history n, suppose*

$$\mathsf{RatStreamNameClassifier}(s, t) \quad \textit{and} \quad \mathsf{UnaryHistory}(n).$$

Then the selected observations satisfy the positive-denominator and unary endpoint package

$$\mathsf{PositiveUnaryDenominator}(s(n)) \wedge \mathsf{PositiveUnaryDenominator}(t(n)) \wedge \mathsf{UnaryHistory}(s(n)) \wedge \mathsf{UnaryHistory}(t(n)).$$

They also satisfy the two empty-history exclusions

$$(\mathsf{hsame}(s(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{e}) \to \mathsf{False}),$$

and for all histories $z_s, z_t : \mathsf{Hist}$ the two zero-headed exclusions

$$(\mathsf{hsame}(s(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{E0}(z_t)) \to \mathsf{False}).$$

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameClassifier observation shape exclusions`

Proof. Read the stream-name classifier at the certified observation (n, ν), where $\nu : \mathsf{UnaryHistory}(n)$. Apply Definition 37.50 to this point classifier. This gives positive unary denominator witnesses for $s(n)$ and $t(n)$.

Apply Definition 37.12 separately to the two positive denominator witnesses. The unary components give $\mathsf{UnaryHistory}(s(n))$ and $\mathsf{UnaryHistory}(t(n))$, and the nonempty components give the two exclusions against e.

For a displayed zero-headed shape $\mathsf{hsame}(s(n), \mathsf{E0}(z_s))$, transport the positive denominator witness for $s(n)$ across that sameness by Definition 37.6. The transported witness contradicts Definition 37.10. The argument for $\mathsf{hsame}(t(n), \mathsf{E0}(z_t))$ is identical, using the positive denominator witness for $t(n)$. □

Theorem 126.16 (Rational stream-name observation full endpoint package)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*, an observation history* n*, and histories* $a, b : \mathsf{Hist}$*, suppose*

$$\mathsf{RatStreamNameClassifier}(s, t) \quad \textit{and} \quad \mathsf{UnaryHistory}(n),$$

and suppose the selected observations are displayed as one-extension endpoints

$$\mathsf{hsame}(s(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t(n), \mathsf{E1}(b)).$$

Then the following selected-observation endpoint package holds:

1. $\mathsf{RatHistoryClassifier}(s(n), t(n))$.
2. $\mathsf{PositiveUnaryDenominator}(s(n)) \wedge \mathsf{PositiveUnaryDenominator}(t(n))$.
3. $\mathsf{UnaryHistory}(s(n)) \wedge \mathsf{UnaryHistory}(t(n))$.
4. $(\mathsf{hsame}(s(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{e}) \to \mathsf{False})$.
5. *For all histories* $z_s, z_t : \mathsf{Hist}$, $(\mathsf{hsame}(s(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{E0}(z_t)) \to \mathsf{False})$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameClassifier observation full endpoint package`

Proof. Unpack Definition 126.2 at the certified observation (n, ν), where $\nu : \mathsf{UnaryHistory}(n)$. The pointwise row gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 126.15 to the same stream classifier and unary observation. This supplies the two positive unary denominator rows, the unary-history rows for the selected observations, the empty-history exclusions, and the zero-headed exclusions for arbitrary z_s, z_t.

Apply Definition 126.14 to the stream classifier, the unary observation, and the two displayed one-extension endpoint rows. This gives $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

The point classifier, the shape rows, and the one-tail readback rows are exactly the listed endpoint package. □

Fit. Traditional target: selected rational-stream observation endpoint package. BEDC source: $\mathsf{Rat}^{\uparrow}$+$\mathsf{StreamName}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$. Stability: pointwise hsame endpoint display at the selected unary observation. Ledger: the selected observation, the two one-extension displays, and arbitrary zero-headed probes. Core theorem: Definition 126.16. Exactness: rational point classifier, endpoint shape exclusions, and one-tail equality readback. Standard bridge: $\mathsf{Real}^{\uparrow}$ contextual endpoint certificates consume this selected-observation package. Status: $\mathsf{ProofTarget}$.

Theorem 126.17 (Rational stream-name transported observation shape exclusions)**.** *For certified observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *and an observation history* n*, suppose*

$$\mathsf{RatStreamNameClassifier}(s, t) \quad \textit{and} \quad \mathsf{UnaryHistory}(n),$$

and suppose the selected endpoints are related pointwise by transport rows

$$\forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(s(k), s'(k)), \quad \forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(t(k), t'(k)).$$

Then the transported selected observations satisfy the positive-denominator and unary endpoint package

$$\mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(t'(n)) \wedge \mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(t'(n)).$$

They also satisfy the two empty-history exclusions

$$(\mathsf{hsame}(s'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{e}) \to \mathsf{False}),$$

and for all histories $z_s, z_t : \mathsf{Hist}$ *the two zero-headed exclusions*

$$(\mathsf{hsame}(s'(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{E0}(z_t)) \to \mathsf{False}).$$

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameClassifier transported observation shape exclusions`

Proof. Read the stream-name classifier at the certified observation (n, ν), where $\nu : \mathsf{UnaryHistory}(n)$. This gives the point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.54 to this classifier and to the two pointwise transport rows evaluated at n. The result is the transported selected classifier $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.

Apply Definition 37.50 to the transported selected classifier. Then apply Definition 37.12 separately to the two positive-denominator witnesses. These rows give the displayed positive denominators, unary endpoint witnesses, and exclusions against e.

It remains to exclude zero-headed displays. If $\mathsf{hsame}(s'(n), \mathsf{E0}(z_s))$ holds, transport $\mathsf{PositiveUnaryDenominator}(s'$ along that sameness by Definition 37.6, then contradict Definition 37.10. The same argument applied to $t'(n)$ and z_t gives the second zero-headed exclusion. □

Fit. Traditional target: transported selected rational-stream observation. BEDC source: $\mathsf{Rat}^{\uparrow}+\mathsf{StreamName}^{\uparrow}$ over BHist. Carrier: $\mathsf{RatStreamNameCarrier}$. Classifier: $\mathsf{RatStreamNameClassifier}$. Stability: pointwise hsame transport. Ledger: the selected observation and two transport rows. Core theorem: Definition 126.17. Exactness: the displayed shape package. Standard bridge: pointwise rational stream equality under observation transport. Status: $\mathsf{ProofTarget}$.

Theorem 126.18 (Rational stream-name transported observation full endpoint package)**.** *For certified observation maps* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$*, an observation history* n*, and histories* $a, b, z_s, z_t : \mathsf{Hist}$*, suppose*

$$\mathsf{RatStreamNameClassifier}(s, t) \quad \textit{and} \quad \mathsf{UnaryHistory}(n),$$

suppose the selected endpoints are related pointwise by transport rows

$$\forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(s(k), s'(k)), \quad \forall k,\ \mathsf{UnaryHistory}(k) \to \mathsf{hsame}(t(k), t'(k)),$$

and suppose the transported selected observations are displayed as one-extension endpoints

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)).$$

Then the transported selected-observation endpoint package holds:

1. $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.
2. $\mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(t'(n))$.
3. $\mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(t'(n))$.
4. $(\mathsf{hsame}(s'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{e}) \to \mathsf{False})$.
5. $(\mathsf{hsame}(s'(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{E0}(z_t)) \to \mathsf{False})$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. Unpack Definition 126.2 at the certified observation (n, ν), where $\nu : \mathsf{UnaryHistory}(n)$. The pointwise row gives the source point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.54 to this source classifier and to the two pointwise transport rows evaluated at n. This gives the transported point classifier $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.

Apply Definition 126.17 to the same stream classifier, unary observation, and transport rows. It supplies the transported positive-denominator rows, unary endpoint rows, empty exclusions, and universal zero-headed exclusions. Instantiate the last two rows at the displayed histories z_s and z_t.

For the one-tail readback, transport $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ along the two displayed one-extension endpoint comparisons. The transported classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b))$.

Apply Definition 37.49 in the forward direction to read back $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$. The transported point classifier, transported shape rows, and one-tail readback rows form exactly the listed endpoint package. □

Theorem 126.19 (Rational stream-name reindexing and map stability)**.** *For certified observation maps $s, t, r : \mathsf{Hist} \to \mathsf{Hist}$, suppose r preserves certified unary observations:*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{UnaryHistory}(r(n)).$$

Then $\mathsf{RatStreamNameCarrier}(s)$ *implies* $\mathsf{RatStreamNameCarrier}(s \circ r)$*, and* $\mathsf{RatStreamNameClassifier}(s, t)$ *implies* $\mathsf{RatStreamNameClassifier}(s \circ r, t \circ r)$*.*

For a history map $F : \mathsf{Hist} \to \mathsf{Hist}$, suppose F preserves RatUp carriers and classifiers:

$$\forall h,\ \mathsf{RatHistoryCarrier}(h) \to \mathsf{RatHistoryCarrier}(F(h))$$

and

$$\forall h, k,\ \mathsf{RatHistoryClassifier}(h, k) \to \mathsf{RatHistoryClassifier}(F(h), F(k)).$$

Then $\mathsf{RatStreamNameCarrier}(s)$ *implies* $\mathsf{RatStreamNameCarrier}(F \circ s)$*, and* $\mathsf{RatStreamNameClassifier}(s, t)$ *implies* $\mathsf{RatStreamNameClassifier}(F \circ s, F \circ t)$*.*

Proof. For reindexing, unfold RatStreamNameCarrier. Fix an observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$. The reindexer hypothesis turns ν into a certified unary observation witness for $r(n)$, and the carrier of s gives the required rational-history carrier at $s(r(n))$.

For the reindexed classifier, unfold RatStreamNameClassifier and fix the same certified observation data (n, ν). The reindexer hypothesis again supplies $\mathsf{UnaryHistory}(r(n))$, so the classifier of s and t can be read at $r(n)$, giving the point classifier between $s(r(n))$ and $t(r(n))$.

For mapped streams, unfold RatStreamNameCarrier and fix n with witness $\nu : \mathsf{UnaryHistory}(n)$. The carrier of s gives $\mathsf{RatHistoryCarrier}(s(n))$, and the carrier-preservation hypothesis for F transports it to $\mathsf{RatHistoryCarrier}(F(s(n)))$.

For mapped classifiers, unfold RatStreamNameClassifier and fix n with witness $\nu : \mathsf{UnaryHistory}(n)$. The classifier of s and t gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the classifier-preservation hypothesis for F transports it to $\mathsf{RatHistoryClassifier}(F(s(n)), F(t(n)))$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName reindexing and map stability`

Theorem 126.20 (Rational stream-name reindexing composition law)**.** *For certified observation maps $s, t, r, q : \mathsf{Hist} \to \mathsf{Hist}$, suppose r and q preserve certified unary observations:*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{UnaryHistory}(r(n))$$

and

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{UnaryHistory}(q(n)).$$

Then the composite observation map $n \mapsto q(r(n))$ preserves certified unary observations. Consequently, $\mathsf{RatStreamNameCarrier}(s)$ *implies* $\mathsf{RatStreamNameCarrier}(s \circ q \circ r)$*, and* $\mathsf{RatStreamNameClassifier}(s, t)$ *implies* $\mathsf{RatStreamNameClassifier}(s \circ q \circ r, t \circ q \circ r)$*.*

Proof. Fix a certified observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$. The preservation witness for r sends ν to a certified unary observation witness for $r(n)$, and the preservation witness for q sends that result to a certified unary observation witness for $q(r(n))$. This proves unary preservation for the composite observation map.

For the carrier consequence, unfold $\mathsf{RatStreamNameCarrier}$ and fix the same certified observation data (n, ν). The composite unary witness just constructed lets the carrier of s be read at $q(r(n))$, giving the required rational-history carrier at $(s \circ q \circ r)(n)$.

For the classifier consequence, unfold $\mathsf{RatStreamNameClassifier}$ and fix (n, ν). The composite unary witness again lets the classifier of s and t be read at $q(r(n))$, yielding the point classifier between $(s \circ q \circ r)(n)$ and $(t \circ q \circ r)(n)$. □

Definition 126.21 (Rational stream-name finite probe window)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, define* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *to mean that every bundled certified unary observation is classified pointwise:*

$$\forall n : \mathsf{Hist}, \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{RatHistoryClassifier}(s(n), t(n)).$$

Theorem 126.22 (Rational stream-name classifier covers finite probe windows)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and every probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$,*

$$\mathsf{RatStreamNameClassifier}(s, t) \Longrightarrow \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}).$$

Proof. Unfold Definition 126.2 and fix a history n with evidence $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{UnaryHistory}(n)$. The stream-name classifier is pointwise on every certified unary observation, so its row at n gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Since the bundled history and unary witness were arbitrary, these pointwise RatUp classifier rows repack exactly as $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$. □

Theorem 126.23 (Rational stream-name finite windows exhaust the classifier)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$,*

$$\mathsf{RatStreamNameClassifier}(s, t) \Longleftrightarrow \forall n : \mathsf{Hist},\ \mathsf{UnaryHistory}(n) \to \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{Bcons}(n$$

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier exhausts classifier`

Proof. For the forward implication, fix a certified unary observation n. Apply Definition 126.22 to the full stream-name classifier and to the singleton bundle $\mathsf{Bcons}(n, \mathsf{Bnil})$. This gives the required finite-window classifier at that observation.

For the reverse implication, fix a certified unary observation n. The singleton-window hypothesis at n, together with the head membership of n in $\mathsf{Bcons}(n, \mathsf{Bnil})$ from Definition 6.1, gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

The carrier fields of this RatUp classifier provide $\mathsf{RatHistoryCarrier}(s(n))$ and $\mathsf{RatHistoryCarrier}(t(n))$. Since n was arbitrary, these fields assemble the two stream-name carriers, and the same pointwise RatUp classifier rows assemble $\mathsf{RatStreamNameClassifier}(s, t)$. □

The ten-field fit is: traditional target finite-window exhaustion; BEDC source $\mathsf{Rat}^{\uparrow}+\mathsf{StreamName}^{\uparrow}+\mathsf{ProbeBundle}$; carrier the existing stream-name carrier rows for s and t; classifier the existing pointwise stream-name classifier row; stability singleton-window coverage over certified unary observations; ledger the singleton observation bundle $\mathsf{Bcons}(n, \mathsf{Bnil})$; core theorem Definition 126.22; exactness singleton membership in the probe bundle; downstream $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}$ finite-prefix consumption; status $\mathsf{ProofTarget}$.

Theorem 126.24 (Rational stream-name finite-window coverage after certified reindexing and point maps)**.** *For certified observation maps $s, t, r : \mathsf{Hist} \to \mathsf{Hist}$, a history point map $F : \mathsf{Hist} \to \mathsf{Hist}$, and a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, suppose r preserves certified unary observations:*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{UnaryHistory}(r(n)).$$

Suppose also that F preserves RatUp carriers and classifiers:

$$\forall h,\ \mathsf{RatHistoryCarrier}(h) \to \mathsf{RatHistoryCarrier}(F(h))$$

and

$$\forall h, k,\ \mathsf{RatHistoryClassifier}(h, k) \to \mathsf{RatHistoryClassifier}(F(h), F(k)).$$

If $\mathsf{RatStreamNameClassifier}(s, t)$*, then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(F \circ s \circ r, F \circ t \circ r, \mathcal{P}).$$

Proof. First apply Definition 126.19 to the observation reindexer r. The reindexing row sends the given stream-name classifier for s and t to a classifier for $s \circ r$ and $t \circ r$.

Apply the mapped-classifier row of the same theorem to the point map F and to the reindexed classifier just obtained. The carrier-preservation hypothesis is part of that stability package, while the classifier-preservation hypothesis transports every RatUp point classifier through F. This yields $\mathsf{RatStreamNameClassifier}(F \circ s \circ r, F \circ t \circ r)$.

Finally apply Definition 126.22 to this transported stream-name classifier and the selected probe bundle $\mathcal{P}$. The result is precisely the displayed finite-window classifier for the reindexed and point-mapped observation streams. □

The ten-field fit is: traditional target finite certified observation windows; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{ProbeBundle}$; carrier observation maps; classifier finite probe-window restriction of the stream classifier; stability pointwise classifier projection; ledger the bundle and selected observation; core theorem Definition 126.22 and Definition 126.24; exactness pointwise RatUp classification on bundled unary observations; standard bridge finite prefixes of rational streams; status $\mathsf{ProofTarget}$.

Theorem 126.25 (Rational stream-name finite-window structural laws)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$ and probe bundles $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$, the finite-window classifier has the following structural laws.*

1. *If every bundled observation of $\mathcal{Q}$ is also bundled in $\mathcal{P}$, then* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *implies* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$.
2. $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q}))$ *implies both* $\mathsf{RatStreamNameFiniteWindowClassi}$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$.
3. *If the finite-window classifier holds on $\mathcal{P}$ and on $\mathcal{Q}$, then it holds on* $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$.

Proof. Unfold Definition 126.21. For the subbundle law, fix an arbitrary history n with $\mathsf{InBundle}(n, \mathcal{Q})$ and a unary witness $\mathsf{UnaryHistory}(n)$. The assumed subbundle map transports the bundle membership from $\mathcal{Q}$ to $\mathcal{P}$, and the classifier on $\mathcal{P}$ gives $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

For the two projections from an appended bundle, fix n first with $\mathsf{InBundle}(n, \mathcal{P})$ and then with $\mathsf{InBundle}(n, \mathcal{Q})$. In each case the append membership lemma for $\mathsf{bundleAppend}$ injects the selected membership into $\mathsf{InBundle}(n, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q}))$. The classifier on the appended bundle then supplies the corresponding point RatUp classifier row.

For recomposition, fix n with membership in $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$ and a unary witness. The kernel equivalence for appended bundles rewrites this membership as a left-or-right membership alternative. In the left case apply the classifier on $\mathcal{P}$; in the right case apply the classifier on $\mathcal{Q}$. Both cases yield the same required conclusion $\mathsf{RatHistoryClassifier}(s(n), t(n))$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier structural laws`

Theorem 126.26 (Rational stream-name finite-window selected zero-head exclusion)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, an observation history n, and histories $z_s, z_t : \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \quad \mathsf{InBundle}(n, \mathcal{P}), \quad \mathsf{UnaryHistory}(n).$$

Then the selected finite-window endpoints cannot be zero-headed:

$$(\mathsf{hsame}(s(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{E0}(z_t)) \to \mathsf{False}).$$

Proof. Unfold Definition 126.21 at the selected observation n, its bundle membership, and its unary witness. This produces the selected RatUp point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.50 to this selected classifier. The result supplies positive-denominator witnesses for both selected endpoint histories $s(n)$ and $t(n)$.

For the left endpoint, assume a displayed zero-headed comparison $\mathsf{hsame}(s(n), \mathsf{E0}(z_s))$. Transport the positive-denominator witness for $s(n)$ along that comparison by Definition 37.6. The transported witness contradicts Definition 37.10.

For the right endpoint, the same argument applied to $\mathsf{hsame}(t(n), \mathsf{E0}(z_t))$ transports the positive-denominator witness for $t(n)$ to a zero-headed denominator witness, again contradicting Definition 37.10. These two contradictions give the displayed conjunction. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier selected zero head exclusion`

The compact ten-field fit is: traditional target finite rational observation endpoint sanity; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{ProbeBundle}$; carrier observation maps; classifier finite-window restriction; ledger selected probe observation; exactness selected RatUp point classification; downstream $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}$ selected endpoint normalization; standard bridge ordinary finite rational stream endpoints; status $\mathsf{ProofTarget}$.

Theorem 126.27 (Rational stream-name finite-window selected full endpoint package)**.** *For certified observation maps $s, t : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, an observation history n, and histories $a, b, z_s, z_t : \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \quad \mathsf{InBundle}(n, \mathcal{P}), \quad \mathsf{UnaryHistory}(n),$$

and suppose the selected observations are displayed as one-extension endpoints

$$\mathsf{hsame}(s(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t(n), \mathsf{E1}(b)).$$

Then the selected finite-window endpoints carry the full endpoint package:

1. $\mathsf{RatHistoryClassifier}(s(n), t(n))$.
2. $\mathsf{PositiveUnaryDenominator}(s(n)) \wedge \mathsf{PositiveUnaryDenominator}(t(n))$.
3. $\mathsf{UnaryHistory}(s(n)) \wedge \mathsf{UnaryHistory}(t(n))$.
4. $(\mathsf{hsame}(s(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{e}) \to \mathsf{False})$.
5. $(\mathsf{hsame}(s(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t(n), \mathsf{E0}(z_t)) \to \mathsf{False})$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. Unfold Definition 126.21 at the selected observation n, its bundle membership, and its unary witness. This gives the selected point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Apply Definition 37.50 to this selected classifier. Then apply Definition 37.12 separately to the two positive-denominator witnesses. These rows give the two positive denominators, the two unary endpoint histories, and the two exclusions against e.

It remains to exclude zero-headed displays. If $\mathsf{hsame}(s(n), \mathsf{E0}(z_s))$ holds, transport the positive-denominator witness for $s(n)$ along that sameness by Definition 37.6, then contradict Definition 37.10. The same argument applied to $t(n)$ and z_t gives the second zero-headed exclusion.

For the one-tail readback, transport the selected point classifier across the two displayed one-extension endpoint comparisons, using Definition 37.54. The transported classifier is $\mathsf{RatHistoryClassifier}(\mathsf{E1}(a), \mathsf{E1}(b$

Apply Definition 37.49 in the forward direction to read back $\mathsf{UnaryHistory}(a)$, $\mathsf{UnaryHistory}(b)$, and $\mathsf{hsame}(a, b)$. The selected classifier, shape rows, zero-headed exclusions, and one-tail readback rows are exactly the listed endpoint package. □

Theorem 126.28 (Rational stream-name finite-window stability fields)**.** *For certified observation maps $s, t, u, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ and a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, the finite-window classifier has the following stability fields.*

1. *If every bundled unary observation of s is carried,*

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{RatHistoryCarrier}(s(n)),$$

   then $\mathsf{RatStreamNameFiniteWindowClassifier}(s, s, \mathcal{P})$.

2. $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *implies* $\mathsf{RatStreamNameFiniteWindowClassifier}(t, s, \mathcal{P})$.
3. $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(t, u, \mathcal{P})$ *imply* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, u, \mathcal{P})$.
4. *If* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *and the endpoints satisfy*

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n))$$

   and

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)),$$

   then $\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P})$.

Proof. Unfold Definition 126.21. For reflexivity, fix a bundled certified unary observation (n, π, ν). The given carrier row supplies $\mathsf{RatHistoryCarrier}(s(n))$, and RatUp classifier reflexivity turns that carrier witness into $\mathsf{RatHistoryClassifier}(s(n), s(n))$.

For symmetry, fix the same bundled unary observation data. The finite-window classifier from s to t supplies the point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and RatUp classifier symmetry reverses it.

For transitivity, fix a bundled unary observation and read the two finite-window classifiers at it. RatUp classifier transitivity composes $\mathsf{RatHistoryClassifier}(s(n), t(n))$ with $\mathsf{RatHistoryClassifier}(t(n), u(n))$, giving the point classifier from $s(n)$ to $u(n)$.

For endpoint transport, fix (n, π, ν) again. The source finite-window classifier gives $\mathsf{RatHistoryClassifier}(s(n), t(n)$ while the two endpoint transport rows give $\mathsf{hsame}(s(n), s'(n))$ and $\mathsf{hsame}(t(n), t'(n))$ at the same bundled unary observation. RatUp point classifier transport along these two hsame rows gives $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$. □

The ten-field fit is: traditional target finite rational observation-window stability; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{ProbeBundle}$; carrier observation maps; classifier finite-window restriction; stability the four displayed rows; ledger the finite probe bundle and endpoint transports; core theorem Definition 126.28; exactness pointwise RatUp classification; standard bridge finite prefixes of rational streams; status $\mathsf{ProofTarget}$.

Theorem 126.29 (StreamName finite-window pointwise reflexivity)**.** *Let $s : \mathsf{Hist} \to \mathsf{Hist}$ be an accepted rational StreamName carrier and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a certified finite bundle of unary observations. Then the restriction of s to $\mathcal{P}$ carries the pointwise rows*

$$\mathsf{InBundle}(n, \mathcal{P}) \to \mathsf{UnaryHistory}(n) \to \mathsf{RatHistoryClassifier}(s(n), s(n))$$

and repacks as the finite-window self-witness

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, s, \mathcal{P}).$$

The witness exposes only the displayed $\mathsf{UnaryHistory}$ *observations, bundle-membership rows, endpoint rational-history carrier rows, pointwise RatUp reflexivity,* hsame *transports already present on the window, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *rows. It contains no quotient stream, host sequence equality, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Unfold the carrier clause of Definition 126.1. For a bundled certified unary observation n, the accepted carrier of s supplies the endpoint rational-history carrier row $\mathsf{RatHistoryCarrier}(s(n))$.

Apply the reflexivity field of Definition 126.13 pointwise at this same observation. Equivalently, the RatUp classifier reflexivity row turns the projected endpoint carrier into $\mathsf{RatHistoryClassifier}(s(n), s(n))$.

Since n was chosen with arbitrary evidence $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{UnaryHistory}(n)$, the pointwise rows are exactly the clauses of Definition 126.21. Repacking them therefore gives $\mathsf{RatStreamNameFiniteWindowClass}$

The surrounding handoff surface is unchanged. The finite bundle contributes only membership, unary-history, displayed transport, continuation, provenance, and naming rows, so the same finite-window self-witness is the local source row consumed by Definition 126.173 and by the $\mathsf{Real}^{\uparrow}$ finite-window boundary without importing quotient-stream equality, host sequence equality, selected limits, or completeness data. □

Theorem 126.30 (StreamName finite-window pointwise symmetry)**.** *Let $s, t : \mathsf{Hist} \to \mathsf{Hist}$ be accepted rational StreamName carriers and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a certified finite bundle of unary observations. If* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *holds, then the same displayed bundle rows, unary observations, endpoint carrier rows, pointwise RatUp symmetry,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows repack as*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(t, s, \mathcal{P}).$$

The repacked classifier contains no quotient stream, host sequence equality, selected limit, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Unfold Definition 126.21. A bundled certified unary observation n supplies the same membership evidence $\mathsf{InBundle}(n, \mathcal{P})$, the same unary-history witness, and the two endpoint rational-history carrier rows already exposed by the source finite window.

At this observation, apply the symmetry field of Definition 126.13 to the displayed pointwise RatUp classifier row $\mathsf{RatHistoryClassifier}(s(n), t(n))$. This yields the reversed point classifier $\mathsf{RatHistoryClassifier}(t(n), s(n))$ without changing the finite bundle or the endpoint carriers.

Since the observation and bundle-membership evidence were arbitrary, the reversed pointwise rows satisfy exactly the clauses of Definition 126.21 for the ordered pair (t, s). Repack them with the unchanged hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows.

All data remain finite-window data. No quotient stream, host sequence equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness witness is read or introduced. □

Theorem 126.31 (StreamName finite-window pointwise transitivity)**.** *Let $s, t, u : \mathsf{Hist} \to \mathsf{Hist}$ be accepted rational StreamName carriers and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a certified finite bundle of unary observations. If $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ and $\mathsf{RatStreamNameFiniteWindowClassifier}(t, u, \mathcal{P})$ hold over this same finite bundle, then the displayed finite-window rows compose pointwise to*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, u, \mathcal{P}).$$

The composition uses only pointwise RatUp transitivity and the displayed finite-window rows; it adds no quotient stream, host sequence equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness row.

Proof. Unfold Definition 126.21 for the two source windows. A bundled certified unary observation n in $\mathcal{P}$ reads a point classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$ from the first window and a point classifier $\mathsf{RatHistoryClassifier}(t(n), u(n))$ from the second.

Apply the transitivity field of Definition 126.13 at this same observation. The middle endpoint carrier row is the displayed $t(n)$ row already shared by the two windows, so the RatUp point classifiers compose to $\mathsf{RatHistoryClassifier}(s(n), u(n))$.

Because both hypotheses use the same finite bundle $\mathcal{P}$, no membership transport or window refinement is needed. The composed pointwise rows therefore satisfy exactly the clauses of Definition 126.21 for (s, u).

Repack the unchanged bundle-membership rows, unary observations, endpoint carrier rows, hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows. The resulting witness remains a finite-window classifier and does not import quotient-stream equality, host sequence equality, selected limits, or completeness data. □

Theorem 126.32 (Rational stream-name finite-window transported selected full endpoint package)**.** *For certified observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, a selected bundled unary observation n, and histories $a, b, z_s, z_t : \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \quad \mathsf{InBundle}(n, \mathcal{P}), \quad \mathsf{UnaryHistory}(n),$$

and suppose the bundled pointwise transport rows

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{P}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(s(m), s'(m))$$

and

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{P}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(t(m), t'(m))$$

hold. If the transported selected observations are displayed as one-extension endpoints

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)),$$

then the transported selected finite-window endpoints carry the full endpoint package:

1. $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$.
2. $\mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(t'(n))$.
3. $\mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(t'(n))$.
4. $(\mathsf{hsame}(s'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{e}) \to \mathsf{False})$.
5. $(\mathsf{hsame}(s'(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(t'(n), \mathsf{E0}(z_t)) \to \mathsf{False})$.
6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First transport the finite-window classifier. For every bundled unary observation m, use the given transport row at m for the left endpoint and the corresponding row at m for the right endpoint. The endpoint-transport field of Definition 126.28 then gives

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P}).$$

Now apply Definition 126.27 to this transported classifier, the selected bundle membership of n, the unary witness for n, and the displayed one-extension endpoints of $s'(n)$ and $t'(n)$. Its six conclusions are exactly the transported selected endpoint package listed above. □

The compact ten-field fit is: traditional target transported finite rational observation endpoint package; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{ProbeBundle}$; carrier certified observation maps; classifier finite-window restriction; stability bundled hsame transport; ledger selected bundled unary probe observation and displayed transported endpoints; core theorems Definition 126.28 and Definition 126.27; exactness transported selected RatUp point classification with positive denominators, shape exclusions, and one-tail agreement; standard bridge ordinary finite rational stream endpoint transport; status $\mathsf{ProofTarget}$.

Theorem 126.33 (Rational stream-name finite-window transported composite endpoint package)**.** *For certified observation maps $s, t, u, s', u' : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, a selected bundled unary observation n, and histories $a, b, z_s, z_u : \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \quad \mathsf{RatStreamNameFiniteWindowClassifier}(t, u, \mathcal{P}), \quad \mathsf{InBundle}(n, \mathcal{P}),$$

and suppose the bundled pointwise transport rows

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{P}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(s(m), s'(m))$$

and

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{P}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(u(m), u'(m))$$

hold. If the transported composite endpoints are displayed as one-extension endpoints

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(u'(n), \mathsf{E1}(b)),$$

then the transported composite finite-window endpoints carry the full endpoint package:

1. $\mathsf{RatHistoryClassifier}(s'(n), u'(n))$.
2. $\mathsf{PositiveUnaryDenominator}(s'(n)) \wedge \mathsf{PositiveUnaryDenominator}(u'(n))$.
3. $\mathsf{UnaryHistory}(s'(n)) \wedge \mathsf{UnaryHistory}(u'(n))$.
4. $(\mathsf{hsame}(s'(n), \mathsf{e}) \to \mathsf{False}) \wedge (\mathsf{hsame}(u'(n), \mathsf{e}) \to \mathsf{False})$.

5. $(\mathsf{hsame}(s'(n), \mathsf{E0}(z_s)) \to \mathsf{False}) \wedge (\mathsf{hsame}(u'(n), \mathsf{E0}(z_u)) \to \mathsf{False})$.

6. $\mathsf{UnaryHistory}(a) \wedge \mathsf{UnaryHistory}(b) \wedge \mathsf{hsame}(a, b)$.

Proof. First compose the finite-window classifiers. The transitivity field of Definition 126.25 applies to the rows from s to t and from t to u, giving

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, u, \mathcal{P}).$$

Next transport the composed classifier. Use the bundled pointwise transport row from s to s' for the left endpoint and the row from u to u' for the right endpoint. The endpoint-transport field of Definition 126.28 gives

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', u', \mathcal{P}).$$

Finally apply Definition 126.27 to the transported classifier, the bundle membership of n, the unary witness for n, and the displayed one-extension endpoints of $s'(n)$ and $u'(n)$. Its six endpoint rows are exactly the package listed above. □

The compact ten-field fit is: traditional target composite transported finite rational observation endpoint package; BEDC source $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow} + \mathsf{ProbeBundle}$; carrier certified observation maps; classifier finite-window RegSeqRat span; stability bundled hsame transport across the selected endpoints; ledger selected bundled unary probe observation, composite middle map, and displayed transported endpoints; core theorems Definition 126.25, Definition 126.28, and Definition 126.27; exactness transported RatUp point classification with positive denominators, shape exclusions, and one-tail agreement; standard bridge ordinary finite rational RegSeqRat stream endpoint transport; status $\mathsf{ProofTarget}$.

Theorem 126.34 (StreamNameUp transported window public readback)**.** *For certified observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, a selected bundled unary observation n, and histories $a, b, z_s, z_t : \mathsf{Hist}$, suppose the hypotheses of Definition 126.32 hold. Then the selected transported finite-window endpoint package is exactly the public finite-window $\mathsf{StreamName}^{\uparrow}$ readback surface for s' and t' at the bundled unary observation n. Its public rows are*

1. $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$;

2. *positive unary denominators for $s'(n)$ and $t'(n)$;*

3. *unary-history rows for $s'(n)$ and $t'(n)$;*

4. *empty-endpoint exclusions for $s'(n)$ and $t'(n)$;*

5. *zero-headed endpoint exclusions for $s'(n)$ and $t'(n)$;*

6. *unary one-tail agreement for the displayed tails a and b.*

No additional host stream, quotient endpoint, or RegSeqRat-internal witness is part of this readback surface.

Proof. Apply Definition 126.32 to the displayed transport hypotheses, bundle membership, unary witness, and one-extension endpoints.

The cited theorem already returns the six endpoint rows listed above: RatHistoryClassifier, positive denominators, unary endpoint shape, empty exclusion, zero-headed exclusion, and one-tail agreement. Repack these rows as the public finite-window $\mathsf{StreamName}^{\uparrow}$ readback interface, leaving the transport proofs and selected-bundle witness in the ledger. □

Theorem 126.35 (StreamNameUp transported subwindow public readback)**.** *For certified observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$, probe bundles $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$, a selected bundled unary observation n, and histories $a, b, z_s, z_t : \mathsf{Hist}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$$

and suppose every certified unary observation displayed by $\mathcal{Q}$ is also displayed by $\mathcal{P}$. Suppose also

$$\mathsf{InBundle}(n, \mathcal{Q}), \quad \mathsf{UnaryHistory}(n),$$

together with the bundled pointwise transport rows

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{Q}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(s(m), s'(m))$$

and

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{Q}) \to \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(t(m), t'(m)).$$

If the selected transported observations are displayed as one-extension endpoints

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)),$$

then the transported $\mathcal{Q}$ subwindow exposes the same public finite-window $\mathsf{StreamName}^{\uparrow}$ readback surface at n:

1. $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$*;*
2. *positive unary denominators for $s'(n)$ and $t'(n)$;*
3. *unary-history rows for $s'(n)$ and $t'(n)$;*
4. *empty-endpoint exclusions for $s'(n)$ and $t'(n)$;*
5. *zero-headed endpoint exclusions for $s'(n)$ and $t'(n)$;*
6. *unary one-tail agreement for the displayed tails a and b.*

No observation outside the subwindow $\mathcal{Q}$ is exported by this readback surface.

Proof. First restrict the ambient finite-window classifier. Apply Definition 126.9 to the $\mathcal{P}$ classifier and the displayed inclusion of certified unary observations from $\mathcal{Q}$ into $\mathcal{P}$. This gives

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}).$$

Next apply Definition 126.32 to the restricted classifier, the selected $\mathcal{Q}$ bundle membership of n, the unary witness for n, the pointwise transport rows over $\mathcal{Q}$, and the displayed one-extension endpoints of $s'(n)$ and $t'(n)$.

The cited package supplies exactly the six public rows: RatHistoryClassifier, positive denominators, unary endpoint shape, empty exclusion, zero-headed exclusion, and one-tail agreement. Repack those rows as the public $\mathsf{StreamName}^{\uparrow}$ subwindow readback, while the ambient window and inclusion row remain certificate ledger data. □

Theorem 126.36 (StreamNameUp transported composite window public readback)**.** *For certified observation maps $s, t, u, s', u' : \mathsf{Hist} \to \mathsf{Hist}$, a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, a selected bundled unary observation n, and histories $a, b, z_s, z_u : \mathsf{Hist}$, suppose the hypotheses of Definition 126.33 hold. Then the composed classifier path from s through t to u, with transported endpoints s' and u', exposes exactly the same public finite-window $\mathsf{StreamName}^{\uparrow}$ readback surface at n. Its public rows are*

1. $\mathsf{RatHistoryClassifier}(s'(n), u'(n))$;
2. *positive unary denominators for* $s'(n)$ *and* $u'(n)$;
3. *unary-history rows for* $s'(n)$ *and* $u'(n)$;
4. *empty-endpoint exclusions for* $s'(n)$ *and* $u'(n)$;
5. *zero-headed endpoint exclusions for* $s'(n)$ *and* $u'(n)$;
6. *unary one-tail agreement for the displayed tails* a *and* b.

The intermediate classifier map t *remains ledger data and is not exported as an extra public endpoint.*

Proof. Apply Definition 126.33. Its proof composes the structural finite-window classifier laws, transports the composed endpoint classifier, and then invokes the selected endpoint package.

The resulting six rows are RatHistoryClassifier, positive denominators, unary endpoint shape, empty exclusion, zero-headed exclusion, and one-tail agreement for the transported endpoints $s'(n)$ and $u'(n)$. Exposing exactly these rows gives the public composite $\mathsf{StreamName}^{\uparrow}$ readback interface, while the middle map and transport chain stay in the certificate ledger. □

Theorem 126.37 (StreamName two-route selected endpoint agreement)**.** *For certified observation maps* $s, t, u, s', t', u' : \mathsf{Hist} \to \mathsf{Hist}$*, a common probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*, a selected bundled unary observation* n*, and displayed one-extension endpoint tails* $a, b, c, z_s, z_t, z_u : \mathsf{Hist}$*, suppose the hypotheses of Definition 126.32 hold for the route from* s *to* t *with transported endpoints* s' *and* t'*:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \quad \mathsf{InBundle}(n, \mathcal{P}), \quad \mathsf{UnaryHistory}(n),$$

the bundled pointwise transport rows for s *to* s' *and* t *to* t'*, and the displayed endpoint rows*

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)).$$

Suppose also that the same selected unary observation and common bundle satisfy the hypotheses of Definition 126.32 for the route from s *to* u *with transported endpoints* s' *and* u'*:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, u, \mathcal{P}),$$

the bundled pointwise transport rows for s *to* s' *and* u *to* u'*, and the displayed endpoint rows*

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(u'(n), \mathsf{E1}(c)).$$

Then the two exported right tails of the selected transported readbacks agree:

$$\mathsf{hsame}(b, c).$$

Proof. Apply Definition 126.32 to the s to t route. Its selected endpoint package includes the one-tail agreement row $\mathsf{hsame}(a, b)$ for the displayed endpoints of $s'(n)$ and $t'(n)$.

Apply the same theorem to the s to u route over the same bundle and selected unary observation. The selected endpoint package for this route includes the one-tail agreement row $\mathsf{hsame}(a, c)$ for $s'(n)$ and $u'(n)$.

Use symmetry of hsame on $\mathsf{hsame}(a, b)$ to obtain $\mathsf{hsame}(b, a)$, and then compose it by transitivity with $\mathsf{hsame}(a, c)$. The resulting relation is $\mathsf{hsame}(b, c)$, which is exactly the agreement of the two exported right tails. □

Theorem 126.38 (StreamNameUp transported window common-refinement public readback)**.** *For certified rational stream observations* $s_1, t_1, s'_1, t'_1, s_2, t_2, s'_2, u'_2 : \mathsf{Hist} \to \mathsf{Hist}$*, let* $\mathcal{P}$ *be a common probe bundle and let* $\mathcal{Q}_1, \mathcal{Q}_2 : \mathsf{ProbeBundle}$ *be two transported finite windows displayed as subwindows of* $\mathcal{P}$*. Suppose* $\mathsf{RatStreamNameFiniteWindowClassifier}(s'_1, t'_1, \mathcal{P})$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(s'_2,$ *hold, and suppose each certified unary observation of* $\mathcal{Q}_i$ *is carried by* $\mathcal{P}$*. If selected observations* n_1 *and* n_2 *belong to* $\mathcal{Q}_1$ *and* $\mathcal{Q}_2$ *with unary endpoint rows and displayed one-tail witnesses, then the two selected public readback surfaces are projections of the same common-refinement finite-window classifier over* $\mathcal{P}$*. Each projection exposes exactly the public rows*

1. *the appropriate* RatHistoryClassifier *row;*
2. *positive denominators for the two selected endpoint histories;*
3. *unary endpoint rows for the selected histories;*
4. *empty-endpoint exclusions;*
5. *zero-headed endpoint exclusions;*
6. *one-tail agreement for the displayed endpoint tails.*

The common bundle, the subwindow inclusion rows, and the transport rows remain ledger data, not additional public endpoints.

Proof. First restrict the common classifiers to the two displayed subwindows. Apply Definition 126.9 to the classifier over $\mathcal{P}$ and the inclusion row $\mathcal{Q}_1 \subseteq \mathcal{P}$ to obtain the finite-window classifier on $\mathcal{Q}_1$. Apply the same theorem to $\mathcal{Q}_2 \subseteq \mathcal{P}$ for the second window.

Next read the selected observations. For the single transported window, apply Definition 126.34 at n_1. For the composite transported window, apply Definition 126.36 at n_2. These two readbacks expose the RatHistoryClassifier rows, positive denominators, unary endpoint rows, empty and zero-headed exclusions, and one-tail agreement.

The common-refinement claim is exactly the fact that both selected readbacks were obtained after projecting finite-window classifiers from the same bundle $\mathcal{P}$. The inclusions $\mathcal{Q}_i \subseteq \mathcal{P}$ and all transport rows are used to choose and compare projections; they are retained in the ledger and do not become public endpoints. □

Theorem 126.39 (StreamNameUp common-refinement denominator and shape agreement)**.** *For certified rational stream observations* $s, t, u, s', t', u' : \mathsf{Hist} \to \mathsf{Hist}$*, a common probe bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*, selected bundled unary observations* n_1, n_2*, and displayed one-extension endpoint tails* $a, b, c : \mathsf{Hist}$*, suppose two finite-window public readback surfaces are obtained from the same common refinement over* $\mathcal{P}$*. Suppose the first surface exposes the transported endpoints* $s'(n_1)$ *and* $t'(n_1)$ *through*

$$\mathsf{hsame}(s'(n_1), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n_1), \mathsf{E1}(b)),$$

and the second surface exposes the same transported observation endpoint and right endpoint through

$$\mathsf{hsame}(s'(n_2), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(u'(n_2), \mathsf{E1}(c)).$$

If the two selected observations name the same transported observation endpoint under hsame*, then the selected public rows agree: their* RatHistoryClassifier *rows are the two projections of the same common-refinement classifier, their positive-denominator rows name* hsame*-equal endpoint histories, their unary endpoint-shape rows name* hsame*-equal endpoint histories, their empty and zero-headed exclusions are the corresponding exclusion rows for those same histories, and their*

displayed one-tail agreement rows have hsame*-equal right tails. In particular the finite-window denominator data and endpoint-shape data exported to* $\mathsf{Real}^{\uparrow}$ *or* RegSeqRat *consumers do not depend on which selected public readback surface of the common refinement is used.*

Proof. First apply Definition 126.38 to the two selected surfaces. This identifies both readbacks as public projections of the same finite-window classifier over $\mathcal{P}$, with the subwindow inclusions and transport rows kept in the ledger.

Next apply Definition 126.34 to each selected surface. For each projection it supplies exactly the six exported rows: RatHistoryClassifier, positive denominators, unary endpoint shape, empty exclusion, zero-headed exclusion, and one-tail agreement for the displayed tails.

Finally use Definition 126.37 on the two routes that share the transported observation endpoint. The theorem gives $\mathsf{hsame}(b, c)$ for the two displayed right tails. Repack the two readback packages along this tail agreement and along the shared endpoint hsame-row; the classifier, denominator, shape, exclusion, and one-tail rows are therefore the same public rows up to hsame. □

The compact ten-field fit is: traditional target common refinement of transported rational observation windows; BEDC source $\mathsf{Rat}^{\uparrow}$+$\mathsf{StreamName}^{\uparrow}$+ProbeBundle; carrier certified observation maps; classifier common-refinement finite-window restriction; stability subwindow restriction plus endpoint transport; ledger bundle inclusions and selected observations; exactness public endpoint readback; standard bridge finite RegSeqRat and RealUp observation windows; status ProofTarget.

Theorem 126.40 (StreamNameUp successor-reindexed selected endpoint package)**.** *Let* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *be certified rational stream-name observations. Suppose* $\mathsf{RatStreamNameClassifier}(s, t)$*,* $\mathsf{UnaryHistory}(n)$*,* $\mathsf{hsame}(s(\mathsf{E1}(n)), \mathsf{E1}(a))$*, and* $\mathsf{hsame}(t(\mathsf{E1}(n)), \mathsf{E1}(b))$*. For the successor-reindexed observations* $s^+(m) = s(\mathsf{E1}(m))$ *and* $t^+(m) = t(\mathsf{E1}(m))$*, the selected endpoint at* n *exposes the finite-window endpoint package:*

1. $\mathsf{RatHistoryClassifier}(s^+(n), t^+(n))$*;*
2. *positive unary denominators for* $s^+(n)$ *and* $t^+(n)$*;*
3. *unary endpoint rows for* $s^+(n)$ *and* $t^+(n)$*;*
4. *empty-endpoint exclusions for* $s^+(n)$ *and* $t^+(n)$*;*
5. *zero-headed endpoint exclusions for* $s^+(n)$ *and* $t^+(n)$*;*
6. *unary one-tail agreement for the displayed endpoint tails* a *and* b*.*

The package is selected at the visible successor observation; it exports no completion object, metric witness, or regular-sequence datum.

Proof. Apply Definition 126.7 to the classifier row $\mathsf{RatStreamNameClassifier}(s, t)$. This gives the successor-reindexed classifier row for s^+ and t^+.

Next apply Definition 126.27 to that reindexed classifier, the unary row for n, and the displayed hsame endpoint facts for $s(\mathsf{E1}(n))$ and $t(\mathsf{E1}(n))$.

The selected finite-window package supplies exactly the six rows listed in the statement. Repack them under the successor observation names $s^+(n)$ and $t^+(n)$, keeping the reindexing proof in the ledger. □

Theorem 126.41 (StreamNameUp successor-reindexed window public readback)**.** *Under the hypotheses of Definition 126.40, the successor-reindexed selected endpoint package is precisely the public finite-window* $\mathsf{StreamName}^{\uparrow}$ *readback for observing* s *and* t *one unary step later. The public*

surface consists of the reindexed RatHistoryClassifier row, positive denominators, unary endpoint rows, empty and zero-headed exclusions, and one-tail agreement for a and b.

This readback deliberately excludes $\mathsf{Real}^{\uparrow}$ *completion data,* $\mathsf{RegSeqRat}$ *regularity data, metric witnesses, and any standard bridge claim. Those objects may consume the finite window, but they are not exported by the successor readback itself.*

Proof. Use Definition 126.40 to obtain the six visible endpoint rows for the successor observation.

The public readback interface records only rows visible at the finite unary window. Therefore the classifier, positivity, endpoint-shape, exclusion, and tail-agreement rows are exported, while completion, regularity, metric, and bridge data remain outside this certificate. □

Theorem 126.42 (StreamNameUp transported successor selected endpoint readback)**.** *Let* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *be certified observation maps. Suppose* $\mathsf{RatStreamNameClassifier}(s, t)$ *and suppose the pointwise transported unary rows*

$$\forall m,\ \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(s(m), s'(m))$$

and

$$\forall m,\ \mathsf{UnaryHistory}(m) \to \mathsf{hsame}(t(m), t'(m))$$

hold. For a unary observation n with $\mathsf{UnaryHistory}(n)$ *and histories* $a, b, z_s, z_t : \mathsf{Hist}$*, suppose the transported selected endpoints are displayed as*

$$\mathsf{hsame}(s'(\mathsf{E1}(n)), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(\mathsf{E1}(n)), \mathsf{E1}(b)).$$

For the transported successor schedules $(s')^+(m) = s'(\mathsf{E1}(m))$ *and* $(t')^+(m) = t'(\mathsf{E1}(m))$*, the selected successor observation at n exposes the following finite-window endpoint rows:*

1. $\mathsf{RatHistoryClassifier}((s')^+(n), (t')^+(n))$*;*
2. *positive unary denominators for* $(s')^+(n)$ *and* $(t')^+(n)$*;*
3. *unary endpoint rows for* $(s')^+(n)$ *and* $(t')^+(n)$*;*
4. *empty-endpoint exclusions for* $(s')^+(n)$ *and* $(t')^+(n)$*;*
5. *zero-headed endpoint exclusions for* $(s')^+(n)$ *and* $(t')^+(n)$*;*
6. *unary one-tail agreement for the displayed endpoint tails a and b.*

The readback is a finite transported successor window. It exports no $\mathsf{Real}^{\uparrow}$ *completion object,* $\mathsf{RegSeqRat}$ *regularity datum, metric witness, or standard bridge claim.*

Proof. First apply Definition 126.8 to the classifier row $\mathsf{RatStreamNameClassifier}(s, t)$ and to the two pointwise hsame transport rows. This gives the successor classifier and, for each probe bundle, the finite-window classifier for the transported successor schedules $(s')^+$ and $(t')^+$.

Next view n as the selected successor observation. Apply Definition 126.27 to that transported finite-window classifier, the unary row for n, and the displayed endpoint rows $\mathsf{hsame}((s')^+(n), \mathsf{E1}(a))$ and $\mathsf{hsame}((t')^+(n), \mathsf{E1}(b))$.

The selected endpoint package supplies the RatHistoryClassifier row, the positive denominators, the unary endpoint rows, the empty and zero-headed exclusions, and the one-tail agreement. The transport rows and the successor reindexing proof remain ledger data; they are not RealUp completion data. □

Theorem 126.43 (StreamNameUp transported successor common-refinement readback)**.** *Let* $s_1, t_1, s_1', t_1', s_2, t_2, s_2', u_2' : \mathsf{Hist} \to \mathsf{Hist}$ *be certified rational stream-name observation maps. Let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a common transported finite-window classifier, and let* $\mathcal{Q}_1, \mathcal{Q}_2 : \mathsf{ProbeBundle}$ *be successor windows projected from* $\mathcal{P}$*. Suppose the two transported successor observations are selected at unary histories* n_1, n_2 *and expose displayed endpoint tails* $a_1, b_1, a_2, c_2 : \mathsf{Hist}$ *by*

$$\mathsf{hsame}((s_1')^+(n_1), \mathsf{E1}(a_1)), \qquad \mathsf{hsame}((t_1')^+(n_1), \mathsf{E1}(b_1)),$$

and

$$\mathsf{hsame}((s_2')^+(n_2), \mathsf{E1}(a_2)), \qquad \mathsf{hsame}((u_2')^+(n_2), \mathsf{E1}(c_2)).$$

If both successor windows are projected from the same common finite-window classifier over $\mathcal{P}$*, the transported successor readbacks have the same public finite-window surface up to* hsame*: their public denominator rows, unary endpoint-shape rows, empty and zero-headed exclusion rows, and one-tail agreement rows are the two projections of the same common-refinement classifier. The common bundle and the pointwise transport rows remain ledger data over* $\mathsf{Rat}^{\uparrow}$+$\mathsf{StreamName}^{\uparrow}$+ProbeBundle*,* UnaryHistory*,* hsame*, and transported finite-window classifiers.*

Proof. First read the common refinement before taking the successor-selected public surface. Apply Definition 126.38 to the two successor subwindows $\mathcal{Q}_1, \mathcal{Q}_2$ inside the common bundle $\mathcal{P}$. This identifies both selected windows as projections of one transported finite-window classifier, with bundle inclusions and transport rows retained in the ledger.

Next build the successor endpoint packages. Apply Definition 126.40 to the underlying successor schedules for each selected observation. The result is the finite-window endpoint package at the visible unary successor, carrying the classifier, denominator, endpoint-shape, exclusion, and one-tail rows.

Finally transport the selected packages. Apply Definition 126.42 to replace the source successor schedules by the pointwise transported schedules. Because both transported successor packages were already projected from the same common finite-window classifier, the public denominator, endpoint-shape, exclusion, and one-tail rows are identified by the same hsame rows that identify the selected transported endpoints. No completion object, metric witness, regularity datum, or standard bridge row is exported. □

Theorem 126.44 (StreamName transported window RegSeqRat RealUp feed)**.** *Let* $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ *be certified rational StreamName observation maps, and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a selected transported finite-window request. Suppose its public endpoint readback is supplied by one of Definition 126.34, Definition 126.36, or Definition 126.43. Then the exported transported rows form exactly the downstream finite-window feed consumed by Definition 38.251, Definition 38.314, and Definition 38.320.*

The feed consists of the selected RatHistoryClassifier *row, the positive unary denominator rows, the unary endpoint-shape rows, the empty and zero-headed endpoint exclusions, the one-tail agreement row, the selected* ProbeBundle *and* InBundle *membership row, and the finite* Cont *and* hsame *transport ledger. These are precisely the RegSeqRat seal inputs and the RealUp finite-window boundary inputs. No host* Nat *index, quotient representative, completion object, metric witness, or* $\mathsf{Real}^{\uparrow}$ *completeness datum is exported.*

Proof. Project the public endpoint rows from the chosen StreamName transported readback theorem. In the direct public case these rows are read from Definition 126.34; in the composite case they are read from Definition 126.36; in the successor common-refinement case they are read from Definition 126.43.

In each case the projected packet has the same finite-window shape: a selected ProbeBundle, its InBundle membership row, a RatHistoryClassifier endpoint row, positive unary denominator rows, unary endpoint-shape rows, empty and zero-headed exclusions, and one-tail agreement. The Cont acceptance row and the pointwise hsame comparisons remain attached as the transport ledger.

Pass that packet through Definition 38.251. The seal reads the finite-window classifier and selected bundle row together with the displayed endpoint and denominator rows, so it introduces no host index or quotient representative.

Finally repack the same sealed packet through Definition 38.314 and Definition 38.320. These RealUp boundary theorems consume only the finite StreamName packet and its transport ledger, so no completion object, metric witness, or completeness datum is added. □

Theorem 126.45 (StreamName RealUp terminal seal boundary)**.** *Let $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ be certified rational StreamName maps, and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a selected transported finite-window request. Suppose a terminal finite-window $\mathsf{Real}^{\uparrow}$ seal consumer receives its endpoint readback from one of Definition 126.34, Definition 126.36, or Definition 126.43. Then the terminal consumer factors through exactly the finite packet listed in Definition 126.44: the selected* RatHistoryClassifier *row, positive unary denominator rows, unary endpoint-shape rows, endpoint exclusions, one-tail agreement, the selected* ProbeBundle *and* InBundle *membership rows, and the finite* Cont *and* hsame *ledger rows.*

Consequently the terminal seal exposes no host Nat *index, quotient representative, completion object, metric witness, order row, arithmetic row, standard bridge, or $\mathsf{Real}^{\uparrow}$ completeness datum.*

Proof. Begin with the chosen endpoint readback. By Definition 126.44, the three admissible readback sources have the same finite downstream feed: a selected ProbeBundle with its InBundle membership row, the RatHistoryClassifier endpoint classifier, positive unary denominator rows, unary endpoint-shape rows, endpoint exclusions, one-tail agreement, and the attached Cont and hsame transport ledger.

Apply Definition 38.251 to this feed. The RegSeqRat seal exactness theorem can read only the displayed finite classifier, bundle, endpoint, denominator, exclusion, tail-agreement, and transport rows. It therefore has no row from which to recover a host index or a quotient representative.

Next pass the same packet through Definition 38.314. The exhaustion obligation consumes the selected finite window and its transport ledger, so the consumer remains inside the finite StreamName-to-RegSeqRat packet.

Finally apply Definition 38.320. The boundary inventory contains exactly the packet rows just named and no further terminal consumer row. Since the inventory has no slot for completion objects, metric witnesses, order or arithmetic rows, standard bridges, or $\mathsf{Real}^{\uparrow}$ completeness data, none of those objects is exposed by the terminal seal. □

Theorem 126.46 (StreamName RealUp terminal seal scope readback)**.** *Let $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ be certified rational StreamName maps, and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a selected transported finite-window request. Suppose a terminal finite-window $\mathsf{Real}^{\uparrow}$ seal consumer is obtained from the StreamName terminal seal boundary of Definition 126.45. Then every scope row available to that consumer factors through exactly the carried StreamName finite-window classifier, the selected endpoint package, the transported denominator package, the RegSeqRat dyadic ledger rows, the* hsame *transport rows, the* Cont *provenance row, and the source* Pkg *rows.*

No readback from this terminal scope exposes a host Nat *index, quotient representative, completion object, metric row, or standard bridge row. Thus the reusable scope handed to later $\mathsf{Real}^{\uparrow}$ and RegSeqRat boundary arguments is precisely the displayed finite packet and nothing outside it.*

Proof. Project the terminal packet supplied by Definition 126.45. Its available rows are the selected finite-window classifier, endpoint and denominator data, endpoint exclusions, one-tail agreement, selected ProbeBundle and InBundle membership, and the attached Cont and hsame transport ledger.

Apply Definition 126.44 to the same packet. This identifies the RegSeqRat and $\mathsf{Real}^{\uparrow}$ handoff rows as the dyadic ledger rows carried by the transported denominator package together with the endpoint classifier and the source Pkg rows already present in the StreamName readback.

Finally use Definition 38.320. Its boundary inventory has slots only for the displayed finite StreamName packet, the RegSeqRat dyadic ledger, the hsame transports, the Cont provenance, and the source package rows. Since the inventory has no slot for host Nat indices, quotient representatives, completion objects, metric rows, or standard bridge rows, the terminal scope-readback exposes none of them. □

Definition 126.47 (Constant rational stream name)**.** *For a history* $d : \mathsf{Hist}$*, write* $\mathsf{RatConstStream}_d(n) = d$ *for every unary observation history* n*.*

Lean4 (definition) `BEDC.Derived.StreamNameUp.RatStreamName constant`
Lean4 (definition) `BEDC.Derived.StreamNameUp.RatConstStream`

Theorem 126.48 (Constant rational stream-name witness)**.** *For histories* $d, e : \mathsf{Hist}$*, every certified unary observation* $n : \mathsf{Hist}$ *satisfies* $\mathsf{hsame}(\mathsf{RatConstStream}_d(n), d)$*. If* $\mathsf{RatHistoryCarrier}(d)$*, then* $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d)$ *and* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_d)$*. If* $\mathsf{RatHistoryClassifier}(d, e)$*, then* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e)$*.*

Proof. Unfold RatConstStream. For any certified unary observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$, the selected observation is definitionally the hidden seed d. History reflexivity gives $\mathsf{hsame}(\mathsf{RatConstStream}_d(n), d)$ at that point.

Unfold RatConstStream and RatStreamNameCarrier. For any certified observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$, the stream value is d, so the supplied $\mathsf{RatHistoryCarrier}(d)$ is the required pointwise carrier witness.

For the self-classifier, unfold RatStreamNameClassifier and fix n with witness $\nu : \mathsf{UnaryHistory}(n)$. The two endpoint observations are both d, and RatUp carrier-reflexivity turns the certified rational observation into $\mathsf{RatHistoryClassifier}(d, d)$.

For a classifier between two constant streams, unfold RatStreamNameClassifier and fix n with witness $\nu : \mathsf{UnaryHistory}(n)$. The endpoint observations are d and e, so the supplied $\mathsf{RatHistoryClassifier}(d, e)$ is already the required pointwise classifier witness. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName constant witness`

Theorem 126.49 (Constant rational stream certified-observation determinacy)**.** *For a history* $d : \mathsf{Hist}$ *and certified unary observations* $m, n : \mathsf{Hist}$*, the constant rational stream reads the same history at both observation sites:*

$$\mathsf{hsame}(\mathsf{RatConstStream}_d(m), \mathsf{RatConstStream}_d(n)).$$

The ten-field fit is:

1. *the traditional target is constant rational observation-name coherence;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow} + \mathsf{StreamName}^{\uparrow}$*;*
3. *the carrier is* RatConstStream *over certified unary observations;*
4. *the classifier/history relation is* hsame*;*

5. *stability is certified-observation invariance;*
6. *the ledger is the hidden rational seed plus two unary observation witnesses;*
7. *the core theorem is this determinacy statement;*
8. *exactness says both observations read back to the same seed;*
9. *the standard bridge is ordinary constant stream extensionality;*
10. *the status is* ProofTarget.

Proof. Apply the point-sameness row of Definition 126.48 to the certified observation m. This gives $\mathsf{hsame}(\mathsf{RatConstStream}_d(m), d)$.

Apply the same row to the certified observation n, obtaining $\mathsf{hsame}(\mathsf{RatConstStream}_d(n), d)$. By symmetry, this gives $\mathsf{hsame}(d, \mathsf{RatConstStream}_d(n))$.

Compose the two history-sameness witnesses by transitivity through the hidden seed d. The result is $\mathsf{hsame}(\mathsf{RatConstStream}_d(m), \mathsf{RatConstStream}_d(n))$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatConstStream certified observation determinacy`

Theorem 126.50 (Rational stream-name constant reindexing classifier)**.** *For histories* $d, e : \mathsf{Hist}$ *and an observation reindexer* $r : \mathsf{Hist} \to \mathsf{Hist}$*, suppose* r *preserves certified unary observations:*

$$\forall n,\ \mathsf{UnaryHistory}(n) \to \mathsf{UnaryHistory}(r(n)).$$

If $\mathsf{RatHistoryCarrier}(d)$*, then* $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$ *and* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStre}$ $r, \mathsf{RatConstStream}_d)$*. If* $\mathsf{RatHistoryClassifier}(d, e)$*, then* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r)$*.*

Proof. For the carrier, unfold RatConstStream and RatStreamNameCarrier. Fix an observation history n with witness $\nu : \mathsf{UnaryHistory}(n)$. The reindexer supplies a certified unary observation witness for $r(n)$, but the reindexed constant stream still has value d, so the supplied $\mathsf{RatHistoryCarrier}(d)$ is the required pointwise carrier witness.

For the classifier against the original constant stream, unfold RatStreamNameClassifier and fix the same certified observation data (n, ν). Both stream values are d after expanding the constant stream, so RatUp carrier-reflexivity turns the carrier witness for d into the required point classifier $\mathsf{RatHistoryClassifier}(d, d)$.

For two reindexed constant streams, unfold RatStreamNameClassifier and fix n with witness $\nu : \mathsf{UnaryHistory}(n)$. The unary-preservation hypothesis certifies $r(n)$ as an admissible observation, while both endpoint stream values remain the seeds d and e. The supplied $\mathsf{RatHistoryClassifier}(d, e)$ is therefore reused pointwise. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName constant reindexing classifier`

Theorem 126.51 (Rational stream-name independently reindexed constant classifier)**.** *For histories* $d, e : \mathsf{Hist}$ *and observation schedules* $r, q : \mathsf{Hist} \to \mathsf{Hist}$*, if* $\mathsf{RatHistoryCarrier}(d)$*, then* $\mathsf{RatStreamNameCarrier}(\mathsf{RatC}$ $r)$*. If* $\mathsf{RatHistoryClassifier}(d, e)$*, then* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q)$*.*

The ten-field fit is:

1. *the traditional target is independently scheduled constant rational stream names;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *over* BHist*;*

3. *the carrier row is* RatStreamNameCarrier*;*
4. *the classifier row is* RatStreamNameClassifier*;*
5. *the schedule row is the pair of observation maps r and q;*
6. *the ledger is the two observation schedules together with the hidden rational seeds d and e;*
7. *the core theorem is this independently reindexed constant classifier;*
8. *exactness is pointwise reduction to the* $\mathsf{Rat}^{\uparrow}$ *classifier;*
9. *the downstream consumer is the* $\mathsf{Real}^{\uparrow}$ *independent reindexed constant bridge;*
10. *the status is* ProofTarget*.*

Proof. For the carrier, unfold RatConstStream and RatStreamNameCarrier. Fix an arbitrary certified unary observation n with witness $\nu : \mathsf{UnaryHistory}(n)$. The composite $\mathsf{RatConstStream}_d \circ r$ still evaluates to d, so the supplied $\mathsf{RatHistoryCarrier}(d)$ is reused as the pointwise carrier witness.

For the classifier, unfold RatConstStream and RatStreamNameClassifier. At the same kind of certified unary observation n, the left endpoint ignores $r(n)$ and evaluates to d, while the right endpoint ignores $q(n)$ and evaluates to e. Thus the supplied $\mathsf{RatHistoryClassifier}(d, e)$ is already the required pointwise classifier. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName independent reindexed constant classifier`

Theorem 126.52 (Rational stream-name independently reindexed constant point exactness)**.** *For histories $d, e : \mathsf{Hist}$ and observation maps $r, q : \mathsf{Hist} \to \mathsf{Hist}$, the pointwise stream-name carrier and classifier of independently reindexed constant streams recover exactly the RatUp history data:*

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r) \iff \mathsf{RatHistoryCarrier}(d)$$

and

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q) \iff \mathsf{RatHistoryClassifier}(d, e).$$

Proof. Unfold RatConstStream and RatStreamNameCarrier. In the forward carrier direction, read the stream carrier at the empty observation e; the reindexed constant stream evaluates to the seed history d there.

In the reverse carrier direction, fix an arbitrary certified unary observation. The composite $\mathsf{RatConstStream}_d \circ r$ still evaluates to d, so the seed carrier $\mathsf{RatHistoryCarrier}(d)$ supplies the pointwise stream carrier witness.

For the classifier equivalence, unfold RatConstStream and RatStreamNameClassifier. In the forward direction, read the stream classifier at e; both independently reindexed constant streams ignore their observation maps and return the endpoint histories d and e.

In the reverse direction, the supplied $\mathsf{RatHistoryClassifier}(d, e)$ already contains the endpoint carrier data and the point classifier between d and e. At every certified unary observation, the two stream endpoints again reduce to d and e, so the same point classifier repacks into the stream-name classifier. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName independent reindexed constant point exactness`

Theorem 126.53 (Rational constant stream empty-observation exactness)**.** *For histories h, k : Hist, the pointwise stream-name carrier and classifier of constant streams are equivalent to their RatUp point data at the empty unary observation:*

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_h) \iff \mathsf{RatHistoryCarrier}(\mathsf{RatConstStream}_h(\mathsf{e}))$$

and

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k) \iff \mathsf{RatHistoryClassifier}(\mathsf{RatConstStream}_h(\mathsf{e}), \mathsf{RatCon}$$

Proof. Unfold $\mathsf{RatStreamNameCarrier}$ and $\mathsf{RatConstStream}$. In the forward carrier direction, read the pointwise carrier witness at the empty unary observation e, obtaining the RatUp carrier at $\mathsf{RatConstStream}_h(\mathsf{e})$.

In the reverse carrier direction, fix an arbitrary certified unary observation. A constant stream has the same value there as at e, so the empty-observation carrier rebuilds the required pointwise carrier witness.

For the classifier equivalence, unfold $\mathsf{RatStreamNameClassifier}$ and the two constant streams. The forward direction reads the pointwise classifier witness at e.

In the reverse direction, the empty-observation classifier contains the endpoint carrier data and the point classifier between $\mathsf{RatConstStream}_h(\mathsf{e})$ and $\mathsf{RatConstStream}_k(\mathsf{e})$. At every certified unary observation, the two constant endpoints again evaluate as they do at e, so the same RatUp classifier repacks into the stream-name classifier. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatConstStream empty point exactness`

Theorem 126.54 (Rational stream-name constant arbitrary observation exactness)**.** *For histories h, k : Hist and an observation history n : Hist with $\mathsf{UnaryHistory}(n)$, the pointwise stream-name carrier and classifier of constant streams recover exactly the RatUp data at that certified observation:*

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_h) \iff \mathsf{RatHistoryCarrier}(\mathsf{RatConstStream}_h(n))$$

and

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k) \iff \mathsf{RatHistoryClassifier}(\mathsf{RatConstStream}_h(n), \mathsf{RatCo}$$

Proof. Unfold $\mathsf{RatStreamNameCarrier}$ and $\mathsf{RatConstStream}$. In the forward carrier direction, read the pointwise carrier witness at the supplied certified unary observation n, obtaining the RatUp carrier at $\mathsf{RatConstStream}_h(n)$.

In the reverse carrier direction, fix an arbitrary certified unary observation m. Since $\mathsf{RatConstStream}_h(m)$ and $\mathsf{RatConstStream}_h(n)$ are both the hidden seed h, the displayed carrier at n supplies the required pointwise carrier witness at m.

For the classifier equivalence, unfold $\mathsf{RatStreamNameClassifier}$ and the two constant streams. The forward direction reads the pointwise classifier witness at the supplied certified unary observation n.

In the reverse direction, the classifier at n contains the endpoint carrier data and the point classifier between $\mathsf{RatConstStream}_h(n)$ and $\mathsf{RatConstStream}_k(n)$. At every certified unary observation m, the two constant endpoints again evaluate to the same hidden seeds, so the same RatUp classifier repacks into the stream-name classifier. □

Theorem 126.55 (Rational stream-name constant point exactness)**.** *For histories $h, k : \mathsf{Hist}$, the pointwise stream-name carrier and classifier of constant streams recover exactly the RatUp history data:*

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_h) \iff \mathsf{RatHistoryCarrier}(h)$$

and

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k) \iff \mathsf{RatHistoryClassifier}(h, k).$$

Proof. Unfold $\mathsf{RatStreamNameCarrier}$ and $\mathsf{RatConstStream}$. In the forward direction, read the pointwise carrier witness at observation index 0. In the reverse direction, fix an arbitrary observation index n; since the constant stream value is h, the supplied $\mathsf{RatHistoryCarrier}(h)$ witness is exactly the required pointwise witness.

For the classifier statement, unfold $\mathsf{RatStreamNameClassifier}$ and the two constant streams. In the forward direction, read the pointwise classifier witness at observation index 0. In the reverse direction, fix an arbitrary observation index n; since the endpoint stream values are h and k, the supplied $\mathsf{RatHistoryClassifier}(h, k)$ witness is exactly the required pointwise classifier at that observation. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName constant point exactness`

Theorem 126.56 (Rational stream-name constant finite-window exactness)**.** *For histories $h, k : \mathsf{Hist}$ and a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, if the bundle contains a certified unary observation,*

$$\exists n : \mathsf{Hist},\ \mathsf{InBundle}(n, \mathcal{P}) \wedge \mathsf{UnaryHistory}(n),$$

then the finite-window classifier on the two constant streams is exactly the hidden RatUp classifier:

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k, \mathcal{P}) \iff \mathsf{RatHistoryClassifier}(h, k).$$

Proof. Unfold $\mathsf{RatStreamNameFiniteWindowClassifier}$ and the two constant streams. In the forward direction, use the assumed bundled unary observation n; applying the finite-window classifier at this inhabited probe reads the point classifier between $\mathsf{RatConstStream}_h(n)$ and $\mathsf{RatConstStream}_k(n)$, which is exactly $\mathsf{RatHistoryClassifier}(h, k)$.

In the reverse direction, fix an arbitrary bundled unary observation m. After unfolding the constant streams, the supplied hidden endpoint classifier $\mathsf{RatHistoryClassifier}(h, k)$ is already the required point classifier at m. Since m was arbitrary among bundled unary observations, the finite-window classifier follows. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier constant exactness`

Theorem 126.57 (Rational constant-stream finite-window coverage)**.** *For histories $h, k : \mathsf{Hist}$ and a probe bundle $\mathcal{P} : \mathsf{ProbeBundle}$, suppose the finite window is inhabited by a certified unary observation:*

$$\exists n : \mathsf{Hist},\ \mathsf{InBundle}(n, \mathcal{P}) \wedge \mathsf{UnaryHistory}(n).$$

Then the window classifier for the two constant rational streams is covered by the hidden RatUp endpoint classifier. More precisely, $\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k, \mathcal{P})$ *determines* $\mathsf{RatHistoryClassifier}(h, k)$*, and conversely any* $\mathsf{RatHistoryClassifier}(h, k)$ *supplies the classifier row for every selected unary observation of $\mathcal{P}$.*

Proof. Apply Definition 126.56 to the inhabited ProbeBundle window. The forward implication reads the hidden endpoint classifier from any bundled unary probe.

For the converse, unfold RatConstStream. At each selected n : Hist with InBundle$(n, \mathcal{P})$ and UnaryHistory(n), both stream observations reduce to h and k. The supplied RatUp endpoint classifier is therefore exactly the point row required by RatStreamNameFiniteWindowClassifier; if the selected endpoint is displayed through hsame, use Definition 37.54 before repacking. □

Theorem 126.58 (Rational constant-stream finite-window soundness)**.** *For histories h, k : Hist, a probe bundle $\mathcal{P}$: ProbeBundle, and a certified selected observation n : Hist, if* InBundle$(n, \mathcal{P})$*,* UnaryHistory(n)*, and*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k, \mathcal{P})$$

hold, then the selected unary observation transports to the RatUp endpoint row

$$\mathsf{RatHistoryClassifier}(\mathsf{RatConstStream}_h(n), \mathsf{RatConstStream}_k(n)).$$

Proof. Unfold RatStreamNameFiniteWindowClassifier. Its soundness clause is a universal row over the selected unary observations of the ProbeBundle.

Apply that row to the displayed n, the membership witness InBundle$(n, \mathcal{P})$, and the certificate UnaryHistory(n). Unfolding RatConstStream identifies the two observed endpoints with h and k, so the returned point classifier is the desired RatUp endpoint row; the reflexive endpoint case is the RatUp classifier row of Definition 37.37. □

Theorem 126.59 (Rational constant-stream finite-window completeness)**.** *For histories h, k : Hist and a probe bundle $\mathcal{P}$: ProbeBundle, suppose the finite window is inhabited by a certified unary observation and* RatHistoryClassifier(h, k) *holds. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(\mathsf{RatConstStream}_h, \mathsf{RatConstStream}_k, \mathcal{P})$$

holds for every such certified finite window.

Proof. Apply the reverse direction of Definition 126.56. The inhabited window supplies the nonempty ProbeBundle premise, and the given RatUp endpoint classifier supplies the hidden row.

Equivalently, fix any selected unary observation. By Definition 126.48, each constant stream is the stream-name witness obtained from its hidden rational endpoint. Unfold RatConstStream; the selected observation again reads h and k, so the same endpoint classifier repacks the finite-window classifier, with component transport handled by Definition 37.54. □

Theorem 126.60 (Rational stream-name constant bridge package)**.** *For histories d, e : Hist and an arbitrary observation reindexer r : Hist $\to$ Hist, every witness ρ :* RatHistoryClassifier(d, e) *determines the concrete constant stream-name bridge package consisting of:*

1. *the carrier rows* RatStreamNameCarrier$(\mathsf{RatConstStream}_d)$ *and* RatStreamNameCarrier$(\mathsf{RatConstStream}_d \circ r)$*;*
2. *the classifier row* RatStreamNameClassifier$(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e)$*;*
3. *the reindexed classifier rows* RatStreamNameClassifier$(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_d)$ *and* RatStreamNameClassifier$(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r)$*;*

4. *the exactness equivalence*

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \iff \mathsf{RatHistoryClassifier}(d, e).$$

The compact ten-field fit is:

1. *the traditional target is ordinary constant rational observation names;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *plus* $\mathsf{StreamName}^{\uparrow}$ *over* BHist*;*
3. *the carrier row is* $\mathsf{RatStreamNameCarrier}$ *on the hidden seed stream and its certified observation reindexing;*
4. *the classifier row is* $\mathsf{RatStreamNameClassifier}$ *on constant rational stream names;*
5. *the stability rows are arbitrary observation scheduling for constant streams and same-seed comparison against the unreindexed constant stream;*
6. *the ledger is the rational seed pair* (d, e) *together with the hidden observation schedule* r*;*
7. *the core theorem is this constant bridge package;*
8. *exactness is the reduction of constant stream-name classification to* $\mathsf{RatHistoryClassifier}(d, e)$*;*
9. *the standard bridge is ordinary constant rational streams read as certified stream names;*
10. *the status is* $\mathsf{ProofTarget}$.

Proof. Project the carrier witness for d from the RatUp classifier $\rho : \mathsf{RatHistoryClassifier}(d, e)$. Apply Definition 126.48 to this carrier witness to obtain the unreindexed carrier row, and apply the same theorem to ρ to obtain the unreindexed classifier between $\mathsf{RatConstStream}_d$ and $\mathsf{RatConstStream}_e$.

Next apply Definition 126.51 with the schedules r and the identity schedule. Its carrier clause supplies $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$. RatUp carrier-reflexivity turns the carrier witness for d into $\mathsf{RatHistoryClassifier}(d, d)$, and the classifier clause with endpoints d, d supplies the same-seed classifier from $\mathsf{RatConstStream}_d \circ r$ to $\mathsf{RatConstStream}_d$.

Apply the same independent reindexing theorem with schedules r and r to the endpoint classifier ρ. This supplies the classifier from $\mathsf{RatConstStream}_d \circ r$ to $\mathsf{RatConstStream}_e \circ r$.

The exactness row is precisely Definition 126.55: after expanding the two constant streams, stream-name classification is equivalent to the RatUp classifier on the two hidden histories d and e.

Repack these carrier, classifier, reindexing, and exactness witnesses with the ledger (d, e, r). The observation schedule is hidden ledger data for constant streams, because the stream value ignores the chosen schedule. The resulting object is the named bridge package for constant rational observation names. □

Theorem 126.61 (Rational stream-name constant bridge package exactness)**.** *For histories* $d, e : \mathsf{Hist}$ *and an arbitrary observation reindexer* $r : \mathsf{Hist} \to \mathsf{Hist}$*, write* $\mathsf{K}(d, e, r)$ *for the conjunction of the following concrete constant stream-name bridge rows:*

1. $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d)$*;*
2. $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$*;*
3. $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e)$*;*

4. $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_d)$;
5. $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ r)$;
6. *the exactness row*

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e) \Longleftrightarrow \mathsf{RatHistoryClassifier}(d, e).$$

Then the whole package is exact over the underlying RatUp classifier:

$$\mathsf{K}(d, e, r) \Longleftrightarrow \mathsf{RatHistoryClassifier}(d, e).$$

The compact ten-field fit is:

1. *the traditional target is* $\mathsf{StreamName}^{\uparrow}$ *constant rational observation names;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *over* BHist*;*
3. *the carrier row is the pair of carrier conjuncts in* $\mathsf{K}(d, e, r)$*;*
4. *the classifier row is the three classifier conjuncts in* $\mathsf{K}(d, e, r)$*;*
5. *the stability rows are arbitrary observation scheduling for constant streams and the two reindexed constant classifier rows;*
6. *the ledger is the triple* (d, e, r)*, with* r *retained only as hidden observation schedule data;*
7. *the core theorem is package-level exactness for constant rational stream names;*
8. *exactness is the displayed iff between* $\mathsf{K}(d, e, r)$ *and* $\mathsf{RatHistoryClassifier}(d, e)$*;*
9. *the downstream consumer is the* $\mathsf{Real}^{\uparrow}$ *constant rational embedding;*
10. *the status is* $\mathsf{ProofTarget}$.

Proof. In the forward direction, assume a witness of $\mathsf{K}(d, e, r)$. Project its unreindexed classifier row $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d, \mathsf{RatConstStream}_e)$ and its exactness row. Applying the forward half of that row gives $\mathsf{RatHistoryClassifier}(d, e)$.

In the reverse direction, assume $\rho : \mathsf{RatHistoryClassifier}(d, e)$. Apply Definition 126.60 to ρ and the arbitrary schedule r. The resulting bridge package supplies exactly the two carrier rows, the unreindexed classifier row, the two reindexed classifier rows, and the constant-point exactness row listed in $\mathsf{K}(d, e, r)$. □

Theorem 126.62 (Rational stream-name independently scheduled constant bridge package exactness)**.** *For histories* $d, e : \mathsf{Hist}$ *and arbitrary observation schedules* $r, q : \mathsf{Hist} \to \mathsf{Hist}$*, write* $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$ *for the conjunction of the following independently scheduled constant stream-name bridge rows:*

1. $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$*;*
2. $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_e \circ q)$*;*
3. $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q)$*;*
4. *the exactness row*

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q) \Longleftrightarrow \mathsf{RatHistoryClassifier}(d, e).$$

Then the independently scheduled package is exact over the underlying RatUp classifier:

$$\mathsf{K}_{\mathrm{ind}}(d, e, r, q) \iff \mathsf{RatHistoryClassifier}(d, e).$$

The compact ten-field fit is:

1. *the traditional target is independently scheduled constant rational stream names;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *plus* $\mathsf{StreamName}^{\uparrow}$ *over* BHist*;*
3. *the carrier row is the pair* $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$ *and* $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStrea}$ $q)$*;*
4. *the classifier row is* $\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q)$*;*
5. *the schedule row is the independent pair of observation schedules* r *and* q*;*
6. *the ledger is the quadruple* (d, e, r, q)*;*
7. *the core theorem is independently scheduled constant bridge package exactness;*
8. *exactness is the displayed iff between* $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$ *and* $\mathsf{RatHistoryClassifier}(d, e)$*;*
9. *the downstream consumer is the* $\mathsf{Real}^{\uparrow}$ *independent reindexed constant rational embedding;*
10. *the status is* $\mathsf{ProofTarget}$.

Proof. In the forward direction, assume a witness of $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$. Project its independently scheduled classifier row and its exactness row. Applying the forward half of that row gives $\mathsf{RatHistoryClassifier}(d, e)$.

In the reverse direction, assume $\rho : \mathsf{RatHistoryClassifier}(d, e)$. First apply Definition 126.51 with the endpoint d and schedule r to the left carrier contained in ρ. Apply the same theorem with endpoint e and schedule q to the right carrier contained in ρ. These two applications supply the carrier rows of $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$.

Next apply Definition 126.51 to ρ with the schedules r and q. This supplies the independently scheduled classifier row.

Finally apply Definition 126.52. Its classifier clause is exactly the displayed equivalence between the independently scheduled stream-name classifier and $\mathsf{RatHistoryClassifier}(d, e)$. Repacking these carrier, classifier, and exactness witnesses gives $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$. □

Theorem 126.63 (Rational stream-name independently scheduled constant bridge package transitivity)**.** *For histories* $d, e, f : \mathsf{Hist}$ *and arbitrary observation schedules* $r, q, p : \mathsf{Hist} \to \mathsf{Hist}$*, if* $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$ *and* $\mathsf{K}_{\mathrm{ind}}(e, f, q, p)$ *hold in the sense of Definition 126.62, then* $\mathsf{K}_{\mathrm{ind}}(d, f, r, p)$ *holds.*

The compact ten-field fit is:

1. *the traditional target is independently scheduled constant rational stream-name composition;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *plus* $\mathsf{StreamName}^{\uparrow}$*;*
3. *the carrier row is the endpoint pair inside the* $\mathsf{K}_{\mathrm{ind}}$ *package;*
4. *the classifier row is the independently scheduled* $\mathsf{K}_{\mathrm{ind}}$ *classifier row;*
5. *the stability row is transitivity of the package relation;*
6. *the ledger is the sextuple* (d, e, f, r, q, p)*;*
7. *the core theorem is this package transitivity;*

8. *exactness is reduction through* RatHistoryClassifier*;*
9. *the standard bridge is composition of constant rational streams;*
10. *the status is* ProofTarget.

Proof. Let κ_{de} witness $\mathsf{K}_{\mathrm{ind}}(d, e, r, q)$, and let κ_{ef} witness $\mathsf{K}_{\mathrm{ind}}(e, f, q, p)$. Apply the forward direction of Definition 126.62 to κ_{de} and κ_{ef}, obtaining $\mathsf{RatHistoryClassifier}(d, e)$ and $\mathsf{RatHistoryClassifier}(e, f)$.

Next apply Definition 37.53 to these two classifier witnesses. This composes the RatUp source data and yields $\mathsf{RatHistoryClassifier}(d, f)$.

Finally apply the reverse direction of Definition 126.62 with endpoints d, f and schedules r, p. The result repacks the two constant carrier rows, the independently scheduled classifier row, and the exactness row into $\mathsf{K}_{\mathrm{ind}}(d, f, r, p)$. □

Theorem 126.64 (Rational stream-name public certificate)**.** *The carrier* RatStreamNameCarrier *and classifier* RatStreamNameClassifier*, read over unary* BHist *observations and RatUp point carriers and classifiers, supply the public* $\mathsf{StreamName}^{\uparrow}$ *certificate with the following ten fields:*

1. *the traditional target is the certified countable rational-observation name interface;*
2. *the BEDC source is* $\mathsf{Rat}^{\uparrow}$ *over* BHist*;*
3. *the carrier row is* RatStreamNameCarrier*;*
4. *the classifier row is* RatStreamNameClassifier*;*
5. *reflexivity, symmetry, transitivity, endpoint-carrier recovery, and history-sameness transport are the rows of Definition 126.13;*
6. *reindexing stability and mapped-stream stability are the two clauses of Definition 126.19;*
7. *composed reindexing stability is Definition 126.20;*
8. *inhabited constant streams are supplied by Definition 126.48;*
9. *constant reindexing classification is Definition 126.50;*
10. *constant-point and independently reindexed constant-point exactness, with finite unary observation data in the ledger and exactness pointwise over the* $\mathsf{Rat}^{\uparrow}$ *classifier, are Definition 126.55 and Definition 126.52.*

Proof. Unfold the two public rows. The carrier row is the predicate assigning to each stream $s : \mathsf{Hist} \to \mathsf{Hist}$ the pointwise requirement $\mathsf{RatHistoryCarrier}(s(n))$ at every certified unary observation $n : \mathsf{BHist}$, and the classifier row assigns to each pair of streams the pointwise RatUp classifier between their two observed histories.

The equivalence-relation and transport rows are exactly the five clauses of Definition 126.13. They read a stream certificate at an arbitrary unary observation and apply the corresponding RatUp point theorem, so no global stream data other than the two unfolded rows is introduced.

The structural action rows come from Definition 126.19 and Definition 126.20. The first theorem handles a single certified observation reindexer and a RatUp-preserving history map; the second repacks two unary-preserving reindexers into their composite and then applies the same pointwise classifier and carrier reading.

The inhabited and constant-name rows are supplied by Definition 126.48 and Definition 126.50. The witness row uses a RatUp history carrier as the seed for a constant stream, while the reindexing row observes that every reindexed constant stream still has the same endpoint seed at each certified unary observation.

Finally, Definition 126.55 identifies the constant stream carrier and classifier with the underlying RatUp carrier and classifier, and Definition 126.52 gives the same exactness after independent observation reindexing on the two constant endpoints. Thus the ledger consists of finite unary observation data together with the pointwise RatUp witnesses, and the exactness field is pointwise over the RatUp classifier. These ten rows are precisely the public $\mathsf{StreamName}^{\uparrow}$ certificate assembled from the named component theorems. □

Theorem 126.65 (StreamName constant finite-window readback)**.** *Let q : Hist be a displayed rational-history seed and let I be a displayed finite unary observation window. A constant rational $\mathsf{StreamName}^{\uparrow}$ read over I factors pointwise through the same hidden seed q: every observed entry is classified by the $\mathsf{Rat}^{\uparrow}$ carrier row for q, the finite window ledger records only the inspected unary observations,* hsame *transport identifies each readback with q,* Cont *replay replays the same finite row family,* Pkg *provenance names the constant-stream source, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row packages these data. The packet exports no quotient stream equality, no selected limit, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Read the constant stream through Definition 126.47. By definition, every unary observation in the finite window I returns the same rational-history seed q.

Apply Definition 126.48 at each displayed observation. The pointwise carrier and classifier rows are therefore the $\mathsf{Rat}^{\uparrow}$ rows for the single seed q, and the hsame row transports every readback back to that seed.

The finite window contributes only the inspected observation ledger. Replay and provenance package this finite family as a local $\mathsf{StreamName}^{\uparrow}$ certificate, while no cited row contains quotient stream equality, a selected limit, countable choice, or ambient real completeness. □

126.3 StreamName constant finite-window boundary

Theorem 126.66 (StreamName constant subwindow exactness)**.** *Let a, b : Hist be rational-history endpoints and let $I \subseteq J$ be inhabited finite unary observation windows. For the constant rational stream window package on J, the restricted package on I recovers exactly the same* $\mathsf{RatHistoryClassifier}(a, b)$ *row. No extra stream classifier, completion witness, or host equality row is read by the subwindow.*

Proof. Project the finite constant stream window on J through Definition 126.9. This gives the same constant-stream carrier and classifier rows over the inhabited subwindow I.

Apply Definition 126.56 to the restricted window. Because every observation in the window reads the same two constant endpoints, the window classifier collapses to the single underlying $\mathsf{RatHistoryClassifier}(a, b)$ row.

Finally use Definition 126.301 to identify the subwindow package as a public $\mathsf{StreamName}^{\uparrow}$ finite-window interface. The interface contributes only the displayed carrier, classifier, and ledger rows, so no additional comparison or completion row is introduced. □

Theorem 126.67 (StreamName constant RealUp boundary)**.** *A finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of the constant rational stream package may read only the* $\mathsf{Rat}^{\uparrow}$ *point classifier row* $\mathsf{RatHistoryClassifier}(a, b)$ *and the finite* $\mathsf{ProbeBundle}$ *ledger carried by the* $\mathsf{StreamName}^{\uparrow}$ *window. It does not obtain a completeness claim, a Cauchy-modulus row, or any equality beyond the displayed rational point classifier.*

Proof. Start with the public finite-window interface of Definition 126.301. It exposes the window carrier, classifier, and finite ProbeBundle ledger as the only rows available to a downstream consumer.

For a constant rational stream window, apply Definition 126.66. The classifier read by any inhabited subwindow is exactly the point classifier $\mathsf{RatHistoryClassifier}(a, b)$.

Thus a $\mathsf{Real}^{\uparrow}$ finite-window consumer can depend on the rational point row and on the ledger naming the observed probes. The cited interface contains no regularity, convergence, or completion field, so those obligations remain outside this boundary. □

Theorem 126.68 (StreamName constant RegSeqRat boundary)**.** *A* RegSeqRat *construction may consume the same constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window evidence as a source of rational point-classifier and* ProbeBundle *ledger data, while the regularity rows of* RegSeqRat *remain separate obligations not supplied by the constant-window package.*

Proof. Read the finite-window evidence through Definition 126.301. The rows available at the boundary are the carried observations, the stream-name classifier, and the finite ledger.

For constant rational streams, combine Definition 126.9 with Definition 126.56. Any inhabited subwindow keeps the same endpoints and therefore recovers the same $\mathsf{RatHistoryClassifier}(a, b)$ row.

Repack these rows as the StreamName input to RegSeqRat. Since none of the cited window theorems asserts a regularity modulus, bounded variation row, or sequence-comparison law, those rows must be supplied by the RegSeqRat certificate itself. □

Theorem 126.69 (StreamName constant window overlap exactness)**.** *Let* $a, b : \mathsf{Hist}$ *be rational-history endpoints, and let* I, J *be inhabited finite unary windows for constant rational streams over the same BEDC source* BHist*. Suppose* K *is a certified common subwindow of* I *and* J*, with every probe in* K *carried by* UnaryHistory *and by the finite* ProbeBundle *ledgers of both windows. Then the restrictions of the two constant-window packages to* K *expose the same* $\mathsf{RatHistoryClassifier}(a, b)$ *row, with the StreamName comparison read through the displayed* hsame *rows on the overlap.*

Consequently, any $\mathsf{Real}^{\uparrow}$ *finite-window read surface built from the two restricted packages has the same* $\mathsf{StreamName}^{\uparrow}$ *component on* K*. The overlap surface contains only the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface rows over* BHist*,* UnaryHistory*,* ProbeBundle*,* hsame*, and* $\mathsf{RatHistoryClassifier}(a, b)$*; it does not use a quotient, completion row, host index equality, or hidden stream datum.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameConstantWindow overlap exactness`

Proof. Restrict the I-window package to the certified overlap K. By Definition 126.66, the restricted package recovers exactly the row $\mathsf{RatHistoryClassifier}(a, b)$, because the ambient stream is constant and every probe of K is an inhabited unary observation.

Repeat the same restriction for the J-window package. The second application of Definition 126.66 yields the same endpoint row $\mathsf{RatHistoryClassifier}(a, b)$ on the same ProbeBundle overlap ledger. The displayed hsame clauses on K therefore identify the two public StreamName components without comparing host streams.

Now read both packages as possible inputs to a finite-window $\mathsf{Real}^{\uparrow}$ consumer. By Definition 126.67, such a consumer may read only the rational point-classifier row and the finite probe ledger from the constant $\mathsf{StreamName}^{\uparrow}$ window.

Finally apply Definition 38.326. Since both restricted read surfaces present the same overlap rows just listed, the RealUp finite-window surface repacks the same StreamName component over K. No quotient, completion witness, host index, or hidden stream equality is added in that repacking. □

Theorem 126.70 (StreamName constant window union exactness)**.** *Let a, b : Hist be rational-history endpoints, and let I, J be inhabited finite unary constant rational* StreamName$^{\uparrow}$ *windows over the same BEDC source* BHist*. Suppose both windows expose the row* RatHistoryClassifier(a, b)*. Then the finite union window represented by* bundleAppend(I, J) *exposes the same classifier row.*

Every selected observation of bundleAppend(I, J) *restricts back to one of the original constant-window rows. A finite-window* Real$^{\uparrow}$ *or* RegSeqRat *consumer of this union window reads only* StreamName$^{\uparrow}$*,* ProbeBundle*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and* RatHistoryClassifier(a, b)*. It receives no completion row, quotient, host stream equality, or hidden selector.*

Proof. Apply Definition 126.66 to the inhabited window I. Its selected constant observations all read the displayed row RatHistoryClassifier(a, b), and the subwindow proof uses only the public finite-window StreamName rows.

Apply the same subwindow exactness theorem to J. Since the endpoints are the same a, b, the second package exposes the identical rational-history classifier row, again without a host stream comparison.

Use the finite-window structural laws recorded by Definition 126.301. The bundleAppend membership law places every selected observation of the union bundle in either the I branch or the J branch, and each branch has already been reduced to the same constant-window classifier row.

Finally apply Definition 126.67 and Definition 126.68. These boundaries say that downstream Real$^{\uparrow}$ and RegSeqRat finite-window consumers can read only the listed StreamName, probe-bundle, unary-history, transport, continuation, package, and rational-history classifier rows. The appended union therefore adds no completion, quotient, host stream equality, or hidden selector. □

Theorem 126.71 (StreamName constant union RegSeqRat Real readback preservation)**.** *Let a, b : Hist be rational-history endpoints, and let I, J be inhabited finite unary constant rational windows over the same* BHist *source. Suppose both windows expose the row* RatHistoryClassifier(a, b)*. Then the union window* bundleAppend(I, J) *preserves the same selected* RegSeqRat$^{\uparrow}$ *tail read and the same finite-window* Real$^{\uparrow}$ *readback surface.*

The preserved readback surface is exactly the rows StreamName$^{\uparrow}$*,* ProbeBundle*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and* RatHistoryClassifier(a, b)*. It contains no completion row, quotient, host stream equality, hidden selector, or selected real limit.*

Proof. First apply Definition 126.70. The appended bundle has no classifier branch except the common row RatHistoryClassifier(a, b), because every selected observation comes from the I side or the J side and both sides expose the same endpoint row.

Next use Definition 126.68. The common classifier row and finite ProbeBundle ledger repack as the RegSeqRat$^{\uparrow}$ finite-window input, while the regularity and tail obligations remain external to the constant-window package.

Then apply Definition 126.67. The Real$^{\uparrow}$ readback uses the same rational point-classifier row and the same finite probe ledger already read by the RegSeqRat$^{\uparrow}$ boundary. It receives no completion field or selected limit from the appended window.

Finally exhaust the coordinates of the public StreamName boundary. The only rows used in the three cited results are StreamName$^{\uparrow}$, ProbeBundle, UnaryHistory, hsame, Cont, Pkg, and RatHistoryClassifier(a, b), so the union cannot contribute a quotient, host stream equality, hidden selector, completion row, or selected real limit. □

Theorem 126.72 (StreamName constant directed-overlap exactness)**.** *Let a, b : Hist be rational-history endpoints, and let $\{I_i\}_{i \in I}$ be a directed finite family of inhabited constant rational* StreamName$^{\uparrow}$ *windows over the same* BHist *source. For every certified common overlap in the directed family,*

the restricted windows expose the same RatHistoryClassifier(a, b) *row, with comparison read only through the displayed* hsame *rows and the finite* ProbeBundle *ledgers.*

Consequently, any RegSeqRat *or* Real$^{\uparrow}$ *finite-window consumer of the directed family reads only that single rational-history classifier row together with the finite* ProbeBundle *ledger. The directed system supplies no quotient row, completion row, host stream equality, hidden regularity modulus, or external finite-window selector.*

Proof. Start with one certified common overlap in the directed family. By Definition 126.69, the two restricted constant-window packages on that overlap expose the same RatHistoryClassifier(a, b) row and the same finite probe ledger.

Now pass along the directed finite family. Iterating the overlap exactness statement over the directed overlaps identifies each exposed constant-window classifier with the same endpoint row. Directedness ensures that every pair of member windows is compared inside a certified common window already carrying the displayed hsame and ProbeBundle rows.

For a RegSeqRat consumer, apply Definition 126.68. The consumer may use the constant StreamName row and the finite probe ledger, while all regularity obligations remain external to the constant-window boundary.

For a Real$^{\uparrow}$ consumer, apply Definition 126.67. It reads the same rational-history classifier row and the same finite probe ledger, with no completion or host equality data added by the directed overlap system. □

Theorem 126.73 (StreamName constant directed RegSeqRat Real tail boundary)**.** *Let* a, b : Hist *be rational-history endpoints, and let* $\{I_i\}_{i \in I}$ *be a directed finite family of inhabited constant rational* StreamName$^{\uparrow}$ *windows over one* BHist *source. Suppose every certified overlap in the directed family exposes the same* RatHistoryClassifier(a, b) *row, the same finite* ProbeBundle *ledger, and the displayed* UnaryHistory*,* hsame*,* Cont*, and* Pkg *rows.*

Then any RegSeqRat *consumer, and any finite-window* Real$^{\uparrow}$ *consumer factoring through that* RegSeqRat *surface, obtains one selected* RegSeqRat *tail-classifier row. The handoff surface is exhausted by the* StreamName$^{\uparrow}$ *finite-window rows, the selected* RegSeqRat *spine,* UnaryHistory*,* ProbeBundle*,* hsame*,* Cont*,* Pkg*, and the rational classifier* RatHistoryClassifier(a, b)*. It does not read completion, quotient, host stream equality, order or arithmetic rows, or standard bridge rows.*

Proof. First apply Definition 126.72. The directed family presents one rational-history classifier row on every certified overlap, and the comparison uses only the displayed hsame rows and finite ProbeBundle ledgers.

Next use Definition 126.68. It repacks the constant StreamName$^{\uparrow}$ evidence as the finite StreamName input that a RegSeqRat construction may consume, while leaving regularity rows as RegSeqRat obligations rather than StreamName data.

For a finite-window Real$^{\uparrow}$ consumer, use Definition 126.67. The RealUp side can read the same rational classifier and the same finite probe ledger, but it does not add completion or host-equality information to the window surface.

Finally apply Definition 38.360. The matching StreamName classifier, selected RegSeqRat spine, unary observation rows, and shared Cont/hsame ledger determine a single selected tail-classifier readback row. Since every input row in that application is one of the displayed constant-window or RegSeqRat rows, no quotient, arithmetic, order, completion, host stream equality, or standard bridge row can enter the selected-tail boundary. □

Theorem 126.74 (StreamName constant directed RealUp source uniqueness)**.** *Let* a, b : Hist *be rational-history endpoints, and let* $\{I_i\}_{i \in I}$ *and* $\{J_j\}_{j \in J}$ *be two directed finite families of inhabited*

constant rational $\mathsf{StreamName}^{\uparrow}$ *windows over the same* BHist *source. Suppose a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of the first family and a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of the second family overlap on a certified common directed subwindow* K*, with the displayed* $\mathsf{UnaryHistory}$*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* Pkg *rows inherited from both directed systems.*

Then the public $\mathsf{StreamName}^{\uparrow}$ *component exposed to the two* $\mathsf{Real}^{\uparrow}$ *consumers on* K *is uniquely the constant rational stream row determined by* $\mathsf{RatHistoryClassifier}(a, b)$ *and the finite* $\mathsf{ProbeBundle}$ *ledger on* K*. No alternative* $\mathsf{RegSeqRat}$ *source row,* $\mathsf{Real}^{\uparrow}$ *source row, or host stream row can be read at that overlap without first passing through the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window carrier and classifier.*

Proof. Restrict both directed constant-window families to the common directed subwindow K. The restricted packages still carry the same BHist source, the inherited unary observation rows, and the finite $\mathsf{ProbeBundle}$ ledgers displayed by the two original systems.

Apply Definition 126.72 to the restricted directed families. It identifies the public StreamName component on K with the single rational-history classifier row $\mathsf{RatHistoryClassifier}(a, b)$, using only the displayed hsame rows and the finite probe ledgers.

Next use Definition 126.67. A finite-window $\mathsf{Real}^{\uparrow}$ consumer may inspect only that StreamName carrier, classifier, and ledger surface. Thus the consumer has no independent access to a completion row, host stream equality, or an external source selector.

Finally apply Definition 126.73. Any $\mathsf{RegSeqRat}$ tail seen by the $\mathsf{Real}^{\uparrow}$ side is selected through the same constant StreamName finite-window rows and the displayed $\mathsf{Cont}/\mathsf{hsame}$ ledger. Therefore every available RealUp source row factors through the unique StreamName component on K. □

Theorem 126.75 (StreamName constant directed RealUp seal classifier determinacy)**.** *Let* $a, b : \mathsf{Hist}$ *be rational-history endpoints, and let* $\{I_i\}_{i \in I}$ *and* $\{J_j\}_{j \in J}$ *be directed finite inhabited constant rational* $\mathsf{StreamName}^{\uparrow}$ *window families over those endpoints. Assume that their directed overlap rows expose the same* $\mathsf{RatHistoryClassifier}(a, b)$ *row, the same finite* $\mathsf{ProbeBundle}$ *ledger, and the displayed* $\mathsf{UnaryHistory}$*,* hsame*,* Cont*, and* Pkg *rows. Let two finite-window* $\mathsf{Real}^{\uparrow}$ *seals consume these families through* $\mathsf{RegSeqRat}$ *stream-certificate spines compatible with the displayed constant-window rows.*

Then the two seals export the same $\mathsf{Real}^{\uparrow}$ *observation classifier row. The classifier factors through the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window package, the* $\mathsf{RegSeqRat}$ *constant stream spine, and the displayed seal handoff. It reads no completion, quotient, host stream equality, hidden selector, arithmetic or order package, or standard bridge.*

Proof. First apply Definition 126.74. The two seals see the same public $\mathsf{StreamName}^{\uparrow}$ source on every common directed subwindow, namely the constant rational stream row determined by $\mathsf{RatHistoryClassifier}(a, b)$ and the shared finite $\mathsf{ProbeBundle}$ ledger.

Next use Definition 38.206. The common constant StreamName row supplies compatible $\mathsf{RegSeqRat}$ stream-certificate spines for the reindexed constant streams. Apply Definition 38.207 to those spines and to the displayed seal classifier row; the resulting package is the RealUp observation classifier exported by each finite-window seal.

Finally use Definition 126.73. That boundary keeps the selected $\mathsf{RegSeqRat}$ tail and the RealUp handoff inside the displayed $\mathsf{UnaryHistory}$, $\mathsf{ProbeBundle}$, hsame, Cont, Pkg, and rational-classifier rows. Since both seals consume exactly that finite surface, no completion, quotient, host stream equality, hidden selector, arithmetic or order package, or standard bridge can distinguish their exported classifier rows. □

Theorem 126.76 (StreamName constant directed RealUp seal terminality)**.** *Let a, b : Hist be rational-history endpoints, and let $\{I_i\}_{i\in I}$ and $\{J_j\}_{j\in J}$ be directed finite inhabited constant rational* StreamName$^{\uparrow}$ *window families over those endpoints and over the same displayed* RatHistoryClassifier(a, b) *source. Suppose their finite rows present the same* ProbeBundle *ledger, the same* UnaryHistory *observations, and the same displayed* hsame, Cont, *and* Pkg *coordinates. Fix the selected* RegSeqRat *stream-certificate spine obtained from the constant-window rows.*

Then every finite-window Real$^{\uparrow}$ *consumer preserving that constant-window source and that selected* RegSeqRat *spine factors through the unique seal-classifier row of Definition 126.75. The factorization exposes no second* Real$^{\uparrow}$ *source row, no hidden selector, no quotient stream equality, no completion witness, no arithmetic or order package, and no standard bridge row.*

Proof. Start with the directed handoff boundary. Apply Definition 126.73 to the common directed finite-window surface. It says that the RegSeqRat tail visible to a downstream RealUp consumer is already selected by the StreamName$^{\uparrow}$ finite rows, the displayed Cont/hsame ledger, and the rational-history classifier.

Next apply Definition 126.74. The Real$^{\uparrow}$ source seen at any common finite subwindow is therefore the single constant rational StreamName row determined by RatHistoryClassifier(a, b) and the shared probe ledger.

Use Definition 38.206 to identify the selected spine as the reindexed constant stream spine, and then use Definition 38.207 to package the seal classifier carried by that spine.

Finally invoke Definition 126.75. It identifies the only RealUp observation classifier available to the two constant directed seals. Since the consumer preserves the displayed source and the selected spine, its carrier coordinates are exactly the finite StreamName, RegSeqRat, unary-observation, probe-ledger, hsame, Cont, and Pkg coordinates already named above. There is therefore no remaining coordinate in which a second RealUp source row, hidden selector, quotient stream equality, completion witness, arithmetic or order package, or standard bridge row could enter. □

126.4 StreamName constant-window transport totality

Definition 126.77 (StreamName constant-window carrier)**.** *The StreamName constant-window carrier is the finite-window packet over a point carrier p :* Hist *and a finite request history W :* ProbeBundle*. Its rows are the window index rows of W, the selected observation row identifying each requested probe with the constant point p, the componentwise* hsame *comparisons between any two displayed constant-window presentations, the displayed* Cont *replay row for the window read, the* Pkg *provenance row recording the request history, and the local* NameCert$_{\text{StreamName}^{\uparrow}}$ *row naming the finite-window surface.*

The carrier is local to the finite request. It contains no quotient stream coordinate and no row comparing host streams outside the displayed window.

Theorem 126.78 (StreamName window transport totality)**.** *Every constant-window read consumed by* RegSeqRat$^{\uparrow}$ *or by* Real$^{\uparrow}$ *passes through the StreamName constant-window carrier of Definition 126.77. The exported consumer surface is exactly the point carrier row, the request-history ledger, the window-index rows, the selected observation rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and the local* NameCert$_{\text{StreamName}^{\uparrow}}$ *row.*

The transport exports no quotient stream equality, selected limit, host sequence equality, or countable-choice selector.

Proof. First read the constant window through Definition 126.67. A Real-facing consumer obtains only the rational point row and the finite probe ledger carried by the public StreamName window.

Next read the same window through Definition 126.68. A RegSeqRat-facing consumer obtains the same constant point row and finite ledger, while regularity and seal data remain outside the constant-window packet.

Apply Definition 126.81. It identifies the common handoff to RegSeqRat and RealUp as the displayed finite-window surface rather than a host stream or completion object.

The remaining coordinates are exactly the rows listed in Definition 126.77. Since the cited boundaries contain no quotient stream equality, selected limit, host sequence equality, or choice selector, the transport cannot export one. □

126.5 StreamName Real and RegSeqRat handoff

Theorem 126.79 (StreamName finite-window handoff to RegSeqRat and RealUp)**.** *Let $\mathcal{W}$ be a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package over* Hist *whose selected observations are certified by a finite* ProbeBundle *ledger and by* InBundle *membership rows. Then the rows handed to* RegSeqRat *or finite-window* $\mathsf{Real}^{\uparrow}$ *consumers are exactly the selected* RatHistoryClassifier *rows, the finite* ProbeBundle *ledger, the* InBundle *membership rows, and the* UnaryHistory *rows naming the observed histories.*

No RegSeqRat *regularity row, no* $\mathsf{Real}^{\uparrow}$ *seal classifier, no completion object, and no host equality row is read from $\mathcal{W}$.*

Proof. Project the public StreamName boundary through Definition 126.301. That theorem lists the carrier, selected classifier, probe-bundle, membership, and unary-history rows available at the public finite-window interface.

For a finite-window $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 38.314 and Definition 38.320. These RealUp rows consume the displayed finite StreamName packet and its ledger, not a completion object or a hidden host stream.

For a RegSeqRat consumer, apply Definition 38.251. The seal reads the selected finite-window classifier and bundle rows as inputs to the downstream regularity and seal surface. Since the StreamName interface theorem lists no regularity row and no seal classifier, those obligations remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier real regseq handoff`

Theorem 126.80 (StreamName singleton RegSeqRat RealUp handoff exactness)**.** *Let s, t : Hist $\to$ Hist be StreamName streams and let n : Hist be a certified unary observation. If* RatStreamNameClassifier(s, t) *holds and n is a* UnaryHistory *row, then the singleton packet* ProbeBundle.Bcons n ProbeBundle.Bnil *supplies the finite-window StreamName classifier consumed by the* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$ *handoff.*

For that singleton packet, the exported consumer surface is exactly the selected rational classifier row RatHistoryClassifier$(s(n), t(n))$*, the singleton* ProbeBundle *row, the corresponding* InBundle *ledger row, the displayed* UnaryHistory *row for n, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support already present on the public StreamName side.*

The packet exports no RegSeqRat *regularity row, no* $\mathsf{Real}^{\uparrow}$ *completeness row, no quotient stream, no host sequence equality row, and no selected limit.*

Proof. Use the singleton diagonal handoff shape for rational StreamName classifiers. It turns the pointwise hypothesis RatStreamNameClassifier(s, t) and the certified unary row for n into a RatStreamNameFiniteWindowClassifier over ProbeBundle.Bcons n ProbeBundle.Bnil, with membership given by the singleton InBundle constructor.

The same shape identifies the only selected classifier row: every member of the singleton packet is judgmentally the displayed observation n, so the finite-window classifier reads the single row $\mathsf{RatHistoryClassifier}(s(n), t(n))$ and no second rational-history coordinate.

Now apply Definition 126.79 to this finite-window package. That theorem lists exactly the rows visible to RegSeqRat and finite-window $\mathsf{Real}^{\uparrow}$ consumers: selected rational classifier data, the finite probe ledger, membership rows, unary-history rows, and the displayed StreamName support rows.

Since the singleton construction supplies only those rows, all regularity, completion, quotient, host-equality, and limit data remain outside the handoff. □

Theorem 126.81 (StreamName constant-window handoff boundary)**.** *Let $a, b : \mathsf{Hist}$ be rational-history endpoints and let $\mathcal{W}$ be a constant rational* $\mathsf{StreamName}^{\uparrow}$ *finite-window package over an inhabited finite unary window. The package may seed the* RegSeqRat *stream side only by exporting the same* $\mathsf{RatHistoryClassifier}(a, b)$ *row together with its finite* ProbeBundle, InBundle, *and* UnaryHistory *ledger. A* RegSeqRat *certificate must still supply regularity rows, and a sealed* $\mathsf{Real}^{\uparrow}$ *candidate must still supply the seal classifier row.*

Proof. Apply Definition 126.67. It says that a finite-window $\mathsf{Real}^{\uparrow}$ consumer of the constant package reads only the rational point classifier and the finite probe ledger.

Apply Definition 126.68. It gives the same boundary for RegSeqRat: the constant StreamName window supplies rational point-classifier and probe-ledger data, while the regularity rows remain RegSeqRat obligations.

Finally use Definition 38.198 and Definition 38.204. These theorems place regularity and seal-interface rows on the RegSeqRat and RealUp side of the boundary, so the constant window is only a seed surface for the displayed finite rows. □

Theorem 126.82 (StreamName RegSeqRat RealUp obligation separation)**.** *Across the StreamName, RegSeqRat, and RealUp handoff, the obligation surface separates into three disjoint sources. The* $\mathsf{StreamName}^{\uparrow}$ *package contributes the stream carrier, selected rational-history classifier, finite* ProbeBundle *ledger,* InBundle *membership, and* UnaryHistory *observation rows. The* RegSeqRat *certificate contributes regularity, dyadic-tolerance, transport, and stream-spine rows. The* $\mathsf{Real}^{\uparrow}$ *seal contributes the seal classifier required by the real classifier specification.*

No finite StreamName window contributes a quotient, a completion witness, a host index equality, a RegSeqRat regularity witness, or the RealUp seal classifier.

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealRegseqObligationSeparation`

Proof. Read the StreamName part first. By Definition 126.301, and in the constant case by Definition 126.67 and Definition 126.68, the StreamName surface exposes exactly the carrier, classifier, bundle, membership, unary, and finite ledger rows.

Read the RegSeqRat part next. The regularity and dyadic-tolerance rows are listed by Definition 38.198, while Definition 38.251 describes how the selected finite-window rows are consumed at the seal boundary.

Read the RealUp part last. The seal classifier is supplied by Definition 38.204 and is consumed through Definition 38.314. Thus every downstream row is accounted for by exactly one side of the handoff, and no completion or host equality row is imported from the finite StreamName package. □

Theorem 126.83 (StreamName-RegSeqRat-Real seal composition)**.** *Let $\mathcal{W}$ be a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package whose selected finite rows are certified by the displayed* RatHistoryClassifier, ProbeBundle, InBundle, UnaryHistory, hsame, *finite* Cont, *and* Pkg *provenance data. Let S be a*

$\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal that consumes exactly those selected rational-history rows together with its regularity, dyadic-tolerance, transport, stream-spine, and seal-interface ledger rows. Let R be a finite-window* $\mathsf{Real}^{\uparrow}$ *ledger-boundary row that consumes the seal S.*

Then every Real-facing consumer row of R factors through exactly the selected finite rows of $\mathcal{W}$, the RegSeqRat *regularity, dyadic-tolerance, transport, stream-spine, and seal rows of S, and the* $\mathsf{Real}^{\uparrow}$ *seal-classifier ledger row. Conversely, those three displayed surfaces repack into the same Real-facing finite-window ledger boundary.*

The composed route exports no quotient stream, selected host limit, host equality, completion witness, countable-choice schedule, or ambient completeness coordinate.

Proof. First isolate the source rows. Apply Definition 126.79. It says that the public StreamName window contributes only the selected rational-history classifier rows, the finite probe ledger, bundle-membership rows, unary observations, finite transport data, and provenance rows.

Next seal those rows on the RegSeqRat side. Use Definition 38.251 to read the selected finite classifier rows as the input surface of S, and use Definition 38.204 to place the regularity, dyadic-tolerance, transport, stream-spine, and seal-interface rows on the RegSeqRat ledger side of the boundary.

Finally pass to the Real-facing row. Apply Definition 38.314 and Definition 38.320. These theorems consume the RegSeqRat seal as a finite ledger boundary and expose only the seal-classifier row required by the RealUp consumer.

The preceding three steps give the forward factorization. The converse repacking follows by reading the same cited theorems in order: the StreamName finite rows supply the source packet, the RegSeqRat exactness and interface rows rebuild the seal packet, and the RealUp boundary theorem rebuilds the finite-window ledger row. Since each cited boundary excludes quotient streams, host equalities, completion witnesses, countable-choice schedules, and ambient completeness coordinates, the composed route exports none of them. □

Theorem 126.84 (StreamName-RegSeqRat-Real finite-window seal pullback)**.** *Let $\mathcal{W}$ be a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window package whose selected rational classifier rows are consumed both by a* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal S and by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal boundary R. Assume the two consumers read the same displayed finite rows of $\mathcal{W}$: the* ProbeBundle *ledger,* InBundle *membership rows,* UnaryHistory *observations, selected* RatHistoryClassifier *rows, displayed* hsame *comparisons, finite* Cont *routes,* Pkg *provenance, and* NameCert *support.*

Then the common consumer surface of $\mathcal{W}$, S, and R is exactly that finite displayed surface. Equivalently, the $\mathsf{RegSeqRat}^{\uparrow}$ *seal and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal pull back along the selected* $\mathsf{StreamName}^{\uparrow}$ *window to the single cover-indexed boundary carried by those* ProbeBundle*,* InBundle*,* UnaryHistory*,* RatHistoryClassifier*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The pullback boundary introduces no quotient stream, host index equality, selected limit, ambient completion, or additional $\mathsf{Real}^{\uparrow}$ *seal classifier outside the displayed finite-window ledger.*

Proof. First apply Definition 126.79. It identifies the only rows exported by the public StreamName finite window to RegSeqRat and RealUp consumers: the selected rational classifier rows, probe-bundle ledger, membership rows, unary observations, finite transport rows, provenance, and displayed public support. Thus both consumers start from the same finite StreamName surface.

Next use Definition 126.82. The separation places carrier, classifier, bundle, membership, and unary-history data on the StreamName side; regularity, dyadic-tolerance, transport, and stream-spine data on the RegSeqRat side; and the seal-classifier obligation on the RealUp side. Hence no obligation is counted twice, and no quotient, host equality, limit, or completion row can enter through the shared source.

Finally apply Definition 126.83. The composition theorem repacks the StreamName finite rows, the RegSeqRat seal rows, and the RealUp seal boundary into one Real-facing finite-window ledger. Since the two consumers were assumed to read the same displayed finite rows, this cover-indexed ledger is the required pullback boundary, with no additional RealUp seal classifier outside the displayed surface. □

Definition 126.85 (StreamName-RegSeqRat scheduled seal package)**.** *A* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$ *scheduled seal package is the scheduled constant observation row together with three finite-window boundary rows:*

1. *the public finite-window* $\mathsf{StreamName}^{\uparrow}$ *row over* BHist *histories, its finite* $\mathsf{ProbeBundle}$ *ledger,* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *observation rows, and the displayed* hsame *comparisons between scheduled endpoints;*
2. *the* $\mathsf{RegSeqRat}$ *finite-window seal row reading the same selected* $\mathsf{RatHistoryClassifier}$ *rows and dyadic ledger rows;*
3. *the finite-window* $\mathsf{Real}^{\uparrow}$ *ledger-boundary row that consumes only the selected rational classifier rows, the probe ledger, and the unary observation rows.*

The package has no further carrier. Its consumer-facing surface is exactly the scheduled observation, finite-window StreamName row, RegSeqRat seal row, and RealUp ledger-boundary row over the displayed BHist, $\mathsf{ProbeBundle}$, $\mathsf{InBundle}$, $\mathsf{UnaryHistory}$, hsame, $\mathsf{Rat}^{\uparrow}$ *classifier, and dyadic ledger data.*

Theorem 126.86 (StreamName-RegSeqRat scheduled seal exhaustion)**.** *Every consumer row of the scheduled seal package of Definition 126.85 is projected from exactly one of the following sources: the scheduled constant observation exactness row, the scheduled window reindex row, the* $\mathsf{RegSeqRat}$ *finite-window seal exactness row, or the finite-window* $\mathsf{Real}^{\uparrow}$ *ledger-boundary row. Conversely, those four source rows repack into the scheduled seal package and determine its whole consumer-facing surface.*

No consumer row of the package exports completion data, quotient data, host index equality, a $\mathsf{RegSeqRat}$ *regularity witness beyond the finite-window seal row, or a* $\mathsf{Real}^{\uparrow}$ *seal classifier outside the displayed ledger boundary.*

Proof. First project the scheduled StreamName source. Apply Definition 126.286 to obtain the constant observation row and Definition 126.287 to identify its finite scheduled windows under reindexing.

Next project the downstream seal rows. The finite-window feed theorem Definition 126.290 places the selected $\mathsf{Rat}^{\uparrow}$ classifier rows, probe-bundle rows, membership rows, unary observation rows, and dyadic ledger rows at the common handoff. The RegSeqRat seal scope theorem Definition 126.291 says that the RegSeqRat consumer reads precisely the finite-window seal row from that handoff.

The separation theorems in this section identify the remaining boundary. Definition 126.79 gives the public finite-window StreamName row, Definition 126.81 specializes it to the constant scheduled source, and Definition 126.82 keeps the RegSeqRat and RealUp obligations disjoint.

For the reverse direction, repack the scheduled observation exactness row, the reindex row, the RegSeqRat finite-window seal exactness row, and the RealUp finite-window ledger boundary into the three displayed components of Definition 126.85. The same cited handoff and separation theorems exclude every row outside those components, so the repacked package is exhaustive. □

Theorem 126.87 (StreamName Real-RegSeq finite-overlap source stability)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be public* StreamName$^{\uparrow}$ *finite-window packages whose observation windows restrict to a certified common finite overlap K. Suppose the two restrictions expose the same* RatHistoryClassifier *rows, the same finite* ProbeBundle *ledger, the same* InBundle *membership rows, the same* UnaryHistory *observation rows, and the same displayed* hsame *comparisons on K. Then any* RegSeqRat *handoff package, and any finite-window* Real$^{\uparrow}$ *handoff package, induced by the two restrictions has the same source surface on K.*

The common source surface is only the restricted StreamName$^{\uparrow}$ *finite-window surface over those rows. It excludes every* RegSeqRat *regularity row, the* Real$^{\uparrow}$ *seal classifier, completion data, quotient data, and host index equality.*

Proof. First restrict both public StreamName packages to K. By Definition 126.79, each restricted handoff exposes only the selected rational-history classifier, finite ProbeBundle ledger, InBundle rows, and UnaryHistory rows.

The assumed overlap equality identifies exactly those rows through the displayed hsame comparisons. In the constant source case, Definition 126.69 gives the same identification from the shared constant rational endpoint row; for a general finite RealUp overlap, the overlap boundary is the one recorded by Definition 38.326.

Now apply Definition 126.82. The RegSeqRat regularity rows and the RealUp seal classifier are downstream obligations, while completion data, quotient data, and host index equality are not exported by the finite StreamName side.

Repacking the two restricted inputs therefore gives the same source handoff surface on K for either downstream consumer, with all non-source rows kept outside the overlap package. □

Theorem 126.88 (StreamName Real-RegSeq overlap unique source)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be public* StreamName$^{\uparrow}$ *finite-window packages whose restrictions to a certified finite overlap K expose the same* RatHistoryClassifier *rows, the same* ProbeBundle *ledger, the same* InBundle *membership rows, the same* UnaryHistory *observation rows, and the same displayed* hsame *rows. Then the overlap determines one StreamName source surface: every* RegSeqRat *handoff package induced from either restriction, and every finite-window* Real$^{\uparrow}$ *handoff package induced from either restriction, has that same source surface on K.*

This source surface consists only of the public finite-window StreamName$^{\uparrow}$ *rows displayed on the certified overlap. It does not include* RegSeqRat *regularity rows,* Real$^{\uparrow}$ *seal classifiers, completion data, quotient data, or host index equality.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealRegseqOverlapUniqueSource`

Proof. First restrict both public finite-window packages to the common overlap K. Definition 126.79 identifies the source-facing rows of each restriction as the rational-history classifier, probe-bundle ledger, membership rows, unary-history rows, and displayed hsame comparisons.

The hypotheses say precisely that those rows agree on K. Definition 126.87 then turns the shared restricted rows into the same source surface for every induced RegSeqRat handoff and every induced finite-window Real$^{\uparrow}$ handoff.

It remains to check that no downstream or host row is being folded into the source. Apply Definition 126.82: RegSeqRat regularity rows and RealUp seal classifiers are downstream obligations rather than StreamName source rows. The same separation keeps completion data, quotient data, and host index equality outside the handoff source.

Finally, Definition 38.326 supplies the RealUp finite-window overlap boundary used by the consumer side. Since the RealUp boundary reads the same certified overlap and the StreamName

source has already been fixed there, both induced consumer packages inherit one source handoff surface on K. □

Theorem 126.89 (StreamName RegSeqRat pointwise seal stability)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packages whose restrictions to a certified finite overlap K agree on every certified unary observation, every selected* RatHistoryClassifier *row, every* ProbeBundle *ledger row, every* InBundle *membership row, and every displayed* hsame *row. Suppose the two induced* RegSeqRat *handoff packages consume the same DyadicRatCore finite ledger over that overlap. Then the* $\mathsf{Real}^{\uparrow}$ *seal classifier rows induced from those two handoffs agree pointwise on K.*

The source used by this pointwise seal comparison is only the public $\mathsf{StreamName}^{\uparrow}$ *finite-window surface together with the consumed* RegSeqRat *dyadic ledger. The comparison does not use completion data, quotient data, host index equality, or a* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameRegseqratPointwiseSealStability`

Proof. First fix the source surface. Apply Definition 126.88 to the two restricted public StreamName packages. The hypotheses are exactly the shared finite-window rows on K, so both induced RegSeqRat handoffs and both finite-window $\mathsf{Real}^{\uparrow}$ handoffs read one source surface there.

Next fix the seal ledger. By Definition 38.262, the RegSeqRat seal reads the refined finite dyadic ledger: carried RatUp endpoints, positive unary denominators, common-radius rows, hsame and Cont provenance, and the scoped DyadicRatCore boundary. Since the two handoffs consume the same DyadicRatCore ledger, the seal input rows coincide pointwise on the overlap.

Now transport the public observation rows along the displayed comparisons. Definition 38.263 says that each hsame-transported DyadicRatCore public observation remains the same $\mathsf{Real}^{\uparrow}$ finite-window input, with the same finite ledger and RatUp endpoint faces.

Finally use classifier stability and transitivity along the displayed hsame rows. The StreamName finite-window classifiers agree on K, the dyadic seal ledger agrees on K, and the transported RealUp finite-window inputs are the same rows. Hence the induced $\mathsf{Real}^{\uparrow}$ seal classifier rows are pointwise identical on the common overlap, without importing any completion, quotient, host-index, or RealUp completeness datum. □

Theorem 126.90 (StreamName RegSeqRat pointwise seal pasting)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be certified finite* $\mathsf{StreamName}^{\uparrow}$ *windows with a certified finite overlap K. Suppose their restrictions to K agree on every certified unary observation, every selected* RatHistoryClassifier *row, every* ProbeBundle *ledger row, every* InBundle *membership row, and every displayed* hsame *row. Suppose also that the induced* RegSeqRat *handoff packages consume the same DyadicRatCore finite ledger on K. Then the* $\mathsf{Real}^{\uparrow}$ *seal classifier rows induced from the two handoffs paste to a single finite-window source surface on $\mathcal{W}_0 \cup \mathcal{W}_1$.*

The pasted surface exports only RatHistoryClassifier *rows,* ProbeBundle *ledger rows,* InBundle *membership rows,* UnaryHistory *observations,* hsame *comparisons, and the shared dyadic ledger. It exports no completion datum, quotient datum, host index equality, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First identify the common source on the overlap. Apply Definition 126.88 to the two restricted certified windows. The displayed agreement rows are exactly the finite StreamName source rows, so the overlap has one source-facing surface for both induced RegSeqRat handoffs.

Next stabilize the seal rows over that common source. By Definition 126.89, the shared DyadicRatCore finite ledger forces the $\mathsf{Real}^{\uparrow}$ seal classifier rows induced from the two handoffs to agree pointwise on every member of K.

It remains to build the union surface. Use finite ProbeBundle induction over the selected rows of $\mathcal{W}_0$ and $\mathcal{W}_1$: rows outside the overlap are inserted from their own certified window, while rows inside K are identified by the pointwise seal stability just obtained.

Finally exhaust the pasted rows. Apply Definition 126.100: every row in the handoff surface is one of the displayed pointwise RatHistoryClassifier observations, bundle and membership rows, unary observations, hsame comparisons, or dyadic ledger rows. Hence the finite union surface contains no completion, quotient, host-index, or RealUp-completeness datum. □

Theorem 126.91 (StreamName RegSeqRat pasted seal associativity)**.** *Let $\mathcal{W}_0, \mathcal{W}_1, \mathcal{W}_2$ be certified finite* StreamName$^{\uparrow}$ *windows. Suppose each adjacent pair carries a certified finite overlap, the restrictions on those overlaps agree on the certified unary observations, selected* RatHistoryClassifier *rows,* ProbeBundle *ledger rows,* InBundle *membership rows, and displayed* hsame *rows, and the three induced* RegSeqRat *handoff packages consume one shared* DyadicRatCore$^{\uparrow}$ *finite ledger on every displayed overlap. Then the two iterated pastings $(\mathcal{W}_0 \cup \mathcal{W}_1) \cup \mathcal{W}_2$ and $\mathcal{W}_0 \cup (\mathcal{W}_1 \cup \mathcal{W}_2)$ induce the same* Real$^{\uparrow}$ *seal classifier surface.*

The common pasted surface is only the finite StreamName source surface, the shared DyadicRatCore ledger, and the pointwise RegSeqRat *handoff rows exported by the three windows. It contains no completion data, quotient data, host index equality, or* Real$^{\uparrow}$ *completeness row.*

Proof. First fix the overlap sources. Apply Definition 126.88 to each certified pairwise overlap. Each overlap therefore has one StreamName source surface for the induced RegSeqRat and finite-window Real$^{\uparrow}$ handoffs.

Next stabilize the pairwise seal rows. By Definition 126.89, the shared DyadicRatCore finite ledger forces the induced Real$^{\uparrow}$ seal classifier rows to agree pointwise on every displayed overlap. These pointwise agreements are the only identifications used by either iterated pasting.

Now paste the left-associated surface. Use Definition 126.90 first on $\mathcal{W}_0$ and $\mathcal{W}_1$, then on the resulting finite union and $\mathcal{W}_2$. The second application reads the same overlap sources and the same shared dyadic ledger already fixed above.

Paste the right-associated surface in the symmetric order. The same theorem first pastes $\mathcal{W}_1$ with $\mathcal{W}_2$ and then pastes $\mathcal{W}_0$ against that finite union. Since the source surfaces and pointwise seal comparisons are the same rows, both constructions classify each member of the three-window union by the same Real$^{\uparrow}$ seal row.

Finally exhaust both iterated surfaces with Definition 126.100. Every row in either surface is one of the displayed StreamName finite-window rows, an hsame comparison, or the shared dyadic ledger row. Completion data, quotient data, host index equality, and RealUp completeness rows have no carrier coordinate in either association, so the two pasted classifier surfaces are identical. □

Theorem 126.92 (StreamName RegSeqRat Real triple-overlap pasting)**.** *Let $\mathcal{W}_0, \mathcal{W}_1, \mathcal{W}_2$ be certified finite* StreamName$^{\uparrow}$ *windows. Suppose each pair carries a certified finite overlap, and suppose the two restrictions on every displayed overlap agree on every certified unary observation, every selected* RatHistoryClassifier *row, every* ProbeBundle *ledger row, every* InBundle *membership row, every displayed* hsame *row, and every consumed* DyadicRatCore$^{\uparrow}$ *finite ledger row. Then the three induced* RegSeqRat$^{\uparrow}$ *handoff packages and the three finite-window* Real$^{\uparrow}$ *seal classifier rows paste to one finite three-window source surface.*

The exported surface is exactly the union of the three finite-window StreamName$^{\uparrow}$ *row sets, the shared dyadic ledgers on their overlaps, and the pointwise* RegSeqRat$^{\uparrow}$ *handoff rows read from those ledgers. It exports no quotient stream, host index equality, selected limit,* Real$^{\uparrow}$ *completeness row, metric-completeness datum, or bridge row.*

Proof. First fix the three pairwise sources. Apply Definition 126.88 to $(\mathcal{W}_0, \mathcal{W}_1)$, to $(\mathcal{W}_1, \mathcal{W}_2)$, and to $(\mathcal{W}_0, \mathcal{W}_2)$. The hypotheses provide exactly the displayed unary observations, classifier rows, bundle ledgers, membership rows, and hsame comparisons required on each overlap.

Next stabilize the seal classifiers on the three overlaps. By Definition 126.89, the shared consumed $\mathsf{DyadicRatCore}^{\uparrow}$ finite ledgers force the $\mathsf{Real}^{\uparrow}$ seal classifier rows induced from either side of each overlap to agree pointwise. These agreements are the only identifications used by the triple source surface.

Now form the pairwise pasted faces. Apply Definition 126.90 to each of the three pairwise unions. Each application exports only finite StreamName rows, the shared dyadic ledger rows, and the pointwise RegSeqRat handoff rows, so no quotient, host-index, selected-limit, completeness, metric, or bridge datum is introduced on a face.

It remains to identify the two possible ways of reading the three faces as a single surface. Use Definition 126.91. The left and right iterated pastings classify each member of $\mathcal{W}_0 \cup \mathcal{W}_1 \cup \mathcal{W}_2$ by the same $\mathsf{Real}^{\uparrow}$ seal row because they read the same pairwise source rows and the same dyadic ledger rows.

Finally exhaust the resulting surface. Apply Definition 126.100 on each window and on each pairwise pasted face, and use Definition 126.82 to keep downstream and ambient rows outside the source. Every exported row is therefore a row from one of the three StreamName finite windows, a shared dyadic ledger row, or a pointwise RegSeqRat handoff row. The listed quotient, host-index, selected-limit, completeness, metric-completeness, and bridge rows have no carrier coordinate in the triple source surface. □

Theorem 126.93 (StreamName RegSeqRat Real finite-cover descent exactness)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *rational windows. Suppose every pairwise certified overlap has the unique source surface of Definition 126.88, and suppose the triple overlaps satisfy the cocycle compatibility witnessed by Definition 126.92. Then the induced* $\mathsf{RegSeqRat}^{\uparrow}$ *handoffs and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal rows descend to one cover-indexed finite surface.*

The descended surface is assembled only from the displayed BMark *and* BHist *observations, selected* RatHistoryClassifier *rows,* ProbeBundle *ledger rows,* InBundle *membership rows, displayed* hsame *and* Cont *transports, consumed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows,* Pkg *provenance, local* NameCert *rows, induced* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and finite-window* $\mathsf{Real}^{\uparrow}$ *seal rows. Conversely, every row of that descended surface restricts on some certified window or certified overlap to exactly one local* $\mathsf{StreamName}^{\uparrow}$ *row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *dyadic-ledger row, or one finite-window* $\mathsf{Real}^{\uparrow}$ *seal row.*

No row of the construction is a quotient stream, host index equality, selected limit, countable-choice selector, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Proceed by finite induction on the cover index set I. For a singleton cover, the claim is exactly Definition 126.100: the visible surface is exhausted by the local StreamName rows, the consumed dyadic ledger, and the finite RealUp seal rows, with no quotient, host, choice, or completeness coordinate.

For the induction step, assume a descended finite surface has already been assembled over a finite subcover and adjoin one further certified window $\mathcal{W}$. On each certified overlap with the existing subcover, apply Definition 126.88 to identify the StreamName source rows, then apply Definition 126.90 to paste the induced $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows and finite-window $\mathsf{Real}^{\uparrow}$ seal rows across that common source.

The result is independent of the order in which the new overlap faces are adjoined. Indeed, whenever three windows meet, the two possible iterated pastings have the same seal classifier

surface by Definition 126.92. Thus the finite induction produces one cover-indexed surface rather than an order-indexed family of surfaces.

It remains to read the rows of the descended surface back to local data. Use Definition 126.82 to keep downstream regularity, seal, quotient, host, choice, and completeness data outside the StreamName source. Then use Definition 126.100 on every window and overlap face. Each exported row is therefore a local StreamName row, a consumed RegSeqRat dyadic-ledger row, or a finite-window RealUp seal row on a certified member of the cover, as required. □

Theorem 126.94 (StreamName RegSeqRat Real finite-cover descent support minimality)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *rational windows satisfying the hypotheses of Definition 126.93. For the descended finite surface produced there, none of the displayed support rows can be deleted while presenting the same descended surface.*

More precisely, deleting any local $\mathsf{StreamName}^{\uparrow}$ *row displayed on a cover member, any consumed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row on an overlap, any* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row induced from such a ledger, any finite-window* $\mathsf{Real}^{\uparrow}$ *seal row, or any shared* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *support row prevents the same cover-indexed finite surface from being presented. Quotient streams, host index equalities, selected limits, countable-choice selectors, metric-completeness rows, bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness rows cannot replace the deleted support.*

Proof. First restrict the alleged presentation of the same descended surface to a cover member containing the deleted local row. By Definition 126.93, every row of the descended surface restricts on that member or on one of its certified overlaps to a unique local $\mathsf{StreamName}^{\uparrow}$ row, consumed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row, induced $\mathsf{RegSeqRat}^{\uparrow}$ handoff row, or finite-window $\mathsf{Real}^{\uparrow}$ seal row. Thus the restricted presentation must recover exactly the deleted row if the descended surface is unchanged.

For a deleted overlap ledger or handoff row, restrict instead to an overlap face. The pairwise source is pinned by the unique-source hypotheses used in the descent theorem, and the compatibility of threefold restrictions is the one supplied by Definition 126.92. Hence two presentations of the same finite surface must read the same shared BHist, hsame, Cont, Pkg, and NameCert support rows on the overlap; omitting one of them changes the overlap face rather than merely its description.

Next apply Definition 126.100 on the affected window or overlap. The theorem exhausts the visible handoff by the StreamName source rows, the consumed dyadic ledger, and the finite seal rows, so a missing displayed row has no second source inside the handoff surface.

Finally use Definition 126.82. It keeps quotient streams, host index equalities, selected limits, countable-choice selectors, metric-completeness data, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the StreamName source and its finite-window RegSeqRat and Real faces. Those downstream or ambient rows therefore cannot substitute for the deleted support row, and the same descended finite surface cannot be presented after the deletion. □

Theorem 126.95 (StreamName RegSeqRat overlap-pasting consumer exhaustion)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be certified finite* $\mathsf{StreamName}^{\uparrow}$ *windows with certified finite overlap K. Suppose the hypotheses of Definition 126.90 hold: the two restrictions to K agree on the certified unary observations, selected* RatHistoryClassifier *rows,* ProbeBundle *ledger rows,* InBundle *membership rows, displayed* hsame *rows, and the induced* RegSeqRat *handoffs consume the same DyadicRatCore finite ledger on K. Then every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer row read from $\mathcal{W}_0 \cup \mathcal{W}_1$ is exactly one of the following rows: a row already read from $\mathcal{W}_0$, a row already read from $\mathcal{W}_1$, or an overlap row identified by Definition 126.89.*

Thus the exhausted consumer surface is precisely the finite inventory of RatHistoryClassifier *rows,* ProbeBundle *rows,* InBundle *rows,* UnaryHistory *rows,* hsame *comparisons, and shared DyadicRatCore ledger rows. It contains no completion datum, quotient datum, host-index equality, metric row, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. First paste the seal surface. Apply Definition 126.90. Its hypotheses are the displayed agreement rows on K together with the common DyadicRatCore finite ledger, so it yields one pasted finite-window source surface over $\mathcal{W}_0 \cup \mathcal{W}_1$.

Next compare the consumer rows on the overlap. By Definition 126.97, any finite-window $\mathsf{Real}^{\uparrow}$ consumer restricted to K reads the same BHist surface as the induced RegSeqRat handoff. The pointwise identifications on K are exactly those supplied by Definition 126.89.

Now run finite bundle induction over the two windows. A selected row outside K is inserted from the unique source window containing it. A selected row inside K is identified through the overlap comparison, so inserting it from either side gives the same pasted consumer row.

Finally apply Definition 126.100 to the pasted surface. It enumerates the only rows visible to RegSeqRat and finite-window $\mathsf{Real}^{\uparrow}$ consumers: rational-history classifier rows, bundle and membership rows, unary observations, hsame comparisons, and the shared dyadic ledger. The exclusion clause rules out completion, quotient, host-index, metric, and RealUp-completeness data, so the union consumer surface is exhausted exactly by the stated rows. □

Theorem 126.96 (StreamName RegSeqRat three-window seal coherence)**.** *Let* $\mathcal{W}_0$*,* $\mathcal{W}_1$*, and* $\mathcal{W}_2$ *be certified finite* $\mathsf{StreamName}^{\uparrow}$ *windows with certified finite pairwise overlaps. Suppose that on every displayed overlap the selected* RatHistoryClassifier *rows,* ProbeBundle *ledger rows,* InBundle *membership rows,* UnaryHistory *observations, displayed* hsame *comparisons, and DyadicRatCore finite ledger rows are compatible. Then the finite-window* $\mathsf{Real}^{\uparrow}$ *seal source surface obtained by pasting* $(\mathcal{W}_0 \cup \mathcal{W}_1) \cup \mathcal{W}_2$ *is the same source surface as the one obtained by pasting* $\mathcal{W}_0 \cup (\mathcal{W}_1 \cup \mathcal{W}_2)$*.*

The common surface is exhausted by the displayed finite StreamName, RegSeqRat*, and DyadicRatCore rows. No completion datum, quotient datum, host index equality, metric-completeness row, Classical principle, or ambient model datum enters the comparison.*

Proof. First compare the sources on each pairwise overlap. Apply Definition 126.88 to the certified restrictions of $\mathcal{W}_0$ and $\mathcal{W}_1$, then to the restrictions of $\mathcal{W}_1$ and $\mathcal{W}_2$, and then to the restrictions of $\mathcal{W}_0$ and $\mathcal{W}_2$. Each comparison is witnessed by the same displayed unary observations, selected classifier rows, bundle rows, membership rows, and hsame rows carried by the pairwise overlap.

Next stabilize the seal rows pointwise. Use Definition 126.89 on each overlap. The shared DyadicRatCore finite ledger rows force the induced $\mathsf{Real}^{\uparrow}$ seal rows to agree wherever two windows meet.

Now paste in the left-associated order. Apply Definition 126.90 to $\mathcal{W}_0$ and $\mathcal{W}_1$, and then apply the same theorem to the pasted surface and $\mathcal{W}_2$. The second application uses the pairwise overlap identifications already obtained to identify every row in $(\mathcal{W}_0 \cup \mathcal{W}_1) \cap \mathcal{W}_2$.

Paste in the right-associated order by the symmetric construction. The same pasting theorem first combines $\mathcal{W}_1$ with $\mathcal{W}_2$ and then combines $\mathcal{W}_0$ with that surface. Because every overlap row has already been identified by source uniqueness and pointwise seal stability, no new row can distinguish the two bracketings.

Finally exhaust both pasted surfaces. Apply Definition 126.100 to the left-associated surface and again to the right-associated surface. Both surfaces contain exactly the displayed finite rows listed in the theorem and therefore present the same source-facing $\mathsf{Real}^{\uparrow}$ seal surface. □

Theorem 126.97 (StreamName Real-RegSeq common-window consumer determinacy)**.** *Let $\mathcal{W}_0$ and $\mathcal{W}_1$ be public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packages restricting to a certified finite overlap K. Suppose that the restrictions agree on every pointwise* RatStreamNameCarrier *observation, every pointwise* RatStreamNameClassifier *observation, the finite* ProbeBundle *ledger, the* InBundle *membership rows, the certified* UnaryHistory *observations, the displayed* hsame *rows, and the consumed DyadicRatCore finite ledger over K.*

Then every induced RegSeqRat *handoff and every finite-window* $\mathsf{Real}^{\uparrow}$ *handoff determined by those packages has the same consumer-facing finite* BHist *surface on K. That surface is exhausted by the public* $\mathsf{StreamName}^{\uparrow}$ *rows together with the* RegSeqRat *and DyadicRatCore finite rows named above, and it contains no quotient datum, completion datum, host index equality, metric-completeness row, Classical datum, or standard-bridge datum.*

Proof. First identify the source. Apply Definition 126.88 to the two public finite-window restrictions on K. The hypotheses list the same pointwise carrier rows, classifier rows, bundle ledger, membership rows, unary observations, and displayed hsame comparisons, so both packages expose one source-facing StreamName surface on the overlap.

Next stabilize the seal classifier rows. Apply Definition 126.89. The common DyadicRatCore finite ledger is exactly the downstream input used by the induced RegSeqRat handoffs, so the finite-window $\mathsf{Real}^{\uparrow}$ seal rows agree pointwise over the same overlap.

Finally exhaust the consumer surface with Definition 126.100. Every row a RegSeqRat or finite-window $\mathsf{Real}^{\uparrow}$ consumer can read is one of the public StreamName point rows, the bundle and membership rows, the unary observation rows, the displayed hsame rows, or the shared DyadicRatCore ledger rows. The exclusion clause of that theorem rules out any remaining quotient, completion, host-index, metric-completeness, Classical, or standard-bridge datum. □

Theorem 126.98 (StreamName RegSeqRat visible-row exhaustion)**.** *Every finite-window row handed from* $\mathsf{StreamName}^{\uparrow}$ *to a* RegSeqRat *consumer or to a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer is exactly one of the pointwise* RatStreamNameCarrier *and* RatStreamNameClassifier *observations at certified* UnaryHistory *indices, one of the displayed componentwise* hsame *rows, one of the consumed* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger rows over the same finite window, one of the finite* ProbeBundle *ledger rows, or one of the* InBundle *membership rows naming the selected observations. Conversely, these pointwise carrier/classifier rows, componentwise comparison rows, dyadic finite ledger rows, bundle ledger rows, and membership rows repack into the same consumer-facing handoff surface.*

Proof. Project the finite-window handoff with Definition 126.79. The projection lists exactly the selected rational-history classifier rows, the finite ProbeBundle ledger, the InBundle membership rows, the certified UnaryHistory observation rows, the displayed hsame comparisons, and the consumed DyadicRatCore finite ledger over the same finite window.

Identify the source rows pointwise using Definition 126.1 and Definition 126.2. The restriction and exhaustion steps are supplied by Definition 126.22 and Definition 126.23, so the finite-window rows are exactly the certified unary-index observations and their selected bundle support.

Finally use Definition 126.97. It keeps the consumed DyadicRatCore rows and the displayed hsame rows on the same public window. Repacking the listed rows through the same finite-window handoff theorem returns the original consumer-facing surface. □

Theorem 126.99 (StreamName RegSeqRat no-extra-data exclusion)**.** *The finite-window handoff surface from* $\mathsf{StreamName}^{\uparrow}$ *to a* RegSeqRat *consumer or to a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer contains no host index equality, quotient datum, completion datum, metric-completeness row, Classical*

datum, standard-bridge row, selected limit, or ambient Real$^{\uparrow}$ *completeness row.* RegSeqRat *regularity and the* Real$^{\uparrow}$ *seal classifier remain downstream obligations, while the DyadicRatCore ledger, displayed* hsame *rows, finite bundle ledger, and membership rows are the visible finite handoff data.*

Proof. Start from Definition 126.82. It separates the source-facing StreamName rows from downstream RegSeqRat regularity and finite-window Real$^{\uparrow}$ seal obligations, so neither downstream obligation can be read as hidden StreamName source data.

Use Definition 126.88 on the same finite window. The unique-source clause rules out alternate host-index, quotient, or completion coordinates for a surface with the displayed pointwise rows, comparisons, bundle rows, membership rows, and finite ledger.

The finite-window handoff theorem Definition 126.79 supplies only the listed public rows, and Definition 126.97 keeps the consumed DyadicRatCore ledger on that window. No metric-completeness row, Classical datum, bridge row, selected limit, or ambient Real$^{\uparrow}$ completeness row is introduced by these finite constructors. □

Theorem 126.100 (StreamName RegSeqRat pointwise handoff exhaustion)**.** *Every finite-window row handed from* StreamName$^{\uparrow}$ *to a* RegSeqRat *consumer or to a finite-window* Real$^{\uparrow}$ *consumer is exhausted by the pointwise* RatStreamNameCarrier *and* RatStreamNameClassifier *observations at certified* UnaryHistory *indices, the displayed componentwise* hsame *rows, the consumed DyadicRatCore finite ledger rows over the same finite window, the finite* ProbeBundle *ledger, and the* InBundle *membership rows naming the selected observations. Conversely, those pointwise observations, displayed comparison rows, dyadic ledger rows, and finite bundle rows repack into the same handoff surface.*

The StreamName side contributes no host index equality, quotient data, completion data, metric-completeness row, Classical datum, or standard-bridge datum. RegSeqRat *regularity and the* Real$^{\uparrow}$ *seal classifier remain downstream obligations, while the DyadicRatCore ledger and displayed* hsame *rows are part of the visible finite handoff surface.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRegSeqRatPointwiseHandoffExhaustion`

Proof. Apply Definition 126.98. Its proof projects the handoff through Definition 126.79, then identifies the pointwise carrier and classifier rows by Definition 126.1, Definition 126.2, Definition 126.22, and Definition 126.23.

The same visible-row theorem uses Definition 126.97 to keep the displayed hsame comparisons and consumed DyadicRatCore ledger on the same finite window. Thus the listed rows exhaust the consumer-facing surface and repack into that surface.

Apply Definition 126.99. Its proof uses Definition 126.82 and Definition 126.88 to keep downstream regularity, the finite-window Real$^{\uparrow}$ seal, host-index equality, quotient data, completion data, metric-completeness rows, Classical data, standard-bridge rows, selected limits, and ambient Real$^{\uparrow}$ completeness outside the source handoff. □

Theorem 126.101 (StreamName RegSeqRat nested-window handoff)**.** *Let $K \subseteq J \subseteq L$ be certified finite windows of a public* StreamName$^{\uparrow}$ *rational stream package. For any* RegSeqRat *consumer, and for any finite-window* Real$^{\uparrow}$ *consumer, the source surface obtained by first restricting the handoff from L to J and then from J to K is the same source surface as the handoff obtained by restricting directly from L to K.*

The common source surface consists only of the pointwise RatStreamNameCarrier *and* RatStreamNameClassifier *observations at certified* UnaryHistory *indices, the finite* ProbeBundle *ledger, the* InBundle *membership rows, and the displayed* hsame *comparisons. It contains no* RegSeqRat *regularity datum,* Real$^{\uparrow}$ *seal datum, completion datum, quotient datum, or host index equality.*

Proof. First restrict the outer window. By Definition 126.79, the handoff from L to J exposes only the selected rational-stream point rows, the finite bundle ledger, the membership rows, and the displayed hsame comparisons. Applying the same theorem again to the certified subwindow $K \subseteq J$ projects the two-step handoff to that same kind of surface over K.

Now compare the direct route. Applying Definition 126.79 directly to $K \subseteq L$ gives a handoff surface whose visible rows are again exactly the certified pointwise carrier and classifier observations, the finite ProbeBundle ledger, the InBundle membership rows, and the displayed hsame comparisons over K.

Use Definition 126.88 first on the overlap J and then on the overlap K. The hypotheses are the same certified point rows and the same finite ledger rows inherited from L, so the two restriction routes cannot produce different StreamName source surfaces on either overlap.

Finally apply Definition 126.100. It collapses both restricted handoffs to the same pointwise RatStreamNameCarrier and RatStreamNameClassifier rows together with the same finite ledger. The exclusion clauses in that exhaustion theorem and in the finite-window handoff theorem keep regularity, seal, completion, quotient, and host-index data outside the common surface. □

Theorem 126.102 (StreamName RegSeqRat nested-window seal determinacy)**.** *Let $K \subseteq J \subseteq L$ be certified finite windows of a public* StreamName$^{\uparrow}$ *rational stream package. Suppose the* RegSeqRat *handoff consumes the same* DyadicRatCore *finite ledger along the direct restriction route from L to K and along the two-step restriction route from L to J and then from J to K. Then the induced finite-window* Real$^{\uparrow}$ *seal classifier rows agree on every certified index of K.*

The common seal row is determined only by the pointwise RatStreamNameCarrier *and* RatStreamNameClassifier *rows, the accepted* DyadicRatCore *ledger entries over K, the displayed* hsame *transports, the finite* Cont *routes, and the shared* Pkg *provenance. It does not use completion data, quotient representatives, host-index equality, or any* Real$^{\uparrow}$ *completeness datum.*

Proof. First compare the two restriction routes at the StreamName source level. By Definition 126.101, the direct route L to K and the two-step route through J expose the same pointwise source rows and the same finite ledger over K.

Next transport those rows to the seal side. Apply Definition 126.89 at each certified index of K. Since the carrier row, classifier row, hsame transport, Cont route, Pkg provenance, and DyadicRatCore ledger entry are the same on both routes, the corresponding Real$^{\uparrow}$ seal classifier row is the same at that index.

Finally use Definition 126.100. It says that the RegSeqRat consumer has no additional public handoff data beyond those pointwise rows and finite ledger entries. The exclusion clauses in the handoff and stability theorems keep completion, quotient, host-index, and Real$^{\uparrow}$-completeness data outside the determinacy claim, so the rowwise agreement over K is exactly the stated finite-window seal determinacy. □

Theorem 126.103 (StreamName RegSeqRat RealUp triple seal cover)**.** *Let $\mathcal{S}$ be an accepted finite* StreamName$^{\uparrow}$ *rational-window package, let $\mathcal{R}$ be the* RegSeqRat *pointwise seal package read from the same certified unary indices, and let $\mathcal{E}$ be the* Real$^{\uparrow}$ *sealing endpoint reached by the finite-window consumer. Then $\mathcal{S}$, $\mathcal{R}$, and $\mathcal{E}$ determine the same public read exactly when their unary observation rows, finite* Cont *ledger,* hsame *endpoint transports, and* Pkg *provenance rows agree.*

Equivalently, the three-way cover contains only the pointwise RatStreamNameCarrier *and* RatStreamNameClassifie *observations, the accepted* RegSeqRat *pointwise seal rows, the* Real$^{\uparrow}$ *endpoint seal row, the finite transport ledger, and the shared provenance package. It contains no quotient representative, completion modulus, host index equality, or additional regularity datum beyond those rows.*

Proof. First compare the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}$ sides pointwise. By Definition 126.89, agreement of the unary observations, hsame transports, finite Cont rows, and Pkg provenance preserves the accepted seal row at each certified index.

Next paste the pointwise seals. Apply Definition 126.90 to the certified finite window. The pasted package has exactly the pointwise rows and the shared finite ledger, so it is independent of any extra quotient or completion representative.

Now exhaust the visible handoff. By Definition 126.100, every public row read by the RegSeqRat consumer is one of the displayed unary observation, transport, ledger, or provenance rows. Thus the RegSeqRat package contributes no additional public datum once those rows agree.

Finally use Definition 126.101. It identifies the finite-window route to the $\mathsf{Real}^{\uparrow}$ sealing endpoint with the same restricted source surface. Therefore the StreamName window package, the RegSeqRat pointwise seal package, and the RealUp endpoint seal determine one public read exactly in the stated agreement conditions, and the reverse direction reads the same four families of rows back into the three packages. □

Theorem 126.104 (StreamName RegSeqRat RealUp tail-selector factorization)**.** *Every accepted finite-window* $\mathsf{StreamName}^{\uparrow}$ *handoff whose* $\mathsf{RegSeqRat}$ *carrier supplies the scheduled-window projection and whose* $\mathsf{Real}^{\uparrow}$ *endpoint read uses finite-prefix exactness factors the tail-selector read through exactly three public rows: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier row, the* $\mathsf{RegSeqRat}$ *scheduled projection and readback row, and the* $\mathsf{Real}^{\uparrow}$ *finite-prefix classifier row.*

The factorization exports no quotient stream, selected limit, host equality, countable choice row, or ambient completeness row. All exported data are already contained in the certified finite $\mathsf{ProbeBundle}$ *ledger, the displayed* Cont *routes, the unary observation rows, the* hsame *transports, and the shared* Pkg *provenance.*

Proof. Start with the three-way cover of Definition 126.103. It identifies the accepted $\mathsf{StreamName}^{\uparrow}$ window, the pointwise $\mathsf{RegSeqRat}$ seal package, and the $\mathsf{Real}^{\uparrow}$ endpoint read on the same unary observations, finite transport ledger, endpoint transports, and provenance rows.

The first factor is the StreamName finite-window classifier. By Definition 126.86, the scheduled seal over the accepted window is exhausted by the visible StreamName carrier and classifier observations together with the finite ledger rows. Thus the tail-selector source read begins at the finite-window classifier rather than at a quotient stream or host-index equality.

The second factor is the RegSeqRat scheduled projection and readback row. Apply Definition 126.100: every public row consumed by the $\mathsf{RegSeqRat}$ carrier is already one of the certified pointwise rows, Cont routes, hsame transports, and Pkg provenance rows. The scheduled projection therefore reads back exactly the finite window selected by the StreamName classifier.

The third factor is the RealUp finite-prefix classifier. Use Definition 126.108 to compare all finite windows selected by the tail read. Since the overlaps are already compatible in the triple cover, the endpoint classifier depends only on the finite prefix ledger presented by the RegSeqRat readback row. This gives the claimed three-row factorization and excludes quotient, selected-limit, countable-choice, host-equality, and ambient-completeness exports. □

Theorem 126.105 (StreamName RegSeqRat tail-budget uniformity)**.** *Let a finite accepted* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$ *handoff packet present StreamName finite observations through a common pointwise classifier, with matching* BHist *rows,* hsame *transports, finite* Cont *routes, shared* Pkg *provenance, and a displayed* $\mathsf{NameCert}$ *for the handoff surface. Suppose the* $\mathsf{RegSeqRat}$ *tail selector carries a dyadic budget row consumed by the finite-prefix* $\mathsf{Real}^{\uparrow}$ *endpoint. Then every later finite window accepted by the same StreamName pointwise classifier factors through the same tail-budget certificate.*

Equivalently, once the packet has fixed the StreamName finite observations, the RegSeqRat tail selector, and the RealUp budget consumer, enlarging the finite window changes only the visible window ledger. It does not export a new tail selector, budget witness, completion datum, quotient stream, host equality, or ambient regularity row outside the certified NameCert*,* Cont*,* hsame*,* BHist*, and* Pkg *surface.*

Proof. First apply Definition 126.104. The tail-selector read factors through the StreamName finite-window classifier, the RegSeqRat scheduled projection and readback row, and the RealUp finite-prefix classifier. Thus the budget row is already located on the displayed finite ledger rather than on a selected completion representative.

Next use Definition 126.102 for a later finite window accepted by the same pointwise StreamName classifier. The nested window has the same BHist source rows, hsame transport witnesses, finite Cont routes, and Pkg provenance on the overlap, so the RegSeqRat scheduled projection reads the same tail-selector certificate on the shared part.

Finally compare all later windows with Definition 126.108. The finite cover determinacy theorem pastes the overlapping StreamName, RegSeqRat, and RealUp rows into one cover-indexed source surface. Therefore each RealUp budget consumer sees the same dyadic tail-budget certificate, and no further completion, quotient, host-index, or off-ledger regularity datum can enter the handoff. □

Theorem 126.106 (StreamName RegSeqRat RealUp cover tail-budget finality)**.** *Let* $\{W_i\}_{i\in I}$ *be a finite public* $\mathsf{StreamName}^{\uparrow}$ *cover feeding the same* RegSeqRat *finite-window consumer and the same finite-window* $\mathsf{Real}^{\uparrow}$ *endpoint. Suppose every displayed pairwise overlap satisfies the nested-window handoff hypotheses, every nested pair satisfies the nested-window seal determinacy hypotheses, and every triple overlap is compatible in the triple seal cover. Then the cover forces a single tail-budget certificate for the* RegSeqRat *consumer, and every endpoint read factors through exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier row, the* RegSeqRat *scheduled projection and readback row, and the* $\mathsf{Real}^{\uparrow}$ *finite-prefix classifier row.*

The forced certificate is a finite-cover surface over the displayed NameCert*,* ProbeBundle*,* InBundle*,* BHist*,* Cont*,* hsame*, and* Pkg *rows. It contains no quotient stream data, selected limit, host index equality, countable choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First identify the source rows on pairwise overlaps. Apply Definition 126.101 to each displayed nested overlap in the finite cover. The direct and two-step restrictions expose the same StreamName classifier rows, unary observations, finite bundle ledger, membership rows, and hsame comparisons.

Next transport those source rows through the endpoint seal. Use Definition 126.102 on the same nested overlaps, and then apply Definition 126.103 on every displayed triple. The RegSeqRat seal rows and the RealUp endpoint rows are therefore read from the same finite transport, provenance, and pointwise observation surface throughout the cover.

Now factor each endpoint read. By Definition 126.104, every accepted read passes through exactly the StreamName finite-window classifier, the RegSeqRat scheduled projection and readback row, and the RealUp finite-prefix classifier row. The factorization clauses rule out quotient stream data, selected limits, host index equality, countable choice, and ambient Real completeness as exported rows.

Finally apply Definition 126.105 to the cover-indexed family of accepted windows. Since all overlaps have the same pointwise classifier, BHist rows, hsame transports, finite Cont routes, shared Pkg provenance, and displayed NameCert surface, the RegSeqRat tail selector carries one dyadic

budget certificate across the finite cover. Thus the endpoint reads are forced by the same finite-cover surface and by exactly the three classifier/projection rows named above. □

Theorem 126.107 (StreamName finite-section tail-window lock)**.** *Let Q be an accepted finite request section of the current L10* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose that Q exposes the same* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row,* $\mathsf{RegSeqRat}^{\uparrow}$ *scheduled projection and readback row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and StreamName tail-window coverage packet as a finite cover satisfying Definition 126.106. Then every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing finite consumer of Q is locked to the same tail-budget certificate forced by that cover.*

Consequently each such consumer factors through exactly the $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *scheduled projection and readback row, and the* $\mathsf{Real}^{\uparrow}$ *finite-prefix classifier row. The lock exports no quotient stream, selected limit, host index equality, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or standard bridge row.*

Proof. First apply Definition 126.586 to the accepted section Q. The shared finite window, dyadic ledger, RegSeqRat readback, and RealUp seal identify Q with the same four-face finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 126.224 to every finite $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer whose source reads the displayed tail-window coverage packet. Each consumer is therefore read only through the packet rows already named by the finite window, the accepted $\mathsf{RegSeqRat}$ spine, the DyadicRatCore ledger, and the shared $\mathsf{Cont}/\mathsf{hsame}/\mathsf{Pkg}$ support.

Now use Definition 126.225. Since the section and the cover expose the same tail-window packet, the possible consumer source is unique on the displayed StreamName, RegSeqRat, DyadicRatCore, continuation, history, package, and ledger rows. There is no second source row from which a quotient representative, host index, completion object, or bridge datum could be read.

Finally invoke Definition 126.106. The finite cover has already forced one tail-budget certificate and one three-row factorization for endpoint reads. Transporting the accepted section Q through the same pullback and tail-window packet therefore locks every eligible consumer to that certificate and to exactly the StreamName classifier row, the RegSeqRat scheduled projection/readback row, and the RealUp finite-prefix classifier row. □

Theorem 126.108 (StreamName RegSeqRat finite-window cover determinacy)**.** *Let $\{W_i\}_{i\in I}$ be a finite public* $\mathsf{StreamName}^{\uparrow}$ *cover of rational windows feeding the same* $\mathsf{RegSeqRat}$ *finite-window consumer and the same finite-window* $\mathsf{Real}^{\uparrow}$ *handoff surface. Suppose every displayed pairwise overlap satisfies the overlap-source and nested-window determinacy hypotheses for the StreamName source rows, and suppose every three-way overlap is compatible in the triple seal cover. Then the induced* $\mathsf{RegSeqRat}$ *seal rows and finite-window* $\mathsf{Real}^{\uparrow}$ *handoff rows determine one cover-indexed source surface: the selected* $\mathsf{RatHistoryClassifier}$ *rows, the* $\mathsf{ProbeBundle}$ *ledger, the* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *observations, and the displayed* hsame *comparisons.*

The determined surface contains no completion datum, quotient datum, host index equality, RegSeqRat regularity claim outside the finite seal rows, or $\mathsf{Real}^{\uparrow}$ *seal classifier outside the displayed ledger boundary.*

Proof. Use Definition 126.87 and Definition 126.88 on each displayed pairwise overlap. They identify the source row, unary observation, bundle membership, and hsame comparison read by the two windows.

Induct over the finite cover. The empty and singleton cases are just the displayed window rows. In the step case, apply Definition 126.102 to paste the new window into the already determined finite surface along the pairwise overlap.

For compatibility of the pasted overlaps, use Definition 126.103. The triple cover keeps the StreamName, RegSeqRat, and RealUp reads on the same finite ledger, so no completion, quotient, host-index, external regularity, or off-ledger classifier row is introduced. □

Theorem 126.109 (StreamName RegSeqRat cover Real handoff descent)**.** *Let $\{W_i\}_{i\in I}$ be a finite public* $\mathsf{StreamName}^{\uparrow}$ *cover feeding one* $\mathsf{RegSeqRat}$ *finite-window consumer and one finite-window* $\mathsf{Real}^{\uparrow}$ *endpoint. Suppose that every displayed pairwise overlap satisfies the overlap-source hypotheses and the nested-window handoff hypotheses, that every nested pair satisfies seal determinacy, and that every displayed triple overlap satisfies the triple seal cover. Then the induced* $\mathsf{Real}^{\uparrow}$ *finite-window handoff descends to the single cover-indexed source surface determined by Definition 126.108.*

The visible rows are exactly the selected $\mathsf{RatHistoryClassifier}$ *rows, the* $\mathsf{ProbeBundle}$ *ledger, the* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *observations, displayed* hsame *comparisons,* $\mathsf{DyadicRatCore}$ *finite ledger rows, and the endpoint seal rows licensed by the triple cover. The descent contains no completion data, quotient representatives, host index equality, countable choice, ambient completeness, or off-ledger* $\mathsf{Real}^{\uparrow}$ *seal classifiers.*

Proof. First apply Definition 126.108 to the finite public cover. Its conclusion supplies a single cover-indexed StreamName source surface for the selected classifier rows, bundle ledger, membership rows, unary observations, and displayed hsame comparisons.

Next stabilize the endpoint-facing seal rows on nested overlaps. By Definition 126.102, two nested routes reading the same $\mathsf{DyadicRatCore}$ finite ledger induce the same finite-window $\mathsf{Real}^{\uparrow}$ seal classifier row on every certified subwindow.

Now compare triples. The compatibility hypothesis feeds Definition 126.103, so the StreamName source rows, the $\mathsf{RegSeqRat}$ finite ledger, and the $\mathsf{Real}^{\uparrow}$ endpoint seal rows read the same public package on each displayed three-way overlap.

Finally exhaust the consumer-visible surface with Definition 126.100. Every row seen by the $\mathsf{RegSeqRat}$ consumer or by the finite-window $\mathsf{Real}^{\uparrow}$ endpoint is one of the listed source, ledger, membership, observation, comparison, finite $\mathsf{DyadicRatCore}$, or licensed endpoint-seal rows. The exclusion clauses in the cover, nested-window, triple-cover, and exhaustion theorems rule out completion data, quotient representatives, host index equality, countable choice, ambient completeness, and off-ledger $\mathsf{Real}^{\uparrow}$ seal classifiers. □

Theorem 126.110 (StreamName RegSeqRat cover Real four-face exit descent)**.** *Let $\{W_i\}_{i\in I}$ be a finite public* $\mathsf{StreamName}^{\uparrow}$ *cover satisfying the hypotheses of Definition 126.109. If the induced* $\mathsf{Real}^{\uparrow}$ *endpoint is consumed by a current Phase Real and Completion read, then that consumer factors through exactly the four displayed faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the visible BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows on the cover-indexed surface.*

The factorization contains no quotient stream, no selected limit, no countable choice row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, no host sequence equality, and no standard bridge row.*

Proof. First apply Definition 126.109. It descends the finite-window $\mathsf{Real}^{\uparrow}$ handoff to the single cover-indexed StreamName source surface, with the selected classifier rows, bundle ledger, membership rows, unary observations, displayed hsame comparisons, dyadic ledger rows, and endpoint seal rows all on one public package.

Next invoke Definition 126.372. The descended endpoint is therefore read as the same four-object exit route: the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal boundary. No additional top-level Real-and-completion coordinate is introduced.

Now use Definition 126.446, Definition 126.449, and Definition 126.450. These identify the current Phase Real and Completion exit packet with exactly the four displayed faces and the visible BHist, hsame, Cont, Pkg, and NameCert support rows. Thus the cover descent and the current open-phase exit packet have the same finite BEDC surface.

Finally apply Definition 126.476. Its exclusion clauses, together with the exclusions in the cover descent and four-object exactness theorems, rule out quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, host sequence equality, and bridge rows. Hence every current consumer of the descended endpoint factors through the four-face exit surface and through no larger object. □

Theorem 126.111 (StreamName-RegSeqRat uniform finite-window seal)**.** *Let W be a finite* BHist *observation window for a public* $\mathsf{StreamName}^{\uparrow}$ *rational stream package, and suppose that the two endpoints of W satisfy* RatStreamNameFiniteWindowClassifier*. Then the StreamName pointwise seal, the* RegSeqRat *scheduled seal, every certified nested-window handoff through W, and the finite-window* $\mathsf{Real}^{\uparrow}$ *readback select the same finite observation packet under* hsame*.*

The common packet consists exactly of the pointwise RatStreamNameCarrier *and* RatStreamNameClassifier *observations at the certified unary indices of W, the accepted* RegSeqRat *pointwise seal rows, the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row, the finite* Cont *ledger, the* ProbeBundle *membership rows, and the shared* Pkg *provenance. It contains no quotient representative, completion modulus, host index equality, or additional regularity datum.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameRegseqratUniformFiniteWindowSeal`

Proof. First form the pointwise seal over the certified endpoints of W. Apply Definition 126.89 at each unary index. The endpoint classifier hypotheses give the same carrier row, classifier row, hsame transport, finite Cont route, Pkg provenance, and accepted DyadicRatCore ledger entry for the StreamName and RegSeqRat reads.

Next paste these pointwise rows over the whole finite window. By Definition 126.90, the scheduled RegSeqRat seal over W is the unique finite packet assembled from those rows and the same finite ledger. No quotient or completion representative is introduced by this pasting step.

Now compare all certified subwindows of W. Use Definition 126.101 to identify the direct and nested handoff routes with the same restricted source surface. Thus every nested-window consumer sees the pointwise packet just constructed, with the same ProbeBundle membership rows and hsame comparisons.

Finally apply Definition 126.103. The triple cover identifies the StreamName window package, the RegSeqRat scheduled seal, and the $\mathsf{Real}^{\uparrow}$ endpoint readback whenever their unary observations, finite Cont ledger, hsame transports, and Pkg provenance agree. Combining the four steps gives one uniform finite-window seal packet under hsame. □

Theorem 126.112 (StreamName RealUp finite-window seal boundary)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window whose endpoints satisfy* RatStreamNameFiniteWindowClassifier*. Suppose that the* RegSeqRat *handoff over W carries accepted pointwise seal rows, the finite* Cont *ledger, the* ProbeBundle *membership rows, the corresponding* InBundle *rows, the* UnaryHistory *observations, and shared* Pkg *provenance. Then these rows determine exactly the finite-window* $\mathsf{Real}^{\uparrow}$ *seal boundary read from W.*

Conversely, every public finite-window Real$^{\uparrow}$ *seal read arising from this handoff factors through those same StreamName source rows, accepted* RegSeqRat *pointwise seal rows, bundle-membership rows, unary observations, finite ledger entries, and provenance rows. The boundary contains no completion object, quotient representative, host index equality, or regularity datum outside the displayed* RegSeqRat *seal.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameRealupSealBoundary`

Proof. First recover the StreamName source surface. Apply Definition 126.79. It identifies the finite public window read with the displayed rational StreamName observations, the unary histories at the certified indices, the finite Cont routes, and the shared Pkg provenance.

Next pass to the scheduled rational seal. By Definition 126.86, the RegSeqRat side contributes only the accepted pointwise seal rows over the same unary observations, the same bundle-membership rows, and the same finite ledger. No additional sequence regularity row is available at this step.

Now compare the three visible seal surfaces. Use Definition 126.103. The StreamName window, the RegSeqRat scheduled seal, and the Real$^{\uparrow}$ endpoint readback cover one another exactly when their unary observations, bundle rows, finite transport ledger, hsame comparisons, and provenance agree.

Finally apply Definition 126.111. Uniformity over nested finite windows prevents a later subwindow from adding completion data, quotient representatives, host index equality, or hidden regularity data. Thus the forward direction gives a unique finite-window Real$^{\uparrow}$ seal boundary, and the reverse direction factors every public read through precisely the displayed rows. □

Theorem 126.113 (StreamName-RegSeqRat-RealUp seal pullback uniqueness)**.** *Let W be a finite public* StreamName$^{\uparrow}$ *rational observation window. Suppose two finite-window* Real$^{\uparrow}$ *seal reads over W arise from the same StreamName window rows, the same accepted* RegSeqRat *pointwise seal rows, the same* ProbeBundle *and* InBundle *membership rows, the same* UnaryHistory *observations, the same finite* Cont *ledger, the same* hsame *transports, and the same* Pkg *provenance. Then the two reads determine the same* Real$^{\uparrow}$ *finite-window seal boundary.*

Conversely, any equality of Real$^{\uparrow}$ *finite-window seal boundaries arising from W factors back through that displayed StreamName–*RegSeqRat *source surface: the common boundary is witnessed by the same StreamName window rows, the same accepted pointwise seal rows, the same bundle-membership rows, the same unary observations, the same finite ledger, the same* hsame *transport surface, and the same provenance. This pullback uses no quotient stream, selected host limit, host index equality, completion witness, or ambient completeness assumption.*

Proof. First expose the source surface for each read. Apply Definition 126.79 to both finite-window handoffs. The common StreamName window rows, unary observations, finite Cont routes, hsame transports, and Pkg provenance give the same public source packet.

Next exhaust the rational scheduled seal. By Definition 126.86, each handoff has only the displayed accepted RegSeqRat pointwise seal rows, the same ProbeBundle and InBundle membership rows, and the same finite ledger available to it.

The three seal surfaces now compare over a single packet. Use Definition 126.103 to identify the StreamName window surface, the RegSeqRat scheduled seal, and the Real$^{\uparrow}$ readback surface. Then Definition 126.111 prevents any nested finite subwindow from changing that packet.

It remains to read off the boundary. Apply Definition 126.112. The forward direction gives equality of the two Real$^{\uparrow}$ finite-window seal boundaries. The converse direction says that any such equality is already a factorization through the same StreamName–RegSeqRat source rows,

so no quotient stream, selected host limit, host index equality, completion witness, or ambient completeness datum can enter the pullback. □

Theorem 126.114 (StreamName-RegSeqRat-RealUp terminal pullback factorization)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window, and let R be a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal read over W. Suppose R is presented through the public StreamName rational window interface and the accepted* RegSeqRat *pointwise seal over that window. Then R factors through exactly the StreamName point rows, the* RegSeqRat *pointwise seal rows, the* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary, the finite* Cont *ledger, the* hsame *transports, the* ProbeBundle *and* InBundle *membership rows, the* UnaryHistory *observations, and the shared* Pkg *provenance.*

Conversely, the displayed StreamName point rows, accepted RegSeqRat *pointwise seal rows,* $\mathsf{Real}^{\uparrow}$ *boundary row, finite* Cont *ledger,* hsame *transports, bundle-membership rows, unary observations, and provenance rows repack into the same terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal boundary. The factorization carries no quotient stream, selected host limit, host index equality, completion witness, ambient completeness row, or metric-completeness row.*

Proof. First build the uniform public packet. Apply Definition 126.111. It identifies the StreamName point rows and accepted RegSeqRat pointwise seal rows over W with one finite observation packet under the same hsame transports, finite Cont ledger, bundle-membership rows, unary observations, and provenance.

Next read the terminal boundary of the $\mathsf{Real}^{\uparrow}$ seal. Use Definition 126.112. The forward direction says that the displayed public packet determines the finite-window $\mathsf{Real}^{\uparrow}$ seal boundary, and the reverse direction repacks those same rows into the boundary read.

Finally compare any two terminal presentations of the read. Apply Definition 126.113. The pullback uniqueness identifies presentations with the same StreamName rows, accepted RegSeqRat rows, bundle-membership rows, unary observations, Cont ledger, hsame transports, and provenance.

The preceding steps give both directions of the factorization. Since each ingredient is read from the displayed finite packet and its $\mathsf{Real}^{\uparrow}$ boundary, there is no row from quotient streams, selected host limits, host index equality, completion witnesses, ambient completeness, or metric-completeness data available to the terminal seal. □

Theorem 126.115 (StreamName-RegSeqRat-DyadicRatCore-RealUp terminal pullback factorization)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window, and let R be a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal read over W. Suppose R is presented through the public StreamName rational window interface, the accepted* RegSeqRat *pointwise seal over that window, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger anchored to the same window. Then R factors through exactly four visible faces: the StreamName point and window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* RegSeqRat *pointwise and readback rows, and the local* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary.*

The only support rows available to this factorization are the displayed BHist *observations, componentwise* hsame *transports, finite* Cont *routes, shared* Pkg *provenance,* ProbeBundle *and* InBundle *membership rows,* UnaryHistory *observations, and local* NameCert *support. Conversely, any packet displaying precisely those four faces and support rows repacks into the same terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal read. It exports no quotient stream equality, selected host limit, host real equality, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or choice datum.*

Proof. Start from the three-face terminal pullback of Definition 126.114. It places the terminal $\mathsf{Real}^{\uparrow}$ finite-window read on the StreamName point rows, accepted RegSeqRat pointwise seal rows, and $\mathsf{Real}^{\uparrow}$ seal boundary, with the displayed finite BHist, hsame, Cont, Pkg, ProbeBundle, InBundle, and UnaryHistory support.

Next insert the dyadic ledger face. Apply Definition 126.612. Its forward direction says that an accepted seal-facing read has exactly the StreamName, RegSeqRat, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, with the listed finite support rows. Its converse direction repacks any packet with those four faces into the same admissible public interface.

It remains to identify the dyadic face with the same finite window used by the regular-rational readback. Use the Dyadic-RegSeq route factorization Definition 126.444, as cited in the four-face exactness proof. This pins every endpoint, radius, distance, and readback coordinate to W before either the RegSeqRat or $\mathsf{Real}^{\uparrow}$ consumer can inspect the packet.

Combining these steps gives both directions of the terminal pullback. The three-face theorem supplies the StreamName–RegSeqRat–$\mathsf{Real}^{\uparrow}$ terminal boundary, the four-face exactness theorem supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger as the fourth visible face, and the route factorization prevents a detached tolerance route from entering the packet.

The excluded objects are therefore not public coordinates of the factorization. There is no quotient stream equality, selected host limit, host real equality, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or choice datum available to the terminal read. □

Theorem 126.116 (StreamName-RegSeqRat-RealUp composite terminal pullback)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window. Suppose two consecutive terminal pullback readings pass first through the displayed* RatStreamNameCarrier *and* RatStreamNameClassifier *rows of W, then through the accepted* RegSeqRat *pointwise seal rows, and finally to the* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary. Then the two-stage terminal reading determines exactly the same finite* BHist *packet as the single composite pullback from the StreamName rows into the* $\mathsf{Real}^{\uparrow}$ *boundary row.*

The shared packet consists precisely of the RatStreamNameCarrier *rows, the* RatStreamNameClassifier *rows, the accepted* RegSeqRat *pointwise seal rows, the* $\mathsf{Real}^{\uparrow}$ *boundary row, the* ProbeBundle *rows, the* InBundle *membership rows, the* UnaryHistory *observations, the finite* Cont *ledger, the componentwise* hsame *transports, the common* Pkg *provenance, and the local* NameCert *witnessing the public StreamName window. It contains no quotient stream, selected host limit, host index equality, completion witness, ambient completeness row, or metric-completeness row.*

Proof. First build the finite public packet over W. Apply Definition 126.111. This identifies the displayed StreamName carrier and classifier rows with the accepted RegSeqRat pointwise seal rows, the bundle rows, the unary observations, the finite Cont ledger, the componentwise hsame transports, and the shared Pkg provenance.

Next read the endpoint seal as a boundary row. Use Definition 126.112. The boundary construction says that the same packet determines the $\mathsf{Real}^{\uparrow}$ finite-window seal boundary, and that every public boundary read factors back through the same StreamName rows, accepted RegSeqRat rows, ProbeBundle and InBundle rows, UnaryHistory observations, finite ledger, transports, and provenance.

Now compare the two consecutive pullback readings with the direct composite reading. Apply Definition 126.113. Since both routes expose the same finite source surface and the same $\mathsf{Real}^{\uparrow}$ boundary row, pullback uniqueness identifies their finite BHist packets component by component.

Finally invoke Definition 126.114. The terminal factorization repacks the identified packet through the local NameCert and shows that neither the two-stage route nor the composite route has access to a quotient stream, selected host limit, host index equality, completion witness, ambient completeness row, or metric-completeness row. Thus the consecutive and composite terminal pullbacks are the same finite public packet. □

Theorem 126.117 (StreamName terminal reindex pullback invariance)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet, and let R be a certified public finite-window reindex route for W. Suppose R preserves the public finite observation section, the certified endpoint rows, the displayed* BHist*, the componentwise* hsame *transports, the finite* Cont *ledger, the shared* Pkg *provenance, and the local* NameCert *rows. Then the terminal pullback read through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is invariant under R.*

The invariant packet has the same visible StreamName window, the same $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *accepted readback rows, the same* $\mathsf{Real}^{\uparrow}$ *boundary row, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

It introduces no quotient stream equality, selected host limit, host real equality, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or choice datum.

Proof. First read the reindexed StreamName side. Apply Definition 126.648. Its hypotheses are exactly the preserved public finite observation section, endpoint rows, displayed BHist, componentwise hsame, finite Cont ledger, shared Pkg, and local NameCert rows, so the reindexed route reaches the same terminal four-face pullback surface.

Next pin the four visible faces of that surface. Apply Definition 126.115. The factorization says that the terminal read has only the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ boundary row, and the listed support rows. Since the reindex route preserves those rows component by component, the visible faces are unchanged.

Finally compare the two-stage and direct terminal reads. By Definition 126.116, both presentations determine one finite BHist packet over the same support surface. Hence the terminal pullback is invariant under R, and no quotient, selected host limit, host real equality, completeness, bridge, trichotomy, open-cover compactness, or choice coordinate is added. □

Theorem 126.118 (StreamName terminal pullback idempotence)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet. Suppose a terminal four-face pullback packet over W already exposes exactly the StreamName finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *accepted readback rows, the* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Suppose further that the same packet displays the corresponding ProbeBundle*,* InBundle*, and* UnaryHistory *rows required by the public finite observation surface. Then applying the terminal pullback reading to this packet again exposes the same finite BEDC packet.*

Equivalently, the terminal four-face pullback is a normal form for certified public StreamName finite-window reads: after the four visible faces and their support rows are present, a further terminal reading cannot add a quotient stream, host equality, selected limit, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or choice coordinate.

Proof. First identify the packet already in hand. Apply Definition 126.115. Its converse direction repacks exactly the displayed StreamName, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, together with the listed BHist, hsame, Cont, Pkg, bundle, unary-history, and NameCert support rows, into the terminal finite-window seal read over W.

Next compare the second reading with the first one. By Definition 126.116, a two-stage terminal read and the direct composite read determine the same finite BHist packet whenever they expose the same StreamName rows, accepted $\mathsf{RegSeqRat}^{\uparrow}$ rows, $\mathsf{Real}^{\uparrow}$ boundary row, finite ledger, transports, provenance, bundle rows, unary observations, and local NameCert support.

Finally account for certified reindex routes. Apply Definition 126.117. Any certified route preserving the displayed finite observation section, endpoint rows, BHist, hsame, Cont, Pkg, and NameCert rows reaches the same terminal four-face surface. Thus the second terminal read has no coordinate outside the normal-form packet, and the excluded quotient, host-equality, selected-limit, completeness, bridge, trichotomy, open-cover compactness, and choice data remain unavailable. □

Theorem 126.119 (StreamName terminal four-face support deletion obstruction)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet whose terminal four-face read exposes exactly the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *faces. Suppose the displayed support surface consists of*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

together with the public finite-observation rows ProbeBundle, InBundle, *and* UnaryHistory. *If any one of the displayed support rows* BHist, hsame, Cont, Pkg, *or* NameCert *is deleted, or if one of the public finite-observation rows* ProbeBundle, InBundle, *or* UnaryHistory *is deleted, the remaining packet is not accepted as a terminal four-face pullback over W.*

The missing row cannot be supplied by quotient stream equality, selected host limits, host equality, countable choice, ambient completeness, metric-completeness, bridge rows, trichotomy, or open-cover compactness.

Proof. First identify the four visible faces and their support surface. Apply Definition 126.115. It says that an accepted terminal read over W factors through exactly the StreamName window, dyadic endpoint-radius-distance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and local $\mathsf{Real}^{\uparrow}$ finite-window boundary, with only the displayed BEDC support rows and public finite-observation rows available.

Next remove one displayed row. For one of BHist, hsame, Cont, Pkg, or NameCert, apply Definition 126.436. That minimality result says that deleting any of these rows destroys the public finite-window support needed by the StreamName packet before the terminal pullback can be formed.

For one of ProbeBundle, InBundle, or UnaryHistory, use the same four-face factorization. These rows are part of the public finite-observation surface read by the factorization itself, so deleting one removes a coordinate required by the accepted terminal packet.

Finally compare with the normal form. By Definition 126.118, a complete support surface is stable under another terminal read. Hence the converse direction of the four-face factorization can repack only the complete surface, not the shortened packet. The excluded quotient, host-limit, host-equality, choice, completeness, bridge, trichotomy, and compactness rows are not coordinates of that normal form, so none of them can fill the missing support row. □

Theorem 126.120 (StreamName terminal pullback support exactness)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet. A terminal four-face pullback packet over W is accepted exactly when it exposes the StreamName finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *accepted readback rows, the local* $\mathsf{Real}^{\uparrow}$ *finite-window seal*

boundary, and exactly the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},$$

together with the public finite-observation rows ProbeBundle, InBundle, *and* UnaryHistory.

Conversely, any packet over W with exactly those visible faces and support rows repacks into the same terminal four-face pullback normal form. The acceptance test exports no quotient stream, selected host limit, host equality, countable choice, ambient completeness row, metric-completeness row, bridge row, trichotomy row, or open-cover compactness row.

Proof. For the forward direction, start from the terminal four-face factorization of Definition 126.115. It identifies every accepted terminal read over W with the StreamName finite-window face, the dyadic endpoint-radius-distance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, the $\mathsf{Real}^{\uparrow}$ finite-window boundary, and the listed finite support rows.

Next use Definition 126.117. Since certified reindex routes preserve the public finite observation section, endpoint rows, BHist, hsame, Cont, Pkg, and NameCert rows, support cannot be enlarged by changing the terminal presentation of the same window.

The local deletion obstruction is Definition 126.119. It says that deleting any displayed BEDC support row, or any public ProbeBundle, InBundle, or UnaryHistory observation row, prevents acceptance, and that quotient, host-limit, host-equality, choice, completeness, bridge, trichotomy, and compactness rows cannot replace the missing coordinate.

For the reverse direction, assume a packet displays exactly those four faces and support rows. The converse direction of Definition 126.115 repacks it into the terminal finite-window seal read over W.

Finally apply Definition 126.118. Once the four faces, the five BEDC support rows, and the three public finite-observation rows are present, another terminal read returns the same packet. Hence the support list is both sufficient and necessary for acceptance, and quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric-completeness rows, bridges, trichotomy, and open-cover compactness remain outside the normal form. □

Theorem 126.121 (StreamName terminal pullback support induction)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet. For the terminal four-face pullback over W, order the support rows as*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ProbeBundle},\quad \mathsf{InBundle},\quad \mathsf{UnaryHistory}.$$

An accepted terminal four-face pullback over W is obtained by finite induction over this ordered support list: the empty prefix carries no accepted terminal extension, and each proper prefix has an accepted extension to the next prefix exactly when the next displayed row in the list is present. The full prefix is the unique terminal normal form over the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *faces.*

No quotient stream, selected host limit, host equality, countable choice, ambient completeness row, metric-completeness row, bridge row, trichotomy row, or open-cover compactness row can replace a missing step of this finite support induction.

Proof. The base stage is the deletion obstruction. By Definition 126.119, removing any one of the displayed BEDC support rows or public finite-observation rows prevents acceptance. In particular the empty prefix, and every prefix missing its next displayed row, cannot be accepted as the terminal four-face pullback over W.

For the induction step, assume a prefix of the displayed list has been built and consider the next row. If that next row is absent, the deletion obstruction again blocks acceptance, and the refused replacement rows cannot fill the coordinate. If the next row is present, the prefix extends by the corresponding finite BEDC support coordinate or public finite-observation coordinate.

At the terminal stage, apply Definition 126.115. It repacks exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces together with the complete displayed support list. Then use Definition 126.120 to identify that complete list as necessary and sufficient for acceptance.

Finally apply Definition 126.118. Once the full support prefix is present, a further terminal read returns the same packet, so no second terminal normal form exists over the four faces. The explicit refusals of quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric-completeness rows, bridge rows, trichotomy, and open-cover compactness remain part of the induction invariant at every prefix. □

Theorem 126.122 (StreamName terminal pullback row-order determinacy)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet. Suppose an accepted terminal four-face pullback over W has support built by the induction of Definition 126.121. Then the support row order is uniquely fixed as*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ProbeBundle},\quad \mathsf{InBundle},\quad \mathsf{UnaryHistory}.$$

Swapping two displayed rows, omitting a displayed row, or replacing a displayed row by any other row fails to present the same terminal normal form over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$.

In particular, quotient streams, selected host limits, host equality, countable choice, ambient completeness rows, metric-completeness rows, bridge rows, trichotomy rows, and open-cover compactness rows are not replacement rows for this terminal support order.

Proof. Start with Definition 126.121. It builds the accepted terminal support by the displayed finite prefix order, so each stage has a distinguished next row and the full prefix is the terminal normal form over the four faces.

The exactness theorem, Definition 126.120, identifies the same displayed list as the necessary and sufficient support for acceptance. Thus omitting any row leaves the exact support surface, and inserting a row outside the list does not repair the missing BEDC or public finite-observation coordinate.

Next apply Definition 126.119. It rules out acceptance after deleting any displayed row and also rules out quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric-completeness rows, bridge rows, trichotomy, and open-cover compactness as substitutes for a missing coordinate.

Now use Definition 126.115. The four-face factorization repacks exactly the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, the local $\mathsf{Real}^{\uparrow}$ boundary, and the displayed support rows. A swapped presentation therefore fails to match the prefix route used to construct that factorized packet.

Finally apply Definition 126.118. Once the ordered support list has reached the terminal packet, a further terminal read returns the same packet. Hence the accepted row order is deterministic, and no swapped, omitted, or replaced support list presents the same terminal normal form. □

Theorem 126.123 (StreamName terminal pullback metric-completion refusal)**.** *Let W be a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window observation packet, and suppose its terminal four-face pullback is accepted. No consumer can reinterpret that accepted packet as a* $\mathsf{Met}^{\uparrow}$ *row, a* $\mathsf{CompleteMetric}^{\uparrow}$

row, or an ambient-completeness witness unless the metric or completion row is separately displayed outside the terminal pullback surface.

Proof. Apply Definition 126.115 to identify the accepted terminal surface. It exposes the StreamName finite window, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint-radius-distance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and the local $\mathsf{Real}^{\uparrow}$ finite-window boundary.

Now read the support list through Definition 126.120. The only support coordinates exported with those four faces are BHist, hsame, Cont, Pkg, NameCert, ProbeBundle, InBundle, and UnaryHistory. A $\mathsf{Met}^{\uparrow}$ row, a $\mathsf{CompleteMetric}^{\uparrow}$ row, and an ambient-completeness witness are not entries of that exact list.

It remains to rule out hidden replacement by an omitted coordinate. Use Definition 126.119: a row missing from the terminal support cannot be supplied by a quotient stream, selected host limit, host equality, countable choice, ambient completeness, metric-completeness row, bridge row, trichotomy, or compactness row.

Therefore any consumer that wants metric structure or completion data must display that data as a separate row outside the terminal pullback surface. The accepted StreamName–DyadicRatCore–RegSeqRat–Real four-face packet itself does not certify a metric space, a complete metric space, or ambient real completeness. □

Theorem 126.124 (StreamName-RegSeqRat finite-window cover refinement exhaustion)**.** *Let $\mathcal{W}$ be a finite cover of public* $\mathsf{StreamName}^{\uparrow}$ *rational observation windows feeding a common* RegSeqRat *consumer and a common finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Suppose that each window in $\mathcal{W}$ carries the displayed* RatStreamNameCarrier *and* RatStreamNameClassifier *rows,* ProbeBundle *and* InBundle *membership,* UnaryHistory *observations,* hsame *transports, a finite* Cont *ledger, accepted* DyadicRatCore *ledger entries, accepted* RegSeqRat *pointwise seal rows, and shared* Pkg *provenance.*

Let $\mathcal{W}'$ be any certified finite refinement cover of $\mathcal{W}$ whose refined subwindows have the same pointwise StreamName carrier and classifier rows, the same bundle-membership rows, the same unary observations, the same hsame *transports, the same finite* Cont *ledger, the same* DyadicRatCore *ledger entries, and the same* Pkg *provenance as their parent windows. Then $\mathcal{W}$ and $\mathcal{W}'$ determine the same exhausted* RegSeqRat *consumer surface and the same finite-window* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

The common exhausted surface consists only of those displayed StreamName source rows, bundle-membership rows, unary observations, hsame *transports, finite* Cont *ledger entries,* DyadicRatCore *ledger entries,* Pkg *provenance rows, the accepted* RegSeqRat *pointwise seal rows, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row. It contains no quotient representative, completion object, host index equality, off-ledger regularity row, Classical datum, or standard-bridge datum.*

Proof. First exhaust the original finite cover. Apply Definition 126.108. It identifies all windows in $\mathcal{W}$ with one finite consumer surface because their StreamName rows, bundle memberships, unary observations, hsame transports, finite ledger entries, accepted dyadic entries, and provenance rows agree on every displayed overlap.

Next compare each refined subwindow with its parent window. Use Definition 126.101. The certified subwindow contributes no new carrier, classifier, membership, observation, transport, ledger, dyadic, or provenance row; it restricts the already exhausted parent surface to the smaller public window.

Now identify the seal rows seen by the common consumer. Apply Definition 126.111 to the original cover and to the refined cover. The accepted RegSeqRat pointwise seal rows and the finite-window $\mathsf{Real}^{\uparrow}$ endpoint readback are the same finite packet in both presentations because the refinement keeps the same displayed ledger and source rows.

Finally use Definition 126.112 and Definition 126.100. The first theorem fixes the $\mathsf{Real}^{\uparrow}$ boundary from the accepted RegSeqRat seal rows, and the second theorem exhausts the consumer surface to exactly the listed public rows. Hence the original cover and every certified finite refinement cover determine the same exhausted surface, with no quotient, completion, host-index, off-ledger regularity, Classical, or standard-bridge datum. □

Theorem 126.125 (StreamName-RegSeqRat-Real finite-window route factorization)**.** *Every public finite-window* $\mathsf{Real}^{\uparrow}$ *seal read accepted through the* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *handoff factors uniquely through the displayed* RatStreamNameCarrier *source rows, the* RatStreamNameClassifier *rows, the* ProbeBundle *and* InBundle *rows, the* UnaryHistory *observations, the* hsame *transports, the finite* Cont *ledger, the* DyadicRatCore *tolerance ledger, the accepted* RegSeqRat *pointwise seal rows, the shared* Pkg *provenance, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row.*

The factorization is unique at the public boundary: any two accepted reads with the same displayed rows determine the same exhausted RegSeqRat *consumer surface and the same finite-window* $\mathsf{Real}^{\uparrow}$ *seal boundary. No quotient representative, completion object, host index equality, Classical datum, or standard-bridge datum participates in the route.*

Proof. First identify the finite cover surface. Apply Definition 126.108. It binds the StreamName source rows, bundle-membership rows, unary observations, hsame transports, finite continuation ledger, dyadic tolerance entries, and provenance rows into one public finite-window consumer surface.

Next seal the rational side uniformly. Use Definition 126.111. The accepted RegSeqRat pointwise rows are therefore read through the same finite ledger on every nested public window, so no second route can be obtained by changing the window presentation.

Then pass to the endpoint boundary. Apply Definition 126.112. The $\mathsf{Real}^{\uparrow}$ readback is fixed by the accepted RegSeqRat seal rows and the displayed endpoint row, not by a completion object or a quotient representative.

Finally use Definition 126.124. It compares every certified finite refinement with the original cover and leaves the same exhausted consumer surface and the same endpoint seal. Thus each public finite-window $\mathsf{Real}^{\uparrow}$ read factors through exactly the listed StreamName, bundle, observation, transport, ledger, seal, provenance, and endpoint rows, with no host index equality, Classical datum, or standard-bridge datum outside the displayed route. □

Theorem 126.126 (StreamName-RegSeqRat-Real terminal nonescape boundary)**.** *Let T be any terminal consumer of a finite-window* $\mathsf{Real}^{\uparrow}$ *seal whose public route factors through the displayed* RatStreamNameCarrier *and* RatStreamNameClassifier *rows, the* ProbeBundle *and* InBundle *rows, the* UnaryHistory *observations, the* hsame *transports, the finite* Cont *ledger, the* DyadicRatCore *tolerance ledger, the accepted* RegSeqRat *pointwise seal rows, the shared* Pkg *provenance, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row. Then the terminal read of T is confined to that finite-name boundary.*

In particular, T has no access through this route to quotient representatives, selected limits, host index equality, Classical data, standard-bridge data, ambient completeness, metric-completeness rows, or any off-route completion witness.

Proof. First apply Definition 126.125. It forces the terminal $\mathsf{Real}^{\uparrow}$ read to pass through exactly the displayed StreamName carrier and classifier rows, bundle rows, unary observations, hsame transports, finite continuation ledger, dyadic tolerance ledger, accepted RegSeqRat pointwise seal rows, shared provenance, and endpoint seal row.

Next use Definition 126.124. Every certified finite refinement of the route has the same exhausted RegSeqRat consumer surface and the same endpoint boundary, so the terminal consumer cannot obtain a second route by changing the public window cover.

Now invoke Definition 126.111. The accepted rational seal rows are read uniformly over the finite public windows, which keeps the tolerance ledger and the pointwise seal packet inside the same finite-name surface.

Finally apply Definition 126.112. The endpoint $\mathsf{Real}^{\uparrow}$ boundary is fixed by those accepted finite rows and not by a quotient representative, a selected limit, host index equality, Classical data, standard-bridge data, ambient completeness, metric-completeness rows, or any off-route completion witness. Hence the terminal consumer remains inside the finite StreamName–$\mathsf{RegSeqRat}$–$\mathsf{Real}^{\uparrow}$ seal boundary. □

Theorem 126.127 (StreamName finite-window observation factorization)**.** *Every accepted finite observation of a* $\mathsf{StreamName}^{\uparrow}$ *packet that is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *reader and then by a terminal* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal Real seal before any endpoint read is exposed.*

Equivalently, the public observation route is exhausted by the RatStreamNameCarrier *and* RatStreamNameClassifie *rows, the* ProbeBundle *and* InBundle *rows, the* UnaryHistory *observations, the* hsame *transports, the finite* Cont *ledger, the accepted dyadic tolerance entries, the accepted* RegSeqRat *pointwise seal rows, the shared* Pkg *provenance, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row. It exposes no selected limit, quotient stream equality, host real equality, completion object, Classical datum, or standard-bridge datum.*

Proof. First use Definition 126.124. It compares the displayed finite window with every certified refinement and keeps the same exhausted RegSeqRat consumer surface and endpoint boundary.

Next apply Definition 126.125. The public read is forced through the StreamName carrier and classifier rows, bundle membership, unary observations, transports, finite ledger, dyadic tolerance ledger, accepted RegSeqRat rows, provenance, and the endpoint seal.

Finally invoke Definition 126.126. The terminal boundary prevents the endpoint read from opening a selected limit, quotient stream equality, host real equality, completion object, Classical datum, or standard-bridge datum. Carrier projection then leaves exactly the finite observation route listed in the statement. □

Theorem 126.128 (StreamName-RegSeqRat-Real seal obligation factorization)**.** *Let C be a Real-facing completion consumer whose route is displayed through a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface over the same selected rational-history rows, and a* $\mathsf{Real}^{\uparrow}$ *seal ledger row. Then the finite Real-facing seal route factors through exactly the following three obligation blocks:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *block consisting of the rational-history classifier rows,* ProbeBundle *and* InBundle *rows,* UnaryHistory *observations, visible* hsame *transports, finite* Cont *routes, and* Pkg *provenance rows;*
2. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *block consisting of regularity, dyadic-tolerance, transport, stream-spine, and seal-interface rows over the same selected rational-history surface;*
3. *the* $\mathsf{Real}^{\uparrow}$ *block consisting of the seal classifier and its finite ledger boundary row.*

Conversely, any packet carrying exactly those three displayed blocks repacks into the same finite Real-facing seal route.

The factorization exports no quotient stream equality, selected host limit, host real equality, completion witness not carried by the packet, choice schedule, or ambient completeness row.

Proof. First isolate the finite StreamName source. Apply Definition 126.79. It identifies the only rows exported by the public finite-window surface as the selected rational-history classifier

rows, the finite probe ledger, membership rows, unary observations, and the visible transport and provenance data.

Next separate the downstream obligations. Apply Definition 126.82. The separation row assigns the classifier, bundle, membership, unary-history, hsame, Cont, and Pkg data to the StreamName block; the regularity, dyadic-tolerance, transport, and stream-spine data to the RegSeqRat block; and the seal classifier to the RealUp block.

Now compose the seal route. Use Definition 126.83 to pass from those three displayed surfaces to the Real-facing finite-window ledger boundary. This gives the forward factorization of C through the StreamName block, the RegSeqRat block, and the RealUp block.

For the reverse direction, read the same composition theorem from the three blocks back to the public route, and then apply Definition 126.112. The RealUp boundary confirms that the repacked row is the same finite seal route because the seal is read only through the displayed finite ledger boundary.

The exclusion clause follows from the same four boundary theorems. None of them introduces quotient stream equality, selected host limits, host real equality, completion witnesses outside the packet, choice schedules, or ambient completeness rows, so the factorized route exports none of those coordinates. □

Theorem 126.129 (StreamName-RegSeqRat-Real four-face seal obligation factorization)**.** *Let C be a Real-facing completion consumer whose route is displayed through a* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface, a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier readback over the same selected rational-history rows, and a* $\mathsf{Real}^{\uparrow}$ *seal ledger row. Then the finite Real-facing seal route factors through exactly the following four obligation blocks:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *block consisting of the endpoint ledger, radius ledger, dyadic-tolerance rows, and their finite handoff boundary;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *block consisting of the rational-history classifier rows,* ProbeBundle *and* InBundle *rows,* UnaryHistory *observations, visible* hsame *transports, finite* Cont *routes, and* Pkg *provenance rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *block consisting of regularity, classifier, dyadic-tolerance transport, stream-spine, and seal-interface rows over the same selected rational-history surface;*
4. *the* $\mathsf{Real}^{\uparrow}$ *block consisting of the seal classifier and its finite ledger boundary row.*

Conversely, any packet carrying exactly these four displayed blocks repacks into the same finite Real-facing seal route.

The factorization exports no quotient stream equality, selected host limit, host real equality, completion witness not carried by the packet, choice schedule, ambient completeness row, metric-completeness row, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First discard the fourth face and read the visible StreamName-RegSeqRat-Real part of the route. Apply Definition 126.128. It gives the three displayed blocks carried by the StreamName finite-window surface, the RegSeqRat regularity and readback surface, and the RealUp seal boundary, and it also records the exclusions for quotient stream equality, selected host limits, host real equality, external completion witnesses, choice schedules, and ambient completeness rows.

Now insert the DyadicRatCore face at the unique upstream position. Apply Definition 38.738. The route places the endpoint and radius ledger before the StreamName finite-window schedule, before the RegSeqRat regularity and classifier rows, and before the local Real seal. Thus the three-block factorization refines to the four blocks listed in the statement.

The order is forced by Definition 38.739. Any route omitting or permuting one of the four faces either loses a displayed ledger row or leaves the finite packet by adding one of the excluded off-table coordinates. Hence the four-block route is not merely a presentation of C but the exact obligation factorization read by the Real-facing seal.

It remains to compare the shared packet data. Apply Definition 38.740. The DyadicRatCore ledger, StreamName schedule, RegSeqRat classifier and transport rows, and RealUp seal row all determine the same finite (BHist, hsame, Cont, Pkg, NameCert) packet. Therefore the reverse direction repackages any packet carrying exactly those four blocks into the same Real-facing seal route.

The exclusion clause follows from the same three inputs. The dependency route, order uniqueness theorem, and classifier-coherence theorem rule out quotient stream equality, selected host limits, host real equality, external completion witnesses, choice schedules, ambient completeness rows, metric-completeness rows, and a second RealUp seal coordinate. □

126.6 StreamName Real-RegSeq window source exhaustion

Theorem 126.130 (StreamName Real-RegSeq finite-window source exhaustion)**.** *Every finite-window* RegSeqRat *consumer read or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer read whose input is a public* $\mathsf{StreamName}^{\uparrow}$ *packet factors through exactly the public StreamName-RegSeqRat-RealUp source rows already exposed at the handoff boundary. The source inventory consists of the selected* RatHistoryClassifier *rows, the finite* ProbeBundle *row, the corresponding* InBundle *membership rows, the* UnaryHistory *observation rows, the visible* hsame *transports, the finite* Cont *ledger, the shared* Pkg *provenance rows, the accepted* DyadicRatCore *ledger entries, the accepted* RegSeqRat *pointwise seal rows, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row.*

The factorization exports no quotient representative, no completion object, no host index equality, no off-ledger regularity row, no Classical datum, and no standard-bridge datum. Thus any finite public consumer read of the StreamName-RegSeqRat-RealUp handoff is exhausted by the displayed finite-window source inventory.

Proof. First separate the three obligation sources. By Definition 126.82, StreamName contributes only the carrier, selected rational-history classifier, finite-bundle, membership, unary-observation, finite ledger, and public package support rows, while RegSeqRat and RealUp supply their own seal-side obligations.

Next pass from a single window to any certified finite cover. Apply Definition 126.124. It says that cover refinement preserves exactly the displayed StreamName rows, bundle-membership rows, unary observations, hsame transports, finite Cont entries, accepted DyadicRatCore entries, Pkg provenance rows, accepted RegSeqRat pointwise seal rows, and the $\mathsf{Real}^{\uparrow}$ endpoint seal row.

Now identify the RegSeqRat read. Use Definition 126.100. The pointwise handoff consumes the accepted RegSeqRat seal rows through the same finite StreamName packet, so no off-ledger regularity row or hidden host-index equality can be introduced by the consumer read.

Finally identify the RealUp read. By Definition 126.112, the endpoint seal is fixed by the accepted RegSeqRat rows and the visible finite StreamName ledger. Since the preceding steps already exhaust the StreamName and RegSeqRat source rows, the RealUp endpoint cannot reopen quotient representatives, completion objects, Classical data, or standard-bridge data. □

Theorem 126.131 (Real L10 terminal source pullback uniqueness)**.** *Let* R_0 *and* R_1 *be terminal* $\mathsf{Real}^{\uparrow}$ *seal reads whose visible route is the four-face L10 chain* $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$. *Suppose the two reads preserve the same displayed* $\mathsf{StreamName}^{\uparrow}$ *finite*

window, the same DyadicRatCore$^{\uparrow}$ *tolerance row, the same* RegSeqRat$^{\uparrow}$ *readback and regularity rows, the same componentwise* hsame *transports, the same finite* Cont *ledger, the same* Pkg *provenance rows, and the same local* NameCert$_{\mathsf{StreamName}^{\uparrow}}$, NameCert$_{\mathsf{DyadicRatCore}^{\uparrow}}$, NameCert$_{\mathsf{RegSeqRat}^{\uparrow}}$, *and* NameCert$_{\mathsf{Real}^{\uparrow}}$ *support rows.*

Then R_0 and R_1 determine the same terminal finite source pullback over the displayed L10 route. The pullback is the finite BHist *packet whose four faces are the dyadic tolerance ledger, the StreamName finite-window surface, the RegSeqRat readback and regularity surface, and the terminal RealUp seal row. No quotient stream equality, host real equality, selected limit, ambient completeness datum, countable-choice datum, metric-completeness datum, or standard-bridge row is exported by this uniqueness claim.*

Proof. First exhaust the source rows available to each terminal read. Apply Definition 126.130. Each read can consume only the displayed finite StreamName window, the accepted DyadicRatCore ledger entries, the RegSeqRat pointwise seal and regularity rows, the visible hsame transports, Cont rows, Pkg provenance, and local NameCert support.

Next identify the StreamName-to-RegSeqRat face. Use Definition 126.100. Since the two reads expose the same finite window and the same componentwise transports, the RegSeqRat face is read from the same public StreamName packet rather than from a host stream equality or a hidden representative.

Now identify the seal-facing RealUp face. Apply Definition 301.123 and Definition 38.718. These theorems place the terminal readback on the same finite RegSeqRat tail-selector route and the same RealUp finite-window seal, so no selected limit, ambient completeness datum, countable-choice datum, or metric-completeness datum remains as a separate source coordinate.

Finally repack the four displayed faces into one finite pullback. The StreamName seal boundary fixes the finite window, the RegSeqRat boundary fixes the readback and regularity rows, and the RealUp seal boundary fixes the endpoint seal row. Because R_0 and R_1 preserve the same rows on every face, the resulting finite BHist pullback is componentwise identical, and there is no standard-bridge row available to distinguish a second terminal source. □

126.7 StreamName finite-window Real handoff coverage

Theorem 126.132 (StreamName finite-window Real handoff coverage)**.** *Every RegSeqRat-to-*Real$^{\uparrow}$ *handoff read mediated by* StreamName$^{\uparrow}$ *factors through displayed finite windows, the* RegSeqRat *pointwise rows, the* Real$^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. More explicitly, the source surface is the finite inventory of* RatHistoryClassifier *observations,* ProbeBundle *ledgers,* InBundle *membership rows,* UnaryHistory *observations, accepted* RegSeqRat *pointwise seal rows, finite continuation rows, transport rows, provenance rows, and naming rows already displayed by the StreamName handoff.*

The coverage theorem exports no quotient stream, no host sequence equality, no ambient real completeness principle, no completion representative, and no Real$^{\uparrow}$ *seal read outside the finite StreamName and RegSeqRat rows.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFiniteWindowRealHandoffCoverage`

Proof. First read the StreamName source surface. Apply Definition 126.79: a public StreamName handoff exposes only selected rational-history classifier rows, the finite ProbeBundle ledger, InBundle membership rows, and UnaryHistory observations.

Next refine that source pointwise. Apply Definition 126.170; the downstream RegSeqRat or finite-window Real consumer receives exactly the StreamName carrier rows, finite-window classifier rows, certified unary observations, membership rows, and pointwise hsame comparisons. No regularity or seal row is created by this refinement.

Now pass to the Real seal boundary. Apply Definition 126.142. It says that a finite-window $\mathsf{Real}^{\uparrow}$ seal read arising from the StreamName-to-RegSeqRat handoff factors through the StreamName point rows, accepted RegSeqRat seal rows, bundle and membership rows, unary observations, finite Cont ledger, displayed hsame transports, and Pkg provenance.

Finally repack the same finite rows with the local naming certificate. The cited handoff and factorization rows already supply the finite windows, pointwise RegSeqRat rows, Real seal boundary, transport, continuation, and provenance. The StreamName NameCert row names that same finite surface, so a consumer has no route to quotient streams, host sequence equality, ambient real completeness, completion representatives, or a Real seal read outside the displayed packet. □

Theorem 126.133 (StreamName rational-ball window localization)**.** *Any accepted* $\mathsf{StreamName}^{\uparrow}$ *finite window used in a* $\mathsf{Real}^{\uparrow}$ *handoff can be localized through a displayed* $\mathsf{RationalInterval}^{\uparrow}$ *containment row and then through a* $\mathsf{RationalBall}^{\uparrow}$ *center-radius ledger before the read reaches the* $\mathsf{Real}^{\uparrow}$ *seal. Conversely, a displayed* $\mathsf{RationalBall}^{\uparrow}$ *containment row over a* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint window supplies the same finite* $\mathsf{StreamName}^{\uparrow}$ *observation route to the Real seal.*

The localized route carries only finite BHist *observations, interval endpoint-containment rows, ball center and nonnegative-radius rows, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, host sequence equality, selected limit, ambient completeness premise, or hidden Real representative.*

Proof. For the forward route, begin with Definition 126.132. It confines the handoff to a finite StreamName observation window, accepted RegSeqRat pointwise rows, the Real seal boundary, transport rows, continuation rows, provenance, and local naming rows.

Refine that finite observation window through the endpoint discipline of chapter 354. The interval row supplies the displayed containment boundary for the rational observations without adding a completed real, quotient stream, or host equality.

Now pass from the interval row to the ball ledger of chapter 355. The center row, nonnegative-radius row, and containment ledger localize the same observations by a finite rational-radius packet before the handoff reaches the Real seal.

For the converse route, start from the displayed $\mathsf{RationalBall}^{\uparrow}$ containment row over a displayed $\mathsf{RationalInterval}^{\uparrow}$ endpoint window. Reading the center-radius ledger back through the interval containment row gives the same finite rational observation packet. Applying the StreamName coverage theorem again routes that packet through the same finite StreamName observation rows and then to the $\mathsf{Real}^{\uparrow}$ seal, with all excluded quotient, host, selected-limit, and completeness coordinates still absent. □

Theorem 126.134 (StreamName RegSeqRat Real window factorization)**.** *Suppose two* $\mathsf{StreamName}^{\uparrow}$*-mediated* RegSeqRat*-to-*$\mathsf{Real}^{\uparrow}$ *handoff reads use the same finite StreamName window, the same accepted* RegSeqRat *pointwise row, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary. Then the two reads have the same public source surface: the same finite packet of* RatHistoryClassifier *observations,* ProbeBundle *and* InBundle *rows,* UnaryHistory *observations,* RegSeqRat *pointwise seal rows, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

This factorization exposes no second $\mathsf{Real}^{\uparrow}$ *seal, no quotient stream, no host sequence equality, no selected completion representative, and no ambient completeness coordinate.*

Proof. First apply Definition 126.132 to both handoff reads. The two inventories are finite packets whose only public rows are the StreamName window, the RegSeqRat pointwise data, the Real seal boundary, the displayed transport and continuation routes, the provenance row, and the local naming rows.

Next apply Definition 126.170 to the shared accepted pointwise row. It identifies the RegSeqRat portion of both packets with the same StreamName carrier rows, classifier observations, unary observations, membership rows, and componentwise hsame comparisons.

Now apply Definition 126.142 to the shared Real seal boundary. The Real read factors through the same accepted RegSeqRat rows, finite bundle rows, displayed Cont routes, Pkg provenance, and transport rows; it does not create an additional Real-facing source coordinate.

Finally use Definition 126.97. Common-window determinacy identifies the two consumer reads once their finite StreamName window, RegSeqRat row, and Real seal boundary agree. Repacking those common rows under the local NameCert surface gives the stated public packet and rules out any quotient, host-equality, representative, or ambient-completeness exposure. □

Theorem 126.135 (StreamName Real handoff pullback exactness)**.** *A public* $\mathsf{Real}^{\uparrow}$ *seal read mediated by* $\mathsf{StreamName}^{\uparrow}$ *and* RegSeqRat *is accepted exactly when it factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the accepted* RegSeqRat *pointwise row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. No accepted read may replace that pullback surface by quotient streams, host sequence equality, completion representatives, or ambient real completeness.*

Proof. For the forward direction, start from an accepted public $\mathsf{Real}^{\uparrow}$ seal read. Apply Definition 126.132 to expose the finite StreamName window, accepted RegSeqRat pointwise rows, Real seal boundary, transport rows, continuation rows, provenance, and local naming rows.

Next apply Definition 126.134. It identifies the public source surface of the read with the same finite StreamName window, the same accepted RegSeqRat pointwise row, and the same Real seal boundary before any downstream Real consumer can read the packet.

Now apply Definition 126.142. The Real-facing row is therefore reached only through the displayed handoff surface: finite StreamName observations, RegSeqRat pointwise acceptance, the seal boundary, hsame transports, Cont routes, Pkg provenance, and the local NameCert rows.

For the reverse direction, suppose such a finite pullback surface is displayed. Repack the listed rows as the StreamName-mediated RegSeqRat-to-Real handoff packet. The coverage theorem accepts exactly that inventory, the window factorization identifies the public source surface, and the Real seal factorization supplies the downstream seal route.

The excluded objects remain outside both directions. Quotient streams, host sequence equality, completion representatives, and ambient real completeness are not coordinates of the displayed pullback surface, so neither acceptance nor reconstruction can cite them. □

Theorem 126.136 (StreamName finite-window budget monotonicity)**.** *Let* $P \leq Q$ *be finite observation budgets for two* $\mathsf{StreamName}^{\uparrow}$*-mediated* RegSeqRat*-to-*$\mathsf{Real}^{\uparrow}$ *handoff reads. Suppose the* Q*-budget packet restricts on* P *to the same selected StreamName rows, the same accepted* RegSeqRat *pointwise row, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary as the* P*-budget packet. Then enlarging the budget from* P *to* Q *preserves the downstream* $\mathsf{Real}^{\uparrow}$ *seal consumer's public source surface.*

The preserved surface consists only of finite BHist *observations,* ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *observations, pointwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows visible in the restricted packet. The monotone handoff exports no quotient stream, no host sequence equality, no selected limit, and no ambient completeness principle.*

Proof. Apply Definition 126.132 to the P-budget handoff and to the Q-budget handoff. Both consumers are therefore confined to finite StreamName, RegSeqRat, Real-boundary, transport, continuation, provenance, and local naming rows.

Use Definition 126.170 on the restriction of the Q packet to P. The hypothesis identifies the selected StreamName rows and the accepted pointwise RegSeqRat row with the P packet, so the componentwise hsame and unary-history coordinates read by the Real consumer are the same on the restricted window.

Next apply Definition 126.142. The RealUp seal still factors through the displayed finite route: bundle membership, accepted RegSeqRat rows, displayed Cont rows, Pkg provenance, and the local transport rows. The larger budget supplies no additional seal-facing coordinate once its restriction has the same boundary row.

Finally apply Definition 126.134 to the shared restricted window. It identifies the public source surface seen by the downstream RealUp seal consumer, and the identified surface contains exactly the finite BEDC rows named in the statement. Quotient streams, host sequence equality, selected limits, and ambient completeness remain outside the handoff packet. □

Theorem 126.137 (StreamName Real-RegSeq common-window lattice coherence)**.** *Suppose two* StreamName$^{\uparrow}$*-mediated* RegSeqRat*-to-*Real$^{\uparrow}$ *handoff reads are compared through a certified common finite-window lattice meet. If their accepted* RegSeqRat *pointwise rows and their* Real$^{\uparrow}$ *seal boundary agree on that meet, then both reads expose the same finite* BHist *source surface. The common surface consists exactly of the* RatHistoryClassifier *observations,* ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *observations, accepted* RegSeqRat *pointwise seal rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows carried by the meet packet.*

This lattice coherence statement exports no quotient stream equality, no host sequence equality, no ambient Real$^{\uparrow}$ *completeness, no selected completion representative, and no metric-completeness row.*

Proof. First apply Definition 126.132 to each handoff read. Each read is already confined to the finite StreamName window, accepted RegSeqRat pointwise rows, the Real seal boundary, transport rows, continuation rows, provenance, and local naming rows.

Next apply Definition 126.134. The assumed agreement of the accepted pointwise rows and the Real seal boundary identifies the RegSeqRat and Real-facing rows that both reads can expose over the same window.

Now use Definition 126.257. The certified meet is a shared finite observation packet, so every RegSeqRat consumer and every finite-window RealUp consumer reads only its classifier, bundle, membership, unary-history, dyadic ledger, transport, continuation, provenance, and local naming coordinates.

Finally apply Definition 126.97. Common-window determinacy identifies the consumer reads on the certified meet surface. Repacking that identified surface gives the same finite BHist source packet for both handoffs and leaves quotient equality, host sequence equality, ambient Real completeness, selected representatives, and metric-completeness rows outside the public lattice surface. □

Theorem 126.138 (StreamName Real seal observation-budget factorization)**.** *Any finite-window* StreamName$^{\uparrow}$ *packet whose read is consumed by* RegSeqRat$^{\uparrow}$ *and sealed as* Real$^{\uparrow}$ *factors through the displayed observation-budget limiter route whenever an accepted* ObservationBudgetLimiter$^{\uparrow}$ *packet is present. The common route is the request row, budget row, cap row, window row, dyadic ledger,* RegSeqRat *readback,* Real$^{\uparrow}$ *seal, and structural rows named by the limiter surface of chapter 903.*

This factorization exports no uncapped schedule, hidden selector, quotient stream, host real equality, selected limit, independent real carrier, or ambient completeness principle.

Proof. Start with the finite-window handoff coverage. By Definition 126.132, the StreamName-to-Real route already exposes only finite StreamName observations, RegSeqRat pointwise rows, the Real seal boundary, transport rows, continuation rows, provenance, and local naming rows.

Refine the route through Definition 126.134. The RegSeqRat readback and Real seal have the same public source surface whenever the finite window, accepted pointwise row, and seal boundary agree.

Now use Definition 126.137. The common-window lattice pins the classifier observations, bundle rows, membership rows, unary-history observations, dyadic ledger, transports, routes, provenance, and local names to one finite observation packet.

When an accepted $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet is present, its request, budget, cap, and window rows are exactly the displayed limiter boundary through which that finite observation packet is read. Since neither the StreamName handoff nor the limiter carrier contains an uncapped schedule, hidden selector, quotient stream, host real equality, selected limit, independent real carrier, or ambient completeness coordinate, none of those objects can be exported by the factorization. □

Theorem 126.139 (StreamName Real observation-budget route)**.** *Any finite-window* $\mathsf{StreamName}^{\uparrow}$ *packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and sealed as* $\mathsf{Real}^{\uparrow}$*, when read by a* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer, factors through the request, budget, and window route of chapter 611. The route first exposes the finite StreamName surface, the* RegSeqRat *pointwise row, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and then repacks them as the* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet*

$$E, W, D, R, S, H, C, P, N$$

over the precision request, selected finite windows, dyadic readbacks and radii, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows.*

This route exports no uncapped schedule, quotient stream, host sequence equality, selected limit, independent real carrier, ambient completeness principle, or host real equality.

Proof. Begin with the finite-window handoff coverage. By Definition 126.132, the StreamName-to-Real packet already exposes only finite StreamName observations, RegSeqRat pointwise rows, the $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont routes, Pkg provenance, and local NameCert rows.

Next apply Definition 126.134. It identifies the public source surface of the consumed packet with the same finite StreamName window, the same accepted RegSeqRat pointwise row, and the same $\mathsf{Real}^{\uparrow}$ seal boundary before the observation-budget consumer reads it.

Now use Definition 126.137. The common finite-window lattice pins the classifier observations, bundle rows, membership rows, unary-history observations, dyadic ledger, transports, routes, provenance, and local names to one finite observation packet.

Apply Definition 126.138 to pass that packet through the displayed observation-budget limiter boundary. The limiter supplies the request, budget, cap, and window discipline needed before a bounded real-facing consumer can inspect the StreamName handoff.

Finally route the bounded packet through chapter 611. Its carrier Definition 611.1 and non-escape theorem Definition 611.6 repack the same finite surface as $E, W, D, R, S, H, C, P, N$, with componentwise hsame, displayed Cont, Pkg, and the local $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ row. Since

neither the source handoff nor the observation-budget carrier contains an uncapped schedule, quotient stream, host sequence equality, selected limit, independent real carrier, ambient completeness row, or host real equality row, the routed consumer cannot export any of those objects. □

Theorem 126.140 (StreamName Real terminal factorization)**.** *Any finite-window* $\mathsf{StreamName}^{\uparrow}$ *packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*, sealed as* $\mathsf{Real}^{\uparrow}$*, and then read by a Real-facing consumer factors through the terminal source lock of Definition 126.145. The common terminal packet is exactly the finite* BHist *window, accepted* RegSeqRat *pointwise row, induced* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

The factorization exports no quotient stream, host sequence equality, selected limit, ambient real completeness, independent Real carrier, or metric-completeness row.

Proof. First project the finite StreamName rows. By Definition 126.132, the handoff read exposes only finite classifier observations, bundle and membership rows, unary observations, pointwise RegSeqRat rows, the $\mathsf{Real}^{\uparrow}$ seal boundary, transport rows, continuation rows, provenance, and local naming rows.

Next identify the public source surface by Definition 126.134. The finite StreamName window, accepted RegSeqRat pointwise row, and $\mathsf{Real}^{\uparrow}$ seal boundary determine the same surface for every downstream Real-facing read.

Apply Definition 126.135. The accepted read is therefore a pullback over exactly the finite window, pointwise row, seal boundary, hsame transports, Cont routes, Pkg provenance, and local NameCert rows.

Now route the bounded consumer through Definition 126.139. The Real-facing observation request reads the same finite packet through $\mathsf{RealObservationBudget}^{\uparrow}$, so the request, budget, and window discipline do not introduce an additional source coordinate.

Finally apply Definition 126.146. The terminal lock identifies the common source surface with the lock packet of Definition 126.145: the finite BHist window, hsame, Cont, Pkg, and NameCert rows together with the accepted RegSeqRat and induced $\mathsf{Real}^{\uparrow}$ seal rows. Since none of the cited routes carries quotient streams, host sequence equality, selected limits, ambient completeness, an independent Real carrier, or metric-completeness data, none is exported by the terminal factorization. □

Theorem 126.141 (StreamName Real terminal quotient refusal)**.** *Any finite-window* $\mathsf{StreamName}^{\uparrow}$ *packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*, sealed as* $\mathsf{Real}^{\uparrow}$*, and then read by a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *consumer factors first through the terminal source lock of Definition 126.145 and then through the refusal ledger of the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *route. The source-facing part of the read is exactly the finite* BHist *window, accepted* RegSeqRat *pointwise row, induced* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row already locked by the terminal source packet.*

Requests for quotient stream equality, host sequence equality, selected limits, hidden tail oracles, ambient completeness, or metric-completeness rows are routed to refusal rather than exported as completion coordinates.

Proof. Start with the terminal factorization. By Definition 126.140, every Real-facing read of the finite StreamName-to-RegSeqRat-to-Real packet has the common source surface consisting only of the finite window, accepted pointwise row, induced seal row, componentwise transport, displayed continuation, provenance, and local naming rows.

Identify that source surface with the explicit lock packet of Definition 126.145. The lock contains no quotient stream, host sequence equality, selected limit, hidden tail oracle, ambient

completeness row, independent Real carrier, or metric-completeness coordinate, so none of those objects is available before the refusal consumer is consulted.

Now pass the locked packet to the QuotientStreamRefusal$^{\uparrow}$ consumer. Apply Definition 998.3: a completion-facing request that leaves the displayed finite route is sent to the refusal row and exports no completion coordinate.

Finally use Definition 998.9. The terminal refusal route exhausts the accepted read by the displayed route and refusal rows, with transport, replay, provenance, and naming attached to that route. Thus the StreamName terminal packet reaches the refusal ledger, while quotient equality, host equality, selected limits, hidden tail data, ambient completeness, and metric-completeness rows remain refused rather than exported. □

126.8 StreamName Real seal factorization boundary

Theorem 126.142 (StreamName Real seal factorization boundary)**.** *Let W be a finite public* StreamName$^{\uparrow}$ *rational observation window. Suppose that W is read through certified unary indices, that an accepted* RegSeqRat *pointwise seal package is displayed over the same certified indices, and that a finite-window* Real$^{\uparrow}$ *seal read arises from that StreamName-to-*RegSeqRat *handoff. Then the* Real$^{\uparrow}$ *seal read factors through exactly the following rows: the StreamName point rows over W, the accepted* RegSeqRat *seal rows, the* ProbeBundle *and* InBundle *membership rows, the* UnaryHistory *observation rows, the finite* Cont *ledger, the displayed* hsame *transports, and the* Pkg *provenance naming the source certificates.*

Conversely, any finite packet consisting of precisely those rows repacks as the same public finite-window Real$^{\uparrow}$ *seal read. The factorization contains no completion data, quotient representative, host index equality, Classical choice, or* Real$^{\uparrow}$ *completeness datum outside the displayed* RegSeqRat *seal rows.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealSealFactorizationBoundary`

Proof. First read the public Real$^{\uparrow}$ boundary through Definition 126.112. This identifies every finite-window endpoint read arising from the handoff with the StreamName source rows, the accepted RegSeqRat pointwise rows, bundle membership, unary observations, finite ledger entries, hsame transports, and provenance rows.

Next use Definition 126.111. It prevents nested finite windows from changing the seal surface: restricting or refining W keeps the same displayed point rows, the same accepted RegSeqRat seal rows, and the same finite Cont/Pkg provenance on the certified indices.

The common seal cover is supplied by Definition 126.103. The StreamName window, the RegSeqRat seal packet, and the Real$^{\uparrow}$ endpoint read agree only through the listed rows, so any public Real$^{\uparrow}$ read from the handoff has the announced factorization.

Finally apply Definition 126.100. The pointwise handoff exhausts the downstream seal rows by the certified StreamName observations and accepted RegSeqRat rows. Since none of the cited boundary theorems introduces completion data, quotient representatives, host index equality, Classical choice, or a Real$^{\uparrow}$ completeness field, the reverse packing uses exactly the displayed rows and no further datum. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameRealSealFactorizationBoundary`

Theorem 126.143 (StreamName Real seal factorization refinement stability)**.** *Let $W \subseteq W'$ be finite public* StreamName$^{\uparrow}$ *rational observation windows related by displayed prefix refinement. Suppose both windows carry accepted* RegSeqRat *seal rows over the certified unary indices, and suppose*

the source tuple for W' restricts to the source tuple for W through the same ProbeBundle*,* InBundle*,* UnaryHistory*,* Cont*,* hsame*, and* Pkg *rows. Then the finite-window* $\mathsf{Real}^{\uparrow}$ *seal factorization over W is exactly the restriction of the factorization over W'.*

The restricted factorization contains no completion datum, quotient representative, host index equality, choice row, or $\mathsf{Real}^{\uparrow}$ *completeness field beyond the displayed accepted* RegSeqRat *seal rows.*

Proof. Apply Definition 126.142 to both finite windows. Each factorization is read through the StreamName point rows, the accepted RegSeqRat seal rows, bundle membership, unary observations, finite continuation ledger, displayed hsame transports, and package provenance.

Use Definition 126.111 on the prefix refinement $W \subseteq W'$. It identifies the seal rows on the smaller window with the restriction of the rows displayed on the larger window.

It remains to identify the source tuple. The determinacy theorem Definition 126.149 says the restricted source is fixed by the displayed StreamName, bundle, history, transport, continuation, and provenance rows. The minimality theorem Definition 126.150 rules out any extra source coordinate outside those rows.

Therefore restricting the W' factorization along the displayed prefix refinement gives exactly the factorization obtained directly over W, with no additional completion, quotient, host-equality, choice, or completeness data. □

Theorem 126.144 (StreamName Real finite-window classifier exactness)**.** *Let W and W' be finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation windows over certified unary indices. Suppose the two windows display the same* ProbeBundle *rows, the same* InBundle *membership rows, the same* UnaryHistory *observations, the same pointwise rational classifier rows, the same accepted* RegSeqRat *seal rows, the same finite* Cont *ledger, the same displayed* hsame *transports, and the same* Pkg *provenance naming the source certificates. Then the finite-window* $\mathsf{Real}^{\uparrow}$ *seal classifier induced by either window is exactly the transport of the* RegSeqRat *classifier read from those displayed rows.*

In particular, no completion representative, quotient equality, hidden host index equality, choice datum, or $\mathsf{Real}^{\uparrow}$ *completeness field can change that classifier once the listed finite StreamName and* RegSeqRat *rows agree.*

Proof. First apply Definition 126.142. The boundary theorem says that every finite-window $\mathsf{Real}^{\uparrow}$ seal classifier arising from the StreamName-to-RegSeqRat handoff is read only through the listed StreamName rows, accepted RegSeqRat seal rows, bundle membership, unary history, continuation ledger, hsame transports, and package provenance.

Next apply Definition 126.143. Any common finite presentation or prefix refinement of W and W' restricts to the same displayed factorization, so changing the window while preserving the listed rows cannot change the induced $\mathsf{Real}^{\uparrow}$ classifier.

The classifier handoff is the equality supplied by Definition 126.268. Since the two windows expose the same pointwise rational classifier rows and the same accepted RegSeqRat seal rows, the transported RegSeqRat classifier is the single classifier available on the common finite surface.

Finally, Definition 126.149 fixes the source tuple from the displayed StreamName, bundle, history, transport, continuation, and provenance rows, and Definition 126.150 rules out any extra source coordinate. Hence the finite-window $\mathsf{Real}^{\uparrow}$ seal classifier is exactly the transported RegSeqRat classifier read from those rows, with no escape through completion, quotient, host-equality, choice, or completeness data. □

126.9 StreamName terminal source lock

Definition 126.145 (StreamName-RegSeqRat-RealUp terminal source lock)**.** *A* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$–$\mathsf{Real}^{\uparrow}$ *terminal source lock is the finite packet*

$$\mathcal{L} = (W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *W is a finite public* $\mathsf{StreamName}^{\uparrow}$ *window over certified unary observations, including the selected rational-history classifier rows, the finite* ProbeBundle *ledger,* InBundle *membership rows, and* UnaryHistory *observation rows;*
2. *R is the accepted* RegSeqRat *finite seal row over the same certified indices, including the regularity, dyadic-tolerance, stream-spine, and pointwise seal classifier rows supplied on the* RegSeqRat *side;*
3. *S is the finite-window* $\mathsf{Real}^{\uparrow}$ *seal row induced by reading W through R;*
4. *H is the displayed componentwise* hsame *row comparing the StreamName point rows, the* RegSeqRat *seal rows, the unary observation rows, the bundle-membership rows, and the induced* $\mathsf{Real}^{\uparrow}$ *seal row;*
5. *C is the displayed* Cont *ledger routing the read from the finite StreamName window through the accepted* RegSeqRat *seal row to the finite* $\mathsf{Real}^{\uparrow}$ *seal row;*
6. *P is the* Pkg *provenance naming the StreamName source certificate, the* RegSeqRat *finite seal certificate, the finite continuation ledger, and the induced* $\mathsf{Real}^{\uparrow}$ *seal read;*
7. *N is the local* NameCert *row for this finite handoff packet.*

The packet has no coordinate for a completion witness, quotient stream equality, host index equality, or off-ledger $\mathsf{Real}^{\uparrow}$ *classifier row.*

Theorem 126.146 (StreamName-RegSeqRat-RealUp terminal source lock)**.** *Every finite* $\mathsf{Real}^{\uparrow}$ *seal read induced from a public* $\mathsf{StreamName}^{\uparrow}$ *window through an accepted* RegSeqRat *finite seal factors through exactly the packet of Definition 126.145. The StreamName coordinates are read from W, the regularity and pointwise seal coordinates from R, the induced finite* $\mathsf{Real}^{\uparrow}$ *seal row from S, and the transport, continuation, provenance, and local naming rows from H, C, P, and N.*

Conversely, any finite packet carrying precisely those rows repacks as the same induced finite-window $\mathsf{Real}^{\uparrow}$ *seal read.*

Proof. First apply Definition 126.82. It separates the visible StreamName source rows, the downstream RegSeqRat regularity and seal rows, and the finite $\mathsf{Real}^{\uparrow}$ seal classifier into their distinct source surfaces.

Next apply Definition 126.111. The accepted RegSeqRat seal is stable under finite-window restriction, so the source window and the seal row keep the same displayed certified indices and the same pointwise classifier rows.

Finally apply Definition 126.142. It states that the finite $\mathsf{Real}^{\uparrow}$ seal read factors through exactly the StreamName point rows, accepted RegSeqRat seal rows, bundle membership, unary observations, finite Cont ledger, displayed hsame transports, and Pkg provenance. These rows are precisely the coordinates of $\mathcal{L}$, with N recording the local naming ledger, and no further coordinate is available for the reverse packing. □

Theorem 126.147 (StreamName-RegSeqRat-RealUp terminal non-escape)**.** *Let $\mathcal{L} = (W, R, S, H, C, P, N)$ be a terminal source lock. A consumer of the finite handoff can read only the finite StreamName window W, the accepted* RegSeqRat *seal row R, the induced finite* $\mathsf{Real}^{\uparrow}$ *seal row S, the displayed* hsame *transports H, the displayed* Cont *routes C, the* Pkg *provenance P, and the local* NameCert *row N. It cannot extract a completion witness, quotient stream equality, host index equality, or off-ledger* $\mathsf{Real}^{\uparrow}$ *classifier row from the lock.*

Proof. Project the consumer read through Definition 126.146. Every induced finite $\mathsf{Real}^{\uparrow}$ seal read is already factored by the displayed lock packet.

The separation theorem Definition 126.82 assigns each visible row to StreamName, RegSeqRat, or finite $\mathsf{Real}^{\uparrow}$ seal data, and it excludes completion witnesses, quotient rows, and host index equalities from the StreamName source surface.

The boundary theorem Definition 126.142 rules out any completion datum, quotient representative, host index equality, choice row, or $\mathsf{Real}^{\uparrow}$ completeness coordinate outside the displayed accepted RegSeqRat seal rows. The uniform finite-window theorem prevents a restriction of the lock from adding an off-ledger classifier row.

Thus the terminal packet is a finite source surface only. Every consumer read stays inside W, R, S, H, C, P, and N. □

Theorem 126.148 (StreamName-RegSeqRat-Real terminal observation coverage)**.** *Let $\mathcal{L} = (W, R, S, H, C, P, N)$ be a terminal source lock. Suppose a certified finite subwindow or finite refinement displays StreamName point rows,* RegSeqRat *seal rows, the induced* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and a local* NameCert *row agreeing with the corresponding restriction of $\mathcal{L}$. Then the induced finite-window* $\mathsf{Real}^{\uparrow}$ *read has the same terminal observation surface as $\mathcal{L}$ on that restricted packet.*

Conversely, every terminal observation read obtained from such a certified subwindow or refinement is covered by the parent lock: it reads only the restricted StreamName window, restricted RegSeqRat *seal rows, restricted induced* $\mathsf{Real}^{\uparrow}$ *seal row, restricted* hsame *and* Cont *rows, restricted* Pkg *provenance, and the restricted local* NameCert *row. The coverage exports no completion witness, no quotient stream equality, no host index equality, no off-ledger* $\mathsf{Real}^{\uparrow}$ *classifier row, and no ambient completeness coordinate.*

Proof. First apply Definition 126.146. It identifies each terminal read with exactly the lock packet coordinates W, R, S, H, C, P, N, so agreement with a restriction of $\mathcal{L}$ fixes the same terminal observation surface on the restricted packet.

Next apply Definition 126.132 and Definition 126.79. These results identify the finite StreamName observations, accepted RegSeqRat rows, finite $\mathsf{Real}^{\uparrow}$ seal row, transport rows, continuation routes, provenance, and local naming rows as the complete handoff surface for certified finite windows.

For refinements, use Definition 126.240 and Definition 126.249. They say that common-refinement and synchronized-window reads restrict to the same displayed finite rows and do not create a second Real-facing source coordinate.

Finally apply Definition 126.147. Since the parent lock has no coordinate for completion witnesses, quotient stream equality, host index equality, off-ledger $\mathsf{Real}^{\uparrow}$ classifier rows, or ambient completeness, neither a certified subwindow nor a finite refinement can export one. Thus every terminal observation read from such a packet is already covered by the parent lock. □

Theorem 126.149 (StreamName RegSeqRat RealUp seal source determinacy)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window, and let $\mathcal{E}_0$ and $\mathcal{E}_1$ be finite-window* $\mathsf{Real}^{\uparrow}$ *seal*

packages induced from W. Suppose both packages read the same accepted RegSeqRat *spine over W, the same DyadicRatCore endpoint ledger, the same* ProbeBundle *and* InBundle *rows, the same unary observation rows, the same finite* Cont *ledger, the same displayed* hsame *endpoint transports, and the same* Pkg *provenance. Then $\mathcal{E}_0$ and $\mathcal{E}_1$ have the same visible seal source.*

That source is exactly the public StreamName finite-window row, the accepted RegSeqRat *finite-window seal row, the dyadic endpoint rows, and the shared ledger boundary. It contains no completion row, quotient row, host index equality, or off-ledger* $\mathsf{Real}^{\uparrow}$ *classifier.*

Proof. First exhaust the scheduled seal package. Apply Definition 126.86. Both induced packages project to the same scheduled StreamName row, accepted RegSeqRat finite-window seal row, and finite-window $\mathsf{Real}^{\uparrow}$ ledger boundary, because the listed public window, spine, dyadic endpoint, bundle, transport, and provenance rows agree.

Next compare the pointwise seal rows. Use Definition 126.89 at each certified unary observation of W. The shared DyadicRatCore endpoint ledger fixes the pointwise seal classifier rows under the displayed hsame transports, so the two packages cannot choose different visible rational endpoints.

Now pass from pointwise rows to the whole finite window. By Definition 126.111, the public StreamName window, the scheduled RegSeqRat seal, and every nested handoff select one finite observation packet with the same ProbeBundle, Cont, hsame, and Pkg rows.

Finally apply Definition 126.112. Every finite-window $\mathsf{Real}^{\uparrow}$ seal read arising from this handoff factors through the same StreamName row, accepted RegSeqRat seal row, dyadic endpoint rows, and shared ledger boundary. Since none of the cited boundary theorems exports completion rows, quotient rows, host index equality, or off-ledger classifiers, the visible seal source of $\mathcal{E}_0$ and $\mathcal{E}_1$ is determined by the stated handoff data. □

Theorem 126.150 (StreamName RegSeqRat RealUp seal source minimality)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *rational observation window, and let $\mathcal{E}$ be a finite-window* $\mathsf{Real}^{\uparrow}$ *seal package induced from W and an accepted* RegSeqRat *spine. Then every visible source row used by $\mathcal{E}$ is one of the public StreamName finite-window row, the accepted* RegSeqRat *finite-window seal row, the dyadic endpoint ledger, the* ProbeBundle *and* InBundle *rows, the unary history rows, the finite* Cont *ledger, the displayed* hsame *endpoint transports, or the* Pkg *provenance.*

In particular, no completion carrier, quotient representative, host-index equality, or off-ledger $\mathsf{Real}^{\uparrow}$ *classifier is a source row of the induced finite-window seal package.*

Proof. Start with Definition 126.149 applied to two reads of the same induced package. The determinacy statement identifies the visible source with the public StreamName finite-window row, the accepted RegSeqRat finite-window seal row, the dyadic endpoint rows, and the shared ledger boundary.

Use Definition 126.86 to expand the scheduled seal boundary. The expansion lists only the accepted spine rows, bundle rows, unary observation rows, continuation rows, endpoint transports, and provenance named in the statement.

Finally apply Definition 126.112. Its boundary factorization is through exactly those finite-window rows, so any alleged source outside the listed tuple would have to enter through a completion, quotient, host-index, or off-ledger classifier row. The boundary theorem exports none of those rows, establishing the stated minimality. □

Theorem 126.151 (StreamName RegSeqRat RealUp seal unique source)**.** *Let $\mathcal{E}_0$ and $\mathcal{E}_1$ be finite-window* $\mathsf{Real}^{\uparrow}$ *seal packages induced from public StreamName windows and accepted* RegSeqRat *spines. Suppose they agree on exactly the minimal source tuple consisting of the public StreamName finite-window row, accepted* RegSeqRat *finite-window seal row, dyadic endpoint ledger,* ProbeBundle

and InBundle *rows, unary history rows, finite* Cont *ledger, displayed* hsame *endpoint transports, and* Pkg *provenance. Then every downstream* RegSeqRat *or* $\mathsf{Real}^{\uparrow}$ *finite-window consumer sees the same visible seal source from* $\mathcal{E}_0$ *and* $\mathcal{E}_1$*.*

Proof. First apply Definition 126.150 to both packages. Each visible source is confined to the same finite tuple of public window, accepted seal, endpoint, bundle, history, continuation, transport, and provenance rows.

The assumed agreement on that tuple supplies the hypotheses of Definition 126.149. Hence the two packages have the same visible seal source.

Any downstream finite-window RegSeqRat or $\mathsf{Real}^{\uparrow}$ consumer reads through that visible source tuple and not through completion, quotient, host-index, or off-ledger classifier data. Therefore the consumer observations obtained from $\mathcal{E}_0$ and $\mathcal{E}_1$ coincide on the visible seal source. □

Theorem 126.152 (StreamName RegSeqRat RealUp finite-window pullback criterion)**.** *Let* S *and* T *be finite* $\mathsf{StreamName}^{\uparrow}$ *handoffs accepted by a* RegSeqRat *finite-window seal consumer and by a* $\mathsf{Real}^{\uparrow}$ *finite-window seal consumer. They classify as the same consumer-visible packet precisely when the following finite source rows agree componentwise:*

1. *the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows;*
2. *the accepted* RegSeqRat *finite-window seal rows;*
3. *the DyadicRatCore endpoint ledgers;*
4. *the displayed* ProbeBundle *and* InBundle *rows;*
5. *the* UnaryHistory *observation rows;*
6. *the finite* Cont *ledger;*
7. *the displayed* hsame *endpoint transports;*
8. *the* Pkg *provenance rows.*

Equivalently, those rows form the finite pullback packet seen by both consumers. The criterion has no coordinate for completion rows, quotient rows, host index equality, off-ledger $\mathsf{Real}^{\uparrow}$ *classifiers, arithmetic packages, order packages, or metric-completeness rows.*

Proof. For the forward direction, assume S and T classify as the same consumer-visible packet. Apply Definition 126.149 to the two finite seal reads. It identifies the visible source with the public StreamName window, accepted RegSeqRat seal row, dyadic endpoint ledger, bundle rows, unary observations, continuation rows, endpoint transports, and package provenance.

Next use Definition 126.150. Since every visible source row of either handoff lies in that finite tuple, classification of the consumer-visible packet projects to componentwise agreement on each listed row.

For the reverse direction, suppose the listed rows agree componentwise. Apply Definition 126.151. The common tuple determines the same visible seal source for every downstream RegSeqRat or $\mathsf{Real}^{\uparrow}$ finite-window consumer.

Now apply Definition 38.413. The public StreamName rows, dyadic endpoint ledgers, accepted regular-rational seal rows, and RealUp seal provenance are exactly the four open-phase faces, and the displayed hsame, Cont, Pkg, and certificate coordinates are the gluing data. Hence the componentwise agreement repacks into one finite pullback packet.

Finally, the same four cited theorems describe all exported source coordinates. None exports completion data, quotient representatives, host index equality, off-ledger $\mathsf{Real}^{\uparrow}$ classifier rows, arithmetic or order packages, or metric-completeness rows, so these data are not part of the criterion. □

Theorem 126.153 (StreamName RegSeqRat RealUp pullback stability)**.** *Let S and T be two finite-window handoff packets addressed to the same class of* RegSeqRat *seal consumers and finite-window* $\mathsf{Real}^{\uparrow}$ *consumers. Suppose the packet comparison transports componentwise the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, accepted* RegSeqRat *finite-window seal rows, DyadicRatCore endpoint ledgers, displayed* ProbeBundle *and* InBundle *rows,* UnaryHistory *observations, finite* Cont *ledgers, displayed* hsame *endpoint transports, and* Pkg *provenance rows. Then S and T determine the same consumer-visible finite pullback packet for every* RegSeqRat *consumer and for every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer.*

This stability statement has no coordinate for completion rows, quotient rows, host index equality, off-ledger $\mathsf{Real}^{\uparrow}$ *classifiers, arithmetic packages, order packages, or metric-completeness rows.*

Proof. Apply Definition 126.152. The criterion reduces equality of the consumer-visible finite pullback packet to equality of the finite source rows listed in its eight public families.

The hypotheses supply exactly those identifications. The transported public $\mathsf{StreamName}^{\uparrow}$ rows identify the visible finite window; the accepted RegSeqRat rows and DyadicRatCore endpoint ledgers identify the regular rational seal boundary; the displayed bundle, unary-history, continuation, endpoint-transport, and provenance rows identify the remaining finite gluing data.

It remains only to exclude extra consumer coordinates. The exclusion clause of the same criterion states that completion data, quotient representatives, host index equality, off-ledger $\mathsf{Real}^{\uparrow}$ classifier rows, arithmetic or order packages, and metric-completeness rows are not exported by the finite pullback packet. Hence no additional row can distinguish the two transported packets for a RegSeqRat or finite-window $\mathsf{Real}^{\uparrow}$ consumer. □

Definition 126.154 (StreamName RegSeqRat reindexed window package)**.** *A* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window package is a finite public handoff row with the following displayed components:*

1. *a finite-window* $\mathsf{StreamName}^{\uparrow}$ *row over rational histories, with its public carrier and selected* RatStreamNameClassifier *row;*
2. *a certified unary reindexer ρ between the source unary window and the consumer unary window;*
3. *the reindexed* RatHistoryClassifier *rows obtained by reading the selected StreamName observations through ρ;*
4. *the finite* ProbeBundle *ledger, the* InBundle *membership rows, and the* UnaryHistory *observation rows for both the source window and the reindexed consumer window;*
5. *the downstream* RegSeqRat *finite-window seal input rows whose source side is exactly the reindexed rational-history classifier rows and their displayed bundle membership evidence.*

The package has no hidden carrier. Its public surface is only the finite StreamName row, the unary reindexer, the reindexed rational-history classifier rows, the bundle and membership rows, the unary observation rows, and the displayed downstream finite-window input rows.

Theorem 126.155 (StreamName RegSeqRat reindexed window exhaustion)**.** *Every consumer row of the reindexed window package of Definition 126.154 is projected from one of the following three sources: the StreamName reindexing and map-stability row of Definition 126.19, the finite-window reindexed map coverage row of Definition 126.24, or the finite-window RegSeqRat handoff row of Definition 126.79. Conversely, those three rows repack into the displayed package and determine its whole consumer surface.*

Proof. First project the reindexing spine. Apply Definition 126.19 to the certified unary reindexer ρ. This gives the transported StreamName carrier row and the selected rational-history classifier rows after reindexing.

Next project the finite window. Apply Definition 126.24. It gives the matching ProbeBundle, InBundle, and UnaryHistory rows on the source and consumer windows, and it identifies the reindexed classifier rows as the only rows visible on the consumer window.

Finally project the downstream handoff. Apply Definition 126.79. It says that the RegSeqRat finite-window consumer reads exactly the selected rational-history classifier rows together with the displayed bundle, membership, and unary observation rows.

For the converse direction, repack those three projections in the order listed in the definition: the public finite-window StreamName row, the certified unary reindexer, the reindexed rational-history classifier rows, the bundle ledger rows, and the downstream finite-window input rows. No additional row is available after these projections, so the repacked package is exhaustive. □

Theorem 126.156 (StreamName RegSeqRat reindexed window imports no seal)**.** *The reindexed window package of Definition 126.154 exports no* RegSeqRat *regularity witness, no* $\mathsf{Real}^{\uparrow}$ *seal classifier, no completion object, no host index equality, and no quotient row. Any such row must be supplied by the downstream* RegSeqRat *or* $\mathsf{Real}^{\uparrow}$ *certificate rather than by the finite reindexed StreamName window.*

Proof. Read the package by the exhaustion theorem Definition 126.155. Its three sources are the reindexing spine, the finite-window reindexed map coverage, and the finite-window StreamName handoff.

The first two sources only transport and cover StreamName observation rows. They mention the unary reindexer, rational-history classifier rows, ProbeBundle membership, and UnaryHistory observations.

The handoff source is separated by Definition 126.79. That theorem places regularity, seal classifiers, completion objects, host equalities, and quotient rows outside the finite StreamName source.

Therefore projecting and repacking the package cannot introduce any of those excluded rows. They remain obligations of the downstream consumer surface. □

Theorem 126.157 (StreamName RegSeqRat reindexed window consumer determinacy)**.** *Let* Ω *and* Ω' *be two* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window packages. Suppose they have the same public StreamName row, the same certified unary reindexer* ρ*, the same selected rational-history classifier rows, and the same finite-window handoff rows after the displayed* hsame *transports. Then every downstream* RegSeqRat *finite-window consumer reads the same source surface and the same selected finite-window input rows from* Ω *and* Ω'*. No regularity witness,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion datum, quotient row, or host index equality can distinguish the two packages.*

Proof. First apply Definition 126.19 to the shared reindexer ρ. The common StreamName source row and the displayed hsame transports identify the reindexed rational-history classifier rows seen by the consumer window.

Next apply Definition 126.24. The selected classifier rows determine the finite ProbeBundle, InBundle, and UnaryHistory rows available on the reindexed window. Thus the two packages expose the same selected finite-window input rows.

Now apply Definition 126.79. The downstream RegSeqRat boundary reads only the reindexed rational-history classifier rows together with the displayed bundle, membership, unary observation, and handoff rows.

Finally use Definition 126.156. It excludes regularity witnesses, $\mathsf{Real}^{\uparrow}$ seal classifiers, completion data, quotient rows, and host index equalities from the package surface, so no extra row can separate Ω from Ω' for a finite-window consumer. □

Theorem 126.158 (StreamName RegSeqRat reindexed window identity handoff)**.** *Assume that the certified unary reindexer in a* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$ *reindexed window package is the identity on every displayed* UnaryHistory *observation. Then the package of Definition 126.154 exposes exactly the unreindexed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row and the same* RegSeqRat *finite-window handoff rows. It exposes no additional regularity row, seal row, completion row, quotient row, or host index-equality row.*

The associated ten-field fit is:

1. *the traditional target is identity reindexing for countable names;*
2. *the BEDC source is* $\mathsf{StreamName}^{\uparrow}$ *together with the* RegSeqRat *finite-window handoff;*
3. *the carrier is the finite rational-observation window;*
4. *the classifier consists of the selected* RatHistoryClassifier *rows;*
5. *stability is identity reindexing together with the displayed* hsame *transports;*
6. *the ledger consists of the* ProbeBundle, InBundle, *and* UnaryHistory *rows;*
7. *the core theorem is this identity handoff;*
8. *exactness is equality with the same finite-window rows;*
9. *the ordinary bridge is finite prefix readback;*
10. *the status is a paper theorem awaiting a formal target.*

Proof. Apply Definition 126.19 to the certified unary reindexer. Under the identity assumption on every displayed UnaryHistory observation, the transported StreamName carrier row and the source carrier row have the same finite observations after the displayed hsame transports.

Apply Definition 126.24. The finite-window coverage row identifies the reindexed RatHistoryClassifier entries with the selected entries already read from the unreindexed StreamName window. The ProbeBundle, InBundle, and UnaryHistory ledger rows are therefore the same displayed finite rows.

Next use Definition 126.79. The RegSeqRat finite-window consumer reads only those selected rational-history classifier rows together with the visible bundle, membership, unary-observation, and handoff rows, so the identity reindexer supplies exactly the unreindexed handoff surface.

The exhaustion theorem Definition 126.155 shows that no other consumer row is present in the package. Finally Definition 126.156 excludes regularity, seal, completion, quotient, and host index-equality rows. Hence the identity case is precisely the unreindexed finite-window handoff. □

Theorem 126.159 (StreamName RegSeqRat reindexed selected-classifier handoff)**.** *For a* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window package, the reindexed selected* RatHistoryClassifier *rows, the finite* ProbeBundle *ledger, the* InBundle *membership rows, and the* UnaryHistory *observations are exactly the common finite-window source surface consumed by both the downstream* RegSeqRat *finite-window seal row and the finite-window* $\mathsf{Real}^{\uparrow}$ *ledger-boundary row. No* RegSeqRat *regularity witness,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion datum, quotient datum, or host index equality is imported into that common surface.*

Proof. First apply Definition 126.155. It identifies the package surface with the public StreamName finite-window row, the certified unary reindexer, the selected rational-history classifier rows, the finite bundle, membership, unary-observation, and handoff rows.

Next use Definition 126.79. The RegSeqRat finite-window consumer reads precisely the selected RatHistoryClassifier rows together with the displayed ProbeBundle, InBundle, and UnaryHistory ledger rows.

Then apply Definition 38.320. The finite-window $\mathsf{Real}^{\uparrow}$ boundary reads the same displayed ledger surface after the StreamName and RegSeqRat rows have been restricted to the selected window.

Finally use Definition 126.156. Since the reindexed package imports no regularity witness, seal classifier, completion row, quotient row, or host index equality, the four selected source families are the whole common input surface for the two downstream consumers. □

Theorem 126.160 (RegSeqRat tail-classifier finite-window handoff exactness)**.** *For a concrete* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *reindexed window package, the reindexed* RatHistoryClassifier *rows, the displayed* ProbeBundle*,* InBundle*,* UnaryHistory*,* hsame*,* Cont*, and* Pkg *rows, and the DyadicRatCore ledger rows determine exactly the downstream selected* RegSeqRat *tail-classifier row read by the corresponding finite-window* $\mathsf{Real}^{\uparrow}$ *boundary. The determination is by the same finite source rows on both sides: no quotient row, host* $\mathbb{N}$ *index equality,* $\mathsf{Real}^{\uparrow}$ *completeness datum, metric-completeness datum, or hidden selector row is admitted.*

Proof. First apply Definition 126.155. It reduces the package to the public StreamName finite window, the certified unary reindexer, the selected rational-history classifier rows, the finite bundle and membership rows, the unary-observation rows, the hsame/Cont transports, and the package provenance.

Next apply Definition 126.159. This identifies those selected classifier rows with the common finite source surface consumed by the downstream RegSeqRat seal row and by the finite-window $\mathsf{Real}^{\uparrow}$ ledger boundary.

Use Definition 38.360 for the RegSeqRat side. Its pullback uniqueness says that the selected tail-classifier row is fixed by the displayed rational-history rows and the DyadicRatCore ledger rows, with no additional quotient or host index coordinate.

Finally apply Definition 38.404. The $\mathsf{Real}^{\uparrow}$ consumer boundary is the same finite source boundary already selected above, so it reads exactly the tail-classifier row determined by the reindexed StreamName and RegSeqRat ledger surface and nothing hidden beyond it. □

Theorem 126.161 (StreamName RegSeqRat reindexed window composition normal form)**.** *Let*

$$W_0 \xrightarrow{\rho_1} W_1 \xrightarrow{\rho_2} \cdots \xrightarrow{\rho_n} W_n$$

be a finite chain of certified unary reindexers between $\mathsf{StreamName}^{\uparrow}$ *finite windows, and let* $\rho_n \circ \cdots \circ \rho_1$ *denote the composite reindexer carried by the displayed unary-history continuations. The package obtained by iterating the reindexed* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *finite-window handoff along the chain exposes the same selected* RatHistoryClassifier *rows,* ProbeBundle*,* InBundle*,* UnaryHistory*, finite* hsame/Cont*, and downstream* RegSeqRat *handoff surface as the single package formed from the composite reindexer.*

No regularity witness, seal classifier, completion datum, quotient datum, or host index-equality row is imported by passing from the chain to the composite package.

Proof. Proceed by induction on the finite chain of certified unary reindexers.

In the empty-chain case, apply Definition 126.158. The identity package exposes exactly the unreindexed selected RatHistoryClassifier rows together with the visible ProbeBundle, InBundle, UnaryHistory, finite hsame/Cont, and RegSeqRat handoff rows.

For the induction step, assume the initial chain has already been repacked as the composite reindexer. Apply Definition 126.155 to the next one-step package and to the composite package. Both surfaces are exhausted by the displayed StreamName finite-window row, the selected rational-history classifier rows, the bundle and membership rows, the unary observations, and the downstream handoff ledger.

Next use Definition 126.157. Since the composite reindexer and the one-step extension display the same finite source rows after the carried hsame and Cont transports, every downstream consumer reads the same package surface.

Finally apply Definition 126.159 to identify the common selected classifier rows with the downstream RegSeqRat handoff surface. The boundary is exactly Definition 126.156: the normal form imports no regularity witness, seal classifier, completion datum, quotient datum, or host index-equality row. □

Theorem 126.162 (StreamName RealUp subwindow ledger determinacy)**.** *Let Ω and Ω' be two reindexed* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *finite-window packages over rational histories. Assume that they display the same public* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same certified unary reindexer, the same selected* ProbeBundle *subwindow, the same* InBundle *membership rows, the same* UnaryHistory *observation rows, and the same transported* RatHistoryClassifier *rows. Then every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer reading the corresponding subwindow sees the same finite* BHist *source surface from Ω and Ω'.*

The common surface consists exactly of the public StreamName classifier rows, the RegSeqRat *handoff rows, the selected endpoint rows, and the finite ledger boundary carried by the displayed bundle, membership, and unary-history rows. It contains no quotient datum, completion object, host-index equality, metric-completeness witness, or standard-bridge row.*

Proof. Start with the two reindexed window packages. By Definition 126.157, the common public StreamName row, certified unary reindexer, selected bundle, membership rows, unary-history rows, and transported rational classifiers force the same finite source surface for every downstream finite-window consumer.

Next project the RegSeqRat side. The finite-window handoff theorem Definition 126.79 reads exactly the selected RatHistoryClassifier rows together with the displayed ProbeBundle, InBundle, and UnaryHistory ledger rows. The restriction theorem Definition 126.175 keeps those rows unchanged when the consumer asks only for the selected subwindow.

Now attach the RealUp read boundary. Apply Definition 38.320 to the common subwindow surface. It identifies the visible $\mathsf{Real}^{\uparrow}$ input as the selected StreamName classifier rows, endpoint rows, RegSeqRat dyadic source rows, and finite ledger boundary already present in the two packages.

Finally use Definition 126.156. Since the reindexed package imports no seal classifier, quotient, completion, host-index equality, metric-completeness witness, or standard-bridge row, no finite-window $\mathsf{Real}^{\uparrow}$ consumer can distinguish Ω from Ω' by such data. The exposed BHist source surface is therefore the same. □

Theorem 126.163 (StreamName RegSeqRat composite reindexed RealUp boundary)**.** *Let Ω_{01} and Ω_{12} be composable reindexed* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *finite-window packages over rational histories. Assume that the output side of Ω_{01} and the input side of Ω_{12} share the middle transported* RatHistoryClassifier *rows, the same selected* ProbeBundle*, the same* InBundle *membership*

rows, and the same UnaryHistory *ledger rows. Then the composite package exposes to every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer exactly the selected StreamName classifier rows, the* RegSeqRat *handoff rows, the selected endpoint rows, and the finite ledger boundary carried by those displayed bundle, membership, and unary-history rows.*

The composite package exposes no quotient datum, completion object, host-index equality, metric-completeness witness, or standard-bridge row. In particular, the middle transported rational classifier rows are boundary glue for the two finite windows, not a source of additional $\mathsf{Real}^{\uparrow}$ *input data.*

Proof. First compare each leg with its finite-window consumer surface. Apply Definition 126.157 to Ω_{01} and to Ω_{12}. The shared middle RatHistoryClassifier rows, selected bundle, membership rows, and unary-history ledger rows force the same visible source surface at the joint window boundary.

Next identify the rows that survive the handoff. By Definition 126.159, the common source surface is exactly the selected StreamName classifier rows together with the RegSeqRat handoff rows and the displayed finite ledger rows. Composability only matches the middle classifier entries; it does not add a second handoff family.

Now read the composite through a finite-window $\mathsf{Real}^{\uparrow}$ consumer. Apply Definition 126.162 to the shared subwindow surface. The $\mathsf{Real}^{\uparrow}$ input is therefore the selected StreamName classifier rows, the selected endpoint rows, the RegSeqRat handoff rows, and the finite ledger boundary already present in the two packages.

Finally apply Definition 126.156 to both legs. Since neither leg exposes quotient data, completion data, host-index equality, metric-completeness witnesses, or standard-bridge rows, their composition cannot expose such a row to the $\mathsf{Real}^{\uparrow}$ consumer. The displayed finite boundary is the whole composite readback surface. □

Theorem 126.164 (StreamName RegSeqRat reindexed window composition associativity)**.** *Let* Ω_{01}*,* Ω_{12}*, and* Ω_{23} *be three composable reindexed* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat *finite-window packages over rational histories. Assume that the two middle interfaces match by the displayed transported* RatHistoryClassifier *rows, selected* ProbeBundle *rows,* InBundle *membership rows, and* UnaryHistory *ledger rows. Then the two composites* $(\Omega_{23}\circ\Omega_{12})\circ\Omega_{01}$ *and* $\Omega_{23}\circ(\Omega_{12}\circ\Omega_{01})$ *expose the same selected StreamName classifier rows, the same* RegSeqRat *handoff rows, the same* ProbeBundle*,* InBundle*, and* UnaryHistory *ledger rows, and the same finite-window* $\mathsf{Real}^{\uparrow}$ *seal surface.*

Neither bracketing introduces a regularity witness, completion object, quotient row, host-index equality, or unrecorded stream datum. Thus every finite-window consumer reads the same BEDC boundary from both bracketings.

Proof. First compare the adjacent composites. Apply Definition 126.157 to the Ω_{01}–Ω_{12} interface and again to the Ω_{12}–Ω_{23} interface. The transported middle classifier rows, selected bundle rows, membership rows, and unary-history ledger rows force the same finite consumer surface at each shared boundary.

Next form the left bracketing. Use Definition 126.163 on Ω_{01} and Ω_{12}, then use the same theorem on the resulting composite and Ω_{23}. The visible output is exactly the selected StreamName classifiers, RegSeqRat handoff rows, endpoint rows, and displayed finite ledger boundary.

For the right bracketing, apply Definition 126.163 first to Ω_{12} and Ω_{23}, and then to Ω_{01} and that composite. The same determinacy rows are used at the same two middle interfaces, so the selected classifiers, handoff rows, bundle rows, membership rows, unary-history rows, and $\mathsf{Real}^{\uparrow}$ boundary are identical.

Finally apply Definition 126.156 to the three component packages. Since no component imports a seal classifier, regularity witness, completion object, quotient row, host-index equality,

or unrecorded stream datum, neither bracketing can create such a distinguishing row. The two bracketings therefore expose one common finite-window boundary. □

Theorem 126.165 (StreamName RegSeqRat reindexed window structural closure)**.** *Let Ω_{01} and Ω_{12} be composable reindexed* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}$ *finite-window packages over rational histories. Assume that their selected probe bundles are joined by* $\mathsf{bundleAppend}$ *and that their reindexing maps compose on the selected rational observation rows. Then the composite package exposes exactly the selected StreamName classifier rows, the* $\mathsf{RegSeqRat}$ *handoff rows, the* $\mathsf{ProbeBundle}$, $\mathsf{InBundle}$, *and* $\mathsf{UnaryHistory}$ *ledger rows, and the finite-window* $\mathsf{Real}^{\uparrow}$ *boundary already exposed by the two factors after reindexing.*

No seal row, quotient row, completion object, host-index equality, regularity witness, or unrecorded stream datum is introduced by the composite package.

Proof. First apply Definition 126.155 to each factor. For each leg, the accepted finite-window consumer surface is exhausted by the selected StreamName classifier rows, the $\mathsf{RegSeqRat}$ handoff rows, and the displayed probe, membership, and unary-history ledger rows.

Next use the structural laws for $\mathsf{bundleAppend}$. Appending the two finite probe bundles preserves exactly the entries already recorded in the two factor ledgers and supplies the composite $\mathsf{InBundle}$ and $\mathsf{UnaryHistory}$ rows by finite concatenation, not by a new stream source.

The reindexing composition law identifies the selected rational observation rows of the appended package with the observation rows obtained by applying the two factor reindexings in sequence. Hence the StreamName classifier rows and $\mathsf{RegSeqRat}$ handoff rows seen by the composite consumer are precisely the transported rows already present in the factors.

Finally apply Definition 126.163 to read the finite $\mathsf{Real}^{\uparrow}$ boundary, and apply Definition 126.156 to the two factors. These theorems exclude seal, quotient, completion, host-index, regularity, and unrecorded-stream rows, so the displayed rows are the whole structural closure of the composite package. □

Theorem 126.166 (StreamName RegSeqRat reindexed window diamond source determinacy)**.** *Let Ω_{01}, Ω_{12}, Ω'_{01}, and Ω'_{12} be two certified composable reindexing routes from the same finite* $\mathsf{StreamName}^{\uparrow}$ *source window to the same finite* $\mathsf{Real}^{\uparrow}$ *boundary. Suppose the two routes display the same transported* $\mathsf{RatHistoryClassifier}$ *rows, selected* $\mathsf{ProbeBundle}$ *rows,* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *ledger rows,* hsame *transport rows,* Cont *continuation rows,* Pkg *package rows, and* $\mathsf{RealLedgerPolicy}$ *rows at the shared boundary. Then the two composite routes expose the same finite* $\mathsf{Real}^{\uparrow}$ *source boundary.*

The common boundary is the selected StreamName classifier surface together with the displayed $\mathsf{RegSeqRat}$ *handoff rows, endpoint rows, finite probe and membership rows, unary-history ledger rows, continuation rows, package rows, and real-ledger-policy rows. No quotient datum, completion object, host-index equality, regularity witness, standard bridge row, or hidden stream row can separate the two routes.*

Proof. First compare the matching legs. Apply Definition 126.157 to Ω_{01} and Ω'_{01}, and again to Ω_{12} and Ω'_{12}. The shared classifier, bundle, membership, unary-history, hsame, Cont, Pkg, and real-ledger-policy rows determine the same visible consumer surface on each side of the diamond.

Next close each route under composition. By Definition 126.165, the composite $\Omega_{12} \circ \Omega_{01}$ and the composite $\Omega'_{12} \circ \Omega'_{01}$ expose only the selected source classifiers, the $\mathsf{RegSeqRat}$ handoff rows, and the displayed finite ledger rows.

The bracketing of the certified route does not change that surface. Apply Definition 126.164 to compare the route read as a two-leg composite with the same route read through any adjacent certified handoff factorization.

Finally read both composites through the finite $\mathsf{Real}^{\uparrow}$ boundary theorem. Use Definition 126.163 to identify the shared source boundary, and use Definition 126.156 to exclude quotient, completion, host-index, regularity, standard-bridge, and hidden stream rows. The two routes therefore expose one finite $\mathsf{Real}^{\uparrow}$ source boundary. □

Theorem 126.167 (StreamName RegSeqRat reindexed window triple-diamond normal form)**.** *Let $\mathcal{R}_0$, $\mathcal{R}_1$, and $\mathcal{R}_2$ be three certified composable reindexing routes from the same finite* $\mathsf{StreamName}^{\uparrow}$ *source window to the same finite* $\mathsf{Real}^{\uparrow}$ *boundary. Suppose the three routes display the same selected* RatHistoryClassifier *rows,* ProbeBundle *rows,* InBundle *witnesses,* UnaryHistory *ledgers,* hsame *transport rows,* Cont *rows,* Pkg *rows, and* RealLedgerPolicy *rows. Then the three routes have a common normal form: they expose one finite* $\mathsf{Real}^{\uparrow}$ *source boundary, namely the selected StreamName classifier surface together with the displayed* RegSeqRat *handoff, probe, membership, unary-history, continuation, package, and real-ledger-policy rows.*

This normal form contains no quotient datum, completion object, host-index equality, regularity witness, standard bridge row, or hidden stream row. Any finite-window $\mathsf{Real}^{\uparrow}$ *consumer of any one of the three routes therefore reads exactly the same source boundary and exactly the same recorded handoff ledger.*

Proof. First compare the routes pairwise. Apply Definition 126.166 to $\mathcal{R}_0$ and $\mathcal{R}_1$, and again to $\mathcal{R}_1$ and $\mathcal{R}_2$. The shared classifier, probe, membership, unary-history, transport, continuation, package, and real-ledger-policy rows force the same finite $\mathsf{Real}^{\uparrow}$ source boundary on both comparisons.

Next normalize the bracketing of each route. Use Definition 126.164 to identify adjacent certified factorizations of the same route, and use Definition 126.165 to read the route through only the selected StreamName classifier rows, the RegSeqRat handoff rows, and the displayed finite ledger rows.

Finally apply Definition 126.163 to each normalized route. The common source boundary delivered by the pairwise diamond comparison is exactly the finite $\mathsf{Real}^{\uparrow}$ boundary delivered by the composite-boundary theorem, so no quotient, completion, host-index equality, regularity witness, standard bridge row, or hidden stream row is available to distinguish the three routes. □

Theorem 126.168 (StreamName RegSeqRat reindexed window triple terminal seal determinacy)**.** *Let $\mathcal{R}_0$, $\mathcal{R}_1$, and $\mathcal{R}_2$ be three certified composable reindexing routes from one finite* $\mathsf{StreamName}^{\uparrow}$ *source window to one terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumer. Suppose the routes share the displayed selected* RatHistoryClassifier *rows,* RegSeqRat *handoff rows, endpoint rows,* ProbeBundle*,* InBundle*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and* RealLedgerPolicy *rows. Then every terminal seal consumer of the three routes reads the same finite* $\mathsf{Real}^{\uparrow}$ *source boundary.*

No quotient, completion object, host-index equality, regularity witness, standard bridge row, or hidden stream row distinguishes the three routes at the terminal seal. The terminal consumer can therefore observe only the common finite StreamName boundary transported through the RegSeqRat *handoff and the displayed finite-window* $\mathsf{Real}^{\uparrow}$ *endpoint rows.*

Proof. First put the three routes into the common finite-window normal form. Apply Definition 126.167 to the shared classifier, handoff, probe, membership, unary-history, transport, continuation, package, and real-ledger-policy rows. This gives one finite source boundary for the three certified routes.

Next read that boundary as a finite $\mathsf{Real}^{\uparrow}$ boundary. The composite-boundary theorem Definition 126.163 identifies the selected StreamName classifier surface and the RegSeqRat handoff rows with the finite $\mathsf{Real}^{\uparrow}$ source boundary available to the endpoint consumer.

Finally pass from the endpoint boundary to the terminal seal. Apply Definition 126.45 to the finite $\mathsf{Real}^{\uparrow}$ boundary just obtained. The terminal seal reads exactly that boundary, and Definition 126.156 excludes quotient, completion, host-index, regularity, standard-bridge, and hidden stream rows from the seal interface. □

Theorem 126.169 (StreamName RegSeqRat RealUp common-window seal)**.** *Let W be a reindexed finite* $\mathsf{StreamName}^{\uparrow}$ *window equipped with selected* RatHistoryClassifier *rows,* ProbeBundle *rows,* InBundle *witnesses,* UnaryHistory *rows,* hsame *transport rows,* Cont *rows,* Pkg *rows, displayed* RegSeqRat *handoff rows, displayed* DyadicRatCore *endpoint rows, and a finite* RealLedgerPolicy *ledger. Suppose the* RegSeqRat *finite-window seal consumer and the finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal consumer are read through certified reindexing routes using exactly these displayed rows. Then the two consumers read the same finite source boundary: the selected* $\mathsf{StreamName}^{\uparrow}$ *classifier surface transported through the displayed* RegSeqRat *handoff,* DyadicRatCore *endpoint, probe, membership, unary-history, continuation, package, and real-ledger-policy rows.*

This common boundary contains no quotient datum, completion object, host-index equality, hidden stream row, or independent regularity row. In particular, the RegSeqRat *finite-window seal and the* $\mathsf{Real}^{\uparrow}$ *terminal seal are consumers of one finite source boundary, not two separately supplied boundary packages.*

Proof. First identify the RegSeqRat side of the boundary. Apply Definition 126.159 to the selected classifier rows, bundle rows, membership witnesses, unary history, hsame transport, continuation, package, and displayed handoff rows. The finite RegSeqRat consumer therefore sees only the selected $\mathsf{StreamName}^{\uparrow}$ source surface transported through the displayed handoff ledger.

Next compare the terminal routes for the finite $\mathsf{Real}^{\uparrow}$ seal. Apply Definition 126.168 to the route that reads the RegSeqRat seal boundary, the route that reads the displayed DyadicRatCore endpoint rows, and the route that reads the finite $\mathsf{Real}^{\uparrow}$ terminal seal. The shared classifier, handoff, endpoint, probe, membership, unary-history, hsame, Cont, Pkg, and RealLedgerPolicy rows force a single finite source boundary for all three consumers.

Finally read that same boundary at the terminal seal. Use Definition 126.45 to identify the finite-window $\mathsf{Real}^{\uparrow}$ consumer with the boundary already selected by the RegSeqRat handoff and endpoint rows. The no-import clause in the terminal determinacy theorem excludes quotient, completion, host-index equality, hidden stream, and independent regularity rows, so the two seal consumers share exactly the displayed finite source boundary. □

126.10 StreamName RegSeqRat Pointwise Handoff

Theorem 126.170 (StreamName pointwise handoff package)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows and let $\mathcal{P}$:* ProbeBundle *be a finite observation bundle. A pointwise handoff from* $\mathsf{StreamName}^{\uparrow}$ *to* RegSeqRat *or to a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer consists exactly of the following source rows:*

1. $\mathsf{RatStreamNameCarrier}(s)$ *and* $\mathsf{RatStreamNameCarrier}(t)$*;*
2. $\mathsf{RatStreamNameClassifier}(s, t)$*, restricted on $\mathcal{P}$ through* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$*;*
3. *for each certified n, the rows* $\mathsf{UnaryHistory}(n)$*,* $\mathsf{InBundle}(n, \mathcal{P})$*, and the pointwise* hsame *comparison carried by the classifier at n.*

Every row handed to RegSeqRat *or to a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer is one of these pointwise StreamName rows. The handoff contains no completion object, quotient row, host-index equality,* RegSeqRat *regularity row, or* $\mathsf{Real}^{\uparrow}$ *seal classifier.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRegSeqRatPointwiseHandoff`

Proof. Project the public finite-window StreamName surface by Definition 126.79. It exposes the selected classifier rows, the finite ProbeBundle ledger, InBundle membership, and UnaryHistory observations.

Now read the selected classifier pointwise. Definitions Definition 126.1 and Definition 126.2 identify the carrier and classifier rows as statements about the observations $s(n)$ and $t(n)$ at certified unary histories. The finite-window classifier row restricts exactly those pointwise observations to the selected bundle.

Finally apply Definition 126.100 and Definition 126.82. These theorems repack the pointwise rows into the same downstream handoff and keep regularity and seal classifiers on the consumer side. □

Theorem 126.171 (StreamName RegSeqRat observation stability)**.** *Let* $s, t, u, s', t' : \mathsf{BHist} \to \mathsf{BHist}$ *and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be finite. Suppose* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *and* $\mathsf{RatStreamNameFiniteWindowClassifier}(t, u, \mathcal{P})$ *hold, and suppose every certified* n *with* $\mathsf{InBundle}(n, \mathcal{P})$ *and* $\mathsf{UnaryHistory}(n)$ *carries the pointwise comparisons* $\mathsf{hsame}(s(n), s'(n))$ *and* $\mathsf{hsame}(t(n), t'(n))$*. Then the same bundle also supports the self, symmetric, transitive, and transported pointwise classifier rows:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, s, \mathcal{P}),$$
$$\mathsf{RatStreamNameFiniteWindowClassifier}(t, s, \mathcal{P}),$$
$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, u, \mathcal{P}),$$
$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P}).$$

Thus a RegSeqRat *handoff or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer may transport the StreamName observation window only along pointwise* hsame *rows already certified in the finite bundle.*

Proof. Unfold the finite-window classifier at a certified member of $\mathcal{P}$. The hypotheses provide the pointwise classifier rows $s(n) \sim t(n)$ and $t(n) \sim u(n)$ together with the carrier row for $s(n)$.

Apply the finite-window stability fields represented on the Lean side by RatStreamNameFiniteWindowClassifier_s Its four projections give the self row for s, the symmetric row from t to s, the transitive row from s to u, and the transported row from s' to t' on the same finite bundle.

The consumer boundary is unchanged. By Definition 126.170, the transported rows are still only pointwise StreamName observations plus bundle membership and unary history evidence, so no downstream regularity or seal row is imported. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier stability fields`

Theorem 126.172 (StreamName RealUp constant observation handoff)**.** *Let* $d, e : \mathsf{BHist}$ *be rational-history endpoints, let* $r, q : \mathsf{BHist} \to \mathsf{BHist}$ *be observation schedules, and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite unary observation bundle. The constant StreamName rows*

$$n \mapsto \mathsf{RatStreamName_constant}(d, r(n)), \qquad n \mapsto \mathsf{RatStreamName_constant}(e, q(n))$$

feed RegSeqRat *and finite-window* $\mathsf{Real}^{\uparrow}$ *only through the pointwise rows of Definition 126.170. The rational endpoint classifier for* d *and* e *supplies the constant point observations, while regularity, dyadic seal data, and the* $\mathsf{Real}^{\uparrow}$ *seal classifier remain downstream obligations.*

Proof. Start from the constant StreamName boundary. Apply Definition 126.81: a constant finite window exports the rational point-classifier row, the finite bundle ledger, membership rows, and unary observation rows, while RegSeqRat regularity remains outside the StreamName source.

Apply Definition 126.286 to the two scheduled constant rows. The Lean definitions RatStreamNameCarrier, RatStreamNameClassifier, and RatStreamName_constant express this as pointwise observations of the fixed rational endpoints d and e at every certified unary history.

Finally pass the resulting finite window through Definition 38.314 and Definition 38.320. These RealUp consumer theorems read only the pointwise StreamName packet and its ledger, so the constant handoff contributes no completion, quotient, host-index equality, regularity witness, or seal classifier. □

Theorem 126.173 (StreamName singleton diagonal handoff)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $n : \mathsf{BHist}$ be a certified unary observation, and let S_n be the singleton probe bundle generated by n. If $\mathsf{RatStreamNameClassifier}(s, t)$ and $\mathsf{UnaryHistory}(n)$ hold, then the exact finite-window packet handed to RegSeqRat or to a finite-window $\mathsf{Real}^{\uparrow}$ consumer is*

$$\begin{gathered}\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, S_n),\\ \mathsf{InBundle}(n, S_n),\\ \mathsf{RatHistoryClassifier}(s(n), t(n)).\end{gathered}$$

This singleton packet exports no RegSeqRat regularity row, no $\mathsf{Real}^{\uparrow}$ seal classifier, no completion object, and no quotient row.

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier singleton diagonal handoff`

Proof. Start from the global StreamName classifier. By Definition 126.2, reading it at the certified unary history n gives the pointwise rational-history classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Now restrict the classifier to the singleton bundle S_n. The finite-window definition, equivalently Definition 126.23, says that the only visible observation slot of S_n is the row selected by $\mathsf{InBundle}(n, S_n)$. Hence the restricted row is exactly $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, S_n)$ together with that membership witness and the pointwise classifier at n.

Finally read the packet through Definition 126.170. The consumer boundary is the one separated in Definition 126.82: regularity, sealing, completion, and quotient data remain downstream, so the singleton source row is precisely the finite-window packet displayed above. □

Theorem 126.174 (StreamName singleton restriction handoff exactness)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $n : \mathsf{BHist}$ be a certified unary observation, and let S_n be the singleton probe bundle generated by n. For any certified restriction of a finite observation bundle through S_n, the restricted RegSeqRat and finite-window $\mathsf{Real}^{\uparrow}$ handoff is exactly the tuple*

$$\begin{gathered}\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, S_n),\\ \mathsf{InBundle}(n, S_n),\\ \mathsf{UnaryHistory}(n),\\ \mathsf{RatHistoryClassifier}(s(n), t(n)).\end{gathered}$$

It exports no RegSeqRat regularity row, $\mathsf{Real}^{\uparrow}$ seal classifier, completion object, quotient row, or host-index equality.

Proof. Begin with the singleton diagonal handoff of Definition 126.173. It identifies the singleton source packet with the finite-window classifier over S_n, the membership witness for n, and the pointwise rational-history classifier at the selected unary observation.

Now apply the finite-window restriction handoff of Definition 126.175 to any certified restriction whose visible observation bundle factors through S_n. Since the target bundle is singleton, every visible row is the same certified n-row together with $\mathsf{InBundle}(n, S_n)$ and $\mathsf{UnaryHistory}(n)$.

Finally read the resulting restricted packet through the pointwise handoff boundary of Definition 126.170. That boundary has carrier, membership, unary-history, finite-window classifier, and pointwise rational-history slots only, so it cannot export a regularity row, a seal classifier, completion data, quotient data, or host-index equality. The displayed tuple is therefore the whole consumer surface. □

Theorem 126.175 (StreamName RegSeqRat finite-window restriction handoff)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P} : \mathsf{ProbeBundle}$ be a finite observation bundle, and let $\mathcal{Q} : \mathsf{ProbeBundle}$ be a certified sub-bundle of $\mathcal{P}$. Any pointwise handoff package from $\mathcal{P}$ restricts to the same StreamName source rows over $\mathcal{Q}$: the* $\mathsf{RatStreamNameCarrier}$ *rows for s and t, the restricted* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ *row,* $\mathsf{UnaryHistory}$*,* $\mathsf{InBundle}(-, \mathcal{Q})$*, and the pointwise* $\mathsf{RatHistoryClassifier}$ *rows. The restricted packet exports no* $\mathsf{RegSeqRat}$ *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Proof. Unfold the finite-window classifier for the smaller certified bundle $\mathcal{Q}$. A visible observation over $\mathcal{Q}$ is a certified unary history n together with $\mathsf{InBundle}(n, \mathcal{Q})$.

Use the sub-bundle evidence to read the same observation as $\mathsf{InBundle}(n, \mathcal{P})$. The pointwise handoff over $\mathcal{P}$ then supplies exactly the source rows named by Definition 126.170: the carried StreamName endpoints, unary history evidence, and the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ row.

Repack those rows with the membership witness for $\mathcal{Q}$. This gives $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ and the same pointwise rational-history classifier at each selected observation.

The final packet is still read through the exhaustion boundary of Definition 126.170. Since that boundary contains only StreamName carrier, finite-window classifier, unary history, bundle membership, and pointwise classifier rows, the restriction introduces no regularity, seal, completion, quotient, or host-index-equality datum. □

Theorem 126.176 (StreamName RegSeqRat appended-bundle handoff exactness)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, and let $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$ be finite observation windows. Suppose $\mathcal{P}$ and $\mathcal{Q}$ each supply the pointwise StreamName handoff rows for s and t: the* $\mathsf{RatStreamNameCarrier}$ *source rows, the finite-window classifier row,* $\mathsf{UnaryHistory}$ *evidence, membership evidence, and the pointwise* $\mathsf{RatHistoryClassifier}$ *rows. Then the appended bundle* $\mathsf{BAppend}(\mathcal{P}, \mathcal{Q})$ *supplies exactly the same kind of* $\mathsf{RegSeqRat}$ *and finite-window* $\mathsf{Real}^{\uparrow}$ *handoff rows over the combined window.*

Conversely, any consumer read over $\mathsf{BAppend}(\mathcal{P}, \mathcal{Q})$ *restricts to one of the two component handoff packets. The appended packet exports no* $\mathsf{RegSeqRat}$ *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier bundle union handoff`

Proof. Apply Definition 126.305 to the two finite observation windows. It identifies the visible observations of $\mathsf{BAppend}(\mathcal{P}, \mathcal{Q})$ with the observations that come from $\mathcal{P}$ or from $\mathcal{Q}$, carrying their original $\mathsf{InBundle}$ and $\mathsf{UnaryHistory}$ witnesses.

On each component observation, use Definition 126.170. The supplied component handoff packet gives the carried StreamName endpoints and the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ row, and the union composition repackages those rows into the finite-window classifier for $\mathsf{BAppend}(\mathcal{P}, \mathcal{Q})$.

For the converse read, restrict the appended window along the selected component inclusion. Then Definition 126.175 recovers exactly the same carrier, classifier, membership, unary-history, and pointwise rational-history rows over that component packet.

Both directions pass through the pointwise handoff boundary and the finite-window restriction boundary. Since those boundaries name no regularity, seal, completion, quotient, or host-index-equality coordinate, the appended handoff cannot export any such row. □

Theorem 126.177 (StreamName RegSeqRat finite reindexing coverage)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P}_0, \ldots, \mathcal{P}_k : \mathsf{ProbeBundle}$ be finite observation bundles, and let the adjacent stages be connected by certified unary reindexing maps. Suppose each adjacent reindexing preserves* UnaryHistory*,* ProbeBundle *membership,* InBundle *rows, pointwise* hsame *comparisons, and the* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_i)$ *rows carried by the stage. Then the terminal handoff over $\mathcal{P}_k$ to* RegSeqRat *or to a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer is covered by exactly the same pointwise StreamName rows as the stagewise handoff: the* RatStreamNameCarrier *rows for s and t, the terminal finite-window classifier, the transported* UnaryHistory *and* $\mathsf{InBundle}(-, \mathcal{P}_k)$ *evidence, and the pointwise* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *rows at the transported observations.*

The coverage statement is only a finite-window BEDC handoff. It exports no RegSeqRat *regularity row, no* $\mathsf{Real}^{\uparrow}$ *seal classifier, no completion object, no quotient row, and no host-index equality.*

Proof. Induct on the finite reindexing schedule. In the empty schedule, the terminal bundle is the initial bundle, and Definition 126.170 gives exactly the listed StreamName carrier, finite-window classifier, unary-history, membership, and pointwise rational-history rows.

For the step, assume the coverage packet has been built over $\mathcal{P}_i$. The certified unary reindexing map transports the visible unary histories and their InBundle witnesses to the next bundle. Apply Definition 126.171 to carry the pointwise hsame comparisons and the pointwise rational-history classifier rows along that transported observation.

The next terminal window may discard observations or assemble adjacent visible pieces. Use Definition 126.175 for the restricted part and Definition 126.176 for the finite bundle repacking. These two handoff results rebuild the terminal $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_{i+1})$ row from the transported pointwise rows.

The induction step therefore preserves the exact public surface named in the statement. Since every stage passes through the pointwise, restriction, and bundle-union handoff boundaries, no stage introduces a regularity row, seal classifier, completion object, quotient row, or host-index equality. □

Theorem 126.178 (StreamName RegSeqRat overlap pullback handoff)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$ be finite observation bundles, and let $\mathcal{R} : \mathsf{ProbeBundle}$ be a certified common sub-bundle of both $\mathcal{P}$ and $\mathcal{Q}$. If the $\mathcal{P}$-packet and the $\mathcal{Q}$-packet each supply the pointwise StreamName-to-*RegSeqRat *handoff rows for s and t, then their restrictions to $\mathcal{R}$ classify the same consumer-facing tuple:* RatStreamNameCarrier*,* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{R})$*,* UnaryHistory*,* $\mathsf{InBundle}(-, \mathcal{R})$*, the pointwise* hsame *rows, and* RatHistoryClassifier*. The common-overlap packet contains no* RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal row, completion row, quotient row, or host-index equality row.*

Proof. Restrict the $\mathcal{P}$-packet to the certified common sub-bundle $\mathcal{R}$ by Definition 126.175. The restricted packet exposes the same StreamName carrier rows, the finite-window classifier over $\mathcal{R}$, the unary-history and $\mathsf{InBundle}(-, \mathcal{R})$ witnesses, and the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ rows.

Apply the same restriction theorem to the $\mathcal{Q}$-packet. Because both restricted reads use the same rows s, t and the same certified bundle $\mathcal{R}$, the pointwise handoff Definition 126.170 identifies both packet readings with the same consumer-facing tuple on each observation of $\mathcal{R}$.

Finally apply Definition 126.180. The envelope is closed under the two certified restrictions and contains only carrier, finite-window, unary-history, membership, pointwise hsame, and rational-history classifier rows. Hence the pullback over the common overlap has no coordinate for regularity, seal, completion, quotient, or host-index equality data. □

Theorem 126.179 (StreamName finite observation RealUp seal scope)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P} : \mathsf{ProbeBundle}$ be finite, and let n be a certified observation with $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$. A finite-window $\mathsf{Real}^{\uparrow}$ seal consumer of $\mathsf{RatStreamNameFiniteWindowClassifier}($ may read from StreamName only the carried endpoints for s and t, the membership row $\mathsf{InBundle}(n, \mathcal{P})$, the unary observation row $\mathsf{UnaryHistory}(n)$, and the pointwise classifier row $\mathsf{RatHistoryClassifier}(s(n), t(n))$.*

All $\mathsf{RegSeqRat}$ regularity rows, $\mathsf{Real}^{\uparrow}$ seal classifiers, completion data, quotient data, and host-index equality remain downstream obligations and are not exported by the finite StreamName observation.

Proof. Begin with the pointwise handoff of Definition 126.170. It exposes the StreamName endpoint rows, the finite bundle membership row, the unary observation row, and the pointwise rational-history classifier at the chosen observation.

For the certified observation n, use Definition 126.173 to identify the singleton diagonal view with the same endpoint, membership, unary-history, and pointwise classifier rows. Then use Definition 126.175 to read the same rows through any certified finite-window restriction of $\mathcal{P}$.

Finally apply the obligation separation of Definition 126.82. It keeps $\mathsf{RegSeqRat}$ regularity, $\mathsf{Real}^{\uparrow}$ seal classifiers, completion structure, quotient structure, and host-index equality on the downstream side of the handoff. The finite observation therefore exports exactly the named StreamName rows and nothing from the later seal boundary. □

Theorem 126.180 (StreamName RegSeqRat observation envelope)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows and let $\mathcal{P}$ be a finite $\mathsf{ProbeBundle}$. The observation packet consisting of $\mathsf{RatStreamNameCarrier}$, $\mathsf{RatStreamNameFiniteWindowClassifier}$, $\mathsf{UnaryHistory}$, $\mathsf{InBundle}$, and the pointwise hsame and $\mathsf{RatHistoryClassifier}$ rows is closed under certified sub-bundle restriction and singleton extraction. The packet contains no $\mathsf{RegSeqRat}$ regularity row, no completion row, no quotient row, and no $\mathsf{Real}^{\uparrow}$ seal row.*

Proof. For the full finite window, apply Definition 126.170. It identifies the handoff rows available to $\mathsf{RegSeqRat}$ with the carried StreamName endpoint, the finite probe bundle, the bundle-membership rows, and the pointwise rational-history classifier.

For a certified sub-bundle, use Definition 126.171. The same $\mathsf{RatStreamNameCarrier}$ rows and pointwise hsame transports are read after restriction, while membership is replaced by the certified $\mathsf{InBundle}$ rows of the smaller bundle.

For singleton extraction, apply Definition 126.173. The extracted observation has exactly one unary history row, its singleton membership witness, and the corresponding rational-history classifier. Since all three steps read only these finite observation rows, the envelope has no place for regularity, completion, quotient, or $\mathsf{Real}^{\uparrow}$ seal data. □

Theorem 126.181 (StreamName RegSeqRat finite-window coverage)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a finite observation bundle. For every row of $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$, every certified $\mathsf{UnaryHistory}(n)$, and every witness $\mathsf{InBundle}(n, \mathcal{P})$, the exposed observation is exactly the pointwise row $\mathsf{RatHistoryClassifier}(s(n), t(n))$ together with the corresponding pointwise* hsame *datum. The coverage surface contains no* RegSeqRat *regularity row, completion row, quotient row, or $\mathsf{Real}^{\uparrow}$ seal row.*

Proof. Start with the global handoff of Definition 126.170. It lists the carrier, finite-window classifier, unary-history, bundle-membership, pointwise hsame, and rational-history classifier rows as the only rows visible at the StreamName-to-RegSeqRat boundary.

The observation stability theorem, Definition 126.171, identifies the same pointwise hsame and rational-history row after passing through the certified bundle observation. The singleton diagonal handoff of Definition 126.173 then reads the selected n-row as the diagonal pointwise classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

For an arbitrary finite row, the restriction handoff Definition 126.175 reduces the visible packet to the certified sub-bundle containing that observation. Finally, the envelope theorem Definition 126.180 removes every regularity, completion, quotient, and $\mathsf{Real}^{\uparrow}$ seal row from the consumer surface. Hence the finite-window coverage is exactly the displayed pointwise classifier row and its membership evidence. □

Theorem 126.182 (StreamName RegSeqRat tail reindexing stability)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P} : \mathsf{ProbeBundle}$ be a finite observation bundle, and let $r : \mathsf{BHist} \to \mathsf{BHist}$ be a certified unary tail schedule whose selected rows preserve* UnaryHistory *and $\mathsf{InBundle}(-, \mathcal{P})$. The StreamName handoff package for s and t transports to the package for $s \circ r$ and $t \circ r$ using only the reindexed pointwise* hsame *rows, the reindexed* UnaryHistory *rows, and the transported* InBundle *rows. No reindexed packet imports* RegSeqRat *regularity, completion, quotient, host-index equality, or $\mathsf{Real}^{\uparrow}$ seal data.*

Proof. Apply the pointwise handoff Definition 126.170 before reindexing. Its source rows are the StreamName carriers, the finite-window classifier, unary histories, bundle membership, pointwise hsame, and pointwise rational-history classifiers.

The certified tail schedule supplies the same observation witnesses after composition with r. By observation stability, Definition 126.171, the transported rows remain pointwise rows; no global regularity or quotient datum is used.

For each selected row, use the singleton diagonal handoff Definition 126.173 to identify the transported observation with $\mathsf{RatHistoryClassifier}(s(r(n)), t(r(n)))$. The finite-window restriction handoff, Definition 126.175, repacks these rows over the reindexed certified sub-bundle.

The observation envelope Definition 126.180 is stable under this repacking because it names only carrier, finite-window, membership, unary-history, pointwise hsame, and pointwise rational classifier rows. Thus the tail-reindexed package is exhausted by the same pointwise handoff surface. □

Theorem 126.183 (StreamName RegSeqRat finite-window union seal determinacy)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be certified finite* ProbeBundle *observation bundles for StreamName rows $s, t : \mathsf{BHist} \to \mathsf{BHist}$, and let $\mathcal{R}$ be their certified finite overlap. Assume that the* RatStreamNameClassifier *rows on $\mathcal{R}$ match and that the induced* RegSeqRat *handoff packages read the same dyadic finite ledger on the overlap. Then the union handoff over $\mathcal{P} \cup \mathcal{Q}$ determines a unique* RegSeqRat *finite-window seal surface, and the restrictions back to $\mathcal{P}$ and to $\mathcal{Q}$ recover the original displayed seal surfaces.*

No regularity row outside the displayed RegSeqRat *handoff, no completion row, no quotient row, no host-index equality row, and no* $\mathsf{Real}^{\uparrow}$ *seal row is introduced by the union.*

Proof. Apply Definition 126.175 to $\mathcal{P}$ and to $\mathcal{Q}$. Each restriction displays only the StreamName carrier rows, the finite-window classifier rows, the unary-history and bundle membership witnesses, and the pointwise rational-history classifier rows.

Use Definition 126.176 to assemble these two displayed packets over $\mathcal{P} \cup \mathcal{Q}$. The matching classifier rows on $\mathcal{R}$ and the common dyadic finite ledger identify the overlap as a single source rather than two competing seal readings.

The overlap rows are then compared by Definition 126.182. Tail reindexing preserves the pointwise hsame rows, the transported unary histories, and the transported membership witnesses, so the union packet has one displayed finite-window seal surface on the common part.

Restrict the union packet back to $\mathcal{P}$ and to $\mathcal{Q}$ using the restriction handoff theorem again. Since the union packet contains no row outside the observation envelope, the two restrictions return exactly the original displayed handoff surfaces, and the finite-window seal determined by the union is unique. □

Theorem 126.184 (StreamName RegSeqRat pointwise classifier handoff exactness)**.** *For StreamName rows* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *and a finite bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*, the consumer-facing handoff from StreamName observations to* RegSeqRat *is exhausted by the* RatStreamNameCarrier *rows, the* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *rows, bundle membership, unary observations, pointwise* hsame *rows, and pointwise* RatHistoryClassifier *rows. Any datum outside this list is not part of the handoff interface.*

Proof. The base handoff theorem Definition 126.170 gives the complete source packet for the consumer boundary. Its rows already name the carriers, finite-window classifier, membership witnesses, unary observations, pointwise hsame, and pointwise rational-history classifiers.

Observation stability, Definition 126.171, shows that moving inside the finite bundle preserves only these named pointwise rows. The singleton diagonal handoff, Definition 126.173, gives the same conclusion at a single certified observation.

The finite-window restriction handoff Definition 126.175 checks that every certified sub-bundle exposes no additional row. The observation envelope theorem, Definition 126.180, closes the interface by excluding RegSeqRat regularity, completion, quotient, host-index equality, and $\mathsf{Real}^{\uparrow}$ seal data. Therefore the listed rows are exactly the consumer-facing handoff surface. □

Theorem 126.185 (StreamName RegSeqRat pointwise hsame closure)**.** *Let* $s, t, s', t' : \mathsf{BHist} \to \mathsf{BHist}$ *be StreamName rows and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be finite. Suppose the handoff for* (s, t) *is the exact pointwise package of Definition 126.184, and suppose every certified bundled observation carries* $\mathsf{hsame}(s(n), s'(n))$ *and* $\mathsf{hsame}(t(n), t'(n))$*. Then the handoff for* (s', t') *is exhausted by the same StreamName carrier rows, restricted finite-window classifier rows,* UnaryHistory *rows,* InBundle *rows, pointwise* hsame *rows, and pointwise* RatHistoryClassifier *rows. It accepts no* RegSeqRat *regularity, completion, quotient, host-index equality, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier pointwise hsame closure`

Proof. Start with the source packet from Definition 126.184. That theorem gives exactly the carrier, finite-window classifier, bundle, unary-history, pointwise hsame, and rational-history classifier rows for the displayed (s, t) handoff.

Transport each certified bundled observation along the given pointwise hsame rows. By Definition 126.171, this transport preserves the unary observations, bundle membership witnesses, and

pointwise rational-history classifier rows, and it does not create any row outside the StreamName observation surface.

For a restricted finite window, apply Definition 126.175. The restriction keeps the same finite carrier rows and classifier rows over the sub-bundle, now read through (s', t').

Finally use Definition 126.180. The envelope excludes RegSeqRat regularity, completion, quotient, host-index equality, and $\mathsf{Real}^{\uparrow}$ seal data from the observation handoff, so the transported packet is closed under pointwise hsame exactly on the listed consumer interface. □

Theorem 126.186 (StreamName RegSeqRat pointwise hsame transport composition)**.** *Let $s, t, s', t', s'', t'' :$ BHist $\to$ BHist be StreamName rows, and let $\mathcal{P}$: ProbeBundle be finite. Suppose the handoff for (s, t) is the exact pointwise package of Definition 126.184. Suppose also that every certified bundled observation n carries* $\mathsf{hsame}(s(n), s'(n))$, $\mathsf{hsame}(t(n), t'(n))$, $\mathsf{hsame}(s'(n), s''(n))$, *and* $\mathsf{hsame}(t'(n), t''(n))$. *Then the handoff for (s'', t'') is exhausted by the same* RatStreamNameCarrier *rows, restricted finite-window classifier rows,* UnaryHistory *rows,* InBundle *rows, pointwise* hsame *rows, and pointwise* RatHistoryClassifier *rows. It accepts no* RegSeqRat *regularity, completion, quotient, host-index equality, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Proof. First apply Definition 126.185 to the exact handoff for (s, t) and the displayed observations from (s, t) to (s', t'). This gives the same carrier, finite-window, unary-history, bundle-membership, pointwise hsame, and rational-history classifier surface for the intermediate handoff.

Next compose the observation rows pointwise. For each certified bundled observation n, transitivity of hsame composes $\mathsf{hsame}(s(n), s'(n))$ with $\mathsf{hsame}(s'(n), s''(n))$, and composes $\mathsf{hsame}(t(n), t'(n))$ with $\mathsf{hsame}(t'(n), t''(n))$. Thus the intermediate handoff has the exact pointwise transport data needed for the second step.

Apply Definition 126.185 again, now from (s', t') to (s'', t''). The conclusion is the displayed handoff for (s'', t'') on the original finite bundle $\mathcal{P}$ with no new row outside the consumer-facing observation surface.

The exclusion clause is inherited at both applications of the closure theorem: neither step reads RegSeqRat regularity, completion, quotient, host-index equality, or $\mathsf{Real}^{\uparrow}$ seal data. Therefore the two transported handoffs compose inside the same finite-window pointwise interface. □

Theorem 126.187 (StreamName RegSeqRat pointwise hsame transport confluence)**.** *Let $s, t, u, v :$* BHist $\to$ BHist *be StreamName rows, and let $\mathcal{P}$:* ProbeBundle *be finite. Let*

$$(s_0, t_0), \ldots, (s_m, t_m),$$
$$(r_0, q_0), \ldots, (r_k, q_k)$$

be two finite pointwise hsame *transport schedules of StreamName row pairs with common endpoints*

$$(s_0, t_0) = (r_0, q_0) = (s, t), \qquad (s_m, t_m) = (r_k, q_k) = (u, v).$$

Assume the handoff for (s, t) is the exact pointwise package of Definition 126.184. Assume further that every adjacent stage of both schedules carries the certified bundled pointwise hsame *rows and preserves that exact pointwise package. Then the two induced handoff packets at (u, v) expose the same* RatStreamNameCarrier *rows, restricted finite-window classifier rows,* UnaryHistory *rows,* InBundle *rows, pointwise* hsame *rows, and pointwise* RatHistoryClassifier *rows. Neither packet exposes a* RegSeqRat *regularity row, completion row, quotient row, host-index equality row, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Proof. Induct over the first finite schedule. The base case is the source handoff itself, whose exact consumer-facing package is supplied by Definition 126.184. For the step case, apply Definition 126.186 to compose the accumulated transport with the next adjacent pointwise hsame stage.

The same induction applies to the second finite schedule. Its base is the same exact package at (s, t), and each step preserves the displayed carrier, finite-window, unary-history, bundle-membership, pointwise hsame, and rational-history classifier rows by the same composition theorem.

Both inductions therefore terminate at the common endpoint (u, v) with the pointwise package of Definition 126.184. Comparing through that endpoint package identifies the exposed carrier, restricted finite-window classifier, UnaryHistory, InBundle, pointwise hsame, and pointwise RatHistoryClassifier rows from the two routes.

Finally apply Definition 126.180 at the common endpoint. The envelope excludes RegSeqRat regularity, completion, quotient, host-index equality, and $\mathsf{Real}^{\uparrow}$ seal rows from the observation handoff, so neither route can expose any row outside the common pointwise StreamName interface. □

Theorem 126.188 (StreamName RegSeqRat restricted pointwise transport confluence)**.** *Let s, t, u, v : BHist $\to$ BHist be StreamName rows, and let $\mathcal{Q} \subseteq \mathcal{P}$ be a certified finite subbundle. Suppose two finite pointwise* hsame *transport schedules for (s, t) have common endpoint (u, v) as in Definition 126.187. After restricting each transported handoff to $\mathcal{Q}$, the two restricted packets expose the same* RatStreamNameCarrier *rows, restricted finite-window classifier rows,* UnaryHistory *rows,* InBundle *rows, pointwise* hsame *rows, and pointwise* RatHistoryClassifier *rows to every* RegSeqRat *handoff and every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer. The restricted packets expose no* RegSeqRat *regularity row, completion row, quotient row, host-index equality row, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Proof. Begin with the two endpoint handoffs supplied by Definition 126.187. That confluence theorem identifies the carrier, finite-window classifier, UnaryHistory, InBundle, pointwise hsame, and RatHistoryClassifier rows at the common endpoint (u, v).

Restrict both endpoint packets along the certified inclusion $\mathcal{Q} \subseteq \mathcal{P}$. By Definition 126.175, restriction forgets only observations outside $\mathcal{Q}$ and preserves every displayed row of the finite-window handoff on the subbundle.

It remains to check the consumer boundary. Apply Definition 126.180 after restriction. The envelope is already stated only in terms of the exposed StreamName observation rows, so it is stable under passing from $\mathcal{P}$ to $\mathcal{Q}$.

Thus every RegSeqRat handoff and every finite-window $\mathsf{Real}^{\uparrow}$ consumer sees the same restricted carrier, classifier, history, membership, pointwise hsame, and rational-history rows through either route. The same envelope excludes regularity, completion, quotient, host-index equality, and $\mathsf{Real}^{\uparrow}$ seal data from the restricted packets. □

Theorem 126.189 (RegSeqRat window stability under StreamName transport)**.** *Let two accepted* $\mathsf{StreamName}^{\uparrow}$ *packets agree pointwise on every finite window inspected by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. If the point certificates for those windows are transported componentwise by* hsame *and replayed through the displayed* Cont *rows of the two StreamName packets, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *window carrier and classifier rows transported through those windows remain stable. The statement is only about the displayed finite StreamName windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows read through them; it introduces no quotient stream equality, selected limit, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite StreamName window row used by the readback. The hypothesis gives pointwise agreement exactly on the window coordinates inspected by the $\mathsf{RegSeqRat}^{\uparrow}$ carrier.

Transport each displayed point certificate componentwise by hsame. Replaying the corresponding Cont rows preserves the same window source, so the readback cannot switch to a hidden stream representative or to a larger tail object.

Finally read the $\mathsf{RegSeqRat}^{\uparrow}$ carrier and classifier rows only through those transported windows. Since no quotient stream equality, selected limit, host real equality, or ambient completeness coordinate is present in the route, the transported window carrier and classifier rows remain stable. □

Theorem 126.190 (StreamName RegSeqRat denominator window separation)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows and let $\mathcal{P} : \mathsf{ProbeBundle}$ be a finite unary observation bundle. For every certified observation n with $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$, the denominator and dyadic observation rows consumed by a $\mathsf{RegSeqRat}$ finite-window handoff factor through only the visible packet*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}),$$
$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}), \qquad \mathsf{RatHistoryClassifier}(s(n), t(n)).$$

The packet may be read by the $\mathsf{RegSeqRat}$ dyadic observation coverage rows, including Definition 38.562, but it exports no $\mathsf{RegSeqRat}$ regularity row, dyadic seal datum, $\mathsf{Real}^{\uparrow}$ seal classifier, completion object, quotient row, or host-index equality. Those objects remain on the downstream side of the $\mathsf{RegSeqRat}$ and Real-and-completion certificates.

Proof. Project the finite StreamName bundle by Definition 126.170. For the selected certified observation n, the projection exposes the finite-window classifier, the unary-history row, the bundle-membership row, and the pointwise rational-history classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Stabilize the exposed row by Definition 126.171. This step permits the consumer to transport the same certified observation along the pointwise hsame rows already visible in the bundle, without importing any regularity or seal evidence from the later $\mathsf{RegSeqRat}$ or $\mathsf{Real}^{\uparrow}$ boundary.

Now read the denominator window through the Real/RegSeqRat coverage surface. Apply Definition 38.562 to the selected unary observation and its pointwise rational classifier. The dyadic endpoint rows are therefore obtained from the displayed finite observation packet.

Finally repack the consumer boundary using Definition 126.179. That theorem keeps $\mathsf{RegSeqRat}$ regularity, dyadic seal data, the $\mathsf{Real}^{\uparrow}$ seal classifier, completion structure, quotient structure, and host-index equality outside the StreamName source packet, so the denominator window has exactly the stated dependency surface. □

Theorem 126.191 (StreamName RegSeqRat denominator subwindow transport)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, and let $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$ be finite observation bundles. Suppose $\mathcal{Q}$ is certified as a subwindow of $\mathcal{P}$ by*

$$\forall m,\ \mathsf{InBundle}(m, \mathcal{Q}) \wedge \mathsf{UnaryHistory}(m) \to \mathsf{InBundle}(m, \mathcal{P}) \wedge \mathsf{UnaryHistory}(m).$$

Then every denominator-window packet accepted over $\mathcal{P}$ restricts to a denominator-window packet over $\mathcal{Q}$ whose public surface is exactly

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}),$$
$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{Q}), \qquad \mathsf{RatHistoryClassifier}(s(n), t(n)),$$
$$\textit{the downstream dyadic observation coverage row.}$$

The restricted packet exports no $\mathsf{RegSeqRat}$ regularity row, $\mathsf{Real}^{\uparrow}$ seal classifier, completion object, quotient row, or host-index equality.

Proof. First restrict the finite StreamName classifier. The certified inclusion of $\mathcal{Q}$ in $\mathcal{P}$ is precisely the input required by Definition 126.9, so the accepted $\mathcal{P}$ packet supplies $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t,$

Now apply Definition 126.190 inside the subwindow $\mathcal{Q}$. For each certified observation n in $\mathcal{Q}$, this exposes the unary-history row, the $\mathcal{Q}$ bundle-membership row, and the pointwise rational-history classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Next read the endpoint through Definition 38.562. The dyadic observation row is therefore attached to the same subwindow observation rather than to any ambient host index.

Finally use Definition 126.179. It keeps the $\mathsf{RegSeqRat}$ regularity row, the $\mathsf{Real}^{\uparrow}$ seal classifier, completion, quotienting, and host-index equality outside the finite StreamName source packet. The transported subwindow packet is exactly the displayed public surface and no larger. □

Theorem 126.192 (StreamName RegSeqRat denominator overlap gluing)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, and let $\mathcal{P}, \mathcal{Q}, \mathcal{O} : \mathsf{ProbeBundle}$ be finite observation windows. Suppose $\mathcal{O}$ is certified as a subwindow of both $\mathcal{P}$ and $\mathcal{Q}$, and suppose denominator-window packets are accepted over $\mathcal{P}$ and $\mathcal{Q}$. If the two packets agree on the public overlap rows selected by $\mathcal{O}$, namely the restricted $\mathsf{RatStreamNameFiniteWindowClassifier}$ row, every certified $\mathsf{UnaryHistory}(n)$ row, each $\mathsf{InBundle}(n, \mathcal{O})$ row, and the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ row, then the two packets glue to a denominator-window packet over $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$.*

The glued packet restricts back to the same public denominator selector on $\mathcal{P}$, on $\mathcal{Q}$, and on $\mathcal{O}$. Its visible selector surface is exactly

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})),$$

$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})), \qquad \mathsf{RatHistoryClassifier}(s(n), t(n)),$$

Cont *routes for the displayed restrictions,* hsame *overlap agreement rows,* Pkg *provenance for the two acc*

It adds no $\mathsf{RegSeqRat}$ *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, host-index equality, or non-public selector.*

Proof. First separate the two ambient denominator windows. Apply Definition 126.190 to the accepted packets over $\mathcal{P}$ and $\mathcal{Q}$. This exposes only the finite StreamName classifier, unary-history, bundle-membership, pointwise rational-history, Cont, hsame, and Pkg rows that can participate in a public denominator selector.

Next restrict both packets to the overlap. Use Definition 126.191 along the two certified inclusions $\mathcal{O} \subseteq \mathcal{P}$ and $\mathcal{O} \subseteq \mathcal{Q}$. The resulting overlap packets have the same public selector coordinates by the assumed agreement on the restricted classifier, unary rows, membership rows, and pointwise rational-history rows.

Now assemble the union window. Apply Definition 126.193 to the two ambient packets, using the overlap agreement to identify the selected public rows that appear from both sides. The finite union packet is therefore carried by $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$ and by the displayed Cont, hsame, and Pkg ledger.

Finally read back the restrictions. Apply Definition 126.191 to the union packet along the evident subwindow inclusions of $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{O}$ into $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$. The readback packets are the original public denominator selectors on $\mathcal{P}$ and $\mathcal{Q}$, and their common restriction to $\mathcal{O}$ is the agreed overlap selector. The separation theorem supplies the stated exclusions. □

Theorem 126.193 (StreamName RegSeqRat denominator-window union stability)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, and let $\mathcal{P}, \mathcal{Q} : \mathsf{ProbeBundle}$ be finite observation windows. Suppose denominator-window packets are accepted over both $\mathcal{P}$ and $\mathcal{Q}$. Then they assemble over*

the certified union window $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$*, and every certified observation* n *in that union has exactly the public coordinates*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})),$$
$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})), \qquad \mathsf{RatHistoryClassifier}(s(n), t(n)),$$
the downstream dyadic observation coverage row.

The assembled packet exports no $\mathsf{RegSeqRat}$ *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion datum, quotient row, or host-index equality.*

Proof. First form the union classifier. The two accepted denominator-window packets expose $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ and $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q})$ by Definition 126.190. The union case of Definition 126.25 assembles these two rows into the classifier over $\mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q})$.

Now fix a certified observation n in the union window. Applying Definition 126.190 to the union classifier exposes $\mathsf{UnaryHistory}(n)$, the displayed $\mathsf{InBundle}(n, \mathsf{bundleAppend}(\mathcal{P}, \mathcal{Q}))$ row, and $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

The subwindow direction is inherited by Definition 126.191. It says that any consumer that later restricts the certified union packet back to $\mathcal{P}$ or $\mathcal{Q}$ sees exactly the same public denominator-window surface, with no additional host-coordinate witness.

Finally apply Definition 38.562 to read the downstream dyadic observation row, and use Definition 126.179 to bracket the consumer boundary. Those two rows keep $\mathsf{RegSeqRat}$ regularity, $\mathsf{Real}^{\uparrow}$ seal classifiers, completion data, quotient rows, and host-index equality outside the assembled StreamName packet. □

Theorem 126.194 (StreamName RealUp denominator-window normal form)**.** *A denominator window handed from* $\mathsf{StreamName}^{\uparrow}$ *to both* $\mathsf{RegSeqRat}$ *and a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer has the following normal form and no other public coordinates:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \qquad \mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the downstream dyadic observation coverage row.}$$

The normal form is exactly the finite StreamName classifier, certified unary history, bundle-membership row, pointwise rational-history classifier, and the dyadic observation row consumed downstream. It excludes $\mathsf{RegSeqRat}$ *regularity, the* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion structure, quotient structure, and host-index equality.*

Proof. Start with Definition 126.190. It projects a certified observation n in the finite bundle $\mathcal{P}$ to the finite StreamName classifier, the unary-history certificate, bundle membership, and the pointwise rational-history classifier.

Next read the downstream dyadic observation through Definition 38.562. That theorem uses the exposed pointwise rational row as the finite observation consumed by the $\mathsf{RegSeqRat}$ side of the handoff.

Finally apply Definition 126.179. It keeps regularity, the $\mathsf{Real}^{\uparrow}$ seal classifier, completion, quotienting, and host-index equality outside the finite StreamName window. The remaining rows are precisely the displayed tuple, which is the claimed normal form. □

Theorem 126.195 (StreamName pointwise denominator-window determinacy)**.** *Fix StreamName rows* $s, t : \mathsf{BHist} \to \mathsf{BHist}$*, a finite* $\mathsf{ProbeBundle}\ \mathcal{P}$*, and a certified observation* n *with* $\mathsf{UnaryHistory}(n)$

and $\mathsf{InBundle}(n, \mathcal{P})$*. If two public denominator-window packets are accepted by both the* $\mathsf{RegSeqRat}$ *handoff and the finite-window* $\mathsf{Real}^{\uparrow}$ *consumer for this same certified observation, then their displayed public coordinates are the same:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \qquad \mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the downstream dyadic observation coverage row.}$$

This determinacy is only a finite-packet statement. It exports no $\mathsf{RegSeqRat}$ *regularity,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Proof. The separation theorem Definition 126.190 first fixes the public dependency surface of any accepted denominator-window packet. For the certified observation n, the handoff can expose only the finite StreamName classifier, the unary-history row, the bundle-membership row, and the pointwise rational-history classifier.

The normal-form theorem Definition 126.194 then says that a packet accepted by the finite-window $\mathsf{Real}^{\uparrow}$ consumer has no additional public coordinate beyond the downstream dyadic observation coverage row. Thus two accepted packets for the same s, t, $\mathcal{P}$, and n must display the same four source coordinates.

It remains to identify the downstream row. The pointwise source coordinate is the one delivered by Definition 126.170, and the finite dyadic-observation row read from it is exactly the row supplied by Definition 38.562. Hence the consumer-side coordinate is also fixed by the same certified observation.

Combining these two fixed surfaces gives one public finite packet. The exclusions in the separation and normal-form statements keep $\mathsf{RegSeqRat}$ regularity, the $\mathsf{Real}^{\uparrow}$ seal classifier, completion, quotienting, and host-index equality outside the packet, so none of them can distinguish the two accepted presentations. □

Theorem 126.196 (StreamName RegSeqRat denominator pullback coverage)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be StreamName rows, let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite unary observation bundle, and let* n *be a certified unary observation. A public denominator-window packet is accepted by the* $\mathsf{RegSeqRat}$ *handoff and by the finite-window* $\mathsf{Real}^{\uparrow}$ *consumer exactly when it factors through the finite pullback surface*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \qquad \mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the downstream dyadic observation coverage row.}$$

Equivalently, the denominator packet is covered by the StreamName finite-window classifier pulled back along the certified unary row and then read by the $\mathsf{RegSeqRat}$ *finite-window dyadic observation row. The accepted packet exports no* $\mathsf{RegSeqRat}$ *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Proof. For the forward direction, start with an accepted public denominator-window packet. By Definition 126.190, the source surface can expose only the finite StreamName classifier, the certified unary history, the bundle-membership row, and the pointwise rational-history classifier.

The finite-window $\mathsf{Real}^{\uparrow}$ consumer then normalizes the same presentation by Definition 126.194. This adds only the downstream dyadic observation coverage row and keeps regularity, sealing, completion, quotienting, and host-index equality outside the public packet.

It remains to identify the pointwise row used by the handoff. Apply Definition 126.195 to the accepted packet over the same s, t, $\mathcal{P}$, and n. The pointwise coordinate is therefore fixed

as $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the downstream coordinate is the row supplied by Definition 38.562.

For the reverse direction, suppose the displayed pullback surface is given. The finite StreamName classifier and the certified unary row supply the pointwise handoff input, and Definition 38.562 reads that input as the dyadic observation coverage row. Repacking these rows through the normal form gives an accepted public denominator-window packet for both consumers, with exactly the stated exclusions. □

Theorem 126.197 (StreamName RegSeqRat denominator-window subbundle inheritance)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P}'$ and $\mathcal{P}$ be finite observation bundles, and let n be a certified observation with $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P}')$. Suppose $\mathcal{P}'$ is displayed as a subbundle of $\mathcal{P}$. If the public denominator-window packet over $\mathcal{P}$ is accepted by the* RegSeqRat *handoff and by the finite-window $\mathsf{Real}^{\uparrow}$ consumer, then the restricted packet over $\mathcal{P}'$ has exactly the same public coordinates, with the ambient membership row replaced by the certified subbundle membership row:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}'),$$
$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}'),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the same downstream dyadic observation coverage row.}$$

The inheritance exports no RegSeqRat *regularity row, $\mathsf{Real}^{\uparrow}$ seal classifier, completion object, quotient row, or host-index equality.*

Proof. First restrict the finite-window classifier. The displayed subbundle inclusion is the input used by Definition 126.302, so the accepted packet over $\mathcal{P}$ presents the finite StreamName classifier over $\mathcal{P}'$.

Now read the certified observation through the pointwise handoff. Apply Definition 126.170 to the same row n; the unary-history row and the pointwise classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$ are unchanged, while the displayed membership coordinate is the subbundle row $\mathsf{InBundle}(n, \mathcal{P}')$.

Next identify the accepted public surface. The separation theorem Definition 126.190 and the normal form theorem Definition 126.194 leave only the finite classifier, unary row, bundle-membership row, pointwise rational-history classifier, and downstream dyadic observation coverage row visible to the two consumers.

Finally apply Definition 126.195. Since the restricted presentation and the ambient presentation are accepted for the same s, t, and certified observation n, their visible pointwise and downstream coordinates cannot differ. The exclusion clauses in the separation and normal form statements keep regularity, sealing, completion, quotienting, and host-index equality outside the inherited packet. □

Theorem 126.198 (StreamName RegSeqRat denominator common refinement)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be StreamName rows, let $\mathcal{P}_0$ and $\mathcal{P}_1$ be finite observation bundles, and let n be a certified observation with $\mathsf{UnaryHistory}(n)$, $\mathsf{InBundle}(n, \mathcal{P}_0)$, and $\mathsf{InBundle}(n, \mathcal{P}_1)$. Suppose the* RegSeqRat *handoff and the finite-window $\mathsf{Real}^{\uparrow}$ consumer accept denominator-window packets over both $\mathcal{P}_0$ and $\mathcal{P}_1$. Then the overlap admits a common displayed subbundle $\mathcal{Q}$ and a refinement packet over $\mathcal{Q}$ whose visible coordinates are exactly*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}),$$
$$\mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{Q}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the downstream dyadic observation coverage row.}$$

The refinement exports no RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Proof. First form the visible overlap. Apply Definition 126.192 to the two accepted denominator-window packets. The gluing row identifies the common finite StreamName classifier on the shared observation coordinates and keeps the certified unary row n visible on both sides.

Next display the overlap as an actual consumer boundary. Use Definition 126.196 to read the same overlap through the finite-window $\mathsf{Real}^{\uparrow}$ pullback, adding only the downstream dyadic observation coverage row needed by that handoff.

Now restrict to the displayed common subbundle $\mathcal{Q}$. By Definition 126.197, the two accepted packets inherit to $\mathcal{Q}$ with the same finite classifier, the same unary row, and the membership row $\mathsf{InBundle}(n, \mathcal{Q})$.

Finally apply Definition 126.195. The pointwise coordinate on the common packet is forced to be $\mathsf{RatHistoryClassifier}(s(n), t(n))$, and the pullback row supplies the same downstream dyadic observation coverage. The separation clauses from the gluing, pullback, and inherited packets exclude all non-visible regularity, sealing, completion, quotient, and host-index fields. □

Theorem 126.199 (StreamName RegSeqRat denominator refinement descent)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be the carried StreamName rows, let* $\mathcal{P}_0$ *and* $\mathcal{P}_1$ *be finite observation bundles, and let* $\mathcal{Q}$ *be a common-refinement bundle for them. Suppose a refinement packet over* $\mathcal{Q}$ *has endpoint rows satisfying the denominator-window classifier, and suppose the selected overlap rows are the rows named by the shared* ProbeBundle*. Then this refinement descends to stable denominator-window evidence over each original window* $\mathcal{P}_i$*. The descended packets preserve the pulled-back* BHist *denominator endpoints under* hsame*, and their overlap restrictions agree on the rows selected by the shared* ProbeBundle*.*

Proof. Project the common-refinement packet to each original window. The existence of the refinement bundle and its visible coordinates are supplied by Definition 126.198, while Definition 126.190 identifies the only denominator-window rows available to those projections.

Next inherit the packet along the displayed inclusions. Apply Definition 126.197 to read the $\mathcal{Q}$ packet as a subbundle packet over each $\mathcal{P}_i$, and then apply Definition 126.191 to move the finite-window classifier, unary row, bundle-membership row, pointwise rational-history classifier, and downstream dyadic coverage back to the original windows.

The pulled-back endpoints are stable because Definition 126.196 gives the same BHist denominator endpoints on the finite-window $\mathsf{Real}^{\uparrow}$ consumer boundary. The transported packets therefore compare these endpoints by the inherited hsame rows rather than by a host-index equality.

Finally compare the two descended packets on the shared rows. Use Definition 126.192 to identify the overlap selected by the common ProbeBundle, and use Definition 126.193 to keep the two original-window packets stable when they are read together. The separation theorem excludes any additional regularity, completion, quotient, or sealing field, so the descended evidence is exactly the stable original-window evidence carried by those visible rows. □

Theorem 126.200 (RegSeqRat denominator-window dyadic source exactness)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be the carried StreamName rows, let* $\mathcal{P}_0$ *and* $\mathcal{P}_1$ *be finite denominator observation bundles, and let* $\mathcal{Q}$ *be a displayed common-refinement bundle selected by a shared* ProbeBundle*. Suppose the refinement packet over* $\mathcal{Q}$ *descends to stable denominator-window evidence for the two original windows. Then the descended evidence exposes precisely the two selected DyadicRatCore observation endpoint rows consumed by* RegSeqRat*, the StreamName classifier row* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}$ *the unary index row* $\mathsf{UnaryHistory}(n)$*, and the membership row* $\mathsf{InBundle}(n, \mathcal{Q})$ *for the shared* ProbeBundle*.*

No $\mathsf{Real}^{\uparrow}$ *seal, completion object, quotient row, or host-index equality is readable from the descended denominator-window evidence.*

Proof. Start with denominator-window separation. By Definition 126.190, every accepted denominator packet exposes only the finite StreamName classifier, the unary observation row, the bundle-membership row, the pointwise rational row, and the downstream dyadic observation rows.

Next use pullback coverage and the common-refinement theorem. The pullback coverage result Definition 126.196 identifies the two DyadicRatCore endpoint observations on the finite-window consumer boundary, while Definition 126.198 places those rows on the displayed common bundle $\mathcal{Q}$ together with the StreamName classifier and the selected observation rows.

Now inherit along the subbundle inclusions. Apply Definition 126.197 to keep the $\mathcal{Q}$ rows stable when read as descendants of the original windows, and apply Definition 126.195 to force the pointwise rational-history row to be the one determined by the same selected endpoint observations.

Finally apply the refinement descent theorem Definition 126.199. Descent preserves the two dyadic endpoint rows, the finite StreamName classifier, the unary index row, and the ProbeBundle membership row. The separation and pullback clauses exclude every $\mathsf{Real}^{\uparrow}$ seal, completion, quotient, and host-index equality, so the listed rows are exactly the source rows consumed by RegSeqRat. □

126.11 StreamName RegSeqRat RealUp Tail-Selector Transport

Definition 126.201 (StreamName RegSeqRat RealUp tail-selector transport packet)**.** *A StreamName RegSeqRat RealUp tail-selector transport packet from a selected finite* $\mathsf{Real}^{\uparrow}$ *window* W_0 *to a selected finite* $\mathsf{Real}^{\uparrow}$ *window* W_1 *is the finite ledger consisting exactly of the following rows:*

1. *a shared* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *row for the public* $\mathsf{StreamName}^{\uparrow}$ *rational stream endpoints* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *over the selected finite* ProbeBundle *window* $\mathcal{P}$*;*
2. *for each certified selected observation* n*, the bundled* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$ *rows, together with the pointwise* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *row exposed by the StreamName observation surface;*
3. *the accepted* RegSeqRat *spine row selected by those pointwise rational history rows and by the finite-window handoff;*
4. *the matching DyadicRatCore endpoint rows read by the two selected* $\mathsf{Real}^{\uparrow}$ *windows from that same* RegSeqRat *spine; and*
5. *the common finite* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *ledger joining the StreamName endpoint rows, the accepted* RegSeqRat *spine row, the DyadicRatCore endpoint rows, and the two selected* $\mathsf{Real}^{\uparrow}$ *readback packets.*

The packet exposes no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order operation, standard bridge row, or selector not already contained in the displayed finite ledger.

Theorem 126.202 (StreamName RegSeqRat RealUp tail-selector transport)**.** *Let* W_0 *and* W_1 *be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same public* $\mathsf{StreamName}^{\uparrow}$ *rational stream endpoints and the same finite* ProbeBundle *window. If* W_0 *and* W_1 *are connected by a StreamName RegSeqRat RealUp tail-selector transport packet, then the selected* RegSeqRat *tail-classifier readback transported from* W_0 *by the packet agrees with the selected* RegSeqRat *tail-classifier readback read from* W_1*.*

The agreement is a finite-window agreement over the displayed RatStreamNameFiniteWindowClassifier *row, the bundled* UnaryHistory/InBundle *rows, the accepted* RegSeqRat *spine row, the matching DyadicRatCore endpoint rows, and the shared* Cont/hsame/Pkg/RealLedgerPolicy *ledger. It does not identify quotient objects, completion objects, Classical representatives, metric-completeness data, arithmetic or order rows, standard bridge rows, or external selector rows.*

Proof. First read the StreamName side of the packet. Apply Definition 126.171 to the displayed RatStreamNameFiniteWindowClassifier row and to each bundled UnaryHistory/InBundle observation. The selected pointwise rational-history rows therefore stay fixed under the packet's hsame transport.

Next form the uniform finite-window handoff. Apply Definition 126.111 to the stabilized StreamName rows and the accepted RegSeqRat spine row. This gives one finite seal surface carrying the same Cont, hsame, Pkg, and DyadicRatCore endpoint rows for both selected windows.

Now read the RealUp boundary through that finite seal. Use Definition 38.354. The two selected $\mathsf{Real}^{\uparrow}$ windows expose the same visible endpoint readback rows and the same RealLedgerPolicy row on the selected finite window.

Finally apply Definition 38.360. Its hypotheses are exactly the common StreamName classifier, the accepted RegSeqRat spine, the matching DyadicRatCore endpoint rows, and the shared finite ledger established above. Hence the transported tail-classifier readback from W_0 is the same selected readback as the one read from W_1, and the scoped exactness theorem supplies the stated exclusions. □

Theorem 126.203 (StreamName tail-selector pointwise transport)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over public* $\mathsf{StreamName}^{\uparrow}$ *rational stream endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$ and the same finite* ProbeBundle *window $\mathcal{P}$. Suppose an accepted StreamName RegSeqRat RealUp tail-selector transport packet connects W_0 and W_1. Then for every certified observation n in $\mathcal{P}$, the transported pointwise row handed from the StreamName window to the* RegSeqRat *finite-window consumer is exactly the tuple*

$$\mathsf{UnaryHistory}(n), \quad \mathsf{InBundle}(n, \mathcal{P}), \quad \mathsf{RatHistoryClassifier}(s(n), t(n)).$$

This tuple is the only rowwise data supplied to the RegSeqRat *finite-window consumer at n. The handoff supplies no quotient row, no completion row, no host-index equality row, no arithmetic or order row, and no selector external to the displayed packet.*

Proof. First fix a certified observation n in $\mathcal{P}$. Apply Definition 126.171 to the accepted transport packet. The StreamName observation row at n is therefore stable as the bundled $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$ data together with the pointwise rational-history classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Next apply Definition 126.170 to that stable row. Its handoff clause is rowwise: the RegSeqRat finite-window consumer receives the named unary-history row, the named bundle-membership row, and the named rational-history classifier row for the same observation n.

Finally use Definition 126.202. The transport theorem fixes the finite-window agreement connecting W_0 and W_1, so the rowwise handoff just identified is the pointwise component of the transported window agreement. Since the three cited theorems expose only the displayed StreamName and RegSeqRat rows, quotient rows, completion rows, host-index equality rows, arithmetic or order rows, and external selector rows remain outside the handoff. □

Theorem 126.204 (StreamName RegSeqRat tail-selector handoff exactness)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same public* $\mathsf{StreamName}^{\uparrow}$ *rational*

stream endpoints and the same finite ProbeBundle *window. If W_0 and W_1 are connected by an accepted StreamName RegSeqRat RealUp tail-selector transport packet, then the packet exported across the Real-and-Completion boundary decomposes exactly into the following public finite rows:*

1. *the displayed* RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{P})$ *row for the public* StreamName$^{\uparrow}$ *endpoints;*
2. *the bundled* UnaryHistory(n) *and* InBundle$(n, \mathcal{P})$ *rows for each selected observation n;*
3. *the pointwise* RatHistoryClassifier$(s(n), t(n))$ *rows read from those selected observations;*
4. *the accepted* RegSeqRat *spine row determined by the finite-window StreamName handoff;*
5. *the DyadicRatCore endpoint rows read from that same accepted spine by the selected* Real$^{\uparrow}$ *windows; and*
6. *the finite* Real$^{\uparrow}$ *ledger rows for* Cont, hsame, Pkg, *and* RealLedgerPolicy *over the selected window.*

The handoff packet exports no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order operation, or selector beyond the displayed finite rows.

Proof. Apply Definition 126.202 to the accepted transport packet. The resulting agreement fixes the selected RegSeqRat tail-classifier readback on the common finite window and restricts the transport to the rows displayed in the packet definition.

Next use Definition 126.171. It identifies the StreamName contribution to the packet as the public finite-window classifier together with the bundled UnaryHistory/InBundle observations and their pointwise rational-history classifier rows.

Use Definition 126.111 to read the accepted RegSeqRat spine from those stabilized StreamName rows. The seal supplies the single finite spine row consumed by the two selected endpoint windows and does not introduce a quotient or completion representative.

Finally apply Definition 38.354. The RealUp readback is exactly the DyadicRatCore endpoint surface and the finite Cont/hsame/Pkg/RealLedgerPolicy ledger over the selected window. Combining these four exactness statements gives precisely the six displayed families of rows, and the exclusion clauses follow because none of the cited theorems exports quotient data, completion data, metric completeness, arithmetic or order operations, or additional selectors. □

Theorem 126.205 (StreamName RegSeqRat tail-selector RealUp window factorization)**.** *Let W be a selected finite* Real$^{\uparrow}$ *observation-window package over public* StreamName$^{\uparrow}$ *rational stream endpoints s, t : BHist $\to$ BHist and a finite* ProbeBundle *window $\mathcal{P}$. If W is accepted through a StreamName RegSeqRat RealUp tail-selector transport packet, then every consumer-facing* Real$^{\uparrow}$ *row exposed by W factors through exactly the finite public surface consisting of:*

1. *the displayed* RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{P})$ *row;*
2. *for each selected observation n, the bundled* UnaryHistory(n) *and* InBundle$(n, \mathcal{P})$ *rows;*
3. *the pointwise* RatHistoryClassifier$(s(n), t(n))$ *rows;*
4. *the accepted* RegSeqRat *spine row selected by those pointwise rational history rows;*
5. *the DyadicRatCore endpoint rows read from that same accepted spine; and*
6. *the finite* Cont, hsame, Pkg, *and* RealLedgerPolicy *ledger over the selected window.*

Thus the Real-and-Completion boundary sees only this finite factorization. The factorization exposes no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order operation, standard bridge row, or hidden selector row.

Proof. First apply Definition 126.204 to the accepted transport packet. The handoff exactness result displays the packet as precisely the StreamName finite-window classifier, the selected UnaryHistory/InBundle observations, the pointwise RatHistoryClassifier rows, the accepted RegSeqRat spine, the DyadicRatCore endpoint rows, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

Next apply Definition 38.354 to that displayed handoff surface. This reads every selected $\mathsf{Real}^{\uparrow}$ observation through the same finite StreamName-to-RegSeqRat window and identifies the consumer-facing rows with the DyadicRatCore endpoint readback and the Cont/hsame/Pkg/RealLedgerPolicy ledger.

Finally apply Definition 38.360. The selected tail row is therefore the same finite public surface already named by the accepted RegSeqRat spine and its pointwise rational-history classifier rows. Since each cited exactness theorem exports only the listed finite rows, no quotient, completion, Classical representative, metric-completeness, arithmetic or order, standard-bridge, or hidden-selector row can appear in the factorization. □

Theorem 126.206 (StreamName RegSeqRat RealUp tail-selector transport composition)**.** *Let W_0, W_1, and W_2 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same public* $\mathsf{StreamName}^{\uparrow}$ *rational stream endpoints, and let the finite* ProbeBundle *windows used by the W_0 to W_1 and W_1 to W_2 packets be compatible on their shared selected observations. If W_0 is connected to W_1 by a StreamName RegSeqRat RealUp tail-selector transport packet and W_1 is connected to W_2 by a second such packet, then the concatenated visible ledger is again a StreamName RegSeqRat RealUp tail-selector transport packet from W_0 to W_2.*

The concatenated packet uses the composed Cont *rows, the composed displayed* hsame *rows, the concatenated* Pkg *provenance, and the combined* RealLedgerPolicy *ledger. It transports the selected* RegSeqRat *tail classifier read from W_0 to the selected finite-window readback at W_2. The assertion is only a finite-window transport statement over the displayed* ProbeBundle *windows and the accepted* RegSeqRat *spine; it adds no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order operation, standard bridge row, or hidden selector row.*

Proof. First apply Definition 126.202 to the W_0 to W_1 packet. This identifies the selected RegSeqRat tail classifier transported from W_0 with the selected finite-window readback visible at W_1.

Next apply the same theorem to the W_1 to W_2 packet. The compatibility of the displayed ProbeBundle windows makes the W_1 readback row used by the first packet the same finite row consumed by the second packet.

Compose the visible transport rows. The hsame classifier rows compose on the selected StreamName observations, on the accepted RegSeqRat spine, and on the matching DyadicRatCore endpoint rows; the Cont rows concatenate the finite routes; the Pkg rows concatenate the displayed provenance; and the RealLedgerPolicy rows remain a single visible finite ledger over the combined window.

Finally use Definition 38.360 on the combined finite-window row. The exactness theorem identifies the final row produced by the composed ledger with the selected RegSeqRat tail-classifier readback at W_2, so the concatenated ledger is precisely the required transport packet from W_0 to W_2 with the stated exclusions. □

Theorem 126.207 (StreamName RegSeqRat RealUp tail denominator selector)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same StreamName endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$, the same finite* ProbeBundle *window $\mathcal{P}$, and the same accepted* RegSeqRat *spine row. Suppose they are connected by a StreamName RegSeqRat RealUp tail-selector transport*

packet. Then the packet carries the public denominator-window selector selected by W_0 to the same public denominator-window selector read from W_1.

Equivalently, for every certified selected observation n with $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$*, the transported tail readback has the same finite public selector surface*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}), \qquad \mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{P}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the downstream dyadic observation coverage row}$$

as the selector read by the finite $\mathsf{Real}^{\uparrow}$ *ledger of W_1. The conclusion is only a finite selector statement over the accepted* $\mathsf{RegSeqRat}$ *spine and the finite* $\mathsf{Real}^{\uparrow}$ *ledger; it adds no quotient row, completion object, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or external selector.*

Proof. First apply Definition 126.202. The transport packet identifies the selected $\mathsf{RegSeqRat}$ tail-classifier readback from W_0 with the selected readback exposed by W_1, using only the shared finite StreamName classifier, the bundled unary observations, the accepted $\mathsf{RegSeqRat}$ spine, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

Next isolate the denominator selector inside that transported readback. By Definition 126.190, each certified observation n exposes only the displayed $\mathsf{RatStreamNameFiniteWindowClassifier}$ row, the $\mathsf{UnaryHistory}(n)$ row, the $\mathsf{InBundle}(n, \mathcal{P})$ row, and the pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ row on the $\mathsf{RegSeqRat}$ side.

Now read the same selector through the finite $\mathsf{Real}^{\uparrow}$ consumer. Apply Definition 126.194. It says that the finite $\mathsf{Real}^{\uparrow}$ ledger contributes exactly the downstream dyadic observation coverage row to that public denominator-window surface and no regularity, seal, completion, quotient, or host-index coordinate.

Finally compare the two presentations by Definition 126.195. Both are accepted over the same s, t, $\mathcal{P}$, certified observation n, and accepted $\mathsf{RegSeqRat}$ spine, so the transported selector from W_0 and the selector read from W_1 have the same public coordinates. The exclusion clauses in the transport, separation, normal-form, and determinacy statements give the stated boundary. □

Theorem 126.208 (StreamName RegSeqRat RealUp tail denominator selector subwindow)**.** *Let W_0 and W_1 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same StreamName endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$, the same finite* $\mathsf{ProbeBundle}$ *window $\mathcal{P}$, and the same accepted* $\mathsf{RegSeqRat}$ *spine row. Suppose a StreamName RegSeqRat RealUp tail-selector transport packet carries the public denominator-window selector from W_0 to W_1. Let $\mathcal{Q}$ be a certified subwindow of $\mathcal{P}$. Then the restrictions of W_0, W_1, and the transport packet to $\mathcal{Q}$ carry the same public denominator-window selector on $\mathcal{Q}$.*

For each certified selected observation n with $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{Q})$*, the subwindow selector conclusion reads only*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}), \qquad \mathsf{UnaryHistory}(n), \qquad \mathsf{InBundle}(n, \mathcal{Q}),$$
$$\mathsf{RatHistoryClassifier}(s(n), t(n)), \qquad \textit{the accepted } \mathsf{RegSeqRat} \textit{ spine}, \qquad \textit{the finite } \mathsf{Real}^{\uparrow} \textit{ ledger restricted to } \mathcal{Q}.$$

It introduces no observation outside $\mathcal{Q}$, quotient row, completion object, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or external selector.

Proof. First restrict the two selected $\mathsf{Real}^{\uparrow}$ windows. Apply Definition 38.172 to the finite $\mathsf{ProbeBundle}$ inclusion $\mathcal{Q} \subseteq \mathcal{P}$. The result is a selected finite-window package for each W_i whose visible ledger is precisely the $\mathsf{Real}^{\uparrow}$ ledger over the certified observations in $\mathcal{Q}$.

Next restrict the StreamName finite-window face of the packet. The handoff rows Definition 126.9 and Definition 126.35 replace the ambient classifier over $\mathcal{P}$ by $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}$ and keep only the rows whose unary index lies in $\mathsf{InBundle}(-, \mathcal{Q})$.

Now apply Definition 126.207 to the restricted packet. The accepted $\mathsf{RegSeqRat}$ spine is unchanged, while the denominator selector is read through the restricted StreamName classifier and the restricted finite $\mathsf{Real}^{\uparrow}$ ledger.

The displayed conclusion follows because every coordinate it reads is one of $\mathsf{RatStreamNameFiniteWindowClassifie}$ $\mathsf{UnaryHistory}$, $\mathsf{InBundle}$, pointwise $\mathsf{RatHistoryClassifier}$, the accepted $\mathsf{RegSeqRat}$ spine, or the finite $\mathsf{Real}^{\uparrow}$ ledger on $\mathcal{Q}$. No row from the complement of $\mathcal{Q}$ remains available after the subwindow extraction. □

Theorem 126.209 (StreamName RegSeqRat RealUp tail denominator selector composition)**.** *Let W_0, W_1, and W_2 be selected finite $\mathsf{Real}^{\uparrow}$ observation-window packages over the same public StreamName endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$ and the same accepted $\mathsf{RegSeqRat}$ spine row. Suppose the W_0 to W_1 packet and the W_1 to W_2 packet are compatible StreamName RegSeqRat RealUp tail-selector transport packets, and suppose each packet carries the public denominator-window selector between its endpoints. Then the two packets compose to carry the same public denominator-window selector from W_0 to W_2.*

The composed readback is entirely finite-window data. It reads only the displayed $\mathsf{RatStreamNameFiniteWindowC}$ *rows, the bundled* $\mathsf{UnaryHistory}$ *and* $\mathsf{InBundle}$ *rows, the pointwise* $\mathsf{RatHistoryClassifier}$ *rows, the accepted* $\mathsf{RegSeqRat}$ *spine, and the finite* $\mathsf{Real}^{\uparrow}$ *ledger carried by the compatible packets and their certified subwindows. It introduces no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or hidden selector row.*

Proof. First compose the underlying tail-selector transport packets. Apply Definition 126.206 to the compatible W_0 to W_1 and W_1 to W_2 packets. The result is a visible finite transport packet from W_0 to W_2 over the same displayed StreamName classifier rows, accepted $\mathsf{RegSeqRat}$ spine, and finite $\mathsf{Real}^{\uparrow}$ ledger.

Next identify the denominator selector transported by the first leg. Apply Definition 126.207 to the W_0 to W_1 packet. This makes the public denominator-window selector read from W_0 the same selector as the one exposed at W_1.

Now identify the denominator selector transported by the second leg. Apply Definition 126.207 to the W_1 to W_2 packet. Compatibility says that the W_1 selector consumed by this leg is the same finite public selector produced by the first leg.

It remains to check that no extra selector coordinates appear when the two windows are read through smaller certified faces. Apply Definition 126.208 to each certified shared subwindow. The conclusion restricts every intermediate readback to $\mathsf{RatStreamNameFiniteWindowClassifier}$, $\mathsf{UnaryHistory}$, $\mathsf{InBundle}$, pointwise $\mathsf{RatHistoryClassifier}$, the accepted $\mathsf{RegSeqRat}$ spine, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

Combining these finite identifications gives a single public denominator-window selector transported from W_0 to W_2. The exclusions come from the three cited transport, denominator-selector, and subwindow statements: no quotient, completion, Classical representative, metric-completeness, arithmetic or order, standard bridge, or hidden selector row is available in the composed readback. □

Theorem 126.210 (StreamName RegSeqRat RealUp tail-selector ledger associativity)**.** *Let W_0, W_1, W_2, W_3 be selected finite $\mathsf{Real}^{\uparrow}$ observation-window packages over the same public StreamName endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$ and the same accepted $\mathsf{RegSeqRat}$ spine row. Suppose the W_0 to W_1, W_1 to W_2, and W_2 to W_3 legs are compatible StreamName RegSeqRat RealUp tail-selector transport packets.*

Then the left and right bracketings of these three legs expose the same finite ledger: the displayed RatStreamNameFiniteWindowClassifier *row, the* UnaryHistory *rows, the* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the finite* $\mathsf{Real}^{\uparrow}$ *ledger, and the visible* Cont/hsame/Pkg *rows carried by the three compatible legs and their certified shared subwindows.*

The ledger identification adds no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or hidden selector row.

Proof. First compare the composed transport packets. Apply Definition 126.206 to the W_0 to W_1 and W_1 to W_2 legs, and then to the resulting W_0 to W_2 packet and the W_2 to W_3 leg. Apply the same theorem to the W_1 to W_2 and W_2 to W_3 legs, and then to the W_0 to W_1 leg and the resulting W_1 to W_3 packet. Both constructions expose the same endpoint rows and the same three named visible legs.

Next compare the public denominator-window coordinates. Apply Definition 126.209 to the two bracketings. The selector rows in the two resulting readbacks are therefore the selectors produced from the same three compatible packets.

Now restrict the comparison to every certified shared face. Apply Definition 126.208 on each subwindow. This identifies the displayed RatStreamNameFiniteWindowClassifier, UnaryHistory, InBundle, pointwise RatHistoryClassifier, accepted RegSeqRat spine, and finite $\mathsf{Real}^{\uparrow}$ ledger rows read by both bracketings.

It remains to compare the visible kernel rows. The hsame rows compose associatively on StreamName observations, on the accepted RegSeqRat spine, and on the dyadic endpoint rows; the Cont routes concatenate associatively; and the Pkg provenance rows name the same three visible legs in either bracketing. Transitivity of these displayed comparisons gives the stated finite-ledger identification and the stated exclusions. □

Theorem 126.211 (StreamName RegSeqRat RealUp tail selector associative readback)**.** *Let* W_0, W_1, W_2, W_3 *be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages over the same public StreamName endpoints* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *and the same accepted* RegSeqRat *spine row. Suppose the* W_0 *to* W_1*,* W_1 *to* W_2*, and* W_2 *to* W_3 *legs are compatible StreamName RegSeqRat RealUp tail-selector transport packets, and suppose each leg carries the public denominator-window selector between its endpoints. Then the two finite-transport bracketings*

$$((W_0 \to W_1) \to W_2) \to W_3,$$
$$W_0 \to (W_1 \to (W_2 \to W_3))$$

read the same public denominator-window selector from W_0 *to* W_3*.*

The common readback surface is only the displayed RatStreamNameFiniteWindowClassifier *row, the* UnaryHistory *rows, the* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the finite* $\mathsf{Real}^{\uparrow}$ *ledger, and the visible* Cont/hsame/Pkg *rows carried by the three compatible legs and their certified shared subwindows. It adds no quotient row, completion row, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or hidden selector row.*

Proof. First compose the transport packets with the left bracketing. Apply Definition 126.206 to the W_0 to W_1 and W_1 to W_2 packets, and then compose the result with the W_2 to W_3 packet. This gives a visible finite packet from W_0 to W_3.

Next compose the same three packets with the right bracketing. Apply the same composition theorem first to the W_1 to W_2 and W_2 to W_3 packets, and then compose the W_0 to W_1 packet with that result. This gives a second visible finite packet from W_0 to W_3 over the same endpoint rows.

Now read the denominator selector through the two-leg selector theorem on each bracketing. For the left bracketing, apply Definition 126.209 to the W_0 to W_2 composite and the W_2 to W_3 leg. For the right bracketing, apply the same theorem to the W_0 to W_1 leg and the W_1 to W_3 composite.

The shared finite faces are identified by Definition 126.208. On every certified subwindow, both bracketings expose the same RatStreamNameFiniteWindowClassifier, UnaryHistory, InBundle, pointwise RatHistoryClassifier, accepted RegSeqRat spine, and finite $\mathsf{Real}^{\uparrow}$ ledger rows.

The displayed ledger comparison is Definition 126.210. It identifies the visible Cont/hsame/Pkg rows and the finite public rows read by both bracketings. This gives the stated selector from W_0 to W_3 and the stated exclusions. □

Theorem 126.212 (StreamName tail-selector composite transport determinacy)**.** *Let s, t : BHist $\to$ BHist be public StreamName endpoints, let Σ be an accepted* RegSeqRat *spine over them, and let W_0, W_1, W_2, W_3 be selected finite* $\mathsf{Real}^{\uparrow}$ *observation-window packages for that spine. Suppose two compatible composite StreamName RegSeqRat RealUp tail-selector transport packets connect the same visible endpoint windows and display the same finite* ProbeBundle *window,* UnaryHistory *rows,* InBundle *rows, pointwise* RatHistoryClassifier *rows, DyadicRatCore endpoint rows, and* Cont, hsame, Pkg, *and* RealLedgerPolicy *ledger rows. Suppose also that the two composites carry the same public denominator-window selector readback at the endpoints. Then the two composites expose the same finite consumer interface to* $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, *and* $\mathsf{Real}^{\uparrow}$.

The determined interface consists exactly of the displayed ProbeBundle *rows,* UnaryHistory *rows,* InBundle *rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine row, the DyadicRatCore endpoint row, the visible* Cont/hsame/Pkg *ledger, and the* RealLedgerPolicy *row. It contains no quotient row, host stream equality row, independent regularity witness,* $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, arithmetic or order row, standard bridge row, or hidden selector row.*

Proof. First normalize the two composites to visible finite packets. Apply Definition 126.206 to the component legs used by each composite. The hypotheses say that the resulting packets have the same endpoint windows, the same displayed ProbeBundle window, and the same accepted RegSeqRat spine.

Next compare the visible ledgers. Apply Definition 126.210 to the two bracketings of the compatible transport legs. This identifies the Cont, hsame, Pkg, and RealLedgerPolicy rows carried by the two finite packets.

Finally read back the endpoint selector. Apply Definition 126.211 to identify the denominator-window selector exposed by the two composites. Tuple exhaustion over the listed finite rows then gives the same consumer interface for $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Real}^{\uparrow}$. Since every step uses only the displayed rows, the excluded quotient, host equality, regularity, completeness, arithmetic, order, bridge, and hidden-selector data are not introduced. □

Theorem 126.213 (StreamName RealUp tail-selector public interface)**.** *Let W be a public* $\mathsf{StreamName}^{\uparrow}$ *package with an accepted* RegSeqRat *spine row, a finite* ProbeBundle *observation window, a public denominator-window selector, and compatible selected* $\mathsf{Real}^{\uparrow}$ *observation windows over the same endpoints. Then the package exports exactly the finite consumer interface available to downstream* $\mathsf{Real}^{\uparrow}$ *and* RegSeqRat *sites: the* RatStreamNameFiniteWindowClassifier *row, the* UnaryHistory *rows, the* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the visible* Cont/hsame/Pkg/RealLedgerPolicy *ledger.*

The exported interface contains no quotient row, completion object, Classical representative, metric-completeness certificate, arithmetic or order row, standard bridge row, or hidden selector.

Proof. First read the source boundary through the public $\mathsf{StreamName}^{\uparrow}$ interface. Apply Definition 126.311. It restricts the countable-name and finite-window rows to the visible public StreamName surface.

Next decompose the Real-and-Completion handoff. Apply Definition 126.204. The handoff rows are exactly the finite StreamName classifier rows, the bundled window observations, the pointwise rational-history rows, the accepted RegSeqRat spine, the DyadicRatCore endpoint rows, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

Now read the selected denominator coordinates. Apply Definition 126.207. The selector visible to the consumer is the public denominator-window selector carried by the same finite handoff ledger.

Finally apply Definition 126.211. It makes the transport readback independent of bracketing, so every compatible selected $\mathsf{Real}^{\uparrow}$ observation window exports the same finite consumer interface. The exclusion clauses are inherited from the public interface, handoff exactness, selector readback, and associative readback theorems. □

Theorem 126.214 (StreamName RealUp tail-selector public interface descent)**.** *Let a downstream* $\mathsf{Real}^{\uparrow}$ *or* RegSeqRat *consumer request factor through the public interface of Definition 126.213. Suppose the selected* $\mathsf{Real}^{\uparrow}$ *windows in that request are compatible over the same DyadicRatCore endpoints and are replaced by an accepted StreamName–*RegSeqRat*–*$\mathsf{Real}^{\uparrow}$ *tail-selector transport packet. Then the replacement preserves exactly the exported finite rows visible to the consumer: the* RatStreamNameFiniteWindowClassifier *row, the* UnaryHistory *rows, the* InBundle *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the visible* Cont/hsame/Pkg/RealLedgerPolicy *ledger.*

The descent exports no quotient row, completion object, Classical representative, metric-completeness data, arithmetic or order row, standard bridge row, or hidden selector. Thus any downstream request that is already expressed through the public interface reads the same finite StreamName, RegSeqRat*, DyadicRatCore, and* $\mathsf{Real}^{\uparrow}$ *ledger rows after transport, and no additional row is made available by the replacement.*

Proof. First reduce the downstream request to the public StreamName surface. Apply Definition 126.213. Since the request factors through that interface, its observable inputs are exactly the finite classifier, bundle, accepted spine, endpoint, and visible ledger rows listed there.

Next read the transport packet through the handoff theorem. Apply Definition 126.204. It identifies the StreamName finite-window classifier, bundled observations, pointwise rational-history rows, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and finite $\mathsf{Real}^{\uparrow}$ ledger carried by the packet.

Now select the denominator-window coordinates. Apply Definition 126.207. The selected $\mathsf{Real}^{\uparrow}$ windows and the accepted spine are read from the same finite handoff ledger, so compatible endpoint windows expose the same consumer-visible denominator data.

Finally apply Definition 126.211. The readback is independent of the bracketing used to assemble the accepted packet. Combining this with the public interface gives exactly the exported finite rows and only those rows; the excluded quotient, completion, representative, metric, arithmetic, order, bridge, and hidden-selector data are absent from every cited surface. □

Theorem 126.215 (StreamName tail-selector finite-window transport)**.** *Let an accepted StreamName–*RegSeqRat*–*$\mathsf{Real}^{\uparrow}$ *tail-selector transport packet expose a finite tail-selector window over the same DyadicRatCore endpoints. If the window is transported through the RegSeqRat handoff exactness surface, then it preserves the StreamName finite section and the DyadicRatCore denominator window before any RealUp public-interface descent is applied.*

The transport exports no quotient row, completion object, Classical representative, ambient metric-completeness data, arithmetic or order row, standard bridge row, or hidden selector.

Proof. First apply Definition 126.204. The handoff identifies the transported StreamName finite-window classifier, bundled observations, pointwise rational-history rows, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and finite Real ledger carried by the packet.

Next apply Definition 126.207. The denominator selector reads the transported finite window before any public-interface descent, so the DyadicRatCore denominator data remain attached to the same finite tail-selector window.

Only after these two finite-window steps apply Definition 126.214. The descent consumes the already transported finite section and denominator window; it does not create the window or replace it with a terminal RealUp seal. □

126.12 StreamName Tail-Window Coverage Packet

Definition 126.216 (StreamName tail-window coverage packet)**.** *A StreamName tail-window coverage packet for a selected finite* $\mathsf{Real}^{\uparrow}$ *consumer window is the finite ledger consisting exactly of these rows:*

1. *a displayed* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$ *row for public* $\mathsf{StreamName}^{\uparrow}$ *rational stream endpoints* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *and the selected finite* $\mathsf{ProbeBundle}$ *window* $\mathcal{P}$*;*
2. *for each selected observation* n*, the bundled* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$ *rows;*
3. *the pointwise* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *row read from the selected StreamName observation;*
4. *the accepted* $\mathsf{RegSeqRat}$ *spine selected by the pointwise rational-history rows and by the finite-window handoff;*
5. *the selected finite* $\mathsf{Real}^{\uparrow}$ *windows whose DyadicRatCore endpoint rows are read from that accepted* $\mathsf{RegSeqRat}$ *spine; and*
6. *the shared* Cont, hsame, Pkg, *and* $\mathsf{RealLedgerPolicy}$ *rows joining the StreamName endpoints, the accepted* $\mathsf{RegSeqRat}$ *spine, the DyadicRatCore endpoint rows, and the selected* $\mathsf{Real}^{\uparrow}$ *readback packets.*

The packet is only this displayed finite ledger. It contains no completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, standard bridge row, or hidden selector.

Theorem 126.217 (StreamName tail-window coverage exhaustion)**.** *Every consumer row of a StreamName tail-window coverage packet is one of the rows displayed in Definition 126.216. In particular, the consumer can read only the* $\mathsf{RatStreamNameFiniteWindowClassifier}$ *row, the bundled* $\mathsf{UnaryHistory}/\mathsf{InBundle}$ *rows, the pointwise* $\mathsf{RatHistoryClassifier}$ *rows, the accepted* $\mathsf{RegSeqRat}$ *spine, the selected DyadicRatCore endpoint rows, and the shared* $\mathsf{Cont}/\mathsf{hsame}/\mathsf{Pkg}/\mathsf{RealLedgerPolicy}$ *ledger.*

Proof. First apply Definition 126.204. It decomposes the handoff exported across the Real-and-Completion boundary into the StreamName finite-window classifier, selected unary observations, bundle membership, pointwise rational-history classifiers, the accepted $\mathsf{RegSeqRat}$ spine, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

Next apply Definition 126.205 to the selected $\mathsf{Real}^{\uparrow}$ window. This says every consumer-facing $\mathsf{Real}^{\uparrow}$ row factors through the same displayed finite public surface and through no additional selector row.

Combining the two exactness statements gives the exhaustion claim. Any row read by the consumer is already named by the coverage packet, and every row excluded by the two cited theorems remains outside the packet boundary. □

Theorem 126.218 (RegSeqRat finite-window classifier coverage)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public $\mathsf{StreamName}^{\uparrow}$ rational stream endpoints, let $\mathcal{P}$ be a finite* ProbeBundle *window, and suppose a StreamName tail-window coverage packet displays the row* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$*. For every certified observation n with* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$*, the pointwise row* $\mathsf{RatHistoryClassifier}(s(n$ *is exposed in the same packet. The accepted* RegSeqRat *spine consumes exactly these exposed pointwise rational-history classifier rows, together with their displayed unary-history and bundle-membership witnesses.*

No completion object, quotient representative, host-index equality, $\mathsf{Real}^{\uparrow}$ *seal classifier, metric-completeness row, arithmetic or order row, or standard bridge row is supplied by this coverage claim.*

Proof. Unfold Definition 126.216. The packet contains the finite-window classifier row, the bundled UnaryHistory/InBundle rows, and the pointwise RatHistoryClassifier rows read from the selected StreamName observations.

Project the component indexed by the certified observation n. The hypotheses $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$ identify n as one of the selected observations of the displayed finite window, so the corresponding pointwise rational-history classifier is precisely $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

The accepted RegSeqRat spine is then the spine named in the same packet from these pointwise rows and the finite-window handoff. Apply Definition 126.217 to rule out any additional consumer row.

Finally apply Definition 126.223. The $\mathsf{Real}^{\uparrow}$ readback can use only the finite windows selected from the accepted spine, while completion, quotient, host-index equality, seal classifier, metric-completeness, arithmetic or order, and standard bridge rows remain outside the packet boundary. □

Theorem 126.219 (StreamName tail-window dyadic endpoint determinacy)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be StreamName tail-window coverage packets over shared public endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$ and the same accepted* RegSeqRat *spine. Suppose their displayed finite-window classifier rows agree, and suppose the selected* UnaryHistory *and* InBundle *membership rows agree by the shown* hsame/Cont/Pkg *provenance. Then the two packets expose identical DyadicRatCore endpoint rows to any finite* $\mathsf{Real}^{\uparrow}$ *consumer.*

The determined finite interface is the shared RatStreamNameFiniteWindowClassifier *row, the selected* ProbeBundle/InBundle/UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows consumed by the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the common* Cont/hsame/Pkg/RealLedgerP *ledger. It contains no completion, quotient, metric-completeness, arithmetic or order, or standard-bridge row.*

Proof. Unfold Definition 126.216. The only selected rows available for comparison are the finite-window classifier, the displayed bundle and unary-history rows, the pointwise rational-history classifiers, the accepted RegSeqRat spine, the DyadicRatCore endpoint rows, and the finite ledger provenance.

Apply Definition 126.218. The shared finite-window classifier and the agreed selected membership rows force the same pointwise RatHistoryClassifier rows to feed the common accepted RegSeqRat spine in both packets.

Next apply Definition 126.217. Since the accepted spine and its pointwise StreamName inputs are the same displayed rows, the DyadicRatCore endpoint rows selected from that spine are the same finite packet rows for both consumers.

Finally use Definition 126.223. The boundary has no slot for completion, quotient, metric-completeness, arithmetic or order, or standard bridge data, so no off-packet row can separate the two exposed DyadicRatCore endpoint surfaces. □

Theorem 126.220 (StreamName tail-window hsame transport stability)**.** *Let $\mathcal{P}$ be a StreamName tail-window coverage packet over public endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$. Suppose the displayed* hsame, Cont, *and* Pkg *rows transport componentwise the* RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{P})$ *row, the selected* UnaryHistory/InBundle *rows, each pointwise* RatHistoryClassifier *row, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the* RealLedgerPolicy *ledger to a second displayed finite window. Then the transported rows form another StreamName tail-window coverage packet.*

Moreover, every finite $\mathsf{Real}^{\uparrow}$ *consumer row of the original packet transports through the same displayed packet surface: its source is still the finite-window classifier, selected unary and bundle rows, pointwise rational history rows, accepted* RegSeqRat *spine, DyadicRatCore endpoint rows, and shared* Cont/hsame/Pkg/RealLedgerPolicy *ledger. The transport supplies no completion object, quotient row, host-index equality, metric-completeness row, arithmetic or order row, or standard bridge row.*

Proof. Unfold Definition 126.216. Each row named in the packet definition has a displayed transported partner by hypothesis: the finite-window classifier, selected unary-history and bundle-membership rows, pointwise rational-history rows, accepted rational-sequence spine, endpoint readbacks, and ledger-policy rows.

Repack those transported partners in the same order. The classifier and membership rows still select a finite StreamName window; the pointwise rational-history rows still feed the accepted RegSeqRat spine; and the DyadicRatCore endpoint rows and ledger rows still give the finite $\mathsf{Real}^{\uparrow}$ readback surface.

Apply Definition 126.217 to the original and transported packets. Every consumer row is one of the rows just transported, so the consumer transport factors through exactly the same displayed packet surface.

Finally apply Definition 126.223. Completion, quotient, host-index equality, metric-completeness, arithmetic or order, and standard bridge rows have no constructor in either finite packet, so the transport cannot introduce any such row. □

Theorem 126.221 (StreamName tail-window common refinement)**.** *Given two StreamName tail-window coverage packets for finite* ProbeBundle *windows $\mathcal{P}$ and $\mathcal{Q}$ over the same public* $\mathsf{StreamName}^{\uparrow}$ *endpoints $s, t : \mathsf{BHist} \to \mathsf{BHist}$ and the same accepted* RegSeqRat *spine, there is a finite coverage packet over the bundle generated by the selected observations of $\mathcal{P}$ and $\mathcal{Q}$. Its restrictions to the two selected subbundles recover the original packets. Every readable row in the common packet remains one of the* RatStreamNameFiniteWindowClassifier *rows, the* UnaryHistory/InBundle *rows, the pointwise* RatHistoryClassifier *rows, the shared* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the shared* Cont/hsame/Pkg/RealLedgerPolicy *rows.*

Proof. Unpack the two coverage packets using Definition 126.216. The public endpoints s, t and the accepted RegSeqRat spine are shared, so the pointwise rational-history classifier rows and the endpoint readback rows are compared inside one displayed ledger.

Form the generated finite bundle by taking the selected UnaryHistory/InBundle rows from $\mathcal{P}$ and $\mathcal{Q}$ and closing under the displayed finite bundle constructor. The restriction to each selected subbundle forgets the other selected rows and therefore recovers the corresponding original packet.

Now apply Definition 126.217. The generated packet has no constructor beyond the rows named in the coverage definition, so the union operation introduces no extra consumer row.

Finally apply Definition 126.223 to the generated packet and to its two restrictions. The completion, quotient, host-index equality, metric-completeness, arithmetic or order, and standard bridge rows remain outside all three packet boundaries. □

Theorem 126.222 (StreamName tail-window refinement associativity)**.** *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be finite* ProbeBundle *windows carrying StreamName tail-window coverage packets over the same public endpoints s, t : BHist $\to$ BHist and the same accepted* RegSeqRat *spine. The common-refinement packet obtained by first refining $(\mathcal{P}, \mathcal{Q})$ and then refining the result with $\mathcal{R}$ has the same public restriction on every selected subwindow as the packet obtained by first refining $(\mathcal{Q}, \mathcal{R})$ and then refining the result with $\mathcal{P}$.*

The shared public restriction consists exactly of the RatStreamNameFiniteWindowClassifier *rows, the selected* UnaryHistory *rows, the* InBundle *membership rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the shared* Cont/hsame/Pkg/RealLedgerP *ledger.*

Proof. Apply Definition 126.221 to $\mathcal{P}$ and $\mathcal{Q}$, and then apply it again to the resulting generated bundle and $\mathcal{R}$. This constructs the left-grouped coverage packet with public endpoints s, t and with the fixed accepted RegSeqRat spine.

Apply the same theorem to $\mathcal{Q}$ and $\mathcal{R}$, and then to $\mathcal{P}$ and the resulting generated bundle. This constructs the right-grouped coverage packet with the same endpoints and the same accepted spine.

The finite bundle generated by selected observations is associative: selecting the rows of $\mathcal{P}$ and $\mathcal{Q}$ and then adjoining the rows of $\mathcal{R}$ gives the same selected observation family as selecting the rows of $\mathcal{Q}$ and $\mathcal{R}$ and then adjoining the rows of $\mathcal{P}$. Restriction to any chosen subwindow therefore forgets the same complement in both grouped packets.

Now apply Definition 126.217 to both grouped packets. Every readable row in either packet is one of the displayed finite-window classifier, unary-history, bundle-membership, rational-history, RegSeqRat, DyadicRatCore, continuation, history, package, or ledger-policy rows, and those rows are indexed by the same selected observation family.

Finally apply Definition 126.223 to both grouped packets and to their selected restrictions. Completion, quotient, host-index equality, metric-completeness, arithmetic or order, standard bridge, and unexposed selector data have no constructor in either public surface. Hence the two grouped common-refinement packets restrict to the same public rows on every selected subwindow. □

Theorem 126.223 (StreamName tail-window RealUp boundary)**.** *A finite* $\mathsf{Real}^{\uparrow}$ *consumer of a StreamName tail-window coverage packet reads only the displayed StreamName,* RegSeqRat*, DyadicRatCore,* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows. It reads no completion row, quotient row, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameTailWindowRealUpBoundary`

Proof. Read the consumer through Definition 126.217. The exhaustion surface lists every available consumer row and identifies it with the finite packet of Definition 126.216.

Now inspect the packet definition. Its StreamName side is the finite-window classifier together with selected unary-history and bundle-membership rows. Its rational-sequence side is the accepted RegSeqRat spine selected by the pointwise rational-history classifiers. Its $\mathsf{Real}^{\uparrow}$ side is only the DyadicRatCore endpoint readback and the shared Cont/hsame/Pkg/RealLedgerPolicy ledger.

The excluded objects have no constructor in this finite ledger. Hence a finite $\mathsf{Real}^{\uparrow}$ consumer cannot read completion, quotient, host-index equality, metric-completeness, arithmetic or order, or standard bridge data from the packet. □

Theorem 126.224 (StreamName tail-window terminal consumer factorization)**.** *Every finite* $\mathsf{Real}^{\uparrow}$ *consumer row whose source reads a StreamName tail-window coverage packet factors through the displayed rows of Definition 126.216: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, the selected* UnaryHistory *observations and* InBundle *membership rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore observation rows, and the shared* Cont/hsame/Pkg/RealLedgerPoli *ledger. If two such consumers are given the same displayed packet, then they have the same finite read surface. They do not read any completion object, quotient row, host sequence equality, or standard real-number structure.*

Proof. Apply Definition 126.217. It enumerates all readable rows of the packet and identifies each one with a row already displayed in Definition 126.216.

Next apply Definition 126.223. The finite $\mathsf{Real}^{\uparrow}$ consumer can therefore see only the StreamName classifier, selected unary observation and bundle rows, rational-history classifier rows, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and shared ledger rows named in the statement.

For two consumers using the same displayed packet, apply Definition 126.221 to place their selected finite windows in one common packet. Then apply Definition 126.222 to see that changing the grouping of those finite windows leaves the restriction to each selected subwindow unchanged.

Thus the factorization is terminal among finite packet readers: every observable row is already in the displayed packet, and any grouped finite read of the same packet restricts back to the same StreamName, RegSeqRat, DyadicRatCore, continuation, history, package, and ledger-policy surface. The objects excluded by the boundary theorem have no row in the factorization. □

Theorem 126.225 (StreamName tail-window terminal source uniqueness)**.** *Let two finite* $\mathsf{Real}^{\uparrow}$ *consumers read StreamName tail-window coverage packets over the same public* $\mathsf{StreamName}^{\uparrow}$ *endpoints* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *and the same accepted* RegSeqRat *spine. If a common-refinement packet restricts to both packets, then both consumers factor through the same displayed StreamName,* RegSeqRat*, DyadicRatCore,* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *source surface. No alternative* $\mathsf{Real}^{\uparrow}$ *source row, quotient representative, completion row, or host-index equality can be read from either finite consumer.*

Proof. First apply Definition 126.221. The two selected finite windows sit in one coverage packet over the shared public StreamName endpoints and the shared accepted RegSeqRat spine, and the restrictions to the selected subwindows recover the two original packets.

Next apply Definition 126.222. Any regrouping of the common finite window has the same public restriction on each selected subwindow, so the two consumers see the same StreamName finite-window classifier rows, selected unary-history and bundle-membership rows, pointwise rational-

history classifier rows, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and shared ledger rows.

Now apply Definition 126.224. Every finite $\mathsf{Real}^{\uparrow}$ consumer row factors through exactly that displayed packet surface, and two consumers reading the same restricted packet have the same finite read surface.

Finally apply Definition 126.223. The boundary removes completion, quotient, host-index equality, metric-completeness, arithmetic or order, standard bridge, and hidden selector data from the readable packet. Hence no other $\mathsf{Real}^{\uparrow}$ source row, quotient representative, completion row, or host-index equality can be read, and the displayed StreamName/RegSeqRat/DyadicRatCore source surface is the unique finite source surface for both consumers. □

Theorem 126.226 (StreamName tail-window global classifier boundary)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public $\mathsf{StreamName}^{\uparrow}$ endpoints. Suppose that every certified unary observation singleton window over the endpoints carries the row $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{Bcons}(n, \mathsf{Bnil}))$. If a finite $\mathsf{Real}^{\uparrow}$ consumer reads a StreamName tail-window coverage packet generated from finitely many selected observations over an accepted RegSeqRat spine, then the consumer factors through the global $\mathsf{RatStreamNameClassifier}(s, t)$ together with exactly the displayed StreamName, RegSeqRat, DyadicRatCore, Cont, hsame, Pkg, and RealLedgerPolicy packet surface.*

The factorization exposes no completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.

Proof. First apply Definition 126.23. The singleton-window hypotheses exhaust the certified finite observations and therefore determine the global $\mathsf{RatStreamNameClassifier}(s, t)$ for the public StreamName endpoints.

Next apply Definition 126.217. The finite packet read by the $\mathsf{Real}^{\uparrow}$ consumer has only the enumerated selected unary-history rows, bundle-membership rows, pointwise rational-history classifier rows, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and shared Cont/hsame/Pkg/RealLedgerPolicy ledger.

Now apply Definition 126.224. Every finite consumer row factors through that displayed packet surface, so the global classifier supplied by the singleton-window exhaustion is the only StreamName classifier row available to the consumer.

Finally apply Definition 126.223. The boundary excludes completion, quotient representatives, host-index equality, metric-completeness, arithmetic and order operations, and standard bridge data. Hence the finite $\mathsf{Real}^{\uparrow}$ consumer factors through the global StreamName classifier and the displayed packet rows, with no off-boundary row exposed. □

Theorem 126.227 (StreamName tail-window global classifier transport stability)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public $\mathsf{StreamName}^{\uparrow}$ endpoints satisfying the singleton-window hypotheses of Definition 126.226. Let $\mathcal{P}$ be a StreamName tail-window coverage packet over s, t, generated from finitely many selected observations over an accepted RegSeqRat spine. Suppose displayed hsame, Cont, and Pkg rows transport s, t and every row of $\mathcal{P}$ to endpoints s', t' and to a second finite tail-window coverage packet $\mathcal{P}'$ over the transported endpoints.*

Then every finite $\mathsf{Real}^{\uparrow}$ consumer of $\mathcal{P}'$ factors through the transported global $\mathsf{RatStreamNameClassifier}(s', t')$ and exactly the transported StreamName, RegSeqRat, DyadicRatCore, Cont, hsame, Pkg, and RealLedgerPolicy packet surface. The transported consumer reads no completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.

Proof. First apply Definition 126.220. The displayed hsame, Cont, and Pkg rows carry the finite-window classifier, selected unary-history and bundle-membership rows, pointwise rational-history classifiers, accepted RegSeqRat spine, DyadicRatCore endpoint rows, and ledger-policy rows into the transported packet $\mathcal{P}'$.

The singleton-window hypotheses transport with those same displayed rows. Apply Definition 126.226 to $\mathcal{P}'$ over s', t'. This supplies the transported global $\mathsf{RatStreamNameClassifier}(s', t')$ as the StreamName classifier row available to any finite $\mathsf{Real}^{\uparrow}$ consumer of the transported packet.

Now apply Definition 126.224. The consumer factors through the transported packet surface: the StreamName classifier rows, selected window rows, pointwise rational-history rows, accepted rational-sequence spine, DyadicRatCore endpoint rows, continuation rows, history comparisons, package provenance, and ledger policy.

Finally apply Definition 126.223. Since the transported finite packet has no constructor for completion, quotient, host-index equality, metric-completeness, arithmetic or order, or standard bridge rows, none of those rows can be read by the transported consumer. □

Theorem 126.228 (StreamName tail-window selected pointwise transport)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public $\mathsf{StreamName}^{\uparrow}$ endpoints, and let $\mathcal{P}$ be a StreamName tail-window coverage packet over a finite* ProbeBundle*. Suppose a selected unary observation row $\mathsf{InBundle}(n, \mathcal{P})$ carries the pointwise classifier $\mathsf{RatHistoryClassifier}(s(n), t(n))$. If the displayed* hsame*,* Cont*, and* Pkg *rows transport that selected observation to a selected row in a transported finite packet $\mathcal{P}'$, then the transported pointwise classifier row remains one of the visible rows of $\mathcal{P}'$.*

*Every finite $\mathsf{Real}^{\uparrow}$ consumer using that selected observation factors through the transported StreamName/*RegSeqRat*/DyadicRatCore packet surface. No completion, quotient, host-index equality, metric-completeness, arithmetic or order row, or standard-bridge row is exposed.*

Proof. First unfold the StreamName tail-window coverage packet of Definition 126.216. The selected $\mathsf{InBundle}(n, \mathcal{P})$ row is one of the finite packet coordinates, together with its unary-history row and pointwise $\mathsf{RatHistoryClassifier}(s(n), t(n))$ row.

Apply Definition 126.220. The displayed hsame, Cont, and Pkg rows repack the selected observation, its bundle-membership row, and its pointwise classifier into the transported packet $\mathcal{P}'$ over the transported endpoints.

Next apply Definition 126.217 to $\mathcal{P}'$. The exhaustion statement enumerates the transported selected unary-history rows, bundle-membership rows, and pointwise rational-history classifier rows, so the transported classifier attached to n is a visible row of $\mathcal{P}'$.

Finally apply Definition 126.224 and Definition 126.223. Any finite $\mathsf{Real}^{\uparrow}$ consumer using the selected row factors through the transported StreamName, RegSeqRat, DyadicRatCore, Cont, hsame, Pkg, and ledger surface, and the boundary has no coordinate for completion, quotient, host-index equality, metric-completeness, arithmetic or order, or a standard bridge. □

Theorem 126.229 (StreamName selected transport coverage)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public $\mathsf{StreamName}^{\uparrow}$ endpoints, let $\mathcal{P}$ be a StreamName tail-window coverage packet over a finite* ProbeBundle*, and let $\mathcal{P}'$ be the transported packet obtained from displayed* hsame*,* Cont*, and* Pkg *rows. Suppose the selected observations of $\mathcal{P}$ are transported to selected observations of $\mathcal{P}'$ by those displayed rows.*

Then every transported selected pointwise RatHistoryClassifier *row, accepted* RegSeqRat *spine row, DyadicRatCore endpoint row, and finite $\mathsf{Real}^{\uparrow}$ consumer row factors through the transported tail-window coverage packet $\mathcal{P}'$. The factorization uses only the transported StreamName,* RegSeqRat*, DyadicRatCore,* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *rows. It exposes no completion object,*

quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.

Proof. Apply Definition 126.228 to each selected observation. The transported selected rows remain visible coordinates of $\mathcal{P}'$, together with their pointwise RatHistoryClassifier rows.

Next apply Definition 126.227. The transported packet supplies the global RatStreamNameClassifier row and preserves the accepted RegSeqRat spine and DyadicRatCore endpoint rows through the displayed hsame, Cont, and Pkg routes.

Finally apply Definition 126.224 and Definition 126.223. Every finite $\mathsf{Real}^{\uparrow}$ consumer of the transported packet factors through the listed packet rows, and the boundary theorem rules out completion objects, quotient representatives, host-index equality, metric-completeness rows, arithmetic or order operations, and standard bridge rows. □

Theorem 126.230 (StreamName tail-window RegSeqRat coverage lock)**.** *Let s, t : BHist $\to$ BHist be public $\mathsf{StreamName}^{\uparrow}$ endpoints, and let $\mathcal{P}$ and $\mathcal{Q}$ be StreamName tail-window coverage packets over finite windows. Suppose their displayed* RatStreamNameFiniteWindowClassifier *rows, selected* UnaryHistory *rows, selected* InBundle *rows, pointwise* RatHistoryClassifier *rows, accepted* RegSeqRat *spine, DyadicRatCore endpoint rows, and* Cont/hsame/Pkg/RealLedgerPolicy *provenance rows agree through the transported packet surface.*

Then every finite $\mathsf{Real}^{\uparrow}$ consumer of $\mathcal{P}$ and every finite $\mathsf{Real}^{\uparrow}$ consumer of $\mathcal{Q}$ read the same DyadicRatCore endpoint rows and the same finite ledger rows from that shared RegSeqRat *spine. This lock exposes no completion data, quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row.*

Proof. Unfold the StreamName tail-window coverage packet of Definition 126.216 for both $\mathcal{P}$ and $\mathcal{Q}$. Apply Definition 126.217 to each packet. The only finite source rows available on either side are the displayed finite-window classifier rows, selected unary-history and bundle-membership rows, pointwise rational-history classifier rows, accepted RegSeqRat spine, DyadicRatCore endpoints, and ledger provenance.

Next apply Definition 126.227 and Definition 126.229. The hypothesized transported packet surface identifies the selected rows of $\mathcal{P}$ with the selected rows of $\mathcal{Q}$ and carries their accepted rational-sequence spine and DyadicRatCore endpoint rows through the same Cont, hsame, Pkg, and RealLedgerPolicy routes.

Now apply Definition 126.225. Since the two finite packets restrict to the same transported selected source surface, their finite $\mathsf{Real}^{\uparrow}$ consumers have the same displayed StreamName/RegSeqRat/DyadicRatCore source surface and the same finite continuation ledger.

Finally apply Definition 126.223. The boundary theorem removes completion data, quotient representatives, host-index equality, metric-completeness rows, arithmetic or order operations, and standard bridge rows from every finite consumer. Thus the only endpoint and ledger rows read by either consumer are the common DyadicRatCore endpoint rows and the common ledger rows carried by the shared RegSeqRat spine. □

Theorem 126.231 (StreamName tail-window prefix-suffix recomposition)**.** *Let s, t : BHist $\to$ BHist be public $\mathsf{StreamName}^{\uparrow}$ endpoints, and let $\mathcal{P}$ be a StreamName tail-window coverage packet over a finite* ProbeBundle*. Suppose the selected observations of $\mathcal{P}$ split into displayed prefix and suffix subbundles with a shared boundary row. Then the restrictions of $\mathcal{P}$ to the prefix and suffix subbundles are StreamName tail-window coverage packets over the same public endpoints and the same accepted* RegSeqRat *spine.*

The common-refinement recomposition of these two restricted packets recovers $\mathcal{P}$ on every selected observation. Every finite $\mathsf{Real}^{\uparrow}$ *consumer row of $\mathcal{P}$ factors through either the prefix packet, the suffix packet, or the boundary-overlap rows. No completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order operation, or standard bridge row is read by the recomposed packet.*

Proof. Unfold the StreamName tail-window coverage packet of Definition 126.216. Restricting to the displayed prefix subbundle keeps the public StreamName endpoints, the selected UnaryHistory/InBundle rows in the prefix window, the pointwise RatHistoryClassifier rows, the accepted RegSeqRat spine, the DyadicRatCore endpoint rows read there, and the inherited Cont/hsame/Pkg/RealLedgerPolicy provenance. The same restriction argument gives the suffix packet.

Apply Definition 126.221 to the prefix and suffix packets. The generated packet is finite, its restrictions recover the two displayed subbundles, and the shared boundary row is carried as one row in the common packet rather than as two independent sources.

Next apply Definition 126.222. It identifies the regrouping of the prefix packet, the boundary overlap, and the suffix packet with the original finite window, so every selected observation of $\mathcal{P}$ is recovered by the recomposed packet.

Now apply Definition 126.225 and Definition 126.230. The finite consumer source surface is therefore the displayed StreamName/RegSeqRat/DyadicRatCore source surface of either restricted packet, together with the shared boundary rows when the consumer reads across the split. The RegSeqRat and DyadicRatCore rows read by the prefix, suffix, and recomposed packet are the same rows of the original finite ledger.

Finally apply Definition 126.223. The finite $\mathsf{Real}^{\uparrow}$ consumer has no row for completion, quotient, host-index equality, metric-completeness, arithmetic or order, standard bridge, or hidden selector data, so the prefix/suffix recomposition cannot introduce any such object. □

Theorem 126.232 (StreamName tail-window cover-chain source confluence)**.** *Let*

$$\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_n$$

be a finite chain of StreamName tail-window coverage packets over the same public $\mathsf{StreamName}^{\uparrow}$ *endpoints and the same accepted* RegSeqRat *spine. Assume each adjacent pair $\mathcal{P}_i, \mathcal{P}_{i+1}$ carries certified overlap rows for the displayed finite windows, and assume the chain is closed under the common-refinement operation of Definition 126.221. Then every packet in the chain induces the same consumer-facing source surface for finite* $\mathsf{Real}^{\uparrow}$ *reads.*

Concretely, the common surface consists exactly of the finite-window classifier rows, the selected ProbeBundle/InBundle/UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the* Cont/hsame/Pkg/RealLedgerPolicy *provenance rows transported along the chain. The confluence statement exports no completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order row, standard bridge row, or hidden selector.*

Proof. Induct on the finite chain length. For the base chain, apply Definition 126.224 and Definition 126.223. A single terminal packet therefore exposes only the finite StreamName/RegSeqRat/DyadicRatCore source surface and its displayed continuation ledger.

For the induction step, compare the already-confluent initial subchain with the last packet. The certified overlap rows allow Definition 126.221 to form the common finite packet for the terminal member of the subchain and the last packet.

Use Definition 126.222 to identify this refinement with the refinement obtained by first recomposing the earlier chain and then attaching the last overlap. Apply Definition 126.225 to the

two terminal packets inside that common refinement, and apply Definition 126.230 to keep the accepted rational-sequence spine and DyadicRatCore endpoint rows fixed.

Finally apply Definition 126.223 to the exported finite consumer. The boundary theorem removes completion, quotient, host-index equality, metric-completeness, arithmetic or order, standard bridge, and hidden-selector data from the readout, so the confluence is a statement only about the displayed finite source rows. □

Theorem 126.233 (StreamName tail-window terminal coverage)**.** *Any terminal* $\mathsf{Real}^{\uparrow}$ *read induced by a finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window cover chain is covered by exactly the common finite source surface of Definition 126.232. Thus the terminal read exposes the finite-window classifier rows, selected* ProbeBundle/InBundle/UnaryHistory *rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, DyadicRatCore endpoint rows, and transported* Cont/hsame/Pkg/RealLedgerPolicy *provenance rows.*

It exposes no completion object, quotient representative, host-index equality, metric-completeness row, arithmetic or order row, standard bridge row, or hidden selector.

Proof. Apply Definition 126.232 to the finite cover chain. The induced terminal source is the common finite surface carried by every packet in the chain.

Next apply Definition 126.224. It routes the terminal RealUp read through the displayed StreamName, RegSeqRat, and DyadicRatCore source rows rather than through an external completion carrier.

Finally use Definition 126.223. The boundary removes quotient representatives, host-index equality, metric-completeness rows, arithmetic and order rows, standard bridge rows, and hidden selectors from the terminal read. Therefore the read is covered exactly by the common finite source surface from the confluence theorem. □

126.13 StreamName Common-Refinement Source Determinacy

Theorem 126.234 (StreamName common-refinement source determinacy)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over certified observation maps* s, t*, and let* $\mathcal{R}$ *be a certified common-refinement package whose* ProbeBundle *window restricts to the selected windows of* $\mathcal{W}_L$ *and* $\mathcal{W}_R$*. Then every* RegSeqRat *handoff and every finite-window* $\mathsf{Real}^{\uparrow}$ *handoff read from* $\mathcal{R}$ *has the same StreamName source tuple, namely the restricted* RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{R})$ *row, the displayed* ProbeBundle *row, the* InBundle *membership rows, the selected* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, and the displayed* hsame/Cont/Pkg *rows carried by the refinement ledger.*

This source tuple is independent of whether the handoff is reached through $\mathcal{W}_L$ *or through* $\mathcal{W}_R$*. It contains no completion data, no quotient row, no host index equality, no* RegSeqRat *regularity row, and no* $\mathsf{Real}^{\uparrow}$ *seal classifier.*

Proof. First read the common refinement at the public StreamName boundary. Apply Definition 126.308; its restriction clauses identify the same finite-window classifier over the common refinement with the two component windows. Then apply Definition 126.309 to each selected observation in the refinement. The selected endpoint rows are exactly the pointwise RatHistoryClassifier rows and the finite endpoint readback rows displayed by the common refinement.

Next pass from the public refinement to the tail-window packet. Apply Definition 126.221. The packet over the generated common window restricts to the two selected subbundles and keeps the same ProbeBundle, InBundle, UnaryHistory, pointwise classifier, hsame, Cont, and Pkg ledger rows.

Now read the downstream handoff. Apply Definition 126.170 to the selected pointwise rows and then apply Definition 126.82. These two theorems separate the StreamName source rows from the $\mathsf{RegSeqRat}$ regularity obligations and from the finite-window $\mathsf{Real}^{\uparrow}$ seal obligation.

The same source tuple is therefore obtained along the left route and along the right route, because both routes are restrictions of the one displayed common-refinement packet. Completion objects, quotient rows, host index equalities, $\mathsf{RegSeqRat}$ regularity rows, and $\mathsf{Real}^{\uparrow}$ seal classifiers have no constructor in that source packet, so they cannot be part of the determined source surface. □

Theorem 126.235 (StreamName directed common-refinement source associativity)**.** *Let $\mathcal{W}_1$, $\mathcal{W}_2$, $\mathcal{W}_3$, and $\mathcal{W}_4$ be public finite-window $\mathsf{StreamName}^{\uparrow}$ packages over the same certified endpoint maps s, t. Let $\mathcal{R}_{(12)3,4}$ and $\mathcal{R}_{1(23),4}$ be two directed certified common-refinement trees whose visible leaves are the same four selected windows, with the first tree grouping $((\mathcal{W}_1, \mathcal{W}_2), \mathcal{W}_3)$ before comparison with $\mathcal{W}_4$, and the second tree grouping $(\mathcal{W}_1, (\mathcal{W}_2, \mathcal{W}_3))$ before comparison with $\mathcal{W}_4$.*

Assume each internal common-refinement package restricts to the selected windows named by its children and carries the displayed certified refinement ledger. Then the two directed trees expose the same StreamName source tuple: the row $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{R})$ read from the common visible refinement, the displayed $\mathsf{ProbeBundle}$ row, the $\mathsf{InBundle}$ membership rows, the selected $\mathsf{UnaryHistory}$ rows, the pointwise $\mathsf{RatHistoryClassifier}$ rows, and the displayed $\mathsf{hsame/Cont/Pkg}$ ledger rows.

Regrouping the directed common-refinement tree introduces no $\mathsf{RegSeqRat}$ regularity row, no $\mathsf{Real}^{\uparrow}$ seal classifier, no quotient row, no completion datum, no metric-completeness datum, and no host-index equality. Those rows remain outside the StreamName source tuple consumed by downstream $\mathsf{RegSeqRat}$ and terminal finite-window $\mathsf{Real}^{\uparrow}$ readback.

Proof. First compare the two triples before the terminal comparison with $\mathcal{W}_4$. Apply Definition 126.239 to the triple $(\mathcal{W}_1, \mathcal{W}_2, \mathcal{W}_3)$ as grouped through $((\mathcal{W}_1, \mathcal{W}_2), \mathcal{W}_3)$, and apply the same theorem to the triple as grouped through $(\mathcal{W}_1, (\mathcal{W}_2, \mathcal{W}_3))$. Both applications read the same finite-window classifier, bundle, membership, selected history, pointwise classifier, and $\mathsf{hsame/Cont/Pkg}$ ledger surface from the displayed triple refinement.

Next compare the overlap carried by $\mathcal{W}_1$ and $\mathcal{W}_2$. Use Definition 126.236. The overlapping pair has one displayed StreamName source tuple, so the intermediate packet seen by either triple grouping has the same classifier, bundle, membership, selected-history, pointwise-classifier, and provenance rows.

Finally compose the displayed ledger rows by transitivity of the $\mathsf{hsame/Cont/Pkg}$ provenance. The terminal comparison with $\mathcal{W}_4$ therefore receives the same source tuple from either regrouping. Since Definition 126.236 and Definition 126.239 separate that source tuple from $\mathsf{RegSeqRat}$ regularity, $\mathsf{Real}^{\uparrow}$ sealing, quotient, completion, metric-completeness, and host-index rows, the regrouping cannot introduce any such row. □

Theorem 126.236 (StreamName common-refinement source confluence)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be public finite-window $\mathsf{StreamName}^{\uparrow}$ packages over the same certified endpoint maps s, t, and let $\mathcal{R}$ be a certified common-refinement package whose generated window restricts to both selected windows. Consider the two finite-window routes*

$$\mathcal{R} \longrightarrow \mathcal{W}_L \longrightarrow \mathsf{RegSeqRat} \quad \textit{and} \quad \mathcal{R} \longrightarrow \mathcal{W}_R \longrightarrow \mathsf{RegSeqRat},$$

followed by the finite-window $\mathsf{Real}^{\uparrow}$ handoff that reads the induced regular rational sequence rows. The two routes expose the same downstream source tuple: the row $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{R})$,

the displayed ProbeBundle*, the* InBundle *membership rows, the selected* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, and the displayed* hsame/Cont/Pkg *rows carried by the refinement ledger.*

Consequently the induced finite-window $\mathsf{Real}^{\uparrow}$ *handoffs read the same* DyadicRatCore *endpoint rows. They do not add completion data, quotient data, metric-completeness data, host-index equality, or a seal classifier to the source surface determined by the common refinement.*

Proof. First apply Definition 126.234. It identifies the StreamName source tuple seen from the left selected window with the tuple seen from the right selected window, because both are restrictions of the one certified common-refinement packet $\mathcal{R}$.

Next apply Definition 126.221. The generated tail-window packet over $\mathcal{R}$ has the same displayed ProbeBundle row, the same InBundle membership rows, the same selected UnaryHistory rows, and the same pointwise rational history classifier rows on both routes.

Now apply Definition 126.170. The RegSeqRat boundary consumes exactly those pointwise rational rows and the displayed hsame/Cont/Pkg ledger rows; it does not introduce an additional route-dependent source coordinate.

Finally apply Definition 126.82. The finite-window $\mathsf{Real}^{\uparrow}$ handoff reads the induced DyadicRatCore endpoint rows from the already identified regular rational sequence surface, while the seal obligation remains separate from the StreamName source tuple. Thus the left route and the right route compose to the same downstream RegSeqRat/$\mathsf{Real}^{\uparrow}$ source surface, with no completion, quotient, metric-completeness, host-index, or seal-classifier row added to that surface. □

Theorem 126.237 (StreamName common-refinement RealUp seal confluence)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same certified endpoint maps* s, t*, and let* $\mathcal{R}$ *be a certified common-refinement package whose generated window restricts to both selected windows. Let* $\mathcal{E}_L$ *and* $\mathcal{E}_R$ *be finite-window* $\mathsf{Real}^{\uparrow}$ *seal packages induced from* $\mathcal{R}$ *through the left and right* RegSeqRat *routes*

$$\mathcal{R} \longrightarrow \mathcal{W}_L \longrightarrow \mathsf{RegSeqRat} \quad \textit{and} \quad \mathcal{R} \longrightarrow \mathcal{W}_R \longrightarrow \mathsf{RegSeqRat}.$$

Then $\mathcal{E}_L$ *and* $\mathcal{E}_R$ *expose the same visible seal source. That source is the restricted* RatStreamNameFiniteWindowClassi *row, the displayed* ProbeBundle *row, the* InBundle *membership rows, the selected* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the induced* DyadicRatCore *endpoint rows, and the displayed* hsame/Cont/Pkg *ledger rows carried by the common refinement.*

No completion row, quotient row, host-index equality, metric-completeness row, or off-ledger $\mathsf{Real}^{\uparrow}$ *classifier is part of this visible finite-window seal source.*

Proof. First apply Definition 126.236. The certified common-refinement packet gives one StreamName source tuple for the left and right selected windows, including the restricted finite-window classifier, bundle membership, selected unary histories, pointwise rational classifiers, and displayed hsame/Cont/Pkg provenance.

Next read both induced seal packages through Definition 126.149. The two RegSeqRat routes consume the same StreamName tuple and therefore present the same accepted finite-window spine, DyadicRatCore endpoint rows, transport ledger, and provenance to the $\mathsf{Real}^{\uparrow}$ seal boundary.

Finally use Definition 126.150 for each induced package. Minimality says that every visible source row of either seal package is one of the displayed finite-window StreamName, RegSeqRat, DyadicRatCore, bundle, history, continuation, transport, or provenance rows. Since the common-refinement confluence already identifies those rows, no completion, quotient, host-index, metric-completeness, or off-ledger $\mathsf{Real}^{\uparrow}$ classifier row can distinguish the two seal sources. □

Theorem 126.238 (StreamName common-refinement successor projection)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same certified endpoint maps s, t, and let $\mathcal{R}$ be a certified common-refinement package whose generated* ProbeBundle *restricts to both selected windows. Let n :* BHist *be a certified unary observation such that* $\mathsf{BHist.e1}(n)$ *is an* InBundle *member of the refinement bundle of $\mathcal{R}$.*

Project $\mathcal{R}$ to the singleton successor window

$$\mathsf{ProbeBundle.Bcons}(\mathsf{BHist.e1}(n), \mathsf{ProbeBundle.Bnil}).$$

Reading this singleton through either selected window determines the same RatStreamNameFiniteWindowClassifier *row, the same pointwise* RatHistoryClassifier *row at* $\mathsf{BHist.e1}(n)$*, and the same downstream source tuple consumed by the* RegSeqRat *handoff and by the finite-window* $\mathsf{Real}^{\uparrow}$ *handoff.*

The determined source tuple is the successor singleton classifier, the singleton ProbeBundle *membership row, the selected* UnaryHistory *row for n, the pointwise rational history classifier, and the displayed* hsame/Cont/Pkg *ledger rows inherited from $\mathcal{R}$. It contains no host index equality, quotient representative, completion data, metric row,* RegSeqRat *regularity row, or* $\mathsf{Real}^{\uparrow}$ *completeness or seal classifier.*

Proof. First use Definition 126.236. The left route through $\mathcal{W}_L$ and the right route through $\mathcal{W}_R$ expose one common StreamName source tuple over $\mathcal{R}$, so the $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{R})$ row and the pointwise RatHistoryClassifier rows are route independent before projection.

Next apply Definition 126.299 to the InBundle membership row for $\mathsf{BHist.e1}(n)$. The theorem restricts the common finite-window classifier to the singleton successor bundle and keeps exactly the pointwise rational row indexed by that successor observation.

Finally read the downstream boundary through Definition 126.170 and Definition 126.82. These rows consume the same successor pointwise classifier and the same displayed hsame/Cont/Pkg provenance rows from either selected window, while the RegSeqRat regularity obligation and the finite-window $\mathsf{Real}^{\uparrow}$ seal obligation remain outside the StreamName source packet. □

Theorem 126.239 (StreamName common-refinement source triple confluence)**.** *Let $\mathcal{W}_1$, $\mathcal{W}_2$, and $\mathcal{W}_3$ be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same certified endpoint maps s, t, and let $\mathcal{R}$ be a certified common-refinement package whose generated window restricts to all three selected windows. For every pair (i, j), the route*

$$\mathcal{R} \longrightarrow \mathcal{W}_i \longrightarrow \mathsf{RegSeqRat} \longrightarrow \mathsf{Real}^{\uparrow}$$

and the route

$$\mathcal{R} \longrightarrow \mathcal{W}_j \longrightarrow \mathsf{RegSeqRat} \longrightarrow \mathsf{Real}^{\uparrow}$$

read the same source tuple from the common refinement. That tuple is the restricted RatStreamNameFiniteWindowCl *row, the displayed* ProbeBundle *row, the* InBundle *membership rows, the selected* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, and the displayed* hsame/Cont/Pkg *ledger rows carried by $\mathcal{R}$.*

Thus the three finite-window handoffs are confluent on the source surface used by RegSeqRat *and by finite-window* $\mathsf{Real}^{\uparrow}$ *readback. No* RegSeqRat *regularity row, seal classifier, completion data, quotient row, metric row, or host index equality is introduced by the triple comparison.*

Proof. Apply Definition 126.234 to the pairs $(\mathcal{W}_1, \mathcal{W}_2)$, $(\mathcal{W}_2, \mathcal{W}_3)$, and $(\mathcal{W}_1, \mathcal{W}_3)$. Each application reads the two selected windows as restrictions of the same certified common-refinement packet $\mathcal{R}$.

For each pair, use Definition 126.236. It identifies the downstream $\mathsf{RegSeqRat}/\mathsf{Real}^{\uparrow}$ source surface with the already displayed StreamName tuple, rather than with any route-specific regularity, completion, quotient, metric, host-index, or seal row.

The displayed common-refinement ledger is the same ledger in all three pairwise comparisons. Transitivity of the ledger equalities for the restricted finite-window classifier, ProbeBundle, InBundle, UnaryHistory, pointwise RatHistoryClassifier, and hsame/Cont/Pkg rows therefore gives one source tuple for all three routes. □

126.14 StreamName Common-Refinement Public Exactness

Theorem 126.240 (StreamName common-refinement public exactness)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same certified endpoint maps s, t, and let $\mathcal{R}$ be a certified common-refinement package whose generated window restricts to both selected windows. Then every downstream* RegSeqRat *handoff and every finite-window* $\mathsf{Real}^{\uparrow}$ *handoff read from $\mathcal{R}$ factors through exactly the common-refinement source tuple:*

$$(\mathsf{RatStreamNameFiniteWindowClassifier}(s,t,\mathcal{R}), \mathsf{ProbeBundle}_{\mathcal{R}}, \mathsf{InBundle}_{\mathcal{R}}, \mathsf{UnaryHistory}_{\mathcal{R}}, \mathsf{RatHistoryClassifier}_{\mathcal{R}}, \mathsf{hsar}$$

The public handoff cannot add completion data, quotient rows, host-index equality, metric-completeness rows, or off-ledger $\mathsf{Real}^{\uparrow}$ *classifier rows.*

Proof. First apply Definition 126.234. It identifies the source tuple read from the left selected window with the source tuple read from the right selected window, because both are restrictions of the same certified common-refinement package $\mathcal{R}$.

Next apply Definition 126.236. The downstream RegSeqRat route and the finite-window $\mathsf{Real}^{\uparrow}$ route receive the same restricted finite-window classifier, bundle row, membership rows, selected histories, pointwise rational classifiers, and displayed hsame/Cont/Pkg ledger rows.

Now use Definition 126.237. The induced $\mathsf{Real}^{\uparrow}$ seal packages have the same visible seal source, so the seal boundary cannot distinguish the two routes by adding an off-ledger classifier row.

Finally apply the RegSeqRat and RealUp handoff separation statements in the StreamName child files, in particular Definition 126.170 and Definition 126.82. They keep regularity, completion, quotient, metric-completeness, host-index, and seal-classifier data outside the common-refinement source tuple. □

Theorem 126.241 (StreamName common-refinement associativity)**.** *Let $\mathcal{W}_0, \mathcal{W}_1, \mathcal{W}_2$ be three compatible public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same certified endpoint maps. If successive common-refinement packages are displayed for both parenthesizations*

$$(\mathcal{W}_0 \cap \mathcal{W}_1) \cap \mathcal{W}_2 \quad \textit{and} \quad \mathcal{W}_0 \cap (\mathcal{W}_1 \cap \mathcal{W}_2),$$

then the two resulting packages expose the same scheduled observation rows, hsame *transports,* Cont *replay rows,* Pkg *provenance, and local* NameCert *row. The common-refinement read exports no completion object, quotient stream equality, host-index equality, metric-completeness row, or off-ledger* $\mathsf{Real}^{\uparrow}$ *classifier row.*

Proof. Project each parenthesized package to its finite-window carrier. Compatibility says that every scheduled observation row is a restriction of the same three displayed windows, so either parenthesization exposes the same source rows.

Transport the common rows componentwise through the displayed hsame witnesses and replay them through the same Cont route. The Pkg provenance and local NameCert row are then read from the same finite source tuple.

Finally use common-refinement public exactness. It keeps every downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff inside the common-refinement source tuple, so neither parenthesization can add completion, quotient, host-index, metric-completeness, or off-ledger classifier data. □

126.15 StreamName Synchronized Window Union

Theorem 126.242 (StreamName synchronized-window carrier union)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be displayed finite* $\mathsf{StreamName}^{\uparrow}$ *windows over certified source maps s, t, and let $\mathcal{R}$ be a certified common-refinement package whose generated* ProbeBundle *restricts to the selected bundles of both windows. If the selected rows of $\mathcal{W}_L$ and $\mathcal{W}_R$ agree through the displayed common-refinement* hsame *and* Cont *ledger, then the two windows repack into one synchronized* $\mathsf{StreamName}^{\uparrow}$ *carrier row over $\mathcal{R}$.*

The synchronized row contains exactly the restricted $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{R})$ *row, the displayed* ProbeBundle *row, the* InBundle *membership rows, the certified* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, and the shared* hsame/Cont/Pkg *provenance rows.*

Proof. Project both windows through Definition 126.234. The common refinement determines the same StreamName source tuple on the left and on the right, so the selected classifier, bundle membership, unary-history, pointwise rational-history, hsame, Cont, and Pkg rows are already one displayed tuple over $\mathcal{R}$.

Use Definition 126.221 to read the generated common window as a finite tail-window packet. Its restriction clauses are precisely the restrictions back to $\mathcal{W}_L$ and $\mathcal{W}_R$.

Repack the projected rows as the synchronized StreamName carrier row. The carrier has no coordinate for a quotient stream, host function, completion object, or external index equality, so the repacking uses only the displayed finite-window rows. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameSynchronizedWindowUnionCarrier`

Theorem 126.243 (StreamName synchronized-window RegSeqRat seal handoff)**.** *Let $\mathcal{S}$ be the synchronized* $\mathsf{StreamName}^{\uparrow}$ *carrier row obtained from Definition 126.242. Then every* RegSeqRat *handoff read from $\mathcal{S}$ preserves the endpoint, radius, regularity, readback,* hsame*,* Cont*,* Pkg*, and* NameCert *rows consumed by the finite* $\mathsf{Real}^{\uparrow}$ *seal.*

Equivalently, the downstream consumer reads one synchronized boundary: the StreamName finite-window source rows of $\mathcal{S}$, the accepted RegSeqRat *endpoint and regularity rows selected from those sources, and the finite* $\mathsf{Real}^{\uparrow}$ *seal ledger rows. No route through the original left or right window contributes an additional consumer-visible row.*

Proof. First apply Definition 126.242. This places both displayed windows over one carrier row whose finite-window source tuple is independent of the left or right route.

Next apply Definition 126.268. The synchronized carrier is a common-window surface, so the RegSeqRat handoff rows transported through the two original windows are the same consumer-facing rows.

Apply Definition 126.270 to identify the DyadicRatCore endpoint and radius rows read from the accepted RegSeqRat spine. Then use Definition 126.274 to list the rows consumed by the terminal finite $\mathsf{Real}^{\uparrow}$ seal.

The listed families include endpoint, radius, regularity, readback, hsame, Cont, Pkg, and NameCert provenance. Since each family is projected from the synchronized row or from the accepted RegSeqRat spine, there is no remaining route-dependent row to expose. □

Theorem 126.244 (StreamName synchronized-window NameCert boundary)**.** *The synchronized-window rows of Definition 126.242 and Definition 126.243 form the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *obligation boundary needed by the* $\mathsf{RegSeqRat}/\mathsf{Real}^{\uparrow}$ *seal consumer. The boundary consists of finite carrier rows, common-refinement source determinacy, synchronized common-window transport, accepted* RegSeqRat *endpoint and regularity rows, and the finite seal ledger consumed by* $\mathsf{Real}^{\uparrow}$*.*

This boundary contains no quotient stream, host function, external real, completion theorem, or real-completeness principle.

Proof. Read the carrier face from Definition 126.242. It supplies the finite synchronized StreamName row and identifies its left and right restrictions through common-refinement source determinacy.

Read the consumer face from Definition 126.243. It preserves the RegSeqRat endpoint, radius, regularity, readback, hsame, Cont, Pkg, and NameCert rows passed to the finite $\mathsf{Real}^{\uparrow}$ seal.

Together these two faces exhaust the local obligation boundary: the producer side supplies finite synchronized StreamName rows, and the consumer side reads only the displayed $\mathsf{RegSeqRat}/\mathsf{Real}^{\uparrow}$ seal rows. The excluded quotient, host-function, external-real, completion, and real-completeness objects have no coordinate in either face. □

Theorem 126.245 (StreamName synchronized-window Real seal handoff)**.** *Let* $\mathcal{S}$ *be a synchronized* $\mathsf{StreamName}^{\uparrow}$ *window whose* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint, radius, regularity, and readback rows are transported through Definition 126.243. Then every route from* $\mathcal{S}$ *to the finite* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly the synchronized* ProbeBundle *row,* InBundle *membership rows,* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The handoff has no coordinate for quotient streams, host functions, ambient real completeness, or route-dependent seal rows from the original left or right window.

Proof. Start with the synchronized carrier union of Definition 126.242. It places the left and right finite windows over one displayed ProbeBundle, with the same InBundle, UnaryHistory, pointwise RatHistoryClassifier, hsame, Cont, and Pkg coordinates.

Next apply Definition 126.268. The endpoint, radius, regularity, and readback rows of the accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine are therefore transported through the common window, not through a left-only or right-only route.

Use Definition 126.270 to identify the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius data read by the seal. Then apply Definition 126.274; the finite $\mathsf{Real}^{\uparrow}$ seal consumes only the displayed StreamName, dyadic, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

The excluded objects have no row in this assembled boundary. Quotient streams, host functions, ambient real completeness, and route-dependent seal rows cannot be introduced after the boundary is exhausted by the synchronized packet. □

Theorem 126.246 (StreamName synchronized-window classifier coverage)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be displayed finite* $\mathsf{StreamName}^{\uparrow}$ *windows over certified maps* s, t*, and let* $\mathcal{R}$ *be a certified common-refinement package whose generated carrier is the synchronized carrier of Definition 126.242. Then the classifier row on the synchronized carrier is covered exactly by the two restricted finite-window classifier reads together with the displayed overlap agreement rows of* $\mathcal{R}$*.*

Concretely, a synchronized classifier read is the same data as the left restricted RatStreamNameFiniteWindowClas *row, the right restricted* RatStreamNameFiniteWindowClassifier *row, and the common overlap agreement on* ProbeBundle*,* InBundle*,* UnaryHistory*, pointwise* RatHistoryClassifier*,* hsame*,* Cont*, and* Pkg

rows. The following RegSeqRat/Real$^{\uparrow}$ *handoff reads exactly this synchronized classifier package and no row whose value depends on whether the left or right window was used to reach the common refinement.*

Proof. Apply Definition 126.242 to place the two displayed windows over the carrier generated by $\mathcal{R}$. This gives one carrier row whose visible coordinates are the restricted classifier rows, the bundle rows, the unary-history rows, the pointwise rational-history rows, and the hsame/Cont/Pkg provenance rows.

For the left-to-right reading, project the synchronized carrier back to the two finite windows. The common-refinement source determinacy theorem, Definition 126.234, identifies the source tuple used by each restriction, and Definition 126.236 identifies the two routes through the overlap ledger. Hence every synchronized classifier coordinate is read as one of the two restricted classifier rows or as an overlap agreement row.

For the converse reading, start with the two restricted finite-window classifier rows and the displayed overlap agreement rows. Determinacy fixes the common source tuple, confluence identifies the two overlap routes, and the carrier theorem repacks the rows as a single synchronized classifier package.

Finally apply Definition 126.243. The downstream RegSeqRat/Real$^{\uparrow}$ boundary reads the synchronized classifier package just constructed, so no additional route-dependent row remains for the handoff. □

Theorem 126.247 (StreamName window-union associativity)**.** *Let $\mathcal{W}_0$, $\mathcal{W}_1$, and $\mathcal{W}_2$ be three composable synchronized finite observation windows over the same accepted* StreamName$^{\uparrow}$ *packet. Suppose the displayed common-refinement rows make the left pair, the right pair, and the two iterated synchronized windows use the same finite observation rows,* hsame *transports,* Cont *routes, and* Pkg *provenance. Then the carrier package obtained by first unioning $\mathcal{W}_0$ with $\mathcal{W}_1$ and then unioning the result with $\mathcal{W}_2$ classifies together with the carrier package obtained by first unioning $\mathcal{W}_1$ with $\mathcal{W}_2$ and then unioning $\mathcal{W}_0$ with the result.*

The common classifier is the displayed finite observation package over the same ProbeBundle*,* InBundle*,* UnaryHistory*, pointwise* RatHistoryClassifier*,* hsame*,* Cont*, and* Pkg *rows; neither bracketing contributes an additional route to the* RegSeqRat/Real$^{\uparrow}$ *window assembly.*

Proof. First project Definition 126.242 to the left pair and to the right pair. Both binary unions are displayed over the same accepted StreamName packet and the same common-refinement source rows, so their visible carrier coordinates are the finite observation rows named in the statement.

Project the carrier-union theorem again for the two iterated unions. The outer union on each bracketing only repacks the already displayed ProbeBundle, InBundle, UnaryHistory, pointwise classifier, hsame, Cont, and Pkg rows.

Next apply Definition 126.246. The classifier on either iterated carrier is exhausted by the same restricted finite-window classifier reads together with the same displayed overlap agreement rows. Thus the two bracketings classify the identical displayed observations.

Finally use Definition 126.250 and Definition 126.249. Tail restriction preserves the listed finite rows, and the terminal source boundary has no coordinate for an extra observation route, quotient stream, completion object, host function, or host real equality. □

Theorem 126.248 (StreamName synchronized-window selected-tail exhaustion)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be synchronized finite* StreamName$^{\uparrow}$ *windows over maps s, t, and let the common carrier and classifier rows satisfy Definition 126.242 and Definition 126.246. For any accepted* RegSeqRat *spine read from this synchronized surface, the selected tail surface consumed by the finite* Real$^{\uparrow}$ *seal*

is exhausted by the displayed ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows,* DyadicRatCore *endpoint and radius rows,* hsame*,* Cont*,* Pkg*, and the finite* RealLedgerPolicy *rows.*

No host index equality, quotient stream, completion object, host function, Classical datum, or host real equality is introduced by this selected-tail read.

Proof. Begin with the synchronized carrier theorem. It identifies the left and right finite StreamName windows over one displayed common-refinement carrier and lists the bundle, membership, unary-history, pointwise rational-history, hsame, Cont, and Pkg rows visible on that carrier.

Next apply the classifier coverage theorem. The classifier read used by the accepted RegSeqRat spine is therefore the synchronized classifier package, not a route-dependent copy from either original window.

Now use the RegSeqRat seal handoff from Definition 126.243 together with the terminal cover gluing in Definition 126.274. These two results identify the accepted spine's endpoint, radius, regularity, and readback rows with the finite DyadicRatCore and RealLedgerPolicy rows consumed by the $\mathsf{Real}^{\uparrow}$ seal.

The listed rows are exactly the selected-tail coordinates available to the seal consumer. Since neither the synchronized carrier nor the RegSeqRat handoff has coordinates for host index equality, quotient streams, completion objects, host functions, Classical data, or host real equality, no such row can be read at the terminal boundary. □

Theorem 126.249 (StreamName synchronized-window terminal source exhaustion)**.** *Let $\mathcal{S}$ be a synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *row whose downstream* RegSeqRat *handoff and finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal are read through the carrier of Definition 126.242 and the classifier coverage of Definition 126.246. Then every source row visible to the terminal* RegSeqRat/$\mathsf{Real}^{\uparrow}$ *consumer is one of the following displayed finite BEDC rows: the synchronized* $\mathsf{StreamName}^{\uparrow}$ *classifier rows, the selected* ProbeBundle *rows, the* InBundle *membership rows, the* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *endpoint, radius, regularity, and readback rows, the* DyadicRatCore *endpoint rows, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows.*

The terminal consumer has no coordinate from which to read quotient streams, host-index equality, hidden stream rows, independent regularity witnesses, completion objects, standard bridge rows, or real-completeness principles.

Proof. Apply Definition 126.242. It gives a single synchronized carrier whose visible source coordinates are the finite classifier, bundle, membership, unary-history, pointwise rational-history, and hsame/Cont/Pkg rows.

Apply Definition 126.246. This identifies every synchronized classifier read with the two restricted finite-window classifier reads and the displayed overlap agreement rows, so the classifier contributes no hidden stream coordinate.

Apply Definition 126.243. The accepted RegSeqRat handoff supplies exactly the endpoint, radius, regularity, readback, DyadicRatCore endpoint, and finite $\mathsf{Real}^{\uparrow}$ seal rows consumed downstream.

It remains to exclude extra sources at the terminal face. The no-extra coordinate clause of Definition 126.244 removes quotient streams, host-index equality, hidden stream rows, independent regularity witnesses, completion objects, standard bridge rows, and real-completeness principles from the consumer boundary. □

Theorem 126.250 (StreamName synchronized-window tail-subwindow stability)**.** *Let $\mathcal{S}$ be a synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *row whose carrier and classifier coverage are the rows of Definition 126.242 and Definition 126.246. For any certified finite tail subwindow of $\mathcal{S}$, restriction preserves the same* ProbeBundle*,* InBundle*,* UnaryHistory*, pointwise* RatHistoryClassifier*,* DyadicRatCore*,* RegSeqRat*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows visible at the terminal consumer.*

The restricted terminal consumer has no added coordinate for quotient streams, host-index equality, completion objects, host functions, Classical data, or host real equality.

Proof. Project the synchronized carrier row given by Definition 126.242. Its finite carrier coordinates are the displayed probe-bundle membership, unary observation, classifier, hsame, Cont, Pkg, and NameCert rows, so a certified tail subwindow is obtained by restricting those visible finite coordinates.

Apply the classifier coverage of Definition 126.246. The pointwise rational-history rows remain attached to the restricted unary observations because coverage is stated over the synchronized finite rows and their overlap agreement, not over a host stream quotient.

Use selected-tail exhaustion through Definition 126.243. The accepted RegSeqRat handoff reads only the selected endpoint, radius, regularity, readback, and DyadicRatCore endpoint rows, all of which are already among the restricted finite coordinates.

Finally apply Definition 126.249. The terminal RegSeqRat/$\mathsf{Real}^{\uparrow}$ consumer sees only the listed source rows, so the tail subwindow exposes no quotient stream, host-index equality, completion object, host function, Classical datum, or host real equality. □

Theorem 126.251 (StreamName finite-observation classifier exhaustion)**.** *Let $\mathcal{S}$ be a synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *row whose downstream* RegSeqRat *handoff and finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal are read through the carrier of Definition 126.242. For every selected certified unary observation consumed by that handoff, the consumer-visible classifier data factors through exactly the displayed* ProbeBundle *row,* InBundle *membership row,* UnaryHistory *row, pointwise* RatHistoryClassifier *row,* hsame *comparison,* Cont *route,* Pkg *provenance, and* NameCert *provenance of the synchronized window.*

Thus the single finite observation seen by the downstream RegSeqRat/$\mathsf{Real}^{\uparrow}$ *consumer has no coordinate for a regularity witness, quotient stream, completion row, host index equality, or host real equality.*

Proof. First project the synchronized row through Definition 126.246. The chosen certified unary observation is one member of the covered finite classifier inventory, so its visible rows are the bundle, membership, unary-history, pointwise rational-history, hsame, Cont, and Pkg coordinates of that inventory.

Next apply Definition 126.170 to the chosen observation. The pointwise handoff reads exactly the displayed StreamName carrier and classifier rows at that observation before it reaches the RegSeqRat or finite-window $\mathsf{Real}^{\uparrow}$ consumer.

Finally use Definition 126.249. Its terminal-source boundary adds the NameCert provenance row and removes regularity witnesses, quotient streams, completion rows, host index equality, and host real equality from the terminal consumer surface. □

Theorem 126.252 (StreamName constant reindexing Real seal boundary)**.** *Let d, e : Hist be rational-history endpoints, let r, q : Hist $\to$ Hist be two observation schedules, and let $\mathcal{S}$ be the synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *row built from the independently reindexed constant rational streams* $\mathsf{RatConstStream}_d \circ r$ *and* $\mathsf{RatConstStream}_e \circ q$*. The finite* $\mathsf{Real}^{\uparrow}$ *seal read from $\mathcal{S}$ sees only the constant* $\mathsf{Rat}^{\uparrow}$ *endpoint data for d and e, the two reindexing schedules, finite-window membership in*

the displayed ProbeBundle*, certified* UnaryHistory *rows, and the pointwise* RatHistoryClassifier *rows selected by the synchronized classifier.*

In particular, the seal does not read a quotient stream, completion object, host real equality, host index equality, or independent regularity witness.

Proof. Apply Definition 126.51. It places the two independently reindexed constant streams over the same StreamName classifier surface and reduces every selected classifier row to the RatUp endpoint classifier between d and e.

Then use Definition 126.52. The carrier and classifier data of the independently reindexed constant streams are exactly the endpoint carrier and classifier data at d and e; the schedules r and q contribute only the displayed observation routing.

Pass this synchronized constant row through Definition 126.251. The finite $\mathsf{Real}^{\uparrow}$ seal therefore reads only the constant RatUp endpoint, the reindexing schedules, the finite-window membership and unary-history rows, and the pointwise classifier rows, with no stronger host-level coordinate. □

Theorem 126.253 (StreamName RegSeqRat common-window terminal seal determinacy)**.** *Let $\mathcal{S}_0$ and $\mathcal{S}_1$ be synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *rows. Suppose that they have the same common-refinement source, the same accepted* RegSeqRat *spine, and the same displayed* DyadicRatCore *endpoint and radius rows. If a terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal is read from either synchronized route, then the two reads produce the same consumer-facing seal row.*

The determining surface is exactly the displayed RatStreamNameFiniteWindowClassifier *row,* ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the* DyadicRatCore *endpoint and radius rows, and the visible* hsame*,* Cont*,* Pkg*,* NameCert*, and* RealLedgerPolicy *rows. The terminal seal has no coordinate for quotient streams, host-index equality, completion objects, independent regularity witnesses, standard bridge rows, or real-completeness principles.*

Proof. Apply Definition 126.246. It projects both synchronized routes to the same finite classifier inventory over the common-refinement source, with the same bundle, membership, unary-history, pointwise rational-history, and overlap rows.

Next apply Definition 126.243. The accepted RegSeqRat spine and the displayed DyadicRatCore endpoint and radius rows are therefore the only rational handoff rows available to the terminal finite-window $\mathsf{Real}^{\uparrow}$ seal.

The source boundary is exhausted by Definition 126.249 and by the seal boundary of Definition 126.244. These two boundaries leave only the displayed RatStreamNameFiniteWindowClassifier, ProbeBundle, InBundle, UnaryHistory, RatHistoryClassifier, RegSeqRat, DyadicRatCore, hsame, Cont, Pkg, NameCert, and RealLedgerPolicy rows.

Both routes have the same rows in that inventory by hypothesis. Since the terminal $\mathsf{Real}^{\uparrow}$ seal has no further coordinate beyond this inventory, the consumer-facing seal read from $\mathcal{S}_0$ is the same row as the seal read from $\mathcal{S}_1$. □

Theorem 126.254 (StreamName RegSeqRat uniform-tail seal confluence)**.** *Let $\mathcal{S}_0$, $\mathcal{S}_1$, and $\mathcal{S}_2$ be synchronized finite* $\mathsf{StreamName}^{\uparrow}$ *windows over one accepted rational stream-name schedule. Suppose each adjacent pair admits a common-window package, and suppose all three windows feed the same* RegSeqRat *handoff and the same terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal rows. Then the terminal seal reads from $\mathcal{S}_0$ and $\mathcal{S}_2$ agree on the visible endpoint, dyadic radius, regularity ledger, classifier ledger,* hsame *comparisons,* Cont *routes,* Pkg *provenance,* NameCert *provenance, and terminal* RealLedgerPolicy *rows.*

The confluence is a finite-window statement over the displayed synchronized ledger. It does not identify quotient streams, host-index equality, completion objects, independent regularity witnesses, standard bridge rows, or real completeness principles.

Proof. Apply Definition 126.243 to the $\mathcal{S}_0/\mathcal{S}_1$ synchronized pair and again to the $\mathcal{S}_1/\mathcal{S}_2$ synchronized pair. Each application exposes the same accepted RegSeqRat handoff, endpoint row, dyadic-radius row, regularity ledger, readback row, and terminal finite-window $\mathsf{Real}^{\uparrow}$ seal surface carried by the common schedule.

Next use Definition 126.253 on the adjacent pairs. The first application identifies the terminal read from $\mathcal{S}_0$ with the read from $\mathcal{S}_1$, and the second identifies the read from $\mathcal{S}_1$ with the read from $\mathcal{S}_2$ on exactly the visible finite inventory.

Compose these displayed row equalities by transitivity in the generated common-window ledger. The middle window contributes no separate coordinate: it is the shared finite $\mathsf{StreamName}^{\uparrow}$ synchronization surface, so the first and third terminal reads agree on the endpoint, radius, regularity, classifier, hsame, Cont, Pkg, NameCert, and RealLedgerPolicy rows and on no stronger host-level object. □

Theorem 126.255 (StreamName RegSeqRat finite-window chain compaction)**.** *Let*

$$\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_n$$

be a finite chain of synchronized $\mathsf{StreamName}^{\uparrow}$ *windows over one accepted rational stream-name schedule. Suppose every adjacent pair* $\mathcal{S}_i, \mathcal{S}_{i+1}$ *satisfies the common-window terminal seal determinacy hypotheses, and suppose every window feeds the same accepted* RegSeqRat *handoff. Then all terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal reads through the chain compact to one seal-facing surface.*

The compacted surface has exactly the shared RatStreamNameFiniteWindowClassifier *rows,* ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine,* DyadicRatCore *endpoint and radius rows, and the visible* hsame, Cont, Pkg, NameCert, *and* RealLedgerPolicy *rows. It contains no quotient stream row, host-index equality row, completion object, standard bridge row, or real-completeness principle.*

Proof. Induct on the displayed finite chain. In the first nonempty adjacent case, Definition 126.253 identifies the two terminal reads through the common synchronized window and leaves exactly the stated StreamName, RegSeqRat, DyadicRatCore, transport, provenance, and Real ledger rows.

For the induction step, assume the initial segment has already compacted to one terminal seal-facing surface. The next adjacent pair satisfies the same determinacy hypotheses, so Definition 126.253 identifies the last read of the compacted segment with the read of the new window.

Apply Definition 126.254 to compose the compacted segment, the shared middle window, and the new terminal window. The confluence theorem keeps the accepted RegSeqRat handoff and every visible row on the same synchronized schedule, so the enlarged chain again has one terminal finite-window $\mathsf{Real}^{\uparrow}$ seal surface.

At no step is there a coordinate from which to read quotient streams, host-index equality, completion objects, standard bridge data, or real completeness. The induction therefore compacts the whole finite chain to the single displayed terminal seal inventory. □

126.16 StreamName finite-window lattice meet

Definition 126.256 (StreamName finite-window lattice meet packet)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be accepted finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over displayed source maps* $s, t : \mathsf{BHist} \to \mathsf{BHist}$*. A*

StreamName finite-window lattice meet packet for $(\mathcal{W}_L, \mathcal{W}_R)$ *is the finite* BHist *surface*

$$\mathcal{M} = (Q, B, I, U, D, H, C, P, N_L, N_R)$$

whose rows are as follows:

1. Q *is the common selected rational classifier row* RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{Q})$ *read on the shared* ProbeBundle $\mathcal{Q}$*;*
2. B *is the displayed* ProbeBundle *row for* $\mathcal{Q}$*, together with its restrictions to the selected bundles of* $\mathcal{W}_L$ *and* $\mathcal{W}_R$*;*
3. I *is the family of* InBundle$(n, \mathcal{Q})$ *membership rows for selected unary observations;*
4. U *is the family of* UnaryHistory(n) *observation rows paired with the pointwise rational classifier rows read by* Q*;*
5. D *is the dyadic ledger surface handed to* RegSeqRat *and to the finite-window* Real$^{\uparrow}$ *consumer, including endpoint, radius, denominator, and finite ledger-policy rows;*
6. H *is the componentwise* hsame *transport comparing selected rational classifier rows, probe rows, membership rows, unary rows, dyadic rows, handoff rows, provenance rows, and local names;*
7. C *is the displayed* Cont *route ledger from each accepted* StreamName$^{\uparrow}$ *window through the meet surface to the* RegSeqRat *handoff and the finite-window* Real$^{\uparrow}$ *seal;*
8. P *is the* Pkg *provenance over the accepted windows, the synchronized-window union, the cover-refinement classifier, the common-window transport, and the Real–RegSeq handoff boundary;*
9. N_L *and* N_R *are the local* NameCert$_{\text{StreamName}^{\uparrow}}$ *rows proving that the left and right accepted packages expose this same meet surface.*

The packet is only this finite observation surface. It has no coordinate for quotient stream equality, host index equality, selected limits, host real equality, or ambient completeness.

This meet packet is the StreamName$^{\uparrow}$*-side sibling surface consumed by chapter 301 and chapter 38, with its dyadic ledger row routed through chapter 291.*

Theorem 126.257 (StreamName finite-window lattice meet exhaustion)**.** *Let* $\mathcal{M}$ *be a StreamName finite-window lattice meet packet for two accepted finite-window* StreamName$^{\uparrow}$ *packages. Every* RegSeqRat *consumer or finite-window* Real$^{\uparrow}$ *consumer of* $\mathcal{M}$ *reads only the common selected rational classifier rows, the* ProbeBundle *rows, the* InBundle *membership rows, the* UnaryHistory *observation rows, the dyadic ledger rows, the componentwise transports, the continuation routes, the package provenance, and the local naming rows displayed in Definition 126.256.*

Consequently the meet packet is a shared finite observation surface for the RegSeqRat and RealUp handoffs. It does not provide quotient stream equality, host index equality, selected limits, host real equality, ambient completeness, or any consumer-visible row outside the finite StreamName, RegSeqRat, and RealUp ledger surfaces listed above.

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFiniteWindowLatticeMeet exhaustion`

Proof. First pass both accepted windows through Definition 126.301. The public interface restricts each window to its finite classifier, probe bundle, membership, unary-history, pointwise rational classifier, hsame, Cont, Pkg, and local naming rows.

Next apply Definition 126.242. The two accepted windows are repacked over one synchronized carrier, so their shared selected observations have one displayed ProbeBundle row, one membership family, one unary-history family, and one provenance ledger.

Use Definition 126.265 to read the common probe bundle as a covered finite-window classifier. The cover-refinement step supplies the common selected rational classifier row and keeps the selected endpoint package on the same finite observation rows.

Now apply Definition 126.268. Any $\mathsf{RegSeqRat}$ handoff through the left or right accepted window reads the same common-window surface, with the same componentwise hsame transports, continuation routes, and package provenance.

Finally use Definition 126.274 together with Definition 126.82. These handoff results identify the dyadic ledger rows and the finite-window RealUp consumer rows, while keeping regularity, sealing, limits, quotient rows, host index equality, host real equality, and ambient completeness outside the meet packet. The displayed coordinates of $\mathcal{M}$ therefore exhaust every consumer read. □

Theorem 126.258 (StreamName finite-window lattice meet associativity)**.** *Let $\mathcal{W}_A, \mathcal{W}_B, \mathcal{W}_C$ be accepted finite-window $\mathsf{StreamName}^{\uparrow}$ packages over displayed source maps. Suppose that $\mathcal{M}_{AB}$, $\mathcal{M}_{BC}$, $\mathcal{M}_{(AB)C}$, and $\mathcal{M}_{A(BC)}$ are accepted StreamName finite-window lattice meet packets whose common selected observations are read on the same finite $\mathsf{ProbeBundle}$ after reassociating the three local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows. Then the two iterated meet packets $\mathcal{M}_{(AB)C}$ and $\mathcal{M}_{A(BC)}$ expose the same common observation surface.*

More precisely, reassociation changes only the bracketing of the three local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows. The selected rational classifier rows, $\mathsf{ProbeBundle}$ rows, $\mathsf{InBundle}$ membership rows, $\mathsf{UnaryHistory}$ observation rows, dyadic ledger rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and the RegSeqRat and finite-window RealUp handoff rows are unchanged. Neither iterated meet exports quotient stream equality, host index equality, selected limits, host real equality, ambient completeness, or any consumer-visible row outside the finite StreamName, RegSeqRat, and RealUp ledger surfaces.

Proof. Apply Definition 126.257 to $\mathcal{M}_{AB}$ and $\mathcal{M}_{BC}$. Each binary meet is exhausted by its selected rational classifier, probe-bundle, membership, unary-history, dyadic, transport, continuation, provenance, and local naming rows.

Apply the same exhaustion theorem to the two iterated packets $\mathcal{M}_{(AB)C}$ and $\mathcal{M}_{A(BC)}$. By hypothesis these packets read the same finite $\mathsf{ProbeBundle}$ after reassociating the local NameCert rows, so the classifier, membership, unary-history, dyadic, transport, continuation, provenance, RegSeqRat, and finite-window RealUp handoff rows are the same displayed rows on both sides.

It remains to compare the local naming coordinates. Use Definition 126.259 to commute the binary meet packets inside either bracketing until the three local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows are read in the chosen reassociated order. Symmetry fixes Q, B, I, U, D, H, C, P and exchanges only already displayed local naming rows, so no downstream consumer receives a new row during the reassociation.

The exhaustion theorem now applies once more to the reassociated iterated surface. Since every RegSeqRat or finite-window RealUp consumer is confined to the displayed finite rows, and those rows agree for $\mathcal{M}_{(AB)C}$ and $\mathcal{M}_{A(BC)}$, the two iterated meets expose the same common observation surface and no excluded quotient, host, limit, real-equality, completeness, or hidden consumer coordinate. □

Lemma 126.259 (StreamName finite-window lattice meet symmetry)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be accepted finite-window $\mathsf{StreamName}^{\uparrow}$ packages over displayed source maps $s, t : \mathsf{BHist} \to \mathsf{BHist}$. Suppose that*

$$\mathcal{M} = (Q, B, I, U, D, H, C, P, N_L, N_R)$$

is an accepted StreamName finite-window lattice meet packet for $(\mathcal{W}_L, \mathcal{W}_R)$*. Then*

$$\mathcal{M}^{\mathrm{op}} = (Q, B, I, U, D, H, C, P, N_R, N_L)$$

is an accepted StreamName finite-window lattice meet packet for $(\mathcal{W}_R, \mathcal{W}_L)$ *over the same common observation surface. The rows* Q, B, I, U, D, H, C, P *are unchanged, and the only exchanged coordinates are the two local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows.*

Consequently every $\mathsf{RegSeqRat}$ *consumer or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of* $\mathcal{M}^{\mathrm{op}}$ *reads exactly the same selected rational classifier row, probe-bundle row, membership rows, unary-history rows, dyadic ledger row, componentwise transport row, continuation route row, and package provenance row as it reads from* $\mathcal{M}$*, with the left and right local naming rows interchanged. The symmetric packet exports no quotient stream equality, host index equality, selected limit, host real equality, ambient completeness, or consumer-visible row outside the finite meet surface.*

Proof. Unpack the meet packet using Definition 126.256. The coordinate Q is read on the shared $\mathsf{ProbeBundle}$, B is the same displayed bundle row together with its two restrictions, and I, U, D, H, C, P are common rows over that same finite surface. None of these rows carries an additional coordinate that distinguishes the ordered pair $(\mathcal{W}_L, \mathcal{W}_R)$ from the ordered pair $(\mathcal{W}_R, \mathcal{W}_L)$ after the same two restrictions are read in the opposite order.

The only ordered coordinates in the displayed packet are the local naming rows. In $\mathcal{M}$, the row N_L proves that the left accepted package $\mathcal{W}_L$ exposes the meet surface, and N_R proves that the right accepted package $\mathcal{W}_R$ exposes the same meet surface. For the ordered pair $(\mathcal{W}_R, \mathcal{W}_L)$, the first local naming row must name $\mathcal{W}_R$ and the second must name $\mathcal{W}_L$. These are precisely N_R and N_L. Thus the tuple $(Q, B, I, U, D, H, C, P, N_R, N_L)$ satisfies the same packet clauses for the swapped ordered pair.

It remains to identify what a downstream consumer can read. By Definition 126.257, every $\mathsf{RegSeqRat}$ consumer and every finite-window $\mathsf{Real}^{\uparrow}$ consumer is confined to the displayed classifier, bundle, membership, unary-history, dyadic, transport, continuation, provenance, and local naming coordinates of the meet packet. The construction of $\mathcal{M}^{\mathrm{op}}$ leaves the common coordinates Q, B, I, U, D, H, C, P fixed and merely exchanges the two already displayed naming rows. Hence the swapped packet has the same consumer-facing finite observation surface, with no route to any excluded quotient, host, limit, real-equality, completeness, or hidden consumer row. □

Theorem 126.260 (StreamName finite-window intersection)**.** *Let* A *and* B *be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *windows over the same source request. Then the lattice-meet rows and common-refinement rows display an intersection window* $A \cap B$ *whose visible surface consists exactly of the shared request classifier, the common finite observation cells, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the finite* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. Later* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal consumers may inspect only this displayed intersection route.*

Proof. Apply the finite-window lattice meet theorem Definition 126.257. It supplies one shared finite packet for the two accepted windows over the same request, with classifier, observation, dyadic, transport, replay, provenance, and local naming rows exposed objectwise.

Next use Definition 126.405. The RegSeqRat readback and Real seal rows are read through the common refinement of the selected StreamName windows, so the downstream consumer sees the meet packet rather than either original window in isolation.

The intersection route is therefore the only route visible to later RegSeqRat and Real-facing consumers. Since the carrier exposes no quotient stream equality, selected limit, host real equal-

ity, or ambient completeness coordinate, no other intersection witness can be inspected by those consumers. □

126.17 StreamName lattice-meet terminal factorization

Theorem 126.261 (StreamName lattice-meet terminal factorization)**.** *A* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet packet determines the terminal finite surface consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and finite* $\mathsf{Real}^{\uparrow}$ *seal handoffs. The terminal surface consists exactly of the shared rational classifier, the* ProbeBundle *row,* InBundle *membership rows,* UnaryHistory *observation rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, the finite* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* hsame *and* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows.*

No terminal consumer of the meet packet may replace that surface by quotient stream equality, host index equality, selected limits, host real equality, ambient completeness, or an off-ledger $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Start with the meet exhaustion theorem Definition 126.257. It identifies the common finite packet read by RegSeqRat and RealUp consumers: classifier, probe bundle, membership, unary-history, dyadic, transport, continuation, provenance, and local naming rows.

Next use the synchronized-window Real seal handoff of Definition 126.245. The synchronized window sends the same selected observations to the finite $\mathsf{Real}^{\uparrow}$ seal route, so the seal-facing read is not independent of the meet packet.

Now apply the RegSeqRat–RealUp source-boundary exhaustion theorem Definition 126.274. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback and finite $\mathsf{Real}^{\uparrow}$ handoff on the same StreamName, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and local naming rows.

Finally combine the pointwise handoff Definition 126.170 with the Real seal factorization boundary Definition 126.142. These rows account for the terminal readback and seal coordinates and leave no carrier slot for quotient streams, host index equality, selected limits, host real equality, ambient completeness, or off-ledger seal data. □

Theorem 126.262 (StreamName lattice-meet terminal source uniqueness)**.** *For a* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet packet, every terminal consumer that reaches the finite* $\mathsf{Real}^{\uparrow}$ *seal through the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger displays the same terminal source rows. These rows are precisely the shared rational classifier,* ProbeBundle *row,* InBundle *rows,* UnaryHistory *rows, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, finite* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Proof. First apply Definition 126.261. It gives the terminal finite surface consumed by the accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback and by the finite $\mathsf{Real}^{\uparrow}$ seal handoff, and it already lists the source rows available to such a consumer.

The uniqueness of that displayed source is the meet exhaustion content of Definition 126.257. The finite-window meet exposes one common packet, so the classifier, bundle, membership, unary-history, dyadic, continuation, provenance, and local naming rows cannot be duplicated by another terminal source inside the same meet route.

Next use Definition 126.245. The synchronized window sends the Real-facing read to the same selected finite observations, so a terminal $\mathsf{Real}^{\uparrow}$ consumer has no second observation window from which to rebuild the seal row.

Finally combine Definition 126.274 with Definition 126.142. The RegSeqRat readback and Real seal boundary are exhausted by the StreamName, dyadic, hsame, Cont, Pkg, and local NameCert rows displayed above, hence every accepted terminal consumer names the same source surface. □

Theorem 126.263 (StreamName lattice-meet terminal pullback nonescape)**.** *Pulling the terminal finite* $\mathsf{Real}^{\uparrow}$ *seal read back along the* $\mathsf{StreamName}^{\uparrow}$ *lattice-meet route cannot introduce quotient stream equality, host index equality, a selected limit, host real equality, ambient completeness, or an off-ledger* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Begin with the terminal source uniqueness theorem Definition 126.262. Any pullback of the terminal seal read through the accepted $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ route must return to the same displayed source rows: classifier, bundle, membership, unary observations, dyadic ledger, readback, finite seal, transport, continuation, provenance, and local naming data.

Now apply Definition 126.261 to that unique source surface. Its nonescape clause states that the terminal meet packet leaves no carrier slot for quotient stream equality, host index equality, selected limits, host real equality, ambient completeness, or off-ledger seal data.

The pullback therefore repacks the existing terminal rows; it does not add a hidden route. The exclusions remain stable under the return along the lattice-meet route because the Real-facing seal coordinate is already fixed by the accepted readback and the finite source boundary. □

Theorem 126.264 (StreamName lattice-meet Real-seal support determinacy)**.** *Let two accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice-meet packets be consumed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and by the same finite* $\mathsf{Real}^{\uparrow}$ *seal handoff. If they agree on the displayed finite-window classifier, the* $\mathsf{ProbeBundle}$ *row,* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *observation rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows, then they determine the same finite* $\mathsf{Real}^{\uparrow}$ *seal support packet.*

No quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness row, or second* $\mathsf{Real}^{\uparrow}$ *source can change that support packet once those common lattice-meet rows are fixed.*

Proof. First apply Definition 126.261. The accepted readback and finite seal handoff factor through the displayed finite surface: classifier, bundle, membership, unary observation, dyadic ledger, readback, transport, continuation, provenance, and local naming rows.

Next apply Definition 126.262. Since the two packets agree on precisely those source rows, every terminal consumer names the same displayed source surface. Thus the finite $\mathsf{Real}^{\uparrow}$ seal handoff has no second source from which to rebuild a different support packet.

Now apply Definition 126.263. Pulling the seal read back along the lattice-meet route only repacks the already displayed rows and cannot introduce quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness, or another Real-facing source coordinate.

Finally use the Real seal handoff from the StreamName common-refinement route, namely Definition 126.221 together with Definition 126.142. The common finite pullback is the only consumer surface available to the finite $\mathsf{Real}^{\uparrow}$ seal, so agreement on the lattice-meet rows determines the same seal support packet. □

Theorem 126.265 (StreamNameUp finite-window cover-refinement classifier)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$ *and probe bundles* $\mathcal{P}_1, \mathcal{P}_2, \mathcal{Q} : \mathsf{ProbeBundle}$*, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_1) \quad \textit{and} \quad \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_2).$$

Assume every requested bundled unary observation of $\mathcal{Q}$ *is covered by one of the two certified windows:*

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{Q}) \to \mathsf{UnaryHistory}(n) \to \mathsf{InBundle}(n, \mathcal{P}_1) \vee \mathsf{InBundle}(n, \mathcal{P}_2).$$

Then the requested window is certified:

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}).$$

Moreover, for any selected n in $\mathcal{Q}$ with $\mathsf{UnaryHistory}(n)$ *and displayed one-extension endpoints*

$$\mathsf{hsame}(s(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t(n), \mathsf{E1}(b)),$$

the selected observation carries the full endpoint package of Definition 126.27.

Proof. First combine the two certified windows. The union clause of Definition 126.25 repacks the $\mathcal{P}_1$ and $\mathcal{P}_2$ classifiers as a classifier on the displayed union bundle.

Next restrict the union classifier to the requested bundle. For a bundled unary observation n of $\mathcal{Q}$, the cover hypothesis chooses whether the membership row comes from $\mathcal{P}_1$ or from $\mathcal{P}_2$. In either case the corresponding inclusion into the displayed union bundle gives the subwindow map required by the subwindow clause of Definition 126.25. This yields

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}).$$

Finally fix a selected requested observation n together with its bundle membership, unary witness, and displayed one-extension endpoints. Apply Definition 126.27 to the $\mathcal{Q}$ classifier just constructed. The resulting point classifier, positive-denominator rows, unary endpoint rows, empty and zero-headed exclusions, and one-tail readback are exactly the full endpoint package for the requested observation. □

Theorem 126.266 (StreamName cover-refinement transport closure)**.** *For certified observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ and probe bundles $\mathcal{P}_1, \mathcal{P}_2, \mathcal{Q} : \mathsf{ProbeBundle}$, suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_1) \quad \textit{and} \quad \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P}_2).$$

Assume every requested bundled unary observation of $\mathcal{Q}$ is covered by one of the two certified windows:

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{Q}) \to \mathsf{UnaryHistory}(n) \to \mathsf{InBundle}(n, \mathcal{P}_1) \vee \mathsf{InBundle}(n, \mathcal{P}_2).$$

Assume also that the endpoints are transported componentwise on every bundled unary observation of $\mathcal{Q}$:

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{Q}) \to \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(s(n), s'(n))$$

and

$$\forall n,\ \mathsf{InBundle}(n, \mathcal{Q}) \to \mathsf{UnaryHistory}(n) \to \mathsf{hsame}(t(n), t'(n)).$$

Then

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{Q}).$$

Moreover, for every selected n in $\mathcal{Q}$ with $\mathsf{UnaryHistory}(n)$ *and histories $a, b : \mathsf{Hist}$ displaying transported one-extension endpoints*

$$\mathsf{hsame}(s'(n), \mathsf{E1}(a)) \quad \textit{and} \quad \mathsf{hsame}(t'(n), \mathsf{E1}(b)),$$

the selected observation exports the same finite endpoint package to the $\mathsf{RegSeqRat}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff surface: the selected point classifier, positive-denominator rows, unary endpoint rows, empty and zero-headed exclusions, one-tail readback,* $\mathsf{InBundle}$ *membership, bundled* $\mathsf{UnaryHistory}$ *row, finite* Cont *ledger, displayed* hsame *transports, and* Pkg *provenance. No host index, quotient representative, completion object, metric row, or* $\mathsf{Real}^{\uparrow}$ *completeness datum is introduced.*

Proof. First build the classifier on the requested cover. Apply Definition 126.265 to the two certified windows and to the displayed cover of $\mathcal{Q}$. This gives

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{Q}).$$

Next transport that classifier along the endpoint rows over $\mathcal{Q}$. The endpoint-transport field of Definition 126.28 applies componentwise to the two displayed hsame transport families, yielding

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{Q}).$$

Finally fix a selected bundled unary observation and its displayed transported one-extension endpoints. Apply Definition 126.27 to the transported classifier. The resulting selected endpoint package is the finite package consumed by the downstream surface, and Definition 126.142 records that this surface factors only through the listed StreamName, RegSeqRat, ProbeBundle, InBundle, UnaryHistory, Cont, hsame, and Pkg rows. □

Theorem 126.267 (StreamName cover-refinement Real seal source invariance)**.** *For certified observation maps $s, t, s', t' : \mathsf{Hist} \to \mathsf{Hist}$ and probe bundles $\mathcal{P}_1, \mathcal{P}_2, \mathcal{Q} : \mathsf{ProbeBundle}$, assume the two-window cover hypotheses for $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{Q}$ from Definition 126.266, together with the same componentwise endpoint transports from s, t to s', t' over $\mathcal{Q}$. Then every downstream finite-window $\mathsf{Real}^{\uparrow}$ seal read over $\mathcal{Q}$ reads the same $\mathsf{StreamName}^{\uparrow}$ source surface after refinement.*

That invariant surface consists exactly of the finite-window classifier, the selected ProbeBundle *row,* InBundle *membership,* UnaryHistory*, the pointwise* RatHistoryClassifier*, displayed* hsame *transports, the* Cont *ledger, and* Pkg *provenance. No host index, quotient representative, completion object, metric row, or external $\mathsf{Real}^{\uparrow}$ completeness datum becomes available under the covered refinement.*

Proof. First apply Definition 126.265 to the two certified windows and the displayed cover of $\mathcal{Q}$. This builds the finite-window classifier on the requested probe bundle using only the already certified source rows.

Next apply Definition 126.266. The componentwise endpoint transports carry the classifier to s', t' and keep the selected endpoint package on the same $\mathcal{Q}$ bundle, with its InBundle membership, UnaryHistory row, point classifier, Cont ledger, displayed hsame rows, and Pkg provenance.

Finally pass to the $\mathsf{StreamName}^{\uparrow}$ Real–RegSeqRat handoff and the $\mathsf{Real}^{\uparrow}$ seal factorization boundary. Those results identify exactly the consumer rows available to a finite-window seal read, so the refinement can transport the source surface but cannot add a host index, quotient representative, completion object, metric row, or external completeness datum. □

126.18 StreamName RegSeqRat common-window transport

Theorem 126.268 (StreamName RegSeqRat common-window transport)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be finite $\mathsf{StreamName}^{\uparrow}$ windows over the same* ProbeBundle *$\mathcal{P}$ and the same displayed source maps $s, t : \mathsf{BHist} \to \mathsf{BHist}$. Suppose that for every certified member n with $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$, the two windows expose pointwise* hsame*-equal rational-history observations for $s(n)$ and $t(n)$. Then the* RegSeqRat *handoff rows transported through $\mathcal{W}_L$ and $\mathcal{W}_R$ are the same consumer-facing rows, and the finite-window $\mathsf{Real}^{\uparrow}$ ledger-boundary rows obtained from those handoffs are the same consumer-facing rows.*

Equivalently, the common bundle determines one visible handoff surface: the restricted RatStreamNameFiniteWi*row, the* ProbeBundle *row, the* InBundle *membership rows, the certified* UnaryHistory *rows, the pointwise rational-history classifier rows, and the displayed* hsame/Cont/Pkg *provenance rows. The*

transport adds no completion object, quotient row, host-index equality, RegSeqRat *regularity row, or* $\mathsf{Real}^{\uparrow}$ *seal classifier.*

Proof. Project both windows to the common bundle. The membership and unary-history coordinates are the same displayed $\mathsf{InBundle}(n, \mathcal{P})$ and $\mathsf{UnaryHistory}(n)$ rows, so the only possible route dependence is in the selected pointwise classifier rows.

Use Definition 126.171. The assumed pointwise hsame comparisons transport the selected classifier rows within the same ProbeBundle, so both windows expose the same finite-window StreamName classifier surface after projection.

Apply Definition 126.170. It reads the RegSeqRat handoff only from the carrier rows, the common bundle membership rows, the certified unary observations, and the transported pointwise classifier rows. Thus the two projected windows supply the same handoff rows to the RegSeqRat consumer.

Finally apply Definition 126.82 and Definition 126.234. The separation theorem keeps regularity and seal obligations on the consumer side, while common-refinement determinacy identifies the source tuple read through the shared bundle. Therefore the downstream finite-window $\mathsf{Real}^{\uparrow}$ ledger-boundary rows are the same consumer-facing rows as well. □

Theorem 126.269 (StreamName public RegSeqRat finite-window composition)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be public finite-window* $\mathsf{StreamName}^{\uparrow}$ *packages over the same* ProbeBundle $\mathcal{P}$ *and displayed maps* $s, t : \mathsf{BHist} \to \mathsf{BHist}$*. Suppose that the public rows exposed by the two packages agree on the finite-window classifier* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$*, the* $\mathsf{InBundle}(-, \mathcal{P})$ *membership rows, the certified* UnaryHistory *rows, the pointwise* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *rows, and the displayed* hsame/Cont/Pkg *provenance rows.*

Then the RegSeqRat *handoff rows read from* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *agree, and the finite-window* $\mathsf{Real}^{\uparrow}$ *ledger-boundary rows obtained from those handoffs agree, as the same consumer-facing common-window surface. The composed surface exports only the listed public StreamName rows, the RegSeqRat handoff rows determined by them, and the finite RealUp ledger-boundary rows. It exports no* RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion object, quotient row, or host-index equality.*

Proof. Apply Definition 126.301 to both public packages. The interface restricts each finite window to the displayed bundle, unary-history, pointwise rational-history classifier, hsame, Cont, and Pkg rows, so both packages present the same public finite-window source surface over $\mathcal{P}$.

Next apply Definition 126.268. Since the two public projections expose the same RatStreamNameFiniteWindowCl and the same pointwise provenance rows, the RegSeqRat handoff read from either package is the same consumer-facing handoff surface, and the finite RealUp ledger-boundary rows obtained from that handoff are the same rows.

Finally use Definition 126.82 and Definition 126.234. The separation theorem keeps regularity, sealing, completion, quotient, and host-index data outside the exported finite-window rows, while source determinacy identifies the common bundle source read by downstream consumers. Repacking therefore keeps exactly the public classifier, membership, unary-history, pointwise classifier, hsame, Cont, Pkg, RegSeqRat handoff, and finite RealUp ledger-boundary rows. □

Theorem 126.270 (StreamName RegSeqRat dyadic source determinacy)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be finite* $\mathsf{StreamName}^{\uparrow}$ *common-window packages over the same* ProbeBundle $\mathcal{P}$ *and the same accepted* RegSeqRat *spine. Suppose that for every selected observation* n*, the displayed* $\mathsf{UnaryHistory}(n)$*,* $\mathsf{InBundle}(n, \mathcal{P})$*, and pointwise* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *rows agree between the two packages*

by the shown hsame/Cont/Pkg *provenance. Then the DyadicRatCore endpoint rows handed to the* $\mathsf{Real}^{\uparrow}$ *finite-window ledger are the same consumer-facing rows.*

The determined handoff surface is exactly the RatStreamNameFiniteWindowClassifier$(s, t, \mathcal{P})$ *row, the* ProbeBundle *row, the* InBundle *membership rows, the* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, the DyadicRatCore endpoint rows, and the shared* Cont/hsame/Pkg/RealLedgerPolicy *provenance. It exposes no completion object, quotient representative, host-index equality, or hidden selector row.*

Proof. First apply Definition 126.268. The shared bundle and the displayed pointwise hsame comparisons identify the StreamName finite-window classifier, selected bundle membership rows, certified unary-history rows, and accepted RegSeqRat handoff rows read from $\mathcal{W}_L$ and $\mathcal{W}_R$.

Next use Definition 126.217. Once the common-window handoff is placed in a tail-window coverage packet, every readable consumer row is one of the displayed classifier, bundle, unary-history, pointwise rational-history, RegSeqRat, DyadicRatCore, and finite ledger rows.

Finally apply Definition 38.314. The finite $\mathsf{Real}^{\uparrow}$ observation can consume the DyadicRatCore endpoint rows only through the endpoint denominator and dyadic source package supplied by the same accepted spine. Since the cited boundaries contain no completion, quotient, host-index equality, or hidden selector row, the two packages hand the same DyadicRatCore source rows to the $\mathsf{Real}^{\uparrow}$ finite-window ledger. □

Theorem 126.271 (StreamName RegSeqRat common-window tail stability)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be finite* $\mathsf{StreamName}^{\uparrow}$ *common-window packages over a shared* ProbeBundle $\mathcal{P}$*. Suppose the two packages determine the same* RegSeqRat *dyadic source rows through Definition 126.270. Then every certified tail subwindow of $\mathcal{P}$ exposes the same* RegSeqRat *handoff rows and the same terminal* $\mathsf{Real}^{\uparrow}$ *ledger-boundary rows.*

The tail restriction is read only through the displayed InBundle *membership rows, certified* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows, the accepted* RegSeqRat *spine, DyadicRatCore endpoint rows, and the shared* Cont/hsame/Pkg/RealLedgerPolicy *provenance. It does not expose regularity, completion, quotient, or host-index data as part of the source packet.*

Proof. Apply Definition 126.268. The shared bundle and pointwise rational-history transport identify the common-window handoff rows before any tail restriction is taken.

Apply Definition 126.270. The hypothesis that the same dyadic source rows are determined fixes the accepted RegSeqRat spine and the DyadicRatCore endpoint rows carried into the terminal $\mathsf{Real}^{\uparrow}$ ledger.

Now restrict to a certified tail subwindow. The tail is still a finite subfamily of the same ProbeBundle, so the inherited InBundle and UnaryHistory rows select the same pointwise classifier rows and the same endpoint entries.

Finally use Definition 126.82. It keeps regularity, completion, quotient, and host-index data outside the source packet, so the tail can expose only the already identified RegSeqRat handoff rows and terminal ledger-boundary rows. □

Theorem 126.272 (StreamName RegSeqRat common-window composition handoff)**.** *Let $\mathcal{W}_0$, $\mathcal{W}_1$, and $\mathcal{W}_2$ be finite* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose $\mathcal{W}_0$ and $\mathcal{W}_1$ share a certified common-window transport surface over the* ProbeBundle *row $\mathcal{P}_{01}$, and suppose $\mathcal{W}_1$ and $\mathcal{W}_2$ share a certified common-window transport surface over the* ProbeBundle *row $\mathcal{P}_{12}$. Then the composed handoff from $\mathcal{W}_0$ to $\mathcal{W}_2$ exposes exactly the same consumer-facing rows read by* RegSeqRat*: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, the* UnaryHistory *rows, the* InBundle *rows, the* RatHistoryClassifier *rows, the DyadicRatCore source rows, and the displayed* hsame/Cont/Pkg *provenance rows.*

The composed handoff exports no RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal classifier, completion datum, quotient row, or host-index equality. Every readable downstream row is inherited from the two certified common-window surfaces and their certified tail restrictions.*

Proof. Apply Definition 126.268 to the $\mathcal{W}_0/\mathcal{W}_1$ surface over $\mathcal{P}_{01}$. This identifies the finite-window classifier, unary-history rows, bundle-membership rows, pointwise rational-history rows, and displayed hsame/Cont/Pkg provenance read by the first handoff leg.

Apply the same theorem to the $\mathcal{W}_1/\mathcal{W}_2$ surface over $\mathcal{P}_{12}$. Since both certified surfaces pass through the same middle window $\mathcal{W}_1$, their visible handoff rows agree at the middle boundary before any downstream consumer reads them.

Use Definition 126.270 on both legs. The dyadic source rows supplied to the RegSeqRat spine and to the terminal finite-window $\mathsf{Real}^{\uparrow}$ ledger are determined by the displayed classifier, bundle, unary-history, rational-history, hsame, Cont, and Pkg rows on the two certified surfaces.

Finally apply Definition 126.271. Certified tail restrictions remain finite subwindows of the displayed bundle rows, so composing the two legs cannot add any row outside the inherited consumer surface. The separation of regularity, seal, completion, quotient, and host-index obligations is therefore preserved by the composed handoff from $\mathcal{W}_0$ to $\mathcal{W}_2$. □

Theorem 126.273 (StreamName RegSeqRat common-window endpoint exhaustion)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be common-window packages over the same* ProbeBundle $\mathcal{P}$ *and the same accepted* RegSeqRat *spine. For any downstream finite-window* $\mathsf{Real}^{\uparrow}$ *ledger consuming the endpoint handoff, every endpoint row it reads is exactly one of the displayed finite rows:*

1. *the* RatStreamNameFiniteWindowClassifier *row;*
2. *the shared* ProbeBundle *row, its* InBundle *rows, and the certified* UnaryHistory *rows;*
3. *the pointwise* RatHistoryClassifier *rows selected by the common window;*
4. *the accepted* RegSeqRat *spine rows;*
5. *the DyadicRatCore endpoint and radius rows determined from that spine;*
6. *the shared* Cont*,* hsame*,* Pkg*, and* RealLedgerPolicy *provenance rows.*

No hidden selector row, host-index equality, quotient row, completion object, or $\mathsf{Real}^{\uparrow}$ *seal classifier is part of the endpoint surface.*

Proof. First project the common-window packages through Definition 126.268. This fixes the visible finite-window classifier, shared bundle, bundle-membership, unary-history, pointwise rational-history, and displayed provenance rows before the downstream endpoint ledger is read.

Next apply Definition 126.270. The accepted RegSeqRat spine determines the DyadicRatCore endpoint and radius rows consumed by the finite-window $\mathsf{Real}^{\uparrow}$ ledger from the same displayed source rows; no source-equivalence transport is needed.

Use Definition 126.271 to pass to the finite tail subwindow actually consumed by the ledger. Tail restriction is projection to a finite subfamily of the same ProbeBundle, so it cannot expose a new selector, host index, quotient, completion, or seal classifier row.

Finally apply Definition 38.314. The terminal $\mathsf{Real}^{\uparrow}$ finite-window consumer reads endpoint data only through the displayed endpoint package, DyadicRatCore source rows, and RealLedgerPolicy boundary. Combining the four projections exhausts the endpoint surface by exactly the listed rows. □

Theorem 126.274 (StreamName-RegSeqRat-RealUp source boundary exhaustion)**.** *Let $\mathcal{W}$ be a certified common finite* $\mathsf{StreamName}^{\uparrow}$ *window over a displayed* ProbeBundle $\mathcal{P}$*. Suppose its selected rational history rows are read by a* RegSeqRat *dyadic-tolerance consumer and then by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then every seal-boundary row consumed by the terminal* $\mathsf{Real}^{\uparrow}$ *package is projected from exactly one of the following visible source families:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *window rows, namely* ProbeBundle, InBundle*, certified* UnaryHistory*, pointwise rational-history classifier rows,* hsame *transport, and* Cont *readback;*
2. *the* RegSeqRat *dyadic endpoint, radius, distance, and regularity rows selected by the accepted dyadic-tolerance spine;*
3. *the finite* $\mathsf{Real}^{\uparrow}$ *seal ledger rows, including the endpoint denominator package, dyadic source rows, and* RealLedgerPolicy *boundary.*

Conversely, any compatible triple of rows from these three families repacks into the same boundary package consumed by the finite-window $\mathsf{Real}^{\uparrow}$ *seal. The boundary contains no quotient sequence, completion object, host index equality, or host real equality.*

Proof. First project the certified common window through Definition 126.268. This fixes the ProbeBundle, InBundle, UnaryHistory, pointwise rational-history classifier, hsame, and Cont rows available to the StreamName face.

Next apply Definition 126.270. The dyadic endpoint rows handed to the downstream consumer are determined by those displayed StreamName rows together with the accepted RegSeqRat spine. Thus the endpoint, radius, distance, and regularity rows form the whole RegSeqRat source family read by the seal.

Use Definition 126.272 to keep the handoff stable under the common-window composition used by downstream readback. Then apply Definition 38.355 and Definition 38.344. These RealUp rows state that the finite seal reads the same StreamName-RegSeqRat handoff together with the endpoint denominator, dyadic source, and RealLedgerPolicy ledger boundary.

For the reverse direction, start with a compatible triple from the three listed families. The StreamName rows repack the finite common window, the RegSeqRat rows repack the accepted dyadic-tolerance source, and the RealUp ledger rows repack the finite seal boundary. The cited pullback and correspondence theorems then identify the repacked package with the original terminal boundary. Since each cited boundary explicitly excludes quotient, completion, host-index equality, and host real equality rows, no additional source row is available. □

Theorem 126.275 (StreamName-RegSeqRat common-window seal functoriality)**.** *Let $\mathcal{W}_0$, $\mathcal{W}_1$, and $\mathcal{W}_2$ be certified common finite* $\mathsf{StreamName}^{\uparrow}$ *windows over the same public endpoints and the same accepted* RegSeqRat *dyadic-tolerance spine. Suppose the displayed transports $\mathcal{W}_0 \to \mathcal{W}_1$ and $\mathcal{W}_1 \to \mathcal{W}_2$ preserve the finite-window classifier, the selected unary observations, the pointwise rational-history rows, the accepted* RegSeqRat *spine, and the DyadicRatCore endpoint and radius rows.*

Then the $\mathsf{Real}^{\uparrow}$ *seal source obtained by applying Definition 126.274 after the two displayed transports is the same finite source surface as the one obtained by first using the common-window composition handoff of Definition 126.272 and then applying source-boundary exhaustion. The common source surface consists only of the displayed StreamName window rows, the accepted RegSeqRat dyadic rows, and the finite RealUp seal ledger rows; it contains no quotient row, completion row, or host real equality.*

Proof. Start with the two displayed common-window transports. By Definition 126.272, their composition preserves the same classifier, selected unary observations, pointwise rational-history rows,

accepted RegSeqRat spine, and DyadicRatCore endpoint and radius rows as the single composed transport $\mathcal{W}_0 \to \mathcal{W}_2$.

Apply Definition 126.273 to both the two-stage transport and the composed transport. In each case the endpoint surface is exhausted by the displayed finite window rows and the accepted dyadic-tolerance spine, so the downstream endpoint ledger has the same visible source families.

Finally apply Definition 126.274. The terminal $\mathsf{Real}^{\uparrow}$ seal reads only the StreamName window rows, the RegSeqRat dyadic endpoint rows, and the finite seal ledger rows. Since both transport routes expose the same rows to this exhaustion theorem, they expose the same finite source surface. The cited boundaries leave no coordinate for quotient rows, completion rows, or host real equality, so no further source can distinguish the two routes. □

Theorem 126.276 (StreamName-RegSeqRat finite-window ledger confluence)**.** *Let $\mathcal{W}_0$, $\mathcal{W}_1$, and $\mathcal{W}_2$ be finite* $\mathsf{StreamName}^{\uparrow}$ *windows over the same public endpoints and the same accepted* RegSeqRat *dyadic-tolerance spine. Suppose the displayed transports $\mathcal{W}_0 \to \mathcal{W}_1$, $\mathcal{W}_1 \to \mathcal{W}_2$, and their composed common-window route preserve the shared* ProbeBundle *row, the* InBundle *membership rows, the certified* UnaryHistory *rows, the pointwise* RatHistoryClassifier *rows, the accepted spine, the* DyadicRatCore *endpoint and radius rows, and the visible* Cont/hsame/Pkg/RealLedgerPolicy *provenance.*

Then every finite-window $\mathsf{Real}^{\uparrow}$ *seal consumer reads the same boundary whether it follows the staged route $\mathcal{W}_0 \to \mathcal{W}_1 \to \mathcal{W}_2$ or the composed common-window route $\mathcal{W}_0 \to \mathcal{W}_2$. The common boundary is exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* RegSeqRat *dyadic endpoint and radius rows, and the finite* $\mathsf{Real}^{\uparrow}$ *seal ledger rows. It contains no quotient row, completion object, host-index equality, host function,* Classical *datum, or host real equality.*

Proof. First apply Definition 126.268. It identifies the rows preserved by each displayed finite-window transport: the shared ProbeBundle row, InBundle membership, certified UnaryHistory, pointwise rational-history classification, accepted spine, dyadic endpoint and radius rows, and the visible provenance rows.

Next use Definition 126.270. The endpoint and radius rows delivered to a downstream seal are forced by the displayed StreamName rows together with the accepted RegSeqRat spine, so the staged and composed routes cannot expose different dyadic source rows.

Apply Definition 126.273 and Definition 126.274. The first theorem exhausts the downstream endpoint surface by the common-window rows and the accepted spine; the second theorem says a finite $\mathsf{Real}^{\uparrow}$ seal consumer reads exactly that exhausted surface together with the finite seal ledger rows.

Finally apply Definition 126.275. It identifies the two staged seal readback routes with the composed route before the consumer reads the boundary. Since the cited exhaustion results leave no coordinate for quotient rows, completion objects, host-index equality, host functions, Classical data, or host real equality, the listed finite rows are the whole common boundary. □

Theorem 126.277 (StreamName RegSeqRat terminal window union)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be two finite* $\mathsf{StreamName}^{\uparrow}$ *windows feeding a common* RegSeqRat *source boundary. If their endpoint rows admit the common-window exhaustion packet and their seal routes compose functorially, then the union window presents the same terminal* RegSeqRat *source to any* $\mathsf{Real}^{\uparrow}$ *boundary consumer.*

The union exposes only finite observation rows, dyadic endpoint ledgers, hsame *transport,* Cont *routes,* Pkg *provenance, and the* RegSeqRat *seal. It exposes no hidden infinite tail, quotient equality, completion object, host index equality, or host real equality.*

Proof. Apply Definition 126.273 to $\mathcal{W}_L$ and $\mathcal{W}_R$. The endpoint rows on both sides are exhausted by the same finite observation families, dyadic endpoint ledgers, accepted RegSeqRat spine rows, and displayed hsame, Cont, and Pkg provenance.

Next apply Definition 126.274. It sends each exhausted endpoint package to the same finite $\mathsf{Real}^{\uparrow}$ source boundary, using only the StreamName window rows, the RegSeqRat dyadic source rows, and the finite seal ledger rows.

Finally use Definition 126.275. The functorial seal route identifies the two-stage and union-window routes before the terminal boundary consumer reads them. Hence the union window presents the same terminal RegSeqRat source, and the listed finite rows are the full consumer surface. □

Theorem 126.278 (StreamName RegSeqRat terminal window intersection)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be two finite* $\mathsf{StreamName}^{\uparrow}$ *windows feeding a common* RegSeqRat *source boundary. Suppose both windows restrict to a certified common sub-bundle K, and suppose the endpoint rows visible on K admit the common-window exhaustion packet.*

Then every finite-window $\mathsf{Real}^{\uparrow}$ *boundary consumer reads the same terminal* RegSeqRat *source from the restricted intersection. The restricted intersection exposes exactly the* $\mathsf{InBundle}(-, K)$ *membership rows, the selected rational-history classifier rows, dyadic endpoint ledgers,* hsame, Cont, Pkg *provenance, and the displayed* RegSeqRat *seal. It exposes no hidden tail, quotient equality, completion object, host index equality, or host real equality.*

Proof. Apply Definition 126.329 to the certified common sub-bundle K. The intersection surface is exhausted by the membership rows $\mathsf{InBundle}(-, K)$, the selected rational-history classifier rows, the shared dyadic endpoint ledgers, and the displayed hsame, Cont, and Pkg provenance.

Next apply Definition 126.273 to the restricted endpoint rows on K. This packages the exposed intersection rows as the common-window endpoint surface feeding the accepted RegSeqRat spine, with no additional row for a hidden tail or host index equality.

Apply Definition 126.274. Every finite-window $\mathsf{Real}^{\uparrow}$ consumer reads only the exhausted endpoint surface, the dyadic source rows, and the finite seal ledger, so both restricted windows present the same visible source families to the consumer.

Finally use Definition 126.275. The seal functoriality identifies the terminal route obtained by restricting through K with the route read by the finite boundary consumer. Since the cited exhaustion theorems leave no coordinate for quotient equality, completion objects, host index equality, or host real equality, the displayed finite rows are the full terminal intersection source. □

Theorem 126.279 (StreamName RegSeqRat terminal window cover gluing)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be finite* $\mathsf{StreamName}^{\uparrow}$ *windows feeding a common accepted* RegSeqRat *dyadic-tolerance spine. Suppose their union satisfies the terminal-window union boundary of Definition 126.277, and suppose their certified common sub-bundle satisfies the terminal-window intersection boundary of Definition 126.278.*

Then every finite-window $\mathsf{Real}^{\uparrow}$ *seal consumer reads one cover-glued terminal source surface. That surface is exactly the union of the displayed* ProbeBundle *rows,* InBundle *membership rows, certified* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows,* DyadicRatCore *endpoint and radius rows, accepted* RegSeqRat *spine rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows, identified along the intersection agreement. It contains no quotient row, completion object, host-index equality, host function,* Classical *datum, or host real equality.*

Proof. Apply Definition 126.277 to the joined window. The union route exposes the displayed finite observation rows, dyadic endpoint ledgers, accepted RegSeqRat spine rows, hsame transport,

Cont routes, Pkg provenance, and finite seal policy rows as the terminal source boundary for the $\mathsf{Real}^{\uparrow}$ consumer.

Apply Definition 126.278 on the certified overlap. The intersection route says that the common sub-bundle has the same membership, rational-history classifier, dyadic endpoint, hsame, Cont, Pkg, and seal rows on both restrictions, so the two sides agree on every row that can be read twice.

Use endpoint exhaustion and source-boundary exhaustion to compare the two routes. By Definition 126.273, every endpoint coordinate reaching the accepted spine is already among the displayed finite StreamName and dyadic rows. By Definition 126.274, the downstream seal reads only those exhausted rows together with the finite RealLedgerPolicy boundary.

Finally apply Definition 126.275. It identifies the cover-glued readback before the $\mathsf{Real}^{\uparrow}$ consumer reads the boundary. Since the union and intersection routes exhaust all readable source rows and agree on the overlap, the cover-glued terminal surface is precisely the listed finite union modulo the intersection agreement, with no quotient, completion, host-index, host-function, Classical, or host-real coordinate. □

Theorem 126.280 (StreamName RegSeqRat cover-glued ledger uniqueness)**.** *Let $\mathcal{W}$ be a finite* $\mathsf{StreamName}^{\uparrow}$ *union, let I be its certified overlap sub-bundle, let S be an accepted* RegSeqRat *dyadic-tolerance spine, and let P be a finite* $\mathsf{Real}^{\uparrow}$ *seal policy. Suppose two terminal source surfaces are built from the same $\mathcal{W}$, I, S, and P by the cover-gluing route of Definition 126.279.*

Then every terminal ledger row read by a finite-window $\mathsf{Real}^{\uparrow}$ *consumer is identical as a consumer-facing row in the two presentations. The visible rows are exactly the* ProbeBundle *rows,* InBundle *membership rows, certified* UnaryHistory *rows, pointwise* RatHistoryClassifier *rows,* DyadicRatCore *endpoint and radius rows, accepted* RegSeqRat *spine rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows. No quotient, completion object, host-index equality, host function,* Classical *datum, or host real equality is exported.*

Proof. Apply Definition 126.279 to the first presentation and again to the second presentation. Both cover-glued terminal surfaces are therefore assembled from the same finite union rows and the same certified intersection rows, with the same accepted RegSeqRat spine and the same finite $\mathsf{Real}^{\uparrow}$ seal policy.

Next apply Definition 126.276. It identifies the staged StreamName-to-RegSeqRat route with the composed StreamName-to-RegSeqRat-to-RealUp route, so a terminal consumer cannot distinguish a row by the route used to present it.

Use Definition 126.273 for the dyadic endpoint and radius coordinates. Use Definition 126.274 for the finite seal boundary. Together they show that every readable coordinate is one of the displayed finite families.

The two presentations agree on all union rows by hypothesis and agree on all intersection rows by the certified overlap. Since the exhaustion theorems leave no other consumer-readable coordinate, each terminal ledger read returns the same consumer-facing row in both presentations. This is only a uniqueness statement for the concrete finite ledger; it does not transport a source equivalence and it exports no host-level equality or completion object. □

Theorem 126.281 (StreamName RegSeqRat common-window selection determinacy)**.** *Let $\mathcal{W}_L$ and $\mathcal{W}_R$ be finite* $\mathsf{StreamName}^{\uparrow}$ *common-window packages over the same* ProbeBundle *and the same accepted* RegSeqRat *dyadic-tolerance spine. Suppose each selected observation row on both windows carries matching* InBundle *membership, pointwise* RatHistoryClassifier *transport,* DyadicRatCore *endpoint and radius rows, and displayed* hsame*,* Cont*, and* Pkg *provenance.*

Then every terminal RegSeqRat *source read through $\mathcal{W}_L$, through $\mathcal{W}_R$, through their finite union, through their certified intersection, or through the cover-gluing route is the same consumer-readable source. The statement introduces no host-index equality, quotient row, hidden selector, completion object, or* $\mathsf{Real}^{\uparrow}$ *arithmetic or order row.*

Proof. Apply Definition 126.268 to each selected row. The shared ProbeBundle membership, pointwise rational-history classifier transport, dyadic endpoint data, hsame, Cont, and Pkg provenance put both windows over the same accepted finite source boundary.

Next use Definition 126.270. Since the accepted dyadic-tolerance spine is common, the transported endpoint and radius rows determine one RegSeqRat source family for every selected terminal read.

The union route is handled by Definition 126.277. It reads the same displayed source rows after adjoining the finite windows, and Definition 126.273 shows that no additional endpoint coordinate is available beyond the displayed finite rows.

The intersection route is handled by Definition 126.278. The certified overlap exposes the same membership, classifier, endpoint, radius, transport, continuation, and provenance rows on both restrictions, so the source read on the overlap is the source already determined by the common spine.

Finally apply Definition 126.279. The cover gluing route is assembled from the terminal union and terminal intersection routes along their agreement, so it has no remaining coordinate by which a different selected window could yield a different consumer-readable source. The excluded host-index, quotient, hidden-selector, completion, and $\mathsf{Real}^{\uparrow}$ arithmetic or order rows therefore stay absent throughout the comparison. □

Theorem 126.282 (StreamName RegSeqRat tail-stable Real seal)**.** *Let s, t : BHist $\to$ BHist be displayed* $\mathsf{StreamName}^{\uparrow}$ *rows, let P and Q be finite* ProbeBundle *windows, and let $\mathcal{W}$ be their common-window transport surface. Suppose that the transport row of Definition 126.268, the tail-stability row of Definition 126.271, the finite-window ledger confluence row of Definition 126.276, and the finite* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 126.112 all hold on the displayed* UnaryHistory *rows,* InBundle *membership rows,* hsame *transport rows, and* RatStreamNameFiniteWindowClassifier *rows selected by $\mathcal{W}$.*

Then the certified tail shift of the common window determines the same consumer-facing $\mathsf{Real}^{\uparrow}$ *finite-window seal package as the original common window. The package consists exactly of the displayed StreamName rows, the selected unary observations, bundle-membership rows, rational-history classifier rows, accepted* RegSeqRat *dyadic source rows, DyadicRatCore endpoint and radius rows, finite ledger rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows. It exports no completion object, quotient row, host-index equality, hidden selector, host function, or extra* RegSeqRat *regularity witness beyond the displayed finite-window seal rows.*

Proof. First apply Definition 126.271. The certified tail subwindow inherits the same InBundle membership, UnaryHistory observations, pointwise RatHistoryClassifier rows, accepted RegSeqRat spine, DyadicRatCore endpoint and radius rows, and Cont/hsame/Pkg/RealLedgerPolicy provenance as the original common window.

Next use Definition 126.281. Since the selected observations on the original window and on its certified tail carry the same displayed classifier and provenance rows, every terminal RegSeqRat source read is the same consumer-readable source.

Apply Definition 126.276. The staged tail route and the composed common-window route therefore present one finite ledger boundary to any downstream $\mathsf{Real}^{\uparrow}$ seal consumer.

Finally invoke Definition 126.112. It reads exactly the StreamName source rows, accepted RegSeqRat pointwise seal rows, bundle-membership rows, unary observations, finite ledger entries, and provenance rows already identified above. The separation clauses in the StreamName handoff chain leave no coordinate for completion objects, quotient representatives, host-index equalities, hidden selectors, host functions, or additional regularity witnesses, so the tail-stable common window exports the same public finite-window $\mathsf{Real}^{\uparrow}$ seal package and nothing outside it. □

Theorem 126.283 (StreamName RegSeqRat glued Real seal uniqueness)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be displayed* $\mathsf{StreamName}^{\uparrow}$ *rows. Let two terminal common-window routes for s and t agree on their* InBundle *rows,* UnaryHistory *rows,* RatStreamNameFiniteWindowClassifier *rows, accepted* RegSeqRat *dyadic source rows, DyadicRatCore endpoint and radius rows, finite ledger rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows.*

Then the glued consumer-facing $\mathsf{Real}^{\uparrow}$ *finite-window seal package determined by those two routes is unique. It consists only of the displayed StreamName source rows, selected unary observations, bundle-membership rows, rational-history classifier rows, accepted* RegSeqRat *dyadic sources, DyadicRatCore endpoint and radius rows, ledger entries,* hsame *transports,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows. No completion point, quotient representative, host-index equality, hidden selector, host function, or extra regularity witness is exported.*

Proof. First apply Definition 126.279. The two terminal routes glue through the same terminal cover because their displayed bundle membership, unary observations, classifiers, dyadic source rows, endpoint-radius data, transport, continuation, provenance, and ledger-policy rows agree on the common window.

Next use Definition 126.280. After the cover has been glued, both routes expose the same finite ledger boundary. The ledger read is therefore the ledger already present in the displayed common-window surface, not a separately chosen downstream object.

Apply Definition 126.281. Since the two glued routes share the displayed classifier, accepted source, endpoint, radius, hsame, Cont, Pkg, and RealLedgerPolicy rows, they determine the same terminal consumer-readable RegSeqRat source.

Finally invoke Definition 126.282. The $\mathsf{Real}^{\uparrow}$ seal reads only the glued finite-window package already identified by the preceding steps. The separation clauses in the tail-stable seal leave no coordinate for a completion point, quotient representative, host-index equality, hidden selector, host function, or extra regularity witness, so both terminal routes produce one glued consumer-facing finite-window seal package. □

Theorem 126.284 (StreamName RegSeqRat terminal tail subwindow factorization)**.** *Let s, t : $\mathsf{BHist} \to \mathsf{BHist}$ be displayed* $\mathsf{StreamName}^{\uparrow}$ *rows. Let a terminal common-window route for s and t determine the glued consumer-facing* $\mathsf{Real}^{\uparrow}$ *finite-window seal package of Definition 126.283. Let a certified tail subwindow of that route preserve the displayed* UnaryHistory *rows,* InBundle *rows,* RatStreamNameFiniteWindowClassifier *rows, accepted* RegSeqRat *spine rows, DyadicRatCore endpoint and radius rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows.*

Then the certified tail subwindow factors through the same consumer-facing $\mathsf{Real}^{\uparrow}$ *finite-window seal package as the terminal common-window route. The factorization exports exactly the displayed StreamName source rows, selected unary observations, bundle-membership rows, rational-history classifier rows, accepted* RegSeqRat *spine rows, DyadicRatCore endpoint and radius rows, ledger entries,* hsame *transports,* Cont *routes,* Pkg *provenance, and* RealLedgerPolicy *rows. It exports no quotient row, completion row, host-index row, hidden selector row, arithmetic row, order row, or bridge row.*

Proof. First apply Definition 126.271. The certified tail subwindow has the same displayed unary, bundle, classifier, accepted spine, endpoint-radius, transport, continuation, provenance, and ledger-policy surface as the terminal route.

Next use Definition 126.282. The tail surface therefore presents the same finite ledger boundary to the $\mathsf{Real}^{\uparrow}$ seal consumer and contributes no coordinate outside the displayed finite-window package.

Finally invoke Definition 126.283. Since the terminal route and its certified tail now read the same public rows, uniqueness identifies their consumer-facing $\mathsf{Real}^{\uparrow}$ seal packages and yields the stated factorization. □

Definition 126.285 (Scheduled constant observation package)**.** *For histories $d, e : \mathsf{Hist}$ and observation schedules $r, q : \mathsf{Hist} \to \mathsf{Hist}$, the scheduled constant observation package is the finite StreamNameUp-facing row tuple*

$$\mathsf{SchedConstObs}(d, e, r, q)$$

consisting of the RatUp seed carrier rows $\mathsf{RatHistoryCarrier}(d)$ *and* $\mathsf{RatHistoryCarrier}(e)$*, the RatUp seed classifier row* $\mathsf{RatHistoryClassifier}(d, e)$*, and the two scheduled constant stream rows*

$$\mathsf{RatConstStream}_d \circ r, \qquad \mathsf{RatConstStream}_e \circ q.$$

Its StreamNameUp carrier entries are $\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$ *and* $\mathsf{RatStreamNameCarrier}(\mathsf{RatC}$ $q)$*, and its StreamNameUp classifier entry is*

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q).$$

The schedules r and q are retained only as observation rows; the rational seed histories remain the only RatUp endpoints exported by the package.

Theorem 126.286 (Scheduled constant observation exactness)**.** *For histories $d, e : \mathsf{Hist}$ and schedules $r, q : \mathsf{Hist} \to \mathsf{Hist}$, the scheduled constant observation package of Definition 126.285 supplies exactly the following StreamNameUp rows:*

$$\mathsf{RatStreamNameCarrier}(\mathsf{RatConstStream}_d \circ r)$$

and

$$\mathsf{RatStreamNameClassifier}(\mathsf{RatConstStream}_d \circ r, \mathsf{RatConstStream}_e \circ q),$$

if and only if the corresponding RatUp seed rows $\mathsf{RatHistoryCarrier}(d)$ *and* $\mathsf{RatHistoryClassifier}(d, e)$ *hold.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName scheduled constant observation exactness`

Proof. For the carrier row, unfold $\mathsf{RatConstStream}_d \circ r$. Every certified unary observation is sent back to the same seed history d, so the StreamNameUp carrier clause is precisely the RatUp carrier clause for d.

For the classifier row, apply Definition 126.52. Its independently scheduled classifier clause identifies the classifier of $\mathsf{RatConstStream}_d \circ r$ against $\mathsf{RatConstStream}_e \circ q$ with the seed classifier $\mathsf{RatHistoryClassifier}(d, e)$.

Combining these two readings gives the forward direction by projection from the scheduled package. The reverse direction repacks the RatUp carrier and classifier rows as the two displayed StreamNameUp rows, with r and q serving only as observation schedules. □

Theorem 126.287 (StreamName scheduled window reindex stability)**.** *Let d, e : Hist be seed histories, let r, q : Hist $\to$ Hist be observation schedules, and let u : Hist $\to$ Hist be a unary reindexing row whose requested finite observation window is carried by the public StreamName finite-window interface of Definition 126.301. Then the constant scheduled observation package* SchedConstObs(d, e, r, q) *of Definition 126.285 transports through u by replacing the displayed schedule rows with $r \circ u$ and $q \circ u$:*

$$\mathsf{SchedConstObs}(d, e, r, q) \rightsquigarrow \mathsf{SchedConstObs}(d, e, r \circ u, q \circ u).$$

The transported package still supplies the same finite-window input consumed by the RealUp independently scheduled constant rational embedding certificate in Definition 126.293.

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName scheduled window reindex stability`

Proof. Unfold the package of Definition 126.285. Reindexing changes only the two observation rows, sending the displayed streams to $\mathsf{RatConstStream}_d \circ r \circ u$ and $\mathsf{RatConstStream}_e \circ q \circ u$; the RatUp seed histories remain d and e.

Apply Definition 126.301 to the requested window. The public interface says that the reindexed finite window is still a StreamNameUp finite-window row, so the carrier projections requested by the consumer are the same finite family of displayed stream-name rows.

Now apply Definition 126.286. Since the two reindexed streams are still constant over the same seed histories, their carrier and classifier rows reduce to the same RatUp carrier and classifier rows for d and e.

Finally use Definition 126.293. RealUp consumes exactly these scheduled StreamNameUp coordinates, so the transported scheduled package feeds the same finite-window input. The construction introduces no quotient, completion object, external metric primitive, or RealUp completeness datum. □

Theorem 126.288 (Scheduled subwindow constant observation coherence)**.** *Let d, e : Hist be seed histories, let r, q : Hist $\to$ Hist be observation schedules, and let U : ProbeBundle be a finite observation window accepted by the public StreamName finite-window interface of Definition 126.301. Let V : ProbeBundle be a certified subwindow whose rows are only* InBundle *membership rows,* UnaryHistory *rows, and pointwise* hsame *rows inherited from U. Then the package obtained by restricting*

$$\mathsf{SchedConstObs}(d, e, r, q)$$

directly to V exposes the same RatHistoryCarrier(d)*,* RatHistoryClassifier(d, e)*,* RatStreamNameCarrier*, and* RatStreamNameClassifier *rows as the package first read over U and then restricted to V. Only the displayed schedule rows are restricted; the RatUp seed rows and the pointwise classifier rows are identical in the two readings.*

Proof. Unfold the package of Definition 126.285. Reading the package over U records the seed carrier rows for d and e, the seed classifier row for (d, e), and the two scheduled constant StreamName rows.

Apply Definition 126.301 to the certified subwindow V. Its public rows are inherited InBundle, UnaryHistory, and pointwise hsame rows, so restricting from U to V does not create a host index, quotient representative, or completion datum.

Now apply Definition 126.286. On either reading, the StreamName carrier and classifier rows reduce to the same RatUp seed carrier and seed classifier rows for d and e.

It remains only to compare the schedule coordinates. Direct restriction to V and restriction through U select the same unary-history and pointwise hsame rows from the finite-window interface, so the schedule rows are the only rows whose displayed domain changes. The exposed RatUp seed and pointwise classifier rows are therefore coherent for the subwindow. □

Theorem 126.289 (Scheduled window RealUp feed)**.** *Let $d, e : \mathsf{Hist}$ be seed histories, let $r, q : \mathsf{Hist} \to \mathsf{Hist}$ be observation schedules, and let $u : \mathsf{Hist} \to \mathsf{Hist}$ be a unary reindexing row accepted by the public finite-window interface of Definition 126.301. Then the transported package*

$$\mathsf{SchedConstObs}(d, e, r \circ u, q \circ u)$$

supplies exactly the StreamName finite-window rows consumed by Definition 38.314 and Definition 38.320. Its exported rows are the two reindexed constant stream carriers, their exact classifier, the RatUp seed rows for d and e, and the package provenance already present in Definition 126.285; it adds no host Nat*, quotient representative, completion object, or unrecorded stream object.*

Proof. Unfold Definition 126.285. The transported tuple has the same seed histories d, e and only replaces the two observation schedules by $r \circ u$ and $q \circ u$.

Apply Definition 126.287. The public finite-window interface accepts the requested unary row u, so the transported scheduled package is still the finite StreamName window package exposed to downstream consumers.

Now apply Definition 126.286. The two reindexed streams remain constant over the same RatUp seed histories, and the classifier row reduces to the displayed RatUp seed classifier. Thus every StreamName row read by the finite window is already one of the package rows.

Finally repack the rows through Definition 38.314 and Definition 38.320. These boundary theorems consume only the finite StreamName carriers, classifier rows, and ledger provenance supplied above, so the RealUp feed has no additional host index, quotient, or hidden stream datum. □

Theorem 126.290 (StreamName-RegSeqRat-RealUp finite-window feed)**.** *Let $d, e : \mathsf{Hist}$ be seed histories, let $r, q : \mathsf{Hist} \to \mathsf{Hist}$ be observation schedules, and let $u : \mathsf{Hist} \to \mathsf{Hist}$ be a unary reindexing row whose finite request bundle $\mathcal{P} : \mathsf{ProbeBundle}$ is accepted by the public finite-window interface of Definition 126.301. Then the transported scheduled constant package*

$$\mathsf{SchedConstObs}(d, e, r \circ u, q \circ u)$$

supplies one concrete finite-window boundary feed for the three consumers Definition 38.251, Definition 38.314, and Definition 38.320. The supplied rows are exactly the two scheduled StreamName carriers, their RatUp seed classifier, the finite-window classifier row selected by $\mathcal{P}$, the endpoint denominator rows, and the RegSeqRat dyadic source rows.

Equivalently, the construction uses only BHist*,* $\mathsf{ProbeBundle}$*,* $\mathsf{InBundle}$*,* $\mathsf{UnaryHistory}$*,* hsame*, RatUp classifiers, StreamName finite-window rows, and RegSeqRat dyadic ledger rows. It does not add a host* Nat *index, quotient representative, completion object, host stream function, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Unfold Definition 126.285. The transported package has seed histories d, e, scheduled streams $\mathsf{RatConstStream}_d \circ r \circ u$ and $\mathsf{RatConstStream}_e \circ q \circ u$, the RatUp carrier rows for the two seeds, and the RatUp seed classifier row for (d, e).

Apply Definition 126.287. Since $\mathcal{P}$ is accepted by the public finite-window interface, the unary row u gives a finite StreamName window rather than a host index family. The two scheduled carriers and the selected finite-window classifier are therefore the same rows read by Definition 126.289.

Next apply Definition 126.286. The two scheduled streams remain constant over the same RatUp seed histories after the reindexing, so the StreamName classifier reduces to the RatUp seed classifier and the endpoint denominator rows remain the finite RatUp endpoint rows attached to d and e.

Finally pass the resulting finite-window classifier through Definition 38.251. The RegSeqRat seal reads the same finite StreamName window together with the dyadic source rows. Then Definition 38.314 and Definition 38.320 repack those rows as the RealUp finite-window ledger boundary. Every displayed component is therefore supplied by the scheduled package plus the RegSeqRat dyadic ledger, with no additional analytic or completion-side input. □

Theorem 126.291 (StreamName-RegSeqRat seal scope)**.** *Every accepted scheduled finite-window feed for*

$$\mathsf{SchedConstObs}(d, e, r \circ u, q \circ u)$$

supplies exactly the StreamName carrier and classifier rows, the RatUp seed classifier, the finite-window ProbeBundle *row, the endpoint denominator rows, and the RegSeqRat dyadic ledger rows required by Definition 38.251. The same packet is then consumable by Definition 38.314 and Definition 38.320 without adding host* Nat *indices, quotient representatives, completion objects, metric data, or* $\mathsf{Real}^{\uparrow}$ *completeness rows.*

Proof. Start from Definition 126.290. It identifies the accepted feed with the two scheduled StreamName carriers, the RatUp seed classifier, the selected finite-window classifier row, the endpoint denominator rows, and the RegSeqRat dyadic source ledger.

Project the finite-window rows through Definition 126.287. The unary reindexing row keeps the window inside the public ProbeBundle interface, so the seal sees a finite StreamName packet rather than a host index family.

Pass the classifier and dyadic source rows through Definition 38.251. This is precisely the RegSeqRat seal obligation: the seed classifier, endpoint denominators, and dyadic ledger rows are the only rows read by the seal.

Finally repack the same packet through Definition 38.314 and Definition 38.320. These RealUp boundary theorems consume the finite StreamName rows and RegSeqRat ledger as given, so no completion-side or host analytic datum is added. □

Theorem 126.292 (StreamName-RegSeqRat-RealUp feed common-row determinacy)**.** *Let* $d, e :$ Hist *be seed histories, let* $r, q :$ Hist $\to$ Hist *be observation schedules, and let* $u :$ Hist $\to$ Hist *be a certified unary reindexing row. Let* $\mathcal{P} :$ ProbeBundle *be the same accepted finite-window request for two scheduled finite-window feeds built from* $\mathsf{SchedConstObs}(d, e, r \circ u, q \circ u)$*. If the two feeds also share the same* RatStreamNameClassifier *seed row for* (d, e) *and the same RegSeqRat dyadic ledger rows, then they expose the same StreamName finite-window carrier rows, the same StreamName classifier row selected by* $\mathcal{P}$*, the same input row for Definition 38.251, and the same RealUp finite-window ledger boundary row.*

Thus common finite source data determine the rows read by the RegSeqRat seal and by the RealUp finite-window consumer. No host index, quotient representative, completion object, metric row, or $\mathsf{Real}^{\uparrow}$ *completeness datum can distinguish the two feeds.*

Proof. Apply Definition 126.290 to each feed. Both applications start from the same BHist seed histories d, e, the same accepted ProbeBundle request $\mathcal{P}$, the same InBundle evidence, and the same UnaryHistory row u.

Project the two inventories through Definition 126.291. The scope theorem says that the visible rows are exactly the scheduled StreamName carriers, the selected finite-window classifier,

the RatUp seed classifier, the endpoint denominator rows, and the RegSeqRat dyadic ledger rows. Since the hypotheses identify those source rows, the two inventories have the same displayed StreamName carrier and classifier rows.

Now read the RegSeqRat input. The shared hsame, Cont, unary reindexing, accepted bundle membership, and dyadic ledger hypotheses are precisely the rows consumed by Definition 38.251. Therefore the seal input row cannot vary between the two feeds.

Finally pass the common seal packet through Definition 38.314 and Definition 38.320. These RealUp boundary statements read the finite StreamName packet and RegSeqRat ledger already identified above, so every RealUp finite-window ledger boundary row is the same for the two feeds and no additional datum remains available to separate them. □

Theorem 126.293 (Scheduled constant observation RealUp feed)**.** *For histories d, e : Hist and schedules r, q : Hist $\to$ Hist, the scheduled constant observation package is the StreamNameUpside input consumed by the RealUp independently scheduled constant rational embedding certificate of Definition 38.538. Equivalently, its two scheduled stream rows and its exact classifier row are the StreamNameUp coordinates used by the downstream target* RealIndependentReindexedConstant_rational_embedding_

Proof. Read the RealUp package in Definition 38.538. Its two StreamNameUp coordinates are exactly $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_e \circ q$, and its classifier chain passes through the independently scheduled StreamNameUp classifier between those rows.

By Definition 126.286, those are precisely the carrier and classifier rows delivered by the scheduled constant observation package. Thus RealUp consumes no additional StreamNameUp data beyond this scheduled package when forming the constant rational embedding certificate. □

Theorem 126.294 (Scheduled constant StreamName RealUp seal exactness)**.** *For histories d, e : Hist and schedules r, q : Hist $\to$ Hist, the package $\mathsf{SchedConstObs}(d, e, r, q)$ supplies exactly the StreamNameUp carrier rows, the StreamNameUp classifier row, the RatUp seed rows, and the provenance rows consumed by Definition 38.314, Definition 38.320, and Definition 126.293. It supplies no host stream function, quotient representative, completion datum, or RealUp completeness witness.*

Proof. Unfold Definition 126.285. The displayed entries are the two scheduled constant StreamName rows $\mathsf{RatConstStream}_d \circ r$ and $\mathsf{RatConstStream}_e \circ q$, the RatUp seed histories d, e, the classifier row between the scheduled names, and the package provenance recording their finite-window source.

Apply Definition 126.286 to the same package. It identifies the carrier and classifier rows as the exact StreamNameUp surface induced by the two RatUp seeds, so there is no additional StreamName classifier or host stream map outside the displayed packet.

Next use Definition 126.293. The RealUp consumer coordinates for the independently scheduled constant rational embedding are precisely the two scheduled StreamName rows and their exact classifier row. Thus the consumer side reads the rows already isolated by the scheduled observation package.

Finally repack these rows through Definition 38.314 and Definition 38.320. Those theorems read only the finite StreamName window, the RatUp seed rows, and the provenance ledger, so the seal exports exactly the finite-window RealUp input surface and not a quotient representative, completion datum, or completeness witness. □

Theorem 126.295 (StreamName constant pointwise inhabited)**.** *Let d : Hist be a certified rational endpoint row, and let r : Hist $\to$ Hist range over any displayed unary schedule in the scheduled*

constant observation package. The package supplies a $\mathsf{StreamName}^{\uparrow}$ *carrier whose pointwise classifier is stable under every such schedule; hence the constant certified point determines an inhabited StreamName witness for each displayed finite observation row. The witness is the finite scheduled row* $\mathsf{RatConstStream}_d \circ r$*, not an infinite stream object.*

Proof. Start with the scheduled constant observation package at the diagonal seeds (d, d) and the chosen displayed schedule r. By Definition 126.286, the carrier row and classifier row are exactly the StreamNameUp rows induced by the certified rational endpoint d.

Apply Definition 126.287. Reindexing along the displayed unary schedule changes only the finite observation window; the pointwise classifier remains the same certified constant classifier after transport through the schedule.

Finally use Definition 126.294. The RealUp seal consumes exactly the scheduled StreamName carrier, classifier, RatUp seed, and provenance rows already displayed by the package. Therefore the constant certified point inhabits the StreamName carrier through each finite scheduled row, with no primitive infinite object or hidden completion datum. □

Theorem 126.296 (StreamName constant observation map row)**.** *For any certified point row over the L10 source, the constant observation schedule determines a* $\mathsf{StreamName}^{\uparrow}$ *carrier row whose pointwise classifier,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row are stable under every unary finite observation. The row is available to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *as a finite StreamName handoff, without claiming real completeness or quotient stream equality.*

Proof. Begin with the constant-stream and constant-window boundary rows. They identify a certified point with the constant StreamName carrier read over each finite observation window, so a unary finite observation changes only the displayed schedule.

Apply constant-window transport totality. The pointwise classifier and hsame transport are stable under the unary observation because the seed point row is unchanged and only the finite window is reindexed.

Now use the scheduled constant observation package. Its Cont replay and Pkg provenance record the same constant schedule as a finite StreamNameUp packet, and the local NameCert row names that packet before downstream consumption.

Finally read the consumers. $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ may consume the finite StreamName carrier, classifier, transport, replay, provenance, and naming rows supplied here; they do not receive a real-completeness claim, quotient stream equality, or hidden completion coordinate from this map row. □

Theorem 126.297 (StreamName pointwise map stability)**.** *Let r be a unary reindexing map on finite histories, and let F be a pointwise map on rational-history observations. If r sends every unary window to a unary window, F preserves the* $\mathsf{RatHistory}^{\uparrow}$ *carrier, and F preserves the* $\mathsf{RatHistory}^{\uparrow}$ *classifier, then the* $\mathsf{StreamName}^{\uparrow}$ *carrier and classifier rows are stable both under reindexing by r and under pointwise mapping by F. The exported route remains a finite StreamName row: it contains only the StreamName carrier, classifier,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data, and it does not introduce quotient stream equality or real-completeness data.*

```
↪variant BEDC.Derived.StreamNameUp.RatStreamName reindexing and map stability
```

Proof. Project the accepted StreamName carrier and classifier rows. For reindexing, a requested unary window n is sent to the unary window $r(n)$, so the original carrier and classifier rows apply directly at the transported observation coordinate.

For pointwise mapping, evaluate the original carrier and classifier rows at the same unary window. The carrier-preservation hypothesis for F transports each displayed rational history, and the classifier-preservation hypothesis transports the paired rational-history classifier row.

Finally repack the transported rows into the public StreamName packet. The route exports the same finite StreamName, hsame, Cont, Pkg, and $\mathsf{NameCert}$ fields after these two transports, so downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive a non-quotient mapped-stream handoff rather than an added completion coordinate. □

Theorem 126.298 (StreamName constant observation map composition)**.** *Let x be a certified point row over the L10 source, let r be a unary reindexing map on finite histories, and let F be a pointwise map on rational-history observations. Suppose r sends every unary window to a unary window, F preserves the* $\mathsf{RatHistory}^{\uparrow}$ *carrier, and F preserves the* $\mathsf{RatHistory}^{\uparrow}$ *classifier. Then the pointwise map of the constant observation* $\mathsf{StreamName}^{\uparrow}$ *packet for x factors through the same finite constant schedule after reindexing by r. The exported row contains only the finite* $\mathsf{StreamName}^{\uparrow}$ *carrier and classifier, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *data, and it exports no quotient stream equality, selected real value, or real-completeness row.*

Proof. Apply Definition 126.296 to the certified point row x. This gives the constant observation StreamName packet on every unary finite window, with its classifier, transport, continuation, package, and local naming rows read from the same constant schedule.

Next apply Definition 126.297 to the reindexing map r and the pointwise rational-history map F. The hypotheses on r and F transport the carrier and classifier rows windowwise, while preserving the finite StreamName support.

The constant schedule has no additional coordinate beyond the certified point row and the displayed unary window. Therefore after reindexing, the mapped packet still factors through the same finite constant schedule and repacks only the StreamName carrier, classifier, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows for downstream RegSeqRat and Real consumers. □

Theorem 126.299 (StreamName successor-window projection determinacy)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be certified StreamName observation maps, let $n : \mathsf{BHist}$ be a certified unary observation, and let $B : \mathsf{ProbeBundle}$ be a finite observation bundle such that $\mathsf{BHist.e1}(n)$ is an $\mathsf{InBundle}$ member of B. If the public finite-window classifier*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B)$$

holds, then it determines the projected singleton successor-window classifier

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{ProbeBundle.Bcons}(\mathsf{BHist.e1}(n), \mathsf{ProbeBundle.Bnil})).$$

The projected row is exactly the successor observation row consumed by the scheduled constant observation package of Definition 126.285, by the scheduled window reindex stability row of Definition 126.287, and by the RealUp finite-window boundary rows Definition 38.314 and Definition 38.320. It adds no host Nat*, quotient representative, completion object, metric row, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Unfold the public finite-window interface in Definition 126.301. The classifier over B is a finite family of pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows indexed by the certified unary observations carried by B.

The hypothesis that $\mathsf{BHist.e1}(n)$ is an $\mathsf{InBundle}$ member of B gives the inclusion of the singleton successor bundle into B. Apply the finite-window restriction export of Definition 126.302 to retain exactly the row indexed by $\mathsf{BHist.e1}(n)$ and to discard every other bundled observation.

Now read that singleton row as a scheduled successor observation. The successor reindexing stability row for StreamName observations identifies the same pointwise classifier after reindexing by the successor observation, and Definition 126.287 repacks it as the finite input of the scheduled constant observation package.

Finally pass the projected scheduled row through Definition 126.289 and Definition 126.290. These rows are the RealUp finite-window feed rows already cited by Definition 38.314 and Definition 38.320. The projection therefore changes only the finite ProbeBundle ledger and introduces no host Nat index, quotient representative, completion object, metric row, or $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 126.300 (StreamNameUp public countable-name interface)**.** *The public* $\mathsf{StreamName}^{\uparrow}$ *interface over* BHist *exports exactly the countable rational stream-name data assembled from* UnaryHistory*,* hsame*,* $\mathsf{Rat}^{\uparrow}$*, and the stream carrier/classifier rows. Its carrier is* RatStreamNameCarrier*, its classifier is* RatStreamNameClassifier*, and its public rows are precisely the RatUp pointwise carrier/classifier fields, the constant stream witness, empty constant-stream point exactness, constant point exactness, reindexing and RatUp-preserving map stability, and the composition law for unary observation reindexing.*

Equivalently, a downstream chapter may cite this interface as a single countable-name package whose exported surface consists of Definition 126.13, Definition 126.48, Definition 126.53, Definition 126.55, Definition 126.19, and Definition 126.20. The export does not supply quotient representatives, completion data, RealUp completeness, metric data, or a standard bridge.

Proof. The carrier and classifier fields are the two unfolded stream rows in Definition 126.13. That theorem also supplies the pointwise RatUp equivalence and transport clauses over every certified unary observation in BHist, so the public package does not introduce any classifier beyond RatStreamNameClassifier.

The inhabited constant-name row is Definition 126.48. Empty observation exactness comes from Definition 126.53, and pointwise constant exactness comes from Definition 126.55. These three rows identify the visible constant stream data with the underlying $\mathsf{Rat}^{\uparrow}$ seed without adding any quotient representative or completion object.

The structural action rows are exactly Definition 126.19 and Definition 126.20. The first handles a single unary observation reindexer together with a RatUp-preserving history map, while the second handles composition of such reindexers. Repacking these six named rows gives the public countable-name interface and leaves all completion, metric, RealUp-completeness, and standard-bridge claims outside the export. □

Theorem 126.301 (StreamNameUp public finite-window interface)**.** *The public finite-window surface of* $\mathsf{StreamName}^{\uparrow}$ *over* BHist *consists exactly of* RatStreamNameFiniteWindowClassifier *together with the pointwise carrier/classifier rows, classifier-to-window coverage, singleton-window exhaustion, certified reindexing and point-map coverage, finite-window structural laws, the selected full endpoint package, and finite-window stability fields, together with the empty-window vacuity boundary.*

Equivalently, downstream $\mathsf{Real}^{\uparrow}$ *and* RegSeqRat *boundary theorems may cite this interface as a single finite-window package whose exported rows are Definition 126.300, Definition 126.22, Definition 126.23, Definition 126.24, Definition 126.25, Definition 126.27, Definition 126.28, and Definition 126.310. The export does not supply RealUp completion, RegSeqRat regularity, metric completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier public interface`

Proof. The public countable-name spine is Definition 126.300. It fixes the carrier and classifier as RatStreamNameCarrier and RatStreamNameClassifier, and it exposes the pointwise RatUp carrier/classifier rows over certified unary observations in BHist.

Unfold Definition 126.21. The finite-window classifier is exactly the restriction of the stream-name classifier to bundled certified unary observations. Thus Definition 126.22 projects every public stream-name classifier to an arbitrary probe bundle, while Definition 126.23 uses singleton probe windows to recover the full pointwise classifier.

The action rows are the certified reindexing and point-map coverage theorem Definition 126.24 and the structural laws in Definition 126.25. Bundle membership provides the restriction, append, and recomposition cases, so no additional finite-window carrier or classifier is introduced.

For selected endpoints, apply Definition 126.27 to a chosen bundled unary observation. The result reads back the RatUp point classifier, positive denominator rows, unary endpoint rows, endpoint nonemptiness, zero-headed exclusion, and one-tail agreement package needed by finite boundary consumers.

Finally, Definition 126.28 reuses RatUp point classifier reflexivity, symmetry, transitivity, and hsame transport at each bundled observation. Repacking these named rows gives precisely the finite-window interface consumed by downstream $\mathsf{Real}^{\uparrow}$ and RegSeqRat boundary theorems. The empty bundle boundary is supplied by the closed bundle-generation case for Bnil: no membership constructor can feed an endpoint classifier row. Completion, regularity, metric, and bridge claims are left outside the package. □

Theorem 126.302 (StreamNameUp finite-window restriction export)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let B' and B be* ProbeBundle *windows. Suppose every certified unary observation carried by B' is an* InBundle *member of B. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B) \implies \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B').$$

The restriction export has the ten-field fit used by downstream RegSeqRat *and* $\mathsf{Real}^{\uparrow}$ *consumers: the BEDC source is* $\mathsf{StreamName}^{\uparrow}$ *together with* ProbeBundle*, the carrier consists of bundled certified unary observations, the classifier is the pointwise* $\mathsf{Rat}^{\uparrow}$ *classification row, the ledger retains exactly the observation rows selected by B', and exactness is supplied by bundle inclusion.*

Proof. Unfold Definition 126.21. A finite-window classifier over B is a pointwise classifier row for each certified unary observation carried by B.

Fix an observation carried by B'. The displayed inclusion witness presents that same certified unary observation as an InBundle member of B, so the classifier over B supplies the corresponding pointwise $\mathsf{Rat}^{\uparrow}$ row for s and t.

Repack these rows over all observations of B'. The carrier remains the same bundled certified-unary-observation carrier, the ledger is the retained subfamily of observation rows, and no completion, quotient representative, or regularity row is introduced. This is precisely the finite-window classifier over B'. □

Theorem 126.303 (StreamNameUp finite-window reindexing stability)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, let B and B' be* ProbeBundle *windows, and let r be a certified unary reindexer. Suppose that every certified unary observation carried by B' is sent by r to an* InBundle *member of B. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B) \implies \mathsf{RatStreamNameFiniteWindowClassifier}(s \circ r, t \circ r, B').$$

The restricted window consumes exactly the reindexed pointwise $\mathsf{Rat}^{\uparrow}$ *classifier rows and the finite observation ledger named by* B'*. It introduces no* $\mathsf{Real}^{\uparrow}$ *completion, no* RegSeqRat *regularity row, no quotient representative, and no metric row.*

Proof. Start from the public finite-window interface of Definition 126.301. It makes the finite-window classifier a pointwise $\mathsf{Rat}^{\uparrow}$ classifier over bundled certified unary observations and includes the certified reindexing rows from Definition 126.24.

Fix a certified unary observation carried by B'. The reindexer sends it to an InBundle member of B, so the classifier over B supplies the pointwise $\mathsf{Rat}^{\uparrow}$ row for s and t at the reindexed observation. Transport that row along the certified point map to read the classifier for $s \circ r$ and $t \circ r$ at the original observation.

Now repack these transported rows over all observations of B'. This is the same restriction shape as Definition 126.302, with the inclusion provided by the reindexed-map coverage row rather than by an identity subwindow. The carrier, classifier, and finite ledger remain those of the public finite-window interface.

Nothing in the construction asks for a completion object, a regular sequence condition, a quotient representative, or a metric witness. The resulting classifier is therefore precisely the restricted probe window needed by $\mathsf{Real}^{\uparrow}$ and RegSeqRat boundary consumers. □

Theorem 126.304 (StreamName finite-window reindexing composition)**.** *Let* s, t *be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, let* B*,* B'*, and* B'' *be* ProbeBundle *windows, and let* r *and* q *be certified unary reindexers. Suppose* r *sends every certified unary observation carried by* B'' *to an* InBundle *member of* B'*, and* q *sends every certified unary observation carried by* B' *to an* InBundle *member of* B*. Then restricting the finite* RatStreamNameFiniteWindowClassifier *window by* r *and then by* q *exposes the same finite* BHist *packet as restricting once by* $q \circ r$*. The classifier, endpoint,* hsame*, and finite ledger rows are the same rows repacked over* B''*.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamName reindexing composition law`

Proof. Apply Definition 126.303 to the restriction from B to B'. This transports the finite-window classifier rows along q and keeps the endpoint, hsame, and finite ledger rows as displayed rows on B'.

Apply Definition 126.303 again to the restriction from B' to B''. The second transport reads the rows already repacked over B' and exposes them on B'' along r.

The checked composition law for certified reindexers supplies the same unary observation witness for the composite map $q \circ r$. Repacking the identical finite-window rows over B'' gives the one-step restriction by $q \circ r$ and does not introduce completion, regularity, quotient, metric, arithmetic, or order data. □

Theorem 126.305 (StreamNameUp finite-window union composition)**.** *Let* s, t *be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let* B_L *and* B_R *be* ProbeBundle *windows. Then the two component classifiers assemble into the appended-window classifier:*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_L) \wedge \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_R) \implies \mathsf{RatStrea}$$

Conversely, the appended-window classifier restricts to both component windows:

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{BAppend}(B_L, B_R)) \implies \mathsf{RatStreamNameFiniteWindowClassifier}(s, t,$$

The union row is a finite observation-window composition row only: it introduces no completion, quotient representative, metric row, or $\mathsf{Real}^{\uparrow}$ *completeness object.*

Proof. For assembly, unfold Definition 126.21. Fix a certified unary observation carried by $\mathsf{BAppend}(B_L, B_R)$. The bundle split theorem BEDC.FKernel.Bundle.inBundle_bundleAppend_iff identifies this membership with membership in B_L or membership in B_R.

In the left case, read the needed pointwise $\mathsf{Rat}^{\uparrow}$ classifier row from the classifier over B_L. In the right case, read the same kind of row from the classifier over B_R. Repacking the case split over all bundled observations gives the classifier over $\mathsf{BAppend}(B_L, B_R)$.

For restriction, use the canonical inclusions of B_L and B_R into the appended bundle, again supplied by the InBundle splitting row for BAppend. Applying Definition 126.302 to each inclusion returns the two component classifiers.

All rows remain the same finite ProbeBundle rows and the same pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows. No completion package, quotient representative, metric structure, or $\mathsf{Real}^{\uparrow}$ completeness claim is used in either direction. □

Theorem 126.306 (StreamNameUp finite-window appended refinement chain)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let B_L, B_R, and B_0 be* ProbeBundle *windows. Suppose every certified unary observation carried by B_0 is an* InBundle *member of* $\mathsf{BAppend}(B_L, B_R)$*. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_L) \wedge \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_R) \implies \mathsf{RatStrea}$$

The row uses only ProbeBundle*,* InBundle*,* BAppend*,* UnaryHistory*, and the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows carried by certified unary observations. It introduces no* $\mathsf{Real}^{\uparrow}$ *completion datum, quotient representative, regularity predicate, or metric row.*

Proof. First assemble the classifier over the appended display. Apply Definition 126.305 to the two component classifiers over B_L and B_R. The assembly direction supplies the finite-window classifier over $\mathsf{BAppend}(B_L, B_R)$.

Next restrict that appended classifier to the displayed subwindow. The assumed inclusion sends every certified unary observation of B_0 to an InBundle member of $\mathsf{BAppend}(B_L, B_R)$, so Definition 126.302 applies directly.

The resulting classifier over B_0 is obtained by retaining pointwise $\mathsf{Rat}^{\uparrow}$ rows along the displayed finite bundle membership. No completion object, quotient datum, regularity row, or metric structure enters the construction. □

Theorem 126.307 (StreamNameUp finite-window refinement transitivity)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let B_2, B_1, and B_0 be* ProbeBundle *windows. Suppose every certified unary observation carried by B_0 is an* InBundle *member of B_1, and every certified unary observation carried by B_1 is an* InBundle *member of B_2. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_2) \implies \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_0).$$

The row uses only ProbeBundle*,* InBundle*,* UnaryHistory*, and the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows carried by certified unary observations. It introduces no* $\mathsf{Real}^{\uparrow}$ *completion datum, quotient representative, regularity predicate, or metric row.*

Proof. First restrict the classifier over B_2 to the intermediate finite window B_1. The displayed inclusion from B_1 into B_2 supplies exactly the witness required by Definition 126.302, so the result is $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_1)$.

Next restrict the intermediate classifier over B_1 to B_0. The displayed inclusion from B_0 into B_1 again matches the premise of Definition 126.302.

Equivalently, for any certified unary observation in B_0, compose its InBundle witness into B_1 with the witness from B_1 into B_2, and retain the same pointwise $\mathsf{Rat}^{\uparrow}$ row. This construction never reads a $\mathsf{Real}^{\uparrow}$ completion, a quotient datum, or any host analytic structure. □

Theorem 126.308 (StreamNameUp finite-window common refinement export)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let B_L, B_R, and B be* ProbeBundle *windows. Suppose every certified unary observation carried by B_L is an* InBundle *member of B, and every certified unary observation carried by B_R is an* InBundle *member of B. Then*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B) \quad \Longrightarrow \quad \mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B_L) \wedge \mathsf{RatStream}$$

Thus a displayed common finite observation window restricts to both component windows and assembles the appended window consumed by finite-partition arguments.

Proof. Start with the classifier over the displayed common window B. Apply Definition 126.302 to the inclusion of B_L in B to obtain the left component classifier.

Apply the same restriction export to the inclusion of B_R in B. This gives the right component classifier over the same certified observation maps s, t.

Now apply Definition 126.305 to the two component classifiers. The assembly direction gives the classifier over $\mathsf{BAppend}(B_L, B_R)$.

Every row used in the construction remains a finite-window row: ProbeBundle membership, InBundle inclusions, pointwise $\mathsf{Rat}^{\uparrow}$ classifiers, hsame transport, and finite ledger rows. No completion object, metric row, quotient representative, regularity condition, or $\mathsf{Real}^{\uparrow}$ completeness datum is introduced. □

Theorem 126.309 (StreamNameUp common-refinement selected endpoint readback)**.** *Let s, t be certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps, and let B_L, B_R, B :* ProbeBundle*. Suppose every certified unary observation carried by B_L is an* InBundle *member of B, and every certified unary observation carried by B_R is an* InBundle *member of B. Assume*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, B).$$

Let n : Hist *be a selected certified unary observation in either component window, so that n is carried by B_L or by B_R, and suppose*

$$\mathsf{UnaryHistory}(n), \quad \mathsf{hsame}(s(n), \mathsf{E1}(a)), \quad \mathsf{hsame}(t(n), \mathsf{E1}(b))$$

for histories a, b : Hist*. Then the selected endpoint exposes exactly the six finite readback rows:*

1. $\mathsf{RatHistoryClassifier}(s(n), t(n))$.
2. *positive unary denominator rows for $s(n)$ and $t(n)$.*
3. *unary endpoint-shape rows for $s(n)$ and $t(n)$.*
4. *empty-endpoint exclusions for $s(n)$ and $t(n)$.*
5. *zero-headed endpoint exclusions for $s(n)$ and $t(n)$.*
6. *one-tail agreement for the displayed tails a and b.*

No RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *seal, metric row, quotient representative, or standard bridge is used.*

Proof. First restrict the common classifier over B to the relevant component window. Apply Definition 126.308. If the selected observation is carried by B_L, take the left component classifier; if it is carried by B_R, take the right component classifier.

Now apply Definition 126.27 to that component classifier, the displayed component membership of n, the unary witness for n, and the two one-extension endpoint displays. The theorem returns

the selected RatHistoryClassifier row, positive unary denominators, unary endpoint shape, empty exclusion, zero-headed exclusion, and one-tail agreement for a and b.

The construction has only restricted a finite ProbeBundle classifier and then read the selected $\mathsf{Rat}^{\uparrow}$ endpoint package. It therefore never invokes RegSeqRat regularity, a RealUp seal, metric data, quotient representatives, or standard-bridge structure. □

Theorem 126.310 (StreamNameUp empty finite-window vacuity)**.** *For certified observation maps* $s, t : \mathsf{Hist} \to \mathsf{Hist}$*,* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{Bnil})$ *holds. Equivalently, the public finite-window interface has a neutral empty observation boundary: it contributes no RatUp endpoint classifier rows and no finite-observation ledger token.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier empty vacuity`

Proof. Unfold Definition 126.21. It remains to show that every certified unary observation bundled in Bnil receives a RatUp point classifier.

Fix $n : \mathsf{Hist}$ together with $\mathsf{InBundle}(n, \mathsf{Bnil})$ and $\mathsf{UnaryHistory}(n)$. By Definition 6.1, Bnil has no membership constructors, and the Lean bundle inversion theorem BEDC.FKernel.Bundle.inBundle_nil_elim eliminates the displayed membership witness.

The desired point classifier follows from this contradiction. Hence the empty finite window is neutral: it imposes no endpoint RatUp classifier row and records no finite observation in the ledger. □

The empty-window certificate fit has ten fields: traditional target empty finite observation window; BEDC source $\mathsf{StreamName}^{\uparrow}$ plus ProbeBundle; carrier certified observation maps; classifier finite-window pointwise RatUp classification; stability vacuous; ledger empty; core theorem Definition 126.310; exactness no bundled observations; standard bridge empty finite prefix; status ProofTarget.

The certificate template has ten fields: traditional target certified countable rational-observation names; BEDC source $\mathsf{Rat}^{\uparrow}$ over BHist; carrier RatStreamNameCarrier; classifier RatStreamNameClassifier; stability obligations as the fields of Definition 126.13, the reindexing and mapped stream stability clauses from Definition 126.19, the composed reindexing stability law from Definition 126.20, the constant stream witness from Definition 126.48, constant reindexing classification from Definition 126.50, and constant-point exactness from Definition 126.55 and Definition 126.52; ledger finite observation index plus pointwise rational witnesses, and for a constant stream the hidden constant seed together with the observation index; core theorems Definition 126.13 and Definition 126.19 and Definition 126.20 and Definition 126.48 and Definition 126.50 and Definition 126.55 and Definition 126.52; exactness pointwise over the RatUp classifier; standard bridge ordinary rational streams; status ProofTarget.

Theorem 126.311 (StreamNameUp public interface export package)**.** *The public* $\mathsf{StreamName}^{\uparrow}$ *export over* BHist *is exactly the union of Definition 126.300 and Definition 126.301. Its exported countable-name rows are the carrier* RatStreamNameCarrier*, the classifier* RatStreamNameClassifier*, the pointwise RatUp carrier/classifier fields, the constant stream rows, reindexing and RatUp-preserving map stability, and unary reindexing composition. Its exported finite-window rows are* RatStreamNameFiniteWindowClassifier*, finite-window coverage/exhaustion, certified reindexing and point-map coverage, finite-window structural laws, the selected endpoint package, and finite-window stability fields, together with empty-window vacuity.*

No further rows are part of this public $\mathsf{StreamName}^{\uparrow}$ *package: it excludes RegSeqRat regularity, RealUp completion and completeness, metric completeness, arithmetic and order closure, and standard-bridge data.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNamePublicInterfaceExportPackage`

Proof. Apply Definition 126.300. It fixes the carrier and classifier as RatStreamNameCarrier and RatStreamNameClassifier and lists the pointwise RatUp carrier/classifier rows, constant stream rows, reindexing and RatUp-preserving map stability, and reindexing composition. These are precisely the countable-name rows of the public export.

Apply Definition 126.301. It adds RatStreamNameFiniteWindowClassifier, window restriction, coverage/exhaustion, certified reindexing and point-map coverage, the finite-window structural laws, selected endpoint package, stability rows, and the empty finite-window boundary. These are precisely the finite-window rows of the public export.

The two cited public theorems list all exported rows and both leave completion, regularity, metric, arithmetic/order, and bridge structure outside their claims. Repacking their rows therefore gives a public $\mathsf{StreamName}^{\uparrow}$ interface package, not a $\mathsf{Real}^{\uparrow}$ completion theorem or a RegSeqRat regularity theorem. □

Theorem 126.312 (StreamNameUp public L10 exit handoff)**.** *The public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface supplies exactly the countable-name observation row required by the L10 Real-and-Completion exit bundle. A* $\mathsf{Real}^{\uparrow}$ *consumer may read that row only as the StreamName face of the four-object route: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback must still be displayed before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Apply Definition 126.301 and Definition 126.311. They identify the public finite-window and countable-name rows exported by $\mathsf{StreamName}^{\uparrow}$ and leave dyadic tolerance, regularity, and Real completion outside the StreamName package.

Therefore the L10 exit bundle may use $\mathsf{StreamName}^{\uparrow}$ as its observation row, but the downstream Real read still factors through the separate $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ faces before the $\mathsf{Real}^{\uparrow}$ seal is visible. □

Theorem 126.313 (StreamNameUp public scoped dependency package)**.** *The public* $\mathsf{StreamName}^{\uparrow}$ *export has the following exact scoped dependency surface for downstream* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$ *boundary theorems. It is scoped over precisely* BHist*,* UnaryHistory*,* hsame*,* RatHistoryCarrier*,* RatHistoryClassifier*,* RatStreamNameCarrier*,* RatStreamNameClassifier*,* ProbeBundle*,* InBundle*, and* RatStreamNameFiniteWindowClassifier*.*

Within that scope its exported rows are exactly the countable-name package, finite-window package, empty-window vacuity, constant-stream rows, reindexing and RatUp-preserving map stability, finite-window coverage/exhaustion, selected endpoint package, and finite-window stability fields. No RegSeqRat *regularity,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Real}^{\uparrow}$ *completeness, metric data, quotient representative, or standard bridge is part of this scoped export.*

Proof. Apply Definition 126.300. It fixes the history scope through BHist, UnaryHistory, and hsame, and it exports the pointwise $\mathsf{Rat}^{\uparrow}$ rows through RatHistoryCarrier and RatHistoryClassifier together with RatStreamNameCarrier and RatStreamNameClassifier.

The same countable-name interface supplies the constant stream rows, reindexing rows, RatUp-preserving map stability, and unary reindexing composition. These are the countable-name rows consumed downstream, and they do not introduce quotient representatives, metric data, or completion objects.

Apply Definition 126.301. It restricts the finite side to ProbeBundle, InBundle, and RatStreamNameFiniteWindow then supplies finite-window coverage/exhaustion, certified reindexing and point-map coverage, structural laws, the selected endpoint package, and finite-window stability fields.

Apply Definition 126.310 for the empty bundle boundary. Repacking the three cited sites gives exactly the dependency surface available to RegSeqRat and Real$^{\uparrow}$ boundary theorems: a public stream-name and finite-window package, with no regularity, seal, completeness, metric, quotient, or standard-bridge row added. □

Theorem 126.314 (StreamNameUp ordinary rational sequence bridge)**.** *The public* StreamName$^{\uparrow}$ *rows over* BHist *admit a conservative ordinary-sequence interpretation indexed by certified unary observations. In that interpretation, a stream is in* RatStreamNameCarrier *exactly when every indexed history lies in the pointwise* Rat$^{\uparrow}$ *carrier row, and two streams are related by* RatStreamNameClassifier *exactly when every indexed pair is related by the pointwise* Rat$^{\uparrow}$ *classifier row. The finite-window classifier* RatStreamNameFiniteWindowClassifier *is interpreted by the bundled finite prefix with the same pointwise classifier rows, with the empty prefix covered by the public vacuity law. Constant streams and certified unary reindexing commute with this interpretation, and finite-window restriction is the ordinary prefix restriction of the same indexed rational-history sequence.*

The interpretation is conservative: it adds no quotient, completion, metric, or Real$^{\uparrow}$ *completeness claim beyond the public* StreamName$^{\uparrow}$ *package.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameUp StdBridge`

Proof. Unpack Definition 126.1 and Definition 126.2. Their rows already describe an indexed family of rational histories together with the pointwise Rat$^{\uparrow}$ carrier and classifier obligations, so the ordinary-sequence carrier and classifier are obtained by reading those fields componentwise.

Apply Definition 126.300. It supplies the public carrier/classifier rows, constant streams, unary reindexing, RatUp preserving maps, and reindexing composition. These clauses give exactly the commutation laws for constants and certified unary reindexing under the ordinary-sequence reading.

Apply Definition 126.301. It supplies RatStreamNameFiniteWindowClassifier together with finite-window restriction, coverage/exhaustion, certified reindexing and point-map coverage, selected endpoints, and finite-window stability. The bundled finite prefix is therefore the ordinary prefix of the same indexed rational-history sequence, with the same pointwise classifier rows.

Apply Definition 126.310 for the empty prefix case. Finally apply Definition 126.311 to identify these countable-name and finite-window rows as the full public StreamName$^{\uparrow}$ package. Since that package explicitly excludes RegSeqRat regularity, Real$^{\uparrow}$ completion and completeness, metric completeness, arithmetic/order closure, and extra bridge structure, the interpretation is conservative. □

Theorem 126.315 (StreamNameUp scoped interface package)**.** *The scoped public package for* StreamName$^{\uparrow}$ *consumed by* RegSeqRat *and* Real$^{\uparrow}$ *is the union of Definition 126.300, Definition 126.301, Definition 126.311, and Definition 126.314. It presents the public carrier* RatStreamNameCarrier*, classifier* RatStreamNameClassifier*, finite-window classifier, constant stream rows, reindexing and RatUp-preserving map stability, finite-window stability, empty-window vacuity, and the conservative ordinary rational-sequence reading indexed by certified unary observations.*

This package does not include RegSeqRat regularity, Real$^{\uparrow}$ *completion or completeness, metric completeness, arithmetic or order closure, or machine-checked bridge data.*

Proof. Apply Definition 126.300. It fixes RatStreamNameCarrier and RatStreamNameClassifier and supplies the pointwise RatUp carrier/classifier fields, constant stream rows, reindexing and RatUp-preserving map stability, and unary reindexing composition.

Apply Definition 126.301. It adds RatStreamNameFiniteWindowClassifier, finite-window coverage/exhaustion, certified reindexing and point-map coverage, finite-window structural laws, the

selected endpoint package, finite-window stability fields, and the empty-window boundary. The empty case is the vacuity theorem Definition 126.310.

Apply Definition 126.311 to repack the countable-name and finite-window rows as the public export. Apply Definition 126.314 to read that export conservatively as ordinary rational-history sequences indexed by certified unary observations.

The same four sites state the negative boundary: no RegSeqRat regularity, no $\mathsf{Real}^{\uparrow}$ completion or completeness, no metric completeness, no arithmetic or order closure, and no machine-checked bridge data. Repacking exactly those rows establishes the scoped $\mathsf{StreamName}^{\uparrow}$ interface and no stronger bridge or completion claim. □

Theorem 126.316 (StreamNameUp RealUp finite-window seal boundary)**.** *A public* $\mathsf{StreamName}^{\uparrow}$ *package, together with finite-window classifier rows, constant and reindexing laws, and RatUp endpoint observations, supplies exactly the finite observation interface consumed by later* $\mathsf{Real}^{\uparrow}$ *and* RegSeqRat *seal rows. The interface consists of the* RatStreamNameCarrier *and* RatStreamNameClassifier *rows over* BHist*, the* RatStreamNameFiniteWindowClassifier *rows over certified finite bundles, the empty-window boundary, endpoint readbacks in* $\mathsf{Rat}^{\uparrow}$*, and the constant/reindexing stability rows.*

This boundary supplies no RegSeqRat regularity, RealUp completion, metric completeness, quotient carrier, arithmetic or order closure, or trichotomy.

Proof. Apply Definition 126.301. It provides the finite-window classifier rows, endpoint package, coverage/exhaustion, reindexing stability, and finite-window structural laws.

Apply Definition 126.303 for certified transport of those windows, and apply Definition 126.311 to repack the countable-name and finite-window rows as the public export.

Finally apply Definition 126.315. Its negative boundary is inherited unchanged, so the exported rows can be read by $\mathsf{Real}^{\uparrow}$ and RegSeqRat seal sites only as finite observations, not as regularity, completion, metric, quotient, or trichotomy data. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$**.**

Constructive story: A stream name is generated by a certified observation map from unary BHist indices to rational histories, so every accepted stream exposes, at each unary observation, a $\mathsf{Rat}^{\uparrow}$ carrier row for the selected endpoint. The carrier accepts exactly those observation maps with pointwise rational-history witnesses, and the finite-window surface accepts exactly bundled unary observations whose selected endpoints carry the corresponding pointwise $\mathsf{Rat}^{\uparrow}$ classifier; maps with no unary schedule, no rational endpoint witness, or no bundle membership row are excluded. Two stream names are classified when both endpoint streams are carried and every certified unary observation is related by the pointwise rational classifier, while finite windows restrict that same classifier to a displayed ProbeBundle rather than creating a second equality. The source specification is UnaryHistory plus $\mathsf{Rat}^{\uparrow}$ over BHist, the pattern specification records constant streams, empty-window vacuity, endpoint shape exclusions, one-tail readback, finite-window coverage and exhaustion, reindexing, RatUp-preserving point maps, and reindexing composition, and the ledger keeps the observation index, bundle membership, endpoint carrier, point classifier, and transport witness visible. The five name-certificate obligations come from the pointwise RatUp certificate: constant rational streams inhabit the carrier, reflexivity, symmetry, and transitivity are proved at each unary observation, and pointwise hsame transport repacks classified streams and windows as accepted rows. A stream packet h lies in the signature gap of a $\mathsf{StreamName}^{\uparrow}$ token p when the source-domain accepts its unary observation map, the signature relation pairs it with an answer stream and evidence ledger, and the token-introduction relation names that answer as p; unfolded, every finite observation selected by p reads the same rational endpoint and classifier ledger. The checked ordinary-sequence bridge reads the public countable name and finite windows as the corresponding rational sequence and finite prefix, but the chapter registers $\mathsf{StreamName}^{\uparrow}$ without adding regularity, real completion, metric completeness, arithmetic/order closure, quotient representatives, or any independent sequence carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 126.311, Definition 126.314, and Definition 126.315 bind the mature paper scope: the public countable-name rows, finite-window rows, constant and reindexing laws, empty-window boundary, ordinary rational-sequence bridge, and the RegSeqRat/RealUp consumer interface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.StreamNameUp.StreamNameUp\_StdBridge`

Bridge status: bridgeChecked
Not claimed: RegSeqRat regularity, RealUp completion or completeness, metric completeness, arithmetic or order closure, quotient representatives, or independent sequence carriers.
Upgrade path: no further upgrade – chapter at top of formal axis

126.19 Public obligation surface

Definition 126.317 (StreamNameUp public reindexing obligation)**.** *The public reindexing obligation for* $\mathsf{StreamName}^{\uparrow}$ *is the finite BEDC row package that, for every certified unary observation reindexer and every* ProbeBundle *window, transports the pointwise* $\mathsf{Rat}^{\uparrow}$ *rows of the public countable-name interface through the finite-window interface and records the resulting observation rows in the same* BHist *ledger. The obligation cites Definition 126.300, Definition 126.301, Definition 126.303, and Definition 126.304; it introduces no completion, metric, or quotient row.*

Theorem 126.318 (StreamNameUp public constant obligation)**.** *Every public constant* $\mathsf{StreamName}^{\uparrow}$ *packet is accepted by the countable-name interface and by every finite-window interface obtained from certified unary observations. Its visible rows are exactly the constant* $\mathsf{Rat}^{\uparrow}$ *seed row, the pointwise carrier/classifier rows, the empty-window boundary, and the finite ledger entries selected by the observation bundle.*

Proof. Apply Definition 126.300. It exports the constant stream witness together with the pointwise $\mathsf{Rat}^{\uparrow}$ carrier and classifier rows over certified unary observations.

Next apply Definition 126.301. A finite window only restricts those pointwise rows to the observations named by the bundle, while the empty-window boundary contributes the vacuous case.

Thus the public constant obligation is the same constant seed read through each selected observation row. No regularity witness, metric datum, completion endpoint, or quotient representative is present in the public packet. □

Theorem 126.319 (StreamNameUp public transport obligation)**.** *Let two certified* $\mathsf{StreamName}^{\uparrow}$ *observation maps be related by displayed* hsame *rows, and let the finite-window* Cont *routes and* Pkg *provenance rows be transported componentwise. Then every public finite-window classifier row for the first map transports to a public finite-window classifier row for the second map over the same certified unary observations.*

Proof. Use Definition 126.300 to read the public carrier and classifier rows as pointwise $\mathsf{Rat}^{\uparrow}$ rows over certified unary observations. The supplied hsame rows transport those pointwise rows.

Then use Definition 126.301 to restrict the transported pointwise rows to the selected ProbeBundle window. The transported Cont and Pkg rows preserve the same finite provenance route through the public interface.

The resulting packet is therefore a public finite-window classifier row for the second map. The construction does not add a completion or regularity row; it only repacks the transported public rows. □

Theorem 126.320 (StreamNameUp public finite-ledger obligation)**.** *For every public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet, the exported ledger is exactly the finite family of certified unary observation rows selected by its* ProbeBundle *together with the corresponding pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows. Reindexing, restriction, and composition do not expose any row outside that finite ledger.*

Proof. Start with Definition 126.301. It identifies the finite-window surface as the restriction of the public countable-name interface to the observations carried by the bundle.

For reindexing, apply Definition 126.303. The reindexed window reads pointwise rows only at observations whose images are members of the original bundle.

For composition, apply Definition 126.304. The two-step and one-step reindexing routes repack the same endpoint, classifier, hsame, and finite ledger rows over the target bundle. Hence no hidden observation source is exported. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier public ledger obligation`

Theorem 126.321 (StreamNameUp public RealUp handoff obligation)**.** *The public* $\mathsf{StreamName}^{\uparrow}$ *handoff to downstream* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$ *consumers is the package consisting of the countable-name interface, the finite-window interface, the public reindexing obligation of Definition 126.317, the constant-stream obligation of Definition 126.318, the transport obligation of Definition 126.319, and the finite-ledger obligation of Definition 126.320. The handoff supplies certified unary observation rows and pointwise rational classifier rows only; it does not supply* RegSeqRat *regularity,* $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Proof. The countable-name and finite-window interfaces provide the public carrier, classifier, pointwise rational rows, and bundle restriction rows. The reindexing obligation records how these rows move through certified unary observation maps.

The constant, transport, and finite-ledger obligations give the three public consumer cases: constant names, hsame/Cont transport, and finite-window ledger readback. Each case cites only the public StreamNameUp rows already listed above.

Downstream RegSeqRat and $\mathsf{Real}^{\uparrow}$ chapters may therefore consume a stable stream-name observation surface, but any regular Cauchy condition, completion witness, or standard bridge must be supplied in the downstream chapter rather than by this handoff. □

Theorem 126.322 (StreamNameUp finite-window schedule exhaustion)**.** *Any downstream finite-window request whose rows consist of a public* $\mathsf{StreamName}^{\uparrow}$ *observation map, certified* UnaryHistory *indices, a* ProbeBundle *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *endpoint rows, and* RegSeqRat *or* $\mathsf{Real}^{\uparrow}$ *consumer provenance is exhausted by the public finite-window interface of Definition 126.301, the public reindexing obligation of Definition 126.317, the transport obligation of Definition 126.319, the finite ledger of Definition 126.320, and the handoff boundary of Definition 126.321. The request introduces no additional* RegSeqRat *regularity row,* $\mathsf{Real}^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or observation source beyond that union.*

Proof. Read the carrier and classifier through Definition 126.301. The public observation map and certified UnaryHistory indices determine the displayed pointwise $\mathsf{Rat}^{\uparrow}$ rows, while the ProbeBundle schedule restricts those rows to the requested finite window.

Next apply Definition 126.317. It repacks the same scheduled rows after any certified unary reindexing, so the displayed bundle is a restriction and reindexing of the public countable-name surface rather than a separate request for stream data.

Use Definition 126.319 for the pointwise rational endpoint rows. The supplied hsame, Cont, and Pkg rows transport the endpoint classifiers componentwise and preserve the same finite-window provenance.

Finally combine Definition 126.320 with Definition 126.321. The ledger identifies all exposed rows as scheduled public observation rows and pointwise rational classifiers, and the handoff theorem states that downstream RegSeqRat and $\mathsf{Real}^{\uparrow}$ consumers must supply their own regularity,

completion, quotient, and limit-selection data. Hence the finite-window request is exhausted by the listed public StreamNameUp obligations. □

Theorem 126.323 (StreamNameUp RegSeqRat denominator-source exhaustion)**.** *Let a downstream* RegSeqRat *finite-window consumer receive as input a public* StreamName$^{\uparrow}$ *finite-window request together with dyadic denominator rows. Then the consumer reads exactly the selected* RatHistoryClassifier *rows, certified* UnaryHistory *observation rows, finite* ProbeBundle *ledger,* InBundle *membership rows, displayed* hsame *comparisons,* Cont *routes,* Pkg *provenance, and the* DyadicRatCore$^{\uparrow}$ *denominator-ledger rows already exported by the dyadic real-phase source boundary. It reads no* RegSeqRat *regularity witness, seal classifier, completion datum, quotient representative, hidden limit selector, or host index equality.*

Proof. Start with the public real handoff of Definition 126.321. It exposes the countable-name and finite-window StreamNameUp rows, the certified unary observation surface, and the public pointwise rational classifier rows.

Next use Definition 126.322. The finite request is therefore exhausted by the displayed observation indices, probe bundle, bundle-membership rows, rational endpoint rows, hsame comparisons, continuation routes, and package provenance.

The RegSeqRat-facing handoff of Definition 126.79 identifies the same finite rows as the rows consumed by the downstream regular-sequence interface. The denominator side is supplied separately by Definition 291.18, which contributes only the dyadic denominator ledger exported by the real-phase source boundary.

Combining these two source boundaries gives the displayed row inventory. None of the cited boundaries exports a regularity witness, seal classifier, completion datum, quotient representative, hidden limit selector, or host index equality, so no such datum can be read by the finite-window consumer. □

Theorem 126.324 (StreamNameUp RegSeqRat window union exhaustion)**.** *Let two downstream finite-window requests over a common public* StreamName$^{\uparrow}$ *observation map be displayed by probe bundles B_1 and B_2. Suppose each request is exhausted by Definition 126.323. Then the combined request over the displayed bundle union is exhausted by the public StreamName finite-window interface, the two displayed* InBundle *membership families for B_1 and B_2, the pointwise* Rat$^{\uparrow}$ *classifier rows, the dyadic denominator ledger rows,* hsame*,* Cont*, and* Pkg *provenance. The union request exports no* RegSeqRat *regularity witness,* Real$^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. Start from the common public classifier supplied by Definition 126.301. It fixes the observation map, the certified UnaryHistory observation rows, and the pointwise Rat$^{\uparrow}$ classifier rows before either downstream finite request is chosen.

Apply Definition 126.322 separately to B_1 and to B_2. This gives two finite schedules with their displayed InBundle membership families, hsame comparisons, Cont routes, and Pkg provenance, all over the same public StreamName observation surface.

Next use Definition 126.323 for each scheduled request. The first request contributes only its displayed observation rows and dyadic denominator ledger rows, and the second request contributes the same kind of displayed rows over its own membership family.

It remains to repack the two finite ledgers. For an observation in the displayed bundle union, case on whether the membership row comes from B_1 or from B_2. In the first case read the row from the first exhausted request; in the second case read it from the second exhausted request. If

a row is displayed by both bundles, the common public classifier and the displayed hsame rows identify the two readbacks as the same public StreamName observation row.

Thus every row in the combined finite request is one of the displayed public observation, membership, rational-classifier, denominator-ledger, continuation, or provenance rows already listed. Since neither component exhaustion theorem exports regularity, completion, quotient, limit-selection, or host index equality data, the bundle-union request exports none of those rows. □

Theorem 126.325 (StreamName finite-window terminal schedule idempotence)**.** *Let a public* StreamName$^{\uparrow}$ *observation map be read over certified* UnaryHistory *indices, and let a finite* ProbeBundle *schedule B display the downstream source request. Suppose a terminal downstream* RegSeqRat *or* Real$^{\uparrow}$ *consumer has source rows exactly the scheduled* RatStreamNameFiniteWindowClassifier *rows, the displayed* InBundle *membership rows for B, the pointwise* RatHistoryClassifier *rows, the dyadic denominator rows,* hsame*,* Cont*, and* Pkg *provenance. Then applying the same schedule boundary a second time exposes the same finite source projection.*

The second pass supplies no additional regularity witness, completion datum, quotient representative, hidden limit selector, or host index row.

Proof. Apply Definition 126.322 to the first pass through B. This identifies the public observation packet as the displayed finite schedule: certified UnaryHistory indices, the RatStreamNameFiniteWindowClassifier rows, membership in B, pointwise rational classifier rows, hsame, Cont, and Pkg provenance.

Attach the denominator side by Definition 126.323. The terminal consumer may therefore read the dyadic denominator boundary only as part of the displayed finite source ledger, not as an independent RegSeqRat regularity or Real$^{\uparrow}$ completion row.

Now use Definition 126.324 with $B_1 = B_2 = B$. The displayed bundle union is again B, so the second boundary request reads the same membership family, the same rational classifier rows, the same denominator rows, and the same continuation and provenance rows as the first request.

Finally apply Definition 126.320 and Definition 126.321. These obligations exclude non-displayed rows from the public StreamName handoff, so a repeated boundary cannot create a fresh regularity witness, completion datum, quotient representative, hidden limit selector, or host index row. The terminal source projection is therefore idempotent under the repeated finite schedule. □

Theorem 126.326 (StreamNameUp pointwise hsame reindexing stability)**.** *Let a public* StreamName$^{\uparrow}$ *observation map be read through certified* UnaryHistory *reindexers, and let a finite* ProbeBundle *window B display the source request. Suppose the source and target observation rows are related pointwise by the displayed* hsame *relation, and suppose the reindexing routes are carried by the same* Cont *and* Pkg *provenance rows. Then the public finite-window classifier packet transports to the reindexed window.*

The transported packet exposes only certified unary observation rows, pointwise Rat$^{\uparrow}$ *classifier rows,* InBundle *membership,* hsame*,* Cont*, and* Pkg *provenance. It supplies no* RegSeqRat *regularity,* Real$^{\uparrow}$ *completion, quotient representative, hidden limit selector, or host index equality.*

Proof. Start from the public reindexing obligation of Definition 126.317. It identifies the source and target observation rows through certified UnaryHistory reindexers and keeps the route inside the displayed Cont and Pkg provenance.

Apply Definition 126.319 to the pointwise hsame comparisons. The rational classifier row attached to each source observation is carried to the corresponding target observation without creating a host index equality or a quotient representative.

Project the finite rows by Definition 126.322. The reindexed window therefore has exactly the certified observation rows, InBundle membership rows, pointwise rational rows, continuation routes, and package provenance listed by the public window schedule.

Finally use Definition 126.320 and the union exhaustion of Definition 126.324 to rule out extra consumer rows. Since those ledgers contain no regularity, completion, quotient, limit-selection, or host-equality coordinate, the transported finite-window packet exposes none of that data. □

Theorem 126.327 (StreamNameUp public reindexing composition)**.** *Let r and s be certified* UnaryHistory *reindexers displayed over finite* ProbeBundle *windows for a public* StreamName$^{\uparrow}$ *observation map. Suppose the r leg transports every displayed public observation row by pointwise* hsame *through its recorded* Cont *route and* Pkg *provenance, and suppose the s leg does the same for the reindexed window. Then the composite reindexer transports the same finite-window classifier packet.*

The composite packet consists only of UnaryHistory *observation rows,* InBundle *membership, pointwise* Rat$^{\uparrow}$ *classifier rows,* hsame *comparisons, composed* Cont *provenance, and the displayed* Pkg *rows. It does not export* RegSeqRat *regularity,* Real$^{\uparrow}$ *completion, quotient representatives, selected limits, or host index equality.*

Proof. Apply the public reindexing obligation of Definition 126.317 to the r leg and then to the s leg. Each application reads only certified UnaryHistory observation rows, displayed membership, pointwise rational classifier rows, and the recorded Cont and Pkg provenance for that leg.

Use Definition 126.319 on the pointwise hsame rows supplied by both legs. The first application transports the source classifier row to the intermediate window, and the second transports the intermediate row to the target window. Composing the two displayed Cont routes gives the route used by the composite packet, while the Pkg rows remain exactly the provenance rows displayed by the two legs.

Project the finite target window by Definition 126.322. The projection lists precisely the UnaryHistory observation rows, InBundle membership, pointwise Rat$^{\uparrow}$ classifier rows, hsame comparisons, continuation routes, and package provenance exposed by the composite finite-window schedule.

Finally combine Definition 126.320 with the window-union exhaustion of Definition 126.324. Any row seen by a consumer of the composite window must already appear in one of the two displayed leg ledgers or in their finite union. Since those ledgers carry no regularity, completion, quotient, selected-limit, or host-index coordinate, the composite public packet exports none of those objects. □

Theorem 126.328 (StreamNameUp public RegSeqRat reindexing cover lock)**.** *Let a finite cover of public* StreamName$^{\uparrow}$ *windows feed downstream* RegSeqRat$^{\uparrow}$ *or finite* Real$^{\uparrow}$ *consumers. Suppose each cover member is carried by certified unary reindexing and by displayed* hsame*,* Cont*, and* Pkg *transport. Then the reindexed cover preserves the same union and overlap inventories: unary observation rows,* InBundle *membership, pointwise* Rat$^{\uparrow}$ *classifier rows,* DyadicRatCore$^{\uparrow}$ *denominator rows, and provenance.*

The cover lock exports no RegSeqRat$^{\uparrow}$ *regularity witness,* Real$^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. Begin with the reindexing composition theorem Definition 126.327. It shows that certified unary reindexing carries exactly the displayed public observation rows, membership rows, pointwise rational classifier rows, hsame comparisons, continuation routes, and package provenance.

Use Definition 126.324 to read the finite cover as one public union inventory. On intersections of cover members, apply Definition 126.329; the overlap rows are the common displayed StreamName observations, membership rows, pointwise rational rows, denominator rows, and provenance.

Finally apply Definition 126.323 to each downstream read. It accounts for the dyadic denominator rows consumed by regular-rational and finite real-facing windows, while the preceding union and intersection steps show that reindexing has added no regularity, completion, quotient, hidden selector, or host-equality coordinate. □

Theorem 126.329 (StreamNameUp RegSeqRat window intersection exhaustion)**.** *Let two downstream finite-window requests over a common public* $\mathsf{StreamName}^{\uparrow}$ *observation map be exhausted by Definition 126.323. Suppose the two requests both restrict to a certified common sub-bundle* K*. Then the common restricted request over* K *is exhausted by the public StreamName finite-window interface, the displayed* $\mathsf{InBundle}(-, K)$ *membership rows, the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the dyadic denominator ledger rows,* hsame*,* Cont*, and* Pkg *provenance. The common restricted request exports no* RegSeqRat *regularity witness,* $\mathsf{Real}^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. Start with the two displayed downstream finite-window requests and restrict each one to the certified common sub-bundle K. By Definition 126.322, each restriction has only the public observation rows, $\mathsf{InBundle}(-, K)$ membership rows, pointwise rational classifier rows, hsame comparisons, Cont routes, and Pkg provenance inherited from the common $\mathsf{StreamName}^{\uparrow}$ observation surface.

Apply Definition 126.323 to each restricted side. The two reads may add the displayed dyadic denominator ledger rows, but they add no regularity witness, completion row, quotient representative, hidden limit selector, or host index equality.

It remains to identify the common row inventory. For an observation in K, membership is witnessed on both restricted requests by the displayed $\mathsf{InBundle}(-, K)$ row. Reading the row from either side gives the same public StreamName observation, because the observation map is common and the displayed hsame rows compare the same pointwise $\mathsf{Rat}^{\uparrow}$ classifier data.

Thus the common restricted request contains only the public finite-window rows, common membership rows, pointwise rational rows, denominator-ledger rows, continuation rows, and provenance rows listed in the statement. Since the cited component exhaustions export none of the excluded downstream data, the common subwindow exports none of it. □

Theorem 126.330 (StreamName pointwise classifier finite-window separation)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be certified rational StreamName observation maps, and let their public* NameCert *rows display the* BHist *carrier,* hsame *transport,* Cont *routes, and* Pkg *provenance required by the finite-window interface. If the pointwise StreamName classifier for* s *and* t *does not hold, then there is a certified unary index* n*, a finite* ProbeBundle *window* K *carrying* n*, and a finite-window* Pkg *row over* K *such that the selected endpoint packages* $s(n)$ *and* $t(n)$ *are not identified by the downstream* $\mathsf{RegSeqRat}/\mathsf{Real}^{\uparrow}$ *seal classifier read from that row.*

Conversely, if a certified finite-window package over some K *carries a certified unary index* n *and displays endpoint packages* $s(n)$ *and* $t(n)$ *which the* $\mathsf{RegSeqRat}/\mathsf{Real}^{\uparrow}$ *seal classifier does not identify, then the pointwise StreamName classifier for* s *and* t *fails. The separating package uses only the displayed* BHist *observations,* hsame*,* Cont*,* Pkg*,* InBundle*, pointwise* $\mathsf{Rat}^{\uparrow}$ *rows, and* NameCert *provenance.*

Proof. For the forward direction, restrict the failed pointwise classifier to a certified unary observation at which its displayed endpoint row is not identified. The finite-window interface supplies

a singleton ProbeBundle carrying that observation, and Definition 126.322 identifies the resulting packet as a public BHist window with the corresponding hsame, Cont, and Pkg rows.

Next pass the singleton packet through the common-window machinery. Apply Definition 126.308 to regard the singleton as a certified subwindow of any displayed comparison window, and use Definition 126.309 to read back the selected endpoint packages $s(n)$ and $t(n)$ from the same finite NameCert provenance row.

Finally use Definition 126.329 together with Definition 126.79. The handoff theorem says the downstream RegSeqRat/Real$^{\uparrow}$ read sees exactly the selected StreamName endpoint packages and the displayed denominator ledger; the intersection exhaustion prevents any hidden regularity, completion, quotient, or host-index row from identifying endpoints that the selected package separates.

For the reverse direction, start with the separating certified package over K. The selected InBundle(n, K) row and the UnaryHistory certificate restrict the public StreamName classifier to the endpoint comparison at n.

If the pointwise classifier for s and t held, the finite-window restriction theorem would transport that pointwise row into the same Pkg packet, and the RegSeqRat/RealUp handoff would read an identified seal endpoint there. This contradicts the displayed separating seal row, so the pointwise classifier fails. □

Theorem 126.331 (StreamName RegSeqRat RealUp observation coverage)**.** *Every finite-window handoff from public* StreamName$^{\uparrow}$ *to a downstream* RegSeqRat *consumer or finite-window* Real$^{\uparrow}$ *consumer is covered exactly by certified* UnaryHistory *observation rows,* ProbeBundle *scheduling rows,* InBundle *membership rows, pointwise* Rat$^{\uparrow}$ *classifier rows, displayed* hsame *comparisons,* Cont *routes,* Pkg *provenance, and the* DyadicRatCore$^{\uparrow}$ *denominator ledger exposed at the real-phase source boundary. The handoff exports no* RegSeqRat *regularity witness,* Real$^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. Apply Definition 126.321. It fixes the public observation surface for downstream RegSeqRat and finite-window Real$^{\uparrow}$ consumers, and it restricts the handoff to displayed StreamNameUp rows rather than to host streams.

Apply Definition 126.322. The finite request is then exhausted by the certified UnaryHistory observation rows, the ProbeBundle schedule, InBundle membership, pointwise Rat$^{\uparrow}$ classifier rows, displayed hsame comparisons, continuation routes, and package provenance.

For a RegSeqRat consumer, use Definition 126.323. It identifies the same public observation rows and adds only the dyadic denominator ledger needed by the regular-sequence interface.

Finally, Definition 291.18 supplies the source boundary for that denominator ledger. Thus the public observation rows, the dyadic denominator rows, and the downstream seal boundary form one finite BEDC source inventory. The exclusion clauses follow because none of the four cited boundaries exports regularity witnesses, completion rows, quotient representatives, hidden limit selectors, or host index equality. □

Theorem 126.332 (StreamName public denominator positive source fit)**.** *Every finite-window public* StreamName$^{\uparrow}$ *handoff accepted by Definition 126.331 has its selected denominator source read from the displayed* DyadicRatCore$^{\uparrow}$ *denominator ledger. The selected denominator rows are positive unary denominator rows in the* Rat$^{\uparrow}$ *classifier sense, are stable under the displayed* hsame*,* Cont*, and* Pkg *transport, and are exactly the source consumed by the downstream* RegSeqRat *or finite-window* Real$^{\uparrow}$ *read. The handoff introduces no regularity witness, completion datum, quotient representative, hidden limit selector, or host index equality.*

Proof. Begin with Definition 126.331. It confines the finite-window handoff to certified UnaryHistory observation rows, the finite ProbeBundle and InBundle rows, pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, displayed transport rows, provenance rows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ denominator ledger.

Next apply Definition 126.323. The downstream regular-sequence read can therefore choose its denominator source only from the same displayed denominator ledger named by the public handoff.

Use Definition 291.18 for the dyadic source row. Its denominator boundary supplies positive unary denominator rows in the $\mathsf{Rat}^{\uparrow}$ classifier sense, and the displayed hsame, Cont, and Pkg rows transport those rows without changing the finite source inventory.

Repack the handoff with that denominator row as the consumed source. Since all cited boundaries are finite ledger statements and none exports regularity, completion, quotient, limit-selection, or host equality data, the selected positive denominator source is exactly the visible source available downstream. □

Theorem 126.333 (StreamName RegSeqRat public overlap lock)**.** *Let two public* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoffs feed downstream* RegSeqRat *consumers or finite-window* $\mathsf{Real}^{\uparrow}$ *consumers over the same observation map. If an observation row lies in the certified overlap of the two displayed* ProbeBundle *windows, then both consumers read the same public* $\mathsf{StreamName}^{\uparrow}$ *observation row, the same pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier row, the same dyadic denominator ledger row, and the same displayed* hsame*,* Cont*, and* Pkg *provenance.*

The overlap lock exposes no RegSeqRat *regularity witness, completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. First restrict both handoff requests to the certified common window. Apply Definition 126.329; each consumer read over the overlap is exhausted by the same public StreamName observation row, the same InBundle membership row, and the same pointwise classifier surface.

Next compare the two finite-window ledgers inside the combined display. Apply Definition 126.324. The union ledger has no extra consumer coordinate beyond the rows already exposed by the two windows, so an overlap row is read through one shared public bundle inventory.

Now identify the denominator source. Use Definition 126.323 together with Definition 126.332. The downstream denominator row is the displayed positive $\mathsf{DyadicRatCore}^{\uparrow}$ source row, and the same hsame, Cont, and Pkg rows carry it through both handoffs.

Finally apply Definition 126.331. Its exclusion clauses rule out regularity witnesses, completion rows, quotient representatives, hidden limit selectors, and host index equality. Hence the overlap determines only the visible public rows listed above. □

Theorem 126.334 (StreamNameUp RegSeqRat cover-refinement exhaustion)**.** *Let* $\mathcal{U} = \{B_i\}_{i \in I}$ *be a finite cover family of public* $\mathsf{StreamName}^{\uparrow}$ *windows over a common observation map. Suppose every* B_i *is exhausted by Definition 126.323, and let* $\mathcal{R}$ *be a certified common refinement cover read by a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer. Then the refinement read is exhausted by the public StreamName finite-window interface, the displayed* InBundle *membership rows for the cover members and for* $\mathcal{R}$*, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator ledger,* hsame*,* Cont*, and* Pkg *provenance. The refinement read exports no* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity witness,* $\mathsf{Real}^{\uparrow}$ *completion row, quotient representative, hidden limit selector, or host index equality.*

Proof. Begin with the public finite-window interface. It fixes the common $\mathsf{StreamName}^{\uparrow}$ observation map, the visible observation histories, the pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, and the public transport and provenance rows before the downstream cover refinement is read.

For each cover member B_i, apply Definition 126.323. The member read contributes only its displayed $\mathsf{InBundle}(-, B_i)$ rows, public observation rows, pointwise rational classifier rows, dyadic denominator ledger rows, hsame comparisons, Cont routes, and Pkg provenance.

Next pass through the finite cover by the union and intersection exhaustion theorems. Use Definition 126.324 to combine the member inventories, and use Definition 126.329 on overlap rows so that a refinement point covered by more than one B_i has one common public row inventory.

It remains to repack the certified common refinement. Case on the displayed $\mathsf{InBundle}$ membership row for a refinement element of $\mathcal{R}$. The certification names a cover member B_i containing that element, so the row is read from the exhausted inventory of B_i; on overlaps, the intersection exhaustion identifies the alternative reads by the common public classifier and the displayed hsame rows.

Thus every row consumed by the downstream $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ refinement read is one of the displayed public StreamName window rows, refinement-membership rows, pointwise rational rows, denominator-ledger rows, continuation rows, or provenance rows. Since the member, union, and intersection exhaustions export none of the excluded regularity, completion, quotient, limit-selection, or host-equality data, the common refinement cover exports none of those rows. □

Theorem 126.335 (StreamNameUp RegSeqRat cover-refinement transitivity)**.** *Let $\mathcal{U} = \{B_i\}_{i \in I}$ be a finite cover family of public $\mathsf{StreamName}^{\uparrow}$ windows over a common observation map, and suppose the cover is exhausted by Definition 126.334. Let $\mathcal{R}$ be a certified refinement cover whose downstream $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ read is exhausted by the same public rows, and let $\mathcal{S}$ be a second certified refinement cover refining $\mathcal{R}$. Then every downstream $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ read over $\mathcal{S}$ is exhausted by the original public StreamName finite-window interface, the displayed memberships*

$$\mathsf{InBundle}(-, B_i),$$
$$\mathsf{InBundle}(-, \mathcal{R}),$$
$$\mathsf{InBundle}(-, \mathcal{S}),$$

the pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ denominator ledger, hsame, Cont, and Pkg provenance.

This transitive refinement read exports no $\mathsf{RegSeqRat}^{\uparrow}$ regularity witness, no $\mathsf{Real}^{\uparrow}$ completion row, no quotient representative, no hidden limit selector, and no host index equality.

Proof. First apply Definition 126.334 to the $\mathcal{R}$ cover. The result identifies every read over $\mathcal{R}$ with the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface, the member $\mathsf{InBundle}(-, B_i)$ rows, the refinement-membership rows for $\mathcal{R}$, the pointwise classifier rows, and the displayed denominator, transport, continuation, and provenance ledger.

Next compose the membership route for the second refinement. Use Definition 126.307 to send each $\mathcal{S}$ row through its certified $\mathcal{R}$ row and then through the containing public window B_i. This gives a single public route

$$\mathcal{S} \to \mathcal{R} \to \mathcal{U}$$

for the displayed $\mathsf{InBundle}$ evidence at all three cover levels.

Now transport the endpoint rows along that route. Apply Definition 126.266. The transported endpoint rows remain the same public observation rows, pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, $\mathsf{DyadicRatCore}^{\uparrow}$ denominator rows, hsame comparisons, Cont continuations, and Pkg provenance rows already named by the first refinement read.

Finally compare overlapping reads in the composed cover. Use Definition 126.324 to combine the visible inventories and Definition 126.329 to identify rows that enter through two cover members. These exhaustion results preserve the same exclusion clauses as the first refinement, so the downstream $\mathcal{S}$ read contains no regularity witness, completion row, quotient representative, hidden limit selector, or host index equality. □

Theorem 126.336 (StreamNameUp L10 architectural gate stability surface)**.** *The L10 countable-name gateway of* $\mathsf{StreamName}^{\uparrow}$ *is exactly the union of the public countable-name interface of Definition 126.300, the public finite-window interface of Definition 126.301, the public reindexing obligation of Definition 126.317, the public constant obligation of Definition 126.318, the public transport obligation of Definition 126.319, the public finite-ledger obligation of Definition 126.320, and the downstream* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff boundary of Definition 126.321.*

Consequently every L10 consumer receives only the displayed $\mathsf{StreamName}^{\uparrow}$ *carrier rows, certified unary observation rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, finite* ProbeBundle *windows,* InBundle *membership, componentwise* hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The gateway supplies no* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity witness,* $\mathsf{Real}^{\uparrow}$ *completion row, quotient representative, host* Nat *stream, hidden limit selector, or standard bridge.*

Proof. Unpack the countable-name interface through Definition 126.300. Its carrier and classifier are the displayed RatStreamNameCarrier and RatStreamNameClassifier rows, read pointwise as $\mathsf{Rat}^{\uparrow}$ data over certified unary observations in BHist.

Restrict those rows through the public finite-window interface of Definition 126.301. A ProbeBundle selects only finitely many certified observations, and the finite-window classifier adds only bundle membership, endpoint, stability, and empty-window boundary rows already listed by that interface.

Next apply the reindexing, constant, transport, and finite-ledger obligations: Definition 126.317, Definition 126.318, Definition 126.319, and Definition 126.320. These clauses respectively repack certified unary reindexing, constant point rows, displayed hsame/Cont/Pkg transport, and the finite observation ledger without adding any off-ledger source.

Finally use the downstream handoff boundary Definition 126.321. It is precisely the interface consumed by later $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ chapters, and it states that regularity, completion, quotient, limit-selection, host-stream, and standard-bridge data must be supplied downstream rather than by $\mathsf{StreamName}^{\uparrow}$. Thus the L10 gateway is exactly the listed union and has the stated non-escape boundary. □

126.20 StreamName Finite-Window Regular Seal Non-Escape

Theorem 126.337 (StreamName finite-window regularity Real seal non-escape)**.** *Let W be a finite public* $\mathsf{StreamName}^{\uparrow}$ *window consumed by a* RegSeqRat *regularity row and then by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal read. If the handoff uses the displayed* ProbeBundle *membership rows, certified unary observations, pointwise rational classifier rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows, then the exported seal ledger consists only of the following finite data:*

1. *the displayed StreamName window W and its* InBundle *membership rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows used by the rational endpoint reads;*
3. *the* RegSeqRat *pointwise classifier row accepted over the same certified observations;*
4. *the componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows naming the packet.*

The exported ledger does not contain quotient stream equality, host sequence equality, a selected limit, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First project the public finite-window interface of Definition 126.301. It supplies exactly the displayed StreamName window, bundle membership, certified unary observations, pointwise rational classifier rows, transport rows, continuation routes, package provenance, and local certificate data.

Next apply the pointwise handoff of Definition 126.170. The RegSeqRat consumer receives only the pointwise StreamName rows over the finite bundle, so the regularity row is read as downstream classifier data rather than as a quotient or host sequence equality.

Finally use the seal boundary of Definition 126.142. It factors the finite-window $\mathsf{Real}^{\uparrow}$ read through the same StreamName, RegSeqRat, hsame, Cont, Pkg, and certificate rows.

Repacking these three routes gives one finite ledger. Since none of the cited public routes has a coordinate for selected limits, metric completeness, quotient stream equality, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of that data is exported by the seal. □

Theorem 126.338 (StreamName finite-window Real seal route exhaustion)**.** *Let W be an accepted finite public* $\mathsf{StreamName}^{\uparrow}$ *window whose observations are consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise row and then by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal read. Then every public row visible to the Real seal consumer is exhausted by the StreamName window row,* ProbeBundle *and* InBundle *membership rows, certified* UnaryHistory *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The route has no coordinate for quotient stream equality, host sequence equality, a selected limit, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an off-ledger seal source.*

Proof. Begin with Definition 126.132. It confines the accepted handoff to the displayed finite StreamName window, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ rows, Real seal boundary, transports, continuations, provenance, and local names.

Next apply Definition 126.356. The RegSeqRat readback and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger are read on the same finite pullback surface as the StreamName window and Real seal row.

Use Definition 126.358 to pin the seal-facing consumer to that four-row surface. Then apply Definition 126.359 to project only the listed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Finally combine this exhaustion with Definition 126.337. The non-escape theorem already rules out quotient stream equality, host sequence equality, selected limits, metric completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so the composed route exports no additional seal source. □

Theorem 126.339 (StreamName RegSeqRat Real budgeted common pullback)**.** *Let P and Q be finite observation budgets for two accepted* $\mathsf{StreamName}^{\uparrow}$ *windows consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and then by finite-window* $\mathsf{Real}^{\uparrow}$ *seal reads. Suppose their certified common budget displays the same StreamName window row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary on the overlap. Then both budgeted reads factor through one common finite pullback packet with only the displayed* BHist *observations,* ProbeBundle *and* InBundle *rows, certified unary observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Consequently changing from P to Q on an accepted common window cannot expose quotient streams, host equality, selected limits, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a second Real seal source.*

Proof. First read each budgeted handoff through Definition 126.132. Both sides are already finite StreamName-to-RegSeqRat-to-Real packets with the same kinds of transport, continuation, provenance, and naming rows.

Apply Definition 126.356 on the certified overlap. The hypotheses identify the four visible sibling rows on the overlap: StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal boundary.

Next use Definition 126.358. The two budgeted seal reads therefore reach the same finite seal-facing surface once the four rows agree.

Finally apply Definition 126.359 and Definition 126.338. The common packet contains only the displayed finite BEDC rows, so neither budget can contribute quotient streams, host equality, selected limits, metric completeness, ambient completeness, or an alternate seal source. □

Theorem 126.340 (StreamName Real handoff completion non-escape)**.** *Every accepted finite-window* $\mathsf{Real}^{\uparrow}$ *seal read obtained from a* $\mathsf{StreamName}^{\uparrow}$ *handoff through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *completes at the level of the displayed pullback packet. Its public completion-facing data are exactly the accepted StreamName window, RegSeqRat readback, DyadicRatCore observation ledger, Real seal boundary, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

No completion consumer can read a quotient stream, host sequence equality, selected limit, metric completeness proof, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or independent completed-real carrier from that handoff.*

Proof. Start from the accepted finite-window Real seal read and apply Definition 126.338. This gives the complete list of visible rows available to the seal consumer before any completion-facing interpretation is made.

Now apply Definition 126.339 to any two finite budgets presenting the same accepted common window. The common pullback packet is unchanged by the budget presentation, so a completion consumer has a single finite StreamName-RegSeqRat-DyadicRatCore-Real route to inspect.

Use the seal boundary from Definition 126.142. The Real read repacks exactly the displayed StreamName point rows, accepted RegSeqRat rows, bundle rows, unary observations, continuations, transports, and provenance; the reverse packing does not introduce completion data.

The non-escape conclusion follows from Definition 126.337 and Definition 126.359. Their coordinate lists omit quotient streams, host sequence equality, selected limits, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, and independent completed-real carriers, so those objects cannot be read through the handoff. □

Theorem 126.341 (StreamName RegSeqRat Real tail-window exactness)**.** *Let W be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *tail window consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation readback, and a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then the seal-facing packet has exactly the displayed rows: the StreamName window W,* ProbeBundle *and* InBundle *membership, certified* UnaryHistory *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

There is no extra coordinate for quotient streams, host sequence equality, selected limits, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an independent completed-real carrier.*

Proof. First apply Definition 126.338. It gives the complete list of public rows visible to a Real seal consumer from an accepted finite StreamName window.

Next use Definition 126.339. The tail-window presentation and any budgeted readback presentation factor through the same finite pullback packet when the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal boundary agree.

Finally apply Definition 126.340. Completion-facing use remains at the displayed pullback packet, so the exact packet is the finite row list already named above.

The exclusions follow because each cited route omits quotient streams, host sequence equality, selected limits, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, and independent completed-real carriers. Since the composition only repacks the same finite rows, it cannot create any of those objects. □

Theorem 126.342 (StreamName finite-window regular seal budget functoriality)**.** *Let $P \leq Q \leq R$ be accepted finite observation budgets presenting the same finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same displayed support rows:* ProbeBundle *and* InBundle *membership, certified* UnaryHistory *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. Then the seal-facing packet obtained by refining from P to Q and then from Q to R is the same finite public packet as the direct refinement from P to R.*

The common packet exports no quotient stream, selected limit, host sequence equality, ambient completeness, metric-completeness datum, countable-choice row, standard bridge row, or off-ledger $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 126.339 to the accepted overlaps $P \leq Q$ and $Q \leq R$. The hypotheses identify the four visible sibling rows on each overlap: the StreamName window, the RegSeqRat readback row, the DyadicRatCore ledger, and the Real seal boundary.

Next apply Definition 126.341. Once the four sibling rows and their displayed support rows agree, the seal-facing packet is exactly the finite row list named in the statement, so the two-step refinement has no separate coordinate left to remember the intermediate budget Q.

Finally use Definition 126.338. The direct refinement from P to R exhausts the same public rows and has no slot for quotient streams, selected limits, host sequence equality, ambient completeness, metric-completeness data, countable-choice rows, standard bridge rows, or off-ledger Real seal sources. Repacking the two certified overlaps therefore gives the same finite public packet as the direct refinement. □

Theorem 126.343 (StreamName finite-window regularity selector nonchoice)**.** *Let W be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed through* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation readback, and a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row visible at the seal is only a displayed downstream budget row in the finite packet. It cannot be re-entered as a hidden selector, host function, countable-choice witness, quotient stream representative, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate, or independent completed-real carrier.*

Equivalently, every seal-facing use of that regularity row factors through the public StreamName window, bundle and membership rows, certified unary observations, pointwise hsame *comparisons,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal boundary, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Proof. Start with the public finite-window interface of Definition 126.301. It exposes the accepted StreamName window as finite BEDC data: the displayed carrier row, bundle membership, certified unary observations, transport rows, continuation routes, package provenance, and local certificate names.

Next use Definition 126.170. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives the same finite pointwise rows, so regularity is read downstream as classifier evidence over the displayed observations rather than as a host stream, selector function, or representative of a quotient stream.

Now apply Definition 126.338. The finite-window $\mathsf{Real}^{\uparrow}$ seal has no public coordinate beyond the listed StreamName, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, seal-boundary, transport, continuation, provenance, and certificate rows.

Finally compose this exhaustion with Definition 126.341. Tail-window exactness says the seal-facing packet is exactly that finite row list. Since the composed packet contains no slot for a selector, host function, countable-choice witness, quotient representative, selected limit, ambient completeness coordinate, or independent completed-real carrier, the regularity row cannot re-enter as any of those data. □

Theorem 126.344 (StreamName Real terminal support minimality)**.** *Let W be an accepted finite terminal* $\mathsf{StreamName}^{\uparrow}$ *window consumed through a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then every support row required to present the same seal-visible source is one of the displayed* BHist *observations,* ProbeBundle *rows,* InBundle *membership rows, certified* UnaryHistory *observations, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows.*

Deleting any one row from this displayed support prevents the same $\mathsf{Real}^{\uparrow}$ *seal-visible source from being presented by the finite StreamName-to-RegSeqRat-to-DyadicRatCore-to-Real route. No quotient stream equality, host sequence equality, selected limit, countable-choice row, metric completeness proof, ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate, standard bridge row, or replacement regularity selector can serve as a substitute support coordinate.*

Proof. First apply Definition 126.338. The seal consumer sees only the StreamName window row, the bundle and membership rows, the certified unary observations, the RegSeqRat readback row, the DyadicRatCore ledger, the Real seal boundary, componentwise transports, continuations, provenance, and local certificate rows.

Next apply Definition 126.341. It identifies the seal-facing packet with exactly that finite row list. Thus presenting the same seal-visible source requires each displayed support row that the packet consumes; removing one row breaks the finite packet rather than moving the obligation to an unlisted coordinate.

Finally use Definition 126.343. The regularity row cannot be re-entered as a hidden selector, host function, countable-choice witness, quotient representative, selected limit, ambient completeness coordinate, or independent completed-real carrier. Combining this with the exhaustion clauses also excludes metric completeness, standard bridge rows, host sequence equality, quotient stream equality, and off-ledger seal sources as substitutes.

The remaining presentation is therefore support-row exact: the same $\mathsf{Real}^{\uparrow}$ seal-visible source is available precisely through the displayed finite BEDC support rows, and deletion of any required displayed row leaves no admissible replacement coordinate inside the StreamName-to-RegSeqRat-to-DyadicRatCore-to-Real route. □

Theorem 126.345 (StreamName finite-window regular seal terminal route)**.** *Let W be an accepted finite terminal* $\mathsf{StreamName}^{\uparrow}$ *window consumed through* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation readback, and a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then the terminal public route consumed by the* $\mathsf{Real}^{\uparrow}$ *seal is the four-face packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist *observations,* ProbeBundle *rows,* InBundle *membership rows, certified* UnaryHistory *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *support rows.*

This route is invariant under accepted budget refinement. It has no coordinate for quotient stream equality, host sequence equality, a selected limit, countable-choice row, metric-completeness datum, standard bridge row, ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate, hidden regularity selector, off-ledger seal source, or independent completed-real carrier.*

Proof. First apply Definition 126.338. It supplies the full row inventory available to the finite-window $\mathsf{Real}^{\uparrow}$ seal consumer: the StreamName window row, bundle and membership rows, unary observations, RegSeqRat readback, DyadicRatCore ledger, Real seal boundary, transports, continuations, provenance, and local certificates.

Next apply Definition 126.341. This identifies the seal-facing inventory with the four visible faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ plus exactly the displayed support rows, so the terminal route is a positive packet rather than only an exclusion list.

The refinement clause follows from Definition 126.342. Accepted refinements compose through the same finite public packet, and the direct refinement sees no additional coordinate.

Finally use Definition 126.343 and Definition 126.344. The selector theorem rules out re-entry through hidden choice, quotient, host, limit, completion, bridge, or off-ledger seal data, while support minimality says the displayed local support rows are the rows actually consumed by the terminal packet. Combining these clauses gives the stated terminal route and its exclusions. □

126.21 StreamName Lattice Dependency

The finite-window $\mathsf{StreamName}^{\uparrow}$ package is consumed by the $\mathsf{RegularCauchyName}^{\uparrow}$ certificate of chapter 371. Its public rows supply the scheduled window, bundle membership, certified unary observations, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier data, transport routes, package provenance, and local NameCert rows that the regular Cauchy name packet reads before exposing a real-name seal.

This edge is the L10 scheduled finite-window handoff from $\mathsf{StreamName}^{\uparrow}$ into the $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ chain. The handoff is recorded before the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows, so the later consumer can cite the StreamName window surface without importing quotient stream equality, host sequence equality, selected limits, or ambient completeness.

Theorem 126.346 (StreamName RatUp to RegSeqRat lattice source)**.** *Let W be an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *row and then by an accepted* $\mathsf{Real}^{\uparrow}$ *seal row. Then every gateway read of W first exposes its point observations as* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier rows: the carrier rows are the pointwise* RatHistoryCarrier *witnesses of Definition 126.1, and the classifier rows are the pointwise* RatHistoryClassifier *witnesses of Definition 126.2. The* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer may read those same point rows only through the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *certificate surfaces of chapter 37, chapter 301, and chapter 38.*

Consequently the downstream lattice edge is sourced by RatUp point rows before any regular-sequence or real-seal row is exposed. The edge exports no quotient stream equality, host sequence equality, selected limit, host real equality, or ambient completeness datum.

Proof. Project the accepted StreamName carrier first. Definitions Definition 126.1 and Definition 126.2 say exactly that a StreamName window reads certified unary observations pointwise as RatHistoryCarrier and RatHistoryClassifier rows. Those are the RatUp carrier and classifier rows named by chapter 37.

Next apply the pointwise handoff theorems in this StreamName chapter. The package theorem Definition 126.170 sends the finite window to $\mathsf{RegSeqRat}^{\uparrow}$ only through the StreamName carrier rows, the finite-window classifier row, certified unary observations, membership rows, and pointwise hsame comparisons. The exhaustion theorem Definition 126.100 identifies those rows with the same pointwise RatUp observations before any regularity or seal-facing datum is read.

Now pass through the displayed downstream certificates. The $\mathsf{RegSeqRat}^{\uparrow}$ row is therefore a consumer of the same point rows through chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal row reads the resulting handoff through chapter 38. The carrier source remains the RatUp point surface; neither consumer introduces a second source for the finite window.

Finally apply the L10 gateway factorization of Definition 126.347. Its route sends the already exposed StreamName point rows through the $\mathsf{RegSeqRat}^{\uparrow}$ regular source row, dyadic ledger, finite tail witness, finite approximation row, and $\mathsf{Real}^{\uparrow}$ seal row. Its boundary clause excludes quotient stream equality, host sequence equality, selected limits, host real equality, and ambient completeness from the gateway source. □

Theorem 126.347 (StreamName L10 gateway factorization)**.** *Let W be an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by accepted* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then every accepted L10 read of W factors through one finite displayed route:*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window rows} \longrightarrow \textit{unary observations} \longrightarrow \mathsf{ProbeBundle/InBundle}\ \textit{ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{regular sour}$$
$$\longrightarrow \textit{dyadic observation ledger} \longrightarrow \textit{finite tail witness} \longrightarrow \textit{finite approximation error row} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal row.}$$

The route is mediated only by the displayed hsame *transports,* Cont *continuations,* Pkg *provenance, and local* NameCert *rows of the consumed packets. It exports no quotient stream equality, selected limit, host real equality, or ambient completeness principle.*

Proof. First build the shared finite-window spine. The StreamName window rows, unary observations, and ProbeBundle/InBundle ledger are the public surface of the accepted $\mathsf{StreamName}^{\uparrow}$ packet, and the $\mathsf{RegSeqRat}^{\uparrow}$ row reads that same surface through the regular source coordinate.

Next compose the sibling handoffs. The finite-window diagonal fusion of Definition 126.348 identifies the common route through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. The real regular-sequence handoff of Definition 126.79 fixes the same window before the real-name seal is exposed.

The remaining coordinates are supplied by the cited consumer chapters. The $\mathsf{RegularCauchyName}^{\uparrow}$ chapter reads the regular source row, the $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ chapter supplies the finite tail witness, and the $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ chapter supplies the dyadic observation ledger and finite approximation error row. The $\mathsf{Real}^{\uparrow}$ chapter receives only the resulting seal-facing finite route.

Coordinate exhaustion gives the boundary clause. None of the displayed rows is a quotient stream, a limit selector, a host real equality witness, or an ambient completeness object, and the transport and continuation rows only replay the listed finite coordinates. □

Theorem 126.348 (StreamName finite-window diagonal fusion)**.** *Let W be an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed simultaneously by an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *row, an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *packet, and an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Then the three consumers determine one shared finite observation route into the* $\mathsf{Real}^{\uparrow}$ *seal: the window rows of W, the regular source row, the dyadic ledger, the selected tail witness, and the finite approximation error row agree through the displayed* hsame *transports and the displayed* Cont *routes.*

Equivalently, every $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through the common route*

$$\mathsf{StreamName}^{\uparrow} \xrightarrow{\mathsf{hsame},\mathsf{Cont}} \mathsf{RegSeqRat}^{\uparrow} \xrightarrow{\mathsf{hsame},\mathsf{Cont}} \mathsf{FiniteTailDiagonalSeal}^{\uparrow} \xrightarrow{\mathsf{hsame},\mathsf{Cont}} \mathsf{FiniteCauchyApproximation}^{\uparrow} \xrightarrow{\mathsf{hsam}}$$

The route exports only finite window rows, the accepted regular source row, dyadic observation ledgers, the selected tail witness, finite approximation error data, componentwise transport, continuation replay, package provenance, and local naming. It does not assert quotient stream equality, selected limits, host real equality, or ambient completeness.

Proof. Start from the accepted $\mathsf{StreamName}^{\uparrow}$ finite-window packet. Its public rows supply the scheduled finite window, bundle membership, unary observations, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier data, componentwise hsame transports, and displayed Cont routes.

The $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads those rows through chapter 371. This fixes the regular source row on the same finite window before any real-seal row is exposed.

Next use chapter 559. The accepted $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ packet reads the same finite window, the same regular source row, and the selected tail witness before its $\mathsf{Real}^{\uparrow}$ seal handoff.

Finally use chapter 601. The accepted $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ packet reads the same source and window together with the dyadic observation ledger and finite error row. Its handoff therefore meets the finite-tail route on the displayed hsame and Cont rows.

Coordinate exhaustion gives the exclusion clause. None of the three accepted consumers contains a coordinate for quotient stream equality, a selected limit, host real equality, or ambient completeness, so the fused route cannot export any of those endpoints. □

Theorem 126.349 (StreamName dyadic RegSeqRat Real pullback exactness)**.** *Let W be an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by accepted* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then the pullback source of the accepted read is the same whether the consumer reads through the StreamName point rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular source row, or the* $\mathsf{Real}^{\uparrow}$ *seal row.*

Equivalently, the four visible routes agree through the displayed BHist *observations,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows of chapter 291, chapter 301, and chapter 38. The common pullback exports no quotient stream equality, selected limit, host real equality, or ambient completeness principle.*

Proof. First identify the source rows. Apply Definition 126.346 to read the accepted StreamName finite window as pointwise carrier and classifier rows before the regular-sequence or real-seal consumer is exposed.

Next factor the L10 route. By Definition 126.347, every accepted L10 read passes through the StreamName window rows, unary observations, bundle ledger, $\mathsf{RegSeqRat}^{\uparrow}$ regular source row, dyadic observation ledger, finite tail witness, finite approximation row, and $\mathsf{Real}^{\uparrow}$ seal row.

Now fuse the finite windows. Apply Definition 126.348 to identify the shared route into the $\mathsf{Real}^{\uparrow}$ seal through the displayed hsame transports and Cont routes.

Project the four sibling rows. The StreamName point rows, dyadic observation ledger, RegSeqRat regular source row, and RealUp seal row all read the same displayed window, transport, route, provenance, and local naming coordinates. Coordinate exhaustion then gives the pullback exactness claim and excludes quotient stream equality, selected limits, host real equality, and ambient completeness from the common source. □

Theorem 126.350 (StreamName budgeted L10 pullback exactness)**.** *Let W be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *packet consumed through accepted* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$,

and $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *rows. Then every consumer-visible route out of* W *factors through one displayed finite packet consisting only of* BHist *observations,* $\mathsf{ProbeBundle}$ *rows,* $\mathsf{InBundle}$ *rows, certified unary observations, dyadic ledger rows, regular-source rows, real-seal rows, budget-limiter request, cap, and window rows, tail-budget service rows, displayed* hsame *transports, displayed* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows.*

The common packet is the pullback source for the StreamName point route, the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular route, the* $\mathsf{Real}^{\uparrow}$ *seal route, the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *service route, and the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *tail-service route. It exports no quotient stream equality, host sequence equality, selected limit, countable-choice selector, host real equality, ambient completeness principle, or alternate seal source.*

Proof. Start with the unbudgeted L10 pullback. Theorem 126.349 identifies one common source for the StreamName point rows, the dyadic ledger, the regular source row, and the real-seal row through the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ coordinates.

Next attach the finite budget route. Theorem 126.391 sends every accepted StreamName budget handoff through the sibling $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ and $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ chapter surfaces before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal rows are consumed.

Now use the budget chapters to fix the visible service rows. Theorem 903.23 contributes the request, cap, and window service triad, and Theorem 906.3 orders the regular source, tail threshold, observation limiter, finite StreamName window, dyadic ledger, readback, and seal as one finite service route.

The displayed rows therefore form a single pullback packet for all six routes named in the statement. Coordinate exhaustion gives the boundary clause: none of the participating packets has a coordinate for quotient stream equality, host sequence equality, selected limits, countable-choice selection, host real equality, ambient completeness, or a second seal source, so none can be exported by the composed route. □

Theorem 126.351 (StreamName L10 four-step source lattice)**.** *Every accepted Phase Real and Completion consumer of a finite* $\mathsf{StreamName}^{\uparrow}$ *packet factors through exactly the ordered source faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The factorization uses only the displayed BHist *observations,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows supplied by the four packet surfaces of chapter 291, this* $\mathsf{StreamName}^{\uparrow}$ *chapter, chapter 301, and chapter 38. It exports no quotient stream equality, host sequence equality, selected limit, countable-choice selector, host real equality, ambient completeness, arithmetic package, order package, or metric-completeness row.*

Proof. First read the unbudgeted four-face pullback. Theorem 126.349 identifies the common source of the StreamName point route, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ regular source row, and the $\mathsf{Real}^{\uparrow}$ seal row through the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ coordinates.

Next impose the accepted L10 consumer boundary. Theorem 126.350 says that the budgeted Phase Real route and the Completion-facing tail route still read the same finite packet before any seal-facing row is consumed. These added service rows do not introduce a second source for the StreamName packet.

The named source faces are then fixed by the local certificates. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ NameCert chapters provide the only displayed rows available at the dyadic ledger, regular-source, and real-seal faces, and the current $\mathsf{StreamName}^{\uparrow}$ chapter provides the finite-window face between them.

Coordinate exhaustion gives both exact ordering and the exclusion clause. The participating packets contain no coordinate for quotient stream equality, host sequence equality, selected limits, countable-choice selection, host real equality, ambient completeness, arithmetic packages, order packages, or metric-completeness rows; hence every accepted Phase Real and Completion consumer factors through exactly the four listed faces and exports none of those objects. □

126.22 StreamName RegularCauchyName Lattice Link

Theorem 126.352 (StreamName scheduled-window RegularCauchyName link)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule consumed by* $\mathsf{RegularCauchyName}^{\uparrow}$ *is read as the scheduled-window sibling route of chapter 371. The route carries the displayed window rows, certified unary observations, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, transport data, continuation data, package provenance, and local naming rows into the regular Cauchy name packet. It exports no host sequence, quotient-stream equality, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. Start with the finite-window rows already displayed in the $\mathsf{StreamName}^{\uparrow}$ packet. The public interface and lattice dependency rows expose scheduled windows, bundle membership, certified unary observations, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier data, componentwise hsame transport, displayed Cont routes, package provenance, and local naming.

The $\mathsf{RegularCauchyName}^{\uparrow}$ handoff of chapter 371 consumes exactly that scheduled-window surface before exposing its dyadic-radius row, regularity ledger, and $\mathsf{Real}^{\uparrow}$ seal. Hence the StreamName schedule is not a free-floating kernel row; it is the window sibling route through which the regular Cauchy name reads its source observations.

The non-escape clause is coordinate exhaustion. Neither the StreamName finite-window package nor the RegularCauchyName handoff contains a host sequence, quotient-stream equality, selected limit, independent real carrier, or ambient completeness coordinate, so the routed lattice link cannot export one. □

Theorem 126.353 (StreamName real-seal lattice route)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule consumed by* $\mathsf{RegularCauchyName}^{\uparrow}$ *reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window rows} \longrightarrow \mathsf{RegularCauchyName}^{\uparrow}\ \textit{schedule, radius, ledger, and seal rows} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{regul}$$

The $\mathsf{RegularCauchyName}^{\uparrow}$ *segment is the schedule-and-radius packet of chapter 371, the* $\mathsf{RegSeqRat}^{\uparrow}$ *segment is the regular source handoff of chapter 301, and the terminal seal is the real-facing packet of chapter 38. The route exports no quotient stream equality, selected limit, host sequence equality, host real equality, or ambient completeness coordinate.*

Proof. Project the StreamName-to-RegularCauchyName lattice edge of Definition 126.352. It identifies the accepted finite-window schedule as the only surface through which $\mathsf{RegularCauchyName}^{\uparrow}$ reads the source observations, together with the displayed certified unary observations, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier rows, transport rows, continuation rows, package provenance, and local naming data.

The $\mathsf{RegularCauchyName}^{\uparrow}$ packet then exposes the dyadic-radius row, regularity ledger, and real-seal-facing row named in chapter 371. The regular source information is the same finite-window $\mathsf{RegSeqRat}^{\uparrow}$ handoff recorded in chapter 301, so the route cannot bypass the regular rational source rows before reaching the real-facing seal.

Finally, the terminal consumer is the $\mathsf{Real}^{\uparrow}$ seal row of chapter 38. Coordinate exhaustion now gives the non-escape clause: none of the displayed packets contains a quotient stream equality, selected limit, host sequence equality, host real equality, or ambient completeness coordinate, and therefore no such coordinate is exported by their composite route. □

Theorem 126.354 (StreamName pointwise reindexing RegularCauchyName route)**.** *Let a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule be reindexed pointwise by unary observation rows and transported componentwise by displayed* hsame *rows, as in Definition 126.3 and Definition 126.8. Then every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *consumer reads the reindexed schedule only through the scheduled-window route of Definition 126.352, and the route to the* $\mathsf{Real}^{\uparrow}$ *seal is exactly the route of Definition 126.353. The displayed rows are precisely the* BHist *observation rows,* UnaryHistory *certificates, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. The consumer exports no quotient stream equality, host sequence equality, selected limit, host real equality, or ambient completeness coordinate.*

Proof. Start with the public finite-window $\mathsf{StreamName}^{\uparrow}$ schedule and its unary reindexing row. The pointwise carrier transport theorem Definition 126.3 carries each certified unary observation along its displayed hsame row, while Definition 126.8 repacks the transported observations as the finite-window classifier surface read by the reindexed schedule.

Now apply Definition 126.352. It forces every accepted $\mathsf{RegularCauchyName}^{\uparrow}$ consumer to read the transported schedule through the same scheduled-window sibling route, so the only available data are the BHist rows, UnaryHistory certificates, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier rows, displayed hsame transports, Cont routes, Pkg provenance, and local NameCert rows.

Finally compose with Definition 126.353. The seal read reaches $\mathsf{Real}^{\uparrow}$ only after the transported StreamName window has passed through the RegularCauchyName schedule, radius, ledger, and seal rows. Coordinate exhaustion gives the non-escape clause: none of these displayed rows contains quotient stream equality, host sequence equality, a selected limit, host real equality, or ambient completeness data, so the reindexing route cannot export any such coordinate. □

126.23 StreamName RegSeqRat Real Lattice Dependency

The $\mathsf{StreamName}^{\uparrow}$ finite-window packet is not an isolated kernel float. Its consumer route is the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal surface of chapter 38; the lattice edge is carried through the child route below rather than by the hub file.

Theorem 126.355 (StreamName RegSeqRat Real lattice dependency)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and then by* $\mathsf{Real}^{\uparrow}$ *factors through the same visible sibling route. The route begins with the StreamName finite-window rows, passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and ends at the* $\mathsf{Real}^{\uparrow}$ *seal rows of chapter 38. Along the entire route the only shared carrier coordinates are the displayed* BHist *observations, the componentwise* hsame *transports, the finite* Cont *continuation rows, the* Pkg *provenance rows, and the local* NameCert *rows naming the consumed packets.*

The dependency exports no quotient stream, no host real equality, no selected limit, and no ambient completeness principle. In particular, the $\mathsf{Real}^{\uparrow}$ *seal is read only as the downstream seal surface reached through the accepted finite StreamName and RegSeqRat packets.*

Proof. Start with the StreamName finite-window surface. By Definition 126.132, the handoff exposes the finite observation window, the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ rows, the $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Next read the RegSeqRat sibling dependency. The pointwise handoff of Definition 126.170 identifies the rows sent to $\mathsf{RegSeqRat}^{\uparrow}$ with the same StreamName carrier rows, finite-window classifier rows, certified unary observations, membership rows, and pointwise hsame comparisons. Thus the RegSeqRat readback surface of chapter 301 is not a second carrier; it is the finite StreamName window read through the accepted pointwise packet.

Finally pass to the Real seal boundary. The factorization theorem Definition 126.142 says that a finite-window $\mathsf{Real}^{\uparrow}$ seal read arising from the StreamName-to-RegSeqRat handoff factors through the StreamName point rows, accepted $\mathsf{RegSeqRat}^{\uparrow}$ seal rows, bundle and membership rows, unary observations, finite Cont ledger, displayed hsame transports, and Pkg provenance. This is exactly the seal-facing surface cited in chapter 38.

Coordinate exhaustion gives the exclusion clause. None of the cited finite-window, pointwise, or seal-factorization rows contains a quotient stream, host real equality witness, selected limit, or ambient completeness coordinate, so composing them cannot export one. □

Theorem 126.356 (StreamName RegSeqRat Real dyadic-window pullback)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *factors through the same finite pullback surface with four visible sibling rows: the StreamName finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38. The pullback coordinates are only the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* $\mathsf{NameCert}$ *rows already visible on those four sibling surfaces.*

Consequently a downstream consumer of the packet may read the dyadic-window ledger as a common finite route from StreamName windows through RegSeqRat readback into the Real seal, but it obtains no quotient stream, no host real equality, no selected limit, and no ambient completeness principle.

Proof. First build the window surface. By Definition 126.132, the accepted finite-window handoff exposes the StreamName observation window together with the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ rows, the $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, finite Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Next read the regular-rational side. The pointwise handoff theorem Definition 126.170 identifies the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows with the same StreamName carrier rows, finite-window classifier rows, certified unary observations, membership rows, and pointwise hsame comparisons. These are precisely the RegSeqRat rows named by chapter 301.

Now attach the dyadic ledger. The finite-window packet already exposes only visible observation rows before the RegSeqRat readback is consumed, and the dyadic observation surface of chapter 291 supplies the endpoint and radius ledger through the same displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates. Thus the StreamName window, RegSeqRat readback, and dyadic ledger occupy one finite pullback surface rather than three independent carriers.

Finally pass to the Real seal. The boundary factorization of Definition 126.142 says that the $\mathsf{Real}^{\uparrow}$ seal read arising from the StreamName-to-RegSeqRat handoff factors through the StreamName point rows, accepted $\mathsf{RegSeqRat}^{\uparrow}$ seal rows, bundle and membership rows, unary observations, finite Cont ledger, displayed hsame transports, and Pkg provenance. This is the Real-facing row cited in chapter 38.

Coordinate exhaustion proves the non-escape clause. The four sibling rows just composed contain no coordinate for a quotient stream, host real equality witness, selected limit, or ambient completeness operation, so the pullback route cannot export any of those objects to its downstream consumer. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRegseqratRealDyadicWindowPullback`

Theorem 126.357 (StreamName RegSeqRat Real window source triangle)**.** *For every accepted finite* StreamName$^{\uparrow}$ *window consumed by a* RegSeqRat$^{\uparrow}$ *row and then by a* Real$^{\uparrow}$ *seal row, the StreamName window row, the* RegSeqRat$^{\uparrow}$ *readback row, the* DyadicRatCore$^{\uparrow}$ *endpoint ledger, and the* Real$^{\uparrow}$ *seal boundary form one finite pullback triangle of sources. Every route among the StreamName, RegSeqRat, and Real-facing surfaces factors through the same* BHist *packet,* ProbeBundle *and* InBundle *membership rows,* UnaryHistory *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The triangle exports no quotient stream equality, no host sequence equality, no selected limit, no metric-completeness witness, and no ambient Real$^{\uparrow}$ *completeness principle.*

Proof. First expose the point source. By Definition 126.346, the accepted window is read through the StreamName row together with the rational and RegSeqRat point rows, bundle membership, unary observations, transport, continuation, provenance, and local naming coordinates.

Next use Definition 126.356. It identifies the shared finite dyadic window on which the StreamName window row, RegSeqRat readback row, DyadicRatCore endpoint ledger, and Real seal boundary are all projected. Thus a route from any one of the three surfaces to another must pass through that same finite packet.

Finally apply the local non-escape clauses in this StreamName lattice section. The cited pullback and consumer-exhaustion rows list all available coordinates, and none is a quotient stream equality, host sequence equality, selected limit, metric-completeness witness, or ambient RealUp completeness principle. Hence the source triangle closes only the finite BEDC rows named in the statement. □

Theorem 126.358 (StreamName RegSeqRat Real dyadic-pullback seal determinacy)**.** *Let two accepted* StreamName$^{\uparrow}$ *finite-window packets be consumed through the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* NameCert *rows. If the two packets induce the same visible StreamName finite-window row, the same* RegSeqRat$^{\uparrow}$ *readback row, the same* DyadicRatCore$^{\uparrow}$ *observation ledger, and the same* Real$^{\uparrow}$ *seal row, then every* Real$^{\uparrow}$ *seal consumer reads the same finite pullback surface.*

In particular, the seal-facing read is determined by those four visible rows. No quotient stream, host real equality, selected limit, or ambient completeness datum can supply an alternate Real$^{\uparrow}$ *seal source for either packet.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRegseqratRealDyadicPullbackSealDeterminacy`

Proof. First use Definition 126.355. It identifies the visible sibling route from the accepted StreamName finite-window rows through the RegSeqRat$^{\uparrow}$ readback rows and into the Real$^{\uparrow}$ seal rows using only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next apply Definition 126.356. Since the two packets have the same StreamName window row, RegSeqRat readback row, DyadicRatCore ledger, and Real seal row, the four-row finite pullback surface available to a downstream seal consumer is the same on both sides.

The Real seal route is pinned by Definition 126.142: any Real$^{\uparrow}$ seal read arising from the StreamName-to-RegSeqRat handoff factors through the displayed finite-window rows, accepted

RegSeqRat seal rows, bundle and membership rows, unary observations, finite Cont ledger, displayed hsame transports, and Pkg provenance. The RegSeqRat readback coordinate is the same coordinate described by Definition 126.170.

Thus every seal-facing consumer reaches one finite pullback surface. The cited rows contain no coordinate for quotient streams, host real equality witnesses, selected limits, or ambient completeness data, so none of those objects can act as a second source for the $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 126.359 (StreamName RegSeqRat Real dyadic-pullback consumer exhaustion)**.** *Any downstream consumer citing the dyadic pullback route can project only the StreamName finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates on those rows. It cannot extract a quotient stream, host real equality, selected limit, or ambient completeness datum from the pullback.*

Proof. Apply Definition 126.356. The pullback route is exactly the four-row finite surface consisting of the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal, with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates attached.

Use Definition 126.358. The Real seal consumer is determined by those four rows, so any cited pullback route must read the same finite surface rather than an external seal source.

Exhaust the listed coordinates. None of the four sibling rows or boundary coordinates contains a quotient stream, host real equality witness, selected limit, or ambient completeness datum, and a downstream consumer cannot recover data absent from the displayed pullback packet. □

Theorem 126.360 (StreamName RegSeqRat Real terminal pullback exactness)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet factors through exactly the shared finite-window surface with four visible sibling rows: the StreamName finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The only shared coordinates available on that surface are the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* NameCert *rows already attached to those four rows.*

Consequently the terminal consumer has no alternate source through quotient streams, host real equality, selected limits, metric completeness, or ambient completeness. Its exact source is the displayed finite pullback packet.

Proof. First isolate the common surface. By Definition 126.356, the accepted finite-window packet is projected simultaneously to the StreamName finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ seal row, with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next pin the terminal seal read. Apply Definition 126.358. Once those four visible rows agree, every $\mathsf{Real}^{\uparrow}$ seal consumer reads the same finite pullback surface rather than a second seal-facing source.

Finally exhaust the consumer interface by Definition 126.359. A terminal consumer can project only the four rows and their displayed coordinates, so quotient streams, host real equality, selected limits, metric completeness, and ambient completeness have no coordinate from which an alternate source could be recovered. □

Theorem 126.361 (StreamName RegSeqRat Real uniform pullback non-escape)**.** *Let an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet be read through the uniform finite-window seal and then consumed by a downstream* $\mathsf{Real}^{\uparrow}$ *seal consumer. The consumer factors through the same four visible*

rows isolated in Definition 126.356: the StreamName finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row. Its only shared coordinates are the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* NameCert *rows attached to those four rows.*

Consequently the uniform finite-window seal and the dyadic pullback present a single finite BEDC carrier route to every such $\mathsf{Real}^{\uparrow}$ *seal consumer. No quotient stream, host real equality, selected limit, trichotomy row, or ambient completeness datum is exported by that route.*

Proof. First apply Definition 126.111. It places the accepted StreamName packet behind the uniform finite-window seal using the visible finite window, the displayed BHist observations, componentwise hsame transport, finite Cont rows, Pkg provenance, and local NameCert coordinates.

Next use Definition 126.356. The same sealed packet is read as the four-row pullback surface consisting of the StreamName finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ seal row. Thus the uniform seal does not create a route parallel to the dyadic pullback; it feeds the same finite carrier surface.

Apply Definition 126.358 to pin the downstream $\mathsf{Real}^{\uparrow}$ seal consumer to those four visible rows. Since the seal-facing read is determined by the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal row, no second seal source can enter through an external equality, limit, or completeness coordinate.

Finally apply Definition 126.359. The consumer can project only the four rows and their displayed BHist, hsame, Cont, Pkg, and NameCert coordinates. The listed surface has no quotient stream, host real equality, selected limit, trichotomy row, or ambient completeness datum, so none of those objects escapes through the uniform pullback route. □

Theorem 126.362 (StreamName RegSeqRat Real route transitivity)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet whose visible StreamName row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row agree with the dyadic pullback route and with the uniform finite-window seal route has a single composite* $\mathsf{Real}^{\uparrow}$ *seal consumer route. That route has only the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* NameCert *rows attached to those four sibling surfaces.*

The composite route exports no quotient stream, no host real equality, no selected limit, no trichotomy row, and no ambient completeness datum. Thus a downstream $\mathsf{Real}^{\uparrow}$ *seal consumer may cite one finite route from StreamName windows through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows into the* $\mathsf{Real}^{\uparrow}$ *seal, rather than separately choosing among the local handoff surfaces.*

Proof. First identify the finite four-row surface. Apply Definition 126.356. It places the accepted StreamName packet, the dyadic observation ledger, the RegSeqRat readback row, and the Real seal row on one pullback surface whose coordinates are exactly the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next compare the uniform seal with that surface. By Definition 126.361, the uniform finite-window seal feeds the same StreamName, DyadicRatCore, RegSeqRat, and Real rows instead of producing a parallel consumer route. Hence the dyadic pullback and the uniform seal name the same finite carrier path for the downstream $\mathsf{Real}^{\uparrow}$ seal read.

Finally exhaust the consumer coordinates. Apply Definition 126.359. A downstream consumer can project only the four displayed sibling rows and their BHist, hsame, Cont, Pkg, and NameCert coordinates. Since those rows contain no quotient stream, host real equality, selected limit, trichotomy row, or ambient completeness datum, no external coordinate can escape through the composite seal route. □

Theorem 126.363 (StreamName Real observation-budget factorization)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by a* $\mathsf{RealObservationBudget}^{\uparrow}$ *precision request factors through the same visible route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may use only the displayed BHist *observations, componentwise* hsame *transports, finite* Cont *routes,* Pkg *provenance, and local* NameCert *rows available on that route. The request supplies no quotient stream, no host equality, no hidden limit selection, and no ambient completeness coordinate.*

Proof. Start with the finite StreamName packet. By Definition 126.356, the packet is already read through the four visible sibling rows: StreamName finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal boundary.

Next attach the observation-budget consumer. The finite-route coverage theorem Definition 611.20 routes every accepted $\mathsf{RealObservationBudget}^{\uparrow}$ seal read through its precision request, finite window family, dyadic readback rows, regular-sequence handoff rows, and real seal rows. The consumer factorization theorem Definition 611.31 confines the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads to the same displayed hsame, Cont, Pkg, and local NameCert data.

Finally use Definition 126.359. The StreamName handoff exposes only the four finite rows and their listed BEDC coordinates, while the observation-budget theorems consume exactly those rows. No cited surface carries a quotient stream, host equality, hidden limit selection, or ambient completeness datum, so the precision request cannot introduce one. □

Theorem 126.364 (StreamName Real budget route determinacy)**.** *Let two accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets induce the same visible StreamName window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal row, and the same accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *precision-request route. Then every budget consumer reads the same finite BEDC route through the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance, and local* NameCert *coordinates.*

No quotient stream, host equality, selected limit, trichotomy row, or ambient completeness datum can provide an alternate budget source for either packet.

Proof. First use Definition 126.356. The shared visible StreamName window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal row place both accepted packets on the same four-row pullback surface.

Next attach the budget route. By Definition 126.363, any $\mathsf{RealObservationBudget}^{\uparrow}$ precision request consuming either packet factors through that StreamName-to-dyadic-to-regular-sequence-to-real route. The consumer route theorem Definition 611.31 confines the budget-facing $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads to the displayed hsame, Cont, Pkg, and local NameCert coordinates.

Apply Definition 126.358. Since the two packets agree on the four visible rows and on the accepted precision request route, the downstream seal-facing consumer has no second row from which to choose a different finite budget source.

Finally use consumer exhaustion for the pullback surface. The available consumer projections are exactly the listed BHist, hsame, Cont, Pkg, and NameCert coordinates, so quotient streams, host equalities, selected limits, trichotomy rows, and ambient completeness data cannot enter as alternate budget inputs. □

Theorem 126.365 (StreamName observation-budget CauchyLimitSeal compatibility)**.** *Let an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet be read both by a* $\mathsf{RealObservationBudget}^{\uparrow}$ *precision*

request and by a CauchyLimitSeal$^{\uparrow}$ *consumer. The two reads factor through the same visible four-row surface: the StreamName finite window, the* DyadicRatCore$^{\uparrow}$ *observation ledger, the* RegSeqRat$^{\uparrow}$ *readback, and the* Real$^{\uparrow}$ *seal row.*

The common route may project only the displayed BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance, and local* NameCert *coordinates on that finite pullback. It exports no quotient stream, host equality, selected limit, ambient completeness, metric-completion row, or standard bridge row.*

Proof. First read the precision-budget route. By Definition 126.363, every RealObservationBudget$^{\uparrow}$ request consumes the accepted StreamName packet through the displayed StreamName window, DyadicRatCore$^{\uparrow}$ ledger, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal, and the visible BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next read the CauchyLimitSeal route. The forward lattice theorem Definition 126.411 places the seal consumer on the StreamName window, RegSeqRat readback, dyadic ledger, Real seal row, and displayed finite-window seal surface. The reverse source exhaustion theorem Definition 126.412 says that no other public source row is available to that consumer.

Finally identify the two routes. Apply Definition 126.356 to read the StreamName finite window, DyadicRatCore$^{\uparrow}$ ledger, RegSeqRat$^{\uparrow}$ readback, and Real$^{\uparrow}$ seal as one finite pullback surface. Then use Definition 126.359: both consumers can project only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates of that pullback. Since neither cited route contains a quotient stream, host equality, selected limit, ambient completeness, metric-completion row, or standard bridge row, the two consumers cannot diverge into hidden completion routes. □

Theorem 126.366 (StreamName Real RegSeqRat seal pullback exhaustion)**.** *Every accepted* Real$^{\uparrow}$ *seal pullback read routed through an accepted* StreamName$^{\uparrow}$ *finite-window packet, the* RegSeqRat$^{\uparrow}$ *classifier row, and the* Real$^{\uparrow}$ *seal factors through exactly the displayed rows of this route: the StreamName finite-window row, the* DyadicRatCore$^{\uparrow}$ *observation ledger, the* RegSeqRat$^{\uparrow}$ *readback and classifier surface, the* Real$^{\uparrow}$ *seal row, and the attached* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Thus the Real seal pullback route exports no quotient stream, no selected limit, no host real equality, and no ambient completeness witness. A downstream seal consumer receives only the same finite packet presented by the StreamName finite window, the RegSeqRat row, and the Real seal boundary.

Proof. Start with the source lattice. By Definition 126.355, the accepted StreamName-to-RegSeqRat-to-Real route is already confined to the visible finite-window rows and the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates. The theorem also records that the route has no quotient stream, host real equality, selected limit, or ambient completeness coordinate.

Next force the finite pullback route. Apply Definition 126.356 to read the StreamName finite-window row, the DyadicRatCore$^{\uparrow}$ ledger, the RegSeqRat$^{\uparrow}$ readback row, and the Real$^{\uparrow}$ seal row as one finite surface. Then use Definition 126.359 to restrict every downstream consumer of that surface to the four displayed rows and their listed BEDC coordinates.

Finally compare the seal consumer. The compatibility theorem Definition 126.365 identifies the observation-budget read and the CauchyLimitSeal$^{\uparrow}$ read on the same finite pullback packet. Hence a Real seal pullback read routed through StreamName, RegSeqRat, and Real cannot select another source: it delivers precisely that finite packet to downstream seal consumers and no hidden completion witness beyond it. □

Theorem 126.367 (StreamName RealCompletionExactBoundary terminal pullback lock)**.** *Every accepted four-face* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *finite pullback consumed by a* $\mathsf{RealCompletionExactBoundaryUp}$ *packet factors through the exact terminal route*

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

The StreamName window is read as W, the RegSeqRat readback as R, the DyadicRatCore ledger as D, and the Real seal as E after the governing seal, classifier, witness-ledger, and budget rows L, K, J, S.

Conversely, the exact-boundary packet reads the same StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal as the four faces of the StreamName pullback. The lock exports no quotient stream, host sequence equality, selected limit, countable choice, ambient completeness, metric completeness, or bridge row.

Proof. First isolate the StreamName-side surface. By Definition 126.356, the accepted finite-window packet exposes exactly the StreamName finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{Real}^{\uparrow}$ seal row, and their displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates.

Next exhaust the seal read. Apply Definition 126.366: a Real seal pullback routed through StreamName, RegSeqRat, and Real delivers precisely that finite packet to downstream seal consumers and supplies no quotient stream, selected limit, host real equality, or ambient completeness witness.

Now read the completion boundary. The four-face exactness theorem Definition 1120.6 says that an accepted $\mathsf{RealCompletionExactBoundaryUp}$ packet exposes the four Phase Real and Completion faces W, R, D, E after the rows L, K, J, S and before the structural rows H, C, P, N. The terminal-route theorem Definition 1120.9 then pins the terminal $\mathsf{Real}^{\uparrow}(E)$ read to the same route $L, K, J, S, W, R, D, E, H, C, P, N$.

It remains to identify the two reads. The StreamName pullback names its four faces as the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal; the exact-boundary packet names the same four faces as W, R, D, E. Since both cited routes expose only their displayed finite BEDC coordinates, there is no additional row on which the two reads could diverge.

The exclusion clause follows from the same inventories. Neither the StreamName pullback exhaustion nor the exact-boundary terminal route contains a quotient stream, host sequence equality, selected limit, countable-choice row, ambient-completeness row, metric-completeness row, or bridge row, so an accepted consumer of one surface cannot import any of those objects into the other. □

Theorem 126.368 (StreamName RegSeqRat RealUp lattice budget cover)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet used as a Phase Real and Completion budget cover factors through exactly four sibling rows: the StreamName finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and classifier surface, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The cover exposes only the displayed* BHist *observations, componentwise* hsame *transports, finite* Cont *continuations,* Pkg *provenance rows, and local* $\mathsf{NameCert}$ *coordinates attached to those four rows.*

Consequently the budget cover supplies a finite route from StreamName windows through dyadic observation and regular-rational readback into the Real seal. It exports no quotient stream, no host real equality, no selected limit, no ambient completeness coordinate, no metric-completion row, and no standard bridge row.

Proof. First isolate the four-row pullback. By Definition 126.356, every accepted finite-window packet consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ is read through the StreamName finite-window row,

the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the $\mathsf{Real}^{\uparrow}$ seal row, with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Next exhaust the consumer projections. The theorem Definition 126.359 says that any downstream consumer citing that dyadic pullback can project only those four rows and their listed coordinates. Thus a budget cover cannot add a second carrier surface while remaining a consumer of the accepted finite-window packet.

Now attach the Phase Real budget route. By Definition 126.363, every $\mathsf{RealObservationBudget}^{\uparrow}$ precision request using the accepted packet factors through the same route from $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. The request supplies no quotient stream, host equality, hidden limit selection, or ambient completeness coordinate.

Finally pass to the Completion exit boundary. Apply Definition 126.366: the Real seal pullback read routed through StreamName, RegSeqRat, and Real delivers exactly the same finite packet to downstream seal consumers. Since the preceding factorization already lists all available BHist, hsame, Cont, Pkg, and NameCert coordinates, there is no remaining coordinate from which a metric-completion row or standard bridge row could be recovered. □

Theorem 126.369 (StreamName hub real-completion lattice pointer)**.** *Every public* $\mathsf{StreamName}^{\uparrow}$ *hub read used by the Phase Real and Completion route is routed through the sibling chain*

$$\textit{chapter } 37 \longrightarrow \textit{chapter } 301 \longrightarrow \textit{chapter } 38$$

and through the local finite-window child theorems of this hub. The route exports only displayed BHist *observations, componentwise* hsame *transports, finite* Cont *windows,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, host sequence equality, selected limit, or ambient completeness row.*

Proof. Read the hub as its ordered child surfaces. The carrier definition, certificate fields, finite-window endpoint package, and public interface expose only finite StreamName windows and their local naming rows.

Pass next through the Real and Completion sibling chain. The Rat surface of chapter 37, the RegSeqRat surface of chapter 301, and the Real surface of chapter 38 are reached through the local handoff and lattice-dependency children already included by the hub.

Use the local finite-window theorems to exhaust the route. The finite-window lattice meet, regular seal non-escape, real seal factorization boundary, and RegSeqRat real lattice dependency rows account for every public read used by the completion-facing consumer.

Since the hub contributes only child rows and no theorem body of its own, no quotient stream equality, host sequence equality, selected limit, or ambient completeness coordinate is exported from the hub read. □

Theorem 126.370 (Top-level StreamName Real-completion dependency)**.** *Every top-level* $\mathsf{StreamName}^{\uparrow}$ *certificate read used by the Real-and-completion route factors through the finite-window* $\mathsf{StreamName}^{\uparrow}$ *rows, the regular-source rows consumed by chapter 301, the dyadic observation ledger read by chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of chapter 38. The route exports no quotient stream equality, selected host limit, countable-choice selector, or ambient completeness principle.*

Proof. Project the top-level $\mathsf{StreamName}^{\uparrow}$ certificate to its finite-window rows. These are the only rows available before the Real-and-completion route reads any regular-source or dyadic observation data.

Send the finite-window rows first through the RegSeqRat dependency in chapter 301. The regular-source read is therefore a finite certificate handoff, not a selected limiting operation or a countable-choice selector.

Next route the same source surface through the DyadicRatCore observation ledger of chapter 291. Dyadic observations are read as displayed ledger rows, so they do not identify streams by quotient equality or by a host-level representative.

Finally close the route at the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The seal receives only the finite-window, regular-source, and dyadic ledger rows already displayed, and it imports no ambient completeness principle or selected host limit. □

Theorem 126.371 (StreamName RegSeqRat DyadicRatCore RealUp endpoint correspondence)**.** *A top-level Real-completion* $\mathsf{StreamName}^{\uparrow}$ *certificate read reaches a* $\mathsf{Real}^{\uparrow}$ *endpoint exactly when the same finite-window surface factors through the regular-source row of chapter 301, the observation ledger of chapter 291, and the seal boundary of chapter 38. The correspondence is a four-object handoff among* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; it is not quotient stream equality, a selected host limit, a countable-choice selector, or an ambient completeness principle.*

Proof. Start with Definition 126.370. It identifies the only top-level Real-completion route available from $\mathsf{StreamName}^{\uparrow}$: finite-window rows followed by regular-source, dyadic observation, and Real seal reads.

For the forward direction, a $\mathsf{Real}^{\uparrow}$ endpoint read must already have passed through the finite-window StreamName surface. The dependency theorem then forces the same displayed surface through the $\mathsf{RegSeqRat}^{\uparrow}$ regular-source row and the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger before the $\mathsf{Real}^{\uparrow}$ seal can be read.

For the reverse direction, assume the finite-window surface has these three sibling readings. The regular-source row supplies the source discipline, the dyadic ledger supplies the observation boundary, and the $\mathsf{Real}^{\uparrow}$ seal supplies the endpoint boundary, so the composed certificate read is exactly the licensed Real-completion endpoint route.

The excluded objects are absent at every stage of the composition. None of the cited routes adds quotient stream equality, selected host limits, countable-choice selectors, or ambient completeness; hence the endpoint correspondence remains a finite certificate handoff. □

Theorem 126.372 (StreamName four-object Real-completion exit exactness)**.** *A top-level Real-and-completion consumer reaches a public* $\mathsf{Real}^{\uparrow}$ *exit if and only if its read factors through exactly the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary named by Definition 126.370 and Definition 126.371. This exact exit route is the handoff surface shared with chapter 301, chapter 291, and chapter 38. No quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, or bridge row is part of the public exit.*

Proof. First apply Definition 126.370. It restricts every top-level Real-and-completion read to the finite $\mathsf{StreamName}^{\uparrow}$ window, the regular-source row, the dyadic observation ledger, and the $\mathsf{Real}^{\uparrow}$ seal boundary, and it excludes quotient stream equality, selected limits, choice selectors, and ambient completeness from the route.

For the forward direction, suppose a consumer reaches a public $\mathsf{Real}^{\uparrow}$ exit. By the endpoint correspondence of Definition 126.371, the same finite window must pass through the $\mathsf{RegSeqRat}^{\uparrow}$ row of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. Thus the public exit read has exactly the four displayed objects as its route.

For the reverse direction, assume a read factors through those four displayed objects and no additional exit coordinate. The dependency theorem supplies the top-level StreamName route, while the endpoint correspondence repacks the regular-source, dyadic-ledger, and seal-boundary readings as the public $\mathsf{Real}^{\uparrow}$ endpoint. Therefore the consumer reaches the public Real-and-completion exit.

The exactness claim is negative as well as positive. The preceding two theorems give no coordinate for quotient stream equality, selected host limits, countable-choice selectors, ambient completeness, or bridge rows; the three cited sibling chapters provide only their named finite certificate surfaces. Hence no extra public RealUp exit route remains beyond the four-object handoff. □

Theorem 126.373 (StreamName four-object exit status product)**.** *A public Real-and-completion exit from the top-level* $\mathsf{StreamName}^{\uparrow}$ *route is licensed exactly by the product of two independent records: the paper-side four-object route of Definition 126.372, and the verification-axis evidence separately attached to the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *targets. The route component supplies the finite-window StreamName read, the regular-source row of chapter 301, the dyadic observation ledger of chapter 291, and the Real seal boundary of chapter 38; the verification component supplies only the objectwise formal-status records governed by section 9467.2. Neither component determines the other, and the product contains no quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, or inference from closure grade to formal status or conversely.*

Proof. First read the paper-side route through Definition 126.370, Definition 126.371, and Definition 126.372. These results identify the only route to a public $\mathsf{Real}^{\uparrow}$ exit as the four displayed objects: the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-source row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ seal boundary.

Next read the status side through section 9467.2. That discipline treats paper closure, formal verification, and bridge status as distinct coordinates, so the four-object route cannot manufacture a formal-status row, and a formal-status row cannot manufacture the paper-side exit route.

Combining the two reads gives the stated product. The sibling chapters named above provide the four object surfaces, while the roadmap axis discipline supplies only the rule that their closure and verification records are reported independently.

The exclusions are inherited coordinatewise. The route theorems contain no quotient stream equality, selected host limit, countable-choice selector, or ambient completeness principle, and the verification-axis record contributes no additional mathematical exit route. Therefore neither direction of the product permits an inference from closure grade to formal status, from formal status to closure grade, or from either axis to any of the excluded completion principles. □

Theorem 126.374 (StreamName four-object exit support minimality)**.** *A public Phase Real and Completion exit from the top-level* $\mathsf{StreamName}^{\uparrow}$ *route is available exactly when one finite support displays all four object faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with the shared certificate faces* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Dropping any one of these displayed faces leaves only a partial interface: without* $\mathsf{StreamName}^{\uparrow}$ *there is no finite source window, without* $\mathsf{DyadicRatCore}^{\uparrow}$ *there is no observation ledger, without* $\mathsf{RegSeqRat}^{\uparrow}$ *there is no regularity row, without* $\mathsf{Real}^{\uparrow}$ *there is no seal boundary, and without one of* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *there is no common certificate surface on which the four object faces can be read as one public exit.*

Proof. The four object faces are forced by Definition 126.372. That theorem identifies the public Real-and-completion exit precisely with the route through the finite $\mathsf{StreamName}^{\uparrow}$ window, the

$\mathsf{RegSeqRat}^{\uparrow}$ regular-source row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ seal boundary.

The status-product theorem Definition 126.373 keeps this route separate from verification-axis bookkeeping. Hence the support-minimality statement is a paper-side route claim about the finite support itself, not an inference from a closure grade, formal status, or bridge coordinate.

It remains to justify the common certificate faces. Apply Definition 126.651: any reindexed support row that can be read by the exit route must carry the displayed BHist, hsame, Cont, Pkg, and NameCert surface. Removing one of those faces leaves at most a source row, observation row, regularity row, or seal row in isolation.

Combining these two requirements gives minimality. The four object faces provide exactly the source, observation, regularity, and seal components of the public exit, while the five certificate faces provide the common finite support on which those components are read together. If any displayed face is absent, one component of the public exit or its shared support is absent, so the remaining data describes only the corresponding partial interface. □

126.24 StreamName finite-window transport obligations

Theorem 126.375 (StreamName finite-window pointwise transport)**.** *Let* $s, t, s', t' : \mathsf{BHist} \to \mathsf{BHist}$ *be certified* $\mathsf{StreamName}^{\uparrow}$ *rows, let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite observation bundle, and suppose*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathcal{P})$$

is displayed together with the finite ledger rows $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$ *for every selected observation. Suppose also that every selected* n *carries the pointwise transport rows*

$$\mathsf{hsame}(s(n), s'(n)) \qquad \textit{and} \qquad \mathsf{hsame}(t(n), t'(n)),$$

and that the consumer route reads the source window only through displayed Cont *rows. Then the transported window exported to any* RegSeqRat *or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer is exactly*

$$\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P})$$

with the same ProbeBundle*,* InBundle*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

No infinite stream object, quotient stream equality, selected real value, host Nat stream, or $\mathsf{Real}^{\uparrow}$ *completeness principle is exported by this transport.*

Proof. First read the source finite window through the public handoff surface of Definition 126.79. The available rows are the selected rational-history classifier rows, the finite ProbeBundle ledger, membership rows, unary observation rows, and their local naming data.

Next apply the pointwise handoff and transport rows of Definition 126.170 and Definition 126.171. These identify the transported classifier by using only the displayed pointwise hsame rows at the certified observations of $\mathcal{P}$.

Finally replay the result through the finite pullback criterion of Definition 126.152. The criterion records the same Cont, Pkg, and local certificate rows as consumer-visible coordinates, so the transported packet remains the finite window over $\mathcal{P}$ and exports none of the excluded stream, quotient, real-value, host-index, or completeness data. □

Theorem 126.376 (StreamName constant-window classifier stability)**.** *Let* $d, e : \mathsf{BHist}$ *be rational-history endpoints and let* $r, q : \mathsf{BHist} \to \mathsf{BHist}$ *be certified observation schedules over a finite bundle*

$\mathcal{P}$: ProbeBundle. *Suppose the constant window displays the rational point classifier for d and e, the selected bundle rows* InBundle$(n, \mathcal{P})$*, the unary ledger rows* UnaryHistory(n)*, pointwise* hsame *transport for the scheduled constant observations, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row. Then every transported constant certified window preserves the same point classifier and ledger rows for the* RegSeqRat *and finite-window* $\mathsf{Real}^{\uparrow}$ *handoff.*

The stable constant-window packet contains no infinite stream object, quotient stream equality, selected real value, host Nat stream, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the constant boundary of Definition 126.81. It says that the constant window contributes only the rational point classifier, finite bundle ledger, membership rows, unary observation rows, and local naming data.

Transport the scheduled constant observations by Definition 126.172. The handoff reads the same endpoint classifier pointwise and leaves regularity, dyadic seal data, and the $\mathsf{Real}^{\uparrow}$ seal classifier on the consumer side.

Now use Definition 126.170 to identify the consumer-facing packet. The pointwise rows, bundle membership, unary ledger, displayed hsame transport, Cont route, Pkg provenance, and local certificate row are precisely the rows named by the theorem statement, so the constant classifier and ledger are stable under the certified transport. □

Theorem 126.377 (StreamName RegSeqRat window handoff exactness)**.** *Let $\mathcal{W}$ be a finite* $\mathsf{StreamName}^{\uparrow}$ *window over s, t :* BHist $\to$ BHist *with selected bundle $\mathcal{P}$:* ProbeBundle*. The exact* RegSeqRat *handoff surface read from $\mathcal{W}$ consists only of the finite window classifier* RatStreamNameFiniteWindowCla *the selected point certificates* RatHistoryClassifier$(s(n), t(n))$*, the pointwise* UnaryHistory *and* InBundle *ledger rows, the displayed* hsame *transports, the* Cont *route rows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row.*

Conversely, those finite rows determine the whole source packet handed to RegSeqRat*; no other StreamName coordinate is read. In particular, the handoff exports no infinite stream object, quotient stream equality, selected real value, host Nat stream,* RegSeqRat *regularity witness supplied by StreamName, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. For the forward direction, project $\mathcal{W}$ through Definition 126.79. It exposes the finite StreamName window rows, selected rational-history classifier rows, probe-bundle ledger, bundle membership, and unary-history observations.

Apply the pointwise handoff theorem Definition 126.170. It refines the same source packet to point certificates RatHistoryClassifier$(s(n), t(n))$ at exactly the selected certified observations of $\mathcal{P}$.

For the reverse direction, repack the displayed classifier, point certificates, ledger rows, hsame transports, route rows, package provenance, and local certificate row through Definition 126.152. The criterion says that these are precisely the finite pullback coordinates seen by the downstream consumers, so the source handoff is fully determined.

Finally use Definition 126.82. It keeps RegSeqRat regularity and $\mathsf{Real}^{\uparrow}$ seal or completeness obligations outside the StreamName source packet, and it rules out the infinite-stream, quotient, selected-real, and host-index data excluded above. □

Theorem 126.378 (StreamName finite-window transport RegSeqRat Real readiness)**.** *A transported finite* $\mathsf{StreamName}^{\uparrow}$ *window is ready for* RegSeqRat *and finite-window* $\mathsf{Real}^{\uparrow}$ *consumers exactly when the finite-window classifier, selected point certificates,* UnaryHistory *rows,* InBundle *rows, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row are all present in the source packet. This readiness surface exports no quotient stream equality, host Nat stream, selected real value,* RegSeqRat *regularity row, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply pointwise transport, Definition 126.375. It moves the finite classifier to the transported window while preserving the same bundle, membership, unary-history, pointwise hsame, route, provenance, and local naming rows.

Use constant-window classifier stability when the selected observations are constant endpoints. Theorem 126.376 shows that the same finite classifier and ledger rows remain the consumer-facing surface for RegSeqRat and finite-window $\mathsf{Real}^{\uparrow}$ handoff.

Next apply RegSeqRat window handoff exactness, Definition 126.377. It identifies the complete source packet as the finite-window classifier, selected point certificates, UnaryHistory and InBundle rows, hsame, Cont, Pkg, and local NameCert data.

Finally cite the pointwise handoff theorem Definition 126.170. It keeps RegSeqRat regularity and $\mathsf{Real}^{\uparrow}$ completion obligations on the downstream side, so the transported StreamName source packet is ready for those consumers without exporting quotient stream equality, a host Nat stream, a selected real value, regularity, or completeness. □

Theorem 126.379 (StreamName finite-window section transport closure)**.** *Let $\mathcal{S}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window section whose displayed rows contain the finite classifier, selected point certificates,* UnaryHistory *observations,* InBundle *ledger, componentwise* BHist *rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *registration. If $\mathcal{S}$ is transported through those displayed* hsame *and* Cont *rows, then the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal remain on that same transported section route.*

Conversely, any current Phase Real and Completion read that exposes the same finite $\mathsf{StreamName}^{\uparrow}$ *window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and displayed* BHist, hsame, Cont, Pkg, NameCert *rows is read as the transported section $\mathcal{S}$. This section closure exports no quotient stream, selected limit, host sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or standard bridge row.*

Proof. For the forward direction, apply the pointwise transport theorem Definition 126.375. It moves the accepted finite window through the displayed hsame rows while preserving the same bundle, membership, unary-history, continuation, provenance, and local naming surface.

Next apply RegSeqRat window handoff exactness, Definition 126.377. The handoff identifies the downstream readback with exactly the transported finite-window classifier and selected point certificates, so the $\mathsf{RegSeqRat}^{\uparrow}$ read remains on the same section route.

Finally use the readiness theorem Definition 126.378. It states that the finite classifier, point certificates, dyadic tolerance ledger, transports, routes, provenance, and local naming are sufficient for the finite-window $\mathsf{Real}^{\uparrow}$ consumer, while regularity and completion obligations stay downstream.

For the converse direction, inspect the exposed Phase Real and Completion read. If it shows the same finite StreamName window, dyadic ledger, RegSeqRat readback, Real seal, and displayed BHist, hsame, Cont, Pkg, NameCert rows, then the preceding exactness and readiness results reconstruct the unique transported section route read by those consumers. The exclusions follow because none of the cited rows contains quotient streams, selected limits, host sequence equality, ambient Real completeness, metric completeness, or standard bridge data. □

Theorem 126.380 (StreamName finite-window section pullback stability)**.** *Let $\mathcal{S}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window section whose source rows display the finite classifier, selected point certificates,* UnaryHistory *observations,* InBundle *ledger, componentwise* BHist *entries,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *registration. If $\mathcal{S}$ is transported through those displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *rows, then every downstream finite* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer reading that section pulls back to the same*

transported four-face route: the transported finite $\mathsf{StreamName}^{\uparrow}$ *classifier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal.*

This pullback stability exports no quotient stream equality, host sequence equality, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, metric-completeness datum, or standard bridge row.*

Proof. First use the pointwise transport theorem Definition 126.375. It transports the finite classifier and selected point certificates through the displayed BHist and hsame rows while preserving the same bundle membership, unary-history, continuation, provenance, and local naming surface.

Next apply RegSeqRat window handoff exactness, Definition 126.377. The handoff says that the $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads exactly the transported finite-window classifier, selected point certificates, support rows, transports, routes, provenance, and local certificate rows; no second StreamName coordinate is available for the readback.

Now use downstream readiness, Definition 126.378. Readiness carries the same transported source packet to the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger and the finite-window $\mathsf{Real}^{\uparrow}$ seal without importing regularity, limit-selection, choice, or completeness rows from StreamName.

Finally compose with the existing section route, Definition 126.379. That closure result identifies the exposed Phase Real and Completion read with the transported section $\mathcal{S}$. Therefore each downstream consumer pulls back to the same four-face route, and the exclusion list follows from the fact that none of the cited handoff, readiness, or section rows contains quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, or standard bridge data. □

Theorem 126.381 (StreamName pointwise transport scope)**.** *Let* $\mathcal{S}$ *be an accepted transported finite* $\mathsf{StreamName}^{\uparrow}$ *packet over a finite bundle* $\mathcal{P} : \mathsf{ProbeBundle}$*, with source and target schedules* $s, t, s', t' : \mathsf{BHist} \to \mathsf{BHist}$*. Before any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer reads* $\mathcal{S}$*, the packet exposes exactly the following StreamName-side rows: the finite classifier* $\mathsf{RatStreamNameFiniteWindowClassifier}(s', t', \mathcal{P})$*, the selected point-certificate rows* $\mathsf{RatHistoryClassifier}(s'(n), t'(n))$ *at the displayed observations, the support rows* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$*, the componentwise* hsame *transports from* $s(n), t(n)$ *to* $s'(n), t'(n)$*, the replayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row.*

The same scope exports no quotient stream equality, host sequence equality, selected limit, $\mathsf{Real}^{\uparrow}$ *completeness principle, metric-completeness datum, countable-choice row, or standard bridge row. Thus the finite-window packet seen by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers is a finite transported row packet, not a hidden stream quotient or completion claim.*

Proof. Start with pointwise transport, Definition 126.375. It transports the finite classifier across the displayed hsame rows and preserves the same bundle, support, continuation, provenance, and local naming surface.

Next use the handoff exactness theorem Definition 126.377. This identifies the rows read by $\mathsf{RegSeqRat}^{\uparrow}$ as the finite classifier, selected point certificates, support rows, transports, routes, provenance, and local certificate, and it rules out any additional StreamName coordinate.

Then apply readiness, Definition 126.378. Readiness says that the same rows are sufficient for the finite-window $\mathsf{Real}^{\uparrow}$ route while quotient equality, host sequence data, selected real values, regularity, and completeness obligations remain outside the source packet.

Finally compose with section transport closure, Definition 126.379. The closure theorem keeps the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the finite-window $\mathsf{Real}^{\uparrow}$ seal on the same transported section route. Its exclusion list supplies the stated absence of selected

limits, metric completeness, and standard bridge rows, so the displayed row list is exactly the scope available before the downstream consumers read the packet. □

Theorem 126.382 (StreamName finite-window global classifier exhaustion)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be certified StreamName schedules. Assume that for every certified unary history* $n : \mathsf{BHist}$ *the singleton window* $\mathsf{Bcons}(n, \mathsf{Bnil})$ *is supplied with a finite row* $\mathsf{RatStreamNameFiniteWindowClassifier}(s, t, \mathsf{Bcons}(n, \mathsf{Bni}$ *its* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathsf{Bcons}(n, \mathsf{Bnil}))$ *support rows, the pointwise classifier* $\mathsf{RatHistoryClassifier}(s(n), t($ *displayed* hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^\uparrow}$ *support. Then the public StreamName source available to any* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*, or finite-window* $\mathsf{Real}^\uparrow$ *consumer is exactly the reconstructed global classifier* $\mathsf{RatStreamNameClassifier}(s, t)$*.*

After this reconstruction, the downstream read is only through the existing handoff rows Definition 126.170, Definition 126.377, and Definition 126.152. It exports no quotient stream equality, host sequence equality, selected limit, regularity row, completeness datum, metric-completion row, or standard bridge row from the StreamName source packet.

Proof. First rebuild the source classifier. Apply Definition 126.23 to the family of singleton-window classifiers. Since every certified unary history is supplied with its singleton finite row, support rows, and pointwise RatHistoryClassifier witness, singleton exhaustion repacks those rows as the global $\mathsf{RatStreamNameClassifier}(s, t)$.

Next identify what the regular-rational consumer can read. Apply Definition 126.170 to the reconstructed classifier and to the displayed singleton rows. The $\mathsf{RegSeqRat}^\uparrow$ interface receives the selected rational rows as StreamName pointwise data, together with the same unary-history, bundle-membership, hsame, Cont, Pkg, and local naming rows.

Now apply the finite-window handoff exactness theorem, Definition 126.377. It says that the complete finite handoff packet is precisely the finite classifier and selected point certificates already used to reconstruct the global StreamName classifier, so the downstream read has no second source coordinate.

Finally apply the finite-window pullback criterion, Definition 126.152. The $\mathsf{Real}^\uparrow$ and $\mathsf{DyadicRatCore}^\uparrow$ consumers therefore pull back only along the same finite StreamName rows. Quotient stream equality, host sequence equality, selected limits, regularity, completeness, metric completion, and standard bridge data remain downstream obligations rather than StreamName source rows. □

Theorem 126.383 (StreamName pointwise finite-window transport)**.** *Let* $s, t, s', t' : \mathsf{BHist} \to \mathsf{BHist}$ *be displayed* $\mathsf{StreamName}^\uparrow$ *rows and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite certified observation window. Suppose the source window exposes*

$$\mathsf{RatStreamNameCarrier}(s), \quad \mathsf{RatStreamNameCarrier}(t), \quad \mathsf{RatStreamNameClassifier}(s, t), \quad \mathsf{RatStreamNameFiniteWi}$$

together with the finite request ledger rows $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$ *for every selected observation. If each selected observation carries pointwise transports* $\mathsf{hsame}(s(n), s'(n))$ *and* $\mathsf{hsame}(t(n), t'(n))$*, and the same* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^\uparrow}$ *rows replay the transported window, then the transported packet preserves StreamCarrier membership, StreamSame classification, and the finite request ledger:*

$$\mathsf{RatStreamNameCarrier}(s'), \quad \mathsf{RatStreamNameCarrier}(t'), \quad \mathsf{RatStreamNameClassifier}(s', t'), \quad \mathsf{RatStreamNameFiniteW}$$

The handoff exports exactly the same finite bundle, unary-history ledger, membership rows, pointwise hsame *transports, replay, provenance, and local naming rows to downstream* $\mathsf{RegSeqRat}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *consumers. It exports no quotient stream equality, host sequence equality, selected real value, countable-choice schedule, or ambient* $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. Project the source finite window. Carrier membership for s and t, the StreamSame classifier, the finite-window classifier, and the ledger rows are all displayed before any downstream consumer reads the packet.

Apply the pointwise transport rows at each certified observation of $\mathcal{P}$. The rows $\mathsf{hsame}(s(n), s'(n))$ and $\mathsf{hsame}(t(n), t'(n))$ transport the opened BHist observations without changing the bundle membership or unary-history ledger.

Use the finite-window transport obligations of Definition 126.375 together with the RegSeqRat pointwise handoff of Definition 126.170. They identify the transported carrier and classifier rows over the same finite bundle and keep regular readback and Real sealing on the consumer side.

Finally replay the transported packet through the displayed Cont row and record its Pkg provenance and local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ row. Since every exported coordinate is one of the displayed finite rows, no quotient stream equality, host sequence equality, selected real value, countable-choice schedule, or ambient completeness datum can enter the handoff. □

126.25 StreamName finite-window nonempty-index handoff

Theorem 126.384 (StreamName finite-window nonempty-index handoff)**.** *Let $\mathcal{W}$ be a finite* $\mathsf{StreamName}^{\uparrow}$ *package consumed by a* $\mathsf{RegSeqRat}$ *certificate or by a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer. Before any regularity row, dyadic ledger row, or* $\mathsf{Real}^{\uparrow}$ *seal row is readable, the handoff from $\mathcal{W}$ must expose at least one displayed observation row through a finite* $\mathsf{ProbeBundle}$*, an* $\mathsf{InBundle}(n, \mathcal{P})$ *membership row, and a certified* $\mathsf{UnaryHistory}(n)$ *index. This exported index is only a finite observation witness for the displayed window.*

The handoff exports no host index equality, quotient stream equality, selected limit, or ambient completeness principle.

Proof. First read the public finite-window handoff. By Definition 126.79, a downstream $\mathsf{RegSeqRat}$ or finite-window $\mathsf{Real}^{\uparrow}$ consumer can receive only the selected rational-history classifier rows, the finite $\mathsf{ProbeBundle}$ ledger, the $\mathsf{InBundle}$ membership rows, and the certified $\mathsf{UnaryHistory}$ observation rows from the StreamName side.

Next place that source row inside the finite meet packet of Definition 126.256. The meet packet lists the common classifier row, probe bundle, membership family, unary observation family, dyadic ledger surface, componentwise transports, continuation routes, package provenance, and local names. Hence any downstream dyadic or seal read already has a visible finite observation coordinate before it can be consumed.

Now apply the exhaustion theorem Definition 126.257. It says that every $\mathsf{RegSeqRat}$ or finite-window $\mathsf{Real}^{\uparrow}$ consumer of the meet reads only the displayed finite observation surface and its ledger rows. Therefore the first consumer-visible index is a certified member of the probe bundle, not a host index equality or a quotient-stream coordinate.

Finally use the sibling dependency theorem Definition 126.355. The route from StreamName through $\mathsf{RegSeqRat}^{\uparrow}$ into $\mathsf{Real}^{\uparrow}$ is the same visible finite-window route, so regularity, dyadic ledger, and Real seal rows are downstream consumers of the already displayed observation witness. The exclusion clauses in the cited handoff, meet, exhaustion, and dependency results rule out quotient stream equality, selected limits, and ambient completeness as exported StreamName data. □

Theorem 126.385 (StreamName finite-window support minimality)**.** *Any downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *finite-window consumer that reads a* $\mathsf{StreamName}^{\uparrow}$ *package must preserve at least one displayed* $\mathsf{ProbeBundle}$ *membership row together with its* $\mathsf{UnaryHistory}$ *index through the same* hsame*,*

Cont, Pkg, *and* NameCert *support. Deleting that row destroys the consumer route rather than moving it to quotient stream equality, selected limits, or ambient completeness.*

Proof. Start with Definition 126.384. It forces any RegSeqRat or Real finite-window consumer to expose a displayed observation through a finite ProbeBundle, its membership row, and the certified UnaryHistory index before regularity or seal data are readable.

Place the same row in the finite-window meet of Definition 126.257. The exhaustion result says that the consumer can only read the displayed classifier, membership, unary-history, dyadic ledger, transport, replay, provenance, and local-name rows carried by that finite surface.

Now apply Definition 126.355. The RegSeqRat readback and Real seal are downstream of the same StreamName finite-window surface, so their hsame, Cont, Pkg, and NameCert support still contains the nonempty observation row.

If that row is deleted, the displayed membership and unary-history coordinates needed by the handoff are absent. The exclusion clauses in the handoff, exhaustion, and dependency results leave no coordinate for quotient stream equality, selected limits, or ambient completeness to supply a replacement route. □

Theorem 126.386 (StreamName window-index route determinacy)**.** *Let $\mathcal{W}$ and $\mathcal{W}'$ be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *packets consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. If both packets expose the same finite* ProbeBundle *member n, the same certified* $\mathsf{UnaryHistory}(n)$ *row, and the same finite-window meet coordinates, then every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal consumer reads the same finite route through the StreamName window, the RegSeqRat readback, the dyadic ledger, and the Real seal.*

No quotient stream equality, host index equality, selected limit, or ambient completeness theorem can provide an alternate route for either packet.

Proof. Apply Definition 126.384 to both packets. The common visible index is a certified member of the finite ProbeBundle and is already paired with the displayed $\mathsf{UnaryHistory}(n)$ row before any RegSeqRat or Real consumer reads the packet.

Place both packets in the finite meet packet of Definition 126.256. The hypothesis that the meet coordinates agree identifies the common classifier row, probe-bundle membership row, unary observation row, dyadic ledger surface, componentwise transport, continuation, provenance, and local name.

Use Definition 126.257 together with Definition 126.355. These results force the RegSeqRat readback and Real seal to be downstream reads of the same finite meet surface, so the consumer route is determined by the displayed coordinates.

Coordinate exhaustion closes the claim. The agreed meet surface has no coordinate for quotient stream equality, host index equality, selected limits, or ambient completeness, so none of those objects can act as a second route to the downstream seal. □

Theorem 126.387 (StreamName window-index sibling route certificate)**.** *Let $\mathcal{W}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *packet whose nonempty index handoff is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal. Then the consumer route factors through the same finite sibling route: the* ProbeBundle *row, the* $\mathsf{InBundle}(n, \mathcal{P})$ *membership row, and the* $\mathsf{UnaryHistory}(n)$ *index row first exposed by the StreamName window; the RegSeqRat readback surface of chapter 301; the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of chapter 291; and the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

This certificate exports no host index equality, quotient stream equality, selected limit, trichotomy, or ambient completeness theorem.

Proof. Start with Definition 126.384. The accepted finite packet must expose a displayed observation row through the finite ProbeBundle, its InBundle membership witness, and the certified UnaryHistory index before any RegSeqRat or Real consumer can read it.

Use Definition 126.386 to make that row the unique finite route for the downstream read. Once the displayed index, unary history, and meet coordinates are fixed, the readback and seal consumers cannot select a second path through host index equality or quotient stream equality.

Now apply Definition 126.355. The RegSeqRat surface named by chapter 301 is a downstream consumer of the StreamName finite-window row, and the Real seal named by chapter 38 is reached only through that displayed readback route.

Finally use Definition 126.356. The pullback inserts the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger of chapter 291 between the RegSeqRat readback and the Real seal without adding any coordinate for a selected limit, trichotomy, or ambient completeness. Hence every accepted consumer route factors through the same four sibling surfaces and through the same nonempty window-index row. □

Theorem 126.388 (StreamName Real RegSeq observation pullback lock)**.** *Let $\mathcal{W}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed first by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then by a* $\mathsf{Real}^{\uparrow}$ *seal. The first observation exposed to that route is only the certified* ProbeBundle *member together with its* UnaryHistory *row, and every downstream seal read factors through the same four-row pullback: the StreamName finite-window row, the RegSeqRat readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, and the* $\mathsf{Real}^{\uparrow}$ *seal row.*

The lock exports no host index equality, quotient stream equality, selected limit, trichotomy, or ambient completeness theorem.

Proof. Start with Definition 126.384. It forces the first visible observation to be a finite ProbeBundle member paired with its certified UnaryHistory row before a RegSeqRat or Real consumer can read the packet.

Use Definition 126.386 to keep that observation on one finite route. Once the displayed window coordinate, membership row, unary history row, and meet coordinates are fixed, no host index equality or quotient stream equality can supply a second downstream path.

Now apply Definition 126.356 and Definition 126.359. These results identify the downstream surface as the four displayed rows: StreamName finite window, RegSeqRat readback, DyadicRatCore ledger, and Real seal.

Coordinate exhaustion gives the lock. The four-row pullback has no coordinate for a selected limit, trichotomy, or ambient completeness, and the initial observation coordinate remains the certified bundle member plus unary history row throughout the downstream seal read. □

Theorem 126.389 (StreamName window-refinement first-observation stability)**.** *Let $\mathcal{W}$ and $\mathcal{W}'$ be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose that $\mathcal{W}'$ refines $\mathcal{W}$ by a displayed* ProbeBundle *inclusion, preserves the same* $\mathsf{InBundle}(n, \mathcal{P})$ *membership row, preserves the certified* $\mathsf{UnaryHistory}(n)$ *observation index, and feeds the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal route as $\mathcal{W}$. Then the first observation exposed to the downstream consumer is the same displayed bundle member n with the same unary-history row in both windows.*

The stable first observation is carried only by the displayed finite-window route. Neither window admits an alternate route through host index equality, quotient stream equality, selected limits, trichotomy, or ambient completeness.

Proof. First apply Definition 126.384 to both accepted windows. Each downstream RegSeqRat or Real consumer must receive a first visible observation through a finite ProbeBundle member, its $\mathsf{InBundle}(n, \mathcal{P})$ row, and the certified $\mathsf{UnaryHistory}(n)$ index.

The refinement hypotheses identify those visible rows. The displayed ProbeBundle inclusion carries the same member n, and the preserved membership and unary-history rows keep the first observation at the same finite coordinate before any readback or seal row is consumed.

Now use Definition 126.386. Once the index, membership row, unary-history row, and finite meet coordinates agree, the downstream route is forced to be the same StreamName window route for the RegSeqRat readback and the Real seal.

Next apply Definition 126.387. The common route factors through the displayed StreamName row, the RegSeqRat readback surface, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the $\mathsf{Real}^{\uparrow}$ seal, so the refined window cannot expose a different first observation to the same consumer.

Finally apply Definition 126.388. The Real seal pulls back along the same four displayed rows and keeps the initial bundle member plus unary-history row fixed. The exclusion clauses in the determinacy, sibling-route, and pullback-lock results leave no coordinate for host index equality, quotient stream equality, selected limits, trichotomy, or ambient completeness to serve as a second route. □

126.26 StreamName finite-window budget handoff

Theorem 126.390 (StreamName Real RegSeq finite-window budget handoff)**.** *Let $\mathcal{W}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *package consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *certificate, by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal route, or by a route that passes through a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *or* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Then any consumer-visible regularity row, dyadic tolerance row, or* $\mathsf{Real}^{\uparrow}$ *seal row must be downstream of the same finite observation-budget surface:*

$$\mathsf{ProbeBundle},\quad \mathsf{InBundle},\quad \mathsf{UnaryHistory},\quad \mathsf{ObservationBudgetLimiter}^{\uparrow},\quad \mathsf{RegularCauchyTailBudget}^{\uparrow}.$$

Thus the StreamName handoff first exposes the finite window, membership, unary-history, budget-limiter, and tail-budget rows before any RegSeqRat regularity, dyadic tolerance, or Real seal row is consumed.

The handoff exports no quotient stream, selected host limit, countable-choice selector, or host real equality.

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealRegseqFiniteWindowBudgetHandoff`

Proof. Start with Definition 126.79. The public StreamName package exports only selected rational-history classifier rows, the finite ProbeBundle ledger, InBundle membership rows, and UnaryHistory observation rows to a $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ consumer.

Next apply Definition 126.82. It places regularity and dyadic-tolerance rows on the $\mathsf{RegSeqRat}^{\uparrow}$ side and the seal classifier on the $\mathsf{Real}^{\uparrow}$ side, so none of those rows can be consumed as StreamName source data.

Now route the same finite observation surface through the sibling budget chapters. The limiter factorization Definition 903.23 forces any $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ use of the limiter row through the request-budget-cap service triad before tail windows, dyadic observations, readback, and seal rows are served.

Finally use Definition 906.3. It orders the regular source, tail threshold, observation limiter, finite StreamName window, dyadic ledger, RegSeqRat readback, and Real seal as one finite route. Since the cited StreamName, limiter, and tail-budget surfaces contain no coordinates for quotient streams, selected host limits, countable choice, or host real equality, those objects cannot be alternate sources for the handoff. □

Theorem 126.391 (StreamName budget handoff sibling chapter route)**.** *Every accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed through the finite observation-budget surface factors through the sibling chapter surface of chapter 903 before the tail-budget service of chapter 906. Its consumer-visible route then passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal rows already named in Definition 126.390. No accepted finite StreamName handoff obtains an alternate source for quotient streams, selected host limits, countable choice, or host real equality.*

Proof. Apply Definition 126.390 to the accepted finite StreamName window. It already identifies the finite observation surface that must be exposed before any $\mathsf{RegSeqRat}^{\uparrow}$ regularity, dyadic-tolerance, or $\mathsf{Real}^{\uparrow}$ seal row is consumed.

Next read the limiter leg through Definition 903.23. The factorization places the handoff on the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ request-budget-cap service triad before any $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ tail window, dyadic observation, readback, or seal row can be served.

Finally use Definition 906.3. It orders the tail-budget route through the regular source, tail threshold, observation limiter, finite StreamName window, dyadic ledger, RegSeqRat readback, and Real seal. The non-escape sentence in Definition 126.390 therefore applies to the whole sibling chapter route. □

Theorem 126.392 (StreamName budget RegSeq Real route coherence)**.** *Any accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *and* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal through the same ordered finite route. The route cannot fork into quotient streams, selected host limits, countable-choice schedules, or host real equality.*

Proof. Start with Definition 126.390. It places the accepted finite StreamName window behind the finite observation-budget surface before any $\mathsf{RegSeqRat}^{\uparrow}$ regularity, dyadic tolerance, or $\mathsf{Real}^{\uparrow}$ seal row can be consumed.

Next apply Definition 126.391. The sibling route first reads the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ surface and then the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ service surface before the regular readback and seal rows are available.

Use Definition 903.23 for the limiter leg and Definition 906.3 for the tail-budget leg. Together they identify one ordered finite route through the limiter request-budget-cap service, the tail threshold, the finite StreamName window, the dyadic ledger, the RegSeqRat readback, and the Real seal.

Since each cited route uses the same displayed finite rows and none carries a quotient stream, selected host limit, countable-choice schedule, or host real equality coordinate, no consumer can fork the budget handoff into an alternate readback or seal route. □

Theorem 126.393 (StreamName regular seal sibling route)**.** *Any finite* $\mathsf{StreamName}^{\uparrow}$ *window routed through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *and* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *reaches a* $\mathsf{Real}^{\uparrow}$ *finite-window seal only through the same rows used by Definition 126.337:*

$$\mathsf{ProbeBundle},\quad \mathsf{InBundle},\quad \mathsf{UnaryHistory},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The sibling route exports no quotient stream, selected host limit, countable-choice selector, host real equality, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 126.390. It places every finite-window regularity, dyadic tolerance, and Real seal read behind the finite observation-budget surface.

Use Definition 126.391 to pass that surface through the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ and $\mathsf{RegularCauchyTailBudget}$ sibling chapters. The route still exposes only the finite window, membership, unary-history, budget, tail-budget, readback, and seal rows.

Finally apply Definition 126.337. Its finite-window Real seal boundary is exactly the row list displayed above, so the sibling handoff cannot export quotient stream equality, selected limits, countable-choice selectors, host real equality, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.394 (StreamName finite-window prefix induction)**.** *Any accepted finite* $\mathsf{StreamName}^{\uparrow}$ *handoff consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or by a* $\mathsf{Real}^{\uparrow}$ *finite-window seal admits induction on the displayed finite prefix rows*

$\mathsf{ProbeBundle}$, $\mathsf{InBundle}$, $\mathsf{UnaryHistory}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$.

At each induction step the same $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary are preserved by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. No later tail, quotient stream, selected host limit, countable-choice schedule, or host real equality can enter the induction.*

Proof. Project the finite handoff theorem Definition 126.390. It places every regularity read, dyadic tolerance read, and finite-window Real seal behind the displayed finite observation rows.

Use the sibling route of Definition 126.393. It identifies the handoff consumed by $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ with the same finite prefix surface routed through $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ and $\mathsf{RegularCauchyTailBudget}^{\uparrow}$.

Induct over the displayed prefix rows in the listed order. The base case is the empty displayed prefix, where the readback and seal boundary are only the structural hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. The step case appends one visible prefix row and replays the same transport and continuation rows, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal boundary remain the inherited ones.

Since the induction variable is the displayed finite prefix, no later tail coordinate, quotient stream, selected host limit, countable-choice schedule, or host real equality coordinate is available to the step. □

Theorem 126.395 (StreamName budget tail regular seal lock)**.** *Any finite* $\mathsf{StreamName}^{\uparrow}$ *window routed through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *and* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and* $\mathsf{Real}^{\uparrow}$ *finite-seal data only through the shared budget cap, tail threshold, finite window, dyadic ledger, readback, and seal rows. The locked route exports no quotient streams, selected limits, countable-choice selectors, host real equality, metric completeness, or ambient completeness row.*

Proof. Start with Definition 126.390. It places every regularity row, dyadic tolerance row, and finite Real seal read behind the shared observation-budget surface.

Next apply Definition 126.391 and Definition 126.393. These force the limiter cap, tail threshold, finite StreamName window, dyadic ledger, RegSeqRat readback, and Real seal rows to remain a single sibling route.

Finally use Definition 903.23, Definition 906.3, and Definition 126.337. Their combined non-escape clauses exclude quotient streams, selected limits, countable-choice selectors, host real equality, metric completeness, and ambient completeness from the locked tail-budget seal read. □

Theorem 126.396 (StreamName budget CauchyLimitSeal pullback)**.** *Let an accepted finite* StreamName$^{\uparrow}$ *window be routed through* ObservationBudgetLimiter$^{\uparrow}$ *and* RegularCauchyTailBudget$^{\uparrow}$*, and let a public* Real$^{\uparrow}$ *or completion-facing* CauchyLimitSeal$^{\uparrow}$ *consumer read that route. Then the consumed route factors through the same finite rows:*

ProbeBundle, InBundle, UnaryHistory, ObservationBudgetLimiter$^{\uparrow}$, RegularCauchyTailBudget$^{\uparrow}$, DyadicRatC

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The pullback exposes exactly the budget cap, tail threshold, finite StreamName window, dyadic ledger, RegSeqRat readback, and Real seal already locked by the StreamName budget route. It exports no quotient streams, selected host limits, countable-choice selectors, host real equality, metric completeness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the finite-window budget handoff of Definition 126.390. It places the accepted StreamName window, ProbeBundle ledger, InBundle membership rows, and UnaryHistory observations before any regularity row, dyadic tolerance row, or Real$^{\uparrow}$ seal row is consumed.

Next attach the sibling route of Definition 126.391. The route passes through the ObservationBudgetLimiter$^{\uparrow}$ budget cap and then through the RegularCauchyTailBudget$^{\uparrow}$ tail service before the dyadic ledger, RegSeqRat$^{\uparrow}$ readback, and Real$^{\uparrow}$ seal are exposed.

Now use the tail regular seal lock of Definition 126.395. It identifies the budget cap, tail threshold, finite window, dyadic ledger, readback, seal rows, and visible BHist, hsame, Cont, Pkg, and NameCert provenance as one locked finite route.

Finally pass the locked route through the CauchyLimitSeal$^{\uparrow}$ consumer boundary. The public route boundary of Definition 386.8 confines every public Real$^{\uparrow}$ or completion-facing read to the displayed route packet, and the downstream completion handoff of Definition 386.15 lets a completion consumer pull back only along that same scheduled window, dyadic observation ledger, regular source row, completion row, and seal row. The non-escape clauses from the cited StreamName and CauchyLimitSeal boundaries rule out quotient streams, selected host limits, countable-choice selectors, host real equality, metric completeness, and ambient Real$^{\uparrow}$ completeness. □

Theorem 126.397 (StreamName budget regular readback exhaustion)**.** *Every accepted finite* StreamName$^{\uparrow}$ *window routed through* ObservationBudgetLimiter$^{\uparrow}$ *and* RegularCauchyTailBudget$^{\uparrow}$ *to a public* Real$^{\uparrow}$ *consumer or a completion-facing* CauchyLimitSeal$^{\uparrow}$ *consumer is exhausted by the same finite readback route:*

ProbeBundle, InBundle, UnaryHistory, ObservationBudgetLimiter$^{\uparrow}$, RegularCauchyTailBudget$^{\uparrow}$, DyadicRatC

The exhausted packet carries exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance of the StreamName budget handoff. It exports no quotient stream, selected host limit, countable-choice selector, host real equality, metric completeness, or ambient completeness.*

Proof. Apply Definition 126.396. It already pulls every public Real$^{\uparrow}$ read and every completion-facing CauchyLimitSeal$^{\uparrow}$ read back to the finite StreamName budget route through ProbeBundle, InBundle, UnaryHistory, the budget limiter, the regular tail budget, the dyadic core, the regular sequence row, and the Real seal.

Next use Definition 126.395. The lock identifies the shared budget cap, tail threshold, finite window, dyadic ledger, readback, and seal rows as a single route with its visible BHist, hsame, Cont, Pkg, and NameCert coordinates.

Finally apply Definition 126.337. Its non-escape clause excludes quotient streams, selected host limits, countable-choice selectors, host real equality, metric completeness, and ambient completeness from any such consumer read. Since the pullback and the lock name the same carrier coordinates, the consumer has no remaining route outside the displayed finite packet. □

Theorem 126.398 (StreamName RegSeqRat RealUp dyadic budget correspondence)**.** *Let $\mathcal{W}$ be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and sealed as a finite-window* $\mathsf{Real}^{\uparrow}$ *route. Then the public route determined by $\mathcal{W}$ is exactly the visible dyadic-budget route carried by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger: finite StreamName window rows,* ProbeBundle*,* InBundle*, certified unary observations, DyadicRatCore ledger and radius rows, RegSeqRat readback rows, RealUp finite seal rows, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Conversely, any public route displaying those same rows factors back through the StreamName finite-window budget handoff before a RegSeqRat readback or RealUp finite seal can be consumed. The correspondence exports no quotient stream, selected host limit, countable-choice selector, host real equality, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, start with Definition 126.390. It places the accepted StreamName window, ProbeBundle ledger, InBundle membership rows, and certified unary observations before any regularity, dyadic tolerance, or Real seal row is read.

Next apply Definition 126.395 and Definition 126.397. These identify the budget cap, tail threshold, finite window, dyadic ledger, RegSeqRat readback, Real seal, and visible BHist, hsame, Cont, Pkg, and NameCert provenance as one exhausted finite packet.

Now read that packet through the sibling ledger and endpoint gates. Use Definition 126.105, Definition 757.3, Definition 906.3, and Definition 38.314. Together they force the StreamName finite window, tail-budget service, DyadicRatCore radius ledger, RegSeqRat readback, and RealUp finite seal to expose the same visible dyadic-budget route.

For the reverse direction, suppose a public route already displays exactly those rows. Tail-budget coverage places its radius read inside the finite precision, tail-window, StreamName-window, and dyadic readback segment, while the regular Cauchy observation gate orders that segment before the RegSeqRat and Real seal reads.

The RealUp finite-window exhaustion obligation then leaves no endpoint row outside the displayed finite seal. Repacking the same rows gives the StreamName finite-window budget handoff of Definition 126.390, and the non-escape clauses in the cited results exclude quotient streams, selected host limits, countable-choice selectors, host real equality, metric completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.399 (StreamName ObserverFilter finite-window handoff)**.** *Every accepted finite* $\mathsf{ObserverFilter}^{\uparrow}$ *selected-subhistory handoff to* $\mathsf{StreamName}^{\uparrow}$ *enters the finite-window budget surface only through the selected row S, observer ledger L, componentwise transports R, continuation routes C, provenance P, and local name N. The omitted rows O remain boundary witnesses in the observer-filter ledger. They do not supply RegSeqRat regularity, Real seal data, quotient stream equality, selected host limits, countable-choice selectors, or host real equality to the StreamName consumer.*

Proof. Start with the ObserverFilter handoff theorem Definition 514.4. It identifies the stream-facing route of an accepted filter packet as the selected subhistory through S, L, R, C, P, N.

Apply Definition 514.6. The finite source ledger is exhausted by the selected side and the omitted side, and the omitted side is kept as O rather than exported as StreamName data.

Now use Definition 126.390. Once the selected subhistory is read as a finite $\mathsf{StreamName}^{\uparrow}$ window, every regularity, dyadic tolerance, and $\mathsf{Real}^{\uparrow}$ seal row must pass through the finite observation-budget surface.

Finally apply Definition 126.393. The sibling route carries no coordinates for quotient stream equality, selected host limits, countable-choice selectors, host real equality, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Since O is not part of the selected StreamName source, omitted rows remain boundary ledger witnesses and cannot serve any of those downstream roles. □

126.27 StreamName Real-RegSeq shared-tail budget route

Theorem 126.400 (StreamName Real-RegSeq shared-tail budget route)**.** *Let W_0 and W_1 be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *windows consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal route, or by a shared-tail route that serves both consumers. Suppose the two reads pass through the same* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *coverage row of Definition 757.3 and the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row of Definition 560.11. Then every public seal or readback consumer of the two windows factors through exactly the finite StreamName window rows,* ProbeBundle, InBundle, UnaryHistory*, the dyadic tail-radius ledger, the shared threshold, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *finite-window ledger boundary.*

The route exports no quotient stream, host real equality, selected limit, independent tail oracle, countable-choice selector, or completeness principle.

Proof. Start with the local finite-window budget handoff. By Definition 126.390, every regularity row, dyadic tolerance row, and $\mathsf{Real}^{\uparrow}$ seal row consumed from a finite $\mathsf{StreamName}^{\uparrow}$ window is downstream of ProbeBundle, InBundle, UnaryHistory, the observation-budget surface, and the tail-budget rows.

Use the sibling StreamName route Definition 126.391 to keep this handoff inside the same finite observation-budget and tail-budget chain. The accepted windows W_0 and W_1 therefore present public data only through their displayed finite window rows and their attached transport, continuation, package, and naming rows.

Now read the common dyadic ledger. Apply Definition 757.3 to place each admitted tail-radius read inside the finite precision, tail-window, StreamName-window, and dyadic readback rows. Then apply Definition 757.6 to route the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal boundary after that same ledger row.

Finally attach the shared threshold. The public shared-threshold certificate Definition 560.11 exposes the paired source, window, modulus, dyadic threshold, comparison, transport, continuation, provenance, and local naming rows, while Definition 560.56 fixes the displayed threshold ledger for duplicate transported reads of the same source-window-modulus route.

The carrier inventory is now exhausted. A consumer can project only the finite StreamName windows, ProbeBundle, InBundle, UnaryHistory, the dyadic tail-radius ledger, the shared threshold, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ finite-window ledger boundary, together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows. None of the cited carriers contains a quotient stream, host real equality witness, selected limit, independent tail oracle, countable-choice selector, or completeness principle, so none can escape through the public route. □

Theorem 126.401 (StreamName shared-tail window pullback exactness)**.** *Let W_0 and W_1 be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and let*

$$S = (M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N)$$

be an accepted CauchySharedTailModulus$^{\uparrow}$ *packet whose displayed windows are the same* W_0 *and* W_1*. Every* Real$^{\uparrow}$ *seal read or* RegSeqRat$^{\uparrow}$ *readback pullback from these windows factors through the shared threshold* τ*, the dyadic ledger* D*, the paired windows* W_0, W_1*, the paired readbacks* Q_0, Q_1*, and the seal route* E, H, C, P, N *displayed by* S*.*

Replacing one side by a private threshold, quotient stream, selected limit, host real equality, or independent inverse-budget route is not a read of the packet S*.*

Proof. Apply Definition 126.400. It places the two finite windows on the common StreamName budget path through the dyadic tail-radius ledger, the shared threshold, the RegSeqRat$^{\uparrow}$ readback boundary, and the Real$^{\uparrow}$ finite-window seal boundary.

Now read the CauchySharedTailModulus$^{\uparrow}$ packet through Definition 983.4. It identifies any seal-facing consumer with the rows $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$. The paired-window service of Definition 983.6 keeps the two windows attached to the same threshold and dyadic ledger.

Finally compare reciprocal-window consumers only through the Definition 703.6. That comparison supplies no private threshold, quotient stream, selected limit, host real equality, or independent inverse-budget route, so every displayed pullback remains the read of the single packet S. □

Theorem 126.402 (StreamName shared-tail terminal face noncollapse)**.** *Every shared-tail packet accepted by the* StreamName$^{\uparrow}$ *route reports its terminal L10 read as four separate local faces: the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* StreamName$^{\uparrow}$ *finite window, the* RegSeqRat$^{\uparrow}$ *readback and classifier row, and the* Real$^{\uparrow}$ *finite-window seal. The packet transports those faces only by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The terminal read cannot be collapsed into quotient stream equality, host real equality, a selected limit, countable choice, ambient completeness, route-level formalstatus*, or an aggregate bridge row. None of these substitutes is a local face of the accepted shared-tail packet.*

Proof. Start with Definition 126.400. It places the accepted windows on the common StreamName route through the dyadic tail-radius ledger, the shared threshold, the RegSeqRat$^{\uparrow}$ readback boundary, and the Real$^{\uparrow}$ finite-window seal boundary.

Next apply Definition 126.401. Any seal read or readback pulled from the shared windows must read the single displayed packet through its threshold, dyadic ledger, paired windows, paired readbacks, and seal rows. Hence the terminal read is already local to the displayed packet.

Use Definition 301.173 to identify the same endpoint as the product of the four local status faces: DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$. The only shared transport permitted in that product is the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply Definition 38.536 together with the two-axis discipline of chapter 9203. Closure rows, verification rows, and bridge rows are read objectwise on the corresponding face; they do not combine into a route-level witness and cannot replace a missing terminal face. The excluded quotient, host-equality, selected-limit, choice, and completeness data are therefore outside the accepted shared-tail terminal read. □

126.28 StreamName Common-Refinement Terminal Real Seal

Theorem 126.403 (StreamName common-refinement terminal Real seal)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be finite-window* StreamName$^{\uparrow}$ *packages over the same certified endpoint maps, and let* $\mathcal{R}$ *be a certified common-refinement package restricting to both selected windows. Then every downstream*

$\mathsf{RegSeqRat}^{\uparrow}$ *handoff followed by a finite-window* $\mathsf{Real}^{\uparrow}$ *seal read factors through the single terminal finite tuple displayed by* $\mathcal{R}$*:*

$(\mathsf{RatStreamNameFiniteWindowClassifier}_{\mathcal{R}}, \mathsf{ProbeBundle}_{\mathcal{R}}, \mathsf{InBundle}_{\mathcal{R}}, \mathsf{UnaryHistory}_{\mathcal{R}}, \mathsf{RatHistoryClassifier}_{\mathcal{R}}, \mathsf{DyadicLed}$

The factorization reads the finite-window classifier, the ProbeBundle *row,* InBundle *membership rows,* UnaryHistory *observations, pointwise* RatHistoryClassifier *rows, dyadic ledger rows, displayed* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows. It does not identify quotient streams, assert host sequence equality, select limits, use ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or import metric-completeness data.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameCommonRefinementTerminalRealSeal`
↪ variant `BEDC.Derived.StreamNameUp.StreamName common refinement terminal real seal`

Proof. Begin with Definition 126.240. It pins the two selected windows to the common-refinement source tuple of $\mathcal{R}$ and keeps the downstream $\mathsf{RegSeqRat}^{\uparrow}$ and finite $\mathsf{Real}^{\uparrow}$ handoff reads on that public packet.

Next apply Definition 126.132. The finite seal read exposes only the StreamName window, pointwise rational rows, the finite Real seal boundary, componentwise transport, continuation routes, provenance, and local names.

Use Definition 126.137 to place the two selected windows in one common finite-window lattice. The classifier observations, bundle rows, membership rows, unary-history observations, dyadic ledger, transports, routes, provenance, and names are therefore read from the same finite packet.

Finally apply Definition 126.261. The meet packet is terminal for the RegSeqRat readback and finite Real seal handoff, so there is no carrier coordinate left for quotient equality, host sequence equality, selected limits, ambient Real completeness, or metric-completeness rows. □

Theorem 126.404 (StreamName Real-RegSeq terminal common-window lock)**.** *Let* $\mathcal{R}$ *be a common-refinement* $\mathsf{StreamName}^{\uparrow}$ *package whose selected finite window is consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and sealed as a finite-window* $\mathsf{Real}^{\uparrow}$ *row. Then the public consumer surface is the single terminal packet consisting of the finite* BHist *window,* RatHistoryClassifier *observations,* ProbeBundle *rows,* InBundle *rows,* UnaryHistory *observations, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the induced* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The packet exports no quotient stream, host sequence equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, independent* $\mathsf{Real}^{\uparrow}$ *carrier, or metric-completeness row.*

Proof. First apply Definition 126.403. It reduces every selected-window RegSeqRat handoff followed by a finite Real seal read to the finite tuple carried by the common-refinement package $\mathcal{R}$.

Next use Definition 126.257. The selected finite window is the meet-facing packet seen by both the StreamName observations and the downstream RegSeqRat consumer, so the classifier, bundle, membership, unary-history, dyadic, transport, continuation, provenance, and naming rows all come from one finite surface.

Apply Definition 126.356 to identify the StreamName window, RegSeqRat readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and Real seal boundary as a common finite pullback. Then use Definition 126.358 to make the Real-facing seal read deterministic on that pullback.

Finally apply Definition 126.140. The downstream consumer therefore factors through the listed terminal packet and has no exported coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, independent Real carriers, or metric-completeness data. □

Theorem 126.405 (StreamName-RegSeqRat-Real common-refinement seal pullback)**.** *Any common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window package consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and sealed as a* $\mathsf{Real}^{\uparrow}$ *finite-window row factors through exactly one finite pullback. The pullback consists of the StreamName window, the RegSeqRat readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It exports no quotient stream, host sequence equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, or metric-completeness row.*

Proof. First apply Definition 126.403. The common-refinement package supplies the finite StreamName observations, bundle membership, unary history, dyadic ledger, transports, routes, provenance, and local names used by the downstream seal read.

Next apply Definition 126.404. The consumer surface is locked to the same selected finite window, RegSeqRat readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and induced $\mathsf{Real}^{\uparrow}$ seal row.

Use Definition 126.356 to read these four sibling rows as one finite pullback. Then apply Definition 126.358 to make the seal-facing route unique on that pullback.

The remaining displayed rows are exactly the BHist observations, hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows already retained by the common-refinement packet. Since the packet contains no quotient-stream coordinate, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ completeness, or metric-completeness row, none of those claims is exported by the pullback. □

Theorem 126.406 (StreamName common-refinement Real seal uniqueness)**.** *Let* $\mathcal{R}_0$ *and* $\mathcal{R}_1$ *be common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packages over the same certified endpoint maps. Suppose that both packages expose the same terminal coordinates listed in Definition 126.403: the finite-window classifier,* $\mathsf{ProbeBundle}$ *row,* $\mathsf{InBundle}$ *membership rows,* $\mathsf{UnaryHistory}$ *observations,* $\mathsf{RatHistoryClassifier}$ *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. If both packages feed the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then every finite-window* $\mathsf{Real}^{\uparrow}$ *seal read produced from* $\mathcal{R}_0$ *and every finite-window* $\mathsf{Real}^{\uparrow}$ *seal read produced from* $\mathcal{R}_1$ *are the same public seal row under those displayed* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *coordinates. This equality is only the finite public seal row exposed by the common-refinement packet: it does not assert quotient-stream equality, host sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, or metric-completeness rows.*

Proof. Apply Definition 126.403 to both $\mathcal{R}_0$ and $\mathcal{R}_1$. Since the exposed terminal coordinates are assumed identical, both downstream reads factor through the same finite-window classifier, bundle rows, membership rows, unary-history observations, rational-history rows, dyadic ledger, transports, routes, provenance, and local names.

Next use Definition 126.404. The accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger are therefore locked to the same terminal common window for both packages, so neither package contributes an independent Real-facing carrier coordinate.

Now apply Definition 126.405. The shared terminal window, shared readback, and shared dyadic ledger place both finite-window $\mathsf{Real}^{\uparrow}$ seal reads on the same finite pullback with the displayed hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally use Definition 126.358. Determinacy on that pullback identifies the two seal reads as the same public seal row. The preceding factorization lists all exported coordinates, so the conclusion leaves no coordinate for quotient-stream equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, or metric-completeness rows. □

Theorem 126.407 (StreamName common-refinement terminal subcover exactness)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets over certified endpoint maps. Suppose that each retained packet is consumed by the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and a finite-window* $\mathsf{Real}^{\uparrow}$ *seal read through the terminal surface of Definition 126.403. For every finite subfamily $J \subseteq I$, restrict the visible windows, classifier rows,* ProbeBundle *rows,* InBundle *membership rows,* UnaryHistory *observations,* RatHistoryClassifier *rows, dyadic ledger rows, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows to the indices in J.*

Then the restriction of the terminal Real-seal packet on I to J is exactly the terminal packet obtained by assembling the common-refinement windows indexed by J alone. The restricted packet has precisely the inherited $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, and it introduces no quotient streams, host sequence equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or independent seal coordinates.*

Proof. Restrict the finite family along $J \subseteq I$. The projection keeps exactly the selected StreamName windows, classifier rows, bundle rows, membership rows, unary-history observations, rational-history rows, dyadic ledger rows, transports, routes, provenance, and local names indexed by J.

Apply Definition 126.403 to the restricted family. Each retained consumer still reaches the same kind of terminal common-refinement packet before its $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and finite $\mathsf{Real}^{\uparrow}$ seal row are inspected.

Next use Definition 126.404. The restricted readback and seal rows are locked to the retained StreamName windows, so the subfamily cannot acquire a separate Real-facing carrier or seal coordinate.

Apply Definition 126.405. The retained StreamName window, accepted RegSeqRat readback, dyadic ledger, and Real seal row form the same finite pullback whether they are read as a restriction of the original family or assembled from the subfamily alone.

Finally use Definition 126.406. The restricted packet and the packet assembled from J have identical displayed terminal coordinates, hence they are the same public finite-window seal packet. The row inventory leaves no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, or independent seal coordinates. □

Theorem 126.408 (StreamName common-refinement terminal subcover associativity)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets whose retained consumers pass through the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal row of Definition 126.403. For nested finite subfamilies $K \subseteq J \subseteq I$, restrict the terminal packet on I first to J and then to K, retaining only the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *finite* BHist *observation,* hsame, Cont, Pkg, *and local* NameCert *rows. Then this two-step restriction is the same public finite-window packet as the direct restriction of the terminal packet on I to K.*

The associativity claim is only about those displayed public rows. It does not identify quotient streams, assert host sequence equality, select limits, use ambient $\mathsf{Real}^{\uparrow}$ *completeness, import metric-completeness data, or introduce independent seal coordinates.*

Proof. Apply Definition 126.407 to the inclusion $J \subseteq I$. The first restriction is exactly the terminal packet assembled from the common-refinement windows indexed by J, with only the displayed StreamName, RegSeqRat, dyadic, Real, history, transport, route, provenance, and naming rows.

Apply the same theorem to the inclusion $K \subseteq J$. The second restriction therefore has precisely the terminal coordinates obtained by assembling the windows indexed by K inside the already restricted J-packet.

Apply Definition 126.407 once more to the direct inclusion $K \subseteq I$. The direct packet and the two-step packet expose identical terminal coordinate lists, so Definition 126.406 identifies them as the same public finite-window seal packet. The coordinate inventory contains no slot for quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, or independent seal coordinates. □

Theorem 126.409 (StreamName common-refinement objectwise status stability)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets whose terminal packets feed the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal row. For nested finite subfamilies $K \subseteq J \subseteq I$, restrict the terminal packet on I first to J and then to K, retaining the displayed finite* BHist *observations,* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *rows, and the four local objectwise status records for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Then the restricted terminal packet preserves exactly those four local objectwise closurestatus, formalstatus, leantarget, and bridgestatus records. Equivalently, the terminal packet restricted to K satisfies the Phase $\mathsf{Real}^{\uparrow}$ *and Completion objectwise exit read exactly when the unrestricted terminal packet satisfies that read after projection to K. The claim excludes quotient stream equality, host sequence equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and standard bridge rows.*

Proof. First apply Definition 126.407. Restriction along a finite subfamily keeps precisely the displayed StreamName window, RegSeqRat readback, dyadic ledger, Real seal row, finite BHist observation, hsame, Cont, Pkg, and local NameCert rows.

Next apply Definition 126.408. The two-step restriction through J and then K has the same public terminal packet as the direct restriction from I to K, so the status read is independent of the route used to reach the retained finite subfamily.

Use Definition 126.405 and Definition 126.404. These two results keep the $\mathsf{StreamName}^{\uparrow}$ window, accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and finite $\mathsf{Real}^{\uparrow}$ seal row on the same finite pullback after restriction.

Finally invoke Definition 38.792. The Phase Real and Completion exit criterion reads exactly the four local face records together with the displayed finite support rows. Since the preceding restrictions preserve those rows objectwise, the projected packet and the unrestricted packet projected to K satisfy the exit read together, and no aggregate formalstatus row, aggregate bridge row, standard bridge row, quotient stream equality, host sequence equality, selected limit, ambient Real completeness, or metric-completeness datum is introduced. □

Theorem 126.410 (StreamName common-refinement objectwise status sharpness)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets whose terminal packets feed the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the finite-window* $\mathsf{Real}^{\uparrow}$ *seal row. For every finite subfamily $K \subseteq I$, the restriction of the common-refinement terminal packet to K is accepted by the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion objectwise exit read if and only if the restricted packet preserves exactly the four local face records for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the displayed finite* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *support rows.*

The equivalence is sharp at that objectwise surface. It does not use or export an aggregate formalstatus row, an aggregate bridge row, a standard bridge substitute, quotient stream equality, host sequence equality, selected limits, countable choice, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For preservation, apply Definition 126.409. It already shows that finite-subfamily restriction keeps the four local objectwise closurestatus, formalstatus, leantarget, and bridgestatus records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, along with the displayed finite support rows.

The route used to reach the retained subfamily contributes no extra status datum. Use Definition 126.407 to identify the restriction of the ambient terminal packet with the packet assembled from the subfamily alone, and then use Definition 126.408 to remove dependence on any intermediate finite subfamily.

For the converse readback, start with the four preserved local face records and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Apply Definition 126.510 to read these rows as exactly the current Phase Real and Completion objectwise threshold. Then apply Definition 301.66 to place the RegSeqRat terminal read on the same four-face support surface.

Combining the two directions gives the stated equivalence for the restricted common-refinement terminal packet. Since every step reads only the four local face records and the listed finite support rows, the certificate has no coordinate for aggregate formalstatus data, aggregate bridge data, standard bridge substitutes, quotient stream equality, host sequence equality, selected limits, countable choice, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.411 (StreamName CauchyLimitSeal forward lattice)**.** *Any common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and then by* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *factors through the StreamName window, the RegSeqRat readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the finite-window seal surface of chapter 386. The forward route carries only displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient streams, host sequence equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or metric-completeness data.*

Proof. First apply Definition 126.403. It places the common-refinement consumer read on the finite StreamName packet before any RegSeqRat or Real-facing row is inspected.

Next apply Definition 126.404 and Definition 126.405. These two results identify the same selected finite window with the RegSeqRat readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the $\mathsf{Real}^{\uparrow}$ seal row, and they make that pullback terminal for the common-refinement packet.

Finally route the seal-facing consumer through the CauchyLimitSeal sibling chapter chapter 386. Its finite-window seal surface consumes the displayed StreamName schedule, RegSeqRat source, dyadic ledger, Real seal, transport, route, provenance, and naming rows; it does not add quotient streams, host sequence equality, selected limits, ambient Real completeness, or metric-completeness data. □

Theorem 126.412 (StreamName CauchyLimitSeal reverse source exhaustion)**.** *Whenever a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer route enters the* $\mathsf{StreamName}^{\uparrow}$ *common-refinement terminal packet through the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *finite-window seal surface, every public row contributed by that route is already one of the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The route exports no additional hidden tail witness, quotient stream, host sequence equality, selected external limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completion row, or standard bridge row.*

Proof. Start with the forward factorization in Definition 126.411. It places the StreamName-to-CauchyLimitSeal route on the selected finite window, the RegSeqRat readback, the dyadic ledger, the Real seal row, and the displayed transport, route, provenance, and naming rows.

On the CauchyLimitSeal side, apply Definition 386.116. Every sealed-limit read must pass through the scheduled $\mathsf{StreamName}^{\uparrow}$ window before the seal row is consumed. Then use Definition 386.121; its sibling-lattice statement keeps the route on the displayed finite seal surface and rules out a host-level substitute.

It remains to exhaust the coordinates of the two public packets. The StreamName terminal packet lists only the selected window, RegSeqRat readback, dyadic ledger, Real seal, BHist observations, hsame transports, Cont routes, Pkg provenance, and local NameCert rows, while the CauchyLimitSeal surface contributes only its displayed finite-window seal coordinates. Their common coordinates are exactly the rows named in the statement, so there is no remaining slot for a hidden tail witness, quotient stream, host sequence equality, selected external limit, ambient Real completeness row, metric-completion row, or standard bridge row. □

Theorem 126.413 (StreamName common-refinement terminal status pullback)**.** *Let $\mathcal{R}$ be a certified common-refinement* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet whose terminal read is consumed by the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion route. Then the terminal read satisfies the face-local status readback exactly when the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

retain their own local `closurestatus`*,* `formalstatus`*,* `leantarget`*, and* `bridgestatus` *rows over the same displayed support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The pullback is only the four-face common-refinement route. It excludes aggregate status rows, quotient streams, selected limits, countable choice, metric-completeness data, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 126.403. The certified common-refinement packet places every terminal Real-facing read on the single finite surface carrying the StreamName window, RegSeqRat readback, dyadic ledger, Real seal row, displayed BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Next apply Definition 126.409. Restricting the common-refinement route preserves exactly the four local objectwise status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ over that same support, without creating an aggregate status coordinate.

For the forward direction, use Definition 126.425. A Phase Real and Completion terminal read obtains its status rows from the concrete sibling faces themselves, so any accepted read through $\mathcal{R}$ must read the four local `closurestatus`, `formalstatus`, `leantarget`, and `bridgestatus` rows on the displayed support.

For the reverse direction, assume those four face-local rows are retained over the displayed BHist, hsame, Cont, Pkg, and NameCert support. Apply Definition 126.427 to repack precisely those rows as the terminal Phase Real and Completion status read. The preceding terminal seal and objectwise-stability results keep the read on the common-refinement packet, so there is no coordinate from which to form aggregate status rows, quotient streams, selected limits, countable choice, metric-completeness data, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.414 (StreamName CauchyLimitSeal downstream orphan route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet used by a completion-facing seal consumer is consumed through the sibling* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route of chapter 386 after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and*

$\mathsf{Real}^{\uparrow}$ *seal rows are displayed. The route carries only the selected StreamName window, regular-rational readback, dyadic ledger, Real seal, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

It exports no quotient streams, selected host limits, host real equality, hidden tail witnesses, ambient completeness, or metric-completion rows.

Proof. Start with Definition 126.411. It places every StreamName-to-CauchyLimitSeal consumer read on the selected finite window, the RegSeqRat readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{Real}^{\uparrow}$ seal row, and the displayed transport, route, provenance, and naming rows.

Use reverse source exhaustion, Definition 126.412. A route entering the StreamName common-refinement packet from the CauchyLimitSeal finite-window seal surface contributes no additional row beyond the selected window, readback, dyadic ledger, Real seal, and local BEDC routing coordinates.

Finally apply the sibling CauchyLimitSeal route, Definition 386.118 and Definition 386.49. These rows force the completion-facing consumer through the displayed StreamName schedule before the seal is consumed, and they leave no carrier coordinate for quotient streams, selected host limits, host real equality, hidden tail witnesses, ambient completeness, or metric-completion data. □

126.29 StreamName Open-Phase Terminal Route

Theorem 126.415 (StreamName open-phase terminal route)**.** *Every Phase Real and Completion finite-window* $\mathsf{StreamName}^{\uparrow}$ *consumer whose route reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and then the* $\mathsf{Real}^{\uparrow}$ *seal surface of chapter 38 factors through one terminal finite packet. The packet consists of the StreamName window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of chapter 291, the* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The boundary is a finite BEDC handoff route. It exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. First apply Definition 126.403. It places the StreamName finite-window consumer on the common-refinement packet carrying the visible window, observation rows, transports, routes, provenance, and local names.

Next apply Definition 126.404. The route from the StreamName window into the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the finite $\mathsf{Real}^{\uparrow}$ seal is locked to a single terminal consumer surface.

Now use Definition 126.356. The StreamName window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal row of chapter 38 occupy one finite pullback rather than independent carrier routes.

Finally apply Definition 126.358. The seal-facing consumer is determined by that pullback, so the Phase Real and Completion boundary reads only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates already present in the terminal packet. □

Theorem 126.416 (StreamName open-phase terminal non-escape)**.** *Every Phase Real and Completion consumer satisfying the terminal finite packet route of Definition 126.415 receives only the StreamName window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It receives no quotient stream, no host sequence equality, no selected limit, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Proof. First invoke Definition 126.415. It identifies the consumer route with the terminal finite packet and names the StreamName window, readback, ledger, seal row, observations, transports, displayed routes, provenance, and local naming rows that the packet exposes.

Next apply Definition 126.403 and Definition 126.404. These results place the Phase Real and Completion read on the common-refinement terminal surface and lock the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal access to the same finite window.

Use Definition 126.356. The StreamName window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row form one finite pullback, so a consumer read has no independent carrier route outside that packet.

Finally apply Definition 126.358. Pullback seal determinacy exhausts the seal-facing read by the displayed packet coordinates, excluding quotient streams, host sequence equality, selected limits, ambient completeness rows, metric-completeness data, and standard bridge rows. □

Theorem 126.417 (StreamName open-phase four-face lock)**.** *Every open-phase terminal consumer for the Phase Real and Completion boundary factors through exactly the ordered four-face route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}.$$

The only transport available along this route is the displayed BHist *observation data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The route exports no quotient stream, no host sequence equality, no selected limit, no ambient completeness row, no metric-completeness datum, and no standard bridge row.*

Proof. First invoke Definition 126.415. It places every Phase Real and Completion consumer on the terminal finite packet and names the StreamName window, readback, ledger, seal row, observations, transports, routes, provenance, and local naming rows.

Next apply Definition 126.416. This excludes any consumer coordinate outside the terminal packet, including quotient streams, host sequence equality, selected limits, ambient completeness rows, metric-completeness data, and bridge rows.

Use Definition 126.356. The StreamName window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal occupy one finite pullback, so they are consumed as one ordered route rather than four independent faces.

Finally apply Definition 126.358. The seal-facing read is determined by that pullback, and the only remaining support rows are the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates. □

Theorem 126.418 (StreamName terminal four-face exit completeness)**.** *An accepted Phase Real and Completion terminal read through* $\mathsf{StreamName}^{\uparrow}$ *is exit-complete exactly when it exposes the four local faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *together with the displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *support rows of the terminal packet. Conversely, those four faces and support rows determine the terminal exit packet for the Phase Real and Completion boundary.*

The exit packet exports no quotient stream, no host sequence equality, no selected limit, no countable-choice coordinate, no metric-completeness datum, no aggregate formalstatus row, no aggregate bridge row, no standard bridge row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply Definition 126.415. It places the accepted Phase Real and Completion read on the terminal finite packet and lists the StreamName window, the readback surface, the dyadic ledger, the Real seal row, observations, transports, routes, provenance, and local names.

Next use Definition 126.416. This removes every coordinate outside the terminal packet, including quotient streams, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, and bridge rows. The same displayed route leaves no countable-choice or aggregate status coordinate because the packet records only local BEDC support rows.

Now apply Definition 126.417. The terminal packet is therefore read in the ordered four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, rather than as a claim about independent carrier surfaces.

Finally invoke Definition 126.358. Once the four local faces and displayed BHist, hsame, Cont, Pkg, and NameCert rows are fixed, seal determinacy leaves one terminal exit packet. The roadmap two-axis discipline keeps these local face status records separate from the construction route, so the theorem records the exit surface without asserting ambient $\mathsf{Real}^{\uparrow}$ completeness or an aggregate verification or bridge row. □

126.30 StreamName Open-Phase Terminal Exactness

Theorem 126.419 (StreamName open-phase terminal seal exactness)**.** *Every Phase Real and Completion seal read available from an open-phase terminal* $\mathsf{StreamName}^{\uparrow}$ *consumer is available exactly through the four-face route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}.$$

The supporting rows are the displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, local* NameCert *rows, the StreamName finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The seal read exports no quotient stream, no host sequence equality, no selected limit, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Proof. Apply Definition 126.415. It places the Phase Real and Completion seal read on the terminal finite packet carrying the StreamName window, the DyadicRatCore ledger, the RegSeqRat readback, the RealUp seal row, observations, transports, routes, provenance, and local names.

Next use Definition 126.615 and Definition 126.488. The open-phase consumer can reach the seal only through the ordered four-face route displayed in the statement.

Finally apply Definition 126.358. The seal-facing read is determined by those four faces and their displayed support rows, so there is no second route through quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, or standard bridge rows. □

Theorem 126.420 (StreamName open-phase terminal ledger coverage)**.** *Every public coordinate of the open-phase terminal route for a Phase Real and Completion consumer is one of the displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, local* NameCert *rows, the StreamName window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or the* $\mathsf{Real}^{\uparrow}$ *seal. In particular, the terminal ledger contains no quotient stream coordinate, no host sequence equality coordinate, no selected-limit coordinate, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate, no metric-completeness coordinate, and no standard bridge coordinate.*

Proof. First apply Definition 126.416. It exhausts the consumer surface by the terminal packet rows and excludes quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, and standard bridge rows.

Now use Definition 126.403. The common-refinement terminal seal theorem supplies the finite public inventory: the displayed window observations, classifier rows, bundle rows, dyadic ledger rows, transports, routes, provenance, and local names that feed the RegSeqRat readback and RealUp seal.

Combining the two inventories leaves exactly the rows named in the statement. Since every public coordinate is already accounted for in the terminal packet and the common-refinement seal surface, no hidden quotient, host sequence, selected-limit, completeness, metric-completeness, or standard bridge coordinate remains. □

Theorem 126.421 (StreamName open-phase terminal readback determinacy)**.** *Let two Phase Real and Completion consumers read the same open-phase terminal route. If they have the same StreamName window face, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the same* $\mathsf{Real}^{\uparrow}$ *seal face, and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *transport and provenance rows, then they determine the same terminal packet. The determined packet is only the finite BEDC packet of the four displayed faces and their support rows; it is not a quotient stream, host sequence equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness object, metric-completeness datum, or standard bridge row.*

Proof. Apply Definition 126.358 to the two consumer reads. Equality of the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and displayed support rows gives the same seal-facing finite pullback surface.

Next apply Definition 126.420. Every public coordinate of either terminal packet is one of the four displayed faces or one of the listed transport, route, provenance, observation, and naming rows.

The two packets therefore have the same complete public inventory. Since the coverage theorem leaves no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, or standard bridge rows, there is no remaining source from which two distinct terminal packets could be read. □

Theorem 126.422 (StreamName Real-RegSeqRat terminal factorization)**.** *Any Phase Real and Completion consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *seal through* $\mathsf{StreamName}^{\uparrow}$ *factors uniquely through the ordered four-face route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}$$

with displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization is the consumer contract for the terminal seal read: no quotient stream, host sequence equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness row, or bridge row can serve as an alternate factor.*

Proof. Apply Definition 126.419. It supplies existence of the ordered route from the StreamName window through the DyadicRatCore ledger and RegSeqRat readback to the RealUp seal.

Use Definition 126.615 and Definition 126.488 to identify that route as the L10 source order.

Next apply Definition 126.420. The coverage theorem exhausts the terminal ledger by the four faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Use Definition 126.421 for uniqueness. Once the four faces and support rows agree, two consumer reads determine the same finite terminal packet.

Finally use Definition 126.423. The boundary theorem rules out quotient stream, host sequence equality, selected limit, ambient Real completeness, metric-completeness, and bridge rows as alternate routes, so the factorization is the unique consumer contract. □

Theorem 126.423 (StreamName-RegSeqRat-RealUp terminal seal boundary)**.** *Any Phase Real and Completion consumer that reaches a quotient-refusal or seal-facing boundary through* $\mathsf{StreamName}^{\uparrow}$ *must expose exactly the terminal four-face route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}$$

together with the displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows before any refusal ledger is read. Consequently the refusal ledger cannot supply quotient streams, host sequence equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. Apply Definition 126.419. It fixes every seal-facing read available through the open-phase terminal $\mathsf{StreamName}^{\uparrow}$ consumer on the four displayed faces and their support rows.

Next use Definition 126.420. The terminal packet has no public coordinate outside the StreamName finite window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and the displayed observation, transport, route, provenance, and local naming rows.

Apply Definition 126.421. Once those rows agree, the terminal packet is determined as the finite BEDC packet and not as a quotient stream, host sequence equality, selected limit, ambient completeness object, metric-completeness datum, or standard bridge row.

Finally use the QuotientStreamRefusal route-ordering surface of Definition 998.8 together with Definition 126.615. A quotient-refusal consumer may read its refusal ledger only after the ordered StreamName, DyadicRatCore, RegSeqRat, and RealUp route has been exposed. At the StreamName terminal seal boundary, the dyadic ledger supplies the finite tolerance face consumed before the RegSeqRat readback, so the refusal row records non-export rather than adding quotient, equality, limit, completeness, metric, or bridge data. □

Theorem 126.424 (StreamName terminal face status readback)**.** *Any Phase Real and Completion terminal read through* $\mathsf{StreamName}^{\uparrow}$ *reports the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces by reading each face's own local* `closurestatus`*,* `formalstatus`*,* `leantarget`*, and* `bridgestatus` *rows. The readback is facewise: no aggregate route-level status, bridge token, quotient stream, selected limit, ambient completeness, metric-completeness datum, countable-choice row, or host equality can replace one face's local status record.*

Proof. Apply Definition 126.419. It fixes the terminal read on the four-face route from the StreamName window through the DyadicRatCore ledger and RegSeqRat readback to the RealUp seal.

Next apply Definition 126.420. The terminal ledger exposes only the four named faces and their displayed observation, transport, route, provenance, and local naming rows, so the status rows available at the terminal boundary are the local rows attached to those faces.

Use Definition 126.421. Once the four faces and their support rows agree, a consumer read determines the same finite terminal packet; there is no second packet whose aggregate status, bridge token, quotient stream, selected limit, ambient completeness, metric-completeness datum, countable-choice row, or host equality could stand in for a local face record.

Finally apply Definition 126.423. The boundary theorem keeps every seal-facing read inside the displayed terminal route before any refusal ledger is read. The two-axis discipline of section 9203.6 then reads paper closure and Lean verification independently at each local face, so the `closurestatus`, `formalstatus`, `leantarget`, and `bridgestatus` rows remain face-local rather than route-level summaries. □

Theorem 126.425 (StreamName terminal sibling status readback)**.** *Any Phase Real and Completion terminal read through* $\mathsf{StreamName}^{\uparrow}$ *reads its terminal face-status records from the concrete*

sibling chapters chapter 291, chapter 301, and chapter 38, together with the local $\mathsf{StreamName}^{\uparrow}$ *face itself. The read is chapter-local at each face:* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *each contribute their own* `closurestatus`, `formalstatus`, `leantarget`, *and* `bridgestatus` *rows. No aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, ambient completeness row, metric-completeness datum, countable-choice row, or host equality row can replace those concrete sibling records.*

Proof. Apply Definition 126.424. It already forces the terminal read to inspect closure, formal status, Lean target, and bridge status face by face rather than through a route-level summary.

Next apply Definition 126.420. The finite terminal inventory contains only the StreamName window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and their displayed observation, transport, route, provenance, and local naming rows.

Use Definition 126.423 to place that inventory on the terminal route before any refusal ledger is read. The concrete sibling chapters chapter 291, chapter 301, and chapter 38 identify the dyadic ledger, regular readback, and real-seal faces, while the present $\mathsf{StreamName}^{\uparrow}$ chapter supplies the finite-window face.

Therefore the terminal status read has exactly four local chapter sources. Since every excluded coordinate lies outside the finite ledger and outside the terminal boundary, none of those coordinates can serve as a substitute status record for any face. □

Theorem 126.426 (StreamName terminal face-local status strengthening)**.** *Let* $\mathcal{T}$ *be a Phase Real and Completion terminal packet read through* $\mathsf{StreamName}^{\uparrow}$ *along the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Let* F *be one of these four faces. Form* $\mathcal{T}_F^{+}$ *by replacing only the local* `formalstatus`, `leantarget`, *or* `bridgestatus` *row attached to* F *by a stronger local row attached to the same face.*

Assume that the other three face records are unchanged and that the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support rows are unchanged. Then* $\mathcal{T}_F^{+}$ *presents the same four-face terminal packet as* $\mathcal{T}$. *Conversely, any Phase Real and Completion terminal packet with the same face-status readback and with the same displayed support rows can differ from* $\mathcal{T}$ *only by such a face-local strengthening on one of the four status, Lean-target, or bridge-status rows.*

This invariance is confined to local face records. It does not introduce an aggregate formalstatus row, aggregate leantarget row, aggregate bridge row, quotient stream, selected limit, countable-choice row, metric-completeness datum, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, or any replacement for the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Proof. For the forward direction, apply Definition 126.424. The terminal read inspects status, Lean-target, and bridge-status data face by face, so replacing a row attached to F by a stronger row attached to the same face keeps the read on the same local source.

Next apply Definition 126.425. The concrete sibling sources for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the local $\mathsf{StreamName}^{\uparrow}$ source, are unchanged except for the stronger row on F. The terminal packet therefore still has the same four chapter-local face sources.

Now use Definition 126.420. Since the other three face records and all displayed BHist, hsame, Cont, Pkg, and NameCert support rows are fixed, the public inventory of the terminal ledger is the same finite four-face inventory before and after the local strengthening.

For the converse direction, suppose another terminal packet has the same face-status readback and the same displayed support rows. The face-status readback theorem fixes all status-bearing rows to local faces, while the sibling readback theorem fixes the concrete chapter source of each face. Ledger coverage leaves no public coordinate outside those four face records and their displayed support rows.

Finally apply Definition 126.423. The terminal boundary rules out alternate reads through quotient streams, selected limits, countable-choice rows, metric-completeness data, aggregate status rows, aggregate Lean-target rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence any remaining difference between the packets is exactly a stronger local `formalstatus`, `leantarget`, or `bridgestatus` row attached to one of the four faces. □

Theorem 126.427 (StreamName open-phase terminal local threshold equivalence)**.** *Let $\mathcal{T}$ be a Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal packet read through* $\mathsf{StreamName}^{\uparrow}$ *along the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Then $\mathcal{T}$ satisfies the current Phase Real and Completion roadmap exit read if and only if each of the four faces exposes its own local scoped-closure and audit-clean threshold rows, and $\mathcal{T}$ carries only the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The equivalence is terminal and face-local. It does not introduce or consume aggregate `formalstatus`*, aggregate* `leantarget`*, or aggregate* `bridgestatus` *rows, quotient streams, selected limits, countable-choice rows, metric-completeness data, standard bridge rows, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, read the terminal packet by Definition 126.424. The Phase Real and Completion terminal read reaches the four displayed faces by their own local status records, not by an aggregate route credential.

Next apply Definition 126.425. The dyadic, finite-window, regular-readback, and real-seal status records are localized at the concrete $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources. The current exit read can therefore be handed to the Real-side threshold criterion as a four-face objectwise packet.

Use Definition 38.796. It identifies the current Phase Real and Completion exit criterion with exactly the product of the four scoped-closure and audit-clean local threshold rows, supported only by BHist, hsame, Cont, Pkg, and NameCert.

For the reverse direction, assume the four local threshold records and the displayed support rows are present. The Real-side threshold theorem repacks those rows as the current roadmap exit read. Apply Definition 126.426 to transport any same-face status refinement without changing the terminal packet or adding a route-level status row.

Finally apply Definition 126.423. The terminal boundary keeps the packet on the displayed four-face route before any refusal ledger is read, so there is no coordinate from which to form aggregate formalstatus, leantarget, or bridgestatus rows, quotient streams, selected limits, countable-choice rows, metric-completeness data, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.428 (StreamName terminal local threshold pullback)**.** *Let $\mathcal{T}$ and $\mathcal{T}'$ be Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal packets read through* $\mathsf{StreamName}^{\uparrow}$ *along the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Assume that a certified pullback or support-preserving refinement from $\mathcal{T}$ to $\mathcal{T}'$ preserves the $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *face rows and preserves the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then $\mathcal{T}$ satisfies the current Phase Real and Completion local-threshold exit read if and only if $\mathcal{T}'$ satisfies the same exit read.*

This pullback invariance is confined to the four L10 faces and their displayed support. It does not introduce or consume quotient streams, selected limits, countable-choice rows, metric-completeness data, aggregate `formalstatus` *rows, aggregate* `leantarget` *rows, aggregate bridge rows, standard bridge rows, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, apply Definition 126.427 to $\mathcal{T}$. The exit read is exactly the four local scoped-closure and audit-clean threshold rows over the displayed BHist, hsame, Cont, Pkg, and NameCert support.

The certified pullback or refinement preserves each of those four face rows and preserves the displayed support rows. Apply Definition 126.426 face by face to transport any same-face status strengthening without changing the terminal packet or creating a route-level status coordinate. Applying Definition 126.427 to $\mathcal{T}'$ then yields the same current exit read.

For the reverse direction, use the same argument with $\mathcal{T}'$ as the packet satisfying the exit read and $\mathcal{T}$ as the packet obtained by reading the preserved four-face support back along the certified pullback. The equivalence theorem again reduces the exit read to the four local threshold rows, and facewise preservation carries those rows back to the original packet.

It remains to exclude non-face support. Apply Definition 126.420 to both packets: every public coordinate is one of the four displayed faces or one of the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Finally apply Definition 126.423. The terminal boundary rules out quotient streams, selected limits, countable-choice rows, metric-completeness data, aggregate formalstatus rows, aggregate leantarget rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so no outside coordinate can affect the threshold read. □

126.31 StreamName Open-Phase Window Budget Routes

Theorem 126.429 (StreamName open-phase window index nonempty)**.** *Every Phase Real and Completion consumer satisfying the open-phase terminal route of Definition 126.415 reads a nonempty finite window-index surface before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of chapter 291, and the finite* $\mathsf{Real}^{\uparrow}$ *handoff rows of chapter 38. The window-index surface is the one carried by the common-refinement terminal Real seal of Definition 126.403; it consists only of displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The route exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. Start with Definition 126.415. It places the Phase Real and Completion consumer on the terminal finite packet whose visible entries include the StreamName window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the finite $\mathsf{Real}^{\uparrow}$ seal row.

Apply Definition 126.403. The common-refinement packet displays the finite-window classifier, probe rows, membership rows, unary-history observations, rational-history rows, dyadic ledger rows, transports, routes, provenance, and local names before the downstream handoff is read.

Now use Definition 126.384. The accepted finite-window handoff must expose a nonempty index row on that displayed packet, so the Phase Real and Completion route has a finite window-index surface before it can reach the RegSeqRat, DyadicRatCore, or Real-facing rows.

Finally apply Definition 126.416. Since the terminal packet contains no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, or metric-completeness data, none of those objects is exported by the nonempty index route. □

Theorem 126.430 (StreamName open-phase finite-window budget)**.** *Every Phase Real and Completion consumer routed through the terminal StreamName packet has a finite-window budget route whose source is the nonempty StreamName window index, whose middle rows are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and whose target is the finite* $\mathsf{Real}^{\uparrow}$ *seal handoff. The budget is consumed on the common-refinement terminal surface of Definition 126.403 and the open-phase terminal route of Definition 126.415.*

The budget route carries only displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. It exports no quotient stream, no host sequence equality, no selected limit, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. First apply Definition 126.429. The consumer has a finite nonempty StreamName window-index surface before any downstream Real-facing read occurs.

Next use Definition 126.136. Enlarging or transporting the selected finite window changes only the displayed budget rows and keeps the consumer inside the finite StreamName packet.

Apply Definition 126.356. The StreamName window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and finite $\mathsf{Real}^{\uparrow}$ seal row form one finite pullback, so the same window budget supports all three downstream faces.

Finally invoke Definition 126.358 and Definition 126.416. Seal determinacy pins the Real-facing read to that finite pullback, and non-escape removes quotient streams, host sequence equality, selected limits, ambient Real completeness, and metric-completeness data from the exported surface. □

Theorem 126.431 (StreamName open-phase budget source determinacy)**.** *Every Phase Real and Completion consumer that reaches a finite* $\mathsf{Real}^{\uparrow}$ *seal through the StreamName finite-window budget has a unique displayed source route. The route is the ordered four-face handoff*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window budget} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{finite seal.}$$

If a second displayed route has the same BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows, then that route factors through the same finite-window budget packet. Thus the source of the finite* $\mathsf{Real}^{\uparrow}$ *seal is determined by the StreamName budget packet together with its RegSeqRat readback and DyadicRatCore ledger.*

The determinacy statement exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. Begin with Definition 126.430. It places the Phase Real and Completion consumer on a displayed finite packet whose source is the nonempty StreamName window budget and whose downstream faces are the RegSeqRat readback, the DyadicRatCore ledger, and the finite Real seal row.

Apply Definition 126.356. The four displayed faces are not independent rows: they form one finite pullback over the same StreamName window budget packet, with the same BHist observations, hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Now use Definition 126.358. Any second displayed route with the same observations, transports, routes, provenance, and name rows reaches the same finite Real seal only by factoring through that pullback. Hence it factors through the original finite-window budget packet and cannot supply a distinct source for the seal.

Finally invoke Definition 126.416. The terminal packet has no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, or metric-completeness data, so the determinacy claim is entirely a finite displayed-route statement. □

Theorem 126.432 (StreamName open-phase tail-budget stability)**.** *Every Phase Real and Completion consumer routed through the finite-window budget of Definition 126.430 keeps the same nonempty StreamName window-budget source after certified successor and tail reindexing. The transported read still factors through the ordered public route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window budget} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{finite seal.}$$

In particular, every $\mathsf{RegSeqRat}^{\uparrow}$ *tail read obtained from that reindexing remains a read of the original finite budget packet, with transported* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, no host sequence equality, no selected limit, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. Start with Definition 126.430. It supplies the nonempty StreamName window-budget source and places the $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and finite $\mathsf{Real}^{\uparrow}$ seal row on the same common-refinement terminal packet.

Apply Definition 126.7 to the certified successor read. Then use Definition 126.221 and Definition 126.202 to transport the visible tail window back through the common StreamName refinement. These steps change only the selected finite window indices and their displayed transport rows; they do not introduce a new source for the budget.

Now invoke Definition 126.356. The transported StreamName window, the RegSeqRat tail readback, the DyadicRatCore ledger, and the finite Real seal row are still one finite pullback, so the successor-tail read factors through the same four-face route as the original budget.

Finally apply Definition 126.416. Since the terminal packet has no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, or metric-completeness data, the transported tail-budget route cannot export any of those objects. □

Theorem 126.433 (StreamName open-phase tail-budget normal form)**.** *Every Phase Real and Completion consumer obtained from a certified successor or tail reindexing of the StreamName finite-window budget returns to the same ordered normal form*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window budget} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{finite seal.}$$

The reindexed read has no additional source surface: it is supported only by displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows on the finite budget packet.*

Consequently every such successor or tail read factors through the same four-face route as the original open-phase budget. It exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no bridge row.*

Proof. First use Definition 126.430. It supplies the finite budget packet whose visible source is the StreamName window and whose downstream faces are the RegSeqRat readback, the DyadicRatCore ledger, and the finite Real seal.

Next apply Definition 126.432. Certified successor and tail reindexing preserve that packet as the source of the transported read, so the consumer cannot leave the finite-window budget while passing to the reindexed RegSeqRat face.

Now invoke Definition 126.434 and Definition 126.435. The first result orders the public route through the four displayed faces, and the second locks the RegSeqRat readback to the same nonempty StreamName budget source.

Finally use terminal non-escape for the open-phase packet. Since the terminal surface contains only displayed BHist, hsame, Cont, Pkg, and NameCert support rows, there is no coordinate from which a quotient stream, host equality, selected limit, ambient completeness row, metric-completeness datum, or bridge row could be exported. □

Theorem 126.434 (StreamName open-phase Real seal route)**.** *Every Phase Real and Completion handoff that begins at a* $\mathsf{StreamName}^{\uparrow}$ *finite window and ends at a finite* $\mathsf{Real}^{\uparrow}$ *seal row factors through the finite-window budget route of Definition 126.430. The handoff therefore has the ordered public route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window budget} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{finite seal.}$$

The route is the terminal BEDC surface already exposed by Definition 126.415 and Definition 126.403.

It exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. Begin with the finite-window budget route of Definition 126.430. It supplies the nonempty window-index source, the RegSeqRat readback, the DyadicRatCore ledger, and the finite Real seal row on one displayed packet.

Next apply Definition 126.404. The RegSeqRat readback and Real seal access are locked to the same common finite window, with the dyadic ledger read from that packet rather than from an independent carrier.

Use Definition 126.405. The common-refinement seal pullback is unique for the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal row.

Finally apply Definition 126.417. The Phase Real and Completion handoff is exactly the displayed four-face route, and the non-escape clauses already attached to that route exclude quotient streams, host sequence equality, selected limits, ambient Real completeness, and metric-completeness data. □

Theorem 126.435 (StreamName open-phase RegSeqRat budget source lock)**.** *Every Phase Real and Completion consumer whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is obtained through the StreamName finite-window budget must consume the same nonempty StreamName window-index source before it reads any downstream row. Its ordered public route is*

$$\mathsf{StreamName}^{\uparrow}\ \textit{nonempty window index} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{finite seal.}$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *readback is therefore locked as a consumer of the same nonempty source that supports the finite-window budget, and the Real-facing seal row is reached only after the DyadicRatCore ledger has been read from that packet.*

The route exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness datum.*

Proof. Begin with Definition 126.429. It supplies the nonempty StreamName window-index source that every Phase Real and Completion consumer must meet before any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row is read.

Apply Definition 126.430. The budget route places the $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and finite $\mathsf{Real}^{\uparrow}$ seal on the same common-refinement packet whose source is that nonempty window index.

Use Definition 126.432. Certified successor and tail reindexing keep the RegSeqRat readback on the original finite budget packet, so the readback cannot acquire a separate source.

Finally invoke Definition 126.434. The Real handoff is the displayed four-face route through StreamName, RegSeqRat, DyadicRatCore, and the finite Real seal, and its non-escape clauses exclude quotient streams, host sequence equality, selected limits, ambient Real completeness, and metric-completeness data. □

Theorem 126.436 (StreamName open-phase finite-window support row minimality)**.** *Every accepted four-face Phase Real and Completion read through the open StreamName route exposes exactly the StreamName finite-window budget, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the finite* $\mathsf{Real}^{\uparrow}$ *seal, supported by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Dropping any one of those support rows leaves the accepted finite packet without the row needed to read the next face of the route.*

Conversely, adding quotient stream equality, selected limit data, ambient completeness, countable choice, metric-completeness, arithmetic or order package data, or a bridge row leaves the current open-phase route. Such data are not coordinates of the terminal finite packet and cannot be read as part of the accepted Phase Real and Completion surface.

Proof. Start from the finite-window budget theorem, Definition 126.430. It identifies the accepted budget as a finite StreamName packet whose downstream faces are exactly the RegSeqRat readback, the DyadicRatCore ledger, and the finite Real seal.

Use the RegSeqRat budget source lock, Definition 126.435. The readback cannot acquire an independent source, so removing the StreamName budget or any displayed support row needed by that budget breaks the route rather than producing another accepted four-face read.

Now apply the Real seal route theorem, Definition 126.434. The Real-facing handoff is reached only through the ordered StreamName, RegSeqRat, DyadicRatCore, and Real faces, so the ledger and seal rows are necessary parts of the same finite packet.

Finally invoke terminal non-escape. Since the terminal packet has no coordinate for quotient stream equality, selected limits, ambient completeness, countable choice, metric-completeness, arithmetic or order packages, or bridge rows, adding any such row moves the consumer outside the open-phase route instead of extending it. □

126.32 StreamName Open-Phase Dependency Lattice

Theorem 126.437 (StreamName RegSeqRat Real open-phase dependency lattice)**.** *Every Phase Real and Completion consumer of an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *window whose route reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and the* $\mathsf{Real}^{\uparrow}$ *seal surface of chapter 38 factors through the ordered finite route*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}.$$

The $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger is the observation ledger of chapter 291. Along the route the only public coordinates available to the consumer are displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows attached to the four finite surfaces.*

Thus the open-phase lattice exposes a finite BEDC route from StreamName windows into RegSeqRat readback, DyadicRatCore ledger data, and the Real seal. It exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Proof. First apply Definition 126.415. It places the Phase Real and Completion consumer on the accepted terminal finite packet and names the StreamName window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{Real}^{\uparrow}$ seal row, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates.

Next use Definition 126.417. The terminal packet is read as the ordered four-face route from the StreamName window through RegSeqRat readback and the DyadicRatCore ledger into the Real seal. This removes any interpretation in which those faces are independent carrier routes.

Now invoke Definition 126.356. The four faces occupy one finite pullback surface, so a consumer reaching the RegSeqRat and Real surfaces must pass through the same finite StreamName-to-RegSeqRat-to-DyadicRatCore-to-Real packet.

Finally apply Definition 126.359. The consumer can project only the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows on that pullback. Since those rows contain no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, or standard bridge rows, none of those objects is exported by the open-phase lattice. □

Theorem 126.438 (StreamName open-phase face-order no shortcut)**.** *An accepted Phase Real and Completion consumer starting from a finite* $\mathsf{StreamName}^{\uparrow}$ *window reaches the public* $\mathsf{Real}^{\uparrow}$ *seal if and only if its displayed support factors through the ordered four faces*

$$\mathsf{StreamName}^{\uparrow}\ \textit{window} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}\ \textit{ledger} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal}$$

with the shared BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows on that finite route. Any route omitting* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$*, swapping their public read order, or replacing a face by quotient stream equality, host sequence equality, a selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or a standard bridge row lies outside the accepted open-phase lattice.*

Proof. For the forward direction, begin with Definition 126.437. It identifies every accepted consumer that reaches the public $\mathsf{Real}^{\uparrow}$ seal through the open phase with the displayed StreamName, RegSeqRat, DyadicRatCore, and Real faces, and it restricts the visible coordinates to the listed finite BEDC rows.

Next apply Definition 126.415 and Definition 126.417. The terminal route names the finite packet, while the lock reads that packet in the public order $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, then $\mathsf{Real}^{\uparrow}$. Thus the consumer support factors through the ordered four-face route and not through independent carrier faces.

For the reverse direction, use Definition 126.356. A displayed support with those four faces and the shared BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows is one finite pullback packet, so its target row is precisely the public $\mathsf{Real}^{\uparrow}$ seal exposed by the open-phase route.

It remains to exclude shortcuts. Apply Definition 126.359. The consumer projections of the pullback packet are exhausted by the four displayed faces and their shared BEDC coordinates. Omitting either middle face, permuting the RegSeqRat and DyadicRatCore public reads, or substituting quotient equality, host equality, selected limits, completeness data, or a standard bridge row would require a coordinate absent from that exhaustion list. Hence such a route is not an accepted open-phase lattice route. □

Theorem 126.439 (StreamName L10 sibling route)**.** *The* $\mathsf{StreamName}^{\uparrow}$ *public finite-window route is the sibling source consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Every such sibling read passes through finite* BHist *observations, componentwise* hsame *transport, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before a regularity, dyadic observation, or terminal Real seal surface is available.*

Proof. Project the public finite-window route from chapter 126. The route exposes the finite StreamName window and its unary history observations as the common source row.

The three sibling chapters consume that source in distinct ways. The $\mathsf{RegSeqRat}^{\uparrow}$ chapter reads it as the regular-rational source schedule, chapter 291 reads it as the window support for dyadic observations, and chapter 38 reads it as the terminal Real seal source.

Because each sibling read is routed through the displayed BHist, hsame, Cont, Pkg, and NameCert rows, the StreamName route is a citeable L10 sibling dependency rather than a hidden host sequence or completed stream. □

Theorem 126.440 (StreamName Cauchy window transducer source route)**.** *The finite-window row exported by* $\mathsf{StreamName}^{\uparrow}$ *is the source row* S *consumed by every accepted* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet. The route passes through chapter 1020 only as displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data.*

Proof. Apply Definition 126.439. It identifies the StreamName public finite-window route as the sibling source consumed by RegSeqRat, DyadicRatCore, and Real-facing rows before any terminal seal can be read.

Now inspect Definition 1020.1. The first analytic coordinate of a transducer packet is exactly the StreamName source row S, and every later row is routed through $D, W, R, E, L, H, C, P, N$.

Coordinate exhaustion gives the exclusion. Since the transducer carrier has no host sequence coordinate and no completed stream coordinate, the StreamName contribution to the packet is precisely the displayed finite-window row transported through BHist, hsame, Cont, Pkg, and NameCert data. □

Theorem 126.441 (StreamName Real-completion sibling route)**.** *Every finite-window* $\mathsf{StreamName}^{\uparrow}$ *read consumed by the Real and Completion route factors through the sibling surfaces chapter 301, chapter 291, and chapter 38 before it can become* $\mathsf{Real}^{\uparrow}$ *seal data. The route exposes finite-window transport and Real handoff rows only; it does not turn a stream window into quotient stream equality or host real equality.*

Proof. Start with the existing Real handoff rows for $\mathsf{StreamName}^{\uparrow}$. They expose a finite-window read before any Real-facing consumer can inspect the stream data.

Apply the finite-window transport rows already carried by the StreamName child files. These rows keep the window index, transport budget, and common refinement data inside the displayed $\mathsf{StreamName}^{\uparrow}$ packet.

The sibling route then passes through chapter 301, chapter 291, and chapter 38. The regular-rational row supplies the finite sequence handoff, the dyadic row supplies the tolerance ledger, and the Real row supplies only the terminal seal data. □

Theorem 126.442 (StreamName DyadicRatCore window anchor)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *finite-window read consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or by a* $\mathsf{Real}^{\uparrow}$ *seal route factors through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and tolerance rows before it becomes a regular-rational handoff. The concrete route is the StreamName finite-window surface, the DyadicRatCore observation ledger of chapter 291, the RegSeqRat handoff surface of chapter 301, and the Real seal boundary of chapter 38; it exports no quotient stream equality, selected host limit, countable-choice selector, or ambient completeness row.*

Proof. Project the accepted StreamName read to its finite-window carrier rows. Those rows are the only visible source available to a downstream regular-rational or Real seal consumer.

Route the window through the DyadicRatCore sibling first. The observation and tolerance ledger of chapter 291 reads the same finite window as displayed BHist, componentwise hsame, finite Cont, Pkg, and local NameCert data, so the dyadic anchor is still a finite certificate row rather than a quotient-stream comparison.

Pass the anchored window to RegSeqRat and then to Real. The handoff surface of chapter 301 consumes the dyadic-window rows as regular rational data, and the seal boundary of chapter 38 receives only that displayed handoff. Hence every such StreamName read factors through DyadicRatCore observation and tolerance before becoming the regular-rational input seen by the Real route. □

The StreamName$^{\uparrow}$ finite-window ledger consumes DyadicRatCore$^{\uparrow}$ point certificates through chapter 291 and RegSeqRat$^{\uparrow}$ regular sequence windows through chapter 301, so its Real-facing route records the open L10 dependency lattice without introducing a second point carrier or quotient stream equality.

Definition 126.443 (StreamName Dyadic-RegSeq route packet)**.** *The* StreamName$^{\uparrow}$ *Dyadic-RegSeq route packet is the finite certificate packet read by a* StreamName$^{\uparrow}$ *finite-window consumer before it reaches a* RegSeqRat$^{\uparrow}$ *readback or a* Real$^{\uparrow}$ *seal boundary. Its coordinates are:*

1. *the StreamName finite-window rows;*
2. *the* DyadicRatCore$^{\uparrow}$ *observation and tolerance ledger of chapter 291;*
3. *the* RegSeqRat$^{\uparrow}$ *regular-source readback rows of chapter 301;*
4. *the* Real$^{\uparrow}$ *seal-boundary rows of chapter 38;*
5. *the displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows that name the packet.*

No other coordinate belongs to the packet.

Theorem 126.444 (StreamName Dyadic-RegSeq route factorization)**.** *Every accepted* StreamName$^{\uparrow}$ *finite-window consumer whose route reaches the* RegSeqRat$^{\uparrow}$ *readback surface or the* Real$^{\uparrow}$ *seal boundary factors through the finite packet of Definition 126.443. The factorization reads the StreamName window first, anchors it in the* DyadicRatCore$^{\uparrow}$ *observation and tolerance ledger, transports the anchored rows to the* RegSeqRat$^{\uparrow}$ *regular-source readback, and only then presents the* Real$^{\uparrow}$ *seal-boundary rows. The only structural data exposed along the route are the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *coordinates of the packet.*

Proof. Start with Definition 126.442. It places every StreamName finite-window read consumed by RegSeqRat$^{\uparrow}$ or by a Real$^{\uparrow}$ seal route on the DyadicRatCore$^{\uparrow}$ observation and tolerance ledger before any regular-rational handoff is read.

The three sibling chapters provide the packet faces. The dyadic ledger comes from chapter 291, the regular-source readback comes from chapter 301, and the terminal seal boundary comes from chapter 38.

Finally, repack the accepted consumer by the finite coordinates listed in Definition 126.443. The packet contains the StreamName rows, the dyadic ledger, the RegSeqRat readback, the Real seal boundary, and the displayed BHist, hsame, Cont, Pkg, and local NameCert data, so the consumer factors through exactly that route. □

Theorem 126.445 (StreamName Dyadic-RegSeq route no-completion leak)**.** *The packet of Definition 126.443 exports no quotient stream equality, host sequence equality, selected limit, countable-choice selector,* $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, or standard bridge row. Therefore an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window consumer that factors through the packet has no coordinate from which to read completion structure beyond the displayed* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Proof. Inspect the coordinates of Definition 126.443. They are exactly the StreamName finite-window rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation and tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-source readback, the $\mathsf{Real}^{\uparrow}$ seal boundary, and the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Apply the factorization theorem, Definition 126.444. Any accepted consumer reaching $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ is read through those coordinates and through no independent route outside the packet.

The coordinate list contains no quotient stream equality, host sequence equality, selected limit, countable-choice selector, $\mathsf{Real}^{\uparrow}$ completeness row, metric-completeness datum, or standard bridge row. Exhaustion of the defined packet therefore excludes each of those reads from the accepted consumer surface. □

126.33 StreamName Open-Phase Exit Boundary

Theorem 126.446 (StreamName open-phase exit boundary)**.** *Every current Phase Real and Completion consumer whose* $\mathsf{StreamName}^{\uparrow}$ *finite-window route reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly one finite BEDC exit boundary. The boundary is the StreamName window, the RegSeqRat readback, the DyadicRatCore ledger, the RealUp seal, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows on those finite surfaces.*

The boundary exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameOpenPhaseExitBoundary`

Proof. First apply Definition 126.415. It places every such Phase Real and Completion consumer on the terminal finite packet and identifies the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, displayed observations, transports, routes, provenance, and local naming rows.

Next use Definition 126.419 and Definition 126.421. Terminal seal exactness fixes the four-face route, and readback determinacy says that two consumers with the same four faces and displayed support rows determine the same terminal packet.

Now invoke Definition 126.442. It anchors every StreamName finite-window read consumed by RegSeqRat or by the Real seal through the DyadicRatCore observation and tolerance rows before the regular-rational handoff.

Finally apply Definition 126.437. The open-phase lattice exhausts the public consumer coordinates by the finite route and its displayed BHist, hsame, Cont, Pkg, and NameCert support. Therefore the exit boundary is exactly that finite BEDC packet, and no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completeness datum, or standard bridge row is exported. □

Theorem 126.447 (StreamName finite-window exit determinacy)**.** *Let two Phase Real and Completion consumers read accepted* $\mathsf{StreamName}^{\uparrow}$ *exit packets. If they expose the same StreamName*

finite window, RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* Real$^{\uparrow}$ *seal, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows, then they determine the same finite BEDC exit boundary. Conversely, every accepted exit boundary in this open phase is read through exactly those four faces and their displayed support rows.*

The determined boundary carries no quotient stream, host sequence equality, selected limit, ambient Real$^{\uparrow}$ *completeness row, metric-completeness datum, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameOpenPhaseFiniteWindowExitDeterminacy`

Proof. Start with Definition 126.446. It identifies the boundary of every Phase Real and Completion consumer as the finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

For determinacy, use Definition 126.419 and Definition 126.421. The seal exactness pins the Real-facing coordinate to the same four-face packet, and readback determinacy says that equality of those displayed rows forces equality of the terminal finite packet.

Now apply Definition 126.442. The StreamName window read is anchored through the DyadicRatCore observation and tolerance ledger before RegSeqRat or Real consumers can inspect it.

Finally use Definition 126.437. The lattice gives the converse direction: any accepted open-phase exit boundary is read through the StreamName, RegSeqRat, DyadicRatCore, and Real faces with the displayed support rows only. Since the cited boundaries expose no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completeness datum, or standard bridge row, the finite BEDC packet is the whole exit boundary. □

Theorem 126.448 (StreamName Real-completion exit certificate)**.** *Every Phase Real and Completion exit read from the current* StreamName$^{\uparrow}$ *open phase factors through the four finite sibling faces: the StreamName finite window, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* RegSeqRat$^{\uparrow}$ *readback, and the* Real$^{\uparrow}$ *seal. The certificate exposes only the displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

on those four faces, together with the finite window, tolerance, readback, and seal coordinates already named by the open-phase boundary.

The certificate exports no quotient stream, no selected limit, no host sequence equality, no Real$^{\uparrow}$ *completeness theorem, and no standard bridge row.*

Proof. First use Definition 126.446. It identifies the exit boundary for Phase Real and Completion consumers as the finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, and the displayed support rows.

Next apply Definition 126.419. The terminal seal makes the Real-facing coordinate exact inside the same finite packet rather than by a selected limit or ambient completeness assertion.

Then use Definition 126.442. The tolerance ledger is pinned to the StreamName finite window before the RegSeqRat and Real consumers can read it.

Finally apply Definition 126.437. The dependency lattice puts the RegSeqRat readback and RealUp seal on the same finite route, so the certificate is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert packet. Since the component theorems expose no quotient stream, host sequence equality, selected limit, RealUp completeness row, or standard bridge, the packaged exit certificate exports none. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionExitCertificate`

Theorem 126.449 (StreamName open-phase four-face terminality)**.** *Let two Phase Real and Completion exit reads expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the two reads determine the same terminal finite BEDC route through those four faces. Conversely, any terminal route available to the current open phase is read through exactly the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the displayed support rows named above.

The terminal route contains no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Proof. Begin with Definition 126.446. It identifies every Phase Real and Completion exit boundary with the finite StreamName, RegSeqRat, DyadicRatCore, and Real faces together with their displayed BHist, hsame, Cont, Pkg, and NameCert support.

Next use Definition 126.448. The certificate repacks the same four faces as the public exit read, so two reads with matching windows, ledgers, readbacks, seals, and displayed support rows have no remaining coordinate on which their terminal finite routes can differ.

Now apply Definition 126.442. The DyadicRatCore tolerance ledger is anchored to the StreamName finite window before either RegSeqRat readback or RealUp seal can be consumed, which prevents a second tolerance route from entering the terminal packet.

Finally invoke Definition 126.437. The lattice supplies the converse direction by exhausting the open-phase route through the four required faces. Since the cited routes export no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completeness datum, or standard bridge row, terminality belongs to that finite BEDC route alone. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier open phase four face terminality`

Theorem 126.450 (StreamName four-face exit readback)**.** *Every Phase Real and Completion exit read that exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

has exactly the finite readback surface named by the open-phase exit boundary. Conversely, every current Phase Real and Completion exit read factors through those four faces and those displayed support rows.

The readback exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier open phase four face exit readback`

Proof. Begin with Definition 126.446. It identifies the public exit surface as the StreamName finite window, the RegSeqRat readback, the DyadicRatCore ledger, the RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 126.448. The certificate places the four faces in one finite packet, so the tolerance ledger, regular-rational readback, and Real-facing seal are read together rather than through an ambient completion object.

Use Definition 126.449 for the reverse direction. Any terminal route available to the current open phase is already exhausted by the same four faces and displayed support rows.

Finally combine terminal seal exactness with terminal readback determinacy. These references identify matching four-face packets with the same finite readback surface and leave no coordinate for quotient streams, host sequence equality, selected limits, Real completeness, metric completeness, or bridge rows. □

Theorem 126.451 (StreamName open-phase four-object sufficiency)**.** *The four finite faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, are sufficient for the paper-side boundary required by Phase Real and Completion. Any remaining failure belongs to the per-object closurestatus entries or to Lean verification for those four faces, not to a hidden fifth mathematical object.*

The sufficient packet contains no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameOpenPhaseFourObjectSufficiency`

Proof. First apply Definition 126.446 and Definition 126.448. The boundary theorem names the current Phase Real and Completion consumer surface, and the certificate repacks that surface as the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and displayed support rows.

Next use Definition 126.449 and Definition 126.450. Terminality says that matching four faces and support rows determine the terminal finite route, while readback says that every current exit read is read through exactly those rows.

Finally invoke Definition 126.500. It aligns the same four faces with the roadmap objects DyadicRatCore, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Therefore the paper-side exit package is already sufficient for Phase Real and Completion, and any later failure must be read from the individual closurestatus or verification axes rather than from an undisplayed fifth object. □

Theorem 126.452 (StreamName open-phase Lean touchpoint triad)**.** *Every current Phase Real and Completion exit read through the four faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is paper-audit-ready exactly along the three existing Lean touchpoints named by the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. This triad exports only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already present on those four faces.*

The triad adds no Lean marker at this paper site, no quotient stream equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness row, and no standard bridge row.*

Proof. First apply Definition 126.451. It says that the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows, already suffice for the current Phase Real and Completion boundary.

Next use Definition 126.450. The readback theorem fixes every current exit read on exactly those four faces and displayed support rows, so the paper audit has no hidden fifth object to inspect.

Finally invoke Definition 126.500. It aligns the same boundary with the roadmap route. The existing machine-side route is then the three-touchpoint path named in Lean as `BEDC.Derived.StreamNameUp.RatStr`

`BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness`, and `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_real_seal_handoff`. These names identify the future formalization route only; they do not create a current paper marker or export quotient equality, selected limits, ambient completeness, metric completeness, or a standard bridge row. □

Theorem 126.453 (StreamName Real-completion exit Lean route boundary)**.** *Every current Phase Real and Completion four-face exit read consumes machine support only through the three primary paper sites already carrying the route markers: the StreamName singleton finite-window handoff of Definition 126.173, the DyadicRatCore RegSeqRat bridge handoff exactness of Definition 291.28, and the RegSeqRat Real seal handoff of Definition 301.22. Equivalently, the route uses*

`BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff`,
`BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness`,
`BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_real_seal_handoff`

only at those primary sites.

This paper site records the route boundary but adds no additional Lean marker. It exports no quotient stream equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, countable choice, or bridge row.*

Proof. First apply Definition 126.451. It fixes the current exit packet as the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the visible BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 126.452. The triad says that the machine-facing support of that packet is exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

Finally invoke Definition 126.500. It identifies the same four-face packet as the Phase Real and Completion route, while the primary markers are already attached at Definition 126.173, Definition 291.28, and Definition 301.22. Therefore this boundary site can only point to those primary sites and cannot create a second marker or an extra coordinate for quotient streams, selected limits, ambient completeness, metric completeness, countable choice, or bridge data. □

Theorem 126.454 (StreamName Real-completion exit obligation ledger)**.** *The current Phase Real and Completion exit obligation ledger is exactly the four faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The ledger has no separate aggregate object beyond those four faces and their support rows.*

Proof. First apply Definition 126.446 and Definition 126.448. These theorems name the exit packet as the finite StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and the displayed support rows.

Next use Definition 126.449 and Definition 126.450. Terminality fixes the route once the four faces and support rows agree, while readback says that every current exit read is consumed through those rows.

Finally apply Definition 126.453. The Lean-facing route is confined to the existing primary touchpoints for the StreamName finite window, DyadicRatCore RegSeqRat handoff, and RegSeqRat Real seal. Hence the paper obligation ledger is exactly the four displayed faces with BHist, hsame, Cont, Pkg, and NameCert support. □

Theorem 126.455 (StreamName open-phase Lean route totality)**.** *Every current Phase Real and Completion exit read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is machine-facing exactly through the three primary Lean touchpoints named in Definition 126.453. Thus the current route is exhausted by the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Any consumer that needs further machine support has left the current open-phase exit ledger.*

Proof. First apply Definition 126.454. It fixes the current ledger as the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Next use Definition 126.453 and Definition 126.452. These identify the machine-facing route of that ledger with exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

Finally invoke Definition 126.451. Since the four listed faces already suffice for the current Phase Real and Completion boundary, a consumer asking for another machine support row is no longer reading the displayed open-phase exit ledger. The theorem adds no marker and asserts no quotient stream, selected limit, completeness row, countable-choice row, or bridge row. □

Theorem 126.456 (StreamName L10 gateway scope)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *gateway read for the current L10 phase is scoped by the finite window row together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Conversely, those four faces and support rows are sufficient for the public gateway packet. The gateway exports no quotient stream equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, countable-choice row, aggregate formalstatus row, aggregate bridge row, or standard bridge row.*

Proof. For necessity, start with Definition 126.446. It identifies the accepted gateway read with the finite StreamName window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next apply Definition 126.447. The finite window row determines the same exit packet whenever the dyadic, regular-sequence, Real seal, and support rows agree, so a gateway read cannot be scoped by a separate quotient or limiting coordinate.

For sufficiency, use Definition 126.448 and Definition 126.437. The certificate packages the four faces as the L10 exit surface, and the dependency lattice supplies the converse route through exactly those rows.

Finally apply Definition 126.455. The Lean-facing side of the current gateway is exhausted by the displayed touchpoints of the same finite route. Thus the public gateway packet is exactly the scoped four-face packet, with no quotient stream equality, selected limit, ambient completeness, metric-completeness datum, countable-choice row, aggregate status, aggregate bridge, or standard bridge row. □

Theorem 126.457 (StreamName Real-completion touchpoint minimality)**.** *A current Phase Real and Completion exit read is paper-audit-complete if and only if it exposes exactly the four paper faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and its machine-facing support is confined to the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.*

Equivalently, the paper audit surface is precisely the four-face ledger named by Definition 126.451, while the machine route is precisely the three-touchpoint route named by Definition 126.453 and exhausted by Definition 126.455. The complete read contains no quotient stream equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, countable-choice row, or standard bridge row.*

Proof. For the forward direction, start from a paper-audit-complete current exit read. Apply Definition 126.451 to identify the paper surface with exactly the four displayed faces and their visible BHist, hsame, Cont, Pkg, and NameCert rows.

Then use Definition 126.453. It confines the machine-facing support to the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Finally apply Definition 126.455; any requested machine coordinate outside those three sites leaves the current open-phase exit ledger.

For the reverse direction, assume the read exposes the four faces and only the three named machine touchpoints. The four-face sufficiency theorem supplies the paper audit packet, while the Lean-route boundary supplies the complete machine-facing route for that packet.

The route-totality theorem rules out missing support inside the current exit read. Since neither the four-face packet nor the three-touchpoint route has a coordinate for quotient stream equality, selected limits, ambient completeness, metric-completeness data, countable choice, or standard bridge rows, the read is paper-audit-complete exactly at the displayed boundary. □

Theorem 126.458 (StreamName Real-completion objectwise status completeness)**.** *A current Phase Real and Completion exit read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is status-complete exactly when it exposes those four faces, confines machine support to the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff, and also reads the four local face closurestatus, formalstatus, leantarget, and bridgestatus rows objectwise.*

Equivalently, the three-touchpoint route is necessary for the machine-facing side of the current roadmap exit certificate, but it is not by itself sufficient for the roadmap exit certificate. The certificate is complete only when the four local face-status records are present alongside the displayed finite support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The status-complete read contains no quotient stream equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, countable-choice row, aggregate formalstatus row, aggregate bridge row, or standard bridge row.*

Proof. First identify the underlying exit surface. By Definition 126.457, the current paper audit surface exposes exactly the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while the machine-facing support is confined to the three named touchpoints.

Next exhaust the route. Apply Definition 126.454 to keep the obligation ledger on the four faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Apply Definition 126.455 to rule out any further machine coordinate inside the current open-phase ledger.

It remains to read the roadmap exit certificate on its own axes. The roadmap discipline of section 9467.2 and the Phase Real and Completion exit criterion of section 9467.19 require the four objects to be read locally on independent closure and verification axes. By Definition 38.792, that read is the objectwise product of the four local closurestatus, formalstatus, leantarget, and bridgestatus records, together with the displayed finite support rows.

Therefore the three-touchpoint route is necessary for the machine-facing support, but it cannot replace the four local status rows. Conversely, when the four faces, the three touchpoints, and the

four objectwise status records are all present, the same cited results supply exactly the current Phase Real and Completion exit certificate.

Finally exclude substitute coordinates. The touchpoint-minimality theorem and the Real-side objectwise face-status theorem jointly leave no coordinate for quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, countable choice, aggregate formal-status rows, aggregate bridge rows, or standard bridge rows. □

Theorem 126.459 (StreamName open-phase four-face scope exactness)**.** *Any current Phase Real and Completion route that exposes the four faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *has no fifth object. In particular it does not contain a quotient stream, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness object, metric-completeness datum, countable-choice row, or standard bridge row.*

Proof. Begin with Definition 126.449. Once the four faces and their displayed support rows are fixed, the terminal finite route is determined.

Use Definition 126.450. It reads every current exit through the same four local faces, so any proposed fifth object would have to appear in that readback surface.

Now apply Definition 126.448 and Definition 126.453. The certificate and machine-route boundary list only the finite window, dyadic ledger, RegSeqRat readback, Real seal, and the three existing primary machine touchpoints. None of those rows is a quotient stream, selected limit, ambient completeness object, metric-completeness datum, countable-choice row, or standard bridge. □

Theorem 126.460 (StreamName Real-RegSeqRat dyadic window pullback)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in the current Phase Real and Completion exit share the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through the finite* $\mathsf{StreamName}^{\uparrow}$ *window. Thus any exit read that preserves the Real seal and the RegSeqRat readback pulls back to the same dyadic tolerance window, rather than to a selected limit, quotient stream, host equality, ambient completeness theorem, or bridge coordinate.*

Proof. First use Definition 126.448. It displays the Real seal and RegSeqRat readback in the same certificate as the finite StreamName window and DyadicRatCore tolerance ledger.

Next apply Definition 126.449 and Definition 126.450. These results force both reads through the same four-face terminal packet, so the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ faces cannot select different dyadic ledgers once the StreamName finite window is fixed.

Finally invoke Definition 126.453. The machine-facing support is exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Those touchpoints align the dyadic tolerance ledger with the finite window and provide no coordinate for selected limits, quotient streams, host equality, ambient completeness, or bridge data. □

Theorem 126.461 (StreamName Real-completion four-face terminal pullback)**.** *Let X and Y be current Phase Real and Completion exit reads. Suppose both reads expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then X and Y pull back to the same terminal finite BEDC route.*

No quotient stream, selected limit, host equality, ambient completeness theorem, metric-completeness datum, choice row, or bridge row can distinguish the two reads once those four faces and displayed support rows agree.

Proof. First apply Definition 126.451. It identifies the current Phase Real and Completion boundary with exactly the four objects $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the visible BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 126.450. Every current exit read is read through precisely those four faces and support rows, so the hypotheses on X and Y place them on the same finite readback surface.

Finally apply Definition 126.453. The machine-facing route is confined to the three primary touchpoints already attached to the StreamName finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Since those touchpoints carry no quotient, limit-selection, host-equality, completeness, choice, or bridge coordinate, the two reads determine the same terminal finite BEDC route. □

Theorem 126.462 (StreamName four-face Lean route consistency)**.** *Every current four-face exit read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumes machine support only through the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Conversely, those three touchpoints cover exactly the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the paper route through the four faces.*

This consistency statement adds no Lean marker at this paper site. It does not claim quotient streams, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, or standard bridge status.*

Proof. First apply Definition 126.446. It identifies the current open-phase exit boundary with the finite StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal, and displayed support rows.

Next use Definition 126.451 and Definition 126.449. These theorems say that the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ suffice for the current exit packet and that matching those faces with their support rows determines the terminal route.

Now apply Definition 126.453. The machine-facing support route is confined to the three named touchpoints: the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

For the converse direction, read those same touchpoints back along the four-face terminal packet. The StreamName handoff covers the finite-window face, the DyadicRatCore handoff covers the dyadic ledger and bridge face, and the RegSeqRat handoff covers the Real seal face; together with the terminality and sufficiency results, these are exactly the displayed BHist, hsame, Cont, Pkg, and NameCert rows of the paper route. No additional machine support coordinate remains in the open-phase boundary. □

Theorem 126.463 (StreamName open-phase facewise record completeness)**.** *A Phase Real and Completion exit read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is complete as a* $\mathsf{StreamName}^{\uparrow}$*-side boundary exactly when the four canonical local face records are displayed separately: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger status, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window status, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and seal status, and the* $\mathsf{Real}^{\uparrow}$ *seal status. These are the local records attached to the four displayed faces, not a composite route token.*

For the reverse direction, the only machine-route sites used are the three primary sites already named by Definition 126.453. No aggregate formalstatus token, aggregate bridge, selected limit, quotient stream, ambient completeness theorem, countable-choice row, or metric-completeness datum can replace a missing face record.

Proof. For the forward direction, apply Definition 126.451. It says that the current Phase Real and Completion boundary is exhausted by the four displayed faces together with the visible BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 126.450. The readback theorem identifies every current exit read with exactly those four faces and their displayed support rows. Thus audit completeness can only be checked componentwise, by reading the StreamName, DyadicRatCore, RegSeqRat, and Real records that already belong to the packet.

For the reverse direction, assume that the four canonical local face records are displayed separately. Apply Definition 126.451 to identify the packet with exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and their visible support rows.

Next apply Definition 126.450. It reads every current exit through those same four face records, so the displayed local rows cover the StreamName-side boundary rather than merely a downstream summary of it.

Finally apply Definition 126.453. That theorem restricts the reverse route to the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff; those are precisely the primary machine-route sites already attached to the four local records. The roadmap two-axis discipline is used only to keep the four local theory and verification reports from collapsing into one token. Hence no aggregate formalstatus token, aggregate bridge, selected limit, quotient stream, ambient completeness theorem, countable-choice row, or metric-completeness datum can repair an absent face record. □

Theorem 126.464 (StreamName–Real open-phase facewise audit correspondence)**.** *A current Phase Real and Completion exit read is complete as a* $\mathsf{StreamName}^{\uparrow}$*-side facewise record exactly when it satisfies the* $\mathsf{Real}^{\uparrow}$*-side facewise audit obstruction. Both conditions require the same four local records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Thus the StreamName-side open-phase completeness criterion and the Real-side L10 audit obstruction name the same componentwise four-face boundary. No aggregate formalstatus token, bridge row, completeness row, quotient stream, selected limit, or host equality row is part of this correspondence.

Proof. For the StreamName-to-Real direction, apply Definition 126.463. A complete StreamName-side read already displays the DyadicRatCore tolerance-ledger record, the StreamName finite-window record, the RegSeqRat readback and seal record, and the Real seal record as separate local rows.

Next use Definition 38.729 and Definition 38.728. The Real-side obstruction asks for exactly those four local records and refuses any aggregate audit substitute. Since the visible support rows are the same BHist, hsame, Cont, Pkg, and NameCert rows, the StreamName-side complete read satisfies the Real-side audit condition.

For the Real-to-StreamName direction, start from the Real-side facewise audit condition. It supplies the four local audit records separately, so Definition 126.451 identifies the current open-phase packet with the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and their displayed support rows.

Finally apply Definition 126.453. The only machine-facing route available to this packet is the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. These are the route sites already attached to the same four local records, so the Real-side audit condition gives the StreamName-side facewise record completeness.

The last sentence of the theorem is the common boundary condition, not an extra certificate. The cited results leave no coordinate for an aggregate formalstatus token, bridge row, completeness row, quotient stream, selected limit, or host equality row. □

Theorem 126.465 (StreamName budgeted real window handoff totality)**.** *Every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *consumer of a* $\mathsf{StreamName}^{\uparrow}$ *finite-window exit read factors through the displayed handoff rows*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{RealWindowBudget}^{\uparrow},\quad \mathsf{BudgetedRealSealRoute}^{\uparrow}$$

together with only the support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The factorization is total for such consumers: the $\mathsf{StreamName}^{\uparrow}$ *finite window is first read through the four local face records, then through the* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, and finally through the* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *carrier and local* $\mathsf{NameCert}$ *surface.*

No selected limit, quotient stream, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, countable-choice row, metric-completeness datum, host equality row, or standard bridge row is part of this consumer handoff.*

Proof. Start with Definition 126.463. It gives the $\mathsf{StreamName}^{\uparrow}$-side exit read as four separately displayed local records: the StreamName finite window, the DyadicRatCore tolerance ledger, the RegSeqRat readback and seal status, and the RealUp seal status.

Next apply Definition 126.473 and Definition 126.474. The pullback places every accepted $\mathsf{RealWindowBudget}^{\uparrow}$ consumer over the same finite request, window, dyadic, regularity, and Real seal rows, while terminality prevents any further discriminator once those rows and the displayed support rows agree.

Now use Definition 1164.4. The $\mathsf{RealWindowBudget}^{\uparrow}$ packet therefore hands the finite StreamName window, dyadic readbacks, RegSeqRat handoff, and Real seal to the Real-facing completion route without adding a selected limit, quotient stream, or ambient completeness coordinate.

Finally read the $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ carrier and NameCert obligations of Definition 913.1 and Definition 913.2. Its accepted packet consumes the displayed selector, threshold, observation-budget, limit-seal, Real readback, transport, continuation, provenance, and local naming rows; along the StreamName exit boundary these are reached only through the already displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealWindowBudget}^{\uparrow}$ rows. The non-escape theorem Definition 913.6 then excludes host equality, selected limits, quotient streams, ambient completeness, countable choice, metric-completeness data, and standard bridge rows from the handoff. □

Theorem 126.466 (StreamName open-phase budget route lock)**.** *Any accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *consumer whose finite window is read through* $\mathsf{StreamName}^{\uparrow}$ *factors through the same four-face open-phase exit boundary*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

with only BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. The budgeted route is therefore a consumer of the boundary, not a fifth Real-completion object, and it cannot smuggle quotient streams, selected limits, host real equality, ambient completeness, countable choice, or metric-completeness data into the exit packet.*

Proof. Start with the open-phase exit boundary, Definition 126.446. It identifies the current Real-completion exit with the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat regularity readback, RealUp seal, and the visible support rows.

Next use budgeted real window handoff totality, Definition 126.465. It already routes an accepted $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ consumer through the four local faces, the $\mathsf{RealWindowBudget}^{\uparrow}$ packet, and only BHist, hsame, Cont, Pkg, and NameCert support.

Finally read the budgeted route selector, TailMeet threshold, Real-seal readback, and non-escape theorems from chapter 913. The selector and threshold rows feed the route into the RealObservationBudget and CauchyLimitSeal handoff, while the Real readback and non-escape rows prevent a separate completion object or hidden stream source from appearing. □

Theorem 126.467 (StreamName open-phase local status nonescape)**.** *For any current Phase Real and Completion exit read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, suppose one of the four local face status records is absent: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger status, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window status, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and seal status, or the* $\mathsf{Real}^{\uparrow}$ *seal status. Then the absent local record is not supplied by any aggregate formalstatus token, aggregate bridge row, selected limit, quotient stream, ambient completeness theorem, countable-choice row, metric-completeness datum, or standard bridge row.*

Equivalently, the current open-phase exit boundary has no escape route from a missing local face record to a route-level verification or bridge coordinate. Each of the four objects must display its own local status row before the $\mathsf{StreamName}^{\uparrow}$*-side boundary is complete.*

Proof. Start with Definition 126.463. It identifies boundary completeness with the four canonical local records displayed separately, so an absent DyadicRatCore, StreamName, RegSeqRat, or Real record is already a failure at the objectwise status surface.

Next apply Definition 126.451 and Definition 126.450. Sufficiency says that the current exit packet has only the four displayed objects and their visible BHist, hsame, Cont, Pkg, and NameCert rows. Readback says that every current exit read is read through exactly those four local faces.

Now use Definition 126.453. The only machine-facing route available to the current read is the three primary touchpoints already attached to the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Those touchpoints can report the route they occupy, but they do not manufacture a missing local status row for any face.

Finally exhaust the possible substitutes. The four-face route has no quotient stream, selected limit, ambient completeness theorem, countable-choice row, metric-completeness datum, or standard bridge row by the cited boundary results. Aggregate formalstatus and bridge prose therefore remain summaries of displayed local records, not replacements for one of them. □

Theorem 126.468 (StreamName open-phase local-status exit lock)**.** *A current Phase Real and Completion exit read is accepted at the* $\mathsf{StreamName}^{\uparrow}$ *side exactly when the four local status records are present together: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window status, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger status, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and seal status, and the* $\mathsf{Real}^{\uparrow}$ *seal status. The four records must be matched to the same displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

When those four records and those support rows agree, they jointly determine the current exit route through the terminal four-face packet. When one record is absent or mismatched, the route is not accepted as the current $\mathsf{StreamName}^{\uparrow}$*-side exit read.*

This lock is local to the open-phase boundary. It does not introduce quotient streams, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, aggregate formal-status rows, or aggregate bridge rows.*

Proof. For necessity, apply Definition 126.463. It says that boundary completeness is exactly the separate display of the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ local status records over the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use Definition 126.467. If any one of the four local records is absent, no route-level status token, bridge row, selected limit, quotient stream, completeness datum, or countable-choice row can replace it. Thus every accepted current exit read must carry all four local records.

For sufficiency, apply Definition 126.470. Once the four face records and the displayed support rows match, public acceptance factors through the terminal four-face classifier surface and has no further coordinate.

Finally apply Definition 126.472. It identifies the accepted route with the unique normal-form packet consisting of the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and the same support rows. Hence the four local records jointly determine the current exit route, while the excluded quotient, limit, completeness, choice, aggregate status, and bridge rows remain outside the packet. □

Theorem 126.469 (StreamName open-phase handoff terminal cover)**.** *Every current Phase Real and Completion handoff that reaches the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal is covered by the terminal finite BEDC packet displayed in this section. The cover is exactly the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

The cover exports no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. Begin with Definition 126.446. It identifies the public handoff boundary with the finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, and the displayed support rows.

Next apply Definition 126.448. The certificate repacks the same boundary as the four-face packet, so the StreamName window, tolerance ledger, readback, and seal are read as one finite BEDC surface.

Use Definition 126.449 and Definition 126.450. Terminality says that matching four faces and support rows determine the terminal finite route, while readback says that every current exit read is already read through exactly those rows.

Finally combine Definition 126.451 with Definition 126.453. The sufficiency theorem aligns the packet with the open construction order, and the machine-touchpoint boundary restricts the route to the named primary sites. Therefore every current handoff reaching those four faces is covered by the displayed terminal packet, with no additional quotient, host equality, selected limit, completeness, choice, or bridge coordinate. □

Theorem 126.470 (StreamName open-phase terminal classifier exactness)**.** *A current Phase Real and Completion read is publicly accepted exactly when its* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

factor through the terminal four-face classifier surface for the open phase.

Equivalently, public acceptance has no further coordinate beyond that terminal classifier packet: no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. For necessity, start with Definition 126.449 and Definition 126.450. Terminality identifies the finite route determined by the four faces and displayed support rows, while readback says every current Phase Real and Completion exit read is read through exactly those faces and rows.

Next apply Definition 126.500. It aligns the same StreamName window, DyadicRatCore ledger, RegSeqRat readback, and RealUp seal with the current Real-completion exit route, so the open-phase terminal route is the four-object route already used by Phase Real and Completion.

For sufficiency, use Definition 126.514. Once the four faces and displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows factor through that classifier surface, the public read is determined componentwise by the same finite packet.

Finally invoke Definition 126.518. Dependency exactness says that the four dependency faces with the displayed support rows are precisely the public acceptance boundary, and it excludes quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, countable choice, and standard bridge rows. Hence public acceptance is exact for the terminal four-face classifier surface and for no larger object. □

Theorem 126.471 (StreamName open-phase four-face ledger exactness)**.** *For every current Phase Real and Completion read accepted through the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

the ledger coordinate is exactly the finite four-face ledger already exposed by those rows. Equivalently, any $\mathsf{RealWindowBudget}^{\uparrow}$ *consumer can project the ledger from the four faces themselves, and any hidden compression, selected limit, quotient stream, host sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness row, countable-choice row, or standard bridge row lies outside the accepted packet.*

Proof. First apply Definition 126.470. It identifies public acceptance with the terminal four-face classifier surface over the StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal, and displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next apply Definition 126.451. The current boundary has only the four objects $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as its exposed mathematical faces, so the ledger coordinate cannot be supplied by a fifth object or an undisplayed completion row.

Now use Definition 126.453. The machine-facing support route is confined to the three primary sites already attached to the StreamName finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff, and the RegSeqRat Real seal handoff; this theorem adds no marker here and contributes no extra ledger coordinate.

Finally read the downstream budget surface. By Definition 1164.1 and Definition 1164.4, a $\mathsf{RealWindowBudget}^{\uparrow}$ packet projects its finite StreamName window, dyadic readbacks, RegSeqRat handoff, and Real seal from the same displayed route. Thus the accepted packet exposes exactly the four-face ledger consumed downstream, and hidden compression, selected limits, quotient streams, host equality, ambient completeness, metric completeness, countable choice, and standard bridge data remain outside the packet. □

Theorem 126.472 (StreamName open-phase four-face normal form)**.** *Every accepted Phase Real and Completion exit read has a unique normal form at the terminal finite BEDC route. The normal form consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Conversely, any packet whose exposed coordinates are exactly those four faces and those support rows is accepted by the terminal classifier surface for the open phase.

The normal form has no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. For existence, apply Definition 126.451. It identifies the current Phase Real and Completion boundary with the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the visible BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 126.450. Every accepted exit read is read through those same faces and rows, so the displayed packet is a normal-form representative of the current open-phase exit read rather than a downstream summary.

For uniqueness, use Definition 126.470. Public acceptance is exact for the terminal four-face classifier surface, so two accepted packets with the same four faces and support rows determine the same terminal finite BEDC route.

For the converse direction, apply Definition 126.471. A packet exposing exactly the StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal, and displayed support rows already supplies the ledger coordinate consumed by the terminal classifier surface.

Finally exhaust the excluded coordinates. The same cited theorems leave no place for quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, countable choice, or standard bridge data. Therefore the four-face packet is the canonical Phase Real and Completion exit object, and any packet with precisely those coordinates is accepted by the terminal classifier surface. □

Theorem 126.473 (StreamName Real-window budget pullback)**.** *Any accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window exit read consumed by a* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet factors through the same finite request,* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row. Conversely, every accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet exposes a* $\mathsf{StreamName}^{\uparrow}$ *finite-window face through its displayed window rows.*

The pullback adds no quotient stream, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, metric-completeness datum, countable-choice row, or standard bridge row.*

Proof. For the forward direction, apply Definition 126.470. It identifies accepted open-phase reads with the terminal four-face classifier surface over the StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal, and displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next read the $\mathsf{RealWindowBudget}^{\uparrow}$ carrier of Definition 1164.1. Its request, window, dyadic, handoff, seal, selector, disclosure, transport, continuation, provenance, and local naming rows are precisely the finite route available to a Real-window budget consumer.

For the reverse direction, project an accepted $\mathsf{RealWindowBudget}^{\uparrow}$ packet onto its V coordinate. By the carrier definition and the Real-completion handoff of Definition 1164.4, V is the StreamName finite-window face of the same packet.

Finally compose with the sibling budget surfaces. The selector stability of Definition 1164.3, the CauchyDiagonalBudget surface of chapter 853, and the RealObservationBudget disclosure route of chapter 611 keep the pullback on the same finite request and support rows. Since none of these surfaces carries quotient streams, host equality, selected limits, ambient completeness, metric completeness, countable choice, or a standard bridge, the pullback exports none. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier streamname pullback route`

Theorem 126.474 (StreamName open-phase budget-consumer terminality)**.** *Every accepted* RealWindowBudget$^{\uparrow}$ *consumer of a* StreamName$^{\uparrow}$ *finite-window exit read determines the same terminal four-face route through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$. *The route is read only through the displayed rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Once two such consumers agree on their budget rows, no quotient stream, selected limit, host equality, ambient completeness theorem, countable-choice row, metric-completeness datum, or standard bridge row can distinguish them.

Proof. Begin with the pullback of Definition 126.473. It says that every accepted budget consumer of the StreamName finite-window exit read factors through the finite request, StreamName window rows, DyadicRatCore readbacks, RegSeqRat handoff, Real seal, componentwise hsame transports, displayed Cont route, Pkg provenance, and local NameCert row.

Next apply Definition 126.470 and Definition 126.500. The classifier exactness theorem makes the accepted open-phase read terminal over the displayed four faces, and the four-object exit route identifies those faces as the StreamName, DyadicRatCore, RegSeqRat, and RealUp route already used by Phase Real and Completion.

Finally read the budget carrier. By Definition 1164.1 and Definition 1164.4, a RealWindowBudget$^{\uparrow}$ consumer exposes its Real-facing route as the same finite chain from the StreamName window through dyadic readback and regular handoff to the Real seal. Thus agreement on the displayed budget rows leaves only that terminal four-face route, while quotient streams, selected limits, host equality, ambient completeness, choice, metric completeness, and standard bridge data remain outside the accepted packet. □

Theorem 126.475 (StreamName open-phase exit route completeness)**.** *A current Phase Real and Completion exit packet is accepted exactly when it is a terminal finite route through the* StreamName$^{\uparrow}$ *finite window, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* RegSeqRat$^{\uparrow}$ *readback, the* Real$^{\uparrow}$ *seal, and the displayed support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, every accepted packet exposes precisely those four L10 faces and those rows, and conversely any packet exposing exactly those coordinates is accepted by the terminal open-phase classifier surface.

The route has no quotient stream equality, no host sequence equality, no selected limit, no ambient Real$^{\uparrow}$ *completeness row, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. For necessity, apply Definition 126.472. It identifies every accepted Phase Real and Completion exit read with the normal-form packet consisting of the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 126.471. The tolerance-ledger coordinate is not an external budget summary; it is the finite ledger already exposed by the four faces and their support rows.

Then pull the downstream consumer back by Definition 126.473. Any $\mathsf{RealWindowBudget}^{\uparrow}$ consumer of the StreamName finite-window exit read factors through the same finite request, window rows, dyadic readbacks, RegSeqRat handoff, Real seal, componentwise transports, continuation route, provenance, and local naming row.

Finally apply Definition 126.474. Once the four faces and displayed budget rows agree, every accepted budget consumer determines the same terminal four-face route. The converse clause of the normal form and ledger exactness therefore accepts any packet exposing exactly those rows at the terminal classifier surface, and the cited exclusions leave no coordinate for quotient stream equality, host sequence equality, selected limits, ambient RealUp completeness, metric-completeness data, countable choice, or standard bridge rows. □

Theorem 126.476 (StreamName open-phase no-fifth-object exhaustion)**.** *Every accepted Phase Real and Completion exit route in the current open phase is exhausted by the four L10 faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. There is no fifth object, aggregate verification token, quotient stream, selected limit, ambient completeness row, metric-completeness datum, countable-choice row, or standard bridge row hidden behind the terminal exit route.*

Proof. First apply the four-face normal form of Definition 126.472. It identifies accepted exit reads with the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and the displayed support rows.

Next use budget-consumer terminality from Definition 126.474. Any accepted budget consumer determines the same terminal route once the four faces and budget rows agree, so a separate consumer token cannot add an object beyond the listed faces.

Apply Definition 126.475. It supplies the converse direction: a packet exposing exactly the four faces and support rows is accepted by the terminal open-phase classifier surface.

Finally use the four-face ledger exactness of Definition 126.471 and the four-face exit readback of Definition 126.450. They keep the tolerance ledger and readback rows inside the same displayed packet, leaving no route for a quotient stream, selected limit, ambient completeness theorem, metric-completeness datum, countable-choice row, standard bridge row, or hidden fifth object. □

Theorem 126.477 (StreamName open-phase roadmap exit conjunction)**.** *The current Phase Real and Completion paper-side boundary condition is the conjunction of the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

read componentwise through their own local closurestatus and formalstatus records and through the shared support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

This conjunction is sufficient only for the current paper-side boundary and audit-readiness route. Full phase exit is read objectwise from each face's own closurestatus and formalstatus records; no aggregate bridge or formalstatus token is created. It does not assert selected limits, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, or standard bridge completion.*

Proof. Start with Definition 126.475. It states that an accepted Phase Real and Completion exit packet is exactly the terminal finite route through the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and the displayed support rows.

Next apply Definition 126.476. This removes any hidden fifth object or aggregate token from the boundary, so the route can only be read as the four displayed faces with their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Align this route with the roadmap by Definition 126.500. The four objects named there are the same DyadicRatCore, StreamName, RegSeqRat, and Real faces, so the open-phase boundary and the Real-completion roadmap boundary have the same visible coordinates.

Finally use Definition 38.728 and Definition 38.729. They require each face to contribute its own closurestatus and formalstatus record and prevent any support row or aggregate audit token from replacing a missing local face record. Therefore the componentwise four-face conjunction is sufficient only for the paper-side boundary and audit-readiness route. Full phase exit remains an objectwise read from the four local records, and the excluded selected-limit, quotient-equality, completeness, choice, metric, aggregate bridge, and aggregate formalstatus claims remain outside the theorem. □

Theorem 126.478 (StreamName open-phase exit audit consolidation)**.** *An accepted Phase Real and Completion audit packet for the current* $\mathsf{StreamName}^{\uparrow}$ *open-phase exit is valid exactly when it exposes exactly the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with each of $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *exposing its own current-roadmap local theory-closure record and a verification record at least as strong as the audit-clean threshold, read from that face's own closurestatus block, with* $\mathsf{bridgeChecked}$ *read only as the local face bridge token where that face records it, together with the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, validity is the conjunction of those four objectwise threshold records and those finite support rows, and nothing else.

The audit packet excludes quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, standard bridge completion, and any aggregate formalstatus or bridge token.*

Proof. For sufficiency, apply Definition 126.477. It identifies the paper-side boundary condition as the four local Phase Real and Completion faces read through their own local records, plus the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Then apply Definition 38.494 as a threshold read. The Real-side criterion says those local records must meet the current objectwise threshold: each face supplies the required local theory-closure row and a local formalstatus row meeting or exceeding $\mathsf{auditClean}$, read from its own closurestatus block. If a face records $\mathsf{bridgeChecked}$, that token is read only as that face's local bridge record; it is not converted into an aggregate formalstatus or bridge token.

Use Definition 126.475 to read that thresholded visible packet as an accepted terminal finite route.

For necessity, apply Definition 126.476. It says every accepted open-phase exit route is exhausted by the same four faces and displayed rows. The ledger exactness of Definition 126.471 fixes the tolerance-ledger coordinate inside the finite packet, while the readback theorem Definition 126.450 fixes every current exit read on exactly those rows.

Finally use Definition 38.494 and Definition 38.729. Mere record presence is not enough: the local rows must satisfy the threshold, and no aggregate support row can compensate for a missing or sub-threshold face. Thus the necessary and sufficient audit criterion is exactly the four facewise threshold records with the displayed support rows, with each face keeping its own local formalstatus token and with no aggregate formalstatus, bridge token, quotient equality, selected limits, ambient completeness, metric completeness, countable choice, or standard bridge completion. □

Theorem 126.479 (StreamName RegSeqRat–Real selector-budget normal form)**.** *Every accepted Phase Real and Completion exit read whose visible faces are*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

factors through one selector-budget packet. The packet consists exactly of the finite request row, selected-window row, dyadic tolerance row, regularity handoff row, $\mathsf{Real}^{\uparrow}$ *seal row, support and provenance rows, and local* NameCert *row.*

The factorization is read through the same surfaces used by the sibling $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *chapters, namely chapter 38, chapter 301, chapter 291, and chapter 885. It excludes quotient streams, host equality, selected limits, ambient completeness, and any fifth object.*

Proof. First pull the Real and RegSeqRat faces back to the same dyadic window. By Definition 126.460, the $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{RegSeqRat}^{\uparrow}$ readback share the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger through the finite $\mathsf{StreamName}^{\uparrow}$ window.

Next expose the selector-budget rows. The Real-window budget pullback of Definition 126.473 identifies the finite request, selected window, dyadic readbacks, regular handoff, Real seal, componentwise transport, continuation route, provenance, and local naming rows as one packet rather than as separate completion data.

Now use Definition 126.474. Agreement on the displayed budget rows determines the terminal four-face consumer route, so any Phase Real or Completion exit read with those visible faces factors through the same selector-budget surface.

Finally apply Definition 126.476. Since the terminal exit route is exhausted by the four displayed faces and support rows, no quotient stream, host equality, selected limit, ambient completeness row, or fifth object remains as an additional factor. □

Theorem 126.480 (StreamName selector-budget defect localization)**.** *Let an accepted Phase Real and Completion exit read expose the visible faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

If this read fails to factor through the selector-budget packet of Definition 126.479, then the defect is witnessed by at least one displayed finite request row, selected-window row, dyadic tolerance row, regularity handoff row, $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *row,* hsame *row,* Cont *row,* Pkg *row, or local* NameCert *support row.*

The defect cannot be witnessed by quotient stream equality, host equality, selected limits, ambient completeness, aggregate formalstatus, aggregate bridge rows, countable choice, or any fifth object.

Proof. Apply Definition 126.479. It identifies the only selector-budget packet available to an accepted exit read with the four displayed faces, so a failure of factorization must be a failure of one coordinate in that packet.

The packet coordinates are local. The dyadic-window pullback of Definition 126.460 fixes the shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger for the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal. The Real-window budget pullback of Definition 126.473 identifies the finite request, selected-window, dyadic tolerance, regularity handoff, seal, support, provenance, and local naming rows as the displayed selector budget.

Now use Definition 126.474. Every accepted consumer coordinate is determined by those displayed rows, including the BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows. Hence a defect in the factorization has to appear on at least one of those finite rows.

Finally apply Definition 126.476. Since the terminal route is exhausted by the four visible faces and their support rows, quotient stream equality, host equality, selected limits, ambient completeness, aggregate formalstatus, aggregate bridge rows, countable choice, and any fifth object are not admissible witnesses for the defect. □

Theorem 126.481 (StreamName selector-budget defect face-status obstruction)**.** *Let an accepted Phase Real and Completion exit read expose exactly the visible faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

If the read either fails to factor through the selector-budget packet of Definition 126.479 or fails the audit-packet criterion of Definition 126.478, then the obstruction is witnessed by at least one displayed finite request row, selected-window row, dyadic tolerance row, regularity handoff row, $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *row,* hsame *row,* Cont *row,* Pkg *row, local* $\mathsf{NameCert}$ *support row, or by one of the four objectwise local closurestatus and formalstatus records attached to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The obstruction cannot be repaired or witnessed by quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, aggregate formalstatus, aggregate bridge rows, or standard bridge completion.*

Proof. First consider failure of selector-budget factorization. By Definition 126.480, every such failure is already localized on one of the finite selector rows or support rows displayed by the four-face packet. The excluded quotient, limit, completeness, choice, aggregate, and bridge coordinates do not occur in that packet.

Now consider failure of the audit-packet criterion. Apply Definition 126.478. It reads audit validity as exactly the four objectwise local closurestatus and formalstatus records, together with the displayed support rows. Hence a failed audit packet is witnessed either by one of those local face-status records or by one of the finite support rows already named above.

Finally combine the two cases with Definition 126.477 and Definition 126.476. The roadmap theorem keeps full phase exit objectwise over the four displayed faces, while no-fifth-object exhaustion removes every hidden route behind the terminal classifier surface. Thus aggregate formalstatus, aggregate bridge rows, standard bridge completion, quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, and countable choice cannot substitute for a missing finite row or a missing local face-status record. □

Theorem 126.482 (StreamName open-phase face-status obstruction localization)**.** *Let a current Phase Real and Completion exit read expose exactly the four visible faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

If the read fails the roadmap exit audit, then the failure is witnessed by one of those four local closurestatus, formalstatus, leantarget, or bridgestatus records, or by one of the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows.*

No aggregate formalstatus, aggregate bridge row, selected limit, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, countable choice, or standard bridge row can substitute for the missing local witness.*

Proof. First identify the audit surface. By Definition 126.478, the StreamName-side audit criterion is exactly the four local Phase Real and Completion face records together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next identify the roadmap boundary. Apply Definition 126.477. It reads the current exit as the same four-face conjunction over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so failure of the roadmap exit audit must occur on a row carried by that conjunction.

Now import the Real-side componentwise obstruction. The objectwise face-status exit of Definition 38.792 fixes the $\mathsf{Real}^{\uparrow}$ L10 boundary as the four named face records plus the same displayed support rows, and Definition 38.729 says an audit failure cannot be absorbed by an aggregate row when one of those local face records is missing or sub-threshold.

Finally apply Definition 126.476. Since the open-phase route has no hidden replacement coordinate beyond the four displayed faces and their support rows, selected limits, quotient stream equality, ambient completeness, metric completeness, countable choice, aggregate formalstatus, aggregate bridge rows, and standard bridge rows are not available substitutes. The failure is therefore localized to one of the four local status records or to one of the displayed support rows. □

126.34 StreamName L10 Real-Completion Sibling Spine

Theorem 126.483 (StreamName L10 Real-completion sibling spine)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *open-phase consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *endpoint surfaces factors through the sibling spine chapter 301, chapter 291, chapter 38, and chapter 386 before any endpoint seal is public. The spine exposes finite stream windows, regular-rational readback, dyadic tolerance ledgers, Real seal rows, Cauchy-limit seal rows, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows only.*

Proof. Start with Definition 126.441. It sends every Real and Completion finite-window read through the RegSeqRat, DyadicRatCore, and Real sibling surfaces before the read can become seal data.

Next apply Definition 126.446. The open-phase exit boundary identifies the public StreamName, RegSeqRat, DyadicRatCore, Real, observation, transport, route, provenance, and local naming rows, and excludes quotient streams, host sequence equality, selected limits, ambient completeness, metric-completeness data, and standard bridge rows.

Finally attach the Cauchy-limit seal sibling. The endpoint seal is public only after the displayed finite-window readback and dyadic ledger have reached the Cauchy-limit seal surface, so the route is the listed sibling spine rather than an additional endpoint selector. □

126.35 StreamName Real-Completion Gateway Package

Definition 126.484 (StreamName Real-completion gateway carrier)**.** *A* $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet is a finite* BHist *packet*

$$G = (S, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by the open phase;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row anchored to the same finite window;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from that anchored window;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after the displayed readback;*
5. *B records the displayed* BHist *observations on S, D, R, E;*
6. *H records componentwise* hsame *transports for the window, ledger, readback, seal, route, provenance, and local naming rows;*
7. *C records displayed* Cont *routes from S to D, from S, D to R, and from R to E;*
8. *P records* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows for the four faces and their displayed support.*

The packet is a gateway for the finite Real-completion route. It contains no coordinate for quotient stream equality, host sequence equality, a selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or a standard bridge row.*

Theorem 126.485 (StreamName Real-completion gateway obligations)**.** *For the carrier of Definition 126.484, the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *obligations are exactly carrier admission for*

$$G = (S, D, R, E, B, H, C, P, N),$$

StreamName finite-window exposure through S, DyadicRatCore tolerance-ledger exposure through D, RegSeqRat readback exposure through R, RealUp seal exposure through E, displayed BHist *observation through B, componentwise* hsame *transport through H, displayed route replay through C, package provenance through P, and local naming through N.*

Proof. Start with Definition 126.446. It identifies every Phase Real and Completion consumer with the finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, and the displayed observation, transport, route, provenance, and local naming rows.

Next apply Definition 126.448. The certificate repacks the same exit boundary as a four-face read and names exactly the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows used by the gateway packet.

Finally use Definition 126.442. The tolerance ledger is pinned to the StreamName finite window before the RegSeqRat readback or the RealUp seal can be consumed. Therefore the listed rows exhaust the local NameCert obligations for the gateway carrier. □

Theorem 126.486 (StreamName Real-completion gateway non-escape)**.** *No accepted* $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet exports quotient stream equality, host sequence equality, a selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable-choice selection, or a standard bridge row. Every accepted gateway read factors through*

$$S, D, R, E, B, H, C, P, N$$

as displayed in Definition 126.484.

Proof. Begin with the gateway obligation theorem. It says that every public read of the packet is one of $S, D, R, E, B, H, C, P, N$ and that the structural rows only transport, replay, package, and name that displayed inventory.

Use Definition 126.446 and Definition 126.448. Both exclude quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, and standard bridge rows from the same four-face boundary.

It remains to exclude a bypass through the dyadic ledger. Apply Definition 126.442: the ledger is anchored to the StreamName finite window before RegSeqRat or Real consumers can inspect it. Thus the gateway has no second route through which any excluded object can enter. □

Theorem 126.487 (StreamName Real-completion gateway terminality)**.** *Let two accepted* $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway reads expose the same StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, displayed* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. Then the two reads determine the same finite BEDC route through the Real-completion gateway. Conversely, every terminal route available to the current Real-completion open phase is read through exactly those four faces and support rows.*

Proof. First apply Definition 126.449. It states that agreement on the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and displayed support rows determines the same terminal finite BEDC route.

Next use Definition 126.437. The lattice gives the converse direction: every terminal open-phase route is exhausted by the same four faces and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally compare with Definition 126.484. The carrier lists exactly those rows and no additional coordinate. Therefore two gateway reads with the same four faces and support rows determine the same finite BEDC gateway route, and every terminal route in the current Real-completion open phase is read through that packet. □

Theorem 126.488 (StreamName Real-completion gateway ordered source readback)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway read consumes the source in the strict order*

$$S \longrightarrow D \longrightarrow R \longrightarrow E.$$

The StreamName finite window S is read first, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D is anchored to that window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R is available only from S, D, and the* $\mathsf{Real}^{\uparrow}$ *seal E is available only after R. The gateway exports no permutation of these four faces, no bypass through D, and no direct $S \longrightarrow E$ route.*

Proof. Project the carrier of Definition 126.484. Its displayed route rows name S before D, name R only from the paired S, D read, and name E only after the readback row.

Apply Definition 126.485. The NameCert surface exposes the same ordered rows as the accepted gateway packet, so every consumer reads the four faces through the displayed BHist, hsame, Cont, Pkg, and local naming support.

Use Definition 126.442 to rule out a dyadic bypass. The tolerance ledger is attached to the StreamName window before any RegSeqRat or RealUp consumer can inspect the route.

Finally apply Definition 126.487. Since all terminal Real-completion routes factor through the same four-face packet, a consumer cannot reorder the faces or jump directly from S to E without leaving the exported gateway carrier. □

Theorem 126.489 (StreamName Real-completion gateway touchpoint factorization)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet whose source readback follows*

$$S \longrightarrow D \longrightarrow R \longrightarrow E.$$

Then the machine-facing surface of G is exactly the factorization through the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. Equivalently, the only Lean-route positions consumed by the gateway are

`BEDC.Derived.StreamNameUp.RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff`,
`BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness`,
`BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_real_seal_handoff`.

The factorization exports no quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable-choice row, standard bridge row, or bridge-row substitute.*

Proof. Start with Definition 126.488. It fixes the gateway source order as S before D, R only from the paired S, D route, and E only after R.

Next apply Definition 126.485. The accepted packet exposes exactly the four faces S, D, R, E and the support rows B, H, C, P, N, so the gateway has no machine-facing coordinate outside the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ inventory.

Now use Definition 126.453 together with Definition 126.452. These bind the same four-face exit read to exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. The three touchpoints are route positions for the existing window, ledger, readback, and seal rows, not additional carrier data.

Finally invoke Definition 126.486. Since every accepted gateway read factors through $S, D, R, E, B, H, C, P, N$, the cited Lean-route boundary cannot introduce quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, a standard bridge row, or any bridge-row substitute. Thus the ordered gateway packet and the three touchpoints form one exact factorization surface. □

Theorem 126.490 (StreamName Real-completion gateway status exactness)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet. The packet is status-exact for the current Phase Real and Completion boundary exactly when the four public status faces are read as the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *closurestatus and formalstatus records, respectively, with only B, H, C, P, N as support rows. No aggregate token, fifth completion object, quotient stream, selected limit, countable-choice row, metric-completeness datum, or standard bridge row can replace any one of the four faces or enter as an additional status face.*

Proof. Project the gateway carrier of Definition 126.484. Its accepted packet exposes exactly the four faces S, D, R, E and the support rows B, H, C, P, N, so status information has no carrier coordinate outside that displayed list.

Use Definition 126.488. The gateway reads S before D, reads R only from the paired S, D route, and reads E only after R. Thus the four status faces cannot be permuted, fused into an aggregate token, or bypassed through a direct completion row.

Apply Definition 126.500 and Definition 126.516. These theorems identify the same four Phase Real and Completion faces as the StreamName finite window, DyadicRatCore finite dyadic budget, RegSeqRat regularity handoff, and RealUp seal route.

Finally use the two-axis discipline recorded by the roadmap closurestatus blocks. The closurestatus and formalstatus rows are read per object, while B, H, C, P, N only observe, transport, replay, package, and name the displayed faces. Therefore the gateway is status-exact precisely for those four per-object records, with no extra completion, quotient, choice, metric-completeness, selected-limit, or bridge status row hidden in the route. □

Theorem 126.491 (StreamName Real-completion gateway face-record noncollapse)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet. The four public record faces of* G *are exactly the local* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *closurestatus, formalstatus, leantarget, and bridgestatus rows. The support rows* B, H, C, P, N *observe, transport, replay, package, and name those faces, respectively, but they do not create an aggregate formalstatus, replacement leantarget, route-level bridgestatus, standard bridge row, selected limit, quotient stream equality, countable-choice row, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the gateway carrier of Definition 126.484. The accepted packet exposes only the four public faces S, D, R, E and the support rows B, H, C, P, N, so every audit-facing record must be read through that displayed objectwise surface.

Apply Definition 126.490. It identifies the closurestatus and formalstatus records as the four local $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, with B, H, C, P, N remaining support rows rather than status carriers.

Next use Definition 126.489. The Lean-facing route has exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. These are route positions attached to the same four faces, not replacement leantargets or route-level bridgestatus data.

Finally invoke Definition 126.486 and Definition 126.488. The source order $S \longrightarrow D \longrightarrow R \longrightarrow E$ has no bypass through quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, countable choice, a standard bridge row, or any bridge-row substitute. Therefore the four public records remain objectwise face records, while B, H, C, P, N only carry the named support roles. □

Theorem 126.492 (StreamName Real-completion gateway no-choice normal form)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet. Every terminal seal consumer or completion-facing consumer of* G *that is status-exact and machine-facing through the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff has the normal form*

$$S \longrightarrow D \longrightarrow R \longrightarrow E$$

with support rows B, H, C, P, N. Any read mentioning quotient stream equality, host sequence equality, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, a standard bridge row, or a bridge-row substitute lies outside the exported gateway carrier.*

Proof. Use Definition 126.488. It fixes the only consumer order exported by the accepted gateway packet as S before D, R only from the paired S, D read, and E only after R.

Apply Definition 126.490 and Definition 126.491. A status-exact consumer reads the four public status faces as the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ objectwise records, while B, H, C, P, N remain support rows rather than aggregate formalstatus, replacement leantarget, route-level bridgestatus, or additional completion data.

Next use Definition 126.489. The machine-facing route is exactly the three existing touchpoints named in the statement, so no fourth machine coordinate can supply quotient equality, a selected limit, choice, completeness, or bridge data.

Finally invoke Definition 126.486. Since every accepted gateway read factors through $S, D, R, E, B, H, C, P, N$, any consumer that mentions one of the excluded coordinates has left the exported carrier. Thus every terminal seal or completion-facing consumer that remains inside the gateway has precisely the displayed no-choice normal form. □

Theorem 126.493 (StreamName Real-completion gateway support-row minimality)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet for the current Phase Real and Completion boundary. The five support rows B, H, C, P, N are independently necessary for the gateway surface:*

1. *without B, there is no displayed observation surface for the four faces S, D, R, E;*
2. *without H, componentwise transport cannot identify the same finite route through the window, ledger, readback, seal, route, provenance, and local naming rows;*
3. *without C, the ordered route $S \longrightarrow D \longrightarrow R \longrightarrow E$ cannot be replayed inside the packet;*
4. *without P, provenance over the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces is absent;*
5. *without N, the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row for the four faces and their displayed support is absent.*

When the four faces and all five support rows are present, Definition 126.492 gives the unique no-choice normal form. The minimality claim exports no quotient stream equality, host sequence equality, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, standard bridge row, or bridge-row substitute.*

Proof. Project the carrier in Definition 126.484. It displays exactly the four faces S, D, R, E and the five support rows B, H, C, P, N, with no extra coordinate that could replace one of those support rows.

Apply Definition 126.485. The accepted local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ surface assigns B to displayed BHist observation, H to componentwise hsame transport, C to displayed Cont route replay, P to Pkg provenance, and N to local naming. Removing any one row removes exactly the corresponding obligation from the accepted packet surface.

1. The B row is necessary because the carrier and obligation theorem expose the four faces through displayed BHist observations. Without B, the packet may still name S, D, R, E, but it has no displayed observation surface for them.
2. The H row is necessary because the same finite route is compared componentwise through hsame across the window, ledger, readback, seal, route, provenance, and local naming rows. Without H, terminal agreement cannot identify that the read is the same finite route.
3. The C row is necessary by Definition 126.488. The source order $S \longrightarrow D \longrightarrow R \longrightarrow E$ is part of the displayed route replay, so deleting C deletes the internal replay of that order.
4. The P row is necessary by Definition 126.489. The machine-facing factorization is over the four named faces and their existing route positions; without P, the packet has no provenance tying those faces to the gateway package.
5. The N row is necessary by Definition 126.490. Status exactness is read through local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows for the four faces and their support, so without N the gateway lacks the local naming row that records the status surface.

With all five support rows present, the same carrier, obligation, ordered-readback, factorization, and status-exactness theorems place the accepted packet back inside the hypotheses of Definition 126.492. That theorem returns the unique no-choice normal form for the current gateway.

The exclusions are preserved because none of the five support rows is a quotient stream equality, host sequence equality, selected limit, countable-choice row, ambient Real completeness datum, metric-completeness datum, standard bridge row, or bridge-row substitute, and the no-choice normal form already rejects every read mentioning those coordinates. □

Theorem 126.494 (StreamName Real-completion gateway face minimality)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet for the current Phase Real and Completion boundary. The packet satisfies the current gateway read only when all four object faces* S, D, R, E *are present:*

1. *deleting the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face* S *leaves no accepted gateway route;*
2. *deleting the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger face* D *leaves no accepted gateway route;*
3. *deleting the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face* R *leaves no accepted gateway route;*
4. *deleting the* $\mathsf{Real}^{\uparrow}$ *seal face* E *leaves no accepted gateway route.*

This face minimality is internal to the displayed gateway carrier and exports no quotient stream, host sequence equality, selected limit, countable-choice row, metric-completeness datum, standard bridge row, bridge-row substitute, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First identify the accepted gateway surface. By Definition 126.485, an accepted packet carries the four faces S, D, R, E together with the displayed support rows B, H, C, P, N as a local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ package; no extra carrier coordinate is available to stand in for a missing face.

Next read the source order. Apply Definition 126.488. The only gateway route has the form

$$S \longrightarrow D \longrightarrow R \longrightarrow E.$$

Thus the window face is the route source, the tolerance-ledger face is the dyadic budget read after that source, the readback face consumes the paired S, D data, and the seal face is available only after the readback.

Now remove one face. If S is absent, the ordered source read has no StreamName window from which to start. If D is absent, the same route has no DyadicRatCore tolerance ledger to pass from the window into the regular sequence read. If R is absent, there is no RegSeqRat readback consuming the S, D pair. If E is absent, there is no RealUp seal endpoint. In each case the ordered readback theorem leaves no accepted gateway route on the remaining three faces.

It remains to rule out replacement data. By Definition 126.490, the status records are exactly the objectwise $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows; by Definition 38.793, the Real-side current Phase Real and Completion exit certificate also needs each of those four faces. These two minimality reads agree with the gateway obligations, so quotient streams, host sequence equality, selected limits, countable choice, metric-completeness data, standard bridge rows, bridge-row substitutes, and $\mathsf{Real}^{\uparrow}$ completeness cannot fill any deleted face. □

Theorem 126.495 (StreamName Real-completion gateway choice-free face determinacy)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

and

$$G' = (S', D', R', E', B', H', C', P', N')$$

be accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packets for the current Phase Real and Completion boundary. Suppose the four object faces agree through the displayed componentwise* hsame *transports*

$$S\mathsf{hsame}S', \qquad D\mathsf{hsame}D', \qquad R\mathsf{hsame}R', \qquad E\mathsf{hsame}E',$$

and suppose the support rows B, H, C, P, N *and* B', H', C', P', N' *agree as local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows for those transported faces. Then every terminal Real-completion consumer of* G *and every terminal Real-completion consumer of* G' *determine the same finite public* $\mathsf{Real}^{\uparrow}$ *seal row through the route*

$$S \longrightarrow D \longrightarrow R \longrightarrow E.$$

The determined public seal row is independent of quotient stream equality, host sequence equality, selected limits, countable choice, metric-completeness data, standard bridge rows, bridge-row substitutes, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 126.491. For each accepted packet, every public record is read through the four objectwise faces and the support rows B, H, C, P, N or B', H', C', P', N'. The displayed hsame transports identify the two StreamName windows, DyadicRatCore ledgers, RegSeqRat readbacks, and RealUp seal faces as the same public face record.

Next use Definition 126.492 for both terminal consumers. Each consumer has the ordered no-choice route

$$S \longrightarrow D \longrightarrow R \longrightarrow E$$

on G, and the transported route with faces S', D', R', E' on G'. Since the four faces agree componentwise, the two ordered routes have the same public RealUp endpoint.

The support rows cannot introduce a second route. By Definition 126.493, the rows B, H, C, P, N are exactly the observation, transport, replay, provenance, and local naming rows needed for the displayed gateway surface. Their agreement as local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ rows therefore makes the support inventory for G and G' the same finite inventory, not an additional choice parameter.

Finally apply Definition 126.494. No replacement face can stand in for S, D, R, E, and no excluded coordinate can fill the RealUp endpoint. Thus the two terminal consumers determine the same finite public $\mathsf{Real}^{\uparrow}$ seal row, with no quotient stream equality, host sequence equality, selected limit, countable-choice row, metric-completeness datum, standard bridge row, bridge-row substitute, or ambient Real completeness coordinate added to the route. □

Theorem 126.496 (StreamName Real-completion gateway audit-touchpoint separation)**.** *Let*

$$G = (S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{StreamName}^{\uparrow}$ *Real-completion gateway packet for the current Phase Real and Completion boundary. The audit-facing object records of G are exactly the four objectwise closurestatus, formalstatus, leantarget, and bridgestatus faces for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The machine-facing touchpoints of G are exactly the three route positions named by Definition 126.489: the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.*

These two readings do not collapse into one another. The audit-facing records do not create a machine route coordinate, and the machine-facing touchpoints do not create an aggregate formalstatus, replacement leantarget, route-level bridgestatus, standard bridge row, selected limit, quotient stream equality, countable-choice row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 126.490. It identifies the audit-facing status surface as the four objectwise records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with closurestatus, formalstatus, leantarget, and bridgestatus read locally on those faces rather than through a single gateway-wide status row.

Next use Definition 126.489. The machine-facing route is exhausted by the three named touchpoints: the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. These are positions in the route, not objectwise closurestatus, formalstatus, leantarget, or bridgestatus records.

The two inventories cannot be identified. By Definition 126.491, the public face records remain objectwise over S, D, R, E and their support rows B, H, C, P, N. Thus a touchpoint cannot replace an audit record, and an audit record cannot supply an additional machine-facing handoff.

Finally invoke Definition 126.492 and Definition 126.486. Every accepted gateway read stays inside the ordered no-choice route and the displayed support rows, so neither reading can introduce aggregate formalstatus data, replacement leantargets, route-level bridgestatus data, standard bridge rows, selected limits, quotient stream equality, countable-choice rows, metric-completeness data, or $\mathsf{Real}^{\uparrow}$ completeness claims. □

Theorem 126.497 (StreamName CauchyWitnessLedger exit route)**.** *Any* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *completion-facing read of the current Phase Real and Completion exit boundary of* $\mathsf{StreamName}^{\uparrow}$ *factors through a finite witness-ledger route. The source is the StreamName finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal named by the open-phase exit certificate, together with the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Only after those four faces and their displayed support rows are fixed may the ledger expose the ordered route

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with structural rows H, C, P, N in the accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet.*

The route exports no quotient stream, no selected limit, no host sequence equality, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. Begin with Definition 126.446. It identifies every current Phase Real and Completion exit read with the StreamName finite window, the RegSeqRat readback, the DyadicRatCore ledger, the RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply Definition 126.448. The certificate repacks the same four faces as a Real-completion exit packet and keeps the tolerance, readback, and seal coordinates on the finite StreamName surface rather than on a quotient stream or selected limit.

Then use Definition 126.449. Once the finite window, tolerance ledger, readback, seal, and support rows are fixed, there is no second terminal route for a completion-facing consumer to choose.

Finally apply Definition 902.5. The accepted ledger consumes that finite source through Q, then B, then S, and only then exposes K, with H, C, P, N preserving transport, replay, provenance, and local naming. Since both the StreamName exit boundary and the ledger factorization have no coordinate for quotient streams, selected limits, host sequence equality, ambient Real completeness, metric completeness, countable choice, or standard bridges, the displayed finite witness-ledger route is the whole export. □

Theorem 126.498 (StreamName CauchyWitnessLedger exit saturation)**.** *Every* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *read of a current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *exit packet is saturated by exactly the four finite faces named by Definition 126.448 and Definition 126.449: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal, with only the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

After those four faces are fixed, the witness ledger has only the ordered route

$$Q \longmapsto B \longmapsto S \longmapsto K$$

from Definition 902.5, with structural rows H, C, P, N preserving transport, replay, provenance, and local naming.

The saturated packet exports no quotient stream, no selected limit, no host sequence equality, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. First apply Definition 126.497. It places every completion-facing ledger read on the finite StreamName, DyadicRatCore, RegSeqRat, and Real route, and it already excludes quotient streams, selected limits, host sequence equality, ambient completeness, metric-completeness data, countable-choice rows, and standard bridge rows from the route.

Next use Definition 126.448 and Definition 126.449. The certificate names exactly the four finite faces of the Real-completion exit packet, while terminality says that once those faces and the displayed BHist, hsame, Cont, Pkg, and NameCert rows are fixed, no alternate finite exit boundary remains.

Finally apply Definition 902.5. The witness ledger consumes the fixed packet through Q, then B, then S, and then K, with H, C, P, N carrying only the structural support for that route. Since neither the four-face boundary nor the factorization supplies any further public coordinate, the ledger saturates the current StreamName exit boundary rather than enlarging it. □

Theorem 126.499 (StreamName CauchyWitnessLedger terminal determinacy)**.** *Let two completion-facing* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *reads of the current* $\mathsf{StreamName}^{\uparrow}$ *Real-completion exit packet agree on the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the two reads expose the same terminal $\mathsf{Real}^{\uparrow}$ *seal row. The ledger may order the shared route through*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with structural rows H, C, P, N*, but it cannot introduce quotient streams, selected limits, host sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, or a standard bridge.*

Proof. First apply Definition 126.497. Each read factors through the same finite StreamName, DyadicRatCore, RegSeqRat, and Real faces, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows available to the ledger.

Next use Definition 126.498. Since both reads agree on those four faces and support rows, the witness ledger has no additional public coordinate from which a distinct terminal seal row could be read.

Now invoke Definition 126.449. The open Phase Real and Completion boundary is terminal once the finite window, tolerance ledger, readback, seal, and support rows are fixed, so both reads must consume the same terminal $\mathsf{Real}^{\uparrow}$ seal row.

Finally apply Definition 902.5. The fixed order $Q \longmapsto B \longmapsto S \longmapsto K$, with H, C, P, N preserving transport, replay, provenance, and local naming, makes the ledger a consumer of the four finite faces rather than a second source of seal data. The exclusions in the route and saturation statements therefore remain absent from the terminal read. □

126.36 StreamName Real-completion four-object exit route

Theorem 126.500 (StreamName Real-completion four-object exit route)**.** *Every current Phase Real and Completion exit read named by the roadmap for* DyadicRatCore*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *factors through exactly the four finite sibling faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The public rows of the exit boundary are only the finite StreamName window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the RealUp seal, and the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

No quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row belongs to this exit boundary.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectExitRoute`

Proof. Begin with Definition 126.446. It identifies every current Phase Real and Completion consumer with a finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply Definition 126.448 and Definition 126.449. The certificate names the four finite faces of the exit packet, while terminality says that once those faces and support rows are fixed, the consumer has no second terminal route through which to read the boundary.

Now use Definition 126.483. The L10 spine places the StreamName route against the sibling surfaces consumed by the Real and Completion phase, so the roadmap's four objects are read as sibling faces rather than as a later completion theorem.

Finally apply Definition 126.497. The witness-ledger route consumes the same finite packet without adding quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows. Therefore the partial exit boundary, certificate, four-face terminality, sibling spine, and witness-ledger route exhaust the roadmap boundary for $\mathsf{DyadicRatCore}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Theorem 126.501 (StreamName open-phase four-face minimality)**.** *For the current Phase Real and Completion exit boundary, the four displayed faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are minimal. Deleting any one of the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, or RealUp seal prevents the exit read from satisfying the roadmap boundary named by Definition 126.500. No additional fifth face may replace a deleted one.*

The minimal packet still carries only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It does not admit quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, metric-completeness data, or bridge substitutes as replacement faces.*

Proof. Start from Definition 126.500. It states the exit read through exactly the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the displayed support rows.

Use Definition 126.446 and Definition 126.449. The boundary theorem fixes the current roadmap surface, while terminality says that the four displayed faces are the terminal route through which that surface is read.

Now separate the four roles. The StreamName window supplies the finite observation window, the DyadicRatCore ledger supplies the tolerance account, the RegSeqRat readback supplies the regular sequence reading, and the RealUp seal supplies the terminal Real-facing boundary.

Finally apply Definition 126.483. The sibling spine keeps those roles distinct and blocks replacement by quotient streams, host equality, selected limits, ambient completeness, choice, metric-completeness rows, or bridges. Therefore deleting any one face removes a required row, and a fifth face cannot restore the current four-face exit boundary. □

Theorem 126.502 (StreamName Real-completion exit consumer exhaustion)**.** *Every downstream consumer of the current Phase Real and Completion exit packet, including completion-facing witness-ledger reads and terminal Real-facing seal reads, factors through the same four finite faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The consumer boundary exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectExitConsumerExhaustion`

Proof. Start with Definition 126.500. It names the current Phase Real and Completion exit packet as exactly the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next compose Definition 126.446 with Definition 126.448. The open-phase boundary identifies the finite consumer coordinates, and the certificate repacks those coordinates as the public Real-completion exit packet.

Apply Definition 126.449. Once the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and displayed support rows are fixed, a terminal Real-facing seal read has no second coordinate through which to leave the packet.

Finally use Definition 126.497. Completion-facing witness-ledger reads consume the same finite packet, so the witness-ledger route and the terminal Real-facing seal route are exhausted by the four faces and the displayed support rows. The exclusions in the cited routes remove quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable-choice rows, and standard bridge rows from every downstream consumer. □

Theorem 126.503 (StreamName Real-completion four-object exit minimality)**.** *Every current Phase Real and Completion exit read loses its certified boundary if any one of the four displayed sibling faces is omitted. Without the* $\mathsf{StreamName}^{\uparrow}$ *finite window there is no scheduled observation source; without the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger there is no tolerance or observation row; without the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback there is no regular-rational spine; without the local* $\mathsf{Real}^{\uparrow}$ *seal there is no public real-name row.*

Thus the four faces

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

are the minimal visible BEDC support for the current Phase Real and Completion boundary.

Proof. Apply Definition 126.500. It identifies the boundary packet with exactly the four displayed faces and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Next use Definition 126.502. Every downstream completion-facing or Real-seal-facing consumer reads only that packet, so deleting one face deletes the corresponding public coordinate rather than revealing a hidden replacement route.

The necessity of the individual faces is read from Definition 126.446. The $\mathsf{StreamName}^{\uparrow}$ finite window supplies the scheduled observation source, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies tolerance and observation rows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback supplies the regular-rational spine, and the local $\mathsf{Real}^{\uparrow}$ seal supplies the public real-name row.

Finally apply Definition 126.449. Terminality says the accepted read is fixed by those four faces and the displayed support rows, so no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completeness datum, countable-choice row, or standard bridge row can replace a missing face. □

Theorem 126.504 (StreamName Real-completion four-object terminal pullback)**.** *Let two current Phase Real and Completion consumer reads expose the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the same displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the two reads determine the same terminal finite BEDC pullback through those four faces. Conversely, every completion-facing consumer and every Real-seal-facing consumer of the current exit packet factors uniquely through that pullback.

The pullback exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectTerminalPullback`

Proof. First apply Definition 126.500. It identifies the current exit packet with exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For uniqueness of the terminal pullback, use Definition 126.502 together with Definition 126.449. Consumer exhaustion removes every downstream coordinate outside the four faces, and four-face terminality identifies two accepted reads once those faces and support rows agree.

For the converse direction, invoke Definition 126.497. Completion-facing witness-ledger reads consume the same finite packet, while the Real-seal-facing reads are already terminal by the open-phase four-face statement.

Combining the route theorem, consumer exhaustion, four-face terminality, and witness-ledger route gives a single terminal finite BEDC pullback for all accepted completion-facing and Real-seal-facing reads. The same cited exclusions leave no place for quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable-choice rows, or standard bridge rows. □

Theorem 126.505 (StreamName Real-completion terminal exactness)**.** *A current Phase Real and Completion read is publicly accepted exactly when it factors through the terminal finite pullback whose four faces are*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and whose support rows are only

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, componentwise agreement on the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and displayed support rows is necessary and sufficient for the public classifier surface of the read.

The exact terminal boundary exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. For necessity, begin with Definition 126.500. It identifies every accepted current Phase Real and Completion read with exactly the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, supported only by BHist, hsame, Cont, Pkg, and NameCert.

Apply Definition 126.504. The accepted read must factor through the terminal finite pullback determined by those four faces and rows, so any public acceptance witness has the displayed terminal boundary.

For sufficiency, use Definition 126.518. It states that exposing precisely the four dependency faces with the displayed support rows is enough for the current paper-side Real-completion boundary.

Finally apply Definition 126.514. Componentwise agreement on the four faces and support rows determines the public classifier surface, and the axis separation theorem Definition 126.506 removes quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, choice, and bridge rows from the boundary. Thus terminal factorization through the four L10 faces is necessary and sufficient for public acceptance. □

Theorem 126.506 (StreamName Real-completion exit axis separation)**.** *The current Phase Real and Completion exit packet supplies only the paper-side finite four-face boundary through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and the support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It does not raise the verification-axis token of any of those four chapters, does not assert ambient $\mathsf{Real}^{\uparrow}$ *completeness, does not supply metric-completeness data, does not add a countable-choice row, and does not create a standard bridge row.*

Consequently, every downstream consumer may use this exit packet only as a paper-side finite boundary. Claims about verification status, bridge status, or completeness for any of the four faces must still be read from that face's own closurestatus block, Lean target, and stated bridge boundary.

Proof. Apply Definition 126.500. It fixes the exported Phase Real and Completion packet as exactly the four displayed faces and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Next use Definition 126.502 and Definition 126.504. Consumer exhaustion and terminal pullback say that every accepted downstream read factors through that same finite packet, with no additional coordinate for quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, or a standard bridge.

Finally read the packet through the two-axis discipline of section 9467.2 and section 9203.3. The four-face boundary is a manuscript-side route, while the verification status of each face is reported only by its own Lean target and closurestatus entry. Therefore the exit packet can be consumed as a finite paper-side boundary, but it cannot by itself change verification-axis tokens, supply completeness data, or create bridge claims for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionExitAxisSeparation`

Theorem 126.507 (StreamName Real-completion face status pullback)**.** *For any accepted current Phase Real and Completion exit read through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

the paper-side boundary pulls back the four face status records only as references. The closure axis grade, formalstatus token, Lean target, and bridge status of each face remain exactly the entries declared in that face's own closurestatus block.

The exit packet promotes no face status. It introduces no fifth status coordinate, no aggregate verification token, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, and no bridge row beyond the face records already declared by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the four-object route of Definition 126.500. It identifies the accepted exit read with the four finite faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply the terminal pullback of Definition 126.504. Every downstream completion-facing or Real-seal-facing read factors through the same finite pullback, so status information can enter the packet only by reference to the already named faces.

Now use the axis separation theorem, Definition 126.506. It blocks the exit packet from changing verification-axis tokens, supplying completeness data, or creating bridge claims for any of the four faces.

Finally read the result through the roadmap two-axis discipline of section 9467.2. Closure axis, formalstatus token, Lean target, and bridge status are reported by each face's own closurestatus block; the exit packet merely gathers those references and adds no promotion or fifth status coordinate. □

Theorem 126.508 (StreamName four-face local record exit boundary)**.** *Assume the four face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are read objectwise from their own local closurestatus blocks. The* $\mathsf{DyadicRatCore}^{\uparrow}$ *face contributes its* bridgeChecked *bridgeChecked finite-route record, the* $\mathsf{StreamName}^{\uparrow}$ *face contributes its* bridgeChecked *finite-window record, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face contributes its* axiomClean *paperBridge regularity-handoff record, and the* $\mathsf{Real}^{\uparrow}$ *face contributes its* bridgeChecked *seal record. Assume also that the finite handoff rows read by the current Phase Real and Completion roadmap are exactly the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the displayed support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then the current Phase Real and Completion exit boundary is exactly the pullback of those four local records: four independent status faces, with the three machine-route positions naming the finite-window, dyadic-ledger, and regular-to-real seal handoffs, and with no additional completion coordinate.

This boundary is not an aggregate audit-clean token and not an aggregate bridge token. It refuses quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, and bridge rows.*

Proof. First apply Definition 126.500. It identifies the current exit read with the four displayed faces and the finite support rows BHist, hsame, Cont, Pkg, and NameCert.

Read the local closurestatus records objectwise. The $\mathsf{DyadicRatCore}^{\uparrow}$ record supplies the dyadic finite-route face at bridgeChecked with bridgeChecked status, and the $\mathsf{RegSeqRat}^{\uparrow}$ record supplies the regularity-handoff face at axiomClean with paperBridge status. The $\mathsf{StreamName}^{\uparrow}$ record supplies the finite-window face at bridgeChecked, and the $\mathsf{Real}^{\uparrow}$ record supplies the seal face at bridgeChecked.

Next use Definition 126.506. Because these grades are read from four separate local closurestatus blocks, the packet cannot turn paper closure into verification status or verification status into paper closure.

Now apply Definition 126.507. The four local records pull back as four independent face references. The exit read therefore has no aggregate status coordinate and no fifth record that could hide a completion theorem.

It remains to bind the finite handoff rows. Use Definition 126.519: the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff are route positions for the existing window, ledger, readback, and seal rows. Together with the four-object route, axis separation, and face-status pullback, these positions make the four local records exactly the current roadmap exit boundary.

The exclusions are the same exclusions carried by the cited route and separation theorems. Quotient stream equality, selected limits, ambient completeness, metric-completeness data, countable choice, host sequence equality, and bridge rows are absent from the four-face packet. □

Theorem 126.509 (StreamName four-face audit-clean threshold readback)**.** *Assume the local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are read from their own closurestatus blocks, and assume each face reports formal verification at least* auditClean*. Then the StreamName-side Phase Real and Completion exit boundary satisfies the roadmap verification threshold objectwise: the accepted boundary is exactly the four local audit-clean face records together with the finite StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

This threshold readback contributes no aggregate formalstatus*, no aggregate bridge token, no completeness row, no quotient stream, no selected limit, and no metric-completeness datum.*

Proof. Start with the four-face local record boundary of Definition 126.508. It identifies the StreamName-side exit boundary with the four local records and the displayed finite handoff rows, rather than with a route-level status token.

Apply the face-status pullback, Definition 126.507. Each closure grade, formalstatus token, Lean target, and bridge row is read only from the corresponding local face closurestatus block. The audit-clean threshold therefore enters only through the explicit hypothesis that every face is at least auditClean, checked object by object.

Next use the exit-axis separation theorem, Definition 126.506. It prevents the paper-side boundary from manufacturing an aggregate verification coordinate, bridge token, completeness row, quotient stream, selected limit, or metric-completeness datum.

Finally compare with the Real-side exit witness, Definition 38.492. That witness states the current Phase Real and Completion exit criterion in exactly the same four local faces, with each verification record read at least auditClean from its own closurestatus block. Therefore the StreamName-side boundary satisfies the roadmap threshold precisely as an objectwise four-face readback and not as any additional completion or bridge datum. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFourFaceAuditCleanThresholdReadback`

Theorem 126.510 (StreamName Real-completion objectwise threshold equivalence)**.** *A StreamName-side Phase Real and Completion exit read satisfies the current objectwise exit threshold if and only if the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *each supply the required local theory-closure row and an audit-clean verification row, read from that face's own closurestatus block.*

This equivalence is only the local four-face threshold. It supplies no aggregate formalstatus*, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, apply Definition 126.509. It reads any accepted StreamName-side threshold witness objectwise through the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with each verification row checked in that face's own closurestatus block.

Use the same read through Definition 38.494. The Real-side formulation identifies the current Phase Real and Completion threshold with exactly those local theory-closure rows and audit-clean verification rows, so the StreamName-side witness has the same objectwise coordinates.

For the reverse direction, suppose the four local face records each supply the required local theory-closure row and an audit-clean verification row. Apply Definition 126.505 to bind these records to the terminal four-object Real-completion boundary.

Next apply Definition 126.518. The dependency boundary is precisely the four faces together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows, so those local records are sufficient for the StreamName-side current exit read.

Finally compare again with Definition 38.494. The two formulations have the same four local faces and the same local closure and audit-clean threshold, while the cited StreamName terminal and dependency exactness theorems keep the packet from acquiring any aggregate formalstatus, aggregate bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness claim. □

Theorem 126.511 (StreamName Real-completion exit status coherence)**.** *A current Phase Real and Completion exit read satisfies the roadmap objectwise exit threshold exactly when the same read is the StreamName four-object route readback of the four local closurestatus records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, the threshold reading and the local-status reading are the same objectwise four-face status packet: the $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic budget face, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity-handoff face, and the* $\mathsf{Real}^{\uparrow}$ *seal face.*

This coherence theorem adds no aggregate formalstatus*, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the threshold-to-status direction, apply Definition 126.510. It rewrites a current exit-threshold witness as the four local theory-closure and audit-clean verification rows read from the closurestatus blocks for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 126.512. The same four local records are exactly the StreamName-side local face-status readback. Use Definition 126.516 to identify those faces with the dyadic budget, finite window, regularity handoff, and seal route, and to retain only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the status-to-threshold direction, start from the four local closurestatus records read through the StreamName four-object route. The local face-status readback gives precisely the same four records, and status-ledger exactness says that no additional status coordinate is present in the accepted exit packet.

Apply Definition 126.510 in the reverse direction. Since the four local records supply exactly the required local theory-closure rows and audit-clean verification rows, the read satisfies the roadmap objectwise exit threshold.

Finally use Definition 126.506. Axis separation prevents the coherence statement from turning the four-face packet into an aggregate formalstatus, an aggregate bridge row, quotient stream equality, a selected limit, countable choice, metric completeness, or a $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Theorem 126.512 (StreamName Real-completion local face status readback)**.** *The current StreamName-side Phase Real and Completion exit packet reads back the four existing local closurestatus records:*

1. $\mathsf{DyadicRatCore}^{\uparrow}$ *supplies its* bridgeChecked *bridgeChecked dyadic finite-route record,*
2. $\mathsf{StreamName}^{\uparrow}$ *supplies its* bridgeChecked *finite-window record,*

3. $\mathsf{RegSeqRat}^{\uparrow}$ *supplies its* axiomClean *paperBridge regularity-handoff record,*

4. $\mathsf{Real}^{\uparrow}$ *supplies its* bridgeChecked *local seal record.*

These four records are only objectwise local face data. They add no aggregate formalstatus*, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 126.507. It reads closure grade, formalstatus token, Lean target, and bridge status from each face's own closurestatus block, so the four displayed entries remain local records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. For the dyadic face, the local readback is the same $\mathsf{DyadicRatCore}^{\uparrow}$ bridge anchor recorded by *BEDC.Derived.DyadicRatCoreUp.DyadicRatCo*

Next use Definition 126.508. The exit boundary is exactly the same four-face read: local theory-closure rows together with their actual local verification rows, with no fifth status coordinate. Apply Definition 126.519 to bind the finite window, dyadic route, regularity handoff, and real seal to the existing BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 126.506. Because the four closurestatus records are cited only as local face records, the exit packet cannot promote them into an aggregate verification status, an aggregate bridge, quotient stream equality, a selected limit, countable choice, metric completeness, or a $\mathsf{Real}^{\uparrow}$ completeness theorem. This is precisely the claimed local face status readback. □

Theorem 126.513 (StreamName Real-completion objectwise exit certificate)**.** *A current Phase Real and Completion exit certificate for the StreamName-side route consists exactly of the four objectwise local records for*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

This certificate is necessary and sufficient for the paper-side roadmap exit boundary: the four local records supply the face statuses, the support rows supply the finite route data, and terminal acceptance is read through the four-object Real-completion pullback.

The certificate exports no quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus*, aggregate bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 126.512. It identifies the four local records as the objectwise closurestatus reads for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and it keeps those records from becoming an aggregate status coordinate.

Next use Definition 126.510. The current Phase Real and Completion threshold is satisfied exactly by those four local records together with their local theory-closure and verification rows, read object by object.

For sufficiency, apply Definition 126.505. Terminal exactness says that the four records, carried by BHist, hsame, Cont, Pkg, and NameCert, are enough for public acceptance of the four-object Real-completion boundary. The same theorem gives necessity, since every accepted current read must factor through that terminal four-face packet.

Finally apply Definition 126.506 and Definition 126.517. Axis separation removes quotient stream equality, selected limits, choice, metric-completeness data, and $\mathsf{Real}^{\uparrow}$ completeness from the certificate, while bridge nonaggregation prevents the four local records from forming an aggregate bridge row. Hence the displayed four records and five support rows are exactly the roadmap exit certificate and nothing more. □

Theorem 126.514 (StreamName Real-completion classifier pullback)**.** *Let two current Phase Real and Completion exit reads be accepted by the four-object route through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

They have the same public classifier surface exactly when their finite StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows agree componentwise.*

No quotient stream, host sequence equality, selected limit, ambient Real$^{\uparrow}$ *completeness theorem, countable-choice row, metric-completeness datum, or standard bridge row can distinguish the two reads at the public classifier surface.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameRealCompletionClassifierPullback`

Proof. For the forward direction, apply the four-object route of Definition 126.500. The route exposes only the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the displayed support rows.

Use the terminal pullback, Definition 126.504. Once those rows agree, the two accepted reads factor through the same finite pullback, so there is no second public coordinate on which the classifier can separate them.

For the reverse direction, start from componentwise agreement of the displayed rows. The face-status pullback of Definition 126.507 keeps the four face records as references to the same face surfaces, rather than adding a status coordinate or a completion principle.

The Lean-facing handoff anchors for the finite window, dyadic ledger, and regular-to-real seal certify the same BEDC route positions used by the paper packet. Together with the route and terminality theorems, these handoffs repack the agreed rows as one public classifier surface.

The exclusions follow from Definition 126.506. Since quotient stream equality, host sequence equality, selected limits, ambient completeness, choice, metric-completeness data, and standard bridges are absent from the packet, none can distinguish the accepted reads. □

Theorem 126.515 (StreamName Real-completion terminal semantic readback)**.** *Any current Phase Real and Completion terminal read through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

carries the terminal SemanticNameCert *surface whose carrier is the readback row, whose pattern is exactly the four faces together with the terminal and readback rows, and whose ledger is the readback* Pkg *row.*

This terminal semantic readback exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate status, aggregate bridge row, or Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameOpenPhaseFourFaceExitReadback`

Proof. First apply Definition 126.500. It fixes the terminal read as the four-face route through StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$, carried by the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 126.504. Terminal pullback identifies every accepted current read with the same finite four-object boundary, so the terminal row and the readback row are not additional objects outside the route.

Now apply Definition 126.514. The classifier surface is determined componentwise by the four faces and the support rows, hence the public readback carrier is exactly the readback row and its ledger is exactly the readback Pkg row.

The Lean readback target packages the same BHist, hsame, Cont, Pkg, and SemanticNameCert rows. Because it uses the four faces together with the terminal and readback rows as the whole pattern, it adds no fifth object and cannot export quotient streams, selected limits, choice, metric completeness, aggregate status, aggregate bridges, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 126.516 (StreamName Real-completion status ledger exactness)**.** *Every current Phase Real and Completion exit read through the StreamName four-object route has exactly the four displayed status faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite dyadic budget,*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *pointwise certified window,*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff,*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal route.*

This status ledger is exact as a four-object paper-side boundary: it exports no ambient completeness theorem, no quotient stream equality, no host real equality, and no verification-axis upgrade for any of the four faces.

Proof. Start with Definition 126.500. It fixes the accepted exit read as the four sibling faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Apply Definition 126.501. The minimality statement prevents any one of the four faces from being deleted or replaced by a fifth face, so the status ledger has exactly the four named entries: finite dyadic budget, pointwise certified window, regularity handoff, and seal route.

Next use Definition 126.504. Terminal pullback identifies all accepted downstream reads through the same finite boundary once the four faces and support rows agree.

Now apply Definition 126.506 and Definition 126.507. Axis separation blocks verification promotion, completeness claims, quotient stream equality, and host real equality from entering the packet, while face status pullback says that closure and verification records are imported only as references to the four faces.

Finally use Definition 126.514. The public classifier surface is determined componentwise by the same four faces and displayed support rows, so the ledger is exact and axis-separated for every current Phase Real and Completion exit read. □

Theorem 126.517 (StreamName Real-completion bridge nonaggregation)**.** *For the current Phase Real and Completion exit packet, bridge status or audit-clean status on any subset of the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

does not supply a standard bridge row for the aggregate packet. Even if one or more face records expose paperBridge, bridgeChecked, auditClean, or stronger local status in their own closurestatus blocks, the four-face packet remains a paper-side exit boundary unless an explicit aggregate bridge theorem names a standard bridge for the whole Phase Real and Completion route.

Thus the packet cannot infer a bridge row from the $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic budget, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, or any proper subfamily of these faces.*

Proof. Start with face-status pullback, Definition 126.507. It says that the closure grade, formal-status token, Lean target, and bridge status of each face are imported only as references to that face's own closurestatus block.

Next apply exit-axis separation, Definition 126.506. The current exit packet has no coordinate for turning a local bridge or audit-clean entry into a packet-level standard bridge row.

Use the three-touchpoint handoff route of Definition 126.519. The StreamName finite-window handoff, DyadicRatCore ledger handoff, and RegSeqRat-to-Real seal handoff are route positions for displayed BHist, hsame, Cont, Pkg, and NameCert rows; they are not an aggregate bridge constructor.

Finally read the four local closurestatus blocks for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Those blocks control their own bridge axes only. Therefore no subset of the face records aggregates into a standard bridge for the current Phase Real and Completion packet, and only an explicit aggregate bridge theorem may change the bridge axis of that packet. □

Theorem 126.518 (StreamName Real-completion dependency exactness)**.** *A current Phase Real and Completion exit read is accepted exactly when it exposes the four finite dependency faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and supports them only by the displayed rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The four faces are necessary and sufficient for the paper-side Real-completion boundary: the $\mathsf{StreamName}^{\uparrow}$ *face supplies the pointwise certified finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face supplies the finite dyadic budget, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face supplies the regularity handoff, and the* $\mathsf{Real}^{\uparrow}$ *face supplies the seal route.*

No quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, countable-choice row, or bridge row is part of this exact dependency boundary.*

Proof. For necessity, begin with Definition 126.500. It identifies any accepted current exit read with the four displayed dependency faces and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Apply Definition 126.501 and Definition 126.516. Minimality prevents deleting or replacing any one face, while status-ledger exactness names the same four dependencies as the finite dyadic budget, pointwise certified window, regularity handoff, and seal route.

For sufficiency, use Definition 126.504. Once the four faces and displayed support rows are exposed, the accepted read factors through the terminal finite pullback of the Real-completion boundary.

Finally apply Definition 126.514. The classifier pullback says the public surface is determined by exactly those component rows, so no additional equality, limit, completeness, choice, metric, or bridge coordinate is needed or available. This gives the exact necessary-and-sufficient dependency boundary for the current paper-side Real-completion exit read. □

Theorem 126.519 (StreamName Real-completion three-touchpoint handoff route)**.** *Every current Phase Real and Completion exit read whose finite-window row is witnessed by the StreamName singleton finite-window handoff, whose tolerance ledger is witnessed by the DyadicRatCore RegSeqRat bridge handoff exactness, and whose real-seal row is witnessed by the RegSeqRat Real seal*

handoff factors through the existing four faces

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The resulting paper obligation carries only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The three machine-side route names identify the handoff path for the later formal target. They do not add a Lean marker at this paper site and do not export quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, or a standard bridge row.*

Proof. Begin with Definition 126.516. It says that every accepted current exit read has exactly the four status faces: the $\mathsf{DyadicRatCore}^{\uparrow}$ finite dyadic budget, the $\mathsf{StreamName}^{\uparrow}$ pointwise certified window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity handoff, and the $\mathsf{Real}^{\uparrow}$ seal route.

Next apply Definition 126.500. The four status faces are the same four finite sibling faces of the Real-completion exit boundary, and the support surface is only BHist, hsame, Cont, Pkg, and NameCert.

It remains to bind the named handoffs to that paper boundary. The prose handoff anchor of Definition 126.452 identifies the three machine-side route names as the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. These are route witnesses for the existing window, ledger, and seal rows, not extra coordinates.

Combining the exact status ledger, the four-object exit route, and the handoff anchor therefore repacks any read witnessed by the three named route positions through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ alone. The same cited exclusions prevent quotient equality, selected limits, completeness data, choice, metric-completeness rows, or bridge rows from entering the obligation. □

Theorem 126.520 (StreamName Real-completion four-face Lean witness composition)**.** *Every current Phase Real and Completion exit read whose four faces are accepted by the StreamName four-object route and whose terminal read is accepted by the terminal pullback has machine-facing support exactly through the three primary touchpoints*

$$\textit{Definition}\ 126.173,\quad \textit{Definition}\ 291.28,\quad \textit{Definition}\ 301.22.$$

The composed witness exposes only the support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It exports no quotient stream equality, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, aggregate formalstatus token, or standard bridge row.*

Proof. Apply Definition 126.500. The accepted four faces are therefore exactly the current $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ exit faces, carried by the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 126.504. Since the terminal read is accepted, every downstream Real-completion read factors through that same finite pullback; no second coordinate can be recovered for quotient stream equality, selected limits, choice, ambient completeness, metric-completeness data, or bridge rows.

Next use Definition 126.519. It identifies the machine-facing support with the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff, which are precisely the three touchpoint route positions named in the statement.

Finally apply Definition 126.506. Closure status and verification status remain objectwise reports attached to the four accepted faces and terminal pullback. Hence the composed witness has no aggregate formalstatus token and exposes only the named BHist, hsame, Cont, Pkg, and NameCert support rows. □

Theorem 126.521 (StreamName Real-completion four-face audit-clean composition)**.** *Assume a current Phase Real and Completion exit read is accepted by the StreamName four-object route and by the terminal pullback, and assume the four local face records for*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

each report formal verification at least auditClean *from that face's own closurestatus block. Then the StreamName-side packet satisfies the same objectwise exit threshold as the Real-side current phase exit witness of Definition 38.492: the accepted packet is exactly the four local audit-clean face records together with the finite StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

This is only an objectwise exit-threshold composition. It adds no aggregate formalstatus*, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the four-face Lean witness composition of Definition 126.520. It binds an accepted four-object route and an accepted terminal pullback to the three machine-facing touchpoints for the StreamName finite-window row, the DyadicRatCore-to-RegSeqRat ledger row, and the RegSeqRat-to-Real seal row.

Next apply the audit-clean threshold readback of Definition 126.509. Under the hypothesis that $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ each report at least auditClean locally, the StreamName-side packet reads the roadmap verification threshold object by object through those four local records.

Use Definition 126.510 to compare that readback with the current objectwise exit-threshold formulation. Then apply the Real-side current phase exit witness, Definition 38.492, which states the same threshold in the Phase Real and Completion exit criterion.

Finally close the boundary exclusions with Definition 126.506 and Definition 126.517. Axis separation prevents the objectwise threshold from producing quotient stream equality, a selected limit, countable choice, metric-completeness data, or a $\mathsf{Real}^{\uparrow}$ completeness theorem, while bridge nonaggregation prevents the four local records from forming a packet-level bridge row. The composed packet therefore satisfies exactly the Real-side objectwise exit threshold and nothing beyond that boundary. □

Theorem 126.522 (StreamName Real-completion no-choice route factorization)**.** *Every current Phase Real and Completion exit read through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

factors through the finite four-face packet together with exactly the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, the route factorization has no coordinate for a countable-choice row, selected limit, quotient stream equality, host sequence equality, metric-completeness datum, or standard bridge row.

Any attempted reading of one of those rows lies outside the packet. It is therefore not part of the current paper obligation carried by the StreamName Real-completion exit route.

Proof. Start with the status ledger exactness theorem, Definition 126.516. It identifies every accepted current exit read with exactly the four status faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and states that the ledger exports no ambient completeness theorem, quotient stream equality, host real equality, or verification-axis promotion.

Apply the four-object exit route theorem, Definition 126.500. This fixes the support surface of the route as BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, with no further packet coordinate.

Next use the terminal pullback, Definition 126.504. Terminality says that any accepted downstream read agreeing on the four faces and support rows factors through the same finite boundary, so a proposed selector, quotient, completeness, or bridge row cannot be recovered as an additional public component of the route.

Finally bind the named handoff positions by Definition 126.519. The StreamName finite-window row, DyadicRatCore tolerance ledger, and RegSeqRat-to-Real seal are witnesses for existing route entries rather than new data rows.

Reading the result through the roadmap two-axis discipline of section 9467.2, the factorization reports only the current closure and verification records already attached to the four faces. Since choice rows, selected limits, quotient stream equality, host sequence equality, metric-completeness data, and standard bridge rows are absent from that boundary, each such attempted row is outside the packet. □

Theorem 126.523 (StreamName Real-completion failure localization)**.** *Let a current Phase Real and Completion exit read be presented through the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Assume the read cannot be accepted as the public four-face boundary. Then the obstruction is localized to failure of at least one displayed face record or at least one displayed support row

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, the failed read has no fifth completion object in which the obstruction can be stored.

In particular, the obstruction is not quotient stream equality, selected limit data, an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, a metric-completeness datum, a countable-choice row, a standard bridge row, or any hidden fifth face.*

Proof. Begin with dependency exactness, Definition 126.518. Acceptance is exactly exposure of the four finite dependency faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with only the support rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Now apply terminal pullback, Definition 126.504. When those faces and rows are present, every accepted downstream read factors through the same finite boundary. Therefore a read that cannot be accepted must fail before this terminal pullback is available, at one of the listed face records or support rows.

Next use no-choice route factorization, Definition 126.522. The factorization has no coordinate for countable choice, selected limits, quotient stream equality, host sequence equality, metric-completeness data, or standard bridge rows, so none of those rows can serve as the missing obstruction carrier.

Finally apply exit-axis separation, Definition 126.506. Axis separation blocks an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem or verification upgrade from becoming an extra paper-side face. The only remaining obstruction sites are the four displayed face records and the five displayed support rows, so the failed read has no hidden fifth object. □

Theorem 126.524 (StreamName Real-completion budget-selector lock)**.** *Let a current Phase Real and Completion consumer take only finite observations mediated by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *request. Then the consumer still factors through exactly the four Real-completion boundary faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The mediated read carries only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The observation-budget selector therefore locks into the same four-face boundary as the StreamName Real-completion exit route. It does not introduce quotient stream equality, host real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, or a standard bridge row.*

Proof. Begin with the exact status ledger of Definition 126.516. Any accepted current Phase Real and Completion exit read has exactly the four status faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no additional status coordinate.

Now insert the finite observation-budget mediation. The StreamName Real observation-budget factorization of Definition 126.363 says that every accepted $\mathsf{StreamName}^{\uparrow}$ finite-window packet consumed by a $\mathsf{RealObservationBudget}^{\uparrow}$ precision request follows the visible route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ using only BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next apply Definition 611.20. The accepted budget request reaches a seal read only through its finite precision, window, dyadic, regular handoff, and seal rows, so the selector supplies a finite access route rather than a new completion object or hidden observation schedule.

Finally use the exit-axis separation theorem, Definition 126.506. Since the four-object boundary has no quotient stream equality, host real equality, selected limit, ambient completeness, metric-completeness datum, countable-choice row, or standard bridge row, the budget-mediated consumer cannot obtain any of those rows by passing through the selector. The same four boundary faces and the same five support rows are therefore locked for every such finite observation-budget read. □

Theorem 126.525 (RegSeqObservationBudget exit pullback)**.** *Let an accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *consumer read its* $\mathsf{Real}^{\uparrow}$ *row as a current Phase Real and Completion exit read. Then the consumer factors through the same L10 four-face boundary*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

as the StreamName Real-completion exit route. Its support surface is only

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Thus the observation-budget consumer is a consumer of the displayed StreamName window, dyadic tolerance ledger, RegSeqRat readback, Real seal, and support rows. It does not receive quotient stream equality, host real equality, hidden diagonal selectors, unbounded schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable choice, or aggregate bridge rows.*

Proof. First order the budget route. Theorem 1053.4 forces the accepted $\mathsf{RegSeqObservationBudget}^{\uparrow}$ consumer to display the StreamName window before the readback, tolerance, selector, or seal rows are used. Theorem 1053.6 then reaches the $\mathsf{Real}^{\uparrow}$ budget row only after the window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic tolerance, and diagonal selector rows have already been displayed.

Now read that ordered route through the current Real and Completion exit. Apply Definition 126.500. The exit packet is exactly the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next apply Definition 126.504. Since the RegSeqObservationBudget consumer reads its $\mathsf{Real}^{\uparrow}$ row as a current exit read, the ordered budget route factors through the same terminal finite pullback as every accepted downstream Real-completion read.

Finally use Definition 126.506 together with Definition 1053.8. Axis separation removes quotient stream equality, host real equality, ambient completeness, metric-completeness data, countable choice, and bridge rows from the exit boundary, while public consumer export says the accepted budget packet exports only the displayed finite route. The hidden selector and unbounded schedule alternatives therefore have no carrier coordinate, and the four faces with the five support rows exhaust the consumer boundary. □

Theorem 126.526 (StreamName dual observation-budget exit cofinality)**.** *Let a current Phase Real and Completion consumer be mediated both by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *request and by an accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *request over the same displayed StreamName finite window, DyadicRatCore endpoint-radius ledger, RegSeqRat readback and classifier rows, RealUp seal row, and support surface*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then the two mediated consumers determine the same finite four-face exit boundary

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

In particular, the RealObservationBudget-mediated read is cofinal with the RegSeqObservationBudget-mediated read inside the terminal pullback: any accepted finite exit-boundary read reached by either budget route factors through the same four faces and the same displayed support rows.

The common boundary contains no quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate $\mathsf{formalstatus}$ *row, aggregate bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply the budget-selector lock, Definition 126.524. It sends every accepted $\mathsf{RealObservationBudget}^{\uparrow}$ mediation for the current Phase Real and Completion consumer through exactly the four displayed faces with only BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next apply the RegSeqObservationBudget exit pullback, Definition 126.525. Since the second mediation reads its $\mathsf{Real}^{\uparrow}$ row as the same current exit read, it also factors through the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the same five support rows.

Now compare the two factorizations with the terminal pullback of Definition 126.504. The hypotheses identify the displayed window, endpoint-radius ledger, readback and classifier rows,

seal row, and support surface, so terminality identifies the two mediated consumers as reads of one finite pullback rather than two independent selector coordinates.

Finally use exit-axis separation, Definition 126.506. The common pullback is only the paper-side four-face boundary. It therefore exports no quotient stream, selected limit, choice row, metric-completeness datum, aggregate verification row, aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. The two observation-budget routes are cofinal precisely because both terminate at that same finite boundary. □

126.37 StreamName finite-observation section packet

Theorem 126.527 (StreamName finite-observation section)**.** *Every certified finite observation section of a* $\mathsf{StreamName}^{\uparrow}$ *window is exhausted by the displayed* ProbeBundle *membership row, the* UnaryHistory *row, the* RatHistoryClassifier *row, the componentwise* hsame *transport, the displayed* Cont *route,* Pkg *provenance, and the local* NameCert *row of the StreamName packet. The section contributes no regularity witness, quotient stream equality, completion row, host-index equality, selected limit, or ambient completeness coordinate.*

Proof. Start from the certified observation section inside the finite StreamName window. Apply Definition 126.251; it projects any selected certified unary observation to the finite ProbeBundle row, InBundle membership row, UnaryHistory row, pointwise RatHistoryClassifier row, hsame comparison, Cont route, Pkg provenance, and NameCert provenance.

The membership and unary-history rows identify the observation as a finite section of the window rather than as a host-indexed stream. The pointwise classifier and hsame rows identify exactly the rational-history data visible at that section, while Cont, Pkg, and NameCert only replay, package, and name the same finite surface.

The exclusions are therefore inherited from the classifier exhaustion theorem. Regularity witnesses, quotient stream equality, completion rows, host-index equality, selected limits, and ambient completeness coordinates are not fields of the section packet, so no certified finite observation section can export them. □

Theorem 126.528 (StreamName finite-observation constant seed)**.** *For a constant rational stream section* $\mathsf{RatConstStream}_d$*, any two certified observations of the same finite* $\mathsf{StreamName}^{\uparrow}$ *window read the same hidden* $\mathsf{Rat}^{\uparrow}$ *seed* d*. The public section row is therefore the common* hsame *class of the two observed histories, together with the same finite membership, unary-history,* Cont*,* Pkg*, and* NameCert *support rows.*

Proof. Apply Definition 126.48 to the first certified observation. The observed history of $\mathsf{RatConstStream}_d$ is identified by hsame with the hidden seed d.

Apply the same witness theorem to the second certified observation. Its observed history is again identified by hsame with d, using only the certified unary observation row and the constant rational stream definition.

Now use Definition 126.49. It composes the two point-sameness witnesses through the seed and gives the same public history row for both certified observations. The finite membership, unary-history, Cont, Pkg, and NameCert rows are the support rows of the section packet, so the constant section reads one RatUp seed across all certified observations of the window. □

Theorem 126.529 (StreamName finite-observation Real exit handoff)**.** *The finite-observation section packet of Definition 126.527 and the constant-seed packet of Definition 126.528 are exactly the* $\mathsf{StreamName}^{\uparrow}$ *contribution to the Phase Real and Completion four-object exit surface. When the*

exit route is read through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, the StreamName side supplies only finite observation rows, constant-seed determinacy where the stream is constant, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Proof. First use Definition 126.179. It places every finite-window $\mathsf{Real}^{\uparrow}$ seal read on the StreamName side over the carried endpoints, finite bundle membership, unary observation, and pointwise rational-history classifier rows.

Combine that scope theorem with Definition 126.527. The section packet is therefore the whole StreamName finite-observation contribution before the RegSeqRat and Real-facing rows are consumed.

For constant rational stream sections, insert Definition 126.528. The same finite section surface now also records that all certified observations read the same hidden RatUp seed, without adding a quotient stream, selected limit, or ambient completeness coordinate.

Finally apply Definition 126.500. The four-object route reads the Phase Real and Completion boundary through exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Thus the section and constant-seed packets are precisely the StreamName-side finite observation handoff into that exit surface. □

Theorem 126.530 (StreamName RegSeqRat tail-budget section lock)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *observation section consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget read and then by a finite* $\mathsf{Real}^{\uparrow}$ *seal factors through the same displayed section packet. The packet consists of the* ProbeBundle *membership row, the* InBundle *membership row, the* UnaryHistory *observation row, the* RatHistoryClassifier *endpoint row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the displayed* hsame *transports, the* Cont *routes,* Pkg *provenance, and the local* NameCert *rows. It exports no quotient stream equality, host sequence equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or second dyadic source.*

Proof. The local section inventory is fixed by Definition 126.527. A certified finite observation section exposes exactly the bundle membership, in-bundle membership, unary-history, rational-history classifier, hsame, Cont, Pkg, and local NameCert rows listed there.

The handoff to the Phase Real and Completion exit surface is the one in Definition 126.529. It says that the StreamName contribution to the four-object route is precisely the finite observation packet before the downstream DyadicRatCore, RegSeqRat, and Real rows are read.

Now route the consumer through Definition 301.109. A tail-budget read is ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and then $\mathsf{Real}^{\uparrow}$, so the dyadic tolerance ledger and the RegSeqRat regularity and classifier rows consume the same finite StreamName section rather than a separate source.

Finally use Definition 126.400. The shared-tail consumer keeps the RegSeqRat readback and the finite Real seal on the same public finite-window route. Since the cited packets contain no fields for a quotient stream equality, host sequence equality, selected limit, ambient Real completeness principle, metric-completeness datum, or second dyadic source, none of those objects can be inserted between the section packet and its tail-budget or seal consumers. □

Theorem 126.531 (StreamName finite-observation section reindex coverage)**.** *Every certified reindexing of a finite* $\mathsf{StreamName}^{\uparrow}$ *observation section preserves the same displayed section packet and the same Phase Real and Completion exit route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The reindexed consumer receives only the transported finite observation rows, the constant scheduled-subwindow coherence when the source is constant, the downstream tail-budget rows,*

and the finite $\mathsf{Real}^{\uparrow}$ *seal rows; it receives no quotient stream equality, host-index equality, selected limit, second dyadic source, or ambient completeness row.*

Proof. Begin with the local finite observation packet from Definition 126.527. It identifies the displayed section solely by the bundle membership, unary-history, rational-history classifier, hsame, Cont, Pkg, and local NameCert rows, with no quotient, host-index, limit, or completeness field available to transport.

Apply Definition 126.20. Certified window reindexing composes inside the StreamName certificate surface, so the consumer reads the reindexed section through the same public observation packet rather than through a second stream source or a host-index equality.

For constant scheduled subwindows, use Definition 126.288. The scheduled window and the reindexed window expose the same constant observation row, so reindexing does not add a selected limit or a second dyadic source before the downstream consumer sees the packet.

Finally combine Definition 126.529 with Definition 126.530. The first theorem fixes the four-object Phase Real and Completion route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; the second keeps the RegSeqRat tail-budget read and the finite Real seal on that same finite section. Thus certified reindexing preserves coverage of the section packet without opening any quotient stream, host-index, selected limit, second-source, or ambient-completeness row. □

Theorem 126.532 (StreamName finite-observation selected transport)**.** *If a certified finite* $\mathsf{StreamName}^{\uparrow}$ *observation section is transported componentwise by displayed* hsame *rows and replayed through the same* Cont *route, then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the transported section through the same four-object Phase Real and Completion route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The transport exports no quotient stream equality, host-index equality, selected limit, second dyadic source, or ambient completeness row.

Proof. Begin with Definition 126.527. A certified finite observation section is exhausted by the finite bundle, membership, unary-history, rational-history classifier, componentwise hsame, displayed Cont, provenance, and local naming rows.

Apply Definition 126.529. It identifies the StreamName contribution to the Phase Real and Completion exit surface as the same finite section before the DyadicRatCore, RegSeqRat, and Real-facing rows are consumed.

Finally use Definition 126.531. Certified reindexing and componentwise transport preserve the same section packet and the same four-object route, so a transported consumer cannot open quotient stream equality, host-index equality, selected limits, second dyadic sources, or ambient completeness rows. □

Theorem 126.533 (StreamName finite-observation diagonal selector boundary)**.** *Every diagonal-limit or Real-completion consumer that reads a finite* $\mathsf{StreamName}^{\uparrow}$ *observation section through an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet factors through the same finite route: the StreamName finite observation section, the selector rows* B, S, W, D, R, E*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The boundary preserves the four-object Phase Real and Completion exit surface through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; it exports no quotient stream equality, host real equality, selected limit, countable choice, hidden diagonal selector, ambient completeness, metric-completeness row, bridge row, or second source for the dyadic observations.*

Proof. Start with the StreamName section itself. By Definition 126.527, a certified finite observation section exposes only the finite membership, history, classifier, hsame, Cont, Pkg, and local NameCert rows of the StreamName packet.

The handoff theorem Definition 126.529 places that section on the Phase Real and Completion route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The reindex coverage theorem Definition 126.531 keeps any certified reindexed consumer on the same finite section surface.

Now read the accepted selector. By Definition 782.10, any diagonal-limit or Real-completion consumer of the selector factors through the displayed finite route $B \to S \to W \to D \to R \to E$ together with its visible support rows. By Definition 782.9, the dyadic observations in D are exhausted by the selected finite $\mathsf{StreamName}^{\uparrow}$ windows in W.

Therefore the selector consumes the existing finite StreamName section as the source for its dyadic ledger rather than creating a second source. Since the cited section, handoff, reindexing, diagonal-boundary, and exhaustion theorems contain no fields for quotient stream equality, host real equality, selected limits, countable choice, hidden diagonal selectors, ambient completeness, metric-completeness rows, or bridge rows, none of those coordinates can be inserted between the finite section and its diagonal or Real-completion consumer. □

Theorem 126.534 (StreamName finite-observation selector coverage exhaustion)**.** *Every accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *consumer of a finite* $\mathsf{StreamName}^{\uparrow}$ *observation section is exhausted by the displayed selector rows* B, S, W, D, R, E*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The accepted selector route exports no quotient stream equality, host real equality, selected limit, countable choice, hidden diagonal selector, ambient completeness, metric-completeness row, bridge row, or second dyadic source.*

Proof. Begin with the finite section. By Definition 126.527, the StreamName side of a certified finite observation exposes only the bundle membership, unary-history, rational-history classifier, hsame, Cont, Pkg, and local NameCert rows.

Next pass to the four-object exit surface using Definition 126.529. The finite section is read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the DyadicRatCore ledger, RegSeqRat readback, and terminal Real seal are downstream consumers of the same finite StreamName observation.

Now apply Definition 782.9. For an accepted selector packet, the dyadic observations in D are already exhausted by the selected finite StreamName windows in W. Thus the displayed selector rows B, S, W, D, R, E cover the observation route without opening a second dyadic source.

Finally combine Definition 782.10 with Definition 126.533. These boundary theorems place every compatible diagonal-limit or Real-completion consumer on the same four-object route and the same visible support rows. Since none of the cited packets carries quotient stream equality, host real equality, selected limits, countable choice, hidden diagonal selectors, ambient completeness, metric-completeness rows, bridge rows, or a second dyadic source, no such coordinate is exported by the accepted selector coverage surface. □

126.38 StreamName RegSeqRat Real budget factorization

Theorem 126.535 (StreamName RegSeqRat Real budget factorization)**.** *Every current* $\mathsf{Real}^{\uparrow}$*-and-Completion consumer that reads a StreamName window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly the four finite faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with only the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, the budget boundary for the open L10 route is the finite StreamName window face, the DyadicRatCore tolerance-ledger face, the RegSeqRat readback face, the RealUp seal face, and the BEDC support rows. No fifth consumer face is needed to read the present Real-and-Completion route.

Proof. First apply Definition 126.500. It names the present Real-and-Completion exit read as the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Next use Definition 126.501. The StreamName window, DyadicRatCore ledger, RegSeqRat readback, and RealUp seal are not optional coordinates of a larger object; each is one required face of the same finite route.

Now invoke Definition 126.504. It identifies any two accepted consumer reads that expose the same four faces and support rows with the same terminal finite BEDC pullback.

Finally read the Real seal through Definition 38.314. The Real-side endpoint is still a finite-window obligation, so the budget route has no ambient completion coordinate beyond the four faces and the displayed support rows. □

Theorem 126.536 (StreamName RegSeqRat Real budget non-escape)**.** *The budget factorization of Definition 126.535 exports no quotient stream equality, no host sequence equality, no selected limit, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, no metric-completeness datum, no countable-choice row, no Classical datum, and no standard bridge row. Every downstream consumer receives only the finite StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, and the rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Proof. Start from Definition 126.535. It reduces the consumer route to the four finite faces and the displayed BEDC support rows.

Apply Definition 126.500 and Definition 126.501. These results list the exclusions already present at the exit boundary and prevent a fifth face from replacing any required coordinate.

Use Definition 126.504 for downstream reads. The terminal pullback has no coordinate in which a quotient equality, host sequence equality, selected limit, completeness datum, choice row, Classical datum, or bridge row could be stored.

Finally cite Definition 38.314. The Real-facing read is exhausted by finite windows, so the budget boundary remains a finite BEDC route rather than an ambient completeness assertion. □

Theorem 126.537 (StreamName RegSeqRat Real budget route exactness)**.** *For every current Phase Real and Completion consumer read, acceptance by the open L10 budget route is equivalent to factoring through exactly the four finite faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

with only the support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Under this equivalence, equality of the public classifier surface is exactly componentwise agreement of the finite StreamName window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the RealUp seal, and those five support rows.

No quotient stream, host sequence equality, selected limit, completeness datum, countable-choice row, Classical datum, or standard bridge row participates in the accepted route or in the public classifier equality.

Proof. For the forward direction, apply Definition 126.535. An accepted consumer read is reduced to the four displayed finite faces and to the BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 126.536. The non-escape boundary removes quotient stream equality, host sequence equality, selected limits, ambient or metric completeness data, countable choice, Classical data, and standard bridge rows from the route packet.

For the reverse direction, start with a read that factors through the displayed four faces and support rows. The terminal pullback of Definition 126.504 repacks that data as the unique finite BEDC pullback available to current completion-facing and Real-seal-facing consumers, so the read is accepted by the open L10 budget route.

Finally apply Definition 126.514. It identifies public classifier equality with componentwise agreement on exactly the four faces and the displayed support rows, while the same non-escape exclusions leave no hidden classifier coordinate outside them. □

Theorem 126.538 (StreamName Real-seal observation factorization)**.** *For every current Phase Real and Completion consumer read, a Real-facing observation is accepted exactly when it factors through the finite StreamName window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Under this acceptance test, equality of Real-facing observation classifiers is componentwise agreement on exactly those four faces and five support rows.

The observation packet has no quotient stream coordinate, host limit, ambient completeness datum, choice row, Classical datum, or bridge row. It is therefore a consumer statement over the current four-face Real-and-Completion route, not an appeal to an external completion object.

Proof. First apply Definition 126.535. It gives the forward factorization through the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and the displayed BEDC support rows.

Next use Definition 126.537. The same finite route is also sufficient for acceptance, and its classifier equality is exactly componentwise agreement on the four faces and five support rows.

Now invoke Definition 126.536. This removes quotient streams, host sequence equality, selected limits, ambient or metric completeness data, countable choice, Classical data, and bridge rows from the observation packet.

Finally use Definition 38.314. The Real-facing seal is read through finite windows, so the combined observation remains a finite BEDC route through the current Phase Real and Completion faces. □

Theorem 126.539 (StreamName RegSeqRat Real terminal pullback naturality)**.** *Any two current Phase Real and Completion terminal consumer reads that preserve the finite StreamName window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

induce the same accepted four-face terminal pullback route. Moreover, any transport of terminal consumer reads preserving exactly those four faces and support rows remains on that accepted route, with the same public classifier surface.

The statement does not assert quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, choice data, Classical data, or standard bridge rows. It is only the naturality of the existing finite terminal pullback route under support-row-preserving transport.*

Proof. First apply Definition 126.535. It identifies the terminal consumer read with the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use Definition 126.537. Since the two reads preserve those faces and rows, both are accepted by the same open L10 route, and classifier equality is exactly componentwise agreement on the displayed data.

Now invoke Definition 126.538. The Real-facing observations of the two reads therefore factor through the same finite seal boundary and add no hidden coordinate beyond the four faces and support rows.

Finally cite Definition 38.715. The Real-side terminal-budget naturality transports the shared finite budget through terminal consumer maps, so any support-row-preserving transport stays on the same accepted four-face pullback route. The non-escape boundary from Definition 126.536 keeps quotient equality, host equality, selected limits, completeness data, choice, Classical data, and bridge rows outside the route. □

Theorem 126.540 (StreamName Real-completion budget synchronization)**.** *Every accepted current Phase Real and Completion read synchronizes the finite StreamName window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

into one shared budget boundary. Equivalently, any transport preserving the StreamName window face, the DyadicRatCore tolerance-ledger face, the RegSeqRat readback face, the RealUp seal face, and those support rows preserves the accepted public classifier surface.

This synchronized boundary carries no quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, metric-completeness datum, countable-choice row, Classical datum, or standard bridge row.*

Proof. First apply Definition 126.535. It places every accepted Real-and-Completion read on the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Next use Definition 126.537. The accepted public classifier surface is exactly componentwise agreement on those faces and rows, so no extra coordinate can distinguish two reads once that displayed data agrees.

Now invoke Definition 126.538. The Real-facing observation is not a separate boundary: it factors through the same finite StreamName window, tolerance ledger, RegSeqRat readback, RealUp seal, and support rows.

Finally apply Definition 126.539. Any transport preserving those four faces and support rows remains on the accepted terminal pullback route and therefore preserves the same public classifier surface. The four cited faces are thus synchronized into one shared budget boundary. □

Theorem 126.541 (StreamName RegSeqRat Real terminal four-face pullback)**.** *Let two current Phase Real and Completion budget reads first pass through a common finite refinement. Suppose that, after this refinement, each read exposes the same finite StreamName window, the same*

$\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback section, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same transported support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the two reads determine the same terminal four-face pullback route through

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the same public classifier surface.

The common-refinement route carries no quotient stream coordinate, no selected limit, no host equality, no choice datum, no metric-completeness datum, no standard bridge row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion. It is a terminality statement for the current finite L10 faces and their BEDC support rows.*

Proof. First apply Definition 126.535 to both refined reads. Each read is placed on exactly the StreamName window face, DyadicRatCore tolerance-ledger face, RegSeqRat readback face, RealUp seal face, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use Definition 126.537. Since the common refinement identifies the four faces and the five support rows for the two reads, acceptance by the open L10 route is the same for both reads, and the public classifier surface is equality on precisely that displayed data.

Now invoke Definition 126.539. Transport along the common finite refinement preserves the support rows, so both refined reads remain on the same accepted terminal pullback route.

Finally cite Definition 126.540. The synchronized boundary identifies the StreamName window, tolerance ledger, RegSeqRat readback, RealUp seal, and BEDC support rows as one shared budget boundary, leaving no coordinate in which quotient streams, selected limits, host equality, choice, metric completeness, bridge rows, or ambient RealUp completeness could distinguish the two routes. □

126.39 StreamName Tail-Compatible Finite-Window Packet

Theorem 126.542 (StreamName tail-compatible window request)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be public* $\mathsf{StreamName}^{\uparrow}$ *rational stream rows, let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite observation window, and let* $n : \mathsf{BHist}$ *be a certified unary observation with* $\mathsf{InBundle}(n, \mathcal{P})$*. A tail-compatible window request from the L10 countable-name gateway to a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or a finite-window* $\mathsf{Real}^{\uparrow}$ *seal is determined by exactly the following finite rows:*

1. *the observation index* n *together with its* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$ *witnesses;*
2. *the tail-selector route carrying the selected StreamName observation window;*
3. *the finite-window budget surface for the same* $\mathcal{P}$*; and*
4. *the pointwise handoff row* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *side.*

No other source row is part of the request. In particular, the request has no coordinate for quotient stream equality, selected limits, host real equality, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tail selector route of Definition 126.202. It fixes the selected StreamName window and the downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback rows through one finite transport packet.

Next read the finite observation row through Definition 126.203. For the certified index n, the transported source row is exactly the bundled $\mathsf{UnaryHistory}(n)$ and $\mathsf{InBundle}(n, \mathcal{P})$ evidence together with $\mathsf{RatHistoryClassifier}(s(n), t(n))$.

Now attach the budget surface by Definition 126.390. It places the same finite ProbeBundle, membership, unary-history, observation-budget, and tail-budget rows before any regularity, dyadic, or Real-seal consumer row.

Finally apply Definition 126.170. The pointwise handoff has only StreamName carrier, finite-window classifier, membership, unary-history, and rational-history classifier slots. The displayed rows therefore determine the request, and the cited non-escape clauses exclude quotient, limit, host-real, metric-completeness, and ambient completeness data. □

Theorem 126.543 (StreamName tail-compatible window coverage)**.** *Every accepted finite-window handoff from* $\mathsf{StreamName}^{\uparrow}$ *to a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or to a finite-window* $\mathsf{Real}^{\uparrow}$ *seal factors through the tail-compatible window request of Definition 126.542, together with the shared support rows*

$$\mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The factorization is finite: it is read from the selected BHist *observation index, the tail selector route, the finite-window budget row, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the displayed support rows. It does not use a quotient stream class, a chosen limit, host real equality, or an ambient completion theorem.*

Proof. Begin with an accepted finite-window StreamName handoff. By Definition 126.204, the handoff decomposes into the public finite-window classifier, bundled UnaryHistory/InBundle rows, pointwise rational-history classifier rows, the accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine, DyadicRatCore endpoint rows, and the finite $\mathsf{Real}^{\uparrow}$ ledger.

The rowwise source is then the request from Definition 126.542. Its determining data are the certified BHist observation index, the selected tail route, the finite-window budget surface, and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Next use Definition 126.205. It reads every consumer-facing $\mathsf{Real}^{\uparrow}$ row through the same finite StreamName-to-RegSeqRat packet and the same DyadicRatCore endpoint rows.

Finally apply Definition 126.500. The exit boundary adds only the visible BHist, hsame, Cont, Pkg, and NameCert support needed by the four sibling faces. Combining these route statements gives the displayed finite factorization and leaves quotient classes, selected limits, host real equality, and ambient completion outside the accepted handoff. □

Theorem 126.544 (StreamName tail-compatible window packet)**.** *The tail-compatible finite-window packet is exactly the tuple consisting of the unary index row, the bundle-membership row, the tail-selector route, the finite-window budget surface, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the support rows*

$$\mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Any accepted downstream $\mathsf{RegSeqRat}^{\uparrow}$ *readback or finite-window* $\mathsf{Real}^{\uparrow}$ *seal read is classified by equality of these displayed rows and by no additional stream-level coordinate.*

Proof. Begin with Definition 126.542. It identifies the request by the certified unary observation, membership in the finite bundle, the selected tail route, the finite-window budget surface, and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Apply Definition 126.543. Every admitted downstream finite-window consumer route factors through that request together with the displayed hsame, Cont, Pkg, and NameCert support rows.

Finally use the pointwise transport and budget handoff rows, Definition 126.204 and Definition 126.390. These rows leave no second stream source, quotient stream class, selected limit, host real equality, metric-completeness datum, or ambient completion coordinate, so equality of the displayed tuple is exactly packet equality for downstream reads. □

Theorem 126.545 (StreamName tail-compatible window nonescape)**.** *The tail-compatible finite-window packet exported from* $\mathsf{StreamName}^{\uparrow}$ *to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *carries only the request rows of Definition 126.542 and the support rows of Definition 126.543. It exports no quotient stream equality, no selected limit, no host real equality, no metric-completeness datum, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 126.542. The request is determined by the certified finite observation, tail-selector route, finite-window budget surface, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Apply Definition 126.543. Any accepted finite-window consumer route factors through that request and the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Finally use Definition 126.500. The four-object exit boundary consumes the same finite StreamName, DyadicRatCore, RegSeqRat, and RealUp faces without adding quotient stream equality, host sequence equality, selected limits, metric-completeness data, countable choice, standard bridges, or ambient completeness. Hence the tail-compatible packet has no escaping coordinate of the listed kinds. □

Theorem 126.546 (StreamName tail-compatible window terminality)**.** *Every accepted finite-window handoff from* $\mathsf{StreamName}^{\uparrow}$ *to a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or to a finite-window* $\mathsf{Real}^{\uparrow}$ *seal factors terminally through the request rows of Definition 126.542 and the support rows of Definition 126.543.*

If two accepted finite-window consumers expose the same finite observation, tail-selector route, budget surface, pointwise handoff, and support rows, then they classify through the same tail-compatible window route. The terminal packet exports no quotient stream equality, selected limit, host real equality, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the request theorem. By Definition 126.542, the finite-window request is determined by the certified observation row, the tail-selector route, the finite-window budget surface, and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff row.

Next add the shared support. Apply Definition 126.543; every accepted handoff to a $\mathsf{RegSeqRat}^{\uparrow}$ readback or finite-window $\mathsf{Real}^{\uparrow}$ seal factors through the same request together with the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now compare two consumers with the same visible inventory. The request rows agree by the first step, and the support rows agree by the second step, so the two consumers replay the same finite StreamName window route into the downstream readback or seal.

Finally use Definition 126.545. It excludes quotient stream equality, selected limits, host real equality, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the packet. Thus the displayed request and support rows are terminal for finite-window consumers. □

Theorem 126.547 (StreamName tail-compatible cofinal bundle lock)**.** *Let* $s, t : \mathsf{BHist} \to \mathsf{BHist}$ *be public* $\mathsf{StreamName}^{\uparrow}$ *rational stream rows, let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite bundle of certified unary observation requests, and suppose one StreamName tail-selector route serves every bundled request* n *equipped with* $\mathsf{UnaryHistory}(n)$ *and* $\mathsf{InBundle}(n, \mathcal{P})$*. Every finite family of bundled downstream reads into a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or a finite-window* $\mathsf{Real}^{\uparrow}$ *seal factors through one displayed packet consisting of exactly the following rows:*

1. *the bundled unary index rows n, their* $\mathsf{UnaryHistory}(n)$ *certificates, and their* $\mathsf{InBundle}(n, \mathcal{P})$ *witnesses;*
2. *the shared tail-selector route serving the bundle;*
3. *the shared finite-window budget surface for $\mathcal{P}$;*
4. *the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows* $\mathsf{RatHistoryClassifier}(s(n), t(n))$ *for the bundled indices; and*
5. *the support rows* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

The packet is a finite cofinal lock for the bundled reads: any accepted downstream read through the same tail-compatible route is classified by its displayed bundle row and the shared support rows. The lock contains no quotient stream equality, no selected limit, no host real equality, no metric-completeness datum, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 126.542 to each bundled request. For every certified index in $\mathcal{P}$, the request row is the unary observation, its membership witness, the shared tail route, the finite-window budget surface, and the corresponding pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Next apply Definition 126.543 to every accepted downstream handoff in the finite family. The same support rows hsame, Cont, Pkg, and $\mathsf{NameCert}$ route each admitted read through the request rows just displayed.

Use Definition 126.544 to identify the common finite packet. Since the bundle is finite, collecting the unary rows, membership witnesses, pointwise handoffs, the shared route, the shared budget surface, and the shared support rows produces one displayed packet rather than a separate hidden route for each consumer.

Finally combine Definition 126.546 with Definition 126.545. Terminality makes the displayed packet the unique route for consumers with the same visible inventory, and nonescape excludes quotient stream equality, selected limits, host real equality, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the bundle lock. □

Theorem 126.548 (StreamName tail-compatible cofinal exhaustion)**.** *Let $s, t : \mathsf{BHist} \to \mathsf{BHist}$ be public* $\mathsf{StreamName}^{\uparrow}$ *rational stream rows, and let $\mathcal{P} : \mathsf{ProbeBundle}$ be the finite packet of certified unary observations served by the shared tail-compatible selector of Definition 126.547. Every finite family of accepted downstream reads from that same bundled packet into a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or a finite-window* $\mathsf{Real}^{\uparrow}$ *seal is terminally exhausted by the bundled unary rows, their* $\mathsf{InBundle}$ *witnesses, the shared tail selector, the shared budget surface, the pointwise* $\mathsf{RatHistoryClassifier}$ *handoff rows, and the support rows* $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$.

Equality of these rows is exactly equality of the consumer route. No quotient stream equality, selected limit, host real equality, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate remains available to distinguish two accepted finite-window consumers over the packet.*

Proof. First use Definition 126.547. It collects the bundled unary rows, their membership witnesses, the shared tail selector, the shared budget surface, pointwise rational-history classifier rows, and the support rows into one finite packet for the family of downstream reads.

Next apply Definition 126.543. Each accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback or finite-window $\mathsf{Real}^{\uparrow}$ seal route factors through that packet, so the consumer has no source coordinate beyond the displayed rows.

Then apply Definition 126.546. If two accepted consumers expose the same bundled rows and support rows, terminality identifies their finite-window route through the same packet.

Finally use Definition 126.545. The excluded quotient, limit, host-real, metric-completeness, and ambient-completion coordinates are not packet rows, so no remaining downstream distinction can be read after the displayed finite inventory agrees. □

Theorem 126.549 (StreamName RegSeqRat triple-overlap source cocycle)**.** *Let $\mathcal{W}_0, \mathcal{W}_1, \mathcal{W}_2$ be certified finite* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose each pair carries a certified finite overlap, the three pairwise restrictions agree on every certified unary observation, selected* RatHistoryClassifier *row,* ProbeBundle *ledger row,* InBundle *membership row, and displayed* hsame *row, and the three overlaps have a certified common triple overlap K_{012}. Then the three induced* RegSeqRat *handoff packages, and the three induced finite-window* $\mathsf{Real}^{\uparrow}$ *handoff packages, determine one StreamName source surface on K_{012}.*

The triple source surface consists only of the restricted StreamName finite-window rows on K_{012}. It excludes completion data, quotient data, host index equality, every RegSeqRat *regularity row, every* $\mathsf{Real}^{\uparrow}$ *seal classifier row, and every* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First restrict the three windows to the common triple overlap K_{012}. Each restricted pair is a certified finite overlap with the same displayed unary observations, classifier rows, bundle rows, membership rows, and hsame comparisons.

Apply Definition 126.88 to $(\mathcal{W}_0, \mathcal{W}_1)$ on K_{012} and again to $(\mathcal{W}_1, \mathcal{W}_2)$ on K_{012}. The first application identifies the source surface of $\mathcal{W}_0$ with that of $\mathcal{W}_1$; the second identifies the source surface of $\mathcal{W}_1$ with that of $\mathcal{W}_2$.

The displayed hsame rows compose through the common middle restriction, so the two identifications give a single source surface for all three windows on the triple overlap. The optional third pair $(\mathcal{W}_0, \mathcal{W}_2)$ reads the same restricted rows and therefore lands on that same surface.

Finally use the exclusion clause in Definition 126.88. The cocycle is a statement about the public StreamName finite-window source only; no completion, quotient, host-index, RegSeqRat regularity, RealUp seal, or RealUp completeness row is part of the source surface. □

Theorem 126.550 (StreamName RegSeqRat triple-overlap seal cocycle)**.** *Let $\mathcal{W}_0, \mathcal{W}_1, \mathcal{W}_2$ be certified finite* $\mathsf{StreamName}^{\uparrow}$ *windows with pairwise certified overlaps and a certified common triple overlap K_{012}. Suppose the source hypotheses of Definition 126.549 hold, and suppose the three induced* RegSeqRat *handoff packages consume one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger on every restriction to K_{012}. Then the* $\mathsf{Real}^{\uparrow}$ *seal classifier rows induced from the three handoffs agree on K_{012}.*

This agreement is only the finite-window seal row agreement produced from the shared StreamName source and shared DyadicRatCore ledger. It does not include completion data, quotient data, host index equality, or a $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First fix the source. By Definition 126.549, the three windows expose one StreamName source surface on K_{012} for their induced RegSeqRat and finite-window RealUp handoffs.

Next compare the first two seal rows. Apply Definition 126.89 to $(\mathcal{W}_0, \mathcal{W}_1)$ after restricting to K_{012}. The source rows are the common rows just fixed, and the DyadicRatCore finite ledger is shared by hypothesis, so the induced RealUp seal classifier rows agree pointwise there.

Compare the second and third seal rows in the same way. The pair $(\mathcal{W}_1, \mathcal{W}_2)$ has the same restricted source and the same consumed dyadic ledger, so pointwise seal stability identifies their induced classifier rows on K_{012}.

These two pointwise agreements have the same middle classifier row. Hence the three induced RealUp seal classifier rows form one row on the triple overlap. The construction never asks for

completion, quotient, host-index, or RealUp completeness data, because pointwise seal stability only reads the finite StreamName source and the shared dyadic ledger. □

Theorem 126.551 (StreamName RegSeqRat finite-cover pasted source exhaustion)**.** *Let* $\{\mathcal{W}_i\}_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose every pair has a certified finite overlap, all pairwise restrictions agree on the displayed StreamName source rows, every triple overlap satisfies the source and seal cocycle hypotheses above, and all induced* RegSeqRat *handoff packages consume the same DyadicRatCore finite ledger on their displayed overlaps. Then the family pastes to one* RegSeqRat *and finite-window* $\mathsf{Real}^{\uparrow}$ *source surface over the finite union of the windows.*

The pasted surface is exhausted by the public StreamName finite-window rows, the displayed hsame *comparisons, and the shared DyadicRatCore ledger rows. It has no completion data, quotient data, host index equality, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Proceed by finite ProbeBundle induction over the cover inventory. The empty inventory has no exposed source row, and the singleton inventory is the certified source surface of its one StreamName window.

For the induction step, assume a finite subcover has already pasted to one RegSeqRat and finite-window RealUp source surface. Let $\mathcal{W}_j$ be the next certified window. On each overlap between $\mathcal{W}_j$ and a member of the subcover, apply Definition 126.88 to obtain the same source rows, and apply Definition 126.89 to stabilize the induced RealUp seal classifier rows over the shared dyadic ledger.

Now paste the new window to the existing subcover surface. Use Definition 126.90 on each displayed overlap, and use Definition 126.91 to see that the order of adjoining the finitely many overlap rows does not change the pasted classifier surface.

The triple-overlap source and seal cocycles supply the compatibility needed when two old windows and the new window meet. They identify the common source and seal rows on each triple intersection, so no additional choice of a source or classifier row is introduced by the induction step.

Finally exhaust the pasted finite union. Every row inserted during the induction is either a public StreamName finite-window row, a displayed hsame comparison, or a shared DyadicRatCore ledger row used by the RegSeqRat handoff. Completion data, quotient data, host index equality, and RealUp completeness rows have no carrier coordinate in this finite pasted source. □

Theorem 126.552 (StreamName RegSeqRat finite-cover pasted seal exhaustion)**.** *Let* $\{\mathcal{W}_i\}_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows satisfying the source hypotheses of Definition 126.551. Assume moreover that every triple overlap satisfies Definition 126.550, and that all displayed overlaps consume one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger in the induced* RegSeqRat *handoff packages. Then the induced finite-window* $\mathsf{Real}^{\uparrow}$ *seal classifier rows paste to one seal surface over the finite union of the windows.*

The pasted seal surface is exhausted by the finite-window RealUp classifier rows forced by the common StreamName source and the shared DyadicRatCore ledger. It excludes completion data, quotient data, host index equality, selected limits, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First fix the source surface. Apply Definition 126.551 to the family of windows. This gives one RegSeqRat and finite-window RealUp source surface over the finite union, with no completion, quotient, host-index, or completeness coordinate.

We now attach seal rows by finite ProbeBundle induction over the cover inventory. The empty inventory has no classifier row to paste, and the singleton inventory carries the seal row induced from its certified StreamName window and shared DyadicRatCore ledger.

For the induction step, assume a finite subcover already carries one pasted seal surface. Let $\mathcal{W}_j$ be the next certified window. On each displayed overlap with the existing subcover, apply Definition 126.89; the source row is the fixed source surface, and the dyadic ledger is the shared ledger from the hypotheses.

Use Definition 126.90 to attach the seal row of $\mathcal{W}_j$ along those overlap rows. When two old windows and $\mathcal{W}_j$ meet, the triple-overlap seal cocycle identifies the three restricted classifier rows, so the attachment does not depend on which overlap is read first.

Finally apply Definition 126.91. The associativity of pasted seal rows, together with the triple-overlap seal cocycle, makes the finite ProbeBundle induction order-independent. Thus all induced finite-window RealUp seal classifier rows determine one seal surface over the finite union.

Every coordinate used in the construction is a finite-window classifier row obtained from the common StreamName source and the shared DyadicRatCore ledger. Completion data, quotient data, host index equality, selected limits, and RealUp completeness are not introduced by pointwise seal stability, pointwise seal pasting, or pasted seal associativity. □

Theorem 126.553 (StreamName RegSeqRat finite-cover pasted seal terminality)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows satisfying the hypotheses of Definition 126.551. Assume every triple overlap satisfies Definition 126.550, and assume all displayed overlaps consume one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger in the induced* RegSeqRat *handoff packages. If $\mathcal{S}$ and $\mathcal{S}'$ are two induced finite-window* $\mathsf{Real}^{\uparrow}$ *pasted seal surfaces over the finite union with the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows,* RegSeqRat *handoff rows, overlap* hsame *data, and displayed* BHist, Cont, Pkg, *and* NameCert *support rows, then $\mathcal{S} = \mathcal{S}'$ as pasted seal surfaces over the finite union.*

This terminality is a finite L10 seal statement. It contains no completion data, quotient data, host index equality, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First fix the source side. By Definition 126.551, the displayed StreamName rows, overlap hsame data, and shared DyadicRatCore ledger rows determine one RegSeqRat and finite-window RealUp source surface over the finite union. Thus $\mathcal{S}$ and $\mathcal{S}'$ are compared over the same finite source surface.

Next apply Definition 126.552 to both candidate pasted seal surfaces. Each candidate is exhausted by the same finite-window RealUp classifier rows forced by that source surface and the same shared DyadicRatCore ledger. The displayed RegSeqRat handoff rows and BHist, Cont, Pkg, and NameCert support rows therefore leave no separate seal coordinate that could distinguish the two candidates.

It remains to identify the overlap choices used in the finite pasting. On each displayed pairwise overlap, Definition 126.89 gives the same pointwise seal row for both candidates, because the source row and dyadic ledger row are the same. The triple-overlap compatibility is exactly Definition 126.550, so the pointwise overlap rows agree on every common triple restriction.

Finally use Definition 126.90 and Definition 126.91. Pointwise pasting attaches the same local seal rows along the same overlap data, and associativity makes the finite attachment order irrelevant. Hence both $\mathcal{S}$ and $\mathcal{S}'$ are the same pasted finite-window RealUp seal surface over the finite union.

The argument only reads finite StreamName rows, DyadicRatCore ledger rows, RegSeqRat handoff rows, overlap hsame data, and displayed support rows. Completion data, quotient data, host index equality, selected limits, countable choice, and RealUp completeness are not carrier coordinates in this terminal pasted seal surface. □

126.40 StreamName finite-request section packet

Theorem 126.554 (StreamName Real endpoint finite-request section)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *window request that is displayed to a* $\mathsf{Real}^{\uparrow}$ *endpoint row factors through a section packet whose visible rows are exactly the endpoint rows, the certified finite-window observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those observations, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget rows used by the read, the componentwise* hsame *transport, the* Cont *replay route,* Pkg *provenance, and the local* NameCert *rows. The section has no coordinate for quotient stream equality, host sequence equality, selected limits, host real equality, or ambient completion.*

Proof. Start with the finite observation packet of Definition 126.527. It projects a displayed window section to the membership, unary-history, rational-history classifier, hsame, Cont, Pkg, and local NameCert rows.

Next apply the Real exit handoff of Definition 126.529. The endpoint consumer reads that same section through $\mathsf{StreamName}^{\uparrow}$ before the downstream $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces are consumed.

Finally use Definition 126.535. It adds exactly the RegSeqRat readback face, the DyadicRatCore budget face, and the RealUp seal face to the StreamName section. Since the cited packets contain no quotient stream, host sequence, selected-limit, host-real, or ambient completion coordinate, none can appear in the endpoint request. □

Theorem 126.555 (StreamName RegSeqRat window-budget section)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *window request consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then by a* $\mathsf{Real}^{\uparrow}$ *endpoint seal is budgeted by one displayed section packet. The packet displays the finite endpoint rows, RegSeqRat readback rows, DyadicRatCore budget rows, StreamName window rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows, and every downstream read is classified by those rows. It exports no quotient stream equality, no host sequence equality, no selected limit, no second dyadic source, and no ambient completion datum.*

Proof. First read the request through Definition 126.542. The finite window is determined by its certified unary observation, bundle membership, tail route, window budget, and pointwise RegSeqRat handoff.

Next apply Definition 126.543. It forces every accepted downstream RegSeqRat or RealUp read to factor through that request together with the displayed hsame, Cont, Pkg, and NameCert support rows.

Now use Definition 126.537. The accepted route is exactly the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the same support rows, so the endpoint, readback, and budget data are one section packet rather than separate routes.

The non-escape clauses in Definition 126.545 and Definition 126.536 remove quotient streams, host sequence equality, selected limits, second dyadic sources, and ambient completion data from the packet. □

Theorem 126.556 (StreamName finite-request section budget exactness)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *request consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then by a public* $\mathsf{Real}^{\uparrow}$ *seal uses one displayed budget section. Its rows are precisely the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius budget rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *rows. Hence two Real-facing presentations over the same finite request cannot be distinguished by a second dyadic budget, a second regular-sequence source, quotient stream equality, a selected limit, host real equality, or any ambient completion coordinate.*

Proof. First apply Definition 126.554. This reads the Real-facing endpoint as a finite $\mathsf{StreamName}^{\uparrow}$ section whose visible rows already include the endpoint, readback, budget, support, package, and naming coordinates.

Next apply Definition 126.555 to the same request. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ budget rows are not parallel witnesses; they are the readback and budget faces of the same displayed section packet.

Now compose with Definition 126.537. The composed route identifies the four public faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the same componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows. Thus the Real-facing presentation is determined by that single finite budget section.

Finally close the boundary with Definition 126.545 and Definition 126.536. These non-escape clauses remove second budget sources, second regular-sequence sources, quotient stream equality, selected limits, host real equality, and ambient completion coordinates from the observation packet, so none can separate two presentations over the same finite request. □

Theorem 126.557 (StreamName Real seal request section nonescape)**.** *Every finite StreamName request that reaches a* $\mathsf{Real}^{\uparrow}$ *seal through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *is exhausted by the finite-request section packet of Definition 126.554 and Definition 126.555. Thus the Real seal can replay only endpoint rows, RegSeqRat readback rows, DyadicRatCore budget rows, StreamName window rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It cannot obtain quotient equality of streams, quotient equality of Cauchy presentations, selected limits, host real equality, metric-completeness data, or ambient completion.*

Proof. Begin with Definition 126.554. It fixes the Real endpoint read as a finite StreamName section together with the RegSeqRat, DyadicRatCore, RealUp, and BEDC support rows.

Then apply Definition 126.555. The RegSeqRat readback and DyadicRatCore budget rows are the same section packet used by the endpoint seal, so the Real seal does not receive an independent readback source or a second budget source.

Finally compose with Definition 126.536. The four-face budget boundary contains no coordinate for quotient stream equality, quotient Cauchy-presentation equality, selected limits, host real equality, metric-completeness data, or ambient completion. The finite-request section therefore reaches the Real seal only through the displayed rows. □

Theorem 126.558 (StreamName finite-request section terminal coherence)**.** *Let Q and Q' be accepted finite request sections for the current* $\mathsf{StreamName}^{\uparrow}$ *Real-and-Completion route. Suppose that Q and Q' expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same displayed* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. Then Q and Q' determine the same terminal four-face route through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Conversely, every current terminal four-face route restricts to a finite request section with exactly those StreamName window, DyadicRatCore budget, RegSeqRat readback, RealUp seal, and displayed support rows. The coherence claim includes no quotient stream equality, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Proof. First read each accepted section through Definition 126.556. The budget row and the RegSeqRat readback row are not separate witnesses; they are faces of the same finite request section under the displayed support rows.

Next apply Definition 126.557. The Real seal cannot add quotient stream equality, Cauchy-presentation equality, selected limits, host equality, metric-completeness data, or ambient completion to the section.

Use Definition 126.587 for the two sections. Since their StreamName window, DyadicRatCore budget, RegSeqRat readback, RealUp seal, and support rows agree, the carrier and classifier verdicts at the terminal route agree.

Finally apply Definition 126.592. Its forward direction identifies same-section reads with the terminal four-face route, and its converse restricts every current terminal route back to such a finite request section. The listed exclusions are exactly the coordinates absent from that terminal completeness surface. □

Theorem 126.559 (StreamName finite-request terminal coherence)**.** *Any finite* $\mathsf{StreamName}^{\uparrow}$ *request that is read both as a* $\mathsf{Real}^{\uparrow}$ *endpoint finite-request section and as a* $\mathsf{RegSeqRat}^{\uparrow}$ *window-budget section determines a unique terminal public row. That row is the four-face Real-and-Completion packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with only the displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The same packet is therefore the terminal exit route for the request, and it contains no quotient stream equality, no quotient equality of Cauchy presentations, no host sequence equality, no selected limit, no host real equality, no ambient or metric-completeness datum, no countable-choice row, no Classical row, and no standard bridge row.

Proof. First apply Definition 126.554. The endpoint reading fixes the finite request as a $\mathsf{Real}^{\uparrow}$ section with the StreamName window, RegSeqRat readback, DyadicRatCore budget, RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 126.555. The RegSeqRat window-budget reading uses the same finite section packet, so the readback and budget faces are not second routes parallel to the endpoint section.

Now compose with Definition 126.557. It removes every Real-seal escape coordinate outside the displayed section rows. Invoke Definition 126.500 to identify those rows with the four-object Real-and-Completion exit packet.

Finally use Definition 126.449. Once the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and displayed support rows agree, there is only one terminal finite BEDC route for the request. The exclusions are exactly the non-escape exclusions of Definition 126.536, so no quotient, selected-limit, ambient-completeness, choice, Classical, or bridge coordinate can replace the terminal four-face packet. □

Theorem 126.560 (StreamName finite-request section terminal reflection)**.** *Let Q be an accepted finite* $\mathsf{StreamName}^{\uparrow}$ *request section for the current L10 Phase Real and Completion route. Suppose the visible rows of Q display one* $\mathsf{StreamName}^{\uparrow}$ *finite window, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, one* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *support rows. Then Q is the public reflection surface for the terminal four-face L10 route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Every accepted finite request section with the same displayed rows reflects to that same terminal route, and every terminal four-face L10 route pulls back to that finite request section. Thus the

finite section is both the only public exit route available to the current L10 packet and the only pullback surface available to later $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and completion-facing consumers. It exports no quotient stream equality, selected limit, host real equality, ambient completion, metric-completeness datum, countable-choice row, Classical row, or standard bridge row.*

Proof. First apply Definition 126.556. The accepted section has exactly one displayed StreamName window, dyadic budget, regular readback, Real seal, and BEDC support packet, so no later consumer can replace the section by a second dyadic source or a second regular-sequence source.

Next apply Definition 126.557. The Real seal reads only the finite request section just described. Hence quotient stream equality, selected limits, host real equality, ambient completion, and metric-completeness data are not coordinates of the route seen by the seal.

Now compare any other accepted finite request section whose visible rows agree with those of Q. By Definition 126.558, the two sections determine the same terminal four-face packet. By Definition 126.587, the public carrier and classifier verdicts are therefore the same displayed section verdicts, not a quotient or host-equality verdict.

For the reverse direction, take a terminal four-face L10 route. The converse clause of Definition 126.558 restricts it to a finite request section with exactly the StreamName window, DyadicRatCore budget, RegSeqRat readback, RealUp seal, and displayed support rows. Applying classifier determinacy again identifies that pullback with Q whenever those visible rows agree.

Therefore the same finite section is the public exit surface and the consumer-facing pullback surface. Since the construction used only the cited budget, non-escape, terminal-coherence, and classifier-determinacy rows, no quotient stream equality, selected limit, host real equality, ambient completion, metric-completeness datum, countable-choice row, Classical row, or standard bridge row can be imported into the reflected route. □

Theorem 126.561 (StreamName finite-request common-refinement exactness)**.** *Let Q and Q' be accepted finite* $\mathsf{StreamName}^{\uparrow}$ *requests. Suppose both are read by the* $\mathsf{Real}^{\uparrow}$ *endpoint section and by the* $\mathsf{RegSeqRat}^{\uparrow}$ *window-budget section. Suppose further that they agree on a displayed common refinement of the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius budget rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then Q and Q' factor through the same terminal four-face refinement packet

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Conversely, every such displayed four-face refinement induces the same terminal public row for the two accepted requests. This exactness statement contains no quotient stream equality, no selected limit, no host real equality, no ambient or metric-completeness datum, no countable-choice row, no Classical row, no aggregate formalstatus *row, and no standard bridge row.*

Proof. First apply Definition 126.556 to Q and Q'. The $\mathsf{StreamName}^{\uparrow}$ finite window, DyadicRatCore endpoint and radius budget, RegSeqRat regularity and readback, RealUp seal, and displayed support rows are the section-budget rows themselves.

Next use Definition 126.557. The Real endpoint reading cannot add quotient stream equality, selected limits, host real equality, ambient completion, metric-completeness data, countable choice, Classical data, an aggregate status row, or a standard bridge row to the common refinement.

Now compare the sibling faces. DyadicRatCore exactness is supplied by Definition 291.66 and Definition 291.28; RegSeqRat exactness is supplied by Definition 301.69; and the RealUp face is

supplied by Definition 38.457. These results put the dyadic budget, regularity readback, and Real seal on the same displayed finite four-face packet.

Finally apply Definition 126.559 to the two refined requests. Since the four faces and the five support rows agree, the terminal public row is the same. Conversely, reading any displayed four-face refinement through the same cited exactness results recovers exactly those section rows, so terminal coherence sends the refinement back to the same public row. □

Theorem 126.562 (StreamName finite-request cover gluing)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of* $\mathsf{StreamName}^{\uparrow}$ *requests. Suppose each $\mathcal{R}_i$ reads both as a* $\mathsf{Real}^{\uparrow}$ *endpoint finite-request section and as a* $\mathsf{RegSeqRat}^{\uparrow}$ *window-budget section. Suppose also that every displayed pairwise overlap agrees on the rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Then the finite cover glues to one terminal four-face packet

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

over the finite union of the requests, together with only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The glued packet contains no quotient stream equality, no quotient equality of Cauchy presentations, no host sequence equality, no selected limit, no host real equality, no ambient or metric-completeness datum, no countable-choice row, no Classical row, and no standard bridge row.*

Proof. For each member of the finite family, apply Definition 126.554 to read it as a Real endpoint finite-request section. Apply Definition 126.555 to the same member to read it as the RegSeqRat window-budget section using the same StreamName, DyadicRatCore, RegSeqRat, RealUp, and displayed support rows.

On a displayed pairwise overlap, the hypotheses identify the BHist, hsame, Cont, Pkg, NameCert, DyadicRatCore, RegSeqRat, and RealUp rows seen from the two requests. Apply Definition 126.559 to each side of the overlap. The two readings therefore land on the same terminal public row, namely the four-face packet

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Now pass from pairwise agreement to finite-cover compatibility. On every triple overlap, use Definition 126.549 to identify the shared StreamName source surface, and use Definition 38.191 for the corresponding RealUp finite-window consumer surface. Thus the pairwise terminal rows satisfy the cocycle condition needed to paste a finite cover.

Finally paste the finite family by finite induction over the request inventory. The singleton case is the terminal packet just constructed. In the induction step, glue the next request along its displayed overlaps; the pairwise terminal coherence gives the common boundary row, and the triple-overlap cocycles make this boundary independent of the order in which overlaps are consumed.

The exclusion surface is inherited from Definition 126.557 and from the non-escape clause of Definition 126.559. Since every gluing step reads only displayed finite rows, no quotient stream equality, quotient Cauchy-presentation equality, host sequence equality, selected limit, host real equality, ambient or metric-completeness datum, countable-choice row, Classical row, or standard bridge row is introduced by the finite cover. □

Theorem 126.563 (StreamName finite-request subcover terminal invariance)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of* $\mathsf{StreamName}^{\uparrow}$ *requests satisfying the hypotheses of Definition 126.562. Let $J \subseteq I$ be a finite subfamily such that every request in $I \setminus J$ has the same terminal four-face packet as the retained subfamily on every displayed overlap. Then deleting the requests indexed by $I \setminus J$ leaves the same terminal packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

over the retained finite request cover, with the same displayed support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The retained terminal packet still contains no quotient stream equality, no quotient equality of Cauchy presentations, no host sequence equality, no selected limit, no host real equality, no ambient or metric-completeness datum, no countable-choice row, no Classical row, and no standard bridge row.

Proof. Apply Definition 126.562 to the full finite family. This produces one terminal four-face packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

On each displayed overlap between a discarded request and the retained subfamily, apply Definition 126.559. The overlap has the same StreamName, DyadicRatCore, RegSeqRat, RealUp, and displayed support rows on both sides, so the discarded request and the retained cover land on the same terminal public row.

It remains to see that the order of deletion is irrelevant. For any three requests meeting on a displayed overlap, use Definition 126.549 for the shared StreamName source surface and Definition 38.191 for the RealUp finite-window consumer surface. These cocycles are the same compatibility data used in Definition 126.562, so removing discarded requests in any finite order preserves the boundary row already seen by the retained subfamily.

The remaining finite cover is therefore pasted with the same terminal packet as the full cover. Its exclusion surface is inherited from the non-escape clause of Definition 126.559 and from Definition 126.562: every step reads only the displayed finite BEDC rows. Thus no quotient stream equality, quotient Cauchy-presentation equality, host sequence equality, selected limit, host real equality, ambient or metric-completeness datum, countable-choice row, Classical row, or standard bridge row appears after the subcover is retained. □

Theorem 126.564 (StreamName finite-request Real seal cocycle determinacy)**.** *Let $\mathcal{R}_0, \mathcal{R}_1, \mathcal{R}_2$ be three finite* $\mathsf{StreamName}^{\uparrow}$ *requests satisfying the hypotheses of Definition 126.562. Suppose that every pairwise overlap among the three requests exposes the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then the Real-facing seal row determined on the triple overlap is independent of the order in which the pairwise overlaps are consumed. Equivalently, the two finite overlap-consumption paths*

$$(\mathcal{R}_0 \cap \mathcal{R}_1) \cap \mathcal{R}_2 \quad \textit{and} \quad \mathcal{R}_0 \cap (\mathcal{R}_1 \cap \mathcal{R}_2)$$

repack the same terminal Real seal row through the finite BEDC support rows. This determinacy asserts only the displayed finite-request surface: it adds no quotient stream equality, no host sequence equality, no selected limit, no ambient or metric-completeness datum, no countable-choice row, no Classical row, and no standard bridge row.

Proof. Start with the three pairwise overlaps. On each pair, apply Definition 126.559. The shared StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows force both requests in the pair to land on the same terminal four-face packet.

The possible ambiguity is only the order in which the three pairwise overlaps are consumed. Use Definition 126.549 to identify the StreamName source surface on the triple overlap, and use Definition 38.191 to identify the corresponding RealUp finite-window consumer surface. These two cocycles make the left-associated and right-associated overlap readings carry the same source row and the same Real-facing consumer row.

Finally repack that common terminal row through the displayed finite BEDC support data. Since the repacking reads only BHist, hsame, Cont, Pkg, and NameCert, it cannot introduce quotient stream equality, host sequence equality, selected limits, ambient or metric-completeness data, countable choice, Classical rows, or standard bridge rows. The Real-facing seal row on the triple overlap is therefore determined independently of the overlap-consumption order. □

Theorem 126.565 (StreamName finite-request cover CauchyLimitSeal stability)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of* $\mathsf{StreamName}^{\uparrow}$ *requests satisfying the hypotheses of Definition 126.562. Suppose also that each $\mathcal{R}_i$ is consumed by the terminal* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route of Definition 386.118. Then the cover determines the same glued four-face packet through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{CauchyLimitSeal}^{\uparrow}.$$

The CauchyLimitSeal consumer reads the glued StreamName schedule, RegSeqRat source, DyadicRatCore ledger, RealUp seal, and displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows without changing the glued packet. It adds no quotient stream equality, selected host limit, host real equality, hidden tail oracle, ambient completeness, metric-completeness datum, countable-choice row, Classical row, or standard bridge row.*

Proof. First apply Definition 126.562. The finite family has one glued four-face packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For each member of the finite family, apply Definition 386.118. Its terminal seal consumer must pass through the same scheduled $\mathsf{StreamName}^{\uparrow}$ row, then the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row. Thus the consumer cannot select a second schedule or replace the finite cover by an external endpoint.

On overlaps, apply Definition 386.49. The seal route first factors through the finite-window meet surface before reading the seal packet, so the pairwise and triple-overlap compatibilities already used by the glued cover are the boundary seen by the CauchyLimitSeal consumer.

Therefore the CauchyLimitSeal read is a terminal consumer of the glued packet, not an additional gluing coordinate. The non-escape clauses in the cited StreamName and CauchyLimitSeal rows rule out quotient stream equality, selected host limits, host real equality, hidden tail oracles, ambient completeness, metric-completeness data, countable-choice rows, Classical rows, and standard bridge rows. □

Theorem 126.566 (StreamName finite-request section status pullback)**.** *Let $\mathcal{R}$ be an accepted finite-request* $\mathsf{StreamName}^{\uparrow}$ *section that is consumed by both the terminal* $\mathsf{Real}^{\uparrow}$ *route and the terminal* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route. Then the Phase Real and Completion status read pulls back to the same four local records*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Each face keeps its own closurestatus row and its own formalstatus row, as read through the two-axis discipline of section 9467.2. The pullback exports no quotient stream equality, selected limit, host real equality, ambient completeness, metric-completeness datum, countable-choice row, Classical row, aggregate formalstatus row, or standard bridge row.

Proof. First apply Definition 126.556. The accepted section has exactly the displayed finite support over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so every downstream consumer must read the same section budget and cannot add another face to the packet.

Next apply Definition 126.557 and Definition 126.558. The terminal $\mathsf{Real}^{\uparrow}$ route therefore consumes the accepted StreamName section through the same scheduled row, dyadic ledger, regular source, and Real seal, with no escape to a selected host limit or host real equality.

Now apply Definition 126.565. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reads the same finite-request surface as a terminal seal route. Its seal read is a consumer of the four-face packet, not a new completion coordinate or a replacement for the Real-facing row.

Finally read both consumers through section 9467.2. That discipline keeps the paper closure axis, Lean verification axis, and bridge status attached to each face separately. Hence the Phase Real and Completion status read pulls back only to the four local records named above, and it cannot export quotient stream equality, selected limits, host real equality, ambient or metric-completeness data, countable-choice rows, Classical rows, aggregate formalstatus rows, or standard bridge rows. □

Theorem 126.567 (StreamName finite-request phase-exit sufficiency)**.** *An accepted finite-request* $\mathsf{StreamName}^{\uparrow}$ *section that exposes exactly the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the shared BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows is sufficient for the current Phase Real and Completion exit read. Conversely, every current Phase Real and Completion exit read pulls back to such an accepted finite-request* $\mathsf{StreamName}^{\uparrow}$ *section.*

The sufficiency and pullback contain only the displayed section packet and the four local face records. They do not assert quotient stream equality, selected limits, host real equality, ambient completeness, metric-completeness, countable choice, an aggregate formalstatus row, or standard bridge rows.

Proof. For sufficiency, apply Definition 126.556. The accepted section has exactly the four named faces and the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows, so the section supplies the finite budget required by the StreamName side of the exit read.

Next apply Definition 126.558. The terminal consumers of that section must pass through the same scheduled StreamName row, dyadic ledger, regular-rational source, and Real-facing seal. Apply Definition 126.592 to identify that terminal surface with the four-face packet visible to the roadmap criterion.

Finally apply Definition 38.733. Its certificate accepts precisely the four-object surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the displayed BEDC support rows. Hence the accepted finite-request section is sufficient for the current Phase Real and Completion exit read.

For the converse, start from any current Phase Real and Completion exit read and apply Definition 38.733. The read is a four-object certificate over the same dyadic, StreamName, RegSeqRat, and Real faces, with only the shared BEDC support rows.

Now apply Definition 126.566. The roadmap status read pulls back to the accepted finite-request $\mathsf{StreamName}^{\uparrow}$ section carrying those four local records. The pullback keeps the closurestatus and formalstatus rows componentwise rather than forming an aggregate row.

The exclusion clauses come from the same budget exactness, terminal coherence, status pullback, four-face completeness, and Real exit-certificate results. Together they leave no coordinate for quotient stream equality, selected limits, host real equality, ambient or metric-completeness, countable choice, aggregate formalstatus, or standard bridge rows. □

126.41 StreamName finite-request RealUp seal lock

Theorem 126.568 (StreamName finite request section RealUp seal lock)**.** *Any finite request section packet accepted by* $\mathsf{StreamName}^{\uparrow}$ *and consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity ledger, componentwise* hsame *transport, the displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *rows. The seal lock exports no quotient stream, no host real equality, no ambient completion point, and no* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Start with the finite request section packet of Definition 126.554 and Definition 126.555. These results make the accepted request a finite StreamName section whose public rows are the schedule, endpoint, readback, budget, transport, continuation, package, and local naming rows.

Next use the RegSeqRat-to-RealUp handoff rows of Definition 301.22, Definition 301.52, and Definition 301.141. They place the Real seal behind the same regularity ledger and finite request rows rather than behind a host sequence equality or selected completed limit.

Then compose with the DyadicRatCore endpoint and radius ledger reads recorded by Definition 301.5, Definition 301.13, and Definition 301.69. The endpoint and radius rows are therefore the displayed dyadic rows carried by the finite section, not independent ambient metric data.

Finally apply the non-escape exclusions of Definition 126.557 and Definition 126.536. Since the composed route has coordinates only for the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity ledger, hsame, Cont, Pkg, and $\mathsf{NameCert}$, it cannot export quotient streams, host real equality, ambient completion points, or RealUp completeness data. □

Theorem 126.569 (StreamName finite request Real seal totality)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *request consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal is exhausted by the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. There is no additional request coordinate outside these rows.*

Proof. Apply Definition 126.568. It already factors the accepted finite request section through the StreamName schedule rows, the DyadicRatCore endpoint-radius ledger, the RegSeqRat regularity ledger, componentwise hsame transport, Cont, Pkg, and local $\mathsf{NameCert}$.

It remains only to name the terminal seal face. The same theorem places the accepted request behind the $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ seal, while Definition 38.713 identifies that seal read with the four displayed object faces and the same support rows. Therefore the finite request has no coordinate beyond the schedule, endpoint, radius, regularity, readback, seal, and support rows listed in the statement. □

Theorem 126.570 (StreamName request section non-escape boundary)**.** *The finite request section of Definition 126.568 does not read a quotient stream, host sequence equality, selected limit, host*

real equality, ambient completion point, countable-choice row, metric-completeness row, or $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. First apply Definition 126.569. The visible packet is exhausted by the finite StreamName schedule, DyadicRatCore endpoint-radius rows, RegSeqRat regularity and readback rows, the RealUp seal row, and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Now use Definition 126.557. That boundary excludes quotient streams, host equalities, selected limits, ambient completion points, and RealUp completeness data from the request section. The totality packet has no row for countable choice or metric completeness either, so those rows cannot be read from the section. □

Theorem 126.571 (StreamName open-phase request handoff)**.** *The finite request section is precisely the local handoff from* $\mathsf{StreamName}^{\uparrow}$ *through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$*: its terminal packet is the four-face route with* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with Definition 126.568. It places the finite request inside the local $\mathsf{StreamName}^{\uparrow}$ section consumed by the $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ seal. Then use Definition 38.713 to read that seal as a terminal four-face packet.

Next apply Definition 126.500. The four faces are exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the only support rows are BHist, hsame, Cont, Pkg, and NameCert.

Finally apply Definition 126.557. It prevents the handoff from carrying quotient streams, selected limits, host real equality, ambient completion points, countable choice, metric completeness, or a RealUp completeness datum. Hence the section is exactly the local open-phase handoff through the displayed four faces and support rows. □

Theorem 126.572 (StreamName finite request Real seal classifier exactness)**.** *For a finite request section consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal, acceptance by the seal-facing classifier is necessary and sufficient for factorization through exactly the visible* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The classifier accepts no quotient stream equality, host real equality, selected limit, choice datum, ambient completion, metric-completeness row, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. For necessity, start from an accepted seal-facing request. Apply Definition 126.568 to factor the request through the finite StreamName section consumed by the Real seal, and then apply Definition 126.569. The accepted coordinates are exhausted by the schedule, endpoint, radius, regularity, readback, seal, and displayed support rows.

For sufficiency, suppose a finite section already factors through exactly those rows. Use Definition 126.571 to read the packet as the local handoff from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. Then Definition 126.500 identifies the same packet as the four-object Real-completion exit route, so the seal-facing classifier accepts it.

Finally apply Definition 126.570. The non-escape boundary rules out quotient stream equality, host real equality, selected limits, choice data, ambient completion, metric-completeness rows, and any second $\mathsf{Real}^{\uparrow}$ seal coordinate. Thus the displayed four faces and support rows are not merely sufficient witnesses for acceptance; they are the exact classifier surface. □

Theorem 126.573 (StreamName finite request Real seal common-refinement lock)**.** *Let* $\mathcal{R}$ *and* $\mathcal{S}$ *be two finite request sections consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal and accepted by the*

seal-facing classifier. Suppose that their displayed rows agree on the $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, the componentwise* hsame *rows, the displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows. Then* $\mathcal{R}$ *and* $\mathcal{S}$ *determine the same finite seal-facing packet. No quotient stream, selected limit, choice datum, ambient completion, metric-completeness row, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate can separate the two accepted requests.*

Proof. Apply Definition 126.572 to $\mathcal{R}$ and to $\mathcal{S}$. Since both requests are accepted, each factors through exactly the displayed StreamName schedule rows, DyadicRatCore endpoint-radius rows, RegSeqRat regularity and readback rows, the RealUp seal row, and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Next use Definition 126.571 for each section. Both presentations are therefore the same four-object route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ once the displayed rows in the hypothesis are identified.

Apply Definition 126.559 to the two presentations, and then apply Definition 126.562 to the finite two-member cover $\{\mathcal{R}, \mathcal{S}\}$. The common displayed rows are precisely the overlap data required by the gluing theorem, so the two accepted sections collapse to one terminal seal-facing packet.

Finally apply Definition 126.570. The common packet has no row for quotient streams, selected limits, choice data, ambient completion, metric completeness, or a second RealUp seal coordinate. Hence no non-displayed coordinate can distinguish the two accepted seal-facing requests. □

126.42 StreamName RegSeqRat Real seal handoff

Theorem 126.574 (StreamName RegSeqRat Real seal handoff)**.** *Let* W *be a finite-window packet accepted by* $\mathsf{StreamName}^{\uparrow}$*. Suppose each point row of* W *is certified by* $\mathsf{DyadicRatCore}^{\uparrow}$*, adjacent windows satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity budget, and the displayed support rows are exactly the finite-window* BHist *rows, componentwise* hsame *transport,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row. Then* W *reaches the* $\mathsf{Real}^{\uparrow}$ *seal boundary through the existing finite-window route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise handoff, and the Real completion sibling route. The handoff exports no quotient stream equality, selected limit, host real equality, or ambient completeness datum.*

Proof. First read the finite-window packet through the hub routes. The finite-window budget and finite-request section rows expose the StreamName schedule, point rows, transport rows, continuation routes, package provenance, and local naming surface before any real-seal consumer is reached.

Next apply the RegSeqRat pointwise handoff. The DyadicRatCore-certified point rows and the adjacent-window regularity budget place the packet inside the displayed $\mathsf{RegSeqRat}^{\uparrow}$ regularity ledger rather than inside quotient stream equality or a selected host sequence.

Then compose with the Real completion sibling route. That route consumes the finite StreamName window and the RegSeqRat regularity ledger as the source face of the $\mathsf{Real}^{\uparrow}$ seal boundary.

The non-escape statement follows from the visible coordinate list. Since the packet contains only finite-window rows, DyadicRatCore point certificates, RegSeqRat regularity rows, the RealUp seal face, hsame, Cont, Pkg, and NameCert, it has no coordinate for quotient stream equality, selected limits, host real equality, or ambient completeness. □

Theorem 126.575 (StreamName RegSeqRat Real tail-section pullback)**.** *Let U and V be two finite tail-section requests of the same accepted* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose the point rows displayed by U and V are* $\mathsf{DyadicRatCore}^{\uparrow}$*-certified, the adjacent windows in both requests satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *budget, and each request reaches the* $\mathsf{Real}^{\uparrow}$ *seal boundary through the handoff of Definition 126.574. Then U and V determine the same seal-visible source exactly when their visible rows admit a common finite pullback over the displayed StreamName window, the DyadicRatCore endpoint and radius ledger, the RegSeqRat regularity and classifier rows, componentwise* hsame *transports,* Cont *replay rows,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows. This pullback exports no quotient stream equality, selected limit, host real equality, or ambient completeness datum.*

Proof. For the forward direction, apply Definition 126.574 to each request. The seal-visible source of either request is read only through the finite StreamName window, the DyadicRatCore point certificates, the RegSeqRat budget rows, the transport and replay rows, provenance, and local naming.

Use the RegSeqRat common-window surface and the tail-selector rows to compare the two finite requests before the $\mathsf{Real}^{\uparrow}$ seal is read. Since the accepted StreamName packet is the same, the overlap is a pullback of displayed rows rather than a comparison of host sequences.

For the reverse direction, start with the common finite pullback. Its projections recover the visible rows of U and V, including the endpoint and radius ledger, regularity/classifier data, hsame transports, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Composing each projection with the seal handoff gives the same $\mathsf{Real}^{\uparrow}$ seal-visible source. The coordinate inventory excludes quotient stream equality, selected limits, host real equality, and ambient completeness, so equality at the seal is exactly the displayed finite pullback condition. □

Theorem 126.576 (StreamName RegSeqRat Real seal correspondence)**.** *Let W be a finite-window packet accepted by* $\mathsf{StreamName}^{\uparrow}$*. Suppose each point row of W is certified by* $\mathsf{DyadicRatCore}^{\uparrow}$*, the adjacent-window rows satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *budget, and the displayed support rows are the finite-window* BHist *rows, componentwise* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. A seal-visible source section determined by W is the same as the seal-visible source section of the corresponding public finite* $\mathsf{RegSeqRat}^{\uparrow}$ *request exactly when both sections factor through the ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the same visible BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. This correspondence exports no quotient stream equality, host real equality, selected limit, choice datum, ambient completeness datum, metric-completeness row, or second seal coordinate.*

Proof. First apply Definition 126.574 to read the accepted $\mathsf{StreamName}^{\uparrow}$ window at the $\mathsf{Real}^{\uparrow}$ seal boundary. The handoff exposes only the finite StreamName window, the DyadicRatCore point certificates, the RegSeqRat adjacent-window budget, and the listed support rows.

Next apply Definition 301.139 to the corresponding public finite $\mathsf{RegSeqRat}^{\uparrow}$ request. Its source section displays the same schedule, dyadic ledger, classifier and regularity rows, componentwise hsame transports, Cont replay rows, Pkg provenance, and local $\mathsf{NameCert}$ surface.

For the forward direction, compose these two readings with Definition 301.141. A seal-visible source section coming from W therefore restricts to the same finite request source section precisely when the public request and the StreamName packet share the displayed rows of the four-face route.

For the reverse direction, start with a common ordered route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The finite request source section repacks along the StreamName handoff, and Definition 38.713 identifies the resulting packet as the same terminal-budget Real seal read.

The non-escape clause is the coordinate inventory of the cited routes. None of them contains a row for quotient stream equality, host real equality, selected limits, choice, ambient completeness, metric-completeness data, or a second seal coordinate, so the correspondence is exactly the equality of the displayed finite BEDC surface. □

126.43 StreamName finite-request section pullback

Theorem 126.577 (StreamName finite-request section pullback)**.** *Every finite request section that faces* $\mathsf{Real}^{\uparrow}$ *through the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal pulls back to the same displayed* $\mathsf{StreamName}^{\uparrow}$ *section schedule, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback surface, the same componentwise* hsame *rows, the same displayed* Cont *routes, the same* Pkg *provenance, and the same local* NameCert *rows. Thus a RealUp-facing request has no route around the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows.*

Proof. Start with the finite endpoint section of Definition 126.554. It reads the RealUp-facing request as a finite $\mathsf{StreamName}^{\uparrow}$ section whose visible coordinates are the endpoint, certified observation, support, and local naming rows.

Next apply Definition 126.555. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ budget rows consumed by the same request are not independent rows; they are the window-budget face of the same finite section.

Now use Definition 126.568 and Definition 126.569. The $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ seal is exhausted by the displayed StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity/readback surface, and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Finally compare with the Real-side route. Apply Definition 38.698 to repack the RealUp-facing request as the same finite packet, and apply Definition 38.713 to rule out a second terminal-budget face. The pullback is therefore the displayed StreamName, DyadicRatCore, RegSeqRat, and support-row section already named above. □

Theorem 126.578 (StreamName RegSeqRat Real finite-request section lock)**.** *Let* $\mathcal{R}$ *and* $\mathcal{S}$ *be two presentations of a* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal over the same finite request. The presentations agree exactly when their displayed section rows agree: the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback surface, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows. No further RealUp-facing coordinate can distinguish the two presentations.*

Proof. For the forward direction, read both presentations through Definition 126.577. Since the finite request is the same, the pullback exposes the same StreamName schedule, DyadicRatCore ledger, RegSeqRat surface, and BEDC support rows for both presentations.

For the reverse direction, suppose the displayed section rows agree. Apply Definition 126.571 to each presentation. Both seal presentations are then the same local handoff through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

It remains to see that the Real-facing seal cannot add a separating row. Use Definition 126.568 to place each presentation behind the same finite section, and use Definition 126.569 together with Definition 38.713 to exhaust the terminal seal surface by those rows. Hence agreement of the displayed section rows is both necessary and sufficient. □

Theorem 126.579 (StreamName finite-request section classifier pullback)**.** *Let Q and Q' be RealUp-facing finite requests on the terminal L10* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *route. Suppose that the two requests display the same* $\mathsf{StreamName}^{\uparrow}$ *section schedule, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback surface, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same support rows* BHist*, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. Then Q and Q' pull back to the same finite carrier verdict and the same classifier verdict on that terminal four-face route.*

The pullback uses only the displayed four faces and the named support rows. It exports no quotient stream equality, no host sequence equality, no selected limit, no ambient completeness row, no countable-choice row, and no bridge row.

Proof. First apply Definition 126.577 to both Q and Q'. The two RealUp-facing requests are read through the displayed StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity/readback surface, RealUp seal boundary, and support rows named in the hypotheses.

Next use Definition 126.578. Since those displayed rows agree, the two seal presentations have the same finite request section at the terminal route.

Now apply Definition 126.587. The common finite section fixes the carrier verdict and classifier verdict read by the $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ four-face route, so there is no second classifier route for either request.

Finally apply Definition 126.584. The common verdict is supported only by the displayed faces and support rows, and the excluded quotient stream equality, host sequence equality, selected limit, ambient completeness, countable-choice, and bridge rows are not part of the pullback. □

Theorem 126.580 (StreamName finite-request cofinal section)**.** *Let Q_0 and Q_1 be terminal L10 finite requests on the* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *route. Suppose the two requests have a common finite observation bound for their StreamName windows, DyadicRatCore endpoint-radius ledgers, RegSeqRat readbacks, Real seal boundaries, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. Then there is a common* $\mathsf{StreamName}^{\uparrow}$ *finite-request section S refining both Q_0 and Q_1. The section S carries the same DyadicRatCore endpoint-radius ledger, RegSeqRat readback, Real seal boundary, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows as its two terminal restrictions.*

The cofinal section is a finite readback object on the displayed L10 route. It does not introduce quotient streams, selected limits, countable choice, host sequence equality, ambient completion points, or a detached $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. First apply Definition 126.577 to the two terminal requests and the common finite observation bound. This places both requests inside one displayed StreamName section with the same dyadic, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.

Next use Definition 126.578. The lock keeps the RegSeqRat readback and Real seal boundary attached to the same finite section, so refining the StreamName schedule cannot add quotient stream equality, a selected limit, or an ambient completion point.

Now apply Definition 126.579. The classifier verdict of each terminal restriction is read from the same four-face section, hence the common section refines both requests without changing the DyadicRatCore endpoint-radius ledger or the RegSeqRat readback.

Finally use the Real terminal-budget exhaustion supplied by Definition 38.713. Since every terminal Real request has already spent its budget on the displayed finite faces and support rows, the common refinement is cofinal among such finite requests and has no coordinate for quotient streams, selected limits, countable choice, or Real completeness. □

Theorem 126.581 (StreamName finite-request section pullback exactness)**.** *Let Q and Q' be RealUp-facing finite requests on the terminal L10* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *route. The two requests determine the same finite carrier verdict and the same classifier verdict if and only if their displayed four-face pullbacks agree on the* $\mathsf{StreamName}^{\uparrow}$ *section schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback surface, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the support rows* BHist*, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

This equivalence is internal to the finite terminal packet. It excludes quotient streams, host sequence equality, selected limits, ambient completeness, countable choice, metric-completeness rows, and bridge rows.

Proof. For the forward direction, assume that Q and Q' determine the same finite carrier verdict and classifier verdict. Apply Definition 126.579 to read each verdict through the displayed terminal four-face route, and apply Definition 126.578 to lock the two seal presentations to the same finite request section.

The lock identifies exactly the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity/readback surface, RealUp seal boundary, and support rows named in the statement. Hence the displayed four-face pullbacks agree on every coordinate available to the finite verdict.

For the reverse direction, assume that the displayed four-face pullbacks agree on those rows. Use Definition 126.577 to identify both requests with the same finite section on the terminal L10 route. The classifier pullback theorem then reads that common section as the same finite carrier verdict and the same classifier verdict.

Finally use Definition 126.584. It rules out off-route separating coordinates: no quotient stream, host sequence equality, selected limit, ambient completeness, countable-choice row, metric-completeness row, or bridge row can distinguish the two requests once the displayed finite packet agrees. □

Theorem 126.582 (StreamName finite-request section kernel pullback)**.** *Every accepted finite-request section whose terminal face is consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *route pulls back to the same finite* BHist *window rows, componentwise* hsame *classifier rows, displayed* Cont *replay rows, and local* NameCert *rows exposed by its* $\mathsf{StreamName}^{\uparrow}$ *section. The pullback cannot introduce a host selector, quotient stream equality, selected limit, ambient completion coordinate, countable-choice row, metric-completeness row, or bridge row.*

Equivalently, the kernel-visible part of any such request is exactly the finite terminal packet named by the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity and readback surface, RealUp seal boundary, BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Proof. Apply Definition 126.581. It reduces kernel agreement for the terminal finite request to agreement of the displayed four-face pullback: the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity and readback surface, RealUp seal boundary, and the named support rows.

Next use Definition 126.583. It shows that accepted cofinal refinements preserve the finite carrier verdict and classifier verdict when those displayed rows are preserved, so the terminal consumer sees the same kernel window rows.

Finally apply Definition 126.584. The nonescape clause removes every off-route coordinate listed in the statement, leaving only the finite BHist, hsame, Cont, Pkg, and local NameCert rows already visible in the pullback. □

Theorem 126.583 (StreamName finite-request section cofinal verdict invariance)**.** *Let Q and Q' be RealUp-facing finite requests on the terminal L10 route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*. Suppose that Q' is obtained from Q by an accepted cofinal refinement preserving the displayed* $\mathsf{StreamName}^{\uparrow}$ *section schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback surface, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the support rows* BHist*, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. Then Q and Q' determine the same finite carrier verdict and the same classifier verdict on the terminal four-face route.*

This invariance is only a finite-request statement. It exports no quotient stream equality, no host sequence equality, no selected limit, no ambient completeness row, no countable-choice row, no metric-completeness row, and no bridge row.

Proof. Apply Definition 126.581. It is enough to identify the displayed four-face pullbacks for Q and Q': the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity and readback surface, RealUp seal boundary, and named BEDC support rows.

The accepted cofinal refinement preserves the displayed StreamName section. Use Definition 126.597 to read this preservation as equality of the finite section restrictions seen by the terminal request.

For the RegSeqRat-to-Real part, apply Definition 301.244. The compatible refinement composition transports the RegSeqRat regularity/readback surface together with the RealUp-facing seal boundary, so the refined request has the same terminal four-face display.

Now use Definition 126.578. The lock prevents the seal presentation from adding a distinguishing coordinate once the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, and support rows agree.

Finally apply Definition 126.584. The excluded quotient stream equality, host sequence equality, selected limit, ambient completeness, countable-choice, metric-completeness, and bridge rows therefore cannot separate the finite carrier verdict or the classifier verdict. □

Theorem 126.584 (StreamName finite-request section nonescape)**.** *The finite-request section pullback of Definition 126.577 exports no quotient stream equality, no host sequence equality, no selected limit, no host real equality, no countable-choice row, no ambient completion, no metric-completeness row, and no* $\mathsf{Real}^{\uparrow}$ *completeness datum. Its public surface is only the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity/readback surface, and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. First apply Definition 126.577. The RealUp-facing request has already been pulled back to the displayed StreamName, DyadicRatCore, RegSeqRat, and support-row section.

Next apply Definition 126.568. The seal lock blocks quotient stream equality, host real equality, ambient completion points, and RealUp completeness data from the seal presentation.

Now use Definition 126.569 and Definition 126.571. These results exhaust the open-phase handoff by the four displayed faces and the support rows, leaving no coordinate for host sequence equality, selected limits, countable choice, or a metric-completeness row.

Finally apply the Real-side finite-request route Definition 38.698 and the terminal-budget exhaustion of Definition 38.713. They confirm that any RealUp-facing consumer reads the same finite packet rather than an ambient completion or completeness datum. □

Theorem 126.585 (StreamName finite-request terminal support minimality)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$*-facing finite request on the* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *route has minimal public support exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius*

ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ *regularity/readback surface, the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Removing any one of these faces prevents the existing classifier pullback, exactness, kernel pullback, and nonescape route from being invoked as a public finite-request certificate.

The support is minimal only for this displayed route. It does not assert a host sequence equality, quotient stream equality, selected limit, countable-choice schedule, metric-completeness row, ambient completion point, or independent $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 126.581. The accepted request is exact only when the StreamName schedule, DyadicRatCore endpoint-radius ledger, RegSeqRat regularity/readback surface, Real seal boundary, and support rows are all present in the same finite section.

Next use Definition 126.582. The kernel part of the route identifies the displayed support rows as the public source of transport, replay, provenance, and local naming. Removing any one of those rows breaks the kernel pullback rather than producing a smaller accepted certificate.

Finally apply Definition 126.584. Since the route exports none of the excluded host, quotient, choice, completion, or Real completeness coordinates, no omitted face can be recovered from an ambient object outside the displayed finite support. □

126.44 StreamName finite-section pullback exactness

Theorem 126.586 (StreamName finite-section pullback exactness)**.** *Let Q be an accepted finite request section of a* $\mathsf{StreamName}^{\uparrow}$ *packet. If the finite windows of Q agree with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the current* $\mathsf{Real}^{\uparrow}$ *seal, then Q determines the same terminal four-face route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*. The support of this route is only the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It exports no quotient stream, selected limit, host sequence equality, ambient completeness row, countable-choice row, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFiniteSectionPullbackExactness`

Proof. First apply Definition 126.577. The accepted request section is already pulled back to the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint-radius ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and readback surface, and the support rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Next use Definition 126.449. Since the finite windows agree with the DyadicRatCore ledger and the RegSeqRat readback used by the current RealUp seal, terminality identifies the resulting route with the unique four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now apply Definition 126.605. The RealUp seal remains on the public finite rows and therefore cannot add a quotient stream, selected limit, host sequence equality, or ambient completeness coordinate to the section.

Finally use Definition 126.610. It exhausts carrier, classifier, stability, ledger, and non-escape obligations by the same four faces and the displayed support rows, leaving no countable-choice row or standard bridge row to export. □

Theorem 126.587 (StreamName finite-section classifier determinacy)**.** *Let Q and Q' be accepted finite request sections of a current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose that Q and Q' have the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary. Then the carrier verdict and*

classifier verdict read at the terminal $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *four-face route are the same for* Q *and* Q'*.*

The common route is supported only by the displayed BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It contains no quotient stream equality, selected limit, host sequence equality, ambient completeness row, countable-choice row, or standard bridge row.*

Proof. Apply Definition 126.586 first to Q and then to Q'. Each accepted section is read through the same StreamName finite window, DyadicRatCore ledger, RegSeqRat readback row, RealUp seal boundary, and displayed support rows.

Next use Definition 126.449. The shared four faces identify the terminal finite BEDC route for both sections, so a carrier or classifier verdict has no second terminal route on which to vary.

Now apply Definition 126.605. The RealUp seal remains public over the finite route and cannot contribute a hidden quotient stream, selected limit, host sequence equality, or ambient completeness coordinate that would separate the two sections.

Finally use Definition 126.610. Carrier, classifier, stability, ledger, and non-escape obligations are exhausted by the shared four faces together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Hence the two accepted sections have the same carrier and classifier verdicts at the terminal route, with no countable-choice or bridge row left outside that support. □

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFiniteSectionClassifierDeterminacy`

Theorem 126.588 (StreamName finite-section pasting exactness)**.** *Let* Q_0 *and* Q_1 *be accepted finite request sections of a current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet, and suppose their finite windows have an accepted overlap* Q_{01}*. If* Q_0 *and* Q_1 *agree on* Q_{01} *in the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, then the pasted finite section determines the same terminal four-face route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Re}$ *and the same public classifier verdict as its two restrictions.*

The pasted route exports no quotient stream, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Proof. Apply Definition 126.586 first to Q_0 and then to Q_1. Each restriction is pulled back to its displayed StreamName finite window, DyadicRatCore ledger row, RegSeqRat readback row, RealUp seal boundary, and support rows.

On the accepted overlap Q_{01}, apply Definition 126.587. The overlap hypotheses identify the four visible faces and the displayed support rows, so the two restrictions have one carrier verdict and one public classifier verdict at the common terminal route.

Now paste the finite windows along that overlap. Since both restrictions are already pullbacks of the same four-face data on Q_{01}, the pasted section has no additional face from which a second terminal route could be read.

Finally apply Definition 126.589. Every Real-facing or completion-facing consumer of the pasted section is locked to the same StreamName, DyadicRatCore, RegSeqRat, and RealUp faces. Thus the public classifier verdict and the terminal four-face route are the same on the pasted section and on its two restrictions, while the excluded quotient, host-equality, limit-selection, completeness, choice, and bridge coordinates remain unavailable. □

Theorem 126.589 (StreamName Real-RegSeq-Dyadic finite lock)**.** *Let* Q *be an accepted finite request section of a current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose that*

Q exposes the same $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal boundary as the terminal four-face route. Then every* $\mathsf{Real}^{\uparrow}$*-facing and completion-facing consumer reads the same finite lock through exactly those four faces and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The lock contains no quotient stream, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamNameFiniteSectionRealRegSeqDyadicLock`

Proof. First use Definition 126.586. The accepted section Q is already pulled back to the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal boundary, and the displayed support rows.

Next apply Definition 126.587. Any accepted request section exposing those same four faces has the same carrier and classifier verdict at the terminal route, so no Real-facing or completion-facing consumer can split the finite lock by rereading one face.

Now use Definition 126.504. It identifies every downstream Real or completion read with the terminal finite pullback once the four faces and support rows agree.

Finally combine this terminal pullback with Definition 126.610. The handoff exhausts carrier, classifier, stability, ledger, and non-escape obligations through the same BHist, hsame, Cont, Pkg, and NameCert rows. The excluded quotient, host-equality, limit-selection, completeness, countable-choice, and bridge coordinates therefore have no row from which a consumer could read them. □

Theorem 126.590 (StreamName finite-section terminal coverage)**.** *Every displayed terminal four-face lock for the current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet is realized by an accepted finite request section. More explicitly, suppose the displayed lock consists of a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the support rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert*. Then there is an accepted finite request section Q whose exposed terminal route is exactly that lock, and every Real-facing or completion-facing consumer reads Q through those rows.*

The realized section contains no quotient stream, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Proof. First apply Definition 126.504. The displayed four faces and their BHist, hsame, Cont, Pkg, and NameCert rows form the terminal pullback surface consumed by the Real and completion endpoints.

Next use Definition 126.586 to read that terminal surface as a finite request-section surface of the same packet. The exactness clause supplies the accepted finite section Q whose exposed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces are the displayed lock.

Apply Definition 126.587 to this section. Determinacy makes the carrier and classifier verdicts of Q unique at the displayed terminal route, so no consumer can choose a different finite section with the same four faces but a different public verdict.

Finally use Definition 126.610. The handoff routes all carrier, classifier, stability, ledger, and non-escape obligations through the same support rows. Thus quotient streams, host sequence equality, selected limits, ambient completeness, metric-completeness data, countable-choice rows, and bridge rows remain outside the realized finite section. □

Theorem 126.591 (StreamName terminal face iff)**.** *Let Q and Q' be accepted finite request sections of the current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet. The public finite-section carrier verdicts and classifier verdicts of Q and Q' agree if and only if Q and Q' expose the same*

terminal four-face route consisting of the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows.*

This criterion contains no quotient stream, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, countable-choice row, or standard bridge row.*

Proof. For the forward direction, assume that the public carrier and classifier verdicts of Q and Q' agree. Apply Definition 126.587 to read both verdicts at the terminal finite route determined by the visible faces.

Next apply Definition 126.589. The agreed verdicts cannot be reread through any Real-facing or completion-facing coordinate outside the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows.

Now use Definition 126.504. It recovers the common four-face terminal pullback from the agreed verdicts and the visible BHist, hsame, Cont, Pkg, and NameCert rows.

For the reverse direction, assume that Q and Q' expose the same terminal four-face route with the displayed support rows. By Definition 126.586, each section is pulled back to exactly that visible route.

Finally apply Definition 126.449. The two pullbacks have no second terminal face on which the carrier or classifier could vary, so their public finite-section carrier verdicts and classifier verdicts agree. The excluded quotient, host-equality, limit-selection, completeness, choice, metric-completion, and bridge coordinates are not part of the route in either direction. □

Theorem 126.592 (StreamName four-face terminal completeness)**.** *For a current L10 Phase Real and Completion terminal read of* $\mathsf{StreamName}^{\uparrow}$*, completeness for the present open exit surface is exactly factorization through the four visible faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *together with the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Conversely, any accepted read exposing those four faces and those support rows supplies the full carrier, classifier, stability, ledger, and non-escape surface available to the current Phase Real and Completion route. This completeness assertion excludes quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.

Proof. For necessity, start with a complete terminal read of the current open exit surface. By Definition 126.586, its accepted finite section is pulled back to the StreamName finite window, the DyadicRatCore ledger, the RegSeqRat readback, the RealUp seal boundary, and the displayed support rows.

Next apply Definition 126.587 and Definition 126.589. These results make the carrier and classifier verdicts determinate at the same four-face finite lock for every Real-facing or completion-facing consumer.

For sufficiency, suppose an accepted read exposes the four visible faces and the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Use Definition 126.504 to identify that packet with the terminal finite pullback for the current Real and Completion boundary.

Finally apply Definition 126.610. The handoff fills exactly the carrier, classifier, stability, ledger, and non-escape obligations from those rows. Since the terminal pullback has no coordinate for quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, or standard bridge data, those excluded surfaces are not part of the completeness claim. □

Theorem 126.593 (StreamName finite-section terminal reflection)**.** *Let Q be an accepted finite request section of a current L10 Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *packet. Then Q supplies the current Phase Real and Completion public reflection surface exactly when every visible row of Q factors through the four faces* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *together with the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The reflected surface contains precisely the carrier, classifier, stability, ledger, and non-escape data readable through those rows. It contains no quotient stream, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Proof. For the forward direction, suppose Q supplies the current Phase Real and Completion public reflection surface. Apply Definition 126.586. The accepted section is pulled back to the StreamName finite window, the DyadicRatCore ledger, the RegSeqRat readback, the RealUp seal boundary, and the displayed support rows.

Next use Definition 126.587. The carrier and classifier verdicts are fixed by that four-face route, so the public reflection surface has no second verdict outside the displayed support.

Apply Definition 126.588 to any two visible restrictions of the reflected finite section. On their accepted overlap, the pasted section retains the same terminal route and the same public classifier verdict, so local visible rows cannot introduce an additional reflection face.

Now apply Definition 126.589. Every Real-facing and completion-facing consumer of Q is locked to the same four faces and support rows. Hence quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows have no coordinate in the reflected surface.

For the reverse direction, suppose every visible row of Q factors through exactly the four faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. By Definition 126.592, those rows supply the full carrier, classifier, stability, ledger, and non-escape surface available to the current Phase Real and Completion route.

The same theorem also excludes every non-public row listed above. Therefore the factorized finite section supplies exactly the current public reflection surface, with no quotient, host-equality, limit-selection, completeness, choice, metric-completion, or bridge data added beyond the four-face terminal support. □

Theorem 126.594 (StreamName finite-section descent gluing)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose that every accepted overlap Q_{ij} has the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows when read from Q_i and from Q_j. Then the family descends to one public reflection surface over the four-face route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *whose restrictions are the given local sections.*

Conversely, any public reflection surface over that four-face route restricts to accepted local finite sections whose pairwise overlaps agree on exactly the same finite-window, ledger, readback, seal, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Both directions exclude quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Proof. For descent, paste the finite family by induction on its finite index set. The two-section step is exactly Definition 126.588: the pairwise overlap agreement keeps the StreamName finite

window, the DyadicRatCore ledger, the RegSeqRat readback, the RealUp seal boundary, and the displayed support rows fixed while the pasted section keeps the same public classifier verdict.

At each induction step, use Definition 126.592. The rows retained by the pasted section are already sufficient for the carrier, classifier, stability, ledger, and non-escape surface of the current Phase Real and Completion route, so no further face is available to distinguish the next local section from the accumulated paste.

Apply Definition 126.593 to the final pasted finite section. It turns the completed four-face factorization into the public reflection surface over $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$. The Real, RegSeq, and Dyadic handoff rows used above keep every read on the L10 four-object packet and leave no quotient-stream, selected-limit, host-equality, ambient completeness, metric-completion, countable-choice, or bridge coordinate to export.

For restriction, start with a public reflection surface over the same four-face route. By Definition 126.593, every visible row of that surface factors through the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal boundary, and displayed support rows. Restricting those rows to any finite local window gives an accepted local section, and restricting them to a pairwise overlap gives the same rows from both sides. □

Theorem 126.595 (StreamName finite-section tail-window terminal equivalence)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections, and let*

$$\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_n$$

be a finite chain of $\mathsf{StreamName}^{\uparrow}$ *tail-window coverage packets over the same public endpoints and the same accepted* RegSeqRat *spine. Suppose each Q_i and each $\mathcal{P}_j$ displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the public reflection surface produced by finite-section descent gluing and the terminal read produced by tail-window cover-chain coverage are the same four-face terminal route through* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqR}$ *with the same carrier and the same classifier verdict.*

Conversely, any terminal tail-window read over this four-face route restricts to compatible accepted finite sections whose overlaps display exactly those finite-window, ledger, readback, seal, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Both directions exclude quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Proof. For the finite-section side, apply Definition 126.594. The accepted local sections glue to one public reflection surface over the four-face route because their overlaps carry the same StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal boundary, and displayed support rows.

Apply Definition 126.593 to that glued surface. The result is a terminal public reflection surface on the current Phase Real and Completion route. By Definition 126.587, the carrier and classifier verdict of this surface are already determined by the displayed four-face rows.

For the tail-window side, use Definition 126.232. Every packet in the cover chain induces the same consumer-facing finite source surface. Then apply Definition 126.233 to identify the terminal RealUp read with exactly that common source surface.

It remains to compare the two terminal surfaces. The common rows assumed for the Q_i and the $\mathcal{P}_j$ are the rows named by Definition 126.589; hence both constructions factor through the same StreamName, DyadicRatCore, RegSeqRat, and RealUp faces, and no additional carrier or classifier coordinate is available on either side.

For the converse, start with a terminal tail-window read over the same four-face route. By Definition 126.232 and Definition 126.233, it is covered by one common finite source surface. Restricting that surface to the finite windows selected by the accepted current Phase Real and Completion requests gives finite sections whose overlaps inherit the same displayed rows.

Finally apply Definition 126.594 to these restricted sections and Definition 126.587 to their visible classifier rows. The resulting finite-section surface is the same terminal four-face route as the original tail-window read, and the boundary exclusions above leave no quotient stream, selected limit, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge row to distinguish them. □

Theorem 126.596 (StreamName cofinal-tail refinement invariance)**.** *Let $\{Q_i\}_{i\in I}$ and $\{R_k\}_{k\in K}$ be finite families of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose both families refine a common cofinal tail-window cover chain*

$$\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_n$$

over the same public endpoints and the same accepted RegSeqRat *spine. Assume that every Q_i, every R_k, and every $\mathcal{P}_j$ displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows on each accepted overlap. Then the finite-section descent gluing surface of $\{Q_i\}_{i\in I}$ and the finite-section descent gluing surface of $\{R_k\}_{k\in K}$ determine the same four-face terminal route through* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, with the same carrier and the same classifier verdict.*

This comparison is internal to the four visible faces. It excludes quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.

Proof. Apply Definition 126.594 to $\{Q_i\}_{i\in I}$. The overlap hypotheses paste the family to one public reflection surface over the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces, while retaining exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Apply the same descent-gluing theorem to $\{R_k\}_{k\in K}$. Since this second family refines the same cofinal tail-window cover chain and carries the same visible rows on accepted overlaps, its glued surface has the same public four-face data as the first glued surface.

Now use Definition 126.595 for each glued surface against the shared tail-window cover chain. Both glued surfaces identify with the same terminal tail-window read over the current Phase Real and Completion route. Hence their carriers and classifier verdicts are the carriers and verdicts of that shared terminal read.

Finally invoke Definition 126.589. The lock keeps the comparison on the displayed StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal boundary, and support rows. Therefore no quotient stream equality, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row can enter the invariance claim. □

Theorem 126.597 (StreamName finite-section restriction exactness)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose that every accepted overlap Q_{ij} displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows when read*

from either side. For every finite subfamily $J \subseteq I$, the restriction $\{Q_j\}_{j\in J}$ has as its descent-gluing surface exactly the terminal four-face route obtained by applying Definition 126.594 to the restricted data.

The route is the same public route through $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *and has the same carrier and classifier verdict as the tail-window terminal read determined by the ambient family. The restriction excludes quotient stream equality, selected limits, host real equality, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Proof. Restrict the finite family to J. Every accepted overlap in the restricted family is an accepted overlap in the ambient family, so the StreamName finite-window row, DyadicRatCore ledger row, RegSeqRat readback row, RealUp seal boundary, and displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows are inherited without adding any new coordinate.

Apply Definition 126.594 to the restricted family. The result is a public reflection surface over the four-face route $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$, and its local restrictions are precisely the selected sections $\{Q_j\}_{j\in J}$.

Compare this surface with the ambient terminal route by Definition 126.595. Both the ambient family and the restricted family read the same finite-window, ledger, readback, seal, and support rows into the tail-window terminal surface, so their carriers and classifier verdicts are the same terminal carrier and verdict.

Now use Definition 126.596. The restricted family is a finite refinement of the same cofinal tail-window data after dropping unused local sections, and the overlap rows retained above are exactly the rows required for the invariance comparison.

Finally apply Definition 126.589. The lock identifies the only Real-facing and completion-facing rows available to the restricted surface: the StreamName finite window, the DyadicRatCore ledger, the RegSeqRat readback, the RealUp seal boundary, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Hence quotient stream equality, selected limits, host real equality, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows have no place in the restricted descent-gluing surface. □

Theorem 126.598 (StreamName finite-section cover descent classifier reflection)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose that every accepted overlap displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows from either side. Then the descent-gluing surface for the cover is the unique public classifier reflection surface for the family.*

Concretely, every finite restriction of the cover has the same carrier verdict and the same classifier verdict as the glued surface, and every Real-facing or completion-facing consumer factors through the same four-face route $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*. The claim excludes quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Proof. First apply Definition 126.594 to the whole finite cover. The accepted-overlap hypotheses paste the local sections into a single public surface over the displayed StreamName, DyadicRatCore, RegSeqRat, and RealUp faces, with the stated BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows as its only support data.

Next apply Definition 126.597 to each finite subfamily. Every restriction inherits the same overlap rows, so its descent-gluing surface is the restricted instance of the same public four-face route and carries the same terminal carrier and classifier verdict.

Now compare refinements by Definition 126.596. Any consumer that sees the family through a finer finite tail-window cover still reads the same StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal boundary, and support rows, so the route available to the consumer is unchanged.

Finally use Definition 126.587 and Definition 126.589. Determinacy makes the carrier and classifier verdicts unique on the public surface, while the lock confines all Real-facing and completion-facing uses to the four visible faces. Hence no quotient stream equality, host sequence equality, selected limit, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row is part of the classifier reflection surface. □

Theorem 126.599 (StreamName common-refinement classifier uniqueness)**.** *Let $\{Q_i\}_{i\in I}$ and $\{R_j\}_{j\in J}$ be finite families of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose that they admit a certified finite common refinement $\{S_k\}_{k\in K}$ with refinement maps $a : K \to I$ and $b : K \to J$. For every $k \in K$, the row S_k restricts to the rows $Q_{a(k)}$ and $R_{b(k)}$, and every accepted overlap in the refinement displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows when read through either cover. Then the cover-descent classifier reflection surface determined by $\{Q_i\}_{i\in I}$ and the cover-descent classifier reflection surface determined by $\{R_j\}_{j\in J}$ are the same public four-face classifier surface.*

Equivalently, the two presentations have the same carrier verdict and the same classifier verdict on the visible route $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*. The comparison refuses quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Proof. Apply Definition 126.598 to the cover $\{Q_i\}_{i\in I}$. Apply the same theorem to the cover $\{R_j\}_{j\in J}$. The hypotheses give public classifier reflection surfaces for both covers over the same four visible faces and the same support rows.

Now apply Definition 126.596 to the certified common refinement $\{S_k\}_{k\in K}$. Each refined row restricts to the corresponding row of both covers, and every accepted refined overlap reads the same finite window, ledger, readback, seal boundary, BHist, hsame, Cont, Pkg, and NameCert rows. Thus the two cover surfaces compare through the same cofinal tail-window surface.

It remains to identify the public verdicts. Use Definition 126.587 on the two surfaces transported to the refinement surface, and use Definition 126.589 to keep the carrier and classifier reads on the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, and RealUp seal boundary.

The resulting comparison is therefore a comparison of the same public four-face classifier surface, not a comparison of quotient streams or host sequences. The lock also excludes selected limits, ambient Real completeness, metric-completeness data, countable-choice rows, and standard bridge rows from the common-refinement uniqueness claim. □

Theorem 126.600 (StreamName finite-section cover descent normal form)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose that a public cover-descent surface restricts to these sections, and that each restriction and each accepted overlap displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, the displayed* BHist*, and the componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the surface is equivalent, as a public cover-descent reflection surface, to the canonical normal form consisting exactly of*

the finite family rows, the accepted overlap compatibility rows, the common-refinement classifier surface, the four-face consumer route $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, and the nonescape exclusions for quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows.*

Conversely, any such finite family rows together with accepted overlap compatibility rows, the common-refinement classifier surface, the four-face consumer route, and the same nonescape exclusions construct a public cover-descent reflection surface with those restrictions.

Proof. Proceed by induction on the finite cover. In the empty and singleton cases the normal form is the displayed restriction row itself, with no hidden consumer route beyond the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, and RealUp seal boundary.

For the induction step, separate the last accepted section from the already glued subcover. Apply Definition 126.594 to paste the subcover surface and the last section along their accepted overlaps. The induction hypothesis identifies the subcover surface with its finite family rows, overlap rows, four-face route, and nonescape exclusions, so the glued surface has no additional support row available.

Next apply Definition 126.597 to the glued surface and to every finite subfamily obtained during the induction. Each restriction reads the same finite-window, ledger, readback, seal boundary, BHist, hsame, Cont, Pkg, and NameCert rows; hence the public surface is determined by the displayed cover data and not by a chosen enumeration of the family.

Now use Definition 126.598 to package the glued data as a cover-descent classifier reflection surface. Apply Definition 126.599 to compare any two presentations through their finite common refinement. The comparison identifies the common-refinement classifier surface as the unique classifier component of the normal form.

Finally apply Definition 126.589. The lock keeps every Real-facing and completion-facing consumer on the four visible faces, so quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable-choice rows, and standard bridge rows cannot enter the public reflection surface.

For the converse direction, start with the finite family rows and accepted overlap compatibility rows. The same descent-gluing theorem constructs the public surface, classifier reflection supplies the classifier row, and common-refinement uniqueness makes the construction independent of the finite presentation. The lock then supplies exactly the stated nonescape exclusions, which is the required public cover-descent reflection surface. □

Theorem 126.601 (StreamName finite-section cover-descent support minimality)**.** *Let* $\{Q_i\}_{i\in I}$ *be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections satisfying the overlap hypotheses of Definition 126.600. Then the canonical cover-descent normal form is accepted exactly on the four visible faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Deleting any one visible face or any one displayed support row prevents the same carrier verdict and the same classifier verdict from being presented to Real-facing or completion-facing consumers. No quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, countable-choice row, standard bridge row, aggregate formalstatus row, or bridge row can replace the deleted coordinate.*

Proof. Apply Definition 126.600. The cover-descent surface reduces to the finite family rows, the accepted overlap compatibility rows, the common-refinement classifier surface, and the four-face consumer route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

On that normal form, apply Definition 126.662. Its terminal support packet is accepted exactly on the four visible faces and the displayed BHist, componentwise hsame, Cont, Pkg, and NameCert rows, and deletion of any one of these coordinates loses the same Real-facing or completion-facing carrier and classifier verdicts.

It remains to exclude substitute coordinates. Use Definition 126.589 on the same normal-form route. The lock keeps every consumer read on the finite StreamName window, DyadicRatCore ledger, RegSeqRat readback, and Real seal boundary, so quotient stream equality, host sequence equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, countable-choice rows, standard bridge rows, aggregate formalstatus rows, and bridge rows cannot restore the deleted face or support row. □

126.45 StreamName finite-section reindex gluing

Theorem 126.602 (StreamName finite-section reindex gluing)**.** *Let Q and Q' be accepted current Phase Real and Completion finite request sections of a* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose that a certified* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex route transports the finite window of Q to the finite window of Q', and suppose that after this transport the two sections agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then Q and Q' determine the same terminal four-face route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *and the same public carrier and classifier verdict at that route.*

The glued route remains finite and displayed. It contains no quotient stream, selected limit, host equality, countable-choice row, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, aggregate status row, or standard bridge row.*

Proof. First apply Definition 126.639. The certified finite-window reindex route, together with the transported BHist, Cont, Pkg, and NameCert rows and the hsame compatibility, identifies the two finite observation sections after transport.

Next use Definition 126.642. The transported finite observation section reaches exactly the four terminal faces $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$, with the same DyadicRatCore ledger, RegSeqRat readback, RealUp seal boundary, and displayed support rows.

Now apply Definition 126.586 to Q and to Q'. Pullback exactness reads both accepted request sections through the same four-face finite route once the endpoint-radius ledger, regularity rows, readback rows, seal boundary, and displayed support rows have been transported to agreement.

Finally apply Definition 126.587. The common four-face route leaves no second public carrier surface or classifier surface on which the verdict could vary. The exclusion clauses follow from the same finite support: the construction never exports a quotient stream, selected limit, host equality, countable-choice row, metric-completeness row, ambient Real completeness row, aggregate status row, or standard bridge row. □

Theorem 126.603 (StreamName finite-section reindex gluing is choice-free)**.** *The gluing construction of Definition 126.602 remains inside the displayed finite windows of the participating* $\mathsf{StreamName}^{\uparrow}$ *packets. It uses only the certified* ProbeBundle *windows, pointwise classifier rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows attached to the two finite request sections and their reindex route.*

Consequently the gluing cannot select host indices, quotient streams, selected limits, countable-choice schedules, ambient metric-completeness witnesses, ambient $\mathsf{Real}^{\uparrow}$ *completeness rows, aggregate status rows, or standard bridge credentials.*

Proof. Start with Definition 126.602. Its hypotheses already name the certified finite-window reindex route and the displayed agreement of the endpoint-radius ledger, regular readback, Real seal, BHist, hsame, Cont, Pkg, and NameCert support rows.

Use Definition 126.639. The transported section is determined by the inspected ProbeBundle windows and pointwise classifier rows, so the construction has no coordinate from which to choose a host index or a second stream representative.

The final route is then read by Definition 126.642. The only terminal faces are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, Re and they are reached through the same displayed finite windows.

Since the carrier rows used in the proof are exactly the local transport, replay, provenance, and naming rows, no quotient stream, selected limit, countable-choice schedule, ambient completeness witness, aggregate status row, or bridge credential is introduced. □

Theorem 126.604 (StreamName finite-section reindex cocycle)**.** *Let Q_0, Q_1, and Q_2 be accepted current Phase Real and Completion finite request sections of a* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose that certified* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes transport the finite window of Q_0 to the finite window of Q_1 and the finite window of Q_1 to the finite window of Q_2. Suppose also that the two transported sections agree on the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the direct composite route from Q_0 to Q_2 determines the same terminal four-face route* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *and the same public carrier and classifier verdict as the two-step route through Q_1.*

The cocycle comparison remains finite and displayed. It contains no quotient stream, selected limit, host equality, countable-choice row, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, aggregate status row, or standard bridge row.*

Proof. First apply Definition 126.602 to the overlap from Q_0 to Q_1 and again to the overlap from Q_1 to Q_2. The shared endpoint-radius ledger, readback rows, regularity rows, seal boundary, and displayed BHist, hsame, Cont, Pkg, and NameCert rows give the two adjacent finite gluing identifications.

Next use Definition 126.641. The certified two-step finite-window route and the direct composite route from Q_0 to Q_2 present the same public reindexing data, so the intermediate section Q_1 cannot contribute a second terminal presentation.

Now apply Definition 126.642. Both the two-step route and the direct composite route reach exactly the same four terminal faces $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$, with no extra face and with the same displayed support rows.

Finally apply Definition 126.587. Since the terminal four-face route is common, the public carrier and classifier verdict read from the composite route agrees with the verdict read from the two-step route. The non-escape exclusions are inherited from the finite displayed routes used in the two gluing steps: no quotient stream, selected limit, host equality, countable choice, metric completeness, ambient Real completeness, aggregate status row, or standard bridge row is introduced. □

126.46 StreamName RegSeqRat Real public seal boundary

Theorem 126.605 (StreamName-RegSeqRat-Real public seal no-quotient boundary)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal route exported from the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff and consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *factors through the finite* BHist *rows, pointwise* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows already exposed by Definition 126.79, Definition 126.83, and Definition 126.112.*

The handoff exports no quotient stream equality, selected limit, host real equality, ambient completeness claim, or open-cover compactness claim.

Proof. First apply Definition 126.79. It identifies the public StreamName source surface as a finite-window packet: the selected history rows, the finite ProbeBundle ledger, membership rows, unary-history rows, and the visible BHist, hsame, Cont, Pkg, and NameCert support.

Next apply Definition 126.83. The RegSeqRat consumer reads the same finite packet through the regular-sequence seal route, so the Real-facing seal is reached through the displayed finite rows and their provenance rather than through a hidden stream equality or a chosen completed value.

Finally apply Definition 126.112. That boundary keeps the RealUp seal at the finite-window interface and excludes quotient streams, selected limits, host real equality, and ambient completeness data. The sibling L10 consumers chapter 301, chapter 291, and chapter 38 therefore receive only the displayed finite rows and local certificates, not an open-cover compactness claim. □

Theorem 126.606 (StreamName-RegSeqRat-Real public seal finite-section determinacy)**.** *Let $\mathcal{S}_0$ and $\mathcal{S}_1$ be public* $\mathsf{Real}^{\uparrow}$ *seal routes exported from the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff and consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*. Suppose they display the same* $\mathsf{StreamName}^{\uparrow}$ *finite section, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the same local* $\mathsf{Real}^{\uparrow}$ *finite-section readback, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{S}_0$ and $\mathcal{S}_1$ determine the same finite public seal section.

This determinacy is only a statement about the public finite section of the BEDC ledger. It does not assert quotient-stream equality, selected limits, host real equality, countable choice, ambient completeness, an aggregate formalstatus row, bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 126.586 and Definition 126.589. The common StreamName finite section is pulled back to the same DyadicRatCore ledger, RegSeqRat readback, local Real seal row, and displayed support rows.

Next apply Definition 301.237. Since the two routes expose the same StreamName window, dyadic ledger, RegSeqRat regularity and classifier rows, local Real readback, and support rows, they come from one common finite RegSeqRat source section.

Now apply Definition 38.720. The local Real readback of each route is accepted exactly through the same ordered four-face packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The two public seal routes are therefore projections of one finite BEDC ledger. Projecting that ledger to the public seal coordinate gives the same finite public seal section for $\mathcal{S}_0$ and $\mathcal{S}_1$, and the excluded quotient, limit, host-equality, choice, completeness, aggregate status, and bridge coordinates have no displayed row from which to vary. □

Theorem 126.607 (StreamName-RegSeqRat-Real public seal window induction)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal section exported from a* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff and consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *is generated by induction over the displayed finite rows. The generating rows are precisely the visible* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *readback row, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The induction step appends one visible window row and preserves the same public seal section exactly when the finite-section determinacy hypotheses of Definition 126.606 are preserved.

The generated section contains no quotient stream equality, selected limit, host real equality, countable choice, metric-completeness datum, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. The base section is the empty displayed window together with its finite support rows. Apply Definition 126.605. It identifies the public seal boundary as a route through the finite BHist, hsame, Cont, Pkg, and NameCert rows and supplies no quotient, limit, host-equality, choice, completeness, or bridge coordinate.

For the induction step, append one visible $\mathsf{StreamName}^{\uparrow}$ window row and its matching dyadic, RegSeqRat, local Real, and support rows. Apply Definition 126.586 and Definition 126.589 to transport the appended window through the same finite four-face packet.

Next apply Definition 301.237. The RegSeqRat consumer sees only the displayed window, ledger, regularity, classifier, readback, and support rows, so the source section after the append is the unique finite source section with those rows.

Finally apply Definition 38.720 and then Definition 126.606. The local Real readback is accepted through the same ordered four-face packet, and determinacy says that preserving the displayed finite rows preserves the same public seal section. Since every step appends only such rows, the whole public section is generated by the stated finite-window induction. □

Theorem 126.608 (StreamName-RegSeqRat-Real public seal support intersection)**.** *Let* $\mathcal{S}_0$ *and* $\mathcal{S}_1$ *be public* $\mathsf{Real}^{\uparrow}$ *seal routes exported from the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff and consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*. Suppose their visible rows agree on the* $\mathsf{StreamName}^{\uparrow}$ *finite section, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *readback, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then $\mathcal{S}_0$ *and* $\mathcal{S}_1$ *have the same finite support intersection inside the public seal table. Any disagreement in that intersection either deletes one of the displayed rows or introduces a coordinate outside the public table exposed by the seal route.*

The assertion is only about the common finite support of the two displayed routes. It does not identify quotient streams, select limits, assert host real equality, add bridge rows, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 126.605. It identifies the public seal inventory as exactly the displayed finite-window rows, dyadic ledger rows, RegSeqRat regularity and classifier rows, local Real readback, and the support rows BHist, hsame, Cont, Pkg, and NameCert.

Next apply Definition 126.606. The hypotheses say that the two routes agree on every row that can appear in the public seal section, so projecting either route to the visible support table gives the same finite section.

Finally apply Definition 126.607. Appending visible rows preserves the section precisely when the displayed window, ledger, classifier, readback, and support rows are preserved. Thus a proposed difference in the support intersection has only two possibilities: it removes a displayed row already fixed by the common section, or it names a coordinate that the no-quotient boundary says is not in the public seal table. □

Theorem 126.609 (StreamName-RegSeqRat-Real public seal support union)**.** *Let* $\mathcal{S}_0$ *and* $\mathcal{S}_1$ *be public* $\mathsf{Real}^{\uparrow}$ *seal routes exported from* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoffs and consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*. Suppose the two routes agree on their finite support intersection in the sense of Definition 126.608. Then their displayed union is again the same finite public seal section exactly*

when the union contains only the $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the local* $\mathsf{Real}^{\uparrow}$ *readback rows, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The union row contains no quotient stream equality, selected limit, host real equality, countable choice, metric or ambient completeness coordinate, aggregate formalstatus row, or bridge row.

Proof. First apply Definition 126.608. The two public routes agree on the overlap of their displayed finite supports, so the proposed union has a common boundary along the rows already visible in both sections.

For the forward direction, apply Definition 126.605. A union that is again a public seal section must still factor through the finite-window inventory: StreamName rows, dyadic ledger rows, RegSeqRat regularity and classifier rows, local Real readback rows, and the support rows BHist, hsame, Cont, Pkg, and NameCert. The same boundary excludes quotient streams, selected limits, host real equality, countable choice, completeness coordinates, aggregate formalstatus data, and bridge rows.

For the reverse direction, assume the displayed union contains only those public rows. Apply Definition 126.607 to generate the union by adjoining the visible window rows one at a time. At each adjoining step the overlap condition supplies the fixed intersection, and Definition 126.606 identifies the resulting display as the same finite public seal section.

Thus the finite-cover gluing route has exactly one admissible public union: the union of the displayed StreamName, DyadicRatCore, RegSeqRat, local Real, and support rows. Any extra coordinate would lie outside the no-quotient public boundary, while omitting one of the displayed rows would break the finite-window induction step or the common support intersection. □

126.47 StreamName RegSeqRat Dyadic Real obligation handoff

Theorem 126.610 (StreamName RegSeqRat Dyadic Real obligation handoff)**.** *Let U be an accepted L10 consumer whose four visible faces are a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and a* $\mathsf{Real}^{\uparrow}$ *seal boundary. Then every carrier, classifier, stability, ledger, and non-escape obligation for U factors through those four faces together with the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows. No quotient stream equality, host sequence equality, selected limit, ambient completeness row, metric-completeness datum, or bridge row is exported by this handoff.*

Proof. First use open-phase terminal exactness, Definition 126.419. It places every seal-facing L10 read on the ordered StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal route, with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply the terminal factorization and boundary theorems, Definition 126.422 and Definition 126.423. They make the four-face route the consumer contract for seal reads and refuse quotient streams, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, and bridge rows as alternate factors.

Now use the DyadicRatCore window handoff supplied by Definition 126.442 and the DyadicRegSeq route factorization of Definition 126.444. These keep the observation ledger and readback row on the same finite StreamName window instead of on a detached host sequence or selected limit.

Finally combine the ledger coverage of Definition 126.420 with the public seal no-quotient boundary of Definition 126.605. The resulting surface is exactly the four visible faces and their displayed support rows, so carrier, classifier, stability, ledger, and non-escape obligations have no remaining source outside that handoff. □

Theorem 126.611 (StreamName RegSeqRat Dyadic Real obligation terminality)**.** *Let U be an accepted L10 consumer whose visible faces are exactly a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and a* $\mathsf{Real}^{\uparrow}$ *seal boundary. Then the carrier, classifier, stability, ledger, and non-escape obligations form the terminal finite* BEDC *interface for that read. Every downstream consumer factors through the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows; and two accepted presentations of the same read are the same whenever those rows agree componentwise. This terminal interface exports no quotient stream equality, host sequence equality, selected limit, ambient completeness row, metric-completeness datum, bridge row, trichotomy row, open-cover compactness row, or choice datum.*

Proof. Begin with the four-face obligation handoff of Definition 126.610. It already places the carrier, classifier, stability, ledger, and non-escape obligations on the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal boundary, supported only by the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

The terminal seal exactness of Definition 126.419 fixes the ordered route through those four faces. The Real-RegSeq factorization of Definition 126.422 and the seal-boundary theorem Definition 126.423 make any accepted downstream consumer read the same finite packet rather than a detached stream, host sequence, limit, completeness assertion, or bridge row.

It remains to exclude an additional source row between the dyadic ledger and the regular-rational readback. Use the window anchor Definition 126.442 together with the Dyadic-RegSeq route factorization Definition 126.444. These results pin every observation and tolerance coordinate to the same finite StreamName window before the RegSeqRat and RealUp consumers can inspect it.

Finally apply the ledger coverage theorem Definition 126.420 and the public seal no-quotient boundary Definition 126.605. Coverage says every public coordinate of the packet is one of the displayed faces or support rows, while the no-quotient boundary excludes quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, and bridge rows as alternate public data.

Trichotomy, open-cover compactness, and choice data are not among the displayed rows and cannot enter through any of the preceding terminal routes. Hence no fifth source row remains: every downstream consumer factors through the listed finite interface, and componentwise agreement of those rows identifies the two accepted consumer presentations. □

Theorem 126.612 (StreamName RegSeqRat Dyadic Real four-face exactness)**.** *An accepted L10 seal-facing read is admissible exactly when its public carrier, classifier, stability, ledger, and non-escape obligations factor through the four displayed faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*, together with the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Conversely, any packet with precisely those four faces and no quotient-stream, host-sequence, selected-limit, ambient-completeness, metric-completeness, bridge, trichotomy, open-cover compactness, or choice row supplies the admissible public interface for the same L10 seal-facing read.*

Proof. For the forward direction, start with Definition 126.610. It places the carrier, classifier, stability, ledger, and non-escape obligations on the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal boundary.

Apply Definition 126.611 and Definition 126.419. These make the same four faces terminal for the accepted read and keep every support coordinate among the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert rows.

It remains to exclude alternate public data. Use Definition 126.605 and Definition 126.444. The first refuses quotient streams, host sequence equality, selected limits, ambient completeness, metric-completeness data, and bridge rows; the second pins the dyadic ledger and regular-rational readback to the same finite StreamName route. Trichotomy, open-cover compactness, and choice rows have no coordinate in that route.

For the reverse direction, suppose a packet displays exactly those four faces and support rows and contains none of the excluded rows. Repack the StreamName, RegSeqRat, DyadicRatCore, and RealUp coordinates through terminality. The no-quotient boundary supplies the non-escape clause, and the route factorization supplies the finite dyadic-readback ledger. The resulting packet is therefore the admissible public interface for the L10 seal-facing read. □

Theorem 126.613 (StreamName L10 four-face exit gate)**.** *An accepted L10 seal-facing read satisfies the current Phase Real and Completion exit gate exactly when its public surface factors through the four displayed faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with displayed* BHist *rows, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. This gate exports no quotient stream, host sequence equality, selected limit, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or choice row.*

Proof. For the forward direction, apply the obligation handoff theorem Definition 126.610. It places every carrier, classifier, stability, ledger, and non-escape obligation on the StreamName finite window, RegSeqRat readback row, DyadicRatCore observation ledger, and RealUp seal boundary, with only the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support rows.

Next apply terminality, Definition 126.611. Terminality says that every downstream consumer of the same accepted read factors through those rows and that componentwise agreement of the rows identifies accepted presentations. Hence the exit surface has no fifth source row.

The four-face exactness theorem Definition 126.612 supplies the converse direction. Any packet displaying precisely the StreamName, RegSeqRat, DyadicRatCore, and RealUp faces with the listed support rows is the admissible public interface for the same L10 seal-facing read.

Finally combine the exclusion clauses in those three results. Quotient streams, host sequence equality, selected limits, ambient completeness, metric-completeness, bridges, trichotomy, open-cover compactness, and choice rows are not coordinates of the displayed terminal packet. The Lean route names `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectExitRoute` and `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectTerminalPullback` identify the corresponding formalization touchpoints for this four-face route. □

Theorem 126.614 (StreamName L10 four-face exit gate reindex stability)**.** *Let U be an accepted L10 seal-facing read satisfying the four-face exit gate of Definition 126.613. Suppose U is transported along a support-row-preserving public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex route. Then the transported read satisfies the same exit gate over the same visible faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with the same displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows. The reindexed gate exports no quotient streams, host sequence equality, selected limits, ambient completeness, metric completeness, bridge rows, trichotomy, open-cover compactness, or choice.*

Proof. Start with the original gate, Definition 126.613. It identifies the accepted read with the four visible faces and the displayed BHist, hsame, Cont, Pkg, and local NameCert support rows, and it excludes all non-displayed source rows.

Transport that terminal surface along the certified public reindex route by Definition 126.647. Support-row preservation keeps the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal on the same public route.

Next apply the pullback statement, Definition 126.648. It says that a reindexed terminal packet whose four faces and support rows agree is read through the same finite four-face surface, not through the shape of the reindexer or a hidden host coordinate.

Finally use the converse admissibility direction of Definition 126.612. Since the transported packet displays exactly the same four faces and the same support rows, it is again an admissible L10 seal-facing read satisfying the exit gate. The exclusion list is preserved because none of quotient streams, host sequence equality, selected limits, ambient completeness, metric completeness, bridge rows, trichotomy, open-cover compactness, or choice is a coordinate of that transported packet. □

Theorem 126.615 (StreamName L10 four-face ordered exit gate)**.** *An accepted Phase Real and Completion seal-facing read satisfies the public L10 exit gate only when its four visible faces are read along the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The $\mathsf{StreamName}^{\uparrow}$ *finite window is read first, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger is anchored to that window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is read from those anchored rows, and the* $\mathsf{Real}^{\uparrow}$ *seal is read last. Merely listing the four faces does not satisfy the gate if the route permutes them, detaches the dyadic ledger from the StreamName window, reads the RegSeqRat row before the ledger, or bypasses directly to the RealUp seal. The ordered gate exports no quotient stream, host sequence equality, selected limit, choice row, metric-completeness row, bridge row, or ambient Real-completeness row.*

Proof. First apply the four-face exit gate, Definition 126.613. It identifies the accepted seal-facing public surface with the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and with the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows.

Next read the gateway order from Definition 126.488. In that gateway the finite source is consumed as $S \longrightarrow D \longrightarrow R \longrightarrow E$: the StreamName finite window precedes the DyadicRatCore ledger, the RegSeqRat readback is available only from the paired window-and-ledger route, and the RealUp seal is available only after that readback.

Apply terminality, Definition 126.487. Since every terminal Real-completion route factors through the same four-face packet and support rows, a consumer cannot satisfy the public gate by permuting the faces, skipping the dyadic ledger, detaching the RegSeqRat row from the anchored ledger, or jumping directly from the StreamName window to the RealUp seal.

Finally use the reindex stability theorem, Definition 126.614. It preserves the same displayed faces and support rows under support-row-preserving public reindexing, so the order-sensitive gate is stable under finite-window transport while retaining the exclusions of quotient streams, host sequence equality, selected limits, choice, metric completeness, bridges, and ambient Real completeness. □

Theorem 126.616 (StreamName L10 ordered exit composition determinacy)**.** *Let U and V be accepted Phase Real and Completion seal-facing reads whose public surfaces both satisfy the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that the two-step read $U \to V$ and a certified composite route both preserve the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support rows. Then the two-step read and the certified composite expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal surface exactly when they agree*

on the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. This determinacy exports no quotient stream, selected limit, choice datum, metric-completeness row, bridge row, or ambient Real-completeness row.*

Proof. For the forward direction, apply the four-face exit gate Definition 126.613. It identifies each accepted public surface with the same four visible faces and with only the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows.

Next use reindex stability, Definition 126.614. The two-step read and the certified composite preserve the support rows by hypothesis, so transporting along either route keeps the same finite StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback row, and RealUp seal row.

Now apply the ordered exit gate, Definition 126.615. Since both routes read the faces in the order $\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$, the terminal Real seal is reached only after the same anchored ledger and readback rows. Agreement on those four rows therefore forces the two terminal seal exposures to coincide.

For the reverse direction, suppose the two-step read and the certified composite expose the same terminal $\mathsf{Real}^{\uparrow}$ seal surface through the ordered route and preserved support rows. Terminality in the four-face gate leaves no fifth coordinate from which they could differ, so any disagreement must already appear on the StreamName window, DyadicRatCore ledger, RegSeqRat readback, or RealUp seal row. Hence equality of the terminal exposure is equivalent to componentwise agreement on those four rows.

The exclusion clauses are inherited from the three gate theorems just used. Quotient streams, selected limits, choice data, metric-completeness rows, bridge rows, and ambient Real-completeness rows are not coordinates of either accepted ordered route, so they cannot witness a distinct composite exit surface. □

Theorem 126.617 (StreamName L10 ordered exit face-status readback)**.** *Let U be an accepted Phase Real and Completion seal-facing read whose public surface satisfies the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then every seal-facing read of closure-status, formal-status, Lean-target, and bridge-status data for U is facewise: the StreamName entry is read from the $\mathsf{StreamName}^{\uparrow}$ *chapter face, the dyadic ledger entry from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *chapter face, the regular-rational readback entry from the* $\mathsf{RegSeqRat}^{\uparrow}$ *chapter face, and the terminal seal entry from the* $\mathsf{Real}^{\uparrow}$ *chapter face. The ordered route creates no route-level closure-status row, formal-status row, Lean target, bridge row, quotient stream, selected limit, choice datum, metric-completeness row, or ambient Real-completeness claim.*

Proof. Apply the ordered exit gate, Definition 126.615. It fixes the only admissible read order as $\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ and places the public exit surface on those four visible faces with their displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows.

Next apply ordered composition determinacy, Definition 126.616. If two accepted seal-facing reads preserve those support rows, their terminal exposure is determined exactly by agreement on the four face rows. Hence a status reader cannot obtain an additional row from the route composite itself; any such row would distinguish two composites with identical face data.

It remains to identify the status sources. Use Definition 126.507, Definition 291.112, Definition 38.833, and Definition 38.834, together with the terminal pullback formalization touchpoint `BEDC.Derived.StreamNameUp.StreamNameRealCompletionFourObjectTerminalPullback`. They

keep closure-status, formal-status, Lean-target, and bridge-status reads attached to the object faces that expose them, rather than to the ordered path between the faces.

The exclusion list follows from the same gate and determinacy results. Quotient streams, selected limits, choice data, metric-completeness rows, bridge rows, route-level Lean targets, and ambient Real-completeness claims are not coordinates of the four-face packet, so the ordered route supplies no credential beyond the four facewise chapter reads. □

Theorem 126.618 (StreamName L10 four-face terminal coherence)**.** *Let U be an accepted Phase Real and Completion seal-facing read whose public surface follows the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the terminal public credential for U is exactly the componentwise agreement of the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows. No route-level closure-status row, formal-status row, Lean target, bridge row, quotient stream, selected limit, choice datum, metric-completeness row, or ambient Real-completeness claim is a fifth coordinate of that credential.*

Proof. First apply the four-face exit gate, Definition 126.613. It identifies the admissible public surface with the four visible object faces and with only the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support rows.

Next use reindex stability, Definition 126.614. Any support-row-preserving finite-window transport preserves the same four object faces and the same support rows, so terminal agreement cannot depend on the presentation of the public route.

Apply the ordered exit gate, Definition 126.615. The order

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

forces the tolerance ledger to be anchored to the StreamName finite window, the RegSeqRat readback to be read from the anchored rows, and the RealUp seal to be read last.

Now apply ordered composition determinacy, Definition 126.616. It makes equality of the terminal seal exposure equivalent to componentwise agreement on the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback row, and RealUp seal row, with the displayed support rows preserved.

Finally use face-status readback, Definition 126.617. Closure-status, formal-status, Lean-target, and bridge-status data remain attached to the four object faces rather than to the route composite, while quotient streams, selected limits, choice data, metric-completeness rows, and ambient Real-completeness claims are excluded by the same status-axis noncollapse. Hence the four object faces are jointly exact, and no route-level credential supplies a fifth coordinate. □

Theorem 126.619 (StreamName L10 ordered exit common-refinement naturality)**.** *Let U and V be accepted Phase Real and Completion seal-facing reads whose public surfaces follow the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that U and V admit a support-row-preserving common finite-window refinement W. Thus the transports from U to W and from V to W preserve the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support. Then the transported terminal public credentials of U and V over W agree if and only if the two transported reads agree on the four face rows over W: the*

$\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. This comparison exports no quotient stream, selected limit, choice datum, metric-completeness row, route-level formal-status row, route-level Lean target, route-level bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First transport both accepted reads to the common finite-window refinement. Apply reindex stability, Definition 126.614, separately to the $U \to W$ and $V \to W$ transports. The support-row-preserving hypothesis keeps the same displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support, so both transported reads still present only the four object faces over W.

For the forward direction, suppose the transported terminal public credentials agree over W. Apply ordered exit composition determinacy, Definition 126.616, to the two transported ordered routes. Since the terminal exposure has no coordinate other than the ordered four-face packet and the displayed support rows, equality of the terminal credential forces agreement of the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and $\mathsf{Real}^{\uparrow}$ seal row over the common refinement.

For the reverse direction, assume those four transported face rows agree over W. Apply Definition 126.616 again to identify the terminal exposure from the componentwise four-face data. Then use terminal coherence, Definition 126.618, to repack that componentwise agreement as equality of the transported terminal public credentials.

It remains to keep the status and bridge data from becoming an additional comparison coordinate. Use face-status readback, Definition 126.617. Closure-status, formal-status, Lean-target, and bridge-status entries remain facewise along the ordered route, so the common refinement compares only the four object rows and their displayed support.

The exclusion clauses follow from the same three inputs. Quotient streams, selected limits, choice data, metric-completeness rows, route-level formal-status rows, route-level Lean targets, route-level bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not produced by the support-row-preserving transports and are not fifth coordinates of the terminal credential. □

Theorem 126.620 (StreamName L10 ordered exit zigzag confluence)**.** *Let $U_0, \ldots, U_n$ be a finite zigzag of accepted Phase Real and Completion seal-facing reads, each of whose public surfaces follows the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that every adjacent edge between U_i and U_{i+1} is witnessed by a support-row-preserving common finite-window refinement, preserving the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support. Then the endpoints U_0 and U_n expose the same terminal public credential along the zigzag if and only if every adjacent comparison agrees on the four face rows: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row, over the displayed support. The zigzag comparison exports no quotient stream, selected limit, countable choice datum, metric-completeness datum, aggregate formal-status row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Proceed by induction on the finite zigzag. The empty and one-node cases are immediate from terminal coherence, Definition 126.618, since there is no adjacent comparison and the terminal credential is exactly the four-face packet with its displayed support.

For the induction step, split the zigzag as an initial finite zigzag from U_0 to U_k followed by one adjacent common-refinement edge from U_k to U_{k+1}. Apply the induction hypothesis to the initial zigzag. This identifies endpoint agreement from U_0 to U_k with agreement on all adjacent four-face rows in that initial segment.

Now apply common-refinement naturality, Definition 126.619, to the final edge between U_k and U_{k+1}. It identifies agreement of the transported terminal credentials on that edge with agreement of the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback row, and RealUp seal row over the common finite-window refinement.

For the forward direction, endpoint equality across the whole zigzag restricts to the initial endpoint comparison and to the final adjacent edge. The induction hypothesis gives the four-face agreements in the initial segment, while common-refinement naturality gives the four-face agreement on the last edge.

For the reverse direction, assume every adjacent comparison agrees on the four face rows. The induction hypothesis gives terminal credential equality from U_0 to U_k, and common-refinement naturality gives terminal credential equality from U_k to U_{k+1}. Compose these equalities using ordered exit composition determinacy, Definition 126.616, and repack the componentwise data by terminal coherence, Definition 126.618.

Finally apply face-status readback, Definition 126.617. Status and bridge reads remain attached to the four object faces at every node and every adjacent refinement, so the finite zigzag creates no aggregate formal-status row, aggregate bridge row, or standard bridge row. Quotient streams, selected limits, countable choice data, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness are likewise absent from the ordered route and therefore cannot become confluence coordinates. □

Theorem 126.621 (StreamName L10 ordered exit zigzag terminality)**.** *Let $U_0, \ldots, U_n$ be a finite zigzag of accepted Phase Real and Completion seal-facing reads, each of whose public surfaces follows the ordered route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume that every adjacent edge between U_i and U_{i+1} is witnessed by a support-row-preserving common finite-window refinement, and that every adjacent comparison agrees on the four face rows: the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. Then the whole zigzag determines a unique terminal public credential, namely the credential determined exactly by the common StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback row, RealUp seal row, displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support. This terminal packet contains no quotient stream, selected limit, countable choice datum, metric-completeness datum, aggregate formal-status row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Induct on the finite zigzag. In the empty and one-node cases, terminal coherence, Definition 126.618, identifies the only public credential with the displayed four-face packet and its support rows.

For the induction step, split the zigzag as an initial zigzag from U_0 to U_k followed by one adjacent edge from U_k to U_{k+1}. Apply the induction hypothesis to obtain the unique terminal credential for the initial zigzag. Apply common-refinement naturality, Definition 126.619, to the final edge, so the final adjacent four-face agreement transports the same credential across the common finite-window refinement.

Next use zigzag confluence, Definition 126.620. It upgrades adjacent agreement on the four face rows to endpoint agreement for the whole finite zigzag, so there is only one candidate terminal public credential compatible with all adjacent refinements.

Finally apply terminal coherence and face-status readback, Definition 126.618 and Definition 126.617. Terminal coherence identifies the unique credential with the common StreamName

finite window, DyadicRatCore tolerance ledger, RegSeqRat readback row, RealUp seal row, displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support. Face-status readback keeps status and bridge data attached to the four object faces, so quotient streams, selected limits, countable choice data, metric-completeness data, aggregate formal-status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot become extra terminal coordinates. □

126.48 StreamName terminal window fusion

Theorem 126.622 (StreamName Real-completion terminal-window fusion)**.** *Every accepted Phase Real and Completion read through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

is determined by the fused terminal window consisting of the StreamName finite request window, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Equivalently, two accepted reads with the same fused terminal window have the same public classifier surface and the same terminal factorization through the four Real and Completion exit faces.

The fused packet exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. Begin with the four-object exit route of Definition 126.500. It identifies every accepted current read with the four displayed faces and with only the BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply the terminal pullback and terminal exactness results, Definition 126.504 and Definition 126.505. These results say that agreement on the StreamName finite request window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the RealUp seal, and the displayed support rows is precisely the terminal boundary for public acceptance.

Now transport this terminal boundary to the classifier surface. Use Definition 126.514: once the fused terminal window agrees componentwise, the public classifier has no second coordinate on which to distinguish the two reads.

It remains to bind the dependency and handoff rows. The dependency exactness theorem Definition 126.518 names the same four faces as necessary and sufficient, while Definition 126.519 reads the StreamName finite-window handoff, DyadicRatCore ledger handoff, and RegSeqRat-to-Real seal as route witnesses for those existing rows.

Finally apply the no-choice route factorization, Definition 126.522. It removes every attempted selector, quotient, completeness, metric, choice, or bridge coordinate from the packet. Therefore the fused terminal window determines the accepted Phase Real and Completion read exactly through the four displayed exit faces and the five displayed support rows. □

Theorem 126.623 (StreamName terminal fusion common-refinement stability)**.** *Let* $\mathcal{W}_L$ *and* $\mathcal{W}_R$ *be accepted Phase Real and Completion reads whose* $\mathsf{StreamName}^{\uparrow}$ *finite windows admit a certified common refinement* $\mathcal{R}$*. Suppose the refinement transports the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the fused terminal-window packet read through $\mathcal{W}_L$ is the same packet read through $\mathcal{W}_R$. In particular the public classifier surface is unchanged by passing to $\mathcal{R}$ and restricting back to either selected window.

The stability statement adds no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. First read the common refinement at the StreamName boundary. Apply Definition 126.308 and Definition 126.234. The refinement $\mathcal{R}$ supplies one finite-window classifier, one ProbeBundle row, the same InBundle membership rows, the same selected histories, and the same displayed hsame/Cont/Pkg ledger rows for both selected windows.

Next pass from the source tuple to the Real and Completion exit packet. Use Definition 126.240: every downstream RegSeqRat handoff and every finite-window $\mathsf{Real}^{\uparrow}$ handoff read from $\mathcal{R}$ factors through that common source tuple. The hypotheses identify the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, and local NameCert rows as the same transported data on the two restrictions.

Now apply the terminal pullback theorem Definition 126.504. Since the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and the displayed support rows agree after transport along $\mathcal{R}$, the two accepted reads determine the same terminal finite BEDC pullback. The terminal-window fusion theorem Definition 126.622 repacks this pullback as one fused terminal-window packet.

Finally transport the equality of terminal packets to the public classifier surface by Definition 126.514. The no-choice route factorization Definition 126.522 removes every selector, quotient, completeness, metric, choice, and bridge coordinate, so a common refinement cannot create a second public presentation of the same terminal route. □

Theorem 126.624 (StreamName terminal fusion common-tail Real boundary pullback)**.** *Let $\mathcal{W}$ be an accepted Phase Real and Completion read, and let its fused terminal window be transported through a* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail window packet with common refinement W_*. The transported fused terminal packet determines exactly the same public* $\mathsf{Real}^{\uparrow}$ *seal boundary as the common-tail Real boundary equivalence of Definition 301.186.*

Equivalently, two such transported reads have the same public classifier surface if and only if they agree on the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, and the displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

No quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row is part of this pullback boundary.*

Proof. First use Definition 126.623. The common refinement makes the selected $\mathsf{StreamName}^{\uparrow}$ finite windows one fused terminal packet after the shared $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, the $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows have been transported through W_*.

Next apply Definition 301.186. The common-tail packet and the finite-request $\mathsf{Real}^{\uparrow}$ seal boundary determine one another exactly on the StreamName schedule and refinement rows, the dyadic endpoint-radius-distance ledger, the RegSeqRat classifier and regularity rows, the Real seal-source rows, and the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Now compare this boundary with the four-face public interface. By Definition 38.775, those same finite rows form the public pullback surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Thus the common-tail presentation cannot add or remove a public row from the fused terminal presentation.

Finally transport the shared four-face surface to the public classifier by Definition 126.514. Agreement on the displayed rows is therefore sufficient for public classifier agreement, and any public classifier agreement reads back to the same displayed rows through the common-tail Real boundary equivalence. The no-choice factorization of Definition 126.522 excludes selectors, quotient stream equality, host sequence equality, chosen limits, completeness, metric-completeness data, countable choice, and bridge rows from serving as another boundary coordinate. □

Theorem 126.625 (StreamName terminal fusion finite-request section exactness)**.** *Let $\mathcal{W}$ be an accepted fused terminal-window packet for the* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *route, and view its terminal request as a finite-request section. Then the displayed rows of $\mathcal{W}$ are exactly the finite-request section pullback rows: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Conversely, any finite-request section carrying precisely those displayed rows determines the same fused terminal-window packet on the public terminal four-face route.

This identification exports no quotient stream equality, no host sequence equality, no selected limit, no ambient completeness row, no metric-completeness data, no countable-choice row, and no bridge row.

Proof. For the forward direction, first read $\mathcal{W}$ through Definition 126.622. The fused terminal packet is exhausted by the finite StreamName window, the DyadicRatCore ledger, the RegSeqRat readback, the Real seal, and the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ support rows.

Next view the same request as a finite-request section and apply Definition 126.577. This identifies the section with the same StreamName finite schedule, DyadicRatCore endpoint, radius, and distance ledger, RegSeqRat classifier and regularity/readback rows, Real seal-source rows, and support provenance.

For the reverse direction, start with a finite-request section carrying those rows. Apply Definition 126.579 to recover the public carrier and classifier verdict on the terminal four-face route. Then apply Definition 126.624 to repack the same row family as the common-tail Real boundary of one fused terminal packet.

Finally use Definition 126.584. It removes quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, countable choice, and bridge rows from the section surface. Hence the finite-request section and the fused terminal-window packet are interchangeable exactly on the displayed BEDC-visible L10 rows. □

Theorem 126.626 (StreamName terminal fusion source exhaustion)**.** *Every accepted current Phase Real and Completion read whose terminal request is presented as a fused* $\mathsf{StreamName}^{\uparrow}$ *window factors through exactly the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and exactly the support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Conversely, a terminal request carrying those four faces and those five support rows determines the same public finite-request section as the fused terminal window presentation.

The source-exhaustion statement exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness data, no countable-choice row, and no bridge row.*

Proof. First apply Definition 126.622. It identifies an accepted current read with the fused terminal packet over the StreamName finite request window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the Real seal, and the displayed support rows.

Next apply Definition 126.625. The finite-request section exposes exactly the same four faces and exactly the same BHist/hsame/Cont/Pkg/NameCert support rows, and no further public row is available on the section surface.

For the reverse direction, start with a terminal request carrying the displayed faces and support rows. Use Definition 126.624 to read those rows as the common-tail $\mathsf{Real}^{\uparrow}$ boundary of the fused terminal presentation, and then read the resulting packet back through finite-request section exactness.

Finally apply Definition 126.522. Quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, countable choice, and bridge rows have no coordinate in the route, so the four displayed faces and the five support rows exhaust the accepted current source. □

Theorem 126.627 (StreamName terminal-window fusion reindexing compatibility)**.** *Let* $\mathcal{W}_0$ *and* $\mathcal{W}_1$ *be accepted Phase Real and Completion terminal-window packets. Suppose they differ only by a certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex route, and suppose that route transports componentwise the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, and the public endpoint rows. Then* $\mathcal{W}_0$ *and* $\mathcal{W}_1$ *determine the same fused terminal-window packet and the same public four-face route through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Conversely, any fused terminal-window packet transported by the certified identity reindexer or by a composable certified reindexer reads back to the same public finite observation section. The compatibility claim introduces no quotient stream equality, host sequence equality, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or standard bridge row.*

Proof. First identify the finite public section. Apply Definition 126.639 to the two displayed $\mathsf{StreamName}^{\uparrow}$ finite-window packets. The certified reindex route supplies the hsame-compatible map, the common endpoint rows, and the componentwise transport of the displayed BHist, Cont, Pkg, and NameCert rows, so both packets expose one public finite observation section.

Next carry that section to the terminal L10 route. Use Definition 126.642. The transported finite section presents the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal as the terminal Phase Real and Completion route.

Now repack the transported rows as a fused terminal packet. Apply Definition 126.622 to read the common four-face route as one fused terminal-window packet, and apply Definition 126.626 to see that no source coordinate remains beyond the four faces and the displayed BHist/hsame/Cont/Pkg/NameCert support rows.

For the converse, read an identity-reindexed or composably reindexed fused packet through the same terminal coverage theorem. Identity and composable certified reindexers return the same

finite observation section before fusion, and the fusion and source-exhaustion theorems repack exactly that section as the same public terminal route.

The exclusions follow because every step is confined to the displayed finite-window route, the transported tolerance ledger, the readback rows, the seal row, and the named support rows. None of the cited results exports quotient stream equality, host sequence equality, selected limits, countable-choice data, ambient completeness, metric-completeness data, or a standard bridge row. □

Theorem 126.628 (StreamName terminal fusion finite-cover cocycle exactness)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows. Suppose the pairwise windows $\mathcal{W}_{ij}$ and triple windows $\mathcal{W}_{ijk}$ carry the pasted-source hypotheses for finite StreamName–*$\mathsf{RegSeqRat}^{\uparrow}$ *covers, each local terminal fused packet transports through a* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail packet, and every local handoff row consumes the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger together with the same* $\mathsf{Real}^{\uparrow}$ *seal-source row.*

Then the family pastes to one terminal fused packet exactly when the pairwise restrictions expose the same common-tail Real boundary. The pasted packet has precisely the visible rows

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

where the $\mathsf{StreamName}^{\uparrow}$ *rows are the given finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows are the endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are the common-tail classifier and regularity rows, and the* $\mathsf{Real}^{\uparrow}$ *row is the seal-source row. Its only support provenance is*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The exactness claim exports no quotient stream equality, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no countable-choice row, and no standard bridge row.*

Proof. First build the finite StreamName source surface. Apply Definition 126.551 to the certified windows and their pairwise and triple overlap data. This produces one pasted finite source whose support rows remain the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Next transport every local fused packet through the common-tail boundary. By Definition 301.186, the $\mathsf{RegSeqRat}^{\uparrow}$ common-tail classifier and regularity rows expose exactly the same Real seal-source boundary as the dyadic endpoint, radius, and distance ledger carried by the local handoff rows.

Now apply Definition 38.774. The common-tail Real boundary is therefore the cocycle condition for pasting the local $\mathsf{Real}^{\uparrow}$ seal-source rows across the finite cover, and the shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger prevents a second radius or distance coordinate from entering the seal.

Finally use Definition 126.624. It identifies agreement of the pairwise common-tail Real boundaries with agreement of the terminal fused StreamName packet on the four visible faces and the five support rows. Hence the local terminal fused packets paste to one terminal fused packet exactly in the stated case, while quotient stream equality, host sequence equality, chosen limits, completeness, metric-completeness data, countable choice, and bridge rows remain outside the boundary. □

Theorem 126.629 (StreamName terminal fusion cocycle section descent)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows satisfying the pasted-source hypotheses of Definition 126.628. Assume each local fused terminal packet is carried through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and regularity rows, and the same* $\mathsf{Real}^{\uparrow}$ *seal-source row. Then pairwise common-tail* $\mathsf{Real}^{\uparrow}$ *boundary agreement is equivalent to descent of the local fused terminal packets to one finite-request section.*

The descended section has exactly the visible rows consisting of the $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

It exports no quotient stream equality, no host sequence equality, no selected limit, no countable-choice row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, and no standard bridge row.*

Proof. For the descent direction, apply Definition 126.628. The pairwise common-tail $\mathsf{Real}^{\uparrow}$ boundary agreement is exactly the cocycle condition for the certified local terminal packets to paste to one fused terminal packet over the finite family.

Next read the pasted terminal packet as a finite request. Use Definition 126.577. The pullback exposes precisely the StreamName windows, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier and regularity rows, the Real seal-source row, and the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ support rows. Thus the pasted packet descends to one finite-request section with no further public coordinate.

For the reverse direction, start with such a finite-request section. The section pullback gives the same displayed rows on every local window, and Definition 301.186 identifies those rows with the common-tail $\mathsf{Real}^{\uparrow}$ boundary read by the RegSeqRat classifier and regularity data.

Finally apply Definition 38.774 and then Definition 126.624. The first theorem reads the shared boundary as the finite-cover seal pasting criterion, while the second reads it back as agreement of the local fused StreamName terminal packets on exactly the visible rows and the five support rows. The excluded quotient, host-sequence, selected-limit, choice, completeness, metric-completeness, and bridge rows therefore cannot be part of the descent datum. □

Theorem 126.630 (StreamName terminal fusion refinement invariance)**.** *Let* $\{\mathcal{W}_i\}_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows satisfying the hypotheses of Definition 126.629. Suppose* $\{\mathcal{R}_a\}_{a\in A}$ *is a certified finite refinement of that family whose restrictions preserve the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and regularity rows, the same* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the finite-request section descended from the refined family is the same section descended from the original family. Consequently the public terminal fused packet over the four faces

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

is unchanged by passing to the certified refinement and restricting back to the original cover.

The invariant packet exports no quotient stream equality, no host sequence equality, no selected limit, no countable-choice row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, and no standard bridge row.*

Proof. First compare each original window with its refined restrictions. Apply Definition 126.623. The preserved endpoint-radius-distance ledger, common-tail classifier and regularity rows, seal-source row, and displayed support rows make every restriction read the same terminal fused packet as the original window from which it is certified.

Next verify the cocycle on the refined cover. Use Definition 301.190 to identify the triple-overlap RegSeqRat common-tail rows, and then apply Definition 126.628. The pairwise common-tail $\mathsf{Real}^{\uparrow}$

boundary condition for the refined windows is therefore the same finite-cover cocycle condition as for the original family.

Now descend both finite covers. By Definition 126.629, the original family and the certified refinement each produce one finite-request section carrying exactly the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Finally restrict the refined section back along the refinement maps. The common-refinement stability already identifies every restricted local terminal packet with its original local packet, while the finite-cover cocycle exactness identifies the pasted packet by precisely those rows. Hence the two descended sections are the same finite-request section, and the public terminal fused packet is invariant under the certified refinement. Since all steps read only the visible four faces and the five support rows, quotient stream equality, host sequence equality, selected limits, choice, completeness, metric-completeness data, and bridge rows remain outside the interface. □

Theorem 126.631 (StreamName terminal fusion phase-exit consumer readback)**.** *Let $\mathcal{W}$ be an accepted fused StreamName terminal-window packet over*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

with support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

If a same-route RealCompletionExactBoundaryUp *face is supplied for the four roadmap faces of $\mathcal{W}$, then $\mathcal{W}$ determines the same five-face consumer readback as an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer: the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* RealCompletionExactBoundaryUp *local records, together with the supporting* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *rows.*

Conversely, any accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer displaying precisely those five local records and those support rows restricts along the four roadmap faces to the fused StreamName finite-window packet carried by $\mathcal{W}$.*

The comparison exports no quotient stream equality, no selected limit, no host completeness row, no countable-choice row, no metric-completeness row, no aggregate formalstatus row, no bridge token, and no fused completion witness.

Proof. For the forward direction, first apply Definition 126.626. It reads the accepted fused terminal-window packet through exactly the four roadmap faces and the displayed BHist/hsame/Cont/Pkg/NameCe support rows.

Next apply Definition 126.625. The same row family is the finite-request section surface, so supplying RealCompletionExactBoundaryUp on that same finite route adds the fifth local face without changing the four-face StreamName packet.

Finally apply Definition 1327.24. Its consumer theorem reads exactly the five local records and the same support rows, with the exact-boundary face constrained to the four roadmap faces. Thus the fused terminal packet determines the phase-exit consumer readback.

For the reverse direction, start with an accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ consumer displaying those five local records. The facewise readback theorem identifies the first four records as the dyadic, stream, regular-rational, and real faces on the same finite route as the exact-boundary face.

Discard the exact-boundary face and keep the displayed support rows. By source exhaustion and finite-request section exactness, those four roadmap faces and support rows are precisely the fused StreamName finite-window packet. The excluded quotient, selected-limit, host-completeness,

choice, metric-completeness, aggregate-status, bridge, and fused-witness rows therefore cannot supply another consumer coordinate. □

Theorem 126.632 (StreamName terminal fusion phase-exit refinement readback invariance)**.** *Let* $\{\mathcal{W}_i\}_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows satisfying the hypotheses of Definition 126.629, and let* $\{\mathcal{R}_a\}_{a\in A}$ *be a certified finite refinement preserving the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and regularity rows, the same* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the same support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Assume moreover that the original descended packet and the refined descended packet are supplied with a same-route $\mathsf{RealCompletionExactBoundaryUp}$ *face for their four roadmap faces. Then the accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer readback obtained from the refined fused terminal packet is the same five-face readback as the one obtained from the original fused terminal packet: the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealCompletionExactBoundaryUp}$ *local records, together with the supporting* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *rows.*

The comparison exports no quotient stream equality, no host sequence equality, no selected limit, no countable-choice row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no metric-completeness datum, no aggregate formalstatus row, no bridge token, and no fused completion witness.*

Proof. First apply Definition 126.630. The certified refinement preserves exactly the endpoint-radius-distance ledger, common-tail classifier and regularity rows, seal-source row, and displayed support rows used by the descent theorem. Hence the original cover and the refined cover descend to the same finite-request section and to the same public fused terminal packet over the four faces.

Next read the original presentation as a phase-exit consumer. Apply Definition 126.631 to the original fused packet and its same-route $\mathsf{RealCompletionExactBoundaryUp}$ face. The theorem returns precisely the five local records and the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ support rows.

Apply the same consumer-readback theorem to the refined presentation. The first step identifies the refined four-face fused packet with the original four-face fused packet, and the hypothesis supplies the same-route $\mathsf{RealCompletionExactBoundaryUp}$ face. Therefore the refined presentation reads back the same five local records and the same support rows.

The no-quotient, no-choice, no-host-completeness, no-metric-completeness, no-aggregate-status, no-bridge, and no-fused-witness exclusions are inherited from the refinement-invariance and phase-exit consumer-readback theorems. Since both applications use only the visible four roadmap faces, the exact boundary face on that route, and the five displayed support rows, no additional consumer coordinate is available. □

Theorem 126.633 (StreamName terminal fusion universal consumer normal form)**.** *Let* $\mathcal{W}$ *be an accepted terminal fused StreamName packet supplied with a same-route* $\mathsf{RealCompletionExactBoundaryUp}$ *face, so that it is readable as an accepted terminal* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer. Every downstream consumer of this terminal Phase Real and Completion packet that preserves BEDC-visible public rows factors through the four roadmap faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

on that route, the same-route $\mathsf{RealCompletionExactBoundaryUp}$ *face, and the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, once the four roadmap faces, the exact-boundary face on that route, and the displayed support rows are fixed, the consumer readback is in normal form: two presentations with the same displayed rows give the same downstream consumer readback.

This normal form exports no quotient stream equality, no host sequence equality, no selected limit, no ambient completeness row, no metric-completeness datum, no countable-choice row, no bridge token, no aggregate formalstatus row, and no fused completion witness.

Proof. First reduce the terminal packet to its source rows. By Definition 126.626, an accepted terminal fused StreamName packet is exhausted by the four roadmap faces and the displayed BHist/hsame/Cont/Pkg/NameCert support rows. Thus a BEDC-visible downstream consumer cannot read an additional StreamName-side coordinate before the phase-exit face is attached.

Next attach the same-route exact-boundary face and read the result as a phase-exit consumer. Apply Definition 126.631. The consumer readback consists of the four roadmap records, the RealCompletionExactBoundaryUp face constrained to that route, and the same support rows; the excluded quotient, limit, choice, completeness, aggregate status, bridge, and fused-witness rows are not part of the exported interface.

It remains to check independence from presentation. If two terminal presentations display the same four roadmap faces, the same exact-boundary face on that route, and the same support rows, then apply Definition 126.632 to any certified common refinement between them. The refined and original presentations give the same five-face phase-exit consumer readback, hence the same downstream readback for every row-preserving consumer.

Finally identify the roadmap substrate. By Definition 126.613, the only roadmap faces available to the terminal route are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The four-face route supplies the roadmap substrate, while the phase-exit consumer readback retains the RealCompletionExactBoundaryUp face as a constrained fifth local record. Combining that gate with source exhaustion, phase-exit readback, and refinement readback invariance gives the stated universal normal form. □

Theorem 126.634 (StreamName terminal fusion universal consumer reflection)**.** *Let $\mathcal{W}$ and $\mathcal{W}'$ be accepted terminal fused StreamName packets whose four roadmap faces are supplied with same-route* RealCompletionExactBoundaryUp *faces, and suppose their row-preserving terminal* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer readbacks agree. Then the displayed terminal rows of the two packets agree: the four roadmap faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

the same-route RealCompletionExactBoundaryUp *face, and the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

are the same on both terminal presentations.

Thus consumer equivalence for accepted terminal fused StreamName packets is reflected exactly by the BEDC-visible four-face terminal surface and its exact-boundary support. Quotient stream equality, selected limits, host sequence equality, ambient completeness, metric-completeness, countable choice, aggregate formalstatus, bridge token, and fused completion witness cannot witness consumer equality.

Proof. First put each terminal packet in universal consumer normal form. Apply Definition 126.633 to $\mathcal{W}$ and to $\mathcal{W}'$. Every row-preserving terminal consumer of either packet factors through its four roadmap faces, same-route RealCompletionExactBoundaryUp face, and displayed BHist/hsame/Cont/Pkg/NameCert rows.

Next read both normal forms through the phase-exit interface. By Definition 126.631, the terminal $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ consumer readback is exactly the five local faces together with the displayed support rows. Since the two consumer readbacks agree by hypothesis, the five local records and support rows named by that theorem agree.

It remains to rule out hidden witnesses. Source exhaustion, Definition 126.626, identifies the StreamName terminal packet before the exact-boundary attachment with only the four roadmap faces and the displayed support rows.

Finally apply the no-choice route factorization, Definition 126.522. It removes quotient stream equality, selected limits, host sequence equality, ambient completeness, metric-completeness, countable choice, aggregate formalstatus, bridge token, and fused completion witness from the terminal route. Therefore no excluded coordinate can account for the common consumer readback, and the agreement reflects back exactly to the displayed terminal rows. □

Theorem 126.635 (StreamName terminal fusion four-face exit status equivalence)**.** *Let $\mathcal{W}$ and $\mathcal{W}'$ be accepted terminal fused StreamName packets whose four roadmap faces are supplied with same-route* $\mathsf{RealCompletionExactBoundaryUp}$ *faces. Row-preserving terminal* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer equivalence between $\mathcal{W}$ and $\mathcal{W}'$ holds if and only if their terminal presentations agree on the four roadmap faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

on the same-route exact-boundary face, and on the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Thus the terminal exit status for the four-object Real-completion route is exactly the equality class of these BEDC-visible rows.

The equivalence has no further coordinates: aggregate formalstatus, bridge token, quotient stream equality, selected limits, host sequence equality, ambient completeness, metric completeness, countable choice, and fused completion witnesses are not part of the exit-status comparison.

Proof. Assume first that the displayed terminal rows agree. Put each packet in normal form by Definition 126.633. Since the two packets have the same four roadmap faces, the same same-route $\mathsf{RealCompletionExactBoundaryUp}$ face, and the same $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ support rows, the normal forms determine the same row-preserving terminal $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ consumer readback.

For the reverse direction, assume that the row-preserving terminal consumer readbacks agree. Apply Definition 126.634. The reflection theorem reads the common consumer back to equality of the four roadmap faces, the same-route exact-boundary face, and the displayed support rows.

It remains to identify the allowed surface. Source exhaustion, Definition 126.626, says that the accepted terminal fused StreamName packet supplies no StreamName-side coordinate beyond the four roadmap faces and the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ rows. The four-face exit gate, Definition 126.613, fixes the roadmap surface as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while Definition 126.631 records the same-route $\mathsf{RealCompletionExactBoundaryUp}$ face as the constrained local readback for that route.

Finally apply the no-choice route factorization, Definition 126.522. The aggregate formalstatus, bridge token, quotient stream equality, selected limits, host sequence equality, ambient completeness, metric completeness, countable choice, and fused completion witness are not consumer coordinates on this terminal route. Therefore the two implications above form the exact four-face exit status equivalence. □

Theorem 126.636 (StreamName terminal fusion support minimality)**.** *Let $\mathcal{W}$ be an accepted terminal fused StreamName packet. Deleting any one roadmap face*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

or any one displayed support row

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

destroys the accepted terminal packet. Equivalently, no proper subpacket of these four faces and five support rows carries the same terminal $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer readback.*

There is no replacement row for the missing support: quotient stream equality, selected limits, countable choice, a metric-completeness row, and bridge rows are outside the accepted terminal surface.

Proof. The terminal-window fusion theorem Definition 126.622 constructs the accepted packet from exactly the four Real-completion faces and the displayed support rows. Thus each listed coordinate is part of the packet admitted by the terminal fusion surface.

Finite-request section exactness, Definition 126.625, shows that each terminal request is read through the displayed finite section, so deleting one face or support row removes a row needed by some accepted finite request. Source exhaustion, Definition 126.626, then rules out any additional StreamName-side coordinate from filling the gap.

Finally apply the four-face exit status equivalence Definition 126.635. The terminal consumer status is exactly the equality class of the four roadmap faces, the same-route exact-boundary face, and the displayed support rows. A proper subpacket therefore cannot have the same accepted terminal status, and the excluded quotient, choice, metric-completeness, and bridge coordinates cannot provide a substitute. □

Theorem 126.637 (StreamName terminal window fusion nonescape)**.** *Let $\mathcal{W}$ be an accepted fused terminal* $\mathsf{StreamName}^{\uparrow}$ *window consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *on the L10 four-face route. Then every such consumer preserves the displayed finite request section, the reindex gluing row, the terminal fusion row, and the support-minimality row. Its readable surface is exactly*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

No consumer of the fused terminal window obtains a selected limit, quotient stream equality, host sequence equality, countable-choice schedule, metric-completeness datum, bridge row, fused completion witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness field from the fusion packet.*

Proof. Begin with the finite request section. Apply Definition 126.625. Every terminal request read by the RegSeqRat and Real consumers is already a displayed section of the fused StreamName packet.

Next apply the reindex and fusion rows. The terminal-window fusion theorem Definition 126.622 keeps the reindexed requests on the same four roadmap faces and the same displayed support, while the universal consumer readback of Definition 126.631 identifies the RegSeqRat and Real-facing reads with that fused terminal surface.

Now use Definition 126.636. Since removing any roadmap face or support row destroys the accepted terminal packet, every consumer that still reads the fused window must preserve those rows rather than replacing them by an external coordinate.

Finally combine source exhaustion with the four-face exit status equivalence. The exhaustion theorem leaves no additional StreamName-side coordinate, and Definition 126.635 identifies the terminal consumer status with the same four faces and support rows. Thus selected limits, quotient stream equality, host sequence equality, choice schedules, metric-completeness data, bridge rows, fused completion witnesses, and ambient Real completeness fields cannot escape through the terminal fusion packet. □

Theorem 126.638 (StreamName RegSeqRat Real budget cocycle)**.** *Let $\mathcal{W}$ be an accepted terminal finite* $\mathsf{StreamName}^{\uparrow}$ *window on the L10 four-face route. Suppose its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and downstream* $\mathsf{Real}^{\uparrow}$ *seal are read through the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the terminal window, the RegSeqRat readback, and the Real seal form one budget cocycle: after terminal fusion, none of the three faces can be independently reindexed away from the shared budget row.

The cocycle exposes no quotient stream equality, selected limit, host sequence equality, countable-choice schedule, metric-completeness datum, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. Begin with Definition 126.622. It constructs the terminal packet from the StreamName finite request window, the DyadicRatCore tolerance ledger, the RegSeqRat readback, the Real seal, and the displayed support rows.

Next apply Definition 126.623 and Definition 126.631. The common refinement and terminal consumer readback keep the RegSeqRat-facing and Real-facing reads on the same fused terminal surface.

Use Definition 126.636. Since deleting the budget-facing support or any of the four roadmap faces destroys the accepted packet, no separate reindexing row can replace the shared dyadic budget after the fusion is read.

Finally apply Definition 126.637. The terminal consumer has no selected limit, quotient stream equality, host sequence equality, choice schedule, metric-completeness datum, bridge row, or ambient Real-completeness coordinate, so the shared budget row is the only readable route through the StreamName, RegSeqRat, and Real faces. □

Theorem 126.639 (StreamName finite-window reindex determinacy)**.** *Let W_0 and W_1 be accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets. Suppose their reindex maps are* hsame*-compatible, their certified endpoint rows are identical, and their displayed* BHist*,* Cont*,* Pkg*, and* NameCert *support rows transport componentwise across the same reindex route. Then W_0 and W_1 determine the same public finite observation section.*

The determinacy statement is internal to the displayed StreamName packet. It does not use quotient stream equality, host sequence equality, selected limits, countable choice, or a completion row.

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameWindowReindexDeterminacy`

Proof. First read both packets through the transported finite-window endpoint package of Definition 126.32. The shared endpoint rows and the hsame-compatible reindex route place both packets on the same finite endpoint surface.

Next apply successor projection determinacy, Definition 126.299. It prevents the reindex route from creating two public successor projections once endpoint rows and support rows have been identified.

Finally use terminal window fusion, Definition 126.622. The fused terminal packet has no extra coordinate beyond the finite window, endpoint data, and displayed support rows, so both accepted packets export the same public finite observation section. □

Theorem 126.640 (StreamName public reindexing identity law)**.** *For every public* StreamName$^{\uparrow}$ *finite-window packet over a* ProbeBundle*, the certified identity reindexer preserves the displayed* RatStreamNameFiniteWindowClassifier*,* UnaryHistory *observation rows,* InBundle *membership rows, pointwise rational classifier rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The resulting public finite observation section is the same section read through the identity route.*

This identity law is a statement about the public finite-window packet only. It does not introduce quotient stream equality, host sequence equality, selected limits, countable choice, a completion row, or a standard bridge row.

Proof. Read the packet through the public finite-window interface and the public reindexing obligation of Definition 126.317. On an identity reindexer, every certified UnaryHistory observation remains the displayed observation, and every InBundle row is read from the same ProbeBundle.

The pointwise rational classifier, hsame comparison, Cont route, Pkg provenance, and local NameCert row therefore reappear as the same displayed rows. No extra endpoint, quotient representative, or host-index coordinate is available in the public obligation surface.

Apply Definition 126.639 to the original packet and to the identity-reindexed packet. The hypotheses are exactly the shared endpoint rows, the identity hsame route, and the componentwise agreement of the displayed support rows, so the exported public finite observation section is unchanged. □

Theorem 126.641 (StreamName public reindexing associative law)**.** *Let r and s be composable certified unary reindexers displayed over public* StreamName$^{\uparrow}$ *finite-window packets. The two-step packet obtained by reindexing along r and then along s exposes the same public finite-window section as the single packet obtained by the composite reindexer. The common section has the same* RatStreamNameFiniteWindowClassifier*,* UnaryHistory *ledger,* InBundle *transport, pointwise rational classifier rows,* hsame *comparisons, composed* Cont *route, displayed* Pkg *provenance, and local* NameCert *rows.*

The law is confined to the finite public surface. It does not claim quotient stream equality, host sequence equality, selected limits, countable choice, completion data, or a standard bridge row.

Proof. First apply the public reindexing obligation of Definition 126.317 to the r leg and then to the s leg. The two applications list only the certified UnaryHistory observations, InBundle membership rows, pointwise rational classifier rows, hsame comparisons, Cont routes, Pkg provenance, and local naming rows displayed by the finite packets.

Next use the finite-window reindexing composition theorem, Definition 126.304. It identifies the two restricted finite BHist packets with the packet obtained by restricting once along the composite reindexer, with the classifier, endpoint, hsame, and finite ledger rows merely repacked over the target ProbeBundle.

Finally apply Definition 126.639 to the two-step packet and the composite packet. Their reindex maps are hsame-compatible, their endpoint rows agree, and their displayed BHist, Cont, Pkg, and NameCert rows transport componentwise, so the exported public finite observation section is the same. □

Theorem 126.642 (StreamName reindex terminal coverage)**.** *A certified public* StreamName$^{\uparrow}$ *finite-window reindex route preserves exactly the terminal four-face L10 exit route. Namely, if*

an accepted reindex route transports the public finite window together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows, then the transported exit packet exposes the same* StreamName$^{\uparrow}$ *window, the same* DyadicRatCore$^{\uparrow}$ *ledger, the same* RegSeqRat$^{\uparrow}$ *readback, and the same* Real$^{\uparrow}$ *seal as its terminal Phase Real and Completion route.*

Conversely, any terminal four-face route transported by the identity certified reindexer or by composable certified reindexers reads the same public finite observation section. The coverage is confined to the finite route and excludes quotient stream equality, host sequence equality, selected limits, countable choice, ambient Real$^{\uparrow}$ *completeness, metric completeness, and standard bridge rows.*

Lean4 ✓ `BEDC.Derived.StreamNameUp.StreamnameReindexTerminalCoverage`

Proof. Start with the public finite-window section. The determinacy theorem, Definition 126.639, says that matching endpoint rows, hsame-compatible reindex maps, and componentwise transported BHist, Cont, Pkg, and NameCert rows determine one public finite observation section.

For the identity route, apply Definition 126.640. It keeps the displayed finite window, unary-history ledger, bundle membership, rational classifier, hsame, Cont, Pkg, and local NameCert rows unchanged, so the terminal packet is read from the same public section.

For composable certified routes, apply Definition 126.641. The two-step and composite reindexings expose the same finite section, hence they present the same support rows to the exit boundary.

Finally use Definition 126.449 and Definition 126.450. Terminality fixes the four faces once the StreamName window, DyadicRatCore ledger, RegSeqRat readback, RealUp seal, and support rows agree, while readback says every terminal exit read factors through exactly those rows. Thus transported identity and composable routes cover precisely the finite four-face exit route, with no additional quotient, host, limit, choice, completeness, or bridge coordinate. □

Theorem 126.643 (StreamName finite-window reindex stability)**.** *Let R be an accepted public* StreamName$^{\uparrow}$ *finite-window reindex route. Suppose R preserves the endpoint rows, unary observation rows,* InBundle *membership rows, pointwise rational classifier rows, componentwise* hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of the finite packet. Then R preserves the public finite observation section presented to downstream* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumers.*

The preserved section is exactly the displayed StreamName window with its endpoint, bundle, rational classifier, transport, replay, provenance, and naming rows. It exposes no quotient stream equality, host sequence equality, selected limit, countable choice, ambient Real completeness, metric-completeness data, or standard bridge row.

Proof. Apply finite-window reindex determinacy, Definition 126.639. The preserved endpoint rows, unary observations, bundle membership, pointwise rational classifiers, and support rows determine one public finite observation section.

Use the public identity law, Definition 126.640, to identify unchanged windows with the same displayed section. This accounts for trivial reindex legs without adding host sequence equality.

Next apply the public associative law, Definition 126.641. Composable certified reindex routes therefore present the same finite section after repacking their displayed support rows.

Finally use terminal coverage, Definition 126.642. The downstream RegSeqRat$^{\uparrow}$ readback and Real$^{\uparrow}$ seal consume only that public finite section and its support rows, so the excluded quotient, choice, limit, completeness, metric, and bridge coordinates cannot alter the consumer-facing section. □

Theorem 126.644 (StreamName public reindexing section cancellation)**.** *Let R_0 and R_1 be certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes displayed over the same* ProbeBundle*. Suppose they expose the same public finite observation section, the same endpoint rows, and the same displayed* BHist*, and suppose their componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows agree over that section. Then every terminal Phase Real and Completion consumer reads the same four-face route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *from R_0 and R_1.*

Thus the consumer cannot distinguish the two routes by reindexer shape, host sequence equality, quotient stream equality, selected limits, countable choice, ambient completeness, metric completeness, or a standard bridge row. The visible data are exactly the displayed finite section, endpoint rows, and support rows.

Proof. First apply Definition 126.639 to the two certified routes. The shared endpoint rows, the common displayed ProbeBundle, the displayed BHist, and the componentwise hsame, Cont, Pkg, and local NameCert support rows identify the public finite observation section read by the two packets.

The identity and composite presentations do not add observable data. Apply Definition 126.640 to any identity leg and Definition 126.641 to any composite presentation. Both laws reduce the public presentation back to the same finite section with the same displayed support rows.

Finally apply Definition 126.642 and Definition 126.622. Terminal coverage says every accepted reindex route reaches the same four-face exit surface once the finite section and support rows agree, while terminal window fusion carries that agreement through the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal. Hence the terminal consumer reads one route, independent of the certified reindexer shape and of the excluded quotient, host, limit, choice, completeness, or bridge coordinates. □

Theorem 126.645 (StreamName public reindexing cancellation boundary)**.** *Let R_0 and R_1 be certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes displayed over a finite* ProbeBundle*, and suppose both feed the same terminal Phase Real and Completion four-face route. Then any public distinction between R_0 and R_1 must appear in one of the displayed finite observation section rows:*

$$\mathsf{RatStreamNameFiniteWindowClassifier},\quad \mathsf{ProbeBundle},\quad \mathsf{InBundle},\quad \mathsf{UnaryHistory},$$

the pointwise $\mathsf{Rat}^{\uparrow}$ *classifier rows, the endpoint* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance rows, or the local* NameCert *rows.*

If all those rows agree, then downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers receive the same terminal route through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The boundary contains no quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or standard bridge row.*

Proof. First use Definition 126.639. The finite window classifier, bundle membership, unary-history display, endpoint hsame data, displayed Cont routes, Pkg provenance, and local NameCert rows are exactly the public rows from which a certified reindex packet is read.

Next apply the identity and associativity laws, Definition 126.640 and Definition 126.641. Identity legs and composite presentations cannot create an additional visible distinction beyond the same finite observation rows.

Now use Definition 126.642. Every certified public reindex route that reaches the terminal Phase Real and Completion surface is read through the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no extra quotient, host, selector, choice, completeness, metric, or bridge coordinate.

Finally apply Definition 126.644. Once the listed finite rows agree, section cancellation identifies the terminal read of R_0 and R_1. The terminal four-face readback used by Definition 126.646 then passes the same route to downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, so any remaining public distinction would have had to occur in the displayed finite rows already listed. □

Theorem 126.646 (StreamName reindex finite-cover terminal descent)**.** *Let $\{\mathcal{W}_i\}_{i\in I}$ be a finite family of certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packets, each equipped with a certified public reindex route to the terminal Phase Real and Completion surface. Suppose that on every overlap the reindexed packets expose the same public finite observation section, the same endpoint rows, the same displayed* BHist*, and the same componentwise* hsame, Cont, Pkg*, and local* $\mathsf{NameCert}$ *support rows. Suppose also that the associated local terminal packets satisfy the common-tail* $\mathsf{Real}^{\uparrow}$ *boundary agreement hypotheses of Definition 126.628.*

Then the reindexed family descends to one finite-request section on the terminal four-face route

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The descended section has only the displayed finite StreamName windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It does not assert quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. First compare the overlap reads after reindexing. Apply Definition 126.644 to each pair of certified overlap routes. The shared finite observation section, endpoint rows, displayed BHist, and componentwise $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ support rows identify the terminal read on the four visible faces.

Next keep the identified reads inside the public terminal surface. By Definition 126.642, every accepted reindex route exposes exactly the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal, with no additional quotient, host, selector, choice, completeness, metric, or bridge coordinate.

Now use the assumed common-tail $\mathsf{Real}^{\uparrow}$ boundary agreement. Apply Definition 126.628 to read that boundary agreement as the finite-cover cocycle condition for the local terminal fused packets on the same four-face surface.

Finally apply Definition 126.629. The cocycle condition descends the local terminal packets to one finite-request section, and the descent theorem lists exactly the displayed StreamName, DyadicRatCore, RegSeqRat, Real, and support rows. Thus the reindex data contribute only the public finite sections already cancelled above. □

Theorem 126.647 (StreamName reindex four-face exit naturality)**.** *Let R be a support-row-preserving certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex route, displayed over a finite* $\mathsf{ProbeBundle}$*, from a local StreamName window to the terminal Phase Real and Completion exit surface. Suppose*

the route preserves the public finite observation section, the endpoint rows, the displayed BHist*, and the componentwise support rows*

$$\mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then transport along R is natural at the terminal exit surface: the source read and the transported read have the same visible four-face Phase Real and Completion surface, namely

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

and the only support rows visible to every terminal consumer are

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

No additional coordinate is transported or identified. In particular, the naturality statement does not assert quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or any standard bridge row.*

Proof. First read R through the public finite observation section. By Definition 126.639, preservation of the finite section, endpoint rows, displayed BHist, and componentwise hsame/Cont/Pkg/NameCert support rows fixes the public data seen before and after the certified reindex route.

Next place that fixed public section on the terminal route. Apply Definition 126.642. The route reaches exactly the four visible faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the listed support rows and with no quotient, host, selector, choice, completeness, metric, or bridge coordinate added to the surface.

The certified route shape itself is immaterial to the terminal consumer. Use Definition 126.644 to compare any two presentations of the same preserved section; section cancellation keeps the terminal read invariant under reindexer shape.

Finally apply Definition 126.646 to the singleton finite family determined by R. Finite-family descent says the terminal section remains a section on the same four-face surface with exactly the same support rows. Hence support-row-preserving certified reindex transport is natural for the terminal Phase Real and Completion exit surface, and it carries none of the excluded quotient, host, limit, choice, completeness, metric, or bridge data. □

Theorem 126.648 (StreamName terminal reindex four-face pullback)**.** *Let R_0 and R_1 be certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes into the terminal Phase Real and Completion exit boundary. Assume both routes preserve the same public finite observation section, the same endpoint rows, the same displayed* BHist*, and the same componentwise support rows*

$$\mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Assume also that the local NameCert *support used by each route is the same finite support read over the preserved section. Then R_0 and R_1 determine the same terminal pullback over the four visible faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Conversely, every terminal pullback produced by such a support-row-preserving reindex route is read from exactly the public finite observation section, the endpoint rows, displayed BHist*, componentwise* hsame/Cont/Pkg *rows, and local* NameCert *support listed above.*

The assertion is only about this four-face terminal pullback. It does not assert quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. First apply Definition 126.647 separately to R_0 and R_1. Each route has the same transported terminal exit surface, and that surface has exactly the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with the listed support rows.

Next compare the two route presentations at the public section. By Definition 126.644, the common finite observation section, endpoint rows, displayed BHist, and componentwise $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ support cancel the reindexer shape. Thus the two routes expose the same terminal read before any four-face packaging is imposed.

Now keep that read inside the terminal surface. Use Definition 126.642 to see that no fifth face or hidden support coordinate is present: the route contributes precisely the StreamName, DyadicRatCore, RegSeqRat, Real, and local support rows named in the statement.

Finally apply Definition 126.646 to the singleton finite family determined by either route. Singleton descent identifies the finite-family terminal section with the same four-face terminal pullback, so R_0 and R_1 give one pullback over the shared rows.

For the converse direction, start with any terminal pullback produced by a support-row-preserving reindex route of the stated kind. The same terminal coverage theorem lists all rows from which it can be read, and the singleton descent theorem repacks those rows without adding quotient, host, selector, choice, completeness, metric, or bridge data. Hence the pullback is read from exactly the public section, endpoint rows, displayed BHist, componentwise $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}$ rows, and local $\mathsf{NameCert}$ support. □

Theorem 126.649 (StreamName reindex tail-window terminal equivalence)**.** *Let $\mathcal{F}$ be a finite family of accepted current Phase Real and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections, all displayed over the same public endpoints and the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *spine. Let $\mathcal{W}$ be a finite chain of* $\mathsf{StreamName}^{\uparrow}$ *tail-window coverage packets over those endpoints and that spine, and let R be a support-row-preserving certified public* $\mathsf{StreamName}^{\uparrow}$ *reindex route for the family. Assume that R preserves the public finite observation section, the endpoint rows, the displayed* BHist*, and the componentwise support rows*

$$\mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then reindexing along R preserves the equivalence between the finite-section descent-gluing public reflection surface of $\mathcal{F}$ and the tail-window terminal read of $\mathcal{W}$. Both sides are identified by the same four-face terminal route

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

with the same carrier rows, the same local $\mathsf{NameCert}$ *support, and the same classifier verdict.*

This equivalence is confined to the displayed public terminal surface. It does not assert quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or standard bridge rows.*

Proof. First place the certified reindex route on the public finite observation section. By Definition 126.648, the reindexed route is read through exactly the same finite section, endpoints, displayed BHist, componentwise $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}$ rows, and local $\mathsf{NameCert}$ support as the original route.

Next descend the reindexed finite family. Apply Definition 126.646 to the family carried by R. The descent-gluing surface remains a finite terminal public reflection surface over the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, with no extra quotient, host, selector, choice, completeness, metric, or bridge coordinate.

Now compare that descended surface with the tail-window terminal read. Use Definition 126.595 for the finite request sections and tail-window coverage packets over the shared public endpoints and accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine. This identifies the descent-gluing public reflection surface with the terminal read of the tail-window chain, preserving the displayed carrier rows, local $\mathsf{NameCert}$ support, and classifier verdict.

Finally keep the comparison inside the four visible faces. By Definition 126.642, the certified public reindex route has no access to a fifth coordinate: the only terminal faces are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the excluded quotient, host, limit, choice, completeness, metric, and bridge data remain outside the assertion. Hence certified reindexing preserves the tail-window terminal equivalence for the finite descent-gluing surface. □

Theorem 126.650 (StreamName public reindex zigzag normal form)**.** *Let Z be a finite zigzag of support-row-preserving certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes whose visible endpoints are the same public finite observation sections and whose spine row is the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. Suppose every leg of Z preserves the displayed public finite section, endpoint rows, displayed* BHist*, and the componentwise support rows*

$$\mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then Z determines one terminal four-face packet over

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with exactly those displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows.*

Conversely, any two displayed public finite-window sections over the same public endpoints and accepted $\mathsf{RegSeqRat}^{\uparrow}$ *spine that determine the same terminal four-face pullback are connected by a finite zigzag of support-row-preserving certified public reindex routes. The assertion is confined to the displayed public terminal surface. It excludes quotient stream equality, host sequence equality, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, standard bridge rows, and any aggregate status row.*

Proof. Start with the forward direction. Insert certified identity legs where a zigzag endpoint is repeated and compose adjacent forward-facing legs whenever their endpoints match. The identity insertion is harmless by Definition 126.640, and the collapse of a two-leg presentation to its composite is harmless by Definition 126.641.

Whenever two neighboring presentations expose the same public finite section, use Definition 126.644. This removes the reindexer shape from the terminal consumer and leaves only the displayed finite section, endpoint rows, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support.

The remaining finite family of displayed sections is then descended by Definition 126.646. The descent surface is a finite terminal public reflection surface with no added quotient, host, selector, choice, completeness, metric, bridge, or aggregate coordinate.

Finally apply Definition 126.648 and Definition 126.649. The first theorem reads every normalized reindex presentation through the same four visible faces, and the second identifies the finite-section descent surface with the tail-window terminal read over the shared accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine. Hence the whole zigzag determines one terminal four-face packet with the displayed support rows named in the statement.

For the converse direction, begin with two displayed public finite-window sections that have the same terminal four-face pullback. Read both sections through Definition 126.648; the common pullback lists exactly the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and the displayed support rows.

Use Definition 126.649 to place both sections on the same tail-window terminal read, then apply Definition 126.646 to repack that read as a finite public family. Identity legs connect each section to its displayed occurrence in the family, associative composition joins consecutive legs, and section cancellation identifies parallel displayed sections. These legs form the required support-row-preserving finite zigzag. □

Theorem 126.651 (StreamName reindex support-row necessity)**.** *A certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex route feeds the terminal four-face Phase Real and Completion route if and only if the same public finite observation section displays the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the shared support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Here hsame *is componentwise,* Cont *is displayed on the same preserved finite section,* Pkg *records the route provenance visible to the terminal consumer, and* NameCert *is the local certificate support for that section.*

No quotient stream equality, host sequence equality, selected limit, countable choice datum, ambient $\mathsf{Real}^{\uparrow}$ *completeness assumption, metric-completeness assumption, or standard bridge row can replace any of these displayed support rows in the terminal route.*

Proof. For the forward direction, start with a certified public $\mathsf{StreamName}^{\uparrow}$ finite-window reindex route feeding the terminal four-face surface. Apply Definition 126.642. The coverage theorem restricts every such terminal read to the four visible faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Now compare the certified route with its terminal pullback presentation. Use Definition 126.648. The pullback inventory says that the terminal route is read from exactly the public finite observation section, endpoint rows, displayed BHist, componentwise hsame/Cont/Pkg rows, and local NameCert support. Thus feeding the terminal four-face route forces the displayed window, ledger, readback, seal, and support rows named in the statement.

For the reverse direction, assume the public finite observation section already displays the four faces and the listed support rows. Apply Definition 126.647 to transport the section along the certified public reindex route without changing the visible terminal surface.

Then apply Definition 126.650. The normal form theorem packs any finite support-row-preserving presentation over the same public endpoints and accepted $\mathsf{RegSeqRat}^{\uparrow}$ spine into one terminal four-face packet. Hence the displayed support rows are sufficient for the certified route to feed the terminal Phase Real and Completion route.

Finally rule out substitutions. By Definition 126.645, cancellation is confined to the preserved public section and its displayed support rows, while the terminal pullback inventory just cited lists no quotient stream, host equality, selected-limit, choice, ambient-completeness, metric-completeness, or bridge coordinate. None of those external rows can replace a required support row in the four-face route. □

Theorem 126.652 (StreamName reindex bundle naturality)**.** *Let* B *be a* ProbeBundle *indexing a public* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet, and let* $B' \subseteq B$ *be a displayed sub-bundle with its* InBundle *rows. Any certified reindex route on* B *that preserves* UnaryHistory *observation rows,* InBundle *membership rows, pointwise rational classifier rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *support commutes with restriction to* B'*.*

Equivalently, restricting the public finite observation section to B' and then reindexing produces the same displayed finite section as reindexing on B and then restricting to B'. The common section exposes exactly the same $\mathsf{StreamName}^{\uparrow}$ *windows, pointwise rational classifier rows, support rows, and NameCert provenance, and exports no quotient stream equality, host sequence equality, selected limit, countable choice datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness assumption, metric-completeness assumption, or standard bridge row.*

Proof. First apply the finite-window reindex determinacy already displayed in this chapter. Since the route preserves UnaryHistory observations, InBundle rows, pointwise classifiers, and the structural support rows, both composites are certified presentations of the same public finite observation section over the same endpoints.

Next use the public identity and associative laws for reindexing. Identity legs identify the displayed restriction of B to B', and associativity compares the route that restricts before reindexing with the route that restricts after reindexing as two factorizations of one certified finite section.

Finally apply Definition 126.651. The terminal section can use only the displayed window, classifier, hsame, Cont, Pkg, and NameCert rows. Thus the two restricted packets have the same public finite observation section, and no off-table quotient, selector, choice, completeness, metric, or bridge row can distinguish them. □

Theorem 126.653 (StreamName terminal four-face support minimality)**.** *Let U be an accepted current Phase Real and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Then the public terminal surface of U is accepted exactly through the four visible faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *together with the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows.*

Moreover, this support is minimal for the accepted public verdict: deleting any one of the four faces, or deleting any one of the displayed support rows, prevents the resulting packet from presenting the same carrier verdict and the same classifier verdict to Real-facing or completion-facing consumers. The minimality claim uses no quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus row, or bridge row.*

Proof. First build the public classifier surface. Apply Definition 126.598 to the accepted terminal request section and its finite cover, then compare any two public presentations with Definition 126.599. These results place the carrier and classifier verdicts on the same visible StreamName, DyadicRatCore, RegSeqRat, and Real faces over the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next lock the StreamName side to the dyadic, regular-rational, and Real sides. The theorem Definition 126.589 says that a Real-facing or completion-facing read of the terminal request section can use only the finite StreamName window, the DyadicRatCore ledger, the RegSeqRat readback, the Real seal boundary, and the same support rows. Thus a packet with one visible face or one support row removed is no longer the packet seen by those consumers.

Now read the same packet from the RegSeqRat terminal boundary. By Definition 301.239, terminal acceptance is equivalent to the presence of exactly the four displayed faces with the shared support rows. By Definition 301.237, any accepted consumer with the same public verdict has the same finite RegSeqRat source. Hence removing a face or support row removes the witness needed to identify the source that presents that verdict.

Finally pass through the Real dependency lattice. The exit totality theorem Definition 38.468 sends every current Phase Real and Completion terminal read through the ordered four-face route

and the displayed support rows. Combining this totality with the classifier reflection and source-uniqueness steps gives the stated minimality: the public carrier and classifier verdicts survive exactly on the four-face support surface and not on any shorter packet.

The exclusion list follows from the same route description. None of the cited surfaces contains a coordinate for quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate formalstatus, or bridge rows, so those objects cannot replace a missing visible face or support row. □

Theorem 126.654 (StreamName terminal support defect localization)**.** *Let U be an accepted current Phase Real and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Suppose a public terminal presentation of U fails to preserve at least one of the carrier verdict, the classifier verdict, or the consumer-facing terminal route supplied by Definition 126.653. Then the failure is witnessed on one of the four visible faces* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, or on one of the displayed support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Consequently, quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, aggregate formalstatus, and bridge rows cannot be the carrier of such a defect.*

Proof. The minimality theorem Definition 126.653 identifies the only support surface that presents the accepted carrier verdict and classifier verdict to Real-facing and completion-facing consumers. Therefore any loss of one of those verdicts, or of the displayed terminal route, must remove something from that support surface.

The readback theorem Definition 126.450 sends the public exit of an accepted terminal request through the same StreamName, DyadicRatCore, RegSeqRat, and Real faces. The Real-side localization theorem Definition 38.838 says that an L10 exit failure is detected on a displayed face or support row rather than on a hidden quotient, limit, choice, completeness, aggregate-status, or bridge coordinate.

Combining these two route descriptions with the two-axis roadmap discipline gives the advertised referee check. The theory axis records which face or support row carries the terminal evidence, while the verification axis records whether that row has been checked; neither axis turns the excluded quotient, limit, completeness, choice, aggregate formalstatus, or bridge rows into support for the StreamName terminal verdict. □

Theorem 126.655 (StreamName terminal support defect pullback)**.** *Let U and V be accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose a consumer-visible terminal presentation detects a carrier defect, a classifier defect, or a terminal route defect on one of the visible faces* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, or on one displayed support row* BHist*, componentwise* hsame*,* Cont*,* Pkg*, or local* NameCert*. Then the defect pulls back to a finite request-section mismatch: it is witnessed by the finite-section pullback, by the Real-seal lock, by window reindex determinacy, or by the terminal-window fusion source and reindexing compatibility surfaces.*

Consequently excluded coordinates such as quotient stream equality, host sequence equality, selected limits, countable choice, metric completeness, aggregate formalstatus, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness cannot serve as a defect carrier and cannot serve as a repair coordinate for the terminal request section consumed by the L10 Real route.*

Proof. First localize the visible failure. Apply Definition 126.654. The carrier, classifier, or route defect is already forced onto one of the four visible faces or onto one displayed BHist, hsame, Cont, Pkg, or NameCert support row.

Now pass from the terminal presentation to the finite request section. Apply Definition 126.577. The visible StreamName terminal read is pulled back to the finite window and request rows that the Phase $\mathsf{Real}^{\uparrow}$ route actually consumes. If the detected defect lies on the Real boundary of that read, apply Definition 126.568; the Real seal identifies the same finite request section rather than a separate ambient completion coordinate.

It remains to account for presentations that differ by window transport. Apply Definition 126.639. The certified reindexing data preserve the finite observation section, so a defect surviving consumer-visible reindexing is still a mismatch of the same finite request rows.

Finally use the fused terminal-window surface. The source-exhaustion theorem Definition 126.626 shows that an accepted terminal-window packet has no source beyond the displayed four-face section and support rows. The reindexing compatibility theorem Definition 126.627 keeps that source fixed under the allowed window transport. Thus every visible support defect pulls back to the finite request-section evidence listed in the statement.

The exclusion clause follows because none of the pullback, seal-lock, determinacy, source-exhaustion, or compatibility surfaces exports a coordinate for quotient stream equality, host sequence equality, selected limits, countable choice, metric completeness, aggregate formalstatus, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Such data therefore cannot carry the detected defect and cannot repair the finite section consumed by the L10 Real route. □

Theorem 126.656 (StreamName terminal support removal witness)**.** *Let U be an accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Delete from the public terminal support of U either one visible face among* $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, or one displayed support row among* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$*. Then the shortened request section cannot present the same carrier verdict, classifier verdict, and consumer-facing terminal route as U.*

The lost verdict is witnessed as a finite request-section mismatch. If the removed coordinate is the finite $\mathsf{StreamName}^{\uparrow}$ *window or one of its displayed request rows, the witness is the finite-section pullback. If the removed coordinate is the* $\mathsf{Real}^{\uparrow}$ *terminal boundary, the witness is the Real-seal lock. If the removed coordinate is a transported window row, the witness is window reindex determinacy. If the removed coordinate is part of the fused terminal-window source, the witness is the terminal-window fusion surface.*

No quotient stream equality, host sequence equality, selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, aggregate formalstatus row, or bridge row can repair the shortened section or replace the finite witness.*

Proof. Begin with the deletion. By Definition 126.653, every visible face and every displayed support row named in the statement is a required coordinate of the accepted terminal packet. Removing one of them therefore loses the public carrier verdict, the public classifier verdict, or the consumer-facing route.

Next localize the loss. Apply Definition 126.654. The failure cannot be carried by an off-route equality, a selected limit, a choice row, a completeness row, an aggregate verification row, or a bridge row. It is located on the deleted face or support row itself.

Now pull the localized failure back to finite BEDC data. Use Definition 126.655. A missing StreamName window or displayed request row is read by the finite-section pullback. A missing Real terminal boundary is read by the Real-seal lock. A missing transported window row is read by window reindex determinacy. A missing coordinate from the fused terminal source is read by the terminal-window fusion source and reindexing compatibility surfaces.

These four witness surfaces exhaust the repair options available to the terminal request section. Since none of them exports quotient stream equality, host sequence equality, selected limits, countable choice, ambient Real completeness, metric completeness, aggregate formalstatus rows, or bridge rows, the shortened section has a finite request-section mismatch and no excluded coordinate can repair it. □

Theorem 126.657 (StreamName terminal four-face support exhaustion)**.** *Let U be an accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. A consumer-visible support coordinate of U is accepted if and only if it is one of the four terminal faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, or one of the displayed rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$*.*

Consequently the accepted support of U has no coordinate for quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, or bridge rows. None of these excluded data can be accepted as a substitute support coordinate for the terminal request section.*

Proof. For the reverse direction, the displayed four faces and support rows are accepted by the terminal packet described in Definition 126.653. That theorem already identifies the visible StreamName, DyadicRatCore, RegSeqRat, and Real faces together with the BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows as the public terminal surface of U.

For the forward direction, let a consumer-visible support coordinate be accepted by the current Phase Real and Completion terminal request section. Apply Definition 126.589. The consumer read is locked to the finite StreamName window, the DyadicRatCore ledger, the RegSeqRat readback, the Real seal boundary, and the same displayed support rows.

Next read the coordinate through the RegSeqRat terminal boundary. By Definition 301.239, terminal acceptance is exactly the four-face packet with those displayed rows. By Definition 301.237, an accepted consumer with the same public verdict has the same finite RegSeqRat source, so no second RegSeqRat-facing support source is available.

Finally pass to the Real exit route. Theorem Definition 38.468 sends every current Phase Real and Completion terminal read through the ordered $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route and the displayed support rows. Therefore every accepted consumer-visible support coordinate is one of exactly those faces or rows.

The excluded objects are not coordinates of that route. Quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate formalstatus rows, and bridge rows occur in neither the StreamName lock, the RegSeqRat exactness and uniqueness surface, nor the Real exit-totality route. Hence none of them is an accepted support coordinate for U. □

Theorem 126.658 (StreamName terminal four-face support transport stability)**.** *Let U and V be accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose their consumer-visible terminal reads expose the same four visible faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and suppose their displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support rows agree after componentwise transport and replay. Then U and V present the same consumer-visible terminal support packet.*

The transport cannot introduce quotient stream equality, host sequence equality, a selected limit, countable choice, metric-completeness data, an aggregate formalstatus row, a bridge row, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness row. The transported packet remains exactly the same four-face BEDC support surface seen by Real-facing and completion-facing consumers.*

Proof. First classify the accepted coordinates. Apply Definition 126.657 to both terminal request sections. Every consumer-visible support coordinate of either section is therefore one of the four displayed faces or one of the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Next use the minimal support theorem. By Definition 126.653, no visible face and no displayed support row can be deleted or substituted while preserving the same Real-facing and completion-facing carrier and classifier verdicts. Since the hypotheses identify the four faces and replay the support rows componentwise, the two sections have the same StreamName-side public surface.

Now pass through the RegSeqRat terminal route. The exactness theorem Definition 301.239 reads the common displayed faces as one terminal RegSeqRat four-face packet, and Definition 301.237 rules out a second RegSeqRat source section with the same public verdict. Thus the transported RegSeqRat component is the same finite source record.

Finally read the common packet through the Real L10 exit lattice. The theorem Definition 38.468 sends every accepted current Phase Real and Completion terminal read through the ordered $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route with the displayed support rows. The two transported presentations therefore determine the same terminal support packet, and the exhaustion theorem excludes every off-route coordinate listed in the statement. □

Theorem 126.659 (StreamName terminal route facewise conservativity)**.** *Let U be an accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section, and let P be a consumer-visible terminal presentation of U. Suppose P preserves the accepted carrier verdict, the accepted classifier verdict, and the consumer-facing terminal route of U. Then P restricts facewise to the same terminal* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces as U, and to the same displayed* BHist, *componentwise* hsame, Cont, Pkg, *and local* NameCert *support rows.*

No quotient stream equality, host sequence equality, selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, aggregate formalstatus row, or bridge row can serve as a substitute coordinate for this conservative terminal presentation.*

Proof. First exhaust the coordinates that a preserved terminal verdict can use. By Definition 126.657, every accepted consumer-visible support coordinate of U is one of the four displayed faces or one of the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert rows.

Assume contrapositively that P does not restrict facewise to this displayed support. Then it either omits one displayed face or row, or it replaces one of them by an off-route coordinate. The defect localization theorem Definition 126.654 forces such a failure to appear on a visible face or displayed support row whenever the carrier verdict, classifier verdict, or terminal route is not preserved.

The hypotheses say that P preserves exactly those three consumer-visible outputs. Hence no such localized defect is available. The exhaustion theorem also says that quotient equality, host sequence equality, selected limits, choice, completeness, aggregate formalstatus, and bridge rows are not accepted support coordinates, so none of them can repair the missing facewise evidence.

It remains to identify the preserved presentation. Read P and U as two accepted terminal presentations with the same visible faces and componentwise replayed support rows. The transport stability theorem Definition 126.658 therefore identifies their consumer-visible terminal support packets. Thus preserving the accepted carrier verdict, classifier verdict, and terminal route is conservative exactly on the four BEDC-visible faces and displayed support rows listed in the statement. □

Theorem 126.660 (StreamName terminal support replay induction)**.** *Let U and V be accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose the replay from U to V begins with the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, and then replays in order the*

DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *visible faces together with the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows. Then finite induction over this ordered support list determines one consumer-visible terminal support packet.*

The induction uses only those displayed faces and rows. It does not use quotient stream equality, host sequence equality, selected limits, countable choice, ambient Real$^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, or bridge rows.*

Proof. Start with the displayed finite StreamName$^{\uparrow}$ window. The minimality theorem Definition 126.653 supplies the base: this window is a required support coordinate and cannot be replaced by any off-route equality, limit, completeness, formalstatus, or bridge datum.

For the induction step, assume the replayed prefix has already determined the same terminal support prefix. Adding the next displayed row preserves determinacy because minimality forbids deleting that row or substituting a different support coordinate while keeping the same Real-facing and completion-facing verdicts.

At the RegSeqRat$^{\uparrow}$ step, the exactness theorem Definition 301.239 identifies the replayed prefix as the terminal four-face RegSeqRat packet. The source uniqueness theorem Definition 301.237 then rules out a second finite RegSeqRat source with the same public verdict.

At the final Real$^{\uparrow}$ step, theorem Definition 38.468 sends every accepted current Phase Real and Completion terminal read through the ordered StreamName$^{\uparrow}$–DyadicRatCore$^{\uparrow}$–RegSeqRat$^{\uparrow}$–Real$^{\uparrow}$ route and the displayed support rows. Thus the induction has exhausted the full terminal route.

By Definition 126.657, every accepted consumer-visible support coordinate is one of these four faces or displayed rows. The componentwise replay hypotheses therefore give exactly the transport hypotheses of Definition 126.658, which concludes that U and V present one terminal support packet. □

Theorem 126.661 (StreamName terminal support Real-seal minimality)**.** *Let U be an accepted current Phase* Real$^{\uparrow}$ *and Completion terminal* StreamName$^{\uparrow}$ *request section. The public Real-facing seal route exported by U is minimal on the displayed* Real$^{\uparrow}$ *seal face and on the support rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert*. Deleting the* Real$^{\uparrow}$ *seal face, or deleting any one of those displayed support rows, prevents the same public Real-facing seal route from being exported.*

The lost route cannot be restored by quotient stream equality, host sequence equality, selected limits, countable choice, ambient Real$^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, or bridge rows.*

Proof. Begin with the terminal four-face support minimality theorem Definition 126.653. It identifies the accepted public verdict with the visible StreamName$^{\uparrow}$–DyadicRatCore$^{\uparrow}$–RegSeqRat$^{\uparrow}$–Real$^{\uparrow}$ route and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Now specialize to the Real-seal coordinate. The open-phase four-face readback Definition 126.450 and the Real-seal lock Definition 126.568 place the terminal Real$^{\uparrow}$ seal after the finite StreamName window, DyadicRatCore ledger, and RegSeqRat readback. Removing that seal therefore removes the public Real-facing endpoint of the route rather than merely changing a presentation of it.

For a displayed support row, apply the defect-localization theorem Definition 126.654. A deleted BHist, hsame, Cont, Pkg, or NameCert row is detected on the same support surface that carries the terminal route, so the export no longer presents the same Real-facing seal route.

Finally use the same open-phase readback lemmas together with minimality. The excluded coordinates are not faces or support rows of the four-face route, and defect localization does not let them carry the missing seal or support coordinate. Hence they cannot restore the deleted Real-seal route. □

Theorem 126.662 (StreamName finite-cover terminal support minimality)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections. Suppose every accepted overlap displays the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support rows when read through either local section. Then the cover-descent terminal support packet for the family is accepted exactly on those four visible faces and displayed support rows.*

Moreover, deleting any one visible face or any one displayed support row prevents the resulting cover packet from presenting the same carrier verdict and the same classifier verdict to Real-facing or completion-facing consumers. This finite-cover minimality claim uses no quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, or bridge rows.*

Proof. First form the cover surface. Apply Definition 126.598 to the finite family. The accepted-overlap hypotheses place the carrier verdict and classifier verdict on the common StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, Real seal boundary, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next compare all choices of cover presentation. The common-refinement theorem Definition 126.599 says that any certified finite refinement reads the same public four-face classifier surface. Thus the cover-descent terminal packet is independent of the chosen finite presentation once the displayed overlap rows agree.

Now work on each terminal section. For every $i \in I$, Definition 126.653 says that the local terminal packet is accepted exactly on the four visible faces and the displayed support rows, and that deleting any one of these coordinates loses the same Real-facing or completion-facing verdict. Therefore a shorter cover packet would already be shorter on some accepted local section or accepted overlap.

It remains to pass from local minimality to cover minimality. Use Definition 126.660 on the ordered list consisting of the StreamName finite window, the DyadicRatCore ledger, the RegSeqRat readback, the Real seal boundary, and the displayed support rows. The induction transports the local terminal packets through the cover in the same order and gives one consumer-visible terminal support packet.

Finally rule out substitutes for the missing coordinate. The exactness theorem Definition 301.239 identifies the RegSeqRat terminal boundary with precisely the four-face support packet, while Definition 301.237 rules out a second finite RegSeqRat source with the same public verdict. The Real exit totality theorem Definition 38.468 then forces every accepted Phase Real and Completion terminal read through the same ordered route.

The coordinates listed outside the statement are absent from that route. Hence quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate formalstatus rows, and bridge rows cannot supply the deleted face or support row, and cannot preserve the same carrier and classifier verdicts. □

Theorem 126.663 (StreamName finite-cover terminal support defect localization)**.** *Let $\{Q_i\}_{i\in I}$ be a finite family of accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections satisfying the accepted-overlap hypotheses of Definition 126.662. Suppose a cover-descent terminal support packet for the family fails to preserve the carrier verdict, the classifier verdict, or the consumer-facing terminal route presented by that finite cover. Then the failure is witnessed on one local face among* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, or on one displayed support row among* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert*.*

The witness is componentwise on one accepted local section or accepted overlap. Quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, and bridge rows cannot serve as the defect carrier and cannot serve as the repair coordinate for the cover-descent terminal packet.*

Proof. First fix the cover support. By Definition 126.662, the finite cover presents its terminal support packet exactly on the StreamName finite-window face, the DyadicRatCore ledger face, the RegSeqRat readback face, the Real seal face, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now restrict the failed cover presentation to one accepted local section or accepted overlap where the carrier verdict, classifier verdict, or terminal route is lost. Apply Definition 126.654 on that local section. The local theorem places the defect on one of the four visible faces or on one displayed support row, and excludes quotient, host-sequence, limit, choice, completeness, aggregate-status, and bridge coordinates.

It remains to show that this local witness is still the cover witness. Use Definition 126.660 on the ordered cover support list. The induction transports the local localization through the StreamName window, DyadicRatCore ledger, RegSeqRat readback, Real seal boundary, and displayed support rows without adding any off-route coordinate.

At the RegSeqRat face, Definition 301.239 identifies the transported boundary with the terminal four-face support packet, and Definition 301.237 rules out a second finite RegSeqRat source with the same public verdict. Thus a RegSeqRat defect cannot be repaired by changing quotient equality, selected limits, or ambient completeness data.

Finally Definition 38.468 sends every accepted current Phase Real and Completion terminal read through the same ordered $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ exit route. The cover-descent packet therefore has no remaining coordinate outside the four faces and displayed rows on which a carrier verdict, classifier verdict, or consumer-facing terminal route defect could live. □

Theorem 126.664 (StreamName finite-cover terminal support removal witness)**.** *Let* $\{Q_i\}_{i\in I}$ *be a finite family of accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion* $\mathsf{StreamName}^{\uparrow}$ *request sections satisfying the accepted-overlap hypotheses of Definition 126.662. Delete from the cover-descent terminal support packet either one visible face among* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, or one displayed support row among* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$*. Then the shortened cover packet cannot present the same carrier verdict, classifier verdict, and consumer-facing terminal route as the original finite cover.*

The lost verdict is witnessed on one accepted local section or accepted overlap. Quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate formalstatus rows, and bridge rows cannot replace the deleted face or displayed row and cannot repair the cover-descent packet.*

Proof. Start with the finite cover support. By Definition 126.662, every visible face and displayed row named in the statement is a required coordinate of the cover-descent terminal packet. Deleting one of those coordinates therefore removes data needed by the accepted carrier verdict, classifier verdict, or consumer-facing terminal route.

Now localize the loss. Apply Definition 126.663 to the shortened cover packet. The defect is witnessed on one accepted local section or accepted overlap, and it lies on a visible $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ face or on one displayed BHist, hsame, Cont, Pkg, or local $\mathsf{NameCert}$ row.

On that local section, use Definition 126.656. The local deletion already prevents presentation of the same carrier verdict, classifier verdict, and consumer-facing terminal route, with the witness read as a finite request-section mismatch.

It remains to transport the local loss back through the cover. Use Definition 126.660 on the ordered cover support list. The induction replays the local witness through the StreamName finite window, DyadicRatCore ledger, RegSeqRat readback, Real seal boundary, and displayed support rows, so the cover cannot regain the lost terminal packet from another section of the same family.

Finally rule out substitutes. The exactness theorem Definition 301.239 identifies the RegSeqRat boundary with precisely the four-face terminal support packet, and Definition 301.237 rules out a second finite RegSeqRat source with the same public verdict. The Real exit totality theorem Definition 38.468 sends every accepted current Phase Real and Completion terminal read through the same ordered route.

Thus quotient stream equality, host sequence equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate formalstatus rows, and bridge rows are off-route for this finite cover. They cannot supply the deleted coordinate and cannot preserve the same carrier verdict, classifier verdict, and consumer-facing terminal route. □

126.49 StreamName Finite-Cover Support Removal Global Obstruction

This finite-cover support surface is the $\mathsf{StreamName}^{\uparrow}$-side read of the L10 route through chapter 291, chapter 301, and chapter 38: those sibling chapters supply the dyadic ledger, regular-sequence readback, and terminal $\mathsf{Real}^{\uparrow}$ seal faces consumed by the theorems in this file.

Theorem 126.665 (StreamName finite-cover support removal global obstruction)**.** *Let $\{Q_i\}_{i\in I}$ be an accepted finite cover of current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request sections feeding the DyadicRatCore–RegSeqRat–Real four-face route. Suppose the accepted overlaps display the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source, and the same* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows. If one displayed local support row is deleted from any member Q_j, then the cover-descent terminal support packet cannot present the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source.*

The obstruction is global over the finite cover: quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a replacement regularity selector cannot supply the deleted row or restore the same cover-descent source.*

Proof. First restrict to the member where the deletion occurs. The local removal witness theorem Definition 126.656 says that deleting a displayed BHist, hsame, Cont, Pkg, or local NameCert row from the accepted terminal packet loses the carrier verdict, classifier verdict, or consumer-facing terminal route of that local section.

Next descend through the finite cover. Apply Definition 126.662. The cover-descent terminal packet is accepted exactly on the common $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the displayed support rows on the accepted overlaps. Therefore a local deletion in Q_j is not confined to a private presentation choice; it removes one coordinate of the cover support itself.

Now localize the resulting cover defect. By Definition 126.663, a failure of the cover-descent carrier verdict, classifier verdict, or terminal route is witnessed on one accepted local section or

accepted overlap, and the witness lies on a displayed face or support row. In the present case the witness is the deleted local support row, so the cover cannot still present the same $\mathsf{Real}^{\uparrow}$ seal-visible source.

Finally rule out hidden replacement data. The selector nonchoice theorem Definition 126.343 prevents the $\mathsf{RegSeqRat}^{\uparrow}$ regularity row from re-entering as a hidden selector, host function, countable-choice witness, quotient representative, selected limit, ambient completeness coordinate, or independent completed-real carrier. Together with the exclusion clauses in the finite-cover minimality and defect localization theorems, this leaves no quotient equality, limit, choice, metric-completeness datum, aggregate formalstatus row, bridge row, ambient $\mathsf{Real}^{\uparrow}$ completeness principle, or replacement selector able to supply the missing row. □

Theorem 126.666 (StreamName finite-cover support invariance under refinement)**.** *Let $\{Q_i\}_{i\in I}$ and $\{R_k\}_{k\in K}$ be accepted finite covers of the same current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Suppose they admit an accepted common refinement $\{S_\ell\}_{\ell\in L}$, and that after pullback to the refinement the two covers agree on the four visible faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and on the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. If one displayed local support row is deleted in either original cover, then every accepted refinement-equivalent presentation fails to present the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source.*

Equivalently, changing the finite-cover presentation cannot hide the missing support row. Quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ *completeness, and replacement regularity selectors are not refinement coordinates for restoring that source.*

Proof. Pull both finite covers to the accepted common refinement $\{S_\ell\}_{\ell\in L}$. By hypothesis the pullbacks display the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and the same BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows. Thus the deleted row is a deleted coordinate of the refined cover support, not merely a feature of one presentation.

Apply Definition 126.662 to the refined cover. The accepted terminal support packet on the refinement is accepted exactly on the four visible faces and the displayed support rows. Deleting one displayed local support row therefore removes a coordinate needed for the same carrier verdict, classifier verdict, and terminal route.

Next localize the failure with Definition 126.663. The defect is witnessed on one accepted refined local section or accepted refined overlap, and it lies on a visible face or on a displayed support row. In this case the witness is the deleted local support row transported to the common refinement.

Finally apply Definition 126.665 on the refined presentation. The theorem rules out quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ completeness, and replacement regularity selectors as substitute coordinates. Hence no presentation whose support agrees after refinement can present the same $\mathsf{Real}^{\uparrow}$ seal-visible source after the row is missing. □

Theorem 126.667 (StreamName finite-cover support preservation exactness)**.** *Let $\{Q_i\}_{i\in I}$ and $\{R_k\}_{k\in K}$ be accepted finite covers of the same current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Suppose they admit an accepted common refinement $\{S_\ell\}_{\ell\in L}$, and that on every accepted refined member and accepted refined overlap the two pullbacks preserve the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source, and the same displayed* BHist*, componentwise*

hsame, Cont, Pkg, *and local* NameCert *support rows. Then the two finite-cover terminal support packets present the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source after descent.*

The exactness criterion has no further coordinates. It does not assume or produce quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or replacement regularity selectors.*

Proof. Apply Definition 126.662 to the accepted common refinement. The accepted terminal support packet is identified by the four visible faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert rows on refined members and overlaps. By hypothesis the two pullbacks preserve exactly these rows, so their refined support packets expose the same seal-visible source.

It remains to exclude a hidden local defect. Use Definition 126.663 contrapositively: if the descended packets failed to present the same $\mathsf{Real}^{\uparrow}$ seal-visible source, the failure would be witnessed on some accepted refined member or accepted refined overlap, and the witness would lie on one of the displayed visible faces or support rows. Each such row is preserved by hypothesis, so no such local witness exists.

Finally compare with the negative boundary supplied by Definition 126.666. If one displayed row were missing, refinement would detect the loss and the same seal-visible source could not be presented. Since no row is missing, the complete finite L10 route is precisely the four visible faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with their displayed local support, and no quotient equality, selected limit, choice principle, metric-completeness datum, bridge row, ambient completeness principle, or replacement selector is an extra coordinate. □

Theorem 126.668 (StreamName finite-cover support preservation reflection)**.** *Let* $\{Q_i\}_{i\in I}$ *and* $\{R_k\}_{k\in K}$ *be accepted finite covers of the same current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section. Suppose they admit an accepted common refinement* $\{S_\ell\}_{\ell\in L}$*, and suppose the descended finite-cover terminal support packets present the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source. Then, after pullback to the accepted common refinement, the two presentations agree on the four visible faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and on the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

The reflection direction has exactly the same boundary as the exactness direction. It reflects only the finite-cover L10 support rows and does not reflect quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, aggregate formalstatus rows, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or replacement regularity selectors.*

Proof. Argue contrapositively on the accepted common refinement. Suppose that after pullback the two presentations differ on one of the four visible faces or on one displayed BHist, componentwise hsame, Cont, Pkg, or local NameCert row.

By Definition 126.663, the changed or missing displayed row localizes a finite-cover terminal defect on some accepted refined member or accepted refined overlap. The defect is not an invisible presentation choice: it lies on one of the rows from which the refined terminal support packet reads its carrier verdict, classifier verdict, or consumer-facing terminal route.

Apply Definition 126.665 to that localized support defect. A refined presentation with such a defect cannot present the same $\mathsf{Real}^{\uparrow}$ seal-visible source merely by deleting, changing, or omitting the displayed row. The obstruction also rules out quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, aggregate formalstatus rows, ambient $\mathsf{Real}^{\uparrow}$ completeness, and replacement regularity selectors as substitute coordinates for the defective row.

Finally use Definition 126.666. Passing between the original finite covers and the accepted common refinement cannot hide the localized defect or convert it into the same descended $\mathsf{Real}^{\uparrow}$

seal-visible source. Therefore the assumed sameness of the descended source forces agreement, after pullback, on all four visible faces and on the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert rows.

Together with Definition 126.667, this gives the bidirectional finite-cover support criterion for the L10 StreamName exit: preserving exactly the displayed finite support yields the same descended $\mathsf{Real}^{\uparrow}$ source, and the same descended $\mathsf{Real}^{\uparrow}$ source reflects preservation of exactly that displayed finite support. □

Theorem 126.669 (StreamName finite-cover support bicriterion)**.** *Let $\{Q_i\}_{i\in I}$ and $\{R_k\}_{k\in K}$ be accepted finite covers of the same current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion terminal* $\mathsf{StreamName}^{\uparrow}$ *request section, and suppose that they admit an accepted common refinement $\{S_\ell\}_{\ell\in L}$. The descended finite-cover terminal support packets present the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source if and only if, after pullback to the accepted common refinement, the two presentations agree on the four visible faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and on the displayed* BHist, *componentwise* hsame, Cont, Pkg, *and local* NameCert *rows.*

This bicriterion is only a criterion for the finite-cover L10 support surface. It excludes quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, aggregate formalstatus rows, ambient $\mathsf{Real}^{\uparrow}$ *completeness, and replacement regularity selectors; none of these is an additional coordinate of the descended support packet, and none is recovered from the agreement assertion.*

Proof. For the forward direction, assume the descended finite-cover terminal support packets present the same $\mathsf{Real}^{\uparrow}$ seal-visible source. Apply Definition 126.668. It reflects that sameness back to agreement, after pullback to the accepted common refinement, on the four visible faces and on the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert rows.

For the reverse direction, assume agreement after pullback on exactly those visible faces and displayed support rows. Apply Definition 126.667. The exactness theorem descends that preserved finite support to the same $\mathsf{Real}^{\uparrow}$ seal-visible source for the two finite-cover terminal support packets.

Finally record the boundary of the equivalence. The exactness direction does not add quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ completeness, or replacement regularity selectors, and the reflection direction does not reflect quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, aggregate formalstatus rows, ambient $\mathsf{Real}^{\uparrow}$ completeness, or replacement regularity selectors. Therefore the criterion binds only the displayed finite-cover support rows and has no hidden coordinate beyond them. □

126.50 StreamName finite-window diagonal section

Theorem 126.670 (StreamName finite-window diagonal section)**.** *Every finite diagonal section read by a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and then by a* $\mathsf{Real}^{\uparrow}$ *seal factors through the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window request. The displayed route is the finite window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, the* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The section exports no quotient stream equality, quotient equality of Cauchy presentations, selected limit, host real equality, countable-choice row, metric-completeness datum, or ambient completion coordinate.*

Proof. Start with the finite-window budget handoff of Definition 126.390. It places every $\mathsf{RegSeqRat}^{\uparrow}$ regularity row, dyadic tolerance row, and $\mathsf{Real}^{\uparrow}$ seal row behind the same displayed StreamName finite observation-budget surface.

Next apply the finite-request section packet of Definition 126.554 and Definition 126.555. These results package the endpoint, readback, budget, StreamName window, hsame, Cont, Pkg, and local NameCert rows as one finite request section rather than as separate diagonal sources.

Use the RegSeqRat handoff of Definition 126.574. The diagonal read reaches the $\mathsf{Real}^{\uparrow}$ seal only after the finite StreamName window and the RegSeqRat readback have been transported by hsame and replayed by Cont through the displayed support rows.

Finally apply the Real seal non-escape boundary of Definition 126.557 and the finite request seal lock of Definition 126.568. They exclude quotient streams, quotient Cauchy-presentation equality, selected limits, host real equality, countable choice, metric completeness, and ambient completion from the route. Thus the diagonal section is exactly the finite StreamName request transported and replayed through the listed BEDC rows. □

Theorem 126.671 (StreamName finite-window nested-refinement induction)**.** *Let*

$$W_0, W_1, \ldots, W_k$$

be an accepted finite chain of $\mathsf{StreamName}^{\uparrow}$ *windows such that each successor row* W_{i+1} *refines the previous window* W_i *by retaining the displayed pointwise rational observations and only appending later finite requests. Then the whole chain preserves the displayed pointwise rational observations, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row exposed by* W_0*. The induction exports no quotient stream equality, selected limit, host sequence equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Proceed by induction on the finite refinement chain.

The base chain has only W_0. The conclusion is exactly the local finite-window diagonal section of Definition 126.670: the displayed observations, transport, replay, provenance, and local NameCert rows are already present, and no excluded coordinate is available.

For the successor step, assume the claim for the chain ending at W_i and read the accepted refinement row from W_i to W_{i+1}. The finite-window pullback of Definition 126.555 keeps the old rational observations visible inside the later window, while the reindexing route of Definition 126.390 places the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers behind that retained window surface.

Transport is componentwise. The hsame row of the successor refinement agrees with the previously displayed observations on the old window and only names the appended finite requests separately. The displayed Cont replay and Pkg provenance therefore compose along the same finite route, and the local $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ row remains the one read by the refined window chain.

Finally, Definition 126.557 and Definition 126.568 exclude quotient streams, selected limits, host sequence equality, countable choice, and ambient Real completeness at each successor stage. The finite induction therefore preserves exactly the listed StreamName rows for the whole chain. □

126.51 StreamName L10 gateway obligation surface

Theorem 126.672 (StreamName L10 gateway NameCert obligation surface)**.** *Every L10 gateway read exported by a* $\mathsf{StreamName}^{\uparrow}$ *packet is pinned to a finite* BHist *observation window before it can be consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*. The public route consists of the finite StreamName carrier rows of Definition 126.1, the certificate fields of Definition 126.13, the finite request sections of Definition 126.554 and Definition 126.555, the terminal Real seal locks of Definition 126.568 and Definition 126.605, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

The gateway exports no quotient stream equality, host sequence equality, selected limit, countable-choice row, metric-completeness datum, or ambient completion coordinate.

Proof. Read the carrier and certificate fields first. The StreamName packet exposes unary finite observations, bundle membership rows, pointwise rational classifier rows, transport rows, replay rows, provenance rows, and local naming rows; it does not expose a host sequence.

Next apply the finite request sections. They package endpoint, window, RegSeqRat budget, dyadic tolerance, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows as one finite gateway request.

Finally use the terminal seal locks. They force every Real-facing read to pass through the finite request section before the $\mathsf{Real}^{\uparrow}$ seal is available, and they refuse quotient streams, selected limits, host real equality, countable choice, metric completeness, and ambient completion. □

Theorem 126.673 (StreamName L10 gateway window reindex stability)**.** *Let W_0 and W_1 be accepted $\mathsf{StreamName}^{\uparrow}$ finite-window packets that are related by a certified reindex route. If their displayed endpoint rows, pointwise classifier rows, BHist support, hsame comparisons, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows transport componentwise, then every L10 gateway read through W_0 determines the same public finite observation section as the corresponding read through W_1.*

The stability statement is internal to the finite StreamName gateway. It does not assert quotient stream equality, host sequence equality, selected limits, countable choice, completion data, or a standard bridge row.

Proof. Start with finite-window reindex determinacy, Definition 126.639. Its hypotheses identify the endpoint rows, reindex route, and displayed support rows that determine a public finite observation section.

Use the public reindexing identity and associative laws, Definition 126.640 and Definition 126.641. They show that certified identity routes and composable certified routes preserve the displayed classifier, unary observation, bundle-membership, hsame, Cont, Pkg, and local naming rows.

Apply the terminal coverage theorem, Definition 126.642. Since the transported packets expose the same public finite observation section, every L10 gateway read reaches the same four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no quotient, host, limit, choice, completion, or bridge coordinate. □

Theorem 126.674 (StreamName L10 gateway Real-RegSeqRat handoff)**.** *Every Real-facing L10 gateway read of a $\mathsf{StreamName}^{\uparrow}$ packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff uses finite observation windows, regular rational readback, dyadic budget rows, terminal real seal locks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows. It does not export quotient stream equality, quotient equality of Cauchy presentations, selected limits, host real equality, countable choice, metric completeness, or ambient completion.

Proof. Begin with the L10 gateway factorization of Definition 126.347. It places the StreamName point rows, unary observations, bundle ledger, regular source row, dyadic observation ledger, finite tail witness, finite approximation row, and $\mathsf{Real}^{\uparrow}$ seal row on one displayed route.

Next use the RegSeqRat handoff and finite-window diagonal section, Definition 126.574 and Definition 126.670. They ensure that the Real-facing read is not detached from the same finite StreamName request.

The terminal Real seal locks then close the route. By Definition 126.568 and Definition 126.605, a consumer receives only the finite observation, regular-rational, dyadic-budget, and Real-seal rows, not quotient equality, selected limits, host equality, choice, completion, or a standard bridge. □

Theorem 126.675 (StreamName L10 face-window realization)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$ *L10 gateway packet consumed as Phase Real and Completion evidence realizes each finite request window through four displayed faces: the* $\mathsf{StreamName}^{\uparrow}$ *finite section, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. These four faces are carried together with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

The realization exports no quotient stream, selected limit, quotient equality of Cauchy presentations, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host completion coordinate.*

Proof. Start with the L10 gateway obligation surface of Definition 126.672. It pins every gateway request to finite StreamName carrier rows, finite request sections, dyadic tolerance data, regular-rational readback, terminal Real locks, componentwise transport, replay, provenance, and local naming.

Apply the reindex-stability theorem, Definition 126.673. Certified reindex routes preserve the endpoint rows, pointwise classifier rows, BHist support, hsame comparisons, Cont routes, Pkg provenance, and local NameCert rows, so the same finite request window can be read through its transported public section.

Finally use the Real-RegSeqRat handoff of Definition 126.674. It orders the request through the StreamName finite section, RegSeqRat readback, DyadicRatCore budget, and Real seal before any Phase Real or Completion consumer sees evidence. Since each cited route refuses quotient streams, selected limits, countable choice, and ambient Real completeness, the four-face realization has the same refusal surface. □

Theorem 126.676 (StreamName finite request obligation surface)**.** *A finite request section of an accepted* $\mathsf{StreamName}^{\uparrow}$ *packet must expose the requested index, opened* BHist *window, point certificate row, pointwise classifier row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may read the stream. The downstream consumer receives only that finite request surface, not quotient stream equality, host sequence equality, selected limits, countable choice, or ambient Real completeness.*

Proof. Start with the finite request vocabulary already used by the StreamName child files. A request section names one index, opens the corresponding finite BHist window, and records the point certificate and pointwise classifier rows attached to that window.

The structural rows then keep the request local. The hsame row transports the opened window and classifier componentwise, the Cont row replays the finite request, and the Pkg and NameCert rows record provenance and local naming.

Only after these rows are visible may the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301 read exactly the finite request rows as a regular rational source, and only after that downstream handoff may a $\mathsf{Real}^{\uparrow}$ consumer inspect a seal route. Since the section contains no quotient stream equality, host sequence equality, selected limit, choice, or ambient completeness coordinate, those data cannot enter through the finite request surface. □

126.52 Pointwise Certified Source

Definition 126.677 (Pointwise certified NameCert source)**.** *A pointwise certified NameCert source for* $\mathsf{StreamName}^{\uparrow}$ *is a finite packet*

$$W = (I, S, B, H, C, P, N)$$

where I is the requested index ledger, S is the certified stream source row, B is the opened BHist *window at I, H is componentwise* hsame *transport for the point row and classifier row, C is displayed* Cont *replay for the finite request, P is* Pkg *provenance over the StreamName, DyadicRatCore, RegSeqRat, and Real routes, and N is the local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row.*

The classifier reads only the displayed point row and the opened finite window. It exposes no quotient stream equality, host sequence equality, selected limit, countable-choice row, or $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Theorem 126.678 (Pointwise classifier stability)**.** *Let $W_0 = (I_0, S_0, B_0, H_0, C_0, P_0, N_0)$ and $W_1 = (I_1, S_1, B_1, H_1, C_1, P_1, N_1)$ be pointwise certified NameCert sources for* $\mathsf{StreamName}^{\uparrow}$*. If a displayed* hsame *route transports the requested index ledger, source row, opened* BHist *window, point classifier row, replay row, provenance row, and local naming row from W_0 to W_1, then every pointwise classifier read accepted at W_0 is accepted at W_1 with the transported finite request surface.*

The transported read remains a StreamName finite-window read that may be consumed by $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *only through their existing NameCert routes. It does not become quotient stream equality, a selected limit, host real equality, or a completeness principle.*

Proof. Start with the carrier rows of Definition 126.677. The pointwise source has only the requested index ledger, certified stream row, opened BHist window, transport row, replay row, provenance row, and local naming row available.

Use the displayed hsame route to move each row componentwise. The point certificate and classifier row are transported together with the opened window, so no host sequence or quotient stream coordinate is introduced during the comparison.

Replay the transported request with the Cont row. The replay keeps the request inside the same finite StreamName surface, after which Pkg provenance records the permitted sibling route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since the packet contains no selected limit, host equality, countable choice, or completeness row, stability cannot export one. □

Theorem 126.679 (Constant certified point stream)**.** *For a constant rational point displayed as a* $\mathsf{StreamName}^{\uparrow}$ *pointwise certified source, every accepted finite request section opens the same point row, the same* BHist *window shape, the same pointwise classifier row, the same* hsame *transport, the same* Cont *replay, the same* Pkg *provenance, and the same local* NameCert *row. The resulting constant stream is a certified point stream for the L10 route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, not a host sequence equality or quotient stream representative.*

Proof. Project the pointwise source of Definition 126.677. Constancy is recorded as a displayed point row inside the finite request surface, so every request opens a finite BHist window of the same certificate shape.

Transport identical requests by the hsame row and replay them by the Cont row. These rows preserve the point certificate, classifier row, provenance, and local naming row without adding any host function or quotient coordinate.

Finally read the downstream route only through Pkg provenance. The consumer may pass the certified point stream to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as finite NameCert data, while selected limits, host real equality, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness remain absent. □

Theorem 126.680 (Pointwise constant finite request section)**.** *A constant rational point displayed as a pointwise certified* $\mathsf{StreamName}^{\uparrow}$ *NameCert source determines, for every accepted finite request, the same requested index ledger, certified source row, opened* BHist *window, point classifier row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. These rows form the finite request section available to any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer before that consumer can read the constant point stream.*

Proof. Use the pointwise certified source of Definition 126.677. Its displayed packet already separates the index ledger, certified source row, opened history window, transport row, replay row, provenance row, and local naming row from any downstream completion-facing interpretation.

Apply Definition 126.679. Constancy identifies the rational point row across accepted requests only inside the finite request surface, so the same classifier row and the same opened BHist window are replayed for each request.

The section handed to $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ is therefore exactly the displayed row package. Since neither Definition 126.677 nor Definition 126.679 supplies quotient stream equality, a selected limit, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, no downstream consumer can read such data from the constant section. □

Theorem 126.681 (Pointwise source reindex cofinality)**.** *Every accepted pointwise certified* $\mathsf{StreamName}^{\uparrow}$ *source whose finite request section is transported along a displayed unary reindexing row still factors through the same pointwise source, componentwise classifier, and local* $\mathsf{NameCert}$ *rows before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may read it. The reindexed request may change the displayed finite ledger index, but it does not export quotient stream equality, host function equality, a selected infinite tail, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the packet of Definition 126.677. The only available data are the requested index ledger, certified stream source row, opened BHist window, componentwise transport, replay, provenance, and local naming row.

Apply Definition 126.678 to the unary reindexing row. The classifier is transported componentwise with the opened finite window, so the target request remains a pointwise certified StreamName request rather than a host-level stream comparison.

The constant-section route of Definition 126.680 supplies only the base case in which the reindexing row leaves the displayed point fixed. General reindexing is therefore cofinal only through the same finite source, classifier, and NameCert rows.

The downstream consumer can then read the reindexed section through Pkg provenance toward $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$. Since no coordinate in the source supplies quotient equality, host function equality, an infinite tail selector, countable choice, or completeness, none is exported by the handoff. □

Theorem 126.682 (Pointwise map composition finite-window stability)**.** *Let* $W = (I, S, B, H, C, P, N)$ *be an accepted pointwise certified* $\mathsf{StreamName}^{\uparrow}$ *NameCert source. Suppose a displayed finite map row* M *is carried in the same packet, with its finite graph, source and target index ledgers, opened* BHist *window, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *row all displayed before the request is handed to any downstream consumer. Then the composed finite-window read along* M *still factors through the original pointwise source classifier and local naming row before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can read it.*

The composed read remains a finite StreamName request surface. It exports no quotient stream equality, no host function equality, no selected infinite tail, no countable-choice row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the source packet of Definition 126.677. The displayed map row contributes only a finite graph, the two finite ledgers, the opened history window, componentwise transport, replay, provenance, and the local naming row, so the composed request has no hidden host-level stream data.

Apply Definition 126.678 to transport the opened window and classifier row across the displayed graph of M. Then apply Definition 126.681 to keep the target request inside the same finite pointwise source surface after the ledger has been mapped.

The consumer route is therefore read only through the existing Pkg provenance. Since neither the source packet nor the displayed map row supplies quotient stream equality, host function equality, a selected infinite tail, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the composed finite-window handoff cannot export any of them. □

126.53 Finite Observation Section Cofinality

Theorem 126.683 (StreamName finite observation section cofinality)**.** *Every finite observation request on a pointwise certified* $\mathsf{StreamName}^{\uparrow}$ *source factors through a finite section window that is cofinal for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff reads. Concretely, after the request opens its index ledger, certified source row, finite* BHist *window, pointwise classifier row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, any later regular-rational or real-seal consumer may refine the request only by another finite section window over the same StreamName source. The handoff exposes finite BHist, hsame, Cont, and Pkg rows rather than quotient stream equality, selected limits, host real equality, or ambient completeness.*

Proof. Start with Definition 126.678. It keeps every accepted pointwise classifier read attached to the requested index ledger, certified source row, opened BHist window, transport row, replay row, provenance row, and local naming row.

Use Definition 126.680 for the section shape. The finite request section is the row package visible to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers before either consumer can read a constant or regular point stream.

Now pass to the RegSeqRat and Real handoff files already carried by the StreamName hub. Their handoff reads consume finite windows and terminal seal rows through the same displayed StreamName source, so a larger downstream request is represented by a finite refinement of the same section window.

Thus the finite section windows are cofinal for the L10 handoff route. Since each step is expressed through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, no quotient stream equality, selected limit, host real equality, or ambient completeness principle is introduced. □

Theorem 126.684 (StreamName finite observation section functoriality)**.** *Let S be a certified* $\mathsf{StreamName}^{\uparrow}$ *source and let*

$$U \longrightarrow V \longrightarrow W$$

be two composable finite observation-section refinements over S. If both refinements preserve the index ledger, finite BHist *windows, pointwise classifier rows,* hsame *transport,* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{StreamName}^{\uparrow}}$ *rows, and the* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *handoff surface, then their composite is again a finite observation-section refinement over S preserving exactly those rows. The composite introduces no quotient stream equality, selected limit, countable-choice row, host real equality, or ambient completeness principle.*

Proof. Apply finite observation section cofinality, Definition 126.683, to the source S and to each displayed refinement. Both arrows are finite section refinements over the same certified source, so their source row and index ledger are already fixed before any downstream consumer reads the handoff.

Use diagonal coherence, Definition 126.686, to compare the middle section V with the source and target windows. The comparison supplies a common finite section in which the classifier rows, BHist windows, hsame transports, Cont replays, Pkg rows, and local naming rows remain attached to S.

Now apply section induction, Definition 126.687, along the finite spine obtained by concatenating the two refinement traces. Monotone extension, Definition 126.688, identifies the old rows inside the concatenated section, so composition changes only the appended finite request rows.

The RegSeqRat and Real handoff surface is therefore preserved by the composite finite section. Since each cited step is expressed only through finite observation rows and their displayed transport, replay, provenance, and NameCert data, the composite has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 126.685 (StreamName finite observation section public cofinality)**.** *Every public* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff read through* $\mathsf{StreamName}^{\uparrow}$ *factors through the finite observation section package of Definition 126.683. Any later refinement remains a finite section refinement over the same certified StreamName source, with the same index ledger, finite* BHist *window, pointwise classifier row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

The public handoff surface therefore exports finite sections only. It contains no quotient stream equality, selected limit, countable-choice row, host real equality, or ambient completeness principle.

Proof. Start with finite observation section cofinality, Definition 126.683. It already states that finite observation requests cofinally cover the RegSeqRat and Real handoff reads over the displayed StreamName source.

Apply pointwise classifier stability, Definition 126.678. The classifier row, finite BHist window, transport row, replay row, provenance row, and local naming row remain attached to the same source when the handoff is refined.

The RegSeqRat and Real handoff inputs consume exactly those finite section rows before reading regular rational or real-seal data. Hence a public handoff cannot bypass the finite observation package; it may only request a larger finite section over the same source.

Since the route never introduces quotient streams, selected limits, countable choice, host real equality, or ambient completeness, the public cofinality interface is the same finite-section interface already witnessed locally. □

Theorem 126.686 (StreamName finite observation diagonal coherence)**.** *Let two finite observation section refinements read the same certified* $\mathsf{StreamName}^{\uparrow}$ *source before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff. Then they admit a common finite refinement over that source whose pointwise classifier row, finite* BHist *window,* hsame *transport row,* Cont *replay row,* Pkg *provenance row, and local* NameCert *row still factor through the same public RegSeqRat and Real handoff surface. The common refinement synchronizes the two observations without adding quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness.*

Proof. Apply finite observation section cofinality, Definition 126.683, to each refinement. Both observations are finite section windows over the same certified StreamName source, so their downstream reads are represented by finite windows in the same index ledger.

Use public cofinality, Definition 126.685, for the handoff surface. The RegSeqRat and Real consumers see only the finite section package, so a section large enough to contain both windows is again a public handoff refinement over the same source.

Finally apply pointwise classifier stability, Definition 126.678. It keeps the classifier row, BHist window, hsame transport, Cont replay, Pkg provenance, and local NameCert row attached to the common source after the finite window is enlarged.

Thus the two observations have a diagonal finite refinement that preserves exactly the rows visible to the RegSeqRat and Real handoff route. Since no step introduces a quotient stream, a selected limit, countable choice, host real equality, or ambient completeness, the synchronized refinement remains a finite-means StreamName observation. □

Theorem 126.687 (StreamName finite observation section induction)**.** *Every finite chain of accepted* $\mathsf{StreamName}^{\uparrow}$ *observation-section refinements over a pointwise certified source is generated from the empty request section by finitely many successor steps. Each successor step opens one finite* BHist *window, preserves the pointwise classifier row for the same certified source, and replays through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff reads the section. Consequently every finite handoff-facing chain is represented by a finite section trace over the same StreamName source, not by quotient stream equality, selected limits, countable-choice rows, host real equality, or ambient completeness.*

Proof. The empty request section is the base surface supplied by finite observation section cofinality, Definition 126.683. It exposes only the index ledger, certified source row, and the finite row package visible to later RegSeqRat and Real consumers.

For the successor step, open the next finite BHist window over the same certified StreamName source. Pointwise classifier stability keeps the classifier row attached to that source, and the cofinality theorem keeps the opened window inside the same hsame, Cont, Pkg, and local NameCert replay package.

For two successor choices at the same finite stage, use diagonal coherence, Definition 126.686. It supplies a common finite refinement that preserves the displayed rows and synchronizes the observations without changing the public handoff surface.

Finally use public cofinality, Definition 126.685, to pass the finished finite chain to a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. The consumer reads only the finite section trace just generated, so the construction never introduces quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 126.688 (StreamName finite observation section monotone extension)**.** *Let an accepted finite observation section over a certified* $\mathsf{StreamName}^{\uparrow}$ *source be enlarged by additional certified unary indices. Then every window, bundle-membership row,* hsame *comparison,* Cont *replay,* Pkg *provenance row, and local* NameCert *row already admitted on the old section remains unchanged on that old section. The enlarged section only appends finite request rows; it does not reinterpret earlier observations, replace the source classifier, or introduce quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness.*

Proof. Induct over the appended finite request spine. In the empty case the enlarged section is the original section, so the old windows, membership rows, transports, replays, provenance rows, and local naming rows are identical by carrier projection.

For the successor step, add one certified unary index to the already enlarged section. Finite observation section cofinality, Definition 126.683, keeps the appended request inside a finite section window over the same certified StreamName source.

Use diagonal coherence, Definition 126.686, to compare the old section with the one-step enlargement. The common refinement restricts back to the old window rows and preserves the same pointwise classifier, hsame transport, Cont replay, Pkg provenance, and local NameCert rows there.

Finally apply section induction, Definition 126.687, to iterate this argument over the whole appended spine. Each step changes only the newly appended finite request row, so every previously admitted row remains fixed on the old section. □

Theorem 126.689 (StreamName finite-window handoff for constructive IVT moduli)**.** *Every accepted* ConstructiveIVTModulus$^{\uparrow}$ *bisection request opens only finitely many* StreamName$^{\uparrow}$ *observation windows before* RegSeqRat$^{\uparrow}$ *readback and the terminal* Real$^{\uparrow}$ *seal. The handoff reads the located interval source, opposite-sign witnesses, finite bisection schedule, and dyadic tolerance ledger of the modulus carrier, then opens exactly the endpoint and midpoint windows requested by that finite schedule. It does not claim a completed stream, selected branch, quotient equality, host root selector, countable choice, or ambient completeness.*

Proof. Project the ConstructiveIVTModulus$^{\uparrow}$ carrier. Its bisection schedule and tolerance ledger are finite rows, so they determine a bounded list of endpoint and midpoint observations.

Apply the StreamName$^{\uparrow}$ finite-window discipline already used by the L10 countable-name gateway. Each observation is a displayed window request in the carrier route, and no completed stream object is produced.

The same windows are then repacked by the RegSeqRat$^{\uparrow}$ readback row and only afterward reach the Real$^{\uparrow}$ seal. Since the carrier has no selected branch, quotient-equality, root-selection, choice, or completeness coordinate, the finite-window handoff is the full exported StreamName boundary. □

Theorem 126.690 (StreamName finite-request cover pullback)**.** *Let a* ConstructiveIVTModulus$^{\uparrow}$ *carrier open a finite family of endpoint or midpoint requests during its bisection schedule. Then that family pulls back to a finite* StreamName$^{\uparrow}$ *cover whose* RegSeqRat$^{\uparrow}$ *readback rows and terminal* Real$^{\uparrow}$ *seal row are unchanged from the original modulus carrier. The pullback only records the displayed request windows, dyadic tolerance rows, regular-rational readback rows, terminal seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows; it does not add completed streams, selected branches, quotient equality, countable choice, or ambient completeness.*

Proof. Project the finite bisection schedule from the ConstructiveIVTModulus$^{\uparrow}$ carrier. Its endpoint and midpoint requests form a bounded list because the carrier stores a finite schedule and a displayed dyadic tolerance ledger.

Apply Definition 126.689 to each request in that list. Each endpoint or midpoint observation is therefore represented by a finite StreamName$^{\uparrow}$ window, with no completed stream or branch selector produced.

Collect the resulting windows into one finite cover indexed by the request list. The cover uses the same dyadic tolerance rows as the original schedule, so the RegSeqRat$^{\uparrow}$ readback rows are the same rows already present in the carrier.

Finally repack the unchanged readbacks into the original terminal Real$^{\uparrow}$ seal row, carrying along the same hsame, Cont, Pkg, and local NameCert data. Since no coordinate of the carrier stores quotient equality, countable choice, or ambient completeness, the finite cover is only the pullback of the displayed StreamName windows. □

Theorem 126.691 (StreamName pointwise classifier induction)**.** *Let W be an accepted finite unary window for a* $\mathsf{StreamName}^{\uparrow}$ *packet. Suppose every displayed index in W carries pointwise classifier evidence, and suppose the window, classifier rows,* BHist *observations,* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows are transported together through the existing finite-window handoff surface. Then the whole finite window has a classified* $\mathsf{StreamName}^{\uparrow}$ *read. The classified read remains a finite observation packet; it exports no quotient stream equality, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the finite unary window W. In the initial case, the window contains only the first displayed observation, so the pointwise classifier evidence at that observation gives the classified StreamName read after the local BHist, hsame, Cont, Pkg, and NameCert rows are projected.

For the successor step, assume the shorter window has already been classified. Add one displayed observation at the successor index, together with its pointwise classifier evidence.

Apply the finite-window transport route to keep the previously classified prefix and the new observation in the same hsame and Cont packet. Then use the pointwise handoff rows to attach the new classifier evidence to the enlarged window without changing the source StreamName packet.

The resulting packet classifies the enlarged finite window through exactly the displayed observations, classifier rows, transport rows, replay rows, provenance, and local naming data. Since the induction never introduces an infinite representative, quotient equality, a selected limit, countable choice, or ambient completeness coordinate, none is exported by the classified read. □

Theorem 126.692 (StreamName finite-prefix pointwise classifier induction)**.** *Let $F = (k, W, D, R, H, C, P, N)$ be an accepted* $\mathsf{FinitePrefixStream}^{\uparrow}$ *carrier whose displayed prefix windows carry pointwise* $\mathsf{StreamName}^{\uparrow}$ *classifier evidence at every displayed index. Then every displayed prefix index $i \leq k$ has a classified* $\mathsf{StreamName}^{\uparrow}$ *read obtained through exactly the finite* BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of F.*

The classified prefix read does not export quotient stream equality, selected tails, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Apply Definition 7253.4 to the accepted carrier F. For each displayed $i \leq k$, the observation route is forced through the corresponding StreamName window row in W, the matching dyadic tolerance row in D, the RegSeqRat handoff row in R, and the structural rows H, C, P, N.

Now apply Definition 126.691 to the finite StreamName window exposed at that index. The assumed pointwise classifier evidence supplies the classifier row, while the observation induction supplies the finite BEDC support rows needed to transport and replay it.

Combining the two routes gives a classified StreamName read at each displayed prefix index. Since both cited routes are finite-window routes and neither has a coordinate for quotient streams, selected tails, countable choice, host real equality, ambient Real completeness, or bridge checking, the combined prefix classifier read exports none of those objects. □

Theorem 126.693 (StreamName pointwise finite-window induction)**.** *Let W be a certified finite observation section for a* $\mathsf{StreamName}^{\uparrow}$ *packet, displayed as a UnaryHistory window with its* BHist *observations, pointwise classifier rows, successor-window projection rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *data. Suppose a finite-window predicate is stable under pointwise classifier transport and under successor-window projection along the displayed UnaryHistory section. If the predicate holds at the initial displayed observation, then it holds on every certified finite observation section of W.*

The induction is internal to the displayed UnaryHistory window. It does not export host $\mathsf{Nat}^{\uparrow}$ *induction as a public primitive, quotient stream equality, selected tails, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Start with the carrier definition of $\mathsf{StreamName}^{\uparrow}$ and read the certified observation section as a finite UnaryHistory packet. The initial displayed observation is covered by the assumed base evidence together with the local BHist, hsame, Cont, Pkg, and NameCert rows.

For the successor step, apply the pointwise classifier induction of Definition 126.691 to transport the predicate across the classifier row attached to the newly displayed observation. Then use finite observation section cofinality from Definition 126.683 to keep the enlarged section inside the certified observation surface.

The successor-window projection determinacy of Definition 126.299 identifies the predecessor section and the enlarged section as the displayed UnaryHistory step named by the certificate. Thus the predicate is propagated only through public finite-window rows.

Iterating over the displayed UnaryHistory constructors gives the claim for every certified finite observation section. The argument never exposes host $\mathsf{Nat}^{\uparrow}$ induction as part of the public certificate and never adds quotient streams, selected tails, countable choice, host real equality, ambient Real completeness, or bridge data. □

Theorem 126.694 (StreamName pointwise tail-section induction bridge)**.** *Let* $\mathcal{T}$ *be a certified* $\mathsf{StreamName}^{\uparrow}$ *tail section consumed by a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or finite-window* $\mathsf{Real}^{\uparrow}$ *handoff. Suppose a finite-window predicate is stable under pointwise classifier transport, successor-window projection, and finite observation-section cofinal refinement over the displayed* BHist *observations, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. If the predicate holds at the initial displayed observation of the source window, then it holds on every certified finite tail section of* $\mathcal{T}$ *that is consumed by the handoff.*

The conclusion is a tail-section handoff statement over the same certified source window. It does not introduce quotient stream equality, selected limit data, countable choice, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Apply the pointwise finite-window induction theorem Definition 126.693 to the certified source window. This propagates the predicate across every displayed finite observation section using only pointwise classifier transport and successor-window projection.

Next use finite observation-section cofinality, Definition 126.683. The tail section read by the downstream $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ handoff is a finite cofinal refinement of the same displayed source, not a new stream object or a chosen external tail.

Finally apply the public cofinality and monotone-extension rows attached to the handoff. They identify the consumed tail section as a finite refinement over the same BHist, hsame, Cont, Pkg, and local NameCert support rows. The predicate therefore reaches every certified tail section consumed by the handoff while staying inside the finite public packet. □

Theorem 126.695 (StreamName terminal-window inhabited section)**.** *Every accepted terminal* $\mathsf{StreamName}^{\uparrow}$ *finite window consumed by a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff has an inhabited displayed UnaryHistory section. Its initial observation, successor projection, pointwise classifier rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows are all visible in the consumed finite window.*

The inhabited section is a finite public section of the displayed terminal window. It is not obtained from quotient stream equality, a selected tail, countable choice, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Apply Definition 126.693 to the terminal finite window with the predicate saying that the current displayed section has a visible initial observation, successor projection interface, classifier row, and structural support. The base row is the initial displayed observation of the UnaryHistory window.

For the consumed terminal tail, apply Definition 126.694. It transports the same predicate from the source window to every finite tail section read by the downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff.

The resulting section is inhabited because the initial observation survives the tail-section handoff and every successor projection is displayed as a finite UnaryHistory row. The support rows are exactly the visible BHist, hsame, Cont, Pkg, and local NameCert rows named by the two induction theorems, so no quotient stream equality, selected tail, countable choice, or ambient Real completeness is introduced. □

Theorem 126.696 (StreamName finite-stream forward route)**.** *Every finite* $\mathsf{StreamName}^{\uparrow}$ *observation section first consumes the finite-stream packet of chapter 5916 before it is reindexed as pointwise StreamName window data. The route reads the finite request list, displayed observation window, componentwise* hsame *transport, finite* Cont *replay,* Pkg *provenance, and local* NameCert *row from the* $\mathsf{FiniteStream}^{\uparrow}$ *packet, then repacks those rows as the pointwise StreamName window surface used by downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers. It exports no infinite stream equality, no selected limit, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite-stream packet cited in chapter 5916. Its public rows are finite request, finite observation, transport, replay, provenance, and local naming data.

The StreamName observation section only reindexes those displayed finite rows as pointwise window data. The reindexing changes the consumer-facing window name but does not add a quotient stream, a hidden selector, or an infinite equality principle.

Consequently every downstream read through $\mathsf{StreamName}^{\uparrow}$ still factors through the same finite packet. Since the input has no coordinate for selected limits or ambient $\mathsf{Real}^{\uparrow}$ completeness, the StreamName route exports none. □

Theorem 126.697 (StreamName finite-stream forward preservation)**.** *The finite-stream-to-*$\mathsf{StreamName}^{\uparrow}$ *route of Definition 126.696 preserves the finite request list, displayed observation window, componentwise* hsame *rows, finite* Cont *replay,* Pkg *provenance, and local* NameCert *rows when the resulting observation section is consumed by* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *handoff packets. Each downstream consumer may refine the observation only through a finite* $\mathsf{StreamName}^{\uparrow}$ *section window over the same finite-stream source, and no consumer obtains infinite stream equality, quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the route.*

Proof. Start with the forward route, Definition 126.696. It repacks the finite-stream request list, finite observation window, componentwise transport, replay, provenance, and local naming rows as a pointwise StreamName window without adding an infinite equality coordinate.

Apply finite observation section cofinality, Definition 126.683, to the resulting StreamName window. Any RegSeqRat or Real consumer sees only a finite section refinement over the same certified source, with the same displayed support rows.

The DyadicRatCore handoff reads the same terminal tolerance surface through those finite StreamName windows before it reaches the RegSeqRat and Real faces. Hence its consumer-facing data is also bounded by the preserved finite request, observation, transport, replay, provenance, and naming rows.

Thus all three downstream packets consume the same finite support supplied by the original finite-stream packet. Since the route and its finite observation refinements never add infinite stream equality, quotient stream equality, selected limits, countable choice, or ambient Real completeness, preservation holds at the handoff surface. □

Theorem 126.698 (StreamName finite-stream forward composition)**.** *Two consecutive finite-stream-to-*$\mathsf{StreamName}^{\uparrow}$ *forward routes compose into one finite observation section. The composite section preserves the finite request lists, displayed observation windows, componentwise* hsame *rows, finite* Cont *replay,* Pkg *provenance, and local* NameCert *rows from the two input sections, and it remains the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface consumed by downstream* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *packets.*

The composed route exports no infinite stream equality, quotient stream equality, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 126.696 to the first finite-stream packet. It repacks the initial request list, observation window, transport, replay, provenance, and local naming rows as a pointwise $\mathsf{StreamName}^{\uparrow}$ observation section.

Apply the same route to the second finite-stream packet, whose source begins at the terminal finite section displayed by the first route. The shared endpoint is identified by componentwise hsame transport and replayed by the displayed Cont rows, so the two sections meet inside finite observation data.

Use finite observation section cofinality, Definition 126.683, to concatenate the consecutive request windows into a single finite $\mathsf{StreamName}^{\uparrow}$ section over the same certified source. The Pkg and NameCert rows are inherited from the two input packets as provenance and local naming for the composed section.

Finally apply preservation, Definition 126.697. Downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers therefore see the composite as one finite-window route and receive no infinite stream equality, quotient stream equality, selected limits, countable choice, or ambient Real completeness. □

Chapter 127

A Concrete Naming Certificate for $\mathsf{TotallyBounded}^{\uparrow}$

The totally-bounded interface $\mathsf{TotallyBounded}^{\uparrow}$ packages a metric carrier admitting, for every $\varepsilon > 0$, an explicit finite ε-net. It depends on $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ (chapter 57) and is the prerequisite for $\mathsf{NameCert}_{\mathsf{CompactMetric}^{\uparrow}}$ (chapter 129).

127.1 Carrier definition

Definition 127.1 (Singleton metric totally-bounded instance)**.** *The singleton metric totally-bounded instance is the* $\mathsf{Met}^{\uparrow}$ *carrier on histories*

$$T_{\mathsf{e}}(x) := \mathsf{hsame}(x, \mathsf{e}).$$

Its classifier is

$$x \sim_T y \quad \Longleftrightarrow \quad T_{\mathsf{e}}(x) \wedge T_{\mathsf{e}}(y) \wedge \mathsf{hsame}(x, y),$$

and its distance witness is the concrete metric row

$$d_T(x, y, \delta) := \mathsf{MetricDistanceWitness}(x, y, \delta).$$

For every positive tolerance history ε, the finite net is the one-point closed bundle $\mathsf{Bcons}(\mathsf{e}, \mathsf{Bnil})$ *with ledger center* e*. The bundle membership row is the head* $\mathsf{InBundle}$ *constructor, and the coverage ledger for a carried point x uses the empty-distance witness* $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, \mathsf{e})$ *together with the carried endpoint row $T_{\mathsf{e}}(x)$.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedUpTasteGate single carrier alignment ChapterTasteGa`

Theorem 127.2 (Singleton metric totally-bounded laws)**.** *The instance of Definition 127.1 satisfies the concrete finite-net laws required by* $\mathsf{TotallyBounded}^{\uparrow}$*:* e *inhabits the carrier, the unique net center* e *lies in the carrier, every carried point is covered by that center at empty distance, the net witness is stable under the singleton classifier $\sim_T$, and the coverage row is compatible with metric empty-distance exactness from Definition 57.7.*

Proof. Carrier inhabitation and center membership are both the reflexive hsame row on e. Thus the ledger center named by the one-point $\mathsf{ProbeBundle}$ is accepted by T_{e}.

For coverage, fix a carried point x. The row $T_{\mathsf{e}}(x)$ is exactly $\mathsf{hsame}(x, \mathsf{e})$, so the singleton classifier records $x \sim_T \mathsf{e}$ using the carried endpoint row, reflexivity of e, and the same endpoint hsame witness.

Apply Definition 57.7 to identify the empty-distance metric row with the two endpoint classifiers to e. This packages $\mathsf{MetricDistanceWitness}(x, \mathsf{e}, \mathsf{e})$ as the coverage witness for the unique center, and the head $\mathsf{InBundle}$ constructor records that the center belongs to $\mathsf{Bcons}(\mathsf{e}, \mathsf{Bnil})$.

Stability under $\sim_T$ follows by unpacking the classifier on the source and target endpoints. Both histories classify with e, so the empty-distance exactness row supplies the same metric coverage after transport, and the closed bundle branch is unchanged. □

Lean4 (definition) `BEDC.Derived.TotallyBoundedUp.SingletonMetricTotallyBoundedInstance`
Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.SingletonMetricTotallyBounded laws`

The ten-field fit is: traditional target total boundedness by finite ε-nets; BEDC source $\mathsf{Met}^{\uparrow}$ over BHist; carrier T_{e}; classifier $\sim_T$; stability from hsame and $\mathsf{MetricDistanceWitness}$ empty-distance exactness; ledger one center and one $\mathsf{ProbeBundle}$ branch; core theorem Definition 127.2; exactness singleton coverage; standard bridge terminal metric space; status $\mathsf{ProofTarget}$.

Definition 127.3 (Totally bounded ProbeBundle net carrier)**.** *Let X be a metric BEDC carrier whose distance rows are encoded by* $\mathsf{MetricDistanceWitness}$*. A totally bounded ProbeBundle net carrier at tolerance history ε is the history relation*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$$

whose accepted row consists only of BHist *ledger data:*

1. *a positive* $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ *tolerance history ε;*
2. *a closed* $\mathsf{ProbeBundle}$ *$\mathcal{P}$ of center names;*
3. *for every probe row* $\mathsf{InBundle}(p, \mathcal{P})$*, a center history c_p admitted by the metric carrier X;*
4. *for every such bundled probe, a positive* $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ *radius history r_p;*
5. *for every source-point history x admitted by X, a bundled probe row* $\mathsf{InBundle}(p, \mathcal{P})$*, a distance history d_p, and a row* $\mathsf{MetricDistanceWitness}(x, c_p, d_p)$ *whose* $\mathsf{Rat}^{\uparrow}$ *comparison ledger certifies $d_p \leq \varepsilon$.*

The finite carrier is the closed $\mathsf{ProbeBundle}$ *inductive together with the displayed* $\mathsf{InBundle}$ *coverage rows. It is not an external enumeration of centers, and no host-level list is exposed as certificate data.*

Lean4 (definition) `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet`

Theorem 127.4 (Totally bounded ProbeBundle net coverage stability)**.** *Let X be a metric BEDC carrier with* $\mathsf{MetricDistanceWitness}$ *classifier transport. Suppose*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$$

holds. If the tolerance history ε, the bundled center histories c_p, the positive radius histories r_p, and all displayed distance histories d_p are transported by hsame*, and if the positive-tolerance, positive-radius, and bound comparisons are transported by the* $\mathsf{Rat}^{\uparrow}$ *classifier, then the same* $\mathsf{ProbeBundle}$ *$\mathcal{P}$ with the transported ledgers is again a valid totally bounded net for the transported tolerance history.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet coverage hsame transport`

Proof. Proceed by induction on the closed ProbeBundle spine. In the Bnil case there is no bundled probe row to transport, and any accepted coverage row is absent by inversion on InBundle.

For a cons bundle $\mathsf{Bcons}(p, \mathcal{P}_0)$, the head probe is handled by the head constructor of InBundle and the tail probes are handled by the induction hypothesis through the tail constructor. The center history and the positive radius history attached to each case are moved along the supplied hsame rows, while the $\mathsf{Rat}^{\uparrow}$ classifier transports positivity of the radius ledger.

Now fix a transported source-point history x'. The original coverage row for an hsame representative x supplies a bundled probe p, a center c_p, and a distance row $\mathsf{MetricDistanceWitness}(x, c_p, d_p)$ with $d_p \leq \varepsilon$. Metric hsame transport carries this row to $\mathsf{MetricDistanceWitness}(x', c'_p, d'_p)$ for the transported center and distance histories.

It remains to repack the bound. The $\mathsf{Rat}^{\uparrow}$ classifier transports the comparison $d_p \leq \varepsilon$ to the comparison $d'_p \leq \varepsilon'$, and the same $\mathsf{InBundle}(p, \mathcal{P})$ constructor records that the chosen probe still belongs to the finite carrier. Thus the transported ledgers satisfy the carrier relation for the same closed bundle. □

127.2 The certificate

Theorem 127.5 (TotallyBoundedUp ProbeBundle NameCert target)**.** *The carrier of Definition 127.3, the classifier given by transported tolerance and bundled ledger histories, the coverage-stability theorem Definition 127.4, and the ledger policy that records the tolerance,* ProbeBundle, InBundle *membership rows, center histories, positive radius histories,* MetricDistanceWitness *rows, and* $\mathsf{Rat}^{\uparrow}$ *bound rows form the paper-level* NameCert *target for* $\mathsf{TotallyBounded}^{\uparrow}$ *over a* $\mathsf{Met}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet name certificate`

Proof. The source field is the underlying $\mathsf{Met}^{\uparrow}$ carrier together with its accepted point histories and metric-distance classifier. The pattern field is the relation $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$ for each positive $\mathsf{Rat}^{\uparrow}$ tolerance history.

The classifier acts on the visible tolerance and on every bundled ledger row: centers, radii, distance histories, and comparison proofs are transported by the displayed hsame and $\mathsf{Rat}^{\uparrow}$ classifier rows. The finite part of the certificate is unchanged because $\mathcal{P}$ is the closed ProbeBundle inductive and membership is witnessed by InBundle constructors.

The stability obligation is exactly Definition 127.4. The ledger policy records the same rows named in the carrier definition, so source soundness, pattern soundness, classifier stability, and ledger soundness are all witnessed by the displayed carrier and stability theorem. This is the required current paper target for $\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}$. □

Theorem 127.6 (Totally bounded ProbeBundle net tolerance weakening)**.** *Let X be a metric BEDC carrier whose distance rows are encoded by* MetricDistanceWitness*, and write $\delta \leq_{\mathsf{Rat}^{\uparrow}} \eta$ for a visible non-strict* $\mathsf{Rat}^{\uparrow}$ *comparison ledger. Suppose*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$$

holds in the sense of Definition 127.3. If ε' is a positive $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ *tolerance history and $\theta : \varepsilon \leq_{\mathsf{Rat}^{\uparrow}} \varepsilon'$ is supplied, then the row $net_{\varepsilon'}$ obtained from net by reusing the same closed* ProbeBundle *spine, bundled centers, radius histories,* InBundle *rows, source-point probe selections,*

and MetricDistanceWitness *rows, and by replacing each displayed bound* $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon$ *with its transitive composite* $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon'$*, satisfies*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon', \mathcal{P}, net_{\varepsilon'}).$$

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet tolerance weakening`

Proof. Unpack the accepted row for *net*. It contains the positive tolerance row for ε, the closed ProbeBundle carrier $\mathcal{P}$, the center and positive-radius ledgers attached to each bundled probe, and, for each source-point history x carried by X, a bundled probe p, a distance history d, a row $\mathsf{MetricDistanceWitness}(x, c_p, d)$, and a comparison ledger $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon$.

To build the row at ε', use the supplied positive tolerance row for ε'. The closed bundle, the InBundle membership rows, the center histories, and the radius histories are copied verbatim from *net*. These clauses of Definition 127.3 do not mention the upper tolerance except through its positivity row.

Now fix a carried source-point history x. Read from *net* the same bundled probe p, the same center c_p, the same distance history d, and the same metric row $\mathsf{MetricDistanceWitness}(x, c_p, d)$. The original coverage comparison gives $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon$. Compose this comparison on the right with $\theta : \varepsilon \leq_{\mathsf{Rat}^\uparrow} \varepsilon'$ by the transitivity field of the non-strict $\mathsf{Rat}^\uparrow$ comparison ledger. The resulting row is $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon'$.

Thus every source point keeps its original bundled probe and metric-distance witness, while only the final rational upper-bound ledger is weakened from ε to ε'. All fields required by the totally bounded ProbeBundle net carrier are therefore present for $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon', \mathcal{P}, net_{\varepsilon'})$. □

Definition 127.7 (Subcarrier-restricted TotallyBounded ProbeBundle net)**.** *Let* $X : \mathsf{BHist} \to \mathsf{Prop}$ *be a metric BEDC carrier with* MetricDistanceWitness *rows, and let* $Y : \mathsf{BHist} \to \mathsf{Prop}$ *be a subcarrier of* X *with inclusion*

$$\iota_Y : \forall y : \mathsf{BHist}.\ Y(y) \to X(y).$$

Given a row

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$$

in the sense of Definition 127.3, the subcarrier-restricted row

$$\mathsf{TotallyBoundedProbeBundleNet}_{X \downarrow Y}(\varepsilon, \mathcal{P}, net|_Y)$$

keeps the positive tolerance history, the closed ProbeBundle $\mathcal{P}$*, every bundled center history* c_p *carried by* X*, every positive radius history* r_p*, and every* $\mathsf{InBundle}(p, \mathcal{P})$ *row from net, while replacing the source-point coverage clause by the same clause quantified only over histories carried by* Y*.*

Lean4 (definition) `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNetSubcarrierRestriction`

Theorem 127.8 (Totally bounded ProbeBundle net subcarrier restriction)**.** *Let* X *be a metric BEDC carrier whose distance rows are encoded by* MetricDistanceWitness*, and let* Y *be a classifier-stable subcarrier of* X*. Thus there are rows*

$$\iota_Y : \forall y : \mathsf{BHist}.\ Y(y) \to X(y)$$

and

$$\sigma_Y : \forall y y' : \mathsf{BHist}.\ Y(y) \to y \sim_X y' \to Y(y'),$$

where $\sim_X$ is the metric classifier of Definition 57.4. If

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$$

holds, then the row obtained by restricting only the source-point coverage quantifier satisfies

$$\mathsf{TotallyBoundedProbeBundleNet}_{X\downarrow Y}(\varepsilon, \mathcal{P}, net|_Y).$$

Moreover, if every bundled center is also carried by Y,

$$\forall p : \mathsf{BHist}.\ \mathsf{InBundle}(p, \mathcal{P}) \to Y(c_p),$$

then the same restricted row is an ordinary $\mathsf{TotallyBoundedProbeBundleNet}(Y, \varepsilon, \mathcal{P}, net|_Y)$ *for the induced metric subcarrier.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet subcarrier restriction`

Proof. Unpack the accepted row for $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}, net)$ using Definition 127.3. It supplies the positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ tolerance history ε, the closed $\mathsf{ProbeBundle}$ spine $\mathcal{P}$, the bundled center histories c_p carried by X, the positive radius histories r_p, and the coverage row for every source-point history carried by X.

The subcarrier-restricted row keeps all bundle-indexed data unchanged. The positive tolerance row, the closed bundle spine, the $\mathsf{InBundle}(p, \mathcal{P})$ witnesses, the ambient center rows $X(c_p)$, and the radius rows are exactly the rows already present in *net*. These clauses do not inspect whether the covered source point is taken from X or from the subcarrier Y.

It remains to verify coverage for a carried point of Y. Fix $y : \mathsf{BHist}$ with $Y(y)$. The inclusion row gives $X(y)$. Applying the coverage clause of *net* to this ambient source point gives a bundled probe p, a membership row $\mathsf{InBundle}(p, \mathcal{P})$, a distance history d_p, a metric row

$$\mathsf{MetricDistanceWitness}(y, c_p, d_p),$$

and a rational comparison ledger

$$d_p \leq_{\mathsf{Rat}^\uparrow} \varepsilon.$$

The restricted row uses precisely these witnesses for the same point y. Thus every Y-carried point is covered by the same closed bundle $\mathcal{P}$ with the same distance and bound ledgers.

The classifier-stability row σ_Y records that the induced carrier is a legitimate classifier-stable subcarrier: replacing a Y-carried endpoint by an ambient-classified endpoint keeps it in Y. No center or distance ledger is altered by this restriction step.

If the additional center-carrier rows $\forall p.\ \mathsf{InBundle}(p, \mathcal{P}) \to Y(c_p)$ are supplied, then the only remaining difference between the relative row and the ordinary $\mathsf{TotallyBoundedProbeBundleNet}$ row over Y disappears. Replace each ambient center clause $X(c_p)$ by the supplied subcarrier center clause $Y(c_p)$, and keep the same positive tolerance, bundle spine, radius rows, coverage witnesses, metric-distance rows, and rational bound ledgers. These are exactly the fields required for $\mathsf{TotallyBoundedProbeBundleNet}(Y, \varepsilon, \mathcal{P}, net|_Y)$. □

Lemma 127.9 (ProbeBundle append membership row separation)**.** *Let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be closed* $\mathsf{ProbeBundle}$ *spines. For every probe history p, a visible membership row*

$$\mathsf{InBundle}(p, \mathsf{bundleAppend}(\mathcal{P}_X, \mathcal{P}_Y))$$

determines either a left membership row $\mathsf{InBundle}(p, \mathcal{P}_X)$ *or a right membership row* $\mathsf{InBundle}(p, \mathcal{P}_Y)$*. Conversely, every left or right membership row injects into the appended bundle. The alternatives are row alternatives: if the same probe name occurs on both sides, the left and right ledger branches remain distinct membership rows.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet append membership separation`

Proof. Induct on the closed spine $\mathcal{P}_X$. If $\mathcal{P}_X = \mathsf{Bnil}$, then $\mathsf{bundleAppend}(\mathsf{Bnil}, \mathcal{P}_Y)$ is $\mathcal{P}_Y$. Thus an appended membership row is exactly a right membership row, and the converse right injection is the identity. There is no left membership row in the empty bundle.

For the cons case write $\mathcal{P}_X = \mathsf{Bcons}(q, \mathcal{P}_0)$. A membership row in

$$\mathsf{InBundle}(p, \mathsf{bundleAppend}(\mathsf{Bcons}(q, \mathcal{P}_0), \mathcal{P}_Y))$$

is either the head equality row $p = q$ or a tail row in $\mathsf{bundleAppend}(\mathcal{P}_0, \mathcal{P}_Y)$. The head equality gives the left membership row in $\mathsf{Bcons}(q, \mathcal{P}_0)$. For a tail row, apply the induction hypothesis. A left-tail result is reinserted by the tail constructor of $\mathsf{InBundle}$ for $\mathsf{Bcons}(q, \mathcal{P}_0)$, and a right result remains a right membership row in $\mathcal{P}_Y$.

The converse injections are obtained by the same constructors. A left head row maps to the appended head row. A left tail row first injects into $\mathsf{bundleAppend}(\mathcal{P}_0, \mathcal{P}_Y)$ by the induction hypothesis and then into the appended cons tail. A right row first injects into $\mathsf{bundleAppend}(\mathcal{P}_0, \mathcal{P}_Y)$ and then into the appended cons tail. This proves both directions and keeps duplicated probe names separated by their visible branch rows. □

Theorem 127.10 (Finite union of totally bounded ProbeBundle nets)**.** *Let $X, Y : \mathsf{BHist} \to \mathsf{Prop}$ be metric BEDC carriers over the same* $\mathsf{MetricDistanceWitness}$ *rows and the same metric classifier of Definition 57.4. Define the union carrier by*

$$(X \cup Y)(z) :\Longleftrightarrow X(z) \vee Y(z).$$

Assume

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_X, net_X) \quad \textit{and} \quad \mathsf{TotallyBoundedProbeBundleNet}(Y, \varepsilon, \mathcal{P}_Y, net_Y)$$

in the sense of Definition 127.3. Let

$$\mathcal{P}_{X \cup Y} := \mathsf{bundleAppend}(\mathcal{P}_X, \mathcal{P}_Y).$$

Then the union carrier $X \cup Y$ carries a totally bounded ProbeBundle net at the same tolerance ε with probe bundle $\mathcal{P}_{X \cup Y}$. Its ledger $net_{X \cup Y}$ is obtained by reading each appended membership row through Definition 127.9: left rows use the corresponding fields of net_X, and right rows use the corresponding fields of net_Y.

Proof. Unpack the two accepted rows by Definition 127.3. The row net_X supplies a positive $\mathsf{NameCert}_{\mathsf{Rat}^\uparrow}$ tolerance history ε, the closed bundle $\mathcal{P}_X$, center histories, positive radius histories, and coverage rows for all points carried by X. The row net_Y supplies the same fields over Y and $\mathcal{P}_Y$ at the same tolerance history. Use either positive-tolerance row for the union net.

The finite carrier of the union net is the closed spine $\mathsf{bundleAppend}(\mathcal{P}_X, \mathcal{P}_Y)$. Since $\mathsf{bundleAppend}$ is generated by recursion on the left closed $\mathsf{ProbeBundle}$ spine, the result is again a closed $\mathsf{ProbeBundle}$.

Fix an appended probe row

$$\alpha : \mathsf{InBundle}(p, \mathcal{P}_{X \cup Y}).$$

By Definition 127.9, α is read as either a left row $\alpha_X : \mathsf{InBundle}(p, \mathcal{P}_X)$ or a right row $\alpha_Y : \mathsf{InBundle}(p, \mathcal{P}_Y)$. In the left case, copy from net_X the center history, positive radius history, and center-carrier row attached to α_X; the center is carried by $X \cup Y$ by the left union injection. In the right case, copy the corresponding fields from net_Y and use the right union injection. If the same

probe name appears in both bundles, these are still two ledger branches selected by the visible membership row, so no side-disjointness assumption is required.

It remains to verify coverage. Let z be a history carried by $X \cup Y$. If $X(z)$ holds, the coverage clause of net_X supplies a probe p, a row $\mathsf{InBundle}(p, \mathcal{P}_X)$, a center history, a distance history d, a metric row

$$\mathsf{MetricDistanceWitness}(z, c_p, d),$$

and a rational comparison ledger $d \leq_{\mathsf{Rat}^\uparrow} \varepsilon$. Inject the left membership row into $\mathsf{InBundle}(p, \mathcal{P}_{X \cup Y})$ using Definition 127.9, and keep the same center, distance, metric witness, and bound ledger. The center-carrier row is promoted from X to $X \cup Y$ by the left union injection.

If $Y(z)$ holds, apply the same argument to net_Y, inject the right membership row into the appended bundle, and promote the center-carrier row by the right union injection. Thus every source point of $X \cup Y$ is covered by a bundled center in $\mathcal{P}_{X \cup Y}$ at distance bounded by the same tolerance ε. All fields required by Definition 127.3 are present for

$$\mathsf{TotallyBoundedProbeBundleNet}(X \cup Y, \varepsilon, \mathcal{P}_{X \cup Y}, net_{X \cup Y}).$$

□

Lean4 ✓ `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet finite union`

Theorem 127.11 (TotallyBoundedUp compact-metric component bridge)**.** *Let X be a certified metric carrier, and suppose that each positive* $\mathsf{Rat}^\uparrow$ *tolerance history ε is assigned a row*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$$

as in Definition 127.3. This family supplies exactly the totally-bounded component required by

$$\mathsf{CompactMetricCertificate}(X)$$

in Definition 129.1: the exported carrier is the same carrier X; the finite carrier at tolerance ε is the closed ProbeBundle *spine $\mathcal{P}_\varepsilon$; membership is read only through rows of* InBundle*; coverage is witnessed by the displayed* MetricDistanceWitness *rows in net_ε; and every bound is one of the displayed* $\mathsf{Rat}^\uparrow$ *comparison ledgers against ε.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.TotallyBoundedCompactMetric component bridge`

Proof. The carrier component is fixed before the net is read. By Definition 129.1, the compact-metric certificate asks for a totally-bounded field over the same certified metric carrier X. By Definition 127.5, each accepted $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ row exports the same target carrier X, so no carrier translation or quotienting step is present.

The finite component is the closed probe spine already carried by the totally-bounded row. The unpacking specified in Definition 127.3 exposes $\mathcal{P}_\varepsilon$ as a closed ProbeBundle, and every center or probe lookup is keyed by a visible InBundle row. Thus the compact-metric field consumes the bundle ledger rather than an ambient finite set or an untracked list of representatives.

The coverage and bound components are also copied without changing their meaning. For each point history carried by X, Definition 127.4 keeps the coverage witness stable under the stated metric carrier interface: the same center row, the same MetricDistanceWitness distance row, and the same $\mathsf{Rat}^\uparrow$ comparison ledger against ε are the rows consumed by the totally-bounded clause of $\mathsf{CompactMetricCertificate}(X)$.

Therefore the bridge row contributes only the totally-bounded component named in Definition 129.1. It does not assert open-cover compactness, does not select a finite subcover, and does not introduce any host-level choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}$.

Constructive story: A totally bounded metric carrier is built by taking an already certified $\mathsf{Met}^{\uparrow}$ carrier and, for each positive $\mathsf{Rat}^{\uparrow}$ tolerance history ε, displaying a closed finite ProbeBundle of center names with membership rows, center histories, radius histories, distance witnesses, and rational bound ledgers. A row is admitted only when every carried source point is assigned one of those bundled probes and a visible MetricDistanceWitness whose distance is classified below ε; data lacking the positive tolerance, closed bundle, center carrier row, membership constructor, or comparison ledger is outside the carrier. Two such net rows are the same when the tolerance and every visible center, radius, distance, and comparison row transport through hsame and the $\mathsf{Rat}^{\uparrow}$ classifier while the same closed bundle spine keeps its membership branches visible. The source is the underlying metric carrier and its point-distance certificate, the pattern is the finite-net coverage package, and the ledger records the tolerance, bundle, membership, centers, radii, coverage distances, and rational inequalities without replacing them by an external finite set. The five name-certificate obligations follow because the singleton carrier at e gives an inhabited example, history-equality and rational classification supply reflexive, symmetric, and transitive comparison, and coverage stability repacks any classified representative as an accepted net. A history package h lies in the signature gap of a $\mathsf{TotallyBounded}^{\uparrow}$ token p when the source-domain accepts its metric carrier and finite-net row, the signature relation pairs that row with an answer net and evidence ledger, and the token-introduction relation names the answer as p; unfolded, the same ProbeBundle centers cover each source point at the positive tolerance encoded by p. The chapter registers this ProbeBundle-net construction under the name $\mathsf{TotallyBounded}^{\uparrow}$; the compact-metric component bridge reads the same finite ledgers downstream, with no open-cover compactness, finite-subcover choice, sequential compactness, or host-level list of centers introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope binds the exported ProbeBundle-net NameCert target over $\mathsf{Met}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$, coverage transport, tolerance weakening, subcarrier restriction, append membership separation, finite union, the $\mathsf{CompactMetric}^{\uparrow}$ component bridge of Definition 127.11, and the consumer references from $\mathsf{CompactMetric}^{\uparrow}$ and the compact uniform-continuity capstone.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TotallyBoundedUp.TotallyBoundedProbeBundleNet\_name\_certificate`

Bridge status: paperBridge

Not claimed: Open-cover compactness, sequential compactness, finite-subcover choice, and bridgeChecked verification are not claimed.

Upgrade path: Only bridgeChecked audit coverage can strengthen the verification axis; the chapter claims no open-cover compactness or sequential compactness principle.

Chapter 128

A Concrete Naming Certificate for $\mathsf{CompleteMetric}^{\uparrow}$

The complete-metric interface $\mathsf{CompleteMetric}^{\uparrow}$ packages a metric carrier in which every regular Cauchy sequence converges to an explicit limit with explicit modulus. It is the L10 sub-milestone certified by the real-completeness capstone (chapter 9238) and the substrate for $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$.

$\mathsf{CompleteMetric}^{\uparrow}$ consumes the $\mathsf{Met}^{\uparrow}$ distance-witness and classifier rows of chapter 57 before adding limit-witness data, so its completion surface is a metric-carrier extension rather than a kernel-only float.

128.1 Carrier definition

Definition 128.1 (Singleton complete metric instance)**.** *The singleton complete metric instance is the metric carrier on histories*

$$C_{\mathsf{e}}(x) := \mathsf{hsame}(x, \mathsf{e}).$$

Its classifier is

$$x \sim_C y \quad \Longleftrightarrow \quad C_{\mathsf{e}}(x) \wedge C_{\mathsf{e}}(y) \wedge \mathsf{hsame}(x, y).$$

A regular Cauchy name for this carrier is a stream of observed histories γ whose every observed history γ_n is carried by C_{e}, together with the regularity ledger asserting that every displayed pair γ_m, γ_n admits the empty-distance metric row $\mathsf{MetricDistanceWitness}(\gamma_m, \gamma_n, \mathsf{e})$ *after transport along their carried endpoint rows. The limit selector sends every such name to* e*. Its modulus is the empty history, and its convergence distance witness at every observation is* $\mathsf{MetricDistanceWitness}(\gamma_n, \mathsf{e}, \mathsf{e})$*.*

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricUpTasteGate single carrier alignment ChapterTasteGa`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton stream witness`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricSingletonLimitSelector visible context metric`
Lean4 (definition) `BEDC.Derived.CompleteMetricUp.CompleteMetricSingletonInstance`
Lean4 (definition) `BEDC.Derived.CompleteMetricUp.CompleteMetricSingletonCarrier`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricSingleton semantic name certificate`

Definition 128.2 (Complete metric limit witness)**.** *Let X be a certified metric carrier over* BHist*, with point classifier, distance rows encoded by* $\mathsf{MetricDistanceWitness}$*, and the* $\mathsf{Rat}^{\uparrow}$ *comparison*

ledgers used for visible tolerances. A complete metric limit witness for a regular Cauchy stream name s is the BEDC relation

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

whose accepted row consists of the following ledgers:

1. *a* $\mathsf{StreamName}^{\uparrow}$ *regular Cauchy stream name s whose observations are carried by X;*
2. *an explicit modulus history μ certifying the regular Cauchy bound for s at each positive* $\mathsf{Rat}^{\uparrow}$ *tolerance;*
3. *a limit history ℓ carried by X;*
4. *for every positive tolerance history ε, a threshold observation history N_ε displayed by μ;*
5. *for every observation history n beyond N_ε, a distance history $d_{n,\varepsilon}$ together with the row* $\mathsf{MetricDistanceWitness}(s($ *and a* $\mathsf{Rat}^{\uparrow}$ *comparison ledger certifying $d_{n,\varepsilon} \le \varepsilon$.*

The witness is therefore a finite-schema history package: the carrier, stream, modulus, limit, metric-distance rows, and rational comparisons are all visible BEDC ledgers, while convergence is recorded by the family of displayed late-observation bounds.

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness metric budget shape lock`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness empty modulus observation endpoint`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness distance modulus positive denominators`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness modulus not empty`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness observation bound transport`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness observation distance package`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness empty limit distance cont readback`
Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton distance rat readback`

128.2 The certificate

Theorem 128.3 (Singleton complete metric laws)**.** *The instance of Definition 128.1 satisfies the complete-metric laws required by* $\mathsf{CompleteMetric}^{\uparrow}$*: every C_{e} regular Cauchy name converges to the selected limit* e*, the selected limit is stable under the classifier $\sim_C$, any two displayed limits are* hsame*, and the convergence ledger is compatible with metric empty-distance exactness from Definition 57.7. This construction is the* $\mathsf{CompleteMetric}^{\uparrow}$ *companion to Definition 127.2.*

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.SingletonCompleteMetric laws`

Proof. Unfold C_{e}. Each observed history γ_n of a regular Cauchy name comes with a carried endpoint row $\mathsf{hsame}(\gamma_n, \mathsf{e})$. The selected limit e is carried by reflexivity of hsame, and the empty modulus asks for no additional history beyond this endpoint ledger.

For convergence, apply Definition 57.7 to the endpoint rows $\mathsf{hsame}(\gamma_n, \mathsf{e})$ and $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$. This identifies the empty-distance metric row $\mathsf{MetricDistanceWitness}(\gamma_n, \mathsf{e}, \mathsf{e})$ with the classifier data already carried by the stream observation and the selected limit.

Stability under $\sim_C$ is obtained by unpacking the classifier. A transported candidate limit is carried by C_{e}, hence it is hsame to e. Transitivity of hsame transports the convergence ledger from

e to that candidate, while the empty-distance exactness theorem supplies the same metric row after the endpoint transport.

Finally, suppose ℓ_1 and ℓ_2 are two displayed limits for the same regular Cauchy name. Each displayed limit is carried by C_{e}, so $\mathsf{hsame}(\ell_1, \mathsf{e})$ and $\mathsf{hsame}(\ell_2, \mathsf{e})$ hold. Symmetry and transitivity of hsame give $\mathsf{hsame}(\ell_1, \ell_2)$, proving uniqueness of displayed limits inside the singleton classifier. □

Theorem 128.4 (Complete metric limit witness stability)**.** *Let X be a certified metric carrier with* $\mathsf{StreamName}^{\uparrow}$ *observation transport,* $\mathsf{MetricDistanceWitness}$ *transport, and* $\mathsf{Rat}^{\uparrow}$ *comparison transport. If*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

holds, and if the stream observations of s, the modulus history μ, the limit history ℓ, every displayed threshold, and every displayed distance history are transported by hsame *together with the corresponding* $\mathsf{Rat}^{\uparrow}$ *positivity and comparison rows, then the transported ledgers form a complete metric limit witness*

$$\mathsf{CompleteMetricLimitWitness}(X, s', \mu', \ell', L')$$

for the transported regular Cauchy stream name s' and transported limit history ℓ'.

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness hsame transport`

Proof. Unpack the witness row. The stream component is pointwise: for each certified observation history, $\mathsf{StreamName}^{\uparrow}$ transport carries the observed history $s(n)$ to the corresponding observed history $s'(n')$, and the regular Cauchy modulus row is moved along the supplied hsame history for μ.

Fix a positive tolerance history ε' transported from ε. The threshold ledger N_ε is carried to $N'_{\varepsilon'}$, and the $\mathsf{Rat}^{\uparrow}$ classifier transports the positivity row for the tolerance together with the order row expressing that an observation is late enough.

For such a late transported observation, the original witness supplies $\mathsf{MetricDistanceWitness}(s(n), \ell, d_{n,\varepsilon})$ and the bound $d_{n,\varepsilon} \le \varepsilon$. Metric-distance transport moves this row to

$$\mathsf{MetricDistanceWitness}(s'(n'), \ell', d'_{n',\varepsilon'}).$$

The remaining ledger is the comparison. Apply the $\mathsf{Rat}^{\uparrow}$ classifier transport to the displayed bound and repack the result as $d'_{n',\varepsilon'} \le \varepsilon'$. Since every field of the carrier row has been transported and the quantification over late observations is preserved pointwise, the transported package is again a complete metric limit witness. □

Theorem 128.5 (CompleteMetricUp limit witness NameCert target)**.** *Let X be a certified* $\mathsf{Met}^{\uparrow}$ *carrier over* BHist*. The carrier of Definition 128.2, the classifier given by transported stream observations, modulus histories, limit histories, threshold ledgers, distance histories, and* $\mathsf{Rat}^{\uparrow}$ *comparison rows, the stability theorem Definition 128.4, and the ledger policy recording* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{MetricDistanceWitness}$ *rows, and* $\mathsf{Rat}^{\uparrow}$ *bounds form the paper-level* $\mathsf{NameCert}$ *target for* $\mathsf{CompleteMetric}^{\uparrow}$ *over X.*

Proof. The source object is the certified metric carrier X together with its $\mathsf{Met}^{\uparrow}$ point classifier and distance-row interface. The BEDC pattern is the regular Cauchy stream name s, the modulus history μ, the carried limit history ℓ, and the family L of late-observation bound ledgers in Definition 128.2.

The classifier is the componentwise transport relation on this package. It requires transported $\mathsf{StreamName}^{\uparrow}$ observations, transported modulus and limit histories, transported threshold histories for each positive tolerance, transported distance histories, and transported $\mathsf{Rat}^{\uparrow}$ positivity and comparison rows. These are exactly the histories named in the public complete-metric witness.

For the stability field, apply Definition 128.4. It takes a displayed complete metric limit witness and the componentwise classifier data just listed, then repacks them as a complete metric limit witness for the transported stream and transported limit.

The ledger policy is therefore visible and finite-schema: it records $\mathsf{StreamName}^{\uparrow}$ observation rows, $\mathsf{MetricDistanceWitness}$ rows from late observations to the selected limit, and $\mathsf{Rat}^{\uparrow}$ comparison rows showing that each displayed distance is below the requested positive tolerance. The remaining entries of the ten-field template are inherited from the same metric carrier: target name $\mathsf{CompleteMetric}^{\uparrow}$, carrier X, exactness by metric endpoint transport, and no separate standard bridge. Thus the listed fields assemble the paper-level $\mathsf{NameCert}$ target for $\mathsf{CompleteMetric}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness name certificate`

Theorem 128.6 (Complete metric limit witness uniqueness)**.** *Let X be a separated certified $\mathsf{Met}^{\uparrow}$ carrier over BHist with point classifier $\sim_X$. Assume the metric certificate supplies symmetry, the triangle comparison row, the positive-half $\mathsf{Rat}^{\uparrow}$ comparison ledger, and zero-distance exactness in the following ledger form: if carried histories a, b have, for every positive $\mathsf{Rat}^{\uparrow}$ tolerance ε, a distance history e_ε with*

$$\mathsf{MetricDistanceWitness}(a, b, e_\varepsilon) \quad \textit{and} \quad e_\varepsilon \le \varepsilon,$$

then $a \sim_X b$. If two displayed complete-metric limit witnesses converge the same regular Cauchy stream name s,

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_0, \ell_0, L_0) \quad \textit{and} \quad \mathsf{CompleteMetricLimitWitness}(X, s, \mu_1, \ell_1, L_1),$$

then their limit histories are point-classifier equal:

$$\ell_0 \sim_X \ell_1.$$

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness singleton uniqueness`

Proof. Unpack the two rows of Definition 128.2. The rows state that ℓ_0 and ℓ_1 are carried by X, and that each positive $\mathsf{Rat}^{\uparrow}$ tolerance supplies a threshold after which the observations of the same stream s are within that tolerance of the displayed limit.

Fix a positive tolerance ε. Choose its positive half ρ_ε together with the comparison ledger $\rho_\varepsilon + \rho_\varepsilon \le \varepsilon$. The first limit witness supplies a threshold N^0_ρ such that every observation n beyond it has a distance row

$$\mathsf{MetricDistanceWitness}(s(n), \ell_0, d^0_{n,\rho}) \quad \text{with} \quad d^0_{n,\rho} \le \rho_\varepsilon.$$

The second witness supplies a threshold N^1_ρ such that every observation n beyond it has a distance row

$$\mathsf{MetricDistanceWitness}(s(n), \ell_1, d^1_{n,\rho}) \quad \text{with} \quad d^1_{n,\rho} \le \rho_\varepsilon.$$

The common-tail ledger of the regular $\mathsf{StreamName}^{\uparrow}$ component gives an observation n_ε beyond both thresholds.

Apply metric symmetry to the first displayed distance row at n_ε, obtaining the same bound for a row from ℓ_0 to $s(n_\varepsilon)$. Apply the triangle comparison row to this symmetric row and to the second displayed row from $s(n_\varepsilon)$ to ℓ_1. This produces a distance history e_ε with

$$\mathsf{MetricDistanceWitness}(\ell_0, \ell_1, e_\varepsilon)$$

and a comparison ledger

$$e_\varepsilon \leq d^0_{n_\varepsilon,\rho} + d^1_{n_\varepsilon,\rho}.$$

The two convergence bounds and the half-tolerance comparison compose in $\mathsf{Rat}^\uparrow$ to give

$$e_\varepsilon \leq \varepsilon.$$

Since ε was an arbitrary positive $\mathsf{Rat}^\uparrow$ tolerance, the pair ℓ_0, ℓ_1 has displayed distance below every positive tolerance. The zero-distance exactness field of the separated metric carrier therefore returns $\ell_0 \sim_X \ell_1$. □

Theorem 128.7 (Complete metric limit witness tolerance weakening)**.** *Let X be a certified $\mathsf{Met}^\uparrow$ carrier over BHist whose distance rows are encoded by $\mathsf{MetricDistanceWitness}$, and write $\delta \leq_{\mathsf{Rat}^\uparrow} \eta$ for a visible non-strict $\mathsf{Rat}^\uparrow$ comparison ledger. Suppose*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

holds in the sense of Definition 128.2. Let Θ be a visible weakening ledger such that, for every positive target tolerance history η, it supplies a positive source tolerance history ε_η and a comparison ledger

$$\theta_\eta : \varepsilon_\eta \leq_{\mathsf{Rat}^\uparrow} \eta.$$

Define L^Θ at η by reusing the branch of L at ε_η: the same threshold N_{ε_η} displayed by μ, the same late-observation distance histories d_{n,ε_η}, and the same rows

$$\mathsf{MetricDistanceWitness}(s(n), \ell, d_{n,\varepsilon_\eta}),$$

while replacing each displayed bound

$$d_{n,\varepsilon_\eta} \leq_{\mathsf{Rat}^\uparrow} \varepsilon_\eta$$

by its transitive composite with θ_η. Then

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L^\Theta)$$

holds for the weakened tolerance ledger family.

Proof. Unpack the accepted row for $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$ using Definition 128.2. It supplies the carried stream name, the modulus history μ, the carried limit history ℓ, and, for every positive source tolerance history ε, a threshold N_ε displayed by μ. For every observation history n beyond that threshold, it also supplies a distance history $d_{n,\varepsilon}$, a row

$$\mathsf{MetricDistanceWitness}(s(n), \ell, d_{n,\varepsilon}),$$

and a rational comparison ledger

$$d_{n,\varepsilon} \leq_{\mathsf{Rat}^\uparrow} \varepsilon.$$

Fix a positive target tolerance history η. The weakening ledger Θ supplies a positive source tolerance ε_η and the comparison row $\theta_\eta : \varepsilon_\eta \leq_{\mathsf{Rat}^\uparrow} \eta$. Read the original witness at that source

tolerance. The threshold N_{ε_η}, the late-observation predicate, the distance history d_{n,ε_η}, and the metric-distance row from $s(n)$ to ℓ are copied unchanged into the target branch.

Only the final upper-bound comparison is altered. For each late observation, the source branch gives

$$d_{n,\varepsilon_\eta} \leq_{\mathsf{Rat}^\uparrow} \varepsilon_\eta.$$

Composing this row on the right with $\theta_\eta : \varepsilon_\eta \leq_{\mathsf{Rat}^\uparrow} \eta$ by transitivity of the non-strict $\mathsf{Rat}^\uparrow$ comparison ledger gives

$$d_{n,\varepsilon_\eta} \leq_{\mathsf{Rat}^\uparrow} \eta.$$

This argument is pointwise in the target tolerance history η. Hence every positive target tolerance has a displayed threshold and every late observation has a displayed metric-distance row bounded by that target tolerance. The carrier, stream, modulus, limit, threshold, and distance ledgers are exactly the rows required by Definition 128.2, so the repacked family L^Θ is a complete metric limit witness for the same stream, modulus, and limit. □

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness tolerance weakening`

Theorem 128.8 (CompleteMetricUp limit-witness standard bridge)**.** *The public* $\mathsf{CompleteMetric}^\uparrow$ *limit-witness interface of Definition 128.2 and Definition 128.5 admits a conservative standard interpretation as explicit Bishop complete-metric convergence data over the same* BHist *carrier. Under this bridge, a BEDC row*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$$

is read as the displayed datum consisting of the stream s, modulus μ, limit ℓ, the convergence rows

$$\mathsf{MetricDistanceWitness}(s(n), \ell, d_{n,\varepsilon}),$$

positive denominator and budget rows for the $\mathsf{Rat}^\uparrow$ *tolerances, tolerance weakening rows, and separated-limit uniqueness. Conversely, any displayed standard convergence datum with exactly these BEDC rows and no additional host primitive repacks as*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L).$$

The bridge asserts only this bidirectional reading of the displayed convergence rows. It does not assert RealUp completeness, Banach completion, compactness, operation descent, host-level quotient equality, or choice.

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness standard bridge row package`

Proof. For the forward reading, start with the public row $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$. The NameCert target Definition 128.5 identifies the carrier, classifier, and ledger policy: the stream observations, modulus history, carried limit, threshold histories, distance histories, and $\mathsf{Rat}^\uparrow$ comparison rows are exactly the public fields of the witness.

Unpacking Definition 128.2 therefore gives the Bishop-style convergence data without changing carrier. For each positive tolerance history ε, the package supplies a threshold displayed by μ; for each late observation it supplies the metric-distance row from $s(n)$ to ℓ and the rational budget row $d_{n,\varepsilon} \leq_{\mathsf{Rat}^\uparrow} \varepsilon$. The positive-denominator and budget readbacks are part of the same visible $\mathsf{Rat}^\uparrow$ ledger surface.

The standard bridge fields are conservative because their stability and comparison behavior are already supplied inside the public interface. Definition 128.4 transports the whole package along

hsame-compatible stream, modulus, limit, threshold, distance, and rational comparison rows. Definition 128.7 supplies the tolerance-weakening component by repacking source tolerance branches through displayed $\mathsf{Rat}^{\uparrow}$ comparison rows. For separated carriers, Definition 128.6 gives the uniqueness of the displayed limit in the point classifier.

For the reverse reading, assume a displayed Bishop complete-metric convergence datum over the same BHist carrier whose entries are precisely the stream s, modulus μ, limit ℓ, positive tolerance rows, thresholds, late-observation metric-distance rows, and $\mathsf{Rat}^{\uparrow}$ budget rows just listed. Place those rows in the five fields of Definition 128.2: stream, modulus, carried limit, tolerance thresholds, and late-observation distance comparisons. Since no extra host equality, quotient, compactness, completion, or selection principle is used, the repacked object is exactly $\mathsf{CompleteMetricLimitWitness}(X, s, \mu, \ell, L)$.

Thus encode and decode are identity-on-rows readings of the same finite BEDC ledger surface. Their round trips are immediate row repackings, theorem preservation is limited to stability, uniqueness, and tolerance weakening as cited above, and the no-host-leak condition holds at paper level because the bridge introduces no public carrier or theorem field beyond those BEDC rows. □

Lean4 ✓ `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness standard bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$.

Constructive story: A complete metric witness is generated over an already certified $\mathsf{Met}^{\uparrow}$ carrier by displaying a regular $\mathsf{StreamName}^{\uparrow}$ Cauchy stream, a modulus history, a carried limit history, and for every positive $\mathsf{Rat}^{\uparrow}$ tolerance a threshold with late-observation distance and bound ledgers. The carrier accepts exactly those packages whose observations stay in the metric carrier and whose late observations have visible MetricDistanceWitness rows to the proposed limit below the requested tolerance; a stream without its modulus, carried limit, threshold row, distance row, or rational comparison is not part of the surface. The classifier is componentwise transport of the stream observations, modulus, limit, thresholds, distances, and rational comparisons by hsame and the $\mathsf{Rat}^{\uparrow}$ classifier, so two witnesses are identified only when the displayed convergence ledger can be repacked field by field. The source is the metric carrier with its stream observations, the pattern is explicit Cauchy convergence to a named limit, and the ledger keeps the observation rows, modulus, thresholds, late distances, and tolerance budgets visible rather than forming a completion quotient. The five name-certificate obligations are supplied by the singleton carrier at e for inhabitation, by the equivalence laws of history-equality and rational comparison for the classifier, and by limit-witness stability for closure under classified replacements. A package h lies in the signature gap of a $\mathsf{CompleteMetric}^{\uparrow}$ token p when the source-domain accepts its regular stream and limit witness, the signature relation pairs it with an answer convergence package and evidence ledger, and the token-introduction relation names that answer as p; concretely, p records the same stream, modulus, limit, and late-distance bounds. The standard bridge reads this finite ledger as Bishop-style complete-metric convergence data and reads it back without adding choice, host completion, quotient equality, compactness, or Banach structure; the chapter registers the construction under the name $\mathsf{CompleteMetric}^{\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature CompleteMetricUp paper-side boundary is bound to the limit-witness NameCert target, the singleton complete metric instance, the singleton limit-selector readback, separated-limit uniqueness, tolerance weakening, StreamNameUp observations, MetricDistanceWitness convergence rows, RatUp comparison ledgers, and the displayed standard bridge for the same finite ledger surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CompleteMetricUp.CompleteMetricLimitWitness\_name\_certificate`

Bridge status: paperBridge

Not claimed: RealUp completeness, Banach completion, compactness, operation descent, host-level quotient equality, choice, or bridgeChecked verification.

Upgrade path: Only a stronger verification grade requires bridgeChecked audit verification for the paper bridge; no further paper closure grade is asserted here.

Chapter 129

A Concrete Naming Certificate for $\mathsf{CompactMetric}^{\uparrow}$

The compact-metric interface $\mathsf{CompactMetric}^{\uparrow}$ packages a metric carrier that is both totally bounded and complete in the senses of $\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}$ (chapter 127) and $\mathsf{NameCert}_{\mathsf{CompleteMetric}^{\uparrow}}$ (chapter 128). It is the metric form of compactness used by the Heine–Cantor capstone (chapter 9237).

129.1 Carrier definition

Definition 129.1 (Compact metric certificate carrier)**.** *Let X be a certified metric carrier over* BHist*. The compact metric certificate carrier*

$$\mathsf{CompactMetricCertificate}(X)$$

is the conjunction of the following BEDC certificate fields:

1. *a* $\mathsf{Met}^{\uparrow}$ *carrier and distance package for X, with distance rows encoded by* $\mathsf{MetricDistanceWitness}$ *and classifier transport on point histories;*
2. *for every positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, a* $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ *package of Definition 127.5;*
3. *for every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream name s whose observations are carried by X, a complete metric limit witness* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$ *as in Definition 128.2.*

Thus compactness is represented by the metric carrier, the finite ProbeBundle-net family at each visible positive tolerance, and the explicit limit-witness family for all regular Cauchy streams. No external compact set or host-level subsequence principle is part of the carrier.

Lean4 (definition) `BEDC.Derived.CompactMetricUp.CompactMetricCertificateCarrier`

129.2 The certificate

Theorem 129.2 (Compact metric carrier obligation)**.** *Let X carry the compact metric certificate relation of Definition 129.1. Then the public carrier row for* $\mathsf{CompactMetric}^{\uparrow}$ *is exactly the underlying* $\mathsf{Met}^{\uparrow}$ *carrier: the same* BHist *point histories are accepted, the same point classifier is used, the displayed distance rows are the inherited* $\mathsf{MetricDistanceWitness}$ *rows, and distance-transport is the*

hsame-*stable transport row supplied by the metric certificate. In the ten-field* NameCert *roadmap template this fixes the source, carrier, classifier, stability, and metric ledger entries before the* TotallyBounded$^{\uparrow}$ *and* CompleteMetric$^{\uparrow}$ *fields are read.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate metric distance transport`

Proof. Unpack the first field of Definition 129.1. It is a Met$^{\uparrow}$ package over BHist, so it already names the accepted point histories, their point classifier, and the distance witness relation used for every metric comparison.

No compact-specific carrier is introduced at this stage. The CompactMetric$^{\uparrow}$ carrier copies those point histories and copies the point classifier, so classifier transport is the same hsame row that the metric certificate uses on endpoints.

It remains to record the ledger surface. Each distance claim exposed by the compact certificate is a MetricDistanceWitness row inherited from Met$^{\uparrow}$, and distance transport is precisely the inherited hsame-stable metric transport. These rows give the carrier, classifier, distance witness, and distance-transport obligations required by the NameCert template. □

Theorem 129.3 (Compact metric finite-net obligation)**.** *Let X carry the compact metric certificate relation of Definition 129.1. For every positive* Rat$^{\uparrow}$ *tolerance history ε, the* TotallyBounded$^{\uparrow}$ *field supplies a closed* ProbeBundle *net with center histories, positive radius histories, bundle-membership rows, metric coverage rows, and rational comparison ledgers. These center, radius, coverage, and ledger rows are stable under* hsame *and the* Rat$^{\uparrow}$ *classifier by Definition 127.4, and they form the finite-net carrier, classifier-stability, and ledger obligations for* CompactMetric$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricTotallyBoundedNet obligation`

Proof. Fix a positive Rat$^{\uparrow}$ tolerance history ε. The second field of Definition 129.1 gives a TotallyBoundedProbeBundleNet$(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ row, where $\mathcal{P}_\varepsilon$ is a closed ProbeBundle and membership is witnessed internally by InBundle constructors.

Unpacking that row gives the finite-net carrier: each bundled probe has a center history carried by the metric source, a positive radius history, and a coverage ledger saying that every carried point has a bundled center within the displayed tolerance. The ledger rows are exactly the tolerance history, center histories, radius histories, membership constructors, MetricDistanceWitness rows, and Rat$^{\uparrow}$ bound comparisons.

Apply Definition 127.4. It transports the positive tolerance, bundled centers, radii, metric distances, and bound comparisons along hsame and the Rat$^{\uparrow}$ classifier while keeping the same closed ProbeBundle carrier. Therefore the finite-net family supplies the required classifier-stability and ledger obligations for the compact metric certificate. □

Theorem 129.4 (Compact metric complete-limit obligation)**.** *Let X carry the compact metric certificate relation of Definition 129.1. For every regular* StreamName$^{\uparrow}$ *Cauchy stream name s whose observations are carried by X, the* CompleteMetric$^{\uparrow}$ *field supplies a* CompleteMetricLimitWitness$(X, s, \mu_s, \ell_s, L_s)$ *row. This row records the stream, modulus, limit classifier, threshold histories, late-observation distance rows, and* Rat$^{\uparrow}$ *comparison ledgers, and it is stable under stream transport, modulus transport, and limit-classifier transport by Definition 128.4. These rows form the complete-limit carrier, classifier-stability, and ledger obligations for* CompactMetric$^{\uparrow}$*.*

Proof. Fix a regular StreamName$^{\uparrow}$ Cauchy stream name s whose observations lie in the inherited Met$^{\uparrow}$ carrier. The third field of Definition 129.1 provides a complete metric limit witness for s, with explicit modulus history μ_s, carried limit history ℓ_s, and late-observation ledger family L_s.

Unpack the witness. Its carrier data are the observed stream histories, the modulus, the limit history, and the displayed threshold histories. Its ledger data are the MetricDistanceWitness rows from late observations to the limit and the corresponding $\mathsf{Rat}^{\uparrow}$ comparisons against each positive tolerance.

Apply Definition 128.4. Stream transport moves the observed histories, modulus transport moves the Cauchy threshold rows, and limit classifier transport moves ℓ_s while preserving the late-distance witnesses and rational bounds. Thus every regular Cauchy stream receives a transported limit-witness package, giving the complete metric part of the $\mathsf{CompactMetric}^{\uparrow}$ NameCert surface. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCompleteLimit obligation`

Theorem 129.5 (Compact metric package from total boundedness and completeness)**.** *For a certified metric carrier X, the three fields of Definition 129.1 form the paper-level* NameCert *target for* $\mathsf{CompactMetric}^{\uparrow}$*. The classifier is inherited from the* $\mathsf{Met}^{\uparrow}$ *point and distance classifier; the totally-bounded field is stable by Definition 127.4; and the complete-metric field is stable by Definition 128.4.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate hsame transport`
Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricSingleton certificate package`

Proof. Start with the metric carrier field. It supplies the source object, point classifier, distance rows, and metric-distance transport used by both subcertificates, so the public $\mathsf{CompactMetric}^{\uparrow}$ carrier can use the same histories and the same MetricDistanceWitness ledger policy.

For total boundedness, fix a transported positive tolerance history. The certificate carrier provides the corresponding closed ProbeBundle net, its bundled center histories, positive radius histories, membership rows, distance witnesses, and rational comparison ledgers. Apply Definition 127.4 to transport this finite-net row along the metric and $\mathsf{Rat}^{\uparrow}$ classifier data.

For completeness, fix a transported regular Cauchy stream name whose observations are carried by X. The certificate carrier supplies the modulus history, limit history, threshold ledgers, and late-observation distance bounds. Apply Definition 128.4 to transport these rows pointwise and recover the displayed limit witness for the transported stream.

The ledger policy for $\mathsf{CompactMetric}^{\uparrow}$ is exactly the union of these three visible packages: metric carrier data, ProbeBundle-net data at each positive tolerance, and complete-limit data for each regular Cauchy stream. Since each component has classifier transport and the shared source classifier is the metric one, the assembled fields satisfy the paper-level NameCert target for $\mathsf{CompactMetric}^{\uparrow}$. □

Theorem 129.6 (Compact metric certificate scope)**.** *For a certified metric carrier X, the source, carrier, classifier, stability, ledger, exactness, and status fields of* $\mathsf{CompactMetricCertificate}(X)$ *are exactly the fields exposed in Definition 129.1: the* $\mathsf{Met}^{\uparrow}$ *carrier and* MetricDistanceWitness *rows, the* TotallyBoundedProbeBundleNet *rows at positive* $\mathsf{Rat}^{\uparrow}$ *tolerances, and the* CompleteMetricLimitWitness *rows for regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream names.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate scope`

Proof. Unpack the three displayed fields of Definition 129.1. The first field supplies the source object, carrier predicate, point-history classifier, distance witness classifier, and exactness ledger used by the compact-metric package.

The second field supplies, for each visible positive tolerance, a closed ProbeBundle net together with its membership, center, radius, distance, and $\mathsf{Rat}^{\uparrow}$ comparison ledgers. Its stability clause is precisely Definition 127.4.

The third field supplies, for each regular Cauchy stream name, the modulus, limit, late-observation threshold, and distance-bound ledgers. Its stability clause is precisely Definition 128.4. There is no fourth field in the displayed carrier, so the compact-metric certificate scope is exactly this metric, finite-net, and limit-witness surface. □

Theorem 129.7 (Compact metric component stability)**.** *Let X be a certified metric carrier equipped with the* $\mathsf{Met}^{\uparrow}$ *point and distance classifier. Suppose the tolerance histories, stream observations, moduli, limit histories, distance histories, and every displayed comparison ledger in* $\mathsf{CompactMetricCertificate}(X)$ *are transported by* hsame *together with the* $\mathsf{Rat}^{\uparrow}$ *classifier. Then the transported rows again assemble a compact-metric certificate over the transported carrier, component by component.*

Proof. Start with the shared metric carrier. The transported point histories and distance rows remain under the same $\mathsf{Met}^{\uparrow}$ classifier, so both subcertificates read the same carrier predicate and the same MetricDistanceWitness transport policy.

For the finite-net component, fix a transported positive tolerance history. Apply Definition 127.4 to the corresponding TotallyBoundedProbeBundleNet row. This preserves the closed ProbeBundle spine, the visible coverage rows, and the $\mathsf{Rat}^{\uparrow}$ bounds against the transported tolerance.

For the complete-limit component, fix a transported regular Cauchy stream name. Apply Definition 128.4 to the corresponding CompleteMetricLimitWitness row. This preserves the modulus, transported limit, late-observation thresholds, metric distance rows, and $\mathsf{Rat}^{\uparrow}$ bounds for the transported stream.

Repack the three transported components using Definition 129.5. Since both subcertificates share the transported metric carrier classifier, their ledgers combine into a single transported $\mathsf{CompactMetric}^{\uparrow}$ certificate. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate component stability`

Theorem 129.8 (Compact metric downstream consumer boundary)**.** *Downstream* $\mathsf{ContinuousMap}^{\uparrow}$ *and Heine–Cantor consumers of* $\mathsf{CompactMetricCertificate}(X)$ *may use only the metric carrier, the finite* ProbeBundle*-net family, and the complete-limit family exposed by the compact-metric certificate. The certificate does not expose open-cover compactness, sequential compactness, or choice-based subsequence extraction.*

Proof. Read the exported ledger union from Definition 129.5. The public package contains metric carrier data, one finite ProbeBundle-net row for each positive tolerance, and one complete-limit witness row for each regular Cauchy stream name.

A $\mathsf{ContinuousMap}^{\uparrow}$ consumer can therefore inspect point histories, distance rows, finite net membership rows, and explicit limit-witness rows. A Heine–Cantor consumer can use the same rows to combine total boundedness, continuity control, and completeness through explicit certificates.

The same exported union contains no row naming open covers, no row selecting a finite subcover, no row asserting sequential compactness for arbitrary sequences, and no row extracting a subsequence by host-level choice. The negative boundary recorded in the $\mathsf{CompactMetric}^{\uparrow}$ closure status is therefore a consequence of the displayed certificate surface. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricDownstreamConsumer boundary`

Theorem 129.9 (Compact metric uniform modulus consumer route)**.** *Any* $\mathsf{UniformModulus}^{\uparrow}$ *or compact-uniform-continuity consumer reading a* $\mathsf{CompactMetric}^{\uparrow}$ *source must factor through the certified* $\mathsf{Met}^{\uparrow}$ *distance rows, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite* ProbeBundle*-net rows, the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness rows, and the* $\mathsf{Rat}^{\uparrow}$ *tolerance comparison ledgers already exposed by Definition 129.1. The consumer route may not replace those rows by open-cover compactness, sequential compactness, or a choice-based finite-subcover principle.*

Proof. First read the metric source. By Definition 129.2, every compact-metric consumer begins with the same $\mathsf{Met}^{\uparrow}$ carrier and the same MetricDistanceWitness ledgers.

Next read the finite-net component. The Definition 129.3 supplies, for each positive $\mathsf{Rat}^{\uparrow}$ tolerance, the displayed ProbeBundle net, membership rows, center rows, radius rows, metric coverage rows, and rational comparison ledgers used by a uniform-modulus request.

Finally read the complete-limit component and the public boundary. The Definition 129.4 supplies the $\mathsf{CompleteMetric}^{\uparrow}$ limit-witness rows for regular Cauchy streams, while Definition 129.7 transports all component rows under the same classifier. The exclusion clause is exactly Definition 129.8: the exported surface contains no open-cover, sequential-compactness, or choice-based subcover row. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCarrier uniform window consumer route`

Public export and downstream consumption

Definition 129.10 (Compact metric public interface)**.** *For a certified metric carrier X, the public compact-metric interface*

$$\mathsf{CompactMetricPublic}(X)$$

is the exported package consisting exactly of the three visible fields of Definition 129.1: the $\mathsf{Met}^{\uparrow}$ *carrier, classifier, and distance rows; the* $\mathsf{TotallyBounded}^{\uparrow}$ *positive-tolerance* ProbeBundle*-net selector $\varepsilon \mapsto (\mathcal{P}_\varepsilon, net_\varepsilon)$; and the* $\mathsf{CompleteMetric}^{\uparrow}$ *regular-Cauchy limit-witness selector $s \mapsto (\mu_s, \ell_s, L_s)$. No open-cover compactness witness, subsequence selector, or host-level compact-object predicate is part of* $\mathsf{CompactMetricPublic}(X)$*.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricTasteGate single carrier alignment`

Theorem 129.11 (Compact metric public interface stability)**.** *For a certified metric carrier X, the package* $\mathsf{CompactMetricPublic}(X)$ *is transported by the inherited metric classifier. The metric carrier and distance rows are carried by the* $\mathsf{Met}^{\uparrow}$ *classifier data; each positive-tolerance finite-net row is transported by Definition 127.4; and each regular-Cauchy limit-witness row is transported by Definition 128.4.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricPublicInterface stability`

Proof. Apply Definition 129.5 to view $\mathsf{CompactMetricPublic}(X)$ as the visible export of $\mathsf{CompactMetricCertificate}(X)$ The inherited metric classifier transports the carrier histories and distance rows, so the public metric field has the same source classifier as the certificate package.

For the totally-bounded selector, fix a transported positive $\mathsf{Rat}^{\uparrow}$ tolerance history. The certificate supplies the matching closed ProbeBundle net and its distance and comparison ledgers, and Definition 127.4 transports that whole row.

For the complete-metric selector, fix a transported regular Cauchy stream name. The certificate supplies the modulus, limit history, threshold rows, and late-observation distance bounds, and Definition 128.4 transports the displayed limit witness. These three transported fields are precisely $\mathsf{CompactMetricPublic}(X)$. □

Theorem 129.12 (Compact metric downstream consumption bridge)**.** *Any downstream* $\mathsf{ContinuousMap}^{\uparrow}$ *or Heine–Cantor argument whose compact input requirements are a metric carrier, explicit finite nets at positive rational tolerances, and regular-Cauchy limit witnesses may consume exactly the three fields of* $\mathsf{CompactMetricPublic}(X)$*. Such an argument gains no additional open-cover compactness principle and no subsequence principle from the compact-metric input.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricPublicInterface downstream consumption bridge`

Proof. Unpack Definition 129.10. A request for the carrier, classifier, or distance rows is routed to the $\mathsf{Met}^{\uparrow}$ field. This is the same metric source used by the total-boundedness and completeness clauses, so no extra carrier equivalence is introduced.

A request for a finite net at a positive rational tolerance is routed to the $\mathsf{TotallyBounded}^{\uparrow}$ selector $\varepsilon \mapsto (\mathcal{P}_\varepsilon, net_\varepsilon)$. Its centers, membership rows, distance witnesses, and rational comparison ledgers are the rows supplied by the certificate carrier.

A request for a limit witness for a regular Cauchy stream is routed to the $\mathsf{CompleteMetric}^{\uparrow}$ selector $s \mapsto (\mu_s, \ell_s, L_s)$. The modulus, limit history, threshold ledger, and late-distance bounds are exactly the rows available to the downstream proof.

The public interface has no fourth field. Therefore a downstream $\mathsf{ContinuousMap}^{\uparrow}$ or Heine–Cantor proof that uses $\mathsf{CompactMetricPublic}(X)$ consumes only metric rows, positive-tolerance ProbeBundle nets, and regular-Cauchy limit witnesses. □

Theorem 129.13 (CompactMetricUp metric source scope)**.** *For every certified metric carrier* X *equipped with* $\mathsf{CompactMetricCertificate}(X)$*, the public* $\mathsf{CompactMetric}^{\uparrow}$ *certificate exposes the same* $\mathsf{Met}^{\uparrow}$ *source carrier, point classifier, and* $\mathsf{MetricDistanceWitness}$ *rows used by the totally-bounded and complete-metric fields. The finite-net family and the complete-limit family are therefore both read over one metric classifier scope, with no independent carrier or distance package.*

Proof. Project the first field of Definition 129.1. It is already the $\mathsf{Met}^{\uparrow}$ carrier and distance package for X, so it supplies the source history object, point classifier, and distance-witness ledger rows.

The remaining certificate fields are indexed by the same carrier X and use the same point histories when they mention net centers, stream observations, and limit histories. Repack these rows under the projected classifier scope. This gives the displayed $\mathsf{CompactMetric}^{\uparrow}$ source data consumed by both the totally-bounded and complete-metric obligations. □

Theorem 129.14 (CompactMetricUp finite-net family coverage)**.** *For every certified metric carrier* X*, every* $\mathsf{CompactMetricCertificate}(X)$*, and every positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history* ε*, the certificate projects a* $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ *package. This package is stable under the metric carrier classifier and rational tolerance classifier transport of Definition 127.5 and Definition 127.4.*

Proof. Fix the positive tolerance row ε. Project the second field of Definition 129.1; it supplies the displayed $\mathsf{TotallyBoundedProbeBundleNet}$ package at that tolerance, including the ProbeBundle centers, membership rows, distance witnesses, and rational comparison ledgers.

The package is a $\mathsf{NameCert}_{\mathsf{TotallyBounded}^{\uparrow}}$ target by Definition 127.5. Apply Definition 127.4 to transport the net along the shared metric classifier and the transported positive $\mathsf{Rat}^{\uparrow}$ tolerance. The resulting row is the finite-net family coverage exported by the $\mathsf{CompactMetric}^{\uparrow}$ certificate. □

Theorem 129.15 (CompactMetricUp complete-limit family coverage)**.** *For every certified metric carrier* X*, every* $\mathsf{CompactMetricCertificate}(X)$*, and every regular* $\mathsf{StreamName}^{\uparrow}$ *Cauchy stream name* s *whose observations are carried by* X*, the certificate projects a* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$ *as in Definition 128.2. This witness is stable under classifier transport by Definition 128.4.*

Proof. Fix the regular Cauchy stream name s and its carrier row over X. Project the third field of Definition 129.1; it gives the modulus history, limit history, threshold ledgers, and late-observation distance bounds that form the displayed $\mathsf{CompleteMetricLimitWitness}$.

Apply Definition 128.4 pointwise to the stream observations, the modulus, the proposed limit, and the late-bound ledger rows. The transported rows again satisfy Definition 128.2, so the complete-limit family is available as a separate $\mathsf{CompactMetric}^{\uparrow}$ obligation site. □

Theorem 129.16 (Compact metric public downstream coverage)**.** *Let X be a certified metric carrier equipped with* $\mathsf{CompactMetricCertificate}(X)$*. Every accepted local consumer of the public packet* $\mathsf{CompactMetricPublic}(X)$ *is exhausted by the following visible rows:*

1. *the* $\mathsf{Met}^{\uparrow}$ *carrier, classifier, and* $\mathsf{MetricDistanceWitness}$ *rows;*
2. *for each positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, the finite* $\mathsf{ProbeBundle}$ *alternatives $\mathcal{P}_\varepsilon$, the* $\mathsf{InBundle}(p, \mathcal{P}_\varepsilon)$ *membership rows, the coverage rows in net_ε, and the displayed* $\mathsf{Rat}^{\uparrow}$ *comparison ledgers;*
3. *for each carried regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream name s, the modulus, limit history, threshold ledger, late-distance rows, and* $\mathsf{Rat}^{\uparrow}$ *comparison rows of* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$*.*

Consequently an open-cover compactness witness, finite-subcover selector, subsequence selector, or host-level compact-object predicate is not an accepted downstream alternative for the public compact-metric packet.

Proof. Unpack Definition 129.10. The public packet has one metric field, one positive-tolerance finite-net selector, and one regular-Cauchy limit-witness selector. Thus a local consumer accepted by this packet must ask for a row through one of these three displayed selectors.

The first selector is the inherited $\mathsf{Met}^{\uparrow}$ field, so carrier, classifier, and distance requests are exhausted by the point-history classifier and the $\mathsf{MetricDistanceWitness}$ ledger rows. By Definition 129.11, transport of this part is still the inherited metric transport; no independent carrier or distance package is introduced.

For a positive $\mathsf{Rat}^{\uparrow}$ tolerance history ε, apply Definition 129.14. The selected finite-net row is

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon).$$

Hence the finite alternatives available to a downstream consumer are exactly the bundled probes in the closed $\mathsf{ProbeBundle}$ spine $\mathcal{P}_\varepsilon$, read through their $\mathsf{InBundle}(p, \mathcal{P}_\varepsilon)$ membership rows. The same theorem identifies the coverage surface: every admitted source point is handled by a listed bundled center, a displayed metric-distance row in net_ε, and a displayed $\mathsf{Rat}^{\uparrow}$ comparison ledger against ε. Stability of the public interface keeps this finite-net row under the same metric carrier classifier and the same rational-tolerance classifier, so transport does not create an additional alternative.

For a carried regular Cauchy stream name s, the third public selector is the complete-limit row

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s).$$

By Definition 129.11, its admissible transport is precisely the componentwise complete-metric stability row. Thus a limit consumer can read only the named modulus, limit history, threshold rows, late-distance witnesses, and rational comparison ledgers already displayed in that witness.

These three cases exhaust the public packet. The defining export contains no fourth selector, and the two stability clauses cited above transport only the metric, finite-net, and limit-witness rows. Therefore open-cover compactness, finite-subcover selection, subsequence selection, and host-level compact-object predicates remain outside the accepted BEDC packet for $\mathsf{CompactMetricPublic}(X)$. □

Definition 129.17 (Public CompactMetricUp export source)**.** *For a certified metric carrier X, the public* $\mathsf{CompactMetric}^{\uparrow}$ *export source is the same shared carrier used in Definition 129.1: the* $\mathsf{Met}^{\uparrow}$ *point carrier and distance package for X, together with the* $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ *family for every positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, and the* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$ *family for every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream name s carried by X.*

The public surface exposes exactly these three component families as the compact-metric constructor source. It does not add an open-cover selector, subsequence extractor, or external compact set; downstream consumers receive the finite-net and complete-limit rows already certified by the component NameCerts.

Lean4 (definition) `BEDC.Derived.CompactMetricUp.CompactMetricPublicExportSource`

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricPublicExportSource component rows`

Theorem 129.18 (Public compact-metric constructor package)**.** *The export source of Definition 129.17 produces the public compact-metric constructor package for* $\mathsf{CompactMetric}^{\uparrow}$*. The package has the same metric carrier, the same positive-tolerance* ProbeBundle *net family, and the same regular-Cauchy limit-witness family as Definition 129.5; classifier transport is inherited componentwise from the metric, totally-bounded, and complete-metric certificates. The natural formal target name for this constructor package is* `BEDC.Derived.CompactMetricUp.CompactMetricPublicConstructor_pack`

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricPublicConstructor package`

Proof. Project the metric field from Definition 129.5. This gives the public source carrier, the point classifier, the distance rows, and the MetricDistanceWitness ledgers used by the constructor package.

For each positive $\mathsf{Rat}^{\uparrow}$ tolerance history ε, project the totally-bounded field. The projected row is already the closed ProbeBundle net $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$, and its stability is the componentwise transport supplied by Definition 127.4.

For each regular Cauchy $\mathsf{StreamName}^{\uparrow}$ stream name s, project the complete-metric field. The projected row is the limit witness $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$, transported by Definition 128.4.

Repack these three projected component families under the shared metric classifier. Since the public export source contains no additional fields, the componentwise transports assemble directly into the public $\mathsf{CompactMetric}^{\uparrow}$ constructor package. □

Theorem 129.19 (Heine–Cantor consumer bridge for CompactMetricUp)**.** *Let*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$$

be a metric $\mathsf{ContinuousMap}^{\uparrow}$ *consumer whose source metric carrier is the same BEDC carrier X as the export source of Definition 129.17. Then the public* $\mathsf{CompactMetric}^{\uparrow}$ *export supplies the compact finite-net input and complete-limit input required by the Heine–Cantor capstone Definition 9237.6. The finite-net input is the positive-tolerance* $\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$ *family, and the complete-limit input is the regular-Cauchy* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$ *family; both are consumed over the same metric distance rows as the continuous-map source certificate. The natural formal target name for this consumer bridge is* `BEDC.Derived.CompactMetricUp.CompactMetric`

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricHeineCantor consumer bridge`

Proof. Read the continuous-map source through Definition 130.29. The consumer therefore uses a categorical graph row, a modulus witness, and a MetricDistanceWitness row over the displayed source carrier.

Identify this source carrier with the metric carrier in the public CompactMetric$^{\uparrow}$ export. The metric classifier then transports the continuous-map distance row and the compact-metric distance row to the same BEDC history relation.

For a positive tolerance history ε, pass the exported ProbeBundle net family to the finite-cover input of the Heine–Cantor construction. This is exactly the located finite-net datum used by Definition 9237.3 and Definition 9237.4.

For a regular Cauchy stream name s, pass the exported complete-limit witness to the complete metric input. The capstone proof may then consume finite-net coverage and complete-limit convergence over the same source metric, so the uniform-modulus ledger is obtained from the public CompactMetric$^{\uparrow}$ export rather than from a private component package. □

Theorem 129.20 (Public compact-metric export surface)**.** *For a certified metric carrier X equipped with* CompactMetricCertificate(X)*, the public* CompactMetric$^{\uparrow}$ *surface exports exactly the metric carrier and distance rows of X, every finite-net row*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon),$$

and every complete-limit row

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s).$$

It does not export an open-cover compactness principle, a subsequence selection principle, or any host-level compact set predicate.

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate public export surface`

Proof. Project the carrier definition of Definition 129.1. The first component is the same Met$^{\uparrow}$ carrier, classifier, and distance witness family, so the public metric rows are copied without changing their histories or classifier ledgers.

For each positive tolerance history ε, the second component gives the displayed TotallyBoundedProbeBundleNet row. Its classifier transport and coverage stability are exactly the rows supplied by Definition 127.4.

For each regular Cauchy stream name s, the third component gives the displayed CompleteMetricLimitWitness row. Its stability is the componentwise transport supplied by Definition 128.4. Since the definition has no further compactness field, the public surface consists precisely of these metric, total-bounded, and complete subcertificates. □

Theorem 129.21 (Continuous-map consumption boundary)**.** *Any* ContinuousMap$^{\uparrow}$ *construction consuming a public* CompactMetric$^{\uparrow}$ *certificate may use only the exported metric classifier, the exported distance witnesses, and the finite* TotallyBoundedProbeBundleNet *rows. It receives no open-cover row, no subsequence compactness row, and no unlisted compact-set eliminator from* CompactMetricCertificate(X)*.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate continuousmap consumption boundary`

Proof. By Definition 129.20, the visible public rows are exactly the metric carrier rows, the finite ProbeBundle-net family, and the complete-limit family. The continuous-map interface can therefore read the point classifier, distance witnesses, and finite-net center and coverage ledgers directly from the exported certificate.

The same projection also proves the boundary. The certificate carrier in Definition 129.1 contains no open-cover predicate, no subsequence selector, and no compact-set eliminator. Hence a $\mathsf{ContinuousMap}^{\uparrow}$ argument consuming this public surface must factor through the displayed metric and total-bounded rows rather than through any classical compactness principle. □

Theorem 129.22 (Heine–Cantor input boundary)**.** *The compact-uniform-continuity capstone may consume a public* $\mathsf{CompactMetric}^{\uparrow}$ *certificate by using the finite* ProbeBundle*-net family together with continuous pointwise modulus ledgers to assemble a uniform-modulus ledger. Completeness enters only through the named* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness field* $\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$.

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate heine cantor input boundary`

Proof. Start from the public export theorem. For each positive tolerance, the $\mathsf{CompactMetric}^{\uparrow}$ surface supplies the finite TotallyBoundedProbeBundleNet row, including bundled centers, membership rows, distance witnesses, and rational comparison ledgers. Apply Definition 127.4 to transport these rows along the metric classifier data used by the capstone.

The continuous input contributes pointwise modulus ledgers at the exported centers. The finite net supplies only finitely many such center rows, so the capstone can combine their visible tolerance ledgers into one uniform-modulus ledger without invoking open covers or subsequences.

Completeness is used only at the point where a regular Cauchy stream requires a named limit witness. That row is exactly the CompleteMetricLimitWitness exported from the certificate, with transport governed by Definition 128.4. No additional compactness principle is consumed. □

Theorem 129.23 (Compact metric carrier-field projection)**.** *Let X be a certified metric carrier equipped with* $\mathsf{CompactMetricCertificate}(X)$*. Then the compact-metric certificate exposes the same* $\mathsf{Met}^{\uparrow}$ *point carrier, the same* MetricDistanceWitness *distance rows, and the same point-classifier transport consumed by Definition 129.5.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate metric field projection`

Proof. Unpack Definition 129.1. Its first field is the metric carrier interface for X, including the accepted point histories, the distance package, and the classifier transport on point histories.

Repack exactly this field as the metric face of the compact-metric certificate. No total-bounded net row or complete-limit row is used in this projection, so the exported carrier, distance ledger, and classifier data are the same rows used by the bundled package theorem. □

Theorem 129.24 (Compact metric total-bounded-field projection)**.** *Let X be a certified metric carrier equipped with* $\mathsf{CompactMetricCertificate}(X)$*. For every positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, the certificate exposes a face*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon)$$

stable under $\mathsf{Rat}^{\uparrow}$ *tolerance transport and metric classifier transport, with coverage stability supplied by Definition 127.4.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate total bounded field projection`

Proof. Unpack the second field of Definition 129.1 at the chosen positive tolerance history ε. It provides the closed ProbeBundle spine $\mathcal{P}_\varepsilon$, the net ledger net_ε, and the BEDC rows recording centers, membership, distance witnesses, and rational bounds against ε.

To transport this face, move the tolerance through the $\mathsf{Rat}^{\uparrow}$ classifier and move the point, center, and distance histories through the metric classifier. Apply Definition 127.4 to repack the transported coverage row, while the closed ProbeBundle carrier and its InBundle membership constructors remain the displayed finite spine. □

Theorem 129.25 (Compact metric complete-field projection)**.** *Let X be a certified metric carrier equipped with* $\mathsf{CompactMetricCertificate}(X)$*. Every regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream s carried by X exposes a face*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s),$$

and this face is stable under hsame *transport of the stream, modulus, and limit, with stability supplied by Definition 128.4.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate complete field projection`

Proof. Unpack the third field of Definition 129.1 at the regular Cauchy stream name s. It provides the modulus history μ_s, the limit history ℓ_s, and the late-observation ledger family L_s recording the displayed MetricDistanceWitness rows and $\mathsf{Rat}^{\uparrow}$ bounds.

Under hsame transport of the stream observations, the modulus, and the limit, the component rows are carried pointwise. Apply Definition 128.4 to repack the transported threshold ledgers, distance witnesses, and comparison rows as the same complete-limit face for the transported stream data. □

Theorem 129.26 (CompactMetric metric source obligation)**.** *Let X be a compact-metric certificate carrier in the sense of Definition 129.1. Its metric source projection is exactly the displayed* $\mathsf{Met}^{\uparrow}$ *carrier on X: the point classifier, the* MetricDistanceWitness *distance rows, and classifier transport are the ones supplied by the metric carrier field. The source projection contains no open-cover predicate, ambient compact subset predicate, finite-subcover principle, or host compactness assertion.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricCertificate metric source obligation`

Proof. Read the first field of Definition 129.1. It is already the certified $\mathsf{Met}^{\uparrow}$ source for X, with the point classifier, the distance rows, and the transport rule for those rows displayed before the totally-bounded and complete-metric fields are considered.

Now apply Definition 129.5. The package theorem uses that metric source as the public classifier for $\mathsf{CompactMetric}^{\uparrow}$, so the source field of the ten-field NameCert template is filled by the same carrier and the same MetricDistanceWitness ledger. Since the carrier definition lists only metric data, ProbeBundle nets, and limit witnesses, no open-cover or host compact-set row can be projected from this source. □

Theorem 129.27 (CompactMetric ProbeBundle net-family obligation)**.** *Let X be a compact-metric certificate carrier. For every carried positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, the totally-bounded component supplies the existing package*

$$\mathsf{TotallyBoundedProbeBundleNet}(X, \varepsilon, \mathcal{P}_\varepsilon, net_\varepsilon).$$

Transport of this family is exactly the componentwise transport provided by Definition 127.4.

Proof. Unpack the second field of Definition 129.1. For each positive tolerance history it gives a closed ProbeBundle spine $\mathcal{P}_\varepsilon$, center histories, membership rows, distance witnesses, and $\mathsf{Rat}^\uparrow$ comparison ledgers in the package named in the statement.

The transport clause is then read through Definition 127.4. That theorem keeps the same metric carrier interface and transports the tolerance, membership, center, distance, and rational-bound rows along the source classifiers.

Finally repack these rows into the $\mathsf{CompactMetric}^\uparrow$ ten-field NameCert template by Definition 129.5. The resulting obligation is only the tolerance-indexed finite-net family; it does not add a finite-subcover choice principle or any open-cover surface. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricProbeBundleNetFamily obligation`

Theorem 129.28 (CompactMetric limit-witness family obligation)**.** *Let X be a compact-metric certificate carrier. Every carried regular Cauchy* $\mathsf{StreamName}^\uparrow$ *stream s has the complete-metric witness*

$$\mathsf{CompleteMetricLimitWitness}(X, s, \mu_s, \ell_s, L_s)$$

of Definition 128.2. Transport of these witnesses is the pointwise transport supplied by Definition 128.4.

Proof. Unpack the third field of Definition 129.1. For a carried regular Cauchy stream it supplies the modulus history μ_s, the carried limit history ℓ_s, the threshold rows, the late-observation distance rows, and the $\mathsf{Rat}^\uparrow$ budget comparisons required by Definition 128.2.

Apply Definition 128.4 to the displayed stream, modulus, limit, threshold, distance, and rational comparison rows. The transport remains internal to the same metric carrier classifier and does not use a host completion or a choice of subsequence.

Repack the transported witness through Definition 129.5. This fills the complete-metric part of the ten-field NameCert template for $\mathsf{CompactMetric}^\uparrow$ while leaving all convergence data visible as the limit-witness ledger. □

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricLimitWitnessFamily obligation`

Theorem 129.29 (CompactMetric ledger exactness obligation)**.** *For a compact-metric certificate carrier X, the* $\mathsf{CompactMetric}^\uparrow$ *ledger is exactly the union of the metric-distance rows of* $\mathsf{Met}^\uparrow$*, the tolerance-indexed finite-net rows of* $\mathsf{TotallyBounded}^\uparrow$*, and the regular-stream limit-witness rows of* $\mathsf{CompleteMetric}^\uparrow$*. Coverage, soundness, and completeness for the exported certificate are read componentwise over these three sources.*

Lean4 ✓ `BEDC.Derived.CompactMetricUp.CompactMetricLedger exactness obligation`

Proof. Start from the carrier inventory in Definition 129.1. It lists one metric source field, one ProbeBundle-net family indexed by positive $\mathsf{Rat}^\uparrow$ tolerances, and one limit-witness family indexed by carried regular Cauchy streams. These are the only rows available to the compact-metric package.

Coverage is componentwise. The metric source covers point classification and distance witnessing; the totally-bounded component covers each carried point by a bundled center at a displayed tolerance; and the complete-metric component covers each carried regular Cauchy stream by its displayed limit witness.

Soundness is also componentwise. Metric rows are compared by the $\mathsf{Met}^\uparrow$ classifier, finite-net rows by Definition 127.4, and limit rows by Definition 128.4. Each comparison stays inside the classifier and ledger named by its source component.

Completeness follows by repacking all three inventories through Definition 129.5. Any exported compact-metric row therefore comes from one of the displayed metric, finite-net, or limit-witness rows, and every such row is accepted by the ten-field NameCert template for CompactMetric$^{\uparrow}$. □

Closure status for NameCert$_{\text{CompactMetric}^{\uparrow}}$.

Constructive story: A compact metric carrier is assembled by keeping one certified Met$^{\uparrow}$ carrier and attaching exactly two visible families to it: a TotallyBounded$^{\uparrow}$ ProbeBundle net at every positive Rat$^{\uparrow}$ tolerance and a CompleteMetric$^{\uparrow}$ limit witness for every carried regular StreamName$^{\uparrow}$ Cauchy stream. A row is admitted only when the metric carrier, finite-net family, and limit-witness family are all present over the same point histories; open covers, subsequence selections, or ambient compact subsets are not carrier data. The classifier is inherited componentwise from the metric point classifier, the ProbeBundle-net transport rows, and the complete-limit transport rows, so two compact-metric packages agree only by transporting those displayed metric, net, and convergence ledgers. The source is the shared metric carrier, the pattern is the conjunction of total boundedness and completeness, and the ledger is the union of distance rows, positive-tolerance finite-net rows, membership and coverage rows, regular-stream moduli, carried limits, and late-observation bounds. The five name-certificate obligations are obtained by the singleton compact metric package for inhabitation, by the equivalence behavior of the inherited classifiers, and by the component-stability theorems that repack transported finite-net and limit-witness rows under the same metric carrier. A package h lies in the signature gap of a CompactMetric$^{\uparrow}$ token p when the source-domain accepts its metric carrier with both component families, the signature relation pairs that triple with an answer compact-metric package and evidence ledger, and the token-introduction relation names the answer as p; unfolded, p encodes precisely the metric rows, tolerance-indexed nets, and Cauchy-limit witnesses. The public bridge lets continuous-map and Heine–Cantor consumers read the same finite-net and complete-limit data, and the chapter registers this componentwise construction under CompactMetric$^{\uparrow}$ without adding open-cover compactness, sequential compactness, products, or function-space compactness.

Theory closure: TheoryGate(NameCert$_{\text{CompactMetric}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope consists of the public compact-metric export surface, the public compact-metric constructor package, the Heine–Cantor consumer bridge, the metric/ProbeBundle/limit-witness projections, and the ledger exactness obligation.

Formal verification: FormalStatus(NameCert$_{\text{CompactMetric}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CompactMetricUp.CompactMetricSingleton\_certificate\_package`

Bridge status: paperBridge

Not claimed: Open-cover compactness, sequential compactness without explicit data, Tychonoff-style products, or compactness of function spaces.

Upgrade path: No stronger theory-closure grade is claimed here; bridgeChecked remains the Lean-side bridge verification target.

The ContinuousMap$^{\uparrow}$ entry point reads its metric distance witness through chapter 57, its pointwise-modulus rows through chapter 59, and its categorical graph and composition rows through chapter 61; compact-uniform consumers read the resulting horizontal lattice through the child files below.

Chapter 130

A Metric-Carrier Interface for $\mathsf{ContinuousMap}^{\uparrow}$

The continuous-map interface $\mathsf{ContinuousMap}^{\uparrow}$ packages a function between metric carriers together with a pointwise modulus of continuity. It depends on $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ (chapter 57) and lifts $\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$ (chapter 59) to general L10 metric carriers.

130.1 Carrier definition

Definition 130.1 (ContinuousMap carrier)**.** *For histories* $source, map, target, modulus, cert, dist$: BHist, $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, dist)$ *is the conjunction of the continuous-function carrier* $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert)$ *and the metric distance witness* $\mathsf{MetricDistanceWitness}(source, target, dist)$.

Lean4 (definition) `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier`

Theorem 130.2 (ContinuousMap carrier category-metric decomposition)**.** *For histories* $source, map, target, modu$ BHist, *the carrier splits exactly into a categorical hom row, a target-side continuous-modulus row, and a metric-distance row:*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$$
$$\iff \mathsf{CategoryHomCarrier}(source, target, map) \wedge \mathsf{ContinuousModulusWitness}(target, modulus, cert) \wedge \mathsf{MetricD}$$

Proof. For the forward direction, unpack $\mathsf{ContinuousMapCarrier}$ into its continuous-function component and its metric-distance component. The continuous-function fields provide unary source, target, and map histories together with the graph continuation, hence the displayed $\mathsf{CategoryHomCarrier}(source, target$ The same continuous-function component also exposes the target-modulus continuation, which is precisely $\mathsf{ContinuousModulusWitness}(target, modulus, cert)$. The remaining projection is the displayed metric-distance witness.

For the reverse direction, unpack the category hom row into unary source, target, and map histories plus the graph continuation. Unpack the modulus row into unary target, modulus, and certificate histories plus the terminal continuation. These two rows rebuild $\mathsf{ContinuousFunctionCarrier}(source, map, tar$ and pairing that carrier with the supplied $\mathsf{MetricDistanceWitness}(source, target, distance)$ gives the claimed $\mathsf{ContinuousMapCarrier}$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier category metric decomposition`

The ten-field fit is: traditional target continuous maps as categorical morphisms with a metric ledger; BEDC source CategoryUp/ContinuousUp/MetricUp/ContinuousMapUp; carrier ContinuousMapCarrier; classifier hsame; stability inherited from CategoryHomCarrier, ContinuousModulusWitness, and MetricDistanceWitness ledger map, modulus, certificate, and distance histories; core theorem Theorem 130.2; exactness the displayed equivalence; standard bridge is deferred; status ProofTarget.

Theorem 130.3 (ContinuousMap visible-context category-metric decomposition)**.** *For histories* $p, q, source, map, target, modulus, cert, distance$: BHist*, the displayed carrier with visible continuous-function context and prefixed metric ledger is equivalent to unary context evidence together with the central categorical, continuity, and metric rows:*

$$\mathsf{ContinuousMapCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q, p \cdot distance)$$
$$\iff \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{CategoryHomCarrier}(source, target, map) \wedge \mathsf{ContinuousModulusWitn}$$

Proof. For the forward direction, unpack ContinuousMapCarrier into its visible continuous-function component and its metric-distance component. Apply Definition 59.169 to the function row to recover unary evidence for p and q together with the central continuous-function carrier.

Apply Definition 59.139 to that central carrier. Its graph row is exactly $\mathsf{CategoryHomCarrier}(source, target, map)$ and its target-modulus row is $\mathsf{ContinuousModulusWitness}(target, modulus, cert)$. Apply Definition 57.11 to the metric row to read back the central $\mathsf{MetricDistanceWitness}(source, target, distance)$.

For the reverse direction, start with the two unary context histories, the central categorical row, the target-modulus witness, and the central metric witness. The categorical row supplies the graph continuation and unary source, target, and map histories.

The modulus row supplies the unary modulus and certificate histories plus the certificate continuation, so the central continuous-function carrier is reassembled.

Apply Definition 59.169 from right to left to restore the visible source, target, modulus, and certificate ledgers. Apply Definition 57.11 from right to left to restore the prefixed metric ledger, and pair the two rows into the displayed ContinuousMapCarrier. □

The ten-field fit is: traditional continuous maps carried with explicit visible context on the continuous-function row and an exposed metric ledger; BEDC source CategoryUp/ContinuousUp/MetricUp/Continuo carrier ContinuousMapCarrier; classifier hsame; stability from visible continuous-function context, category hom readback, modulus readback, and metric prefix exactness; ledger visible contexts, map, modulus, certificate, and distance histories; core theorem Theorem 130.3; exactness the displayed equivalence; standard bridge is deferred; status ProofTarget.

Theorem 130.4 (ContinuousMap canonical distance carrier)**.** *For histories* $source, map, target, modulus, cert$: BHist*, the carrier with canonical distance ledger* $\mathsf{append}(source, target)$ *is exactly the underlying continuous-function carrier:*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, \mathsf{append}(source, target)) \iff \mathsf{ContinuousFunctionCarrier}($$

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier canonical distance iff`

Proof. The forward direction projects the function-carrier component. Conversely, the function carrier supplies unary source and target histories; unary append closure and the reflexive continuation equation build the canonical metric distance witness. □

Theorem 130.5 (ContinuousMap carrier empty distance exactness)**.** *For histories* $source, map, modulus, cert$: BHist*,*

$$\mathsf{ContinuousMapCarrier}(source, map, \mathsf{e}, modulus, cert, \mathsf{e})$$
$$\iff \mathsf{UnaryHistory}(source) \wedge \mathsf{UnaryHistory}(map) \wedge \mathsf{UnaryHistory}(modulus) \wedge source \sim \mathsf{e} \wedge map \sim \mathsf{e} \wedge cert \sim m$$

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty distance exact`

Proof. Unpack the carrier into the continuous-function row and the metric-distance row. Empty distance forces the source endpoint to be e, while the continuous-function graph row into e forces the map component to be e. The remaining certificate row is the left-unit continuation from e to $modulus$, hence $cert \sim modulus$. Conversely, the displayed unary and boundary data build the two continuation rows and the empty-distance metric witness directly. □

Theorem 130.6 (ContinuousMap empty map with empty distance)**.** *For histories source, modulus, and cert, the empty graph map at the source together with an empty metric distance witness is exact for the source being empty and the modulus ledger being certified:*

$$\mathsf{ContinuousFunctionCarrier}(source, \mathsf{e}, source, modulus, cert) \wedge \mathsf{MetricDistanceWitness}(source, source, \mathsf{e})$$
$$\iff \mathsf{ContinuousModulusWitness}(source, modulus, cert) \wedge \mathsf{hsame}(source, \mathsf{e}).$$

Proof. From the carrier, unpack the target-certificate continuation and the unary source and modulus ledgers; unary continuation closure supplies the certificate ledger, hence the displayed modulus witness. The empty-distance witness forces the source endpoint to be empty. Conversely, the modulus witness builds the empty-map identity carrier, and the empty source endpoint builds the empty metric distance witness. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty map empty distance iff`

Theorem 130.7 (ContinuousMap empty certificate exactness)**.** *For histories* $source, map, target, modulus, dist$: BHist,

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, \mathsf{e}, dist)$$

holds exactly when

$$source\mathsf{hsamee}, \quad map\mathsf{hsamee}, \quad target\mathsf{hsamee}, \quad modulus\mathsf{hsamee}, \quad dist\mathsf{hsamee}.$$

Proof. The function-carrier half gives empty source, map, target, and modulus by the empty-certificate theorem for $\mathsf{ContinuousFunctionCarrier}$. With source and target reduced to e, the metric-distance continuation is compared with the right unit continuation, forcing the displayed distance to be empty. The reverse direction reassembles the function witness and the empty distance witness after the five endpoint equalities are exposed. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty cert iff`

Theorem 130.8 (Empty identity maps reflect metric witnesses)**.** *For histories* $x, y, x', y', d, m_x, m_y, c_x, c_y$: BHist, *if* $\mathsf{ContinuousFunctionCarrier}(x, \mathsf{e}, x', m_x, c_x)$ *and* $\mathsf{ContinuousFunctionCarrier}(y, \mathsf{e}, y', m_y, c_y)$ *are empty-map continuous carriers, then every displayed metric witness* $\mathsf{MetricDistanceWitness}(x', y', d)$ *reflects to a source-side witness* $\mathsf{MetricDistanceWitness}(x, y, d)$.

Proof. The empty-map carrier readback identifies x' with x and y' with y. Unary endpoint membership and the continuation field of the metric witness transport along these two hsame witnesses, while the distance history is held fixed. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty identity metric witness reflects`

Theorem 130.9 (Empty identity maps transport metric witnesses exactly)**.** *For histories* $x, y, x', y', d, m_x, m_y, c_x, c$ BHist*, if* ContinuousFunctionCarrier$(x, \mathsf{e}, x', m_x, c_x)$ *and* ContinuousFunctionCarrier$(y, \mathsf{e}, y', m_y, c_y)$ *are empty-map continuous carriers, then the image-side and source-side metric witnesses are equivalent:*

$$\mathsf{MetricDistanceWitness}(x', y', d) \iff \mathsf{MetricDistanceWitness}(x, y, d).$$

Proof. For the forward direction, apply Theorem 130.8 to the two empty-map continuous carriers and the displayed image-side metric witness.

For the reverse direction, apply Theorem 59.13 to the two carriers. It reads back $x'\mathsf{hsame}x$ and $y'\mathsf{hsame}y$; symmetry orients these witnesses as $x\mathsf{hsame}x'$ and $y\mathsf{hsame}y'$. Combine them with reflexivity $d\mathsf{hsame}d$ and apply Theorem 57.347 to transport the source-side metric fields. Repacking those transported unary fields and the transported continuation ledger gives MetricDistanceWitness(x', y', d). □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty identity metric witness iff`

The ten-field fit is: traditional target empty continuous-map metric readback, BEDC source ContinuousMapUp, carrier ContinuousFunctionCarrier, classifier hsame, empty-map graph ledger e, distance ledger d, metric certificate MetricDistanceWitness, core theorem Theorem 130.9, exactness the displayed iff, and downstream consumer compact-continuous metric readback.

Theorem 130.10 (Empty identity maps preserve metric result determinacy)**.** *For histories* $x, y, x', y', d, d', m_x, m_y$ BHist*, if the two endpoints are carried by empty continuous maps, then an image-side metric witness* MetricDistanceWitness(x', y', d) *and a source-side metric witness* MetricDistanceWitness(x, y, d') *have* hsame *distance results.*

Proof. First reflect the image-side witness through the empty continuous maps to get MetricDistanceWitness(x, y, d). Metric witness result determinacy at reflexive source endpoints then compares the two displayed distance histories. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty identity metric result deterministic`

Theorem 130.11 (ContinuousMap empty-target metric rebased carrier)**.** *For histories* $source, modulus, cert, dist,$ *suppose*

ContinuousFunctionCarrier$(source, \mathsf{e}, source, modulus, cert)$, MetricDistanceWitness$(source, \mathsf{e}, dist)$, Cont(

Then the metric distance ledger itself supports the empty-map continuous function carrier

$$\mathsf{ContinuousFunctionCarrier}(dist, \mathsf{e}, dist, modulus, cert').$$

Proof. Unpack the continuous-function carrier and the metric distance witness. The metric witness supplies unary membership of $dist$, while the continuous carrier supplies unary membership of $modulus$. The empty map is unary, the right-unit continuation gives Cont$(dist, \mathsf{e}, dist)$, and the displayed continuation gives the terminal certificate row. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty target metric rebased carrier`

Theorem 130.12 (ContinuousMap empty-target metric certificate determinacy)**.** *For histories* $source, modulus, cert, dist, cert',$ *suppose*

ContinuousFunctionCarrier$(source, \mathsf{e}, source, modulus, cert)$, MetricDistanceWitness$(source, \mathsf{e}, dist)$, Cont(

Then the original continuous-function certificate and the metric-rebased certificate classify together:

$$\mathsf{hsame}(cert, cert').$$

Proof. The empty-right metric witness identifies *dist* with *source* while preserving the unary distance ledger. The original carrier supplies the continuation $\mathsf{Cont}(source, modulus, cert)$; transporting the source side across the metric identification and comparing it with $\mathsf{Cont}(dist, modulus, cert')$ gives $\mathsf{hsame}(cert, cert')$ by continuation determinacy under hsame. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap empty target metric cert deterministic`

Theorem 130.13 (ContinuousMap target, certificate, and distance determinacy)**.** *For histories source, map, target, target′, modulus, cert, cert′, distance, and distance′, two concrete continuous-map carriers with the same source, map, and modulus determine the target, certificate, and distance ledgers up to* hsame*:*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance) \Longrightarrow \mathsf{ContinuousMapCarrier}(source, map, targe$$
$$\Longrightarrow \mathsf{hsame}(target, target') \wedge \mathsf{hsame}(cert, cert') \wedge \mathsf{hsame}(distance, distance') \wedge \mathsf{Cont}(source, target, distance).$$

Proof. The continuous-function components have the same source, map, and modulus, so continuous-function target-certificate determinacy gives $\mathsf{hsame}(target, target')$ and $\mathsf{hsame}(cert, cert')$. The two metric distance witnesses then share source histories and target histories up to hsame; metric result determinacy identifies *distance* and *distance′*. The continuation ledger $\mathsf{Cont}(source, target, distance)$ is the final field of the left metric distance witness. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier target cert distance deterministic`

Theorem 130.14 (ContinuousMap carrier canonical distance exactness)**.** *For histories source, map, target, modul* BHist*, the concrete continuous-map carrier is exactly the continuous-function carrier together with the canonical source-target distance classifier:*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance) \Longleftrightarrow \mathsf{ContinuousFunctionCarrier}(source, map, t$$

Proof. In the forward direction, project the continuous-function carrier from the given continuous-map carrier. Theorem 130.4 then supplies the carrier with distance ledger $\mathsf{append}(source, target)$, and Theorem 130.13 compares the two carriers with the same source, map, target, modulus, and certificate, identifying *distance* with $\mathsf{append}(source, target)$.

Conversely, start from the displayed continuous-function carrier and the classifier $\mathsf{hsame}(distance, \mathsf{append}(sourc$ Theorem 130.4 builds the canonical-distance carrier. Its metric component contains the continuation $\mathsf{Cont}(source, target, \mathsf{append}(source, target))$ and unary membership of the canonical distance ledger. Transport the continuation by BEDC.Derived.MetricUp.MetricDistanceWitness_cont_hsame_transport, transport unary membership of the distance ledger along hsame, and repack these fields as the required $\mathsf{MetricDistanceWitness}(source, target, distance)$. Together with the original continuous-function carrier this is precisely $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$.

The ten-field fit is: traditional target normalized continuous-map distance ledger; BEDC source ContinuousFunctionCarrier, MetricDistanceWitness, and Cont; carrier ContinuousMapCarrier; classifier hsame; stability obligations distance determinacy and metric hsame transport; ledger the hidden distance history; core theorem this iff; exactness the displayed iff; standard bridge deferred; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier canonical distance exactness`

Theorem 130.15 (ContinuousMap categorical canonical distance exactness)**.** *For histories source, map, target, m*
BHist*, the continuous-map carrier is exactly the composite classifier obtained from the categorical source-target hom row, the target-side continuity-modulus row, and the canonical source-target distance ledger:*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$$
$$\iff \mathsf{CategoryHomCarrier}(source, target, map) \wedge \mathsf{ContinuousModulusWitness}(target, modulus, cert) \wedge \mathsf{hsame}(d$$

Proof. In the forward direction, apply Theorem 130.2 to the given carrier. This yields the categorical hom row and the target-side modulus row, together with the metric distance witness.

Apply Theorem 130.14 to the same carrier. Its distance component replaces the metric witness by the canonical classifier $\mathsf{hsame}(distance, \mathsf{append}(source, target))$, so the displayed three-part conjunction follows.

Conversely, unpack $\mathsf{CategoryHomCarrier}(source, target, map)$ into the unary source, target, and map histories plus the graph continuation. Unpack $\mathsf{ContinuousModulusWitness}(target, modulus, cert)$ into the target, modulus, and certificate histories plus the modulus continuation. These rows rebuild $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert)$.

Now use Theorem 130.14 in the reverse direction with the rebuilt continuous-function carrier and the displayed classifier $\mathsf{hsame}(distance, \mathsf{append}(source, target))$. The result is precisely $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$.

The ten-field fit is: traditional target continuous maps as morphisms with a canonical endpoint distance ledger; BEDC source CategoryUp/ContinuousUp/MetricUp/ContinuousMapUp; carrier ContinuousMapCarrier; classifier hsame; stability supplied by the categorical hom row, the modulus witness, and canonical distance exactness; ledger source, target, map, modulus, certificate, and distance histories; core theorem the displayed equivalence; exactness replacement of the metric witness by $\mathsf{hsame}(distance, \mathsf{append}(source, target))$; standard bridge is the same morphism-layer reading; downstream consumer $\mathsf{ContinuousMap}^{\uparrow}$ as the morphism layer for compact-continuous certificates. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier categorical canonical distance exactness`

Theorem 130.16 (ContinuousMap composition category-metric package)**.** *For histories source, mid, target, mapl*
BHist*, assume the two concrete continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, distG),$$

and assume the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

Then the composite exposes a category-metric package:

1. *the canonical composite continuous-map carrier,*

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{append}(source, target));$$

2. *the composite categorical hom row,*

$$\mathsf{CategoryHomCarrier}(source, target, mapFG);$$

3. the composite target-side modulus row,

$$\mathsf{ContinuousModulusWitness}(target, modFG, certFG);$$

4. the canonical metric distance row,

$$\mathsf{MetricDistanceWitness}(source, target, \mathsf{append}(source, target));$$

5. every displayed composite carrier with the same visible fields has the canonical distance ledger,

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, displayed) \Rightarrow \mathsf{hsame}(displayed, \mathsf{append}(source,$$

Proof. Project the continuous-function halves from the two ContinuousMapCarrier hypotheses. Apply Theorem 59.7 to those halves and the three continuation ledgers, obtaining the composite $\mathsf{ContinuousFunctionCarrier}(source, mapFG, target, modFG, certFG)$.

Package the canonical source-target distance by applying Theorem 130.4 from right to left to this composite continuous-function carrier. This gives the first package row,

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{append}(source, target)).$$

Apply Theorem 130.2 to the canonical composite carrier. Its three projections are exactly the displayed CategoryHomCarrier row, the ContinuousModulusWitness row, and the canonical MetricDistanceWitness row.

For the final determinacy clause, compare any displayed composite carrier with the canonical composite carrier by Theorem 130.13. The two carriers have the same source, map, target, modulus, and certificate histories, so the distance component of the determinacy conclusion is precisely $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier composition category metric package`

Theorem 130.17 (ContinuousMap carrier hsame-field transport with depth ledger)**.** *For histories* $source, source', map, map', target, target', modulus, modulus', cert, cert', distance, distance'$: BHist*, suppose*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$$

and

$$\mathsf{hsame}(source, source') \wedge \mathsf{hsame}(map, map') \wedge \mathsf{hsame}(target, target') \wedge \mathsf{hsame}(modulus, modulus') \wedge \mathsf{hsame}(cert, cert')$$

hold. Then the transported histories again form a continuous-map carrier, and the transported distance exposes the endpoint continuation and depth ledger:

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(source', map', target', modulus', cert', distance')\\ &\wedge\ \mathsf{Cont}(source', target', distance')\\ &\wedge\ \mathsf{MetricDistanceDepth}(distance') = \mathsf{MetricDistanceDepth}(source') + \mathsf{MetricDistanceDepth}(target').\end{aligned}$$

Proof. Unpack the displayed ContinuousMapCarrier into its ContinuousFunctionCarrier half and its MetricDistanceWitness half. Apply Definition 59.14 to the five function fields $source, map, target, modulus, cert$ and their displayed hsame images. This gives the transported continuous-function half.

For the metric half, apply Definition 57.347 to $source, target, distance$ and $source', target', distance'$. The resulting unary fields and transported continuation repack as $\mathsf{MetricDistanceWitness}(source', target', distance')$

Apply Definition 57.348 to the same three hsame witnesses and the original metric witness. It supplies the displayed $\mathsf{Cont}(source', target', distance')$ and the displayed depth equation. Pairing the transported continuous-function half with the transported metric witness gives the carrier component; pairing this with the continuation and depth ledger gives the stated conjunction.

The ten-field fit is: traditional target stable uniformly continuous map under classifier transport; BEDC source ContinuousFunctionCarrier, MetricDistanceWitness, and hsame; carrier ContinuousMapCarrier; classifier hsame; stability obligation six-field transport plus transported metric depth ledger; ledger hidden distance history; core theorem this block; exactness transported carrier and distance-depth equation; standard bridge deferred; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier hsame field transport depth`

Theorem 130.18 (ContinuousMap function carrier metric graph exactness)**.** *For histories $source, map, target, mo$ BHist, if*

$$\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert)$$

and

$$\mathsf{MetricDistanceWitness}(source, map, dist)$$

both hold, then the carrier exposes the graph continuation and the metric distance endpoint is exactly the same history as the graph target:

$$\mathsf{Cont}(source, map, target) \wedge \mathsf{hsame}(dist, target).$$

Proof. Read back the continuous-function carrier to obtain its graph continuation $\mathsf{Cont}(source, map, target)$. The metric witness supplies the continuation $\mathsf{Cont}(source, map, dist)$ over the same source and map. Continuation determinacy identifies the two endpoints, giving $\mathsf{hsame}(dist, target)$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapFunctionCarrier metric graph exactness`

Theorem 130.19 (ContinuousMap empty map with empty distance boundaries)**.** *For histories source, target, modulus, and cert, a concrete continuous-map carrier with empty graph map and empty distance ledger is equivalent to empty source and target boundaries together with the empty-source modulus witness:*

$$\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, target, modulus, cert, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(target, \mathsf{e}) \wedge \mathsf{ContinuousModu}$$

Proof. In the forward direction, empty-map exactness for the continuous-function component gives a modulus witness over the target, while empty-distance exactness for the metric witness collapses both endpoint histories to e. Transporting across the target collapse presents the modulus witness over e.

Conversely, the empty endpoint data presents the graph carrier as the empty-map exactness case, and metric empty-distance exactness builds the distance witness over the two empty endpoints. Their conjunction is exactly the concrete continuous-map carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty map empty distance boundaries iff`

Theorem 130.20 (ContinuousMap prefix canonical distance closure)**.** *For histories* $p, source, map, target, modulu$ BHist*, if* p *is unary and* $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$ *holds, then the prefixed endpoints carry the same map and modulus with the prefixed certificate and canonical prefixed distance:*

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier prefix canonical distance closed`

Proof. Prefix closure for the continuous-function carrier gives the prefixed function half. The metric half is built canonically: the prefixed source and target are unary by unary append closure, their appended distance is unary by another append-closure step, and the distance continuation is the defining append equation. □

Theorem 130.21 (ContinuousMap prefixed carrier canonical distance determinacy)**.** *For histories* $p, source, map, target, modulus, cert, distance, distance' : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$*,* $\mathsf{ContinuousMapCarrier}(source,$ *and*

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), distance')$$

hold, then the displayed prefixed carrier has the canonical prefixed distance ledger:

$$\mathsf{hsame}(distance', \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

Proof. First build the canonical prefixed carrier from the original one. Apply Theorem 130.20 to the unary prefix and the original carrier.

Now compare the displayed prefixed carrier with that canonical prefixed carrier. They have the same prefixed source, map, prefixed target, modulus, and prefixed certificate, so Theorem 130.13 determines the distance component.

Projecting that distance component gives exactly the displayed orientation: $\mathsf{hsame}(distance', \mathsf{append}(\mathsf{append}(p,$ □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier prefix canonical distance deterministic`

Theorem 130.22 (ContinuousMap prefixed carrier canonical distance exactness)**.** *For histories* $p, source, map, target, modulus, cert, distance, distance' : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$ *and* $\mathsf{ContinuousMapCarrier}(sour$ *then the prefixed continuous-map carrier is equivalent to the displayed canonical prefixed distance classifier:*

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), distance') \iff \mathsf{hsame}(distance', \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

Proof. In the forward direction, apply Theorem 130.21 to the unary prefix, the original carrier, and the displayed prefixed carrier. This identifies $distance'$ with the canonical prefixed source-target ledger.

Conversely, first build the canonical prefixed carrier by applying Theorem 130.20 to $\mathsf{UnaryHistory}(p)$ and the original continuous-map carrier. Project its continuous-function half.

Now apply Theorem 130.14 in the reverse direction to that continuous-function half and the displayed classifier $\mathsf{hsame}(distance', \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target)))$. This repacks the prefixed continuous-function data with the arbitrary displayed distance ledger $distance'$.

The ten-field fit is: traditional target prefixed continuous-map distance normalization; BEDC source ContinuousMapCarrier, ContinuousFunctionCarrier, MetricDistanceWitness, and Cont; carrier ContinuousMapCarrier; classifier hsame; stability from prefix closure, distance determinacy, and canonical distance exactness; ledger the hidden prefixed distance history; core theorem this iff; exactness the displayed iff; downstream use compact-continuous map endpoint and composition normalization; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier prefix canonical distance exactness`

Theorem 130.23 (ContinuousMap prefixed canonical distance package)**.** *The ten-field fit is: traditional target continuous-map prefix-normalization package; BEDC source* ContinuousMapCarrier *plus* MetricDistanceWitness*; carrier* ContinuousMapCarrier*; classifier* hsame*; stability prefix closure, determinacy, and exactness; ledger prefix and displayed distance; core theorem this package; exactness the iff row; standard bridge deferred; status* ProofTarget*.*

For histories $p, source, map, target, modulus, cert, distance, displayed : \mathsf{BHist}$*, if* $\mathsf{UnaryHistory}(p)$ *and* $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$*, then the following three facts hold:*

1.

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

2.

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), displayed) \to \mathsf{hsame}(displayed, \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

3.

$$\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), displayed) \iff \mathsf{hsame}(displayed, \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).$$

Proof. For the canonical carrier clause, apply Theorem 130.20 to the unary prefix and the original continuous-map carrier.

For the displayed determinacy clause, fix a displayed prefixed carrier with the same source, map, target, modulus, and prefixed certificate fields. Apply Theorem 130.21 to the unary prefix, the original carrier, and this displayed prefixed carrier.

For the exact classifier clause, apply Theorem 130.22 to the same unary prefix and original carrier. Its conclusion is precisely the iff between the displayed prefixed carrier and the hsame classifier for the canonical prefixed distance ledger. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier prefixed canonical distance package`

Theorem 130.24 (ContinuousMap empty map and empty distance certificate readback)**.** *For histories source, target, modulus, and cert, if* $\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, target, modulus, cert, \mathsf{e})$*, then the endpoint collapses and the modulus certificate reads back as a unary history equivalent to the modulus:*

$$\mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(target, \mathsf{e}) \wedge \mathsf{UnaryHistory}(cert) \wedge \mathsf{hsame}(cert, modulus).$$

Proof. Apply Theorem 130.19 to the carrier. The forward implication yields $\mathsf{hsame}(source, \mathsf{e})$, $\mathsf{hsame}(target, \mathsf{e})$, and $\mathsf{ContinuousModulusWitness}(\mathsf{e}, modulus, cert)$.

Apply the empty-source modulus-witness readback theorem from the continuous chapter to the final witness. It gives $\mathsf{UnaryHistory}(cert)$ and $\mathsf{hsame}(modulus, cert)$; symmetry of hsame presents the last conjunct in the displayed certificate-to-modulus orientation. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty map empty distance certificate readback`

Theorem 130.25 (ContinuousMap empty map and empty distance certificate exactness)**.** *For histories source, target, modulus, and cert, a concrete continuous-map carrier with empty graph map and empty distance ledger is equivalent to empty endpoint boundaries together with a readable modulus certificate:*

$$\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, target, modulus, cert, \mathsf{e}) \iff \mathsf{hsame}(source, \mathsf{e}) \wedge \mathsf{hsame}(target, \mathsf{e}) \wedge \mathsf{UnaryHistory}(cer$$

Proof. In the forward direction, apply Theorem 130.24. This is exactly the displayed endpoint collapse and certificate readback.

For the reverse direction, start from the four displayed conjuncts. Apply Theorem 59.4 with target $cert$: the unary certificate field and symmetry of $\mathsf{hsame}(cert, modulus)$ give $\mathsf{ContinuousModulusWitness}(\mathsf{e}, modul$

Together with $\mathsf{hsame}(source, \mathsf{e})$ and $\mathsf{hsame}(target, \mathsf{e})$, this is the right-hand side of Theorem 130.19. Applying that theorem in the reverse direction rebuilds $\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, target, modulu$ □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty map empty distance certificate exactness`
Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty map empty distance certificate iff`

The ten-field fit is: traditional target empty continuous-map readback, BEDC source ContinuousMapUp with ContinuousModulusWitness, carrier ContinuousMapCarrier, classifier hsame, graph ledger e, distance ledger e, certificate ledger $cert$, core theorem Theorem 130.24, exactness the displayed conjunction, and downstream consumer compact-continuous map endpoint readback.

Theorem 130.26 (ContinuousMap visible-context canonical distance exactness)**.** *For histories* $p, q, source, map, target, modulus, cert, distance : \mathsf{BHist}$*, the visible continuous-map carrier is exactly the central continuous-function carrier together with the canonical source-target distance classifier:*

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(p \cdot source, map, p \cdot target, modulus \cdot q, (p \cdot cert) \cdot q, p \cdot distance)\\ &\quad\iff \quad \mathsf{UnaryHistory}(p) \wedge \mathsf{UnaryHistory}(q) \wedge \mathsf{ContinuousFunctionCarrier}(source, map, target, modulus, cert) \wedge \mathsf{hsa}\end{aligned}$$

Proof. For the forward direction, apply Theorem 130.3 to the displayed carrier. This produces unary evidence for p and q, the central category hom row, the central modulus row, and $\mathsf{MetricDistanceWitness}(sour$

The category hom row and the modulus row repack as $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulus,$ Compare the resulting central continuous-map carrier with its canonical-distance form by Theorem 130.14. This replaces the central metric witness by the classifier $\mathsf{hsame}(distance, \mathsf{append}(source, target))$.

For the reverse direction, start from the two unary context histories, the central continuous-function carrier, and the displayed canonical classifier. Theorem 130.14 builds the central continuous-map carrier, equivalently transporting the canonical metric witness for $\mathsf{append}(source, target)$ along $\mathsf{hsame}(distance, \mathsf{append}(source, target))$.

Project the central carrier into its category hom, modulus, and metric rows. Then apply Theorem 130.3 from right to left with the two unary context histories. The result is precisely the displayed visible-context ContinuousMapCarrier. □

Theorem 130.27 (ContinuousMapUp public export surface)**.** *The metric certificate bridge of Definition 130.39, together with the existing identity, composition, canonical-distance, field-transport, and associativity rows, exports the public* $\mathsf{ContinuousMap}^{\uparrow}$ *surface over* BHist*. The exported data are exactly the source row, pattern row, componentwise classifier, stability rows, canonical distance ledger, identity rows, and composition rows.*

This public surface does not claim host functions, function-space topology, standard topological-space bridges, or pointwise or uniform convergence.

Proof. The source row, pattern row, componentwise classifier, stability rows, and canonical distance ledger are the five certificate rows packaged by Definition 130.39. The componentwise classifier and its stability are transported by Definition 130.17, while the canonical distance ledger is the endpoint append ledger used by the same bridge.

The public identity rows are the empty source and target carriers of Definition 130.40. Their two-sided behavior inside composition is supplied by Definition 130.41.

The public composition rows are closed by Definition 130.47 and repackaged by Definition 130.52. The associative bracketing row is the canonical package of Definition 130.57.

These cited rows exhaust the displayed $\mathsf{ContinuousMap}^{\uparrow}$ public surface over BHist. Since none of them introduces host functions, a function-space topology, a standard topological-space bridge, or a pointwise/uniform convergence interface, the theorem exports only the stated BEDC surface. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapPublic export surface`

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$.

Constructive story: A continuous map is built as a six-history package: source, map, target, modulus, certificate, and distance, where the first five form a ContinuousFunctionCarrier and the final history is a MetricDistanceWitness between the source and target. The carrier accepts only packages that decompose into a categorical hom row, a target-side modulus witness, and a metric distance row; missing graph continuation, modulus continuation, or distance ledger data leaves the package outside the surface. Two packages are classified together by componentwise hsame on all six visible histories, with the hidden canonical-distance row identifying the displayed distance with the endpoint append that identity and composition will consume. The source is the category, continuity, and metric triple, the pattern is the six-field continuous-map carrier, and the ledger records the graph, modulus, certificate, distance, canonical distance exactness, identities, composition rows, and associative bracketing. The five name-certificate obligations follow from empty identity carriers for inhabitation, from history-equality for reflexive, symmetric, and transitive comparison, and from field transport plus composition closure for preservation of admitted carriers under the classifier. A package h lies in the signature gap of a $\mathsf{ContinuousMap}^{\uparrow}$ token p when the source-domain accepts its category-continuity-metric rows, the signature relation pairs the six-history carrier with an answer package and evidence ledger, and the token-introduction relation names that answer as p; concretely, p records the same map, modulus, certificate, target, and canonical distance ledgers. The category-metric terminal extension bridge reads the same carrier after extending the terminal modulus certificate, and the chapter registers this package under $\mathsf{ContinuousMap}^{\uparrow}$ without host functions, function-space topology, pointwise convergence, uniform convergence, or a standard topological bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper-side scope is exactly the public export surface of Definition 130.27, the metric certificate bridge of Definition 130.39, and the category-metric terminal extension bridge of Definition 130.55.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier\_category\_metric\_terminal\_extension\_bridge`

Bridge status: bridgeChecked

Not claimed: Host functions, function-space topology, sheafified continuity, exponentiability, pointwise or uniform convergence, and any standard topological-space bridge stronger than the checked BEDC schema bridge.

Upgrade path: no further upgrade – chapter at top of formal axis

Theorem 130.28 (ContinuousMap visible-context composition category-metric exactness)**.** *For histories* $p, q, source, middle, target, f, g, fg, modF, modG, modFG, certF, certG, cert, distF, distG, displayed$: BHist*, assume the two visible-context carriers*

$$\mathsf{ContinuousMapCarrier}(p \cdot source, f, p \cdot middle, modF \cdot q, (p \cdot certF) \cdot q, p \cdot distF),$$
$$\mathsf{ContinuousMapCarrier}(p \cdot middle, g, p \cdot target, modG \cdot q, (p \cdot certG) \cdot q, p \cdot distG),$$

and assume the graph, modulus, and target-certificate ledgers

$$\mathsf{Cont}(f, g, fg), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, cert).$$

Then visible-context composition exposes the full category-metric package:

1. the composite categorical hom row,

$$\mathsf{CategoryHomCarrier}(source, target, fg);$$

2. the composite target-side modulus row,

$$\mathsf{ContinuousModulusWitness}(target, modFG, cert);$$

3. the canonical source-target metric distance row,

$$\mathsf{MetricDistanceWitness}(source, target, \mathsf{append}(source, target));$$

4. the visible composite carrier has exactly the canonical distance classifier,

$$\mathsf{ContinuousMapCarrier}(p \cdot source, fg, p \cdot target, modFG \cdot q, (p \cdot cert) \cdot q, p \cdot displayed)$$
$$\iff \quad \mathsf{hsame}(displayed, \mathsf{append}(source, target)).$$

Proof. Apply Theorem 130.3 to the first visible carrier. This recovers unary evidence for p and q, the central categorical row $\mathsf{CategoryHomCarrier}(source, middle, f)$, the central modulus row $\mathsf{ContinuousModulusWitness}(middle, modF, certF)$, and the metric row $\mathsf{MetricDistanceWitness}(source, middle, distF$

Apply the same decomposition theorem to the second visible carrier. The result has the same visible contexts and exposes $\mathsf{CategoryHomCarrier}(middle, target, g)$, $\mathsf{ContinuousModulusWitness}(target, modG, certG$ and $\mathsf{MetricDistanceWitness}(middle, target, distG)$.

The two category and modulus rows repack as the two central ContinuousFunctionCarrier hypotheses. Apply Theorem 59.7 with the three displayed continuation ledgers. The composite central carrier $\mathsf{ContinuousFunctionCarrier}(source, fg, target, modFG, cert)$ follows.

Apply Theorem 59.139 to this composite central carrier. Its graph projection is exactly $\mathsf{CategoryHomCarrier}(source, target, fg)$, and its target-modulus projection is exactly $\mathsf{ContinuousModulusWitness}(tar$

Use Theorem 130.4 from right to left on the composite central carrier. Decomposing the resulting canonical ContinuousMapCarrier by Theorem 130.2 gives the canonical row $\mathsf{MetricDistanceWitness}(source, target$

It remains to classify visible displayed distances. The unary witnesses for p and q are already available, and the composite central ContinuousFunctionCarrier has been constructed. Therefore Theorem 130.26 specializes the displayed visible carrier exactly to $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$. □

Lean4 ✓ BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier visible context composition category metric exactness

The categorical hom row in Theorem 130.28 is the concrete sibling surface of chapter 61. The canonical source-target distance row is the metric sibling surface of chapter 57. The theorem consumes both through the visible-context carrier and does not create either structure.

The ten-field fit is: traditional uniformly continuous maps with visible source and target context under composition; BEDC source CategoryUp/ContinuousUp/MetricUp/ContinuousMapUp; carrier ContinuousMapCarrier; classifier hsame; stability from visible carrier decomposition, central continuous-function composition, graph and modulus readback, canonical metric packaging, and visible canonical-distance exactness; ledger visible contexts, source, middle, target, graph maps, moduli, certificates, and metric-distance histories; core theorem Theorem 130.28; exactness the displayed category hom row, target-side modulus row, canonical metric row, and visible distance classifier; standard bridge is deferred; status ProofTarget. The top-level $\mathsf{ContinuousMap}^{\uparrow}$ node reads the metric carrier through chapter 57, the pointwise-modulus surface through chapter 59, and the graph and composition surface through chapter 61. Compact-uniform consumers read this combined horizontal lattice through the existing child rows, rather than through an isolated kernel-only entry.

130.2 The certificate

Definition 130.29 (Metric ContinuousMapUp source specification)**.** *For histories source, map, target, modulus, ce* BHist*, the metric* $\mathsf{ContinuousMap}^{\uparrow}$ *source row is the conjunction*

$$\mathsf{CategoryHomCarrier}(source, target, map) \wedge \mathsf{ContinuousModulusWitness}(target, modulus, cert) \wedge \mathsf{MetricDistanceWi}$$

It names the categorical graph carrier, the target-side modulus witness, and the endpoint distance witness carried by the six-history certificate object.

Lean4 (definition) BEDC.Derived.ContinuousMapUp.ContinuousMapMetricSourceSpec
Lean4 ✓ BEDC.Derived.ContinuousMapUp.ContinuousMapMetricSourceSpec canonical carrier

Definition 130.30 (Metric ContinuousMapUp pattern specification)**.** *The metric* $\mathsf{ContinuousMap}^{\uparrow}$ *pattern is the six-field carrier formation*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance).$$

The displayed carrier is the public pattern consumed by compact-continuous and Heine–Cantor arguments: it packages the source, map, target, modulus, certificate, and distance histories in one BEDC object.

Lean4 (definition) BEDC.Derived.ContinuousMapUp.ContinuousMapMetricPatternSpec
Lean4 ✓ BEDC.Derived.ContinuousMapUp.ContinuousMapMetricPatternSpec carrier readback

Definition 130.31 (Metric ContinuousMapUp classifier specification)**.** *For two metric* $\mathsf{ContinuousMap}^{\uparrow}$ *carriers with histories* $source, map, target, modulus, cert, distance$ *and* $source', map', target', modulus', cert', dista$ *the classifier is componentwise* hsame *on the six displayed histories:*

$$\mathsf{hsame}(source, source') \wedge \mathsf{hsame}(map, map') \wedge \mathsf{hsame}(target, target') \wedge \mathsf{hsame}(modulus, modulus') \wedge \mathsf{hsame}(cert, c$$

The same classifier row records the hidden canonical-distance exactness

$$\mathsf{hsame}(distance, \mathsf{append}(source, target))$$

for each admitted carrier, and similarly for the primed carrier.

Lean4 (definition) `BEDC.Derived.ContinuousMapUp.ContinuousMapMetricClassifierSpec`

Definition 130.32 (Metric ContinuousMapUp stability certificate)**.** *The metric* ContinuousMap$^{\uparrow}$ *stability certificate consists of:*

1. *field transport of the six histories under the componentwise* hsame *classifier;*
2. *canonical-distance exactness for each admitted carrier;*
3. *two-sided identity closure by empty maps and empty distance ledgers;*
4. *composition closure by categorical graph composition, modulus composition, and canonical endpoint-distance normalization;*
5. *associativity of the composed canonical package.*

Lean4 (definition) `BEDC.Derived.ContinuousMapUp.ContinuousMapMetricStabilityCertificate`
Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapMetricStabilityCertificate field transport`

Definition 130.33 (Metric ContinuousMapUp ledger policy)**.** *The metric* ContinuousMap$^{\uparrow}$ *ledger policy records the six histories* $source, map, target, modulus, cert, distance$*, the visible source row of Definition 130.29, and the hidden canonical-distance ledger* hsame$(distance,$ append$(source, target))$*. Under composition it keeps the graph and modulus continuation ledgers visible and rewrites the displayed distance to the canonical endpoint append.*

Lean4 (definition) `BEDC.Derived.ContinuousMapUp.ContinuousMapMetricLedgerPolicy`

Theorem 130.34 (ContinuousMap CompleteMetric CompactMetric uniform route)**.** *Let a metric* ContinuousMap$^{\uparrow}$ *carrier have source row supplied by a* CompactMetric$^{\uparrow}$ *certificate and target row supplied by a* CompleteMetric$^{\uparrow}$ *certificate. Every uniform-modulus read factors through the displayed six-history ContinuousMap carrier, the* CompactMetric$^{\uparrow}$ *total-bounded finite-net and complete-limit rows, and the* CompleteMetric$^{\uparrow}$ *limit-witness rows.*

The route exports no open-cover compactness, no choice-selected subnet, and no host function equality. It is a finite certificate route over the ContinuousMap source, map, target, modulus, certificate, distance, hsame*,* Cont*,* Pkg*, and* NameCert *rows together with the compact and complete metric ledgers.*

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier compact complete uniform route`

Proof. Project the ContinuousMap side through Definition 130.29, Definition 130.30, and Definition 130.33. These rows expose exactly the six histories, the categorical graph row, the modulus row, the metric source and target rows, and the canonical endpoint-distance ledger.

Read the compact source through Definition 129.5 and Definition 129.20. The source contributes total-bounded finite-net rows and complete-limit witness rows over the same metric carrier, not open-cover compactness or a selected subnet.

Read the complete target through the complete-metric limit-witness surface, in particular Definition 128.2 and Definition 128.4. These rows provide the target-side limit witness and its transport without adding a host function equality test.

Repack the ContinuousMap, CompactMetric, and CompleteMetric rows as one uniform-modulus consumer route. The visible hsame, Cont, Pkg, and NameCert ledgers carry the route, so every uniform-modulus read remains finite and excludes open-cover compactness, choice-selected subnets, and host function equality. □

Theorem 130.35 (ContinuousMap compact-complete uniform finite route sharpening)**.** *Let a metric* $\mathsf{ContinuousMap}^{\uparrow}$ *carrier be displayed by the six histories* $source, map, target, modulus, cert, distance$: BHist. *Suppose the source metric row is read through a* $\mathsf{CompactMetric}^{\uparrow}$ *certificate and the target metric row is read through a* $\mathsf{CompleteMetric}^{\uparrow}$ *certificate. Then every uniform-modulus consumer route factors through exactly the following finite surface:*

1. *the displayed ContinuousMap histories* $source, map, target, modulus, cert, distance$*;*
2. *the compact finite-net rows and complete-limit rows exported by the* $\mathsf{CompactMetric}^{\uparrow}$ *source;*
3. *the target complete-limit witness rows exported by* $\mathsf{CompleteMetric}^{\uparrow}$*;*
4. *the* $\mathsf{MetricDistanceWitness}$ *row for the source-target metric read;*
5. *componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

The route exports no open-cover compactness, no selected subnet, no host function equality, no quotient stream equality, and no ambient completeness principle. In particular, any later compact-uniform-continuity or Real-completion consumer reading the route receives only this finite $\mathsf{NameCert}$ *consumer surface.*

Proof. First project the ContinuousMap carrier by Definition 130.29, Definition 130.30, and Definition 130.33. These definitions expose the six histories, the categorical graph row, the target modulus row, the metric distance witness, the componentwise hsame classifier, displayed Cont rows, Pkg provenance, and local $\mathsf{NameCert}$ bookkeeping.

Next apply Definition 130.34. This already forces every uniform-modulus read through the ContinuousMap carrier, the compact finite-net and complete-limit rows, and the complete-metric limit-witness rows, while refusing open-cover compactness, selected subnets, and host function equality.

Read the compact coordinate through Definition 129.5 and Definition 129.20. The compact source therefore contributes only finite-net coverage, total-bounded rows, complete-limit rows, and their displayed metric ledgers.

Read the complete coordinate through Definition 128.2 and Definition 128.4. The target contributes only named limit witnesses, threshold rows, late-distance witnesses, and their classifier-stable transport rows.

Finally repack the three readings as the finite consumer surface listed in the theorem. Since each coordinate is displayed as a BHist row with its own hsame, Cont, Pkg, and $\mathsf{NameCert}$ evidence, no quotient stream equality, ambient completeness principle, or host-level equality test can be read from the route. □

Theorem 130.36 (ContinuousMap compact-complete uniform route refinement)**.** *Let a metric* $\mathsf{ContinuousMap}^{\uparrow}$ *carrier be displayed by the histories* $source, map, target, modulus, cert, distance$: BHist*, with the source row read through a* $\mathsf{CompactMetric}^{\uparrow}$ *certificate and the target row read through a* $\mathsf{CompleteMetric}^{\uparrow}$ *certificate. Then every compact-complete uniform-modulus consumer of this carrier has the exact finite surface consisting only of:*

1. *the six* $\mathsf{ContinuousMap}^{\uparrow}$ *histories* $source, map, target, modulus, cert, distance$*;*
2. *the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows for the source metric;*
3. *the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness rows for the target metric;*
4. *the* $\mathsf{MetricDistanceWitness}(source, target, distance)$ *row;*
5. *componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

No open-cover compactness, selected subnet, quotient stream equality, or host function equality is part of the consumer surface.

Proof. Begin with Definition 130.34. It places every uniform-modulus read behind the six-history $\mathsf{ContinuousMap}^{\uparrow}$ carrier, the compact metric rows, and the complete-metric limit-witness rows.

Sharpen the carrier boundary with Definition 130.35. This identifies the visible consumer coordinates as the displayed histories, finite-net rows, target limit-witness rows, the source-target MetricDistanceWitness row, and the componentwise hsame, Cont, Pkg, and NameCert ledgers.

The compact coordinate is then read through Definition 129.5 and Definition 129.20. These theorems keep the source contribution on finite-net and complete-limit rows, so an open-cover compactness principle or selected subnet has no exported row to consume.

The target coordinate is read through Definition 128.2 and Definition 128.4. The target contribution is therefore a named limit-witness surface with stable transported rows, not quotient stream equality or a host function equality test.

Combining these projections leaves exactly the five listed families of rows. Every later compact-uniform or Real-completion consumer must enter through that finite NameCert surface, and the excluded objects remain outside the route because none of the cited certificate surfaces exports them. □

Theorem 130.37 (ContinuousMap compact-complete uniform exact consumer surface)**.** *Let a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *carrier have histories* $source, map, target, modulus, cert, distance$: BHist*. Suppose the source metric row is supplied by a* $\mathsf{CompactMetric}^{\uparrow}$ *certificate, the target metric row is supplied by a* $\mathsf{CompleteMetric}^{\uparrow}$ *certificate, and the endpoint distance row is displayed as* MetricDistanceWitness($sourc$ *A compact-complete uniform-modulus consumer of* ContinuousMapCarrier($source, map, target, modulus, cert, distan$ *is accepted exactly when its finite surface consists of the following rows:*

1. *the six displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *histories* $source, map, target, modulus, cert, distance$*;*
2. *the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net and complete-limit rows supplied by Definition 129.5 and Definition 129.20;*
3. *the target* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness rows supplied by Definition 128.2 and Definition 128.4;*
4. *the row* MetricDistanceWitness($source, target, distance$)*;*
5. *componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *provenance for the displayed rows.*

Equivalently, every accepted consumer projects to precisely these rows, and any display of precisely these rows repacks as the same finite compact-complete uniform consumer surface. No open-cover compactness, selected subnet, quotient stream equality, host function equality, or ambient completeness principle is exported.

Proof. For the forward direction, first apply Definition 130.34. It places the consumer behind the six-history $\mathsf{ContinuousMap}^{\uparrow}$ carrier, the compact source rows, and the complete target rows.

Apply Definition 130.35 and Definition 130.36 to identify the visible rows. These route theorems leave only the displayed histories, the compact finite-net contribution, the target limit-witness contribution, the metric-distance witness, and the componentwise hsame, Cont, Pkg, and NameCert provenance.

The compact rows are then read through Definition 129.5 and Definition 129.20. Thus the compact source coordinate contributes finite-net and complete-limit rows, not an open-cover compactness principle or a selected subnet.

The target rows are read through Definition 128.2 and Definition 128.4. Thus the complete target coordinate contributes named limit-witness and stable transport rows, not quotient stream equality, host function equality, or an ambient completeness principle.

For the reverse direction, start from a displayed finite surface containing exactly the five listed families of rows. The six ContinuousMap histories, the metric-distance witness, and the componentwise hsame, Cont, Pkg, and NameCert provenance rebuild the displayed carrier boundary.

The compact finite-net and complete-limit rows supply the source side, while the complete limit-witness rows supply the target side. Repacking these rows through the same three route theorems gives the compact-complete uniform-modulus consumer and no additional exported object. Hence acceptance is exactly the displayed finite surface. □

Theorem 130.38 (ContinuousMap CompactMetric MetricTriangle handoff)**.** *Let a metric* $\mathsf{ContinuousMap}^{\uparrow}$ *carrier be displayed by the six histories* $source, map, target, modulus, cert, distance : \mathsf{BHist}$*. Suppose its source row is read through a compact* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net certificate, its target row is read through a* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness certificate, and an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier uses the same compact-net center, the two* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise target-estimate rows, the target* $\mathsf{Met}^{\uparrow}$ *triangle row, and the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. Then the combined handoff factors through exactly the finite surface of the six* $\mathsf{ContinuousMap}^{\uparrow}$ *histories, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows, the* $\mathsf{CompleteMetric}^{\uparrow}$ *target limit-witness rows, the source and target metric-triangle rows, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

The handoff exports no open-cover compactness, selected subnet, choice principle, host function equality, quotient stream equality, semantic continuity theorem, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with Definition 130.36. It places the ContinuousMap consumer behind the six displayed histories, the compact finite-net source rows, the complete target limit-witness rows, the metric distance witness, and the visible hsame, Cont, Pkg, and NameCert ledgers.

Next read the accepted $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ packet through Definition 1159.31. This binds the same compact-net center to the two pointwise target estimates and to the target metric triangle row before the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer can read the estimate.

Use Definition 1159.30 to identify the consumer boundary. The boundary contains the selected finite-net center, source metric rows, pointwise target estimates, target triangle row, and support rows, so it agrees with the ContinuousMap route on the finite carrier data rather than on host-level functions or quotient streams.

Finally apply Definition 1159.34. It packages the public $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ export as the same finite BHist, hsame, Cont, Pkg, and NameCert surface required by the ContinuousMap refinement. Since none of the cited surfaces exports open-cover compactness, selected subnets, choice, semantic continuity, $\mathsf{Real}^{\uparrow}$ completeness, host function equality, or quotient stream equality, the composed handoff cannot supply those objects to a downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer. □

Theorem 130.39 (Metric ContinuousMapUp certificate bridge)**.** *The five rows of Definition 130.29–Definition 130.33 form the paper-side* NameCert *obligations for* $\mathsf{ContinuousMap}^{\uparrow}$ *with carrier*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance).$$

The bridge identifies the source row with the CategoryUp*,* ContinuousUp*, and* MetricUp *components; exposes componentwise* hsame *on the six displayed histories; and uses the hidden canonical-distance exactness* $\mathsf{hsame}(distance, \mathsf{append}(source, target))$ *as the distance certificate consumed by identity and composition.*

Proof. First unpack the carrier through Definition 130.15. This gives the categorical hom row, the target-side modulus row, and the canonical endpoint-distance classifier. Together with Definition 130.29, these are exactly the visible source obligations.

Next use Definition 130.17 to transport the six displayed histories along the componentwise classifier. The transport step supplies the classifier and stability fields without changing the public carrier shape.

Identity closure is supplied by Definition 130.41. It shows that empty identity maps and empty distance ledgers serve as the two-sided units required by the certificate policy.

Finally apply Definition 130.57. It composes the graph, modulus, certificate, and distance rows while normalizing the distance back to the canonical endpoint append. This is the ledger behavior stated in Definition 130.33, so the five displayed rows give the advertised NameCert obligations for $\mathsf{ContinuousMap}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.continuousmap semantic name certificate`

Theorem 130.40 (ContinuousMap empty identity carriers)**.** *For histories* $source, target, map, modulus, cert, dist$ BHist*, assume*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance).$$

Then the empty-map identities at the source and target are continuous-map carriers:

1. *the source identity carrier is*

$$\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, source, \mathsf{e}, source, \mathsf{append}(source, source));$$

2. *the target identity carrier is*

$$\mathsf{ContinuousMapCarrier}(target, \mathsf{e}, target, \mathsf{e}, target, \mathsf{append}(target, target));$$

Proof. Project the continuous-function carrier from the given ContinuousMapCarrier. Its fields supply unary endpoint data and the certificate continuation $\mathsf{Cont}(target, modulus, cert)$.

For the source identity, use the unary source field and the left-unit continuation $\mathsf{Cont}(source, \mathsf{e}, source)$ to form $\mathsf{ContinuousModulusWitness}(source, \mathsf{e}, source)$. Apply Theorem 59.12, then package the canonical distance by Theorem 130.4.

For the target identity, the same construction uses the unary target field and the left-unit continuation $\mathsf{Cont}(target, \mathsf{e}, target)$. The empty-map identity theorem builds the continuous-function carrier at $target$, and canonical distance packaging gives the displayed target identity carrier. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty identities closed`

Theorem 130.41 (ContinuousMap empty identities are two-sided composition units)**.** *For histories* $source, target, map, modulus, cert, distance, displayed_L, displayed_R$: BHist*, assume*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance).$$

Then the empty-map identities are two-sided units for composition:

1. *composing the source identity on the left with the displayed carrier, using*

$$\mathsf{Cont}(\mathsf{e}, map, map), \qquad \mathsf{Cont}(\mathsf{e}, modulus, modulus), \qquad \mathsf{Cont}(target, modulus, cert),$$

closes as the canonical carrier

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, \mathsf{append}(source, target));$$

2. *composing the displayed carrier with the target identity on the right, using*

$$\mathsf{Cont}(map, \mathsf{e}, map), \qquad \mathsf{Cont}(modulus, \mathsf{e}, modulus), \qquad \mathsf{Cont}(target, modulus, cert),$$

closes as the same canonical carrier

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, \mathsf{append}(source, target));$$

3. *if the left and right unit composites are displayed with distances* $displayed_L$ *and* $displayed_R$*, respectively, then*

$$\mathsf{hsame}(displayed_L, \mathsf{append}(source, target)) \wedge \mathsf{hsame}(displayed_R, \mathsf{append}(source, target)).$$

Proof. Apply Theorem 130.40 to the displayed carrier. This supplies the source and target empty identity carriers used as the unit factors.

Now apply Theorem 130.47 to the source identity carrier and the displayed carrier, with the BEDC left-unit laws $\mathsf{Cont}(\mathsf{e}, map, map)$ and $\mathsf{Cont}(\mathsf{e}, modulus, modulus)$ and the projected certificate ledger $\mathsf{Cont}(target, modulus, cert)$. This gives the left unit composite over $\mathsf{append}(source, target)$.

Apply the same composition-closure theorem to the displayed carrier and the target identity carrier, using the right-unit laws $\mathsf{Cont}(map, \mathsf{e}, map)$ and $\mathsf{Cont}(modulus, \mathsf{e}, modulus)$ together with the same certificate ledger. This gives the right unit composite over the same canonical distance.

Finally, apply Theorem 130.50 to each unit composite presentation. Comparing any displayed left or right unit-composite distance with the corresponding canonical composite identifies it by hsame with $\mathsf{append}(source, target)$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty identities two sided composition units`

Theorem 130.42 (ContinuousMap identity carrier distance determinacy)**.** *For histories* $source, map, target, modu$ BHist*, assume*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance).$$

If the displayed source and target identity carriers are

$$\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, source, \mathsf{e}, source, displayedSource)$$

and

$$\mathsf{ContinuousMapCarrier}(target, \mathsf{e}, target, \mathsf{e}, target, displayedTarget),$$

then their hidden distance ledgers are the canonical self-distance ledgers:

$$\mathsf{hsame}(displayedSource, \mathsf{append}(source, source)) \wedge \mathsf{hsame}(displayedTarget, \mathsf{append}(target, target)).$$

Proof. Apply Theorem 130.40 to the given continuous-map carrier. Project its source-identity row to obtain the canonical carrier $\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, source, \mathsf{e}, source, \mathsf{append}(source, source))$. Project its target-identity row to obtain $\mathsf{ContinuousMapCarrier}(target, \mathsf{e}, target, \mathsf{e}, target, \mathsf{append}(target, target))$.

Compare the displayed source identity carrier with the canonical source identity carrier by Theorem 130.13. The two carriers have the same source, graph map, and modulus ledgers, so the distance component of the comparison gives $\mathsf{hsame}(displayedSource, \mathsf{append}(source, source))$.

Compare the displayed target identity carrier with the canonical target identity carrier in the same way. Projecting the distance component gives $\mathsf{hsame}(displayedTarget, \mathsf{append}(target, target))$, and pairing the two classifiers gives the displayed conjunction.

The ten-field fit is: traditional target identity maps with stable self-distance ledgers; BEDC source BHist with Cont and hsame; carrier ContinuousMapCarrier; classifier hsame; stability obligation identity-carrier distance determinacy; ledger the self-distance ledger $\mathsf{append}(source, source)$ or $\mathsf{append}(target, target)$; core theorem this identity-carrier comparison; exactness the displayed conjunction; standard bridge deferred; downstream consumer $\mathsf{Cat}^{\uparrow}$ identity-morphism normalization. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty identity distance deterministic`

Theorem 130.43 (ContinuousMap composite graph depth addition)**.** *For histories source, mid, target, mapF, ma*
assume the two continuous-function carriers

$$\mathsf{ContinuousFunctionCarrier}(source, mapF, mid, modF, certF),$$
$$\mathsf{ContinuousFunctionCarrier}(mid, mapG, target, modG, certG),$$

and the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG),$$

then the composite graph has metric depth

$$\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapFG).$$

Proof. Composition closure for continuous-function carriers builds the composite carrier from the two displayed carriers and the continuation ledgers. Its graph fields are exactly a metric distance witness with endpoints *source*, *mapFG*, and *target*.

Metric-distance depth addition for that witness gives the displayed equality. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMap comp graph depth add`

Theorem 130.44 (ContinuousMap carrier graph depth addition)**.** *For histories source, map, target, modulus, cert*
BHist*, every continuous-map carrier*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$$

exposes its graph continuation and the corresponding metric depth identity:

$$\mathsf{Cont}(source, map, target) \quad \wedge \quad \mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDeptl}$$

Proof. Project the continuous-function carrier from the displayed ContinuousMapCarrier. Its graph field is exactly the continuation $\mathsf{Cont}(source, map, target)$. Repackage the source, map, and target unary rows with this graph continuation as a metric-distance witness from *source* and *map* to *target*, then apply metric-distance depth addition. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier graph cont depth add`

Theorem 130.45 (ContinuousMap carrier distance empty iff)**.** *For histories source, map, target, modulus, cert, di*
BHist*, every continuous-map carrier satisfies*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance) \Longrightarrow (\mathsf{hsame}(distance, \mathsf{e}) \Longleftrightarrow \mathsf{hsame}(source, \mathsf{e})$$

Proof. Project the metric-distance witness from the displayed ContinuousMapCarrier. If the displayed distance is hsame to e, transport it to the empty distance ledger and apply metric empty-distance readback to identify both endpoints with e. Conversely, empty endpoint classifiers build the metric-distance witness with empty distance ledger; metric result determinacy compares this witness with the carrier's displayed distance. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier distance empty iff`

Theorem 130.46 (ContinuousMap carrier zero distance iff empty endpoints)**.** *For histories source, map, target, m* BHist*, every continuous-map carrier satisfies*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance) \Longrightarrow (\mathsf{MetricDistanceDepth}(distance) = 0 \Longleftrightarrow$$

Proof. Metric depth zero is equivalent to the distance history being hsame to e. Apply the preceding continuous-map carrier distance boundary theorem to translate that empty distance classifier exactly into empty source and target endpoints. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier distance zero iff empty endpoints`

Theorem 130.47 (ContinuousMap carrier composition closure)**.** *For histories source, mid, target, mapF, mapG, n* BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, distG),$$

and the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

Then the composite is again a continuous-map carrier with the canonical source-target distance ledger:

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{append}(source, target)).$$

Proof. Project the continuous-function halves from the two ContinuousMapCarrier hypotheses. Apply Theorem 59.7 to those halves and the three continuation ledgers. This produces the composite $\mathsf{ContinuousFunctionCarrier}(source, mapFG, target, modFG, certFG)$.

Now package the canonical distance ledger. Apply Theorem 130.4 in the reverse direction to the composite continuous-function carrier, yielding the displayed ContinuousMapCarrier with distance $\mathsf{append}(source, target)$.

The ten-field fit is: traditional target composition of uniformly continuous maps, BEDC source ContinuousMapCarrier plus Cont, carrier ContinuousMapCarrier, classifier hsame, stability from continuous-function composition and canonical metric ledger packaging, ledger the hidden canonical distance, core theorem this closure, exactness canonical-distance carrier iff, standard bridge deferred, and status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier comp closed`

Theorem 130.48 (ContinuousMap empty distance composition closure)**.** *For histories source, mid, target, mapF,* BHist*, assume the two continuous-map carriers have empty distance ledgers:*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, \mathsf{e}),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, \mathsf{e}),$$

and assume the continuation ledgers for graph composition, modulus composition, and final certificate formation. Then source, mid, and target are all hsame *to* e*, and the composite carrier also has empty distance ledger:*

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{e}).$$

Proof. Apply the empty-distance metric readback to the two input carriers. The first carrier identifies *source* and *mid* with e, while the second identifies *mid* and *target* with e. Composition closure supplies the composite continuous-function carrier. The empty endpoint comparisons repackage the metric-distance witness for the composite as the empty distance ledger. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty distance comp closed`

Theorem 130.49 (ContinuousMap empty-distance empty-graph composition certificate readback)**.** *For histories* $source, mid, target, mapF, mapG, modF, modG, modFG, certF, certG, certFG$: BHist*, assume the two continuous-map carriers have empty distance ledgers:*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, \mathsf{e}),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, \mathsf{e}),$$

and assume

$$\mathsf{Cont}(mapF, mapG, \mathsf{e}), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

Then source, mid, and target are all hsame *to* e*, and the final composition certificate is a unary history satisfying*

$$\mathsf{UnaryHistory}(certFG) \qquad \textit{and} \qquad certFG\mathsf{hsame}modFG.$$

Proof. Apply Definition 130.48 with the empty composite graph. This gives the three empty endpoint readbacks and a composite continuous-map carrier $\mathsf{ContinuousMapCarrier}(source, \mathsf{e}, target, modFG, certFG, \mathsf{e}$ The empty-map, empty-distance certificate readback theorem then identifies $certFG$ as unary and hsame to $modFG$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier empty distance empty graph comp certificate readback`

Theorem 130.50 (ContinuousMap carrier composition distance determinacy)**.** *For histories* $source, mid, target, m$ BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, distG),$$

the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG),$$

and a displayed composite carrier

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, displayed).$$

Then the displayed composite distance is the canonical source-target ledger:

$$\mathsf{hsame}(displayed, \mathsf{append}(source, target)).$$

Proof. Theorem 130.47 applied to the two input carriers and the three continuation ledgers builds the canonical composite carrier

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{append}(source, target)).$$

Compare the displayed composite carrier with this canonical carrier by Theorem 130.13, with the same source, map, target, modulus, and certificate histories on both sides. The distance component of that determinacy conclusion gives $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier comp distance deterministic`

Theorem 130.51 (ContinuousMap composite canonical distance exactness)**.** *For histories* $source, mid, target, ma$ BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, distG),$$

and the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

Then the displayed composite carrier predicate is exactly the canonical source-target distance classifier:

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, displayed) \iff \mathsf{hsame}(displayed, \mathsf{append}(source, t$$

Proof. In the forward direction, suppose the displayed composite carrier is given. Apply Theorem 130.50 to the two input carriers, the three continuation ledgers, and the displayed composite carrier. This gives $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$.

Conversely, suppose the displayed distance satisfies $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$. Theorem 130.47 applied to the two input carriers and the three continuation ledgers builds the canonical composite carrier with distance $\mathsf{append}(source, target)$. Project its continuous-function half.

Now apply Theorem 130.14 in the reverse direction to that continuous-function half and the displayed hsame row. The result is the composite $\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, disp$

The ten-field fit is: traditional target composition of uniformly continuous maps with exact hidden distance ledger; BEDC source two ContinuousMapCarrier hypotheses plus three continuation ledgers; carrier ContinuousMapCarrier; classifier hsame on the displayed distance; stability from composition closure and canonical-distance exactness; ledger the hidden source-target distance; core theorem this iff; exactness the displayed equivalence; standard bridge ordinary composition distance bookkeeping; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier comp displayed distance canonical iff`

Theorem 130.52 (ContinuousMap composition canonical certificate package)**.** *For histories* $source, mid, target, n$ BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF),$$
$$\mathsf{ContinuousMapCarrier}(mid, mapG, target, modG, certG, distG),$$

and the continuation ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

Then the composition data determine the canonical certificate package:

1. *the composite carrier closes over the canonical source-target distance ledger,*

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, \mathsf{append}(source, target));$$

2. *every displayed composite carrier with the same visible fields has that canonical distance ledger, namely if*

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG, displayed),$$

then

$$\mathsf{hsame}(displayed, \mathsf{append}(source, target));$$

3. *the composite graph obeys metric-depth accounting,*

$$\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapFG).$$

Proof. For the carrier row, apply Theorem 130.47 to the two input carriers and the three continuation ledgers. This gives the displayed ContinuousMapCarrier with distance $\mathsf{append}(source, target)$.

For distance uniqueness, suppose a displayed composite carrier with the same source, map, target, modulus, and certificate histories is given. Compare the displayed carrier with the canonical carrier just obtained by Theorem 130.13. The distance component gives $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$.

For metric-depth accounting, project the continuous-function halves from the two input ContinuousMapCarrier hypotheses. Apply Theorem 130.43 to those halves and the same three continuation ledgers, producing the stated equality for MetricDistanceDepth. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier composition canonical certificate package`

Theorem 130.53 (ContinuousMap composition certificate determinacy)**.** *For histories* $source, mid, target, mapF,$ BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF), \qquad \mathsf{ContinuousMapCarrier}(mid, mapG, target, mod$$

and the composition ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

If another displayed composite carrier has the same source, map, target, modulus, and a certificate $certFG_2$*,*

$$\mathsf{ContinuousMapCarrier}(source, mapFG, target, modFG, certFG_2, displayed),$$

then the certificate histories agree:

$$\mathsf{hsame}(certFG, certFG_2).$$

Proof. First close the canonical composite carrier from the two input carriers and the three continuation ledgers by Theorem 130.47. Compare this canonical carrier with the displayed carrier by Theorem 130.13. The certificate component of that comparison is the required $\mathsf{hsame}(certFG, certFG_2)$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier comp certificate deterministic`

Theorem 130.54 (ContinuousMap composite target, certificate, and distance determinacy)**.** *For histories* $source, mid, target, target_2, mapF, mapG, mapFG, modF, modG, modFG, certF, certG, certFG, certFG_2$ BHist*, assume the two continuous-map carriers*

$$\mathsf{ContinuousMapCarrier}(source, mapF, mid, modF, certF, distF), \qquad \mathsf{ContinuousMapCarrier}(mid, mapG, target, mod$$

and the composition ledgers

$$\mathsf{Cont}(mapF, mapG, mapFG), \qquad \mathsf{Cont}(modF, modG, modFG), \qquad \mathsf{Cont}(target, modFG, certFG).$$

If a displayed composite carrier uses the same source, composite map, and modulus but displays target $target_2$*, certificate* $certFG_2$*, and distance displayed, then the displayed rows agree with the canonical composite:*

$$\mathsf{hsame}(target, target_2) \wedge \mathsf{hsame}(certFG, certFG_2) \wedge \mathsf{hsame}(displayed, \mathsf{append}(source, target)) \\ \wedge \mathsf{Cont}(source, target, \mathsf{append}(source, target)).$$

Proof. First close the canonical composite carrier from the two input carriers and the three continuation ledgers. Its target is *target*, its certificate is *certFG*, and its distance ledger is $\mathsf{append}(source, target)$. Compare this canonical carrier with the displayed composite carrier by Definition 130.13. The comparison gives target and certificate determinacy, identifies the canonical distance with *displayed*, and exposes the canonical endpoint continuation. Symmetry of hsame orients the displayed distance as $\mathsf{hsame}(displayed, \mathsf{append}(source, target))$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier comp target cert distance deterministic`

Theorem 130.55 (ContinuousMap category-metric terminal extension bridge)**.** *For histories* $source, map, target, modulus, cert, extra, modulus', cert', distance$: BHist*, assume the category, continuity, and metric rows*

$$\mathsf{CategoryHomCarrier}(source, target, map),$$
$$\mathsf{ContinuousModulusWitness}(target, modulus, cert),$$
$$\mathsf{MetricDistanceWitness}(source, target, distance),$$

and assume the terminal modulus-extension data

$$\mathsf{ContinuousModulusWitness}(cert, extra, cert') \qquad \textit{and} \qquad \mathsf{Cont}(modulus, extra, modulus').$$

Then these three sources assemble the extended continuous-map carrier

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus', cert', distance).$$

Proof. Use the reverse direction of Theorem 130.2 on the displayed category, continuity, and metric rows. This gives the base carrier $\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance)$.

Unpack this base carrier by Definition 130.1. The continuous-function half is $\mathsf{ContinuousFunctionCarrier}(source,$ while the metric half remains the supplied $\mathsf{MetricDistanceWitness}(source, target, distance)$.

Apply Theorem 59.162 to the continuous-function half, the witness $\mathsf{ContinuousModulusWitness}(cert, extra, cert'$ and the ledger $\mathsf{Cont}(modulus, extra, modulus')$. This yields $\mathsf{ContinuousFunctionCarrier}(source, map, target, modulu$

Pair that extended continuous-function carrier with the unchanged metric witness by Definition 130.1. The result is the displayed $\mathsf{ContinuousMapCarrier}(source, map, target, modulus', cert', distance)$.

The ten-field fit is: traditional target a continuous map whose modulus certificate is extended at the terminal end; BEDC source CategoryHomCarrier, ContinuousModulusWitness, MetricDistanceWitness,

and Cont; carrier ContinuousMapCarrier; classifier hsame on the preserved visible rows; stability from the category-metric decomposition and terminal modulus-extension closure; ledger source, map, target, modulus, certificate, extra modulus, extended modulus, extended certificate, and distance histories; core theorem this bridge; exactness the displayed carrier conclusion; standard bridge deferred; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier category metric terminal extension bridge`

Theorem 130.56 (ContinuousMap prefixed categorical canonical distance exactness)**.** *For histories* $p, source, map, target, modulus, cert, distance, displayed$: BHist*, assume* UnaryHistory(p)*, the categorical and continuity rows*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(source, target, map),\\&\mathsf{ContinuousModulusWitness}(target, modulus, cert),\end{aligned}$$

and the canonical source-target distance classifier

$$\mathsf{hsame}(distance, \mathsf{append}(source, target)).$$

Then the prefixed continuous-map carrier is exactly the prefixed canonical distance classifier:

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(\mathsf{append}(p, source), map, \mathsf{append}(p, target), modulus, \mathsf{append}(p, cert), displayed)\\&\iff \mathsf{hsame}(displayed, \mathsf{append}(\mathsf{append}(p, source), \mathsf{append}(p, target))).\end{aligned}$$

Proof. First assemble the unprefixed central carrier. Apply Theorem 130.15 in the reverse direction to the categorical hom row, the target-side modulus witness, and hsame$(distance, \mathsf{append}(source, target))$. This gives ContinuousMapCarrier$(source, map, target, modulus, cert, distance)$.

With that carrier in hand, apply Theorem 130.22 to the unary prefix p and the central carrier. Its conclusion is exactly the displayed equivalence between the prefixed carrier and the canonical prefixed source-target distance classifier.

The ten-field fit is: traditional target continuous maps read as categorical arrows with visible modulus data under a unary prefix; BEDC source CategoryHomCarrier, ContinuousModulusWitness, hsame, and UnaryHistory; carrier ContinuousMapCarrier; classifier hsame on the displayed prefixed distance; stability from categorical canonical-distance exactness and prefix canonical-distance exactness; ledger prefix, source, map, target, modulus, certificate, and displayed distance histories; core theorem this iff; exactness the displayed prefixed classifier; downstream use compact-continuous and category-bridge normalization; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier prefixed categorical canonical distance exactness`

Theorem 130.57 (ContinuousMap associative composition canonical package)**.** *For histories* $source, mid_1, mid_2, target, mapF, mapG, mapH, mapFG, mapGH, mapL, mapR, modF, modG, modH, modFG, m$ BHist*, assume the three concrete continuous-map carriers*

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(source, mapF, mid_1, modF, certF, distF),\\&\mathsf{ContinuousMapCarrier}(mid_1, mapG, mid_2, modG, certG, distG),\\&\mathsf{ContinuousMapCarrier}(mid_2, mapH, target, modH, certH, distH),\end{aligned}$$

the left-bracketed continuation ledgers

$$\begin{aligned}&\mathsf{Cont}(mapF, mapG, mapFG), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(mid_2, modFG, certFG),\\&\mathsf{Cont}(mapFG, mapH, mapL), \quad \mathsf{Cont}(modFG, modH, modL), \quad \mathsf{Cont}(target, modL, certL),\end{aligned}$$

and the right-bracketed continuation ledgers

$$\begin{array}{lll} \mathsf{Cont}(mapG, mapH, mapGH), & \mathsf{Cont}(modG, modH, modGH), & \mathsf{Cont}(target, modGH, certGH), \\ \mathsf{Cont}(mapF, mapGH, mapR), & \mathsf{Cont}(modF, modGH, modR), & \mathsf{Cont}(target, modR, certR). \end{array}$$

Then the threefold composition data determine the associative canonical certificate package:

1. *the left-bracketed composite carrier closes over the canonical source-target distance ledger,*

$$\mathsf{ContinuousMapCarrier}(source, mapL, target, modL, certL, \mathsf{append}(source, target));$$

2. *the right-bracketed composite carrier closes over the same canonical source-target distance ledger,*

$$\mathsf{ContinuousMapCarrier}(source, mapR, target, modR, certR, \mathsf{append}(source, target));$$

3. *the two composite graph-map histories are classified together,*

$$\mathsf{hsame}(mapL, mapR);$$

4. *the two composite modulus histories are classified together,*

$$\mathsf{hsame}(modL, modR);$$

5. *every displayed carrier with either bracketed visible fields has the canonical distance ledger:*

$$\mathsf{ContinuousMapCarrier}(source, mapL, target, modL, certL, displayedL) \Rightarrow \mathsf{hsame}(displayedL, \mathsf{append}(source, t$$
$$\mathsf{ContinuousMapCarrier}(source, mapR, target, modR, certR, displayedR) \Rightarrow \mathsf{hsame}(displayedR, \mathsf{append}(source,$$

Proof. Build the left bracketing in two closure steps. First apply Theorem 130.47 to the $source \to mid_1$ and $mid_1 \to mid_2$ carriers, using the graph, modulus, and certificate ledgers ending at $certFG$. Apply the same theorem again to that intermediate carrier and the $mid_2 \to target$ carrier, using the ledgers ending at $certL$. This gives the left composite carrier with distance $\mathsf{append}(source, target)$.

Build the right bracketing symmetrically. Apply Theorem 130.47 to the $mid_1 \to mid_2$ and $mid_2 \to target$ carriers, using the ledgers ending at $certGH$. Then compose the $source \to mid_1$ carrier with that intermediate carrier using the ledgers ending at $certR$. This gives the right composite carrier with the same displayed distance ledger $\mathsf{append}(source, target)$.

For the graph-map row, apply continuation associativity up to history sameness, Theorem 4.56, to $\mathsf{Cont}(mapF, mapG, mapFG)$, $\mathsf{Cont}(mapFG, mapH, mapL)$, $\mathsf{Cont}(mapG, mapH, mapGH)$, and $\mathsf{Cont}(mapF, mapGH, mapR)$. This is the paper form of the checked kernel theorem `continuation_associativity_` uniqueness of the associative continuation endpoint is also recorded by `cont_assoc_unique`.

For the modulus row, use the same continuation-associativity argument on $\mathsf{Cont}(modF, modG, modFG)$, $\mathsf{Cont}(modFG, modH, modL)$, $\mathsf{Cont}(modG, modH, modGH)$, and $\mathsf{Cont}(modF, modGH, modR)$. The resulting classifier row is $\mathsf{hsame}(modL, modR)$.

Finally apply Theorem 130.50 to the left two-step composition data for any displayed left-bracketed carrier, and apply the same theorem to the right two-step composition data for any displayed right-bracketed carrier. The two applications give the two displayed distance-determinacy clauses. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier composition associative canonical package`

Theorem 130.58 (ContinuousMap associative composition certificate determinacy)**.** *Under the hypotheses of Definition 130.57, the two target-modulus certificate ledgers selected by the two bracketings determine the same certificate history:*

$$\mathsf{hsame}(certL, certR).$$

Proof. Apply Definition 130.57 to the three input carriers and the two displayed continuation spines. Its modulus row gives $\mathsf{hsame}(modL, modR)$. Combine this modulus sameness with reflexivity of the target history and the two target-modulus continuation ledgers $\mathsf{Cont}(target, modL, certL)$ and $\mathsf{Cont}(target, modR, certR)$. The continuation respect theorem for hsame then gives $\mathsf{hsame}(certL, certR)$. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier composition associative certificate deterministic`

Theorem 130.59 (ContinuousMap associative composition depth synchronization)**.** *For histories* $source, mid_1, mid_2, target, mapF, mapG, mapH, mapFG, mapGH, mapL, mapR, modF, modG, modH, modFG, m$ BHist*, assume the three concrete continuous-map carriers*

$$\begin{aligned}&\mathsf{ContinuousMapCarrier}(source, mapF, mid_1, modF, certF, distF),\\&\mathsf{ContinuousMapCarrier}(mid_1, mapG, mid_2, modG, certG, distG),\\&\mathsf{ContinuousMapCarrier}(mid_2, mapH, target, modH, certH, distH),\end{aligned}$$

the left-bracketed continuation ledgers

$$\begin{aligned}&\mathsf{Cont}(mapF, mapG, mapFG), \quad \mathsf{Cont}(modF, modG, modFG), \quad \mathsf{Cont}(mid_2, modFG, certFG),\\&\mathsf{Cont}(mapFG, mapH, mapL), \quad \mathsf{Cont}(modFG, modH, modL), \quad \mathsf{Cont}(target, modL, certL),\end{aligned}$$

and the right-bracketed continuation ledgers

$$\begin{aligned}&\mathsf{Cont}(mapG, mapH, mapGH), \quad \mathsf{Cont}(modG, modH, modGH), \quad \mathsf{Cont}(target, modGH, certGH),\\&\mathsf{Cont}(mapF, mapGH, mapR), \quad \mathsf{Cont}(modF, modGH, modR), \quad \mathsf{Cont}(target, modR, certR).\end{aligned}$$

Then both bracketings synchronize the endpoint depth with their displayed composite graph histories:

$$\begin{aligned}&\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapL),\\&\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapR).\end{aligned}$$

Proof. First build the left intermediate carrier. Apply Theorem 130.47 to the first two continuous-map carriers and the ledgers ending at $certFG$. Project its continuous-function half, and project the continuous-function half of the $mid_2 \to target$ carrier.

Apply Theorem 130.43 to those two continuous-function carriers and the left-bracketed ledgers ending at $certL$. This gives

$$\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapL).$$

Next build the right intermediate carrier. Apply Theorem 130.47 to the second and third continuous-map carriers and the ledgers ending at $certGH$. Project its continuous-function half, and project the continuous-function half of the $source \to mid_1$ carrier.

Apply Theorem 130.43 to the $source \to mid_1$ continuous-function carrier, the projected $mid_1 \to target$ continuous-function carrier, and the right-bracketed ledgers ending at $certR$. This gives

$$\mathsf{MetricDistanceDepth}(target) = \mathsf{MetricDistanceDepth}(source) + \mathsf{MetricDistanceDepth}(mapR).$$

The two displayed equalities are exactly the depth-synchronization conclusion. □

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier associative composition depth synchronizatio`

Theorem 130.60 (ContinuousMap identity sandwich coherence)**.** *For histories* $source, target, map, modulus, cert$ BHist*, assume*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance),$$

and assume the left-bracketed identity-sandwich ledger spine

$$\begin{array}{lll} \mathsf{Cont}(\mathsf{e}, map, map_{LF}), & \mathsf{Cont}(\mathsf{e}, modulus, mod_{LF}), & \mathsf{Cont}(target, mod_{LF}, cert_{LF}), \\ \mathsf{Cont}(map_{LF}, \mathsf{e}, map_L), & \mathsf{Cont}(mod_{LF}, \mathsf{e}, mod_L), & \mathsf{Cont}(target, mod_L, cert_L), \end{array}$$

and the right-bracketed identity-sandwich ledger spine

$$\begin{array}{lll} \mathsf{Cont}(map, \mathsf{e}, map_{RT}), & \mathsf{Cont}(modulus, \mathsf{e}, mod_{RT}), & \mathsf{Cont}(target, mod_{RT}, cert_{RT}), \\ \mathsf{Cont}(\mathsf{e}, map_{RT}, map_R), & \mathsf{Cont}(\mathsf{e}, mod_{RT}, mod_R), & \mathsf{Cont}(target, mod_R, cert_R). \end{array}$$

Then the two bracketed identity sandwiches are coherent with the displayed carrier:

1. *the left-bracketed sandwich closes as*

$$\mathsf{ContinuousMapCarrier}(source, map_L, target, mod_L, cert_L, \mathsf{append}(source, target));$$

2. *the right-bracketed sandwich closes as*

$$\mathsf{ContinuousMapCarrier}(source, map_R, target, mod_R, cert_R, \mathsf{append}(source, target));$$

3. *the selected graph histories are the original graph history,*

$$\mathsf{hsame}(map_L, map) \wedge \mathsf{hsame}(map_R, map);$$

4. *the selected modulus histories are the original modulus history,*

$$\mathsf{hsame}(mod_L, modulus) \wedge \mathsf{hsame}(mod_R, modulus);$$

5. *the selected certificate histories and the original certificate history are pairwise classified together,*

$$\mathsf{hsame}(cert_L, cert) \wedge \mathsf{hsame}(cert_R, cert) \wedge \mathsf{hsame}(cert_L, cert_R).$$

Lean4 ✓ `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier identity sandwich coherence`

Proof. Apply Definition 130.40 to the displayed ContinuousMapCarrier. This supplies the source and target empty identity carriers used as the outer factors of the sandwich.

For the left bracketing, first apply Definition 130.47 to the source identity carrier and the displayed carrier, using the ledgers ending at $cert_{LF}$. Apply the same closure theorem again to that intermediate carrier and the target identity carrier, using the ledgers ending at $cert_L$. This gives the left sandwich carrier over $\mathsf{append}(source, target)$.

For the right bracketing, first compose the displayed carrier with the target identity carrier, using the ledgers ending at $cert_{RT}$. Then compose the source identity carrier with that intermediate carrier, using the ledgers ending at $cert_R$. The same closure theorem gives the right sandwich carrier over the same canonical source-target distance ledger.

Apply Definition 130.57 to the source identity carrier, the displayed carrier, and the target identity carrier. Its graph and modulus rows identify the two bracketed selected histories. Apply Definition 130.58 to the same three-carrier spine to get $\mathsf{hsame}(cert_L, cert_R)$.

The remaining readback is by continuation units. The ledgers $\mathsf{Cont}(\mathsf{e}, map, map_{LF})$ and $\mathsf{Cont}(map_{LF}, \mathsf{e}, map_L)$ combine `cont_left_unit_result` and `cont_right_unit_result` to give $\mathsf{hsame}(map_L, map)$, and the right-bracketed graph ledgers give $\mathsf{hsame}(map_R, map)$ in the same way. The identical argument applied to *modulus* gives the two modulus classifiers.

Project the original certificate ledger $\mathsf{Cont}(target, modulus, cert)$ from the carrier. With target reflexivity, the modulus classifiers, and the displayed target-modulus certificate ledgers, `cont_respects_hsame` identifies $cert_L$ and $cert_R$ with $cert$. Together with the associative certificate row, these are exactly the pairwise certificate classifiers.

The ten-field fit is: traditional target identity-sandwich coherence for uniformly continuous maps, BEDC source BHist with Cont, hsame, and ContinuousMapCarrier, carrier ContinuousMapCarrier, classifier hsame, stability from unit continuation and associative composition, ledger graph/modulus/certificate histories, core theorem this coherence package, exactness the canonical source-target distance rows, standard bridge deferred, and status ProofTarget. □

↪ variant `BEDC.Derived.ContinuousMapUp.ContinuousMapCarrier identity sandwich coherence`

The sibling lattice for $\mathsf{ContinuousMap}^{\uparrow}$ is horizontal rather than kernel-isolated. It consumes the $\mathsf{Met}^{\uparrow}$ distance witness from chapter 57, the $\mathsf{Continuous}^{\uparrow}$ pointwise-modulus rows from chapter 59, and the $\mathsf{Cat}^{\uparrow}$ graph, identity, and composition rows from chapter 61. Downstream compact-uniform packets consume this $\mathsf{ContinuousMap}^{\uparrow}$ surface as the common entry point for finite metric rows, pointwise continuity certificates, and categorical composition data.

Theorem 130.61 (ContinuousMap sibling lattice exactness)**.** *An accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *packet is admitted exactly through the* $\mathsf{Met}^{\uparrow}$ *distance witness, the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus rows, and the* $\mathsf{Cat}^{\uparrow}$ *graph, identity, and composition rows, together with the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

The admitted surface excludes host functions, function-space topology, pointwise convergence, uniform convergence, and standard bridge rows. In particular, downstream compact, complete, and uniform consumers may read only the horizontal sibling lattice determined by the metric, continuity, and category coordinates plus the displayed BEDC support rows.

Proof. Apply Definition 130.2 to split the carrier into the categorical hom row, the target-side modulus row, and the metric-distance row. These are precisely the $\mathsf{Cat}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ sibling coordinates named in the theorem statement.

Apply Definition 130.39 to read the same surface through the certificate packet. The bridge exposes the visible BHist, hsame, Cont, Pkg, and NameCert bookkeeping without adding a host function or a function-space topology.

Apply Definition 130.16 to the composition rows. The package shows that identities and composites remain inside the same category-metric lattice and that the displayed distance ledger is fixed by the finite carrier data.

Finally apply Definition 130.34. Its consumer route uses the ContinuousMap surface as input while refusing pointwise convergence, uniform convergence as an ambient topology, and standard bridge rows. Hence the exact admitted surface is the stated horizontal lattice and no larger host-level structure is exported. □

Chapter 131

A Concrete Naming Certificate for $\mathsf{Eq}^{\uparrow}$

The equality-type interface $\mathsf{Eq}^{\uparrow}$ packages the constructive identity type over a carrier, with reflexivity as the sole constructor and explicit transport for kernel-licit substitution. It supplies the foundational equality layer underlying every psame classifier.

Its visible endpoint absurdity and anchor-readback rows supply the same constructor-separation discipline consumed by the $\mathsf{Bool}^{\uparrow}$ finite-data classifier of chapter 32. In particular, $\mathsf{Bool}^{\uparrow}$ may use the empty-anchor and visible-endpoint no-confusion rows of $\mathsf{Eq}^{\uparrow}$ as the equality-class boundary for branch classifiers.

131.1 Carrier definition

Theorem 131.1 (EqType e1 anchor tail readback)**.** *For BEDC histories anchor, tail, and h, if h belongs to the equality-type class anchored at* $\mathsf{E1}(anchor)$ *and* $h\mathsf{hsameE1}(tail)$*, then* $tail\mathsf{hsame}anchor$*.*

Proof. The class predicate is the transparent history-sameness class. Compose the class comparison with the reverse of the displayed comparison to obtain $\mathsf{E1}(tail)\mathsf{hsameE1}(anchor)$, then invert the e1 constructor under hsame. □

Closure status for $\mathsf{NameCert}_{\mathsf{Eq}^{\uparrow}}$**.**

Constructive story: The equality-type certificate is generated from the minimal $BHist$ anchors that witness equality classes. The carrier first reads the anchor tail, visible context, and pair rows that identify when a history is in the class of another history. The $hsame$ relation is the generating comparison, so transport is not a host equality assertion but a ledger-visible movement from one anchor to another. Continuations enter only through contexts whose visible core is already determined by the anchor rows. The chapter then derives determinacy, absurdity for incompatible anchors, semantic certification, anchor transport, and transitivity from those visible class-carrier conventions.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Eq}^{\uparrow}}$, seed)

Scope closed: This chapter binds the EqType carrier sections, anchor readback and determinacy theorems, identity semantic NameCert, anchor transport, and transitivity.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Eq}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim host definitional equality, quotient equality of all histories, or equality principles beyond the displayed $hsame$ class carrier.

Upgrade path: Reaching obligationClosure requires a formal EqType carrier whose anchor readback, transport, and transitivity theorems are checked in Lean.

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier e1 anchor tail readback`

Theorem 131.2 (EqType e0 anchor tail readback)**.** *For BEDC histories anchor, tail, and h, if h belongs to the equality-type class anchored at* E0(*anchor*) *and h*hsameE0(*tail*)*, then tail*hsame*anchor.*

Proof. The class predicate is the transparent history-sameness class. Compose the class comparison with the reverse of the displayed comparison to obtain E0(*tail*)hsameE0(*anchor*), then invert the e0 constructor under hsame. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier e0 anchor tail readback`

Theorem 131.3 (EqType e0 anchor pair determinism)**.** *For BEDC histories anchor, left, and right, if both left and right belong to the equality-type class anchored at* E0(*anchor*)*, then left*hsame*right.*

Proof. Both carrier witnesses are concrete comparisons with the same anchor E0(*anchor*). Compose the first comparison with the reverse of the second comparison to obtain *left*hsame*right.* □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier e0 anchor pair deterministic`

Theorem 131.4 (EqType e0 anchor e1 tail absurdity)**.** *For BEDC histories anchor, tail, and h, if h belongs to the equality-type class anchored at* E0(*anchor*) *and h*hsameE1(*tail*)*, then contradiction follows.*

Proof. The carrier comparison identifies *h* with E0(*anchor*). Composing it with the reverse of the displayed comparison gives E1(*tail*)hsameE0(*anchor*), which is impossible by constructor separation for history sameness. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier e0 anchor e1 tail absurd`

Theorem 131.5 (EqType e1 anchor e0 tail absurdity)**.** *For BEDC histories anchor, tail, and h, if h belongs to the equality-type class anchored at* E1(*anchor*) *and h*hsameE0(*tail*)*, then contradiction follows.*

Proof. The carrier comparison identifies *h* with E1(*anchor*). Composing it with the reverse of the displayed comparison gives E0(*tail*)hsameE1(*anchor*), which is impossible by constructor separation for history sameness. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier e1 anchor e0 tail absurd`

Theorem 131.6 (EqType empty anchor visible absurdity)**.** *For BEDC histories tail and h, if h belongs to the equality-type class anchored at* e*, then h cannot also classify to either visible endpoint* E0(*tail*) *or* E1(*tail*)*.*

Proof. The class predicate unfolds to *h*hsamee. Compose its reverse with the displayed visible comparison. The zero-headed branch gives ehsameE0(*tail*), and the one-headed branch gives ehsameE1(*tail*). Constructor separation for histories rules out both comparisons. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier empty anchor visible absurd`

Theorem 131.7 (EqType visible-context anchor readback)**.** *For BEDC histories anchor, h, and core, if h belongs to the equality-type class anchored at anchor and the display of h inside a common left and right append context agrees with the corresponding display of core, then core*hsame*anchor.*

Proof. Reassociate the two displays into a common-context form and cancel the common prefix and suffix, obtaining *h*hsame*core*. Symmetry composes this readback with the carrier comparison *h*hsame*anchor*, so *core*hsame*anchor*. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier visible context anchor readback`

Theorem 131.8 (EqType visible-context e0 anchor e1 tail absurdity)**.** *For BEDC histories p, r, anchor, h, and tail, if h belongs to the equality-type class anchored at* $\mathsf{E0}(anchor)$*, then the display of h inside a common visible context cannot agree with the corresponding display of* $\mathsf{E1}(tail)$*:*

$$\mathsf{EqtypeClassCarrier}(\mathsf{E0}(anchor), h) \Longrightarrow \mathsf{hsame}((p \cdot h) \cdot r, (p \cdot \mathsf{E1}(tail)) \cdot r) \Longrightarrow \mathsf{False}.$$

Proof. Visible-context anchor readback identifies the displayed core $\mathsf{E1}(tail)$ with the zero-headed anchor $\mathsf{E0}(anchor)$. BEDC history constructor separation rules out this $\mathsf{E1}/\mathsf{E0}$ comparison. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier visible context e0 anchor e1 tail absurd`

Theorem 131.9 (EqType visible-context empty anchor visible core absurdity)**.** *For BEDC histories p, r, h, core, and tail, if h belongs to the equality-type class anchored at* e *and the common visible context displays h as core, then core cannot classify to either visible endpoint* $\mathsf{E0}(tail)$ *or* $\mathsf{E1}(tail)$*.*

Proof. Visible-context anchor readback identifies *core* with the empty anchor. The empty-anchor visible absurdity theorem then rules out both visible endpoint alternatives for *core* by BEDC history constructor separation. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier visible context empty anchor visible core absurd`

Theorem 131.10 (EqType visible-context core determinism)**.** *For BEDC histories p, r, anchor, $left$, $right$, $coreLeft$, and $coreRight$, if both $left$ and $right$ belong to the equality-type class anchored at anchor, and the common visible context displays $left$ as $coreLeft$ and $right$ as $coreRight$, then $coreLeft\,\mathsf{hsame}\,coreRight$.*

Proof. Apply visible-context anchor readback to both displays. This identifies each displayed core with the shared anchor. Composing the left readback with the reverse of the right readback gives the core comparison. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier visible context cores deterministic`

Theorem 131.11 (EqType visible-context pair determinacy)**.** *For BEDC histories p, r, anchor, $left$, $right$, $leftCore$, and $rightCore$, suppose $left$ and $right$ both belong to the equality-type class anchored at anchor. If the display of $left$ inside the common context p, r agrees with the corresponding display of $leftCore$, and the display of $right$ inside the same common context agrees with the display of $rightCore$, then $leftCore\,\mathsf{hsame}\,rightCore$.*

Proof. Apply visible-context anchor readback to the left comparison to obtain $leftCore\,\mathsf{hsame}\,anchor$. Apply the same readback to the right comparison to obtain $rightCore\,\mathsf{hsame}\,anchor$. The first comparison and the reverse of the second compose by transitivity of hsame, giving $leftCore\,\mathsf{hsame}\,rightCore$. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier visible context pair deterministic`

131.2 The certificate

Theorem 131.12 (EqType identity semantic name certificate)**.** *The identity classifier on histories forms a semantic name certificate when source, pattern, and ledger are all the self-identity predicate:*

$$\mathsf{SemanticNameCert}(\lambda h.\ h\mathsf{hsame}h, \lambda h.\ h\mathsf{hsame}h, \lambda h.\ h\mathsf{hsame}h, \mathsf{hsame}).$$

Proof. The empty history supplies inhabitation. Reflexivity, symmetry, transitivity, and carrier transport are exactly the corresponding closure laws of hsame. The pattern and ledger maps are the identity maps on the self-identity predicate. □

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeIdentity semanticNameCert`

Theorem 131.13 (EqType class carrier anchor transport)**.** *For BEDC histories anchor, anchor′, and h, if h belongs to the equality-type class anchored at anchor and anchor*hsame*anchor′, then h belongs to the equality-type class anchored at anchor′:*

$$\mathsf{EqtypeClassCarrier}(anchor, h) \Longrightarrow anchor\mathsf{hsame}anchor' \Longrightarrow \mathsf{EqtypeClassCarrier}(anchor', h).$$

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier anchor hsame transport`

Proof. By the carrier definition in section 131.1, membership in the class anchored at *anchor* is the comparison $h\mathsf{hsame}anchor$. Compose this comparison with the displayed anchor comparison $anchor\mathsf{hsame}anchor'$ by transitivity of hsame, obtaining $h\mathsf{hsame}anchor'$. This is exactly the carrier witness for the equality-type class anchored at $anchor'$. □

Theorem 131.14 (EqType class carrier transitivity)**.** *For BEDC histories anchor, h, and h′, equality-type class membership is transitive along class anchors:*

$$\mathsf{EqtypeClassCarrier}(anchor, h) \Longrightarrow \mathsf{EqtypeClassCarrier}(h, h') \Longrightarrow \mathsf{EqtypeClassCarrier}(anchor, h').$$

Lean4 ✓ `BEDC.Derived.EqtypeUp.EqtypeClassCarrier transitive`

Proof. By the carrier definition in section 131.1, the first hypothesis is the comparison $h\mathsf{hsame}anchor$. Apply Definition 131.13 to the second hypothesis, viewed as membership of h' in the class anchored at h, and to this anchor comparison $h\mathsf{hsame}anchor$. The result is exactly membership of h' in the class anchored at *anchor*. □

Chapter 132

A Concrete Naming Certificate for $\mathsf{Func}^{\uparrow}$

The function-object interface $\mathsf{Func}^{\uparrow}$ packages the carrier of functions $A \to B$ between two naming certificates, together with the pointwise hsame-respecting structure. It supplies the substrate for $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ (chapter 48) and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ (chapter 130) as restrictions of the general function object.

132.1 Carrier definition

Theorem 132.1 (Function object linear singleton continuous-map components are empty)**.** *For histories source, map, target, modulus, cert, distance* : BHist*, suppose* ContinuousMapCarrier(*source, map, target, m holds and the source, map, and modulus histories lie in the linear-map singleton carrier. Then*

$$\mathsf{hsame}(target, \mathsf{e}) \wedge \mathsf{hsame}(cert, \mathsf{e}) \wedge \mathsf{hsame}(distance, \mathsf{e}).$$

Proof. The singleton carrier identifies *source*, *map*, and *modulus* with e. The continuous-map graph continuation compares with the right-unit continuation at e, forcing *target*hsamee. The certificate continuation then compares against the same empty right unit, forcing *cert*hsamee. Finally, the metric-distance continuation from *source* to *target* compares against the empty right unit after both endpoints have been identified with e, so *distance*hsamee. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map components empty`

Theorem 132.2 (Function object linear singleton continuous-map graphs are empty)**.** *For histories source, map, target, modulus, cert, distance* : BHist*, suppose* ContinuousMapCarrier(*source, map, target, modul holds and the source, map, and modulus histories lie in the linear-map singleton carrier. Then the displayed graph, certificate, and distance continuations can be read over the empty components:*

$$\mathsf{Cont}(source, map, \mathsf{e}) \wedge \mathsf{Cont}(\mathsf{e}, modulus, cert) \wedge \mathsf{Cont}(source, \mathsf{e}, distance).$$

Proof. The component-empty theorem identifies *target*, *cert*, and *distance* with e. Transport the continuous-map graph continuation across *target*hsamee. Transport the certificate continuation across *target*hsamee on the source side, and transport the metric-distance continuation across *target*hsamee on the right endpoint. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map empty graphs`

Theorem 132.3 (Function object linear singleton continuous-map empty components iff)**.** *For histories* $source, map, target, modulus, cert, distance : \mathsf{BHist}$*, if the source, map, and modulus histories lie in the linear-map singleton carrier, then*

$$\mathsf{ContinuousMapCarrier}(source, map, target, modulus, cert, distance) \iff \mathsf{hsame}(target, \mathsf{e}) \wedge \mathsf{hsame}(cert, \mathsf{e}) \wedge \mathsf{hsame}(d$$

Proof. The forward direction is the component-empty theorem for linear singleton continuous maps. Conversely, the three singleton carrier assumptions identify *source*, *map*, and *modulus* with e, while the displayed component equalities identify *target*, *cert*, and *distance* with e. The continuous-function carrier and metric-distance witness are then assembled from unary emptiness and the empty right-unit continuation rows. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map empty components iff`

Theorem 132.4 (Function object linear singleton continuous-map target graph empty iff)**.** *For histories* $source, map, target, modulus, cert, distance : \mathsf{BHist}$*, suppose* $\mathsf{ContinuousMapCarrier}(source, map, target, m$ *holds and the source and map histories lie in the linear-map singleton carrier. Then the empty graph continuation is equivalent to the target endpoint being empty:*

$$\mathsf{Cont}(source, map, \mathsf{e}) \iff \mathsf{hsame}(target, \mathsf{e}).$$

Proof. The continuous-map carrier contains the graph continuation $\mathsf{Cont}(source, map, target)$. In the forward direction, the singleton carrier identifies *source* and *map* with e, so this graph continuation compares with the right-unit continuation at e and forces $target\mathsf{hsame}\mathsf{e}$. Conversely, if $target\mathsf{hsame}\mathsf{e}$, transport the graph continuation result along this history comparison to obtain $\mathsf{Cont}(source, map, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map target graph empty iff`

Theorem 132.5 (Function object linear singleton continuous-map cert graph empty iff)**.** *For histories* $source, map, target, modulus, cert, distance : \mathsf{BHist}$*, suppose* $\mathsf{ContinuousMapCarrier}(source, map, target, modul$ *holds and the target and modulus histories lie in the linear-map singleton carrier. Then the empty certificate graph is equivalent to the certificate being empty:*

$$\mathsf{Cont}(target, modulus, \mathsf{e}) \iff \mathsf{hsame}(cert, \mathsf{e}).$$

Proof. The continuous-map carrier contains the certificate continuation $\mathsf{Cont}(target, modulus, cert)$. The forward direction compares it with the displayed empty certificate graph by continuation determinism. Conversely, if $cert\mathsf{hsame}\mathsf{e}$, transport the result of the certificate continuation to obtain $\mathsf{Cont}(target, modulus, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map cert graph empty iff`

Theorem 132.6 (Function object linear singleton continuous-map distance graph empty iff)**.** *For histories* $source, map, target, modulus, cert, distance : \mathsf{BHist}$*, suppose* $\mathsf{ContinuousMapCarrier}(source, map, target, m$ *holds. Then the empty distance graph is equivalent to the distance endpoint being empty:*

$$\mathsf{Cont}(source, target, \mathsf{e}) \iff \mathsf{hsame}(distance, \mathsf{e}).$$

Proof. The continuous-map carrier contains the distance continuation $\mathsf{Cont}(source, target, distance)$. The forward direction compares this witness with the displayed empty distance graph by continuation determinism. Conversely, if $distance\mathsf{hsame}\mathsf{e}$, transport the continuation result to obtain $\mathsf{Cont}(source, target, \mathsf{e})$. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map distance graph empty iff`

Lemma 132.7 (Function object linear singleton continuous-map distance graph empty)**.** *For histories* $source, map, target, modulus, cert, distance$: BHist*, suppose* ContinuousMapCarrier$(source, map, target, modul$ *holds and the source, map, and modulus histories lie in the linear-map singleton carrier. Then*

$$\mathsf{Cont}(source, target, \mathsf{e}).$$

Proof. The component-empty theorem identifies the distance witness with e. The continuous-map carrier contains the metric-distance continuation $\mathsf{Cont}(source, target, distance)$, and result transport along $distance$hsamee gives the displayed empty distance graph. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncObjLinearSingleton continuous map distance graph empty`

132.2 The certificate

Theorem 132.8 (Function-object pointwise hom carrier semantic certificate)**.** *For a unary prefix* p *and a categorical hom carrier* CategoryHomCarrier(a, b, f)*, the pointwise function-object carrier*

$$\mathsf{CategoryHomCarrier}(a, b, t) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, b), t)$$

has a semantic naming certificate with source, pattern, and ledger predicates all equal to this carrier and classifier hsame*.*

Proof. The witness is the original hom history f. Prefix preservation for the unary-continuation category supplies the prefixed hom carrier at the same history. The classifier laws are reflexivity, symmetry, and transitivity of hsame. Transport along a classifier proof keeps the endpoints fixed and moves both the base and prefixed hom-carrier conjuncts across the displayed history comparison. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncobjPointwiseHomCarrier semanticNameCert`

Theorem 132.9 (Function-object pointwise hom carrier composition readback)**.** *For a unary prefix* p*, categorical hom carriers* CategoryHomCarrier(a, b, f) *and* CategoryHomCarrier(b, c, g)*, and a continuation* Cont(f, g, fg)*, the composite history* fg *carries both the base composite hom carrier and the pointwise prefixed composite hom carrier:*

$$\mathsf{CategoryHomCarrier}(a, c, fg) \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, c), fg).$$

Moreover, every displayed history carrying both of these composite carrier fields is hsame *to* fg*.*

Proof. The base composite is supplied by the categorical composition readback theorem. Prefix preservation transports that base composite through the unary prefix p, giving the pointwise prefixed carrier at the same history fg. If a displayed history carries both fields, the base field alone is enough for the categorical readback uniqueness proof, which identifies the displayed history with fg. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncobjPointwiseHomCarrier comp public readback`

Theorem 132.10 (Function-object pointwise hom carrier composite transport)**.** *For a unary prefix p, categorical hom carriers* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and* $\mathsf{CategoryHomCarrier}(b, c, g)$*, a continuation* $\mathsf{Cont}(f, g, fg)$*, and a history comparison* $fg\mathsf{hsame}fg'$*, the transported composite history* fg' *carries both the base composite hom carrier and the pointwise prefixed composite hom carrier:*

$$\mathsf{CategoryHomCarrier}(a, c, fg') \wedge \mathsf{CategoryHomCarrier}(\mathsf{append}(p, a), \mathsf{append}(p, c), fg').$$

Proof. First read back the base and prefixed composite carriers at fg from the pointwise composition theorem. Then transport each carrier along $fg\mathsf{hsame}fg'$, keeping the source and target endpoints fixed by reflexive history sameness. □

Lean4 ✓ `BEDC.Derived.FuncobjUp.FuncobjPointwiseHomCarrier comp hsame transport`

Closure status for $\mathsf{NameCert}_{\mathsf{Func}\uparrow}$**.**

Constructive story: The function-object certificate is generated from $BHist$ rows that name source histories, target histories, hom carriers, pointwise prefixes, and composite histories. The carrier convention does not begin with a host function space; it reads a graph or hom row from the visible ledger and then uses $hsame$ to transport that row across representative changes. Continuation data appear as $\mathsf{Cont}(f, g, fg)$, the explicit composition witness for two hom rows. The chapter first eliminates empty linear-singleton graph components, then builds the pointwise hom carrier and its semantic NameCert, composition readback, and composite transport from those visible rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Func}\uparrow}, \mathsf{seed})$

Scope closed: This chapter binds the function-object carrier sections, empty graph component theorems, pointwise hom semantic NameCert, composition readback, and composite transport.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Func}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim an ambient exponential object, total host function equality, or composition outside the recorded Cont witness.

Upgrade path: Reaching obligationClosure requires a formal hom-carrier packet and checked preservation of pointwise composition under $hsame$.

Chapter 133

A Concrete Naming Certificate for $\mathsf{RiemannianMetric}^{\uparrow}$

The Riemannian-metric interface $\mathsf{RiemannianMetric}^{\uparrow}$ packages a smooth field of positive-definite inner products on the tangent fibres of a manifold carrier. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ (chapter 86), and supplies the metric substrate for length, geodesic, and curvature certificates.

133.1 Carrier definition

Theorem 133.1 (RiemannianMetricUp source fibre carrier row)**.** *A* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *certificate carries the following source-fibre rows. Its base carrier is a* $\mathsf{Manifold}^{\uparrow}$ *point carrier over* BHist*. For each carried point history* x*, it displays a tangent fibre row* T_x *over* x*. For each displayed fibre* T_x*, it displays an* $\mathsf{InnerProduct}^{\uparrow}$ *row*

$$g_x : T_x \times T_x \to \mathsf{BHist}$$

on that fibre. The visible ledger records only the manifold point provenance, the tangent-witness provenance over the point, and the fibre inner-product provenance; it contains no length, geodesic, curvature, or bridge row.

Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricSingleton source fibre carrier row`
Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricSingletonFibreSurface carrier rows`

Proof. Read the certificate through the ten-field $\mathsf{NameCert}$ template. The only outer carrier named by the template is the $\mathsf{Manifold}^{\uparrow}$ carrier, so every base point is a BHist point accepted by that source.

For a displayed point x, the fibre entry is a dependent row over the same BHist witness. The row has no independent endpoint: its source is the carried $\mathsf{Manifold}^{\uparrow}$ point and its classifier is inherited through the point row.

On each such fibre, the metric entry is exactly an $\mathsf{InnerProduct}^{\uparrow}$ row. The ledger projection of the ten-field template therefore has three visible provenance classes: the $\mathsf{Manifold}^{\uparrow}$ point row, the tangent row over that point, and the $\mathsf{InnerProduct}^{\uparrow}$ row on the displayed fibre. □

Theorem 133.2 (RiemannianMetricUp classifier stability obligation)**.** *For a* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *certificate, classifier stability has the following obligation shape. If point histories* x *and* y *are related by* hsame *and by the* $\mathsf{Manifold}^{\uparrow}$ *point classifier, then every tangent row over* x *transports to*

a tangent row over y. If tangent witnesses u, v over x transport to tangent witnesses u', v' over y, then the $\mathsf{InnerProduct}^{\uparrow}$ *classifier transports the fibrewise metric endpoint*

$$g_x(u, v)$$

to

$$g_y(u', v').$$

Proof. Start with the ten-field NameCert classifier field. Its base comparison is the hsame comparison on BHist histories together with the $\mathsf{Manifold}^{\uparrow}$ point classifier, so the transported endpoint remains inside the same carried point surface.

The tangent row is dependent on that transported point. Applying the NameCert stability field moves the displayed tangent witness from the source point to the target point without adding any classifier beyond hsame and the $\mathsf{Manifold}^{\uparrow}$ point classifier.

It remains to compare metric endpoints. The metric row on each fibre is an $\mathsf{InnerProduct}^{\uparrow}$ row, so its own classifier transports the endpoint through the transported tangent witnesses and yields the displayed target endpoint. □

Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricSingletonFibreSurface classifier stability`

Theorem 133.3 (RiemannianMetricUp fibrewise positivity and chart compatibility obligation)**.** *A* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *certificate carries the following fibrewise metric obligations. For every displayed point x and nonzero tangent witness $u : T_x$, the fibre row has a positive-definite entry*

$$g_x(u, u) > 0.$$

For every displayed fibre, symmetry and bilinearity are inherited from the $\mathsf{InnerProduct}^{\uparrow}$ *row on that fibre. For every overlap in the* $\mathsf{Manifold}^{\uparrow}$ *atlas, chart-change transport carries tangent witnesses through the atlas classifier and preserves the displayed fibre metric endpoint under the* $\mathsf{InnerProduct}^{\uparrow}$ *classifier. These obligations do not assert length, geodesic, curvature, or a standard bridge.*

Proof. Unpack the ten-field NameCert template at a displayed BHist point. The point is supplied only by $\mathsf{Manifold}^{\uparrow}$, and the tangent witness is supplied only as a fibre row over that point.

The metric row on the fibre is an $\mathsf{InnerProduct}^{\uparrow}$ row. Its positive, symmetric, and bilinear entries give the positive-definite, symmetry, and bilinearity obligations after restricting all arguments to the displayed tangent fibre.

For a chart overlap, use the $\mathsf{Manifold}^{\uparrow}$ atlas classifier to transport the point and tangent witnesses across the overlap. Then apply the $\mathsf{InnerProduct}^{\uparrow}$ classifier on the target fibre to identify the metric endpoint after transport.

The template has no row whose endpoint is a length functional, geodesic equation, curvature tensor, or standard bridge. Hence those interfaces are outside the stated obligation scope. □

Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricSingletonFibreSurface fibrewise compatibility scope`

Theorem 133.4 (RiemannianMetricUp public source-fibre NameCert surface)**.** *The public exported* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *source-fibre surface is exactly the conjunction of the source-fibre carrier row of Definition 133.1, the classifier stability obligation of Definition 133.2, and the fibrewise positivity and chart-compatibility obligation of Definition 133.3. Downstream length, geodesic, and curvature consumers may cite this theorem as the single* $\mathsf{RiemannianMetric}^{\uparrow}$ *surface, but the surface does not assert those downstream interfaces or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricPublicSourceFibre name certificate`

Proof. Project the public $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ surface onto its source-fibre carrier component. This gives exactly the manifold point row, the tangent row over each displayed point, and the $\mathsf{InnerProduct}^{\uparrow}$ metric row of Definition 133.1.

Project the same surface onto its classifier component. The ten-field roadmap template exposes precisely the point transport, tangent-row transport, and fibrewise metric endpoint transport recorded in Definition 133.2.

Finally project the fibrewise metric obligation component. The resulting positive-definite, symmetric, bilinear, and chart-compatible fibre rows are exactly those of Definition 133.3. Repacking these three projections under the roadmap ten-field template gives a single exported source-fibre surface, and the absence of length, geodesic, curvature, and bridge rows is preserved by the same repacking. □

Theorem 133.5 (RiemannianMetricUp source-fibre standard bridge)**.** *The public source-fibre surface of Definition 133.4 gives a conservative paper-level bridge from* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *to the ordinary fibrewise metric presentation. Forward readback exposes exactly the* $\mathsf{Manifold}^{\uparrow}$ *point row, the tangent-fibre row, and the* $\mathsf{InnerProduct}^{\uparrow}$ *metric row of Definition 133.1. Reverse admission accepts only such displayed rows. Classifier preservation is exactly Definition 133.2, and fibre positivity together with chart compatibility is exactly Definition 133.3. The bridge exports no length, geodesic, curvature, or host metric object.*

Proof. Start from the public export theorem Definition 133.4. Its carrier projection is precisely the source-fibre carrier row of Definition 133.1, so forward readback has only the manifold point row, the tangent-fibre row over that point, and the displayed $\mathsf{InnerProduct}^{\uparrow}$ metric row.

For reverse admission, use the same public export theorem in the opposite direction of packaging. A displayed ordinary fibrewise metric row is admitted only when it has the three rows named by Definition 133.1; no additional row is available in the exported surface.

The transport component is supplied by Definition 133.2, while the positive-definite and chart-compatible fibre obligations are supplied by Definition 133.3. Composing these two obligations with the public export theorem gives the conservative bridge, and the same boundary excludes length, geodesic, curvature, and host metric objects. □

Lean4 ✓ `BEDC.Derived.RiemannianMetricUp.RiemannianMetricPublicSourceFibre standard bridge`

133.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$**.**

Constructive story: A Riemannian metric is admitted only after a $\mathsf{Manifold}^{\uparrow}$ point row has been displayed, a tangent-fibre row has been placed over that same history, and an $\mathsf{InnerProduct}^{\uparrow}$ row has been attached fibrewise; histories lacking any one of those three rows stay outside the carrier. Two displayed metric packets classify together when the point histories are matched by the manifold classifier and hsame, the tangent rows transport across that point match, and the inner-product endpoint on transported tangent witnesses is classified fibre by fibre. The source therefore covers exactly manifold points with tangent fibres and inner-product rows, the pattern records positive-definiteness, symmetry, bilinearity, and chart-overlap compatibility on those fibres, and the ledger keeps visible only the point provenance, tangent-witness provenance, and fibre inner-product provenance. Nonemptiness comes from the displayed source-fibre row, while reflexivity, symmetry, transitivity, and carrier closure are inherited componentwise from hsame, $\mathsf{Manifold}^{\uparrow}$, and $\mathsf{InnerProduct}^{\uparrow}$, so the five naming-certificate obligations never ask for a host metric space. A history h lies in the signature gap of a $\mathsf{RiemannianMetric}^{\uparrow}$ token p when the source accepts the manifold point and fibre packet for h, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, this says that p records precisely the same carried point, tangent fibre, and positive compatible inner-product row. The paper bridge

reads the package as an ordinary smooth field of positive-definite inner products on tangent fibres and reads it back by the same three provenance rows. The chapter registers this fibrewise construction under the name RiemannianMetric$^{\uparrow}$; length, geodesics, curvature, and a host metric object are not part of the primitive data.

Theory closure: TheoryGate(NameCert$_{\mathsf{RiemannianMetric}\uparrow}$, maturePackageClosed)

Scope closed: The public paper-side surface is Definition 133.4, and the conservative paper-level standard bridge is Definition 133.5, built from the carrier row of Definition 133.1, the classifier-stability row of Definition 133.2, and the positivity and chart-compatibility row of Definition 133.3.

Formal verification: FormalStatus(NameCert$_{\mathsf{RiemannianMetric}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.RiemannianMetricUp.RiemannianMetricSingleton\_source\_fibre\_carrier\_row`

Bridge status: paperBridge

Not claimed: This chapter does not close an exactness theorem, length/geodesic/curvature interface, Lean-side bridgeChecked verification, or a host metric object.

Upgrade path: Remaining strengthening is Lean-side bridgeChecked verification and downstream length, geodesic, and curvature consumers, not a further paper-side closure grade.

Chapter 134

A Concrete Naming Certificate for $\mathsf{Connection}^{\uparrow}$

The connection interface $\mathsf{Connection}^{\uparrow}$ packages a covariant derivative on a fibre bundle, with linearity in the section argument and the Leibniz rule against scalar functions. It depends on $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$ (chapter 97) and $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96), and supplies the parallel-transport and curvature substrate.

134.1 Carrier definition

Definition 134.1 (Connection carrier packet)**.** *A* $\mathsf{Connection}^{\uparrow}$ *carrier packet is a displayed* BHist *tuple*

$$\mathcal{C} = (\mathcal{B}, \mathcal{M}, b, f, s, t, \nabla, \eta_{\mathcal{B}}, \eta_{\mathcal{M}}, \rho)$$

where $\mathcal{B}$ *is the* $\mathsf{Bundle}^{\uparrow}$ *source certificate,* $\mathcal{M}$ *is the* $\mathsf{Manifold}^{\uparrow}$ *source certificate,* $b : \mathsf{BHist}$ *is a base-point row,* $f : \mathsf{BHist}$ *is the fibre row over* b*,* $s : \mathsf{BHist}$ *is a section row with endpoint in* f*,* $t : \mathsf{BHist}$ *is a tangent-direction row at* b*, and* $\nabla : \mathsf{BHist}$ *is the covariant-derivative endpoint row. The witnesses* $\eta_{\mathcal{B}}$ *and* $\eta_{\mathcal{M}}$ *retain the two dependency* $\mathsf{NameCert}$ *rows, while* ρ *records that the visible source rows are compared only by the inherited* hsame *and classifier data supplied by* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{Manifold}^{\uparrow}$*.*

The packet carries no ambient manifold object beyond those rows. Its public surface is the base point, fibre, section, tangent direction, derivative endpoint, dependency witnesses, and finite Cont *ledger connecting the derivative endpoint back to the displayed source rows.*

Lean4 (definition) `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket`
Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionTasteGate single carrier alignment`

Definition 134.2 (Connection classifier and derivative row)**.** *Two connection packets* $\mathcal{C}$ *and* $\mathcal{C}'$ *are classified by* $\mathsf{ConnCls}$ *when their base-point rows are related by the* $\mathsf{Manifold}^{\uparrow}$ *point classifier, their fibre rows are related by the* $\mathsf{Bundle}^{\uparrow}$ *fibre classifier, their section rows are related by the* $\mathsf{Bundle}^{\uparrow}$ *section classifier, their tangent-direction rows are related by the* $\mathsf{Manifold}^{\uparrow}$ *tangent classifier, and their derivative endpoint rows are related by* hsame*. The* Pkg *provenance rows of the two dependency certificates remain part of the comparison, so a classifier witness also records the source packets from which the displayed derivative rows were read.*

Thus the derivative row is a carried BHist *endpoint, not a host-level function value. It is accepted precisely when the source base, fibre, section, and tangent rows are accepted by the inherited certificates and the endpoint comparison is witnessed by* hsame *inside the same finite history ledger.*

Lean4 (definition) `BEDC.Derived.ConnectionUp.ConnectionClassifierDerivativeRow`

Theorem 134.3 (Connection section-linearity obligation)**.** *For the carrier of Definition 134.1, the derivative row is stable under the section-addition and scalar-action rows supplied by the* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{Manifold}^{\uparrow}$ *dependency sources. If* s_1*,* s_2*, and* a *are certified section and scalar rows over the same base and tangent packet, then the derivative endpoints for* $s_1 + s_2$ *and* $a \cdot s_1$ *are classified by the same* ConnCls *rows obtained by applying the corresponding carried addition and scalar-action ledgers to the derivative endpoints of* s_1 *and* s_2*.*

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket section linearity obligation`

Proof. Read the two section rows, their sum row, and the scalar-action row through the $\mathsf{Bundle}^{\uparrow}$ certificate. The $\mathsf{Manifold}^{\uparrow}$ source keeps the base point and tangent-direction rows fixed under the displayed classifier.

Apply endpoint transport for hsame along the finite Cont ledgers representing section addition and scalar action. The derivative endpoint row is then compared componentwise with the transported endpoint rows, while the Pkg provenance remains the original pair of dependency certificates.

Repack the transported base, fibre, section, tangent, and derivative comparisons into ConnCls. No external vector-space equality is used, because all linearity data is read from the dependency rows carried by the packet. □

Theorem 134.4 (Connection parallel-transport obligation)**.** *Let a displayed path spine be a finite* BHist *chain of base-point rows whose adjacent tangent-direction and derivative endpoint rows are compatible by* ConnCls*. Then the chain determines a parallel-transport packet whose initial and terminal section endpoints remain in the carried fibre classifier of the* $\mathsf{Bundle}^{\uparrow}$ *source.*

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket parallel transport obligation`

Proof. Induct on the displayed path spine. For the empty spine, the transport packet is the identity Cont row at the initial fibre, and the fibre classifier holds by reflexivity of the inherited $\mathsf{Bundle}^{\uparrow}$ classifier.

For a nonempty spine, use the head derivative row to append one certified transport step. The compatibility premise supplies the hsame endpoint comparison needed to move from the current fibre row to the next carried fibre row.

Apply the induction hypothesis to the remaining spine and compose the two finite ledgers by Cont associativity. The terminal endpoint is therefore classified with the carried terminal fibre row by the same $\mathsf{Bundle}^{\uparrow}$ classifier used at every visible step. □

Theorem 134.5 (Connection curvature-boundary obligation)**.** *Given two displayed tangent-direction rows for a connection packet, the commutator of the corresponding derivative rows is a visible* BHist *boundary row. This boundary row is consumable by* $\mathsf{Curvature}^{\uparrow}$ *as the source row for the curvature packet, with base, fibre, section, tangent-pair, derivative-endpoint, and* Pkg *provenance data inherited from Definition 134.1.*

Proof. Build the two derivative composites by appending the first tangent direction and then the second, and by appending them in the reverse order. Each composite is a finite Cont ledger over the same base, fibre, and section rows.

The commutator boundary is the displayed comparison between these two endpoint ledgers. Its source rows are exactly the two tangent-direction rows and the two derivative endpoint rows already accepted by ConnCls.

Package that comparison as a BHist boundary row. Since the packet exposes the dependency certificates and provenance rows, $\mathsf{Curvature}^{\uparrow}$ can read the boundary without adding a new carrier object or quotienting the derivative endpoints. □

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket curvature boundary obligation`

Theorem 134.6 (Connection stability-ledger exactness obligation)**.** *The displayed connection surface has exactly the classifier reflexivity, symmetry, and transitivity rows inherited from* hsame*,* $\mathsf{Bundle}^{\uparrow}$*, and* $\mathsf{Manifold}^{\uparrow}$*; the carrier transport rows generated by those classifiers; the ledger readback rows for base, fibre, section, tangent, derivative, and* Pkg *provenance; and the exactness rows for section-linearity, parallel transport, and curvature-boundary formation. No additional coordinate chart, host connection operator, or standard bridge row is part of the obligation surface.*

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket stability ledger exactness obligation`

Proof. Classifier stability is componentwise. Reflexivity, symmetry, and transitivity for derivative endpoints come from hsame, while the base, fibre, section, and tangent components are supplied by the two dependency certificates.

Carrier transport follows by applying those componentwise classifier rows to the tuple of Definition 134.1. Ledger readback is by projection from the tuple: each accepted connection row exposes exactly its base, fibre, section, tangent, derivative endpoint, dependency certificate, and provenance fields.

Exactness is the reverse reading. The section-linearity, parallel-transport, and curvature-boundary obligations use only the projected rows and finite Cont ledgers already listed above, so any accepted certificate step is one of the displayed connection rows and every displayed row is used by one of those certificate steps. □

134.2 The certificate

Theorem 134.7 (ConnectionUp public certificate consumer surface)**.** *A public consumer of* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *may read exactly the carrier packet of Definition 134.1, the* ConnCls *classifier of Definition 134.2, the section-linearity row, the parallel-transport row, the curvature-boundary row, the stability-ledger exactness row, and the displayed* $\mathsf{Bundle}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *dependencies carried by those rows. It does not receive a coordinate chart, a host connection operator, a curvature theory beyond the displayed boundary row, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket public certificate consumer surface`

Proof. Project the consumer request to the carrier and classifier rows. The carrier packet supplies exactly the $\mathsf{Bundle}^{\uparrow}$ and $\mathsf{Manifold}^{\uparrow}$ dependency rows, base, fibre, section, tangent, derivative endpoint, provenance, and finite readback data, while ConnCls supplies the public classifier.

Apply Definition 134.3, Definition 134.4, and Definition 134.5. These are precisely the derivative, transport, and curvature rows a consumer may follow from the carrier packet.

Finally apply Definition 134.6. Its exactness statement excludes any additional public ledger row, so the listed rows repack as the whole consumer surface for $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$. □

Theorem 134.8 (ConnectionUp standard bridge boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet determines a bridge-facing boundary consisting exactly of the section-linearity row, the parallel-transport row, the curvature-boundary row, and the stability-ledger exactness row displayed in this chapter. Any standard connection interpretation may read those rows together with their* BHist*,*

Cont, Pkg, *and* hsame *provenance, but it may not import a coordinate connection, smooth atlas, or host bundle operation.*

Lean4 ✓ `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket standard bridge boundary`

Proof. Start from the public consumer surface of Definition 134.7. It allows a consumer to project only the displayed carrier, classifier, dependency, and certificate rows of $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$.

Now select the four rows that face a standard interpretation. The section-linearity row is Definition 134.3, the parallel-transport row is Definition 134.4, the curvature-boundary row is Definition 134.5, and the stability row is Definition 134.6.

These four rows form a finite row family. Their proofs already project all permitted BHist endpoints, Cont ledgers, Pkg provenance, and hsame comparisons from the public packet, so a bridge can read exactly that provenance without adding another row sort.

The exclusion clause follows from the same consumer-surface and ledger exactness statements. Since no coordinate connection, smooth atlas, or host bundle operation is among the projected public rows, none is part of the bridge-facing boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$.

Constructive story: A connection packet is generated by taking the already certified Bundle$^\uparrow$ and Manifold$^\uparrow$ rows, then displaying a base point, a fibre over that point, a section endpoint in the fibre, a tangent-direction row, and a covariant-derivative endpoint tied back to them by finite Cont provenance. A packet is accepted only when all of those source rows are visible through the two dependency certificates; two packets classify together when their base, fibre, section, tangent, dependency-provenance, and derivative endpoint rows are compared componentwise by the inherited classifiers and hsame. The source covers exactly these displayed derivative packets, the pattern records section-linearity, finite parallel-transport chains, and the commutator boundary consumable by Curvature$^\uparrow$, while the ledger keeps the base, fibre, section, tangent, derivative, package provenance, and finite continuation rows aligned. The five certificate obligations follow componentwise: a displayed packet inhabits the carrier, reflexivity, symmetry, and transitivity are supplied by hsame, Bundle$^\uparrow$, and Manifold$^\uparrow$, and classifier transport moves each accepted field without producing an ambient connection operator. A history h lies in the signature gap of a Connection$^\uparrow$ token p when the source accepts such a packet for h, the signature relation pairs it with an answer derivative packet and evidence ledger, and token introduction identifies that answer as p; concretely, h presents the same base, fibre, section, tangent, derivative, and provenance rows that p encodes. The displayed obligations also say how section linearity, parallel transport, and curvature-boundary formation read from this finite ledger, but no coordinate semantics or checked standard bridge is exported. The chapter registers this displayed derivative-packet construction under the name Connection$^\uparrow$; no Leibniz theorem, host covariant derivative, or independent chart-level connection primitive is added here.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$, maturePackageClosed)

Scope closed: The mature scope is the displayed public Connection$^\uparrow$ packet over Bundle$^\uparrow$, Manifold$^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert, together with the four bridge-facing obligation rows and the downstream Curvature$^\uparrow$/ChernWeil$^\uparrow$ consumer boundary already cited.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.ConnectionUp.ConnectionCarrierPacket\_stability\_ledger\_exactness\_obligation`

Bridge status: none

Not claimed: This chapter does not close Leibniz, coordinate-chart semantics, or curvature compatibility.

Upgrade path: Only Lean-side bridgeChecked verification remains on the separate verification axis.

Chapter 135

A Concrete Naming Certificate for $\mathsf{Curvature}^{\uparrow}$

The curvature interface $\mathsf{Curvature}^{\uparrow}$ packages the antisymmetric bracket of a connection, certifying the Bianchi identities and the tensorial behaviour under section change. It depends on $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ (chapter 134) and supplies the local invariants on which Riemannian and gauge-field certificates rest.

135.1 Carrier definition

Definition 135.1 (Curvature bracket carrier)**.** *A* $\mathsf{Curvature}^{\uparrow}$ *carrier packet is a displayed* BHist *tuple*

$$\mathcal{R} = (\mathcal{C}, b, f, s, u, v, d_{uv}, d_{vu}, \kappa, \beta, \rho)$$

where $\mathcal{C}$ *is a source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet,* $b : \mathsf{BHist}$ *is the base row,* $f : \mathsf{BHist}$ *is the fibre row over* b*,* $s : \mathsf{BHist}$ *is the section row,* $u, v : \mathsf{BHist}$ *are the two tangent-direction rows,* $d_{uv}, d_{vu} : \mathsf{BHist}$ *are the two iterated derivative endpoint rows,* $\kappa : \mathsf{BHist}$ *is the displayed commutator boundary supplied by Definition 134.5,* β *is the finite* Cont *ledger reading that boundary from the two derivative orders, and* ρ *records the* Pkg *provenance of the connection packet.*

The carrier is accepted only when the base, fibre, section, tangent, derivative endpoint, commutator-boundary, ledger, and provenance rows all project from the same connection certificate. Thus curvature is a bracket package over $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$*, not an ambient tensor object added beside the BEDC history rows.*

Lean4 (definition) `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier`

Theorem 135.2 (Curvature boundary source obligation)**.** *A* $\mathsf{Curvature}^{\uparrow}$ *source row is read from a* $\mathsf{Connection}^{\uparrow}$ *carrier packet, two tangent-direction* BHist *rows, the two ordered derivative-composite* Cont *ledgers, and the commutator-boundary* hsame *row exposed by Definition 134.5. The resulting source packet carries no coordinate chart and no host curvature operator; it is exactly the displayed boundary row together with its connection, tangent, derivative, ledger, and* Pkg *provenance projections.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier boundary source obligation`

Proof. Project the source connection packet through Definition 134.5. This gives the base, fibre, section, tangent-pair, derivative-endpoint, commutator-boundary, and provenance rows already accepted by the connection certificate.

Read the two derivative orderings as the ordered Cont ledgers carried by the same connection packet. Their displayed endpoint comparison is the commutator-boundary hsame row.

Repack these projected rows as the $\mathsf{Curvature}^{\uparrow}$ source packet. Since every component is obtained by projection from the connection boundary obligation, the curvature carrier adds no independent coordinate datum and no further quotient of the history ledger. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBoundarySource connection projection`

Theorem 135.3 (Curvature connection classifier stability)**.** *Transport of the source connection packet by* ConnCls*, together with* $\mathsf{Manifold}^{\uparrow}$ *transport of the tangent pair, transports the* $\mathsf{Curvature}^{\uparrow}$ *boundary source row componentwise by* hsame*. In particular, classified replacements of the connection packet and of the two tangent-direction rows preserve the two derivative-composite ledgers and the commutator boundary endpoint row read by* $\mathsf{Curvature}^{\uparrow}$*.*

Proof. Start with the ConnCls comparison between the two connection packets. It transports the base, fibre, section, derivative endpoint, and provenance components by the classifier rows already exposed by $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$.

Use the $\mathsf{Manifold}^{\uparrow}$ classifier on the two tangent-direction rows. Componentwise transport sends the ordered tangent pair to the classified ordered tangent pair, so both derivative composites retain the same source and target projections after the connection transport.

Apply endpoint transport to the two Cont ledgers and then to the displayed commutator boundary. The result is an hsame comparison of the transported boundary rows, which is exactly the classifier stability required of the $\mathsf{Curvature}^{\uparrow}$ source packet. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier connection classifier stability`

Theorem 135.4 (Curvature carrier boundary row)**.** *Every accepted* $\mathsf{Curvature}^{\uparrow}$ *endpoint is exactly a displayed* BHist *commutator-boundary row inherited from Definition 134.5. Conversely, every curvature carrier packet of Definition 135.1 whose two derivative endpoints are accepted by the source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *determines an accepted curvature endpoint with the same base, fibre, section, tangent-direction,* Cont*, and* Pkg *provenance rows.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier carrier boundary row`

Proof. Project an accepted curvature endpoint to its carrier packet. By the carrier definition, the endpoint component is the row κ, and κ is declared only as the commutator boundary read from the two derivative orders of the source connection packet.

Apply Definition 134.5 to the displayed tangent rows u and v of that packet. The theorem supplies a BHist boundary whose source data are precisely the base, fibre, section, tangent-pair, derivative-endpoint, and Pkg rows already present in the curvature source row.

For the converse, start with such a carrier packet and the two accepted derivative endpoints. The connection boundary theorem produces κ; repack κ together with the unchanged Pkg and Cont ledger as the curvature endpoint. No coordinate curvature object is introduced, because the endpoint is only the displayed boundary row. □

Theorem 135.5 (Curvature classifier transport row)**.** *Let $\mathcal{R}$ and $\mathcal{R}'$ be curvature carrier packets over connection packets $\mathcal{C}$ and $\mathcal{C}'$. If the inherited* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *classifier transports the base, fibre, section, tangent-direction, and derivative-endpoint rows of $\mathcal{C}$ to those of $\mathcal{C}'$, if the two commutator boundaries are related by* hsame*, and if both rows carry the same* Pkg *provenance through a finite* Cont *reassociation ledger, then the* $\mathsf{Curvature}^{\uparrow}$ *classifier transports the curvature endpoint of $\mathcal{R}$ to the curvature endpoint of $\mathcal{R}'$.*

Proof. Unpack the two curvature carrier packets. The connection classifier supplies the comparisons for the base, fibre, section, tangent-direction, and derivative-endpoint components, while the common provenance row supplies the package comparison.

Reassociate the finite Cont ledger so that the two derivative orders are read with the same source grouping. This reassociation does not change any endpoint; it only exposes the two commutator-boundary rows as comparable BHist entries in the same provenance context.

Use the assumed hsame comparison between the two boundary rows. Repack the transported connection classifier rows, the boundary hsame witness, and the provenance ledger as the curvature classifier witness. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier classifier transport row`

Theorem 135.6 (Curvature Bianchi ledger row)**.** *For any cyclic triple of tangent-direction rows* u, v, w : BHist *over the same curvature carrier packet, the three commutator boundaries for* (u, v)*,* (v, w)*, and* (w, u) *carry a visible* SigRel *ledger witnessing the Bianchi boundary. This ledger is a finite* Cont *consequence of the three inherited connection boundary rows and their shared* Pkg *provenance, and it does not assert coordinate equality, exterior-calculus equality, or a host-level curvature identity.*

Proof. Read u, v, and w through the same source connection packet and the same finite Cont spine. For each cyclic ordered pair, apply Definition 134.5 to obtain the displayed commutator boundary row with the common base, fibre, section, and provenance projections.

Collect the three boundary rows into the signature ledger whose entries are the cyclic boundary readings. The SigRel witness records that the three readings belong to one finite certificate surface, while the Cont ledger records the append order of the boundary rows.

The Bianchi row is this ledger. It is visible because every entry is a carried BHist or SigRel row, and it is conservative because no equality outside those displayed rows is used. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier bianchi ledger row`

Theorem 135.7 (Curvature tensorial section-change row)**.** *Replacing the section row or fibre representative of a curvature carrier packet through the inherited* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *classifier transports the curvature endpoint through the same* BHist *boundary row and* Pkg *ledger. The derivative endpoint rows, the commutator boundary, and the cyclic* SigRel *ledger are carried along by the corresponding* hsame *and* Cont *witnesses, so the transported carrier packet presents the same* $\mathsf{Curvature}^{\uparrow}$ *classifier class.*

Proof. Begin with the source connection classifier witness for the section or fibre replacement. It identifies the displayed base, fibre, section, tangent, and derivative endpoint rows by the inherited component classifiers and exposes the needed hsame rows.

Transport the two derivative endpoint rows along those component comparisons. Applying the connection curvature-boundary theorem to the transported derivative orders gives the same boundary construction in the transported provenance context.

Finally carry the cyclic SigRel ledger through the same Cont and hsame witnesses. Repacking these rows yields the transported curvature carrier packet, and the endpoint remains in the same classifier class because the boundary and package ledgers have not changed their source certificate. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier tensorial section change row`

Theorem 135.8 (Curvature public obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Curvature}^\uparrow}$ *obligation surface over* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *consists exactly of the carrier packet of Definition 135.1, the boundary row of Definition 135.4, the classifier transport row of Definition 135.5, the Bianchi ledger row of Definition 135.6, and the tensorial section-change row of Definition 135.7. A downstream certificate such as* $\mathsf{ChernWeil}^\uparrow$ *may consume these rows as* BHist*,* Cont*,* hsame*,* Pkg*, and* SigRel *data, but it may not assume any additional coordinate curvature form or standard bridge from* $\mathsf{Curvature}^\uparrow$ *alone.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier public obligation surface`

Proof. Assemble the carrier by projecting the source connection packet, the tangent rows, the derivative endpoint rows, the commutator boundary, the package provenance, and the finite ledger from Definition 135.1.

The boundary theorem identifies the accepted endpoint; the classifier theorem transports it under inherited connection classification; the Bianchi theorem records the cyclic signature ledger; and the section-change theorem carries the endpoint through replacement of the section or fibre representative.

Conversely, every row exposed by the public surface appears in one of those five clauses. Thus the obligation surface is exact: it provides a concrete certificate surface for downstream geometry while leaving coordinate curvature forms, de Rham representatives, and standard bridges to their own certificate chapters. □

Theorem 135.9 (Curvature connection-boundary source projection)**.** *Every* BHist *curvature-boundary row accepted from the* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *source of Definition 134.5 projects to the visible rows required by the* $\mathsf{Curvature}^\uparrow$ *source surface: the base row supplied by* $\mathsf{Manifold}^\uparrow$*, the fibre row supplied by* $\mathsf{Bundle}^\uparrow$*, the section row, the ordered tangent-pair rows, the two derivative-endpoint rows, and the* Pkg *provenance rows of the* $\mathsf{Bundle}^\uparrow$ *and* $\mathsf{Manifold}^\uparrow$ *dependency certificates. The projection is a finite* Cont *readback from the accepted boundary row; it does not inspect host-level equality of base points, fibres, sections, tangent directions, or derivative endpoints.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureCarrierPacket connection boundary source projection`

Proof. Read the boundary row through the $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ packet. Its construction in Definition 134.5 already lists the base, fibre, section, tangent-pair, derivative-endpoint, and dependency rows used to build the commutator ledger.

Project those entries one at a time along the finite Cont readback. The base component is the $\mathsf{Manifold}^\uparrow$ point row, the fibre and section components are the $\mathsf{Bundle}^\uparrow$ rows, and the derivative endpoints are the two BHist endpoint rows compared by the boundary.

The Pkg entries are retained rather than recomputed. Thus the $\mathsf{Curvature}^\uparrow$ source receives exactly the connection boundary surface, with all comparison data still mediated by hsame and the dependency classifiers. □

Theorem 135.10 (Curvature antisymmetric boundary classifier)**.** *For a* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *boundary row whose ordered tangent pair is* (t_1, t_2)*, swapping the displayed tangent rows to* (t_2, t_1) *reverses the commutator boundary classifier by the inherited* hsame *endpoint comparisons and the* SigRel *orientation row. The reversed row has the same base, fibre, section, derivative-endpoint carrier rows, and* Pkg *provenance rows, but its classifier is the opposite boundary relation in* SigRel*, not a judgmental or host equality between the two commutator histories.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier antisymmetric boundary classifier`

Proof. Start with the two finite Cont composites that define the accepted boundary: first t_1 then t_2, and first t_2 then t_1. Swapping the tangent rows interchanges these two composites while leaving their base, fibre, section, and provenance rows unchanged.

Use symmetry of hsame on the derivative endpoint comparison to reverse the displayed endpoint relation. The reversal is recorded as a SigRel orientation row attached to the boundary classifier.

Repack the same source rows with the opposite SigRel classifier. Since no host equality of histories is asserted, the antisymmetry belongs to the BEDC classifier data exposed by $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ and consumed by $\mathsf{Curvature}^\uparrow$. □

Theorem 135.11 (Curvature tensorial section transport)**.** *Let a* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *curvature-boundary row be accepted over a section row and fibre row carried by the* $\mathsf{Bundle}^\uparrow$ *source. If the section classifier and fibre classifier are transported inside the same* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *dependency by finite* Cont *ledgers, then the transported section and fibre rows preserve the accepted* BHist *curvature boundary row. The transported boundary has the same* $\mathsf{Manifold}^\uparrow$ *base row, tangent-pair rows, derivative-endpoint comparison up to* hsame*, and* Pkg *provenance rows, so* $\mathsf{Curvature}^\uparrow$ *reads it as the same tensorial source boundary.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier tensorial section transport`

Proof. Apply the $\mathsf{Bundle}^\uparrow$ fibre and section classifiers supplied inside the $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ dependency. These classifiers transport the fibre and section rows by finite Cont ledgers while keeping the $\mathsf{Manifold}^\uparrow$ base row and tangent-pair rows fixed.

Transport the two derivative endpoint rows along those ledgers and compare the transported endpoints by hsame. The commutator boundary row is therefore carried to a boundary row with the same ordered tangent pair and the same source provenance.

The Pkg rows of the dependency certificates are copied through the transport. Hence the $\mathsf{Curvature}^\uparrow$ source surface sees tensorial section and fibre transport as classifier-preserving readback, not as a change of the underlying boundary object. □

135.2 The certificate

Theorem 135.12 (Curvature manifold-base dependency row)**.** *For every* $\mathsf{Curvature}^\uparrow$ *carrier packet* $\mathcal{R}$ *of Definition 135.1, the displayed base row* b *and tangent rows* u, v *are exactly the* $\mathsf{Manifold}^\uparrow$ BHist *dependency rows read from the source* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *packet. Any accepted transport of these rows is mediated by the inherited* hsame *endpoint comparisons and finite* Cont *ledgers, and no further base or tangent carrier is available to the curvature certificate.*

Proof. Unpack $\mathcal{R} = (\mathcal{C}, b, f, s, u, v, d_{uv}, d_{vu}, \kappa, \beta, \rho)$ as a curvature bracket carrier. The components b, u, v occur there only as source rows of the connection certificate $\mathcal{C}$.

Project the $\mathsf{Manifold}^\uparrow$ dependency rows through the source $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ packet. The same projection supplies the endpoint hsame comparisons and the finite Cont ledgers used when those rows are transported.

Conversely, every base or tangent row exposed by the curvature surface must be one of b, u, v in the carrier tuple. Repacking the projected rows therefore gives the exact $\mathsf{Manifold}^\uparrow$ consumption row for $\mathsf{Curvature}^\uparrow$. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier manifold base dependency row`

Theorem 135.13 (Curvature bundle-fibre dependency row)**.** *For every* $\mathsf{Curvature}^{\uparrow}$ *carrier packet over a fixed base packet, the fibre row* f *and section row* s *are exactly the* $\mathsf{Bundle}^{\uparrow}$ *dependency rows supplied by the same source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet. Their accepted transports are the inherited bundle classifier transports over the already fixed* $\mathsf{Manifold}^{\uparrow}$ *base row, recorded as* hsame *comparisons and finite* Cont *ledger entries inside the curvature packet.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier bundle fibre dependency row`

Proof. Keep the base row b fixed through the previous dependency row. In the carrier tuple of Definition 135.1, the rows f and s are read from the same connection certificate $\mathcal{C}$ over that base.

Project the $\mathsf{Bundle}^{\uparrow}$ fibre and section dependencies from $\mathcal{C}$. The connection certificate carries their classifier transport ledgers, so curvature receives the transported fibre and section rows by readback rather than by constructing a new bundle object.

For the reverse containment, inspect the curvature packet. The only fibre and section entries visible to downstream certificates are f and s, together with their inherited hsame and Cont data, so the bundle dependency surface is exact. □

Theorem 135.14 (Curvature connection-derivative dependency row)**.** *For every* $\mathsf{Curvature}^{\uparrow}$ *carrier packet, the derivative endpoint rows* d_{uv} *and* d_{vu} *and the commutator boundary row* κ *are projected from* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *by Definition 134.5. The curvature packet may repackage these rows with its boundary ledger* β*, but it does not add an independent derivative operator, endpoint equality, or commutator carrier outside that connection boundary obligation.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier connection derivative dependency row`

Proof. Apply Definition 134.5 to the source connection packet $\mathcal{C}$ and the ordered tangent rows u, v. It supplies the two ordered derivative endpoint rows and the displayed commutator boundary in BHist form.

Read d_{uv}, d_{vu}, and κ from that boundary surface through the finite ledger β. The ledger records the two derivative orders and the endpoint comparison used by the curvature packet.

Repack these three rows as curvature data. Since each row is obtained by projection from the connection boundary theorem, any downstream use of the derivative surface factors through $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$. □

Theorem 135.15 (Curvature ChernWeil export dependency row)**.** *The complete* $\mathsf{ChernWeil}^{\uparrow}$ *export surface of a* $\mathsf{Curvature}^{\uparrow}$ *packet consists of the* BHist *curvature boundary row* κ*, the cyclic ledger recorded by the Bianchi* SigRel *rows, the classifier rows exposed by the curvature transport obligations, and the* Pkg *provenance* ρ *already listed in Definition 135.1. No additional coordinate form, host tensor equality, or exterior-calculus representative is exported from* $\mathsf{Curvature}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier chernweil export dependency surface`

Proof. Project the boundary row κ and package provenance ρ from the curvature carrier. The same carrier exposes the finite Cont ledger β that downstream certificates use to read the boundary from the ordered derivative endpoints.

Add the cyclic Bianchi ledger and classifier rows from the existing curvature obligation rows. These rows are already presented as SigRel, hsame, Cont, and Pkg data, so they can be consumed by $\mathsf{ChernWeil}^{\uparrow}$ without changing the carrier.

Conversely, inspect the public curvature obligation surface. Every row named for downstream consumption is one of the boundary, cyclic ledger, classifier, or provenance rows just listed, and no theorem in the packet grants a coordinate curvature form or host tensor equality. □

Theorem 135.16 (Curvature source row coverage)**.** *Every accepted* $\mathsf{Curvature}^{\uparrow}$ *carrier packet projects exactly the source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet, the* $\mathsf{Manifold}^{\uparrow}$ *base row, the* $\mathsf{Bundle}^{\uparrow}$ *fibre and section rows, the ordered tangent-pair rows, the two derivative endpoint rows, the commutator-boundary row, the finite* Cont *ledger reading that boundary, and the* Pkg *provenance row. Conversely, any such rows over one accepted source connection packet repack as the source surface of the carrier in Definition 135.1.*

Proof. Project an accepted carrier through the tuple of Definition 135.1. Its fields name the source connection packet, the base, fibre, section, tangent-pair, derivative endpoint, boundary, ledger, and provenance rows without asking for a coordinate chart or host tensor.

Apply Definition 134.5 to the projected tangent pair. This identifies the commutator boundary as the displayed BHist row read from the two derivative orders, with the same source connection and Pkg provenance.

For the converse, start with exactly those rows over the same source packet. The carrier definition gives the constructor pattern, and the boundary obligation supplies the compatibility condition for the boundary and ledger entries, so the listed rows repack as the $\mathsf{Curvature}^{\uparrow}$ source surface and nothing else is needed. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier source row coverage`

Theorem 135.17 (Curvature boundary row determinacy)**.** *Fix a source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet, an ordered tangent pair, the two derivative* Cont *ledgers, and the* Pkg *provenance row. Two accepted* $\mathsf{Curvature}^{\uparrow}$ *boundary rows classify together precisely by the carried* hsame *boundary comparison and its* SigRel *orientation row. This determinacy uses no host curvature tensor, no coordinate representative, and no quotient of the* BHist *ledger.*

Proof. With the source packet, tangent pair, derivative ledgers, and provenance fixed, Definition 134.5 leaves only the displayed commutator boundary row to compare. The carrier of Definition 135.1 records that row as κ and records its finite readback as the Cont ledger.

If the two accepted rows carry an hsame boundary comparison with the same SigRel orientation, the curvature classifier repacks that comparison over the fixed source fields. The fixed rows guarantee that no further geometric datum can distinguish the two boundary packets.

Conversely, any classification of the two accepted boundary rows must project to the boundary component of the carrier. Since all other source fields are fixed, that projection is exactly the carried hsame comparison together with the SigRel boundary row, not an equality of host curvature tensors and not a quotient of histories. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier boundary row determinacy`
↪ variant `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier boundary determinacy`

Theorem 135.18 (Curvature ChernWeil visible input exhaustion)**.** *The only* $\mathsf{Curvature}^{\uparrow}$ *rows exported to* $\mathsf{ChernWeil}^{\uparrow}$ *are the curvature boundary row, the cyclic Bianchi ledger, the tensorial transport row, the source connection provenance, and the finite* Cont *and* Pkg *ledgers. De Rham representatives, characteristic-class rows, invariant-polynomial exactness rows, and standard bridge data remain obligations of* $\mathsf{ChernWeil}^{\uparrow}$ *or* $\mathsf{DeRham}^{\uparrow}$*, not exports of* $\mathsf{Curvature}^{\uparrow}$*.*

Proof. Project the public curvature carrier by Definition 135.1 and the scope theorem Definition 135.45. The visible source entries are the boundary row, the connection provenance, and the finite ledger rows that carry the bracket packet.

Next apply Definition 135.42 and Definition 135.43. These add only the cyclic SigRel ledger and the tensorial transport row, both still carried as BHist, Cont, hsame, and Pkg data.

The exhaustion clause is the reverse projection. Every row named by the export appears in one of the listed carrier or obligation projections, while de Rham representatives and characteristic-class rows do not appear in those projections. They therefore belong to the downstream certificates that consume the curvature packet. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope visible input exhaustion`

Definition 135.19 (Curvature Chern-Weil source envelope)**.** *The* $\mathsf{Curvature}^{\uparrow}$ *export to* $\mathsf{ChernWeil}^{\uparrow}$ *is the finite source envelope*

$$\mathsf{CurvatureChernWeilSourceEnvelope}(\kappa, \delta, \rho, \lambda, \chi)$$

where $\kappa : \mathsf{BHist}$ *is the curvature commutator-boundary row,* $\delta : \mathsf{BHist}$ *is the displayed de Rham target row supplied by the consumer,* $\rho : \mathsf{BHist}$ *is the shared* Pkg *provenance row,* $\lambda : \mathsf{BHist}$ *is the connection ledger row, and* $\chi : \mathsf{BHist}$ *is the classifier row read by the Chern-Weil packet.*

The envelope is admitted only when $\mathsf{UnaryHistory}(\kappa)$*,* $\mathsf{UnaryHistory}(\delta)$*,* $\mathsf{UnaryHistory}(\rho)$*,* $\mathsf{UnaryHistory}(\lambda)$*, and* $\mathsf{UnaryHistory}(\chi)$ *hold, when finite* Cont *ledgers read* ρ *from* κ *and* δ *and read* χ *from* ρ *and* λ*, and when the carried* hsame *readback identifies* χ *with the finite appended history obtained by first pairing the curvature boundary with the de Rham row and then appending the connection ledger. The* Pkg *provenance binds all five rows to the same accepted curvature source and consumer-facing de Rham display.*

Lean4 (definition) `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope`

Theorem 135.20 (Curvature Chern-Weil source envelope coverage)**.** *Every accepted* $\mathsf{Curvature}^{\uparrow}$ *carrier of Definition 135.1, together with a displayed de Rham consumer row* $\delta : \mathsf{BHist}$ *whose provenance is tied to the same finite* Pkg *ledger, repacks as a* $\mathsf{CurvatureChernWeilSourceEnvelope}(\kappa, \delta, \rho, \lambda, \chi)$*. The row* κ *is the exported commutator boundary,* ρ *is the shared provenance row,* λ *is the chosen connection ledger row, and* χ *is the classifier row obtained by the finite* Cont *readback from those rows.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope coverage`

Proof. Project the accepted curvature carrier through Definition 135.1. This gives the boundary row κ, the source connection packet, the derivative ledgers, and the Pkg provenance row already tied to the same finite history surface.

Apply Definition 135.24. Its forward direction lists every exported curvature witness and its reverse direction repacks exactly those rows as the visible curvature ledger.

Add the displayed consumer row δ with the assumed provenance tie. The finite Cont rows first bind κ and δ into ρ and then bind ρ and the connection ledger λ into χ, while the carried hsame readback supplies the displayed appended history required by Definition 135.19. □

Theorem 135.21 (Curvature Chern-Weil source envelope readback)**.** *An admitted* CurvatureChernWeilSourceEnvelo *projects exactly the five rows consumed by* $\mathsf{ChernWeil}^{\uparrow}$*: the curvature boundary row* κ*, the de Rham target row* δ*, the shared provenance row* ρ*, the connection ledger row* λ*, and the classifier row* χ*. These projections are the same row positions as the existing Chern-Weil source envelope consumes, with* κ *occupying the curvature input and* δ *occupying the de Rham input.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope derham transport readback`
Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope readback`

Proof. Unpack the envelope definition. The tuple has five displayed BHist coordinates, and each coordinate is constrained by UnaryHistory inside the same finite source packet.

Read the two finite Cont rows. The first identifies ρ as the provenance row produced from the curvature and de Rham coordinates; the second identifies χ as the classifier endpoint produced from that provenance row and the connection ledger.

Finally use the hsame readback clause. It states that the classifier row is precisely the appended history over κ, δ, and λ, so projection returns no hidden coordinate besides the five displayed rows. □

Theorem 135.22 (Curvature Chern-Weil source envelope exactness)**.** *The envelope of Definition 135.19 exports no coordinate curvature form, no de Rham cohomology class, no characteristic-class equality, and no standard bridge. Its public content is exactly the curvature boundary row, the displayed de Rham target row, the shared* Pkg *provenance row, the connection ledger row, the classifier row, and the finite* Cont *and* hsame *witnesses tying those rows together.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope exactness`
Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope semantic name certificate`

Proof. Use Definition 135.25 to identify the curvature side exported to the invariant-polynomial consumer. That theorem supplies the source connection certificate, boundary row, transported endpoint ledger, visible classifier rows, and provenance row, while excluding de Rham classes, characteristic-class equalities, and standard bridges from $\mathsf{Curvature}^{\uparrow}$ itself.

The source envelope adds only the displayed row δ and the two finite readback ledgers required to match the Chern-Weil input shape. By Definition 135.24, every curvature contribution is already one of the listed BHist, Cont, hsame, or Pkg rows.

Therefore any attempted coordinate curvature form, cohomology class, characteristic-class equality, or standard bridge would have to be a sixth public coordinate. No such coordinate appears in Definition 135.19, so the export is exact. □

Theorem 135.23 (Curvature tensorial endpoint transport)**.** *Bundle, manifold, and section classifier transport of the section, fibre, base, and tangent rows preserves the curvature endpoint row. After transporting the two derivative orderings by the source* $\mathsf{Connection}^{\uparrow}$ *section-linearity and stability-ledger exactness obligations, the transported commutator boundary is* hsame *to the boundary row of the transported* $\mathsf{Curvature}^{\uparrow}$ *packet.*

Proof. Transport the base and tangent rows by the $\mathsf{Manifold}^{\uparrow}$ classifiers and the fibre and section rows by the inherited bundle and section classifiers. The source connection certificate keeps these transports tied to the same carrier packet and provenance row.

Apply the section-linearity and stability-ledger exactness obligations of $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ to the first derivative ordering. Repeat the same construction for the reversed derivative ordering, so the two transported endpoints remain the endpoints of certified Cont ledgers.

Compare the two transported endpoint ledgers through the commutator-boundary row. The boundary comparison is componentwise hsame, hence the curvature endpoint row is preserved by the transported $\mathsf{Curvature}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier tensorial endpoint transport`

Theorem 135.24 (Curvature visible ledger coverage)**.** *Every exported* $\mathsf{Curvature}^{\uparrow}$ *witness is one of the source connection certificate rows, the base, fibre, section, tangent-pair, two derivative-composite ledger rows, the commutator boundary row, the classifier transport rows, or the* Pkg *provenance rows. Conversely, exactly these rows repack as the visible* $\mathsf{Curvature}^{\uparrow}$ *ledger.*

Proof. For the forward reading, project an accepted curvature witness through the carrier tuple of Definition 135.1. The projections are precisely the source connection certificate, geometric rows, tangent pair, derivative composites, boundary row, classifier transports, and provenance rows.

For the reverse reading, begin with those listed rows over a common source connection packet. The boundary source obligation repacks the connection, tangent, derivative, boundary, and provenance rows, while the classifier stability and endpoint transport obligations repack the transport rows.

Thus the public ledger is exact. Each exported witness is a listed row, and each listed row has a constructor role in the visible $\mathsf{Curvature}^{\uparrow}$ certificate surface. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier visible ledger coverage`

Theorem 135.25 (Curvature ChernWeil consumer boundary)**.** *Any accepted* $\mathsf{Curvature}^{\uparrow}$ *row exports exactly the invariant-polynomial input packet consumed by* $\mathsf{ChernWeil}^{\uparrow}$*: the source connection certificate, the curvature boundary row, the transported endpoint ledger, the visible Bianchi and classifier rows when present, and the* Pkg *provenance row. This export does not assert a de Rham class, a characteristic-class equality, a standard bridge, or a Lean verification target for* $\mathsf{ChernWeil}^{\uparrow}$*.*

Proof. Project the accepted curvature row to the fields listed in Definition 135.1 and to the public certificate rows in this section. These projections provide the boundary row and transport ledgers that an invariant-polynomial consumer can read.

Package those projections as the input side of $\mathsf{NameCert}_{\mathsf{ChernWeil}^{\uparrow}}$. The packet contains only the curvature boundary, the connection provenance, and the visible ledger rows needed to form polynomial expressions over the accepted curvature data.

The exclusions follow from ledger coverage. Since no de Rham row, characteristic-class comparison, standard bridge, or Lean target is part of the exported $\mathsf{Curvature}^{\uparrow}$ ledger, none is available to $\mathsf{ChernWeil}^{\uparrow}$ without an independent certificate. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope consumer boundary`

Definition 135.26 (Curvature-to-ChernWeil input envelope)**.** *The* $\mathsf{Curvature}^{\uparrow}$ *export to* $\mathsf{ChernWeil}^{\uparrow}$ *is admitted as a five-row source envelope*

$$(\mathsf{curvature}, \mathsf{derham}, \mathsf{provenance}, \mathsf{connectionLedger}, \mathsf{classRow})$$

when $\mathsf{curvature}$ *is the carried curvature boundary row of Definition 135.25,* derham *is the de Rham target row consumed by* $\mathsf{ChernWeil}^{\uparrow}$*,* $\mathsf{provenance}$ *is obtained by* $\mathsf{Cont}(\mathsf{curvature}, \mathsf{derham}, \mathsf{provenance})$*, and* $\mathsf{classRow}$ *is obtained by* $\mathsf{Cont}(\mathsf{provenance}, \mathsf{connectionLedger}, \mathsf{classRow})$*. The row* $\mathsf{connectionLedger}$ *records the chosen-connection face already carried by the curvature provenance packet, so the envelope contains only displayed* BHist *and* Cont *data.*

Lean4 (definition) `BEDC.Derived.CurvatureUp.CurvatureChernWeilConsumerRow`

Theorem 135.27 (Curvature ChernWeil source-envelope projection)**.** *Every envelope of Definition 135.26 projects to unary rows for* $\mathsf{curvature}$*,* derham*,* $\mathsf{provenance}$*,* $\mathsf{connectionLedger}$*, and* $\mathsf{classRow}$*; to the continuation rows* $\mathsf{Cont}(\mathsf{curvature}, \mathsf{derham}, \mathsf{provenance})$ *and* $\mathsf{Cont}(\mathsf{provenance}, \mathsf{connectionLedger}, \mathsf{class}$ *and to the readback row*

$$\mathsf{hsame}(\mathsf{classRow}, \mathsf{append}(\mathsf{append}(\mathsf{curvature}, \mathsf{derham}), \mathsf{connectionLedger})).$$

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope source envelope projection`

Proof. Unpack the five-row envelope. Its admission clauses give the unary curvature row, unary de Rham row, unary connection-ledger row, and the two displayed Cont rows.

Apply closure of unary history under the first Cont row to obtain the provenance unary row. Apply the same closure principle to the second Cont row to obtain the class-row unary projection.

Finally compose the readbacks carried by the two Cont rows. The first readback identifies provenance with append(curvature, derham); the second appends connectionLedger and identifies the endpoint with classRow. This is exactly the displayed hsame row. □

Theorem 135.28 (Curvature ChernWeil consumer exactness)**.** *The* $\mathsf{Curvature}^{\uparrow}$ *export to* $\mathsf{ChernWeil}^{\uparrow}$ *consists exactly of the source envelope rows of Definition 135.26 together with the existing connection provenance exposed by Definition 135.24. It exports no de Rham class, characteristic-class equality, or standard bridge beyond those rows.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope consumer exactness`

Proof. First project the accepted curvature packet by Definition 135.24 and Definition 135.25. These give the curvature boundary row, transported endpoint ledger, visible classifier and Bianchi rows, and the connection provenance available to a $\mathsf{ChernWeil}^{\uparrow}$ consumer.

Next form the five-row envelope of Definition 135.26. By Definition 135.27, every public row of that envelope is one of the unary row projections, one of the two continuation rows, or the displayed hsame readback of the class row.

Combining the two projections gives exactness. The only rows exported from $\mathsf{Curvature}^{\uparrow}$ are the existing connection provenance and the source-envelope rows just listed; a de Rham class, characteristic-class equality, or standard bridge would require an additional certificate not present in the curvature ledger. □

Definition 135.29 (Curvature Chern-Weil consumer row)**.** *A curvature Chern-Weil consumer row is a concrete* BHist *packet*

$$\mathsf{CurvatureChernWeilConsumerRow}(\kappa, d, p, \gamma, \lambda)$$

whose coordinates are the exported curvature boundary row κ, the displayed de Rham row d, the shared provenance row p, the connection ledger row γ, and the characteristic-class endpoint row λ. The packet is accepted exactly when these five rows form the source envelope consumed by Definition 272.1: κ, d, and γ are unary source rows, p is the shared Cont *provenance over the curvature and de Rham inputs, and λ is the final* Cont *endpoint with its carried* hsame *readback against the appended curvature, de Rham, and connection ledger.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureTasteGate single carrier alignment`

Theorem 135.30 (Curvature Chern-Weil consumer stability)**.** *Let* $\mathsf{CurvatureChernWeilConsumerRow}(\kappa, d, p, \gamma, \lambda)$ *be accepted. If an exported curvature boundary κ' is related to κ by the* hsame *transport supplied by* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$*, and the invariant-polynomial row, provenance row p, de Rham row d, and connection ledger row γ are kept in the same carrier classifier, then the Chern-Weil endpoint transported through that classifier is again the endpoint row λ of an accepted consumer packet.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope consumer stability`

Proof. Project the accepted consumer packet to its curvature source. By Definition 135.45, the boundary row, classifier rows, finite ledgers, and provenance row are all public rows of the same $\mathsf{Curvature}^{\uparrow}$ certificate.

Transport the boundary along the displayed hsame comparison while keeping the invariant-polynomial, de Rham, provenance, and connection ledger rows fixed in the carrier classifier. The projected rows therefore still occupy the same source-envelope positions.

Apply Definition 272.10. Its classifier stability clause transports the invariant-polynomial endpoint across the curvature boundary comparison, so the final class row is read as the same λ through the carrier classifier. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope curvature transport`

Theorem 135.31 (Curvature Chern-Weil consumer envelope projection)**.** *Every accepted packet of Definition 135.29 projects to the exact rows of the Chern-Weil source envelope: the unary curvature row κ, the unary de Rham row d, the unary connection ledger row γ, the shared* Cont *provenance row p, the final* Cont *class row λ, and the readback* hsame *of λ against the appended curvature, de Rham, and connection ledger.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope consumer envelope projection`

Proof. Unpack the consumer packet. Its five coordinates are concrete BHist rows, and acceptance says that they occupy the curvature, de Rham, provenance, connection ledger, and endpoint positions of the source envelope of Definition 272.1.

Project the unary row obligations for κ, d, and γ, then project the two finite Cont ledgers. The first ledger supplies the shared provenance row p from the curvature and de Rham inputs, and the second supplies the endpoint row λ from that provenance row and the connection ledger.

Apply Definition 272.14 to identify the consumer-facing curvature and de Rham rows as the Chern-Weil input surface. The checked source-envelope projection cited in the Chern-Weil chapter then reads back the same rows and the displayed hsame comparison of λ with the appended curvature, de Rham, and connection ledger. □

Theorem 135.32 (Curvature Chern-Weil source-envelope row)**.** *Let a* $\mathsf{Curvature}^{\uparrow}$ *row be accepted from the* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet in Definition 135.1. Its export to the* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope supplies exactly the five source-envelope rows consumed by the Chern-Weil projection theorem: the curvature boundary row, the de Rham target row slot, the shared* Cont *provenance row, the selected-connection ledger face, and the characteristic-class endpoint row slot. The visible curvature-side fields are projected only as* BHist *rows,* hsame *endpoint comparisons, and finite* Cont *ledgers, with the* Pkg *provenance retained from the source connection packet.*

Proof. Project the accepted row through Definition 135.1. This gives the source $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ packet, tangent-pair rows, derivative endpoints, the commutator boundary, the finite boundary ledger, and the Pkg provenance.

Apply Definition 134.5 to identify the exported curvature boundary as a BHist row. Apply Definition 135.43 and Definition 135.44 to keep the endpoint comparisons and finite readback inside the same curvature packet.

The remaining Chern-Weil rows are interface slots rather than curvature-side conclusions. The de Rham target row slot and characteristic-class endpoint row slot are left for $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ and $\mathsf{ChernWeil}^{\uparrow}$, while the selected-connection ledger face and shared provenance are copied from the projected source connection packet. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope source envelope row`

Theorem 135.33 (Curvature Chern-Weil provenance ledger)**.** *The provenance exported by* $\mathsf{Curvature}^{\uparrow}$ *for a* $\mathsf{ChernWeil}^{\uparrow}$ *consumer is exactly the source connection packet, the curvature boundary row, the Bianchi ledger, the tensorial transport row, and the* Pkg *provenance certified by Definition 135.45. No additional coordinate connection form, analytic curvature form, or uncarried equivalence relation is part of this ledger.*

Proof. Start with the carrier tuple in Definition 135.1. Its first projection is the source connection packet, and Definition 134.5 supplies the boundary row read from the two derivative orders.

Project the cyclic row by Definition 135.42. Project the section and tangent transport row by Definition 135.43. These two ledgers are finite Cont consequences of the same boundary data and the same source packet.

Finally apply Definition 135.45. That theorem collects the carrier, boundary, Bianchi, tensorial transport, exactness, and provenance rows as the public $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ surface, so the listed rows are precisely the provenance visible to the Chern-Weil consumer. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope provenance ledger transport`

Theorem 135.34 (Curvature Chern-Weil visible class boundary)**.** *The* $\mathsf{Curvature}^{\uparrow}$ *export licenses only the visible input side of the* $\mathsf{ChernWeil}^{\uparrow}$ *characteristic-class boundary. It supplies the accepted curvature boundary, its* hsame *transport, its finite* Cont *ledgers, and its* Pkg *provenance, but it does not license a de Rham class, an analytic characteristic class, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope visible class boundary`

Proof. Read the visible input from Definition 135.1. The boundary, derivative endpoints, transport comparisons, and provenance are all displayed as BEDC rows, and Definition 135.44 states that every public curvature row is generated from this listed surface.

Apply Definition 135.45 to repack those rows as the $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ export. This gives exactly the input face that $\mathsf{ChernWeil}^{\uparrow}$ may project before introducing any de Rham dependency.

The negative clauses follow from the same projection. No de Rham carrier, analytic characteristic-class equality, or standard bridge occurs among the projected rows, so those objects require their own certificate and cannot be read from $\mathsf{Curvature}^{\uparrow}$ alone. □

Definition 135.35 (Curvature Chern-Weil handoff packet)**.** *A Curvature-to-Chern-Weil handoff packet is a displayed* BHist *tuple*

$$\mathcal{H} = (\mathcal{R}, \mathcal{W}, \kappa, \beta, \tau, \lambda, \rho)$$

where $\mathcal{R}$ *is an accepted* $\mathsf{Curvature}^{\uparrow}$ *carrier row,* $\mathcal{W}$ *is the* $\mathsf{ChernWeil}^{\uparrow}$ *source-envelope row fed by* $\mathcal{R}$*,* $\kappa : \mathsf{BHist}$ *is the curvature boundary row,* β *is the cyclic Bianchi ledger,* τ *is the tensorial transport row for section and tangent replacement,* λ *is the finite* Cont *projection from the curvature rows into the Chern-Weil envelope, and* ρ *is the shared* Pkg *provenance row. The packet is accepted exactly when these fields project from the carrier of Definition 135.1, the public export of Definition 135.45, and the source-envelope interface of Definition 272.1.*

Theorem 135.36 (Curvature handoff Bianchi transport)**.** *For any handoff packet of Definition 135.35, classifier-preserving replacement of the* $\mathsf{Curvature}^{\uparrow}$ *section row and tangent-direction rows transports both the Bianchi ledger* β *and the closed-endpoint input consumed by* $\mathsf{ChernWeil}^{\uparrow}$*. The transported handoff has the same* Pkg *provenance and reaches the Chern-Weil source envelope by the transported* hsame *comparisons and the same finite* Cont *projection.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilHandoffPacket bianchi transport`

Proof. Project the handoff packet to its curvature carrier row. The Bianchi component is the cyclic ledger supplied by Definition 135.42, and the section and tangent replacements are governed by the tensorial row of Definition 135.43.

Transport the curvature boundary, Bianchi ledger, and endpoint rows through the inherited hsame comparisons. The tensorial transport keeps the base, fibre, connection source, and Pkg provenance fixed, so the transported packet still has the same dependency surface.

Apply the Chern-Weil closed-form endpoint obligation to the transported curvature input. Since the projection λ is a finite Cont row from the handoff packet into the Chern-Weil source envelope, the closed-endpoint input is transported with the envelope rather than recomputed as an external form. □

Theorem 135.37 (Curvature handoff provenance exactness)**.** *Every field of a Curvature-to-Chern-Weil handoff packet is either a public* Curvature$^{\uparrow}$ *row or a* ChernWeil$^{\uparrow}$ *envelope row reached by the displayed* Cont *projection. The handoff exports no de Rham class, no characteristic-class equality, no coordinate curvature form, and no standard bridge.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilHandoffPacket provenance exactness`

Proof. Project the handoff fields from Definition 135.35. The curvature carrier, boundary row, Bianchi ledger, tensorial transport row, and provenance row are exactly the public rows covered by Definition 135.45.

Use the ledger exactness obligation of Definition 135.44 to rule out unlisted curvature data. The only consumer-facing row beyond those projections is the finite Cont projection into the Chern-Weil source envelope.

The Chern-Weil side is therefore an envelope readback, not an added bridge theorem. De Rham classes, characteristic-class equalities, coordinate curvature forms, and standard bridges are absent because no field of the handoff packet has such a source or constructor role. □

Theorem 135.38 (Curvature handoff consumer coverage)**.** *Every* ChernWeil$^{\uparrow}$ *source envelope whose curvature component is supplied by an accepted* Curvature$^{\uparrow}$ *row admits a handoff packet of Definition 135.35. Conversely, each accepted handoff exports exactly the invariant-polynomial input rows that* ChernWeil$^{\uparrow}$ *may consume from* Curvature$^{\uparrow}$*: the boundary row, Bianchi ledger, tensorial transport row, finite projection, and shared* Pkg *provenance.*

Proof. For coverage, begin with a Chern-Weil source envelope whose curvature component is supplied by an accepted curvature row. Apply Definition 135.25 to project the boundary row, transported endpoint ledger, visible Bianchi and classifier rows, and provenance row.

Repack those rows with the source envelope as the tuple of Definition 135.35. The source-envelope definition supplies the finite Cont projection into ChernWeil$^{\uparrow}$, while the curvature carrier supplies the accepted boundary, Bianchi, transport, and provenance fields.

For the converse, project an accepted handoff packet. Its consumer-facing fields are precisely the curvature rows listed in the tuple and the finite projection into the envelope; by provenance exactness, there is no additional de Rham endpoint, characteristic-class equality, coordinate form, or bridge row available for consumption. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilHandoffPacket consumer coverage`

Theorem 135.39 (Curvature connection-carrier scope)**.** *Every* $\mathsf{Curvature}^{\uparrow}$ *carrier packet of Definition 135.1 has a visible* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet as its dependency source. The base row, fibre row, section row, tangent-pair rows, iterated derivative rows, commutator boundary row, ledger row, and provenance row are obtained by projection from that packet together with the boundary row of Definition 134.5.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier connection carrier scope`

Proof. Project the connection certificate $\mathcal{C}$ from the displayed tuple. Its base, fibre, section, tangent, derivative, dependency, and provenance rows are already named by the connection certificate surface.

Apply Definition 134.5 to the two tangent-direction rows. This gives the commutator boundary as a BHist row whose source data is the same base, fibre, section, tangent-pair, derivative-endpoint, and Pkg provenance data.

Read the ledger β from that boundary row and repack the projected rows into the curvature tuple. Since every row used in the repacking is either a projection from $\mathcal{C}$ or the displayed commutator boundary, the carrier scope contains no additional geometric object. □

Theorem 135.40 (Curvature bracket classifier obligation)**.** *For packets of Definition 135.1, the curvature classifier is stable under* hsame *on the commutator-boundary row and under the connection classifier transport inherited from* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$*. If two source connection packets classify the same base, fibre, section, tangent-direction, and derivative-endpoint rows, then their curvature packets classify the same bracket precisely when the transported boundary rows are related by* hsame *inside the carried* Cont *ledger.*

Proof. Project the two curvature packets to their source connection packets. The inherited connection classifier supplies the base, fibre, section, tangent-direction, derivative-endpoint, and provenance comparisons.

Transport the commutator boundary along those projected comparisons. Since the boundary row is the row provided by Definition 134.5, the two derivative orders remain attached to the same connection package after transport.

Repack the transported rows as a curvature classifier witness. The only additional comparison is hsame on the displayed boundary row, so classifier stability is obtained without introducing a host-level curvature tensor or quotienting the history ledger. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier classifier obligation`

Theorem 135.41 (Curvature antisymmetric bracket obligation)**.** *For a curvature packet* $\mathcal{R}$*, exchanging the tangent-direction rows* u *and* v *exchanges the iterated derivative endpoint rows* d_{uv} *and* d_{vu} *and reverses the displayed commutator boundary. The two curvature packets are classified through the same* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet and the* hsame *transport of the reversed boundary row.*

Proof. Project the shared connection certificate and the two tangent rows from $\mathcal{R}$. The connection classifier keeps the base, fibre, section, and provenance rows fixed, so the exchanged tuple has the same visible dependency source.

Apply Definition 134.5 first to (u, v) and then to (v, u). The two constructions expose the same iterated derivative rows in opposite order and therefore display opposite commutator boundaries.

Transport the reversed boundary endpoint along hsame and retain the inherited classifier witness for the connection component. Repacking these transported rows gives the required curvature-packet comparison. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier antisymmetric bracket obligation`

Theorem 135.42 (Curvature Bianchi ledger obligation)**.** *Given three tangent-direction rows u, v, w :* BHist *over the same base row of a curvature packet, the first Bianchi row is the cyclic* BHist *ledger comparing the three displayed commutator boundaries for (u, v, w), (v, w, u), and (w, u, v). The certificate obligation is exactly that this cyclic row is a finite* Cont *consequence of the three connection-boundary rows and their shared* Pkg *provenance.*

Proof. Read the three tangent rows through the same source $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ packet. For each ordered pair, apply Definition 134.5 to obtain the displayed commutator boundary row with the same base, fibre, section, and provenance projections.

Form the cyclic ledger by appending the three boundary rows in the displayed order (u, v, w), (v, w, u), and (w, u, v). Each append is a Cont step between rows already carried by the curvature packet, and each endpoint comparison is an hsame comparison of boundary rows.

The Bianchi witness is this finite cyclic ledger. It asserts no coordinate identity beyond the listed rows; it records that the cyclic commutator comparison is present as a BEDC history package generated from the connection boundary surface. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier bianchi ledger obligation`

Theorem 135.43 (Curvature tensorial transport obligation)**.** *The curvature bracket packet is tensorial with respect to certified replacement of its section row and tangent-direction rows. Replacing s, u, or v by rows classified with them by the source connection packet transports the derivative endpoints, the commutator boundary κ, and the Bianchi ledger by* Cont *and* hsame*, while preserving the same base, fibre, and* Pkg *provenance rows.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier tensorial transport obligation`

Proof. Begin with the source connection classifier witnesses for the section row and the two tangent-direction rows. They keep the base and fibre rows fixed up to the inherited dependency classifiers and expose the derivative endpoint transport rows.

Use those endpoint transports to rebuild the two derivative orders d_{uv} and d_{vu}. The commutator boundary is then transported by the same boundary construction used in Definition 134.5.

For the Bianchi row, repeat the argument for each cyclic placement of the transported tangent rows. The three transported boundary rows append to the transported cyclic ledger, and the unchanged provenance row identifies the result as the same curvature package read through classified source rows. □

Theorem 135.44 (Curvature ledger exactness obligation)**.** *The* $\mathsf{Curvature}^{\uparrow}$ *obligation surface consists exactly of the source* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *packet, the base, fibre, section, two tangent-direction rows, two derivative endpoint rows, the connection commutator-boundary row, the antisymmetric bracket transport row, the cyclic Bianchi ledger row, the classifier transport rows generated by* hsame *and the connection classifier, and the* Pkg *provenance row. No coordinate chart, host tensor object, covariant exterior derivative, characteristic-class row, or external bracket operation is part of the certificate surface.*

Proof. Every accepted curvature row projects from the tuple in Definition 135.1. Projection gives the source certificate, the geometric rows, the derivative endpoints, the commutator boundary, the ledger row, and the provenance row.

Conversely, each listed row is consumed by one of the certificate obligations. The classifier obligation uses the source comparisons and boundary hsame row; antisymmetry uses the exchanged

tangent pair and reversed boundary row; the Bianchi obligation uses the three cyclic boundary rows; tensorial transport uses the section and tangent replacement rows.

Therefore the ledger is exact: all public certificate moves are generated by the displayed BHist, Cont, hsame, and Pkg rows, and every displayed row has a named role in one of those moves. Nothing remains to be read from an uncarried host tensor. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier ledger exactness obligation`

Theorem 135.45 (Curvature NameCert scope obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{Curvature}^\uparrow}$ *surface exported to* $\mathsf{ChernWeil}^\uparrow$ *is the curvature bracket carrier, its connection-carrier scope, its classifier stability, its antisymmetric bracket row, its first Bianchi cyclic ledger, its tensorial transport row, and its ledger exactness row over* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$*. Any downstream packet consuming* $\mathsf{NameCert}_{\mathsf{Curvature}^\uparrow}$ *may project the bracket boundary* κ*, the cyclic ledger, and the provenance rows, but it may not assume a stronger analytic curvature form or a standard bridge.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier semantic name certificate`

Proof. Collect the carrier of Definition 135.1 together with the five obligation rows proved above. These rows provide the public carrier, scope, classifier, stability, ledger, and exactness fields required of a BEDC naming certificate.

The downstream projection to $\mathsf{ChernWeil}^\uparrow$ is by ordinary tuple projection. It receives the boundary row, the cyclic ledger, the source connection certificate, and the provenance fields, all still expressed as finite BHist and Cont data.

The final clause is exactness again: because the exported surface contains only those rows, no downstream consumer may read a coordinate curvature form, a de Rham class, or a standard bridge from $\mathsf{NameCert}_{\mathsf{Curvature}^\uparrow}$ alone. Such data must be supplied by its own certificate. □

Theorem 135.46 (Curvature exactness predicate surface)**.** *For* $\mathsf{Curvature}^\uparrow$*, exactness is precisely coverage, soundness, and completeness for the displayed bracket carrier, boundary classifier, Bianchi ledger, tensorial transport row, and* $\mathsf{ChernWeil}^\uparrow$ *consumer projection. The surface is read only over* BHist*,* Cont*,* hsame*,* Pkg*,* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$*, and* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$*: coverage says every displayed row has one of these roles; soundness says every accepted curvature move is generated by the displayed rows; completeness says every downstream curvature projection factors through them.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier exactness predicate surface`

Proof. Project the carrier row from Definition 135.1. It supplies the base, fibre, section, tangent-direction, derivative endpoint, commutator-boundary, provenance, and source-certificate rows, with the $\mathsf{Bundle}^\uparrow$ and $\mathsf{Manifold}^\uparrow$ dependencies inherited through $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$.

Next project the classifier, antisymmetry, Bianchi, tensorial transport, ledger exactness, and consumer-boundary obligations. These theorems account for every public use of the bracket carrier: boundary comparison, exchange of tangent directions, cyclic ledger formation, certified replacement of section and tangent rows, and projection to $\mathsf{ChernWeil}^\uparrow$.

Coverage follows because each displayed row is named in one of those obligations. Soundness follows because each accepted move is built by finite Cont transport, hsame comparison, tuple projection, and the recorded Pkg provenance. Completeness follows because the downstream consumer receives exactly the boundary, cyclic ledger, source certificate, dependency certificate, and provenance rows already carried by the same tuple. □

Theorem 135.47 (Curvature standard bridge boundary)**.** *Any standard reading of* $\mathsf{Curvature}^{\uparrow}$ *may expose only the displayed connection-boundary row, antisymmetric classifier row, Bianchi ledger row, tensorial transport row, and exactness surface. It may not expose a coordinate curvature tensor, a de Rham representative, a characteristic class, host equality, or a* bridgeChecked *claim from* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *alone.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier standard bridge boundary`

Proof. Apply the exactness surface of Definition 135.46. A standard reader can project only rows that are covered by the carrier and generated by the named certificate obligations.

The permitted rows are therefore the connection-boundary row, the antisymmetric classifier row, the Bianchi ledger row, the tensorial transport row, and the exactness predicate itself. These are precisely the rows already exported by the NameCert scope theorem.

Apply Definition 135.45 for the negative boundary. Since the public certificate contains no coordinate curvature tensor, de Rham representative, characteristic class, host equality row, or standard-bridge witness, none of those objects can be recovered from the $\mathsf{Curvature}^{\uparrow}$ certificate without an additional certificate carrying that data. □

Theorem 135.48 (Curvature ten-field status record)**.** *The roadmap ten-field template for* $\mathsf{Curvature}^{\uparrow}$ *is populated as follows:*

1. *the traditional target is the curvature bracket of a connection;*
2. *the BEDC source is* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *together with the* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{Manifold}^{\uparrow}$ *dependency rows;*
3. *the carrier is Definition 135.1;*
4. *the classifier entry is the boundary classifier row of this section;*
5. *the stability entry is the antisymmetry, Bianchi, and tensorial row collection;*
6. *the ledger entry is visible ledger exactness;*
7. *the core theorem list is the named obligation row collection in this section;*
8. *the exactness entry is Definition 135.46;*
9. *the standard bridge entry is Definition 135.47;*
10. *the status entry keeps closure and verification as independent axes.*

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureUp StdBridge`

Proof. Read the template fields from the carrier and obligation rows already established in this section. The traditional target is the bracket interpretation of the connection boundary, while the BEDC source and dependency rows are projected from $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ and its $\mathsf{Bundle}^{\uparrow}$ and $\mathsf{Manifold}^{\uparrow}$ certificates.

The carrier, classifier, antisymmetry, Bianchi, tensorial transport, ledger exactness, and core obligation entries are exactly the displayed rows named by the section labels. The exactness and standard-bridge entries are supplied by Definition 135.46 and Definition 135.47.

The status field records no extra mathematical object. It reports the theory closure and verification state on separate axes, so a paper-side certificate surface does not by itself assert a Lean theorem, host equality, or standard bridge. □

Theorem 135.49 (Curvature NameCert finite basis)**.** *The public* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *basis is exactly the finite row family consisting of the carrier tuple of Definition 135.1, the connection-carrier scope row of Definition 135.39, the bracket classifier obligation of Definition 135.40, the antisymmetric bracket obligation of Definition 135.41, the Bianchi ledger obligation of Definition 135.42, the tensorial transport obligation of Definition 135.43, the ledger exactness obligation of Definition 135.44, and the scope obligation of Definition 135.45. Every public curvature certificate row is a projection from this family, and every row in the family is consumed by at least one public* $\mathsf{Curvature}^{\uparrow}$ *certificate field.*

Proof. Project first from the carrier tuple in Definition 135.1. This gives the source connection certificate, base row, fibre row, section row, two tangent-direction rows, derivative endpoints, commutator boundary, cyclic ledger, classifier transports, and provenance row.

The named obligation rows then identify how those projections are used. The connection-carrier scope row fixes the inherited source surface; the bracket classifier and antisymmetric bracket rows control boundary comparison; the Bianchi and tensorial transport rows control cyclic and replacement ledgers; and ledger exactness states that no additional public row is available.

Conversely, each listed row has a named role in the public certificate fields. Therefore the displayed family is a finite basis for $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ rather than a sample of a larger analytic curvature interface. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier finite basis`

Theorem 135.50 (Curvature consumer row coverage)**.** *Every downstream consumer of the finite basis in Definition 135.49 receives only* BHist*,* Cont*,* hsame*,* Pkg*, and* SigRel *rows. Moreover, each row visible to such a consumer has a source in one of the basis theorems: carrier and boundary rows come from the carrier tuple and connection-carrier scope row, classifier rows from the bracket classifier obligation, reversal rows from the antisymmetric bracket obligation, cyclic rows from the Bianchi ledger obligation, replacement rows from the tensorial transport obligation, provenance rows from ledger exactness, and export permission from the scope obligation.*

Proof. Case split over the finite basis named in Definition 135.49.

1. The carrier tuple and connection-carrier scope cases expose BHist rows together with inherited Pkg provenance from $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$.
2. The bracket classifier, antisymmetric bracket, Bianchi ledger, and tensorial transport cases expose only Cont, hsame, and SigRel comparison rows between already carried histories.
3. The ledger exactness and scope cases expose no new row sort. They identify which carried BHist, Cont, hsame, Pkg, and SigRel rows may be projected by a downstream certificate.

These cases cover the whole basis. Hence a consumer cannot obtain a coordinate curvature form, host tensor operation, de Rham class, or standard bridge by projecting $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ alone. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureBracketCarrier consumer row coverage`

Theorem 135.51 (Curvature Chern-Weil export determinacy)**.** *Let* $\mathcal{R}$ *and* $\mathcal{R}'$ *be two public* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *packets whose rows agree on the finite basis of Definition 135.49. Then their projections to the* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope determine the same curvature boundary* κ*, Bianchi ledger, tensorial transport row, and provenance packet. Consequently the curvature component of the* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope is determined by the finite* $\mathsf{Curvature}^{\uparrow}$ *basis and not by any uncarried connection form or host-level curvature object.*

Proof. Apply Definition 135.49 to both packets. Agreement on the basis identifies the carrier projections, the boundary row, the cyclic ledger, the tensorial transport row, and the Pkg provenance row.

Next use Definition 135.25. That theorem packages exactly those curvature projections as the input side consumed by $\mathsf{ChernWeil}^{\uparrow}$, while excluding any de Rham row, characteristic-class equality, standard bridge, or Lean verification target.

Finally project the source envelope of Definition 272.1. Its curvature component is the public $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ packet carrying the boundary and provenance exported by the scope rule, so the equal basis rows force the same curvature source component in both envelopes. □

Lean4 ✓ `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope export determinacy`

Closure status for $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$.

Constructive story: A curvature row starts from a carried $\mathsf{Connection}^{\uparrow}$ packet and selects two displayed tangent-direction derivative routes whose commutator boundary is recorded as a BHist bracket endpoint. Histories are accepted only when that bracket carrier can project back to its connection source, the two derivative rows, the antisymmetric bracket row, and the finite provenance rows; two accepted rows classify together when connection classifiers and hsame transport all bracket, boundary, and source projections. The source covers these connection-boundary packets and the Chern–Weil envelope rows derived from them, the pattern records antisymmetry, Bianchi ledger transport, tensorial section transport, and consumer coverage, and the ledger keeps connection provenance, bracket endpoints, finite continuation routes, and Chern–Weil handoff rows visible. The carrier is inhabited by the displayed boundary source, and the classifier obligations are the inherited reflexive, symmetric, and transitive transports of hsame together with connection-carrier stability, so carrier closure is just repackaging the same transported bracket row. A history h lies in the signature gap of a $\mathsf{Curvature}^{\uparrow}$ token p when the source accepts a displayed curvature bracket for h, the signature relation pairs it with an answer bracket and evidence ledger, and token introduction names that answer as p; unfolded, the connection source, tangent-pair boundary, bracket endpoint, and Chern–Weil envelope projections match the rows encoded by p. The consumer rows let downstream Chern–Weil and ledger exactness arguments read the curvature packet, but no coordinate curvature tensor, de Rham representative, characteristic class, or standard bridge is part of this surface. The chapter registers this boundary-bracket construction under the name $\mathsf{Curvature}^{\uparrow}$; curvature is a finite connection-ledger row here, not a host tensor field assumed from outside.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The BHist curvature bracket carrier, ConnectionUp boundary projection, Chern-Weil source-envelope certificate, and standard bridge boundary are closed over the finite NameCert basis rows displayed in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CurvatureUp.CurvatureChernWeilSourceEnvelope\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: No coordinate curvature tensor, de Rham representative, or characteristic class is closed here.

Upgrade path: No further upgrade – chapter at top of formal axis.

The $\mathsf{DiffForm}^{\uparrow}$ hub routes antisymmetric multilinear carrier rows through chapter 96 and chapter 87, and its exterior-derivative boundary is the sibling input consumed by chapter 137.

Chapter 136

A Concrete Naming Certificate for $\mathsf{DiffForm}^{\uparrow}$

The differential-form interface $\mathsf{DiffForm}^{\uparrow}$ packages the carrier of antisymmetric multilinear maps on the tangent bundle of a manifold, together with the wedge product. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87), and is the substrate for the de Rham complex.

136.1 Carrier definition

Definition 136.1 (Differential-form BHist form carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T *whose tensor rows are read over the tangent-bundle histories of* M*. A differential-form BHist row is a tuple of histories*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda) : \mathsf{BHist}^6$$

accepted by the predicate $\mathsf{DiffFormBHistCarrier}_{M,T}$ *exactly when the following BEDC rows are all present:*

1. $\mathsf{UnaryHistory}(d)$ *records the form degree.*
2. $\mathcal{P} : \mathsf{ProbeBundle}$ *is the visible finite spine of tangent probe names, and each bundled probe row is carried by the tangent-source surface exposed by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*.*
3. τ *is a* $\mathsf{TensorProduct}^{\uparrow}$ *multilinear row over the bundled tangent probes, with continuation ledgers* Cont *recording the ordered tensor concatenations used by the row.*
4. σ *is the scalar endpoint produced from that tensor row, classified by the scalar classifier retained by the manifold and tensor sources.*
5. α *is the visible permutation ledger: for every displayed adjacent swap of bundled probe rows, the transported scalar endpoint is classified with the signed endpoint demanded by antisymmetry.*
6. λ *records the source names* M *and* T*, the degree row, the probe bundle, the tensor row, the scalar endpoint row, and the antisymmetry rows as the only accepted provenance for the form row.*

Definition 136.2 (Differential-form BHist form classifier)**.** *For carried form rows*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and

$$\omega' = (d', \mathcal{P}', \tau', \sigma', \alpha', \lambda'),$$

the classifier $\omega \sim_{\mathsf{DiffForm}^\uparrow} \omega'$ *holds when the degree rows are* hsame*, the scalar endpoints agree pointwise under the retained scalar classifier for every matched multilinear probe row, and the antisymmetry and tensor-source ledgers match as visible BHist rows:*

$$\mathsf{hsame}(d, d')$$

and for every matched bundled multilinear row, $\sigma \sim_R \sigma'$

$$\textit{and } \mathsf{hsame}(\alpha, \alpha') \wedge \mathsf{hsame}(\tau, \tau') \wedge \mathsf{hsame}(\lambda, \lambda').$$

Here $\sim_R$ *is the scalar classifier already present in the upstream scalar surface; the differential-form classifier introduces no classifier outside* hsame *and the displayed source classifiers.*

Lean4 (definition) `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier`

136.2 The certificate

Definition 136.3 (Differential-form BHist NameCert core rows)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *source* T *as in Definition 136.1. The* $\mathsf{DiffForm}^\uparrow$ *BHist NameCert core-row package consists of the following ten displayed template fields:*

1. *the carrier row* $\mathsf{DiffFormBHistCarrier}_{M,T}$*;*
2. *the classifier row* $\sim_{\mathsf{DiffForm}^\uparrow}$ *of Definition 136.2;*
3. *the componentwise* hsame *transport row for carried forms;*
4. *the degree history row* d*;*
5. *the tangent-probe* ProbeBundle *row* $\mathcal{P}$*;*
6. *the* $\mathsf{TensorProduct}^\uparrow$ *multilinear row* τ*;*
7. *the tensor-continuation ledger* Cont *attached to* τ*;*
8. *the scalar endpoint classifier row retained from the upstream manifold and tensor sources;*
9. *the antisymmetry permutation ledger* α*;*
10. *the standard-bridge slot, fixed to absent data.*

These fields are read as template data for the certificate only. The package does not introduce an exterior-derivative row, a closedness predicate, a nondegeneracy predicate, or a standard bridge.

Theorem 136.4 (Differential-form BHist carrier obligation)**.** *The carrier row in Definition 136.3 is exhausted by the six coordinates*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

of Definition 136.1. In particular, accepting a $\mathsf{DiffForm}^\uparrow$ *carrier row adds no seventh coordinate and no hidden source outside the displayed degree, probe-bundle, tensor, scalar-endpoint, antisymmetry, and source-ledger histories.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistCarrier coordinate ledger`

Proof. Invert the acceptance predicate $\mathsf{DiffFormBHistCarrier}_{M,T}$ displayed in Definition 136.1. A carried row is presented as the six-tuple $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, and each acceptance clause projects one of those six histories or one relation among histories already named by the tuple.

The degree clause projects d, the bundled tangent-probe clause projects $\mathcal{P}$, the multilinear tensor clause projects τ, the scalar endpoint clause projects σ, the permutation ledger clause projects α, and the source-provenance clause projects λ. No clause has a constructor that allocates another coordinate.

It remains to check that the core-row package of Definition 136.3 does not enlarge this carrier. Its transport, classifier, continuation, endpoint-classifier, and bridge slots refer back to the six displayed carrier histories and to upstream certificate sources. Thus carrier acceptance is exactly the six coordinate surface of Definition 136.1. □

Theorem 136.5 (Differential-form BHist classifier obligation)**.** *The classifier row in Definition 136.3 uses only degree* hsame*, matched tensor and scalar comparisons, and visible ledger* hsame*. It introduces no classifier other than* hsame *on displayed BHist rows and the scalar classifier already retained by the manifold and tensor sources.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier obligation`

Proof. Project the classifier definition from Definition 136.2. The first clause is exactly $\mathsf{hsame}(d, d')$, so degree comparison is a visible BHist comparison and not a new differential-form relation.

For the tensor and scalar part, the statement quantifies only over matched bundled multilinear rows. The tensor match is the visible comparison of the displayed tensor histories, while the endpoint comparison is the retained scalar classifier $\sim_R$ supplied by the upstream scalar surface. The classifier therefore reuses source comparisons already present in M and T.

The remaining clauses are $\mathsf{hsame}(\alpha, \alpha')$, $\mathsf{hsame}(\tau, \tau')$, and $\mathsf{hsame}(\lambda, \lambda')$. These are visible ledger comparisons on the antisymmetry, tensor, and source-provenance histories. Hence every accepted classifier step is one of the claimed degree, matched tensor/scalar, or ledger comparisons. □

Theorem 136.6 (Differential-form BHist ledger exactness obligation)**.** *Every ledger row accepted by the core-row package of Definition 136.3 is visible BHist provenance from BHist,* hsame*,* Cont*,* ProbeBundle*,* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$*. No accepted ledger row may cite a hidden differential-form primitive or a standard bridge.*

Proof. First inspect the carrier ledger. By Definition 136.1, λ records only the source names M and T, the degree row, the probe bundle, the tensor row, the scalar endpoint row, and the antisymmetry rows. These are all BHist provenance entries or rows exported by the two named upstream certificates.

Next inspect the tensor and probe ledgers. The probe data are a finite ProbeBundle spine whose tangent rows are supplied by $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$. The tensor data are supplied by $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$, and their ordered concatenations are recorded by Cont. Thus the only tensor-continuation provenance admitted by the core package is visible continuation data over the displayed probe rows.

Finally inspect the classifier and bridge slots. Classifier ledgers are hsame comparisons on degree, tensor, antisymmetry, and source histories, together with the scalar endpoint classifier retained from the upstream sources. The standard-bridge slot in Definition 136.3 is fixed to absent data, so there is no additional bridge ledger to accept. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistLedger exactness obligation`

Theorem 136.7 (Differential-form classifier reflexivity obligation)**.** *Every row carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *satisfies its own* $\mathsf{DiffForm}^{\uparrow}$ *classifier:*

$$\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega.$$

Equivalently, the BHist row accepted by the displayed carrier gives a reflexive NameCert classifier row for the $\mathsf{DiffForm}^{\uparrow}$ *certificate over the fixed* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *sources.*

Proof. Unpack the classifier in Definition 136.2. The degree, tensor, antisymmetry, and source-ledger clauses are all reflexive hsame comparisons between the corresponding visible BHist rows.

For each matched multilinear probe row in the displayed ProbeBundle, the scalar endpoint is compared with itself by reflexivity of the scalar-source classifier retained by the upstream $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ data. These componentwise reflexivity clauses are exactly the clauses required by the $\mathsf{DiffForm}^{\uparrow}$ classifier. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier reflexivity obligation`

Theorem 136.8 (Differential-form classifier symmetry obligation)**.** *For rows carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, the classifier of Definition 136.2 is symmetric: if* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \eta$*, then* $\eta \sim_{\mathsf{DiffForm}^{\uparrow}} \omega$*.*

Proof. Unpack the given classifier row through Definition 136.2. The degree comparison and the tensor, antisymmetry, and source-ledger comparisons are visible hsame relations on BHist rows, so symmetry of hsame reverses each of those clauses.

The matched scalar endpoints are compared by the scalar-source classifier retained from the $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ sources. Reversing that retained classifier comparison at each bundled ProbeBundle row gives the scalar clause for $\eta \sim_{\mathsf{DiffForm}^{\uparrow}} \omega$, and the reversed hsame ledger clauses supply the remaining classifier fields. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier symmetry obligation`

Theorem 136.9 (Differential-form classifier transitivity obligation)**.** *For rows carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, the classifier of Definition 136.2 is transitive: if* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \eta$ *and* $\eta \sim_{\mathsf{DiffForm}^{\uparrow}} \theta$*, then* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \theta$*.*

Proof. Unpack the two classifier rows using Definition 136.2. Transitivity of hsame composes the degree comparisons, and the same transitive hsame composition applies to the tensor, antisymmetry, and source-ledger BHist rows.

For each matched multilinear row in the common ProbeBundle surface, compose the scalar endpoint comparisons through the scalar-source classifier retained by $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$. The composed scalar clauses and the composed hsame clauses give precisely the classifier fields required for $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \theta$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier transitivity obligation`

Theorem 136.10 (Differential-form carrier hsame transport)**.** *Let* ω *be a row accepted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, and let* ω' *be obtained by replacing its six displayed histories with* hsame*-equal histories while keeping the same* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *sources. Then* ω' *is also accepted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*.*

Proof. Unpack the carrier row of Definition 136.1. The degree condition is transported by hsame for UnaryHistory.

The probe bundle is a closed ProbeBundle spine, so its bundled tangent rows are transported pointwise through the displayed manifold source classifier. The tensor row is then transported

through the $\mathsf{TensorProduct}^{\uparrow}$ classifier, retaining the same Cont ledgers after source, probe, and endpoint histories are replaced by hsame-equal rows.

It remains to transport the scalar and antisymmetry rows. The scalar endpoint uses the retained scalar classifier, and its pointwise transport follows from classifier stability in the source surface. Each antisymmetry row is a visible permutation ledger over the same probe bundle; transporting the two scalar endpoints of the swap row gives the required signed comparison. Repacking these transported fields gives the carried row for ω'. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistCarrier hsame transport`

Theorem 136.11 (Differential-form classifier stability)**.** *For rows carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, the classifier of Definition 136.2 is stable under carrier transport: if* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \eta$ *and both rows are transported to* ω' *and* η' *by componentwise* hsame *comparisons, then* $\omega' \sim_{\mathsf{DiffForm}^{\uparrow}} \eta'$*.*

Proof. The degree comparison is the transitive composite of the transport comparison for ω, the original degree comparison, and the symmetric transport comparison for η.

For each bundled multilinear probe row, use the tensor-source classifier to identify the matched tensor rows after transport. The scalar endpoints are then compared by the scalar classifier stability already retained in the upstream scalar surface.

The antisymmetry and source-ledger clauses are pure hsame clauses on visible BHist rows. Composing their original comparisons with the two transport comparisons gives the required ledger matches for the transported rows. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistClassifier hsame component stability`

Theorem 136.12 (Differential-form probe-slot arity transport)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and

$$\omega' = (d', \mathcal{P}', \tau', \sigma', \alpha', \lambda')$$

be rows carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$*. Suppose* $\mathsf{hsame}(d, d')$ *and the displayed* $\mathsf{ProbeBundle}$ *spines of* $\mathcal{P}$ *and* $\mathcal{P}'$ *are matched componentwise through the tangent-source classifier supplied by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*. Then the visible arity of the probe slots in* $\mathcal{P}$ *is the same as the visible arity of the probe slots in* $\mathcal{P}'$*. Moreover, every transported multilinear tensor row remains classified by the* $\mathsf{TensorProduct}^{\uparrow}$ *source over those transported tangent probes.*

Proof. Unpack the carrier rows in Definition 136.1. The degree entries are unary history rows, so $\mathsf{hsame}(d, d')$ transports the displayed degree without creating or deleting a probe slot.

The probe component is a closed $\mathsf{ProbeBundle}$ spine. Induct on this spine. The empty spine has arity zero on both sides. In the cons case, the head tangent probe is transported through the classifier retained by $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and the induction hypothesis preserves the arity of the tail. Therefore the visible slot count is preserved at each spine constructor.

It remains to place the transported multilinear row back in the tensor source. The tensor component of a carried differential-form row is accepted only as a $\mathsf{TensorProduct}^{\uparrow}$ row over the bundled tangent probes. Since the probe classifier transport is componentwise and the tensor row retains its displayed Cont provenance, tensor-source classifier transport reclassifies τ' as a multilinear row over the transported probe bundle. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeAligned hsame arity transport`

Theorem 136.13 (Differential-form adjacent-swap involution ledger)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

be carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$*. If a displayed row of the antisymmetry ledger* α *records an adjacent swap of two neighboring probe slots and classifies the resulting scalar endpoint with the signed endpoint required by antisymmetry, then* α *also determines the reverse adjacent swap row. Applying the displayed adjacent swap and then its reverse returns endpoints classified with the original scalar endpoint under the retained scalar classifier, and the provenance for both swap rows remains recorded in* λ*.*

Proof. Read the chosen adjacent-swap row from the visible antisymmetry ledger of Definition 136.1. Its endpoints are scalar rows classified by the scalar classifier retained from the manifold and tensor sources, and its provenance is one of the rows listed in λ.

The reverse row uses the same two neighboring probe slots in the opposite order. The scalar comparison for the reverse row is obtained by symmetry of the retained scalar classifier, while the comparison between the double-swap endpoint and the original endpoint is the transitive composite of the forward and reverse scalar comparisons.

No hidden row is introduced in this construction. Both the displayed swap row and the reverse swap row are retained as BHist provenance in λ, so the two applications return to endpoints classified with the original scalar endpoint inside the same carried differential-form row. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormAdjacentSwapInvolutionLedger reverse closure`

Theorem 136.14 (Differential-form wedge degree additivity ledger)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and

$$\eta = (e, \mathcal{Q}, \upsilon, \rho, \beta, \mu)$$

be carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$*. If an admitted wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ *is carried with output degree row* $d \star e$*, then its visible degree ledger contains the unary continuation row* $\mathsf{Cont}(d, e, d \star e)$*. If the two input form rows are transported by componentwise classifier comparisons with degree rows* d' *and* e' *satisfying* $\mathsf{hsame}(d, d')$ *and* $\mathsf{hsame}(e, e')$*, the wedge degree row is transported to a row* $d' \star e'$ *classified with the same unary continuation/append degree construction.*

Proof. Start with the admitted wedge row and apply Definition 136.15. The wedge carrier contains the visible tensor concatenation row joining the tensor component of ω to the tensor component of η, and the carrier provenance records the degree row of the wedge output.

The input degree rows are unary histories. Closure of unary histories under continuation supplies the displayed degree ledger $\mathsf{Cont}(d, e, d \star e)$ for the output degree row. This ledger is part of the same BHist provenance that carries the wedge row, so it is visible to downstream exterior-derivative consumers.

Now transport the two input form rows. The degree comparisons $\mathsf{hsame}(d, d')$ and $\mathsf{hsame}(e, e')$ transport the two unary inputs, and the componentwise classifier comparisons transport the probe, tensor, scalar, and antisymmetry rows. Applying unary-history closure under the transported continuation yields $d' \star e'$ with its corresponding Cont row.

Repackage the transported probe bundle, tensor concatenation, scalar endpoint, antisymmetry ledger, and degree ledger as the wedge carrier row. The resulting wedge row has degree additivity classified by the same differential-form carrier and classifier surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDegreeLedger degree additivity transport`

Theorem 136.15 (Differential-form wedge ledger coverage)**.** *A wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ *is admitted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *only when its ledger visibly contains a* $\mathsf{TensorProduct}^{\uparrow}$ *concatenation row joining the tensor row of* ω *to the tensor row of* η*. When that concatenation ledger is present, carrier and classifier stability of the wedge row follow by transporting the two input form rows through the tensor classifier and then through the retained scalar classifier.*

Proof. Read the wedge candidate through the carrier definition. The only provenance accepted for a form row is its displayed ledger, and the tensor component of a wedge must therefore be the visible tensor concatenation recorded there. If no such $\mathsf{TensorProduct}^{\uparrow}$ continuation row appears, the candidate has no accepted carrier witness.

Assume the concatenation row is present. Transport the left and right form rows by Definition 136.10. The $\mathsf{TensorProduct}^{\uparrow}$ source classifies the concatenated tensor endpoint after the two input tensor rows have been transported, and the scalar endpoint is classified by applying the retained scalar classifier to the transported multilinear endpoints.

The antisymmetry ledger of the wedge is the concatenation of the two input antisymmetry ledgers together with the displayed cross-permutation rows. Each row is a visible BHist row, so hsame transport preserves it. These fields reassemble into the carried wedge row and into the classifier comparison for the transported wedge endpoints. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeLedger coverage`

Definition 136.16 (Differential-form wedge probe concatenation ledger)**.** *For carried form rows*

$$\omega = (d_\omega, \mathcal{P}_\omega, \tau_\omega, \sigma_\omega, \alpha_\omega, \lambda_\omega)$$

and

$$\eta = (d_\eta, \mathcal{P}_\eta, \tau_\eta, \sigma_\eta, \alpha_\eta, \lambda_\eta),$$

the wedge probe concatenation ledger is the visible BHist package whose output probe bundle is

$$\mathcal{P}_{\omega\wedge\eta} = \mathsf{bundleAppend}(\mathcal{P}_\omega, \mathcal{P}_\eta).$$

It records the left input ledger λ_ω*, the right input ledger* λ_η*, the two input* $\mathsf{ProbeBundle}$ *spines, the displayed output* $\mathsf{ProbeBundle}$ *spine, and the same tensor-concatenation provenance row used by the wedge ledger coverage of Definition 136.15.*

Theorem 136.17 (Differential-form wedge probe concatenation coverage)**.** *Every wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ *admitted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *carries the wedge probe concatenation ledger of Definition 136.16. Its output probe bundle has length equal to the sum of the two input probe-bundle lengths, and a probe row is in the output bundle exactly when it is in the left input spine or in the right input spine.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeProbeConcatenationLedger coverage`

Proof. Apply Definition 136.15. The tensor concatenation row in that ledger is read over the same two probe spines that occur in the two carried input form rows, so the output probe surface is forced to be $\mathsf{bundleAppend}(\mathcal{P}_\omega, \mathcal{P}_\eta)$.

Now induct on the left $\mathsf{ProbeBundle}$ spine. The empty case exposes the right input spine as the whole output, giving the length and membership claims directly from the right bundle.

In the cons case, the head probe remains the head of the output spine and the induction hypothesis applies to the tail. Reassembling the head case with the tail comparison gives length additivity for bundleAppend and the membership iff splitting membership into the left spine or the right spine.

The BHist row of Definition 136.16 stores exactly these input ledgers, the output bundle, and the tensor provenance row. Therefore every admitted wedge row carries the displayed probe concatenation surface. □

Theorem 136.18 (Differential-form wedge probe concatenation classifier stability)**.** *Let (ω, η) and (ω', η') be two pairs of carried* $\mathsf{DiffForm}^{\uparrow}$ *rows with $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$ and $\eta \sim_{\mathsf{DiffForm}^{\uparrow}} \eta'$. If their wedge rows are admitted, then the wedge probe concatenation ledgers of Definition 136.16 are related by* hsame*, and the transported wedge outputs remain classified under $\sim_{\mathsf{DiffForm}^{\uparrow}}$.*

Proof. Use Definition 136.11 to transport the two input form classifiers componentwise. The probe-slot part of that transport is controlled by Definition 136.12, so the left input probe bundles match and the right input probe bundles match with the same visible arities.

Apply Definition 136.15 to the two admitted wedge rows. Their tensor concatenation rows transport through the $\mathsf{TensorProduct}^{\uparrow}$ classifier, and the scalar endpoint rows transport through the retained scalar classifier.

Finally transport the $\mathsf{bundleAppend}(\mathcal{P}_\omega, \mathcal{P}_\eta)$ row componentwise along the two input probe-bundle comparisons. The length and membership clauses are preserved by Definition 136.17, and the left ledger, right ledger, output probe bundle, and tensor provenance rows repack as the required hsame comparison between the two probe concatenation ledgers. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeProbeConcatenation classifier stability`

Definition 136.19 (Differential-form wedge associator ledger)**.** *For carried form rows ω, η, and θ in* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, the visible wedge associator ledger is the BHist package recording four rows: the* $\mathsf{TensorProduct}^{\uparrow}$ *concatenation spine for $(\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta) \wedge_{\mathsf{DiffForm}^{\uparrow}} \theta$, the* $\mathsf{TensorProduct}^{\uparrow}$ *concatenation spine for $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} (\eta \wedge_{\mathsf{DiffForm}^{\uparrow}} \theta)$, the* Cont *associativity comparison between those tensor spines, and the common scalar endpoint classifier obtained by transporting through the retained scalar classifier. It also records the permutation-ledger comparison required to identify the two antisymmetry ledgers after the tensor associativity comparison has been applied.*

Theorem 136.20 (Differential-form wedge associator closure)**.** *Let ω, η, and θ be carried* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *rows. If the two binary wedge ledgers needed for $(\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta) \wedge_{\mathsf{DiffForm}^{\uparrow}} \theta$ and the two binary wedge ledgers needed for $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} (\eta \wedge_{\mathsf{DiffForm}^{\uparrow}} \theta)$ are visible, then the associator ledger of Definition 136.19 carries both parenthesized wedge rows and classifies them as equal under $\sim_{\mathsf{DiffForm}^{\uparrow}}$.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDegreeLedger associator closure`

Proof. Apply Definition 136.15 to $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ and then to the wedge with θ. This constructs the left-parenthesized carrier row and its scalar endpoint classifier from the two visible binary ledgers.

Apply the same theorem to $\eta \wedge_{\mathsf{DiffForm}^{\uparrow}} \theta$ and then to the wedge with ω. This constructs the right-parenthesized carrier row and its scalar endpoint classifier from the other two visible binary ledgers.

The two tensor endpoints are compared by the Cont associativity row in the visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation ledger. Transport the scalar endpoint classifier across that comparison and the retained scalar classifier from Definition 136.2.

It remains to identify the antisymmetry ledgers. The associator ledger carries the required permutation comparison, and each permutation row is a visible BHist row. Composing those rows by hsame transitivity gives the $\mathsf{DiffForm}^{\uparrow}$ classifier equality between the two parenthesized wedge rows. □

Theorem 136.21 (Differential-form exterior-algebra obligation surface)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *exterior-algebra surface available to* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *consists of carrier transport, classifier stability, binary wedge ledger coverage, wedge probe concatenation coverage, the wedge associator ledger, and the wedge degree ledger. This surface does not assert* $d^2 = 0$*, closedness, nondegeneracy, or contact nonintegrability.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormUp StdBridge`

Proof. Carrier transport and classifier stability are supplied by Definition 136.10 and Definition 136.11. Binary wedge admission and transport are exactly Definition 136.15, while the visible output probe spine is the probe concatenation ledger of Definition 136.16 with coverage and classifier stability from Definition 136.17 and Definition 136.18.

Associativity of the exposed exterior product is not an unrecorded algebraic law. It is the visible associator ledger of Definition 136.19, closed under the carrier and classifier by Definition 136.20. Degree bookkeeping is the visible unary-history ledger of Definition 136.22.

Combining these rows with the NameCert surface of Definition 136.165 and the downstream scope of Definition 136.61 gives exactly the surface available to de Rham, symplectic, and contact consumers. The operator identity $d^2 = 0$ and the geometric conditions named in the statement are not rows of this $\mathsf{DiffForm}^{\uparrow}$ surface, so they remain obligations of the downstream certificates that use it. □

Definition 136.22 (Differential-form wedge degree ledger)**.** *For carried form rows*

$$\omega = (d_\omega, \mathcal{P}_\omega, \tau_\omega, \sigma_\omega, \alpha_\omega, \lambda_\omega)$$

and

$$\eta = (d_\eta, \mathcal{P}_\eta, \tau_\eta, \sigma_\eta, \alpha_\eta, \lambda_\eta),$$

a wedge degree ledger is the visible BHist row $\delta_{\omega,\eta}$ *accepted by* $\mathsf{DiffFormWedgeDegreeLedger}_{M,T}$ *exactly when it displays the two input degree rows, an output degree row* $d_{\omega\wedge\eta}$*, and a continuation row*

$$\mathsf{Cont}(d_\omega, d_\eta, d_{\omega\wedge\eta})$$

whose unary-history endpoint records the concatenation/addition of the two input degrees. The only accepted provenance for $\delta_{\omega,\eta}$ *is the pair of input form ledgers* $\lambda_\omega, \lambda_\eta$*, the two* UnaryHistory *witnesses for* d_ω *and* d_η*, the displayed* Cont *row, the output* UnaryHistory *witness, and the tensor concatenation row used by the wedge ledger of Definition 136.15.*

Theorem 136.23 (Differential-form wedge degree classifier stability)**.** *Let* (ω, η) *and* (ω', η') *be two pairs of carried* $\mathsf{DiffForm}^{\uparrow}$ *rows with* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$ *and* $\eta \sim_{\mathsf{DiffForm}^{\uparrow}} \eta'$*. If* $\delta_{\omega,\eta}$ *and* $\delta_{\omega',\eta'}$ *are the wedge degree ledgers of Definition 136.22, and if the corresponding wedge rows are admitted by the wedge ledger coverage of Definition 136.15, then*

$$\mathsf{hsame}(\delta_{\omega,\eta}, \delta_{\omega',\eta'})$$

and the wedge rows remain classified under $\sim_{\mathsf{DiffForm}^{\uparrow}}$*.*

Proof. The classifier comparison on the two input form pairs includes hsame comparisons on their degree rows by Definition 136.2. Transporting UnaryHistory along these comparisons gives the same unary degree witnesses for the two transported inputs.

The displayed continuation row of Definition 136.22 is a BHist row. Continuation associativity and unit stability identify the output degree row obtained from the transported input degrees with the output degree row in the second wedge degree ledger, so the two degree ledgers are related by hsame.

For the remaining wedge fields, apply Definition 136.15. The tensor concatenation row transports through the $\mathsf{TensorProduct}^{\uparrow}$ classifier, and the scalar endpoint transports through the retained scalar classifier. Together with the degree ledger comparison above and the visible antisymmetry ledger transport, these rows repack as the required $\mathsf{DiffForm}^{\uparrow}$ classifier comparison for the wedge rows. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDegreeLedger classifier stability`

Theorem 136.24 (Differential-form de Rham input boundary)**.** *Any* $\mathsf{DeRham}^{\uparrow}$ *exterior-derivative row sourced from* $\mathsf{DiffForm}^{\uparrow}$ *may depend on the* $\mathsf{DiffForm}^{\uparrow}$ *source only through the carried* ProbeBundle*, the unary degree row, the multilinear* $\mathsf{TensorProduct}^{\uparrow}$ *tensor row, the scalar endpoint classifier, the antisymmetry ledger, wedge ledger coverage, and the wedge probe concatenation ledger of Definition 136.16 together with the wedge degree ledger of Definition 136.22. It cannot import* $d^2 = 0$*, closedness, nondegeneracy, or contact nonintegrability from the* $\mathsf{DiffForm}^{\uparrow}$ *certificate.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDerhamInputBoundary source scope`

Proof. Inspect the carrier of Definition 136.1. The rows visible to an exterior-derivative construction are the probe bundle, unary degree row, tensor row, scalar endpoint, antisymmetry ledger, and source ledger, all carried as BHist data.

The wedge interfaces exported by $\mathsf{DiffForm}^{\uparrow}$ are exactly the tensor coverage of Definition 136.15, the probe concatenation surface of Definition 136.16, and the degree ledger of Definition 136.22. Their stability is supplied by Definition 136.18 and Definition 136.23 together with the carrier and classifier stability theorems above.

The identities $d^2 = 0$, closedness, nondegeneracy, and contact nonintegrability require operator or geometric rows not listed in the $\mathsf{DiffForm}^{\uparrow}$ carrier. Therefore a downstream $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, or $\mathsf{Contact}^{\uparrow}$ certificate must state those rows in its own surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeInputRow derham input boundary`

136.3 DiffForm degree-probe support

Definition 136.25 (Differential-form degree-probe support row)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T *as in Definition 136.1. A differential-form degree-probe support row is the ten-field template whose traditional target is the finite degree surface of differential forms, and whose BEDC source data are BHist,* UnaryHistory*,* ProbeBundle*,* Cont*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*

The carrier field is the six-coordinate row $\mathsf{DiffFormBHistCarrier}_{M,T}$ *with displayed coordinates* $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$ *from Definition 136.1. The classifier field is* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *from Definition 136.2. The stability obligations are degree and probe transport, wedge concatenation, and carrier transport. The ledger fields are the unary degree witness* d*, the visible probe spine* $\mathcal{P}$*, the tensor row* τ*, the scalar endpoint* σ*, the antisymmetry ledger* α*, and the source ledger* λ*.*

The core theorem list consists of Definition 136.26, Definition 136.27, and Definition 136.28. The support row is accepted if and only if the displayed carrier row supplies exactly the unary degree witness and the visible ProbeBundle *spine consumed by the tensor, scalar, antisymmetry, and source-ledger rows above. There is no standard bridge field in this support row, and its verification status is a paper proof target.*

Theorem 136.26 (Differential-form degree-probe carrier admissibility)**.** *Every row carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *supplies the degree-probe support row of Definition 136.25. In particular, the degree coordinate is the visible* UnaryHistory *row* d*, the arity surface is the visible* ProbeBundle *spine* $\mathcal{P}$*, and the tensor, scalar-endpoint, antisymmetry, and source-ledger rows consume no hidden probe slot outside* $\mathcal{P}$*.*

Proof. Invert the carrier acceptance predicate displayed in Definition 136.1. A carried row is exactly the six-tuple $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, and the first two acceptance clauses name d as a UnaryHistory row and $\mathcal{P}$ as the finite ProbeBundle spine.

The remaining carrier clauses do not allocate another arity source. The tensor row τ is read over the bundled tangent probes in $\mathcal{P}$, the scalar endpoint σ is produced from that tensor row, and the antisymmetry ledger α ranges over displayed adjacent swaps of the same bundled probe rows.

Finally inspect the source ledger λ. Its accepted provenance is the source pair M, T together with the degree row, the probe bundle, the tensor row, the scalar endpoint row, and the antisymmetry rows. Repacking these visible fields gives precisely the ten-field support row, with no additional probe or degree coordinate to certify. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeSupport carrier admissibility`

Theorem 136.27 (Differential-form degree-probe transport)**.** *Let* ω *and* ω' *be rows carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *with* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$*. If* ω *supplies the degree-probe support row of Definition 136.25, then* ω' *supplies the transported degree-probe support row with the same visible probe-slot arity.*

Proof. Project the classifier comparison through Definition 136.2. Its degree clause gives hsame between the two unary degree histories, and its tensor, antisymmetry, and source-ledger clauses give visible hsame comparisons on the rows that record how the probe spine is consumed.

Apply Definition 136.12 to the two carried rows. The componentwise tangent-source transport of the two ProbeBundle spines preserves the visible number of probe slots, and the transported multilinear tensor row remains a $\mathsf{TensorProduct}^{\uparrow}$ row over those transported tangent probes.

It remains to move the support classifier itself. Use Definition 136.11 on the carried rows after the componentwise transport. The degree, tensor, scalar-endpoint, antisymmetry, and source-ledger comparisons repack into the same ten-field support row for ω', and the previous paragraph supplies the preserved probe arity. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeSupport transport`

Theorem 136.28 (Differential-form degree-probe wedge compatibility)**.** *Let* ω *and* η *be carried* $\mathsf{DiffForm}^{\uparrow}$ *rows supplying degree-probe support rows. If the wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ *is admitted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, then its support row is obtained by concatenating the two visible probe spines and by using the continuation row of Definition 136.22 to produce the output unary degree. Thus the wedge consumes exactly the arity displayed by the two input support rows.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeSupport wedge compatibility`

Proof. First apply Definition 136.17. The admitted wedge row carries the visible output bundle $\mathsf{bundleAppend}(\mathcal{P}_\omega, \mathcal{P}_\eta)$, its length is the sum of the two input bundle lengths, and membership in the output bundle is exactly membership in the left or right input spine.

Next use Definition 136.22. Its row displays the two input unary degree histories, the output unary degree history, and the continuation row

$$\mathsf{Cont}(d_\omega, d_\eta, d_{\omega\wedge\eta})$$

whose endpoint records the degree of the admitted wedge.

Finally repack the admitted wedge carrier row. The tensor concatenation row is read over the concatenated probe bundle, the scalar endpoint is produced from that tensor row, and the antisymmetry and source ledgers cite only the two input ledgers, the output probe bundle, and the displayed degree continuation. Therefore the wedge support row consumes exactly the visible arity supplied by the two input support rows and no hidden exterior-input slot. □

136.4 DiffForm degree-probe alignment

Definition 136.29 (Differential-form degree-probe alignment row)**.** *For a unary degree history d and a visible probe bundle $\mathcal{P}$, the row* $\mathsf{DegreeProbeAligned}(d, \mathcal{P})$ *is the closed structural relation generated by the following constructors:*

1. $\mathsf{DegreeProbeAligned}(\mathsf{e}, \mathsf{ProbeBundle.Bnil})$.
2. *If* $\mathsf{DegreeProbeAligned}(d, \mathcal{P})$ *holds, and if p is a tangent-probe row admitted by the manifold source, then* $\mathsf{DegreeProbeAligned}(\mathsf{E1}(d), \mathsf{ProbeBundle.Bcons}(p, \mathcal{P}))$.

Thus the unary degree row and the ProbeBundle *spine expose the same visible length. The relation is part of the displayed differential-form row: it does not import a host tangent-space dimension and does not allocate a second arity counter outside $BHist$.*

Lean4 (definition) `BEDC.Derived.DiffFormUp.DegreeProbeAligned`

Theorem 136.30 (Differential-form carrier refines to degree-probe alignment)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

be carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$*. Suppose that the tensor row τ is read over exactly the displayed bundled tangent probes in $\mathcal{P}$, with one unary degree constructor in d for each visible probe slot and with no tensor input outside the bundle. Then ω supplies* $\mathsf{DegreeProbeAligned}(d, \mathcal{P})$.

Proof. First unpack the carrier through Definition 136.1. The carrier exposes $\mathsf{UnaryHistory}(d)$, the finite ProbeBundle spine $\mathcal{P}$, and the tensor row τ over that same bundle.

Induct simultaneously on the visible probe spine and on the unary degree row selected by the tensor-input reading. In the empty bundle case, the carrier clauses and the exact-reading hypothesis leave no tensor input slot. The unary row is therefore the empty degree row, and the empty constructor of Definition 136.29 applies.

In the cons case, the carrier exposes a head tangent probe and a tail ProbeBundle. The exact-reading hypothesis strips one E1 constructor from the degree row for that head probe and leaves the tail tensor row over the tail bundle. The induction hypothesis gives alignment for the tail, and the Bcons constructor of Definition 136.29 rebuilds alignment for the whole row.

Finally apply Definition 136.26. It says that the carried row supplies the displayed degree row and the visible probe spine as the support surface consumed by the tensor, scalar, antisymmetry,

and source ledgers. Hence no hidden probe slot or hidden degree constructor remains outside the aligned row. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeAligned empty iff`

Theorem 136.31 (Differential-form degree-probe alignment transport)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and

$$\omega' = (d', \mathcal{P}', \tau', \sigma', \alpha', \lambda')$$

be rows carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$ *with* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$*. If* $\mathsf{DegreeProbeAligned}(d, \mathcal{P})$ *holds, then* $\mathsf{DegreeProbeAligned}(d', \mathcal{P}')$ *holds.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeAligned hsame transport`

Proof. Project the classifier comparison through Definition 136.2. The degree clause gives $\mathsf{hsame}(d, d')$, while the probe and tensor clauses match the displayed ProbeBundle spines componentwise through the manifold and tensor sources.

Apply Definition 136.27. The support row carried by ω transports to the support row carried by ω', with the same visible probe-slot arity.

Now apply Definition 136.12. Since the probe slot count is preserved and the unary degree row is transported by hsame, each constructor step in Definition 136.29 has a matching transported constructor step. Rebuilding the transported spine gives $\mathsf{DegreeProbeAligned}(d', \mathcal{P}')$. □

Theorem 136.32 (Differential-form wedge preserves degree-probe alignment)**.** *Let* ω *and* η *be carried* $\mathsf{DiffForm}^{\uparrow}$ *rows with* $\mathsf{DegreeProbeAligned}(d_\omega, \mathcal{P}_\omega)$ *and* $\mathsf{DegreeProbeAligned}(d_\eta, \mathcal{P}_\eta)$*. If the wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$ *is admitted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *with output degree* $d_{\omega\wedge\eta}$ *and output probe bundle* $\mathcal{P}_{\omega\wedge\eta}$*, then* $\mathsf{DegreeProbeAligned}(d_{\omega\wedge\eta}, \mathcal{P}_{\omega\wedge\eta})$ *holds.*

Proof. Expose the admitted wedge probe surface with Definition 136.28. The output bundle is the visible append

$$\mathcal{P}_{\omega\wedge\eta} = \mathsf{bundleAppend}(\mathcal{P}_\omega, \mathcal{P}_\eta),$$

and the tensor row consumes exactly that appended spine.

Induct on the left probe bundle $\mathcal{P}_\omega$. In the empty case, the left alignment row identifies d_ω with the empty unary degree, and the append surface reduces to the right input probe bundle. The Cont left-unit row in Definition 136.22 identifies the output degree with d_η, so the right input alignment supplies the output alignment.

In the cons case, the left alignment row strips one head probe from $\mathcal{P}_\omega$ and one E1 constructor from d_ω. The induction hypothesis aligns the wedge of the left tail with η. Reapplying the Bcons constructor in Definition 136.29 aligns the restored head probe with the restored unary constructor.

It remains to justify that the degree endpoint is the wedge endpoint rather than an auxiliary row. Use the continuation row

$$\mathsf{Cont}(d_\omega, d_\eta, d_{\omega\wedge\eta})$$

from Definition 136.22. Continuation associativity and the unit rows already cited by that ledger identify the inductive endpoint with the admitted wedge degree. Repacking with the visible appended bundle gives $\mathsf{DegreeProbeAligned}(d_{\omega\wedge\eta}, \mathcal{P}_{\omega\wedge\eta})$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDegreeProbeAligned bundleAppend cont`

136.5 DiffForm root-unblock obligations

Definition 136.33 (Differential-form root-unblock source surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *source* T *whose tensor rows are read over the tangent-bundle histories of* M*. The* $\mathsf{DiffForm}^\uparrow$ *root-unblock source surface is the following ten-field BEDC surface:*

1. *the ambient* BHist *history carrier for the displayed differential-form rows;*
2. *the visible* hsame *transport relation on all degree, tensor, scalar-endpoint, antisymmetry, and source-ledger histories;*
3. *the* Cont *ledger for ordered tensor concatenation and degree continuation rows;*
4. *the* ProbeBundle *spine of tangent-probe names;*
5. *the upstream* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *source for tangent probes and scalar endpoints;*
6. *the upstream* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *source for multilinear tensor rows;*
7. *the six-coordinate carrier* $\mathsf{DiffFormBHistCarrier}_{M,T}(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$ *of Definition 136.1;*
8. *the classifier* $\sim_{\mathsf{DiffForm}^\uparrow}$ *of Definition 136.2;*
9. *the wedge rows consisting only of tensor concatenation, probe-bundle append, wedge-degree bookkeeping, and the displayed antisymmetry ledger;*
10. *the exterior-input rows consumed by* $\mathsf{DeRham}^\uparrow$*,* $\mathsf{Contact}^\uparrow$*, and* $\mathsf{Symplectic}^\uparrow$*, restricted to degree, wedge, antisymmetry, scalar endpoint, and source-ledger projections.*

The surface contains no row for $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, or a standard bridge.*

Theorem 136.34 (Differential-form root-unblock carrier and classifier scope)**.** *Every carrier and classifier row readable from the Definition 136.33 is determined by the displayed BEDC source surface. The carrier is the six-coordinate row*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda),$$

and the classifier uses only degree hsame*, matched tensor and scalar comparisons, and visible* hsame *comparison of the tensor, antisymmetry, and source ledgers.*

Proof. Project the carrier through Definition 136.1. Its acceptance predicate lists only the degree row d, the probe bundle $\mathcal{P}$, the tensor row τ, the scalar endpoint σ, the antisymmetry ledger α, and the source ledger λ.

Next project the classifier through Definition 136.2. Degree comparison is hsame, matched tensor and scalar endpoints use the displayed tensor row and the retained scalar classifier from M and T, and the ledger comparisons are $\mathsf{hsame}(\alpha, \alpha')$, $\mathsf{hsame}(\tau, \tau')$, and $\mathsf{hsame}(\lambda, \lambda')$.

Finally use the carrier and classifier obligations Definition 136.4 and Definition 136.5. They exhaust the same six-coordinate carrier and the same displayed classifier clauses, so no carrier or classifier datum can be read outside the ten-field root-unblock surface. □

Theorem 136.35 (Differential-form root-unblock wedge continuation closure)**.** *The wedge input closure readable from the Definition 136.33 is exactly probe-bundle concatenation plus* Cont *ledger closure. An admitted wedge of carried rows may read the output probe spine from the append ledger and the tensor concatenation from* Cont*, together with the displayed degree and antisymmetry rows; it may not read an exterior-derivative law or a downstream geometric law from wedge admission.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootUnblock wedge cont closure`

Proof. Start with a wedge row admitted by Definition 136.15. The admitted row requires a visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation of the two tensor components, and the continuation data for that concatenation are exactly Cont ledger entries over the displayed input rows.

The probe part is then determined by Definition 136.16 and Definition 136.17. These rows identify the output probe bundle as the append of the two input $\mathsf{ProbeBundle}$ spines, with no additional probe source.

The remaining wedge-visible data are the unary degree bookkeeping of Definition 136.22 and the antisymmetry and associator ledgers bounded by Definition 136.20. Hence wedge closure is precisely the displayed probe append together with the Cont tensor ledger and the already-carried degree and antisymmetry rows. □

Theorem 136.36 (Differential-form root-unblock de Rham input consumption)**.** *Every* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, *or* $\mathsf{Symplectic}^{\uparrow}$ *consumer of the Definition 136.33 may read only the degree row, wedge rows, antisymmetry ledger, scalar endpoint row, and source-ledger rows exposed there. In particular, de Rham exterior input may consume the degree-shift and wedge-input surfaces, while contact and symplectic consumers must supply their nonintegrability, closedness, and nondegeneracy laws in their own certificate surfaces.*

Proof. First read the de Rham boundary from Definition 136.24. It permits an exterior-derivative row to depend on the probe bundle, degree row, tensor row, scalar endpoint, and source ledger, and on the wedge probe and degree ledgers. It explicitly gives no $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability row.

Next combine the exterior-input rows Definition 136.57, Definition 136.58, and Definition 136.56. They allow only one-step degree continuation, carried input and output rows, wedge-input transport, scalar endpoint transport, and the restricted source ledger.

Finally apply the downstream boundary rows Definition 136.61 and Definition 136.74. These rows identify $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, and $\mathsf{Symplectic}^{\uparrow}$ as consumers of the displayed DiffForm carrier, wedge, scalar, antisymmetry, and source data only. The operator and geometric laws therefore remain obligations of those downstream chapters, not projections from the DiffForm root-unblock surface. □

Definition 136.37 (DiffForm root obligation source)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *root obligation source is the visible tuple consisting of the BHist carrier rows of Definition 136.1, the classifier rows of Definition 136.2, the source certificates* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*, the wedge ledger rows of Definition 136.15, the antisymmetry chain ledgers of Definition 136.77, and the exterior-derivative input rows named by Definition 136.66. It exposes only those rows as the root surface for downstream consumers.*

Theorem 136.38 (DiffForm root source transport obligation)**.** *For the root obligation source of Definition 136.37, replacing every displayed carrier and classifier history by an* hsame*-equal history preserves the accepted* $\mathsf{DiffForm}^{\uparrow}$ *source surface. In particular, the carrier and classifier parts of the tuple transport without changing the retained* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *or* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *sources.*

Proof. Start with the carrier component of the root tuple. Apply Definition 136.10 to each displayed carried row; the retained manifold and tensor-product certificates are part of the same row, so no separate source replacement is introduced.

Next transport the classifier component. Use Definition 136.11 on the transported endpoints of each classifier row. The degree, tensor, scalar, antisymmetry, and source-ledger clauses are exactly the clauses exposed in Definition 136.2.

Repack the transported carrier rows and transported classifier rows with the unchanged source certificates. This is precisely the accepted source surface specified by Definition 136.37. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootSource transport obligation`

Theorem 136.39 (DiffForm root wedge and antisymmetry obligation)**.** *The root obligation source of Definition 136.37 closes the visible wedge and antisymmetry obligations: every admitted wedge row carries the required tensor concatenation ledger, its probe concatenation classifier is stable, its associator row remains in the carried surface, and its antisymmetry chain is classified by the displayed chain ledger.*

Proof. First read the binary wedge row through Definition 136.15. This supplies the visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation row and transports the two input form rows through the retained carrier and classifier surfaces.

The probe part of the wedge is then controlled by Definition 136.18. It identifies the output probe bundle after transporting the two input bundles and keeps the wedge output classified under $\sim_{\mathsf{DiffForm}^{\uparrow}}$.

For three carried rows, apply Definition 136.20. The two parenthesized wedge surfaces are carried by the same visible associator ledger, so the root tuple contains the needed associativity comparison.

Finally use Definition 136.77. The antisymmetry chain rows are part of the root source, and their classifier stability supplies the visible chain obligation for every transported wedge surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootWedgeAntisymmetry obligation`

Theorem 136.40 (DiffForm root exterior-derivative boundary obligation)**.** *The exterior-derivative component of the root obligation source records only the admitted input and output rows for the exported derivative boundary. It does not assert closedness, nilpotence, or a de Rham quotient; it binds the candidate derivative row to the visible input/output ledger and to the wedge boundary row consumed by downstream wedge formulas.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootExteriorDerivative boundary obligation`

Proof. Begin with Definition 136.66. It states that the exported derivative surface is exactly the displayed input/output ledger row, so the root tuple records no additional derivative law.

Transport the candidate derivative classifier by Definition 136.67. This keeps the candidate row attached to the same carrier and classifier surface under hsame replacement of the displayed histories.

The wedge-side boundary is then read from Definition 136.80. That theorem supplies the only boundary row consumed by wedge formulas, and the root surface merely keeps that row available beside the input/output ledger. □

Theorem 136.41 (DiffForm root NameCert surface)**.** *The root obligation source of Definition 136.37 presents a single* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *surface: source transport, wedge and antisymmetry transport, and exterior-derivative boundary exposure assemble into the displayed* $\mathsf{DiffForm}^{\uparrow}$ *obligation surface.*

Proof. Use Definition 136.38 to bind the carrier and classifier rows to the retained $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ sources. This gives the source part of the root certificate.

Use Definition 136.39 for the wedge ledger, probe concatenation, associator, and antisymmetry chain fields. These are exactly the visible algebraic rows required by the differential-form certificate surface.

Use Definition 136.40 for the exported derivative boundary. The derivative row is retained only as an input/output and wedge-boundary surface.

Finally apply Definition 136.165. Its certificate surface contains the same source, wedge, antisymmetry, and exterior-derivative rows, so the three root obligations repack as the displayed $\mathsf{NameCert}_{\mathsf{DiffForm}^\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootNameCert surface`

Definition 136.42 (Differential-form obligation inventory)**.** *The paper-side obligation inventory for* $\mathsf{DiffForm}^\uparrow$ *is the following ten-field roadmap:*

1. *the traditional target is the certificate surface for antisymmetric multilinear forms on tangent probes;*
2. *the BEDC sources are* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$*;*
3. *the carrier is* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *from Definition 136.1;*
4. *the classifier is* $\sim_{\mathsf{DiffForm}^\uparrow}$ *from Definition 136.2;*
5. *carrier stability is the* hsame *transport row of Definition 136.10;*
6. *classifier stability is Definition 136.11;*
7. *the ledger rows are exactly the degree, probe, tensor, scalar, antisymmetry, and source-provenance rows;*
8. *the core theorem list consists of carrier exhaustion, classifier reuse, ledger exactness, transport stability, wedge coverage, and the exterior-input boundary theorems already displayed in this chapter;*
9. *exactness is row exhaustion: every admitted downstream use must factor through one of the displayed rows;*
10. *no standard bridge is supplied here, so the inventory is the displayed paper-side obligation surface and not a bridge certificate.*

Theorem 136.43 (Differential-form carrier support obligation)**.** *The carrier support promised by the* $\mathsf{DiffForm}^\uparrow$ *obligation inventory is exactly the six-coordinate surface*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

of Definition 136.1. It is therefore exhausted by the degree row, probe-bundle row, tensor row, scalar endpoint row, antisymmetry ledger, and source-provenance ledger, as projected by Definition 136.4.

Proof. Unfold Definition 136.42. Its carrier field points to $\mathsf{DiffFormBHistCarrier}_{M,T}$ and introduces no second carrier predicate.

Apply Definition 136.1. A carried row is displayed as the tuple $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, with one acceptance clause for each of its six coordinates.

Now apply Definition 136.4. That theorem states that carrier acceptance adds no seventh coordinate and no hidden source outside the displayed histories. The support field of the inventory is therefore exactly the six-coordinate carrier surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistCarrier support obligation`
Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormCarrier support obligation`

Theorem 136.44 (Differential-form classifier transport obligation)**.** *The classifier obligation recorded by Definition 136.42 consists of reflexivity, symmetry, transitivity, carrier transport, and classifier stability for the displayed* $\mathsf{DiffForm}^{\uparrow}$ *rows. These obligations are projections of Definition 136.2, Definition 136.5, Definition 136.10, and Definition 136.11.*

Proof. For reflexivity, read each classifier component in Definition 136.2 against itself. The degree, tensor, antisymmetry, and source-ledger clauses are reflexive hsame rows, and the scalar endpoint clause uses the retained source classifier on the same endpoint row.

For symmetry and transitivity, compose the same displayed clauses componentwise. The hsame clauses supply the ledger-side comparisons, while the scalar endpoint comparison stays inside the upstream scalar classifier named in the definition.

It remains to identify the transport obligations. Carrier transport is exactly Definition 136.10. Once transported rows are accepted, classifier stability is exactly Definition 136.11. Thus the inventory records no classifier law beyond the existing displayed classifier and transport theorems. □

Theorem 136.45 (Differential-form exterior input boundary obligation)**.** *Downstream* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *consumers of* $\mathsf{DiffForm}^{\uparrow}$ *may use only the listed* $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, *wedge-ledger, and exterior-input rows. In particular, their access is bounded by Definition 136.24 and Definition 136.61.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorInputBoundary obligation`

Proof. First unfold the source and carrier entries of Definition 136.42. They name only $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, the carried $BHist$ rows, $\mathsf{ProbeBundle}$, hsame, and the Cont ledgers attached to tensor concatenation.

Next apply Definition 136.24. It permits exterior-derivative consumers to read the probe bundle, degree row, tensor row, scalar endpoint classifier, antisymmetry ledger, wedge coverage, wedge probe concatenation ledger, and wedge degree ledger, and it excludes $d^2 = 0$, closedness, nondegeneracy, and contact nonintegrability from the $\mathsf{DiffForm}^{\uparrow}$ source.

Finally apply Definition 136.61. The downstream scope theorem adds the exterior-derivative candidate ledger and confirms that the three named consumer chapters must state any operator or geometric laws inside their own surfaces. The obligation boundary is therefore exactly the listed rows and no broader import surface. □

Definition 136.46 (Differential-form zero-degree scalar boundary)**.** *Fix* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *sources* M, T *as in Definition 136.1. A differential-form zero-degree scalar boundary row is a carried* $\mathsf{DiffForm}^{\uparrow}$ *row*

$$\omega_0 = (d_0, \mathcal{P}_0, \tau_0, \sigma, \alpha_0, \lambda_0)$$

accepted by $\mathsf{DiffFormBHistCarrier}_{M,T}$ *with the following visible coordinates:*

1. d_0 *is the empty unary-zero history, so* $\mathsf{hsame}(d_0, \mathsf{e})$ *and* $\mathsf{UnaryHistory}(d_0)$.
2. $\mathcal{P}_0$ *is the empty probe spine* $\mathsf{ProbeBundle.Bnil}$.
3. τ_0 *is the empty tensor-unit surface of the* $\mathsf{TensorProduct}^{\uparrow}$ *source over* $\mathcal{P}_0$.
4. α_0 *is the empty antisymmetry ledger, with no adjacent-swap row and no cross-permutation row.*

5. *σ is the retained scalar classifier endpoint row supplied by the upstream manifold and tensor sources.*

6. *λ_0 records exactly $M, T, d_0, \mathcal{P}_0, \tau_0, \sigma$, and α_0 as the source ledger for this boundary row.*

Thus the row is a scalar-valued 0-form surface inside the displayed $\mathsf{DiffForm}^{\uparrow}$ carrier, not a host-level exception outside the carrier and classifier of Definition 136.1 and Definition 136.2.

Theorem 136.47 (Differential-form zero-degree empty-probe exhaustion)**.** *Let ω_0 be a differential-form zero-degree scalar boundary row of Definition 136.46. For every probe row p, the membership claim $\mathsf{InBundle}(p, \mathcal{P}_0)$ is impossible. Equivalently, every downstream use of ω_0 sees an empty probe spine, with no hidden tangent input slot.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree empty probe exhaustion`

Proof. Unpack the boundary row through Definition 136.46. Its probe coordinate is the closed constructor ProbeBundle.Bnil.

Invert an assumed $\mathsf{InBundle}(p, \mathcal{P}_0)$ claim. The ProbeBundle membership constructors can only expose a head probe from a nonempty spine or recurse into a nonempty tail. The constructor ProbeBundle.Bnil has neither, so no membership constructor can produce the assumed claim.

The same empty-spine inversion is compatible with the carrier and classifier rows of Definition 136.1 and Definition 136.2. The scalar endpoint remains present as σ, but no probe row can be read from the boundary. □

Theorem 136.48 (Differential-form zero-degree scalar classifier readback)**.** *Let*

$$\omega_0 = (d_0, \mathcal{P}_0, \tau_0, \sigma, \alpha_0, \lambda_0)$$

and

$$\omega_0' = (d_0', \mathcal{P}_0', \tau_0', \sigma', \alpha_0', \lambda_0')$$

be two differential-form zero-degree scalar boundary rows over the same M, T sources. Then $\omega_0 \sim_{\mathsf{DiffForm}^{\uparrow}} \omega_0'$ is exactly the retained scalar endpoint classifier comparison $\sigma \sim_R \sigma'$, together with the visible source-ledger comparisons

$$\mathsf{hsame}(d_0, d_0'), \quad \mathsf{hsame}(\tau_0, \tau_0'), \quad \mathsf{hsame}(\alpha_0, \alpha_0'), \quad \mathsf{hsame}(\lambda_0, \lambda_0').$$

No probe-indexed classifier clause remains after empty ProbeBundle.Bnil *inversion.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree scalar classifier readback`

Proof. Read the two boundary rows through Definition 136.46. Both probe spines are ProbeBundle.Bnil, both degree rows are the empty unary-zero history, both tensor rows are empty tensor-unit surfaces, and both antisymmetry ledgers are empty.

Now unfold the differential-form classifier of Definition 136.2. The degree, tensor, antisymmetry, and source-ledger clauses are exactly the displayed hsame comparisons on the boundary rows. The scalar endpoint clause is the retained source classifier comparison $\sigma \sim_R \sigma'$.

It remains to rule out any further probe-indexed clause. By Definition 136.47, no InBundle claim exists for either boundary spine. Hence every matched multilinear probe row in the classifier is exhausted by the empty tensor-unit endpoint, and the public readback surface is precisely the scalar endpoint classifier plus the visible hsame ledger comparisons. □

Theorem 136.49 (Differential-form zero-degree wedge Cont unit boundary)**.** *Let ω_0 be a differential-form zero-degree scalar boundary row and let*

$$\eta = (d_\eta, \mathcal{P}_\eta, \tau_\eta, \rho, \beta, \mu)$$

be any carried $\mathsf{DiffForm}^\uparrow$ *row over the same M, T sources. Whenever the wedge rows $\omega_0 \wedge_{\mathsf{DiffForm}^\uparrow} \eta$ and $\eta \wedge_{\mathsf{DiffForm}^\uparrow} \omega_0$ are admitted by Definition 136.15, their probe and degree surfaces are the nonempty side's surfaces:*

$$\mathsf{bundleAppend}(\mathsf{ProbeBundle.Bnil}, \mathcal{P}_\eta) \quad \textit{and} \quad \mathsf{bundleAppend}(\mathcal{P}_\eta, \mathsf{ProbeBundle.Bnil})$$

are classified with $\mathcal{P}_\eta$, and the degree ledgers of Definition 136.22 reduce through the Cont *left and right unit rows to d_η. The tensor-unit component of ω_0 therefore contributes no additional probe slot and no additional degree history to either wedge boundary.*

Proof. First expose the wedge probe ledger. Apply Definition 136.16 and Definition 136.15. The output probe spine of the left wedge is $\mathsf{bundleAppend}(\mathsf{ProbeBundle.Bnil}, \mathcal{P}_\eta)$, and the output probe spine of the right wedge is $\mathsf{bundleAppend}(\mathcal{P}_\eta, \mathsf{ProbeBundle.Bnil})$.

For the left wedge, invert the empty left spine. The bundleAppend empty-left law identifies the output spine with $\mathcal{P}_\eta$, and Definition 136.47 rules out an additional probe contributed by ω_0.

For the right wedge, induct on $\mathcal{P}_\eta$. The empty case is the same ProbeBundle.Bnil endpoint. In the cons case, the head probe is preserved and the induction hypothesis identifies the appended tail with the original tail, so the empty-right law identifies the output spine with $\mathcal{P}_\eta$.

It remains to read the degree surface. By Definition 136.22, both admitted wedges carry visible continuation rows for their output degrees. Since the boundary degree of ω_0 is e, the left wedge uses the Cont left-unit row $\mathsf{Cont}(\mathsf{e}, d_\eta, d_\eta)$, and the right wedge uses the Cont right-unit row $\mathsf{Cont}(d_\eta, \mathsf{e}, d_\eta)$. Thus the wedge output degree is classified with d_η on both sides.

Finally, the tensor component of ω_0 is the empty tensor-unit surface from Definition 136.46. The tensor concatenation ledger named by Definition 136.15 therefore records only the unit-row concatenation with τ_η, while the scalar endpoint comparison remains in the retained source classifier described by Definition 136.2. Repacking these probe, degree, tensor, and endpoint rows gives the claimed unit boundary for both wedge orders. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree wedge cont unit boundary`

Definition 136.50 (Differential-form zero-degree consumer-neutrality surface)**.** *Fix* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *sources M, T as in Definition 136.1. The differential-form zero-degree consumer-neutrality surface is the zero-degree scalar boundary row of Definition 136.46, read together with the shared rows visible to every downstream consumer:*

1. *the carried BHist row, its* hsame *ledger, and the displayed* Cont *unit rows on the empty degree history;*
2. *the empty probe spine* ProbeBundle.Bnil *and its absence of probe membership rows;*
3. *the* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *source ledgers supplying the manifold and tensor endpoints;*
4. *the retained scalar endpoint classifier row;*
5. *the empty antisymmetry ledger and the source ledger rows recorded by the boundary coordinate λ_0.*

This surface is a shared root-face package only. It contains the data needed to reuse the scalar 0-form boundary, but it does not add a consumer-specific law row.

Lean4 (definition) `BEDC.Derived.DiffFormUp.DiffFormZeroDegreeConsumerNeutralitySurface`

Theorem 136.51 (Differential-form zero-degree shared consumer face)**.** *The zero-degree consumer-neutrality surface of Definition 136.50 lies wholly in the shared root face consumed by* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$.

Proof. Unpack the surface through Definition 136.50. Its probe coordinate is ProbeBundle.Bnil, so Definition 136.47 removes every probe-indexed membership row from the exported data.

The remaining classifier information is exactly the retained scalar endpoint and the visible source-ledger comparisons by Definition 136.48. These rows are carried $BHist$, classifier, scalar endpoint, empty antisymmetry, wedge-unit, and source-ledger rows.

Apply Definition 136.117. The shared root face is precisely the face that may be consumed by the three downstream chapters without assigning a row to a consumer-specific law. Hence the whole zero-degree surface lies in that shared face. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree shared consumer face`

Theorem 136.52 (Differential-form zero-degree consumer law separation)**.** *The zero-degree consumer-neutrality surface exports no* $d^2 = 0$ *row, no closedness row, no nondegeneracy row, and no contact nonintegrability row.*

Proof. By Definition 136.51, the surface lies inside the shared root face. Apply Definition 136.118: nilpotence is owned by $\mathsf{DeRham}^{\uparrow}$, closedness and nondegeneracy by $\mathsf{Symplectic}^{\uparrow}$, and contact nonintegrability by $\mathsf{Contact}^{\uparrow}$.

It remains to check that the wedge contribution does not smuggle in one of those laws. The wedge row available at zero degree is only the Cont unit boundary of Definition 136.49. That theorem identifies the empty probe and empty degree contribution of the scalar endpoint; it supplies no exterior-derivative square, no symplectic closedness or nondegeneracy clause, and no contact nonintegrability clause.

Therefore every law row named in the statement remains outside the exported zero-degree surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree consumer law separation`

Theorem 136.53 (Differential-form zero-degree deps-ready consumer seal)**.** *The definition Definition 136.50, the shared consumer-face theorem Definition 136.51, and the consumer law separation theorem Definition 136.52 form the citable zero-degree shared package for the* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *consumers.*

Proof. The definition names the exact rows carried by the scalar 0-form boundary. The shared consumer-face theorem places those rows in the common root face seen by the three downstream chapters.

The consumer law separation theorem records the complementary exclusion: nilpotence, closedness, nondegeneracy, and contact nonintegrability are not part of the zero-degree root package. Thus downstream consumers may cite this package for the shared zero-degree boundary while keeping their own laws separate. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormZeroDegree deps ready consumer seal`

Definition 136.54 (Differential-form exterior-derivative candidate ledger)**.** *For a carried* $\mathsf{DiffForm}^{\uparrow}$ *row* ω *and a displayed candidate output* $\partial\omega$*, the visible package* $\mathsf{DiffFormExteriorDerivativeLedger}_{M,T}(\omega, \partial\omega)$ *is the BHist ledger whose accepted rows are exactly the input carrier row for* ω*, the output carrier row for* $\partial\omega$*, the unary degree continuation row from* d_ω *to the displayed successor degree* $d_{\partial\omega}$*, the transported probe-bundle endpoint row, the transported tensor endpoint row, and the transported scalar endpoint row. Its provenance is restricted to* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*

Lean4 (definition) `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeCandidateLedger`

Theorem 136.55 (Differential-form exterior-derivative degree-raise ledger)**.** *Any row accepted by* $\mathsf{DiffFormExteriorDerivativeLedger}_{M,T}(\omega, \partial\omega)$ *exposes the input degree row* d_ω*, the output degree row* $d_{\partial\omega}$*, and the unary continuation row*

$$\mathsf{Cont}(d_\omega, d_{\partial\omega})$$

connecting the input degree to the displayed successor degree. This is only a degree-raising ledger for the candidate exterior-derivative row; it does not assert $d^2 = 0$*.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger degree raise`

Proof. Unfold Definition 136.54. The accepted package contains the input $\mathsf{DiffForm}^{\uparrow}$ carrier row for ω and the output $\mathsf{DiffForm}^{\uparrow}$ carrier row for $\partial\omega$, so projecting those two rows exposes d_ω and $d_{\partial\omega}$ through Definition 136.1.

The same definition requires the displayed unary continuation row from d_ω to $d_{\partial\omega}$. Repack that row with the two projected carrier degree rows as the visible unary-history provenance of the candidate exterior-derivative ledger.

No nilpotence row is present in the accepted package. The construction records only the successor-degree continuation, so an assertion of $d^2 = 0$ must be supplied by a downstream $\mathsf{DeRham}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivative degree raise ledger`

Theorem 136.56 (Differential-form exterior-derivative wedge-input stability)**.** *Let* $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$*, and suppose* $\mathsf{DiffFormExteriorDerivativeLedger}_{M,T}(\omega, \partial\omega)$ *accepts a candidate exterior-derivative ledger from* ω *to* $\partial\omega$*. Componentwise transport along the* $\mathsf{DiffForm}^{\uparrow}$ *classifier carries this data to a candidate ledger* $\mathsf{DiffFormExteriorDerivativeLedger}_{M,T}(\omega', \partial\omega')$ *whose output row is classified with* $\partial\omega$ *through the transported probe-bundle, tensor, scalar endpoint, antisymmetry, wedge, and degree rows.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivative wedge input stability`

Proof. Start with the classifier comparison $\omega \sim_{\mathsf{DiffForm}^{\uparrow}} \omega'$. By Definition 136.11, the carrier components of ω transport to the carrier components of ω' while preserving the visible probe-bundle, tensor, scalar endpoint, antisymmetry, and source-ledger comparisons.

Apply Definition 136.23 to the wedge and degree bookkeeping exposed at the boundary. The unary degree row used by the candidate ledger moves to the corresponding degree row for ω', and the successor continuation remains a visible Cont row.

For the scalar endpoint, use Definition 136.64. Together with the transported probe-bundle and tensor endpoint rows, this repacks the accepted rows of Definition 136.54 as a candidate ledger from ω' to an output row $\partial\omega'$ classified with $\partial\omega$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger classifier transport`
Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger hsame transport degree raise`

Definition 136.57 (Differential-form exterior-derivative input row)**.** *An accepted* $\mathsf{DiffForm}^{\uparrow}$ *input row for a downstream exterior-derivative candidate is the visible BEDC package*

$$(\omega, \eta, d, d^{+}, \kappa, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

where ω *is the source carried form row,* η *is the target carried form row,* d *is the source unary degree row,* d^{+} *is the target unary degree row, and* κ *is the one-step continuation row*

$$\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^{+}).$$

The remaining entries are the carried ProbeBundle *row* $\mathcal{P}$*, multilinear* $\mathsf{TensorProduct}^{\uparrow}$ *tensor row* τ*, retained scalar endpoint row* σ*, antisymmetry ledger* α*, and source ledger* λ *exposed by Definition 136.24. This input row is only admissible data for an exterior-derivative candidate; it carries no claim that* $d^2 = 0$*.*

Lean4 (definition) `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeInputRow`

Theorem 136.58 (Differential-form exterior-derivative degree-shift boundary)**.** *Every accepted input row of Definition 136.57 exposes exactly the source degree* d*, the target degree* d^{+}*, and the one-step continuation ledger* $\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^{+})$ *as its degree-raising boundary. The boundary may also cite the wedge degree ledger of Definition 136.22 when the same source participates in a wedge row, but it asserts no exterior-derivative identity.*

Proof. Unpack the accepted input row. Its displayed degree entries are d, d^{+}, and the continuation row κ, so the degree-raising boundary is already visible as BHist data.

The source form row is one of the carried rows bounded by Definition 136.24. Thus a downstream consumer can read its unary degree row, probe bundle, tensor row, scalar endpoint, antisymmetry ledger, and source ledger without importing any operator law.

When the same form is used in a wedge row, the compatible degree bookkeeping is the ledger of Definition 136.22. That ledger supplies only degree provenance for the wedge, so repacking it with the one-step row still leaves $d^2 = 0$ outside the $\mathsf{DiffForm}^{\uparrow}$ boundary. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeInputRow degree shift boundary`

Theorem 136.59 (Differential-form exterior-derivative scalar-transport boundary)**.** *Let* R *and* R' *be accepted input rows of Definition 136.57. If their source form rows are related componentwise by* hsame *on probe bundles, tensor rows, antisymmetry ledgers, and source ledgers, and if the retained scalar classifier relates their scalar endpoint rows, then the accepted scalar endpoint part of* R *transports to the accepted scalar endpoint part of* R'*.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivative scalar transport boundary`

Proof. First transport the source carried form. The componentwise hsame hypotheses are exactly the carrier entries used by Definition 136.10, so the transported source form row remains accepted by the $\mathsf{DiffForm}^{\uparrow}$ carrier.

Next preserve the classifier row. By Definition 136.11, the scalar endpoint clause in the form classifier is stable under the same componentwise transport.

Finally apply the scalar endpoint boundary itself. The scalar part of an accepted input row is the retained scalar classifier entry bounded by Definition 136.64. Therefore the scalar endpoint accepted for R transports to the corresponding accepted scalar endpoint for R' without adding a new exterior-derivative law. □

Theorem 136.60 (Differential-form exterior-derivative two-step continuation ledger)**.** *Let $R_{\omega\eta}$ and $R_{\eta\theta}$ be accepted exterior-derivative input rows of Definition 136.57. Suppose $R_{\omega\eta}$ has source row ω, target row η, source degree d_ω, and target degree d_η, while $R_{\eta\theta}$ has source row η, target row θ, source degree d_η, and target degree d_θ. Then the shared middle row exposes the two-step degree continuation row*

$$\mathsf{Cont}(d_\omega, \mathsf{E1}(\mathsf{E1}(\mathsf{e})), d_\theta).$$

This composite is only a degree-continuation ledger from ω through η to θ; it does not assert that the second exterior derivative vanishes.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger degree continuation composite`

Proof. Apply Definition 136.58 to $R_{\omega\eta}$. Its degree boundary supplies

$$\mathsf{Cont}(d_\omega, \mathsf{E1}(\mathsf{e}), d_\eta).$$

Apply the same boundary theorem to $R_{\eta\theta}$. Since the target row of the first accepted input is the source row of the second, the middle degree row is the same displayed d_η, and the second boundary supplies

$$\mathsf{Cont}(d_\eta, \mathsf{E1}(\mathsf{e}), d_\theta).$$

By Definition 21.31, the two continuation ledgers compose through the shared middle degree. It remains only to compute the exposed tail. The constructor clause in the generated continuation relation of Definition 21.7 gives

$$\mathsf{Cont}(\mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{e}), \mathsf{E1}(\mathsf{E1}(\mathsf{e}))).$$

Substituting this tail in the composite-tail equivalence yields the displayed row

$$\mathsf{Cont}(d_\omega, \mathsf{E1}(\mathsf{E1}(\mathsf{e})), d_\theta).$$

All rows used in the construction are the two accepted input rows and the kernel continuation composition rule. The conclusion records only that two successor-degree continuations are composable as BHist data, so it leaves any $d^2 = 0$ assertion to the downstream $\mathsf{DeRham}^\uparrow$ surface. □

Theorem 136.61 (Differential-form exterior-derivative downstream scope)**.** *Downstream exterior-derivative chapters may use a* $\mathsf{DiffForm}^\uparrow$ *row only through its degree row, its carried* ProbeBundle *of tangent probes, its multilinear* $\mathsf{TensorProduct}^\uparrow$ *tensor row, its scalar endpoint classifier, its anti-symmetry ledger, wedge ledger coverage, the wedge associator ledger, the wedge degree ledger, and the exterior-derivative candidate ledger of Definition 136.54. In particular, the DiffForm surface supplies the carrier, classifier, exterior-algebra substrate, degree-raise candidate row, and endpoint transport substrate for* $\mathsf{DeRham}^\uparrow$, $\mathsf{Symplectic}^\uparrow$, *and* $\mathsf{Contact}^\uparrow$, *but it does not by itself assert $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivative downstream scope`

Proof. Inspect the exported fields of Definition 136.165. Each accessible component is one of the displayed carrier, classifier, stability, wedge, or ledger rows.

The exterior derivative needs the degree row and the multilinear scalar endpoint surface in order to state degree raising and scalar endpoint transport. Those rows are available from the carrier and classifier; the wedge degree ledger supplies the visible input-degree bookkeeping allowed at the de Rham boundary. The candidate ledger of Definition 136.54 supplies the positive output-carrier

and successor-degree row allowed at that boundary. Stability is supplied by the two transport theorems, Definition 136.23, and Definition 136.56.

The identities $d^2 = 0$, closedness, nondegeneracy, and contact nonintegrability require additional operator or geometric rows. They are not fields of the $\mathsf{DiffForm}^{\uparrow}$ surface, so downstream chapters must state and certify them in their own $\mathsf{NameCert}$ surfaces. □

Theorem 136.62 (Differential-form source scope)**.** *Every row accepted by* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *depends only on* $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$. *No accepted row may import a carrier component, classifier clause, or endpoint comparison from outside those six displayed sources.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistCarrier source scope`

Proof. Project the accepted row through Definition 136.1. Its six components are exactly the degree history, probe bundle, tensor row, scalar endpoint, antisymmetry ledger, and source ledger. The only structural rows mentioned there are $BHist$, $\mathsf{ProbeBundle}$, the $\mathsf{TensorProduct}^{\uparrow}$ continuation ledgers Cont, and the two upstream $\mathsf{NameCert}$ sources.

Now read the classifier clauses in Definition 136.2. Degree, tensor, antisymmetry, and source-ledger comparisons are hsame comparisons on visible histories, while scalar endpoint comparison is the retained scalar classifier supplied by the manifold and tensor sources. Thus every accepted classifier step remains inside the stated source scope. □

Theorem 136.63 (Differential-form wedge degree ledger)**.** *For a carried wedge row* $\omega \wedge_{\mathsf{DiffForm}^{\uparrow}} \eta$, *the visible BHist provenance exposes the degree row of* ω, *the degree row of* η, *the* $\mathsf{TensorProduct}^{\uparrow}$ *concatenation ledger supplied by Definition 136.15, and the resulting unary degree row of the wedge. This ledger states provenance for the wedge degree and does not assert any exterior-derivative identity.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDegreeLedger visible source rows`

Proof. Unpack the wedge row by Definition 136.15. Admission of the wedge requires a visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation row joining the tensor component of the left carried form to the tensor component of the right carried form.

Read the two input degree rows from Definition 136.1. They are the first carrier components of ω and η, and the wedge row records them in its source ledger together with the tensor concatenation row.

Repack those displayed histories as the wedge degree ledger. The output degree row is a unary history visible in the carried wedge row, so downstream chapters may cite the two input degree rows, the concatenation provenance, and the output degree row without adding any law about d. □

Theorem 136.64 (Differential-form scalar endpoint transport)**.** *Let* ω *and* ω' *be* $\mathsf{DiffForm}^{\uparrow}$ *rows with componentwise* hsame *comparisons on their probe bundles, tensor rows, antisymmetry ledgers, and source ledgers. If the scalar endpoint of* ω *is accepted by the retained scalar classifier from the upstream manifold and tensor sources, then the scalar endpoint comparison transports to* ω' *through the* $\mathsf{DiffForm}^{\uparrow}$ *classifier.*

Proof. First transport the carrier row. The componentwise comparisons on the probe bundle, tensor row, antisymmetry ledger, and source ledger are exactly the visible histories required by Definition 136.10, so the transported row is again accepted by $\mathsf{DiffFormBHistCarrier}_{M,T}$.

Next apply classifier stability. By Definition 136.11, the scalar endpoint clause of Definition 136.2 is preserved under the same componentwise transport.

Finally, the scalar comparison used in that clause is not a new $\mathsf{DiffForm}^{\uparrow}$ primitive. It is the retained scalar classifier from the manifold and tensor sources, so endpoint comparison moves along the transported probe, tensor, antisymmetry, and source ledgers already present in the carried row. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBHistCarrier scalar endpoint transport`

Definition 136.65 (Differential-form exterior-derivative candidate row)**.** *Fix carried rows ω and η of* $\mathsf{DiffFormBHistCarrier}_{M,T}$*. A differential-form exterior-derivative candidate row $D(\omega, \eta)$ is an accepted visible BHist package consisting exactly of the following data:*

1. *the input carrier row ω and the output carrier row η;*
2. *the unary input degree row d_ω, the unary output degree row d_η, and the displayed one-step degree-raise ledger* $\mathsf{Cont}(d_\omega, \mathsf{one}, d_\eta)$*;*
3. *the transported tangent-probe bundle ledger from ω to η;*
4. *the transported* $\mathsf{TensorProduct}^{\uparrow}$ *multilinear row ledger from the input tensor row to the output tensor row;*
5. *the transported scalar endpoint ledger classified by the retained scalar classifier of the manifold and tensor sources;*
6. *a source ledger naming only $BHist$,* hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *as admissible sources for the displayed input-output package.*

This row is only the input-output surface for an exterior derivative supplied by a downstream certificate. It does not assert $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or any second-derivative identity.

Theorem 136.66 (Differential-form exterior-derivative input-output ledger exactness)**.** *Every accepted $D(\omega, \eta)$ exposes exactly the input form row, the output form row, the degree-raise* Cont *ledger, the probe transport ledger, the* $\mathsf{TensorProduct}^{\uparrow}$ *transport ledger, the scalar endpoint transport ledger, and the source ledger listed in Definition 136.65. No closedness, nondegeneracy, contact condition, or second-derivative identity can be projected from such a row.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeInputRow ledger exactness`
Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger input output exactness`

Proof. Unfold Definition 136.65. The first two displayed fields are carried $\mathsf{DiffFormBHistCarrier}_{M,T}$ rows, and their carrier status is the same visible status transported by Definition 136.10.

The output classifier is read only through Definition 136.11. The scalar endpoint entry is one of the displayed transport ledgers, so its admissible comparison is exactly the scalar endpoint transport of Definition 136.64.

The remaining provenance is the source ledger. By Definition 136.62, that ledger is restricted to $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$. These are the only fields displayed by the candidate row, so exhausting the list gives the claimed input-output ledger surface and leaves the excluded geometric and operator identities unavailable. □

Theorem 136.67 (Differential-form exterior-derivative candidate classifier transport)**.** *Let $D(\omega, \eta)$ be an accepted exterior-derivative candidate row. Suppose ω' is related to ω by $\sim_{\mathsf{DiffForm}^{\uparrow}}$, and suppose every displayed degree, probe, tensor, scalar, and source ledger of $D(\omega, \eta)$ is transported componentwise by* hsame *and by the retained source classifiers named in the candidate source ledger. Then the transported output row η' is related to η by $\sim_{\mathsf{DiffForm}^{\uparrow}}$, and the transported package $D(\omega', \eta')$ is again an accepted exterior-derivative candidate row.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger candidate classifier transport`

Proof. Transport the input and output carrier rows through Definition 136.10. The hypothesis gives the componentwise hsame comparisons for the displayed degree, probe, tensor, scalar, and source ledgers, and the source ledger permits only the retained manifold and tensor classifiers for non-hsame endpoint comparisons.

The scalar endpoint clause is transported by Definition 136.64. The output form classifier then follows from Definition 136.11, applied to the transported degree, tensor, antisymmetry, scalar endpoint, and source ledger fields of η and η'.

It remains to rebuild the candidate row. Repack the transported input carrier, transported output carrier, transported one-step degree-raise Cont ledger, transported probe ledger, transported tensor ledger, transported scalar endpoint ledger, and the same restricted source ledger. This is exactly the field list of Definition 136.65, so $D(\omega', \eta')$ is accepted. □

Definition 136.68 (Differential-form exterior-derivative visible input package)**.** *Let*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

be a source row carried by $\mathsf{DiffFormBHistCarrier}_{M,T}$*, and let*

$$D\omega = (d^+, \mathcal{P}^+, \tau^+, \sigma^+, \alpha^+, \lambda^+)$$

be a carried output candidate. The package $\mathsf{DiffFormExteriorInput}_{M,T}(\omega, D\omega)$ *is accepted exactly when its visible provenance consists of the two displayed* $\mathsf{DiffForm}^{\uparrow}$ *carrier rows, the unary degree rows* $\mathsf{UnaryHistory}(d)$ *and* $\mathsf{UnaryHistory}(d^+)$*, the degree-raising continuation row*

$$\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^+),$$

the transported ProbeBundle *rows connecting $\mathcal{P}$ with $\mathcal{P}^+$, the retained* $\mathsf{TensorProduct}^{\uparrow}$ *tensor row for the multilinear output candidate, the retained scalar classifier comparison between the source and output scalar endpoints, and the operator-owned source ledger recording that these rows are the only inputs supplied to the exterior-derivative construction. The package asserts no row for $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability.*

Lean4 (definition) `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeVisibleInputPackage`

Theorem 136.69 (Differential-form exterior-derivative degree successor ledger)**.** *Every package accepted by* $\mathsf{DiffFormExteriorInput}_{M,T}(\omega, D\omega)$ *exposes the source degree row* $\mathsf{UnaryHistory}(d)$*, the output degree row* $\mathsf{UnaryHistory}(d^+)$*, and the visible successor continuation row*

$$\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^+).$$

These are the only degree rows made available to a downstream exterior-derivative consumer by the package.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger degree successor nonempty`

Proof. Unpack the accepted package of Definition 136.68. Its degree fields are listed as the two unary-history rows and the displayed $\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^+)$ row.

The source and output are carried rows of Definition 136.1, so their degree components are visible BHist carrier entries rather than hidden operator data. Applying the source-scope boundary of Definition 136.62 shows that no other degree provenance may be imported through this package. □

Theorem 136.70 (Differential-form exterior-derivative input classifier stability)**.** *Let* $\mathsf{DiffFormExteriorInput}_{M,T}(\omega,$ *be accepted. If the source and output carrier rows are transported componentwise by* hsame *to* ω' *and* $D\omega'$*, then* $\mathsf{DiffFormExteriorInput}_{M,T}(\omega', D\omega')$ *is accepted. Under the same transport, the retained scalar endpoint comparison between the source and output scalar endpoints remains the scalar comparison carried by the transported package.*

Proof. First transport the two carrier rows. By Definition 136.10, the transported source ω' and transported output $D\omega'$ remain accepted $\mathsf{DiffForm}^{\uparrow}$ carrier rows.

Next transport the classifier data. Apply Definition 136.11 to the componentwise hsame comparisons of the two carrier rows. The degree rows, probe bundles, tensor rows, antisymmetry ledgers, and operator-owned source ledger remain the same visible rows up to hsame, and the successor continuation row $\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^+)$ transports to the corresponding continuation row for the transported degree endpoints.

Finally, apply Definition 136.64. The retained scalar endpoint comparison in the accepted package is preserved by the same componentwise transport, so the transported rows reassemble as $\mathsf{DiffFormExteriorInput}_{M,T}(\omega', D\omega')$ with its scalar comparison still visible. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeInputRow classifier transport`

Definition 136.71 (Differential-form downstream obligation inventory)**.** *The downstream obligation inventory for* $\mathsf{DiffForm}^{\uparrow}$ *is the ten-field boundary used when* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$ *consume the differential-form surface:*

1. *traditional target:* $\mathsf{DiffForm}^{\uparrow}$ *as the boundary source for* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$*;*
2. *BEDC source: BHist,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*;*
3. *carrier:* $\mathsf{DiffFormBHistCarrier}_{M,T}$*;*
4. *classifier:* $\sim_{\mathsf{DiffForm}^{\uparrow}}$*;*
5. *stability obligations: carrier* hsame *transport, form classifier stability, scalar endpoint transport, and exterior-input classifier stability;*
6. *ledger: only degree, probe, tensor, scalar, antisymmetry, and source rows may be projected;*
7. *core theorem: the carrier, exterior-input, and consumer-boundary rows displayed in this file;*
8. *exactness: row exhaustion of the displayed carrier and exterior-input packages;*
9. *standard bridge: absent data;*
10. *status:* $\mathsf{ProofTarget}$*.*

This inventory is a boundary statement. It supplies the rows that later chapters may consume, and it does not assert $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, or any standard bridge.*

Theorem 136.72 (Differential-form downstream carrier-source obligation)**.** *Every downstream consumer of a* $\mathsf{DiffForm}^{\uparrow}$ *carrier row may project only the six carrier coordinates*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and the source scope

$$BHist,\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{ProbeBundle},\ \mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}},\ \mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}.$$

No $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, or* $\mathsf{Contact}^{\uparrow}$ *consumer obtains an additional carrier coordinate, hidden classifier, operator row, or geometric law from the* $\mathsf{DiffForm}^{\uparrow}$ *carrier alone.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDownstreamCarrierSource obligation`

Proof. Project the carrier row by Definition 136.1. A carried form is exactly the six-tuple $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, so the degree, probe-bundle, tensor, scalar endpoint, antisymmetry, and source-ledger entries are the full carrier surface.

Next apply Definition 136.62. The source ledger for any accepted carrier row is restricted to $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

Combining the carrier projection with the source-scope projection exhausts what a downstream consumer can read. Operator identities and geometric laws are not carrier coordinates and are not named by the source ledger, so they cannot be projected from the carrier alone. □

Theorem 136.73 (Differential-form downstream exterior-input obligation)**.** *An accepted exterior-derivative input package exposes only the source carried row, the output carried row, the one-step degree continuation ledger, the probe transport ledger, the* $\mathsf{TensorProduct}^{\uparrow}$ *tensor transport ledger, the retained scalar endpoint transport, and the restricted source ledger. It exposes no* d^2 *row, closedness row, nondegeneracy row, contact nonintegrability row, or second-derivative law.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger downstream input obligation`

Proof. Unpack Definition 136.68. The package lists the source and output carried rows, the unary degree histories, the displayed one-step continuation $\mathsf{Cont}(d, \mathsf{E1}(\mathsf{e}), d^{+})$, the probe transport rows, the tensor row, the scalar endpoint comparison, and the operator-owned source ledger.

Transport stability is exactly the row supplied by Definition 136.70. It keeps the package accepted under componentwise hsame transport and preserves the retained scalar endpoint comparison as the scalar comparison of the transported package.

The displayed list contains no row for d^2, closedness, nondegeneracy, contact nonintegrability, or a second application of the exterior derivative. By exhausting the visible package fields, those laws remain outside the $\mathsf{DiffForm}^{\uparrow}$ exterior-input obligation. □

Theorem 136.74 (Differential-form downstream consumer boundary)**.** *The chapters* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$ *consume only the displayed* $\mathsf{DiffForm}^{\uparrow}$ *carrier, source, and exterior-input rows of Definition 136.71. Each such chapter must supply its own operator law or geometric law:* $d^2 = 0$ *for de Rham structure, closedness and nondegeneracy for symplectic structure, and contact nonintegrability for contact structure.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDownstreamConsumer boundary`

Proof. First use Definition 136.72. It bounds every carrier projection by the six displayed carrier coordinates and the six named BEDC sources.

Next use Definition 136.73. It permits an exterior-derivative consumer to read the source and output carried rows, one-step degree bookkeeping, probe and tensor transport, scalar endpoint transport, and the restricted source ledger, but not d^2, closedness, nondegeneracy, or contact nonintegrability.

Finally apply Definition 136.61. That theorem identifies $\mathsf{DiffForm}^{\uparrow}$ as the carrier, classifier, exterior-algebra substrate, degree-raise candidate row, and endpoint-transport substrate for $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$. Composing it with the two obligation rows above leaves exactly the displayed DiffForm rows available to downstream consumers, so their operator and geometric laws must be supplied in their own certificate surfaces. □

Definition 136.75 (Differential-form antisymmetry chain ledger)**.** *For carried rows*

$$\omega = (d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and

$$\omega' = (d, \mathcal{P}', \tau', \sigma', \alpha', \lambda')$$

in $\mathsf{DiffFormBHistCarrier}_{M,T}$*, a differential-form antisymmetry chain ledger is a visible BHist row* $\mathsf{AntiChain}_{\omega,\omega'}$ *accepted by* $\mathsf{DiffFormAntisymmetryChainLedger}_{M,T}$ *exactly when it displays a finite spine of adjacent-swap rows*

$$\mathcal{P} = \mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_n = \mathcal{P}'$$

over the carried ProbeBundle*, with one scalar endpoint row and one signed scalar comparison row recorded for each adjacent step* $\mathcal{P}_i \rightsquigarrow \mathcal{P}_{i+1}$*. Its accepted source rows are only BHist,* hsame*,* Cont*, the displayed* ProbeBundle *rows,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*; the ledger does not contain any permutation oracle outside those rows.*

Theorem 136.76 (Differential-form antisymmetry chain coverage)**.** *Every admitted finite permutation comparison between two carried* $\mathsf{DiffForm}^{\uparrow}$ *rows over the same degree row is accepted only through a differential-form antisymmetry chain ledger of Definition 136.75. In particular, each comparison row in the ledger is a visible adjacent-swap row, each adjacent swap has the reverse row supplied by Definition 136.13, and the composite comparison uses no source beyond BHist,* hsame*,* Cont*,* ProbeBundle*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*

Proof. Read the admitted permutation comparison inside the carrier row of Definition 136.1. The only place where antisymmetry data may enter a carried $\mathsf{DiffForm}^{\uparrow}$ row is the displayed ledger α together with the provenance ledger λ, so a permutation comparison without visible rows has no carrier witness.

Induct on the displayed finite adjacent-swap spine. The empty spine compares the probe bundle with itself, and the endpoint comparison is the retained scalar classifier clause already present in Definition 136.2.

For the cons step, the head row is one adjacent swap of neighboring carried probe slots. Apply Definition 136.13 to obtain the reverse row and the scalar comparison returning the double swap to the original endpoint. The induction hypothesis supplies coverage for the tail of the chain, and hsame transitivity composes the visible BHist rows.

All rows used in the construction are the rows listed in Definition 136.75. Therefore the finite permutation comparison is exhausted by the adjacent-swap ledger, with no hidden permutation rule or external classifier. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormAntisymmetryChainLedger coverage`

Theorem 136.77 (Differential-form antisymmetry chain classifier stability)**.** *Let ω and η be carried* $\mathsf{DiffForm}^{\uparrow}$ *rows whose permutation comparison is witnessed by a differential-form antisymmetry chain ledger. If ω and η are transported componentwise by* hsame *comparisons on their degree rows, probe bundles, tensor rows, scalar endpoint rows, antisymmetry ledgers, and source ledgers, then the transported rows carry the transported antisymmetry chain ledger and remain compared by* $\sim_{\mathsf{DiffForm}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormAntisymmetryChainLedger classifier stability`

Proof. First transport the two carrier rows by Definition 136.10. Componentwise hsame comparisons carry the probe bundles, tensor rows, scalar endpoints, antisymmetry ledgers, and source ledgers to the transported rows while retaining the same manifold and tensor source certificates.

Next transport the finite chain ledger. Each row of Definition 136.75 is a visible BHist row, and each adjacent swap in the chain is a row over the transported ProbeBundle. Transport the scalar endpoint comparison for each step through Definition 136.64.

Apply Definition 136.76 to the transported chain. The theorem identifies the transported finite permutation comparison as exactly the composite of adjacent swaps in the visible ledger, and it retains the reverse rows supplied by Definition 136.13.

Finally apply Definition 136.11. The degree, tensor, antisymmetry, and source-ledger clauses are hsame clauses, while the scalar endpoint clause has already been transported through the retained scalar classifier. These clauses reassemble into the required $\sim_{\mathsf{DiffForm}^{\uparrow}}$ comparison for the transported rows. □

Theorem 136.78 (Differential-form derived antisymmetry normal form)**.** *Every finite permutation comparison admitted for carried* $\mathsf{DiffForm}^{\uparrow}$ *rows over the same degree row factors through the differential-form antisymmetry chain ledger of Definition 136.75. Each step of the factorization is a visible adjacent swap, reverse rows are supplied by Definition 136.13, and any transported comparison remains a* $\mathsf{DiffForm}^{\uparrow}$ *classifier comparison by Definition 136.77.*

Proof. Begin with the carried rows in Definition 136.1. Their permutation data are not an external oracle: the carrier exposes only the antisymmetry ledger and the source ledger, so any admitted comparison must be read as a displayed finite spine of probe-bundle rows.

Apply Definition 136.75 to identify this spine as adjacent swaps over the carried ProbeBundle rows. For each step, Definition 136.13 supplies the reverse row and the scalar endpoint comparison needed to return along the same visible data.

Finally transport the comparison componentwise. The stability theorem Definition 136.77 carries the degree, probe, tensor, scalar endpoint, antisymmetry, and source rows to the transported carrier rows, so the derived comparison remains internal to $\sim_{\mathsf{DiffForm}^{\uparrow}}$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDerivedAntisymmetry normal form`

Theorem 136.79 (Differential-form de Rham boundary consumption)**.** *A downstream* $\mathsf{DeRham}^{\uparrow}$ *exterior-derivative row can consume* $\mathsf{DiffForm}^{\uparrow}$ *only through the carried form rows, their classifier rows, wedge ledger coverage, the wedge degree ledger, exterior-derivative candidate and input rows, scalar endpoint transport, and the antisymmetry normal form of Definition 136.78. It cannot read $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability from* $\mathsf{DiffForm}^{\uparrow}$.

Proof. The carrier and classifier entries are exactly the displayed $\mathsf{DiffFormBHistCarrier}_{M,T}$ and $\sim_{\mathsf{DiffForm}^\uparrow}$ rows. Wedge admission is bounded by Definition 136.15, and its degree bookkeeping is the ledger of Definition 136.22.

The exterior-derivative interface is bounded by Definition 136.24 and Definition 136.61. These theorems list the visible candidate rows, input rows, probe/tensor/scalar transport rows, and source ledgers available to a consumer.

It remains to control permutation comparisons inside the same boundary. Apply Definition 136.78: every admitted comparison is decomposed into the chain ledger, adjacent-swap reverse rows, and classifier-stable transported rows.

None of these cited rows contains an operator-square identity, a closedness predicate, a nondegeneracy predicate, or a contact nonintegrability predicate. Those obligations therefore must be stated by the consuming $\mathsf{DeRham}^\uparrow$, $\mathsf{Symplectic}^\uparrow$, or $\mathsf{Contact}^\uparrow$ surface rather than read from $\mathsf{DiffForm}^\uparrow$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormAntisymmetryBoundary derham consumption separation`
Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger classifier transport nonempty boundary`

Theorem 136.80 (Differential-form wedge derivative input boundary)**.** *When the same carried form appears both in a wedge row and in an exterior-derivative input package, the shared visible data are exactly the unary degree row, the wedge degree continuation ledger, the one-step derivative continuation ledger, the probe, tensor, and scalar transport rows, and the source ledger. No Leibniz rule and no nilpotence row are supplied by this shared boundary.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDerivativeInput boundary`

Proof. Read the wedge side first. The available multiplicative data are the tensor coverage theorem Definition 136.15 and the unary degree continuation ledger of Definition 136.22; both are rows over the same carried degree, probe, tensor, scalar endpoint, and source entries.

Read the exterior-derivative side through Definition 136.24 and Definition 136.61. These rows expose the derivative candidate, the one-step degree continuation, and the transport clauses for the probe bundle, tensor row, scalar endpoint, and source ledger.

The intersection of the two readings is therefore the visible carrier data they both name. Antisymmetry comparisons inside that intersection are governed by Definition 136.78, so even transported permutation data stay ledger-based.

A Leibniz rule would relate a derivative of a wedge to a sum of two derivative terms, and nilpotence would relate two derivative steps. Neither row is among the wedge coverage, wedge degree, exterior-derivative input, transport, or source-ledger rows listed above, so the shared boundary supplies neither one. □

Definition 136.81 (Differential-form admissible root row)**.** *For fixed sources* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$*, an admissible* $\mathsf{DiffForm}^\uparrow$ *root row is exactly the visible BEDC tuple*

$$(BHist, \mathsf{hsame}, \mathsf{Cont}, \mathsf{ProbeBundle}, \mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}, \mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}, \mathsf{Carr}, \mathsf{Class}, \mathsf{Wedge}, \mathsf{Der})$$

where Carr *is the carrier row of Definition 136.1,* Class *is the classifier row of Definition 136.2,* Wedge *is the visible family consisting of Definition 136.15, Definition 136.16, Definition 136.19, and Definition 136.22, and* Der *is the exterior-derivative input family consisting of Definition 136.65 and Definition 136.66. No further carrier, classifier, closedness, nilpotence, nondegeneracy, contact nonintegrability, or bridge row is part of the root row.*

Theorem 136.82 (Differential-form root-row carrier scope)**.** *Every admissible root row of Definition 136.81 exposes only the carried form rows of Definition 136.1, the classifier rows of Definition 136.2, and the source boundary of Definition 136.62 on its carrier side.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootRow carrier scope`

Proof. Unpack the root row. Its carrier coordinate is named as Carr and is definitionally the row of Definition 136.1; its classifier coordinate is Class and is definitionally the row of Definition 136.2.

Project the carried row through Definition 136.62. The only structural sources available to the carrier and classifier are $BHist$, hsame, Cont, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

The definition of the root row contains no second carrier slot and no additional classifier slot. Hence every carrier-side read factors through the two displayed definitions and the cited source boundary. □

Theorem 136.83 (Differential-form root-row wedge scope)**.** *Every wedge use admitted by Definition 136.81 is limited to Definition 136.15, Definition 136.16, Definition 136.19, and Definition 136.22.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootRow wedge scope`

Proof. Inspect the Wedge coordinate of the root row. It is declared as the four displayed wedge rows and no operator law beyond those rows.

The coverage theorem supplies the admitted tensor concatenation surface. The probe-concatenation ledger records the probe spine, the associator ledger records parenthesized tensor transport, and the degree ledger records the visible unary degree bookkeeping.

Since the root row has no other wedge coordinate, any downstream wedge read must choose one of these four rows. A Leibniz rule, closedness row, or nilpotence row is therefore not present in the $\mathsf{DiffForm}^{\uparrow}$ root surface. □

Theorem 136.84 (Differential-form root-row derivative scope)**.** *Every exterior-derivative input admitted by Definition 136.81 is one of the candidate rows of Definition 136.65 together with the exactness rows of Definition 136.66.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootRow derivative scope`

Proof. Read the Der coordinate. It lists the candidate row definition and the input-output ledger exactness theorem as the complete exterior-derivative input family.

The candidate row supplies only the source carried row, target carried row, degree entries, one-step continuation row, probe bundle, tensor row, scalar endpoint, antisymmetry ledger, and source ledger. The exactness theorem then states that accepted input-output use is exhausted by those visible rows.

No additional derivative law is declared in the root row. Thus a second derivative identity, closedness assertion, or bridge comparison must come from a consumer certificate rather than from $\mathsf{DiffForm}^{\uparrow}$. □

Theorem 136.85 (Differential-form root-row downstream boundary)**.** *The chapters* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$ *may consume the* $\mathsf{DiffForm}^{\uparrow}$ *surface only through the admissible root row of Definition 136.81. They must supply closedness, nondegeneracy, and contact nonintegrability in their own certificates.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootRow downstream boundary`

Proof. Combine the three scope projections above. The carrier projection gives only the carrier, classifier, and source rows; the wedge projection gives only the four wedge ledgers; the derivative projection gives only the candidate rows and input-output exactness rows.

A $\mathsf{DeRham}^{\uparrow}$ consumer may therefore read the displayed exterior-derivative input family, but it receives no $d^2 = 0$ or closedness row. A $\mathsf{Symplectic}^{\uparrow}$ consumer may read carried forms and wedge ledgers, but it receives no nondegeneracy row.

A $\mathsf{Contact}^{\uparrow}$ consumer may read the same carrier, wedge, and derivative surface, but it receives no contact nonintegrability row. These three geometric obligations are not coordinates of the admissible root row, so each must be certified on the consuming surface itself. □

Definition 136.86 (Differential-form root source obligation package)**.** *The root source obligation package for* $\mathsf{DiffForm}^{\uparrow}$ *is the following per-object template, read over the certificate section section 136.2.*

1. *The traditional target is the differential-form surface of antisymmetric multilinear tangent probes equipped with wedge.*
2. *The BEDC source language is* BHist*, with comparison rows supplied by* hsame*.*
3. *Continuation rows are exactly the visible* Cont *rows attached to tensor concatenation, wedge degree bookkeeping, and the exported exterior-derivative input boundary.*
4. *Probe data are finite* $\mathsf{ProbeBundle}$ *rows over the tangent source surface.*
5. *The upstream certificate sources are* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*
6. *The carrier choice is* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *of Definition 136.1.*
7. *The classifier choice is* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *of Definition 136.2.*
8. *Stability obligations are the carrier transport, classifier stability, wedge probe transport, wedge associator transport, wedge degree stability, antisymmetry transport, and exterior-derivative input transport rows already displayed in the chapter.*
9. *The source ledger is the visible ledger over* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*
10. *The core theorem list consists of carrier obligation, classifier obligation, ledger exactness, classifier reflexivity, wedge ledger coverage, wedge probe concatenation coverage, wedge associator closure, wedge degree stability, and the exported input-boundary statements.*
11. *The exactness predicate is ledger exhaustion: every accepted root row is one of the displayed carrier, classifier, continuation, probe, upstream certificate, wedge, antisymmetry, or input-boundary rows.*
12. *The standard bridge slot is absent, and the current status is a paper-side root obligation surface rather than a standard-object bridge.*

No closedness predicate, nondegeneracy predicate, contact predicate, or $d^2 = 0$ *row belongs to this root source package.*

Theorem 136.87 (Differential-form root carrier source obligation)**.** *Every admitted root* $\mathsf{DiffForm}^{\uparrow}$ *row factors through the six displayed carrier coordinates*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

of Definition 136.1 and through the upstream $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *certificate rows. The root carrier has no host tangent-space carrier and no external alternating-map carrier outside those displayed BEDC sources.*

Proof. Unpack the carrier definition. A row accepted by $\mathsf{DiffFormBHistCarrier}_{M,T}$ is presented by the six histories $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, and the acceptance clauses name only degree, probe, tensor, scalar endpoint, antisymmetry, and source-ledger data.

The probe coordinate is read through the tangent-source rows retained by $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$. The tensor and continuation coordinates are read through $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$, while the scalar endpoint classifier is the scalar classifier retained by the same upstream sources.

Finally inspect the source ledger λ. It records the names M and T and the five visible carrier components already listed. Since no acceptance clause constructs another carrier, a host tangent-space object or an external alternating-map object has no entry point into the root row. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootCarrierSource obligation`

Theorem 136.88 (Differential-form root wedge operation obligation)**.** *The wedge operation obligation for root* $\mathsf{DiffForm}^\uparrow$ *rows is exactly the visible concatenation, degree-additivity, associator, and classifier-stability ledger supplied by Definition 136.15, Definition 136.16, Definition 136.17, Definition 136.18, Definition 136.19, Definition 136.20, Definition 136.22, and Definition 136.23.*

Proof. Read wedge admission first. The coverage theorem Definition 136.15 bounds the operation by the two input carrier rows, the concatenated tensor row, the output scalar endpoint, the inherited antisymmetry data, and the displayed source ledger.

Next read the probe and degree ledgers. The probe spine is the visible concatenation ledger of Definition 136.16, with coverage and classifier transport supplied by Definition 136.17 and Definition 136.18. The degree row is the unary-history ledger of Definition 136.22, and its classifier transport is Definition 136.23.

It remains to account for parenthesizing. The associator data are exactly the visible ledger of Definition 136.19, and closure under the carried classifier is supplied by Definition 136.20. These cited rows exhaust the operation obligation, so no additional wedge source is being assumed. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootWedgeOperation obligation`

Theorem 136.89 (Differential-form root source ledger obligation)**.** *All accepted source ledger entries for the root* $\mathsf{DiffForm}^\uparrow$ *package are exhausted by* $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *rows. No standard bridge, closedness predicate, nondegeneracy predicate, contact predicate, or* $d^2 = 0$ *row is part of the root obligation.*

Proof. Begin with the root package of Definition 136.86. Its source language, comparison rows, continuation rows, probe rows, and upstream certificate rows are explicitly listed as $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$.

The carrier and classifier definitions do not widen that list. Definition 136.1 exposes only the six carried coordinates and their upstream sources, while Definition 136.2 uses only hsame and the retained scalar classifier. The core ledger exactness statement Definition 136.6 gives the same exhaustion for the core certificate rows.

The remaining root operation rows are wedge rows. By Definition 136.88, these are the visible concatenation, degree, associator, and classifier-stability ledgers already named above. None of those ledgers contains a standard bridge, a closedness predicate, a nondegeneracy predicate, a contact predicate, or a two-step exterior-derivative identity. □

Definition 136.90 (Differential-form root obligation package)**.** *The root obligation package for* $\mathsf{DiffForm}^{\uparrow}$ *is the displayed* $BHist$ *package over* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *whose public rows are the following four surfaces.*

1. *The visible source surface is the carrier and classifier surface of Definition 136.1 and Definition 136.3, with no source outside the six displayed inputs.*
2. *The antisymmetry ledger is the finite adjacent-swap chain surface of Definition 136.75, transported through the* $\mathsf{DiffForm}^{\uparrow}$ *classifier.*
3. *The wedge-probe ledger is the wedge degree, tensor-continuation, and probe-concatenation surface carried by admitted wedge rows.*
4. *The downstream consumption row is the boundary through which* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$ *may consume* $\mathsf{DiffForm}^{\uparrow}$ *data.*

The package exports no standard bridge and no geometric identity beyond these listed rows.

Theorem 136.91 (Differential-form visible source surface obligation)**.** *The root obligation package of Definition 136.90 exposes no source beyond* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*. Its carrier, classifier, and core ledger rows are exactly the rows of Definition 136.1, Definition 136.3, Definition 136.6, and Definition 136.62.*

Proof. Read the carrier first. By Definition 136.1, a carried form row consists of the degree history, probe bundle, tensor row, scalar endpoint row, antisymmetry ledger, and source ledger. Each entry is a visible row over the displayed $BHist$, $\mathsf{ProbeBundle}$, Cont, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, or $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ surface.

The classifier and core rows introduce no further source. The core rows of Definition 136.3 are exhausted by the displayed carrier, classifier, transport, wedge, antisymmetry, and source provenance clauses, and Definition 136.6 states that every accepted ledger row is visible BHist provenance.

Finally apply Definition 136.62. It identifies the source boundary of every accepted row with the same six inputs named in the root package. Hence the root package exposes the displayed kernel and upstream NameCert rows, and nothing else. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootObligationPackage visible source surface`

Theorem 136.92 (Differential-form root antisymmetry ledger obligation)**.** *In the root obligation package of Definition 136.90, antisymmetry enters only through the finite* $\mathsf{ProbeBundle}$ *permutation chain ledger of Definition 136.75. Every admitted antisymmetry comparison is covered by Definition 136.76 and transports through the* $\mathsf{DiffForm}^{\uparrow}$ *classifier by Definition 136.77.*

Proof. Locate the antisymmetry data inside the carried form row. The carrier has a single antisymmetry ledger slot, and Definition 136.75 identifies its public content as a finite spine of adjacent-swap rows over the carried $\mathsf{ProbeBundle}$.

Apply Definition 136.76. The theorem exhausts an admitted finite permutation comparison by the displayed adjacent swaps and their scalar endpoint rows. No permutation oracle or hidden classifier clause is available outside that ledger.

It remains to check transport. By Definition 136.77, componentwise hsame transport of the carried rows transports the finite chain ledger and reassembles the transported endpoints as a $\mathsf{DiffForm}^{\uparrow}$ classifier comparison. Thus the root package treats antisymmetry only as visible ledger data. □

Theorem 136.93 (Differential-form root wedge-probe obligation)**.** *Every wedge row exposed by the root obligation package of Definition 136.90 exposes precisely the degree additivity ledger, probe concatenation ledger, tensor-continuation provenance, and classifier stability rows supplied by Definition 136.14, Definition 136.15, Definition 136.17, and Definition 136.18.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootWedgeProbe obligation`

Proof. Begin with an admitted wedge row. The coverage theorem Definition 136.15 requires a visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation row joining the tensor component of the left form to the tensor component of the right form. This is the tensor continuation provenance available at the root boundary.

Next read the degree and probe rows. The degree additivity statement of Definition 136.14 records the unary degree continuation for the wedge output. The probe coverage theorem Definition 136.17 records the output probe bundle as the append of the two input probe spines, with the visible length and membership clauses.

Finally transport the row. The classifier stability theorem Definition 136.18 transports the concatenated probe ledger together with the tensor and scalar endpoint comparisons. These are exactly the wedge rows listed in the root package, so no further multiplicative source is exposed. □

Theorem 136.94 (Differential-form root downstream consumption obligation)**.** *The downstream consumers* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *receive from the root obligation package of Definition 136.90 only the carried* $\mathsf{DiffForm}^{\uparrow}$ *rows, classifier rows, wedge rows, antisymmetry rows, and exterior-input boundary rows described by Definition 136.61, Definition 136.79, Definition 136.80, and Definition 136.165. The identities* $d^2 = 0$*, closedness, nondegeneracy, and contact nonintegrability remain obligations of the consuming chapters.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDownstreamConsumption obligation`

Proof. Inspect the exported surface first. Definition 136.165 lists the carrier, classifier, transport, wedge, antisymmetry, exterior-input, and downstream scope rows that the $\mathsf{DiffForm}^{\uparrow}$ surface exports. This is the entire public obligation surface of the root package.

Now apply the downstream boundary statements. Definition 136.61 bounds exterior-derivative consumers to the degree row, probe bundle, tensor row, scalar endpoint row, antisymmetry ledger, wedge ledgers, and exterior-derivative candidate ledger. Definition 136.79 gives the same boundary for $\mathsf{DeRham}^{\uparrow}$ consumption, including the normal-form antisymmetry row.

The shared wedge and derivative input boundary is controlled by Definition 136.80. It permits only the visible unary degree row, degree-continuation ledgers, probe, tensor, scalar, and source rows common to the wedge and exterior-input readings.

None of the cited rows contains nilpotence, closedness, nondegeneracy, or contact nonintegrability. Therefore those four facts are not imported from $\mathsf{DiffForm}^{\uparrow}$; they must be supplied by the $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ chapters that use the root package. □

Theorem 136.95 (Differential-form root source obligation)**.** *The root* $\mathsf{DiffForm}^{\uparrow}$ *chapter reads every displayed differential-form row only from* $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*, as witnessed by the carrier of Definition 136.1. In particular, the root surface has no source row outside the six displayed entries*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda).$$

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootSource obligation`

Proof. Read the root chapter through the carrier definition. The row $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$ displays the degree history, the probe bundle, the tensor row, the scalar endpoint, the antisymmetry ledger, and the source ledger.

Each coordinate is sourced by one of the named BEDC rows. The degree, endpoint, antisymmetry, and provenance entries are $BHist$ rows; comparison and transport use hsame; tensor concatenation uses Cont; probe exposure uses ProbeBundle; and the only upstream certificates are $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$.

No clause of Definition 136.1 allocates another source. Therefore the root chapter may be scanned as exactly this displayed source surface. □

Theorem 136.96 (Differential-form root carrier and classifier obligation)**.** *The carrier and classifier obligations visible at the* $\mathsf{DiffForm}^\uparrow$ *root are exactly the carrier of Definition 136.1, the classifier of Definition 136.2, carrier transport of Definition 136.10, and classifier stability of Definition 136.11. No further root classifier or carrier transport principle is exposed.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootCarrierClassifier obligation`

Proof. The carrier obligation is the accepted BHist row of Definition 136.1. Its six displayed coordinates give the complete carrier data used by all later wedge and boundary rows.

The classifier obligation is the comparison of Definition 136.2. It compares degrees by hsame, matched tensor and ledger rows by visible BHist comparisons, and scalar endpoints by the retained scalar classifier from the upstream sources.

The two stability obligations are precisely the transport theorem Definition 136.10 and the classifier theorem Definition 136.11. Those results repack the same displayed rows after componentwise hsame transport, so the root obligation set contains no additional carrier or classifier field. □

Theorem 136.97 (Differential-form root wedge ledger obligation)**.** *The wedge surface visible at the* $\mathsf{DiffForm}^\uparrow$ *root is exhausted by wedge ledger coverage in Definition 136.15, the probe-concatenation ledger of Definition 136.16, the wedge-degree ledger of Definition 136.22, and the associator ledger of Definition 136.19. All four ledgers are rows over the displayed BEDC carrier and classifier surface.*

Proof. Start with an admitted wedge row. By Definition 136.15, admission requires a visible $\mathsf{TensorProduct}^\uparrow$ concatenation row joining the tensor components of the two carried inputs.

The output probe data are then bounded by Definition 136.16; its coverage theorem identifies the output bundle as the displayed append of the two input probe spines. The degree data are bounded by Definition 136.22, which records only the two input degree rows, the concatenation provenance, and the output unary degree row.

Associativity is not a hidden algebraic law at the root. It is the displayed ledger of Definition 136.19, with closure stated by Definition 136.20. These four rows exhaust the wedge surface because each refers back to the same carried form rows and their visible hsame, Cont, probe, tensor, scalar, and antisymmetry ledgers. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootWedgeLedger obligation`

Theorem 136.98 (Differential-form root exterior boundary obligation)**.** *Exterior-derivative consumers of* $\mathsf{DiffForm}^\uparrow$ *may use only the candidate ledger of Definition 136.54, the input row of Definition 136.57, and the boundary theorems already displayed for degree shift, scalar transport, wedge-input stability, and downstream scope. The identities* $d^2 = 0$*, closedness, nondegeneracy, and contact nonintegrability are outside this* $\mathsf{DiffForm}^\uparrow$ *root surface.*

Proof. First read the candidate ledger. It exposes an input carrier row, an output carrier row, the successor-degree continuation, and transported probe, tensor, and scalar endpoint rows. Its degree-raise theorem records only this degree-raising candidate data.

Next read the accepted input row. The row of Definition 136.57 supplies the source and target carried forms, the source and target degree rows, the one-step continuation, and the carrier ledgers exposed by the de Rham input boundary.

Finally apply the boundary theorems already present in the chapter: Definition 136.56, Definition 136.58, Definition 136.59, and Definition 136.61. These theorems permit downstream consumers to cite the displayed candidate and input ledgers, while leaving nilpotence, closedness, nondegeneracy, and contact nonintegrability to the downstream certificate surfaces. □

136.6 DiffForm root export threshold

Definition 136.99 (Differential-form root export threshold source)**.** *Fix the* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and the* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T *used in Definition 136.1. The root export threshold source for* $\mathsf{DiffForm}^{\uparrow}$ *is the displayed row package consisting exactly of* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*, the carrier and classifier rows of Definition 136.1 and Definition 136.2, the wedge-ledger rows cited by Definition 136.97, the exterior-derivative input boundary cited by Definition 136.98, and the downstream-consumer boundary of Definition 136.74.*

This threshold source is an export boundary. It names the rows that a consumer may read from the $\mathsf{DiffForm}^{\uparrow}$ *root surface, and it names no nilpotence, closedness, nondegeneracy, contact nonintegrability, or standard bridge row.*

Theorem 136.100 (Differential-form carrier and classifier export threshold)**.** *At the threshold of Definition 136.99, the exported carrier is exactly the six-coordinate row*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

of Definition 136.1. The exported classifier is only the comparison of degree rows by hsame*, matched tensor and scalar endpoint comparisons through the displayed source rows, and visible ledger comparisons by* hsame *as stated in Definition 136.2. No additional carrier coordinate, classifier, operator row, or geometric law is exported at the root threshold.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerExport exactness`

Proof. Read the source boundary through Definition 136.95. That theorem restricts every root row to the displayed $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ sources, with no source row outside the six carrier entries.

Project the carrier with Definition 136.1. The row is the tuple $(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$, and each coordinate is respectively the degree history, probe bundle, tensor row, scalar endpoint, antisymmetry ledger, or source ledger. Therefore the export carrier has no seventh coordinate.

Project the classifier with Definition 136.2. Its degree clause is hsame, its tensor and scalar clauses compare matched displayed rows using the retained source classifiers, and its remaining clauses are visible ledger hsame comparisons. These are precisely the classifier rows permitted by the threshold source. □

Theorem 136.101 (Differential-form wedge export threshold)**.** *At the threshold of Definition 136.99, the exported wedge surface is exactly the visible tensor concatenation row, the probe-append ledger,*

the degree-bookkeeping ledger, and the antisymmetry ledgers over the carried input rows. The root threshold exports no exterior law beyond these wedge rows.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootExport wedge threshold`

Proof. Start from the wedge boundary of Definition 136.97. It says that an admitted wedge row is exhausted by wedge ledger coverage, probe concatenation, degree bookkeeping, and associator ledger rows over the displayed carrier and classifier surface.

The tensor component is the visible $\mathsf{TensorProduct}^{\uparrow}$ concatenation row joining the tensor entries of the two carried forms. The probe component is the displayed append of the two finite $\mathsf{ProbeBundle}$ spines, and the degree component is the ledger recording the two input degrees and the output degree row.

The antisymmetry contribution is not a hidden exterior identity. It is the visible permutation and associator ledger attached to the carrier rows and transported through hsame. Since each wedge row cited by the root wedge obligation refers back to those displayed histories, the threshold exports exactly the stated wedge surface. □

Theorem 136.102 (Differential-form downstream law separation)**.** *The root threshold of Definition 136.99 separates the usable* $\mathsf{DiffForm}^{\uparrow}$ *surface from downstream exterior laws:* $\mathsf{DeRham}^{\uparrow}$ *must supply nilpotence,* $\mathsf{Symplectic}^{\uparrow}$ *must supply closedness and nondegeneracy, and* $\mathsf{Contact}^{\uparrow}$ *must supply contact nonintegrability in their own certificate surfaces. None of those laws is exported by the* $\mathsf{DiffForm}^{\uparrow}$ *root threshold.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootExport downstream law separation`

Proof. First apply Definition 136.98. It permits exterior-derivative consumers to use only the displayed candidate ledger, input row, degree shift, scalar transport, wedge-input stability, and downstream scope rows.

Next apply Definition 136.74. That theorem identifies the rows consumed by $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$, and states that $d^2 = 0$, closedness, nondegeneracy, and contact nonintegrability must be supplied by the corresponding downstream chapters.

Combining the two boundary statements gives the separation. The root threshold can export carried forms, classifier data, wedge ledgers, and exterior-input rows, but it cannot certify an operator identity or geometric law that is not one of those displayed rows. □

Theorem 136.103 (Differential-form NameCert root threshold)**.** *The paper-side* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *root threshold is assembled by the rows of Definition 136.99: the six-coordinate carrier, the displayed classifier, the wedge ledger surface, the exterior-derivative input boundary, and the downstream-consumer separation. This assembly exports the usable* $\mathsf{DiffForm}^{\uparrow}$ *root surface and leaves downstream operator and geometric laws off the root table.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootExport namecert threshold`

Proof. Use Definition 136.95 for the source rows and Definition 136.100 for the carrier and classifier rows. These two statements bind the export surface to the six displayed carrier coordinates and the classifier clauses already available from the upstream manifold and tensor sources.

Use Definition 136.101 for the wedge surface. It contributes exactly tensor concatenation, probe append, degree bookkeeping, and antisymmetry ledger data over the carried forms.

Finally use Definition 136.102. That statement imports the exterior-input boundary of Definition 136.98 and the consumer boundary of Definition 136.74. The assembled package is therefore

a root threshold for $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ and not a certificate for $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability. □

Definition 136.104 (Differential-form downstream boundary package)**.** *The root downstream boundary package for* $\mathsf{DiffForm}^{\uparrow}$ *is the visible certificate surface*

$$\mathsf{DiffFormDownstreamBoundary}_{M,T}.$$

It consists exactly of the carried-form rows of Definition 136.1, the classifier rows of Definition 136.2, the exterior-input rows of Definition 136.68, the wedge rows supplied by Definition 136.22, Definition 136.15, Definition 136.17, Definition 136.18, and Definition 136.23, the finite antisymmetry rows of Definition 136.75, and the downstream scope rows of Definition 136.61, Definition 136.24, and Definition 136.80. The package has no field for $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, or a standard bridge.*

Theorem 136.105 (Differential-form exterior input root closure)**.** *Every row of* $\mathsf{DiffFormExteriorInput}_{M,T}(\omega, D\omega)$ *that a downstream* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, or* $\mathsf{Contact}^{\uparrow}$ *chapter may read is already a row of* $\mathsf{DiffFormDownstreamBoundary}_{M,T}$*. In particular, the source carrier, output carrier, successor-degree continuation, transported* $\mathsf{ProbeBundle}$ *rows, transported* $\mathsf{TensorProduct}^{\uparrow}$ *tensor row, retained scalar endpoint comparison, and restricted source ledger are all root-visible rows, and no second exterior-derivative or closedness row is introduced by passing through the package.*

Proof. Unpack the exterior-input package of Definition 136.68. Its source and output are carried form rows of Definition 136.1; its comparison clauses are exactly the hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{TensorProduct}^{\uparrow}$, scalar endpoint, and source-ledger rows listed there and in Definition 136.2.

Repack those rows into Definition 136.104. The boundary package keeps the exterior-input row as a visible component, and it retains the classifier-stability interface through the same componentwise histories.

Finally apply the downstream scope theorem Definition 136.61. That theorem allows downstream exterior-derivative chapters to consume the carrier, classifier, degree-raise candidate, endpoint transport, and exterior-algebra substrate, but it does not add a second derivative, closedness, nondegeneracy, or contact nonintegrability row. Thus exterior input closes at the root boundary package without importing any downstream law. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorInput root transport closure`

Theorem 136.106 (Differential-form wedge derivative boundary compatibility)**.** *When a carried row participates both in an admitted wedge row and in an accepted exterior-input row, their common downstream boundary is exactly the intersection recorded by Definition 136.80 inside* $\mathsf{DiffFormDownstreamBoundary}_{M,T}$*. The common rows are the carried form row, its unary degree row, the wedge degree continuation ledger, the one-step exterior-derivative continuation ledger, the transported probe, tensor, scalar endpoint, antisymmetry, and source rows, and their* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *classifier comparisons.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDownstreamBoundary exterior wedge common rows`

Proof. Read the wedge side first. Wedge admission is bounded by Definition 136.15; the probe output is bounded by Definition 136.17; the degree bookkeeping is the visible ledger of Definition 136.22. Classifier transport for these rows is supplied by Definition 136.18 and Definition 136.23.

Read the exterior-input side next. By Definition 136.68, an accepted input exposes the source and output carried rows, the successor-degree continuation, transported probe and tensor rows, scalar endpoint comparison, and the operator-owned source ledger.

Now apply Definition 136.80. It states that the shared visible data are precisely the rows named by both readings, with no Leibniz rule and no nilpotence row. The antisymmetry portion is controlled by Definition 136.75 and remains classifier-stable through the displayed $\mathsf{DiffForm}^{\uparrow}$ comparison rows.

These rows are all components of Definition 136.104. Therefore the wedge and exterior-input interfaces are compatible at the downstream boundary without importing any closedness, nondegeneracy, contact nonintegrability, or operator-square assertion. □

Theorem 136.107 (Differential-form downstream boundary exhaustion)**.** *The package* DiffFormDownstreamBound *exhausts the* $\mathsf{DiffForm}^{\uparrow}$ *root rows visible to* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$. *Any downstream consumption of* $\mathsf{DiffForm}^{\uparrow}$ *must factor through the carrier and classifier rows, exterior-input rows, wedge rows, antisymmetry-chain rows, and the three downstream scope rows listed in Definition 136.104; no downstream consumer can project a closedness, nondegeneracy, contact nonintegrability, nilpotence, or bridge row from* $\mathsf{DiffForm}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormDownstreamBoundary exhaustion`

Proof. Start with an arbitrary downstream read of a $\mathsf{DiffForm}^{\uparrow}$ row. The carrier and classifier projections are exhausted by Definition 136.1, Definition 136.2, Definition 136.10, and Definition 136.11. These rows mention only $BHist$, hsame, Cont, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

For derivative input, apply Definition 136.105. It repacks every accepted exterior-input row into the boundary package and leaves all operator and geometric laws outside the root surface.

For wedge interaction, apply Definition 136.106. It identifies the wedge-degree, probe-concatenation, antisymmetry, and exterior-input rows that can be shared at the downstream boundary, and it excludes the Leibniz and nilpotence rows.

Finally use the downstream scope rows Definition 136.61, Definition 136.24, and Definition 136.80. They name $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ as the downstream consumers and restrict their access to the same visible carrier, classifier, wedge, antisymmetry, exterior-input, transport, and source ledgers. Since Definition 136.104 contains exactly those rows, every downstream read factors through it and no hidden row remains to be consumed. □

136.7 Boundary exhaustion for DiffFormUp

Definition 136.108 (Differential-form boundary exhaustion source)**.** *The boundary exhaustion source for* $\mathsf{DiffForm}^{\uparrow}$ *is the complete paper-side obligation surface displayed by the chapter. It consists of the following ten fields:*

1. *the traditional target surface of differential forms;*
2. *the BEDC sources* $BHist$, hsame, Cont, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*;*
3. *the carrier and classifier rows of Definition 136.33, Definition 136.37, and Definition 136.96;*
4. *the wedge ledger rows visible through Definition 136.39;*
5. *the exterior-derivative candidate row of Definition 136.65;*
6. *the input-output exactness row of Definition 136.66;*

7. *the downstream boundary rows for* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$*;*
8. *the exactness scope asserting that every displayed row is read as one of the named carrier, classifier, wedge, antisymmetry, exterior-input, transport, source-ledger, or downstream rows;*
9. *the absent standard bridge slot;*
10. *the status row saying that this is a paper-side obligation surface, not a standard-object bridge and not a Lean-checked theorem package.*

The source contains no row for $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, a de Rham quotient, or a standard bridge.*

Theorem 136.109 (Differential-form boundary carrier coverage)**.** *The carrier and classifier rows of Definition 136.108 are exactly the root source rows already displayed in Definition 136.33, Definition 136.37, and Definition 136.96. No additional carrier over tangent spaces, alternating maps, quotient forms, or external classifier is part of the* $\mathsf{DiffForm}^{\uparrow}$ *boundary exhaustion source.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBoundaryExhaustion carrier coverage`

Proof. Read the source surface first. The ten-field root-unblock surface lists the ambient $BHist$ carrier, the hsame comparisons, the Cont rows, the ProbeBundle spine, the two upstream NameCert sources, the displayed six-coordinate carrier, and the displayed $\mathsf{DiffForm}^{\uparrow}$ classifier.

Next read the root obligation source. It selects the same carried rows from Definition 136.1 and the same classifier rows from Definition 136.2, with the retained $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ sources.

Finally apply Definition 136.96. That theorem repackages the displayed carrier and classifier obligations without adding a host tangent-space carrier, an alternating-map carrier, or a quotient classifier. Therefore the carrier and classifier entries of Definition 136.108 are exactly the displayed root source entries. □

Theorem 136.110 (Differential-form wedge and derivative routing)**.** *The wedge, antisymmetry, degree, and exterior-derivative candidate rows of Definition 136.108 are exactly the rows supplied by Definition 136.39, Definition 136.40, Definition 136.65, and Definition 136.66. The boundary source routes those rows as visible ledger data and does not add a Leibniz rule, a nilpotence row, or a closedness row.*

Proof. Start with the wedge and antisymmetry side. The root wedge theorem supplies the admitted tensor concatenation ledger, the probe concatenation classifier stability row, the associator row, and the finite antisymmetry chain ledger. These are the only multiplicative and permutation rows named by the boundary source.

Now read the exterior-derivative side. The root exterior-derivative boundary obligation keeps only the candidate derivative row, the displayed input-output ledger, and the wedge-side boundary row consumed by downstream wedge formulas.

Unfold the candidate row. Its fields are the input carried form, the output carried form, the one-step degree-raise Cont ledger, the transported probe, tensor, scalar endpoint, and source ledgers, all over the same BEDC sources named in the boundary source.

Finally apply the input-output exactness theorem. It says that the candidate row exposes exactly those fields and no closedness, nondegeneracy, contact condition, or second-derivative identity. Hence the wedge, antisymmetry, degree, and derivative entries of the boundary source are precisely the cited ledger rows. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBoundaryExhaustion wedge derivative routing`

Theorem 136.111 (Differential-form downstream export exhaustion)**.** *The downstream chapters* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *may consume* $\mathsf{DiffForm}^{\uparrow}$ *only through the rows named by Definition 136.108: carrier rows, classifier rows, wedge rows, antisymmetry rows, the exterior-derivative candidate and input-output rows, transport rows, source-ledger rows, and the displayed downstream boundary rows. They receive no* $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, quotient, or standard-bridge row from* $\mathsf{DiffForm}^{\uparrow}$*.*

Proof. Project the common input first. By Definition 136.109, every downstream read begins with the same carried $\mathsf{DiffForm}^{\uparrow}$ rows and the same $\sim_{\mathsf{DiffForm}^{\uparrow}}$ classifier rows already displayed at the root boundary.

Project the operation rows next. By Definition 136.110, the only available wedge, antisymmetry, degree, and exterior-derivative entries are the visible ledger rows named by the root obligation and input-output exactness statements.

Now use the downstream boundary package. Its scope rows name $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ as consumers of the same carrier, classifier, wedge, antisymmetry, exterior-input, transport, and source-ledger surface, and they exclude nilpotence and geometric laws from the exported DiffForm boundary.

Repack these projections as Definition 136.108. Every downstream row is one of the displayed source fields, and none of the cited projections contains a row for $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, a quotient, or a standard bridge. Thus the displayed rows exhaust the paper-side $\mathsf{DiffForm}^{\uparrow}$ obligation surface available downstream. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormBoundaryExhaustion downstream export`

136.8 DiffForm root consumer threshold certificate

Definition 136.112 (Differential-form root consumer threshold certificate)**.** *Fix the* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and the* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T *used by Definition 136.1. The differential-form root consumer threshold certificate is the following ten-field surface:*

1. *the ambient* $BHist$ *history carrier for all displayed form rows;*
2. *the* hsame *transport relation on degree, tensor, scalar endpoint, antisymmetry, and source-ledger histories;*
3. *the* Cont *continuation ledger for ordered tensor concatenation and degree bookkeeping;*
4. *the* $\mathsf{ProbeBundle}$ *spine of tangent-probe names;*
5. *the upstream* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source for tangent probes and scalar endpoint data;*
6. *the upstream* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source for multilinear tensor rows;*
7. *the six-coordinate carrier* $\mathsf{DiffFormBHistCarrier}_{M,T}(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$ *of Definition 136.1;*
8. *the displayed classifier* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *of Definition 136.2;*
9. *the wedge and exterior-input rows consisting of tensor concatenation, probe-bundle append, degree bookkeeping, antisymmetry ledgers, scalar endpoint transport, and source-ledger projections;*
10. *the downstream consumer slots for* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, *and* $\mathsf{Symplectic}^{\uparrow}$*.*

The certificate explicitly excludes $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, and any standard bridge.

Theorem 136.113 (Differential-form root consumer carrier readback)**.** *Every consumer admitted by the Definition 136.112 reads exactly the six-coordinate carrier*

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda)$$

and the displayed classifier $\sim_{\mathsf{DiffForm}^\uparrow}$ from Definition 136.2. No downstream consumer slot may read an additional carrier coordinate, replacement classifier, operator law, or standard bridge from the $\mathsf{DiffForm}^\uparrow$ *root.*

Proof. First apply Definition 136.34. It reads the root surface through Definition 136.1 and Definition 136.2, so the only carrier available to a consumer is the displayed six-tuple and the only classifier available is the displayed $\mathsf{DiffForm}^\uparrow$ classifier.

Next apply Definition 136.100. The export threshold states the same carrier and classifier boundary at the public root table, with no seventh carrier coordinate and no replacement classifier.

Finally use Definition 136.165. The exported NameCert surface lists carrier, classifier, transport, wedge, exterior-input, and downstream scope rows, and it names no standard bridge or downstream operator law. Therefore every threshold consumer reads exactly the stated carrier and classifier surface. □

Theorem 136.114 (Differential-form root consumer wedge input readback)**.** *Every accepted wedge or exterior-input consumer row admitted by the Definition 136.112 reads only probe append,* Cont *tensor and degree ledgers, antisymmetry ledgers, scalar endpoint transport, and source ledgers from the displayed* $\mathsf{DiffForm}^\uparrow$ *root surface.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerWedgeInput readback`

Proof. Start with the wedge side. Apply Definition 136.35. It identifies wedge closure with probe-bundle concatenation, visible tensor concatenation through Cont, degree bookkeeping, and the displayed antisymmetry ledger.

Next read the exterior-input boundary through Definition 136.36 and Definition 136.24. These rows allow exterior-input consumers to inspect carried input and output rows, degree shift, wedge-input stability, scalar endpoint transport, and the restricted source ledger.

The export side contributes no broader input. By Definition 136.165 and Definition 136.101, the NameCert root exports the wedge ledger and exterior-input rows only through those displayed histories. Hence accepted wedge and exterior-input consumers read exactly the stated probe, continuation, antisymmetry, scalar, and source surfaces. □

Theorem 136.115 (Differential-form root consumer downstream law separation)**.** *The downstream slots of the Definition 136.112 are input slots only.* $\mathsf{DeRham}^\uparrow$, $\mathsf{Contact}^\uparrow$, *and* $\mathsf{Symplectic}^\uparrow$ *must supply their operator and geometric laws in their own certificate surfaces; the* $\mathsf{DiffForm}^\uparrow$ *root certificate does not project $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or any standard bridge.*

Proof. Apply Definition 136.36 to identify the consumer boundary at the root-unblock surface. It permits downstream chapters to consume degree, wedge, antisymmetry, scalar endpoint, and source-ledger rows, while leaving their laws outside the DiffForm root.

Then apply Definition 136.165 together with the boundary rows Definition 136.61, Definition 136.74, and Definition 136.102. These rows give the same separation at the NameCert and export surfaces.

Thus the consumer threshold is citable by $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, and $\mathsf{Symplectic}^{\uparrow}$ as a root input certificate, but each downstream chapter must close its own nilpotence, closedness, nondegeneracy, nonintegrability, or bridge obligations within its own surface. □

Definition 136.116 (Differential-form root consumer face classifier)**.** *Fix the root export threshold source of Definition 136.99. Its consumer face classifier partitions the exported rows into three consumer faces and one shared root face.*

1. *The* $\mathsf{DeRham}^{\uparrow}$ *face contains the exterior-derivative candidate and input rows, the one-step degree-shift row, the wedge-input stability row, and the downstream exterior-derivative scope row.*
2. *The* $\mathsf{Symplectic}^{\uparrow}$ *face contains the carried form rows, wedge ledgers, antisymmetry ledgers, scalar endpoint rows, and Manifold/TensorProduct source ledgers needed to state a symplectic input surface. It excludes closedness and nondegeneracy.*
3. *The* $\mathsf{Contact}^{\uparrow}$ *face contains the same carried-form, wedge-ledger, and derivative-input surface needed to state contact data. It excludes contact nonintegrability.*
4. *The shared root face consists only of the six-coordinate* $BHist$ *carrier, the displayed* $\mathsf{DiffForm}^{\uparrow}$ *classifier, wedge ledger rows, scalar endpoint rows, and the* $\mathsf{Manifold}^{\uparrow}/\mathsf{TensorProduct}^{\uparrow}$ *source ledgers.*

Rows not named in these four clauses are not exported from the $\mathsf{DiffForm}^{\uparrow}$ *root threshold.*

Theorem 136.117 (Differential-form root consumer face coverage)**.** *Every row exported by Definition 136.99 to* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, or* $\mathsf{Contact}^{\uparrow}$ *belongs to exactly one of the three consumer faces of Definition 136.116, or to the explicitly shared root face. No exported row lies outside those four faces.*

Proof. Project the exported root table with Definition 136.103. It lists the complete threshold surface as the six-coordinate carrier, the displayed classifier, wedge ledgers, the exterior-derivative input boundary, and the downstream consumer separation.

Next apply Definition 136.74. The de Rham consumer receives the exterior-input and degree-shift surface, while the symplectic and contact consumers receive the carried form, wedge, antisymmetry, scalar endpoint, and source-ledger surface needed to state their own data.

Finally apply Definition 136.111. The exhaustion statement says every downstream row is one of the displayed carrier, classifier, wedge, antisymmetry, exterior-input, transport, source-ledger, or downstream rows. Matching these rows against the four clauses of Definition 136.116 gives coverage, and the last sentence of that definition excludes any fifth face. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFace coverage`

Theorem 136.118 (Differential-form root consumer face disjointness)**.** *The consumer-specific laws separated by the Definition 136.116 are disjoint: nilpotence remains* $\mathsf{DeRham}^{\uparrow}$*-specific, closedness and nondegeneracy remain* $\mathsf{Symplectic}^{\uparrow}$*-specific, and contact nonintegrability remains* $\mathsf{Contact}^{\uparrow}$*-specific. Only carried* $BHist$ *rows, classifier rows, wedge ledgers, scalar endpoint rows, and source ledgers may be shared across consumer faces.*

Proof. Start from Definition 136.102. It states that the $\mathsf{DiffForm}^{\uparrow}$ root threshold exports no $d^2 = 0$, closedness, nondegeneracy, or contact nonintegrability row. Those rows therefore cannot belong to the shared root face.

Now use Definition 136.74. That boundary assigns $d^2 = 0$ to the de Rham certificate surface, closedness and nondegeneracy to the symplectic certificate surface, and contact nonintegrability to the contact certificate surface. Hence the laws themselves are pairwise consumer-specific.

The rows allowed to cross consumer faces are exactly the displayed carrier, classifier, wedge, scalar endpoint, and source-ledger rows named in the shared root face of Definition 136.116. Since the law rows are absent from the root export threshold, no consumer-specific law is shared by the classifier. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFace disjointness`

Theorem 136.119 (Differential-form root cross-consumer threshold package)**.** *The definition Definition 136.116, the coverage theorem Definition 136.117, and the disjointness theorem Definition 136.118 form the cross-consumer root threshold package for the* $\mathsf{DiffForm}^{\uparrow}$ *surface consumed by* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFace cross consumer threshold package`

Proof. The classifier supplies the four faces: de Rham, symplectic, contact, and the shared root face.

Coverage then shows that every exported downstream row is assigned to one of those faces. Disjointness shows that the downstream laws remain outside the shared root face and are owned by the corresponding consumer chapters.

Together these two properties make the root export threshold citable as a single cross-consumer package: downstream chapters can reuse the shared root rows without reading nilpotence, closedness, nondegeneracy, or contact nonintegrability back into the $\mathsf{DiffForm}^{\uparrow}$ root. □

Definition 136.120 (Differential-form root consumer face exhaustion source)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *root consumer face exhaustion source is the following ten-field package.*

1. *The traditional target is the finite-degree differential-form surface consumed by* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$.
2. *The BEDC source is* $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.
3. *The carrier is* $\mathsf{DiffFormBHistCarrier}$ *on the displayed manifold, tensor, degree, probe, scalar endpoint, and route histories.*
4. *The classifier is* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *transported by the root classifier rows.*
5. *The stability rows are carrier transport, classifier transport, wedge routing, and exterior-input routing.*
6. *The ledger rows are exactly the degree, probe, tensor, manifold, and* Cont *route ledgers named by the root* $BHist$ *surface.*
7. *The core theorem list is Definition 136.117, Definition 136.118, Definition 136.121, Definition 136.122, and Definition 136.123.*
8. *Exactness means consumer-row exhaustion: every downstream row exported from the root threshold is shared root data or belongs to exactly one of the* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *faces, and no other row is exported.*

9. *The standard bridge field is absent.*
10. *The status is the paper-side obligation surface for the three downstream geometry consumers.*

Theorem 136.121 (Differential-form root consumer face exhaustion)**.** *Every row exported by the root threshold of Definition 136.120 is either shared root data or belongs to exactly one of the* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *consumer faces.*

Proof. Start with the finite face classifier of Definition 136.116. Its four clauses name the de Rham face, the symplectic face, the contact face, and the shared root face.

Apply Definition 136.117. It assigns every row exported to the downstream consumers to one of those four faces and excludes any fifth face.

Finally apply Definition 136.111. The downstream export is exhausted by carrier rows, classifier rows, wedge rows, antisymmetry rows, exterior-derivative input-output rows, transport rows, source-ledger rows, and the displayed downstream boundary rows. Matching those finite rows against the classifier clauses gives the stated shared-or-unique consumer assignment. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFaceExhaustionSource exhaustion`

Theorem 136.122 (Differential-form root consumer boundary exactness)**.** *The exhaustion source of Definition 136.120 admits no* $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, quotient, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFaceExhaustionSource boundary exactness`

Proof. First apply Definition 136.118. It keeps nilpotence, closedness, nondegeneracy, and contact nonintegrability outside the shared root face and assigns those laws only to their downstream consumer surfaces.

Next use Definition 136.102. The root threshold exports carried forms, classifier data, wedge ledgers, and exterior-input rows, but no downstream exterior or geometric law.

It remains to exclude quotient and standard-bridge rows. Apply Definition 136.131; the visible ledger witnesses are exactly the degree, probe, tensor, manifold, wedge, exterior-input, and Cont route histories. Since that exactness statement contains neither quotienting nor a standard bridge, the exhaustion source contains neither row. □

Theorem 136.123 (Differential-form root downstream consumer unblock package)**.** *The definition Definition 136.120, the exhaustion theorem Definition 136.121, and the boundary exactness theorem Definition 136.122 form the citable* $\mathsf{DiffForm}^{\uparrow}$ *root package consumed by* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$.

Proof. The definition supplies the ten named fields of the root consumer surface: traditional target, BEDC source, carrier, classifier, stability rows, ledgers, core theorems, exactness predicate, absent standard bridge, and obligation status.

The exhaustion theorem supplies the positive consumer boundary. Every exported row is shared root data or belongs to the de Rham, symplectic, or contact face.

The boundary exactness theorem supplies the negative boundary. The package exports no nilpotence, closedness, nondegeneracy, contact nonintegrability, quotient, or standard-bridge row. Therefore the three downstream chapters may cite this block as the complete root $BHist$ consumer boundary. □

Definition 136.124 (Differential-form manifold-tensor source envelope)**.** *The differential-form manifold-tensor source envelope is the visible BEDC package*

$$\mathsf{DiffFormSourceEnvelope}_{M,T}.$$

Its rows are exactly the BHist carrier rows of Definition 136.1, the classifier rows of Definition 136.2, the hsame *transport rows used by the classifier, the* Cont *rows used by degree and wedge continuation, the displayed* ProbeBundle *spines, the tangent-probe rows certified by* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$*, and the multilinear tensor rows certified by* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$*. It also contains the wedge probe concatenation ledger of Definition 136.16, the degree-probe alignment rows of Definition 136.29, and the exterior-derivative input rows of Definition 136.57. The envelope has no field for nilpotence, closedness, nondegeneracy, contact nonintegrability, or a standard bridge.*

Theorem 136.125 (Differential-form manifold-tensor source scope)**.** *Every carrier witness and every classifier witness read from* $\mathsf{DiffFormSourceEnvelope}_{M,T}$ *is obtained only from the displayed BHist,* hsame*,* Cont*,* ProbeBundle*,* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *rows of Definition 136.124. No downstream* $\mathsf{DeRham}^\uparrow$*,* $\mathsf{Symplectic}^\uparrow$*, or* $\mathsf{Contact}^\uparrow$ *law is a source for the* $\mathsf{DiffForm}^\uparrow$ *carrier or classifier.*

Proof. Unpack the envelope definition. Its carrier part is exactly the six-coordinate *BHist* row of Definition 136.1; its classifier part is exactly the componentwise comparison of Definition 136.2. Those two rows expose the unary degree history, the visible probe bundle, the tensor row, the scalar endpoint, the antisymmetry ledger, and the source ledger.

Project the manifold and tensor inputs. Tangent probes are read through the displayed ProbeBundle spine and the $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ source. Multilinear tensor rows are read through the same probe spine and the $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ source. The remaining classifier comparisons are hsame and retained Cont rows over the same BHist provenance.

Finally compare this list with the downstream consumer face classifier of Definition 136.116. That classifier places nilpotence, closedness, nondegeneracy, and contact nonintegrability outside the shared root face. Hence carrier and classifier witnesses are sourced only from the envelope rows named above. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormManifoldTensorSource scope`

Theorem 136.126 (Differential-form wedge continuation input closure)**.** *Let* ω *and* η *be carried rows in* $\mathsf{DiffFormSourceEnvelope}_{M,T}$*. If their admitted wedge row is obtained through the visible continuation and probe append data, then the wedge input and output rows remain inside* $\mathsf{DiffFormSourceEnvelope}_{M,T}$*. The retained rows are the wedge probe concatenation ledger, the coverage row for its output probe bundle, and the degree-probe alignment row for the wedge output.*

Proof. Start with the two carried input rows. By Definition 136.124, each input row already carries its *BHist* carrier witness, classifier witness, probe bundle, Manifold source row, TensorProduct source row, and degree-probe alignment row.

Apply Definition 136.16. The wedge input is recorded by the two input ledgers, the two input probe spines, the output probe spine, and the tensor-concatenation provenance row. These are all rows named in the source envelope.

Next apply Definition 136.17. It identifies the output probe bundle as the visible append of the two input bundles and shows that every output probe comes from one of the two input spines. Thus the manifold and tensor source readings remain the same displayed source readings.

Finally apply Definition 136.32. The output degree row and output probe bundle are aligned by the wedge continuation row. Repack the two input rows, the output append row, the tensor provenance, and the aligned degree-probe row into Definition 136.124. The wedge continuation therefore stays inside the source envelope. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeContinuation input closure`

Theorem 136.127 (Differential-form exterior-derivative consumer coverage)**.** *Every witness in* $\mathsf{DiffFormSourceEnvelope}_{M,T}$ *that satisfies the exterior-derivative input boundary supplies the downstream exterior-derivative consumer face for* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$*. The supplied face is exactly the carrier, classifier, degree-shift, wedge-input, endpoint-transport, source-ledger, and downstream boundary surface already listed for those consumers, with no downstream law imported into* $\mathsf{DiffForm}^{\uparrow}$*.*

Proof. Begin with an envelope witness satisfying the exterior-derivative input boundary. The input boundary row is the visible package of Definition 136.57, and its downstream visible form is the package of Definition 136.68.

Use Definition 136.105. It repacks the source carrier, output carrier, successor-degree continuation, transported probe and tensor rows, scalar endpoint comparison, and restricted source ledger into $\mathsf{DiffFormDownstreamBoundary}_{M,T}$.

Now combine the downstream boundary rows. The exterior-derivative scope row Definition 136.61 names the carrier, classifier, degree-raise candidate, endpoint transport, and exterior-algebra substrate consumed downstream. The consumer boundary theorem Definition 136.74 identifies $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ as the downstream consumers of that surface.

Finally apply Definition 136.111. It says the downstream export is exhausted by the carrier, classifier, wedge, antisymmetry, exterior-input, transport, source-ledger, and downstream rows. The consumer face classifier of Definition 136.116 assigns those rows to the de Rham, symplectic, contact, or shared root faces. Thus every exterior-input envelope witness supplies the advertised downstream consumer face and still leaves nilpotence, closedness, nondegeneracy, and contact nonintegrability to the consuming chapters. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormSourceEnvelope exterior derivative consumer coverage`
↪ variant `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivative consumer coverage`

Definition 136.128 (Differential-form root BHist NameCert surface)**.** *The root BHist NameCert surface for* $\mathsf{DiffForm}^{\uparrow}$ *is the ten-field obligation surface*

$$\mathsf{DiffFormRootBHistNameCert}_{M,T}.$$

Its fields are:

1. *traditional target: finite-degree differential forms;*
2. *BEDC source: BHist,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*;*
3. *carrier:* $\mathsf{DiffFormBHistCarrier}_{M,T}$*;*
4. *classifier:* $\sim_{\mathsf{DiffForm}^{\uparrow}}$*;*
5. *stability rows: carrier transport, degree-probe alignment transport, wedge degree routing, and exterior-derivative input routing;*
6. *ledger: degree histories, probe bundles, tensor rows, manifold rows, and* Cont *route histories;*

7. *core theorem list: Definition 136.129, Definition 136.130, Definition 136.131, and Definition 136.132;*
8. *exactness boundary: the exhaustion rows of Definition 136.108;*
9. *standard bridge: absent;*
10. *status:* (obligation, unformalized).

The surface contains no row for $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, a de Rham quotient, or a standard-object bridge.

Theorem 136.129 (Differential-form root BHist carrier obligation)**.** *Every admissible root row of Definition 136.81 supplies the carrier* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, the degree-probe alignment row, the manifold source row* $\mathsf{NameCert}_{\mathsf{Manifold}\uparrow}$*, and the tensor source row* $\mathsf{NameCert}_{\mathsf{TensorProduct}\uparrow}$ *displayed by Definition 136.128.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBHistCarrier obligation`

Proof. Unpack Definition 136.81. Its carrier coordinate is the carried form row of Definition 136.1, and its source coordinates are exactly the displayed manifold and tensor NameCert rows.

Apply Definition 136.109. The boundary exhaustion theorem identifies the visible carrier and classifier rows with the root source rows and excludes tangent-space, alternating-map, quotient, or external classifier carriers.

It remains to bind the degree and probe entries. Apply Definition 136.30; the carried row supplies the unary degree history and the ProbeBundle spine over the same tensor input reading. Repack these projections as the carrier field of Definition 136.128. □

Theorem 136.130 (Differential-form root BHist classifier stability obligation)**.** *Classifier transport along* $\sim_{\mathsf{DiffForm}\uparrow}$ *preserves the carrier, degree-probe alignment, wedge routing, and exterior-derivative input rows of Definition 136.128.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBHistClassifier stability obligation`

Proof. First transport the carried row. Apply Definition 136.10 to the displayed histories; the transported endpoint remains a carried $\mathsf{DiffFormBHistCarrier}_{M,T}$ row over the same BEDC source surface.

Next transport the degree-probe alignment. Apply Definition 136.31; the hsame degree comparison and transported ProbeBundle spine rebuild the aligned row at the target endpoint.

Finally read the operation boundary. Apply Definition 136.106. The common wedge and exterior-input rows are exactly the carried form, unary degree, probe, tensor, scalar endpoint, antisymmetry, source, and classifier rows already transported. Hence classifier stability preserves all operation routing named by the root surface. □

Theorem 136.131 (Differential-form root BHist ledger exactness obligation)**.** *The visible ledger witnesses of Definition 136.128 are exactly degree histories, probe bundles, tensor source rows, manifold source rows, wedge rows, exterior-derivative input rows, and* Cont *route histories. No closedness, nondegeneracy, nilpotence, contact nonintegrability, quotient, or standard bridge row is a ledger witness of the root surface.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBHistNameCert semantic name certificate`

Proof. Begin with the carrier side. Apply Definition 136.109; it restricts the carrier and classifier ledger entries to the displayed root rows over $BHist$, hsame, Cont, ProbeBundle, and the two source NameCert rows.

Now read the operation side. Apply Definition 136.110; the wedge, antisymmetry, degree, and exterior-derivative entries are exactly the visible ledger rows named by the root obligation and input-output exactness statements.

Finally apply Definition 136.111. Downstream export is exhausted by the same carrier, classifier, wedge, antisymmetry, exterior-input, transport, and source-ledger rows. Since none of the cited exhaustion theorems contains a row for closedness, nondegeneracy, nilpotence, contact nonintegrability, quotienting, or a standard bridge, the displayed ledger is exact. □

Theorem 136.132 (Differential-form root BHist downstream boundary)**.** *The chapters* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *consume* $\mathsf{DiffForm}^{\uparrow}$ *only through the root surface of Definition 136.128. Their downstream reads factor through the carried row, classifier row, degree-probe alignment, wedge routing, exterior-input routing, source NameCert rows, and* Cont *route histories named there.*

Proof. Start from an arbitrary downstream read. Apply Definition 136.107; every read by $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, or $\mathsf{Contact}^{\uparrow}$ factors through the carrier and classifier rows, exterior-input rows, wedge rows, antisymmetry rows, and downstream scope rows listed in the boundary package.

For exterior input, apply Definition 136.105. It repacks every accepted input row into the root-visible carrier, output carrier, successor-degree continuation, transported probe bundle, transported tensor row, scalar endpoint comparison, and source ledger.

Finally apply Definition 136.85. The admissible root row supplies no $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or bridge row, so those obligations cannot be projected from $\mathsf{DiffForm}^{\uparrow}$. The only remaining reads are precisely the fields of Definition 136.128. □

136.9 DiffForm root degree-classifier package

Definition 136.133 (DiffForm root degree-classifier package)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T*. The root degree-classifier package for* $\mathsf{DiffForm}^{\uparrow}$ *is the citable surface*

$$\mathsf{DiffFormRootDegreeClassifier}_{M,T}.$$

Its BEDC source is the tuple consisting of $BHist$, hsame, Cont, UnaryHistory, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

Its carrier is the pair of the concrete carried row $\mathsf{DiffFormBHistCarrier}_{M,T}$ *and the alignment witness* $\mathsf{DegreeProbeAligned}(d, \mathcal{P})$ *for the displayed unary degree row* d *and visible probe bundle* $\mathcal{P}$*. Its classifier is equality of the transported unary degree row and equality of the transported visible probe arity under* $\sim_{\mathsf{DiffForm}^{\uparrow}}$*; the remaining transported carrier rows are read only through that same* $BHist$ *classifier surface.*

Its ledger consists exactly of the displayed degree history d*, the probe bundle* $\mathcal{P}$*, the tensor row* τ*, the scalar endpoint* σ*, the antisymmetry row* α*, the source ledger* λ*, and the continuation route that records the degree endpoint used by wedge and exterior-input routing. The package contains no row for nilpotence, closedness, nondegeneracy, contact nonintegrability, quotienting, or a standard bridge.*

Theorem 136.134 (DiffForm root degree-classifier coverage)**.** *Every root* $BHist$ *surface row of Definition 136.128 supplies the package* $\mathsf{DiffFormRootDegreeClassifier}_{M,T}$*. In particular, the carrier*

part is the concrete $\mathsf{DiffFormBHistCarrier}_{M,T}$ *row, the arity part is the visible* ProbeBundle *spine aligned with its unary degree row, and the ledger part is exhausted by the displayed degree, probe, tensor, scalar endpoint, antisymmetry, source, and continuation rows.*

Proof. First read the carrier field of the root $BHist$ surface. Apply Definition 136.129. It supplies the concrete carrier row together with the manifold and tensor source rows named by the package.

Next bind the degree and arity interface. Apply Definition 136.30; the carried row exposes the unary degree history and the visible probe bundle as an aligned pair, with no second arity counter outside the displayed bundle.

Finally read the ledger. Apply Definition 136.131. Its exactness statement lists precisely the degree histories, probe bundles, tensor source rows, manifold source rows, wedge rows, exterior-input rows, and Cont route histories available at the root surface. Repacking those rows gives the announced degree-classifier package. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDegreeClassifier coverage`

Theorem 136.135 (DiffForm root degree-classifier transport)**.** *Transport along* $\sim_{\mathsf{DiffForm}^\uparrow}$ *preserves* $\mathsf{DiffFormRootDegreeClassifier}_{M,T}$*. If a carried row with degree* d *and probe bundle* $\mathcal{P}$ *supplies the package, then every* $\sim_{\mathsf{DiffForm}^\uparrow}$*-equivalent carried row supplies the transported package with the transported unary degree row and the same visible probe arity.*

Proof. Begin with classifier stability for the root surface. Apply Definition 136.130. The transported endpoint remains in the concrete carrier surface and keeps the root classifier rows available.

Now transport the aligned pair. Apply Definition 136.31. The degree comparison carried by hsame and the transported probe bundle rebuild DegreeProbeAligned at the target endpoint.

It remains to identify the arity read. Apply Definition 136.12. The visible number of probe slots is preserved by the transported bundle, so the classifier interface is exactly equality of transported degree rows together with equality of transported visible probe arity under $\sim_{\mathsf{DiffForm}^\uparrow}$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDegreeClassifier transport`

Theorem 136.136 (DiffForm root degree-classifier downstream consumption)**.** *The downstream chapters* $\mathsf{DeRham}^\uparrow$*,* $\mathsf{Symplectic}^\uparrow$*, and* $\mathsf{Contact}^\uparrow$ *consume the* $\mathsf{DiffForm}^\uparrow$ *degree and arity data only through* $\mathsf{DiffFormRootDegreeClassifier}_{M,T}$*. Their reads of degree rows and probe-slot arity factor through the root* $BHist$ *surface, the aligned degree-probe pair, and the root consumer face classifier; they do not read nilpotence, closedness, nondegeneracy, contact nonintegrability, quotienting, or a standard bridge from this package.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDegreeClassifier downstream consumption`

Proof. Start from a downstream read by one of the three consumer chapters. Apply Definition 136.132. It factors every such read through the root carrier row, classifier row, degree-probe alignment, wedge routing, exterior-input routing, source NameCert rows, and Cont route histories.

Next apply Definition 136.117. The consumer face classifier assigns every exported row to the de Rham, symplectic, contact, or shared root face, with no row outside those faces.

The shared degree and arity interface is therefore exactly the package surface: transported unary degree equality and transported visible probe-arity equality under $\sim_{\mathsf{DiffForm}^\uparrow}$. The consumer-specific laws remain in their own faces, so this package carries no nilpotence, closedness, nondegeneracy, contact nonintegrability, quotienting, or standard bridge claim. □

Definition 136.137 (Differential-form root degree-probe consumer face)**.** *The root degree-probe consumer face of* $\mathsf{DiffForm}^{\uparrow}$ *is the subface of Definition 136.128 whose rows are exactly:*

1. *the unary degree history of a carried* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *row;*
2. *the visible* $\mathsf{ProbeBundle}$ *spine carried by the same row;*
3. *the degree-probe alignment transport row of Definition 136.31;*
4. *the wedge degree routing row exposed by the wedge ledger;*
5. *the exterior-derivative successor-degree input routing row of Definition 136.105;*
6. *the displayed* $BHist$*,* hsame*,* Cont*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source rows.*

The face contains no row for nilpotence, closedness, nondegeneracy, contact nonintegrability, a de Rham quotient, or a standard bridge. It is a consumer face for downstream reads of degree and probe data only; all consumer-specific laws remain outside $\mathsf{DiffForm}^{\uparrow}$*.*

Theorem 136.138 (Differential-form root degree-probe face coverage)**.** *Every* $\mathsf{DiffForm}^{\uparrow}$ *row read by* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, or* $\mathsf{Contact}^{\uparrow}$ *for degree bookkeeping, visible probe bookkeeping, degree-probe alignment, wedge degree routing, or exterior-derivative successor-degree input routing belongs to Definition 136.137 or to the shared root rows named by Definition 136.116.*

Proof. First read the root carrier surface. Apply Definition 136.129. It identifies the carried degree history, the visible $\mathsf{ProbeBundle}$ spine, and the ManifoldUp and TensorProductUp source rows as fields of the root $BHist$ NameCert surface.

Next isolate the degree-probe row. Apply Definition 136.30. The carried form row refines to the unary degree history and the matching probe bundle over the same tensor input reading, so the downstream degree/probe read is one of the rows named in Definition 136.137.

For operation routing, use Definition 136.105 and Definition 136.132. These theorems place the successor-degree exterior input, wedge degree routing, carried row, classifier row, source rows, and Cont histories on the root-visible surface.

Finally apply Definition 136.117. Every downstream exported row is in a consumer face or in the shared root face. Intersecting that coverage with the rows listed in Definition 136.137 gives the stated coverage for degree and probe reads, with no residual row outside the two named faces. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDegreeProbeFace coverage`

Theorem 136.139 (Differential-form root degree-probe face stability)**.** *Classifier transport along* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *preserves the root degree-probe consumer face. In particular, transported downstream reads of degree history,* $\mathsf{ProbeBundle}$ *spine, degree-probe alignment, wedge degree routing, exterior-derivative successor-degree input routing, and the displayed source rows remain inside Definition 136.137.*

Proof. Transport the root surface first. Apply Definition 136.130; carrier transport preserves the degree history, probe bundle, wedge routing, exterior-input routing, and displayed source rows of Definition 136.128.

Now transport the aligned subrow. Apply Definition 136.31. The classifier comparison carries the hsame degree comparison and the transported $\mathsf{ProbeBundle}$ spine to the target endpoint, rebuilding the aligned degree/probe row there.

It remains to keep consumer laws out of the transported face. Apply Definition 136.118. Nilpotence, closedness, nondegeneracy, and contact nonintegrability are consumer-specific law rows, and the root degree-probe face contains no quotient or standard bridge row. Thus classifier transport preserves the face without enlarging it. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootDegreeProbeFace stability`

Theorem 136.140 (Differential-form root degree-probe downstream factorization)**.** *Every read by* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, or* $\mathsf{Contact}^{\uparrow}$ *that uses* $\mathsf{DiffForm}^{\uparrow}$ *degree or probe data factors first through Definition 136.137, and only after that through the law rows owned by the downstream chapter.*

Proof. Start with an arbitrary downstream read of $\mathsf{DiffForm}^{\uparrow}$ degree or probe data. Apply Definition 136.132; downstream access to $\mathsf{DiffForm}^{\uparrow}$ factors through the carried row, classifier row, degree-probe alignment, wedge routing, exterior-input routing, source NameCert rows, and Cont route histories.

Refine the degree/probe component with Definition 136.30 and transport it with Definition 136.31. The consumer therefore reads the unary degree history and the $\mathsf{ProbeBundle}$ spine through the face defined in Definition 136.137, not through a separate downstream carrier.

For exterior-derivative input, apply Definition 136.105. The successor-degree input row is repacked from the same carried row, transported probe bundle, tensor row, scalar endpoint comparison, and source ledger. For wedge routing, use the wedge degree routing already listed by the root $BHist$ downstream boundary.

Finally apply Definition 136.117 and Definition 136.118. Coverage places all downstream reads in the named consumer or shared root faces, while disjointness keeps nilpotence, closedness, nondegeneracy, contact nonintegrability, quotient data, and bridge data outside this face. Hence $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ first factor through the degree-probe consumer face and then state their own laws on their own surfaces. □

136.10 DiffForm root consumer-readable package

Definition 136.141 (DiffForm root consumer-readable package)**.** *The root consumer-readable package for* $\mathsf{DiffForm}^{\uparrow}$ *is the ten-field interface*

$$\mathsf{DiffFormRootConsumerPackage}_{M,T}.$$

Its fields are:

1. *traditional target: finite-degree differential forms as consumed by downstream exterior-calculus chapters;*
2. *BEDC source:* $BHist$*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*;*
3. *carrier: the six-coordinate surface consisting of the carried form row, degree history, probe bundle, tensor source row, manifold source row, and* Cont *route history;*
4. *classifier: the displayed relation* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *on the carried root surface;*
5. *stability: carrier transport, classifier stability, degree-probe alignment transport, wedge routing, and exterior-input routing;*
6. *ledger: exactly the histories and source rows named in the carrier, together with the wedge and exterior-input boundary rows;*
7. *core theorem list: Definition 136.129, Definition 136.130, Definition 136.131, and Definition 136.132;*
8. *exactness boundary: no consumer receives a row outside the boundary exhausted by Definition 136.128;*

9. *absent bridge: no standard bridge or quotient bridge is exported;*

10. *status:* (obligation, unformalized).

The package is only a reader-facing projection of Definition 136.128; it introduces no additional exterior law.

Theorem 136.142 (DiffForm root consumer carrier projection)**.** *Every consumer read of Definition 136.141 projects only the six-coordinate carrier consisting of the carried form row, degree history, probe bundle, tensor source row, manifold source row, and* Cont *route history, together with the displayed classifier* $\sim_{\mathsf{DiffForm}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerPackage carrier projection`

Proof. Unpack the package definition. Its carrier and classifier fields are copied from Definition 136.128, so a consumer read has no independent carrier coordinate to inspect.

Apply Definition 136.129. The root obligation supplies exactly the carried form row, the aligned degree and probe entries, and the two source NameCert rows named by the package.

It remains to exclude hidden classifier data. Apply Definition 136.130; classifier transport preserves the displayed root rows and does not create a tangent, quotient, standard-object, or external classifier coordinate. Hence the consumer projection is exactly the six-coordinate carrier plus $\sim_{\mathsf{DiffForm}^{\uparrow}}$. □

Theorem 136.143 (DiffForm root consumer operation routing)**.** *Every wedge read and every exterior-input read through Definition 136.141 routes only through the root wedge boundary row and the root exterior-input boundary row already listed by Definition 136.128.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerPackage operation routing`

Proof. Begin with a wedge read. The package ledger names the wedge row only as part of the root surface, so the read is routed through the wedge boundary component already exposed by Definition 136.128.

For an exterior-input read, use the same ledger field. The exterior input is the root input row carried by the displayed $BHist$, hsame, Cont, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ source data.

Apply Definition 136.131. Its ledger exactness statement says that the visible operation witnesses are only degree histories, probe bundles, tensor and manifold source rows, wedge rows, exterior-derivative input rows, and Cont route histories. Therefore the consumer package has no separate operation-routing surface. □

Theorem 136.144 (DiffForm root consumer NameCert boundary)**.** *The downstream chapters* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$ *may cite Definition 136.141 as the root* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *boundary. This citation supplies the carrier, classifier, stability, ledger, wedge-routing, exterior-input, and source NameCert rows named there, but supplies no* $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, quotient, or standard bridge row.*

Proof. Start from a downstream citation by $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, or $\mathsf{Contact}^{\uparrow}$. Apply Definition 136.132; every such read factors through the carried row, classifier row, degree-probe alignment, wedge routing, exterior-input routing, source NameCert rows, and Cont route histories of the root surface.

Now replace the root surface by the consumer package. The definition of Definition 136.141 lists exactly those fields and names the same core theorem list, so the downstream citation is a readable package citation rather than a new source of structure.

Finally apply Definition 136.131. Since the root ledger has no row for $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, quotienting, or a standard bridge, none of those obligations can be received from the package. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerPackage namecert boundary`

136.11 DiffForm root consumer entry

Definition 136.145 (DiffForm root consumer entry source)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *root consumer entry source is the tuple*

$$\mathsf{DiffFormRootConsumerEntry}_{M,T}.$$

Its entries are the BHist carrier and classifier rows of Definition 136.128, the root consumer carrier readback of Definition 136.113, the wedge input readback of Definition 136.114, the exterior-derivative input row covered by Definition 136.127, and the downstream consumer boundary packaged by Definition 136.119. The entry source contains no row for $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, or a standard bridge.*

Theorem 136.146 (DiffForm root consumer entry carrier)**.** *Every carrier or classifier read admitted by Definition 136.145 factors through the root BHist surface of Definition 136.128. The only carrier projected to a downstream consumer is* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, and the only classifier projected to a downstream consumer is the displayed* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *classifier.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerEntry carrier`

Proof. Start with the entry tuple. Its carrier and classifier coordinates are defined to be the root *BHist* rows of Definition 136.128.

Apply Definition 136.129. It projects the carried form row, degree-probe alignment row, manifold source row, and tensor source row from the displayed root surface.

Now apply Definition 136.113. That readback theorem states that every threshold consumer reads exactly the six-coordinate carrier and displayed classifier, with no additional carrier coordinate, replacement classifier, operator law, or standard bridge. The carrier entry therefore factors through precisely the advertised root surface. □

Theorem 136.147 (DiffForm root consumer entry wedge)**.** *Every wedge input read admitted by Definition 136.145 is covered by the root consumer threshold certificate. Its visible data are probe append,* Cont *tensor and degree ledgers, antisymmetry ledgers, scalar endpoint transport, and source-ledger projections from the* $\mathsf{DiffForm}^{\uparrow}$ *root.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerEntry wedge`

Proof. Read the wedge coordinate of the entry source. By definition it is the wedge input readback of Definition 136.114.

Next use the cross-consumer threshold package Definition 136.119. The package assigns exported downstream rows to de Rham, symplectic, contact, or shared root faces while keeping downstream laws outside the shared root face.

Combining these two rows gives exactly the visible wedge surface: probe append, Cont tensor and degree ledgers, antisymmetry ledgers, scalar endpoint transport, and source-ledger projections. No wedge input read can project a de Rham, symplectic, or contact law from the root entry. □

Theorem 136.148 (DiffForm root consumer entry derivative)**.** *Every exterior-derivative input read admitted by Definition 136.145 routes through the manifold-tensor source envelope of Definition 136.124. The route supplies carrier, classifier, degree-shift, wedge-input, endpoint-transport, source-ledger, and downstream boundary data, but it does not assert $d^2 = 0$.*

Proof. Begin with the exterior-derivative entry. It is the input row covered by Definition 136.127.

Use Definition 136.125 to identify the source envelope. Carrier and classifier witnesses in that envelope come only from the displayed $BHist$, hsame, Cont, ProbeBundle, $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ rows.

Now apply Definition 136.127. The coverage theorem routes the accepted exterior-derivative input through the carrier, classifier, degree-shift, wedge-input, endpoint-transport, source-ledger, and downstream boundary surface. Its statement leaves nilpotence to the downstream chapter, so the entry source supplies input data without claiming $d^2 = 0$. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerEntry derivative boundary`

Theorem 136.149 (DiffForm root consumer entry threshold)**.** *The carrier, wedge, and exterior-derivative projections of Definition 136.145 form the root entry threshold for* $\mathsf{DeRham}^\uparrow$, $\mathsf{Symplectic}^\uparrow$, *and* $\mathsf{Contact}^\uparrow$. *Downstream consumers may cite this entry threshold for their shared* $\mathsf{DiffForm}^\uparrow$ *input surface, while their nilpotence, closedness, nondegeneracy, contact nonintegrability, and bridge obligations remain outside the root entry.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerEntry threshold`

Proof. First use Definition 136.146. It supplies the shared carrier and classifier surface from the root $BHist$ NameCert rows.

Next use Definition 136.147. It supplies the shared wedge input surface through the root consumer threshold certificate.

Then use Definition 136.148. It supplies the exterior-derivative input route through the manifold-tensor source envelope, without importing $d^2 = 0$ into $\mathsf{DiffForm}^\uparrow$.

Finally apply Definition 136.119. The package places the de Rham, symplectic, and contact laws in their consumer faces and keeps the shared carrier, classifier, wedge, scalar endpoint, and source-ledger rows in the root face. The three projections therefore assemble as the advertised root entry threshold. □

Definition 136.150 (Differential-form root boundary consumer package)**.** *The differential-form root boundary consumer package is the tuple*

$$\mathsf{DiffFormRootBoundaryConsumer}_{M,T}.$$

It has ten fields.

1. *The source field is the pair* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *already exposed by the root $BHist$ surface.*
2. *The carrier field is* $\mathsf{DiffFormBHistCarrier}_{M,T}$.
3. *The classifier field is the displayed* $\sim_{\mathsf{DiffForm}^\uparrow}$ *classifier.*

4. *The carrier-transport stability field is the* hsame *transport row of Definition 136.10.*
5. *The classifier-stability field is the transported classifier row of Definition 136.11.*
6. *The wedge-transport field is the wedge and derivative boundary compatibility row of Definition 136.106.*
7. *The exterior-input transport field is the root closure row of Definition 136.105.*
8. *The ledger field consists of* $\mathsf{DiffFormDownstreamBoundary}_{M,T}$ *from Definition 136.104 together with the entry rows of Definition 136.145.*
9. *The exactness field says that these fields exhaust the shared rows consumed by* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *and* $\mathsf{Contact}^{\uparrow}$.
10. *The bridge-status field contains no standard bridge row and records the package as a proof target.*

Theorem 136.151 (Differential-form root boundary consumer carrier scope)**.** *Every carrier or classifier read admitted by* $\mathsf{DiffFormRootBoundaryConsumer}_{M,T}$ *factors through the root* $BHist$ *surface of Definition 136.128, the entry carrier theorem Definition 136.146, and the carrier coverage theorem Definition 136.109. In particular, the package exposes no carrier beyond* $\mathsf{DiffFormBHistCarrier}_{M,T}$ *and no classifier beyond* $\sim_{\mathsf{DiffForm}^{\uparrow}}$.

Proof. Read the carrier and classifier fields of $\mathsf{DiffFormRootBoundaryConsumer}_{M,T}$. By definition, they are the carrier and classifier fields of Definition 136.128.

Apply Definition 136.146. It states that the root consumer entry projects exactly the root $BHist$ carrier and the displayed $\mathsf{DiffForm}^{\uparrow}$ classifier, with no replacement carrier and no downstream law.

Apply Definition 136.109. The coverage theorem identifies the carrier and classifier rows visible to downstream consumers with the displayed source rows. Therefore every carrier or classifier read of the package factors through the named root surface and through the two carrier-scope theorems. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBoundaryConsumer carrier scope`

Theorem 136.152 (Differential-form root boundary consumer classifier scope)**.** *Componentwise* hsame *transport preserves* $\mathsf{DiffFormRootBoundaryConsumer}_{M,T}$. *The carrier field is preserved by Definition 136.10, the classifier field by Definition 136.11, the exterior-input field by Definition 136.105, and the shared wedge and derivative boundary field by Definition 136.106.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBoundaryConsumer classifier scope`

Proof. Start with componentwise hsame data on the displayed source histories. For the carrier field, apply Definition 136.10. It transports the carried form row without changing the root source.

For the classifier field, apply Definition 136.11. It gives stability of $\sim_{\mathsf{DiffForm}^{\uparrow}}$ under the same componentwise histories and keeps the classifier inside the root surface.

For the exterior-input field, use Definition 136.105. That theorem repacks the accepted exterior-input rows into the downstream boundary package after the same transported carrier and classifier rows have been fixed.

For the wedge and derivative boundary field, apply Definition 136.106. It identifies the common transported rows used by wedge admission and exterior-input admission. The ledger and exactness fields are defined from these transported rows, so the whole package is preserved. □

Theorem 136.153 (Differential-form root boundary consumer exactness)**.** *Every* $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, *or* $\mathsf{Contact}^{\uparrow}$ *read of the* $\mathsf{DiffForm}^{\uparrow}$ *shared root factors through* $\mathsf{DiffFormRootBoundaryConsumer}_{M,T}$. *No such read can project* $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, nilpotence, or a bridge row from this package.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootBoundaryConsumer exactness`

Proof. First route carrier and classifier reads through Definition 136.151. This accounts for the shared carried-form row, the displayed $\mathsf{DiffForm}^{\uparrow}$ classifier, and the source certificates visible from $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

Next route exterior-input reads through Definition 136.105. The theorem places source carrier, output carrier, successor-degree continuation, transported probe rows, tensor rows, scalar endpoint comparison, and source ledger inside $\mathsf{DiffFormDownstreamBoundary}_{M,T}$.

Now route wedge and mixed wedge-derivative reads through Definition 136.106. The common rows are exactly the carried form, unary degree row, wedge degree ledger, one-step exterior continuation ledger, transported probe, tensor, scalar endpoint, antisymmetry, source rows, and their classifier comparisons.

Finally apply Definition 136.107 together with Definition 136.149. These theorems name $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$ as the downstream consumers of the shared boundary and root-entry rows. Since the package contains exactly those ledgers and its bridge-status field contains no standard bridge, every downstream read factors through the package and none can project nilpotence, closedness, nondegeneracy, contact nonintegrability, or bridge structure from $\mathsf{DiffForm}^{\uparrow}$. □

136.12 DiffForm root consumer factorization

Definition 136.154 (DiffForm root consumer factorization surface)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *root consumer factorization surface is the tuple*

$$\mathsf{DiffFormRootConsumerFactor}_{M,T}.$$

Its coordinates are the entry source of Definition 136.145, the carrier row of Definition 136.146, the wedge row of Definition 136.147, the exterior-derivative row of Definition 136.148, and the threshold row of Definition 136.149. The surface is a consumer-facing route table, not a downstream law table: it contains no row for $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, quotienting, or a standard bridge.*

Theorem 136.155 (DiffForm root carrier factorization)**.** *Every downstream carrier or classifier read from* $\mathsf{DiffForm}^{\uparrow}$ *factors through Definition 136.154. The only carrier route is the root* $BHist$ *carrier* $\mathsf{DiffFormBHistCarrier}_{M,T}$*, and the only classifier route is the displayed* $\sim_{\mathsf{DiffForm}^{\uparrow}}$ *classifier bound by the root* $BHist$ *surface.*

Proof. Start with an arbitrary carrier or classifier read by $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, or $\mathsf{Contact}^{\uparrow}$. Apply Definition 136.132. It places every downstream read through the carried row, classifier row, degree-probe alignment, wedge routing, exterior-input routing, source NameCert rows, and Cont histories named by the root $BHist$ surface.

Now apply Definition 136.146. This entry row identifies the consumer carrier coordinate with $\mathsf{DiffFormBHistCarrier}_{M,T}$ and the consumer classifier coordinate with the displayed $\sim_{\mathsf{DiffForm}^{\uparrow}}$ classifier, with no replacement carrier or classifier.

Finally read these two coordinates inside Definition 136.154. Since the factorization surface contains the entry source and the carrier entry row, the downstream carrier and classifier read factors through exactly that surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFactorization carrier route closure`

Theorem 136.156 (DiffForm wedge and derivative factorization)**.** *Every downstream wedge read or exterior-derivative input read from* $\mathsf{DiffForm}^{\uparrow}$ *factors through Definition 136.154. Wedge reads use the wedge-entry row, and exterior-derivative reads use the exterior-derivative entry row; neither route projects a downstream law from the* $\mathsf{DiffForm}^{\uparrow}$ *root.*

Proof. First consider a downstream wedge read. Apply Definition 136.147. It routes accepted wedge input through probe append, Cont tensor and degree ledgers, antisymmetry ledgers, scalar endpoint transport, and source-ledger projections from the $\mathsf{DiffForm}^{\uparrow}$ root.

Next consider an exterior-derivative input read. Apply Definition 136.148. It routes the input through the manifold-tensor source envelope, supplying carrier, classifier, degree-shift, wedge-input, endpoint-transport, source-ledger, and downstream boundary data while leaving $d^2 = 0$ outside the root.

Finally apply Definition 136.119. The package assigns the de Rham, symplectic, and contact laws to their consumer faces and keeps the shared carrier, classifier, wedge, scalar endpoint, and source-ledger rows in the root face. Since Definition 136.154 contains both entry rows and the threshold row, every wedge or exterior-input read factors through the shared surface. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerFactorization wedge derivative`

Theorem 136.157 (DiffForm threshold consumer factorization)**.** *The three consumer faces* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Symplectic}^{\uparrow}$*, and* $\mathsf{Contact}^{\uparrow}$ *may cite only Definition 136.154 for their shared* $\mathsf{DiffForm}^{\uparrow}$ *input. Nilpotence, closedness, nondegeneracy, and contact nonintegrability must be supplied in the corresponding downstream chapters and are not projected by the* $\mathsf{DiffForm}^{\uparrow}$ *root.*

Proof. Begin with the shared input side. Apply Definition 136.149. It assembles the carrier, wedge, and exterior-derivative projections as the root entry threshold for $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Symplectic}^{\uparrow}$, and $\mathsf{Contact}^{\uparrow}$.

Now apply Definition 136.119. It separates the shared root rows from the de Rham, symplectic, and contact law rows, so consumer chapters receive only the common input surface from $\mathsf{DiffForm}^{\uparrow}$.

Finally apply Definition 136.132. The root $BHist$ boundary excludes $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, and bridge rows from the $\mathsf{DiffForm}^{\uparrow}$ projection. Thus the three consumer faces may cite the shared factorization surface for input, and their own chapters must supply the corresponding downstream laws. □

136.13 DiffForm root consumer export packet

Definition 136.158 (DiffForm root consumer export packet)**.** *The* $\mathsf{DiffForm}^{\uparrow}$ *root consumer export packet is the paper-side union*

$$\mathsf{DiffFormRootConsumerExport}_{M,T}.$$

Its positive surface consists exactly of the root-unblock source surface of Definition 136.33, the carrier and classifier scope theorem Definition 136.34, the wedge and Cont *closure theorem Definition 136.35, the exterior-input consumption theorem Definition 136.36, the consumer face classifier*

of Definition 136.116, the face exhaustion theorem Definition 136.121, and the consumer-readable package of Definition 136.141 with its boundary theorem Definition 136.144. The packet contains no row outside these cited root-unblock, face-classifier, face-exhaustion, and readable-package surfaces.

Theorem 136.159 (DiffForm root consumer export coverage)**.** *Every* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Contact}^{\uparrow}$*, or* $\mathsf{Symplectic}^{\uparrow}$ *consumer row admitted at the* $\mathsf{DiffForm}^{\uparrow}$ *root factors through Definition 136.158.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerExport coverage`

Proof. Start with a consumer row admitted at the $\mathsf{DiffForm}^{\uparrow}$ root. By the source surface in Definition 136.33, such a row may read only the displayed $BHist$, hsame, Cont, probe-bundle, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, carrier, classifier, wedge, and exterior-input coordinates.

Apply Definition 136.36. It restricts $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, and $\mathsf{Symplectic}^{\uparrow}$ consumers to the degree, wedge, antisymmetry, scalar endpoint, and source-ledger rows exposed by the root-unblock surface.

Now apply Definition 136.121. Every row exported by the root threshold is shared root data or belongs to exactly one of the de Rham, symplectic, and contact faces.

Finally use Definition 136.144. The downstream citation surface is the consumer-readable package named in Definition 136.158, so the admitted consumer row factors through that packet and has no independent root route. □

Theorem 136.160 (DiffForm root consumer export exactness)**.** *The packet Definition 136.158 exposes exactly the six-coordinate carrier, displayed classifier, wedge-continuation rows, antisymmetry ledger, scalar endpoint row, source ledger, and exterior-input rows named by the cited root-unblock and consumer-face surfaces.*

Proof. Project the carrier and classifier coordinates with Definition 136.34. The carrier is the six-coordinate row

$$(d, \mathcal{P}, \tau, \sigma, \alpha, \lambda),$$

and the classifier reads only the displayed degree, tensor, scalar endpoint, antisymmetry, tensor-ledger, and source-ledger comparisons.

Next project the operation rows with Definition 136.35. Wedge admission reads only probe append, the Cont tensor ledger, degree bookkeeping, and the displayed antisymmetry ledger.

The exterior side is supplied by Definition 136.36. It restricts exterior consumers to degree, wedge, antisymmetry, scalar endpoint, and source-ledger projections.

Finally use Definition 136.121 and Definition 136.144. Face exhaustion says there is no exported row beyond the shared root and three consumer faces, and the readable boundary repacks the same carrier, classifier, stability, ledger, wedge-routing, exterior-input, and source NameCert rows. Hence the packet is exactly the stated export surface. □

Theorem 136.161 (DiffForm root consumer export no extra laws)**.** *The packet Definition 136.158 contains no* $d^2 = 0$*, closedness, nondegeneracy, contact nonintegrability, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumerExport no extra laws`

Proof. First inspect the defining source surface Definition 136.33. It explicitly contains no row for $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or a standard bridge.

Apply Definition 136.36. The theorem keeps de Rham nilpotence, symplectic closedness and nondegeneracy, and contact nonintegrability in the downstream certificate surfaces.

Then apply Definition 136.121 together with Definition 136.122. Exhaustion assigns every exported row to the shared root or a unique consumer face, while boundary exactness excludes the downstream law rows and any standard bridge.

The readable package boundary Definition 136.144 names the same negative boundary. Therefore the export packet carries the root-unblocking surface and no extra exterior or geometric law. □

136.14 DiffForm consumer boundary obligations

Theorem 136.162 (DiffForm consumer source scope)**.** *Every* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Contact}^{\uparrow}$*, or* $\mathsf{Symplectic}^{\uparrow}$ *consumer row exported from* $\mathsf{DiffForm}^{\uparrow}$ *is a projection of the root-unblock source surface of Definition 136.33. Hence such a row may read only* $BHist$*,* hsame*,* Cont*,* ProbeBundle*, the* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{TensorProduct}^{\uparrow}$ NameCert *sources, the six-coordinate carrier, and the* $\mathsf{DiffForm}^{\uparrow}$ *classifier.*

Proof. Start from the displayed root-unblock source surface. Its entries list the only histories and certificates made available to downstream consumers, so a consumer row must be obtained by forgetting part of that displayed surface.

Apply Definition 136.34 to restrict the visible carrier and classifier data to the six-coordinate carrier and the $\mathsf{DiffForm}^{\uparrow}$ classifier.

Finally apply Definition 136.131. It identifies the exposed source ledger with the stated $BHist$, hsame, Cont, ProbeBundle, $\mathsf{Manifold}^{\uparrow}$, and $\mathsf{TensorProduct}^{\uparrow}$ sources, leaving no additional source row for a downstream consumer to read. □

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormRootConsumer source projection scope`

Theorem 136.163 (DiffForm wedge degree ledger consumption)**.** *Any* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Contact}^{\uparrow}$*, or* $\mathsf{Symplectic}^{\uparrow}$ *consumer using a* $\mathsf{DiffForm}^{\uparrow}$ *wedge may consume the wedge degree ledger only through its* UnaryHistory *endpoints and the displayed* $\mathsf{Cont}(d_\omega, d_\eta, d_{\mathrm{out}})$ *row. This consumption rule is stable under* hsame *transport of the two input degree histories and the output degree history.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormWedgeDegreeLedger consumer boundary`

Proof. Read the ledger in Definition 136.22. The displayed data consist of the two input unary histories, the output unary history, and the single continuation row joining them.

Classifier stability is then exactly Definition 136.23: transporting the input form rows by hsame transports the degree histories and preserves the wedge classifier row.

The root-unblock wedge closure of Definition 136.35 identifies the same continuation row as the wedge-visible Cont entry. Therefore a downstream consumer can use the wedge degree ledger only through those unary endpoints and that displayed continuation row, and the rule is invariant under hsame transport. □

Theorem 136.164 (DiffForm exterior consumer boundary)**.** *The chapters* $\mathsf{DeRham}^{\uparrow}$*,* $\mathsf{Contact}^{\uparrow}$*, and* $\mathsf{Symplectic}^{\uparrow}$ *must supply nilpotence, closedness, nondegeneracy, and contact nonintegrability in their own certificate surfaces. The* $\mathsf{DiffForm}^{\uparrow}$ *export supplies only carrier/classifier rows, wedge rows, degree rows, source rows, and the ledger data made visible by the preceding consumer-source and wedge-degree obligations.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorConsumer boundary`

Proof. First apply Definition 136.162. This bounds every downstream consumer source to the displayed root-unblock carrier, classifier, history, continuation, probe, and NameCert rows.

Next apply Definition 136.163. It bounds all wedge-degree use to unary-history endpoints and the visible continuation row, with no extra operator law.

Finally combine these two bounds with Definition 136.36 and Definition 136.24. Those rows already separate exterior-derivative input consumption from $d^2 = 0$, closedness, nondegeneracy, and contact nonintegrability. The listed geometric laws therefore remain obligations of the $\mathsf{DeRham}^{\uparrow}$, $\mathsf{Contact}^{\uparrow}$, and $\mathsf{Symplectic}^{\uparrow}$ certificate surfaces rather than exports of $\mathsf{DiffForm}^{\uparrow}$. □

Theorem 136.165 (Differential-form NameCert obligation surface)**.** *The paper-side* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *obligation surface consists exactly of the carrier of Definition 136.1, the classifier of Definition 136.2, carrier transport, classifier stability, wedge ledger coverage, the wedge associator ledger, the wedge probe concatenation ledger, wedge probe concatenation classifier stability, the wedge degree ledger, wedge degree classifier stability, the exterior-derivative candidate ledger, its degree-raise and wedge-input stability rows, the antisymmetry permutation ledger, the de Rham input boundary of Definition 136.24, and the downstream scope row of Definition 136.61. It asserts no standard bridge beyond these BEDC rows and no identity* $d^2 = 0$*, closedness, nondegeneracy, or contact nonintegrability.*

Lean4 ✓ `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger semantic name certificate`

Proof. The carrier and classifier fields are the two displayed definitions. Carrier transport is Definition 136.10, and classifier stability is Definition 136.11.

The multiplicative operation exposed by the surface is the wedge row, whose admission and stability are exactly the visible tensor-concatenation statement of Definition 136.15. Its output probe spine is the visible concatenation ledger of Definition 136.16, with coverage and transport supplied by Definition 136.17 and Definition 136.18. Its associativity obligation is the visible ledger of Definition 136.19 and the closure statement of Definition 136.20. Its degree bookkeeping is the visible unary-history ledger of Definition 136.22, and classifier stability for that ledger is Definition 136.23. Antisymmetry is not hidden in the wedge operation: it is the explicit permutation ledger retained in every carrier row.

The exterior-derivative candidate interface is the visible package of Definition 136.54. Its degree bookkeeping is Definition 136.55, and its transport along the wedge-input classifier is Definition 136.56.

The remaining exported row is the downstream scope for exterior derivative consumers, bounded by Definition 136.24. Since all fields are stated with $BHist$, hsame, Cont, $\mathsf{ProbeBundle}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, this is a BEDC obligation surface and not a claim of equivalence with a standard differential-form object. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$**.**

Constructive story: A differential form is generated as a six-coordinate BHist row over fixed $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{TensorProduct}^{\uparrow}$ sources: a unary degree, a finite tangent-probe ProbeBundle, a multilinear tensor row, a scalar endpoint, an antisymmetry permutation ledger, and a source-provenance ledger. The carrier accepts exactly those rows whose probes are carried by the manifold source, whose tensor and scalar endpoints are certified by the tensor and scalar sources, and whose antisymmetry and provenance are displayed; any hidden alternating-map object, quotient, closedness predicate, or operator identity is excluded. Two carried forms are classified by degree hsame, matched tensor and scalar comparisons through retained source classifiers, and visible hsame on the antisymmetry, tensor, and provenance ledgers. The source is the manifold-plus-tensor certificate surface, the pattern records degree/probe/tensor/scalar/antisymmetry shape together with wedge and exterior-input routing, and the ledger keeps wedge concatenation, degree bookkeeping, permutation rows, and exterior-derivative candidate inputs visible. The five name-certificate obligations are supplied by a displayed carried form for inhabitation, by componentwise reflexivity, symmetry, and transitivity of hsame and the retained scalar classifier,

and by carrier and classifier transport for closure under recognized representatives. A row h lies in the signature gap of a DiffForm$^{\uparrow}$ token p when the source-domain accepts its six-coordinate form data, the signature relation pairs it with an answer form and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records the same degree, probe, tensor, scalar, antisymmetry, wedge, and exterior-input rows. The chapter registers this displayed form surface under DiffForm$^{\uparrow}$; downstream de Rham, symplectic, and contact consumers may read only these BEDC ledgers, with no $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or standard bridge introduced here.

Theory closure: TheoryGate(NameCert$_{\mathsf{DiffForm}\uparrow}$, maturePackageClosed)

Scope closed: Definition 136.165 is the paper-side scope for the displayed BHist, hsame, Cont, ProbeBundle, ManifoldUp, TensorProductUp, wedge-ledger, exterior-derivative input, and downstream boundary rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DiffForm}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.DiffFormUp.DiffFormExteriorDerivativeLedger\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close $d^2 = 0$, closedness, nondegeneracy, contact nonintegrability, or standard bridge closure.

Upgrade path: No further upgrade – chapter at top of the formal axis.

Chapter 137

A Concrete Naming Certificate for $\mathsf{DeRham}^{\uparrow}$

The de Rham interface $\mathsf{DeRham}^{\uparrow}$ packages the exterior-derivative chain complex on differential forms, certifying $d^2 = 0$ and hsame-stability. It depends on $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ (chapter 136) and $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ (chapter 99), and supplies the de Rham cohomology of a manifold.

137.1 Carrier definition

137.2 The certificate

Definition 137.1 (De Rham chain-complex boundary row)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *cochain differential* $d_{\mathrm{dR}} : \mathsf{BHist} \to \mathsf{BHist}$*. A row* $b : \mathsf{BHist}$ *is accepted by the* $\mathsf{DeRham}^{\uparrow}$ *chain-complex boundary predicate when it is in the image of the same cochain differential:*

$$\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(b) :\Longleftrightarrow \exists a : \mathsf{BHist},\ \mathsf{hsame}(b, d_{\mathrm{dR}}(a)).$$

The witness a *is the cochain preimage row. The predicate adds no quotient row and no host-level differential-form operation beyond the displayed cochain differential.*

Lean4 (definition) `BEDC.Derived.DeRhamUp.DeRhamBoundary`

Definition 137.2 (De Rham double exterior-derivative packet)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *source* T *as in Definition 136.1. Fix a* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *cochain differential* $d_{\mathrm{dR}} : \mathsf{BHist} \to \mathsf{BHist}$ *and a zero cochain endpoint* $z_{\mathrm{dR}} : \mathsf{BHist}$*. For carried* $\mathsf{DiffForm}^{\uparrow}$ *rows* ω*,* η*, and* θ*, the packet*

$$\mathsf{DeRhamDoubleExterior}_{M,T,d_{\mathrm{dR}}}(\omega, \eta, \theta, z_{\mathrm{dR}})$$

is accepted exactly when its visible fields are the following rows:

1. ω*,* η*, and* θ *are carried by* $\mathsf{DiffFormBHistCarrier}_{M,T}$*.*
2. *The two successive exterior-input rows* $\mathsf{DiffFormExteriorInput}_{M,T}(\omega, \eta)$ *and* $\mathsf{DiffFormExteriorInput}_{M,T}(\eta, \theta)$ *are accepted.*
3. *The cochain graph rows identify the carried exterior outputs with the same cohomology differential:*

$$\eta \mathsf{hsame} d_{\mathrm{dR}}(\omega) \quad \textit{and} \quad \theta \mathsf{hsame} d_{\mathrm{dR}}(\eta).$$

4. The cohomology differential respects hsame*: for all* $a, b : \mathsf{BHist}$*,*

$$\mathsf{hsame}(a, b) \Longrightarrow \mathsf{hsame}(d_{\mathrm{dR}}(a), d_{\mathrm{dR}}(b)).$$

5. The de Rham square-zero row and zero endpoint row are present:

$$\forall a : \mathsf{BHist},\ \mathsf{hsame}(d_{\mathrm{dR}}(d_{\mathrm{dR}}(a)), z_{\mathrm{dR}}) \quad \textit{and} \quad \mathsf{hsame}(z_{\mathrm{dR}}, \mathsf{e}).$$

The two $\mathsf{DiffForm}^{\uparrow}$ *exterior-input rows provide only the carried input-output surface of Definition 136.68; the square-zero row is a* $\mathsf{DeRham}^{\uparrow}$ *field read through the* $\mathsf{Cohomology}^{\uparrow}$ *cochain differential.*

Lean4 (definition) `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorPacket`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamTasteGate single carrier alignment`

Theorem 137.3 (De Rham double exterior derivative boundary)**.** *If* $\mathsf{DeRhamDoubleExterior}_{M,T,d_{\mathrm{dR}}}(\omega, \eta, \theta, z_{\mathrm{dR}})$ *is accepted, then the carried second exterior derivative row* θ *is classified as the zero cochain endpoint and is an accepted* $\mathsf{DeRham}^{\uparrow}$ *chain-complex boundary row:*

$$\mathsf{hsame}(\theta, z_{\mathrm{dR}}) \quad \wedge \quad \mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(\theta).$$

Moreover the intermediate carried row is a $\mathsf{Cohomology}^{\uparrow}$ *cocycle:*

$$C_{d_{\mathrm{dR}}}(\eta).$$

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorPacket boundary`
↪ variant `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorDerivative boundary`

Proof. Unpack the accepted packet of Definition 137.2. The two accepted $\mathsf{DiffForm}^{\uparrow}$ exterior-input rows are successive by Definition 136.68: the first has source ω and output η, while the second has source η and output θ. By Definition 136.69, these rows expose only the one-step exterior-input ledgers. The nilpotence row therefore comes from the de Rham packet, not from the $\mathsf{DiffForm}^{\uparrow}$ boundary exhaustion of Definition 136.107.

Let the graph comparisons be $\eta\mathsf{hsame}d_{\mathrm{dR}}(\omega)$ and $\theta\mathsf{hsame}d_{\mathrm{dR}}(\eta)$. Applying the hsame-respect row of d_{dR} to the first comparison gives

$$\mathsf{hsame}(d_{\mathrm{dR}}(\eta), d_{\mathrm{dR}}(d_{\mathrm{dR}}(\omega))).$$

The square-zero row at ω gives

$$\mathsf{hsame}(d_{\mathrm{dR}}(d_{\mathrm{dR}}(\omega)), z_{\mathrm{dR}}).$$

Transitivity with the second graph comparison yields $\mathsf{hsame}(\theta, z_{\mathrm{dR}})$.

For the boundary predicate, use the cochain preimage η. The second graph comparison is exactly the witness $\mathsf{hsame}(\theta, d_{\mathrm{dR}}(\eta))$ required by Definition 137.1. Hence $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(\theta)$ holds.

Finally, the same differential comparison, the square-zero row at ω, and the zero endpoint comparison $\mathsf{hsame}(z_{\mathrm{dR}}, \mathsf{e})$ give $d_{\mathrm{dR}}(\eta)\mathsf{hsame}\mathsf{e}$. By Definition 99.28, this is exactly $C_{d_{\mathrm{dR}}}(\eta)$. □

Theorem 137.4 (De Rham zero-endpoint boundary transport)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Cohomology}^\uparrow}$ *cochain differential* $d_{\mathrm{dR}} : \mathsf{BHist} \to \mathsf{BHist}$*. If a displayed row* $b : \mathsf{BHist}$ *is accepted by* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ *through a cochain preimage* $a : \mathsf{BHist}$ *and a transported endpoint* $b' : \mathsf{BHist}$ *satisfies* $\mathsf{hsame}(b', b)$*, then the transported row is again an accepted de Rham boundary row:*

$$\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(b')$$

with the same preimage witness a*. In particular, if* b *is compared by* hsame *to the zero endpoint* z_{dR}*, any displayed endpoint transported to that zero endpoint remains in the same boundary predicate.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundary zero endpoint hsame transport`
↪ variant `BEDC.Derived.DeRhamUp.DeRhamBoundary zero endpoint transport`

Proof. Unpack Definition 137.1. The accepted boundary row supplies a witness a and a comparison $\mathsf{hsame}(b, d_{\mathrm{dR}}(a))$.

The transported endpoint supplies $\mathsf{hsame}(b', b)$. By transitivity of hsame, the row b' compares with $d_{\mathrm{dR}}(a)$:

$$\mathsf{hsame}(b', d_{\mathrm{dR}}(a)).$$

Repack the same cochain preimage a in Definition 137.1. No new differential-form row, quotient row, or cohomology carrier is introduced; only the displayed endpoint has been transported along hsame. □

Theorem 137.5 (De Rham cocycle readback from square-zero packets)**.** *Every accepted packet*

$$\mathsf{DeRhamDoubleExterior}_{M,T,d_{\mathrm{dR}}}(\omega, \eta, \theta, z_{\mathrm{dR}})$$

reads its intermediate carried row η *as a* $\mathsf{Cohomology}^\uparrow$ *cocycle:*

$$C_{d_{\mathrm{dR}}}(\eta).$$

The readback uses only the packet's cochain graph row $\eta\mathsf{hsame}d_{\mathrm{dR}}(\omega)$*, the square-zero row for* d_{dR}*, and the zero endpoint comparison* $\mathsf{hsame}(z_{\mathrm{dR}}, \mathsf{e})$*.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorPacket cohomology cocycle readback`

Proof. Unpack Definition 137.2. The graph row identifies the intermediate carried row with the first cochain differential: $\eta\mathsf{hsame}d_{\mathrm{dR}}(\omega)$.

Apply the hsame-respect field of the cochain differential to this graph row. It gives a comparison from $d_{\mathrm{dR}}(\eta)$ to $d_{\mathrm{dR}}(d_{\mathrm{dR}}(\omega))$.

The square-zero row at ω compares $d_{\mathrm{dR}}(d_{\mathrm{dR}}(\omega))$ with z_{dR}, and the zero endpoint comparison identifies z_{dR} with e. Chaining these hsame comparisons yields $d_{\mathrm{dR}}(\eta)\mathsf{hsame}\mathsf{e}$.

By Definition 99.28, this is exactly the $\mathsf{Cohomology}^\uparrow$ cocycle predicate $C_{d_{\mathrm{dR}}}(\eta)$. □

Theorem 137.6 (De Rham NameCert obligation surface)**.** *The de Rham rows in this chapter expose the* $\mathsf{NameCert}_{\mathsf{DeRham}^\uparrow}$ *obligation surface over* $\mathsf{DiffForm}^\uparrow$ *and* $\mathsf{Cohomology}^\uparrow$*. The carrier is the differential-form carried row supplied by Definition 136.1; the classifier is the boundary predicate of Definition 137.1; the stability field is the endpoint transport of Definition 137.4; the exactness row is the square-zero boundary theorem Definition 137.3; and the ledger row is the cocycle readback of Definition 137.5. Together these rows supply the carrier, classifier, stability, exactness, and ledger entries required by the named de Rham certificate surface, without asserting a de Rham isomorphism or any quotient-level cohomology presentation.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundary semanticNameCert`

Proof. The carrier component is inherited from the visible $\mathsf{DiffForm}^{\uparrow}$ form carrier cited in the statement. The De Rham boundary row then classifies the second exterior derivative by a concrete cochain preimage in Definition 137.1.

Stability is supplied by Definition 137.4: once a boundary row is witnessed by a preimage, endpoint transport along hsame preserves the same predicate using that witness.

Exactness is supplied by Definition 137.3. It turns an accepted double exterior packet into both the zero endpoint comparison for the second row and the boundary predicate for that row.

The ledger is the readback Definition 137.5. It records that the intermediate row seen by the square-zero packet is a $\mathsf{Cohomology}^{\uparrow}$ cocycle. These named rows are precisely the advertised certificate obligations and no additional bridge or quotient surface is used. □

Definition 137.7 (De Rham boundary source packet)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *cochain differential* $d_{\mathrm{dR}} : \mathsf{BHist} \to \mathsf{BHist}$ *and a zero endpoint* $z_{\mathrm{dR}} : \mathsf{BHist}$*. A De Rham boundary source packet consists of the following* BHist *rows:*

1. *the cochain differential row* d_{dR} *and its zero endpoint comparison* $\mathsf{hsame}(z_{\mathrm{dR}}, \mathsf{e})$*;*
2. *carried* $\mathsf{DiffForm}^{\uparrow}$ *input rows* ω*,* η*, and* θ *with the two exterior-input rows of Definition 137.2;*
3. *the* $\mathsf{Cohomology}^{\uparrow}$ *cocycle row* $C_{d_{\mathrm{dR}}}(\eta)$ *read from the intermediate carried row;*
4. *the boundary predicate* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(\theta)$ *of Definition 137.1;*
5. *the finite ledger containing the cochain graph rows* $\eta\mathsf{hsame}d_{\mathrm{dR}}(\omega)$ *and* $\theta\mathsf{hsame}d_{\mathrm{dR}}(\eta)$*, the square-zero row for* d_{dR}*, and the displayed* $\mathsf{DiffForm}^{\uparrow}$ *and* $\mathsf{Cohomology}^{\uparrow}$ *source rows.*

The packet exports only these boundary-source rows. It does not add a quotient cohomology carrier, a de Rham isomorphism row, or an ambient manifold-level equality witness.

Lean4 (definition) `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket`

Theorem 137.8 (De Rham boundary source stability)**.** *Let* P *be a De Rham boundary source packet as in Definition 137.7. If the visible boundary endpoint* θ *is transported to* θ' *by* $\mathsf{hsame}(\theta', \theta)$*, then the same packet data transport to a boundary source packet with boundary endpoint* θ'*. The cochain differential, zero endpoint row,* $\mathsf{DiffForm}^{\uparrow}$ *input rows,* $\mathsf{Cohomology}^{\uparrow}$ *cocycle row, and finite ledger rows are unchanged except for replacing the displayed boundary endpoint by* θ'*.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourcePacket stability`

Proof. Unpack the packet. Its boundary component is $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(\theta)$, witnessed by the cochain preimage row named in Definition 137.1.

Apply Definition 137.4 to the displayed comparison $\mathsf{hsame}(\theta', \theta)$. The theorem keeps the same preimage witness and gives $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}(\theta')$.

All other packet rows are source or ledger rows that do not mention the new endpoint except through the boundary predicate just transported. Repacking the unchanged cochain differential, zero endpoint row, $\mathsf{DiffForm}^{\uparrow}$ inputs, cocycle row, and finite ledger with the transported boundary predicate gives the claimed packet. □

Theorem 137.9 (De Rham boundary consumer exactness)**.** *Every downstream consumer of a De Rham boundary source packet reads exactly the boundary row of Definition 137.1, the zero endpoint row of Definition 137.2, the cocycle row supplied by Definition 137.5, and the finite ledger rows displayed in Definition 137.7. The packet therefore supplies no additional cohomology quotient, no*

de Rham isomorphism row, and no hidden differential-form operation beyond the accepted BHist *boundary-source data.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket consumer exactness`

Proof. Start with an accepted packet. Its boundary row is part of the data in Definition 137.7, and Definition 137.3 identifies the accepted double exterior packet as the source of that boundary predicate.

The zero endpoint row is also explicit: it is the comparison $\mathsf{hsame}(z_{\mathrm{dR}}, \mathsf{e})$ carried by Definition 137.2. A consumer can read it only through that packet field and through endpoint transport by Definition 137.4.

The cocycle component is fixed by Definition 137.5. It reads the intermediate row η as $C_{d_{\mathrm{dR}}}(\eta)$ from the cochain graph row and the square-zero row, so the consumer receives exactly the displayed $\mathsf{Cohomology}^{\uparrow}$ cocycle row.

Finally, inspect the finite ledger in Definition 137.7. It lists the two cochain graph rows, the square-zero row, and the visible $\mathsf{DiffForm}^{\uparrow}$ and $\mathsf{Cohomology}^{\uparrow}$ source rows. Since the definition lists no quotient, isomorphism, or additional operation row, every downstream readback is exhausted by those rows. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourcePacket consumer exactness`

Theorem 137.10 (De Rham root carrier obligation)**.** *The root carrier obligation exposed by* $\mathsf{DeRham}^{\uparrow}$ *is exactly the displayed* $\mathsf{DiffForm}^{\uparrow}$ *row together with the* BHist *cochain differential packet. Concretely, the carrier reads a differential-form row* $\omega : \mathsf{BHist}$ *through the* $\mathsf{DiffForm}^{\uparrow}$ *source, a cochain differential* $d_{\mathrm{dR}} : \mathsf{BHist} \to \mathsf{BHist}$*, and the boundary preimage row of Definition 137.1; the square-zero data are the packet of Definition 137.2. No host manifold carrier, quotient carrier, or external cochain complex carrier is part of the root surface.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamRootCarrier obligation`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorPacket root carrier obligation`

Proof. Read the carrier side of Definition 137.6. Its visible source row is the $\mathsf{DiffForm}^{\uparrow}$ form row, while the cochain side is represented by the displayed BHist differential d_{dR}.

The boundary preimage component is exactly Definition 137.1. The square-zero endpoint data are exactly the packet of Definition 137.2.

Apply Definition 137.3 to identify the packet's second differential row as a boundary row, and apply Definition 137.5 to read the intermediate cochain row as a cocycle. These two readings account for the carried De Rham data without adding any carrier beyond the displayed $\mathsf{DiffForm}^{\uparrow}$ and BHist rows. □

Theorem 137.11 (De Rham root classifier obligation)**.** *The root classifier obligation for* $\mathsf{DeRham}^{\uparrow}$ *is exactly* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$*. If a boundary row is accepted by this classifier and its endpoint is transported by* hsame*, the transported endpoint is again classified by* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ *through the same cochain preimage witness. Thus classifier stability is precisely the endpoint-transport row of Definition 137.4.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundary hsame transport with empty endpoint`

Proof. Unpack Definition 137.1. Classification by $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ is the assertion that the public row has a displayed cochain preimage and compares by hsame with its d_{dR} image.

The preservation clause is Definition 137.4. It keeps the same preimage witness and changes only the visible endpoint along the supplied hsame comparison.

The square-zero packet of Definition 137.2, together with Definition 137.3, supplies the canonical boundary instance consumed by this classifier. The assembly theorem Definition 137.6 records that no second classifier row is exported. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundary packet classifier transport`

Theorem 137.12 (De Rham root ledger exactness)**.** *The public ledger for the* DeRham$^{\uparrow}$ *root surface consists only of the boundary preimage row of Definition 137.1, the square-zero endpoint row of Definition 137.2, the cocycle readback row of Definition 137.5, and the displayed source rows imported from* DiffForm$^{\uparrow}$ *and* Cohomology$^{\uparrow}$*. Its exactness entry is the boundary conclusion of Definition 137.3, transported at endpoints by Definition 137.4.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryGraphLedger endpoint empty from rows`

Proof. List the public rows named by Definition 137.6. They are the DiffForm$^{\uparrow}$ carrier row, the Cohomology$^{\uparrow}$ cochain and cocycle rows, the boundary preimage row, the endpoint-transport row, the square-zero boundary row, and the cocycle readback row.

The exactness component is provided by Definition 137.3: from the displayed double exterior packet it produces the accepted boundary predicate for the second differential row.

The endpoint component is stable by Definition 137.4, and the readback component is exactly Definition 137.5. Since each row is already named in the obligation surface, the ledger has no additional bridge, quotient, or cohomology-isomorphism entry. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket root ledger exactness`

Definition 137.13 (De Rham root consumer readback packet)**.** *The root consumer readback packet for* DeRham$^{\uparrow}$ *is the finite list of rows that a downstream root consumer may inspect before any standard boundary bridge. It consists of the* DiffForm$^{\uparrow}$ *carried-form row, the* Cohomology$^{\uparrow}$ *cochain and cocycle rows, the* Manifold$^{\uparrow}$ *source row only as the domain provenance already visible to* DiffForm$^{\uparrow}$*, the* TensorProduct$^{\uparrow}$ *row only as the algebraic source already visible to* DiffForm$^{\uparrow}$*, the* BHist *cochain endpoints, the* hsame *endpoint transport rows, the* Cont *ledgers for the cochain differential and boundary preimage, and the* Pkg *provenance row tying these entries to one package.*

Lean4 (definition) `BEDC.Derived.DeRhamUp.DeRhamRootConsumerReadbackPacket`

Theorem 137.14 (De Rham root consumer readback stability)**.** *If two root consumer readback packets for* DeRham$^{\uparrow}$ *are related componentwise by* hsame*, then the transported packet has the same accepted root readback surface. The transported surface keeps the same* DiffForm$^{\uparrow}$*,* Cohomology$^{\uparrow}$*,* Manifold$^{\uparrow}$*, and* TensorProduct$^{\uparrow}$ *source rows, the same finite* Cont *ledger shape, and the same* Pkg *provenance, with only the visible* BHist *endpoints replaced by their* hsame *mates.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket endpoint transport`

Proof. Project the packet of Definition 137.13 into source, endpoint, ledger, transport, and provenance rows. The source rows are read through the already displayed DiffForm$^{\uparrow}$ and Cohomology$^{\uparrow}$ entries, while the Manifold$^{\uparrow}$ and TensorProduct$^{\uparrow}$ rows remain provenance of the differential-form source.

Transport the visible BHist endpoints by the supplied hsame comparisons. The boundary classifier stability of Definition 137.4 keeps the same preimage row, and the root ledger exactness of Definition 137.12 keeps the same finite ledger list.

Repack the transported rows into the readback packet. Since no step introduces a quotient cohomology carrier, de Rham isomorphism, or host cochain complex, the transported packet exposes exactly the same root consumer surface. □

Theorem 137.15 (De Rham root unblock threshold surface)**.** *The root carrier obligation, root classifier obligation, root ledger exactness, root consumer readback packet, and root consumer readback stability theorem are exactly the* $\mathsf{DeRham}^{\uparrow}$ *root-unblock surface. A downstream consumer may read only the* $\mathsf{DiffForm}^{\uparrow}$*,* $\mathsf{Cohomology}^{\uparrow}$*,* $\mathsf{Manifold}^{\uparrow}$*,* $\mathsf{TensorProduct}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* Pkg *rows displayed in those blocks.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamRootUnblock threshold surface`
↪ variant `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket root unblock threshold surface`

Proof. Project the root carrier block Definition 137.10. It supplies the carried $\mathsf{DiffForm}^{\uparrow}$ row, the cochain differential, the boundary preimage row, and the square-zero packet.

Project the classifier and ledger blocks Definition 137.11 and Definition 137.12. These blocks account for $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$, endpoint transport by hsame, cocycle readback, and the finite ledger rows.

Finally add the readback packet and its stability theorem. They name the allowed downstream rows and show that their hsame movement is closed within the same finite packet. Therefore the listed blocks exhaust the root-unblock surface and exclude quotient cohomology, de Rham isomorphism, and external cochain-complex data. □

Theorem 137.16 (De Rham boundary public classifier exactness)**.** *The boundary rows of Definition 137.1, the endpoint stability of Definition 137.4, the double-exterior boundary row of Definition 137.3, the cocycle readback of Definition 137.5, and the root ledger exactness of Definition 137.12 assemble the public* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *classifier-exactness surface over* $\mathsf{DiffForm}^{\uparrow}$*,* $\mathsf{Cohomology}^{\uparrow}$*,* BHist*,* hsame*, and the displayed cochain differential* d_{dR}*.*

This public surface exports the carried differential-form row, the boundary classifier $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$*, endpoint stability under* hsame*, cocycle readback, and the finite ledger rows named above. It does not export a de Rham isomorphism, a quotient-cohomology carrier, or any cohomology equality beyond the displayed boundary and cocycle rows.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundary public classifier exactness`

Proof. Project the carrier field from Definition 137.1: the public object is the $\mathsf{DiffForm}^{\uparrow}$ source row together with the BHist cochain differential and its displayed preimage row.

Project the classifier field from Definition 137.11. The classifier is $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$, and Definition 137.4 supplies the stability field by keeping the same cochain preimage while transporting the endpoint by hsame.

Project exactness from Definition 137.3 and project the cohomology-facing readback from Definition 137.5. These two rows give the boundary and cocycle fields without introducing a quotient carrier.

Finally repack the finite list in Definition 137.12 as the public ledger field. Since that ledger names only the displayed $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, hsame, and cochain-differential rows, the repacked surface contains no de Rham isomorphism row and no quotient-cohomology surface. □

Theorem 137.17 (De Rham bridge input source scope)**.** *The bridge input rows for the* $\mathsf{DeRham}^{\uparrow}$ *public boundary surface are exactly the* $\mathsf{DiffForm}^{\uparrow}$ *carried-form rows, the* $\mathsf{Cohomology}^{\uparrow}$ *cochain differential rows, the* BHist *endpoints, and the displayed square-zero row* $d_{\mathrm{dR}}(d_{\mathrm{dR}}\omega) = 0$ *read*

through Definition 137.2. No host smooth manifold, quotient-cohomology representative, de Rham isomorphism, or external chain-complex carrier is part of this input scope.

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamDoubleExteriorPacket bridge input source scope`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket input source scope`

Proof. Start from the De Rham packet Definition 137.2. It displays the carried $\mathsf{DiffForm}^{\uparrow}$ row, the two cochain graph rows, the intermediate BHist endpoint, and the zero endpoint for the second differential.

Apply Definition 137.3. The theorem turns the packet into the boundary row and records the square-zero endpoint seen by the public surface.

Use Definition 137.4 and Definition 137.5 to keep the visible endpoint and cocycle readback inside the same displayed packet. The root ledger and public classifier exactness theorems, Definition 137.12 and Definition 137.16, then repackage only these input rows, so no outside manifold, quotient, or carrier row is exported. □

Theorem 137.18 (De Rham bridge classifier source scope)**.** *All classifier movement in the* $\mathsf{DeRham}^{\uparrow}$ *public boundary surface is by* hsame *and by the cochain differential stability row of Definition 137.4. The classifier source scope contains no equality of quotient cohomology classes, no chosen representative of a cohomology class, and no analytic comparison beyond the displayed* BHist *endpoint transport.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket bridge classifier source scope`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket bridge classifier source scope`

Proof. Project the classifier from Definition 137.16. Its visible predicate is $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ over the boundary row of Definition 137.1.

The only allowed movement of that predicate is the hsame endpoint transport in Definition 137.4. The preimage cochain is kept fixed, and the transported endpoint stays in the same boundary predicate.

The packet exactness theorem Definition 137.3, the cocycle readback theorem Definition 137.5, and the root ledger exactness theorem Definition 137.12 add no classifier movement beyond that stability row. Thus the classifier source scope is precisely hsame transport plus cochain differential stability. □

Theorem 137.19 (De Rham bridge ledger source scope)**.** *The* $\mathsf{DeRham}^{\uparrow}$ *bridge ledger exports only the boundary preimage, the zero endpoint, the cocycle readback, the finite graph rows, and the package provenance named by the public source packet. It does not export a host cochain complex, a quotient carrier, an isomorphism theorem, or any ledger row not already displayed by the De Rham packet and boundary source surface.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket bridge ledger source scope`

Proof. Read the ledger entries from Definition 137.7. They are the boundary preimage, the double-exterior packet, the cocycle row, and the finite source rows over $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, and BHist.

Apply Definition 137.12. It identifies the exported ledger with exactly the boundary row, zero endpoint, cocycle readback, and displayed source rows.

Finally use Definition 137.16 to move that finite ledger into the public classifier surface. The boundary theorem Definition 137.3, endpoint transport Definition 137.4, and cocycle readback

Definition 137.5 account for every row, so the ledger has no remaining branch for external chain-complex, quotient, or isomorphism data. □

Definition 137.20 (De Rham standard boundary bridge packet)**.** *The De Rham standard boundary bridge packet is the bridge-facing row obtained by packaging the public boundary source packet of Definition 137.7, the public classifier exactness surface of Definition 137.16, and the three bridge source-scope rows Definition 137.17, Definition 137.18, and Definition 137.19.*

Its exported fields are the boundary preimage, the zero endpoint, the cocycle readback, the finite graph rows over $\mathsf{DiffForm}^{\uparrow}$*,* $\mathsf{Cohomology}^{\uparrow}$*, and* BHist*, the public boundary classifier* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$*, and the provenance tying those rows to the De Rham packet. It exports no de Rham isomorphism, quotient carrier, host cochain complex, or external comparison row.*

Lean4 (definition) `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket`

Theorem 137.21 (De Rham standard boundary classifier compatibility)**.** *The standard boundary bridge packet of Definition 137.20 reads the same boundary classifier as the encoded classifier* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ *of Definition 137.1, represented in Lean by* BEDC.Derived.DeRhamUp.DeRhamBoundary*. For every bridge-visible boundary endpoint, classifier acceptance in the bridge packet is equivalent to acceptance by that displayed boundary predicate.*

Proof. Start with the double-exterior packet boundary. By Definition 137.3, the accepted packet produces the zero endpoint comparison, the boundary predicate $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$, and the cocycle row.

Transport the endpoint only through Definition 137.4. This keeps the same cochain preimage and therefore keeps the same boundary classifier rather than introducing a quotient or external cochain-complex equality.

Read back the cocycle by Definition 137.5. The cocycle row is a consequence of the displayed differential and zero endpoint, so it does not change the boundary classifier.

Finally apply Definition 137.16. It repacks the double-exterior boundary, zero-endpoint transport, cocycle readback, and finite ledger rows into the public classifier surface. Since the standard bridge packet is defined from exactly that public surface and the three bridge source-scope theorems, it reads the same $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ classifier. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket classifier compatibility`

Theorem 137.22 (De Rham bridge consumer ledger exhaustion)**.** *Every bridge consumer row of the standard boundary bridge packet of Definition 137.20 is one of the boundary preimage row, the zero endpoint row, the cocycle readback row, the finite graph rows over* $\mathsf{DiffForm}^{\uparrow}$*,* $\mathsf{Cohomology}^{\uparrow}$*, and* BHist*, or the package provenance row linking those fields to the public De Rham boundary source.*

Conversely, those rows repack into the standard boundary bridge packet. No consumer row exports a de Rham isomorphism, quotient carrier, external cochain complex, host manifold comparison, or cohomology representative.

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket consumer ledger exhaustion`

Proof. Project the packet fields from Definition 137.20. Its source component is Definition 137.7, so the available ledger rows are the boundary preimage, double-exterior packet data, cocycle row, finite source rows, and package provenance.

Apply the bridge input and classifier scope theorems. The input scope Definition 137.17 admits only the displayed $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, and square-zero rows. The classifier scope Definition 137.18 admits only hsame endpoint transport and cochain differential stability.

Apply Definition 137.19 for the final ledger projection. It identifies the bridge ledger with the boundary preimage, zero endpoint, cocycle readback, finite graph rows, and package provenance, with no remaining branch for quotient, isomorphism, or external cochain-complex data.

For the converse direction, repack those projected rows with Definition 137.16. The classifier compatibility theorem above supplies the same boundary predicate, and the three source-scope theorems close the bridge-facing packet. □

Theorem 137.23 (De Rham standard-boundary source exhaustion)**.** *Every standard-boundary bridge source row for the public* $\mathsf{DeRham}^{\uparrow}$ *boundary surface projects to one of the displayed* $\mathsf{DiffForm}^{\uparrow}$ *form rows,* $\mathsf{Cohomology}^{\uparrow}$ *cochain rows,* BHist *endpoint rows, exterior-square-zero rows,* hsame *endpoint-transport rows, or* Pkg *provenance rows of Definition 137.20. Conversely, those rows repack the standard-boundary bridge source surface.*

No quotient cohomology carrier, de Rham isomorphism row, host cochain complex, host manifold comparison, or selected cohomology representative is part of the standard-boundary source.

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket source ledger repack`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryGraphLedger source exhaustion`

Proof. Project the source packet through Definition 137.20. Its visible source is Definition 137.7, so the only available raw rows are the boundary preimage, the double-exterior packet, the cocycle readback, the finite graph rows over $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, and BHist, and the package provenance.

Apply Definition 137.17. It identifies the input side with the displayed carried forms, cochain-differential rows, BHist endpoints, and the square-zero row $d_{\mathrm{dR}}(d_{\mathrm{dR}}\omega) = 0$.

Apply Definition 137.18 and Definition 137.19. The classifier scope admits only hsame endpoint transport and cochain-differential stability, while the ledger scope admits only boundary preimage, zero endpoint, cocycle readback, finite graph rows, and package provenance.

For the converse direction, use Definition 137.16 together with Definition 137.21. The displayed rows reconstruct the same boundary predicate and the same bridge-facing source packet, with no branch for quotient, isomorphism, host-comparison, or representative data. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket source exhaustion`

Theorem 137.24 (De Rham standard-boundary classifier transport)**.** *Classifier transport across the standard-boundary bridge for the public* $\mathsf{DeRham}^{\uparrow}$ *boundary surface uses only* hsame *endpoint transport and cochain-differential stability. In particular, the transported classifier is the same displayed boundary predicate* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ *on the transported endpoint, and the transport does not read a quotient class, a de Rham isomorphism, a host manifold comparison, or a chosen cohomology representative.*

Proof. Start with the classifier identified by Definition 137.21. It is the same predicate $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ as the public boundary classifier of Definition 137.16.

Transport the endpoint by Definition 137.4. That theorem keeps the cochain preimage fixed, moves only the displayed endpoint by hsame, and preserves the cochain-differential row that witnesses the boundary predicate.

Now apply Definition 137.18. It says that all bridge-visible classifier movement is exhausted by precisely that hsame endpoint transport plus cochain-differential stability.

The input and ledger scope theorems, Definition 137.17 and Definition 137.19, add only the finite displayed rows and package provenance. Therefore the classifier transport has no remaining source from which quotient, isomorphism, host-manifold, or representative data could enter. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket classifier transport`

Theorem 137.25 (De Rham standard-boundary consumer readback)**.** *Every downstream consumer of the standard-boundary bridge packet reads only the boundary preimage row, the cocycle row, the finite graph rows over* $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, *and* BHist, *and the package provenance rows displayed by Definition 137.20. No consumer row reads a quotient carrier, a de Rham isomorphism, a host manifold comparison, or a selected cohomology representative.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryGraphLedger consumer readback boundary`

Proof. Project the consumer surface by Definition 137.22. It lists exactly the boundary preimage, zero endpoint, cocycle readback, finite graph rows, and package provenance rows available to a bridge consumer.

Use Definition 137.23 to identify the source side of those consumer rows. The source contributes only the displayed $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, square-zero, hsame, and Pkg rows.

Use Definition 137.24 for the classifier side. It keeps consumer-visible classifier movement inside hsame endpoint transport and cochain-differential stability.

Finally repack through Definition 137.20. Since both the source and classifier projections have already been exhausted by displayed rows, the consumer readback cannot contain quotient, isomorphism, host-manifold, or representative data. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket consumer readback`
↪ variant `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryGraphLedger append consumer readback`

Theorem 137.26 (De Rham standard-boundary zero cocycle example)**.** *For the zero cochain row carried by the public* $\mathsf{DeRham}^{\uparrow}$ *boundary packet, the standard-boundary bridge exports a zero cocycle consumer row. The row consists of the zero endpoint in* BHist, *the square-zero* $d_{\mathrm{dR}}(d_{\mathrm{dR}}0) = 0$ *witness, the* $\mathsf{Cohomology}^{\uparrow}$ *cocycle readback, and the* Pkg *provenance already displayed by Definition 137.20.*

Proof. Read the zero cochain through Definition 137.25. The consumer surface allows only the boundary preimage, cocycle, finite graph, and provenance rows of the standard-boundary packet.

Apply Definition 137.23. The zero cochain row is one of the displayed $\mathsf{Cohomology}^{\uparrow}$ cochain rows, and the zero endpoint is one of the displayed BHist endpoint rows.

Finally use Definition 137.24. It keeps the boundary classifier inside hsame endpoint transport and cochain-differential stability, so the zero endpoint yields the cocycle row without adding quotient or representative data. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket zero cocycle example`

Theorem 137.27 (De Rham standard-boundary coboundary consumer example)**.** *For any displayed boundary preimage* ω *in the public* $\mathsf{DeRham}^{\uparrow}$ *boundary packet, the standard-boundary bridge exports the coboundary consumer row*

$$\omega \xrightarrow{d_{\mathrm{dR}}} d_{\mathrm{dR}}\omega \xrightarrow{d_{\mathrm{dR}}} 0.$$

The consumer receives the boundary preimage row, the zero endpoint row, the cocycle readback row, and the finite $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, *and* Pkg *rows of the standard-boundary packet.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket coboundary consumer row`

Proof. Start with the preimage row supplied by Definition 137.23. It is one of the displayed cochain rows in the public boundary source packet, and its endpoint is one of the displayed BHist rows.

Use Definition 137.24 to keep the boundary classifier on the same displayed endpoint after hsame transport. The cochain-differential row therefore remains a boundary row for $d_{\mathrm{dR}}\omega$ with zero endpoint.

Project the exported consumer surface through Definition 137.25. It gives exactly the preimage, zero endpoint, cocycle readback, finite graph, and provenance rows, which are the displayed coboundary consumer row. □

Theorem 137.28 (De Rham standard-boundary finite graph completeness)**.** *Every consumer row of the standard-boundary bridge packet factors through the finite rows already listed by the bridge packet:* $\mathsf{DiffForm}^{\uparrow}$ *form rows,* $\mathsf{Cohomology}^{\uparrow}$ *cochain and cocycle rows,* BHist *endpoint rows,* hsame *endpoint transports, and* Pkg *provenance. Thus the standard-boundary packet is complete for finite graph consumers of the displayed boundary surface.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryGraphLedger finite graph completeness`

Proof. Apply Definition 137.25. It restricts every downstream consumer to the boundary preimage, cocycle, finite graph, and provenance rows of the standard-boundary packet.

Apply Definition 137.23 for the source factorization. The source side is exhausted by the displayed $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, BHist, square-zero, hsame, and Pkg rows.

Finally apply Definition 137.24. The classifier side has no additional branch beyond hsame endpoint transport and cochain-differential stability. Combining the two projections gives finite graph completeness for the bridge packet. □

Theorem 137.29 (De Rham root boundary preimage threshold)**.** *Every accepted standard-boundary source row of the public* $\mathsf{DeRham}^{\uparrow}$ *bridge packet projects to a concrete* BHist *boundary preimage witness for the classifier* $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$*. The witness is the source cochain row named by Definition 137.20; its endpoint is the boundary row of Definition 137.1, and every accepted source is exhausted by the rows listed in Definition 137.23.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket boundary preimage threshold`

Proof. Project the accepted source through Definition 137.20. The bridge packet exposes only the public boundary source packet, the displayed boundary classifier, and the finite source rows over $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, and BHist.

Apply Definition 137.23. It says that an accepted standard-boundary source row is one of the displayed form, cochain, endpoint, square-zero, hsame, or Pkg rows and conversely that these rows repack the source.

Finally read the boundary predicate from Definition 137.1. The cochain row supplies the preimage, the endpoint row supplies the concrete BHist boundary target, and the displayed predicate $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$ records exactly that preimage witness. □

Theorem 137.30 (De Rham root cocycle ledger threshold)**.** *The same standard-boundary source packet exports its intermediate row as a* $\mathsf{Cohomology}^{\uparrow}$ *cocycle row and not as a quotient representative. The exported ledger is the cocycle readback row of Definition 137.5, filtered through the consumer ledger exhaustion theorem Definition 137.22.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket root cocycle ledger threshold`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamRootCocycleLedger threshold`
Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket cocycle ledger threshold`

Proof. Start with the source packet of Definition 137.20. Its cochain component is the row already used by the boundary classifier, so the intermediate $\mathsf{Cohomology}^{\uparrow}$ object is visible before any quotient or representative choice could be introduced.

Apply Definition 137.5. The zero endpoint for $d_{\mathrm{dR}}(d_{\mathrm{dR}}\omega)$ reads back as a cocycle row in $\mathsf{Cohomology}^{\uparrow}$.

Now apply Definition 137.22. The consumer ledger admits boundary preimage, zero endpoint, cocycle readback, finite graph rows, and package provenance only. Since quotient carriers and selected cohomology representatives are excluded there, the intermediate row exported by the packet is the cocycle ledger row itself. □

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamBoundarySourceLedgerPacket cocycle ledger threshold`

Theorem 137.31 (De Rham root standard-bridge consumer threshold)**.** *Downstream standard-bridge consumers of the public* $\mathsf{DeRham}^{\uparrow}$ *boundary packet see exactly the boundary preimage row, the zero endpoint row, the cocycle row, the finite graph rows over* $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, *and* BHist, *and the* Pkg *provenance rows. No consumer row sees a quotient carrier, a de Rham isomorphism, a host cochain complex, a host manifold comparison, or a chosen cohomology representative.*

Lean4 ✓ `BEDC.Derived.DeRhamUp.DeRhamStandardBoundaryBridgePacket consumer threshold`

Proof. First transport the classifier through Definition 137.24. It keeps the consumer-visible classifier inside hsame endpoint transport and cochain-differential stability for $\mathsf{DeRhamBoundary}_{d_{\mathrm{dR}}}$.

Next read the consumer surface by Definition 137.25. The permitted consumer rows are the boundary preimage, cocycle row, finite graph rows over $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, and BHist, and package provenance.

Finally apply Definition 137.22. It adds the zero endpoint row and proves that the listed rows exhaust the consumer ledger. The excluded quotient, isomorphism, host-comparison, and representative rows therefore have no source branch in the standard boundary bridge packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$**.**

Constructive story: A de Rham boundary is built on top of $\mathsf{DiffForm}^{\uparrow}$ and $\mathsf{Cohomology}^{\uparrow}$ by displaying a cochain differential d_{dR}, a zero endpoint, carried differential-form rows, and the graph comparisons that identify successive exterior outputs with applications of that same differential. The carrier accepts a boundary row only when some cochain preimage a is displayed with $\mathsf{hsame}(b, d_{\mathrm{dR}}(a))$, and the double-exterior packet additionally carries the two form inputs, square-zero row, zero endpoint row, and cohomology cocycle readback; quotient cohomology classes and de Rham isomorphism data are absent. The classifier is the displayed boundary predicate transported by endpoint hsame, keeping the same cochain preimage and the same differential row rather than comparing quotient representatives. The source is the differential-form and cohomology certificate surface with the cochain differential, the pattern is the square-zero boundary and cocycle packet, and the ledger records boundary preimages, zero endpoints, finite graph rows, cocycle readback, and package provenance. The five name-certificate obligations are supplied by accepted boundary packets for inhabitation, by history-equality for reflexive, symmetric, and transitive endpoint movement, and by the zero-endpoint boundary transport theorem for closure of the boundary predicate under classified representatives. A row h lies in the signature gap of a $\mathsf{DeRham}^{\uparrow}$ token p when the source-domain accepts its displayed form, cochain, boundary, and cocycle rows, the signature relation pairs them with an answer boundary packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the same preimage, square-zero endpoint, and cocycle ledger are what p records. The standard boundary bridge reads exactly this predicate and ledger surface and reads it back without exporting a host cochain complex, quotient carrier, selected representative, or de Rham isomorphism; the chapter registers the construction under $\mathsf{DeRham}^{\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The standard-boundary bridge, mature examples, and root threshold bind DeRhamUp to the displayed DiffFormUp, CohomologyUp, BHist, hsame, and cochain-differential rows by Definition 137.21 and Definition 137.22.

Formal verification: FormalStatus(NameCert$_{\mathsf{DeRham}\uparrow}$, encodedDef)
Lean target: `BEDC.Derived.DeRhamUp.DeRhamBoundary`
Bridge status: paperBridge
Not claimed: No de Rham isomorphism, quotient cohomology construction, or mature package completeness theorem is claimed here.
Upgrade path: The remaining step is Lean-side bridge checking for the existing standard boundary bridge.

Chapter 138

A Concrete Naming Certificate for $\mathsf{Symplectic}^{\uparrow}$

The symplectic-structure interface $\mathsf{Symplectic}^{\uparrow}$ packages a closed nondegenerate 2-form on an even-dimensional manifold carrier. It depends on $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ (chapter 136) and $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96), and is the substrate for Hamiltonian dynamics and Poisson-bracket certificates.

138.1 Carrier definition

Definition 138.1 (SymplecticUp carrier-classifier surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *source* Ω *over* BHist*. A* $\mathsf{Symplectic}^{\uparrow}$ *carrier-classifier surface is the displayed BEDC package*

$$\mathsf{SymplecticCarrierClassifier}_{M,\Omega}(\omega, d\omega).$$

Its traditional target is a closed nondegenerate two-form on an even-dimensional manifold. Its displayed rows are exactly:

1. *the* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source row and the* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *source row over the same* BHist *endpoint;*
2. *the shared manifold endpoint carrier transported from* M *to* Ω*;*
3. *an even-dimensional witness for the shared manifold endpoint;*
4. *a carried degree-two differential-form row* ω *supplied by* Ω*;*
5. *the exterior-derivative input row* $d\omega$ *routed through the* $\mathsf{DiffForm}^{\uparrow}$ *exterior-derivative boundary;*
6. *the closedness row* $d\omega = 0$ *attached only to the displayed degree-two form;*
7. *the nondegeneracy row asserting that* ω *pairs tangent rows on the shared manifold carrier without a hidden kernel;*
8. *the classifier obtained by restricting* hsame *to rows that preserve the shared manifold endpoint, the even-dimensional witness,* ω*,* $d\omega$*, closedness, and nondegeneracy;*
9. *ledger rows for chart projection, degree projection, derivative-input projection, wedge and pairing projection, and hidden source projections back to* M *and* Ω*;*
10. *exactness predicates saying that the surface has coverage for every certified symplectic read, soundness for every listed row, and completeness for every row named above.*

The surface contains no standard bridge row.

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierSurface semantic name certificate`

138.2 The certificate

Definition 138.2 (Symplectic obligation boundary)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{DiffForm}^\uparrow}$ *source* Ω *over* BHist*. The* $\mathsf{Symplectic}^\uparrow$ *obligation boundary is the BEDC package whose source is the paired* $\mathsf{DiffForm}^\uparrow$ *and* $\mathsf{Manifold}^\uparrow$ *certificate boundary, whose carrier is a manifold carrier equipped with a degree-two admissible differential form row* ω*, and whose classifier is the paired manifold classifier together with the* $\mathsf{DiffForm}^\uparrow$ *form classifier restricted to that row.*

The stability rows are carrier transport for the shared manifold carrier, closedness transport for the displayed two-form row, nondegeneracy transport for the same paired manifold/form carrier, and endpoint transport through hsame *and* Cont *wherever the downstream consumer boundary rows expose continuation data. The ledger records exactly the hidden atlas witnesses supplied by* M*, the hidden form witnesses supplied by* Ω*, the degree-two row, the closedness witness for* ω*, the nondegeneracy witness for* ω*, and the coverage, soundness, and completeness rows saying that every certified* $\mathsf{Symplectic}^\uparrow$ *read factors through this boundary.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary dependency nonempty`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary readback rows`

Theorem 138.3 (Symplectic carrier and classifier obligations)**.** *The source displayed in Definition 138.2 supplies the carrier and classifier rows required by* $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$*: the* $\mathsf{Manifold}^\uparrow$ *carrier row, the* $\mathsf{DiffForm}^\uparrow$ *degree-two admissible form row, the paired classifier on manifold and form components, and the carrier-transport row for the paired boundary over* BHist*.*

Proof. Unpack Definition 138.2. The first source component is the $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ boundary and the second is the $\mathsf{NameCert}_{\mathsf{DiffForm}^\uparrow}$ boundary, both read over the same BHist substrate.

The carrier row is the shared manifold endpoint together with the degree-two admissible form row. The classifier is therefore the paired classifier obtained by taking the manifold classifier from M and the form classifier from Ω on the displayed row ω.

Carrier transport is componentwise. The $\mathsf{Manifold}^\uparrow$ certificate transports the manifold carrier, the $\mathsf{DiffForm}^\uparrow$ certificate transports the admissible form row, and the paired boundary repacks these transported rows as the carrier and classifier data required by $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$. □

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierSurface carrier classifier obligations`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierSurface obligations`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary carrier classifier obligations`

Theorem 138.4 (Symplectic closed two-form obligation)**.** *Closedness of the selected two-form* ω *is an explicit* $\mathsf{Symplectic}^\uparrow$ *obligation transported along the* $\mathsf{DiffForm}^\uparrow$ *classifier. The* $\mathsf{DiffForm}^\uparrow$ *source supplies the admissible degree-two form row and the exterior-input boundary needed to state closedness, but it does not make* $d\omega = 0$ *automatic for every downstream consumer read.*

Proof. Begin with the degree-two row ω from Definition 138.2. The $\mathsf{DiffForm}^\uparrow$ boundary supplies the form classifier, exterior-input routing, and downstream consumer boundary rows needed to express the closedness predicate on that row.

Closedness is then recorded as its own ledger row. Transport along the $\mathsf{DiffForm}^{\uparrow}$ classifier moves the admissible form row and the exterior-input boundary to the classified endpoint, so the closedness row remains a $\mathsf{Symplectic}^{\uparrow}$ obligation on the transported form.

The last clause is the consumer-boundary separation. $\mathsf{DiffForm}^{\uparrow}$ exposes the rows needed by $\mathsf{Symplectic}^{\uparrow}$ to ask for $d\omega = 0$, while the truth of that row belongs to the $\mathsf{Symplectic}^{\uparrow}$ certificate boundary itself. □

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary closed two form row`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary closed two form transport`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierSurface closed two form obligation`

Theorem 138.5 (Symplectic nondegeneracy stability obligation)**.** *Nondegeneracy of ω is a separate classifier-stable row over the paired manifold/form carrier of Definition 138.2. It is transported by the paired* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{DiffForm}^{\uparrow}$ *classifiers and is not identified with the admissible-form carrier row or with the closedness row.*

Proof. Use the paired carrier from Definition 138.2. Nondegeneracy is stated only after the manifold carrier and the degree-two form row have both been selected, so it is a row over the paired surface rather than a row of either dependency alone.

Transport is again componentwise. The $\mathsf{Manifold}^{\uparrow}$ classifier transports the carrier on which nondegeneracy is evaluated, the $\mathsf{DiffForm}^{\uparrow}$ classifier transports the form row, and the paired classifier records the transported nondegeneracy witness as a $\mathsf{Symplectic}^{\uparrow}$ ledger row.

This separates the three obligations consumed downstream: admissibility of the two-form, closedness of that two-form, and nondegeneracy on the paired manifold/form carrier. □

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary nondegeneracy endpoint split`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary nondegeneracy row`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary nondegeneracy transport`

Theorem 138.6 (SymplecticUp closed two-form obligation)**.** *For a* $\mathsf{Symplectic}^{\uparrow}$ *carrier-classifier surface of Definition 138.1, the closedness obligation is local to the displayed* $\mathsf{DiffForm}^{\uparrow}$ *degree-two form row ω and its exterior-derivative input row $d\omega$. If two accepted surface reads are related by the restricted classifier of the same definition, then the closedness row $d\omega = 0$ transports with the degree-two form row.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierEndpointSurface closed two form transport`
Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierSurface closed two form transport`

Proof. Unpack Definition 138.1. The closedness row is listed only after the carried degree-two form row ω and the exterior-derivative input row $d\omega$ have been fixed.

Apply the $\mathsf{DiffForm}^{\uparrow}$ exterior-derivative routing named by Definition 136.61. It permits the symplectic consumer to read the source carrier, degree row, and exterior-derivative candidate ledger, while it contributes no closedness predicate of its own.

The classifier clause of the displayed surface is the restriction of hsame to the shared manifold endpoint, ω, $d\omega$, and the closedness row. Transport therefore changes only the accepted surface read, not the local obligation that $d\omega$ is the zero exterior-derivative row. □

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary closed successor not empty`

Theorem 138.7 (SymplecticUp nondegeneracy obligation)**.** *For a* $\mathsf{Symplectic}^{\uparrow}$ *carrier-classifier surface of Definition 138.1, nondegeneracy is a* $\mathsf{Symplectic}^{\uparrow}$*-local predicate on the shared* $\mathsf{Manifold}^{\uparrow}$ *carrier and the displayed degree-two form row* ω*. If the surface classifier transports the shared manifold endpoint and the degree-two form row, then it transports the nondegeneracy row on that same carrier.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierEndpointSurface nondegeneracy transport`

Proof. Begin with the shared manifold endpoint row in Definition 138.1. The $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ source supplies the carrier on which tangent rows and chart projections are read.

The degree-two form row ω is supplied by the $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ source over that same endpoint. Nondegeneracy is therefore stated as a local pairing predicate for ω on the transported manifold carrier, rather than as an independent $\mathsf{DiffForm}^{\uparrow}$ root law.

Finally read the classifier clause. It restricts hsame to preserve the shared endpoint, ω, and the nondegeneracy row, so the nondegeneracy predicate transports exactly with the carrier-classifier surface. □

Theorem 138.8 (SymplecticUp ledger exactness obligation)**.** *The* $\mathsf{Symplectic}^{\uparrow}$ *ledger and exactness obligation consists exactly of the carrier, classifier, closedness, nondegeneracy, stability, ledger, and dependency rows displayed in Definition 138.1. Coverage says every certified symplectic read factors through that surface, soundness says every such read is one of the listed rows, and completeness says every listed row is available to the* $\mathsf{Symplectic}^{\uparrow}$ *carrier-classifier surface.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary ledger exactness obligation`

Proof. Inspect the rows of Definition 138.1. The carrier and dependency cases are the $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ source, the $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ source, the shared endpoint, and the even-dimensional witness.

The law cases are the degree-two form row, the exterior-derivative input row, the closedness row of Definition 138.6, and the nondegeneracy row of Definition 138.7.

The remaining cases are the restricted hsame classifier, the stability carried by that restriction, the chart, degree, derivative-input, wedge and pairing ledger rows, and the hidden source projections. These cases are exactly the coverage, soundness, and completeness clauses stated in the theorem. □

Theorem 138.9 (SymplecticUp public NameCert export)**.** *The carrier-classifier surface of Definition 138.1, the obligation boundary of Definition 138.2, the carrier and classifier obligations of Definition 138.3, the closed two-form obligation of Definition 138.6, the nondegeneracy obligation of Definition 138.7, and the ledger exactness obligation of Definition 138.8 assemble the public* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *surface over* BHist*. The exported surface contains the shared* $\mathsf{Manifold}^{\uparrow} + \mathsf{DiffForm}^{\uparrow}$ *carrier, the restricted* hsame *classifier, closedness, nondegeneracy, and exact ledger rows, and it contains no* $\mathsf{StandardBridge}$ *row.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary semantic name certificate`
↪ variant `BEDC.Derived.SymplecticUp.SymplecticCarrierClassifierEndpointSurface semantic name certificate`

Proof. Unpack Definition 138.1. Its rows name the two dependency certificates over BHist, the shared manifold endpoint, the degree-two form row, the exterior-derivative input row, closedness, nondegeneracy, the restricted classifier, and the ledger projections.

The boundary in Definition 138.2 supplies the same data as a BEDC obligation package. Apply Definition 138.3 to read the carrier and classifier fields, then apply Definition 138.6 and Definition 138.7 to attach the two public form laws on the paired carrier.

Finally apply Definition 138.8. Coverage, soundness, and completeness repack all listed rows as the public $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$ surface, while the defining surface explicitly excludes any $\mathsf{StandardBridge}$ row. □

Theorem 138.10 (SymplecticUp closed nondegenerate public stability)**.** *For the public* $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$ *surface of Definition 138.9, the restricted* hsame *classifier transports closedness and nondegeneracy together on the same paired* $\mathsf{Manifold}^\uparrow + \mathsf{DiffForm}^\uparrow$ *carrier. Thus an accepted surface read cannot transport* $d\omega = 0$ *without transporting the manifold endpoint, the degree-two form row* ω*, and the nondegeneracy row for that same pair.*

Proof. Start from the classifier clause in Definition 138.1. It is not the ambient classifier on arbitrary rows; it is hsame restricted to preservation of the shared endpoint, the even-dimensional witness, ω, $d\omega$, closedness, and nondegeneracy.

Apply Definition 138.6. Closedness transports only with the displayed degree-two form row and its exterior-derivative input.

Apply Definition 138.7. Nondegeneracy transports only on the same shared manifold carrier and displayed form row. The public export theorem then repacks these two transported laws into one $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$ surface, so no accepted read separates closedness from nondegeneracy or from the carrier on which both are stated. □

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary closed nondegenerate public stability`

Theorem 138.11 (SymplecticUp Hamiltonian and Poisson consumer frontier)**.** *Downstream Hamiltonian dynamics and Poisson-bracket certificates may read from* $\mathsf{Symplectic}^\uparrow$ *only the shared manifold carrier, the degree-two form row, exterior-derivative routing, closedness, nondegeneracy, the pairing ledger, and classifier transport supplied by Definition 138.9. From* $\mathsf{Symplectic}^\uparrow$ *alone they may not infer a Poisson bracket, a Hamiltonian flow, a Jacobi identity, or a standard model.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary hamiltonian poisson consumer frontier`

Proof. Inspect the exactness rows of Definition 138.1. The permitted consumer reads are precisely the rows named in the theorem: carrier, degree-two form, derivative input, closedness, nondegeneracy, pairing ledger, and classifier transport.

Use Definition 138.8 to convert that list into a coverage, soundness, and completeness boundary. Soundness rules out any additional consumer row not displayed by the surface.

The excluded objects require structure not present in the displayed rows. A Poisson bracket needs its own bracket operation and Jacobi witness, a Hamiltonian flow needs a flow construction, and a standard model needs a $\mathsf{StandardBridge}$ row. Since the public export in Definition 138.9 carries none of those rows, the consumer frontier is exactly the stated boundary. □

Theorem 138.12 (SymplecticUp mature consumer completeness boundary)**.** *Every Hamiltonian, Poisson, or standard-model consumer row admitted by this chapter either factors through the public* $\mathsf{NameCert}_{\mathsf{Symplectic}^\uparrow}$ *export, the closed/nondegenerate public stability row, and the Hamiltonian/Poisson consumer frontier, or is explicitly outside the* $\mathsf{Symplectic}^\uparrow$ *boundary. In particular, the chapter supplies the shared manifold carrier, degree-two form, exterior-derivative routing,*

closedness, nondegeneracy, pairing ledger, and classifier transport rows; it does not supply a Poisson bracket, Hamiltonian flow, Jacobi witness, concrete carrier implementation, or StandardBridge *row.*

Lean4 ✓ `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary mature consumer completeness`

Proof. First apply Definition 138.9. It supplies the public carrier-classifier surface over BHist with the shared manifold and degree-two form rows, the restricted hsame classifier, closedness, nondegeneracy, and exact ledger rows.

Next apply Definition 138.10. This ensures that any admitted consumer of closedness or nondegeneracy consumes those rows on the same paired $\mathsf{Manifold}^{\uparrow} + \mathsf{DiffForm}^{\uparrow}$ carrier and cannot separate the law row from the endpoint to which it belongs.

Finally apply Definition 138.11. Its soundness clause lists the permitted Hamiltonian and Poisson-facing reads and excludes the Poisson bracket, Hamiltonian flow, Jacobi witness, and standard bridge. Thus every admitted consumer row factors through the displayed public export and stability rows, while every other row is outside the boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$.

Constructive story: A symplectic surface is generated by sharing a $\mathsf{Manifold}^{\uparrow}$ carrier with a $\mathsf{DiffForm}^{\uparrow}$ degree-two form row, then recording the exterior-derivative input, the closedness row, an even-dimensional witness, and the nondegeneracy row on that same carrier. A history is accepted only through this joint manifold/form surface, and two accepted reads classify together when hsame preserves the shared endpoint, the degree-two form, the derivative input, closedness, and nondegeneracy. The source covers exactly the displayed $\mathsf{Manifold}^{\uparrow} + \mathsf{DiffForm}^{\uparrow}$ rows, the pattern records that the two-form is closed and nondegenerate on the transported carrier, and the ledger keeps chart, degree, derivative-input, wedge, pairing, and hidden source projections aligned. The five naming-certificate obligations are supplied by the exported public surface: the carrier row gives an inhabitant, the restricted classifier is reflexive, symmetric, and transitive because each component is transported by hsame, and carrier closure holds only while closedness and nondegeneracy remain attached to the same displayed surface. A history h lies in the signature gap of a $\mathsf{Symplectic}^{\uparrow}$ token p when the source accepts the shared carrier and two-form rows for h, the signature relation pairs them with an answer surface and evidence ledger, and token introduction names that answer as p; unfolded, h carries the same even-dimensional manifold endpoint, closed degree-two form, and nondegenerate pairing rows encoded by p. Downstream Hamiltonian and Poisson consumers may read the shared carrier, form, derivative routing, pairing ledger, and classifier transport, but no bracket, flow, concrete closedness proof, or standard model is exported. The chapter registers this closed nondegenerate two-form construction under the name $\mathsf{Symplectic}^{\uparrow}$; no standalone Poisson structure, Hamiltonian dynamics, or host symplectic manifold object is introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature scope is the displayed $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ carrier-classifier surface, the hsame-restricted classifier, exterior-derivative routing, ledger rows, public $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ export, closed/nondegenerate stability, and Hamiltonian/Poisson consumer completeness boundary cited in Definition 138.1, Definition 138.2, Definition 138.3, Definition 138.9, Definition 138.10, Definition 138.11, and Definition 138.12.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SymplecticUp.SymplecticObligationBoundary\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not claim a concrete $\mathsf{Symplectic}^{\uparrow}$ carrier implementation, a closedness proof, a nondegeneracy proof, a Hamiltonian or Poisson structure, or a standard bridge.

Upgrade path: Only formal bridge checking is separate; the theory axis is already at maturePackageClosed.

Chapter 139

A Concrete Naming Certificate for $\mathsf{Contact}^{\uparrow}$

The contact-structure interface $\mathsf{Contact}^{\uparrow}$ packages a maximally nonintegrable hyperplane field on an odd-dimensional manifold carrier, recorded as a 1-form whose top-degree wedge is nondegenerate. It depends on $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ (chapter 136) and $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96).

139.1 Carrier definition

Definition 139.1 (ContactUp carrier-classifier surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source* M *and a* $\mathsf{NameCert}_{\mathsf{DiffForm}^{\uparrow}}$ *source* Ω*. A* $\mathsf{Contact}^{\uparrow}$ *carrier-classifier surface over* $BHist$ *is the displayed BEDC package*

$$\mathsf{ContactCarrierClassifier}_{M,\Omega}.$$

Its source rows are the $BHist$ *carrier rows inherited from* M *and* Ω*, the odd-dimensional manifold witness on the shared manifold endpoint, a carried* $\mathsf{DiffForm}^{\uparrow}$ *degree-one form row* α*, a contact hyperplane row* $\ker(\alpha)$ *on the same manifold carrier, an exterior-derivative input row* $d\alpha$*, the wedge ladder rows for* $\alpha \wedge (d\alpha)^k$*, and the top wedge row* $\alpha \wedge (d\alpha)^n$ *when the displayed dimension row is* $2n + 1$*.*

Its classifier is the restriction of hsame *to rows that preserve the shared* $\mathsf{Manifold}^{\uparrow}$ *carrier, the* $\mathsf{DiffForm}^{\uparrow}$ *one-form row, the hyperplane row, the exterior-derivative input row, and every displayed wedge-ladder row. The surface contains no nonintegrability primitive and no standard bridge row.*

Lean4 (definition) `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface`
Lean4 ✓ `BEDC.Derived.ContactUp.ContactTasteGate single carrier alignment`

Theorem 139.2 (ContactUp form-row obligation surface)**.** *Every* $\mathsf{Contact}^{\uparrow}$ *carrier-classifier surface of Definition 139.1 exposes exactly the* $\mathsf{Manifold}^{\uparrow}$ *source, the* $\mathsf{DiffForm}^{\uparrow}$ *contact consumer face, the degree-one form row* α*, the exterior-derivative input row* $d\alpha$*, and the wedge ladder rows* $\alpha \wedge (d\alpha)^k$ *routed by the* $\mathsf{DiffForm}^{\uparrow}$ *root threshold. It exposes no nilpotence row, no symplectic closedness row, no contact nonintegrability witness, and no standard bridge as part of the* $\mathsf{DiffForm}^{\uparrow}$ *root face.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface form row obligation`

Proof. Unpack Definition 139.1. The source fields are the shared $\mathsf{Manifold}^{\uparrow}$ carrier and the $\mathsf{DiffForm}^{\uparrow}$ carried rows for the one-form, exterior-input, and wedge ladder.

Apply Definition 136.117 and Definition 136.118. They place the contact reads in the $\mathsf{Contact}^{\uparrow}$ consumer face and separate them from the de Rham, symplectic, and shared root laws.

Finally apply Definition 136.101. The threshold exports carried forms, classifier data, wedge ledgers, and exterior-input routing, so the $\mathsf{Contact}^{\uparrow}$ surface receives exactly the form row, derivative input row, and wedge ladder rows stated above. □

139.2 The certificate

Theorem 139.3 (ContactUp nondegeneracy obligation)**.** *For a* $\mathsf{Contact}^{\uparrow}$ *carrier-classifier surface over a displayed dimension row* $2n+1$*, top-degree wedge nondegeneracy is a* $\mathsf{Contact}^{\uparrow}$*-local obligation transported by the* $\mathsf{Contact}^{\uparrow}$ *classifier:*

$$\mathsf{Nondegenerate}_M(\alpha \wedge (d\alpha)^n).$$

The $\mathsf{DiffForm}^{\uparrow}$ *root supplies the carried form row, exterior-derivative input row, and wedge routing needed to form the top row, but it does not supply this nondegeneracy predicate as a primitive.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface top wedge transport`
Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface nondegeneracy obligation`

Proof. Begin with the top wedge row named in Definition 139.1. It is assembled from the degree-one row, the exterior-derivative input row, and the wedge ladder exported by Definition 136.101.

Now use the classifier clause of the same definition. It transports the shared manifold carrier, the one-form row, the derivative input row, and the wedge ladder rows, so the nondegeneracy predicate is stated on the transported $\mathsf{Contact}^{\uparrow}$ surface.

The exclusion clause follows from Definition 136.118: $\mathsf{DiffForm}^{\uparrow}$ exports routing rows to the contact consumer face, while contact nonintegrability remains outside the shared root primitive surface. □

↪ variant `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface nondegeneracy obligation`

Theorem 139.4 (ContactUp ledger exactness obligation)**.** *The* $\mathsf{Contact}^{\uparrow}$ *ledger and exactness obligation records exactly the hidden manifold chart rows supplied by the* $\mathsf{Manifold}^{\uparrow}$ *source, the one-form source row* α*, the exterior-derivative input row* $d\alpha$*, the wedge ladder rows* $\alpha \wedge (d\alpha)^k$*, the top wedge row* $\alpha \wedge (d\alpha)^n$*, the nonintegrability witness of Definition 139.3, and a coverage, soundness, and completeness predicate for the carrier-classifier surface of Definition 139.1.*

The obligation is exact only when every certified contact read factors through that carrier-classifier surface and every listed ledger row is classified by the restricted hsame *transport named there.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactUpLedgerExactness obligation`

Proof. Unpack Definition 139.1. Its source and classifier clauses list the $\mathsf{Manifold}^{\uparrow}$ chart carrier, the $\mathsf{DiffForm}^{\uparrow}$ one-form row, the exterior-input row, the wedge ladder, the top wedge row, and the hyperplane row.

Apply Definition 139.2 to identify the contact consumer face and the routed form rows, then apply Definition 139.3 for the local nonintegrability witness.

The coverage predicate says every certified contact read lands in this displayed package. Soundness says each such read is one of the listed rows. Completeness says each listed row is available to the carrier-classifier. These three clauses are precisely the ledger exactness obligation. □

Theorem 139.5 (ContactUp public NameCert export)**.** *The carrier-classifier surface of Definition 139.1, the form-row obligation of Definition 139.2, the nondegeneracy obligation of Definition 139.3, and the ledger exactness obligation of Definition 139.4 assemble the public* $\mathsf{NameCert}_{\mathsf{Contact}^\uparrow}$ *surface over* BHist*, the* $\mathsf{Manifold}^\uparrow$ *carrier, the* $\mathsf{DiffForm}^\uparrow$ *one-form row, the exterior-input row, the wedge ladder, the top wedge, the restricted* hsame *classifier, and the exact ledger.*

This public surface exports the carried contact rows and their exact ledger. It does not export a separate nonintegrability primitive and does not include a standard bridge.

Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface public namecert export`

Proof. Project the carrier field from Definition 139.1. The carrier consists of the shared $\mathsf{Manifold}^\uparrow$ endpoint, the $\mathsf{DiffForm}^\uparrow$ degree-one form row, the exterior-input row, the wedge ladder, and the top wedge row over BHist.

Project the classifier field from the same definition. It is the restriction of hsame to transports preserving the manifold carrier, one-form row, hyperplane row, exterior-input row, and wedge-ladder rows.

Project the routed form face from Definition 139.2 and the local top-wedge condition from Definition 139.3. These two rows provide the form and nondegeneracy fields without adding a nonintegrability primitive.

Finally repack the coverage, soundness, completeness, and row list of Definition 139.4 as the exact ledger field. Because every projected row is already part of the displayed carrier-classifier surface, the public certificate contains no standard bridge row. □

Definition 139.6 (ContactUp standard bridge surface)**.** *The* $\mathsf{Contact}^\uparrow$ *standard bridge surface is the public* $\mathsf{Contact}^\uparrow$ *carrier-classifier rows, the form-row obligation, the nondegeneracy row, the ledger exactness row, and the public* NameCert *export, read as the source side of the standard contact-structure interpretation. Its source side is exactly the surface displayed by Definition 139.5.*

Lean4 (definition) `BEDC.Derived.ContactUp.ContactStandardBridgeSurface`

Theorem 139.7 (ContactUp public bridge compatibility)**.** *Every row interpreted by the bridge surface of Definition 139.6 is already one of the public* $\mathsf{Contact}^\uparrow$ *rows, and every public* $\mathsf{Contact}^\uparrow$ *row has a corresponding bridge-side row. The bridge adds no host manifold, no open-cover compactness condition, and no external equality beyond the public carrier, classifier, obligation, ledger, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactStandardBridgeSurface public bridge compatibility`
Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface public bridge compatibility`

Proof. Use Definition 139.2 for the routed one-form, exterior-input, and wedge-ladder rows, then Definition 139.3 for the contact-local top-wedge row.

Apply Definition 139.4. Its coverage, soundness, and completeness clauses say that the listed bridge reads are exactly the certified public contact reads.

Finally use Definition 139.5. It repacks those rows as the public $\mathsf{NameCert}_{\mathsf{Contact}^\uparrow}$ surface, so the correspondence is row-for-row and introduces no host manifold, compactness, or external equality datum. □

Theorem 139.8 (ContactUp mature consumer completeness)**.** *Every* $\mathsf{Contact}^\uparrow$ *consumer of the public bridge reads exactly the* $\mathsf{Manifold}^\uparrow$ *carrier, the* $\mathsf{DiffForm}^\uparrow$ *one-form row, the exterior-input row, the wedge ladder rows, the top-wedge nondegeneracy row, the exact ledger, and the public*

bridge compatibility rows already named by Definition 139.5 and Definition 139.7. No extra host manifold, compactness condition, or equality source is exported.

Lean4 ✓ `BEDC.Derived.ContactUp.ContactStandardBridgeSurface mature consumer completeness`
Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface mature consumer completeness`

Proof. First apply Definition 139.2. It identifies the contact consumer face with the routed $\mathsf{DiffForm}^{\uparrow}$ one-form, exterior-input, and wedge rows over the shared $\mathsf{Manifold}^{\uparrow}$ carrier.

Next apply Definition 139.3 and Definition 139.4. These supply the local top-wedge nondegeneracy row and the exact coverage, soundness, and completeness ledger for the displayed carrier-classifier surface.

Finally use Definition 139.5 to repack those rows as the public $\mathsf{NameCert}_{\mathsf{Contact}^{\uparrow}}$ surface, and use Definition 139.7 to compare the public surface with the bridge surface. The compatibility theorem states that the bridge adds no host manifold, compactness, or external equality datum, so the listed rows are the complete consumer boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Contact}^{\uparrow}}$.

Constructive story: A contact surface is built over a shared $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{DiffForm}^{\uparrow}$ source by displaying an odd-dimensional manifold row, a carried degree-one form α, its hyperplane row, an exterior-derivative input $d\alpha$, the wedge ladder $\alpha \wedge (d\alpha)^k$, and the top-degree wedge row. A history enters the carrier only through those routed form rows, and two accepted surfaces classify together when hsame preserves the shared manifold endpoint, the one-form, the hyperplane row, the exterior-input row, and every wedge-ladder row. The source covers exactly the manifold plus routed one-form data, the pattern records the top wedge nondegeneracy condition on dimension $2n+1$, and the ledger keeps chart rows, the one-form source, derivative input, wedge ladder, top wedge, and local nonintegrability witness aligned. The public certificate obligations are therefore the displayed carrier row, restricted classifier reflexivity, symmetry, transitivity, and closure of all listed form rows under that classifier, with coverage, soundness, and completeness supplied by the exact ledger. A history h lies in the signature gap of a $\mathsf{Contact}^{\uparrow}$ token p when the source accepts such a form surface for h, the signature relation pairs it with an answer surface and evidence ledger, and token introduction identifies that answer as p; unfolded, h carries the same odd-dimensional endpoint, one-form, derivative input, wedge ladder, and nondegenerate top wedge encoded by p. The surface deliberately reads the nonintegrability condition as this top-wedge row rather than as a separate primitive, and it exports no standard bridge. The chapter registers this routed one-form construction under the name $\mathsf{Contact}^{\uparrow}$; no closed carrier implementation, host contact manifold, or independent nonintegrability theorem is assumed.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Contact}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public boundary is exactly the carrier-classifier surface, form-row obligation, nondegeneracy obligation, and exact ledger assembled in Definition 139.5.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Contact}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a closed ContactUp carrier implementation, a proved nonintegrability theorem, Lean bridge verification, or mature examples.

Upgrade path: The paper-side theory axis is complete; Lean bridge verification remains a separate verification-axis obligation.

Definition 139.9 (ContactUp coannihilating integrability row)**.** *Let C be a $\mathsf{Contact}^{\uparrow}$ carrier-classifier surface of Definition 139.1 over a displayed dimension row $2n+1$ with $n \geq 1$. Write its shared manifold carrier as M, its carried one-form row as α, its hyperplane row as $\ker(\alpha)$, its exterior-input row as $d\alpha$, and its wedge ladder as $\alpha \wedge (d\alpha)^k$.*

A displayed row $I_{\mathcal{D}}$ is a $\mathsf{Contact}^{\uparrow}$ coannihilating integrability row on C when its contact-relevant fields are all on the same $\mathsf{Manifold}^{\uparrow}$ carrier M and consist of the following data:

- *a hyperplane-distribution row $\mathcal{D}$ classified with the displayed contact hyperplane row $\ker(\alpha)$;*
- *an annihilator-identification row $\mathsf{Ann}(\mathcal{D}) \sim \alpha$;*
- *a codimension-one Frobenius row for that annihilator, transported along $\mathsf{Ann}(\mathcal{D}) \sim \alpha$;*

- *the resulting Frobenius wedge-collapse ledger along the displayed* $\mathsf{Contact}^{\uparrow}$ *wedge ladder:*

$$\forall k\ (1 \le k \le n \Rightarrow \alpha \wedge (d\alpha)^k \sim 0).$$

The row is admitted only when these fields are transported by the restricted $\mathsf{Contact}^{\uparrow}$ *classifier, so the manifold carrier, one-form row, hyperplane row, exterior-input row, and wedge-ladder rows are the same rows read by* C*.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactCoannihilatingIntegrabilityRow transport with empty`

Theorem 139.10 (Contact top wedge excludes coannihilating integrability)**.** *Let* C *be a* $\mathsf{Contact}^{\uparrow}$ *carrier-classifier surface of Definition 139.1 over a displayed dimension row* $2n+1$ *with* $n \ge 1$*. Suppose* C *carries the top-wedge nondegeneracy row*

$$\alpha \wedge (d\alpha)^n \not\sim 0.$$

Then no $\mathsf{Contact}^{\uparrow}$ *coannihilating integrability row* $I_{\mathcal{D}}$ *of Definition 139.9 is admitted on the same carrier. Equivalently, no displayed integrable-distribution row* $\mathcal{D}$ *on the same* $\mathsf{Manifold}^{\uparrow}$ *carrier whose annihilator is classified with* α *can coexist with the displayed* $\mathsf{Contact}^{\uparrow}$ *top-wedge row.*

Lean4 ✓ `BEDC.Derived.ContactUp.ContactCarrierClassifierSurface top wedge excludes coannihilating integrability`

Proof. Assume that such a row $I_{\mathcal{D}}$ is admitted. By Definition 139.9, the distribution row lies on the same $\mathsf{Manifold}^{\uparrow}$ carrier as C, its hyperplane row is classified with $\ker(\alpha)$, and its annihilator row is classified with α.

The Frobenius part of the admitted row is therefore read on the same carried one-form row and the same exterior-input row $d\alpha$. Since $n \ge 1$, the Frobenius wedge-collapse ledger in Definition 139.9 applies to the displayed ladder index $k = n$ and gives

$$\alpha \wedge (d\alpha)^n \sim 0.$$

It remains to check that this is the same top-wedge endpoint as the contact nondegeneracy row. The carrier definition Definition 139.1 includes the one-form row, hyperplane row, exterior-input row, every wedge-ladder row, and the top wedge row on one shared manifold endpoint. The form-row obligation Definition 139.2 routes these rows through the $\mathsf{Contact}^{\uparrow}$ consumer face, and Definition 139.3 places the nondegeneracy predicate on that transported top wedge. Thus the zero endpoint obtained from $I_{\mathcal{D}}$ and the nondegenerate endpoint carried by C address the same row $\alpha \wedge (d\alpha)^n$.

The displayed nondegeneracy hypothesis says that this row is not classified with zero, while the Frobenius wedge-collapse ledger says that it is classified with zero. An admitted row cannot satisfy both predicates on the same classified endpoint. Hence the assumed $I_{\mathcal{D}}$ is not admitted. □

The $\mathsf{LieGroup}^{\uparrow}$ packet consumes the algebraic source of chapter 41 and the chart source of chapter 96; the child routes below expose how those dependencies enter the certificate.

Chapter 140

A Concrete Naming Certificate for $\mathsf{LieGroup}^{\uparrow}$

The Lie-group interface $\mathsf{LieGroup}^{\uparrow}$ packages a group carrier whose underlying set is a manifold and whose multiplication and inverse are constructively smooth. It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96), and is the substrate for representation, exponential, and root-system certificates.

140.1 Carrier definition

Definition 140.1 (Singleton smooth group carrier)**.** *The concrete* $\mathsf{LieGroup}^{\uparrow}$ *carrier is the shared singleton history carrier*

$$L_{\mathsf{e}}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e})$$

on BHist*. Its group rows are the singleton* $\mathsf{Group}^{\uparrow}$ *rows*

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{inv}(h) := \mathsf{e}, \qquad e := \mathsf{e},$$

with classifier

$$R_{\mathsf{e}}(h, k) :\Longleftrightarrow L_{\mathsf{e}}(h) \wedge L_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Its smooth-manifold rows are the singleton $\mathsf{Manifold}^{\uparrow}$ *chart rows with the one chart* χ_{e}*, top domain* L_{e}*, constant value* $0_{\mathsf{Real}^{\uparrow}}$*, and the self-overlap constant transition. Thus the algebraic carrier and the manifold carrier are the same predicate on histories, not two transported copies.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupTasteGate single carrier alignment`

140.2 The certificate

Theorem 140.2 (LieGroup singleton carrier obligation)**.** *The carrier* L_{e} *of Definition 140.30 simultaneously supplies the* $\mathsf{Group}^{\uparrow}$ *singleton carrier and classifier rows and the* $\mathsf{Manifold}^{\uparrow}$ *singleton chart rows. Concretely, every accepted history is classified with* e *by* R_{e}*, multiplication and inverse stay inside* L_{e}*, and the sole chart domain is exactly the same carried predicate* L_{e}*.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton carrier obligation`

Proof. For the algebraic side, unfold Definition 140.30. The carrier, classifier, multiplication, unit, and inverse rows are exactly the rows used by Definition 41.4; the checked Lean row is BEDC.Derived.GroupUp.GroupSingletonHistory_laws.

For the manifold side, the chart-domain row is the singleton empty-chart row of Definition 96.32. Its carried predicate is $\mathsf{hsame}(h, \mathsf{e})$, which is definitionally the same predicate as L_{e}.

The two sides therefore share the same history carrier and the same restricted hsame classifier. Projection from the group row gives the algebraic carrier and classifier fields, while projection from the manifold row gives the chart domain, chart value, and singleton atlas fields. □

Theorem 140.3 (LieGroup singleton operation smoothness obligation)**.** *For histories carried by L_{e}, both* mul *and* inv *of Definition 140.30 are smooth operations in the singleton atlas. Every accepted multiplication result and every accepted inverse result is classifier-equal to* e*, so each operation is read by the sole chart as a constant transition on the singleton overlap.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton operation smoothness`

Proof. Start with multiplication. If $L_{\mathsf{e}}(h)$ and $L_{\mathsf{e}}(k)$ hold, the group singleton laws identify $\mathsf{append}(h, k)$ with e and keep the result inside the singleton carrier. This is the multiplication-closure row of BEDC.Derived.GroupUp.GroupSingletonHistory_laws.

For inverse, $\mathsf{inv}(h)$ is definitionally e. The inverse-closure row of the same group theorem gives the carrier witness, and the classifier row identifies the result with the unit endpoint.

Now read both operations through the manifold atlas. By Definition 96.32, every carried input and every accepted output lies in the only chart domain and has the constant chart value $0_{\mathsf{Real}^{\uparrow}}$. The overlap transition is therefore the singleton constant transition of Definition 96.33, whose checked Lean row is BEDC.Derived.ManifoldUp.ManifoldSingleton_transition_smoothness. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton operation smoothness obligation`
Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingletonOperation smoothness`

Definition 140.4 (LieGroup singleton public smooth group package)**.** *The public smooth group package underlying the singleton* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *has BEDC source* BHist*, carrier L_{e}, classifier R_{e}, multiplication graph*

$$\mathsf{mul}(h, k) = \mathsf{append}(h, k),$$

inverse

$$\mathsf{inv}(h) = \mathsf{e},$$

and unit e*. The smooth rows use the singleton chart χ_{e} on the same carrier predicate, with constant chart value $0_{\mathsf{Real}^{\uparrow}}$ and self-overlap transition certified by* Cont*. Thus every public algebraic operation is read through the displayed BEDC operation rows rather than through an external standard-space representative.*

Theorem 140.5 (LieGroup singleton classifier transport)**.** *For the package of Definition 140.33, classifier equivalence is congruent for multiplication and inverse. If $R_{\mathsf{e}}(h, h')$ and $R_{\mathsf{e}}(k, k')$, then*

$$R_{\mathsf{e}}(\mathsf{mul}(h, k), \mathsf{mul}(h', k')).$$

If $R_{\mathsf{e}}(h, h')$, then

$$R_{\mathsf{e}}(\mathsf{inv}(h), \mathsf{inv}(h')).$$

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton classifier transport`

Proof. Project the endpoint carrier witnesses from $R_{\mathsf{e}}(h, h')$ and $R_{\mathsf{e}}(k, k')$. Each projection gives membership in L_{e} together with an hsame path to the empty endpoint.

For multiplication, the singleton group laws identify both $\mathsf{append}(h, k)$ and $\mathsf{append}(h', k')$ with e while keeping both results in L_{e}. The classifier row is then the pair of carrier witnesses plus the hsame comparison obtained by transitivity through e.

For inverse, both operation outputs are definitionally e. Reflexivity gives the carrier witnesses and the endpoint comparison, so the restricted classifier R_{e} holds for the two inverse outputs. □

Theorem 140.6 (LieGroup singleton chart operation readback)**.** *The multiplication and inverse outputs covered by Definition 140.34 are read by the singleton chart as the constant endpoint* $0_{\mathsf{Real}^{\uparrow}}$*. Consequently their chart self-overlap row is the same singleton transition row certified for* $\mathsf{Manifold}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton chart operation readback`

Proof. Apply Definition 140.34 to obtain the classifier witnesses for the operation outputs. In each case the classifier contains the carrier predicate L_{e} for the output history.

Apply Definition 96.32 to those carrier witnesses. The unique chart χ_{e} reads every carried endpoint as the constant value $0_{\mathsf{Real}^{\uparrow}}$, so multiplication and inverse have the same chart readback.

Finally apply Definition 96.33. Since the only overlap is the singleton self-overlap and both chart values are constant, the operation readback has exactly the displayed transition row. □

Theorem 140.7 (LieGroup singleton public NameCert export)**.** *The public* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *surface for the singleton smooth group exports the following ten fields.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e*.*

Carrier. *The predicate* $L_{\mathsf{e}}(h)$*.*

Classifier. *The restricted classifier* $R_{\mathsf{e}}(h, k)$*.*

Algebraic rows. *Multiplication* append*, inverse* e*, unit* e*, and the singleton* $\mathsf{Group}^{\uparrow}$ *laws.*

Smooth rows. *The singleton* $\mathsf{Manifold}^{\uparrow}$ *chart* χ_{e}*, constant value* $0_{\mathsf{Real}^{\uparrow}}$*, and the singleton transition row.*

Classifier transport. *Multiplication and inverse preserve* R_{e} *as in Definition 140.34.*

Smooth operation congruence. *Operation outputs related by the classifier have the same chart readback and transition row as in Definition 140.35.*

Exactness. *Every accepted public row reduces to* BHist, hsame, Cont, append, e*, the displayed singleton* $\mathsf{Group}^{\uparrow}$ *laws, and the displayed singleton* $\mathsf{Manifold}^{\uparrow}$ *chart and transition.*

Bridge and status. *No* $\mathsf{StandardBridge}$ *row is claimed; the paper-side target is under* $\mathsf{BEDC.Derived.LieGroupUp}$*.*

Proof. The carrier and classifier fields are supplied by Definition 140.30, and the public operation symbols are fixed in Definition 140.33.

The algebraic rows are the singleton $\mathsf{Group}^{\uparrow}$ laws. They give the endpoint carrier facts and identify multiplication and inverse outputs with e.

The smooth rows are the singleton $\mathsf{Manifold}^{\uparrow}$ rows. Chart coverage reads every accepted endpoint as $0_{\mathsf{Real}^{\uparrow}}$, and transition smoothness gives the constant self-overlap row.

The transport and smooth-operation congruence rows are exactly Definition 140.34 and Definition 140.35. Repacking these rows gives the ten-field public certificate surface, with exactness reduced to the listed singleton BEDC data and with no standard bridge component. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton public namecert export`

Theorem 140.8 (LieGroup singleton multiplication smooth homomorphism)**.** *On the concrete carrier L_{e} of Definition 140.30, the multiplication row*

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k)$$

is carried by L_{e} whenever both inputs are carried, is classifier-respecting in both arguments for R_{e}, and is read by the singleton chart as the constant transition on the sole overlap.

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton multiplication smooth homomorphism`

Proof. The carrier and classifier boundary is the singleton history certificate of Definition 41.4, with checked row BEDC.Derived.GroupUp.GroupSingletonHistory_laws. Its multiplication closure row sends carried h and k to a carried $\mathsf{append}(h, k)$, and its singleton classifier rows identify the result with the endpoint e.

Classifier respect in the first and second arguments is therefore the same singleton calculation: replacing either input by an R_{e}-equal carried input leaves both histories classifier-equal to e, so the product row is again R_{e}-equal to the same endpoint.

Finally apply Definition 96.32 to place the product in the unique chart domain. The self-overlap transition is the constant transition of Definition 96.33, so the chart readback of multiplication is smooth and constant. □

Theorem 140.9 (LieGroup singleton inverse smooth endomap)**.** *On the concrete carrier L_{e} of Definition 140.30, the inverse row*

$$\mathsf{inv}(h) := \mathsf{e}$$

is carried by L_{e}, is classifier-respecting for R_{e}, is involutive up to R_{e}, and is read in the singleton atlas as a constant smooth endomap.

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton inverse smooth endomap`

Proof. The inverse value is definitionally e. The inverse closure and inverse law rows of BEDC.Derived.GroupUp.Grou cited by Definition 41.4, provide the carried witness and identify the inverse endpoint with the singleton classifier class.

Classifier respect follows because every carried input is already R_{e}-equal to e, and applying inv returns e again. Applying the same row twice again returns e, so the double inverse is R_{e}-equal to the original carried singleton point.

The chart statement is the singleton manifold statement. By Definition 96.32, both the input and inverse value lie in the only chart domain; by Definition 96.33, the readback is the constant smooth self-overlap transition. □

Theorem 140.10 (LieGroup singleton smooth-group NameCert export)**.** *The carrier obligation Definition 140.31, the multiplication homomorphism theorem Definition 140.37, and the inverse endomap theorem Definition 140.38 supply the concrete $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ export over BHist. The export contains the carrier L_{e}, classifier R_{e}, smooth multiplication, smooth inverse, ledger rows, exactness condition, and the absence of standard bridge rows.*

Proof. The carrier theorem fixes the shared algebraic and manifold predicate on BHist and supplies the classifier row. Hence every exported row has the same underlying object boundary and no separate carrier transport is needed.

The multiplication theorem supplies the smooth binary operation row: closure in L_{e}, respect for R_{e} in each argument, and constant singleton chart readback. The inverse theorem supplies the smooth unary operation row: closure, classifier respect, double-inverse agreement up to R_{e}, and the same constant chart readback.

The ledger is therefore exactly the displayed singleton ledger consisting of BHist, hsame, Cont, append, e, the singleton $\mathsf{Group}^{\uparrow}$ laws, and the singleton $\mathsf{Manifold}^{\uparrow}$ chart and transition rows. Exactness says that an accepted smooth-group row is one of these rows after unfolding the concrete carrier. No $\mathsf{StandardBridge}$ row is part of this export. □

Definition 140.11 (LieGroup singleton translation and conjugation rows)**.** *For a carried singleton parameter s and carried singleton input x, the action rows attached to Definition 140.30 are*

$$\mathsf{Left}_s(x) := \mathsf{mul}(s, x), \qquad \mathsf{Right}_s(x) := \mathsf{mul}(x, s),$$

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

The adjoint readback row is the differential of Conj_s at e, read on the singleton tangent carrier as the constant zero endomap of the singleton $\mathsf{LieAlgebra}^{\uparrow}$ carrier. Thus translation, conjugation, and adjoint readback all use the same carried predicate L_{e} and the same singleton chart endpoint.

Lean4 (definition) `BEDC.Derived.LieGroupUp.LieGroupSingletonTranslationRows`

Theorem 140.12 (LieGroup singleton left-translation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Left}_s(x)$ from Definition 140.40 is carried by L_{e}, is classified by the singleton classifier with endpoint e, and is read in the singleton atlas as the constant smooth transition on the sole overlap.*

Proof. The group row of $\mathsf{BEDC.Derived.GroupUp.GroupSingletonHistory_laws}$ gives closure of $\mathsf{mul}(s, x)$ under the singleton carrier whenever both inputs are carried by L_{e}. This is exactly the left-translation carrier row after unfolding Definition 140.40.

By Definition 140.31, the carried group result is the same object as the singleton manifold point. Therefore the sole chart domain accepts the result and reads it at the constant endpoint $0_{\mathsf{Real}^{\uparrow}}$.

The overlap transition is the singleton constant transition supplied by Definition 96.33. Hence the left translation is smooth as a singleton-atlas self-map. □

Theorem 140.13 (LieGroup singleton right-translation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Right}_s(x)$ from Definition 140.40 is carried by L_{e}, is classified by the singleton classifier with endpoint e, and is read in the singleton atlas as the constant smooth transition on the sole overlap.*

Proof. The multiplication-closure row of $\mathsf{BEDC.Derived.GroupUp.GroupSingletonHistory_laws}$ applies symmetrically to $\mathsf{mul}(x, s)$. Since $L_{\mathsf{e}}(x)$ and $L_{\mathsf{e}}(s)$ hold, the right-translation value remains inside the singleton carrier.

The carrier identification in Definition 140.31 then moves this accepted group value into the singleton chart domain without changing the predicate being used. The classifier endpoint is still e.

Finally, Definition 96.33 reads the only self-overlap as a constant smooth transition. Thus the right translation is a smooth singleton-atlas self-map. □

Theorem 140.14 (LieGroup singleton conjugation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Conj}_s(x)$ from Definition 140.40 is carried by L_{e}, classified with endpoint e, and read by the singleton atlas as a constant smooth self-map.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation smooth`

Proof. First apply Definition 140.41 to s and x. The result $\mathsf{mul}(s, x)$ is carried by L_{e} and is read in the singleton chart as the constant endpoint.

Next use the inverse-closure row from BEDC.Derived.GroupUp.GroupSingletonHistory_laws on the carried parameter s. Since $\mathsf{inv}(s)$ is definitionally e, it is also accepted by the singleton carrier and classifier.

Apply the right-translation row Definition 140.42 to the carried value $\mathsf{mul}(s, x)$ and the carried inverse $\mathsf{inv}(s)$. The resulting history is precisely $\mathsf{Conj}_s(x)$ after unfolding the definition, so it is carried, classified with e, and read as the singleton constant smooth self-map. □

Theorem 140.15 (LieGroup singleton adjoint readback)**.** *For every carried singleton parameter s, the differential at e of the singleton conjugation row Conj_s is the constant zero endomap on the singleton tangent carrier, equivalently on the singleton $\mathsf{LieAlgebra}^{\uparrow}$ carrier.*

Proof. By Definition 140.43, conjugation by any carried singleton parameter is read in the unique chart as a constant self-map at endpoint $0_{\mathsf{Real}^{\uparrow}}$. A constant singleton chart map has the zero tangent readback on the sole tangent point.

The $\mathsf{LieAlgebra}^{\uparrow}$ singleton rows identify the Lie-algebra carrier with the same singleton tangent carrier at e. Therefore the conjugation differential is the constant zero endomap on that carrier, which is exactly the adjoint readback row. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton adjoint readback`

Theorem 140.16 (LieGroup singleton shared source scope)**.** *For the carrier of Definition 140.30, every displayed $\mathsf{LieGroup}^{\uparrow}$ row is scoped by the same BEDC history source. The group carrier, manifold chart domain, restricted classifier, unit, multiplication, inverse, chart value, and transition rows are all given by BHist, hsame, e, append, Cont, the singleton $\mathsf{Group}^{\uparrow}$ laws, and the singleton $\mathsf{Manifold}^{\uparrow}$ rows. No row requires equality of host-language functions; the only equality test exposed to the certificate is the restricted hsame classifier on carried histories.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton shared source scope`

Proof. Unfold the carrier definition. The algebraic carrier is the predicate $L_{\mathsf{e}}(h)$, and the chart domain is the same predicate rather than a second copy. The classifier row is therefore the displayed relation R_{e}, which mentions only L_{e} and hsame.

The group operation rows are the singleton rows: the unit is e, multiplication is append, and inverse is the constant map to e. Their closure and law witnesses are exactly the singleton $\mathsf{Group}^{\uparrow}$ laws.

The manifold rows use the singleton chart, constant value $0_{\mathsf{Real}^{\uparrow}}$, and the self-overlap transition controlled by Cont. Since both sides name the same BHist source, every interface projection is scoped by BHist and by the restricted classifier, with no appeal to host function equality. □

Theorem 140.17 (LieGroup singleton smooth operation readback)**.** *For histories carried by L_{e}, the multiplication output $\mathsf{append}(h, k)$ and the inverse output e are classifier-equal to e. Reading either output through the sole singleton chart gives the constant value $0_{\mathsf{Real}^{\uparrow}}$, and every operation overlap is the constant self-transition of the singleton atlas.*

Proof. Apply Definition 41.4. Its singleton history laws give closure of $\mathsf{append}(h, k)$, closure of e as the inverse output, and classifier equality of both accepted outputs with the unit endpoint.

Apply Definition 96.32. Every accepted input and accepted output lies in the only chart domain, and the chart value on that domain is the constant $0_{\mathsf{Real}^{\uparrow}}$.

Apply Definition 96.33. The only overlap available in the singleton atlas is the self-overlap, so the operation readback through the chart is the constant transition row already supplied by the manifold certificate. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton smooth operation readback`

Theorem 140.18 (LieGroup singleton dependent interface package)**.** *Any downstream* ExpMap*,* AdjointRep*, or* RootSystem *consumer of the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package may project only the displayed unit, multiplication, inverse, chart, transition, and ledger rows. These projections are stable under the restricted* hsame *classifier: replacing an accepted history by a classifier-equal accepted history preserves the projected carrier witness, operation witness, chart-domain witness, chart value, and transition witness.*

Proof. The allowed projections are exactly the rows named in Definition 140.30. They expose the unit, the append multiplication, the constant inverse, the singleton chart, the constant transition, and the ledger rows BHist, hsame, e, append, and Cont.

Stability follows from the shared source scope of Definition 140.45. Every projected row is defined over L_{e} and R_{e}, so classifier-equal accepted histories remain in the same carried predicate and have the same singleton chart value.

The smooth readback theorem supplies the operation-facing part of the interface. Multiplication and inverse outputs return to the singleton endpoint, and the only chart transition available to the consumer is the constant self-transition. □

Theorem 140.19 (LieGroup singleton downstream unblock package)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *certificate consists of the following ten-field roadmap package.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e*.*

Carrier. *The predicate* $L_{\mathsf{e}}(h)$*, equivalently* $\mathsf{hsame}(h, \mathsf{e})$*.*

Classifier. *The restricted classifier* R_{e} *on accepted histories.*

Stability obligations. *Carrier reuse, operation closure, chart-domain reuse, chart-value constancy, transition constancy, and classifier transport under* hsame*.*

Ledger. BHist*,* hsame*,* e*,* append*,* Cont*, the singleton* $\mathsf{Group}^{\uparrow}$ *laws, and the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart and transition rows.*

Core theorem list. *Definition 140.31, Definition 140.32, Definition 140.45, Definition 140.46, Definition 140.47, and Definition 140.49.*

Exactness predicate. *A projected Lie-group row is accepted exactly when it factors through the displayed carrier, classifier, operation, chart, and transition rows.*

Standard bridge absent. *No* StandardBridge *row is part of this package.*

Status. *A paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *target for* BEDC.Derived.LieGroupUp*.*

Proof. The traditional target, BEDC source, carrier, and classifier fields are the surface rows of Definition 140.30 and Definition 140.31. They identify the algebraic and manifold carriers with the same singleton history predicate.

The stability and ledger fields are supplied by Definition 140.45 and Definition 140.46. These theorems name the shared BHist source, the restricted hsame classifier, the append operation, the constant inverse, and the constant chart transition.

The dependent-interface field is supplied by Definition 140.47. It records the rows that later ExpMap, AdjointRep, and RootSystem packages may project, and it states their stability under the restricted classifier.

The exactness, bridge, and status fields are bookkeeping consequences of those rows: a row belongs to the singleton package only when it factors through the displayed carrier and operation data, no standard bridge is included, and the target namespace is BEDC.Derived.LieGroupUp. □

Theorem 140.20 (LieGroup singleton NameCert obligation surface)**.** *The rows of Definition 140.30, Definition 140.33, Definition 140.31, Definition 140.32, Definition 140.34, Definition 140.35, Definition 140.36, Definition 140.37, Definition 140.38, Definition 140.39, Definition 140.40, Definition 140.41, Definition 140.42, Definition 140.43, Definition 140.44, Definition 140.45, Definition 140.46, and Definition 140.47 assemble the paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *obligation surface for the singleton smooth group carrier.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e*.*

Carrier. *The predicate* $L_{\mathsf{e}}(h)$*, equivalently* $\mathsf{hsame}(h, \mathsf{e})$*.*

Classifier. *The restricted singleton classifier* R_{e}*.*

Stability. *Carrier transport, group operation closure, classifier transport for multiplication and inverse, chart-domain transport, chart-value constancy, smooth multiplication congruence, smooth inverse congruence, transition-row constancy, left and right translation smoothness, conjugation smoothness, and adjoint readback constancy.*

Ledger. *The rows* BHist*,* hsame*,* Cont*,* append*,* e*, the singleton* $\mathsf{Group}^{\uparrow}$ *laws, the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart, the singleton constant transition, translation rows, conjugation rows, and the zero singleton adjoint readback.*

Core theorem list. *Definition 41.4, Definition 96.37, Definition 140.31, Definition 140.32, Definition 140.34, Definition 140.35, Definition 140.36, Definition 140.37, Definition 140.38, Definition 140.39, Definition 140.41, Definition 140.42, Definition 140.43, Definition 140.44, Definition 140.45, Definition 140.46, and Definition 140.47.*

Exactness. *A Lie-group operation row is accepted only when its carrier, classifier, chart-domain, and transition witnesses all reduce to the displayed singleton rows.*

No standard bridge. *The surface contains no* StandardBridge *row.*

Status. *A paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *obligation target under* BEDC.Derived.LieGroupUp*.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton semantic name certificate`

Proof. The carrier theorem supplies the shared object boundary: one predicate L_{e} is both the group carrier and the manifold point carrier. This prevents any separate transport problem between algebraic and smooth rows.

The group rows are inherited from Definition 41.4, backed by BEDC.Derived.GroupUp.GroupSingletonHistory_law They give inhabitance, multiplication closure, inverse closure, unit laws, and inverse laws for the append operation with constant inverse.

The manifold rows are inherited from Definition 96.37, backed by BEDC.Derived.ManifoldUp.ManifoldSingleton_s and the singleton transition smoothness row. They give the sole chart domain, constant chart value, and constant self-overlap transition.

The operation-smoothness, classifier-transport, multiplication, inverse, public export, and smooth-group export theorems tie these projections together. Once a group operation result is accepted, it is again classifier-equal to e, hence it is read by the only chart as the same constant endpoint. The translation, conjugation, and adjoint rows are the same singleton readback applied to the displayed public operation rows. Repacking the listed fields gives the obligation surface in the statement, with no standard bridge component claimed. □

Theorem 140.21 (LieGroup singleton standard bridge)**.** *The singleton BEDC* $\mathsf{LieGroup}^{\uparrow}$ *packet of Definition 140.33 interprets as the standard one-point Lie group. The bridge sends the carrier* L_{e} *to the singleton carrier, reads multiplication and inverse from the singleton* $\mathsf{Group}^{\uparrow}$ *rows, and reads smoothness from the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart rows. It carries no non-singleton Lie theory, exponential theory, representation theory, root-system theory, or host quotient data.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton standard bridge`

Proof. The carrier bridge is the singleton carrier of Definition 140.31. Every accepted history is R_{e}-classified with e, so the target carrier has one point.

The algebraic operation bridge is supplied by Definition 140.37 and Definition 140.38. Multiplication and inverse stay in L_{e} and are classifier-equal to the singleton endpoint.

The smooth bridge is supplied by Definition 140.35. The only chart is the singleton chart, and every operation readback is the constant smooth self-overlap row. Repacking these rows gives the standard one-point Lie group interpretation and no further Lie-group structure. □

Theorem 140.22 (LieGroup singleton source lock)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *source uses the single predicate* $L_{\mathsf{e}}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e})$ *as the* $\mathsf{Group}^{\uparrow}$ *carrier, the* $\mathsf{Manifold}^{\uparrow}$ *point and chart-domain carrier, and the* $\mathsf{LieGroup}^{\uparrow}$ *carrier. Its classifier is the same restricted* hsame *row* $R_{\mathsf{e}}(h, k)$ *from Definition 140.30; no transported second source, auxiliary carrier, or duplicate classifier is introduced.*

Proof. Unfold Definition 140.30. The displayed predicate L_{e} is the only carrier predicate named there, and the classifier is the displayed restricted row R_{e}.

Project the singleton group rows from Definition 41.4. These rows use BHist, e, append, constant inverse, and the restricted hsame classifier; their checked source is BEDC.Derived.GroupUp.GroupSingletonHistory_law

Project the singleton chart rows from Definition 96.37. The chart domain is the same predicate $\mathsf{hsame}(h, \mathsf{e})$, the chart value is constant, and the overlap row is the singleton transition row.

Thus both projected packages identify their visible carrier with L_{e}. The $\mathsf{LieGroup}^{\uparrow}$ package is assembled by sharing that predicate and classifier rather than by transporting data across a second source. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton source lock`

Theorem 140.23 (LieGroup manifold and group anchor)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *source is accepted only after the* $\mathsf{Group}^{\uparrow}$ *operation rows of chapter 41 and the* $\mathsf{Manifold}^{\uparrow}$ *chart rows of chapter 96 have been read. Its public source ledger projects the shared predicate* L_{e} *through those two dependency surfaces and exports no independent ambient smooth-group carrier.*

Proof. Begin with Definition 140.51. The carrier predicate L_{e} and the restricted classifier R_{e} are the single source used by the group carrier, manifold point and chart-domain carrier, and Lie-group carrier.

Project the $\mathsf{Group}^{\uparrow}$ rows through chapter 41. The operation, unit, inverse, and law rows are the singleton history rows named by the source-lock theorem, so the Lie-group packet cannot be accepted before those algebraic rows are visible.

Project the $\mathsf{Manifold}^{\uparrow}$ rows through chapter 96. The chart domain, chart value, and transition row are the singleton chart rows named by the same source lock, so the smooth side is read from the displayed manifold surface rather than from a separate smooth carrier.

Therefore the source ledger is a joint read of the displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Manifold}^{\uparrow}$ dependency rows. Since every exported carrier and classifier is the shared L_{e} and R_{e} projection, no independent ambient smooth-group carrier is available to downstream consumers. □

Theorem 140.24 (LieGroup singleton exact ledger)**.** *For the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package, the accepted ledger consists exactly of* BHist, hsame, e, append/Cont, *the singleton* $\mathsf{Group}^{\uparrow}$ *laws, the singleton chart domain and chart value, and the singleton constant transition row. It excludes any standard bridge, any host smooth-manifold equality, any extra carrier, and any external topology row.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton exact ledger`

Proof. The source-lock theorem fixes the boundary of the ledger: all carrier and classifier rows are projections of the one history predicate L_{e} and the restricted hsame relation.

The algebraic entries are exactly the singleton $\mathsf{Group}^{\uparrow}$ rows. They are supplied by Definition 41.4 and BEDC.Derived.GroupUp.GroupSingletonHistory_laws, namely BHist, hsame, e, append, Cont, and the singleton group laws.

The smooth entries are exactly the singleton $\mathsf{Manifold}^{\uparrow}$ rows. They are supplied by Definition 96.37, BEDC.Derived.ManifoldUp.ManifoldSingleton_semanticNameCert, and BEDC.Derived.ManifoldUp.Manifol the singleton chart domain, constant chart value, and constant transition.

Finally, Definition 140.32 accounts for the operation rows by reading multiplication and inverse through the same singleton atlas. No cited row provides a StandardBridge, host equality of smooth manifolds, an additional carrier predicate, or an external topology, so none belongs to the accepted ledger. □

Theorem 140.25 (LieGroup singleton downstream export)**.** *The packages* $\mathsf{ExpMap}^{\uparrow}$, $\mathsf{AdjointRep}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, *and representation certificates may consume the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package only through the exported carrier, classifier, operation, chart, and transition rows. The singleton package makes no downstream claim of connectedness, completeness, nontrivial Lie algebra, or standard bridge.*

Proof. By Definition 140.51, every visible carrier and classifier row of the singleton $\mathsf{LieGroup}^{\uparrow}$ package is a projection of the shared history predicate L_{e} and the restricted hsame classifier. These are the only source rows a dependent certificate can name.

By Definition 140.53, the remaining visible rows are precisely the group operation rows, singleton chart rows, and singleton transition row. The NameCert surface in Definition 140.49 repacks those rows as the public obligation surface for $\mathsf{LieGroup}^{\uparrow}$.

Therefore any $\mathsf{ExpMap}^{\uparrow}$, $\mathsf{AdjointRep}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, or representation certificate built over this singleton source can project only those exported rows. Since the ledger contains no connectedness, completeness, nontrivial Lie-algebra, or $\mathsf{StandardBridge}$ row, no downstream consumer receives such a claim from this package. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton downstream export`

Definition 140.26 (Singleton LieGroup conjugation action)**.** *On the singleton smooth group carrier Definition 140.30, define the* BHist*-valued conjugation action by*

$$\mathsf{Ad}_{\mathsf{e}}(s, x) := \mathsf{append}(\mathsf{append}(s, x), \mathsf{e}).$$

The inputs are carried histories $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$, and the terminal factor e *is the singleton inverse row of the group carrier.*

Lean4 (definition) `BEDC.Derived.LieGroupUp.LieGroupSingletonConjugationAction`

Theorem 140.27 (LieGroup singleton conjugation carrier and smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Ad}_{\mathsf{e}}(s, x)$ is again carried by L_{e}. Moreover the singleton manifold chart reads this conjugation endpoint as the same constant smooth transition that reads multiplication and inverse in Definition 140.32.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation carrier and smoothness`

Proof. Unfold Definition 140.55. The endpoint is the append-composite $(s * x) * \mathsf{e}$ in the singleton group rows. Since s and x are carried, Definition 41.4 gives closure of $s*x$, and the right multiplication by e stays in the same carrier. The Lean-side row BEDC.Derived.GroupUp.GroupSingletonClassifier_conjugation_ records the same terminal collapse for the singleton classifier.

For smoothness, the carried input pair and the carried output all lie in the sole singleton chart domain. Applying Definition 96.33 reads the source and target charts as the constant value $0_{\mathsf{Real}^{\uparrow}}$, so the displayed action has the constant smooth transition row already used by the multiplication and inverse obligations. The corresponding Lean-side row is BEDC.Derived.ManifoldUp.ManifoldSingleton_transition □

Theorem 140.28 (LieGroup singleton conjugation action law)**.** *For carried histories $L_{\mathsf{e}}(s)$, $L_{\mathsf{e}}(t)$, and $L_{\mathsf{e}}(x)$, the two iterated endpoints*

$$\mathsf{Ad}_{\mathsf{e}}(s, \mathsf{Ad}_{\mathsf{e}}(t, x))$$

and

$$\mathsf{Ad}_{\mathsf{e}}(\mathsf{append}(s, t), x)$$

are both classified by the singleton classifier R_{e}, hence represent the same carried singleton endpoint.

Proof. First unfold the action on the left. The inner endpoint is $\mathsf{append}(\mathsf{append}(t, x), \mathsf{e})$, and Definition 140.56 carries it back into L_{e}. Applying the same theorem to s and this inner endpoint therefore carries the whole left-hand endpoint.

Next unfold the action on the right. Group singleton closure carries $\mathsf{append}(s, t)$, and another application of Definition 140.56 carries $\mathsf{Ad}_{\mathsf{e}}(\mathsf{append}(s, t), x)$.

It remains to compare the classifier rows. By the singleton group classifier, every carried endpoint is hsame to e. The Cont associativity and readback rows allow the nested append words on both sides to be read as singleton terminal words, and the group singleton closure removes the terminal factor. Thus both displayed endpoints satisfy R_{e} against e, and therefore against each other by the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation action law`

Theorem 140.29 (LieGroup singleton AdjointRep source surface)**.** *The rows of Definition 140.55, Definition 140.56, and Definition 140.57 supply the* $\mathsf{LieGroup}^{\uparrow}$*-side source surface consumed by* $\mathsf{AdjointRep}^{\uparrow}$*: the operation* Ad_{e} *is* BHist*-valued, classifier-stable on the singleton carrier, smooth in the singleton atlas, and carries no standard bridge row.*

Proof. The definition provides the concrete BHist operation. The carrier and smoothness theorem supplies the two structural rows needed by a representation source: accepted inputs give an accepted endpoint, and the singleton atlas reads that endpoint through the constant smooth transition.

The action-law theorem supplies the compatibility row. Both iterated conjugation and conjugation by the product word land in the singleton classifier class, so $\mathsf{AdjointRep}^{\uparrow}$ can consume the action without adding a separate carrier transport or chart transition.

No standard bridge row is part of this surface. The package is only the paper-side singleton $\mathsf{LieGroup}^{\uparrow}$ action source over BHist, with the ordinary Lie-group bridge left outside the displayed rows. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton adjointrep source surface`

Closure status for $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$**.**

Constructive story: The concrete Lie group here is the singleton history carrier L_{e}: a history is accepted exactly when it is hsame to e, and the same predicate serves simultaneously as the $\mathsf{Group}^{\uparrow}$ carrier, the $\mathsf{Manifold}^{\uparrow}$ point carrier, and the $\mathsf{LieGroup}^{\uparrow}$ carrier. Two accepted histories classify together by the restricted row R_{e}, so multiplication by append, inverse returning e, the unit e, and the singleton chart all stay inside one shared classifier rather than being transported across duplicate sources. The source covers the single BHist carrier and its group and chart rows, the pattern records singleton group laws, constant chart coverage, smooth multiplication and inverse readback, and conjugation source rows, while the ledger keeps e, hsame, Cont, append, the singleton chart value, and the constant transition visible. Nonemptiness is witnessed by e, reflexivity, symmetry, and transitivity are those of hsame restricted to L_{e}, and carrier closure follows because every public operation endpoint is again classified with e. A history h lies in the signature gap of a $\mathsf{LieGroup}^{\uparrow}$ token p when the source accepts h as a singleton group/manifold row, the signature relation pairs it with an answer endpoint and evidence ledger, and token introduction names that answer as p; unfolded, h and the answer are both the empty singleton endpoint with the displayed operation, chart, and conjugation rows. The paper bridge reads this package as the standard one-point Lie group and reads it back through the same singleton carrier, group laws, and chart rows. The chapter registers this singleton smooth-group construction under the name $\mathsf{LieGroup}^{\uparrow}$; no non-singleton Lie group, exponential theory, representation theory, or root-system theory is part of the primitive package.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public singleton boundary is the shared BHist carrier L_{e}, restricted hsame classifier, singleton $\mathsf{Group}^{\uparrow}$ laws, singleton $\mathsf{Manifold}^{\uparrow}$ chart rows, classifier transport, smooth operation readback, exact ledger, downstream export, and conjugation source rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LieGroupUp.LieGroupSingleton\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not close non-singleton Lie groups, exponential theory, representation theory, root-system theory, or bridgeChecked verification.

Upgrade path: The remaining advancement is bridgeChecked verification for the displayed paper bridge, not additional theory-axis maturity for the singleton Lie-group package.

140.3 Carrier definition

Definition 140.30 (Singleton smooth group carrier)**.** *The concrete* $\mathsf{LieGroup}^{\uparrow}$ *carrier is the shared singleton history carrier*

$$L_{\mathsf{e}}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e})$$

on BHist*. Its group rows are the singleton* $\mathsf{Group}^{\uparrow}$ *rows*

$$\mathsf{mul}(h, k) := \mathsf{append}(h, k), \qquad \mathsf{inv}(h) := \mathsf{e}, \qquad e := \mathsf{e},$$

with classifier

$$R_{\mathsf{e}}(h, k) :\Longleftrightarrow L_{\mathsf{e}}(h) \wedge L_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h, k).$$

Its smooth-manifold rows are the singleton $\mathsf{Manifold}^{\uparrow}$ *chart rows with the one chart* χ_{e}*, top domain* L_{e}*, constant value* $0_{\mathsf{Real}^{\uparrow}}$*, and the self-overlap constant transition. Thus the algebraic carrier and the manifold carrier are the same predicate on histories, not two transported copies.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupTasteGate single carrier alignment`

140.4 The certificate

Theorem 140.31 (LieGroup singleton carrier obligation)**.** *The carrier* L_{e} *of Definition 140.30 simultaneously supplies the* $\mathsf{Group}^{\uparrow}$ *singleton carrier and classifier rows and the* $\mathsf{Manifold}^{\uparrow}$ *singleton chart rows. Concretely, every accepted history is classified with* e *by* R_{e}*, multiplication and inverse stay inside* L_{e}*, and the sole chart domain is exactly the same carried predicate* L_{e}*.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton carrier obligation`

Proof. For the algebraic side, unfold Definition 140.30. The carrier, classifier, multiplication, unit, and inverse rows are exactly the rows used by Definition 41.4; the checked Lean row is BEDC.Derived.GroupUp.GroupSingletonHistory_laws.

For the manifold side, the chart-domain row is the singleton empty-chart row of Definition 96.32. Its carried predicate is $\mathsf{hsame}(h, \mathsf{e})$, which is definitionally the same predicate as L_{e}.

The two sides therefore share the same history carrier and the same restricted hsame classifier. Projection from the group row gives the algebraic carrier and classifier fields, while projection from the manifold row gives the chart domain, chart value, and singleton atlas fields. □

Theorem 140.32 (LieGroup singleton operation smoothness obligation)**.** *For histories carried by* L_{e}*, both* mul *and* inv *of Definition 140.30 are smooth operations in the singleton atlas. Every accepted multiplication result and every accepted inverse result is classifier-equal to* e*, so each operation is read by the sole chart as a constant transition on the singleton overlap.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton operation smoothness`

Proof. Start with multiplication. If $L_{\mathsf{e}}(h)$ and $L_{\mathsf{e}}(k)$ hold, the group singleton laws identify $\mathsf{append}(h, k)$ with e and keep the result inside the singleton carrier. This is the multiplication-closure row of BEDC.Derived.GroupUp.GroupSingletonHistory_laws.

For inverse, $\mathsf{inv}(h)$ is definitionally e. The inverse-closure row of the same group theorem gives the carrier witness, and the classifier row identifies the result with the unit endpoint.

Now read both operations through the manifold atlas. By Definition 96.32, every carried input and every accepted output lies in the only chart domain and has the constant chart value $0_{\mathsf{Real}^{\uparrow}}$. The overlap transition is therefore the singleton constant transition of Definition 96.33, whose checked Lean row is BEDC.Derived.ManifoldUp.ManifoldSingleton_transition_smoothness. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton operation smoothness obligation`
Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingletonOperation smoothness`

Definition 140.33 (LieGroup singleton public smooth group package)**.** *The public smooth group package underlying the singleton* $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$ *has BEDC source* BHist*, carrier* L_{e}*, classifier* R_{e}*, multiplication graph*

$$\mathsf{mul}(h,k) = \mathsf{append}(h,k),$$

inverse

$$\mathsf{inv}(h) = \mathsf{e},$$

and unit e*. The smooth rows use the singleton chart* χ_{e} *on the same carrier predicate, with constant chart value* $0_{\mathsf{Real}^\uparrow}$ *and self-overlap transition certified by* Cont*. Thus every public algebraic operation is read through the displayed BEDC operation rows rather than through an external standard-space representative.*

Theorem 140.34 (LieGroup singleton classifier transport)**.** *For the package of Definition 140.33, classifier equivalence is congruent for multiplication and inverse. If* $R_{\mathsf{e}}(h,h')$ *and* $R_{\mathsf{e}}(k,k')$*, then*

$$R_{\mathsf{e}}(\mathsf{mul}(h,k), \mathsf{mul}(h',k')).$$

If $R_{\mathsf{e}}(h,h')$*, then*

$$R_{\mathsf{e}}(\mathsf{inv}(h), \mathsf{inv}(h')).$$

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton classifier transport`

Proof. Project the endpoint carrier witnesses from $R_{\mathsf{e}}(h,h')$ and $R_{\mathsf{e}}(k,k')$. Each projection gives membership in L_{e} together with an hsame path to the empty endpoint.

For multiplication, the singleton group laws identify both $\mathsf{append}(h,k)$ and $\mathsf{append}(h',k')$ with e while keeping both results in L_{e}. The classifier row is then the pair of carrier witnesses plus the hsame comparison obtained by transitivity through e.

For inverse, both operation outputs are definitionally e. Reflexivity gives the carrier witnesses and the endpoint comparison, so the restricted classifier R_{e} holds for the two inverse outputs. □

Theorem 140.35 (LieGroup singleton chart operation readback)**.** *The multiplication and inverse outputs covered by Definition 140.34 are read by the singleton chart as the constant endpoint* $0_{\mathsf{Real}^\uparrow}$*. Consequently their chart self-overlap row is the same singleton transition row certified for* $\mathsf{Manifold}^\uparrow$*.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton chart operation readback`

Proof. Apply Definition 140.34 to obtain the classifier witnesses for the operation outputs. In each case the classifier contains the carrier predicate L_{e} for the output history.

Apply Definition 96.32 to those carrier witnesses. The unique chart χ_{e} reads every carried endpoint as the constant value $0_{\mathsf{Real}^\uparrow}$, so multiplication and inverse have the same chart readback.

Finally apply Definition 96.33. Since the only overlap is the singleton self-overlap and both chart values are constant, the operation readback has exactly the displayed transition row. □

Theorem 140.36 (LieGroup singleton public NameCert export)**.** *The public* $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$ *surface for the singleton smooth group exports the following ten fields.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e.

Carrier. *The predicate* $L_{\mathsf{e}}(h)$.

Classifier. *The restricted classifier* $R_{\mathsf{e}}(h,k)$.

Algebraic rows. *Multiplication* append, *inverse* e, *unit* e, *and the singleton* $\mathsf{Group}^{\uparrow}$ *laws.*

Smooth rows. *The singleton* $\mathsf{Manifold}^{\uparrow}$ *chart* χ_{e}, *constant value* $0_{\mathsf{Real}^{\uparrow}}$, *and the singleton transition row.*

Classifier transport. *Multiplication and inverse preserve* R_{e} *as in Definition 140.34.*

Smooth operation congruence. *Operation outputs related by the classifier have the same chart readback and transition row as in Definition 140.35.*

Exactness. *Every accepted public row reduces to* BHist, hsame, Cont, append, e, *the displayed singleton* $\mathsf{Group}^{\uparrow}$ *laws, and the displayed singleton* $\mathsf{Manifold}^{\uparrow}$ *chart and transition.*

Bridge and status. *No* StandardBridge *row is claimed; the paper-side target is under* BEDC.Derived.LieGroupUp.

Proof. The carrier and classifier fields are supplied by Definition 140.30, and the public operation symbols are fixed in Definition 140.33.

The algebraic rows are the singleton $\mathsf{Group}^{\uparrow}$ laws. They give the endpoint carrier facts and identify multiplication and inverse outputs with e.

The smooth rows are the singleton $\mathsf{Manifold}^{\uparrow}$ rows. Chart coverage reads every accepted endpoint as $0_{\mathsf{Real}^{\uparrow}}$, and transition smoothness gives the constant self-overlap row.

The transport and smooth-operation congruence rows are exactly Definition 140.34 and Definition 140.35. Repacking these rows gives the ten-field public certificate surface, with exactness reduced to the listed singleton BEDC data and with no standard bridge component. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton public namecert export`

Theorem 140.37 (LieGroup singleton multiplication smooth homomorphism)**.** *On the concrete carrier* L_{e} *of Definition 140.30, the multiplication row*

$$\mathsf{mul}(h,k) := \mathsf{append}(h,k)$$

is carried by L_{e} *whenever both inputs are carried, is classifier-respecting in both arguments for* R_{e}, *and is read by the singleton chart as the constant transition on the sole overlap.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton multiplication smooth homomorphism`

Proof. The carrier and classifier boundary is the singleton history certificate of Definition 41.4, with checked row BEDC.Derived.GroupUp.GroupSingletonHistory_laws. Its multiplication closure row sends carried h and k to a carried $\mathsf{append}(h,k)$, and its singleton classifier rows identify the result with the endpoint e.

Classifier respect in the first and second arguments is therefore the same singleton calculation: replacing either input by an R_{e}-equal carried input leaves both histories classifier-equal to e, so the product row is again R_{e}-equal to the same endpoint.

Finally apply Definition 96.32 to place the product in the unique chart domain. The self-overlap transition is the constant transition of Definition 96.33, so the chart readback of multiplication is smooth and constant. □

Theorem 140.38 (LieGroup singleton inverse smooth endomap)**.** *On the concrete carrier L_{e} of Definition 140.30, the inverse row*

$$\mathsf{inv}(h) := \mathsf{e}$$

is carried by L_{e}, is classifier-respecting for R_{e}, is involutive up to R_{e}, and is read in the singleton atlas as a constant smooth endomap.

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton inverse smooth endomap`

Proof. The inverse value is definitionally e. The inverse closure and inverse law rows of BEDC.Derived.GroupUp.Grou cited by Definition 41.4, provide the carried witness and identify the inverse endpoint with the singleton classifier class.

Classifier respect follows because every carried input is already R_{e}-equal to e, and applying inv returns e again. Applying the same row twice again returns e, so the double inverse is R_{e}-equal to the original carried singleton point.

The chart statement is the singleton manifold statement. By Definition 96.32, both the input and inverse value lie in the only chart domain; by Definition 96.33, the readback is the constant smooth self-overlap transition. □

Theorem 140.39 (LieGroup singleton smooth-group NameCert export)**.** *The carrier obligation Definition 140.31, the multiplication homomorphism theorem Definition 140.37, and the inverse endomap theorem Definition 140.38 supply the concrete* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *export over* BHist*. The export contains the carrier L_{e}, classifier R_{e}, smooth multiplication, smooth inverse, ledger rows, exactness condition, and the absence of standard bridge rows.*

Proof. The carrier theorem fixes the shared algebraic and manifold predicate on BHist and supplies the classifier row. Hence every exported row has the same underlying object boundary and no separate carrier transport is needed.

The multiplication theorem supplies the smooth binary operation row: closure in L_{e}, respect for R_{e} in each argument, and constant singleton chart readback. The inverse theorem supplies the smooth unary operation row: closure, classifier respect, double-inverse agreement up to R_{e}, and the same constant chart readback.

The ledger is therefore exactly the displayed singleton ledger consisting of BHist, hsame, Cont, append, e, the singleton $\mathsf{Group}^{\uparrow}$ laws, and the singleton $\mathsf{Manifold}^{\uparrow}$ chart and transition rows. Exactness says that an accepted smooth-group row is one of these rows after unfolding the concrete carrier. No $\mathsf{StandardBridge}$ row is part of this export. □

Definition 140.40 (LieGroup singleton translation and conjugation rows)**.** *For a carried singleton parameter s and carried singleton input x, the action rows attached to Definition 140.30 are*

$$\mathsf{Left}_s(x) := \mathsf{mul}(s, x), \qquad \mathsf{Right}_s(x) := \mathsf{mul}(x, s),$$

$$\mathsf{Conj}_s(x) := \mathsf{mul}(\mathsf{mul}(s, x), \mathsf{inv}(s)).$$

The adjoint readback row is the differential of Conj_s at e, read on the singleton tangent carrier as the constant zero endomap of the singleton $\mathsf{LieAlgebra}^{\uparrow}$ carrier. Thus translation, conjugation, and adjoint readback all use the same carried predicate L_{e} and the same singleton chart endpoint.

Lean4 (definition) `BEDC.Derived.LieGroupUp.LieGroupSingletonTranslationRows`

Theorem 140.41 (LieGroup singleton left-translation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Left}_s(x)$ from Definition 140.40 is carried by L_{e}, is classified by the singleton classifier with endpoint e, and is read in the singleton atlas as the constant smooth transition on the sole overlap.*

Proof. The group row of BEDC.Derived.GroupUp.GroupSingletonHistory_laws gives closure of $\mathsf{mul}(s, x)$ under the singleton carrier whenever both inputs are carried by L_{e}. This is exactly the left-translation carrier row after unfolding Definition 140.40.

By Definition 140.31, the carried group result is the same object as the singleton manifold point. Therefore the sole chart domain accepts the result and reads it at the constant endpoint $0_{\mathsf{Real}^{\uparrow}}$.

The overlap transition is the singleton constant transition supplied by Definition 96.33. Hence the left translation is smooth as a singleton-atlas self-map. □

Theorem 140.42 (LieGroup singleton right-translation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Right}_s(x)$ from Definition 140.40 is carried by L_{e}, is classified by the singleton classifier with endpoint e, and is read in the singleton atlas as the constant smooth transition on the sole overlap.*

Proof. The multiplication-closure row of BEDC.Derived.GroupUp.GroupSingletonHistory_laws applies symmetrically to $\mathsf{mul}(x, s)$. Since $L_{\mathsf{e}}(x)$ and $L_{\mathsf{e}}(s)$ hold, the right-translation value remains inside the singleton carrier.

The carrier identification in Definition 140.31 then moves this accepted group value into the singleton chart domain without changing the predicate being used. The classifier endpoint is still e.

Finally, Definition 96.33 reads the only self-overlap as a constant smooth transition. Thus the right translation is a smooth singleton-atlas self-map. □

Theorem 140.43 (LieGroup singleton conjugation smoothness)**.** *If $L_{\mathsf{e}}(s)$ and $L_{\mathsf{e}}(x)$ hold, then $\mathsf{Conj}_s(x)$ from Definition 140.40 is carried by L_{e}, classified with endpoint e, and read by the singleton atlas as a constant smooth self-map.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation smooth`

Proof. First apply Definition 140.41 to s and x. The result $\mathsf{mul}(s, x)$ is carried by L_{e} and is read in the singleton chart as the constant endpoint.

Next use the inverse-closure row from BEDC.Derived.GroupUp.GroupSingletonHistory_laws on the carried parameter s. Since $\mathsf{inv}(s)$ is definitionally e, it is also accepted by the singleton carrier and classifier.

Apply the right-translation row Definition 140.42 to the carried value $\mathsf{mul}(s, x)$ and the carried inverse $\mathsf{inv}(s)$. The resulting history is precisely $\mathsf{Conj}_s(x)$ after unfolding the definition, so it is carried, classified with e, and read as the singleton constant smooth self-map. □

Theorem 140.44 (LieGroup singleton adjoint readback)**.** *For every carried singleton parameter s, the differential at e of the singleton conjugation row Conj_s is the constant zero endomap on the singleton tangent carrier, equivalently on the singleton $\mathsf{LieAlgebra}^{\uparrow}$ carrier.*

Proof. By Definition 140.43, conjugation by any carried singleton parameter is read in the unique chart as a constant self-map at endpoint $0_{\mathsf{Real}^{\uparrow}}$. A constant singleton chart map has the zero tangent readback on the sole tangent point.

The $\mathsf{LieAlgebra}^{\uparrow}$ singleton rows identify the Lie-algebra carrier with the same singleton tangent carrier at e. Therefore the conjugation differential is the constant zero endomap on that carrier, which is exactly the adjoint readback row. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton adjoint readback`

Theorem 140.45 (LieGroup singleton shared source scope)**.** *For the carrier of Definition 140.30, every displayed* $\mathsf{LieGroup}^{\uparrow}$ *row is scoped by the same BEDC history source. The group carrier, manifold chart domain, restricted classifier, unit, multiplication, inverse, chart value, and transition rows are all given by* BHist*,* hsame*,* e*,* append*,* Cont*, the singleton* $\mathsf{Group}^{\uparrow}$ *laws, and the singleton* $\mathsf{Manifold}^{\uparrow}$ *rows. No row requires equality of host-language functions; the only equality test exposed to the certificate is the restricted* hsame *classifier on carried histories.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton shared source scope`

Proof. Unfold the carrier definition. The algebraic carrier is the predicate $L_{\mathsf{e}}(h)$, and the chart domain is the same predicate rather than a second copy. The classifier row is therefore the displayed relation R_{e}, which mentions only L_{e} and hsame.

The group operation rows are the singleton rows: the unit is e, multiplication is append, and inverse is the constant map to e. Their closure and law witnesses are exactly the singleton $\mathsf{Group}^{\uparrow}$ laws.

The manifold rows use the singleton chart, constant value $0_{\mathsf{Real}^{\uparrow}}$, and the self-overlap transition controlled by Cont. Since both sides name the same BHist source, every interface projection is scoped by BHist and by the restricted classifier, with no appeal to host function equality. □

Theorem 140.46 (LieGroup singleton smooth operation readback)**.** *For histories carried by* L_{e}*, the multiplication output* $\mathsf{append}(h, k)$ *and the inverse output* e *are classifier-equal to* e*. Reading either output through the sole singleton chart gives the constant value* $0_{\mathsf{Real}^{\uparrow}}$*, and every operation overlap is the constant self-transition of the singleton atlas.*

Proof. Apply Definition 41.4. Its singleton history laws give closure of $\mathsf{append}(h, k)$, closure of e as the inverse output, and classifier equality of both accepted outputs with the unit endpoint.

Apply Definition 96.32. Every accepted input and accepted output lies in the only chart domain, and the chart value on that domain is the constant $0_{\mathsf{Real}^{\uparrow}}$.

Apply Definition 96.33. The only overlap available in the singleton atlas is the self-overlap, so the operation readback through the chart is the constant transition row already supplied by the manifold certificate. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton smooth operation readback`

Theorem 140.47 (LieGroup singleton dependent interface package)**.** *Any downstream* ExpMap*,* AdjointRep*, or* RootSystem *consumer of the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package may project only the displayed unit, multiplication, inverse, chart, transition, and ledger rows. These projections are stable under the restricted* hsame *classifier: replacing an accepted history by a classifier-equal accepted history preserves the projected carrier witness, operation witness, chart-domain witness, chart value, and transition witness.*

Proof. The allowed projections are exactly the rows named in Definition 140.30. They expose the unit, the append multiplication, the constant inverse, the singleton chart, the constant transition, and the ledger rows BHist, hsame, e, append, and Cont.

Stability follows from the shared source scope of Definition 140.45. Every projected row is defined over L_{e} and R_{e}, so classifier-equal accepted histories remain in the same carried predicate and have the same singleton chart value.

The smooth readback theorem supplies the operation-facing part of the interface. Multiplication and inverse outputs return to the singleton endpoint, and the only chart transition available to the consumer is the constant self-transition. □

Theorem 140.48 (LieGroup singleton downstream unblock package)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *certificate consists of the following ten-field roadmap package.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e.

Carrier. *The predicate* $L_{\mathsf{e}}(h)$*, equivalently* $\mathsf{hsame}(h, \mathsf{e})$.

Classifier. *The restricted classifier* R_{e} *on accepted histories.*

Stability obligations. *Carrier reuse, operation closure, chart-domain reuse, chart-value constancy, transition constancy, and classifier transport under* hsame.

Ledger. BHist, hsame, e, append, Cont, *the singleton* $\mathsf{Group}^{\uparrow}$ *laws, and the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart and transition rows.*

Core theorem list. *Definition 140.31, Definition 140.32, Definition 140.45, Definition 140.46, Definition 140.47, and Definition 140.49.*

Exactness predicate. *A projected Lie-group row is accepted exactly when it factors through the displayed carrier, classifier, operation, chart, and transition rows.*

Standard bridge absent. *No* $\mathsf{StandardBridge}$ *row is part of this package.*

Status. *A paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *target for* $\mathsf{BEDC.Derived.LieGroupUp}$.

Proof. The traditional target, BEDC source, carrier, and classifier fields are the surface rows of Definition 140.30 and Definition 140.31. They identify the algebraic and manifold carriers with the same singleton history predicate.

The stability and ledger fields are supplied by Definition 140.45 and Definition 140.46. These theorems name the shared BHist source, the restricted hsame classifier, the append operation, the constant inverse, and the constant chart transition.

The dependent-interface field is supplied by Definition 140.47. It records the rows that later ExpMap, $\mathsf{AdjointRep}$, and $\mathsf{RootSystem}$ packages may project, and it states their stability under the restricted classifier.

The exactness, bridge, and status fields are bookkeeping consequences of those rows: a row belongs to the singleton package only when it factors through the displayed carrier and operation data, no standard bridge is included, and the target namespace is $\mathsf{BEDC.Derived.LieGroupUp}$. □

Theorem 140.49 (LieGroup singleton NameCert obligation surface)**.** *The rows of Definition 140.30, Definition 140.33, Definition 140.31, Definition 140.32, Definition 140.34, Definition 140.35, Definition 140.36, Definition 140.37, Definition 140.38, Definition 140.39, Definition 140.40, Definition 140.41, Definition 140.42, Definition 140.43, Definition 140.44, Definition 140.45, Definition 140.46, and Definition 140.47 assemble the paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *obligation surface for the singleton smooth group carrier.*

Traditional target. *The one-point Lie group.*

BEDC source. BHist *with distinguished empty history* e.

Carrier. *The predicate* $L_{\mathsf{e}}(h)$*, equivalently* $\mathsf{hsame}(h, \mathsf{e})$.

Classifier. *The restricted singleton classifier* R_{e}.

Stability. *Carrier transport, group operation closure, classifier transport for multiplication and inverse, chart-domain transport, chart-value constancy, smooth multiplication congruence, smooth inverse congruence, transition-row constancy, left and right translation smoothness, conjugation smoothness, and adjoint readback constancy.*

Ledger. *The rows* BHist, hsame, Cont, append, e, *the singleton* $\mathsf{Group}^{\uparrow}$ *laws, the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart, the singleton constant transition, translation rows, conjugation rows, and the zero singleton adjoint readback.*

Core theorem list. *Definition 41.4, Definition 96.37, Definition 140.31, Definition 140.32, Definition 140.34, Definition 140.35, Definition 140.36, Definition 140.37, Definition 140.38, Definition 140.39, Definition 140.41, Definition 140.42, Definition 140.43, Definition 140.44, Definition 140.45, Definition 140.46, and Definition 140.47.*

Exactness. *A Lie-group operation row is accepted only when its carrier, classifier, chart-domain, and transition witnesses all reduce to the displayed singleton rows.*

No standard bridge. *The surface contains no* StandardBridge *row.*

Status. *A paper-side* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *obligation target under* BEDC.Derived.LieGroupUp.

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton semantic name certificate`

Proof. The carrier theorem supplies the shared object boundary: one predicate L_{e} is both the group carrier and the manifold point carrier. This prevents any separate transport problem between algebraic and smooth rows.

The group rows are inherited from Definition 41.4, backed by BEDC.Derived.GroupUp.GroupSingletonHistory_law They give inhabitance, multiplication closure, inverse closure, unit laws, and inverse laws for the append operation with constant inverse.

The manifold rows are inherited from Definition 96.37, backed by BEDC.Derived.ManifoldUp.ManifoldSingleton_s and the singleton transition smoothness row. They give the sole chart domain, constant chart value, and constant self-overlap transition.

The operation-smoothness, classifier-transport, multiplication, inverse, public export, and smooth-group export theorems tie these projections together. Once a group operation result is accepted, it is again classifier-equal to e, hence it is read by the only chart as the same constant endpoint. The translation, conjugation, and adjoint rows are the same singleton readback applied to the displayed public operation rows. Repacking the listed fields gives the obligation surface in the statement, with no standard bridge component claimed. □

Theorem 140.50 (LieGroup singleton standard bridge)**.** *The singleton BEDC* $\mathsf{LieGroup}^{\uparrow}$ *packet of Definition 140.33 interprets as the standard one-point Lie group. The bridge sends the carrier* L_{e} *to the singleton carrier, reads multiplication and inverse from the singleton* $\mathsf{Group}^{\uparrow}$ *rows, and reads smoothness from the singleton* $\mathsf{Manifold}^{\uparrow}$ *chart rows. It carries no non-singleton Lie theory, exponential theory, representation theory, root-system theory, or host quotient data.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton standard bridge`

Proof. The carrier bridge is the singleton carrier of Definition 140.31. Every accepted history is R_{e}-classified with e, so the target carrier has one point.

The algebraic operation bridge is supplied by Definition 140.37 and Definition 140.38. Multiplication and inverse stay in L_{e} and are classifier-equal to the singleton endpoint.

The smooth bridge is supplied by Definition 140.35. The only chart is the singleton chart, and every operation readback is the constant smooth self-overlap row. Repacking these rows gives the standard one-point Lie group interpretation and no further Lie-group structure. □

Theorem 140.51 (LieGroup singleton source lock)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *source uses the single predicate* $L_{\mathsf{e}}(h) :\Longleftrightarrow \mathsf{hsame}(h, \mathsf{e})$ *as the* $\mathsf{Group}^{\uparrow}$ *carrier, the* $\mathsf{Manifold}^{\uparrow}$ *point and chart-domain carrier, and the* $\mathsf{LieGroup}^{\uparrow}$ *carrier. Its classifier is the same restricted* hsame *row* $R_{\mathsf{e}}(h, k)$ *from Definition 140.30; no transported second source, auxiliary carrier, or duplicate classifier is introduced.*

Proof. Unfold Definition 140.30. The displayed predicate L_{e} is the only carrier predicate named there, and the classifier is the displayed restricted row R_{e}.

Project the singleton group rows from Definition 41.4. These rows use BHist, e, append, constant inverse, and the restricted hsame classifier; their checked source is BEDC.Derived.GroupUp.GroupSingletonHistory_lav

Project the singleton chart rows from Definition 96.37. The chart domain is the same predicate $\mathsf{hsame}(h, \mathsf{e})$, the chart value is constant, and the overlap row is the singleton transition row.

Thus both projected packages identify their visible carrier with L_{e}. The $\mathsf{LieGroup}^{\uparrow}$ package is assembled by sharing that predicate and classifier rather than by transporting data across a second source. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton source lock`

Theorem 140.52 (LieGroup manifold and group anchor)**.** *The singleton* $\mathsf{LieGroup}^{\uparrow}$ *source is accepted only after the* $\mathsf{Group}^{\uparrow}$ *operation rows of chapter 41 and the* $\mathsf{Manifold}^{\uparrow}$ *chart rows of chapter 96 have been read. Its public source ledger projects the shared predicate* L_{e} *through those two dependency surfaces and exports no independent ambient smooth-group carrier.*

Proof. Begin with Definition 140.51. The carrier predicate L_{e} and the restricted classifier R_{e} are the single source used by the group carrier, manifold point and chart-domain carrier, and Lie-group carrier.

Project the $\mathsf{Group}^{\uparrow}$ rows through chapter 41. The operation, unit, inverse, and law rows are the singleton history rows named by the source-lock theorem, so the Lie-group packet cannot be accepted before those algebraic rows are visible.

Project the $\mathsf{Manifold}^{\uparrow}$ rows through chapter 96. The chart domain, chart value, and transition row are the singleton chart rows named by the same source lock, so the smooth side is read from the displayed manifold surface rather than from a separate smooth carrier.

Therefore the source ledger is a joint read of the displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Manifold}^{\uparrow}$ dependency rows. Since every exported carrier and classifier is the shared L_{e} and R_{e} projection, no independent ambient smooth-group carrier is available to downstream consumers. □

Theorem 140.53 (LieGroup singleton exact ledger)**.** *For the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package, the accepted ledger consists exactly of* BHist, hsame, e, $\mathsf{append}/\mathsf{Cont}$, *the singleton* $\mathsf{Group}^{\uparrow}$ *laws, the singleton chart domain and chart value, and the singleton constant transition row. It excludes any standard bridge, any host smooth-manifold equality, any extra carrier, and any external topology row.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton exact ledger`

Proof. The source-lock theorem fixes the boundary of the ledger: all carrier and classifier rows are projections of the one history predicate L_{e} and the restricted hsame relation.

The algebraic entries are exactly the singleton $\mathsf{Group}^{\uparrow}$ rows. They are supplied by Definition 41.4 and BEDC.Derived.GroupUp.GroupSingletonHistory_laws, namely BHist, hsame, e, append, Cont, and the singleton group laws.

The smooth entries are exactly the singleton $\mathsf{Manifold}^{\uparrow}$ rows. They are supplied by Definition 96.37, BEDC.Derived.ManifoldUp.ManifoldSingleton_semanticNameCert, and BEDC.Derived.ManifoldUp.Manifol the singleton chart domain, constant chart value, and constant transition.

Finally, Definition 140.32 accounts for the operation rows by reading multiplication and inverse through the same singleton atlas. No cited row provides a StandardBridge, host equality of smooth manifolds, an additional carrier predicate, or an external topology, so none belongs to the accepted ledger. □

Theorem 140.54 (LieGroup singleton downstream export)**.** *The packages* $\mathsf{ExpMap}^{\uparrow}$, $\mathsf{AdjointRep}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, *and representation certificates may consume the singleton* $\mathsf{LieGroup}^{\uparrow}$ *package only through the exported carrier, classifier, operation, chart, and transition rows. The singleton package makes no downstream claim of connectedness, completeness, nontrivial Lie algebra, or standard bridge.*

Proof. By Definition 140.51, every visible carrier and classifier row of the singleton $\mathsf{LieGroup}^{\uparrow}$ package is a projection of the shared history predicate L_{e} and the restricted hsame classifier. These are the only source rows a dependent certificate can name.

By Definition 140.53, the remaining visible rows are precisely the group operation rows, singleton chart rows, and singleton transition row. The NameCert surface in Definition 140.49 repacks those rows as the public obligation surface for $\mathsf{LieGroup}^{\uparrow}$.

Therefore any $\mathsf{ExpMap}^{\uparrow}$, $\mathsf{AdjointRep}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, or representation certificate built over this singleton source can project only those exported rows. Since the ledger contains no connectedness, completeness, nontrivial Lie-algebra, or StandardBridge row, no downstream consumer receives such a claim from this package. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton downstream export`

Definition 140.55 (Singleton LieGroup conjugation action)**.** *On the singleton smooth group carrier Definition 140.30, define the* BHist*-valued conjugation action by*

$$\mathsf{Ad}_{\mathsf{e}}(s, x) := \mathsf{append}(\mathsf{append}(s, x), \mathsf{e}).$$

The inputs are carried histories $L_{\mathsf{e}}(s)$ *and* $L_{\mathsf{e}}(x)$*, and the terminal factor* e *is the singleton inverse row of the group carrier.*

Lean4 (definition) `BEDC.Derived.LieGroupUp.LieGroupSingletonConjugationAction`

Theorem 140.56 (LieGroup singleton conjugation carrier and smoothness)**.** *If* $L_{\mathsf{e}}(s)$ *and* $L_{\mathsf{e}}(x)$ *hold, then* $\mathsf{Ad}_{\mathsf{e}}(s, x)$ *is again carried by* L_{e}*. Moreover the singleton manifold chart reads this conjugation endpoint as the same constant smooth transition that reads multiplication and inverse in Definition 140.32.*

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation carrier and smoothness`

Proof. Unfold Definition 140.55. The endpoint is the append-composite $(s * x) * \mathsf{e}$ in the singleton group rows. Since s and x are carried, Definition 41.4 gives closure of $s*x$, and the right multiplication by e stays in the same carrier. The Lean-side row BEDC.Derived.GroupUp.GroupSingletonClassifier_conjugation_
records the same terminal collapse for the singleton classifier.

For smoothness, the carried input pair and the carried output all lie in the sole singleton chart domain. Applying Definition 96.33 reads the source and target charts as the constant value $0_{\mathsf{Real}^{\uparrow}}$, so the displayed action has the constant smooth transition row already used by the multiplication and inverse obligations. The corresponding Lean-side row is BEDC.Derived.ManifoldUp.ManifoldSingleton_transition
□

Theorem 140.57 (LieGroup singleton conjugation action law)**.** *For carried histories $L_{\mathsf{e}}(s)$, $L_{\mathsf{e}}(t)$, and $L_{\mathsf{e}}(x)$, the two iterated endpoints*

$$\mathsf{Ad}_{\mathsf{e}}(s, \mathsf{Ad}_{\mathsf{e}}(t, x))$$

and

$$\mathsf{Ad}_{\mathsf{e}}(\mathsf{append}(s, t), x)$$

are both classified by the singleton classifier R_{e}, hence represent the same carried singleton endpoint.

Proof. First unfold the action on the left. The inner endpoint is $\mathsf{append}(\mathsf{append}(t, x), \mathsf{e})$, and Definition 140.56 carries it back into L_{e}. Applying the same theorem to s and this inner endpoint therefore carries the whole left-hand endpoint.

Next unfold the action on the right. Group singleton closure carries $\mathsf{append}(s, t)$, and another application of Definition 140.56 carries $\mathsf{Ad}_{\mathsf{e}}(\mathsf{append}(s, t), x)$.

It remains to compare the classifier rows. By the singleton group classifier, every carried endpoint is hsame to e. The Cont associativity and readback rows allow the nested append words on both sides to be read as singleton terminal words, and the group singleton closure removes the terminal factor. Thus both displayed endpoints satisfy R_{e} against e, and therefore against each other by the restricted singleton classifier. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton conjugation action law`

Theorem 140.58 (LieGroup singleton AdjointRep source surface)**.** *The rows of Definition 140.55, Definition 140.56, and Definition 140.57 supply the* $\mathsf{LieGroup}^{\uparrow}$*-side source surface consumed by* $\mathsf{AdjointRep}^{\uparrow}$*: the operation* Ad_{e} *is* BHist*-valued, classifier-stable on the singleton carrier, smooth in the singleton atlas, and carries no standard bridge row.*

Proof. The definition provides the concrete BHist operation. The carrier and smoothness theorem supplies the two structural rows needed by a representation source: accepted inputs give an accepted endpoint, and the singleton atlas reads that endpoint through the constant smooth transition.

The action-law theorem supplies the compatibility row. Both iterated conjugation and conjugation by the product word land in the singleton classifier class, so $\mathsf{AdjointRep}^{\uparrow}$ can consume the action without adding a separate carrier transport or chart transition.

No standard bridge row is part of this surface. The package is only the paper-side singleton $\mathsf{LieGroup}^{\uparrow}$ action source over BHist, with the ordinary Lie-group bridge left outside the displayed rows. □

Lean4 ✓ `BEDC.Derived.LieGroupUp.LieGroupSingleton adjointrep source surface`

Theorem 140.59 (LieGroup singleton TopGroup sibling route)**.** *The singleton smooth-group package of* $\mathsf{LieGroup}^{\uparrow}$ *is readable by the* $\mathsf{TopGroup}^{\uparrow}$ *sibling surface through the same singleton* $\mathsf{Group}^{\uparrow}$ *operation rows. The product and inverse rows of Definition 140.32 land on the continuous product and inverse obligation families of chapter 204, as organized by Definition 204.18 and Definition 204.22. This route exports no non-singleton Lie theory, ambient topology beyond the displayed continuous operation rows, or* bridgeChecked *claim.*

Proof. By Definition 140.51, the $\mathsf{LieGroup}^{\uparrow}$ carrier and classifier are the singleton history predicate L_{e} and the restricted hsame row already used by the singleton $\mathsf{Group}^{\uparrow}$ package. Thus the operation side of the package exposes multiplication, inverse, and unit through the same finite group rows rather than through a separate Lie-theoretic carrier.

The smoothness obligation Definition 140.32 reads those two operation graphs through the singleton chart as constant smooth maps. The $\mathsf{TopGroup}^{\uparrow}$ chapter reads exactly the analogous public operation boundary as product-neighbourhood and inverse-neighbourhood continuity rows; its Definition 204.18 repacks carrier compatibility, product transport, and inverse transport, and Definition 204.22 exhausts the finite Cont ledger by those two operation families.

Therefore the sibling route is the common singleton $\mathsf{Group}^{\uparrow}$ operation surface together with the displayed continuity rows. The route does not add a non-singleton Lie group, external topology source, exponential or representation theory, or a verified standard bridge. □

Theorem 140.60 (LieGroupUp manifold and topological-group route)**.** *Every accepted* $\mathsf{LieGroup}^{\uparrow}$ *carrier first exposes the* $\mathsf{Manifold}^{\uparrow}$ *chart and atlas rows of chapter 96, then exposes the* $\mathsf{TopGroup}^{\uparrow}$ *multiplication and inversion continuity rows of chapter 204, before it presents the combined group-manifold* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *surface. Thus a downstream smooth-group consumer reads chart domains, chart transport, group multiplication, inverse, continuity, smoothness, replay, provenance, and local naming through the displayed sibling rows, not through an ambient manifold object or an external topological group.*

Proof. Project the accepted $\mathsf{LieGroup}^{\uparrow}$ packet to its smooth rows. The chart-domain, chart-readback, atlas-transition, and smoothness entries are exactly the $\mathsf{Manifold}^{\uparrow}$ rows already carried by the singleton source surface, so the manifold side is read before any smooth-group export is available.

Project the same packet to its group-operation rows. Multiplication, inverse, unit, and their continuity readbacks are the $\mathsf{TopGroup}^{\uparrow}$-facing rows consumed by the topological-group sibling surface.

The local $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ row packages these two routes together with the shared BHist classifier, componentwise hsame transport, displayed Cont replay, and Pkg provenance. Since the carrier contains no separate ambient manifold object or external topological group coordinate, every smooth-group read factors through the named $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{TopGroup}^{\uparrow}$ dependency rows. □

Theorem 140.61 (LieGroup group and manifold sibling route)**.** *Every accepted* $\mathsf{LieGroup}^{\uparrow}$ *carrier reads the* $\mathsf{Group}^{\uparrow}$ *multiplication, inverse, and unit rows of chapter 41 together with the* $\mathsf{Manifold}^{\uparrow}$ *chart, atlas, and transition rows of chapter 96 before exporting the smooth-group* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *surface. The exported route packages group operations, manifold chart data, smooth multiplication and inverse readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; it does not add a second group carrier, a second manifold carrier, or an ambient smooth object outside the sibling rows.*

In particular, the smooth multiplication and inversion names are downstream labels: the carrier first consumes the $\mathsf{Group}^{\uparrow}$ *multiplication/inverse obligation row and the* $\mathsf{Manifold}^{\uparrow}$ *chart-*

compatibility row, and only then records their smooth compatibility inside the local $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *route.*

Proof. Project the accepted $\mathsf{LieGroup}^{\uparrow}$ carrier to its algebraic rows. The multiplication, inverse, and unit data are read as the $\mathsf{Group}^{\uparrow}$ operation surface before the smoothness rows are exported.

Project the same carrier to its geometric rows. The chart domains, atlas compatibility, and transition data are read as the $\mathsf{Manifold}^{\uparrow}$ surface before any smooth-group NameCert is formed.

The local $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ row then binds these two sibling routes with the displayed smooth multiplication and inverse readback, transport, replay, and provenance rows. Since the carrier has no independent ambient group or manifold coordinate, every exported smooth-group certificate factors through the cited $\mathsf{Group}^{\uparrow}$ and $\mathsf{Manifold}^{\uparrow}$ rows. □

Chapter 141

A Concrete Naming Certificate for $\mathsf{LieAlgebra}^{\uparrow}$

The Lie-algebra interface $\mathsf{LieAlgebra}^{\uparrow}$ packages a vector-space carrier endowed with an antisymmetric bilinear bracket satisfying the Jacobi identity. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47), and is both the tangent-space-at-identity datum of a Lie group and the algebraic object on which root-system certificates are built.

141.1 Carrier definition

141.2 The certificate

Definition 141.1 (Lie-algebra adjoint action)**.** *Let L carry a $\mathsf{LieAlgebra}^{\uparrow}$ certificate. Write C_L for its vector carrier, $\sim_L$ for its carrier classifier, and $[-,-]_L$ for its bilinear bracket. For each carried endpoint $x : C_L$, define the adjoint action determined by x on carried endpoints by*

$$\mathrm{ad}_x^L(t) := [x, t]_L.$$

Bracket closure makes $\mathrm{ad}_x^L(t)$ carried, and bracket classifier congruence makes this a classifier-respecting endomap of the vector carrier. This is the Lie-algebra side of the ad *coordinate named in chapter 143.*

Lean4 (definition) `BEDC.Derived.LieAlgebraUp.LieAlgebraAdjointAction`
Lean4 (definition) `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjointAction`

Theorem 141.2 (Adjoint action derivation from Jacobi)**.** *Let L carry a $\mathsf{LieAlgebra}^{\uparrow}$ certificate as in chapter 141. Write $+_L$, 0_L, and $-_L$ for the additive vector-space operations inherited from chapter 47. If x, y, z are carried by C_L, then*

$$\mathrm{ad}_x^L([y, z]_L) \sim_L [\mathrm{ad}_x^L(y), z]_L +_L [y, \mathrm{ad}_x^L(z)]_L.$$

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint derivation from jacobi`

Proof. All bracketed endpoints below are carried by the bracket-closure field, and all displayed sums and negatives are carried by the vector-space reduct. Set

$$A := [x, [y, z]_L]_L, \qquad B := [[x, y]_L, z]_L, \qquad C := [y, [x, z]_L]_L.$$

The Jacobi field of the $\mathsf{LieAlgebra}^{\uparrow}$ certificate gives

$$(A +_L [y, [z, x]_L]_L) +_L [z, [x, y]_L]_L \sim_L 0_L.$$

Antisymmetry gives $[z, x]_L \sim_L -_L[x, z]_L$ and $[z, [x, y]_L]_L \sim_L -_L B$. By bracket congruence and linearity in the second argument, the first comparison transports through $[y, -]_L$ and gives

$$[y, [z, x]_L]_L \sim_L -_L C.$$

Transporting the Jacobi comparison along these two classifier equalities yields

$$(A +_L (-_L C)) +_L (-_L B) \sim_L 0_L.$$

Using associativity and commutativity of the additive vector-space law, this is the same classifier comparison as

$$A +_L (-_L(B +_L C)) \sim_L 0_L.$$

Add $B +_L C$ to both sides and use additive congruence:

$$(A +_L (-_L(B +_L C))) +_L (B +_L C) \sim_L 0_L +_L (B +_L C).$$

The additive inverse and unit laws reduce the left endpoint to A and the right endpoint to $B +_L C$, all inside the classifier $\sim_L$. Substituting the definitions of A, B, and C, and unfolding Definition 141.1, gives the displayed adjoint derivation law. □

Theorem 141.3 (Inner derivation commutator identity)**.** *Let L carry a $\mathsf{LieAlgebra}^{\uparrow}$ certificate as in chapter 141. Write $+_L$, 0_L, and $-_L$ for the additive vector-space operations inherited from chapter 47, write $\sim_L$ for the carrier classifier, and write $[-, -]_L$ for the Lie bracket. For carried endpoints x, y, z of C_L, read the bracket of adjoint endomaps pointwise as*

$$[\mathrm{ad}^L_x, \mathrm{ad}^L_y](t) := \mathrm{ad}^L_x(\mathrm{ad}^L_y(t)) +_L (-_L \mathrm{ad}^L_y(\mathrm{ad}^L_x(t))).$$

Then $[\mathrm{ad}^L_x, \mathrm{ad}^L_y](z)$ and $\mathrm{ad}^L_{[x,y]_L}(z)$ are carried by C_L, and

$$[\mathrm{ad}^L_x, \mathrm{ad}^L_y](z) \sim_L \mathrm{ad}^L_{[x,y]_L}(z).$$

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint commutator identity`

Proof. Bracket closure carries $[x, y]_L$, $\mathrm{ad}^L_y(z)$, $\mathrm{ad}^L_x(z)$, $\mathrm{ad}^L_x(\mathrm{ad}^L_y(z))$, $\mathrm{ad}^L_y(\mathrm{ad}^L_x(z))$, and $\mathrm{ad}^L_{[x,y]_L}(z)$. The additive closure and negation closure of the vector-space reduct therefore carry the displayed endomap commutator endpoint.

Set

$$P := \mathrm{ad}^L_x(\mathrm{ad}^L_y(z)), \qquad Q := \mathrm{ad}^L_y(\mathrm{ad}^L_x(z)), \qquad R := \mathrm{ad}^L_{[x,y]_L}(z).$$

Unfolding Definition 141.1, the comparison in Definition 141.2 applied to x, y, z gives

$$P \sim_L R +_L Q.$$

Add $-_L Q$ on the right by additive congruence:

$$P +_L (-_L Q) \sim_L (R +_L Q) +_L (-_L Q).$$

Associativity in the additive vector-space law rewrites the right endpoint as $R +_L (Q +_L (-_L Q))$. The inverse law and the additive unit law classify this endpoint with R. By transitivity of $\sim_L$,

$$P +_L (-_L Q) \sim_L R.$$

Substituting the definitions of P, Q, and R, and using the pointwise endomap-commutator reading above, gives the displayed identity. □

Theorem 141.4 (Adjoint action preserves vector addition)**.** *Let L carry a* $\mathsf{LieAlgebra}^{\uparrow}$ *certificate. Write C_L for its vector carrier, $\sim_L$ for its carrier classifier, $+_L$ for vector addition in the vector-space reduct, and $[-,-]_L$ for the Lie bracket. If x, y, z are carried by C_L, then $\mathrm{ad}^L_x(y +_L z)$ and $\mathrm{ad}^L_x(y) +_L \mathrm{ad}^L_x(z)$ are carried by C_L, and*

$$\mathrm{ad}^L_x(y +_L z) \sim_L \mathrm{ad}^L_x(y) +_L \mathrm{ad}^L_x(z).$$

Proof. The additive closure field of the vector-space reduct carries $y +_L z$. The bracket-closure field of the $\mathsf{LieAlgebra}^{\uparrow}$ certificate therefore carries $[x, y +_L z]_L$, and it also carries $[x, y]_L$ and $[x, z]_L$. Applying additive closure once more carries $[x, y]_L +_L [x, z]_L$.

The second-argument additivity field in the bilinear bracket, applied to the carried endpoints x, y, z, gives

$$[x, y +_L z]_L \sim_L [x, y]_L +_L [x, z]_L.$$

Unfolding the adjoint action from Definition 141.1 on the three bracket endpoints identifies this comparison with the displayed adjoint additive-linearity law, and the closure statements above identify both sides as carried endpoints of C_L. □

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint additive linearity`

Theorem 141.5 (Adjoint action preserves scalar multiplication)**.** *Let L carry a* $\mathsf{LieAlgebra}^{\uparrow}$ *certificate. Write S_L for its scalar-field carrier, C_L for its vector carrier, $\sim_L$ for its vector carrier classifier, $\odot_L$ for the scalar action in the vector-space reduct, and $[-,-]_L$ for the Lie bracket. If r is carried by S_L and x, y are carried by C_L, then $\mathrm{ad}^L_x(r \odot_L y)$ and $r \odot_L \mathrm{ad}^L_x(y)$ are carried by C_L, and*

$$\mathrm{ad}^L_x(r \odot_L y) \sim_L r \odot_L \mathrm{ad}^L_x(y).$$

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint scalar linearity`

Proof. Scalar-action closure in the $\mathsf{VecSpace}^{\uparrow}$ reduct carries $r \odot_L y$. Bracket closure in the $\mathsf{LieAlgebra}^{\uparrow}$ certificate therefore carries $[x, r \odot_L y]_L$. The same bracket-closure field carries $[x, y]_L$, and scalar-action closure carries $r \odot_L [x, y]_L$.

The second-argument scalar-linearity field in the bilinear bracket, applied to the carried scalar endpoint r and carried vector endpoints x, y, gives

$$[x, r \odot_L y]_L \sim_L r \odot_L [x, y]_L.$$

Unfolding the adjoint action from Definition 141.1 on both bracket endpoints identifies this comparison with the displayed scalar-linearity law for ad^L_x, and the closure statements above identify both sides as carried endpoints of C_L. □

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint scalar endpoint closure`

Theorem 141.6 (Adjoint action acting-endpoint scalar linearity)**.** *Let L carry a* $\mathsf{LieAlgebra}^{\uparrow}$ *certificate. Write S_L for its scalar-field carrier, C_L for its vector carrier, $\sim_L$ for its vector carrier classifier, $\odot_L$ for the scalar action in the vector-space reduct, and $[-,-]_L$ for the Lie bracket. If r is carried by S_L and x, y are carried by C_L, then $\mathrm{ad}^L_{r \odot_L x}(y)$ and $r \odot_L \mathrm{ad}^L_x(y)$ are carried by C_L, and*

$$\mathrm{ad}^L_{r \odot_L x}(y) \sim_L r \odot_L \mathrm{ad}^L_x(y).$$

Proof. Scalar-action closure in the $\mathsf{VecSpace}^{\uparrow}$ reduct carries $r \odot_L x$. Bracket closure in the $\mathsf{LieAlgebra}^{\uparrow}$ certificate therefore carries $[r \odot_L x, y]_L$. The same bracket-closure field carries $[x, y]_L$, and scalar-action closure carries $r \odot_L [x, y]_L$.

The first-argument scalar-linearity field in the bilinear bracket, applied to the carried scalar endpoint r and carried vector endpoints x, y, gives

$$[r \odot_L x, y]_L \sim_L r \odot_L [x, y]_L.$$

Unfolding the adjoint action from Definition 141.1 on both bracket endpoints identifies this comparison with the displayed scalar-linearity law for the acting endpoint of ad^L, and the closure statements above identify both sides as carried endpoints of C_L. □

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint acting endpoint scalar linearity`

Theorem 141.7 (Lie-algebra adjoint LinearMap endomap package)**.** *Let L carry a $\mathsf{LieAlgebra}^{\uparrow}$ certificate. Write F_L for the scalar field of its vector-space reduct, $F_{L,\mathsf{Ring}}$ for the ring reduct of that scalar field, C_L for the vector carrier, $\sim_L$ for the vector classifier, $+_L$ and 0_L for vector addition and zero, $\odot_L$ for scalar action, and $[-,-]_L$ for the Lie bracket. If x is carried by C_L, define the endomap identifier*

$$\mathsf{adEnd}^L_x(t) := \mathrm{ad}^L_x(t) = [x, t]_L.$$

Then adEnd^L_x carries the $\mathsf{LinMap}^{\uparrow}$ endomap package on the vector carrier. Equivalently, after the vector-space reduct is read through Definition 47.24,

$$\mathsf{LinearMapCert}_{F_{L,\mathsf{Ring}}}(C_L, C_L; \mathsf{adEnd}^L_x)$$

holds.

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint endomap package`

Proof. The $\mathsf{LieAlgebra}^{\uparrow}$ certificate contains a $\mathsf{VecSpace}^{\uparrow}(F_L, C_L)$ reduct. By Definition 47.24, deleting only the inverse-indexed field rows gives a $\mathsf{Module}^{\uparrow}(F_{L,\mathsf{Ring}}, C_L)$ certificate with the same vector carrier, classifier, vector addition, zero, scalar action, and non-inverse ledgers. These are the source and target module carriers used by Definition 48.59.

For output closure, let t be carried by C_L. Bracket closure in the $\mathsf{LieAlgebra}^{\uparrow}$ certificate carries $[x, t]_L$, so $\mathsf{adEnd}^L_x(t)$ is carried by C_L.

For source-classifier descent, assume $t \sim_L u$ between carried vector endpoints. Classifier reflexivity gives $x \sim_L x$, and the bracket classifier-congruence row cited in Definition 141.1 transports these two comparisons to

$$[x, t]_L \sim_L [x, u]_L.$$

Unfolding the endomap identifier gives

$$\mathsf{adEnd}^L_x(t) \sim_L \mathsf{adEnd}^L_x(u).$$

For additivity, Definition 141.4 gives, for carried $y, z : C_L$,

$$\mathsf{adEnd}^L_x(y +_L z) \sim_L \mathsf{adEnd}^L_x(y) +_L \mathsf{adEnd}^L_x(z).$$

For scalar compatibility, Definition 141.5 gives, for each carried scalar $r : F_L$ and carried $y : C_L$,

$$\mathsf{adEnd}^L_x(r \odot_L y) \sim_L r \odot_L \mathsf{adEnd}^L_x(y).$$

The scalar carrier and scalar action are unchanged by the ring reduct, so this is exactly the scalar row required over $F_{L,\mathsf{Ring}}$.

It remains to record the zero row. The vector-space additive unit law gives

$$0_L +_L 0_L \sim_L 0_L.$$

The descent row just proved transports this comparison through adEnd_x^L, and the additive row at $0_L, 0_L$ gives

$$\mathsf{adEnd}_x^L(0_L) \sim_L \mathsf{adEnd}_x^L(0_L) +_L \mathsf{adEnd}_x^L(0_L).$$

By Definition 47.25, the additive vector carrier of C_L is an $\mathsf{AbGroup}^\uparrow$ certificate with the same $+_L$, 0_L, and $\sim_L$. Applying Definition 48.56 to $w = \mathsf{adEnd}_x^L(0_L)$ yields

$$\mathsf{adEnd}_x^L(0_L) \sim_L 0_L.$$

Thus output closure, source-classifier descent, additive preservation, scalar compatibility, zero preservation, and ledger data are all available: the ledger records the identifier adEnd_x^L, the fixed carried input x, the bracket-closure and bracket-congruence witnesses, and the two second-argument bilinearity rows. By Definition 48.59, these rows form $\mathsf{LinearMapCert}_{F_{L,\mathsf{Ring}}}(C_L, C_L; \mathsf{adEnd}_x^L)$. Since Definition 48.47, Definition 48.48, Definition 48.49, Definition 48.50, and Definition 48.52 read the vector-space endomap package as precisely this function-with-law formation over a common scalar source, the same rows carry the $\mathsf{LinMap}^\uparrow$ endomap package on C_L. □

Theorem 141.8 (Adjoint action acting-endpoint additive linearity)**.** *Let L carry a $\mathsf{LieAlgebra}^\uparrow$ certificate. Write C_L for its vector carrier, $\sim_L$ for its vector carrier classifier, $+_L$ for vector addition in the vector-space reduct, and $[-,-]_L$ for the Lie bracket. If x, z, y are carried by C_L, then $\mathrm{ad}^L_{x+_L z}(y)$ and $\mathrm{ad}^L_x(y) +_L \mathrm{ad}^L_z(y)$ are carried by C_L, and*

$$\mathrm{ad}^L_{x+_L z}(y) \sim_L \mathrm{ad}^L_x(y) +_L \mathrm{ad}^L_z(y).$$

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint acting endpoint additive linearity`

Proof. The additive vector law retained in the $\mathsf{VecSpace}^\uparrow$ reduct, read through Definition 47.19 and Definition 46.18, carries $x +_L z$. Bracket closure in the $\mathsf{LieAlgebra}^\uparrow$ certificate therefore carries $[x +_L z, y]_L$. The same bracket-closure row carries $[x, y]_L$ and $[z, y]_L$, and additive vector closure carries $[x, y]_L +_L [z, y]_L$.

The first-input additivity row of the bilinear bracket, applied to the carried endpoints x, z, y, gives

$$[x +_L z, y]_L \sim_L [x, y]_L +_L [z, y]_L.$$

Unfolding the adjoint action from Definition 141.1 at the three bracket endpoints turns this comparison into the displayed acting-endpoint additive-linearity law, and the carrier statements above identify both sides as endpoints carried by C_L. □

Theorem 141.9 (Lie bracket left-zero annihilation)**.** *Let L carry a $\mathsf{LieAlgebra}^\uparrow$ certificate. Write C_L for its vector carrier, $\sim_L$ for its vector classifier, $+_L$ and 0_L for vector addition and zero in the vector-space reduct, and $[-,-]_L$ for the Lie bracket. If x is carried by C_L, then $[0_L, x]_L$ is carried by C_L and*

$$[0_L, x]_L \sim_L 0_L.$$

The right-zero endpoint is carried and classified with zero as well:

$$[x, 0_L]_L \sim_L 0_L.$$

Lean4 ✓ `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonBracket left zero annihilation`

Proof. The $\mathsf{LieAlgebra}^{\uparrow}$ certificate contains a $\mathsf{VecSpace}^{\uparrow}$ reduct on C_L. Its zero-closure row carries 0_L, its additive closure row carries $0_L +_L 0_L$, and its additive unit law gives

$$0_L +_L 0_L \sim_L 0_L.$$

Bracket closure therefore carries $[0_L, x]_L$ and $[x, 0_L]_L$. By Definition 47.25, the additive vector carrier of C_L is an $\mathsf{AbGroup}^{\uparrow}$ certificate with the same $+_L$, 0_L, and $\sim_L$.

Set $a := [0_L, x]_L$. Apply Definition 141.8 with acting endpoints 0_L and 0_L and target endpoint x. After unfolding Definition 141.1, this gives

$$[0_L +_L 0_L, x]_L \sim_L a +_L a.$$

The bracket classifier-congruence row, applied to $0_L +_L 0_L \sim_L 0_L$ and classifier reflexivity for the carried endpoint x, gives

$$[0_L +_L 0_L, x]_L \sim_L a.$$

Classifier symmetry from Definition 9.7 and transitivity from Definition 9.8 compose the last two comparisons into

$$a \sim_L a +_L a.$$

Since a is carried, Definition 48.56 applied in the additive vector carrier yields

$$[0_L, x]_L = a \sim_L 0_L.$$

Set $b := [x, 0_L]_L$. Apply Definition 141.4 with fixed acting endpoint x and target endpoints 0_L and 0_L. Unfolding Definition 141.1 gives

$$[x, 0_L +_L 0_L]_L \sim_L b +_L b.$$

Bracket classifier congruence, using classifier reflexivity for x and the additive unit comparison $0_L +_L 0_L \sim_L 0_L$, gives

$$[x, 0_L +_L 0_L]_L \sim_L b.$$

Again symmetry and transitivity give $b \sim_L b +_L b$, and duplicate cancellation in the same $\mathsf{AbGroup}^{\uparrow}$ reduct gives

$$[x, 0_L]_L = b \sim_L 0_L.$$

□

Closure status for $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$.

Constructive story: A Lie algebra is accepted by first taking the certified $\mathsf{VecSpace}^{\uparrow}$ carrier, then attaching a bilinear antisymmetric bracket whose outputs remain in that same vector carrier and whose Jacobi row can be read by the classifier. Two algebra packets classify together when the vector endpoints are classified by the $\mathsf{VecSpace}^{\uparrow}$ reduct and every bracket, additive, scalar, and adjoint-endomap endpoint transports through that same carrier classifier. The source covers carried vector endpoints with bracket rows, the pattern records closure, bilinearity, antisymmetry, Jacobi, zero-bracket annihilation, and the inner derivation identities, while the ledger keeps the bracket provenance and the $\mathsf{LinMap}^{\uparrow}$ adjoint-endomap package visible to downstream consumers. The carrier is inhabited by the zero vector row, and the five certificate obligations follow because the vector classifier is reflexive, symmetric, transitive, and stable under addition, scalar action, bracket formation, and transported bracket endpoints. A history h lies in the signature gap of a $\mathsf{LieAlgebra}^{\uparrow}$ token p when the source accepts the vector-plus-bracket packet for h, the signature relation pairs it with an answer algebra packet and evidence ledger, and token introduction names that answer as p; unfolded, the vector carrier, bracket endpoint, Jacobi-derived adjoint action, and linear endomap rows match the data encoded by p. The paper bridge exposes these rows to adjoint-representation and root-system consumers, but it does not supply a Lie-group integration bridge or

classification theorem. The chapter registers this bracket-on-vector-carrier construction under the name LieAlgebra$^{\uparrow}$; no ambient algebra object, hidden tangent space, or external Lie-theoretic classification is assumed.

Theory closure: TheoryGate(NameCert$_{\mathsf{LieAlgebra}\uparrow}$, maturePackageClosed)

Scope closed: The mature package surface is bound to bracket closure, the VecSpace$^{\uparrow}$ reduct, the adjoint-endomap LinMap$^{\uparrow}$ package, inner derivation identities, zero-bracket annihilation, and the downstream bridge consumers exposed by the chapter.

Formal verification: FormalStatus(NameCert$_{\mathsf{LieAlgebra}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.LieAlgebraUp.LieAlgebraSingletonAdjoint\_endomap\_package`

Bridge status: paperBridge

Not claimed: It does not claim a LieGroup$^{\uparrow}$ bridge, a RootSystem$^{\uparrow}$ bridge, or downstream Lie-theoretic classification.

Upgrade path: Bridge-checked closure requires a checked standard bridge from the mature Lie-algebra package inventory to its downstream bridge consumers.

Chapter 142

A Concrete Naming Certificate for $\mathsf{ExpMap}^{\uparrow}$

The exponential-map interface $\mathsf{ExpMap}^{\uparrow}$ packages the canonical map $\mathfrak{g} \to G$ from the Lie algebra to the Lie group of a connected Lie-group carrier. It depends on $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140) and $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ (chapter 141).

142.1 Carrier definition

Definition 142.1 (Certified exponential-map interface carrier)**.** *A certified* $\mathsf{ExpMap}^{\uparrow}$ *interface carrier is a tuple*

$$\mathcal{E} = (\mathcal{G}, \mathcal{L}, \tau, \gamma, \phi, \eta_{\mathcal{G}}, \eta_{\mathcal{L}})$$

where $\mathcal{G}$ *is a* $\mathsf{LieGroup}^{\uparrow}$ *source certificate,* $\mathcal{L}$ *is a* $\mathsf{LieAlgebra}^{\uparrow}$ *source certificate,* $\tau : \mathsf{BHist}$ *is the tangent input row,* $\gamma : \mathsf{BHist}$ *is the group endpoint row,* ϕ *is a local-flow witness from* τ *to* γ*, and* $\eta_{\mathcal{G}}$ *and* $\eta_{\mathcal{L}}$ *are the corresponding* NameCert *source witnesses. The interface classifier is the restricted* hsame *relation on the tangent, endpoint, and local-flow rows. The carrier is accepted exactly when these rows are scoped through the two source* NameCert *witnesses and the local-flow witness records the exponential endpoint determined by the tangent history.*

Lean4 (definition) `BEDC.Derived.ExpMapUp.ExpMapCertifiedInterfaceCarrier`

Theorem 142.2 (ExpMap carrier obligation surface)**.** *The* $\mathsf{ExpMap}^{\uparrow}$ *carrier is the* BHist *graph package whose source projection is licensed by the* $\mathsf{LieAlgebra}^{\uparrow}$ *tangent carrier and whose target projection is licensed by the* $\mathsf{LieGroup}^{\uparrow}$ *carrier. A carried row consists of a source tangent row* x*, a target group row* g*, and a displayed finite BEDC flow ledger connecting them through* Cont *and* hsame *rows. The graph row is generated only by that finite ledger; it does not assert equality of host functions, analytic exponentials, or ambient manifold maps.*

Its ten-field fit is: traditional target the exponential graph $\mathfrak{g} \to G$*; BEDC source the paired* $\mathsf{LieAlgebra}^{\uparrow}/\mathsf{LieGroup}^{\uparrow}$ *certificates; carrier the* BHist *graph package; classifier the componentwise source-target classifier with ledger transport; stability the transport fields supplied by the two dependencies together with* hsame *on the ledger; ledger the finite* Cont *flow display; core theorem list this carrier surface, classifier stability, and zero-flow composition; exactness the assertion that every public row is one such graph package; standard bridge none; status proof target for the* $\mathsf{NameCert}_{\mathsf{ExpMap}^{\uparrow}}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapGraphCarrier obligation surface`

↪ variant `BEDC.Derived.ExpMapUp.ExpMapFlowLedger carrier obligation surface`

↪ variant `BEDC.Derived.ExpMapUp.ExpMapCarrier obligation surface`

Proof. Project the $\mathsf{LieAlgebra}^{\uparrow}$ NameCert surface to obtain the admissible tangent-source rows, and project the $\mathsf{LieGroup}^{\uparrow}$ NameCert surface to obtain the admissible target group rows. These two projections give the endpoints of the graph package without mentioning a host-level function.

Unfold the graph package as a source row, a target row, and a finite ledger of Cont steps with hsame endpoint transport. The displayed ledger is then recorded as the hidden data in the per-object certificate template, so every carried $\mathsf{ExpMap}^{\uparrow}$ row is generated by BEDC rows alone. □

Theorem 142.3 (ExpMap classifier stability obligation surface)**.** *Two* $\mathsf{ExpMap}^{\uparrow}$ *graph rows are classified exactly by agreement of their* $\mathsf{LieAlgebra}^{\uparrow}$ *source classifiers, agreement of their* $\mathsf{LieGroup}^{\uparrow}$ *target classifiers, and* hsame *transport of the displayed finite flow ledger. The classifier is reflexive, symmetric, transitive, and carrier-transporting by the corresponding source certificate field, target certificate field, and* hsame *transport field.*

Its ten-field fit is: traditional target graph equality for the exponential map; BEDC source the source classifier, target classifier, and finite ledger classifier; carrier the graph package of Definition 142.2; classifier the three-part relation just displayed; stability componentwise transport for source, target, and ledger rows; ledger the transported finite flow display; core theorem list the carrier surface and this stability surface; exactness public graph classification by the three visible components; standard bridge none; status proof target for classifier stability in $\mathsf{NameCert}_{\mathsf{ExpMap}^{\uparrow}}$.

Proof. Apply the classifier fields from the $\mathsf{LieAlgebra}^{\uparrow}$ and $\mathsf{LieGroup}^{\uparrow}$ dependency certificates componentwise. Reflexivity, symmetry, transitivity, and carrier transport for the source and target projections are inherited from those two fields.

For the ledger component, use hsame reflexivity for the unchanged ledger, hsame symmetry to reverse a displayed transport, and hsame transitivity to concatenate transports. Repacking the three transported components gives exactly the graph-row classifier and its carrier-transport clause. □

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapClassifier stability obligation surface`

Theorem 142.4 (ExpMap zero and flow-composition obligation surface)**.** *The zero-tangent row of the* $\mathsf{LieAlgebra}^{\uparrow}$ *source maps to the identity row of the* $\mathsf{LieGroup}^{\uparrow}$ *target, and composable finite* $\mathsf{ExpMap}^{\uparrow}$ *flow rows compose by* Cont *associativity up to* hsame*. The ledger provenance for the composed row is restricted to the zero row, the two displayed flow rows, and the* Cont *associativity witness; no host flow equation or analytic uniqueness principle is part of the obligation.*

Its ten-field fit is: traditional target zero-to-identity and exponential-flow composition; BEDC source the $\mathsf{LieAlgebra}^{\uparrow}$ *zero row, the* $\mathsf{LieGroup}^{\uparrow}$ *identity row, and finite* Cont *ledgers; carrier the graph package of Definition 142.2; classifier the stability surface of Definition 142.3; stability* hsame *transport for zero, identity, and composed ledgers; ledger the empty ledger and the two composable finite ledgers; core theorem list zero, unit, and associativity rows; exactness every composed public row has precisely that provenance; standard bridge none; status proof target for zero and local-flow compatibility in* $\mathsf{NameCert}_{\mathsf{ExpMap}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapGraphCarrier zero flow composition obligation`

↪ variant `BEDC.Derived.ExpMapUp.ExpMapFlowLedger zero flow composition obligations`

↪ variant `BEDC.Derived.ExpMapUp.ExpMapZeroFlow obligation surface`

Proof. Represent the zero tangent by the empty finite flow ledger. Reading this ledger through cont_left_unit and cont_right_unit identifies its target endpoint with the LieGroup$^{\uparrow}$ identity row up to hsame.

For two composable flow rows, concatenate their finite Cont ledgers. The associativity witness cont_assoc_hsame transports the concatenated display to the carried graph row, and the classifier stability surface transports the source and target endpoints along with it. Thus the composed row has exactly the displayed ledger provenance. □

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapZeroFlow composition obligation`
Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapFlowLedger zero composition obligation`

142.2 The certificate

Theorem 142.5 (Exponential-map carrier source obligations)**.** *Every certified* ExpMap$^{\uparrow}$ *interface carrier in Definition 142.1 projects the* LieGroup$^{\uparrow}$ *source row, the* LieAlgebra$^{\uparrow}$ *source row, the tangent* BHist *row, and the group-endpoint* BHist *row required by the exponential-map certificate.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapCarrier source obligations`

Proof. Unpack the carrier tuple. The first two components are already the source certificates for LieGroup$^{\uparrow}$ and LieAlgebra$^{\uparrow}$, and the last two source witnesses are the named NameCert scopes for those components.

The tangent and endpoint rows are the third and fourth components of the same tuple. Since the carrier definition accepts the tuple only when these histories are scoped through the source witnesses, projecting them gives exactly the four required rows: group source, algebra source, tangent input, and group endpoint. □

Theorem 142.6 (Exponential-map classifier stability obligations)**.** *For the certified* ExpMap$^{\uparrow}$ *carrier, the restricted* hsame *classifier on tangent, endpoint, and local-flow rows has reflexivity, symmetry, transitivity, and carrier transport obligations supplied by* hsame *and by the two source* NameCert *certificates.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapClassifier stability obligations`
Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapGraphCarrier hsame transport`
↪ variant `BEDC.Derived.ExpMapUp.ExpMapGraphCarrier classifier transport`

Proof. Reflexivity, symmetry, and transitivity are the corresponding structural clauses of hsame restricted to the rows named in Definition 142.1. The restriction does not change the witnesses; it only records that their endpoints are tangent, endpoint, or local-flow rows of the same carrier.

Carrier transport uses the two source certificates. Transport across the LieAlgebra$^{\uparrow}$ source keeps tangent histories in the accepted algebra row, and transport across the LieGroup$^{\uparrow}$ source keeps endpoint histories in the accepted group row. The local-flow component is transported by applying these two transports to its source and target histories and retaining the same hsame comparison as the interface classifier witness. □

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapGraphClassifier stability obligations`

Theorem 142.7 (Exponential-map flow and zero obligations)**.** *The certified* ExpMap$^{\uparrow}$ *carrier has two flow obligations: the zero tangent history is classified with the identity group endpoint, and* Cont*-composition of local-flow histories is compatible with the exponential endpoint classifier.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapFlowAndZero obligations`

Proof. The zero tangent obligation is the local-flow row whose input is the zero row of the $\mathsf{LieAlgebra}^{\uparrow}$ source certificate. Its endpoint is the identity row of the $\mathsf{LieGroup}^{\uparrow}$ source certificate, and the accepted local-flow witness records the hsame comparison between this endpoint and the carried group endpoint.

For composition, take two compatible local-flow witnesses whose middle endpoint rows are hsame-classified. The Cont row composes the two histories, the $\mathsf{LieGroup}^{\uparrow}$ source certificate transports the composed endpoint through the middle comparison, and the $\mathsf{LieAlgebra}^{\uparrow}$ source certificate transports the tangent input row along the algebra classifier. The resulting composed local-flow witness has the same source and target rows required by the exponential endpoint classifier. □

Theorem 142.8 (Exponential-map ledger and dependency obligations)**.** *The certified* $\mathsf{ExpMap}^{\uparrow}$ *obligation ledger contains exactly the* $\mathsf{LieGroup}^{\uparrow}$ *source, the* $\mathsf{LieAlgebra}^{\uparrow}$ *source, the tangent* BHist *row, the endpoint* BHist *row, the restricted* hsame *classifier rows, the* Cont *local-flow composition rows, and the two source* NameCert *witnesses.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapLedger dependency obligations`
Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapFlowLedger dependency exactness`

Proof. Soundness is by reading each accepted ledger row from the carrier tuple or from the source certificate named in the tuple. No further row is needed to state the exponential-map obligations: the carrier projections give the sources and histories, hsame gives the classifier witnesses, Cont gives composition of local-flow histories, and the two NameCert witnesses scope all source transport.

Exactness is the reverse reading. Any row accepted by the $\mathsf{ExpMap}^{\uparrow}$ ledger must be one of the rows listed in the definition of the certified carrier or one of the structural rows named by its source certificates. Therefore the dependency surface is exactly $\mathsf{LieGroup}^{\uparrow}$, $\mathsf{LieAlgebra}^{\uparrow}$, BHist, hsame, Cont, and NameCert, with no additional bridge or Lean verification target claimed here. □

Theorem 142.9 (ExpMap standard-flow bridge surface)**.** *The public* $\mathsf{ExpMap}^{\uparrow}$ *graph package over* BHist *has a standard-flow bridge whose source rows are exactly the* $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ *row, the* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *row, the tangent row, the endpoint row, the finite* Cont *flow ledger, and the componentwise* hsame *classifier. Its exported standard reading is the finite graph of the exponential-flow interface, not a host analytic exponential map.*

Lean4 ✓ `BEDC.Derived.ExpMapUp.ExpMapUp StdBridge`

Proof. Start with Definition 142.2. It identifies the carrier as a BHist graph package with a $\mathsf{LieAlgebra}^{\uparrow}$ tangent source, a $\mathsf{LieGroup}^{\uparrow}$ endpoint target, and a finite BEDC flow ledger.

Use Definition 142.3 to transport the source row, target row, and ledger componentwise. This gives exactly the hsame classifier row required by the standard-flow reading.

Next apply Definition 142.4. The zero and composition rows show that the displayed graph behaves as a finite flow interface under Cont composition, with no appeal to host uniqueness of analytic flows.

Finally project the dependency rows through Definition 142.8. The ledger names precisely $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, the tangent row, the endpoint row, finite Cont, and componentwise hsame. Repacking these projections gives the standard-flow bridge surface, and its standard reading is only the finite graph carried by the same tuple. □

Closure status for $\mathsf{NameCert}_{\mathsf{ExpMap}^\uparrow}$**.**

Constructive story: An exponential-map row is generated as a graph packet over the two dependency certificates: a $\mathsf{LieAlgebra}^\uparrow$ tangent-source history, a $\mathsf{LieGroup}^\uparrow$ target history, and a finite local-flow ledger connecting them through Cont and hsame. A packet is accepted only when both endpoints are licensed by those source certificates and the flow witness is part of the displayed history data; two packets classify together componentwise by the algebra source classifier, the group target classifier, and transport of the finite flow ledger. The source covers exactly these tangent–group graph packages, the pattern records zero tangent to identity, composable flow rows, and graph stability, and the ledger keeps the two dependency certificates, tangent row, endpoint row, finite continuation route, and componentwise history-sameness visible. Nonemptiness is supplied by the zero-flow graph, while reflexivity, symmetry, transitivity, and carrier closure are inherited from the two source classifiers together with hsame transport and Cont associativity of displayed flow rows. A history h lies in the signature gap of an $\mathsf{ExpMap}^\uparrow$ token p when the source accepts a graph packet for h, the signature relation pairs it with an answer graph and evidence ledger, and token introduction names that answer as p; unfolded, the tangent input, group endpoint, and finite flow route are exactly the graph encoded by p. The paper bridge reads the package as the standard finite-flow graph surface from $\mathfrak{g}$ to G, but only through the same dependency, source, target, and ledger rows. The chapter registers this finite-flow graph construction under the name $\mathsf{ExpMap}^\uparrow$; analytic exponential uniqueness and host-level smooth map equality are not assumed.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ExpMap}^\uparrow}$, maturePackageClosed)

Scope closed: The bridged paper boundary is exactly the carrier graph package, componentwise classifier, zero/flow composition row, ledger/dependency surface, and standard-flow bridge witness over $\mathsf{LieGroup}^\uparrow$, $\mathsf{LieAlgebra}^\uparrow$, BHist, hsame, Cont, and NameCert in Definition 142.2, Definition 142.3, Definition 142.4, Definition 142.8, and Definition 142.9.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ExpMap}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.ExpMapUp.ExpMapLedger\_dependency\_obligations`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim analytic exponential uniqueness or host-level smooth map equality for $\mathsf{ExpMap}^\uparrow$.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 143

A Concrete Naming Certificate for $\mathsf{AdjointRep}^{\uparrow}$

The adjoint-representation interface $\mathsf{AdjointRep}^{\uparrow}$ packages the conjugation action $\mathrm{Ad}\colon G \to \mathrm{Aut}(\mathfrak{g})$ together with its differential ad. It depends on $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140) and $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ (chapter 141).

143.1 The certificate

Theorem 143.1 (AdjointRep dependency source obligation)**.** *The dependency source for* $\mathsf{AdjointRep}^{\uparrow}$ *is exactly the displayed* $\mathsf{LieGroup}^{\uparrow}$ *surface of chapter 140 together with the displayed* $\mathsf{LieAlgebra}^{\uparrow}$ *surface of chapter 141. Every* $\mathsf{AdjointRep}^{\uparrow}$ *row is read from these two* NameCert *sources: the group source supplies the carrier, classifier, identity, multiplication, inverse, and conjugation surface, while the algebra source supplies the carrier, classifier, bracket, and adjoint endomap surface.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepDependencySource obligation`

Proof. Unpack the $\mathsf{AdjointRep}^{\uparrow}$ interface as a dependent certificate over the two cited public surfaces. No third ambient carrier is named in the chapter, and no host equality is used to identify the two dependency rows.

The BEDC package discipline therefore fixes the source boundary before the action rows are interpreted. The group side contributes only $\mathsf{LieGroup}^{\uparrow}$-classified histories, and the algebra side contributes only $\mathsf{LieAlgebra}^{\uparrow}$-classified histories.

Thus every subsequent Ad, ad, transport, stability, and target row has these two sources as its complete input surface. □

Theorem 143.2 (AdjointRep conjugation carrier obligation)**.** *The* Ad *row is carried by the* $\mathsf{LieGroup}^{\uparrow}$ *singleton conjugation carrier. For each accepted group history and accepted algebra history, the conjugation endpoint remains a* BHist *action endpoint rather than a host function outside the BEDC ledger.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepConjugation carrier obligation`

Proof. Start with the $\mathsf{LieGroup}^{\uparrow}$ source surface fixed by Definition 143.1. Its singleton carrier already contains the conjugation source row and the classifier that licenses group histories.

Feed an accepted algebra history from the $\mathsf{LieAlgebra}^{\uparrow}$ source into that row. The result is recorded as a BEDC history endpoint, so the visible action does not leave the BHist carrier.

The obligation is exactly this carrier check: Ad must be displayed as an action endpoint in the ledger, with all representative choices hidden behind hsame transport. □

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepConjugationCarrier obligation`

Theorem 143.3 (AdjointRep differential action obligation)**.** *The* ad *row is the* LieAlgebra$^{\uparrow}$ *adjoint endomap package generated by the bracket. Its input and output endpoints are classified by the* LieAlgebra$^{\uparrow}$ *surface, and the row is not a separate derivative object over an unlicensed tangent carrier.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepDifferentialAction obligation`

Proof. Use the LieAlgebra$^{\uparrow}$ dependency named in Definition 143.1. That source already exposes the bracket classifier and the adjoint endomap package used by the algebra carrier.

Reading the differential at the identity group endpoint can therefore only select the displayed adjoint endomap row. It does not create a second tangent carrier, because the source package has no public row for one.

The certificate obligation is to keep ad attached to this bracket endomap package and to record its endpoints through the same BHist and hsame classifier discipline as the algebra source. □

Theorem 143.4 (AdjointRep automorphism target obligation)**.** *The target row for* Ad *is an automorphism-style carrier over the* LieAlgebra$^{\uparrow}$ *classifier. Its inverse and composition data are read from the* LieGroup$^{\uparrow}$ *inverse and multiplication rows together with the* LieAlgebra$^{\uparrow}$ *endomap rows.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepAutomorphism target obligation`

Proof. Apply the conjugation carrier obligation to obtain the visible Ad endpoint. Apply the group inverse and multiplication rows from LieGroup$^{\uparrow}$ to read inverse and composite group histories inside the same classifier boundary.

Next use the LieAlgebra$^{\uparrow}$ endomap package. It supplies the target classifier over which composition of algebra endpoints is interpreted.

The automorphism-style row is then a repacking of these two source surfaces: group inverse gives the inverse action endpoint, group multiplication gives composition of action endpoints, and algebra endomaps keep the target over the Lie-algebra classifier. □

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepAutomorphismTarget obligation`

Theorem 143.5 (AdjointRep classifier stability obligation)**.** *Transported* LieGroup$^{\uparrow}$ *and* LieAlgebra$^{\uparrow}$ *inputs preserve the displayed* Ad *and* ad *endpoints through* hsame*,* Cont*, and NameCert classifier transport. Consequently the action row, differential row, and automorphism-target row are stable under the two source classifiers.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepSingleton classifier stability obligation`
↪ variant `BEDC.Derived.AdjointRepUp.AdjointRepClassifier stability obligation`

Proof. First transport the group input along the LieGroup$^{\uparrow}$ classifier and hsame. The conjugation carrier obligation reads the resulting Ad endpoint from the same singleton action row, so the endpoint is unchanged up to the displayed BEDC classifier.

Next transport the algebra input along the LieAlgebra$^{\uparrow}$ classifier, hsame, and the continuation ledger Cont. The differential action obligation keeps ad inside the bracket endomap package, and the automorphism target obligation keeps Ad inside the algebra endomap target.

These two transports are the whole stability check. No row is stabilized by host equality; every preserved endpoint is read through the two NameCert classifiers and the BEDC continuation relation. □

Closure status for $\mathsf{NameCert}_{\mathsf{AdjointRep}^\uparrow}$.

Constructive story: An adjoint-representation row is generated only after the $\mathsf{LieGroup}^\uparrow$ and $\mathsf{LieAlgebra}^\uparrow$ sources are fixed: the group side supplies carrier, multiplication, inverse, and conjugation rows, while the algebra side supplies the bracket carrier, classifier, and adjoint endomap package. A displayed Ad endpoint is accepted when conjugation sends a carried group history and carried algebra history to a BHist action endpoint, and the ad endpoint is accepted when it is the bracket-generated adjoint endomap already present in the Lie-algebra package. Two accepted rows classify together by transporting the group input, algebra input, action endpoint, differential endpoint, and automorphism target through the two source classifiers, hsame, and the visible Cont ledger. The source covers exactly the two dependency surfaces, the pattern records conjugation carrier, differential action, automorphism target, and classifier stability, and the ledger keeps the group, algebra, action, differential, inverse, composition, and source-provenance rows aligned. The five naming-certificate obligations are inherited from the two dependencies plus the stability theorem: accepted source rows inhabit the carrier, classifier reflexivity, symmetry, and transitivity are componentwise, and transported endpoints remain within the displayed action and endomap packages. A history h lies in the signature gap of an $\mathsf{AdjointRep}^\uparrow$ token p when the source accepts its dependency/action packet, the signature relation pairs it with an answer representation row and evidence ledger, and token introduction names that answer as p; unfolded, h carries the same Ad, ad, automorphism-target, and classifier-stability rows encoded by p. The chapter registers this dependency-driven conjugation-and-differential construction under the name $\mathsf{AdjointRep}^\uparrow$; Lie integration uniqueness and any closure beyond the displayed adjoint rows are not assumed.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AdjointRep}^\uparrow}$, maturePackageClosed)

Scope closed: The public exported surface is the checked dependency, conjugation, differential, automorphism-target, and classifier-stability package over $\mathsf{LieGroup}^\uparrow$, $\mathsf{LieAlgebra}^\uparrow$, BHist, hsame, Cont, and the two source NameCert classifiers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AdjointRep}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.AdjointRepUp.AdjointRepSingleton\_classifier\_stability\_obligation`

Bridge status: paperBridge

Not claimed: It does not claim Lie integration uniqueness, bridgeChecked verification, or closure outside the displayed public adjoint-representation rows.

Upgrade path: There is no further paper-side closure upgrade beyond standard-bridge verification.

Theorem 143.6 (AdjointRep multiplicative action row)**.** *Let g and h be carried* $\mathsf{LieGroup}^\uparrow$ *histories in the source surface of* $\mathsf{AdjointRep}^\uparrow$*, and let t be a carried* $\mathsf{LieAlgebra}^\uparrow$ *history. Write gh for the product history read by the* $\mathsf{LieGroup}^\uparrow$ *multiplication row, write $\mathrm{Ad}_g(t)$ for the carried adjoint-action endpoint supplied by the* $\mathsf{AdjointRep}^\uparrow$ *conjugation row, and write $\sim_{\mathfrak{g}}$ for the target* $\mathsf{LieAlgebra}^\uparrow$ *endomap classifier. Then the product action endpoint and the iterated action endpoint are both carried by the displayed automorphism-target row, and*

$$\mathrm{Ad}_{gh}(t) \sim_{\mathfrak{g}} \mathrm{Ad}_g(\mathrm{Ad}_h(t)).$$

This comparison is a pointwise endomap-classifier row over the $\mathsf{LieAlgebra}^\uparrow$ *target, not an equality of host maps and not an additional group-action bridge.*

Lean4 ✓ `BEDC.Derived.AdjointRepUp.AdjointRepMultiplicativeAction row`

Proof. By Definition 143.1, the only group-side rows available to $\mathsf{AdjointRep}^\uparrow$ are the $\mathsf{LieGroup}^\uparrow$ carrier, classifier, identity, multiplication, inverse, and conjugation rows, and the only algebra-side target rows are the $\mathsf{LieAlgebra}^\uparrow$ carrier, classifier, bracket, and adjoint endomap surface. Thus gh is the product history supplied by the group multiplication row inside the same source boundary.

Apply Definition 143.2 first to h and t. The endpoint $\mathrm{Ad}_h(t)$ is a displayed BHist action endpoint over the algebra input row. Applying the same carrier obligation to g and this endpoint carries the iterated endpoint $\mathrm{Ad}_g(\mathrm{Ad}_h(t))$. Applying it to gh and t carries the product endpoint $\mathrm{Ad}_{gh}(t)$.

The comparison row comes from the $\mathsf{LieGroup}^{\uparrow}$ source, not from an external action. The concrete singleton source surface of Definition 140.58 exports the conjugation operation and its compatibility row; the compatibility itself is Definition 140.57, which classifies conjugation by the product word with the corresponding iterated conjugation endpoint. Reading that row with the action input supplied by the $\mathsf{AdjointRep}^{\uparrow}$ carrier gives the BEDC endpoint comparison between $\mathrm{Ad}_{gh}(t)$ and $\mathrm{Ad}_g(\mathrm{Ad}_h(t))$.

It remains only to identify the target classifier. By Definition 143.4, the Ad target is an automorphism-style carrier over the $\mathsf{LieAlgebra}^{\uparrow}$ classifier: group multiplication supplies composition of action endpoints, while the algebra endomap rows keep the result over the algebra target classifier. Therefore the endpoint comparison just obtained is read pointwise as

$$\mathrm{Ad}_{gh}(t) \sim_{\mathfrak{g}} \mathrm{Ad}_g(\mathrm{Ad}_h(t)).$$

Finally, Definition 143.5 records that all transport used in this reading is through hsame, Cont, and the two source $\mathsf{NameCert}$ classifiers. Hence the multiplicative action row is internal to the displayed $\mathsf{AdjointRep}^{\uparrow}$ ledger and invokes no host group action. □

Chapter 144

A Concrete Naming Certificate for $\mathsf{RootSystem}^{\uparrow}$

The root-system interface $\mathsf{RootSystem}^{\uparrow}$ packages a finite set of nonzero vectors in a finite-dimensional inner-product space closed under the reflections they generate, with integer Cartan pairings. It depends on $\mathsf{NameCert}_{\mathsf{InnerProduct}^{\uparrow}}$ (chapter 86), $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112), and $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$ (chapter 31), and is the combinatorial backbone of semisimple Lie-algebra classification.

144.1 Carrier definition

Definition 144.1 (RootSystem BHist certificate surface)**.** *A* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface is a displayed package with the following ten fields:*

1. *the traditional target is a finite root system: a finite set of nonzero vectors in an inner-product vector carrier, closed under the displayed root reflections, with integer Cartan pairings;*
2. *the BEDC source is the product of the visible* $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, *and* $\mathsf{Int}^{\uparrow}$ *certificate surfaces;*
3. *the vector carrier* $V : \mathsf{BHist} \to \mathsf{Prop}$ *and vector classifier* $\sim_V$ *are inherited from the* $\mathsf{InnerProduct}^{\uparrow}$ *source;*
4. *the root carrier is a finite root-row predicate* $R : \mathsf{BHist} \to \mathsf{Prop}$ *over* V*, presented by a* $\mathsf{FinSet}^{\uparrow}$ *enumeration whose entries are carried by* V*;*
5. *the public root classifier combines the inner-product vector classifier* $\sim_V$ *with the finite-set membership classifier, so two root rows match only when their displayed vector representatives and finite-support rows match;*
6. *each root row records a nonzero witness, expressed as failure of the vector-zero classification under* $\sim_V$*;*
7. *for carried roots* α *and* β*, the reflection row displays a carried vector* $s_\alpha(\beta)$ *together with a* Cont *route witnessing that the row is computed from the inner-product and vector-space operations;*
8. *for carried roots* α *and* β*, the Cartan endpoint row displays an* $\mathsf{Int}^{\uparrow}$*-carried history* $c_{\alpha,\beta}$ *for the pairing* $2\langle\beta,\alpha\rangle/\langle\alpha,\alpha\rangle$*;*
9. *the stability rows transport nonzero witnesses, reflection closure, Cartan pairing endpoints, and ledger descent through the root classifier and* hsame*;*

10. *the ledger records finite support, nonzero witnesses, reflection routes, and integer Cartan endpoints, with every row descending to the displayed sources named in the second field.*

The surface therefore names a finite root-row certificate inside BEDC histories. It does not name an ambient Lie algebra, a Weyl group, or a classification theorem.

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemTasteGate single carrier alignment`

144.2 The certificate

Theorem 144.2 (RootSystem nonzero finite carrier obligation)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface, the public root carrier exposes only the* $\mathsf{FinSet}^{\uparrow}$ *finite row* R *of nonzero* $\mathsf{InnerProduct}^{\uparrow}$ *vectors. Every displayed root history is an entry of the finite-support row, is carried by the inner-product vector carrier* V*, and has a recorded witness that it is not classified as the vector zero by* $\sim_V$*.*

Proof. Project the finite root-row predicate from Definition 144.1. The fourth field gives the $\mathsf{FinSet}^{\uparrow}$ enumeration, and its entries are explicitly carried by the vector carrier inherited from $\mathsf{InnerProduct}^{\uparrow}$.

Project the nonzero row from the sixth field. It is attached to each displayed entry of the finite-support row and is stated through the vector classifier $\sim_V$. Thus a public root has no access path except through the finite enumeration and the nonzero witness recorded beside that entry. □

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemFiniteSupportCarrier nonzero rows`
Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemFiniteSupportCarrier project rows`

Theorem 144.3 (RootSystem carrier classifier obligation)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface, root-carrier transport is exactly transport along the inherited vector classifier and the finite-set membership classifier. If two root rows have* hsame *finite-support entries and their vector representatives are classified by* $\sim_V$*, then all visible carrier data of the two root rows agree by the public root classifier.*

Proof. Unpack the classifier field of Definition 144.1. It has two visible components: the $\mathsf{InnerProduct}^{\uparrow}$ vector classifier on representatives and the $\mathsf{FinSet}^{\uparrow}$ classifier on the finite membership presentation.

Use the hsame witness on the finite-support entries to identify the displayed rows selected from the enumeration. Use the vector classifier witness to transport the inherited vector-carrier facts and the attached nonzero witness. Combining these two projections is precisely the public root classifier described by the fifth field. □

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemFiniteSupportCarrier classifier transport`

Theorem 144.4 (RootSystem reflection closure obligation)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface and carried roots* α *and* β*, each displayed reflection row sends* β *to a carried root* $s_\alpha(\beta)$*. The closure witness uses the recorded* Cont *route for the reflected vector and the inherited inner-product vector classifier transport; no external reflection action is added.*

Proof. Read the reflection row from the seventh field of Definition 144.1. It supplies both the reflected vector history $s_\alpha(\beta)$ and the Cont route expressing how that history is built from the displayed inner-product and vector-space operations.

Apply the vector-carrier and classifier transport rows inherited from $\mathsf{InnerProduct}^{\uparrow}$ along that route. The reflection-closure component of the surface then identifies the reflected vector with an

entry of the finite root row. Since membership is classified by the public root classifier, the result is a carried root row rather than a host-level vector equality. □

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemReflectionClosure result unary`

Theorem 144.5 (RootSystem Cartan ledger obligation)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface and carried roots* α *and* β*, each Cartan endpoint* $c_{\alpha,\beta}$ *is an* $\mathsf{Int}^{\uparrow}$*-carried history. If the input root rows are transported by the public root classifier, the corresponding Cartan endpoints transport through the* $\mathsf{Int}^{\uparrow}$ *classifier and the ledger records the same integer endpoint row.*

Proof. Project the Cartan endpoint row from the eighth field of Definition 144.1. Its codomain is the integer-history carrier supplied by $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$, so the endpoint is not merely a scalar quotient in the inner-product source.

Transport the two input roots through the public root classifier. The stability field carries the numerator, denominator, and displayed integer endpoint through the inherited vector and scalar classifiers and then through the $\mathsf{Int}^{\uparrow}$ classifier. The ledger component records this transported endpoint beside the same pair of root rows, which gives stable integer Cartan data. □

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemCartanLedger cont transport`

Theorem 144.6 (RootSystem NameCert obligation surface)**.** *The nonzero finite carrier obligation of Definition 144.2, the carrier classifier obligation of Definition 144.3, the reflection closure obligation of Definition 144.4, and the Cartan ledger obligation of Definition 144.5 assemble the paper-side* $\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}$ *obligation surface. This surface does not claim Weyl-group closure, Lie-algebra classification, or standard bridge closure.*

Proof. Start from the certificate surface of Definition 144.1. The finite root-row carrier and its nonzero witnesses are supplied by Definition 144.2.

The public classifier and its hsame transport behavior are supplied by Definition 144.3. The reflection rows and their closure under carried roots are supplied by Definition 144.4.

Finally, the integer Cartan endpoints and their ledger stability are supplied by Definition 144.5. These four projections cover carrier, classifier, stability, and ledger data for the naming certificate. The listed exclusions are separate downstream interfaces, so they are not packaged into this $\mathsf{RootSystem}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemFiniteSupport semanticNameCert`

Theorem 144.7 (RootSystem mature bridge consumption)**.** *The* $\mathsf{RootSystem}^{\uparrow}$ *finite-support carrier, public classifier transport, reflection closure, and Cartan ledger rows of Definition 144.6 are exactly the* $\mathsf{RootSystem}^{\uparrow}$*-side data consumed by the* $\mathsf{WeylGroup}^{\uparrow}$ *terminal standard bridge of Definition 145.11 and Definition 145.12. This gives the chapter's mature paper surface without claiming Lie-algebra classification or crystallographic classification.*

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupRootSystem mature bridge consumption`

Proof. Apply Definition 144.6. Its four displayed rows give the finite root carrier, public root classifier transport, reflection closure, and integer Cartan ledger boundary for the $\mathsf{RootSystem}^{\uparrow}$ source.

The $\mathsf{WeylGroup}^{\uparrow}$ standard bridge proof rows then consume those same rows: Definition 145.6 reads the carrier and classifier, Definition 145.7 reads reflection closure, and Definition 145.8 with Definition 145.9 reads the Cartan and action-stability ledgers.

Finally, Definition 145.11 and Definition 145.12 package exactly those consumed rows as the terminal standard bridge surface. The two excluded classification claims are not fields of those bridge rows, so they remain outside the $\mathsf{RootSystem}^{\uparrow}$ closure claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}$.

Constructive story: A root-system packet is generated from three visible sources: an $\mathsf{InnerProduct}^{\uparrow}$ vector carrier and classifier, a $\mathsf{FinSet}^{\uparrow}$ finite-support enumeration of root rows, and $\mathsf{Int}^{\uparrow}$ endpoints for the Cartan pairings. A history is accepted as a root only when it appears in the finite support, is carried by the inner-product vector carrier, and carries a nonzero witness; two root rows classify together when their finite-support entries and vector representatives are matched by the finite-set and vector classifiers. The source covers these finite nonzero vector rows with reflection and Cartan data, the pattern records reflection closure, integer Cartan endpoints, and nonzero readback, and the ledger keeps finite support, nonzero witnesses, reflection routes, and integer Cartan endpoints descending to the displayed sources. The carrier is inhabited by a displayed finite root row, and reflexivity, symmetry, transitivity, and carrier closure are supplied by the inherited classifiers plus transport of nonzero witnesses, reflected vectors, and Cartan endpoints through hsame. A history h lies in the signature gap of a $\mathsf{RootSystem}^{\uparrow}$ token p when the source accepts the finite-support root packet for h, the signature relation pairs it with an answer root-system packet and evidence ledger, and token introduction names that answer as p; unfolded, the finite root list, vector representatives, reflection closure rows, and integer Cartan ledger match the data encoded by p. The mature bridge consumption reads precisely these rows into the terminal $\mathsf{WeylGroup}^{\uparrow}$ reflection-group bridge, while Lie-algebra and crystallographic classification stay outside the package. The chapter registers this finite-support reflection construction under the name $\mathsf{RootSystem}^{\uparrow}$; no ambient Lie algebra, Weyl-group classification theorem, or hidden host vector equality is introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 144.7 is the mature RootSystem-side bridge-consumption surface for the finite-support carrier, public classifier transport, reflection closure, and Cartan ledger rows consumed by the WeylGroup terminal standard bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.WeylGroupUp.WeylGroupStandardReflectionGroup\_bridge`

Bridge status: paperBridge

Not claimed: The chapter does not close Lie-algebra classification or crystallographic classification.

Upgrade path: No further theory step; bridgeChecked verification requires a RootSystemUp standard-bridge target.

Theorem 144.8 (RootSystem simple-reflection involution row)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface, let* α *and* β *be carried root rows, and let the displayed reflection row along* α *be read through its Cartan-normalized* Cont *route*

$$s_\alpha(\gamma) \sim_V \gamma - c_{\alpha,\gamma} \cdot_V \alpha$$

for every carried input γ*, where* $c_{\alpha,\gamma}$ *is the integer Cartan endpoint supplied by the same surface. Then the iterated displayed row is classified with the original input root:*

$$s_\alpha(s_\alpha(\beta)) \sim_V \beta.$$

The witness uses only the inherited $\mathsf{InnerProduct}^{\uparrow}$ *vector classifier, the displayed vector-space rows, the reflection* Cont *route, and the Cartan integer endpoint ledger.*

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemReflectionClosure involution row`

Proof. By Definition 144.4, the first reflected row $s_\alpha(\beta)$ is a carried root, so the reflection row may be applied to it again. The same theorem keeps both displayed reflection endpoints inside the public root carrier, and Definition 144.3 says that their comparison is the inherited vector classifier $\sim_V$ together with the finite-support classifier.

Write $c = c_{\alpha,\beta}$. The Cartan-normalized reflection route for the first step gives the vector-classifier row

$$s_\alpha(\beta) \sim_V \beta - c \cdot_V \alpha.$$

The Cartan endpoint is the displayed integer history for

$$c = 2\langle \beta, \alpha \rangle / \langle \alpha, \alpha \rangle$$

by Definition 144.5. Apply the scalar projection and form-law rows of the $\mathsf{InnerProduct}^{\uparrow}$ root basis, namely Definition 86.60 and Definition 86.58, to the carried vectors α and $\beta - c \cdot_V \alpha$. Sesquilinearity gives the retained scalar-classifier computation

$$\begin{aligned} 2\langle s_\alpha(\beta), \alpha \rangle &\sim_{\mathsf{scal}} 2\langle \beta - c \cdot_V \alpha, \alpha \rangle \\ &\sim_{\mathsf{scal}} 2\langle \beta, \alpha \rangle - 2c\langle \alpha, \alpha \rangle \\ &\sim_{\mathsf{scal}} -c\langle \alpha, \alpha \rangle. \end{aligned}$$

The last comparison uses the defining Cartan endpoint row for c, transported through the retained scalar classifier. Since α is a root, Definition 144.2 supplies the nonzero witness for α, and the positive-definite exactness in Definition 86.61 prevents the diagonal denominator $\langle \alpha, \alpha \rangle$ from being scalar-zero. Thus the displayed Cartan endpoint for the reflected input is classified as

$$c_{\alpha, s_\alpha(\beta)} \sim_{\mathsf{Int}^{\uparrow}} -c.$$

This is an integer-ledger comparison, not a host scalar equality, because Definition 144.5 keeps Cartan endpoints in the $\mathsf{Int}^{\uparrow}$ carrier and transports them through its classifier.

Now apply the reflection route a second time:

$$s_\alpha(s_\alpha(\beta)) \sim_V s_\alpha(\beta) - c_{\alpha, s_\alpha(\beta)} \cdot_V \alpha.$$

Substitute the first reflection row and the Cartan comparison just obtained, using vector addition, scalar multiplication, additive inverse, and classifier congruence from the inherited $\mathsf{VecSpace}^{\uparrow}$ rows exposed by Definition 86.57. The right endpoint is classified as

$$\begin{aligned} (\beta - c \cdot_V \alpha) - (-c) \cdot_V \alpha &\sim_V \beta + ((-c) + c) \cdot_V \alpha \\ &\sim_V \beta + 0 \cdot_V \alpha \\ &\sim_V \beta. \end{aligned}$$

Composing these vector-classifier witnesses by the public root classifier gives $s_\alpha(s_\alpha(\beta)) \sim_V \beta$. The proof has used the two displayed reflection Cont routes, the Cartan endpoint ledger, the inner-product form laws, and the inherited vector-space classifier congruence; no ambient Weyl group, host-level equality of vectors, or quotient of reflection words is introduced. □

Theorem 144.9 (RootSystem simple reflection negates its axis)**.** *For a* $\mathsf{RootSystem}^{\uparrow}$ *BHist certificate surface, let* α *be a carried root row. Let the displayed reflection row along* α *be read through its Cartan-normalized* Cont *route*

$$s_\alpha(\gamma) \sim_V \gamma - c_{\alpha,\gamma} \cdot_V \alpha$$

for every carried input γ*, where* $c_{\alpha,\gamma}$ *is the integer Cartan endpoint supplied by the same surface. Then the reflection of* α *through its own displayed root axis is classified with the vector-space additive inverse of* α*:*

$$s_\alpha(\alpha) \sim_V -_V \alpha.$$

The witness is internal to the $\mathsf{RootSystem}^{\uparrow}$ *surface: it uses the inherited* $\mathsf{InnerProduct}^{\uparrow}$ *vector classifier, the displayed vector-space additive rows, the reflection* Cont *route, and the Cartan endpoint ledger.*

Lean4 ✓ `BEDC.Derived.RootSystemUp.RootSystemReflectionClosure axis negation`

Proof. By Definition 144.2, the carried root row α is a carried vector endpoint in the inherited $\mathsf{InnerProduct}^{\uparrow}$ vector carrier and carries a witness that α is not classified with the vector zero 0_V. Therefore the vector-space reduct carries $-_V\alpha$ and all displayed endpoints formed from α by addition and scalar action.

Read the reflection row from the seventh field of Definition 144.1 and specialize its displayed route to the carried input $\gamma = \alpha$. This gives

$$s_\alpha(\alpha) \sim_V \alpha - c_{\alpha,\alpha} \cdot_V \alpha.$$

It remains to identify the diagonal Cartan endpoint $c_{\alpha,\alpha}$ with the carried integer 2 and then simplify inside the vector-space reduct.

The eighth field of Definition 144.1, read through Definition 144.5, says that $c_{\alpha,\alpha}$ is the $\mathsf{Int}^{\uparrow}$ endpoint for

$$2\langle\alpha,\alpha\rangle/\langle\alpha,\alpha\rangle.$$

The root form-law surface of Definition 86.58 places the diagonal endpoint $\langle\alpha,\alpha\rangle$ in the retained scalar source. Since α is not vector-zero, the exact diagonal zero test of Definition 86.61 prevents $\langle\alpha,\alpha\rangle$ from being scalar-zero. Hence the scalar field cancellation row applies to the displayed nonzero diagonal denominator and classifies

$$2\langle\alpha,\alpha\rangle/\langle\alpha,\alpha\rangle \sim_{\mathsf{scal}} 2.$$

Transporting this scalar comparison through the integer endpoint ledger, by the Cartan endpoint transport component of Definition 144.5, gives the Cartan-classifier comparison

$$c_{\alpha,\alpha} \sim_{\mathsf{Int}^{\uparrow}} 2.$$

This is a comparison of displayed Cartan endpoint histories, not a host-level scalar equality.

Use the inherited $\mathsf{VecSpace}^{\uparrow}$ rows exposed by Definition 86.20. The scalar-action congruence field listed in Definition 46.18, inherited by Definition 47.19, transports the Cartan comparison to

$$c_{\alpha,\alpha} \cdot_V \alpha \sim_V 2 \cdot_V \alpha.$$

The scalar-addition distributivity and unit-action rows listed in Definition 46.18, inherited by Definition 47.19, give the vector-space doubling comparison between $2 \cdot_V \alpha$ and $\alpha +_V \alpha$, and subtraction is addition by the additive inverse. Thus

$$\begin{aligned}
\alpha - c_{\alpha,\alpha} \cdot_V \alpha &\sim_V \alpha - 2 \cdot_V \alpha \\
&\sim_V \alpha +_V (-_V(\alpha +_V \alpha)) \\
&\sim_V \alpha +_V ((-_V\alpha) +_V (-_V\alpha)) \\
&\sim_V (\alpha +_V (-_V\alpha)) +_V (-_V\alpha) \\
&\sim_V 0_V +_V (-_V\alpha) \\
&\sim_V -_V\alpha.
\end{aligned}$$

The displayed steps use only additive inverse congruence, associativity, commutativity where needed to expose the inverse pair, the additive inverse law, and the additive unit law in the inherited vector-space reduct.

Composing the specialized reflection route with this vector-space simplification by transitivity of the public vector classifier gives

$$s_\alpha(\alpha) \sim_V -_V \alpha.$$

By Definition 144.4, the left endpoint is also a carried root row, while the comparison itself remains the inherited vector classifier comparison required by the statement. □

Chapter 145

A Concrete Naming Certificate for $\mathsf{WeylGroup}^{\uparrow}$

The Weyl-group interface $\mathsf{WeylGroup}^{\uparrow}$ packages the finite reflection group generated by the simple roots of a root system. It depends on $\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}$ (chapter 144) and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41), and is the discrete symmetry datum on which character-theoretic certificates rest.

145.1 Carrier definition

Definition 145.1 (WeylGroup BHist source row)**.** *The paper-side* $\mathsf{WeylGroup}^{\uparrow}$ *source row is the product of the* $\mathsf{RootSystem}^{\uparrow}$ *finite-support carrier of Definition 144.1 and the singleton* $\mathsf{Group}^{\uparrow}$ *history carrier of Definition 41.3. A Weyl word is a* BHist *endpoint generated by a finite list of simple-reflection rows from the root source, with word composition read by* Cont *and append in the group source. Two Weyl words are compared by the restricted* hsame *classifier that preserves the carried root endpoint, the singleton* $\mathsf{Group}^{\uparrow}$ *word endpoint, and the finite root-row ledger.*

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupTasteGate single carrier alignment`

145.2 The certificate

Theorem 145.2 (WeylGroup RootSystem source obligation)**.** *A* $\mathsf{WeylGroup}^{\uparrow}$ *certificate source is exactly a scoped* $\mathsf{RootSystem}^{\uparrow}$ *finite-support certificate together with the* $\mathsf{Group}^{\uparrow}$ *carrier and classifier rows used to compose reflected-root histories. The source exposes the finite root rows, the reflection-closure rows, the Cartan ledger rows, and the singleton composition scope; it does not add a separate host-level equality relation on roots or actions.*

Proof. Project the $\mathsf{RootSystem}^{\uparrow}$ finite-support certificate. Its public rows give the finite root carrier, reflection closure for displayed simple reflections, and the Cartan integer endpoints used to read relation data.

Project the $\mathsf{Group}^{\uparrow}$ source beside it. The carrier and classifier rows provide the singleton composition scope in which reflected-root histories are composed. Combining these projections gives exactly the source surface named by the statement, with all equality routed through the displayed BEDC classifiers. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupRootSystem source obligation`

Theorem 145.3 (WeylGroup simple reflection carrier obligation)**.** *Each carried simple root row exports a simple-reflection generator history whose endpoint is a* BHist *group-carried action on root rows. The generator is carried by the* $\mathsf{Group}^{\uparrow}$ *composition surface, and its action on roots is transported by the* $\mathsf{RootSystem}^{\uparrow}$ *reflection-closure row.*

Proof. Start with a carried simple root row from the finite root support of $\mathsf{RootSystem}^{\uparrow}$. The reflection-closure row supplies the reflected root endpoint for every carried input root, so the simple reflection is a displayed operation on root rows rather than an external map.

Transport that displayed operation through the $\mathsf{Group}^{\uparrow}$ carrier. The singleton composition scope packages the reflected-root endpoint as a BHist action history, and the group classifier records that this history is carried by the certified group surface. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupSimpleReflection carrier obligation`

Theorem 145.4 (WeylGroup Coxeter ledger obligation)**.** *The* $\mathsf{WeylGroup}^{\uparrow}$ *ledger records finite generator words, Cartan integer endpoints, and Coxeter relation rows without quotienting by host equality. Relation witnesses are packaged as* BHist *rows and descend through* hsame *classifier transport.*

Proof. Read finite support from the $\mathsf{RootSystem}^{\uparrow}$ source. Since the simple roots are displayed as finite rows, the generator words are finite lists of carried simple-reflection histories and have no need for an ambient free group outside the certificate.

Read the Cartan endpoints from the same source and attach the Coxeter relation rows to the corresponding finite words. The relation witnesses are then transported through hsame and the public classifiers, so the ledger records classified rows rather than quotient classes imposed by the host theory. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupCoxeterWordLedger relation transport`

Theorem 145.5 (WeylGroup action classifier stability obligation)**.** *If two generator words are related by the Coxeter ledger classifier, then their induced actions on carried roots are related by the* $\mathsf{RootSystem}^{\uparrow}$ *public classifier. The stability is computed inside the displayed word ledger using simple-reflection carrier transport, the* $\mathsf{Group}^{\uparrow}$ *associativity and unit rows, and* $\mathsf{RootSystem}^{\uparrow}$ *classifier transport.*

Proof. Induct over the displayed finite-word ledger. In the empty word case, the $\mathsf{Group}^{\uparrow}$ unit row identifies the induced action with the carried root row, and the $\mathsf{RootSystem}^{\uparrow}$ classifier witnesses stability of that root.

In the generator step, use the simple-reflection carrier obligation to transport the head generator across the carried root row. The induction hypothesis transports the remaining word, and the $\mathsf{Group}^{\uparrow}$ associativity row rebrackets the composed action without leaving the displayed carrier.

For a Coxeter relation row, read the relation witness from the ledger and transport it through hsame. The resulting endpoints are compared by the $\mathsf{RootSystem}^{\uparrow}$ public classifier, which gives the required stability of the two induced root actions. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupActionClassifier stability obligation`

Theorem 145.6 (WeylGroup root-system group carrier row)**.** *The source row of Definition 145.1 extracts the* $\mathsf{WeylGroup}^{\uparrow}$ *carrier and classifier from exactly two displayed surfaces: the finite nonzero* $\mathsf{RootSystem}^{\uparrow}$ *carrier with its public root classifier, and the singleton* $\mathsf{Group}^{\uparrow}$ *BHist carrier with its append operation, empty unit, constant inverse, and classifier* R_{e}*. A carried Weyl endpoint is*

therefore a root-system source row together with a singleton group word row, and its classifier is the restricted hsame *comparison on those two rows.*

Proof. Project the root side from Definition 144.2 and Definition 144.3. These rows supply the finite root support, the inherited vector classifier, and the finite-set membership classifier that can be transported through hsame.

Project the word side from Definition 41.4 and Definition 41.5. They supply the singleton history carrier, the append operation, the unit row, inverse closure, and the group classifier over BHist.

The Weyl carrier is the conjunction of these two source reads. The restricted classifier is the NameCert classifier transport applied simultaneously to the root row and the group word row, so no host-level equality of root-system presentations or group words is introduced. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupRootSystemGroupCarrier row`

Theorem 145.7 (WeylGroup simple-reflection word closure row)**.** *For every carried root row α in the* $\mathsf{RootSystem}^{\uparrow}$ *source, the simple-reflection generator s_α determines a carried Weyl word endpoint. If two carried Weyl words are represented by endpoints w_0 and w_1, and* $\mathsf{Cont}(w_0, w_1, w_{01})$ *records their append product, then w_{01} is again a carried Weyl word acting on carried roots by the displayed reflection closure rows of* $\mathsf{RootSystem}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupSourceRow append closure`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupSimpleReflection word closure`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupBHistSourceRow simple reflection word closure`

Proof. Start with the generator row. The reflection closure theorem Definition 144.4 supplies, for carried roots α and β, a carried root $s_\alpha(\beta)$ and the Cont route that computes it from the root-system source.

Read the same generator as a singleton $\mathsf{Group}^{\uparrow}$ word endpoint using Definition 41.4. The singleton group laws give closure of append products, the empty unit row, and classifier transport for the word endpoint.

For a composite word, use continuation associativity from Definition 21.29 and the unit rows cited by the group laws to reassociate the displayed append path. Each reflection step remains over carried roots by the root-system reflection row, and NameCert classifier transport repacks the final endpoint as a carried Weyl word. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupSimpleReflection word closure row`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupSimpleReflectionWord closure row`

Theorem 145.8 (WeylGroup Coxeter word ledger row)**.** *Coxeter relations between simple-reflection words in the Definition 145.1 surface are ledger rows, not host-level equalities of words. For each displayed pair of simple roots α, β, the Cartan endpoint ledger of* $\mathsf{RootSystem}^{\uparrow}$ *records the integer relation data that licenses the corresponding braid row, and the* $\mathsf{Group}^{\uparrow}$ *singleton word source records the two append spines being compared under the restricted* hsame *classifier.*

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupCoxeterWordLedger row`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupCoxeterWordLedger append spine classifier`

Proof. Read the relation parameter from Definition 144.5. The Cartan endpoint is an $\mathsf{Int}^{\uparrow}$-carried history, and its transport is already stated as ledger stability under the public root classifier.

Build the two braid-word spines by iterating the simple-reflection word row from Definition 145.7. The group singleton laws and continuation associativity keep the two append spines inside the singleton $\mathsf{Group}^{\uparrow}$ word carrier while preserving the displayed ledger entries for each generator occurrence.

The comparison is then a recorded classifier row between the two carried spines. It names the Coxeter relation as BEDC provenance data from the Cartan ledger and the two word ledgers; it does not quotient the Weyl-word carrier by an external equality relation. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupCoxeterWordLedger empty relation readback`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupCoxeterLedger append spine rows`

Theorem 145.9 (WeylGroup action classifier stability row)**.** *The Weyl action classifier is stable under* hsame *on both inputs. If root rows are transported by the* $\mathsf{RootSystem}^{\uparrow}$ *public classifier and Weyl words are transported by the restricted* $\mathsf{Group}^{\uparrow}$ *word classifier of Definition 145.1, then the displayed action endpoints on carried roots are transported by the same* $\mathsf{RootSystem}^{\uparrow}$ *classifier, with all generator, append, Coxeter-ledger, and dependency rows preserved.*

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupActionTrace classifier stability row`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupBHistSourceRow action classifier stability`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupActionClassifier stability transport`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupActionClassifier append stability row`
Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupActionTrace public boundary`

Proof. Transport the root input through Definition 144.3. This preserves the finite-support row, the inherited vector representative, and the nonzero witness used by the reflection closure theorem.

Transport the word input through Definition 145.6. The word classifier is the restricted hsame row supplied by the singleton $\mathsf{Group}^{\uparrow}$ source and the NameCert classifier transport, so append products and unit insertions stay within the same carried word scope.

Apply the simple-reflection closure row generator by generator, using Definition 145.7. Reassociate composites by the continuation associativity and unit rows, and read Coxeter comparisons through Definition 145.8. The final root endpoint is therefore classified by the $\mathsf{RootSystem}^{\uparrow}$ public classifier with the same dependency ledger. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupRootSystemGroupCarrier action classifier transport`

Theorem 145.10 (WeylGroup public NameCert export)**.** *The rows of Definition 145.1, Definition 145.6, Definition 145.7, Definition 145.8, and Definition 145.9 package the public paper-side* $\mathsf{NameCert}_{\mathsf{WeylGroup}^{\uparrow}}$ *over the* $\mathsf{RootSystem}^{\uparrow}$ *finite-support classifier and the* $\mathsf{Group}^{\uparrow}$ *singleton word classifier. The export exposes the carried root input, the singleton word input, the action endpoint, the Coxeter ledger rows, and the action-stability row as the public Weyl-group certificate boundary, with no host word equality and no Coxeter quotient.*

Proof. Project the root-system component from Definition 145.1 and Definition 145.6. This supplies the finite-support root carrier, its public classifier, and the endpoint transport row inherited from $\mathsf{RootSystem}^{\uparrow}$.

Project the word component from the same source row and the singleton $\mathsf{Group}^{\uparrow}$ carrier. The simple-reflection word closure theorem supplies the generator action rows, and the Coxeter word ledger theorem records braid data as carried ledger entries rather than as an equality quotient on host words.

Finally apply the action classifier stability row. It transports carried roots and carried words through the public classifiers already projected, so the root-system carrier, group carrier, word closure, Coxeter ledger, and action-stability rows repack exactly as the public $\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$ boundary. No additional equality principle or Coxeter quotient is introduced. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupPublicNameCert export`

Theorem 145.11 (WeylGroup finite reflection-group bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$ *surface of Definition 145.10 carries a paper-side bridge with the standard finite reflection-word presentation. In the forward direction, every BEDC carried Weyl word from Definition 145.1 has a finite reflection-word interpretation obtained by reading the* $\mathsf{RootSystem}^\uparrow$ *finite root rows and composing the displayed simple reflections through the* $\mathsf{Group}^\uparrow$ *singleton word carrier. If two BEDC words are related by the Coxeter ledger rows, then their interpreted reflection words induce the same action on carried roots under the* $\mathsf{RootSystem}^\uparrow$ *public classifier.*

In the reverse direction, every finite word in displayed simple reflections whose letters are $\mathsf{RootSystem}^\uparrow$ *root rows has a BEDC word endpoint. The endpoint is obtained by the simple-reflection word closure row and by iterated* Cont *append through the singleton* $\mathsf{Group}^\uparrow$ *carrier, so the standard word reads back into the same public* $\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$ *boundary.*

Proof. For the forward direction, read the BEDC word through Definition 145.6. This separates the finite $\mathsf{RootSystem}^\uparrow$ root-row carrier from the singleton $\mathsf{Group}^\uparrow$ word carrier while preserving the restricted classifier on carried Weyl endpoints.

Next interpret each displayed generator by Definition 145.7. The resulting finite list of simple reflections is composed inside the singleton word carrier. When the input BEDC words are Coxeter-ledger-related, Definition 145.8 supplies the recorded braid rows, and Definition 145.9 transports the two induced actions to the same $\mathsf{RootSystem}^\uparrow$ public classifier class.

For the reverse direction, start from a finite standard word whose letters are displayed root rows. Apply Definition 145.7 to the first letter, then iterate the append clause using Cont for the remaining letters. The root-system and group carrier rows from Definition 145.6 keep every intermediate endpoint inside the public source surface.

Finally package the interpreted word and the readback endpoint by Definition 145.10. The bridge therefore changes only the paper-side reading between BEDC carried words and finite reflection-words; it does not add a host word quotient or advance the Lean verification axis. □

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupStandardReflectionGroup bridge`

Theorem 145.12 (WeylGroup root-system terminal standard bridge)**.** *The public* $\mathsf{WeylGroup}^\uparrow$ *surface of Definition 145.10 supplies a paper-level* $\mathsf{StandardBridge}(\mathsf{WeylGroup}^\uparrow, T^*_W)$*, where* T^*_W *is the terminal host-facing Weyl-word action on the displayed* $\mathsf{RootSystem}^\uparrow$ *support. The encode field sends the unique host Weyl word to the BEDC empty singleton group word over the carried root row. The decode field sends every carried Weyl word to the unique host word. The decode-encode round trip is terminal equality in* T^*_W*, and the encode-decode round trip is the restricted* $\mathsf{Group}^\uparrow$ *singleton classifier together with the* $\mathsf{RootSystem}^\uparrow$ *public classifier. The theorem-preservation field is the finite list consisting of the root-system group carrier row, simple-reflection word closure, Coxeter word ledger, action classifier stability, and public NameCert export. The no-host-leak field confines the host terminal object to this bridge theorem.*

Proof. We check the six fields of section 9204.1.

1. The encode field is the constant map from the unique word of T_W^* to the empty singleton $\mathsf{Group}^\uparrow$ word carried over a displayed $\mathsf{RootSystem}^\uparrow$ row. The word endpoint is public by Definition 41.194, and the root row is public by Definition 144.6.
2. The decode field sends any carried Weyl word of Definition 145.1 to the unique host Weyl word. Since T_W^* is terminal, classifier-respect on the host side is terminal equality.
3. The decode-encode round trip starts from the unique host word, encodes it as the empty singleton word over the carried root row, and decodes back to the same host word. This is terminal equality in T_W^*.
4. The encode-decode round trip starts from a carried Weyl word. The word component is classified by the restricted singleton $\mathsf{Group}^\uparrow$ classifier as in Definition 41.194; the root component is classified by the public $\mathsf{RootSystem}^\uparrow$ classifier of Definition 144.6. Together these are exactly the restricted classifier of Definition 145.6.
5. The theorem-preservation list is finite and explicit. Its rows are Definition 145.6, Definition 145.7, Definition 145.8, Definition 145.9, and Definition 145.10. The checked action transport row records the same carrier, word, ledger, and classifier dependencies.
6. The no-host-leak field follows because the host terminal object appears only in the encode, decode, and terminal-equality clauses above. The exported surface remains the BEDC $\mathsf{RootSystem}^\uparrow$ support, singleton $\mathsf{Group}^\uparrow$ word carrier, Coxeter ledger rows, action classifier stability row, and public NameCert boundary.

□

Lean4 ✓ `BEDC.Derived.WeylGroupUp.WeylGroupUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$.

Constructive story: A Weyl-group element is carried by combining two already certified surfaces: a finite-support $\mathsf{RootSystem}^\uparrow$ row supplies the simple roots and reflection-closure data, while the singleton $\mathsf{Group}^\uparrow$ carrier supplies the word endpoint and append operation used to compose displayed simple reflections. The carrier accepts exactly finite lists of simple-reflection rows whose actions on carried roots are recorded as BHist endpoints; rows without the root-system support, singleton group word, finite root ledger, or reflection-closure witness are outside the surface. Two Weyl words are classified by the restricted hsame comparison that preserves the carried root endpoint, singleton group word endpoint, Coxeter ledger rows, and action trace, rather than by quotienting host words by Coxeter relations. The source covers the root-system finite support and singleton group word carrier, the pattern records simple-reflection word closure, append closure, Coxeter word ledger rows, action classifier stability, and public export, and the ledger keeps Cartan endpoints, generator occurrences, braid rows, and action readback visible. The five name-certificate obligations are inherited from $\mathsf{RootSystem}^\uparrow$ and $\mathsf{Group}^\uparrow$: their inhabited carriers supply endpoints, their classifiers give reflexive, symmetric, and transitive transport, and simple-reflection closure plus action stability repack transported words as carried Weyl endpoints. A history h lies in the signature gap of a $\mathsf{WeylGroup}^\uparrow$ token p when the source-domain accepts its root support and word row, the signature relation pairs it with an answer word and evidence ledger, and token introduction names the answer as p; unfolded, the finite simple-reflection word, Coxeter ledger, and induced action on carried roots are the data encoded by p. The paper bridge reads carried words as finite reflection-word presentations and reads such words back through the same singleton group append surface, while the terminal bridge confines the host object to a terminal readback; the chapter registers this construction under the name $\mathsf{WeylGroup}^\uparrow$ with no nonterminal Coxeter quotient, host word equality, or full traditional classification introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$, maturePackageClosed)

Scope closed: The paper-side mature surface covers the public $\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$ export, the finite reflection-word interpretation/readback bridge, the root-system terminal standard bridge, and the explicit theorem-preservation list for the BHist source row, RootSystemUp finite-support and classifier rows, GroupUp singleton carrier and word classifier rows, simple-reflection word closure, Coxeter word ledger, and action classifier stability.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WeylGroup}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.WeylGroupUp.WeylGroupRootSystemGroupCarrier\_action\_classifier\_transport`

Bridge status: bridgeChecked

Not claimed: This paper-side maturity claim does not assert bridgeChecked verification, nonterminal Coxeter quotient exactness, non-ledger word equality, or full traditional Weyl-group classification.

Upgrade path: No further upgrade: chapter at top of formal axis.

Chapter 146

A Concrete Naming Certificate for $\mathsf{BilinForm}^{\uparrow}$

The bilinear-form interface $\mathsf{BilinForm}^{\uparrow}$ packages a bilinear pairing on a module carrier together with its symmetry, antisymmetry, and nondegeneracy classifiers. It depends on $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ (chapter 46) and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47), and is the universal datum from which inner products, symplectic forms, and Clifford algebras are licensed.

146.1 Carrier definition

Definition 146.1 (BilinForm module-pairing source row)**.** *The* $\mathsf{BilinForm}^{\uparrow}$ *BEDC source row is the paired source* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}} \times \mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *together with a displayed pairing endpoint row. Its module side supplies the additive carrier, zero row, additive inverse row, and scalar-action rows. Its vector-space side supplies the scalar-field carrier and the vector classifier used to read linear endpoints. The pairing endpoint is a* BHist *row* $\beta(x, y)$ *packaged by* Pkg *with provenance from the two input carrier rows, the scalar endpoint row, a* ProbeBundle *recording the visible input probes, and the ledger entry that names the pairing operation.*

The public BilinForm carrier is therefore not a host function type. It is the visible package of two source histories, one scalar-valued endpoint history, and the BHist *ledger that records how the endpoint was obtained from the certified module and vector-space sources.*

Lean4 (definition) `BEDC.Derived.BilinFormUp.BilinFormModulePairingSourceRow`

Theorem 146.2 (BilinForm pairing classifier row)**.** *For two* $\mathsf{BilinForm}^{\uparrow}$ *pairing rows over the source of Definition 146.1, equality of public pairing classifiers is displayed as* hsame *on the two endpoint histories together with common package provenance. Concretely, the classifier compares the source module row, the source vector-space row, the two input carrier rows, the scalar endpoint row, and the* Pkg *token that records the pairing ledger. No host-level equality of bilinear maps is part of the classifier.*

Proof. Unpack the source row. Each public pairing entry consists of two input histories, one scalar endpoint history, one ProbeBundle of visible input probes, and a package token whose provenance names the module and vector-space sources.

The forward comparison is the displayed hsame comparison on these histories. The common package provenance fixes the source certificates and the pairing ledger, so the two endpoint rows are compared inside the same BEDC surface rather than by an external function equality.

Conversely, given the hsame endpoint comparison and the common package provenance, repack the two source histories, the probe bundle, and the endpoint row into the public classifier. The resulting witness is exactly the classifier equality claimed in the statement. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingSourceRow pairing classifier row`

Theorem 146.3 (BilinForm bilinearity transport row)**.** *For a carried* $\mathsf{BilinForm}^{\uparrow}$ *pairing row, additive transport in each input and scalar-action transport in each input are inherited from the* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *sources. If the module source transports* x_0, x_1, y_0, y_1 *through its additive rows and the vector-space source transports scalar actions through its scalar rows, then the pairing endpoint rows transport through the scalar endpoint classifier by the corresponding* hsame *package route.*

Proof. Project the additive rows from the module source. They give the displayed histories for addition, zero, and inverse on each input row, with carrier transport stated in the module classifier.

Project the scalar-action rows from the vector-space source. These rows state how a scalar history acts on a vector history and how that action transports through the source classifier.

Now apply the pairing ledger to the transported input rows. The ledger names the same pairing operation and the same scalar endpoint package, so the transported endpoint is related to the original endpoint by hsame. Repack that endpoint comparison as the BilinForm pairing classifier of Definition 146.2. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface bilinearity transport row`
Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingPackageRow bilinearity transport row`

Theorem 146.4 (BilinForm dual symmetry row)**.** *The* $\mathsf{BilinForm}^{\uparrow}$ *duality row records the certificate choice between symmetry and antisymmetry as visible* SigRel *data. A symmetric choice displays a* SigRel *row comparing* $\beta(x, y)$ *with* $\beta(y, x)$*, while an antisymmetric choice displays a* SigRel *row comparing* $\beta(y, x)$ *with the additive inverse of* $\beta(x, y)$ *in the scalar endpoint source. The certificate exposes the chosen row and does not silently infer the other choice.*

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingSourceRow dual symmetry row`

Proof. Read the duality field of the BilinForm source as a tagged displayed row. The tag names either the symmetric classifier or the antisymmetric classifier, and the payload is a SigRel witness over the two swapped pairing endpoints.

In the symmetric case, the witness is repacked directly as the comparison between $\beta(x, y)$ and $\beta(y, x)$. All source provenance remains the same, only the order of the two carried input rows changes.

In the antisymmetric case, first project the scalar additive inverse row from the module and vector-space scalar source. The displayed inverse endpoint is then compared with $\beta(y, x)$ by the recorded SigRel witness. Repacking this witness preserves the visible certificate choice and adds no unstated symmetry claim. □

Theorem 146.5 (BilinForm nondegeneracy witness ledger row)**.** *Nondegeneracy for* $\mathsf{BilinForm}^{\uparrow}$ *is witnessed by displayed partner rows and an output-separation package. For every carried input row not classified with the zero row, the ledger records a carried partner row and a scalar endpoint package whose classifier separates the output from the scalar zero row. The claim is a BEDC ledger witness, not host-level extensionality of the pairing.*

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingPackageRow nondegeneracy witness ledger row`
Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface nondegeneracy witness transport`

Proof. Start with a carried input row and its displayed failure to be classified as zero. This is data in the module and vector-space sources, so it can be transported only through the public source classifiers.

The nondegeneracy ledger then supplies a partner history and a scalar endpoint history obtained by the pairing row. It also supplies the output-separation package, which records that this scalar endpoint is not classified with the scalar zero row.

Repack these three witnesses as the public nondegeneracy row: the original input, the partner input, and the separated scalar endpoint. Since every piece is a displayed history or package token, the proof never appeals to extensional equality of maps outside BEDC. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingSourceRow nondegeneracy witness ledger row`

Theorem 146.6 (BilinForm exactness dependency row)**.** *The source row of Definition 146.1, the classifier row of Definition 146.2, the bilinearity transport row of Definition 146.3, the duality row of Definition 146.4, and the nondegeneracy ledger row of Definition 146.5 form the obligation surface needed for the* $\mathsf{BilinForm}^{\uparrow}$ *certificate over* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$*. The exactness boundary says that every public BilinForm row is obtained from these five rows and that every consumer-facing dependency descends to one of the two named source certificates.*

Proof. First collect the carrier data from Definition 146.1. This gives the two source certificates, the input histories, the scalar endpoint history, and the probe bundle with its package token recording the pairing ledger.

Next add classifier and stability. The classifier theorem supplies hsame and common Pkg provenance for endpoint comparison, while the bilinearity transport theorem supplies additive and scalar-action stability inherited from the two source certificates.

Finally, add the visible duality and nondegeneracy rows. The duality theorem records the chosen SigRel row, and the nondegeneracy theorem records partner histories with output separation packages.

These projections exhaust the public BilinForm certificate surface: carrier, classifier, bilinearity, duality, nondegeneracy, ledger, exactness, and source dependencies. Repacking the BHist rows, Pkg tokens, hsame comparisons, $\mathsf{ProbeBundle}$ data, and SigRel witnesses gives the stated obligation surface over $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormModulePairingPackageRow exactness dependency row`

Definition 146.7 (BilinForm root pairing surface)**.** *A* $\mathsf{BilinForm}^{\uparrow}$ *root datum is read from two source certificates, a module certificate* $\mathcal{M} : \mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ *and a vector-space certificate* $\mathcal{V} : \mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$*, together with a scalar endpoint row supplied by the shared scalar source. Its visible pairing surface is a ternary* BHist *relation*

$$\mathsf{Pair}_B(x, y, z)$$

where x *and* y *are carried vector endpoints accepted by the module and vector-space classifiers, and* z *is the carried scalar endpoint produced by the pairing row.*

The carrier-aware classifier on pairing outputs declares two displayed rows $\mathsf{Pair}_B(x, y, z)$ *and* $\mathsf{Pair}_B(x', y', z')$ *equivalent exactly when the two input endpoints are transported by the* $\mathsf{Module}^{\uparrow}$ *and*

$\mathsf{VecSpace}^{\uparrow}$ *classifiers and the scalar endpoints are transported by the scalar classifier exposed by those source certificates. Thus the public* $\mathsf{BilinForm}^{\uparrow}$ *object is the pairing relation together with its source certificates and scalar-output classifier, not a host-level function on an external carrier.*

Lean4 (definition) `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface`

Theorem 146.8 (BilinForm root bilinearity rows)**.** *For the root pairing surface of Definition 146.7, the* $\mathsf{BilinForm}^{\uparrow}$ *certificate contains the following three rows.*

1. *Left additivity: if* x_1 *and* x_2 *are carried source endpoints and* y *is a carried partner, then the pairing row for the module sum of* x_1 *and* x_2 *with* y *is classified with the scalar sum of the two rows* $\mathsf{Pair}_B(x_1, y, z_1)$ *and* $\mathsf{Pair}_B(x_2, y, z_2)$*.*
2. *Right additivity: if* y_1 *and* y_2 *are carried source endpoints and* x *is a carried partner, then the pairing row for* x *with the module sum of* y_1 *and* y_2 *is classified with the scalar sum of the two rows* $\mathsf{Pair}_B(x, y_1, z_1)$ *and* $\mathsf{Pair}_B(x, y_2, z_2)$*.*
3. *Scalar compatibility: scalar multiplication on either input is transported to the scalar-output endpoint through the scalar-action classifier supplied by* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$*.*

Proof. Package the three displayed clauses as certificate rows over the same BHist pairing ledger. The left and right additive clauses use the module addition rows from $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ and the vector endpoint classifier from $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ to keep both input positions inside the carried source surface.

For scalar compatibility, read the scalar action through the shared scalar source of the module and vector-space certificates. Classifier transport in those two dependency certificates moves equivalent input rows to equivalent scalar-output rows.

The resulting ledger is bilinear because every visible pairing output is justified by one of the three rows and every transport step is recorded through the source classifiers named in the root pairing surface. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface bilinearity rows`
Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface input transport`

Theorem 146.9 (BilinForm root symmetry and antisymmetry rows)**.** *A* $\mathsf{BilinForm}^{\uparrow}$ *certificate over Definition 146.7 must choose exactly one public exchange row for the pairing surface. The symmetric row states that carried endpoints satisfy*

$$\mathsf{Pair}_B(x, y, z) \quad \textit{exactly when} \quad \mathsf{Pair}_B(y, x, z).$$

The antisymmetric row states that exchanging the carried endpoints sends the scalar endpoint to its certified additive inverse:

$$\mathsf{Pair}_B(x, y, z) \quad \textit{exactly when} \quad \mathsf{Pair}_B(y, x, -z).$$

The selected row is part of the certificate ledger; it is not a consequence of the three bilinearity rows alone.

Proof. Start from the pairing relation and the two source classifiers. Bilinearity records additive and scalar behavior in each input slot, but it does not determine how the two slots compare after exchange.

The certificate therefore carries an explicit exchange row. In the symmetric case, the exchanged ledger keeps the same scalar endpoint and transports only the input endpoints through the two source classifiers.

In the antisymmetric case, the exchanged ledger also records the additive inverse endpoint in the scalar source. The inverse is certified by the scalar-output classifier, so the exchanged row remains a $\mathsf{BilinForm}^{\uparrow}$ pairing row rather than an external sign convention. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface symmetry rows`

Theorem 146.10 (BilinForm root nondegeneracy row)**.** *For the root pairing surface of Definition 146.7, nondegeneracy is the pair of kernel classifier rows below.*

1. *Left kernel row: if a carried endpoint x pairs to the zero scalar endpoint with every carried partner y, then x is classified with the zero vector endpoint of the module and vector-space source.*
2. *Right kernel row: if a carried endpoint y is paired to the zero scalar endpoint by every carried partner x, then y is classified with the zero vector endpoint of the module and vector-space source.*

These two rows are the nondegeneracy obligations consumed by Clifford, symplectic, and inner-product certificates.

Proof. For the left kernel row, quantify over all carried partners accepted by the source classifiers and assume each pairing ledger ends at the certified scalar zero. The module and vector-space certificates supply the zero vector endpoint and the classifier used to compare x with it.

For the right kernel row, use the same argument with the two input slots exchanged. If the certificate chose the symmetric exchange row, the exchange transport can reuse the left kernel ledger. If it chose the antisymmetric exchange row, the scalar zero is fixed by additive inverse, so the right kernel ledger still repacks through the same zero endpoint.

Thus nondegeneracy is a classifier statement at the BHist boundary: a vector is zero exactly at the source classifier when all certified pairings against it land at the scalar zero endpoint. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface nondegeneracy row`

Theorem 146.11 (BilinForm root ledger exactness boundary)**.** *The $\mathsf{BilinForm}^{\uparrow}$ obligation boundary consists exactly of the carrier rows of Definition 146.7, the pairing-output classifier, the three bilinearity rows of Definition 146.8, the selected symmetry-or-antisymmetry row of Definition 146.9, the two nondegeneracy rows of Definition 146.10, stability under* hsame *and source-classifier transport, the finite pairing ledger, exactness of visible scalar outputs against that ledger, and the dependency rows for* $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormRootPairingSurface ledger exactness boundary`

Proof. Collect the visible rows in the order displayed by the boundary statement. Carrier and classifier rows determine which BHist endpoints may be paired and how scalar outputs are compared.

The bilinearity, exchange, and nondegeneracy rows then give the algebraic obligations that downstream geometry consumers may cite. Stability under hsame and under the two source classifiers ensures that replacing any carried endpoint by an equivalent endpoint preserves the same visible scalar-output classification.

Finally, ledger exactness says that every public pairing output arises from the finite pairing ledger and every ledger row has a public scalar-output classifier. This is the precise boundary exported to Clifford, symplectic, and inner-product certificates. □

Definition 146.12 (BilinForm BHist obligation surface)**.** *A* $\mathsf{BilinForm}^{\uparrow}$ BHist *obligation surface is a package*

$$\mathcal{B} = (\mathcal{M}, \mathcal{V}, \ell, r, s, a, \nu, \eta_{\mathcal{M}}, \eta_{\mathcal{V}}, \lambda)$$

where $\mathcal{M}$ *is the* $\mathsf{Module}^{\uparrow}$ *source certificate,* $\mathcal{V}$ *is the* $\mathsf{VecSpace}^{\uparrow}$ *vector-source certificate,* $\ell : \mathsf{BHist}$ *and* $r : \mathsf{BHist}$ *are the hidden left and right vector-source rows,* $s : \mathsf{BHist}$ *is the scalar-action provenance row,* $a : \mathsf{BHist}$ *is the additive provenance row,* $\nu : \mathsf{BHist}$ *is the pairing endpoint row,* $\eta_{\mathcal{M}}$ *and* $\eta_{\mathcal{V}}$ *are the displayed* $\mathsf{NameCert}$ *witnesses for the two sources, and* λ *is the exact ledger binding the left source, right source, scalar actions, additive actions, and pairing endpoint to those two source certificates.*

The bilinear pairing row is the BHist*-indexed predicate* $\mathsf{BilinPair}_{\mathcal{B}}(\ell, r, \nu)$ *saying that the endpoint* ν *is generated from the two carried vector-source rows by the displayed scalar-action and additive ledgers. The classifier on two packages is hsame-stable endpoint agreement:*

$$\mathsf{BilinCls}(\mathcal{B}, \mathcal{B}') := \mathsf{hsame}(\ell, \ell') \wedge \mathsf{hsame}(r, r') \wedge \mathsf{hsame}(\nu, \nu')$$

together with the requirement that the transported endpoint comparison remains within the same $\mathsf{BilinPair}$ *row under the* $\mathsf{Module}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *source classifiers. The exported ledger is exactly the hidden left vector-source provenance, hidden right vector-source provenance, scalar-action provenance, additive provenance, and pairing-endpoint provenance; it exports no host bilinear map, quotient representative, matrix coordinate system, standard bridge, or Lean target.*

```
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface ledger empty inversion
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface endpoint empty inversion
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface empty ledger source empty
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface empty ledger inversion
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface semantic name certificate
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface swapped ledger transport
Lean4 ✓  BEDC.Derived.BilinFormUp.BilinFormUp StdBridge
```

146.2 The certificate

Theorem 146.13 (BilinForm carrier and classifier obligations)**.** *For the surface of Definition 146.12, carrier transport, bilinearity provenance, and classifier reflexivity, symmetry, and transitivity are explicit certificate obligations. Carrier transport must send accepted* $\mathsf{BilinPair}_{\mathcal{B}}(\ell, r, \nu)$ *rows to accepted rows whenever the* $\mathsf{Module}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *source certificates transport the two vector-source rows and the* Cont *scope transports the scalar-action and additive provenance. Bilinearity must be recorded by four* BHist *rows: left scalar action, right scalar action, left additive composition, and right additive composition, each transporting its pairing endpoint through the same classifier. The classifier must be reflexive, symmetric, and transitive by the componentwise* hsame *rows on* ℓ*,* r*, and* ν*, with the transported endpoint still scoped by the same source* $\mathsf{NameCert}$ *witnesses.*

Proof. Project the two source certificates from the carrier package. The $\mathsf{VecSpace}^{\uparrow}$ row licenses the left and right vector-source histories, while the $\mathsf{Module}^{\uparrow}$ row licenses scalar action and addition on those histories.

Carrier transport is then componentwise. Use the source NameCert rows to keep the transported vector histories inside the accepted sources, and use Cont scope to keep the scalar-action and additive ledgers attached to the transported endpoint row. The pairing predicate is defined only from those displayed rows, so the transported package remains an accepted BilinPair row.

Bilinearity is read from the same provenance. Left scalar action, right scalar action, left additive composition, and right additive composition are all finite Cont ledgers over the module source, and the endpoint of each ledger is classified by the displayed BilinCls relation.

Reflexivity, symmetry, and transitivity of the classifier are the corresponding hsame operations on the left source, right source, and endpoint components. Repacking these three comparisons with the transported source certificates gives the required classifier rows without any ambient bilinear-map equality. □

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface carrier classifier obligations`

Theorem 146.14 (BilinForm symmetry and antisymmetry obligations)**.** *The same* BHist *pairing carrier supports two alternative certificate rows. A symmetric* $\mathsf{BilinForm}^{\uparrow}$ *row must carry, for every accepted pair* (ℓ, r)*, a transported endpoint comparison between* $\mathsf{BilinPair}_{\mathcal{B}}(\ell, r, \nu)$ *and the swapped row* $\mathsf{BilinPair}_{\mathcal{B}}(r, \ell, \nu')$ *with* $\mathsf{hsame}(\nu, \nu')$*. An antisymmetric* $\mathsf{BilinForm}^{\uparrow}$ *row must carry the same swapped comparison together with the* $\mathsf{Module}^{\uparrow}$ *additive inverse provenance certifying that* ν' *is the inverse endpoint of* ν *under the module-addition classifier.*

These rows are alternatives over the same carrier and classifier; the certificate may expose either symmetry, antisymmetry, both when the inverse row is classified with the endpoint row, or neither when only bare bilinearity is being recorded.

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface symmetry antisymmetry obligations`

Proof. For the symmetric row, swap the two vector-source histories and use the $\mathsf{VecSpace}^{\uparrow}$ source certificate to keep both endpoints accepted. The pairing endpoint comparison is exactly the displayed $\mathsf{hsame}(\nu, \nu')$ row, transported through the same Cont scope that generated the original scalar-action and additive ledgers.

For the antisymmetric row, perform the same swap, then read the inverse endpoint through the additive structure supplied by $\mathsf{Module}^{\uparrow}$. The ledger records the original endpoint, the swapped endpoint, and the additive inverse provenance; classifier transport turns this provenance into the displayed endpoint comparison.

Both rows use the carrier of Definition 146.12. The distinction is only which certificate row is required at the swapped endpoint, so no second pairing carrier or host coordinate presentation is introduced. □

Theorem 146.15 (BilinForm nondegeneracy ledger and exactness obligations)**.** *The nondegeneracy readback obligation says that a carried left vector-source row* ℓ *is classified with the zero vector row of* $\mathsf{VecSpace}^{\uparrow}$ *whenever every accepted right vector-source row* r *has pairing endpoint classified with the zero scalar endpoint through* $\mathsf{BilinPair}_{\mathcal{B}}(\ell, r, \nu)$*. The right readback obligation is the symmetric statement with left and right exchanged.*

The ledger-boundary obligation says that every exported witness used by carrier transport, bilinearity, symmetry, antisymmetry, or nondegeneracy is one of the two source NameCert *rows, the left vector-source row, the right vector-source row, the scalar-action provenance, the additive provenance, the endpoint row, or the* Cont *scope connecting those rows. The exactness obligation says conversely that every row in that boundary repacks as part of the public* $\mathsf{BilinForm}^{\uparrow}$ *obligation surface and that no host bilinear map, basis, matrix, quotient representative, Clifford relation, or standard bridge row is exported.*

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface nondegeneracy ledger exactness obligations`

Proof. For left nondegeneracy, assume the displayed zero-endpoint comparison for every accepted right source row. Apply the $\mathsf{VecSpace}^{\uparrow}$ zero classifier to the right argument chosen by the nondegeneracy readback row and use the $\mathsf{Module}^{\uparrow}$ scalar-action and additive provenance to read the pairing endpoint as a finite Cont ledger. The readback row then classifies ℓ with the zero vector row of the vector source. The right readback repeats the same construction with the left and right source rows exchanged.

The ledger boundary is obtained by projection from the package of Definition 146.12. Each carrier, classifier, symmetry, antisymmetry, bilinearity, and nondegeneracy witness consumes only one of the projected rows or one Cont scope step connecting them.

Exactness is the reverse packaging. Any boundary row listed above is already a field of the obligation surface or a source-certificate row named by that field, so it repacks into the public $\mathsf{BilinForm}^{\uparrow}$ certificate ledger. Rows outside the boundary are not exported, which leaves Clifford construction, coordinate matrices, and standard bridges to their own certificates. □

Closure status for $\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$.

Constructive story: A bilinear-form row is generated from two certified sources, $\mathsf{Module}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$, by displaying left and right vector-source histories, scalar-action and additive provenance histories, a scalar endpoint, and a package ledger tying those rows to the two source certificates. The carrier accepts only pairing rows obtained from this visible BHist ledger; a host bilinear map, basis, coordinate matrix, quotient representative, or Clifford relation is not part of the carrier. Two rows are classified by hsame on the left input, right input, and scalar endpoint, together with the requirement that the transported endpoint remains within the same pairing row under the module and vector-space source classifiers. The source is the paired module and vector-space certificate surface, the pattern records left and right additivity, scalar compatibility, the chosen symmetry or antisymmetry row, and nondegeneracy readback, and the ledger keeps the two source certificates, probe data, scalar-action provenance, additive provenance, endpoint row, and continuation scope visible. The five name-certificate obligations are recorded on this surface: accepted pairing rows inhabit the carrier, componentwise hsame gives reflexive, symmetric, and transitive classifier behavior, and carrier transport repacks any source-classified replacement with the same provenance. A row h lies in the signature gap of a $\mathsf{BilinForm}^{\uparrow}$ token p when the source-domain accepts its two inputs and pairing ledger, the signature relation pairs it with an answer pairing row and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records the same scalar endpoint together with its bilinearity, exchange, and nondegeneracy witnesses. The chapter registers this ledgered pairing surface under $\mathsf{BilinForm}^{\uparrow}$; it remains a scoped dependency boundary for downstream geometry and Clifford consumers, with no public coordinate presentation, standard bridge, or independent host bilinear-form object introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge scope is the exported BHist pairing carrier and classifier together with the $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, symmetry, antisymmetry, nondegeneracy, ledger exactness, and Clifford/geometry consumer-boundary rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close a coordinate-matrix presentation, Clifford relation, standard bridge, or standard bilinear-form schema.

Upgrade path: No further formal-axis upgrade remains for this chapter.

Theorem 146.16 (BilinForm zero-left pairing vanishing row)**.** *Let $\mathcal{B}$ be a carried* $\mathsf{BilinForm}^{\uparrow}$ *pairing surface over the source row of Definition 146.1. Write 0_V for the zero vector endpoint supplied by the* $\mathsf{Module}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *source classifiers, write 0_K for the zero scalar endpoint in the shared scalar source, and write $\sim_K$ for the scalar endpoint classifier. If y is any carried right vector-source row and*

$$\mathsf{BilinPair}_{\mathcal{B}}(0_V, y, \nu)$$

is an accepted pairing endpoint row, then the endpoint is classified with the scalar zero:

$$\nu \sim_K 0_K.$$

The proof uses only the zero row of the source certificates, the left-additive part of Definition 146.3, classifier transport for the pairing endpoint, and additive cancellation in the scalar source.

Lean4 ✓ `BEDC.Derived.BilinFormUp.BilinFormBHistObligationSurface zero left pairing vanishing row`

Proof. Fix a carried right vector-source row y and an accepted pairing endpoint $\mathsf{BilinPair}_{\mathcal{B}}(0_V, y, \nu)$. The module and vector-space source certificates carry the zero-vector addition row

$$0_V +_V 0_V \sim_V 0_V.$$

Transport the accepted pairing endpoint along this source-classifier comparison in the left input. By Definition 146.3, the transported endpoint for the row with left input $0_V +_V 0_V$ remains in the same pairing ledger and is classified by the scalar endpoint classifier with ν.

Apply the left-additive row from the same bilinearity transport theorem to the two displayed copies of the accepted row

$$\mathsf{BilinPair}_{\mathcal{B}}(0_V, y, \nu) \quad \text{and} \quad \mathsf{BilinPair}_{\mathcal{B}}(0_V, y, \nu).$$

It classifies the pairing endpoint for the left sum $0_V +_V 0_V$ with the scalar sum $\nu +_K \nu$. Composing this comparison with the input-transport comparison from the preceding paragraph gives

$$\nu \sim_K \nu +_K \nu.$$

All comparisons occur in the scalar endpoint classifier exposed by the same $\mathsf{BilinForm}^{\uparrow}$ source, so no exchange row, nondegeneracy row, coordinate presentation, or host bilinear map is used.

The scalar source is additive through the certified scalar carrier of the module-and-vector-space source. Its right-zero row and additive left-cancellation row are available as the additive component of that source, in the generic form recorded by Definition 46.28. Applying that cancellation lemma to the displayed comparison $\nu \sim_K \nu +_K \nu$ yields

$$\nu \sim_K 0_K.$$

Thus every accepted pairing of the zero vector with the carried partner y is classified with the scalar zero endpoint. □

Chapter 147

A Concrete Naming Certificate for $\mathsf{Clifford}^{\uparrow}$

The Clifford-algebra interface $\mathsf{Clifford}^{\uparrow}$ packages the universal associative algebra generated by a vector-space carrier and a quadratic form q, modulo the relation $v \cdot v = q(v)$. It depends on $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87) and $\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$ (chapter 146), and is the substrate for the Spin and Pin certificates.

147.1 Carrier definition

Definition 147.1 (Clifford carrier package)**.** *Fix a* $\mathsf{VecSpace}^{\uparrow}$ *source carrier, a* $\mathsf{BilinForm}^{\uparrow}$ *source certificate whose diagonal row presents the quadratic form q, and the* $\mathsf{TensorProduct}^{\uparrow}$ *tensor-word source over the same vector histories. The* $\mathsf{Clifford}^{\uparrow}$ *carrier package is the* BHist *presentation whose accepted rows are generated by four constructors: the unit row, vector-injection rows for accepted vector histories, product rows formed by* Cont*-composing tensor-word histories, and boundary rows identifying the product of a vector row with itself with the scalar endpoint classified by q.*

Thus a carried Clifford row is not a host quotient algebra. It is a finite BEDC ledger containing the tensor-word BHist *spine, the bilinear-form diagonal row, and the* hsame *or* Cont *witnesses that place each quadratic boundary inside the same generated history package.*

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordTasteGate single carrier alignment`

Definition 147.2 (Clifford classifier package)**.** *The* $\mathsf{Clifford}^{\uparrow}$ *classifier is the least BEDC classifier on the carrier of Definition 147.1 generated by the* $\mathsf{TensorProduct}^{\uparrow}$ *classifier rows, product congruence for* Cont*-composed tensor-word products, and the quadratic-relation row*

$$v \cdot v \sim q(v) \cdot 1$$

whenever the vector endpoint v is accepted by the $\mathsf{VecSpace}^{\uparrow}$ *source and the scalar endpoint $q(v)$ is classified by the diagonal row of* $\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$.

Classifier transport is NameCert-level transport: replacing a tensor row by a $\mathsf{TensorProduct}^{\uparrow}$*-classified row, a vector row by a source-classified row, or a diagonal scalar row by a* $\mathsf{BilinForm}^{\uparrow}$*-classified row preserves the Clifford classifier through the displayed* hsame *and* Cont *witnesses.*

Lean4 (definition) `BEDC.Derived.CliffordUp.CliffordClassifierPackage`

Theorem 147.3 (Clifford carrier and classifier obligations)**.** *The Clifford word carrier has visible* BHist *endpoints for every accepted finite vector-letter history, and the quadratic-relation classifier of Definition 147.2 is an equivalence relation on these endpoints. Its constructors are exactly tensor-word congruence, bilinear-form transport, local quadratic rows, and the three equivalence closures.*

Proof. Carrier coverage is read from Definition 147.1. Each letter is already a $\mathsf{VecSpace}^{\uparrow}$-carried endpoint, and each finite word carries its $\mathsf{TensorProduct}^{\uparrow}$ bracketing witness, so the endpoint is a concrete BHist history rather than an external algebra element.

For the classifier, induct on the generated relation. The generator cases are the three displayed rows in Definition 147.2. Tensor congruence keeps the same vector-letter ledger up to the $\mathsf{TensorProduct}^{\uparrow}$ classifier, bilinear-form transport keeps the $\mathsf{BilinForm}^{\uparrow}$ row, and the local quadratic row supplies the single visible reduction $v \cdot v \sim q(v) \cdot 1$.

The reflexive, symmetric, and transitive cases are the closure constructors of the least relation. Because every constructor targets the same Clifford word carrier, the induction also proves that no classifier endpoint leaves the accepted BHist carrier. □

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage carrier classifier obligations`

Theorem 147.4 (Clifford product relation stability obligations)**.** *If two Clifford word endpoints are related by the quadratic-relation classifier, then their products with any accepted left or right Clifford word context are again related. In particular, tensor-word concatenation preserves carrier membership and the classifier is stable under the two context operations induced by* $\mathsf{TensorProduct}^{\uparrow}$*.*

Proof. First build the product carrier. Concatenating an accepted context word with an accepted endpoint concatenates two finite vector-letter ledgers and uses the $\mathsf{TensorProduct}^{\uparrow}$ row to record the new bracketing witness. Hence both left and right products remain Clifford word endpoints.

Now induct on the given classifier derivation. Tensor congruence is stable because the context is another tensor word read by the same $\mathsf{TensorProduct}^{\uparrow}$ source. Bilinear-form transport is stable because the transported row remains a $\mathsf{BilinForm}^{\uparrow}$-classified scalar row after the context is attached outside the local letter pair.

For a local quadratic row, the context constructors record exactly the same replacement inside a larger finite word: the subword $v \cdot v$ is replaced by $q(v)$, while the surrounding left and right word ledgers are retained. The equivalence closure steps are preserved by applying the same context operation to each premise in the induction. □

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage product relation stability obligations`
Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage product relation stability`

Theorem 147.5 (Clifford universal ledger exactness obligations)**.** *The Clifford obligation ledger records exactly four kinds of witnesses: vector-letter membership in the* $\mathsf{VecSpace}^{\uparrow}$ *source, tensor bracketing rows from* $\mathsf{TensorProduct}^{\uparrow}$*, bilinear and quadratic rows from* $\mathsf{BilinForm}^{\uparrow}$*, and relation-generation steps for the quadratic classifier. Therefore the Clifford certificate imports no host quotient, no external associative algebra, and no equality principle outside the displayed* BHist *ledger.*

Proof. Coverage follows by unpacking an accepted Clifford word. Its data are a finite vector-letter ledger, the tensor bracketing row for that ledger, and the endpoint BHist. These are the first two witness kinds in the stated ledger.

Soundness of scalar and quadratic information is supplied by the $\mathsf{NameCert}_{\mathsf{BilinForm}^\uparrow}$ source. The bilinear transport row and the local quadratic row $v \cdot v \sim q(v)$ are recorded as rows of the generated classifier, so the scalar endpoint is never read from a host quotient.

Exactness is the induction principle for the least classifier relation. Every accepted relation step is either one of the listed generators or one of the equivalence and context closures already recorded by the relation-generation ledger. Thus any purported Clifford equality must expose one of these rows, which rules out imported algebraic equality as a hidden source. □

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage ledger exactness obligations`
Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage universal ledger exactness`

147.2 The certificate

Theorem 147.6 (Clifford quadratic relation obligation)**.** *For every accepted vector* BHist *row v in the* $\mathsf{VecSpace}^\uparrow$ *source and every diagonal scalar row classified by* $\mathsf{NameCert}_{\mathsf{BilinForm}^\uparrow}$ *as $q(v)$, the* $\mathsf{Clifford}^\uparrow$ *certificate contains a classifier witness identifying the product history $v \cdot v$ with the scalar-unit endpoint $q(v) \cdot 1$. The witness is transported along* hsame, Cont, *the* $\mathsf{TensorProduct}^\uparrow$ *tensor-word classifier, and the* $\mathsf{BilinForm}^\uparrow$ *diagonal classifier; no host-level quotient equality is part of the obligation.*

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordClassifierPackage quadratic relation obligation`

Proof. Start with the vector-injection constructor of Definition 147.1. Applying it twice gives two tensor-word endpoints in the $\mathsf{TensorProduct}^\uparrow$ source, and the product constructor records their product as a BHist row through Cont.

Use the $\mathsf{BilinForm}^\uparrow$ dependency to read the diagonal scalar row $q(v)$. The scalar endpoint is inserted into the Clifford carrier by the unit constructor and the scalar-row classifier supplied by the source certificate.

The quadratic boundary constructor then contributes the displayed classifier row. Transport of the vector representative uses hsame and the $\mathsf{TensorProduct}^\uparrow$ classifier; transport of the scalar endpoint uses the $\mathsf{BilinForm}^\uparrow$ classifier. Induction on the generated boundary witness has the unit and vector cases empty, the product case closed by congruence, and the boundary case exactly the relation above. □

Theorem 147.7 (Clifford quadratic boundary confluence)**.** *Suppose two accepted Clifford word histories are obtained from the same accepted tensor-word context by applying quadratic-boundary rows to two visible vector-square subwords, possibly in different orders, with accepted left and right tensor contexts around each local square. Then the two resulting endpoints are related by the generated* $\mathsf{Clifford}^\uparrow$ *classifier. The comparison uses only the local quadratic rows, tensor-context stability, and transitivity of the generated classifier; it does not use host quotient-algebra equality.*

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordClassifierPackage quadratic boundary confluence`

Proof. First isolate the two local visible squares inside the accepted tensor word. For each square, Definition 147.6 supplies the classifier row from $v \cdot v$ to the corresponding scalar-unit endpoint $q(v) \cdot 1$.

Next place each local classifier row back inside its accepted left and right tensor contexts. Apply Definition 147.4 to close the row under the surrounding product contexts, so each local reduction becomes a classifier row between accepted whole-word histories.

If the two reductions are disjoint, perform them in either order by applying the same context-stability step to the remaining visible square. If one context is nested in the other, first regard the inner reduction as a classifier row inside the outer tensor context, and then apply the outer quadratic row with the inner endpoint already transported.

In both cases the generated classifier contains the two contextualized local steps. Transitivity of the generated relation composes the first-order path with the reverse of the second-order path, yielding a classifier witness between the two endpoints. Since every step is generated inside the Clifford BHist carrier, no host quotient equality is read. □

Theorem 147.8 (Clifford product stability obligation)**.** *If a, a', b, b' are carried Clifford* BHist *rows, a is classified with a', and b is classified with b', then every product row classified from $a \cdot b$ is transported to a product row classified from $a' \cdot b'$. The transport uses* Cont *for product composition,* hsame *for endpoint replacement, the* $\mathsf{TensorProduct}^{\uparrow}$ *classifier on tensor-word spines, and NameCert-level classifier transport for every* $\mathsf{BilinForm}^{\uparrow}$ *quadratic boundary appearing in the witnesses.*

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage product stability obligation`

Proof. Induct on the classifier witness for the left factor. The reflexive generator keeps the product ledger unchanged, the tensor-product generator is transported by the $\mathsf{TensorProduct}^{\uparrow}$ classifier row, and the quadratic generator is transported by Definition 147.6.

Induct on the classifier witness for the right factor in the same product context. The Cont product row is held fixed while the endpoint is replaced by hsame, and product congruence repacks the resulting history as a Clifford product row.

For composite classifier witnesses, concatenate the two transported product ledgers. Associativity of the visible Cont spine supplies the endpoint comparison needed to read the concatenation as the same product history, so the final row remains inside the least classifier of Definition 147.2. □

Theorem 147.9 (Clifford universal factorization obligation)**.** *Let A be any BEDC algebra package whose multiplication is recorded by* BHist *product rows and whose classifier is stable under* hsame*,* Cont*, and NameCert-level transport. If a vector-row map from the shared* $\mathsf{VecSpace}^{\uparrow}$ *source into A sends each accepted vector v to an endpoint whose square is classified with the scalar endpoint $q(v)$ supplied by* $\mathsf{NameCert}_{\mathsf{BilinForm}^{\uparrow}}$*, then every Clifford carrier row has a unique induced classified image in A obtained by recursion on the generated tensor-word presentation. The recursion factors through the* $\mathsf{TensorProduct}^{\uparrow}$ *tensor-word source and respects the quadratic-relation classifier of Definition 147.2.*

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage universal factorization obligation`

Proof. Define the image on generators first. The unit row maps to the unit row of A, and a vector-injection row maps by the given vector-row map. These two clauses are compatible with the source classifiers by the assumed NameCert-level transport for A.

For a product row, use the multiplication row of A and the Cont ledger that composes the already constructed images of the two factors. The induction hypothesis supplies classified images for the factors, and product stability in A transports the composite image across representative changes.

For a quadratic boundary row, use the hypothesis that the square of the vector image is classified with the scalar endpoint $q(v)$. The $\mathsf{BilinForm}^{\uparrow}$ diagonal classifier transports the scalar representative, and Definition 147.6 supplies the corresponding Clifford-side boundary.

Uniqueness is by induction on the same generated presentation. Any factorization must agree on the unit and vector constructors, must preserve product rows because its multiplication is read through Cont, and must preserve the quadratic boundary because its square clause is exactly the diagonal $\mathsf{BilinForm}^{\uparrow}$ clause. □

Theorem 147.10 (Clifford ledger exactness obligation)**.** *The* $\mathsf{Clifford}^{\uparrow}$ *certificate ledger contains exactly the shared* $\mathsf{VecSpace}^{\uparrow}$ *source rows, the* $\mathsf{TensorProduct}^{\uparrow}$ *tensor-word rows, the* $\mathsf{BilinForm}^{\uparrow}$ *diagonal scalar rows, the unit, vector-injection, product, and quadratic-boundary constructors of Definition 147.1, and the classifier witnesses generated in Definition 147.2. Every accepted Clifford* BHist *row and every accepted classifier transport is obtained from this list by finite induction on the generated carrier and classifier witnesses.*

Proof. Soundness follows by reading each constructor. Unit and vector-injection rows are listed directly, product rows are listed with their Cont composition witness, and quadratic-boundary rows are listed with their $\mathsf{TensorProduct}^{\uparrow}$ and $\mathsf{BilinForm}^{\uparrow}$ source witnesses.

Classifier soundness is also by induction. Tensor-product classifier rows, product congruence, and the quadratic-relation row are exactly the three generators in Definition 147.2; reflexive, symmetric, and transitive uses only repack these finite witnesses through hsame.

For exactness, take any accepted Clifford ledger row. Inverting its generated carrier witness gives one of the four carrier constructors. Inverting any accepted classifier witness gives either a source classifier row, a product-congruence row, or a quadratic-boundary row. Hence no additional host quotient, analytic algebra, Spin-row, or Pin-row datum is present in the Clifford certificate. □

Lean4 ✓ `BEDC.Derived.CliffordUp.CliffordCarrierPackage ledger exactness obligation`

Closure status for $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$**.**

Constructive story: A Clifford row is generated as a finite BHist word package over a $\mathsf{VecSpace}^{\uparrow}$ source, a $\mathsf{TensorProduct}^{\uparrow}$ tensor-word source, and a $\mathsf{BilinForm}^{\uparrow}$ diagonal row presenting the quadratic form q. The carrier accepts exactly the unit row, vector-injection rows for accepted vector histories, product rows formed by Cont-composing tensor words, and boundary rows that place the visible square $v \cdot v$ beside the scalar-unit endpoint $q(v) \cdot 1$; host quotient algebras and analytic modules are outside the carrier. The classifier is the least displayed relation generated by tensor-word congruence, product congruence, bilinear-form transport, the local quadratic boundary row, and reflexive, symmetric, and transitive closure within the same accepted word surface. The source is the vector, tensor, and bilinear certificate triple, the pattern records finite word generation, product stability, quadratic confluence, and universal factorization, and the ledger records vector-letter membership, tensor bracketing, bilinear diagonal scalars, unit/vector/product constructors, and generated classifier witnesses. The five name-certificate obligations hold because the unit and vector constructors inhabit the carrier, the generated classifier supplies the equivalence rows by construction, and product-relation stability keeps every classified representative accepted under left and right tensor contexts. A word package h lies in the signature gap of a $\mathsf{Clifford}^{\uparrow}$ token p when the source-domain accepts its vector letters, tensor bracketing, and quadratic boundary rows, the signature relation pairs the word with an answer endpoint and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records exactly the generated word modulo the displayed $v \cdot v = q(v)$ boundary relation. The chapter registers this generated word-and-boundary construction under $\mathsf{Clifford}^{\uparrow}$; it exports no Spin or Pin group, host quotient algebra, analytic Clifford module, or standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface binds the universal factorization and ledger-exactness interface consumed by $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{PinGroup}^{\uparrow}$ to the concrete BHist, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CliffordUp.CliffordCarrierPackage\_product\_relation\_stability`

Bridge status: paperBridge

Not claimed: No Spin or Pin group, host quotient algebra, analytic Clifford module, or standard bridge is closed here.

Upgrade path: No further paper theory-axis grade is recorded for this Clifford word-and-boundary surface.

Chapter 148

A Concrete Naming Certificate for $\mathsf{SpinGroup}^{\uparrow}$

The Spin-group interface $\mathsf{SpinGroup}^{\uparrow}$ packages the connected double cover of the orthogonal group as a subgroup of the Clifford-algebra units. It depends on $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ (chapter 147) and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41).

148.1 Carrier definition

Definition 148.1 (SpinGroupUp BHist carrier packet)**.** *Fix the* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *source of chapter 147 and the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source of chapter 41. A* $\mathsf{SpinGroup}^{\uparrow}$ *carrier packet is the finite* BHist *packet whose visible rows consist of:*

1. *the Clifford unit row and its transported* $\mathsf{Group}^{\uparrow}$ *identity row;*
2. *vector rows accepted by the Clifford source and their norm-one classifier witnesses;*
3. *finite Clifford product rows closed under the carried group operation;*
4. *a cover row naming the induced orthogonal action on vector histories;*
5. *a subgroup row saying the accepted product histories lie inside the Clifford unit group surface;*
6. *an action row recording conjugation of vector histories by accepted spin histories;*
7. *a classifier row generated by the Clifford classifier, the Group classifier,* hsame *endpoint transport, and finite* Cont *product transport;*
8. Pkg *provenance rows for every Clifford source row, Group source row, cover row, subgroup row, action row, classifier row, and finite ledger row.*

The packet is a BEDC carrier presentation. It does not identify spin histories by a host quotient or by a topological connected-component predicate.

Definition 148.2 (SpinGroup root BHist carrier)**.** *The concrete* $\mathsf{SpinGroup}^{\uparrow}$ *root carrier is the finite* BHist *packet whose visible rows are all of the following:*

1. *a carried* $\mathsf{Clifford}^{\uparrow}$ *endpoint row supplied by* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$*;*
2. *a singleton-history* $\mathsf{Group}^{\uparrow}$ *law row supplied by* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *for the same visible endpoint shape;*

3. *a unit-like Clifford endpoint row, namely a Clifford carrier row classified with the Clifford unit endpoint;*
4. Cont *and* hsame *witnesses recording that the displayed multiplication and inverse rows are transported through the Clifford endpoint classifier.*

The packet contains no host orthogonal group, no quotient algebra, and no ambient matrix representation; its carrier evidence is exactly the listed BHist, Cont, hsame, NameCert, *and* UnaryHistory *data.*

Theorem 148.3 (SpinGroup source scope)**.** *Every row of Definition 148.2 is sourced through* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *or* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$*. The Clifford endpoint, unit-like endpoint, product transport, and inverse transport rows are read from the Clifford certificate; the singleton-history associativity, unit, and inverse law rows are read from the group certificate over* UnaryHistory*. No additional source row is admitted by the* $\mathsf{SpinGroup}^\uparrow$ *root carrier.*

Proof. Unpack the root carrier packet. Each constructor in the packet names either a Clifford carrier row, a Clifford classifier row, a group singleton-history law row, or a transport witness connecting these rows.

The Clifford rows are discharged by the carrier, unit, product, and classifier clauses of $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$. The group law rows are discharged by the singleton-history law surface of $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$, with the visible endpoint supplied by the same UnaryHistory spine.

It remains only to account for the transport witnesses. These are ordinary Cont and hsame rows whose endpoints are already present in the two certificates, so repacking them adds no third source. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier source scope`

Theorem 148.4 (Clifford unit lift for SpinGroup)**.** *If a* $\mathsf{Clifford}^\uparrow$ *carrier row is classified with the Clifford unit endpoint, then the same row lifts to the* $\mathsf{SpinGroup}^\uparrow$ *root carrier by pairing it with the singleton* $\mathsf{Group}^\uparrow$ *identity law row and the reflexive* Cont *and* hsame *transport witnesses. The lift preserves the visible Clifford endpoint classifier and records the row as unit-like inside the SpinGroup packet.*

Proof. Start with the displayed Clifford carrier row and its classifier witness to the unit endpoint. The carrier clause of $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ supplies the endpoint row, while classifier reflexivity supplies the stable unit comparison needed by the packet.

Next choose the singleton-history identity row from $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$. Since the endpoint spine is the same unary history, the group row and the Clifford unit row share the visible boundary required by the root carrier.

Finally repack the Clifford row, the singleton identity law, and the reflexive Cont and hsame witnesses into Definition 148.2. No multiplication or inverse row is consumed in this special case. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford unit lift`
Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier unit lift`

Theorem 148.5 (SpinGroup group-law transport)**.** *Let* a*,* b*,* a'*, and* b' *be* $\mathsf{SpinGroup}^\uparrow$ *root-carrier rows whose Clifford endpoints are classified by the* $\mathsf{Clifford}^\uparrow$ *endpoint classifier and whose group-law rows are singleton-history* $\mathsf{Group}^\uparrow$ *rows. If* $\mathsf{hsame}(a, a')$ *and* $\mathsf{hsame}(b, b')$ *hold through the Clifford classifier, then multiplication and inverse rows transported by* Cont *remain inside the* $\mathsf{SpinGroup}^\uparrow$ *root carrier and carry the corresponding* $\mathsf{Group}^\uparrow$ *multiplication and inverse laws.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier group law transport`

Proof. Unpack the two root-carrier rows. The Clifford parts expose carried endpoints and classifier witnesses, while the group parts expose the singleton-history multiplication and inverse law rows.

For multiplication, compose the two Clifford endpoint transports with the Clifford product transport supplied by $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$. The group singleton-history multiplication row is then read through $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and placed over the transported endpoint.

For inverse, use the same endpoint classifier transport with the Clifford inverse endpoint row. The singleton-history inverse law supplies the group side, and the Cont witness repacks the result as a root-carrier inverse row.

The final packet is again of the form in Definition 148.2: the visible Clifford endpoint has moved only by hsame, and every group-law row remains a $\mathsf{Group}^\uparrow$ row over the same unary history. □

Theorem 148.6 (SpinGroup NameCert obligation surface)**.** *The rows in Definition 148.2, together with Definition 148.3, Definition 148.4, and Definition 148.5, form the* $\mathsf{SpinGroup}^\uparrow$ *root* NameCert *obligation surface. The surface consists precisely of Clifford source scope, unit lift, multiplication transport, inverse transport, singleton-history group laws, and finite ledger exactness for these rows.*

Proof. Read the obligation surface row by row. Source scope is supplied by Definition 148.3; the unit face is supplied by Definition 148.4; multiplication and inverse transport are supplied by Definition 148.5.

The classifier side is finite because every row is a named component of the carrier packet. Inverting a proposed $\mathsf{SpinGroup}^\uparrow$ root obligation therefore yields either a Clifford carrier/classifier row, a singleton-history $\mathsf{Group}^\uparrow$ law row, or a displayed Cont or hsame witness between those endpoints.

Repacking these cases gives the stated root NameCert surface. No Spin or Pin global presentation, orthogonal action, matrix model, or host quotient algebra is used by this obligation surface. □

Theorem 148.7 (SpinGroup root ledger exhaustion)**.** *Every finite ledger row exposed by Definition 148.2 is one of the displayed* BHist *rows,* Cont *witnesses,* hsame *witnesses,* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *source rows,* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *source rows, or singleton* UnaryHistory *group-law rows. Conversely, each displayed row of these forms is accepted by the root carrier ledger.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier ledger semantic exhaustion`

Proof. Project an arbitrary ledger row from the root carrier packet. Constructor inversion leaves only the four packet coordinates listed in Definition 148.2: the Clifford endpoint row, the singleton-history group-law row, the unit-like Clifford endpoint row, and the displayed Cont/hsame transport witnesses.

Apply Definition 148.3 to the source rows. It confines the Clifford side to $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ and the group-law side to $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ over the same UnaryHistory spine.

For the converse direction, repack each displayed row through the same constructor projections. The unit-like case uses Definition 148.4, while product and inverse ledger rows use Definition 148.5. Thus no ledger row lies outside the concrete BEDC packet, and no displayed packet row is omitted. □

Theorem 148.8 (SpinGroup root transport closure)**.** *Let x and y be rows of the concrete* $\mathsf{SpinGroup}^\uparrow$ *root carrier of Definition 148.2. If x*hsame*y through the Clifford endpoint classifier and the connecting row is carried by* Cont *over the same* UnaryHistory *group-law spine, then every unit, multiplication, and inverse row admitted from x transports to an admitted root-carrier row over y.*

Proof. Unpack x and y as root carrier packets. Their Clifford endpoint rows are sourced by $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$, and their singleton-history group laws are sourced by $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ by Definition 148.3.

The unit row transports first. If the row is the unit-like Clifford endpoint, Definition 148.4 supplies the accepted target packet after the hsame comparison has moved the visible endpoint.

For multiplication and inverse, apply Definition 148.5. The theorem composes the Clifford endpoint transport with the product or inverse Cont row and then reads the matching singleton-history group law from $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$. The transported row is therefore again a row of Definition 148.2. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier transport closure`

Theorem 148.9 (SpinGroup root NameCert threshold package)**.** *The concrete root packet of Definition 148.2 reaches the* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *threshold determined by* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$*: its carrier rows are* BHist *rows, its classifier movement is by* hsame*, its transport rows are by* Cont*, its group-law evidence is singleton* UnaryHistory *evidence, and its ledger is exhausted by Definition 148.7.*

Proof. Start with the obligation surface of Definition 148.6. It identifies the required source, unit, multiplication, inverse, singleton-history group-law, and ledger faces of the root certificate.

The carrier and source faces are supplied by constructor projection from Definition 148.2 together with Definition 148.3. The unit face is supplied by Definition 148.4.

The transport faces are supplied by Definition 148.5 and Definition 148.8. Finally, Definition 148.7 shows that the ledger has exactly the displayed BEDC rows, so the package has no hidden orthogonal-group, matrix, or quotient-algebra obligation. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier root namecert threshold package`

Definition 148.10 (SpinGroup root public consumer boundary packet)**.** *The* $\mathsf{SpinGroup}^\uparrow$ *root public consumer boundary packet is the finite* BHist *packet whose visible rows are exactly the Clifford source row from* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$*, the singleton* $\mathsf{Group}^\uparrow$ *row from* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$*, the unit endpoint, product endpoint, and inverse endpoint rows, the even-parity* UnaryHistory *ledger, the displayed* Cont *transport rows, the displayed* hsame *classifier rows, and the* Pkg *provenance rows tying these entries to the root package.*

The packet has no field for a host orthogonal group, a Clifford quotient, a matrix model, or a standard bridge row.

Theorem 148.11 (SpinGroup root public consumer boundary coverage)**.** *Every public* $\mathsf{SpinGroup}^\uparrow$ *consumer row at the root boundary projects from Definition 148.10. In particular, the readable rows are the Clifford source row, singleton* $\mathsf{Group}^\uparrow$ *row, unit, product, inverse, even-parity ledger,* Cont*,* hsame*, and* Pkg *rows displayed by that packet.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier public consumer boundary coverage`

Proof. Start with the root threshold package of Definition 148.9. It confines the root certificate to the Clifford source, singleton group-law source, transport rows, classifier rows, unit/product/inverse endpoints, and finite ledger rows.

Project the consumer read through Definition 148.7. The ledger inversion leaves only displayed BHist rows, Cont rows, hsame rows, $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ source rows, $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ source rows, singleton UnaryHistory rows, and Pkg provenance.

Finally repack those projected rows into Definition 148.10. The packet contains exactly the row kinds left by the threshold package and ledger exhaustion, so every public root consumer row is covered by it. □

Theorem 148.12 (SpinGroup root public consumer boundary exhaustion)**.** *Every row displayed by Definition 148.10 repacks as a public* $\mathsf{SpinGroup}^{\uparrow}$ *root consumer row, and every public root consumer row is one of those displayed rows. Thus the root public boundary is exhausted by the Clifford source row, singleton* $\mathsf{Group}^{\uparrow}$ *row, unit/product/inverse endpoints, even-parity* UnaryHistory *ledger,* Cont*,* hsame*, and* Pkg *provenance.*

No host orthogonal group row, Clifford quotient row, matrix-model row, or standard bridge row is in scope for this public root boundary.

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier public consumer boundary exhaustion`

Proof. For coverage, use Definition 148.11. It projects each public root consumer row to one coordinate of the boundary packet.

For repacking, read the coordinates of Definition 148.10. The Clifford source and singleton group-law rows repack through Definition 148.3; the unit endpoint repacks through Definition 148.4; and the product and inverse endpoints repack through Definition 148.5.

The even-parity ledger, Cont, hsame, and Pkg rows repack through Definition 148.7 and then through the threshold package of Definition 148.9. The definition of the boundary packet provides no coordinate for host orthogonal groups, Clifford quotients, matrix models, or standard bridges, so the exhausted public surface cannot read any such row. □

Theorem 148.13 (SpinGroup root public boundary transport stability)**.** *Let r be a public* $\mathsf{SpinGroup}^{\uparrow}$ *root consumer row displayed by Definition 148.10. If r is transported by the displayed* hsame *and* Cont *rows while keeping the same Clifford source row, singleton* $\mathsf{Group}^{\uparrow}$ *row, even-parity* UnaryHistory *ledger, and* Pkg *provenance, then the transported row is again a public* $\mathsf{SpinGroup}^{\uparrow}$ *root consumer row.*

No host orthogonal group, matrix model, Clifford quotient, or standard bridge row enters this public boundary under the transport.

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier public boundary transport stability`

Proof. Project the original public row through Definition 148.11. The projected coordinates are exactly the Clifford source row, singleton $\mathsf{Group}^{\uparrow}$ row, unit, product, inverse, even-parity ledger, displayed Cont/hsame rows, and Pkg provenance.

Transport only the displayed classifier and continuation coordinates. The source coordinates remain fixed by Definition 148.3, and the product and inverse coordinates remain admitted by Definition 148.5 over the same singleton group-law row and unary ledger.

Repack the transported coordinates by Definition 148.12. Since the boundary packet has no coordinate for host orthogonal groups, matrix models, Clifford quotients, or standard bridges, the transported row is still a public root consumer row and no external row has entered the boundary. □

Theorem 148.14 (SpinGroup root public consumer NameCert boundary)**.** *The root public consumer surface for* $\mathsf{SpinGroup}^{\uparrow}$ *is exactly the finite* BHist NameCert *boundary consisting of the Clifford source row, the singleton* $\mathsf{Group}^{\uparrow}$ *row, unit, product, and inverse endpoints, the even-parity* UnaryHistory *ledger, the displayed* Cont *rows, the displayed* hsame *rows, and the* Pkg *provenance*

rows of Definition 148.10. Coverage, exhaustion, and transport stability are precisely the three public boundary fields for this surface.

No host orthogonal group, matrix model, Clifford quotient, or standard bridge row is exported by this public NameCert *boundary.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier public consumer namecert boundary`

Proof. The coverage field is Definition 148.11. It sends any public root consumer read to one of the displayed Clifford, singleton $\mathsf{Group}^{\uparrow}$, endpoint, parity-ledger, Cont, hsame, or Pkg rows.

The exhaustion field is Definition 148.12. It repacks each displayed coordinate as a public root consumer row and proves that no other coordinate is available at the root boundary.

The stability field is Definition 148.13. It shows that transport along the displayed Cont and hsame rows preserves the same finite public packet while keeping the Clifford source, singleton $\mathsf{Group}^{\uparrow}$ source, parity ledger, and Pkg provenance fixed.

Together these three fields are exactly the public consumer NameCert boundary for the finite packet of Definition 148.10. Since each field is formulated only in terms of that packet, the public boundary exports no host orthogonal group, matrix model, Clifford quotient, or standard bridge row. □

Theorem 148.15 (SpinGroup root public boundary no extra row)**.** *Every public* $\mathsf{SpinGroup}^{\uparrow}$ *root consumer row that is stable under the displayed* Cont *and* hsame *transports is one of the rows of Definition 148.10. In particular, no host orthogonal group row, matrix-model row, Clifford quotient row, or standard bridge row can be recovered by transport from this public boundary.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier public boundary no extra row`

Proof. Project the stable public consumer row through Definition 148.11. The projection lands in the finite packet containing only the Clifford source, singleton $\mathsf{Group}^{\uparrow}$ row, unit, product, inverse, even-parity ledger, Cont, hsame, and Pkg coordinates.

Use Definition 148.13 for the transport step. Stability keeps the row inside the same displayed packet and does not create a coordinate outside the packet.

Finally apply Definition 148.12. It says that the packet rows exhaust the public root boundary, so every transported stable row is one of the displayed coordinates. Because the packet has no coordinate for host orthogonal groups, matrix models, Clifford quotients, or standard bridges, none of those rows can be recovered by transport. □

Lemma 148.16 (SpinGroup root local consumer downstream coverage)**.** *Let c be an accepted local consumer of the* $\mathsf{SpinGroup}^{\uparrow}$ *root* BHist *packet. Then the consumer read is exhausted by the finite rows already displayed in Definition 148.2 and Definition 148.10. More precisely, c factors through one of the following local alternatives:*

1. *a* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *or* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row covered by Definition 148.3;*
2. *a unit, product, inverse, singleton* $\mathsf{Group}^{\uparrow}$*, or even-parity* UnaryHistory *row covered by Definition 148.7;*
3. *a displayed* Cont*,* hsame*, or* Pkg *ledger coordinate of the root packet;*
4. *a public coverage coordinate of Definition 148.10, as covered by Definition 148.11 and Definition 148.12.*

Consequently every consumer request for a host orthogonal group, a Clifford quotient, a matrix model, an analytic covering datum, a connected-component predicate, or a bridge row has no accepted coordinate in the $\mathsf{SpinGroup}^{\uparrow}$ *root* BHist *packet.*

Proof. Start with an accepted local consumer read. By Definition 148.11, the read projects to the finite boundary packet of Definition 148.10. The projected coordinate is therefore one of the displayed Clifford source, singleton $\mathsf{Group}^{\uparrow}$, unit, product, inverse, even-parity $\mathsf{UnaryHistory}$, Cont, hsame, or Pkg rows.

Apply Definition 148.3 to the source coordinates. It confines every source row of the root carrier to $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ or $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ over the same unary group-law spine, so no third source can be read by the consumer.

Apply Definition 148.7 to the remaining ledger coordinates. It exhausts every finite ledger row exposed by the root carrier by the displayed BHist rows, Cont witnesses, hsame witnesses, the two $\mathsf{NameCert}$ source rows, and singleton $\mathsf{UnaryHistory}$ group-law rows. The unit case repacks through the displayed unit endpoint, while the product and inverse cases repack through the displayed transport rows already present in the same root packet.

It remains only to check that no accepted local consumer is omitted. This is the converse direction of Definition 148.12: every row displayed by the boundary packet repacks as a public $\mathsf{SpinGroup}^{\uparrow}$ root consumer row, and every public root consumer row is one of those displayed rows. Thus the four alternatives in the statement cover all accepted local consumer reads.

For the excluded requests, suppose a local consumer asks for a host orthogonal group, Clifford quotient, matrix model, analytic covering datum, connected-component predicate, or bridge row. Coverage would have to project that request to a coordinate of Definition 148.10. The definition has no such coordinate, Definition 148.3 admits no third source row, and Definition 148.7 admits no ledger row beyond the displayed finite alternatives. Hence those requests remain outside the accepted $\mathsf{SpinGroup}^{\uparrow}$ root BHist packet. □

Definition 148.17 (SpinGroup double-cover interface row)**.** *A* $\mathsf{SpinGroup}^{\uparrow}$ *double-cover interface row is the finite* BHist *packet over Definition 148.2 whose visible coordinates are the Clifford endpoint row, the singleton* $\mathsf{Group}^{\uparrow}$ *law row, the unit-like Clifford endpoint, the* Cont *and* hsame *transport witnesses between those endpoints, the* $\mathsf{UnaryHistory}$ *group-law spine, and the* Pkg *provenance rows used by the root carrier. The interface exports only these rows to the public double-cover boundary. It does not export a host Clifford quotient, a matrix realization, or an orthogonal-group primitive as an independent source.*

Theorem 148.18 (SpinGroup double-cover source exhaustion)**.** *Every public double-cover row admitted by Definition 148.17 projects through the* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, Cont, hsame, $\mathsf{UnaryHistory}$, *and* Pkg *rows already named by Definition 148.3, Definition 148.5, and Definition 148.7. Conversely, any row obtained from those named sources and repacked into the finite interface packet is accepted as a public double-cover row.*

Proof. Project the public row from Definition 148.17. Constructor inversion leaves the same displayed coordinates as the root carrier: the Clifford endpoint, the singleton group-law row, the unit-like endpoint, the transport witnesses, the unary history spine, and the provenance row.

The source classification is then inherited without a new case split. Apply Definition 148.3 to the Clifford and group-facing coordinates, and apply Definition 148.5 to the product and inverse transport coordinates.

It remains to account for ledger visibility. Apply Definition 148.7; it says that the displayed BHist rows, Cont rows, hsame rows, $\mathsf{UnaryHistory}$ rows, and Pkg provenance rows exhaust the root

ledger. Repacking those rows into the double-cover interface gives the converse inclusion and leaves no host quotient, matrix, or orthogonal primitive outside the named BEDC sources. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier double cover source exhaustion`

Theorem 148.19 (SpinGroup NameCert root obligation package)**.** *The interface of Definition 148.17 forms a root* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *obligation package with the ten visible fields: carrier* BHist *row, Clifford source classifier, singleton* $\mathsf{Group}^\uparrow$ *source classifier,* hsame *stability,* Cont *stability, unit endpoint, multiplication endpoint, inverse endpoint, exact finite ledger, and absence of a standard bridge claim. Each field is read from Definition 148.2, Definition 148.9, and Definition 148.18.*

Proof. Start from the threshold package of Definition 148.9. It supplies the carrier, classifier movement, transport rows, singleton-history group-law evidence, and finite ledger exactness required of the root certificate.

Now refine the package into the ten visible fields. The carrier field is the finite BHist row of Definition 148.2. The two source classifiers are supplied by Definition 148.3. The hsame and Cont stability fields, together with unit, multiplication, and inverse endpoints, are supplied by Definition 148.5 and Definition 148.8.

The ledger and boundary fields are final projections. The ledger is exact by Definition 148.7, while Definition 148.18 shows that the public double-cover interface has no additional source row. Thus the obligation package contains the root certificate surface but does not assert a standard bridge. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier root namecert obligation package`

Theorem 148.20 (SpinGroup Clifford even-unit carrier obligation)**.** *The* $\mathsf{SpinGroup}^\uparrow$ *carrier is the* BHist*-coded row of Clifford unit data whose displayed product is even and whose projection lies in the orthogonal action boundary. Its visible carrier entries are inherited from* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$*: Clifford unit rows, even product witnesses, group operation and inverse rows,* hsame *transport,* Cont *and* Pkg *provenance, and ledger entries naming the projection to the orthogonal action interface. The carrier does not export a host Clifford algebra element apart from these certified rows.*

Proof. Read the Clifford unit rows from chapter 147. They provide the visible BHist product, unit, inverse, even-product, and projection rows used by the spin carrier.

Read the group-facing operation boundary from chapter 41. It supplies the multiplication, unit, inverse, carrier transport, and ledger slots needed to regard the Clifford unit rows as group rows.

Repack the shared hsame, Cont, Pkg, and ledger provenance as the carrier for $\mathsf{SpinGroup}^\uparrow$. The repacking keeps the Clifford and group rows visible and records the orthogonal projection as a boundary row, so it adds no analytic or host-level carrier beyond the displayed BEDC histories. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford even unit carrier obligation`

148.2 The certificate

Theorem 148.21 (SpinGroup carrier obligation)**.** *A* $\mathsf{SpinGroup}^\uparrow$ *certificate carries its object row as a* BHist *predicate over Clifford-even lift rows, rotation/projection endpoint rows, and a source row supplied by* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$*. The carrier is not a host spin group: every admitted spin row is a finite BEDC packet whose visible fields are a Clifford carrier row, an even-parity* UnaryHistory *ledger, a product-unit row, and a projection row into the already certified orthogonal surface.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier carrier obligation`

Proof. Read the carrier slot of the roadmap ten-field template. The traditional target is a spin lift, but the BEDC source is the $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ row together with the group operation rows named by $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$.

The BHist carrier predicate therefore accepts exactly the finite Clifford lift packet, the even-parity UnaryHistory ledger, and the projection endpoint row. The projection is read by Cont from the lift packet to the rotation endpoint, while hsame supplies only classifier transport between carried representatives.

No host spin group is introduced. The certificate surface exports the rows named by the NameCert dependencies and the displayed BEDC ledgers, so any downstream carrier use must project through those rows. □

Theorem 148.22 (SpinGroup classifier obligation)**.** *Two* $\mathsf{SpinGroup}^\uparrow$ *carrier rows are classified exactly when their Clifford lift rows, double-cover projection rows, and even-Clifford ledger rows are compared by* hsame *in the same* BHist *scope. The classifier is componentwise over the lift, projection, and parity ledger; it does not identify two rows by appealing to an external Lie-group equality.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier classifier obligation`

Proof. Use the classifier field of the roadmap ten-field template after Definition 148.21 has fixed the carried rows. The source classifier comes from $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$, and the visible projection endpoint is a BHist row, so the only available comparison is hsame.

For two carrier packets, project their Clifford lift rows, their projection rows, and their even-parity UnaryHistory ledgers. Componentwise hsame on these three rows is stable under the NameCert transport fields, and the Cont projection row keeps the lift endpoint attached to the same public rotation surface.

Conversely, any classifier row for $\mathsf{SpinGroup}^\uparrow$ must be readable through these same projections, because the carrier has no additional equality field. Hence the displayed componentwise comparison is the whole classifier obligation. □

Theorem 148.23 (SpinGroup lift parity ledger)**.** *The hidden ledger of a* $\mathsf{SpinGroup}^\uparrow$ *certificate records two rows: the even-Clifford parity row as* UnaryHistory *data over* BHist *words, and the two-to-one projection row as* Cont *data from the Clifford lift to the orthogonal endpoint. These rows are ledger data of the certificate, not new public carrier primitives.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier lift parity ledger`

Proof. Project the ledger field of the roadmap ten-field template. The Clifford source supplies word histories and product rows, while the spin certificate adds the parity and projection ledgers needed to read a lift as a double-cover point.

The parity row is unary because it records the even length of the Clifford word presentation. It is scoped as UnaryHistory over BHist rather than as an ambient integer parity predicate, and hsame transport is inherited from the Clifford word classifier.

The projection row is a Cont witness from the lift packet to the rotation endpoint. It is hidden ledger data because it witnesses how the public orthogonal row is read from the lift, while the public boundary remains the projection endpoint itself. □

Theorem 148.24 (SpinGroup concrete lift closure)**.** *If two admitted* $\mathsf{SpinGroup}^{\uparrow}$ *lift rows are composable through the Clifford product ledger, then* Cont*-composition of the two lift packets preserves the carrier and transports the classifier. The resulting packet has a Clifford-even lift row, a composed projection row, and a transported even-parity ledger in the same* BHist *certificate scope.*

Proof. Unpack the operation and stability fields of the roadmap ten-field template. Each input lift is already a BHist packet with a Clifford carrier row, an even UnaryHistory ledger, and a projection row.

Compose the lift packets using the Clifford product continuation supplied by $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$. The two even-parity ledgers compose to another even ledger inside the same unary history scope, and the two projection rows compose by Cont to the corresponding orthogonal endpoint.

Classifier transport is componentwise. The lift comparison is transported by the Clifford NameCert classifier, the projection comparison by the composed Cont rows, and the parity comparison by hsame on the unary ledger. These are exactly the carrier fields of Definition 148.21. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier concrete lift closure`

Theorem 148.25 (SpinGroup spin double cover boundary)**.** *The public boundary of* $\mathsf{SpinGroup}^{\uparrow}$ *is the double-cover projection row from a Clifford-even lift to the already certified orthogonal or special-orthogonal surface. The boundary is a* BHist *projection ledger classified by* hsame *and transported by* Cont*; it is not an external Lie group supplied outside the BEDC certificate.*

Proof. Read the exactness and boundary fields of the roadmap ten-field template. The carrier and ledger obligations have already fixed the Clifford lift, the even UnaryHistory row, and the projection endpoint as the complete visible surface.

The public row is therefore the projection endpoint, together with its Cont witness from the Clifford-even lift. The orthogonal or special-orthogonal target is already certified by its own NameCert row, so $\mathsf{SpinGroup}^{\uparrow}$ exports only the double-cover boundary into that surface.

Exactness runs in the reverse direction. Any public spin-boundary use must recover the lift row, parity ledger, and projection continuation from the same BHist packet; rows outside those fields are not part of this certificate. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier spin double cover boundary`

Theorem 148.26 (SpinGroupUp root obligation surface)**.** *The* $\mathsf{SpinGroup}^{\uparrow}$ *NameCert obligation surface consists exactly of the carrier packet of Definition 148.1, the generated classifier on its accepted histories, stability under* hsame *and finite* Cont *product transport, and the finite ledger rows naming unit, vector, product, cover, subgroup, action, classifier, and* Pkg *provenance data. In the roadmap ten-field template, the traditional target is the spin group inside the Clifford units; the BEDC sources are the cited* $\mathsf{Clifford}^{\uparrow}$ *and* $\mathsf{Group}^{\uparrow}$ *certificates; the carrier, classifier, stability, ledger, exactness, and status rows are precisely the rows displayed in the packet; the standard bridge row is empty.*

Proof. Read the carrier field from Definition 148.1. The unit, vector, and product rows are Clifford rows, while the identity and product laws are read through the $\mathsf{Group}^{\uparrow}$ source of chapter 41.

The classifier field is generated only from the two source classifiers, hsame endpoint replacement, and finite Cont product transport. Thus carrier stability follows componentwise: Clifford transport keeps the word-level rows inside chapter 147, and Group transport keeps the unit and product rows inside the group source.

The remaining roadmap fields are the finite ledger and exactness fields. The cover, subgroup, action, and provenance rows are named in the packet, so any public $\mathsf{SpinGroup}^{\uparrow}$ row is obtained by repacking those rows into the ten-field certificate surface. No additional row is available for topology, quotient equality, analytic connectedness, or a standard bridge. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier root namecert obligation surface`

Theorem 148.27 (SpinGroupUp Clifford unit carrier obligation)**.** *A* $\mathsf{SpinGroup}^{\uparrow}$ *carrier is a* BHist *packet whose visible rows are a* $\mathsf{Clifford}^{\uparrow}$ *unit row, an orthogonal-action endpoint row, and a* $\mathsf{Group}^{\uparrow}$ *carrier row. The packet exports no host Lie group, matrix equality, or ambient Clifford algebra equality beyond the displayed BEDC rows.*

Proof. Use the dependency surface stated at the start of this chapter. The $\mathsf{Clifford}^{\uparrow}$ certificate supplies the unit row inside the Clifford-algebra interface, while the $\mathsf{Group}^{\uparrow}$ certificate supplies the public group-carrier row read by downstream group consumers.

The $\mathsf{SpinGroup}^{\uparrow}$ carrier is the ten-field certificate packet that places those two dependency rows beside the orthogonal-action endpoint row. Since the carrier is a BHist packet, all equality and transport visible at the interface is through hsame and the named classifier rows, not through a host matrix or Lie-group equality. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford unit carrier obligation`

Theorem 148.28 (SpinGroupUp Clifford–Group consumer boundary)**.** *A downstream consumer of a* $\mathsf{SpinGroup}^{\uparrow}$ *certificate may read only the* $\mathsf{Clifford}^{\uparrow}$ *source rows, the* $\mathsf{Group}^{\uparrow}$ *source rows, the cover row, the subgroup row, the action row, the generated classifier row, the* hsame *and* Cont *transport witnesses, and the* Pkg *provenance rows appearing in Definition 148.1. In particular, every consumer request name factors through those rows and no request may inspect a host Lie group, a quotient representative, or an analytic connected-component object.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford group consumer boundary`

Proof. Unpack a consumer request against the displayed carrier packet. The only source rows named by the packet are the Clifford rows from chapter 147 and the Group rows from chapter 41.

The cover and action requests are finite ledger reads: the cover row names the induced action on vector histories, while the action row records conjugation by carried spin histories. Both rows retain their Pkg provenance and use the generated classifier for transport.

For a product or classifier request, the consumer must use the same finite Cont spine and hsame endpoint transport carried by the packet. Since no other row is present, the public surface exposes exactly the Clifford source, Group source, cover, action, subgroup, classifier, transport, and provenance rows listed in the statement. □

Theorem 148.29 (SpinGroupUp ledger exclusion)**.** *The public* $\mathsf{SpinGroup}^{\uparrow}$ *ledger excludes host Lie-group topology, quotient equality, analytic connectedness, and external double-cover data. Any accepted spin row must instead exhibit a finite* BHist *packet over the* $\mathsf{Clifford}^{\uparrow}$ *and* $\mathsf{Group}^{\uparrow}$ *sources, with cover, subgroup, action, classifier, transport, and* Pkg *provenance rows as in Definition 148.1.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier ledger exclusion`

Proof. Inspect the ledger rows of the carrier packet. The visible rows are all finite BEDC rows: Clifford unit and vector rows, Group unit and product rows, finite product spines, cover and action rows, subgroup rows, classifier rows, and provenance rows.

Host Lie-group topology and analytic connectedness have no row in this packet. A consumer cannot reconstruct them from hsame or Cont, since those witnesses only transport endpoints and finite product spines already present in the BEDC ledger.

Quotient equality and external double-cover data are also absent. The cover row is a named finite action row on vector histories, not a quotient object; the subgroup row is a carried Group row over Clifford units, not an imported subgroup equality. Therefore every public spin certificate is exactly the finite packet described above. □

Theorem 148.30 (SpinGroupUp double-cover ledger obligation)**.** *The double-cover ledger for a* SpinGroup$^{\uparrow}$ *packet records the Clifford unit endpoint, the projected orthogonal-action endpoint, and the kernel-sign witness as* BHist*,* hsame*,* Cont*, and* Pkg *rows. No additional covering space, topology, or host quotient is exported as public carrier data.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier double cover ledger obligation`

Proof. Read the same dependency rows through the NameCert ten-field template. The Clifford$^{\uparrow}$ side contributes the unit endpoint, and the projected orthogonal-action endpoint records the public target of the double-cover map.

The kernel-sign witness is ledger data: it is carried by BEDC rows using BHist endpoints, hsame transport, continuation provenance Cont, and package evidence Pkg. Repacking those rows into the ledger slot of the SpinGroup$^{\uparrow}$ certificate yields the stated double-cover ledger without adding a host quotient or topological covering object. □

Theorem 148.31 (SpinGroupUp group interface boundary)**.** *Any* Group$^{\uparrow}$ *consumer of a displayed* SpinGroup$^{\uparrow}$ *packet may read only the multiplication, identity, inverse, and classifier rows supplied by that packet. The Clifford multiplication rows, orthogonal projection rows, and kernel-sign witness remain ledger rows and are not exported as extra group operations.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier group interface boundary`

Proof. Project the Group$^{\uparrow}$ dependency from the carrier packet described in Definition 148.27. The public group interface consists of its multiplication, identity, inverse, and classifier rows, all carried through the same BHist packet.

The double-cover data is read separately through Definition 148.30. It supplies provenance for why the group rows come from Clifford units and the orthogonal projection, but it stays in the ledger slot of the ten-field NameCert template. Thus a Group$^{\uparrow}$ consumer receives exactly the group interface and no hidden Clifford or projection operation. □

Theorem 148.32 (SpinGroupUp Clifford-unit source obligation)**.** *A* SpinGroup$^{\uparrow}$ *source row is a displayed* Clifford$^{\uparrow}$ *unit endpoint together with even-product provenance, a* Group$^{\uparrow}$ *multiplication row, and* BHist*,* Cont*, and* Pkg *ledgers tying the unit read and the product read to the same carried source. The row records multiplication and inverse access only through the inherited Clifford-unit and group certificates, and it does not assert an uncarried host-level spin subgroup.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford unit source obligation`

Proof. Read the Clifford dependency as a unit endpoint in the Clifford$^{\uparrow}$ carrier. The even-product provenance is a finite BHist ledger whose factors are carried Clifford generators and whose product endpoint is the displayed unit row.

Read the group dependency as the multiplication and inverse package on those unit endpoints. The $\mathsf{Group}^{\uparrow}$ row supplies the operation laws only for the displayed carried endpoints, while Cont records the product computation and Pkg records the shared provenance.

Combining the two dependency reads gives the $\mathsf{SpinGroup}^{\uparrow}$ source row. Every field is inherited from $\mathsf{Clifford}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, BHist, Cont, and Pkg, so no ambient subgroup predicate or external Clifford algebra quotient is added. □

Theorem 148.33 (SpinGroupUp double-cover fiber classifier)**.** *The double-cover classifier for* $\mathsf{SpinGroup}^{\uparrow}$ *compares two carried spin endpoints by their Clifford-unit carrier rows together with the shared* $\mathsf{OrthogonalGroup}^{\uparrow}$ *action endpoint. Two spin rows are in the same fiber exactly when the Clifford-unit comparisons and the displayed orthogonal-action readback agree in the package ledger; the classifier does not quotient by host-level signs or by an unrecorded kernel object.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier double cover fiber classifier`

Proof. Unpack two $\mathsf{SpinGroup}^{\uparrow}$ source rows using Definition 148.32. Each row exposes a carried Clifford unit, even-product provenance, multiplication and inverse rows, and package provenance.

Compare the Clifford-unit coordinates with the inherited $\mathsf{Clifford}^{\uparrow}$ classifier. Then compare the displayed orthogonal-action endpoint using the shared action readback recorded in the same Pkg ledger.

The fiber classifier is the conjunction of these displayed comparisons. Since the sign ambiguity is represented only by carried Clifford-unit rows and their action endpoint, the construction never forms a host sign quotient or an unrecorded double-cover kernel. □

Theorem 148.34 (SpinGroupUp group consumer boundary)**.** *Downstream* $\mathsf{LieGroup}^{\uparrow}$ *or* $\mathsf{PinGroup}^{\uparrow}$ *consumers of* $\mathsf{SpinGroup}^{\uparrow}$ *may read only the Clifford unit row, multiplication and inverse rows, the double-cover classifier of Definition 148.33, and* NameCert *provenance. They may not inspect a host Lie group, a host sign quotient, or an uncarried Clifford algebra carrier outside the displayed BEDC source package.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier group consumer boundary`

Proof. The source obligation exposes the carried unit and the group operation rows. These are sufficient for group-style consumers to read multiplication, inverse, identity provenance, and the finite even-product ledger.

The double-cover classifier exposes exactly the additional fiber data used by geometry consumers: the Clifford-unit comparison and the shared orthogonal action endpoint. The consumer therefore reads a package of named BHist rows, not an ambient quotient.

The NameCert provenance binds these rows to the $\mathsf{SpinGroup}^{\uparrow}$ interface. Any $\mathsf{LieGroup}^{\uparrow}$ or $\mathsf{PinGroup}^{\uparrow}$ consumer that needs more structure must add its own displayed source rows rather than reading an unrecorded host object through this boundary. □

Theorem 148.35 (SpinGroup even-product classifier obligation)**.** *The* $\mathsf{SpinGroup}^{\uparrow}$ *classifier is inherited from the Clifford unit classifier together with group classifier transport. For carried spin rows it is stable under multiplication, inverse, and replacement of even-product witnesses, and it preserves the projection ledger into the orthogonal action boundary. The classifier surface uses only* BHist *rows,* hsame *transport,* Cont/Pkg *provenance,* NameCert *fields, and the Clifford and group ledgers.*

Proof. Apply chapter 147 to read the Clifford unit classifier and the even-product witness transport rows. These rows compare the displayed Clifford products through certified BHist data.

Apply chapter 41 to transport the same classifier through multiplication and inverse. The group boundary supplies the operation and inverse stability rows without replacing the BEDC comparison by host equality.

The projection ledger is carried along the same Cont and Pkg provenance used by the carrier theorem. Thus multiplication, inverse, and even-witness replacement preserve the classifier and leave the orthogonal action boundary unchanged. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier even product classifier obligation`

Theorem 148.36 (SpinGroup NameCert obligation surface)**.** *The carrier row of Definition 148.20 and the classifier row of Definition 148.35 assemble the* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *obligation surface. This surface contains the Clifford even-unit carrier, group transport rows, classifier stability, inverse and multiplication stability, even-product witness stability, projection ledger,* hsame *transport,* Cont/Pkg *provenance, and the exact visible ledger boundary. It explicitly does not assert connected Lie topology, smooth-manifold structure, analytic covering data, or a topological double cover theorem.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupUp StdBridge`

Proof. Start with the carrier theorem. It fixes the accepted histories as Clifford unit rows with even-product witnesses and an orthogonal-action projection ledger, all read through chapter 147 and chapter 41.

Then apply the classifier theorem. It supplies classifier transport under group multiplication and inverse and preserves the even-product and projection ledger rows.

Repack these rows into the NameCert fields: source, carrier, classifier, stability, exactness, and ledger. Since every row is a BHist, hsame, Cont, or Pkg row inherited from the Clifford and group boundaries, the resulting $\mathsf{SpinGroup}^{\uparrow}$ surface is a BEDC certificate surface only and carries no Lie-topological or analytic covering assertion. □

Theorem 148.37 (SpinGroup spin-boundary inclusion transport)**.** *Let a* $\mathsf{SpinGroup}^{\uparrow}$ *root carrier row be admitted by Definition 148.3. Transport it by* hsame *along the Clifford endpoint and by* Cont *along the shared unary group-law spine. The transported row is still an admitted Spin boundary row: its Clifford unit component, even-word component, and singleton group-law component remain inside the* BHist *surface named by the SpinGroup certificate.*

Proof. Start with the root source scope of Definition 148.3. It exposes the accepted row only as a Clifford source row together with the displayed group-law spine.

Apply Definition 148.8. The hsame Clifford endpoint transport preserves the unit and even-word witnesses, and the Cont group-law transport preserves the singleton group interface.

Finally apply Definition 148.31. It identifies the transported unary group-law spine with the same boundary read by the SpinGroup carrier. Repacking these transported rows gives an admitted Spin boundary row and adds no host orthogonal group or quotient row. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier spin boundary inclusion transport`

Theorem 148.38 (SpinGroup Clifford-unit ledger coverage)**.** *Every ledger row visible at the* $\mathsf{SpinGroup}^{\uparrow}$ *spin boundary is one of the following finite* BHist *rows: a Clifford unit row, a singleton Group law row, a* Cont *transport row, an* hsame *transport row, or a* Pkg *provenance row. No spin-boundary ledger row is supplied by an ambient orthogonal group, matrix model, quotient algebra, or analytic covering datum.*

Proof. Apply Definition 148.7. It exhausts the root ledger by the visible Clifford endpoint, even-product witness, group-law transport, Cont readback, hsame comparison, and Pkg provenance rows.

Then apply Definition 148.27. This identifies the Clifford side of the ledger with unit rows already accepted by the consumed $\mathsf{Clifford}^{\uparrow}$ certificate.

The remaining ledger entries are the singleton group-law row and the transport or provenance rows listed by the root exhaustion theorem. Hence every visible row has one of the five forms in the statement, and there is no further external ledger source to inspect. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier clifford unit ledger coverage`

Theorem 148.39 (SpinGroup PinGroup consumer source exactness)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *spin-inclusion consumer reads exactly the* $\mathsf{SpinGroup}^{\uparrow}$ *boundary rows: the Clifford unit rows, singleton group-law rows,* Cont *transport rows,* hsame *transport rows, and* Pkg *provenance rows exposed by the SpinGroup certificate. It reads no host orthogonal group, matrix model, quotient algebra, analytic covering datum, or separate source outside that boundary.*

Proof. First use Definition 148.37. It shows that the rows transported into the consumer remain Spin boundary rows rather than host-level Clifford or group equalities.

Next use Definition 148.38. The ledger visible to the consumer is exhausted by Clifford unit, singleton group-law, Cont, hsame, and Pkg rows.

Finally apply Definition 148.36 and Definition 149.7. The SpinGroup obligation surface is the exported boundary, and the PinGroup inclusion surface consumes that boundary as its source. Therefore the consumer source is exact and admits none of the excluded host, model, quotient, analytic, or covering data. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier pin consumer source exactness`

Theorem 148.40 (SpinGroup root threshold obligation triple)**.** *An accepted* $\mathsf{SpinGroup}^{\uparrow}$ *root packet exposes exactly three threshold rows: the* BHist *Clifford unit carrier row, the even-word parity row, and the group-law transport row. These rows are visible only through* Cont*,* hsame*, and* NameCert *provenance from Definition 148.1 and Definition 148.2. No additional root row is available outside the threshold package of Definition 148.9 and the obligation surface of Definition 148.36.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier threshold obligation triple`

Proof. Project the accepted packet through Definition 148.2. Its constructors expose only the Clifford endpoint row, the singleton group-law row, the unit-like Clifford row, and the displayed Cont/hsame transport witnesses.

Next compare this projection with the wider packet of Definition 148.1. The wider packet contributes the finite Clifford product rows and the parity ledger, while Definition 148.9 confines them to the same $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ provenance.

Finally use Definition 148.36. It repacks the carrier, classifier, stability, exactness, and ledger faces and leaves only the Clifford unit carrier, even-word parity, and group-law transport as the root threshold rows. Thus the packet has no host quotient, ambient orthogonal group equality, or matrix-model row. □

Theorem 148.41 (SpinGroup root even-word classifier inversion)**.** *Any accepted classifier comparison between* $\mathsf{SpinGroup}^{\uparrow}$ *root packets projects to the even Clifford word row, the inherited group*

carrier row, and the double-cover fiber row. The inversion is internal to the displayed BHist *packet: it reads the even-word component from Definition 148.1, the group carrier component from Definition 148.2, and the fiber component from Definition 148.33. It does not use a host quotient or an ambient orthogonal group equality.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier even word classifier inversion`

Proof. Start with the accepted classifier comparison and project it through Definition 148.36. The classifier face is a BEDC row assembled from hsame, Cont, Pkg, and the displayed Clifford and group certificate rows.

The even component is obtained by projecting the Clifford product part of Definition 148.1. Because Definition 148.9 fixes the root source surface, this projection is the even Clifford word row rather than an ambient parity predicate.

The inherited group carrier row is the singleton-history group-law component of Definition 148.2. The remaining projection is the double-cover fiber classifier supplied by Definition 148.33. These three projected rows exhaust the accepted comparison, so no host quotient or external orthogonal equality is read. □

Theorem 148.42 (SpinGroup root double-cover consumer exhaustion)**.** *A downstream* $\mathsf{PinGroup}^{\uparrow}$ *or* $\mathsf{Clifford}^{\uparrow}$ *consumer of the* $\mathsf{SpinGroup}^{\uparrow}$ *root double-cover boundary can read only the visible Clifford unit source, the parity classifier, inverse and product witnesses, the double-cover fiber classifier, finite* Cont *readbacks, and* Pkg *provenance. These are exactly the rows exposed by Definition 148.1, Definition 148.2, Definition 148.9, Definition 148.33, and Definition 148.36.*

Proof. Project a consumer read through Definition 148.36. The obligation surface lists the carrier, classifier, stability, exactness, and ledger rows available to any downstream consumer.

Apply Definition 148.9 to the root part. It confines carrier reads to the visible Clifford unit source, parity ledger, inverse and product witnesses, finite Cont transport, hsame comparison, and Pkg provenance. The concrete packet definitions provide no other constructor to inspect.

Finally apply Definition 148.33 to the projection boundary. The only extra classifier available there is the double-cover fiber classifier, and it is again a displayed finite BHist row. Repacking the projected rows gives exactly the consumer surface named in the statement, with no host quotient, ambient orthogonal group equality, analytic covering datum, or standard-bridge datum. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier double cover consumer exhaustion`

Corollary 148.43 (SpinGroup BHist packet strict obstruction)**.** *Let a displayed* $\mathsf{SpinGroup}^{\uparrow}$ BHist *packet row be used as a failure, refusal, separation, or countermodel row against a converse double-cover read or against reuse by a downstream consumer. Then the obstruction is strict and local: it blocks only the named route whose readback asks for one of the exported packet coordinates, namely the Clifford-even lift row, the even* UnaryHistory *parity ledger, the double-cover projection endpoint, the projection* Cont *witness, the* hsame *classifier row, or the cited* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source rows. It cannot be reinterpreted as a global impossibility for spin groups, as a host theorem about Clifford or orthogonal groups, or as hidden external validation of an unlisted route.*

Proof. Start with the displayed obstruction row and read it through the public double-cover boundary of Definition 148.25. That boundary has only one consumer-facing readback: the projection endpoint together with its Cont witness from the Clifford-even lift. Exactness in the same theorem forces any public spin-boundary use to recover the Clifford-even lift row, the even UnaryHistory

parity ledger, and the projection continuation from the same finite BHist packet. A row that is not one of these readback coordinates is therefore not a boundary coordinate for the converse route.

Now apply the source-scope theorem Definition 148.3. Every root carrier row is sourced through $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ or $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$: the Clifford endpoint, unit-like endpoint, product transport, and inverse transport rows are read from the Clifford certificate, while the singleton-history group-law rows are read from the group certificate over the same unary history. No third source row is admitted by the $\mathsf{SpinGroup}^\uparrow$ root carrier.

Consequently any consumer-facing use of the obstruction must factor through the displayed list: source rows from $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$, the Clifford-even lift, the parity ledger, the projection endpoint, the projection Cont witness, and the hsame classifier movement. If the proposed converse or reuse route asks for one of those rows and the displayed failure, refusal, separation, or countermodel row denies that coordinate, the route is blocked.

The same factorization also rejects stronger readings. A global impossibility statement would require a row ranging over spin groups outside this finite packet; a host theorem would require a Clifford, orthogonal, quotient, matrix, or topological source not supplied by the two certificates; and hidden external validation would require an accepted route not listed among the boundary readback and source-scope rows. The cited boundary and source-scope theorems provide none of those coordinates. The obstruction is therefore only a local negative certificate for the named converse or reuse route inside the displayed $\mathsf{SpinGroup}^\uparrow$ BHist packet. □

Theorem 148.44 (SpinGroup double-cover fiber ledger)**.** *The double-cover fiber ledger row exposed by* $\mathsf{SpinGroup}^\uparrow$ *is exactly the Clifford-unit comparison, the parity* UnaryHistory *row, the orthogonal-action endpoint, the* Cont *projection witness, and the* Pkg *provenance. These five rows are the fiber face of Definition 148.1 and Definition 148.33; no host sign quotient or ambient orthogonal-group equality is part of the ledger.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier double cover fiber ledger`

Proof. Project the carried packet through Definition 148.1. The visible coordinates include the Clifford unit data, parity history, action endpoint, continuation witness, and package provenance.

Apply Definition 148.33. It compares the two carried spin endpoints only by the Clifford-unit comparison and the shared orthogonal-action endpoint recorded in the package ledger.

The parity row and the Cont witness are the transport data that make this comparison a fiber row. Since the classifier theorem exposes no other constructor, the five displayed rows exhaust the ledger. □

Theorem 148.45 (SpinGroup conjugation action readback)**.** *The action row in Definition 148.1 reads conjugation only from the displayed Clifford product rows, Clifford inverse rows, and* hsame *transport. It does not read a host action, a matrix action, or an unrecorded orthogonal representation outside the* $\mathsf{SpinGroup}^\uparrow$ *packet.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier conjugation action readback`

Proof. Unpack the action coordinate of Definition 148.1. Its source rows are the Clifford-unit endpoint, the multiplication ledger, the inverse ledger, and the displayed hsame transports between the selected endpoints.

Use Definition 148.34 to restrict downstream reads to Clifford unit, multiplication, inverse, double-cover classifier, and provenance rows. The conjugation action is therefore reconstructed from the product-inverse-product spine already present in the packet.

The hsame rows transport the readback between classifier-equal endpoints. No host action row is available after this projection, so the action readback is exactly the displayed Clifford product and inverse readback. □

Theorem 148.46 (SpinGroup boundary consumer exactness)**.** *Every* $\mathsf{PinGroup}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, *or* $\mathsf{Clifford}^{\uparrow}$ *consumer of the* $\mathsf{SpinGroup}^{\uparrow}$ *boundary reads precisely the exported Clifford-unit source, parity classifier, multiplication and inverse rows, conjugation readback, double-cover fiber ledger, finite* Cont *witnesses, and* Pkg *provenance. No host orthogonal group, host quotient, or additional Clifford carrier row is exported.*

Proof. Begin with Definition 148.42. It already confines root double-cover consumers to the visible Clifford unit, parity, inverse, product, fiber classifier, continuation, and provenance rows.

Apply Definition 148.44 to replace the fiber classifier by its exact ledger rows. Apply Definition 148.45 for the action coordinate; that readback contributes only Clifford product, inverse, and hsame transport rows.

Finally use Definition 148.34. Its consumer projection admits the same rows for $\mathsf{PinGroup}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, and $\mathsf{Clifford}^{\uparrow}$ consumers and admits no host orthogonal group or quotient row. Thus the exported boundary is exact. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier boundary consumer exactness`

Theorem 148.47 (SpinGroup conjugation action product law)**.** *Let S and T be accepted* $\mathsf{SpinGroup}^{\uparrow}$ *packets of Definition 148.1. Let s and t be their visible Clifford-unit endpoints, let $s \cdot t$ be the endpoint supplied by the packet's finite Clifford product row, and let v be a Clifford-vector row accepted by the same* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *source. Write*

$$\mathsf{Act}_s(v) := (s \cdot v) \cdot s^{-1}$$

for the conjugation endpoint read by the action row, and write $\sim_{\mathsf{Clifford}^{\uparrow}}$ *for the generated* $\mathsf{Clifford}^{\uparrow}$ *classifier. Then the product action endpoint and the iterated action endpoint are both accepted Clifford-vector action rows in the* $\mathsf{SpinGroup}^{\uparrow}$ *packet, and*

$$\mathsf{Act}_{s\cdot t}(v) \sim_{\mathsf{Clifford}^{\uparrow}} \mathsf{Act}_s(\mathsf{Act}_t(v)).$$

The comparison keeps the same orthogonal-action endpoint and the same Pkg *provenance ledger. It is a Clifford-row classifier comparison, not equality of host linear maps, matrices, or an unrecorded orthogonal-group action.*

Proof. Project the two accepted packets through Definition 148.1. The visible data include the Clifford-unit endpoints s and t, the Clifford product row for $s \cdot t$, the inverse rows for s, t, and $s \cdot t$, the action row on Clifford-vector histories, the orthogonal-action endpoint, and the Pkg provenance rows. By Definition 148.28 and Definition 148.34, no other action source is available to the consumer.

Apply Definition 148.45. The action endpoint for the product packet is read as the Clifford word

$$((s \cdot t) \cdot v) \cdot (s \cdot t)^{-1}.$$

The iterated action endpoint is read as the Clifford word

$$(s \cdot ((t \cdot v) \cdot t^{-1})) \cdot s^{-1}.$$

Both words are accepted Clifford rows because the action readback uses only Clifford product rows, Clifford inverse rows, and hsame transport already present in the packet.

The word comparison is the same BEDC group calculation as Definition 41.96, now read inside the Clifford-unit surface. First, the inverse row for the product is the reversed inverse product row supplied by Definition 41.65:

$$(s \cdot t)^{-1} \sim_{\mathsf{Clifford}^{\uparrow}} t^{-1} \cdot s^{-1}.$$

Multiplication congruence and associativity for the displayed product spine transport this comparison through the outer Clifford products, so

$$((s \cdot t) \cdot v) \cdot (s \cdot t)^{-1} \sim_{\mathsf{Clifford}^{\uparrow}} ((s \cdot t) \cdot v) \cdot (t^{-1} \cdot s^{-1}).$$

Reassociating the same finite product spine gives

$$((s \cdot t) \cdot v) \cdot (t^{-1} \cdot s^{-1}) \sim_{\mathsf{Clifford}^{\uparrow}} (s \cdot ((t \cdot v) \cdot t^{-1})) \cdot s^{-1}.$$

The transport of each reassociation through the Clifford classifier is exactly the product-stability row of Definition 147.8 inside the generated classifier of Definition 147.2. Transitivity of that classifier gives the displayed comparison between $\mathsf{Act}_{s \cdot t}(v)$ and $\mathsf{Act}_s(\mathsf{Act}_t(v))$.

It remains to identify the endpoint and provenance read by the public spin boundary. The double-cover row is not recomputed from a host orthogonal group: Definition 148.33 compares spin packets by the Clifford-unit comparison together with the shared orthogonal-action endpoint in the package ledger. The calculation above changes only the parenthesization of the displayed Clifford product-inverse-product spine and uses the product inverse row already named by the packet. Therefore the orthogonal-action endpoint and the Pkg provenance row remain the same, and the action product law is internal to the displayed $\mathsf{SpinGroup}^{\uparrow}$ boundary. □

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier conjugation action product law`

Theorem 148.48 (SpinGroup conjugation action identity law)**.** *Let S be an accepted* $\mathsf{SpinGroup}^{\uparrow}$ *packet of Definition 148.1 whose visible Clifford-unit endpoint is the identity endpoint e. Let v be any Clifford-vector row accepted by the same* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *source. Write*

$$\mathsf{Act}_e(v) := (e \cdot v) \cdot e^{-1}$$

for the conjugation endpoint read by the action row, and write $\sim_{\mathsf{Clifford}^{\uparrow}}$ for the generated $\mathsf{Clifford}^{\uparrow}$ *classifier. Then $\mathsf{Act}_e(v)$ is an accepted Clifford-vector action row in the same* $\mathsf{SpinGroup}^{\uparrow}$ *packet and*

$$\mathsf{Act}_e(v) \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

The comparison uses only the displayed Clifford identity, inverse, product, hsame, Cont, *and* Pkg *rows; it does not use a host linear action, a matrix model, or an ambient orthogonal-group equality.*

Lean4 ✓ `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier conjugation action identity law`

Proof. Project S through Definition 148.1. The visible coordinates include the Clifford identity endpoint e, the inverse row for e, the finite product rows with Clifford-vector rows, the action row, the classifier row generated by the Clifford classifier and hsame transport, and the Pkg provenance for these rows. By Definition 148.28 and Definition 148.34, no other action source is available to the consumer.

Apply Definition 148.45. The action endpoint for the identity packet is read as the Clifford word

$$(e \cdot v) \cdot e^{-1}.$$

This word is an accepted Clifford action row because the readback uses only the packet's Clifford product rows, inverse rows, and endpoint transports.

The identity endpoint is carried as the same unit row on the Clifford and $\mathsf{Group}^{\uparrow}$ sides of the packet. The inverse row for the identity endpoint is therefore classified with the identity endpoint by the group inverse law over that carried Clifford-unit surface. Transporting this inverse comparison through the right product context gives

$$(e \cdot v) \cdot e^{-1} \sim_{\mathsf{Clifford}^{\uparrow}} (e \cdot v) \cdot e.$$

The transport is an instance of the product-stability row of Definition 147.8 inside the generated classifier of Definition 147.2.

Now apply the two displayed identity laws in the same finite product spine. The right identity law collapses the final factor:

$$(e \cdot v) \cdot e \sim_{\mathsf{Clifford}^{\uparrow}} e \cdot v.$$

The left identity law collapses the initial factor:

$$e \cdot v \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

Transitivity of the generated $\mathsf{Clifford}^{\uparrow}$ classifier composes these three rows and gives

$$(e \cdot v) \cdot e^{-1} \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

Substituting the action readback notation yields $\mathsf{Act}_e(v) \sim_{\mathsf{Clifford}^{\uparrow}} v$.

The calculation has changed only the displayed Clifford product-inverse-product word by inverse and unit rows already present in the packet. The orthogonal readback and Pkg provenance are not recomputed from any host representation, so the identity action law is internal to the public $\mathsf{SpinGroup}^{\uparrow}$ boundary. □

Theorem 148.49 (SpinGroup conjugation action inverse-involution law)**.** *Let S be an accepted $\mathsf{SpinGroup}^{\uparrow}$ packet of Definition 148.1 with visible Clifford-unit endpoint s, inverse endpoint s^{-1}, identity endpoint e, and displayed left-inverse row*

$$s^{-1} \cdot s \sim_{\mathsf{Clifford}^{\uparrow}} e.$$

Let v be a Clifford-vector row accepted by the same $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ source. Write

$$\mathsf{Act}_s(v) := (s \cdot v) \cdot s^{-1}$$

for the conjugation endpoint read by the action row, and write $\sim_{\mathsf{Clifford}^{\uparrow}}$ for the generated $\mathsf{Clifford}^{\uparrow}$ classifier. Then the iterated endpoint

$$\mathsf{Act}_{s^{-1}}(\mathsf{Act}_s(v))$$

is an accepted Clifford-vector action row in the same $\mathsf{SpinGroup}^{\uparrow}$ packet and

$$\mathsf{Act}_{s^{-1}}(\mathsf{Act}_s(v)) \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

The comparison is the displayed inverse-cancellation row of the Clifford-unit ledger transported through the conjugation readback; it is not equality of host linear maps, matrices, or an ambient orthogonal-group action.

Proof. Project S through Definition 148.1. The packet contains the Clifford-unit endpoint s, its inverse endpoint s^{-1}, the identity endpoint e, product rows for $s^{-1} \cdot s$, the action row on Clifford-vector histories, the generated $\mathsf{Clifford}^{\uparrow}$ classifier, and the Pkg provenance for these rows. By Definition 148.28 and Definition 148.34, this is the whole readable surface for the downstream action calculation.

Apply Definition 148.47 with outer spin endpoint s^{-1} and inner spin endpoint s. The product endpoint and the iterated action endpoint are accepted Clifford-vector action rows in the same packet, and the product law gives

$$\mathsf{Act}_{s^{-1} \cdot s}(v) \sim_{\mathsf{Clifford}^{\uparrow}} \mathsf{Act}_{s^{-1}}(\mathsf{Act}_s(v)).$$

By symmetry of the generated classifier, it is enough to compare $\mathsf{Act}_{s^{-1} \cdot s}(v)$ with v.

The displayed left-inverse row in the carried $\mathsf{Group}^{\uparrow}$ surface identifies the Clifford-unit product endpoint with the identity endpoint:

$$s^{-1} \cdot s \sim_{\mathsf{Clifford}^{\uparrow}} e.$$

This row is read on the Clifford side through the same multiplication and inverse coordinates used by the action readback. Applying Definition 148.45, the two action endpoints are the Clifford words

$$((s^{-1} \cdot s) \cdot v) \cdot (s^{-1} \cdot s)^{-1}$$

and

$$(e \cdot v) \cdot e^{-1}.$$

Transport the comparison $s^{-1} \cdot s \sim_{\mathsf{Clifford}^{\uparrow}} e$ through the left factor of the first word by Definition 147.8. For the inverse factor, apply inverse congruence from the carried $\mathsf{Group}^{\uparrow}$ rows, equivalently the inverse-congruence field listed in Definition 41.55, and then transport the resulting comparison through the right product context using the same Clifford product stability. Thus

$$\mathsf{Act}_{s^{-1} \cdot s}(v) \sim_{\mathsf{Clifford}^{\uparrow}} \mathsf{Act}_e(v).$$

Now apply Definition 148.48 to the identity endpoint in the same packet:

$$\mathsf{Act}_e(v) \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

Transitivity of the generated classifier composes the product-law comparison, the transported inverse-cancellation comparison, and the identity-action comparison, giving

$$\mathsf{Act}_{s^{-1}}(\mathsf{Act}_s(v)) \sim_{\mathsf{Clifford}^{\uparrow}} v.$$

The accepted-row assertion for the iterated endpoint was already supplied by the product-law application, and every transport used above is a displayed Clifford product, inverse, hsame, Cont, or Pkg row of the same $\mathsf{SpinGroup}^{\uparrow}$ packet. □

Theorem 148.50 (SpinGroup conjugation action classifier congruence)**.** *Let S be an accepted $\mathsf{SpinGroup}^{\uparrow}$ packet of Definition 148.1. Let s and s' be visible Clifford-unit endpoints of S with $s \sim_{\mathsf{Clifford}^{\uparrow}} s'$. Let v and v' be Clifford-vector rows accepted by the same $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ source with $v \sim_{\mathsf{Clifford}^{\uparrow}} v'$. Write*

$$\mathsf{Act}_s(v) := (s \cdot v) \cdot s^{-1}$$

for the conjugation endpoint read by the action row, and write $\sim_{\mathsf{Clifford}^\uparrow}$ *for the generated* $\mathsf{Clifford}^\uparrow$ *classifier. Then* $\mathsf{Act}_s(v)$ *and* $\mathsf{Act}_{s'}(v')$ *are accepted Clifford-vector action rows in the same* $\mathsf{SpinGroup}^\uparrow$ *packet, and*

$$\mathsf{Act}_s(v) \sim_{\mathsf{Clifford}^\uparrow} \mathsf{Act}_{s'}(v').$$

The comparison is the classifier transport of the displayed product-inverse-product word; it does not use equality of host linear maps, matrices, or an unrecorded orthogonal action.

Proof. Project S through Definition 148.1. The visible rows include the Clifford-unit endpoints s and s', the Clifford-vector rows v and v', the finite Clifford product rows, the inverse rows supplied through the carried $\mathsf{Group}^\uparrow$ surface, the action row on vector histories, the generated $\mathsf{Clifford}^\uparrow$ classifier, and the Pkg provenance tying these rows to the same packet. By Definition 148.28 and Definition 148.34, no other action source is available to a consumer of this packet.

Apply Definition 148.45. The two action endpoints are read as the Clifford words

$$(s \cdot v) \cdot s^{-1}$$

and

$$(s' \cdot v') \cdot (s')^{-1}.$$

These are accepted action rows because the readback uses only Clifford product rows, Clifford inverse rows, and hsame transport already present in S.

The displayed comparison $s \sim_{\mathsf{Clifford}^\uparrow} s'$ lies in the Clifford-unit surface carried by the same packet. The inverse-congruence row listed in Definition 41.55, read through that carried group surface, transports it to

$$s^{-1} \sim_{\mathsf{Clifford}^\uparrow} (s')^{-1}.$$

Now apply the product-stability row of Definition 147.8 inside the generated classifier of Definition 147.2. Applied first to $s \sim_{\mathsf{Clifford}^\uparrow} s'$ and $v \sim_{\mathsf{Clifford}^\uparrow} v'$, it gives

$$s \cdot v \sim_{\mathsf{Clifford}^\uparrow} s' \cdot v'.$$

Applied again to this product comparison and to the inverse comparison above, it gives

$$(s \cdot v) \cdot s^{-1} \sim_{\mathsf{Clifford}^\uparrow} (s' \cdot v') \cdot (s')^{-1}.$$

Substituting the action readback notation yields the asserted classifier row between $\mathsf{Act}_s(v)$ and $\mathsf{Act}_{s'}(v')$.

All rows used in the calculation are read from the same $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ source and the same accepted $\mathsf{SpinGroup}^\uparrow$ packet: the two input classifier comparisons, the inverse-congruence row, the two Clifford product-stability transports, the action readback, and the provenance ledger. Hence the congruence is internal to the packet and keeps the action endpoint within the same public $\mathsf{SpinGroup}^\uparrow$ boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$**.**

Constructive story: A spin row is generated by taking the displayed $\mathsf{Clifford}^\uparrow$ unit surface, pairing it with the singleton-history $\mathsf{Group}^\uparrow$ law surface, and keeping the Clifford endpoint, unit-like endpoint, multiplication and inverse transports, parity history, projection row, continuation witnesses, and provenance in one finite packet. The carrier accepts only those BHist rows whose Clifford, group, UnaryHistory, Cont, hsame, and Pkg fields are present; a host orthogonal group, matrix realization, Clifford quotient, or topological connected-component predicate is not part of the root boundary. Two accepted spin packets classify together by componentwise hsame on the Clifford lift, group-law row, parity ledger, projection endpoint, multiplication and inverse rows, and package provenance, with finite continuation routes carrying the double-cover and conjugation readbacks. The source specification is the combined $\mathsf{Clifford}^\uparrow$ and $\mathsf{Group}^\uparrow$ boundary, the

pattern specification records unit, multiplication, inverse, even-parity, double-cover fiber, and conjugation-action rows, and the ledger keeps the Clifford-unit comparison, parity row, orthogonal-action endpoint, projection witness, and provenance visible. The five name-certificate obligations are supplied by the root obligation package: an accepted packet inhabits the carrier, componentwise hsame gives reflexive, symmetric, and transitive classifier movement, and source-scope plus ledger-exhaustion repack every transported unit, product, inverse, and fiber row inside the same boundary. A packet history h lies in the signature gap of a $\mathsf{SpinGroup}^{\uparrow}$ token p when the source-domain accepts its Clifford-group boundary, the signature relation pairs it with an answer spin packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the unit, multiplication, inverse, parity, projection, and fiber rows are exactly those encoded by p. The chapter registers this Clifford-unit boundary construction under the name $\mathsf{SpinGroup}^{\uparrow}$; it introduces no full double-cover theorem, no topological connectedness proof, and no standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation surface binds the carrier, classifier, transport, unit, multiplication, inverse, and ledger rows over $\mathsf{Clifford}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, BHist, Cont, hsame, UnaryHistory, and Pkg.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SpinGroupUp.SpinGroupRootCarrier\_root\_namecert\_obligation\_package`

Bridge status: bridgeChecked

Not claimed: This boundary surface does not close the full double-cover theorem, topological connectedness, or a standard bridge.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 149

A Concrete Naming Certificate for $\mathsf{PinGroup}^{\uparrow}$

The Pin-group interface $\mathsf{PinGroup}^{\uparrow}$ packages the double cover of the full orthogonal group, extending Spin by a reflection generator. It depends on $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ (chapter 148) and $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ (chapter 147).

149.1 Carrier definition

Definition 149.1 (PinGroupUp reflection-extension carrier)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *reflection-extension carrier is the finite* BHist *carrier obtained from the* $\mathsf{SpinGroup}^{\uparrow}$ *carrier of chapter 148 by adjoining one displayed reflection-generator row inside the* $\mathsf{Clifford}^{\uparrow}$ *carrier of chapter 147. Its classifier is inherited from* hsame*-compatible Clifford transport: two displayed pin rows are classified exactly when the underlying Clifford carrier rows are connected by the finite* Cont *ledger accepted by the source* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *and their spin subrows are classified by the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *surface.*

Theorem 149.2 (PinGroup BHist carrier obligation)**.** *Every* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *carrier is a* BHist *predicate built from the consumed* $\mathsf{SpinGroup}^{\uparrow}$ *and* $\mathsf{Clifford}^{\uparrow}$ *certificate rows. Its accepted histories record the Spin carrier row, the Clifford multiplication and grading rows, a finite reflection-generator ledger, and the classifier boundary used by the displayed orthogonal-action surface. It adds no host Pin-group carrier, host orthogonal group, or external Clifford algebra object.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionExtensionCarrier bhist carrier obligation`

Proof. Unpack the ten-field NameCert template. The source rows are the $\mathsf{SpinGroup}^{\uparrow}$ certificate of chapter 148 and the $\mathsf{Clifford}^{\uparrow}$ certificate of chapter 147, so the carrier field is read as a predicate on BHist histories already accepted by those rows.

The remaining fields attach only classifier, stability, exactness, and ledger information. The reflection generator and orthogonal-action boundary therefore enter as certificate rows inside the same history predicate, not as a host carrier.

Thus the Pin carrier is the displayed BHist predicate generated by Spin rows, Clifford rows, and the finite generator ledger. Since the template has no further carrier slot, no external Pin object is exported. □

Theorem 149.3 (PinGroup reflection-generator obligation)**.** *The reflection-generator row of a* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ *certificate is carried by a finite* Cont *ledger witness over the* $\mathsf{Clifford}^\uparrow$ *certificate rows. The row records a unary generator, its classifier-stable Clifford action, and its compatibility with the Pin carrier. It is not a host orthogonal transformation and does not import a host reflection group.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface reflection generator obligation`

↪ variant `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier reflection generator obligation`

Proof. Read the generator row inside the certificate ledger supplied by the $\mathsf{Clifford}^\uparrow$ source chapter (chapter 147). The ledger witness is a finite Cont entry whose endpoints are carried histories, so the generator is available only through BEDC rows.

Stability is then the stability field of the same ten-field template: replacing endpoints by hsame histories preserves the Clifford action row and the Pin carrier predicate. Exactness says that the generator row is accepted precisely when those carried Clifford and ledger witnesses are present.

Therefore the reflection generator is a finite ledger witness internal to the certificate. The displayed orthogonal meaning is a classifier boundary read from that row, not an imported host transformation. □

Theorem 149.4 (PinGroup Spin-extension obligation)**.** *The* $\mathsf{PinGroup}^\uparrow$ *carrier includes the consumed* $\mathsf{SpinGroup}^\uparrow$ *carrier as its even subcarrier. A Spin witness accepted by chapter 148 embeds into the Pin carrier through the Clifford even-grade rows and the shared classifier, and its image uses the identity reflection-generator ledger entry.*

Proof. Unpack the source field for the $\mathsf{SpinGroup}^\uparrow$ dependency. Its carrier, classifier, and exactness rows are already BEDC rows over BHist, so they may be copied into the Pin carrier field without changing their classifier.

Use the $\mathsf{Clifford}^\uparrow$ grading rows to identify these copied histories as the even subcarrier inside the Pin predicate. The generator ledger contributes the identity row, so no odd reflection generator is required for Spin witnesses.

The template's exactness field then states that the embedded Spin rows are accepted by the Pin carrier exactly through those even Clifford and identity ledger witnesses. This gives the Spin inclusion as a certificate row rather than a separate host subgroup. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier spin extension obligation`

Theorem 149.5 (PinGroup full-orthogonal projection obligation)**.** *The full-orthogonal projection row of a* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ *certificate records the classifier boundary from the Pin carrier to the displayed orthogonal-action surface. The row is sourced from the Clifford action and the finite reflection-generator ledger, and it exports only the carried action classifier, not a host orthogonal group element.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionExtensionCarrier full orthogonal projection obligation`

Proof. Start with a Pin carrier witness. By Definition 149.2, it consists of Spin and Clifford source rows together with the finite reflection-generator ledger.

Apply the classifier field of the ten-field template to read the displayed orthogonal-action surface from those rows. The Clifford action supplies the even part, and Definition 149.3 supplies the finite reflection-generator part.

The exactness field identifies the projection row with that classifier boundary. Hence the public projection records only the BEDC action classifier and its ledger provenance, while excluding a host orthogonal-group carrier. □

149.2 The certificate

Theorem 149.6 (PinGroupUp reflection-generator admission)**.** *The reflection-generator row displayed in Definition 149.1 is admitted by the* $\mathsf{PinGroup}^{\uparrow}$ *carrier precisely as a* BHist *row already carried by the source* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$*, together with the finite* Cont *provenance ledger identifying it as the non-spin generator row of the pin surface.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionGeneratorCarrier admission`

Proof. Project the generator from the Clifford carrier in chapter 147. The row is a displayed BHist carrier entry, so it is available to the pin carrier without forming any host orthogonal-group element.

The hsame classifier and finite Cont ledger carried by the Clifford certificate provide the admissibility certificate for transporting the generator row. Repacking those rows with the source $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ provenance gives the reflection-generator field of Definition 149.1.

No quotient of Clifford units and no equality in a host orthogonal group is used; the admitted object is only the displayed BHist generator row and its $\mathsf{NameCert}$ provenance. □

Theorem 149.7 (SpinGroupUp inclusion surface for PinGroupUp)**.** *Every* BHist *carrier row supplied by* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *embeds into the Definition 149.1 as the spin subrow of the* $\mathsf{PinGroup}^{\uparrow}$ *carrier. Its classifier,* Cont *transport ledger, and* $\mathsf{NameCert}$ *provenance are the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *rows, read inside the ambient Clifford carrier.*

Proof. Start with a carried spin row from chapter 148. By the construction of Definition 149.1, the pin carrier keeps the whole spin carrier and adds only the displayed reflection-generator row.

The spin classifier is not recomputed. It is copied as the inherited hsame-compatible classifier row and transported by the same finite Cont ledger already accepted by $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$.

The Clifford dependency supplies the ambient carrier in which both the spin subrow and the reflection-generator row live. Therefore the inclusion surface is a kernel-level BHist inclusion with $\mathsf{NameCert}$ provenance, not an identification with a host orthogonal subgroup. □

Theorem 149.8 (PinGroupUp Clifford conjugation ledger)**.** *The Clifford conjugation data visible to* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the finite ledger consisting of the spin subrow, the reflection-generator row, the* hsame *classifier transports between their Clifford carrier representatives, and the* Cont *provenance rows supplied by* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$*. This ledger records provenance for pin conjugation without exporting a quotient action or host orthogonal-group equality.*

Proof. Read the two source rows from Definition 149.1: the inherited $\mathsf{SpinGroup}^{\uparrow}$ carrier row and the displayed Clifford reflection-generator row. Both are already BHist rows in the Clifford carrier.

Apply the hsame-compatible transport surface of chapter 147. Its finite Cont ledger records how the Clifford representatives are transported under the displayed conjugation rows, and its $\mathsf{NameCert}$ provenance records the source of each projection.

Combining these projections gives the exact conjugation ledger exposed by $\mathsf{PinGroup}^{\uparrow}$. Since the ledger consists only of BHist, hsame, Cont, and $\mathsf{NameCert}$ rows, it exports no quotient of Clifford representatives and no equality in a host orthogonal group. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface clifford conjugation ledger`

Theorem 149.9 (PinGroupUp Clifford-reflection product closure)**.** *Let a displayed* $\mathsf{PinGroup}^{\uparrow}$ *carrier expose the inherited spin row* S*, the Clifford carrier row* C*, the reflection-generator row* R*, and the Clifford product ledger* μ *supplied by the source* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$*. If* x *is carried by* S *and* R *is carried by* C*, then the displayed product row* $\mu(x, R)$ *is again a carried* $\mathsf{PinGroup}^{\uparrow}$ *row. Its Clifford source is* C*, its reflection source is* R*, and its transport is exhausted by the* hsame *comparisons and* Cont *entries already present in the displayed carrier.*

Proof. Project the spin carrier row, Clifford carrier row, reflection row, product ledger, continuation ledger, and provenance row from Definition 149.1 and the source $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ certificate. These projections give all endpoints used by the product row $\mu(x, R)$.

The Clifford endpoint is read through C, and the reflection endpoint is read through R. The public classifier compares both endpoints only by hsame, so transported product endpoints remain in the displayed finite packet.

Repack the product row by appending the product entry already named by μ to the finite BHist ledger and transporting the same entry by the displayed Cont row. The resulting packet has the same source rows, reflection row, and provenance row, hence it is again a $\mathsf{PinGroup}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier reflection product closure`

Theorem 149.10 (PinGroup odd reflection coset exhaustion)**.** *Every carried non-spin* $\mathsf{PinGroup}^{\uparrow}$ *row is generated by the displayed Clifford reflection row, a* $\mathsf{SpinGroup}^{\uparrow}$ *source row, and the finite* Cont *ledger entries of Definition 149.1. The odd side contains no additional source row beyond the reflection generator and the already displayed Clifford product ledger.*

Proof. Start with a carried row whose classifier is not read through the inherited spin subcarrier. By Definition 149.1, the available non-spin data are the displayed Clifford reflection row, the inherited $\mathsf{SpinGroup}^{\uparrow}$ source row, and the finite Cont ledger.

Apply Definition 149.6 to identify the reflection endpoint as the generator row. Apply Definition 149.7 to keep the spin endpoint inside the inherited source row.

Finally apply Definition 149.9. It repacks the spin endpoint, the reflection endpoint, and the Clifford product ledger as a carried $\mathsf{PinGroup}^{\uparrow}$ row, so every non-spin row is exhausted by those finite entries. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier odd reflection coset exhaustion`

Theorem 149.11 (PinGroup reflection parity determinacy)**.** *The finite reflection ledger of Definition 149.1 determines whether a carried* $\mathsf{PinGroup}^{\uparrow}$ *row is read through the* $\mathsf{SpinGroup}^{\uparrow}$ *subcarrier or through the odd reflection side. This parity reading is invariant under* hsame *transport of the displayed Clifford representatives.*

Proof. Inspect the finite BHist ledger in Definition 149.1. A row with no active reflection-generator entry is read through the spin subcarrier by Definition 149.7.

When the reflection-generator entry is active, apply Definition 149.6. The row is then read on the odd reflection side, and Definition 149.9 records the same choice after multiplying by the displayed Clifford product row.

It remains to check transport. The classifier only compares Clifford representatives by hsame, and the continuation data are finite Cont entries already present in the same ledger. Transport therefore preserves which of the two displayed readings the row uses. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface reflection parity determinacy`

Theorem 149.12 (PinGroup source ledger coverage)**.** *Every public* $\mathsf{PinGroup}^{\uparrow}$ *source projection is covered by exactly the displayed* $\mathsf{SpinGroup}^{\uparrow}$ *and* $\mathsf{Clifford}^{\uparrow}$ *source rows together with the reflection-generator ledger of Definition 149.1. No further public source certificate is exported by the* $\mathsf{PinGroup}^{\uparrow}$ *boundary.*

Proof. Read the public source projections from Definition 149.1. The displayed source rows are the inherited $\mathsf{SpinGroup}^{\uparrow}$ row and the $\mathsf{Clifford}^{\uparrow}$ row carrying the reflection generator and Clifford product ledger.

Apply Definition 149.6 for the generator projection and Definition 149.7 for the spin projection. Apply Definition 149.9 to cover the product rows that combine those two displayed sources.

The remaining public entries are hsame comparisons and finite Cont provenance rows over the same ledger. They do not introduce a new source projection, so the listed SpinGroupUp, CliffordUp, and reflection-generator rows cover the whole public boundary. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface source ledger coverage`

Theorem 149.13 (PinGroupUp Spin and Clifford source scope)**.** *Every source row admitted by a* $\mathsf{PinGroup}^{\uparrow}$ *carrier is one of the displayed* $\mathsf{SpinGroup}^{\uparrow}$ *or* $\mathsf{Clifford}^{\uparrow}$ *source rows of Definition 149.1. The reflection-generator row, the Clifford-product row, and the spin-inclusion row are internal rows over those two sources and do not introduce a third source certificate.*

Proof. Induct over the finite BHist ledger underlying the displayed carrier. The initial source entries are exactly the inherited spin row and the Clifford row from Definition 149.1.

For a reflection-generator entry, the carrier definition reads its source and target through the Clifford row, so the entry contributes no new source. For a spin inclusion entry, the domain is the inherited spin row and the codomain is the displayed Clifford row.

For a Clifford-product entry, the ledger row is μ and its endpoints are the already carried spin and reflection rows. The continuation step preserves the same finite source list by Cont, so the induction closes with only the two displayed source certificates. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier spin clifford source scope`

Theorem 149.14 (PinGroup NameCert boundary surface)**.** *The* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *boundary packages the carrier, classifier, reflection-generator row, Spin inclusion, full-orthogonal projection, exactness, and ledger rows into one certificate surface. The carrier is the* BHist *predicate of Definition 149.2; the reflection row is the finite* Cont *witness of Definition 149.3; the Spin row is the even-subcarrier embedding of Definition 149.4; and the projection row is the classifier boundary of Definition 149.5.*

Proof. Collect the four obligation rows. The carrier row supplies the predicate over BHist, the reflection row supplies the finite generator ledger, the Spin-extension row supplies the even subcarrier, and the projection row supplies the classifier boundary to the orthogonal-action surface.

The ten-field template repacks these rows by placing the consumed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ certificates in the source field, the Pin classifier in the classifier field, the displayed transport clauses in the stability field, and the finite generator and projection witnesses in the ledger field.

Exactness is then the statement that a history is accepted exactly when the listed source, classifier, reflection, Spin-inclusion, projection, and ledger rows are present. This is the required PinGroup boundary surface and contains no host Pin group or host orthogonal transformation. □

Theorem 149.15 (PinGroupUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *obligation surface consists exactly of the reflection-extension carrier of Definition 149.1, its componentwise* hsame *classifier, the reflection-generator row, the Clifford-product ledger, the spin-inclusion surface, and the finite* Cont/Pkg *ledger scope. It claims no data beyond the displayed* $\mathsf{SpinGroup}^{\uparrow}$ *and* $\mathsf{Clifford}^{\uparrow}$ *source certificates.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier semantic name certificate`

Proof. Unpack the carrier packet. The projections expose the spin source, Clifford source, reflection generator, Clifford product, spin inclusion, continuation ledger, and provenance row.

Apply Definition 149.6 to identify the admitted reflection row. Apply Definition 149.9 to identify the product row as a carried $\mathsf{PinGroup}^{\uparrow}$ row. Apply Definition 149.13 to restrict all source rows to the displayed spin and Clifford surfaces.

The classifier contains only componentwise hsame transport and Cont transport of the finite ledger. Repacking those projections gives precisely the named obligation surface, with no additional carrier or source certificate available to downstream consumers. □

Definition 149.16 (PinGroupUp reflection-parity carrier)**.** *The visible reflection-parity carrier of* $\mathsf{PinGroup}^{\uparrow}$ *is the* BHist *predicate generated by two displayed packets: the Spin-inclusion row of Definition 149.4 and the finite reflection-generator ledger of Definition 149.3. A public row lies in the inherited Spin branch when its source projection factors through the Spin-inclusion row. It lies in the reflection branch when its source projection factors through one listed Clifford reflection generator and the Clifford-product ledger of Definition 149.9. The carrier contains no third parity source and no host-level parity predicate.*

Theorem 149.17 (PinGroupUp reflection-parity stability)**.** *The reflection-parity carrier of Definition 149.16 is stable under the transport rows already present in the public* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface. Transporting the Spin row by* hsame *preserves the inherited Spin branch, and transporting the finite reflection ledger by* Cont *preserves the reflection branch.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier stability`

Proof. For the inherited branch, unpack the Spin-inclusion row from Definition 149.15. Its public comparison is componentwise hsame, so an hsame transport of the Spin source and target is again read through the same Spin-inclusion row.

For the reflection branch, unpack the reflection generator and product ledger. Apply Definition 149.6 to identify the admitted generator row, then apply Definition 149.9 to keep products inside the displayed finite BHist ledger.

The continuation row is exactly the Cont field already exposed by Definition 149.15. Repacking the transported Spin row and the transported reflection ledger gives the same two-branch carrier of Definition 149.16. □

Theorem 149.18 (PinGroupUp reflection-parity exactness)**.** *The public* $\mathsf{PinGroup}^{\uparrow}$ *classifier distinguishes exactly the two branches of Definition 149.16. A public row is classified as inherited precisely when the listed* BHist, hsame, Cont, Pkg, *and* NameCert *rows factor through the Spin-inclusion source, and it is classified as reflective precisely when those same rows factor through the Clifford reflection-generator source and its product ledger.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier exactness`

Proof. Start from the public obligation packet of Definition 149.15. Its source fields are only the displayed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ certificates, and its classifier fields are componentwise hsame comparisons plus the finite Cont ledger.

If the row factors through the Spin-inclusion source, the classifier reads the row through the inherited Spin branch. The Pkg and $\mathsf{NameCert}$ projections name the same source certificate, so no reflection-generator witness is used.

If the row factors through a Clifford reflection generator, apply Definition 149.6 to obtain the generator row and Definition 149.9 to obtain the closed product row. The classifier then reads the row through the reflection branch, with the same Cont ledger and provenance row.

Conversely, any row classified by the public packet has one of those two factorizations, because Definition 149.13 exhausts all available source rows. Therefore the classifier is exact for the two visible parity branches. □

Theorem 149.19 (PinGroupUp source exhaustion)**.** *Every public* $\mathsf{PinGroup}^{\uparrow}$ *row is generated either by the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *source or by the* $\mathsf{Clifford}^{\uparrow}$ *reflection-generator source listed in Definition 149.1. No public row of the* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface has a source outside these two displayed origins.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier source exhaustion`

Proof. Unpack a public $\mathsf{PinGroup}^{\uparrow}$ row through Definition 149.15. The carrier, classifier, ledger, and provenance projections are exactly the projections of Definition 149.1.

Apply Definition 149.13. It states that the only admitted source rows are the inherited Spin source and the Clifford source, while reflection-generator, product, and Spin-inclusion rows are internal rows over those sources.

The reflection side is closed under the finite product ledger by Definition 149.9. Repacking the two cases gives source exhaustion for the whole public surface. □

Theorem 149.20 (PinGroup reflection ledger closure)**.** *Every reflection-generator row of the public* $\mathsf{PinGroup}^{\uparrow}$ *surface is a finite* Cont *ledger over carried* $\mathsf{Clifford}^{\uparrow}$ *rows, and this ledger remains stable under* hsame *transport. Thus the reflection branch is anchored in* BHist *rows supplied by the Clifford source certificate, with no external reflection object.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface reflection ledger closure`

Proof. Start from Definition 149.3. It reads the reflection-generator row as a finite Cont witness whose endpoints are carried by the consumed $\mathsf{Clifford}^{\uparrow}$ certificate.

Use the Clifford source certificate recorded in chapter 147. Its carrier and classifier rows are BHist rows stable under hsame and certificate-level transport, so the Pin reflection ledger inherits the same stability.

Repack the transported rows through the public packet of Definition 149.15. The result is exactly the finite reflection ledger visible to $\mathsf{PinGroup}^{\uparrow}$ consumers. □

Theorem 149.21 (PinGroup Spin boundary exhaustion)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *carrier reads the Spin subcarrier exactly through the consumed* $\mathsf{SpinGroup}^{\uparrow}$ *row together with the even-grade Clifford rows already carried by the source certificate. No additional Spin boundary is available outside the displayed* BHist *carrier, classifier, and* Cont *transport rows.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface spin boundary exhaustion`

Proof. Apply Definition 149.4. It embeds the consumed $\mathsf{SpinGroup}^{\uparrow}$ carrier as the Spin branch of the Pin reflection-extension carrier.

Next apply Definition 149.15. The public certificate surface exposes the Spin branch only through its source certificate, the Clifford even-grade rows used by that source, and the finite Cont ledger that transports those rows into the Pin carrier.

Since the surface lists no separate Spin object, every Spin-boundary row of $\mathsf{PinGroup}^{\uparrow}$ is exhausted by those consumed BHist rows and their hsame transports. □

Theorem 149.22 (PinGroup Clifford action consumer surface)**.** *Downstream consumers of the public* $\mathsf{PinGroup}^{\uparrow}$ *Clifford action may read only the reflection ledger, Clifford action row, Spin boundary,* Pkg *provenance,* Cont *transport, and* hsame *witnesses displayed by the* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface. They receive no host orthogonal group and no external Pin object beyond these* BHist*-anchored rows.*

Proof. Read the public action data from Definition 149.15. Its exported fields are the carrier, classifier, source certificates, finite Cont ledger, hsame transport witnesses, and Pkg provenance rows.

Apply Definition 149.20 for the reflection ledger and Definition 149.21 for the Spin boundary. The Clifford action row is the same carried source row used in Definition 149.8.

The source exhaustion theorem Definition 149.19 leaves no further public source. Therefore a downstream consumer cannot read a host orthogonal group or an external Pin object from the public $\mathsf{PinGroup}^{\uparrow}$ action surface. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupUp StdBridge`

Definition 149.23 (PinGroup consumer readback packet)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet is the finite public packet obtained from* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *by retaining exactly the reflection ledger, the Clifford action row, the Spin boundary row, the* Pkg *provenance rows, the finite* Cont *transport rows, and the* hsame *witnesses carried by the public certificate. It contains no host orthogonal group, no external Pin object, and no row not exported by the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface.*

Theorem 149.24 (PinGroup consumer readback packet soundness)**.** *Every accepted public* $\mathsf{PinGroup}^{\uparrow}$ *consumer row projects to the* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet of Definition 149.62.*

Proof. Apply Definition 149.15. It says that the accepted public surface of $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ is carried by the finite carrier, classifier, source, transport, witness, and provenance rows listed in the certificate.

The reflection component is then covered by Definition 149.20. That theorem identifies every reflection read with the displayed reflection ledger and its finite BHist provenance.

The Spin component is covered by Definition 149.21. Hence the remaining public action read is the Clifford action row already exposed by the certificate surface, and all transport or identity data are the displayed Cont and hsame rows. These are exactly the rows retained in the readback packet. □

Theorem 149.25 (PinGroup consumer readback packet exactness)**.** *Any* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet reconstructs the public* $\mathsf{PinGroup}^{\uparrow}$ *Clifford-action consumer surface and exposes no host orthogonal group or external Pin object.*

Proof. Start with a packet as in Definition 149.62. Its reflection ledger, Clifford action row, Spin boundary, Pkg provenance, finite Cont transport, and hsame witnesses are precisely the rows admitted by the public consumer surface of Definition 149.22.

Apply Definition 149.19. The only sources available to the public Pin certificate are the displayed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ source rows together with their transported histories.

Therefore repacking the packet rows reconstructs the same consumer surface, while the source-exhaustion and consumer-surface theorems exclude any host orthogonal group or external Pin object outside those rows. □

Theorem 149.26 (PinGroup downstream unblock package)**.** *The carrier definition of* $\mathsf{PinGroup}^{\uparrow}$ *together with Definition 149.62, Definition 149.63, and Definition 149.64 forms the ten-field downstream package consumed by later Clifford-family and Spin-family chapters: carrier, classifier, reflection ledger, Clifford action row, Spin boundary row, source exhaustion,* Pkg *provenance, finite* Cont *transport,* hsame *witnesses, and exclusion of host orthogonal or external Pin reads.*

Proof. The carrier and classifier are the public $\mathsf{PinGroup}^{\uparrow}$ rows already fixed by Definition 149.15. The reflection ledger, Clifford action row, and Spin boundary are exactly the three rows retained in Definition 149.62.

Soundness is supplied by Definition 149.63; exact readback and exclusion of host data are supplied by Definition 149.64. The remaining provenance, transport, and witness fields are the packet's Pkg, Cont, and hsame rows.

Thus a later Clifford-family or Spin-family consumer may cite this single package instead of separately collecting the carrier, source, ledger, transport, and exclusion rows. □

Theorem 149.27 (PinGroup full-orthogonal action exactness)**.** *The public full-orthogonal action row of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the classifier boundary assembled from the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *source, the Clifford reflection generator, the Clifford product ledger,* hsame*,* Cont*, and* Pkg *rows of the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*. It admits no additional host orthogonal action row outside that boundary.*

Proof. Apply Definition 149.12 to locate the inherited $\mathsf{SpinGroup}^{\uparrow}$ source and the Clifford source inside the finite source ledger. Apply Definition 149.18 to identify the classifier boundary with the parity carrier induced by those two source branches.

Apply Definition 149.19. The only public projections remaining after the carrier read are the listed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ rows, their Cont transports, their hsame comparisons, and their Pkg provenance.

Finally apply Definition 149.65. Its consumer package records the reflection ledger, Clifford action row, Spin boundary row, source exhaustion, transport, witnesses, and provenance as one exported surface, so the full-orthogonal action row is exactly that classifier boundary. □

Theorem 149.28 (PinGroup reflection-parity boundary)**.** *Every public* $\mathsf{PinGroup}^{\uparrow}$ *row factors through exactly one branch of Definition 149.16: either the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *branch or the Clifford reflection branch. No public row factors through both branches except by the inherited Spin classification already recorded in the carrier.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier source projection no confusion`

Proof. Use Definition 149.12 to read the public source ledger as the two displayed sources and their transported histories. Apply Definition 149.18 to classify each public row by the reflection-parity carrier.

For a row classified by the inherited Spin branch, the carrier records exactly the $\mathsf{SpinGroup}^{\uparrow}$ inclusion and its hsame, Cont, and Pkg rows. For a row classified by the Clifford reflection branch, the carrier records the reflection generator and Clifford product ledger.

Apply Definition 149.19 to rule out a third branch. Then apply Definition 149.65 to see that the consumer-visible package exports precisely these two branches and the Spin overlap already classified by the inherited branch. □

Theorem 149.29 (PinGroup source projection separation)**.** *The public source projections of* $\mathsf{PinGroup}^{\uparrow}$ *to* $\mathsf{SpinGroup}^{\uparrow}$ *and* $\mathsf{Clifford}^{\uparrow}$ *are disjoint except for rows classified by the inherited Spin branch of Definition 149.16. Consequently the public certificate has no third source projection.*

Proof. Apply Definition 149.12. It places every public source read in the displayed $\mathsf{SpinGroup}^{\uparrow}$ or $\mathsf{Clifford}^{\uparrow}$ source row, with the advertised BHist histories and no additional projection field.

Apply Definition 149.18. The classifier separates inherited Spin rows from Clifford reflection rows, and it identifies the only common public classification as the inherited Spin branch already present in the reflection-parity carrier.

Apply Definition 149.19 to exclude any remaining public source projection. The downstream package of Definition 149.65 then repacks exactly the two source projections, their transports, witnesses, and provenance for later consumers. □

Definition 149.30 (PinGroup scoped primitive binding)**.** *The scoped primitive binding for* $\mathsf{PinGroup}^{\uparrow}$ *is the finite dependency package whose only primitive rows are* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$. *The inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row and the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row are read only through those two source certificates, and every reflection-parity carrier, classifier, transport, and provenance row of this chapter is a row in that package.*

Theorem 149.31 (PinGroup parity scope transport)**.** *Every reflection-parity carrier transport used by* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the primitive package of Definition 149.69. In particular, parity transport may use only the carried* BHist *histories, displayed* hsame *comparisons, finite* Cont *routes,* Pkg *provenance, and the two source certificate rows from* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface parity scope transport`

Proof. Use Definition 149.83 to identify the parity rows as the inherited Spin branch, Clifford reflection branch, transport rows, and provenance rows.

Then apply Definition 149.86. It repacks the same rows as the scoped reflection-parity source and excludes host orthogonal, matrix, quotient, and third-source data. Hence every carrier transport factors through exactly the primitive binding package. □

Theorem 149.32 (PinGroup consumer boundary exhaustion)**.** *Every downstream consumer of the scoped* $\mathsf{PinGroup}^{\uparrow}$ *surface receives only the two source projections, the reflection-parity classifier, the transport rows, and the provenance package. No downstream consumer receives a host orthogonal group, matrix model, quotient Clifford unit, or additional Pin source row.*

Proof. Project the consumer through Definition 149.88. This gives the consumer readback packet, reflection-parity classifier, Spin inclusion row, Clifford reflection product row, transport rows, finite continuation routes, and provenance rows.

Finally apply Definition 149.65. It identifies the downstream package with the same finite rows, so the exported consumer boundary has exactly the two source projections, classifier, transport, and provenance data stated above. □

Theorem 149.33 (PinGroup reflection-generator transport closure)**.** *The* hsame *transport of the Clifford reflection-generator row preserves* $\mathsf{PinGroup}^{\uparrow}$ *carrier admission exactly through the finite* Cont *ledger of the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*; no transported reflection-generator row is admitted without the carrier row, the reflection-generator obligation, and the closed reflection ledger.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface reflection generator transport closure`

Proof. Project the carrier admission row of Definition 149.1. It exposes the Clifford reflection-generator source, its hsame witness, and its finite Cont transport as the only carrier-facing generator data.

Apply Definition 149.3 to identify the accepted generator row, then apply Definition 149.20. The ledger closure repacks exactly the transported row and excludes any host generator outside the displayed Cont ledger. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionGenerator transport closure`

Theorem 149.34 (PinGroup Spin subrow classifier readback)**.** *Every* $\mathsf{SpinGroup}^{\uparrow}$ *source row embedded in the* $\mathsf{PinGroup}^{\uparrow}$ *carrier is read by the inherited Spin classifier together with the identity reflection ledger. The embedded row therefore contributes no independent Clifford reflection generator beyond the Spin boundary already carried by* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*.*

Proof. Apply Definition 149.4. It places the $\mathsf{SpinGroup}^{\uparrow}$ source row inside the carrier with the inherited classifier, the identity reflection entry, hsame, Cont, and Pkg provenance.

Apply Definition 149.21. The exhaustion theorem rules out a second reflection-generator read on the Spin subrow, so the classifier readback is precisely the inherited Spin row plus the identity reflection ledger. □

Theorem 149.35 (PinGroup orthogonal projection ledger exhaustion)**.** *Every public orthogonal-action projection row of* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the inherited Spin source, the Clifford action row, the reflection-generator parity row, the listed* hsame *transports, the finite* Cont *ledger, and the* Pkg *provenance row of* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*.*

Proof. Apply Definition 149.5 to reduce a public orthogonal-action read to the certificate projection ledger. Apply Definition 149.12 to identify its Spin source, Clifford action, and reflection-generator parity components.

Finally apply Definition 149.22. It exports exactly those source rows with their hsame, Cont, and Pkg data, so no further public orthogonal-action projection remains. □

Definition 149.36 (PinGroup root consumer threshold packet)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is the finite public packet whose rows are the* $\mathsf{SpinGroup}^{\uparrow}$ *source row, the* $\mathsf{Clifford}^{\uparrow}$ *source row, the reflection-parity row of Definition 149.16, the orthogonal-action projection row, the* hsame *transport rows, the finite* Cont *ledger rows, and the* Pkg *provenance rows displayed by* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*. A downstream root consumer reads only this packet when it asks for the Pin reflection boundary.*

Theorem 149.37 (PinGroup root source-threshold coverage)**.** *Every source row visible to the* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is covered by the consumed* $\mathsf{SpinGroup}^{\uparrow}$ *source, the consumed* $\mathsf{Clifford}^{\uparrow}$ *source, and the reflection-parity split of Definition 149.16. No third source projection is visible at the threshold.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupSpinSubrowClassifier readback`

Proof. Use Definition 149.12. It places every public source read in the Spin or Clifford source ledger and records the same BHist histories in the finite certificate packet.

Apply Definition 149.73 to read the Spin subrow through the inherited classifier, and apply Definition 149.18 to separate the Clifford reflection branch from the inherited Spin branch.

Finally apply Definition 149.19. The exhaustion result rules out any source outside those two rows, so the threshold packet has exactly the advertised source coverage. □

Theorem 149.38 (PinGroup root reflection-threshold exactness)**.** *The reflection part of the* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is accepted exactly when the Clifford reflection-generator obligation, the reflection-parity classifier, the orthogonal-action projection, and the finite* hsame *and* Cont *transport rows of* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *are present. It exports no host reflection, matrix reflection, or orthogonal-group element.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier root reflection threshold exactness`

Proof. Project the reflection generator through Definition 149.72. This identifies the admitted reflection row with the carrier row, the generator obligation, and the closed finite reflection ledger.

Apply Definition 149.18. The classifier accepts precisely the inherited Spin branch or the Clifford reflection branch, and the reflection threshold is the latter branch with its displayed parity row.

Now apply Definition 149.74. The orthogonal-action readback is exhausted by the Spin source, Clifford action, parity row, hsame transports, Cont ledger, and Pkg provenance already listed in the threshold packet. Hence no host reflection object is exported. □

Theorem 149.39 (PinGroup root downstream threshold surface)**.** *The downstream root-facing surface of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the consumer threshold packet of Definition 149.75. It supplies the Spin source row, Clifford source row, reflection-parity row, orthogonal projection row,* hsame *transport rows,* Cont *ledger rows, and* Pkg *provenance rows needed by Spin-facing and Pin-facing root consumers, and it supplies no analytic covering, quotient algebra, matrix model, or host orthogonal-group surface.*

Proof. First use Definition 149.76. It fixes the only source rows available to the threshold packet and keeps them inside the Spin and Clifford source ledgers.

Next use Definition 149.77. It fixes the reflection branch by the generator obligation, parity classifier, orthogonal projection, and finite transport ledgers.

Finally apply Definition 149.65 and Definition 149.22. These results repack the same source, reflection, action, hsame, Cont, and Pkg rows for downstream consumers. The packet is therefore the full root threshold surface, with all host analytic and algebraic models excluded. □

Theorem 149.40 (PinGroup root carrier obligation scope)**.** *The carrier rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the* $\mathsf{SpinGroup}^{\uparrow}$ *source row, the* $\mathsf{Clifford}^{\uparrow}$ *source row, the reflection-parity carrier row, and the orthogonal-action projection row displayed by Definition 149.75. No matrix model, analytic cover, quotient algebra, or host orthogonal-group carrier is part of the obligation scope.*

Proof. Begin with Definition 149.75. Its public packet lists the Spin source, Clifford source, reflection-parity carrier, and orthogonal-action projection rows as the carrier-facing data.

Use Definition 149.76. It proves that every source row visible at the threshold is covered by the Spin and Clifford source rows together with the reflection-parity split.

Apply Definition 149.77 and Definition 149.78. These results place the reflection and projection rows inside the same finite threshold packet and exclude all host carrier models from the downstream surface. □

Theorem 149.41 (PinGroup root classifier obligation scope)**.** *The classifier rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *classifier on Spin subrows and the reflection-parity classifier separating the inherited Spin branch from the Clifford reflection branch. No independent Clifford classifier, matrix parity classifier, or quotient classifier is exported.*

Proof. Read the inherited Spin classifier from Definition 149.73. A Spin source row embedded in the Pin carrier is classified by the inherited Spin row together with the identity reflection ledger.

Read the reflection branch from Definition 149.77. It uses Definition 149.18 to identify the accepted reflection side with the reflection-parity classifier and the Clifford reflection-generator obligation.

Finally use Definition 149.78. The downstream threshold surface repacks exactly those classifier rows, so there is no remaining classifier obligation outside the inherited Spin and reflection-parity rows. □

Theorem 149.42 (PinGroup root ledger obligation scope)**.** *The ledger rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the listed* hsame *transports, finite* Cont *rows,* Pkg *provenance rows, and the orthogonal projection ledger exhausted by Definition 149.74. These rows are sufficient for the downstream root surface and supply no analytic covering ledger, quotient-algebra ledger, matrix ledger, or host orthogonal-group ledger.*

Proof. Start from Definition 149.75. The packet lists the hsame transport rows, finite Cont ledger rows, and Pkg provenance rows read by the root consumer.

Apply Definition 149.74. It exhausts the public orthogonal-action projection by the inherited Spin source, Clifford action row, reflection-generator parity row, listed transports, finite Cont ledger, and Pkg provenance.

Use Definition 149.76 and Definition 149.77 to identify the source and reflection parts of the same ledger. Then Definition 149.78 repacks those rows as the downstream root threshold surface, excluding any host analytic, quotient, matrix, or orthogonal-group ledger. □

Theorem 149.43 (PinGroup root scoped obligation package)**.** *The reflection-generator transport, Spin subrow classifier readback, orthogonal projection ledger exhaustion, and the three root scope theorems jointly form the scoped* $\mathsf{PinGroup}^{\uparrow}$ *obligation package over* BHist*,* hsame*,* Cont*,* NameCert*, and the displayed* $\mathsf{Clifford}^{\uparrow}$ *and* $\mathsf{SpinGroup}^{\uparrow}$ *source rows. The package binds the root carrier, classifier, and ledger obligations to the finite threshold packet of Definition 149.75 and exports no analytic covering, quotient algebra, matrix model, or host orthogonal-group surface.*

Proof. First read the root threshold packet. By Definition 149.72, Definition 149.73, and Definition 149.74, every reflection-generator, inherited Spin classifier, and orthogonal projection row is projected through the same finite BHist packet with its hsame, Cont, and Pkg ledgers.

Next bind the three obligation faces. Apply Definition 149.79, Definition 149.80, and Definition 149.81. These results identify the carrier rows, classifier rows, and ledger rows that can appear at the root threshold.

Finally project and repack the finite packet. The first three theorems give the available source and transport projections, while the three scope theorems say that no row outside those projections has an obligation slot. Repacking them through the displayed $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ therefore produces the scoped root package over the listed BEDC kernel objects and the Clifford/SpinGroup sources alone. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface root scoped obligation package`

Theorem 149.44 (PinGroup source projection obligation)**.** *The* $\mathsf{PinGroup}^\uparrow$ *source projection surface is exactly the pair consisting of the consumed* $\mathsf{SpinGroup}^\uparrow$ *source row and the consumed* $\mathsf{Clifford}^\uparrow$ *source row, transported by the displayed* hsame *witnesses and finite* Cont *ledger of* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$*. No public source projection is available unless it factors through one of those two rows and the corresponding* Pkg *provenance entry.*

Proof. Begin with Definition 149.12. It places every public source read in the Spin or Clifford source ledger and records the same BHist histories in the carrier-facing certificate packet.

Apply Definition 149.18. The inherited Spin branch is the Spin projection, while the reflective branch is read through the Clifford source together with the reflection-generator product ledger.

Now use Definition 149.19. It rules out a third source projection, and the downstream package of Definition 149.65 repacks the two source rows with exactly their hsame, Cont, and Pkg witnesses. □

Theorem 149.45 (PinGroup parity classifier obligation)**.** *The reflection-parity carrier of Definition 149.16 is the complete public classifier for* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$*: every admitted public row is classified either by the inherited* $\mathsf{SpinGroup}^\uparrow$ *branch or by the* $\mathsf{Clifford}^\uparrow$ *reflection branch, and the classifier is exact under the carried* hsame *transport rows.*

Proof. Project a public row through Definition 149.67. The row is read through one branch of the reflection-parity carrier, except for the Spin overlap already recorded by the inherited Spin branch.

Apply Definition 149.18. It identifies the Spin branch with rows factoring through the Spin-inclusion source and the reflective branch with rows factoring through the Clifford reflection-generator source and product ledger.

Finally use Definition 149.19 and Definition 149.65. The first excludes any outside classifier source, and the second exports the same parity carrier, transport witnesses, continuation ledger, and provenance rows to downstream consumers. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier parity classifier obligation`

Theorem 149.46 (PinGroup Clifford action ledger obligation)**.** *The Clifford action visible at the* $\mathsf{PinGroup}^\uparrow$ *obligation surface is exhausted by the reflection ledger, Clifford action row, inherited Spin boundary, finite* Cont *ledger,* hsame *witnesses, and* Pkg *provenance rows displayed by* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$*. The obligation surface exports no host orthogonal action, matrix action, quotient algebra, or external Pin object beyond those* BHist *rows.*

Proof. Start with Definition 149.22. It lists the action rows a consumer may inspect: reflection ledger, Clifford action, Spin boundary, provenance, continuation transport, and hsame witnesses.

Apply Definition 149.20 to anchor the reflection part in carried Clifford rows, and apply Definition 149.21 to anchor the Spin part in the consumed Spin source and even-grade Clifford rows.

Use Definition 149.19 to exclude any source outside the Spin and Clifford rows. Then Definition 149.65 repacks the same action, ledger, transport, and provenance rows as the downstream obligation surface. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface clifford action ledger obligation`

Theorem 149.47 (PinGroup parity carrier inversion)**.** *Every accepted row of the reflection-parity carrier of Definition 149.16 has a finite inversion record inside the public* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ *surface. The record is a* BHist *row whose source projection is either the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *Spin row or the* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *Clifford reflection row, together with the public parity classifier, the displayed* Cont *transport ledger, the matching* hsame *witness, and the* Pkg *provenance row. No accepted parity-carrier row has a third source projection.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier parity carrier inversion`

Proof. Begin with Definition 149.67. It factors every public $\mathsf{PinGroup}^\uparrow$ row through the inherited Spin branch or the Clifford reflection branch of the displayed parity carrier.

Use Definition 149.18. The exactness theorem identifies the accepted branch with the public parity classifier and keeps the transported row inside the BHist carrier data.

Finally apply Definition 149.56 and Definition 149.68. These results attach the Cont, hsame, and Pkg rows to the same source projection and exclude any source beyond the consumed $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ rows. □

Theorem 149.48 (PinGroup parity transport ledger exhaustion)**.** *The finite transport ledger visible to the* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ *reflection-parity surface is exhausted by the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *inclusion row, the* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *Clifford reflection product row, the parity classifier row, the source-projection rows, and the consumer rows exported by the displayed* BHist *packet. Its public witnesses are exactly the listed* hsame, Cont, *and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface exhaustion`

Proof. Start from Definition 149.12. It places each public source read in the Spin or Clifford source ledger and records the same BHist histories for the carrier-facing packet.

Apply Definition 149.73 to identify the Spin inclusion row, and apply Definition 149.58 to identify the Clifford reflection product and action rows.

Next use Definition 149.57 and Definition 149.56. They fix the parity classifier and source projections as the only classifier-facing source rows.

Finally apply Definition 149.65. It exports the same consumer rows with their hsame, Cont, and Pkg witnesses, so no further transport ledger remains. □

Theorem 149.49 (PinGroup parity NameCert obligation surface)**.** *The* $\mathsf{PinGroup}^\uparrow$ *obligation-Closure surface is the finite* BHist *package obtained from the parity carrier inversion of Definition 149.59, the transport-ledger exhaustion of Definition 149.60, the source ledger coverage of Definition 149.12, and the parity exactness of Definition 149.18. It consists exactly of the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *rows, the* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$ *reflection rows, the parity classifier, the displayed* hsame *and* Cont *transport rows, the* Pkg *provenance rows, and the exported consumer rows of* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$.

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier namecert obligation surface`

Proof. First use Definition 149.59. It gives the row-wise inversion data and restricts every accepted parity-carrier row to the Spin or Clifford source projections.

Then use Definition 149.60. This fixes the finite ledger as the Spin inclusion, Clifford reflection product, parity classifier, source projections, and exported consumer rows.

Combine those two statements with Definition 149.12 and Definition 149.18. The source theorem fixes the two certificate inputs, while the exactness theorem fixes the branch classifier and its transport behavior.

Apply Definition 149.65 to repack the same BHist rows as the public $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ consumer surface. The resulting package is the full obligation surface and contains no host orthogonal group, quotient algebra, matrix model, or third source. □

Theorem 149.50 (PinGroup parity source scope package)**.** *The reflection-parity source rows of Definition 149.16, Definition 149.12, and Definition 149.67 are scoped exactly by the finite* BHist *carrier rows, the displayed* hsame *classifier rows, the finite* Cont *transport rows, the* Pkg *provenance rows, and the source certificate rows* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Proof. Project the carrier through Definition 149.16. Its two branches are the inherited $\mathsf{SpinGroup}^{\uparrow}$ branch and the Clifford reflection branch.

Apply Definition 149.12. The source ledger names only $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$, their carried BHist histories, and their Cont provenance.

Finally use Definition 149.67. The parity classifier reads those two branches through hsame and Pkg rows, with no host orthogonal group and no matrix model available behind the source scope. □

Theorem 149.51 (PinGroup reflection ledger scope package)**.** *The reflection ledger rows of Definition 149.8, Definition 149.20, and Definition 149.66 are scoped exactly by the carried* BHist *Clifford reflection entries, the finite* Cont *ledger, the displayed* hsame *classifier rows,* Pkg *provenance, and the source certificate rows* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Proof. Project the reflection ledger from Definition 149.8. It consists of the spin subrow, the reflection-generator row, the Clifford product row, and the finite provenance ledger.

Apply Definition 149.20. Transport keeps every reflection entry inside the same carried BHist ledger over $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$, with the inherited spin boundary read through $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$.

The exactness theorem Definition 149.66 identifies the public action row with this same ledger boundary. Hence the scoped package reads no quotient Clifford unit, no host full orthogonal group, and no external matrix representation. □

Theorem 149.52 (PinGroup consumer boundary scope package)**.** *The consumer rows of Definition 149.62, Definition 149.63, Definition 149.64, and Definition 149.65 are scoped exactly by the reflection ledger, Clifford action row, Spin boundary row, source projection rows,* BHist *carrier rows,* hsame, Cont, Pkg, *and the source certificates* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Proof. Project a consumer row through Definition 149.62. The packet retains precisely the reflection ledger, Clifford action row, Spin boundary row, source rows, transport rows, witnesses, and provenance.

Apply Definition 149.63 and Definition 149.64. Soundness projects an accepted consumer into that finite packet, and exactness repacks the same rows as the public Clifford-action consumer surface.

Finally apply Definition 149.65. The exported consumer package is exactly the ten-field packet over the displayed BEDC primitive rows, so no host orthogonal group, quotient Clifford unit, or matrix model is part of the consumer boundary. □

Theorem 149.53 (PinGroup scoped reflection-parity source)**.** *The scoped reflection-parity source of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the finite carrier surface consisting of the reflection-parity carrier of Definition 149.16, the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row, the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row, the displayed* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance rows supplied by* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$*.*

Every accepted consumer read of the parity source factors through those rows, and any packet with those rows repacks as the same scoped $\mathsf{PinGroup}^{\uparrow}$ *reflection-parity source. No host orthogonal group, quotient Clifford unit, matrix representation, or third source row is part of the scoped source.*

Proof. Start with Definition 149.83. It identifies the source rows as the inherited Spin branch, the Clifford reflection branch, the carried BHist histories, and the displayed transport and provenance rows.

Then apply Definition 149.59. The inversion record attaches each accepted parity-carrier row to exactly one of the two source projections and carries the same hsame, Cont, and Pkg data.

Finally use Definition 149.61. It repackages the listed rows as the scoped $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ obligation surface and excludes host orthogonal, matrix, quotient, and third-source data. □

Theorem 149.54 (PinGroup scoped Spin-Clifford boundary)**.** *The scoped Spin-Clifford boundary of* $\mathsf{PinGroup}^{\uparrow}$ *is the common finite boundary between the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row and the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row. It is determined by the Spin boundary row, the Clifford reflection product row, the reflection-parity classifier row, the* hsame *endpoint transports, the finite* Cont *ledger, and the shared* Pkg *provenance.*

Two scoped boundary packets determine the same public boundary exactly when those rows agree. The boundary exports no independent Pin carrier datum beyond the reflection-parity carrier and no host full-orthogonal action beyond the displayed Clifford action row.

Proof. First read the inherited Spin side through Definition 149.73. It shows that a Spin source row contributes only the inherited classifier and the identity reflection ledger.

Next read the Clifford side through Definition 149.84. The Clifford product and reflection ledger are exactly the carried reflection entries, the finite transport ledger, and the source certificate rows.

Finally combine these rows using Definition 149.60. The exhaustion statement fixes the shared parity classifier, endpoint transports, ledger routes, and provenance, so agreement on those rows is precisely agreement of the scoped Spin-Clifford boundary. □

Theorem 149.55 (PinGroup scoped consumer projection exhaustion)**.** *Every scoped consumer projection from* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the consumer readback packet of Definition 149.62, the reflection-parity carrier, the Spin inclusion row, the Clifford reflection product row, the Clifford action row, the* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance rows. Conversely, any finite packet carrying exactly those rows is the public scoped consumer surface for later* $\mathsf{PinGroup}^{\uparrow}$ *consumers.*

Proof. Project the consumer through Definition 149.85. It records precisely the reflection ledger, Clifford action row, Spin boundary row, source projections, transport rows, witnesses, provenance, and source certificates.

Then apply Definition 149.65. The downstream package exports the same finite consumer rows and does not import a host orthogonal group, quotient Clifford unit, or matrix model.

Finally use Definition 149.78. It identifies the root-facing consumer threshold packet with the same scoped rows, so every later consumer projection is exhausted by the displayed readback packet and every such packet repacks as the scoped public surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$.

Constructive story: A pin-group row is generated by taking the already certified $\mathsf{SpinGroup}^\uparrow$ carrier inside the already certified $\mathsf{Clifford}^\uparrow$ surface and adjoining one finite reflection-generator ledger; a history is admitted only when it factors through the inherited Spin branch or through the displayed Clifford reflection product branch, and a third parity source is not accepted. Two admitted rows are classified by the componentwise hsame and Cont transport already visible in those source certificates: the Spin subrow, the reflection generator, the Clifford product, and the provenance package must all move through the same finite ledger rather than through equality in a host orthogonal group. The source specification is therefore the pair of consumed Spin and Clifford certificate rows, the pattern specification is the two-branch reflection-parity display with its finite product ledger, and the ledger policy keeps the raw Spin inclusion, Clifford reflection product, source projection, transported endpoints, continuation rows, and Pkg provenance visible to downstream consumers. The five name-certificate obligations close branch by branch: an inherited Spin row or a displayed reflection product gives the carrier witness, hsame supplies reflexivity, symmetry, and transitivity on visible endpoints, and the reflection-parity stability theorem transports either branch back into the same two-source carrier. A history h lies in the signature gap of a $\mathsf{PinGroup}^\uparrow$ token p when the source-domain accepts h as one of these parity rows, the signature relation pairs it with an answer parity row and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is exactly the Spin branch or Clifford reflection branch encoded by p, together with the required Cont, hsame, and Pkg rows. The chapter registers this reflection-parity obligation surface under the name $\mathsf{PinGroup}^\uparrow$; it introduces no host orthogonal equality, quotient Clifford algebra, matrix model, standard bridge, or source beyond the displayed $\mathsf{SpinGroup}^\uparrow$ and $\mathsf{Clifford}^\uparrow$ certificates.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$, maturePackageClosed)

Scope closed: The scoped surface binds the reflection-parity carrier, SpinGroup inclusion, Clifford reflection product, source projection, consumer readback packet, consumer boundary exhaustion, BHist, Cont, hsame, and Pkg rows to $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier\_stability`

Bridge status: bridgeChecked

Not claimed: The full orthogonal group, analytic covering theory, matrix models, quotient algebras, and standard bridge data are outside this chapter.

Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 149.56 (PinGroup source projection obligation)**.** *The* $\mathsf{PinGroup}^\uparrow$ *source projection surface is exactly the pair consisting of the consumed* $\mathsf{SpinGroup}^\uparrow$ *source row and the consumed* $\mathsf{Clifford}^\uparrow$ *source row, transported by the displayed* hsame *witnesses and finite* Cont *ledger of* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$. *No public source projection is available unless it factors through one of those two rows and the corresponding* Pkg *provenance entry.*

Proof. Begin with Definition 149.12. It places every public source read in the Spin or Clifford source ledger and records the same BHist histories in the carrier-facing certificate packet.

Apply Definition 149.18. The inherited Spin branch is the Spin projection, while the reflective branch is read through the Clifford source together with the reflection-generator product ledger.

Now use Definition 149.19. It rules out a third source projection, and the downstream package of Definition 149.65 repacks the two source rows with exactly their hsame, Cont, and Pkg witnesses. □

Theorem 149.57 (PinGroup parity classifier obligation)**.** *The reflection-parity carrier of Definition 149.16 is the complete public classifier for* $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$*: every admitted public row is classified either by the inherited* $\mathsf{SpinGroup}^\uparrow$ *branch or by the* $\mathsf{Clifford}^\uparrow$ *reflection branch, and the classifier is exact under the carried* hsame *transport rows.*

Proof. Project a public row through Definition 149.67. The row is read through one branch of the reflection-parity carrier, except for the Spin overlap already recorded by the inherited Spin branch.

Apply Definition 149.18. It identifies the Spin branch with rows factoring through the Spin-inclusion source and the reflective branch with rows factoring through the Clifford reflection-generator source and product ledger.

Finally use Definition 149.19 and Definition 149.65. The first excludes any outside classifier source, and the second exports the same parity carrier, transport witnesses, continuation ledger, and provenance rows to downstream consumers. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier parity classifier obligation`

Theorem 149.58 (PinGroup Clifford action ledger obligation)**.** *The Clifford action visible at the* $\mathsf{PinGroup}^{\uparrow}$ *obligation surface is exhausted by the reflection ledger, Clifford action row, inherited Spin boundary, finite* Cont *ledger,* hsame *witnesses, and* Pkg *provenance rows displayed by* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*. The obligation surface exports no host orthogonal action, matrix action, quotient algebra, or external Pin object beyond those* BHist *rows.*

Proof. Start with Definition 149.22. It lists the action rows a consumer may inspect: reflection ledger, Clifford action, Spin boundary, provenance, continuation transport, and hsame witnesses.

Apply Definition 149.20 to anchor the reflection part in carried Clifford rows, and apply Definition 149.21 to anchor the Spin part in the consumed Spin source and even-grade Clifford rows.

Use Definition 149.19 to exclude any source outside the Spin and Clifford rows. Then Definition 149.65 repacks the same action, ledger, transport, and provenance rows as the downstream obligation surface. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface clifford action ledger obligation`

Theorem 149.59 (PinGroup parity carrier inversion)**.** *Every accepted row of the reflection-parity carrier of Definition 149.16 has a finite inversion record inside the public* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface. The record is a* BHist *row whose source projection is either the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *Spin row or the* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *Clifford reflection row, together with the public parity classifier, the displayed* Cont *transport ledger, the matching* hsame *witness, and the* Pkg *provenance row. No accepted parity-carrier row has a third source projection.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier parity carrier inversion`

Proof. Begin with Definition 149.67. It factors every public $\mathsf{PinGroup}^{\uparrow}$ row through the inherited Spin branch or the Clifford reflection branch of the displayed parity carrier.

Use Definition 149.18. The exactness theorem identifies the accepted branch with the public parity classifier and keeps the transported row inside the BHist carrier data.

Finally apply Definition 149.56 and Definition 149.68. These results attach the Cont, hsame, and Pkg rows to the same source projection and exclude any source beyond the consumed $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ rows. □

Theorem 149.60 (PinGroup parity transport ledger exhaustion)**.** *The finite transport ledger visible to the* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *reflection-parity surface is exhausted by the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *inclusion row, the* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *Clifford reflection product row, the parity classifier row, the source-projection rows, and the consumer rows exported by the displayed* BHist *packet. Its public witnesses are exactly the listed* hsame, Cont, *and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface exhaustion`

Proof. Start from Definition 149.12. It places each public source read in the Spin or Clifford source ledger and records the same BHist histories for the carrier-facing packet.

Apply Definition 149.73 to identify the Spin inclusion row, and apply Definition 149.58 to identify the Clifford reflection product and action rows.

Next use Definition 149.57 and Definition 149.56. They fix the parity classifier and source projections as the only classifier-facing source rows.

Finally apply Definition 149.65. It exports the same consumer rows with their hsame, Cont, and Pkg witnesses, so no further transport ledger remains. □

Theorem 149.61 (PinGroup parity NameCert obligation surface)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *obligation-Closure surface is the finite* BHist *package obtained from the parity carrier inversion of Definition 149.59, the transport-ledger exhaustion of Definition 149.60, the source ledger coverage of Definition 149.12, and the parity exactness of Definition 149.18. It consists exactly of the inherited* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *rows, the* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$ *reflection rows, the parity classifier, the displayed* hsame *and* Cont *transport rows, the* Pkg *provenance rows, and the exported consumer rows of* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier namecert obligation surface`

Proof. First use Definition 149.59. It gives the row-wise inversion data and restricts every accepted parity-carrier row to the Spin or Clifford source projections.

Then use Definition 149.60. This fixes the finite ledger as the Spin inclusion, Clifford reflection product, parity classifier, source projections, and exported consumer rows.

Combine those two statements with Definition 149.12 and Definition 149.18. The source theorem fixes the two certificate inputs, while the exactness theorem fixes the branch classifier and its transport behavior.

Apply Definition 149.65 to repack the same BHist rows as the public $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ consumer surface. The resulting package is the full obligation surface and contains no host orthogonal group, quotient algebra, matrix model, or third source. □

Definition 149.62 (PinGroup consumer readback packet)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet is the finite public packet obtained from* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *by retaining exactly the reflection ledger, the Clifford action row, the Spin boundary row, the* Pkg *provenance rows, the finite* Cont *transport rows, and the* hsame *witnesses carried by the public certificate. It contains no host orthogonal group, no external Pin object, and no row not exported by the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface.*

Theorem 149.63 (PinGroup consumer readback packet soundness)**.** *Every accepted public* $\mathsf{PinGroup}^{\uparrow}$ *consumer row projects to the* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet of Definition 149.62.*

Proof. Apply Definition 149.15. It says that the accepted public surface of $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ is carried by the finite carrier, classifier, source, transport, witness, and provenance rows listed in the certificate.

The reflection component is then covered by Definition 149.20. That theorem identifies every reflection read with the displayed reflection ledger and its finite BHist provenance.

The Spin component is covered by Definition 149.21. Hence the remaining public action read is the Clifford action row already exposed by the certificate surface, and all transport or identity data are the displayed Cont and hsame rows. These are exactly the rows retained in the readback packet. □

Theorem 149.64 (PinGroup consumer readback packet exactness)**.** *Any* $\mathsf{PinGroup}^{\uparrow}$ *consumer readback packet reconstructs the public* $\mathsf{PinGroup}^{\uparrow}$ *Clifford-action consumer surface and exposes no host orthogonal group or external Pin object.*

Proof. Start with a packet as in Definition 149.62. Its reflection ledger, Clifford action row, Spin boundary, Pkg provenance, finite Cont transport, and hsame witnesses are precisely the rows admitted by the public consumer surface of Definition 149.22.

Apply Definition 149.19. The only sources available to the public Pin certificate are the displayed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ source rows together with their transported histories.

Therefore repacking the packet rows reconstructs the same consumer surface, while the source-exhaustion and consumer-surface theorems exclude any host orthogonal group or external Pin object outside those rows. □

Theorem 149.65 (PinGroup downstream unblock package)**.** *The carrier definition of* $\mathsf{PinGroup}^{\uparrow}$ *together with Definition 149.62, Definition 149.63, and Definition 149.64 forms the ten-field downstream package consumed by later Clifford-family and Spin-family chapters: carrier, classifier, reflection ledger, Clifford action row, Spin boundary row, source exhaustion,* Pkg *provenance, finite* Cont *transport,* hsame *witnesses, and exclusion of host orthogonal or external Pin reads.*

Proof. The carrier and classifier are the public $\mathsf{PinGroup}^{\uparrow}$ rows already fixed by Definition 149.15. The reflection ledger, Clifford action row, and Spin boundary are exactly the three rows retained in Definition 149.62.

Soundness is supplied by Definition 149.63; exact readback and exclusion of host data are supplied by Definition 149.64. The remaining provenance, transport, and witness fields are the packet's Pkg, Cont, and hsame rows.

Thus a later Clifford-family or Spin-family consumer may cite this single package instead of separately collecting the carrier, source, ledger, transport, and exclusion rows. □

Theorem 149.66 (PinGroup full-orthogonal action exactness)**.** *The public full-orthogonal action row of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the classifier boundary assembled from the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *source, the Clifford reflection generator, the Clifford product ledger,* hsame*,* Cont*, and* Pkg *rows of the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*. It admits no additional host orthogonal action row outside that boundary.*

Proof. Apply Definition 149.12 to locate the inherited $\mathsf{SpinGroup}^{\uparrow}$ source and the Clifford source inside the finite source ledger. Apply Definition 149.18 to identify the classifier boundary with the parity carrier induced by those two source branches.

Apply Definition 149.19. The only public projections remaining after the carrier read are the listed $\mathsf{SpinGroup}^{\uparrow}$ and $\mathsf{Clifford}^{\uparrow}$ rows, their Cont transports, their hsame comparisons, and their Pkg provenance.

Finally apply Definition 149.65. Its consumer package records the reflection ledger, Clifford action row, Spin boundary row, source exhaustion, transport, witnesses, and provenance as one exported surface, so the full-orthogonal action row is exactly that classifier boundary. □

Theorem 149.67 (PinGroup reflection-parity boundary)**.** *Every public* $\mathsf{PinGroup}^{\uparrow}$ *row factors through exactly one branch of Definition 149.16: either the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *branch or the Clifford reflection branch. No public row factors through both branches except by the inherited Spin classification already recorded in the carrier.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier source projection no confusion`

Proof. Use Definition 149.12 to read the public source ledger as the two displayed sources and their transported histories. Apply Definition 149.18 to classify each public row by the reflection-parity carrier.

For a row classified by the inherited Spin branch, the carrier records exactly the $\mathsf{SpinGroup}^{\uparrow}$ inclusion and its hsame, Cont, and Pkg rows. For a row classified by the Clifford reflection branch, the carrier records the reflection generator and Clifford product ledger.

Apply Definition 149.19 to rule out a third branch. Then apply Definition 149.65 to see that the consumer-visible package exports precisely these two branches and the Spin overlap already classified by the inherited branch. □

Theorem 149.68 (PinGroup source projection separation)**.** *The public source projections of* $\mathsf{PinGroup}^{\uparrow}$ *to* $\mathsf{SpinGroup}^{\uparrow}$ *and* $\mathsf{Clifford}^{\uparrow}$ *are disjoint except for rows classified by the inherited Spin branch of Definition 149.16. Consequently the public certificate has no third source projection.*

Proof. Apply Definition 149.12. It places every public source read in the displayed $\mathsf{SpinGroup}^{\uparrow}$ or $\mathsf{Clifford}^{\uparrow}$ source row, with the advertised BHist histories and no additional projection field.

Apply Definition 149.18. The classifier separates inherited Spin rows from Clifford reflection rows, and it identifies the only common public classification as the inherited Spin branch already present in the reflection-parity carrier.

Apply Definition 149.19 to exclude any remaining public source projection. The downstream package of Definition 149.65 then repacks exactly the two source projections, their transports, witnesses, and provenance for later consumers. □

Definition 149.69 (PinGroup scoped primitive binding)**.** *The scoped primitive binding for* $\mathsf{PinGroup}^{\uparrow}$ *is the finite dependency package whose only primitive rows are* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$. *The inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row and the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row are read only through those two source certificates, and every reflection-parity carrier, classifier, transport, and provenance row of this chapter is a row in that package.*

Theorem 149.70 (PinGroup parity scope transport)**.** *Every reflection-parity carrier transport used by* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the primitive package of Definition 149.69. In particular, parity transport may use only the carried* BHist *histories, displayed* hsame *comparisons, finite* Cont *routes,* Pkg *provenance, and the two source certificate rows from* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface parity scope transport`

Proof. Use Definition 149.83 to identify the parity rows as the inherited Spin branch, Clifford reflection branch, transport rows, and provenance rows.

Then apply Definition 149.86. It repacks the same rows as the scoped reflection-parity source and excludes host orthogonal, matrix, quotient, and third-source data. Hence every carrier transport factors through exactly the primitive binding package. □

Theorem 149.71 (PinGroup consumer boundary exhaustion)**.** *Every downstream consumer of the scoped* $\mathsf{PinGroup}^{\uparrow}$ *surface receives only the two source projections, the reflection-parity classifier, the transport rows, and the provenance package. No downstream consumer receives a host orthogonal group, matrix model, quotient Clifford unit, or additional Pin source row.*

Proof. Project the consumer through Definition 149.88. This gives the consumer readback packet, reflection-parity classifier, Spin inclusion row, Clifford reflection product row, transport rows, finite continuation routes, and provenance rows.

Finally apply Definition 149.65. It identifies the downstream package with the same finite rows, so the exported consumer boundary has exactly the two source projections, classifier, transport, and provenance data stated above. □

Theorem 149.72 (PinGroup reflection-generator transport closure)**.** *The* hsame *transport of the Clifford reflection-generator row preserves* $\mathsf{PinGroup}^{\uparrow}$ *carrier admission exactly through the finite* Cont *ledger of the displayed* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*; no transported reflection-generator row is admitted without the carrier row, the reflection-generator obligation, and the closed reflection ledger.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface reflection generator transport closure`

Proof. Project the carrier admission row of Definition 149.1. It exposes the Clifford reflection-generator source, its hsame witness, and its finite Cont transport as the only carrier-facing generator data.

Apply Definition 149.3 to identify the accepted generator row, then apply Definition 149.20. The ledger closure repacks exactly the transported row and excludes any host generator outside the displayed Cont ledger. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionGenerator transport closure`

Theorem 149.73 (PinGroup Spin subrow classifier readback)**.** *Every* $\mathsf{SpinGroup}^{\uparrow}$ *source row embedded in the* $\mathsf{PinGroup}^{\uparrow}$ *carrier is read by the inherited Spin classifier together with the identity reflection ledger. The embedded row therefore contributes no independent Clifford reflection generator beyond the Spin boundary already carried by* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*.*

Proof. Apply Definition 149.4. It places the $\mathsf{SpinGroup}^{\uparrow}$ source row inside the carrier with the inherited classifier, the identity reflection entry, hsame, Cont, and Pkg provenance.

Apply Definition 149.21. The exhaustion theorem rules out a second reflection-generator read on the Spin subrow, so the classifier readback is precisely the inherited Spin row plus the identity reflection ledger. □

Theorem 149.74 (PinGroup orthogonal projection ledger exhaustion)**.** *Every public orthogonal-action projection row of* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the inherited Spin source, the Clifford action row, the reflection-generator parity row, the listed* hsame *transports, the finite* Cont *ledger, and the* Pkg *provenance row of* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*.*

Proof. Apply Definition 149.5 to reduce a public orthogonal-action read to the certificate projection ledger. Apply Definition 149.12 to identify its Spin source, Clifford action, and reflection-generator parity components.

Finally apply Definition 149.22. It exports exactly those source rows with their hsame, Cont, and Pkg data, so no further public orthogonal-action projection remains. □

Definition 149.75 (PinGroup root consumer threshold packet)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is the finite public packet whose rows are the* $\mathsf{SpinGroup}^{\uparrow}$ *source row, the* $\mathsf{Clifford}^{\uparrow}$ *source row, the reflection-parity row of Definition 149.16, the orthogonal-action projection row, the* hsame *transport rows, the finite* Cont *ledger rows, and the* Pkg *provenance rows displayed by* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$*. A downstream root consumer reads only this packet when it asks for the Pin reflection boundary.*

Theorem 149.76 (PinGroup root source-threshold coverage)**.** *Every source row visible to the* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is covered by the consumed* $\mathsf{SpinGroup}^{\uparrow}$ *source, the consumed* $\mathsf{Clifford}^{\uparrow}$ *source, and the reflection-parity split of Definition 149.16. No third source projection is visible at the threshold.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupSpinSubrowClassifier readback`

Proof. Use Definition 149.12. It places every public source read in the Spin or Clifford source ledger and records the same BHist histories in the finite certificate packet.

Apply Definition 149.73 to read the Spin subrow through the inherited classifier, and apply Definition 149.18 to separate the Clifford reflection branch from the inherited Spin branch.

Finally apply Definition 149.19. The exhaustion result rules out any source outside those two rows, so the threshold packet has exactly the advertised source coverage. □

Theorem 149.77 (PinGroup root reflection-threshold exactness)**.** *The reflection part of the* $\mathsf{PinGroup}^{\uparrow}$ *root consumer threshold packet is accepted exactly when the Clifford reflection-generator obligation, the reflection-parity classifier, the orthogonal-action projection, and the finite* hsame *and* Cont *transport rows of* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *are present. It exports no host reflection, matrix reflection, or orthogonal-group element.*

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityCarrier root reflection threshold exactness`

Proof. Project the reflection generator through Definition 149.72. This identifies the admitted reflection row with the carrier row, the generator obligation, and the closed finite reflection ledger.

Apply Definition 149.18. The classifier accepts precisely the inherited Spin branch or the Clifford reflection branch, and the reflection threshold is the latter branch with its displayed parity row.

Now apply Definition 149.74. The orthogonal-action readback is exhausted by the Spin source, Clifford action, parity row, hsame transports, Cont ledger, and Pkg provenance already listed in the threshold packet. Hence no host reflection object is exported. □

Theorem 149.78 (PinGroup root downstream threshold surface)**.** *The downstream root-facing surface of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the consumer threshold packet of Definition 149.75. It supplies the Spin source row, Clifford source row, reflection-parity row, orthogonal projection row,* hsame *transport rows,* Cont *ledger rows, and* Pkg *provenance rows needed by Spin-facing and Pin-facing root consumers, and it supplies no analytic covering, quotient algebra, matrix model, or host orthogonal-group surface.*

Proof. First use Definition 149.76. It fixes the only source rows available to the threshold packet and keeps them inside the Spin and Clifford source ledgers.

Next use Definition 149.77. It fixes the reflection branch by the generator obligation, parity classifier, orthogonal projection, and finite transport ledgers.

Finally apply Definition 149.65 and Definition 149.22. These results repack the same source, reflection, action, hsame, Cont, and Pkg rows for downstream consumers. The packet is therefore the full root threshold surface, with all host analytic and algebraic models excluded. □

Theorem 149.79 (PinGroup root carrier obligation scope)**.** *The carrier rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the* $\mathsf{SpinGroup}^{\uparrow}$ *source row, the* $\mathsf{Clifford}^{\uparrow}$ *source*

row, the reflection-parity carrier row, and the orthogonal-action projection row displayed by Definition 149.75. No matrix model, analytic cover, quotient algebra, or host orthogonal-group carrier is part of the obligation scope.

Proof. Begin with Definition 149.75. Its public packet lists the Spin source, Clifford source, reflection-parity carrier, and orthogonal-action projection rows as the carrier-facing data.

Use Definition 149.76. It proves that every source row visible at the threshold is covered by the Spin and Clifford source rows together with the reflection-parity split.

Apply Definition 149.77 and Definition 149.78. These results place the reflection and projection rows inside the same finite threshold packet and exclude all host carrier models from the downstream surface. □

Theorem 149.80 (PinGroup root classifier obligation scope)**.** *The classifier rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *classifier on Spin subrows and the reflection-parity classifier separating the inherited Spin branch from the Clifford reflection branch. No independent Clifford classifier, matrix parity classifier, or quotient classifier is exported.*

Proof. Read the inherited Spin classifier from Definition 149.73. A Spin source row embedded in the Pin carrier is classified by the inherited Spin row together with the identity reflection ledger.

Read the reflection branch from Definition 149.77. It uses Definition 149.18 to identify the accepted reflection side with the reflection-parity classifier and the Clifford reflection-generator obligation.

Finally use Definition 149.78. The downstream threshold surface repacks exactly those classifier rows, so there is no remaining classifier obligation outside the inherited Spin and reflection-parity rows. □

Theorem 149.81 (PinGroup root ledger obligation scope)**.** *The ledger rows required for the* $\mathsf{PinGroup}^{\uparrow}$ *root seed-to-obligation surface are exactly the listed* hsame *transports, finite* Cont *rows,* Pkg *provenance rows, and the orthogonal projection ledger exhausted by Definition 149.74. These rows are sufficient for the downstream root surface and supply no analytic covering ledger, quotient-algebra ledger, matrix ledger, or host orthogonal-group ledger.*

Proof. Start from Definition 149.75. The packet lists the hsame transport rows, finite Cont ledger rows, and Pkg provenance rows read by the root consumer.

Apply Definition 149.74. It exhausts the public orthogonal-action projection by the inherited Spin source, Clifford action row, reflection-generator parity row, listed transports, finite Cont ledger, and Pkg provenance.

Use Definition 149.76 and Definition 149.77 to identify the source and reflection parts of the same ledger. Then Definition 149.78 repacks those rows as the downstream root threshold surface, excluding any host analytic, quotient, matrix, or orthogonal-group ledger. □

Theorem 149.82 (PinGroup root scoped obligation package)**.** *The reflection-generator transport, Spin subrow classifier readback, orthogonal projection ledger exhaustion, and the three root scope theorems jointly form the scoped* $\mathsf{PinGroup}^{\uparrow}$ *obligation package over* BHist, hsame, Cont, $\mathsf{NameCert}$, *and the displayed* $\mathsf{Clifford}^{\uparrow}$ *and* $\mathsf{SpinGroup}^{\uparrow}$ *source rows. The package binds the root carrier, classifier, and ledger obligations to the finite threshold packet of Definition 149.75 and exports no analytic covering, quotient algebra, matrix model, or host orthogonal-group surface.*

Proof. First read the root threshold packet. By Definition 149.72, Definition 149.73, and Definition 149.74, every reflection-generator, inherited Spin classifier, and orthogonal projection row is projected through the same finite BHist packet with its hsame, Cont, and Pkg ledgers.

Next bind the three obligation faces. Apply Definition 149.79, Definition 149.80, and Definition 149.81. These results identify the carrier rows, classifier rows, and ledger rows that can appear at the root threshold.

Finally project and repack the finite packet. The first three theorems give the available source and transport projections, while the three scope theorems say that no row outside those projections has an obligation slot. Repacking them through the displayed $\mathsf{NameCert}_{\mathsf{PinGroup}^\uparrow}$ therefore produces the scoped root package over the listed BEDC kernel objects and the Clifford/SpinGroup sources alone. □

Lean4 ✓ `BEDC.Derived.PinGroupUp.PinGroupReflectionParityLedgerSurface root scoped obligation package`

Theorem 149.83 (PinGroup parity source scope package)**.** *The reflection-parity source rows of Definition 149.16, Definition 149.12, and Definition 149.67 are scoped exactly by the finite* BHist *carrier rows, the displayed* hsame *classifier rows, the finite* Cont *transport rows, the* Pkg *provenance rows, and the source certificate rows* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$.

Proof. Project the carrier through Definition 149.16. Its two branches are the inherited $\mathsf{SpinGroup}^\uparrow$ branch and the Clifford reflection branch.

Apply Definition 149.12. The source ledger names only $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$, their carried BHist histories, and their Cont provenance.

Finally use Definition 149.67. The parity classifier reads those two branches through hsame and Pkg rows, with no host orthogonal group and no matrix model available behind the source scope. □

Theorem 149.84 (PinGroup reflection ledger scope package)**.** *The reflection ledger rows of Definition 149.8, Definition 149.20, and Definition 149.66 are scoped exactly by the carried* BHist *Clifford reflection entries, the finite* Cont *ledger, the displayed* hsame *classifier rows,* Pkg *provenance, and the source certificate rows* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$.

Proof. Project the reflection ledger from Definition 149.8. It consists of the spin subrow, the reflection-generator row, the Clifford product row, and the finite provenance ledger.

Apply Definition 149.20. Transport keeps every reflection entry inside the same carried BHist ledger over $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$, with the inherited spin boundary read through $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$.

The exactness theorem Definition 149.66 identifies the public action row with this same ledger boundary. Hence the scoped package reads no quotient Clifford unit, no host full orthogonal group, and no external matrix representation. □

Theorem 149.85 (PinGroup consumer boundary scope package)**.** *The consumer rows of Definition 149.62, Definition 149.63, Definition 149.64, and Definition 149.65 are scoped exactly by the reflection ledger, Clifford action row, Spin boundary row, source projection rows,* BHist *carrier rows,* hsame*,* Cont*,* Pkg*, and the source certificates* $\mathsf{NameCert}_{\mathsf{SpinGroup}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^\uparrow}$.

Proof. Project a consumer row through Definition 149.62. The packet retains precisely the reflection ledger, Clifford action row, Spin boundary row, source rows, transport rows, witnesses, and provenance.

Apply Definition 149.63 and Definition 149.64. Soundness projects an accepted consumer into that finite packet, and exactness repacks the same rows as the public Clifford-action consumer surface.

Finally apply Definition 149.65. The exported consumer package is exactly the ten-field packet over the displayed BEDC primitive rows, so no host orthogonal group, quotient Clifford unit, or matrix model is part of the consumer boundary. □

Theorem 149.86 (PinGroup scoped reflection-parity source)**.** *The scoped reflection-parity source of* $\mathsf{PinGroup}^{\uparrow}$ *is exactly the finite carrier surface consisting of the reflection-parity carrier of Definition 149.16, the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row, the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row, the displayed* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance rows supplied by* $\mathsf{NameCert}_{\mathsf{SpinGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Clifford}^{\uparrow}}$.

Every accepted consumer read of the parity source factors through those rows, and any packet with those rows repacks as the same scoped $\mathsf{PinGroup}^{\uparrow}$ *reflection-parity source. No host orthogonal group, quotient Clifford unit, matrix representation, or third source row is part of the scoped source.*

Proof. Start with Definition 149.83. It identifies the source rows as the inherited Spin branch, the Clifford reflection branch, the carried BHist histories, and the displayed transport and provenance rows.

Then apply Definition 149.59. The inversion record attaches each accepted parity-carrier row to exactly one of the two source projections and carries the same hsame, Cont, and Pkg data.

Finally use Definition 149.61. It repackages the listed rows as the scoped $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ obligation surface and excludes host orthogonal, matrix, quotient, and third-source data. □

Theorem 149.87 (PinGroup scoped Spin-Clifford boundary)**.** *The scoped Spin-Clifford boundary of* $\mathsf{PinGroup}^{\uparrow}$ *is the common finite boundary between the inherited* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row and the* $\mathsf{Clifford}^{\uparrow}$ *reflection product row. It is determined by the Spin boundary row, the Clifford reflection product row, the reflection-parity classifier row, the* hsame *endpoint transports, the finite* Cont *ledger, and the shared* Pkg *provenance.*

Two scoped boundary packets determine the same public boundary exactly when those rows agree. The boundary exports no independent Pin carrier datum beyond the reflection-parity carrier and no host full-orthogonal action beyond the displayed Clifford action row.

Proof. First read the inherited Spin side through Definition 149.73. It shows that a Spin source row contributes only the inherited classifier and the identity reflection ledger.

Next read the Clifford side through Definition 149.84. The Clifford product and reflection ledger are exactly the carried reflection entries, the finite transport ledger, and the source certificate rows.

Finally combine these rows using Definition 149.60. The exhaustion statement fixes the shared parity classifier, endpoint transports, ledger routes, and provenance, so agreement on those rows is precisely agreement of the scoped Spin-Clifford boundary. □

Theorem 149.88 (PinGroup scoped consumer projection exhaustion)**.** *Every scoped consumer projection from* $\mathsf{PinGroup}^{\uparrow}$ *is exhausted by the consumer readback packet of Definition 149.62, the reflection-parity carrier, the Spin inclusion row, the Clifford reflection product row, the Clifford action row, the* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance rows. Conversely, any finite packet carrying exactly those rows is the public scoped consumer surface for later* $\mathsf{PinGroup}^{\uparrow}$ *consumers.*

Proof. Project the consumer through Definition 149.85. It records precisely the reflection ledger, Clifford action row, Spin boundary row, source projections, transport rows, witnesses, provenance, and source certificates.

Then apply Definition 149.65. The downstream package exports the same finite consumer rows and does not import a host orthogonal group, quotient Clifford unit, or matrix model.

Finally use Definition 149.78. It identifies the root-facing consumer threshold packet with the same scoped rows, so every later consumer projection is exhausted by the displayed readback packet and every such packet repacks as the scoped public surface. □

Theorem 149.89 (PinGroup Spin-Clifford sibling lattice)**.** *Every public* $\mathsf{PinGroup}^{\uparrow}$ *reflection-parity read factors through exactly the consumed* $\mathsf{SpinGroup}^{\uparrow}$ *inclusion row of chapter 148 and the consumed* $\mathsf{Clifford}^{\uparrow}$ *reflection-product and action rows of chapter 147. The factorization is recorded by the displayed* BHist *carrier rows, componentwise* hsame *transports, finite* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *support row.*

No public reflection-parity consumer may replace this lattice by a host orthogonal group, a quotient Clifford algebra, a matrix representation, or an independent Pin carrier. Conversely, a finite packet carrying exactly the Spin inclusion row, the Clifford reflection-product and action rows, and the listed BEDC support rows is the same public $\mathsf{PinGroup}^{\uparrow}$ *reflection-parity surface.*

Proof. First read the source of the public parity surface through Definition 149.86. It identifies the visible source rows as the reflection-parity carrier, the inherited Spin inclusion, the Clifford reflection product, the displayed hsame transports, the finite Cont routes, and the Pkg provenance rows supplied by the two source certificates.

Next apply the scoped boundary theorem Definition 149.87. The common boundary is determined by the Spin boundary row and the Clifford reflection-product row, with the parity classifier, endpoint transports, ledger entries, and provenance fixed componentwise.

Finally use Definition 149.88. It exhausts every public consumer projection by the readback packet, the Spin inclusion, the Clifford reflection-product and action rows, and the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. The exclusion clauses in the three cited theorems rule out any host orthogonal group, quotient Clifford algebra, matrix representation, or additional Pin carrier, so the public read factors through exactly the stated sibling lattice. □

Theorem 149.90 (PinGroup Clifford-Spin sibling route)**.** *Every accepted* $\mathsf{PinGroup}^{\uparrow}$ *carrier consumes the* $\mathsf{Clifford}^{\uparrow}$ *multiplication and quadratic-form rows of chapter 147, then consumes the* $\mathsf{SpinGroup}^{\uparrow}$ *even-subgroup route of chapter 148, before it exports the reflection-cover row of the local* $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ *surface. Thus the Pin packet is a finite Clifford reflection-cover extension of the Spin route, not an independent orthogonal-group carrier or a host matrix representation.*

Proof. Project the accepted carrier to its Clifford-facing rows. The multiplication row and quadratic-form row are read first, so reflection products are typed inside the displayed $\mathsf{Clifford}^{\uparrow}$ sibling surface.

Next project to the Spin-facing rows. The even-subgroup route is the displayed $\mathsf{SpinGroup}^{\uparrow}$ handoff, and it is consumed before any reflection-cover row is exported.

The local $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ row packages these sibling reads with the parity classifier, componentwise hsame transport, displayed Cont replay, and Pkg provenance. Since no carrier coordinate stores an ambient orthogonal group, a host matrix model, or a quotient Clifford algebra, every accepted Pin read factors through the Clifford rows and the Spin route before the reflection-cover row becomes public. □

Theorem 149.91 (PinGroup Spin-Clifford sibling route)**.** *The* $\mathsf{PinGroup}^{\uparrow}$ *packet consumes the* $\mathsf{Clifford}^{\uparrow}$ *unit surface of chapter 147 and extends the even-parity* $\mathsf{SpinGroup}^{\uparrow}$ *route of chapter 148 by adding the reflection-parity row. Thus the public Pin surface is not a detached orthogonal-group object: it is the finite Clifford reflection-product ledger together with the Spin inclusion route and one visible parity extension. The PinGroup reflection-parity row is meaningful only because chapter 147 supplies the Clifford unit and product rows and chapter 148 supplies the even-subgroup route.*

Proof. Project the carrier rows. First read the Clifford unit and reflection-product rows, then read the Spin even-parity inclusion rows, and finally append the Pin reflection-parity classifier.

The componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{PinGroup}^{\uparrow}}$ row keep the three reads on one finite sibling surface. The route exports no quotient Clifford algebra, host matrix representation, ambient orthogonal equality, or independent Pin carrier. □

Chapter 150

A Concrete Naming Certificate for $\mathsf{PreSheaf}^{\uparrow}$

The presheaf interface $\mathsf{PreSheaf}^{\uparrow}$ packages a contravariant functor from the open-sets category of a topological carrier to a target category, with restriction maps respecting composition and identity. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) and $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62).

150.1 Carrier definition

Theorem 150.1 (Presheaf source and target carrier obligations)**.** *Every* $\mathsf{PreSheaf}^{\uparrow}$ *certificate must expose the following carrier obligations. Its source carrier is the BEDC open-object category whose objects are* BHist*-indexed open rows and whose arrows are certified inclusion witnesses. Its target carrier is a* $\mathsf{Cat}^{\uparrow}$ *certificate carrying section objects and target hom rows. For each certified inclusion* $U \leq V$*, the certificate must carry a restriction-hom row from the section object over* V *to the section object over* U*. The classifier ledger must also contain the source-open, source-inclusion, target-object, and target-hom rows over* BHist*.*

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface carrier rows`

Proof. Unpack the ten-field roadmap. The traditional target is the ordinary contravariant presheaf assignment from opens to a target category, so the BEDC source must be a $\mathsf{Cat}^{\uparrow}$ surface of open objects and inclusion rows, and the target must be a $\mathsf{Cat}^{\uparrow}$ surface of section objects and target homs.

The carrier choice then names four rows: open objects over BHist, certified inclusions, target section objects, and target homs. The classifier choice names the target hom classifier, while the visible source boundary keeps every open and inclusion row attached to BHist membership.

The remaining roadmap fields record why these carriers are obligations rather than completed construction. Identity, composition, transport, hidden restriction witnesses, restriction-row exactness, the standard bridge, and the status row are still obligations of the same certificate surface, with theory status obligation and verification status unformalized. □

Theorem 150.2 (Presheaf identity restriction obligation)**.** *For every carried open object* U *of the BEDC source category, the restriction row attached to the certified identity inclusion* $U \leq U$ *must be classified as the identity morphism of the target* $\mathsf{Cat}^{\uparrow}$ *section object over* U*.*

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface empty restriction identity row`

Proof. Start with the traditional presheaf identity law. In BEDC form, the source category supplies the identity inclusion witness on the open object U, while the target $\mathsf{Cat}^{\uparrow}$ supplies the identity hom on the section object over U.

The carrier choice fixes both rows before any comparison is made. The classifier choice is the target hom classifier, so the exact identity obligation is not host equality of maps but classification of the restriction hom by the target identity row.

The ledger records the hidden restriction witness and the identity comparison witness. Classifier transport is inactive in this row, composition is not invoked, the standard bridge remains outside the certificate, and the status is obligation on the paper side with unformalized on the Lean side. □

Theorem 150.3 (Presheaf composition restriction obligation)**.** *For carried inclusions $U \leq V \leq W$ in the BEDC source category, the direct restriction row from W to U must be classified as the target-category composite of the restriction row from W to V followed by the restriction row from V to U.*

Proof. The traditional target is the contravariant presheaf composition law. The BEDC source supplies the composable inclusion witnesses $U \leq V$ and $V \leq W$, together with the carried composite inclusion $U \leq W$.

The carrier rows give the three target section objects and the three restriction hom witnesses. The classifier row is the target hom classifier, so the exactness statement compares the direct W-to-U restriction with the target $\mathsf{Cat}^{\uparrow}$ composite of the two displayed restriction homs.

The ledger boundary records the two source inclusion witnesses, their composite witness, the two-step target hom witnesses, and the direct composite restriction witness. Identity and transport remain separate rows, the standard bridge is not claimed, and the status is obligation with unformalized. □

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface restriction composition row`

Theorem 150.4 (Presheaf classifier transport and ledger obligation)**.** *Classified source opens, classified source inclusions, and classified target homs must transport every restriction row of a $\mathsf{PreSheaf}^{\uparrow}$ certificate. The exact ledger records precisely the source inclusion witness, the target hom witness, and the composite restriction witness needed for the transported restriction row, and it records no host open equality, cover quotient, stalk quotient, external sheafification, or standard bridge witness.*

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface ledger extension surface`
Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface section restriction transport`

Proof. The traditional target presheaf supplies restriction rows stable under change of certified presentation. BEDC names this as transport along the source-open classifier, the source-inclusion classifier, and the target hom classifier, all indexed over BHist.

The carrier choice fixes the open-object and target-object rows, and the classifier choice fixes the target hom classifier. Transport therefore moves restriction witnesses only through named classifier rows; it does not appeal to an ambient equality of host opens or to an external quotient of sections.

The ledger exactness row is the boundary of the certificate. It records the source inclusion witness, the target hom witness, and the composite restriction witness used by identity, composition, and transport obligations. Sheaf locality, gluing, stalk quotients, and the standard bridge remain outside this certificate boundary. □

Definition 150.5 (PreSheaf indexed restriction surface)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *in the sense of Definition 88.1, with point carrier* BHist*, visible membership row* $\mathsf{OpenAt}(i, b)$*, and membership transport along* hsame*. Write $j \leq_{\mathcal{T}} i$ for the displayed inclusion row saying that every* BHist *point of j is a* BHist *point of i. A* $\mathsf{PreSheaf}^{\uparrow}$ *indexed restriction surface over $\mathcal{T}$ is the following ten-field template.*

1. *Traditional target: a contravariant presheaf surface from the open-index category of $\mathcal{T}$ to a target certificate category.*
2. *BEDC source: the open-index, inclusion-witness, and composition rows supplied by* $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, *and* $\mathsf{Topology}^{\uparrow}$.
3. *Naming source: a* $\mathsf{NameCert}$ *row whose visible carrier is built from* BHist *and whose membership rows are stable under* hsame.
4. *Section carrier: for every open index i, a carrier* $\mathsf{Sec}(i)$.
5. *Section classifier: for every open index i, a classifier $\sim_i$ on* $\mathsf{Sec}(i)$*, with reflexive, symmetric, and transitive rows.*
6. *Restriction maps: for every inclusion witness $j \leq_{\mathcal{T}} i$, a map $\rho_{i,j} : \mathsf{Sec}(i) \to \mathsf{Sec}(j)$.*
7. *Identity row: for every $s : \mathsf{Sec}(i)$, the self-restriction satisfies $\rho_{i,i}(s) \sim_i s$.*
8. *Composition row: for every chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$ and every $s : \mathsf{Sec}(i)$,*
$$\rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s).$$
9. *Restriction classifier-stability row: every classifier comparison $s \sim_i t$ transports along every displayed restriction:*
$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$
10. *Exact ledger: the exported ledger consists only of indexed opens, displayed inclusion witnesses, section rows, and restriction provenance, all over* BHist *membership and* hsame*-stable transport.*

No row in this surface exports host open equality, cover quotients, stalk quotients, or external sheafification data.

Lean4 (definition) `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface`

Theorem 150.6 (PreSheaf indexed restriction functoriality surface)**.** *If* $\mathsf{PreSheafIndexedRestrictionSurface}(sec, \rho, id$ *holds,* $identity\mathsf{hsame}identity'$*, and* $\mathsf{Cont}(\rho, identity', composite')$*, then* $\mathsf{PreSheafIndexedRestrictionSurface}(sec, \rho, ide$ $composite\mathsf{hsame}composite'$*, and* $\mathsf{UnaryHistory}(composite')$ *all hold.*

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface functoriality surface`

Proof. Unpack Definition 150.5. The old surface supplies $\mathsf{UnaryHistory}(sec)$, $\mathsf{UnaryHistory}(\rho)$, $\mathsf{Cont}(sec, \rho, identity)$, and $\mathsf{Cont}(\rho, identity, composite)$.

Transport the first continuation result along $identity\mathsf{hsame}identity'$ to obtain $\mathsf{Cont}(sec, \rho, identity')$. Unary closure under Cont then gives $\mathsf{UnaryHistory}(identity')$, and the displayed continuation $\mathsf{Cont}(\rho, identity', comp$ gives $\mathsf{UnaryHistory}(composite')$. These four rows repack as $\mathsf{PreSheafIndexedRestrictionSurface}(sec, \rho, identity', compo$

It remains to compare the composite rows. Continuation respect for hsame applies to the reflexive comparison on ρ, the displayed comparison $identity\mathsf{hsame}identity'$, and the two continuation ledgers from ρ to $identity$ and from ρ to $identity'$. The result is $composite\mathsf{hsame}composite'$, completing the displayed functoriality surface. □

150.2 The certificate

Theorem 150.7 (PreSheaf indexed restriction standard bridge)**.** *Let $\mathcal{T}$ be an indexed open topology carrier over* BHist *in the sense of Definition 88.1. Any* PreSheafIndexedRestrictionSurface$(sec, \rho, identity, composite$ *over $\mathcal{T}$ determines a standard contravariant presheaf from the certified open-index category of $\mathcal{T}$ to the target certificate category: an open index i is sent to the section object $sec(i)$, a certified inclusion $j \leq_{\mathcal{T}} i$ is sent to the restriction row $\rho_{i,j} : sec(i) \to sec(j)$, identities are witnessed by identity, composites are witnessed by composite, and classifier transport is exactly the restriction classifier-stability row carried by the surface.*

Conversely, any standard contravariant presentation whose opens, inclusions, section objects, restriction maps, identity law, composition law, classifier transport, and exact provenance ledger are carried by the displayed BEDC rows over BHist *induces the same indexed restriction surface. This correspondence preserves identity, composition, classifier transport, and the exact ledger. It does not assert sheaf locality, gluing, stalk quotient formation, or equality of host opens outside the certified open-index category.*

Lean4 ✓ `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface StdBridge`

Proof. For the forward direction, unpack Definition 150.5. The open-index objects and certified inclusion rows give the source category, while the section carrier and target-hom classifier give the target certificate category. The identity and composition rows are exactly the presheaf functoriality laws on certified inclusions.

Apply Definition 150.2 for the identity comparison and Definition 150.3 for the contravariant composite comparison. Apply Definition 150.4 to transport restriction rows through the displayed classifiers and to identify the exported provenance ledger. The remaining stability of the surface under the displayed continuation data is Definition 150.6.

For the reverse direction, start with a standard contravariant presentation whose rows are already carried by the BEDC data. Package its open-index source, naming source, section carrier, classifiers, restriction maps, identity law, composition law, classifier-stability law, and exact ledger into the ten fields of Definition 150.5.

No additional quotient or host equality is used in this packaging. The data exported in both directions are the same indexed opens, displayed inclusion witnesses, section rows, restriction provenance, identity row, composition row, classifier transport row, and ledger boundary, so the bridge closes only the standard presheaf presentation of the indexed restriction surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PreSheaf}^{\uparrow}}$**.**

Constructive story: A presheaf surface is generated from an indexed open topology over BHist: open objects are indexed membership rows, arrows are certified inclusion witnesses, and each inclusion $j \leq_{\mathcal{T}} i$ carries a restriction row from the section object over i to the section object over j. The carrier accepts only the displayed open-index source, naming source, section carriers, section classifiers, restriction maps, identity row, composition row, classifier-stability row, and exact ledger; host open equality, cover quotients, stalk quotients, gluing data, and sheafification data are excluded. Two section or restriction presentations are classified by the section classifiers and by hsame transport through the open, inclusion, and target-hom rows, so restriction comparison is a target-category classifier row rather than equality of host maps. The source is the $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, and $\mathsf{Topology}^{\uparrow}$ open-index machinery, the pattern records contravariant identity, composition, and restriction-stability laws, and the ledger keeps indexed opens, inclusion witnesses, section rows, restriction provenance, identity witnesses, and composite restriction witnesses visible. The five name-certificate obligations are supplied by the carried section classifiers, whose reflexive, symmetric, and transitive rows classify sections, while the functoriality theorem transports identity and composite continuation data and repacks the same indexed restriction surface after classifier movement. A history h lies in the signature gap of a $\mathsf{PreSheaf}^{\uparrow}$ token p when the source-domain accepts this indexed restriction surface, the signature relation pairs it with an answer surface and evidence ledger, and token introduction names that answer as p; unfolded, h carries exactly the opens, inclusions, sections, restrictions, identity, composition, and transport rows encoded by p. The standard bridge reads the same rows as an ordinary contravariant presheaf on

the certified open-index category and reads back only presentations whose data are already carried by those rows, so the chapter registers this construction under the name PreSheaf$^{\uparrow}$ without adding sheaf locality, gluing, stalk quotients, or external sheafification.

Theory closure: TheoryGate(NameCert$_{\mathsf{PreSheaf}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper-side treatment consists of the indexed restriction surface, identity, composition, and transport rows, the standard bridge, the exact ledger boundary, and explicit non-claims for the BHist-indexed PreSheaf$^{\uparrow}$ construction.

Formal verification: FormalStatus(NameCert$_{\mathsf{PreSheaf}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PreSheafUp.PreSheafIndexedRestrictionSurface\_functoriality\_surface`

Bridge status: paperBridge

Not claimed: This chapter keeps sheaf locality, gluing, stalk quotient formation, external sheafification, and bridgeChecked verification outside the PreSheaf$^{\uparrow}$ indexed restriction surface.

Upgrade path: No higher paper-side theory grade remains for this PreSheaf$^{\uparrow}$ surface; bridgeChecked verification remains a separate verification-axis obligation.

The Sheaf$^{\uparrow}$ package consumes the presheaf restriction surface of chapter 150 over the indexed-open carrier of chapter 88; the sibling files below record the locality, gluing, refinement, and downstream-access rows for that dependency.

Chapter 151

A Concrete Naming Certificate for $\mathsf{Sheaf}^{\uparrow}$

The sheaf interface $\mathsf{Sheaf}^{\uparrow}$ packages a presheaf satisfying the locality and gluing axioms over open covers. It depends on $\mathsf{NameCert}_{\mathsf{PreSheaf}^{\uparrow}}$ (chapter 150) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88), and is the substrate for ringed-space and scheme certificates.

151.1 Carrier definition

151.2 The certificate

Definition 151.1 (Indexed section presheaf carrier)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ in the sense of Definition 88.1. Write*

$$j \leq_{\mathcal{T}} i :\Longleftrightarrow \forall x : X,\ \mathsf{OpenAt}(j, x) \Longrightarrow \mathsf{OpenAt}(i, x)$$

for inclusion of indexed opens. An indexed section presheaf carrier for $\mathsf{PreSheaf}^{\uparrow}$ over $\mathcal{T}$ is a tuple

$$\mathcal{P} = (\mathsf{Sec}, \sim, \rho, \mathsf{Eqv}, \mathsf{Id}, \mathsf{Comp}, \mathsf{Res})$$

with the following data. For every open index $i : \mathsf{OpenIx}$ there is a section carrier $\mathsf{Sec}(i)$ and a classifier $\sim_i$ on $\mathsf{Sec}(i)$; the field Eqv says that each $\sim_i$ is reflexive, symmetric, and transitive. For every inclusion $j \leq_{\mathcal{T}} i$ there is a restriction map

$$\rho_{i,j} : \mathsf{Sec}(i) \to \mathsf{Sec}(j).$$

The identity field says $\rho_{i,i}(s) \sim_i s$ for every $s : \mathsf{Sec}(i)$, and the composition field says that whenever $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$,

$$\rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

for every $s : \mathsf{Sec}(i)$. The restriction classifier-stability field says that, for every inclusion $j \leq_{\mathcal{T}} i$ and every $s, t : \mathsf{Sec}(i)$,

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

This is the section-level concrete form of the contravariant restriction data described in chapter 150.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafIndexedSectionPresheafCarrier`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedSectionPresheafCarrier endpoint unary`

Definition 151.2 (Indexed sheaf cover, compatibility, and locality)**.** *Let $\mathcal{P}$ be an indexed section presheaf carrier over $\mathcal{T}$ as in Definition 151.154. An indexed open cover of an open index $i : \mathsf{OpenIx}$ is a tuple $\mathcal{U} = (A, \iota, h, \gamma)$ consisting of an indexing type A, a family $\iota : A \to \mathsf{OpenIx}$, inclusion witnesses*

$$h_a : \iota(a) \leq_{\mathcal{T}} i$$

for all $a : A$, and the pointwise cover law

$$\gamma : \forall x : X,\ \mathsf{OpenAt}(i, x) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), x).$$

For a family $s_a : \mathsf{Sec}(\iota(a))$, compatibility over $\mathcal{U}$ means that for all $a, b : A$ the two restrictions to the indexed intersection agree:

$$\rho_{\iota(a),\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)}(s_a) \sim_{\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)} \rho_{\iota(b),\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)}(s_b),$$

where the needed inclusions are supplied by the finite-intersection law of Definition 88.1. A sheaf locality field for $\mathcal{P}$ says that, for every such cover $\mathcal{U}$ of i and every pair $g, h : \mathsf{Sec}(i)$,

$$(\forall a : A,\ \rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h)) \Longrightarrow g \sim_i h.$$

A sheaf gluing field says that every compatible local family over $\mathcal{U}$ has a section $g : \mathsf{Sec}(i)$ whose restriction to each $\iota(a)$ is classifier-equal to s_a. An indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ is an indexed section presheaf carrier equipped with both of these fields.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedCoverCompatibilityLocality common refinement`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedCoverCompatibilityLocality global rows`

Definition 151.3 (Stable sheaf restriction row)**.** *Let $\mathcal{P}$ be an indexed section presheaf carrier over $\mathcal{T}$ as in Definition 151.154. A stable sheaf restriction row for $\mathcal{P}$ is the field saying that classifier equality is preserved by every restriction: for every inclusion $j \leq_{\mathcal{T}} i$ and every pair $s, t : \mathsf{Sec}(i)$,*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

Thus section classifiers transport along the indexed open restriction maps carried by the presheaf surface.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafStableRestrictionRow restricted comparison readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafStableRestrictionRow transport`

Theorem 151.4 (Sheaf carrier support obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Then the carrier support exposed by $\mathcal{S}$ is only the indexed section presheaf carrier over the $\mathsf{Topology}^{\uparrow}$ open-index surface, together with the sheaf locality and gluing fields. Concretely, every carrier row is one of the section carriers $\mathsf{Sec}(i)$, section classifiers $\sim_i$, equivalence fields, restriction maps $\rho_{i,j}$, identity and composition laws, indexed open-cover data, compatible local-family predicates, locality implications, or gluing witnesses described in Definition 151.154 and Definition 151.155.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCarrierSupport obligation`

Proof. Project the indexed section presheaf component of $\mathcal{S}$. Its displayed rows are precisely the open-indexed section carriers, classifiers, equivalence witnesses, restriction maps, and restriction identity and composition laws of Definition 151.154.

Project the sheaf component of $\mathcal{S}$. By Definition 151.155, this contributes only indexed open covers, compatible local families, the locality field, and the gluing field. No additional carrier row is introduced beyond those two displayed interfaces, so the asserted carrier support is exhausted by their projections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCarrierSupport obligation rows`

Theorem 151.5 (Sheaf classifier transport obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$, and suppose its indexed section presheaf carrier carries a stable sheaf restriction row in the sense of Definition 151.156. Then section classifiers transport through every restriction. Moreover, for every indexed open cover $\mathcal{U} = (A, \iota, h, \gamma)$ of i and every pair $g, h' : \mathsf{Sec}(i)$,*

$$(\forall a : A,\ \rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h')) \Longrightarrow g \sim_i h'.$$

Thus restriction stability gives classifier transport along the presheaf maps, and sheaf locality turns coverwise classifier equality back into classifier equality on the covered open.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger classifier transport obligation`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted global transport closure`

Proof. For a single inclusion $j \leq_{\mathcal{T}} i$, the transport claim is exactly the stable restriction row of Definition 151.156: a classifier equality $s \sim_i t$ restricts to $\rho_{i,j}(s) \sim_j \rho_{i,j}(t)$. The source and target classifiers are those supplied by the section presheaf carrier, and their reflexive, symmetric, and transitive behavior is supplied by the equivalence field of Definition 151.154.

For a cover $\mathcal{U}$, assume the displayed coverwise equality of the restrictions of g and h'. This is exactly the antecedent required by the sheaf locality field in Definition 151.155. Applying that field over the same indexed cover gives $g \sim_i h'$, which is the global classifier equality claimed. □

Theorem 151.6 (Sheaf cover ledger exactness obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. An accepted cover and gluing ledger row for $\mathcal{S}$ consists exactly of indexed open-cover data $\mathcal{U} = (A, \iota, h, \gamma)$, the finite-intersection rows of the $\mathsf{Topology}^{\uparrow}$ carrier, a compatible local section family $s_a : \mathsf{Sec}(\iota(a))$, and a global section $g : \mathsf{Sec}(i)$ with displayed restriction equalities*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The ledger uses no host open-cover equality and no quotient primitive: all comparison data is expressed through indexed opens, restriction maps, and the section classifiers of Definition 151.154.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger exactness rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger exactness obligation`

Proof. Unfold the indexed open cover and compatibility clauses in Definition 151.155. The cover ledger records only the indexing type A, the open-index map ι, the inclusion witnesses $h_a : \iota(a) \leq_{\mathcal{T}} i$, and the pointwise cover law γ. Compatibility is recorded by restricting local sections to the finite-intersection index supplied by the $\mathsf{Topology}^{\uparrow}$ carrier and comparing them with the classifier on that intersection.

Unfold the gluing clause of the same definition. A gluing ledger row adds a section $g : \mathsf{Sec}(i)$ and the displayed restriction equalities against the compatible family. The theorem Definition 151.170 checks that restrictions of a global section produce a compatible local family, while Definition 151.169 checks that two displayed gluings of the same compatible family are classifier-equal.

The two definitions just unfolded contain no host-side equality of open covers and no quotient constructor. Their comparison mechanism is the indexed inclusion relation $j \leq_{\mathcal{T}} i$, the finite-intersection row of the topology carrier, the restriction maps, and the section classifiers. Hence the listed rows are exact for the accepted cover and gluing ledger. □

Definition 151.7 (Sheaf BHist cover nerve ledger)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *whose point carrier is* BHist*, whose membership classifier is* OpenAt*, and whose finite-intersection index is written* $\wedge_{\mathsf{Topology}^{\uparrow}}$*. Let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. The* BHist *cover nerve ledger of $\mathcal{U}$ is the finite-intersection ledger surface with the following rows.*

1. *For each $a : A$, the member row is the indexed open $\iota(a)$ with its inclusion witness $h_a : \iota(a) \leq_{\mathcal{T}} i$ and its visible membership predicate* $\mathsf{OpenAt}(\iota(a), b)$ *for* $b : \mathsf{BHist}$*.*
2. *For each $a, b : A$, the pair-overlap row is $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, together with the two finite-intersection projections $k_{a,b} \leq_{\mathcal{T}} \iota(a)$ and $k_{a,b} \leq_{\mathcal{T}} \iota(b)$ and the two restriction rows from* $\mathsf{Sec}(\iota(a))$ *and* $\mathsf{Sec}(\iota(b))$ *to* $\mathsf{Sec}(k_{a,b})$*.*
3. *For each $a, b, c : A$, the triple-overlap row is $\ell_{a,b,c} := k_{a,b} \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c)$, together with the finite-intersection projections $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b}$ and $\ell_{a,b,c} \leq_{\mathcal{T}} \iota(c)$ and the restriction rows obtained by composing through $k_{a,b}$ or directly from any member whose inclusion route is displayed.*

The ledger contains only OpenAt *rows, finite-intersection projections, inclusion witnesses, presheaf restriction rows, section classifiers, and visible* BHist *provenance. It does not identify cover presentations by host equality and does not quotient overlap rows.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger`

Theorem 151.8 (Sheaf BHist cover nerve overlap projections)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist*, let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. For a local family $s_a : \mathsf{Sec}(\iota(a))$, compatibility over $\mathcal{U}$ is exactly the assertion that every pair-overlap row of the* BHist *cover nerve ledger carries classifier equality between the two projected restrictions:*

$$\forall a, b : A, \quad \rho_{\iota(a),k_{a,b}}(s_a) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(s_b),$$

where $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ and the inclusions are the finite-intersection projections of Definition 88.1.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison overlap projection rows`

Proof. Unfold the compatibility predicate in Definition 151.155. Its only comparison clauses are indexed by pairs $a, b : A$, and each clause restricts s_a and s_b to the common intersection $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ before comparing them by the section classifier at that intersection.

Unfold the cover nerve ledger of Definition 151.160. Its pair-overlap rows are exactly the same indices $k_{a,b}$, with the same two finite-intersection projections supplied by Definition 88.1. Therefore every compatibility clause is a classifier equality on one pair-overlap ledger row.

Conversely, assume every pair-overlap ledger row carries the displayed classifier equality. Reading those rows through the same two projections gives precisely the clauses required by Definition 151.155. Restriction composition is not extra data here; it only confirms that any displayed route to $k_{a,b}$ reads as the same section classifier row as the direct pair-overlap restriction. □

Theorem 151.9 (Sheaf BHist cover nerve gluing readback)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist*, let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ is a gluing section with*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

Then the BHist *cover nerve ledger reads g back to the local family on every member row and to the induced compatible family on every pair-overlap row. Explicitly, for every $a : A$,*

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a,$$

and for every $a, b : A$, writing $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^\uparrow} \iota(b)$,

$$\rho_{\iota(a),k_{a,b}}(\rho_{i,\iota(a)}(g)) \sim_{k_{a,b}} \rho_{\iota(a),k_{a,b}}(s_a)$$

and

$$\rho_{\iota(b),k_{a,b}}(\rho_{i,\iota(b)}(g)) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(s_b).$$

Consequently the two restrictions of g to each pair-overlap ledger row recover the same classifier row as the compatible family there.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger gluing readback`

Proof. The member readback is the gluing equality displayed in the statement. It is exactly the gluing clause of Definition 151.155 evaluated at the member row $\iota(a)$.

For a pair $a, b : A$, write $k = \iota(a) \wedge_{\mathsf{Topology}^\uparrow} \iota(b)$. The finite-intersection field of Definition 88.1 supplies $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$, which are the two pair-overlap projections recorded by Definition 151.160.

Transport the gluing equality $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$ along $k \leq_{\mathcal{T}} \iota(a)$ using the restriction classifier-stability field from Definition 151.154. This gives the first pair-overlap readback equality. The same transport along $k \leq_{\mathcal{T}} \iota(b)$ gives the second one.

It remains to compare the two readings of g on k. Restriction composition identifies the restriction of $\rho_{i,\iota(a)}(g)$ to k and the restriction of $\rho_{i,\iota(b)}(g)$ to k with the direct restriction $\rho_{i,k}(g)$, up to the classifier $\sim_k$. The compatibility of s_a gives the corresponding equality between the two local-family restrictions. Transitivity and symmetry of $\sim_k$ combine these rows, so the pair-overlap ledger reads the gluing section back to the same classifier row as the compatible local family. □

Theorem 151.10 (Sheaf NameCert carrier obligation)**.** *Every indexed sheaf carrier for* $\mathsf{Sheaf}^\uparrow$ *over an indexed open topology carrier $\mathcal{T}$ determines the carrier-facing rows required by the* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *template. The rows are fixed as follows.*

1. *The traditional target is sheaf-theoretic section data over open covers.*
2. *The BEDC source is the presheaf restriction source of* $\mathsf{PreSheaf}^\uparrow$ *together with the indexed-open source of* $\mathsf{Topology}^\uparrow$*.*
3. *The carrier is the indexed section presheaf of Definition 151.154, equipped with the locality and gluing fields of Definition 151.155.*
4. *The classifier row is the family of section classifiers $\sim_i$.*
5. *The stability row is supplied by the restriction identity, restriction composition, restriction classifier-stability, and classifier-equivalence fields.*
6. *The ledger is the displayed data of covers, intersections, inclusion witnesses, and restriction maps.*
7. *The exactness row is visible cover-local readback: compatible local sections glue to a global section and locality identifies any two global readbacks with the same cover restrictions.*

8. *The standard bridge row is absent; this certificate surface records the BEDC sheaf interface without claiming an external set-theoretic sheaf bridge.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger semantic name certificate`

Proof. Unpack the indexed sheaf carrier. Its presheaf part is exactly the tuple $\mathcal{P} = (\mathsf{Sec}, \sim, \rho, \mathsf{Eqv}, \mathsf{Id}, \mathsf{Comp}, \mathsf{Res})$ of Definition 151.154, so the carrier, classifier, and restriction-stability rows are not extra data: they are the displayed section carriers, displayed section classifiers, and the identity, composition, equivalence, and restriction classifier-stability fields.

Unpack the cover part. The topology side contributes open indices, inclusions, and finite-intersection witnesses by Definition 88.1; the presheaf side contributes contravariant restriction scope by chapter 150. The remaining sheaf fields in Definition 151.155 are precisely locality and gluing over those covers, which give the visible cover-local readback row and leave no separate standard-bridge row to construct. □

Theorem 151.11 (Sheaf restriction stability obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. For inclusions $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$, restriction identity and restriction composition keep every section row inside the displayed classifier family. Explicitly, for $s : \mathsf{Sec}(i)$ one has*

$$\rho_{i,i}(s) \sim_i s$$

and

$$\rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s).$$

When k is an indexed intersection supplied by the finite-intersection field of Definition 88.1, the same restriction-composition statement applies to the two induced inclusion chains, so cover restrictions remain stable under the topology intersection witnesses.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restriction stability obligation`

Proof. The first displayed relation is the identity field of the indexed section presheaf carrier in Definition 151.154. The second displayed relation is the composition field for the chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. Both conclusions land in the classifier specified by the target open index, namely $\sim_i$ for identity and $\sim_k$ for composition.

For the intersection case, use Definition 88.1 to obtain the open-index intersection and its two projection inclusions. Combining either projection with a cover inclusion gives one of the admissible chains governed by the contravariant restriction interface of chapter 150. Applying the same composition field to those chains gives stability on every displayed intersection row. □

Theorem 151.12 (Sheaf locality and gluing NameCert obligation)**.** *For an indexed sheaf carrier $\mathcal{S}$ for $\mathsf{Sheaf}^{\uparrow}$, the locality and gluing fields supply the certificate rows consumed by Definition 151.169, Definition 151.170, and Definition 151.171. Namely, every compatible local family over an indexed open cover has a global readback section, and locality identifies any two global sections whose restrictions agree on every cover member.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison locality gluing semantic name certificate`

Proof. Fix an indexed cover $\mathcal{U} = (A, \iota, h, \gamma)$ of i and a compatible family $s_a : \mathsf{Sec}(\iota(a))$. The cover and compatibility predicates are those of Definition 151.155; their intersection clauses are meaningful because Definition 88.1 supplies finite-intersection witnesses, and their restriction clauses live in the contravariant presheaf scope of chapter 150.

Apply the gluing field to obtain a section $g : \mathsf{Sec}(i)$ whose restrictions classify as the family s_a. Apply the locality field to any pair of such global sections whose memberwise restrictions agree; this is exactly the premise used by Definition 151.169. The compatibility theorem uses the same restriction-composition row on intersections, and the idempotence corollary repacks gluing plus uniqueness for the local family obtained from one global section. □

Theorem 151.13 (SheafUp BHist obligation surface)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *whose point carrier is* BHist*, with classifier* hsame*, membership row* OpenAt*, and finite-intersection index $\wedge_{\mathsf{Topology}^{\uparrow}}$. Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$ in the sense of Definition 151.154 and Definition 151.155. Then $\mathcal{S}$ supplies the* $\mathsf{Sheaf}^{\uparrow}$ *carrier, section classifiers, restriction maps, restriction identity, restriction composition, indexed-cover rows, compatibility predicate, locality field, gluing field, and visible* BHist *ledger rows required by the paper-side* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation surface. The exposed data depend only on* $\mathsf{NameCert}_{\mathsf{PreSheaf}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$*, the row* OpenAt*, the finite-intersection rows of $\mathcal{T}$, and the classifier-equivalence fields on the section carriers.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistObligationSurface restriction cover package`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistObligationSurface cover restriction classifier`

Proof. Project the carrier and classifier rows from the indexed section presheaf carrier. For each open index i, the section carrier $\mathsf{Sec}(i)$ and classifier $\sim_i$ are visible, and the equivalence field supplies the reflexive, symmetric, and transitive classifier rules required by the certificate surface.

Project the restriction rows from the same tuple. The identity and composition fields are exactly the $\mathsf{PreSheaf}^{\uparrow}$ restriction identity and restriction composition obligations, with inclusions interpreted through the indexed-open preorder $j \leq_{\mathcal{T}} i$ defined by OpenAt. The finite-intersection row of $\mathcal{T}$ supplies the common target index used in compatibility checks.

Finally project the cover, compatibility, locality, and gluing rows from Definition 151.155. The visible ledger is BHist-scoped because every open membership and every inclusion witness is expressed by OpenAt on BHist, and no host powerset equality is part of the exposed surface. □

Theorem 151.14 (SheafUp restriction hsame transport obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over a* BHist*-scoped indexed open topology carrier $\mathcal{T}$. Suppose a second displayed carrier is obtained by componentwise* hsame *transport of the topology row* OpenAt*, the section carriers and classifiers, the restriction maps, and the indexed-cover rows. Then restriction compatibility, locality, and gluing for $\mathcal{S}$ transport to the displayed carrier. In particular, a compatible family remains compatible after transport, a locality hypothesis on transported restrictions yields transported classifier equality of global sections, and a gluing witness transports to a gluing witness for the transported local family.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger hsame transport readback`

Proof. Transport the topology row first. The $\mathsf{Topology}^{\uparrow}$ membership-stability field moves each occurrence of $\mathsf{OpenAt}(i, h)$ across the displayed hsame witness, so the inclusion witnesses $j \leq_{\mathcal{T}} i$ and the cover law for $\mathcal{U} = (A, \iota, \eta, \gamma)$ remain valid after transport.

Next transport the section rows. The componentwise transport hypothesis identifies each section carrier, classifier, and restriction map with its transported counterpart. Compatibility is a conjunction of classifier equalities after restricting to the finite-intersection index, so it follows by applying the transported restriction rows and then reading the section classifier equivalence fields in the transported carrier.

For locality, begin with transported agreement of two global sections on every cover member. Move that agreement back along the componentwise classifier transport, apply the locality field of $\mathcal{S}$, and transport the resulting global classifier equality forward. For gluing, transport the compatible family back, apply the gluing field of $\mathcal{S}$ to obtain a global section, and transport that witness and its restriction equalities forward. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger restriction hsame transport obligation`

Theorem 151.15 (SheafUp cover gluing NameCert export)**.** *The* BHist*-scoped surface of Definition 151.166, together with the transport obligation of Definition 151.167, the uniqueness theorem Definition 151.169, and the global-restriction compatibility theorem Definition 151.170, assembles the paper-side* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *obligation surface consumed by* $\mathsf{RingedSpace}^\uparrow$ *and* $\mathsf{Scheme}^\uparrow$*. The exported surface provides sheaf carrier data, restriction coherence, cover compatibility, locality, gluing, uniqueness up to section classifier equality, and compatibility of restrictions of global sections; it claims no standard bridge and no external powerset equality.*

Proof. Start with the obligation surface of Definition 151.166. It supplies the carrier, classifier, restriction, cover, compatibility, locality, gluing, and visible ledger rows over BHist, all routed through $\mathsf{NameCert}_{\mathsf{PreSheaf}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$.

Use Definition 151.167 to make those rows stable under the componentwise hsame transports seen by downstream chapters. This gives $\mathsf{RingedSpace}^\uparrow$ and $\mathsf{Scheme}^\uparrow$ a transport discipline for topology membership through OpenAt and for section equality through the displayed section classifiers.

The gluing consequences are then the two local theorems already present in the chapter. Apply Definition 151.169 for uniqueness of a glued global section up to the section classifier, and apply Definition 151.170 for compatibility of the local family obtained by restricting a global section.

Combining these rows gives exactly the paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ obligation surface needed by the dependent ringed-space and scheme certificates. Since every cited row stays inside indexed opens, OpenAt, hsame, and section classifiers, the package contains neither a standard sheaf bridge nor an equality of host powersets. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverGluingNameCert export`

Theorem 151.16 (Sheaf gluing uniqueness from locality)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^\uparrow$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $s_a : \mathsf{Sec}(\iota(a))$ be a compatible local family over $\mathcal{U}$. If $g, h : \mathsf{Sec}(i)$ both restrict to that family on every cover member, in the sense that*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$$

and

$$\forall a : A, \quad \rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a,$$

then the two cover-global sections are classifier-equal:

$$g \sim_i h.$$

Thus the gluing supplied by the $\mathsf{Sheaf}^\uparrow$ *carrier is unique up to the section classifier on the covered open.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingClassifier uniqueness from locality`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger gluing uniqueness from locality`

Proof. Fix a cover member $a : A$. The first restriction hypothesis gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The second restriction hypothesis gives

$$\rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a.$$

By symmetry of the section classifier on $\mathsf{Sec}(\iota(a))$, supplied by the equivalence field of Definition 151.154, this yields

$$s_a \sim_{\iota(a)} \rho_{i,\iota(a)}(h).$$

Transitivity of the same classifier then gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h).$$

Since a was arbitrary, the restrictions of g and h agree on every member of $\mathcal{U}$. Applying the sheaf locality field from Definition 151.155 gives $g \sim_i h$. □

Theorem 151.17 (Sheaf global restrictions form a compatible local family)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $g : \mathsf{Sec}(i)$. Define the local family $s_a : \mathsf{Sec}(\iota(a))$ by*

$$s_a = \rho_{i,\iota(a)}(g).$$

Then s_a is compatible over $\mathcal{U}$ in the sense of Definition 151.155. Explicitly, for every $a, b : A$, putting $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, one has

$$\rho_{\iota(a),k_{a,b}}(\rho_{i,\iota(a)}(g)) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(\rho_{i,\iota(b)}(g)),$$

where the restriction maps use the inclusions supplied by the finite-intersection field and the cover inclusions h_a and h_b.

Proof. Fix $a, b : A$ and write $k = \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$. The finite-intersection field of Definition 88.1 gives

$$\mathsf{OpenAt}(k, x) \Longrightarrow (\mathsf{OpenAt}(\iota(a), x) \wedge \mathsf{OpenAt}(\iota(b), x))$$

for every $x : X$. Projecting the two components gives inclusions $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$. Together with the cover inclusions $h_a : \iota(a) \leq_{\mathcal{T}} i$ and $h_b : \iota(b) \leq_{\mathcal{T}} i$, these are the two chains needed for presheaf restriction composition.

Apply the composition field of Definition 151.154 to the chain $k \leq_{\mathcal{T}} \iota(a) \leq_{\mathcal{T}} i$ and the section g. This gives

$$\rho_{\iota(a),k}(\rho_{i,\iota(a)}(g)) \sim_k \rho_{i,k}(g).$$

Applying the same composition field to $k \leq_{\mathcal{T}} \iota(b) \leq_{\mathcal{T}} i$ gives

$$\rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)) \sim_k \rho_{i,k}(g).$$

By symmetry of the section classifier on $\mathsf{Sec}(k)$, supplied by the equivalence field of Definition 151.154, the second displayed relation yields

$$\rho_{i,k}(g) \sim_k \rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)).$$

Transitivity of the same classifier combines this with the first displayed relation and gives

$$\rho_{\iota(a),k}(\rho_{i,\iota(a)}(g)) \sim_k \rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)).$$

Since a and b were arbitrary, this is exactly compatibility of the family $a \mapsto \rho_{i,\iota(a)}(g)$ over $\mathcal{U}$ as defined in Definition 151.155. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger global restrictions compatible`

Corollary 151.18 (Sheaf gluing of global restrictions is idempotent)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, \eta, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $g : \mathsf{Sec}(i)$. Define the local family $s_a : \mathsf{Sec}(\iota(a))$ by*

$$s_a = \rho_{i,\iota(a)}(g).$$

If $h : \mathsf{Sec}(i)$ is a gluing of this family, in the sense that

$$\forall a : A, \quad \rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a,$$

then

$$h \sim_i g.$$

Proof. By Definition 151.170, the family s_a is compatible over $\mathcal{U}$. For each $a : A$, the displayed definition of s_a and reflexivity of the classifier on $\mathsf{Sec}(\iota(a))$, supplied by Definition 151.154, give

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The gluing hypothesis gives

$$\rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a$$

for the same a. Since a was arbitrary, both cover-global sections h and g restrict to the compatible local family s_a on every member of $\mathcal{U}$. Applying Definition 151.169 with h as the first cover-global section and g as the second gives $h \sim_i g$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingGlobalRestrictions idempotent`

Theorem 151.19 (Sheaf restriction classifier transport)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. For every inclusion $j \leq_{\mathcal{T}} i$ and every pair $s, t : \mathsf{Sec}(i)$, classifier equality on the larger open transports along restriction:*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

Thus restriction rows are morphisms of the local section classifiers.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restriction classifier transport`

Proof. The indexed sheaf carrier contains the indexed section presheaf carrier of Definition 151.154. Apply its restriction classifier-stability field to the inclusion $j \leq_{\mathcal{T}} i$ and the given classifier row $s \sim_i t$. □

Theorem 151.20 (Sheaf compatible-family stability)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $t_a : \mathsf{Sec}(\iota(a))$ is a second local family with*

$$\forall a : A, \quad t_a \sim_{\iota(a)} s_a.$$

Then t_a is compatible over $\mathcal{U}$. Explicitly, for all $a, b : A$, writing $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, one has

$$\rho_{\iota(a),k_{a,b}}(t_a) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(t_b).$$

Proof. Fix $a, b : A$ and write $k = \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$. The finite-intersection field of Definition 88.1 supplies the inclusions $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$ needed to restrict both local sections to the overlap.

Apply Definition 151.172 to the classifier row $t_a \sim_{\iota(a)} s_a$ along $k \leq_{\mathcal{T}} \iota(a)$. This gives

$$\rho_{\iota(a),k}(t_a) \sim_k \rho_{\iota(a),k}(s_a).$$

Apply the same transport row to $t_b \sim_{\iota(b)} s_b$ along $k \leq_{\mathcal{T}} \iota(b)$, and then use symmetry of the classifier on $\mathsf{Sec}(k)$, to get

$$\rho_{\iota(b),k}(s_b) \sim_k \rho_{\iota(b),k}(t_b).$$

The compatibility of the original family gives

$$\rho_{\iota(a),k}(s_a) \sim_k \rho_{\iota(b),k}(s_b).$$

Transitivity of $\sim_k$ first composes the transported a-side row with the original compatibility row, and then composes the result with the reversed transported b-side row. The resulting row is exactly

$$\rho_{\iota(a),k}(t_a) \sim_k \rho_{\iota(b),k}(t_b).$$

Since a and b were arbitrary, the family t_a is compatible over $\mathcal{U}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingCompatibleFamily stability semantic certificate`

Theorem 151.21 (Sheaf cover refinement obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. A displayed refinement of $\mathcal{V}$ through $\mathcal{U}$ is a map $r : B \to A$ together with inclusion witnesses*

$$\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$$

for every $b : B$, compatible with the two cover rows in the sense that the composite inclusion $\kappa(b) \leq_{\mathcal{T}} \iota(r(b)) \leq_{\mathcal{T}} i$ is the displayed route from the refined cover member to i. This is a certificate-level obligation only over the indexed-open inclusion relation of Definition 151.155 and the carrier rows of Definition 88.1; it introduces no host powerset equality of covers and no quotient comparison between cover presentations.

Proof. Unpack the two cover rows. The original cover supplies $h_a : \iota(a) \leq_{\mathcal{T}} i$, and the refined cover supplies $q_b : \kappa(b) \leq_{\mathcal{T}} i$. A refinement chooses, for each refined row b, an original row $r(b)$ together with the displayed inclusion $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$.

The topology carrier already gives the meaning of $\leq_{\mathcal{T}}$ as indexed-open inclusion and supplies the finite-intersection rows used by the sheaf compatibility predicate. Therefore the refinement datum is fully expressed by the map r and the witnesses ϵ_b, with the target cover inclusion $h_{r(b)}$ recording the route onward to i.

No comparison of host open subsets is used. All later transport across the refinement is forced to pass through ϵ_b, restriction composition, and the section classifiers of the indexed presheaf carrier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace refinement obligation`

Theorem 151.22 (Sheaf compatible family pullback along refinement)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement*

of $\mathcal{V}$ through $\mathcal{U}$ in the sense of Definition 151.174. If $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$, then the pulled-back family

$$t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$$

is compatible over $\mathcal{V}$. Explicitly, for every $b, c : B$, writing $m_{b,c} := \kappa(b) \wedge_{\mathsf{Topology}^\uparrow} \kappa(c)$, one has

$$\rho_{\kappa(b),m_{b,c}}(t_b) \sim_{m_{b,c}} \rho_{\kappa(c),m_{b,c}}(t_c).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger refinement pullback`

Proof. Fix $b, c : B$ and write $m = \kappa(b) \wedge_{\mathsf{Topology}^\uparrow} \kappa(c)$. The finite-intersection field of Definition 88.1 gives $m \leq_{\mathcal{T}} \kappa(b)$ and $m \leq_{\mathcal{T}} \kappa(c)$. Composing these with the refinement inclusions gives routes $m \leq_{\mathcal{T}} \iota(r(b))$ and $m \leq_{\mathcal{T}} \iota(r(c))$.

Use restriction composition on the chain $m \leq_{\mathcal{T}} \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$. It identifies the left restriction of t_b with the direct restriction of $s_{r(b)}$ to m, up to the classifier $\sim_m$. The same argument on the c side identifies the right restriction of t_c with the direct restriction of $s_{r(c)}$ to m.

The original compatibility of s_a over $\mathcal{U}$ gives equality after restricting $s_{r(b)}$ and $s_{r(c)}$ to the overlap $\iota(r(b)) \wedge_{\mathsf{Topology}^\uparrow} \iota(r(c))$. Since m lies inside both original cover members by the refinement inclusions and the finite-intersection projections, restriction composition transports that compatibility row down to m.

Finally, transitivity of $\sim_m$ composes the two restriction-composition rows with the transported original compatibility row. The resulting equality is precisely the compatibility clause for t_b and t_c over the refined cover $\mathcal{V}$. □

Theorem 151.23 (Sheaf gluing descends along refined covers)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^\uparrow$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues it:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

Then g glues the pulled-back family $t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$ over $\mathcal{V}$:

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} t_b.$$

Proof. Fix $b : B$. The gluing row over $\mathcal{U}$ gives

$$\rho_{i,\iota(r(b))}(g) \sim_{\iota(r(b))} s_{r(b)}.$$

Apply restriction classifier transport along the refinement inclusion $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$ to obtain equality after restriction to $\kappa(b)$.

Restriction composition on the chain $\kappa(b) \leq_{\mathcal{T}} \iota(r(b)) \leq_{\mathcal{T}} i$ identifies the restriction of $\rho_{i,\iota(r(b))}(g)$ to $\kappa(b)$ with the direct restriction $\rho_{i,\kappa(b)}(g)$. By definition, the restriction of $s_{r(b)}$ along the same refinement inclusion is t_b.

Transitivity of the classifier on $\mathsf{Sec}(\kappa(b))$ composes the composition row with the transported gluing row. Since b was arbitrary, the displayed equality holds for every member of $\mathcal{V}$, so g glues the pulled-back family over the refined cover. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementGluing descent cont`

Theorem 151.24 (Sheaf locality detection under refinement)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Suppose $g, h' : \mathsf{Sec}(i)$ satisfy refined-cover equality*

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} \rho_{i,\kappa(b)}(h').$$

Assume moreover that every original cover member is detected by its refinement rows: for every $a : A$, the subfamily of indices $b : B$ with $r(b) = a$ is an indexed open cover of $\iota(a)$ through the inclusions $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(a)$. Then locality over $\mathcal{U}$ gives

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementLocality global detection`

Proof. Fix $a : A$. By the detection hypothesis, the rows with $r(b) = a$ form an indexed cover of $\iota(a)$. For each such row b, the refined equality compares the direct restrictions of g and h' to $\kappa(b)$.

Restriction composition on $\kappa(b) \leq_{\mathcal{T}} \iota(a) \leq_{\mathcal{T}} i$ rewrites those direct restrictions as the restrictions of $\rho_{i,\iota(a)}(g)$ and $\rho_{i,\iota(a)}(h')$ along ϵ_b. Thus the two sections $\rho_{i,\iota(a)}(g)$ and $\rho_{i,\iota(a)}(h')$ agree on every member of the detected cover of $\iota(a)$.

Apply the sheaf locality field of Definition 151.155 on this detected cover of $\iota(a)$. It gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h').$$

Since a was arbitrary, these equalities hold over all members of $\mathcal{U}$.

Applying the same locality field over the original cover $\mathcal{U}$ yields $g \sim_i h'$, which is the claimed global classifier equality. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refined locality detection`

Theorem 151.25 (Sheaf refined-cover gluing uniqueness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Let $s_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$, and write*

$$t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$$

for its pulled-back family over $\mathcal{V}$. Suppose $g, h' : \mathsf{Sec}(i)$ both glue this pulled-back family:

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} t_b, \qquad \forall b : B, \quad \rho_{i,\kappa(b)}(h') \sim_{\kappa(b)} t_b.$$

Assume moreover that every original cover member is detected by its refinement rows in the sense of Definition 151.177. Then

$$g \sim_i h'.$$

Proof. Fix $b : B$. The first gluing row compares $\rho_{i,\kappa(b)}(g)$ with t_b, and the second compares $\rho_{i,\kappa(b)}(h')$ with the same section t_b.

Use symmetry of the classifier on $\mathsf{Sec}(\kappa(b))$ to reverse the second equality, then transitivity to obtain

$$\rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} \rho_{i,\kappa(b)}(h').$$

Since b was arbitrary, this is refined-cover equality of g and h' over $\mathcal{V}$.

Apply Definition 151.177 to this refined-cover equality and to the displayed detection hypothesis for the rows with $r(b) = a$. It returns the global classifier equality $g \sim_i h'$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinedCover gluing uniqueness`

Theorem 151.26 (Sheaf refined-cover exactness obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. For every compatible family $s_a : \mathsf{Sec}(\iota(a))$ over $\mathcal{U}$, the pulled-back family $t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$ satisfies the following three exactness rows.*

1. *The family t_b is compatible over $\mathcal{V}$.*
2. *Every gluing $g : \mathsf{Sec}(i)$ of s_a over $\mathcal{U}$ also glues t_b over $\mathcal{V}$.*
3. *If every original cover member is detected by its refinement rows, then any two sections $g, h' : \mathsf{Sec}(i)$ gluing t_b over $\mathcal{V}$ satisfy $g \sim_i h'$.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinedCover exactness obligation`

Proof. The first row is exactly Definition 151.175: compatibility of the original family transports along the displayed refinement.

The second row is exactly Definition 151.176: a section gluing the original family restricts to a gluing of the pulled-back family over the refined cover.

The third row is Definition 151.178. Its hypotheses are precisely two displayed gluings of the same pulled-back family, plus the locality-detection row for the refinement fibers over each original cover member. □

Theorem 151.27 (Sheaf NameCert refinement exactness export)**.** *The paper-side obligation surface* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *exports refined-cover exactness over indexed opens. Concretely, for any indexed sheaf carrier over an indexed open topology carrier, every displayed refinement of indexed covers carries compatible families forward, descends original gluings to refined gluings, and makes any two refined gluings of the same pulled-back family globally classifier-equal whenever the original members are detected by their refinement rows. This exported row uses only* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *restriction composition, section classifiers, locality, and gluing; it does not use host powerset equality or quotient comparison.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCertRefinement exactness export`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCertRefinementExactness export`
↪ variant `BEDC.Derived.SheafUp.SheafRefinementExactness namecert export`

Proof. Project the carrier row from Definition 151.154. This supplies the open-indexed section carriers, classifiers, restriction maps, restriction composition, and classifier transport used in the refinement arguments.

Project the cover, compatibility, locality, and gluing rows from Definition 151.155. These rows supply indexed covers, overlap compatibility, the local-to-global classifier implication, and displayed gluing witnesses.

Assemble the export by applying Definition 151.179. Its three rows are already expressed in the projected data, so $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ exposes a single refined-cover exactness obligation to downstream ringed-space and scheme certificates without adding a host-level cover comparison. □

Theorem 151.28 (Sheaf cover self-refinement identity)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. Then the identity map*

$$\mathrm{id}_A : A \to A$$

together with the pointwise reflexive inclusion witnesses

$$\epsilon_a : \iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a))$$

forms a displayed refinement of $\mathcal{U}$ through itself in the sense of Definition 151.174.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCover self refinement`

Proof. Unpack Definition 151.174. To exhibit a displayed refinement of $\mathcal{U}$ through itself, it is enough to give a map from the refined index type to the original index type and, for each refined row $a : A$, an inclusion witness from the refined member to the chosen original member. Take the map to be $r := \mathrm{id}_A$.

Fix $a : A$. Since $\mathrm{id}_A(a) = a$, the required witness

$$\epsilon_a : \iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a))$$

is exactly the pointwise implication

$$\forall x : X,\ \mathsf{OpenAt}(\iota(a), x) \Longrightarrow \mathsf{OpenAt}(\iota(a), x),$$

which holds immediately from the definition of $\leq_{\mathcal{T}}$ in Definition 151.154. The indexed-open membership row OpenAt is part of the topology carrier recorded in Definition 88.1, so this is a valid displayed inclusion witness in the same ledger as every other cover inclusion.

The onward route from the refined member to the ambient open i is then

$$\iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a)) \leq_{\mathcal{T}} i.$$

Its second leg is the original cover inclusion $h_a : \iota(a) \leq_{\mathcal{T}} i$, because $\mathrm{id}_A(a) = a$. Therefore the refinement data are exactly the identity selector together with the reflexive inclusion witnesses, and these satisfy the compatibility clause required in Definition 151.174. Hence $\mathcal{U}$ is a displayed refinement of itself. □

Theorem 151.29 (Sheaf identity-refinement pullback identity)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$, and let $s_a : \mathsf{Sec}(\iota(a))$ be a local family over $\mathcal{U}$. Use the identity displayed refinement of $\mathcal{U}$ through itself from Definition 151.181, and write its pulled-back family as*

$$t_a := \rho_{\iota(\mathrm{id}_A(a)),\iota(a)}(s_{\mathrm{id}_A(a)}).$$

Then for every $a : A$,

$$t_a \sim_{\iota(a)} s_a.$$

In particular, for every compatible family over $\mathcal{U}$, pullback along the identity refinement is pointwise the original family in the section classifier:

$$\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a.$$

Proof. Fix $a : A$. By Definition 151.181, the identity refinement chooses the selector $\mathrm{id}_A : A \to A$ and the reflexive inclusion witness

$$\iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a)).$$

Since $\mathrm{id}_A(a) = a$, this is exactly the reflexive inclusion

$$\iota(a) \leq_{\mathcal{T}} \iota(a).$$

Thus the pulled-back row at a is the identity restriction of the same local section:

$$t_a = \rho_{\iota(a),\iota(a)}(s_a)$$

inside the section carrier indexed by $\iota(a)$.

The identity field of the indexed section presheaf carrier in Definition 151.154 gives, for every section $s : \mathsf{Sec}(\iota(a))$,

$$\rho_{\iota(a),\iota(a)}(s) \sim_{\iota(a)} s.$$

Applying this field to s_a gives

$$\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a.$$

Substituting the displayed description of t_a gives $t_a \sim_{\iota(a)} s_a$. The argument is pointwise in a and does not use any overlap clause, so it applies in particular when the family s_a is compatible over $\mathcal{U}$ in the sense of Definition 151.155. □

Theorem 151.30 (Sheaf refinement composition obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Suppose (r, ϵ) is a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$ and (s, η) is a displayed refinement of $\mathcal{W}$ through $\mathcal{V}$:*

$$r : B \to A, \qquad \epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b)),$$

$$s : C \to B, \qquad \eta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c)).$$

Then the composite map $r \circ s : C \to A$ together with the composite inclusions

$$\theta_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(s(c)))$$

is a displayed refinement of $\mathcal{W}$ through $\mathcal{U}$. The cover row for $\mathcal{W}$ remains the pointwise indexed-open implication

$$\forall x : X, \quad \mathsf{OpenAt}(i, x) \Longrightarrow \exists c : C,\ \mathsf{OpenAt}(\lambda(c), x),$$

and the route onward to i is the inclusion chain $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c))) \leq_{\mathcal{T}} i$.

Proof. For each $c : C$, the refinement row (s, η) gives $\eta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$. The refinement row (r, ϵ) gives $\epsilon_{s(c)} : \kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c)))$.

Transitivity of indexed-open inclusion in the topology carrier composes these two rows and gives the displayed inclusion $\theta_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(s(c)))$.

The cover data of $\mathcal{W}$ are not changed. Its row ω still says that every point of the ambient open i lies in some indexed member $\lambda(c)$, expressed by the displayed OpenAt implication. Together with the map $r \circ s$ and the inclusions θ_c, this is exactly the refinement obligation of Definition 151.174 for $\mathcal{W}$ through $\mathcal{U}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement composition`

Theorem 151.31 (Sheaf compatible-family pullback composes over refinements)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of $i : \mathsf{OpenIx}$. Let $(r, \epsilon) : \mathcal{V} \to \mathcal{U}$ and $(s, \eta) : \mathcal{W} \to \mathcal{V}$ be displayed refinements, and let $(r \circ s, \theta) : \mathcal{W} \to \mathcal{U}$ be their composite refinement from Definition 151.183. If $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$, define*

$$v_b := \rho_{\iota(r(b)),\kappa(b)}(u_{r(b)})$$

and

$$w_c := \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then w_c is classifier-equal to the one-step pullback

$$z_c := \rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))})$$

for every $c : C$, and the family w_c is compatible over $\mathcal{W}$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement pullback composition`

Proof. Apply Definition 151.175 to the refinement (r, ϵ). It gives compatibility of the family v_b over $\mathcal{V}$.

Apply the same theorem to the refinement (s, η) and the compatible family v_b. This gives compatibility of the tower-pulled family w_c over $\mathcal{W}$.

It remains to compare the two descriptions of the local section at $c : C$. By definition, w_c restricts $u_{r(s(c))}$ first along $\kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c)))$ and then along $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$. Restriction composition for the indexed section presheaf carrier identifies this two-step restriction with the direct restriction along the composite inclusion θ_c.

The classifier on $\mathsf{Sec}(\lambda(c))$ supplies symmetry and transitivity, so the restriction-composition row gives $w_c \sim_{\lambda(c)} z_c$. Since compatibility of w_c was already obtained from the second pullback application, the composite tower pullback is compatible over $\mathcal{W}$ and agrees with the one-step pullback along $(r \circ s, \theta)$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementPullback composition`

Theorem 151.32 (Sheaf gluing descent composes over refinement towers)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of $i : \mathsf{OpenIx}$. Let $(r, \epsilon) : \mathcal{V} \to \mathcal{U}$ and $(s, \eta) : \mathcal{W} \to \mathcal{V}$ be displayed refinements, with composite $(r \circ s, \theta) : \mathcal{W} \to \mathcal{U}$. Suppose $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues it:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} u_a.$$

Then g glues the composite pulled-back family over $\mathcal{W}$:

$$\forall c : C, \quad \rho_{i,\lambda(c)}(g) \sim_{\lambda(c)} \rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))}).$$

Proof. Apply Definition 151.176 to (r, ϵ) and the gluing row for g. This shows that g glues the pulled-back family $v_b := \rho_{\iota(r(b)),\kappa(b)}(u_{r(b)})$ over $\mathcal{V}$.

Apply Definition 151.176 again to (s, η) and the displayed gluing of v_b. This shows that g glues the two-step pulled-back family $w_c := \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)})$ over $\mathcal{W}$.

By Definition 151.184, w_c is classifier-equal to the one-step pullback $\rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))})$ for every $c : C$. Transitivity of the classifier on $\mathsf{Sec}(\lambda(c))$ composes this comparison with the two-step gluing row, giving the displayed gluing row for the composite refinement. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement tower gluing`

Definition 151.33 (Sheaf displayed common refinement span)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be*

indexed open covers of the same open index $i : \mathsf{OpenIx}$. A displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$ is a tuple

$$\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$$

where C is an indexing type, $\lambda : C \to \mathsf{OpenIx}$ is an indexed-open family, $r : C \to A$ and $s : C \to B$ choose the displayed members of the two covers, and for every $c : C$ there are inclusion witnesses

$$\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c)), \qquad \zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c)).$$

The row ω says that λ itself is an indexed open cover of i, with member inclusions obtained through either composite $\lambda(c) \leq_{\mathcal{T}} \iota(r(c)) \leq_{\mathcal{T}} i$ or $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c)) \leq_{\mathcal{T}} i$, and with the pointwise cover law

$$\forall x : X,\ \mathsf{OpenAt}(i, x) \Longrightarrow \exists c : C,\ \mathsf{OpenAt}(\lambda(c), x).$$

Thus the common refinement is displayed entirely inside indexed opens, inclusion rows, and presheaf restriction rows; it contains no host-level equality of covers.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan base change gluing package`

Theorem 151.34 (Sheaf displayed common refinement gives paired refinements)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$. If $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ is a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$ in the sense of Definition 151.186, then $\mathcal{W}$ is a displayed refinement of its cover through $\mathcal{U}$ and also a displayed refinement of its cover through $\mathcal{V}$:*

$$(r, \epsilon) : \mathcal{W} \to \mathcal{U}, \qquad (s, \zeta) : \mathcal{W} \to \mathcal{V}.$$

Both refinements are in the sense of Definition 151.174.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan paired refinements`

Proof. Unpack the displayed span. Its row ω makes $\lambda : C \to \mathsf{OpenIx}$ an indexed cover of i, so $\mathcal{W}$ has the cover side required to be the refined cover in Definition 151.174.

The first projection consists of the map $r : C \to A$ together with the inclusions $\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c))$. These are exactly the displayed refinement rows from $\mathcal{W}$ through $\mathcal{U}$, with the route onward to i supplied by the cover inclusion $h_{r(c)}$.

The second projection is identical with $\mathcal{V}$ in place of $\mathcal{U}$: the map $s : C \to B$ and the inclusions $\zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$ form the displayed refinement rows, with the cover inclusion $q_{s(c)}$ supplying the route to i. □

Theorem 151.35 (Sheaf common refinement gluing invariance)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ be a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$. Let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$ and let $v_b : \mathsf{Sec}(\kappa(b))$ be compatible over $\mathcal{V}$. Suppose their pullbacks to $\mathcal{W}$ are classifier-equal:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

If $g : \mathsf{Sec}(i)$ glues u_a over $\mathcal{U}$ and $h' : \mathsf{Sec}(i)$ glues v_b over $\mathcal{V}$, then

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCommonRefinementGluing carrier invariance`

Proof. Project the common refinement span by Definition 151.187. This gives the displayed refinement (r, ϵ) from $\mathcal{W}$ through $\mathcal{U}$ and the displayed refinement (s, ζ) from $\mathcal{W}$ through $\mathcal{V}$.

Apply Definition 151.176 to the first refinement and the gluing row for g. The result says that g glues the pulled-back family

$$c \mapsto \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)})$$

over $\mathcal{W}$.

Apply the same descent theorem to the second refinement and the gluing row for h'. This says that h' glues the pulled-back family

$$c \mapsto \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)})$$

over $\mathcal{W}$. The displayed classifier equality between the two pulled-back families lets the compatible-family stability row Definition 151.173 replace the second family by the first one without changing the gluing comparison.

Now both g and h' are displayed gluings of the same compatible family over the indexed cover $\mathcal{W}$. Apply Definition 151.169 over $\mathcal{W}$ to obtain $g \sim_i h'$. □

Theorem 151.36 (Sheaf cover presentation exactness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Let $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ be a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$. Suppose a compatible family $u_a : \mathsf{Sec}(\iota(a))$ over $\mathcal{U}$ and a compatible family $v_b : \mathsf{Sec}(\kappa(b))$ over $\mathcal{V}$ agree after pullback to $\mathcal{W}$:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then any global section class determined by gluing u_a over $\mathcal{U}$ is the same classifier class as any global section class determined by gluing v_b over $\mathcal{V}$. Equivalently, if $g, h' : \mathsf{Sec}(i)$ are displayed gluings of the two families, then $g \sim_i h'$. The exactness row uses only indexed cover presentations, displayed common refinement, restriction transport, and the classifiers $\sim_j$; it does not compare host powersets and does not form a quotient of cover presentations.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentation exactness`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentation common refinement global comparison`

Proof. Choose displayed gluings $g : \mathsf{Sec}(i)$ and $h' : \mathsf{Sec}(i)$ for the two compatible families. The hypotheses give exactly the common-refinement equality of the two pulled-back local families.

Apply Definition 151.188 to $\mathcal{U}$, $\mathcal{V}$, the displayed common refinement $\mathcal{W}$, the two compatible families, their pulled-back equality, and the two displayed gluing rows. It returns $g \sim_i h'$.

Since the argument never leaves the indexed cover ledger, the conclusion identifies the two global classifier classes without choosing a preferred cover presentation, comparing powersets of host opens, or adding any quotient primitive. □

Theorem 151.37 (Sheaf NameCert obligation surface)**.** *The concrete $\mathsf{Sheaf}^{\uparrow}$ certificate surface is the following ten-field obligation surface.*

1. *The traditional target is a sheaf of sections over indexed opens of an indexed topology carrier.*
2. *The BEDC source data are $\mathsf{Topology}^{\uparrow}$ together with $\mathsf{PreSheaf}^{\uparrow}$.*
3. *The carrier is the indexed section presheaf carrier of Definition 151.154.*

4. *The classifier family is the open-indexed relation* $\sim_i$ *on* $\mathsf{Sec}(i)$.
5. *The stability obligations are classifier equivalence, identity restriction, composition of restriction, restriction classifier transport, locality, gluing, uniqueness of gluing, compatibility of global restrictions, compatible-family stability, cover-refinement transport, refined-cover exactness, refinement tower composition, displayed common-refinement comparison, and cover-presentation exactness.*
6. *The ledger consists of open-cover indices, cover-member inclusions, finite-intersection overlap indices, and restriction rows to those overlaps.*
7. *The core theorem list is Definition 151.172, Definition 151.173, Definition 151.174, Definition 151.175, Definition 151.176, Definition 151.177, Definition 151.178, Definition 151.179, Definition 151.180, Definition 151.183, Definition 151.184, Definition 151.185, Definition 151.187, Definition 151.188, Definition 151.189, Definition 151.169, Definition 151.170, and Definition 151.171.*
8. *Exactness is locality together with existence and uniqueness of gluing for compatible local families, stable under displayed common refinement of indexed cover presentations.*
9. *There is no standard bridge attached to this surface.*
10. *The status is a paper-side obligation surface for downstream certificates using* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCert obligation surface`

Proof. Rows one through four are the data fixed by Definition 151.154 and Definition 151.155: indexed opens come from $\mathsf{Topology}^{\uparrow}$, section carriers and contravariant restriction rows come from $\mathsf{PreSheaf}^{\uparrow}$, and classifier equality is indexed by the same opens as the section carriers.

Row five is exactly the list of fields and consequences exposed in this section. Equivalence, identity, composition, and restriction transport are part of the carrier surface; locality and gluing are the sheaf fields; uniqueness, compatible global restrictions, idempotence of global gluing, compatible-family stability, cover-refinement transport, refined-cover exactness, refinement tower composition, common-refinement comparison, and cover-presentation exactness are the theorem rows cited in row seven.

Rows six through ten record how downstream constructions consume the surface. A ringed-space or scheme certificate can name cover indices, overlaps, refinement maps, common-refinement spans, and restriction rows without choosing a standard-model bridge; the exactness content available to it is precisely local detection plus existence and uniqueness of gluing, stable under passing to displayed refined covers, refinement towers, and comparison of two cover presentations through a displayed common refinement. □

Corollary 151.38 (Sheaf carried-section pullback associativity along composite refinement)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$. *Let* $\mathcal{U} = (A, \alpha, h, \mu)$, $\mathcal{V} = (B, \beta, q, \nu)$, *and* $\mathcal{W} = (C, \gamma, p, \omega)$ *be indexed open covers of* $i : \mathsf{OpenIx}$. *Suppose* $(r, \epsilon) : \mathcal{U} \to \mathcal{V}$ *and* $(s, \eta) : \mathcal{V} \to \mathcal{W}$ *are displayed refinements in the sense of Definition 151.174:*

$$r : A \to B, \qquad \epsilon_a : \alpha(a) \leq_{\mathcal{T}} \beta(r(a)),$$

$$s : B \to C, \qquad \eta_b : \beta(b) \leq_{\mathcal{T}} \gamma(s(b)).$$

Let $(s \circ r, \theta) : \mathcal{U} \to \mathcal{W}$ be their composite displayed refinement from Definition 151.183, where $\theta_a : \alpha(a) \leq_{\mathcal{T}} \gamma(s(r(a)))$ is the composite inclusion. If $\sigma_c : \mathsf{Sec}(\gamma(c))$ is a compatible carried local family over $\mathcal{W}$, define the two-stage pullback by

$$\tau_b := \rho_{\gamma(s(b)),\beta(b)}(\sigma_{s(b)}), \qquad \upsilon_a := \rho_{\beta(r(a)),\alpha(a)}(\tau_{r(a)}),$$

and define the direct composite pullback by

$$\zeta_a := \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

Then the section classifier on the refined member classifies the two pullbacks together:

$$\forall a : A, \qquad \upsilon_a \sim_{\alpha(a)} \zeta_a.$$

Moreover, the two-stage family υ_a is compatible over $\mathcal{U}$.

Proof. The composite refinement is obtained by composing the displayed inclusion rows. For each $a : A$, the refinement (r, ϵ) supplies $\epsilon_a : \alpha(a) \leq_{\mathcal{T}} \beta(r(a))$, and the refinement (s, η) supplies $\eta_{r(a)} : \beta(r(a)) \leq_{\mathcal{T}} \gamma(s(r(a)))$. Their transitive composite is the inclusion $\theta_a : \alpha(a) \leq_{\mathcal{T}} \gamma(s(r(a)))$ used by the one-step pullback along $s \circ r$.

Since σ_c is compatible over $\mathcal{W}$, Definition 151.175 applied to (s, η) gives compatibility of the pulled-back family τ_b over $\mathcal{V}$. Applying the same theorem to (r, ϵ) and the compatible family τ_b gives compatibility of υ_a over $\mathcal{U}$.

It remains to identify the two descriptions of the member at a. Expanding the definitions gives

$$\upsilon_a = \rho_{\beta(r(a)),\alpha(a)}(\rho_{\gamma(s(r(a))),\beta(r(a))}(\sigma_{s(r(a))})),$$

while the direct composite pullback is

$$\zeta_a = \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

These are the two restriction routes along the same chain

$$\alpha(a) \leq_{\mathcal{T}} \beta(r(a)) \leq_{\mathcal{T}} \gamma(s(r(a))).$$

The composition row of the indexed section presheaf carrier in Definition 151.154 compares the iterated restriction with the direct restriction in the target classifier $\sim_{\alpha(a)}$:

$$\rho_{\beta(r(a)),\alpha(a)}(\rho_{\gamma(s(r(a))),\beta(r(a))}(\sigma_{s(r(a))})) \sim_{\alpha(a)} \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

Substituting the definitions of υ_a and ζ_a gives $\upsilon_a \sim_{\alpha(a)} \zeta_a$ for every $a : A$. Thus pullback along the refinement tower and pullback along the composite refinement determine the same carried section class on each refined member. □

Definition 151.39 (Sheaf restricted-open carrier)**.** *Let $\mathcal{T}$ be a* BHist*-scoped indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$*, and let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$. Fix an open index $i : \mathsf{OpenIx}$. The restricted-open carrier $\mathcal{S}|_i$ is the carrier indexed by pairs (j, η_j) with*

$$\eta_j : j \leq_{\mathcal{T}} i.$$

On such an index it inherits

$$\mathsf{Sec}_{|i}(j, \eta_j) := \mathsf{Sec}(j), \qquad \sim_{|i,(j,\eta_j)} := \sim_j .$$

For an inclusion $\theta : k \leq_{\mathcal{T}} j$ between restricted indices, its restriction map is the ambient sheaf restriction

$$\rho^{|i}_{(j,\eta_j),(k,\eta_k)} := \rho_{j,k}.$$

The target witness $\eta_k : k \leq_{\mathcal{T}} i$ is the composite of θ with η_j, so every carrier, classifier, and restriction row stays inside the fixed open i without adding a host subspace, powerset restriction, quotient, or sheafification primitive.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier`

Theorem 151.40 (Sheaf restricted-open restriction laws)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. For every restricted index (j, η_j) and section $s : \mathsf{Sec}_{|i}(j, \eta_j)$, the inherited identity row satisfies*

$$\rho^{|i}_{(j,\eta_j),(j,\eta_j)}(s) \sim_{|i,(j,\eta_j)} s.$$

For every chain of restricted indices

$$(k, \eta_k) \leq_{\mathcal{T}} (j, \eta_j) \leq_{\mathcal{T}} (l, \eta_l)$$

and every $s : \mathsf{Sec}_{|i}(l, \eta_l)$, the inherited composition row satisfies

$$\rho^{|i}_{(j,\eta_j),(k,\eta_k)}(\rho^{|i}_{(l,\eta_l),(j,\eta_j)}(s)) \sim_{|i,(k,\eta_k)} \rho^{|i}_{(l,\eta_l),(k,\eta_k)}(s).$$

Moreover, for every restricted inclusion $(k, \eta_k) \leq_{\mathcal{T}} (j, \eta_j)$, classifier equality transports along the inherited restriction:

$$s \sim_{|i,(j,\eta_j)} t \Longrightarrow \rho^{|i}_{(j,\eta_j),(k,\eta_k)}(s) \sim_{|i,(k,\eta_k)} \rho^{|i}_{(j,\eta_j),(k,\eta_k)}(t).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier restriction laws`

Proof. Unfold Definition 151.236. The section carrier and classifier at (j, η_j) are exactly $\mathsf{Sec}(j)$ and $\sim_j$, and the identity restriction is the ambient row $\rho_{j,j}$.

Apply the identity field of the indexed section presheaf carrier in Definition 151.154. This gives the first displayed classifier equality.

For the composition row, forget only the fixed target witness to i. The displayed restricted chain is an ambient chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} l$, so the composition field of Definition 151.154 gives the second displayed equality after replacing each restricted restriction map by the corresponding ambient ρ.

The classifier-transport row is identical. A restricted classifier equality is an ambient equality $s \sim_j t$, and the restricted map from (j, η_j) to (k, η_k) is $\rho_{j,k}$. The restriction classifier-stability field of Definition 151.154 therefore gives the final displayed implication. □

Theorem 151.41 (Sheaf restricted-open locality and gluing)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let (j, η_j) be a restricted index and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of j whose member inclusions satisfy $h_a : \iota(a) \leq_{\mathcal{T}} j$. Regard it as a cover inside $\mathcal{S}|_i$ by assigning each member the composite witness*

$$\eta_{\iota(a)} : \iota(a) \leq_{\mathcal{T}} i.$$

Then locality and gluing hold in the restricted carrier. Thus, if $g, h' : \mathsf{Sec}_{|i}(j, \eta_j)$ and

$$\forall a : A, \quad \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(g) \sim_{|i,(\iota(a),\eta_{\iota(a)})} \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(h'),$$

then $g \sim_{|i,(j,\eta_j)} h'$. *If* $s_a : \mathsf{Sec}_{|i}(\iota(a), \eta_{\iota(a)})$ *is compatible on the restricted cover, then there is* $g : \mathsf{Sec}_{|i}(j, \eta_j)$ *with*

$$\forall a : A, \quad \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(g) \sim_{|i,(\iota(a),\eta_{\iota(a)})} s_a.$$

Compatible local families remain compatible after every further restriction inside $\mathcal{S}|_i$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier locality gluing descent`

Proof. View the restricted cover as the ambient indexed open cover of j with the same index type A, the same open-index map ι, the same inclusion witnesses h_a, and the same cover law γ. The extra witnesses $\eta_{\iota(a)}$ only record that each member also lies below the fixed open i.

After unfolding Definition 151.236, the displayed memberwise equality is exactly the ambient antecedent for the locality field of Definition 151.155. Applying that field to the ambient cover of j gives $g \sim_j h'$, which is the restricted classifier equality at (j, η_j).

For gluing, a compatible family in the restricted carrier is the same family $s_a : \mathsf{Sec}(\iota(a))$ with the same overlap equalities as in the ambient sheaf. Apply the gluing field of Definition 151.155 to obtain $g : \mathsf{Sec}(j)$ whose ambient restrictions classify as s_a. Unfolding the restricted carrier turns those rows into the displayed restricted gluing rows.

It remains to check that compatibility is stable under further restriction inside the fixed open. Fix two local members and restrict them to a smaller restricted index below their overlap. The finite-intersection row of Definition 88.1 supplies the overlap inclusions, and Definition 151.237 supplies restriction composition along the chains through that overlap. Transporting the original compatibility row by the same theorem and composing the resulting classifier equalities proves that the further-restricted family is still compatible. □

Theorem 151.42 (Sheaf restricted-open NameCert surface)**.** *For every* BHist*-scoped indexed open topology carrier* $\mathcal{T}$*, indexed sheaf carrier* $\mathcal{S}$ *over* $\mathcal{T}$*, and open index* $i : \mathsf{OpenIx}$*, the restricted-open carrier* $\mathcal{S}|_i$ *supplies the paper-side* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *surface used by open-local consumers such as ringed-space and scheme certificates. Its rows are:*

1. *carrier: restricted indices* (j, η_j) *with* $j \leq_{\mathcal{T}} i$ *and inherited section carriers* $\mathsf{Sec}(j)$*;*
2. *classifier: the inherited section classifiers* $\sim_j$*;*
3. *stability: identity, composition, and classifier transport from Definition 151.237, with classifier transport agreeing with Definition 151.172;*
4. *ledger: open-cover indices below* i*, cover-member inclusions, finite-intersection overlap rows, refinement maps, and restriction paths;*
5. *exactness: restricted locality and gluing from Definition 151.238, stable under refinement by Definition 151.175 and Definition 151.176;*
6. *standard bridge: absent, as in the global obligation surface of Definition 151.190.*

Thus an open-local consumer may work entirely below i *while reusing the global* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *carrier, classifier, stability, ledger, exactness, and absent-standard-bridge rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier semantic name certificate`

Proof. The carrier and classifier rows are immediate from Definition 151.236: every restricted index names an ambient open below i, and no section carrier or classifier is rebuilt.

The stability row is supplied by Definition 151.237. Its classifier-transport clause is the restricted form of Definition 151.172, because both statements use the same ambient restriction map $\rho_{j,k}$ and the same classifier family $\sim$.

The ledger row is inherited from the ambient cover and refinement ledgers. Covers below i are ambient covers whose target open also carries a witness to i, while refinement pullback and gluing descent are exactly the rows of Definition 151.175 and Definition 151.176 read inside the fixed-open slice.

Finally, Definition 151.238 gives the exactness row for locality and gluing below i. The global obligation surface of Definition 151.190 already records that no standard bridge is attached, and restricting to a fixed open adds no such bridge. These rows are precisely the $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ surface required by open-local consumers. □

151.3 Displayed base change surface

Definition 151.43 (Sheaf displayed base-change square)**.** *Let $\mathcal{T}$ and $\mathcal{T}'$ be indexed open topology carriers, and let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^\uparrow$ over $\mathcal{T}$. A displayed base-change square from $\mathcal{T}'$ to $\mathcal{T}$ consists of:*

1. *a map of indexed open carriers $f : \mathsf{OpenIx}' \to \mathsf{OpenIx}$ preserving finite intersections, cover ledgers, and inclusion rows;*
2. *a target open $i : \mathsf{OpenIx}$ and a source open $i' : \mathsf{OpenIx}'$;*
3. *a displayed inverse-image row saying that $f^{-1}(i)$ is represented by i' in the source carrier;*
4. *for every indexed cover $\mathcal{U} = (A, \iota, h, \gamma)$ of i, a displayed pulled-back cover $\mathcal{U}' = (A, \iota', h', \gamma')$ of i' with the same index type A;*
5. *for every $a : A$, a carrier row identifying $f(\iota'(a))$ with $\iota(a)$ below i, together with the induced restriction path from i' to $\iota'(a)$.*

The square is displayed: it records the index map, the represented inverse image, and the memberwise cover pullback without replacing the indexed-open carrier by a powerset model or adding a quotient of covers.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDisplayedBaseChangeSquare`

Theorem 151.44 (Sheaf base-change cover-pullback compatibility)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^\uparrow$ over $\mathcal{T}$, and let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square as in Definition 151.204. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i, let $\mathcal{U}' = (A, \iota', h', \gamma')$ be its displayed pulled-back cover of i', and let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$. Then the pulled-back local family*

$$u'_a := \rho_{\iota(a), f(\iota'(a))}(u_a)$$

is compatible over the displayed cover $\mathcal{U}'$ after the carrier identification $f(\iota'(a)) \cong \iota(a)$ supplied by the square.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change cover pullback`

Proof. Unfold compatibility over $\mathcal{U}'$. For two members $a, b : A$, the source overlap is represented by the finite intersection of $\iota'(a)$ and $\iota'(b)$ in $\mathcal{T}'$.

The base-change square preserves finite intersections, so applying f sends this source overlap to the target overlap of $\iota(a)$ and $\iota(b)$ below i. The displayed carrier rows identify the two restriction paths from $\iota(a)$ and $\iota(b)$ to that target overlap with the images of the corresponding source paths.

Use the compatibility row for u_a and u_b on the target overlap. Restriction composition in the indexed sheaf carrier transports this equality along the displayed source paths, and restriction classifier transport replaces each transported section by the displayed pullback u'_a and u'_b. This is exactly the required compatibility row over $\mathcal{U}'$. □

Theorem 151.45 (Sheaf base-change gluing pullback)**.** *Let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square from Definition 151.204. Let $\mathcal{U}$ be an indexed cover of i and let $\mathcal{U}'$ be its displayed pulled-back cover of i'. Suppose $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues u_a over $\mathcal{U}$:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} u_a.$$

Then the displayed pullback $g' := \rho_{i,f(i')}(g)$ glues the pulled-back family u'_a over $\mathcal{U}'$ after transporting along the represented inverse-image row $f^{-1}(i) \simeq i'$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger base change gluing pullback`

Proof. Fix $a : A$. The gluing row for g restricts g from i to $\iota(a)$ and compares it with u_a.

Transport this row through the displayed base-change square. The inverse-image row represents the source target by i', and the member row represents the pulled-back member by $\iota'(a)$. Restriction composition identifies the path from g' to $\iota'(a)$ with the target path from g through $\iota(a)$ followed by the displayed carrier transport.

Apply restriction classifier transport to the original gluing equality along that path. The right-hand side is precisely the displayed pulled-back section u'_a, and the left-hand side is the restriction of g' to the pulled-back member. Since a was arbitrary, these rows give the displayed gluing of the pulled-back family over $\mathcal{U}'$. □

Theorem 151.46 (Sheaf base-change common-refinement invariance)**.** *Let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square from Definition 151.204. Let $\mathcal{U}$ and $\mathcal{V}$ be indexed cover presentations of the same target open i, with compatible families u_a and v_b. Let $\mathcal{W}$ be a displayed common refinement on which the two presentations agree:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then the displayed base-changed cover presentations over i' agree after the pulled-back displayed common refinement. Consequently their glued global classifier classes over the represented source open are the same whenever the corresponding displayed gluings exist.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange common refinement projection`

Proof. Apply Definition 151.205 to $\mathcal{U}$ and to $\mathcal{V}$. This gives compatible pulled-back families over the two displayed source covers.

Pull back the common refinement ledger $\mathcal{W}$ along the same square. Finite-intersection preservation and the memberwise carrier rows give a displayed common refinement between the two source cover presentations, with the same refinement maps r and s on indices.

Transport the target equality on $\mathcal{W}$ along this pulled-back common refinement. Restriction composition and classifier transport identify the transported rows with the two source pulled-back families restricted to the pulled-back refinement members.

Now apply Definition 151.189 to the two source cover presentations and their pulled-back common refinement. It returns equality of the global classifier classes determined by the two displayed gluings over i'. □

Theorem 151.47 (Sheaf base-change NameCert obligation surface)**.** *For every displayed base-change square of indexed open carriers, the* $\mathsf{Sheaf}^{\uparrow}$ *base-change surface supplies the following* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation rows for downstream ringed-space and scheme consumers.*

1. *carrier: the restricted-open carrier of Definition 151.236, read at the represented source open* i'*;*
2. *classifier: the inherited sheaf classifier* $\sim_j$ *on every displayed source open below* i'*;*
3. *stability: restriction identity, composition, classifier transport, and restricted-open stability from Definition 151.239, together with cover-pullback compatibility from Definition 151.205;*
4. *ledger: the base-change square of Definition 151.204, its displayed cover pullback rows, pulled-back common refinements, and the refinement ledgers used by Definition 151.175;*
5. *exactness: gluing pullback from Definition 151.206, refinement gluing descent from Definition 151.176, and common-refinement invariance from Definition 151.207 together with Definition 151.189;*
6. *standard bridge: absent, as in Definition 151.239.*

Thus base change is exposed as a paper-side obligation surface on indexed opens, not as a host-space powerset construction, and downstream certificates can cite it for cover pullback, gluing pullback, and common-refinement invariance.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange name certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange obligation surface`

Proof. The carrier and classifier rows are inherited from Definition 151.236. The represented source open i' determines the same fixed-open slice used by restricted-open consumers, so no new section carrier or classifier is introduced.

The stability and ledger rows are exactly the displayed data of Definition 151.204. Cover members, overlaps, and refinement maps remain indexed rows, while Definition 151.239 supplies the ambient restricted-open stability and Definition 151.175 supplies refinement pullback inside the ledger.

The exactness row combines the three base-change obligations. Cover-pullback compatibility is Definition 151.205; gluing pullback is Definition 151.206; common-refinement invariance is Definition 151.207, with Definition 151.176 and Definition 151.189 providing the existing sheaf exactness rows that the surface reuses.

Finally, the restricted-open surface already records that no standard bridge is attached. The base-change square only displays how indexed rows move across source and target opens, so it adds no bridge row and remains a paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface for downstream certificates. □

151.4 Sheaf base-change functoriality

Definition 151.48 (Sheaf composable base-change squares)**.** *Let* $\mathcal{T}$*,* $\mathcal{T}'$*, and* $\mathcal{T}''$ *be indexed open topology carriers, and let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over* $\mathcal{T}$*. Let*

$$\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$$

be displayed base-change squares in the sense of Definition 151.204. The two squares are composable when the following rows assemble into the displayed square $\mathcal{T}'' \to \mathcal{T}$*:*

1. *the open-index map is the composite $f \circ g$ and preserves finite intersections by composing the finite-intersection rows of g and f;*
2. *the represented inverse-image row for a target open i identifies the source open i'' with the two-step inverse image obtained first in $\mathcal{T}'$ and then in $\mathcal{T}''$;*
3. *cover ledgers compose pointwise: an indexed cover $\mathcal{U}$ of i pulls back to a cover $\mathcal{U}'$ of i' and then to a cover $\mathcal{U}''$ of i'' with the same index type and the same member order;*
4. *inclusion rows compose from each member $\iota''(a)$ through $\iota'(a)$ to $\iota(a)$, and the route to the target open is the composite of the two displayed routes;*
5. *memberwise pullback rows for sections, overlaps, and common-refinement members are classifier-equal to the corresponding rows read directly along the composite square.*

Thus a composable pair is not extra carrier data. It is the assertion that the displayed index maps, inverse-image representatives, finite-intersection rows, cover ledgers, inclusion rows, and memberwise pullback rows form the same downstream-readable base-change surface as the one-step square.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafComposableBaseChangeSquares`

Theorem 151.49 (Sheaf base-change identity square neutrality)**.** *For an indexed sheaf carrier $\mathcal{S}$ over $\mathcal{T}$, the identity displayed base-change square on $\mathcal{T}$ preserves the cover, restriction, gluing, and common-refinement rows of Definition 151.208. In particular, reading the $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface through the identity square gives the same indexed cover members, overlap witnesses, restriction composites, displayed gluing rows, and common-refinement comparisons as reading the surface directly.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChangeIdentitySquare restriction neutrality`

Proof. The identity square uses the identity map on open indices. Its inverse-image representative for an open i is i itself, and its displayed cover pullback sends each cover member $\iota(a)$ to the same indexed member with the same inclusion row.

Cover compatibility follows because the overlap of two unchanged members is the original finite intersection. Restriction identity in the indexed section carrier identifies each pulled-back local section with the section already present in the target cover ledger.

For gluing, Definition 151.206 specializes to the identity restriction path. The displayed row is therefore classifier-equal to the original gluing row by restriction identity and classifier reflexivity.

For common refinements, the displayed span, the refinement maps, and the member inclusions are unchanged. Applying Definition 151.207 along the identity square returns the same comparison row, and Definition 151.189 reads the same global classifier class. □

Theorem 151.50 (Sheaf base-change cover-pullback composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i in $\mathcal{T}$. Pulling $\mathcal{U}$ through the composite square gives the same cover-member witnesses, member inclusion routes, finite-intersection overlap rows, and compatible local-family rows as pulling $\mathcal{U}$ first through f and then through g, up to the classifier identifications recorded by the composable-square rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change composition`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change cover pullback composition`

Proof. Start with a compatible family over $\mathcal{U}$. Apply Definition 151.205 to the square f. This gives a compatible displayed family over the pulled-back cover $\mathcal{U}'$ in $\mathcal{T}'$.

Apply the same theorem to g and the compatible family over $\mathcal{U}'$. The result is a compatible family over the two-step cover $\mathcal{U}''$ in $\mathcal{T}''$.

By Definition 151.209, the composite square has the same cover index type A, the same displayed member order, and the same member inclusion route as the two-step cover ledger. Its finite-intersection preservation row is the composite of the two overlap rows.

Restriction composition identifies the two-step section pullback with the direct pullback along the composite route. Classifier transport then replaces the two-step overlap comparisons by the one-step comparisons, so the downstream cover-pullback surface is the same in both readings. □

Theorem 151.51 (Sheaf base-change gluing pullback composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U}$ be an indexed cover of i and let u_a be a compatible local family glued by $s : \mathsf{Sec}(i)$. The displayed gluing row transported along the composite square is classifier-equal to the row obtained by first transporting the gluing row along f and then transporting the resulting row along g.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger base change gluing pullback composition`

Proof. Apply Definition 151.206 to f. It transports the gluing row for s to a displayed gluing row over the cover pulled back to $\mathcal{T}'$.

Apply Definition 151.206 again to g. This transports the intermediate gluing row to the two-step pulled-back cover over $\mathcal{T}''$.

The composability rows identify the source open of the one-step composite square with the two-step represented inverse image. They also identify each pulled-back cover member and member inclusion route with the corresponding two-step member route.

Restriction composition in the sheaf carrier identifies the restriction of s along the composite route with the successive restrictions through the intermediate open. Classifier transport along the displayed member identifications converts the two-step gluing comparison into the direct composite-square comparison, giving the claimed classifier equality of gluing rows. □

Theorem 151.52 (Sheaf base-change common-refinement composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U}$ and $\mathcal{V}$ be indexed cover presentations of i whose compatible local families agree on a displayed common refinement. The common-refinement comparison obtained after pulling through the composite square agrees with the comparison obtained after successive pullbacks through f and g. Consequently the global classifier class supplied by Definition 151.189 is independent of whether downstream consumers read the composite square in one step or as the two displayed squares.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan base change composition`

Proof. Apply Definition 151.207 to f. This produces pulled-back cover presentations over $\mathcal{T}'$ and transports the displayed common-refinement comparison to the intermediate common refinement.

Apply the same theorem to g. The intermediate comparison becomes a comparison over the two-step pulled-back common refinement in $\mathcal{T}''$.

The composable-square definition identifies the one-step pulled-back covers, refinement maps, member inclusions, and finite-intersection rows with their two-step counterparts. In particular, each common-refinement member has the same downstream restriction route after composing the displayed inclusion rows.

Restriction composition and classifier transport move the two-step comparison to the direct composite comparison. Then Definition 151.189 reads either comparison as the same global classifier class over the represented source open, so the common-refinement surface is independent of the chosen displayed factorization. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange common refinement composition`

Theorem 151.53 (Sheaf base-change three-square associativity)**.** *Let*

$$\mathcal{T}''' \xrightarrow{h} \mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$$

be three displayed base-change squares for an indexed sheaf carrier $\mathcal{S}$ over $\mathcal{T}$. Assume that the adjacent pairs are composable in the sense of Definition 151.209, and that both induced pairwise composites are again composable with the remaining square. Then the two bracketed displayed composites from $\mathcal{T}'''$ to $\mathcal{T}$ determine the same downstream base-change surface. More explicitly, first composing f with g and then with h, or first composing g with h and then with f, gives the same classified rows for every pulled-back indexed cover, every pulled-back compatible family, every transported gluing row, and every pulled-back common-refinement presentation.

Proof. Fix an indexed cover $\mathcal{U} = (A, \iota, r, \gamma)$ of an open i in $\mathcal{T}$. Apply Definition 151.211 to the pair (f, g). The cover obtained over $\mathcal{T}''$ by first pulling through f and then through g has the same index type A, the same member order, the same finite-intersection overlap rows, and the same compatible-family restriction rows as the cover read through the composite square $f \circ g$. Applying the same theorem to the resulting composite square and h identifies the left-bracketed cover over $\mathcal{T}'''$ with the three-step cover obtained by pulling through f, then g, then h.

Apply Definition 151.211 instead to the pair (g, h), and then to f together with that composite. This identifies the right-bracketed cover over $\mathcal{T}'''$ with the same three-step cover ledger: the index type is still A, the member attached to $a : A$ is the inverse-image representative obtained by the same three displayed member rows, and the inclusion route to i is the same composite of displayed inclusion routes. Associativity of ordinary composition of the displayed open-index maps and of the restriction-route concatenations identifies the two parenthesizations of that route. Hence both bracketings classify the same cover-pullback rows and the same pulled-back compatible local-family rows.

Now let u_a be a compatible family over $\mathcal{U}$ and let $s : \mathsf{Sec}(i)$ be a displayed gluing section for it. Applying Definition 151.212 to (f, g) and then to the composite with h identifies the left-bracketed gluing row with the row obtained by restricting s successively along the three displayed square routes. Applying the same theorem to (g, h) and then to f identifies the right-bracketed gluing row with that identical successive restriction row. Restriction composition in the indexed sheaf carrier removes the parentheses from the route, and classifier transport along the composable-square member identifications gives equality of the two transported gluing rows.

Finally take two indexed cover presentations of i with compatible families agreeing on a displayed common refinement. By Definition 151.213, the left bracketing has the same pulled-back refinement indices, refinement maps, member inclusions, and comparison rows as the three-step pullback. The right bracketing is identified with that same three-step common-refinement ledger by applying the same theorem in the other pairwise order. The common-refinement members therefore restrict through the same downstream routes, and Definition 151.189 reads the two transported comparisons as the same global classifier class over the represented open of $\mathcal{T}'''$. □

Definition 151.54 (Sheaf cover-presentation classifier)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed*

open covers of the same open index $i : \mathsf{OpenIx}$. *Let* $u_a : \mathsf{Sec}(\iota(a))$ *be compatible over* $\mathcal{U}$ *and let* $v_b : \mathsf{Sec}(\kappa(b))$ *be compatible over* $\mathcal{V}$.

The two cover presentations $(\mathcal{U}, u)$ *and* $(\mathcal{V}, v)$ *are classifier-equal when there exists a displayed common refinement span*

$$\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$$

from $\mathcal{U}$ *to* $\mathcal{V}$ *such that their pulled-back local families agree pointwise by the section classifiers:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

This classifier is a presentation-level row: it compares indexed covers only through displayed common refinements, restriction maps, and the already exposed section classifiers $\sim_j$.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafCoverPresentationClassifier`

Theorem 151.55 (Sheaf cover-presentation classifier soundness)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$. *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *and* $\mathcal{V} = (B, \kappa, q, \delta)$ *be indexed open covers of the same open index* $i : \mathsf{OpenIx}$. *Let* $u_a : \mathsf{Sec}(\iota(a))$ *be compatible over* $\mathcal{U}$ *and let* $v_b : \mathsf{Sec}(\kappa(b))$ *be compatible over* $\mathcal{V}$. *If* $(\mathcal{U}, u)$ *and* $(\mathcal{V}, v)$ *are classifier-equal in the sense of Definition 151.219, then any displayed global gluing* $g : \mathsf{Sec}(i)$ *of* u_a *and any displayed global gluing* $h' : \mathsf{Sec}(i)$ *of* v_b *satisfy*

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison soundness`

Proof. Unpack the cover-presentation classifier. It supplies a displayed common refinement span $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ from $\mathcal{U}$ to $\mathcal{V}$ and a pointwise classifier equality between the two pulled-back local families over $\mathcal{W}$.

Apply Definition 151.189 to the same two covers, compatible families, displayed common refinement, pullback equality, and displayed gluings. The conclusion is exactly $g \sim_i h'$. □

Theorem 151.56 (Sheaf cover-presentation classifier symmetry)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$. *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *and* $\mathcal{V} = (B, \kappa, q, \delta)$ *be indexed open covers of the same open index* $i : \mathsf{OpenIx}$. *Let* $u_a : \mathsf{Sec}(\iota(a))$ *be compatible over* $\mathcal{U}$ *and let* $v_b : \mathsf{Sec}(\kappa(b))$ *be compatible over* $\mathcal{V}$. *If* $(\mathcal{U}, u)$ *is classifier-equal to* $(\mathcal{V}, v)$ *in the sense of Definition 151.219, then* $(\mathcal{V}, v)$ *is classifier-equal to* $(\mathcal{U}, u)$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentationClassifier symmetric`
↪ variant `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan symmetric`
↪ variant `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan symm`

Proof. Unpack the classifier witness. It consists of a displayed common refinement span $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ and, for every $c : C$, a classifier row from the $\mathcal{U}$-pullback to the $\mathcal{V}$-pullback.

Build the reversed displayed common refinement by keeping the same cover index C, the same open family λ, and the same cover row ω, while swapping the two projection maps and their inclusion witnesses. Thus the reversed span has projection $s : C \to B$ to $\mathcal{V}$ and projection $r : C \to A$ to $\mathcal{U}$.

For each $c : C$, apply symmetry of the section classifier on $\mathsf{Sec}(\lambda(c))$ to the original pullback equality. The resulting pointwise classifier row is exactly the equality required for the reversed cover-presentation classifier. □

Theorem 151.57 (Sheaf restricted-open cover-presentation classifier descent)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$, and fix an open index $i : \mathsf{OpenIx}$. Let $\mathcal{S}|_i$ be the fixed-open restricted carrier of Definition 151.239. For two compatible local-family presentations over covers of i, the cover-presentation classifier of Definition 151.219 restricts to the same classifier on $\mathcal{S}|_i$: if the ambient presentations are classifier-equal by a displayed common refinement, then their restricted-open global gluings are identified by the restricted classifier on the fixed-open carrier.*

Proof. Read the restricted-open carrier through Definition 151.239. Its carrier is the fixed open section surface $\mathsf{Sec}(i)$ with the inherited classifier $\sim_i$, so a restricted-open global gluing is an ambient global gluing whose target open is fixed to i.

Use Definition 151.238 to read the locality and gluing rows of $\mathcal{S}|_i$ as the ambient sheaf locality and gluing rows over covers of the same fixed open. Thus the displayed restricted-open gluings provide exactly the gluing hypotheses required by the ambient cover-presentation comparison.

Apply Definition 151.220. Since the restricted classifier is inherited from $\sim_i$, the ambient conclusion $g \sim_i h'$ is the desired restricted-open classifier equality. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted open descent`

Definition 151.58 (Sheaf restricted common-refinement ledger)**.** *Let $\mathcal{T}$ be a* BHist*-scoped indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$*, let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$, and fix an open index $i : \mathsf{OpenIx}$. Work in the restricted-open carrier $\mathcal{S}|_i$ of Definition 151.236. For a restricted target (j, η_j), let*

$$\mathcal{U}_{|i} = (A, \iota, \eta^{\iota}, h, \gamma), \qquad \mathcal{V}_{|i} = (B, \kappa, \eta^{\kappa}, q, \delta)$$

be two indexed open covers of (j, η_j), where each $\eta^{\iota}_a : \iota(a) \leq_{\mathcal{T}} i$ and $\eta^{\kappa}_b : \kappa(b) \leq_{\mathcal{T}} i$ records that the member lies below the fixed open i. A restricted common-refinement ledger from $\mathcal{U}_{|i}$ to $\mathcal{V}_{|i}$ is a tuple

$$\mathcal{W}_{|i} = (C, \lambda, \eta^{\lambda}, r, s, \epsilon, \zeta, \omega)$$

such that C is an indexing type, $\lambda : C \to \mathsf{OpenIx}$ is an indexed-open family, every $\eta^{\lambda}_c : \lambda(c) \leq_{\mathcal{T}} i$ lies below the fixed open, $r : C \to A$ and $s : C \to B$ choose members of the two covers, and

$$\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c)), \qquad \zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$$

for every $c : C$. The row ω says that λ is an indexed open cover of j, with member inclusions obtained by composing through either displayed leg and with pointwise cover law over $\mathsf{OpenAt}(j, x)$*.*

Thus the ledger is a displayed common-refinement span entirely inside the slice below i: all carrier rows are restricted indices, all maps are inherited restrictions of $\mathcal{S}$, and all comparisons are made by the section classifiers of Definition 151.236.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinementLedger`

Theorem 151.59 (Sheaf restricted common-refinement pullback compatibility)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let $\mathcal{U}_{|i}$ and $\mathcal{V}_{|i}$ be restricted covers of a restricted target (j, η_j), and let $\mathcal{W}_{|i}$ be a restricted common-refinement ledger between them in the sense of Definition 151.192. If*

$$u_a : \mathsf{Sec}_{|i}(\iota(a), \eta^{\iota}_a)$$

is compatible over $\mathcal{U}_{|i}$ and

$$v_b : \mathsf{Sec}_{|i}(\kappa(b), \eta^{\kappa}_b)$$

is compatible over $\mathcal{V}_{|i}$, then their pullbacks to the restricted common refinement are compatible families over $\mathcal{W}_{|i}$. Moreover, if the displayed comparison

$$\forall c : C, \quad \rho^{|i}_{(\iota(r(c)),\eta^{\iota}_{r(c)}),(\lambda(c),\eta^{\lambda}_{c})}(u_{r(c)}) \sim_{|i,(\lambda(c),\eta^{\lambda}_{c})} \rho^{|i}_{(\kappa(s(c)),\eta^{\kappa}_{s(c)}),(\lambda(c),\eta^{\lambda}_{c})}(v_{s(c)})$$

holds on $\mathcal{W}_{|i}$, then it is the same comparison as the ambient common-refinement comparison after forgetting only the witnesses to i.

Proof. Unfold the restricted carrier using Definition 151.236. The restricted section carriers and classifiers at $\iota(a)$, $\kappa(b)$, and $\lambda(c)$ are exactly the ambient carriers and classifiers at those open indices, and the restricted restriction maps are the ambient maps ρ.

The restricted ledger of Definition 151.192 forgets to a displayed common-refinement span in the ambient sheaf. By Definition 151.187, its two projections are displayed refinements through $\mathcal{U}$ and $\mathcal{V}$.

Apply the pullback part of the ambient refinement exactness surface to each projection. Equivalently, read Definition 151.184 with the identity tower on the displayed span: the pulled-back families are compatible over the common refinement, and the two-step and one-step restriction descriptions agree by classifier equality.

Finally, Definition 151.237 identifies the restricted restriction composition and classifier transport rows with the ambient ones. Therefore the displayed equality in the restricted carrier is exactly the ambient equality on the same local sections, with the witnesses η^{λ}_{c} recording only that the comparison remains below i. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinement pullback compatibility`

Theorem 151.60 (Sheaf restricted common-refinement gluing invariance)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let $\mathcal{U}_{|i}$ and $\mathcal{V}_{|i}$ be restricted covers of a restricted target (j, η_j), and let $\mathcal{W}_{|i}$ be a restricted common-refinement ledger between them. Let u_a and v_b be compatible local families over those two restricted covers. Suppose their pullbacks to $\mathcal{W}_{|i}$ agree:*

$$\forall c : C, \quad \rho^{|i}_{(\iota(r(c)),\eta^{\iota}_{r(c)}),(\lambda(c),\eta^{\lambda}_{c})}(u_{r(c)}) \sim_{|i,(\lambda(c),\eta^{\lambda}_{c})} \rho^{|i}_{(\kappa(s(c)),\eta^{\kappa}_{s(c)}),(\lambda(c),\eta^{\lambda}_{c})}(v_{s(c)}).$$

If $g : \mathsf{Sec}_{|i}(j, \eta_j)$ glues u_a over $\mathcal{U}_{|i}$ and $h' : \mathsf{Sec}_{|i}(j, \eta_j)$ glues v_b over $\mathcal{V}_{|i}$, then

$$g \sim_{|i,(j,\eta_j)} h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinement gluing invariance`

Proof. First forget the target witnesses to i. By Definition 151.236, the two restricted covers become ambient indexed covers of j, the local families are the same ambient section families, and the displayed gluings are ambient gluing rows.

The common-refinement ledger becomes the ambient displayed span required by Definition 151.187. The agreement hypothesis is the ambient pullback comparison, because the restricted classifier at $(\lambda(c), \eta^{\lambda}_{c})$ is $\sim_{\lambda(c)}$.

Apply Definition 151.188 to the two ambient covers, the ambient span, the two compatible families, their pullback comparison, and the two gluing rows. This gives $g \sim_j h'$ in the ambient sheaf.

Return to the slice below i. The classifier at the restricted target (j, η_j) is $\sim_j$, so the ambient conclusion is exactly the displayed restricted classifier equality. The locality and gluing rows used in this passage are the restricted rows of Definition 151.238 read through the carrier identification. □

Theorem 151.61 (Sheaf restricted-open refinement exactness export)**.** *For every fixed open index $i : \mathsf{OpenIx}$, the restricted-open carrier $\mathcal{S}|_i$ exports the local cover-presentation exactness surface consumed by* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$*. Its rows are:*

1. *restricted carrier, classifier, and restriction laws from Definition 151.236 and Definition 151.237;*
2. *restricted locality and gluing for covers below i from Definition 151.238;*
3. *restricted common-refinement ledgers below i from Definition 151.192;*
4. *pullback compatibility over a restricted common refinement from Definition 151.193;*
5. *classifier equality of restricted gluings whose pulled-back families agree from Definition 151.194.*

The exported surface is local in the fixed open i, uses no host powerset equality of covers, and introduces no quotient comparison between cover presentations.

Proof. The first two rows are already established by the restricted-open surface. Apply Definition 151.236 to identify the carriers, classifiers, and inherited restriction maps, then apply Definition 151.237 and Definition 151.238 for the restriction, locality, and gluing fields.

The third row is the ledger of Definition 151.192. It is exactly the ambient displayed common-refinement span of Definition 151.187, equipped with witnesses that every member remains below the fixed open i.

For compatibility, use Definition 151.193. The proof of that theorem reads the ambient refinement pullback composition row Definition 151.184 through the restricted carrier, so the local consumers see compatible pulled-back families without leaving the slice.

For presentation independence, use Definition 151.194. It is the restricted form of Definition 151.188: two displayed gluings of compatible families with equal pullbacks to the restricted common refinement become classifier-equal at (j, η_j).

Combining these rows gives the local exactness surface required by ringed-space and scheme certificates. Every row is expressed in indexed opens, restricted inclusion witnesses, inherited restrictions, and section classifiers, so the export remains inside the BEDC sheaf certificate surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenRefinement exactness export`

Definition 151.62 (Sheaf BHist triple-overlap route ledger)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist *with membership predicate* OpenAt*, history classifier* hsame*, and finite-intersection operation $\wedge_{\mathsf{Topology}^{\uparrow}}$. Let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. For $a, b, c : A$, write*

$$k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b), \qquad \ell_{a,b,c} := k_{a,b} \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c).$$

The BHist *triple-overlap route ledger of $\mathcal{U}$ is the subledger of Definition 151.160 whose rows are the member rows for $\iota(a)$, $\iota(b)$, and $\iota(c)$, the pair-overlap row $k_{a,b}$, the triple-overlap row $\ell_{a,b,c}$, the displayed projections from $\ell_{a,b,c}$ to each participating member through the available finite-intersection routes, the induced restriction maps into $\mathsf{Sec}(\ell_{a,b,c})$, and any optional* Cont *histories that certify equality of displayed route witnesses. Membership transport along the row is only the visible* hsame *transport on* BHist *cells:*

$$\mathsf{OpenAt}(\ell_{a,b,c}, x) \wedge_{\mathsf{Topology}^{\uparrow}} x \mathsf{hsame} y \Longrightarrow \mathsf{OpenAt}(\ell_{a,b,c}, y).$$

The ledger records route data, restriction targets, section classifiers, and BHist *provenance. It does not add a quotient of cover presentations or a host-level equality between overlap routes.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRouteLedger`

Theorem 151.63 (Sheaf BHist triple-overlap route coherence)**.** *Let $\mathcal{T}$, $\mathcal{S}$, and $\mathcal{U}$ be as in Definition 151.200. For every $a, b, c : A$, every section $s : \mathsf{Sec}(\iota(a))$, and every two displayed routes from $\iota(a)$ to $\ell_{a,b,c}$ in the triple-overlap route ledger, the two induced restrictions have the same classifier at the triple-overlap row:*

$$\rho_{k_{a,b},\ell_{a,b,c}}(\rho_{\iota(a),k_{a,b}}(s)) \sim_{\ell_{a,b,c}} \rho_{\iota(a),\ell_{a,b,c}}(s).$$

The same conclusion holds for the routes beginning at $\iota(b)$ and $\iota(c)$ whenever the corresponding projection route is displayed. Optional Cont *histories compare only the route witnesses; they do not change the target carrier $\mathsf{Sec}(\ell_{a,b,c})$ or the classifier $\sim_{\ell_{a,b,c}}$.*

Proof. Unfold the triple-overlap route ledger. Its routes are the pair-overlap and triple-overlap projection rows already present in the cover nerve ledger of Definition 151.160, with projections read by Definition 151.161.

Apply Definition 151.304 to the displayed chain $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b} \leq_{\mathcal{T}} \iota(a)$. The two-step restriction and the direct displayed restriction land in the same carrier $\mathsf{Sec}(\ell_{a,b,c})$ and are compared by $\sim_{\ell_{a,b,c}}$.

The arguments for $\iota(b)$ and $\iota(c)$ are the same after choosing the displayed finite-intersection projection route from the ledger. If a Cont history is attached, the cited route-stability theorem treats it as route provenance, so the classifier comparison at $\ell_{a,b,c}$ is unchanged. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger triple overlap route coherence`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRoute coherence`

Theorem 151.64 (Sheaf BHist triple-overlap compatible-family descent)**.** *Let $\mathcal{T}$, $\mathcal{S}$, and $\mathcal{U}$ be as in Definition 151.200. If $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$, then for every $a, b, c : A$ the three restrictions of the local family to $\ell_{a,b,c}$ lie in one classifier class:*

$$\rho_{\iota(a),\ell_{a,b,c}}(s_a) \sim_{\ell_{a,b,c}} \rho_{\iota(b),\ell_{a,b,c}}(s_b), \qquad \rho_{\iota(a),\ell_{a,b,c}}(s_a) \sim_{\ell_{a,b,c}} \rho_{\iota(c),\ell_{a,b,c}}(s_c).$$

Thus every compatible family descends from pair-overlap compatibility to a coherent triple-overlap row inside the BHist *cover nerve.*

Proof. Start with pair-overlap compatibility from Definition 151.161. It gives the classifier equality between the restrictions of s_a and s_b on $k_{a,b}$.

Restrict that equality along $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b}$. By Definition 151.304, the transported equality is the same as the direct comparison of the two route restrictions to $\ell_{a,b,c}$. The route-coherence theorem Definition 151.201 identifies the composed and direct displays.

For the comparison with s_c, use the pair-overlap rows for (a, c) and (b, c) and pull them to the displayed triple-overlap row. The finite-intersection projections and route-history clauses are those of Definition 151.200, so all comparisons land in $\sim_{\ell_{a,b,c}}$ on the single carrier $\mathsf{Sec}(\ell_{a,b,c})$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger cover descent exhaustion`

Theorem 151.65 (Sheaf BHist triple-overlap downstream exactness)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback constructions may cite the triple-overlap route ledger as a closed three-face overlap package. For every accepted* BHist *cover row, the package consists of:*

1. *the triple-overlap index $\ell_{a,b,c} = (\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c)$ and its* OpenAt *membership rows, transported only by* hsame*;*

2. the one-step and two-step restriction routes into $\mathsf{Sec}(\ell_{a,b,c})$*, with optional* Cont *histories;*

3. the classifier coherence rows of Definition 151.201 and Definition 151.202;

4. the restricted-open exactness rows consumed below a fixed open.

Consequently a consumer below a fixed open i *can pull the triple-overlap route package through restricted common refinements without reconstructing pairwise compatibility from prose and without adding a host cover quotient.*

Proof. The first three rows are read directly from Definition 151.200, Definition 151.201, and Definition 151.202. They are all internal to BHist cells, indexed opens, restriction maps, and section classifiers.

For passage through a refinement, apply Definition 151.184. It identifies the one-step and two-step descriptions of pulled-back local sections, so the triple-overlap route comparisons survive along displayed refinement towers.

For consumers working below a fixed open, apply Definition 151.195. Its restricted common-refinement surface is the same ambient carrier, restriction, classifier, and gluing data with the additional witness that every index remains below the fixed open.

Combining these with Definition 151.304 gives the advertised downstream package. The conclusion is a reusable exactness row for three-face overlap routes; it introduces no new carrier beyond the existing cover nerve ledger and no comparison other than hsame on histories and $\sim_{\ell_{a,b,c}}$ on sections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRoute downstream exactness`

Definition 151.66 (Sheaf BHist point-germ ledger)**.** *Let* $\mathcal{T}$ *be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *whose point carrier is* BHist*, whose membership row is* OpenAt*, and whose finite-intersection index is written* $\wedge_{\mathsf{Topology}^{\uparrow}}$*. Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over* $\mathcal{T}$*.*

For $b : \mathsf{BHist}$*, a displayed point-germ row over* b *is a triple*

$$(i, o_i, s)$$

where $i : \mathsf{OpenIx}$*,* $o_i : \mathsf{OpenAt}(i, b)$*, and* $s : \mathsf{Sec}(i)$*. Two rows* (i, o_i, s) *and* (j, o_j, t) *compare over* b *when there is a displayed common neighborhood* $k : \mathsf{OpenIx}$ *with*

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad \mathsf{OpenAt}(k, b),$$

and classifier equality after restriction:

$$\rho_{i,k}(s) \sim_k \rho_{j,k}(t).$$

The point-germ ledger is the collection of these rows and comparisons, with common neighborhoods supplied by the finite-intersection rows of $\mathcal{T}$ *and all section comparison performed by the classifiers* $\sim_k$*.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger section scope`

Theorem 151.67 (Sheaf BHist point-germ carrier scope)**.** *The point-germ ledger of Definition 151.231 exposes no carrier beyond the* BHist *point row, the membership row* OpenAt*, indexed opens, finite-intersection rows, presheaf restrictions, and the section classifiers of the indexed sheaf carrier. In particular, each ledger row over* $b : \mathsf{BHist}$ *is determined by an indexed open containing* b *and a section on that open, and each comparison is determined by a displayed common neighborhood below the two opens together with one classifier equality after restriction.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger common open comparison`

Proof. Unfold the displayed point-germ row. Its components are exactly $b : \mathsf{BHist}$, an indexed open i, a membership witness $o_i : \mathsf{OpenAt}(i, b)$, and a section $s : \mathsf{Sec}(i)$.

Unfold the comparison row. It asks only for a common index k below the two source opens, a membership witness $\mathsf{OpenAt}(k, b)$, and the classifier row $\rho_{i,k}(s) \sim_k \rho_{j,k}(t)$.

The existence of a canonical common-neighborhood shape is supplied by the finite-intersection field of Definition 88.1, while restriction maps and section classifiers are exactly the rows of Definition 151.154. No host stalk, quotient carrier, or powerset equality is used by either the row data or the comparison data. □

Theorem 151.68 (Sheaf BHist point-germ restriction readback)**.** *Let (i, o_i, s) be a displayed point-germ row over $b : \mathsf{BHist}$ in the sense of Definition 151.231. If $j \leq_{\mathcal{T}} i$ and $o_j : \mathsf{OpenAt}(j, b)$, then the restricted row*

$$(j, o_j, \rho_{i,j}(s))$$

compares to the original row over the common neighborhood j. Explicitly, the comparison is witnessed by $j \leq_{\mathcal{T}} i$, the identity inclusion $j \leq_{\mathcal{T}} j$, the membership row o_j, and the identity classifier row

$$\rho_{j,j}(\rho_{i,j}(s)) \sim_j \rho_{i,j}(s).$$

Proof. The hypotheses give the restricted point-germ row: the target open is j, the visible membership witness is o_j, and the section is the ambient restriction $\rho_{i,j}(s)$.

Use j itself as the displayed common neighborhood. One projection is the given inclusion $j \leq_{\mathcal{T}} i$, the other is the identity inclusion $j \leq_{\mathcal{T}} j$, and the point witness is o_j.

It remains only to supply the classifier equality required by Definition 151.231. The identity field of Definition 151.154 applied to the section $\rho_{i,j}(s) : \mathsf{Sec}(j)$ gives $\rho_{j,j}(\rho_{i,j}(s)) \sim_j \rho_{i,j}(s)$, which is exactly the comparison row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restriction readback`

Theorem 151.69 (Sheaf BHist point-germ gluing readback)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$ for an indexed sheaf carrier $\mathcal{S}$ over a BHist-scoped topology carrier $\mathcal{T}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ is a gluing section with*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

For every $b : \mathsf{BHist}$ with $o_i : \mathsf{OpenAt}(i, b)$, the cover law supplies $a : A$ and $o_a : \mathsf{OpenAt}(\iota(a), b)$, and the point-germ row (i, o_i, g) reads back through that member:

$$(i, o_i, g)$$

compares over b to

$$(\iota(a), o_a, s_a).$$

If g' is another gluing section for the same compatible local family, then locality gives $g \sim_i g'$, so the resulting point-germ readback is unique up to the same section classifier and its restrictions to displayed common neighborhoods.

Proof. Apply the cover law γ to b and o_i. It returns a cover member $a : A$ together with $o_a : \mathsf{OpenAt}(\iota(a), b)$.

Use $\iota(a)$ as the common neighborhood for the two point-germ rows. The projection $\iota(a) \leq_{\mathcal{T}} i$ is the cover inclusion h_a, the projection $\iota(a) \leq_{\mathcal{T}} \iota(a)$ is the identity inclusion, and the membership witness is o_a.

The gluing row gives $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$. Combining this with the identity restriction row $\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a$, using symmetry and transitivity of the classifier on $\mathsf{Sec}(\iota(a))$, gives the comparison required by Definition 151.231.

For uniqueness, let g' be another gluing of the same compatible family. Apply Definition 151.169 to the two displayed gluing rows to obtain $g \sim_i g'$. The restriction classifier-stability field of Definition 151.154 transports this equality to any displayed common neighborhood used in the point-germ ledger, so the readback does not depend on the chosen gluing representative beyond the existing section classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger gluing readback`

Theorem 151.70 (Sheaf BHist point-germ comparison transitivity)**.** *Let (i, o_i, s), (j, o_j, t), and (ℓ, o_ℓ, u) be displayed point-germ rows over the same $b : \mathsf{BHist}$ in the sense of Definition 151.231. Suppose the first two rows compare on a displayed common neighborhood k:*

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad o_k : \mathsf{OpenAt}(k, b),$$

with

$$\alpha : \rho_{i,k}(s) \sim_k \rho_{j,k}(t),$$

and suppose the last two rows compare on a displayed common neighborhood m:

$$m \leq_{\mathcal{T}} j, \qquad m \leq_{\mathcal{T}} \ell, \qquad o_m : \mathsf{OpenAt}(m, b),$$

with

$$\beta : \rho_{j,m}(t) \sim_m \rho_{\ell,m}(u).$$

Put

$$n := k \wedge_{\mathsf{Topology}^{\uparrow}} m.$$

Then n is a displayed common neighborhood below i and ℓ at b, and the first and third rows compare there:

$$\rho_{i,n}(s) \sim_n \rho_{\ell,n}(u).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison trans`
↪ variant `BEDC.Derived.SheafUp.SheafBHistPointGermLedger shared open classifier transitivity`

Proof. The finite-intersection field of Definition 88.1 gives, for every $x : \mathsf{BHist}$,

$$\mathsf{OpenAt}(n, x) \iff (\mathsf{OpenAt}(k, x) \wedge \mathsf{OpenAt}(m, x)).$$

The forward implication, followed by projection, gives inclusions $n \leq_{\mathcal{T}} k$ and $n \leq_{\mathcal{T}} m$. At $x = b$, the reverse implication applied to o_k and o_m gives $o_n : \mathsf{OpenAt}(n, b)$. Composing the pointwise inclusions gives $n \leq_{\mathcal{T}} i$ and $n \leq_{\mathcal{T}} \ell$; composing with $k \leq_{\mathcal{T}} j$ and with $m \leq_{\mathcal{T}} j$ gives the two chains to j used below.

Restrict the comparison row α along $n \leq_{\mathcal{T}} k$ by the classifier-stability field of Definition 151.154. This gives

$$\rho_{k,n}(\rho_{i,k}(s)) \sim_n \rho_{k,n}(\rho_{j,k}(t)).$$

The composition field of the same presheaf carrier, applied to $n \leq_{\mathcal{T}} k \leq_{\mathcal{T}} i$ and to $n \leq_{\mathcal{T}} k \leq_{\mathcal{T}} j$, gives

$$\rho_{k,n}(\rho_{i,k}(s)) \sim_n \rho_{i,n}(s)$$

and

$$\rho_{k,n}(\rho_{j,k}(t)) \sim_n \rho_{j,n}(t).$$

Symmetry and transitivity of the classifier $\sim_n$ therefore yield

$$\rho_{i,n}(s) \sim_n \rho_{j,n}(t).$$

Restrict the comparison row β along $n \leq_{\mathcal{T}} m$. The classifier-stability field gives

$$\rho_{m,n}(\rho_{j,m}(t)) \sim_n \rho_{m,n}(\rho_{\ell,m}(u)).$$

The composition field for $n \leq_{\mathcal{T}} m \leq_{\mathcal{T}} j$ and $n \leq_{\mathcal{T}} m \leq_{\mathcal{T}} \ell$ gives

$$\rho_{m,n}(\rho_{j,m}(t)) \sim_n \rho_{j,n}(t)$$

and

$$\rho_{m,n}(\rho_{\ell,m}(u)) \sim_n \rho_{\ell,n}(u).$$

Again using symmetry and transitivity of $\sim_n$, obtain

$$\rho_{j,n}(t) \sim_n \rho_{\ell,n}(u).$$

Transitivity of $\sim_n$ combines the last displayed row with $\rho_{i,n}(s) \sim_n \rho_{j,n}(t)$ and gives

$$\rho_{i,n}(s) \sim_n \rho_{\ell,n}(u).$$

Together with the witnesses $n \leq_{\mathcal{T}} i$, $n \leq_{\mathcal{T}} \ell$, and $o_n : \mathsf{OpenAt}(n, b)$, this is exactly the point-germ comparison row required by Definition 151.231. □

Definition 151.71 (Sheaf root stalk-free point-germ source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root stalk-free point-germ source is the following ten-field source over* BHist.

1. *The topology field is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier of Definition 88.1, with open indices, membership rows* $\mathsf{OpenAt}(i, b)$*, inclusion rows* $j \leq_{\mathcal{T}} i$*, identity inclusions, and finite-intersection rows* $i \wedge_{\mathsf{Topology}^{\uparrow}} j$.
2. *The presheaf field is the* $\mathsf{PreSheaf}^{\uparrow}$ *indexed section carrier of Definition 151.154, with section carriers* $\mathsf{Sec}(i)$*, restriction rows* $\rho_{i,j}$*, restriction identity, restriction composition, and restriction classifier-stability.*
3. *The sheaf field is the* $\mathsf{Sheaf}^{\uparrow}$ *locality and gluing surface over the same indexed topology and presheaf carrier.*
4. *The naming field is the root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation surface, read from the carrier, classifier, locality, gluing, refinement, and exactness rows already exposed by the indexed sheaf carrier.*
5. *The history field is the point carrier* BHist *together with* hsame*-stable transport of displayed membership rows.*
6. *The point-germ ledger field is the displayed ledger of Definition 151.231; a row over* $b : \mathsf{BHist}$ *is a triple* (i, o_i, s) *with* $o_i : \mathsf{OpenAt}(i, b)$ *and* $s : \mathsf{Sec}(i)$.

7. *The common-neighborhood field is supplied by finite intersections in the topology carrier, including the projections from $i \wedge_{\mathsf{Topology}^\uparrow} j$ to i and j and the displayed membership witness obtained from the two input membership rows.*
8. *The restriction-comparison field compares two point-germ rows only after restricting their sections to a displayed common neighborhood and applying the section classifier $\sim_k$ there.*
9. *The classifier-law field uses the reflexivity, symmetry, and transitivity rows of the section classifier on each $\mathsf{Sec}(k)$, together with restriction identity and restriction composition, to move comparisons across displayed neighborhoods.*
10. *The boundary field exports no host point carrier, no host stalk quotient, no quotient of section representatives, and no powerset equality of opens; all point-germ evidence remains inside indexed opens,* BHist *membership, restrictions, finite intersections, section classifiers, and* hsame *transport.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootStalkFreeGermSource`

Theorem 151.72 (Sheaf root point-germ classifier reflexivity)**.** *Every displayed point-germ row in the source of Definition 151.286 compares to itself. Concretely, if (i, o_i, s) is a row over $b : \mathsf{BHist}$ in the sense of Definition 151.231, then the comparison from (i, o_i, s) to (i, o_i, s) is witnessed by the common neighborhood i, the identity inclusions $i \leq_{\mathcal{T}} i$, the membership row o_i, and the identity classifier row*

$$\rho_{i,i}(s) \sim_i \rho_{i,i}(s).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison reflexive`

Proof. Use the row's own indexed open as the displayed common neighborhood. The two projections required by Definition 151.231 are both the identity inclusion $i \leq_{\mathcal{T}} i$, and the membership witness is the row's own witness $o_i : \mathsf{OpenAt}(i, b)$.

It remains to fill the classifier comparison after restriction. The identity restriction field of Definition 151.154 identifies $\rho_{i,i}(s)$ with s in the section classifier on $\mathsf{Sec}(i)$, and the reflexivity field of that classifier gives $\rho_{i,i}(s) \sim_i \rho_{i,i}(s)$. This is exactly the displayed comparison row. □

Theorem 151.73 (Sheaf root point-germ classifier symmetry)**.** *The displayed point-germ comparison in the source of Definition 151.286 is symmetric. If rows (i, o_i, s) and (j, o_j, t) over the same $b : \mathsf{BHist}$ compare on a displayed common neighborhood k with projections*

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad o_k : \mathsf{OpenAt}(k, b),$$

and classifier row

$$\rho_{i,k}(s) \sim_k \rho_{j,k}(t),$$

then the reverse comparison from (j, o_j, t) to (i, o_i, s) is witnessed by the same k, the projections swapped, the same membership row o_k, and the symmetric classifier row

$$\rho_{j,k}(t) \sim_k \rho_{i,k}(s).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison symmetric fields`

Proof. Keep the displayed common neighborhood k. The comparison definition asks only for a neighborhood below the two source opens and a membership row at b, so the same witness o_k remains valid and the two projections are read in the opposite order.

The only row whose direction changes is the section classifier comparison on $\mathsf{Sec}(k)$. Apply the symmetry field of the classifier supplied by Definition 151.154 to the given row $\rho_{i,k}(s) \sim_k \rho_{j,k}(t)$. The result is $\rho_{j,k}(t) \sim_k \rho_{i,k}(s)$, so the swapped neighborhood data form the reverse point-germ comparison. □

Theorem 151.74 (Sheaf root point-germ classifier equivalence boundary)**.** *On the displayed point-germ rows of Definition 151.286, the comparison relation of Definition 151.231 is an equivalence relation. Reflexivity is supplied by Definition 151.287, symmetry by Definition 151.288, and transitivity by Definition 151.235. The resulting equivalence boundary exports only displayed rows and comparison witnesses over indexed neighborhoods; it exports no stalk quotient, no host point carrier, and no quotient carrier of point-germ classes.*

Proof. For reflexivity, apply Definition 151.287 to the displayed row. The witness is its own indexed open, with identity restriction and the section classifier's reflexivity row.

For symmetry, apply Definition 151.288 to any displayed comparison witness. This keeps the same common neighborhood and reverses only the two projection slots and the section classifier row.

For transitivity, take two composable displayed comparison witnesses between rows over the same $b : \mathsf{BHist}$. Apply Definition 151.235; its finite-intersection construction supplies the refined common neighborhood and composes the restricted classifier rows.

These three clauses are stated entirely in the ledger language of Definition 151.231. Since Definition 151.286 contains no host stalk carrier and no quotient operation, the equivalence relation is the boundary relation available to downstream consumers, not a construction of stalk elements. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison common transport`

Theorem 151.75 (Sheaf root cover locality and gluing obligation)**.** *Let $\mathcal{T}$ be an indexed-open carrier for $\mathsf{Topology}^{\uparrow}$, and let $\mathcal{S}$ be the indexed section presheaf carrier over $\mathcal{T}$ supplied by the $\mathsf{PreSheaf}^{\uparrow}$ restriction surface. Then the indexed sheaf carrier of Definition 151.154, with classifier family $\sim_i$ and indexed covers as in Definition 151.155, supplies the carrier, classifier, locality, and gluing rows required by the root $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ ten-field template of Definition 151.190.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverLocalityGluing obligation surface`

Proof. Unfold the indexed section presheaf carrier. Its carrier row is the family $\mathsf{Sec}(i)$ over open indices $i : \mathsf{OpenIx}$, and its restriction row is the stable transport $\rho_{i,j}$ along displayed inclusions $j \leq_{\mathcal{T}} i$.

Unfold the indexed cover compatibility and locality surface. An indexed cover records member opens, cover inclusions, and finite-intersection overlap rows, so a compatible family is already expressed entirely in the $\mathsf{Topology}^{\uparrow}$ and $\mathsf{PreSheaf}^{\uparrow}$ ledgers.

The sheaf fields then provide exactly the two root rows needed by the certificate: locality detects equality of global sections from equality after restriction to every cover member, and gluing supplies a global section for every compatible local family. These are the carrier, classifier, locality, and gluing entries named in the root template row of Definition 151.190. □

Theorem 151.76 (Sheaf root restriction refinement obligation)**.** *For an indexed sheaf carrier over $\mathcal{T}$, every displayed refinement of indexed covers preserves compatible families and transports gluing rows through restriction. Hence the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root surface may replace a cover by any displayed refinement without changing the local-family compatibility ledger or the global gluing classifier row.*

Proof. Start with a displayed refinement of indexed covers. The refinement obligation of Definition 151.174 records the member map and the inclusion rows that route each refined open into the original cover member.

Apply Definition 151.175. A compatible family on the original indexed cover pulls back along the displayed refinement to a compatible family on the refined cover, because every overlap comparison is transported by the restriction composition and classifier-transport rows.

Apply Definition 151.176. If a global section glues the original compatible family, then the same global section glues the pulled-back family after the restriction transport supplied by the refinement. Thus the root certificate can consume either cover presentation with the same global classifier class. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRestrictionRefinement obligation`

Theorem 151.77 (Sheaf root common-refinement exactness obligation)**.** *Let two indexed cover presentations of the same open carry compatible local families. If the families agree after pullback to a displayed common refinement span, then their displayed gluings determine the same global section classifier class. This exactness uses only indexed cover data, displayed common refinement, and restriction transport; it does not quotient cover presentations.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCommonRefinement exactness obligation`
↪ variant `BEDC.Derived.SheafUp.SheafRootCommonRefinement obligation carrier exactness`

Proof. Unfold the displayed common-refinement span of Definition 151.186. The span consists of one indexed cover together with two displayed refinement projections into the two cover presentations being compared.

By Definition 151.187, the span yields the two refinement maps needed to pull both local families to the common cover. The hypothesis says that these two pulled-back families are classifier-equal at every member of that common cover.

Apply Definition 151.188. The two global sections, each viewed as a gluing after descent along its refinement projection, are equal in the classifier $\sim_i$ of the base open.

Finally apply Definition 151.189. The conclusion is an equality of global section classifier classes for the two cover presentations themselves, with no host open-cover equality and no quotient of cover presentations introduced into the root surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCommonRefinement route exactness`

Definition 151.78 (Sheaf root NameCert field inventory)**.** *The root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *field inventory is the following ten-field surface.*

1. *The traditional target is a sheaf of sections over indexed opens, read over accepted indexed covers rather than host open-cover equality.*
2. *The BEDC source is* $\mathsf{Topology}^{\uparrow}$ *together with the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction surface.*
3. *The carrier row is the indexed section family* $\mathsf{Sec}(i)$ *of Definition 151.154.*
4. *The classifier row is the open-indexed section classifier family* $\sim_i$*.*

5. *The carrier-transport stability row keeps section carriers and classifiers aligned under the displayed indexed-open and* BHist *transport rows.*
6. *The restriction-stability row consists of restriction identity, restriction composition, and classifier preservation by restriction.*
7. *The locality row detects global classifier equality from memberwise classifier equality on an indexed cover.*
8. *The gluing row supplies a global section readback for every compatible local family.*
9. *The refinement-exactness row transports compatible families and gluing witnesses through displayed refinements and common refinements.*
10. *The ledger and presentation row consists of indexed covers, common-refinement spans, and cover-presentation invariance; it carries no standard bridge.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootNameCertFieldInventory`

Theorem 151.79 (Sheaf root NameCert carrier and classifier fields)**.** *Let $\mathcal{T}$ be an indexed-open carrier for* $\mathsf{Topology}^{\uparrow}$*, and let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$. Then the carrier, classifier, carrier-transport, and restriction-stability fields of Definition 151.279 are supplied by Definition 151.154, Definition 151.155, Definition 151.157, Definition 151.158, and Definition 151.164.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertCarrierClassifier fields`

Proof. Project the indexed section carrier from $\mathcal{S}$. By Definition 151.154, the visible carrier rows are the section families $\mathsf{Sec}(i)$, and the classifier rows are the relations $\sim_i$ with their equivalence fields.

Apply Definition 151.157. It exhausts the carrier support by indexed sections, section classifiers, restriction maps, restriction identity and composition rows, indexed covers, compatibility predicates, locality, and gluing. Thus the root inventory has no additional carrier primitive to construct.

Apply Definition 151.158 and Definition 151.164. These provide classifier transport along restrictions, identity restriction, composition of two displayed restriction routes, and stability on finite-intersection rows supplied by $\mathsf{Topology}^{\uparrow}$.

Finally read the cover-facing source through Definition 151.155. It contributes only indexed covers and compatible local-family predicates, so the projected carrier and classifier fields repack exactly as the carrier, classifier, carrier-transport, and restriction-stability fields of the root inventory. □

Theorem 151.80 (Sheaf root NameCert locality and gluing fields)**.** *For an indexed sheaf carrier over $\mathcal{T}$, the locality and gluing fields of Definition 151.279 are supplied by Definition 151.283, Definition 151.169, and Definition 151.170.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertLocalityGluing fields`

Proof. Begin with an indexed cover $\mathcal{U} = (A, \iota, h, \gamma)$ of an open index i and a compatible local family $s_a : \mathsf{Sec}(\iota(a))$. The cover, overlap, and compatibility rows are exactly the rows of Definition 151.155.

Apply Definition 151.283. It identifies locality and gluing as root $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ rows over the same indexed cover and the same classifier family $\sim_i$.

The uniqueness side is then the projection of Definition 151.169. Two global sections that display the same compatible local family are equal in the global classifier because locality detects equality from all cover members.

The readback side is supplied by Definition 151.170. Restricting a global section to the members of an indexed cover produces a compatible local family, so the gluing row and the locality row are closed under the named cover ledger. □

Theorem 151.81 (Sheaf root NameCert refinement exactness fields)**.** *For an indexed sheaf carrier over $\mathcal{T}$, the refinement, common-refinement, and cover-presentation exactness fields of Definition 151.279 are supplied by Definition 151.284, Definition 151.285, Definition 151.189, and Definition 151.293.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertRefinementExactness fields`

Proof. Start with a displayed refinement of indexed covers. Apply Definition 151.284; it transports compatible local families and gluing rows along the refinement map using only displayed inclusion routes and restriction stability.

For two cover presentations, pass to a displayed common-refinement span. Apply Definition 151.285; agreement after pullback to the common cover identifies the two global gluing classifier classes.

Apply Definition 151.189. This repacks the common-refinement comparison as invariance of the cover presentation itself, without adding host cover equality or a quotient of cover presentations.

Finally apply Definition 151.293. The route-exactness row records the same inclusion and restriction-route comparisons as NameCert-consumable ledger data, so the refinement and presentation fields close inside the root inventory of Definition 151.279. □

151.5 Root obligations for downstream certificates

Definition 151.82 (SheafUp root-unblock BEDC source)**.** *The root-unblock source for* $\mathsf{Sheaf}^{\uparrow}$ *is the ten-field certificate surface consumed by ringed-space, scheme, and restricted-open sheaf constructions.*

1. *The traditional target is sheaves of sections over indexed opens.*
2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* NameCert.
3. *The history ledger is the* BHist *cover-nerve ledger of Definition 151.160.*
4. *The carrier is the indexed section presheaf carrier of Definition 151.154, equipped with the locality and gluing fields of Definition 151.155.*
5. *The classifier family is the open-indexed relation* $\sim_i$ *on* $\mathsf{Sec}(i)$.
6. *The cover ledger records cover indices* A, *cover-member maps* $\iota : A \to \mathsf{OpenIx}$, *and pointwise cover rows over* BHist.
7. *The overlap ledger records finite intersections* $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(a')$ *and their projection rows.*
8. *The restriction ledger records routes from a target open to cover members and overlaps, with identity, composition, and classifier-stability rows.*
9. *Optional route histories are recorded by* Cont *rows, and history-cell membership is transported only by* hsame.

10. *The surface contains no quotient of cover presentations and no equality of host powersets of opens.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource member point germ`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource cover membership`

Theorem 151.83 (Sheaf root-unblock cover membership obligation)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an accepted cover row of an open index i over a* BHist *topology ledger. Then every displayed history membership in the covered open has a displayed cover-member witness:*

$$\forall b : \mathsf{BHist},\ \mathsf{OpenAt}(i, b) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), b).$$

All overlap information used downstream is recorded only through finite-intersection rows and hsame*-stable* BHist *membership, not through host powerset equality.*

Proof. Unfold the accepted cover row. Its pointwise cover component is exactly the displayed implication from $\mathsf{OpenAt}(i, b)$ to a cover index $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$.

For overlap rows, use the finite-intersection field of the indexed topology carrier as in Definition 151.161. It supplies the overlap index and the two projection rows back to the participating cover members.

History transport is then read from the BHist cover-nerve ledger of Definition 151.160. That ledger admits only hsame-stable membership transport, so the exported cover-membership row stays internal to indexed opens and history cells. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockCoverMembership obligation`

Theorem 151.84 (Sheaf root-unblock restriction stability obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Restriction identity, restriction composition, stable restriction rows, and* Cont *associativity for route histories preserve the target section classifier along one-step and two-step restriction routes. Concretely, for $j \leq_{\mathcal{T}} i$ and $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$, the rows*

$$\rho_{i,i}(s) \sim_i s, \qquad s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t), \qquad \rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

are the classifier-preserving route rows exposed to downstream certificates.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockRestrictionStability semantic name certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock restriction stability rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route history target classifier stability`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock restriction stability surface`

Proof. The identity and composition rows are fields of the indexed section presheaf carrier in Definition 151.154. The one-step classifier preservation row is the stable restriction row of Definition 151.156, also cited as Definition 151.172.

For a two-step route, first use restriction classifier transport along $j \leq_{\mathcal{T}} i$, then compose with the route from j to k. The displayed composition row identifies this two-step restriction with the direct restriction from i to k in the classifier $\sim_k$.

If the ledger also carries Cont histories for the same routes, their associativity only compares displayed route histories. It does not alter the open index k or the classifier $\sim_k$, so it preserves the same target section classifier consumed by the certificate. □

Theorem 151.85 (Sheaf root-unblock locality and gluing obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i. Compatible local families with displayed gluings have a* NameCert*-consumable exactness row: locality gives global classifier equality for any two global readbacks whose restrictions agree on the cover, and gluing supplies existence of a global readback without quotienting cover presentations.*

Proof. The existence half is the gluing field in Definition 151.155. Given a compatible local family, it supplies a section $g : \mathsf{Sec}(i)$ together with displayed restriction rows to each $\iota(a)$.

The uniqueness half is locality. If two global readbacks restrict to classifier-equal sections on every cover member, the locality field returns equality in the global classifier $\sim_i$; Definition 151.169 packages the same argument for two displayed gluings of one compatible family.

Refinement and presentation changes are handled by the existing refinement rows Definition 151.176, Definition 151.179, Definition 151.188, and Definition 151.189. Each row uses displayed cover maps, common refinements, restriction stability, Cont route histories when present, and hsame-stable membership; none introduces a quotient of cover presentations. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock locality gluing faces`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead locality gluing refinement inversion`

Definition 151.86 (Sheaf root cover-nerve export surface)**.** *The root cover-nerve export surface for* $\mathsf{Sheaf}^{\uparrow}$ *is the three-row* BHist*-scoped certificate surface consumed by downstream chapters.*

1. *The traditional target is sheaves over indexed opens, read through section carriers on a cover nerve rather than through host open covers.*
2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* NameCert.
3. *The carrier is the family of indexed sections over* BHist *opens, using* $\mathsf{Sec}(i)$ *from Definition 151.154 for each open index i.*
4. *The classifier family is the displayed section relation $\sim_i$ on* $\mathsf{Sec}(i)$.
5. *The stability rows are restriction identity, restriction composition, classifier transport, and optional* Cont *route-history transport.*
6. *The ledger rows are cover-member cells, pair-overlap cells, and route-history cells in the* BHist *cover nerve of Definition 151.160.*
7. *The exactness rows are cover membership, restriction route stability, and locality/gluing.*
8. *The cover-membership row sends every accepted* $\mathsf{OpenAt}(i, b)$ *cell to a displayed cover member and its pair-overlap ledger context.*
9. *The downstream export is exactly the rows listed here:* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers receive no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*
10. *There is no standard bridge in this surface.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootCoverNerveExportSurface`

Theorem 151.87 (Sheaf root cover-nerve membership exhaustion)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an accepted cover row of an indexed open i over the* BHist *topology ledger. Every displayed cell* $\mathsf{OpenAt}(i, b)$ *in this accepted cover has a cover-member witness and a pair-overlap row inside the* BHist *cover nerve:*

$$\forall b : \mathsf{BHist},\ \mathsf{OpenAt}(i, b) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), b)$$

and for every second member $a' : A$ the ledger also contains the finite-intersection row for $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(a')$ with its two projections to the participating members.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve membership exhaustion`

Proof. Apply the cover-membership obligation of Definition 151.299. It reads the accepted cover row as a BHist-indexed implication from an ambient displayed cell to a cover-member witness, so every $\mathsf{OpenAt}(i, b)$ cell is exhausted by a displayed member $\iota(a)$.

The pair-overlap row is then the overlap projection surface of Definition 151.161, unpacked inside the cover-nerve ledger of Definition 151.160. For each pair $a, a' : A$, the finite-intersection row supplies the overlap index and the two projections back to $\iota(a)$ and $\iota(a')$.

Because the ledger transports membership only by hsame, this exhaustion stays scoped to BHist cells and indexed opens. No host cover equality or quotient of cover presentations is introduced. □

Theorem 151.88 (Sheaf root cover-nerve restriction route stability)**.** *Let $\mathcal{S}$ be the indexed section carrier of Definition 151.154 over an indexed open topology carrier. One-step and two-step restrictions preserve section classifiers:*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t), \qquad \rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

for every displayed chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. Optional Cont *route histories compare route witnesses only and do not alter the target classifier $\sim_j$ or $\sim_k$.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger route hsame transport`

Proof. The first displayed row is the restriction classifier-transport field of Definition 151.154. It is also the one-step stability clause isolated by Definition 151.300.

For the two-step row, apply the composition field of the same carrier to the chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. The comparison lands in $\sim_k$, so the direct restriction and the composed restriction have the same target classifier.

If a Cont history is present, use the route-history part of Definition 151.300. Those histories record associativity of displayed routes; they do not change the open index of the restricted section, hence cannot change the classifier family consumed downstream. □

Theorem 151.89 (Sheaf root cover-nerve downstream coverage)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback constructions receive exactly the rows of Definition 151.302: cover-member cells, pair-overlap cells, one-step and two-step restriction stability rows, optional* Cont *route histories, and locality/gluing exactness rows. They receive no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Proof. The carrier part is fixed by Definition 151.154: downstream chapters can read only indexed section carriers, section classifiers, restrictions, and the identity, composition, and classifier-transport rows attached to them.

The cover part is fixed by Definition 151.160 and Definition 151.303. These provide cover-member cells and pair-overlap cells over BHist, with overlap projections supplied by Definition 151.161.

The exactness part is the union of the three root obligations. Cover membership comes from Definition 151.299; restriction route stability comes from Definition 151.300; locality and gluing come from Definition 151.301.

Those cited rows are precisely the rows listed in Definition 151.302. Since none of them introduces host cover equality, a cover quotient, a stalk quotient, or an external sheafification operation, no downstream consumer obtains such a row from this export surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve downstream coverage`

Theorem 151.90 (Sheaf root cover-nerve exact globalization package)**.** *The root cover-nerve export surface of Definition 151.302 forms an* ExactGlobalizeBase*-style package over the indexed-open* BHist *cover nerve. Its coverage face is Definition 151.303; its base soundness face is Definition 151.304; its base completeness and exactness face is the locality-gluing row of Definition 151.301. Hence* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *may consume one classifier-exact cover-nerve package, with no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve exact globalize package`

Proof. Unpack Definition 151.302. The carrier is the indexed family $\mathsf{Sec}(i)$ over BHist opens, the classifier is the displayed relation $\sim_i$, and the ledger is the cover nerve of Definition 151.160.

The coverage face is exactly Definition 151.303. It sends each displayed $\mathsf{OpenAt}(i, b)$ cell to a cover member and records the pair-overlap rows needed by downstream restriction routes.

The base soundness face is Definition 151.304. One-step restriction, two-step restriction, classifier transport, and optional Cont route histories preserve the section classifier on the target indexed open.

The base completeness and exactness face is Definition 151.301. Gluing constructs the global readback from compatible local rows, and locality identifies any two global readbacks whose restrictions agree on the accepted cover.

These are precisely the coverage, soundness, and completeness fields required by the exact-globalization constructor of Definition 9199.16, with the relation field constrained as in Definition 9199.11. Repacking the three displayed faces therefore gives the single classifier-exact cover-nerve package consumed by the downstream chapters, and Definition 151.305 fixes that no quotient or external sheafification row is exported. □

151.6 Identity-cover root package

Definition 151.91 (Sheaf identity-cover root package)**.** *For an indexed open i in the* $\mathsf{Sheaf}^{\uparrow}$ *root-obligation surface, the identity-cover root package is the accepted one-member cover of i with cover index type* Unit *and member map $\iota : \mathsf{Unit} \to \mathsf{OpenIx}$ constantly equal to i. Its membership row is the identity row on $\mathsf{OpenAt}(i, b)$ for each $b : \mathsf{BHist}$, its only overlap row is the finite-intersection cell $i \wedge_{\mathsf{Topology}^{\uparrow}} i$, and its exposed restriction route is the identity restriction $\rho_{i,i}$.*

The package ledger is restricted to BHist *membership cells,* hsame *transport of those cells, the self-overlap projections for $i \wedge_{\mathsf{Topology}^{\uparrow}} i$, and the presheaf identity row identifying $\rho_{i,i}(s)$ with s in the classifier $\sim_i$. It carries no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafIdentityCoverRootPackage`

Theorem 151.92 (Sheaf identity-cover membership exactness)**.** *Let i be an indexed open and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of Definition 151.227. For every $b : \mathsf{BHist}$, the displayed membership row is exact:*

$$\mathsf{OpenAt}(i, b) \iff \exists a : \mathsf{Unit},\ \mathsf{OpenAt}(\iota(a), b).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger identity open membership cont exactness`

Proof. For the forward direction, use the unique member of Unit. Since ι is constantly i, the required displayed cover-member cell is the original cell $\mathsf{OpenAt}(i, b)$ read through the identity membership row.

For the reverse direction, destruct the displayed witness $a : \mathsf{Unit}$. The member map again reads as i, so the cover-member cell is exactly a cell of $\mathsf{OpenAt}(i, b)$.

Both directions stay inside the BHist ledger and use only hsame transport when a history cell is reindexed. No equality of ambient host covers is used. □

Theorem 151.93 (Sheaf identity-cover restriction readback)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier, and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of an indexed open i. For every section $s : \mathsf{Sec}(i)$, the restriction route exposed by $\mathcal{U}_{\mathrm{id}}$ is classifier-equal to $\rho_{i,i}(s)$ and hence classifier-equal to s by the presheaf identity row:*

$$\rho^{\mathcal{U}_{\mathrm{id}}}(s) \sim_i \rho_{i,i}(s), \qquad \rho_{i,i}(s) \sim_i s.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIdentityCover restriction readback`

Proof. Unpack the identity-cover root package. Its only cover member is i, and the route from the ambient open to that member is the identity route, so the exposed restriction readback is the displayed route $\rho_{i,i}(s)$.

Apply the identity row of the indexed section presheaf carrier from Definition 151.154. That row identifies $\rho_{i,i}(s)$ with s in the classifier $\sim_i$.

The self-overlap cell $i \wedge_{\mathsf{Topology}^{\uparrow}} i$ supplies no additional restriction target; it only records the overlap ledger row already attached to the identity route. □

Theorem 151.94 (Sheaf identity-cover gluing collapse)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier, and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of an indexed open i. A compatible local family on $\mathcal{U}_{\mathrm{id}}$ is classifier-equivalent to exactly one global section over i: gluing supplies a global readback, and locality identifies any two global readbacks whose identity-cover restrictions agree.*

Proof. First apply the gluing part of Definition 151.301 to the compatible local family over $\mathcal{U}_{\mathrm{id}}$. This produces a global section $g : \mathsf{Sec}(i)$ whose restriction to the unique cover member agrees with the displayed local section.

The agreement row is read back through Definition 151.229. Because the only exposed route is $\rho_{i,i}$ and the presheaf identity row identifies $\rho_{i,i}(g)$ with g, the local family has no separate cover presentation beyond its single global readback.

For uniqueness, take two global readbacks whose restrictions agree on the identity cover. The membership exactness row of Definition 151.228 reduces coverwise agreement to agreement on the lone Unit member, and Definition 151.229 turns that agreement into agreement of the two global sections over i.

Apply the locality part of Definition 151.301. It returns classifier equality in $\sim_i$, so the compatible local family collapses to one global section up to the section classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIdentityCover gluing collapse`

Definition 151.95 (Sheaf downstream root source)**.** *The downstream root source for $\mathsf{Sheaf}^{\uparrow}$ is the displayed certificate surface obtained by taking the indexed-open carrier of $\mathsf{Topology}^{\uparrow}$, the restriction carrier of $\mathsf{PreSheaf}^{\uparrow}$, the locality and gluing surface of $\mathsf{Sheaf}^{\uparrow}$, the cover-nerve ledger of*

Definition 151.160, the point-germ ledger of Definition 151.231, and the refinement exactness rows of Definition 151.179 and Definition 151.189.

This source exposes only indexed opens, BHist *membership rows, restriction maps, section classifiers, compatible local-family predicates, point-germ comparisons, displayed refinements, and cover-presentation exactness. It contains no host powerset equality and no quotient of cover presentations.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDownstreamRootSource`

Theorem 151.96 (Sheaf RingedSpace local ring row)**.** *Let $\mathcal{S}$ be an indexed* $\mathsf{Sheaf}^{\uparrow}$ *carrier over an indexed-open* $\mathsf{Topology}^{\uparrow}$ *carrier $\mathcal{T}$. Suppose each section carrier* $\mathsf{Sec}(i)$ *carries the* $\mathsf{CommRing}^{\uparrow}$ *operation rows required for a local-ring row, and suppose every restriction map $\rho_{i,j}$ preserves addition, multiplication, zero, one, unit detection, maximal-ideal detection, and the section classifier. Then the source of Definition 151.223 exposes the structure-sheaf rows consumed by* $\mathsf{RingedSpace}^{\uparrow}$*: open-indexed commutative ring sections, classifier-stable restrictions, pointwise local-ring readbacks through the point-germ ledger, and local compatibility of the section classifier.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRingedSpaceLocalRing row`

Proof. Project the downstream source of Definition 151.223. Its carrier rows are the indexed sections, classifiers, restrictions, locality and gluing fields, cover ledger, point-germ ledger, and refinement exactness rows already supplied by $\mathsf{Sheaf}^{\uparrow}$.

Use Definition 151.300 for restriction stability. The assumed operation preservation rows ride on the same restriction routes, so the ring operations and the section classifier are transported together along every inclusion $j \leq_{\mathcal{T}} i$.

For local-ring readback, use the point-germ ledger of Definition 151.231. A row (i, o_i, s) records an indexed open containing a history cell and a section over that open; comparison is performed on a displayed common neighborhood by restriction and the section classifier. Thus unit and maximal-ideal rows are repacked as point-germ rows without forming a stalk quotient.

The resulting package has exactly the rows requested by the $\mathsf{RingedSpace}^{\uparrow}$ consumer: a structure sheaf indexed by opens, ring-preserving restrictions, classifier-stable pointwise readback, and local-ring data visible through the existing ledger rows. □

Theorem 151.97 (Sheaf Scheme gluing cover row)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be a displayed affine cover presentation of an indexed open i for an indexed* $\mathsf{Sheaf}^{\uparrow}$ *carrier $\mathcal{S}$. Suppose each member $\iota(a)$ carries a local ringed chart row, and suppose the chart rows are compatible on the finite-intersection overlap rows $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ after restriction of sections, local-ring operations, and section classifiers. Then the downstream root source exposes a global section classifier row consumed by* $\mathsf{Scheme}^{\uparrow}$*: displayed local chart sections glue to a global section over i, and any two displayed gluings determine the same global classifier class after common-refinement comparison.*

Proof. First read cover membership from Definition 151.299. Every displayed history cell of the ambient open has a cover member witness, and pairwise overlap rows are supplied by finite intersections in the $\mathsf{Topology}^{\uparrow}$ carrier.

The assumed overlap compatibility is exactly the compatible local-family predicate of Definition 151.155, with the local ringed chart rows transported by restriction. Applying Definition 151.301 gives a displayed global section and identifies any two global readbacks that agree on the cover.

Refinement descent then supplies invariance under a different displayed affine cover. Use Definition 151.176 to pass gluing along refinements, Definition 151.179 for the refined exactness row, and Definition 151.189 for common-refinement comparison.

Thus $\mathsf{Scheme}^{\uparrow}$ receives a global section classifier row assembled from cover membership, finite-intersection overlaps, locality, gluing, refinement descent, and cover-presentation exactness, with no quotient of cover presentations. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeGluingCoverRow chart trace cons readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace cons readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace result deterministic`

Theorem 151.98 (Sheaf topology pullback cover row)**.** *Let $f : Y \to X$ be a classifier-respecting* BHist *map, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover row for a* $\mathsf{Sheaf}^{\uparrow}$ *carrier over X. Pull the cover back through the* $\mathsf{Topology}^{\uparrow}$ *pullback-open rows of Definition 88.60. Then the pulled-back cover preserves cover membership, restriction stability, and compatible local-family predicates: each source history cell covered by the pulled-back ambient open has a pulled-back cover-member witness, restriction maps commute with classifier transport along the pulled-back inclusion rows, and compatible local sections remain compatible after pullback to finite-intersection overlap rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafTopologyPullbackCover membership restriction compatibility`

Proof. Start with the topology pullback-open rows. The map row for f transports source classifier witnesses to target classifier witnesses, while Definition 88.62 preserves finite intersections and Definition 88.64 preserves displayed cover unions.

Apply Definition 151.299 to the target cover row and then precompose the resulting membership witnesses with f. The pulled-back cover-member witness is the same index $a : A$, read through the pulled-back predicate of Definition 88.60.

For restrictions, use Definition 151.300. The source inclusion rows are obtained by pulling back the target inclusions, and the classifier-respecting row for f supplies the transport needed to compare the source-side section classifiers.

Finally, compatibility of local families is checked on pairwise finite-intersection overlap rows. Since finite meets commute with the displayed pullback row and restriction classifier transport is stable along the same routes, every target overlap equality pulls back to the corresponding source overlap equality. This is the compatible local-family predicate required by the downstream topology pullback consumer. □

Definition 151.99 (Sheaf root export BHist closure surface)**.** *The root export surface for* $\mathsf{Sheaf}^{\uparrow}$ *is the following ten-field* BHist*-closed package carried by* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*.*

1. *The traditional target is sheaves of sections over indexed opens.*
2. *The BEDC source is the joint* $\mathsf{Topology}^{\uparrow}$ *and* $\mathsf{PreSheaf}^{\uparrow}$ *surface over* BHist*, read through* $\mathsf{NameCert}$*.*
3. *The history ledger is the cover-nerve ledger of Definition 151.160, with membership transport only through* hsame*.*
4. *The carrier is the indexed section presheaf of Definition 151.154.*
5. *The classifier is the open-indexed section classifier family $\sim_i$.*
6. *The stability rows are identity, composition, restriction classifier transport, and route-history associativity when* Cont *rows are present.*

7. *The cover ledger consists of indexed cover members, finite-intersection overlap rows, displayed refinement maps, and common-refinement spans.*
8. *The exactness rows are locality, gluing, refinement pullback, refinement gluing descent, and common-refinement gluing invariance.*
9. *The downstream threshold exports only these indexed-open, section, restriction, locality, gluing, and refinement rows.*
10. *The standard bridge row is absent; no host powerset equality, cover quotient, stalk quotient, or external sheafification primitive is exported.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootExportSurface`
Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootExportBHistClosureSurface`

Theorem 151.100 (Sheaf root export carrier scope)**.** *Let $\mathcal{T}$ be the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier and let $\mathcal{S}$ be the* $\mathsf{PreSheaf}^{\uparrow}$ *indexed section carrier equipped with the sheaf locality and gluing rows. The root export surface of Definition 151.267 exposes exactly the following carrier data: open indices and inclusion rows from $\mathcal{T}$, indexed section carriers* $\mathsf{Sec}(i)$*, section classifiers $\sim_i$, restriction maps $\rho_{i,j}$,* hsame*-stable* BHist *membership, and the cover-nerve ledger of Definition 151.160. It exports no carrier whose elements are host opens, quotiented covers, or external stalk representatives.*

Proof. Project the topology part first. The indexed-open carrier supplies open indices, inclusion rows, finite intersections, and the OpenAt membership predicate; Definition 151.160 records those rows over BHist and allows membership transport only through hsame.

Project the presheaf part next. By Definition 151.154, the only section-level carrier rows are $\mathsf{Sec}(i)$, $\sim_i$, the restriction maps, and their identity, composition, equivalence, and classifier-transport fields.

Finally use Definition 151.157 and Definition 151.299. These identify the carrier support and cover membership obligations as indexed-open and BHist-ledger data, so no host open carrier, cover quotient, or stalk representative is part of the export. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport carrier scope`

Theorem 151.101 (Sheaf root export stability and exactness package)**.** *For every indexed sheaf carrier $\mathcal{S}$ over an indexed-open carrier $\mathcal{T}$, the root export surface of Definition 151.267 packages the complete stability and exactness rows needed by downstream certificates:*

1. *cover-member witnesses and overlap projections from the* BHist *cover ledger;*
2. *identity, composition, and restriction classifier-stability for section restrictions;*
3. *locality and gluing exactness over indexed covers;*
4. *refinement pullback and refined-cover gluing descent;*
5. *common-refinement presentation exactness, including the restricted-open common-refinement rows below a fixed open.*

Every row is expressed in indexed opens, restriction maps, section classifiers, displayed refinements, common-refinement spans, Cont *route histories when present, and* hsame*-stable history membership.*

Proof. The cover-member and overlap rows are those of Definition 151.299, together with the pair-overlap projections of Definition 151.161. They keep the cover ledger internal to indexed opens and BHist cells.

The restriction rows are exactly Definition 151.300. Its identity, composition, classifier-stability, and optional Cont route-history clauses are the stability rows exported by the root package.

The locality and gluing rows are supplied by Definition 151.301. Refinement contributes Definition 151.175, Definition 151.176, and Definition 151.179.

Common-refinement exactness is read from Definition 151.188 and Definition 151.189; below a fixed open it is read from Definition 151.195. These rows assemble the displayed package without adding any host-level cover equality or quotient comparison. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport stability exactness package`

Theorem 151.102 (Sheaf root export downstream threshold)**.** *The root export surface of Definition 151.267 is the carried* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *threshold available to* $\mathsf{RingedSpace}^\uparrow$, $\mathsf{Scheme}^\uparrow$, *and restricted-open consumers. Such consumers receive only the indexed-open carrier, the section classifier family, restriction identity and composition rows, restriction classifier-stability rows, locality and gluing rows, the refinement ledger, common-refinement exactness, and the absent-standard-bridge row. In particular, they do not receive a standard bridge, host powerset cover equality, cover quotient, stalk quotient, or external sheafification operation from* $\mathsf{Sheaf}^\uparrow$.

Proof. For the global sheaf surface, apply Definition 151.190, Definition 151.283, and Definition 151.284. These rows expose the carrier, classifier, restriction, locality, gluing, and refinement obligations already present in $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$.

For the local consumers, apply Definition 151.239 and Definition 151.195. They show that working below a fixed open reuses the same ambient section carriers, classifiers, restrictions, cover ledgers, and exactness rows.

The negative part is part of the same export boundary. The root-unblock source of Definition 151.298 and the global obligation surface both state that no standard bridge or host-level cover comparison is exported, so ringed-space, scheme, and restricted-open chapters must cite the indexed threshold rather than reconstructing a larger classical sheaf interface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport downstream threshold`

Definition 151.103 (Sheaf root threshold certificate source)**.** *The* $\mathsf{Sheaf}^\uparrow$ *root threshold certificate source is the* BHist*-indexed package consisting of the* $\mathsf{Topology}^\uparrow$ *topology carrier, the* $\mathsf{PreSheaf}^\uparrow$ *presheaf restriction carrier, the open-indexed section classifier family, the cover-nerve ledger, the point-germ ledger, the restricted-open ledger, the displayed base-change rows, and the refinement and common-refinement exactness rows. It is a source of threshold certificates only through indexed opens,* BHist *membership, section classifiers, restriction maps, displayed refinements, displayed base-change squares, restricted-open slices, and* hsame*-stable transport.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootThresholdCertificateSource`

Theorem 151.104 (Sheaf root threshold carrier and classifier obligation)**.** *The certificate source of Definition 151.294 supplies the carrier and classifier obligations of the* $\mathsf{Sheaf}^\uparrow$ *root solely through indexed opens,* BHist *membership, the section classifier family, restriction maps, and* hsame*-stable transport. In particular, the carrier side is exhausted by open indices, section carriers* $\mathsf{Sec}(i)$*, classifier rows* $\sim_i$*, restriction rows* $\rho_{i,j}$*, cover membership over* BHist*, and point-germ rows read on displayed neighborhoods.*

Proof. Read the carrier row from Definition 151.154. It supplies the open-indexed section carriers, the section classifiers, the restriction maps, and the identity, composition, equivalence, and classifier-transport rows attached to them.

Read the history side from Definition 151.160. Its cover-member rows are indexed by open carriers and BHist cells, and transport across history witnesses is only the displayed hsame transport already recorded in the ledger.

For pointwise readback, use Definition 151.231. A point-germ row chooses an indexed neighborhood containing the history cell and compares sections by restricting to displayed common neighborhoods, so it adds no host point, stalk quotient, or external carrier beyond the classifier and restriction rows already named. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold carrier classifier semantic certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold semantic name certificate`

Theorem 151.105 (Sheaf root threshold gluing and refinement obligation)**.** *The certificate source of Definition 151.294 assembles the exactness obligation of the* $\mathsf{Sheaf}^{\uparrow}$ *root from locality, gluing, refinement pullback, refinement gluing descent, restricted common-refinement exactness, and base-change gluing pullback. Each exactness row is expressed through indexed covers, compatible local families, displayed refinements, restricted-open slices, displayed base-change squares, section classifiers, and restriction transport.*

Proof. Begin with the root export exactness package of Definition 151.269. It already groups locality, gluing, refinement pullback, refinement gluing descent, and common-refinement presentation exactness as the stability and exactness rows exported by $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

The restricted-open part is read in the same package from the restricted common-refinement rows below a fixed indexed open. These rows keep the ambient section classifiers and restriction maps, and only narrow the cover ledger to the displayed slice.

For movement across a displayed source and target, apply Definition 151.208. Its cover-pullback, gluing-pullback, and common-refinement rows reuse the same exactness package after the displayed base-change square identifies the relevant source opens and pulled-back cover members. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold gluing refinement obligation`

Theorem 151.106 (Sheaf root threshold downstream coverage obligation)**.** *Every* $\mathsf{Sheaf}^{\uparrow}$ *row consumed by* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers is covered by the threshold source of Definition 151.294. Thus downstream chapters receive their sheaf-side data from the topology carrier, restriction carrier, section classifiers, cover-nerve ledger, point-germ ledger, restricted-open ledger, displayed base-change rows, and refinement or common-refinement exactness rows of that single source.*

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, apply Definition 151.197. Its sheaf-side requests are precisely the indexed section carriers, classifiers, restriction stability, cover-gluing exactness, restricted-open surface, and displayed base-change surface listed in the threshold source.

For $\mathsf{Scheme}^{\uparrow}$, apply Definition 151.198. Each affine or ringed-space chart first consumes the ringed-space sheaf boundary, and overlap compatibility is discharged by the cover-presentation and common-refinement exactness rows already included in the source.

For topology pullback consumers, apply Definition 151.226. The pulled-back cover rows are displayed base-change rows over the same indexed-open and BHist ledgers, so they are covered by the threshold source without adding a host powerset cover comparison or external sheaf model. □

Lean4 ✓ BEDC.Derived.SheafUp.SheafRootThreshold downstream coverage obligation

Definition 151.107 (Sheaf root consumer-threshold projection source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root consumer-threshold projection source is the* BHist*-indexed package obtained from Definition 151.294 by exposing three consumer faces: the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing face, and the topology pullback face. It is recorded by the following ten fields.*

1. *The traditional target is the sheaf-side data of a structure sheaf that can be read by ringed spaces, schemes, and pullback-open consumers.*
2. *The BEDC source is the single threshold source of Definition 151.294, not a separate host sheaf object.*
3. *The carrier row is the indexed-open section carrier* $\mathsf{Sec}(i)$*, with restricted-open carriers and displayed source carriers obtained only by slicing or pulling back indexed opens.*
4. *The classifier row is the section classifier family* $\sim_i$*, together with its restricted-open and displayed base-change readbacks.*
5. *The restriction row is restriction identity, restriction composition, and classifier transport for every displayed inclusion used by the three consumers.*
6. *The RingedSpaceUp projection row is the local-ring consumer surface: section carriers, section classifiers, ring-preserving restriction routes, point-germ local-ring readback, restricted-open rows, and base-change rows.*
7. *The SchemeUp projection row is the affine-chart gluing surface: chart-wise ringed-space projections, displayed affine cover membership, overlap compatibility, and common-refinement comparison.*
8. *The topology pullback projection row is the displayed base-change surface: cover pullback, gluing pullback, pulled-back common refinement, and classifier transport along the displayed square.*
9. *The ledger and exactness row is the cover-nerve ledger, point-germ ledger, restricted-open ledger, refinement ledger, cover-presentation exactness, and base-change exactness already present in the threshold source.*
10. *The standard-bridge field is absent: the projection source exports no host powerset cover equality, no cover quotient, no stalk quotient, no external sheafification operation, and no* $\mathsf{StandardBridge}$*.*

Thus the same BHist*-indexed source is projected separately to the three consumer families without duplicating the sheaf root obligations.*

Lean4 (definition) BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjectionSource
Lean4 ✓ BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjectionSource projection faces
Lean4 (definition) BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjection source

Theorem 151.108 (Sheaf root RingedSpace consumer projection)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring consumer receives exactly the indexed sections, section classifiers, restriction identity, restriction composition, restriction transport, point-germ local-ring readback, restricted-open rows, and base-change rows of Definition 151.256. No additional sheaf carrier, cover comparison, stalk quotient, or standard bridge is available through this projection.*

Lean4 ✓ BEDC.Derived.SheafUp.SheafRootConsumerRingedSpaceProjection semantic name certificate
Lean4 ✓ BEDC.Derived.SheafUp.SheafRootRingedSpaceConsumerProjection scope

Proof. Start from Definition 151.197. Its sheaf-side list asks for section carriers, classifiers, restriction stability, cover-gluing exactness, restricted-open rows, and base-change rows, together with non-sheaf commutative-ring data supplied outside $\mathsf{Sheaf}^{\uparrow}$.

The carrier and classifier entries are supplied by Definition 151.295. This gives the open-indexed section carrier, the classifier family, restriction identity and composition, classifier transport, BHist cover membership, and point-germ rows read on displayed neighborhoods.

The exactness and transport entries are supplied by Definition 151.296. Locality, gluing, refinement pullback, restricted-open exactness, and displayed base-change exactness are already fields of the root threshold source, so the ringed-space projection is precisely the listed face of Definition 151.256. □

Theorem 151.109 (Sheaf root Scheme consumer projection)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing consumer receives the same threshold source by chart-wise* $\mathsf{RingedSpace}^{\uparrow}$ *projection, followed by cover-presentation exactness and common-refinement exactness on chart overlaps. Thus affine-chart sections, overlap classifiers, glued section classes, and comparison across common refinements are all read from Definition 151.256.*

Proof. Apply Definition 151.198. Each affine or ringed-space chart first reads the sheaf boundary through the ringed-space consumer, so the chart-wise carrier, classifier, restriction, restricted-open, point-germ, and base-change rows are the rows identified in Definition 151.257.

The coverage claim is Definition 151.297: every sheaf row consumed by $\mathsf{Scheme}^{\uparrow}$ is already covered by the threshold source. This keeps the chart gluing surface inside the same BHist-indexed package.

For independence of the chosen affine cover presentation, use Definition 151.189. The displayed common refinement compares the pulled-back local families and identifies the resulting global classifier class, so no quotient of cover presentations or external sheaf model is introduced. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace unary result`

Theorem 151.110 (Sheaf root topology pullback consumer projection)**.** *Topology pullback consumers receive only the displayed base-change, cover-pullback, gluing-pullback, and common-refinement rows of Definition 151.256. These rows preserve pulled-back cover membership, compatible local-family predicates, glued section classifiers, and common-refinement comparison, without exporting the ringed-space local-ring face or the scheme affine-chart face.*

Proof. Begin with Definition 151.226. It identifies the consumer request as a displayed pullback of indexed cover rows: cover membership, restriction stability, and compatible local-family predicates are transported along the topology pullback-open rows.

The complete base-change surface is Definition 151.208. Its displayed square supplies cover pullback, gluing pullback, and common-refinement invariance using the same indexed-open carrier, classifier family, cover ledger, and refinement exactness fields recorded in Definition 151.256.

Since this consumer reads only displayed source and target rows, the projection discards the ringed-space local-ring field and the scheme affine-chart field. What remains is exactly the base-change and common-refinement face of the root threshold source, with no host powerset cover comparison or standard bridge. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootConsumerPullbackProjection base change face`

Definition 151.111 (Sheaf root face exactness source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root face exactness source is the four-face package obtained from the* BHist *root-unblock source, the cover-nerve export surface,*

and the consumer-threshold projection source. It is read over BHist *histories,* hsame *transport, optional* Cont *route histories, the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier, the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction carrier, and the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation rows.*

1. *The cover-membership face consists of accepted cover rows, member witnesses* $\mathsf{OpenAt}(\iota(a), b)$*, finite-intersection overlap rows, and* hsame*-stable history transport from Definition 151.160.*
2. *The restriction-route face consists of the section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, restrictions* $\rho_{i,j}$*, identity rows, composition rows, classifier transport, and optional* Cont *route-history associativity rows.*
3. *The locality, gluing, and refinement exactness face consists of compatible local-family predicates, gluing witnesses, locality comparison, displayed cover refinement, refinement pullback, refinement gluing descent, displayed common-refinement comparison, and cover-presentation exactness.*
4. *The downstream projection face consists only of the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring projection, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing projection, and the topology pullback projection named in Definition 151.256.*

The package contains no host powerset equality of open covers, no cover quotient, no stalk quotient, no external sheafification operation, and no StandardBridge *field.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootFaceExactnessSource`

Theorem 151.112 (Sheaf root face classifier coverage)**.** *Every row of the root-unblock source of Definition 151.298 and every row of the cover-nerve export surface of Definition 151.302 lands in exactly one of the first three faces of Definition 151.271. The downstream projection face receives only projections of those faces and adds no root carrier, root classifier, or root exactness row.*

Proof. For the cover rows, read the cover ledger in Definition 151.160. The accepted cover, cover-member witness, finite-intersection overlap, pair-overlap projection, and hsame membership transport rows are precisely the cover-membership face. The exhaustion and projection theorems Definition 151.303 and Definition 151.161 show that no additional cover row is required.

For the restriction rows, use Definition 151.300 and Definition 151.304. Their identity, composition, classifier-preservation, two-step route, and optional Cont history clauses are exactly the restriction-route face.

For the exactness rows, use Definition 151.301, Definition 151.175, Definition 151.176, Definition 151.179, and Definition 151.189. These rows cover locality, gluing, displayed refinement, common-refinement comparison, and cover-presentation exactness, so they land in the locality, gluing, and refinement exactness face.

Finally compare with the consumer-threshold source of Definition 151.256. Its ringed-space, scheme, and topology pullback rows cite the same carrier, classifier, cover, restriction, and exactness rows through projection, so they occupy the downstream projection face rather than adding root data. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead coverage`

Theorem 151.113 (Sheaf root face gluing exactness)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over a* BHist*-scoped indexed open topology carrier* $\mathcal{T}$*, and let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an accepted indexed cover of* $i : \mathsf{OpenIx}$*. For every compatible local family* $s_a : \mathsf{Sec}(\iota(a))$*, locality plus gluing plus refinement exactness gives a* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*-consumable exactness row:*

1. *a global section* $g : \mathsf{Sec}(i)$ *restricts to the local family;*
2. *any two global readbacks whose cover restrictions agree are classifier-equal in* $\sim_i$*;*
3. *after any displayed refinement or displayed common refinement, the pulled-back local families determine the same global classifier class.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted global soundness`

Proof. Begin with the compatible local family over $\mathcal{U}$. The gluing field in Definition 151.155 supplies a global section $g : \mathsf{Sec}(i)$ and the displayed member equalities $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$.

Use locality from the same definition for uniqueness. If two global readbacks agree on every member row, locality returns classifier equality in $\sim_i$; the same argument is packaged by Definition 151.169.

Now pass to a displayed refinement. The compatibility row is transported by Definition 151.175, and the gluing row descends by Definition 151.176. The combined refined-cover statement is Definition 151.179.

For comparison of two cover presentations, use the displayed common-refinement surface of Definition 151.186. The invariance theorem Definition 151.188, together with Definition 151.189, identifies the two resulting global classifier classes.

All cited steps use indexed covers, BHist membership, hsame transport, section classifiers, restriction composition, and optional Cont route histories. Therefore the exactness row is consumable by $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ without adding a host cover comparison or quotient row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restricted global soundness`

Theorem 151.114 (Sheaf root face downstream separation)**.** *The* $\mathsf{RingedSpace}^\uparrow$*,* $\mathsf{Scheme}^\uparrow$*, and topology pullback consumers receive only projections of the four faces of Definition 151.271. Consumer-specific rows such as local-ring structure, affine-chart spectra data, and displayed pullback-open choices do not feed back into the root* $\mathsf{Sheaf}^\uparrow$ *carrier, root classifier, cover ledger, restriction-route face, or locality and gluing exactness face.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead downstream separation`

Proof. For $\mathsf{RingedSpace}^\uparrow$, apply Definition 151.257. Its sheaf-side input is the indexed section carrier, section classifier, restriction stability, point-germ readback, restricted-open rows, and base-change rows projected from the root source. The commutative-ring and local-ring rows are supplied outside the sheaf root.

For $\mathsf{Scheme}^\uparrow$, apply Definition 151.258. Each chart first reads the ringed-space projection, while affine-chart and spectra data remain chart-side data. The sheaf contribution is only cover-presentation exactness and common-refinement comparison already present in the root exactness face.

For topology pullback consumers, apply Definition 151.259 and Definition 151.226. The pullback row transports cover membership, compatible families, gluing, and common refinements through the displayed base-change surface, but it does not introduce a ringed-space local-ring row or a scheme affine-chart row.

The downstream consumption boundary of Definition 151.196 and the exhaustion theorem Definition 151.199 close the separation. They state that consumers can only read the listed sheaf-side projections, so consumer-specific structure cannot be returned as a new root $\mathsf{Sheaf}^\uparrow$ obligation. □

Lean4 ✓ BEDC.Derived.SheafUp.SheafRootFaceRead downstream projection separation

Definition 151.115 (SheafUp root consumer projection seal)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root consumer projection seal is the three-consumer package assembled from the consumer-threshold projection source of Definition 151.256, the root face exactness source of Definition 151.271, and the downstream consumption boundary of Definition 151.196. Its three public projections are the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring projection, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart projection, and the topology pullback projection. Each projection is a read-only face of the same* BHist*-indexed sheaf root package, and none of the projections exports host powerset equality, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Lean4 (definition) BEDC.Derived.SheafUp.SheafRootConsumerProjectionSeal

Theorem 151.116 (SheafUp RingedSpace local-ring projection seal)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face of Definition 151.252 may read only the* $\mathsf{Sheaf}^{\uparrow}$ *section carriers, section classifiers, restriction stability rows, cover-gluing exactness rows, restricted-open rows, and displayed base-change rows. The commutative-ring and local-ring data used by the consumer are supplied from* $\mathsf{CommRing}^{\uparrow}$ *outside* $\mathsf{Sheaf}^{\uparrow}$*. Through this sealed projection, the consumer cannot read host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Proof. The consumer-threshold source of Definition 151.256 names the ringed-space projection row as a face of the root threshold package. Its existing projection theorem Definition 151.257 identifies exactly the section carrier, classifier, restriction, point-germ readback, restricted-open, and base-change rows available to $\mathsf{RingedSpace}^{\uparrow}$.

The root face coverage and exactness results Definition 151.272, Definition 151.273, and Definition 151.274 show that these rows are projections of the root cover-membership, restriction-route, and locality-gluing-refinement faces. They add no root carrier and no consumer-specific exactness row.

Finally, Definition 151.199 exhausts the downstream sheaf-side reads. The local-ring and commutative-ring structure therefore enters only from $\mathsf{CommRing}^{\uparrow}$, while host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure remain outside the sealed projection. □

Lean4 ✓ BEDC.Derived.SheafUp.SheafRootRingedSpaceLocalRingProjection seal

Theorem 151.117 (SheafUp Scheme affine-chart projection seal)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face of Definition 151.252 may read affine-chart gluing only through cover-presentation rows, displayed common-refinement rows, and the* $\mathsf{RingedSpace}^{\uparrow}$ *projection rows sealed in Definition 151.253. Spectra data and chart-ring data are supplied outside* $\mathsf{Sheaf}^{\uparrow}$*. Through this sealed projection, the consumer cannot read host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Lean4 ✓ BEDC.Derived.SheafUp.SheafRootSchemeAffineChartProjection seal

Proof. The consumer-threshold source of Definition 151.256 names the scheme projection row as chart-wise access through ringed-space projections, cover-presentation exactness, and common-refinement comparison. The existing projection theorem Definition 151.258 supplies this chart-wise reading of the same threshold source.

The root face exactness package then separates the sheaf rows from chart data. By Definition 151.272 and Definition 151.273, cover-presentation comparison and common-refinement invariance are already part of the locality, gluing, and refinement exactness face; by Definition 151.274, chart-local spectra and ring rows do not feed back into the root sheaf package.

Apply Definition 151.199 to the scheme consumption boundary. Every sheaf row consumed by $\mathsf{Scheme}^{\uparrow}$ is already listed there, so spectra data, chart-ring data, host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure are not supplied by $\mathsf{Sheaf}^{\uparrow}$. □

Theorem 151.118 (SheafUp topology pullback projection seal)**.** *The topology pullback face of Definition 151.252 may read only pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and common-refinement transport along the displayed base-change surface. It does not read the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face or the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and it exports no host powerset equality, cover quotient, stalk quotient, external sheafification, or standard bridge closure.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootTopologyPullbackProjection seal`

Proof. The consumer-threshold source of Definition 151.256 names the topology pullback projection as the displayed base-change face. The existing projection theorem Definition 151.259 identifies the available rows as cover pullback, gluing pullback, pulled-back common refinement, and classifier transport along the displayed square.

The root face coverage and exactness rows control those reads. Use Definition 151.272 for cover membership and restriction-route stability, and use Definition 151.273 for compatible-family transport, gluing pullback, and common-refinement comparison.

The separation theorem Definition 151.274, together with Definition 151.199, removes the other two consumer faces from the topology pullback projection. Thus the pullback consumer receives only the displayed base-change rows and no host powerset equality, cover quotient, stalk quotient, external sheafification, or standard bridge closure. □

Definition 151.119 (Sheaf consumer access trace source)**.** *A finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access trace is the freely generated finite list syntax whose constructors are the following read events:*

1. *carrier and classifier reads of the indexed section carriers* $\mathsf{Sec}(i)$ *and classifiers* $\sim_i$*;*
2. *restriction-route reads of identity, composition, classifier transport, and optional* Cont *route-history rows;*
3. *cover-membership reads of accepted covers, member witnesses, finite-intersection overlaps, and* hsame *membership transport;*
4. *locality and gluing reads of compatible-family predicates, local agreement, global witnesses, and uniqueness rows;*
5. *refinement and common-refinement reads of displayed refinements, refinement pullbacks, gluing descent, and common-refinement comparison rows;*
6. *point-germ reads of named pointwise readback rows without forming stalk quotients;*
7. *base-change reads of displayed base-change squares, pulled-back cover membership, pulled-back compatible families, and gluing transport;*
8. *the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring consumer projection;*
9. *the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart consumer projection;*
10. *the topology pullback consumer projection.*

Every constructor is formed over BHist *histories,* hsame *transport, optional* Cont *route histories, the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier, the* $\mathsf{PreSheaf}^{\uparrow}$ *section carrier, and the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *rows. The trace source has no constructor for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerAccessTraceSource`

Theorem 151.120 (Sheaf consumer access trace face coverage)**.** *Every constructor of Definition 151.244 lands in exactly one face: either one of the root faces of Definition 151.271 or one of the three consumer projections sealed in Definition 151.252.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead restriction route readback`

Proof. Proceed by cases on the freely generated trace constructor.

The carrier, classifier, restriction-route, cover-membership, locality, gluing, refinement, common-refinement, point-germ, and base-change cases are root-facing reads. The carrier and classifier rows land in the restriction-route face, the accepted-cover and overlap rows land in the cover-membership face, and the compatibility, locality, gluing, refinement, common-refinement, point-germ, and base-change rows land in the locality, gluing, and refinement exactness face. This placement is the coverage statement of Definition 151.272, with exactness supplied by Definition 151.273.

The remaining cases are the three downstream consumer constructors. The $\mathsf{RingedSpace}^{\uparrow}$ constructor lands in Definition 151.253, the $\mathsf{Scheme}^{\uparrow}$ constructor lands in Definition 151.254, and the topology pullback constructor lands in Definition 151.255.

These targets are disjoint by construction of Definition 151.271: the first three faces contain root sheaf rows, while the downstream face contains only projections. Thus each constructor has exactly one landing face. □

Theorem 151.121 (Sheaf consumer access trace no-feedback)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback constructors of Definition 151.244 do not create further* $\mathsf{Sheaf}^{\uparrow}$ *carrier rows, classifier rows, exactness rows, or bridge rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace no feedback`

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, the local-ring constructor reads the sheaf-side carrier, classifier, restriction, point-germ, restricted-open, and base-change rows only through the sealed projection. By Definition 151.274, the commutative-ring and local-ring data are outside the sheaf root and cannot return as root carrier, classifier, or exactness rows.

For $\mathsf{Scheme}^{\uparrow}$, the affine-chart constructor first passes through the ringed-space projection and then reads cover-presentation and common-refinement rows. The same separation theorem keeps spectra data and chart-ring data outside the sheaf root, so the chart constructor contributes no root row.

For topology pullback, the constructor reads displayed base-change transport, cover pullback, compatible-family transport, gluing pullback, and common-refinement transport. The pullback-open choices are consumer-side data by Definition 151.274, not root sheaf obligations.

Finally apply Definition 151.199. The downstream boundary exhausts all sheaf-side reads and excludes host powerset equality, cover quotients, external sheaf models, and bridge closure, so no consumer constructor can manufacture a further $\mathsf{Sheaf}^{\uparrow}$ root row. □

Theorem 151.122 (Sheaf consumer access trace exhaustion)**.** *Every allowed downstream* $\mathsf{Sheaf}^{\uparrow}$ *access by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors through a finite access trace of Definition 151.244.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append exhaustion`

Proof. Inspect the downstream row lists in Definition 151.196. Carrier and classifier access is represented by the carrier-classifier constructor; restriction stability is represented by the restriction-route constructor; cover ledgers are represented by the cover-membership constructor; locality, gluing, refinement, common-refinement, restricted-open, point-germ, and base-change access are represented by their corresponding trace constructors.

Now inspect the consumer projections in Definition 151.252. The $\mathsf{RingedSpace}^{\uparrow}$ local-ring projection is represented by the ringed-space consumer constructor, the $\mathsf{Scheme}^{\uparrow}$ affine-chart projection by the scheme consumer constructor, and the displayed pullback projection by the topology pullback constructor.

By Definition 151.245, each selected constructor lands in the correct root face or sealed consumer projection. By Definition 151.246, selecting a consumer constructor does not add any further sheaf row that would require another constructor outside the trace grammar.

The relevant downstream proof uses only finitely many rows from the displayed boundary and projection lists. Listing those rows in their order of use gives the required finite trace, and every allowed access is therefore factored through Definition 151.244. □

Definition 151.123 (Sheaf root access normal form)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root access normal form is the finite classification of downstream-visible root reads generated by Definition 151.244. Its classes are:*

1. *carrier, classifier, and restriction-route rows for the indexed section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, restrictions* $\rho_{i,j}$*, identity rows, composition rows, classifier transport, and optional* Cont *route histories;*
2. *cover-member and overlap rows for accepted covers, member witnesses, finite-intersection overlaps, pair-overlap projections, and* hsame *membership transport;*
3. *locality, gluing, refinement, common-refinement, and base-change rows for compatible-family predicates, global witnesses, uniqueness rows, displayed refinements, refinement pullbacks, gluing descent, cover-presentation exactness, displayed common refinements, displayed base-change squares, cover pullback, compatible-family transport, and gluing pullback;*
4. *point-germ comparison rows that expose named pointwise readbacks without forming stalk quotients;*
5. *the three downstream projection faces: the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring projection, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart projection, and the topology pullback projection.*

These classes are read-only faces of the same BHist*-indexed sheaf root package. They contain no class for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, standard bridges, or consumer-private algebraic, chart, or pullback data.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootAccessNormalFormSourceCase`

Theorem 151.124 (Sheaf root access normal-form coverage)**.** *Every admitted* $\mathsf{Sheaf}^{\uparrow}$ *root access trace of Definition 151.244 lands in one class of Definition 151.240.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead locality gluing readback`

Proof. Proceed by cases on the trace constructor of Definition 151.244.

Carrier, classifier, and restriction-route reads land in the first normal-form class. The root face source of Definition 151.271 places section carriers, section classifiers, restrictions, identity rows,

composition rows, classifier transport, and optional route histories in the restriction-route face, and Definition 151.272 gives the corresponding coverage.

Cover-membership and overlap reads land in the second class. The accepted-cover row, member witness, finite-intersection overlap, pair-overlap projection, and hsame transport are exactly the cover-membership rows isolated by Definition 151.245.

Locality, gluing, refinement, common-refinement, and base-change reads land in the third class. Their exactness rows are the locality, gluing, and refinement face of Definition 151.271, with gluing exactness supplied by Definition 151.273; the downstream boundary of Definition 151.196 records the same restricted-open, base-change, refinement, and exactness rows.

Point-germ reads land in the fourth class because Definition 151.244 admits named pointwise readbacks only as point-germ comparison rows. The same definition excludes stalk quotient formation, so the point-germ case does not branch into an additional quotient class.

The remaining constructors are the three downstream consumer projections. They land in the fifth class by the projection seal of Definition 151.252, and the face coverage theorem Definition 151.245 identifies no further trace constructor. Thus every admitted access has a normal-form landing class. □

Theorem 151.125 (Sheaf root access normal-form consumer factorization)**.** *Every downstream* $\mathsf{Sheaf}^{\uparrow}$ *access used by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors first through Definition 151.240 and then through the existing projection seals.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace consumer access trace`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootAccessNormalForm consumer factorization`

Proof. Begin with an allowed downstream access. By Definition 151.247, it is represented by a finite trace of Definition 151.244. Apply Definition 151.241 to send each trace constructor to its normal-form class.

For $\mathsf{RingedSpace}^{\uparrow}$, the relevant classes are carrier, classifier, restriction, cover-gluing, restricted-open, base-change, and point-germ comparison rows. The local-ring projection seal Definition 151.253 reads exactly those sheaf-side classes and supplies no commutative-ring or local-ring row from the sheaf root.

For $\mathsf{Scheme}^{\uparrow}$, first pass through the ringed-space projection and then use the cover-presentation and displayed common-refinement classes. The affine-chart projection seal Definition 151.254 identifies spectra and chart-ring rows as consumer-side data, while Definition 151.276 lists the remaining sheaf rows visible to the joint face.

For topology pullback, use the base-change class together with cover-membership, restriction-route, compatible-family, gluing, and common-refinement rows. The pullback projection seal Definition 151.255 is exactly this factor, and Definition 151.277 prevents the other two consumer faces from entering it.

The factorization therefore has two stages: access traces land in the finite normal form, and the three downstream consumers read their respective normal-form classes only through the sealed projections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace root access factorization`

Theorem 151.126 (Sheaf root access normal form exports no extra rows)**.** *The normal form of Definition 151.240 exports no host powerset equality, no cover quotient, no stalk quotient, no external sheafification row, no standard bridge row, and no consumer-private algebraic, chart, or pullback data.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootAccessNormalForm no extra rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead normal form readback`

Proof. For host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridges, inspect the source grammar. The trace source of Definition 151.244 has no constructor for those rows, and the root face exactness source of Definition 151.271 explicitly keeps them outside the four root faces.

For consumer-private algebraic rows, apply Definition 151.253. It says the commutative-ring and local-ring data used by $\mathsf{RingedSpace}^{\uparrow}$ are supplied outside $\mathsf{Sheaf}^{\uparrow}$, so the normal form cannot export them as sheaf root rows.

For affine-chart and spectra rows, apply Definition 151.254. The scheme face reads sheaf rows through ringed-space projections, cover-presentation exactness, and common-refinement comparison, while spectra data and chart-ring data remain outside the sheaf root.

For pullback-private choices, apply Definition 151.255. The topology pullback face reads displayed base-change transport, pulled-back cover membership, compatible-family transport, gluing pullback, and common-refinement transport; it does not export consumer-specific pullback-open choices as root data.

Finally combine the joint face and downstream boundary. The joint consumer face noninterference theorem Definition 151.277 prevents private consumer data from feeding back into the root, and Definition 151.199 exhausts the sheaf-side rows consumed downstream. Thus the normal form exports only the listed finite classes and no extra row. □

Definition 151.127 (Sheaf consumer trace composition source)**.** *Let T_1 and T_2 be finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces of Definition 151.244. Their consumer trace composition $T_1 \star T_2$ is the finite trace obtained by listing every constructor of T_1 in its given order and then listing every constructor of T_2 in its given order. The empty trace is a two-sided unit for this operation.*

The source of this operation is restricted to the same objects as the access trace grammar: BHist *histories,* hsame *transport, optional* Cont *route histories,* $\mathsf{Topology}^{\uparrow}$ *indexed opens,* $\mathsf{PreSheaf}^{\uparrow}$ *section carriers, and* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *rows. No constructor of $T_1 \star T_2$ may mention host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows except through the sealed consumer projections already present in Definition 151.244.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerAccessTraceCompositionSource`
Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerTraceCompositionSource`

Theorem 151.128 (Sheaf consumer trace concatenation coverage)**.** *For any finite traces T_1 and T_2 of Definition 151.244, every constructor in the composed trace $T_1 \star T_2$ of Definition 151.215 lands in the same root face or sealed consumer projection that it occupied before composition.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append membership coverage`

Proof. Proceed by the ordered list presentation of $T_1 \star T_2$.

For a constructor appearing in the first segment, apply Definition 151.245 to its occurrence in T_1. Its landing face is one of the root faces of Definition 151.271 or one of the sealed consumer projections of Definition 151.252; the concatenation operation only preserves the occurrence and its order inside the first segment.

For a constructor appearing in the second segment, apply the same coverage theorem to its occurrence in T_2. The first segment supplies no wrapper and no coercion around the second segment, so the second occurrence keeps the landing face it had before composition.

Thus every constructor in the concatenated trace has exactly the face assigned to it by the original access trace coverage theorem. □

Theorem 151.129 (Sheaf consumer trace composition no-feedback)**.** *Composing finite* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback consumer traces by Definition 151.215 creates no additional* $\mathsf{Sheaf}^{\uparrow}$ *carrier row, classifier row, exactness row, or bridge row.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append no zero extension`

Proof. First apply Definition 151.246 to the consumer constructors in T_1. Each $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, and topology pullback event reads through the sealed projection and contributes no root carrier, classifier, exactness, or bridge row.

Apply the same theorem to the consumer constructors in T_2. The composition source merely places the second trace after the first, so no event in the second segment obtains a new sheaf row from the preceding segment.

It remains to rule out feedback across the boundary between the two segments. By Definition 151.274, consumer-specific algebraic, chart, and pullback data are outside the root sheaf faces. By Definition 151.199, all allowed downstream sheaf reads have already been exhausted by the listed root rows and sealed projections.

Therefore the concatenation boundary carries only the finite order of reads. It supplies no constructor capable of manufacturing a further $\mathsf{Sheaf}^{\uparrow}$ carrier, classifier, exactness, or bridge row. □

Theorem 151.130 (Sheaf consumer trace factorization composes)**.** *Suppose two allowed downstream* $\mathsf{Sheaf}^{\uparrow}$ *accesses by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factor through finite traces* T_1 *and* T_2 *of Definition 151.244. Their sequential use factors through the composed trace* $T_1 \star T_2$ *of Definition 151.215.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger trace factorization composes`

Proof. Choose the trace $T_1 \star T_2$ by listing the constructors witnessing the first downstream access and then listing the constructors witnessing the second downstream access. This is a finite trace because both inputs are finite and the composition source is ordinary finite concatenation.

The first access factors through the first segment by the assumed factorization of T_1. The second access factors through the second segment by the assumed factorization of T_2, and the preceding segment changes only the position of the second segment in the list.

By Definition 151.216, every constructor in both segments still lands in its original root face or sealed projection. By Definition 151.217, the boundary between the two segments creates no additional sheaf row that would require another access constructor.

The sequential downstream use is therefore represented by first reading along T_1 and then reading along T_2, which is exactly the composed finite trace. □

Definition 151.131 (Sheaf consumer access trace composition source)**.** *Let* P *be the fixed* BHist*-indexed* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root package used by Definition 151.244. If* T_1 *and* T_2 *are finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces over* P*, their composite* $T_1 \mathbin{++} T_2$ *is the finite list concatenation obtained by reading all constructors of* T_1 *followed by all constructors of* T_2*. The source constructors of* $T_1 \mathbin{++} T_2$ *are still exactly the constructors of Definition 151.244: carrier and classifier reads, restriction-route reads, cover-membership reads, locality and gluing reads, refinement and common-refinement reads, point-germ reads, base-change reads, and the three sealed consumer projection reads for* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback.*

The composite carries no additional constructor for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows. Its only extra operation is finite list concatenation inside the same BHist*-scoped root package P.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTraceCompositionSource membership split`

Theorem 151.132 (Sheaf consumer access trace concatenation closure)**.** *Accepted finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces over one* BHist*-indexed* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root package are closed under the composition operation of Definition 151.248. In particular, if T_1 and T_2 are generated by Definition 151.244, then $T_1 \mathbin{++} T_2$ is generated by the same grammar and contains no constructor for host powerset equality, cover quotients, stalk quotients, external sheafification, or standard bridge rows.*

Proof. Induct on the second trace T_2 as a finite list of constructors. In the empty case, $T_1 \mathbin{++} T_2$ is just T_1, so acceptance is the given acceptance of the first trace.

In the cons case, write T_2 as a shorter trace followed by one final constructor c. By the induction hypothesis, concatenating T_1 with the shorter trace remains an accepted trace over the same root package.

It remains to append c. The constructor cases are exactly the cases listed in Definition 151.244: carrier and classifier, restriction route, cover membership, locality and gluing, refinement and common refinement, point germ, base change, and the three downstream consumer projections. Each case appends one already admitted constructor, so the resulting list is still accepted by the same grammar.

The excluded host-level operations are not constructor cases. Since finite concatenation only appends constructors already present in the grammar, it cannot create a host powerset equality row, a cover quotient, a stalk quotient, an external sheafification step, or a standard bridge row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append closed`

Theorem 151.133 (Sheaf consumer access trace face-map stability)**.** *For finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces T_1 and T_2 over the same* BHist*-indexed root package, the landing-face assignment of Definition 151.245 sends $T_1 \mathbin{++} T_2$ to the finite concatenation of the landing-face reads of T_1 and the landing-face reads of T_2. The same assignment preserves the no-feedback separation of Definition 151.246 across the composite trace.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace face map stability`

Proof. Proceed by constructor cases on the appended segment T_2. The empty appended segment contributes no landing-face read, so the landing-face list of the composite is the landing-face list of T_1.

For a root-facing constructor, Definition 151.245 gives its unique root face. Appending the constructor appends exactly that face read after the face reads already assigned to the previous part of the composite.

For a downstream consumer constructor, Definition 151.245 sends the constructor to its sealed consumer projection. By Definition 151.246, that projection does not return a new $\mathsf{Sheaf}^{\uparrow}$ carrier row, classifier row, exactness row, or bridge row.

Thus the landing-face assignment commutes with finite list concatenation, and the downstream no-feedback separation remains valid after the two traces are read sequentially. □

Theorem 151.134 (Sheaf composite consumer access exhaustion)**.** *Every sequential downstream* $\mathsf{Sheaf}^{\uparrow}$ *access by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors through one finite* BHist*-scoped trace obtained by concatenating the finite traces for the individual accesses.*

Proof. Apply Definition 151.247 to the first downstream access. This gives a finite trace over the displayed BHist-indexed $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ root package whose constructors are those of Definition 151.244.

Apply Definition 151.247 again to the next downstream access, or to each member of the finite sequential access list. Each application gives another finite trace over the same root package, because the downstream consumers read the sealed $\mathsf{Sheaf}^{\uparrow}$ root package rather than creating a replacement root.

Concatenate the resulting finite traces in their order of downstream use. By Definition 151.249, the concatenation is still an accepted finite trace in the same grammar.

Finally use Definition 151.250. The composite trace preserves the landing-face assignment and the no-feedback separation, so the sequential access never leaves the finite BHist-scoped trace surface and never imports host-level sheaf machinery. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace composite exhaustion`

Theorem 151.135 (Sheaf root downstream consumer carrier scope)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback receive their* $\mathsf{Sheaf}^{\uparrow}$ *carrier rows only from the indexed section carrier over* BHist *opens in Definition 151.154, together with the cover-nerve ledger of Definition 151.160. Their sheaf-side carrier scope is exactly the indexed open carrier, the section carriers* $\mathsf{Sec}(i)$*, the section classifiers* $\sim_i$*, the restriction maps* $\rho_{i,j}$*, cover-member cells, pair-overlap cells, and* hsame*-stable ledger transport. No host powerset equality, cover quotient, stalk quotient, or external sheafification row is exported to these consumers.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer carrier scope`

Proof. Read the carrier source from Definition 151.298. Its carrier clause points to Definition 151.154, so every section-level carrier read by a downstream consumer is one of the open-indexed section carriers, classifiers, restrictions, and identity or composition rows already listed there.

Read the history side from Definition 151.160 and the cover-nerve export surface of Definition 151.302. The only additional cells are cover-member cells, finite-intersection pair-overlap cells, route-history cells when present, and membership transport through hsame.

Finally apply the downstream coverage theorem Definition 151.305 and the projection seal Definition 151.252. These rows are projected read-only to $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, and topology pullback, so none of the three consumers obtains host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row from the root package. □

Theorem 151.136 (Sheaf root downstream consumer classifier stability)**.** *Every restriction route exposed from the* $\mathsf{Sheaf}^{\uparrow}$ *root to* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback preserves the relevant section classifier. One-step and two-step routes preserve classifiers by Definition 151.304; membership reindexing is transported only through* hsame*; optional* Cont *route histories compare displayed route witnesses without changing the target open or the target classifier. Hence consumer restriction routes do not add any host powerset equality, cover quotient, stalk quotient, or external sheafification row.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer classifier stability`

Proof. For section restrictions, apply Definition 151.304. It states that the one-step restriction of classifier-equal sections lands in the target classifier and that the composed two-step route agrees with the direct route in the classifier of the final open.

For membership movement, use the cover-nerve ledger of Definition 151.160. Its transport row is hsame-stable membership transport over BHist cells, so route changes do not replace the indexed open carrier by a host open or a quotient carrier.

If optional Cont histories are present, use the route-history clause cited in Definition 151.300. Those histories compare the witnesses for displayed routes; they do not alter the open index on which $\sim_j$ or $\sim_k$ is read.

The three projection seals Definition 151.253, Definition 151.254, and Definition 151.255 then pass the same classifier-stable routes to the downstream consumers without exporting host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row. □

Theorem 151.137 (Sheaf root downstream consumer exactness coverage)**.** *The* Sheaf$^{\uparrow}$ *exactness surface consumed by* RingedSpace$^{\uparrow}$*,* Scheme$^{\uparrow}$*, and topology pullback is exhausted by cover membership, locality and gluing, refinement descent, common-refinement exactness, and the downstream projection seals. Concretely, cover membership is supplied by Definition 151.299 and Definition 151.303; locality and gluing are supplied by Definition 151.301; refinement descent and refined exactness are supplied by Definition 151.176 and Definition 151.179; common-refinement exactness is supplied by Definition 151.188 and Definition 151.189; and consumer access is sealed by the three projection theorems. No host powerset equality, cover quotient, stalk quotient, or external sheafification row is exported.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer exactness coverage`

Proof. Start with cover membership. The accepted-cover cell is unpacked by Definition 151.299, and Definition 151.303 records the corresponding cover-member and pair-overlap cells inside the BHist cover nerve.

For local exactness, use Definition 151.301. It gives existence of a global readback from a compatible local family and classifier uniqueness for global readbacks whose restrictions agree on the cover.

For refinement, apply Definition 151.176 and the package theorem Definition 151.179. These rows keep the same global classifier class after passing to displayed refinements. For comparison of cover presentations, use Definition 151.188 and Definition 151.189.

Finally pass to consumers through Definition 151.253, Definition 151.254, and Definition 151.255. Each consumer receives a projection of the listed cover, restriction, locality, gluing, refinement, and common-refinement rows; none receives host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row. □

Definition 151.138 (Sheaf root joint consumer face)**.** *The* Sheaf$^{\uparrow}$ *root joint consumer face is the shared* BHist*-indexed projection package assembled from the projection seal of Definition 151.252, the consumer-threshold projection source of Definition 151.256, the root face exactness source of Definition 151.271, and the downstream consumption boundary of Definition 151.196. It has one root package and three read-only consumer faces: the* RingedSpace$^{\uparrow}$ *local-ring face, the* Scheme$^{\uparrow}$ *affine-chart face, and the topology pullback face. Each face reads the same* BHist*-scoped sheaf root rows for section carriers, section classifiers, restriction transport, cover-gluing exactness, restricted opens, displayed base change, cover presentations, and displayed common refinements.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace chart trace e0 endpoint absurd`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace normal form boundary`

Theorem 151.139 (Sheaf root joint consumer face exhaustion)**.** *Every sheaf-side row visible through the joint consumer face of Definition 151.275 to* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *or topology pullback consumers is one of the following rows: indexed section carrier, section classifier, restriction identity, restriction composition, restriction classifier transport, cover-locality and cover-gluing exactness, restricted-open carrier and inherited sheaf rows, displayed base-change transport, cover-presentation exactness, or displayed common-refinement comparison. No other sheaf-side row is exported by the joint face.*

Proof. Split over the three projections named in Definition 151.252.

For the $\mathsf{RingedSpace}^{\uparrow}$ projection, apply Definition 151.253. Its sheaf-side reads are the section carrier, classifier, restriction-stability, cover-gluing, restricted-open, and base-change rows. The root face coverage theorem Definition 151.272 places the carrier, classifier, and restriction rows inside the first three faces of the root exactness source.

For the $\mathsf{Scheme}^{\uparrow}$ projection, apply Definition 151.254. Chart-wise reads factor through the ringed-space projection, and the only additional sheaf-side comparisons are cover-presentation exactness and displayed common-refinement comparison. These are exactly the locality, gluing, and refinement rows supplied by Definition 151.273.

For the topology pullback projection, apply Definition 151.255. The available rows are pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and displayed common-refinement transport along the base-change surface.

Finally use Definition 151.199. It exhausts downstream sheaf reads by the carrier, classifier, restriction, cover-gluing, restricted-open, base-change, refinement, and exactness rows already listed in the statement, so the joint consumer face exports no further sheaf-side row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace exhaustion`

Theorem 151.140 (Sheaf root joint consumer face noninterference)**.** *None of the three consumer projections of Definition 151.275 exports another consumer's private data back into the* $\mathsf{Sheaf}^{\uparrow}$ *root face. The* $\mathsf{RingedSpace}^{\uparrow}$ *projection does not return private algebraic or local-ring rows to the root face; the* $\mathsf{Scheme}^{\uparrow}$ *projection does not return private affine-chart, spectra, or chart-ring rows; and the topology pullback projection does not return private pullback-open choices or consumer-specific pullback witnesses to the root face.*

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, the projection seal Definition 151.253 says that commutative-ring and local-ring data are supplied outside $\mathsf{Sheaf}^{\uparrow}$. Thus those algebraic rows are read by the consumer only after the sheaf-side projection has been fixed.

For $\mathsf{Scheme}^{\uparrow}$, the projection seal Definition 151.254 keeps spectra data and chart-ring data outside $\mathsf{Sheaf}^{\uparrow}$. The scheme face reads sheaf rows through chart-wise ringed-space projections and common-refinement comparisons, but it does not add a chart carrier, chart classifier, or spectra row to the root package.

For topology pullback, the projection seal Definition 151.255 restricts the face to displayed base-change, cover pullback, gluing pullback, and common-refinement transport. It explicitly does not read the ringed-space local-ring face or the scheme affine-chart face.

The separation theorem Definition 151.274 identifies all three consumers as projections of the same four root faces. Together with Definition 151.199, this prevents consumer-specific algebraic, chart, or pullback data from re-entering the $\mathsf{Sheaf}^{\uparrow}$ carrier, classifier, cover ledger, restriction-route face, or locality and gluing exactness face. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace noninterference`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace no feedback unary boundary`

Theorem 151.141 (Sheaf root joint consumer face NameCert boundary)**.** *The joint consumer face of Definition 151.275 exposes the following ten-field* $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ *boundary.*

1. *The traditional target is the sheaf-side consumer projection package read by ringed spaces, schemes, and topology pullback consumers.*
2. *The BEDC sources are* $\mathsf{Topology}^\uparrow$, $\mathsf{PreSheaf}^\uparrow$, *and* $\mathsf{NameCert}$.
3. *The carrier field is the indexed section presheaf carrier* $\mathsf{Sec}(i)$ *and its restricted-open or displayed base-change readbacks.*
4. *The classifier field is the section classifier relation* $\sim_i$ *and its restricted-open or displayed base-change transports.*
5. *The stability field consists of restriction identity, restriction composition, classifier transport, locality, gluing, refinement transport, and base-change transport.*
6. *The ledger field consists of the* BHist *cover nerve, cover-member rows, overlap rows,* hsame *transport, and optional route histories.*
7. *The exactness field consists of the joint exhaustion row of Definition 151.276 and the joint noninterference row of Definition 151.277.*
8. *The downstream-use field consists exactly of the* $\mathsf{RingedSpace}^\uparrow$ *local-ring face, the* $\mathsf{Scheme}^\uparrow$ *affine-chart face, and the topology pullback face.*
9. *The boundary exports no host powerset equality, cover quotient, stalk quotient, external sheafification operation, or consumer-private root data.*
10. *No standard bridge is attached to this boundary.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace semantic name certificate`

Proof. Begin with the root consumer-threshold projection source of Definition 151.256. It already names the traditional consumer target, the single threshold source, the carrier and classifier rows, the restriction row, the three consumer projection rows, the ledger and exactness row, and the absent standard-bridge field.

The root face exactness source of Definition 151.271 refines these fields into cover-membership, restriction-route, locality-gluing-refinement, and downstream projection faces. Use Definition 151.272 for carrier, classifier, cover ledger, and restriction-route coverage, and use Definition 151.273 for locality, gluing, refinement, cover-presentation, and base-change exactness.

The projection seals identify the downstream-use field. Apply Definition 151.253, Definition 151.254, and Definition 151.255 to split the boundary into the three consumer faces.

It remains to justify exactness and exclusion. The joint exhaustion theorem Definition 151.276 supplies the listed public sheaf rows, while the joint noninterference theorem Definition 151.277 and the downstream separation theorem Definition 151.274 remove consumer-private algebraic, chart, and pullback rows from the root face. The downstream consumption exhaustion theorem Definition 151.199 then leaves no standard bridge, host powerset equality, cover quotient, stalk quotient, or external sheafification operation in the boundary. □

Definition 151.142 (Sheaf root route exactness packet)**.** *The* $\mathsf{Sheaf}^\uparrow$ *root route exactness packet is the following ten-field* BHist*-scoped surface.*

1. *The traditional target is the sheaf route from an accepted indexed cover cell to the section row consumed by downstream certificates.*

2. *The BEDC source is the joint* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{NameCert}$ *surface, read over* BHist *histories and optional* Cont *route histories.*
3. *The carrier is the indexed-open carrier of* $\mathsf{Topology}^{\uparrow}$ *together with the section carrier* $\mathsf{Sec}(i)$ *of Definition 151.154.*
4. *The classifier is the open-indexed section relation* $\sim_i$ *on* $\mathsf{Sec}(i)$*, transported only along displayed restriction routes.*
5. *The stability rows are restriction identity, restriction composition, one-step classifier transport, two-step classifier transport, and route-history associativity when* Cont *rows are present.*
6. *The ledger rows are cover-member cells, pair-overlap cells, route witnesses, and* hsame *membership transport in the* BHist *cover nerve.*
7. *The exactness rows are cover-cell membership exhaustion, restriction-route classifier transport, locality and gluing, displayed refinement descent, and common-refinement comparison.*
8. *The downstream consumers are exactly the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and topology pullback constructions.*
9. *The packet contains no host powerset equality of open covers, no quotient of cover presentations, no stalk quotient, and no external sheafification operation.*
10. *The status is a root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *route-exactness surface: it records the rows available for downstream use and does not assert standard bridge closure.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootRouteExactnessPacket`

Theorem 151.143 (Sheaf route exactness membership transport)**.** *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an accepted indexed cover of* i *over the* BHist *topology ledger. For every displayed cell* $\mathsf{OpenAt}(i, b)$ *and every displayed restriction route from* i *to a cover member or pair-overlap row, the packet of Definition 151.290 transports the cover-member witness through the route and preserves the target section classifier on the reached open.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRouteExactness membership transport`

Proof. First build the cover witness. Apply Definition 151.303 to the accepted cover row. It sends the displayed cell $\mathsf{OpenAt}(i, b)$ to a member $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$ and supplies the finite-intersection row needed for comparison with any second member.

Next transport along the restriction route. Apply Definition 151.304. Its one-step and two-step clauses keep the section comparison inside the classifier of the reached member or overlap open.

The transported member witness and the transported classifier row use the same indexed open route. Since both cited rows are recorded over BHist and use hsame for membership transport, the result stays inside the cover nerve and introduces no host cover equality or cover quotient. □

Theorem 151.144 (Sheaf route exactness Cont composition)**.** *When optional* Cont *histories are present in the* $\mathsf{Sheaf}^{\uparrow}$ *route ledger, their composition rows compose displayed route witnesses only. They preserve the target open index and therefore preserve the target section classifier* $\sim_j$ *or* $\sim_k$ *already transported by restriction stability.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route history stability`

Proof. Start with the one-step route. The restriction stability obligation of Definition 151.300 gives classifier transport from $\sim_i$ to $\sim_j$ along $j \leq_{\mathcal{T}} i$.

For a composite route $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$, use the composition clause of Definition 151.300. The direct restriction from i to k and the route through j are compared in the same target classifier $\sim_k$.

Now read the optional Cont row. By Definition 151.304, such a history compares route witnesses and does not change the reached open. Hence route-history composition can only rebracket the displayed path; the classifier already attached to the reached open is unchanged. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route cont composition`

Theorem 151.145 (Sheaf route exactness NameCert consumption)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers receive exactly the route-exactness packet of Definition 151.290: cover-member cells, restriction classifier transport, optional* Cont *route-history composition, and the downstream projection rows sealed for those consumers. No consumer receives a larger sheaf interface from this packet.*

Proof. First identify the common downstream surface. Apply Definition 151.305. It lists the cover-member cells, pair-overlap cells, one-step and two-step restriction stability rows, optional Cont histories, and locality and gluing exactness rows available to downstream consumers.

For the ringed-space consumer, apply Definition 151.253. The local-ring face may read the sheaf section carrier, classifier, restriction stability, cover-gluing exactness, restricted-open rows, and displayed base-change rows, while local-ring structure itself is supplied outside $\mathsf{Sheaf}^{\uparrow}$.

For the scheme consumer, apply Definition 151.254. Its affine-chart read is routed through cover-presentation rows, displayed common-refinement rows, and the sealed ringed-space projection, so it receives the same route-exactness packet rather than a separate sheaf surface.

For topology pullback consumers, apply Definition 151.255. The pullback face reads pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and common-refinement transport along displayed base change.

The three projections exhaust the consumers named in Definition 151.290. Since each cited seal excludes host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure, downstream NameCert use is exactly the route-exactness packet and nothing larger. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRouteExactness namecert consumption`

Definition 151.146 (Sheaf root cover-descent source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root cover-descent source is the following* BHist*-scoped source tuple. It is read over a* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier with point carrier* BHist*, membership row* OpenAt*, inclusion relation* $\leq_{\mathcal{T}}$*, and finite-intersection rows; an indexed* $\mathsf{Sheaf}^{\uparrow}$ *carrier with section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, and restriction maps* $\rho_{i,j}$*; an indexed cover* $\mathcal{U} = (A, \iota, h, \gamma)$ *of an ambient open* i*; member inclusions* $h_a : \iota(a) \leq_{\mathcal{T}} i$*; displayed restriction routes to members and finite-intersection overlaps; a compatible local family* $s_a : \mathsf{Sec}(\iota(a))$*; and the section classifier rows used to compare restrictions on members, overlaps, and glued readbacks.*

The source admits only these displayed data: BHist *histories, indexed open membership, cover-member witnesses, member and overlap inclusions, restriction-route witnesses, compatible-family overlap equalities, locality, gluing, and section classifiers. It excludes stalks, quotient carriers, host powerset equality of covers, and external sheafification.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootCoverDescentSource`

Theorem 151.147 (Sheaf root cover-descent carrier exactness)**.** *The source of Definition 151.260 exposes exactly* BHist, OpenAt, *cover membership, restriction maps, finite-intersection overlap rows, and section classifiers. In particular, every carrier or classifier row visible to cover descent is one of the indexed-open carrier rows of* $\mathsf{Topology}^{\uparrow}$*, the indexed section rows of* $\mathsf{Sheaf}^{\uparrow}$*, or the compatible-family and locality rows of the sheaf cover surface.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent carrier exactness`

Proof. Unfold Definition 151.260. The topology component contributes only BHist, indexed opens, OpenAt, inclusion witnesses, and finite-intersection rows. These are exactly the displayed carrier and membership rows of Definition 88.1.

Unfold the indexed sheaf component. By Definition 151.154, the section-level rows are the carriers $\mathsf{Sec}(i)$, classifiers $\sim_i$, restriction maps $\rho_{i,j}$, and restriction stability fields attached to displayed inclusion routes.

Finally unfold the cover and local-family component through Definition 151.155. It adds cover membership, member inclusions, finite-intersection overlap comparisons, locality, and gluing. None of the three unfolded sources contains a stalk carrier, a quotient carrier, host powerset equality, or external sheafification, so the displayed list is exact. □

Theorem 151.148 (Sheaf root cover-descent gluing exhaustion)**.** *Every root cover-descent comparison route for the source of Definition 151.260 factors through a cover member or a displayed common refinement, then through restriction-route exactness, and finally through the locality or gluing classifier row. Thus the cover-descent route surface is exhausted by cover membership, finite-intersection overlap rows, restriction transport, point-germ readback, and gluing uniqueness.*

↪variant `BEDC.Derived.SheafUp.SheafBHistPointGermLedger cover descent exhaustion`

Proof. First apply the cover law γ in the indexed cover $\mathcal{U}$. Each history cell $b : \mathsf{BHist}$ lying in the ambient open is sent to a member $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$, and the finite-intersection rows of Definition 88.1 supply the displayed overlaps used for pairwise comparison.

Next route the comparison through the sheaf restriction surface. Apply Definition 151.291 to transport the member witness through the displayed route, and apply Definition 151.292 when the comparison is represented by a composed route history. Both steps keep the target section classifier attached to the reached open.

It remains to read the local family through gluing. Apply Definition 151.234 for the memberwise readback of a glued section, and apply Definition 151.169 to identify any two global readbacks that agree on all displayed members. The route therefore ends at the existing locality or gluing classifier row and introduces no further carrier or classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent common refinement germ exactness`

Theorem 151.149 (Sheaf root cover-descent downstream interface)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers may consume only the cover-descent source of Definition 151.260, the carrier exactness row of Definition 151.261, the gluing exhaustion row of Definition 151.262, and the downstream exactness coverage row of Definition 151.266. They receive no additional sheaf carrier and no additional sheaf classifier.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent downstream interface`

Proof. For the ringed-space consumer, apply Definition 151.224. Its sheaf-side data are section carriers, classifiers, restriction stability, point-germ readback, and local compatibility rows, all already present in the cover-descent source and its carrier exactness row.

For the scheme consumer, apply Definition 151.225. The chart gluing request is routed through cover membership, finite-intersection overlaps, compatible local families, gluing, and common-refinement comparison, which are precisely the rows exhausted by Definition 151.262.

For topology pullback, apply Definition 151.226. Pullback cover membership and compatible-family transport are displayed over the same indexed-open and restriction-route rows, so they add no carrier beyond Definition 151.260.

Finally apply Definition 151.266. It seals the three consumer projections to cover membership, locality and gluing, refinement descent, common-refinement exactness, and projection rows. Hence downstream access is exactly the cover-descent interface named in the statement. □

151.7 Downstream consumption boundary

Definition 151.150 (Sheaf downstream consumption boundary)**.** *The downstream consumption boundary exported by* $\mathsf{Sheaf}^{\uparrow}$ *to* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *is the following ten-field boundary.*

1. *The traditional target is an open-indexed sheaf of sections over an indexed open topology carrier.*
2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{NameCert}$.
3. *The carrier row is the indexed section presheaf carrier of Definition 151.154.*
4. *The classifier row is the section classifier family* $\sim_i$ *on* $\mathsf{Sec}(i)$ *from the same carrier.*
5. *The locality and gluing rows are the indexed-cover compatibility, locality, and gluing fields of Definition 151.155.*
6. *The cover ledger is the indexed cover row: cover indices, cover-member maps, inclusion witnesses, pointwise cover membership, and finite-intersection overlap rows.*
7. *The refinement ledger is the displayed refinement, refinement-tower, and common-refinement surface used by Definition 151.189.*
8. *The restricted-open ledger is the slice carrier and its inherited restriction, locality, and gluing rows from Definition 151.239.*
9. *The base-change ledger is the displayed base-change square and its cover-pullback, gluing-pullback, and common-refinement rows from Definition 151.208.*
10. *The exactness row is cover presentation exactness together with restricted common-refinement invariance, and no standard bridge is attached to this boundary.*

This boundary is the only sheaf-side interface available to downstream chapters: it names indexed sections, section classifiers, displayed cover rows, displayed refinements, restricted opens, and displayed base change, but it does not import a host-space model of sheaves.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDownstreamConsumptionBoundary`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafUp StdBridge`

Theorem 151.151 (Ringed-space consumption obligation)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *certificate consumes exactly the* $\mathsf{Sheaf}^{\uparrow}$ *carrier, section classifier, restriction stability, cover-gluing exactness, restricted-open surface, and base-change surface listed in Definition 151.196, together with the*

$\mathsf{CommRing}^{\uparrow}$ *rows asserting that the section carriers and stalk rows carry commutative-ring structure. On the sheaf side, every row required by* $\mathsf{RingedSpace}^{\uparrow}$ *is one of:*

1. *the indexed section carriers* $\mathsf{Sec}(i)$ *and classifiers* $\sim_i$*;*
2. *restriction identity, restriction composition, and restriction classifier transport;*
3. *locality, gluing, and cover-presentation exactness for indexed covers;*
4. *the restricted-open surface below a named open;*
5. *the displayed base-change surface for inverse-image and cover-pullback rows.*

No further sheaf row is needed to state the paper-side ringed-space certificate boundary.

Proof. The carrier and classifier rows are the first projections of the $\mathsf{Sheaf}^{\uparrow}$ boundary. They are the section carriers and section classifiers of Definition 151.154, while restriction identity, restriction composition, and classifier transport are the stability rows already exported in Definition 151.168.

For local ringed-space data, restrict the sheaf to each displayed open. The required open-local carrier, classifier, restriction, locality, gluing, and no-bridge rows are exactly the restricted-open surface of Definition 151.239.

For inverse-image or change-of-base comparisons, use the displayed base-change surface of Definition 151.208. The remaining rows mentioned in the statement are the commutative-ring and stalk rows supplied on the $\mathsf{CommRing}^{\uparrow}$ side; they do not ask $\mathsf{Sheaf}^{\uparrow}$ for any host powerset equality, cover quotient, external sheaf model, or standard bridge. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRingedSpaceConsumption obligation`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer chart endpoint classifier`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer RingedSpace boundary rows`

Theorem 151.152 (Scheme consumption obligation)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *certificate consumes the same* $\mathsf{Sheaf}^{\uparrow}$ *downstream boundary only through a cover by affine or ringed-space charts and the* $\mathsf{CommRing}^{\uparrow}$ *spectra rows attached to those charts. Concretely, each chart reads the sheaf rows through Definition 151.197; compatibility across chart overlaps uses the cover-presentation exactness and common-refinement gluing invariance rows of* $\mathsf{Sheaf}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeConsumption obligation`

Proof. On each affine or ringed-space chart, apply Definition 151.197. This identifies the sheaf-side data available to the chart as the carrier, classifier, restriction stability, cover-gluing exactness, restricted-open surface, and base-change surface in Definition 151.196.

On overlaps between charts, compare the two displayed cover presentations through a displayed common refinement. The equality of glued section classes is supplied by Definition 151.189, and the comparison of two gluings after passing to a common refinement is supplied by Definition 151.188.

The affine and spectra content is therefore external to the sheaf boundary: it comes from the ringed-space chart and $\mathsf{CommRing}^{\uparrow}$ spectra rows. The sheaf chapter contributes only the displayed indexed-cover, restricted-open, and base-change rows already enumerated in the boundary. □

Theorem 151.153 (Sheaf downstream consumption exhaustion)**.** *Every* $\mathsf{Sheaf}^{\uparrow}$ *row used by* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *in the two downstream consumption obligations is one of the rows in Definition 151.196. Consequently, the downstream chapters import no host powerset equality of opens, no quotient of cover presentations, no external sheaf model, and no bridge closure from* $\mathsf{Sheaf}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumption joint exhaustion`

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, inspect Definition 151.197. Its sheaf-side rows are precisely the carrier, classifier, restriction-stability, cover-gluing, restricted-open, and base-change entries of Definition 151.196. The commutative-ring and stalk rows named there are not sheaf rows.

For $\mathsf{Scheme}^{\uparrow}$, inspect Definition 151.198. Each chart first passes through the ringed-space consumption obligation, and every overlap comparison is discharged by cover-presentation exactness or common-refinement gluing invariance, both already included in the boundary's refinement and exactness rows.

Thus the union of sheaf rows consumed by the two downstream chapters is exhausted by the ten-field boundary. Since the boundary itself excludes host powerset equality, cover quotients, external sheaf models, and standard bridges, none of those structures is imported downstream through $\mathsf{Sheaf}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumption boundary joint exhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

Constructive story: A sheaf row starts with a $\mathsf{Topology}^{\uparrow}$ indexed-open carrier and a $\mathsf{PreSheaf}^{\uparrow}$ restriction surface, then attaches section carriers over each open index, section classifiers, restriction maps, indexed cover rows, overlap intersections, and the locality and gluing fields. The carrier admits exactly those indexed presheaf packets whose cover data is displayed by the topology ledger and whose local families satisfy the overlap compatibility rows; a cover presentation with missing inclusions, missing overlap comparisons, or no gluing/locality witness is outside the sheaf surface. Two sheaf presentations are classified together when the section classifiers identify the relevant global or restricted sections after transport through restrictions, refinements, common refinements, restricted opens, and base-change squares, not when host open covers or stalk quotients are identified. The source specification is the topology-plus-presheaf cover surface, the pattern specification records restriction identity and composition, restriction classifier stability, locality, gluing, refinement invariance, common-refinement exactness, point-germ rows, and downstream access traces, and the ledger keeps cover indices, member maps, inclusion proofs, overlap restrictions, base-change routes, and consumer projections visible. The five name-certificate obligations are supplied by the indexed section carrier and its classifier family: an admitted section surface inhabits the carrier, each local classifier is reflexive, symmetric, and transitive, and restriction/refinement transport keeps classified representatives inside the same displayed cover-gluing interface. A sheaf packet h lies in the signature gap of a $\mathsf{Sheaf}^{\uparrow}$ token p when the source-domain accepts its indexed cover and section data, the signature relation pairs it with an answer sheaf packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, the compatible local family, its gluing, and all refinement or restricted-open rows must be the ones encoded by p. The checked bridge here reads the same indexed section, restriction, locality, and gluing rows as the standard sheaf interface for downstream ringed-space and scheme consumers, while still refusing host powerset equality, cover quotients, stalk quotients, and external sheafification.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper-side package is the indexed sheaf downstream boundary of section 151.10, bound by Definition 151.247, Definition 151.251, Definition 151.276, and Definition 151.263.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SheafUp.SheafUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim a machine-checked bridge, host powerset equality, cover quotient, stalk quotient, external sheafification, $\mathsf{RingedSpace}^{\uparrow}$ closure, or $\mathsf{Scheme}^{\uparrow}$ closure.

Upgrade path: no further upgrade – chapter at top of formal axis

151.8 Carrier definition

151.9 The certificate

Definition 151.154 (Indexed section presheaf carrier)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *in the sense of Definition 88.1. Write*

$$j \leq_{\mathcal{T}} i :\Longleftrightarrow \forall x : X,\ \mathsf{OpenAt}(j, x) \Longrightarrow \mathsf{OpenAt}(i, x)$$

for inclusion of indexed opens. An indexed section presheaf carrier for $\mathsf{PreSheaf}^{\uparrow}$ *over* $\mathcal{T}$ *is a tuple*

$$\mathcal{P} = (\mathsf{Sec}, \sim, \rho, \mathsf{Eqv}, \mathsf{Id}, \mathsf{Comp}, \mathsf{Res})$$

with the following data. For every open index $i : \mathsf{OpenIx}$ *there is a section carrier* $\mathsf{Sec}(i)$ *and a classifier* $\sim_i$ *on* $\mathsf{Sec}(i)$*; the field* Eqv *says that each* $\sim_i$ *is reflexive, symmetric, and transitive. For every inclusion* $j \leq_{\mathcal{T}} i$ *there is a restriction map*

$$\rho_{i,j} : \mathsf{Sec}(i) \to \mathsf{Sec}(j).$$

The identity field says $\rho_{i,i}(s) \sim_i s$ *for every* $s : \mathsf{Sec}(i)$*, and the composition field says that whenever* $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$*,*

$$\rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

for every $s : \mathsf{Sec}(i)$*. The restriction classifier-stability field says that, for every inclusion* $j \leq_{\mathcal{T}} i$ *and every* $s, t : \mathsf{Sec}(i)$*,*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

This is the section-level concrete form of the contravariant restriction data described in chapter 150.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafIndexedSectionPresheafCarrier`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedSectionPresheafCarrier endpoint unary`

Definition 151.155 (Indexed sheaf cover, compatibility, and locality)**.** *Let* $\mathcal{P}$ *be an indexed section presheaf carrier over* $\mathcal{T}$ *as in Definition 151.154. An indexed open cover of an open index* $i : \mathsf{OpenIx}$ *is a tuple* $\mathcal{U} = (A, \iota, h, \gamma)$ *consisting of an indexing type* A*, a family* $\iota : A \to \mathsf{OpenIx}$*, inclusion witnesses*

$$h_a : \iota(a) \leq_{\mathcal{T}} i$$

for all $a : A$*, and the pointwise cover law*

$$\gamma : \forall x : X,\ \mathsf{OpenAt}(i, x) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), x).$$

For a family $s_a : \mathsf{Sec}(\iota(a))$*, compatibility over* $\mathcal{U}$ *means that for all* $a, b : A$ *the two restrictions to the indexed intersection agree:*

$$\rho_{\iota(a),\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)}(s_a) \sim_{\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)} \rho_{\iota(b),\iota(a)\wedge_{\mathsf{Topology}^{\uparrow}}\iota(b)}(s_b),$$

where the needed inclusions are supplied by the finite-intersection law of Definition 88.1. A sheaf locality field for $\mathcal{P}$ *says that, for every such cover* $\mathcal{U}$ *of* i *and every pair* $g, h : \mathsf{Sec}(i)$*,*

$$(\forall a : A,\ \rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h)) \Longrightarrow g \sim_i h.$$

A sheaf gluing field says that every compatible local family over $\mathcal{U}$ *has a section* $g : \mathsf{Sec}(i)$ *whose restriction to each* $\iota(a)$ *is classifier-equal to* s_a*. An indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *is an indexed section presheaf carrier equipped with both of these fields.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedCoverCompatibilityLocality common refinement`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafIndexedCoverCompatibilityLocality global rows`

Definition 151.156 (Stable sheaf restriction row)**.** *Let $\mathcal{P}$ be an indexed section presheaf carrier over $\mathcal{T}$ as in Definition 151.154. A stable sheaf restriction row for $\mathcal{P}$ is the field saying that classifier equality is preserved by every restriction: for every inclusion $j \leq_{\mathcal{T}} i$ and every pair $s, t : \mathsf{Sec}(i)$,*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

Thus section classifiers transport along the indexed open restriction maps carried by the presheaf surface.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafStableRestrictionRow restricted comparison readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafStableRestrictionRow transport`

Theorem 151.157 (Sheaf carrier support obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Then the carrier support exposed by $\mathcal{S}$ is only the indexed section presheaf carrier over the $\mathsf{Topology}^{\uparrow}$ open-index surface, together with the sheaf locality and gluing fields. Concretely, every carrier row is one of the section carriers $\mathsf{Sec}(i)$, section classifiers $\sim_i$, equivalence fields, restriction maps $\rho_{i,j}$, identity and composition laws, indexed open-cover data, compatible local-family predicates, locality implications, or gluing witnesses described in Definition 151.154 and Definition 151.155.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCarrierSupport obligation`

Proof. Project the indexed section presheaf component of $\mathcal{S}$. Its displayed rows are precisely the open-indexed section carriers, classifiers, equivalence witnesses, restriction maps, and restriction identity and composition laws of Definition 151.154.

Project the sheaf component of $\mathcal{S}$. By Definition 151.155, this contributes only indexed open covers, compatible local families, the locality field, and the gluing field. No additional carrier row is introduced beyond those two displayed interfaces, so the asserted carrier support is exhausted by their projections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCarrierSupport obligation rows`

Theorem 151.158 (Sheaf classifier transport obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$, and suppose its indexed section presheaf carrier carries a stable sheaf restriction row in the sense of Definition 151.156. Then section classifiers transport through every restriction. Moreover, for every indexed open cover $\mathcal{U} = (A, \iota, h, \gamma)$ of i and every pair $g, h' : \mathsf{Sec}(i)$,*

$$(\forall a : A,\ \rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h')) \Longrightarrow g \sim_i h'.$$

Thus restriction stability gives classifier transport along the presheaf maps, and sheaf locality turns coverwise classifier equality back into classifier equality on the covered open.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger classifier transport obligation`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted global transport closure`

Proof. For a single inclusion $j \leq_{\mathcal{T}} i$, the transport claim is exactly the stable restriction row of Definition 151.156: a classifier equality $s \sim_i t$ restricts to $\rho_{i,j}(s) \sim_j \rho_{i,j}(t)$. The source and target classifiers are those supplied by the section presheaf carrier, and their reflexive, symmetric, and transitive behavior is supplied by the equivalence field of Definition 151.154.

For a cover $\mathcal{U}$, assume the displayed coverwise equality of the restrictions of g and h'. This is exactly the antecedent required by the sheaf locality field in Definition 151.155. Applying that field over the same indexed cover gives $g \sim_i h'$, which is the global classifier equality claimed. □

Theorem 151.159 (Sheaf cover ledger exactness obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. An accepted cover and gluing ledger row for $\mathcal{S}$ consists exactly of indexed open-cover data $\mathcal{U} = (A, \iota, h, \gamma)$, the finite-intersection rows of the* $\mathsf{Topology}^{\uparrow}$ *carrier, a compatible local section family $s_a : \mathsf{Sec}(\iota(a))$, and a global section $g : \mathsf{Sec}(i)$ with displayed restriction equalities*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The ledger uses no host open-cover equality and no quotient primitive: all comparison data is expressed through indexed opens, restriction maps, and the section classifiers of Definition 151.154.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger exactness rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger exactness obligation`

Proof. Unfold the indexed open cover and compatibility clauses in Definition 151.155. The cover ledger records only the indexing type A, the open-index map ι, the inclusion witnesses $h_a : \iota(a) \leq_{\mathcal{T}} i$, and the pointwise cover law γ. Compatibility is recorded by restricting local sections to the finite-intersection index supplied by the $\mathsf{Topology}^{\uparrow}$ carrier and comparing them with the classifier on that intersection.

Unfold the gluing clause of the same definition. A gluing ledger row adds a section $g : \mathsf{Sec}(i)$ and the displayed restriction equalities against the compatible family. The theorem Definition 151.170 checks that restrictions of a global section produce a compatible local family, while Definition 151.169 checks that two displayed gluings of the same compatible family are classifier-equal.

The two definitions just unfolded contain no host-side equality of open covers and no quotient constructor. Their comparison mechanism is the indexed inclusion relation $j \leq_{\mathcal{T}} i$, the finite-intersection row of the topology carrier, the restriction maps, and the section classifiers. Hence the listed rows are exact for the accepted cover and gluing ledger. □

Definition 151.160 (Sheaf BHist cover nerve ledger)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *whose point carrier is* BHist*, whose membership classifier is* OpenAt*, and whose finite-intersection index is written $\wedge_{\mathsf{Topology}^{\uparrow}}$. Let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. The* BHist *cover nerve ledger of $\mathcal{U}$ is the finite-intersection ledger surface with the following rows.*

1. *For each $a : A$, the member row is the indexed open $\iota(a)$ with its inclusion witness $h_a : \iota(a) \leq_{\mathcal{T}} i$ and its visible membership predicate* $\mathsf{OpenAt}(\iota(a), b)$ *for $b :$* BHist*.*
2. *For each $a, b : A$, the pair-overlap row is $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, together with the two finite-intersection projections $k_{a,b} \leq_{\mathcal{T}} \iota(a)$ and $k_{a,b} \leq_{\mathcal{T}} \iota(b)$ and the two restriction rows from* $\mathsf{Sec}(\iota(a))$ *and* $\mathsf{Sec}(\iota(b))$ *to* $\mathsf{Sec}(k_{a,b})$*.*
3. *For each $a, b, c : A$, the triple-overlap row is $\ell_{a,b,c} := k_{a,b} \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c)$, together with the finite-intersection projections $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b}$ and $\ell_{a,b,c} \leq_{\mathcal{T}} \iota(c)$ and the restriction rows obtained by composing through $k_{a,b}$ or directly from any member whose inclusion route is displayed.*

The ledger contains only OpenAt *rows, finite-intersection projections, inclusion witnesses, presheaf restriction rows, section classifiers, and visible* BHist *provenance. It does not identify cover presentations by host equality and does not quotient overlap rows.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger`

Theorem 151.161 (Sheaf BHist cover nerve overlap projections)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist*, let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. For a local family $s_a : \mathsf{Sec}(\iota(a))$, compatibility over $\mathcal{U}$ is exactly the assertion that every pair-overlap row of the* BHist *cover nerve ledger carries classifier equality between the two projected restrictions:*

$$\forall a, b : A, \quad \rho_{\iota(a),k_{a,b}}(s_a) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(s_b),$$

where $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ and the inclusions are the finite-intersection projections of Definition 88.1.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison overlap projection rows`

Proof. Unfold the compatibility predicate in Definition 151.155. Its only comparison clauses are indexed by pairs $a, b : A$, and each clause restricts s_a and s_b to the common intersection $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ before comparing them by the section classifier at that intersection.

Unfold the cover nerve ledger of Definition 151.160. Its pair-overlap rows are exactly the same indices $k_{a,b}$, with the same two finite-intersection projections supplied by Definition 88.1. Therefore every compatibility clause is a classifier equality on one pair-overlap ledger row.

Conversely, assume every pair-overlap ledger row carries the displayed classifier equality. Reading those rows through the same two projections gives precisely the clauses required by Definition 151.155. Restriction composition is not extra data here; it only confirms that any displayed route to $k_{a,b}$ reads as the same section classifier row as the direct pair-overlap restriction. □

Theorem 151.162 (Sheaf BHist cover nerve gluing readback)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist*, let $\mathcal{S}$ be an indexed sheaf carrier over $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ is a gluing section with*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

Then the BHist *cover nerve ledger reads g back to the local family on every member row and to the induced compatible family on every pair-overlap row. Explicitly, for every $a : A$,*

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a,$$

and for every $a, b : A$, writing $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$,

$$\rho_{\iota(a),k_{a,b}}(\rho_{i,\iota(a)}(g)) \sim_{k_{a,b}} \rho_{\iota(a),k_{a,b}}(s_a)$$

and

$$\rho_{\iota(b),k_{a,b}}(\rho_{i,\iota(b)}(g)) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(s_b).$$

Consequently the two restrictions of g to each pair-overlap ledger row recover the same classifier row as the compatible family there.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger gluing readback`

Proof. The member readback is the gluing equality displayed in the statement. It is exactly the gluing clause of Definition 151.155 evaluated at the member row $\iota(a)$.

For a pair $a, b : A$, write $k = \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$. The finite-intersection field of Definition 88.1 supplies $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$, which are the two pair-overlap projections recorded by Definition 151.160.

Transport the gluing equality $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$ along $k \leq_{\mathcal{T}} \iota(a)$ using the restriction classifier-stability field from Definition 151.154. This gives the first pair-overlap readback equality. The same transport along $k \leq_{\mathcal{T}} \iota(b)$ gives the second one.

It remains to compare the two readings of g on k. Restriction composition identifies the restriction of $\rho_{i,\iota(a)}(g)$ to k and the restriction of $\rho_{i,\iota(b)}(g)$ to k with the direct restriction $\rho_{i,k}(g)$, up to the classifier $\sim_k$. The compatibility of s_a gives the corresponding equality between the two local-family restrictions. Transitivity and symmetry of $\sim_k$ combine these rows, so the pair-overlap ledger reads the gluing section back to the same classifier row as the compatible local family. □

Theorem 151.163 (Sheaf NameCert carrier obligation)**.** *Every indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$ *determines the carrier-facing rows required by the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *template. The rows are fixed as follows.*

1. *The traditional target is sheaf-theoretic section data over open covers.*
2. *The BEDC source is the presheaf restriction source of* $\mathsf{PreSheaf}^{\uparrow}$ *together with the indexed-open source of* $\mathsf{Topology}^{\uparrow}$*.*
3. *The carrier is the indexed section presheaf of Definition 151.154, equipped with the locality and gluing fields of Definition 151.155.*
4. *The classifier row is the family of section classifiers* $\sim_i$*.*
5. *The stability row is supplied by the restriction identity, restriction composition, restriction classifier-stability, and classifier-equivalence fields.*
6. *The ledger is the displayed data of covers, intersections, inclusion witnesses, and restriction maps.*
7. *The exactness row is visible cover-local readback: compatible local sections glue to a global section and locality identifies any two global readbacks with the same cover restrictions.*
8. *The standard bridge row is absent; this certificate surface records the BEDC sheaf interface without claiming an external set-theoretic sheaf bridge.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger semantic name certificate`

Proof. Unpack the indexed sheaf carrier. Its presheaf part is exactly the tuple $\mathcal{P} = (\mathsf{Sec}, \sim, \rho, \mathsf{Eqv}, \mathsf{Id}, \mathsf{Comp}, \mathsf{Res})$ of Definition 151.154, so the carrier, classifier, and restriction-stability rows are not extra data: they are the displayed section carriers, displayed section classifiers, and the identity, composition, equivalence, and restriction classifier-stability fields.

Unpack the cover part. The topology side contributes open indices, inclusions, and finite-intersection witnesses by Definition 88.1; the presheaf side contributes contravariant restriction scope by chapter 150. The remaining sheaf fields in Definition 151.155 are precisely locality and gluing over those covers, which give the visible cover-local readback row and leave no separate standard-bridge row to construct. □

Theorem 151.164 (Sheaf restriction stability obligation)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$*. For inclusions* $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$*, restriction identity and restriction composition keep every section row inside the displayed classifier family. Explicitly, for* $s : \mathsf{Sec}(i)$ *one has*

$$\rho_{i,i}(s) \sim_i s$$

and

$$\rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s).$$

When k is an indexed intersection supplied by the finite-intersection field of Definition 88.1, the same restriction-composition statement applies to the two induced inclusion chains, so cover restrictions remain stable under the topology intersection witnesses.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restriction stability obligation`

Proof. The first displayed relation is the identity field of the indexed section presheaf carrier in Definition 151.154. The second displayed relation is the composition field for the chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. Both conclusions land in the classifier specified by the target open index, namely $\sim_i$ for identity and $\sim_k$ for composition.

For the intersection case, use Definition 88.1 to obtain the open-index intersection and its two projection inclusions. Combining either projection with a cover inclusion gives one of the admissible chains governed by the contravariant restriction interface of chapter 150. Applying the same composition field to those chains gives stability on every displayed intersection row. □

Theorem 151.165 (Sheaf locality and gluing NameCert obligation)**.** *For an indexed sheaf carrier $\mathcal{S}$ for $\mathsf{Sheaf}^{\uparrow}$, the locality and gluing fields supply the certificate rows consumed by Definition 151.169, Definition 151.170, and Definition 151.171. Namely, every compatible local family over an indexed open cover has a global readback section, and locality identifies any two global sections whose restrictions agree on every cover member.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison locality gluing semantic name certificate`

Proof. Fix an indexed cover $\mathcal{U} = (A, \iota, h, \gamma)$ of i and a compatible family $s_a : \mathsf{Sec}(\iota(a))$. The cover and compatibility predicates are those of Definition 151.155; their intersection clauses are meaningful because Definition 88.1 supplies finite-intersection witnesses, and their restriction clauses live in the contravariant presheaf scope of chapter 150.

Apply the gluing field to obtain a section $g : \mathsf{Sec}(i)$ whose restrictions classify as the family s_a. Apply the locality field to any pair of such global sections whose memberwise restrictions agree; this is exactly the premise used by Definition 151.169. The compatibility theorem uses the same restriction-composition row on intersections, and the idempotence corollary repacks gluing plus uniqueness for the local family obtained from one global section. □

Theorem 151.166 (SheafUp BHist obligation surface)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^{\uparrow}$ whose point carrier is BHist, with classifier hsame, membership row OpenAt, and finite-intersection index $\wedge_{\mathsf{Topology}^{\uparrow}}$. Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over $\mathcal{T}$ in the sense of Definition 151.154 and Definition 151.155. Then $\mathcal{S}$ supplies the $\mathsf{Sheaf}^{\uparrow}$ carrier, section classifiers, restriction maps, restriction identity, restriction composition, indexed-cover rows, compatibility predicate, locality field, gluing field, and visible BHist ledger rows required by the paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface. The exposed data depend only on $\mathsf{NameCert}_{\mathsf{PreSheaf}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$, the row OpenAt, the finite-intersection rows of $\mathcal{T}$, and the classifier-equivalence fields on the section carriers.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistObligationSurface restriction cover package`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistObligationSurface cover restriction classifier`

Proof. Project the carrier and classifier rows from the indexed section presheaf carrier. For each open index i, the section carrier $\mathsf{Sec}(i)$ and classifier $\sim_i$ are visible, and the equivalence field supplies the reflexive, symmetric, and transitive classifier rules required by the certificate surface.

Project the restriction rows from the same tuple. The identity and composition fields are exactly the $\mathsf{PreSheaf}^{\uparrow}$ restriction identity and restriction composition obligations, with inclusions interpreted through the indexed-open preorder $j \leq_{\mathcal{T}} i$ defined by OpenAt. The finite-intersection row of $\mathcal{T}$ supplies the common target index used in compatibility checks.

Finally project the cover, compatibility, locality, and gluing rows from Definition 151.155. The visible ledger is BHist-scoped because every open membership and every inclusion witness is expressed by OpenAt on BHist, and no host powerset equality is part of the exposed surface. □

Theorem 151.167 (SheafUp restriction hsame transport obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over a BHist-scoped indexed open topology carrier $\mathcal{T}$. Suppose a second displayed carrier is obtained by componentwise hsame transport of the topology row OpenAt, the section carriers and classifiers, the restriction maps, and the indexed-cover rows. Then restriction compatibility, locality, and gluing for $\mathcal{S}$ transport to the displayed carrier. In particular, a compatible family remains compatible after transport, a locality hypothesis on transported restrictions yields transported classifier equality of global sections, and a gluing witness transports to a gluing witness for the transported local family.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger hsame transport readback`

Proof. Transport the topology row first. The $\mathsf{Topology}^{\uparrow}$ membership-stability field moves each occurrence of $\mathsf{OpenAt}(i, h)$ across the displayed hsame witness, so the inclusion witnesses $j \leq_{\mathcal{T}} i$ and the cover law for $\mathcal{U} = (A, \iota, \eta, \gamma)$ remain valid after transport.

Next transport the section rows. The componentwise transport hypothesis identifies each section carrier, classifier, and restriction map with its transported counterpart. Compatibility is a conjunction of classifier equalities after restricting to the finite-intersection index, so it follows by applying the transported restriction rows and then reading the section classifier equivalence fields in the transported carrier.

For locality, begin with transported agreement of two global sections on every cover member. Move that agreement back along the componentwise classifier transport, apply the locality field of $\mathcal{S}$, and transport the resulting global classifier equality forward. For gluing, transport the compatible family back, apply the gluing field of $\mathcal{S}$ to obtain a global section, and transport that witness and its restriction equalities forward. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger restriction hsame transport obligation`

Theorem 151.168 (SheafUp cover gluing NameCert export)**.** *The BHist-scoped surface of Definition 151.166, together with the transport obligation of Definition 151.167, the uniqueness theorem Definition 151.169, and the global-restriction compatibility theorem Definition 151.170, assembles the paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface consumed by $\mathsf{RingedSpace}^{\uparrow}$ and $\mathsf{Scheme}^{\uparrow}$. The exported surface provides sheaf carrier data, restriction coherence, cover compatibility, locality, gluing, uniqueness up to section classifier equality, and compatibility of restrictions of global sections; it claims no standard bridge and no external powerset equality.*

Proof. Start with the obligation surface of Definition 151.166. It supplies the carrier, classifier, restriction, cover, compatibility, locality, gluing, and visible ledger rows over BHist, all routed through $\mathsf{NameCert}_{\mathsf{PreSheaf}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$.

Use Definition 151.167 to make those rows stable under the componentwise hsame transports seen by downstream chapters. This gives $\mathsf{RingedSpace}^{\uparrow}$ and $\mathsf{Scheme}^{\uparrow}$ a transport discipline for topology membership through OpenAt and for section equality through the displayed section classifiers.

The gluing consequences are then the two local theorems already present in the chapter. Apply Definition 151.169 for uniqueness of a glued global section up to the section classifier, and apply Definition 151.170 for compatibility of the local family obtained by restricting a global section.

Combining these rows gives exactly the paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface needed by the dependent ringed-space and scheme certificates. Since every cited row stays inside indexed opens, OpenAt, hsame, and section classifiers, the package contains neither a standard sheaf bridge nor an equality of host powersets. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverGluingNameCert export`

Theorem 151.169 (Sheaf gluing uniqueness from locality)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $s_a : \mathsf{Sec}(\iota(a))$ be a compatible local family over $\mathcal{U}$. If $g, h : \mathsf{Sec}(i)$ both restrict to that family on every cover member, in the sense that*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$$

and

$$\forall a : A, \quad \rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a,$$

then the two cover-global sections are classifier-equal:

$$g \sim_i h.$$

Thus the gluing supplied by the $\mathsf{Sheaf}^{\uparrow}$ carrier is unique up to the section classifier on the covered open.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingClassifier uniqueness from locality`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger gluing uniqueness from locality`

Proof. Fix a cover member $a : A$. The first restriction hypothesis gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The second restriction hypothesis gives

$$\rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a.$$

By symmetry of the section classifier on $\mathsf{Sec}(\iota(a))$, supplied by the equivalence field of Definition 151.154, this yields

$$s_a \sim_{\iota(a)} \rho_{i,\iota(a)}(h).$$

Transitivity of the same classifier then gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h).$$

Since a was arbitrary, the restrictions of g and h agree on every member of $\mathcal{U}$. Applying the sheaf locality field from Definition 151.155 gives $g \sim_i h$. □

Theorem 151.170 (Sheaf global restrictions form a compatible local family)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $g : \mathsf{Sec}(i)$. Define the local family $s_a : \mathsf{Sec}(\iota(a))$ by*

$$s_a = \rho_{i,\iota(a)}(g).$$

Then s_a is compatible over $\mathcal{U}$ in the sense of Definition 151.155. Explicitly, for every $a, b : A$, putting $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, one has

$$\rho_{\iota(a),k_{a,b}}(\rho_{i,\iota(a)}(g)) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(\rho_{i,\iota(b)}(g)),$$

where the restriction maps use the inclusions supplied by the finite-intersection field and the cover inclusions h_a and h_b.

Proof. Fix $a, b : A$ and write $k = \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$. The finite-intersection field of Definition 88.1 gives

$$\mathsf{OpenAt}(k, x) \Longrightarrow (\mathsf{OpenAt}(\iota(a), x) \wedge \mathsf{OpenAt}(\iota(b), x))$$

for every $x : X$. Projecting the two components gives inclusions $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$. Together with the cover inclusions $h_a : \iota(a) \leq_{\mathcal{T}} i$ and $h_b : \iota(b) \leq_{\mathcal{T}} i$, these are the two chains needed for presheaf restriction composition.

Apply the composition field of Definition 151.154 to the chain $k \leq_{\mathcal{T}} \iota(a) \leq_{\mathcal{T}} i$ and the section g. This gives

$$\rho_{\iota(a),k}(\rho_{i,\iota(a)}(g)) \sim_k \rho_{i,k}(g).$$

Applying the same composition field to $k \leq_{\mathcal{T}} \iota(b) \leq_{\mathcal{T}} i$ gives

$$\rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)) \sim_k \rho_{i,k}(g).$$

By symmetry of the section classifier on $\mathsf{Sec}(k)$, supplied by the equivalence field of Definition 151.154, the second displayed relation yields

$$\rho_{i,k}(g) \sim_k \rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)).$$

Transitivity of the same classifier combines this with the first displayed relation and gives

$$\rho_{\iota(a),k}(\rho_{i,\iota(a)}(g)) \sim_k \rho_{\iota(b),k}(\rho_{i,\iota(b)}(g)).$$

Since a and b were arbitrary, this is exactly compatibility of the family $a \mapsto \rho_{i,\iota(a)}(g)$ over $\mathcal{U}$ as defined in Definition 151.155. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger global restrictions compatible`

Corollary 151.171 (Sheaf gluing of global restrictions is idempotent)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, \eta, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$, and let $g : \mathsf{Sec}(i)$. Define the local family $s_a : \mathsf{Sec}(\iota(a))$ by*

$$s_a = \rho_{i,\iota(a)}(g).$$

If $h : \mathsf{Sec}(i)$ is a gluing of this family, in the sense that

$$\forall a : A, \quad \rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a,$$

then

$$h \sim_i g.$$

Proof. By Definition 151.170, the family s_a is compatible over $\mathcal{U}$. For each $a : A$, the displayed definition of s_a and reflexivity of the classifier on $\mathsf{Sec}(\iota(a))$, supplied by Definition 151.154, give

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

The gluing hypothesis gives

$$\rho_{i,\iota(a)}(h) \sim_{\iota(a)} s_a$$

for the same a. Since a was arbitrary, both cover-global sections h and g restrict to the compatible local family s_a on every member of $\mathcal{U}$. Applying Definition 151.169 with h as the first cover-global section and g as the second gives $h \sim_i g$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingGlobalRestrictions idempotent`

Theorem 151.172 (Sheaf restriction classifier transport)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. For every inclusion $j \leq_{\mathcal{T}} i$ and every pair $s, t : \mathsf{Sec}(i)$, classifier equality on the larger open transports along restriction:*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t).$$

Thus restriction rows are morphisms of the local section classifiers.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restriction classifier transport`

Proof. The indexed sheaf carrier contains the indexed section presheaf carrier of Definition 151.154. Apply its restriction classifier-stability field to the inclusion $j \leq_{\mathcal{T}} i$ and the given classifier row $s \sim_i t$. □

Theorem 151.173 (Sheaf compatible-family stability)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of an open index $i : \mathsf{OpenIx}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $t_a : \mathsf{Sec}(\iota(a))$ is a second local family with*

$$\forall a : A, \quad t_a \sim_{\iota(a)} s_a.$$

Then t_a is compatible over $\mathcal{U}$. Explicitly, for all $a, b : A$, writing $k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$, one has

$$\rho_{\iota(a),k_{a,b}}(t_a) \sim_{k_{a,b}} \rho_{\iota(b),k_{a,b}}(t_b).$$

Proof. Fix $a, b : A$ and write $k = \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$. The finite-intersection field of Definition 88.1 supplies the inclusions $k \leq_{\mathcal{T}} \iota(a)$ and $k \leq_{\mathcal{T}} \iota(b)$ needed to restrict both local sections to the overlap.

Apply Definition 151.172 to the classifier row $t_a \sim_{\iota(a)} s_a$ along $k \leq_{\mathcal{T}} \iota(a)$. This gives

$$\rho_{\iota(a),k}(t_a) \sim_k \rho_{\iota(a),k}(s_a).$$

Apply the same transport row to $t_b \sim_{\iota(b)} s_b$ along $k \leq_{\mathcal{T}} \iota(b)$, and then use symmetry of the classifier on $\mathsf{Sec}(k)$, to get

$$\rho_{\iota(b),k}(s_b) \sim_k \rho_{\iota(b),k}(t_b).$$

The compatibility of the original family gives

$$\rho_{\iota(a),k}(s_a) \sim_k \rho_{\iota(b),k}(s_b).$$

Transitivity of $\sim_k$ first composes the transported a-side row with the original compatibility row, and then composes the result with the reversed transported b-side row. The resulting row is exactly

$$\rho_{\iota(a),k}(t_a) \sim_k \rho_{\iota(b),k}(t_b).$$

Since a and b were arbitrary, the family t_a is compatible over $\mathcal{U}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafGluingCompatibleFamily stability semantic certificate`

Theorem 151.174 (Sheaf cover refinement obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. A displayed refinement of $\mathcal{V}$ through $\mathcal{U}$ is a map $r : B \to A$ together with inclusion witnesses*

$$\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$$

for every $b : B$, compatible with the two cover rows in the sense that the composite inclusion $\kappa(b) \leq_{\mathcal{T}} \iota(r(b)) \leq_{\mathcal{T}} i$ is the displayed route from the refined cover member to i. This is a certificate-level obligation only over the indexed-open inclusion relation of Definition 151.155 and the carrier rows of Definition 88.1; it introduces no host powerset equality of covers and no quotient comparison between cover presentations.

Proof. Unpack the two cover rows. The original cover supplies $h_a : \iota(a) \leq_{\mathcal{T}} i$, and the refined cover supplies $q_b : \kappa(b) \leq_{\mathcal{T}} i$. A refinement chooses, for each refined row b, an original row $r(b)$ together with the displayed inclusion $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$.

The topology carrier already gives the meaning of $\leq_{\mathcal{T}}$ as indexed-open inclusion and supplies the finite-intersection rows used by the sheaf compatibility predicate. Therefore the refinement datum is fully expressed by the map r and the witnesses ϵ_b, with the target cover inclusion $h_{r(b)}$ recording the route onward to i.

No comparison of host open subsets is used. All later transport across the refinement is forced to pass through ϵ_b, restriction composition, and the section classifiers of the indexed presheaf carrier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace refinement obligation`

Theorem 151.175 (Sheaf compatible family pullback along refinement)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$ in the sense of Definition 151.174. If $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$, then the pulled-back family*

$$t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$$

is compatible over $\mathcal{V}$. Explicitly, for every $b, c : B$, writing $m_{b,c} := \kappa(b) \wedge_{\mathsf{Topology}^{\uparrow}} \kappa(c)$, one has

$$\rho_{\kappa(b),m_{b,c}}(t_b) \sim_{m_{b,c}} \rho_{\kappa(c),m_{b,c}}(t_c).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger refinement pullback`

Proof. Fix $b, c : B$ and write $m = \kappa(b) \wedge_{\mathsf{Topology}^{\uparrow}} \kappa(c)$. The finite-intersection field of Definition 88.1 gives $m \leq_{\mathcal{T}} \kappa(b)$ and $m \leq_{\mathcal{T}} \kappa(c)$. Composing these with the refinement inclusions gives routes $m \leq_{\mathcal{T}} \iota(r(b))$ and $m \leq_{\mathcal{T}} \iota(r(c))$.

Use restriction composition on the chain $m \leq_{\mathcal{T}} \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$. It identifies the left restriction of t_b with the direct restriction of $s_{r(b)}$ to m, up to the classifier $\sim_m$. The same argument on the c side identifies the right restriction of t_c with the direct restriction of $s_{r(c)}$ to m.

The original compatibility of s_a over $\mathcal{U}$ gives equality after restricting $s_{r(b)}$ and $s_{r(c)}$ to the overlap $\iota(r(b)) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(r(c))$. Since m lies inside both original cover members by the refinement inclusions and the finite-intersection projections, restriction composition transports that compatibility row down to m.

Finally, transitivity of $\sim_m$ composes the two restriction-composition rows with the transported original compatibility row. The resulting equality is precisely the compatibility clause for t_b and t_c over the refined cover $\mathcal{V}$. □

Theorem 151.176 (Sheaf gluing descends along refined covers)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Suppose $s_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues it:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

Then g glues the pulled-back family $t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$ over $\mathcal{V}$:

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} t_b.$$

Proof. Fix $b : B$. The gluing row over $\mathcal{U}$ gives

$$\rho_{i,\iota(r(b))}(g) \sim_{\iota(r(b))} s_{r(b)}.$$

Apply restriction classifier transport along the refinement inclusion $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b))$ to obtain equality after restriction to $\kappa(b)$.

Restriction composition on the chain $\kappa(b) \leq_{\mathcal{T}} \iota(r(b)) \leq_{\mathcal{T}} i$ identifies the restriction of $\rho_{i,\iota(r(b))}(g)$ to $\kappa(b)$ with the direct restriction $\rho_{i,\kappa(b)}(g)$. By definition, the restriction of $s_{r(b)}$ along the same refinement inclusion is t_b.

Transitivity of the classifier on $\mathsf{Sec}(\kappa(b))$ composes the composition row with the transported gluing row. Since b was arbitrary, the displayed equality holds for every member of $\mathcal{V}$, so g glues the pulled-back family over the refined cover. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementGluing descent cont`

Theorem 151.177 (Sheaf locality detection under refinement)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Suppose $g, h' : \mathsf{Sec}(i)$ satisfy refined-cover equality*

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} \rho_{i,\kappa(b)}(h').$$

Assume moreover that every original cover member is detected by its refinement rows: for every $a : A$, the subfamily of indices $b : B$ with $r(b) = a$ is an indexed open cover of $\iota(a)$ through the inclusions $\epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(a)$. Then locality over $\mathcal{U}$ gives

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementLocality global detection`

Proof. Fix $a : A$. By the detection hypothesis, the rows with $r(b) = a$ form an indexed cover of $\iota(a)$. For each such row b, the refined equality compares the direct restrictions of g and h' to $\kappa(b)$.

Restriction composition on $\kappa(b) \leq_{\mathcal{T}} \iota(a) \leq_{\mathcal{T}} i$ rewrites those direct restrictions as the restrictions of $\rho_{i,\iota(a)}(g)$ and $\rho_{i,\iota(a)}(h')$ along ϵ_b. Thus the two sections $\rho_{i,\iota(a)}(g)$ and $\rho_{i,\iota(a)}(h')$ agree on every member of the detected cover of $\iota(a)$.

Apply the sheaf locality field of Definition 151.155 on this detected cover of $\iota(a)$. It gives

$$\rho_{i,\iota(a)}(g) \sim_{\iota(a)} \rho_{i,\iota(a)}(h').$$

Since a was arbitrary, these equalities hold over all members of $\mathcal{U}$.

Applying the same locality field over the original cover $\mathcal{U}$ yields $g \sim_i h'$, which is the claimed global classifier equality. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refined locality detection`

Theorem 151.178 (Sheaf refined-cover gluing uniqueness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. Let $s_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$, and write*

$$t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$$

for its pulled-back family over $\mathcal{V}$. Suppose $g, h' : \mathsf{Sec}(i)$ both glue this pulled-back family:

$$\forall b : B, \quad \rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} t_b, \qquad \forall b : B, \quad \rho_{i,\kappa(b)}(h') \sim_{\kappa(b)} t_b.$$

Assume moreover that every original cover member is detected by its refinement rows in the sense of Definition 151.177. Then

$$g \sim_i h'.$$

Proof. Fix $b : B$. The first gluing row compares $\rho_{i,\kappa(b)}(g)$ with t_b, and the second compares $\rho_{i,\kappa(b)}(h')$ with the same section t_b.

Use symmetry of the classifier on $\mathsf{Sec}(\kappa(b))$ to reverse the second equality, then transitivity to obtain

$$\rho_{i,\kappa(b)}(g) \sim_{\kappa(b)} \rho_{i,\kappa(b)}(h').$$

Since b was arbitrary, this is refined-cover equality of g and h' over $\mathcal{V}$.

Apply Definition 151.177 to this refined-cover equality and to the displayed detection hypothesis for the rows with $r(b) = a$. It returns the global classifier equality $g \sim_i h'$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinedCover gluing uniqueness`

Theorem 151.179 (Sheaf refined-cover exactness obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let (r, ϵ) be a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$. For every compatible family $s_a : \mathsf{Sec}(\iota(a))$ over $\mathcal{U}$, the pulled-back family $t_b := \rho_{\iota(r(b)),\kappa(b)}(s_{r(b)})$ satisfies the following three exactness rows.*

1. *The family t_b is compatible over $\mathcal{V}$.*
2. *Every gluing $g : \mathsf{Sec}(i)$ of s_a over $\mathcal{U}$ also glues t_b over $\mathcal{V}$.*
3. *If every original cover member is detected by its refinement rows, then any two sections $g, h' : \mathsf{Sec}(i)$ gluing t_b over $\mathcal{V}$ satisfy $g \sim_i h'$.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinedCover exactness obligation`

Proof. The first row is exactly Definition 151.175: compatibility of the original family transports along the displayed refinement.

The second row is exactly Definition 151.176: a section gluing the original family restricts to a gluing of the pulled-back family over the refined cover.

The third row is Definition 151.178. Its hypotheses are precisely two displayed gluings of the same pulled-back family, plus the locality-detection row for the refinement fibers over each original cover member. □

Theorem 151.180 (Sheaf NameCert refinement exactness export)**.** *The paper-side obligation surface* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *exports refined-cover exactness over indexed opens. Concretely, for any indexed sheaf carrier over an indexed open topology carrier, every displayed refinement of indexed covers carries compatible families forward, descends original gluings to refined gluings, and makes any two refined gluings of the same pulled-back family globally classifier-equal whenever the original members are detected by their refinement rows. This exported row uses only* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *restriction composition, section classifiers, locality, and gluing; it does not use host powerset equality or quotient comparison.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCertRefinement exactness export`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCertRefinementExactness export`
↪ variant `BEDC.Derived.SheafUp.SheafRefinementExactness namecert export`

Proof. Project the carrier row from Definition 151.154. This supplies the open-indexed section carriers, classifiers, restriction maps, restriction composition, and classifier transport used in the refinement arguments.

Project the cover, compatibility, locality, and gluing rows from Definition 151.155. These rows supply indexed covers, overlap compatibility, the local-to-global classifier implication, and displayed gluing witnesses.

Assemble the export by applying Definition 151.179. Its three rows are already expressed in the projected data, so $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ exposes a single refined-cover exactness obligation to downstream ringed-space and scheme certificates without adding a host-level cover comparison. □

Theorem 151.181 (Sheaf cover self-refinement identity)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$*. Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an indexed open cover of* $i : \mathsf{OpenIx}$*. Then the identity map*

$$\mathrm{id}_A : A \to A$$

together with the pointwise reflexive inclusion witnesses

$$\epsilon_a : \iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a))$$

forms a displayed refinement of $\mathcal{U}$ *through itself in the sense of Definition 151.174.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCover self refinement`

Proof. Unpack Definition 151.174. To exhibit a displayed refinement of $\mathcal{U}$ through itself, it is enough to give a map from the refined index type to the original index type and, for each refined row $a : A$, an inclusion witness from the refined member to the chosen original member. Take the map to be $r := \mathrm{id}_A$.

Fix $a : A$. Since $\mathrm{id}_A(a) = a$, the required witness

$$\epsilon_a : \iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a))$$

is exactly the pointwise implication

$$\forall x : X,\ \mathsf{OpenAt}(\iota(a), x) \Longrightarrow \mathsf{OpenAt}(\iota(a), x),$$

which holds immediately from the definition of $\leq_{\mathcal{T}}$ in Definition 151.154. The indexed-open membership row OpenAt is part of the topology carrier recorded in Definition 88.1, so this is a valid displayed inclusion witness in the same ledger as every other cover inclusion.

The onward route from the refined member to the ambient open i is then

$$\iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a)) \leq_{\mathcal{T}} i.$$

Its second leg is the original cover inclusion $h_a : \iota(a) \leq_{\mathcal{T}} i$, because $\mathrm{id}_A(a) = a$. Therefore the refinement data are exactly the identity selector together with the reflexive inclusion witnesses, and these satisfy the compatibility clause required in Definition 151.174. Hence $\mathcal{U}$ is a displayed refinement of itself. □

Theorem 151.182 (Sheaf identity-refinement pullback identity)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of $i : \mathsf{OpenIx}$, and let $s_a : \mathsf{Sec}(\iota(a))$ be a local family over $\mathcal{U}$. Use the identity displayed refinement of $\mathcal{U}$ through itself from Definition 151.181, and write its pulled-back family as*

$$t_a := \rho_{\iota(\mathrm{id}_A(a)),\iota(a)}(s_{\mathrm{id}_A(a)}).$$

Then for every $a : A$,

$$t_a \sim_{\iota(a)} s_a.$$

In particular, for every compatible family over $\mathcal{U}$, pullback along the identity refinement is pointwise the original family in the section classifier:

$$\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a.$$

Proof. Fix $a : A$. By Definition 151.181, the identity refinement chooses the selector $\mathrm{id}_A : A \to A$ and the reflexive inclusion witness

$$\iota(a) \leq_{\mathcal{T}} \iota(\mathrm{id}_A(a)).$$

Since $\mathrm{id}_A(a) = a$, this is exactly the reflexive inclusion

$$\iota(a) \leq_{\mathcal{T}} \iota(a).$$

Thus the pulled-back row at a is the identity restriction of the same local section:

$$t_a = \rho_{\iota(a),\iota(a)}(s_a)$$

inside the section carrier indexed by $\iota(a)$.

The identity field of the indexed section presheaf carrier in Definition 151.154 gives, for every section $s : \mathsf{Sec}(\iota(a))$,

$$\rho_{\iota(a),\iota(a)}(s) \sim_{\iota(a)} s.$$

Applying this field to s_a gives

$$\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a.$$

Substituting the displayed description of t_a gives $t_a \sim_{\iota(a)} s_a$. The argument is pointwise in a and does not use any overlap clause, so it applies in particular when the family s_a is compatible over $\mathcal{U}$ in the sense of Definition 151.155. □

Theorem 151.183 (Sheaf refinement composition obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Suppose (r, ϵ) is a displayed refinement of $\mathcal{V}$ through $\mathcal{U}$ and (s, η) is a displayed refinement of $\mathcal{W}$ through $\mathcal{V}$:*

$$r : B \to A, \qquad \epsilon_b : \kappa(b) \leq_{\mathcal{T}} \iota(r(b)),$$

$$s : C \to B, \qquad \eta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c)).$$

Then the composite map $r \circ s : C \to A$ together with the composite inclusions

$$\theta_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(s(c)))$$

is a displayed refinement of $\mathcal{W}$ through $\mathcal{U}$. The cover row for $\mathcal{W}$ remains the pointwise indexed-open implication

$$\forall x : X, \quad \mathsf{OpenAt}(i, x) \Longrightarrow \exists c : C,\ \mathsf{OpenAt}(\lambda(c), x),$$

and the route onward to i is the inclusion chain $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c))) \leq_{\mathcal{T}} i$.

Proof. For each $c : C$, the refinement row (s, η) gives $\eta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$. The refinement row (r, ϵ) gives $\epsilon_{s(c)} : \kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c)))$.

Transitivity of indexed-open inclusion in the topology carrier composes these two rows and gives the displayed inclusion $\theta_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(s(c)))$.

The cover data of $\mathcal{W}$ are not changed. Its row ω still says that every point of the ambient open i lies in some indexed member $\lambda(c)$, expressed by the displayed OpenAt implication. Together with the map $r \circ s$ and the inclusions θ_c, this is exactly the refinement obligation of Definition 151.174 for $\mathcal{W}$ through $\mathcal{U}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement composition`

Theorem 151.184 (Sheaf compatible-family pullback composes over refinements)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of $i : \mathsf{OpenIx}$. Let $(r, \epsilon) : \mathcal{V} \to \mathcal{U}$ and $(s, \eta) : \mathcal{W} \to \mathcal{V}$ be displayed refinements, and let $(r \circ s, \theta) : \mathcal{W} \to \mathcal{U}$ be their composite refinement from Definition 151.183. If $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$, define*

$$v_b := \rho_{\iota(r(b)),\kappa(b)}(u_{r(b)})$$

and

$$w_c := \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then w_c is classifier-equal to the one-step pullback

$$z_c := \rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))})$$

for every $c : C$, and the family w_c is compatible over $\mathcal{W}$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement pullback composition`

Proof. Apply Definition 151.175 to the refinement (r, ϵ). It gives compatibility of the family v_b over $\mathcal{V}$.

Apply the same theorem to the refinement (s, η) and the compatible family v_b. This gives compatibility of the tower-pulled family w_c over $\mathcal{W}$.

It remains to compare the two descriptions of the local section at $c : C$. By definition, w_c restricts $u_{r(s(c))}$ first along $\kappa(s(c)) \leq_{\mathcal{T}} \iota(r(s(c)))$ and then along $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$. Restriction composition for the indexed section presheaf carrier identifies this two-step restriction with the direct restriction along the composite inclusion θ_c.

The classifier on $\mathsf{Sec}(\lambda(c))$ supplies symmetry and transitivity, so the restriction-composition row gives $w_c \sim_{\lambda(c)} z_c$. Since compatibility of w_c was already obtained from the second pullback application, the composite tower pullback is compatible over $\mathcal{W}$ and agrees with the one-step pullback along $(r \circ s, \theta)$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRefinementPullback composition`

Theorem 151.185 (Sheaf gluing descent composes over refinement towers)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$, $\mathcal{V} = (B, \kappa, q, \delta)$, and $\mathcal{W} = (C, \lambda, p, \omega)$ be indexed open covers of $i : \mathsf{OpenIx}$. Let $(r, \epsilon) : \mathcal{V} \to \mathcal{U}$ and $(s, \eta) : \mathcal{W} \to \mathcal{V}$ be displayed refinements, with composite $(r \circ s, \theta) : \mathcal{W} \to \mathcal{U}$. Suppose $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues it:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} u_a.$$

Then g glues the composite pulled-back family over $\mathcal{W}$:

$$\forall c : C, \quad \rho_{i,\lambda(c)}(g) \sim_{\lambda(c)} \rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))}).$$

Proof. Apply Definition 151.176 to (r, ϵ) and the gluing row for g. This shows that g glues the pulled-back family $v_b := \rho_{\iota(r(b)),\kappa(b)}(u_{r(b)})$ over $\mathcal{V}$.

Apply Definition 151.176 again to (s, η) and the displayed gluing of v_b. This shows that g glues the two-step pulled-back family $w_c := \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)})$ over $\mathcal{W}$.

By Definition 151.184, w_c is classifier-equal to the one-step pullback $\rho_{\iota(r(s(c))),\lambda(c)}(u_{r(s(c))})$ for every $c : C$. Transitivity of the classifier on $\mathsf{Sec}(\lambda(c))$ composes this comparison with the two-step gluing row, giving the displayed gluing row for the composite refinement. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger refinement tower gluing`

Definition 151.186 (Sheaf displayed common refinement span)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. A displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$ is a tuple*

$$\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$$

where C is an indexing type, $\lambda : C \to \mathsf{OpenIx}$ is an indexed-open family, $r : C \to A$ and $s : C \to B$ choose the displayed members of the two covers, and for every $c : C$ there are inclusion witnesses

$$\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c)), \qquad \zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c)).$$

The row ω says that λ itself is an indexed open cover of i, with member inclusions obtained through either composite $\lambda(c) \leq_{\mathcal{T}} \iota(r(c)) \leq_{\mathcal{T}} i$ or $\lambda(c) \leq_{\mathcal{T}} \kappa(s(c)) \leq_{\mathcal{T}} i$, and with the pointwise cover law

$$\forall x : X,\ \mathsf{OpenAt}(i, x) \Longrightarrow \exists c : C,\ \mathsf{OpenAt}(\lambda(c), x).$$

Thus the common refinement is displayed entirely inside indexed opens, inclusion rows, and presheaf restriction rows; it contains no host-level equality of covers.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan base change gluing package`

Theorem 151.187 (Sheaf displayed common refinement gives paired refinements)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$. If $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ is a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$ in the sense of Definition 151.186, then $\mathcal{W}$ is a displayed refinement of its cover through $\mathcal{U}$ and also a displayed refinement of its cover through $\mathcal{V}$:*

$$(r, \epsilon) : \mathcal{W} \to \mathcal{U}, \qquad (s, \zeta) : \mathcal{W} \to \mathcal{V}.$$

Both refinements are in the sense of Definition 151.174.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan paired refinements`

Proof. Unpack the displayed span. Its row ω makes $\lambda : C \to \mathsf{OpenIx}$ an indexed cover of i, so $\mathcal{W}$ has the cover side required to be the refined cover in Definition 151.174.

The first projection consists of the map $r : C \to A$ together with the inclusions $\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c))$. These are exactly the displayed refinement rows from $\mathcal{W}$ through $\mathcal{U}$, with the route onward to i supplied by the cover inclusion $h_{r(c)}$.

The second projection is identical with $\mathcal{V}$ in place of $\mathcal{U}$: the map $s : C \to B$ and the inclusions $\zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$ form the displayed refinement rows, with the cover inclusion $q_{s(c)}$ supplying the route to i. □

Theorem 151.188 (Sheaf common refinement gluing invariance)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of $i : \mathsf{OpenIx}$, and let $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ be a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$. Let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$ and let $v_b : \mathsf{Sec}(\kappa(b))$ be compatible over $\mathcal{V}$. Suppose their pullbacks to $\mathcal{W}$ are classifier-equal:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

If $g : \mathsf{Sec}(i)$ glues u_a over $\mathcal{U}$ and $h' : \mathsf{Sec}(i)$ glues v_b over $\mathcal{V}$, then

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCommonRefinementGluing carrier invariance`

Proof. Project the common refinement span by Definition 151.187. This gives the displayed refinement (r, ϵ) from $\mathcal{W}$ through $\mathcal{U}$ and the displayed refinement (s, ζ) from $\mathcal{W}$ through $\mathcal{V}$.

Apply Definition 151.176 to the first refinement and the gluing row for g. The result says that g glues the pulled-back family

$$c \mapsto \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)})$$

over $\mathcal{W}$.

Apply the same descent theorem to the second refinement and the gluing row for h'. This says that h' glues the pulled-back family

$$c \mapsto \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)})$$

over $\mathcal{W}$. The displayed classifier equality between the two pulled-back families lets the compatible-family stability row Definition 151.173 replace the second family by the first one without changing the gluing comparison.

Now both g and h' are displayed gluings of the same compatible family over the indexed cover $\mathcal{W}$. Apply Definition 151.169 over $\mathcal{W}$ to obtain $g \sim_i h'$. □

Theorem 151.189 (Sheaf cover presentation exactness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Let $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ be a displayed common refinement span from $\mathcal{U}$ to $\mathcal{V}$. Suppose a compatible family $u_a : \mathsf{Sec}(\iota(a))$ over $\mathcal{U}$ and a compatible family $v_b : \mathsf{Sec}(\kappa(b))$ over $\mathcal{V}$ agree after pullback to $\mathcal{W}$:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then any global section class determined by gluing u_a over $\mathcal{U}$ is the same classifier class as any global section class determined by gluing v_b over $\mathcal{V}$. Equivalently, if $g, h' : \mathsf{Sec}(i)$ are displayed gluings of the two families, then $g \sim_i h'$. The exactness row uses only indexed cover presentations, displayed common refinement, restriction transport, and the classifiers $\sim_j$; it does not compare host powersets and does not form a quotient of cover presentations.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentation exactness`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentation common refinement global comparison`

Proof. Choose displayed gluings $g : \mathsf{Sec}(i)$ and $h' : \mathsf{Sec}(i)$ for the two compatible families. The hypotheses give exactly the common-refinement equality of the two pulled-back local families.

Apply Definition 151.188 to $\mathcal{U}$, $\mathcal{V}$, the displayed common refinement $\mathcal{W}$, the two compatible families, their pulled-back equality, and the two displayed gluing rows. It returns $g \sim_i h'$.

Since the argument never leaves the indexed cover ledger, the conclusion identifies the two global classifier classes without choosing a preferred cover presentation, comparing powersets of host opens, or adding any quotient primitive. □

Theorem 151.190 (Sheaf NameCert obligation surface)**.** *The concrete* $\mathsf{Sheaf}^{\uparrow}$ *certificate surface is the following ten-field obligation surface.*

1. *The traditional target is a sheaf of sections over indexed opens of an indexed topology carrier.*
2. *The BEDC source data are* $\mathsf{Topology}^{\uparrow}$ *together with* $\mathsf{PreSheaf}^{\uparrow}$*.*
3. *The carrier is the indexed section presheaf carrier of Definition 151.154.*
4. *The classifier family is the open-indexed relation $\sim_i$ on $\mathsf{Sec}(i)$.*
5. *The stability obligations are classifier equivalence, identity restriction, composition of restriction, restriction classifier transport, locality, gluing, uniqueness of gluing, compatibility of global restrictions, compatible-family stability, cover-refinement transport, refined-cover exactness, refinement tower composition, displayed common-refinement comparison, and cover-presentation exactness.*
6. *The ledger consists of open-cover indices, cover-member inclusions, finite-intersection overlap indices, and restriction rows to those overlaps.*
7. *The core theorem list is Definition 151.172, Definition 151.173, Definition 151.174, Definition 151.175, Definition 151.176, Definition 151.177, Definition 151.178, Definition 151.179, Definition 151.180, Definition 151.183, Definition 151.184, Definition 151.185, Definition 151.187, Definition 151.188, Definition 151.189, Definition 151.169, Definition 151.170, and Definition 151.171.*
8. *Exactness is locality together with existence and uniqueness of gluing for compatible local families, stable under displayed common refinement of indexed cover presentations.*

9. *There is no standard bridge attached to this surface.*
10. *The status is a paper-side obligation surface for downstream certificates using* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafNameCert obligation surface`

Proof. Rows one through four are the data fixed by Definition 151.154 and Definition 151.155: indexed opens come from $\mathsf{Topology}^{\uparrow}$, section carriers and contravariant restriction rows come from $\mathsf{PreSheaf}^{\uparrow}$, and classifier equality is indexed by the same opens as the section carriers.

Row five is exactly the list of fields and consequences exposed in this section. Equivalence, identity, composition, and restriction transport are part of the carrier surface; locality and gluing are the sheaf fields; uniqueness, compatible global restrictions, idempotence of global gluing, compatible-family stability, cover-refinement transport, refined-cover exactness, refinement tower composition, common-refinement comparison, and cover-presentation exactness are the theorem rows cited in row seven.

Rows six through ten record how downstream constructions consume the surface. A ringed-space or scheme certificate can name cover indices, overlaps, refinement maps, common-refinement spans, and restriction rows without choosing a standard-model bridge; the exactness content available to it is precisely local detection plus existence and uniqueness of gluing, stable under passing to displayed refined covers, refinement towers, and comparison of two cover presentations through a displayed common refinement. □

Corollary 151.191 (Sheaf carried-section pullback associativity along composite refinement)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$*. Let* $\mathcal{U} = (A, \alpha, h, \mu)$*,* $\mathcal{V} = (B, \beta, q, \nu)$*, and* $\mathcal{W} = (C, \gamma, p, \omega)$ *be indexed open covers of* $i : \mathsf{OpenIx}$*. Suppose* $(r, \epsilon) : \mathcal{U} \to \mathcal{V}$ *and* $(s, \eta) : \mathcal{V} \to \mathcal{W}$ *are displayed refinements in the sense of Definition 151.174:*

$$r : A \to B, \qquad \epsilon_a : \alpha(a) \leq_{\mathcal{T}} \beta(r(a)),$$

$$s : B \to C, \qquad \eta_b : \beta(b) \leq_{\mathcal{T}} \gamma(s(b)).$$

Let $(s \circ r, \theta) : \mathcal{U} \to \mathcal{W}$ *be their composite displayed refinement from Definition 151.183, where* $\theta_a : \alpha(a) \leq_{\mathcal{T}} \gamma(s(r(a)))$ *is the composite inclusion. If* $\sigma_c : \mathsf{Sec}(\gamma(c))$ *is a compatible carried local family over* $\mathcal{W}$*, define the two-stage pullback by*

$$\tau_b := \rho_{\gamma(s(b)),\beta(b)}(\sigma_{s(b)}), \qquad \upsilon_a := \rho_{\beta(r(a)),\alpha(a)}(\tau_{r(a)}),$$

and define the direct composite pullback by

$$\zeta_a := \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

Then the section classifier on the refined member classifies the two pullbacks together:

$$\forall a : A, \qquad \upsilon_a \sim_{\alpha(a)} \zeta_a.$$

Moreover, the two-stage family υ_a *is compatible over* $\mathcal{U}$*.*

Proof. The composite refinement is obtained by composing the displayed inclusion rows. For each $a : A$, the refinement (r, ϵ) supplies $\epsilon_a : \alpha(a) \leq_{\mathcal{T}} \beta(r(a))$, and the refinement (s, η) supplies $\eta_{r(a)} : \beta(r(a)) \leq_{\mathcal{T}} \gamma(s(r(a)))$. Their transitive composite is the inclusion $\theta_a : \alpha(a) \leq_{\mathcal{T}} \gamma(s(r(a)))$ used by the one-step pullback along $s \circ r$.

Since σ_c is compatible over $\mathcal{W}$, Definition 151.175 applied to (s, η) gives compatibility of the pulled-back family τ_b over $\mathcal{V}$. Applying the same theorem to (r, ϵ) and the compatible family τ_b gives compatibility of υ_a over $\mathcal{U}$.

It remains to identify the two descriptions of the member at a. Expanding the definitions gives

$$\upsilon_a = \rho_{\beta(r(a)),\alpha(a)}\big(\rho_{\gamma(s(r(a))),\beta(r(a))}(\sigma_{s(r(a))})\big),$$

while the direct composite pullback is

$$\zeta_a = \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

These are the two restriction routes along the same chain

$$\alpha(a) \leq_{\mathcal{T}} \beta(r(a)) \leq_{\mathcal{T}} \gamma(s(r(a))).$$

The composition row of the indexed section presheaf carrier in Definition 151.154 compares the iterated restriction with the direct restriction in the target classifier $\sim_{\alpha(a)}$:

$$\rho_{\beta(r(a)),\alpha(a)}\big(\rho_{\gamma(s(r(a))),\beta(r(a))}(\sigma_{s(r(a))})\big) \sim_{\alpha(a)} \rho_{\gamma(s(r(a))),\alpha(a)}(\sigma_{s(r(a))}).$$

Substituting the definitions of υ_a and ζ_a gives $\upsilon_a \sim_{\alpha(a)} \zeta_a$ for every $a : A$. Thus pullback along the refinement tower and pullback along the composite refinement determine the same carried section class on each refined member. □

Definition 151.192 (Sheaf restricted common-refinement ledger)**.** *Let $\mathcal{T}$ be a* BHist*-scoped indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$*, let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$, and fix an open index i :* OpenIx*. Work in the restricted-open carrier $\mathcal{S}|_i$ of Definition 151.236. For a restricted target (j, η_j), let*

$$\mathcal{U}_{|i} = (A, \iota, \eta^{\iota}, h, \gamma), \qquad \mathcal{V}_{|i} = (B, \kappa, \eta^{\kappa}, q, \delta)$$

be two indexed open covers of (j, η_j), where each $\eta^{\iota}_a : \iota(a) \leq_{\mathcal{T}} i$ and $\eta^{\kappa}_b : \kappa(b) \leq_{\mathcal{T}} i$ records that the member lies below the fixed open i. A restricted common-refinement ledger from $\mathcal{U}_{|i}$ to $\mathcal{V}_{|i}$ is a tuple

$$\mathcal{W}_{|i} = (C, \lambda, \eta^{\lambda}, r, s, \epsilon, \zeta, \omega)$$

such that C is an indexing type, $\lambda : C \to$ OpenIx *is an indexed-open family, every $\eta^{\lambda}_c : \lambda(c) \leq_{\mathcal{T}} i$ lies below the fixed open, $r : C \to A$ and $s : C \to B$ choose members of the two covers, and*

$$\epsilon_c : \lambda(c) \leq_{\mathcal{T}} \iota(r(c)), \qquad \zeta_c : \lambda(c) \leq_{\mathcal{T}} \kappa(s(c))$$

for every $c : C$. The row ω says that λ is an indexed open cover of j, with member inclusions obtained by composing through either displayed leg and with pointwise cover law over $\mathsf{OpenAt}(j, x)$*.*

Thus the ledger is a displayed common-refinement span entirely inside the slice below i: all carrier rows are restricted indices, all maps are inherited restrictions of $\mathcal{S}$, and all comparisons are made by the section classifiers of Definition 151.236.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinementLedger`

Theorem 151.193 (Sheaf restricted common-refinement pullback compatibility)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let $\mathcal{U}_{|i}$ and $\mathcal{V}_{|i}$ be restricted covers of a restricted target (j, η_j), and let $\mathcal{W}_{|i}$ be a restricted common-refinement ledger between them in the sense of Definition 151.192. If*

$$u_a : \mathsf{Sec}_{|i}(\iota(a), \eta_a^\iota)$$

is compatible over $\mathcal{U}_{|i}$ and

$$v_b : \mathsf{Sec}_{|i}(\kappa(b), \eta_b^\kappa)$$

is compatible over $\mathcal{V}_{|i}$, then their pullbacks to the restricted common refinement are compatible families over $\mathcal{W}_{|i}$. Moreover, if the displayed comparison

$$\forall c : C, \quad \rho^{|i}_{(\iota(r(c)),\eta^\iota_{r(c)}),(\lambda(c),\eta^\lambda_c)}(u_{r(c)}) \sim_{|i,(\lambda(c),\eta^\lambda_c)} \rho^{|i}_{(\kappa(s(c)),\eta^\kappa_{s(c)}),(\lambda(c),\eta^\lambda_c)}(v_{s(c)})$$

holds on $\mathcal{W}_{|i}$, then it is the same comparison as the ambient common-refinement comparison after forgetting only the witnesses to i.

Proof. Unfold the restricted carrier using Definition 151.236. The restricted section carriers and classifiers at $\iota(a)$, $\kappa(b)$, and $\lambda(c)$ are exactly the ambient carriers and classifiers at those open indices, and the restricted restriction maps are the ambient maps ρ.

The restricted ledger of Definition 151.192 forgets to a displayed common-refinement span in the ambient sheaf. By Definition 151.187, its two projections are displayed refinements through $\mathcal{U}$ and $\mathcal{V}$.

Apply the pullback part of the ambient refinement exactness surface to each projection. Equivalently, read Definition 151.184 with the identity tower on the displayed span: the pulled-back families are compatible over the common refinement, and the two-step and one-step restriction descriptions agree by classifier equality.

Finally, Definition 151.237 identifies the restricted restriction composition and classifier transport rows with the ambient ones. Therefore the displayed equality in the restricted carrier is exactly the ambient equality on the same local sections, with the witnesses η^λ_c recording only that the comparison remains below i. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinement pullback compatibility`

Theorem 151.194 (Sheaf restricted common-refinement gluing invariance)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let $\mathcal{U}_{|i}$ and $\mathcal{V}_{|i}$ be restricted covers of a restricted target (j, η_j), and let $\mathcal{W}_{|i}$ be a restricted common-refinement ledger between them. Let u_a and v_b be compatible local families over those two restricted covers. Suppose their pullbacks to $\mathcal{W}_{|i}$ agree:*

$$\forall c : C, \quad \rho^{|i}_{(\iota(r(c)),\eta^\iota_{r(c)}),(\lambda(c),\eta^\lambda_c)}(u_{r(c)}) \sim_{|i,(\lambda(c),\eta^\lambda_c)} \rho^{|i}_{(\kappa(s(c)),\eta^\kappa_{s(c)}),(\lambda(c),\eta^\lambda_c)}(v_{s(c)}).$$

If $g : \mathsf{Sec}_{|i}(j, \eta_j)$ glues u_a over $\mathcal{U}_{|i}$ and $h' : \mathsf{Sec}_{|i}(j, \eta_j)$ glues v_b over $\mathcal{V}_{|i}$, then

$$g \sim_{|i,(j,\eta_j)} h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedCommonRefinement gluing invariance`

Proof. First forget the target witnesses to i. By Definition 151.236, the two restricted covers become ambient indexed covers of j, the local families are the same ambient section families, and the displayed gluings are ambient gluing rows.

The common-refinement ledger becomes the ambient displayed span required by Definition 151.187. The agreement hypothesis is the ambient pullback comparison, because the restricted classifier at $(\lambda(c), \eta_c^\lambda)$ is $\sim_{\lambda(c)}$.

Apply Definition 151.188 to the two ambient covers, the ambient span, the two compatible families, their pullback comparison, and the two gluing rows. This gives $g \sim_j h'$ in the ambient sheaf.

Return to the slice below i. The classifier at the restricted target (j, η_j) is $\sim_j$, so the ambient conclusion is exactly the displayed restricted classifier equality. The locality and gluing rows used in this passage are the restricted rows of Definition 151.238 read through the carrier identification. □

Theorem 151.195 (Sheaf restricted-open refinement exactness export)**.** *For every fixed open index i : OpenIx, the restricted-open carrier $\mathcal{S}|_i$ exports the local cover-presentation exactness surface consumed by $\mathsf{RingedSpace}^\uparrow$ and $\mathsf{Scheme}^\uparrow$. Its rows are:*

1. *restricted carrier, classifier, and restriction laws from Definition 151.236 and Definition 151.237;*
2. *restricted locality and gluing for covers below i from Definition 151.238;*
3. *restricted common-refinement ledgers below i from Definition 151.192;*
4. *pullback compatibility over a restricted common refinement from Definition 151.193;*
5. *classifier equality of restricted gluings whose pulled-back families agree from Definition 151.194.*

The exported surface is local in the fixed open i, uses no host powerset equality of covers, and introduces no quotient comparison between cover presentations.

Proof. The first two rows are already established by the restricted-open surface. Apply Definition 151.236 to identify the carriers, classifiers, and inherited restriction maps, then apply Definition 151.237 and Definition 151.238 for the restriction, locality, and gluing fields.

The third row is the ledger of Definition 151.192. It is exactly the ambient displayed common-refinement span of Definition 151.187, equipped with witnesses that every member remains below the fixed open i.

For compatibility, use Definition 151.193. The proof of that theorem reads the ambient refinement pullback composition row Definition 151.184 through the restricted carrier, so the local consumers see compatible pulled-back families without leaving the slice.

For presentation independence, use Definition 151.194. It is the restricted form of Definition 151.188: two displayed gluings of compatible families with equal pullbacks to the restricted common refinement become classifier-equal at (j, η_j).

Combining these rows gives the local exactness surface required by ringed-space and scheme certificates. Every row is expressed in indexed opens, restricted inclusion witnesses, inherited restrictions, and section classifiers, so the export remains inside the BEDC sheaf certificate surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenRefinement exactness export`

151.10 Downstream consumption boundary

Definition 151.196 (Sheaf downstream consumption boundary)**.** *The downstream consumption boundary exported by $\mathsf{Sheaf}^\uparrow$ to $\mathsf{RingedSpace}^\uparrow$ and $\mathsf{Scheme}^\uparrow$ is the following ten-field boundary.*

1. *The traditional target is an open-indexed sheaf of sections over an indexed open topology carrier.*

2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{NameCert}$.
3. *The carrier row is the indexed section presheaf carrier of Definition 151.154.*
4. *The classifier row is the section classifier family* $\sim_i$ *on* $\mathsf{Sec}(i)$ *from the same carrier.*
5. *The locality and gluing rows are the indexed-cover compatibility, locality, and gluing fields of Definition 151.155.*
6. *The cover ledger is the indexed cover row: cover indices, cover-member maps, inclusion witnesses, pointwise cover membership, and finite-intersection overlap rows.*
7. *The refinement ledger is the displayed refinement, refinement-tower, and common-refinement surface used by Definition 151.189.*
8. *The restricted-open ledger is the slice carrier and its inherited restriction, locality, and gluing rows from Definition 151.239.*
9. *The base-change ledger is the displayed base-change square and its cover-pullback, gluing-pullback, and common-refinement rows from Definition 151.208.*
10. *The exactness row is cover presentation exactness together with restricted common-refinement invariance, and no standard bridge is attached to this boundary.*

This boundary is the only sheaf-side interface available to downstream chapters: it names indexed sections, section classifiers, displayed cover rows, displayed refinements, restricted opens, and displayed base change, but it does not import a host-space model of sheaves.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDownstreamConsumptionBoundary`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafUp StdBridge`

Theorem 151.197 (Ringed-space consumption obligation)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *certificate consumes exactly the* $\mathsf{Sheaf}^{\uparrow}$ *carrier, section classifier, restriction stability, cover-gluing exactness, restricted-open surface, and base-change surface listed in Definition 151.196, together with the* $\mathsf{CommRing}^{\uparrow}$ *rows asserting that the section carriers and stalk rows carry commutative-ring structure. On the sheaf side, every row required by* $\mathsf{RingedSpace}^{\uparrow}$ *is one of:*

1. *the indexed section carriers* $\mathsf{Sec}(i)$ *and classifiers* $\sim_i$*;*
2. *restriction identity, restriction composition, and restriction classifier transport;*
3. *locality, gluing, and cover-presentation exactness for indexed covers;*
4. *the restricted-open surface below a named open;*
5. *the displayed base-change surface for inverse-image and cover-pullback rows.*

No further sheaf row is needed to state the paper-side ringed-space certificate boundary.

Proof. The carrier and classifier rows are the first projections of the $\mathsf{Sheaf}^{\uparrow}$ boundary. They are the section carriers and section classifiers of Definition 151.154, while restriction identity, restriction composition, and classifier transport are the stability rows already exported in Definition 151.168.

For local ringed-space data, restrict the sheaf to each displayed open. The required open-local carrier, classifier, restriction, locality, gluing, and no-bridge rows are exactly the restricted-open surface of Definition 151.239.

For inverse-image or change-of-base comparisons, use the displayed base-change surface of Definition 151.208. The remaining rows mentioned in the statement are the commutative-ring and stalk rows supplied on the $\mathsf{CommRing}^{\uparrow}$ side; they do not ask $\mathsf{Sheaf}^{\uparrow}$ for any host powerset equality, cover quotient, external sheaf model, or standard bridge. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRingedSpaceConsumption obligation`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer chart endpoint classifier`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer RingedSpace boundary rows`

Theorem 151.198 (Scheme consumption obligation)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *certificate consumes the same* $\mathsf{Sheaf}^{\uparrow}$ *downstream boundary only through a cover by affine or ringed-space charts and the* $\mathsf{CommRing}^{\uparrow}$ *spectra rows attached to those charts. Concretely, each chart reads the sheaf rows through Definition 151.197; compatibility across chart overlaps uses the cover-presentation exactness and common-refinement gluing invariance rows of* $\mathsf{Sheaf}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeConsumption obligation`

Proof. On each affine or ringed-space chart, apply Definition 151.197. This identifies the sheaf-side data available to the chart as the carrier, classifier, restriction stability, cover-gluing exactness, restricted-open surface, and base-change surface in Definition 151.196.

On overlaps between charts, compare the two displayed cover presentations through a displayed common refinement. The equality of glued section classes is supplied by Definition 151.189, and the comparison of two gluings after passing to a common refinement is supplied by Definition 151.188.

The affine and spectra content is therefore external to the sheaf boundary: it comes from the ringed-space chart and $\mathsf{CommRing}^{\uparrow}$ spectra rows. The sheaf chapter contributes only the displayed indexed-cover, restricted-open, and base-change rows already enumerated in the boundary. □

Theorem 151.199 (Sheaf downstream consumption exhaustion)**.** *Every* $\mathsf{Sheaf}^{\uparrow}$ *row used by* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *in the two downstream consumption obligations is one of the rows in Definition 151.196. Consequently, the downstream chapters import no host powerset equality of opens, no quotient of cover presentations, no external sheaf model, and no bridge closure from* $\mathsf{Sheaf}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumption joint exhaustion`

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, inspect Definition 151.197. Its sheaf-side rows are precisely the carrier, classifier, restriction-stability, cover-gluing, restricted-open, and base-change entries of Definition 151.196. The commutative-ring and stalk rows named there are not sheaf rows.

For $\mathsf{Scheme}^{\uparrow}$, inspect Definition 151.198. Each chart first passes through the ringed-space consumption obligation, and every overlap comparison is discharged by cover-presentation exactness or common-refinement gluing invariance, both already included in the boundary's refinement and exactness rows.

Thus the union of sheaf rows consumed by the two downstream chapters is exhausted by the ten-field boundary. Since the boundary itself excludes host powerset equality, cover quotients, external sheaf models, and standard bridges, none of those structures is imported downstream through $\mathsf{Sheaf}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumption boundary joint exhaustion`

Definition 151.200 (Sheaf BHist triple-overlap route ledger)**.** *Let* $\mathcal{T}$ *be an indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$ *on* BHist *with membership predicate* OpenAt*, history classifier* hsame*, and finite-intersection operation* $\wedge_{\mathsf{Topology}^{\uparrow}}$*. Let* $\mathcal{S}$ *be an indexed sheaf carrier over* $\mathcal{T}$*, and let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an indexed open cover of* $i : \mathsf{OpenIx}$*. For* $a, b, c : A$*, write*

$$k_{a,b} := \iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b), \qquad \ell_{a,b,c} := k_{a,b} \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c).$$

The BHist *triple-overlap route ledger of* $\mathcal{U}$ *is the subledger of Definition 151.160 whose rows are the member rows for* $\iota(a)$*,* $\iota(b)$*, and* $\iota(c)$*, the pair-overlap row* $k_{a,b}$*, the triple-overlap row* $\ell_{a,b,c}$*, the displayed projections from* $\ell_{a,b,c}$ *to each participating member through the available finite-intersection routes, the induced restriction maps into* $\mathsf{Sec}(\ell_{a,b,c})$*, and any optional* Cont *histories that certify equality of displayed route witnesses. Membership transport along the row is only the visible* hsame *transport on* BHist *cells:*

$$\mathsf{OpenAt}(\ell_{a,b,c}, x) \wedge_{\mathsf{Topology}^{\uparrow}} x\mathsf{hsame}y \Longrightarrow \mathsf{OpenAt}(\ell_{a,b,c}, y).$$

The ledger records route data, restriction targets, section classifiers, and BHist *provenance. It does not add a quotient of cover presentations or a host-level equality between overlap routes.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRouteLedger`

Theorem 151.201 (Sheaf BHist triple-overlap route coherence)**.** *Let* $\mathcal{T}$*,* $\mathcal{S}$*, and* $\mathcal{U}$ *be as in Definition 151.200. For every* $a, b, c : A$*, every section* $s : \mathsf{Sec}(\iota(a))$*, and every two displayed routes from* $\iota(a)$ *to* $\ell_{a,b,c}$ *in the triple-overlap route ledger, the two induced restrictions have the same classifier at the triple-overlap row:*

$$\rho_{k_{a,b},\ell_{a,b,c}}(\rho_{\iota(a),k_{a,b}}(s)) \sim_{\ell_{a,b,c}} \rho_{\iota(a),\ell_{a,b,c}}(s).$$

The same conclusion holds for the routes beginning at $\iota(b)$ *and* $\iota(c)$ *whenever the corresponding projection route is displayed. Optional* Cont *histories compare only the route witnesses; they do not change the target carrier* $\mathsf{Sec}(\ell_{a,b,c})$ *or the classifier* $\sim_{\ell_{a,b,c}}$*.*

Proof. Unfold the triple-overlap route ledger. Its routes are the pair-overlap and triple-overlap projection rows already present in the cover nerve ledger of Definition 151.160, with projections read by Definition 151.161.

Apply Definition 151.304 to the displayed chain $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b} \leq_{\mathcal{T}} \iota(a)$. The two-step restriction and the direct displayed restriction land in the same carrier $\mathsf{Sec}(\ell_{a,b,c})$ and are compared by $\sim_{\ell_{a,b,c}}$.

The arguments for $\iota(b)$ and $\iota(c)$ are the same after choosing the displayed finite-intersection projection route from the ledger. If a Cont history is attached, the cited route-stability theorem treats it as route provenance, so the classifier comparison at $\ell_{a,b,c}$ is unchanged. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger triple overlap route coherence`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRoute coherence`

Theorem 151.202 (Sheaf BHist triple-overlap compatible-family descent)**.** *Let* $\mathcal{T}$*,* $\mathcal{S}$*, and* $\mathcal{U}$ *be as in Definition 151.200. If* $s_a : \mathsf{Sec}(\iota(a))$ *is compatible over* $\mathcal{U}$*, then for every* $a, b, c : A$ *the three restrictions of the local family to* $\ell_{a,b,c}$ *lie in one classifier class:*

$$\rho_{\iota(a),\ell_{a,b,c}}(s_a) \sim_{\ell_{a,b,c}} \rho_{\iota(b),\ell_{a,b,c}}(s_b), \qquad \rho_{\iota(a),\ell_{a,b,c}}(s_a) \sim_{\ell_{a,b,c}} \rho_{\iota(c),\ell_{a,b,c}}(s_c).$$

Thus every compatible family descends from pair-overlap compatibility to a coherent triple-overlap row inside the BHist *cover nerve.*

Proof. Start with pair-overlap compatibility from Definition 151.161. It gives the classifier equality between the restrictions of s_a and s_b on $k_{a,b}$.

Restrict that equality along $\ell_{a,b,c} \leq_{\mathcal{T}} k_{a,b}$. By Definition 151.304, the transported equality is the same as the direct comparison of the two route restrictions to $\ell_{a,b,c}$. The route-coherence theorem Definition 151.201 identifies the composed and direct displays.

For the comparison with s_c, use the pair-overlap rows for (a, c) and (b, c) and pull them to the displayed triple-overlap row. The finite-intersection projections and route-history clauses are those of Definition 151.200, so all comparisons land in $\sim_{\ell_{a,b,c}}$ on the single carrier $\mathsf{Sec}(\ell_{a,b,c})$. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger cover descent exhaustion`

Theorem 151.203 (Sheaf BHist triple-overlap downstream exactness)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback constructions may cite the triple-overlap route ledger as a closed three-face overlap package. For every accepted* BHist *cover row, the package consists of:*

1. *the triple-overlap index* $\ell_{a,b,c} = (\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(c)$ *and its* OpenAt *membership rows, transported only by* hsame*;*
2. *the one-step and two-step restriction routes into* $\mathsf{Sec}(\ell_{a,b,c})$*, with optional* Cont *histories;*
3. *the classifier coherence rows of Definition 151.201 and Definition 151.202;*
4. *the restricted-open exactness rows consumed below a fixed open.*

Consequently a consumer below a fixed open i *can pull the triple-overlap route package through restricted common refinements without reconstructing pairwise compatibility from prose and without adding a host cover quotient.*

Proof. The first three rows are read directly from Definition 151.200, Definition 151.201, and Definition 151.202. They are all internal to BHist cells, indexed opens, restriction maps, and section classifiers.

For passage through a refinement, apply Definition 151.184. It identifies the one-step and two-step descriptions of pulled-back local sections, so the triple-overlap route comparisons survive along displayed refinement towers.

For consumers working below a fixed open, apply Definition 151.195. Its restricted common-refinement surface is the same ambient carrier, restriction, classifier, and gluing data with the additional witness that every index remains below the fixed open.

Combining these with Definition 151.304 gives the advertised downstream package. The conclusion is a reusable exactness row for three-face overlap routes; it introduces no new carrier beyond the existing cover nerve ledger and no comparison other than hsame on histories and $\sim_{\ell_{a,b,c}}$ on sections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistTripleOverlapRoute downstream exactness`

151.11 Displayed base change surface

Definition 151.204 (Sheaf displayed base-change square)**.** *Let* $\mathcal{T}$ *and* $\mathcal{T}'$ *be indexed open topology carriers, and let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over* $\mathcal{T}$*. A displayed base-change square from* $\mathcal{T}'$ *to* $\mathcal{T}$ *consists of:*

1. *a map of indexed open carriers* $f : \mathsf{OpenIx}' \to \mathsf{OpenIx}$ *preserving finite intersections, cover ledgers, and inclusion rows;*
2. *a target open* $i : \mathsf{OpenIx}$ *and a source open* $i' : \mathsf{OpenIx}'$*;*
3. *a displayed inverse-image row saying that* $f^{-1}(i)$ *is represented by* i' *in the source carrier;*
4. *for every indexed cover* $\mathcal{U} = (A, \iota, h, \gamma)$ *of* i*, a displayed pulled-back cover* $\mathcal{U}' = (A, \iota', h', \gamma')$ *of* i' *with the same index type* A*;*
5. *for every* $a : A$*, a carrier row identifying* $f(\iota'(a))$ *with* $\iota(a)$ *below* i*, together with the induced restriction path from* i' *to* $\iota'(a)$*.*

The square is displayed: it records the index map, the represented inverse image, and the memberwise cover pullback without replacing the indexed-open carrier by a powerset model or adding a quotient of covers.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDisplayedBaseChangeSquare`

Theorem 151.205 (Sheaf base-change cover-pullback compatibility)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over $\mathcal{T}$, and let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square as in Definition 151.204. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i, let $\mathcal{U}' = (A, \iota', h', \gamma')$ be its displayed pulled-back cover of i', and let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$. Then the pulled-back local family*

$$u'_a := \rho_{\iota(a), f(\iota'(a))}(u_a)$$

is compatible over the displayed cover $\mathcal{U}'$ after the carrier identification $f(\iota'(a)) \cong \iota(a)$ supplied by the square.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change cover pullback`

Proof. Unfold compatibility over $\mathcal{U}'$. For two members $a, b : A$, the source overlap is represented by the finite intersection of $\iota'(a)$ and $\iota'(b)$ in $\mathcal{T}'$.

The base-change square preserves finite intersections, so applying f sends this source overlap to the target overlap of $\iota(a)$ and $\iota(b)$ below i. The displayed carrier rows identify the two restriction paths from $\iota(a)$ and $\iota(b)$ to that target overlap with the images of the corresponding source paths.

Use the compatibility row for u_a and u_b on the target overlap. Restriction composition in the indexed sheaf carrier transports this equality along the displayed source paths, and restriction classifier transport replaces each transported section by the displayed pullback u'_a and u'_b. This is exactly the required compatibility row over $\mathcal{U}'$. □

Theorem 151.206 (Sheaf base-change gluing pullback)**.** *Let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square from Definition 151.204. Let $\mathcal{U}$ be an indexed cover of i and let $\mathcal{U}'$ be its displayed pulled-back cover of i'. Suppose $u_a : \mathsf{Sec}(\iota(a))$ is compatible over $\mathcal{U}$ and $g : \mathsf{Sec}(i)$ glues u_a over $\mathcal{U}$:*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} u_a.$$

Then the displayed pullback $g' := \rho_{i,f(i')}(g)$ glues the pulled-back family u'_a over $\mathcal{U}'$ after transporting along the represented inverse-image row $f^{-1}(i) \simeq i'$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger base change gluing pullback`

Proof. Fix $a : A$. The gluing row for g restricts g from i to $\iota(a)$ and compares it with u_a.

Transport this row through the displayed base-change square. The inverse-image row represents the source target by i', and the member row represents the pulled-back member by $\iota'(a)$. Restriction composition identifies the path from g' to $\iota'(a)$ with the target path from g through $\iota(a)$ followed by the displayed carrier transport.

Apply restriction classifier transport to the original gluing equality along that path. The right-hand side is precisely the displayed pulled-back section u'_a, and the left-hand side is the restriction of g' to the pulled-back member. Since a was arbitrary, these rows give the displayed gluing of the pulled-back family over $\mathcal{U}'$. □

Theorem 151.207 (Sheaf base-change common-refinement invariance)**.** *Let $f : \mathcal{T}' \to \mathcal{T}$ be a displayed base-change square from Definition 151.204. Let $\mathcal{U}$ and $\mathcal{V}$ be indexed cover presentations of the same target open i, with compatible families u_a and v_b. Let $\mathcal{W}$ be a displayed common refinement on which the two presentations agree:*

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

Then the displayed base-changed cover presentations over i' agree after the pulled-back displayed common refinement. Consequently their glued global classifier classes over the represented source open are the same whenever the corresponding displayed gluings exist.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange common refinement projection`

Proof. Apply Definition 151.205 to $\mathcal{U}$ and to $\mathcal{V}$. This gives compatible pulled-back families over the two displayed source covers.

Pull back the common refinement ledger $\mathcal{W}$ along the same square. Finite-intersection preservation and the memberwise carrier rows give a displayed common refinement between the two source cover presentations, with the same refinement maps r and s on indices.

Transport the target equality on $\mathcal{W}$ along this pulled-back common refinement. Restriction composition and classifier transport identify the transported rows with the two source pulled-back families restricted to the pulled-back refinement members.

Now apply Definition 151.189 to the two source cover presentations and their pulled-back common refinement. It returns equality of the global classifier classes determined by the two displayed gluings over i'. □

Theorem 151.208 (Sheaf base-change NameCert obligation surface)**.** *For every displayed base-change square of indexed open carriers, the $\mathsf{Sheaf}^{\uparrow}$ base-change surface supplies the following $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation rows for downstream ringed-space and scheme consumers.*

1. *carrier: the restricted-open carrier of Definition 151.236, read at the represented source open i';*
2. *classifier: the inherited sheaf classifier $\sim_j$ on every displayed source open below i';*
3. *stability: restriction identity, composition, classifier transport, and restricted-open stability from Definition 151.239, together with cover-pullback compatibility from Definition 151.205;*
4. *ledger: the base-change square of Definition 151.204, its displayed cover pullback rows, pulled-back common refinements, and the refinement ledgers used by Definition 151.175;*
5. *exactness: gluing pullback from Definition 151.206, refinement gluing descent from Definition 151.176, and common-refinement invariance from Definition 151.207 together with Definition 151.189;*
6. *standard bridge: absent, as in Definition 151.239.*

Thus base change is exposed as a paper-side obligation surface on indexed opens, not as a host-space powerset construction, and downstream certificates can cite it for cover pullback, gluing pullback, and common-refinement invariance.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange name certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange obligation surface`

Proof. The carrier and classifier rows are inherited from Definition 151.236. The represented source open i' determines the same fixed-open slice used by restricted-open consumers, so no new section carrier or classifier is introduced.

The stability and ledger rows are exactly the displayed data of Definition 151.204. Cover members, overlaps, and refinement maps remain indexed rows, while Definition 151.239 supplies the ambient restricted-open stability and Definition 151.175 supplies refinement pullback inside the ledger.

The exactness row combines the three base-change obligations. Cover-pullback compatibility is Definition 151.205; gluing pullback is Definition 151.206; common-refinement invariance is Definition 151.207, with Definition 151.176 and Definition 151.189 providing the existing sheaf exactness rows that the surface reuses.

Finally, the restricted-open surface already records that no standard bridge is attached. The base-change square only displays how indexed rows move across source and target opens, so it adds no bridge row and remains a paper-side $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface for downstream certificates. □

151.12 Sheaf base-change functoriality

Definition 151.209 (Sheaf composable base-change squares)**.** *Let $\mathcal{T}$, $\mathcal{T}'$, and $\mathcal{T}''$ be indexed open topology carriers, and let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over $\mathcal{T}$. Let*

$$\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$$

be displayed base-change squares in the sense of Definition 151.204. The two squares are composable when the following rows assemble into the displayed square $\mathcal{T}'' \to \mathcal{T}$:

1. *the open-index map is the composite $f \circ g$ and preserves finite intersections by composing the finite-intersection rows of g and f;*
2. *the represented inverse-image row for a target open i identifies the source open i'' with the two-step inverse image obtained first in $\mathcal{T}'$ and then in $\mathcal{T}''$;*
3. *cover ledgers compose pointwise: an indexed cover $\mathcal{U}$ of i pulls back to a cover $\mathcal{U}'$ of i' and then to a cover $\mathcal{U}''$ of i'' with the same index type and the same member order;*
4. *inclusion rows compose from each member $\iota''(a)$ through $\iota'(a)$ to $\iota(a)$, and the route to the target open is the composite of the two displayed routes;*
5. *memberwise pullback rows for sections, overlaps, and common-refinement members are classifier-equal to the corresponding rows read directly along the composite square.*

Thus a composable pair is not extra carrier data. It is the assertion that the displayed index maps, inverse-image representatives, finite-intersection rows, cover ledgers, inclusion rows, and memberwise pullback rows form the same downstream-readable base-change surface as the one-step square.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafComposableBaseChangeSquares`

Theorem 151.210 (Sheaf base-change identity square neutrality)**.** *For an indexed sheaf carrier $\mathcal{S}$ over $\mathcal{T}$, the identity displayed base-change square on $\mathcal{T}$ preserves the cover, restriction, gluing, and common-refinement rows of Definition 151.208. In particular, reading the $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ obligation surface through the identity square gives the same indexed cover members, overlap witnesses,*

restriction composites, displayed gluing rows, and common-refinement comparisons as reading the surface directly.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChangeIdentitySquare restriction neutrality`

Proof. The identity square uses the identity map on open indices. Its inverse-image representative for an open i is i itself, and its displayed cover pullback sends each cover member $\iota(a)$ to the same indexed member with the same inclusion row.

Cover compatibility follows because the overlap of two unchanged members is the original finite intersection. Restriction identity in the indexed section carrier identifies each pulled-back local section with the section already present in the target cover ledger.

For gluing, Definition 151.206 specializes to the identity restriction path. The displayed row is therefore classifier-equal to the original gluing row by restriction identity and classifier reflexivity.

For common refinements, the displayed span, the refinement maps, and the member inclusions are unchanged. Applying Definition 151.207 along the identity square returns the same comparison row, and Definition 151.189 reads the same global classifier class. □

Theorem 151.211 (Sheaf base-change cover-pullback composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i in $\mathcal{T}$. Pulling $\mathcal{U}$ through the composite square gives the same cover-member witnesses, member inclusion routes, finite-intersection overlap rows, and compatible local-family rows as pulling $\mathcal{U}$ first through f and then through g, up to the classifier identifications recorded by the composable-square rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change composition`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger base change cover pullback composition`

Proof. Start with a compatible family over $\mathcal{U}$. Apply Definition 151.205 to the square f. This gives a compatible displayed family over the pulled-back cover $\mathcal{U}'$ in $\mathcal{T}'$.

Apply the same theorem to g and the compatible family over $\mathcal{U}'$. The result is a compatible family over the two-step cover $\mathcal{U}''$ in $\mathcal{T}''$.

By Definition 151.209, the composite square has the same cover index type A, the same displayed member order, and the same member inclusion route as the two-step cover ledger. Its finite-intersection preservation row is the composite of the two overlap rows.

Restriction composition identifies the two-step section pullback with the direct pullback along the composite route. Classifier transport then replaces the two-step overlap comparisons by the one-step comparisons, so the downstream cover-pullback surface is the same in both readings. □

Theorem 151.212 (Sheaf base-change gluing pullback composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U}$ be an indexed cover of i and let u_a be a compatible local family glued by $s : \mathsf{Sec}(i)$. The displayed gluing row transported along the composite square is classifier-equal to the row obtained by first transporting the gluing row along f and then transporting the resulting row along g.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger base change gluing pullback composition`

Proof. Apply Definition 151.206 to f. It transports the gluing row for s to a displayed gluing row over the cover pulled back to $\mathcal{T}'$.

Apply Definition 151.206 again to g. This transports the intermediate gluing row to the two-step pulled-back cover over $\mathcal{T}''$.

The composability rows identify the source open of the one-step composite square with the two-step represented inverse image. They also identify each pulled-back cover member and member inclusion route with the corresponding two-step member route.

Restriction composition in the sheaf carrier identifies the restriction of s along the composite route with the successive restrictions through the intermediate open. Classifier transport along the displayed member identifications converts the two-step gluing comparison into the direct composite-square comparison, giving the claimed classifier equality of gluing rows. □

Theorem 151.213 (Sheaf base-change common-refinement composition)**.** *Let $\mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$ be composable displayed base-change squares from Definition 151.209. Let $\mathcal{U}$ and $\mathcal{V}$ be indexed cover presentations of i whose compatible local families agree on a displayed common refinement. The common-refinement comparison obtained after pulling through the composite square agrees with the comparison obtained after successive pullbacks through f and g. Consequently the global classifier class supplied by Definition 151.189 is independent of whether downstream consumers read the composite square in one step or as the two displayed squares.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan base change composition`

Proof. Apply Definition 151.207 to f. This produces pulled-back cover presentations over $\mathcal{T}'$ and transports the displayed common-refinement comparison to the intermediate common refinement.

Apply the same theorem to g. The intermediate comparison becomes a comparison over the two-step pulled-back common refinement in $\mathcal{T}''$.

The composable-square definition identifies the one-step pulled-back covers, refinement maps, member inclusions, and finite-intersection rows with their two-step counterparts. In particular, each common-refinement member has the same downstream restriction route after composing the displayed inclusion rows.

Restriction composition and classifier transport move the two-step comparison to the direct composite comparison. Then Definition 151.189 reads either comparison as the same global classifier class over the represented source open, so the common-refinement surface is independent of the chosen displayed factorization. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBaseChange common refinement composition`

Theorem 151.214 (Sheaf base-change three-square associativity)**.** *Let*

$$\mathcal{T}''' \xrightarrow{h} \mathcal{T}'' \xrightarrow{g} \mathcal{T}' \xrightarrow{f} \mathcal{T}$$

be three displayed base-change squares for an indexed sheaf carrier $\mathcal{S}$ over $\mathcal{T}$. Assume that the adjacent pairs are composable in the sense of Definition 151.209, and that both induced pairwise composites are again composable with the remaining square. Then the two bracketed displayed composites from $\mathcal{T}'''$ to $\mathcal{T}$ determine the same downstream base-change surface. More explicitly, first composing f with g and then with h, or first composing g with h and then with f, gives the same classified rows for every pulled-back indexed cover, every pulled-back compatible family, every transported gluing row, and every pulled-back common-refinement presentation.

Proof. Fix an indexed cover $\mathcal{U} = (A, \iota, r, \gamma)$ of an open i in $\mathcal{T}$. Apply Definition 151.211 to the pair (f, g). The cover obtained over $\mathcal{T}''$ by first pulling through f and then through g has the same index type A, the same member order, the same finite-intersection overlap rows, and the same compatible-family restriction rows as the cover read through the composite square $f \circ g$. Applying

the same theorem to the resulting composite square and h identifies the left-bracketed cover over $\mathcal{T}'''$ with the three-step cover obtained by pulling through f, then g, then h.

Apply Definition 151.211 instead to the pair (g, h), and then to f together with that composite. This identifies the right-bracketed cover over $\mathcal{T}'''$ with the same three-step cover ledger: the index type is still A, the member attached to $a : A$ is the inverse-image representative obtained by the same three displayed member rows, and the inclusion route to i is the same composite of displayed inclusion routes. Associativity of ordinary composition of the displayed open-index maps and of the restriction-route concatenations identifies the two parenthesizations of that route. Hence both bracketings classify the same cover-pullback rows and the same pulled-back compatible local-family rows.

Now let u_a be a compatible family over $\mathcal{U}$ and let $s : \mathsf{Sec}(i)$ be a displayed gluing section for it. Applying Definition 151.212 to (f, g) and then to the composite with h identifies the left-bracketed gluing row with the row obtained by restricting s successively along the three displayed square routes. Applying the same theorem to (g, h) and then to f identifies the right-bracketed gluing row with that identical successive restriction row. Restriction composition in the indexed sheaf carrier removes the parentheses from the route, and classifier transport along the composable-square member identifications gives equality of the two transported gluing rows.

Finally take two indexed cover presentations of i with compatible families agreeing on a displayed common refinement. By Definition 151.213, the left bracketing has the same pulled-back refinement indices, refinement maps, member inclusions, and comparison rows as the three-step pullback. The right bracketing is identified with that same three-step common-refinement ledger by applying the same theorem in the other pairwise order. The common-refinement members therefore restrict through the same downstream routes, and Definition 151.189 reads the two transported comparisons as the same global classifier class over the represented open of $\mathcal{T}'''$. □

Definition 151.215 (Sheaf consumer trace composition source)**.** *Let T_1 and T_2 be finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces of Definition 151.244. Their consumer trace composition $T_1 \star T_2$ is the finite trace obtained by listing every constructor of T_1 in its given order and then listing every constructor of T_2 in its given order. The empty trace is a two-sided unit for this operation.*

The source of this operation is restricted to the same objects as the access trace grammar: BHist *histories,* hsame *transport, optional* Cont *route histories,* $\mathsf{Topology}^{\uparrow}$ *indexed opens,* $\mathsf{PreSheaf}^{\uparrow}$ *section carriers, and* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *rows. No constructor of $T_1 \star T_2$ may mention host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows except through the sealed consumer projections already present in Definition 151.244.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerAccessTraceCompositionSource`
Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerTraceCompositionSource`

Theorem 151.216 (Sheaf consumer trace concatenation coverage)**.** *For any finite traces T_1 and T_2 of Definition 151.244, every constructor in the composed trace $T_1 \star T_2$ of Definition 151.215 lands in the same root face or sealed consumer projection that it occupied before composition.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append membership coverage`

Proof. Proceed by the ordered list presentation of $T_1 \star T_2$.

For a constructor appearing in the first segment, apply Definition 151.245 to its occurrence in T_1. Its landing face is one of the root faces of Definition 151.271 or one of the sealed consumer projections of Definition 151.252; the concatenation operation only preserves the occurrence and its order inside the first segment.

For a constructor appearing in the second segment, apply the same coverage theorem to its occurrence in T_2. The first segment supplies no wrapper and no coercion around the second segment, so the second occurrence keeps the landing face it had before composition.

Thus every constructor in the concatenated trace has exactly the face assigned to it by the original access trace coverage theorem. □

Theorem 151.217 (Sheaf consumer trace composition no-feedback)**.** *Composing finite* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback consumer traces by Definition 151.215 creates no additional* $\mathsf{Sheaf}^{\uparrow}$ *carrier row, classifier row, exactness row, or bridge row.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append no zero extension`

Proof. First apply Definition 151.246 to the consumer constructors in T_1. Each $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, and topology pullback event reads through the sealed projection and contributes no root carrier, classifier, exactness, or bridge row.

Apply the same theorem to the consumer constructors in T_2. The composition source merely places the second trace after the first, so no event in the second segment obtains a new sheaf row from the preceding segment.

It remains to rule out feedback across the boundary between the two segments. By Definition 151.274, consumer-specific algebraic, chart, and pullback data are outside the root sheaf faces. By Definition 151.199, all allowed downstream sheaf reads have already been exhausted by the listed root rows and sealed projections.

Therefore the concatenation boundary carries only the finite order of reads. It supplies no constructor capable of manufacturing a further $\mathsf{Sheaf}^{\uparrow}$ carrier, classifier, exactness, or bridge row. □

Theorem 151.218 (Sheaf consumer trace factorization composes)**.** *Suppose two allowed downstream* $\mathsf{Sheaf}^{\uparrow}$ *accesses by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factor through finite traces* T_1 *and* T_2 *of Definition 151.244. Their sequential use factors through the composed trace* $T_1 \star T_2$ *of Definition 151.215.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger trace factorization composes`

Proof. Choose the trace $T_1 \star T_2$ by listing the constructors witnessing the first downstream access and then listing the constructors witnessing the second downstream access. This is a finite trace because both inputs are finite and the composition source is ordinary finite concatenation.

The first access factors through the first segment by the assumed factorization of T_1. The second access factors through the second segment by the assumed factorization of T_2, and the preceding segment changes only the position of the second segment in the list.

By Definition 151.216, every constructor in both segments still lands in its original root face or sealed projection. By Definition 151.217, the boundary between the two segments creates no additional sheaf row that would require another access constructor.

The sequential downstream use is therefore represented by first reading along T_1 and then reading along T_2, which is exactly the composed finite trace. □

Definition 151.219 (Sheaf cover-presentation classifier)**.** *Let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier* $\mathcal{T}$*. Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *and* $\mathcal{V} = (B, \kappa, q, \delta)$ *be indexed open covers of the same open index* $i : \mathsf{OpenIx}$*. Let* $u_a : \mathsf{Sec}(\iota(a))$ *be compatible over* $\mathcal{U}$ *and let* $v_b : \mathsf{Sec}(\kappa(b))$ *be compatible over* $\mathcal{V}$*.*

The two cover presentations $(\mathcal{U}, u)$ and $(\mathcal{V}, v)$ are classifier-equal when there exists a displayed common refinement span

$$\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$$

from $\mathcal{U}$ to $\mathcal{V}$ such that their pulled-back local families agree pointwise by the section classifiers:

$$\forall c : C, \quad \rho_{\iota(r(c)),\lambda(c)}(u_{r(c)}) \sim_{\lambda(c)} \rho_{\kappa(s(c)),\lambda(c)}(v_{s(c)}).$$

This classifier is a presentation-level row: it compares indexed covers only through displayed common refinements, restriction maps, and the already exposed section classifiers $\sim_j$.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafCoverPresentationClassifier`

Theorem 151.220 (Sheaf cover-presentation classifier soundness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$ and let $v_b : \mathsf{Sec}(\kappa(b))$ be compatible over $\mathcal{V}$. If $(\mathcal{U}, u)$ and $(\mathcal{V}, v)$ are classifier-equal in the sense of Definition 151.219, then any displayed global gluing $g : \mathsf{Sec}(i)$ of u_a and any displayed global gluing $h' : \mathsf{Sec}(i)$ of v_b satisfy*

$$g \sim_i h'.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison soundness`

Proof. Unpack the cover-presentation classifier. It supplies a displayed common refinement span $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ from $\mathcal{U}$ to $\mathcal{V}$ and a pointwise classifier equality between the two pulled-back local families over $\mathcal{W}$.

Apply Definition 151.189 to the same two covers, compatible families, displayed common refinement, pullback equality, and displayed gluings. The conclusion is exactly $g \sim_i h'$. □

Theorem 151.221 (Sheaf cover-presentation classifier symmetry)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^{\uparrow}$ over an indexed open topology carrier $\mathcal{T}$. Let $\mathcal{U} = (A, \iota, h, \gamma)$ and $\mathcal{V} = (B, \kappa, q, \delta)$ be indexed open covers of the same open index $i : \mathsf{OpenIx}$. Let $u_a : \mathsf{Sec}(\iota(a))$ be compatible over $\mathcal{U}$ and let $v_b : \mathsf{Sec}(\kappa(b))$ be compatible over $\mathcal{V}$. If $(\mathcal{U}, u)$ is classifier-equal to $(\mathcal{V}, v)$ in the sense of Definition 151.219, then $(\mathcal{V}, v)$ is classifier-equal to $(\mathcal{U}, u)$.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafCoverPresentationClassifier symmetric`
↪ variant `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan symmetric`
↪ variant `BEDC.Derived.SheafUp.SheafDisplayedCommonRefinementSpan symm`

Proof. Unpack the classifier witness. It consists of a displayed common refinement span $\mathcal{W} = (C, \lambda, r, s, \epsilon, \zeta, \omega)$ and, for every $c : C$, a classifier row from the $\mathcal{U}$-pullback to the $\mathcal{V}$-pullback.

Build the reversed displayed common refinement by keeping the same cover index C, the same open family λ, and the same cover row ω, while swapping the two projection maps and their inclusion witnesses. Thus the reversed span has projection $s : C \to B$ to $\mathcal{V}$ and projection $r : C \to A$ to $\mathcal{U}$.

For each $c : C$, apply symmetry of the section classifier on $\mathsf{Sec}(\lambda(c))$ to the original pullback equality. The resulting pointwise classifier row is exactly the equality required for the reversed cover-presentation classifier. □

Theorem 151.222 (Sheaf restricted-open cover-presentation classifier descent)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$, and fix an open index i :* OpenIx*. Let $\mathcal{S}|_i$ be the fixed-open restricted carrier of Definition 151.239. For two compatible local-family presentations over covers of i, the cover-presentation classifier of Definition 151.219 restricts to the same classifier on $\mathcal{S}|_i$: if the ambient presentations are classifier-equal by a displayed common refinement, then their restricted-open global gluings are identified by the restricted classifier on the fixed-open carrier.*

Proof. Read the restricted-open carrier through Definition 151.239. Its carrier is the fixed open section surface $\mathsf{Sec}(i)$ with the inherited classifier $\sim_i$, so a restricted-open global gluing is an ambient global gluing whose target open is fixed to i.

Use Definition 151.238 to read the locality and gluing rows of $\mathcal{S}|_i$ as the ambient sheaf locality and gluing rows over covers of the same fixed open. Thus the displayed restricted-open gluings provide exactly the gluing hypotheses required by the ambient cover-presentation comparison.

Apply Definition 151.220. Since the restricted classifier is inherited from $\sim_i$, the ambient conclusion $g \sim_i h'$ is the desired restricted-open classifier equality. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted open descent`

Definition 151.223 (Sheaf downstream root source)**.** *The downstream root source for* $\mathsf{Sheaf}^{\uparrow}$ *is the displayed certificate surface obtained by taking the indexed-open carrier of* $\mathsf{Topology}^{\uparrow}$*, the restriction carrier of* $\mathsf{PreSheaf}^{\uparrow}$*, the locality and gluing surface of* $\mathsf{Sheaf}^{\uparrow}$*, the cover-nerve ledger of Definition 151.160, the point-germ ledger of Definition 151.231, and the refinement exactness rows of Definition 151.179 and Definition 151.189.*

This source exposes only indexed opens, BHist *membership rows, restriction maps, section classifiers, compatible local-family predicates, point-germ comparisons, displayed refinements, and cover-presentation exactness. It contains no host powerset equality and no quotient of cover presentations.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafDownstreamRootSource`

Theorem 151.224 (Sheaf RingedSpace local ring row)**.** *Let $\mathcal{S}$ be an indexed* $\mathsf{Sheaf}^{\uparrow}$ *carrier over an indexed-open* $\mathsf{Topology}^{\uparrow}$ *carrier $\mathcal{T}$. Suppose each section carrier* $\mathsf{Sec}(i)$ *carries the* $\mathsf{CommRing}^{\uparrow}$ *operation rows required for a local-ring row, and suppose every restriction map $\rho_{i,j}$ preserves addition, multiplication, zero, one, unit detection, maximal-ideal detection, and the section classifier. Then the source of Definition 151.223 exposes the structure-sheaf rows consumed by* $\mathsf{RingedSpace}^{\uparrow}$*: open-indexed commutative ring sections, classifier-stable restrictions, pointwise local-ring readbacks through the point-germ ledger, and local compatibility of the section classifier.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRingedSpaceLocalRing row`

Proof. Project the downstream source of Definition 151.223. Its carrier rows are the indexed sections, classifiers, restrictions, locality and gluing fields, cover ledger, point-germ ledger, and refinement exactness rows already supplied by $\mathsf{Sheaf}^{\uparrow}$.

Use Definition 151.300 for restriction stability. The assumed operation preservation rows ride on the same restriction routes, so the ring operations and the section classifier are transported together along every inclusion $j \leq_{\mathcal{T}} i$.

For local-ring readback, use the point-germ ledger of Definition 151.231. A row (i, o_i, s) records an indexed open containing a history cell and a section over that open; comparison is performed

on a displayed common neighborhood by restriction and the section classifier. Thus unit and maximal-ideal rows are repacked as point-germ rows without forming a stalk quotient.

The resulting package has exactly the rows requested by the $\mathsf{RingedSpace}^{\uparrow}$ consumer: a structure sheaf indexed by opens, ring-preserving restrictions, classifier-stable pointwise readback, and local-ring data visible through the existing ledger rows. □

Theorem 151.225 (Sheaf Scheme gluing cover row)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be a displayed affine cover presentation of an indexed open i for an indexed $\mathsf{Sheaf}^{\uparrow}$ carrier $\mathcal{S}$. Suppose each member $\iota(a)$ carries a local ringed chart row, and suppose the chart rows are compatible on the finite-intersection overlap rows $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(b)$ after restriction of sections, local-ring operations, and section classifiers. Then the downstream root source exposes a global section classifier row consumed by $\mathsf{Scheme}^{\uparrow}$: displayed local chart sections glue to a global section over i, and any two displayed gluings determine the same global classifier class after common-refinement comparison.*

Proof. First read cover membership from Definition 151.299. Every displayed history cell of the ambient open has a cover member witness, and pairwise overlap rows are supplied by finite intersections in the $\mathsf{Topology}^{\uparrow}$ carrier.

The assumed overlap compatibility is exactly the compatible local-family predicate of Definition 151.155, with the local ringed chart rows transported by restriction. Applying Definition 151.301 gives a displayed global section and identifies any two global readbacks that agree on the cover.

Refinement descent then supplies invariance under a different displayed affine cover. Use Definition 151.176 to pass gluing along refinements, Definition 151.179 for the refined exactness row, and Definition 151.189 for common-refinement comparison.

Thus $\mathsf{Scheme}^{\uparrow}$ receives a global section classifier row assembled from cover membership, finite-intersection overlaps, locality, gluing, refinement descent, and cover-presentation exactness, with no quotient of cover presentations. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeGluingCoverRow chart trace cons readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace cons readback`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace result deterministic`

Theorem 151.226 (Sheaf topology pullback cover row)**.** *Let $f : Y \to X$ be a classifier-respecting BHist map, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover row for a $\mathsf{Sheaf}^{\uparrow}$ carrier over X. Pull the cover back through the $\mathsf{Topology}^{\uparrow}$ pullback-open rows of Definition 88.60. Then the pulled-back cover preserves cover membership, restriction stability, and compatible local-family predicates: each source history cell covered by the pulled-back ambient open has a pulled-back cover-member witness, restriction maps commute with classifier transport along the pulled-back inclusion rows, and compatible local sections remain compatible after pullback to finite-intersection overlap rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafTopologyPullbackCover membership restriction compatibility`

Proof. Start with the topology pullback-open rows. The map row for f transports source classifier witnesses to target classifier witnesses, while Definition 88.62 preserves finite intersections and Definition 88.64 preserves displayed cover unions.

Apply Definition 151.299 to the target cover row and then precompose the resulting membership witnesses with f. The pulled-back cover-member witness is the same index $a : A$, read through the pulled-back predicate of Definition 88.60.

For restrictions, use Definition 151.300. The source inclusion rows are obtained by pulling back the target inclusions, and the classifier-respecting row for f supplies the transport needed to compare the source-side section classifiers.

Finally, compatibility of local families is checked on pairwise finite-intersection overlap rows. Since finite meets commute with the displayed pullback row and restriction classifier transport is stable along the same routes, every target overlap equality pulls back to the corresponding source overlap equality. This is the compatible local-family predicate required by the downstream topology pullback consumer. □

151.13 Identity-cover root package

Definition 151.227 (Sheaf identity-cover root package)**.** *For an indexed open i in the* $\mathsf{Sheaf}^{\uparrow}$ *root-obligation surface, the identity-cover root package is the accepted one-member cover of i with cover index type* Unit *and member map $\iota : \mathsf{Unit} \to \mathsf{OpenIx}$ constantly equal to i. Its membership row is the identity row on* $\mathsf{OpenAt}(i, b)$ *for each $b : \mathsf{BHist}$, its only overlap row is the finite-intersection cell $i \wedge_{\mathsf{Topology}^{\uparrow}} i$, and its exposed restriction route is the identity restriction $\rho_{i,i}$.*

The package ledger is restricted to BHist *membership cells,* hsame *transport of those cells, the self-overlap projections for $i \wedge_{\mathsf{Topology}^{\uparrow}} i$, and the presheaf identity row identifying $\rho_{i,i}(s)$ with s in the classifier $\sim_i$. It carries no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafIdentityCoverRootPackage`

Theorem 151.228 (Sheaf identity-cover membership exactness)**.** *Let i be an indexed open and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of Definition 151.227. For every $b : \mathsf{BHist}$, the displayed membership row is exact:*

$$\mathsf{OpenAt}(i, b) \iff \exists a : \mathsf{Unit},\ \mathsf{OpenAt}(\iota(a), b).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger identity open membership cont exactness`

Proof. For the forward direction, use the unique member of Unit. Since ι is constantly i, the required displayed cover-member cell is the original cell $\mathsf{OpenAt}(i, b)$ read through the identity membership row.

For the reverse direction, destruct the displayed witness $a : \mathsf{Unit}$. The member map again reads as i, so the cover-member cell is exactly a cell of $\mathsf{OpenAt}(i, b)$.

Both directions stay inside the BHist ledger and use only hsame transport when a history cell is reindexed. No equality of ambient host covers is used. □

Theorem 151.229 (Sheaf identity-cover restriction readback)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier, and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of an indexed open i. For every section $s : \mathsf{Sec}(i)$, the restriction route exposed by $\mathcal{U}_{\mathrm{id}}$ is classifier-equal to $\rho_{i,i}(s)$ and hence classifier-equal to s by the presheaf identity row:*

$$\rho^{\mathcal{U}_{\mathrm{id}}}(s) \sim_i \rho_{i,i}(s), \qquad \rho_{i,i}(s) \sim_i s.$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIdentityCover restriction readback`

Proof. Unpack the identity-cover root package. Its only cover member is i, and the route from the ambient open to that member is the identity route, so the exposed restriction readback is the displayed route $\rho_{i,i}(s)$.

Apply the identity row of the indexed section presheaf carrier from Definition 151.154. That row identifies $\rho_{i,i}(s)$ with s in the classifier $\sim_i$.

The self-overlap cell $i \wedge_{\mathsf{Topology}^\uparrow} i$ supplies no additional restriction target; it only records the overlap ledger row already attached to the identity route. □

Theorem 151.230 (Sheaf identity-cover gluing collapse)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^\uparrow$ over an indexed open topology carrier, and let $\mathcal{U}_{\mathrm{id}}$ be the identity-cover root package of an indexed open i. A compatible local family on $\mathcal{U}_{\mathrm{id}}$ is classifier-equivalent to exactly one global section over i: gluing supplies a global readback, and locality identifies any two global readbacks whose identity-cover restrictions agree.*

Proof. First apply the gluing part of Definition 151.301 to the compatible local family over $\mathcal{U}_{\mathrm{id}}$. This produces a global section $g : \mathsf{Sec}(i)$ whose restriction to the unique cover member agrees with the displayed local section.

The agreement row is read back through Definition 151.229. Because the only exposed route is $\rho_{i,i}$ and the presheaf identity row identifies $\rho_{i,i}(g)$ with g, the local family has no separate cover presentation beyond its single global readback.

For uniqueness, take two global readbacks whose restrictions agree on the identity cover. The membership exactness row of Definition 151.228 reduces coverwise agreement to agreement on the lone Unit member, and Definition 151.229 turns that agreement into agreement of the two global sections over i.

Apply the locality part of Definition 151.301. It returns classifier equality in $\sim_i$, so the compatible local family collapses to one global section up to the section classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafIdentityCover gluing collapse`

Definition 151.231 (Sheaf BHist point-germ ledger)**.** *Let $\mathcal{T}$ be an indexed open topology carrier for $\mathsf{Topology}^\uparrow$ whose point carrier is BHist, whose membership row is OpenAt, and whose finite-intersection index is written $\wedge_{\mathsf{Topology}^\uparrow}$. Let $\mathcal{S}$ be an indexed sheaf carrier for $\mathsf{Sheaf}^\uparrow$ over $\mathcal{T}$.*

For $b : \mathsf{BHist}$, a displayed point-germ row over b is a triple

$$(i, o_i, s)$$

where $i : \mathsf{OpenIx}$, $o_i : \mathsf{OpenAt}(i, b)$, and $s : \mathsf{Sec}(i)$. Two rows (i, o_i, s) and (j, o_j, t) compare over b when there is a displayed common neighborhood $k : \mathsf{OpenIx}$ with

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad \mathsf{OpenAt}(k, b),$$

and classifier equality after restriction:

$$\rho_{i,k}(s) \sim_k \rho_{j,k}(t).$$

The point-germ ledger is the collection of these rows and comparisons, with common neighborhoods supplied by the finite-intersection rows of $\mathcal{T}$ and all section comparison performed by the classifiers $\sim_k$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger section scope`

Theorem 151.232 (Sheaf BHist point-germ carrier scope)**.** *The point-germ ledger of Definition 151.231 exposes no carrier beyond the* BHist *point row, the membership row* OpenAt*, indexed opens, finite-intersection rows, presheaf restrictions, and the section classifiers of the indexed sheaf carrier. In particular, each ledger row over* $b : \mathsf{BHist}$ *is determined by an indexed open containing* b *and a section on that open, and each comparison is determined by a displayed common neighborhood below the two opens together with one classifier equality after restriction.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger common open comparison`

Proof. Unfold the displayed point-germ row. Its components are exactly $b : \mathsf{BHist}$, an indexed open i, a membership witness $o_i : \mathsf{OpenAt}(i, b)$, and a section $s : \mathsf{Sec}(i)$.

Unfold the comparison row. It asks only for a common index k below the two source opens, a membership witness $\mathsf{OpenAt}(k, b)$, and the classifier row $\rho_{i,k}(s) \sim_k \rho_{j,k}(t)$.

The existence of a canonical common-neighborhood shape is supplied by the finite-intersection field of Definition 88.1, while restriction maps and section classifiers are exactly the rows of Definition 151.154. No host stalk, quotient carrier, or powerset equality is used by either the row data or the comparison data. □

Theorem 151.233 (Sheaf BHist point-germ restriction readback)**.** *Let* (i, o_i, s) *be a displayed point-germ row over* $b : \mathsf{BHist}$ *in the sense of Definition 151.231. If* $j \leq_{\mathcal{T}} i$ *and* $o_j : \mathsf{OpenAt}(j, b)$*, then the restricted row*

$$(j, o_j, \rho_{i,j}(s))$$

compares to the original row over the common neighborhood j*. Explicitly, the comparison is witnessed by* $j \leq_{\mathcal{T}} i$*, the identity inclusion* $j \leq_{\mathcal{T}} j$*, the membership row* o_j*, and the identity classifier row*

$$\rho_{j,j}(\rho_{i,j}(s)) \sim_j \rho_{i,j}(s).$$

Proof. The hypotheses give the restricted point-germ row: the target open is j, the visible membership witness is o_j, and the section is the ambient restriction $\rho_{i,j}(s)$.

Use j itself as the displayed common neighborhood. One projection is the given inclusion $j \leq_{\mathcal{T}} i$, the other is the identity inclusion $j \leq_{\mathcal{T}} j$, and the point witness is o_j.

It remains only to supply the classifier equality required by Definition 151.231. The identity field of Definition 151.154 applied to the section $\rho_{i,j}(s) : \mathsf{Sec}(j)$ gives $\rho_{j,j}(\rho_{i,j}(s)) \sim_j \rho_{i,j}(s)$, which is exactly the comparison row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restriction readback`

Theorem 151.234 (Sheaf BHist point-germ gluing readback)**.** *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an indexed open cover of* $i : \mathsf{OpenIx}$ *for an indexed sheaf carrier* $\mathcal{S}$ *over a* BHist*-scoped topology carrier* $\mathcal{T}$*. Suppose* $s_a : \mathsf{Sec}(\iota(a))$ *is compatible over* $\mathcal{U}$ *and* $g : \mathsf{Sec}(i)$ *is a gluing section with*

$$\forall a : A, \quad \rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a.$$

For every $b : \mathsf{BHist}$ *with* $o_i : \mathsf{OpenAt}(i, b)$*, the cover law supplies* $a : A$ *and* $o_a : \mathsf{OpenAt}(\iota(a), b)$*, and the point-germ row* (i, o_i, g) *reads back through that member:*

$$(i, o_i, g)$$

compares over b *to*

$$(\iota(a), o_a, s_a).$$

If g' is another gluing section for the same compatible local family, then locality gives $g \sim_i g'$, so the resulting point-germ readback is unique up to the same section classifier and its restrictions to displayed common neighborhoods.

Proof. Apply the cover law γ to b and o_i. It returns a cover member $a : A$ together with $o_a : \mathsf{OpenAt}(\iota(a), b)$.

Use $\iota(a)$ as the common neighborhood for the two point-germ rows. The projection $\iota(a) \leq_{\mathcal{T}} i$ is the cover inclusion h_a, the projection $\iota(a) \leq_{\mathcal{T}} \iota(a)$ is the identity inclusion, and the membership witness is o_a.

The gluing row gives $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$. Combining this with the identity restriction row $\rho_{\iota(a),\iota(a)}(s_a) \sim_{\iota(a)} s_a$, using symmetry and transitivity of the classifier on $\mathsf{Sec}(\iota(a))$, gives the comparison required by Definition 151.231.

For uniqueness, let g' be another gluing of the same compatible family. Apply Definition 151.169 to the two displayed gluing rows to obtain $g \sim_i g'$. The restriction classifier-stability field of Definition 151.154 transports this equality to any displayed common neighborhood used in the point-germ ledger, so the readback does not depend on the chosen gluing representative beyond the existing section classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger gluing readback`

Theorem 151.235 (Sheaf BHist point-germ comparison transitivity)**.** *Let (i, o_i, s), (j, o_j, t), and (ℓ, o_ℓ, u) be displayed point-germ rows over the same $b : \mathsf{BHist}$ in the sense of Definition 151.231. Suppose the first two rows compare on a displayed common neighborhood k:*

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad o_k : \mathsf{OpenAt}(k, b),$$

with

$$\alpha : \rho_{i,k}(s) \sim_k \rho_{j,k}(t),$$

and suppose the last two rows compare on a displayed common neighborhood m:

$$m \leq_{\mathcal{T}} j, \qquad m \leq_{\mathcal{T}} \ell, \qquad o_m : \mathsf{OpenAt}(m, b),$$

with

$$\beta : \rho_{j,m}(t) \sim_m \rho_{\ell,m}(u).$$

Put

$$n := k \wedge_{\mathsf{Topology}^{\uparrow}} m.$$

Then n is a displayed common neighborhood below i and ℓ at b, and the first and third rows compare there:

$$\rho_{i,n}(s) \sim_n \rho_{\ell,n}(u).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison trans`
↪ variant `BEDC.Derived.SheafUp.SheafBHistPointGermLedger shared open classifier transitivity`

Proof. The finite-intersection field of Definition 88.1 gives, for every $x : \mathsf{BHist}$,

$$\mathsf{OpenAt}(n, x) \iff (\mathsf{OpenAt}(k, x) \wedge \mathsf{OpenAt}(m, x)).$$

The forward implication, followed by projection, gives inclusions $n \leq_{\mathcal{T}} k$ and $n \leq_{\mathcal{T}} m$. At $x = b$, the reverse implication applied to o_k and o_m gives $o_n : \mathsf{OpenAt}(n, b)$. Composing the pointwise

inclusions gives $n \leq_{\mathcal{T}} i$ and $n \leq_{\mathcal{T}} \ell$; composing with $k \leq_{\mathcal{T}} j$ and with $m \leq_{\mathcal{T}} j$ gives the two chains to j used below.

Restrict the comparison row α along $n \leq_{\mathcal{T}} k$ by the classifier-stability field of Definition 151.154. This gives

$$\rho_{k,n}(\rho_{i,k}(s)) \sim_n \rho_{k,n}(\rho_{j,k}(t)).$$

The composition field of the same presheaf carrier, applied to $n \leq_{\mathcal{T}} k \leq_{\mathcal{T}} i$ and to $n \leq_{\mathcal{T}} k \leq_{\mathcal{T}} j$, gives

$$\rho_{k,n}(\rho_{i,k}(s)) \sim_n \rho_{i,n}(s)$$

and

$$\rho_{k,n}(\rho_{j,k}(t)) \sim_n \rho_{j,n}(t).$$

Symmetry and transitivity of the classifier $\sim_n$ therefore yield

$$\rho_{i,n}(s) \sim_n \rho_{j,n}(t).$$

Restrict the comparison row β along $n \leq_{\mathcal{T}} m$. The classifier-stability field gives

$$\rho_{m,n}(\rho_{j,m}(t)) \sim_n \rho_{m,n}(\rho_{\ell,m}(u)).$$

The composition field for $n \leq_{\mathcal{T}} m \leq_{\mathcal{T}} j$ and $n \leq_{\mathcal{T}} m \leq_{\mathcal{T}} \ell$ gives

$$\rho_{m,n}(\rho_{j,m}(t)) \sim_n \rho_{j,n}(t)$$

and

$$\rho_{m,n}(\rho_{\ell,m}(u)) \sim_n \rho_{\ell,n}(u).$$

Again using symmetry and transitivity of $\sim_n$, obtain

$$\rho_{j,n}(t) \sim_n \rho_{\ell,n}(u).$$

Transitivity of $\sim_n$ combines the last displayed row with $\rho_{i,n}(s) \sim_n \rho_{j,n}(t)$ and gives

$$\rho_{i,n}(s) \sim_n \rho_{\ell,n}(u).$$

Together with the witnesses $n \leq_{\mathcal{T}} i$, $n \leq_{\mathcal{T}} \ell$, and $o_n : \mathsf{OpenAt}(n, b)$, this is exactly the point-germ comparison row required by Definition 151.231. □

Definition 151.236 (Sheaf restricted-open carrier)**.** *Let $\mathcal{T}$ be a* BHist*-scoped indexed open topology carrier for* $\mathsf{Topology}^{\uparrow}$*, and let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over $\mathcal{T}$. Fix an open index $i : \mathsf{OpenIx}$. The restricted-open carrier $\mathcal{S}|_i$ is the carrier indexed by pairs (j, η_j) with*

$$\eta_j : j \leq_{\mathcal{T}} i.$$

On such an index it inherits

$$\mathsf{Sec}_{|i}(j, \eta_j) := \mathsf{Sec}(j), \qquad \sim_{|i,(j,\eta_j)} := \sim_j .$$

For an inclusion $\theta : k \leq_{\mathcal{T}} j$ between restricted indices, its restriction map is the ambient sheaf restriction

$$\rho^{|i}_{(j,\eta_j),(k,\eta_k)} := \rho_{j,k}.$$

The target witness $\eta_k : k \leq_{\mathcal{T}} i$ is the composite of θ with η_j, so every carrier, classifier, and restriction row stays inside the fixed open i without adding a host subspace, powerset restriction, quotient, or sheafification primitive.

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier`

Theorem 151.237 (Sheaf restricted-open restriction laws)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. For every restricted index (j, η_j) and section $s : \mathsf{Sec}_{|i}(j, \eta_j)$, the inherited identity row satisfies*

$$\rho^{|i}_{(j,\eta_j),(j,\eta_j)}(s) \sim_{|i,(j,\eta_j)} s.$$

For every chain of restricted indices

$$(k, \eta_k) \leq_{\mathcal{T}} (j, \eta_j) \leq_{\mathcal{T}} (l, \eta_l)$$

and every $s : \mathsf{Sec}_{|i}(l, \eta_l)$, the inherited composition row satisfies

$$\rho^{|i}_{(j,\eta_j),(k,\eta_k)}(\rho^{|i}_{(l,\eta_l),(j,\eta_j)}(s)) \sim_{|i,(k,\eta_k)} \rho^{|i}_{(l,\eta_l),(k,\eta_k)}(s).$$

Moreover, for every restricted inclusion $(k, \eta_k) \leq_{\mathcal{T}} (j, \eta_j)$, classifier equality transports along the inherited restriction:

$$s \sim_{|i,(j,\eta_j)} t \Longrightarrow \rho^{|i}_{(j,\eta_j),(k,\eta_k)}(s) \sim_{|i,(k,\eta_k)} \rho^{|i}_{(j,\eta_j),(k,\eta_k)}(t).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier restriction laws`

Proof. Unfold Definition 151.236. The section carrier and classifier at (j, η_j) are exactly $\mathsf{Sec}(j)$ and $\sim_j$, and the identity restriction is the ambient row $\rho_{j,j}$.

Apply the identity field of the indexed section presheaf carrier in Definition 151.154. This gives the first displayed classifier equality.

For the composition row, forget only the fixed target witness to i. The displayed restricted chain is an ambient chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} l$, so the composition field of Definition 151.154 gives the second displayed equality after replacing each restricted restriction map by the corresponding ambient ρ.

The classifier-transport row is identical. A restricted classifier equality is an ambient equality $s \sim_j t$, and the restricted map from (j, η_j) to (k, η_k) is $\rho_{j,k}$. The restriction classifier-stability field of Definition 151.154 therefore gives the final displayed implication. □

Theorem 151.238 (Sheaf restricted-open locality and gluing)**.** *Let $\mathcal{S}|_i$ be the restricted-open carrier of Definition 151.236. Let (j, η_j) be a restricted index and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed open cover of j whose member inclusions satisfy $h_a : \iota(a) \leq_{\mathcal{T}} j$. Regard it as a cover inside $\mathcal{S}|_i$ by assigning each member the composite witness*

$$\eta_{\iota(a)} : \iota(a) \leq_{\mathcal{T}} i.$$

Then locality and gluing hold in the restricted carrier. Thus, if $g, h' : \mathsf{Sec}_{|i}(j, \eta_j)$ and

$$\forall a : A, \quad \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(g) \sim_{|i,(\iota(a),\eta_{\iota(a)})} \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(h'),$$

then $g \sim_{|i,(j,\eta_j)} h'$. If $s_a : \mathsf{Sec}_{|i}(\iota(a), \eta_{\iota(a)})$ is compatible on the restricted cover, then there is $g : \mathsf{Sec}_{|i}(j, \eta_j)$ with

$$\forall a : A, \quad \rho^{|i}_{(j,\eta_j),(\iota(a),\eta_{\iota(a)})}(g) \sim_{|i,(\iota(a),\eta_{\iota(a)})} s_a.$$

Compatible local families remain compatible after every further restriction inside $\mathcal{S}|_i$.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier locality gluing descent`

Proof. View the restricted cover as the ambient indexed open cover of j with the same index type A, the same open-index map ι, the same inclusion witnesses h_a, and the same cover law γ. The extra witnesses $\eta_{\iota(a)}$ only record that each member also lies below the fixed open i.

After unfolding Definition 151.236, the displayed memberwise equality is exactly the ambient antecedent for the locality field of Definition 151.155. Applying that field to the ambient cover of j gives $g \sim_j h'$, which is the restricted classifier equality at (j, η_j).

For gluing, a compatible family in the restricted carrier is the same family $s_a : \mathsf{Sec}(\iota(a))$ with the same overlap equalities as in the ambient sheaf. Apply the gluing field of Definition 151.155 to obtain $g : \mathsf{Sec}(j)$ whose ambient restrictions classify as s_a. Unfolding the restricted carrier turns those rows into the displayed restricted gluing rows.

It remains to check that compatibility is stable under further restriction inside the fixed open. Fix two local members and restrict them to a smaller restricted index below their overlap. The finite-intersection row of Definition 88.1 supplies the overlap inclusions, and Definition 151.237 supplies restriction composition along the chains through that overlap. Transporting the original compatibility row by the same theorem and composing the resulting classifier equalities proves that the further-restricted family is still compatible. □

Theorem 151.239 (Sheaf restricted-open NameCert surface)**.** *For every* BHist*-scoped indexed open topology carrier* $\mathcal{T}$*, indexed sheaf carrier* $\mathcal{S}$ *over* $\mathcal{T}$*, and open index* $i : \mathsf{OpenIx}$*, the restricted-open carrier* $\mathcal{S}|_i$ *supplies the paper-side* $\mathsf{NameCert}_{\mathsf{Sheaf}\uparrow}$ *surface used by open-local consumers such as ringed-space and scheme certificates. Its rows are:*

1. *carrier: restricted indices* (j, η_j) *with* $j \leq_{\mathcal{T}} i$ *and inherited section carriers* $\mathsf{Sec}(j)$*;*
2. *classifier: the inherited section classifiers* $\sim_j$*;*
3. *stability: identity, composition, and classifier transport from Definition 151.237, with classifier transport agreeing with Definition 151.172;*
4. *ledger: open-cover indices below* i*, cover-member inclusions, finite-intersection overlap rows, refinement maps, and restriction paths;*
5. *exactness: restricted locality and gluing from Definition 151.238, stable under refinement by Definition 151.175 and Definition 151.176;*
6. *standard bridge: absent, as in the global obligation surface of Definition 151.190.*

Thus an open-local consumer may work entirely below i *while reusing the global* $\mathsf{NameCert}_{\mathsf{Sheaf}\uparrow}$ *carrier, classifier, stability, ledger, exactness, and absent-standard-bridge rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRestrictedOpenCarrier semantic name certificate`

Proof. The carrier and classifier rows are immediate from Definition 151.236: every restricted index names an ambient open below i, and no section carrier or classifier is rebuilt.

The stability row is supplied by Definition 151.237. Its classifier-transport clause is the restricted form of Definition 151.172, because both statements use the same ambient restriction map $\rho_{j,k}$ and the same classifier family $\sim$.

The ledger row is inherited from the ambient cover and refinement ledgers. Covers below i are ambient covers whose target open also carries a witness to i, while refinement pullback and gluing descent are exactly the rows of Definition 151.175 and Definition 151.176 read inside the fixed-open slice.

Finally, Definition 151.238 gives the exactness row for locality and gluing below i. The global obligation surface of Definition 151.190 already records that no standard bridge is attached, and

restricting to a fixed open adds no such bridge. These rows are precisely the $\mathsf{NameCert}_{\mathsf{Sheaf}^\uparrow}$ surface required by open-local consumers. □

Definition 151.240 (Sheaf root access normal form)**.** *The* $\mathsf{Sheaf}^\uparrow$ *root access normal form is the finite classification of downstream-visible root reads generated by Definition 151.244. Its classes are:*

1. *carrier, classifier, and restriction-route rows for the indexed section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, restrictions* $\rho_{i,j}$*, identity rows, composition rows, classifier transport, and optional* Cont *route histories;*
2. *cover-member and overlap rows for accepted covers, member witnesses, finite-intersection overlaps, pair-overlap projections, and* hsame *membership transport;*
3. *locality, gluing, refinement, common-refinement, and base-change rows for compatible-family predicates, global witnesses, uniqueness rows, displayed refinements, refinement pullbacks, gluing descent, cover-presentation exactness, displayed common refinements, displayed base-change squares, cover pullback, compatible-family transport, and gluing pullback;*
4. *point-germ comparison rows that expose named pointwise readbacks without forming stalk quotients;*
5. *the three downstream projection faces: the* $\mathsf{RingedSpace}^\uparrow$ *local-ring projection, the* $\mathsf{Scheme}^\uparrow$ *affine-chart projection, and the topology pullback projection.*

These classes are read-only faces of the same BHist*-indexed sheaf root package. They contain no class for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, standard bridges, or consumer-private algebraic, chart, or pullback data.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootAccessNormalFormSourceCase`

Theorem 151.241 (Sheaf root access normal-form coverage)**.** *Every admitted* $\mathsf{Sheaf}^\uparrow$ *root access trace of Definition 151.244 lands in one class of Definition 151.240.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead locality gluing readback`

Proof. Proceed by cases on the trace constructor of Definition 151.244.

Carrier, classifier, and restriction-route reads land in the first normal-form class. The root face source of Definition 151.271 places section carriers, section classifiers, restrictions, identity rows, composition rows, classifier transport, and optional route histories in the restriction-route face, and Definition 151.272 gives the corresponding coverage.

Cover-membership and overlap reads land in the second class. The accepted-cover row, member witness, finite-intersection overlap, pair-overlap projection, and hsame transport are exactly the cover-membership rows isolated by Definition 151.245.

Locality, gluing, refinement, common-refinement, and base-change reads land in the third class. Their exactness rows are the locality, gluing, and refinement face of Definition 151.271, with gluing exactness supplied by Definition 151.273; the downstream boundary of Definition 151.196 records the same restricted-open, base-change, refinement, and exactness rows.

Point-germ reads land in the fourth class because Definition 151.244 admits named pointwise readbacks only as point-germ comparison rows. The same definition excludes stalk quotient formation, so the point-germ case does not branch into an additional quotient class.

The remaining constructors are the three downstream consumer projections. They land in the fifth class by the projection seal of Definition 151.252, and the face coverage theorem Definition 151.245 identifies no further trace constructor. Thus every admitted access has a normal-form landing class. □

Theorem 151.242 (Sheaf root access normal-form consumer factorization)**.** *Every downstream* $\mathsf{Sheaf}^{\uparrow}$ *access used by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors first through Definition 151.240 and then through the existing projection seals.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace consumer access trace`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootAccessNormalForm consumer factorization`

Proof. Begin with an allowed downstream access. By Definition 151.247, it is represented by a finite trace of Definition 151.244. Apply Definition 151.241 to send each trace constructor to its normal-form class.

For $\mathsf{RingedSpace}^{\uparrow}$, the relevant classes are carrier, classifier, restriction, cover-gluing, restricted-open, base-change, and point-germ comparison rows. The local-ring projection seal Definition 151.253 reads exactly those sheaf-side classes and supplies no commutative-ring or local-ring row from the sheaf root.

For $\mathsf{Scheme}^{\uparrow}$, first pass through the ringed-space projection and then use the cover-presentation and displayed common-refinement classes. The affine-chart projection seal Definition 151.254 identifies spectra and chart-ring rows as consumer-side data, while Definition 151.276 lists the remaining sheaf rows visible to the joint face.

For topology pullback, use the base-change class together with cover-membership, restriction-route, compatible-family, gluing, and common-refinement rows. The pullback projection seal Definition 151.255 is exactly this factor, and Definition 151.277 prevents the other two consumer faces from entering it.

The factorization therefore has two stages: access traces land in the finite normal form, and the three downstream consumers read their respective normal-form classes only through the sealed projections. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace root access factorization`

Theorem 151.243 (Sheaf root access normal form exports no extra rows)**.** *The normal form of Definition 151.240 exports no host powerset equality, no cover quotient, no stalk quotient, no external sheafification row, no standard bridge row, and no consumer-private algebraic, chart, or pullback data.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootAccessNormalForm no extra rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead normal form readback`

Proof. For host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridges, inspect the source grammar. The trace source of Definition 151.244 has no constructor for those rows, and the root face exactness source of Definition 151.271 explicitly keeps them outside the four root faces.

For consumer-private algebraic rows, apply Definition 151.253. It says the commutative-ring and local-ring data used by $\mathsf{RingedSpace}^{\uparrow}$ are supplied outside $\mathsf{Sheaf}^{\uparrow}$, so the normal form cannot export them as sheaf root rows.

For affine-chart and spectra rows, apply Definition 151.254. The scheme face reads sheaf rows through ringed-space projections, cover-presentation exactness, and common-refinement comparison, while spectra data and chart-ring data remain outside the sheaf root.

For pullback-private choices, apply Definition 151.255. The topology pullback face reads displayed base-change transport, pulled-back cover membership, compatible-family transport, gluing pullback, and common-refinement transport; it does not export consumer-specific pullback-open choices as root data.

Finally combine the joint face and downstream boundary. The joint consumer face noninterference theorem Definition 151.277 prevents private consumer data from feeding back into the root, and Definition 151.199 exhausts the sheaf-side rows consumed downstream. Thus the normal form exports only the listed finite classes and no extra row. □

Definition 151.244 (Sheaf consumer access trace source)**.** *A finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access trace is the freely generated finite list syntax whose constructors are the following read events:*

1. *carrier and classifier reads of the indexed section carriers* $\mathsf{Sec}(i)$ *and classifiers* $\sim_i$*;*
2. *restriction-route reads of identity, composition, classifier transport, and optional* Cont *route-history rows;*
3. *cover-membership reads of accepted covers, member witnesses, finite-intersection overlaps, and* hsame *membership transport;*
4. *locality and gluing reads of compatible-family predicates, local agreement, global witnesses, and uniqueness rows;*
5. *refinement and common-refinement reads of displayed refinements, refinement pullbacks, gluing descent, and common-refinement comparison rows;*
6. *point-germ reads of named pointwise readback rows without forming stalk quotients;*
7. *base-change reads of displayed base-change squares, pulled-back cover membership, pulled-back compatible families, and gluing transport;*
8. *the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring consumer projection;*
9. *the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart consumer projection;*
10. *the topology pullback consumer projection.*

Every constructor is formed over BHist *histories,* hsame *transport, optional* Cont *route histories, the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier, the* $\mathsf{PreSheaf}^{\uparrow}$ *section carrier, and the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *rows. The trace source has no constructor for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafConsumerAccessTraceSource`

Theorem 151.245 (Sheaf consumer access trace face coverage)**.** *Every constructor of Definition 151.244 lands in exactly one face: either one of the root faces of Definition 151.271 or one of the three consumer projections sealed in Definition 151.252.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead restriction route readback`

Proof. Proceed by cases on the freely generated trace constructor.

The carrier, classifier, restriction-route, cover-membership, locality, gluing, refinement, common-refinement, point-germ, and base-change cases are root-facing reads. The carrier and classifier rows

land in the restriction-route face, the accepted-cover and overlap rows land in the cover-membership face, and the compatibility, locality, gluing, refinement, common-refinement, point-germ, and base-change rows land in the locality, gluing, and refinement exactness face. This placement is the coverage statement of Definition 151.272, with exactness supplied by Definition 151.273.

The remaining cases are the three downstream consumer constructors. The $\mathsf{RingedSpace}^{\uparrow}$ constructor lands in Definition 151.253, the $\mathsf{Scheme}^{\uparrow}$ constructor lands in Definition 151.254, and the topology pullback constructor lands in Definition 151.255.

These targets are disjoint by construction of Definition 151.271: the first three faces contain root sheaf rows, while the downstream face contains only projections. Thus each constructor has exactly one landing face. □

Theorem 151.246 (Sheaf consumer access trace no-feedback)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback constructors of Definition 151.244 do not create further* $\mathsf{Sheaf}^{\uparrow}$ *carrier rows, classifier rows, exactness rows, or bridge rows.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace no feedback`

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, the local-ring constructor reads the sheaf-side carrier, classifier, restriction, point-germ, restricted-open, and base-change rows only through the sealed projection. By Definition 151.274, the commutative-ring and local-ring data are outside the sheaf root and cannot return as root carrier, classifier, or exactness rows.

For $\mathsf{Scheme}^{\uparrow}$, the affine-chart constructor first passes through the ringed-space projection and then reads cover-presentation and common-refinement rows. The same separation theorem keeps spectra data and chart-ring data outside the sheaf root, so the chart constructor contributes no root row.

For topology pullback, the constructor reads displayed base-change transport, cover pullback, compatible-family transport, gluing pullback, and common-refinement transport. The pullback-open choices are consumer-side data by Definition 151.274, not root sheaf obligations.

Finally apply Definition 151.199. The downstream boundary exhausts all sheaf-side reads and excludes host powerset equality, cover quotients, external sheaf models, and bridge closure, so no consumer constructor can manufacture a further $\mathsf{Sheaf}^{\uparrow}$ root row. □

Theorem 151.247 (Sheaf consumer access trace exhaustion)**.** *Every allowed downstream* $\mathsf{Sheaf}^{\uparrow}$ *access by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors through a finite access trace of Definition 151.244.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append exhaustion`

Proof. Inspect the downstream row lists in Definition 151.196. Carrier and classifier access is represented by the carrier-classifier constructor; restriction stability is represented by the restriction-route constructor; cover ledgers are represented by the cover-membership constructor; locality, gluing, refinement, common-refinement, restricted-open, point-germ, and base-change access are represented by their corresponding trace constructors.

Now inspect the consumer projections in Definition 151.252. The $\mathsf{RingedSpace}^{\uparrow}$ local-ring projection is represented by the ringed-space consumer constructor, the $\mathsf{Scheme}^{\uparrow}$ affine-chart projection by the scheme consumer constructor, and the displayed pullback projection by the topology pullback constructor.

By Definition 151.245, each selected constructor lands in the correct root face or sealed consumer projection. By Definition 151.246, selecting a consumer constructor does not add any further sheaf row that would require another constructor outside the trace grammar.

The relevant downstream proof uses only finitely many rows from the displayed boundary and projection lists. Listing those rows in their order of use gives the required finite trace, and every allowed access is therefore factored through Definition 151.244. □

Definition 151.248 (Sheaf consumer access trace composition source)**.** *Let P be the fixed* BHist*-indexed* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root package used by Definition 151.244. If T_1 and T_2 are finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces over P, their composite $T_1 \mathbin{++} T_2$ is the finite list concatenation obtained by reading all constructors of T_1 followed by all constructors of T_2. The source constructors of $T_1 \mathbin{++} T_2$ are still exactly the constructors of Definition 151.244: carrier and classifier reads, restriction-route reads, cover-membership reads, locality and gluing reads, refinement and common-refinement reads, point-germ reads, base-change reads, and the three sealed consumer projection reads for* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback.*

The composite carries no additional constructor for host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge rows. Its only extra operation is finite list concatenation inside the same BHist*-scoped root package P.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTraceCompositionSource membership split`

Theorem 151.249 (Sheaf consumer access trace concatenation closure)**.** *Accepted finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces over one* BHist*-indexed* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root package are closed under the composition operation of Definition 151.248. In particular, if T_1 and T_2 are generated by Definition 151.244, then $T_1 \mathbin{++} T_2$ is generated by the same grammar and contains no constructor for host powerset equality, cover quotients, stalk quotients, external sheafification, or standard bridge rows.*

Proof. Induct on the second trace T_2 as a finite list of constructors. In the empty case, $T_1 \mathbin{++} T_2$ is just T_1, so acceptance is the given acceptance of the first trace.

In the cons case, write T_2 as a shorter trace followed by one final constructor c. By the induction hypothesis, concatenating T_1 with the shorter trace remains an accepted trace over the same root package.

It remains to append c. The constructor cases are exactly the cases listed in Definition 151.244: carrier and classifier, restriction route, cover membership, locality and gluing, refinement and common refinement, point germ, base change, and the three downstream consumer projections. Each case appends one already admitted constructor, so the resulting list is still accepted by the same grammar.

The excluded host-level operations are not constructor cases. Since finite concatenation only appends constructors already present in the grammar, it cannot create a host powerset equality row, a cover quotient, a stalk quotient, an external sheafification step, or a standard bridge row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace append closed`

Theorem 151.250 (Sheaf consumer access trace face-map stability)**.** *For finite* $\mathsf{Sheaf}^{\uparrow}$ *consumer access traces T_1 and T_2 over the same* BHist*-indexed root package, the landing-face assignment of Definition 151.245 sends $T_1 \mathbin{++} T_2$ to the finite concatenation of the landing-face reads of T_1 and the landing-face reads of T_2. The same assignment preserves the no-feedback separation of Definition 151.246 across the composite trace.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace face map stability`

Proof. Proceed by constructor cases on the appended segment T_2. The empty appended segment contributes no landing-face read, so the landing-face list of the composite is the landing-face list of T_1.

For a root-facing constructor, Definition 151.245 gives its unique root face. Appending the constructor appends exactly that face read after the face reads already assigned to the previous part of the composite.

For a downstream consumer constructor, Definition 151.245 sends the constructor to its sealed consumer projection. By Definition 151.246, that projection does not return a new $\mathsf{Sheaf}^{\uparrow}$ carrier row, classifier row, exactness row, or bridge row.

Thus the landing-face assignment commutes with finite list concatenation, and the downstream no-feedback separation remains valid after the two traces are read sequentially. □

Theorem 151.251 (Sheaf composite consumer access exhaustion)**.** *Every sequential downstream* $\mathsf{Sheaf}^{\uparrow}$ *access by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback factors through one finite* BHist*-scoped trace obtained by concatenating the finite traces for the individual accesses.*

Proof. Apply Definition 151.247 to the first downstream access. This gives a finite trace over the displayed BHist-indexed $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ root package whose constructors are those of Definition 151.244.

Apply Definition 151.247 again to the next downstream access, or to each member of the finite sequential access list. Each application gives another finite trace over the same root package, because the downstream consumers read the sealed $\mathsf{Sheaf}^{\uparrow}$ root package rather than creating a replacement root.

Concatenate the resulting finite traces in their order of downstream use. By Definition 151.249, the concatenation is still an accepted finite trace in the same grammar.

Finally use Definition 151.250. The composite trace preserves the landing-face assignment and the no-feedback separation, so the sequential access never leaves the finite BHist-scoped trace surface and never imports host-level sheaf machinery. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafConsumerAccessTrace composite exhaustion`

Definition 151.252 (SheafUp root consumer projection seal)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root consumer projection seal is the three-consumer package assembled from the consumer-threshold projection source of Definition 151.256, the root face exactness source of Definition 151.271, and the downstream consumption boundary of Definition 151.196. Its three public projections are the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring projection, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart projection, and the topology pullback projection. Each projection is a read-only face of the same* BHist*-indexed sheaf root package, and none of the projections exports host powerset equality, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootConsumerProjectionSeal`

Theorem 151.253 (SheafUp RingedSpace local-ring projection seal)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face of Definition 151.252 may read only the* $\mathsf{Sheaf}^{\uparrow}$ *section carriers, section classifiers, restriction stability rows, cover-gluing exactness rows, restricted-open rows, and displayed base-change rows. The commutative-ring and local-ring data used by the consumer are supplied from* $\mathsf{CommRing}^{\uparrow}$ *outside* $\mathsf{Sheaf}^{\uparrow}$*. Through this sealed projection, the consumer cannot read host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Proof. The consumer-threshold source of Definition 151.256 names the ringed-space projection row as a face of the root threshold package. Its existing projection theorem Definition 151.257 identifies exactly the section carrier, classifier, restriction, point-germ readback, restricted-open, and base-change rows available to $\mathsf{RingedSpace}^{\uparrow}$.

The root face coverage and exactness results Definition 151.272, Definition 151.273, and Definition 151.274 show that these rows are projections of the root cover-membership, restriction-route, and locality-gluing-refinement faces. They add no root carrier and no consumer-specific exactness row.

Finally, Definition 151.199 exhausts the downstream sheaf-side reads. The local-ring and commutative-ring structure therefore enters only from $\mathsf{CommRing}^{\uparrow}$, while host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure remain outside the sealed projection. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRingedSpaceLocalRingProjection seal`

Theorem 151.254 (SheafUp Scheme affine-chart projection seal)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face of Definition 151.252 may read affine-chart gluing only through cover-presentation rows, displayed common-refinement rows, and the* $\mathsf{RingedSpace}^{\uparrow}$ *projection rows sealed in Definition 151.253. Spectra data and chart-ring data are supplied outside* $\mathsf{Sheaf}^{\uparrow}$*. Through this sealed projection, the consumer cannot read host powerset equality of opens, cover quotients, stalk quotients, external sheafification, or standard bridge closure.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootSchemeAffineChartProjection seal`

Proof. The consumer-threshold source of Definition 151.256 names the scheme projection row as chart-wise access through ringed-space projections, cover-presentation exactness, and common-refinement comparison. The existing projection theorem Definition 151.258 supplies this chart-wise reading of the same threshold source.

The root face exactness package then separates the sheaf rows from chart data. By Definition 151.272 and Definition 151.273, cover-presentation comparison and common-refinement invariance are already part of the locality, gluing, and refinement exactness face; by Definition 151.274, chart-local spectra and ring rows do not feed back into the root sheaf package.

Apply Definition 151.199 to the scheme consumption boundary. Every sheaf row consumed by $\mathsf{Scheme}^{\uparrow}$ is already listed there, so spectra data, chart-ring data, host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure are not supplied by $\mathsf{Sheaf}^{\uparrow}$. □

Theorem 151.255 (SheafUp topology pullback projection seal)**.** *The topology pullback face of Definition 151.252 may read only pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and common-refinement transport along the displayed base-change surface. It does not read the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face or the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and it exports no host powerset equality, cover quotient, stalk quotient, external sheafification, or standard bridge closure.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootTopologyPullbackProjection seal`

Proof. The consumer-threshold source of Definition 151.256 names the topology pullback projection as the displayed base-change face. The existing projection theorem Definition 151.259 identifies the available rows as cover pullback, gluing pullback, pulled-back common refinement, and classifier transport along the displayed square.

The root face coverage and exactness rows control those reads. Use Definition 151.272 for cover membership and restriction-route stability, and use Definition 151.273 for compatible-family transport, gluing pullback, and common-refinement comparison.

The separation theorem Definition 151.274, together with Definition 151.199, removes the other two consumer faces from the topology pullback projection. Thus the pullback consumer receives only the displayed base-change rows and no host powerset equality, cover quotient, stalk quotient, external sheafification, or standard bridge closure. □

Definition 151.256 (Sheaf root consumer-threshold projection source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root consumer-threshold projection source is the* BHist*-indexed package obtained from Definition 151.294 by exposing three consumer faces: the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing face, and the topology pullback face. It is recorded by the following ten fields.*

1. *The traditional target is the sheaf-side data of a structure sheaf that can be read by ringed spaces, schemes, and pullback-open consumers.*
2. *The BEDC source is the single threshold source of Definition 151.294, not a separate host sheaf object.*
3. *The carrier row is the indexed-open section carrier* $\mathsf{Sec}(i)$*, with restricted-open carriers and displayed source carriers obtained only by slicing or pulling back indexed opens.*
4. *The classifier row is the section classifier family* $\sim_i$*, together with its restricted-open and displayed base-change readbacks.*
5. *The restriction row is restriction identity, restriction composition, and classifier transport for every displayed inclusion used by the three consumers.*
6. *The RingedSpaceUp projection row is the local-ring consumer surface: section carriers, section classifiers, ring-preserving restriction routes, point-germ local-ring readback, restricted-open rows, and base-change rows.*
7. *The SchemeUp projection row is the affine-chart gluing surface: chart-wise ringed-space projections, displayed affine cover membership, overlap compatibility, and common-refinement comparison.*
8. *The topology pullback projection row is the displayed base-change surface: cover pullback, gluing pullback, pulled-back common refinement, and classifier transport along the displayed square.*
9. *The ledger and exactness row is the cover-nerve ledger, point-germ ledger, restricted-open ledger, refinement ledger, cover-presentation exactness, and base-change exactness already present in the threshold source.*
10. *The standard-bridge field is absent: the projection source exports no host powerset cover equality, no cover quotient, no stalk quotient, no external sheafification operation, and no* $\mathsf{StandardBridge}$*.*

Thus the same BHist*-indexed source is projected separately to the three consumer families without duplicating the sheaf root obligations.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjectionSource`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjectionSource projection faces`
Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootConsumerThresholdProjection source`

Theorem 151.257 (Sheaf root RingedSpace consumer projection)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring consumer receives exactly the indexed sections, section classifiers, restriction identity, restriction*

composition, restriction transport, point-germ local-ring readback, restricted-open rows, and base-change rows of Definition 151.256. No additional sheaf carrier, cover comparison, stalk quotient, or standard bridge is available through this projection.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootConsumerRingedSpaceProjection semantic name certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRingedSpaceConsumerProjection scope`

Proof. Start from Definition 151.197. Its sheaf-side list asks for section carriers, classifiers, restriction stability, cover-gluing exactness, restricted-open rows, and base-change rows, together with non-sheaf commutative-ring data supplied outside $\mathsf{Sheaf}^{\uparrow}$.

The carrier and classifier entries are supplied by Definition 151.295. This gives the open-indexed section carrier, the classifier family, restriction identity and composition, classifier transport, BHist cover membership, and point-germ rows read on displayed neighborhoods.

The exactness and transport entries are supplied by Definition 151.296. Locality, gluing, refinement pullback, restricted-open exactness, and displayed base-change exactness are already fields of the root threshold source, so the ringed-space projection is precisely the listed face of Definition 151.256. □

Theorem 151.258 (Sheaf root Scheme consumer projection)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing consumer receives the same threshold source by chart-wise* $\mathsf{RingedSpace}^{\uparrow}$ *projection, followed by cover-presentation exactness and common-refinement exactness on chart overlaps. Thus affine-chart sections, overlap classifiers, glued section classes, and comparison across common refinements are all read from Definition 151.256.*

Proof. Apply Definition 151.198. Each affine or ringed-space chart first reads the sheaf boundary through the ringed-space consumer, so the chart-wise carrier, classifier, restriction, restricted-open, point-germ, and base-change rows are the rows identified in Definition 151.257.

The coverage claim is Definition 151.297: every sheaf row consumed by $\mathsf{Scheme}^{\uparrow}$ is already covered by the threshold source. This keeps the chart gluing surface inside the same BHist-indexed package.

For independence of the chosen affine cover presentation, use Definition 151.189. The displayed common refinement compares the pulled-back local families and identifies the resulting global classifier class, so no quotient of cover presentations or external sheaf model is introduced. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafSchemeChartGluingTrace unary result`

Theorem 151.259 (Sheaf root topology pullback consumer projection)**.** *Topology pullback consumers receive only the displayed base-change, cover-pullback, gluing-pullback, and common-refinement rows of Definition 151.256. These rows preserve pulled-back cover membership, compatible local-family predicates, glued section classifiers, and common-refinement comparison, without exporting the ringed-space local-ring face or the scheme affine-chart face.*

Proof. Begin with Definition 151.226. It identifies the consumer request as a displayed pullback of indexed cover rows: cover membership, restriction stability, and compatible local-family predicates are transported along the topology pullback-open rows.

The complete base-change surface is Definition 151.208. Its displayed square supplies cover pullback, gluing pullback, and common-refinement invariance using the same indexed-open carrier, classifier family, cover ledger, and refinement exactness fields recorded in Definition 151.256.

Since this consumer reads only displayed source and target rows, the projection discards the ringed-space local-ring field and the scheme affine-chart field. What remains is exactly the base-change and common-refinement face of the root threshold source, with no host powerset cover comparison or standard bridge. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootConsumerPullbackProjection base change face`

Definition 151.260 (Sheaf root cover-descent source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root cover-descent source is the following* BHist*-scoped source tuple. It is read over a* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier with point carrier* BHist*, membership row* OpenAt*, inclusion relation* $\leq_{\mathcal{T}}$*, and finite-intersection rows; an indexed* $\mathsf{Sheaf}^{\uparrow}$ *carrier with section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, and restriction maps* $\rho_{i,j}$*; an indexed cover* $\mathcal{U} = (A, \iota, h, \gamma)$ *of an ambient open* i*; member inclusions* $h_a : \iota(a) \leq_{\mathcal{T}} i$*; displayed restriction routes to members and finite-intersection overlaps; a compatible local family* $s_a : \mathsf{Sec}(\iota(a))$*; and the section classifier rows used to compare restrictions on members, overlaps, and glued readbacks.*

The source admits only these displayed data: BHist *histories, indexed open membership, cover-member witnesses, member and overlap inclusions, restriction-route witnesses, compatible-family overlap equalities, locality, gluing, and section classifiers. It excludes stalks, quotient carriers, host powerset equality of covers, and external sheafification.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootCoverDescentSource`

Theorem 151.261 (Sheaf root cover-descent carrier exactness)**.** *The source of Definition 151.260 exposes exactly* BHist*,* OpenAt*, cover membership, restriction maps, finite-intersection overlap rows, and section classifiers. In particular, every carrier or classifier row visible to cover descent is one of the indexed-open carrier rows of* $\mathsf{Topology}^{\uparrow}$*, the indexed section rows of* $\mathsf{Sheaf}^{\uparrow}$*, or the compatible-family and locality rows of the sheaf cover surface.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent carrier exactness`

Proof. Unfold Definition 151.260. The topology component contributes only BHist, indexed opens, OpenAt, inclusion witnesses, and finite-intersection rows. These are exactly the displayed carrier and membership rows of Definition 88.1.

Unfold the indexed sheaf component. By Definition 151.154, the section-level rows are the carriers $\mathsf{Sec}(i)$, classifiers $\sim_i$, restriction maps $\rho_{i,j}$, and restriction stability fields attached to displayed inclusion routes.

Finally unfold the cover and local-family component through Definition 151.155. It adds cover membership, member inclusions, finite-intersection overlap comparisons, locality, and gluing. None of the three unfolded sources contains a stalk carrier, a quotient carrier, host powerset equality, or external sheafification, so the displayed list is exact. □

Theorem 151.262 (Sheaf root cover-descent gluing exhaustion)**.** *Every root cover-descent comparison route for the source of Definition 151.260 factors through a cover member or a displayed common refinement, then through restriction-route exactness, and finally through the locality or gluing classifier row. Thus the cover-descent route surface is exhausted by cover membership, finite-intersection overlap rows, restriction transport, point-germ readback, and gluing uniqueness.*

↪variant `BEDC.Derived.SheafUp.SheafBHistPointGermLedger cover descent exhaustion`

Proof. First apply the cover law γ in the indexed cover $\mathcal{U}$. Each history cell $b : \mathsf{BHist}$ lying in the ambient open is sent to a member $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$, and the finite-intersection rows of Definition 88.1 supply the displayed overlaps used for pairwise comparison.

Next route the comparison through the sheaf restriction surface. Apply Definition 151.291 to transport the member witness through the displayed route, and apply Definition 151.292 when the comparison is represented by a composed route history. Both steps keep the target section classifier attached to the reached open.

It remains to read the local family through gluing. Apply Definition 151.234 for the memberwise readback of a glued section, and apply Definition 151.169 to identify any two global readbacks that agree on all displayed members. The route therefore ends at the existing locality or gluing classifier row and introduces no further carrier or classifier. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent common refinement germ exactness`

Theorem 151.263 (Sheaf root cover-descent downstream interface)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers may consume only the cover-descent source of Definition 151.260, the carrier exactness row of Definition 151.261, the gluing exhaustion row of Definition 151.262, and the downstream exactness coverage row of Definition 151.266. They receive no additional sheaf carrier and no additional sheaf classifier.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverDescent downstream interface`

Proof. For the ringed-space consumer, apply Definition 151.224. Its sheaf-side data are section carriers, classifiers, restriction stability, point-germ readback, and local compatibility rows, all already present in the cover-descent source and its carrier exactness row.

For the scheme consumer, apply Definition 151.225. The chart gluing request is routed through cover membership, finite-intersection overlaps, compatible local families, gluing, and common-refinement comparison, which are precisely the rows exhausted by Definition 151.262.

For topology pullback, apply Definition 151.226. Pullback cover membership and compatible-family transport are displayed over the same indexed-open and restriction-route rows, so they add no carrier beyond Definition 151.260.

Finally apply Definition 151.266. It seals the three consumer projections to cover membership, locality and gluing, refinement descent, common-refinement exactness, and projection rows. Hence downstream access is exactly the cover-descent interface named in the statement. □

Theorem 151.264 (Sheaf root downstream consumer carrier scope)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback receive their* $\mathsf{Sheaf}^{\uparrow}$ *carrier rows only from the indexed section carrier over* BHist *opens in Definition 151.154, together with the cover-nerve ledger of Definition 151.160. Their sheaf-side carrier scope is exactly the indexed open carrier, the section carriers* $\mathsf{Sec}(i)$*, the section classifiers* $\sim_i$*, the restriction maps* $\rho_{i,j}$*, cover-member cells, pair-overlap cells, and* hsame*-stable ledger transport. No host powerset equality, cover quotient, stalk quotient, or external sheafification row is exported to these consumers.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer carrier scope`

Proof. Read the carrier source from Definition 151.298. Its carrier clause points to Definition 151.154, so every section-level carrier read by a downstream consumer is one of the open-indexed section carriers, classifiers, restrictions, and identity or composition rows already listed there.

Read the history side from Definition 151.160 and the cover-nerve export surface of Definition 151.302. The only additional cells are cover-member cells, finite-intersection pair-overlap cells, route-history cells when present, and membership transport through hsame.

Finally apply the downstream coverage theorem Definition 151.305 and the projection seal Definition 151.252. These rows are projected read-only to $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, and topology pullback, so none of the three consumers obtains host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row from the root package. □

Theorem 151.265 (Sheaf root downstream consumer classifier stability)**.** *Every restriction route exposed from the* $\mathsf{Sheaf}^{\uparrow}$ *root to* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback preserves the relevant section classifier. One-step and two-step routes preserve classifiers by Definition 151.304; membership reindexing is transported only through* hsame*; optional* Cont *route histories compare displayed route witnesses without changing the target open or the target classifier. Hence consumer restriction routes do not add any host powerset equality, cover quotient, stalk quotient, or external sheafification row.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer classifier stability`

Proof. For section restrictions, apply Definition 151.304. It states that the one-step restriction of classifier-equal sections lands in the target classifier and that the composed two-step route agrees with the direct route in the classifier of the final open.

For membership movement, use the cover-nerve ledger of Definition 151.160. Its transport row is hsame-stable membership transport over BHist cells, so route changes do not replace the indexed open carrier by a host open or a quotient carrier.

If optional Cont histories are present, use the route-history clause cited in Definition 151.300. Those histories compare the witnesses for displayed routes; they do not alter the open index on which $\sim_j$ or $\sim_k$ is read.

The three projection seals Definition 151.253, Definition 151.254, and Definition 151.255 then pass the same classifier-stable routes to the downstream consumers without exporting host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row. □

Theorem 151.266 (Sheaf root downstream consumer exactness coverage)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *exactness surface consumed by* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback is exhausted by cover membership, locality and gluing, refinement descent, common-refinement exactness, and the downstream projection seals. Concretely, cover membership is supplied by Definition 151.299 and Definition 151.303; locality and gluing are supplied by Definition 151.301; refinement descent and refined exactness are supplied by Definition 151.176 and Definition 151.179; common-refinement exactness is supplied by Definition 151.188 and Definition 151.189; and consumer access is sealed by the three projection theorems. No host powerset equality, cover quotient, stalk quotient, or external sheafification row is exported.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafDownstreamConsumer exactness coverage`

Proof. Start with cover membership. The accepted-cover cell is unpacked by Definition 151.299, and Definition 151.303 records the corresponding cover-member and pair-overlap cells inside the BHist cover nerve.

For local exactness, use Definition 151.301. It gives existence of a global readback from a compatible local family and classifier uniqueness for global readbacks whose restrictions agree on the cover.

For refinement, apply Definition 151.176 and the package theorem Definition 151.179. These rows keep the same global classifier class after passing to displayed refinements. For comparison of cover presentations, use Definition 151.188 and Definition 151.189.

Finally pass to consumers through Definition 151.253, Definition 151.254, and Definition 151.255. Each consumer receives a projection of the listed cover, restriction, locality, gluing, refinement, and common-refinement rows; none receives host powerset equality, a cover quotient, a stalk quotient, or an external sheafification row. □

Definition 151.267 (Sheaf root export BHist closure surface)**.** *The root export surface for* $\mathsf{Sheaf}^{\uparrow}$ *is the following ten-field* BHist*-closed package carried by* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*.*

1. *The traditional target is sheaves of sections over indexed opens.*
2. *The BEDC source is the joint* $\mathsf{Topology}^{\uparrow}$ *and* $\mathsf{PreSheaf}^{\uparrow}$ *surface over* BHist*, read through* NameCert*.*
3. *The history ledger is the cover-nerve ledger of Definition 151.160, with membership transport only through* hsame*.*
4. *The carrier is the indexed section presheaf of Definition 151.154.*
5. *The classifier is the open-indexed section classifier family* $\sim_i$*.*
6. *The stability rows are identity, composition, restriction classifier transport, and route-history associativity when* Cont *rows are present.*
7. *The cover ledger consists of indexed cover members, finite-intersection overlap rows, displayed refinement maps, and common-refinement spans.*
8. *The exactness rows are locality, gluing, refinement pullback, refinement gluing descent, and common-refinement gluing invariance.*
9. *The downstream threshold exports only these indexed-open, section, restriction, locality, gluing, and refinement rows.*
10. *The standard bridge row is absent; no host powerset equality, cover quotient, stalk quotient, or external sheafification primitive is exported.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootExportSurface`
Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootExportBHistClosureSurface`

Theorem 151.268 (Sheaf root export carrier scope)**.** *Let* $\mathcal{T}$ *be the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier and let* $\mathcal{S}$ *be the* $\mathsf{PreSheaf}^{\uparrow}$ *indexed section carrier equipped with the sheaf locality and gluing rows. The root export surface of Definition 151.267 exposes exactly the following carrier data: open indices and inclusion rows from* $\mathcal{T}$*, indexed section carriers* $\mathsf{Sec}(i)$*, section classifiers* $\sim_i$*, restriction maps* $\rho_{i,j}$*,* hsame*-stable* BHist *membership, and the cover-nerve ledger of Definition 151.160. It exports no carrier whose elements are host opens, quotiented covers, or external stalk representatives.*

Proof. Project the topology part first. The indexed-open carrier supplies open indices, inclusion rows, finite intersections, and the OpenAt membership predicate; Definition 151.160 records those rows over BHist and allows membership transport only through hsame.

Project the presheaf part next. By Definition 151.154, the only section-level carrier rows are $\mathsf{Sec}(i)$, $\sim_i$, the restriction maps, and their identity, composition, equivalence, and classifier-transport fields.

Finally use Definition 151.157 and Definition 151.299. These identify the carrier support and cover membership obligations as indexed-open and BHist-ledger data, so no host open carrier, cover quotient, or stalk representative is part of the export. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport carrier scope`

Theorem 151.269 (Sheaf root export stability and exactness package)**.** *For every indexed sheaf carrier $\mathcal{S}$ over an indexed-open carrier $\mathcal{T}$, the root export surface of Definition 151.267 packages the complete stability and exactness rows needed by downstream certificates:*

1. *cover-member witnesses and overlap projections from the* BHist *cover ledger;*
2. *identity, composition, and restriction classifier-stability for section restrictions;*
3. *locality and gluing exactness over indexed covers;*
4. *refinement pullback and refined-cover gluing descent;*
5. *common-refinement presentation exactness, including the restricted-open common-refinement rows below a fixed open.*

Every row is expressed in indexed opens, restriction maps, section classifiers, displayed refinements, common-refinement spans, Cont *route histories when present, and* hsame*-stable history membership.*

Proof. The cover-member and overlap rows are those of Definition 151.299, together with the pair-overlap projections of Definition 151.161. They keep the cover ledger internal to indexed opens and BHist cells.

The restriction rows are exactly Definition 151.300. Its identity, composition, classifier-stability, and optional Cont route-history clauses are the stability rows exported by the root package.

The locality and gluing rows are supplied by Definition 151.301. Refinement contributes Definition 151.175, Definition 151.176, and Definition 151.179.

Common-refinement exactness is read from Definition 151.188 and Definition 151.189; below a fixed open it is read from Definition 151.195. These rows assemble the displayed package without adding any host-level cover equality or quotient comparison. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport stability exactness package`

Theorem 151.270 (Sheaf root export downstream threshold)**.** *The root export surface of Definition 151.267 is the carried* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *threshold available to* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and restricted-open consumers. Such consumers receive only the indexed-open carrier, the section classifier family, restriction identity and composition rows, restriction classifier-stability rows, locality and gluing rows, the refinement ledger, common-refinement exactness, and the absent-standard-bridge row. In particular, they do not receive a standard bridge, host powerset cover equality, cover quotient, stalk quotient, or external sheafification operation from* $\mathsf{Sheaf}^{\uparrow}$.

Proof. For the global sheaf surface, apply Definition 151.190, Definition 151.283, and Definition 151.284. These rows expose the carrier, classifier, restriction, locality, gluing, and refinement obligations already present in $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

For the local consumers, apply Definition 151.239 and Definition 151.195. They show that working below a fixed open reuses the same ambient section carriers, classifiers, restrictions, cover ledgers, and exactness rows.

The negative part is part of the same export boundary. The root-unblock source of Definition 151.298 and the global obligation surface both state that no standard bridge or host-level

cover comparison is exported, so ringed-space, scheme, and restricted-open chapters must cite the indexed threshold rather than reconstructing a larger classical sheaf interface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootExport downstream threshold`

Definition 151.271 (Sheaf root face exactness source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root face exactness source is the four-face package obtained from the* BHist *root-unblock source, the cover-nerve export surface, and the consumer-threshold projection source. It is read over* BHist *histories,* hsame *transport, optional* Cont *route histories, the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier, the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction carrier, and the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation rows.*

1. *The cover-membership face consists of accepted cover rows, member witnesses* $\mathsf{OpenAt}(\iota(a), b)$*, finite-intersection overlap rows, and* hsame*-stable history transport from Definition 151.160.*
2. *The restriction-route face consists of the section carriers* $\mathsf{Sec}(i)$*, classifiers* $\sim_i$*, restrictions* $\rho_{i,j}$*, identity rows, composition rows, classifier transport, and optional* Cont *route-history associativity rows.*
3. *The locality, gluing, and refinement exactness face consists of compatible local-family predicates, gluing witnesses, locality comparison, displayed cover refinement, refinement pullback, refinement gluing descent, displayed common-refinement comparison, and cover-presentation exactness.*
4. *The downstream projection face consists only of the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring projection, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart gluing projection, and the topology pullback projection named in Definition 151.256.*

The package contains no host powerset equality of open covers, no cover quotient, no stalk quotient, no external sheafification operation, and no StandardBridge *field.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootFaceExactnessSource`

Theorem 151.272 (Sheaf root face classifier coverage)**.** *Every row of the root-unblock source of Definition 151.298 and every row of the cover-nerve export surface of Definition 151.302 lands in exactly one of the first three faces of Definition 151.271. The downstream projection face receives only projections of those faces and adds no root carrier, root classifier, or root exactness row.*

Proof. For the cover rows, read the cover ledger in Definition 151.160. The accepted cover, cover-member witness, finite-intersection overlap, pair-overlap projection, and hsame membership transport rows are precisely the cover-membership face. The exhaustion and projection theorems Definition 151.303 and Definition 151.161 show that no additional cover row is required.

For the restriction rows, use Definition 151.300 and Definition 151.304. Their identity, composition, classifier-preservation, two-step route, and optional Cont history clauses are exactly the restriction-route face.

For the exactness rows, use Definition 151.301, Definition 151.175, Definition 151.176, Definition 151.179, and Definition 151.189. These rows cover locality, gluing, displayed refinement, common-refinement comparison, and cover-presentation exactness, so they land in the locality, gluing, and refinement exactness face.

Finally compare with the consumer-threshold source of Definition 151.256. Its ringed-space, scheme, and topology pullback rows cite the same carrier, classifier, cover, restriction, and exactness rows through projection, so they occupy the downstream projection face rather than adding root data. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead coverage`

Theorem 151.273 (Sheaf root face gluing exactness)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over a* BHist*-scoped indexed open topology carrier $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an accepted indexed cover of $i : \mathsf{OpenIx}$. For every compatible local family $s_a : \mathsf{Sec}(\iota(a))$, locality plus gluing plus refinement exactness gives a* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*-consumable exactness row:*

1. *a global section $g : \mathsf{Sec}(i)$ restricts to the local family;*
2. *any two global readbacks whose cover restrictions agree are classifier-equal in $\sim_i$;*
3. *after any displayed refinement or displayed common refinement, the pulled-back local families determine the same global classifier class.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison restricted global soundness`

Proof. Begin with the compatible local family over $\mathcal{U}$. The gluing field in Definition 151.155 supplies a global section $g : \mathsf{Sec}(i)$ and the displayed member equalities $\rho_{i,\iota(a)}(g) \sim_{\iota(a)} s_a$.

Use locality from the same definition for uniqueness. If two global readbacks agree on every member row, locality returns classifier equality in $\sim_i$; the same argument is packaged by Definition 151.169.

Now pass to a displayed refinement. The compatibility row is transported by Definition 151.175, and the gluing row descends by Definition 151.176. The combined refined-cover statement is Definition 151.179.

For comparison of two cover presentations, use the displayed common-refinement surface of Definition 151.186. The invariance theorem Definition 151.188, together with Definition 151.189, identifies the two resulting global classifier classes.

All cited steps use indexed covers, BHist membership, hsame transport, section classifiers, restriction composition, and optional Cont route histories. Therefore the exactness row is consumable by $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ without adding a host cover comparison or quotient row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger restricted global soundness`

Theorem 151.274 (Sheaf root face downstream separation)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, and topology pullback consumers receive only projections of the four faces of Definition 151.271. Consumer-specific rows such as local-ring structure, affine-chart spectra data, and displayed pullback-open choices do not feed back into the root* $\mathsf{Sheaf}^{\uparrow}$ *carrier, root classifier, cover ledger, restriction-route face, or locality and gluing exactness face.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead downstream separation`

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, apply Definition 151.257. Its sheaf-side input is the indexed section carrier, section classifier, restriction stability, point-germ readback, restricted-open rows, and base-change rows projected from the root source. The commutative-ring and local-ring rows are supplied outside the sheaf root.

For $\mathsf{Scheme}^{\uparrow}$, apply Definition 151.258. Each chart first reads the ringed-space projection, while affine-chart and spectra data remain chart-side data. The sheaf contribution is only cover-presentation exactness and common-refinement comparison already present in the root exactness face.

For topology pullback consumers, apply Definition 151.259 and Definition 151.226. The pullback row transports cover membership, compatible families, gluing, and common refinements

through the displayed base-change surface, but it does not introduce a ringed-space local-ring row or a scheme affine-chart row.

The downstream consumption boundary of Definition 151.196 and the exhaustion theorem Definition 151.199 close the separation. They state that consumers can only read the listed sheaf-side projections, so consumer-specific structure cannot be returned as a new root $\mathsf{Sheaf}^{\uparrow}$ obligation. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead downstream projection separation`

Definition 151.275 (Sheaf root joint consumer face)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root joint consumer face is the shared* BHist*-indexed projection package assembled from the projection seal of Definition 151.252, the consumer-threshold projection source of Definition 151.256, the root face exactness source of Definition 151.271, and the downstream consumption boundary of Definition 151.196. It has one root package and three read-only consumer faces: the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and the topology pullback face. Each face reads the same* BHist*-scoped sheaf root rows for section carriers, section classifiers, restriction transport, cover-gluing exactness, restricted opens, displayed base change, cover presentations, and displayed common refinements.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace chart trace e0 endpoint absurd`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace normal form boundary`

Theorem 151.276 (Sheaf root joint consumer face exhaustion)**.** *Every sheaf-side row visible through the joint consumer face of Definition 151.275 to* $\mathsf{RingedSpace}^{\uparrow}$*,* $\mathsf{Scheme}^{\uparrow}$*, or topology pullback consumers is one of the following rows: indexed section carrier, section classifier, restriction identity, restriction composition, restriction classifier transport, cover-locality and cover-gluing exactness, restricted-open carrier and inherited sheaf rows, displayed base-change transport, cover-presentation exactness, or displayed common-refinement comparison. No other sheaf-side row is exported by the joint face.*

Proof. Split over the three projections named in Definition 151.252.

For the $\mathsf{RingedSpace}^{\uparrow}$ projection, apply Definition 151.253. Its sheaf-side reads are the section carrier, classifier, restriction-stability, cover-gluing, restricted-open, and base-change rows. The root face coverage theorem Definition 151.272 places the carrier, classifier, and restriction rows inside the first three faces of the root exactness source.

For the $\mathsf{Scheme}^{\uparrow}$ projection, apply Definition 151.254. Chart-wise reads factor through the ringed-space projection, and the only additional sheaf-side comparisons are cover-presentation exactness and displayed common-refinement comparison. These are exactly the locality, gluing, and refinement rows supplied by Definition 151.273.

For the topology pullback projection, apply Definition 151.255. The available rows are pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and displayed common-refinement transport along the base-change surface.

Finally use Definition 151.199. It exhausts downstream sheaf reads by the carrier, classifier, restriction, cover-gluing, restricted-open, base-change, refinement, and exactness rows already listed in the statement, so the joint consumer face exports no further sheaf-side row. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace exhaustion`

Theorem 151.277 (Sheaf root joint consumer face noninterference)**.** *None of the three consumer projections of Definition 151.275 exports another consumer's private data back into the* $\mathsf{Sheaf}^{\uparrow}$ *root face. The* $\mathsf{RingedSpace}^{\uparrow}$ *projection does not return private algebraic or local-ring rows to the root*

face; the $\mathsf{Scheme}^{\uparrow}$ *projection does not return private affine-chart, spectra, or chart-ring rows; and the topology pullback projection does not return private pullback-open choices or consumer-specific pullback witnesses to the root face.*

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, the projection seal Definition 151.253 says that commutative-ring and local-ring data are supplied outside $\mathsf{Sheaf}^{\uparrow}$. Thus those algebraic rows are read by the consumer only after the sheaf-side projection has been fixed.

For $\mathsf{Scheme}^{\uparrow}$, the projection seal Definition 151.254 keeps spectra data and chart-ring data outside $\mathsf{Sheaf}^{\uparrow}$. The scheme face reads sheaf rows through chart-wise ringed-space projections and common-refinement comparisons, but it does not add a chart carrier, chart classifier, or spectra row to the root package.

For topology pullback, the projection seal Definition 151.255 restricts the face to displayed base-change, cover pullback, gluing pullback, and common-refinement transport. It explicitly does not read the ringed-space local-ring face or the scheme affine-chart face.

The separation theorem Definition 151.274 identifies all three consumers as projections of the same four root faces. Together with Definition 151.199, this prevents consumer-specific algebraic, chart, or pullback data from re-entering the $\mathsf{Sheaf}^{\uparrow}$ carrier, classifier, cover ledger, restriction-route face, or locality and gluing exactness face. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace noninterference`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace no feedback unary boundary`

Theorem 151.278 (Sheaf root joint consumer face NameCert boundary)**.** *The joint consumer face of Definition 151.275 exposes the following ten-field* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *boundary.*

1. *The traditional target is the sheaf-side consumer projection package read by ringed spaces, schemes, and topology pullback consumers.*
2. *The BEDC sources are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{NameCert}$.
3. *The carrier field is the indexed section presheaf carrier* $\mathsf{Sec}(i)$ *and its restricted-open or displayed base-change readbacks.*
4. *The classifier field is the section classifier relation* $\sim_i$ *and its restricted-open or displayed base-change transports.*
5. *The stability field consists of restriction identity, restriction composition, classifier transport, locality, gluing, refinement transport, and base-change transport.*
6. *The ledger field consists of the* BHist *cover nerve, cover-member rows, overlap rows,* hsame *transport, and optional route histories.*
7. *The exactness field consists of the joint exhaustion row of Definition 151.276 and the joint noninterference row of Definition 151.277.*
8. *The downstream-use field consists exactly of the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and the topology pullback face.*
9. *The boundary exports no host powerset equality, cover quotient, stalk quotient, external sheafification operation, or consumer-private root data.*
10. *No standard bridge is attached to this boundary.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootJointConsumerFace semantic name certificate`

Proof. Begin with the root consumer-threshold projection source of Definition 151.256. It already names the traditional consumer target, the single threshold source, the carrier and classifier rows, the restriction row, the three consumer projection rows, the ledger and exactness row, and the absent standard-bridge field.

The root face exactness source of Definition 151.271 refines these fields into cover-membership, restriction-route, locality-gluing-refinement, and downstream projection faces. Use Definition 151.272 for carrier, classifier, cover ledger, and restriction-route coverage, and use Definition 151.273 for locality, gluing, refinement, cover-presentation, and base-change exactness.

The projection seals identify the downstream-use field. Apply Definition 151.253, Definition 151.254, and Definition 151.255 to split the boundary into the three consumer faces.

It remains to justify exactness and exclusion. The joint exhaustion theorem Definition 151.276 supplies the listed public sheaf rows, while the joint noninterference theorem Definition 151.277 and the downstream separation theorem Definition 151.274 remove consumer-private algebraic, chart, and pullback rows from the root face. The downstream consumption exhaustion theorem Definition 151.199 then leaves no standard bridge, host powerset equality, cover quotient, stalk quotient, or external sheafification operation in the boundary. □

Definition 151.279 (Sheaf root NameCert field inventory)**.** *The root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *field inventory is the following ten-field surface.*

1. *The traditional target is a sheaf of sections over indexed opens, read over accepted indexed covers rather than host open-cover equality.*
2. *The BEDC source is* $\mathsf{Topology}^{\uparrow}$ *together with the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction surface.*
3. *The carrier row is the indexed section family* $\mathsf{Sec}(i)$ *of Definition 151.154.*
4. *The classifier row is the open-indexed section classifier family* $\sim_i$.
5. *The carrier-transport stability row keeps section carriers and classifiers aligned under the displayed indexed-open and* BHist *transport rows.*
6. *The restriction-stability row consists of restriction identity, restriction composition, and classifier preservation by restriction.*
7. *The locality row detects global classifier equality from memberwise classifier equality on an indexed cover.*
8. *The gluing row supplies a global section readback for every compatible local family.*
9. *The refinement-exactness row transports compatible families and gluing witnesses through displayed refinements and common refinements.*
10. *The ledger and presentation row consists of indexed covers, common-refinement spans, and cover-presentation invariance; it carries no standard bridge.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootNameCertFieldInventory`

Theorem 151.280 (Sheaf root NameCert carrier and classifier fields)**.** *Let* $\mathcal{T}$ *be an indexed-open carrier for* $\mathsf{Topology}^{\uparrow}$*, and let* $\mathcal{S}$ *be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over* $\mathcal{T}$*. Then the carrier, classifier, carrier-transport, and restriction-stability fields of Definition 151.279 are supplied by Definition 151.154, Definition 151.155, Definition 151.157, Definition 151.158, and Definition 151.164.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertCarrierClassifier fields`

Proof. Project the indexed section carrier from $\mathcal{S}$. By Definition 151.154, the visible carrier rows are the section families $\mathsf{Sec}(i)$, and the classifier rows are the relations $\sim_i$ with their equivalence fields.

Apply Definition 151.157. It exhausts the carrier support by indexed sections, section classifiers, restriction maps, restriction identity and composition rows, indexed covers, compatibility predicates, locality, and gluing. Thus the root inventory has no additional carrier primitive to construct.

Apply Definition 151.158 and Definition 151.164. These provide classifier transport along restrictions, identity restriction, composition of two displayed restriction routes, and stability on finite-intersection rows supplied by $\mathsf{Topology}^{\uparrow}$.

Finally read the cover-facing source through Definition 151.155. It contributes only indexed covers and compatible local-family predicates, so the projected carrier and classifier fields repack exactly as the carrier, classifier, carrier-transport, and restriction-stability fields of the root inventory. □

Theorem 151.281 (Sheaf root NameCert locality and gluing fields)**.** *For an indexed sheaf carrier over $\mathcal{T}$, the locality and gluing fields of Definition 151.279 are supplied by Definition 151.283, Definition 151.169, and Definition 151.170.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertLocalityGluing fields`

Proof. Begin with an indexed cover $\mathcal{U} = (A, \iota, h, \gamma)$ of an open index i and a compatible local family $s_a : \mathsf{Sec}(\iota(a))$. The cover, overlap, and compatibility rows are exactly the rows of Definition 151.155.

Apply Definition 151.283. It identifies locality and gluing as root $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ rows over the same indexed cover and the same classifier family $\sim_i$.

The uniqueness side is then the projection of Definition 151.169. Two global sections that display the same compatible local family are equal in the global classifier because locality detects equality from all cover members.

The readback side is supplied by Definition 151.170. Restricting a global section to the members of an indexed cover produces a compatible local family, so the gluing row and the locality row are closed under the named cover ledger. □

Theorem 151.282 (Sheaf root NameCert refinement exactness fields)**.** *For an indexed sheaf carrier over $\mathcal{T}$, the refinement, common-refinement, and cover-presentation exactness fields of Definition 151.279 are supplied by Definition 151.284, Definition 151.285, Definition 151.189, and Definition 151.293.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootNameCertRefinementExactness fields`

Proof. Start with a displayed refinement of indexed covers. Apply Definition 151.284; it transports compatible local families and gluing rows along the refinement map using only displayed inclusion routes and restriction stability.

For two cover presentations, pass to a displayed common-refinement span. Apply Definition 151.285; agreement after pullback to the common cover identifies the two global gluing classifier classes.

Apply Definition 151.189. This repacks the common-refinement comparison as invariance of the cover presentation itself, without adding host cover equality or a quotient of cover presentations.

Finally apply Definition 151.293. The route-exactness row records the same inclusion and restriction-route comparisons as $\mathsf{NameCert}$-consumable ledger data, so the refinement and presentation fields close inside the root inventory of Definition 151.279. □

Theorem 151.283 (Sheaf root cover locality and gluing obligation)**.** *Let $\mathcal{T}$ be an indexed-open carrier for* $\mathsf{Topology}^{\uparrow}$*, and let $\mathcal{S}$ be the indexed section presheaf carrier over $\mathcal{T}$ supplied by the* $\mathsf{PreSheaf}^{\uparrow}$ *restriction surface. Then the indexed sheaf carrier of Definition 151.154, with classifier family $\sim_i$ and indexed covers as in Definition 151.155, supplies the carrier, classifier, locality, and gluing rows required by the root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *ten-field template of Definition 151.190.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverLocalityGluing obligation surface`

Proof. Unfold the indexed section presheaf carrier. Its carrier row is the family $\mathsf{Sec}(i)$ over open indices $i : \mathsf{OpenIx}$, and its restriction row is the stable transport $\rho_{i,j}$ along displayed inclusions $j \leq_{\mathcal{T}} i$.

Unfold the indexed cover compatibility and locality surface. An indexed cover records member opens, cover inclusions, and finite-intersection overlap rows, so a compatible family is already expressed entirely in the $\mathsf{Topology}^{\uparrow}$ and $\mathsf{PreSheaf}^{\uparrow}$ ledgers.

The sheaf fields then provide exactly the two root rows needed by the certificate: locality detects equality of global sections from equality after restriction to every cover member, and gluing supplies a global section for every compatible local family. These are the carrier, classifier, locality, and gluing entries named in the root template row of Definition 151.190. □

Theorem 151.284 (Sheaf root restriction refinement obligation)**.** *For an indexed sheaf carrier over $\mathcal{T}$, every displayed refinement of indexed covers preserves compatible families and transports gluing rows through restriction. Hence the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *root surface may replace a cover by any displayed refinement without changing the local-family compatibility ledger or the global gluing classifier row.*

Proof. Start with a displayed refinement of indexed covers. The refinement obligation of Definition 151.174 records the member map and the inclusion rows that route each refined open into the original cover member.

Apply Definition 151.175. A compatible family on the original indexed cover pulls back along the displayed refinement to a compatible family on the refined cover, because every overlap comparison is transported by the restriction composition and classifier-transport rows.

Apply Definition 151.176. If a global section glues the original compatible family, then the same global section glues the pulled-back family after the restriction transport supplied by the refinement. Thus the root certificate can consume either cover presentation with the same global classifier class. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRestrictionRefinement obligation`

Theorem 151.285 (Sheaf root common-refinement exactness obligation)**.** *Let two indexed cover presentations of the same open carry compatible local families. If the families agree after pullback to a displayed common refinement span, then their displayed gluings determine the same global section classifier class. This exactness uses only indexed cover data, displayed common refinement, and restriction transport; it does not quotient cover presentations.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCommonRefinement exactness obligation`
↪ variant `BEDC.Derived.SheafUp.SheafRootCommonRefinement obligation carrier exactness`

Proof. Unfold the displayed common-refinement span of Definition 151.186. The span consists of one indexed cover together with two displayed refinement projections into the two cover presentations being compared.

By Definition 151.187, the span yields the two refinement maps needed to pull both local families to the common cover. The hypothesis says that these two pulled-back families are classifier-equal at every member of that common cover.

Apply Definition 151.188. The two global sections, each viewed as a gluing after descent along its refinement projection, are equal in the classifier $\sim_i$ of the base open.

Finally apply Definition 151.189. The conclusion is an equality of global section classifier classes for the two cover presentations themselves, with no host open-cover equality and no quotient of cover presentations introduced into the root surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCommonRefinement route exactness`

Definition 151.286 (Sheaf root stalk-free point-germ source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root stalk-free point-germ source is the following ten-field source over* BHist*.*

1. *The topology field is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier of Definition 88.1, with open indices, membership rows* $\mathsf{OpenAt}(i, b)$*, inclusion rows* $j \leq_{\mathcal{T}} i$*, identity inclusions, and finite-intersection rows* $i \wedge_{\mathsf{Topology}^{\uparrow}} j$*.*
2. *The presheaf field is the* $\mathsf{PreSheaf}^{\uparrow}$ *indexed section carrier of Definition 151.154, with section carriers* $\mathsf{Sec}(i)$*, restriction rows* $\rho_{i,j}$*, restriction identity, restriction composition, and restriction classifier-stability.*
3. *The sheaf field is the* $\mathsf{Sheaf}^{\uparrow}$ *locality and gluing surface over the same indexed topology and presheaf carrier.*
4. *The naming field is the root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *obligation surface, read from the carrier, classifier, locality, gluing, refinement, and exactness rows already exposed by the indexed sheaf carrier.*
5. *The history field is the point carrier* BHist *together with* hsame*-stable transport of displayed membership rows.*
6. *The point-germ ledger field is the displayed ledger of Definition 151.231; a row over* $b : \mathsf{BHist}$ *is a triple* (i, o_i, s) *with* $o_i : \mathsf{OpenAt}(i, b)$ *and* $s : \mathsf{Sec}(i)$*.*
7. *The common-neighborhood field is supplied by finite intersections in the topology carrier, including the projections from* $i \wedge_{\mathsf{Topology}^{\uparrow}} j$ *to* i *and* j *and the displayed membership witness obtained from the two input membership rows.*
8. *The restriction-comparison field compares two point-germ rows only after restricting their sections to a displayed common neighborhood and applying the section classifier* $\sim_k$ *there.*
9. *The classifier-law field uses the reflexivity, symmetry, and transitivity rows of the section classifier on each* $\mathsf{Sec}(k)$*, together with restriction identity and restriction composition, to move comparisons across displayed neighborhoods.*
10. *The boundary field exports no host point carrier, no host stalk quotient, no quotient of section representatives, and no powerset equality of opens; all point-germ evidence remains inside indexed opens,* BHist *membership, restrictions, finite intersections, section classifiers, and* hsame *transport.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootStalkFreeGermSource`

Theorem 151.287 (Sheaf root point-germ classifier reflexivity)**.** *Every displayed point-germ row in the source of Definition 151.286 compares to itself. Concretely, if* (i, o_i, s) *is a row over* $b : \mathsf{BHist}$

in the sense of Definition 151.231, then the comparison from (i, o_i, s) to (i, o_i, s) is witnessed by the common neighborhood i, the identity inclusions $i \leq_{\mathcal{T}} i$, the membership row o_i, and the identity classifier row

$$\rho_{i,i}(s) \sim_i \rho_{i,i}(s).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison reflexive`

Proof. Use the row's own indexed open as the displayed common neighborhood. The two projections required by Definition 151.231 are both the identity inclusion $i \leq_{\mathcal{T}} i$, and the membership witness is the row's own witness $o_i : \mathsf{OpenAt}(i, b)$.

It remains to fill the classifier comparison after restriction. The identity restriction field of Definition 151.154 identifies $\rho_{i,i}(s)$ with s in the section classifier on $\mathsf{Sec}(i)$, and the reflexivity field of that classifier gives $\rho_{i,i}(s) \sim_i \rho_{i,i}(s)$. This is exactly the displayed comparison row. □

Theorem 151.288 (Sheaf root point-germ classifier symmetry)**.** *The displayed point-germ comparison in the source of Definition 151.286 is symmetric. If rows (i, o_i, s) and (j, o_j, t) over the same $b : \mathsf{BHist}$ compare on a displayed common neighborhood k with projections*

$$k \leq_{\mathcal{T}} i, \qquad k \leq_{\mathcal{T}} j, \qquad o_k : \mathsf{OpenAt}(k, b),$$

and classifier row

$$\rho_{i,k}(s) \sim_k \rho_{j,k}(t),$$

then the reverse comparison from (j, o_j, t) to (i, o_i, s) is witnessed by the same k, the projections swapped, the same membership row o_k, and the symmetric classifier row

$$\rho_{j,k}(t) \sim_k \rho_{i,k}(s).$$

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison symmetric fields`

Proof. Keep the displayed common neighborhood k. The comparison definition asks only for a neighborhood below the two source opens and a membership row at b, so the same witness o_k remains valid and the two projections are read in the opposite order.

The only row whose direction changes is the section classifier comparison on $\mathsf{Sec}(k)$. Apply the symmetry field of the classifier supplied by Definition 151.154 to the given row $\rho_{i,k}(s) \sim_k \rho_{j,k}(t)$. The result is $\rho_{j,k}(t) \sim_k \rho_{i,k}(s)$, so the swapped neighborhood data form the reverse point-germ comparison. □

Theorem 151.289 (Sheaf root point-germ classifier equivalence boundary)**.** *On the displayed point-germ rows of Definition 151.286, the comparison relation of Definition 151.231 is an equivalence relation. Reflexivity is supplied by Definition 151.287, symmetry by Definition 151.288, and transitivity by Definition 151.235. The resulting equivalence boundary exports only displayed rows and comparison witnesses over indexed neighborhoods; it exports no stalk quotient, no host point carrier, and no quotient carrier of point-germ classes.*

Proof. For reflexivity, apply Definition 151.287 to the displayed row. The witness is its own indexed open, with identity restriction and the section classifier's reflexivity row.

For symmetry, apply Definition 151.288 to any displayed comparison witness. This keeps the same common neighborhood and reverses only the two projection slots and the section classifier row.

For transitivity, take two composable displayed comparison witnesses between rows over the same $b : \mathsf{BHist}$. Apply Definition 151.235; its finite-intersection construction supplies the refined common neighborhood and composes the restricted classifier rows.

These three clauses are stated entirely in the ledger language of Definition 151.231. Since Definition 151.286 contains no host stalk carrier and no quotient operation, the equivalence relation is the boundary relation available to downstream consumers, not a construction of stalk elements. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermComparison common transport`

Definition 151.290 (Sheaf root route exactness packet)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root route exactness packet is the following ten-field* BHist*-scoped surface.*

1. *The traditional target is the sheaf route from an accepted indexed cover cell to the section row consumed by downstream certificates.*
2. *The BEDC source is the joint* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{PreSheaf}^{\uparrow}$*, and* $\mathsf{NameCert}$ *surface, read over* BHist *histories and optional* Cont *route histories.*
3. *The carrier is the indexed-open carrier of* $\mathsf{Topology}^{\uparrow}$ *together with the section carrier* $\mathsf{Sec}(i)$ *of Definition 151.154.*
4. *The classifier is the open-indexed section relation* $\sim_i$ *on* $\mathsf{Sec}(i)$*, transported only along displayed restriction routes.*
5. *The stability rows are restriction identity, restriction composition, one-step classifier transport, two-step classifier transport, and route-history associativity when* Cont *rows are present.*
6. *The ledger rows are cover-member cells, pair-overlap cells, route witnesses, and* hsame *membership transport in the* BHist *cover nerve.*
7. *The exactness rows are cover-cell membership exhaustion, restriction-route classifier transport, locality and gluing, displayed refinement descent, and common-refinement comparison.*
8. *The downstream consumers are exactly the* $\mathsf{RingedSpace}^{\uparrow}$ *local-ring face, the* $\mathsf{Scheme}^{\uparrow}$ *affine-chart face, and topology pullback constructions.*
9. *The packet contains no host powerset equality of open covers, no quotient of cover presentations, no stalk quotient, and no external sheafification operation.*
10. *The status is a root* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *route-exactness surface: it records the rows available for downstream use and does not assert standard bridge closure.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootRouteExactnessPacket`

Theorem 151.291 (Sheaf route exactness membership transport)**.** *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an accepted indexed cover of* i *over the* BHist *topology ledger. For every displayed cell* $\mathsf{OpenAt}(i, b)$ *and every displayed restriction route from* i *to a cover member or pair-overlap row, the packet of Definition 151.290 transports the cover-member witness through the route and preserves the target section classifier on the reached open.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRouteExactness membership transport`

Proof. First build the cover witness. Apply Definition 151.303 to the accepted cover row. It sends the displayed cell $\mathsf{OpenAt}(i, b)$ to a member $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$ and supplies the finite-intersection row needed for comparison with any second member.

Next transport along the restriction route. Apply Definition 151.304. Its one-step and two-step clauses keep the section comparison inside the classifier of the reached member or overlap open.

The transported member witness and the transported classifier row use the same indexed open route. Since both cited rows are recorded over BHist and use hsame for membership transport, the result stays inside the cover nerve and introduces no host cover equality or cover quotient. □

Theorem 151.292 (Sheaf route exactness Cont composition)**.** *When optional* Cont *histories are present in the* $\mathsf{Sheaf}^{\uparrow}$ *route ledger, their composition rows compose displayed route witnesses only. They preserve the target open index and therefore preserve the target section classifier* $\sim_j$ *or* $\sim_k$ *already transported by restriction stability.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route history stability`

Proof. Start with the one-step route. The restriction stability obligation of Definition 151.300 gives classifier transport from $\sim_i$ to $\sim_j$ along $j \leq_{\mathcal{T}} i$.

For a composite route $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$, use the composition clause of Definition 151.300. The direct restriction from i to k and the route through j are compared in the same target classifier $\sim_k$.

Now read the optional Cont row. By Definition 151.304, such a history compares route witnesses and does not change the reached open. Hence route-history composition can only rebracket the displayed path; the classifier already attached to the reached open is unchanged. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route cont composition`

Theorem 151.293 (Sheaf route exactness NameCert consumption)**.** *The* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers receive exactly the route-exactness packet of Definition 151.290: cover-member cells, restriction classifier transport, optional* Cont *route-history composition, and the downstream projection rows sealed for those consumers. No consumer receives a larger sheaf interface from this packet.*

Proof. First identify the common downstream surface. Apply Definition 151.305. It lists the cover-member cells, pair-overlap cells, one-step and two-step restriction stability rows, optional Cont histories, and locality and gluing exactness rows available to downstream consumers.

For the ringed-space consumer, apply Definition 151.253. The local-ring face may read the sheaf section carrier, classifier, restriction stability, cover-gluing exactness, restricted-open rows, and displayed base-change rows, while local-ring structure itself is supplied outside $\mathsf{Sheaf}^{\uparrow}$.

For the scheme consumer, apply Definition 151.254. Its affine-chart read is routed through cover-presentation rows, displayed common-refinement rows, and the sealed ringed-space projection, so it receives the same route-exactness packet rather than a separate sheaf surface.

For topology pullback consumers, apply Definition 151.255. The pullback face reads pulled-back cover membership, restriction-route stability, compatible-family transport, gluing pullback, and common-refinement transport along displayed base change.

The three projections exhaust the consumers named in Definition 151.290. Since each cited seal excludes host powerset equality, cover quotients, stalk quotients, external sheafification, and standard bridge closure, downstream NameCert use is exactly the route-exactness packet and nothing larger. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootRouteExactness namecert consumption`

Definition 151.294 (Sheaf root threshold certificate source)**.** *The* $\mathsf{Sheaf}^{\uparrow}$ *root threshold certificate source is the* BHist*-indexed package consisting of the* $\mathsf{Topology}^{\uparrow}$ *topology carrier, the* $\mathsf{PreSheaf}^{\uparrow}$ *presheaf restriction carrier, the open-indexed section classifier family, the cover-nerve ledger, the point-germ ledger, the restricted-open ledger, the displayed base-change rows, and the refinement and common-refinement exactness rows. It is a source of threshold certificates only through indexed opens,* BHist *membership, section classifiers, restriction maps, displayed refinements, displayed base-change squares, restricted-open slices, and* hsame*-stable transport.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootThresholdCertificateSource`

Theorem 151.295 (Sheaf root threshold carrier and classifier obligation)**.** *The certificate source of Definition 151.294 supplies the carrier and classifier obligations of the* $\mathsf{Sheaf}^{\uparrow}$ *root solely through indexed opens,* BHist *membership, the section classifier family, restriction maps, and* hsame*-stable transport. In particular, the carrier side is exhausted by open indices, section carriers* $\mathsf{Sec}(i)$*, classifier rows* $\sim_i$*, restriction rows* $\rho_{i,j}$*, cover membership over* BHist*, and point-germ rows read on displayed neighborhoods.*

Proof. Read the carrier row from Definition 151.154. It supplies the open-indexed section carriers, the section classifiers, the restriction maps, and the identity, composition, equivalence, and classifier-transport rows attached to them.

Read the history side from Definition 151.160. Its cover-member rows are indexed by open carriers and BHist cells, and transport across history witnesses is only the displayed hsame transport already recorded in the ledger.

For pointwise readback, use Definition 151.231. A point-germ row chooses an indexed neighborhood containing the history cell and compares sections by restricting to displayed common neighborhoods, so it adds no host point, stalk quotient, or external carrier beyond the classifier and restriction rows already named. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold carrier classifier semantic certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold semantic name certificate`

Theorem 151.296 (Sheaf root threshold gluing and refinement obligation)**.** *The certificate source of Definition 151.294 assembles the exactness obligation of the* $\mathsf{Sheaf}^{\uparrow}$ *root from locality, gluing, refinement pullback, refinement gluing descent, restricted common-refinement exactness, and base-change gluing pullback. Each exactness row is expressed through indexed covers, compatible local families, displayed refinements, restricted-open slices, displayed base-change squares, section classifiers, and restriction transport.*

Proof. Begin with the root export exactness package of Definition 151.269. It already groups locality, gluing, refinement pullback, refinement gluing descent, and common-refinement presentation exactness as the stability and exactness rows exported by $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$.

The restricted-open part is read in the same package from the restricted common-refinement rows below a fixed indexed open. These rows keep the ambient section classifiers and restriction maps, and only narrow the cover ledger to the displayed slice.

For movement across a displayed source and target, apply Definition 151.208. Its cover-pullback, gluing-pullback, and common-refinement rows reuse the same exactness package after the displayed base-change square identifies the relevant source opens and pulled-back cover members. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold gluing refinement obligation`

Theorem 151.297 (Sheaf root threshold downstream coverage obligation)**.** *Every* $\mathsf{Sheaf}^{\uparrow}$ *row consumed by* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers is covered by the threshold source of Definition 151.294. Thus downstream chapters receive their sheaf-side data from the topology carrier, restriction carrier, section classifiers, cover-nerve ledger, point-germ ledger, restricted-open ledger, displayed base-change rows, and refinement or common-refinement exactness rows of that single source.*

Proof. For $\mathsf{RingedSpace}^{\uparrow}$, apply Definition 151.197. Its sheaf-side requests are precisely the indexed section carriers, classifiers, restriction stability, cover-gluing exactness, restricted-open surface, and displayed base-change surface listed in the threshold source.

For $\mathsf{Scheme}^{\uparrow}$, apply Definition 151.198. Each affine or ringed-space chart first consumes the ringed-space sheaf boundary, and overlap compatibility is discharged by the cover-presentation and common-refinement exactness rows already included in the source.

For topology pullback consumers, apply Definition 151.226. The pulled-back cover rows are displayed base-change rows over the same indexed-open and BHist ledgers, so they are covered by the threshold source without adding a host powerset cover comparison or external sheaf model. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootThreshold downstream coverage obligation`

151.14 Root obligations for downstream certificates

Definition 151.298 (SheafUp root-unblock BEDC source)**.** *The root-unblock source for* $\mathsf{Sheaf}^{\uparrow}$ *is the ten-field certificate surface consumed by ringed-space, scheme, and restricted-open sheaf constructions.*

1. *The traditional target is sheaves of sections over indexed opens.*
2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* $\mathsf{NameCert}$.
3. *The history ledger is the* BHist *cover-nerve ledger of Definition 151.160.*
4. *The carrier is the indexed section presheaf carrier of Definition 151.154, equipped with the locality and gluing fields of Definition 151.155.*
5. *The classifier family is the open-indexed relation* $\sim_i$ *on* $\mathsf{Sec}(i)$.
6. *The cover ledger records cover indices* A, *cover-member maps* $\iota : A \to \mathsf{OpenIx}$, *and pointwise cover rows over* BHist.
7. *The overlap ledger records finite intersections* $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(a')$ *and their projection rows.*
8. *The restriction ledger records routes from a target open to cover members and overlaps, with identity, composition, and classifier-stability rows.*
9. *Optional route histories are recorded by* Cont *rows, and history-cell membership is transported only by* hsame.
10. *The surface contains no quotient of cover presentations and no equality of host powersets of opens.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource`

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource member point germ`

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockBEDCSource cover membership`

Theorem 151.299 (Sheaf root-unblock cover membership obligation)**.** *Let $\mathcal{U} = (A, \iota, h, \gamma)$ be an accepted cover row of an open index i over a* BHist *topology ledger. Then every displayed history membership in the covered open has a displayed cover-member witness:*

$$\forall b : \mathsf{BHist},\ \mathsf{OpenAt}(i, b) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), b).$$

All overlap information used downstream is recorded only through finite-intersection rows and hsame*-stable* BHist *membership, not through host powerset equality.*

Proof. Unfold the accepted cover row. Its pointwise cover component is exactly the displayed implication from $\mathsf{OpenAt}(i, b)$ to a cover index $a : A$ with $\mathsf{OpenAt}(\iota(a), b)$.

For overlap rows, use the finite-intersection field of the indexed topology carrier as in Definition 151.161. It supplies the overlap index and the two projection rows back to the participating cover members.

History transport is then read from the BHist cover-nerve ledger of Definition 151.160. That ledger admits only hsame-stable membership transport, so the exported cover-membership row stays internal to indexed opens and history cells. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockCoverMembership obligation`

Theorem 151.300 (Sheaf root-unblock restriction stability obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$. Restriction identity, restriction composition, stable restriction rows, and* Cont *associativity for route histories preserve the target section classifier along one-step and two-step restriction routes. Concretely, for $j \leq_{\mathcal{T}} i$ and $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$, the rows*

$$\rho_{i,i}(s) \sim_i s, \qquad s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t), \qquad \rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

are the classifier-preserving route rows exposed to downstream certificates.

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblockRestrictionStability semantic name certificate`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock restriction stability rows`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistPointGermLedger route history target classifier stability`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock restriction stability surface`

Proof. The identity and composition rows are fields of the indexed section presheaf carrier in Definition 151.154. The one-step classifier preservation row is the stable restriction row of Definition 151.156, also cited as Definition 151.172.

For a two-step route, first use restriction classifier transport along $j \leq_{\mathcal{T}} i$, then compose with the route from j to k. The displayed composition row identifies this two-step restriction with the direct restriction from i to k in the classifier $\sim_k$.

If the ledger also carries Cont histories for the same routes, their associativity only compares displayed route histories. It does not alter the open index k or the classifier $\sim_k$, so it preserves the same target section classifier consumed by the certificate. □

Theorem 151.301 (Sheaf root-unblock locality and gluing obligation)**.** *Let $\mathcal{S}$ be an indexed sheaf carrier for* $\mathsf{Sheaf}^{\uparrow}$ *over an indexed open topology carrier $\mathcal{T}$, and let $\mathcal{U} = (A, \iota, h, \gamma)$ be an indexed cover of i. Compatible local families with displayed gluings have a* NameCert*-consumable exactness row: locality gives global classifier equality for any two global readbacks whose restrictions agree on the cover, and gluing supplies existence of a global readback without quotienting cover presentations.*

Proof. The existence half is the gluing field in Definition 151.155. Given a compatible local family, it supplies a section $g : \mathsf{Sec}(i)$ together with displayed restriction rows to each $\iota(a)$.

The uniqueness half is locality. If two global readbacks restrict to classifier-equal sections on every cover member, the locality field returns equality in the global classifier $\sim_i$; Definition 151.169 packages the same argument for two displayed gluings of one compatible family.

Refinement and presentation changes are handled by the existing refinement rows Definition 151.176, Definition 151.179, Definition 151.188, and Definition 151.189. Each row uses displayed cover maps, common refinements, restriction stability, Cont route histories when present, and hsame-stable membership; none introduces a quotient of cover presentations. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootUnblock locality gluing faces`
Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootFaceRead locality gluing refinement inversion`

Definition 151.302 (Sheaf root cover-nerve export surface)**.** *The root cover-nerve export surface for* $\mathsf{Sheaf}^{\uparrow}$ *is the three-row* BHist*-scoped certificate surface consumed by downstream chapters.*

1. *The traditional target is sheaves over indexed opens, read through section carriers on a cover nerve rather than through host open covers.*
2. *The BEDC source chapters are* $\mathsf{Topology}^{\uparrow}$, $\mathsf{PreSheaf}^{\uparrow}$, *and* NameCert.
3. *The carrier is the family of indexed sections over* BHist *opens, using* $\mathsf{Sec}(i)$ *from Definition 151.154 for each open index* i.
4. *The classifier family is the displayed section relation* $\sim_i$ *on* $\mathsf{Sec}(i)$.
5. *The stability rows are restriction identity, restriction composition, classifier transport, and optional* Cont *route-history transport.*
6. *The ledger rows are cover-member cells, pair-overlap cells, and route-history cells in the* BHist *cover nerve of Definition 151.160.*
7. *The exactness rows are cover membership, restriction route stability, and locality/gluing.*
8. *The cover-membership row sends every accepted* $\mathsf{OpenAt}(i, b)$ *cell to a displayed cover member and its pair-overlap ledger context.*
9. *The downstream export is exactly the rows listed here:* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback consumers receive no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*
10. *There is no standard bridge in this surface.*

Lean4 (definition) `BEDC.Derived.SheafUp.SheafRootCoverNerveExportSurface`

Theorem 151.303 (Sheaf root cover-nerve membership exhaustion)**.** *Let* $\mathcal{U} = (A, \iota, h, \gamma)$ *be an accepted cover row of an indexed open* i *over the* BHist *topology ledger. Every displayed cell* $\mathsf{OpenAt}(i, b)$ *in this accepted cover has a cover-member witness and a pair-overlap row inside the* BHist *cover nerve:*

$$\forall b : \mathsf{BHist},\ \mathsf{OpenAt}(i, b) \Longrightarrow \exists a : A,\ \mathsf{OpenAt}(\iota(a), b)$$

and for every second member $a' : A$ *the ledger also contains the finite-intersection row for* $\iota(a) \wedge_{\mathsf{Topology}^{\uparrow}} \iota(a')$ *with its two projections to the participating members.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve membership exhaustion`

Proof. Apply the cover-membership obligation of Definition 151.299. It reads the accepted cover row as a BHist-indexed implication from an ambient displayed cell to a cover-member witness, so every $\mathsf{OpenAt}(i, b)$ cell is exhausted by a displayed member $\iota(a)$.

The pair-overlap row is then the overlap projection surface of Definition 151.161, unpacked inside the cover-nerve ledger of Definition 151.160. For each pair $a, a' : A$, the finite-intersection row supplies the overlap index and the two projections back to $\iota(a)$ and $\iota(a')$.

Because the ledger transports membership only by hsame, this exhaustion stays scoped to BHist cells and indexed opens. No host cover equality or quotient of cover presentations is introduced. □

Theorem 151.304 (Sheaf root cover-nerve restriction route stability)**.** *Let $\mathcal{S}$ be the indexed section carrier of Definition 151.154 over an indexed open topology carrier. One-step and two-step restrictions preserve section classifiers:*

$$s \sim_i t \Longrightarrow \rho_{i,j}(s) \sim_j \rho_{i,j}(t), \qquad \rho_{j,k}(\rho_{i,j}(s)) \sim_k \rho_{i,k}(s)$$

for every displayed chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. Optional Cont *route histories compare route witnesses only and do not alter the target classifier $\sim_j$ or $\sim_k$.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafBHistCoverNerveLedger route hsame transport`

Proof. The first displayed row is the restriction classifier-transport field of Definition 151.154. It is also the one-step stability clause isolated by Definition 151.300.

For the two-step row, apply the composition field of the same carrier to the chain $k \leq_{\mathcal{T}} j \leq_{\mathcal{T}} i$. The comparison lands in $\sim_k$, so the direct restriction and the composed restriction have the same target classifier.

If a Cont history is present, use the route-history part of Definition 151.300. Those histories record associativity of displayed routes; they do not change the open index of the restricted section, hence cannot change the classifier family consumed downstream. □

Theorem 151.305 (Sheaf root cover-nerve downstream coverage)**.** *The downstream consumers* $\mathsf{RingedSpace}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, *and topology pullback constructions receive exactly the rows of Definition 151.302: cover-member cells, pair-overlap cells, one-step and two-step restriction stability rows, optional* Cont *route histories, and locality/gluing exactness rows. They receive no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Proof. The carrier part is fixed by Definition 151.154: downstream chapters can read only indexed section carriers, section classifiers, restrictions, and the identity, composition, and classifier-transport rows attached to them.

The cover part is fixed by Definition 151.160 and Definition 151.303. These provide cover-member cells and pair-overlap cells over BHist, with overlap projections supplied by Definition 151.161.

The exactness part is the union of the three root obligations. Cover membership comes from Definition 151.299; restriction route stability comes from Definition 151.300; locality and gluing come from Definition 151.301.

Those cited rows are precisely the rows listed in Definition 151.302. Since none of them introduces host cover equality, a cover quotient, a stalk quotient, or an external sheafification operation, no downstream consumer obtains such a row from this export surface. □

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve downstream coverage`

Theorem 151.306 (Sheaf root cover-nerve exact globalization package)**.** *The root cover-nerve export surface of Definition 151.302 forms an* ExactGlobalizeBase*-style package over the indexed-open* BHist *cover nerve. Its coverage face is Definition 151.303; its base soundness face is Definition 151.304; its base completeness and exactness face is the locality-gluing row of Definition 151.301. Hence* $\mathsf{RingedSpace}^{\uparrow}$ *and* $\mathsf{Scheme}^{\uparrow}$ *may consume one classifier-exact cover-nerve package, with no host cover equality, cover quotient, stalk quotient, or external sheafification operation.*

Lean4 ✓ `BEDC.Derived.SheafUp.SheafRootCoverNerve exact globalize package`

Proof. Unpack Definition 151.302. The carrier is the indexed family $\mathsf{Sec}(i)$ over BHist opens, the classifier is the displayed relation $\sim_i$, and the ledger is the cover nerve of Definition 151.160.

The coverage face is exactly Definition 151.303. It sends each displayed $\mathsf{OpenAt}(i, b)$ cell to a cover member and records the pair-overlap rows needed by downstream restriction routes.

The base soundness face is Definition 151.304. One-step restriction, two-step restriction, classifier transport, and optional Cont route histories preserve the section classifier on the target indexed open.

The base completeness and exactness face is Definition 151.301. Gluing constructs the global readback from compatible local rows, and locality identifies any two global readbacks whose restrictions agree on the accepted cover.

These are precisely the coverage, soundness, and completeness fields required by the exact-globalization constructor of Definition 9199.16, with the relation field constrained as in Definition 9199.11. Repacking the three displayed faces therefore gives the single classifier-exact cover-nerve package consumed by the downstream chapters, and Definition 151.305 fixes that no quotient or external sheafification row is exported. □

Chapter 152

A Concrete Naming Certificate for $\mathsf{RingedSpace}^{\uparrow}$

The ringed-space interface $\mathsf{RingedSpace}^{\uparrow}$ packages a topological carrier with a sheaf of commutative rings on it, certifying that stalks form local rings. It depends on $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ (chapter 151) and $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44).

152.1 Carrier definition

Theorem 152.1 (RingedSpace carrier and stalk obligations)**.** *A* $\mathsf{RingedSpace}^{\uparrow}$ *certificate must expose a* $\mathsf{Topology}^{\uparrow}$ *carrier over* BHist*, a* $\mathsf{Sheaf}^{\uparrow}$ *row of section carriers on that topology, a* $\mathsf{CommRing}^{\uparrow}$ *fibre row on every displayed open, and a* $\mathsf{LocalRingUp}$ *stalk row at every displayed point history. The carrier ledger consists exactly of the base-open index, the membership row, the section carrier, the restriction route, the commutative-ring operation tuple on each fibre, and the local-ring witness obtained from the sheaf germ classifier at the stalk.*

Proof. The named carrier is the displayed topological carrier supplied by $\mathsf{Topology}^{\uparrow}$ together with the open-membership row that decides which histories lie in each displayed open. Over that carrier, the $\mathsf{Sheaf}^{\uparrow}$ row supplies section carriers and restriction routes for displayed inclusions of opens.

For each displayed open, the fibre is the corresponding section carrier equipped with the $\mathsf{CommRing}^{\uparrow}$ operation row. The zero, one, addition, multiplication, negation, and unit rows are therefore indexed by the same open ledger as the sheaf restriction maps.

At a displayed point history, the stalk row is read from the $\mathsf{Sheaf}^{\uparrow}$ germ classifier over displayed neighbourhoods. The local-ring witness is then a $\mathsf{LocalRingUp}$ row on that germ carrier, with its operations induced from the $\mathsf{CommRing}^{\uparrow}$ fibre rows through restriction.

Thus the ringed-space carrier obligation is not a separate host object. It is the joint BEDC row formed by $\mathsf{Topology}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, and $\mathsf{LocalRingUp}$ stalk data, all indexed by BHist and by the displayed open ledger. □

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface carrier classifier rows`

Theorem 152.2 (RingedSpace classifier transport obligations)**.** *For a* $\mathsf{RingedSpace}^{\uparrow}$ *certificate,* hsame *transport, the* $\mathsf{Topology}^{\uparrow}$ *base-open classifier, and the* $\mathsf{Sheaf}^{\uparrow}$ *section classifier must transport section rows, restriction witnesses,* $\mathsf{CommRing}^{\uparrow}$ *operation endpoints, germ rows, and* $\mathsf{LocalRingUp}$ *stalk-locality witnesses. In particular, if two base presentations are related by* hsame *and their*

opens are matched by the topology classifier, then section comparison and stalk-locality are decided only by the transported sheaf and ring classifiers.

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton sheaf commring stalk locality obligation`

Proof. Begin with a displayed hsame witness between base histories. The $\mathsf{Topology}^{\uparrow}$ classifier transports open names, membership rows, and inclusion rows across that witness, so the source and target presentations have the same displayed neighbourhood ledger.

The $\mathsf{Sheaf}^{\uparrow}$ classifier then transports sections along the matched open rows. Because restriction witnesses are part of the sheaf ledger, the transported section row has the same restriction comparison as the source row after reindexing by the topology classifier.

Next transport the $\mathsf{CommRing}^{\uparrow}$ endpoints on fibres. Operation compatibility says that zero, one, addition, multiplication, negation, and unit detection commute with the transported section comparison, so fibre operations remain section rows after the same classifier pass.

Finally, stalk-locality is transported through the sheaf germ classifier. A local-ring witness at a point history is accepted on the target side only when it is represented by a common displayed neighbourhood, transported restrictions, and the transported $\mathsf{CommRing}^{\uparrow}$ endpoint classifier. □

Theorem 152.3 (RingedSpace exact ledger obligations)**.** *A* $\mathsf{RingedSpace}^{\uparrow}$ *certificate must state coverage, soundness, completeness, and exact ledger rows for base opens, restriction maps, commutative-ring operations, and local-stalk witnesses. Coverage says every named ringed-space read is represented by those rows. Soundness says every accepted row is stable under the displayed classifiers. Completeness says every classifier-stable base-open, section, operation, and stalk-locality witness appears in the ledger. Exactness says no host open equality, section quotient, stalk quotient, or standard bridge witness is silently added.*

Proof. Coverage is obtained by projecting a displayed ringed-space read to its four named rows: the $\mathsf{Topology}^{\uparrow}$ base-open row, the $\mathsf{Sheaf}^{\uparrow}$ restriction row, the $\mathsf{CommRing}^{\uparrow}$ operation row, and the LocalRingUp stalk-locality row. Each projection is part of the certificate surface described in Definition 152.1.

Soundness follows by applying the transport obligations of Definition 152.2. A row accepted by the ledger remains accepted after base-open transport, section transport, operation-endpoint transport, and germ transport.

Completeness is the converse reading of the same rows. If the base-open, restriction, operation, and stalk-locality witnesses are all stable under the displayed classifiers, the certificate repacks them as a $\mathsf{RingedSpace}^{\uparrow}$ row without asking for any additional carrier.

Exactness is the boundary condition. The ledger contains precisely the base-open, restriction-map, commutative-ring operation, and local-stalk rows, so any later $\mathsf{Scheme}^{\uparrow}$ or variety certificate citing $\mathsf{RingedSpace}^{\uparrow}$ as a source receives only those obligations and not a hidden standard bridge. □

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton cover nerve empty boundary`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface restriction exact ledger`

Theorem 152.4 (RingedSpaceUp carrier and sheaf obligation)**.** *A* $\mathsf{RingedSpace}^{\uparrow}$ NameCert *is sourced by a* BHist *topological carrier together with a* $\mathsf{Sheaf}^{\uparrow}$ *row assigning to each carried open witness a* $\mathsf{CommRing}^{\uparrow}$ *section carrier. Its classifier is inherited only from* hsame*, the* $\mathsf{Topology}^{\uparrow}$ *open classifier, the* $\mathsf{Sheaf}^{\uparrow}$ *restriction classifier, and the* $\mathsf{CommRing}^{\uparrow}$ *classifier. The carrier ledger records the open witness, membership row, section carrier, restriction row, and commutative-ring endpoint surface for each displayed open.*

Proof. Unpack the ten-field NameCert template. The source row supplies the BHist topological carrier, so every open witness is read through the carried Topology$^{\uparrow}$ classifier and transported only by hsame.

Read the Sheaf$^{\uparrow}$ NameCert over the indexed opens. For each displayed open, its section row gives the section carrier and the restriction classifier visible along displayed inclusions.

Finally restrict each section value to the CommRing$^{\uparrow}$ certificate surface. The result is the displayed carrier and classifier row needed by RingedSpace$^{\uparrow}$, with no extra host algebra or standard bridge data. □

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton carrier sheaf obligation`

Theorem 152.5 (RingedSpaceUp stalk-locality obligation)**.** *For every carried point history and every admissible neighbourhood-filter row of a* RingedSpace$^{\uparrow}$ *certificate, the stalk projection is a local* CommRing$^{\uparrow}$ *row. Restriction-compatible germs have a unique maximal non-unit classifier branch, and germ comparison is stable under* hsame *transport of the point and under* Sheaf$^{\uparrow}$ *refinement. Local-ring status is recorded as an obligation row of the displayed certificate rather than as an external model.*

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonPackage restricted stalk locality obligation`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface stalk locality readback`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton stalk restriction commring stability`

Proof. Use the Sheaf$^{\uparrow}$ restriction and refinement exactness rows to compare two representatives of a germ on a common displayed neighbourhood. The common row supplies the section classifier needed to identify the restricted section endpoints.

Apply the CommRing$^{\uparrow}$ carrier and classifier rows to those section endpoints. Unit detection, non-unit comparison, and maximal-branch uniqueness are then read from the displayed commutative-ring row attached to the germ endpoint.

Transport stability follows because the point history is moved by hsame and the neighbourhood row is refined through Sheaf$^{\uparrow}$. Repacking these witnesses gives the stalk-local CommRing$^{\uparrow}$ obligation row without introducing a host stalk quotient. □

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonStalk semantic name certificate`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton stalk locality restriction obligation`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface restricted stalk locality`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonPackage stalk locality transport`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface stalk locality common neighborhood`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton stalk locality obligation`

Theorem 152.6 (RingedSpaceUp stability, exactness, and ledger obligation)**.** *Carrier transport, restriction compatibility, stalk transport, and exactness for visible* RingedSpace$^{\uparrow}$ *packages are exactly the conjunction of* Topology$^{\uparrow}$ *carrier transport,* Sheaf$^{\uparrow}$ *restriction and gluing classifier transport, and* CommRing$^{\uparrow}$ *algebraic classifier transport. The ledger records the open witness, restriction route, section endpoint, and stalk-germ provenance, and it records no* Scheme$^{\uparrow}$*, affine-cover, spectrum, or standard bridge data.*

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface stability ledger rows`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface stability ledger obligation`
Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingleton stability ledger obligation`

Proof. Compose the three named certificate surfaces through the NameCert stability field. The $\mathsf{Topology}^{\uparrow}$ surface transports open witnesses, the $\mathsf{Sheaf}^{\uparrow}$ surface transports restrictions and gluing classifiers, and the $\mathsf{CommRing}^{\uparrow}$ surface transports algebraic endpoints.

Project the ledger coordinates after this composition. The open witness comes from the topology row, the restriction route and section endpoint come from the sheaf row, and the stalk-germ provenance is obtained by the stalk-locality row of Definition 152.5.

Exactness is therefore the conjunction of those three transport surfaces as seen by the visible package. Since every ledger coordinate has already been projected, no scheme, affine-cover, spectrum, or standard bridge row is present in the $\mathsf{RingedSpace}^{\uparrow}$ certificate. □

Lean4 ✓ `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonSurface restriction stability ledger`

152.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{RingedSpace}^{\uparrow}}$.

Constructive story: A ringed space is assembled by taking a $\mathsf{Topology}^{\uparrow}$ carrier over BHist, placing a $\mathsf{Sheaf}^{\uparrow}$ row of section carriers on its displayed opens, reading each fibre through $\mathsf{CommRing}^{\uparrow}$, and then taking the stalk at a displayed point through the sheaf germ comparison until the local-ring row is visible. The carrier accepts only packets with that open-membership row, section row, restriction route, commutative-ring operation tuple, germ provenance, and local-stalk witness; host open equality, a stalk quotient, a spectrum row, and any $\mathsf{Scheme}^{\uparrow}$ data stay outside the carrier. Two accepted packets classify together when hsame transports the base histories and the upstream topology, sheaf, commutative-ring, and stalk-locality classifiers carry the same opens, sections, restrictions, operations, and germs across the comparison. The source specification is precisely this topology-sheaf-ring-stalk packet, the pattern records the base-open membership, restriction compatibility, fibre operations, and local germ behaviour, and the ledger keeps the raw open witness, section endpoint, restriction route, and stalk provenance visible in the same order in which the packet was built. The five naming-certificate obligations are discharged because the singleton surface supplies an admitted packet, the component classifiers are reflexive, symmetric, and transitive, and every accepted row is transported back into the carrier by the same displayed hsame and upstream classifier rows. A history h lies in the signature gap of a $\mathsf{RingedSpace}^{\uparrow}$ token p only when the source domain accepts that whole packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; for this chapter that condition contracts to saying that the topology, sheaf, fibre-ring, and local-stalk rows of h are exactly the transported rows named by p. The chapter registers this row-by-row stalk construction under the name $\mathsf{RingedSpace}^{\uparrow}$; it exports the surface needed by $\mathsf{Scheme}^{\uparrow}$ without adding a spectrum bridge, host stalk quotient, or independent locally ringed space primitive.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RingedSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scope is the existing $\mathsf{RingedSpace}^{\uparrow}$ carrier, classifier transport, exact ledger, stalk-locality, and stability-ledger theorem surface exported for $\mathsf{Scheme}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RingedSpace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RingedSpaceUp.RingedSpaceSingletonStalk\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not close $\mathsf{Scheme}^{\uparrow}$, affine or projective variety closure, the spectrum standard bridge, or bridge verification.

Upgrade path: No further theory-axis upgrade is pending; bridge verification remains a formal-axis and bridge-status matter.

Chapter 153

A Concrete Naming Certificate for $\mathsf{Scheme}^{\uparrow}$

The scheme interface $\mathsf{Scheme}^{\uparrow}$ packages a locally ringed space that admits an open cover by spectra of commutative rings. It depends on $\mathsf{NameCert}_{\mathsf{RingedSpace}^{\uparrow}}$ (chapter 152) and $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44), and is the substrate for affine and projective variety certificates.

153.1 Carrier definition

Theorem 153.1 (Scheme carrier obligation)**.** *Every* $\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$ *certificate must supply a carrier* $C_{\mathsf{Scheme}^{\uparrow}} : \mathsf{BHist} \to \mathsf{Prop}$ *whose witnesses consist only of a* $\mathsf{RingedSpace}^{\uparrow}$ *source row, certified* $\mathsf{CommRing}^{\uparrow}$ *affine-chart rows, and the affine-cover ledger. The carrier has no host-level scheme object, host spectrum object, host topological-space primitive, host open-set equality, or external sheafification row.*

Proof. Unpack the ten-field NameCert template. Its source field supplies the locally ringed-space row already named by chapter 152, so the base carrier is read as a BEDC history predicate over BHist.

The chapter source sentence names exactly the remaining inputs: certified $\mathsf{CommRing}^{\uparrow}$ affine chart rows and the cover ledger. Each chart is therefore recorded as a certificate row attached to a cover index, not as a host spectrum primitive.

Since the ten-field template has no extra carrier slot after source, classifier, stability, exactness, and ledger are fixed, there is no place for a hidden scheme object. The carrier obligation is precisely the displayed BHist predicate generated by those source and ledger rows. □

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeSingletonPackage carrier source obligation`

Theorem 153.2 (Scheme classifier obligation)**.** *The* $\mathsf{Scheme}^{\uparrow}$ *classifier must be a BEDC classifier over* $C_{\mathsf{Scheme}^{\uparrow}}$*, stable under* hsame*, and determined by the* $\mathsf{RingedSpace}^{\uparrow}$ *classifier together with affine-chart classifier agreement from the certified* $\mathsf{CommRing}^{\uparrow}$ *chart rows. Two scheme carrier witnesses are identified only when their ringed-space rows match and their affine chart rows agree after the displayed cover-refinement and restriction transports.*

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverLedger classifier refinement transport`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverClassifier overlap locality`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeSingleton affine cover classifier locality obligation`

Proof. Use the classifier-choice field of the per-object roadmap template to choose the displayed relation on the carrier from the cited source classifiers. The $\mathsf{RingedSpace}^{\uparrow}$ row supplies the point, open, section, restriction, and local-ring comparisons visible in the locally-ringed-space source.

For each affine chart, read the endpoint comparison from the $\mathsf{CommRing}^{\uparrow}$ certificate row. On overlaps, transport the two chart rows along the cover-refinement and restriction routes already present in the carrier ledger, then compare the transported chart classifiers.

The stability-obligation field of the same template requires this relation to respect hsame. That stability follows by composing the $\mathsf{RingedSpace}^{\uparrow}$ transport with the chart-level $\mathsf{CommRing}^{\uparrow}$ transports, so the scheme classifier is determined by those two classifier surfaces and no additional equality. □

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineChartLedger overlap locality obligation`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverLedger overlap classifier locality`

Theorem 153.3 (Scheme affine-cover obligation)**.** *The affine-spectrum cover row of a* $\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$ *certificate must provide coverage, restriction compatibility, locality, and gluing data as* Pkg *and ledger information. These rows are exactly the cover, exactness, ledger, and consumer-boundary data consumable by* $\mathsf{AffineVar}^{\uparrow}$ *and* $\mathsf{ProjectiveVar}^{\uparrow}$*, with no additional host spectrum, quotient-cover, or external variety row.*

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverLedger restricted global soundness`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverLedger exactness obligation`

Proof. Read the affine-spectrum cover row already named in the seed scope as a finite Pkg-indexed ledger. Coverage says that every accepted carrier witness is assigned to one of those cover packages together with its $\mathsf{RingedSpace}^{\uparrow}$ source row and certified $\mathsf{CommRing}^{\uparrow}$ chart row.

Restriction compatibility and locality are the overlap clauses of the same ledger. They record the displayed restriction route, the common overlap witness, and the chart-classifier agreement after transport through the ringed-space and commutative-ring certificate surfaces.

Gluing is then a ledger exactness statement: compatible local chart rows repack into the same $\mathsf{Scheme}^{\uparrow}$ carrier witness, and every consumer readback for $\mathsf{AffineVar}^{\uparrow}$ or $\mathsf{ProjectiveVar}^{\uparrow}$ must pass through those listed packages. Thus the cover, exactness, ledger, and consumer rows named in the chapter are the whole affine-cover obligation surface. □

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeSingleton affine cover ledger exactness`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeSingleton affine cover nerve empty boundary`
Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverLedger restriction exactness`

153.2 The certificate

Theorem 153.4 (Scheme public NameCert surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$ *export is the SemanticNameCert surface whose source is the* $\mathsf{RingedSpace}^{\uparrow}$ *row together with certified* $\mathsf{CommRing}^{\uparrow}$ *affine-chart rows and the affine-cover ledger. Its classifier is the ringed-space classifier together with chart agreement under restriction and refinement. Its exactness and consumer boundary are exactly the carrier, classifier, and affine-cover obligations proved in Definition 153.1, Definition 153.2, and Definition 153.3.*

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeAffineCoverRow semantic name certificate`

Theorem 153.5 (Scheme public standard bridge)**.** *The public* $\mathsf{Scheme}^{\uparrow}$ *surface of Definition 153.4 admits a standard bridge to the displayed affine-cover presentation. Forward interpretation reads BEDC* $\mathsf{RingedSpace}^{\uparrow}$ *rows as locally ringed-space rows, certified* $\mathsf{CommRing}^{\uparrow}$ *chart rows as affine chart rings, and the affine-cover ledger as the standard affine cover. Conversely, any displayed standard affine-cover row whose point, restriction, chart, and overlap data are presented by the same BEDC carriers repacks as the public* $\mathsf{Scheme}^{\uparrow}$ *surface. The bridge adds no host spectrum object, quotient cover, or external sheafification row.*

Proof. Apply Definition 153.1. It fixes the bridge domain as a BHist carrier assembled from the $\mathsf{RingedSpace}^{\uparrow}$ source row, the certified $\mathsf{CommRing}^{\uparrow}$ chart rows, and the affine-cover ledger, so the forward interpretation has no additional scheme carrier to read.

Apply Definition 153.2. The locally ringed-space comparisons and the affine chart-ring comparisons are therefore exactly the displayed classifier transports across restrictions, refinements, and overlaps.

Apply Definition 153.3. Coverage, restriction compatibility, locality, and gluing are the ledger rows that the standard affine cover reads, and the same theorem excludes any host spectrum object, quotient cover, or external sheafification row from the public surface.

For the reverse direction, start with a displayed standard affine-cover row whose point, restriction, chart, and overlap data are already carried by those BEDC rows. Repack its locally ringed-space source, chart-ring certificates, cover ledger, classifier transports, and exactness boundary through Definition 153.4. The resulting package is precisely the public $\mathsf{Scheme}^{\uparrow}$ surface, with no extra row beyond the carriers and ledgers already listed. □

Lean4 ✓ `BEDC.Derived.SchemeUp.SchemeUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$.

Constructive story: A scheme is generated by starting with the $\mathsf{RingedSpace}^{\uparrow}$ row already carried over BHist, attaching certified $\mathsf{CommRing}^{\uparrow}$ affine-chart rows, and recording a finite affine-cover ledger whose restriction and overlap routes are displayed as Cont and Pkg data. The carrier accepts only histories containing that locally ringed-space source, the chart-ring rows, and the cover package; there is no host scheme object, host spectrum object, quotient cover, open-set equality, or external sheafification row inside the carrier. Two accepted presentations classify together when the ringed-space classifier matches the underlying point, open, section, restriction, and local-ring rows, and the affine chart classifiers agree after the displayed refinement and overlap transports. The source specification is the ringed-space row plus the certified affine charts and their cover ledger, the pattern records coverage, restriction compatibility, locality, and gluing, and the ledger keeps the raw cover packages, restriction routes, overlap witnesses, and chart-ring classifiers visible to later variety consumers. The five naming-certificate obligations hold because a displayed affine-cover row inhabits the carrier, the inherited ringed-space and chart classifiers provide reflexivity, symmetry, and transitivity, and the cover ledger repacks every classifier-stable transported row back into the same carrier predicate. A history h lies in the signature gap of a $\mathsf{Scheme}^{\uparrow}$ token p when the source domain accepts this affine-cover packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names the answer as p; unfolded, the locally ringed-space row and every chart and overlap row of h must match the cover presentation encoded by p. The standard bridge reads the same BEDC rows as a displayed affine-cover presentation and reads any such presentation back by repacking those rows, so the chapter registers this cover-ledger construction under $\mathsf{Scheme}^{\uparrow}$ without adding a hidden spectrum carrier or external scheme equality.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface is Definition 153.4, and the standard bridge is Definition 153.5, binding the carrier predicate to BHist, transporting it by hsame, recording Cont restriction rows and Pkg affine-cover ledger rows, and fixing the $\mathsf{RingedSpace}^{\uparrow}$ source rows, $\mathsf{CommRing}^{\uparrow}$ chart classifier rows, and the $\mathsf{AffineVar}^{\uparrow}$/$\mathsf{ProjectiveVar}^{\uparrow}$ consumer boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SchemeUp.SchemeAffineCoverRow\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close full affine or projective variety certificates or bridgeChecked verification.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 154

A Concrete Naming Certificate for $\mathsf{AffineVar}^{\uparrow}$

The affine-variety interface $\mathsf{AffineVar}^{\uparrow}$ packages the zero-locus of a finite system of polynomial equations in the affine n-space of a commutative-ring carrier. It depends on $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

154.1 Carrier definition

154.2 The certificate

Definition 154.1 (Affine finite-family zero-locus)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ *with scalar classifier* $\sim_R$ *and scalar zero* 0_R*. Fix a dimension* n*, an affine point carrier* $\mathsf{AffPoint}_{R,n} : \mathsf{BHist} \to \mathsf{Prop}$*, and a polynomial evaluation row* $\mathsf{Eval}_{R,n}(p, x, y)$ *for polynomial-code histories over that affine* n*-space. Let* $\mathsf{PolyCode}_{R,n}(p)$ *denote the polynomial-code carrier read from the normalized finite coefficient-sequence substrate of Definition 50.3.*

For a polynomial code p *and point history* x*, define the zero-evaluation predicate by*

$$\mathsf{PolyEvalZero}_{R,n}(p, x) \iff \mathsf{PolyCode}_{R,n}(p) \wedge \exists y : \mathsf{BHist}.\ \mathsf{Eval}_{R,n}(p, x, y) \wedge y \sim_R 0_R.$$

For a finite BEDC-history spine $F : \mathsf{HistSpine}$ *of polynomial codes, define the affine finite-family zero-locus carrier by*

$$\mathsf{V}_{R,n}(F)(x) \iff \mathsf{AffPoint}_{R,n}(x) \wedge \forall i : \mathsf{Pos}(F).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x).$$

This is the finite-equation carrier used by the concrete $\mathsf{AffineVar}^{\uparrow}$ *presentation: the point row is shared by the ambient affine space, while the equation row is read from the finite polynomial spine.*

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineVarTasteGate single carrier alignment`

Theorem 154.2 (Affine zero-locus intersection by equation concatenation)**.** *In the setup of Definition 154.1, let* $F, G : \mathsf{HistSpine}$ *be finite polynomial-family spines over the same affine* n*-space. For every point history* $x : \mathsf{BHist}$*,*

$$\mathsf{V}_{R,n}(F{+}{+}G)(x) \iff \mathsf{V}_{R,n}(F)(x) \wedge \mathsf{V}_{R,n}(G)(x).$$

In particular, if x *is classified in both zero-loci* $\mathsf{V}_{R,n}(F)$ *and* $\mathsf{V}_{R,n}(G)$*, then* x *is classified in the zero-locus of the concatenated polynomial family* $F{+}{+}G$*.*

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus intersection concat`

Proof. Unfold Definition 154.1. The finite-spine append split is the positional form underlying Definition 112.18: a position of $F{+}{+}G$ is either a left position coming from F or a right position coming from G, and the corresponding readback is respectively entry_F or entry_G.

For the forward implication, assume $\mathsf{V}_{R,n}(F{+}{+}G)(x)$. Its first component gives $\mathsf{AffPoint}_{R,n}(x)$. To prove $\mathsf{V}_{R,n}(F)(x)$, keep this same affine-point witness and fix a position $i : \mathsf{Pos}(F)$. Embed i as the left position of $F{+}{+}G$. The appended-family zero row gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_{F{+}{+}G}(\iota_F(i)), x),$$

and the left-readback clause identifies this entry with $\mathsf{entry}_F(i)$. This proves the required equation row for F. The proof for G is identical, using the right position embedding $\iota_G(j) : \mathsf{Pos}(F{+}{+}G)$ for each $j : \mathsf{Pos}(G)$. Thus both $\mathsf{V}_{R,n}(F)(x)$ and $\mathsf{V}_{R,n}(G)(x)$ hold.

For the reverse implication, assume $\mathsf{V}_{R,n}(F)(x) \wedge \mathsf{V}_{R,n}(G)(x)$. The affine-point component for the concatenated zero-locus is the one supplied by the F component. Now fix a position $k : \mathsf{Pos}(F{+}{+}G)$. By the same finite-spine append split, k is either a left position $\iota_F(i)$ or a right position $\iota_G(j)$. In the left case, the equation row of $\mathsf{V}_{R,n}(F)(x)$ gives $\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x)$, which is the required row for $\mathsf{entry}_{F{+}{+}G}(k)$. In the right case, the equation row of $\mathsf{V}_{R,n}(G)(x)$ gives $\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_G(j), x)$, again with the required readback equality for the appended spine.

The affine-point witness and this case analysis over every position of $F{+}{+}G$ repack as $\mathsf{V}_{R,n}(F{+}{+}G)(x)$, giving the displayed equivalence and the stated intersection implication. □

Theorem 154.3 (Affine empty-family zero-locus is the ambient affine space)**.** *In the setup of Definition 154.1, let* $\mathsf{nil} : \mathsf{HistSpine}$ *denote the empty finite polynomial-family spine. For every point history* $x : \mathsf{BHist}$*,*

$$\mathsf{V}_{R,n}(\mathsf{nil})(x) \iff \mathsf{AffPoint}_{R,n}(x).$$

In particular, $\mathsf{AffPoint}_{R,n}(x)$ *implies* $\mathsf{V}_{R,n}(\mathsf{nil})(x)$*.*

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus empty family iff`

Proof. Unfold Definition 154.1. The forward implication is projection from the defining conjunction, since every zero-locus point carries its ambient affine-point witness.

For the reverse implication, assume $\mathsf{AffPoint}_{R,n}(x)$. It remains only to supply the equation row

$$\forall i : \mathsf{Pos}(\mathsf{nil}).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_{\mathsf{nil}}(i), x).$$

The nil finite spine has no positions, by the finite-spine nil clause of Definition 36.57 and the empty-position reading used in Definition 112.18. Hence this universal row has no instances to check. Pairing the given affine-point witness with the empty equation row gives $\mathsf{V}_{R,n}(\mathsf{nil})(x)$. □

Definition 154.4 (Affine finite-family equation inclusion)**.** *In the setup of Definition 154.1, for finite polynomial-family spines* $F, G : \mathsf{HistSpine}$ *over the same affine* n*-space, write*

$$\mathsf{EqFamIncl}(F, G) \iff \forall i : \mathsf{Pos}(F).\ \exists j : \mathsf{Pos}(G).\ \mathsf{entry}_F(i) = \mathsf{entry}_G(j).$$

This is the position-readback form of saying that every equation occurrence read from F *is also read from* G*; it uses the same finite-position convention as Definition 112.1.*

Lean4 (definition) `BEDC.Derived.AffineVarUp.AffineFiniteFamilyEquationInclusion`

Lemma 154.5 (Equation rows restrict along finite-family inclusion)**.** *In the setup of Definition 154.1, let $F, G : \mathsf{HistSpine}$ be finite polynomial-family spines over the same affine n-space. If $\mathsf{EqFamIncl}(F, G)$ holds, then for every point history $x : \mathsf{BHist}$,*

$$\Big(\forall j : \mathsf{Pos}(G).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_G(j), x)\Big) \Longrightarrow \Big(\forall i : \mathsf{Pos}(F).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x)\Big).$$

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyEquationRows restrict`

Proof. Assume $\mathsf{EqFamIncl}(F, G)$ and assume the displayed equation row for G. Fix $i : \mathsf{Pos}(F)$. By Definition 154.4, choose $j : \mathsf{Pos}(G)$ with

$$\mathsf{entry}_F(i) = \mathsf{entry}_G(j).$$

The equation row for G at j gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_G(j), x).$$

Transport this predicate across the displayed readback equality to obtain

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x).$$

Since i was arbitrary, the required equation row for F follows. □

Theorem 154.6 (Affine finite-family inclusion reverses zero-loci)**.** *In the setup of Definition 154.1, let $F, G : \mathsf{HistSpine}$ be finite polynomial-family spines over the same affine n-space. If $\mathsf{EqFamIncl}(F, G)$ holds, then for every point history $x : \mathsf{BHist}$,*

$$\mathsf{V}_{R,n}(G)(x) \Longrightarrow \mathsf{V}_{R,n}(F)(x).$$

Thus the affine finite-family zero-locus assignment is contravariant with respect to finite-position equation inclusion.

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus inclusion contravariant`
Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus inclusion reverse`

Proof. Assume $\mathsf{EqFamIncl}(F, G)$ and fix $x : \mathsf{BHist}$ with $\mathsf{V}_{R,n}(G)(x)$. Unfold Definition 154.1. The zero-locus witness for G consists of an ambient witness

$$\mathsf{AffPoint}_{R,n}(x)$$

and an equation row

$$\forall j : \mathsf{Pos}(G).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_G(j), x).$$

Keep the same ambient witness for F. The equation row for F is supplied by Definition 154.5, applied to the displayed inclusion and the displayed equation row for G. Pairing these two components gives $\mathsf{V}_{R,n}(F)(x)$. □

Definition 154.7 (Affine equation head insertion)**.** *In the setup of Definition 154.1, for a polynomial-code history $p : \mathsf{BHist}$ and a finite polynomial-family spine $F : \mathsf{HistSpine}$, define the affine equation insertion spine by*

$$\mathsf{insert}_{\mathrm{eq}}(p, F) := \mathsf{cons}(p, F).$$

Thus insertion places the displayed equation as the head row and leaves the original finite equation family as the tail spine.

Lean4 (definition) `BEDC.Derived.AffineVarUp.AffineEquationHeadInsertion`

Lemma 154.8 (Affine zero-loci contain occurred equation rows)**.** *In the setup of Definition 154.1, let* $F : \mathsf{HistSpine}$ *be a finite polynomial-family spine, and let* $p, x : \mathsf{BHist}$*. If*

$$\mathsf{SpineMember}(p, F) \qquad \textit{and} \qquad \mathsf{V}_{R,n}(F)(x)$$

hold, then

$$\mathsf{PolyEvalZero}_{R,n}(p, x)$$

holds.

Proof. By Definition 112.11, the occurrence $\mathsf{SpineMember}(p, F)$ supplies a position $j : \mathsf{Pos}(F)$ with

$$\mathsf{entry}_F(j) = p.$$

Unfolding Definition 154.1 in $\mathsf{V}_{R,n}(F)(x)$ gives the equation row

$$\forall i : \mathsf{Pos}(F).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x).$$

Evaluating this row at j gives $\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(j), x)$. Rewriting along $\mathsf{entry}_F(j) = p$ gives the displayed zero-evaluation predicate for p at x. □

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus occurred equation row`

Theorem 154.9 (Affine duplicate equation insertion invariance)**.** *In the setup of Definition 154.1, let* $F : \mathsf{HistSpine}$ *be a finite polynomial-family spine, and let* $p, x : \mathsf{BHist}$*. If*

$$\mathsf{SpineMember}(p, F)$$

holds, then

$$\mathsf{V}_{R,n}(\mathsf{insert}_{\mathrm{eq}}(p, F))(x) \Longleftrightarrow \mathsf{V}_{R,n}(F)(x).$$

Equivalently, placing at the head a polynomial equation that already occurs in the finite equation spine does not change the affine finite-family zero-locus predicate.

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus duplicate head insert`

Proof. Use Definition 154.7 to read $\mathsf{insert}_{\mathrm{eq}}(p, F)$ as $\mathsf{cons}(p, F)$.

For the forward implication, assume $\mathsf{V}_{R,n}(\mathsf{cons}(p, F))(x)$. Unfold Definition 154.1. The affine-point witness is the first component of the cons-spine zero-locus witness. For the equation row of F, fix $i : \mathsf{Pos}(F)$ and view it as the tail position of $\mathsf{cons}(p, F)$. The cons-position readback from the finite-spine clauses of Definition 36.57 identifies this tail entry with $\mathsf{entry}_F(i)$. The equation row of $\mathsf{V}_{R,n}(\mathsf{cons}(p, F))(x)$ at that tail position therefore gives $\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x)$. Since i was arbitrary, these data repack as $\mathsf{V}_{R,n}(F)(x)$.

For the reverse implication, assume $\mathsf{V}_{R,n}(F)(x)$. Unfold Definition 154.1. The affine-point component for $\mathsf{cons}(p, F)$ is the one supplied by $\mathsf{V}_{R,n}(F)(x)$. It remains to verify every equation row of the cons spine. A position of $\mathsf{cons}(p, F)$ is either the head position or a tail position. In the head case, Definition 154.8, applied to the displayed occurrence of p in F and to $\mathsf{V}_{R,n}(F)(x)$, gives $\mathsf{PolyEvalZero}_{R,n}(p, x)$, which is exactly the head equation row. In the tail case, the position is inherited from some $i : \mathsf{Pos}(F)$, and the tail readback is $\mathsf{entry}_F(i)$. The equation row of $\mathsf{V}_{R,n}(F)(x)$ at i gives the required predicate for that tail entry.

The affine-point witness and the head-or-tail verification of every cons-spine position repack as $\mathsf{V}_{R,n}(\mathsf{cons}(p, F))(x)$, and hence as $\mathsf{V}_{R,n}(\mathsf{insert}_{\mathrm{eq}}(p, F))(x)$. □

Theorem 154.10 (Affine mutual family inclusion gives equal zero-loci)**.** *In the setup of Definition 154.1, let $F, G : \mathsf{HistSpine}$ be finite polynomial-family spines over the same affine n-space. If both*

$$\mathsf{EqFamIncl}(F, G) \qquad \textit{and} \qquad \mathsf{EqFamIncl}(G, F)$$

hold, then for every point history $x : \mathsf{BHist}$,

$$\mathsf{V}_{R,n}(F)(x) \Longleftrightarrow \mathsf{V}_{R,n}(G)(x).$$

Thus mutually included finite affine equation presentations classify exactly the same zero-locus points.

Proof. Assume $\mathsf{EqFamIncl}(F, G)$ and $\mathsf{EqFamIncl}(G, F)$, and fix a point history $x : \mathsf{BHist}$. To prove the forward implication, assume $\mathsf{V}_{R,n}(F)(x)$. Apply Definition 154.6 to the inclusion $\mathsf{EqFamIncl}(G, F)$, with the ordered pair of families read as (G, F). Since inclusion reverses zero-loci, this gives $\mathsf{V}_{R,n}(G)(x)$.

For the reverse implication, assume $\mathsf{V}_{R,n}(G)(x)$. Apply Definition 154.6 to the inclusion $\mathsf{EqFamIncl}(F, G)$, with the ordered pair of families read as (F, G). This gives $\mathsf{V}_{R,n}(F)(x)$.

The two implications together give the displayed equivalence for the fixed x, and x was arbitrary. □

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus mutual inclusion iff`

Theorem 154.11 (AffineVarUp public zero-locus certificate)**.** *The public $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ export is the finite-family zero-locus interface of Definition 154.1 together with equation inclusion, row restriction, concatenation as intersection, empty-family ambient-space exactness, duplicate insertion invariance, and mutual-inclusion classification. Its exact exported rows are the carrier predicate $\mathsf{V}_{R,n}(F)$, the finite equation-ledger positions of F, the inclusion relation $\mathsf{EqFamIncl}$ from Definition 154.4, and the zero-locus transport and classifier rows proved in this chapter.*

Proof. The carrier row is Definition 154.1. It exposes exactly the ambient affine-point witness and the finite equation-row ledger indexed by positions of the polynomial-family spine.

The stability rows are supplied by the checked zero-locus clauses in this chapter. Apply Definition 154.2 for concatenation as intersection, Definition 154.3 for empty-family ambient-space exactness, Definition 154.5 for row restriction along equation inclusion, and Definition 154.9 for duplicate insertion invariance.

The public classifier row is the mutual-inclusion exactness theorem Definition 154.10. Together with the equation-inclusion definition, it identifies precisely when two finite affine equation presentations export the same zero-locus predicate. These rows repack the finite-family interface as the public $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ certificate surface; projective consumer, scheme consumer, and full $\mathsf{AffineVar}^{\uparrow}$ $\mathsf{NameCert}$ fields remain outside this finite-family surface. □

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus head insert equation rows`

Theorem 154.12 (Affine finite-family standard zero-locus bridge)**.** *In the setup of Definition 154.1, the BEDC finite-family zero-locus carrier $\mathsf{V}_{R,n}(F)$, together with equation inclusion $\mathsf{EqFamIncl}$ from Definition 154.4 and the mutual-inclusion classifier of Definition 154.10, interprets as the standard finite polynomial zero-locus predicate on affine points. Conversely, every standard finite-family zero-locus membership row is represented by the same $\mathsf{HistSpine}$ positions and the same $\mathsf{PolyEvalZero}$ rows used by $\mathsf{V}_{R,n}(F)$.*

Proof. Read $\mathsf{V}_{R,n}(F)$ by Definition 154.1. A BEDC member is an affine point x together with one zero-evaluation row for each finite position $i : \mathsf{Pos}(F)$. The readback $\mathsf{entry}_F(i)$ is exactly the polynomial equation at that position, so the BEDC carrier gives the usual standard statement that every equation in the finite family vanishes at x.

Conversely, suppose a standard finite-family zero-locus row is given for the same family spine F. For each position $i : \mathsf{Pos}(F)$, read the polynomial equation as $\mathsf{entry}_F(i)$ and place the standard vanishing datum in the corresponding $\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x)$ row. Together with the ambient affine-point row, these finite-position rows repack as $\mathsf{V}_{R,n}(F)(x)$ by the same definition.

The equation-inclusion bridge is positional rather than quotient-based. By Definition 154.4, $\mathsf{EqFamIncl}(F, G)$ is exactly the assertion that every equation position of F has a readback match in G. Applying Definition 154.5 projects the standard zero rows along this inclusion, and Definition 154.10 gives equality of the represented zero-locus predicates under mutual inclusion. No quotienting of presentations and no host equality of carriers is used. □

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus standard zero locus bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{AffineVar}^\uparrow}$.

Constructive story: An affine variety here is built from a commutative-ring certificate, an affine point carrier, and a finite HistSpine of polynomial-code histories, with a point history accepted precisely when it is an affine point and every displayed equation position evaluates to the scalar zero row. The carrier $\mathsf{V}_{R,n}(F)$ includes the affine points satisfying all equations in the finite family and excludes points lacking the ambient row, polynomial codes outside the normalized finite sequence substrate, and presentations not accounted for by finite positions. Two finite equation presentations classify the same zero locus when their equation families include one another position by position, so the classifier is mutual equation inclusion plus the induced equivalence of zero-locus predicates rather than host equality of varieties. The source specification covers the affine point row, polynomial-code row, zero-evaluation witness, and finite family spine, while the pattern records empty-family exactness, intersection by concatenation, row restriction, duplicate head insertion, and mutual-inclusion exactness. The ledger keeps equation positions, readback entries, polynomial evaluation rows, and the affine point witness visible, so membership is aligned with the finite family that generated it; the five obligations then come from the ambient carrier and empty-family case, mutual inclusion as a reflexive, symmetric, and transitive classifier, and row restriction keeping transported finite families inside the carrier. A history h lies in the signature gap of an $\mathsf{AffineVar}^\uparrow$ token p when the source domain accepts h as such a finite zero-locus packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; concretely, each equation position in p has a zero-evaluation row at the affine point represented by h. The standard zero-locus bridge reads the BEDC rows as the finite polynomial vanishing predicate and reads a standard membership row back into the same positions, so the chapter registers this finite-equation construction under $\mathsf{AffineVar}^\uparrow$ without quotienting presentations or adding a host variety primitive.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AffineVar}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper-side surface consists of the finite-family carrier $\mathsf{V}_{R,n}(F)$, the equation-inclusion classifier EqFamIncl, row restriction, intersection by concatenation, empty-family exactness, duplicate insertion, mutual-inclusion exactness, the finite-position ledger, and the paper standard zero-locus bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AffineVar}^\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus\_mutual\_inclusion\_iff`

Bridge status: paperBridge

Not claimed: This chapter does not close projective-variety consumer certificates, scheme-level consumer certificates, the full $\mathsf{AffineVar}^\uparrow$ NameCert, or machine-checked bridge closure.

Upgrade path: The next verification step is bridgeCheckedV and audit packaging for the same finite-family bridge, not a stronger paper closure claim.

Theorem 154.13 (Affine singleton-equation zero-locus exactness)**.** *In the setup of Definition 154.1, let* $p : \mathsf{BHist}$ *be a polynomial-code history and let* $\mathsf{cons}(p, \mathsf{nil}) : \mathsf{HistSpine}$ *be the singleton polynomial-family spine. For every point history* $x : \mathsf{BHist}$*,*

$$\mathsf{V}_{R,n}(\mathsf{cons}(p, \mathsf{nil}))(x) \iff \mathsf{AffPoint}_{R,n}(x) \wedge \mathsf{PolyEvalZero}_{R,n}(p, x).$$

Thus the one-equation affine finite-family zero-locus is exactly the ambient affine-point predicate together with vanishing of the displayed polynomial code.

Lean4 ✓ `BEDC.Derived.AffineVarUp.AffineFiniteFamilyZeroLocus singleton equation exactness`

Proof. Unfold Definition 154.1. A witness of

$$\mathsf{V}_{R,n}(\mathsf{cons}(p, \mathsf{nil}))(x)$$

is a pair consisting of

$$\mathsf{AffPoint}_{R,n}(x)$$

and an equation row

$$\forall i : \mathsf{Pos}(\mathsf{cons}(p, \mathsf{nil}))\,.\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_{\mathsf{cons}(p,\mathsf{nil})}(i), x).$$

By the cons-spine position clauses of Definition 36.57, the singleton spine has one head position and no tail positions, since the tail is nil. At the head position, the readback is p.

For the forward implication, assume $\mathsf{V}_{R,n}(\mathsf{cons}(p, \mathsf{nil}))(x)$. The first projection gives $\mathsf{AffPoint}_{R,n}(x)$. Evaluating the displayed equation row at the head position of $\mathsf{cons}(p, \mathsf{nil})$ gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_{\mathsf{cons}(p,\mathsf{nil})}(\mathsf{head}), x).$$

The singleton head readback identifies $\mathsf{entry}_{\mathsf{cons}(p,\mathsf{nil})}(\mathsf{head})$ with p, so this is exactly $\mathsf{PolyEvalZero}_{R,n}(p, x)$. Pairing these two projections gives the right-hand side.

For the reverse implication, assume

$$\mathsf{AffPoint}_{R,n}(x) \wedge \mathsf{PolyEvalZero}_{R,n}(p, x).$$

The affine-point component of $\mathsf{V}_{R,n}(\mathsf{cons}(p, \mathsf{nil}))(x)$ is the first component of this pair. It remains to verify the equation row for every position of $\mathsf{cons}(p, \mathsf{nil})$. Such a position is the head position or a tail position inherited from nil. In the head case, the readback is p, and the second component supplies $\mathsf{PolyEvalZero}_{R,n}(p, x)$. In the tail case there is no position to check, because nil has no positions. Hence the equation row is complete.

Combining the ambient affine-point witness with this singleton equation row repacks as $\mathsf{V}_{R,n}(\mathsf{cons}(p, \mathsf{nil}))(x)$ by Definition 154.1. □

Definition 154.14 (Affine finite-family equation union presentation)**.** *In the setup of Definition 154.1, let F, G, U : HistSpine be finite polynomial-family spines over the same affine n-space. Say that U presents the finite equation-family union of F and G, and write $\mathsf{EqFamUnion}(F, G; U)$, when the following three clauses hold:*

$$\mathsf{EqFamIncl}(F, U), \qquad \mathsf{EqFamIncl}(G, U),$$

and

$$\forall k : \mathsf{Pos}(U)\,.\ (\exists i : \mathsf{Pos}(F).\ \mathsf{entry}_U(k) = \mathsf{entry}_F(i)) \vee (\exists j : \mathsf{Pos}(G).\ \mathsf{entry}_U(k) = \mathsf{entry}_G(j))\,.$$

The first two clauses say that every displayed equation occurrence of F and G occurs in the union presentation. The last clause says that every displayed equation occurrence of U comes from one of the two input families. Thus the definition is an exact finite-position union presentation rather than an ambient host-set quotient.

Theorem 154.15 (Affine zero-locus of an equation-family union)**.** *In the setup of Definition 154.1, let $F, G, U : \mathsf{HistSpine}$ be finite polynomial-family spines over the same affine n-space. If $\mathsf{EqFamUnion}(F, G; U)$ holds, then for every point history $x : \mathsf{BHist}$,*

$$\mathsf{V}_{R,n}(U)(x) \Longleftrightarrow \mathsf{V}_{R,n}(F)(x) \wedge \mathsf{V}_{R,n}(G)(x).$$

Consequently, membership in the affine zero-locus of the displayed finite equation-family union is exactly simultaneous membership in the two affine zero-loci.

Proof. Assume $\mathsf{EqFamUnion}(F, G; U)$, and write its three components as

$$\mathsf{EqFamIncl}(F, U), \qquad \mathsf{EqFamIncl}(G, U),$$

and the union-coverage row

$$\forall k : \mathsf{Pos}(U).\ (\exists i : \mathsf{Pos}(F).\ \mathsf{entry}_U(k) = \mathsf{entry}_F(i)) \vee (\exists j : \mathsf{Pos}(G).\ \mathsf{entry}_U(k) = \mathsf{entry}_G(j))\,.$$

For the forward implication, assume $\mathsf{V}_{R,n}(U)(x)$. Apply Definition 154.6 to $\mathsf{EqFamIncl}(F, U)$ to obtain $\mathsf{V}_{R,n}(F)(x)$, and apply the same theorem to $\mathsf{EqFamIncl}(G, U)$ to obtain $\mathsf{V}_{R,n}(G)(x)$. Pairing these two projected zero-locus witnesses gives

$$\mathsf{V}_{R,n}(F)(x) \wedge \mathsf{V}_{R,n}(G)(x).$$

For the reverse implication, assume

$$\mathsf{V}_{R,n}(F)(x) \wedge \mathsf{V}_{R,n}(G)(x).$$

Unfold Definition 154.1. The affine-point witness for $\mathsf{V}_{R,n}(U)(x)$ is the affine-point component of $\mathsf{V}_{R,n}(F)(x)$. It remains to verify every equation row of U. Fix $k : \mathsf{Pos}(U)$. By the union-coverage row, k is covered by one of the two input families.

In the first case, choose $i : \mathsf{Pos}(F)$ with

$$\mathsf{entry}_U(k) = \mathsf{entry}_F(i).$$

The equation row of $\mathsf{V}_{R,n}(F)(x)$ at i gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(i), x).$$

Transporting this predicate along the displayed readback equality gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_U(k), x).$$

In the second case, choose $j : \mathsf{Pos}(G)$ with

$$\mathsf{entry}_U(k) = \mathsf{entry}_G(j).$$

The equation row of $\mathsf{V}_{R,n}(G)(x)$ at j gives

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_G(j), x),$$

and transport along the displayed readback equality gives the required predicate

$$\mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_U(k), x).$$

Since k was arbitrary, every equation row of U vanishes at x. Pairing this row with the affine-point witness repacks as $\mathsf{V}_{R,n}(U)(x)$ by Definition 154.1. The two implications give the displayed equivalence. □

Theorem 154.16 (AffineVar zero-locus is antitone under equation inclusion)**.** *In the setup of Definition 154.1, let* $E, F : \mathsf{HistSpine}$ *be finite polynomial-family spines over the same affine n-space. If*

$$\mathsf{EqFamIncl}(E, F)$$

holds, then for every point history $x : \mathsf{BHist}$*,*

$$\mathsf{V}_{R,n}(F)(x) \Longrightarrow \mathsf{V}_{R,n}(E)(x).$$

Thus adding equation occurrences to a finite affine family can only shrink its zero-locus predicate: equation-family inclusion is sent to reverse implication of zero-locus membership.

Proof. Assume $\mathsf{EqFamIncl}(E, F)$, fix $x : \mathsf{BHist}$, and assume $\mathsf{V}_{R,n}(F)(x)$. By Definition 154.1, the latter witness consists of an ambient affine-point row

$$\mathsf{AffPoint}_{R,n}(x)$$

and the equation row

$$\forall j : \mathsf{Pos}(F).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_F(j), x).$$

Keep the same ambient affine-point row for E. It remains to prove

$$\forall i : \mathsf{Pos}(E).\ \mathsf{PolyEvalZero}_{R,n}(\mathsf{entry}_E(i), x).$$

This row is exactly the restriction of the displayed F-row along the inclusion $\mathsf{EqFamIncl}(E, F)$: by Definition 154.5, every position $i : \mathsf{Pos}(E)$ has a matching position $j : \mathsf{Pos}(F)$ with the same equation readback, and the zero-evaluation predicate transports across that readback equality.

Pairing the retained affine-point row with this restricted equation row repacks, by Definition 154.1, as $\mathsf{V}_{R,n}(E)(x)$. Since x was arbitrary, the zero-locus assignment is antitone with respect to finite-position equation inclusion. □

Chapter 155

A Concrete Naming Certificate for $\mathsf{ProjectiveVar}^{\uparrow}$

The projective-variety interface $\mathsf{ProjectiveVar}^{\uparrow}$ packages the zero-locus of a homogeneous polynomial ideal in the projective n-space of a commutative-ring carrier. It depends on $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ (chapter 154) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

155.1 Carrier definition

155.2 The certificate

Theorem 155.1 (ProjectiveVarUp carrier obligation)**.** *The* $\mathsf{ProjectiveVar}^{\uparrow}$ *carrier is a* BHist *predicate assembled from the visible* $\mathsf{AffineVar}^{\uparrow}$ *chart zero-locus source, the* $\mathsf{Poly}^{\uparrow}$ *homogeneous-family rows, the projective-space carrier row, and the zero-evaluation package. A carried endpoint is accepted exactly when those four rows are present with their* Pkg *provenance and probe reads; no host quotient, host orbit set, or external projective point equality is part of the carrier predicate.*

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier endpoint exactness`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier carrier obligation`

Proof. Read the carrier row through the roadmap ten-field template:

1. traditional target: the projective zero-locus of a homogeneous polynomial family;
2. BEDC source: a BHist endpoint carrying chart, homogeneous-family, projective-space, and zero-evaluation rows;
3. carrier field: the visible predicate formed by those four rows and their Pkg provenance;
4. dependency fields: $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ from chapter 154 and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ from chapter 50;
5. classifier field: reserved for projective scaling and chart-compatible hsame transport;
6. stability field: chart zero-locus rows and homogeneous rows are transported together;
7. exactness field: zero evaluation is read only through the displayed package;
8. ledger field: chart choice, family presentation, scale witness, and routing evidence remain hidden;

9. theorem-list field: the carrier, classifier, exactness, and ledger obligations named in this section;
10. boundary field: downstream consumers see only the projective carrier and classifier.

The carrier projection is therefore just the third entry of the template. If an endpoint supplies the four displayed rows, it repacks as a carried $\mathsf{ProjectiveVar}^{\uparrow}$ history; if a carried endpoint is projected back out, the same rows are recovered. Host quotients and orbit representatives occur only as ledger readings and cannot enlarge the predicate. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarFiniteChartCarrier endpoint transport`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarCarrier obligation rows`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarCarrier obligation`

Theorem 155.2 (ProjectiveVarUp classifier stability obligation)**.** *The* $\mathsf{ProjectiveVar}^{\uparrow}$ *classifier is projective scaling together with chart-compatible* hsame *transport. Whenever two carried endpoints are related by a scale witness and their chart zero-locus rows transport through the corresponding* $\mathsf{AffineVar}^{\uparrow}$ *chart classifiers, the projective carrier is preserved and each homogeneous* $\mathsf{Poly}^{\uparrow}$ *row is transported to a row with the same zero-evaluation outcome.*

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarZeroLocusPackage hsame transport`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarCarrier scale chart transport`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarCarrier chart homogeneous transport`

Proof. Read the classifier row through the roadmap ten-field template:

1. traditional target: equality of projective zero-locus points up to nonzero scalar multiplication;
2. BEDC source: two carried BHist endpoints with chart rows, homogeneous-family rows, projective-space rows, and zero-evaluation packages;
3. carrier field: the carrier predicate of Definition 155.1 on both endpoints;
4. dependency fields: $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ from chapter 154 and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ from chapter 50;
5. classifier field: scale witness plus chart-compatible hsame transport;
6. stability field: transported chart rows preserve carrier membership;
7. exactness field: homogeneous zero-evaluation is invariant under the transported presentation;
8. ledger field: the chosen scale and chart route remain hidden provenance;
9. theorem-list field: this classifier row feeds the exactness and ledger obligations below;
10. boundary field: consumers may use only the resulting carrier-preserving classifier.

Apply the scale witness to the projective-space row and the hsame transport to the chart row. The homogeneous-family rows are then read through the $\mathsf{Poly}^{\uparrow}$ source certificate, so evaluation at transported representatives remains a zero-evaluation package fact rather than a host equality fact.

Thus the classifier preserves the carrier predicate. Conversely, any classifier witness visible at the $\mathsf{ProjectiveVar}^{\uparrow}$ boundary must decompose into exactly the scale route, chart transport, homogeneous-row transport, and provenance entries listed above. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier classifier stability`

Theorem 155.3 (ProjectiveVarUp homogeneous zero-locus exactness obligation)**.** *The visible zero-locus package for* $\mathsf{ProjectiveVar}^{\uparrow}$ *is exact relative to the homogeneous* $\mathsf{Poly}^{\uparrow}$ *family and the* $\mathsf{AffineVar}^{\uparrow}$ *chart exactness rows: coverage says every carried projective endpoint has a chart read and homogeneous zero-evaluation read; soundness says every accepted read satisfies all displayed homogeneous rows on every compatible chart; completeness says every compatible chart-family zero-locus read repacks as a carried projective endpoint.*

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarHomogeneousZeroLocusCarrier bundle append exactness`
Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarHomogeneousZeroLocus visible package`

Proof. Read the exactness row through the roadmap ten-field template:

1. traditional target: a projective homogeneous-family zero-locus;
2. BEDC source: chart-local zero-locus reads tied to homogeneous polynomial-code rows;
3. carrier field: the projective carrier predicate of Definition 155.1;
4. dependency fields: $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ from chapter 154 and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ from chapter 50;
5. classifier field: the scaling and chart-transport row of Definition 155.2;
6. stability field: compatible affine charts carry the same homogeneous-family zero tests;
7. exactness field: coverage, soundness, and completeness for visible zero-locus reads;
8. ledger field: chart selections and evaluation routing remain provenance data;
9. theorem-list field: carrier and classifier obligations supply the projections consumed here;
10. boundary field: downstream consumers receive an exact visible zero-locus package, not host algebraic geometry data.

For coverage, project a carried endpoint to its chart read and homogeneous-family zero-evaluation read. Both are already visible components of the carrier row, so no extra witness is introduced.

For soundness, use the $\mathsf{AffineVar}^{\uparrow}$ chart exactness rows to interpret the chart zero-locus source and the $\mathsf{Poly}^{\uparrow}$ homogeneous-family rows to interpret each zero-evaluation entry. The classifier stability obligation transports these entries across compatible charts.

For completeness, begin with a compatible chart-family read satisfying the listed zero-evaluation rows. Repack the chart source, homogeneous family, projective-space row, and routing package into the carrier predicate, then attach the ledger entries for chart choice and scale routing. The result is a carried $\mathsf{ProjectiveVar}^{\uparrow}$ endpoint with exactly the advertised visible zero-locus package. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier zero locus exactness`

Theorem 155.4 (ProjectiveVarUp scaling ledger obligation)**.** *The* $\mathsf{ProjectiveVar}^{\uparrow}$ *ledger hides exactly chart choice, homogeneous-family presentation, scale witnesses, and zero-evaluation package routing. It exposes only the projective carrier of Definition 155.1 and the projective classifier of Definition 155.2; no hidden chart, polynomial presentation, scalar representative, or evaluation route becomes public carrier data.*

Proof. Read the ledger row through the roadmap ten-field template:

1. traditional target: presentation independence for a projective homogeneous zero-locus;
2. BEDC source: the carrier and classifier rows already displayed at the $\mathsf{ProjectiveVar}^{\uparrow}$ boundary;

3. carrier field: the projective carrier predicate;
4. dependency fields: $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ from chapter 154 and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ from chapter 50;
5. classifier field: projective scaling with chart-compatible hsame transport;
6. stability field: hidden presentation changes preserve the same visible carrier and classifier;
7. exactness field: zero-locus coverage, soundness, and completeness remain routed through the visible package;
8. ledger field: chart choice, homogeneous-family presentation, scale witness, and zero-evaluation routing are the hidden entries;
9. theorem-list field: the four obligations of this section are the whole certificate surface at this grade;
10. boundary field: consumers cannot inspect anything beyond the projective carrier and classifier.

Projecting the ledger reveals only the entries listed in the eighth template field. Chart choices are justified by the affine source certificate, homogeneous-family presentations by the polynomial source certificate, scale witnesses by the classifier row, and evaluation routing by the zero-locus package.

Since each hidden entry is consumed by one of those visible rows, exposing it would duplicate a presentation witness rather than add a BEDC carrier field. The public boundary is therefore exactly the carrier and classifier named in the statement. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier scaling ledger`

Theorem 155.5 (ProjectiveVarUp scoped primitive binding)**.** *The four rows in Definition 155.1, Definition 155.2, Definition 155.3, and Definition 155.4 are scoped exactly by the BEDC primitives* BHist, Pkg*, probe reads,* hsame *transport, the projective scaling witness,* $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*. Chart choice, homogeneous-family presentation, scale witness, and zero-evaluation routing remain ledger data, and no host quotient or external projective-point equality enters the public carrier.*

Proof. Project the carrier row first. Its roadmap dependency fields expose only the BHist predicate, the visible Pkg provenance, the relevant probe reads, and the upstream $\mathsf{AffineVar}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ certificates named in the statement.

Project the classifier row next. Its dependency and classifier fields bind the projective scaling witness to chart-compatible hsame transport, while the carrier field remains the predicate supplied by Definition 155.1.

Project the exactness row. Coverage, soundness, and completeness consume the same chart reads, homogeneous-family rows, zero-evaluation package, and upstream certificate exactness already present in Definition 155.3; they do not add a host quotient or an external equality relation.

Finally project the ledger row. The ledger field names exactly chart choice, homogeneous-family presentation, scale witness, and zero-evaluation routing as hidden provenance, so those witnesses can be transported and consumed but not promoted into the public carrier. Combining the four projections gives the reusable scoped boundary for $\mathsf{ProjectiveVar}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarTasteGate single carrier alignment`

Theorem 155.6 (ProjectiveVarUp public NameCert export)**.** *The four rows in Definition 155.1, Definition 155.2, Definition 155.3, and Definition 155.4 assemble a public* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$

surface. Its carrier is the visible projective zero-locus predicate, its classifier is projective scaling together with chart-compatible hsame *transport, its exactness field is the visible homogeneous zero-locus package, and its ledger hides only chart choice, homogeneous-family presentation, scale witnesses, and zero-evaluation routing.*

Proof. Repack the four scoped rows through the roadmap ten-field template. The traditional-target field is the presentation-independent projective homogeneous zero-locus, and the BEDC-source field is the visible package built from the $\mathsf{AffineVar}^{\uparrow}$ chart certificate, the $\mathsf{Poly}^{\uparrow}$ homogeneous-family certificate, the projective-space carrier row, BHist endpoints, Pkg provenance, probe reads, scale witnesses, and zero-evaluation routing.

The carrier field is exactly the visible projective zero-locus predicate supplied by Definition 155.1. The dependency fields are the upstream $\mathsf{NameCert}_{\mathsf{AffineVar}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ surfaces already consumed by that carrier row, so no hidden chart or polynomial presentation is exported as carrier data.

The classifier and stability fields come from Definition 155.2: projective scaling witnesses identify representative changes, while chart-compatible hsame transport keeps the visible zero tests stable across compatible affine reads. The exactness field is Definition 155.3, which gives coverage, soundness, and completeness for the visible homogeneous zero-locus package.

Finally, the ledger field is Definition 155.4. It hides only chart choice, homogeneous-family presentation, scale witnesses, and zero-evaluation routing, and it exposes only the carrier, classifier, stability, and exactness fields named above. The theorem-list and boundary fields are therefore public: downstream consumers may use the assembled $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ surface, but they cannot inspect any presentation witness beyond the advertised ledger. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier public namecert export`

Theorem 155.7 (ProjectiveVarUp public zero-locus downstream bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ *surface of Definition 155.6 is consumed by downstream zero-locus interfaces exactly through the visible projective carrier, the scaling-and-chart classifier, the homogeneous zero-locus exactness package, and the scaling ledger. Any* $\mathsf{AffineVar}^{\uparrow}$ *chart read,* $\mathsf{Poly}^{\uparrow}$ *homogeneous-family read, or* $\mathsf{ProjectiveSpace}^{\uparrow}$ *representative read used by a downstream consumer factors through those four rows; conversely those four rows supply the consumer boundary without exposing chart choice, homogeneous presentation, scale witness, or zero-evaluation routing as public carrier data.*

Proof. Project the public certificate of Definition 155.6. Its carrier projection is the visible BHist predicate supplied by Definition 155.1, so every downstream zero-locus read begins with the same projective endpoint boundary rather than with a chart-local carrier.

The classifier projection is the scaling-and-chart row of Definition 155.2. It is the only public route by which an $\mathsf{AffineVar}^{\uparrow}$ chart read or a $\mathsf{ProjectiveSpace}^{\uparrow}$ representative read may be transported, and the transport keeps the carried endpoint inside the visible projective carrier.

The exactness projection is Definition 155.3. It supplies coverage, soundness, and completeness for each $\mathsf{Poly}^{\uparrow}$ homogeneous-family read used by the consumer, so zero-evaluation is consumed through the visible package rather than through a separate presentation of the family.

Finally, the ledger projection is Definition 155.4. It records chart choice, homogeneous presentation, scale witness, and zero-evaluation routing as hidden provenance, while the public boundary retains only the carrier, classifier, exactness package, and ledger rows. Repacking those four projections gives the downstream consumer boundary, and projecting that boundary back recovers exactly the same four rows without exposing any hidden witness as public carrier data. □

Lean4 ✓ `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier downstream zero locus bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$.

Constructive story: A projective variety is generated by taking the visible $\mathsf{AffineVar}^{\uparrow}$ chart zero-locus source, the homogeneous $\mathsf{Poly}^{\uparrow}$ family rows, a projective-space carrier row, and the zero-evaluation package, then tying them to one BHist endpoint with Pkg provenance and probe reads. The carrier accepts exactly endpoints with those chart, homogeneous-family, projective-space, and zero-evaluation rows, while host orbit sets, projective-point quotients, unrecorded chart representatives, and external equality of projective points are excluded. Two accepted endpoints classify together when a projective scaling witness and chart-compatible hsame transport carry the affine chart rows and homogeneous polynomial rows to the same zero-evaluation outcomes. The source specification is this visible projective zero-locus package, the pattern records homogeneous zero-locus exactness across compatible charts, and the ledger hides only chart choice, homogeneous-family presentation, scale witnesses, and evaluation routing while exposing the carrier, classifier, stability, and exactness rows. The five naming-certificate obligations follow because the displayed package inhabits the carrier, the scaling-and-chart classifier is reflexive, symmetric, and transitive through its carried transports, and every transported chart or homogeneous row repacks as another accepted projective endpoint. A history h lies in the signature gap of a $\mathsf{ProjectiveVar}^{\uparrow}$ token p when the source domain accepts its projective zero-locus packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; after unfolding, the chart rows, scale witness, homogeneous equations, and zero-evaluation ledger of h are the ones encoded by p. The downstream zero-locus bridge lets later consumers read only this visible carrier, scaling-and-chart classifier, homogeneous exactness package, and scaling ledger, so the chapter registers the construction under $\mathsf{ProjectiveVar}^{\uparrow}$ without exporting hidden chart data, host projective equality, or an external standard model.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper-side boundary consists exactly of the visible projective carrier, the scaling-and-chart classifier, homogeneous zero-locus exactness, the scaling ledger, the public $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ export, and the downstream zero-locus consumer boundary of Definition 155.7.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ProjectiveVarUp.ProjectiveVarVisibleCarrier\_public\_namecert\_export`

Bridge status: paperBridge

Not claimed: This chapter does not claim a standard-model bridge outside the carrier, classifier, exactness, ledger, public export, and downstream zero-locus consumer boundary named above.

Upgrade path: A bridge-checked grade would require a complete standard-model bridge for the same public $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ surface and downstream zero-locus consumer interface.

Chapter 156

A Concrete Naming Certificate for $\mathsf{Homotopy}^{\uparrow}$

The homotopy interface $\mathsf{Homotopy}^{\uparrow}$ packages the equivalence-of-paths relation between continuous maps under a continuous deformation parameter. It depends on $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ (chapter 130) and $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$ (chapter 56), and is the substrate for fundamental-group and higher-homotopy certificates.

156.1 Carrier definition

Definition 156.1 ($\mathsf{Homotopy}^{\uparrow}$ BHist source packet)**.** *A* $\mathsf{Homotopy}^{\uparrow}$ *source packet is a finite* BHist *tuple* $(f, g, H, I, \rho, \lambda)$*. The rows* f *and* g *are the two endpoint* $\mathsf{ContinuousMap}^{\uparrow}$ *rows accepted by* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, the row* H *is the displayed deformation row between them, and* I *is the* $\mathsf{Interval}^{\uparrow}$ *parameter row accepted by* $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$*.*

The row ρ *is the* Pkg *provenance tying the endpoint faces, deformation face, and interval face to those two dependency certificates. The row* λ *is the finite* Cont *ledger recording the endpoint reads of* H *at the two displayed interval faces and every* hsame *transport used by the packet classifier. No host quotient, imported path equality, or external function equality is part of the source packet.*

Definition 156.2 (Homotopy source envelope)**.** *A* $\mathsf{Homotopy}^{\uparrow}$ *source envelope is a finite* BHist *packet*

$$\mathcal{H} = (F, G, I, H, p_0, p_1, \eta, \rho)$$

where F *and* G *are source* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *rows for the displayed endpoint maps,* I *is a* $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$ *parameter row,* H *is the continuous-map row for the deformation family,* p_0 *and* p_1 *are endpoint ledgers selecting the interval endpoints,* η *is the finite* Cont *readback ledger from the deformation row to the two endpoint maps, and* ρ *records the* Pkg *provenance binding the continuous-map and interval sources.*

The envelope is admitted only when F*,* G*,* I*,* H*,* p_0*,* p_1*,* η*, and* ρ *occur as displayed rows of the same finite BEDC packet. Thus* $\mathsf{Homotopy}^{\uparrow}$ *carries a deformation certificate assembled from named continuous-map and interval rows, not a host path space of maps.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyUp single carrier alignment`

Definition 156.3 (Homotopy classifier surface)**.** *Two* $\mathsf{Homotopy}^{\uparrow}$ *source envelopes* $\mathcal{H}$ *and* $\mathcal{H}'$ *classify the same homotopy object exactly when* hsame *transports the two endpoint* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$

rows, the $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$ *parameter row, and the finite* Cont *ledgers reading the deformation endpoint rows. The classifier compares only the displayed endpoint maps, interval parameter, deformation readback, and* Pkg *provenance; it does not quantify over an ambient space of paths between maps.*

Theorem 156.4 (Homotopy ContinuousMap dependency surface)**.** *For every admitted* $\mathsf{Homotopy}^\uparrow$ *source envelope of Definition 156.2, every deformation slice and both endpoint maps project through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$*. The only continuous-map data exposed to a downstream consumer are the endpoint rows* F *and* G*, the deformation row* H*, the finite* Cont *readback ledger* η*, and the shared* Pkg *provenance* ρ*.*

Proof. Unpack the source envelope. Its first, second, and fourth coordinates are declared $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ rows, so the endpoint maps and the deformation carrier are already projected through the continuous-map certificate surface.

The slice information is not a separate host function space. It is read from H using the interval parameter row I and the finite Cont ledger η, so each admitted slice has the same continuous-map dependency row as the displayed deformation packet.

Repacking these projections gives exactly the dependency surface stated above. No projection of the envelope returns an uncarried map, topology, or path object outside $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$. □

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket continuousmap dependency surface`

Theorem 156.5 (Homotopy interval endpoint obligation surface)**.** *For every admitted* $\mathsf{Homotopy}^\uparrow$ *source envelope of Definition 156.2, the two interval endpoint rows* p_0 *and* p_1 *select the displayed source and target maps by finite* Cont *readback from the deformation row* H*. The source endpoint readback returns* F*, and the target endpoint readback returns* G*, both under the interval classifier inherited from* $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket interval endpoint determinacy`

Proof. Project the interval coordinate I and the endpoint ledgers p_0 and p_1 from the finite packet. By admission, these rows are part of the same BHist envelope as the deformation row H and the endpoint continuous-map rows F and G.

Read the deformation row along p_0. The ledger η records this readback as a finite Cont row whose endpoint is the displayed source map row F, with interval transport supplied by the $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$ classifier.

Read the deformation row along p_1 in the same way. The corresponding endpoint of the same finite ledger is G, so the target map is selected without introducing an external endpoint-evaluation operation. □

Theorem 156.6 (Homotopy certificate obligation surface)**.** *The* $\mathsf{Homotopy}^\uparrow$ *certificate obligation surface consists exactly of the source envelope of Definition 156.2, the classifier surface of Definition 156.3, the continuous-map dependency surface of Definition 156.4, the interval endpoint obligation surface of Definition 156.5, the classifier-stability transport by* hsame*, the finite* Cont *ledgers, and the* Pkg *provenance over* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket namecert obligation surface`

Proof. Collect the carrier and classifier definitions. They expose the endpoint continuous-map rows, the interval parameter row, the deformation row, the endpoint ledgers, the finite readback ledger, and the provenance row as the complete carrier-classifier surface.

Use Definition 156.4 to account for every continuous-map consumer. Use Definition 156.5 to account for the two interval endpoint consumers. These two projection theorems cover the dependency and exactness rows used by downstream homotopy and knot certificates.

Classifier stability is the stated hsame transport of the endpoint rows, interval row, and finite Cont ledgers. Since all admitted comparisons are generated by those transported rows and the retained Pkg provenance, there is no remaining public field through which a consumer could demand a stronger host-level homotopy object. □

156.2 The certificate

Theorem 156.7 ($\mathsf{Homotopy}^{\uparrow}$ endpoint classifier row)**.** *For every* $\mathsf{Homotopy}^{\uparrow}$ *source packet* $(f, g, H, I, \rho, \lambda)$ *of Definition 156.1, the two accepted endpoint faces classify only through the* $\mathsf{ContinuousMap}^{\uparrow}$ *classifier rows carried by* f *and* g*. The* $\mathsf{Homotopy}^{\uparrow}$ *classifier may read the endpoint rows, the* BHist *deformation row* H*, the interval row* I*, the* Pkg *provenance* ρ*, and the* Cont *ledger* λ*, but it does not compare endpoints by host maps, path quotients, or external equality.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket endpoint classifier row`

Proof. Project the endpoint faces from the displayed finite packet. The rows f and g already arrive with $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ provenance, so their admissibility and endpoint comparison are exactly the classifier fields of that dependency certificate.

The remaining rows only scope the readback. The BHist deformation row H and the $\mathsf{Interval}^{\uparrow}$ row I determine which endpoint reads are visible; ρ records their Pkg source; and λ records the finite Cont and hsame transport needed to expose those reads. None of these rows creates a second endpoint classifier. □

Theorem 156.8 ($\mathsf{Homotopy}^{\uparrow}$ interval-parameter transport)**.** *Let* $(f, g, H, I, \rho, \lambda)$ *be a* $\mathsf{Homotopy}^{\uparrow}$ *source packet. If the* $\mathsf{Interval}^{\uparrow}$ *parameter row* I *is transported to a row* I' *by* hsame *under* $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$*, and the endpoint read of* H *along* I *is recorded by the finite* Cont *ledger* λ*, then the transported endpoint read along* I' *is accepted by the same* $\mathsf{Homotopy}^{\uparrow}$ *classifier with the transported* Cont *ledger and the inherited* Pkg *provenance.*

Proof. Use the $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$ classifier to transport the parameter row from I to I'. The transport is an hsame row over BHist, so it does not replace the deformation row H or the endpoint rows f and g.

Next transport the endpoint readback ledger. The finite Cont entries in λ are reindexed along the displayed hsame parameter comparison, and the resulting endpoint row is classified by the same dependency reads from $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$. The Pkg provenance ρ is retained because the dependency certificates and displayed faces are unchanged. □

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket interval parameter transport`

Theorem 156.9 ($\mathsf{Homotopy}^{\uparrow}$ deformation-composition row)**.** *Suppose* $(f, g, H, I, \rho, \lambda)$ *and* $(g', k, H', I', \rho', \lambda')$ *are* $\mathsf{Homotopy}^{\uparrow}$ *source packets whose middle endpoint faces are classified by the* $\mathsf{ContinuousMap}^{\uparrow}$ *classifier carried by* g *and* g'*. Then their compatible deformation rows concatenate to a single* $\mathsf{Homotopy}^{\uparrow}$ *packet from* f *to* k*, with composed finite* Cont *ledger and inherited* Pkg *provenance from* ρ *and* ρ'*.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket deformation composition row`

Proof. First compare the middle faces using only the endpoint classifier supplied by $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$. This gives the BEDC compatibility row needed to identify the end of H with the start of H' through hsame, not through host-level equality of maps.

Now concatenate the two deformation ledgers. The Cont rows in λ and λ' compose as a finite BEDC ledger, and the middle hsame comparison is inserted at the junction. The interval rows I and I' remain the parameter faces licensed by $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$.

Finally repack the tuple. The new source packet has endpoints f and k, a concatenated deformation row, the displayed parameter rows, composed Cont ledger, and the combined Pkg provenance inherited from ρ and ρ'. These are exactly the rows allowed by Definition 156.1. □

Theorem 156.10 ($\mathsf{Homotopy}^\uparrow$ reversal-symmetry row)**.** *For every* $\mathsf{Homotopy}^\uparrow$ *source packet* $(f, g, H, I, \rho, \lambda)$*, reversing the displayed* $\mathsf{Interval}^\uparrow$ *row swaps the two endpoint faces and gives the symmetric* $\mathsf{Homotopy}^\uparrow$ *classifier row from* g *to* f*. The reversed packet uses the same* BHist *deformation data, the* hsame *symmetry rows from the endpoint and interval classifiers, the same* Pkg *provenance, and the reversed finite* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket reversal symmetry row`

Proof. Apply the symmetry field of the $\mathsf{NameCert}_{\mathsf{Interval}^\uparrow}$ classifier to the displayed interval row. This reverses the parameter face inside BHist and supplies the hsame transport that exchanges the two endpoint reads of H.

Then apply the symmetry field of $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ to the two endpoint classifier rows. The endpoint comparison is therefore reversed inside the continuous-map certificate rather than by an external path symmetry principle.

The final packet keeps the same provenance row ρ and reverses the finite Cont ledger λ. Since the carrier definition exports only BHist rows, hsame transports, Cont ledgers, Pkg provenance, and the two dependency certificates, the reversed tuple is a valid $\mathsf{Homotopy}^\uparrow$ classifier row from g to f. □

Theorem 156.11 ($\mathsf{Homotopy}^\uparrow$ constant-deformation reflexivity row)**.** *Every endpoint continuous-map row accepted by* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ *admits a constant deformation packet over the* $\mathsf{Interval}^\uparrow$ *endpoint ledger. The packet has the same displayed endpoint row at both interval faces, carries the same* BHist *deformation row through both reads, and gives the reflexive* $\mathsf{Homotopy}^\uparrow$ *classifier row using only the finite* Cont *ledger,* hsame *identity transport, and the retained* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket constant deformation reflexivity row`

Proof. Project the endpoint row through $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$. This supplies the accepted continuous-map classifier row that may appear at both faces of a $\mathsf{Homotopy}^\uparrow$ packet.

Choose the constant displayed deformation row whose two endpoint reads are the same projected row. The $\mathsf{Interval}^\uparrow$ endpoint ledger supplies the two interval faces, and both reads use the same finite Cont entry.

Finally repack the tuple with identity hsame transport between the two displayed endpoint rows. Since the carrier surface admits exactly the endpoint rows, interval row, deformation row, finite Cont ledger, and Pkg provenance, this is the reflexive $\mathsf{Homotopy}^\uparrow$ classifier row. □

Theorem 156.12 ($\mathsf{Homotopy}^\uparrow$ endpoint-composition transitivity row)**.** *Suppose compatible* $\mathsf{Homotopy}^\uparrow$ *packets classify* f *with* g *and* g *with* k*, and suppose the middle endpoint rows are classified by* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$*. Then the two packets compose to a* $\mathsf{Homotopy}^\uparrow$ *packet from* f *to* k*, with*

the middle hsame *row recorded in the finite ledger and with the endpoint classifiers inherited from the two continuous-map dependency rows.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket endpoint composition transitivity row`

Proof. Apply Definition 156.9 to the two compatible deformation packets. Its construction concatenates the displayed deformation rows and carries forward the Pkg provenance from both packets.

The compatibility hypothesis supplies the middle endpoint comparison inside $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$. Record that comparison as the inserted hsame row at the junction of the two finite Cont ledgers.

The repacked packet has endpoints f and k, the composed deformation row, and the composed ledger with the middle comparison made explicit. These are precisely the rows exposed by the $\mathsf{Homotopy}^{\uparrow}$ classifier surface, so the result is the transitive endpoint-composition row. □

Theorem 156.13 ($\mathsf{Homotopy}^{\uparrow}$ reversal classifier symmetry row)**.** *If a* $\mathsf{Homotopy}^{\uparrow}$ *packet classifies* f *with* g*, then reversing the interval parameter and transporting both endpoint classifiers gives the symmetric classifier row from* g *to* f*. The reversed packet uses the* $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$ *reversal row for the parameter, the* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *symmetry rows for the two endpoints, and the reversed finite* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket reversal classifier symmetry row`

Proof. Apply Definition 156.10 to the original packet. It reverses the displayed interval parameter, swaps the endpoint reads, retains the Pkg provenance, and reverses the finite Cont ledger.

The interval reversal is classified by $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$, while the two endpoint transports are classified by $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$. These symmetry fields provide the required hsame rows without appealing to an external equality of maps or paths.

Repacking those transported rows gives a valid $\mathsf{Homotopy}^{\uparrow}$ classifier row from g to f. Its public data are still only the reversed interval row, the swapped endpoint rows, the same deformation data, the reversed ledger, and the dependency certificate provenance. □

Theorem 156.14 ($\mathsf{Homotopy}^{\uparrow}$ identity deformation row)**.** *Every accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *endpoint row* f *determines a* $\mathsf{Homotopy}^{\uparrow}$ *source packet from* f *to itself. The packet uses the two* $\mathsf{Interval}^{\uparrow}$ *endpoint faces, the reflexive* hsame *row on the endpoint classifier, a finite constant* Cont *ledger for the displayed deformation, and the shared* Pkg *provenance carried by the accepted endpoint row.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket identity deformation row`

Proof. Unpack the carrier rows of Definition 156.1. The endpoint contribution is the single accepted $\mathsf{ContinuousMap}^{\uparrow}$ row f read twice, so the endpoint classifier supplies the reflexive hsame comparison without introducing any external equality of maps.

The interval contribution is the displayed pair of $\mathsf{Interval}^{\uparrow}$ endpoint faces. Over those faces the deformation ledger is constant at f, and its continuity witness is the finite Cont ledger obtained by reusing the accepted endpoint certificate at each displayed face.

Finally repack the data with the same provenance row. The tuple contains only BHist rows, the reflexive hsame comparison, the finite Cont ledger, the two dependency certificates, and shared Pkg provenance, so it is a valid $\mathsf{Homotopy}^{\uparrow}$ packet from f to f. □

Theorem 156.15 (Homotopy$^{\uparrow}$ classifier equivalence obligation)**.** *The* Homotopy$^{\uparrow}$ *classifier carries the reflexive, symmetric, and transitive rows required of an equivalence surface. Reflexivity is generated by Definition 156.14, symmetry by Definition 156.10, and transitivity by Definition 156.9, using only the* ContinuousMap$^{\uparrow}$ *classifier, the* Interval$^{\uparrow}$ *classifier,* hsame*,* Cont*, and* Pkg *rows already present in the source-packet definition.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket classifier equivalence obligation`

Proof. For reflexivity, take an accepted endpoint row and apply Definition 156.14. The resulting packet has the same endpoint on both faces and records the required classifier row entirely inside the BHist carrier.

For symmetry, start from any classifier packet and use Definition 156.10. The construction reverses the interval face, applies the available hsame symmetries, keeps the same Pkg provenance, and reverses the finite Cont ledger.

For transitivity, take two packets whose middle endpoint faces are classified by the ContinuousMap$^{\uparrow}$ classifier and apply Definition 156.9. The middle comparison is the displayed hsame row, and the composed packet records the concatenated finite Cont ledger with inherited provenance.

These three constructions exhaust the equivalence obligation because Definition 156.1 exposes no further classifier inputs: only endpoint certificates, interval certificates, hsame transports, finite Cont ledgers, and Pkg provenance are available. □

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyTasteGate single carrier alignment`

Theorem 156.16 (Homotopy$^{\uparrow}$ composition associativity row)**.** *For three composable* Homotopy$^{\uparrow}$ *deformation packets, the two bracketings of their pairwise composites are classified by the same* Homotopy$^{\uparrow}$ *row. The finite* Cont *ledgers reassociate, the two middle endpoint transports are the same* hsame *comparisons, and the resulting packet has the same endpoint faces and combined* Pkg *provenance.*

Proof. Unpack the three packets as source rows in Definition 156.1. Both bracketings use Definition 156.9 twice and therefore expose the same outer endpoint faces together with the same two displayed middle endpoint comparisons.

The only apparent difference is the placement of parentheses in the finite Cont ledger. Since the ledger is a finite BEDC row list, reassociation changes only the grouping of the repacking step; it does not create a new continuity obligation or a new endpoint transport.

The Pkg provenance is the combined provenance of the same three packets in both bracketings. Repacking those rows gives a single Homotopy$^{\uparrow}$ classifier row relating the two composites, with the middle faces witnessed by the same hsame comparisons used in the two applications of the composition row. □

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket composition associativity row`

Theorem 156.17 (Homotopy$^{\uparrow}$ double-reversal classifier row)**.** *Let*

$$\mathcal{H} = (f, g, H, I, \rho, \lambda)$$

be a Homotopy$^{\uparrow}$ *source packet of Definition 156.1. Let* $\mathcal{H}^{\mathrm{rev}}$ *be the reversed packet supplied by Definition 156.10, and let*

$$\mathcal{H}^{\mathrm{rr}} = (\mathcal{H}^{\mathrm{rev}})^{\mathrm{rev}}$$

be obtained by applying the same reversal construction a second time. Then $\mathcal{H}^{\mathrm{rr}}$ *is a* Homotopy$^{\uparrow}$ *source packet from* f *to* g *that classifies the same homotopy object as* $\mathcal{H}$*: its deformation row is*

hsame *to the original deformation row H, its interval-parameter row is* hsame *to I through the double interval-classifier transport, its endpoint* $\mathsf{ContinuousMap}^{\uparrow}$ *rows are* hsame *to f and g in their original order, its finite* Cont *ledger is the double reverse of λ and is therefore the same finite ledger order as λ, and its* Pkg *provenance is the original row ρ.*

Lean4 ✓ `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket double reversal classifier row`

Proof. Apply Definition 156.10 to $\mathcal{H}$. The resulting packet has endpoint order g, f, keeps the same displayed deformation data H, transports the interval row by the $\mathsf{NameCert}_{\mathsf{Interval}^{\uparrow}}$ symmetry row, keeps the same provenance row ρ, and replaces the finite ledger λ by its reversed ledger.

Apply Definition 156.10 again to this reversed packet. The endpoint order is exchanged a second time, so the visible endpoint rows return to f, g. The endpoint transports used in the two applications are the two symmetry rows of the $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ classifier; composed with the reflexive classifier rows at f and at g, they give the required hsame comparisons with the original endpoint rows.

The deformation coordinate is never replaced by either reversal construction. Each application only changes the interval reading of the same displayed row H, so the double-reversed deformation coordinate is classified with H by the reflexive hsame row on that displayed BHist deformation data.

For the interval coordinate, the first reversal uses the interval symmetry row that reads the endpoint ledger in the opposite direction. The second reversal applies the same interval-classifier symmetry to the reversed parameter row. By the transitivity and symmetry rows of the interval classifier, the composite double transport classifies the resulting parameter row with the original row I. This is precisely the interval component compared by the Homotopy classifier surface of Definition 156.3.

For the ledger coordinate, the first construction reverses the finite Cont ledger λ, and the second construction reverses that finite ledger again. Since a finite ledger is a displayed BEDC row list, reversing the row list twice restores the original row order and the same endpoint-read entries. Thus the double-reversed ledger is componentwise hsame to λ.

Both reversal steps retain the provenance row rather than producing a new package scope, so the provenance of $\mathcal{H}^{\mathrm{rr}}$ is the same ρ. The classifier surface of Definition 156.3 compares exactly the endpoint rows, interval parameter row, deformation readback ledger, and provenance just identified. Repacking these componentwise comparisons gives a $\mathsf{Homotopy}^{\uparrow}$ source packet from f to g that classifies the same homotopy object as $\mathcal{H}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$**.**

Constructive story: A homotopy is assembled as a finite BHist packet containing two endpoint $\mathsf{ContinuousMap}^{\uparrow}$ rows, an $\mathsf{Interval}^{\uparrow}$ parameter row, a displayed deformation row between the endpoints, endpoint-face ledgers, a finite Cont readback ledger, and Pkg provenance for the two dependency certificates. The carrier accepts only packets whose endpoint maps, interval faces, deformation row, endpoint reads, and provenance occur in the same displayed envelope, and it excludes a host path space, imported path equality, ambient function equality, or quotient of maps. Two accepted envelopes classify together when hsame transports the endpoint continuous-map rows, the interval parameter row, and the finite ledgers reading the deformation endpoints, so classification is deformation data with endpoint and interval transport rather than equality in an external mapping space. The source specification is the endpoint-deformation-interval packet, the pattern records endpoint determinacy, reversal, constant deformation, composition, identity, and associativity rows, and the ledger keeps the raw deformation row aligned with the two visible endpoint reads. The five naming-certificate obligations are obtained by the identity deformation for inhabitance and reflexivity, reversal for symmetry, composition for transitivity, and the source-envelope transport rows for closure of accepted packets under the classifier. A history h lies in the signature gap of a $\mathsf{Homotopy}^{\uparrow}$ token p when the source domain accepts the deformation packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; in this surface, that says the two endpoint maps, interval parameter, deformation row, endpoint reads, and composed ledger of h are the packet

encoded by p. The chapter registers this finite-deformation construction under Homotopy$^{\uparrow}$; it uses only ContinuousMap$^{\uparrow}$, Interval$^{\uparrow}$, BHist, hsame, Cont, and Pkg rows, with no host path quotient or full homotopy theory added.

Theory closure: TheoryGate(NameCert$_{\text{Homotopy}\uparrow}$, maturePackageClosed)

Scope closed: The exported finite-deformation public package consists of the BHist source packet, source envelope, endpoint classifier, interval endpoint determinacy, reversal, constant deformation, composition, identity, classifier equivalence, associativity, finite Cont readback, hsame transport, and Pkg provenance rows over NameCert$_{\text{ContinuousMap}\uparrow}$ and NameCert$_{\text{Interval}\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{Homotopy}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.HomotopyUp.HomotopyBHistSourcePacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close host path quotients or full homotopy theory.

Upgrade path: No further paper theory-axis upgrade is recorded; only bridgeChecked verification remains on the separate verification axis.

Chapter 157

A Concrete Naming Certificate for $\mathsf{SimplicialSet}^{\uparrow}$

The simplicial-set interface $\mathsf{SimplicialSet}^{\uparrow}$ packages a contravariant functor from the simplex category Δ to a certified set carrier, with face and degeneracy operators satisfying the simplicial identities. It depends on $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ (chapter 62) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

157.1 Carrier definition

Definition 157.1 (SimplicialSet BHist simplex-row carrier)**.** *Fix a certified simplex-category source supplied by* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ *and finite simplex-row sources supplied by* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$*. A* $\mathsf{SimplicialSet}^{\uparrow}$ *BHist simplex-row carrier is a finite displayed packet*

$$(F, X, D, P, \rho, \lambda)$$

with the following fields.

1. *F is the functor-source row: it consumes the* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ *presentation of the contravariant source from the simplex category and exposes only its object, morphism, identity, and composition boundary.*
2. *X is the finite family of simplex rows: for each displayed simplex degree, the row is consumed through the* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *carrier and classifier, so simplex entries are finite BHist rows rather than host-side sets.*
3. *D is the face-and-degeneracy endpoint ledger: every displayed coface or codegeneracy arrow has named source and target simplex rows, with the endpoint read visible in* BHist*.*
4. *P is the package row that records which functor, finite-row, and endpoint reads are public at the certificate boundary.*
5. *ρ is the* Pkg *provenance ledger for the reads in F, X, and D.*
6. *λ is the continuation ledger: its entries are* Cont *routes and* hsame *transports showing that functor-source reads, finite simplex rows, face endpoints, degeneracy endpoints, and package provenance remain in the same displayed packet after transport.*

The carrier exposes the finite packet and its displayed ledgers only. It does not expose a host presheaf, a host simplex category, or host equality of face and degeneracy maps.

Definition 157.2 (SimplicialSet face-degeneracy classifier)**.** *For two* $\mathsf{SimplicialSet}^{\uparrow}$ *BHist simplex-row carriers*

$$(F, X, D, P, \rho, \lambda)$$

and

$$(F', X', D', P', \rho', \lambda'),$$

the face-degeneracy classifier relates them when the following componentwise transports are displayed.

1. *The functor-source rows* F *and* F' *are related by the* hsame *transport supplied by the* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ *source.*
2. *The finite simplex-row families* X *and* X' *are related degreewise by the* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *carrier and classifier rows.*
3. *The face and degeneracy ledgers* D *and* D' *carry matching source and target endpoints after the functor-source and finite-row transports.*
4. *The package rows* P *and* P' *have matching* Pkg *provenance entries through* ρ *and* ρ'*.*
5. *The continuation ledgers* λ *and* λ' *transport the same* Cont *routes and* hsame *witnesses for all functor, finite-row, endpoint, and provenance reads.*

Thus classification is componentwise hsame *transport of the functor source, finite simplex rows, face and degeneracy ledgers, provenance ledger, and continuation ledger.*

157.2 The certificate

Theorem 157.3 (SimplicialSet functor-source boundary)**.** *For a* $\mathsf{SimplicialSet}^{\uparrow}$ *BHist simplex-row carrier of Definition 157.1, the visible functor-source boundary is exactly the* F *component together with the endpoint rows in* D *that read its displayed face and degeneracy arrows. The* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ *source supplies the object, morphism, identity, and composition rows consumed by* F*, while the finite simplex rows and package provenance remain separate carrier fields.*

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier functor source boundary`

Proof. Unpack the packet $(F, X, D, P, \rho, \lambda)$ and project the first component. By definition, F is the only field that consumes the $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ source, so its public reads are the object, morphism, identity, and composition boundary of that source.

Next project D. Its entries name the source and target simplex rows for each displayed face or degeneracy endpoint, so the endpoint boundary for the functor read is visible without merging it with the finite-row family X.

The remaining fields are ledgers rather than functor-source data. The package row P, provenance row ρ, and continuation row λ record how the same displayed reads remain inside the packet, while the finite simplex family X is consumed through $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$. Repacking these projections gives precisely the stated boundary. □

Theorem 157.4 (SimplicialSet face-degeneracy ledger stability)**.** *Let* $S = (F, X, D, P, \rho, \lambda)$ *and* $S' = (F', X', D', P', \rho', \lambda')$ *be* $\mathsf{SimplicialSet}^{\uparrow}$ *BHist simplex-row carriers. If* S *and* S' *are related by the classifier of Definition 157.2, then every displayed face or degeneracy endpoint in* D *transports to the corresponding endpoint in* D' *through the functor-source* hsame *row, the degreewise* $\mathsf{FinSet}^{\uparrow}$ *classifier, and the continuation ledger. The transported endpoint remains recorded with matching* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetFaceDegeneracyClassifier ledger stability`

Proof. Start from one displayed endpoint in D. The classifier supplies the comparison between F and F', so the functor-source read carrying the corresponding face or degeneracy arrow transports by hsame.

Use the degreewise finite-row comparison between X and X' to transport the named source and target simplex rows of that endpoint. This step consumes the $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ carrier and classifier rows and does not choose a host-side representative of the simplex set.

Finally read λ and λ'. Their continuation entries identify the same Cont route after the functor and finite-row transports, while ρ and ρ' match the package provenance. The transported endpoint therefore repacks as the corresponding entry of D' with the same public ledger boundary. □

Theorem 157.5 (SimplicialSet NameCert obligation surface)**.** *The carrier of Definition 157.1, the classifier of Definition 157.2, the functor-source boundary of Definition 157.3, and the ledger stability of Definition 157.4 assemble the public obligation surface expected of* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$*. The surface exposes exactly the functor-source row F, the finite simplex-row family X, the face-and-degeneracy endpoint ledger D, the package row P, the provenance ledger ρ, and the continuation ledger λ.*

Proof. First take the carrier packet from Definition 157.1. It lists every visible field consumed by the simplicial-set certificate: functor source, finite simplex rows, endpoint ledger, package row, provenance ledger, and continuation ledger.

Next apply the classifier of Definition 157.2. It supplies componentwise hsame transport for the functor source, degreewise finite-row transport, endpoint-ledger transport, provenance matching, and continuation matching.

Apply Definition 157.3 to separate the functor-source boundary from the finite-row and package ledgers. Apply Definition 157.4 to keep every displayed face and degeneracy endpoint stable under the classifier.

The four clauses repack as the public $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ obligation surface. No additional host presheaf, host simplex category, host function equality, or unrecorded simplicial-identity proof is exported outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier namecert obligation surface`

Theorem 157.6 (SimplicialSet face-degeneracy endpoint coverage)**.** *For every* $\mathsf{SimplicialSet}^{\uparrow}$ *BHist simplex-row carrier $(F, X, D, P, \rho, \lambda)$ of Definition 157.1, each displayed face or degeneracy endpoint in D is covered by the functor-source row F, the finite simplex-row family X, and the finite* Cont *and* Pkg *ledgers. The source and target simplex rows are read through X, the displayed arrow is read through F, and the endpoint remains public only through the continuation and provenance rows recorded by λ and ρ.*

Proof. Project a displayed endpoint entry from D. Its arrow component is one of the face or degeneracy arrows carried by the functor-source row F, so the source-category reading is already inside the $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$ boundary.

Next project the source and target simplex rows named by that endpoint. They are entries of the finite family X, hence are consumed through the $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ carrier and classifier surface.

Finally read the ledgers. The package provenance row ρ records the dependency origin for the functor and finite-row reads, while λ records the finite Cont route that exposes the endpoint in the

packet. Repacking these projections gives endpoint coverage without introducing a host presheaf endpoint. □

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier face degeneracy endpoint coverage`

Theorem 157.7 (SimplicialSet simplicial-identity ledger coverage)**.** *Every identity or composition law for displayed faces and degeneracies in a* $\mathsf{SimplicialSet}^{\uparrow}$ *carrier is consumed as a finite* BHist *ledger row inside* D*,* ρ*, and* λ*. The public certificate uses those ledger rows to compare endpoints, and it does not require host presheaf equality of face maps, degeneracy maps, or their composites.*

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier simplicial identity ledger coverage`

Proof. Start from an identity or composition law appearing in the displayed face-and-degeneracy data. The law is not read as equality in an external presheaf; it is a named row in the same BHist packet as the endpoint entries.

Use the classifier of Definition 157.2. It transports the functor-source row, the finite simplex-row family, and the endpoint ledger by componentwise hsame witnesses.

The continuation ledger then identifies the two displayed routes that the simplicial identity compares, while the package provenance records that both routes belong to the same public packet. Thus the identity is consumed as ledger coverage, not as a host-side equation of presheaf maps. □

Theorem 157.8 (SimplicialSet public dependency scope)**.** *The public* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *certificate depends exactly on* $\mathsf{NameCert}_{\mathsf{Functor}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$*,* BHist*,* hsame*,* Cont*, and* Pkg*. Its public carrier, classifier, endpoint coverage, and simplicial-identity rows are obtained by projection and repacking of those dependencies alone.*

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier public dependency scope`

Proof. Read the carrier of Definition 157.1. The row F is the only functor-source dependency, and the row X is the only finite simplex-row dependency.

The remaining public objects are kernel ledgers. Endpoints and identity rows are BHist entries, comparisons are hsame transports, continuations are finite Cont routes, and dependency provenance is recorded by Pkg.

Apply Definition 157.5, Definition 157.6, and Definition 157.7. These rows exhaust the public surface, so any dependency outside the listed objects would have no projection in the certificate. □

Theorem 157.9 (SimplicialSet face endpoint scope binding)**.** *Each displayed face or dependency endpoint in a public* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *packet is bound to the displayed* $\mathsf{Functor}^{\uparrow}$ *source row, the* $\mathsf{FinSet}^{\uparrow}$ *simplex-row source, the finite* BHist *endpoint ledger, the carried* hsame *transports, the finite* Cont *continuation rows, and the shared* Pkg *provenance. The binding exports no host presheaf endpoint and no equality of ambient face maps beyond those rows.*

Proof. Project the face endpoint through Definition 157.6. The source side is the functor row F together with the finite simplex-row family X, so the endpoint has no independent host presheaf carrier.

Use the classifier of Definition 157.2. Its comparison rows are precisely the hsame transports carried by the same packet.

The continuation and provenance fields then bind the endpoint to the finite Cont route and the shared Pkg row. Repacking these projections gives the stated dependency scope for every displayed face endpoint. □

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier face endpoint scope binding`

Theorem 157.10 (SimplicialSet identity ledger scope binding)**.** *Every simplicial identity row used by the public* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *certificate is bound as a finite* BHist *ledger row over the displayed* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{FinSet}^{\uparrow}$ *dependencies, with comparison only through* hsame*,* Cont*, and* Pkg*. The certificate does not require host equality of presheaves, face maps, degeneracy maps, or their composites.*

Proof. Start from an identity row accepted by the public packet. By Definition 157.7, the row is already a finite ledger entry, not an equation in an ambient presheaf.

The endpoints of the two displayed routes are bound by Definition 157.9. Thus both sides of the identity read the same $\mathsf{Functor}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, and Pkg surface.

Repack the two route reads with the ledger row. This gives scope binding for the identity while excluding any separate host presheaf equality or external simplicial-law witness. □

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier identity ledger scope binding`

Theorem 157.11 (SimplicialSet scoped obligation package)**.** *The scoped* $\mathsf{SimplicialSet}^{\uparrow}$ *obligation package is exactly the package generated by the public dependency scope of Definition 157.8, the face endpoint scope binding of Definition 157.9, and the identity ledger scope binding of Definition 157.10. It binds the carrier, classifier, endpoint, and simplicial-identity rows to* $\mathsf{Functor}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* Pkg*, and it exports no additional host presheaf structure.*

Proof. First take the public dependency inventory from Definition 157.8. It lists the only source and kernel objects projected by the public certificate.

Next attach Definition 157.9. This binds every face or dependency endpoint row to the displayed dependency scope.

Finally attach Definition 157.10. This binds the simplicial identity rows to finite ledgers over the same dependency scope. Together the three projections exhaust the carrier, classifier, endpoint, and identity surfaces, so the scoped package has no field through which a host presheaf equality or external simplicial structure could enter. □

Lean4 ✓ `BEDC.Derived.SimplicialSetUp.SimplicialSetTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$**.**

Constructive story: A simplicial set is generated as a finite BHist packet whose functor-source row consumes the $\mathsf{Functor}^{\uparrow}$ presentation of the simplex-category direction, whose finite simplex-row family is consumed through $\mathsf{FinSet}^{\uparrow}$, and whose face and degeneracy endpoints are recorded by displayed ledgers. The carrier accepts packets containing the functor source, finite simplex rows, face-and-degeneracy endpoint ledger, package row, provenance ledger, and continuation ledger, while a host presheaf, host simplex category, ambient face-map equality, and unrecorded simplicial-identity proof stay outside. Two packets classify together when the functor-source row, each finite simplex row, each face or degeneracy endpoint, the provenance row, and the continuation ledger transport componentwise by the available hsame and dependency classifiers. The source specification is this six-field packet, the pattern records the contravariant functor boundary, face and degeneracy endpoint coverage, and simplicial-identity ledger rows, and the ledger keeps raw functor reads, finite simplex entries, endpoint routes, and provenance visible together. The five naming-certificate obligations are supplied by an admitted finite packet, by componentwise reflexivity, symmetry, and transitivity of the transported rows, and by ledger stability showing that every transported face or degeneracy endpoint remains in the carrier. A history h lies

in the signature gap of a SimplicialSet$^{\uparrow}$ token p when the source domain accepts this simplex-row packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; concretely, the degrees, finite simplex entries, face and degeneracy routes, and identity ledgers of h are the rows encoded by p. The chapter registers this displayed functor-and-ledger construction under SimplicialSet$^{\uparrow}$; it closes only the scoped Functor$^{\uparrow}$, FinSet$^{\uparrow}$, BHist, hsame, Cont, and Pkg surface, with no standard bridge or full simplicial homotopy theory introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{SimplicialSet}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature SimplicialSet$^{\uparrow}$ paper scope covers the BHist simplex-row carrier, face-degeneracy endpoint ledger, simplicial-identity ledger, public dependency scope, and displayed simplex-row bridge surface over Functor$^{\uparrow}$, FinSet$^{\uparrow}$, BHist, hsame, Cont, and Pkg.

Formal verification: FormalStatus(NameCert$_{\mathsf{SimplicialSet}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SimplicialSetUp.SimplicialSetBHistSimplexRowCarrier\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: Full simplicial homotopy theory and standard bridge data are not closed by this chapter.

Upgrade path: No further paper theory-axis step is claimed; verification-axis bridge work remains separate.

Chapter 158

A Concrete Naming Certificate for $\mathsf{ModelCat}^{\uparrow}$

The model-category interface $\mathsf{ModelCat}^{\uparrow}$ packages a category carrier with three distinguished classes of morphisms (cofibrations, fibrations, weak equivalences) satisfying lifting and factorisation axioms. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61), and is the homotopy-theoretic substrate beneath higher-category certificates.

158.1 Carrier definition

Definition 158.1 (ModelCatUp BHist source packet)**.** *A* $\mathsf{ModelCat}^{\uparrow}$ BHist *source packet is a finite tuple*

$$\mathcal{M} = (C, \mathsf{Cof}, \mathsf{Fib}, \mathsf{W}, \mathsf{Lift}, \mathsf{Fact}, P, \rho, \lambda)$$

whose coordinates are all displayed BEDC rows:

1. *C is the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row, carrying the object, morphism, identity, composition, and morphism-classifier surface.*
2. Cof, Fib, *and* W *are the cofibration, fibration, and weak-equivalence classifier rows on morphisms of C.*
3. Lift *records the finite lifting ledgers between cofibration rows and fibration rows, including the acyclic cases whose weak-equivalence coordinates are read through* W.
4. Fact *records the finite factorisation ledgers that split each displayed morphism through a cofibration followed by an acyclic fibration and through an acyclic cofibration followed by a fibration.*
5. *P is the* Pkg *provenance row for the CategoryUp dependency and for the displayed classifier ledgers.*
6. *ρ and λ are the* $\mathsf{Cont}/\mathsf{hsame}$ *transport ledgers that keep the category row, the three morphism-classifier rows, the lifting rows, the factorisation rows, and the package provenance stable under rowwise transport.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatTasteGate single carrier alignment`

Theorem 158.2 (ModelCatUp root obligation surface)**.** *An admitted* $\mathsf{ModelCat}^{\uparrow}$ *source packet in the sense of Definition 158.1 exposes exactly the finite* BHist *rows C,* Cof, Fib, W, Lift, Fact, *P, ρ, and λ. Consequently the root obligation surface for* $\mathsf{NameCert}_{\mathsf{ModelCat}^{\uparrow}}$ *consists of the consumed*

$\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ *row, the cofibration row, the fibration row, the weak-equivalence row, the lifting and factorisation ledgers, the* Pkg *provenance, and the* Cont/hsame *transport ledger. It does not expose a host localization, a quotient category, an external weak-equivalence calculus, a derived mapping space, or a bridge row.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket root obligation surface`

Proof. Project an admitted packet $\mathcal{M} = (C, \mathsf{Cof}, \mathsf{Fib}, \mathsf{W}, \mathsf{Lift}, \mathsf{Fact}, P, \rho, \lambda)$. By Definition 158.1, these nine coordinates are the only carrier fields. The category coordinate is already a $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ row, so object, morphism, identity, composition, and morphism-classifier reads pass through that dependency row rather than through an ambient host category.

The next three projections give the displayed cofibration, fibration, and weak-equivalence classifier rows. The lifting projection reads only finite squares whose left and right morphism faces are classified by those rows, and the factorisation projection reads only the two displayed factorisation ledgers. Acyclic clauses use the same weak-equivalence coordinate W; they do not open a separate localization calculus.

Transport is rowwise. If a consumer replaces a displayed coordinate by an hsame-classified representative, the ρ and λ ledgers carry the CategoryUp dependency, the three classifier rows, the lifting rows, the factorisation rows, and the Pkg provenance to the transported packet.

Repacking these projections gives exactly the stated root obligation surface. Since the packet has no coordinate for a host localization, quotient category, external weak-equivalence calculus, derived mapping space, or bridge datum, none of those objects can be consumed from the $\mathsf{ModelCat}^\uparrow$ source surface. □

Theorem 158.3 (ModelCatUp weak-equivalence classifier laws)**.** *For an admitted* $\mathsf{ModelCat}^\uparrow$ *source packet* $\mathcal{M} = (C, \mathsf{Cof}, \mathsf{Fib}, \mathsf{W}, \mathsf{Lift}, \mathsf{Fact}, P, \rho, \lambda)$*, the displayed weak-equivalence row* W *supplies the reflexive, symmetric, and transitive classifier transport available to* $\mathsf{NameCert}_{\mathsf{ModelCat}^\uparrow}$ *consumers. These laws are read only through the category dependency row* C*, the* hsame *transport ledger, and the* Pkg *provenance row* P*; they do not import a host localization, a quotient category, or an external weak-equivalence calculus.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket weak equivalence classifier laws`

Proof. Project the admitted packet and read its weak-equivalence coordinate W. By Definition 158.1, this coordinate is a displayed morphism-classifier row over the consumed $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ dependency C, not a row in an ambient homotopy category.

The reflexive, symmetric, and transitive classifier movements are therefore the movements internal to W. When a morphism row is replaced by an hsame-classified representative, the transport ledgers ρ and λ carry the category row, the weak-equivalence classifier, and the package provenance together.

Repack these projected fields as the public classifier law surface. Since the packet contains no coordinate for localization, quotienting, or derived mapping-space data, the resulting laws are exactly the displayed W-through-hsame and Pkg laws. □

Theorem 158.4 (ModelCatUp lifting square boundary)**.** *Every lifting obligation exposed by an admitted* $\mathsf{ModelCat}^\uparrow$ *source packet reads only the cofibration row* Cof*, the fibration row* Fib*, the weak-equivalence row* W *for acyclic cases, the finite lifting ledger* Lift*, the rowwise* Cont/hsame *transport ledger, and the* Pkg *provenance. Thus a lifting square has no projection to a host localization, quotient category, external weak-equivalence calculus, derived mapping space, or bridge row.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket lifting square boundary`

Proof. Project an admitted packet. The left face of a displayed lifting square is read through Cof, and the right face is read through Fib. Acyclic variants use the same weak-equivalence coordinate W named in the packet.

The finite witness that the square is a lifting obligation is exactly the ledger coordinate Lift. Transport along an hsame replacement uses ρ and λ to move the category row, the three classifier rows, the lifting ledger, and the provenance row together.

Repacking these rows gives the claimed boundary. No other coordinate occurs in Definition 158.1, so no consumer of the lifting surface can read a host localization, quotient category, external calculus, derived mapping space, or bridge datum. □

Theorem 158.5 (ModelCatUp factorization ledger determinacy)**.** *For an admitted* $\mathsf{ModelCat}^{\uparrow}$ *source packet, the cofibration–acyclic-fibration factorisation route and the acyclic-cofibration–fibration factorisation route are determined by the finite ledger* Fact *together with the displayed* Cof*,* Fib*, and* W *classifier rows, the* Cont/hsame *transport ledger, and the* Pkg *provenance. The determinacy surface cannot import a host localization, quotient category, derived mapping space, external weak-equivalence calculus, or bridge row.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket factorization ledger determinacy`

Proof. Project the factorisation coordinate Fact from the admitted packet. Its first component records the cofibration followed by acyclic fibration route; its second component records the acyclic cofibration followed by fibration route.

Each classifier read needed by those routes is already one of Cof, Fib, or W. Acyclicity is therefore a displayed weak-equivalence classifier read, not a localization or quotient construction.

Transport is rowwise. The ledgers ρ and λ carry Fact, the three classifier rows, the CategoryUp dependency, and the provenance row under Cont/hsame replacement.

Repack the transported tuple as the public factorisation determinacy surface. Because the packet has no field for any host localization, quotient category, derived mapping space, external weak-equivalence calculus, or bridge datum, the two factorisation routes are determined exactly by the displayed ledger and classifier rows. □

Theorem 158.6 (ModelCatUp finite factorization transport surface)**.** *Let* $\mathcal{M} = (C, \mathsf{Cof}, \mathsf{Fib}, \mathsf{W}, \mathsf{Lift}, \mathsf{Fact}, P, \rho, \lambda)$ *be an admitted* $\mathsf{ModelCat}^{\uparrow}$ BHist *source packet. Suppose* $\mathcal{M}'$ *is a transported presentation whose displayed category row, cofibration classifier, fibration classifier, weak-equivalence classifier, lifting ledger, factorisation ledger, provenance row,* Cont/hsame *transport rows, and* Pkg *rows are preserved by the packet transport.*

Then $\mathcal{M}'$ *exposes the same finite factorisation and lifting obligation surface as* $\mathcal{M}$*. The transported surface consists exactly of the displayed category dependency, the three morphism-classifier rows, the finite lifting and factorisation ledgers, rowwise* Cont/hsame *transport, and the* Pkg *provenance. It contains no host localization, quotient category, derived mapping space, standard bridge row, or external weak-equivalence calculus.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket finite factorization transport surface`

Proof. Project $\mathcal{M}$ by Definition 158.1. The only available coordinates are the category dependency, the three classifier rows, the finite lifting and factorisation ledgers, the provenance row, and the rowwise transport ledgers.

Apply Definition 158.4 to the transported presentation. Preservation of C, Cof, Fib, W, Lift, P, and the Cont/hsame ledgers carries every displayed lifting square to the same finite obligation surface.

Apply Definition 158.5 to the same transported packet. Preservation of Fact together with Cof, Fib, W, P, and rowwise transport fixes both factorisation routes as displayed finite ledger rows.

Repack the lifted and factorised projections as the public transport surface. Since neither theorem projects a coordinate outside Definition 158.1, the transported presentation has no source from which to read localization, quotienting, derived mapping spaces, a standard bridge, or an external weak-equivalence calculus. □

158.2 The certificate

Definition 158.7 (ModelCatUp NameCert obligation package)**.** *The* $\mathsf{ModelCat}^{\uparrow}$ NameCert *obligation package is the ten-field package whose source is the* BHist *source packet of Definition 158.1. Its pattern is the displayed* $\mathsf{Cat}^{\uparrow}$ *dependency together with* Cof*,* Fib*,* W*,* Lift*, and* Fact*. Its classifier is the weak-equivalence row transported by* Cont/hsame*. Its stability fields are the classifier laws of Definition 158.3, the lifting boundary of Definition 158.4, and the factorisation determinacy of Definition 158.5. Its ledger field is P together with the displayed lifting and factorisation ledgers.*

Lean4 (definition) `BEDC.Derived.ModelCatUp.ModelCatNameCertObligationPackage`

Theorem 158.8 (ModelCatUp NameCert obligation surface)**.** *Every admitted* $\mathsf{ModelCat}^{\uparrow}$ NameCert *obligation package exposes exactly the* $\mathsf{Cat}^{\uparrow}$ *row, the cofibration classifier row, the fibration classifier row, the weak-equivalence classifier row, the lifting ledger, the factorisation ledger, package provenance, and rowwise* Cont/hsame *transport. It exposes no host localization, quotient category, derived mapping space, or external weak-equivalence calculus.*

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket namecert obligation surface`

Proof. Begin with the root obligation surface of Definition 158.2. It enumerates the only carrier fields available from an admitted source packet: the $\mathsf{Cat}^{\uparrow}$ dependency, the three morphism-classifier rows, the two finite ledgers, the package provenance, and rowwise transport.

Apply the weak-equivalence classifier laws, the lifting square boundary, and the factorisation ledger determinacy. These three results identify the stability supplied to the obligation package: classifier transport is read through W, lifting reads through Lift and the displayed classifier rows, and factorisation reads through Fact and the same transport ledgers.

Repack those projections as the ten-field package of Definition 158.7. Since the source packet has no coordinate for localization, quotienting, derived mapping spaces, or an external weak-equivalence calculus, none of those objects can appear in the obligation surface. □

Theorem 158.9 (ModelCatUp CategoryUp dependency boundary)**.** *Every* $\mathsf{ModelCat}^{\uparrow}$ *consumer reads objects, morphisms, identities, and composition only through the C coordinate of the source packet in Definition 158.1. The* $\mathsf{ModelCat}^{\uparrow}$ *package has no ambient category field outside that dependency coordinate.*

Proof. Project an admitted source packet $\mathcal{M} = (C, \mathsf{Cof}, \mathsf{Fib}, \mathsf{W}, \mathsf{Lift}, \mathsf{Fact}, P, \rho, \lambda)$. The first coordinate is the consumed $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ row, and Definition 158.1 assigns object, morphism, identity, composition, and morphism-classifier reads to that coordinate.

All remaining coordinates classify morphism classes, lifting ledgers, factorisation ledgers, provenance, and rowwise transport over the same dependency. They do not introduce a second category carrier.

Thus any consumer that asks for categorical structure must project C first. Because no ambient category field occurs in the packet, there is no alternative source from which objects, morphisms, identities, or composition can be read. □

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket category dependency boundary`

Theorem 158.10 (ModelCatUp root-threshold obligation triad)**.** *Every admitted* $\mathsf{ModelCat}^{\uparrow}$ *source packet supplies exactly three downstream root surfaces. The first is the consumed* $\mathsf{Cat}^{\uparrow}$ *dependency boundary, read through the* C *coordinate. The second is the three morphism-classifier rows, together with weak-equivalence classifier transport through* W*. The third is the finite lifting and factorisation ledger surface, transported by the displayed* $\mathsf{Cont/hsame}$ *rows and by* Pkg *provenance.*

These three surfaces are the only rows available to downstream consumers of the root packet. The packet exposes no host localization, quotient category, derived mapping space, external weak-equivalence calculus, or bridge row.

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket root threshold obligation triad`

Proof. Project an admitted packet by Definition 158.1. Its coordinates are the consumed $\mathsf{Cat}^{\uparrow}$ dependency, the three classifier rows, the two finite ledgers, package provenance, and rowwise $\mathsf{Cont/hsame}$ transport.

The carrier part is the root surface of Definition 158.2. The classifier part is transported by Definition 158.3, which isolates the weak-equivalence classifier as the only weak-equivalence row exposed by the packet.

For the finite ledger surface, apply Definition 158.4 to the lifting coordinate and Definition 158.5 to the factorisation coordinate. Both results read their data through the displayed classifier rows, package provenance, and rowwise transport.

It remains to identify the threshold through which categorical structure can be consumed. By Definition 158.9, objects, morphisms, identities, and composition are forced through the C coordinate. Since no coordinate for localization, quotienting, derived mapping spaces, external weak-equivalence calculus, or bridges occurs in the source packet, none of those rows is available downstream. □

Theorem 158.11 (ModelCatUp public certificate boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{ModelCat}^{\uparrow}}$ *surface is exactly the* $\mathsf{Cat}^{\uparrow}$ *dependency coordinate, the cofibration, fibration, and weak-equivalence classifier rows, the finite lifting-square ledger, the finite factorisation ledger,* Pkg *provenance, and rowwise* $\mathsf{Cont/hsame}$ *transport. It excludes host localization, quotient category, derived mapping space, and any external weak-equivalence calculus.*

Proof. Project the admitted source packet through Definition 158.2. This gives the consumed $\mathsf{Cat}^{\uparrow}$ coordinate, the three classifier rows, the lifting and factorisation ledgers, provenance, and rowwise transport.

Use Definition 158.3 to isolate the weak-equivalence classifier among the three displayed classifier rows. The cofibration and fibration rows remain parallel classifier coordinates rather than host localization data.

Apply Definition 158.4, Definition 158.5, and Definition 158.6. These identify the finite lifting-square and factorisation surfaces and their allowed transport through Pkg, Cont, and hsame.

Finally combine Definition 158.9 with Definition 158.10. The category dependency forces ordinary categorical reads through C, while the triad states that the dependency, classifier, and finite-ledger surfaces are the only downstream root surfaces. Thus the public certificate boundary is exactly the displayed finite surface and contains no localization, quotient category, derived mapping space, or external weak-equivalence calculus. □

Lean4 ✓ `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket public certificate boundary`

Theorem 158.12 (ModelCat finite model-structure bridge boundary)**.** *The public* $\mathsf{ModelCat}^{\uparrow}$ *finite packet interprets the standard finite model-category schema only through the consumed* $\mathsf{Cat}^{\uparrow}$ *row, the displayed cofibration, fibration, and weak-equivalence classifiers, the finite lifting ledger, the finite factorisation ledger,* Pkg *provenance,* Cont *routes, and* hsame *transport.*

Thus every bridge-facing read of the finite model-structure surface is already one of the rows certified by the public packet. The surface contains no host localization, quotient category, derived mapping space, external weak-equivalence calculus, or unrecorded factorisation choice.

Proof. Project the public certificate boundary of Definition 158.11. It identifies the consumed $\mathsf{Cat}^{\uparrow}$ dependency row, the three displayed morphism classifiers, the lifting and factorisation ledgers, provenance, and rowwise Cont/hsame transport as the public finite packet.

The weak-equivalence component is isolated by Definition 158.3. The cofibration and fibration components remain the two parallel classifier rows named by the root obligation surface, not external model-structure data.

The lifting and factorisation surfaces are fixed by Definition 158.4, Definition 158.5, and Definition 158.6. These results show that bridge-facing reads of lifting squares and factorisation rows pass through the displayed finite ledgers with their Pkg, Cont, and hsame provenance.

Finally, Definition 158.9 forces ordinary categorical reads through the consumed category row, and Definition 158.10 exhausts the remaining root surfaces. Hence the finite bridge boundary is exactly the public packet surface and exports none of the excluded host data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModelCat}^{\uparrow}}$.

Constructive story: A ModelCat packet is built from a consumed $\mathsf{Cat}^{\uparrow}$ row, three displayed morphism-classifier rows, finite lifting and factorisation ledgers, a package row, provenance, and rowwise Cont/hsame transport. The checked category dependency boundary forces ordinary categorical reads through the consumed dependency row, while the root, classifier, lifting, factorisation, and threshold surfaces expose only the displayed finite packet rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModelCat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature $\mathsf{ModelCat}^{\uparrow}$ paper scope covers the BHist source packet, weak-equivalence classifier laws, lifting square boundary, factorization ledger determinacy, public certificate boundary, and finite model-structure bridge boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModelCat}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ModelCatUp.ModelCatBHistSourcePacket\_category\_dependency\_boundary`

Bridge status: paperBridge

Not claimed: Host localization, quotient category, derived mapping space, external weak-equivalence calculus, checked bridge status, and downstream localization or replacement consumers are not closed.

Upgrade path: No further paper theory-axis step is claimed; bridgeChecked verification remains separate.

Chapter 159

A Concrete Naming Certificate for $\mathsf{InfCat}^{\uparrow}$

The ∞-category interface $\mathsf{InfCat}^{\uparrow}$ packages a simplicial-set carrier satisfying the inner-horn lifting condition (a quasicategory). It depends on $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ (chapter 157) and $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61).

159.1 Carrier definition

Definition 159.1 (InfCatUp BHist source packet)**.** *An* $\mathsf{InfCat}^{\uparrow}$ BHist *source packet is a finite tuple*

$$\mathcal{I} = (S, C, H, L, \rho, \lambda)$$

whose coordinates are all displayed BEDC rows. The row S is the consumed $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *row, carrying the simplicial carrier, face and degeneracy rows, and the simplicial classifier surface. The row C is the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row used for ordinary categorical source faces read by downstream consumers.*

The row H is the inner-horn ledger. It records finite horn-boundary rows, the missing inner face, the displayed filler endpoint, and the hsame *comparisons that identify the filler boundary with the named horn boundary inside the simplicial-set carrier S. The row L is the lifting-witness ledger: each lifting entry is a finite* $\mathsf{ProbeBundle}$ *row whose source and target faces are read through S, whose ordinary categorical endpoint is read through C when present, and whose continuation endpoint is recorded by* Cont.

The row ρ is the Pkg *provenance row for the consumed simplicial-set and category certificates, the horn ledger, and the lifting ledger. The row λ is the* $\mathsf{Cont}/\mathsf{hsame}$ *transport ledger comparing the simplicial faces, horn boundaries, fillers, lifting witnesses, and provenance entries under rowwise replacement. The packet has no coordinate for a host quasicategory, a localization, or a simplicial mapping object.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket provenance transport spine`

159.2 The certificate

Theorem 159.2 (InfCatUp obligation inventory fields)**.** *Every accepted* $\mathsf{InfCat}^{\uparrow}$ BHist *source packet contains exactly the displayed inventory needed by the next obligation layer: the consumed*

$\mathsf{NameCert}_{\mathsf{SimplicialSet}^\uparrow}$ *row, the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ *row, the finite inner-horn ledger, the lifting ledger, the* ProbeBundle *rows supporting the lifting entries, the* Cont *transport rows, the* hsame *comparison rows, and the* Pkg *provenance fields.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket obligation inventory fields`

Proof. Unfold Definition 159.1. An accepted packet is the tuple $\mathcal{I} = (S, C, H, L, \rho, \lambda)$, where S and C are precisely the consumed simplicial-set and category certificate rows.

The ledgers are also displayed tuple fields. The horn ledger H records the finite inner-horn rows, while the lifting ledger L records the lifting entries and their ProbeBundle support. The remaining tuple fields ρ and λ provide the Pkg provenance and the Cont/hsame transport rows. No additional coordinate is present, so this list is the full inventory. □

Theorem 159.3 (InfCatUp source and classifier obligations)**.** *The source and classifier obligations of an accepted* $\mathsf{InfCat}^\uparrow$ *packet are exactly the fields listed in Definition 159.2, together with componentwise* hsame/Cont *transport across the displayed packet coordinates. They include no host quasicategory coordinate and no localization coordinate.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket source classifier obligations`

Proof. The source side is obtained by unfolding Definition 159.1: every admissible source read is one of S, C, H, L, ρ, or λ, or a finite row projected from the horn or lifting ledgers.

For the classifier side, apply Definition 159.6. The classifier compares the same packet coordinates through the consumed dependency classifiers, componentwise hsame, Cont transport, and shared Pkg provenance.

Since neither the packet definition nor the classifier surface supplies a host quasicategory or localization slot, neither kind of coordinate can appear among the source or classifier obligations. □

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket source classifier obligation surface`

Theorem 159.4 (InfCatUp stability, exactness, and ledger obligations)**.** *For an accepted* $\mathsf{InfCat}^\uparrow$ *packet, the stability, exactness, ledger, and downstream-consumer obligations are discharged by the inner-horn ledger boundary, the quasicategory classifier surface, the NameCert obligation surface, and the consumer boundary stated in this chapter.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket stability exactness ledger obligations`

Proof. Begin with the ledger part. The inner-horn ledger boundary theorem Definition 159.5 restricts every horn read to the displayed simplicial-set row, finite horn ledger, provenance row, and transport ledger.

Next use the classifier surface of Definition 159.6. It supplies the componentwise hsame/Cont transport needed for stability under displayed replacement of the packet coordinates.

The exact obligation package is then read through Definition 159.7. Finally, Definition 159.8 identifies the downstream consumer boundary with the same consumed certificate rows, finite ledgers, provenance, and transport fields. These cited surfaces together exhaust the claimed obligation families. □

Theorem 159.5 (InfCatUp inner-horn ledger boundary)**.** *Every accepted* $\mathsf{InfCat}^\uparrow$ *source packet of Definition 159.1 exposes its inner-horn data only through the consumed* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^\uparrow}$ *row* S*, the finite inner-horn ledger* H*, the* Pkg *provenance row* ρ*, and the* Cont/hsame *transport ledger*

λ. *Thus a consumer of the inner-horn boundary cannot read a host quasicategory, a localization, or a simplicial mapping object.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket inner horn ledger boundary`

Proof. Project an accepted packet $\mathcal{I} = (S, C, H, L, \rho, \lambda)$. The horn boundary, missing inner face, filler endpoint, and boundary comparison rows are exactly the entries of H, and each of those entries is scoped by the simplicial-set carrier and classifier rows supplied by S.

The provenance and transport reads are also displayed. The row ρ records which packet face supplied the horn entry, while λ transports the horn boundary, filler endpoint, and comparison rows through Cont and hsame.

No other tuple coordinate is available for an inner-horn read. In particular, the category dependency C and lifting ledger L are not host quasicategory data, and the tuple has no localization or simplicial mapping-object slot. Repacking the projected rows gives exactly the stated boundary. □

Theorem 159.6 (InfCatUp quasicategory classifier surface)**.** *For two* $\mathsf{InfCat}^{\uparrow}$ *source packets, the quasicategory classifier surface compares only the displayed* BHist *rows* S, C, H, L, ρ, *and* λ *by the simplicial-set classifier, the category classifier, componentwise* hsame, Cont *transport, and shared* Pkg *provenance. It supplies no classifier for a host quasicategory, a localization, or a simplicial mapping object.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket quasicategory classifier surface`
↪ variant `BEDC.Derived.InfCatUp.InfCatQuasicategoryClassifierSurface transport`

Proof. Begin with the two consumed certificate rows. The simplicial-set dependency classifies simplicial carrier, face, degeneracy, and simplex rows, while the category dependency classifies the ordinary categorical source faces that are explicitly present in the packet.

Next compare the horn and lifting ledgers. The entries of H and L are finite BHist rows, so their comparisons are componentwise hsame rows carried by the ledger λ. Continuation endpoints move through the displayed Cont entries, and the package row ρ keeps the same provenance attached to the transported packet.

Repack these comparisons as the classifier surface. Since every compared coordinate is one of S, C, H, L, ρ, or λ, there is no remaining field from which a host quasicategory classifier, localization classifier, or mapping-object classifier could be read. □

Theorem 159.7 (InfCatUp NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{InfCat}^{\uparrow}}$ *obligation surface of an accepted packet* $\mathcal{I} = (S, C, H, L, \rho, \lambda)$ *consists exactly of the* BHist *source packet of Definition 159.1, the inner-horn ledger boundary of Definition 159.5, and the quasicategory classifier surface of Definition 159.6. Its rows are the consumed* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *row, the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row, the inner-horn ledger, the lifting witness ledger, the* Pkg *provenance row, and the* $\mathsf{Cont}/\mathsf{hsame}$ *transport ledger, with no host quasicategory, localization, or simplicial mapping object.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket namecert obligation surface`

Proof. Soundness is by projection from the source packet. The carrier row exposes only S, C, H, L, ρ, and λ, the horn theorem reads the inner-horn boundary from S, H, ρ, and λ, and the classifier theorem compares those same displayed packet coordinates.

Completeness is the reverse reading. Any row used by carrier admission, classifier transport, horn filling, lifting witness readback, provenance, or continuation transport is one of the six packet coordinates or a projected entry of H, L, ρ, or λ.

Therefore the naming-certificate obligation surface is exactly the displayed finite packet and its projected horn and classifier rows. The proof never opens a host quasicategory, localization, or simplicial mapping-object coordinate, because the packet has no such coordinate. □

Theorem 159.8 (InfCatUp consumer boundary)**.** *Every downstream consumer of the* $\mathsf{InfCat}^{\uparrow}$ *root reads through the displayed* BHist *packet* $(S, C, H, L, \rho, \lambda)$*, the two consumed* NameCert *rows* S *and* C*, the finite inner-horn and lifting ledgers* H *and* L*, the* Pkg *provenance row* ρ*, and the* Cont/hsame *transport ledger* λ*. Conversely those rows supply the complete consumer boundary for the root, and no consumer obtains a host quasicategory, localization, or simplicial mapping object.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket consumer boundary`

Proof. Project any accepted consumer read to the packet coordinates. A simplicial read factors through S, an ordinary categorical source read factors through C, an inner-horn read factors through H, and a lifting-witness read factors through L.

The remaining consumer information is provenance and transport. The package row ρ records which dependency or ledger supplies the read, and the ledger λ records the Cont and hsame movements needed to keep the read stable under displayed replacement.

For the converse, repack these projected rows into the source packet and use Definition 159.7. It identifies the same rows as the complete naming-certificate surface, so they already supply every consumer-facing obligation. Since no projected coordinate names a host quasicategory, localization, or simplicial mapping object, no downstream consumer can read one from this root. □

Theorem 159.9 (InfCatUp public certificate boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{InfCat}^{\uparrow}}$ *surface is exactly the consumed* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *row, the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row, the finite inner-horn ledger, the lifting ledger, the package provenance row, and the* Cont/hsame *transport ledger. No host quasicategory, localization, or simplicial mapping object is part of the public certificate boundary.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket public certificate boundary`

Proof. First project the finite horn part by Definition 159.5. The inner-horn boundary is read through the consumed simplicial-set row, the horn ledger, provenance, and displayed transport.

Next use Definition 159.6. The classifier surface compares only the displayed packet coordinates and their componentwise hsame and Cont rows, so it adds no host quasicategory, localization, or mapping-object coordinate.

Now apply Definition 159.7. It identifies the NameCert obligations with the consumed simplicial-set and category rows, inner-horn ledger, lifting ledger, package provenance, and transport ledger.

Finally read the downstream boundary through Definition 159.8. Its converse direction repacks exactly those rows as the complete consumer surface, so the public certificate boundary is precisely the displayed finite packet surface and nothing outside it. □

Theorem 159.10 (InfCatUp inner-horn consumer determinacy)**.** *For any accepted* $\mathsf{InfCat}^{\uparrow}$ *packet* $\mathcal{I} = (S, C, H, L, \rho, \lambda)$*, every downstream consumer read of an inner-horn or lifting obligation is determined entirely by the displayed* BHist *rows* $(S, C, H, L, \rho, \lambda)$*, the consumed* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *rows, the finite* ProbeBundle *horn and lifting ledger entries,* Pkg *provenance, and* Cont/hsame *transport. No host quasicategory, localization, or simplicial mapping-object coordinate can change the read.*

Proof. Project the consumer through Definition 159.8. Any inner-horn read is already a read of S, H, ρ, and λ, while any lifting-obligation read is already a read of S, C, L, ρ, and λ.

Restrict the inner-horn part by Definition 159.5. The horn boundary, missing face, filler endpoint, and comparison rows are finite ledger entries carried by H, with provenance supplied by ρ and displayed transport supplied by λ.

Transport equality of consumer reads through Definition 159.6. Since the classifier compares only the six packet coordinates and their consumed certificate rows, equal displayed packets give equal downstream reads of the horn and lifting surfaces.

Finally repack the exact surface by Definition 159.7. The repacked surface contains only the consumed certificate rows, the finite horn and lifting ledgers, provenance, and Cont/hsame transport. A host quasicategory, localization, or simplicial mapping-object coordinate is therefore not available to affect the read. □

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket inner horn consumer determinacy`

Theorem 159.11 (InfCatUp root-threshold obligation triad)**.** *Every accepted* $\mathsf{InfCat}^{\uparrow}$ *source packet supplies exactly three downstream root surfaces: the consumed* $\mathsf{NameCert}_{\mathsf{SimplicialSet}^{\uparrow}}$ *row, the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row, and the finite inner-horn plus lifting-ledger surface carried by* ProbeBundle, Cont, hsame, *and* Pkg *rows. Host quasicategory, localization, and simplicial mapping-object data are not available to downstream category consumers at this root threshold.*

Lean4 ✓ `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket root threshold obligation triad`

Proof. Project the accepted packet as $\mathcal{I} = (S, C, H, L, \rho, \lambda)$. The two consumed certificate rows are the S and C coordinates, and the finite obligation surface is read through the horn and lifting ledgers H and L with provenance ρ and transport λ.

Apply Definition 159.5 to identify the finite inner-horn side of the surface. Apply Definition 159.6 to see that the classifier compares only displayed packet coordinates and the componentwise hsame/Cont rows.

Now use Definition 159.7 and Definition 159.8. Together they repackage the two consumed NameCert rows and the horn-plus-lifting ledger as the full downstream consumer boundary. Since no projected coordinate names a host quasicategory, localization, or simplicial mapping object, those data remain unavailable at the root threshold. □

Closure status for $\mathsf{NameCert}_{\mathsf{InfCat}^{\uparrow}}$**.**

Constructive story: An InfCat packet is generated from consumed $\mathsf{SimplicialSet}^{\uparrow}$ and $\mathsf{Cat}^{\uparrow}$ certificate rows, finite inner-horn and lifting ledgers, one provenance package, and rowwise Cont/hsame transport. The checked consumer boundary exposes only those displayed packet rows to downstream category consumers.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InfCat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public certificate boundary of Definition 159.9 binds the consumed $\mathsf{SimplicialSet}^{\uparrow}$ and $\mathsf{Cat}^{\uparrow}$ rows, finite inner-horn and lifting ledgers, ProbeBundle, Cont, hsame, and Pkg rows displayed by the scoped source-packet surface of Definition 159.5, Definition 159.6, Definition 159.7, and Definition 159.8.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InfCat}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InfCatUp.InfCatBHistSourcePacket\_public\_certificate\_boundary`

Bridge status: paperBridge

Not claimed: Host quasicategories, localization, and simplicial mapping objects are not closed.

Upgrade path: No higher paper theory-axis grade remains for the displayed InfCat packet; bridge verification is tracked separately on the verification axis.

Chapter 160

A Concrete Naming Certificate for $\mathsf{DerivedCat}^{\uparrow}$

The derived-category interface $\mathsf{DerivedCat}^{\uparrow}$ packages the localisation of a chain-complex category at quasi-isomorphisms. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) and $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ (chapter 99).

160.1 Carrier definition

Definition 160.1 (DerivedCatUp NameCert obligation surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *presenting a chain-complex category* $\mathcal{C}$ *and a* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *presenting cohomology readback on those complexes. The* $\mathsf{DerivedCat}^{\uparrow}$ *obligation surface is the following visible localization interface.*

The traditional target is the localization of the chain-complex category at quasi-isomorphisms. The BEDC source is the paired source $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$*. The carrier is a* BHist *package whose rows expose chain-complex object histories, homotopy-class morphism histories, and quasi-isomorphism witnesses read by the cohomology certificate. The classifier* $\sim_{\mathsf{DerivedCat}^{\uparrow}}$ *is generated by zigzags whose elementary steps are quasi-isomorphism witnesses and category-level morphism classifiers, with no primitive quotient constructor.*

The stability obligations are reflexivity, symmetry, and transitivity of displayed zigzags; transport of the carrier along the generated classifier; congruence of represented roof composition; and descent of the visible ledger along classifier-equivalent representatives. The ledger records roofs, quasi-isomorphism witnesses, cone and cohomology rows, and composition witnesses. Exactness is the visible localization readback: two displayed representatives are classified exactly when the ledger supplies a finite zigzag through quasi-isomorphism rows and category morphism rows. The standard bridge field is absent.

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationCarrier surface rows`

160.2 The certificate

Theorem 160.2 (Derived localization carrier obligation)**.** *The carrier in Definition 160.1 exposes exactly the chain-complex objects and homotopy-class morphism histories supplied by* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$*, together with the quasi-isomorphism witnesses supplied by* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$*. It carries no object, morphism, or weak-equivalence row except through these displayed certificate fields.*

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatLocalizationRoof carrier obligation`

Proof. Project the carrier row of Definition 160.1. Its object and morphism coordinates are the object and morphism rows of the chosen $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$, so every carried localization endpoint has a displayed category object or homotopy-class morphism history.

The weak-equivalence coordinate is read from the cohomology certificate: a morphism row is marked invertible for localization only when the $\mathsf{NameCert}_{\mathsf{Cohomology}^\uparrow}$ ledger supplies the corresponding quasi-isomorphism witness. Hence every exposed quasi-isomorphism is certificate-visible.

Conversely, any category object row, homotopy-class morphism row, and cohomology quasi-isomorphism witness may be injected into the carrier by the displayed package constructor. Constructor inversion on that package shows there are no additional carrier cases. This proves the exact exposure claim. □

Theorem 160.3 (Derived quasi-isomorphism classifier obligation)**.** *The classifier $\sim_{\mathsf{DerivedCat}^\uparrow}$ of Definition 160.1 is the finite zigzag closure generated by cohomology quasi-isomorphism witnesses and the existing category morphism classifiers. It is not an independent quotient primitive.*

Proof. The forward direction is induction on the displayed classifier derivation. The reflexive case is the empty zigzag. A category-morphism classifier step gives a one-step zigzag labelled by the $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ row. A quasi-isomorphism step gives a one-step zigzag labelled by the $\mathsf{NameCert}_{\mathsf{Cohomology}^\uparrow}$ witness. Symmetry reverses the finite zigzag, and transitivity concatenates the two finite zigzags.

For the reverse direction, induct on the finite zigzag. The empty zigzag gives reflexivity. In the cons step, the head is either a category-morphism classifier row or a cohomology quasi-isomorphism witness, hence gives an elementary classifier step. Compose it with the induction hypothesis for the tail by transitivity of the generated classifier.

All constructors used in both directions are the displayed generator constructors and the reflexive, symmetric, and transitive closure rules. Since no step appeals to an unlabelled endpoint identification, the classifier is precisely the generated zigzag closure. □

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationClassifier endpoints`

Theorem 160.4 (Derived composition descent obligation)**.** *Composition of represented roofs descends along $\sim_{\mathsf{DerivedCat}^\uparrow}$. If two left roofs and two right roofs are pairwise classified by the generated zigzag classifier, then the composite roofs obtained from the $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ composition rows and the $\mathsf{NameCert}_{\mathsf{Cohomology}^\uparrow}$ quasi-isomorphism stability rows are again classified by $\sim_{\mathsf{DerivedCat}^\uparrow}$.*

Proof. Represent each morphism by a roof ledger row. The composition row first forms the category-level composite using the associativity and unit witnesses already exposed by $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$. These witnesses identify the usual alternative parenthesizations and inserted identity rows inside the category morphism classifier.

Now induct over the displayed zigzag comparing the left roof representatives. For an elementary category-classifier step, composition congruence in $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ transports the step through the composite. For an elementary quasi-isomorphism step, the cohomology stability row says that the pulled-back or pasted weak-equivalence witness remains a quasi-isomorphism witness, so it is again a generator for $\sim_{\mathsf{DerivedCat}^\uparrow}$.

Repeat the same induction over the right roof comparison. The two transported zigzags concatenate, and the category associativity/unit rows align the middle object choices used by the two roof-composition ledgers. The resulting finite zigzag is a classifier witness between the two composite roofs, which is exactly descent of composition. □

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationClassifier composition descent`

Theorem 160.5 (DerivedCatUp NameCert exactness obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ *fields determined by Definition 160.1 are exactly the visible localization carrier, the generated zigzag classifier, the stability rows, the roof and quasi-isomorphism ledger, the visible exactness readback, and the absent standard-bridge row.*

Proof. Unpack the public certificate fields. The carrier field is the package of Definition 160.2, and the classifier field is the generated zigzag classifier of Definition 160.3. Thus the public object and morphism surfaces are exactly the displayed localization rows.

The stability fields are supplied by the reflexive, symmetric, and transitive zigzag constructors, by carrier transport along generated classifier steps, and by composition descent from Definition 160.4. The ledger field is the visible list of roofs, quasi-isomorphism witnesses, cone and cohomology rows, and composition witnesses used by those constructions.

It remains to identify exactness and the bridge row. Exactness is constructor inversion on the generated classifier: a public equality witness is precisely a finite ledger zigzag through the displayed category and cohomology rows, and any such zigzag repacks as a public classifier witness. The standard bridge field is the designated absent row, so the certificate exposes no unrecorded external bridge data. □

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationClassifier name certificate`

Theorem 160.6 (DerivedCatUp roof identity unit laws)**.** *Let r be a displayed* $\mathsf{DerivedCat}^{\uparrow}$ *roof ledger row from X to Y in the visible localization interface of Definition 160.1. Let* id_X *and* id_Y *be displayed identity roof ledger rows at the two endpoints, with category components supplied by the empty identity hom-carriers of Definition 61.2 and with their weak-equivalence components recorded by the displayed cohomology quasi-isomorphism rows. If ℓ is a ledger result for composing* id_Y *with r and ρ is a ledger result for composing r with* id_X*, then both identity composites are classified with the original roof:*

$$\ell \sim_{\mathsf{DerivedCat}^{\uparrow}} r \quad \textit{and} \quad \rho \sim_{\mathsf{DerivedCat}^{\uparrow}} r.$$

Proof. Unpack the roof ledger for r. It exposes the source and target object histories, the roof middle object, the category morphism leg, the cohomology quasi-isomorphism leg, and the composition witnesses used by Definition 160.1. The two displayed identity roofs expose only empty identity hom-carrier rows at X and Y, together with the cohomology rows marking those identity legs as quasi-isomorphisms.

For the composite $\mathsf{id}_Y \circ r$, the category morphism part is a displayed right-identity composite of the category leg of r. By Definition 61.105, that composite carries the same hom-carrier and its morphism history is hsame to the original category leg. The weak-equivalence part is obtained by pasting the displayed cohomology identity row with the quasi-isomorphism row already present in r; the cohomology ledger transports this pasted witness across the same identity comparison, so no new weak-equivalence generator is introduced.

Thus the ledger for ℓ differs from the ledger for r by a finite list of displayed generator steps: one category morphism-classifier step collapsing the empty identity composite, followed by the reflexive or transported quasi-isomorphism rows on the weak-equivalence side. By Definition 160.3, such a finite list is exactly a witness for $\ell \sim_{\mathsf{DerivedCat}^{\uparrow}} r$.

The composite $r \circ \mathsf{id}_X$ is symmetric at the source endpoint. Apply Definition 61.103 to collapse the category leg containing the source identity, and use the displayed cohomology identity row to transport the quasi-isomorphism leg back to the original ledger row. These two displayed

steps again form a finite zigzag made only of category morphism classifiers and quasi-isomorphism witnesses. The generated classifier therefore gives $\rho \sim_{\mathsf{DerivedCat}^{\uparrow}} r$. □

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationClassifier roof identity units`

Theorem 160.7 (DerivedCatUp localization example boundary)**.** *The visible localization classifier of Definition 160.1 supplies two finite example roofs inside the displayed* $\mathsf{DerivedCat}^{\uparrow}$ *package: the identity-roof example at a chain-complex object and the one-step quasi-isomorphism roof example for a displayed quasi-isomorphism. Both examples use only the displayed chain-complex carrier, quasi-isomorphism classifier, roof identity and unit rows, and* Pkg *provenance. They export no host localization, quotient object, or derived functor calculus.*

Proof. The carrier rows come from Definition 160.2. At an object X, the identity roof uses the displayed category identity row as its category leg and the cohomology identity quasi-isomorphism witness as its weak-equivalence leg. The unit laws of Definition 160.6 classify this roof with the visible identity representative.

For a displayed quasi-isomorphism row, use Definition 160.3. The one-step roof has that quasi-isomorphism as its weak-equivalence leg and the displayed category morphism row as its other leg. Its classifier witness is a single generator in the finite zigzag closure.

The obligation surface of Definition 160.1 records both examples in the same Pkg provenance ledger. Since each construction uses only carrier rows, quasi-isomorphism generators, roof identity/unit rows, and package provenance, neither example creates a host localization, quotient object, or derived functor operation. □

Lean4 ✓ `BEDC.Derived.DerivedCatUp.DerivedCatLocalizationBoundary example boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$**.**

Constructive story: A derived-category object is built by pairing a $\mathsf{Cat}^{\uparrow}$ presentation of chain-complex objects and morphism histories with a $\mathsf{Cohomology}^{\uparrow}$ row that marks the quasi-isomorphism witnesses used for localization. The carrier accepts only the displayed package of chain-complex object rows, homotopy-class morphism rows, roof ledgers, cone and cohomology rows, and quasi-isomorphism witnesses; it excludes a primitive quotient category, unlabelled weak equivalences, and any standard bridge row. Two displayed representatives classify together by a finite zigzag generated from category morphism classifiers and cohomology quasi-isomorphism witnesses, with symmetry given by reversing the zigzag and transitivity by concatenating it. The source specification is the paired category/cohomology localization surface, the pattern records roof formation, identity roof units, composition descent, and visible exactness, and the ledger keeps roofs, weak-equivalence witnesses, cone/cohomology rows, and composition witnesses visible as finite data. The five naming-certificate obligations hold because empty zigzags and identity roofs give carrier witnesses and reflexivity, zigzag reversal and concatenation give symmetry and transitivity, and carrier transport plus composition descent keep every classified roof inside the accepted localization surface. A history h lies in the signature gap of a $\mathsf{DerivedCat}^{\uparrow}$ token p when the source domain accepts this visible localization packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, the finite zigzag through the category and cohomology rows is exactly the localization evidence encoded by p. The chapter registers this generated-zigzag localization under $\mathsf{DerivedCat}^{\uparrow}$; it adds no quotient constructor, no unrecorded weak-equivalence oracle, and no standard derived-category bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public certificate surface is the visible $\mathsf{Cat}^{\uparrow}$/$\mathsf{Cohomology}^{\uparrow}$ localization classifier together with the roof identity rows, roof composition descent, exactness readback, ledger rows, and absent bridge row.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DerivedCatUp.DerivedCatVisibleLocalizationClassifier\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not close standard bridges for derived categories or full localization maturity beyond the displayed certificate rows.

Upgrade path: No further paper theory-axis grade remains for this displayed derived-category localization package.

Chapter 161

A Concrete Naming Certificate for $\mathsf{FieldExt}^{\uparrow}$

The field-extension interface $\mathsf{FieldExt}^{\uparrow}$ packages an embedding of one field carrier into another, recording the larger field as a vector space over the smaller. It depends on $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45) and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47), and is the substrate for Galois, separable, and algebraic-closure certificates.

Every accepted $\mathsf{FieldExt}^{\uparrow}$ carrier read first exposes the base and extension field rows through chapter 45, the scalar-action and vector-space-over-base row through chapter 47, and the module compatibility row through chapter 46. Only after those rows are visible may an extension-degree, embedding, number-field, or Galois-extension consumer inspect the local $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ surface.

161.1 Carrier definition

Definition 161.1 (Concrete singleton empty-history field extension instance)**.** *Let the base field carrier and extension field carrier both be* FieldSingletonCarrier *over* BHist*, with field classifier* FieldSingletonClassifier*. Equivalently, this is the singleton empty-history carrier* A_{e} *with shared classifier*

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

Both field endpoints use the singleton $\mathsf{Field}^{\uparrow}$ *operations of Definition 45.2 for both the base and extension fields:* FieldSingletonAdd*,* FieldSingletonMul*,* FieldSingletonZero*,* FieldSingletonOne*,* FieldSingletonNeg*, and the proof-indexed inverse* FieldSingletonInv*. Define the embedding by* $\iota_{\mathsf{e}}(h) := h$ *on* BHist*.*

Read the extension as the singleton $\mathsf{VecSpace}^{\uparrow}$ *carrier* VecSpaceSingletonCarrier *over the singleton scalar field. Its vector classifier is* VecSpaceSingletonClassifier *and its scalar action is* VecSpaceSingletonSmul*. The ledger is inherited from the singleton* $\mathsf{Field}^{\uparrow}$ *certificate on the two field sides and from the singleton* $\mathsf{VecSpace}^{\uparrow}$ *certificate on the vector-space-over-base side.*

The certificate fit has ten fields:

1. *Traditional target: the terminal field extension.*
2. *BEDC source:* BHist*.*
3. *Carrier choice:* A_{e} *for the base field, extension field, and vector carrier.*
4. *Classifier choice:* ρ_{e} *on all displayed endpoints.*
5. *Stability obligations: the singleton field laws together with the singleton vector-space laws.*

6. *Ledger: the identity embedding and vector-space provenance over the same empty-history endpoint.*
7. *Core theorem list: Theorems 161.4, 161.5, 161.28, 161.29, 161.31, 161.32, 161.33, 161.36, 161.37, and 161.34.*
8. *Exactness predicate: singleton endpoint collapse.*
9. *Standard bridge: the terminal field extension.*
10. *Status:* ProofTarget.

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton empty history instance surface`

Definition 161.2 (Singleton FieldExt obligation scope)**.** *The terminal* $\mathsf{FieldExt}^{\uparrow}$ *instance over* A_{e} *is the following ten-field paper template. The traditional target is the one-point field extension. The BEDC source is* BHist*. The base field carrier, extension field carrier, and extension vector carrier are all* A_{e}*. The classifier is the shared singleton relation* ρ_{e} *from Definition 161.1. The embedding is the identity map* $\iota_{\mathsf{e}}(h) := h$*. The extension-field vector-space-over-base action is the singleton vector-space action over the singleton* $\mathsf{Field}^{\uparrow}$ *source.*

The stability obligations are the singleton field rows of Theorem 45.37 for the base and extension fields, together with the singleton vector-space rows of Theorem 47.3. The ledger consists of identity-embedding provenance and singleton scalar-action provenance over the same empty-history endpoint. The downstream consumers are the Galois, number-field, and algebraic-closure certificate families, each of which reads the same carrier, classifier, exactness, scalar-action, and ledger fields.

Definition 161.3 (RatUp reflexive field-extension carrier)**.** *Let the base field carrier and the ambient field carrier both be the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *carrier of Definition 45.488:*

$$C_{\mathsf{base}} = C_{\mathsf{amb}} = \mathsf{RatHistoryCarrier}, \qquad R_{\mathsf{base}} = R_{\mathsf{amb}} = \mathsf{RatHistoryClassifier}.$$

The classifier for the extension package is the RatUp classifier on the same BHist *endpoints. The embedding is the identity map* $\iota_{\mathsf{Rat}}(h) := h$ *on* BHist*.*

The ambient field is also used as its own vector carrier over the base field. Scalar multiplication is the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *multiplication row of Definition 45.489, and vector addition, zero, and negation are the additive rows retained by the same FieldUp certificate through the commutative-ring and ring reducts. Thus the displayed object is the reflexive field extension*

$$\mathsf{Rat}^{\uparrow} \hookrightarrow \mathsf{Rat}^{\uparrow}$$

with no change of BEDC carrier, classifier, ledger source, or operation table.

Lean4 (definition) `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveCarrier`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveCarrier rat history closure`

Theorem 161.4 (Singleton field-extension embedding obligations)**.** *For the concrete data of Definition 161.1, let* h, k : BHist *be carried by* FieldSingletonCarrier*. Then* $\iota_{\mathsf{e}}(h)$ *is carried by the extension field,*

$$\mathsf{FieldSingletonClassifier}(h, k) \quad \textit{if and only if} \quad \mathsf{FieldSingletonClassifier}(\iota_{\mathsf{e}}(h), \iota_{\mathsf{e}}(k)),$$

and the embedding preserves the concrete singleton zero, one, addition, negation, multiplication, and proof-indexed inverse up to ρ_{e}*. Thus the base endpoint and its embedded extension endpoint have the same singleton classification for each displayed field operation.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonEmbedding laws`

Proof. Unfold ι_{e}. The carrier statement is immediate from the assumption $A_{\mathsf{e}}(h)$, because the base and extension field carriers are the same singleton carrier A_{e}.

The classifier comparison has the same two directions. After unfolding the identity embedding, both sides are the identical classifier assertion $\rho_{\mathsf{e}}(h, k)$.

For zero, one, addition, negation, and multiplication, apply Theorem 45.37 to obtain the field-law rows over $\mathsf{FieldSingletonCarrier}$ and $\mathsf{FieldSingletonClassifier}$. Then use Theorem 45.48 to read each concrete operation endpoint as the singleton empty-history endpoint.

For the inverse rows, fix a proof-indexed witness in the singleton nonzero domain. The same operation-table collapse theorem gives the inverse endpoint row for $\mathsf{FieldSingletonInv}(h, p)$, and the law package uses that same proof-indexed domain. Hence base inverse and embedded extension inverse are classified by ρ_{e} at the same singleton endpoint. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton embedding obligations`

Theorem 161.5 (Singleton FieldExt embedding exactness)**.** *For the concrete data of Definition 161.1, let $h, k : \mathsf{BHist}$ be carried by A_{e}. Then*

$$\rho_{\mathsf{e}}(h, k) \quad \textit{if and only if} \quad \rho_{\mathsf{e}}(\iota_{\mathsf{e}}(h), \iota_{\mathsf{e}}(k)).$$

Moreover, for every displayed singleton field operation in the base package, its embedded extension endpoint is classified with the corresponding source endpoint by ρ_{e}.

Proof. Unfold ι_{e}. The displayed equivalence is the classifier comparison recorded in Theorem 161.4; after the identity embedding is unfolded, both sides are the same singleton classifier assertion.

For the operation endpoints, apply Theorem 161.4 to the carried inputs. It gives preservation of zero, one, addition, negation, multiplication, and the proof-indexed inverse up to ρ_{e}.

The concrete endpoint identification comes from the singleton operation-table collapse in Theorem 45.48. Thus each embedded operation row lands in the same ρ_{e} class as its source row. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonEmbedding exactness`

161.2 The certificate

Theorem 161.6 (RatUp reflexive embedding transport)**.** *For the concrete data of Definition 161.3, the identity embedding ι_{Rat} transports carrier membership and classifier comparisons exactly. If $h, k : \mathsf{BHist}$, then $\mathsf{RatHistoryCarrier}(h)$ gives $\mathsf{RatHistoryCarrier}(\iota_{\mathsf{Rat}}(h))$, and $\mathsf{RatHistoryClassifier}(h, k)$ gives and is given by $\mathsf{RatHistoryClassifier}(\iota_{\mathsf{Rat}}(h), \iota_{\mathsf{Rat}}(k))$. The same identity transport preserves the zero, one, addition, negation, multiplication, apartness, and proof-indexed inverse rows of the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveEmbedding transport fields`

Proof. Unfold ι_{Rat}. Carrier preservation is immediate because the base and ambient carriers are both $\mathsf{RatHistoryCarrier}$.

The classifier transport has identical source and target assertions after unfolding the identity map. Reflexivity, symmetry, transitivity, and endpoint carrier extraction are the RatUp classifier fields recorded in Definition 37.90.

For the operation rows, read the base and ambient field packages from Definition 45.489. They use the same carrier, classifier, denominator tuple, contextual action, and affine source. Since the embedding is identity, every displayed FieldUp operation endpoint is compared with itself under RatHistoryClassifier, including the proof-indexed inverse endpoint whenever the apartness witness is supplied. □

Theorem 161.7 (RatUp reflexive vector-space package)**.** *For the concrete data of Definition 161.3, the ambient* $\mathsf{Rat}^{\uparrow}$ *carrier is a vector space over the base* $\mathsf{Rat}^{\uparrow}$ *carrier. Its vector addition, zero, and negation are the FieldUp additive rows, and its scalar action is FieldUp multiplication. The vector carrier classifier is* RatHistoryClassifier*, the scalar classifier is the same RatUp classifier, and scalar-action congruence is the multiplication congruence row of the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveVectorSpace scalar action congruence`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive vector space package`

Proof. Start from the concrete field certificate Definition 45.489. Its carrier and classifier are RatHistoryCarrier and RatHistoryClassifier, and its algebraic rows include additive abelian-group laws, multiplication closure, distributivity, unit laws, and classifier congruence for multiplication.

Use the field multiplication row as scalar action. Associativity of scalar action is exactly associativity of FieldUp multiplication. Left and right distributivity of scalar action are the two distributivity rows of the same FieldUp certificate, read with the first factor as scalar and the second factor as vector.

The vector addition, zero, and negation rows are the additive rows already present in the field certificate. The projection principles Definition 47.24 and Definition 47.25 identify these as the module-shaped and additive carrier rows expected by $\mathsf{VecSpace}^{\uparrow}$ once the scalar field is fixed. Therefore the self-action package supplies the paper-side vector-space obligations over the unchanged RatUp carrier. □

Theorem 161.8 (RatUp reflexive FieldExtUp certificate obligations)**.** *The data of Definition 161.3 satisfy the paper-side* $\mathsf{FieldExt}^{\uparrow}$ NameCert *obligation package. The package contains the base and ambient* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificates, the RatUp classifier, identity embedding stability, scalar-action congruence, the self-vector-space rows, exact endpoint classification, and ledger provenance from the same concrete FieldUp operation table.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive certificate obligations`

Proof. The two field endpoints are supplied by a single certificate: Definition 45.489. Reading it twice fixes the base and ambient carriers to RatHistoryCarrier and fixes both classifiers to RatHistoryClassifier.

Embedding stability is Definition 161.6. It records carrier transport, classifier transport, and operation transport for the identity map on BHist.

The vector-space component is Definition 161.7. It uses FieldUp multiplication as scalar action and the FieldUp additive rows as vector addition, zero, and negation. The scalar-action congruence row is therefore the multiplication classifier-congruence row already present in the concrete FieldUp certificate.

The ledger clauses do not introduce a second provenance source. Every scalar, vector, embedding, and operation endpoint is read from the same $\mathsf{RatUpFieldSource}_F$ tuple of Definition 45.488.

The exactness clauses are the RatUp classifier and FieldUp operation-table exactness rows assembled in Definition 45.489. These rows are precisely the carrier, classifier, embedding, vector-space, exactness, and ledger obligations named by the FieldExtUp certificate boundary. □

Definition 161.9 (RatUp reflexive ledger interface)**.** *For the concrete data of Definition 161.3, the RatUp reflexive* $\mathsf{FieldExt}^{\uparrow}$ *ledger interface has every displayed carrier and endpoint read from the same RatUp table. The base carrier, ambient carrier, vector carrier, and scalar carrier are all* $\mathsf{RatHistoryCarrier}$*, and their classifier is* $\mathsf{RatHistoryClassifier}$*.*

The embedding endpoint is the identity map $\iota_{\mathsf{Rat}}(h) := h$*. The base-field and ambient-field operation endpoints are the corresponding zero, one, addition, negation, multiplication, apartness, and proof-indexed inverse endpoints of the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table in Theorem 45.489. The scalar-action endpoints are the same multiplication endpoints, read through the self-vector-space package of Theorem 161.7.*

Theorem 161.10 (RatUp reflexive exact endpoint classification)**.** *For the concrete data of Definition 161.3, let* $h, k : \mathsf{BHist}$ *be carried by* $\mathsf{RatHistoryCarrier}$*. Every base-field operation endpoint, ambient-field operation endpoint, identity-embedding endpoint, and scalar-action endpoint displayed by the reflexive* $\mathsf{FieldExt}^{\uparrow}$ *package is classified by* $\mathsf{RatHistoryClassifier}$ *against the corresponding* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *endpoint.*

Proof. First apply Theorem 45.489. It supplies the concrete RatUp carrier, classifier, and operation table for the base and ambient field rows, so the two field sides compare endpoints against the same $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ table.

The identity-embedding endpoints are handled by Theorem 161.6. Since ι_{Rat} unfolds to the identity on BHist, the embedded endpoint and its RatUp source endpoint have the same classifier class.

The scalar-action endpoints are handled by Theorem 161.7. Its scalar action is precisely the multiplication row of the RatUp field table, and its congruence row is the multiplication classifier-congruence row of the same certificate.

Finally use Theorem 37.90 to read each carried endpoint through $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$. This gives the displayed exact endpoint classification for field operations, the identity embedding, and scalar action. □

Theorem 161.11 (RatUp reflexive ledger coverage)**.** *For the concrete data of Definition 161.3, every* $\mathsf{FieldExt}^{\uparrow}$ *source row, pattern row, classifier row, stability row, exactness row, and ledger row is covered by exactly one of the following provenance classes: the identity embedding row, the RatUp* $\mathsf{Field}^{\uparrow}$ *operation-table row, or the self-vector-space scalar-action row.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive ledger coverage`

Proof. Start with the ledger interface of Definition 161.9. It assigns the four carrier displays to $\mathsf{RatHistoryCarrier}$, assigns every classifier display to $\mathsf{RatHistoryClassifier}$, and separates the public rows into the identity embedding, RatUp operation table, and self-vector-space scalar action.

Next apply Theorem 161.10. It places each displayed field-operation endpoint, embedding endpoint, and scalar-action endpoint in the exact RatUp classifier class specified by that interface.

It remains to check that these rows exhaust the certificate boundary. Apply Theorem 161.8. Its obligation list is precisely the carrier, classifier, embedding-stability, vector-space, exactness, and ledger surface of the RatUp reflexive $\mathsf{FieldExt}^{\uparrow}$ package, so the three provenance classes cover every listed row exactly once. □

Theorem 161.12 (RatUp reflexive FieldExt source, pattern, and classifier obligations)**.** *For the data of Definition 161.3, the* $\mathsf{FieldExt}^{\uparrow}$ *source field, extension field, vector carrier, embedding, and classifier slots are the concrete RatUp rows. The base and extension field sources are both the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *source on* RatHistoryCarrier*, the vector carrier is the same* RatHistoryCarrier *read as the ambient field over the base, the embedding is* $\iota_{\mathsf{Rat}}(h) := h$*, and every source, pattern, and classifier endpoint is classified by* RatHistoryClassifier*. These rows are exactly the FieldExtUp source, pattern, and classifier obligations for the reflexive rational field extension.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive source pattern classifier obligations`

Proof. Unfold Definition 161.3. Its displayed carrier equations identify the base field carrier, ambient field carrier, and extension vector carrier with RatHistoryCarrier, and its displayed classifier equations identify the two endpoint classifiers with RatHistoryClassifier.

The source field and extension field certificates are not separate objects. Apply Theorem 45.489 to read the same $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ operation table as both endpoints of the extension. The RatUp semantic name certificate, Theorem 37.90, supplies the classifier row for the same carried histories.

The remaining FieldExtUp source and pattern slots are the identity embedding and the self-vector-carrier reading. Both are named explicitly in Definition 161.3. Therefore no carrier or classifier conversion is needed: the source field, extension field, vector carrier, embedding, and RatHistoryClassifier rows are the required source-pattern-classifier surface for the reflexive certificate. □

Theorem 161.13 (RatUp reflexive FieldExt stability and exactness obligations)**.** *For the same RatUp reflexive carrier, the* $\mathsf{FieldExt}^{\uparrow}$ *stability and exactness rows are supplied by identity-embedding transport, scalar-action congruence, the vector-space-over-base laws, and exact endpoint classification for the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table. In particular, embedding transport preserves carrier membership, classifier equivalence, and all displayed field operations; scalar action is FieldUp multiplication; vector addition, zero, and negation are the FieldUp additive rows; and every endpoint read by these rows is classified by* RatHistoryClassifier*.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive stability exactness obligations`

Proof. First apply Theorem 161.6. Since ι_{Rat} is the identity on BHist, carrier transport, classifier transport, and operation transport all compare an endpoint with itself under RatHistoryClassifier.

Next apply Theorem 161.7. It reads FieldUp multiplication as scalar action and the additive FieldUp rows as vector addition, zero, and negation over the unchanged RatUp carrier. The scalar-action congruence obligation is therefore the multiplication classifier-congruence row of the same FieldUp operation table.

The exact endpoint classifications come from Theorem 45.489. Its operation table already contains the carrier, classifier, additive, multiplicative, and inverse exactness rows used by the identity embedding and the self-vector-space package.

Combining these rows gives the stability and exactness surface required by $\mathsf{FieldExt}^{\uparrow}$: every transported endpoint, scalar-action endpoint, and vector-space endpoint is a RatUp FieldUp endpoint classified by RatHistoryClassifier. □

Theorem 161.14 (RatUp reflexive FieldExt ledger and standard bridge obligations)**.** *For the data of Definition 161.3, the* $\mathsf{FieldExt}^{\uparrow}$ *ledger has one provenance source, namely the* $\mathsf{RatUpFieldSource}_F$ *operation table, and the standard bridge is the reflexive rational field extension*

$$\mathsf{Rat}^{\uparrow} \hookrightarrow \mathsf{Rat}^{\uparrow}.$$

The ledger records no independent extension-field, vector-space, or embedding source: all of them are readings of the RatUp FieldUp carrier, classifier, and operation table already used by the assembled reflexive certificate.

Proof. Read the ledger paragraph of Definition 161.3. It states that the base field, ambient field, vector carrier, classifier, ledger source, and operation table are unchanged across the displayed reflexive extension.

The provenance source is therefore the tuple $\mathsf{RatUpFieldSource}_F$ of Definition 45.488. The exactness rows cited in Theorem 161.8 are the RatUp classifier and FieldUp operation-table exactness rows assembled by Theorem 45.489.

Since the embedding is $\iota_{\mathsf{Rat}}(h) := h$ and the ambient field is the base field itself, the standard bridge has no additional data beyond the ordinary reflexive rational field extension. Thus the ledger and standard bridge obligations are both read from the same concrete RatUp FieldUp table. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive ledger standard bridge obligations`

Theorem 161.15 (RatUp reflexive FieldExtUp tower composition)**.** *Compose two copies of the reflexive FieldExtUp certificate of Definition 161.3. The resulting two-step tower has base, middle, and top carrier all equal to* RatHistoryCarrier*, has base, middle, and top classifier all equal to* RatHistoryClassifier*, has composite embedding equal to the identity embedding on* BHist*, and uses the same FieldUp multiplication scalar action in each vector-space-over-base row. The tower therefore carries no endpoint outside the concrete RatUp FieldUp operation table.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive tower composition`

Proof. For each field endpoint in the tower, read the same concrete FieldUp certificate from Definition 45.489. This fixes the base, middle, and top carriers to RatHistoryCarrier and fixes the three classifiers to RatHistoryClassifier.

The embedding rows are identity rows. Apply Definition 161.6 to each copy. Since the underlying map is the identity on BHist, composing the two displayed embeddings again gives the identity embedding and preserves the same carrier, classifier, and operation rows.

The vector-space rows are supplied by Definition 161.7. Its scalar action is FieldUp multiplication, and its scalar-action congruence is the multiplication congruence row of the same RatUp FieldUp certificate.

Finally apply Definition 161.8. It records that the carrier, classifier, embedding, scalar-action, exactness, and ledger rows all come from the single concrete RatUp FieldUp source tuple, so the two-step tower has the stated reflexive package. □

Theorem 161.16 (RatUp reflexive FieldExtUp tower scalar action)**.** *In the two-step reflexive tower of Definition 161.15, let r, m :* BHist *be carried RatUp scalar and vector endpoints. The tower scalar-action endpoint obtained by acting with r on m is classified by* RatHistoryClassifier *with the corresponding RatUp FieldUp multiplication endpoint for r and m.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveTower scalar action`

Proof. The tower scalar action is the self-vector-space scalar action selected in Definition 161.7. By that theorem, the action row is FieldUp multiplication on the same RatUp carrier.

The multiplication endpoint and the scalar-action endpoint therefore name the same operation row of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate. The required classifier comparison is exactly the multiplication congruence row supplied by Definition 45.489, read through RatHistoryClassifier.

Since Definition 161.15 keeps the identity embedding at both tower levels, no transport changes the endpoints. The classification row remains the RatUp multiplication row for the original carried scalar and vector endpoints. □

Theorem 161.17 (RatUp reflexive FieldExtUp tower ledger coverage)**.** *Every carrier row, classifier row, embedding row, scalar-action row, exactness row, and ledger row in the two-step reflexive tower of Definition 161.15 is covered by three concrete sources: the single RatUp FieldUp source tuple* $\mathsf{RatUpFieldSource}_F$*, the identity embedding row on* BHist*, and the self-vector-space row whose scalar action is FieldUp multiplication.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveTower ledger coverage`

Proof. The carrier and classifier rows are covered by the concrete RatUp FieldUp source tuple via Definition 45.489. The same source also supplies the operation-table rows used at the base, middle, and top field levels.

The embedding rows are covered by Definition 161.6. The two embeddings and their composite are all the identity on BHist, so the tower has one embedding policy and one classifier-transport row.

The scalar-action rows are covered by Definition 161.7 and Definition 161.16. They identify the tower action with FieldUp multiplication and classify its endpoints by the RatUp multiplication congruence row.

The exactness and ledger rows are then those assembled in Definition 161.8. Together with Definition 161.15, this accounts for every row of the two-step reflexive tower without naming any source beyond the RatUp FieldUp tuple, identity embedding, and self-vector-space package. □

Theorem 161.18 (RatUp reflexive FieldExt operation-table coverage)**.** *For the concrete package of Definition 161.3, every base-field row, ambient-field row, vector addition row, vector zero row, vector negation row, and scalar-action row is read from Definition 45.489 over* $\mathsf{RatHistoryCarrier}$ *and* $\mathsf{RatHistoryClassifier}$*. The base-field and ambient-field operation tables are the same concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table, while the vector-space additive rows and scalar-action row are the corresponding additive and multiplication rows of that table.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveEmbedding denominator package`

Proof. Unfold Definition 161.3. The embedding ι_{Rat} is the identity on BHist, so the base and ambient endpoints are not transported to a distinct carrier or classifier.

Read Definition 45.489 once for the base field and once for the ambient field. Both readings use $\mathsf{RatHistoryCarrier}$, $\mathsf{RatHistoryClassifier}$, and the concrete RatUp operation table.

For the vector-space rows, apply Definition 47.24 to retain the scalar action and module-shaped compatibility from the FieldUp multiplication and distributivity rows. Then apply Definition 47.25 to retain the vector addition, zero, negation, carrier, and classifier. Thus the FieldExt operation table introduces no row outside the displayed RatUp FieldUp table. □

Theorem 161.19 (RatUp reflexive scalar-action readback)**.** *For the concrete package of Definition 161.3, the* $\mathsf{FieldExt}^{\uparrow}$ *scalar action is exactly the multiplication row of the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate. For every scalar endpoint* $r : \mathsf{BHist}$ *and vector endpoint* $m : \mathsf{BHist}$ *carried by* $\mathsf{RatHistoryCarrier}$*, the scalar-action endpoint* $r \odot_{\mathsf{Rat}} m$ *is the FieldUp product endpoint* $r \cdot_{\mathsf{Rat}} m$*, and classifier stability is the multiplication congruence row already included in Definition 45.489.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive scalar action readback`

Proof. Unfold the scalar action in Definition 161.3. It is defined to be the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ multiplication row, with no change of carrier, classifier, or endpoint domain.

The assembled FieldUp certificate supplies multiplication closure and multiplication classifier congruence over RatHistoryClassifier. Because the scalar classifier, vector classifier, and ambient-field classifier are all the same RatUp classifier, the congruence row repacks directly as scalar-action stability for the reflexive $\mathsf{FieldExt}^{\uparrow}$ package. □

Theorem 161.20 (RatUp reflexive FieldExt ledger provenance coverage)**.** *For the concrete package of Definition 161.3, the source, pattern, classifier, stability, exactness, and ledger rows are exhausted by the identity embedding row, the single concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table, and the self-vector-space projection over* RatHistoryCarrier*. No additional carrier, classifier, scalar-action table, or ledger source is introduced by the reflexive* $\mathsf{FieldExt}^{\uparrow}$ *reading.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveEmbedding scalar action cont readback`

Proof. The identity embedding row is Definition 161.6. It fixes endpoint transport, classifier transport, and operation transport to the unchanged RatUp carrier and classifier.

The field and operation-table rows are supplied by Definition 161.18. The scalar-action readback is Definition 161.19, so the vector-space scalar action uses exactly the multiplication row already present in the concrete FieldUp table.

Finally apply Definition 161.7 and Definition 161.8. These package the self-vector-space projection, exact endpoint classification, and ledger provenance as the displayed $\mathsf{FieldExt}^{\uparrow}$ NameCert obligations. Hence every ledger row named by the reflexive FieldExt certificate is traced to the identity embedding, the RatUp FieldUp certificate, or the self-vector-space projection. □

Definition 161.21 (RatUp reflexive exactness and ledger interface)**.** *For the concrete data of Definition 161.3, the* $\mathsf{FieldExt}^{\uparrow}$ *exactness and ledger surface is fixed by the single RatUp history source. The base-field carrier, ambient-field carrier, vector carrier, and scalar carrier are all* RatHistoryCarrier*, and all endpoint comparisons use* RatHistoryClassifier*.*

The embedding row is the identity map $\iota_{\mathsf{Rat}}(h) := h$*. The base and ambient field operation rows are the rows of the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *table, and the scalar action is the same table read through multiplication:*

$$r \cdot_{\mathsf{Rat}} m := \mathsf{RatUpFieldMul}(r, m).$$

Thus the ledger has no separate extension source: it consists of the RatUp FieldUp operation table, the identity embedding row, and the self vector-space row over the same classifier.

Lean4 (definition) `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveExactnessLedgerInterface`
Lean4 (definition) `BEDC.Derived.FieldExtUp.FieldExtRatReflexive exactness ledger interface`

Theorem 161.22 (RatUp reflexive exact endpoint classification)**.** *For the concrete data of Definition 161.3, every displayed base-field endpoint, ambient-field endpoint, identity-embedding endpoint, and scalar-action endpoint is classified by* RatHistoryClassifier *against the RatUp endpoint named by the same concrete operation row. In particular, the reflexive extension has one exact classifier surface for field operations, embedding transport, and self scalar action.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveEmbedding classifier endpoint package`

Proof. The base and ambient field endpoints are read from the same concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate. Apply Theorem 45.489 to obtain the operation-table rows over RatHistoryCarrier and RatHistoryClassifier.

The identity-embedding endpoints are handled by Theorem 161.6. Since ι_{Rat} is the identity on BHist, the embedded endpoint and the source endpoint are the same carried history, compared by the RatUp classifier.

The scalar-action endpoints are handled by Theorem 161.7. Its scalar action is FieldUp multiplication, so the scalar-action endpoint is classified by the same multiplication congruence and endpoint row already present in the RatUp FieldUp certificate. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive exact endpoint classification`

Theorem 161.23 (RatUp reflexive ledger provenance)**.** *For the concrete data of Definition 161.3, every* $\mathsf{FieldExt}^{\uparrow}$ *ledger row is inherited from one of three displayed sources: the single RatUp* $\mathsf{Field}^{\uparrow}$ *operation table, the identity embedding row* ι_{Rat}*, or the self-vector-space row whose scalar action is RatUp multiplication. No ledger row uses a carrier, classifier, operation endpoint, or scalar-action endpoint outside* $\mathsf{RatHistoryCarrier}$ *and* $\mathsf{RatHistoryClassifier}$*.*

Proof. The operation-table provenance is the concrete RatUp FieldUp certificate of Theorem 45.489. The base field and ambient field read that certificate with identical carrier, classifier, and operation endpoints.

The embedding provenance is Theorem 161.6. It supplies the identity carrier row, the identity classifier row, and the preservation of the displayed field operations without adding a second source.

The self-vector-space provenance is Theorem 161.7. It uses the FieldUp additive rows for vector addition, zero, and negation, and uses the FieldUp multiplication row as scalar action. These are exactly the remaining ledger rows required by the reflexive $\mathsf{FieldExt}^{\uparrow}$ package. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive ledger provenance`

Theorem 161.24 (RatUp reflexive public NameCert interface)**.** *The RatUp reflexive construction exposes the public paper-side* $\mathsf{FieldExt}^{\uparrow}\ \mathsf{NameCert}$ *interface with carrier* $\mathsf{RatHistoryCarrier}$*, classifier* $\mathsf{RatHistoryClassifier}$*, identity embedding* ι_{Rat}*, base and ambient* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificates, self-vector-space rows, exact endpoint classification, and ledger provenance over the RatUp operation table.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive public name certificate`

Proof. The carrier and classifier are fixed by Definition 161.3. The base and ambient field certificates are the two readings of Theorem 45.489.

The stability component is supplied by Theorem 161.6, and the vector-space component is supplied by Theorem 161.7. Together they give the identity embedding and self scalar-action surface.

The exactness and ledger interface is Definition 161.21. Its two coverage theorems, Theorems 161.22 and 161.23, assemble the remaining public rows into the displayed $\mathsf{FieldExt}^{\uparrow}\ \mathsf{NameCert}$ interface. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive semantic name certificate`

Theorem 161.25 (RatUp reflexive FieldExt source-pattern lock)**.** *For the reflexive package of Definition 161.3, every displayed source and pattern field is fixed by the concrete RatUp data already named in Definition 45.488 and certified by Theorem 45.489. The base field carrier, ambient field carrier, and vector carrier are* $\mathsf{RatHistoryCarrier}$ *over* BHist*; the field classifier, extension classifier, and vector classifier are* $\mathsf{RatHistoryClassifier}$*; the embedding is the identity map* $\iota_{\mathsf{Rat}}(h) := h$*; the scalar action is the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *multiplication row; and the ledger source is the same concrete FieldUp source tuple on both endpoints.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexiveEmbedding ledger source lock`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive source pattern lock`

Proof. Unfold Definition 161.3. Its base field, ambient field, and vector carrier clauses are not separate choices: each names RatHistoryCarrier over BHist, and each classifier clause names RatHistoryClassifier.

The embedding row is locked by Theorem 161.6. Since ι_{Rat} is identity on BHist, the source endpoint and the displayed embedded endpoint are the same RatUp history.

The vector and action rows are locked by Theorem 161.7. It reads vector addition, zero, and negation from the additive FieldUp rows and reads scalar action from FieldUp multiplication. Hence the package has no carrier, classifier, embedding, action, or ledger source outside the RatUp FieldUp certificate boundary. □

Theorem 161.26 (RatUp reflexive FieldExt operation-table source coverage)**.** *In the reflexive FieldExt package of Definition 161.3, the zero, one, addition, negation, multiplication, apartness, and proof-indexed inverse rows of the base and ambient fields are read from the same concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table. The vector addition, vector zero, vector negation, and scalar-action rows are read from that same table: the vector additive rows are the FieldUp additive rows, and scalar action is the FieldUp multiplication row over* RatHistoryCarrier *and* RatHistoryClassifier*.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive embedded scalar action source readback`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive operation table source coverage`

Proof. Apply Theorem 45.489. It provides the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ operation table over RatHistoryCarrier and RatHistoryClassifier, including zero, one, addition, negation, multiplication, apartness, and the proof-indexed inverse.

The reflexive package reads the same certificate twice, once as the base field and once as the ambient field. Therefore every displayed field-operation row on either endpoint has the same operation-table source.

For the vector part, use Theorem 161.7. Its self-vector-space package takes vector addition, zero, and negation from the FieldUp additive rows and takes scalar action from FieldUp multiplication. Thus the complete operation surface of the reflexive FieldExt package is covered by the single RatUp FieldUp operation table. □

Theorem 161.27 (RatUp reflexive FieldExt certificate row exhaustion)**.** *For the concrete data of Definition 161.3, every displayed* $\mathsf{FieldExt}^{\uparrow}$ *source, pattern, classifier, stability, exactness, scalar-action, and ledger row for the RatUp reflexive package is covered by exactly one of the following concrete row families:*

1. *the two reads of the* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *endpoint certificate from Theorem 45.489;*
2. *the identity embedding row of Theorem 161.6;*
3. *the self-vector-space and scalar-action row of Theorem 161.7.*

No additional carrier, classifier, operation table, exactness source, or ledger source is used by the reflexive certificate.

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtRatReflexive certificate row exhaustion`

Proof. The source and pattern clauses are fixed by Theorem 161.25. It assigns the base field, ambient field, vector carrier, classifiers, identity embedding, scalar action, and ledger source to the concrete RatUp FieldUp data.

The operation clauses are covered by Theorem 161.26. It accounts for the two field endpoint reads and for the vector additive and scalar-action rows read from the same operation table.

Finally apply Theorem 161.8. Its obligation list consists only of the base and ambient FieldUp certificates, identity embedding stability and exactness, self-vector-space compatibility, scalar-action congruence, exact endpoint classification, and ledger provenance. These are precisely the three row families listed above, so the displayed RatUp reflexive FieldExt certificate has no uncovered row. □

Theorem 161.28 (Singleton field extension as a vector space over its base)**.** *For the concrete data of Definition 161.1, the extension carrier read as* VecSpaceSingletonCarrier *with scalar action* VecSpaceSingletonSmul *satisfies the* $\mathsf{VecSpace}^{\uparrow}$ *row over the base* $\mathsf{Field}^{\uparrow}$ *source. Moreover, for every base scalar* $r : \mathsf{BHist}$ *and extension vector endpoint* $m : \mathsf{BHist}$ *carried by* VecSpaceSingletonCarrier*, scalar multiplication by the embedded scalar agrees, up to the singleton classifier, with singleton field multiplication in the extension field.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton vector space over base`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton vector action field mul context readback`

Proof. The vector-space rows are exactly Theorem 47.3, applied to the same scalar carrier, vector carrier, and classifier displayed in Definition 161.1. Its semantic certificate packaging is recorded by Theorem 47.14.

The scalar-action endpoint collapses to e by Theorem 47.4. The extension-field multiplication endpoint also collapses to e by Theorem 45.48.

Since both endpoints are carried by A_{e} and identified with the same empty-history endpoint, the classifier ρ_{e} relates the scalar-action endpoint to the extension-field multiplication endpoint. This gives the compatibility of embedded scalar multiplication with the singleton extension field product. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonVectorSpace smul mul compatible`

Theorem 161.29 (Singleton FieldExt embedded scalar-action collapse)**.** *For the concrete data of Definition 161.1, let* $r, m : \mathsf{BHist}$ *be carried by* A_{e}*. The vector-space scalar action of the embedded singleton scalar* $\iota_{\mathsf{e}}(r)$ *on* m *and the singleton extension-field multiplication endpoint for* r *and* m *lie in the same* ρ_{e} *class.*

Proof. Apply Theorem 161.28. It gives the semantic compatibility statement between embedded scalar multiplication and extension-field multiplication for carried scalar and vector endpoints.

The vector-space action endpoint collapses to e by Theorem 47.4. The extension-field multiplication endpoint collapses to e by Theorem 45.48.

Since both endpoints are carried by A_{e} and both collapse to the same empty-history endpoint, the shared classifier ρ_{e} relates them. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton scalar action mul endpoint classified`

Definition 161.30 (Singleton field-extension ledger policy)**.** *For the concrete data of Definition 161.1, the endpoint ledger records exactly the base carrier* A_{e}*, the extension carrier* A_{e}*, the vector carrier* A_{e}*, the identity embedding* $\iota_{\mathsf{e}}(h) := h$*, the singleton field operations on the base and extension fields, the singleton vector action over the base field, and empty-history endpoint provenance for each displayed row.*

The ledger classifier attached to every carrier, embedding, field-operation, and vector-action endpoint is the concrete classifier

$$\rho_{\mathsf{e}}(h,k) := A_{\mathsf{e}}(h) \wedge A_{\mathsf{e}}(k) \wedge \mathsf{hsame}(h,k).$$

No additional endpoint object is recorded by the singleton field-extension certificate.

Lean4 (definition) `BEDC.Derived.FieldExtUp.FieldExtSingletonLedgerPolicy`

Theorem 161.31 (Singleton field-extension carrier coincidence)**.** *For the concrete data of Definition 161.1 and the ledger policy of Definition 161.30, the base field carrier, extension field carrier, and vector carrier are all A_{e}. The embedding ι_{e} preserves and reflects carrier membership:*

$$A_{\mathsf{e}}(h) \quad \textit{if and only if} \quad A_{\mathsf{e}}(\iota_{\mathsf{e}}(h)).$$

Every displayed classifier in the field-extension certificate is the concrete ρ_{e} classifier.

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonCarrier coincidence`

Proof. Unfold the concrete carrier definition and the ledger policy. The three carrier slots are all assigned the same singleton predicate A_{e}, so the carrier coincidence is definitional.

The embedding is the identity on histories. Hence preserving and reflecting carrier membership both reduce to the same assertion $A_{\mathsf{e}}(h)$.

The classifier rows are the rows already used by Theorem 161.4 and Theorem 161.28. Each row is displayed with the same concrete predicate ρ_{e}, so no transport across distinct classifiers is required. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonLedgerPolicy carrier coincidence`

Theorem 161.32 (Singleton identity field-extension tower composition)**.** *Compose two copies of the singleton field-extension certificate of Definition 161.1, both governed by the ledger policy of Definition 161.30. The composite tower again has base carrier A_{e}, middle carrier A_{e}, top carrier A_{e}, identity composite embedding $\iota_{\mathsf{e}} \circ \iota_{\mathsf{e}} = \iota_{\mathsf{e}}$, singleton field-operation rows, and the same singleton vector-space-over-base scalar action. All embedding and scalar-action rows in the composite tower are classified by ρ_{e}.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonEmbedding identity tower package`

Proof. The embedding part follows from Theorem 161.4. Since both embeddings are the identity on BHist, their composite is again the identity embedding ι_{e}.

The field-operation rows in the base, middle, and top fields collapse to the same singleton endpoint by Theorem 45.48. Thus tower composition introduces no field endpoint beyond the empty-history endpoint.

The scalar-action rows are supplied by Theorem 161.28. The endpoint collapse for the singleton vector action is exactly Theorem 47.4, so the composite tower keeps the same ρ_{e} classifier on its action rows. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton identity tower continuation classified`

Theorem 161.33 (Singleton field-extension exact endpoint ledger)**.** *For the concrete data of Definition 161.1 and the ledger policy of Definition 161.30, every embedding endpoint, every extension-field operation endpoint, and every vector-space-over-base action endpoint recorded in the ledger collapses to the empty history e and is classified by ρ_{e}.*

Proof. For embedding endpoints, apply Theorem 161.4. The embedding is the identity, so the only carried endpoint is the same singleton history already classified by ρ_{e}.

For extension-field operation endpoints, apply Theorem 45.48. The zero, one, addition, negation, multiplication, and proof-indexed inverse rows all collapse to e in the singleton field table.

For vector-space-over-base action endpoints, use Theorem 161.28 together with Theorem 47.4. The scalar action collapses to the same empty-history endpoint as the compatible extension-field multiplication row.

The ledger policy records no further endpoint kind. Therefore every recorded endpoint row in the singleton field-extension certificate is exact at e and classified by ρ_{e}. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonLedgerPolicy exact endpoint ledger`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton exact endpoint ledger`

Theorem 161.34 (Singleton field-extension certificate obligation package)**.** *The concrete data of Definition 161.1 assemble the paper-side* $\mathsf{FieldExt}^{\uparrow}$ *obligation package: base field source, extension field source, embedding stability and exactness, vector-space-over-base row, exact endpoint classification, and inherited ledger provenance.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton certificate obligation package`
Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton semantic name certificate`

Proof. The source field certificate and extension field certificate are both supplied by Theorem 45.81, because the base and extension field carriers are the same singleton field package.

Embedding stability is supplied by Theorem 161.4. It records carrier preservation, classifier preservation, and preservation of the singleton field operations up to $\mathsf{FieldSingletonClassifier}$. Exactness is the same identity-on-BHist fact: every carried base endpoint is already the extension endpoint named by ι_{e}.

Vector-space-over-base compatibility is supplied by Theorem 161.28, whose vector-space component is the law package of Theorem 47.3. Its paper-side semantic packaging is recorded by Theorem 47.14.

Exactness and ledger provenance are supplied by Theorems 161.36 and 161.37. These clauses assemble the listed $\mathsf{FieldExt}^{\uparrow}$ $\mathsf{NameCert}$ obligations for the concrete singleton field-extension carrier. □

Definition 161.35 (Singleton field-extension exactness and ledger interface)**.** *For the concrete data of Definition 161.1, the* $\mathsf{FieldExt}^{\uparrow}$ *exactness and ledger interface is read entirely through the singleton carrier* A_{e} *and classifier* ρ_{e}*. Thus the displayed base-field carrier, extension-field carrier, identity embedding endpoint, vector scalar-action endpoint, and extension-field multiplication endpoint all have the same source* A_{e}*, and every endpoint comparison used by the certificate is a* ρ_{e} *row over that source.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtUp StdBridge`

Theorem 161.36 (Singleton field-extension exact endpoint classification)**.** *For the concrete data of Definition 161.1, every displayed base-field operation endpoint, every displayed extension-field operation endpoint, the vector scalar-action endpoint, and the identity embedding endpoint are carried by* A_{e} *and classified with* e *by* ρ_{e}*. In particular, the operation table of the base field, the operation table of the extension field, and the scalar-action table of the extension vector space have one exact endpoint class.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton exact endpoint classification`

Proof. For the base and extension field operation endpoints, apply Theorem 45.48. Both field structures in Definition 161.1 use the same singleton $\mathsf{Field}^{\uparrow}$ operation table.

For the vector scalar-action endpoint, apply Theorem 47.4. Its scalar-action clause reads the endpoint as e over the same A_{e} and ρ_{e}.

The embedding endpoint is immediate from $\iota_{\mathsf{e}}(h) := h$. If h is carried by A_{e}, then the singleton carrier identifies h with e, and unfolding ρ_{e} gives the displayed classifier row. □

Theorem 161.37 (Singleton field-extension ledger provenance)**.** *For the concrete data of Definition 161.1, the* $\mathsf{FieldExt}^{\uparrow}$ *ledger records only the identity embedding row together with the singleton field and vector-space provenance rows. The field rows are the two copies of the singleton empty-history field certificate, the vector row is the singleton empty-history vector-space semantic certificate, and the embedding row is the identity stability package for* ι_{e}*.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton ledger provenance`

Proof. The two field ledger rows are supplied by Theorem 45.81, once for the base field and once for the extension field. The two uses have identical carrier, classifier, and singleton operation endpoints.

The vector-space provenance row is supplied by Theorem 47.14. It uses the same scalar source, vector source, classifier, and empty-history scalar-action endpoint as the displayed $\mathsf{FieldExt}^{\uparrow}$ data.

The embedding row is supplied by Theorem 161.4. Since ι_{e} is the identity on histories, no additional ledger source is introduced beyond the singleton field and vector-space rows. □

↪ variant `BEDC.Derived.FieldExtUp.FieldExtSingleton ledger provenance`

Theorem 161.38 (Singleton field-extension certificate projections)**.** *For the concrete data of Definition 161.1, the* $\mathsf{FieldExt}^{\uparrow}$ *certificate has the following three projections over the shared singleton carrier* A_{e} *and classifier* ρ_{e}*:*

1. *its source-field projection is the singleton* $\mathsf{Field}^{\uparrow}$ *certificate on* A_{e}*;*
2. *its extension-field projection is the singleton* $\mathsf{Field}^{\uparrow}$ *certificate on the same* A_{e}*;*
3. *its vector-space projection is the singleton* $\mathsf{VecSpace}^{\uparrow}$*-over-base certificate whose scalar carrier, vector carrier, and endpoint classifier are again* A_{e} *and* ρ_{e}*.*

The embedding row connecting the first two projections is the identity map $\iota_{\mathsf{e}}(h) := h$*.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton certificate projections`

Proof. The source and extension projections use the same displayed singleton field data. Apply Theorem 45.81 once for the source field and once for the extension field.

The vector-space projection is the semantic singleton vector-space package. Apply Theorem 47.14 with the scalar carrier, vector carrier, and classifier specified in Definition 161.1.

The projection map from the source field into the extension field is ι_{e}. By its definition, it is the identity on histories, so it preserves the displayed carrier and classifier rows without introducing a second endpoint choice. □

Theorem 161.39 (Singleton field-extension operation readback exactness)**.** *For the concrete data of Definition 161.1, every zero, one, addition, negation, multiplication, proof-indexed inverse, and scalar-action endpoint in the singleton field-extension package reads back to the singleton empty-history endpoint. Each such endpoint is classified by* ρ_{e} *against the corresponding embedded source-field endpoint or extension-field endpoint displayed by the* $\mathsf{FieldExt}^{\uparrow}$ *package.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonOperation readback exactness`

Proof. For the field operations, apply Theorem 45.48. It collapses zero, one, addition, negation, multiplication, and the proof-indexed inverse endpoint to the singleton empty-history endpoint.

The embedding comparison is supplied by Theorem 161.4. Since the embedding is the identity on histories, the embedded source endpoint and the corresponding extension-field endpoint are compared by the same classifier ρ_{e}.

For scalar action, apply Theorem 47.4 to read the vector-space action endpoint as the singleton empty-history endpoint. Then apply Theorem 161.28 to identify that scalar-action endpoint with the matching extension-field multiplication endpoint up to ρ_{e}. □

Theorem 161.40 (Singleton field-extension nonzero domain absurdity)**.** *For the concrete data of Definition 161.1, no base-field or extension-field proof-indexed inverse domain is inhabited. Equivalently, if a : BHist is carried by the singleton field carrier and $\mathsf{FieldSingletonNonZero}(a)$ is the displayed nonzero premise used by the base or extension field inverse row, then contradiction follows from the singleton carrier collapse of a to e.*

Proof. Unfold the concrete singleton nonzero predicate. Its witness simultaneously places a at the empty history endpoint and at the endpoint $\mathsf{e0}(\mathsf{e})$.

The singleton carrier clause supplies the collapse of the same history to e. Composing the two endpoint identifications gives $\mathsf{e} \sim \mathsf{e0}(\mathsf{e})$ in the history classifier.

This is impossible by the constructor no-confusion principle for BHist: the empty history constructor and the one-step empty-marker constructor are distinct. Hence the assumed nonzero proof parameter has no inhabitant, both for the source field and for the extension field, because those two projected fields use the same singleton carrier and the same singleton nonzero predicate. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonNonzeroDomain absurd`

Theorem 161.41 (Singleton field-extension inverse-row vacuous exactness)**.** *For the concrete data of Definition 161.1, every proof-indexed inverse obligation in the source field and in the extension field is exact. The exactness is vacuous: before any inverse endpoint can be chosen, its nonzero proof parameter belongs to the empty domain identified in Theorem 161.40.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingletonInverse row vacuous exactness`

Proof. Fix any displayed inverse obligation in either projected singleton field. Its input consists of a history a together with a proof that $\mathsf{FieldSingletonNonZero}(a)$ holds.

Apply Theorem 161.40 to that proof parameter. The resulting contradiction closes the obligation immediately, so no separate endpoint case for inverse evaluation remains.

The same argument covers the base and extension projections because Definition 161.1 gives them the same singleton field data. Thus every proof-indexed inverse source, stability, and exactness row is exact over an empty domain. □

Theorem 161.42 (Singleton field-extension inverse ledger coverage)**.** *For the concrete data of Definition 161.1, the inverse rows in the $\mathsf{FieldExt}^{\uparrow}$ source, stability, exactness, and ledger fields are covered by the same empty-domain class. They introduce no endpoint beyond the singleton field operation table already used for the zero, one, addition, negation, and multiplication rows.*

Proof. The source and extension inverse rows first reduce to their proof-indexed nonzero premises. By Theorem 161.41, those premises are uninhabited, so the corresponding inverse obligations are exact without selecting a new endpoint.

For the remaining operation rows, apply Theorem 161.39. It assigns all displayed field endpoints to the singleton empty-history operation table and compares them by ρ_{e}.

Finally combine this empty-domain inverse class with Theorem 161.43. The ledger coverage therefore contains the identity embedding row, the singleton field operation table row, the singleton vector-space scalar-action row, and no additional inverse endpoint class. □

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton inverse ledger coverage`

Theorem 161.43 (Singleton field-extension exactness and ledger coverage)**.** *For the concrete data of Definition 161.1, every* $\mathsf{FieldExt}^{\uparrow}$ *source row, pattern row, classifier row, stability row, exactness row, and ledger row is covered by exactly one of the following provenance classes: the identity embedding row, the singleton field operation table row, or the singleton vector-space scalar-action row.*

Lean4 ✓ `BEDC.Derived.FieldExtUp.FieldExtSingleton exactness ledger coverage`

Proof. First take the projected certificate rows from Theorem 161.38. These account for the source-field, extension-field, and vector-space certificate surfaces over the common A_{e} and ρ_{e}.

Next use Theorem 161.39. It assigns the field operation endpoints to the singleton field operation table row and the scalar-action endpoints to the singleton vector-space scalar-action row, with all endpoint comparisons made by ρ_{e}.

It remains to account for the cross-carrier rows. The identity embedding row from Definition 161.1 and Theorem 161.4 covers carrier preservation, classifier preservation, and embedded source-operation stability.

Finally apply Theorem 161.34. Its assembled $\mathsf{FieldExt}^{\uparrow}$ $\mathsf{NameCert}$ obligation list is precisely the union of these three provenance classes, so the source, pattern, classifier, stability, exactness, and ledger rows are all covered. □

161.3 Sibling dependency lattice

The $\mathsf{FieldExt}^{\uparrow}$ carrier consumes the base and ambient field surfaces from chapter 45. Its reflexive and singleton instances read both endpoints through the displayed $\mathsf{Field}^{\uparrow}$ operation tables, classifiers, and ledger rows rather than through a separate host-field equality.

The vector-space-over-base row is supplied by chapter 47. This is the row that lets the ambient field be read as a module over the base field while keeping scalar action, vector addition, zero, negation, classifier stability, and provenance visible as $\mathsf{VecSpace}^{\uparrow}$ data.

Downstream Galois, number-field, separability, algebraic-closure, and tower consumers therefore read $\mathsf{FieldExt}^{\uparrow}$ through a horizontal dependency lattice: two $\mathsf{Field}^{\uparrow}$ endpoints, one $\mathsf{VecSpace}^{\uparrow}$-over-base row, one embedding row, and the local $\mathsf{FieldExt}^{\uparrow}$ naming ledger. No downstream consumer may replace this lattice by an ambient tower object whose base field, extension field, or vector-space row is not displayed.

Theorem 161.44 (FieldExt downstream dependency lattice exactness)**.** *For either concrete* $\mathsf{FieldExt}^{\uparrow}$ *packet displayed in this chapter, namely the singleton packet of Definition 161.1 or the RatUp reflexive packet of Definition 161.3, the* $\mathsf{FieldExt}^{\uparrow}$ *input part of any accepted downstream Galois, number-field, separability, algebraic-closure, or tower read is accepted exactly through the following five-coordinate lattice:*

1. *the displayed base* $\mathsf{Field}^{\uparrow}$ *endpoint row;*
2. *the displayed ambient* $\mathsf{Field}^{\uparrow}$ *endpoint row;*
3. *the displayed* $\mathsf{VecSpace}^{\uparrow}$*-over-base row for the ambient carrier;*
4. *the displayed embedding row from the base endpoint into the ambient endpoint;*
5. *the local* $\mathsf{FieldExt}^{\uparrow}$ *naming ledger for the same packet.*

Equivalently, the consumer-visible $\mathsf{FieldExt}^{\uparrow}$ *readback is the finite packet obtained by repacking these five rows. A read that replaces any one of them by an ambient tower object whose base field, extension field, or vector-space-over-base row is not displayed is not an accepted* $\mathsf{FieldExt}^{\uparrow}$ *input read.*

Proof. Start with the singleton packet. Its carrier definition fixes the base field, extension field, vector carrier, identity embedding, scalar action, and ledger in Definition 161.1. The obligation scope in Definition 161.2 names the same carrier, classifier, exactness, scalar-action, and ledger fields as the rows read by downstream certificate families.

The five projections are explicit. The two field endpoints are the two singleton $\mathsf{Field}^{\uparrow}$ readings supplied by Definition 161.38. The $\mathsf{VecSpace}^{\uparrow}$-over-base coordinate is Definition 161.28. The embedding coordinate is Definition 161.4, and its endpoint exactness is Definition 161.5. The ledger coordinate is Definition 161.37. Finally, Definition 161.43 exhausts every source, pattern, classifier, stability, exactness, and ledger row of the singleton $\mathsf{FieldExt}^{\uparrow}$ package by the identity embedding row, the singleton field operation-table row, and the singleton vector-space scalar-action row. Thus there is no sixth singleton coordinate through which a downstream read can enter the packet.

For the RatUp reflexive packet, the carrier definition Definition 161.3 identifies the base field, ambient field, and vector carrier with the same RatUp history carrier and identifies the embedding with the identity map. The two endpoint rows are the two readings of the concrete $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ certificate recorded in Definition 161.18. The $\mathsf{VecSpace}^{\uparrow}$-over-base row is Definition 161.7; its scalar action is the RatUp FieldUp multiplication row. The embedding row is Definition 161.6. The naming ledger is fixed by Definition 161.23 and exposed through the public interface in Definition 161.24.

The RatUp reflexive exhaustion theorem Definition 161.27 says that every displayed source, pattern, classifier, stability, exactness, scalar-action, and ledger row is covered by exactly the two endpoint FieldUp reads, the identity embedding row, and the self-vector-space row. The tower case does not add an escape coordinate: by Definition 161.15 and Definition 161.17, even a two-step reflexive tower keeps the same RatUp FieldUp table, identity embedding, and self-vector-space scalar action.

Combining the singleton and RatUp cases gives the stated inversion for every concrete $\mathsf{FieldExt}^{\uparrow}$ packet displayed in this chapter. If a downstream read is accepted as a $\mathsf{FieldExt}^{\uparrow}$ input, the cited projection and exhaustion rows recover the base FieldUp endpoint, ambient FieldUp endpoint, VecSpaceUp-over-base row, embedding row, and local naming ledger. Conversely, when those five rows are present, the corresponding obligation package is assembled by Definition 161.34 in the singleton case and by Definition 161.8 in the RatUp reflexive case. Therefore the accepted $\mathsf{FieldExt}^{\uparrow}$ input surface is exactly the displayed five-coordinate lattice, and an ambient tower object with any undisplayed endpoint or vector-space row has no coordinate in the accepted readback packet. □

Theorem 161.45 (FieldExt field and vector-space sibling dependency)**.** *Every accepted* $\mathsf{FieldExt}^{\uparrow}$ *carrier read must pass through the displayed base* $\mathsf{Field}^{\uparrow}$ *row, the displayed extension* $\mathsf{Field}^{\uparrow}$ *row,*

the embedding row between them, and the $\mathsf{VecSpace}^{\uparrow}$ *scalar-action row before any extension consumer reads the local* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *surface. The field endpoints are the substantive algebraic dependencies of chapter 45, and the scalar-action row is the substantive vector-space dependency of chapter 47. A consumer that reads the extension as an ambient tower while omitting one of these rows is not reading the accepted* $\mathsf{FieldExt}^{\uparrow}$ *packet.*

Proof. Project the displayed carrier rows. The base and extension endpoints are both $\mathsf{Field}^{\uparrow}$ rows, and the embedding row is typed between those endpoints, so its exactness cannot be read before both field tables are present.

The scalar-action coordinate is then read as a $\mathsf{VecSpace}^{\uparrow}$ row over the base field. It supplies the module-facing action consumed by extension consumers, while the embedding row supplies the endpoint comparison consumed by field consumers.

The local $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ row is reached only after those four sibling rows are displayed. Thus the read route is the base field, extension field, embedding, vector-space action, and local naming ledger, with no coordinate for an undisplayed host-field tower. □

Closure status for $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$.

Constructive story: A field extension at this boundary is built by taking the already certified one-point field surface and the one-point vector-space surface, then using the identity embedding so every accepted endpoint is the same empty-history carrier viewed simultaneously as base field, extension field, and scalar module. The classifier therefore asks for both the field classifier and the vector-space classifier on the same displayed history, which collapses to hsame at the empty endpoint rather than to equality of host fields or quotient presentations. The source covers exactly the singleton field carrier, while the pattern and ledger keep the vector-space-over-base row, the identity embedding row, scalar-action readbacks, exact endpoint classification, and RatUp-reflexive tower rows visible through the same Cont and Pkg provenance. Because e inhabits the carrier, hsame supplies reflexivity, symmetry, and transitivity, and both component carriers are closed under the displayed classifier, the five naming-certificate obligations are discharged by projecting and repacking the same singleton rows. A history h lies in the signature GAP of a $\mathsf{FieldExt}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer endpoint and evidence ledger, and token introduction identifies that answer as p; here the three tests reduce to the statement that h is the identity-extension endpoint whose scalar action and field multiplication both read back to e. The paper bridge sends this package to the ordinary terminal field extension and reads it back by the identity embedding, so the chapter registers the construction as $\mathsf{FieldExt}^{\uparrow}$ without adding nontrivial towers, Galois data, separability data, algebraic closure, or a separate extension carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature singleton field-extension package is exactly the semantic certificate, vector-space-over-base row, identity embedding obligations, exact endpoint classification, ledger rows, and RatUp-reflexive tower rows displayed in this chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FieldExtUp.FieldExtSingleton\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: Nontrivial extension towers, Galois, separability, algebraic-closure claims, and bridgeChecked verification are not claimed here.

Upgrade path: No further upgrade: chapter is at the top of the formal axis.

Chapter 162

A Concrete Naming Certificate for $\mathsf{AlgClosure}^{\uparrow}$

The algebraic-closure interface $\mathsf{AlgClosure}^{\uparrow}$ packages a field extension in which every nonconstant polynomial over the base field has a root. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

162.1 Carrier definition

An $\mathsf{AlgClosure}^{\uparrow}$ carrier packet is a finite BHist row family over the consumed $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ sources. It contains a field-extension row F, a nonconstant polynomial request row p, a displayed root endpoint r, a classifier transport row H, a finite root ledger L, and a package provenance row $\Pi : \mathsf{Pkg}$.

Theorem 162.1 (AlgClosureUp carrier source obligation)**.** *Every accepted* $\mathsf{AlgClosure}^{\uparrow}$ *carrier packet is sourced by a* $\mathsf{FieldExt}^{\uparrow}$ *extension row, a* $\mathsf{Poly}^{\uparrow}$ *nonconstant-polynomial request row over that extension, the displayed root endpoint row, the* hsame *classifier transport row, the shared* Pkg *provenance row, and the finite root-ledger rows. Conversely, these rows repack as the carrier surface of the same* $\mathsf{AlgClosure}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket source obligation`

Proof. Unpack the carrier packet into its displayed BHist coordinates. The $\mathsf{FieldExt}^{\uparrow}$ row supplies the base and extension histories, and the $\mathsf{Poly}^{\uparrow}$ row supplies the finite coefficient and nonconstant-request histories over that same extension row.

The root endpoint is not read from a host algebraic closure. It is a carried history endpoint whose acceptance is recorded by the finite ledger L, whose transport is the displayed hsame row H, and whose provenance is the single package row Π.

For the converse, take precisely those six row families and place them back into the packet slots. The carrier predicate asks for no further data, so the same rows reconstruct the accepted carrier surface. □

Theorem 162.2 (AlgClosureUp polynomial-root classifier obligation)**.** *Two accepted* $\mathsf{AlgClosure}^{\uparrow}$ *root-witness packets classify together exactly when their* $\mathsf{FieldExt}^{\uparrow}$ *source rows classify together, their* $\mathsf{Poly}^{\uparrow}$ *nonconstant request rows classify over the transported field-extension source, their displayed root endpoints are related by the packet* hsame *row, and their finite root ledgers and* Pkg *provenance rows are the corresponding transported rows.*

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket polynomial root classifier obligation`

Proof. Read the two packets through Definition 162.1. Each packet exposes the same six kinds of rows: field-extension source, polynomial request, root endpoint, classifier transport, root ledger, and provenance.

Compare the field-extension projections first. The $\mathsf{FieldExt}^{\uparrow}$ classifier binds the base and extension histories, so the polynomial request in the second packet is read over the transported extension source rather than over a fresh ambient field.

Now compare the polynomial and root projections. The $\mathsf{Poly}^{\uparrow}$ classifier transports the nonconstant request row, the displayed hsame row transports the root endpoint, and the ledger/provenance clauses say that L and Π are carried along the same comparison. These are all classifier fields, hence the packets classify together exactly under this componentwise finite comparison. □

Theorem 162.3 (AlgClosureUp ledger readback obligation)**.** *The visible root-certificate ledger of an accepted* $\mathsf{AlgClosure}^{\uparrow}$ *packet is exactly the finite family consisting of the* $\mathsf{FieldExt}^{\uparrow}$ *source row, the* $\mathsf{Poly}^{\uparrow}$ *nonconstant-polynomial request row, the displayed root endpoint row, the root-satisfaction row, the* hsame *classifier transport row, and the* Pkg *provenance row. No other root certificate row is visible downstream.*

Proof. Begin with the carrier source obligation. It identifies the only source and endpoint rows in the packet, namely F, p, r, H, L, and Π.

Project the finite root ledger L. Its readback rows are exactly the polynomial request, the displayed endpoint, and the carried satisfaction history witnessing that the request has the endpoint as a root. The same ledger also records the hsame transport used by the classifier obligation.

Since the packet has only one provenance coordinate, every visible ledger row returns to $\Pi : \mathsf{Pkg}$. Thus downstream readback is exhausted by the listed finite family. □

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket ledger readback obligation`

Theorem 162.4 (AlgClosureUp root-witness transport coverage)**.** *If two accepted* $\mathsf{AlgClosure}^{\uparrow}$ *root-witness packets classify together, then every visible root-satisfaction row in the first packet transports to a visible root-satisfaction row in the second packet over the transported* $\mathsf{FieldExt}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *sources. The transported row keeps the same* Pkg *provenance, and it introduces no ambient algebraic-closure object and no quotient root.*

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket root witness transport coverage`

Proof. First unpack both root-witness packets through Definition 162.1. Each side exposes only the field-extension source row, the nonconstant polynomial request, the displayed root endpoint, the classifier transport row, the finite ledger, and the package provenance.

Next apply Definition 162.2. The packet classifier transports the $\mathsf{FieldExt}^{\uparrow}$ source first, then reads the $\mathsf{Poly}^{\uparrow}$ request over that transported source, and then carries the endpoint, ledger, and provenance rows along the same comparison.

Finally re-read the target packet ledger with Definition 162.3. The root-satisfaction row is one of the visible ledger rows, so the transported endpoint and polynomial request determine the corresponding visible satisfaction row in the second packet. Because the readback ledger contains only finite BEDC rows and the single Pkg coordinate, no ambient algebraic closure or quotient root is exported. □

Theorem 162.5 (AlgClosureUp standard boundary obligation)**.** *The* $\mathsf{AlgClosure}^{\uparrow}$ *carrier packet contains no host algebraic-closure object, no quotient root object, and no uncarried root set. Its boundary is exactly the finite* BHist, Cont, hsame, $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$, *and* Pkg *surface displayed by the accepted packet.*

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket standard boundary obligation`

Proof. Use Definition 162.3 to list the visible root-certificate rows. Each row is a carried history, a continuation ledger entry, an hsame transport, a consumed naming certificate, or a package provenance row.

A host algebraic closure would require an additional carrier object with global roots for all polynomials. A quotient root object would require an additional quotient equality or orbit representative. Neither field is present in the accepted packet, and neither is part of the readback ledger.

Therefore the boundary is the displayed BEDC surface alone. The chapter records root existence only as finite carried certificates for the requested nonconstant polynomial rows. □

Theorem 162.6 (AlgClosureUp downstream consumer obligation)**.** *Every downstream consumer of an accepted* $\mathsf{AlgClosure}^{\uparrow}$ *packet may read only the finite* $\mathsf{FieldExt}^{\uparrow}$ *source rows,* $\mathsf{Poly}^{\uparrow}$ *nonconstant-polynomial request rows, displayed root rows, classifier transport rows, finite root ledger rows, and* Pkg *provenance rows. A consumer receives no ambient algebraic closure, root quotient, or unlisted host-algebra datum.*

Proof. First apply Definition 162.1. This fixes the carrier-facing rows available to the consumer: the field-extension source, the polynomial request, the root endpoint, transport, ledger, and provenance.

Next apply Definition 162.2. Any consumer-side comparison must use the same $\mathsf{FieldExt}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ source comparisons, the displayed root hsame row, and the transported ledger and provenance rows.

Finally apply Definition 162.5. It removes host algebraic closures, quotient roots, and uncarried root sets from the consumer interface. The downstream surface is therefore exactly the finite row family listed in the statement. □

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket downstream consumer obligation`

162.2 The certificate

Theorem 162.7 (AlgClosureUp root carrier obligation surface)**.** *An* $\mathsf{AlgClosure}^{\uparrow}$ *carrier row is exactly a* $\mathsf{FieldExt}^{\uparrow}$ *source row together with a* $\mathsf{Poly}^{\uparrow}$ *root-ledger row over the BEDC primitives* BHist, hsame, Cont, *and* Pkg. *The carrier exposes the base and extension histories, the nonconstant polynomial request, the displayed root endpoint, the transported classifier row, the finite satisfaction ledger, and the package provenance, and it exposes no ambient algebraic-closure carrier outside those rows.*

Proof. Unpack the seed carrier packet. Its first projection is the consumed $\mathsf{FieldExt}^{\uparrow}$ source, so the base-field and extension histories are already fixed as displayed BHist rows with their classifier transport.

Project the polynomial side next. The consumed $\mathsf{Poly}^{\uparrow}$ request supplies the finite coefficient ledger and the nonconstant-row witness over the same field-extension source, while the root endpoint and satisfaction row are carried as finite histories connected by the displayed hsame and Cont data.

Repack these projections with the single Pkg provenance row. The packet has no constructor for a host algebraic closure or unlisted root carrier, so the listed $\mathsf{FieldExt}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ rows are the whole carrier obligation surface. □

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket root carrier obligation surface`

Theorem 162.8 (AlgClosureUp polynomial root ledger obligation)**.** *Every admitted polynomial-root row in an* $\mathsf{AlgClosure}^{\uparrow}$ *packet projects to the displayed* $\mathsf{Poly}^{\uparrow}$ *coefficient ledger, the inherited* $\mathsf{Field}^{\uparrow}$ *operation rows carried by the* $\mathsf{FieldExt}^{\uparrow}$ *source, and the* hsame *transport that relates the displayed root endpoint to the polynomial satisfaction row. The projection preserves the same finite* BHist *ledger and the same* Pkg *provenance.*

Proof. Start from the admitted root row and read it through Definition 162.1. This gives the $\mathsf{FieldExt}^{\uparrow}$ source, the $\mathsf{Poly}^{\uparrow}$ request, the endpoint, transport, ledger, and provenance coordinates.

The $\mathsf{Poly}^{\uparrow}$ projection is the coefficient ledger for the requested nonconstant polynomial. Since that request is over the displayed field extension, its scalar operations are exactly the inherited $\mathsf{Field}^{\uparrow}$ rows available through the $\mathsf{FieldExt}^{\uparrow}$ source rather than operations from a host field object.

Finally project the root endpoint and satisfaction row from the finite ledger. The hsame coordinate transports the endpoint along the classifier comparison, and the ledger repacks with the same BHist rows and Pkg provenance. No quotient root or choice-selected representative is read by this projection. □

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureTasteGate single carrier alignment`

Theorem 162.9 (AlgClosureUp FieldExt consumer boundary)**.** *Downstream* $\mathsf{FieldExt}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *consumers of an accepted* $\mathsf{AlgClosure}^{\uparrow}$ *packet read only the displayed field-extension rows, polynomial coefficient ledgers, root endpoint rows, satisfaction ledgers,* hsame *transports, finite* Cont *readbacks, and* Pkg *provenance. They do not receive a quotient field, a classical algebraic closure, or a choice-picked root outside the displayed BEDC rows.*

Proof. First apply Definition 162.7. This confines every consumer-facing carrier read to the consumed $\mathsf{FieldExt}^{\uparrow}$ source, the polynomial request, the displayed endpoint, transport, ledger, and provenance rows.

Next apply Definition 162.8. It identifies the coefficient ledger, inherited $\mathsf{Field}^{\uparrow}$ operations, root satisfaction row, and hsame transport available to polynomial consumers.

The remaining consumer read is a boundary question. Use Definition 162.6 to repack the projected rows as the exported surface. Since each projection came from the finite packet coordinates, no quotient field, classical algebraic closure, or choice-picked root is present. □

Theorem 162.10 (AlgClosure public root-witness export)**.** *The carrier obligation, the polynomial-root ledger obligation, and the* $\mathsf{FieldExt}^{\uparrow}$ *consumer boundary jointly export the public* $\mathsf{AlgClosure}^{\uparrow}$ *root-witness surface over* $\mathsf{FieldExt}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* BHist*,* Cont*,* hsame*, and* Pkg*. The exported surface consists exactly of the displayed field-extension source, polynomial request, root endpoint, root-satisfaction ledger, transport rows, finite readbacks, and provenance. It exports no ambient algebraic closure, quotient root, host algebraic-closure object, or choice-picked root.*

Lean4 ✓ `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket public root witness export`

Proof. The carrier obligation supplies the accepted finite packet: a displayed $\mathsf{FieldExt}^{\uparrow}$ source, a $\mathsf{Poly}^{\uparrow}$ request over that source, a root endpoint, transport rows, a finite satisfaction ledger, and Pkg provenance.

The polynomial-root ledger obligation identifies which rows a polynomial consumer may read. It exposes the coefficient ledger, inherited field-extension operations, root endpoint, satisfaction row, and hsame transport without leaving the displayed packet.

Finally apply the $\mathsf{FieldExt}^{\uparrow}$ consumer boundary. It repacks the carrier and ledger rows as the public consumer surface over $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, Cont, hsame, and Pkg. Since no step supplies an ambient algebraic closure, quotient root, host algebraic-closure object, or choice-picked root, none belongs to the exported surface. □

Theorem 162.11 (AlgClosure finite root-witness consumer completeness)**.** *Every finite polynomial-root consumer read available from this* $\mathsf{AlgClosure}^{\uparrow}$ *chapter is exhausted by the displayed* $\mathsf{FieldExt}^{\uparrow}$ *source, the* $\mathsf{Poly}^{\uparrow}$ *request, the root endpoint, the root-satisfaction ledger, the transport rows, the finite readbacks, and the* Pkg *provenance. The consumer surface also contains the component rows needed to read those data: the carrier packet, the polynomial-root classifier obligation, the transported root-witness coverage, the standard boundary, and the public root-witness export.*

There is no ambient algebraic closure, quotient root, host algebraic-closure object, choice-picked root, or unlisted carrier available to such a finite consumer.

Proof. Start with Definition 162.7 and Definition 162.1. These identify the accepted carrier as a finite packet over the displayed $\mathsf{FieldExt}^{\uparrow}$ source, $\mathsf{Poly}^{\uparrow}$ request, root endpoint, root-satisfaction ledger, transports, finite readbacks, and Pkg provenance.

Next apply Definition 162.2 and Definition 162.8. They say that a polynomial-root consumer reads the coefficient and satisfaction ledgers through the same source and classifier rows, with no additional field object or separate root representative.

The transport and boundary rows are supplied by Definition 162.4 and Definition 162.5. These keep the root endpoint, ledger, and provenance inside the finite packet while excluding host algebraic closures, quotient roots, and uncarried root sets.

Finally apply Definition 162.10. The export theorem repacks the carrier, classifier, transport, boundary, and downstream consumer rows as the public finite root-witness surface. Hence every available consumer read is one of the listed BEDC rows, and no ambient algebraic closure, quotient root, host algebraic-closure object, choice-picked root, or unlisted carrier is exposed. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlgClosure}^{\uparrow}}$**.**

Constructive story: A root witness is generated only after a $\mathsf{FieldExt}^{\uparrow}$ extension row and a $\mathsf{Poly}^{\uparrow}$ nonconstant-polynomial request row have both been displayed over the same BEDC source. The carrier accepts the finite packet containing that field-extension row, the polynomial request, a carried root endpoint, the hsame transport row, the finite root-satisfaction ledger, and the single Pkg provenance row; anything needing an ambient algebraic-closure object, a quotient root, or an unlisted root carrier is outside the packet. Two accepted packets classify together by transporting the field-extension source first, reading the polynomial request over that transported source, and then carrying the root endpoint, root ledger, and provenance along the same componentwise comparison. The source specification is the consumed field-extension and polynomial-request surface, the pattern specification is satisfaction of the requested non-constant polynomial by the displayed endpoint, and the ledger keeps the coefficient rows, inherited field operations, root endpoint, satisfaction row, transport, and provenance visible. The five name-certificate obligations are inherited from the dependency classifiers and the finite repacking theorems: an accepted packet inhabits the carrier, hsame gives reflexive, symmetric, and transitive transport componentwise, and transported root-satisfaction rows repack as accepted packets. A packet history h lies in the signature gap of an $\mathsf{AlgClosure}^{\uparrow}$ token p when the source-domain accepts its field-extension and polynomial request rows, the signature relation pairs it with an answer root packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the root endpoint and satisfaction row are exactly the finite root witness encoded by p. The chapter registers this finite polynomial-root packet under the name $\mathsf{AlgClosure}^{\uparrow}$; it introduces no global algebraic-closure carrier, no quotient root object, and no choice-picked host root.

Theory closure: TheoryGate(NameCert$_{\mathsf{AlgClosure}\uparrow}$, maturePackageClosed)

Scope closed: Definition 162.10 binds the public root-witness surface from the carrier, polynomial-root classifier, finite root ledger readback, transport coverage, standard boundary, and downstream consumer rows over FieldExt$^{\uparrow}$, Poly$^{\uparrow}$, BHist, Cont, hsame, and Pkg.

Formal verification: FormalStatus(NameCert$_{\mathsf{AlgClosure}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.AlgClosureUp.AlgClosureCarrierPacket\_public\_root\_witness\_export`

Bridge status: paperBridge

Not claimed: This chapter does not close ambient algebraic closures, quotient roots, host algebraic-closure objects, or unlisted root carriers.

Upgrade path: No paper-axis grade remains; formal bridge checking remains a separate verification-axis obligation for the cited Lean target.

Chapter 163

A Concrete Naming Certificate for $\mathsf{GaloisExt}^{\uparrow}$

The Galois-extension interface $\mathsf{GaloisExt}^{\uparrow}$ packages a normal and separable field extension. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161) and $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ (chapter 166).

163.1 Carrier definition

Definition 163.1 (GaloisExtUp BHist source packet)**.** *A* $\mathsf{GaloisExt}^{\uparrow}$ *source packet is a finite* BHist *tuple* $(F, S, n, s, a, c, \rho, \lambda)$*. The row* F *is the consumed* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *dependency packet, and the row* S *is the consumed* $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ *dependency packet over the same displayed field extension carrier. The row* n *is the normality and root-splitting row: each conjugate endpoint and each displayed root-splitting endpoint is a* BHist *endpoint read through the carrier, classifier, exactness, and ledger rows of* F*. The row* s *is the separability row: it is the minimal-polynomial simple-root row of* S *after transport into the carrier displayed by* F*.*

The row a *is the automorphism source row consumed by the Galois-group interface. It is a finite* BHist *row whose domain and codomain endpoints are read through the FieldExtUp carrier rows, whose base-fixing comparisons are* hsame *rows over the displayed base carrier, whose action on displayed roots is named by a finite* Cont *action ledger, and whose source is recorded by the same* Pkg *provenance that records the FieldExtUp dependency.*

The row c *is the classifier row binding the FieldExtUp and SeparableExtUp dependency packets. It records that their endpoint comparisons are read by* hsame *on the same finite history endpoints, with no host-field equality and no quotient endpoint. The row* ρ *is the* Pkg *provenance showing which dependency packet supplies each displayed face. The finite ledger* λ *is the* Cont *list containing exactly the FieldExtUp dependency row, the SeparableExtUp dependency row, the normality row, the separability row, the automorphism source row, the base-fixing rows, the root-orbit action rows, the classifier row, the stability rows, and the exactness rows consumed by the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *interface.*

Lean4 (definition) `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket`

Definition 163.2 (GaloisExtUp source classifier)**.** *For two* $\mathsf{GaloisExt}^{\uparrow}$ *source packets*

$$P = (F, S, n, s, a, c, \rho, \lambda) \qquad \text{and} \qquad P' = (F', S', n', s', a', c', \rho', \lambda'),$$

the source classifier $P \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P'$ *is the finite agreement row consisting of the following data:*

1. *an* hsame *transport row between the FieldExtUp dependency faces of F and F';*
2. *an* hsame *transport row between the SeparableExtUp dependency faces of S and S' over the same displayed field-extension endpoints;*
3. *an* hsame *transport row carrying the normality and root-splitting row n to n';*
4. *an* hsame *transport row carrying the separability row s to s';*
5. *an* hsame *transport row carrying the automorphism source row a to a';*
6. *an* hsame *transport row carrying the classifier row c to c';*
7. *a shared* Pkg *provenance row comparing ρ with ρ';*
8. *a finite* Cont *ledger comparison comparing λ with λ' entry by entry;*
9. *a* SigRel *row asserting that these transported FieldExtUp, SeparableExtUp, normality, separability, automorphism, classifier, provenance, and ledger faces belong to one finite* $\mathsf{GaloisExt}^{\uparrow}$ *signature surface.*

Lean4 (definition) `BEDC.Derived.GaloisExtUp.GaloisExtSourceClassifier`

Theorem 163.3 (GaloisExtUp source classifier reflexivity)**.** *Every* $\mathsf{GaloisExt}^{\uparrow}$ *source packet* $P = (F, S, n, s, a, c, \rho, \lambda)$ *satisfies* $P \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P$.

Proof. Project the packet to its eight displayed rows. The FieldExtUp dependency face, the SeparableExtUp dependency face, the normality row, the separability row, the automorphism source row, and the classifier row each compare with itself by the reflexive hsame row on the corresponding BHist endpoint.

The provenance and ledger data are retained without alteration. Use the same Pkg row ρ and the same finite Cont list λ, comparing each ledger entry with itself.

Finally repack these componentwise reflexive rows into the SigRel surface named in Definition 163.2. Since every face is the original face of P, the finite signature surface is again the displayed $\mathsf{GaloisExt}^{\uparrow}$ source surface. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourceClassifier reflexive`

Theorem 163.4 (GaloisExtUp source classifier symmetry)**.** *For* $\mathsf{GaloisExt}^{\uparrow}$ *source packets P and P', if* $P \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P'$, *then* $P' \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P$.

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourceClassifier symmetric`

Proof. Unpack the classifier row of Definition 163.2. The FieldExtUp face, the SeparableExtUp face, the normality row, the separability row, the automorphism source row, and the classifier row are all transported by displayed hsame comparisons between finite BHist endpoints.

Reverse each of those comparisons by symmetry of hsame. The shared Pkg provenance row is the same provenance surface read in the opposite direction, and the finite Cont comparison is reversed entry by entry along the same ledger.

The SigRel row already states that the two displayed packets lie on one finite $\mathsf{GaloisExt}^{\uparrow}$ signature surface. Repack the reversed endpoint comparisons, the reversed ledger comparison, and the same signature surface to obtain $P' \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P$. □

Theorem 163.5 (GaloisExtUp source classifier transitivity)**.** *For* $\mathsf{GaloisExt}^{\uparrow}$ *source packets P, P', and P'', if* $P \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P'$ *and* $P' \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P''$, *then* $P \sim^{\mathrm{src}}_{\mathsf{GaloisExt}^{\uparrow}} P''$.

Proof. Unpack the two classifier rows. Their FieldExtUp dependency comparisons share the middle endpoint F', and their SeparableExtUp dependency comparisons share the middle endpoint S' over the same displayed field-extension surface. Transitivity of hsame composes these two dependency transports.

Apply the same transitive hsame composition to the normality rows, the separability rows, the automorphism source rows, and the classifier rows. The middle packet contributes no additional public data; it only supplies the endpoint at which the two transport chains meet.

For provenance and ledgers, compose the displayed Pkg comparisons and the entrywise Cont comparisons through the middle rows. The two SigRel witnesses describe compatible finite $\mathsf{GaloisExt}^{\uparrow}$ signature surfaces sharing the middle packet, so their outer faces repack into one finite signature surface for P and P''. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourceClassifier transitive`

163.2 The certificate

Theorem 163.6 (GaloisExtUp normality obligation row)**.** *For a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the normality face of* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *is accepted exactly when every displayed conjugate witness and every displayed root-splitting witness in* n *is read from the* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *packet* F *through finite* BHist *endpoints,* Cont *ledger rows,* Pkg *provenance rows, and the classifier rows retained by* F*.*

Proof. First read the normality row n from the source packet. By Definition 163.1, its endpoints are not independent field elements; they are displayed BHist endpoints whose carrier, classifier, exactness, and ledger provenance are supplied by F.

For the forward direction, acceptance of the normality face exposes exactly the conjugate and root-splitting entries carried by n. Project each entry to its Cont row and then to its Pkg source. The definition of the packet sends those projections back to the FieldExtUp dependency row, so the accepted face contains only rows read from F.

For the reverse direction, assume every displayed conjugate and root-splitting witness is read from F with the stated history, ledger, and provenance support. Repack those rows as the tuple entry n, attach the classifier row c, and use the same finite ledger λ. This constructs the normality face required by $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ without introducing any endpoint outside the FieldExtUp packet. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket normality obligation row`

Theorem 163.7 (GaloisExtUp separability obligation row)**.** *For a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the separability face of* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *is precisely the minimal-polynomial simple-root row of the consumed* $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ *packet* S*, transported by the classifier row* c *into the same FieldExtUp carrier displayed by* F*.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket separability obligation row`

Proof. Read the row s from the source packet. The definition states that s is the simple-root row of S after it has been placed over the carrier displayed by F.

The classifier row c supplies the needed transport. It compares the FieldExtUp and SeparableExtUp endpoints by hsame on the common BHist source, so the minimal-polynomial endpoint, the root endpoint, and the simple-root witness remain the same finite history rows after transport.

Thus the separability face has no independent certificate data. Projecting it recovers the simple-root row of S, and repacking that row with c, ρ, and λ gives the separability face consumed by $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$. □

Theorem 163.8 (GaloisExtUp classifier transport row)**.** *Let* $(F, S, n, s, a, c, \rho, \lambda)$ *and* $(F', S', n', s', a', c', \rho', \lambda')$ *be two* $\mathsf{GaloisExt}^{\uparrow}$ *source packets. If the FieldExtUp dependency rows of* F *and* F' *are transported by* hsame*, and the SeparableExtUp dependency rows of* S *and* S' *are transported by* hsame *over the same finite* BHist *endpoints, then the normality and separability faces are transported from* (n, s) *to* (n', s') *by the classifier rows* c *and* c' *without host field equality or quotienting.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtClassifier transport row`

Proof. Start with the FieldExtUp transport. The hypothesis identifies the carrier, classifier, exactness, and ledger endpoints of F and F' by hsame. Therefore every conjugate and root-splitting witness read from F has the corresponding displayed endpoint in F'.

Next use the SeparableExtUp transport. It identifies the minimal-polynomial simple-root rows of S and S' over the same history endpoints. The classifier rows c and c' place those rows in the FieldExtUp carriers already matched in the previous paragraph.

Combining the two transports sends n to n' and s to s'. Since all comparisons are hsame comparisons on finite BHist endpoints, the argument never requires equality of ambient host fields, quotient representatives, or an external field carrier. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket classifier transport`

Theorem 163.9 (GaloisExtUp normal separable stability)**.** *Every* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *source packet of Definition 163.1 is stable under replacing its displayed rows by* hsame*-classified* BHist *rows through the same* Pkg *provenance. The stable rows are exactly the* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *field-extension dependency, the* $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ *separability dependency, the finite normality and root-splitting row, the finite automorphism source row, the base-fixing rows, the root-orbit action rows, and the finite* Cont *ledgers belonging to the enclosing* NameCert *packet.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket normal separable stability`
Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket endpoint empty inversion`

Proof. Apply the classifier transport row of Definition 163.8 to the transported FieldExtUp and SeparableExtUp dependency rows. This handles the carrier, classifier, exactness, and ledger endpoints through the inherited classifiers of the two dependency packets.

For the normality component, transport every conjugate and root-splitting row along the displayed hsame comparisons and retain the finite Cont ledger that reads those rows from the field-extension source F.

For the automorphism component, transport the endpoint rows of a along the same FieldExtUp carrier comparisons. The base-fixing rows are already hsame rows over the base carrier, and the action ledger is a finite Cont row, so it is carried by the same provenance-preserving repacking.

For the separability component, use the displayed $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ dependency and its simple-root row. The shared Pkg provenance row ρ is carried through the repacking, so the transported packet has the same certificate surface as the original packet. □

Theorem 163.10 (GaloisExtUp public obligation boundary)**.** *For a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *obligation surface is exactly the following finite row family: the carrier row from* F*, the normality row* n*, the separability row* s*, the automorphism*

source row a, the base-fixing rows over the base carrier, the root-orbit action rows named by the automorphism ledger, the classifier row c, the stability rows inherited from the FieldExtUp and SeparableExtUp packets, the finite Cont *ledger λ, and the exactness rows named by those packets. This is the complete surface consumed by* $\mathsf{GaloisGroup}^{\uparrow}$ *through* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket semantic name certificate`
Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket public obligation boundary`

Proof. Project the packet to its dependency rows. The FieldExtUp packet supplies the carrier, field-extension classifier, stability, exactness, and ledger rows. The SeparableExtUp packet supplies the minimal-polynomial simple-root row, and Definition 163.7 identifies that row with the separability face of the GaloisExtUp surface.

The normality face is supplied by Definition 163.6. The classifier compatibility needed to place both faces over one carrier is supplied by Definition 163.8. The automorphism source row is the tuple entry a, whose endpoints, base-fixing comparisons, action ledger, and provenance are all named by the same FieldExtUp carrier and ledger rows. Thus every public row is a projection of F, S, n, s, a, c, ρ, or λ.

Conversely, each listed row is part of the source packet and is named by the public GaloisExtUp interface: carrier, normality, separability, automorphism, base-fixing, root action, classifier, stability, ledger, and exactness. Repacking these rows gives exactly the $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ surface available to the downstream $\mathsf{GaloisGroup}^{\uparrow}$ carrier, so no additional obligation row remains hidden. □

Theorem 163.11 (GaloisExtUp dependency exactness ledger)**.** *For the source packet $(F, S, n, s, a, c, \rho, \lambda)$ of Definition 163.1, the dependency ledger for* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *has coverage, soundness, and completeness for exactly these displayed rows: the carrier, classifier, stability, exactness, and ledger rows from* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$*; the simple-root row from* $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$*; the finite normality and root-splitting row; the automorphism source row a; the base-fixing* hsame *rows over the base carrier; the finite root-orbit* Cont *action rows; the* hsame *classifier rows; the finite* Cont *readback rows; and the shared* Pkg *provenance row. No other* NameCert *row is exported by the* $\mathsf{GaloisExt}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket dependency exactness ledger`

Proof. Coverage follows by projecting the tuple in Definition 163.1. Each field listed in the ledger is one of the packet entries or one of the dependency rows displayed by $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ or $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$.

Soundness follows from the preceding rows. The normality projection is Definition 163.6, the separability consumption is Definition 163.7, and the classifier and stability steps are Definition 163.8 and Definition 163.9. Each step uses only BHist projection, hsame comparison, finite Cont readback, and the recorded Pkg provenance.

Completeness follows because a downstream consumer can project only those rows from the packet. The tuple contains no separate host field equality, ambient algebraic-closure object, or hidden separability witness. Therefore every accepted $\mathsf{GaloisExt}^{\uparrow}$ dependency read factors through exactly the displayed ledger rows. □

Definition 163.12 (GaloisExtUp automorphism source row)**.** *For a source packet $(F, S, n, s, a, c, \rho, \lambda)$ of Definition 163.1, the automorphism source row is the finite row a consisting of four displayed parts: the automorphism endpoint rows read through the FieldExtUp carrier in F; the base-fixing* hsame *comparisons over the displayed base carrier; the finite* Cont *action ledger on the roots named*

by n; and the Pkg *provenance readback showing that these rows belong to the same source packet as F.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.TasteGate.GaloisExtTasteGate single carrier alignment`

Theorem 163.13 (GaloisExtUp base-fixed automorphism readback)**.** *For a source packet $(F, S, n, s, a, c, \rho, \lambda)$ of Definition 163.1, every automorphism endpoint displayed by a reads back to a FieldExtUp endpoint in F, and every base-fixing comparison displayed by a is an* hsame *row over the base carrier displayed by F. Consequently the automorphism face exported by* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *contains no host-field equality, quotient representative, or external base carrier.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtAutomorphismSourceRow base fixed readback`

Proof. Project the row a using Definition 163.12. Its endpoint entries are declared to be FieldExtUp carrier endpoints, so their domain and codomain readbacks factor through the carrier and ledger rows of F.

The base-fixing part of a is already expressed as hsame over the base carrier displayed by F. Transport along those comparisons keeps the automorphism endpoints inside the same finite BHist packet and does not require an equality of ambient fields.

Finally read the provenance component through ρ and the ledger component through λ. These two projections record the same source packet for the endpoint rows and the base-fixing rows, so the exported automorphism face is exactly the base-fixed automorphism row named by a. □

Theorem 163.14 (GaloisExtUp normal root orbit closure)**.** *For a source packet $(F, S, n, s, a, c, \rho, \lambda)$ of Definition 163.1, the action ledger in a sends each displayed root endpoint named by the normality row n to another root endpoint named by the same row n. The exported root-orbit surface is therefore a finite* Cont *ledger over the FieldExtUp endpoints already used by the normality face.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtAutomorphismSourceRow normal root orbit closure`
↪ variant `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket normal root orbit closure`

Proof. First project the normality row n. By Definition 163.6, each displayed root endpoint in n is read through the FieldExtUp packet F with its classifier, exactness, ledger, and provenance rows.

Next project the action component of a. By Definition 163.12, this component is a finite Cont ledger whose source and target endpoints are the FieldExtUp endpoints named by the same packet.

Composing the two readbacks gives a root endpoint from n as input and a root endpoint from n as output. The Pkg provenance row ρ records that both endpoints and the action row belong to the same GaloisExtUp source packet, so no additional orbit object is exported. □

Theorem 163.15 (GaloisExtUp GaloisGroup consumer source surface)**.** *For a source packet $(F, S, n, s, a, c, \rho, \lambda)$ of Definition 163.1, the part of* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *consumed by* $\mathsf{GaloisGroup}^{\uparrow}$ *is exactly the FieldExtUp carrier rows, the normal root row n, the separability row s, the automorphism source row a, the base-fixing* hsame *rows, the root-orbit* Cont *action rows, the classifier row c, the shared* Pkg *provenance row ρ, and the finite ledger λ.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket galoisgroup consumer source surface`

Proof. The public obligation boundary Definition 163.10 lists the complete GaloisExtUp surface. Projecting that list to the rows needed by a GaloisGroupUp carrier keeps the FieldExtUp carrier data, the normal and separable faces, and the automorphism source row.

The base-fixed readback theorem Definition 163.13 supplies the base-fixing comparisons from a. The root-orbit theorem Definition 163.14 supplies the finite action rows from a and n.

Conversely, these projected rows are precisely the rows of the source packet that a group construction can consume: endpoints from F, roots from n, separability from s, automorphisms from a, classifiers from c, provenance from ρ, and ledgers from λ. Repacking them gives the same consumer surface, so there is no remaining GaloisGroupUp dependency row outside the displayed packet. □

Theorem 163.16 (GaloisExtUp shared carrier coverage)**.** *For every accepted* $\mathsf{NameCert}_{\mathsf{GaloisExt}^\uparrow}$ *source packet* $(F, S, n, s, a, c, \rho, \lambda)$*, the exported carrier data project to one shared* $\mathsf{NameCert}_{\mathsf{FieldExt}^\uparrow}$ *carrier row from* F *and to one* $\mathsf{NameCert}_{\mathsf{SeparableExt}^\uparrow}$ *simple-root row from* S *over the same* BHist *endpoints. The comparison between the two faces is the* hsame *classifier row* c*, so the accepted packet records no host field equality, quotient carrier, or ambient algebraic-closure equality.*

Proof. Project the accepted packet to its FieldExtUp dependency row. The packet definition makes the carrier, classifier, exactness, and ledger endpoints of the GaloisExtUp carrier surface read from F.

Project the same packet to its SeparableExtUp dependency row. The separability row s is the simple-root row of S after classifier transport into the carrier displayed by F, as in Definition 163.7.

It remains to identify the comparison used by the two projections. Apply Definition 163.8 to the displayed dependency rows. The transport is an hsame comparison on the common finite BHist endpoints, so the shared carrier coverage is exactly the pair of projected rows and does not introduce equality of host fields. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket shared carrier coverage`

Theorem 163.17 (GaloisExtUp normal separable combined exactness)**.** *For every source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the paired normality and separability faces of* $\mathsf{NameCert}_{\mathsf{GaloisExt}^\uparrow}$ *have coverage, soundness, and completeness exactly through the FieldExtUp exactness rows of* F*, the SeparableExtUp simple-root row of* S*, the* hsame *classifier row* c*, the finite* Cont *ledger* λ*, and the shared* Pkg *provenance row* ρ*.*

Proof. Coverage is obtained by projecting the two faces from the source packet. The normality and root-splitting row is n, and Definition 163.6 reads it through the FieldExtUp carrier, classifier, exactness, and ledger rows. The separability row is s, and Definition 163.7 reads it as the transported simple-root row of S.

For soundness, combine those two projections with the dependency ledger. By Definition 163.11, every accepted read of the paired faces is backed by the displayed Cont rows, the hsame classifier row, and the recorded Pkg provenance. Hence an accepted normality or separability face cannot name any row outside the FieldExtUp exactness surface and the SeparableExtUp simple-root surface.

For completeness, start with rows from those two dependency surfaces together with c, ρ, and λ. Repack the normality rows as n and the simple-root rows as s. The same ledger theorem states that these are exactly the exported dependency rows of the GaloisExtUp packet, so the paired normality and separability faces are fully recovered. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket normal separable combined exactness`

Theorem 163.18 (GaloisExtUp GaloisGroup consumer boundary)**.** *Every downstream* $\mathsf{GaloisGroup}^\uparrow$ *consumer of* $\mathsf{NameCert}_{\mathsf{GaloisExt}^\uparrow}$ *may read only the shared carrier row, the normality row, the separability row, the automorphism source row, the base-fixing rows, the root-orbit action rows, the*

classifier row, the stability rows, the exactness rows, the finite ledger rows, and the provenance rows exported by the source packet $(F, S, n, s, a, c, \rho, \lambda)$*. No additional host-field, ambient-closure, quotient, or hidden witness row is part of the consumer surface.*

Proof. Begin with the public obligation boundary. By Definition 163.10, the public surface of the certificate consists of the carrier, normality, separability, automorphism, base-fixing, root-orbit, classifier, stability, ledger, and exactness rows named by the source packet.

Next apply the dependency ledger. The exactness statement of Definition 163.11 says that those rows have coverage, soundness, and completeness only through the displayed FieldExtUp dependency, SeparableExtUp simple-root dependency, hsame classifier, Cont readback, and Pkg provenance.

Finally project a $\mathsf{GaloisGroup}^{\uparrow}$ consumer through this public surface. Since the source packet has no additional host-field equality, ambient-closure row, quotient carrier, or hidden witness entry, the consumer boundary is exactly the listed row family exported by $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket galoisgroup consumer boundary`

Definition 163.19 (GaloisExtUp automorphism consumer surface)**.** *The* $\mathsf{GaloisExt}^{\uparrow}$ *automorphism consumer surface exported to* $\mathsf{GaloisGroup}^{\uparrow}$ *is the finite* BHist *row family read from a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1. It consists of the FieldExtUp carrier and classifier rows, the SeparableExtUp simple-root rows transported into that carrier, the normality and root-splitting rows* n*, the base-endpoint fixed rows obtained from the FieldExtUp base endpoints by* hsame *comparison, the automorphism source row* a*, the automorphism-action rows whose source and target extension endpoints and transported root-splitting endpoints are read through the same packet, the finite* Cont *ledgers that read those rows, and the shared* Pkg *provenance row* ρ*.*

No row of this surface is an ambient automorphism object. Each row is either a dependency projection from F *or* S*, a normality or separability face of the GaloisExtUp packet, an* hsame *comparison between displayed endpoints, a finite* Cont *readback row, or the package provenance recording that source.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtUp StdBridge`

Theorem 163.20 (GaloisExtUp fixed-base automorphism source)**.** *For a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, every fixed-base automorphism row accepted by Definition 163.19 is projected only from the FieldExtUp base endpoints of* F *together with the* hsame *comparisons recorded in the finite GaloisExtUp ledger. Conversely, any such finite family of FieldExtUp base endpoints and* hsame *comparisons repacks as a fixed-base row of the automorphism consumer surface, with the same* Pkg *provenance and no host-field equality clause.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket fixed base automorphism source`

Proof. Project the source packet through Definition 163.1. The FieldExtUp row F supplies the displayed extension carrier, its base endpoints, classifier, exactness rows, and finite ledger. The classifier transport row of Definition 163.8 keeps those endpoints inside the same finite BHist comparison system.

For the forward direction, read an accepted fixed-base row from Definition 163.19. Its base endpoint is one of the endpoints exposed by F, and its fixedness assertion is an hsame comparison in the GaloisExtUp ledger λ. The dependency exactness ledger of Definition 163.11 excludes any further NameCert row, so the projection has no additional source.

For the reverse direction, start with FieldExtUp base endpoints and the displayed hsame comparisons. Repack them as the base-endpoint fixed rows of the consumer surface, attach the same Cont readback rows and Pkg provenance ρ, and use the public obligation boundary of Definition 163.10 to place the result inside the exported GaloisExtUp surface. □

Theorem 163.21 (GaloisExtUp automorphism action source)**.** *For a source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, an automorphism-action row is accepted by Definition 163.19 only when its source extension endpoint, target extension endpoint, and transported root-splitting endpoints are all read through the same GaloisExtUp packet. The action row is therefore a finite* BHist *row over* F, S, n, s, a, c, ρ, *and* λ, *not a row imported from an ambient algebraic-closure carrier.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket automorphism action source`

Proof. Begin with an accepted action row. Its source and target endpoints are extension endpoints of the FieldExtUp carrier displayed by F. Its root-splitting endpoints are the normality endpoints of n, and Definition 163.6 reads those endpoints from F by finite BHist projections.

The separability face contributes only the transported simple-root rows of S. By Definition 163.7, these rows enter the action surface after transport by c into the FieldExtUp carrier. Thus the action source, target, and transported splitting data all share the same packet provenance.

Finally apply Definition 163.11. The ledger has no row for an ambient algebraic-closure object or external automorphism carrier. Repacking the displayed source endpoint, target endpoint, root-splitting endpoints, Cont rows, and Pkg provenance therefore gives exactly the accepted automorphism-action row. □

Theorem 163.22 (GaloisExtUp GaloisGroupUp consumer automorphism surface)**.** *For every source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the surface of Definition 163.19 is exactly the* $\mathsf{GaloisExt}^{\uparrow}$ *consumer surface available to* $\mathsf{GaloisGroup}^{\uparrow}$. *It exposes the FieldExtUp carrier and classifier rows, SeparableExtUp simple-root rows, normality and root-splitting rows, fixed-base endpoint rows, the automorphism source row, automorphism-action rows, finite* Cont *ledgers, and* Pkg *provenance. It exposes no host automorphism equality, no quotient of automorphisms, and no ambient algebraic-closure object.*

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtAutomorphismConsumerSurface galoisgroup boundary`

Proof. First project the public GaloisExtUp packet. The public obligation boundary of Definition 163.10 gives the extension carrier, classifier, normality, separability, stability, ledger, exactness, and provenance rows. The dependency exactness ledger of Definition 163.11 says these are all the dependency rows exported by the packet.

Next add the automorphism-facing rows. Fixed-base rows are exactly the projections described in Definition 163.20; automorphism-action rows are exactly the projections described in Definition 163.21. Both constructions use only BHist endpoints, hsame comparisons, finite Cont readback, and Pkg provenance already present in the GaloisExtUp packet.

Conversely, the GaloisGroupUp carrier definition consumes no more than these rows from its GaloisExtUp dependency: its fixed-base and action clauses are read through displayed endpoints and ledgers. Since the GaloisExtUp packet contains no host automorphism equality, no automorphism quotient, and no ambient algebraic-closure object, the listed row family is exactly the consumer automorphism surface available downstream. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket galoisgroup consumer automorphism surface`

Theorem 163.23 (GaloisExtUp public ledger coverage)**.** *For every scoped source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the public ledger visible to* $\mathsf{GaloisExt}^{\uparrow}$ *is exhausted by the* $\mathsf{FieldExt}^{\uparrow}$ *carrier and classifier rows, the* $\mathsf{SeparableExt}^{\uparrow}$ *simple-root rows, the automorphism-source and action rows, the finite* BHist *endpoints, the displayed* hsame *transport rows, the* Cont *ledgers, and the* Pkg *provenance. No public ledger row exports a normal-basis theorem, host quotient field, ambient algebraic closure, or automorphism quotient.*

Proof. Project the scoped source packet through Definition 163.10. The projected packet lists the FieldExtUp, SeparableExtUp, normality, separability, stability, ledger, exactness, and provenance rows available at the public boundary.

Apply Definition 163.11. It identifies those projected rows as the complete dependency ledger and rules out extra host quotient-field, algebraic-closure, and automorphism-quotient slots.

Finally add the automorphism-facing projections of Definition 163.20 and Definition 163.21. These projections are still read through finite BHist endpoints, hsame transports, Cont ledgers, and Pkg provenance, so the listed public ledger is exhausted. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket public ledger coverage`

Theorem 163.24 (GaloisExtUp public interface export)**.** *For every scoped source packet* $(F, S, n, s, a, c, \rho, \lambda)$ *of Definition 163.1, the scoped* $\mathsf{FieldExt}^{\uparrow}$ *carrier and classifier rows,* $\mathsf{SeparableExt}^{\uparrow}$ *simple-root rows, finite* BHist *endpoints, displayed* hsame *transports,* Cont *ledgers,* Pkg *provenance, and automorphism-source rows assemble a public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *interface. The exported interface contains exactly the public rows covered by Definition 163.23; it exports no normal-basis theorem and no host quotient field.*

Proof. First project the scoped source packet. The public obligation boundary of Definition 163.10 supplies the FieldExtUp carrier and classifier rows, the SeparableExtUp simple-root rows, the normality and separability rows, the finite BHist endpoints, the hsame transports, the Cont ledgers, and the Pkg provenance.

Next transport the classifier-facing rows through the already displayed source surface. The source packet theorem Definition 163.10 packages these rows as the scoped NameCert surface, and Definition 163.23 says that the public ledger has no further dependency row to export.

Finally repackage the resulting boundary as a public $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ interface. The automorphism-source rows are exactly the fixed-base and action rows of Definition 163.20 and Definition 163.21; the ledger exactness prevents any normal-basis theorem or host quotient field from entering the exported interface. □

Lean4 ✓ `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket public interface export`

Closure status for $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$**.**

Constructive story: A Galois extension is assembled as a finite history packet whose first face is a $\mathsf{FieldExt}^{\uparrow}$ source row, whose second face is a $\mathsf{SeparableExt}^{\uparrow}$ simple-root row over the same displayed extension, and whose remaining faces record normality, separability, base-fixed automorphism sources, action rows, classifier transport, ledger entries, and package provenance. The carrier accepts exactly packets whose normality and separability faces are read through those dependency certificates, while rows requiring an ambient algebraic closure, host-field equality, or an automorphism quotient have no field in the packet. Two accepted packets classify together only when their FieldExt, SeparableExt, normality, separability, automorphism, classifier, provenance, and finite ledger faces are transported by hsame inside one signature surface. The source specification is the whole accepted packet, the pattern specification is the visible normal-and-separable field-extension shape together with the automorphism source surface, and the ledger keeps each raw dependency row aligned with its visible public face through Cont and Pkg provenance. Since an accepted packet supplies an endpoint, hsame supplies reflexivity, symmetry, and transitivity, and the classifier-transport rows repack any classified endpoint back into the same carrier, the five naming-certificate obligations hold at this scoped surface. A history h lies in the signature GAP of a $\mathsf{GaloisExt}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer

packet and evidence ledger, and token introduction names that answer as p; after unpacking, this says exactly that the FieldExt carrier, SeparableExt simple-root row, normality row, and base-fixed action rows all share the same finite BHist provenance. The chapter registers this construction as $\mathsf{GaloisExt}^{\uparrow}$ without closing the Galois correspondence, algebraic closure, automorphism quotient, or standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface binds the current FieldExtUp carrier and classifier rows, SeparableExtUp simple-root rows, normality and separability obligations, automorphism-source and action rows, BHist endpoints, hsame classifier transport, Cont ledgers, and Pkg provenance surface displayed in this chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.GaloisExtUp.GaloisExtSourcePacket\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close the full Galois correspondence, algebraic closure, automorphism quotient, or standard bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 164

A Concrete Naming Certificate for $\mathsf{GaloisGroup}^{\uparrow}$

The Galois-group interface $\mathsf{GaloisGroup}^{\uparrow}$ packages the group of field automorphisms of a Galois extension fixing the base field pointwise. It depends on $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ (chapter 163) and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41).

164.1 Carrier definition

Definition 164.1 (GaloisGroupUp BHist automorphism-action packet)**.** *A* $\mathsf{GaloisGroup}^{\uparrow}$ *automorphism-action packet is a finite* BHist *tuple* $(E, G, b, a, m, i, c, \rho, \lambda)$ *over two consumed dependency packets. The row* E *is a* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *packet, read through its public extension carrier, normality, separability, classifier, exactness, and ledger surface. The row* G *is a* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *packet, read through its operation, unit, inverse, associativity, congruence, and inverse-law surface.*

The row b *is the fixed-base row. It records, for each displayed base-field endpoint of* E*, that the automorphism endpoint leaves that base endpoint in the same* hsame *class. The row* a *is the automorphism-action row: source and target extension endpoints are finite* BHist *endpoints from* E*, and the action is carried only by* Cont *rows and the displayed classifier of* E*. The unit row is read from the* $\mathsf{Group}^{\uparrow}$ *unit endpoint of* G *and the GaloisExtUp identity classifier of* E*. The row* m *is the composition row, aligning the* $\mathsf{Group}^{\uparrow}$ *multiplication operation of* G *with composition of the displayed automorphism-action rows. The row* i *is the inverse row, aligning the* $\mathsf{Group}^{\uparrow}$ *inverse operation of* G *with inverse automorphism-action rows.*

The row c *is the classifier row. It compares automorphism packets by* hsame *on their finite extension endpoints, fixed-base rows, action rows, composition rows, and inverse rows, without requiring equality of host fields, ambient function spaces, or quotient representatives. The row* ρ *is the* Pkg *provenance row assigning every displayed face to either the Galois-extension packet* E*, the group packet* G*, or the Galois-group automorphism surface. The finite ledger* λ *is the* Cont *list containing exactly the dependency rows, fixed-base rows, identity rows, action rows, composition rows, inverse rows, classifier rows, provenance rows, and exactness rows consumed by the public* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *interface.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket dependency surface`

164.2 The certificate

Theorem 164.2 (GaloisGroupUp fixed-base carrier obligation)**.** *For an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1, carrier acceptance for* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *is exactly acceptance of fixed-base automorphism rows whose extension endpoints are read from the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *surface of* E *and whose operation endpoints are read from the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *surface of* G*. No carrier clause asks for equality of host fields, ambient endomorphism sets, or uncarried base-field elements.*

Proof. Project the packet to E and G. The Galois-extension public boundary of Definition 163.10 supplies the extension carrier, classifier, exactness, normality, separability, and finite ledger rows. The group certificate supplies the operation surface, inverse surface, congruence rows, and inverse-law rows through the $\mathsf{Group}^{\uparrow}$ public packet.

For the forward direction, carrier acceptance exposes a fixed-base automorphism row. Reading the row through Definition 164.1 projects each extension endpoint to E, each multiplication or inverse endpoint to G, and each fixed-base comparison to an hsame row in the finite ledger λ. Thus the accepted carrier is only a displayed BEDC row family over the two consumed packets.

For the reverse direction, start with fixed-base automorphism rows whose endpoints are already read from E and G with the stated ledger and package provenance. Repack those rows as b and a, attach the classifier row c, and retain the same Pkg row ρ and Cont ledger λ. This constructs the carrier face of $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ without adding any host-field equality clause. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket fixed base carrier obligation`

Theorem 164.3 (GaloisGroupUp fixed-base classifier)**.** *For two accepted automorphism-action packets* $(E, G, b, a, m, i, c, \rho, \lambda)$ *and* $(E', G', b', a', m', i', c', \rho', \lambda')$ *of Definition 164.1, the fixed-base automorphism classifier is the conjunction of the inherited* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *classifier on the extension packets, the inherited* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *classifier on the operation packets, and the* hsame *comparisons of the displayed fixed-base, action, composition, and inverse rows. In particular, two rows are classified only through the dependency surfaces and endpoint comparison inside the finite automorphism-action packet.*

Proof. Project each accepted automorphism-action packet to its E and G rows. Definition 163.10 says that the available extension information is precisely the carrier, normality, separability, classifier, stability, ledger, and exactness surface of the extension packet. The group certificate supplies only the operation, unit, inverse, congruence, and inverse-law rows exposed by its public packet.

Now inspect the automorphism-specific rows. The fixed-base, action, composition, and inverse rows are all finite BHist rows inside Definition 164.1. Their comparisons are therefore exactly the restricted hsame rows recorded in the finite ledger λ and its counterpart.

Combining the projected dependency classifiers with these row comparisons gives the stated classifier. Reflexive, symmetric, and transitive uses are inherited from the dependency classifiers and from the structural hsame laws, and repacking those rows returns an accepted fixed-base automorphism-action packet. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket fixed base classifier public surface`

Theorem 164.4 (GaloisGroupUp composition closure)**.** *Let* x *and* y *be accepted fixed-base automorphism-action rows of Definition 164.1. If the target endpoint of* x *is* hsame*-classified with the source endpoint of* y*, and the packet contains a displayed* Cont *row composing their automorphism ledgers,*

then the composed row is again accepted by the $\mathsf{GaloisGroup}^{\uparrow}$ *automorphism-action packet. Its fixed-base witness is obtained by composing the two fixed-base witness rows and transporting the middle endpoint through* hsame*.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket composition closure`

Proof. Read the two accepted rows through the same packet shape. Each supplies a $\mathsf{GaloisExt}^{\uparrow}$ projection, a $\mathsf{Group}^{\uparrow}$ projection, a displayed automorphism-action row, endpoint rows, a fixed-base row, classifier rows, and Pkg provenance through Definition 164.1.

The displayed Cont row concatenates the two automorphism ledgers. The middle endpoint comparison supplies the transport needed to align the target of x with the source of y, so the concatenated ledger has the source endpoint of x and the target endpoint of y.

For the fixed-base face, compose the two fixed-base witness rows. Since each row fixes the base endpoint pointwise inside the inherited GaloisExtUp surface, the composite fixes the same base endpoint after transporting the middle row by hsame.

Finally, repack the dependency projections, composed display, endpoints, fixed-base witness, Cont ledger, classifier rows, and Pkg provenance. The result is one of the finite packets accepted by the carrier, so composition is closed. □

Theorem 164.5 (GaloisGroupUp inverse closure)**.** *Every accepted automorphism-action row of Definition 164.1 exposes an inverse row through the* $\mathsf{Group}^{\uparrow}$ *dependency surface. The inverse row reverses the displayed source and target endpoints, carries the inverse automorphism ledger supplied by* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$*, preserves the same fixed-base endpoint up to* hsame*, and is accepted by the same* $\mathsf{GaloisGroup}^{\uparrow}$ *packet.*

Proof. Unpack the accepted automorphism-action row into its source GaloisExtUp packet, GroupUp packet, displayed action row, endpoints, fixed-base witness, classifier rows, composition ledger, inverse row, and Pkg provenance.

Use the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ dependency cited in chapter 41. Its inverse surface supplies the inverse operation row together with the left-unit and right-unit classification rows for the automorphism ledger. These rows reverse the displayed source and target endpoints.

The fixed-base witness is stable under this reversal because the original row fixes the base endpoint and the inverse ledger is classified with the identity row on that endpoint by the GroupUp inverse laws. Transport the displayed base comparison by hsame to obtain the fixed-base row for the inverse.

Repack the reversed endpoints, inverse ledger, fixed-base witness, inherited dependency projections, classifier rows, and Pkg provenance. This is again a finite packet of the form required by Definition 164.1. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket inverse action boundary`
↪ variant `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket inverse closure`

Theorem 164.6 (GaloisGroupUp unit automorphism row)**.** *For an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1, the unit row inherited from the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *dependency* G *repacks as the fixed-base identity automorphism row on every endpoint accepted by the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *boundary of* E*. The resulting row is classified by the same* $\mathsf{GaloisGroup}^{\uparrow}$ *classifier as the displayed identity action, and its ledger uses only the unit, operation, congruence, and endpoint rows already carried by* G*,* E*, and* λ*.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket unit automorphism row`

Proof. Project the packet by Definition 164.1. The Galois-extension side is the public boundary of Definition 163.10, so every endpoint under discussion is an accepted extension endpoint with classifier, exactness, and finite ledger rows in E.

On the group side, read the operation, unit, associativity, congruence, and two-sided inverse-law surface from the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ packet named in chapter 41. The unit operation endpoint is therefore already a carried row of G.

Repack that unit endpoint as the identity automorphism-action row. Its fixed-base face is the identity comparison on each base endpoint of E, and the same endpoint is transported through the hsame rows in λ. Definition 164.4 supplies compatibility with the composition row, while Definition 164.5 supplies the inverse-facing row needed by the same packet.

The resulting finite tuple has exactly the rows required by Definition 164.1. Hence the inherited GroupUp unit row is the GaloisGroupUp fixed-base identity automorphism row. □

Theorem 164.7 (GaloisGroupUp composition associativity row)**.** *Let x, y, and z be accepted fixed-base automorphism-action rows of an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1. Suppose the target endpoint of x is* hsame*-classified with the source endpoint of y, the target endpoint of y is* hsame*-classified with the source endpoint of z, and the two finite* Cont *parenthesizations $(z \circ y) \circ x$ and $z \circ (y \circ x)$ are both displayed by the composition row m. Then the two parenthesized rows are classified by the inherited* $\mathsf{Group}^{\uparrow}$ *associativity row, transported along the endpoint* hsame *rows from the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *boundary of E.*

Proof. Project x, y, and z through Definition 164.1. The GaloisExtUp public boundary of Definition 163.10 supplies the accepted source, middle, and target endpoints together with the classifier rows that identify the two displayed middle endpoints.

Apply Definition 164.4 first to x and y, then to the resulting row and z. This constructs the finite Cont row for $(z \circ y) \circ x$ with the required fixed-base witness and provenance.

Apply Definition 164.4 in the other order, first to y and z and then to x with the result. This constructs the finite Cont row for $z \circ (y \circ x)$ over the same outer endpoints.

The $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ dependency G supplies the operation, unit, associativity, congruence, and two-sided inverse-law surface named in chapter 41. Its associativity row classifies the two operation parenthesizations in G, and the endpoint hsame rows from E transport that classification to the displayed automorphism-action rows.

Repacking the transported classifier, the two composition ledgers, the fixed-base witnesses, and the Pkg provenance gives the claimed GaloisGroupUp classifier row. □

Theorem 164.8 (GaloisGroupUp inverse cancellation rows)**.** *For every accepted fixed-base automorphism-action row x of an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, let x^{-1} be the inverse row supplied by Definition 164.5. The composed rows $x^{-1} \circ x$ and $x \circ x^{-1}$ are accepted by Definition 164.4 and are classified with the unit automorphism row of Definition 164.6 by the inherited left and right inverse laws of the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *dependency, with endpoint transport through the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *boundary.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionCompositionPacket inverse cancellation rows`

Proof. Start with the accepted row x and unpack it using Definition 164.1. The endpoint rows, fixed-base row, classifier row, inverse row, provenance row, and finite ledger all live in the same packet, while Definition 163.10 supplies the public GaloisExtUp classifier rows needed for endpoint transport.

Use Definition 164.5 to obtain the accepted inverse row x^{-1}. Its source and target are the reversed endpoints of x, and its fixed-base witness is transported by the same hsame comparisons on the base endpoint.

Apply Definition 164.4 to x followed by x^{-1}, and again to x^{-1} followed by x. These applications produce the two finite composition ledgers and preserve the fixed-base witness on both sides.

The $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ packet G supplies the operation, unit, congruence, associativity, and two-sided inverse-law rows from the GroupUp surface named in chapter 41. The right inverse law classifies $x \circ x^{-1}$ with the unit row, and the left inverse law classifies $x^{-1} \circ x$ with the unit row.

Finally, Definition 164.6 identifies that unit row with the fixed-base identity automorphism row over E. Repacking the two inverse-law classifications with the endpoint hsame transports and the ledger λ gives the two claimed GaloisGroupUp cancellation rows. □

Theorem 164.9 (GaloisGroupUp identity row closure)**.** *For an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1, the* $\mathsf{Group}^{\uparrow}$ *unit row of* G*, read through the GaloisExtUp identity classifier of* E *and the fixed-base row* b*, is an accepted* $\mathsf{GaloisGroup}^{\uparrow}$ *automorphism-action row. It fixes every displayed base endpoint by* hsame*, carries only rows listed in* λ*, and uses no equality of host fields or ambient automorphism sets.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket identity row closure`

Proof. Project the packet to its consumed rows E and G. The Galois-extension projection supplies the displayed extension endpoints, the identity classifier, and the exactness rows on the public extension surface. The group projection supplies the unit endpoint, the unit classifier rows, and the provenance needed to read the unit as a finite BHist row.

Now combine those projections with the fixed-base row b. For each displayed base endpoint, the identity classifier leaves the endpoint in its original hsame class, and b records that this comparison is part of the finite automorphism ledger. Thus the unit action fixes every displayed base endpoint.

Finally, repack the base row, identity action row, classifier row c, provenance row ρ, and the same finite ledger λ. The resulting row has the packet shape required by Definition 164.1, so it is accepted by the public $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ carrier. □

Theorem 164.10 (GaloisGroupUp composition classifier congruence)**.** *Let* x, x', y, y' *be accepted fixed-base automorphism-action rows in an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1. Suppose* x *is classified with* x'*,* y *is classified with* y'*, the target endpoint of* x *is classified with the source endpoint of* y*, and the target endpoint of* x' *is classified with the source endpoint of* y'*. Then the composite of* x *with* y *is classified with the composite of* x' *with* y'*. The classification uses only inherited GaloisExtUp classifier transport,* $\mathsf{Group}^{\uparrow}$ *operation congruence,* hsame*,* Cont*, and* Pkg *provenance from the packet.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket composition classifier congruence`
Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionCompositionPacket classifier sameSig congruence`

Proof. First transport the extension faces. The GaloisExtUp classifier of E transports source, middle, and target endpoints across the displayed hsame rows, and the Cont rows in λ record the two compatible composition ledgers.

Next transport the fixed-base faces. Since x is classified with x' and y is classified with y', their fixed-base witnesses compare by the classifier row c. The middle endpoint comparisons align the

two ways of reading the shared extension endpoint, so the composed fixed-base witness is again compared by hsame.

Apply the $\mathsf{Group}^{\uparrow}$ operation congruence from G to the operation endpoints underlying the two composites. The Pkg row ρ identifies these operation endpoints as dependency rows from G, while the acted-on extension endpoints remain dependency rows from E.

Repack the transported source, middle, target, fixed-base, operation, and ledger rows as the composed classifier row. This gives classifier-equivalent composites without introducing any host automorphism equality. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionCompositionPacket classifier congruence`

Theorem 164.11 (GaloisGroupUp automorphism ledger exactness)**.** *For an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, the public ledger λ hides exactly the dependency rows, fixed-base rows, identity rows, action rows, composition rows, inverse rows, classifier rows, provenance rows, and exactness rows exposed by the* $\mathsf{GaloisGroup}^{\uparrow}$ *automorphism surface. Every public consumer of the packet factors through those rows and does not require host automorphism equality, equality of ambient function spaces, or equality of uncarried quotient representatives.*

Proof. Project the packet coordinates. The rows E and G give the dependency surfaces, b gives the fixed-base surface, the unit endpoint of G with the identity classifier of E gives the identity surface, m gives composition, i gives inverse, c gives classifier comparison, and ρ gives provenance. The exactness rows are precisely the exactness rows already named by the dependency packets and the automorphism-action surface.

These projected rows account for every public operation used by a $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ consumer. Carrier acceptance uses the dependency and fixed-base rows; identity uses the unit and identity-classifier rows; composition and inverse use m and i; classifier transport uses c together with hsame and Cont; provenance uses ρ.

Conversely, start from exactly those finite rows and their Pkg assignments. Repacking them as E, G, b, a, m, i, c, ρ, and λ reconstructs the public automorphism-action packet. Since no coordinate contains host automorphism equality, no consumer can depend on such an equality through the public ledger. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket automorphism ledger exactness`

Theorem 164.12 (GaloisGroupUp inverse involution row)**.** *For every accepted fixed-base automorphism-action row x of an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, let x^{-1} be the inverse row supplied by Definition 164.5, and let $(x^{-1})^{-1}$ be the inverse row obtained by applying the same closure row to x^{-1}. Then $(x^{-1})^{-1}$ is accepted by the same packet and is classified with x by the inherited inverse-involutivity row of the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *dependency, transported through the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *endpoint classifier. The classification uses only the fixed-base row b, the inverse row i, the classifier row c, the provenance row ρ, and the finite ledger λ.*

Proof. Unpack the accepted row x through Definition 164.1. Its source and target endpoints are public GaloisExtUp endpoints from E, its operation endpoint is a public GroupUp endpoint from G, and its fixed-base face is an hsame comparison over the displayed base endpoint.

Apply Definition 164.5 to x. The resulting row x^{-1} is accepted by the same packet, has the source and target endpoints reversed, and carries the inverse operation endpoint supplied by the

$\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ surface of G. Its fixed-base face is transported along the same base-endpoint hsame comparisons listed in λ.

Apply Definition 164.5 again to x^{-1}. This gives an accepted row $(x^{-1})^{-1}$ whose displayed endpoints have been reversed back to the orientation of x. The public boundary Definition 163.10 supplies the GaloisExtUp classifier rows identifying these twice-transported endpoints with the original source and target endpoints of x.

On the group side, the inverse operation endpoint of $(x^{-1})^{-1}$ is classified with the operation endpoint of x by the inverse-involutivity law of the GroupUp dependency, for instance Definition 41.57. The same law may be read through the right-inverse presentation of Definition 41.58; both readings land in the classifier surface already named by G.

Transport that GroupUp classifier comparison across the endpoint classifier of E. The fixed-base witnesses match because both x and $(x^{-1})^{-1}$ fix the same displayed base endpoint after the two endpoint transports, and the classifier row c compares exactly the endpoint, fixed-base, action, and inverse faces recorded in the packet. Repacking these comparisons with ρ and λ gives the claimed GaloisGroupUp classifier row between $(x^{-1})^{-1}$ and x. □

Theorem 164.13 (GaloisGroupUp inverse of composition row)**.** *Let x and y be accepted fixed-base automorphism-action rows of an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1. Suppose the target endpoint of y is* hsame*-classified with the source endpoint of x, so that $x \circ y$ is an accepted composite by Definition 164.4. Let $(x \circ y)^{-1}$ be the inverse row supplied by Definition 164.5, and let x^{-1} and y^{-1} be the inverse rows of x and y. Then the row $y^{-1} \circ x^{-1}$ is accepted by the same packet and is classified with $(x \circ y)^{-1}$ by the public GaloisGroupUp classifier. The comparison is the inverse-of-product row of the inherited* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *surface, transported through the endpoint classifier of the public* $\mathsf{NameCert}_{\mathsf{GaloisExt}^\uparrow}$ *boundary.*

Proof. Unpack x and y through Definition 164.1. Their source and target endpoints are public GaloisExtUp endpoints from E, their operation endpoints are public GroupUp endpoints from G, and the displayed middle endpoint comparison is an hsame row in the finite ledger λ.

Apply Definition 164.4 to the composable pair y followed by x. This gives the accepted composite $x \circ y$, with source endpoint the source of y, target endpoint the target of x, fixed-base witness obtained by composing the two fixed-base witnesses, and operation endpoint read through the multiplication row m of G.

Apply Definition 164.5 to this composite. The row $(x \circ y)^{-1}$ is accepted, has the reversed outer endpoints, and carries the inverse operation endpoint of the GroupUp product underlying $x \circ y$. Apply the same inverse-closure row separately to x and to y. The target endpoint of x^{-1} and the source endpoint of y^{-1} are the same middle endpoint comparison, read in the reversed direction through the public boundary of Definition 163.10. Hence another use of Definition 164.4 accepts the composite $y^{-1} \circ x^{-1}$ over the same reversed outer endpoints as $(x \circ y)^{-1}$.

On the GroupUp side, let the operation endpoints of x and y be read as g_x and g_y inside the packet G. The composition row m reads the operation endpoint of $x \circ y$ as the product $g_x g_y$, while the inverse row i reads the operation endpoints of x^{-1} and y^{-1} as g_x^{-1} and g_y^{-1}. The inherited inverse-of-product law Definition 41.65 gives the internal classifier row

$$\mathsf{hsame}((g_x g_y)^{-1}, g_y^{-1} g_x^{-1}).$$

Thus the GroupUp operation endpoint carried by $(x \circ y)^{-1}$ is classified with the GroupUp operation endpoint carried by $y^{-1} \circ x^{-1}$.

Transport this GroupUp classifier comparison through the GaloisExtUp endpoint classifier. The source endpoint of $(x \circ y)^{-1}$ and the source endpoint of $y^{-1} \circ x^{-1}$ are both the target endpoint

of x, and their target endpoints are both the source endpoint of y, up to the displayed hsame rows from E. The fixed-base witnesses also match: inverse closure preserves the fixed-base comparison for each row, and composition closure composes those fixed-base comparisons along the same middle endpoint transport.

Finally use the classifier surface described in Definition 164.3. Its clauses compare exactly the dependency classifiers, extension endpoints, fixed-base rows, action rows, composition rows, inverse rows, and Pkg provenance recorded by $(E, G, b, a, m, i, c, \rho, \lambda)$. Repacking the endpoint transports, the GroupUp inverse-of-product classifier, the two composition ledgers, the fixed-base witnesses, and the shared provenance row ρ gives the claimed GaloisGroupUp classifier row between $(x \circ y)^{-1}$ and $y^{-1} \circ x^{-1}$. □

Theorem 164.14 (GaloisGroupUp identity automorphism row)**.** *For an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, the unit row of the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *surface of G, together with the classifier and endpoint rows of the* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *surface of E, determines an accepted fixed-base identity automorphism row on every displayed extension endpoint. The row is a finite* BHist *row whose action ledger is a* Cont *identity readback, whose fixed-base witness is repacked through* hsame*, and whose* Pkg *provenance remains the pair of dependency packets E and G.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket identity automorphism row`

Proof. Project the packet to its Galois-extension row E and group row G. The $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ dependency supplies the unit endpoint and the group unit surface, while $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ supplies the displayed extension endpoints, classifier rows, exactness rows, and finite endpoint ledger.

Choose one displayed extension endpoint of E. The identity action keeps that source and target endpoint in the same hsame class, so its action face is a finite BHist row read by the singleton Cont ledger for the endpoint. The fixed-base face is the same base endpoint comparison already recorded by the GaloisExtUp classifier.

Repack the unit endpoint from G, the identity action row over E, the fixed-base hsame witness, the classifier row c, the provenance row ρ, and the ledger λ. This is exactly one accepted fixed-base automorphism-action row of the packet, with no ambient automorphism equality added. □

Theorem 164.15 (GaloisGroupUp composition associativity ledger)**.** *Let x, y, and z be accepted fixed-base automorphism-action rows of Definition 164.1 whose displayed source and target endpoints are composable by* hsame *comparisons in the* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *packet E. The two parenthesized finite* Cont *composition ledgers, first composing x with y and then with z, or first composing y with z and then with x, classify the same final extension endpoint. The comparison is the inherited associativity row of* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *transported through the GaloisExtUp endpoint classifier, with the same* BHist *rows and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket composition associativity public surface`

Proof. Read x, y, and z through the common packet. Each row exposes a BHist source endpoint, target endpoint, fixed-base witness, action ledger, classifier row, and Pkg provenance, while the composability assumptions give the two middle hsame transports inside E.

Form the two finite Cont ledgers. On the group side, the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ associativity row identifies the two parenthesized operation rows. On the extension side, the $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$

classifier transports the intermediate endpoints so both ledgers have the same source endpoint from x and the same target endpoint from z.

The fixed-base witnesses compose along the same endpoint transports, so both parenthesized rows keep every displayed base endpoint in its original hsame class. Repacking the common endpoints, the reassociated ledger, the classifier row, and the unchanged provenance row ρ gives a single GaloisGroupUp classifier comparison between the two ledgers. □

Theorem 164.16 (GaloisGroupUp unit action laws)**.** *For every accepted fixed-base automorphism-action row x of Definition 164.1, composing x on the left or on the right with the identity row of Definition 164.14 classifies back to x. Both unit compositions are finite* BHist *rows: their* Cont *ledgers use the left and right unit rows of* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$*, their endpoint transports use the* $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ *classifier, their fixed-base witnesses are carried by* hsame*, and their* Pkg *provenance is unchanged.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket unit action laws`

Proof. Start with the accepted row x and project its dependency data. The identity row supplied by Definition 164.14 has the same displayed endpoint system from E and the unit endpoint from G, so it can be composed with x on either side whenever the finite endpoint ledgers match by hsame.

For the left composition, use the left unit law in the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ surface and transport the acted-on endpoint through the $\mathsf{NameCert}_{\mathsf{GaloisExt}^{\uparrow}}$ classifier. For the right composition, use the right unit law and the same endpoint classifier. In both cases the resulting Cont ledger is classified with the original action ledger of x.

It remains to check the fixed-base face. The fixed-base carrier obligation of Definition 164.2 says this face is only the displayed base endpoint plus its hsame comparison, and the identity row preserves that comparison. Repacking the original row x, the unit law transport, and the same Pkg provenance gives the two stated unit classifications. □

Theorem 164.17 (GaloisGroupUp composition and inverse obligation)**.** *For an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, the composition row m and inverse row i remain in the same Galois-group classifier exactly when their displayed endpoints are transported by finite* Cont *rows,* hsame *comparisons, and* Pkg *provenance from the Galois-extension packet E and the group-operation packet G.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket composition inverse obligation`

Proof. First isolate the composition row. The operation endpoint in m is the $\mathsf{Group}^{\uparrow}$ multiplication row, while the acted-on endpoints are extension histories from E. Group congruence transports the multiplication endpoint, and the Galois-extension classifier transports the acted-on endpoints. The ledger λ records both transports as finite Cont rows.

Next isolate the inverse row. The inverse endpoint in i is the $\mathsf{Group}^{\uparrow}$ inverse row, and the inverse action endpoints are again displayed extension histories from E. The inverse congruence and inverse laws from the group surface keep the operation in the same classifier, while the fixed-base row b keeps the base endpoints in their original hsame classes.

Combining the two rows gives the stated equivalence. If composition and inverse endpoints are transported by Cont, hsame, and Pkg, then m and i repack into the classifier row c. Conversely, projecting c exposes precisely those finite transports and no additional ambient automorphism equality. □

Theorem 164.18 (GaloisGroupUp identity action row)**.** *For an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1, the unit row of the group packet* G *gives a displayed identity automorphism-action row on every extension endpoint carried by* E*. This row fixes every displayed base-field endpoint by* hsame *and is accepted by the same finite* $\mathsf{GaloisGroup}^{\uparrow}$ *automorphism-action packet using only* BHist*,* Cont*,* Pkg*, and the inherited* $\mathsf{GaloisExt}^{\uparrow}$ *classifier.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket identity action row`

Proof. Unpack the automorphism-action packet. By Definition 164.1, the only available dependency data are the Galois-extension packet E, the group packet G, the fixed-base row b, the action row a, the composition row m, the inverse row i, the classifier row c, the package row ρ, and the finite ledger λ.

The public Galois-extension boundary of Definition 163.10 supplies the extension endpoints and their classifier rows. The fixed-base carrier obligation of Definition 164.2 supplies the hsame base-field comparison for a displayed automorphism row over those endpoints.

Read the unit endpoint from the $\mathsf{Group}^{\uparrow}$ dependency packet G. Its displayed unit ledger acts as the identity action on each extension endpoint, and the fixed-base row transports the base endpoint back to its original hsame class. The composition and inverse controls of Definition 164.4, Definition 164.5, and Definition 164.17 ensure that this unit row lies in the same accepted automorphism-action classifier as the rows already carried by the packet.

Repack the dependency projections, unit action ledger, fixed-base witness, classifier row, Cont readbacks, and shared Pkg provenance. The result is a finite accepted row of the original packet, so the identity action is public without exporting a host automorphism function. □

Theorem 164.19 (GaloisGroupUp associative composition row)**.** *For three composable accepted automorphism-action rows in an automorphism-action packet of Definition 164.1, the left-associated and right-associated composite ledgers are classified by the same* $\mathsf{GaloisGroup}^{\uparrow}$ *row. The comparison is obtained from the* $\mathsf{Group}^{\uparrow}$ *associativity row together with the middle-endpoint* hsame *transports exposed by the* $\mathsf{GaloisExt}^{\uparrow}$ *surface.*

Proof. Unpack the three accepted action rows through Definition 164.1. Each row has extension endpoints from E, group-operation endpoints from G, fixed-base witnesses, classifier entries, Cont ledger entries, and the shared Pkg provenance row.

The endpoint alignments come from the Galois-extension public boundary Definition 163.10. Those classifier rows transport the first middle endpoint to the second source endpoint and the second middle endpoint to the third source endpoint inside the finite ledger λ.

Apply the composition closure of Definition 164.4 twice, once to form the left-associated ledger and once to form the right-associated ledger. The group packet G contributes the $\mathsf{Group}^{\uparrow}$ associativity row, and Definition 164.17 identifies the two composition ledgers through the same finite Cont and hsame transports.

The inverse closure of Definition 164.5 supplies the same accepted classifier controls for the operation endpoints that appear in the association comparison. Repack the two associated ledgers, their fixed-base witnesses, the dependency projections, and the shared package row. The repacked classifier proves that associativity is an action-ledger row of the original finite packet. □

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket associative action row`

Theorem 164.20 (GaloisGroupUp action ledger exactness)**.** *For an automorphism-action packet* $(E, G, b, a, m, i, c, \rho, \lambda)$ *of Definition 164.1, the public action ledger is exactly the fixed-base row,*

identity row, composition row, inverse row, classifier row, dependency projections to E and G, finite Cont *readbacks, and shared* Pkg *provenance. No host automorphism function space, quotient of automorphisms, or uncarried equality of field presentations is exported.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket action ledger exactness`

Proof. Project the packet along Definition 164.1. The projection exposes only E, G, b, a, m, i, c, ρ, and λ together with the dependency rows recorded inside the two consumed NameCert packets.

The Galois-extension public boundary Definition 163.10 accounts for all extension carrier, classifier, exactness, and ledger rows from E. The fixed-base carrier obligation Definition 164.2 accounts for the base-fixing face of the action ledger.

The identity row is the unit action row supplied above. The composition and inverse faces are exactly the rows controlled by Definition 164.4, Definition 164.5, and Definition 164.17. These references exhaust the accepted action, association, and reversal data available in the finite packet.

Conversely, every listed row is a displayed packet row or a dependency projection already carried by λ and tagged by ρ. Repacking those rows gives the same public $\mathsf{GaloisGroup}^{\uparrow}$ automorphism-action packet, so the ledger is exact and has no additional ambient automorphism object behind it. □

Theorem 164.21 (GaloisGroupUp public NameCert obligation boundary)**.** *For an automorphism-action packet $(E, G, b, a, m, i, c, \rho, \lambda)$ of Definition 164.1, the public $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ obligation surface is exactly the following finite row family: the Galois-extension dependency packet E, the group dependency packet G, the fixed-base row b, the automorphism-action row a, the identity row read from the unit of G, the composition row m, the inverse row i, the classifier row c, the* Pkg *provenance row ρ, the finite* Cont *ledger λ, and the exactness rows named by those packets. This is the public obligation boundary consumed by downstream Galois and representation-facing chapters.*

Lean4 ✓ `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket semantic name certificate`

Proof. Project the packet to its dependency surfaces. The Galois-extension side is the public boundary of Definition 163.10; the group side is the operation and inverse surface named by the $\mathsf{Group}^{\uparrow}$ certificate. These two dependency projections account for the extension carrier, group operation, classifier, stability, ledger, and exactness rows available to the Galois-group packet.

The fixed-base carrier obligation is supplied by Definition 164.2. The identity row is supplied by Definition 164.9; classifier-stable composition is supplied by Definition 164.10; and the composition and inverse obligation is supplied by Definition 164.17. Therefore every public row is a projection of E, G, b, a, the identity row, m, i, c, ρ, or λ.

Conversely, each listed row is either a coordinate of Definition 164.1 or a projected row read from those coordinates. Repacking those rows gives the carrier, identity, operation, inverse, classifier, provenance, ledger, and exactness surface needed by downstream Galois and representation-facing uses, so no hidden host automorphism datum is part of the public boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$**.**

Constructive story: A Galois group here is not a host automorphism set; it is a finite automorphism-action packet built from a $\mathsf{GaloisExt}^{\uparrow}$ source row, a $\mathsf{Group}^{\uparrow}$ law row, fixed-base rows, action rows, composition and inverse rows, and the continuation ledgers that concatenate these rows to a public endpoint. The carrier accepts exactly those packets whose extension, group, fixed-base, action, composition, and inverse faces are unary histories tied by Cont to provenance, classifier, ledger, and endpoint rows, while quotient automorphism carriers and arbitrary ambient groups are outside the packet. Two accepted action packets classify together when the fixed-base row, action row, composition row, inverse row, provenance row, and endpoint row move by hsame through the same finite readback ledgers. The source specification is the accepted automorphism-action packet, the pattern specification records base-fixing, action, composition, inverse,

identity, and group-law rows inherited from GaloisExt$^{\uparrow}$ and Group$^{\uparrow}$, and the ledger keeps the raw action data visible as the same provenance and law endpoints. The five naming-certificate obligations come from the displayed packet endpoint for inhabitedness, the equivalence laws of hsame, and the transport theorems that rebuild a carrier packet after componentwise classifier movement. A history h lies in the signature GAP of a GaloisGroup$^{\uparrow}$ token p when the source accepts h, the signature relation pairs it to an answer action packet and evidence ledger, and token introduction identifies that answer with p; concretely, h must be the endpoint produced by the extension-plus-group provenance row and the composition-plus-inverse ledger for the same base-fixed action surface. The chapter registers this construction as GaloisGroup$^{\uparrow}$ without supplying a host automorphism group, an automorphism quotient, arbitrary Galois-group carriers, or a standard bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{GaloisGroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped surface is the four-file GaloisGroupUp obligation package: the automorphism action carrier, fixed-base classifier, group-law rows, inverse rows, and ledger exactness are all bound to the GaloisExtUp and GroupUp source certificates.

Formal verification: FormalStatus(NameCert$_{\mathsf{GaloisGroup}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GaloisGroupUp.GaloisGroupAutomorphismActionPacket\_semantic\_name\_certificate`

Bridge status: none

Not claimed: The chapter does not close host automorphism groups, quotient automorphism carriers, arbitrary Galois groups, or a standard bridge.

Upgrade path: No higher paper-closure grade remains for this chapter; standard bridge status remains separately reported by `\bridgestatus`.

Chapter 165

A Concrete Naming Certificate for $\mathsf{SplittingField}^{\uparrow}$

The splitting-field interface $\mathsf{SplittingField}^{\uparrow}$ packages the smallest field extension over which a given polynomial factors into linear terms. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

165.1 Carrier definition

Definition 165.1 (SplittingFieldUp root carrier packet)**.** *A* $\mathsf{SplittingField}^{\uparrow}$ *root carrier packet is a finite* BHist *packet*

$$\mathcal{S} = (E, P, R, F, C, H, \Pi).$$

The row E is the consumed $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *source row. The row P is the consumed* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *polynomial row over that source. The row R records the displayed root endpoints for the carried polynomial, and the row F records the displayed factor rows whose product is the polynomial row P inside the carried extension.*

The finite ledger C is the Cont *factorization ledger. The row H records the* hsame *transport witnesses for the field-extension source, polynomial row, root endpoints, and factor rows. The row Π is the* Pkg *provenance packet recording exactly the consumed FieldExtUp source, PolynomialUp source, endpoint, factorization, continuation, and transport rows.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket provenance transport`

165.2 The certificate

Theorem 165.2 (SplittingFieldUp root classifier obligation)**.** *For a root carrier packet $\mathcal{S} = (E, P, R, F, C, H, \Pi)$ of Definition 165.1, the public classifier is determined by the FieldExtUp source row E, the PolynomialUp factor row P, the displayed root endpoints R, the factor rows F, and the* hsame *transport witnesses H. It does not classify quotient fields, ambient algebraic closures, or host polynomial roots not present in the carried ledger.*

Proof. Project the packet to E and P. The source of the classifier is the consumed field-extension certificate together with the consumed polynomial certificate, so every public read begins from those two BHist rows.

Project next to the root and factorization part. The rows R and F are the only endpoint and factor rows carried by the packet, and H is the only transport datum allowed to compare such rows.

The reverse direction repacks the same E, P, R, F, H rows into the packet classifier. Since no carrier field names a quotient field, algebraic closure, or host root outside R, those objects cannot be public classifier data. □

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket classifier obligation`

Theorem 165.3 (SplittingFieldUp root ledger obligation)**.** *Every public ledger read of a* $\mathsf{SplittingField}^{\uparrow}$ *root carrier packet is a row of the factorization continuation ledger C or the provenance packet Π attached to E, P, R, F, H. Conversely, every listed continuation or provenance row repacks into the same public splitting-field packet.*

Proof. Project the Cont component C. It records finite factorization reads for the polynomial row P, the root endpoints R, the factor rows F, and the transport witnesses H.

Project the Pkg component Π. Its provenance entries are exactly the consumed FieldExtUp row, the consumed PolynomialUp row, the endpoint rows, the factor rows, the continuation rows, and the transport rows named in the carrier.

For the converse, take any row on that finite list. The carrier packet Definition 165.1 has a field for it, so it reconstructs the same public splitting-field ledger without adding quotient fields, algebraic closures, or host polynomial roots. □

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket root ledger obligation`
↪ variant `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket ledger obligation`

Theorem 165.4 (SplittingFieldUp root NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *public surface for the root packet is the finite packet consisting of the FieldExtUp source row, PolynomialUp factor row, root endpoint rows, factorization rows,* Cont *factorization ledger,* hsame *transport witnesses, and* Pkg *provenance rows of Definition 165.1. This surface exports no quotient-field carrier, algebraic-closure carrier, or host-level root set.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket root namecert surface`

Proof. Use Definition 165.2 to fix the classifier from E, P, R, F, H. These are the only source, polynomial, endpoint, factorization, and transport rows available to the public surface.

Use Definition 165.3 to fix the ledger from C and Π. The continuation and provenance rows exhaust all public factorization reads attached to the classifier rows.

Repack the classifier and ledger rows through Definition 165.1. The result is exactly the public $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ surface, and all host-level splitting objects are excluded because the packet has no field for them. □

Theorem 165.5 (SplittingFieldUp root downstream threshold)**.** *Downstream CyclotomicUp and algebraic-field consumers of* $\mathsf{SplittingField}^{\uparrow}$ *may read exactly the root NameCert surface of Definition 165.4: the FieldExtUp source, PolynomialUp polynomial row, root endpoints, factor rows, continuation ledger, transport witnesses, and provenance packet. No downstream consumer receives a quotient field, algebraic closure, or host polynomial-root object unless it is represented by one of those carried rows.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket root downstream threshold`

Proof. Begin with Definition 165.4. It packages the classifier and ledger rows into the public splitting-field surface seen by consumers.

For a CyclotomicUp consumer, the polynomial row and root endpoint rows are read through P and R, while the factorization requirement is read through F and C. For an algebraic-field consumer, the field-extension source and provenance rows are read through E and Π.

The two consumer paths therefore share the same finite packet. Since Definition 165.2 and Definition 165.3 exclude quotient fields, algebraic closures, and host roots outside the packet, the downstream threshold is exactly the carried root surface. □

Theorem 165.6 (SplittingFieldUp root classifier transport)**.** *For a* $\mathsf{SplittingField}^{\uparrow}$ *root carrier packet*

$$\mathcal{S} = (E, P, R, F, C, H, \Pi),$$

the root classifier transports exactly the displayed FieldExtUp source row E, PolynomialUp row P, root endpoint rows R, factor rows F, and transport row H along hsame*. The transported classifier keeps the same factorization ledger C and the same provenance packet Π, and it exports no quotient field, algebraic closure, or host polynomial-root object.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket root classifier transport`

Proof. Project the classifier through Definition 165.2. Its public data are the FieldExtUp source row, the PolynomialUp row, the displayed roots, the displayed factors, and the hsame transport witnesses.

The carrier packet of Definition 165.1 keeps C and Π attached to those same rows. Transport therefore changes only the displayed endpoints along H and leaves the finite continuation ledger and provenance packet in place.

Repack the transported E, P, R, F, H rows with the same C and Π. Because the classifier obligation has no field for a quotient field, algebraic closure, or host root set, no such object is transported. □

Theorem 165.7 (SplittingFieldUp root factor-ledger exhaustion)**.** *Every public factorization read from a* $\mathsf{SplittingField}^{\uparrow}$ *root packet $\mathcal{S} = (E, P, R, F, C, H, \Pi)$ is exhausted by the finite* Cont *ledger C and the provenance packet Π. Conversely, every factor row listed in F with its root endpoint in R, transport row in H, continuation entry in C, and provenance entry in Π repacks as an accepted public factorization read of the same packet.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket factor ledger exhaustion`

Proof. Read the ledger obligation Definition 165.3. It says that public ledger reads are rows of C or provenance rows of Π attached to the displayed source, polynomial, root, factor, and transport data.

Induct over the finite factor rows F. In the empty case, C and Π have no factor row to bind, so no public factorization read can be projected.

In the cons case, the head factor is tied to the polynomial row P, a root endpoint in R, a transport witness in H, a continuation entry in C, and a provenance entry in Π. The induction hypothesis exhausts the tail factors through the same finite ledgers.

For the converse, attach each listed factor row to its displayed endpoint, transport, continuation, and provenance entries, then repack through Definition 165.1. The result is the same public factorization packet and adds no host-level factor source. □

Theorem 165.8 (SplittingFieldUp Cyclotomic consumer source boundary)**.** *A* $\mathsf{Cyclotomic}^{\uparrow}$ *consumer reading from* $\mathsf{SplittingField}^{\uparrow}$ *may consume exactly the FieldExtUp source row* E*, PolynomialUp row* P*, root endpoint rows* R*, factor rows* F*, finite factorization ledger* C*,* hsame *transport row* H*, and* Pkg *provenance packet* Π *of a carried packet* $\mathcal{S} = (E, P, R, F, C, H, \Pi)$*. These are the only root, factor, polynomial, FieldExtUp,* Cont*,* hsame*, and* Pkg *rows available at the consumer boundary.*

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket cyclotomic consumer source boundary`

Proof. Begin with Definition 165.5. It already restricts downstream consumers to the root NameCert surface consisting of the FieldExtUp, PolynomialUp, root, factor, continuation, transport, and provenance rows.

Apply Definition 165.6. This fixes the consumer's transport reads to the displayed E, P, R, F, H classifier packet, with the same C and Π still attached.

Apply Definition 165.7. This exhausts the factor side by the finite continuation ledger C and provenance packet Π, so every factor read consumed by $\mathsf{Cyclotomic}^{\uparrow}$ is already a row of the carried packet.

Combining these projections gives the exact consumer boundary. No quotient field, algebraic closure, host root set, or host factorization source is available unless it is represented by one of E, P, R, F, C, H, Π. □

Theorem 165.9 (SplittingFieldUp standard bridge boundary)**.** *The public root packet* $\mathcal{S} = (E, P, R, F, C, H, \Pi)$ *described by the* $\mathsf{SplittingField}^{\uparrow}$ *carrier gives the BEDC-side boundary of the standard splitting-field reading. Roots and linear factors are read only from the displayed* $\mathsf{FieldExt}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*, root endpoint, factor,* Cont*,* hsame*, and* Pkg *rows of that packet. The bridge introduces no quotient fields, algebraic-closure carrier, host root set, or external factorization.*

Proof. Start from the root NameCert surface of Definition 165.4. It exposes the accepted packet as the displayed E, P, R, F, C, H, Π rows and no additional field, polynomial, root, or factor carrier.

Apply Definition 165.7. Every factor read in the standard interpretation is forced through the finite continuation ledger C and the package provenance Π, so the factor side is already exhausted by displayed packet data.

Use the downstream threshold of Definition 165.5 together with the consumer boundary of Definition 165.8. These project every standard root, linear-factor, transport, and provenance read to one of the displayed rows.

Finally invert the carrier of Definition 165.1. Since the carrier has no coordinate for quotient fields, algebraic closures, host root sets, or external factorization procedures, those data cannot be introduced by the BEDC-side bridge boundary. □

Lean4 ✓ `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket standard bridge boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$.

Constructive story: A splitting field is presented as a root carrier packet whose entries are a $\mathsf{FieldExt}^{\uparrow}$ source row, a $\mathsf{Poly}^{\uparrow}$ row over that source, displayed root endpoints, displayed factor rows, a finite factorization ledger, transport witnesses, and package provenance. The carrier accepts only packets where those rows are unary histories tied by the stated Cont displays, so quotient-field carriers, algebraic-closure carriers, and host root sets never become accepted inputs. Its classifier is the transported agreement of the field-extension row, polynomial row, root endpoints, factor rows, and transport row along hsame, with the same factorization ledger and provenance kept attached. The source specification is the whole root packet, the pattern specification is the visible factorization of the carried polynomial into the listed linear factors and roots, and the ledger records exactly the continuation and provenance rows that make each raw factor read visible. At this obligation boundary, inhabitedness is supplied by an accepted packet when one is displayed, reflexivity,

symmetry, and transitivity are the hsame rows demanded of the classifier, and carrier closure is the transport obligation that repacks the same E, P, R, F, C, H, Π data after classification. A history h lies in the signature GAP of a SplittingField$^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer root packet and evidence ledger, and token introduction names that answer as p; here this contracts to the claim that h is exactly the endpoint whose FieldExt row, Polynomial row, roots, factors, Cont ledger, hsame transport, and Pkg provenance are the carried packet rows. The chapter registers this construction as SplittingField$^{\uparrow}$ without adding a quotient field, an algebraic-closure object, a host-level root set, or a standard bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{SplittingField}^{\uparrow}}$, maturePackageClosed)

Scope closed: The exported public interface is exactly the root NameCert surface of Definition 165.4 together with the downstream threshold of Definition 165.5: the FieldExtUp source row, PolynomialUp row, root endpoints, factor rows, Cont factorization ledger, hsame transport, Pkg provenance, and the Cyclotomic and algebraic-field consumer boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{SplittingField}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SplittingFieldUp.SplittingFieldRootCarrierPacket\_standard\_bridge\_boundary`

Bridge status: none

Not claimed: No quotient-field carrier, algebraic-closure carrier, host root set, or `bridgeCheckedV` formal status is claimed.

Upgrade path: No further paper theory-axis step is recorded for this chapter; bridge verification remains on the separate formal axis.

Chapter 166

A Concrete Naming Certificate for $\mathsf{SeparableExt}^{\uparrow}$

The separable-extension interface $\mathsf{SeparableExt}^{\uparrow}$ packages a field extension whose minimal polynomials all have simple roots. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

166.1 Carrier definition

166.2 The certificate

Theorem 166.1 (Separable-extension FieldExt and Polynomial source surface)**.** *A* $\mathsf{SeparableExt}^{\uparrow}$ *source row is the joint BEDC source consisting of a* $\mathsf{FieldExt}^{\uparrow}$ *extension packet, a* $\mathsf{Poly}^{\uparrow}$ *minimal-polynomial packet over that extension, and shared provenance rows in* BHist *and* Pkg*. The two dependency witnesses are scoped by* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*, and the shared provenance requires the displayed base-field, extension-field, generator, and minimal-polynomial histories to be compared by* hsame *inside the same package ledger.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceRow fieldext polynomial source`

Proof. Project the $\mathsf{FieldExt}^{\uparrow}$ dependency certificate to obtain the base row, extension row, extension classifier, and package provenance. Project the $\mathsf{Poly}^{\uparrow}$ dependency certificate to obtain the polynomial carrier, coefficient ledger, and polynomial classifier over the displayed extension row.

The source row for $\mathsf{SeparableExt}^{\uparrow}$ is then the common pullback of these two projections over BHist and Pkg. The hsame rows identify the base and extension histories seen by the polynomial packet with the histories supplied by the field-extension packet, so the two dependencies describe one BEDC source rather than two unrelated presentations. □

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtJointSource fieldext polynomial source`
Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface dependency ledger closure`
Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceRow classifier stability`

Theorem 166.2 (Separable-extension carrier and classifier surface)**.** *The carrier surface for separable-extension histories consists of the joint source row of Definition 166.1 together with the displayed generator and minimal-polynomial histories. Two such carrier rows are classified by the inherited*

FieldExt$^{\uparrow}$ *classifier on the extension packet and by the* Poly$^{\uparrow}$ *equality classifier on the displayed minimal-polynomial rows, with the shared* BHist *and* Pkg *provenance transported along* hsame*.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceRow carrier classifier surface`
Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtJointSource carrier classifier surface`

Proof. Unpack a carried row into its extension packet, polynomial packet, generator history, minimal-polynomial history, and package ledger. The field-extension component supplies the comparison of base and extension histories, while the polynomial component supplies comparison of coefficient and polynomial rows.

For classification, compare the extension projections using the FieldExt$^{\uparrow}$ classifier and compare the minimal-polynomial projections using the Poly$^{\uparrow}$ equality classifier. The remaining carrier fields are provenance fields, so they move by the hsame transports recorded in the joint package ledger. □

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface carrier classifier surface`

Theorem 166.3 (Separable-extension simple-root obligation)**.** *The separability row for* SeparableExt$^{\uparrow}$ *is a displayed* Poly$^{\uparrow}$ *witness that the minimal-polynomial row has only simple roots. Concretely, the row records a derivative or gcd-style witness in* BHist*, continuation evidence in* Cont*, classifier transports in* hsame*, and package provenance in* Pkg*; it does not assert an uncarried host-algebra separability predicate.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSimpleRoot obligation`

Proof. Start from the minimal-polynomial packet supplied by Definition 166.1. Its Poly$^{\uparrow}$ ledger exposes the polynomial row, the derivative row or equivalent gcd-style comparison row, and the coefficient histories over the same extension packet.

The simple-root obligation is the package row that stores these displayed histories together with the Cont evidence computing the derivative or gcd comparison. Soundness is only a BEDC soundness statement: every accepted separability row is one of these displayed polynomial ledgers, transported by hsame and scoped by the shared Pkg provenance. □

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface simple root obligation`

Theorem 166.4 (Separable-extension classifier stability)**.** *The simple-root row is stable under the classifier of Definition 166.2. Transport along the inherited* FieldExt$^{\uparrow}$ *classifier and the inherited* Poly$^{\uparrow}$ *minimal-polynomial classifier carries the derivative or gcd-style witness, the* Cont *computation row, the* hsame *transports, and the* Pkg *provenance to the classified target carrier row.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface classifier stability`

Proof. Transport the field-extension part first. The FieldExt$^{\uparrow}$ classifier identifies the base, extension, and generator histories seen by the source row, so the minimal-polynomial packet remains over the transported extension packet.

Next transport the polynomial part. The Poly$^{\uparrow}$ equality classifier carries the minimal-polynomial row and its displayed derivative or gcd-style witness to the target packet. The Cont computation row is stable under the transported input and output histories, and the hsame witnesses compose with the package provenance to repack the same simple-root obligation at the target. □

Theorem 166.5 (Separable-extension ledger exactness)**.** *The* $\mathsf{SeparableExt}^{\uparrow}$ *ledger has coverage, soundness, and completeness for exactly the carried extension row, minimal-polynomial row, simple-root row, classifier-stability row, ledger row, and dependency rows supplied by* $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, NameCert, BHist, Cont, hsame, *and* Pkg*. No additional host-field algebra, root set, splitting-field data, or Galois action is part of this certificate surface.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface ledger exactness`
Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface semantic name certificate`
Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceRow semantic name certificate`

Proof. Coverage follows by reading the roadmap ten-field template. The traditional target is separability of a field extension; the BEDC sources are the $\mathsf{FieldExt}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ certificates; the carrier, classifier, stability, ledger, theorem-list, exactness, bridge, and status slots are the rows named in the four preceding theorems, with the bridge slot empty.

Soundness reads every accepted public row back to that list. A carried extension row comes from the field-extension dependency, a minimal-polynomial row comes from the polynomial dependency, and the simple-root row is the displayed derivative or gcd-style package with its Cont, hsame, and Pkg provenance.

Completeness is the reverse packaging. Given exactly those dependency rows and provenance rows, the ten-field template assembles a $\mathsf{SeparableExt}^{\uparrow}$ certificate row, and Definition 166.4 supplies the classifier transport needed to keep the row stable under certified comparison. □

Theorem 166.6 (Separable-extension mature example-family completeness)**.** *Any carried separable-extension example assembled from a* $\mathsf{FieldExt}^{\uparrow}$ *source packet, a* $\mathsf{Poly}^{\uparrow}$ *minimal-polynomial packet, shared* BHist *and* Pkg *provenance, and a simple-root derivative or gcd row is completely read by the five surfaces of this chapter: source dependency, carrier and classifier, simple-root obligation, classifier stability, and ledger exactness. The resulting family is mature at this certificate boundary because every accepted example is recovered from those five surfaces, and every such package reassembles the same* $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface mature example family`

Proof. First read the example through Definition 166.1. This identifies the two dependency packets and forces their base-field, extension-field, generator, and minimal-polynomial histories to share the same BHist and Pkg provenance.

Next apply Definition 166.2 and Definition 166.3. These two surfaces expose the carried extension row, the displayed minimal-polynomial row, and the derivative or gcd-style simple-root witness without adding a host-field root predicate.

Then use Definition 166.4 to transport the simple-root row across the inherited $\mathsf{FieldExt}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ classifiers. The derivative or gcd ledger, the continuation row, the hsame transports, and the package provenance remain inside the same carried surface.

Finally apply Definition 166.5. Its coverage, soundness, and completeness clauses say that the five surfaces enumerate exactly the public $\mathsf{SeparableExt}^{\uparrow}$ certificate rows. Root sets, splitting fields, and Galois actions are not exported by this family. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableExt}^{\uparrow}}$**.**

Constructive story: A separable extension is built by putting a $\mathsf{FieldExt}^{\uparrow}$ extension packet beside a $\mathsf{Poly}^{\uparrow}$ minimal-polynomial packet over the same displayed generator, then adding the simple-root row that is witnessed by a derivative or gcd-style finite history computation. The carrier accepts only this joint source surface with shared BHist and Pkg provenance, excluding uncarried host-field algebra, external root sets, splitting-field data, and Galois actions. Two accepted

rows classify together by the inherited FieldExt classifier on the extension packet, the inherited Polynomial classifier on the minimal-polynomial row, and the recorded hsame transports for generator, derivative, simple-root, and provenance faces. The source specification is the FieldExt-plus-Polynomial dependency row, the pattern specification is the visible minimal-polynomial simple-root witness, and the ledger keeps the raw derivative or gcd computation aligned with its public separability face through Cont and package rows. The five naming-certificate obligations are supplied at this scoped surface because an accepted joint row gives an inhabitant, hsame supplies the three equivalence moves, and classifier stability transports the simple-root witness back into the same carrier. A history h lies in the signature GAP of a SeparableExt$^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer row and evidence ledger, and token introduction identifies the answer as p; after projection, this says that h is the shared extension-and-minimal-polynomial endpoint whose simple-root witness is exactly the displayed polynomial ledger. The chapter registers this construction as SeparableExt$^{\uparrow}$ without closing a public separability theorem, any FieldExt or Polynomial bridge, a Galois bridge, or a standard bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{SeparableExt}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature example-family scope is exactly the carried FieldExt$^{\uparrow}$ source packet, Poly$^{\uparrow}$ minimal-polynomial surface, simple-root derivative or gcd witness, classifier-stability rows, ledger exactness, shared BHist, Cont, hsame, Pkg, and NameCert provenance named in Theorems 166.1–166.6.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparableExt}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SeparableExtUp.SeparableExtSourceSurface\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not close a scoped separable-extension carrier, public separability theorem, FieldExtUp bridge, PolynomialUp bridge, standard bridge, GaloisExtUp bridge, or Lean verification target.

Upgrade path: No further paper theory-axis grade is recorded for this chapter.

Chapter 167

A Concrete Naming Certificate for $\mathsf{Transcendence}^{\uparrow}$

The transcendence interface $\mathsf{Transcendence}^{\uparrow}$ packages the supremum of cardinalities of algebraically independent subsets of a field extension. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161).

167.1 Carrier definition

Definition 167.1 (Transcendence BHist carrier)**.** *A* $\mathsf{Transcendence}^{\uparrow}$ *carrier packet is a finite* BHist *packet over the* $\mathsf{FieldExt}^{\uparrow}$ *source rows. Its displayed rows consist of a certified field extension packet, a finite family row* $S = (s_1, \ldots, s_n)$ *in the extension endpoint, a base-field coefficient ledger, a finite list of polynomial-relation test rows for the family,* hsame *transports for the field-extension and family rows,* Cont *readbacks for each test row, and* Pkg *provenance pointing back to the* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *source packet.*

Lean4 (definition) `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket`

Definition 167.2 (Transcendence classifier)**.** *Two* $\mathsf{Transcendence}^{\uparrow}$ *carrier packets are classified as the same when their* $\mathsf{FieldExt}^{\uparrow}$ *source packets are matched by the displayed* $\mathsf{FieldExt}^{\uparrow}$ *classifier, their finite family rows are paired coordinatewise by* hsame*, and their polynomial-relation test ledgers have the same finite readback through* Cont*. The classifier compares only the listed rows; it does not compare supremum cardinalities, quotient fields, chosen transcendence bases, or maximal independent subsets.*

Lean4 (definition) `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket classifier`
Lean4 (definition) `BEDC.Derived.TranscendenceUp.TranscendenceClassifier`
Lean4 (definition) `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacketClassifier`

167.2 The certificate

Theorem 167.3 (Transcendence algebraic-independence ledger)**.** *For a* $\mathsf{Transcendence}^{\uparrow}$ *carrier packet, the finite family row* S *is accepted as algebraically independent over the displayed* $\mathsf{FieldExt}^{\uparrow}$ *base exactly when every polynomial-relation test row in the packet reads through* Cont *as the zero relation only at the zero coefficient ledger. This certificate is a finite ledger statement over the displayed family and coefficient rows, not a statement about the supremum of all independent cardinalities.*

Lean4 ✓ `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket algebraic independence ledger`

Proof. Read the carrier of Definition 167.1. It supplies a finite family row, a finite coefficient ledger, and a finite list of polynomial-relation test rows, all sourced from the same $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ packet.

For each displayed test row, the Cont readback records the evaluated relation endpoint. Ledger exactness says that the relation endpoint is zero only when the coefficient ledger is the zero ledger, so each listed polynomial test witnesses the independence condition for the finite family.

The converse uses the same finite rows. If the packet records the zero-only outcome for every displayed polynomial-relation test, repacking those readbacks gives the algebraic-independence ledger for S. No step selects a basis, maximizes a family, forms a quotient field, or invokes a Zorn-style argument. □

Theorem 167.4 (Transcendence FieldExt source boundary)**.** *Every source read used by a* $\mathsf{Transcendence}^{\uparrow}$ *carrier packet factors through the displayed* $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ *rows: base-field endpoint, extension-field endpoint, inclusion row, field-operation rows, finite family membership rows, polynomial coefficient rows,* hsame *transports,* Cont *readbacks, and* Pkg *provenance. The boundary exports no ambient quotient-field construction, algebraic closure, transcendence basis choice, or maximality principle.*

Lean4 ✓ `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket fieldext source transport boundary`

Proof. Start from the source component of Definition 167.1. Each row is tagged by the $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ provenance, so the base endpoint, extension endpoint, inclusion row, and field-operation rows are inherited as finite displayed source data.

The family and polynomial-test rows consume only those inherited endpoints and the finite coefficient ledger. Their stability is read by the hsame transports named in the carrier, and their observations are returned by the listed Cont rows.

Constructor inspection of the source boundary leaves no slot for a quotient field, algebraic closure, chosen transcendence basis, or maximality witness. Thus downstream $\mathsf{Transcendence}^{\uparrow}$ consumers receive only the displayed $\mathsf{FieldExt}^{\uparrow}$ packet and the finite independence ledger. □

Lean4 ✓ `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket fieldext source boundary`

Theorem 167.5 (Transcendence NameCert obligation surface)**.** *The carrier of Definition 167.1 and the classifier of Definition 167.2 determine the* NameCert *obligation surface for* $\mathsf{Transcendence}^{\uparrow}$*: carrier membership, classifier reflexivity, classifier symmetry, classifier transitivity, and carrier stability under the classifier are all witnessed by the finite* $\mathsf{FieldExt}^{\uparrow}$ *source packet, coordinatewise* hsame *transports, polynomial-test ledger exactness,* Cont *readbacks, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket namecert obligation surface`

Proof. Carrier membership is the finite packet described in Definition 167.1. The family row, coefficient ledger, polynomial tests, transports, readbacks, and provenance are all explicit rows, so no extra host-level algebraic object is needed to admit the packet.

Reflexivity uses reflexive hsame rows for the displayed $\mathsf{FieldExt}^{\uparrow}$ source and each family coordinate, together with identical polynomial-test readbacks. Symmetry reverses the same coordinatewise transports. Transitivity composes the source and family transports and then reuses the finite ledger exactness rows for the polynomial tests.

For carrier stability, transport the $\mathsf{FieldExt}^{\uparrow}$ packet and finite family row through the classifier, then carry each polynomial-relation test row along the named hsame transport. The Cont

readbacks preserve the zero-only coefficient ledger, so the transported packet is again accepted by the transcendence carrier. The resulting surface is the displayed $\mathsf{NameCert}_{\mathsf{Transcendence}^{\uparrow}}$ obligation packet, with no appeal to supremum cardinalities, quotient fields, basis choice, or maximality. □

Closure status for $\mathsf{NameCert}_{\mathsf{Transcendence}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Transcendence}^{\uparrow}}$, publicClosed)

Scope closed: The chapter scopes the $\mathsf{FieldExt}^{\uparrow}$ source rows, finite family rows, polynomial-test ledgers, hsame transports, Cont readbacks, and Pkg provenance used by the displayed carrier packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Transcendence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TranscendenceUp.TranscendenceCarrierPacket\_namecert\_obligation\_surface`

Bridge status: none

Constructive story: The carrier is a finite BHist packet whose rows are assembled from the displayed $\mathsf{FieldExt}^{\uparrow}$ source, a finite family, polynomial-test ledgers, transport readbacks, and package provenance.

Not claimed: This chapter does not close quotient fields, transcendence basis choice, maximality principles, or Zorn-style closure.

Upgrade path: Bridged closure requires a standard bridge for the exported transcendence certificate surface over the displayed $\mathsf{FieldExt}^{\uparrow}$ source, finite-family, polynomial-test, transport, readback, and provenance rows.

Chapter 168

A Concrete Naming Certificate for $\mathsf{NumField}^{\uparrow}$

The number-field interface $\mathsf{NumField}^{\uparrow}$ packages a finite field extension of the rational-number certificate. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161) and $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$ (chapter 37).

168.1 Carrier definition

Definition 168.1 (Reflexive rational field-extension carrier for NumFieldUp)**.** *The concrete* $\mathsf{NumField}^{\uparrow}$ *carrier is the RatUp reflexive field-extension carrier of Definition 161.3, read as a number-field certificate surface. Its base carrier and ambient carrier are both* RatHistoryCarrier*, its base and ambient classifiers are both* RatHistoryClassifier*, and its field-operation table is the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *operation table recorded in the RatUp certificate chapter.*

The embedding row is the identity embedding

$$\iota_{\mathsf{Rat}} : \mathsf{Rat}^{\uparrow} \hookrightarrow \mathsf{Rat}^{\uparrow},$$

and the finite-degree row is the singleton unary basis row. Thus the displayed module basis has one vector, the coordinate readback is the RatUp endpoint itself, and the scalar-action row is RatUp multiplication. The carrier has no separate algebraic endpoint outside the BHist *rows already used by the RatUp and FieldExtUp certificates.*

Lean4 (definition) `BEDC.Derived.NumFieldUp.NumFieldReflexiveRationalCarrier`
Lean4 (definition) `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveCarrier`

Theorem 168.2 (RatUp reflexive NumField carrier and classifier)**.** *For the carrier of Definition 168.1, the* $\mathsf{NumField}^{\uparrow}$ *carrier is exactly* RatHistoryCarrier *and the* $\mathsf{NumField}^{\uparrow}$ *endpoint classifier is exactly* RatHistoryClassifier*. Every displayed number-field endpoint is therefore a RatUp history endpoint, and every endpoint comparison is a RatUp classifier comparison.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive carrier classifier`

Proof. Unfold Definition 168.1. The carrier and classifier fields are specified directly as RatHistoryCarrier and RatHistoryClassifier.

The RatUp semantic name certificate supplies the carried-history interpretation for these fields. No quotient, completion, polynomial carrier, or new endpoint classifier is introduced by the $\mathsf{NumField}^{\uparrow}$ reading, so each number-field endpoint is read through the same RatUp history row. □

Theorem 168.3 (RatUp reflexive FieldExt consumption for NumFieldUp)**.** *For the carrier of Definition 168.1, the finite-extension component consumed by* $\mathsf{NumField}^{\uparrow}$ *is the identity* $\mathsf{FieldExt}^{\uparrow}$ *row of Definition 161.3. Its base field and ambient field are both the concrete* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ *certificate, its embedding is* $\iota_{\mathsf{Rat}}(h) := h$*, and its scalar action is the RatUp multiplication row.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive fieldext consumption`
Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational fieldext consumption`

Proof. Read the finite-extension witness from Definition 161.24. That theorem exposes the public $\mathsf{FieldExt}^{\uparrow}$ interface whose carrier, classifier, base field, ambient field, identity embedding, self-vector-space rows, exactness, and ledger all live over the RatUp operation table.

The $\mathsf{NumField}^{\uparrow}$ source only consumes that interface. Because the consumed row is the identity extension, the base and ambient field certificates are the same RatUp field certificate, and the scalar-action row is the same multiplication row already present in $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$. □

Theorem 168.4 (RatUp reflexive NumField ledger exactness)**.** *For the carrier of Definition 168.1, the* $\mathsf{NumField}^{\uparrow}$ *exactness and ledger rows are inherited from the RatUp reflexive* $\mathsf{FieldExt}^{\uparrow}$ *public interface. In particular, every carrier display, classifier display, field-operation endpoint, identity-embedding endpoint, vector-space endpoint, scalar-action endpoint, and ledger provenance row is one of the rows listed by Definition 161.24.*

Proof. First apply Definition 168.3. It identifies the finite-extension component used by $\mathsf{NumField}^{\uparrow}$ with the public RatUp reflexive $\mathsf{FieldExt}^{\uparrow}$ interface.

Next read the exactness and ledger clauses from Definition 161.24. Its public surface includes exact endpoint classification and ledger provenance over the RatUp operation table. Since the carrier and classifier are unchanged by Definition 168.2, those clauses serve as the displayed $\mathsf{NumField}^{\uparrow}$ exactness and ledger rows without any additional transport. □

Theorem 168.5 (RatUp reflexive NumField NameCert obligation surface)**.** *The carrier of Definition 168.1 assembles the paper-side* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *obligation surface. The surface records the RatUp history carrier, the RatUp history classifier, the identity finite-extension witness, the inherited FieldExt stability rows, the inherited exact endpoint classification, and the inherited ledger provenance from the RatUp reflexive* $\mathsf{FieldExt}^{\uparrow}$ *public interface.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational semantic name certificate`

Proof. Use Definition 168.2 for the carrier and classifier slots. These slots are the RatUp history carrier and classifier, so the $\mathsf{NumField}^{\uparrow}$ source is rooted in BHist.

Use Definition 168.3 for the finite field-extension slot. It supplies the identity embedding, the shared base-and-ambient RatUp field certificates, and the self-vector-space row required by the number-field reading.

Use Definition 168.4 for the exactness and ledger slots. Together these three rows cover the carrier, classifier, finite-extension, stability, exactness, and ledger obligations named by the paper-side $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ surface. □

168.2 The certificate

Theorem 168.6 (NumFieldUp reflexive rational source scope)**.** *For the carrier of Definition 168.1, every source, carrier, classifier, stability, ledger, exactness, and absent standard bridge field of the paper-side* $\mathsf{NumField}^{\uparrow}$ *surface is fixed by the displayed RatUp and FieldExtUp rows. The source is*

the concrete RatUp $\mathsf{Field}^{\uparrow}$ *table, the carrier is* RatHistoryCarrier*, the classifier is* RatHistoryClassifier*, stability is the identity-embedding and self-vector-space stability of the reflexive FieldExtUp row, exactness is endpoint classification by* RatHistoryClassifier*, the ledger is the RatUp operation-table ledger, and no standard bridge to host number fields is claimed.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational source scope`

Proof. Unfold Definition 168.1. The carrier is just the reflexive field-extension carrier of Definition 161.3, so the base and ambient history predicates, classifiers, and embedding row are the RatUp rows already used by FieldExtUp.

Use Definition 161.18 to account for the field operations. It says that the base-field operations, ambient-field operations, vector addition, vector zero, vector negation, and scalar action are all read from the single RatUp $\mathsf{Field}^{\uparrow}$ operation table.

Next apply Definition 161.22 and Definition 161.23. These identify every displayed endpoint with the RatUp classifier class named by the row and record the ledger provenance inside the RatUp operation table.

Finally use Definition 161.24 together with the RatUp certificate chapter. The public FieldExtUp interface supplies the visible extension surface, and the RatUp certificate supplies the rational source rows. Since the NumFieldUp carrier only re-reads these rows and adds no bridge tuple, the standard bridge field remains absent. □

Theorem 168.7 (NumFieldUp reflexive rational finite-extension witness)**.** *For the carrier of Definition 168.1, the finite-extension witness data are the singleton basis data over the reflexive FieldExtUp vector-space row. The basis list is the one-row unary basis* $[\mathbf{1}_{\mathsf{Rat}}]$*, every carried ambient endpoint* m *has the coordinate row* $m \cdot \mathbf{1}_{\mathsf{Rat}}$*, and coordinate equality is classified by* RatHistoryClassifier*.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational finite extension witness`

Proof. Read the vector-space part of the reflexive FieldExtUp package from Definition 161.3. Its ambient vector carrier is RatHistoryCarrier, its scalar carrier is the same RatUp carrier, and its scalar action is RatUp multiplication.

The singleton basis row chooses the RatUp unit endpoint as the only basis vector. For a carried ambient endpoint m, use the scalar m and the basis vector $\mathbf{1}_{\mathsf{Rat}}$; the multiplication row in the RatUp $\mathsf{Field}^{\uparrow}$ table classifies the product with m.

Apply Definition 161.18 for coverage of the multiplication and scalar-action rows. Then apply Definition 161.22 to read the coordinate endpoint in the same RatHistoryClassifier class. Thus the degree witness is the displayed singleton basis over the reflexive FieldExtUp vector-space row. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive finite basis witness`

Theorem 168.8 (RatUp reflexive NumField coordinate determinacy)**.** *For* $m, coord : \mathsf{BHist}$*, if* $\mathsf{RatHistoryCarrier}(m)$ *and* $\mathsf{Cont}(m, \mathsf{e}, coord)$*, then the coordinate readback is determined inside the RatUp classifier:* $\mathsf{RatHistoryClassifier}(coord, m)$*. The same row also forces* RatHistoryCarrier(FieldExtSingletonEmbed *and* $\mathsf{RatHistoryClassifier}(\mathsf{FieldExtSingletonEmbedding}(m), m)$*.*

Proof. Apply Definition 168.7. The witness identifies the displayed coordinate row for a carried endpoint m with $m \cdot \mathbf{1}_{\mathsf{Rat}}$ over the singleton basis.

The right-unit continuation $\mathsf{Cont}(m, \mathsf{e}, coord)$ reads that product as the RatUp endpoint named by m. Hence the coordinate endpoint remains in the RatHistoryClassifier class of m.

For the embedded endpoint, use the identity embedding row in Definition 168.1. It does not change the BHist endpoint, so the RatUp carrier predicate and the same classifier row apply to FieldExtSingletonEmbedding(m). □

Theorem 168.9 (RatUp reflexive NumField basis row exhaustion)**.** *Every accepted basis display in Definition 168.1 is the singleton rational unit row* $[\mathbf{1}_{\mathsf{Rat}}]$*. Every coordinate readback is classified by Definition 168.8. Thus the RatUp reflexive* $\mathsf{NumField}^{\uparrow}$ *carrier admits no second basis vector and no classifier row beyond* RatHistoryClassifier*.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveSingletonBasisRow support exactness`

Proof. First use Definition 168.6. It fixes the $\mathsf{NumField}^{\uparrow}$ source to the RatUp and FieldExtUp rows and leaves the finite basis field as the displayed singleton unary basis.

Next apply Definition 168.8. For each carried coordinate readback, the right-unit row and the identity embedding keep the coordinate in the RatUp classifier class of the source endpoint.

Finally apply Definition 168.25. Its row exhaustion says that the FieldExt rows are already exhausted by the RatUp reflexive certificate and that the NumField contribution is only the singleton basis row. Therefore neither a second basis vector nor an additional classifier row is accepted. □

Theorem 168.10 (RatUp reflexive NumField degree-one public package)**.** *The RatUp reflexive carrier of Definition 168.1, the source scope of Definition 168.6, the finite-extension witness of Definition 168.7, the basis-row exhaustion of Definition 168.9, and the ledger exactness of Definition 168.25 form the public degree-one* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *package. The package exposes carrier* RatHistoryCarrier*, classifier* RatHistoryClassifier*, the identity embedding, the singleton basis, RatUp scalar action, visible ledger exactness, and no standard bridge.*

Proof. Start with the source certificate Definition 37.90. It supplies the concrete RatUp carrier, classifier, operation table, exact endpoint classification, and ledger rows consumed by the number-field surface.

Next apply Definition 161.24. This repacks the same RatUp source as the reflexive FieldExtUp public interface, including the identity embedding and self-vector-space scalar action.

Use Definition 168.7 and Definition 168.9 for the degree-one part. These results identify the only basis display with the singleton rational unit row and classify every coordinate readback by the RatUp classifier.

Finally combine Definition 168.6 with Definition 168.25. The former fixes the visible $\mathsf{NumField}^{\uparrow}$ fields and the absent standard bridge, while the latter shows that the ledger contains no row beyond the RatUp reflexive FieldExt table and the singleton basis row. Repacking these fields gives the stated public degree-one $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ package. □

Definition 168.11 (NumFieldUp RatUp reflexive singleton basis row)**.** *For the carrier of Definition 168.1, the singleton basis row is the FieldExtUp vector-space singleton row. Its support is the one-element FieldExt vector carrier entry named by* FieldSingletonOne := BHist.Empty*.*

The RatUp data in the row live only in coordinate readback. For each carried m : BHist*, a coordinate endpoint coord* : BHist *is accepted when* Cont m BHist.Empty *coord holds, and the embedded ambient endpoint is* FieldExtSingletonEmbedding m*. Thus the basis support is a FieldExt singleton-vector support, while the coordinate endpoint and the embedded endpoint are the RatUp-classified rows attached to* m*.*

Lean4 (definition) `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveSingletonBasisRow`

Theorem 168.12 (NumFieldUp RatUp reflexive basis carrier boundary)**.** *In the singleton basis row of Definition 168.11, the basis support is carried by the FieldExtUp singleton vector carrier. The paper does not assert* RatHistoryCarrier BHist.Empty *for that support. RatUp carrier hypotheses apply to coordinate endpoints and to embedded ambient endpoints, not to the bare FieldExt singleton vector marker.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive basis support boundary`

Proof. Unfold the singleton basis row of Definition 168.11. The displayed support entry is FieldSingletonOne := BHist.Empty, so its carrier obligation is the FieldExt singleton vector carrier obligation.

The RatUp carrier predicate is read on the coordinate side. For a carried m : BHist, the accepted coordinate row has Cont m BHist.Empty *coord*, and the embedded endpoint is FieldExtSingletonEmbedding m. Apply RatHistoryCarrier_not_empty to separate this coordinate-side RatUp carrier from any assertion that BHist.Empty itself is a RatUp history carrier endpoint. □

Theorem 168.13 (NumFieldUp RatUp reflexive coordinate exactness span)**.** *Let* h, k, c, d : BHist. *If* RatHistoryClassifier h k*,* Cont h BHist.Empty c*, and* Cont k BHist.Empty d*, then* RatHistoryClassifier c d *and* RatHistoryClassifier (FieldExtSingletonEmbedding h)(FieldExtSingletonEmbedding k)*.*

Proof. Apply cont_right_unit_result to the two coordinate readbacks. This identifies c with h and d with k in the history classifier class named by right multiplication with the empty endpoint.

Next use RatHistoryClassifier_hsame_transport to transport the given classifier RatHistoryClassifier h k across these two right-unit identifications. This yields RatHistoryClassifier c d.

For the embedded endpoints, apply FieldExtRatReflexive_exact_endpoint_classification to the same classifier input. The reflexive FieldExt embedding is the singleton embedding row, so the embedded histories FieldExtSingletonEmbedding h and FieldExtSingletonEmbedding k remain in the same RatUp classifier class. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive coordinate exactness span`

Theorem 168.14 (NumFieldUp reflexive rational coordinate readback uniqueness)**.** *For* m, c_0, c_1 : BHist*, assume* RatHistoryCarrier(m)*,* Cont(m, e, c_0)*, and* Cont(m, e, c_1)*. Then* RatHistoryClassifier(c_0, c_1)*, and both* RatHistoryClassifier(c_0, m) *and* RatHistoryClassifier(c_1, m) *hold. Thus the singleton coordinate row attached to a carried RatUp endpoint is determined by the endpoint itself.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational coordinate readback pair determinacy`
Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational coordinate embedding pair readback`

Proof. Apply the continuation right-unit law Definition 18.18 to m. The supplied continuations Cont(m, e, c_0) and Cont(m, e, c_1) are therefore right-unit readbacks of the same history, so continuation determinacy identifies c_0 with m and c_1 with m up to hsame.

Transport the RatUp carrier row along these two hsame identifications. The RatUp semantic name certificate reads endpoint sameness through RatHistoryClassifier, so the transported rows give RatHistoryClassifier(c_0, m) and RatHistoryClassifier(c_1, m).

Finally compose the two classifier identifications through m. Transitivity of the RatUp classifier gives RatHistoryClassifier(c_0, c_1), which is the required uniqueness of degree-one coordinate readback. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational coordinate readback uniqueness`

Theorem 168.15 (NumFieldUp reflexive rational embedding-coordinate compatibility)**.** *For* $m, c_0 :$ BHist*, assume* $\mathsf{RatHistoryCarrier}(m)$ *and* $\mathsf{Cont}(m, \mathsf{e}, c_0)$*. Then* $\mathsf{RatHistoryClassifier}(\mathsf{FieldExtSingletonEmbedding}(m), c$ *Equivalently, the identity embedding row consumed from the reflexive FieldExtUp package is classified with the singleton coordinate row used by the degree-one NumFieldUp package.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational embedding coordinate classifier`

Proof. Use Definition 168.7 for the singleton finite-extension surface. It identifies the basis as the one-vector RatUp basis and reads the coordinate of a carried endpoint through the same $\mathsf{RatHistoryClassifier}$ row.

Unfold the displayed identity embedding from Definition 168.1. The embedding endpoint $\mathsf{FieldExtSingletonEmbedding}(m)$ is the same RatUp history endpoint m read through the FieldExtUp identity embedding row.

Apply Definition 168.14 to the carried endpoint m, the supplied coordinate c_0, and the right-unit coordinate m. The resulting classifier comparison identifies c_0 with m, and the identity-embedding display identifies $\mathsf{FieldExtSingletonEmbedding}(m)$ with the same classifier class. □

Theorem 168.16 (NumFieldUp reflexive rational degree-one row exhaustion)**.** *Every displayed finite-basis, coordinate, embedding, scalar-action, operation, exactness, and ledger row admitted by Definition 168.1 is either one of the exhausted RatUp reflexive FieldExtUp rows or the singleton degree-one coordinate row. The singleton row is classified by Definition 168.14 and is compatible with the identity embedding by Definition 168.15.*

Proof. Unfold Definition 168.1. The carrier contains the RatUp reflexive FieldExtUp source pattern together with one finite-basis field: the singleton basis vector, its coordinate readback, and the coordinate classifier row.

For the FieldExt part, apply Definition 161.27. This exhausts the base-field row, ambient-field row, identity embedding row, self-vector-space row, scalar-action row, operation-table row, exactness row, and ledger provenance row by the concrete RatUp $\mathsf{Field}^{\uparrow}$ table.

It remains to account for the basis field. The basis list has one displayed entry, the RatUp unit vector, and every coordinate readback is a right-unit continuation from the carried endpoint. The coordinate uniqueness theorem classifies any two such readbacks in the same $\mathsf{RatHistoryClassifier}$ class.

The embedding-coordinate compatibility theorem then identifies the singleton coordinate row with the identity embedding row. Hence the extra NumFieldUp surface adds exactly one degree-one coordinate row and no further finite-basis, embedding, scalar-action, exactness, or ledger row. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational degree one row exhaustion`

Definition 168.17 (NumFieldUp reflexive singleton basis coordinate surface)**.** *The reflexive singleton-basis coordinate surface for* $\mathsf{NumField}^{\uparrow}$ *fixes the concrete BEDC source as RatUp together with the reflexive* $\mathsf{FieldExt}^{\uparrow}$ *row of Definition 161.3. Its carrier is* $\mathsf{RatHistoryCarrier}$*, its classifier is* $\mathsf{RatHistoryClassifier}$*, and its finite-basis row is the one-element row* $\mathbf{1}_{\mathsf{Rat}}$*.*

For each carried ambient endpoint $m : \mathsf{BHist}$*, the coordinate readback is* m *itself and the reconstruction endpoint is the scalar-action product* $m \cdot \mathbf{1}_{\mathsf{Rat}}$*. The ledger is inherited from the RatUp operation table and the reflexive FieldExtUp row, and the surface is a paper-proof target without a standard bridge to an external number-field representation.*

Theorem 168.18 (NumFieldUp reflexive singleton basis exhaustion)**.** *For the coordinate surface of Definition 168.17, every accepted finite-basis row in the carrier is either the unique unit-basis*

row $\mathbf{1}_{\mathsf{Rat}}$ *or a coordinate row over* RatHistoryClassifier*. In particular, no accepted basis, coordinate, reconstruction, exactness, or ledger row has a coordinate source outside the RatUp history carrier and classifier.*

Proof. Unfold Definition 168.17. The surface only exposes the RatUp source, the reflexive FieldExtUp row, and the one-element basis field $\mathbf{1}_{\mathsf{Rat}}$.

Apply Definition 168.7. It identifies the finite-extension witness with the displayed singleton basis and reads each coordinate as the carried ambient endpoint m over RatHistoryClassifier.

For the rows inherited from FieldExtUp, apply Definition 161.27. This exhausts the carrier, classifier, embedding, scalar-action, operation, exactness, and ledger rows by the RatUp reflexive FieldExtUp certificate.

Finally use the RatUp semantic certificate Definition 37.90. It makes every accepted rational endpoint and endpoint comparison a RatUp history-classifier row. Thus the only remaining finite-basis row is the unit singleton, and every coordinate readback is classified by RatHistoryClassifier. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveSingletonBasisRow coordinate transport`

Theorem 168.19 (NumFieldUp reflexive embedding coordinate compatibility)**.** *For the coordinate surface of Definition 168.17, each carried ambient endpoint* m : BHist *has both its identity embedding coordinate* m *and its reconstructed endpoint* $m \cdot \mathbf{1}_{\mathsf{Rat}}$ *classified by* RatHistoryClassifier*. The embedding coordinate and the reconstruction endpoint therefore live on the same RatUp classifier surface.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive embedding coordinate product compatibility`

Proof. Read the embedding and vector-space rows from the reflexive FieldExtUp public interface in Definition 161.24. In the reflexive row, the embedding is the identity on RatHistoryCarrier and the scalar action is RatUp multiplication.

Use Definition 161.18. The operation-table coverage includes the multiplication row that interprets the scalar-action product $m \cdot \mathbf{1}_{\mathsf{Rat}}$ inside the RatUp operation ledger.

The right-unit row of the RatUp $\mathsf{Field}^{\uparrow}$ certificate identifies the product with the original endpoint m. Since Definition 37.90 supplies the RatUp history classifier for the endpoint m, the same classifier also classifies the reconstructed product endpoint. □

Theorem 168.20 (NumFieldUp reflexive degree-one basis exhaustion)**.** *For the carrier of Definition 168.1, the complete finite basis surface is the singleton unary row. Its only displayed basis endpoint is the RatUp unit endpoint* $\mathbf{1}_{\mathsf{Rat}}$*, every basis comparison is classified by* RatHistoryClassifier*, and no polynomial-root row, primitive-element row, or host-field row belongs to the basis surface.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational degree one coordinate transport`

Proof. Unfold Definition 168.1. The finite basis field is the basis component of the checked singleton witness in Definition 168.7; it has one displayed endpoint and uses the same RatUp classifier as the ambient endpoint carrier.

The operation coverage is inherited from Definition 161.18. It accounts for the RatUp multiplication row used by the scalar action and leaves no separate operation table for a root, primitive element, or host field.

Finally apply Definition 161.27. The FieldExt rows are exhausted by the RatUp base and ambient certificates, the identity embedding, and the self-vector-space rows. Projecting the remaining basis component gives exactly the singleton unit row and the RatHistoryClassifier comparison row. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive degree one basis endpoint`
Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive degree one basis exhaustion`

Theorem 168.21 (NumFieldUp reflexive coordinate readback determinacy)**.** *Let $m : \mathsf{BHist}$ be a carried ambient endpoint in the carrier of Definition 168.1. If $c, c' : \mathsf{BHist}$ are displayed coordinate histories whose coordinate rows read back through $\mathsf{Cont}(m, \mathsf{BHist.Empty}, c)$ and $\mathsf{Cont}(m, \mathsf{BHist.Empty}, c')$, then c and c' are both $\mathsf{RatHistoryClassifier}$-equivalent to m. Consequently c and c' are $\mathsf{RatHistoryClassifier}$-equivalent to each other.*

Proof. Use the singleton finite-extension witness of Definition 168.7 on the first displayed coordinate row. The witness reads the coordinate as the scalar m against the unique basis vector $\mathbf{1}_{\mathsf{Rat}}$, so RatUp multiplication classifies the endpoint c with m.

Apply the same witness to the second displayed coordinate row. The same singleton basis and scalar-action row classify c' with m through $\mathsf{RatHistoryClassifier}$.

The RatUp classifier laws are the semantic certificate laws supplied by Definition 37.90. Symmetry turns the first classification into the needed direction when necessary, and transitivity combines the two classifications to obtain the displayed equivalence between c and c'. □

Theorem 168.22 (NumFieldUp reflexive finite-extension exactness package)**.** *The concrete finite-extension part consumed by $\mathsf{NumField}^{\uparrow}$ in Definition 168.1 is exhausted by the identity embedding, the singleton basis of Definition 168.20, the coordinate-readback determinacy row of Definition 168.21, the RatUp scalar-action row, and the RatUp/FieldExt ledger rows. No additional finite-extension clause is available to downstream $\mathsf{NumField}^{\uparrow}$ consumers.*

Proof. Start with the concrete carrier consumption statement Definition 168.3. It identifies the underlying FieldExt surface as the RatUp reflexive extension with identity embedding and self-vector-space scalar action.

The source side is fixed by Definition 168.6. It says that the rational carrier, classifier, stability, exactness, and bridge status are read from the RatUp and FieldExt rows already named there.

For the finite basis field, apply Definition 168.20. For coordinate readback, apply Definition 168.21. These two rows isolate the singleton basis endpoint and the deterministic coordinate classifier used by every carried ambient endpoint.

Finally combine those rows with Definition 168.25. The ledger theorem exhausts the FieldExt rows and the singleton basis row, so repacking with the identity embedding and RatUp scalar action gives the exact finite-extension package consumed by $\mathsf{NumField}^{\uparrow}$. □

Theorem 168.23 (NumFieldUp reflexive rational NameCert obligation package)**.** *The carrier of Definition 168.1, the source scope of Definition 168.6, and the finite-extension witness of Definition 168.7, together with the coordinate uniqueness, embedding-coordinate compatibility, and degree-one row exhaustion of Definition 168.14, Definition 168.15, and Definition 168.16, assemble the paper-side $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ obligation package. The package consists of the RatUp rational source, the reflexive FieldExtUp extension carrier, the identity embedding, the singleton finite basis, the RatUp classifier exactness row, the RatUp operation-table ledger, and an absent standard-bridge field.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational namecert obligation package`

Proof. Use Definition 168.1 for the public carrier field. It names the same BHist carrier and classifier as RatUp and imports the reflexive FieldExtUp embedding and vector-space rows.

Use Definition 168.6 for source, classifier, stability, exactness, ledger, and standard-bridge status. Its rows are precisely the RatUp and FieldExtUp rows cited there.

Use Definition 168.7 for the finite-extension field. The singleton basis supplies a concrete degree witness and the coordinate exactness remains RatUp classifier exactness.

Use Definition 168.14, Definition 168.15, and Definition 168.16 for the reusable degree-one surface. These clauses show that right-unit coordinate readback is unique, the identity embedding sits in the same RatUp classifier class as the singleton coordinate, and the displayed rows are exhausted by the FieldExt table plus that singleton coordinate row.

The public NameCert interface of Definition 161.24 packages the underlying extension surface, while the RatUp certificate chapter supplies the rational source certificate. Repacking those rows with the singleton basis gives the displayed paper-side $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ obligation package and introduces no parameter-transport schema. □

Theorem 168.24 (RatUp reflexive NumField classifier transport)**.** *For accepted package histories P, Q : BHist in the carrier of Definition 168.1, if the displayed package rows of P and Q are componentwise related by* hsame*, then the* $\mathsf{NumField}^{\uparrow}$ *classifier transports the carrier of P to the carrier of Q. The transported rows are exactly the RatUp carrier row, the RatUp FieldUp base row, the RatUp FieldUp ambient row, the identity embedding row, the singleton basis row, and the scalar-action row.*

Proof. Unfold Definition 168.1. The source pattern is the RatUp reflexive $\mathsf{FieldExt}^{\uparrow}$ pattern with a singleton basis field, so the carrier rows of P and Q have the same visible components.

Apply Definition 161.25. It fixes the base carrier, ambient carrier, vector carrier, classifiers, identity embedding, and scalar action to the concrete RatUp data already certified by the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ source.

The RatUp classifier component is read from Definition 37.90. Thus componentwise hsame transports the rational endpoint rows, and the identity embedding does not alter either endpoint.

It remains to transport the singleton basis row. The row has one displayed unit vector and one coordinate classifier, both over the same $\mathsf{RatHistoryCarrier}$ and $\mathsf{RatHistoryClassifier}$. Projection from the componentwise hsame hypothesis therefore gives the transported basis row, and repacking the transported FieldExt rows with this basis row gives the claimed $\mathsf{NumField}^{\uparrow}$ carrier transport. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveSingletonBasis scalar transport`
Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive classifier transport`

Theorem 168.25 (RatUp reflexive NumField ledger exactness)**.** *Every accepted* $\mathsf{NumField}^{\uparrow}$ *carrier row, embedding row, basis row, scalar-action row, operation row, exactness row, and ledger row in Definition 168.1 is one of the displayed rows of the RatUp reflexive* $\mathsf{FieldExt}^{\uparrow}$ *certificate, together with the singleton basis row. In particular, the FieldExt rows are exhausted by Definition 161.27, and the basis row adds only the unit vector over* $\mathsf{RatHistoryCarrier}$ *with coordinate classifier* $\mathsf{RatHistoryClassifier}$*.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive ledger exactness`

Proof. Begin with an accepted $\mathsf{NumField}^{\uparrow}$ row and unfold Definition 168.1. The definition admits only rows belonging to the RatUp reflexive $\mathsf{FieldExt}^{\uparrow}$ source pattern and the singleton basis field.

For carrier, classifier, embedding, scalar-action, operation, exactness, and ledger rows, apply Definition 161.27. It says that the FieldExt part is covered by the two reads of the $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ endpoint certificate, the identity embedding row, and the self-vector-space row whose scalar action is RatUp multiplication.

The remaining row is the singleton basis field. Its support has one basis entry, the rational unit history, and its coordinate comparison is the same RatUp classifier. The basis ledger therefore records no operation endpoint beyond the unit row already present in the RatUp FieldUp table and no classifier beyond RatHistoryClassifier.

Combining these projections gives row exactness for the complete $\mathsf{NumField}^{\uparrow}$ carrier: every accepted row is either one of the exhausted FieldExt rows or the displayed singleton basis row. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexiveSingletonBasis ledger exactness`
↪ variant `BEDC.Derived.NumFieldUp.NumFieldRatReflexive ledger exactness`

Theorem 168.26 (RatUp reflexive NumField scoped NameCert certificate)**.** *The RatUp reflexive* $\mathsf{NumField}^{\uparrow}$ *construction carries a scoped* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *certificate with the following ten fields.*

1. *The traditional target is the finite field extension of RatUp presented by the reflexive identity extension and the singleton rational basis.*
2. *The BEDC source is RatUp together with the reflexive* $\mathsf{FieldExt}^{\uparrow}$ *extension over* BHist*.*
3. *The public carrier is Definition 168.1.*
4. *The classifier is* RatHistoryClassifier *on the RatUp endpoint histories.*
5. *The classifier-stability field is supplied by Definition 168.24.*
6. *The exactness field is supplied by Definition 168.25.*
7. *The ledger field is the RatUp operation-table ledger together with the singleton basis row, as exhausted by Definition 168.25.*
8. *The core witness list is Definition 168.7 together with Definition 168.23.*
9. *The standard-bridge field is absent.*
10. *The formalization target is the* BEDC.Derived.NumFieldUp *RatUp reflexive certificate surface.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive scoped namecert certificate`

Proof. The carrier and classifier fields are projections of Definition 168.1. That carrier is already a package over BHist whose visible endpoint classifier is RatHistoryClassifier.

The stability field is Definition 168.24. The exactness and ledger fields are the two projections of Definition 168.25: one projection says that accepted rows are exactly the displayed RatUp reflexive FieldExt rows, and the other isolates the singleton basis row.

The finite-extension witness is Definition 168.7, and the assembled NameCert obligations are Definition 168.23. Repacking these named fields gives the scoped certificate. No additional bridge row is constructed, so the standard-bridge field remains absent. □

Theorem 168.27 (RatUp reflexive NumField public NameCert interface)**.** *The scoped certificate of Definition 168.26 exports the public* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *surface whose data are the RatUp carrier,* RatHistoryClassifier*, the identity embedding, the singleton finite basis, scalar action by RatUp multiplication, exact endpoint classification, and the RatUp operation-table ledger.*

Proof. Project the scoped certificate along its carrier and classifier fields. These are the RatUp history carrier and RatHistoryClassifier because Definition 168.1 is built from the RatUp reflexive FieldExt carrier.

Project the extension fields from Definition 161.24. They give the identity embedding, the self-vector-space row, and scalar action by RatUp multiplication.

Project the finite-basis field from Definition 168.7. Its basis has exactly the rational unit vector, and its coordinate classifier is the RatUp endpoint classifier.

Finally project exactness and ledger provenance from Definition 168.25. Repacking these projections yields the displayed public $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ interface. □

Theorem 168.28 (RatUp reflexive NumField standard-bridge boundary)**.** *The certificate of Definition 168.26 claims no host number-field bridge. Any standard bridge for this surface must separately interpret the RatUp identity field extension and singleton basis. The public BEDC surface established here remains entirely within RatUp,* $\mathsf{FieldExt}^{\uparrow}$*,* hsame*, and the displayed* BHist *rows.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldUp StdBridge`

Proof. Read the bridge field of the scoped certificate. It is explicitly absent, so the certificate exports only BEDC package data and no host interpretation.

The exported BEDC data are still concrete: the RatUp source certificate, the reflexive FieldExt carrier, identity embedding, singleton basis, classifier transport, exact endpoint classification, and operation-table ledger. These are precisely the fields cited in Definition 168.27.

Therefore any host number-field bridge has to add an interpretation of the RatUp identity extension and its singleton basis as a separate construction. The present certificate stops at the public BEDC interface and does not claim a standard bridge. □

Theorem 168.29 (NumFieldUp reflexive rational FieldExt scope)**.** *The NumFieldUp source over Definition 168.1 consumes exactly the RatUp reflexive* $\mathsf{FieldExt}^{\uparrow}$ *public interface. Its base field and ambient field are both* $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$*, its embedding is the identity row, its vector carrier is* RatHistoryCarrier*, its scalar action is RatUp multiplication, and it introduces no additional* BHist *endpoint or classifier beyond the RatUp reflexive source.*

Proof. Read the public extension surface from Definition 161.24. The interface lists the two $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$ endpoint certificates, the identity embedding, and the self-vector-space carrier over RatHistoryCarrier.

Apply Definition 161.18 to the vector and scalar rows. It identifies vector addition, vector zero, vector negation, and scalar action with rows of the RatUp operation table, so the scalar action is RatUp multiplication.

The carrier of Definition 168.1 only repacks this public $\mathsf{FieldExt}^{\uparrow}$ surface with the singleton basis field. Therefore the NumFieldUp source has no separate endpoint, classifier, or operation row outside the RatUp reflexive FieldExt scope. □

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldRatReflexive fieldext scope`

Theorem 168.30 (NumFieldUp reflexive rational singleton-basis exhaustion)**.** *Every finite-basis row admitted by the carrier of Definition 168.1 is the singleton rational unit basis row. Its only basis endpoint is the RatUp unit history, and every coordinate comparison is classified by* RatHistoryClassifier*.*

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational singleton basis transport`

Proof. Use Definition 168.7. The finite-extension witness has basis list $[\mathbf{1}_{\mathsf{Rat}}]$ and represents each carried ambient endpoint by the coordinate row $m \cdot \mathbf{1}_{\mathsf{Rat}}$.

Apply Definition 168.25. It exhausts every accepted NumField row by the RatUp reflexive FieldExt rows together with the displayed singleton basis row.

Since there is no second basis field in Definition 168.1, any admitted finite-basis row is that singleton row. The coordinate classifier is the RatUp classifier already named by the finite-extension witness, namely RatHistoryClassifier. □

Theorem 168.31 (NumFieldUp reflexive rational exactness package)**.** *The carrier, classifier, FieldExt scope, singleton basis, stability, endpoint exactness, and ledger rows of Definition 168.1 assemble a concrete paper-side exactness package for* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *over* BHist.

Lean4 ✓ `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational exactness package`

Proof. Start with the carrier and classifier fields from Definition 168.24. They place the RatUp reflexive package rows in BHist and transport accepted packages componentwise by hsame.

Use Definition 168.29 for the extension surface. It fixes the base and ambient fields to $\mathsf{Field}^{\uparrow}(\mathsf{Rat}^{\uparrow})$, the embedding to the identity row, the vector carrier to RatHistoryCarrier, and scalar action to RatUp multiplication.

Use Definition 168.30 for the finite-basis clause. It shows that the only basis row is the rational unit row and that coordinate comparisons are read by RatHistoryClassifier.

Finally combine Definition 168.25 with Definition 37.90. The former exhausts the ledger and exactness rows by the RatUp FieldExt table plus the singleton basis, while the latter supplies the RatUp endpoint certificate. These rows are exactly the carrier, classifier, stability, endpoint exactness, and ledger fields required for the paper-side $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ exactness package. □

Theorem 168.32 (NumField reflexive dependency lattice)**.** *Every accepted reflexive* $\mathsf{NumField}^{\uparrow}$ *carrier read consumes the* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier rows of chapter 37 and the identity-extension rows of chapter 161 before it exposes the singleton-basis number-field row. In particular, the reflexive number-field packet is not an isolated carrier: its public read factors through the displayed rational source, the RatUp field table, the* $\mathsf{FieldExt}^{\uparrow}$ *identity embedding, the inherited vector-space rows, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Start with the reflexive carrier rows displayed earlier in this chapter. The base field, ambient field, endpoint classifier, and scalar-action rows are all inherited from the rational certificate and the identity $\mathsf{FieldExt}^{\uparrow}$ packet.

The singleton-basis row is exposed only after that inherited lattice has been read. The $\mathsf{Rat}^{\uparrow}$ rows supply the carrier and classifier surface, while the $\mathsf{FieldExt}^{\uparrow}$ rows supply the identity embedding and vector-space package.

The local $\mathsf{NumField}^{\uparrow}$ naming row repackages those displayed dependencies through its own transport, continuation, package, and certificate rows. Since no independent number-field carrier is present before the rational and identity-extension rows are consumed, every public reflexive read factors through the two sibling certificates. □

Theorem 168.33 (NumFieldUp mature reflexive coverage)**.** *The degree-one RatUp reflexive* $\mathsf{NumField}^{\uparrow}$ *package exhausts the paper-side mature treatment currently claimed for this chapter. Its visible surface is exactly the carrier and classifier row, FieldExt consumption row, finite-extension witness, singleton basis, coordinate readback, ledger exactness rows, public interface, and exactness package for the reflexive rational package. Nontrivial number fields and algebraic-integer theory remain outside this claim.*

Proof. Read the carrier and classifier from Definition 168.2, and read the FieldExt consumption row from Definition 168.3.

The finite-extension witness and singleton basis are supplied by Definition 168.7 and Definition 168.30. Coordinate readback is then fixed by the coordinate exactness, readback, and degree-one exhaustion rows in the reflexive certificate file.

Finally apply the ledger exactness rows, the public-interface row, and Definition 168.31. These rows repack the displayed reflexive rational package without adding a nontrivial extension, integral basis, class group, or algebraic-integer field. □

Closure status for $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$.

Constructive story: The number-field surface closed here is the reflexive rational extension: start with a $\mathsf{Rat}^\uparrow$ history endpoint, use the same $\mathsf{Rat}^\uparrow$ carrier and classifier for both the base and ambient fields, and consume the identity $\mathsf{FieldExt}^\uparrow$ row whose embedding sends each rational endpoint to itself. A history is admitted exactly when it is already a RatUp carrier history in this degree-one extension; no polynomial root, second basis vector, algebraic-integer endpoint, or separate ambient carrier is accepted. The classifier is simply the RatUp history classifier, transported through the identity embedding and the right-unit coordinate readback, so two displayed number-field endpoints are the same precisely when the underlying rational histories are classified as the same rational. The source specification is the RatUp field-operation table together with the reflexive $\mathsf{FieldExt}^\uparrow$ interface, the pattern specification is identity embedding, singleton unary basis, scalar action by RatUp multiplication, coordinate readback, basis exhaustion, and exact endpoint classification, and the ledger keeps the carrier, operation-table, finite-extension, basis, coordinate, embedding, scalar-action, and provenance rows visible. The five name-certificate obligations are inherited from RatUp and FieldExtUp: an accepted rational endpoint inhabits the carrier, the RatUp classifier supplies reflexivity, symmetry, and transitivity, and identity transport keeps every accepted endpoint inside the same degree-one package. A history h lies in the signature gap of a $\mathsf{NumField}^\uparrow$ token p when the source-domain accepts h as a rational endpoint in this reflexive extension, the signature relation pairs it with an answer endpoint and the singleton-basis evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is the rational coordinate encoded by p and its embedding coordinate is the same RatUp endpoint. The chapter registers this degree-one reflexive construction under the name $\mathsf{NumField}^\uparrow$; it introduces no nontrivial number field, Galois structure, algebraic-integer theory, public-object closure, or standard bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 168.28 is the paper-side bridge boundary for the RatUp reflexive NumFieldUp interface over FieldExtUp: BHist, RatHistoryCarrier, RatHistoryClassifier, the reflexive degree-one carrier, finite-extension witness, coordinate readback, singleton-basis exhaustion, ledger exactness, the NameCert obligation package, and the exactness package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.NumFieldUp.NumFieldReflexiveRational\_exactness\_package`

Bridge status: bridgeChecked

Not claimed: Nontrivial number fields, algebraic integer theory, Galois structure, public-object closure, and standard bridges are not claimed here.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 169

A Concrete Naming Certificate for $\mathsf{RingOfIntegers}^{\uparrow}$

The ring-of-integers interface $\mathsf{RingOfIntegers}^{\uparrow}$ packages the integral closure of $\mathsf{Int}^{\uparrow}$ inside a number-field carrier. It depends on $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168) and $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$ (chapter 31).

169.1 Carrier definition

Definition 169.1 (RingOfIntegers Dedekind-source carrier)**.** *A* $\mathsf{RingOfIntegers}^{\uparrow}$ *Dedekind-source carrier is a finite displayed* BHist *packet*

$$\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho)$$

over $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$*. The row* $\mathcal{N}$ *is the accepted number-field carrier row, and* $\mathcal{Z}$ *is the accepted embedded integer row read through* $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$ *inside the same field-operation boundary. The row* η *records the displayed embedding provenance from the integer source into the number-field source.*

The finite ledger E *lists integral-equation rows over the visible field addition, multiplication, zero, one, and negation operations supplied by* $\mathcal{N}$*, with coefficients read from* $\mathcal{Z}$*. The classifier row* κ *records the* hsame *transports for the number-field carrier, embedded integer rows, field-operation endpoints, and every equation row in* E*. The row* τ *is the finite* Cont *provenance tying the equation ledger to the two dependency certificates, and* ρ *is the shared* Pkg *provenance. No host quotient of algebraic integers, host polynomial equality, or ambient choice of a number-field model is a field of the carrier.*

Lean4 (definition) `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier`

169.2 The certificate

Theorem 169.2 (RingOfIntegers integral-closure obligation surface)**.** *The* $\mathsf{RingOfIntegers}^{\uparrow}$ *root obligation surface consists of the Dedekind-source carrier of Definition 169.1, its classifier* κ*, the finite integral-equation ledger* E*, the transport rows for* $\mathcal{N}$*,* $\mathcal{Z}$*,* η*, and* E*, and the no-host-quotient ledger recording that integral closure is read only from displayed* BHist*,* hsame*,* Cont*, and* Pkg *data. These rows form the root* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *obligation surface over* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$*.*

Every accepted root row projects to the number-field carrier row, the embedded integer row, the integral-equation ledger, classifier transport, continuation provenance, and package provenance listed above. Conversely, any finite row family with exactly these fields repacks as a $\mathsf{RingOfIntegers}^{\uparrow}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier integral closure obligation surface`

Proof. Read the two dependency certificates first. The $\mathsf{NumField}^{\uparrow}$ certificate supplies the visible field carrier and operation endpoints used by $\mathcal{N}$, while the $\mathsf{Int}^{\uparrow}$ certificate supplies the embedded integer row and the coefficient rows used by $\mathcal{Z}$ and η.

Project the carrier of Definition 169.1. Its visible fields are precisely $\mathcal{N}$, $\mathcal{Z}$, η, the integral-equation ledger E, the classifier row κ, the continuation row τ, and the package row ρ. The no-host-quotient clause in the definition excludes any additional ambient quotient or host polynomial-equality witness.

It remains to check the equation ledger. Induct over the finite ledger E. The empty ledger contributes no integral equation and leaves only the dependency, classifier, continuation, and package rows. In the cons step, the head equation is projected to its displayed field-operation endpoints and integer-coefficient rows, while the induction hypothesis projects the tail ledger.

The same induction gives the reverse packaging. Given the listed rows, rebuild the empty ledger immediately, and in the cons case attach the displayed head equation to the carrier already rebuilt for the tail. The transport rows for $\mathcal{N}$, $\mathcal{Z}$, η, and E supply exactly the classifier stability required by κ.

Collecting these fields yields the stated $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ obligation surface. Since every field has been projected from, or repacked into, the displayed finite packet, the surface has no hidden host quotient or detached integral-closure carrier. □

Theorem 169.3 (RingOfIntegers dependency projection boundary)**.** *Every accepted* $\mathsf{RingOfIntegers}^{\uparrow}$ *Dedekind-source carrier*

$$\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho)$$

projects exactly to the $\mathsf{NumField}^{\uparrow}$ *carrier and operation rows carried by* $\mathcal{N}$*, the embedded* $\mathsf{Int}^{\uparrow}$ *rows carried by* $\mathcal{Z}$ *and* η*, the displayed* Cont *provenance* τ*, the* hsame *classifier row* κ*, and the* Pkg *provenance* ρ*. The dependency projection has no host polynomial-equality row, host integral-closure carrier, or quotient carrier.*

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier dependency projection boundary`

Proof. Unpack the carrier of Definition 169.1. Its dependency fields are precisely the number-field row $\mathcal{N}$, embedded integer row $\mathcal{Z}$, embedding provenance η, classifier κ, continuation row τ, and package row ρ.

Apply Definition 169.2. The obligation surface projects accepted rows to those displayed dependency rows and to the finite equation ledger E, with no hidden host carrier.

Finally drop the equation entries and retain only the rows that describe the dependency boundary. Because the carrier definition has no field for host polynomial equality, a quotient carrier, or a detached integral-closure object, the projection is exhausted by the listed $\mathsf{NumField}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, Cont, hsame, and Pkg rows. □

Theorem 169.4 (RingOfIntegers equation-ledger transport closure)**.** *For an accepted* $\mathsf{RingOfIntegers}^{\uparrow}$ *Dedekind-source carrier*

$$\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho),$$

transport the finite integral-equation ledger E *by the displayed classifier row* κ*. The transported ledger, together with the same* $\mathsf{NumField}^{\uparrow}$ *carrier row, embedded* $\mathsf{Int}^{\uparrow}$ *rows,* Cont *provenance, and* Pkg *provenance, repacks as the same accepted* $\mathsf{RingOfIntegers}^{\uparrow}$ *obligation surface. No host polynomial-equality row, quotient carrier, or detached integral-closure carrier is introduced by the transport.*

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier equation ledger transport closure`

Proof. Unpack Definition 169.1. The classifier row κ contains the hsame transports for the number-field carrier, embedded integer rows, field-operation endpoints, and every integral-equation row in E.

Induct over the finite ledger E, following the ledger argument in Definition 169.2. In the empty case, transport preserves only the dependency, classifier, continuation, and package rows, so the obligation surface repacks immediately.

In the cons case, the head equation is transported through the displayed field-operation endpoints and integer-coefficient rows recorded by κ. The induction hypothesis transports the tail ledger and repacks it over the same $\mathcal{N}$, $\mathcal{Z}$, η, τ, and ρ rows.

Attach the transported head equation to the transported tail and repack through the obligation surface. Since every transported equation row remains a row of the displayed finite ledger and the carrier has no host polynomial-equality or quotient field, the accepted surface is unchanged. □

Theorem 169.5 (RingOfIntegers root classifier transport)**.** *For a* $\mathsf{RingOfIntegers}^{\uparrow}$ *Dedekind-source carrier*

$$\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho),$$

the root classifier transports exactly the displayed packet fields $\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho$ *along* hsame*. A transported accepted row has the same number-field source row, embedded integer row, embedding-provenance row, finite integral-equation ledger, classifier row,* Cont *provenance row, and* Pkg *provenance row. No host integral-closure carrier, quotient object, or detached polynomial-equality row is transported.*

Proof. Project the carrier through Definition 169.1. Its named fields are exactly $\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho$, and the classifier row κ records the allowed hsame transports for these fields.

Apply the obligation surface of Definition 169.2. It says that accepted root rows project only to the number-field source, embedded integer source, embedding provenance, integral-equation ledger, classifier transport, continuation provenance, and package provenance.

Repack the transported projections with the same finite ledger and provenance rows. Since neither the carrier definition nor the obligation surface has a field for a host integral-closure carrier, quotient object, or detached polynomial-equality row, transport is exhausted by the displayed packet fields. □

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier classifier transport`

Theorem 169.6 (RingOfIntegers integral-equation ledger exhaustion)**.** *Every integral-equation read accepted by a* $\mathsf{RingOfIntegers}^{\uparrow}$ *root surface is one of the finite ledger rows in* E *of the Dedekind-source carrier* $\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho)$*. Conversely, every row of* E*, together with its* hsame *classifier entry in* κ*, its finite* Cont *provenance in* τ*, and the shared* Pkg *row* ρ*, repacks as an accepted integral-equation read of the same root surface.*

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersEquationLedgerCarrier finite ledger readback`

Proof. Induct on the finite ledger E. In the empty case, the carrier still exposes the dependency rows, classifier row, continuation row, and package row, but it exposes no integral-equation read.

In the cons case, project the head equation to its displayed field-operation endpoints and integer-coefficient rows. The classifier component of κ and the continuation component of τ are the only rows that bind this head equation to the root packet.

Apply the induction hypothesis to the tail ledger. It exhausts all remaining integral-equation reads through the same $\mathcal{N}, \mathcal{Z}, \eta$ source rows and the same shared package provenance.

The reverse direction uses the same induction. The empty ledger repacks immediately, and the cons step attaches the displayed head equation before repacking the tail. Thus the accepted equation reads are exactly the finite rows of E, with no host integral-closure or quotient source. □

Theorem 169.7 (RingOfIntegers DirichletUnit source readback)**.** *A downstream* $\mathsf{DirichletUnit}^{\uparrow}$ *consumer reading from* $\mathsf{RingOfIntegers}^{\uparrow}$ *may read only the embedded integer row* $\mathcal{Z}$*, the integral-equation ledger* E*, the classifier row* κ*, the finite* Cont *provenance row* τ*, and the shared* Pkg *provenance row* ρ *carried by the accepted Dedekind-source packet* $\mathcal{O} = (\mathcal{N}, \mathcal{Z}, \eta, E, \kappa, \tau, \rho)$*. The consumer may also retain the number-field source row* $\mathcal{N}$ *and embedding provenance* η *as dependency scope, but it receives no host integral-closure carrier, quotient object, or analytic unit datum from* $\mathsf{RingOfIntegers}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier dirichletunit source readback`

Proof. Project the $\mathsf{RingOfIntegers}^{\uparrow}$ source through Definition 169.2. The dependency rows available to any consumer are the number-field row, embedded integer row, embedding provenance, integral-equation ledger, classifier, continuation provenance, and package provenance.

Use Definition 169.5 to identify the transported classifier data. A $\mathsf{DirichletUnit}^{\uparrow}$ reader may compare accepted integer and equation rows only through the displayed hsame entries of κ.

Use Definition 169.6 to exhaust the equation side. Every equation read is a member of E and is tied to the same finite continuation and package rows.

Therefore the downstream readback surface is exactly the embedded integer, equation-ledger, classifier, Cont, and Pkg rows, with $\mathcal{N}$ and η serving only as dependency scope. No additional host integral-closure, quotient, or analytic unit row is available to the consumer. □

Theorem 169.8 (RingOfIntegers scoped dependency package)**.** *The* $\mathsf{RingOfIntegers}^{\uparrow}$ *carrier, integral-equation ledger, dependency projection, classifier transport, ledger exhaustion, and* $\mathsf{DirichletUnit}^{\uparrow}$ *readback rows of this chapter are exhausted by the scoped primitive package consisting of* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$*.*

No public read of the chapter uses a host integral-closure predicate, quotient integer object, analytic unit datum, or extra dependency beyond those primitive rows and the displayed source certificates.

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier scoped dependency package`

Proof. First use Definition 169.2. It lists the carrier and integral-equation ledger rows available to the chapter and ties them to the number-field and integer source certificates.

Next apply Definition 169.3 and Definition 169.5. These identify the dependency projection and classifier transport as rows over BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$.

Finally use Definition 169.6 together with Definition 169.7. The ledger exhaustion and downstream readback clauses show that every consumer row is one of the displayed primitive rows, so the obligation surface is a scoped dependency package. □

Theorem 169.9 (RingOfIntegers scoped BHist source binding)**.** *The Dedekind-source carrier, dependency projection boundary, equation-ledger transport closure, root classifier transport, integral-equation ledger exhaustion, and* $\mathsf{DirichletUnit}^{\uparrow}$ *source readback bind the* $\mathsf{RingOfIntegers}^{\uparrow}$ *obligation surface entirely to* BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$, *and* Pkg *rows.*

No host polynomial equality, quotient object, detached integral-closure carrier, analytic unit datum, or standard bridge row is in scope for this binding.

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersClassifierTransportCarrier scoped bhist source binding`

Proof. First project the carrier and obligation rows through Definition 169.1 and Definition 169.2. These two blocks identify the accepted surface with the displayed number-field source, embedded integer source, integral-equation ledger, classifier, continuation, and package provenance.

Next bind the dependency and transport faces. Apply Definition 169.3, Definition 169.4, and Definition 169.5. They restrict all dependency projections, equation transport, and root classifier movement to the same BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$, and Pkg rows.

Finally use Definition 169.6 and Definition 169.7. The equation ledger is exhausted by the finite displayed rows, and the downstream $\mathsf{DirichletUnit}^{\uparrow}$ readback receives only the embedded integer, equation ledger, classifier, continuation, package, and dependency-scope rows already listed.

Repacking these projections gives the scoped source binding. Since none of the cited blocks exports a field for host polynomial equality, a quotient object, a detached integral-closure carrier, an analytic unit datum, or a standard bridge, those rows are outside the $\mathsf{RingOfIntegers}^{\uparrow}$ obligation surface. □

Lean4 ✓ `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier scoped bhist source binding`

Closure status for $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$**.**

Constructive story: A ring of integers is presented as a Dedekind-source packet over a $\mathsf{NumField}^{\uparrow}$ carrier and an embedded $\mathsf{Int}^{\uparrow}$ row, with the embedding provenance followed by a finite ledger of integral equations written in the visible field operations. The carrier accepts exactly the number-field row, embedded-integer row, embedding row, integral-equation ledger, classifier row, continuation provenance, and package provenance displayed in that packet, while host polynomial equality, quotient algebraic-integer objects, and detached integral-closure carriers have no slot. Its classifier transports the number-field carrier, embedded integer rows, operation endpoints, and every equation row by hsame, then repacks the same finite ledger with the same Cont and Pkg provenance. The source specification is the combined NumField and Int dependency surface, the pattern specification is the displayed integral-equation witness family, and the ledger keeps each raw equation row visible as one of the finite rows tied to the dependency certificates. The five naming-certificate obligations are represented by the displayed dependency package: an accepted packet provides a carrier endpoint, hsame gives reflexive, symmetric, and transitive comparison, and the equation-ledger transport theorem supplies closure under classified replacement. A history h lies in the signature GAP of a $\mathsf{RingOfIntegers}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it to an answer packet and evidence ledger, and token introduction names that answer as p; in this chapter, that means h is the endpoint whose number-field source, embedded integer source, integral-equation ledger, classifier, continuation row, and package provenance are exactly the carried rows. The chapter registers this construction as $\mathsf{RingOfIntegers}^{\uparrow}$ without closing full algebraic-integer theory, class-group structure, analytic units, or a standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature package consists of the displayed Dedekind-source carrier, integral-equation ledger, classifier transport, ledger exhaustion, and DirichletUnit readback rows over BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{Int}^{\uparrow}}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RingOfIntegersUp.RingOfIntegersDedekindSourceCarrier\_scoped\_bhist\_source\_binding`

Bridge status: none

Not claimed: It does not close full algebraic-integer theory, class-group structure, analytic units, or standard bridge rows.

Upgrade path: No further paper theory-axis strengthening is recorded for the displayed mature package; any standard bridge remains a separate bridge-axis matter.

Chapter 170

A Concrete Naming Certificate for $\mathsf{IdealClass}^{\uparrow}$

The ideal-class interface $\mathsf{IdealClass}^{\uparrow}$ packages the quotient group of fractional ideals modulo principal ideals in a Dedekind domain. It depends on $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ (chapter 172) and $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$ (chapter 82).

170.1 Carrier definition

Theorem 170.1 (IdealClass fractional-ideal carrier obligation)**.** *An* $\mathsf{IdealClass}^{\uparrow}$ *carrier endpoint is a* BHist *row carrying the* $\mathsf{Dedekind}^{\uparrow}$ *fractional-ideal source, the nonzero and invertible fractional-ideal ledger, and the* $\mathsf{QuotientGroup}^{\uparrow}$ *source boundary. The endpoint is admitted only when its* BHist*,* NameCert*,* hsame*, and* Pkg *rows identify the same Dedekind carrier and the same quotient-group boundary.*

Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassFractionalIdealCarrier obligation`
Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassFractionalCarrier source obligation`

Proof. Start from the opening dependencies on $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$. The carrier obligation reads the Dedekind fractional-ideal source as the BEDC source field, chooses BHist rows as the carrier, and records nonzero and invertible fractional-ideal evidence in the ledger before any quotient is formed.

The ten-field fit is: traditional target ideal classes of a Dedekind domain; BEDC source the cited $\mathsf{Dedekind}^{\uparrow}$ certificate together with the $\mathsf{QuotientGroup}^{\uparrow}$ source boundary; carrier BHist rows for fractional ideals; classifier pending principal-ideal equivalence; stability pending quotient transport; ledger the Dedekind source plus nonzero and invertible evidence; core theorem carrier endpoint admission; exactness exclusion of detached fractional-ideal representatives; standard bridge none; status ProofTarget. □

Theorem 170.2 (IdealClass principal classifier obligation)**.** *The* $\mathsf{IdealClass}^{\uparrow}$ *classifier relates two carried fractional-ideal endpoints exactly when their quotient product is principal. Reflexivity, symmetry, transitivity, and carrier transport are inherited from the* $\mathsf{Dedekind}^{\uparrow}$ *fractional-ideal rows and the* $\mathsf{QuotientGroup}^{\uparrow}$ *classifier boundary, so the classifier is a carried BEDC relation rather than a host quotient relation.*

Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassPrincipalClassifier semanticNameCert`

Proof. Use the fractional-ideal carrier fixed by Definition 170.1. The classifier row is the principal quotient-product witness relation over the same Dedekind source, so equality of endpoints is certified by a visible principal ideal and not by eliminating a quotient type.

The ten-field fit is: traditional target ideal-class equality; BEDC source $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ with the $\mathsf{QuotientGroup}^{\uparrow}$ source boundary; carrier carried fractional-ideal endpoints; classifier principal quotient-product equivalence; stability reflexivity, symmetry, transitivity, and carrier transport; ledger the two endpoints and the principal witness; core theorem classifier admission and equivalence laws; exactness absence of host quotient soundness; standard bridge none; status ProofTarget. □

Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassPrincipalClassifier empty boundary laws`

Theorem 170.3 (IdealClass quotient operation exactness obligation)**.** *Fractional-ideal multiplication descends through the principal-class classifier of Definition 170.2. The* $\mathsf{IdealClass}^{\uparrow}$ *ledger hides only the principal witness while preserving the* BHist*,* NameCert*,* hsame*, and* Pkg *provenance rows for the Dedekind source, the carried representatives, and the* $\mathsf{QuotientGroup}^{\uparrow}$ *operation boundary.*

Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassQuotientOperation exactness obligation`
Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassQuotientOperation empty product exactness`

Proof. First apply Definition 170.1 and Definition 170.2 to fix the carried representatives and the principal classifier. Multiplication is then read as a $\mathsf{QuotientGroup}^{\uparrow}$ operation row over that classifier, with identity and inverse data transported through the same source boundary.

The ten-field fit is: traditional target the ideal-class group operation; BEDC source the shared $\mathsf{Dedekind}^{\uparrow}$ fractional-ideal source and $\mathsf{NameCert}_{\mathsf{QuotientGroup}^{\uparrow}}$; carrier carried fractional-ideal endpoints; classifier principal quotient-product equivalence; stability multiplication descent, identity descent, inverse descent, and representative independence; ledger preserved BHist, NameCert, hsame, and Pkg provenance with only the principal witness hidden; core theorem quotient operation exactness; exactness no detached quotient and no lost provenance row; standard bridge none; status ProofTarget. □

170.2 The certificate

Theorem 170.4 (IdealClass scoped source certificate)**.** *The carrier obligation of Definition 170.1, the principal classifier obligation of Definition 170.2, and the quotient operation exactness obligation of Definition 170.3 jointly determine the scoped* $\mathsf{IdealClass}^{\uparrow}$ *source certificate over the cited* $\mathsf{Dedekind}^{\uparrow}$ *and* $\mathsf{QuotientGroup}^{\uparrow}$ *sources. Its carrier endpoints are* BHist *rows with Dedekind fractional-ideal provenance and quotient boundary provenance; its classifier equality is the principal quotient-product witness relation; its multiplication, identity, and inverse descent are read through the* $\mathsf{QuotientGroup}^{\uparrow}$ *operation boundary; and its ledger hides only the principal witness while preserving* BHist*,* NameCert*,* hsame*, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.IdealClassUp.IdealClassScoped source certificate`

Proof. Repack the carrier site as the carrier and ledger source of the certificate. The admitted endpoints are the BHist rows already tied to the $\mathsf{Dedekind}^{\uparrow}$ fractional-ideal source and the $\mathsf{QuotientGroup}^{\uparrow}$ boundary, with NameCert, hsame, and Pkg rows kept in the same source scope.

Repack the classifier site as the equality field. The relation is not a host quotient equality: it is the principal quotient-product witness relation on the carried endpoints, and its equivalence behavior is the one transported through the named Dedekind and quotient-group sources.

Repack the operation site as the stability and exactness fields. Multiplication, identity, and inverse descent are read through the $\mathsf{QuotientGroup}^{\uparrow}$ operation boundary, and the exactness ledger hides only the principal witness while retaining the visible BHist, NameCert, hsame, and Pkg provenance rows.

These three repackings fill the roadmap ten-field template for $\mathsf{IdealClass}^{\uparrow}$: traditional target, BEDC source, carrier, classifier, stability, ledger, core theorem surface, exactness exclusion, standard bridge status, and verification status. Thus the scoped source certificate is determined within exactly the $\mathsf{Dedekind}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, BHist, NameCert, hsame, and Pkg scope recorded above. □

Closure status for $\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$.

Constructive story: An ideal class is built from carried fractional-ideal endpoints over the displayed $\mathsf{Dedekind}^{\uparrow}$ source and the $\mathsf{QuotientGroup}^{\uparrow}$ boundary, with nonzero, invertible, and quotient-operation evidence kept as certificate rows before any quotient object is formed. The carrier accepts exactly those BHist rows whose Dedekind provenance and quotient-group provenance identify the same fractional-ideal boundary, while detached representatives and host quotient types are left outside the source. Two accepted endpoints classify together when the quotient product between them is witnessed as principal inside the same carried Dedekind and QuotientGroup rows, so equality of ideal classes is a visible principal-witness relation rather than elimination into a quotient set. The source specification is the paired Dedekind and QuotientGroup surface, the pattern specification is fractional-ideal multiplication with identity and inverse descent through principal witnesses, and the ledger hides only the principal witness while retaining the endpoint, classifier, hsame, and Pkg provenance rows. The five naming-certificate obligations are supplied by the scoped source certificate: an accepted fractional-ideal endpoint inhabits the carrier, the principal classifier inherits reflexivity, symmetry, and transitivity from the quotient boundary, and carrier transport follows from the same Dedekind and QuotientGroup source rows. A history h lies in the signature GAP of an $\mathsf{IdealClass}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer fractional-ideal row and evidence ledger, and token introduction names that answer as p; concretely, h must carry the Dedekind fractional-ideal provenance, the quotient boundary, and the principal-witness ledger that identifies its class. The chapter registers this construction as $\mathsf{IdealClass}^{\uparrow}$ without closing a public NameCert surface, a standard bridge, or a separate Lean theorem for the full ideal-class interface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridged paper surface is the $\mathsf{Dedekind}^{\uparrow}$ fractional-ideal endpoint, $\mathsf{QuotientGroup}^{\uparrow}$ principal-classifier boundary, quotient-operation descent, and visible BHist, NameCert, hsame, and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.IdealClassUp.IdealClassScoped\_source\_certificate`

Bridge status: none

Not claimed: This chapter does not close a standard bridge or a separate host quotient model for IdealClassUp.

Upgrade path: No paper-side theory-grade obligation remains before a separate standard bridge is supplied; bridge verification is not claimed here.

Chapter 171

A Concrete Naming Certificate for $\mathsf{DirichletUnit}^{\uparrow}$

The Dirichlet-unit interface $\mathsf{DirichletUnit}^{\uparrow}$ packages the structure of the unit group of a ring of integers as a finitely generated abelian group of rank r_1+r_2-1. It depends on $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ (chapter 169) and $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ (chapter 42).

171.1 Carrier definition

Definition 171.1 (DirichletUnit carrier packet)**.** *A* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packet is a displayed* BHist *tuple*

$$\mathcal{D} = (\mathcal{O}, \mathcal{A}, u, \iota, \nu, \beta, \rho)$$

whose entries are read as follows. The row $\mathcal{O}$ *is the dependency* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$*, the row* $\mathcal{A}$ *is the dependency* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$*, the row* u *is a visible unit row over* $\mathcal{O}$*, the row* ι *records the inverse and unit-product witnesses for* u*, the row* ν *is a unary free-rank witness, the row* β *is the finite basis and logarithmic ledger row, and the row* ρ *is the* Pkg *provenance tying these entries to the two dependency certificates. The packet also carries the finite* Cont *readbacks from* $\mathcal{O}$ *to* u *and* ι*, from* $\mathcal{A}$ *to the public abelian-group law rows, and from* ρ *to every exported dependency slot.*

The packet is a history-facing carrier. It exposes no host-level unit type, no analytic logarithm equality, no determinant, and no standard bridge; it exposes only the named rows, their readback ledgers, and their package provenance.

Lean4 (definition) `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier`

Theorem 171.2 (DirichletUnit visible unit row obligation)**.** *Every accepted* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packet exposes the visible unit row* u *and the inverse/unit-product row* ι *as carried* BHist *rows over the dependency certificate* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$*. These rows are exported only through the finite* Cont *readback and* Pkg *provenance of Definition 171.1; they are not host units and do not assert host-level multiplication outside the displayed history.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier visible unit rows`

Proof. Project an accepted packet to its tuple entries. The rows $\mathcal{O}$, u, and ι are named components of the same displayed BHist packet, and the carrier definition includes the Cont readback from $\mathcal{O}$ to the two unit-facing rows.

The provenance row ρ records that the readback is part of the same package. Repacking these projections gives exactly the exported unit and inverse surface, with no additional host carrier available to inspect. □

Theorem 171.3 (DirichletUnit classifier obligation)**.** *Classifier transport for* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packets is componentwise* hsame *together with dependency provenance transport. If packets* $\mathcal{D}$ *and* $\mathcal{D}'$ *are classified, then* hsame *transports the visible unit rows, inverse/unit-product rows, rank witness rows, finite basis/log-ledger rows, and package provenance rows, while the dependency slots remain tied to* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *by their displayed* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier classifier transport`

Proof. Unpack the classifier as comparisons between the corresponding displayed rows of $\mathcal{D}$ and $\mathcal{D}'$. Each comparison is an hsame witness at a named projection of the carrier packet.

Transport the unit and inverse rows through the $\mathsf{RingOfIntegers}^{\uparrow}$ readback, then transport the rank and basis/log-ledger rows through their own packet projections. The package row transports the dependency provenance at the same time.

Repack the transported rows with the target packet's readback ledgers. The result is componentwise classifier transport for the whole Dirichlet-unit surface. □

Theorem 171.4 (DirichletUnit free-rank witness ledger)**.** *The free-rank witness* ν *and the finite basis/logarithmic ledger row* β *of a* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packet are visible ledger rows. Their stability under accepted carrier comparison is carried by* Cont*,* hsame*, and* Pkg *provenance, and does not assert equality of analytic logarithms or equality of regulator values.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier free rank witness ledger`

Proof. Project the rank and basis/log-ledger entries from the carrier packet. They are listed as displayed BHist rows, so their visibility is part of the packet rather than a theorem about an external analytic object.

For a classified pair, apply componentwise hsame to the rank row and to the ledger row. The Cont entries preserve the finite readback pattern, and ρ supplies the package provenance tying those rows to the same dependency boundary.

The proof stops at the ledger boundary. Since no analytic logarithm map or determinant row is a component of the packet, no such equality can be projected from this obligation. □

Theorem 171.5 (DirichletUnit abelian-group dependency obligation)**.** *The public abelian-group laws used by* $\mathsf{DirichletUnit}^{\uparrow}$ *are inherited through the dependency certificate* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *and are restricted to unit rows accepted by the* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *dependency. Thus the group-law surface is available only where the ring-of-integers certificate supplies the visible unit and inverse rows of Definition 171.1.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier abgroup dependency obligation`

Proof. Project the dependency row $\mathcal{A}$ and its Cont readback from the carrier packet. This is the only source of the public abelian-group law rows.

Project the ring-of-integers row $\mathcal{O}$ together with u and ι. The law rows from $\mathcal{A}$ are paired with these unit-facing rows by the same package provenance row ρ.

Repack the dependency projections. The inherited law surface is therefore confined to accepted unit rows over $\mathcal{O}$, rather than a free-standing group law on an ambient host type. □

Theorem 171.6 (DirichletUnit NameCert surface obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{DirichletUnit}^\uparrow}$ *surface is assembled from the carrier packet of Definition 171.1, the componentwise classifier of Definition 171.3, the dependency certificates* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{AbGroup}^\uparrow}$*, the visible unit and inverse rows, the rank and finite basis/log-ledger rows, the* Cont *stability rows, the* Pkg *provenance rows, and the exactness claim that every exported Dirichlet-unit row is one of these displayed packet entries.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier namecert obligation surface`

Proof. Start with an accepted carrier packet. Its displayed entries give the carrier, dependency, unit, inverse, rank, ledger, and provenance rows.

Use the classifier obligation to supply transport for comparisons between accepted packets. Use the visible-unit obligation and the abelian-group dependency obligation to bind the unit-facing rows to $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^\uparrow}$ and the law-facing rows to $\mathsf{NameCert}_{\mathsf{AbGroup}^\uparrow}$.

The exactness statement is obtained by projection and repacking. Any exported certificate row must be one of the packet entries or one of the finite readbacks named in the packet, and those entries repack as the public $\mathsf{NameCert}_{\mathsf{DirichletUnit}^\uparrow}$ surface. □

Theorem 171.7 (DirichletUnit Regulator consumer boundary)**.** *A downstream* $\mathsf{Regulator}^\uparrow$ *certificate may consume from* $\mathsf{DirichletUnit}^\uparrow$ *only the visible unit rows, inverse/unit-product rows, unary rank witness, finite basis/logarithmic ledger rows, dependency rows,* Cont *readbacks, and* Pkg *provenance rows described above. It may not read a determinant, an analytic regulator value, or a standard bridge from* $\mathsf{DirichletUnit}^\uparrow$ *alone.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier regulator consumer boundary`

Proof. Project the public $\mathsf{NameCert}_{\mathsf{DirichletUnit}^\uparrow}$ surface from Definition 171.6. The listed consumer inputs are exactly the rows present in that surface.

Inspect the carrier packet: determinant rows, analytic regulator values, and standard bridge rows are not entries of $\mathcal{D}$, nor are they reachable by the finite Cont readbacks named by the packet.

Therefore a $\mathsf{Regulator}^\uparrow$ construction can cite the unit, inverse, rank, ledger, dependency, readback, and provenance rows. Any determinant or analytic regulator row must come from a separate regulator-side construction. □

Theorem 171.8 (DirichletUnit Regulator unit-row source exhaustion)**.** *Every* $\mathsf{Regulator}^\uparrow$ *consumer read of a* $\mathsf{DirichletUnit}^\uparrow$ *unit row or inverse row factors through the visible unit row* u*, the inverse/unit-product row* ι*, the ring-of-integers dependency row* $\mathcal{O}$*, the finite* Cont *readback, and the* Pkg *provenance of Definition 171.1. No other unit-facing source is exported by the carrier packet.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier regulator unit row source exhaustion`

Proof. Project the accepted packet through Definition 171.1. The only unit-facing entries are u and ι, and their source dependency is the displayed row $\mathcal{O}$.

Apply Definition 171.6 to pass from packet entries to public certificate rows. Then apply Definition 171.12 where the read asks for the inherited inverse or product law row.

The finite Cont readback and the provenance row ρ are the remaining packet entries tying these unit-facing rows to the exported surface. Repacking these projections gives the stated source exhaustion for every $\mathsf{Regulator}^\uparrow$ unit-row read. □

Theorem 171.9 (DirichletUnit Regulator rank-ledger source)**.** *Every* $\mathsf{Regulator}^{\uparrow}$ *read of a* $\mathsf{DirichletUnit}^{\uparrow}$ *rank row, basis row, or logarithmic-ledger row factors through the unary rank witness* ν *and the finite ledger row* β *of Definition 171.1. The factorization asserts no determinant, analytic logarithm equality, or regulator value.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier regulator rank ledger source`

Proof. Project the carrier packet. Its rank-facing component is the unary witness ν, and its basis or logarithm-facing component is the finite ledger row β.

Apply Definition 171.11. This identifies the rank row as a $\mathsf{UnaryHistory}$ ledger and keeps the visible expression inside finite readback.

Finally apply Definition 171.6 to read these two rows as public $\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$ entries. Since Definition 171.1 contains no determinant or analytic logarithm equality row, the consumer read cannot project either one from $\mathsf{DirichletUnit}^{\uparrow}$ alone. □

Theorem 171.10 (DirichletUnit Regulator consumer threshold package)**.** *The exact* $\mathsf{DirichletUnit}^{\uparrow}$ *threshold package available to* $\mathsf{Regulator}^{\uparrow}$ *consists of the unit-row source surface of Definition 171.8, the rank-ledger source surface of Definition 171.9, and the exclusion boundary of Definition 171.7. Thus a* $\mathsf{Regulator}^{\uparrow}$ *construction may consume visible unit and inverse rows, the unary rank witness, finite basis/logarithmic ledger rows, dependency rows,* Cont *readbacks, and* Pkg *provenance, but not determinant, analytic regulator, or standard-bridge rows from* $\mathsf{DirichletUnit}^{\uparrow}$ *alone.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier regulator consumer threshold package`

Proof. First project the unit-facing surface. Apply Definition 171.8; this gives the rows u, ι, $\mathcal{O}$, their finite readback, and their package provenance.

Next project the rank-facing surface. Apply Definition 171.9; this gives the unary witness ν and the finite ledger row β, without analytic determinant or logarithm equality.

Finally apply Definition 171.7. It excludes every regulator-facing row not listed by the two source projections. Repacking the surviving rows gives exactly the threshold package stated above. □

Theorem 171.11 (Dirichlet-unit rank ledger)**.** *Every accepted carrier of Definition 171.1 contains a rank witness as a* $\mathsf{UnaryHistory}$ *row recording the visible ledger expression* $r_1 + r_2 - 1$*. This row is stable under the row classifier of Definition 171.3, and it asserts no host arithmetic beyond finite ledger readback.*

Proof. Project the finite package underlying the carrier. The rank row is read as a unary history from the same source certificate that supplies the ring of integers and its real and complex place-count ledgers.

The visible expression $r_1 + r_2 - 1$ is therefore a ledger label: its content is the displayed finite readback row, not an assertion about an external arithmetic object. The $\mathsf{UnaryHistory}$ witness records exactly that visible row.

The classifier compares the source, unit, inverse, law, and provenance rows through finite Cont readback. Hence the same unary rank row remains attached to every classified carrier. □

Theorem 171.12 (Dirichlet-unit AbGroup projection)**.** *Every accepted* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier projects to the* $\mathsf{AbGroup}^{\uparrow}$ *operation, inverse, identity, and classifier rows carried by its dependency certificate. The Dirichlet-unit carrier adds the ring-of-integers unit source, the unit and inverse rows, and the finite readback ledgers, but it does not replace the inherited abelian-group interface.*

Proof. Unpack the tuple of Definition 171.1. Its $\mathcal{A}$ projection is the dependency row from $\mathsf{NameCert}_{\mathsf{AbGroup}^\uparrow}$.

Read $\mathcal{A}$ through the abelian-group certificate. This exposes the operation row, inverse row, identity row, and classifier row for the inherited finitely generated abelian carrier.

The remaining projections $\mathcal{O}$, u, u^{-1}, λ, μ, and ρ witness that the inherited row is being used as a unit group for the ring-of-integers source. Repacking with $\mathcal{A}$ therefore gives exactly the $\mathsf{AbGroup}^\uparrow$ projection fields, with the unit rows retained as additional dependency data. □

171.2 The certificate

Theorem 171.13 (DirichletUnit inherited unit-product closure)**.** *Let*

$$\mathcal{D}_0 = (\mathcal{O}, \mathcal{A}, u_0, \iota_0, \nu, \beta, \rho) \qquad \textit{and} \qquad \mathcal{D}_1 = (\mathcal{O}, \mathcal{A}, u_1, \iota_1, \nu, \beta, \rho)$$

be accepted $\mathsf{DirichletUnit}^\uparrow$ *carrier packets with the same* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^\uparrow}$ *dependency row* $\mathcal{O}$*, the same* $\mathsf{NameCert}_{\mathsf{AbGroup}^\uparrow}$ *dependency row* $\mathcal{A}$*, and the same* Pkg *provenance row* ρ*. Let* u_0 *and* u_1 *be their visible unit rows, and let* ι_0 *and* ι_1 *be the inverse/unit-product witness rows supplied by the carrier packets. If* $u_0 \star_{\mathcal{A}} u_1$ *denotes the endpoint of the inherited* $\mathsf{AbGroup}^\uparrow$ *operation row projected from* $\mathcal{A}$*, then* $u_0 \star_{\mathcal{A}} u_1$ *is again a visible* $\mathsf{DirichletUnit}^\uparrow$ *unit row over* $\mathcal{O}$*. Its inverse/unit-product witness is the displayed row obtained by applying the same inherited operation to the two inverse endpoints in* ι_0 *and* ι_1*, and its public provenance is exactly the shared* Cont *and* Pkg *provenance of* $\mathcal{D}_0$ *and* $\mathcal{D}_1$*.*

Proof. Project both accepted packets through Definition 171.1. The two projections expose the same dependency rows $\mathcal{O}$ and $\mathcal{A}$, the same structural rows ν and β, and the same provenance row ρ. By Definition 171.2, the rows u_0, u_1 and their witness rows ι_0, ι_1 are visible unit-facing history rows over $\mathcal{O}$, read only through finite Cont entries and the shared Pkg provenance.

Project the abelian-group operation from $\mathcal{A}$ using Definition 171.12. The restriction clause in Definition 171.5 says that this operation is available precisely on unit rows accepted by the $\mathsf{RingOfIntegers}^\uparrow$ dependency. Since u_0 and u_1 are accepted unit rows over the same $\mathcal{O}$, the inherited operation has a displayed endpoint $u_0 \star_{\mathcal{A}} u_1$ inside the same dependency boundary.

It remains to read this endpoint as a $\mathsf{DirichletUnit}^\uparrow$ unit row rather than as a detached abelian-group product. The witness rows ι_0 and ι_1 give inverse endpoints u_0^{-1} and u_1^{-1} together with the unit-product comparisons for u_0 and u_1. Apply the inherited operation row of $\mathcal{A}$ to these inverse endpoints. The $\mathsf{AbGroup}^\uparrow$ stability certificate contains multiplication congruence, associativity, inverse laws, and commutativity, and Definition 42.14 packages exactly the resulting comparison

$$(u_0 \star_{\mathcal{A}} u_1) \star_{\mathcal{A}} (u_0^{-1} \star_{\mathcal{A}} u_1^{-1})\mathsf{hsame}e_{\mathcal{A}}.$$

Thus the product endpoint carries its own inverse/unit-product witness row, built only from ι_0, ι_1, and the operation rows already projected from $\mathcal{A}$.

Repack the endpoint $u_0 \star_{\mathcal{A}} u_1$ with this inverse/unit-product witness, the unchanged dependency rows $\mathcal{O}$ and $\mathcal{A}$, the unchanged structural rows ν and β, and the same provenance row ρ. This is exactly the visible unit-row slot described in Definition 171.1: the ring-of-integers readback still points to $\mathcal{O}$, the group-law readback still points to $\mathcal{A}$, and the exported row is still tied to the same Cont and Pkg ledgers. No host-level multiplication, analytic logarithm equality, determinant row, or standard bridge is introduced. □

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier inherited unit product closure`

Theorem 171.14 (DirichletUnit root inverse-row inversion)**.** *Let $\mathcal{D} = (\mathcal{O}, \mathcal{A}, u, \iota, \nu, \beta, \rho)$ be an accepted* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packet. Any accepted inverse row read from $\mathcal{D}$ is uniquely projected to the visible unit row u, the* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *dependency row $\mathcal{O}$, the inherited* $\mathsf{AbGroup}^{\uparrow}$ *inverse row carried by $\mathcal{A}$, the finite* Cont *readback tying these rows together, and the* Pkg *provenance row ρ. No other inverse source is available at the root boundary.*

Lean4 ✓ `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier root inverse row inversion`

Proof. Project the accepted packet through Definition 171.1. The tuple exposes exactly one visible unit row u, exactly one inverse/unit-product witness row ι, and the dependency rows $\mathcal{O}$ and $\mathcal{A}$ to which they are attached.

Read $\mathcal{A}$ through Definition 171.12. The inherited inverse row is the only abelian-group inverse row exported to $\mathsf{DirichletUnit}^{\uparrow}$, and Definition 171.5 restricts it to unit rows accepted by $\mathcal{O}$.

The Cont readback and Pkg provenance are the remaining packet entries that bind the inverse witness to the visible unit row. Repacking these projections gives the stated inversion surface, and the carrier packet contains no second inverse row, host unit carrier, or analytic witness from which a competing root inverse source could be read. □

Theorem 171.15 (DirichletUnit root product-witness inversion)**.** *Let $\mathcal{D}_0 = (\mathcal{O}, \mathcal{A}, u_0, \iota_0, \nu, \beta, \rho)$ and $\mathcal{D}_1 = (\mathcal{O}, \mathcal{A}, u_1, \iota_1, \nu, \beta, \rho)$ be accepted* $\mathsf{DirichletUnit}^{\uparrow}$ *carrier packets over the same* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *dependency rows. A product witness in the inherited* $\mathsf{AbGroup}^{\uparrow}$ *row is a valid* $\mathsf{DirichletUnit}^{\uparrow}$ *product witness exactly at the packet boundary where both visible unit rows u_0, u_1, both inverse/unit-product witnesses ι_0, ι_1, the inherited operation row, the shared* Cont *readbacks, and the shared* Pkg *provenance are present over $\mathcal{O}$.*

Proof. For the forward reading, start from a product witness accepted as a $\mathsf{DirichletUnit}^{\uparrow}$ product row. Inverting the packet definition exposes the two visible unit rows and their inverse/unit-product witness rows, and Definition 171.14 identifies their inverse evidence with the inherited inverse row of $\mathcal{A}$ over the same $\mathcal{O}$.

Project the inherited operation row from $\mathcal{A}$ using Definition 171.12. Its availability is restricted by Definition 171.5, so a product witness cannot be detached from the two visible unit rows accepted by the ring-of-integers dependency.

Conversely, assume the two visible unit rows, their inverse/unit-product witnesses, the inherited operation row, the shared Cont readbacks, and the shared Pkg provenance are all present over the same $\mathcal{O}$. Apply Definition 171.13 to repack the inherited product endpoint as a visible $\mathsf{DirichletUnit}^{\uparrow}$ unit row with its own inverse/unit-product witness.

These two readings use the same rows and the same dependency boundary. Hence validity of the inherited product witness for $\mathsf{DirichletUnit}^{\uparrow}$ is exactly the presence of the listed packet data over the shared $\mathsf{RingOfIntegers}^{\uparrow}$ dependency. □

Theorem 171.16 (DirichletUnit root Regulator input exhaustion)**.** *Every* $\mathsf{Regulator}^{\uparrow}$ *consumer reading from a* $\mathsf{DirichletUnit}^{\uparrow}$ *root packet can read only the unary rank witness ν, the finite basis and logarithmic ledger row β, visible unit rows, inverse rows, inherited product rows, the* $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *dependency rows, finite* Cont *readbacks, and* Pkg *provenance. It cannot read a determinant, analytic regulator value, host logarithm equality, or standard bridge from* $\mathsf{DirichletUnit}^{\uparrow}$ *alone.*

Proof. Project the consumer surface from Definition 171.7. The only rank-facing input is the unary rank witness ν, and the only basis or logarithm-facing input is the finite ledger row β carried by Definition 171.1.

Next project the unit-facing rows. The visible unit rows and inverse rows are supplied by Definition 171.2 and inverted by Definition 171.14; inherited product rows are available exactly under Definition 171.15.

Finally inspect the dependency and provenance boundary. The packet exposes $\mathcal{O}$, $\mathcal{A}$, finite Cont readbacks, and ρ as Pkg provenance, and no other rows. A determinant, analytic regulator value, host logarithm equality, or standard bridge is therefore outside the $\mathsf{DirichletUnit}^{\uparrow}$ root input surface and must be supplied by a regulator-side certificate. □

Theorem 171.17 (DirichletUnit Regulator row exclusion)**.** *Let $\mathcal{D} = (\mathcal{O}, \mathcal{A}, u, \iota, \nu, \beta, \rho)$ be an accepted $\mathsf{DirichletUnit}^{\uparrow}$ carrier packet. A $\mathsf{Regulator}^{\uparrow}$ consumer whose input is read from $\mathcal{D}$ cannot project a determinant row, an analytic regulator value row, a host logarithm equality row, or a standard bridge row from that packet. The only regulator-facing rows available at the packet boundary are the visible unit rows, inverse/unit-product rows, unary rank row, finite basis/log ledger row, dependency rows, finite Cont readbacks, and Pkg provenance named in Definition 171.16.*

Proof. Project $\mathcal{D}$ through Definition 171.1. Its named entries are $\mathcal{O}$, $\mathcal{A}$, u, ι, ν, β, and ρ, together with the finite Cont readbacks attached to those entries.

Use Definition 171.15 to identify the product-facing part of the packet. The inherited product witness is available only where the two dependency certificates, visible unit rows, inverse rows, finite readbacks, and package provenance are simultaneously present.

Finally apply Definition 171.16. That theorem lists the complete root input surface for a $\mathsf{Regulator}^{\uparrow}$ consumer. Since determinants, analytic regulator values, host logarithm equalities, and standard bridge rows are absent from both the carrier packet and the exhausted consumer surface, no such row can be projected from $\mathcal{D}$. □

Theorem 171.18 (DirichletUnit visible rank-ledger projection)**.** *For an accepted $\mathsf{DirichletUnit}^{\uparrow}$ carrier packet $\mathcal{D} = (\mathcal{O}, \mathcal{A}, u, \iota, \nu, \beta, \rho)$, the unary rank witness ν and the finite basis/log ledger β are the only rank-facing rows exposed to $\mathsf{Regulator}^{\uparrow}$. Every rank-side readback factors through these two rows, the finite Cont ledgers of the packet, and the shared Pkg provenance ρ.*

Proof. Project the rank-facing entries from Definition 171.1. The packet has exactly one unary rank witness slot ν and exactly one finite basis/log ledger slot β; the dependency rows and provenance rows scope these slots but do not create a second rank source.

The visibility theorem Definition 171.4 identifies ν and β as displayed BHist rows stable under componentwise hsame, finite Cont readback, and package provenance. Thus the rank-side consumer reads a finite ledger surface rather than a host rank computation.

Apply Definition 171.16 to the same packet. Its exhausted $\mathsf{Regulator}^{\uparrow}$ surface includes no rank-facing row besides ν and β, so every accepted rank readback factors through those two displayed rows and the shared provenance ρ. □

Theorem 171.19 (DirichletUnit unit-product consumer scope)**.** *Let*

$$\mathcal{D}_0 = (\mathcal{O}, \mathcal{A}, u_0, \iota_0, \nu, \beta, \rho) \qquad \text{and} \qquad \mathcal{D}_1 = (\mathcal{O}, \mathcal{A}, u_1, \iota_1, \nu, \beta, \rho)$$

be accepted $\mathsf{DirichletUnit}^{\uparrow}$ carrier packets over the same dependency and provenance rows. The inherited product and inverse witnesses are consumable by a $\mathsf{Regulator}^{\uparrow}$ reader only at the packet boundary carrying the shared $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ row $\mathcal{O}$, the shared $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ row $\mathcal{A}$, the finite Cont readbacks, and the shared Pkg row ρ.

Proof. Project the two packets through Definition 171.1. The visible unit rows u_0, u_1, their inverse/unit-product rows ι_0, ι_1, and the dependency rows $\mathcal{O}, \mathcal{A}$ are all components of the same displayed boundary.

Apply Definition 171.15. It states that a product witness in the inherited $\mathsf{AbGroup}^{\uparrow}$ row is valid for $\mathsf{DirichletUnit}^{\uparrow}$ exactly where both visible unit rows, both inverse rows, the inherited operation row, the shared finite readbacks, and the shared package provenance are present over $\mathcal{O}$.

The inverse side is controlled by Definition 171.14. It projects every accepted inverse row back to the visible unit, the $\mathsf{NameCert}_{\mathsf{RingOfIntegers}^{\uparrow}}$ dependency, the inherited $\mathsf{AbGroup}^{\uparrow}$ inverse row, the finite readback, and ρ.

Combining these projections with Definition 171.16 gives the stated consumer scope. A $\mathsf{Regulator}^{\uparrow}$ reader can consume product and inverse evidence only as packet-boundary evidence tied to the two shared dependency certificates, Cont, and Pkg. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$.

Constructive story: A Dirichlet unit is carried by a finite packet whose entries are a $\mathsf{RingOfIntegers}^{\uparrow}$ dependency row, an $\mathsf{AbGroup}^{\uparrow}$ law row, a visible unit row, an inverse and unit-product witness row, a unary rank witness, a finite basis or logarithmic ledger row, and package provenance. The carrier accepts only packets whose unit-facing rows read back through the ring-of-integers dependency and whose law-facing rows read back through the abelian-group dependency, excluding host unit types, analytic logarithm equalities, determinant rows, regulator values, and bridge rows. Its classifier is componentwise hsame on the visible unit, inverse, law, rank, basis-ledger, readback, and provenance rows, with dependency provenance transported at the same time. The source specification is the RingOfIntegers-plus-AbGroup packet, the pattern specification is the visible unit group with inverse, product, rank, and finite basis/log-ledger rows, and the ledger keeps raw unit evidence, group-law evidence, and rank evidence visible through the named Cont readbacks. The five naming-certificate obligations come from the accepted packet as inhabitant, the equivalence laws of hsame, and the classifier-transport theorem that rebuilds the packet after each component has moved to a classified endpoint. A history h lies in the signature GAP of a $\mathsf{DirichletUnit}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer unit packet and evidence ledger, and token introduction names that answer as p; unpacked here, h is exactly the packet endpoint whose visible unit, inverse/product row, unary rank row, finite ledger, dependency rows, and package provenance all line up. The chapter registers this construction as $\mathsf{DirichletUnit}^{\uparrow}$ without closing analytic regulator values, determinant formulas, logarithmic embeddings, or a standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$, maturePackageClosed)

Scope closed: The visible-unit carrier, inverse and product witnesses, rank-ledger, $\mathsf{Regulator}^{\uparrow}$ consumer boundary, and unsupported-feature boundary rows are closed as the mature $\mathsf{DirichletUnit}^{\uparrow}$ package surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DirichletUnitUp.DirichletUnitHistoryCarrier\_visible\_unit\_rows`

Bridge status: none

Not claimed: This does not close analytic regulator values, determinant formulas, logarithmic embeddings, or standard bridge rows.

Upgrade path: Mature closure requires completeness examples and explicit unsupported-feature boundaries for analytic regulator values, determinant formulas, logarithmic embeddings, and standard bridge rows.

Chapter 172

A Concrete Naming Certificate for $\mathsf{Dedekind}^{\uparrow}$

The Dedekind-domain interface $\mathsf{Dedekind}^{\uparrow}$ packages an integrally closed Noetherian commutative domain in which every nonzero prime ideal is maximal. It depends on $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44) and $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81).

172.1 Carrier definition

Theorem 172.1 (Dedekind source obligation)**.** *A* $\mathsf{Dedekind}^{\uparrow}$ *source consists of a shared* BHist *carrier and classifier equipped with the cited* $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ *surfaces. The commutative-ring certificate fixes the visible ring package, and the ideal certificate is read over the same carrier and classifier rather than over a second ambient presentation.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton shared source obligation`
Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton source obligation`

Proof. Unpack the seed boundary for $\mathsf{Dedekind}^{\uparrow}$: the seed already names the $\mathsf{CommRing}^{\uparrow}$ source and the $\mathsf{Ideal}^{\uparrow}$ source as required inputs. The roadmap ten-field template then asks first for a carrier and classifier surface before any domain law can be read.

Those two inputs must therefore share the same BHist carrier and the same classifier transport. The proof records only this interface shape: it does not assert that the Dedekind certificate fields have been constructed. □

Theorem 172.2 (Dedekind domain-law obligation)**.** *For the shared* $\mathsf{Dedekind}^{\uparrow}$ *classifier, the domain-specific rows are integrally closedness, Noetherianity, and the assertion that every nonzero prime ideal is maximal. Each row is a certificate field over the shared classifier, with its own witness and stability requirement.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton domain law obligation`

Proof. The seed closure lists the integrally closed row, the Noetherian row, and the nonzero-prime-maximal row as the precise domain obligations. The roadmap ten-field template turns each listed row into visible data: a field, a witness, and classifier stability.

Since the carrier and ideal classifier have already been fixed by the source obligation, the three domain laws cannot be supplied as detached host properties. They must be read as certificate rows over that same BEDC classifier. □

Theorem 172.3 (Dedekind Noetherian source factorization)**.** *For any accepted* $\mathsf{Dedekind}^{\uparrow}$ *packet, the Noetherian witness row of Definition 172.2 factors through the* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier of Definition 3637.1 over the same* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$ *classifier. The integrally closed row and the nonzero-prime-maximal row remain Dedekind-specific rows over that shared classifier.*

Proof. Project the $\mathsf{Dedekind}^{\uparrow}$ source obligation from Definition 172.1. It fixes the commutative-ring source and the ideal source before the domain-law rows are read, so the Noetherian row has the same ambient carrier and classifier required by the $\mathsf{NoetherianRing}^{\uparrow}$ carrier.

Apply Definition 3637.2 to read the Noetherian witness as finite ideal-chain stabilization data over those two source rows. The $\mathsf{Dedekind}^{\uparrow}$ packet therefore consumes the reusable $\mathsf{NoetherianRing}^{\uparrow}$ surface rather than treating Noetherianity as an isolated internal row.

The remaining two domain-law rows are not absorbed by $\mathsf{NoetherianRing}^{\uparrow}$. Integrally closedness and nonzero-prime-maximality are projected from Definition 172.2 and stay attached to the Dedekind classifier as the Dedekind-specific part of the packet. □

Theorem 172.4 (Dedekind classifier ledger obligation)**.** *Classifier transport and the* $\mathsf{Dedekind}^{\uparrow}$ *ledger expose exactly the commutative-ring source, the ideal source, the integrally closed witness, the Noetherian witness, and the nonzero-prime-maximal witness. The ledger exposes no quotient data, no classical-choice data, and no* $\mathsf{StandardBridge}$ *data.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton classifier ledger obligation`

Proof. Unpack the exact-ledger boundary in the seed closure. Its upgrade path names the commutative-ring source, ideal source, three domain-law witnesses, classifier transport, and ledger boundary as the full local inventory.

The roadmap ten-field template then separates local certificate data from quotient, classical, and standard-bridge comparisons. The theorem therefore fixes the exposed ledger shape without claiming a completed $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$. □

172.2 The certificate

Theorem 172.5 (Dedekind scoped certificate packet)**.** *The scoped* $\mathsf{Dedekind}^{\uparrow}$ *certificate packet consists exactly of the shared* BHist *carrier and classifier, the cited* $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ *surfaces, the integrally closed row, the Noetherian row, the nonzero-prime-maximal row, classifier transport, and the exact* $\mathsf{Dedekind}^{\uparrow}$ *ledger boundary. This packet is scoped over the existing NameCert, CommRing, and Ideal primitives, and it contains no quotient presentation, no classical-choice datum, and no completed public* $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ *surface.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton scoped certificate packet`

Proof. Apply Definition 172.1 to fix the shared BHist carrier, the classifier, and the two cited source surfaces. Apply Definition 172.2 to read integrally closedness, Noetherianity, and nonzero-prime-maximality as rows over that same classifier.

Apply Definition 172.4 to identify the classifier transport and exact ledger boundary. Unpacking the roadmap ten-field template then packages these rows as the scoped local certificate inventory.

The scoped packet itself is not yet the exported public certificate surface for $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$. It is the local inventory from which that surface is assembled, and it does not pass through quotient representatives or classical choice. □

Theorem 172.6 (Dedekind certificate stability rows)**.** *Over the shared* BHist *carrier and classifier fixed by Definition 172.1, the integrally closed row, the Noetherian row, and the nonzero-prime-maximal row of Definition 172.2 are classifier-stable* $\mathsf{Dedekind}^{\uparrow}$ *certificate rows. Each row is transported only along the same classifier used by the* $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ *source surfaces, so no row is read over a detached carrier or a second classifier.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton certificate stability rows`

Proof. Begin with the shared carrier and classifier supplied by Definition 172.1. That theorem fixes the BHist presentation before any domain-law row is interpreted.

Apply Definition 172.2 to identify the three domain rows as certificate fields over that same classifier. Their witnesses are therefore stability data for the displayed $\mathsf{Dedekind}^{\uparrow}$ packet, not external predicates attached after the carrier has been chosen.

The conclusion is classifier stability row by row: integrally closedness, Noetherianity, and nonzero-prime-maximality all move only through the classifier already shared by the commutative-ring and ideal source surfaces. □

Theorem 172.7 (Dedekind exact ledger public surface)**.** *The public* $\mathsf{Dedekind}^{\uparrow}$ *ledger exposes exactly the* $\mathsf{CommRing}^{\uparrow}$ *source, the* $\mathsf{Ideal}^{\uparrow}$ *source, the integrally closed witness, the Noetherian witness, the nonzero-prime-maximal witness, classifier transport, and the exact* $\mathsf{Dedekind}^{\uparrow}$ *ledger boundary. It exposes no quotient data, no classical-choice data, and no* StandardBridge *data.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton exact ledger public surface`

Proof. Apply Definition 172.4. Its ledger boundary names the commutative-ring source, the ideal source, the three domain witnesses, and classifier transport as the visible inventory.

Exactness is read negatively as well as positively. The same ledger theorem excludes quotient representatives, classical-choice witnesses, and StandardBridge fields from the $\mathsf{Dedekind}^{\uparrow}$ certificate surface.

Thus the public ledger has exactly the displayed rows: every named row is exposed, and every excluded kind of host or bridge data remains outside the surface. □

Theorem 172.8 (Dedekind public NameCert surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ *surface over the concrete* BHist *carrier and the same classifier is assembled from Definition 172.1, Definition 172.2, Definition 172.4, Definition 172.6, and Definition 172.7. Its exported rows are exactly the shared* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$ *source surfaces, the three classifier-stable domain witnesses, classifier transport, and the exact ledger boundary.*

Lean4 ✓ `BEDC.Derived.DedekindUp.DedekindSingleton public namecert surface`

Proof. First fix the public carrier data. Apply Definition 172.1 to obtain the concrete BHist carrier, the common classifier, and the two source surfaces.

Next fix the domain content. Apply Definition 172.2 and Definition 172.6 to read integrally closedness, Noetherianity, and nonzero-prime-maximality as classifier-stable certificate rows over that same carrier.

Finally apply Definition 172.4 and Definition 172.7. These identify the transport and exact ledger rows, exclude quotient, classical-choice, and StandardBridge data, and repack the displayed inventory as the public $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$.

Constructive story: A Dedekind domain is carried by a shared BHist source where the $\mathsf{CommRing}^{\uparrow}$ surface and the $\mathsf{Ideal}^{\uparrow}$ surface use the same carrier and classifier before the domain-law rows are attached. The carrier accepts exactly that shared commutative-ring and ideal source together with the integrally closed, Noetherian, and nonzero-prime-maximal witness rows, while quotient presentations, classical-choice data, and standard-bridge fields are absent from the packet. Two accepted presentations classify together only through the same classifier used by the CommRing and Ideal source surfaces, and each of the three domain-law rows must be stable under that classifier. The source specification is the paired CommRing and Ideal dependency surface, the pattern specification is the three visible domain-law witnesses over that surface, and the ledger exposes exactly those sources, witnesses, classifier-transport rows, and exact boundary rows. The five naming-certificate obligations are arranged by the public surface: the shared carrier gives an inhabitant, hsame and the inherited classifier rows give reflexivity, symmetry, and transitivity, and classifier stability repacks any transported domain-law witness inside the same accepted carrier. A history h lies in the signature GAP of a $\mathsf{Dedekind}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it to an answer Dedekind packet and evidence ledger, and token introduction names that answer as p; concretely, the CommRing source, Ideal source, integrally closed row, Noetherian row, prime-maximal row, transport rows, and ledger boundary must all be the displayed rows for h. The chapter registers this construction as $\mathsf{Dedekind}^{\uparrow}$ without adding a quotient construction, a classical-choice presentation, a standard bridge, or a separate verification target.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$, maturePackageClosed)

Scope closed: The paper-side scope is the public $\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$ surface in Definition 172.8, grounded by the shared-source row, the three domain-law rows, classifier stability, classifier transport, and the exact ledger boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dedekind}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DedekindUp.DedekindSingleton\_public\_namecert\_surface`

Bridge status: none

Not claimed: This chapter does not close a Lean verification target, standard bridge, quotient construction, or classical-choice presentation.

Upgrade path: No further paper theory-axis upgrade is recorded for this chapter; bridge verification remains a separate axis.

Chapter 173

A Concrete Naming Certificate for $\mathsf{LocalField}^{\uparrow}$

The local-field interface $\mathsf{LocalField}^{\uparrow}$ packages a complete discretely valued field with finite residue field. It depends on $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45) and $\mathsf{NameCert}_{\mathsf{Padic}^{\uparrow}}$ (chapter 100).

173.1 Carrier definition

Definition 173.1 (LocalField valued BHist carrier)**.** *A* $\mathsf{LocalField}^{\uparrow}$ *carrier is a finite* BHist *packet consisting of a displayed* $\mathsf{Field}^{\uparrow}$ *row, a* $\mathsf{Padic}^{\uparrow}$ *valuation row on the same field endpoint, a finite residue row, a completeness-witness ledger row, and the visible endpoint data for the valued field, valuation, unit surface, and residue surface. The packet is accepted only when these rows are present in one finite ledger and the valuation row supplies the route from field elements to residue data through the displayed dependency certificates.*

The carrier is the visible valued-field packet. It does not include arbitrary completions, quotient fields, class-field reciprocity data, or a standard bridge object.

Lean4 (definition) `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier`

Theorem 173.2 (LocalField valuation classifier stability)**.** *Let two* $\mathsf{LocalField}^{\uparrow}$ *carrier packets have* $\mathsf{Field}^{\uparrow}$ *rows and* $\mathsf{Padic}^{\uparrow}$ *valuation rows related by displayed* hsame *classifiers. If the residue row, unit row, and completeness-witness ledger are read only through those transported field and valuation rows, then the transported endpoints repack as an accepted* $\mathsf{LocalField}^{\uparrow}$ *carrier with the same valued-field ledger shape.*

Lean4 ✓ `BEDC.Derived.LocalFieldUp.LocalFieldValuationClassifier stability`

Proof. Project the $\mathsf{Field}^{\uparrow}$ dependency certificate. It transports the displayed field endpoint and the field rows used by the local-field packet.

Project the $\mathsf{Padic}^{\uparrow}$ dependency certificate. The valuation endpoint is transported together with the visible valuation row, so the residue and unit rows still have their valuation source after transport.

Replay the carrier ledger on the transported rows. Since the completeness witness is a ledger row tied to the same field and valuation endpoints, it is read from the transported packet without adding a new host completion. □

Theorem 173.3 (LocalField residue ledger exactness)**.** *For every accepted* $\mathsf{LocalField}^{\uparrow}$ *carrier, the residue ledger exposes exactly the valuation row, unit row, residue row, and their displayed endpoint data. It does not expose an arbitrary completion, quotient field, class-field reciprocity structure, or external local-field model.*

Proof. Inspect the carrier of Definition 173.1. The residue side of the packet consists of the valuation row, unit row, residue row, and endpoints recorded in the finite BHist ledger.

The completeness witness is a ledger entry for the same displayed endpoint data. It certifies that the local packet has the advertised finite surface, but it does not introduce a host completion or quotient field.

Thus every readable residue datum is one of the listed rows, and none of the excluded host structures is part of the accepted packet. □

Lean4 ✓ `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier residue ledger exactness`

Theorem 173.4 (LocalField NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocalField}^{\uparrow}}$ *surface is the carrier of Definition 173.1, the valuation classifier over the displayed* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{Padic}^{\uparrow}$ *rows, the stability clause of Definition 173.2, the residue ledger exactness clause of Definition 173.3, and the source coverage assertion that every accepted local-field row is read from those finite packet rows.*

Lean4 ✓ `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier namecert obligation surface`

Proof. Use the valued BHist carrier as the source predicate. The carrier lists the field, valuation, residue, completeness, and endpoint rows allowed in the local certificate.

The classifier is the displayed comparison of field and valuation rows. Its transport obligation is exactly Definition 173.2.

The ledger and exactness obligations are exactly Definition 173.3 together with the carrier definition. These rows cover every accepted local-field packet and exclude all host structures outside the finite ledger. □

Theorem 173.5 (LocalField bridge boundary exclusions)**.** *The* $\mathsf{LocalField}^{\uparrow}$ *boundary supplies only the BEDC valued-field packet over the displayed* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{Padic}^{\uparrow}$ *dependency rows. It does not assert arbitrary completions, quotient fields, full local class field theory, or standard-model bridge claims.*

Proof. Read the boundary through Definition 173.4. The exported rows are the carrier, classifier, stability, residue ledger, exactness, and source coverage rows.

Each exported row factors through the finite packet of Definition 173.1. That packet has no constructor for host completions, quotient fields, reciprocity theory, or a standard-model bridge.

Therefore the boundary records only the displayed BEDC valued-field surface. □

Lean4 ✓ `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier bridge boundary exclusions`

Theorem 173.6 (LocalField valued-field consumer threshold)**.** *Any consumer of* $\mathsf{NameCert}_{\mathsf{LocalField}^{\uparrow}}$ *may project only the displayed* $\mathsf{Field}^{\uparrow}$ *row, the* $\mathsf{Padic}^{\uparrow}$ *valuation row, the unit row, the residue row, the completeness-witness ledger row, endpoint data, and visible* $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}$ *provenance from Definition 173.4. No arbitrary completion, quotient field, reciprocity structure, external local-field model, or standard bridge becomes available.*

Proof. Project the $\mathsf{Field}^{\uparrow}$ and $\mathsf{Padic}^{\uparrow}$ dependency certificates from the obligation surface. These projections give the field row, valuation row, unit row, completeness-witness ledger row, endpoint data, and displayed provenance without adding a host carrier.

Use Definition 173.3 to read the residue row as finite ledger data. The readable residue entries are therefore exactly the ones already present in the valued BHist packet.

Finally apply Definition 173.5. The excluded host completions, quotient fields, reciprocity structures, external models, and standard bridges are outside the certificate exported to downstream consumers. □

Lean4 ✓ `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier valued field consumer threshold`

173.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{LocalField}^{\uparrow}}$.

Constructive story: A local-field packet is generated from a valued BHist carrier whose displayed $\mathsf{Field}^{\uparrow}$ row, $\mathsf{Padic}^{\uparrow}$ valuation row, residue row, completeness-witness ledger, endpoint data, hsame transport, Cont displays, and Pkg provenance are all visible in one finite certificate surface. The valuation classifier, residue ledger, and bridge-boundary exclusions keep arbitrary host completions, quotient local fields, reciprocity data, external models, and standard bridges outside the exported carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalField}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The obligation scope consists of the valued BHist carrier, valuation classifier stability, residue ledger exactness, bridge-boundary exclusions, and valued-field consumer threshold over $\mathsf{Field}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, hsame, Cont, and Pkg.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalField}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LocalFieldUp.LocalFieldValuedBHistCarrier\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: No host completion, quotient local field, reciprocity theorem, external local-field model, or standard bridge is claimed.

Upgrade path: Bridged closure requires a standard comparison surface for the valued-field consumer threshold.

Chapter 174

A Concrete Naming Certificate for $\mathsf{Cyclotomic}^{\uparrow}$

The cyclotomic-field interface $\mathsf{Cyclotomic}^{\uparrow}$ packages the splitting field of $X^n - 1$ over the rational-number certificate. It depends on $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168) and $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ (chapter 165).

174.1 Carrier definition

Definition 174.1 (Cyclotomic root carrier)**.** *A* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier is a* BHist *packet*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

where $\mathcal{N}$ *is a source* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *row,* $n : \mathsf{BHist}$ *is a positive unary exponent row,* $p : \mathsf{BHist}$ *is the rational polynomial ledger for* $X^n - 1$*,* $\mathcal{S}$ *is a source* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *row over that polynomial ledger,* $\zeta : \mathsf{BHist}$ *is the chosen primitive-root endpoint,* α *is the finite* Cont *row accepting* ζ *as a root and recording the factorization rows inside* $\mathcal{S}$*,* λ *is the* hsame *comparison tying the NumFieldUp source and the SplittingFieldUp base row, and* ρ *records the* Pkg *provenance of the two source certificates.*

The accepted carrier is exactly this displayed history tuple. It carries no quotient of primitive roots and no host equality of fields; every endpoint read is mediated by BHist *rows,* hsame *transport,* Cont *ledgers, and the paired* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *provenance rows.*

Definition 174.2 (Cyclotomic rational polynomial classifier)**.** *The* $\mathsf{Cyclotomic}^{\uparrow}$ *classifier relates two carriers of Definition 174.1 when their* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source rows, positive unary exponent rows, rational polynomial ledgers for* $X^n - 1$*,* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source rows, and primitive-root endpoints are connected by* hsame *rows whose endpoints are supported by the displayed* Cont *and* Pkg *provenance ledgers. The classifier transports the NumFieldUp source, exponent, polynomial row, splitting-field row, and primitive-root endpoint as carried* BHist *data, not as equality of host fields or as a quotient of roots.*

Definition 174.3 (Cyclotomic root-layer obligation surface)**.** *The root-layer obligation surface for* $\mathsf{Cyclotomic}^{\uparrow}$ *is the concrete* BHist *row tuple*

$$(\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho).$$

Here $\mathcal{N}$ is the NumFieldUp source certificate row, n is the positive unary exponent row, p is the rational polynomial ledger for $X^n - 1$, $\mathcal{S}$ is the SplittingFieldUp source certificate row over p, ζ is the primitive-root endpoint, α is the finite Cont *row accepting ζ as a root and recording the factorization reads, λ is the* hsame *base-comparison row, and ρ is the* Pkg *provenance row for the two source certificates.*

This surface is a root-layer source packet, not a host splitting field. It contains no valuation row, class-field reciprocity datum, automorphism-group structure, or equality of ambient fields.

Theorem 174.4 (Cyclotomic root-layer source coverage)**.** *Every accepted carrier of Definition 174.1 covers exactly the rows of Definition 174.3. The coverage reads $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ as visible* BHist *data, and it exports no additional root-layer source row.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier root layer source coverage`

Proof. Unpack the displayed carrier of Definition 174.1. Its accepted history tuple is precisely $(\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$.

Compare this tuple with Definition 174.3. Each entry has the same role: source NumFieldUp certificate, unary exponent, rational polynomial ledger, source SplittingFieldUp certificate, primitive-root endpoint, root-acceptance continuation, base comparison, and provenance package.

Since the carrier definition contains no further root source coordinate, the projection is exact. Valuations, reciprocity data, automorphism groups, and host field equality are outside the displayed carrier and cannot be exported by this coverage step. □

Theorem 174.5 (Cyclotomic root-layer classifier transport)**.** *If two accepted* Cyclotomic$^{\uparrow}$ *carriers are related by the classifier of Definition 174.2, then their root-layer obligation surfaces are transported componentwise by the carried* hsame *rows, the root-acceptance* Cont *rows, and the* Pkg *provenance rows. The transport preserves the NumFieldUp source, exponent, polynomial ledger, SplittingFieldUp source, primitive-root endpoint, and the base-comparison evidence as* BHist *data.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier root layer classifier transport`

Proof. Project the classifier row using Definition 174.2. The projected data are the componentwise hsame rows for $\mathcal{N}$, n, p, $\mathcal{S}$, and ζ, together with the displayed Cont and Pkg support.

The Cont rows keep the primitive-root acceptance and factorization reads attached to the transported endpoints. The Pkg rows keep the NumFieldUp and SplittingFieldUp certificate provenance attached to the same source packet, while λ supplies the base comparison between the two source layers.

Repack these transported components in the order of Definition 174.3. The result is the root-layer surface on the target carrier, with no passage through a quotient of roots or host-field equality. □

Theorem 174.6 (Cyclotomic root-layer consumer boundary)**.** *Any downstream consumer of the* Cyclotomic$^{\uparrow}$ *root layer may read only the surface of Definition 174.3 and its classifier transport from Definition 174.5. In particular, the root-layer boundary supplies no valuation theory, no class-field reciprocity law, no automorphism-group action, and no equality between host fields.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier consumer boundary`

Proof. Start from the source coverage of Definition 174.4. It limits the exported root-layer rows to the eight visible entries of Definition 174.3.

Next apply Definition 174.5. It states that consumer-side transport is exactly componentwise hsame plus the carried Cont and Pkg ledgers, so any transported read remains inside the same eight-row surface.

The excluded structures are not coordinates of the carrier, not generators of the classifier, and not fields of the provenance package. They therefore cannot be consumed at this boundary without a separate certificate. □

Theorem 174.7 (Cyclotomic classifier reflexivity)**.** *For every accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, the classifier of Definition 174.2 relates $\mathcal{C}_\zeta$ *to itself. The witnessing row is the componentwise reflexive* hsame *support on the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source* $\mathcal{N}$*, the exponent row* n*, the rational polynomial ledger* p *for* $X^n - 1$*, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source* $\mathcal{S}$*, and the primitive-root endpoint* ζ*, together with the unchanged* Cont *row* α*, base comparison row* λ*, and* Pkg *provenance row* ρ*.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier source triad obligation`

Proof. Start from the displayed carrier. Each carried BHist component has the reflexive hsame row at its own endpoint: $\mathcal{N}$, n, p, $\mathcal{S}$, and ζ are therefore transported to themselves.

The finite Cont row α already accepts ζ as a root and records the factorization rows inside $\mathcal{S}$. The comparison row λ already ties the NumFieldUp source to the SplittingFieldUp base row, and ρ is already the provenance package for the two source certificates.

Repacking these five reflexive hsame rows with the displayed Cont and Pkg ledgers gives exactly the classifier row from $\mathcal{C}_\zeta$ to $\mathcal{C}_\zeta$. □

Theorem 174.8 (Cyclotomic classifier symmetry)**.** *Let*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

and

$$\mathcal{C}'_\eta = (\mathcal{N}', n', p', \mathcal{S}', \eta, \alpha', \lambda', \rho')$$

be accepted $\mathsf{Cyclotomic}^{\uparrow}$ *carriers. If the classifier of Definition 174.2 relates* $\mathcal{C}_\zeta$ *to* $\mathcal{C}'_\eta$*, then it also relates* $\mathcal{C}'_\eta$ *to* $\mathcal{C}_\zeta$ *by reversing the carried* hsame *rows for the NumFieldUp sources, exponent rows, polynomial ledgers, SplittingFieldUp sources, and primitive-root endpoints, while reading the same* Cont *acceptance rows and* Pkg *provenance rows from the opposite side.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootClassifier symm`

Proof. Project the given classifier row to its five hsame components. These components compare $\mathcal{N}$ with $\mathcal{N}'$, n with n', p with p', $\mathcal{S}$ with $\mathcal{S}'$, and ζ with η.

Reverse each projected hsame row. The reversed rows compare the same carried histories in the order required for a classifier from $\mathcal{C}'_\eta$ to $\mathcal{C}_\zeta$.

It remains to account for the ledgers. The acceptance rows α' and α still support the endpoint and factorization reads at their respective sides, while λ', λ, ρ', and ρ provide the same base-comparison and provenance evidence with source and target interchanged. Repacking these rows gives the opposite classifier. □

Theorem 174.9 (Cyclotomic classifier transitivity)**.** *Let*

$$\mathcal{C}_0 = (\mathcal{N}_0, n_0, p_0, \mathcal{S}_0, \zeta_0, \alpha_0, \lambda_0, \rho_0), \quad \mathcal{C}_1 = (\mathcal{N}_1, n_1, p_1, \mathcal{S}_1, \zeta_1, \alpha_1, \lambda_1, \rho_1), \quad \mathcal{C}_2 = (\mathcal{N}_2, n_2, p_2, \mathcal{S}_2, \zeta_2, \alpha_2, \lambda_2, \rho_2)$$

be accepted $\mathsf{Cyclotomic}^{\uparrow}$ *carriers. If the classifier relates* $\mathcal{C}_0$ *to* $\mathcal{C}_1$ *and also relates* $\mathcal{C}_1$ *to* $\mathcal{C}_2$*, then it relates* $\mathcal{C}_0$ *to* $\mathcal{C}_2$ *by composing the* hsame *rows componentwise through the shared middle carrier for the NumFieldUp source, exponent, polynomial ledger, SplittingFieldUp source, and primitive-root endpoint, and by repacking the shared* Cont *acceptance rows and* Pkg *provenance rows.*

Proof. Project the first classifier row to the comparisons from $\mathcal{C}_0$ to $\mathcal{C}_1$, and project the second classifier row to the comparisons from $\mathcal{C}_1$ to $\mathcal{C}_2$. The middle carrier supplies the common endpoint for each pair of projected hsame rows.

Compose the NumFieldUp source comparisons through $\mathcal{N}_1$, the exponent comparisons through n_1, the polynomial-ledger comparisons through p_1, the SplittingFieldUp source comparisons through $\mathcal{S}_1$, and the primitive-endpoint comparisons through ζ_1.

The Cont rows α_0, α_1, and α_2 keep the endpoint and factorization reads attached to the three carriers, and the provenance rows ρ_0, ρ_1, and ρ_2 keep the source packages aligned. Repacking the composed hsame rows with these ledgers yields the classifier from $\mathcal{C}_0$ to $\mathcal{C}_2$. □

174.2 The certificate

Theorem 174.10 (Cyclotomic source-triad obligation)**.** *For every accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, the $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *source triad is exactly the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *row* $\mathcal{N}$*, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *row* $\mathcal{S}$ *over the rational polynomial ledger* p *for* $X^n - 1$*, and the primitive-root endpoint row* ζ *accepted by* α*. The triad is covered by the displayed* BHist *rows, transported by the classifier of Definition 174.2, and closed under the* hsame*,* Cont*, and* Pkg *rows* λ*,* α*, and* ρ*; no fourth source row is exported by the certificate.*

Proof. Project the carrier to $\mathcal{N}$, $\mathcal{S}$, and ζ. The exponent row n and polynomial ledger p locate $\mathcal{S}$ at the displayed equation $X^n - 1$, while the finite Cont row α attaches the primitive endpoint to the root and factorization surface.

Use Definition 174.15 and Definition 174.16 to transport the NumFieldUp and SplittingFieldUp rows through hsame. Use Definition 174.18 for the endpoint component.

Finally repack the three projected source rows with λ, α, and ρ. The obligation surface of Definition 174.17 lists no additional source component, so the triad is both covered and complete. □

Theorem 174.11 (Cyclotomic p-adic consumer obligation)**.** *A p-adic consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *may read only the carrier rows* $\mathcal{N}$*,* n*,* p*,* $\mathcal{S}$*,* ζ*,* α*,* λ*, and* ρ *from Definition 174.1, with* p *interpreted as the rational polynomial ledger for* $X^n - 1$ *and not as a valuation or completion row. Any valuation, completion, or measure row used downstream must be supplied by a separate* $\mathsf{Padic}^{\uparrow}$ *or* $\mathsf{Iwasawa}^{\uparrow}$ *certificate and may enter only through its own* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *interface.*

Proof. Begin with the displayed cyclotomic carrier. Its rows provide a NumFieldUp source, a positive exponent, the rational polynomial ledger, the SplittingFieldUp source, the primitive endpoint, the acceptance ledger, the base comparison, and the provenance package.

Apply Definition 174.25 to identify the polynomial row as the $X^n - 1$ ledger and Definition 174.28 to identify the accepted factorization rows. Apply Definition 174.32 for the downstream completion boundary.

The carrier and classifier contain no valuation, completion, or measure field. Therefore any p-adic consumer starts from the cyclotomic source package alone and must cite the separate certificate that provides the missing analytic rows. □

Theorem 174.12 (Cyclotomic Iwasawa tower obligation)**.** *An* $\mathsf{Iwasawa}^{\uparrow}$ *tower that consumes* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *receives exactly the cyclotomic layer formed by the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *row* $\mathcal{N}$*, the exponent row* n*, the primitive-root endpoint* ζ*, the splitting-field factorization rows over* p*, the acceptance row* α*, and the provenance row* ρ*. The tower may transport these rows through the carried* hsame *and* Cont *structure, but* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *alone supplies no tower transition map, no* p*-adic measure row, no class-group row, and no host-field equality.*

Proof. Project the accepted carrier to the source, exponent, polynomial, splitting-field, endpoint, acceptance, comparison, and provenance rows. These are the only rows available before an Iwasawa-specific certificate is attached.

Use Definition 174.26 and Definition 174.29 to restrict the tower input surface to the projected cyclotomic source rows. Use Definition 174.30 to repack those rows as the cyclotomic layer of the tower.

The excluded tower maps and analytic rows do not occur in the carrier, the classifier, or the obligation surface. They must therefore be supplied by later NameCert rows rather than inferred from the cyclotomic packet. □

Theorem 174.13 (Cyclotomic class-field source obligation)**.** *A* $\mathsf{ClassField}^{\uparrow}$ *source interface may cite* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *only through the displayed* BHist *rows* $\mathcal{N}$*,* n*,* p*,* $\mathcal{S}$*, and* ζ*, the* Cont *row* α*, the* hsame *comparison* λ*, and the* Pkg *provenance row* ρ *of Definition 174.1. This interface provides cyclotomic source data for a later class-field certificate, but it does not provide a class group, reciprocity map, local symbol, norm residue map, or equality of ambient fields.*

Proof. Project the carrier rows and read them through the classifier of Definition 174.2. The only transport available is the carried hsame support on the source, exponent, polynomial, splitting-field, and endpoint rows, together with the displayed Cont and Pkg provenance.

Apply Definition 174.37. Its interface is the repacking of the cyclotomic source boundary, the p-adic input boundary, and the classifier transport already present in this chapter.

Since the named class-field operations are not entries of the cyclotomic carrier and are not produced by its classifier, the class-field consumer receives a source interface only. Any class-field operation must arrive with a separate NameCert row. □

Theorem 174.14 (Cyclotomic Galois transport obligation)**.** *Every Galois-facing transport read from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *is a transport of the primitive-root endpoint and the splitting-field root and factorization rows along the* hsame *support carried by the classifier of Definition 174.2. The transport is mediated by the* BHist *rows of Definition 174.1, the* Cont *acceptance row, and the* Pkg *provenance row; it exports no automorphism-group object, no quotient of roots, and no host-field equality.*

Proof. Project the classifier to the source, exponent, polynomial, splitting-field, and endpoint comparisons. These hsame rows are the only Galois-facing transport data available from the cyclotomic certificate.

Use Definition 174.24 for the primitive endpoint and Definition 174.35 for the polynomial and splitting-field rows. Use Definition 174.31 and Definition 174.33 to keep the readback inside endpoint and factorization transport.

The proof closes by inspecting the carrier and classifier fields. Neither contains an automorphism group, a quotient-root representative, or a host-field equality datum, so no downstream Galois consumer can read such data from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ alone. □

Theorem 174.15 (Cyclotomic splitting-field stability)**.** *For any two* $\mathsf{Cyclotomic}^\uparrow$ *carriers related by the classifier of Definition 174.2,* hsame *transport along the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *classifier preserves the accepted primitive-root endpoint, the root row for* $X^n - 1$*, and the factorization rows carried by the* BHist *splitting-field ledger. The transported certificate keeps the same* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *source comparison, the same positive exponent comparison, and the same* Cont *and* Pkg *provenance rows.*

Proof. Project the two cyclotomic packets to their $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ components. The splitting-field classifier supplies the transported root row and the transported factorization rows for the polynomial ledger classified with $X^n - 1$.

Use the carrier comparison to identify the exponent row and the rational polynomial row by hsame. The primitive-root endpoint is then read through the same Cont acceptance ledger, so its root status and the listed factorization rows remain attached to the transported splitting-field package.

The $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ comparison and the Pkg provenance row are projected unchanged from the cyclotomic carrier. Repacking these projected rows gives another accepted $\mathsf{Cyclotomic}^\uparrow$ carrier with the same public root and factorization evidence. □

Theorem 174.16 (Cyclotomic number-field embedding ledger)**.** *When a* $\mathsf{Cyclotomic}^\uparrow$ *carrier is read as a number-field extension, its embedding and provenance surface is the* BHist *ledger inherited from* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *together with the* hsame *bridge to the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *base row, the rational polynomial ledger, and the primitive-root endpoint. The readback preserves the* Cont *acceptance rows and the* Pkg *provenance rows that certify the cyclotomic field as generated by a root of* $X^n - 1$*.*

Proof. Start with the NumFieldUp source row in the carrier. Its embedding and provenance ledger is already a public row of $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$, so reading the cyclotomic packet as a number-field extension first projects that ledger.

Next attach the hsame comparison tying the NumFieldUp row to the base row consumed by $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$. The polynomial row and primitive-root endpoint are carried in the same tuple, and the Cont acceptance row records that the endpoint is read from the splitting-field root surface.

Thus the number-field readback retains both sources of provenance. It does not infer an embedding from an ambient field equality; it projects the displayed BHist rows and the carried Pkg ledger. □

Theorem 174.17 (Cyclotomic NameCert obligation surface)**.** *The paper-side* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *obligation surface consists of the* BHist *carrier of Definition 174.1, the* hsame *classifier of Definition 174.2, the splitting-field stability row over* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$*, the number-field embedding ledger over* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$*, the exact root and factorization* Cont *rows for* $X^n - 1$*, and the shared* Pkg *provenance rows. A downstream* $\mathsf{Iwasawa}^\uparrow$ *packet may consume exactly those rows from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier namecert obligation surface`

Proof. Collect the carrier tuple with its classifier rows. The carrier supplies the NumFieldUp source, exponent, rational polynomial ledger, SplittingFieldUp source, primitive-root endpoint, acceptance ledger, and provenance rows.

Add the two certificate obligations above. Splitting-field stability provides the transported root and factorization rows, while the number-field embedding ledger provides the readback surface needed by a downstream number-theoretic certificate.

Exactness is by projection from this list. Every public cyclotomic move uses one of the displayed BHist, hsame, Cont, or Pkg rows, and no consumer receives a stronger analytic, quotient, or host-field equality principle from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ alone. □

Theorem 174.18 (Cyclotomic primitive-root readback exactness)**.** *Every accepted carrier of Definition 174.1 reads its primitive-root endpoint ζ only through the displayed positive exponent row, the rational polynomial ledger for $X^n - 1$, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *source row, and the* Cont *acceptance ledger α. Conversely, those projected rows repack the primitive-root readback of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *without adding host equality of fields or quotienting primitive roots.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier primitive root readback exactness`

Proof. Project an accepted cyclotomic carrier to the entries n, p, $\mathcal{S}$, ζ, and α in the tuple of Definition 174.1. The positive exponent row fixes the displayed power n, the rational polynomial row fixes the ledger for $X^n - 1$, and the splitting-field source supplies the public root surface from which ζ is read.

The acceptance ledger α is the only row in the carrier that attaches the endpoint to the root and factorization surface. By Definition 174.15, classifier transport preserves that attachment through hsame rows rather than through an ambient equality of fields.

For the converse direction, start with the same projected exponent, polynomial, splitting-field, endpoint, and acceptance rows, then reattach the NumFieldUp source comparison and the Pkg provenance supplied by the carrier. Definition 174.16 identifies these as the number-field readback rows, so the repacked certificate has exactly the primitive-root readback advertised by $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$. □

Theorem 174.19 (Cyclotomic factorization ledger coverage)**.** *Each public factorization row exposed by* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *is projected from the carried* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *packet and transported by the classifier of Definition 174.2. No factorization row is available outside the displayed* BHist*,* hsame*,* Cont*, and* Pkg *rows of the cyclotomic carrier.*

Proof. Begin with a public factorization row of the cyclotomic certificate. The obligation surface of Definition 174.17 lists factorization evidence only in the splitting-field ledger and the acceptance ledger attached to the primitive-root endpoint.

Project the carried $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ packet. Its BHist factorization rows are then transported along the cyclotomic classifier, whose definition requires hsame support for the NumFieldUp source, exponent row, rational polynomial ledger, splitting-field row, and endpoint.

The remaining carrier entries are provenance rows. They record Cont acceptance and Pkg origin, but they do not introduce an independent factorization surface. Thus every public factorization row is covered by the splitting-field projection and classifier transport, and there is no separate source from which another row can be read. □

Theorem 174.20 (Cyclotomic splitting-field source exactness)**.** *The concrete source surface consumed by downstream* $\mathsf{Cyclotomic}^{\uparrow}$ *users consists exactly of the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row, the positive exponent row, the rational polynomial ledger for* $X^n - 1$*, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source row, the primitive-root endpoint, the root row, the factorization rows, and the shared provenance rows. These rows are covered by the carrier, sound under the classifier, and complete for every downstream readback of the cyclotomic source surface.*

Proof. Coverage follows by projecting the carrier of Definition 174.1. Its displayed tuple contains the NumFieldUp source, exponent row, rational polynomial ledger, splitting-field source, primitive-root endpoint, acceptance ledger, comparison row, and provenance row; the root and factorization entries are the rows recorded by the splitting-field packet and the acceptance ledger.

Soundness is the transport statement of Definition 174.15. The classifier of Definition 174.2 carries the source, polynomial, splitting-field, and endpoint comparisons by hsame, while the Cont and Pkg rows keep the accepted root and factorization evidence attached to the same provenance.

Completeness is the reverse repacking step. A downstream user of $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ receives only the rows listed by Definition 174.17; Definition 174.16 accounts for the NumFieldUp readback, and the preceding two exactness theorems account for the primitive-root and factorization readbacks. Reassembling these projections gives the whole concrete source surface and nothing stronger. □

Theorem 174.21 (Cyclotomic root-power ledger coverage)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, the displayed positive exponent row n*, the rational polynomial ledger* p *for* $X^n - 1$*, the splitting-field row* $\mathcal{S}$*, the primitive endpoint* ζ*, and the* Cont *acceptance row* α *cover every public read of a root power from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*. No separate root-power ledger is exported outside these* BHist*,* Cont*,* hsame*, and* Pkg *rows.*

Proof. Project the carrier of Definition 174.1. The exponent coordinate n, polynomial coordinate p, splitting-field coordinate $\mathcal{S}$, endpoint coordinate ζ, and acceptance coordinate α are the only rows from which a root-power read can be formed.

Use Definition 174.25 to identify p as the carried rational polynomial ledger for $X^n - 1$. Use Definition 174.27 to attach ζ to that displayed exponent and polynomial row through the same Cont ledger.

The splitting-field and factorization surface is already exact by Definition 174.20. The provenance row ρ and comparison row λ record transport and package origin; they do not add another source for root powers. Thus every public root-power read is covered by the listed row family. □

Theorem 174.22 (Cyclotomic primitive endpoint classifier stability)**.** *If two accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carriers are related by the classifier of Definition 174.2, then the primitive endpoint and its root-power ledger transport componentwise by the displayed* hsame *rows on the NumFieldUp source, exponent, rational polynomial ledger, SplittingFieldUp source, endpoint,* Cont *acceptance ledger, and shared* Pkg *provenance. The transport does not quotient primitive roots and does not assert equality of ambient host fields.*

Proof. Unpack the classifier of Definition 174.2. It supplies componentwise hsame rows for the source, exponent, polynomial, splitting-field, endpoint, acceptance, comparison, and provenance entries of the two carriers.

Apply Definition 174.21 to the source carrier and to the target carrier. The root-power surface on each side is the same displayed family of exponent, polynomial, splitting-field, endpoint, and acceptance rows.

Finally use Definition 174.15 and Definition 174.27. These statements transport the endpoint and its root-power ledger through the carried hsame rows, so the primitive-root surface remains classifier-stable without quotienting roots or identifying host fields. □

Theorem 174.23 (Cyclotomic downstream source exactness)**.** *An* $\mathsf{Iwasawa}^{\uparrow}$*,* $\mathsf{Padic}^{\uparrow}$*, or* $\mathsf{ClassField}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *receives exactly the carrier rows* $\mathcal{N}, n, p, \mathcal{S}, \zeta$*, the acceptance row* α*, the base comparison row* λ*, and the provenance row* ρ*, together with their displayed* BHist*,* Cont*,* hsame*, and* Pkg *support. It receives no valuation row, class-group row, reciprocity row, local-symbol row, completion row, or host-field equality row from the cyclotomic certificate alone.*

Proof. Project the carrier of Definition 174.1. Its displayed tuple contains exactly the rows $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ.

For the $\mathsf{Iwasawa}^{\uparrow}$ and $\mathsf{Padic}^{\uparrow}$ reads, apply Definition 174.29 and Definition 174.32. These statements repack the same projected rows as source data while excluding completion, measure, and valuation data unless another certificate supplies them.

For the $\mathsf{ClassField}^{\uparrow}$ read, apply Definition 174.37. The classifier transport is the componentwise transport of Definition 174.22, so the consumer receives exactly the listed rows and no class group, reciprocity map, local symbol, or host-field equality. □

Theorem 174.24 (Cyclotomic primitive-root endpoint transport)**.** *For classified* $\mathsf{Cyclotomic}^{\uparrow}$ *carriers, the primitive-root endpoint is transported by the carried* hsame *row of Definition 174.2. The transport preserves the displayed* Cont *acceptance ledger inside the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *row, including the root row for* $X^n - 1$*, the factorization rows listed by that splitting-field certificate, and the shared* Pkg *provenance of the NumFieldUp and SplittingFieldUp sources.*

Proof. Project the classifier of Definition 174.2. Its endpoint component is an hsame comparison whose endpoints are supported by the same cyclotomic carrier tuple and by the displayed Cont ledger.

Apply Definition 174.15 to the splitting-field components. This transports the root row for $X^n - 1$ and the factorization rows without replacing them by a host-field equality or by a quotient of primitive roots.

The remaining data are the source rows and provenance rows already carried by the packet. Repack the NumFieldUp source comparison, the SplittingFieldUp row, the transported endpoint, and the Pkg provenance as the same public cyclotomic endpoint surface. □

Theorem 174.25 (Cyclotomic polynomial-ledger exactness)**.** *Every accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier exposes precisely the positive unary exponent row, the rational polynomial ledger for* $X^n - 1$*, the SplittingFieldUp root and factorization rows, the primitive-root endpoint, and the shared NumFieldUp/SplittingFieldUp provenance rows. Conversely, such rows, with the* hsame *base comparison and the displayed* Cont *acceptance ledger, repack as the public* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *source surface of Definition 174.17.*

Proof. For the forward direction, unfold the carrier of Definition 174.1. The tuple exposes the NumFieldUp source, positive unary exponent, rational polynomial row, SplittingFieldUp source, primitive-root endpoint, acceptance row, base comparison, and provenance row as BHist data.

Use Definition 174.16 to identify the number-field embedding and provenance surface. Use Definition 174.24 for the endpoint together with the root and factorization rows read inside the splitting-field ledger.

For the reverse direction, assume the displayed rows with the hsame base comparison, the Cont acceptance ledger, and the shared provenance. Pack them in the order prescribed by Definition 174.1; the obligation surface of Definition 174.17 then reads this packet as the public $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ source surface. □

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier polynomial ledger exactness`

Theorem 174.26 (Cyclotomic Iwasawa consumer boundary)**.** *An* $\mathsf{Iwasawa}^\uparrow$ *consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *may read only the NumFieldUp embedding ledger, the SplittingFieldUp factorization ledger, the primitive-root endpoint, the* Cont *acceptance row, and the* Pkg *provenance exposed by the cyclotomic certificate. It receives no p-adic measure row, class-group row, Galois-action row, quotient-root representative, or host-field equality datum from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *alone.*

Proof. Project the public rows listed by Definition 174.25. They supply the number-field embedding ledger, the polynomial and splitting-field rows, the primitive-root endpoint, the acceptance ledger, and the provenance rows.

Combine this projection with Definition 174.17. The obligation surface says that a downstream $\mathsf{Iwasawa}^\uparrow$ packet consumes exactly the rows displayed by the cyclotomic source certificate, so every permitted read is mediated by BHist, hsame, Cont, or Pkg data.

The excluded rows are not components of Definition 174.1 and are not produced by the classifier of Definition 174.2. Thus p-adic, class-group, Galois, quotient-root, and host-field equality data must come from separate certificates, not from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ itself. □

Theorem 174.27 (Cyclotomic primitive-root endpoint determinacy)**.** *For any two* $\mathsf{Cyclotomic}^\uparrow$ *carriers related by the classifier of Definition 174.2, the displayed primitive-root endpoints are the same up to the carried* hsame *row. The determining data are exactly the positive exponent row, the rational polynomial ledger for* $X^n - 1$*, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *source row over that ledger, and the finite* Cont *acceptance row that records the endpoint as a primitive root inside the splitting-field factorization surface.*

Proof. Project the classifier to the exponent component and the rational polynomial component. These two hsame rows identify the displayed $X^n - 1$ ledger as a carried history row, so the endpoint is not read from an ambient field but from the same polynomial surface.

Project next to the $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ source comparison. By Definition 174.15, transport along this source comparison preserves the root row and the factorization rows attached to the displayed polynomial ledger.

Finally read the primitive-root endpoint through the finite Cont acceptance row. Since the acceptance row is part of the classifier-related carrier data, the endpoint comparison is precisely the carried hsame comparison supplied for the endpoint, with no additional quotient of primitive roots or host-field equality. □

Theorem 174.28 (Cyclotomic factorization ledger exactness)**.** *For a* $\mathsf{Cyclotomic}^\uparrow$ *carrier, the accepted factorization rows are exactly the rows projected from the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *source for the displayed rational polynomial ledger* $X^n - 1$*. A classifier-related carrier has the same accepted factorization ledger up to the carried* hsame *rows on the NumFieldUp source, the positive exponent, the rational polynomial classifier, and the splitting-field source.*

Proof. Start with the displayed polynomial row in the carrier. The rational polynomial classifier fixes this row as the $X^n - 1$ ledger determined by the NumFieldUp source and the positive exponent, so any accepted factorization row must be read against that ledger.

Apply Definition 174.15 to the $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ source row. It preserves the root row and the factorization rows for the polynomial ledger classified with $X^n - 1$.

The reverse containment is just the carrier acceptance condition: the finite Cont row records exactly the factorization rows projected from the SplittingFieldUp source and lists no independent cyclotomic factorization surface. Combining the two projections gives exactness of the accepted ledger. □

Theorem 174.29 (Cyclotomic Iwasawa source boundary)**.** *An* $\mathsf{Iwasawa}^\uparrow$ *consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *may read only the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *source row, the positive exponent row, the primitive-root endpoint, the factorization ledger for* $X^n - 1$*, and the shared* Pkg *provenance package. In particular, the consumer receives no quotient of primitive roots, no host equality of fields, and no factorization row not already present in the displayed* BHist *ledger.*

Proof. Use Definition 174.17 to enumerate the public rows exposed by $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$. The carrier contributes the NumFieldUp source, the positive exponent, the polynomial ledger, the SplittingFieldUp source, the primitive-root endpoint, the Cont acceptance row, and the provenance package.

Use Definition 174.27 to restrict endpoint readback to the carried hsame endpoint comparison. Use Definition 174.28 to restrict the factorization readback to the SplittingFieldUp ledger for the displayed $X^n - 1$ polynomial.

After these projections, the remaining rows visible to an IwasawaUp packet are exactly the NumFieldUp source row, the positive exponent row, the primitive-root endpoint, the accepted factorization ledger, and the shared Pkg provenance package. The negative clauses follow because none of the projected rows contains a quotient principle, a host-field equality, or an additional factorization oracle. □

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier iwasawa source boundary`

Theorem 174.30 (Cyclotomic Iwasawa tower source readback)**.** *An* $\mathsf{Iwasawa}^\uparrow$ *tower consuming* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *may read exactly the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *source row, the positive exponent row, the primitive-root endpoint, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *factorization ledger for* $X^n - 1$*, the* Cont *acceptance row, and the shared* Pkg *provenance. These rows repack as the cyclotomic layer of the tower, and the readback carries no p-adic measure row, class-group row, or host-field equality datum.*

Proof. Project the carrier of Definition 174.1. Its displayed tuple supplies the NumFieldUp source, exponent, polynomial row, SplittingFieldUp source, primitive-root endpoint, Cont acceptance row, hsame base comparison, and shared provenance.

Apply Definition 174.17 and Definition 174.25. These identify the public cyclotomic source rows as the BHist and Cont entries already present in the carrier, with the factorization data read through the SplittingFieldUp ledger for $X^n - 1$.

Finally use Definition 174.29 to restrict the downstream tower surface. Repacking the projected rows gives the cyclotomic layer because every entry is mediated by BHist, hsame, Cont, or Pkg; the omitted analytic and class-group data are not entries of the cyclotomic certificate. □

Theorem 174.31 (Cyclotomic Galois-action boundary)**.** *Any Galois-action data visible from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *is limited to* BHist *endpoint transport of the primitive-root and factorization*

rows through the existing $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *provenance. The certificate exports no automorphism group, no quotient of roots, and no ambient field equality.*

Proof. Start with the classifier of Definition 174.2. It transports the NumFieldUp source, exponent, polynomial ledger, SplittingFieldUp source, and primitive-root endpoint by carried hsame rows supported by the displayed Cont and Pkg ledgers.

Use Definition 174.25 to read factorization evidence only from the SplittingFieldUp packet for $X^n - 1$. Use Definition 174.29 to keep downstream readback inside those projected source rows.

Thus the only action-like information exposed by the cyclotomic certificate is endpoint and factorization transport through existing BHist provenance. Since Definition 174.17 lists no automorphism group, quotient-root object, or host-field equality, none is exported. □

Theorem 174.32 (Cyclotomic p-adic completion input boundary)**.** *Any p-adic or* $\mathsf{Iwasawa}^{\uparrow}$ *completion consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *receives only the displayed cyclotomic source rows as input data: the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row, positive exponent row, primitive-root endpoint, SplittingFieldUp factorization ledger for* $X^n - 1$*,* Cont *acceptance row, and shared* Pkg *provenance. Valuation rows, completion rows, and measure rows must be supplied by separate* $\mathsf{Padic}^{\uparrow}$ *or* $\mathsf{Iwasawa}^{\uparrow}$ *certificates.*

Proof. Project the accepted carrier in Definition 174.1. The projected BHist entries and the Cont row are exactly the cyclotomic source data described by Definition 174.17.

Apply Definition 174.25 to identify the polynomial and factorization entries with the SplittingFieldUp ledger for $X^n - 1$. The classifier of Definition 174.2 transports these entries only by hsame comparisons, while the provenance remains the carried Pkg row.

The boundary theorem Definition 174.29 then excludes every row not displayed in that source surface. Therefore a completion consumer starts from the cyclotomic source rows alone and obtains valuation, completion, and measure data only from separate PadicUp or IwasawaUp certificates. □

Theorem 174.33 (Cyclotomic Galois-action NameCert boundary)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_{\zeta} = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, every Galois-action consumer row read from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *is a projection of the displayed* BHist *rows* $\mathcal{N}$*,* n*,* p*,* $\mathcal{S}$*, and* ζ*, the* Cont *row* α*, the* hsame *row* λ*, and the* Pkg *provenance row* ρ*. The allowed transport is exactly the endpoint and factorization transport supplied by Definition 174.2, Definition 174.25, Definition 174.27, Definition 174.28, and Definition 174.31; no automorphism-group object, quotient-root representative, or host-field equality is exported.*

Proof. Project the carrier to $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ. These are the only displayed rows in Definition 174.1, and the classifier of Definition 174.2 transports them by carried hsame support.

Use Definition 174.25 to identify the polynomial row and Definition 174.28 to restrict factorization evidence to the $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ ledger for $X^n - 1$. Use Definition 174.27 for the endpoint component.

The Galois-facing boundary of Definition 174.31 then repacks these projected rows as endpoint and factorization transport only. Since neither the carrier nor the classifier contains an automorphism group, quotient-root object, or host-field equality field, no such datum can be read by the downstream Galois consumer. □

Theorem 174.34 (Cyclotomic p-adic tower consumer exactness)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, a p-adic tower or $\mathsf{Iwasawa}^{\uparrow}$ *consumer receives exactly the projected source package consisting of* $\mathcal{N}$*,* n*,* ζ*, the factorization rows over* p *inside* $\mathcal{S}$*, the acceptance row* α*, and the provenance row* ρ*. This package is the common projection described by Definition 174.29 and Definition 174.32, with polynomial and factorization exactness supplied by Definition 174.25 and Definition 174.28.*

Proof. Project the carrier rows displayed in Definition 174.1. The source row $\mathcal{N}$, exponent row n, polynomial ledger p, splitting-field row $\mathcal{S}$, endpoint ζ, acceptance row α, comparison row λ, and provenance row ρ form the whole input surface.

Apply Definition 174.25 to read p as the rational polynomial ledger for $X^n - 1$. Apply Definition 174.28 to identify the accepted factorization rows as exactly the rows projected from $\mathcal{S}$ over that ledger.

Finally use Definition 174.29 and Definition 174.32. They remove valuation, completion, measure, and class-group rows from the cyclotomic source package, so the downstream tower receives precisely the displayed projection and must obtain any further p-adic structure from a separate certificate. □

Theorem 174.35 (Cyclotomic splitting-polynomial classifier transport)**.** *For two accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carriers*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

and

$$\mathcal{C}'_{\zeta'} = (\mathcal{N}', n', p', \mathcal{S}', \zeta', \alpha', \lambda', \rho')$$

related by the classifier of Definition 174.2, the rational polynomial row for $X^n - 1$ *transports to the rational polynomial row for* $X^{n'} - 1$ *together with the transported* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source and its root and factorization rows. The transport is the* BHist *and* hsame *transport asserted by Definition 174.25, Definition 174.27, and Definition 174.28, not an equality of ambient splitting fields.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier splitting polynomial classifier transport`

Proof. Project the classifier to the source, exponent, polynomial, splitting-field, and endpoint comparisons. These are exactly the comparisons listed in Definition 174.2, with endpoints supported by the two displayed carriers of Definition 174.1.

Use Definition 174.25 to transport the rational polynomial ledger along the source and exponent rows. The primitive-root endpoint remains attached to that transported ledger by Definition 174.27.

The splitting-field factorization rows then follow from Definition 174.28. Repacking the transported polynomial row, splitting-field row, endpoint row, Cont rows, and Pkg provenance gives the classifier transport claimed by a splitting-polynomial consumer, with no appeal to ambient field equality. □

Theorem 174.36 (Cyclotomic root-of-unity ledger coverage)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, every root-of-unity row visible to a downstream consumer is covered by the positive exponent row n, the rational polynomial ledger p for X^n-1, the $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *row $\mathcal{S}$, the primitive-root endpoint ζ, and the acceptance row α. The coverage is exact by Definition 174.25, Definition 174.27, and Definition 174.28; there is no additional root ledger outside the displayed* BHist *and* Cont *rows.*

Proof. Project the carrier to n, p, $\mathcal{S}$, ζ, and α. These rows are the only places where a root-of-unity consumer can read an exponent, a polynomial ledger, a splitting-field root surface, an endpoint, or an acceptance certificate.

Apply Definition 174.25 to identify the ledger as X^n-1. Apply Definition 174.27 to show that the primitive endpoint is determined by the displayed exponent, polynomial, splitting-field, and Cont rows.

Apply Definition 174.28 to cover all accepted factorization rows by the carried $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ source. The provenance rows λ and ρ record transport and origin, but they add no independent root-of-unity ledger, so the displayed projection is complete. □

Theorem 174.37 (Cyclotomic class-field source interface)**.** *For an accepted* $\mathsf{Cyclotomic}^\uparrow$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, a $\mathsf{ClassField}^\uparrow$ *source interface may cite exactly the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *source row $\mathcal{N}$, the exponent row n, the polynomial ledger p, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *row $\mathcal{S}$, the primitive-root endpoint ζ, the acceptance row α, the base comparison λ, and the provenance row ρ. This interface is obtained by repacking the source boundary of Definition 174.29, the completion-input boundary of Definition 174.32, and the classifier transport of Definition 174.2; it supplies no class group, reciprocity map, local symbol, or host-field equality.*

Proof. Start with the carrier projection from Definition 174.1. The NumFieldUp row, exponent row, polynomial ledger, SplittingFieldUp row, endpoint, acceptance row, base comparison, and provenance row are all displayed BHist, Cont, hsame, or Pkg data.

Use Definition 174.25 and Definition 174.28 to make the polynomial and factorization rows exact. Use Definition 174.27 to keep the endpoint readback tied to the same displayed rows.

Finally combine Definition 174.29 with Definition 174.32. These boundary statements identify the reusable cyclotomic source package while excluding rows that must come from other certificates, so a ClassFieldUp consumer may cite this package as source data but receives no class-field operations from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ alone. □

Theorem 174.38 (Cyclotomic class-field reciprocity input boundary)**.** *For an accepted* $\mathsf{Cyclotomic}^\uparrow$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, a $\mathsf{ClassField}^\uparrow$ *consumer that requires a reciprocity map may read from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *only the displayed rows $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ together with their* BHist, hsame, Cont, *and* Pkg *provenance. Any reciprocity map, Artin pairing, or class-field operation used by that consumer must be supplied by a separate* $\mathsf{NameCert}_{\mathsf{ClassField}^\uparrow}$ *certificate and cannot be inferred from the cyclotomic certificate alone.*

Proof. Project the accepted carrier. The only visible components are the NumFieldUp source, the positive exponent row, the polynomial ledger for X^n-1, the SplittingFieldUp source, the primitive endpoint, the acceptance row, the base comparison, and the provenance row.

Apply Definition 174.37 to repack those rows as the class-field source interface. Apply Definition 174.36 to identify the root-of-unity rows covered by the same exponent, polynomial, splitting-field, endpoint, and acceptance data.

Finally use Definition 174.43. It excludes reciprocity and class-group data from the cyclotomic export surface, so a consumer that needs an actual reciprocity map must obtain it from its own $\mathsf{ClassField}^{\uparrow}$ certificate. □

Theorem 174.39 (Cyclotomic local-symbol exclusion)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *exports no local symbol, valuation row, completion row, or residue-field row. A downstream local consumer may use the displayed carrier rows and their* BHist*,* hsame*,* Cont*, and* Pkg *provenance as input, but every local symbol, valuation, completion, or residue-field datum must enter through a separate* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{Iwasawa}^{\uparrow}$*, or local certificate.*

Proof. Project the carrier of Definition 174.1. Its components are exactly $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ, and none of those coordinates is a valuation, completion, residue field, or local-symbol operation.

Use Definition 174.32 to identify what a completion-facing consumer may read from the cyclotomic source. Use Definition 174.34 to keep the tower input at the same projected source package.

The absence of local-symbol coordinates in Definition 174.1 then gives the exclusion. Any such datum must be carried by a certificate whose own rows name the local object. □

Theorem 174.40 (Cyclotomic norm-residue boundary)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, a norm-residue consumer may read the $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row* $\mathcal{N}$*, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source row* $\mathcal{S}$*, the positive exponent row* n*, the polynomial ledger* p*, the primitive endpoint* ζ*, the factorization rows covered by* $\mathcal{S}$*, and the provenance rows* α*,* λ*, and* ρ*. It may not read a norm-residue map, Artin symbol, class group, or ambient-field quotient from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*.*

Proof. Begin with Definition 174.37. It gives the NumFieldUp source, exponent, polynomial ledger, SplittingFieldUp source, endpoint, acceptance, comparison, and provenance rows as the whole class-field source package exposed by the cyclotomic certificate.

Apply Definition 174.33 to keep Galois-facing reads inside endpoint and factorization transport. This accounts for the splitting-field factorization surface without introducing an Artin symbol or an ambient-field quotient.

Finally use Definition 174.43. Since norm-residue maps, class groups, and reciprocity data are outside the exported carrier, any norm-residue consumer must obtain those objects from a separate certificate while using $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ only for the displayed source and provenance rows. □

Theorem 174.41 (Cyclotomic positive exponent row coverage)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, the positive unary exponent row n *is the only exponent datum exported by* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*. Every downstream read of the polynomial* $X^n - 1$ *factors through the carried* BHist *row* n*, the polynomial ledger* p*, the* Cont *acceptance row* α*, and the* Pkg *provenance row* ρ*; no separate degree row, order row, or ambient exponent classifier is exported by the cyclotomic certificate.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier positive exponent row coverage`

Proof. Project the displayed carrier of Definition 174.1. The exponent component is the single row n, while p, α, and ρ record the polynomial ledger, acceptance evidence, and package provenance attached to that row.

Apply Definition 174.25. It identifies the polynomial ledger read by a consumer as the ledger for $X^n - 1$ and ties that readback to the same exponent row and acceptance data.

Since the carrier contains no second exponent field and no degree or order certificate, every use of $X^n - 1$ must pass through the projected row n together with p, α, and ρ. This gives the announced coverage and excludes any independent exponent export from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$. □

Theorem 174.42 (Cyclotomic primitive-order witness boundary)**.** *For accepted* $\mathsf{Cyclotomic}^\uparrow$ *carriers related by the classifier of Definition 174.2, the primitive-root information visible downstream is exactly the endpoint* ζ *together with the positive exponent row and the splitting-field factorization rows transported by* hsame*. The certificate exports this endpoint-and-ledger witness surface, not a minimal-order proof object or an ambient Galois-theoretic order computation.*

Proof. Use Definition 174.41 to isolate the exported exponent datum as the positive row n carried by the source certificate. Use Definition 174.36 to read the endpoint ζ together with the polynomial and splitting-field rows that cover the root-of-unity surface.

Next apply Definition 174.27. It makes the visible primitive witness the endpoint attached to the displayed exponent, polynomial ledger, splitting-field source, and acceptance row.

Finally apply Definition 174.35. The classifier transports the endpoint and factorization rows by the available hsame comparisons, so the downstream witness boundary is exactly the transported endpoint, exponent row, and splitting-field factorization surface, with no exported minimal-order or Galois-order datum. □

Theorem 174.43 (Cyclotomic minimal-polynomial nonexport)**.** *For an accepted* $\mathsf{Cyclotomic}^\uparrow$ *carrier,* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *does not export a minimal polynomial, Euler totient degree, automorphism group, reciprocity map, or class-group datum. Any downstream consumer requiring those data must obtain them from a* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{Padic}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{Iwasawa}^\uparrow}$*, or* $\mathsf{NameCert}_{\mathsf{ClassField}^\uparrow}$ *certificate rather than from the cyclotomic certificate alone.*

Proof. The positive-exponent surface is fixed by Definition 174.41. The primitive witness surface is fixed by Definition 174.42. Together they account for the exported exponent, endpoint, polynomial ledger, splitting-field rows, acceptance data, classifier transport, and provenance data.

Use Definition 174.33 to exclude an exported automorphism group or host-field equality package. Use Definition 174.34 to exclude valuation, completion, measure, and class-group rows from the cyclotomic source surface.

Finally apply Definition 174.37. It permits a class-field consumer to cite the cyclotomic source package, but it supplies no reciprocity map, local symbol, or class-group operation. Since none of the carrier rows named above is a minimal-polynomial, totient-degree, automorphism, reciprocity, or class-group field, those data must come from the corresponding separate certificates. □

Theorem 174.44 (Cyclotomic splitting-field provenance conservation)**.** *For accepted* $\mathsf{Cyclotomic}^\uparrow$ *carriers related by the classifier of Definition 174.2, every splitting-field consumer receives exactly the rational polynomial row* p *for* $X^n - 1$*, the primitive endpoint row* ζ*, the factorization rows carried by* $\mathcal{S}$*, the* Cont *acceptance row* α*, the transported* hsame *comparisons, and the displayed* Pkg *provenance row* ρ*. The transported surface is the same* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *source package*

described by Definition 174.35; no host-field equality, additional factorization ledger, or independent root carrier is exported by $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$.

Proof. Start from the classifier projections in Definition 174.2. They expose the source, exponent, polynomial, splitting-field, endpoint, acceptance, comparison, and provenance rows attached to the two accepted carriers.

Apply Definition 174.35. It transports the rational polynomial row, the primitive endpoint, and the $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ source through the available BHist and hsame comparisons.

Use Definition 174.28 to identify the factorization rows as exactly the rows projected from $\mathcal{S}$ over the transported polynomial ledger. Use Definition 174.25 to keep that ledger tied to $X^n - 1$ and to the same Cont and Pkg rows.

The carrier of Definition 174.1 has no coordinate for a host-field equality, a second factorization ledger, or a separate root carrier. Thus the splitting-field consumer surface is precisely the conserved polynomial, root, factorization, Cont, hsame, and Pkg family. □

Theorem 174.45 (Cyclotomic root-source provenance confluence)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier, the source package obtained by root-of-unity coverage and the package obtained by splitting-polynomial transport are the same displayed row family: the positive exponent row n, the rational polynomial row p for $X^n - 1$, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source $\mathcal{S}$, the primitive endpoint ζ, the acceptance row α, the* BHist *provenance, the* Cont *rows, and the* Pkg *row ρ. This confluence is internal to the cyclotomic certificate and supplies no additional minimal-order proof object or ambient field equality.*

Proof. Use Definition 174.36 to read the root-of-unity surface from the displayed exponent, polynomial ledger, splitting-field source, primitive endpoint, and acceptance row. This gives the source package seen by a root consumer.

Use Definition 174.42 to keep the primitive witness at the endpoint-and-ledger level. The theorem excludes a separate minimal-order proof object and keeps the visible witness attached to n, p, $\mathcal{S}$, ζ, and α.

Apply Definition 174.44. The splitting-polynomial route conserves exactly the same polynomial, endpoint, factorization, Cont, hsame, and Pkg rows.

Both routes are projections of the single carrier of Definition 174.1. Therefore the root-source package and the splitting-polynomial package are the same displayed BHist, Cont, and Pkg row family, with no further ambient equality claim. □

Theorem 174.46 (Cyclotomic class-field repack exactness)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier, a* $\mathsf{ClassField}^{\uparrow}$ *consumer repacks exactly the confluence family of Definition 174.45 together with the positive exponent row and the displayed provenance rows. The repacked interface consists of the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row $\mathcal{N}$, the positive exponent row n, the polynomial row p, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source $\mathcal{S}$, the primitive endpoint ζ, the acceptance row α, the base comparison λ, and the provenance row ρ; it receives no class group, reciprocity map, minimal polynomial, totient degree, or host-field equality from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$.

Proof. Begin with Definition 174.37. It displays the rows that a class-field source interface may cite from an accepted cyclotomic carrier: $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ.

Use Definition 174.45 to identify the root-of-unity route and the splitting-polynomial route with the same BHist, Cont, and Pkg row family. Use Definition 174.41 to make the exponent row n the sole exponent datum in that family.

Finally apply Definition 174.43. It excludes minimal polynomials, totient degrees, host-field equalities, class groups, and reciprocity maps from the cyclotomic export surface.

Thus the $\mathsf{ClassField}^{\uparrow}$ consumer receives exactly the displayed source, exponent, polynomial, splitting-field, endpoint, acceptance, comparison, and provenance rows, and every class-field operation must enter through a separate class-field certificate. □

Theorem 174.47 (CyclotomicUp unary exponent source closure)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

of Definition 174.1, every displayed positive exponent row used by the cyclotomic root carrier is the unary-history BHist *row* n*. This row is stable under the classifier of Definition 174.2 and under the* Cont *source ledger recorded by* α *and* λ*.*

Proof. Project the cyclotomic carrier. Its exponent coordinate is the single positive row n, and the polynomial row p is read as the ledger for $X^n - 1$.

Apply Definition 174.41. It identifies n as the only exponent datum exported by the certificate and ties every read of $X^n - 1$ to the same BHist row, polynomial ledger, continuation entry, and provenance row.

Finally use the classifier clauses for exponent and source comparison. They transport n through hsame and keep the same Cont source ledger attached to the accepted carrier, so every displayed exponent row remains a unary BHist source row. □

Theorem 174.48 (CyclotomicUp root-action package provenance)**.** *For accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carriers related by the classifier of Definition 174.2, every Galois-action or splitting-field consumer row for the primitive endpoint* ζ *factors through the same* Pkg *provenance row* ρ *as the root carrier, polynomial classifier, and splitting ledger. No action row receives an independent provenance package outside the carried* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *packet.*

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier root action pkg provenance`

Proof. Begin with Definition 174.44. It places the splitting-field consumer surface at the polynomial row p, the primitive endpoint ζ, the factorization rows of $\mathcal{S}$, the Cont acceptance row, the transported hsame comparisons, and the package row ρ.

Use Definition 174.33 to read the Galois-facing action surface from the same endpoint and factorization transport. The action row is therefore attached to the transported carrier surface rather than to a host-field equality package.

Finally apply Definition 174.45. The root route and splitting-polynomial route share the same BHist, Cont, and Pkg row family, so each action or splitting consumer factors through ρ. □

Theorem 174.49 (CyclotomicUp downstream NameCert obligation packet)**.** *The* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{Iwasawa}^{\uparrow}$*,* $\mathsf{ClassField}^{\uparrow}$*,* $\mathsf{GaloisGroup}^{\uparrow}$*,* $\mathsf{NumField}^{\uparrow}$*, and* $\mathsf{SplittingField}^{\uparrow}$ *consumers of an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carrier receive exactly the* BHist *root carrier, the unary exponent row, the polynomial classifier, the root-action provenance, and the ledger exactness rows listed in this chapter. They receive no valuation, completion, class group, reciprocity map, ambient automorphism quotient, minimal polynomial, or host-field equality from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*.*

Proof. Use Definition 174.47 to fix the exponent input as the single unary BHist row n. Use Definition 174.25 and Definition 174.28 to identify the polynomial and splitting ledgers attached to that row.

Apply Definition 174.48 for the Galois-action and splitting-field provenance surface. Apply Definition 174.46 for the class-field consumer and Definition 174.34 for the local tower consumers.

The remaining exclusions are supplied by Definition 174.43 and Definition 174.39. Thus each listed downstream NameCert consumes only the displayed root, exponent, polynomial, factorization, action-provenance, Cont, and Pkg rows. □

Definition 174.50 (Cyclotomic root downstream consumer surface)**.** *The downstream consumer surface of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *is the tuple of rows already present in Definition 174.1: the* BHist *entries* $\mathcal{N}$*,* n*,* p*,* $\mathcal{S}$*, and* ζ*, the* Cont *acceptance row* α*, the* hsame *comparison row* λ*, and the* Pkg *provenance row* ρ*. A consumer may address this tuple through the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *projections, the carrier/classifier rows of Definition 174.1 and Definition 174.2, and the certificate surface of Definition 174.17. No host field equality, quotient of roots, valuation row, class-group row, or class-field operation is part of this surface.*

Theorem 174.51 (Cyclotomic root carrier classifier scope)**.** *For every accepted* $\mathsf{Cyclotomic}^\uparrow$ *carrier, the classifier scope available to downstream consumers is exactly the componentwise* hsame *transport on the* BHist *rows of Definition 174.50, together with the displayed* Cont *and* Pkg *ledgers. In particular, the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *projections remain the only source certificates visible to an* $\mathsf{Iwasawa}^\uparrow$ *or* $\mathsf{ClassField}^\uparrow$ *consumer.*

Proof. Project the carrier of Definition 174.1. The visible rows are precisely $\mathcal{N}$, n, p, $\mathcal{S}$, ζ, α, λ, and ρ.

Apply Definition 174.2 and Definition 174.5. They transport the source, exponent, polynomial, splitting-field, and endpoint rows by hsame and keep the endpoint acceptance and provenance attached by Cont and Pkg.

The two projected source certificates are therefore the only certificates available at the root boundary. Since no other coordinate appears in the carrier or classifier, downstream consumers receive no host equality or quotient-root datum. □

Theorem 174.52 (Cyclotomic root ledger stability package)**.** *The ledger package on the cyclotomic root surface is stable under classifier transport: the rational polynomial ledger for* $X^n - 1$*, the primitive endpoint readback, the splitting-field root and factorization rows, and the* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *provenance remain the same public rows after transport through* hsame*,* Cont*, and* Pkg *support.*

Proof. Use Definition 174.25 for the polynomial ledger and Definition 174.18 for the endpoint readback. These identify the public root rows as projections of the same accepted carrier.

Next apply Definition 174.15 and Definition 174.16. The first transports the splitting-field root and factorization rows, while the second preserves the NumFieldUp provenance ledger.

Repack those projections into the surface of Definition 174.50. The repacking uses only the listed carrier and certificate rows, so it adds no host equality, valuation, or class-field row. □

Theorem 174.53 (Cyclotomic root downstream unblock package)**.** *The pair consisting of Definition 174.51 and Definition 174.52 supplies the complete root obligation surface needed by downstream* $\mathsf{Iwasawa}^\uparrow$ *and* $\mathsf{ClassField}^\uparrow$ *fronts: a* BHist *carrier tuple,* hsame *classifier transport, the* Cont *acceptance ledger, the* Pkg *provenance row, and the two source* NameCert *projections for* $\mathsf{NumField}^\uparrow$ *and* $\mathsf{SplittingField}^\uparrow$*.*

Proof. Begin with the carrier/classifier scope theorem. It fixes the exported rows and the only allowed transport mechanism for each projected root component.

Use the ledger stability package to carry the polynomial, primitive endpoint, splitting-field, factorization, and NumFieldUp provenance rows through that transport. These rows are exactly the rows listed in Definition 174.50.

Therefore an $\mathsf{Iwasawa}^{\uparrow}$ or $\mathsf{ClassField}^{\uparrow}$ consumer can cite the cyclotomic root surface as a stable source package. The cited package still contains no tower transition map, class group, reciprocity map, local symbol, completion row, quotient root, or equality of ambient fields. □

Theorem 174.54 (Cyclotomic root Iwasawa and ClassField consumer boundary)**.** *When* $\mathsf{Iwasawa}^{\uparrow}$ *or* $\mathsf{ClassField}^{\uparrow}$ *consumes* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*, the consumed data are exactly the root downstream unblock package of Definition 174.53.* $\mathsf{Iwasawa}^{\uparrow}$ *may read the cyclotomic layer as source data for later tower rows, and* $\mathsf{ClassField}^{\uparrow}$ *may read the same layer as source data for later reciprocity rows, but neither consumer receives tower maps, class groups, reciprocity maps, local symbols, completion data, quotient-root carriers, or host-field equality from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *alone.*

Proof. For the Iwasawa side, apply Definition 174.29 and Definition 174.30. They read the cyclotomic carrier as a layer source while keeping tower transition and measure rows outside the cyclotomic packet.

For the class-field side, apply Definition 174.37. It reads the same BHist, hsame, Cont, Pkg, and NameCert rows as source data and leaves class-field operations to their own certificate.

Both projections factor through Definition 174.53. Hence the downstream boundary is shared and exact, with no extra root quotient or ambient-field equality introduced by either consumer. □

Definition 174.55 (Cyclotomic root obligation inventory)**.** *The root obligation inventory of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *is the finite packet consisting of the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source row, the positive unary exponent row, the primitive-root classifier row, the rational polynomial ledger for* $X^n - 1$*, the finite* Cont *readback, the* hsame *transport row, the* Pkg *provenance row, the exactness row identifying the displayed root surface, and the downstream consumer rows for* $\mathsf{Iwasawa}^{\uparrow}$*,* $\mathsf{ClassField}^{\uparrow}$*, and* $\mathsf{Regulator}^{\uparrow}$*. Each component is a displayed* BHist*,* Cont*,* hsame*, or* Pkg *row of the cyclotomic carrier and its source certificates.*

Theorem 174.56 (Cyclotomic root carrier classifier obligations)**.** *The carrier and classifier obligations of the cyclotomic root inventory are exactly the displayed* BHist *rows of Definition 174.55: the NumFieldUp source, the SplittingFieldUp source, the positive unary exponent, the primitive-root classifier, and the rational polynomial ledger, together with their finite* Cont*,* hsame*, and* Pkg *support rows.*

Proof. Project the carrier of Definition 174.1. Its visible entries are the source field, exponent, polynomial, splitting-field, root, acceptance, transport, and provenance rows.

Apply Definition 174.2 to identify the polynomial ledger and primitive-root classifier rows. Apply Definition 174.51 to see that the classifier boundary is only the componentwise hsame transport on those displayed rows, with Cont and Pkg kept as ledger support.

Thus every carrier and classifier obligation is one of the displayed rows of the inventory, and no ambient cyclotomic-field carrier is required. □

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootClassifier downstream consumer transport`

Theorem 174.57 (Cyclotomic root ledger exactness obligation)**.** *The ledger of Definition 174.55 is exact: after reading the polynomial, primitive endpoint, splitting-field, transport, and provenance rows, the cyclotomic root packet contains no host cyclotomic field, no quotient of roots, and no object beyond the displayed finite rows.*

Proof. Use Definition 174.25 for the rational polynomial ledger and Definition 174.18 for the endpoint readback. These two results identify the polynomial and primitive-root data as finite projections of the accepted carrier.

Next apply Definition 174.52. It carries the splitting-field, factorization, endpoint, transport, and NumFieldUp provenance rows through the same hsame, Cont, and Pkg support.

The resulting packet is precisely the inventory. Since each coordinate has already been projected from the displayed carrier, there is no remaining coordinate in which a host cyclotomic field or quotient root could be exported. □

Theorem 174.58 (Cyclotomic root downstream consumer obligations)**.** *Downstream* Iwasawa$^{\uparrow}$, ClassField$^{\uparrow}$, *and* Regulator$^{\uparrow}$ *readers consume only the packet boundary listed in Definition 174.55: source certificates, positive exponent and primitive-root classifier rows, polynomial ledger, finite* Cont *readback,* hsame *transport,* Pkg *provenance, and the exactness row of the cyclotomic root surface.*

Proof. For the Iwasawa and ClassField readers, apply Definition 174.54. It factors both consumers through the cyclotomic root downstream package and excludes tower maps, reciprocity maps, class groups, local symbols, completion data, quotient roots, and host-field equality.

For the Regulator reader, use the same packet boundary: the reader can address the NumFieldUp source, the polynomial ledger, and the finite root readback only through the displayed BHist, Cont, hsame, and Pkg rows. These rows are exactly the rows named by the inventory.

Therefore every downstream consumer cited here reads the same finite packet boundary, and none receives a host algebra primitive outside that packet. □

Theorem 174.59 (Cyclotomic root obligation threshold)**.** *The accepted* Cyclotomic$^{\uparrow}$ *root boundary supplies the threshold obligation surface consumed by* Iwasawa$^{\uparrow}$, ClassField$^{\uparrow}$, *and* Regulator$^{\uparrow}$*: the root carrier of Definition 174.1, the rational-polynomial classifier transport of Definition 174.5, the polynomial ledger of Definition 174.25, the primitive-root readback of Definition 174.18, and the finite* Cont, hsame, *and* Pkg *rows listed by Definition 174.55. It exports that finite root surface without adding a host cyclotomic field, quotient root, class group, tower map, or reciprocity map.*

Proof. Start from Definition 174.55. It names the NumFieldUp and SplittingFieldUp source rows, the positive exponent, the primitive-root classifier, the rational polynomial ledger for $X^n - 1$, finite readback, transport, provenance, and downstream consumer rows.

Apply Definition 174.56. The carrier and classifier clauses are exactly those displayed BHist rows with their Cont, hsame, and Pkg support.

Next use Definition 174.57. It identifies the polynomial, primitive endpoint, splitting-field, transport, and provenance rows as an exact finite ledger, so no host cyclotomic field or quotient-root coordinate is available.

Finally apply Definition 174.58. The three downstream fronts consume only the inventory boundary. Repacking these three cited obligations gives the stated threshold surface. □

Theorem 174.60 (Cyclotomic root classifier readback threshold)**.** *Every downstream read of the* Cyclotomic$^{\uparrow}$ *root classifier by* Iwasawa$^{\uparrow}$, ClassField$^{\uparrow}$, *or* Regulator$^{\uparrow}$ *factors through the same displayed* BHist *root row, the rational-polynomial classifier row, primitive readback,* Cont *acceptance,* hsame *transport, and* Pkg *provenance of Definition 174.55. Conversely, these rows repack into the root classifier surface of Definition 174.59.*

Proof. For the forward projection, read a downstream classifier access through Definition 174.54 for Iwasawa$^{\uparrow}$ and ClassField$^{\uparrow}$, and through Definition 174.58 for the shared Regulator-facing root surface.

The projected rows are then controlled by Definition 174.51. That theorem leaves only componentwise hsame transport on the displayed root rows, with Cont and Pkg kept as finite support.

For the converse direction, take exactly the displayed root carrier, classifier, primitive readback, transport, acceptance, and provenance rows. The threshold theorem repacks them as the accepted root obligation surface, so the downstream readback closes without a host root quotient or ambient field equality. □

Lean4 ✓ `BEDC.Derived.CyclotomicUp.CyclotomicRootClassifier readback threshold`

Theorem 174.61 (Cyclotomic root ledger exactness threshold)**.** *The* $\mathsf{Cyclotomic}^{\uparrow}$ *root ledger exported at the threshold surface is exact: once the root carrier, rational-polynomial classifier transport, polynomial ledger, primitive-root readback, finite* Cont*, componentwise* hsame*, and* Pkg *rows are fixed, every accepted consumer row for* $\mathsf{Iwasawa}^{\uparrow}$*,* $\mathsf{ClassField}^{\uparrow}$*, or* $\mathsf{Regulator}^{\uparrow}$ *reads the same finite root boundary and no additional host algebra primitive.*

Proof. Use Definition 174.52. The polynomial ledger, primitive endpoint readback, splitting-field root and factorization rows, and NumFieldUp provenance remain stable under the same hsame, Cont, and Pkg support.

Combine this with Definition 174.57. The exactness obligation says that after those rows are read, the packet has no remaining coordinate for a host cyclotomic field, root quotient, valuation row, or class-field operation.

It remains to identify the consumers. Apply Definition 174.59 and Definition 174.60; together they show that $\mathsf{Iwasawa}^{\uparrow}$, $\mathsf{ClassField}^{\uparrow}$, and $\mathsf{Regulator}^{\uparrow}$ all factor through the same finite root boundary. □

Theorem 174.62 (Cyclotomic source triad readback completeness)**.** *For an accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *root packet, every downstream readback of the source triad is complete from the displayed* BHist *rows: the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *row, the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *row over the rational polynomial ledger for* $X^n - 1$*, and the primitive endpoint row. The readback uses only the carried* hsame *comparisons, finite* Cont *acceptance,* Pkg *provenance, and the* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *classifier rows listed by Definition 174.10.*

Proof. Start with the source triad of Definition 174.10. It names the NumFieldUp source, SplittingFieldUp source, and primitive endpoint as the entire cyclotomic source family over the carrier of Definition 174.1.

Use Definition 174.23 to identify every downstream source read with the same displayed BHist, Cont, hsame, and Pkg support. Then use Definition 174.25 for the polynomial ledger and Definition 174.24 for the primitive endpoint.

These projections repack the whole $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ source triad. There is no fourth source row, because the cited obligation surface has no coordinate outside the displayed carrier, classifier, acceptance, and provenance rows. □

Theorem 174.63 (Cyclotomic downstream valuation exclusion)**.** *At the* $\mathsf{Cyclotomic}^{\uparrow}$ *root threshold, a downstream* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{Iwasawa}^{\uparrow}$*, or* $\mathsf{ClassField}^{\uparrow}$ *consumer may cite the* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *source triad only through the displayed* BHist *carrier,* hsame *transport, finite* Cont *acceptance,* Pkg *provenance, and* $\mathsf{NameCert}$ *rows. The threshold exports no valuation row, completion row, measure row, reciprocity map, local symbol, or host field equality.*

Proof. Read the downstream interface through Definition 174.32 and Definition 174.37. Both boundaries allow the cyclotomic packet to serve only as source data for later certificates.

Apply Definition 174.25. The polynomial ledger is the $X^n - 1$ row attached to the same SplittingFieldUp source, not a valuation or completion datum. Apply Definition 174.24 to keep primitive endpoint transport inside the carried hsame and Cont rows.

Finally use Definition 174.23. Since the downstream source surface is exhausted by the displayed BHist, Cont, hsame, Pkg, and NameCert rows, all valuation, completion, measure, reciprocity, local-symbol, and host-equality data must come from another certificate. □

Theorem 174.64 (Cyclotomic root-layer consumer threshold)**.** *The root-layer consumer threshold for* $\mathsf{Cyclotomic}^{\uparrow}$ *is exactly the source triad readback, polynomial-ledger exactness, and primitive endpoint transport surface over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Once these rows are fixed,* $\mathsf{Iwasawa}^{\uparrow}$*,* $\mathsf{Padic}^{\uparrow}$*, and* $\mathsf{ClassField}^{\uparrow}$ *consumers have all cyclotomic source data they may read, and no analytic, quotient, valuation, or host-equality row is supplied by the cyclotomic certificate.*

Proof. First collect the complete source triad from Definition 174.62. This supplies the NumFieldUp source, the SplittingFieldUp source, and the primitive endpoint over the displayed carrier and classifier support.

Next attach the ledger and endpoint controls. The polynomial row is exact by Definition 174.25, and primitive endpoint transport is exactly the transported endpoint surface of Definition 174.24.

The consumer side is then bounded by Definition 174.59 and Definition 174.63. Together they show that the listed source rows are sufficient for the root-layer consumers and that the excluded analytic, quotient, valuation, and host-equality rows remain outside the $\mathsf{Cyclotomic}^{\uparrow}$ certificate. □

Definition 174.65 (Cyclotomic root obligation packet)**.** *The* $\mathsf{Cyclotomic}^{\uparrow}$ *root obligation packet is the finite row family*

$$(\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho),$$

where $\mathcal{N}$ *is the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source row,* n *is the positive unary* BHist *exponent,* p *is the rational polynomial ledger for* $X^n - 1$*,* $\mathcal{S}$ *is the* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$ *source over* p*,* ζ *is the primitive-root endpoint,* α *is the finite* Cont *acceptance and factorization ledger,* λ *is the transported* hsame *comparison between source base rows, and* ρ *is the* Pkg *provenance for the paired source certificates.*

The packet is a row package for downstream consumers. It is not a host cyclotomic field, not a quotient of roots, and not a class-field, valuation, or completion object.

Theorem 174.66 (Cyclotomic root source row exhaustion)**.** *Every source row of the* $\mathsf{Cyclotomic}^{\uparrow}$ *root boundary is exhausted by the root obligation packet of Definition 174.65. Conversely, the displayed packet repacks into the root carrier, classifier transport, polynomial ledger, primitive endpoint, and threshold surface of the accepted cyclotomic root interface.*

Proof. Start with the carrier of Definition 174.1. Its visible coordinates are exactly the NumFieldUp source, exponent, polynomial, SplittingFieldUp source, primitive endpoint, acceptance, comparison, and provenance rows displayed in Definition 174.65.

Apply Definition 174.5. It transports those carrier rows componentwise through hsame, while keeping the Cont acceptance and Pkg provenance attached to the same finite packet.

Use Definition 174.25 for the polynomial row and Definition 174.24 for the primitive endpoint. These theorems account for the source ledger and endpoint readback without adding a quotient root or ambient field equality.

Finally apply Definition 174.59. It repacks the carrier, classifier transport, exact ledger, endpoint, Cont, hsame, and Pkg rows as the threshold surface. Since no other source coordinate occurs in the carrier or threshold theorem, the packet is exhaustive. □

Theorem 174.67 (Cyclotomic primitive-root ledger transport)**.** *For accepted* $\mathsf{Cyclotomic}^{\uparrow}$ *carriers related by the rational-polynomial classifier, the primitive-root endpoint, its polynomial ledger for* $X^n - 1$*, the SplittingFieldUp factorization rows, the* Cont *acceptance row, and the* Pkg *provenance*

are transported as the same root obligation packet. No transported row becomes a host-field equality, quotient of primitive roots, or independent minimal-order datum.

Proof. Project the related carriers through Definition 174.2. The classifier exposes only the displayed source, exponent, polynomial, splitting-field, endpoint, acceptance, comparison, and provenance rows.

Use Definition 174.5 to move those rows componentwise through the available hsame comparisons. The finite Cont and Pkg rows remain attached to the transported endpoint and source certificates.

Apply Definition 174.25 and Definition 174.24. The polynomial ledger and primitive endpoint are therefore the transported rows of the same packet, not new algebraic objects.

Repack the transported family by Definition 174.65 and the threshold theorem Definition 174.59. The exclusions follow from the carrier definition and the threshold surface, both of which name no host field equality, root quotient, or separate order witness. □

Theorem 174.68 (Cyclotomic SplittingField consumer boundary)**.** *A downstream* $\mathsf{SplittingField}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *may read exactly the splitting-field source row* $\mathcal{S}$*, the polynomial row* p*, the primitive endpoint* ζ*, the factorization and acceptance row* α*, the transported* hsame *comparisons, and the* Pkg *provenance contained in the root obligation packet. It receives no extra splitting field object, host equality of fields, valuation row, class-field operation, or completion datum from* $\mathsf{Cyclotomic}^{\uparrow}$ *alone.*

Proof. Begin with the SplittingField-facing coordinates of Definition 174.65: $\mathcal{S}$, p, ζ, α, λ, and ρ. These are the only rows of the packet addressed to a splitting-field consumer.

Apply Definition 174.35. It transports the polynomial and splitting-field source rows through the displayed classifier support. Then use Definition 174.28 to identify the factorization rows as the ledger covered by $\mathcal{S}$.

The primitive endpoint is controlled by Definition 174.24, and the root threshold is fixed by Definition 174.59. These results repack exactly the displayed splitting-field consumer rows.

No remaining coordinate of Definition 174.65 names a second splitting field, ambient field equality, valuation, class-field operation, or completion object. Therefore a SplittingFieldUp consumer is bounded by the listed packet rows. □

Definition 174.69 (Cyclotomic scoped root obligation package)**.** *The scoped root obligation package for* $\mathsf{Cyclotomic}^{\uparrow}$ *is the finite BEDC row family*

$$\mathcal{R}_{\zeta} = (\mathcal{C}_{\zeta}, \Theta_{\mathbb{Q}[X]}, \zeta, \alpha, \lambda, \rho, \nu, \sigma),$$

where $\mathcal{C}_{\zeta}$ *is the root carrier of Definition 174.1,* $\Theta_{\mathbb{Q}[X]}$ *is the rational-polynomial classifier of Definition 174.2,* ζ *is the primitive endpoint,* α *is the finite* Cont *root and factorization ledger,* λ *is the displayed* hsame *base-row comparison,* ρ *is the* Pkg *provenance row for* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$*,* ν *is the finite row family consumed by* $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Regulator}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{SplittingField}^{\uparrow}}$*, and* σ *is the displayed* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *row obtained by repacking these components.*

The package is scoped to the rows named above. It is not a host cyclotomic field, quotient of primitive roots, reciprocity map, regulator determinant, valuation, completion, or ambient splitting-field equality.

Theorem 174.70 (Cyclotomic root BHist scope binding)**.** *Every accepted scoped root obligation package* $\mathcal{R}_{\zeta}$ *of Definition 174.69 binds its carrier, rational-polynomial classifier, primitive endpoint,*

Cont *ledger,* hsame *base comparison,* Pkg *provenance, and source* $\mathsf{NameCert}$ *rows to displayed* BHist *primitives. Conversely, exactly those displayed rows repack as the scoped package.*

Proof. Project the carrier coordinate $\mathcal{C}_\zeta$. By Definition 174.1, its visible components are the $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ source row, the positive unary exponent, the rational polynomial ledger for $X^n - 1$, the $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ source row, the primitive endpoint, the finite acceptance row, the base comparison row, and the provenance row.

Apply Definition 174.66. It identifies this same row family as the root obligation packet and shows that no other source coordinate is available at the root boundary.

Next use Definition 174.67 together with Definition 174.59. These theorems bind the primitive endpoint, polynomial ledger, factorization rows, Cont ledger, hsame transport, and Pkg provenance to the displayed BHist carrier surface.

For the converse direction, start with exactly the displayed carrier, classifier, endpoint, continuation, sameness, provenance, and source-certificate rows. The root carrier definition and the root obligation packet definition repack them as $\mathcal{C}_\zeta$ and then as $\mathcal{R}_\zeta$. No host field equality or root quotient is introduced in the repacking step. □

Theorem 174.71 (Cyclotomic root consumer scope exhaustion)**.** *A downstream* $\mathsf{Iwasawa}^\uparrow$, $\mathsf{ClassField}^\uparrow$, $\mathsf{Regulator}^\uparrow$, *or* $\mathsf{SplittingField}^\uparrow$ *reader of* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *receives exactly the rows of the scoped root obligation package: the root carrier, the rational-polynomial classifier, the primitive endpoint, the finite* Cont *ledger, the displayed* hsame *comparisons, the* Pkg *provenance, and the source* $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$ *rows. No reader receives a class group, reciprocity map, regulator determinant, valuation, completion, quotient-root object, or host splitting-field equality from this package alone.*

Proof. First consider the splitting-field reader. Apply Definition 174.68. It restricts that consumer to the splitting-field source row, polynomial row, primitive endpoint, factorization and acceptance rows, transported hsame comparisons, and Pkg provenance contained in the root obligation packet.

For $\mathsf{Iwasawa}^\uparrow$ and $\mathsf{ClassField}^\uparrow$, use Definition 174.54 and Definition 174.37. These statements permit the cyclotomic source package to be cited as input rows while excluding tower maps, class groups, reciprocity maps, local symbols, completion data, and host-field equality.

For the regulator-facing reader, use the same source-row exhaustion supplied by Definition 174.66 and the nonexport boundary of Definition 174.43. Since no coordinate of the carrier or classifier is a determinant, norm form, or regulator ledger, the reader receives only the displayed root source rows and provenance.

The four cases exhaust the consumer row family ν in Definition 174.69. Repacking the cases gives exactly the listed BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ rows, and no additional algebraic object is exported. □

Theorem 174.72 (Cyclotomic root scoped closure boundary)**.** *The package of Definition 174.69, the scope binding theorem Definition 174.70, and the consumer exhaustion theorem Definition 174.71 close the scoped root-obligation boundary of* $\mathsf{Cyclotomic}^\uparrow$ *over* BHist, Cont, hsame, Pkg, $\mathsf{NameCert}_{\mathsf{NumField}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{SplittingField}^\uparrow}$. *The boundary does not export a host cyclotomic field, primitive-root quotient, Galois bridge, class-field bridge, regulator bridge, or standard splitting-field model.*

Proof. The definition supplies the finite package: carrier, classifier, endpoint, continuation, sameness, provenance, source certificates, consumer rows, and the repacked $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ row.

Apply Definition 174.70. It ties every package coordinate to displayed BEDC primitives and gives the converse repacking from those primitives.

Apply Definition 174.71. It verifies that the named downstream readers can consume only the same finite row family and cannot read the excluded host or bridge objects from the cyclotomic certificate.

Therefore the scoped boundary is exactly the displayed root-obligation surface. Any public cyclotomic field, quotient of roots, reciprocity structure, regulator structure, or standard model must be supplied by a separate certificate or bridge. □

Theorem 174.73 (Cyclotomic Galois root dependency link)**.** *Let*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$$

be an accepted $\mathsf{Cyclotomic}^\uparrow$ *carrier. Any Galois-extension dependency read that exports this root-of-unity packet as a cyclotomic* $\mathsf{NameCert}$ *must consume the* $\mathsf{FieldExt}^\uparrow$ *and* $\mathsf{SplittingField}^\uparrow$ *sibling carriers through the polynomial-root and factorization rows* $p, \mathcal{S}, \zeta, \alpha$*, and must consume the* $\mathsf{GaloisExt}^\uparrow$ *and* $\mathsf{GaloisGroup}^\uparrow$ *sibling carriers through the automorphism-action rows supplied by chapter 163 and chapter 164. The cyclotomic side contributes the endpoint and factorization surface of Definition 174.1, while the Galois side contributes normality, separability, base-fixing, and automorphism-action rows before the local* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *export is read.*

Thus the dependency lattice for a Galois-facing cyclotomic read is

$$\mathsf{FieldExt}^\uparrow,\ \mathsf{SplittingField}^\uparrow,\ \mathsf{GaloisExt}^\uparrow,\ \mathsf{GaloisGroup}^\uparrow,\ \mathsf{Cyclotomic}^\uparrow,$$

with the root data routed through BHist, hsame, Cont, *and* Pkg *rows. The route exports no quotient of roots, no host field equality, no ambient automorphism set, and no class-field or Iwasawa datum from the cyclotomic certificate alone.*

Proof. Project the accepted carrier by Definition 174.1. The cyclotomic rows available before naming are exactly the NumField source, exponent, polynomial ledger, SplittingField source, primitive-root endpoint, root-acceptance continuation, base comparison, and package provenance.

Use Definition 174.25, Definition 174.27, and Definition 174.28. These identify the polynomial-root and factorization rows as the only cyclotomic data that can be offered to a Galois-extension consumer.

The automorphism information is then read from the sibling Galois certificates, not manufactured by the cyclotomic carrier. The Galois-extension chapter supplies the normal and separable extension surface over $\mathsf{FieldExt}^\uparrow$, while the Galois-group chapter supplies the base-fixing automorphism-action rows over that extension.

Finally apply Definition 174.31 and Definition 174.33. They keep every Galois-facing cyclotomic read inside endpoint and factorization transport, so the combined lattice can export a cyclotomic NameCert only after the FieldExt, SplittingField, GaloisExt, and GaloisGroup sibling rows have supplied their own dependency data. □

Definition 174.74 (Cyclotomic Iwasawa classgroup consumer packet)**.** *The Cyclotomic Iwasawa classgroup consumer packet is the finite row family read by an* $\mathsf{Iwasawa}^\uparrow$ *classgroup consumer from an accepted* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^\uparrow}$ *source. It consists of the cyclotomic layer rows* $\mathcal{N}, n, p, \mathcal{S}, \zeta$*, the primitive-root endpoint row* ζ*, the factorization ledger rows of* $\mathcal{S}$ *over the polynomial ledger* p*, the classifier transport supplied by the cyclotomic rational-polynomial classifier, the finite* Cont *acceptance and factorization provenance row* α*, and the shared* Pkg *provenance row* ρ*.*

The packet is only a BEDC consumer surface. It is not a host cyclotomic field, not a class-field bridge, not analytic Iwasawa data, and not a quotient carrier.

Theorem 174.75 (Cyclotomic Iwasawa classgroup consumer projection)**.** *For every accepted* Cyclotomic$^{\uparrow}$ *source carrier*

$$\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho),$$

the Iwasawa$^{\uparrow}$ *classgroup consumer rows projected from that carrier are exactly the rows of Definition 174.74. The projection uses the cyclotomic source boundary, the primitive-root endpoint rows, the factorization ledger rows, the classifier transport,* Cont*, and* Pkg *provenance already present in the source carrier.*

Proof. Start with the cyclotomic source boundary of Definition 174.29. It projects the NumField source row, the positive exponent row, the primitive-root endpoint, the factorization ledger for $X^n - 1$, and the shared package provenance to the Iwasawa-facing surface.

Next use the root obligation closure rows. The packet of Definition 174.65 carries the polynomial ledger, the SplittingField source, the primitive endpoint, the finite Cont acceptance row, the transported hsame comparison, and the Pkg provenance row. The transport theorem Definition 174.67 keeps these rows tied to the same source carrier.

Repack the projected rows by Definition 174.74. None of the cited source-boundary or root-obligation rows contains a host cyclotomic field, class-field bridge, analytic Iwasawa datum, or quotient carrier, so none is exported by the projection. □

Theorem 174.76 (Cyclotomic Iwasawa classgroup consumer exhaustion)**.** *Every Iwasawa-facing cyclotomic read used by a classgroup consumer is one of the rows of Definition 174.74, and every such row repacks into the same consumer packet. The surface has no host cyclotomic-field carrier, class-field bridge, analytic Iwasawa row, or quotient carrier.*

Proof. Project any Iwasawa-facing read through Definition 174.66. The source row must be one of the root carrier, classifier transport, polynomial ledger, primitive endpoint, threshold, Cont, hsame, or Pkg rows of the root obligation packet.

Now restrict to the classgroup consumer view using Definition 174.54. The Iwasawa side receives only the source, exponent, primitive endpoint, factorization, and provenance rows needed for its finite classgroup-facing input; class-field operations remain outside the cyclotomic packet.

Conversely, each row listed in Definition 174.74 is a coordinate of the cyclotomic source boundary or of the root obligation closure package. Therefore the rows repack into the same consumer packet, and the exclusions follow because neither cited boundary contains a host field, analytic Iwasawa object, class-field bridge, or quotient carrier. □

The Cyclotomic$^{\uparrow}$ root layer consumes the number-field source of chapter 168 and the root-factorization source of chapter 165 before exposing its primitive-root endpoint.

Closure status for NameCert$_{\text{Cyclotomic}^{\uparrow}}$**.**

Constructive story: A cyclotomic root layer is generated as the finite packet $\mathcal{C}_\zeta = (\mathcal{N}, n, p, \mathcal{S}, \zeta, \alpha, \lambda, \rho)$: a NumField$^{\uparrow}$ source row, a positive unary exponent, the rational polynomial ledger for $X^n - 1$, a SplittingField$^{\uparrow}$ source over that ledger, a primitive-root endpoint, an acceptance and factorization Cont row, a base hsame comparison, and Pkg provenance. The carrier accepts exactly that displayed tuple and excludes quotient roots, host field equality, valuation rows, class-field reciprocity data, automorphism groups, completions, and regulator rows. Two accepted packets classify together when the NumField source, exponent, polynomial ledger, splitting-field source, and primitive endpoint are transported by componentwise hsame with support from the displayed acceptance and provenance rows. The source specification is the same root-layer tuple, the pattern records source coverage, polynomial-ledger exactness, primitive endpoint determinacy, factorization coverage, and downstream consumer exhaustion, and the ledger keeps the exponent, polynomial, root, factorization, base comparison, and provenance rows visible as the same packet. The five naming-certificate obligations are supplied by the accepted root tuple, reflexive, symmetric, and transitive component transports, and the root-layer classifier transport theorem that repacks transported rows as another accepted carrier. A history h lies in the signature gap of a Cyclotomic$^{\uparrow}$ token p when the source domain accepts the root-layer packet, the signature relation pairs it with an

answer packet and evidence ledger, and token introduction names that answer as p; here that means the positive exponent, polynomial $X^n - 1$ ledger, splitting-field root rows, primitive endpoint, and provenance of h are the rows named by p. The chapter registers this scoped root-obligation construction under Cyclotomic$^{\uparrow}$; it gives downstream number-theoretic consumers a finite source package, but no public cyclotomic-field quotient, Galois bridge, class-field bridge, or standard splitting-field model.

Theory closure: TheoryGate(NameCert$_{\text{Cyclotomic}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 174.72 binds the scoped root package to the displayed BHist, hsame, Cont, NameCert, and Pkg primitives.

Formal verification: FormalStatus(NameCert$_{\text{Cyclotomic}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CyclotomicUp.CyclotomicRootCarrier\_root\_layer\_classifier\_transport`

Bridge status: none

Not claimed: No standard cyclotomic field, class-field bridge, Galois bridge, or public cyclotomic-field quotient is claimed here.

Upgrade path: Only a later standard bridge verification target can raise the separate formal bridge status.

Chapter 175

A Concrete Naming Certificate for $\mathsf{EllipticCurve}^{\uparrow}$

The elliptic-curve interface $\mathsf{EllipticCurve}^{\uparrow}$ packages a smooth projective genus-one curve over a field carrier together with a chosen base point. It depends on $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ (chapter 155) and $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45).

175.1 Carrier definition

Definition 175.1 (Elliptic-curve BHist carrier packet)**.** *An* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier is a* BHist *packet*

$$\mathcal{E} = (\mathcal{F}, \mathcal{P}, a_1, a_2, a_3, a_4, a_6, \gamma, \sigma, o, \lambda_F, \lambda_P, \rho)$$

where $\mathcal{F}$ *is a source* $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ *row,* $\mathcal{P}$ *is a source* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ *row over the same field row,* $a_1, a_2, a_3, a_4, a_6 : \mathsf{BHist}$ *are field-coefficient rows,* $\gamma : \mathsf{BHist}$ *is the homogeneous cubic and projective-incidence row,* $\sigma : \mathsf{BHist}$ *is the smooth genus-one boundary row,* $o : \mathsf{BHist}$ *is the chosen base-point row,* λ_F *and* λ_P *are the field and projective classifier rows witnessed through* hsame*, and* ρ *records the* Pkg *provenance of the two source certificates and the displayed rows.*

The accepted endpoint is exactly the displayed history packet. It contains no ambient curve equality, no external point set, and no host discriminant predicate; all coefficient, incidence, smoothness, base-point, classifier, continuation, and provenance data are rows of BHist*,* hsame*,* Cont*, and* Pkg*.*

Lean4 (definition) `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket`

175.2 The certificate

Theorem 175.2 (Elliptic-curve field source obligation)**.** *In every carrier packet of Definition 175.1, the coefficient rows* a_1, a_2, a_3, a_4, a_6*, the field operation rows used to form the Weierstrass cubic, and the nonzero-discriminant row are projected from the* $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ *dependency. The* $\mathsf{EllipticCurve}^{\uparrow}$ *boundary may read these rows only as* BHist *data with the* Pkg *provenance inherited from* $\mathcal{F}$*.*

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket field source obligation`

Proof. Project the carrier packet to its $\mathcal{F}$ component and to the five displayed coefficient rows. The FieldUp certificate supplies the addition, multiplication, negation, unit, and nonzero rows needed to interpret the coefficients and the discriminant expression.

The Weierstrass cubic row is then formed by reading those operations through the same BHist ledger. The discriminant row is not a host-side assertion about a curve; it is the nonzero field row reached through the projected FieldUp source.

Repack the coefficient rows, operation rows, and nonzero-discriminant row with the original Pkg provenance. This gives the exact field-source part of the elliptic-curve certificate surface. □

Theorem 175.3 (Elliptic-curve projective genus-one obligation)**.** *In every carrier packet of Definition 175.1, the projective equation row, projective-incidence row, smoothness row, and genus-one boundary row are carried by the* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ *dependency as BEDC rows. They are not imported as host curves, external schemes, or ambient geometric equalities.*

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket projective genus one obligation`

Proof. Project the packet to its $\mathcal{P}$ component and to the homogeneous cubic row γ. The ProjectiveVarUp source supplies the projective carrier, homogeneous zero-locus, incidence, classifier, and ledger rows needed to read γ as a projective BEDC row.

Next project the smooth genus-one boundary row σ. Its smoothness and genus-one readings are attached to the same projective package, so the row is routed through the ProjectiveVarUp carrier and exactness surface rather than through an external curve object.

Repacking $\mathcal{P}$, γ, and σ records the projective equation, incidence, smoothness, and genus-one data as the projective dependency portion of the $\mathsf{EllipticCurve}^{\uparrow}$ carrier. □

Theorem 175.4 (Elliptic-curve base-point stability obligation)**.** *The chosen base-point row o in Definition 175.1 is stable under the inherited projective classifier and the inherited field classifier. If the field source is transported by λ_F and the projective source is transported by λ_P, the transported packet carries a base-point row connected to o by* hsame *and accepted by the same* Cont *provenance surface.*

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket basepoint stability obligation`

Proof. Start with the base-point row o and the projective classifier row λ_P. Projective transport preserves the projective incidence row, so the image of o remains accepted by the transported projective carrier.

Use the field classifier row λ_F to transport the coordinate and coefficient readings used by the base-point incidence test. The same FieldUp source rows supply the operation and nonzero data needed after transport.

The continuation row records that the transported endpoint is read through the same base-point provenance route. Repack the transported field source, projective source, incidence row, and base-point row to obtain the stable $\mathsf{EllipticCurve}^{\uparrow}$ packet. □

Theorem 175.5 (Elliptic-curve classifier transport obligation)**.** *Classified replacement of the field source and the projective source transports the whole carrier packet of Definition 175.1. Componentwise* hsame *transports the coefficient, cubic, smoothness, genus-one, and base-point rows, while the displayed* Cont *rows preserve their acceptance and the* Pkg *row preserves their provenance.*

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket classifier transport obligation`

Proof. Project a classifier witness into its field part λ_F and projective part λ_P. The field part transports the coefficient and operation rows; the projective part transports the homogeneous cubic, incidence, smoothness, and genus-one rows.

Apply componentwise hsame to the displayed BHist entries. Each transported entry is accepted through its original Cont row after replacing the source endpoint by the classified endpoint.

Finally carry the provenance row ρ along the same componentwise route. Repacking the transported entries gives the full replacement carrier packet, with no additional quotient or host equality inserted at the boundary. □

Theorem 175.6 (Base-point readback obligation)**.** *For every* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier packet of Definition 175.1, the base-point row* o *is readable as a point row of the carried* $\mathsf{ProjectiveVar}^{\uparrow}$ *surface. If* $o\mathsf{hsame}o'$ *and the* $\mathsf{ProjectiveVar}^{\uparrow}$ *classifier transports the point row from* o *to* o'*, then replacing* o *by* o' *gives a carrier packet with the same coefficient, cubic, smooth genus-one, field, classifier, and* Pkg *provenance rows.*

Proof. Project the $\mathsf{ProjectiveVar}^{\uparrow}$ source certificate $\mathcal{P}$ from the carrier packet. Its point readback clause supplies the carrier row containing o, and the incidence ledger records that o lies on the displayed cubic row γ.

Apply the hsame transport hypothesis to the point row. The $\mathsf{ProjectiveVar}^{\uparrow}$ classifier transports incidence along this row, so the replacement point o' remains attached to the same projective carrier and the same cubic row.

The remaining rows are unchanged projections of the original packet. Repacking them with o' yields the transported $\mathsf{EllipticCurve}^{\uparrow}$ packet, and no field coefficient, smooth genus-one boundary, classifier, or provenance row is altered. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket basepoint readback obligation`

Theorem 175.7 (Smooth projective carrier obligation)**.** *An accepted* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier is accepted only through the* $\mathsf{ProjectiveVar}^{\uparrow}$ *projective-carrier, incidence, smoothness, genus-one, and point rows together with the* $\mathsf{Field}^{\uparrow}$ *scalar rows for the displayed equation coefficients. In particular, smoothness and genus-one status are certificate rows over the carried projective variety and field source, not host-level assertions about an unrecorded curve.*

Proof. Unpack an accepted carrier packet. The projective carrier and incidence rows are projections of $\mathcal{P}$, while the coefficient and scalar rows used by γ are projections of $\mathcal{F}$.

The smooth genus-one boundary row σ is then checked against those same projections. Its scope is the displayed projective carrier over the displayed field coefficients, so there is no separate ambient curve object to validate.

Finally attach the base-point row by Definition 175.6. The accepted carrier is therefore exactly the smooth genus-one projective carrier with the named base point and the named field dependency rows. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket smooth projective carrier obligation`

Theorem 175.8 (Field dependency ledger obligation)**.** *All coefficient and scalar comparisons in an* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier packet descend through the* $\mathsf{Field}^{\uparrow}$ *classifier and the* Pkg *provenance ledger. If two packets have* hsame*-transported equation rows and the* $\mathsf{Field}^{\uparrow}$ *source certificate classifies their coefficient rows as the same scalar data, then every scalar comparison used by the projective incidence, smoothness, genus-one, and base-point clauses is transported by the same finite ledger.*

Proof. Read the coefficient rows a_1, a_2, a_3, a_4, a_6 from the carrier packet of Definition 175.1. Those rows are not interpreted directly; they are projected through the $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ source certificate $\mathcal{F}$.

Use the $\mathsf{Field}^{\uparrow}$ classifier to transport each scalar comparison along the displayed hsame relation on coefficient and cubic rows. The finite Pkg provenance row records the order in which these transported scalar rows feed the projective incidence, smooth genus-one, and base-point checks.

Repack the transported scalar rows with the unchanged projective certificate rows. The result is the same dependency ledger for the elliptic-curve packet, now read over the transported coefficient and cubic rows. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket field dependency ledger obligation`

Theorem 175.9 (EllipticCurveUp NameCert exactness obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$ *obligation surface consists exactly of the carrier packet of Definition 175.1, the field-source rows of Definition 175.2, the projective genus-one rows of Definition 175.3, the base-point stability row of Definition 175.4, the classifier transport row of Definition 175.5, the base-point readback of Definition 175.6, the smooth projective validation of Definition 175.7, the field dependency transport of Definition 175.8, and the ledger exactness asserting that no row outside* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ *is exported.*

Proof. Soundness is obtained by projecting a public elliptic-curve certificate to its listed packet rows. The carrier definition supplies the field source, projective-variety source, equation, smoothness, genus-one, base-point, and provenance rows.

Completeness goes in the reverse direction. Given exactly those rows with the field-source, projective genus-one, base-point, classifier, readback, smooth projective, and field dependency clauses, assemble the tuple of Definition 175.1 and read it as the public $\mathsf{EllipticCurve}^{\uparrow}$ surface.

The exclusion clause follows from the same projection list. Since every accepted row is one of the displayed BHist rows or one of the two source $\mathsf{NameCert}$ projections, the certificate exports no unrecorded host curve, no external equation equality, and no standard bridge row. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket semantic name certificate`

Theorem 175.10 (Elliptic-curve Weierstrass coefficient ledger)**.** *In the carrier packet of Definition 175.1, the five coefficient rows* a_1, a_2, a_3, a_4, a_6 *and every scalar operation used by the displayed Weierstrass cubic are projected through the* $\mathsf{Field}^{\uparrow}$ *source certificate and the* Pkg *ledger. The cubic row is therefore a BEDC row over the inherited field source, not a host polynomial equality.*

Proof. Project the packet to its field source $\mathcal{F}$ and to the coefficient rows a_1, a_2, a_3, a_4, a_6. The field source obligation of Definition 175.2 supplies the addition, multiplication, negation, unit, and nonzero rows used by the cubic.

Next read the provenance component ρ of the packet. It records that the coefficient and scalar-operation rows enter the cubic through the same Pkg ledger as the $\mathsf{Field}^{\uparrow}$ source.

Repacking these projections gives the coefficient ledger. Since every scalar row comes from $\mathcal{F}$ and ρ, there is no unrecorded host polynomial equality in the certificate surface. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket weierstrass coefficient ledger`

Theorem 175.11 (Elliptic-curve base-point incidence exactness)**.** *Base-point acceptance in an* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier packet is exactly projective point carriage plus incidence with the displayed cubic under the inherited* $\mathsf{ProjectiveVar}^{\uparrow}$ *classifier. Replacing it by an ambient point-set membership predicate or by host curve equality changes the certificate surface.*

Proof. Project the packet to its projective source $\mathcal{P}$, cubic row γ, and base-point row o. The projective genus-one obligation of Definition 175.3 supplies the projective carrier and incidence rows attached to γ.

Use Definition 175.6 to read o as a point row of the carried projective surface. The base-point stability row of Definition 175.4 transports that reading through the inherited projective classifier.

The exactness obligation of Definition 175.9 excludes rows outside the displayed packet and its two source certificates. Thus base-point acceptance is precisely carriage plus incidence with γ under the inherited classifier. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket basepoint incidence exactness`

Theorem 175.12 (Elliptic-curve field-projective source compatibility)**.** *Transporting the* $\mathsf{Field}^{\uparrow}$ *source and the* $\mathsf{ProjectiveVar}^{\uparrow}$ *source together transports the Weierstrass cubic, smooth genus-one row, and base-point row of an* $\mathsf{EllipticCurve}^{\uparrow}$ *packet. The transported packet adds no host curve equality and no external comparison between projective models.*

Proof. Project the classifier transport witnesses λ_F and λ_P from the carrier packet. The classifier transport obligation of Definition 175.5 applies λ_F to the coefficient and scalar rows and applies λ_P to the cubic, incidence, smoothness, genus-one, and base-point rows.

The field dependency ledger of Definition 175.8 keeps every scalar comparison on the $\mathsf{Field}^{\uparrow}$ side. The base-point stability row of Definition 175.4 keeps point acceptance on the $\mathsf{ProjectiveVar}^{\uparrow}$ side.

Repack the transported field source, projective source, cubic, smooth genus-one row, base-point row, classifier rows, and provenance row. The result is a transported $\mathsf{EllipticCurve}^{\uparrow}$ packet whose compatibility is witnessed entirely by the two source certificates and their BEDC rows. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket field projective source compatibility`

Closure status for $\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$.

Constructive story: An elliptic curve is generated as a BHist packet containing a $\mathsf{Field}^{\uparrow}$ source row, a $\mathsf{ProjectiveVar}^{\uparrow}$ source row over the same field, five field-coefficient rows, a homogeneous cubic and incidence row, a smooth genus-one boundary row, a chosen base-point row, classifier transports for the two sources, and Pkg provenance. The carrier accepts exactly packets with those coefficient, cubic, smoothness, genus-one, base-point, field, projective, continuation, and provenance rows; it excludes an ambient curve object, host discriminant predicate, external point set, group law, Weierstrass standard bridge, and equality between curve models. Two accepted packets classify together when the inherited field classifier transports coefficient and scalar-operation rows, the inherited projective classifier transports the cubic, incidence, smoothness, genus-one, and base-point rows, and componentwise hsame plus Cont keeps each row accepted. The source specification is this field-projective carrier packet, the pattern records field-source obligations, projective genus-one obligations, base-point stability, classifier transport, smooth projective validation, and field dependency ledgers, while the ledger keeps the raw coefficient rows, cubic row, base point, source classifiers, and provenance visible. The five naming-certificate obligations hold because an accepted packet inhabits the carrier, the inherited field and projective classifiers provide reflexivity, symmetry, and transitivity, and the classifier transport theorem repacks every transported coefficient, incidence, and base-point row as another accepted packet. A history h lies in the signature gap of an $\mathsf{EllipticCurve}^{\uparrow}$ token p when the source domain accepts this packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, the field row, projective row, Weierstrass coefficients, cubic incidence, smooth genus-one row, and base point of h are the rows encoded by p. The chapter registers this smooth projective genus-one packet under $\mathsf{EllipticCurve}^{\uparrow}$; it introduces no elliptic-curve group operation, no external Weierstrass bridge, and no independent curve-equality primitive.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$, maturePackageClosed)

Scope closed: The semantic certificate is the public exported $\mathsf{EllipticCurve}^{\uparrow}$ carrier packet over the inherited $\mathsf{Field}^{\uparrow}$ and $\mathsf{ProjectiveVar}^{\uparrow}$ sources, with the displayed coefficient, cubic, base-point, transport, classifier, and provenance rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.EllipticCurveUp.EllipticCurveCarrierPacket\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: The chapter does not close an elliptic-curve group law, a standard Weierstrass bridge, or external equality between curve models.

Upgrade path: No further formal-axis upgrade remains for the recorded field-projective genus-one bridge.

Definition 175.13 (Elliptic-curve Weierstrass admission profile)**.** *A Weierstrass admission profile for the public* $\mathsf{EllipticCurve}^{\uparrow}$ *surface is a finite row family*

$$W = (\mathcal{F}, \mathcal{P}, a_1, a_2, a_3, a_4, a_6, \gamma, \sigma, o, \lambda_F, \lambda_P, \rho)$$

with the following boundary readings. The row $\mathcal{F}$ *is a* $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ *source row; the five rows* $a_1, a_2, a_3, a_4, a_6 : \mathsf{BHist}$ *and the scalar operations used in the Weierstrass cubic are read through that field source;* $\mathcal{P}$ *is a* $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ *source row over the same field row;* $\gamma : \mathsf{BHist}$ *is the homogeneous cubic incidence row over the five displayed coefficients;* $\sigma : \mathsf{BHist}$ *is the smooth genus-one boundary row over that projective cubic;* $o : \mathsf{BHist}$ *is the chosen base-point incidence row;* λ_F *and* λ_P *are the inherited field and projective classifier transport rows; and* ρ *is the* Pkg *provenance row naming exactly these source, coefficient, cubic, smoothness, base-point, and transport entries.*

The profile has no coordinate for an elliptic-curve group operation, no coordinate for equality with an external curve model, and no coordinate for host-level equality between a standard Weierstrass presentation and a projective carrier. Such data may be discussed only after a separate row is supplied outside this admission profile.

Lean4 (definition) `BEDC.Derived.EllipticCurveUp.EllipticCurveUp`

Theorem 175.14 (Elliptic-curve Weierstrass bridge boundary)**.** *Let* W *be a Weierstrass admission profile as in Definition 175.13. If every standard-presentation datum in* W *is admitted only through the* $\mathsf{Field}^{\uparrow}$ *source row, the* $\mathsf{ProjectiveVar}^{\uparrow}$ *source row, the five coefficient rows, the cubic incidence row, the smooth genus-one row, the base-point row, the two classifier transport rows, and the* Pkg *provenance row listed in Definition 175.1, then* W *repacks as the same public* $\mathsf{EllipticCurve}^{\uparrow}$ *carrier packet. Projecting the public* $\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$ *surface recovers exactly those rows, and exports no elliptic-curve group law, no external curve equality, and no equality between a host Weierstrass model and the carried projective model.*

Proof. Start with the field side of the profile. By Definition 175.2, the five coefficient rows and the scalar operation rows used by the cubic are readable only through the $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ source $\mathcal{F}$. By Definition 175.10, their contribution to the Weierstrass equation is the displayed cubic row γ together with its Pkg provenance, not a host polynomial equality.

Project the geometric side next. By Definition 175.3, the cubic incidence row γ and the smooth genus-one row σ are carried by the $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$ dependency as BEDC rows. By Definition 175.7, smoothness and genus-one status are checked against the same field and projective projections, so the profile does not introduce a second curve object.

For the chosen point, Definition 175.11 identifies base-point acceptance with projective point carriage plus incidence with γ under the inherited projective classifier. The base-point row therefore repacks with $\mathcal{P}$, γ, and σ as the same point coordinate o of the public packet; it is not an external point-set membership predicate.

Now apply the two transport rows. The classifier transport obligation of Definition 175.5 transports coefficients, cubic, smoothness, genus-one, and base-point rows componentwise through λ_F, λ_P, hsame, and Cont. The field-projective compatibility theorem, Definition 175.12, says that this simultaneous transport still repacks as an $\mathsf{EllipticCurve}^{\uparrow}$ packet and adds no host comparison between projective models.

It remains to identify the public export. By Definition 175.9, the public $\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$ surface consists exactly of the carrier packet, the field-source rows, the projective genus-one rows, the base-point rows, the classifier transport rows, and the ledger exactness over BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{ProjectiveVar}^{\uparrow}}$. These are precisely the coordinates of W. Hence projecting the public surface recovers the same tuple, while any group law row, host curve-equality row, or external Weierstrass-to-projective equality row would require a coordinate absent from both Definition 175.13 and Definition 175.1. □

Lean4 ✓ `BEDC.Derived.EllipticCurveUp.EllipticCurveUp StdBridge`

Chapter 176

A Concrete Naming Certificate for $\mathsf{AbelianCat}^{\uparrow}$

The abelian-category interface $\mathsf{AbelianCat}^{\uparrow}$ packages an additive category in which every morphism has a kernel and cokernel and the canonical epi-mono factorisation is an isomorphism. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) and $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ (chapter 42).

176.1 Carrier definition

Definition 176.1 (AbelianCatUp additive kernel-cokernel carrier)**.** *An* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier is a concrete* BHist *package*

$$\mathcal{A} = (\mathcal{C}, \mathcal{H}, \mathcal{Z}, \mathcal{B}, \mathcal{G}, \mathcal{K}, \mathcal{Q}, \mathcal{F})$$

with the following displayed rows.

1. $\mathcal{C}$ *is a* $\mathsf{Cat}^{\uparrow}$ *carrier and classifier row, with object carrier, morphism carrier, identity rows, composition rows, associativity, unit laws, and componentwise* hsame *transport.*
2. $\mathcal{H}(A, B)$ *is the visible hom carrier row for each displayed pair of objects* A, B*, classified by the morphism classifier inherited from* $\mathcal{C}$*.*
3. $\mathcal{Z}(A, B)$ *gives the visible zero morphism row* $0_{A,B} : A \to B$*.*
4. $\mathcal{B}(A, B)$ *gives the visible binary biproduct rows for* A *and* B*, including the object* $A \oplus B$*, projections, injections, and their four classifier equations.*
5. $\mathcal{G}(A, B)$ *is an* $\mathsf{AbGroup}^{\uparrow}$ *enrichment row on* $\mathcal{H}(A, B)$*, with addition, zero, inverse, commutativity, and classifier congruence displayed in the hom row.*
6. $\mathcal{K}(f)$ *gives, for every visible morphism* $f : A \to B$*, an object* $\ker(f)$*, a morphism* $k_f : \ker(f) \to A$*, a zero-composite row, and the universal factorisation ledger through* k_f*.*
7. $\mathcal{Q}(f)$ *gives, for every visible morphism* $f : A \to B$*, an object* $\operatorname{coker}(f)$*, a morphism* $q_f : B \to \operatorname{coker}(f)$*, a zero-composite row, and the universal cofactorisation ledger through* q_f*.*
8. $\mathcal{F}(f)$ *is the visible epi–mono factorisation ledger for* f*, displaying image and coimage objects, the quotient-side arrow, the inclusion-side arrow, the canonical comparison, and the classifier row saying that the recomposed morphism is* f*.*

No object in this carrier is supplied by an external quotient or hidden subobject classifier. The kernel, cokernel, image, coimage, and comparison objects are selected rows of the same BHist *package.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatTasteGate single carrier alignment`

Theorem 176.2 (AbelianCatUp carrier classifier obligation)**.** *Let $\mathcal{A}$ be an* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier. If two presentations of $\mathcal{A}$ are related by componentwise* hsame *on the* $\mathsf{Cat}^{\uparrow}$ *carrier, on every hom carrier, on every zero, biproduct,* $\mathsf{AbGroup}^{\uparrow}$ *enrichment, kernel, cokernel, image, coimage, comparison, and factorisation row, then the package carrier and every displayed row are transported to a classified presentation of the same* $\mathsf{AbelianCat}^{\uparrow}$ *carrier.*

Proof. Start with the $\mathsf{Cat}^{\uparrow}$ carrier row. Its object, morphism, identity, and composition classifiers are part of $\mathcal{C}$, so componentwise hsame transports the category carrier and the source and target data of each visible morphism.

For each pair of transported objects A, B, the hom row $\mathcal{H}(A, B)$ is classified by the transported morphism classifier. The zero row, biproduct object, projections, and injections are displayed rows over the same endpoints, so the $\mathsf{Cat}^{\uparrow}$ identity, composition, and classifier stability fields transport their typing and equations.

It remains to transport the additive and exactness rows. The $\mathsf{AbGroup}^{\uparrow}$ enrichment supplies hsame-stable addition, zero, inverse, and commutativity on each hom row. The kernel, cokernel, image, coimage, comparison, and factorisation ledgers are rows indexed by visible morphisms, so transporting the morphism row transports the indexed ledger row and its displayed composition equations. Repacking these transported projections gives the same $\mathsf{AbelianCat}^{\uparrow}$ carrier presentation. □

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier classifier transport`

Theorem 176.3 (AbelianCatUp additive zero-biproduct obligation)**.** *For an* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier $\mathcal{A}$, the zero morphisms, hom-row addition, and binary biproduct projections and injections are exactly the visible additive rows inherited from* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *on each hom carrier and from* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *on identities, composition, and biproduct equations.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier zero biproduct rows`

Proof. Fix displayed objects A and B. The hom row $\mathcal{H}(A, B)$ carries the $\mathsf{AbGroup}^{\uparrow}$ enrichment row $\mathcal{G}(A, B)$, so its zero element is the displayed zero morphism $0_{A,B}$ and its binary operation is the displayed hom-addition row. The $\mathsf{AbGroup}^{\uparrow}$ classifier congruence field supplies the transport of these rows under hom-classifier equivalence.

The biproduct row $\mathcal{B}(A, B)$ is a $\mathsf{Cat}^{\uparrow}$ row: it supplies the object $A \oplus B$, projections, injections, and the four equations expressed by the category morphism classifier. The zero morphisms appearing in those equations are the same $\mathcal{Z}$ rows selected by the hom $\mathsf{AbGroup}^{\uparrow}$ zero fields.

Thus the additive part is not an extra semantic layer. The hom addition and zero rows are the $\mathsf{AbGroup}^{\uparrow}$ visible operation rows, and the biproduct typing and equations are the $\mathsf{Cat}^{\uparrow}$ visible composition rows applied to the displayed projections and injections. □

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier additive readback`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatKernelCokernelCarrier zero biproduct readback`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier zero additive kernel boundary`

Theorem 176.4 (AbelianCatUp kernel-cokernel factorisation obligation)**.** *For every visible morphism $f : A \to B$ in an* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier, the package exposes kernel, cokernel, image, coimage, canonical comparison, and epi–mono factorisation rows. The classified factorisation of f is obtained from these rows and no hidden quotient object or external subobject is used.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatKernelCokernelCarrier factorization rows`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatKernelCokernel visible factorization`

Proof. Project the kernel row $\mathcal{K}(f)$. It contains $\ker(f)$, the morphism $k_f : \ker(f) \to A$, the zero-composite classifier for $f \circ k_f$, and the factorisation ledger for every displayed morphism whose composite with f is classified as zero.

Project the cokernel row $\mathcal{Q}(f)$. It contains $\operatorname{coker}(f)$, the morphism $q_f : B \to \operatorname{coker}(f)$, the zero-composite classifier for $q_f \circ f$, and the cofactorisation ledger for every displayed morphism out of B whose precomposition with f is classified as zero.

The image, coimage, comparison, and epi–mono factorisation data are the visible rows of $\mathcal{F}(f)$. Category composition stability types the recomposition, while the factorisation classifier row identifies that recomposition with f. Since every object and arrow used in the argument is a projection of $\mathcal{K}(f)$, $\mathcal{Q}(f)$, or $\mathcal{F}(f)$, the proof does not appeal to any object outside the displayed BHist package. □

176.2 The certificate

Definition 176.5 (AbelianCatUp zero-biproduct-kernel surface)**.** *The* $\mathsf{AbelianCat}^{\uparrow}$ *zero-biproduct-kernel surface over* BHist *is the displayed obligation package carried by an* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier $\mathcal{A}$. It consists of the following rows.*

1. *The* $\mathsf{Cat}^{\uparrow}$ *object and morphism carriers, with identity, composition, associativity, unit, source, target, and morphism-classifier rows.*
2. *For each displayed object pair A, B, a hom carrier $\mathcal{H}(A, B)$ and an* $\mathsf{AbGroup}^{\uparrow}$ *classifier on that hom row, including visible addition, zero, inverse, commutativity, and additive classifier congruence.*
3. *A zero-object row $0_{\mathcal{A}}$ with displayed maps $0_{\mathcal{A}} \to A$ and $A \to 0_{\mathcal{A}}$ classified through the inherited* $\mathsf{Cat}^{\uparrow}$ *morphism classifier.*
4. *For each displayed pair A, B, a binary biproduct row $A \oplus B$ with projections, injections, the two diagonal identity rows, and the two off-diagonal zero rows read in the hom-wise* $\mathsf{AbGroup}^{\uparrow}$ *classifiers.*
5. *For each visible morphism $f : A \to B$, a kernel row $\ker(f) \xrightarrow{k_f} A$ with zero-composite and universal factorisation ledger entries.*
6. *For each visible morphism $f : A \to B$, a cokernel row $B \xrightarrow{q_f} \operatorname{coker}(f)$ with zero-composite and universal cofactorisation ledger entries.*
7. *An exactness row whose image, coimage, canonical comparison, and epi–mono recomposition entries are visible projections of the same* BHist *package.*
8. *A ledger provenance row recording the source row for every carrier, classifier, structural map, universal witness, and recomposition certificate used by the preceding rows.*

This surface is a $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *obligation interface: it names the rows downstream category and homological chapters may consume.*

Lean4 (definition) `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface`

Theorem 176.6 (AbelianCatUp zero-object obligation)**.** *For the surface of Definition 176.5, the zero-object row is classifier-stable. If A and A' are related by the* $\mathsf{Cat}^{\uparrow}$ *object classifier, then the displayed rows $0_{\mathcal{A}} \to A$, $A \to 0_{\mathcal{A}}$, $0_{\mathcal{A}} \to A'$, and $A' \to 0_{\mathcal{A}}$ are related by the inherited morphism classifier, and the two composites through $0_{\mathcal{A}}$ are read as hom-wise* $\mathsf{AbGroup}^{\uparrow}$ *zero elements.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface zero object transport`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface zero object boundary`

Proof. The zero object itself is a displayed carrier row of the same BHist package as the $\mathsf{Cat}^{\uparrow}$ object carrier. Transporting A to A' along the object classifier therefore transports the source and target typing of the two visible maps attached to $0_{\mathcal{A}}$.

The morphism classifier of $\mathsf{Cat}^{\uparrow}$ then transports the two maps and their composites. When a composite is viewed inside a hom row, the $\mathsf{AbGroup}^{\uparrow}$ classifier on that hom row identifies the displayed composite with the listed zero element.

Thus classifier stability is obtained by projecting the object row, then the morphism row, then the hom-wise zero classifier. No object outside the displayed carrier is introduced. □

Theorem 176.7 (AbelianCatUp biproduct additive hom obligation)**.** *For the surface of Definition 176.5, each binary biproduct row preserves the hom-wise* $\mathsf{AbGroup}^{\uparrow}$ *structure. The projections and injections of $A \oplus B$ are classified morphisms, their diagonal composites are the hom-wise additive identity rows, and their off-diagonal composites are the hom-wise zero rows.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface factor tail transport`

Proof. Fix displayed objects A and B. The biproduct row supplies the object $A \oplus B$ together with projections and injections typed by the $\mathsf{Cat}^{\uparrow}$ source and target rows.

Composition in $\mathsf{Cat}^{\uparrow}$ types the four projection-injection composites. The two diagonal composites are classified as the identity rows in the appropriate hom carriers, while the two off-diagonal composites are classified as the zero rows selected by the hom-wise $\mathsf{AbGroup}^{\uparrow}$ structures.

The additive compatibility is therefore hom-local. Each equation is read inside a single $\mathcal{H}(X, Y)$, and the $\mathsf{AbGroup}^{\uparrow}$ classifier congruence transports addition, zero, and inverse across the same morphism classifier that types the biproduct maps. □

Theorem 176.8 (AbelianCatUp kernel-cokernel obligation)**.** *For the surface of Definition 176.5, kernel and cokernel rows are stable under the* $\mathsf{Cat}^{\uparrow}$ *morphism classifier. If visible morphisms f and g are related by that classifier, then the displayed rows* $\ker(f)$, k_f, $\operatorname{coker}(f)$, *and q_f transport to the corresponding rows for g together with their zero-composite and universal ledger entries.*

Proof. Begin with a classifier relation between the visible morphisms f and g. The kernel and cokernel rows are indexed by visible morphism rows, so transport along the morphism classifier carries the index f to the index g.

For kernels, this transport carries the object $\ker(f)$, the map k_f, the zero-composite row for $f \circ k_f$, and the factorisation ledger through k_f. The target statement for g is exactly the transported indexed row.

For cokernels, the same argument is dual at the displayed-row level: the object $\operatorname{coker}(f)$, the map q_f, the zero-composite row for $q_f \circ f$, and the cofactorisation ledger transport along the same morphism-classifier relation. □

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatKernelCokernelCarrier classifier transport`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier factor unary closure`

Theorem 176.9 (AbelianCatUp exact-sequence obligation)**.** *For the surface of Definition 176.5, the exactness row is read only through displayed kernel and cokernel factorisation classifiers. Every exact-sequence assertion in the surface is a statement about the listed kernel, cokernel, image, coimage, comparison, and recomposition rows of the same* BHist *package.*

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatKernelCokernel recomposition audit row`
Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatAdditiveCarrier factor append readback`

Proof. An exactness assertion is first reduced to the visible morphism whose image and coimage rows are listed in the package. The kernel and cokernel rows provide the zero-composite classifiers and the universal ledger entries required to factor through those rows.

The image, coimage, and comparison entries are then projected from the factorisation ledger. Their recomposition row is typed by $\mathsf{Cat}^{\uparrow}$ composition and classified as the original visible morphism by the displayed recomposition classifier.

Consequently the exactness predicate has no separate hidden carrier. It is a reading discipline for the kernel, cokernel, image, coimage, comparison, and factorisation rows already present in the surface. □

Theorem 176.10 (AbelianCatUp NameCert obligation package)**.** *The surface of Definition 176.5, together with Definition 176.6, Definition 176.7, Definition 176.8, and Definition 176.9, repacks as the paper-side* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *obligation package for downstream category and homological chapters.*

Proof. The carrier and classifier fields are the first two rows of Definition 176.5. They expose the $\mathsf{Cat}^{\uparrow}$ object and morphism rows, the hom carriers, and the hom-wise $\mathsf{AbGroup}^{\uparrow}$ classifiers.

The structural stability fields are supplied by the four obligation theorems. The zero-object theorem gives the classifier-stable zero-object row; the biproduct theorem gives the additive hom compatibility of projections and injections; the kernel-cokernel theorem gives stability of universal rows under the morphism classifier; the exact-sequence theorem restricts exactness to the displayed factorisation classifiers.

The ledger field is the final row of the surface. Repacking these projections gives a single $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ obligation package whose consumer chapters cite concrete rows rather than an unspecified abelian-category interface. □

Theorem 176.11 (AbelianCatUp NameCert obligation surface)**.** *An* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier* $\mathcal{A}$ *satisfying Definition 176.2, Definition 176.3, and Definition 176.4 supplies the following ten-field* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *obligation surface.*

1. *Traditional target: an abelian category with additive hom groups, binary biproducts, kernels, cokernels, and canonical epi–mono factorisations.*
2. *BEDC source: the concrete* BHist *package of Definition 176.1.*
3. *Carrier: the displayed object and morphism rows of the* $\mathsf{Cat}^{\uparrow}$ *component together with the visible hom carriers.*
4. *Classifier: the componentwise* hsame *classifier on objects, morphisms, hom elements, additive rows, kernel rows, cokernel rows, and factorisation rows.*

5. *Stability obligations:* $\mathsf{Cat}^{\uparrow}$ *composition and identity stability,* $\mathsf{AbGroup}^{\uparrow}$ *hom-addition stability, biproduct equation stability, and componentwise* hsame *transport.*
6. *Ledger: source, hom, zero, addition, biproduct, kernel, cokernel, image, coimage, comparison, and epi–mono factorisation provenance rows.*
7. *Core theorem list: Definition 176.2, Definition 176.3, and Definition 176.4.*
8. *Exactness predicate: every visible morphism has displayed kernel, cokernel, image, coimage, comparison, and classified recomposition rows.*
9. *Absent standard bridge: no external category of abelian groups, modules, sheaves, or complexes is identified here.*
10. *Status:* ProofTarget *for a later Lean theorem in* BEDC.Derived.AbelianCatUp.

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface semantic name certificate`

Proof. The carrier, classifier, and stability fields are projections of Definition 176.1. The transport requirement is exactly Definition 176.2, which repacks componentwise hsame transport across the category, hom, additive, kernel, cokernel, and factorisation rows.

The additive entries of the surface are supplied by Definition 176.3. That theorem identifies the hom addition and zero morphism rows with the $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ visible operation rows and identifies the biproduct typing and equations with the $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ visible composition rows.

The exactness and ledger entries are supplied by Definition 176.4. For each visible morphism it projects kernel, cokernel, image, coimage, comparison, and recomposition rows from the package and records that no hidden quotient object is used. These three projections assemble the stated ten-field $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ obligation surface. □

Theorem 176.12 (AbelianCatUp hom zero morphism uniqueness)**.** *Let* $\mathcal{A}$ *be an* $\mathsf{AbelianCat}^{\uparrow}$ *additive kernel-cokernel carrier and view its zero-biproduct-kernel surface as in Definition 176.5. Fix displayed objects* A *and* B*. Write* $\mathcal{H}(A,B)$ *for the displayed hom carrier,* $\sim_{A,B}$ *for its inherited hom classifier,* $+_{A,B}$ *for its displayed hom addition, and* $0_{A,B}$ *for its displayed zero morphism. If a carried morphism* $u \in \mathcal{H}(A,B)$ *satisfies*

$$u +_{A,B} f \sim_{A,B} f \qquad \text{and} \qquad f +_{A,B} u \sim_{A,B} f$$

for every carried $f \in \mathcal{H}(A,B)$*, then*

$$u \sim_{A,B} 0_{A,B}.$$

Proof. Unpack the hom row selected by Definition 176.1. It carries the $\mathsf{AbGroup}^{\uparrow}$ enrichment row $\mathcal{G}(A,B)$. By Definition 176.3, the addition operation of this enrichment is $+_{A,B}$ and its identity element is the displayed zero morphism $0_{A,B}$.

Apply Definition 42.33 to the enrichment and then Definition 41.122 to the resulting group reduct. The retained monoid certificate has the same carrier $\mathcal{H}(A,B)$, the same classifier $\sim_{A,B}$, the same operation $+_{A,B}$, and identity $0_{A,B}$. Hence the monoid right-identity row says that $0_{A,B}$ is a right identity for every carried hom element.

The first displayed hypothesis says that u is a left identity for every carried hom element. Since u is carried and $0_{A,B}$ is carried by the zero row, the inputs of Definition 40.15 are satisfied with $e = u$ and $e' = 0_{A,B}$. Its conclusion is exactly $u \sim_{A,B} 0_{A,B}$, because the monoid reduct did not change the hom classifier. □

Lean4 ✓ `BEDC.Derived.AbelianCatUp.AbelianCatHomZeroMorphism uniqueness`

Closure status for $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$.

Constructive story: An abelian category is carried here as one displayed BHist package built on an already certified $\mathsf{Cat}^{\uparrow}$ surface and hom-wise $\mathsf{AbGroup}^{\uparrow}$ rows: the package accepts object and morphism histories only when it also exposes zero morphisms, biproducts, additive hom groups, kernels, cokernels, image–coimage comparison, and epi–mono factorisation rows, while any hidden quotient, subobject classifier, module category, or sheaf carrier is left outside. Two such packages count as the same only by componentwise history-equality through the category carrier, every hom carrier, the additive rows, and the kernel–cokernel and exactness ledgers, so the classifier is conditional transport of all displayed rows rather than equality of ambient categories. The source is the paired $\mathsf{Cat}^{\uparrow}$ and $\mathsf{AbGroup}^{\uparrow}$ certificate surface, the pattern is the zero-biproduct-kernel-cokernel inventory with the exactness comparison row, and the ledger keeps the raw category, hom, structural-map, universal-factorisation, and recomposition witnesses visible for downstream homological use. Because the category and abelian-group classifiers are already equivalence relations and every listed structural row is transported componentwise, an admitted package supplies carrier habitation, the classifier is reflexive, symmetric, and transitive, and any classified presentation repacks as an admitted abelian-category package. A package h lies in the signature gap of an $\mathsf{AbelianCat}^{\uparrow}$ token p when the source-domain accepts its category and hom-additive rows, the signature relation pairs that package with an answer package and evidence ledger, and the token-introduction relation names the answer as p; unfolded, this says that the same displayed kernels, cokernels, biproducts, hom zeros, and exact factorisation rows are the data encoded by p. The chapter registers this displayed additive exactness surface under the name $\mathsf{AbelianCat}^{\uparrow}$; it introduces no standard abelian category, module category, sheaf model, or external equality beyond the existing BHist rows and their transport.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public surface exports the zero-biproduct, kernel-cokernel, exactness, NameCert, and hom-zero uniqueness rows carried by the displayed BHist package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.AbelianCatUp.AbelianCatZeroBiproductKernelSurface\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not identify standard abelian categories, modules, sheaves, complexes, or a checked external bridge.

Upgrade path: No further paper-axis upgrade remains for the displayed additive exactness surface.

Theorem 176.13 (AbelianCatUp zero morphism right composition absorption)**.** *Let $\mathcal{A}$ be an $\mathsf{AbelianCat}^{\uparrow}$ additive kernel-cokernel carrier and view its zero-biproduct-kernel surface as in Definition 176.5. Fix displayed objects X, A, and B. Write $\mathcal{H}(Y,Z)$ for the displayed hom carrier, $\sim_{Y,Z}$ for its hom classifier, $+_{Y,Z}$ for hom addition, and $0_{Y,Z}$ for the displayed zero morphism. If $g \in \mathcal{H}(A,B)$ is carried, then the displayed composite with the zero morphism in the right composition slot is classified with the displayed zero morphism in the target hom row:*

$$g \circ 0_{X,A} \sim_{X,B} 0_{X,B}.$$

Proof. The hom rows $\mathcal{H}(X,A)$ and $\mathcal{H}(X,B)$ are part of the carrier of Definition 176.1. By Definition 176.3, their displayed additions and zero morphisms are the additions and units of the corresponding hom-wise $\mathsf{AbGroup}^{\uparrow}$ enrichment rows. In particular,

$$0_{X,A} +_{X,A} 0_{X,A} \sim_{X,A} 0_{X,A}.$$

The $\mathsf{Cat}^{\uparrow}$ composition row types the composite $u := g \circ 0_{X,A}$ as a carried element of $\mathcal{H}(X,B)$.

Project the right-slot additivity of composition from the additive $\mathsf{AbelianCat}^{\uparrow}$ carrier, read through the biproduct additive hom obligation of Definition 176.7. For the fixed carried morphism $g : A \to B$, postcomposition sends hom addition in $\mathcal{H}(X,A)$ to hom addition in $\mathcal{H}(X,B)$, so at the zero pair it supplies

$$g \circ (0_{X,A} +_{X,A} 0_{X,A}) \sim_{X,B} (g \circ 0_{X,A}) +_{X,B} (g \circ 0_{X,A}).$$

Composition classifier stability transports the source zero-unit comparison through the same right slot, giving

$$g \circ (0_{X,A} +_{X,A} 0_{X,A}) \sim_{X,B} g \circ 0_{X,A}.$$

By symmetry and transitivity of $\sim_{X,B}$, these two comparisons classify u as additively idempotent:

$$u +_{X,B} u \sim_{X,B} u.$$

Now read the hom-wise $\mathsf{AbGroup}^{\uparrow}$ enrichment of $\mathcal{H}(X, B)$ through Definition 42.33. The group reduct has the same carrier, the same classifier $\sim_{X,B}$, the same operation $+_{X,B}$, and the same unit $0_{X,B}$. Applying Definition 41.114 in additive notation, with $a = u$ and $b = u$, forces the right repeated summand to be the additive unit:

$$u \sim_{X,B} 0_{X,B}.$$

Substituting $u = g \circ 0_{X,A}$ gives the asserted classification. □

Theorem 176.14 (AbelianCatUp hom additive inverse uniqueness)**.** *Let $\mathcal{A}$ be an $\mathsf{AbelianCat}^{\uparrow}$ additive kernel-cokernel carrier and view its zero-biproduct-kernel surface as in Definition 176.5. Fix displayed objects A and B. Write $\mathcal{H}(A, B)$ for the displayed hom carrier, $\sim_{A,B}$ for its inherited hom classifier, $+_{A,B}$ for its displayed hom addition, and $0_{A,B}$ for its displayed zero morphism. If carried morphisms $f, u, v \in \mathcal{H}(A, B)$ satisfy*

$$u +_{A,B} f \sim_{A,B} 0_{A,B} \qquad \text{and} \qquad f +_{A,B} u \sim_{A,B} 0_{A,B}$$

with $\nu = u$, and also satisfy

$$v +_{A,B} f \sim_{A,B} 0_{A,B} \qquad \text{and} \qquad f +_{A,B} v \sim_{A,B} 0_{A,B},$$

then

$$u \sim_{A,B} v$$

with $\nu = u$. Equivalently, any two carried two-sided additive inverse presentations of the same carried hom morphism are identified by the inherited hom classifier.

Proof. Unpack the hom row selected by Definition 176.1. The row $\mathcal{H}(A, B)$ carries the $\mathsf{AbGroup}^{\uparrow}$ enrichment $\mathcal{G}(A, B)$. By Definition 176.3, the operation of this enrichment is the displayed hom addition $+_{A,B}$ and its identity is the displayed zero morphism $0_{A,B}$.

Apply Definition 42.33 to this enrichment. The retained $\mathsf{Group}^{\uparrow}$ package has the same carried hom elements, the same classifier $\sim_{A,B}$, the same operation $+_{A,B}$, the same identity $0_{A,B}$, and the inverse laws and multiplication congruence required for the group inverse-uniqueness argument.

Now read Definition 41.59 in this retained additive group package, with middle element f, left inverse candidate u, and right inverse candidate v. The hypothesis

$$u +_{A,B} f \sim_{A,B} 0_{A,B}$$

with $\nu = u$ is exactly the left-inverse comparison $u +_{A,B} f \sim_{A,B} 0_{A,B}$, and the hypothesis

$$f +_{A,B} v \sim_{A,B} 0_{A,B}$$

is exactly the right-inverse comparison for v. The theorem therefore gives

$$u \sim_{A,B} v$$

with $\nu = u$, because the group reduct did not change the hom classifier. The remaining two displayed comparisons record that both candidates are two-sided inverse presentations, but the uniqueness comparison itself needs only one left-inverse presentation and one right-inverse presentation against the same carried middle morphism. □

Chapter 177

A Concrete Naming Certificate for $\mathsf{DerivedFunctor}^{\uparrow}$

The derived-functor interface $\mathsf{DerivedFunctor}^{\uparrow}$ packages the right- or left-derived R^iF / L_iF of an additive functor between abelian categories. It depends on $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ (chapter 176) and $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ (chapter 98).

177.1 Carrier definition

Definition 177.1 (Derived-functor history carrier)**.** *For $h : \mathsf{Hist}$, the predicate* $\mathsf{DerivedFunctorCarrier}(h)$ *holds exactly when there are histories*

$$F, R, H, i, c : \mathsf{Hist}$$

read respectively as an additive-functor trace, a chosen resolution trace, a homology trace, a unary degree row, and the intermediate resolved trace, such that

$$\mathsf{UnaryHistory}(i) \wedge \mathsf{Cont}(F, R, c) \wedge \mathsf{Cont}(c, H, h).$$

The carrier is the concrete BHist *predicate generated by this tuple. The degree row is not a host-level integer choice; it is the displayed unary history used by the homology trace, while the two* Cont *rows record first the functor-to-resolution passage and then the homology readback that produces h.*

Lean4 (definition) `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier`

Definition 177.2 (DerivedFunctorUp resolution-spine carrier)**.** *A* $\mathsf{DerivedFunctor}^{\uparrow}$ *resolution-spine carrier is a* BHist *packet whose public rows consist of an* $\mathsf{AbelianCat}^{\uparrow}$ *source certificate, a* $\mathsf{Homology}^{\uparrow}$ *source certificate, an object row $X : \mathsf{BHist}$, a finite resolution spine*

$$R_0 \to R_1 \to \cdots \to R_n \to X,$$

cycle rows $Z_i : \mathsf{BHist}$, boundary rows $B_i : \mathsf{BHist}$, a displayed additive-functor row $F : \mathsf{BHist}$, and a Pkg *provenance row for the resolution ledger. The arrows in the finite spine are admissible exactly when their source and target rows are classified by the* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source surface, and the rows Z_i and B_i are read through the* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ *cycle and boundary classifiers at the displayed degree.*

Theorem 177.3 (Resolution-spine carrier obligation)**.** *The resolution-spine carrier of Definition 177.2 is closed under* hsame *transport of the object row and under* Cont *extension of the finite resolution spine. Transporting X to X' along XhsameX' replaces the endpoint object row while preserving the admissible* $\mathsf{AbelianCat}^{\uparrow}$ *source and arrow rows; appending one admissible* Cont *step to the finite spine gives another carried resolution ledger with the same* Pkg *provenance extended by that visible step.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorResolutionSpineCarrier obligation`

Proof. Unpack the carrier packet into its $\mathsf{AbelianCat}^{\uparrow}$ source rows, $\mathsf{Homology}^{\uparrow}$ source rows, object row, finite spine, functor row, and Pkg provenance row.

For object transport, apply the $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ classifier transport to the object row. The admissible arrow rows are then read with the transported endpoint, and the homology cycle and boundary rows remain attached to the same displayed degree because no cycle row is changed.

For spine extension, append the extra admissible arrow as one more Cont ledger row. The finite spine is still a BHist packet, the new endpoint is classified by the same $\mathsf{AbelianCat}^{\uparrow}$ source surface, and the provenance row is repacked as the old Pkg row followed by the displayed continuation step. □

Theorem 177.4 (Cycle-boundary classifier obligation)**.** *For two accepted* $\mathsf{DerivedFunctor}^{\uparrow}$ *resolution-spine packets, the public classifier compares derived-functor rows by the* $\mathsf{Homology}^{\uparrow}$ *cycle classifier modulo the displayed boundary rows. The* $\mathsf{AbelianCat}^{\uparrow}$ *source certificate supplies the admissible object and arrow transport needed to compare the selected resolution spines, while the* $\mathsf{Homology}^{\uparrow}$ *source certificate supplies the cycle and boundary comparison rows.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier cycle boundary classifier obligation`

Proof. Start with two carried packets and project their object, arrow, cycle, boundary, functor, and provenance rows. The $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ source compares the object row and every admissible arrow row in the selected finite spines, so both packets are read over classified resolution data.

Next project the $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ source. Its cycle classifier compares the displayed cycle rows at the selected degree, and its boundary classifier records exactly which boundary rows may be used to identify two cycles.

Repack these two comparisons as the $\mathsf{DerivedFunctor}^{\uparrow}$ classifier. The abelian-category part accounts only for transport of the resolution spine, while the homology part decides the public endpoint class by cycle comparison modulo the listed boundaries. □

Theorem 177.5 (Homology readback obligation)**.** *Every accepted* $\mathsf{DerivedFunctor}^{\uparrow}$ *endpoint reads back to a* $\mathsf{Homology}^{\uparrow}$ *cycle class attached to the selected resolution degree. If two accepted endpoints are boundary-equivalent through the displayed boundary rows, then they have the same public* $\mathsf{DerivedFunctor}^{\uparrow}$ *classifier.*

Proof. Read an accepted endpoint by unpacking its resolution-spine carrier. The selected degree row chooses the cycle row Z_i and boundary row B_i already carried by the packet, and the $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ source classifies Z_i as the visible cycle class.

For two endpoints, use the displayed boundary-equivalence row as a $\mathsf{Homology}^{\uparrow}$ boundary classifier witness. The homology certificate transports the cycle classifier across that boundary row, producing the same public homology class.

The $\mathsf{DerivedFunctor}^{\uparrow}$ classifier is defined by this homology readback after the $\mathsf{AbelianCat}^{\uparrow}$ resolution rows have been transported. Therefore boundary-equivalent endpoints have the same public classifier. □

Theorem 177.6 (Functorial transport obligation)**.** *An* $\mathsf{AbelianCat}^{\uparrow}$ *morphism row transported through the displayed additive functor sends accepted resolution spines to accepted resolution spines and preserves the* $\mathsf{Homology}^{\uparrow}$ *classifier on endpoints. The transported packet keeps the same* Pkg *provenance form, with each functorial arrow image recorded as a visible* Cont *row.*

Proof. Project the source morphism row and the displayed additive-functor row from the carrier. The $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ source transports the morphism through admissible object and arrow rows, so every arrow in the resolution spine has an admissible functorial image.

Assemble the image spine by the same finite Cont order as the original spine. Each image arrow contributes one continuation row, and the Pkg provenance row records these continuations as the functorial image of the original ledger.

Finally project the endpoint cycle and boundary rows through the $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ classifier. Since the functorial image respects the displayed boundary rows, the endpoint cycle class is transported to the same public classifier value in the image packet. □

Theorem 177.7 (DerivedFunctorUp ledger exactness obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ *surface consists exactly of the* $\mathsf{AbelianCat}^{\uparrow}$ *source rows,* $\mathsf{Homology}^{\uparrow}$ *cycle and boundary rows, the finite resolution spine, the functor row, the classifier transport rows, and* Pkg *provenance. It exports no host quotient of complexes and no external abelian-category equality beyond the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{AbelianCat}^{\uparrow}$*, and* $\mathsf{Homology}^{\uparrow}$ *rows.*

Proof. Soundness is obtained by unpacking a carried packet. The only public rows are the two source certificates, the finite spine, the functor row, the cycle and boundary rows, classifier transport, and the provenance row.

Completeness repacks exactly those rows. Given the $\mathsf{AbelianCat}^{\uparrow}$ object and arrow rows, the $\mathsf{Homology}^{\uparrow}$ cycle and boundary rows, the finite Cont spine, the functor row, and the Pkg provenance row, the resolution-spine carrier of Definition 177.2 is reconstructed without adding any host quotient.

The exactness boundary follows because every classifier theorem above projects from this same packet and every accepted endpoint reads back through the $\mathsf{Homology}^{\uparrow}$ cycle classifier. There is no exported row naming an equality of complexes or an external abelian-category equality, so such data cannot enter the public $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ surface. □

Theorem 177.8 (DerivedFunctorUp source-coupled homology readback)**.** *Every packet satisfying the resolution-spine carrier of Definition 177.2 projects one finite* BHist *resolution spine whose object and arrow rows are consumed by* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *and whose cycle, boundary, and unary-degree rows are consumed by* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$*. The two readings have the same* Pkg *provenance row, so the public* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ *surface cannot split them into unrelated resolution and homology sources.*

Proof. Unpack the carried packet into its $\mathsf{AbelianCat}^{\uparrow}$ source certificate, $\mathsf{Homology}^{\uparrow}$ source certificate, object row, finite resolution spine, functor row, displayed unary degree, cycle rows, boundary rows, and Pkg provenance.

Project the object and arrow rows to the $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ source surface. These are exactly the rows that classify the endpoint object and every admissible arrow in the finite spine.

Project the displayed unary degree, cycle rows, and boundary rows to the $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ source surface. They are read at the same selected degree and over the spine already classified by the abelian-category source rows.

The shared Pkg provenance row is part of the same carrier packet, not a separate input. Therefore the abelian-category transport side and the homology readback side are two projections of one BHist ledger, and no unrelated second spine is available to the public certificate. □

Theorem 177.9 (DerivedFunctorUp boundary over transported resolution)**.** *After* hsame *transport of the object row and* Cont *extension of the finite resolution spine, the selected boundary-equivalence row still compares the* $\mathsf{Homology}^{\uparrow}$ *cycle rows over the transported* $\mathsf{AbelianCat}^{\uparrow}$ *resolution data. Thus boundary readback remains attached to the transported source resolution rather than to an independent homology packet.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier boundary over transported resolution`

Proof. Use Definition 177.3 to transport the endpoint object row and extend the finite spine by the displayed continuation step. The result is again a carried resolution ledger with the same visible provenance form.

Apply Definition 177.4 to the transported carried packets. The abelian-category rows compare the transported object and arrow data, while the homology rows compare the selected cycles modulo the displayed boundary rows.

Finally read the endpoint through Definition 177.5. The boundary-equivalence witness is interpreted by the $\mathsf{Homology}^{\uparrow}$ source over the transported resolution data, so the public derived-functor classifier is preserved by the combined transport and readback. □

Theorem 177.10 (DerivedFunctorUp downstream consumer package)**.** *A downstream* $\mathsf{DerivedCat}^{\uparrow}$ *or* $\mathsf{SpectralSeq}^{\uparrow}$ *consumer may receive precisely the shared source certificates, finite spine, functor row, unary degree, cycle-boundary rows, classifier transport, and* Pkg *provenance carried by the public* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ *surface. It receives no host quotient of complexes and no external abelian-category equality.*

Proof. Start with the coupled source packet of Definition 177.8. It supplies one finite BHist spine together with the $\mathsf{AbelianCat}^{\uparrow}$ source rows and the $\mathsf{Homology}^{\uparrow}$ cycle, boundary, and degree rows that read that same spine.

Use Definition 177.9 for the consumer-facing transport step. The classifier rows remain tied to the transported resolution data, and the selected boundary comparison continues to read the homology endpoint over those transported rows.

Combine these projections with Definition 177.7. The exact public package is the source certificates, finite spine, functor row, unary degree, cycle-boundary rows, classifier transport, and Pkg provenance, with no exported host quotient of complexes and no external abelian-category equality. □

Definition 177.11 (DerivedFunctorUp exact-triangle boundary carrier)**.** *A* $\mathsf{DerivedFunctor}^{\uparrow}$ *exact-triangle boundary carrier is a concrete* BHist *packet whose public rows consist of one* $\mathsf{AbelianCat}^{\uparrow}$ *source row, the* $\mathsf{Homology}^{\uparrow}$ *cycle and boundary rows at the displayed degree, three resolution-spine packets*

$$R_A \to A, \qquad R_B \to B, \qquad R_C \to C,$$

the connecting-morphism row $\partial : C \to A[1]$*, and a* Pkg *provenance row for the boundary ledger. Each resolution-spine packet is a row of the form named in Definition 177.2; the* $\mathsf{AbelianCat}^{\uparrow}$ *source row classifies the admissible arrows in the exact-triangle source row, the* $\mathsf{Homology}^{\uparrow}$ *rows classify cycles and displayed boundaries, and the* Pkg *row records precisely the three spines, the connecting row, and the visible continuation ledgers that relate them.*

Lean4 (definition) `BEDC.Derived.DerivedFunctorUp.DerivedFunctorExactTriangleBoundaryCarrier`

Theorem 177.12 (Exact-triangle boundary carrier scope)**.** *The carrier of Definition 177.11 is closed under* hsame *transport of the* $\mathsf{AbelianCat}^{\uparrow}$ *source row and under* Cont *extension of each of the three displayed resolution spines. Transporting the source row preserves the admissible arrow rows for the exact triangle, and extending any one of* $R_A \to A$*,* $R_B \to B$*, or* $R_C \to C$ *by a displayed continuation step produces another exact-triangle boundary carrier with the same connecting-morphism row and with the* Pkg *provenance row extended by that visible step.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorExactTriangleBoundaryCarrier scope`

Proof. Unpack Definition 177.11. Each of the three spine rows is an instance of the resolution-spine packet of Definition 177.2, so the object rows, arrow rows, cycle rows, boundary rows, functor row, and provenance row are all visible BHist data.

For hsame transport of the source row, apply Definition 177.3 to read each spine over the transported $\mathsf{AbelianCat}^{\uparrow}$ classifier. The homology cycle and boundary rows remain attached to the same displayed degree, and the connecting-morphism row is transported as an admissible arrow row over the same source surface.

For Cont extension, apply Definition 177.3 separately to the affected spine. The other two spines and the connecting row are unchanged, and the boundary carrier is repacked by appending the displayed continuation row to the Pkg provenance ledger. □

Theorem 177.13 (Connecting-morphism readback)**.** *For every accepted exact-triangle boundary carrier, the connecting morphism* $\partial : C \to A[1]$ *is read back from the* $\mathsf{Homology}^{\uparrow}$ *boundary classifier together with the* $\mathsf{AbelianCat}^{\uparrow}$ *admissible-arrow transport. The boundary classifier supplies the row identifying the target cycle class modulo displayed boundaries, while the admissible-arrow transport supplies the source and target rows over the three resolution spines.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorExactTriangleBoundaryCarrier connecting morphism readback`

Proof. Begin with an accepted carrier and project the three resolution-spine packets supplied by Definition 177.11. By Definition 177.4, their endpoint comparisons are computed by the $\mathsf{Homology}^{\uparrow}$ cycle classifier modulo the displayed boundary rows.

Next project the connecting-morphism row. The source and target rows are admissible because the same $\mathsf{AbelianCat}^{\uparrow}$ source surface classifies the exact-triangle arrows, and Definition 177.6 transports those admissible rows through the displayed functorial image of the resolution data.

Finally read the target endpoint with Definition 177.5. The $\mathsf{Homology}^{\uparrow}$ boundary classifier is exactly the displayed witness that turns the target cycle into the connecting class, so the public readback of ∂ is obtained without adding a quotient of complexes or any external equality of abelian categories. □

Theorem 177.14 (Long-exact ledger surface)**.** *The long-exact ledger surface of an accepted exact-triangle boundary carrier consists exactly of the three resolution-spine rows, the displayed cycle and boundary rows, the connecting-morphism row, the* Cont *ledgers relating the three spines and their endpoint readbacks, and the* Pkg *provenance row. It exports no host quotient of complexes and no external exact-category equality beyond the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{AbelianCat}^{\uparrow}$*, and* $\mathsf{Homology}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorExactTriangleBoundaryCarrier long exact ledger surface`

Proof. Soundness follows by unpacking the boundary carrier of Definition 177.11. The only public rows named by the packet are the $\mathsf{AbelianCat}^{\uparrow}$ source row, the $\mathsf{Homology}^{\uparrow}$ cycle and boundary rows, the three resolution spines, the connecting row, the Cont ledgers, and the Pkg provenance row.

Completeness repacks exactly those rows. Use Definition 177.2 to recognize each of the three spines, then use Definition 177.12 to close the packet under the displayed hsame and Cont transports.

The connecting part is supplied by Definition 177.13, and the endpoint classes are the homology readbacks of Definition 177.5. Applying Definition 177.7 then shows that the public surface is exhausted by the listed rows, so no host quotient of complexes or external exact-category equality is exported. □

Theorem 177.15 (Derived-functor resolution source)**.** *If* $\mathsf{DerivedFunctorCarrier}(h)$ *holds, then there are histories* $F, R, H, i, c : \mathsf{Hist}$ *witnessing the additive-functor trace, resolution trace, homology trace, unary degree row, and intermediate resolved trace with*

$$\mathsf{UnaryHistory}(i) \wedge \mathsf{Cont}(F, R, c) \wedge \mathsf{Cont}(c, H, h).$$

The functor and resolution rows are scoped by the cited $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source surface, and the homology and degree rows are scoped by the cited* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ *source surface.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier resolution source scope`

Proof. Unpack Definition 177.1. The carrier witness is already the five-history tuple (F, R, H, i, c) together with the displayed unary and continuation rows.

The additive-functor and resolution histories are interpreted in the abelian-category dependency because the resolution is a chain-object witness over the same abelian source. Thus their carrier and classifier obligations are read through $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$.

The homology trace and the degree row are interpreted in the homology dependency. The row $\mathsf{UnaryHistory}(i)$ supplies the degree index, and the second continuation row $\mathsf{Cont}(c, H, h)$ records the homology readback from the resolved trace to the carried derived-functor history. □

Theorem 177.16 (Derived-functor classifier stability)**.** *Let* $h, h' : \mathsf{Hist}$ *be carried histories with* $\mathsf{DerivedFunctorCarrier}(h)$ *and* $\mathsf{DerivedFunctorCarrier}(h')$*. If* $h\mathsf{hsame}h'$*, then the* $\mathsf{DerivedFunctor}^{\uparrow}$ *classifier transports from* h *to* h'*: the additive-functor, resolution, homology, unary-degree, and intermediate rows of any carrier witness for* h *may be replaced by* hsame*-related rows in a carrier witness for* h'*, and the two continuation rows remain valid by* Cont *congruence.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier classifier stability`

Proof. Start with a carrier witness for h and unpack it as (F, R, H, i, c) with $\mathsf{UnaryHistory}(i)$, $\mathsf{Cont}(F, R, c)$, and $\mathsf{Cont}(c, H, h)$. Also unpack the carrier witness for h' as (F', R', H', i', c') with the corresponding unary and continuation rows.

The source certificate $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ transports the additive-functor and resolution rows along their classifier comparisons. The source certificate $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ transports the homology trace and the unary degree row, preserving the unary carrier of the degree row.

It remains to transport the two continuation rows. Apply the Cont congruence facts to the transported functor and resolution rows to obtain the resolved intermediate row, and then apply Cont congruence again to the transported intermediate and homology rows. The final endpoint is identified with h' by the displayed hypothesis $h\mathsf{hsame}h'$, so the transported tuple is again a valid $\mathsf{DerivedFunctor}^{\uparrow}$ classifier witness. □

177.2 The certificate

Theorem 177.17 (Derived-functor NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ *obligation surface consists of the carrier predicate of Definition 177.1, the classifier transported in Definition 177.16, the source-witness extraction of Definition 177.15, the exactness clause that every accepted derived-functor row unpacks into exactly those* BHist *witnesses, and the ledger clause that records only the resolution, degree, homology, and boundary rows named by the carrier.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier namecert obligation surface`

Proof. The carrier field is Definition 177.1: it is a predicate on Hist with concrete additive-functor, resolution, homology, degree, and intermediate witnesses. The source field is Definition 177.15, which assigns those witnesses to $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$.

The classifier and stability fields are exactly Definition 177.16. Transport is not a new relation; it is the restriction of hsame to the five carrier rows, with Cont congruence preserving the two displayed continuation ledgers.

Exactness is the bidirectional reading of the same data. Soundness reads each accepted row from the carrier tuple, and completeness repacks any tuple with $\mathsf{UnaryHistory}(i)$, $\mathsf{Cont}(F, R, c)$, and $\mathsf{Cont}(c, H, h)$ into the carrier. These rows reassemble the public $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ obligation surface. □

Theorem 177.18 (Derived-functor ledger policy scope)**.** *The* $\mathsf{DerivedFunctor}^{\uparrow}$ *ledger records only the chosen resolution row* R*, the unary degree row* i*, the homology row* H*, the two continuation rows* $\mathsf{Cont}(F, R, c)$ *and* $\mathsf{Cont}(c, H, h)$*, and the exactness boundary that repacks these rows into* $\mathsf{DerivedFunctorCarrier}(h)$*. It records no quotient representatives, no choice data, and no additional host presentation of derived functors.*

Proof. Read the ledger from Definition 177.1. The only hidden rows in the carrier are the resolution row, the unary degree row, the homology row, and the intermediate continuation boundary from the additive-functor trace through the resolved trace to the carried history.

Apply Definition 177.15 to identify which source certificate scopes each row. The abelian-category certificate scopes the functor and resolution source, while the homology certificate scopes the homology and unary degree source.

Finally apply Definition 177.17. Its exactness clause says that accepted rows are exhausted by this carrier tuple. Since neither the carrier nor the exactness repacking contains quotient representatives, choice data, or an ambient host presentation, the ledger cannot expose any such row. □

Definition 177.19 (Derived-functor delta-splice surface)**.** *A* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface is a displayed* BHist *packet over* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ *whose public rows consist of short-exact source rows*

$$0 \to A \to B \to C \to 0,$$

selected finite resolution rows for A*,* B*, and* C*, cycle rows* $Z_i(A)$*,* $Z_i(B)$*,* $Z_i(C)$*, boundary rows* $B_i(A)$*,* $B_i(B)$*,* $B_i(C)$*, and a connecting row*

$$\delta_i : Z_i(C)/B_i(C) \to Z_{i-1}(A)/B_{i-1}(A).$$

The packet also carries finite Cont *projection ledgers from the displayed boundary rows to* δ_i*,* hsame *transport witnesses comparing the chosen resolution and homology rows across the short-exact source row, and a* Pkg *provenance row naming the AbelianCatUp and HomologyUp certificates used to read the packet.*

Theorem 177.20 (Derived-functor short-exact source scope)**.** *The packet scope of a* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface is exactly the* $\mathsf{AbelianCat}^{\uparrow}$ *short-exact source-row data together with the* $\mathsf{Homology}^{\uparrow}$ *cycle and boundary data named in Definition 177.19. It exports no additional complex equality, no ambient abelian-category object equality, and no host quotient representative for the displayed homology rows.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorDeltaSpliceSurface short exact source scope`

Proof. Project the delta-splice packet into its named rows. The short-exact source row, the three selected resolution rows, and the admissible arrows between them are read through $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$.

The cycle rows and boundary rows are read through $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ at the displayed degrees. These rows supply the only homology data used by the packet; the notation $Z_i(-)/B_i(-)$ is the public readback of those rows, not an extra host quotient.

Conversely, given exactly these AbelianCatUp source rows and HomologyUp cycle and boundary rows, the packet is recovered by repacking the displayed Cont ledgers, hsame witnesses, and Pkg provenance. No row remains available for an ambient equality of complexes or for an external object equality, so the scope is precisely the listed source data. □

Theorem 177.21 (Derived-functor connecting row ledger)**.** *In every* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface, the connecting row* δ_i *is reached by finite* Cont *readback from the displayed HomologyUp boundary rows and transports along the listed* hsame *witnesses. Thus replacing the selected resolutions or cycle-boundary representatives by* hsame*-related rows preserves the same public connecting row.*

Lean4 ✓ `BEDC.Derived.DerivedFunctorUp.DerivedFunctorDeltaSpliceSurface connecting row ledger`

Proof. Unpack the packet and isolate the boundary rows for C and A in adjacent degrees. The finite projection ledger contains a chain of Cont rows from these displayed boundary rows to the row named δ_i.

Use Cont associativity to compose the projection rows into a single readback ledger. This produces the connecting row from the same visible boundary data and does not require a chosen host representative of a quotient class.

Now apply the listed hsame transport witnesses to the selected resolutions and to the cycle-boundary rows. The HomologyUp classifier transports boundary readback along these witnesses, and the composed Cont ledger transports with it. Repacking the transported rows gives the same public δ_i row in the delta-splice surface. □

Theorem 177.22 (Derived-functor long exact splice coverage)**.** *Adjacent degree rows in a* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface splice into the visible long exact row by* Cont *associativity and HomologyUp boundary transport. Each neighboring pair uses only the carried short-exact source row, the selected resolution rows, the displayed cycle and boundary rows, the connecting row* δ_i*, and their* Pkg *provenance.*

Proof. Fix two adjacent displayed degree rows in the packet. Project their cycle rows, boundary rows, and connecting row from Definition 177.19.

The HomologyUp boundary classifier supplies the transport from cycles modulo boundaries at degree i to the boundary data read at degree $i-1$. The connecting row ledger of Definition 177.21 identifies this transport with the finite Cont readback named δ_i.

Compose the degree-i readback, the connecting-row readback, and the degree-$(i-1)$ readback by Cont associativity. The resulting finite ledger is exactly the adjacent segment of the long exact row, and repeating the same projection for each neighboring pair covers the displayed long exact splice without adding any row beyond the packet. □

Theorem 177.23 (Derived-functor AbelianCat-Homology consumer surface)**.** *Every downstream consumer of a* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface receives only the AbelianCatUp short-exact source rows, the HomologyUp cycle and boundary rows, the connecting row* δ_i*, the finite* Cont *ledgers, the* hsame *witnesses, and the* Pkg *provenance. Any consumer row that is invariant under these projections is therefore invariant under the whole delta-splice packet.*

Proof. Project the packet to the rows listed in Definition 177.19. These are exactly the source rows identified by Definition 177.20 together with the connecting ledger identified by Definition 177.21.

For a downstream consumer, supply those projected rows as its input packet. The long exact coverage theorem then provides every adjacent splice row as a finite Cont consequence of the same input, so no consumer has access to a hidden equality of complexes, an unlisted representative, or an external abelian-category presentation.

If the consumer predicate is invariant under the projected rows, transport the input packet by the displayed hsame witnesses and repack the same Pkg provenance. The consumer output is unchanged because every visible row it can read is either one of the projections or a finite Cont composite of them. □

Theorem 177.24 (Derived-functor triangle splice consumer boundary)**.** *A downstream consumer of the* $\mathsf{DerivedFunctor}^{\uparrow}$ *triangle-splice surface receives exactly the* $\mathsf{AbelianCat}^{\uparrow}$ *short-exact source rows, the* $\mathsf{Homology}^{\uparrow}$ *cycle-boundary rows, the connecting row, the finite* Cont *ledgers, the displayed* hsame *witnesses, and the* Pkg *provenance named by Definition 177.19. No consumer projection exposes an additional complex equality, an ambient object equality, or a host quotient representative.*

Proof. Unpack the displayed delta-splice surface. Its source rows are precisely the short-exact row, the selected finite resolution rows, the cycle rows, and the boundary rows listed in Definition 177.19.

Use Definition 177.20 to identify the $\mathsf{AbelianCat}^{\uparrow}$ source part, and use Definition 177.21 to identify the connecting row as a finite Cont readback from HomologyUp boundary data and hsame transports.

Now apply Definition 177.23. It says that a consumer reads only these projections and their finite Cont consequences. Repacking those rows gives the stated boundary: BHist packet rows, hsame witnesses, finite Cont ledgers, and Pkg provenance, with no additional host equality or quotient representative. □

Theorem 177.25 (Derived-functor triangle splice provenance exactness)**.** *The* Pkg *provenance row of a* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface is complete for the displayed splice packet: it names exactly the* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ *certificates needed to read the short-exact source, cycle-boundary, connecting,* Cont*, and* hsame *rows. It carries no host quotient of complexes and no external representative for the homology notation.*

Proof. Project the Pkg row in Definition 177.19. Its AbelianCatUp component accounts for the short-exact source row and the selected finite resolution rows, while its HomologyUp component accounts for the displayed cycle and boundary rows.

The connecting row is not a separate provenance source. By Definition 177.21, it is reached from the same boundary rows by finite Cont readback and transported by the listed hsame witnesses.

The long-exact splice coverage theorem Definition 177.22 composes adjacent rows without adding a new source package. Therefore the same Pkg provenance is exact for the whole delta-splice surface and leaves host quotients of complexes outside the exported packet. □

Theorem 177.26 (Derived-functor spectral-sequence input surface)**.** *A* $\mathsf{SpectralSeq}^{\uparrow}$ *consumer of a* $\mathsf{DerivedFunctor}^{\uparrow}$ *delta-splice surface may read only adjacent long-exact splice rows produced by Definition 177.22. Its input surface consists of the neighboring cycle-boundary rows, connecting row, finite* Cont *composites,* hsame *transport witnesses, and* Pkg *provenance of those adjacent rows.*

Proof. Fix a candidate $\mathsf{SpectralSeq}^{\uparrow}$ consumer and restrict attention to two neighboring displayed degrees in the delta-splice surface. Unpack Definition 177.19 at those degrees to obtain the local cycle rows, boundary rows, and connecting row.

Apply Definition 177.22. The theorem constructs the adjacent long-exact splice segment by Cont associativity and HomologyUp boundary transport, using only the carried short-exact source row, selected resolution rows, cycle-boundary rows, connecting row, and Pkg provenance.

Finally apply Definition 177.23. Since the consumer can read only projected rows and finite Cont consequences, the allowed $\mathsf{SpectralSeq}^{\uparrow}$ input is exactly the adjacent splice surface just constructed, with no parameter transport schema or host quotient row added. □

Theorem 177.27 (DerivedFunctorUp public NameCert package)**.** *The public package exported by* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ *is exactly the* $\mathsf{AbelianCat}^{\uparrow}$ *source rows, the* $\mathsf{Homology}^{\uparrow}$ *cycle and boundary rows, the finite resolution spine, the additive-functor row, the unary degree row, the classifier transports, the finite* Cont *ledgers, and the* Pkg *provenance already isolated by the scoped theorems. It exports no host quotient of complexes and no derived-category equality.*

Proof. Begin with the scoped obligation surface of Definition 177.17. It identifies the carrier, source, classifier, exactness, and ledger rows of $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ as the finite resolution spine, additive-functor row, homology row, unary degree, two continuation ledgers, and their source certificates.

Apply Definition 177.10 to pass from the carrier packet to the consumer-facing package. This keeps exactly the $\mathsf{AbelianCat}^{\uparrow}$ source rows, $\mathsf{Homology}^{\uparrow}$ cycle and boundary rows, visible classifier transports, finite Cont consequences, and Pkg provenance.

For triangle-splice consumers, use Definition 177.24 and Definition 177.25. These two theorems show that the connecting row, adjacent splice rows, and provenance are read from the same finite packet rather than from an ambient quotient of complexes.

Finally apply Definition 177.26. Every $\mathsf{SpectralSeq}^{\uparrow}$ input row is a neighboring long-exact splice segment obtained from those same projections and finite continuation composites. Therefore the exported public package is exhausted by the listed BEDC rows and contains no host quotient of complexes or derived-category equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$**.**

Constructive story: A derived functor is generated by taking an additive-functor trace, a chosen finite resolution trace, a homology trace, a unary degree row, and two kernel continuations: first the functor trace is continued by the resolution trace to a resolved row, then that row is continued by the homology trace to the carried endpoint. The carrier admits exactly those packets whose degree is a unary history and whose resolution, homology, exact-triangle, and splice ledgers are read through the cited $\mathsf{AbelianCat}^{\uparrow}$ and $\mathsf{Homology}^{\uparrow}$ certificates; host quotients of complexes and independent equalities of abelian categories are not accepted. Two packets are classified together when the functor, resolution, homology, degree, resolved row, and endpoint transport by hsame and the two continuation displays can be repacked, so the public equality is boundary-aware homology readback over transported resolution data rather than an ambient derived-category equality. The source is the coupled abelian-category and homology certificate pair, the pattern is the finite resolution spine with cycle-boundary and long-exact splice rows, and the ledger keeps Cont transitions, connecting rows, unary degrees, and

Pkg provenance visible for SpectralSeq$^{\uparrow}$ consumers. The five name-certificate obligations follow because an admitted resolution packet gives the carrier witness, history-equality and homology-boundary comparison give reflexive, symmetric, and transitive classification, and classifier stability transports the same continuation rows into another admitted packet. A carried endpoint h lies in the signature gap of a DerivedFunctor$^{\uparrow}$ token p when the source-domain accepts the functor, resolution, homology, and degree rows, the signature relation pairs them with an answer endpoint and evidence ledger, and the token-introduction relation names that answer as p; concretely, p records the same derived homology row at the displayed unary degree over the same resolution spine. The chapter registers this coupled resolution-and-homology construction under the name DerivedFunctor$^{\uparrow}$; no derived-category calculus, quotient of complexes, or standard bridge is introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{DerivedFunctor}^{\uparrow}}$, maturePackageClosed)

Scope closed: The closed scope is the resolution-spine carrier, the AbelianCat$^{\uparrow}$ and Homology$^{\uparrow}$ source rows, the downstream consumer package, the triangle-splice consumer boundary, the exported completeness boundary, the Cont ledger, and the Pkg provenance.

Formal verification: FormalStatus(NameCert$_{\mathsf{DerivedFunctor}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DerivedFunctorUp.DerivedFunctorCarrier\_classifier\_stability`

Bridge status: paperBridge

Not claimed: Full derived-category calculus and the standard derived-functor bridge are not claimed by this chapter.

Upgrade path: The next verification-axis obligation is the machine-checked standard bridge row for the existing paper bridge.

Chapter 178

A Concrete Naming Certificate for $\mathsf{SpectralSeq}^{\uparrow}$

The spectral-sequence interface $\mathsf{SpectralSeq}^{\uparrow}$ packages a sequence of bigraded objects with differentials whose successive cohomologies converge to a target. It depends on $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ (chapter 98) and $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ (chapter 176).

178.1 Carrier definition

Definition 178.1 (SpectralSeqUp BHist page carrier)**.** *A* $\mathsf{SpectralSeq}^{\uparrow}$ BHist *page carrier is a finite page packet*

$$\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$$

over the source rows $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$*. The row A is the displayed abelian-category surface in which the page objects and differential arrows live. The finite family $E = (E_r^{p,q})$ is the page-object ledger; each $E_r^{p,q}$ is an A-classified object row. The finite family $d = (d_r^{p,q})$ is the differential ledger, with*

$$d_r^{p,q} : E_r^{p,q} \to E_r^{p+r,q-r+1}$$

recorded as an A-classified morphism row. The row h records the homology readback identifying each successor page with the $\mathsf{Homology}^{\uparrow}$ *cycle row of d_r modulo its displayed boundary row. The row c records the finite convergence and filtration rows selected for the target object. The row λ is the* Cont *ledger ordering the finite page-to-page transitions, and ρ is the* Pkg *provenance row for the displayed source certificates, page rows, differential rows, homology readbacks, convergence rows, and continuation entries.*

Definition 178.2 (SpectralSeqUp page classifier)**.** *For two* $\mathsf{SpectralSeq}^{\uparrow}$ *page carriers $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ and $\mathcal{S}' = (A', E', d', h', c', \lambda', \rho')$, the page classifier holds when the following componentwise transport rows are displayed.*

1. $A\mathsf{hsame}A'$ *as* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source surfaces.*
2. $E_r^{p,q}\mathsf{hsame}E_r'^{p,q}$ *for every carried page object.*
3. $d_r^{p,q}\mathsf{hsame}d_r'^{p,q}$ *for every differential row, with source and target read through the transported page rows.*
4. $h_r^{p,q}\mathsf{hsame}h_r'^{p,q}$ *for every* $\mathsf{Homology}^{\uparrow}$ *cycle-boundary readback row relating a differential to the successor page.*

5. $c$hsamec' *for the displayed filtration and convergence rows.*
6. $\mathsf{Cont}(\lambda, \lambda')$ *for the finite page-transition ledger.*
7. ρhsameρ' *as* Pkg *provenance rows for exactly the listed page, differential, homology, convergence, and continuation entries.*

Thus the public classifier compares the finite spectral-sequence packet by transporting the displayed rows themselves, not by importing a host exact couple or an ambient equality of abelian categories.

Lean4 (definition) `BEDC.Derived.SpectralSeqUp.SpectralSeqPageClassifier`

Theorem 178.3 (SpectralSeqUp zero-page carrier)**.** *Fix an* $\mathsf{AbelianCat}^{\uparrow}$ *object row* A_0 *and a* Pkg *provenance row* ρ_0*. The finite packet with one page object row* $E_0 = A_0$*, empty differential ledger, identity* $\mathsf{Homology}^{\uparrow}$ *readback for* E_0*, empty convergence row, empty* Cont *transition ledger, and provenance* ρ_0 *is a valid* $\mathsf{SpectralSeq}^{\uparrow}$ BHist *page carrier.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier zero page carrier`

Proof. Project the selected $\mathsf{AbelianCat}^{\uparrow}$ object row as the source surface A and as the sole page-object entry E_0. The empty differential ledger supplies no morphism obligation, so the homology readback is the identity row on the same displayed object.

The convergence and transition components are the empty finite ledgers. Repack the object row, identity readback, empty Cont ledger, and Pkg provenance as the seven components of Definition 178.1. No exact-couple or quotient row is introduced. □

Theorem 178.4 (SpectralSeqUp identity transition ledger)**.** *Let* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ *be a* $\mathsf{SpectralSeq}^{\uparrow}$ BHist *page carrier. If* λ^{id} *is obtained from* λ *by adjoining a displayed identity* Cont *row from a carried page row to itself, and* ρ^{id} *records the same provenance entry, then*

$$\mathcal{S}^{\mathrm{id}} = (A, E, d, h, c, \lambda^{\mathrm{id}}, \rho^{\mathrm{id}})$$

is again a $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier, and the page-classifier rows between* $\mathcal{S}$ *and* $\mathcal{S}^{\mathrm{id}}$ *are the original component rows together with the identity transition row.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier identity transition ledger`

Proof. Project the carrier components of $\mathcal{S}$. The source surface, page objects, differentials, homology readbacks, and convergence rows are unchanged, so their classifier transports are the original hsame rows.

The only new coordinate is the displayed identity Cont row in the finite transition ledger. Append that row to λ and append its provenance entry to ρ. Since the endpoint page is the same carried page row on both sides, the classifier clause for transitions is the identity continuation entry.

Repack the unchanged five carrier coordinates with λ^{id} and ρ^{id}. The seven clauses of Definition 178.2 are therefore preserved exactly by projection and the new identity transition row. □

Theorem 178.5 (SpectralSeqUp page classifier no-confusion)**.** *Every witness of the page classifier between* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ *and* $\mathcal{S}' = (A', E', d', h', c', \lambda', \rho')$ *decomposes into exactly the seven component transports listed in Definition 178.2: the* $\mathsf{AbelianCat}^{\uparrow}$ *source transport, page-object transports, differential transports,* $\mathsf{Homology}^{\uparrow}$ *readback transports, convergence transport, finite* Cont *ledger transport, and* Pkg *provenance transport. It contains no host exact-couple row and no ambient quotient row.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqPageClassifier no confusion`

Proof. Unfold the page classifier. A witness is a finite tuple whose coordinates are the seven displayed component transports, so projection gives the $\mathsf{AbelianCat}^{\uparrow}$ source row, page rows, differential rows, homology readbacks, convergence row, continuation ledger, and provenance row.

Conversely, repack those seven projected rows in the order of Definition 178.2. This rebuilds the original classifier witness without adding any coordinate.

No further row can be projected from the witness. In particular, neither a host exact couple nor an ambient quotient object appears in the classifier definition, so constructor inspection of the displayed tuple leaves exactly the seven transported components. □

178.2 The certificate

Theorem 178.6 (SpectralSeqUp obligation surface)**.** *Every* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier exposes exactly the source abelian-category surface, the source homology surface, the finite page-object ledger, the page classifier, the differential-square rows, the homology-page readback rows, the finite convergence rows, the* Cont *transition ledger, and the* Pkg *provenance row. These rows form the public* $\mathsf{NameCert}_{\mathsf{SpectralSeq}^{\uparrow}}$ *obligation surface available to downstream* $\mathsf{Homology}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, *and derived-category consumers.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier obligation surface`

Proof. Project the carrier of Definition 178.1. The projections give the abelian-category source row, the homology source row, the finite page ledger, the differential ledger, the homology readback row, the convergence row, the continuation ledger, and the provenance row.

The classifier of Definition 178.2 is then read component by component. It transports page objects, differential rows, homology readbacks, convergence rows, continuation entries, and provenance without adding any unlisted carrier.

Repack these projections as the named obligation surface. Since every exposed entry is a projection of the displayed packet or the classifier attached to that packet, the public certificate is exhausted by the listed rows. □

Theorem 178.7 (SpectralSeqUp differential ledger stability)**.** *If two* $\mathsf{SpectralSeq}^{\uparrow}$ *page carriers are related by the page classifier, then each transported differential row* $d_r^{p,q}$ *remains an admissible* $\mathsf{AbelianCat}^{\uparrow}$ *morphism row with the transported source and target pages, and the corresponding successor-page homology readback remains classified by the* $\mathsf{Homology}^{\uparrow}$ *source surface. Finite* Cont *extension of the page-transition ledger preserves the same property for the extended carried ledger.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier differential ledger`

Proof. Start with one differential row $d_r^{p,q} : E_r^{p,q} \to E_r^{p+r,q-r+1}$ from the carrier. The page classifier supplies hsame transport for the source page, target page, and the differential row itself, so the transported row is again typed by the transported $\mathsf{AbelianCat}^{\uparrow}$ source surface.

Project the homology readback row attached to this differential. The classifier also transports the corresponding $\mathsf{Homology}^{\uparrow}$ cycle-boundary entry, so the successor page continues to be read as the displayed homology page of the transported differential.

For a finite Cont extension, append the displayed transition row to λ and extend the provenance row by the same entry. The page packet is then repacked with all previous differential and homology rows unchanged and with the new transition recorded in the finite ledger. □

Theorem 178.8 (SpectralSeqUp convergence boundary)**.** *The* $\mathsf{SpectralSeq}^{\uparrow}$ *certificate exports only the displayed page, differential, homology-readback, filtration, convergence,* Cont*, and* Pkg *rows of Definition 178.1. It does not export a host exact couple, a quotient object outside the displayed* $\mathsf{Homology}^{\uparrow}$ *boundary rows, or an ambient equality of abelian categories beyond the transported* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source surface.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier convergence boundary`

Proof. Unpack the carrier and inspect the convergence component c. Its public content is the finite filtration and convergence ledger selected in the BHist packet, together with the page and homology rows needed to name its source.

The quotient information used by a successor page appears only through the $\mathsf{Homology}^{\uparrow}$ cycle-boundary readback row h. The theorem therefore reads no separate quotient object: it uses the displayed boundary classifier already carried by the homology source surface.

Finally, the abelian-category comparison is exactly the source-surface transport in the page classifier. Since the carrier has no row naming a host exact couple or an ambient category equality, those data cannot be exported by the public $\mathsf{NameCert}_{\mathsf{SpectralSeq}^{\uparrow}}$ boundary. □

Theorem 178.9 (SpectralSeqUp page-transition bridge)**.** *For a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier*

$$\mathcal{S} = (A, E, d, h, c, \lambda, \rho),$$

the finite Cont *ledger* λ *of Definition 178.1, transported through the page classifier of Definition 178.2, gives a* BHist*-stable bridge between each successive page row* $E_r^{p,q}$ *and the corresponding* $\mathsf{Homology}^{\uparrow}$ *readback row carried by* h*. The bridge exposes exactly the transported page-object rows, differential rows, successor homology rows,* Cont *transition entries, and* Pkg *provenance rows already listed in the spectral-sequence packet.*

Proof. Project the carrier to obtain the page ledger E, the differential ledger d, the homology readback ledger h, the transition ledger λ, and the provenance row ρ.

For each carried transition, apply the classifier clauses for the source page, target page, differential row, and homology readback row. These clauses transport the page endpoint and its successor readback componentwise through hsame while the same finite Cont entry records the page-to-page step.

Finally repack the transported page endpoint, its differential, its $\mathsf{Homology}^{\uparrow}$ readback, the relevant continuation entry, and the provenance row. Because every coordinate is projected from the original packet or from the classifier transport, the resulting bridge is stable as a BHist row and exports no extra exact-couple data. □

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier page transition bridge`

Theorem 178.10 (SpectralSeqUp convergence-filtration bridge)**.** *For* $\mathsf{SpectralSeq}^{\uparrow}$ *page carriers related by the classifier of Definition 178.2, the convergence row* c *is stable under* hsame *of the page packet and exposes exactly the finite filtration boundary needed by a derived-functor consumer. The exposed boundary consists of the transported convergence row, the page rows named by the finite filtration, the endpoint readbacks supplied by* h*, the* Cont *ledger* λ*, and the* Pkg *provenance row* ρ*.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier convergence filtration bridge`

Proof. Start with the classifier clause $c\mathsf{hsame}c'$. It transports the displayed convergence and filtration rows as rows of the same page packet rather than as ambient host objects.

Next combine that clause with the classifier clauses for λ and ρ. The continuation ledger records which finite page transitions support the filtration boundary, while the provenance row keeps those transitions tied to the same page, differential, homology, and convergence entries.

Apply Definition 178.8. It excludes host exact couples and quotient objects outside the displayed $\mathsf{Homology}^{\uparrow}$ boundary rows. The remaining exported data are precisely the finite filtration boundary rows and their Cont and Pkg witnesses. □

Theorem 178.11 (SpectralSeqUp DerivedFunctorUp consumer surface)**.** *A downstream* $\mathsf{DerivedFunctor}^{\uparrow}$ *row consuming a* $\mathsf{SpectralSeq}^{\uparrow}$ *certificate may use only the* $\mathsf{AbelianCat}^{\uparrow}$ *source row, the* $\mathsf{Homology}^{\uparrow}$ *source row, the page-object rows, the differential rows, the homology readbacks, the convergence-filtration rows, the* Cont *transition ledger, and the* Pkg *provenance listed in this chapter. It consumes no host exact couple, ambient quotient category, or equality of abelian categories outside the displayed* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source transport.*

Proof. Use Definition 178.6 to enumerate the rows exported by the spectral-sequence certificate. These are the source certificates, page objects, differentials, homology readbacks, convergence rows, continuation ledger, and provenance row.

Apply Definition 178.9 for the page-to-page bridge surface and Definition 178.10 for the convergence-facing boundary. Together they provide the transition and filtration rows needed by a derived-functor consumer.

Finally apply Definition 178.8. Since the certificate exports neither a host exact couple nor an ambient quotient category, the downstream row can consume only the displayed BEDC rows and their listed Cont and Pkg witnesses. □

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier derivedfunctor consumer surface`

Theorem 178.12 (SpectralSeq boundary rows enter successor cycles)**.** *Let* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ *be a* $\mathsf{SpectralSeq}^{\uparrow}$ BHist *page carrier. Fix a carried differential row* $d_r^{p,q}$*, and write its* $\mathsf{Homology}^{\uparrow}$ *readback on the BEDC history surface again as*

$$d_r^{p,q} : \mathsf{BHist} \to \mathsf{BHist}.$$

Assume that the displayed differential-square row for this differential is

$$\forall u : \mathsf{BHist},\ \mathsf{hsame}(d_r^{p,q}(d_r^{p,q}(u)), \mathsf{Empty}).$$

If a representative b *lies in the displayed boundary row for this differential,*

$$B_{d_r^{p,q}}(b),$$

then the successor-page readback row $h_r^{p,q}$ *classifies the same representative through the cycle row:*

$$Z_{d_r^{p,q}}(b).$$

Equivalently, every representative read by $h_r^{p,q}$ *from the displayed boundary subrow of* $d_r^{p,q}$ *enters the successor surface* $E_{r+1}^{p,q}$ *only through the cycle row used by the homology-page quotient.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier boundary row successor cycle readback`

Proof. The carrier description in Definition 178.1 gives, for the fixed differential row, a homology readback row identifying the successor page with the $\mathsf{Homology}^{\uparrow}$ cycle row of $d_r^{p,q}$ modulo its displayed boundary row. The obligation surface in Definition 178.6 exposes the differential-square row for the same displayed differential together with the matching homology-page readback row $h_r^{p,q}$.

Apply Definition 98.3 to the $\mathsf{Homology}^{\uparrow}$ differential read back from $d_r^{p,q}$ and to the representative b. The boundary hypothesis supplies the boundary-carrier premise, while the displayed differential-square row supplies the universal square-zero premise. The conclusion is exactly $Z_{d_r^{p,q}}(b)$ before taking the boundary quotient. Since $h_r^{p,q}$ reads the successor page from that cycle row modulo the displayed boundary row, the boundary representative is admitted to the successor-page surface as a cycle representative. □

Theorem 178.13 (SpectralSeqUp successor-page closure)**.** *Suppose a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier supplies a differential row* $d_r^{p,q}$*, its transported source and target page rows* $E_r^{p,q}$ *and* $E_r^{p+r,q-r+1}$*, and the displayed* $\mathsf{Homology}^{\uparrow}$ *cycle-boundary readback row identifying the successor page. Then the successor page is repacked as a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier over the same* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source, with the* Cont *page-transition ledger extended by the successor entry and the* Pkg *provenance retained for the displayed page, differential, homology-readback, convergence, and transition rows. No host quotient, host exact couple, or ambient category equality is introduced by the page transition.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier successor page closure`

Proof. Project the differential row, its source and target page rows, and the homology readback row from the carrier of Definition 178.1. These projections give the only data used to name the successor page.

Use the page classifier of Definition 178.2 to transport the source page, target page, differential, and homology readback componentwise. By Definition 178.7, the transported differential remains an admissible $\mathsf{AbelianCat}^{\uparrow}$ morphism row, and the transported readback remains classified by the displayed $\mathsf{Homology}^{\uparrow}$ source surface.

Repack the successor page by extending the finite Cont ledger with the displayed page-transition entry and retaining the same Pkg provenance rows. The boundary restriction of Definition 178.8 excludes any host quotient or exact-couple object outside the carried $\mathsf{Homology}^{\uparrow}$ cycle-boundary row, so the repacked object is again a $\mathsf{SpectralSeq}^{\uparrow}$ page carrier. □

Theorem 178.14 (SpectralSeq page-boundary exhaustion)**.** *Let* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ *be a* $\mathsf{SpectralSeq}^{\uparrow}$ BHist *page carrier. For every successor-page representative exposed by a displayed row* $h_r^{p,q}$*, the representative is exhausted by exactly the displayed page row* $E_r^{p,q}$*, differential row* $d_r^{p,q}$*,* $\mathsf{Homology}^{\uparrow}$ *cycle-boundary row,* Cont *transition entry, and* Pkg *provenance row carried by* $\mathcal{S}$*. No additional host exact couple, ambient quotient category, or uncarried page object contributes a boundary row to the successor representative.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier page boundary exhaustion`

Proof. Project the page row, differential row, homology readback row, continuation entry, and provenance entry from the carrier of Definition 178.1. These projections are the only rows on which the displayed successor readback $h_r^{p,q}$ is defined.

The classifier of Definition 178.2 transports these projected rows componentwise. Apply Definition 178.7 to keep the differential row in the carried $\mathsf{AbelianCat}^{\uparrow}$ morphism ledger, and apply Definition 178.13 to repack the transported cycle-boundary readback as the successor-page carrier.

Finally, Definition 178.8 rules out any boundary source outside the displayed $\mathsf{Homology}^{\uparrow}$ cycle-boundary row and the listed Cont and Pkg witnesses. Hence every representative exposed by $h_r^{p,q}$ is exhausted by the finite BEDC page ledger just projected. □

Theorem 178.15 (SpectralSeq visible target-filtration coverage)**.** *For a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$*, the visible target-filtration row* c *covers exactly the page rows, transition rows, homology readback rows, and convergence witnesses listed in Definition 178.1. It does not quantify over a host exact couple, an ambient quotient category, or any quotient object not named by the displayed* BHist *carrier.*

Proof. Read the target-filtration row c as one of the carried components in Definition 178.1. Its domain is therefore the same finite page inventory that supplies E, d, h, λ, and ρ.

Transport this inventory through the classifier of Definition 178.2. The differential part is constrained by Definition 178.7, while the successor-page part is constrained by Definition 178.13; both constraints remain inside the carried page rows.

The boundary statement of Definition 178.8 gives the exclusion clause. Since it admits only the displayed page, differential, homology, continuation, and provenance rows, the target-filtration row covers exactly those witnesses and no host exact-couple or ambient quotient-category data. □

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier visible target filtration coverage`

Theorem 178.16 (SpectralSeq finite continuation page-ledger closure)**.** *Let* $\mathcal{S}$ *be a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier and let* $\mathcal{L}'$ *be obtained from its finite* Cont *page ledger by adjoining finitely many displayed successor-page transition entries. If each adjoined entry is read through the page carrier, page classifier, differential ledger, convergence boundary, and successor-page closure of this section, then* $\mathcal{L}'$ *preserves the page carrier, classifier transport, and* $\mathsf{Homology}^{\uparrow}$ *readback rows of* $\mathcal{S}$*.*

Proof. Induct on the finite list of adjoined Cont entries. The empty list is the original ledger of Definition 178.1, so the carrier, classifier transport, and homology readback rows are unchanged.

For the induction step, project the next transition from the displayed page carrier and transport it through Definition 178.2. Apply Definition 178.7 to the associated differential row and Definition 178.8 to the associated boundary row.

Definition 178.13 repacks that transported entry as a successor-page carrier over the same source certificate. Appending this one entry to the induction ledger therefore preserves the carrier, classifier transport, and $\mathsf{Homology}^{\uparrow}$ readback rows; the finite induction gives the stated closure for $\mathcal{L}'$. □

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier cont page ledger closure`

Theorem 178.17 (SpectralSeqUp continuous page-transition closure)**.** *Let a* $\mathsf{SpectralSeq}^{\uparrow}$ *page carrier contain a displayed differential row* $d_r^{p,q}$*, the source and target page rows for that differential, a* $\mathsf{Homology}^{\uparrow}$ *cycle-boundary readback naming the successor page, and a finite* Cont *page-transition ledger whose next row records that readback. Then the successor page is again a* $\mathsf{SpectralSeq}^{\uparrow}$ *carrier over the same* $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ *source surface. Under the same transition, the transported differential remains an* $\mathsf{AbelianCat}^{\uparrow}$ *morphism row, the successor homology readback remains a* $\mathsf{Homology}^{\uparrow}$ *boundary row, and the finite transition ledger carries exactly the displayed page, differential, homology-readback, convergence, transition, and* Pkg *provenance rows. No ambient exact couple, host quotient equality, analytic convergence datum, or external page object is exported.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier cont page transition closure`

Proof. First read the differential surface. By Definition 178.7, the displayed row $d_r^{p,q}$ and its transported source and target rows remain within the $\mathsf{NameCert}_{\mathsf{AbelianCat}^\uparrow}$ morphism ledger selected by the page carrier. This gives the morphism component needed by a downstream derived-functor consumer.

Next form the successor carrier. Apply Definition 178.13 to the same differential row, the same source and target page rows, and the same $\mathsf{Homology}^\uparrow$ cycle-boundary readback. The resulting carrier is obtained by extending the finite Cont transition ledger with the displayed successor entry and by retaining the corresponding Pkg provenance rows.

Finally restrict the boundary of the repacked page. The exclusion clause of Definition 178.8 says that the only quotient information visible to the public page transition is the carried $\mathsf{Homology}^\uparrow$ cycle-boundary readback row. Hence the composite transition presents one closed consumer surface: successor page carrier, transported abelian morphism row, homology boundary row, finite Cont ledger, and provenance, with no additional host object. □

Definition 178.18 (SpectralSeqUp filtration ledger)**.** *For a* $\mathsf{SpectralSeq}^\uparrow$ *page carrier* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$*, the filtration ledger is the finite* BHist *packet* c *already named in Definition 178.1. It consists of the displayed filtration rows for the target object, the convergence rows relating stable page representatives to those filtration rows, the* $\mathsf{Homology}^\uparrow$ *cycle and boundary readbacks used to name each representative, and the* Pkg *provenance entries tying those rows to the carried page, differential, and transition ledgers.*

The public filtration ledger is therefore a subpacket of the carrier. It is not a host exact couple, an external quotient object, or an ambient abutment object outside the displayed BEDC rows.

Theorem 178.19 (SpectralSeqUp filtration transport)**.** *Let* $\mathcal{S} = (A, E, d, h, c, \lambda, \rho)$ *and* $\mathcal{S}' = (A', E', d', h', c', \lambda', \rho')$ *be* $\mathsf{SpectralSeq}^\uparrow$ *page carriers related by the page classifier of Definition 178.2. If* λ^+ *is obtained from* λ *by a finite* Cont *extension recording a displayed page transition, then the filtration ledger* c *transports to the filtration ledger* c' *and to the extended ledger over* λ^+*. The transport uses only the classifier rows* $c\mathsf{hsame}c'$*, the transported page, differential, and homology readback rows, and the finite* Cont *extension; it imports no host exact couple.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier filtration transport`

Proof. Project the filtration ledger c from Definition 178.18. Its entries are already part of the carrier of Definition 178.1.

Use the page classifier of Definition 178.2. It supplies $c\mathsf{hsame}c'$ and transports the page objects, differentials, and $\mathsf{Homology}^\uparrow$ cycle-boundary readback rows that name the filtration entries.

For the finite extension of λ, apply Definition 178.13. The successor step extends only the carried Cont transition ledger and retains the same Pkg provenance scope. By Definition 178.8, there is no exported exact couple or ambient quotient object to transport, so the filtration ledger remains the displayed BEDC subpacket. □

Theorem 178.20 (SpectralSeqUp abutment readback boundary)**.** *Every abutment representative exported by the filtration ledger* c *of a* $\mathsf{SpectralSeq}^\uparrow$ *page carrier is read through the displayed* $\mathsf{Homology}^\uparrow$ *cycle-boundary rows and the* Pkg *provenance row of the carrier. No representative is exported through a quotient object, exact-couple object, or abutment carrier not named in Definition 178.1.*

Lean4 ✓ `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier abutment readback boundary`

Proof. Unpack the filtration ledger using Definition 178.18. Each exported representative is attached to a convergence row in c and to the provenance entry ρ for the same carried page packet.

The representative itself is read back through the homology component h of Definition 178.1. The boundary statement Definition 178.8 identifies this as the displayed $\mathsf{Homology}^{\uparrow}$ cycle row modulo its displayed boundary row, not as a separate host quotient.

Finally apply Definition 178.13 to the page transition feeding the stable representative. The transition repacks the successor page using the carried differential, homology readback, filtration, Cont, and Pkg rows. Thus every abutment readback remains inside the displayed BEDC ledger. □

Theorem 178.21 (SpectralSeqUp page-classifier transport exactness)**.** *If two* $\mathsf{SpectralSeq}^{\uparrow}$ *page carriers are related by Definition 178.2, then every visible consumer package transported from one carrier to the other is exhausted by the source* $\mathsf{AbelianCat}^{\uparrow}$ *row, the* $\mathsf{Homology}^{\uparrow}$ *source row, the page-object rows, the differential rows, the successor-page readback rows, the convergence-filtration rows, the finite* Cont *transition ledger, and the* Pkg *provenance row already listed in the classifier. No host exact couple, quotient object outside the displayed* $\mathsf{Homology}^{\uparrow}$ *boundary, or ambient equality of abelian categories contributes a further transported row.*

Proof. Project the classifier clauses of Definition 178.2. They give the transported $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ source surface, every page-object row, every differential row with its transported source and target pages, every $\mathsf{Homology}^{\uparrow}$ cycle-boundary readback row, the filtration row, the finite Cont ledger, and the Pkg provenance row.

Next apply Definition 178.7. This keeps each differential inside the transported $\mathsf{AbelianCat}^{\uparrow}$ morphism ledger and keeps the corresponding successor readback classified by the displayed $\mathsf{Homology}^{\uparrow}$ source surface. Apply Definition 178.13 to repack the successor page over the same source certificate and the extended finite transition ledger.

Now apply Definition 178.19 and Definition 178.20. The first transports the convergence-filtration subpacket through the classifier and finite Cont extension. The second says that every abutment representative is read only through the displayed $\mathsf{Homology}^{\uparrow}$ cycle-boundary rows and the carried Pkg provenance.

Finally repack the transported rows into the consumer surface. The boundary exclusion in Definition 178.8 removes host exact couples, quotient objects outside the displayed homology boundary, and ambient equalities of abelian categories. Hence the transported package is exactly the surface named by the classifier and the four cited transport theorems. □

Theorem 178.22 (SpectralSeqUp AbCat and derived-functor scope binding)**.** *The page-carrier rows and transition rows of* $\mathsf{SpectralSeq}^{\uparrow}$ *consume only the displayed* $\mathsf{AbelianCat}^{\uparrow}$ *source rows and* $\mathsf{DerivedFunctor}^{\uparrow}$ *source rows named by Definition 178.6. In particular, a page object, differential, successor-page transition, or derived-functor consumer cannot read an ambient abelian category, an unlisted functor, or a host exact couple outside those displayed rows.*

Proof. Project the page-carrier and transition clauses exposed by Definition 178.6. The projected rows are the $\mathsf{NameCert}_{\mathsf{AbelianCat}^{\uparrow}}$ source surface, the page-object rows, the differential rows, and the finite transition ledger attached to the displayed $\mathsf{DerivedFunctor}^{\uparrow}$ source.

Next use Definition 178.11 to identify the derived-functor consumers with the same carrier and transition rows. The consumer surface has no extra field for an ambient abelian category or an unlisted functor.

Repack those projected rows as the scope witness. Since the repacked packet is literally the obligation-surface packet with no added spectral-sequence data, all page and transition consumers remain bound to the displayed $\mathsf{AbelianCat}^{\uparrow}$ and $\mathsf{DerivedFunctor}^{\uparrow}$ rows. □

Theorem 178.23 (SpectralSeqUp homology readback scope binding)**.** *The abutment rows and convergence rows of* $\mathsf{SpectralSeq}^{\uparrow}$ *factor through the displayed* $\mathsf{Homology}^{\uparrow}$ *readback rows already listed in Definition 178.6. They do not require a quotient object, convergence target, or boundary representative outside those cycle-boundary readback rows.*

Proof. Project from Definition 178.6 the homology readback rows, the successor-page rows, the convergence-filtration rows, and the abutment provenance row. These are the rows that name the visible $\mathsf{Homology}^{\uparrow}$ cycle and boundary data.

Apply Definition 178.20 to the abutment projection. It states that the abutment representative is read through the displayed cycle-boundary boundary and not through a host quotient.

Apply Definition 178.8 and Definition 178.19 to the convergence projection. They keep convergence and filtration transport inside the same displayed $\mathsf{Homology}^{\uparrow}$ readback rows. Repacking these projected rows gives the claimed scope binding. □

Theorem 178.24 (SpectralSeqUp continuation and package ledger scope binding)**.** *Every transition row, filtration row, and ledger row used by* $\mathsf{SpectralSeq}^{\uparrow}$ *is a finite* Cont*,* hsame*, or* Pkg *row from the displayed packet in Definition 178.6. No infinite transition ledger, ambient equality, or external package provenance is consumed by the chapter's certificate.*

Proof. Project the finite transition ledger, the filtration ledger, the hsame rows, and the Pkg provenance rows from Definition 178.6. These projections are already the ledger rows used by the page classifier and its transported consumer surface.

Use Definition 178.16 and Definition 178.17 to keep the transition steps inside the finite Cont ledger. Use Definition 178.21 to identify the transported package with the same hsame and Pkg rows.

Repack the projected ledger as the scope witness. Because the repacking adds no new row beyond the displayed packet, every transition, filtration, and package read is exactly one of the listed Cont, hsame, or Pkg dependencies. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralSeq}^{\uparrow}}$.

Constructive story: A spectral sequence is carried as a finite page packet whose histories name the abelian-category source, the homology source, the page-object ledger, the differential ledger, the homology readback, the convergence row, the page-transition row, and the package provenance. The carrier accepts only packets in which the displayed page and differential rows are unary histories connected by finite Cont steps and sealed by a Pkg signature, while exact couples, infinite transition ledgers, and ambient quotient objects are excluded. Two page packets are classified together by transporting each component row: the abelian source, page objects, differentials, homology readbacks, convergence data, transition ledger, and provenance must match by hsame or the listed continuation comparison, not by an outside equality of spectral sequences. The source is the paired $\mathsf{AbelianCat}^{\uparrow}$ and $\mathsf{Homology}^{\uparrow}$ certificate surface, the pattern records successive page differentials and the rule that each next page is the displayed homology of the previous differential, and the ledger keeps every finite transition, filtration, readback, and provenance entry visible. Since the empty one-page packet gives an admitted carrier and the component classifiers transport by the already checked history and package rows, the name-certificate obligations reduce to carrier habitation, reflexive/symmetric/transitive rowwise comparison, and closure under successor-page transport. A packet h lies in the signature gap of a $\mathsf{SpectralSeq}^{\uparrow}$ token p when the source-domain accepts the abelian, homology, page, differential, and convergence rows, the signature relation pairs the resulting finite packet with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records that same finite page ledger and its homology-to-successor readbacks. The chapter registers this finite page-ledger construction under the name $\mathsf{SpectralSeq}^{\uparrow}$; it supplies a scoped certificate surface, not full convergence theory or a standard spectral-sequence bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralSeq}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is the finite page-ledger construction, the displayed $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{DerivedFunctor}^{\uparrow}$, and $\mathsf{Homology}^{\uparrow}$ dependencies, the page-transition bridge of Definition 178.9, the convergence-filtration bridge of Definition 178.10, and the explicit exclusion of full convergence theory.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralSeq}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SpectralSeqUp.SpectralSeqBHistPageCarrier\_obligation\_surface`

Bridge status: paperBridge

Not claimed: Full spectral-sequence convergence theory and machine-checked standard bridge data are not closed by this chapter.

Upgrade path: The remaining elevation is machine-checked standard bridge data, not additional paper theory closure.

Chapter 179

A Concrete Naming Certificate for $\mathsf{TriangulatedCat}^{\uparrow}$

The triangulated-category interface $\mathsf{TriangulatedCat}^{\uparrow}$ packages an additive category with a translation autoequivalence and a class of distinguished triangles satisfying the octahedral axiom. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) and $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ (chapter 160).

179.1 Carrier definition

Definition 179.1 (TriangulatedCat finite carrier)**.** *A* $\mathsf{TriangulatedCat}^{\uparrow}$ *carrier is the finite* BHist *packet* $\mathsf{TriangulatedCatPkt}(\mathcal{T})$ *with the following displayed rows.*

1. *A retained* $\mathsf{Cat}^{\uparrow}$ *row for the underlying category object, morphism, identity, and composition histories.*
2. *A retained* $\mathsf{DerivedCat}^{\uparrow}$ *row for the visible localization boundary used by the triangulated presentation.*
3. *An additive-category row recording the zero object, biproduct, addition on hom histories, and additive inverse histories.*
4. *A translation-autoequivalence row recording a shift history* Σ*, its inverse shift history, and the displayed unit and counit comparison histories.*
5. *A distinguished-triangle row recording triples* $X \to Y \to Z \to \Sigma X$ *as finite morphism-history ledgers.*
6. *An octahedral ledger row recording only the named faces, comparison morphisms, and* Cont *witnesses that are explicitly displayed in the packet.*
7. *Componentwise* hsame *rows for all category, derived-category, additive, shift, triangle, and octahedral histories.*
8. Cont *rows for shift transport, triangle rotation, triangle morphism composition, and octahedral-face composition.*
9. *A* Pkg *provenance row tying the retained dependency certificates and all displayed triangulated rows to one source packet.*

The carrier accepts no row that is not one of these displayed BHist, hsame, Cont, *or* Pkg *fields.*

Lean4 (definition) `BEDC.Derived.TriangulatedCatUp.TriangulatedCatFiniteCarrier`

Definition 179.2 (TriangulatedCat classifier)**.** *Two* TriangulatedCatPkt *carriers classify together when their retained* $\mathsf{Cat}^{\uparrow}$ *and* $\mathsf{DerivedCat}^{\uparrow}$ *rows classify by the corresponding dependency certificates and every additive, translation, triangle, octahedral-ledger,* Cont*, and* Pkg *row is related by the displayed componentwise* hsame *comparison. This classifier is a comparison of finite packet rows; it is not a host localization, a Verdier quotient, or an unstated equality of triangulated categories.*

Lean4 (definition) `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketClassifier`

179.2 The certificate

Theorem 179.3 (TriangulatedCat carrier obligation)**.** *Every accepted* TriangulatedCatPkt($\mathcal{T}$) *row projects to one of the displayed dependency, additive, translation, distinguished-triangle, octahedral-ledger,* hsame*,* Cont*, or* Pkg *rows of Definition 179.1. No host Verdier quotient object and no unlogged octahedral datum is carried by the* $\mathsf{TriangulatedCat}^{\uparrow}$ *surface.*

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier carrier obligation`

Proof. Unpack the carrier definition. The packet constructor lists the retained $\mathsf{Cat}^{\uparrow}$ and $\mathsf{DerivedCat}^{\uparrow}$ rows first, then the additive, translation-autoequivalence, distinguished-triangle, octahedral-ledger, componentwise hsame, Cont, and Pkg rows.

Projecting an accepted carrier therefore selects one of those coordinates. The categorical and derived rows are read through their dependency certificates, while the triangulated-specific rows are histories inside the same finite packet.

Conversely, each displayed coordinate repacks into the same packet by the carrier constructor. Constructor inversion leaves no separate case for a host Verdier quotient, an unlabelled localization equality, or octahedral data outside the listed ledger row. □

Theorem 179.4 (TriangulatedCat classifier stability obligation)**.** *The classifier of Definition 179.2 transports the finite* $\mathsf{TriangulatedCat}^{\uparrow}$ *carrier surface: classified packets have classified dependency rows and componentwise transported additive, shift, triangle, octahedral-ledger,* Cont*, and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier classifier stability`

Proof. Begin with a classifier witness between two displayed packets. Its dependency fields are the retained classifiers for the $\mathsf{Cat}^{\uparrow}$ and $\mathsf{DerivedCat}^{\uparrow}$ rows, so those two rows transport by the existing certificate surfaces.

The remaining classifier fields are componentwise hsame comparisons. Apply them separately to the additive-category row, the shift and inverse shift rows, the unit and counit comparison rows, the distinguished-triangle ledger, and the octahedral ledger.

The Cont rows transport because their source, middle, and target histories are among the same componentwise comparisons. The provenance row transports by the displayed Pkg comparison. Repacking these transported rows with the carrier constructor gives a classified target packet and no extra semantic equality is used. □

Theorem 179.5 (TriangulatedCat translation-triangle obligation scope)**.** *Every translated distinguished-triangle readback in an accepted* $\mathsf{TriangulatedCat}^{\uparrow}$ *packet is read only through the displayed translation-autoequivalence row, inverse shift row, unit and counit comparison histories, distinguished-triangle ledger, componentwise* hsame*,* Cont *rows for shift transport and triangle rotation,* Pkg *provenance,*

and the retained $\mathsf{Cat}^{\uparrow}$ *and* $\mathsf{DerivedCat}^{\uparrow}$ *dependency rows. It does not read host Verdier quotient semantics, unlogged exact-triangle objects, or ambient triangulated-category equality.*

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier translation triangle obligation scope`

Proof. Unpack rows 20–31 of Definition 179.1. A translated distinguished triangle is represented by the displayed shift histories, inverse shift row, unit and counit comparisons, the distinguished-triangle ledger, and the associated continuation rows.

Apply Definition 179.4. The classifier transports each listed component by its hsame comparison, while the retained $\mathsf{Cat}^{\uparrow}$ and $\mathsf{DerivedCat}^{\uparrow}$ rows supply the dependency certificates and the Pkg row records provenance.

Repack the visible shift and triangle rows through the carrier constructor. Constructor inversion has no coordinate for a host Verdier quotient, an unlogged exact-triangle object, or an ambient equality of triangulated categories, so every accepted readback stays inside the displayed packet surface. □

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier translation triangle scope`

Theorem 179.6 (TriangulatedCat shift autoequivalence obligation)**.** *The shift row of a* TriangulatedCatPkt *exports only the displayed* Σ *history, inverse-shift history, unit and counit comparison histories, their* BHist *entries, componentwise* hsame *comparisons, the* Cont *rows that transport the shift route, and the* Pkg *provenance row of Definition 179.1. It does not export an equality of host autoequivalences or an unlogged inverse functor.*

Proof. Unpack the carrier of Definition 179.1. The shift coordinate is one finite row in the same packet as the additive, distinguished-triangle, octahedral-ledger, Cont, and Pkg rows.

Apply the componentwise classifier transport of Definition 179.4 to the Σ, inverse-shift, unit, and counit histories. The transported data stay inside the listed BHist and hsame coordinates.

It remains to rule out a host autoequivalence equality. Constructor inversion on the packet has cases only for the displayed shift histories, their comparison rows, the relevant Cont witnesses, and the provenance row, so no separate equality coordinate can be projected. □

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier shift autoequivalence obligation`

Theorem 179.7 (TriangulatedCat distinguished-triangle rotation obligation)**.** *Every rotation read of a distinguished triangle in a* TriangulatedCatPkt *factors through the logged triangle row, the shift obligation of Definition 179.6, and the displayed* Cont *route for triangle rotation. No rotation is read from an ambient triangulated-category structure outside the finite packet.*

Proof. Unpack the distinguished-triangle coordinate of Definition 179.1. A triangle is present as a finite morphism-history ledger ending in the displayed Σ row.

The endpoint shift is therefore the shift row governed by Definition 179.6. By Definition 179.4, classifier transport carries the triangle histories, the shift endpoint, and the rotation continuation componentwise.

Constructor inversion on the packet leaves only the triangle ledger and the listed Cont rotation route as sources for a rotation read. Hence the certificate consumes rotation through the logged finite data rather than through a host triangulated-category operation. □

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier distinguished triangle rotation obligation`

Theorem 179.8 (TriangulatedCat octahedral Cont face coverage)**.** *Every octahedral face consumed by the* $\mathsf{TriangulatedCat}^{\uparrow}$ *certificate is one of the logged finite faces of the octahedral ledger, together with its comparison-morphism history and displayed* Cont *witness. The certificate has no access path to an octahedral face outside those packet rows.*

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatOctahedralLedger face cont coverage`
↪ variant `BEDC.Derived.TriangulatedCatUp.TriangulatedCatOctahedralLedger face cont witness`

Proof. Start with the octahedral coordinate of Definition 179.1. Its accepted rows are the finite face histories, comparison morphism histories, and Cont witnesses named in the ledger.

Use Definition 179.4 to transport each face together with its comparison morphism and continuation witness. This is componentwise transport of displayed packet rows, not a semantic octahedral axiom from outside the carrier.

Finally perform constructor inversion on the finite packet. A consumed face must project from one of the logged octahedral rows, and each such row carries the comparison history and Cont witness required by the certificate. □

Theorem 179.9 (TriangulatedCat octahedral ledger boundary)**.** *The octahedral obligation in the* $\mathsf{TriangulatedCat}^{\uparrow}$ *packet is bounded by the displayed octahedral ledger: it records finite faces, comparison morphism histories, and* Cont *witnesses, and it does not assert any unrecorded octahedral filler or external triangulated-category semantics.*

Proof. Read the octahedral coordinate of Definition 179.1. Its data are the named triangle faces, the comparison morphism histories between those faces, and the finite Cont rows witnessing the displayed composites.

Any use of the octahedral obligation must project through that coordinate. Thus a face comparison or composite can be cited only when the corresponding history and continuation row appears in the ledger.

If a purported octahedral datum is not in the displayed ledger, constructor inversion gives no packet coordinate from which to project it. Therefore the chapter exposes an octahedral boundary ledger, not an ambient octahedral oracle. □

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatOctahedralLedger boundary`

Theorem 179.10 (TriangulatedCat NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{TriangulatedCat}^{\uparrow}}$ *surface determined by Definition 179.1 and Definition 179.2 consists of the finite carrier rows, the componentwise classifier, carrier transport, the displayed octahedral ledger boundary, and the* Pkg *provenance row. It excludes host Verdier quotient semantics and octahedral data not logged in the packet.*

Proof. First project the carrier by Definition 179.3. This accounts for all dependency, additive, translation, distinguished-triangle, octahedral-ledger, hsame, Cont, and Pkg rows in the public surface.

Next apply Definition 179.4. The classifier is componentwise over those same rows, and its transport clauses repack the transported histories as another accepted carrier.

Finally apply Definition 179.9 to identify the octahedral part of the certificate. Since the boundary theorem permits only logged faces, comparison morphisms, and Cont witnesses, the public certificate has no remaining source for a host Verdier quotient or unlogged octahedral datum. □

Lean4 ✓ `BEDC.Derived.TriangulatedCatUp.TriangulatedCatFiniteCarrier obligation surface`

Closure status for $\mathsf{NameCert}_{\mathsf{TriangulatedCat}^{\uparrow}}$.

Constructive story: A triangulated-category packet is generated from finite dependency rows for $\mathsf{Cat}^{\uparrow}$ and $\mathsf{DerivedCat}^{\uparrow}$, additive and translation rows, distinguished-triangle rows, the logged octahedral faces, and the Cont/Pkg provenance that makes those rows visible at the BEDC boundary.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriangulatedCat}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped boundary binds the finite BHist carrier, the additive and shift rows, the distinguished-triangle rows, the octahedral ledger, the componentwise hsame classifier, the Cont rows, the Pkg provenance row, and the dependency rows for $\mathsf{Cat}^{\uparrow}$ and $\mathsf{DerivedCat}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriangulatedCat}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TriangulatedCatUp.TriangulatedCatPacketCarrier\_classifier\_stability`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host Verdier quotient semantics, ambient triangulated-category equality, or octahedral data not logged in the displayed packet.

Upgrade path: No further paper-side theory closure upgrade remains for this chapter; formal bridge checking remains on the verification axis.

Chapter 180

A Concrete Naming Certificate for $\mathsf{HopfAlg}^{\uparrow}$

The Hopf-algebra interface $\mathsf{HopfAlg}^{\uparrow}$ packages a bialgebra carrier together with an antipode satisfying the convolution-inverse axiom. It depends on $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ (chapter 43) and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87).

180.1 Carrier definition

Definition 180.1 (HopfAlgUp BHist bialgebra carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *row* R *with multiplication ledger* μ_R *and unit ledger* η_R*, and fix a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *row* T *with tensor ledger* $\otimes_T$*. A concrete* $\mathsf{HopfAlg}^{\uparrow}$ *bialgebra carrier over* BHist *is the predicate*

$$\mathsf{HopfBialgCarrier}_{R,T}(h) \iff \exists a, t, m, c, u, e : \mathsf{BHist}.\ \mathsf{hsame}(h, a) \wedge \mathsf{RingCarrier}_R(a) \wedge \mathsf{TensorCarrier}_T(t) \wedge \mathsf{Cont}(\otimes_T, \mu_R, m$$

Here a *is the bialgebra row,* t *is the tensor-product row,* m *is the multiplication provenance row,* c *is the comultiplication provenance row,* u *is the unit ledger, and* e *is the counit ledger. The carrier is therefore anchored in* BHist *rows, with multiplication read through* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *and tensor/comultiplication provenance read through* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*.*

Definition 180.2 (HopfAlgUp antipode ledger classifier)**.** *For two concrete Hopf-algebra rows*

$$H = (a, t, s, u, e) \qquad \text{and} \qquad H' = (a', t', s', u', e')$$

over the same $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *sources, define*

$$H \sim_{\mathsf{HopfAlg}^{\uparrow}} H' \iff \mathsf{hsame}(a, a') \wedge \mathsf{hsame}(t, t') \wedge \mathsf{hsame}(s, s') \wedge \mathsf{hsame}(u, u') \wedge \mathsf{hsame}(e, e').$$

Thus the classifier compares the bialgebra row, the tensor-product row, the antipode row, the unit ledger, and the counit ledger by hsame*.*

180.2 The certificate

Theorem 180.3 (HopfAlgUp tensor-product row stability)**.** *Let* $H = (a, t, s, u, e)$ *be a concrete Hopf-algebra row satisfying Definition 180.1. If* $H \sim_{\mathsf{HopfAlg}^{\uparrow}} H'$ *with* $H' = (a', t', s', u', e')$*, then the tensor-product row of* H' *is again a valid* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *row for the transported bialgebra carrier. In particular, the multiplication and comultiplication provenance rows read by* Cont *from* t *transport to corresponding rows read from* t'*.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier tensor product row stability`
Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfBialgCarrier tensor product row stability`

Proof. Unpack the concrete carrier. The witness exposes the bialgebra row, the tensor-product row, and the two Cont ledgers that read multiplication and comultiplication provenance from the tensor source.

The classifier hypothesis supplies $t\mathsf{hsame}t'$ and $a\mathsf{hsame}a'$. Transport the tensor carrier and the two continuation ledgers along these hsame comparisons, using the hsame stability field of $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ for the tensor row and continuation transport for the displayed Cont rows.

The resulting rows have the same ring and tensor sources as the original package, with endpoints replaced by their transported representatives. Hence t' remains the tensor-product row for the transported bialgebra carrier. □

Theorem 180.4 (HopfAlgUp antipode convolution-inverse obligation)**.** *Let $H = (a, t, s, u, e)$ satisfy the concrete carrier and classifier of Definition 180.1 and Definition 180.2. The antipode row s is admissible for* $\mathsf{HopfAlg}^{\uparrow}$ *exactly when there are two* BHist *ledger rows*

$$\mathsf{ConvLeft}(a, t, s, u, e) \qquad \textit{and} \qquad \mathsf{ConvRight}(a, t, s, u, e)$$

whose Cont *provenance composes the comultiplication row, the antipode row, the multiplication row, the unit ledger, and the counit ledger so that the two convolution composites land on the unit–counit endpoint.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfBialgCarrier antipode convolution inverse obligation`

Proof. Start from the carrier witness. It gives the bialgebra row a, the tensor row t, the multiplication provenance read through $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$, the comultiplication provenance read through $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, and the unit and counit ledgers.

For the left convolution row, compose the comultiplication continuation with the tensor row that applies s on the first component and the identity ledger on the second component. Then compose with the multiplication continuation and compare the result with the unit–counit ledger endpoint.

For the right convolution row, use the same continuation spine with s placed on the second component. The two displayed rows are the two constructive obligations; no extra carrier is required beyond the BHist ledgers already unpacked from the concrete package. □

Theorem 180.5 (HopfAlgUp NameCert obligation surface)**.** *The data of Definition 180.1, Definition 180.2, Definition 180.3, and Definition 180.4 form the concrete obligation surface for a* $\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$ *over* BHist*. The surface consists of the carrier, the* hsame *classifier, tensor-row stability, multiplication and comultiplication provenance through* Cont*, and the two antipode convolution-inverse ledger rows.*

Proof. The carrier and classifier provide the object rows and equality policy. Reflexivity, symmetry, and transitivity are inherited componentwise from hsame over the bialgebra row, tensor row, antipode row, unit ledger, and counit ledger.

Tensor-row stability supplies the transport rule needed by the certificate: replacing a representative by an hsame representative preserves the tensor source and the multiplication/comultiplication provenance read from it. The ring multiplication row is retained from $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$, and the tensor row is retained from $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$.

Finally, the antipode theorem packages the two convolution-inverse rows. These rows are exactly the Hopf-algebra-specific obligations beyond the bialgebra carrier, so the listed data are the $\mathsf{NameCert}_{\mathsf{HopfAlg}^\uparrow}$ obligation surface. □

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier namecert obligation surface`

Lemma 180.6 (Two-step continuation middle cancellation)**.** *For histories* $c, m, s, s', L, L', q, q' : \mathsf{BHist}$*, assume*

$$\mathsf{Cont}(c, s, L) \wedge \mathsf{Cont}(L, m, q)$$

and

$$\mathsf{Cont}(c, s', L') \wedge \mathsf{Cont}(L', m, q').$$

If $\mathsf{hsame}(q, q')$*, then* $\mathsf{hsame}(s, s')$*.*

Proof. Apply the third cancellation clause of Definition 4.52 to the two continuation spines

$$c \xrightarrow{s} L \xrightarrow{m} q \qquad \text{and} \qquad c \xrightarrow{s'} L' \xrightarrow{m} q'.$$

The shared prefix is c, the shared suffix is m, and the endpoint comparison is the displayed $\mathsf{hsame}(q, q')$. The cancellation clause returns $\mathsf{hsame}(s, s')$. □

Theorem 180.7 (HopfAlgUp antipode uniqueness from convolution-inverse witnesses)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Ring}^\uparrow}$ *row* R *and a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^\uparrow}$ *row* T*. Let* $h : \mathsf{BHist}$ *satisfy* $\mathsf{HopfBialgCarrier}_{R,T}(h)$*, and let* $a, t, m, c, u, e : \mathsf{BHist}$ *be carrier rows exposed by Definition 180.1. Let* $s, s' : \mathsf{BHist}$ *be two antipode rows over these same carrier rows.*

Assume both s *and* s' *satisfy the full antipode obligation of Definition 180.4: each has a* ConvLeft *packet and a* ConvRight *packet landing on a unit–counit endpoint. Write the left packets as* L *and* L'*, and write their unit–counit endpoints as* q *and* q'*. Assume the endpoint packets are represented by*

$$\mathsf{Cont}(u, e, q) \qquad \textit{and} \qquad \mathsf{Cont}(u, e, q'),$$

and that the left convolution packets expose the continuation faces

$$\mathsf{Cont}(c, s, L) \wedge \mathsf{Cont}(L, m, q)$$

and

$$\mathsf{Cont}(c, s', L') \wedge \mathsf{Cont}(L', m, q').$$

Then

$$\mathsf{hsame}(s, s').$$

Consequently, for $H = (a, t, s, u, e)$ *and* $H' = (a, t, s', u, e)$*, the classifier of Definition 180.2 gives* $H \sim_{\mathsf{HopfAlg}^\uparrow} H'$*.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgAntipodeClassifier convolution inverse unique`

Proof. The two unit–counit endpoint rows have the same source u and the same tail e. Applying Definition 4.42 to $\mathsf{Cont}(u, e, q)$ and $\mathsf{Cont}(u, e, q')$, with reflexivity on u, e, and q, gives the endpoint comparison $\mathsf{hsame}(q, q')$.

Now apply Definition 180.6 to the two displayed left convolution spines. This yields $\mathsf{hsame}(s, s')$.

The right convolution packets certify that both rows are full two-sided antipode witnesses for $\mathsf{HopfAlg}^\uparrow$; uniqueness is already forced by the shared left convolution face and the common unit–counit endpoint. Since the bialgebra row, tensor row, unit ledger, and counit ledger are the same in H and H', Definition 180.2 then packages the componentwise comparison as $H \sim_{\mathsf{HopfAlg}^\uparrow} H'$. □

↪variant BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier antipode convolution inverse unique

Theorem 180.8 (HopfAlgUp antipode classifier transport)**.** *Let $H = (a, t, s, u, e)$ and $H' = (a', t', s', u', e')$ be two $\mathsf{HopfAlg}^{\uparrow}$ rows whose ring, tensor, unit, and counit sources are paired by the carrier classifier. Assume the left and right convolution packets of their antipode obligations land on unit–counit endpoints that are themselves classified. Then $\mathsf{hsame}(s, s')$, and the two five-tuples are related by the classifier of Definition 180.2.*

Proof. Start with the antipode obligation rows supplied by Definition 180.4. They expose left and right convolution packets for s and s', and by hypothesis their unit–counit endpoints are classified.

Transport the bialgebra, tensor, unit, and counit source rows along the given carrier classifier comparisons. This places the two left convolution spines in the same continuation shape, with classified source and endpoint rows.

Apply Definition 180.6 to the transported left convolution packets. The classified unit–counit endpoints give the endpoint comparison required by the cancellation lemma, so the middle antipode rows satisfy $\mathsf{hsame}(s, s')$.

The right convolution packets ensure that both middle rows are full two-sided antipode witnesses for the same transported Hopf algebra surface. Applying Definition 180.7 packages this middle-row comparison with the source comparisons.

The classifier of Definition 180.2 is exactly the five-coordinate ledger classifier for (a, t, s, u, e) and (a', t', s', u', e'), so these comparisons give $H \sim_{\mathsf{HopfAlg}^{\uparrow}} H'$. □

Lean4 ✓ BEDC.Derived.HopfAlgUp.HopfAlgAntipodeClassifier transport

Theorem 180.9 (HopfAlgUp bialgebra obligation inventory)**.** *The carrier obligations for the displayed $\mathsf{HopfAlg}^{\uparrow}$ certificate are exactly the $\mathsf{Ring}^{\uparrow}$ source row, the $\mathsf{TensorProduct}^{\uparrow}$ source row, the bialgebra row, the multiplication and comultiplication Cont rows, and the unit and counit rows carried by Definition 180.1.*

Proof. Project the carrier witness of Definition 180.1. It names the bialgebra row a, the tensor-product row t, the multiplication provenance m, the comultiplication provenance c, the unit ledger u, and the counit ledger e.

The source certificates are exactly the two rows cited by the definition: $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ supplies the ring source for multiplication and unit, and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ supplies the tensor source for tensor and comultiplication provenance. The stability theorem Definition 180.3 shows that these are the rows transported by the classifier, so no additional carrier obligation is present. □

Theorem 180.10 (HopfAlgUp antipode ledger obligation)**.** *The antipode obligations for the displayed $\mathsf{HopfAlg}^{\uparrow}$ certificate are exactly the two convolution-inverse ledger rows $\mathsf{ConvLeft}(a, t, s, u, e)$ and $\mathsf{ConvRight}(a, t, s, u, e)$ together with their unit-counit endpoint comparisons.*

Lean4 ✓ BEDC.Derived.HopfAlgUp.HopfAlgAntipodeClassifier ledger obligation

Proof. Start with Definition 180.2. The classifier places the antipode row s in the same five-coordinate ledger as the bialgebra, tensor, unit, and counit rows.

Apply Definition 180.4. It exposes precisely the left and right convolution packets and identifies their endpoint comparisons with the unit-counit ledger. The transport theorem Definition 180.8 then shows that classified antipode rows preserve those two packets, leaving no further antipode ledger obligation outside the two convolution faces. □

Theorem 180.11 (HopfAlgUp consumer boundary obligation)**.** *Any downstream consumer of the displayed* $\mathsf{HopfAlg}^{\uparrow}$ *certificate may project only the carrier row, the classifier row, tensor-row stability, antipode uniqueness and transport, and the* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *provenance rows.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier consumer boundary obligation`

Proof. The carrier and provenance rows are those of Definition 180.1. The classifier row is Definition 180.2, and tensor stability is Definition 180.3.

For antipode information, a consumer can use only the two convolution-inverse rows of Definition 180.4, the uniqueness theorem Definition 180.7, and the transport theorem Definition 180.8. These rows exhaust the public boundary: bialgebra carrier data, classifier data, tensor-row transport, antipode uniqueness and transport, and the two source-certificate provenance rows. □

Theorem 180.12 (HopfAlgUp bialgebra source consumer readback)**.** *From the displayed* $\mathsf{HopfAlg}^{\uparrow}$ *carrier witness, a downstream consumer can read exactly the* $\mathsf{Ring}^{\uparrow}$ *source row, the* $\mathsf{TensorProduct}^{\uparrow}$ *source row, the bialgebra carrier row, the multiplication and comultiplication* Cont *rows, and the unit and counit rows. These are the source rows named by the carrier and no further algebra object is exported.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier source consumer readback`

Proof. Project the carrier witness through Definition 180.1. Its source coordinates are the $\mathsf{Ring}^{\uparrow}$ and $\mathsf{TensorProduct}^{\uparrow}$ $\mathsf{NameCert}$ rows, and its operation coordinates are the multiplication, comultiplication, unit, and counit rows.

Use Definition 180.3 to read these rows through the classifier without changing the source certificates. The theorem identifies the tensor row and the bialgebra row as transported data rather than as new carrier choices.

Thus consumer readback is exhausted by the two source certificates, the bialgebra carrier, and the four operation-ledger rows. The carrier definition contains no additional public algebra row to project. □

Theorem 180.13 (HopfAlgUp antipode endpoint ledger exhaustion)**.** *The antipode obligation exposes only the two convolution packets* $\mathsf{ConvLeft}(a, t, s, u, e)$ *and* $\mathsf{ConvRight}(a, t, s, u, e)$ *together with their unit-counit endpoint comparisons. No further multiplication, comultiplication, tensor, or endpoint equality row is part of the antipode ledger.*

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgAntipodeClassifier endpoint ledger exhaustion`

Proof. Read the antipode ledger classifier from Definition 180.2. It places the antipode row inside the same endpoint ledger as the bialgebra, tensor, unit, and counit coordinates.

Apply Definition 180.10. The obligation statement exposes exactly the two convolution faces and their unit-counit endpoint comparisons.

Finally apply Definition 180.8. Transport preserves those packets and endpoint comparisons, so it cannot add an independent tensor row, operation row, or host equality witness to the antipode boundary. □

Theorem 180.14 (HopfAlgUp obligation public handoff)**.** *The public obligation handoff for downstream* $\mathsf{HopfAlg}^{\uparrow}$ *consumers is the package consisting of the bialgebra source-consumer readback of Definition 180.12, the antipode endpoint ledger exhaustion of Definition 180.13, and classifier transport through the displayed* hsame *rows.*

Proof. First collect the source and operation rows from Definition 180.12. This supplies the $\mathsf{Ring}^{\uparrow}$ row, the $\mathsf{TensorProduct}^{\uparrow}$ row, the bialgebra carrier, and the four operation-ledger rows.

Next attach the antipode rows from Definition 180.13. The only antipode data added to the handoff are the two convolution packets and their unit-counit endpoint comparisons.

Use Definition 180.8 and Definition 180.3 for classifier transport across these rows. Repacking the transported source, operation, and antipode inventories gives the public handoff surface, and the exhaustion clauses above rule out any additional consumer row. □

Closure status for $\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$.

Constructive story: A Hopf algebra is assembled from a displayed ring row and tensor-product row by continuing the tensor source into multiplication and comultiplication ledgers, then attaching unit, counit, and antipode histories to the same bialgebra endpoint. The carrier accepts exactly those BHist packages whose algebra row is carried by $\mathsf{Ring}^{\uparrow}$, whose tensor row is carried by $\mathsf{TensorProduct}^{\uparrow}$, and whose multiplication, comultiplication, unit, and counit rows are witnessed by Cont; an external algebra object or host tensor category is not part of the carrier. Two Hopf rows are classified componentwise by history-equality of the bialgebra row, tensor row, antipode row, unit ledger, and counit ledger, and antipode transport is further checked by the left and right convolution continuations landing on the same unit–counit endpoint. The source is the paired ring and tensor-product certificate surface, the pattern records bialgebra provenance plus the two convolution-inverse faces, and the ledger keeps multiplication, comultiplication, antipode, unit, counit, and package dependency rows visible. The five obligations hold because an admitted bialgebra row supplies habitation, componentwise hsame gives reflexivity, symmetry, and transitivity, and tensor-row stability plus convolution transport repack classified representatives as admitted Hopf rows. A package h lies in the signature gap of a $\mathsf{HopfAlg}^{\uparrow}$ token p when the source-domain accepts its ring and tensor rows, the signature relation pairs the bialgebra-and-antipode package with an answer package and evidence ledger, and the token-introduction relation names that answer as p; unfolded, this says that the multiplication, comultiplication, unit, counit, and two-sided antipode convolution witnesses are exactly the data named by p. The chapter registers this convolution-ledger construction under the name $\mathsf{HopfAlg}^{\uparrow}$; no standard Hopf algebra bridge, external tensor semantics, or independent antipode primitive is introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public handoff theorem exports the $\mathsf{Ring}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, bialgebra, antipode, unit, counit, BHist, Cont, hsame, and Pkg rows carried by the HopfAlgUp certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.HopfAlgUp.HopfAlgBialgCarrier\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: A standard bridge and full external Hopf-algebra theory are not claimed by this chapter.

Upgrade path: BridgeChecked status requires formal bridge-schema acceptance for the displayed HopfAlgUp surface.

Theorem 180.15 (HopfAlgUp antipode fixes the unit row)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *row* R *with multiplication ledger* μ_R *and unit ledger* η_R*, and fix a* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *row* T *with tensor ledger* $\otimes_T$*. Let* $h : \mathsf{BHist}$ *satisfy* $\mathsf{HopfBialgCarrier}_{R,T}(h)$*, and let*

$$a, t, m, c, u, e, s : \mathsf{BHist}$$

be the bialgebra row, tensor row, multiplication provenance row, comultiplication provenance row, unit ledger, counit ledger, and antipode row exposed by Definition 180.1 and Definition 180.2. Assume that s *satisfies the antipode obligation of Definition 180.4. Suppose the unit-row specialization of the left convolution packet exposes histories* $S_u, L_u, q_u : \mathsf{BHist}$ *with*

$$\mathsf{Cont}(u, s, S_u) \wedge \mathsf{Cont}(c, S_u, L_u) \wedge \mathsf{Cont}(L_u, m, q_u) \wedge \mathsf{Cont}(u, e, q_u),$$

and suppose the bialgebra unit row exposes the same unit–counit endpoint through the unit ledger:

$$\mathsf{Cont}(u, e, u).$$

Then the antipode endpoint on the unit row is classified with the unit row:

$$\mathsf{hsame}(S_u, u).$$

Consequently, if $H = (a, t, S_u, u, e)$ and $H_1 = (a, t, u, u, e)$ are the two five-coordinate Hopf rows obtained by replacing only the displayed antipode endpoint, then $H \sim_{\mathsf{HopfAlg}^\uparrow} H_1$ under the classifier of Definition 180.2.

Proof. The carrier witness supplies the unit ledger u and counit ledger e as the endpoint row of the bialgebra package. The antipode obligation supplies the two convolution-inverse packets; on the unit-row input, the displayed left packet is the continuation spine

$$u \xrightarrow{s} S_u \xrightarrow{c} L_u \xrightarrow{m} q_u,$$

with endpoint q_u classified as the unit–counit endpoint by the row $\mathsf{Cont}(u, e, q_u)$.

Compare this endpoint row with the bialgebra unit endpoint row $\mathsf{Cont}(u, e, u)$. Applying Definition 4.42 with reflexivity on the shared source u, on the shared tail e, and on the target representative gives

$$\mathsf{hsame}(q_u, u).$$

Thus the specialized convolution composite lands on the same endpoint as the unit row.

Now read the specialized convolution spine against the unit endpoint. The bialgebra unit specialization says that, on the unit input, the comultiplication and multiplication faces contribute no independent endpoint beyond the shared unit–counit row: the left convolution packet has the same outer source and terminal endpoint as the displayed unit row. Cancelling the common convolution context by the third clause of Definition 4.52, equivalently by the two-step cancellation form of Definition 180.6 after aligning the endpoint by $\mathsf{hsame}(q_u, u)$, identifies the middle endpoint selected by the antipode with the unit endpoint. Hence

$$\mathsf{hsame}(S_u, u).$$

For the final claim, the two Hopf rows have identical bialgebra row a, tensor row t, unit ledger u, and counit ledger e, and the antipode coordinates are related by the comparison just proved. The five-coordinate classifier in Definition 180.2 therefore gives $H \sim_{\mathsf{HopfAlg}^\uparrow} H_1$. □

Lean4 ✓ `BEDC.Derived.HopfAlgUp.HopfAlgAntipodeClassifier unit row fixed`

Chapter 181

A Concrete Naming Certificate for $\mathsf{MonoidalCat}^{\uparrow}$

The monoidal-category interface $\mathsf{MonoidalCat}^{\uparrow}$ packages a category with a tensor-product bifunctor, unit object, and coherent associator and unitor isomorphisms. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61).

181.1 Carrier definition

181.2 The certificate

Definition 181.1 (MonoidalCat singleton tensor instance)**.** *The singleton* BHist *monoidal-category instance uses the unary-continuation category of Definition 61.325 as its underlying* $\mathsf{Cat}^{\uparrow}$ *carrier. Its objects are carried unary histories, its displayed hom rows are* $\mathsf{CategoryHomCarrier}(a, b, f)$ *rows, and its tensor is continuation append:*

$$a \otimes b := \mathsf{append}(a, b), \qquad f \otimes g := \mathsf{append}(f, g).$$

The unit object is e*. The associator row from* $(x \otimes y) \otimes z$ *to* $x \otimes (y \otimes z)$ *and the left and right unitor rows from* $\mathsf{e} \otimes x$ *and* $x \otimes \mathsf{e}$ *to* x *are the* hsame *witnesses supplied by continuation associativity and the empty-continuation unit laws on the displayed* BHist *rows.*

Lean4 (definition) `BEDC.Derived.MonoidalCatUp.MonoidalCatSingletonTensor`

Theorem 181.2 (MonoidalCat singleton tensor carrier)**.** *In the singleton tensor instance of Definition 181.1, tensoring carried unary objects remains a carried unary object. If*

$$\mathsf{CategoryHomCarrier}(a, b, f)$$

and

$$\mathsf{CategoryHomCarrier}(c, d, g)$$

are displayed hom histories, then the tensor hom history $\mathsf{append}(f, g)$ *is carried by the same unary-continuation hom surface from* $\mathsf{append}(a, c)$ *to* $\mathsf{append}(b, d)$*.*

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton tensor carrier`

Proof. Unary object closure is the ordinary append closure for unary BHist rows.

For hom rows, first use the endpoint stability law package of Definition 61.326 to read both displayed morphisms as continuation tails over their endpoints. Then apply Definition 61.19 to the compatible tail rows after appending the source and target endpoints.

The resulting composite row has exactly the source $\mathsf{append}(a, c)$, target $\mathsf{append}(b, d)$, and displayed morphism $\mathsf{append}(f, g)$ required by the tensor carrier surface. □

Theorem 181.3 (MonoidalCat singleton tensor preserves identity homs)**.** *In the singleton tensor instance of Definition 181.1, let a and b be carried unary objects. The tensor of the identity hom histories $\mathsf{e} : a \to a$ and $\mathsf{e} : b \to b$ is classified as the identity hom on $a \otimes b$. Concretely,* $\mathsf{append}(\mathsf{e}, \mathsf{e})$ *is* hsame *to* e*, and the row gives*

$$\mathsf{CategoryHomCarrier}(\mathsf{append}(a, b), \mathsf{append}(a, b), \mathsf{e}).$$

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton tensor identity hom`

Proof. The identity rows for a and b are the empty continuation rows supplied by Definition 61.325. Unary append closure gives the tensor object $\mathsf{append}(a, b)$ as a carried unary object.

Tensoring the two displayed identity histories gives $\mathsf{append}(\mathsf{e}, \mathsf{e})$. The empty-continuation unit laws of Definition 40.3 identify this displayed history with e by hsame.

Transport the identity row on $\mathsf{append}(a, b)$ along that hsame classifier row. The resulting displayed hom carrier is exactly $\mathsf{CategoryHomCarrier}(\mathsf{append}(a, b), \mathsf{append}(a, b), \mathsf{e})$, so the tensor preserves identity hom histories on the concrete singleton carrier. □

Theorem 181.4 (MonoidalCat singleton associator and unitors)**.** *For carried unary histories x, y, and z in the singleton tensor instance, the displayed histories $(x \otimes y) \otimes z$ and $x \otimes (y \otimes z)$ are related by* hsame*. The displayed histories $\mathsf{e} \otimes x$ and $x \otimes \mathsf{e}$ are each related to x by* hsame *on the same concrete* BHist *carrier.*

Proof. The associator row is continuation associativity applied to the three visible tails x, y, and z. Both bracketings reduce to the same iterated continuation history, so the classifier row is the corresponding hsame witness.

For the left unitor, use the left empty-continuation law to identify $\mathsf{append}(\mathsf{e}, x)$ with x. For the right unitor, use the right empty-continuation law to identify $\mathsf{append}(x, \mathsf{e})$ with x.

The category continuation-morphism tail theorems transport these three hsame rows through the displayed hom carrier, so the associator and both unitors live on the concrete unary-continuation category rather than on an external categorical carrier. □

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton associator unitors`

Theorem 181.5 (MonoidalCat singleton coherence laws)**.** *In the singleton tensor instance, the pentagon and triangle diagrams commute under the morphism classifier. For any carried unary histories appearing in either diagram, the two displayed paths have* hsame*-equal classifier rows because both paths reduce to the same iterated* Cont *tail.*

Proof. For the pentagon, expand each edge into the associator row of Definition 181.4. The two boundary paths are the two standard ways to reassociate a fourfold continuation tail, and the category fourfold associativity witness Definition 61.131 identifies their displayed histories by hsame.

For the triangle, expand the associator and the two unitor rows. The left path and the right path both normalize to the same twofold continuation tail after applying the empty-continuation unit laws.

Finally compose the classifier rows by transitivity of hsame. The displayed hom carrier is preserved along the comparison by the category stability law package, so the commuting diagrams are certified inside the BEDC morphism classifier. □

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton coherence laws`

Theorem 181.6 (MonoidalCat NameCert obligation surface)**.** *The singleton tensor instance assembles the following paper-side* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *obligation surface:*

1. *Traditional target: a monoidal category.*
2. *BEDC source:* $\mathsf{Cat}^{\uparrow}$, Cont, hsame, *and* CategoryHomCarrier.
3. *Carrier: unary* BHist *objects and continuation hom histories.*
4. *Classifier:* hsame *on displayed object and hom histories.*
5. *Stability: tensor carrier closure, identity-hom preservation, composition preservation, associator rows, unitor rows, pentagon coherence, and triangle coherence.*
6. *Ledger: tensor tails and the two coherence-path ledgers.*
7. *Core theorem list: Definition 181.2, Definition 181.3, Definition 181.7, Definition 181.4, Definition 181.5, and Definition 61.325.*
8. *Exactness: classifier equality of the displayed coherence paths.*
9. *Standard bridge: absent.*
10. *Status: paper-side obligation surface.*

Proof. The carrier and classifier fields are supplied by the unary-continuation category and its CategoryHomCarrier rows. The tensor field is the append operation in Definition 181.1.

The stability fields are discharged by Definition 181.2, Definition 181.3, Definition 181.7, and Definition 181.4. The coherence fields are discharged by Definition 181.5.

The ledger records only the displayed continuation tails and the two coherence paths. Exactness is the statement that the two paths in each coherence diagram are equal under hsame, and no standard external monoidal-category bridge is asserted here. □

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton semantic name certificate`

Theorem 181.7 (MonoidalCat singleton tensor preserves composition)**.** *In the singleton tensor instance of Definition 181.1, suppose*

$$\mathsf{CategoryHomCarrier}(a_0, a_1, f), \quad \mathsf{CategoryHomCarrier}(a_1, a_2, f'), \quad \mathsf{CategoryHomCarrier}(c_0, c_1, g), \quad \mathsf{CategoryHomCa}$$

are two composable pairs of displayed hom rows. Then the tensor composite and the tensor of component composites are both displayed hom rows from $\mathsf{append}(a_0, c_0)$ *to* $\mathsf{append}(a_2, c_2)$*, and their morphism classifiers agree:*

$$\begin{aligned}&\mathsf{CategoryHomCarrier}(\mathsf{append}(a_0, c_0), \mathsf{append}(a_2, c_2), \mathsf{append}(\mathsf{append}(f, g), \mathsf{append}(f', g'))),\\ &\mathsf{CategoryHomCarrier}(\mathsf{append}(a_0, c_0), \mathsf{append}(a_2, c_2), \mathsf{append}(\mathsf{append}(f, f'), \mathsf{append}(g, g'))),\\ &\quad \mathsf{hsame}(\mathsf{append}(\mathsf{append}(f, g), \mathsf{append}(f', g')), \mathsf{append}(\mathsf{append}(f, f'), \mathsf{append}(g, g'))).\end{aligned}$$

Equivalently, $(f \otimes g)$ *followed by* $(f' \otimes g')$ *is classifier-equal to* $(f' \circ f) \otimes (g' \circ g)$ *in the concrete singleton* $\mathsf{MonoidalCat}^{\uparrow}$ *certificate.*

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton tensor preserves composition`

Proof. By Definition 181.2, the tensor rows $\mathsf{append}(f, g)$ and $\mathsf{append}(f', g')$ are hom-carriers from $\mathsf{append}(a_0, c_0)$ to $\mathsf{append}(a_1, c_1)$ and from $\mathsf{append}(a_1, c_1)$ to $\mathsf{append}(a_2, c_2)$. Applying Definition 61.19 to these two tensor rows gives the first displayed carrier.

Applying Definition 61.19 to the pair f, f' gives the component composite $\mathsf{append}(f, f')$ from a_0 to a_2, and applying it to g, g' gives $\mathsf{append}(g, g')$ from c_0 to c_2. A second use of Definition 181.2 gives the second displayed carrier.

It remains to compare the displayed morphism histories. The four input hom-carrier rows expose unary membership of f, f', g, and g'. Hence Definition 9200.14, applied to the canonical append continuations for g followed by f' and for f' followed by g, gives

$$\mathsf{hsame}(\mathsf{append}(g, f'), \mathsf{append}(f', g)).$$

Repeated use of Definition 4.56 identifies the tensor composite with

$$\mathsf{append}(f, \mathsf{append}(\mathsf{append}(g, f'), g'))$$

and identifies the tensor of component composites with

$$\mathsf{append}(f, \mathsf{append}(\mathsf{append}(f', g), g')).$$

The reverse direction of Definition 4.51, with common left context f and common right context g', transports the middle comparison into the two displayed fourfold histories. Transitivity of hsame with the two associativity comparisons gives the required classifier equality. □

Theorem 181.8 (MonoidalCat singleton standard bridge)**.** *The singleton continuation tensor package of Definition 181.1, together with the tensor carrier row of Definition 181.2, the associator and unitor rows of Definition 181.4, the coherence rows of Definition 181.5, the semantic certificate of Definition 181.6, and the tensor composition row of Definition 181.7, determines the standard bridge rows for* $\mathsf{MonoidalCat}^{\uparrow}$ *on the singleton BEDC surface.*

The bridge reads objects as unary-history rows, morphisms as displayed unary-history hom rows, tensor as Cont *append, the unit as* e*, and equality as* hsame*. It exports exactly this singleton continuation package and no non-singleton monoidal category, host bifunctor, external object carrier, or ambient categorical coherence data.*

Proof. The carrier theorem supplies the object and morphism rows: carried unary histories remain carried after tensor append, and displayed hom histories tensor by appending their Cont tails.

The associator and unitor theorem supplies the structural comparison rows. The coherence theorem identifies the pentagon and triangle boundary paths by hsame, so the bridge has the standard coherence entries required for the singleton tensor surface.

The semantic certificate records the visible $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ carrier, classifier, stability, ledger, and exactness rows. The tensor-preserves-composition theorem supplies the bifunctorial composition row on the same displayed histories, while the identity-hom theorem supplies the identity row.

Thus every bridge entry is a row already present in the singleton continuation package: objects and morphisms are unary-history rows, tensor is Cont append, unit is e, and equality is hsame. Because no carrier coordinate names non-singleton objects or a host bifunctor, the bridge exports none of that data. □

Lean4 ✓ `BEDC.Derived.MonoidalCatUp.MonoidalCatUp StdBridge`

Theorem 181.9 (MonoidalCat singleton tensor consumer coverage)**.** *Every consumer read of the singleton tensor bridge of Definition 181.8 is one of the unary object rows, unary hom rows,* Cont *tensor rows,* e *unit rows,* hsame *associator, unitor, and coherence rows, identity-hom rows, composition-preservation rows, or the semantic* NameCert *ledger already listed for* $\mathsf{MonoidalCat}^{\uparrow}$*.*

The covered surface exports exactly the singleton continuation tensor package and no non-singleton object carrier, host bifunctor, external coherence object, or ambient monoidal-category structure.

Proof. Use Definition 181.2 for the unary object and hom rows and for tensor as continuation append. The identity-hom reads are the rows of Definition 181.3.

Use Definition 181.4 and Definition 181.5 for the associator, unitor, pentagon, and triangle rows. These rows are hsame comparisons over the same unary-history surface.

Use Definition 181.7 for the composition-preservation readers and Definition 181.6 for the semantic certificate ledger. The standard bridge of Definition 181.8 collects exactly these rows, so the listed families exhaust all consumer reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$**.**

Constructive story: A monoidal category at this seed level is the unary-continuation category with tensor given by appending the underlying histories: carried unary objects tensor to append(a, b), carried hom histories tensor to append(f, g), and the unit object is e. The carrier therefore accepts only the singleton tensor instance whose objects and morphisms are already admitted by the $\mathsf{Cat}^{\uparrow}$ continuation surface, while non-singleton monoidal categories and host bifunctor data remain outside the seed. Two displayed tensor rows are classified by the same history-equality used for the category object and hom rows; associator and unitor comparisons are the hsame witnesses supplied by continuation associativity and the empty-continuation laws, not an imported coherence theorem. The source is $\mathsf{Cat}^{\uparrow}$ over unary histories, the pattern records tensor closure, identity-hom preservation, composition preservation, associator, unitors, pentagon, and triangle rows, and the ledger keeps the tensor tails and the two coherence-path ledgers visible. Because e is carried, unary histories are closed under append, and hsame is an equivalence transported through the hom carrier, the five name-certificate obligations are obtained by the same componentwise carrier witness, reflexive/symmetric/transitive classifier, and closure under classified tensor representatives. A row h lies in the signature gap of a $\mathsf{MonoidalCat}^{\uparrow}$ token p when the source-domain accepts the unary category and tensor ledgers, the signature relation pairs the singleton tensor package with an answer package and evidence ledger, and the token-introduction relation names that answer as p; concretely, p encodes the continuation tensor together with its associator, unitor, and coherence rows. The chapter registers this singleton continuation tensor under the name $\mathsf{MonoidalCat}^{\uparrow}$ with a paper-side standard bridge whose rows are precisely unary objects, unary hom histories, Cont append tensor, e unit, and hsame equality; it does not register a general monoidal-category carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature surface is the singleton standard bridge for $\mathsf{MonoidalCat}^{\uparrow}$: tensor carrier, identity-hom, tensor-composition, associator, unitor, coherence, semantic certificate, and singleton bridge rows over the CategoryUp continuation surface and the kernel Cont and hsame dependencies used by those rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.MonoidalCatUp.MonoidalCatSingleton\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close an external standard model, host bifunctor data, or non-singleton monoidal categories.

Upgrade path: No further formal-axis upgrade is pending for the checked singleton continuation-tensor bridge.

Chapter 182

A Concrete Naming Certificate for $\mathsf{EnrichedCat}^{\uparrow}$

The enriched-category interface $\mathsf{EnrichedCat}^{\uparrow}$ packages a collection of objects with hom-objects valued in a fixed monoidal category, with composition and identity morphisms in that monoidal category. It depends on $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ (chapter 181) and $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61).

182.1 Carrier definition

Definition 182.1 (EnrichedCat BHist packet)**.** *An* $\mathsf{EnrichedCat}^{\uparrow}$ *carrier packet is the finite* BHist *tuple*

$$\mathcal{E} = (C_{\mathrm{obj}}, M_{\otimes}, H_{\mathrm{hom}}, H_{\mathrm{id}}, H_{\mathrm{comp}}, T_{\mathsf{hsame}}, P_{\mathsf{Pkg}}, R_{\mathsf{Cont}})$$

whose coordinates are, in order, the consumed $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *object boundary* C_{obj}*, the consumed* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *tensor boundary* $M_{\otimes}$*, the hom-object rows* H_{hom}*, the identity rows* H_{id}*, the composition rows* H_{comp}*, the componentwise* hsame *transport rows* T_{hsame}*, the* Pkg *provenance* P_{Pkg}*, and the finite* Cont *readbacks* R_{Cont}*.*

The carrier accepts exactly those packets for which the two consumed boundary coordinates are public certificate rows, every hom-object, identity, and composition coordinate is a displayed BHist *row over those boundaries, every transport coordinate is an* hsame *row between displayed coordinates, the provenance coordinate is a* Pkg *row for the same packet, and the readback coordinate is a finite* Cont *ledger for those rows.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatTasteGate single carrier alignment`

Theorem 182.2 (EnrichedCat hom-object carrier obligation)**.** *Every hom-object row used by the* $\mathsf{EnrichedCat}^{\uparrow}$ *certificate is a coordinate of the packet in Definition 182.1, with its object boundary read from the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *coordinate, its tensor boundary read from the consumed* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *coordinate, and its visible provenance and continuation reads carried by the same* Pkg *and* Cont *coordinates. No hom-object obligation is supplied by an implicit enriched-category carrier outside the displayed packet.*

Proof. Project the packet of Definition 182.1 to the H_{hom} coordinate. Its source and target object rows are read from C_{obj}, while its value object lies under the tensor boundary $M_{\otimes}$.

The visible provenance is the coordinate P_{Pkg}, and the permitted readbacks are exactly the finite routes in R_{Cont}. These are the same rows enumerated by Definition 182.4.

Repack the projected hom-object row with the same boundary, provenance, and readback coordinates. Classifier transport is supplied componentwise by Definition 182.5, and ledger exactness is supplied by Definition 182.6. □

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatSourceSurface hom object carrier obligation`

Theorem 182.3 (EnrichedCat tensor-composition scope)**.** *Every identity or composition row used by the* $\mathsf{EnrichedCat}^{\uparrow}$ *certificate is scoped by the packet in Definition 182.1: identities read the hom-object and tensor-unit rows, compositions read the two adjacent hom-object rows and the tensor-composition boundary, and all coherence, transport, provenance, and finite readback data are supplied by* T_{hsame}*,* P_{Pkg}*, and* R_{Cont}*. The scope does not include unlisted host-level enriched-category data.*

Proof. Project first to the identity coordinate H_{id}. The identity row uses the object boundary from C_{obj}, the hom-object row from H_{hom}, and the unit part of $M_{\otimes}$.

Project next to the composition coordinate H_{comp}. Its inputs are the adjacent hom-object rows and the tensor-composition part of $M_{\otimes}$, so the monoidal dependency is exactly the one described in Definition 182.7.

The component transport rows are then read from T_{hsame}, while provenance and finite ledgers are read from P_{Pkg} and R_{Cont}. Repacking these same coordinates gives the public certificate rows by Definition 182.4 and Definition 182.6. □

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatSourceSurface tensor composition scope`

182.2 The certificate

Theorem 182.4 (EnrichedCatUp source obligation)**.** *Every public source row of an* $\mathsf{EnrichedCat}^{\uparrow}$ *packet is one of the displayed* BHist *rows for the consumed* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *object boundary, the consumed* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *tensor boundary, the hom-object rows, the identity rows, the composition rows, the* hsame *transport rows, the* Pkg *provenance rows, and the finite* Cont *readback rows.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatCertificateSurface source obligation`

Proof. Unpack the packet as the named BHist tuple. The first projections are the two consumed certificates: the object boundary is the public boundary of $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$, and the tensor boundary is the public boundary of $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$.

The remaining projections are the enriched rows themselves. They record the hom-object assignment, the identity witnesses, composition witnesses, component hsame transports, package provenance, and the finite continuation readbacks.

Conversely, each displayed row is inserted into the packet by the corresponding constructor coordinate. Projection followed by repacking is judgmentally the same named tuple, so no further public source row can appear. □

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatSourceSurface source obligation`

Theorem 182.5 (EnrichedCatUp classifier transport obligation)**.** *The public classifier comparison for an* $\mathsf{EnrichedCat}^{\uparrow}$ *packet is the componentwise* hsame *transport of the object-boundary, tensor-boundary, hom-object, identity, composition, provenance, and continuation rows, and is not host equality of enriched categories.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatPublicRows classifier transport obligation`

Proof. Project the classifier comparison to each displayed row. On the consumed $\mathsf{NameCert}_{\mathsf{Cat}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{MonoidalCat}^\uparrow}$ boundaries, the comparison is exactly the already exported boundary hsame comparison.

For the enriched rows, transport is pointwise: hom-objects transport along the object boundary, identity and composition rows transport along the tensor boundary, and provenance and continuation rows transport by their named BHist coordinates.

Repack the transported coordinates into the public classifier row. The construction never compares host-level enriched categories; it compares only the displayed BHist rows and the hsame witnesses attached to them. □

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatSourceSurface classifier transport obligation`
Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatPublicPacket classifier transport obligation`

Theorem 182.6 (EnrichedCatUp ledger exactness obligation)**.** *Projection from an* $\mathsf{EnrichedCat}^\uparrow$ *certificate to its hom-object, identity, composition, tensor, provenance, and continuation ledger rows, followed by repacking those same rows in the displayed order, recovers exactly the public certificate ledger.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatPublicPacket ledger exactness obligation`

Proof. Start with the public certificate ledger and project the hom-object, identity, composition, tensor, Pkg, and Cont coordinates. These projections are the named coordinates of the certificate tuple.

Repacking uses those coordinates in the same order and with the same boundary certificates. The object and tensor rows are not recomputed; they are the consumed rows already present in the tuple.

Therefore the repacked ledger has the same projections as the original public ledger on every displayed coordinate. Since the certificate exposes no other ledger coordinate, the projection-and-repacking operation is exact. □

Theorem 182.7 (EnrichedCatUp monoidal-category boundary)**.** *The* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^\uparrow}$ *dependency of an* $\mathsf{EnrichedCat}^\uparrow$ *certificate is read only through tensor, unit, associator and unit-transport, and* Pkg *rows; no other monoidal-category host data is part of the public obligation surface.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatPublicPacket monoidal category boundary`

Proof. Inspect the monoidal dependency coordinates in the certificate tuple. The tensor and unit rows provide the objects used by the hom-object, identity, and composition entries.

The associator and unit-transport rows are the only coherence transports read by the enriched composition and identity rows. Provenance is recorded through the displayed Pkg coordinate attached to the consumed $\mathsf{NameCert}_{\mathsf{MonoidalCat}^\uparrow}$ packet.

All public reads of the monoidal dependency factor through these rows. Hence any two monoidal witnesses with the same displayed tensor, unit, coherence transport, and provenance rows are indistinguishable to the $\mathsf{EnrichedCat}^\uparrow$ certificate. □

Theorem 182.8 (EnrichedCatUp consumer readback boundary)**.** *A downstream* $\mathsf{Functor}^\uparrow$ *or* $\mathsf{NatTrans}^\uparrow$ *consumer of an* $\mathsf{EnrichedCat}^\uparrow$ *certificate reads only the displayed object, hom-object, identity, composition, transport, provenance, and finite* Cont *rows.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatSourceSurface consumer readback boundary`

Proof. Give the consumer its public readback map. By the source obligation, the map can project only from the displayed BHist tuple and its finite Cont readback rows.

The object and hom-object rows supply the source and target data for enriched functors. The identity, composition, and transport rows supply the compatibility witnesses read by functor and natural-transformation certificates.

Package provenance is available only through the Pkg rows, and finite continuations are available only through the listed Cont rows. Repacking these rows gives the same public readback packet, so the consumer boundary is exactly the displayed surface. □

Theorem 182.9 (EnrichedCat public packet source boundary)**.** *For an accepted* $\mathsf{EnrichedCat}^{\uparrow}$ *carrier, the public packet exposes exactly the consumed* $\mathsf{Cat}^{\uparrow}$ *object boundary, the consumed* $\mathsf{MonoidalCat}^{\uparrow}$ *tensor boundary, the hom-object rows, identity rows, composition rows, componentwise* hsame *transports,* Pkg *provenance, and finite* Cont *readbacks of Definition 182.1. Every* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer factors its public read through these rows and through no additional host enriched-category datum.*

Lean4 ✓ `BEDC.Derived.EnrichedCatUp.EnrichedCatPublicPacket source boundary`

Proof. Start from the packet of Definition 182.1. Its first two coordinates are the consumed $\mathsf{Cat}^{\uparrow}$ and $\mathsf{MonoidalCat}^{\uparrow}$ boundaries, so the available source rows are constrained by Definition 182.4.

The remaining coordinates are the enriched rows themselves. Classifier movement is componentwise by Definition 182.5, and the monoidal dependency is the displayed tensor boundary isolated by Definition 182.7.

Finally apply Definition 182.8. It already identifies the $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ readback surface with the object, hom-object, identity, composition, transport, provenance, and finite continuation rows. Repacking those projections gives the same public packet, so every such consumer factors through exactly the listed boundary. □

Theorem 182.10 (EnrichedCat two-step composition reassociation)**.** *Let*

$$\mathcal{E} = (C_{\mathrm{obj}}, M_{\otimes}, H_{\mathrm{hom}}, H_{\mathrm{id}}, H_{\mathrm{comp}}, T_{\mathsf{hsame}}, P_{\mathsf{Pkg}}, R_{\mathsf{Cont}})$$

be an accepted $\mathsf{EnrichedCat}^{\uparrow}$ *carrier packet. Suppose* H_{hom} *contains three composable displayed hom-object rows*

$$H_{ab}, \quad H_{bc}, \quad H_{cd},$$

and suppose H_{comp} *contains the displayed composition rows needed for both two-step routes from these three rows to endpoint rows* L_{ad} *and* R_{ad}*. Thus the left route first composes* H_{cd} *with* H_{bc} *and then composes the result with* H_{ab}*, while the right route first composes* H_{bc} *with* H_{ab} *and then composes* H_{cd} *with the result.*

If the consumed $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *tensor boundary* $M_{\otimes}$ *supplies the associator row comparing the two tensor sources*

$$(H_{cd} \otimes H_{bc}) \otimes H_{ab} \quad \text{and} \quad H_{cd} \otimes (H_{bc} \otimes H_{ab}),$$

then L_{ad} *and* R_{ad} *are classified by the public* $\mathsf{EnrichedCat}^{\uparrow}$ *packet surface. The classifying comparison is a displayed* hsame *row read through* T_{hsame}*, with provenance in* P_{Pkg} *and finite readback in* R_{Cont}*; it uses no host-level enriched-category equality and no coherence datum outside the consumed* $\mathsf{MonoidalCat}^{\uparrow}$ *boundary.*

Proof. Unpack the accepted packet by Definition 182.1. The rows H_{ab}, H_{bc}, and H_{cd} are projected from H_{hom}, so their object boundary is read from C_{obj} and their tensor boundary is read from $M_{\otimes}$.

By Definition 182.3, each displayed composition row in the two routes is projected from H_{comp}. The left route uses the adjacent pair H_{cd}, H_{bc} and then the adjacent pair represented by that first composite together with H_{ab}. The right route uses the adjacent pair H_{bc}, H_{ab} and then the adjacent pair H_{cd} together with that first composite. In both cases the tensor-composition boundary is the same coordinate $M_{\otimes}$, and the route remains inside the displayed hom-object and composition rows of $\mathcal{E}$.

The only rebracketing needed to compare the two tensor sources is the associator row already exposed by the consumed $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ coordinate. By Definition 182.7, this associator is one of the public monoidal rows available to the enriched packet, not an additional host coherence theorem.

Transport the two-step composition source along that associator row. The component transport coordinate T_{hsame} carries the transported tensor source through the two displayed composition routes, so the resulting endpoint rows L_{ad} and R_{ad} are compared by a displayed hsame row. The provenance and finite continuation reads of this comparison are exactly the rows P_{Pkg} and R_{Cont} attached to the same packet.

Finally apply Definition 182.4 and Definition 182.5. They identify every public source and classifier row of the $\mathsf{EnrichedCat}^{\uparrow}$ certificate with one of the displayed BHist, hsame, Pkg, and Cont coordinates of $\mathcal{E}$. Repacking the transported endpoint comparison through those coordinates gives the public comparison between L_{ad} and R_{ad} on the same packet surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{EnrichedCat}^{\uparrow}}$.

Constructive story: An $\mathsf{EnrichedCat}^{\uparrow}$ packet is generated from a CategoryUp object boundary, a MonoidalCatUp tensor boundary, hom-object rows, identity rows, composition rows, componentwise hsame transport, Pkg provenance, and finite Cont readbacks. The classifier reads enriched structure only by projecting those finite BHist coordinates and transporting them componentwise; associativity and two-step composition are routed through the monoidal boundary already consumed by the packet. The ledger story is enrichment by visible dependency rows: hom objects, tensor composition, identity, reassociation, provenance, and readback remain attached to the same carrier, so no host-level enriched-category equality or off-packet coherence datum is introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EnrichedCat}^{\uparrow}}$, seed)

Scope closed: This chapter binds the EnrichedCat BHist packet definition and the hom-object, tensor-composition, source, classifier-transport, ledger-exactness, monoidal-boundary, consumer-readback, public-source, and two-step-reassociation theorems for $\mathsf{EnrichedCat}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EnrichedCat}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim host-level enriched-category equality, coherence data outside the consumed monoidal boundary, or an implicit enriched carrier beyond displayed rows.

Upgrade path: To reach obligationClosure, the existing checked theorem surfaces must be summarized into a formal closure target that records the full packet-level obligation assembly.

Chapter 183

A Concrete Naming Certificate for $\mathsf{Topos}^{\uparrow}$

The topos interface $\mathsf{Topos}^{\uparrow}$ packages a category with finite limits, exponentials, and a subobject classifier. It depends on $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ (chapter 61) and $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ (chapter 151), and is the substrate for sheaf-of-sets cohomology and internal logic.

183.1 Carrier definition

Definition 183.1 (Topos finite carrier)**.** *A* $\mathsf{Topos}^{\uparrow}$ *carrier is a finite* BHist *packet displaying a* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row, a* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *row, finite-limit rows, exponential rows, a subobject-classifier row, componentwise* hsame *comparisons,* Cont *rows connecting those structures to the public readback, and* Pkg *provenance for the category and sheaf sources. The carrier exports no ambient universe of sets, no host internal language, and no unlogged categorical object.*

Theorem 183.2 (Topos carrier obligation)**.** *Every accepted* $\mathsf{Topos}^{\uparrow}$ *packet is generated by the finite* BHist *carrier of Definition 183.1: its* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *row,* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *row, finite-limit rows, exponential rows, subobject-classifier row,* hsame *comparisons,* Cont *ledger, and* Pkg *provenance are all displayed. These are the only rows exported to downstream categorical or internal-logic consumers.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposFiniteCarrier obligation surface`

Proof. Project the carrier. It records the category and sheaf sources as explicit certificate rows and attaches the finite-limit, exponential, and subobject-classifier data to those sources.

The accepted classifier is componentwise over the displayed hsame rows. The continuation ledger connects each structural row to the public packet readback, while the package rows record source provenance.

Since the BHist packet has no ambient set universe or host internal-language coordinate, the carrier exports exactly the displayed finite rows. □

Theorem 183.3 (Topos finite-limit exponential scope)**.** *The finite-limit and exponential data of a* $\mathsf{Topos}^{\uparrow}$ *packet are read only from the displayed* BHist *rows, the* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *dependency row, the* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *dependency row, componentwise* hsame *transport, the* Cont *ledger, and* Pkg *provenance. The packet does not import an ambient complete category, external cartesian-closed structure, or host set-theoretic semantics.*

Proof. Read a finite-limit or exponential coordinate from the carrier. Its source is the displayed category row, with sheaf provenance available only through the named dependency certificate.

Transport is componentwise. The hsame rows move finite-limit, exponential, continuation, and provenance coordinates together without changing the packet shape.

Thus the finite-limit and exponential scope is exactly the displayed BEDC surface, not an ambient categorical universe. □

Lean4 ✓ `BEDC.Derived.ToposUp.ToposFiniteCarrier finite limit exponential scope`

183.2 The certificate

Theorem 183.4 (Topos subobject-classifier ledger exactness)**.** *Every subobject-classifier readback in an accepted* $\mathsf{Topos}^{\uparrow}$ *packet is determined by the displayed subobject-classifier row, finite-limit rows, exponential rows,* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *provenance rows, componentwise* hsame *transport, the* Cont *ledger, and* Pkg *data. It exposes no host truth-value object or unlogged internal predicate.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposSubobjectClassifierLedger exactness`

Proof. Project the subobject-classifier coordinate. It is stored as a finite row in the same packet as the finite-limit and exponential coordinates.

The dependency route is visible: the category and sheaf certificates provide provenance, while the continuation ledger carries the classifier row to the public readback. Componentwise hsame transport preserves this route.

No row in the carrier names a host truth-value object or external predicate semantics, so the subobject-classifier readback is exactly the displayed ledger entry. □

Theorem 183.5 (Topos subobject-classifier pullback boundary)**.** *Any pullback read of a displayed subobject-classifier row in a* $\mathsf{Topos}^{\uparrow}$ *packet factors only through the* $\mathsf{Cat}^{\uparrow}$ *site row, the* $\mathsf{Sheaf}^{\uparrow}$ *cover/gluing row, finite-limit rows, exponential rows, the displayed subobject-classifier row, componentwise* hsame*, finite* Cont*, and* Pkg *provenance. It exposes no host truth values, ambient set semantics, or unlogged predicate extensions.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposSubobjectClassifierLedger pullback boundary`
↪ variant `BEDC.Derived.ToposUp.ToposSubobjectClassifierLedger pullback boundary certificate`
↪ variant `BEDC.Derived.ToposUp.ToposFiniteCarrier subobject classifier pullback boundary`
↪ variant `BEDC.Derived.ToposUp.ToposSubobjectClassifier pullback boundary`

Proof. Start with Definition 183.3. It confines the structural part of any pullback request to the displayed category and sheaf source rows, finite-limit rows, exponential rows, componentwise transport, continuation, and package provenance.

Combine this with Definition 183.4. The classifier row itself is a displayed finite ledger entry in the same packet and carries no host truth-value object or external predicate semantics.

Repack the pullback request as one of the finite-limit rows already present in the carrier, with the displayed subobject-classifier row as its classified input. The repacking uses only the visible hsame, Cont, and Pkg routes, so no ambient set semantics or unlogged predicate extension is introduced. □

Theorem 183.6 (Topos site-sheaf classifier obligation)**.** *Any subobject-classifier readback in a* $\mathsf{Topos}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Cat}^{\uparrow}$ *site row, the* $\mathsf{Sheaf}^{\uparrow}$ *cover/gluing row, finite-limit and exponential rows, componentwise* hsame*, finite* Cont*, and* Pkg *provenance. It exports no host truth-value object or ambient set semantics.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposFiniteCarrier site sheaf classifier obligation`

Proof. Project the carrier. The packet stores the subobject-classifier row together with the displayed category and sheaf source rows, so every classifier readback starts from those finite BHist coordinates.

Use Definition 183.3 for the structural categorical rows. The finite-limit and exponential data are already routed through the $\mathsf{Cat}^{\uparrow}$ and $\mathsf{Sheaf}^{\uparrow}$ dependencies, componentwise hsame, finite Cont, and Pkg provenance.

Use Definition 183.4 for the predicate-like classifier row. Repacking these dependency routes gives a finite BEDC obligation surface, and no remaining coordinate can supply a host truth-value object or ambient set semantics. □

Theorem 183.7 (Topos certificate boundary)**.** *Every downstream internal-logic consumer of* $\mathsf{Topos}^{\uparrow}$ *reads only the displayed finite-limit, exponential, subobject-classifier,* $\mathsf{NameCert}_{\mathsf{Cat}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and provenance rows of Definition 183.1. The certificate boundary excludes host set semantics, unlogged predicates, and any internal-language object not displayed in the packet.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposFiniteCarrier certificate boundary`
↪ variant `BEDC.Derived.ToposUp.ToposFiniteCarrier classified boundary transport`

Proof. Start from an internal-logic readback. The carrier obligation theorem confines it to the finite packet rows.

Use the finite-limit exponential theorem for the structural categorical coordinates and the subobject-classifier theorem for predicate-like readback. These account for the category and sheaf dependency rows, the continuation ledger, and the provenance data.

With no remaining coordinate for host semantics or an external internal-language object, the consumer can read only the displayed certificate rows. □

Theorem 183.8 (Topos NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Topos}^{\uparrow}}$ *fields determined by Definition 183.1 are exactly the displayed* $\mathsf{Cat}^{\uparrow}$ *and* $\mathsf{Sheaf}^{\uparrow}$ *source rows, finite-limit rows, exponential rows, subobject-classifier row, componentwise* hsame *transport, finite* Cont *ledger,* Pkg *provenance, and the consumer boundary of Definition 183.7. The obligation surface excludes host set semantics, an ambient universe, unlogged predicates, and any external internal language.*

Lean4 ✓ `BEDC.Derived.ToposUp.ToposFiniteCarrier namecert obligation surface`

Proof. Project the carrier obligation. It supplies the displayed $\mathsf{Cat}^{\uparrow}$ and $\mathsf{Sheaf}^{\uparrow}$ source rows, the finite-limit and exponential rows, the subobject-classifier row, componentwise hsame transport, finite Cont, and Pkg provenance.

Apply Definition 183.3. The structural reads remain inside the finite carrier rows and cannot import an ambient complete category, an external cartesian-closed structure, or host set-theoretic semantics.

Apply Definition 183.4 and Definition 183.5. The classifier readback is the displayed finite ledger entry, and pullback access to it factors only through the same $\mathsf{Cat}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, finite-limit, exponential, hsame, Cont, and Pkg routes.

Finally apply Definition 183.7. It exhausts downstream internal-logic reads by the listed certificate rows, leaving no coordinate for host set semantics, an ambient universe, unlogged predicates, or an external internal language. □

Theorem 183.9 (Topos public certificate export)**.** *Downstream internal-logic and sheaf consumers of* $\mathsf{Topos}^{\uparrow}$ *may read exactly the finite* BHist *packet rows listed in Definition 183.8: the* $\mathsf{Cat}^{\uparrow}$ *row, the* $\mathsf{Sheaf}^{\uparrow}$ *row, finite-limit rows, exponential rows, subobject-classifier row, componentwise* hsame*, finite* Cont*,* Pkg *provenance, and the displayed provenance rows.*

The public certificate exports no host set semantics, ambient universe, unlogged predicate, or external internal-language object.

Proof. Start with Definition 183.2. It places every consumer read inside the displayed finite BHist packet with the category, sheaf, finite-limit, exponential, subobject-classifier, classifier, continuation, and package rows.

Apply Definition 183.3 for the structural rows, and apply Definition 183.4 for the predicate-like row. The pullback boundary of Definition 183.5 keeps classifier access on the same finite ledger.

Finally use Definition 183.7 and Definition 183.8. Together they repack the obligation surface as the public certificate export and leave no coordinate for host set semantics, an ambient universe, an unlogged predicate, or an external internal-language object. □

Closure status for $\mathsf{NameCert}_{\mathsf{Topos}^{\uparrow}}$**.**

Constructive story: The topos packet is a finite BHist carrier over displayed $\mathsf{Cat}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, finite-limit, exponential, subobject-classifier, classifier, continuation, and package rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Topos}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface closes the displayed $\mathsf{Cat}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, finite-limit, exponential, subobject-classifier, BHist, hsame, Cont, Pkg, provenance, and consumer-boundary rows of the topos packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Topos}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ToposUp.ToposFiniteCarrier\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host set semantics, full internal-language semantics, ambient universe assumptions, or a standard bridge.

Upgrade path: The paper-side mature treatment is closed by the displayed carrier, classifier, public export, and consumer-boundary rows; the remaining work is bridge checking.

Chapter 184

A Concrete Naming Certificate for $\mathsf{ProbSpace}^{\uparrow}$

The probability-space interface $\mathsf{ProbSpace}^{\uparrow}$ packages a measure space whose total measure is the unit. It depends on $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92) and $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38).

184.1 Carrier definition

184.2 The certificate

Theorem 184.1 (ProbSpace complement mass additivity)**.** *For a* $\mathsf{ProbSpace}^{\uparrow}$ *carrier as in section 184.1, write* Ω *for its total event,* $\mathcal{A}$ *for the designated* σ*-algebra inherited from the underlying* $\mathsf{Measure}^{\uparrow}$ *carrier,* μ *for its* $\mathsf{Real}^{\uparrow}$*-valued measure map,* 1_R *for the probability unit selected by the* $\mathsf{ProbSpace}^{\uparrow}$ *total-mass row, and* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, A^{\complement} \in \mathcal{A}$ *be disjoint measurable events whose measurable binary union is the total event* Ω*. Suppose* $\varnothing \in \mathcal{A}$*, the underlying* $\mathsf{Measure}^{\uparrow}$ *certificate supplies* σ*-additivity for every pairwise-disjoint measurable sequence, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^{\uparrow}$ *sums, and the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row with zero* 0_R*. If the* $\mathsf{ProbSpace}^{\uparrow}$ *total-mass row gives*

$$\mu(\Omega) \sim_R 1_R,$$

then

$$\mu(A) + \mu(A^{\complement}) \sim_R 1_R.$$

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceComplementMass additive readback`

Proof. Let $U := A \cup A^{\complement}$ be the measurable binary union supplied by the σ-algebra row. By hypothesis, $U = \Omega$. Applying Definition 92.26 to the disjoint pair $A, A^{\complement}$ gives

$$\mu(U) \sim_R \mu(A) + \mu(A^{\complement}).$$

Reading U as Ω gives

$$\mu(\Omega) \sim_R \mu(A) + \mu(A^{\complement}).$$

The probability-space total-mass row gives

$$\mu(\Omega) \sim_R 1_R.$$

The classifier $\sim_R$ is the $\mathsf{Real}^\uparrow$ classifier, so its symmetry and transitivity fields compose these two comparisons. Symmetry of the finite-additivity comparison gives

$$\mu(A) + \mu(A^{\complement}) \sim_R \mu(\Omega),$$

and transitivity with the total-mass comparison yields

$$\mu(A) + \mu(A^{\complement}) \sim_R 1_R.$$

□

Theorem 184.2 (ProbSpace complement mass as one minus event mass)**.** *For a* $\mathsf{ProbSpace}^\uparrow$ *carrier as in section 184.1, write* Ω *for its total event,* $\mathcal{A}$ *for the designated* σ*-algebra inherited from the underlying* $\mathsf{Measure}^\uparrow$ *carrier,* μ *for its* $\mathsf{Real}^\uparrow$*-valued measure map,* 1_R *for the probability unit selected by the* $\mathsf{ProbSpace}^\uparrow$ *total-mass row, and* $x \sim_R y$ *for the* $\mathsf{Real}^\uparrow$ *classifier of Definition 38.9. Let* $A, A^{\complement} \in \mathcal{A}$ *be disjoint measurable events whose measurable binary union is the total event* Ω*. Suppose* $\varnothing \in \mathcal{A}$*, the underlying* $\mathsf{Measure}^\uparrow$ *certificate supplies* σ*-additivity for every pairwise-disjoint measurable sequence, classifier-stable head-tail readback and pointwise classifier stability for the resulting countable* $\mathsf{Real}^\uparrow$ *sums, and the target* $\mathsf{Real}^\uparrow$ *addition is carried by its additive* $\mathsf{AbGroup}^\uparrow$ *row with zero* 0_R *and inverse operation* $-$*. If the* $\mathsf{ProbSpace}^\uparrow$ *total-mass row gives*

$$\mu(\Omega) \sim_R 1_R,$$

then

$$\mu(A^{\complement}) \sim_R 1_R + (-\mu(A)).$$

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceComplementMass right solution`
↪ variant `BEDC.Derived.ProbSpaceUp.ProbSpaceComplementMass one minus singleton`

Proof. By Definition 184.1 applied to the displayed pair and these hypotheses,

$$\mu(A) + \mu(A^{\complement}) \sim_R 1_R.$$

The additive $\mathsf{AbGroup}^\uparrow$ row of Definition 42.1 contains the same carrier classifier $\sim_R$, the same addition $+$, the zero 0_R, the inverse operation $-$, addition congruence, and commutativity. By Definition 42.33, omitting the commutativity field leaves a $\mathsf{Group}^\uparrow$ row for the same data.

Use the commutativity field to exchange the two summands:

$$\mu(A^{\complement}) + \mu(A) \sim_R \mu(A) + \mu(A^{\complement}).$$

Transitivity with the complement-additivity comparison gives the right-multiplication equation

$$\mu(A^{\complement}) + \mu(A) \sim_R 1_R.$$

Now read Definition 41.68 in the retained additive group, with multiplication written as $+$, inverse written as $-$, $x = \mu(A^{\complement})$, $a = \mu(A)$, and $b = 1_R$. It solves the displayed equation as

$$\mu(A^{\complement}) \sim_R 1_R + (-\mu(A)).$$

□

Theorem 184.3 (ProbSpace monotone event bounds)**.** *For a* $\mathsf{ProbSpace}^{\uparrow}$ *carrier obtained from the probability substrate of Definition 92.71, write* Ω *for its total event,* $\mathcal{A}$ *for the displayed measurable-event family inherited from the underlying* $\mathsf{Measure}^{\uparrow}$ *carrier,* μ *for its* $\mathsf{Real}^{\uparrow}$*-valued measure map,* 1_R *for the endpoint selected by the total-mass row, and* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* $A, B \in \mathcal{A}$ *be displayed events in the same carried fiber, and suppose the public event readback displays*

$$A \subseteq B \subseteq \Omega.$$

Assume the inherited complement/relative-difference rows give the two measurable decompositions required by Definition 92.99, and that the target RealAlgOrder order row is stable under $\sim_R$ *and transitive. If the total-mass row gives*

$$\mu(\Omega) \sim_R 1_R,$$

then

$$\mu(A) \leq_R \mu(B), \qquad \mu(B) \leq_R 1_R, \qquad \mu(A) \leq_R 1_R.$$

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceMonotoneEvent bounds`

Proof. By Definition 92.71, the $\mathsf{ProbSpace}^{\uparrow}$ consumer reads the same displayed measurable events, relative-difference rows, finite-additivity rows, nonnegative value rows, and RealAlgOrder additive order rows as the underlying $\mathsf{Measure}^{\uparrow}$ package, with the total-event row and unit endpoint added by the probability surface. The total-mass comparison is the normalization row isolated in Definition 92.59.

Apply the inherited complement/relative-difference row of Definition 92.16 to the displayed inclusion $A \subseteq B$. It supplies a measurable event $B \setminus A$, disjointness from A, and a binary-union readback classifying $A \cup (B \setminus A)$ as B. These are exactly the decomposition inputs for Definition 92.99, so the inherited MeasureUp monotonicity row gives

$$\mu(A) \leq_R \mu(B).$$

Apply the same row to the displayed inclusion $B \subseteq \Omega$. It supplies $\Omega \setminus B$ and a disjoint decomposition of Ω with first summand B. A second application of Definition 92.99 gives

$$\mu(B) \leq_R \mu(\Omega).$$

The total-mass row gives $\mu(\Omega) \sim_R 1_R$. Stability of the RealAlgOrder order row under the $\mathsf{Real}^{\uparrow}$ classifier transports the right endpoint of the last comparison across this classifier equality, yielding

$$\mu(B) \leq_R 1_R.$$

Finally, transitivity of the same order row composes $\mu(A) \leq_R \mu(B)$ with $\mu(B) \leq_R 1_R$, and gives

$$\mu(A) \leq_R 1_R.$$

□

Definition 184.4 (ProbSpace public event packet)**.** *For a* $\mathsf{ProbSpace}^{\uparrow}$ *carrier obtained from the probability substrate of Definition 92.71, the public event packet is the tuple whose carrier entries are the* $\mathsf{Measure}^{\uparrow}$ *measurable-event family* $\mathcal{A}$*, the total event* Ω*, the* $\mathsf{Real}^{\uparrow}$*-valued measure map* μ*, and the selected endpoint* 1_R*. Its classifier entries are the displayed measurable-event readback classifier inherited from* $\mathsf{Measure}^{\uparrow}$*, the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$*, the RealAlgOrder comparison row* $\leq_R$*, and the classifier-stability fields for addition, order, complement, and relative difference. Its ledger*

entries are the underlying BHist *histories, the carried package row* Pkg*, the* $\mathsf{NameCert}_{\mathsf{Measure}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Real}^\uparrow}$ *rows read by the probability substrate, and the* $\mathsf{NameCert}_{\mathsf{ProbSpace}^\uparrow}$ *row that records the total-mass comparison*

$$\mu(\Omega) \sim_R 1_R.$$

The packet is public when these entries are exposed as named BEDC rows, with no appeal to a host probability-space axiom outside the displayed BHist*,* Pkg*, and* NameCert *ledger.*

Lean4 (definition) `BEDC.Derived.ProbSpaceUp.ProbSpacePublicEventPacket`

Theorem 184.5 (ProbSpace total event normalization row)**.** *For the public event packet of Definition 184.4, the displayed total-mass comparison*

$$\mu(\Omega) \sim_R 1_R$$

is exported as a $\mathsf{ProbSpace}^\uparrow$ *BEDC row. It is a row of the carried certificate surface, not an assertion that the ambient host theory supplies probability normalization for arbitrary outside measures.*

Proof. Project the total-mass entry from the $\mathsf{NameCert}_{\mathsf{ProbSpace}^\uparrow}$ ledger in Definition 184.4. The projected row has left endpoint $\mu(\Omega)$, right endpoint 1_R, and classifier $\sim_R$.

The classifier is the same $\mathsf{Real}^\uparrow$ classifier used in Definition 38.9. Hence the row is read as the BEDC comparison $\mu(\Omega) \sim_R 1_R$ inside the carried package, and no host normalization principle is introduced. □

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpacePublicEventPacket normalization bounds`

Theorem 184.6 (ProbSpace event classifier transport row)**.** *For the public event packet of Definition 184.4, measurable-event readback transport together with* $\mathsf{Real}^\uparrow$ *classifier transport preserves the measure endpoint, the complement packet, and the relative-difference rows used by Definition 184.3. In particular, if displayed events A and B are transported along the inherited event readback classifier and their measured endpoints are transported along $\sim_R$, then the rows presenting $A^\complement$, $B \setminus A$, $\mu(A)$, $\mu(B)$, $\mu(A^\complement)$, and $\mu(B \setminus A)$ remain rows of the same public packet.*

Proof. Start with the MeasureUp complement and relative-difference row of Definition 92.16. It is one of the inherited entries of the public event packet, so event readback transport keeps the displayed complement and relative-difference events in the same measurable-event family $\mathcal{A}$.

The measured endpoints are carried by the $\mathsf{Real}^\uparrow$-valued measure map μ. Apply the same $\mathsf{Real}^\uparrow$ classifier transport used in Definition 184.3; it transports endpoint comparisons without changing the named packet entries.

Combining these two transports preserves the complement packet, the relative-difference packet, and the RealAlgOrder rows used for monotonicity. Thus the transported rows remain available on the public $\mathsf{ProbSpace}^\uparrow$ surface. □

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpacePublicEventPacket transport rows`

Theorem 184.7 (ProbSpace public certificate export)**.** *The public event packet of Definition 184.4 exports the probability-space certificate surface* $\mathsf{NameCert}_{\mathsf{ProbSpace}^\uparrow}$*. The exported surface contains the complement-mass additivity row of Definition 184.1, the one-minus complement row of Definition 184.2, the monotone event-bounds row of Definition 184.3, the total event normalization row of Definition 184.5, and the event classifier transport row of Definition 184.6. Consequently, downstream* $\mathsf{RandomVar}^\uparrow$*,* $\mathsf{Distribution}^\uparrow$*,* $\mathsf{Independence}^\uparrow$*, and* $\mathsf{CondExp}^\uparrow$ *objects may cite this single* $\mathsf{NameCert}_{\mathsf{ProbSpace}^\uparrow}$ *surface for probability-space rows rather than reconstructing those rows from separate consequences.*

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpacePublicEventPacket public certificate export`

Proof. The packet already carries the $\mathsf{Measure}^{\uparrow}$ measurable-event family, complement and relative-difference rows, the $\mathsf{Real}^{\uparrow}$ classifier and order rows, the total event, the measure map, and the unit endpoint. By Definition 184.5, its total-mass comparison is a named certificate row.

Add the three scoped consequences already proved in this section. The complement-mass additivity row comes from Definition 184.1; the one-minus row comes from Definition 184.2; and the monotone event-bounds row comes from Definition 184.3.

It remains to make these rows stable under the event and endpoint classifiers read by downstream objects. This is exactly Definition 184.6, so the rows assemble into one public $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ surface. □

Theorem 184.8 (ProbSpace MeasureUp dependency route)**.** *Every public* $\mathsf{ProbSpace}^{\uparrow}$ *read that consumes a probability-specific event bound first exposes the underlying* $\mathsf{Measure}^{\uparrow}$ *surface of chapter 92. Concretely, the consumer must read the measurable-event family* $\mathcal{A}$*, the total event* Ω*, the* $\mathsf{Real}^{\uparrow}$*-valued measure row* μ*, the complement and relative-difference rows used by Definition 184.1 and Definition 184.3, and the total-mass normalization row of Definition 184.5 before it may consume the public certificate export of Definition 184.7.*

Proof. Start from the public event packet of Definition 184.4. Its first carrier entries are the $\mathsf{Measure}^{\uparrow}$ measurable-event family, total event, measure row, complement and relative-difference rows, event classifier transport, and $\mathsf{Real}^{\uparrow}$ endpoint classifier. These are inherited rows, not probability-specific consequences.

The probability surface adds the unit endpoint and the total-mass normalization row. By Definition 184.5, that row is the displayed comparison $\mu(\Omega) \sim_R 1_R$ inside the carried packet.

Now read any event bound through Definition 184.7. The exported complement, one-minus, and monotone-bound rows cite the inherited complement and relative-difference rows together with the normalization row. Therefore a downstream $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, or $\mathsf{CondExp}^{\uparrow}$ consumer reaches probability-specific bounds only after the $\mathsf{Measure}^{\uparrow}$ event surface and total-mass row have already been exposed. □

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceMeasureDependencyRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$**.**

Constructive story: A probability space is obtained by taking a carried $\mathsf{Measure}^{\uparrow}$ packet, keeping its measurable-event family $\mathcal{A}$, total event Ω, and $\mathsf{Real}^{\uparrow}$-valued measure row μ, and then adding the displayed normalization row that classifies $\mu(\Omega)$ with the selected unit endpoint 1_R. The carrier admits only those event packets whose measure, complement, relative-difference, order, and additive rows are already visible through the $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{AbGroup}^{\uparrow}$ surfaces; two packets are the same when the inherited event readback classifier and the $\mathsf{Real}^{\uparrow}$ classifier transport all measured endpoints, order comparisons, complements, and differences to matching rows. The source therefore covers the public event packet, the pattern records total-mass normalization, complement additivity, one-minus readback, and monotone bounds, and the ledger keeps the raw BHist histories, Pkg provenance, and consumed NameCert rows aligned with the visible probability rows. Since the underlying measure and real classifiers are equivalence relations and the transported event packet keeps the same carrier entries, an admitted normalized packet supplies the nonempty witness, reflexive, symmetric, and transitive classifier comparisons, and carrier closure under classifier replacement. A history packet h lies in the signature gap of a $\mathsf{ProbSpace}^{\uparrow}$ token p when the source-domain accepts the displayed event-and-measure packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; unfolded, this says that h is exactly the normalized MeasureUp packet whose total event has mass 1_R, with the same complement, difference, and order rows that p exposes. The paper bridge is conservative: it reads the ordinary probability-space surface only through the displayed event classifier and $\mathsf{Real}^{\uparrow}$ endpoint comparisons, never through a host probability-space axiom. The chapter registers this normalized-measure construction under the name $\mathsf{ProbSpace}^{\uparrow}$; probability is the MeasureUp ledger plus one total-mass row, with no independent random-variable, distribution, conditional-expectation, or standard-space primitive introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public packet of Definition 184.7 exports the MeasureUp event rows, RealUp classifier and order transport, AbGroupUp additive inverse rows, BHist histories, hsame transport, Pkg provenance, and NameCert ledger rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProbSpace}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.ProbSpaceUp.ProbSpaceMonotoneEvent\_bounds`

Bridge status: paperBridge

Not claimed: This obligation packet does not close random variables, distributions, independence, conditional expectation, martingales, standard probability spaces, or any standard bridge.

Upgrade path: No further theory-axis upgrade remains; standard bridge checking remains a separate formal-axis matter.

Obligation binders invoked by the following definition.

1. **closure: RESOLVED.** The displayed binary-union, intersection, and relative-difference event rows are the finite-event closure rows of the Measure$^\uparrow$ surface, specifically the finite disjoint event envelope of Chapter 94.15 and the event-value closure surface of Chapter 94.16.

2. **coverage: RESOLVED.** The outside-piece decomposition is the public relative-difference coverage row of Theorem 94.100, read through the same event classifier rather than through host event equality.

3. **bridge: RESOLVED.** The ProbSpace$^\uparrow$ consumer inherits the finite Measure$^\uparrow$ event surface by the probability-substrate row of Theorem 94.70.

Definition 184.9 (Binary cover-difference ledger)**.** *Let P be a* ProbSpace$^\uparrow$ *carrier obtained from the probability substrate of a* Measure$^\uparrow$ *carrier. Write $\mathcal{A}$ for the displayed measurable-event family inherited from the underlying* Measure$^\uparrow$ *carrier, write μ for the* $\mathsf{Real}^\uparrow$*-valued measure map, and write $E \sim_{\mathcal{A}} F$ for the public measurable-event classifier. For displayed events $A, B \in \mathcal{A}$, a binary cover-difference ledger for (A, B) consists of displayed events*

$$U, I, D \in \mathcal{A}$$

and public rows saying that U is the displayed binary union of A and B, I is the displayed binary intersection of A and B, and D is the displayed outside-A part of B, together with the two disjoint cover readbacks

$$A \perp D, \qquad A \cup D \sim_{\mathcal{A}} U,$$

and

$$I \perp D, \qquad I \cup D \sim_{\mathcal{A}} B.$$

The ledger is a BEDC *event-surface object: it records measurable closure, disjointness, and classifier readback rows, and it does not add host set subtraction or quotient equality of events.*

Obligation binders invoked by the following theorem.

1. **closure: RESOLVED.** The measurable union and value-transport steps are supplied by the Measure$^\uparrow$ finite disjoint event envelope of Chapter 94.15, the event-value closure surface of Chapter 94.16, and the finite disjoint-union additivity row of Theorem 94.26.

2. **coverage: RESOLVED.** The two cover decompositions consumed in the proof are exactly the two cover rows packaged in Definition 184.9, whose outside-piece source is the relative-difference disjoint decomposition of Theorem 94.100.

3. **bridge: RESOLVED.** The use of these Measure$^\uparrow$ rows inside a ProbSpace$^\uparrow$ carrier is discharged by Theorem 94.70, which states that the probability consumer adds only the total-event and unit-mass rows on top of the inherited measure substrate.

Theorem 184.10 (ProbSpace binary inclusion-exclusion identity)**.** *Let P be a* ProbSpace$^{\uparrow}$ *carrier obtained from the probability substrate of a* Measure$^{\uparrow}$ *carrier. Write $\mathcal{A}$ for its displayed measurable-event family, write μ for its* $\mathsf{Real}^{\uparrow}$*-valued measure map, and write $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier. Let $A, B \in \mathcal{A}$ be displayed events in one carried fiber. Suppose that (A, B) has a binary cover-difference ledger with displayed events U, I, D in the sense of Definition 184.9. Suppose also that the finite disjoint-union additivity row applies to the disjoint pairs (A, D) and (I, D), and that the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then*

$$\mu(U) + \mu(I) \sim_R \mu(A) + \mu(B).$$

Consequently, when the public notation $A \cup B$ names the displayed event U and the public notation $A \cap B$ names the displayed event I, one obtains

$$\mu(A \cup B) + \mu(A \cap B) \sim_R \mu(A) + \mu(B).$$

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceBinaryCoverDifferenceLedger inclusion exclusion identity`

Proof. The first disjoint cover row in the ledger gives $A \perp D$ and

$$A \cup D \sim_{\mathcal{A}} U.$$

Apply the finite disjoint-union additivity row to the displayed disjoint pair (A, D). This gives

$$\mu(A \cup D) \sim_R \mu(A) + \mu(D).$$

The event-value transport row carries the event classifier comparison $A \cup D \sim_{\mathcal{A}} U$ to the $\mathsf{Real}^{\uparrow}$ endpoint classifier. Composing that transport with the displayed finite-additivity comparison gives

$$\mu(U) \sim_R \mu(A) + \mu(D).$$

The second disjoint cover row in the ledger gives $I \perp D$ and

$$I \cup D \sim_{\mathcal{A}} B.$$

Apply the same finite disjoint-union additivity row to the displayed disjoint pair (I, D). This gives

$$\mu(I \cup D) \sim_R \mu(I) + \mu(D).$$

Transporting the left endpoint through $I \cup D \sim_{\mathcal{A}} B$ and composing in the $\mathsf{Real}^{\uparrow}$ classifier gives

$$\mu(B) \sim_R \mu(I) + \mu(D).$$

Now compute in the additive $\mathsf{AbGroup}^{\uparrow}$ row carried by the target $\mathsf{Real}^{\uparrow}$ endpoint surface. Addition congruence applied to the comparison

$$\mu(U) \sim_R \mu(A) + \mu(D)$$

gives

$$\mu(U) + \mu(I) \sim_R (\mu(A) + \mu(D)) + \mu(I).$$

Associativity gives

$$(\mu(A) + \mu(D)) + \mu(I) \sim_R \mu(A) + (\mu(D) + \mu(I)).$$

Commutativity gives

$$\mu(D) + \mu(I) \sim_R \mu(I) + \mu(D).$$

The earlier comparison

$$\mu(B) \sim_R \mu(I) + \mu(D)$$

therefore gives, by symmetry, transitivity, and addition congruence with the fixed left summand $\mu(A)$,

$$\mu(A) + (\mu(D) + \mu(I)) \sim_R \mu(A) + \mu(B).$$

Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier composes the displayed chain:

$$\mu(U) + \mu(I) \sim_R \mu(A) + \mu(B).$$

Substituting the public displayed representatives $U = A \cup B$ and $I = A \cap B$ gives

$$\mu(A \cup B) + \mu(A \cap B) \sim_R \mu(A) + \mu(B).$$

□

Definition 184.11 (Ternary inclusion-exclusion ledger)**.** *Let P be a $\mathsf{ProbSpace}^{\uparrow}$ carrier obtained from the probability substrate of a $\mathsf{Measure}^{\uparrow}$ carrier. Write $\mathcal{A}$ for the displayed measurable-event family inherited from the underlying $\mathsf{Measure}^{\uparrow}$ carrier, write μ for the $\mathsf{Real}^{\uparrow}$-valued measure map, and write $E \sim_{\mathcal{A}} F$ for the public measurable-event classifier. For displayed events $A, B, C \in \mathcal{A}$, a ternary inclusion-exclusion ledger for (A, B, C) consists of displayed events*

$$U, V, I_{AB}, I_{AC}, I_{BC}, K, T \in \mathcal{A}$$

and the following public rows:

1. *binary cover-difference ledgers for the original pairs (A, B), (A, C), and (B, C) in the sense of Definition 184.9, whose displayed intersections are respectively I_{AB}, I_{AC}, and I_{BC}, and whose displayed union for (B, C) is V;*
2. *a binary cover-difference ledger for (A, V) whose displayed union is U and whose displayed intersection is K;*
3. *a binary cover-difference ledger for (I_{AB}, I_{AC}) whose displayed union is K and whose displayed intersection is T.*

Thus U is the displayed ternary union of A, B, C, K is the displayed event $A \cap (B \cup C)$, and T is the displayed triple intersection read as $(A \cap B) \cap (A \cap C)$ on the carried event surface. The ledger records these as measurable-event rows and classifier readbacks, not as host set-theoretic equalities outside the BEDC *event surface.*

Lean4 (definition) `BEDC.Derived.ProbSpaceUp.ProbSpaceTernaryInclusionExclusionLedger`

Theorem 184.12 (ProbSpace ternary inclusion-exclusion identity)**.** *Let P be a $\mathsf{ProbSpace}^{\uparrow}$ carrier obtained from the probability substrate of a $\mathsf{Measure}^{\uparrow}$ carrier. Write $\mathcal{A}$ for its displayed measurable-event family, write μ for its $\mathsf{Real}^{\uparrow}$-valued measure map, and write $x \sim_R y$ for the $\mathsf{Real}^{\uparrow}$ classifier. Let $A, B, C \in \mathcal{A}$ be displayed events in one carried fiber, and suppose that (A, B, C) has a ternary inclusion-exclusion ledger with displayed events*

$$U, V, I_{AB}, I_{AC}, I_{BC}, K, T$$

in the sense of Definition 184.11. Suppose also that the finite disjoint-union additivity row applies to the disjoint pairs consumed by the three binary cover-difference ledgers (B, C), (A, V), *and* (I_{AB}, I_{AC}), *and that the target* $\mathsf{Real}^{\uparrow}$ *addition is carried by its additive* $\mathsf{AbGroup}^{\uparrow}$ *row. Then, with four-term sums read in the displayed left-associated bracketing,*

$$((\mu(U) + \mu(I_{AB})) + \mu(I_{AC})) + \mu(I_{BC}) \sim_R ((\mu(A) + \mu(B)) + \mu(C)) + \mu(T).$$

Consequently, when the public notation $A \cup B \cup C$ *names* U, $A \cap B$ *names* I_{AB}, $A \cap C$ *names* I_{AC}, $B \cap C$ *names* I_{BC}, *and* $A \cap B \cap C$ *names* T, *one obtains*

$$\mu(A \cup B \cup C) + \mu(A \cap B) + \mu(A \cap C) + \mu(B \cap C) \sim_R \mu(A) + \mu(B) + \mu(C) + \mu(A \cap B \cap C).$$

Lean4 ✓ `BEDC.Derived.ProbSpaceUp.ProbSpaceTernaryInclusionExclusionLedger identity`

Proof. Apply Definition 184.10 to the binary cover-difference ledger for (B, C). Its displayed union is V and its displayed intersection is I_{BC}, so

$$\mu(V) + \mu(I_{BC}) \sim_R \mu(B) + \mu(C).$$

Apply the same theorem to the binary cover-difference ledger for (A, V). Its displayed union is U and its displayed intersection is K, so

$$\mu(U) + \mu(K) \sim_R \mu(A) + \mu(V).$$

Apply the same theorem a third time to the binary cover-difference ledger for (I_{AB}, I_{AC}). Its displayed union is K and its displayed intersection is T, so

$$\mu(K) + \mu(T) \sim_R \mu(I_{AB}) + \mu(I_{AC}).$$

Use addition congruence in the additive $\mathsf{AbGroup}^{\uparrow}$ row to add $\mu(I_{BC})$ to the comparison for (A, V):

$$(\mu(U) + \mu(K)) + \mu(I_{BC}) \sim_R (\mu(A) + \mu(V)) + \mu(I_{BC}).$$

Associativity rewrites the right endpoint as $\mu(A) + (\mu(V) + \mu(I_{BC}))$. Addition congruence with the fixed left summand $\mu(A)$ and the comparison for (B, C) gives

$$\mu(A) + (\mu(V) + \mu(I_{BC})) \sim_R \mu(A) + (\mu(B) + \mu(C)).$$

Associativity rewrites the right endpoint as $(\mu(A) + \mu(B)) + \mu(C)$. By transitivity of $\sim_R$,

$$(\mu(U) + \mu(K)) + \mu(I_{BC}) \sim_R (\mu(A) + \mu(B)) + \mu(C).$$

Add $\mu(T)$ to both sides by addition congruence:

$$((\mu(U) + \mu(K)) + \mu(I_{BC})) + \mu(T) \sim_R ((\mu(A) + \mu(B)) + \mu(C)) + \mu(T).$$

It remains only to replace the middle event K by the two displayed pairwise intersections, with the triple intersection carried to the right endpoint. Associativity and commutativity in the additive $\mathsf{AbGroup}^{\uparrow}$ row give

$$((\mu(U) + \mu(K)) + \mu(I_{BC})) + \mu(T) \sim_R (\mu(U) + (\mu(K) + \mu(T))) + \mu(I_{BC}).$$

Addition congruence with the comparison for (I_{AB}, I_{AC}) gives

$$(\mu(U) + (\mu(K) + \mu(T))) + \mu(I_{BC}) \sim_R (\mu(U) + (\mu(I_{AB}) + \mu(I_{AC}))) + \mu(I_{BC}).$$

A final associativity rewrite gives

$$(\mu(U) + (\mu(I_{AB}) + \mu(I_{AC}))) + \mu(I_{BC}) \sim_R ((\mu(U) + \mu(I_{AB})) + \mu(I_{AC})) + \mu(I_{BC}).$$

Symmetry and transitivity of the $\mathsf{Real}^{\uparrow}$ classifier compose these displayed comparisons with the preceding four-term comparison, yielding

$$((\mu(U) + \mu(I_{AB})) + \mu(I_{AC})) + \mu(I_{BC}) \sim_R ((\mu(A) + \mu(B)) + \mu(C)) + \mu(T).$$

Substituting the public displayed representatives $U = A \cup B \cup C$, $I_{AB} = A \cap B$, $I_{AC} = A \cap C$, $I_{BC} = B \cap C$, and $T = A \cap B \cap C$ gives the stated public form. □

The $\mathsf{RandomVar}^{\uparrow}$ hub routes measurable preimage exactness through chapter 184, chapter 92, and the downstream distribution readback of chapter 186.

Chapter 185

A Concrete Naming Certificate for $\mathsf{RandomVar}^{\uparrow}$

The random-variable interface $\mathsf{RandomVar}^{\uparrow}$ packages a measurable function from a probability space to a target measure space. It depends on $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ (chapter 184) and $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92).

185.1 Carrier definition

185.2 The certificate

Theorem 185.1 (RandomVar preimage measure and probability lattice link)**.** *Every* $\mathsf{RandomVar}^{\uparrow}$ *preimage exactness theorem in this section consumes the event classifier, sigma-algebra, empty-event, union, complement, and relative-difference rows of chapter 92 together with the source probability carrier of chapter 184. The preimage rows transport target* $\mathsf{Measure}^{\uparrow}$ *events back to source* $\mathsf{ProbSpace}^{\uparrow}$ *events; they do not introduce an independent event lattice or a second source sigma-algebra.*

Proof. Inspect the hypotheses of the preimage exactness theorems below. Each theorem starts with a carried $\mathsf{RandomVar}^{\uparrow}$ map from a $\mathsf{ProbSpace}^{\uparrow}$ source to a target $\mathsf{Measure}^{\uparrow}$ carrier, then writes $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families.

The event comparisons use the source classifier of Definition 92.7. The closure rows for binary unions, empty events, complements, and relative differences are cited from Definition 92.4 and the corresponding displayed Measure rows, while the source carrier itself is the probability space of section 184.1.

The $\mathsf{RandomVar}^{\uparrow}$ certificate contributes only the measurable-preimage row and its membership readback. Therefore the preimage exactness surface is a horizontal consumer of Measure and ProbSpace data, not a new event lattice. □

Theorem 185.2 (RandomVar preimage preserves disjoint binary unions)**.** *Let* $X : S \to T$ *be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write* $\mathcal{A}_S$ *and* $\mathcal{A}_T$ *for the source and target measurable-event families, and write* $E \sim_{\mathcal{A}_S} F$ *for the source event classifier of Definition 92.7. Let* B, C, U_T *be target events such that* $B, C, U_T \in \mathcal{A}_T$ *and the displayed target binary-union row reads* U_T *as* $B \cup C$*:*

$$t \in U_T \quad \Longleftrightarrow \quad t \in B \text{ or } t \in C.$$

Assume B and C are disjoint: for every carried target point t, simultaneous witnesses $t \in B$ and $t \in C$ are eliminated. Let

$$A_B := X^{-1}(B), \qquad A_C := X^{-1}(C), \qquad A_U := X^{-1}(U_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to its source measurable preimage, then $A_B, A_C, A_U \in \mathcal{A}_S$. Moreover, if U_S is the source binary union of A_B and A_C supplied by the source* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra row of Definition 92.4, then*

$$U_S \in \mathcal{A}_S, \qquad A_U \sim_{\mathcal{A}_S} U_S,$$

and A_B and A_C are disjoint source events.

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage disjoint binary union exactness`

Proof. The measurable-preimage row of the $\mathsf{RandomVar}^{\uparrow}$ certificate applies to the three target measurable events B, C, U_T. This gives $A_B, A_C, A_U \in \mathcal{A}_S$.

Since the source probability space inherits its measurable family from the underlying $\mathsf{Measure}^{\uparrow}$ carrier, Definition 92.4 supplies the binary-union row for the measurable pair A_B, A_C. Let U_S be that displayed source union. Its row gives $U_S \in \mathcal{A}_S$ and the membership readback

$$s \in U_S \quad \Longleftrightarrow \quad s \in A_B \text{ or } s \in A_C.$$

It remains to compare A_U with U_S through the source event classifier. For a carried source point s, unfold the preimage rows:

$$s \in A_U \quad \Longleftrightarrow \quad X(s) \in U_T.$$

The target binary-union readback gives

$$X(s) \in U_T \quad \Longleftrightarrow \quad X(s) \in B \text{ or } X(s) \in C.$$

Unfolding the two preimage rows again gives

$$X(s) \in B \text{ or } X(s) \in C \quad \Longleftrightarrow \quad s \in A_B \text{ or } s \in A_C.$$

The source binary-union readback identifies the right side with $s \in U_S$. Thus A_U and U_S have the same carried source memberships. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$A_U \sim_{\mathcal{A}_S} U_S.$$

Finally suppose s has witnesses $s \in A_B$ and $s \in A_C$. Unfolding preimage gives witnesses $X(s) \in B$ and $X(s) \in C$. The displayed target disjointness row for B and C eliminates such a pair, so no carried source point can inhabit both A_B and A_C. Hence the preimage pair is disjoint. □

Theorem 185.3 (RandomVar preimage relative-difference exactness)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, and write $E \sim_{\mathcal{A}_S} F$ for the source event classifier of Definition 92.7. Let A and B be target events in one carried target fiber with $A, B \in \mathcal{A}_T$, and suppose the public target readback*

displays $B \subseteq A$. Suppose the target complement/relative-difference row of Definition 92.16 supplies a measurable event

$$D_T := A \setminus B$$

with public membership readback

$$t \in D_T \quad \Longleftrightarrow \quad t \in A \text{ and } t \notin B.$$

Define source preimages

$$A_S := X^{-1}(A), \qquad B_S := X^{-1}(B), \qquad D_X := X^{-1}(D_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to its source measurable preimage, then $A_S, B_S, D_X \in \mathcal{A}_S$, and the public source readback displays $B_S \subseteq A_S$. Let D_S be the source relative-difference event $A_S \setminus B_S$ supplied by the source* $\mathsf{Measure}^{\uparrow}$ *complement/relative-difference row. Then*

$$D_S \in \mathcal{A}_S, \qquad D_X \sim_{\mathcal{A}_S} D_S,$$

and the equality is certified with the source relative-difference ledger: B_S and D_S are disjoint, and the displayed source binary union $B_S \cup D_S$ is classified as A_S.

Proof. The target complement/relative-difference row gives the measurable target event D_T. Applying the measurable-preimage row of the $\mathsf{RandomVar}^{\uparrow}$ certificate to A, B, and D_T gives

$$A_S, B_S, D_X \in \mathcal{A}_S.$$

To read the source inclusion, fix a carried source point s with a witness $s \in B_S$. The preimage readback for B gives $X(s) \in B$. The displayed target inclusion $B \subseteq A$ gives $X(s) \in A$, and the preimage readback for A gives $s \in A_S$. Hence the public source readback displays $B_S \subseteq A_S$.

Apply the source complement/relative-difference row of Definition 92.16, equivalently the row projected in Definition 92.101, to the displayed source inclusion $B_S \subseteq A_S$. It supplies the measurable event $D_S := A_S \setminus B_S$, the disjointness witness for B_S and D_S, and the binary-union readback classifying $B_S \cup D_S$ as A_S. Its relative-difference readback is

$$s \in D_S \quad \Longleftrightarrow \quad s \in A_S \text{ and } s \notin B_S.$$

It remains to compare D_X with D_S. For every carried source point s, the displayed rows give

$$\begin{aligned} s \in D_X &\Longleftrightarrow X(s) \in D_T \\ &\Longleftrightarrow X(s) \in A \text{ and } X(s) \notin B \\ &\Longleftrightarrow s \in A_S \text{ and } s \notin B_S \\ &\Longleftrightarrow s \in D_S. \end{aligned}$$

The first and third equivalences are the preimage readbacks, the second is the target relative-difference readback, and the last is the source relative-difference readback.

Thus D_X and D_S have the same carried source memberships. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$D_X \sim_{\mathcal{A}_S} D_S.$$

The measurable, disjointness, and binary-union statements are the corresponding fields of the same source relative-difference ledger. □

Lean4 ✓ BEDC.Derived.RandomVarUp.RandomVarPreimage relative difference exactness

Theorem 185.4 (RandomVar preimage complement and relative-difference exactness)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write Ω_S and Ω_T for the source and target total events, write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, and write $E \sim_{\mathcal{A}_S} F$ for the source event classifier of Definition 92.7. Let A and B be target events in one carried target fiber with $A, B \in \mathcal{A}_T$, and suppose the public target readback displays $B \subseteq A$. Suppose the target complement/relative-difference row of Definition 92.16 supplies a measurable event*

$$D_T := A \setminus B$$

with public membership readback

$$t \in D_T \iff t \in A \text{ and } t \notin B.$$

Define source preimages

$$A_S := X^{-1}(A), \qquad B_S := X^{-1}(B), \qquad D_X := X^{-1}(D_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to its source measurable preimage, then $A_S, B_S, D_X \in \mathcal{A}_S$, and the public source readback displays $B_S \subseteq A_S$. Let $D_S := A_S \setminus B_S$ be the source relative-difference event supplied by the source* $\mathsf{Measure}^{\uparrow}$ *complement/relative-difference row. Then*

$$D_S \in \mathcal{A}_S, \qquad D_X \sim_{\mathcal{A}_S} D_S.$$

In the complement case, take the ambient event A to be Ω_T and write $B_T^{\complement} := \Omega_T \setminus B$. If the same $\mathsf{RandomVar}^{\uparrow}$ *row identifies $X^{-1}(\Omega_T)$ with Ω_S through the source event classifier, with source-event transport read through Definition 26.3, and if $B_S^{\complement} := \Omega_S \setminus B_S$ is the source complement event supplied by the source complement/relative-difference row, then*

$$X^{-1}(B_T^{\complement}) \sim_{\mathcal{A}_S} B_S^{\complement}.$$

The two classifications are certified with the corresponding source relative-difference ledgers: the second component is disjoint from B_S, and its displayed binary union with B_S is classified as the ambient source event.

Proof. The measurable-preimage row of the $\mathsf{RandomVar}^{\uparrow}$ certificate applies to the three target measurable events A, B, and D_T. This gives

$$A_S, B_S, D_X \in \mathcal{A}_S.$$

To read the source inclusion, fix a carried source point s with a witness $s \in B_S$. The preimage readback for B gives $X(s) \in B$. The displayed target inclusion $B \subseteq A$ gives $X(s) \in A$, and the preimage readback for A gives $s \in A_S$. Hence the public source readback displays $B_S \subseteq A_S$.

Apply the source complement/relative-difference row of Definition 92.16 to this displayed source inclusion. It supplies the measurable event $D_S := A_S \setminus B_S$, the disjointness witness for B_S and D_S, and the binary-union readback classifying $B_S \cup D_S$ as A_S. Its relative-difference readback is

$$s \in D_S \iff s \in A_S \text{ and } s \notin B_S.$$

For every carried source point s, the displayed rows give

$$\begin{aligned} s \in D_X &\iff X(s) \in D_T \\ &\iff X(s) \in A \text{ and } X(s) \notin B \\ &\iff s \in A_S \text{ and } s \notin B_S \\ &\iff s \in D_S. \end{aligned}$$

The first and third equivalences are the preimage readbacks, the second is the target relative-difference readback, and the last is the source relative-difference readback. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$D_X \sim_{\mathcal{A}_S} D_S.$$

Now take $A = \Omega_T$ and $D_T = B_T^{\complement}$. The source ambient preimage is $A_S = X^{-1}(\Omega_T)$, and the source-event transport is the one cited in Definition 26.3. For every carried source point s, the total-preimage classifier gives

$$s \in A_S \quad \iff \quad s \in \Omega_S.$$

The source complement event $B_S^{\complement} := \Omega_S \setminus B_S$ has readback

$$s \in B_S^{\complement} \quad \iff \quad s \in \Omega_S \text{ and } s \notin B_S.$$

Combining these two readbacks with the relative-difference computation above gives

$$s \in X^{-1}(B_T^{\complement}) \quad \iff \quad s \in B_S^{\complement}$$

for every carried source point s. The same source event-classifier component, using the source-event transport of Definition 26.3, therefore yields

$$X^{-1}(B_T^{\complement}) \sim_{\mathcal{A}_S} B_S^{\complement}.$$

The disjointness and binary-union statements are the corresponding ledger fields of the source complement/relative-difference row. □

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage complement difference exactness`

Theorem 185.5 (RandomVar preimage of the empty event is empty)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write $\varnothing_S$ and $\varnothing_T$ for the source and target empty events, write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, and write $E \sim_{\mathcal{A}_S} F$ for the source event classifier of Definition 92.7. Suppose the source and target empty-event rows supplied by Definition 92.4 give*

$$\varnothing_S \in \mathcal{A}_S, \qquad \varnothing_T \in \mathcal{A}_T,$$

and their displayed public readbacks eliminate every carried membership witness in the corresponding empty event. Let

$$A_\varnothing := X^{-1}(\varnothing_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to a source measurable preimage with membership readback, then*

$$A_\varnothing \in \mathcal{A}_S, \qquad A_\varnothing \sim_{\mathcal{A}_S} \varnothing_S.$$

Proof. The target empty event is measurable, so the measurable-preimage row of the $\mathsf{RandomVar}^{\uparrow}$ certificate applies to $\varnothing_T$. This gives

$$A_\varnothing \in \mathcal{A}_S.$$

It remains to compare $A_\varnothing$ with the source empty event through the source event classifier. Fix a carried source point s. The preimage readback gives

$$s \in A_\varnothing \quad \Longleftrightarrow \quad X(s) \in \varnothing_T.$$

Because X is carried, $X(s)$ is a carried target point. Hence any witness $s \in A_\varnothing$ reads as a witness $X(s) \in \varnothing_T$, and the target empty-event row eliminates it. By empty elimination this gives the implication from $s \in A_\varnothing$ to $s \in \varnothing_S$.

Conversely, any witness $s \in \varnothing_S$ is eliminated by the source empty-event row. By empty elimination this gives the implication from $s \in \varnothing_S$ to $s \in A_\varnothing$. Therefore, for every carried source point s,

$$s \in A_\varnothing \quad \Longleftrightarrow \quad s \in \varnothing_S.$$

The two events have the same carried source memberships. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$A_\varnothing \sim_{\mathcal{A}_S} \varnothing_S.$$

□

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage empty event exactness`

Theorem 185.6 (RandomVar preimage symmetric-difference exactness)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, and write $E \sim_{\mathcal{A}_S} F$ for the source event classifier of Definition 92.7. Let A and B be target events in one carried target fiber with $A, B \in \mathcal{A}_T$. Suppose the target difference rows supply measurable events*

$$D_T^{A\setminus B} := A \setminus B, \qquad D_T^{B\setminus A} := B \setminus A$$

with public membership readbacks

$$t \in D_T^{A\setminus B} \quad \Longleftrightarrow \quad t \in A \text{ and } t \notin B,$$

$$t \in D_T^{B\setminus A} \quad \Longleftrightarrow \quad t \in B \text{ and } t \notin A.$$

Let Δ_T be the displayed target binary union of these two difference events, so

$$t \in \Delta_T \quad \Longleftrightarrow \quad t \in D_T^{A\setminus B} \text{ or } t \in D_T^{B\setminus A}.$$

Define source preimages

$$A_S := X^{-1}(A), \qquad B_S := X^{-1}(B), \qquad \Delta_X := X^{-1}(\Delta_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to a source measurable preimage with membership readback, then $A_S, B_S, \Delta_X \in \mathcal{A}_S$. Let the source difference rows supply measurable events*

$$D_S^{A\setminus B} := A_S \setminus B_S, \qquad D_S^{B\setminus A} := B_S \setminus A_S$$

with readbacks

$$s \in D_S^{A\setminus B} \iff s \in A_S \text{ and } s \notin B_S,$$
$$s \in D_S^{B\setminus A} \iff s \in B_S \text{ and } s \notin A_S.$$

Let Δ_S be the source binary union of $D_S^{A\setminus B}$ and $D_S^{B\setminus A}$ supplied by the source $\mathsf{Measure}^{\uparrow}$ *sigma-algebra row of Definition 92.4. Then*

$$\Delta_S \in \mathcal{A}_S, \qquad \Delta_X \sim_{\mathcal{A}_S} \Delta_S.$$

Moreover, the two source difference components are disjoint.

Proof. The target difference rows make $D_T^{A\setminus B}$ and $D_T^{B\setminus A}$ measurable, and the target binary-union row makes Δ_T measurable. Applying the measurable-preimage row of the $\mathsf{RandomVar}^{\uparrow}$ certificate to A, B, and Δ_T gives

$$A_S, B_S, \Delta_X \in \mathcal{A}_S.$$

The same row gives the three preimage readbacks

$$s \in A_S \iff X(s) \in A,$$
$$s \in B_S \iff X(s) \in B,$$
$$s \in \Delta_X \iff X(s) \in \Delta_T.$$

The source difference rows supply $D_S^{A\setminus B}, D_S^{B\setminus A} \in \mathcal{A}_S$ and the two displayed source readbacks. They are disjoint: if a carried source point s has witnesses in both source difference events, the first readback gives $s \in A_S$ and $s \notin B_S$, while the second gives $s \in B_S$ and $s \notin A_S$. Either negative component eliminates the corresponding positive witness.

Therefore the source sigma-algebra row of Definition 92.4 supplies the binary union Δ_S of the two source difference events, with

$$\Delta_S \in \mathcal{A}_S$$

and membership readback

$$s \in \Delta_S \iff s \in D_S^{A\setminus B} \text{ or } s \in D_S^{B\setminus A}.$$

It remains to compare Δ_X with Δ_S. For every carried source point s, the displayed rows give

$$\begin{aligned}
s \in \Delta_X &\iff X(s) \in \Delta_T \\
&\iff X(s) \in D_T^{A\setminus B} \text{ or } X(s) \in D_T^{B\setminus A} \\
&\iff (X(s) \in A \text{ and } X(s) \notin B) \text{ or } (X(s) \in B \text{ and } X(s) \notin A) \\
&\iff (s \in A_S \text{ and } s \notin B_S) \text{ or } (s \in B_S \text{ and } s \notin A_S) \\
&\iff s \in D_S^{A\setminus B} \text{ or } s \in D_S^{B\setminus A} \\
&\iff s \in \Delta_S.
\end{aligned}$$

The first equivalence is the preimage readback for Δ_T; the second is the target binary-union readback; the third uses the two target difference readbacks; the fourth uses the preimage readbacks for A and B in both positive and negative directions; the fifth uses the two source difference readbacks; and the last is the source binary-union readback.

Thus Δ_X and Δ_S have the same carried source memberships. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$\Delta_X \sim_{\mathcal{A}_S} \Delta_S.$$

□

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage symmetric difference exactness`

Theorem 185.7 (RandomVar preimage binary-intersection exactness)**.** *Let $X : S \to T$ be a carried* RandomVar$^{\uparrow}$ *map as in section 185.1, from a* ProbSpace$^{\uparrow}$ *source as in section 184.1 to a target* Measure$^{\uparrow}$ *carrier as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, and write $E \sim_{\mathcal{A}_S} F$ for the source event classifier of Definition 92.7. Let B, C, I_T be target events such that $B, C, I_T \in \mathcal{A}_T$ and the displayed target binary-intersection row reads I_T as $B \cap C$:*

$$t \in I_T \quad \Longleftrightarrow \quad t \in B \text{ and } t \in C.$$

Define source preimages

$$A_B := X^{-1}(B), \qquad A_C := X^{-1}(C), \qquad A_I := X^{-1}(I_T).$$

If the RandomVar$^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to its source measurable preimage with membership readback, then*

$$A_B, A_C, A_I \in \mathcal{A}_S.$$

Moreover, if I_S is the source binary intersection of A_B and A_C supplied by the source Measure$^{\uparrow}$ *sigma-algebra row of Definition 92.4, then*

$$I_S \in \mathcal{A}_S, \qquad A_I \sim_{\mathcal{A}_S} I_S.$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage binary intersection exactness`

Proof. The measurable-preimage row of the RandomVar$^{\uparrow}$ certificate applies to B, C, and I_T. This gives

$$A_B, A_C, A_I \in \mathcal{A}_S.$$

Since the source probability space inherits its measurable family from the underlying Measure$^{\uparrow}$ carrier, Definition 92.4 supplies the binary-intersection row for the measurable pair A_B, A_C. Let I_S be that displayed source intersection. Its row gives $I_S \in \mathcal{A}_S$ and the membership readback

$$s \in I_S \quad \Longleftrightarrow \quad s \in A_B \text{ and } s \in A_C.$$

It remains to compare A_I with I_S through the source event classifier. For a carried source point s, unfold the preimage row for I_T:

$$s \in A_I \quad \Longleftrightarrow \quad X(s) \in I_T.$$

The target binary-intersection readback gives

$$X(s) \in I_T \quad \Longleftrightarrow \quad X(s) \in B \text{ and } X(s) \in C.$$

Unfolding the two preimage rows for B and C gives

$$X(s) \in B \text{ and } X(s) \in C \quad \Longleftrightarrow \quad s \in A_B \text{ and } s \in A_C.$$

The source binary-intersection readback identifies the right side with $s \in I_S$. Thus A_I and I_S have the same carried source memberships. By the source event-classifier component of Definition 92.7, this pointwise readback is exactly

$$A_I \sim_{\mathcal{A}_S} I_S.$$

□

Theorem 185.8 (B-524; closure/bridge: RandomVar countable preimage intersection exactness)**.** *Let S, T be BEDC measurable-event certificates whose event carriers are $\mathcal{A}_S, \mathcal{A}_T$, whose event classifiers are $\equiv_S, \equiv_T$, and whose membership readbacks are written*

$$s \Vdash_S A, \qquad t \Vdash_T B.$$

Assume that the source event classifier is readback-exact in the following minimal sense:

$$(\forall s,\ s \Vdash_S E \Longleftrightarrow s \Vdash_S F) \Longrightarrow E \equiv_S F.$$

Let $X : S \to T$ be a carried $\mathsf{RandomVar}^{\uparrow}$ *map with preimage constructor*

$$X^{-1} : \mathcal{A}_T \longrightarrow \mathcal{A}_S$$

satisfying the RandomVar readback bridge

$$s \Vdash_S X^{-1}(B) \Longleftrightarrow X(s) \Vdash_T B$$

for every $B \in \mathcal{A}_T$. Let $(B_n)_{n\in\mathbb{N}}$ be a target measurable-event sequence, $B_n \in \mathcal{A}_T$, and let $I_T \in \mathcal{A}_T$ be an exact target intersection event:

$$t \Vdash_T I_T \Longleftrightarrow \forall n,\ t \Vdash_T B_n.$$

Define

$$A_n := X^{-1}(B_n), \qquad A_I := X^{-1}(I_T).$$

Let

$$\bigcap\nolimits_S A_\bullet$$

be any displayed source measurable event whose readback is exact for the preimage family:

$$s \Vdash_S \bigcap\nolimits_S A_\bullet \Longleftrightarrow \forall n,\ s \Vdash_S A_n.$$

The displayed source witness is an explicit hypothesis here; no primitive source countable-intersection row is read from the $\mathsf{Measure}^{\uparrow}$ *surface. Then each A_n and A_I lies in $\mathcal{A}_S$, and*

$$A_I \equiv_S \bigcap\nolimits_S A_\bullet.$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarCountablePreimageIntersection exactness`

Proof. For every n, $B_n \in \mathcal{A}_T$; hence $A_n = X^{-1}(B_n) \in \mathcal{A}_S$ by the carried RandomVar preimage-closure field. Likewise $I_T \in \mathcal{A}_T$, so $A_I = X^{-1}(I_T) \in \mathcal{A}_S$. It remains to identify A_I with the displayed source countable intersection. Fix a source endpoint s. The RandomVar readback bridge gives

$$s \Vdash_S A_I \Longleftrightarrow X(s) \Vdash_T I_T.$$

The target intersection certificate gives

$$X(s) \Vdash_T I_T \Longleftrightarrow \forall n,\ X(s) \Vdash_T B_n.$$

Applying the RandomVar readback bridge to each B_n gives

$$\forall n,\ X(s) \Vdash_T B_n \Longleftrightarrow \forall n,\ s \Vdash_S A_n.$$

Finally, the displayed source intersection witness gives

$$\forall n,\ s \Vdash_S A_n \iff s \Vdash_S \bigcap_S A_\bullet.$$

Thus, for all source endpoints s,

$$s \Vdash_S A_I \iff s \Vdash_S \bigcap_S A_\bullet.$$

Source readback-extensionality therefore yields

$$A_I \equiv_S \bigcap_S A_\bullet.$$

No probability-value, integration, or conditional-expectation field is used; the proof consumes only the explicit source intersection witness and RandomVar preimage-readback certificates, in accordance with the BEDC certificate posture for derived Measure-type structure. □

Corollary 185.9 (B-525; determinacy: independence of the target intersection witness)**.** *Under the hypotheses of Definition 185.8, suppose $I_T, I'_T \in \mathcal{A}_T$ are two exact target intersection events for the same sequence (B_n):*

$$t \Vdash_T I_T \iff \forall n,\ t \Vdash_T B_n, \qquad t \Vdash_T I'_T \iff \forall n,\ t \Vdash_T B_n.$$

Then

$$X^{-1}(I_T) \equiv_S X^{-1}(I'_T).$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarCountablePreimageIntersection witness determinacy`

Proof. Apply Definition 185.8 to I_T. It gives

$$X^{-1}(I_T) \equiv_S \bigcap_S X^{-1}(B_\bullet).$$

Apply the same theorem to I'_T. It gives

$$X^{-1}(I'_T) \equiv_S \bigcap_S X^{-1}(B_\bullet).$$

Symmetry and transitivity of the source event classifier yield

$$X^{-1}(I_T) \equiv_S X^{-1}(I'_T).$$

□

Proposition 185.10 (B-526; uniqueness: source greatest-lower-bound exactness)**.** *Keep the hypotheses of Definition 185.8. Define the source event preorder by*

$$E \preceq_S F \quad :\iff \quad \forall s,\ s \Vdash_S E \Rightarrow s \Vdash_S F.$$

Then $A_I = X^{-1}(I_T)$ is the greatest lower bound of the preimage family (A_n) up to $\equiv_S$. Explicitly:

$$A_I \preceq_S A_n \quad \textit{for every } n,$$

and whenever $C \in \mathcal{A}_S$ satisfies $C \preceq_S A_n$ for every n, one has

$$C \preceq_S A_I.$$

Consequently, if $K \in \mathcal{A}_S$ is any other event satisfying the same two lower-bound clauses, then

$$K \equiv_S A_I.$$

Proof. For $A_I \preceq_S A_n$, assume $s \Vdash_S A_I$. By the readback chain in Definition 185.8,

$$s \Vdash_S A_I \iff \forall m,\ s \Vdash_S A_m.$$

Taking $m = n$ gives $s \Vdash_S A_n$. Now let $C \in \mathcal{A}_S$ and assume $C \preceq_S A_n$ for every n. If $s \Vdash_S C$, then $s \Vdash_S A_n$ for every n. The same readback chain gives $s \Vdash_S A_I$. Hence $C \preceq_S A_I$. Finally, let K satisfy the same lower-bound clauses. Since $K \preceq_S A_n$ for every n, the previous paragraph gives $K \preceq_S A_I$. Since $A_I \preceq_S A_n$ for every n, the greatest-lower-bound clause for K gives $A_I \preceq_S K$. Thus, for every s,

$$s \Vdash_S K \iff s \Vdash_S A_I.$$

Source readback-extensionality yields $K \equiv_S A_I$. □

Proposition 185.11 (B-527; composition bridge: two-stage RandomVar countable intersections)**.** *Let*

$$S \xrightarrow{X} T \xrightarrow{Y} U$$

be carried $\mathsf{RandomVar}^{\uparrow}$ *maps with exact preimage readback bridges. Let* (C_n) *be a* U*-measurable event sequence and* I_U *an exact* U*-intersection event:*

$$u \Vdash_U I_U \iff \forall n,\ u \Vdash_U C_n.$$

Put

$$B_n := Y^{-1}(C_n), \qquad B_I := Y^{-1}(I_U),$$

and

$$A_n := X^{-1}(B_n) = X^{-1}(Y^{-1}(C_n)).$$

Assume the source event classifier on S *is readback-exact, and let*

$$\bigcap\nolimits_S A_\bullet$$

be a displayed source measurable event with exact readback

$$s \Vdash_S \bigcap\nolimits_S A_\bullet \iff \forall n,\ s \Vdash_S A_n.$$

Then B_I *is an exact* T*-intersection event for the sequence* (B_n)*, and*

$$X^{-1}(B_I) \equiv_S \bigcap\nolimits_S A_\bullet.$$

Equivalently, for the composite RandomVar $Y \circ X$*,*

$$(Y \circ X)^{-1}(I_U) \equiv_S \bigcap\nolimits_S (Y \circ X)^{-1}(C_\bullet).$$

Proof. First show that B_I is the exact T-intersection of the B_n. For every t,

$$t \Vdash_T B_I \iff Y(t) \Vdash_U I_U \iff \forall n,\ Y(t) \Vdash_U C_n \iff \forall n,\ t \Vdash_T B_n.$$

Thus B_I satisfies the target-intersection hypothesis of Definition 185.8 for the RandomVar $X : S \to T$. Applying that theorem gives

$$X^{-1}(B_I) \equiv_S \bigcap\nolimits_S X^{-1}(B_\bullet) = \bigcap\nolimits_S A_\bullet.$$

The composite-preimage identity is not an additional axiom: its readback is

$$s \Vdash_S X^{-1}(Y^{-1}(C)) \iff X(s) \Vdash_T Y^{-1}(C) \iff Y(X(s)) \Vdash_U C,$$

which is exactly the RandomVar readback bridge for $Y \circ X$. Substituting $C = I_U$ and $C = C_n$ gives the displayed composite form. □

Proposition 185.12 (B-528; obstruction: closure and classifier-respect alone do not imply B-524)**.** *The RandomVar readback bridge in Definition 185.8 cannot be replaced by mere event-carrier closure and event-classifier respect of a preimage-like transformer.*

Proof. Consider a finite source event skeleton with two source events

$$0_S,\ 1_S$$

distinguished by the source classifier. Let the displayed source intersection witness of the constant sequence $(0_S, 0_S, \ldots)$ be 0_S. Let the target event skeleton contain pairwise classifier-distinct events

$$I_T,\ B_0,\ B_1, \ldots$$

with I_T certified as the exact countable intersection of the B_n. Define a preimage-like transformer P on this displayed fragment by

$$P(I_T) := 1_S, \qquad P(B_n) := 0_S \quad \text{for every } n.$$

This transformer preserves event-carrier membership on the displayed fragment. It also respects the target classifier on that fragment, because the only classifier identifications are reflexive. However,

$$P(I_T) = 1_S,$$

whereas

$$\bigcap\nolimits_S P(B_\bullet) = \bigcap\nolimits_S (0_S, 0_S, \ldots) = 0_S.$$

Since 0_S and 1_S are classifier-distinct, the desired conclusion

$$P(I_T) \equiv_S \bigcap\nolimits_S P(B_\bullet)$$

fails. Thus the missing obligation is not a closure obligation and not a classifier-respect obligation. It is precisely the bridge obligation

$$s \Vdash_S X^{-1}(B) \Longleftrightarrow X(s) \Vdash_T B,$$

or an equivalent countable-meet preimage bridge. Without that bridge, the countable preimage intersection exactness theorem is not derivable. □

The $\mathsf{RandomVar}^{\uparrow}$ packet sits between the source probability surface of chapter 184 and the distribution readback surface of chapter 186. This dependency is substantive: the source sample carrier and probability ledger come from $\mathsf{ProbSpace}^{\uparrow}$, while the measurable-output route supplies the pushforward and readback surface later consumed by $\mathsf{Distribution}^{\uparrow}$. Thus a carried random variable is not an isolated measurable map; it is the finite route by which source probability rows are made available to distribution-facing consumers.

Theorem 185.13 (RandomVar ProbSpace Distribution sibling route)**.** *Every accepted* $\mathsf{RandomVar}^{\uparrow}$ *preimage packet that is later consumed by* $\mathsf{Distribution}^{\uparrow}$ *first reads the source probability carrier from chapter 184 and the target measurable-event surface from chapter 92. It then exports only the finite preimage rows, the source and target event classifiers,* hsame, Cont, Pkg, *and* NameCert *rows to the distribution readback surface of chapter 186. No host function equality, independent probability-space bridge, target event outside the displayed* $\mathsf{Measure}^{\uparrow}$ *surface, or distribution row outside the carried preimage readback is exported.*

Proof. Start with the preimage exactness surface of Definition 185.2, Definition 185.3, Definition 185.4, and Definition 185.5. Each row uses the same carried map $X : S \to T$, reads the source carrier through $\mathsf{ProbSpace}^{\uparrow}$, and reads the target event family through $\mathsf{Measure}^{\uparrow}$.

Next use the scoped carried preimage package of Definition 185.27. It collects the finite, total, terminal, and countable preimage rows over the same source and target classifiers, so the downstream consumer receives a single carried package rather than a second event lattice.

For a $\mathsf{Distribution}^{\uparrow}$ readback, retain only the rows that the distribution surface can consume: the displayed finite preimages, the classifiers that compare source and target events, and the structural hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. The source probability ledger remains the $\mathsf{ProbSpace}^{\uparrow}$ input, while the measurable target surface remains the $\mathsf{Measure}^{\uparrow}$ input.

Thus the sibling route runs from the probability carrier through the carried random-variable preimage packet to the distribution readback surface. Since every exported row has already appeared on that route, no host function equality, independent probability-space bridge, target event outside the displayed measurable surface, or off-route distribution row is available. □

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarProbSpaceDistribution sibling route`

Definition 185.14 (Total-readback certificate for a carried RandomVarUp)**.** *Let S, T be BEDC event interfaces with event carriers $\mathcal{E}_S, \mathcal{E}_T$, event classifiers*

$$A \simeq_S A', \qquad B \simeq_T B',$$

measurability predicates $\mathsf{Meas}_S, \mathsf{Meas}_T$, and total events

$$\Omega_S \in \mathcal{E}_S, \qquad \Omega_T \in \mathcal{E}_T.$$

A carried $\mathsf{RandomVar}^{\uparrow}$ map $X : S \to T$ is said to carry a total-readback certificate if it is equipped with a BEDC preimage readback relation

$$\mathsf{Pre}_X(B, A) \qquad (B \in \mathcal{E}_T,\ A \in \mathcal{E}_S)$$

and a chosen readback endpoint $\mathrm{pre}_X(B) \in \mathcal{E}_S$ satisfying

$$\mathsf{Pre}_X(B, \mathrm{pre}_X(B))$$

for each target event B, together with the following four clauses:

target-total measurability:	$\mathsf{Meas}_T(\Omega_T)$,
measurable-preimage readback:	$\mathsf{Meas}_T(B) \wedge \mathsf{Pre}_X(B, A) \Longrightarrow \mathsf{Meas}_S(A)$,
classifier-determinacy of readback:	$B \simeq_T C,\ \mathsf{Pre}_X(B, A),\ \mathsf{Pre}_X(C, A') \Longrightarrow A \simeq_S A'$,
carried-total bridge:	$\mathsf{Pre}_X(\Omega_T, \Omega_S)$.

The object $\mathrm{pre}_X(\Omega_T)$ is the BEDC source endpoint denoted informally by

$$X^{-1}(\Omega_T).$$

Lean4 (definition) `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate`

Theorem 185.15 (RandomVar total-event preimage exactness)**.** *Under the total-readback certificate of Definition 185.14, the chosen total-target preimage is source-measurable and source-event-classifier equal to the source total event:*

$$\mathsf{Meas}_S(\mathrm{pre}_X(\Omega_T)) \quad \textit{and} \quad \mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Proof. The chosen readback clause gives

$$\mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)).$$

Since $\mathsf{Meas}_T(\Omega_T)$, measurable-preimage readback yields

$$\mathsf{Meas}_S(\mathrm{pre}_X(\Omega_T)).$$

For classifier exactness, apply classifier-determinacy to the reflexive target-classifier witness

$$\Omega_T \simeq_T \Omega_T,$$

the chosen readback

$$\mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)),$$

and the carried-total bridge

$$\mathsf{Pre}_X(\Omega_T, \Omega_S).$$

The conclusion is precisely

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

□

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate total event preimage exactness`

Theorem 185.16 (Terminal readback uniqueness over the target total event)**.** *Assume the total-readback certificate of Definition 185.14. If $A \in \mathcal{E}_S$ is any source event satisfying*

$$\mathsf{Pre}_X(\Omega_T, A),$$

then

$$A \simeq_S \Omega_S.$$

Consequently, if A, A' are two source events with

$$\mathsf{Pre}_X(\Omega_T, A), \qquad \mathsf{Pre}_X(\Omega_T, A'),$$

then

$$A \simeq_S A'.$$

Proof. Apply classifier-determinacy to

$$\Omega_T \simeq_T \Omega_T, \qquad \mathsf{Pre}_X(\Omega_T, A), \qquad \mathsf{Pre}_X(\Omega_T, \Omega_S).$$

This gives $A \simeq_S \Omega_S$. The same argument gives $A' \simeq_S \Omega_S$. By symmetry and transitivity of the source event-classifier,

$$A \simeq_S \Omega_S \simeq_S A',$$

hence $A \simeq_S A'$. □

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate terminal readback uniqueness`

Proposition 185.17 (Exactness and the carried-total bridge under source-endpoint closure)**.** *Assume the data of Definition 185.14 except possibly the carried-total bridge. Suppose moreover that the readback relation is closed under source-event classifier replacement:*

$$\mathsf{Pre}_X(B, A) \wedge A \simeq_S A' \Longrightarrow \mathsf{Pre}_X(B, A').$$

Then the following are equivalent:

$$\mathsf{Pre}_X(\Omega_T, \Omega_S) \quad \Longleftrightarrow \quad \mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate carried bridge chosen preimage exactness iff`

Proof. The forward implication is the classifier part of Definition 185.15. Conversely, suppose

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

The chosen readback clause supplies

$$\mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)).$$

By source-endpoint closure of Pre_X along the displayed classifier witness, with source-event transport read through Definition 26.3, this readback transports to

$$\mathsf{Pre}_X(\Omega_T, \Omega_S).$$

Thus the carried-total bridge is not an additional hidden axiom once the chosen total preimage is already classifier-exact and readback is closed under source-classifier replacement. □

Theorem 185.18 (Minimal obstruction to total-event exactness)**.** *Assume $X : S \to T$ has chosen readbacks and classifier-determinacy of readback. If*

$$\mathsf{Meas}_T(\Omega_T), \qquad \mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)), \qquad \mathsf{Pre}_X(\Omega_T, \Omega_S)$$

all hold, then

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Equivalently, any failure of

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S$$

forces failure of at least one of the following three certificate components:

$$\mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)), \qquad \mathsf{Pre}_X(\Omega_T, \Omega_S), \qquad \textit{classifier-determinacy of } \mathsf{Pre}_X.$$

If in addition measurable-preimage readback is available, then failure of

$$\mathsf{Meas}_S(\mathrm{pre}_X(\Omega_T))$$

forces failure of at least one of

$$\mathsf{Meas}_T(\Omega_T), \qquad \mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)), \qquad \textit{measurable-preimage readback.}$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate minimal obstruction`

Proof. The first assertion is exactly the classifier-determinacy argument of Definition 185.15, stripped to its minimal hypotheses. Its contrapositive is the stated obstruction claim. For the measurability assertion, if

$$\mathsf{Meas}_T(\Omega_T) \quad \text{and} \quad \mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T))$$

hold, measurable-preimage readback gives

$$\mathsf{Meas}_S(\mathrm{pre}_X(\Omega_T)).$$

Therefore failure of this conclusion forces failure of one of the three displayed premises. □

Theorem 185.19 (Composition preserves total-event preimage exactness with a composite ledger)**.** *Let*

$$X : S \to T, \qquad Y : T \to U$$

be carried $\mathsf{RandomVar}^{\uparrow}$ *maps with preimage readback relations*

$$\mathsf{Pre}_X, \qquad \mathsf{Pre}_Y.$$

Define the composite readback ledger by

$$\mathsf{Pre}_{Y \circ X}(C, A) \quad :\Longleftrightarrow \quad \exists B \in \mathcal{E}_T \ \big(\mathsf{Pre}_Y(C, B) \wedge \mathsf{Pre}_X(B, A)\big).$$

Assume:

$$\mathsf{Pre}_Y(\Omega_U, \Omega_T), \qquad \mathsf{Pre}_X(\Omega_T, \Omega_S),$$

both component readbacks satisfy classifier-determinacy, and both satisfy measurable-preimage readback. If

$$\mathsf{Meas}_U(\Omega_U),$$

then

$$\mathsf{Meas}_S(\mathrm{pre}_{Y \circ X}(\Omega_U)) \quad \textit{and} \quad \mathrm{pre}_{Y \circ X}(\Omega_U) \simeq_S \Omega_S.$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalPreimage composition exactness`
↪ variant `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate composition total event preimage exactness`

Proof. Let

$$A := \mathrm{pre}_{Y \circ X}(\Omega_U).$$

By chosen composite readback,

$$\mathsf{Pre}_{Y \circ X}(\Omega_U, A).$$

Hence there exists $B \in \mathcal{E}_T$ such that

$$\mathsf{Pre}_Y(\Omega_U, B), \qquad \mathsf{Pre}_X(B, A).$$

Since $\mathsf{Meas}_U(\Omega_U)$, measurable-preimage readback for Y gives

$$\mathsf{Meas}_T(B).$$

Then measurable-preimage readback for X gives

$$\mathsf{Meas}_S(A).$$

It remains to prove classifier exactness. The displayed carried-total bridges produce a composite total bridge:

$$\mathsf{Pre}_{Y\circ X}(\Omega_U, \Omega_S)$$

with intermediate witness $B = \Omega_T$. We first record composite classifier-determinacy. Suppose

$$\mathsf{Pre}_{Y\circ X}(C, A_1), \qquad \mathsf{Pre}_{Y\circ X}(C, A_2).$$

Choose intermediate witnesses B_1, B_2 with

$$\mathsf{Pre}_Y(C, B_1), \quad \mathsf{Pre}_X(B_1, A_1),$$

$$\mathsf{Pre}_Y(C, B_2), \quad \mathsf{Pre}_X(B_2, A_2).$$

Classifier-determinacy for Y, applied to $C \simeq_U C$, gives

$$B_1 \simeq_T B_2.$$

Classifier-determinacy for X, applied to this target-event classifier witness and the two X-readbacks, gives

$$A_1 \simeq_S A_2.$$

Thus the composite readback is classifier-deterministic. Apply this composite classifier-determinacy to

$$\mathsf{Pre}_{Y\circ X}(\Omega_U, A) \quad \text{and} \quad \mathsf{Pre}_{Y\circ X}(\Omega_U, \Omega_S).$$

The result is

$$A \simeq_S \Omega_S,$$

that is,

$$\mathrm{pre}_{Y\circ X}(\Omega_U) \simeq_S \Omega_S.$$

□

Definition 185.20 (Total-defect event of a RandomVarUp)**.** *Assume the source event interface carries BEDC Boolean event operations*

$$\bot_S, \qquad A \mapsto A^{\complement}, \qquad A \wedge A'$$

with classifier-stable Boolean exactness. For a carried $\mathsf{RandomVar}^{\uparrow}$ *map* $X : S \to T$*, define the total-defect event*

$$\mathfrak{D}_X := \Omega_S \wedge \big(\mathrm{pre}_X(\Omega_T)\big)^{\complement}.$$

It is the source-side ledger of histories for which the target-total readback fails to cover the source total event.

Lean4 (definition) `BEDC.Derived.RandomVarUp.RandomVarTotalDefectEvent`

Theorem 185.21 (Total-event exactness is equivalent to vanishing total defect)**.** *Assume the source Boolean event interface satisfies the classifier-exact total-complement law*

$$A \simeq_S \Omega_S \quad \Longleftrightarrow \quad \Omega_S \wedge A^{\complement} \simeq_S \bot_S$$

for every source-measurable event A*. If*

$$\mathsf{Meas}_S(\mathrm{pre}_X(\Omega_T)),$$

then

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S \quad \Longleftrightarrow \quad \mathfrak{D}_X \simeq_S \bot_S.$$

In particular, under the hypotheses of Definition 185.15,

$$\mathfrak{D}_X \simeq_S \bot_S.$$

↪ variant BEDC.Derived.RandomVarUp.RandomVarTotalDefectEvent vanishing total exactness iff

Proof. Put

$$A = \mathrm{pre}_X(\Omega_T).$$

By definition,

$$\mathfrak{D}_X = \Omega_S \wedge A^{\complement}.$$

Since A is source-measurable, the classifier-exact total-complement law applies and gives

$$A \simeq_S \Omega_S \quad \Longleftrightarrow \quad \Omega_S \wedge A^{\complement} \simeq_S \bot_S.$$

Substituting the definition of $\mathfrak{D}_X$ gives the asserted equivalence. Under Definition 185.15, $A \simeq_S \Omega_S$; hence $\mathfrak{D}_X \simeq_S \bot_S$. □

Theorem 185.22 (Total-target reflection criterion)**.** *Let $X : S \to T$ satisfy the total-readback certificate of Definition 185.14. Suppose X is target-total reflective, meaning that for every target-measurable event B,*

$$\mathrm{pre}_X(B) \simeq_S \Omega_S \Longrightarrow B \simeq_T \Omega_T.$$

Then, for every target-measurable B,

$$\mathrm{pre}_X(B) \simeq_S \Omega_S \quad \Longleftrightarrow \quad B \simeq_T \Omega_T.$$

Lean4 ✓ BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate total target reflection criterion

Proof. The forward implication is exactly target-total reflectivity. Conversely, assume $B \simeq_T \Omega_T$. The chosen readbacks provide

$$\mathsf{Pre}_X(B, \mathrm{pre}_X(B)), \qquad \mathsf{Pre}_X(\Omega_T, \mathrm{pre}_X(\Omega_T)).$$

Classifier-determinacy of readback applied to $B \simeq_T \Omega_T$ yields

$$\mathrm{pre}_X(B) \simeq_S \mathrm{pre}_X(\Omega_T).$$

By Definition 185.15,

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Transitivity of $\simeq_S$ gives

$$\mathrm{pre}_X(B) \simeq_S \Omega_S.$$

□

Theorem 185.23 (Source coverage induced by total-event exactness)**.** *Let* $\Vdash_S (h, A)$ *denote source-event incidence of a carried source history* h *in a source event* A*. Assume incidence respects the source event classifier:*

$$A \simeq_S A' \wedge \Vdash_S (h, A) \Longrightarrow \Vdash_S (h, A'),$$

and assume the source total event covers every carried source history:

$$\mathsf{Src}_S(h) \Longrightarrow \Vdash_S (h, \Omega_S).$$

Under the total-readback certificate of Definition 185.14,

$$\mathsf{Src}_S(h) \Longrightarrow \Vdash_S (h, \mathrm{pre}_X(\Omega_T)).$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate source coverage`

Proof. Let $\mathsf{Src}_S(h)$. By total coverage,

$$\Vdash_S (h, \Omega_S).$$

By Definition 185.15,

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Symmetry gives

$$\Omega_S \simeq_S \mathrm{pre}_X(\Omega_T).$$

Incidence transport along this classifier witness, read through Definition 26.3, yields

$$\Vdash_S (h, \mathrm{pre}_X(\Omega_T)).$$

□

Proposition 185.24 (Partiality obstruction detected by the target-total preimage)**.** *Assume source-event incidence is classifier-stable and* Ω_S *covers every carried source history as in Definition 185.23. If there exists a carried source history* h *such that*

$$\mathsf{Src}_S(h) \quad \textit{and} \quad \neg \Vdash_S (h, \mathrm{pre}_X(\Omega_T)),$$

then X *cannot carry a total-readback certificate satisfying Definition 185.14. Equivalently, any carried* $\mathsf{RandomVar}^{\uparrow}$ *total-readback certificate forces the target-total preimage to cover the entire source carrier.*

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTotalReadbackCertificate partiality obstruction`

Proof. Suppose, toward contradiction, that X carries the certificate of Definition 185.14. Then Definition 185.23 applies and gives

$$\mathsf{Src}_S(h) \Longrightarrow \Vdash_S (h, \mathrm{pre}_X(\Omega_T)).$$

For the displayed h, this contradicts

$$\neg \Vdash_S (h, \mathrm{pre}_X(\Omega_T)).$$

Thus no such total-readback certificate can exist. □

Corollary 185.25 (Terminal preimage exactness is the unique no-loss boundary condition)**.** *Assume the source Boolean defect law of Definition 185.21 and the source incidence discipline of Definition 185.23. For a carried* $\mathsf{RandomVar}^{\uparrow}$ *map* $X : S \to T$ *satisfying measurable-preimage readback, the following three statements are equivalent whenever the relevant events are source-measurable:*

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S,$$

$$\mathfrak{D}_X \simeq_S \bot_S,$$

$$\forall h \ \big(\mathsf{Src}_S(h) \Longrightarrow \Vdash_S (h, \mathrm{pre}_X(\Omega_T))\big),$$

provided the source event classifier is incidence-complete in the following sense:

$$\Big(\forall h\, [\mathsf{Src}_S(h) \Rightarrow (\Vdash_S (h, A) \Longleftrightarrow \Vdash_S (h, A'))]\Big) \Longrightarrow A \simeq_S A'.$$

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarTerminalPreimage no loss boundary`
↪ variant `BEDC.Derived.RandomVarUp.RandomVarTerminalPreimage coverage defect iff`

Proof. The equivalence between

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S$$

and

$$\mathfrak{D}_X \simeq_S \bot_S$$

is Definition 185.21. If

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S,$$

then the coverage statement follows by the proof of Definition 185.23. Conversely, assume

$$\forall h \ \big(\mathsf{Src}_S(h) \Rightarrow \Vdash_S (h, \mathrm{pre}_X(\Omega_T))\big).$$

Since Ω_S covers every carried source history, for every h with $\mathsf{Src}_S(h)$ both

$$\Vdash_S (h, \Omega_S) \quad \text{and} \quad \Vdash_S (h, \mathrm{pre}_X(\Omega_T))$$

hold. Hence, on the source carrier,

$$\Vdash_S (h, \mathrm{pre}_X(\Omega_T)) \Longleftrightarrow \Vdash_S (h, \Omega_S).$$

Incidence-completeness of the event classifier gives

$$\mathrm{pre}_X(\Omega_T) \simeq_S \Omega_S.$$

Therefore all three conditions are equivalent. □

Theorem 185.26 (RandomVar countable preimage union exactness)**.** *Let* $X : S \to T$ *be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write* $\mathcal{A}_S$ *and* $\mathcal{A}_T$ *for the source and target measurable-event families, and write* $E \sim_{\mathcal{A}_S} F$ *for the source event classifier of Definition 92.7. Let* $B_\bullet$ *be a target measurable-event sequence, so* $B_n \in \mathcal{A}_T$ *for every* $n : \mathbb{N}$*, and let* $U_T \in \mathcal{A}_T$ *be the displayed target countable-union row*

$$t \in U_T \quad \Longleftrightarrow \quad \exists n : \mathbb{N} \ \ t \in B_n.$$

For each n*, set*

$$A_n := X^{-1}(B_n), \qquad A_U := X^{-1}(U_T).$$

If the $\mathsf{RandomVar}^{\uparrow}$ *certificate row of section 185.2 sends every target measurable event to its source measurable preimage with membership readback, then* $A_U \in \mathcal{A}_S$ *and* $A_n \in \mathcal{A}_S$ *for every* n*. Let* U_S *be the source countable union of the measurable sequence* $A_\bullet$ *supplied by the source* $\mathsf{Measure}^{\uparrow}$ *sigma-algebra row of Definition 92.4. Then*

$$U_S \in \mathcal{A}_S, \qquad A_U \sim_{\mathcal{A}_S} U_S.$$

Moreover, if $B_\bullet$ *is pairwise disjoint in the target carried fiber, meaning that for* $m \neq n$ *no carried target point has witnesses in both* B_m *and* B_n*, then* $A_\bullet$ *is pairwise disjoint in the source carried fiber.*

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarPreimage countable union exactness`

Proof. The measurable-preimage row applies to U_T and to each target event B_n. Hence

$$A_U \in \mathcal{A}_S, \qquad A_n \in \mathcal{A}_S \quad \text{for every } n : \mathbb{N}.$$

Thus $A_\bullet$ is a source measurable sequence. The source $\mathsf{Measure}^{\uparrow}$ sigma-algebra row supplies its countable union U_S, with $U_S \in \mathcal{A}_S$ and membership readback

$$s \in U_S \quad \Longleftrightarrow \quad \exists n : \mathbb{N} \;\; s \in A_n.$$

Fix a carried source point s. The preimage and target countable-union readbacks give

$$s \in A_U \quad \Longleftrightarrow \quad X(s) \in U_T \quad \Longleftrightarrow \quad \exists n : \mathbb{N} \;\; X(s) \in B_n.$$

For each index n, the preimage readback for B_n gives

$$X(s) \in B_n \quad \Longleftrightarrow \quad s \in A_n.$$

Therefore

$$s \in A_U \quad \Longleftrightarrow \quad \exists n : \mathbb{N} \;\; s \in A_n \quad \Longleftrightarrow \quad s \in U_S.$$

The two events have the same carried source memberships. By the source event-classifier component of Definition 92.7, this is exactly

$$A_U \sim_{\mathcal{A}_S} U_S.$$

Assume $B_\bullet$ is pairwise disjoint. Let $m, n : \mathbb{N}$ with $m \neq n$, and suppose a carried source point s has witnesses $s \in A_m$ and $s \in A_n$. The two preimage readbacks give witnesses $X(s) \in B_m$ and $X(s) \in B_n$. Since $X(s)$ is a carried target point, target pairwise disjointness eliminates this pair. Hence no carried source point can inhabit both A_m and A_n for distinct m, n, which is exactly pairwise disjointness of $A_\bullet$ in the source fiber. □

Theorem 185.27 (RandomVar scoped closure package)**.** *Let* $X : S \to T$ *be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map from a displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source to a displayed* $\mathsf{Measure}^{\uparrow}$ *target, with the source-event classifier transported through the certificate rows of section 185.2 and with endpoint comparisons read through the relevant* $\mathsf{Real}^{\uparrow}$ *classifiers. The carried* $\mathsf{RandomVar}^{\uparrow}$ *surface supplies the following scoped package:*

1. *finite measurable-preimage exactness, including disjoint binary unions, relative differences, complements over the total event, and the empty event;*
2. *total-event exactness for* $\mathrm{pre}_X(\Omega_T)$*, uniqueness of the total-readback boundary, and source coverage forced by the target-total preimage;*

3. *terminal no-loss equivalence at the target total event; and*
4. *countable measurable-preimage union exactness, with pairwise disjointness transported from the target sequence to the source sequence.*

This is the scoped carried preimage package available to downstream Distribution$^{\uparrow}$, Independence$^{\uparrow}$, *and* CondExp$^{\uparrow}$ *constructions over the same displayed source and target rows.*

Proof. The finite preimage rows are the checked component sites Definition 185.2, Definition 185.3, Definition 185.4, and Definition 185.5. They share the same carried map X, the same source-event classifier, and the same measurable-preimage readback, so they form one finite exactness surface.

For the total boundary, Definition 185.15 identifies the target-total preimage with the source total event, Definition 185.16 gives uniqueness of the total-readback boundary, and Definition 185.23 turns the exactness row into source coverage by the target-total preimage. The no-loss boundary is the equivalence recorded in Definition 185.25.

Finally, Definition 185.26 supplies the countable union row and transports pairwise disjointness from the target sequence to the source sequence. Collecting these rows gives exactly the carried measurable-preimage and total-readback package stated above; no additional bridge to standard probability spaces is used. □

Closure status for NameCert$_{\text{RandomVar}^{\uparrow}}$.

Constructive story: A random variable in this package is a carried BEDC map from a displayed ProbSpace$^{\uparrow}$ source to a displayed Measure$^{\uparrow}$ target, accepted only when every target measurable event named by the certificate has a source measurable preimage with a public membership readback. Two carried maps are compared through the source and target event classifiers, not by host function extensionality: their visible preimage rows must classify the same source events, and the relevant Real$^{\uparrow}$ endpoint comparisons are read through the measure and probability ledgers already on the surface. The source specification covers carried maps together with the source probability carrier, target measure carrier, total-event row, and measurable-preimage operation, while the pattern specification records finite preimage exactness, complement and relative-difference exactness, empty-event exactness, terminal no-loss at the target total event, and countable-union exactness. The ledger policy keeps each target event, its source preimage, the pointwise membership equivalence, disjointness transport, total-event defect row, and countable-union witness visible, so no probability-space bridge is needed to recover what was checked. The five name-certificate obligations follow from the supplied carried map and from the event classifiers of ProbSpace$^{\uparrow}$ and Measure$^{\uparrow}$: the source and target carriers give inhabited rows when a map is displayed, reflexivity, symmetry, and transitivity are inherited by the preimage event comparisons, and carrier closure is exactly transport of measurable preimages along those classifiers. A map history h lies in the signature gap of a RandomVar$^{\uparrow}$ token p when the source-domain accepts h as a carried measurable map, the signature relation pairs it with an answer preimage package and evidence ledger, and the token-introduction relation names that package as p; unfolded, the preimage of every finite, terminal, or countable target event is the source event encoded by p with the same classifier readback. The chapter registers this carried preimage package under the name RandomVar$^{\uparrow}$; it introduces no independence theory, conditional expectation, distribution package beyond preimages, host function equality, or standard probability-space bridge.

Theory closure: TheoryGate(NameCert$_{\text{RandomVar}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope is the finite, total, terminal, and countable preimage package for carried RandomVar$^{\uparrow}$ maps over displayed ProbSpace$^{\uparrow}$, Measure$^{\uparrow}$, and Real$^{\uparrow}$ endpoint classifiers.

Formal verification: FormalStatus(NameCert$_{\text{RandomVar}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RandomVarUp.RandomVarPreimage\_countable\_union\_exactness`

Bridge status: paperBridge

Not claimed: The chapter does not close independence, conditional expectation, distributions beyond the preimage package, or standard probability-space bridges.

Upgrade path: The remaining work is Lean-side bridgeCheckedV for a standard probability-space bridge; no higher theory-axis grade remains.

Theorem 185.28 (RandomVar martingale-filtration handoff)**.** *Every* Martingale$^{\uparrow}$ *or* StoppingTime$^{\uparrow}$ *consumer of a* RandomVar$^{\uparrow}$ *endpoint must read the source* ProbSpace$^{\uparrow}$ *row, event-preimage exactness rows, total-readback certificate, filtration visibility row, componentwise* hsame *transport,*

displayed Cont *replay,* Pkg *provenance, and local* NameCert *row before conditional-expectation or stopping decisions are exposed. The handoff exports no host measurable function, completed sigma-algebra, almost-sure quotient equality, or unbounded stopping choice.*

Proof. Start with the carrier and certificate surface of section 185.1 and section 185.2. The source probability row is supplied by chapter 184, and the preimage rows are the finite event-readback rows recorded in Definition 185.1.

Read the exactness children next. The disjoint-union, relative-difference, complement, empty-event, binary-intersection, terminal, and countable preimage rows in the existing random-variable files keep every target-event observation inside the displayed source event ledger. The total-readback certificate of Definition 185.14 supplies the terminal source coverage needed before an endpoint can be treated as a process value.

For a martingale consumer, the endpoint row is then exposed only through the finite process surface of chapter 190. A stopping-time consumer reads the same endpoint only through the filtration-prefix route of chapter 344, so the visibility row is a finite ledger read rather than a host filtration.

Finally replay the structural rows. Componentwise hsame, displayed Cont, Pkg, and the local NameCert preserve the same source-probability, preimage, total-readback, and filtration-visibility route before $\mathsf{CondExp}^{\uparrow}$ or bounded stopping data are consumed. Since no row stores a host measurable function, completed sigma-algebra, almost-sure quotient equality, or unbounded stopping choice, the handoff cannot export them. □

Lean4 ✓ `BEDC.Derived.RandomVarUp.RandomVarMartingaleFiltration handoff`

Chapter 186

A Concrete Naming Certificate for $\mathsf{Distribution}^{\uparrow}$

The distribution interface $\mathsf{Distribution}^{\uparrow}$ packages the pushforward measure of a random variable on its target space. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92).

186.1 Carrier definition

Definition 186.1 (Distribution pushforward carrier)**.** *A carried* $\mathsf{Distribution}^{\uparrow}$ *datum consists of a carried* $\mathsf{RandomVar}^{\uparrow}$ *map* $X : S \to T$ *as in section 185.1, a* $\mathsf{ProbSpace}^{\uparrow}$ *source carrier whose measurable-event family and probability measure are written* $\mathcal{A}_S$ *and* μ_S*, and a target* $\mathsf{Measure}^{\uparrow}$ *carrier whose measurable-event family is written* $\mathcal{A}_T$*. For each target measurable event* $B \in \mathcal{A}_T$*, the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row supplies a source measurable event*

$$X^{-1}(B) \in \mathcal{A}_S.$$

The pushed-forward endpoint is the RealUp-valued row

$$\mu_X(B) := \mu_S(X^{-1}(B)).$$

Thus the $\mathsf{Distribution}^{\uparrow}$ *carrier is the target measurable family* $\mathcal{A}_T$ *together with the endpoint assignment* $B \mapsto \mu_X(B)$ *whose source is the probability measure of the carried source.*

Lean4 (definition) `BEDC.Derived.DistributionUp.DistributionPushforwardSourceSpec`

Definition 186.2 (Distribution pushforward classifier)**.** *Let* $(\mathcal{A}_T, \mu_X)$ *and* $(\mathcal{A}'_T, \mu'_Y)$ *be two displayed* $\mathsf{Distribution}^{\uparrow}$ *pushforward rows carried by target event classifiers* $\sim_{\mathcal{A}_T}$ *and by the* $\mathsf{Real}^{\uparrow}$ *classifier* $\sim_R$ *of Definition 38.9. The rows are classified when, for every pair of displayed target measurable events* $B \in \mathcal{A}_T$ *and* $B' \in \mathcal{A}'_T$ *with*

$$B \sim_{\mathcal{A}_T} B',$$

their pushed-forward endpoints agree under the real classifier:

$$\mu_X(B) \sim_R \mu'_Y(B').$$

Equivalently, the event classifier decides which target events may be compared, and the RealUp classifier decides equality of their measured pushforward endpoints.

Lean4 (definition) `BEDC.Derived.DistributionUp.DistributionPushforwardClassifier`
Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforwardClassifier row`

186.2 The certificate

Theorem 186.3 (Distribution pushforward row)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map from a* $\mathsf{ProbSpace}^{\uparrow}$ *source to a target* $\mathsf{Measure}^{\uparrow}$ *carrier, and let μ_X be the* $\mathsf{Distribution}^{\uparrow}$ *endpoint assignment of Definition 186.1. Write $\mathcal{A}_T$ for the target measurable-event family, μ_S for the source probability measure, and $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier. Then for every target measurable event $B \in \mathcal{A}_T$,*

$$\mu_X(B) \sim_R \mu_S(X^{-1}(B)).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward row`

Proof. By Definition 186.1, the endpoint $\mu_X(B)$ is defined by applying the source probability measure to the measurable preimage supplied by the $\mathsf{RandomVar}^{\uparrow}$ row:

$$\mu_X(B) := \mu_S(X^{-1}(B)).$$

The measurable-preimage row ensures that $X^{-1}(B)$ is a legitimate source measurable event before the source measure is read. The source $\mathsf{ProbSpace}^{\uparrow}$ carrier supplies the measure row μ_S on that event.

The displayed equality is therefore an endpoint identity in $\mathsf{Real}^{\uparrow}$. Reflexivity of the $\mathsf{Real}^{\uparrow}$ classifier gives

$$\mu_S(X^{-1}(B)) \sim_R \mu_S(X^{-1}(B)).$$

Transport along the definitional carrier equality reads this as

$$\mu_X(B) \sim_R \mu_S(X^{-1}(B)).$$

□

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforwardCarrier row`

Theorem 186.4 (Distribution NameCert obligation packet)**.** *The* $\mathsf{Distribution}^{\uparrow}$ *pushforward surface supplies the NameCert obligation packet for a carried random variable $X : S \to T$ over a source* $\mathsf{ProbSpace}^{\uparrow}$ *carrier and a target* $\mathsf{Measure}^{\uparrow}$ *carrier. Its ten-field roadmap fit is:*

1. ***Traditional target****: the distribution law of X, regarded as the pushforward probability measure on target measurable events.*
2. ***BEDC source****: the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage certificate, the source* $\mathsf{ProbSpace}^{\uparrow}$ *measure rows, the target* $\mathsf{Measure}^{\uparrow}$ *event classifier, and the shared* $\mathsf{Real}^{\uparrow}$ *classifier.*
3. ***Carrier choice****: the target measurable family $\mathcal{A}_T$ equipped with the endpoint assignment $B \mapsto \mu_S(X^{-1}(B))$ from Definition 186.1.*
4. ***Classifier choice****: the row classifier of Definition 186.2, combining target-event classification with RealUp endpoint classification.*
5. ***Stability obligations****: target-event transport through the* $\mathsf{Measure}^{\uparrow}$ *classifier, source-preimage transport through the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage rows, source-measure transport through the source* $\mathsf{Measure}^{\uparrow}$ *ledger, and reflexive, symmetric, transitive, and congruent use of the* $\mathsf{Real}^{\uparrow}$ *classifier.*
6. ***Ledger****: the source event $X^{-1}(B)$ is retained as the hidden witness explaining each target endpoint, while the public row exposes only the target event B and the real endpoint $\mu_X(B)$.*

7. ***Core theorem list**: the pushforward row of Definition 186.3, total-mass inheritance Definition 186.5, empty-event inheritance Definition 186.6, finite additivity Definition 186.7, monotonicity Definition 186.11, probability bounds Definition 186.12, complement-mass decomposition Definition 186.13, countable disjoint sigma-additivity Definition 186.14, and continuity from below Definition 186.15.*

8. ***Exactness predicate**: for every target measurable event B, the row exactness predicate is $\mu_X(B) \sim_R \mu_S(X^{-1}(B))$, with target-event classifier transport and source-event ledger transport preserving the displayed endpoint.*

9. ***Standard bridge**: the ordinary measure-theoretic pushforward law $X_*\mu_S$ on the target measurable space, read conservatively through the target event classifier and the RealUp endpoint classifier.*

10. ***Status**: the packet records the carrier and certificate obligations at the paper level, with Lean formalization reserved for the* BEDC.*Derived.DistributionUp namespace.*

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforwardCarrier semantic name certificate`

Proof. The carrier and classifier fields are exactly Definition 186.1 and Definition 186.2. The carrier uses the $\mathsf{RandomVar}^{\uparrow}$ measurable-preimage row to make $X^{-1}(B)$ a source measurable event, and then reads the source $\mathsf{ProbSpace}^{\uparrow}$ measure row at that event.

The stability and exactness fields follow from the same chain. Target-event transport is supplied by the $\mathsf{Measure}^{\uparrow}$ event classifier; source-preimage transport is supplied by the RandomVarUp measurable-preimage certificate; source-measure transport is read through the source $\mathsf{Measure}^{\uparrow}$ ledger; endpoint comparison is composed by RealUp classifier reflexivity and transitivity.

The ledger field records that the source event is kept as hidden witness data for each public target endpoint. This is precisely the information consumed by the downstream mass, empty-event, additivity, monotonicity, sigma-additivity, and continuity-from-below rows in this chapter.

The standard bridge is the usual pushforward measure surface, but only as a conservative readback of the displayed BEDC rows. No host quotient or host event equality is imported into the carrier; all comparisons pass through the target-event and RealUp classifiers. □

Theorem 186.5 (Distribution pushforward total mass is the probability unit)**.** *Let X be a carried $\mathsf{RandomVar}^{\uparrow}$ map as in section 185.1, from a $\mathsf{ProbSpace}^{\uparrow}$ source as in section 184.1 to a $\mathsf{Measure}^{\uparrow}$ target as in section 92.1. Write Ω_S for the source total event, Ω_T for the target total event, $\mathcal{A}_S, \mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the $\mathsf{Distribution}^{\uparrow}$ pushforward measure as in section 186.1, 1_R for the probability unit, and $x \sim_R y$ for the $\mathsf{Real}^{\uparrow}$ classifier of Definition 38.9. Suppose:*

- *$\Omega_S \in \mathcal{A}_S$ and $\Omega_T \in \mathcal{A}_T$;*
- *the $\mathsf{RandomVar}^{\uparrow}$ certificate row of section 185.2 identifies the target-total preimage as the source total event,*
$$X^{-1}(\Omega_T) = \Omega_S;$$
- *the $\mathsf{Distribution}^{\uparrow}$ pushforward row of section 186.2 gives, at every target measurable event $B \in \mathcal{A}_T$,*
$$\mu_X(B) \sim_R \mu_S(X^{-1}(B));$$
- *the $\mathsf{ProbSpace}^{\uparrow}$ total-mass row of section 184.2 gives*
$$\mu_S(\Omega_S) \sim_R 1_R.$$

Then

$$\mu_X(\Omega_T) \sim_R 1_R.$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward total mass unit`
↪ variant `BEDC.Derived.DistributionUp.DistributionPushforwardWitness total mass unit`

Proof. Apply the pushforward row to the target total event $B := \Omega_T$. Since $\Omega_T \in \mathcal{A}_T$, this gives

$$\mu_X(\Omega_T) \sim_R \mu_S(X^{-1}(\Omega_T)).$$

The random-variable target-total row identifies $X^{-1}(\Omega_T)$ with Ω_S as a measurable source event, and source-event transport through Definition 26.3 permits the source-measure value to be read at the transported event. Thus the preceding comparison reads

$$\mu_X(\Omega_T) \sim_R \mu_S(\Omega_S).$$

The probability-space total-mass row gives

$$\mu_S(\Omega_S) \sim_R 1_R.$$

The relation $\sim_R$ is the $\mathsf{Real}^\uparrow$ classifier, hence its transitivity field composes these two classifications and yields

$$\mu_X(\Omega_T) \sim_R 1_R.$$

□

Theorem 186.6 (Distribution empty target event has zero mass)**.** *Let X be a carried* $\mathsf{RandomVar}^\uparrow$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^\uparrow$ *source as in section 184.1 to a* $\mathsf{Measure}^\uparrow$ *target as in section 92.1. Write $\varnothing_S$ and $\varnothing_T$ for the source and target empty events, $\mathcal{A}_S, \mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^\uparrow$ *pushforward measure as in section 186.1, 0_R for the additive zero in* $\mathsf{Real}^\uparrow$*, and $x \sim_R y$ for the* $\mathsf{Real}^\uparrow$ *classifier of Definition 38.9. Suppose:*

- *$\varnothing_S \in \mathcal{A}_S$ and $\varnothing_T \in \mathcal{A}_T$;*
- *the* $\mathsf{RandomVar}^\uparrow$ *certificate row of section 185.2 identifies the target-empty preimage as the source empty event,*
$$X^{-1}(\varnothing_T) = \varnothing_S;$$
- *the* $\mathsf{Distribution}^\uparrow$ *pushforward row of section 186.2 gives, at every target measurable event $B \in \mathcal{A}_T$,*
$$\mu_X(B) \sim_R \mu_S(X^{-1}(B));$$
- *the source* $\mathsf{Measure}^\uparrow$ *certificate and the shared* $\mathsf{Real}^\uparrow$ *additive* $\mathsf{AbGroup}^\uparrow$ *row satisfy the hypotheses of Definition 92.25 for $\mathcal{A}_S, \mu_S, \varnothing_S$, with additive zero 0_R.*

Then

$$\mu_X(\varnothing_T) \sim_R 0_R.$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward empty target event zero mass`

Proof. Apply the pushforward row to the target empty event $B := \varnothing_T$. Since $\varnothing_T \in \mathcal{A}_T$, this gives

$$\mu_X(\varnothing_T) \sim_R \mu_S(X^{-1}(\varnothing_T)).$$

The random-variable target-empty row identifies $X^{-1}(\varnothing_T)$ with $\varnothing_S$ as a measurable source event, and source-event transport through Definition 26.3 permits the source-measure value to be read at the transported event. Thus the preceding comparison reads

$$\mu_X(\varnothing_T) \sim_R \mu_S(\varnothing_S).$$

By Definition 92.25 applied to the source $\mathsf{Measure}^{\uparrow}$ instance,

$$\mu_S(\varnothing_S) \sim_R 0_R.$$

The relation $\sim_R$ is the $\mathsf{Real}^{\uparrow}$ classifier, hence its transitivity field composes these two classifications and yields

$$\mu_X(\varnothing_T) \sim_R 0_R.$$

□

Theorem 186.7 (Distribution pushforward finite disjoint-union additivity)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a* $\mathsf{Measure}^{\uparrow}$ *target as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section 186.1, and $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let B, C, U_T be target events such that $B, C, U_T \in \mathcal{A}_T$ and the displayed target binary-union row reads U_T as $B \cup C$:*

$$t \in U_T \quad \Longleftrightarrow \quad t \in B \text{ or } t \in C.$$

Assume B and C are disjoint. Define

$$A_B := X^{-1}(B), \qquad A_C := X^{-1}(C), \qquad A_U := X^{-1}(U_T).$$

Let U_S be the source binary union of A_B and A_C supplied by the source $\mathsf{Measure}^{\uparrow}$ *sigma-algebra row. Suppose:*

- *the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row and the source sigma-algebra row satisfy the hypotheses of Definition 185.2 for B, C, U_T;*
- *the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row of section 186.2 gives, at every target measurable event $E \in \mathcal{A}_T$,*
$$\mu_X(E) \sim_R \mu_S(X^{-1}(E));$$
- *the source* $\mathsf{Measure}^{\uparrow}$ *certificate and the shared* $\mathsf{Real}^{\uparrow}$ *additive* $\mathsf{AbGroup}^{\uparrow}$ *row satisfy the hypotheses of Definition 92.26 for the disjoint measurable pair A_B, A_C and its measurable binary union U_S;*
- *source-event transport through Definition 26.3 permits μ_S to be read across the source event classifier supplied by Definition 92.7.*

Then

$$\mu_X(U_T) \sim_R \mu_X(B) + \mu_X(C).$$

Proof. Apply Definition 185.2 to the displayed target union $U_T = B \cup C$. It gives

$$A_B, A_C, A_U \in \mathcal{A}_S,$$

identifies U_S as a measurable source binary union, classifies the target-union preimage with that source union,

$$A_U \sim_{\mathcal{A}_S} U_S,$$

and shows that A_B and A_C are disjoint source events.

Apply the pushforward row to U_T, B, and C. This gives

$$\mu_X(U_T) \sim_R \mu_S(A_U),$$

$$\mu_X(B) \sim_R \mu_S(A_B), \qquad \mu_X(C) \sim_R \mu_S(A_C).$$

The source-event comparison $A_U \sim_{\mathcal{A}_S} U_S$, read through the source measure ledger by Definition 26.3, gives

$$\mu_S(A_U) \sim_R \mu_S(U_S).$$

Since A_B and A_C are disjoint measurable source events and U_S is their measurable binary union, Definition 92.26 applied to the source $\mathsf{Measure}^{\uparrow}$ carrier yields

$$\mu_S(U_S) \sim_R \mu_S(A_B) + \mu_S(A_C).$$

Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier composes the three comparisons into

$$\mu_X(U_T) \sim_R \mu_S(A_B) + \mu_S(A_C).$$

By symmetry of $\sim_R$, the pushforward comparisons for B and C also read

$$\mu_S(A_B) \sim_R \mu_X(B), \qquad \mu_S(A_C) \sim_R \mu_X(C).$$

The additive congruence field of the shared $\mathsf{Real}^{\uparrow}$ additive $\mathsf{AbGroup}^{\uparrow}$ row transports these two comparisons through $+$, giving

$$\mu_S(A_B) + \mu_S(A_C) \sim_R \mu_X(B) + \mu_X(C).$$

A final transitivity step gives

$$\mu_X(U_T) \sim_R \mu_X(B) + \mu_X(C).$$

□

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward finite disjoint union additivity`

Theorem 186.8 (Distribution nonnegative value inheritance)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a* $\mathsf{Measure}^{\uparrow}$ *target as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section 186.1, $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9, and* NonNeg_R *for the nonnegative predicate exposed by the shared* $\mathsf{RealAlgOrder}$ *row. Let B be a target event with $B \in \mathcal{A}_T$, and define*

$$A_B := X^{-1}(B).$$

Suppose:

- *the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row of section 185.2 sends the target measurable event* B *to the source measurable event* A_B*, so* $A_B \in \mathcal{A}_S$*;*
- *the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row of section 186.2 gives, at the target event* B*,*

$$\mu_X(B) \sim_R \mu_S(A_B);$$

- *the source* $\mathsf{Measure}^{\uparrow}$ *component of the* $\mathsf{ProbSpace}^{\uparrow}$ *source exposes the nonnegative value row of Definition 92.98, and the same* $\mathsf{RealAlgOrder}$ *nonnegative predicate is stable under the* $\mathsf{Real}^{\uparrow}$ *classifier.*

Then

$$\mathsf{NonNeg}_R(\mu_X(B)).$$

Proof. The measurable-preimage row makes A_B a displayed measurable event in the source measurable family. Reading the source probability space through its underlying $\mathsf{Measure}^{\uparrow}$ component, Definition 92.98 supplies

$$\mathsf{NonNeg}_R(\mu_S(A_B)).$$

The pushforward row identifies the pushed-forward endpoint with the source endpoint:

$$\mu_X(B) \sim_R \mu_S(A_B).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier, this also reads

$$\mu_S(A_B) \sim_R \mu_X(B).$$

Stability of the $\mathsf{RealAlgOrder}$ nonnegative predicate under $\sim_R$ transports the displayed source nonnegative row across this classifier comparison, giving

$$\mathsf{NonNeg}_R(\mu_X(B)).$$

□

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward nonnegative value inheritance`

Theorem 186.9 (Distribution pushforward monotone under target inclusion)**.** *Let* $X : S \to T$ *be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a* $\mathsf{Measure}^{\uparrow}$ *target as in section 92.1. Write* $\mathcal{A}_S$ *and* $\mathcal{A}_T$ *for the source and target measurable-event families,* μ_S *for the source measure,* μ_X *for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section 186.1, and* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* B *and* C *be target events in one carried target fiber with* $B, C \in \mathcal{A}_T$*, and suppose the public target readback displays* $B \subseteq C$*. Define*

$$A_B := X^{-1}(B), \qquad A_C := X^{-1}(C).$$

Suppose:

- *the target complement/relative-difference row of Definition 92.16, the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row, and the source complement/relative-difference row satisfy the hypotheses of Definition 185.3 for the ambient target event* C *and subevent* B*;*
- *the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row of section 186.2 gives, at every target measurable event* $E \in \mathcal{A}_T$*,*

$$\mu_X(E) \sim_R \mu_S(X^{-1}(E));$$

- *the source* $\mathsf{Measure}^{\uparrow}$ *certificate satisfies the hypotheses of Definition 92.99 for the displayed source inclusion* $A_B \subseteq A_C$ *and its measurable relative-difference row;*
- *the RealAlgOrder order row on* $\mathsf{Real}^{\uparrow}$ *is stable under* $\sim_R$ *on both endpoints.*

Then

$$\mu_X(B) \leq_R \mu_X(C).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward monotone under target inclusion`

Proof. Apply Definition 185.3 with the ambient target event $A := C$ and the subevent B. It gives

$$A_B, A_C \in \mathcal{A}_S$$

and reads the target inclusion $B \subseteq C$ as the source inclusion

$$A_B \subseteq A_C.$$

It also supplies the source relative-difference ledger: the event $A_C \setminus A_B$ is measurable, is disjoint from A_B, and its displayed binary union with A_B is classified as A_C in the source event classifier.

These are exactly the inclusion-decomposition inputs for Definition 92.99 in the source $\mathsf{Measure}^{\uparrow}$ carrier. Hence

$$\mu_S(A_B) \leq_R \mu_S(A_C).$$

Apply the pushforward row to B and C. Since both are target measurable,

$$\mu_X(B) \sim_R \mu_S(A_B), \qquad \mu_X(C) \sim_R \mu_S(A_C).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier, the first comparison also reads $\mu_S(A_B) \sim_R \mu_X(B)$. Transport the left endpoint of the source order comparison across this classifier equality, and then transport the right endpoint across $\mu_S(A_C) \sim_R \mu_X(C)$ using stability of the RealAlgOrder order row. The resulting comparison is

$$\mu_X(B) \leq_R \mu_X(C).$$

□

Theorem 186.10 (Distribution pushforward relative-difference additivity)**.** *Let* $X : S \to T$ *be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section 92.1. Write* $\mathcal{A}_S$ *and* $\mathcal{A}_T$ *for the source and target measurable-event families,* μ_S *for the source measure,* μ_X *for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section 186.1, and* $x \sim_R y$ *for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9. Let* A *and* B *be target events in one carried target fiber with* $A, B \in \mathcal{A}_T$*, and suppose the public target readback displays* $B \subseteq A$*. Suppose the target complement/relative-difference row of Definition 92.16 supplies a measurable target event*

$$D_T := A \setminus B$$

with public membership readback

$$t \in D_T \iff t \in A \text{ and } t \notin B.$$

Define source preimages

$$A_S := X^{-1}(A), \qquad B_S := X^{-1}(B), \qquad D_X := X^{-1}(D_T).$$

Let $D_S := A_S \setminus B_S$ *be the source relative-difference event supplied by the source* $\mathsf{Measure}^{\uparrow}$ *complement/relative-difference row once the source inclusion* $B_S \subseteq A_S$ *is displayed. Suppose:*

- *the* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row and the source complement/relative-difference row satisfy the hypotheses of Definition 185.3 for* A, B, D_T*;*
- *the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row of section 186.2 gives, at every target measurable event* $E \in \mathcal{A}_T$*,*
$$\mu_X(E) \sim_R \mu_S(X^{-1}(E));$$
- *the source* $\mathsf{Measure}^{\uparrow}$ *certificate and the shared* $\mathsf{Real}^{\uparrow}$ *additive* $\mathsf{AbGroup}^{\uparrow}$ *row satisfy the hypotheses of Definition 92.102 for the displayed source inclusion* $B_S \subseteq A_S$*;*
- *source-event transport through Definition 26.3 permits* μ_S *to be read across the source event classifier supplied by Definition 92.7;*
- *the shared* $\mathsf{Real}^{\uparrow}$ *additive* $\mathsf{AbGroup}^{\uparrow}$ *row supplies addition congruence for* $\sim_R$*.*

Then
$$\mu_X(D_T) \sim_R \mu_S(D_S)$$
and
$$\mu_X(A) \sim_R \mu_X(B) + \mu_X(D_T).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward relative difference additivity`

Proof. Apply Definition 185.3 to the displayed target inclusion $B \subseteq A$ and to the target relative-difference event D_T. It gives
$$A_S, B_S, D_X \in \mathcal{A}_S,$$
reads the target inclusion as the source inclusion $B_S \subseteq A_S$, supplies $D_S \in \mathcal{A}_S$, and classifies the preimage of the target difference with the source difference:
$$D_X \sim_{\mathcal{A}_S} D_S.$$
The same source relative-difference ledger states that B_S and D_S are disjoint and that the displayed binary union $B_S \cup D_S$ is classified as A_S.

Apply the pushforward row to the three target measurable events A, B, and D_T. This gives
$$\mu_X(A) \sim_R \mu_S(A_S),$$
$$\mu_X(B) \sim_R \mu_S(B_S), \qquad \mu_X(D_T) \sim_R \mu_S(D_X).$$
Transport the source measure endpoint across $D_X \sim_{\mathcal{A}_S} D_S$ using the source event ledger. Hence
$$\mu_S(D_X) \sim_R \mu_S(D_S).$$
Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier composes the pushforward comparison for D_T with this transport and yields
$$\mu_X(D_T) \sim_R \mu_S(D_S).$$

The source relative-difference ledger gives exactly the inputs for Definition 92.102 in the source $\mathsf{Measure}^{\uparrow}$ carrier. Therefore
$$\mu_S(A_S) \sim_R \mu_S(B_S) + \mu_S(D_S).$$
Composing with the pushforward comparison for A gives
$$\mu_X(A) \sim_R \mu_S(B_S) + \mu_S(D_S).$$

By symmetry of $\sim_R$, the pushforward comparison for B reads $\mu_S(B_S) \sim_R \mu_X(B)$, and the displayed endpoint comparison for D_T reads $\mu_S(D_S) \sim_R \mu_X(D_T)$. Addition congruence in the shared $\mathsf{Real}^\uparrow$ additive $\mathsf{AbGroup}^\uparrow$ row transports these two endpoint comparisons through $+$, so

$$\mu_S(B_S) + \mu_S(D_S) \sim_R \mu_X(B) + \mu_X(D_T).$$

A final transitivity step gives

$$\mu_X(A) \sim_R \mu_X(B) + \mu_X(D_T).$$

□

Theorem 186.11 (Distribution pushforward monotone events)**.** *Let $X : S \to T$ be a carried $\mathsf{RandomVar}^\uparrow$ map as in section 185.1, from a $\mathsf{ProbSpace}^\uparrow$ source as in section 184.1 to a target $\mathsf{Measure}^\uparrow$ carrier as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the $\mathsf{Distribution}^\uparrow$ pushforward measure as in section 186.1, $x \sim_R y$ for the $\mathsf{Real}^\uparrow$ classifier of Definition 38.9, and $\leq_R$ for the order relation exposed by the target $\mathsf{RealAlgOrder}$ row. Let A and B be target events in one carried target fiber with $A, B \in \mathcal{A}_T$, and suppose the public target readback displays $B \subseteq A$. Define*

$$A_S := X^{-1}(A), \qquad B_S := X^{-1}(B).$$

Suppose:

- *the target complement/relative-difference row of Definition 92.16, the $\mathsf{RandomVar}^\uparrow$ measurable-preimage row, and the source complement/relative-difference row satisfy the hypotheses of Definition 185.3 for the ambient target event A and subevent B;*
- *the $\mathsf{Distribution}^\uparrow$ pushforward row of section 186.2 gives, at every target measurable event $E \in \mathcal{A}_T$,*
$$\mu_X(E) \sim_R \mu_S(X^{-1}(E));$$
- *the source $\mathsf{Measure}^\uparrow$ certificate satisfies the hypotheses of Definition 92.99 for the displayed source inclusion $B_S \subseteq A_S$ and the measurable relative-difference ledger supplied by the source row;*
- *the target $\mathsf{RealAlgOrder}$ order row on $\mathsf{Real}^\uparrow$ is stable under $\sim_R$ on both endpoints.*

Then

$$\mu_X(B) \leq_R \mu_X(A).$$

Proof. Let $D_T := A \backslash B$ be the target relative-difference event supplied by the target complement/relative-difference row, and let $D_X := X^{-1}(D_T)$. Apply Definition 185.3 to the displayed target inclusion $B \subseteq A$. It gives

$$A_S, B_S, D_X \in \mathcal{A}_S$$

and reads the target inclusion as the source inclusion

$$B_S \subseteq A_S.$$

It also supplies the source relative-difference event $D_S := A_S \setminus B_S$, with $D_S \in \mathcal{A}_S$, disjointness of B_S and D_S, and a displayed binary-union readback classifying $B_S \cup D_S$ as A_S.

These are exactly the inclusion-decomposition inputs for Definition 92.99 in the source $\mathsf{Measure}^\uparrow$ carrier, using B_S as the included event, A_S as the ambient event, and D_S as the displayed difference witness. Hence the source measure row gives

$$\mu_S(B_S) \leq_R \mu_S(A_S).$$

Apply the pushforward row to the target measurable events B and A. This gives

$$\mu_X(B) \sim_R \mu_S(B_S), \qquad \mu_X(A) \sim_R \mu_S(A_S).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier, the first comparison also reads $\mu_S(B_S) \sim_R \mu_X(B)$. Transport the left endpoint of $\mu_S(B_S) \leq_R \mu_S(A_S)$ across this classifier comparison, and transport the right endpoint across $\mu_S(A_S) \sim_R \mu_X(A)$, using the endpoint stability of the target $\mathsf{RealAlgOrder}$ order row. The transported comparison is

$$\mu_X(B) \leq_R \mu_X(A).$$

□

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward monotone events`

Theorem 186.12 (Distribution pushforward probability bounds)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section 185.1, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section 184.1 to a* $\mathsf{Measure}^{\uparrow}$ *target as in section 92.1. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, Ω_S and Ω_T for the source and target total events, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section 186.1, 1_R for the probability unit, $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9,* NonNeg_R *for the nonnegative predicate exposed by the shared* $\mathsf{RealAlgOrder}$ *row, and $\leq_R$ for its order relation. Let B be a target event with $B \in \mathcal{A}_T$, and suppose the public target readback displays $B \subseteq \Omega_T$. Define*

$$B_S := X^{-1}(B).$$

Suppose:

- *the hypotheses of Definition 186.8 hold for the target event B, so the source preimage B_S is measurable and the pushforward row compares $\mu_X(B)$ with $\mu_S(B_S)$;*
- *the hypotheses of Definition 186.11 hold for the displayed inclusion $B \subseteq \Omega_T$, with $A := \Omega_T$ as ambient target event;*
- *$\Omega_S \in \mathcal{A}_S$, $\Omega_T \in \mathcal{A}_T$, the* $\mathsf{RandomVar}^{\uparrow}$ *total-preimage row identifies $X^{-1}(\Omega_T) = \Omega_S$, and the* $\mathsf{ProbSpace}^{\uparrow}$ *total-mass row gives $\mu_S(\Omega_S) \sim_R 1_R$, so the hypotheses of Definition 186.5 hold at Ω_T;*
- *the target* $\mathsf{RealAlgOrder}$ *order row on* $\mathsf{Real}^{\uparrow}$ *is stable under $\sim_R$ on the right endpoint.*

Then both probability bounds hold:

$$\mathsf{NonNeg}_R(\mu_X(B))$$

and

$$\mu_X(B) \leq_R 1_R.$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward probability bounds`

Proof. Apply Definition 186.8 to the target measurable event B. Its source-preimage, pushforward-comparison, source nonnegative-measure, and $\mathsf{RealAlgOrder}$ stability hypotheses are exactly the first group of displayed assumptions, so it gives the lower endpoint

$$\mathsf{NonNeg}_R(\mu_X(B)).$$

Next apply Definition 186.11 to the inclusion $B \subseteq \Omega_T$. With the ambient event specialized to $A := \Omega_T$, the monotonicity row gives

$$\mu_X(B) \leq_R \mu_X(\Omega_T).$$

Apply Definition 186.5 to the same carried map and the target total event Ω_T. The total-preimage and source total-mass assumptions classify the right endpoint by the probability unit:

$$\mu_X(\Omega_T) \sim_R 1_R.$$

Finally transport the right endpoint of $\mu_X(B) \leq_R \mu_X(\Omega_T)$ across this $\mathsf{Real}^{\uparrow}$ classifier comparison using the right-endpoint stability field of the target $\mathsf{RealAlgOrder}$ row. The transported comparison is

$$\mu_X(B) \leq_R 1_R.$$

□

Theorem 186.13 (Distribution complement mass decomposition)**.** *Let $X : S \to T$ be a carried $\mathsf{RandomVar}^{\uparrow}$ map as in section 185.1, from a $\mathsf{ProbSpace}^{\uparrow}$ source as in section 184.1 to a $\mathsf{Measure}^{\uparrow}$ target as in section 92.1. Write Ω_S and Ω_T for the source and target total events, $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the $\mathsf{Distribution}^{\uparrow}$ pushforward measure as in section 186.1, 1_R for the probability unit, and $x \sim_R y$ for the $\mathsf{Real}^{\uparrow}$ classifier of Definition 38.9. Let B be a target event with $B \in \mathcal{A}_T$, and suppose the public target readback displays $B \subseteq \Omega_T$. Let*

$$B_T^{\complement} := \Omega_T \setminus B$$

be the target complement supplied by the target complement/relative-difference row, with public membership readback

$$t \in B_T^{\complement} \iff t \in \Omega_T \text{ and } t \notin B.$$

Suppose:

- *$\Omega_S \in \mathcal{A}_S$, $\Omega_T \in \mathcal{A}_T$, the $\mathsf{RandomVar}^{\uparrow}$ total-preimage row identifies $X^{-1}(\Omega_T) = \Omega_S$, the $\mathsf{Distribution}^{\uparrow}$ pushforward row is available at Ω_T, and the $\mathsf{ProbSpace}^{\uparrow}$ total-mass row gives $\mu_S(\Omega_S) \sim_R 1_R$, so the hypotheses of Definition 186.5 hold at Ω_T;*
- *the hypotheses of Definition 186.10 hold with ambient target event $A := \Omega_T$, subevent B, and displayed target difference $D_T := B_T^{\complement}$;*
- *the $\mathsf{Real}^{\uparrow}$ classifier $\sim_R$ supplies symmetry and transitivity.*

Then

$$\mu_X(\Omega_T) \sim_R \mu_X(B) + \mu_X(B_T^{\complement})$$

and

$$1_R \sim_R \mu_X(B) + \mu_X(B_T^{\complement}).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward complement mass decomposition`

Proof. Apply Definition 186.10 with ambient event $A := \Omega_T$, subevent B, and target difference $D_T := B_T^{\complement}$. Its displayed difference row identifies the relative difference with the target complement inside the total target event, and its endpoint row gives

$$\mu_X(\Omega_T) \sim_R \mu_X(B) + \mu_X(B_T^{\complement}).$$

Apply Definition 186.5 at the target total event Ω_T. The total-preimage and source total-mass rows give

$$\mu_X(\Omega_T) \sim_R 1_R.$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier, the total-mass comparison also reads $1_R \sim_R \mu_X(\Omega_T)$. Transitivity with the complement decomposition just obtained yields

$$1_R \sim_R \mu_X(B) + \mu_X(B_T^{\complement}).$$

□

Theorem 186.14 (Distribution pushforward countable disjoint-union sigma-additivity)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map from a* $\mathsf{ProbSpace}^{\uparrow}$ *source to a* $\mathsf{Measure}^{\uparrow}$ *target. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure, $E \simeq_S F$ for the source event classifier, and $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier. Let $(B_n)_{n\in\mathbb{N}}$ be target events with $B_n \in \mathcal{A}_T$ for every n, and let $U_T \in \mathcal{A}_T$ be a displayed target countable union:*

$$t \in U_T \iff \exists n \in \mathbb{N},\, t \in B_n.$$

Assume that the target row is pairwise disjoint: if $i \neq j$, no carried target point has simultaneous membership in B_i and B_j. For each n, put $A_n := X^{-1}(B_n)$ and put $A_U := X^{-1}(U_T)$. Suppose the following rows are available.

1. *The* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage row sends every target measurable event in this display to a source measurable event and gives the public readback*

$$s \in X^{-1}(E) \iff X(s) \in E.$$

2. *The source countable-union row supplies the source measurable event*

$$U_S := \bigvee\nolimits^{S}_{n\in\mathbb{N}} A_n \in \mathcal{A}_S,$$

and the countable preimage-union bridge supplies the source event classifier comparison

$$A_U \simeq_S U_S.$$

3. *The source* $\mathsf{Measure}^{\uparrow}$ *surface is sigma-additive on displayed pairwise-disjoint source-measurable countable unions:*

$$\mu_S(U_S) \sim_R \sum_{n\in\mathbb{N}} \mu_S(A_n).$$

4. *The* $\mathsf{Distribution}^{\uparrow}$ *pushforward row reads every target measurable event $E \in \mathcal{A}_T$ by*

$$\mu_X(E) \sim_R \mu_S(X^{-1}(E)).$$

5. *The source measure is readable through the source event classifier:*

$$E \simeq_S F \Longrightarrow \mu_S(E) \sim_R \mu_S(F).$$

6. *The countable* $\mathsf{Real}^{\uparrow}$ *sum endpoint is stable under pointwise* $\mathsf{Real}^{\uparrow}$ *classifier comparison:*

$$(\forall n \in \mathbb{N},\, r_n \sim_R q_n) \Longrightarrow \sum_{n\in\mathbb{N}} r_n \sim_R \sum_{n\in\mathbb{N}} q_n.$$

Then the pushed-forward target measure classifies the displayed target countable disjoint union by the countable sum of the pushed-forward component measures:

$$\mu_X(U_T) \sim_R \sum_{n\in\mathbb{N}} \mu_X(B_n).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward countable disjoint sigma additivity`

Proof. Invert the measurable-preimage row at U_T and at each B_n. This gives $A_U \in \mathcal{A}_S$, gives $A_n \in \mathcal{A}_S$ for every n, and gives the readbacks

$$s \in A_U \iff X(s) \in U_T$$

and

$$s \in A_n \iff X(s) \in B_n.$$

The source countable-union row supplies $U_S \in \mathcal{A}_S$, and the countable preimage-union bridge gives

$$A_U \simeq_S U_S.$$

The source sequence $(A_n)_{n\in\mathbb{N}}$ is pairwise disjoint. Indeed, let $i \neq j$ and suppose that a carried source point s has witnesses $s \in A_i$ and $s \in A_j$. The preimage readback for B_i gives $X(s) \in B_i$, and the preimage readback for B_j gives $X(s) \in B_j$. This contradicts the displayed target disjointness of B_i and B_j. Hence the hypotheses of the source sigma-additivity row apply to (A_n) and U_S, giving

$$\mu_S(U_S) \sim_R \sum_{n\in\mathbb{N}} \mu_S(A_n).$$

Apply the pushforward row to the target union U_T. Since $U_T \in \mathcal{A}_T$,

$$\mu_X(U_T) \sim_R \mu_S(A_U).$$

Read the source event comparison $A_U \simeq_S U_S$ through the source measure classifier-transport row. This gives

$$\mu_S(A_U) \sim_R \mu_S(U_S).$$

By transitivity of the $\mathsf{Real}^{\uparrow}$ classifier,

$$\mu_X(U_T) \sim_R \mu_S(U_S).$$

Composing this comparison with the source sigma-additivity comparison yields

$$\mu_X(U_T) \sim_R \sum_{n\in\mathbb{N}} \mu_S(A_n).$$

For each $n \in \mathbb{N}$, the pushforward row applied to the target measurable event B_n gives

$$\mu_X(B_n) \sim_R \mu_S(A_n).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier,

$$\mu_S(A_n) \sim_R \mu_X(B_n).$$

The countable-sum stability row applied to the pointwise comparisons gives

$$\sum_{n\in\mathbb{N}} \mu_S(A_n) \sim_R \sum_{n\in\mathbb{N}} \mu_X(B_n).$$

A final transitivity step for the $\mathsf{Real}^{\uparrow}$ classifier gives

$$\mu_X(U_T) \sim_R \sum_{n\in\mathbb{N}} \mu_X(B_n).$$

□

Theorem 186.15 (Distribution pushforward continuity from below)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^{\uparrow}$ *map as in section* 185.1*, from a* $\mathsf{ProbSpace}^{\uparrow}$ *source as in section* 184.1 *to a target* $\mathsf{Measure}^{\uparrow}$ *carrier as in section* 92.1*. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, μ_X for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure as in section* 186.1*, and $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition* 38.9*. Let $(B_n)_{n\in\mathbb{N}}$ be target events with $B_n \in \mathcal{A}_T$ for every n, and suppose the target readback supplies both*

$$B_n \subseteq B_{n+1} \quad \textit{for every } n \in \mathbb{N},$$

and

$$B := \bigcup_{n\in\mathbb{N}} B_n \qquad \textit{with} \qquad t \in B \iff \exists n \in \mathbb{N},\ t \in B_n.$$

For each n, let $A_n := X^{-1}(B_n)$ and let $A_B := X^{-1}(B)$. Suppose the hypotheses of Definition 185.26 *and Definition* 92.42 *are satisfied for this data, including source measurability and the source countable-union row*

$$U_S := \bigvee\nolimits^{S}_{n\in\mathbb{N}} A_n \in \mathcal{A}_S \qquad \textit{and} \qquad A_B \sim_{\mathcal{A}_S} U_S.$$

Then

$$\mu_X(B) \sim_R \lim_{n\to\infty} \mu_X(B_n).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward continuity from below prefix readback`

Proof. For each n, apply the measurable-preimage readback to get

$$A_n \in \mathcal{A}_S, \qquad A_B \in \mathcal{A}_S.$$

From the target inclusion row $B_n \subseteq B_{n+1}$ and the preimage readbacks

$$s \in A_n \iff X(s) \in B_n, \qquad s \in A_{n+1} \iff X(s) \in B_{n+1},$$

it follows pointwise that $A_n \subseteq A_{n+1}$. Hence $(A_n)_{n\in\mathbb{N}}$ is an increasing source measurable sequence.

Apply *Definition* 92.42 to (A_n) and its source union U_S. This provides a source disjoint layer chain with prefix readbacks

$$\Sigma_\mu(D_0, \ldots, D_n) \sim_R \mu_S(A_n) \quad \text{for every } n \in \mathbb{N},$$

and the endpoint comparison

$$\mu_S(U_S) \sim_R \sum_{n\in\mathbb{N}} \mu_S(D_n).$$

By the $\mathsf{Distribution}^{\uparrow}$ pushforward row of *section* 186.2,

$$\mu_X(B) \sim_R \mu_S(A_B),$$

and for each n

$$\mu_X(B_n) \sim_R \mu_S(A_n).$$

Composing the first comparison with $A_B \sim_{\mathcal{A}_S} U_S$ through source measure transport gives

$$\mu_X(B) \sim_R \mu_S(U_S).$$

Combining this with the continuity-from-below endpoint row yields

$$\mu_X(B) \sim_R \sum_{n\in\mathbb{N}} \mu_S(D_n).$$

Together with the prefix readbacks, each finite prefix is also transport-equivalent to the pushed-forward target prefix value:

$$\Sigma_\mu(D_0, \ldots, D_n) \sim_R \mu_X(B_n) \quad \text{for every } n \in \mathbb{N}.$$

By the continuity-from-below readback convention for this displayed chain, the endpoint on the right is exactly the monotone limit of the target pushed-forward prefix sequence, hence

$$\mu_X(B) \sim_R \lim_{n\to\infty} \mu_X(B_n).$$

□

Theorem 186.16 (Distribution pushforward probability package)**.** *Let $X : S \to T$ be a carried* $\mathsf{RandomVar}^\uparrow$ *map as in section* 185.1*, from a* $\mathsf{ProbSpace}^\uparrow$ *source as in section* 184.1 *to a target* $\mathsf{Measure}^\uparrow$ *carrier as in section* 92.1*. Write $\mathcal{A}_T$ for the target measurable-event family, μ_X for the* $\mathsf{Distribution}^\uparrow$ *pushforward row on target events, 0_R and 1_R for the distinguished* $\mathsf{Real}^\uparrow$ *endpoints, and $\sim_R$ for the* $\mathsf{Real}^\uparrow$ *classifier.*

Assume that the hypotheses of Definition 186.5*, Definition* 186.6*, Definition* 186.8*, Definition* 186.11*, Definition* 186.7*, Definition* 186.10*, Definition* 186.13*, Definition* 186.14*, and Definition* 186.15 *are all satisfied for this same displayed pushforward surface.*

Then μ_X is a probability-measure package on the target event surface: $\mu_X(T) \sim_R 1_R$; $\mu_X(\varnothing_T) \sim_R 0_R$; every $B \in \mathcal{A}_T$ has nonnegative $\mathsf{Real}^\uparrow$ *endpoint; target inclusion is monotone; finite disjoint binary unions are additive; relative differences decompose additively; complements decompose total mass over an event and its target complement; countable disjoint unions are sigma-additive; and increasing countable unions satisfy continuity from below.*

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward probability package`

Proof. Under the simultaneous hypotheses, the unit row is the row supplied by *Definition* 186.5, and the empty-event row is supplied by *Definition* 186.6. Nonnegativity of every target measurable value is the component row from *Definition* 186.8.

The order-sensitive rows are also part of the assumed component surface. Monotonicity is the row of *Definition* 186.11, finite disjoint binary additivity is the row of *Definition* 186.7, and relative-difference additivity is the row of *Definition* 186.10. The complement-mass row is supplied by *Definition* 186.13.

For countable structure, the assumed surface includes *Definition* 186.14 for disjoint target families and *Definition* 186.15 for increasing target families. These nine rows have the same carried map X, the same target event family $\mathcal{A}_T$, and the same displayed pushforward row μ_X, so collecting the assumed component rows gives the announced probability-measure package on the target event surface. □

Theorem 186.17 (Distribution null-completion random-variable descent)**.** *Let $X, Y : S \to T$ be carried* $\mathsf{RandomVar}^{\uparrow}$ *maps from the same* $\mathsf{ProbSpace}^{\uparrow}$ *source to the same* $\mathsf{Measure}^{\uparrow}$ *target, and let μ_X and μ_Y be their* $\mathsf{Distribution}^{\uparrow}$ *pushforward endpoint assignments. Write $\mathcal{A}_S$ and $\mathcal{A}_T$ for the source and target measurable-event families, μ_S for the source measure, $E \simeq_S F$ for the source event classifier, and $x \sim_R y$ for the* $\mathsf{Real}^{\uparrow}$ *classifier. Suppose that for every target event $B \in \mathcal{A}_T$ the following displayed rows are available:*

1. *the preimage rows $A_X := X^{-1}(B)$ and $A_Y := Y^{-1}(B)$ lie in $\mathcal{A}_S$;*
2. *a source symmetric-difference row supplies $\Delta_B := A_X \triangle A_Y \in \mathcal{A}_S$ with the public readback $s \in \Delta_B$ iff s lies in exactly one of A_X, A_Y;*
3. *the null row classifies the symmetric difference with zero measure, $\mu_S(\Delta_B) \sim_R 0_R$;*
4. *the source measure ledger reads equality up to null symmetric difference as endpoint equality: if $\mu_S(E \triangle F) \sim_R 0_R$, then $\mu_S(E) \sim_R \mu_S(F)$ for measurable source events E, F;*
5. *the* $\mathsf{Distribution}^{\uparrow}$ *pushforward rows read $\mu_X(B) \sim_R \mu_S(A_X)$ and $\mu_Y(B) \sim_R \mu_S(A_Y)$.*

Then for every $B \in \mathcal{A}_T$,

$$\mu_X(B) \sim_R \mu_Y(B).$$

Lean4 ✓ `BEDC.Derived.DistributionUp.DistributionPushforward null completion random variable descent`

Proof. Fix a target measurable event B. The displayed preimage rows give source measurable events A_X and A_Y. The symmetric-difference row supplies the source event $\Delta_B = A_X \triangle A_Y$, and the null hypothesis gives

$$\mu_S(\Delta_B) \sim_R 0_R.$$

Apply the source measure ledger row for equality up to null symmetric difference to A_X and A_Y. Its hypothesis is exactly the displayed null row for Δ_B, so it gives

$$\mu_S(A_X) \sim_R \mu_S(A_Y).$$

Now read both pushforward rows through the distribution carrier. The row for X gives $\mu_X(B) \sim_R \mu_S(A_X)$, and the row for Y gives $\mu_Y(B) \sim_R \mu_S(A_Y)$. Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier composes the X pushforward readback with the null-difference endpoint comparison and then with symmetry of the Y pushforward readback. Hence

$$\mu_X(B) \sim_R \mu_Y(B).$$

Since B was arbitrary, the comparison holds for every target measurable event. This is a distribution-classifier descent statement only: it consumes the displayed source null ledger and pushforward rows, and it introduces no host quotient of events, no analytic completion object, and no external probability-space equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$.

Constructive story: A distribution is built from a carried random variable $X : S \to T$ by asking the $\mathsf{RandomVar}^{\uparrow}$ row to turn every target measurable event B into a source measurable event $X^{-1}(B)$, then reading the source $\mathsf{ProbSpace}^{\uparrow}$ measure at that preimage. The carrier accepts exactly the target measurable family together with the endpoint assignment $B \mapsto \mu_S(X^{-1}(B))$, while the classifier compares two such rows only after the target-event classifier has matched the events and the $\mathsf{Real}^{\uparrow}$ classifier has matched the resulting measured endpoints. Thus the source covers carried random-variable maps from normalized probability sources into measurable targets, the pattern is the pushforward law plus total mass, empty-event, nonnegativity, additivity, monotonicity, sigma-additivity, and continuity-from-below rows, and the ledger hides

the preimage event that explains each public target endpoint. Because target-event transport, random-variable preimage transport, source-measure transport, and the Real$^{\uparrow}$ classifier all preserve their carried rows, the certificate has an inhabited pushforward carrier, reflexive, symmetric, and transitive row comparison, and closure under classifier replacement. A row h lies in the signature gap of a Distribution$^{\uparrow}$ token p when the source-domain accepts the carried map and target event surface, the signature relation pairs the displayed pushforward assignment with an answer row and evidence ledger, and token introduction names that answer as p; concretely, every public value of h is the source probability of the certified preimage event encoded by p. The standard bridge reads this as the ordinary pushforward probability measure $X_*\mu_S$, but only after passing through the target-event classifier and the Real$^{\uparrow}$ endpoint classifier. The chapter registers this pushforward construction under the name Distribution$^{\uparrow}$; a law of a random variable is a preimage ledger over ProbSpace$^{\uparrow}$, with no separate host measure, convergence-of-laws package, or quotient of events introduced.

Theory closure: TheoryGate(NameCert$_{\text{Distribution}^{\uparrow}}$, maturePackageClosed)

Scope closed: The displayed pushforward carrier and classifier are closed for carried RandomVar$^{\uparrow}$ maps from ProbSpace$^{\uparrow}$ sources to Measure$^{\uparrow}$ target events, with endpoint comparisons read through the Real$^{\uparrow}$ classifier.

Formal verification: FormalStatus(NameCert$_{\text{Distribution}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DistributionUp.DistributionPushforward\_probability\_package`

Bridge status: paperBridge

Not claimed: The chapter does not close conditional expectation, independence, convergence of laws, or machine-checked bridge verification.

Upgrade path: No higher paper-side closure grade remains; machine-checked bridge verification and standard-model comparison stay outside this chapter.

Chapter 187

A Concrete Naming Certificate for $\mathsf{Independence}^{\uparrow}$

The independence interface $\mathsf{Independence}^{\uparrow}$ packages the property that the joint distribution of a finite family of random variables factors as the product of marginals. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186).

187.1 Carrier definition

Definition 187.1 (Finite-family carrier for IndependenceUp)**.** *A finite-family carrier for* $\mathsf{Independence}^{\uparrow}$ *consists of a carried* $\mathsf{ProbSpace}^{\uparrow}$ *source* S*, a finite* $\mathsf{FinSet}^{\uparrow}$ *index carrier* I*, and for every* $i \in I$ *a carried* $\mathsf{RandomVar}^{\uparrow}$ *map*

$$X_i : S \to T_i.$$

For each i*, the carrier includes the displayed* $\mathsf{Distribution}^{\uparrow}$ *marginal endpoint* μ_i *obtained as the pushforward of* X_i*, together with a displayed joint* $\mathsf{Distribution}^{\uparrow}$ *endpoint* μ_X^J *for the tuple map into the finite product target* $\prod_{i \in I} T_i$*.*

The readback ledger assigns to every finite family of measurable target events $B = (B_i)_{i \in I}$ *a product-cylinder event* $\mathrm{Cyl}_X(B)$ *in the joint target and records the source-event readback*

$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \iff \forall i \in I.\ X_i(s) \in B_i.$$

Thus the carrier exposes exactly the source, finite index spine, random-variable spine, marginal distribution spine, joint distribution endpoint, and product-cylinder readback needed for the finite factorisation row.

Definition 187.2 (Finite factorisation classifier for IndependenceUp)**.** *Let* R_X *and* R_Y *be two finite-family rows of Definition 187.1. An* $\mathsf{Independence}^{\uparrow}$ *factorisation classifier from* R_X *to* R_Y *consists of the following compatible comparisons:*

1. *a* $\mathsf{ProbSpace}^{\uparrow}$ *classifier identifying the carried sources;*
2. *a* $\mathsf{FinSet}^{\uparrow}$ *classifier identifying the finite index carriers and transporting the finite-position spine;*
3. *for every matched index, a* $\mathsf{RandomVar}^{\uparrow}$ *classifier between the corresponding carried random-variable maps;*
4. *for every matched index and measurable target event, a* $\mathsf{Distribution}^{\uparrow}$ *classifier between the corresponding marginal endpoints;*

5. *compatibility of the two product-cylinder readbacks under the source, index, and random-variable classifiers;*
6. $\mathsf{Real}^{\uparrow}$ *classifier agreement between the two marginal product folds and between the two joint-cylinder endpoints for every transported measurable event family.*

The classifier holds when these comparisons compose to the displayed endpoint condition

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)) \Longleftrightarrow \mu_Y^J(\mathrm{Cyl}_Y(B')) \sim_R \Pi_R(\mathsf{MargSp}_Y(B'))$$

for each event family B and its transported family B'.

187.2 The certificate

Definition 187.3 (Finite factorisation row for IndependenceUp)**.** *Let S be a carried* $\mathsf{ProbSpace}^{\uparrow}$ *source, let I be a finite index carrier supplied by a* $\mathsf{FinSet}^{\uparrow}$ *certificate, and let*

$$X_i : S \to T_i \qquad (i \in I)$$

be a finite family of carried $\mathsf{RandomVar}^{\uparrow}$ *maps as in section 185.1. For each i, write μ_i for the* $\mathsf{Distribution}^{\uparrow}$ *pushforward measure of X_i as in section 186.1, and write μ_X^J for the displayed joint* $\mathsf{Distribution}^{\uparrow}$ *endpoint of the tuple family. A finite factorisation row for* $\mathsf{Independence}^{\uparrow}$ *consists of the following data:*

1. *for every finite family of measurable target events $B = (B_i)_{i\in I}$, a product-cylinder event $\mathrm{Cyl}_X(B)$ in the joint target, read by*

$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \Longleftrightarrow \forall i \in I.\ X_i(s) \in B_i;$$

2. *a finite* $\mathsf{Real}^{\uparrow}$ *product fold Π_R on the marginal-value spine*

$$\mathsf{MargSp}_X(B)(i) := \mu_i(B_i),$$

   using the multiplication and classifier rows of the $\mathsf{Real}^{\uparrow}$ *commutative ring surface of Definition 43.26 and Definition 44.33;*

3. *the joint-distribution factorisation row*

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B))$$

   for every such measurable event family B, where $\sim_R$ is the $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9.*

This is the concrete paper-side factorisation surface carried by $\mathsf{Independence}^{\uparrow}$ *in section 187.1 and section 187.2.*

Lean4 (definition) `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow`

Theorem 187.4 (Independence obligation stability under certified transport)**.** *Let R_X and R_Y be finite-family carriers for* $\mathsf{Independence}^{\uparrow}$ *as in Definition 187.1. Suppose they are related by the classifier of Definition 187.2, using the* $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *classifier rows named there. If R_X carries the finite factorisation row of Definition 187.3, then R_Y carries the corresponding finite factorisation row for every transported measurable event family.*

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow stability transport`

Proof. Fix a measurable event family B' for R_Y and choose its transported source family B along the $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{RandomVar}^{\uparrow}$ classifier data. The product-cylinder compatibility in Definition 187.2 identifies the two source-event readbacks, so the $\mathsf{Distribution}^{\uparrow}$ rows compare the two joint-cylinder endpoints.

The marginal clauses of the same classifier compare each marginal endpoint in the transported spine. The finite-position comparison and, when the presentation order changes, the $\mathsf{Permutation}^{\uparrow}$ row reduce the product comparison to Definition 187.11.

Apply the factorisation row for R_X to B. The joint-endpoint comparison, the transported marginal-product comparison, and transitivity of the $\mathsf{Real}^{\uparrow}$ classifier give the factorisation row for R_Y at B'. Since B' was arbitrary, stability holds for the whole finite family. □

Theorem 187.5 (Independence obligation ledger exactness)**.** *For a finite-family carrier R_X of Definition 187.1, the* $\mathsf{Independence}^{\uparrow}$ *ledger exposes exactly four rows: the product-cylinder readback, the marginal distribution spine, the finite* $\mathsf{Real}^{\uparrow}$ *product fold, and the joint distribution endpoint. The ledger is exact precisely when, for every finite measurable event family B, the displayed factorisation row*

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B))$$

holds with the cylinder, spine, fold, and endpoint supplied by those four rows.

Proof. The carrier definition lists the only data read by the factorisation judgment: source membership in the cylinder, the marginal endpoint attached to each index, the finite product of those endpoints, and the joint endpoint of the tuple map. No additional event, source witness, or hidden distribution endpoint is read by Definition 187.3.

If the ledger is exact, evaluating it at B returns those four entries, and the final entry is classified with the product fold by the recorded factorisation row. This gives the displayed equation for every measurable event family.

Conversely, assume the displayed equation holds for every B using the four ledger rows. Then every value consumed by the independence judgment is one of the exposed rows, and the judgment is true exactly when the displayed equation is true. Hence the ledger is exact for the finite-family factorisation surface. □

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow ledger exactness`

Theorem 187.6 (Independence obligation certificate packet)**.** *The data consisting of the carrier of Definition 187.1, the classifier of Definition 187.2, the stability theorem of Definition 187.4, and the ledger exactness theorem of Definition 187.5 form the finite-family* $\mathsf{Independence}^{\uparrow}$ *obligation packet. This packet fixes the paper-side obligations for closing* $\mathsf{NameCert}_{\mathsf{Independence}^{\uparrow}}$ *on finite families of carried random variables over a carried* $\mathsf{ProbSpace}^{\uparrow}$ *source.*

Proof. The carrier supplies the source, finite index, random-variable spine, marginal endpoints, joint endpoint, and cylinder readback. The classifier states exactly when two such rows have the same finite-family independence content after certified transport.

The stability theorem records that the factorisation judgment is preserved by the named classifiers. The ledger exactness theorem records that no field outside the cylinder readback, marginal spine, product fold, and joint endpoint is used by the judgment.

Together these rows are the required carrier, classifier, stability, ledger, and exactness faces of the finite-family naming certificate. Therefore they assemble into the $\mathsf{Independence}^{\uparrow}$ obligation packet for the stated finite scope. □

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow obligation certificate packet`

Theorem 187.7 (Independence scoped primitive boundary)**.** *The finite-family* $\mathsf{Independence}^{\uparrow}$ *certificate surface is scoped by exactly the displayed* $\mathsf{NameCert}$ *rows for* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the BEDC kernel carriers and classifiers* $BHist$, $hsame$, *and* $Cont$, *the product-cylinder readback, finite product fold, joint endpoint, and marginal spine of Definition 187.3. It consumes no conditional-expectation row, martingale row, sigma-completion row, host equality row, host function equality row, or standard probability-space bridge row.*

Proof. Apply Definition 187.6. The packet assembles the finite-family carrier, classifier, stability, ledger, and exactness faces from the displayed source, index, random-variable, distribution, permutation, and real rows.

The generated transport scope is the one stated in Definition 187.9. Its generators are certified finite reindexing, readable-preimage transport, distribution endpoint transport, and finite-product classifier transport; each is supplied by the displayed $\mathsf{NameCert}$ rows and the kernel classifiers $BHist$, $hsame$, and $Cont$.

The exact ledger boundary is the one stated in Definition 187.10. It reads only the product-cylinder readback, marginal distribution spine, finite $\mathsf{Real}^{\uparrow}$ product fold, and joint distribution endpoint of Definition 187.3.

The nontrivial finite-family stability faces are exactly the checked rows Definition 187.11, Definition 187.12, and Definition 187.13. Those rows cover permutation readback, finite reindexing, and measurable-image transport, so no conditional-expectation, martingale, sigma-completion, host equality, host function equality, or standard bridge row remains in the scoped boundary. □

Theorem 187.8 (Independence carrier obligation)**.** *The carrier for* $\mathsf{Independence}^{\uparrow}$ *is exactly the finite-family package consisting of one carried* $\mathsf{ProbSpace}^{\uparrow}$ *source* S, *a certified* $\mathsf{FinSet}^{\uparrow}$ *index carrier* I, *carried* $\mathsf{RandomVar}^{\uparrow}$ *maps* $X_i : S \to T_i$ *for* $i \in I$, *their* $\mathsf{Distribution}^{\uparrow}$ *marginal endpoints* μ_i, *the displayed joint* $\mathsf{Distribution}^{\uparrow}$ *endpoint* μ_X^J, *and the finite factorisation row of Definition 187.3. No further carrier datum is part of the* $\mathsf{Independence}^{\uparrow}$ *surface.*

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow carrier obligation`

Proof. Unpack the finite factorisation row. Its inputs name one source probability-space carrier, one finite index carrier, and one carried random-variable map at each finite index. The row then records exactly the marginal pushforward endpoint for each map and the joint pushforward endpoint for the tuple family.

The remaining fields of Definition 187.3 are certificate rows over those endpoints: product-cylinder readback, finite $\mathsf{Real}^{\uparrow}$ product fold, and the joint-distribution factorisation comparison. Each field is typed over the same displayed source, index, random-variable, marginal, and joint endpoints. Thus the carrier is closed by these finite data, and there is no separate conditional, completed, or standard-space carrier to add. □

Theorem 187.9 (Independence classifier and stability obligation)**.** *The classifier comparison for* $\mathsf{Independence}^{\uparrow}$ *is generated by certified index reindexing, readable-preimage transport for carried* $\mathsf{RandomVar}^{\uparrow}$ *maps,* $\mathsf{Distribution}^{\uparrow}$ *endpoint transport along the displayed pushforward rows, and finite-product classifier transport over* $\mathsf{Real}^{\uparrow}$. *Under these generators, the finite factorisation row is stable under certified finite reindexing and under measurable-image transport of a binary pair.*

Proof. The classifier generators are the comparison rows already present in the input certificates. A certified reindexing compares index positions by the $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Permutation}^{\uparrow}$ rows. A readable-preimage transport compares target events through the $\mathsf{RandomVar}^{\uparrow}$ readback clauses. The $\mathsf{Distribution}^{\uparrow}$ pushforward row transports the resulting event comparison to marginal and joint endpoints, and the $\mathsf{Real}^{\uparrow}$ classifier transports the finite product fold.

For the finite-family face, apply Definition 187.12. It uses the certified inverse position map, product-cylinder readback, the original factorisation row, and Definition 187.11 to rebuild the same factorisation comparison after reindexing.

For the binary measurable-image face, apply Definition 187.13. It reads preimages through the displayed maps, transports the marginal and joint distribution endpoints, and then composes the original binary factorisation comparison with $\mathsf{Real}^{\uparrow}$ classifier transitivity. These two stability faces exhibit the generated classifier comparison for the independence certificate. □

Theorem 187.10 (Independence ledger and exactness obligation)**.** *The exact ledger exported by* $\mathsf{Independence}^{\uparrow}$ *consists only of the source probability-space row, finite index evidence, random-variable preimage rows, distribution pushforward rows, product-cylinder readback, and finite* $\mathsf{Real}^{\uparrow}$ *product-fold provenance. It exports no conditional expectation, martingale, sigma-algebra completion, or standard probability-space bridge.*

Proof. Read the ledger entries from Definition 187.3. The source row supplies the carried probability space, the finite index evidence supplies the allowed family positions, and the random-variable rows supply readable preimages for the events used in each product cylinder.

The distribution rows supply only the marginal and joint pushforward endpoints attached to those same readable events. The product-cylinder readback records the equivalence between membership in the tuple cylinder and membership in every displayed component event. The $\mathsf{Real}^{\uparrow}$ product-fold provenance records how marginal values are multiplied and compared by the real classifier.

Exactness follows because the factorisation equation mentions only these exported rows:

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Neither the statement nor the construction calls for conditional expectation, martingale filtrations, sigma-algebra completion, or a standard probability-space bridge. The ledger is therefore exact for the finite-family independence surface and silent about those additional probability-theoretic structures. □

Lemma 187.11 (Finite RealUp product fold is invariant under certified reindexing)**.** *Let xs and ys be finite spines of carried* $\mathsf{Real}^{\uparrow}$ *endpoints, and let*

$$\rho : \mathsf{Pos}(xs) \cong \mathsf{Pos}(ys)$$

be a certified finite-position bijection in the sense of Definition 112.2. If each matched pair satisfies

$$\mathsf{entry}_{xs}(i) \sim_R \mathsf{entry}_{ys}(\rho(i)),$$

then the finite product fold Π_R satisfies

$$\Pi_R(xs) \sim_R \Pi_R(ys).$$

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteProduct reindexing readback`

Proof. Induct on the target spine ys. If ys is empty, the certified inverse position map forces $\mathsf{Pos}(xs)$ to be empty as well. Both product folds are the multiplicative unit endpoint, so the $\mathsf{Real}^{\uparrow}$ classifier reflexivity row gives the claim.

In the cons case, write the head of ys as b and its tail as t. Let $i_0 := \rho^{-1}(\text{head})$ be the source position matched with the target head. Removing i_0 from the source position spine and removing the head from ys leaves a certified finite-position bijection between the residual source spine and t; this is the same inverse-map data used in Definition 112.24, restricted away from the matched pair.

By the induction hypothesis, the product of the residual source spine is classified with $\Pi_R(t)$. The matched-pair hypothesis gives

$$\mathsf{entry}_{xs}(i_0) \sim_R b.$$

Multiplication congruence from Definition 43.26 transports these two comparisons to the product of the selected head and the residual product. It remains only to compare the original source fold with the fold in which the selected occurrence i_0 is placed first. A finite sequence of adjacent swaps moves i_0 to the head of the source spine; each adjacent swap is justified by multiplicative commutativity from Definition 44.33, multiplicative associativity from Definition 43.26, and classifier transitivity. Hence the original source product is classified with $\mathsf{entry}_{xs}(i_0) \cdot_R \Pi_R(\text{residual})$, while the target product is classified with $b \cdot_R \Pi_R(t)$. The preceding congruence and transitivity give

$$\Pi_R(xs) \sim_R \Pi_R(ys).$$

□

Theorem 187.12 (Independence finite reindexing invariance)**.** *Let I be a finite $\mathsf{FinSet}^{\uparrow}$ index carrier, let $\pi : I \cong I$ be a $\mathsf{Permutation}^{\uparrow}$ certified bijection as in section 116.1, and let $(X_i)_{i\in I}$ be a finite family of carried $\mathsf{RandomVar}^{\uparrow}$ maps over one carried $\mathsf{ProbSpace}^{\uparrow}$ source. Suppose the family carries the finite factorisation row of Definition 187.3. Define the reindexed family and marginals by*

$$X_i^{\pi} := X_{\pi(i)}, \qquad \mu_i^{\pi} := \mu_{\pi(i)}.$$

For every reindexed measurable event family $B = (B_i)_{i\in I}$, define the pulled-back original event family

$$C_j := B_{\pi^{-1}(j)}.$$

Let the reindexed joint $\mathsf{Distribution}^{\uparrow}$ endpoint be read through the same product-cylinder ledger, so that the cylinder for X^{π} at B is classified with the cylinder for X at C. Then the reindexed family also carries the finite $\mathsf{Independence}^{\uparrow}$ factorisation row:

$$\mu^J_{X^{\pi}}(\mathrm{Cyl}_{X^{\pi}}(B)) \sim_R \Pi_R(\mathsf{MargSp}_{X^{\pi}}(B)).$$

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteReindexing invariance`

Proof. Fix a measurable reindexed event family $B = (B_i)_{i\in I}$ and put $C_j := B_{\pi^{-1}(j)}$. Since π is a certified bijection, every target index j has the unique source index $\pi^{-1}(j)$ and every source index i is sent to $\pi(i)$. For each carried sample point s, the cylinder readbacks give

$$s \in X_{\pi}^{-1}(\mathrm{Cyl}_{X^{\pi}}(B)) \iff \forall i \in I.\ X_{\pi(i)}(s) \in B_i \iff \forall j \in I.\ X_j(s) \in B_{\pi^{-1}(j)} \iff s \in X^{-1}(\mathrm{Cyl}_X(C)).$$

Thus the reindexed cylinder and the original cylinder have the same displayed source-event readback. The $\mathsf{Distribution}^{\uparrow}$ pushforward row of section 186.2, read through the event transport of Definition 26.3, gives

$$\mu^J_{X^{\pi}}(\mathrm{Cyl}_{X^{\pi}}(B)) \sim_R \mu^J_X(\mathrm{Cyl}_X(C)).$$

Apply the original $\mathsf{Independence}^{\uparrow}$ factorisation row from Definition 187.3 to the event family C. This yields

$$\mu_X^J(\mathrm{Cyl}_X(C)) \sim_R \Pi_R(\mathsf{MargSp}_X(C)).$$

It remains to compare the two marginal products. The original marginal spine at a position j has entry

$$\mathsf{MargSp}_X(C)(j) = \mu_j(C_j) = \mu_j(B_{\pi^{-1}(j)}),$$

while the reindexed marginal spine at i has entry

$$\mathsf{MargSp}_{X^\pi}(B)(i) = \mu_i^\pi(B_i) = \mu_{\pi(i)}(B_i).$$

Using the inverse certified bijection $j \mapsto \pi^{-1}(j)$ from Definition 112.24, the entry at j in the original spine is matched with the entry at $\pi^{-1}(j)$ in the reindexed spine, and these two entries are the same marginal endpoint. Classifier reflexivity gives the matched-entry comparison. Therefore Definition 187.11 gives

$$\Pi_R(\mathsf{MargSp}_X(C)) \sim_R \Pi_R(\mathsf{MargSp}_{X^\pi}(B)).$$

Transitivity of the $\mathsf{Real}^{\uparrow}$ classifier from Definition 38.9 composes the reindexed joint comparison, the original factorisation row, and the finite-product reindexing comparison. Hence

$$\mu_{X^\pi}^J(\mathrm{Cyl}_{X^\pi}(B)) \sim_R \Pi_R(\mathsf{MargSp}_{X^\pi}(B)).$$

Since B was arbitrary, the reindexed family carries the stated finite $\mathsf{Independence}^{\uparrow}$ factorisation row. □

Theorem 187.13 (Independence$^{\uparrow}$ measurable-image bridge for two variables)**.** *Let S be a carried* $\mathsf{ProbSpace}^{\uparrow}$ *source. Let*

$$X : S \to T, Y : S \to U$$

be carried $\mathsf{RandomVar}^{\uparrow}$ *maps as in section 185.1, and suppose the pair (X, Y) carries the binary finite factorisation row of* $\mathsf{Independence}^{\uparrow}$ *from Definition 187.3. Let*

$$f : T \to T', \qquad g : U \to U'$$

be carried $\mathsf{RandomVar}^{\uparrow}$ *maps satisfying the readable preimage row of section 185.2. Assume for every target-measurable $C_0 \in \mathcal{A}_{T'}$ and $C_1 \in \mathcal{A}_{U'}$ that the pullbacks*

$$A_0 := f^{-1}(C_0) \in \mathcal{A}_T, \qquad A_1 := g^{-1}(C_1) \in \mathcal{A}_U$$

are readable and satisfy

$$t \in A_0 \iff f(t) \in C_0, \qquad u \in A_1 \iff g(\nu) \in C_1.$$

Define the compositions

$$H := f \circ X, \qquad K := g \circ Y$$

and write $\mu_H, \mu_K, \mu_{H,K}^J$ for their pushforwards.

Fix measurable $C_0 \in \mathcal{A}_{T'}$ and $C_1 \in \mathcal{A}_{U'}$. For every carried source point s the readable preimage clauses give

$$s \in H^{-1}(C_0) \iff X(s) \in A_0, \qquad s \in K^{-1}(C_1) \iff Y(s) \in A_1,$$

and hence

$$s \in \mathrm{Cyl}_{(H,K)}(C_0, C_1) \iff \mathrm{Cyl}_{(X,Y)}(A_0, A_1).$$

Applying the distribution pushforward row of section 186.2 to the two displayed cylinder events gives

$$\mu^J_{H,K}(\mathrm{Cyl}_{(H,K)}(C_0, C_1)) \sim_R \mu^J_{(X,Y)}(\mathrm{Cyl}_{(X,Y)}(A_0, A_1)),$$
$$\mu_H(C_0) \sim_R \mu_X(A_0), \qquad \mu_K(C_1) \sim_R \mu_Y(A_1).$$

The factorisation row of (X, Y) from Definition 187.3 yields

$$\mu^J_{(X,Y)}(\mathrm{Cyl}_{(X,Y)}(A_0, A_1)) \sim_R \mu_X(A_0) \cdot_R \mu_Y(A_1).$$

Transitivity of $\sim_R$ then gives

$$\mu^J_{H,K}(\mathrm{Cyl}_{(H,K)}(C_0, C_1)) \sim_R \mu_H(C_0) \cdot_R \mu_K(C_1).$$

Since C_0, C_1 were arbitrary measurable events, this is exactly the binary finite factorisation equation for the pair (H, K). Therefore $f \circ X$ is independent of $g \circ Y$ under $\mathsf{Independence}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceBinaryFactorization measurable image bridge`

Theorem 187.14 (Independence public NameCert export)**.** *The finite-family carrier, factorisation classifier, stability transport, ledger exactness, finite reindexing row, and measurable-image bridge assemble the public reusable* $\mathsf{NameCert}_{\mathsf{Independence}^{\uparrow}}$ *export for the finite-family scope. Its carrier and classifier are the rows of Definition 187.6; its primitive boundary is the scoped surface of Definition 187.7; its finite-product and reindexing stability are supplied by Definition 187.11 and Definition 187.12; and its binary measurable-image transport is the bridge of Definition 187.13. This public export is reusable by downstream finite-family probability chapters without adding conditional-expectation, martingale, sigma-completion, host equality, host function equality, or standard bridge rows.*

Proof. Apply Definition 187.6. It supplies the finite-family carrier, factorisation classifier, stability transport, ledger exactness, and certificate packet for carried random-variable families over the displayed $\mathsf{ProbSpace}^{\uparrow}$ source.

Apply Definition 187.7 to fix the visible primitive boundary. The boundary identifies the exact $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist-readable cylinder, and product-fold rows consumed by the packet, and it excludes the probability-theoretic rows outside the finite-family surface.

For reusable finite-family stability, apply Definition 187.11 and Definition 187.12. These rows make the product fold and the factorisation comparison invariant under certified finite reindexing.

Finally apply Definition 187.13. It gives the binary measurable-image transport row over the same factorisation classifier. Collecting these rows produces the public $\mathsf{NameCert}_{\mathsf{Independence}^{\uparrow}}$ export for the stated finite-family scope. □

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow public namecert export`

Closure status for $\mathsf{NameCert}_{\mathsf{Independence}^{\uparrow}}$**.**

Constructive story: An independence packet starts with a carried $\mathsf{ProbSpace}^{\uparrow}$ source, a finite $\mathsf{FinSet}^{\uparrow}$ index spine, and one carried $\mathsf{RandomVar}^{\uparrow}$ map at each index, then displays both the marginal $\mathsf{Distribution}^{\uparrow}$ endpoints and the joint distribution endpoint of the tuple map. The carrier accepts only finite families that also show, for every finite family of target events, a product-cylinder readback, a finite $\mathsf{Real}^{\uparrow}$ product fold of marginal masses, and the factorisation comparison between the joint cylinder mass and that product; the classifier compares two presentations by transporting the source, finite positions, random-variable rows, marginal and joint distributions, cylinders, and real products through the named dependency classifiers. The source therefore covers finite carried random-variable families, the pattern is the product-cylinder factorisation equation together with finite reindexing and measurable-image transport, and the ledger keeps

exactly the cylinder readback, marginal spine, product fold, joint endpoint, and provenance rows visible. Since finite-position reindexing, distribution endpoint transport, readable-preimage transport, and Real$^{\uparrow}$ multiplication congruence preserve the factorisation row, an admitted finite family supplies the carrier witness, the classifier is reflexive, symmetric, and transitive componentwise, and transported families remain in the carrier. A finite-family row h lies in the signature gap of an Independence$^{\uparrow}$ token p when the source-domain accepts its probability source, index spine, random variables, and factorisation ledger, the signature relation pairs that row with an answer family and evidence ledger, and token introduction names the answer as p; unfolded, every cylinder mass in h is classified with the product of the corresponding marginal masses encoded by p. The paper bridge is the measurable-image and finite-reindexing readback for this finite factorisation surface, not a bridge to completed probability spaces. The chapter registers this finite-factorisation construction under the name Independence$^{\uparrow}$; independence is a displayed product equation over carried distributions, with no conditional-expectation row, sigma-completion primitive, or host function equality introduced.

Theory closure: TheoryGate(NameCert$_{\text{Independence}\uparrow}$, maturePackageClosed)

Scope closed: The scoped finite-family carrier, factorisation classifier, stability transport, ledger exactness, finite reindexing, and measurable-image bridge are bound over the displayed RandomVar$^{\uparrow}$, Distribution$^{\uparrow}$, FinSet$^{\uparrow}$, Permutation$^{\uparrow}$, Real$^{\uparrow}$, BHist-readable cylinder, and product-fold rows.

Formal verification: FormalStatus(NameCert$_{\text{Independence}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.IndependenceUp.IndependenceBinaryFactorization\_measurable\_image\_bridge`

Bridge status: paperBridge

Not claimed: Full probability theory, sigma-algebra completion, conditional expectation, martingales, and standard bridge data are not claimed here.

Upgrade path: No further theory-axis step remains; the chapter is at mature closure, with bridge verification tracked independently.

Theorem 187.15 (Independence empty index family satisfies the finite factorisation row)**.** *Let R be a finite-family carrier for* Independence$^{\uparrow}$ *as in Definition 187.1, with carried source S, finite index carrier I, displayed joint endpoint μ_X^J, and finite* Real$^{\uparrow}$ *product fold Π_R. Assume that I has no positions, and let B be the unique measurable target-event family on I. Write $\Omega_\varnothing$ for the total event of the empty-product target of the empty tuple family. Suppose:*

1. *the displayed product-cylinder ledger for R reads the empty family as the target total event,*

$$\mathrm{Cyl}_X(B) = \Omega_\varnothing;$$

2. *the hypotheses of Definition 186.5 hold for the empty tuple map, its joint endpoint μ_X^J, and the target total event $\Omega_\varnothing$;*

3. *the empty marginal spine folds to the multiplicative unit,*

$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R 1_R.$$

Then the displayed empty-family data satisfy the factorisation clause

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Consequently, these displayed data carry the finite factorisation row of Definition 187.3 for the empty index family.

Lean4 ✓ `BEDC.Derived.IndependenceUp.IndependenceFiniteFactorizationRow empty index factorization row`

Proof. Because I has no positions, there is only the empty measurable event family B. By the cylinder hypothesis, the left endpoint may be read at the total event of the empty-product target:

$$\mu_X^J(\mathrm{Cyl}_X(B)) = \mu_X^J(\Omega_\varnothing).$$

Apply Definition 186.5 to the empty tuple map and its joint endpoint μ_X^J. This gives

$$\mu_X^J(\Omega_\varnothing) \sim_R 1_R.$$

Reading the cylinder as $\Omega_\varnothing$ yields

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R 1_R.$$

The empty-fold hypothesis gives

$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R 1_R.$$

By symmetry of the $\mathsf{Real}^\uparrow$ classifier,

$$1_R \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Transitivity of the same classifier from Definition 38.9 therefore yields

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

This is exactly the factorisation clause of Definition 187.3 for the unique empty event family. Since the product-cylinder ledger and the finite product fold are already part of the displayed data, the empty family carries the stated finite factorisation row. □

Theorem 187.16 (Independence singleton index family satisfies the finite factorisation row)**.** *Let R be a finite-family carrier for* $\mathsf{Independence}^\uparrow$ *as in Definition 187.1, with carried source S, finite index carrier I, random-variable spine $(X_i)_{i\in I}$, marginal endpoints $(\mu_i)_{i\in I}$, displayed joint endpoint μ_X^J, and finite* $\mathsf{Real}^\uparrow$ *product fold Π_R. Assume that I has exactly one certified position i_0, and let B be a measurable target-event family on I. Suppose:*

1. *the singleton index certificate identifies every quantified index in the cylinder ledger with i_0;*
2. *the product-cylinder readback for R reduces at B to the unique marginal event, so for every carried source point s,*
$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \Longleftrightarrow X_{i_0}(s) \in B_{i_0};$$
3. *the* $\mathsf{Distribution}^\uparrow$ *pushforward rows for the singleton tuple endpoint and the marginal endpoint read this source-event equivalence as*
$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \mu_{i_0}(B_{i_0});$$
4. *the singleton* $\mathsf{Real}^\uparrow$ *product fold evaluates the marginal spine at its unique position:*
$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R \mu_{i_0}(B_{i_0}).$$

Then the displayed singleton-family data satisfy the factorisation clause

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Consequently, when the four displayed hypotheses hold for every measurable singleton event family B, these data carry the finite factorisation row of Definition 187.3 for the singleton index family.

Proof. Fix the measurable singleton event family B. Since I has exactly one certified position i_0, the universal clause in the product-cylinder ledger has no component other than i_0. Hence the cylinder readback reduces to

$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \Longleftrightarrow X_{i_0}(s) \in B_{i_0}.$$

By the $\mathsf{Distribution}^{\uparrow}$ pushforward rows named in the hypotheses, this displayed source-event equivalence compares the joint singleton-cylinder endpoint with the marginal endpoint:

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \mu_{i_0}(B_{i_0}).$$

The marginal spine has exactly one entry, namely

$$\mathsf{MargSp}_X(B)(i_0) = \mu_{i_0}(B_{i_0}).$$

The singleton product-fold hypothesis therefore gives

$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R \mu_{i_0}(B_{i_0}).$$

By symmetry of the $\mathsf{Real}^{\uparrow}$ classifier from Definition 38.9, this comparison reverses to

$$\mu_{i_0}(B_{i_0}) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Transitivity of the same classifier composes the joint-to-marginal comparison with the reversed singleton-fold comparison and yields

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

This is exactly the factorisation clause of Definition 187.3 at the singleton event family B.

If the displayed hypotheses hold for every measurable singleton event family, the same argument applies to each B. The carrier already supplies the product-cylinder ledger, marginal distribution spine, finite product fold, and joint endpoint required by Definition 187.3; hence the singleton index family carries the finite factorisation row. □

Theorem 187.17 (Independence two-element index family satisfies the finite factorisation row)**.** *Let R be a finite-family carrier for $\mathsf{Independence}^{\uparrow}$ as in Definition 187.1, with carried source S, finite index carrier I, random-variable spine $(X_i)_{i\in I}$, marginal endpoints $(\mu_i)_{i\in I}$, displayed joint endpoint μ_X^J, and finite $\mathsf{Real}^{\uparrow}$ product fold Π_R. Assume that I has exactly two certified positions i_0 and i_1, and let B be a measurable target-event family on I. Suppose:*

1. *the two-position index certificate identifies every quantified index in the cylinder ledger with exactly one of i_0 and i_1;*
2. *the product-cylinder readback for R reduces at B to the binary component event, so for every carried source point s,*

$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \Longleftrightarrow \big(X_{i_0}(s) \in B_{i_0}\big) \wedge \big(X_{i_1}(s) \in B_{i_1}\big);$$

3. *the displayed product-cylinder ledger together with the $\mathsf{Distribution}^{\uparrow}$ pushforward rows reads the joint cylinder endpoint as the binary product of the two marginal endpoints,*

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \mu_{i_0}(B_{i_0}) \cdot_R \mu_{i_1}(B_{i_1});$$

4. the two-element $\mathsf{Real}^{\uparrow}$ *product fold evaluates the marginal spine in the certified order as*

$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R \mu_{i_0}(B_{i_0}) \cdot_R \mu_{i_1}(B_{i_1}).$$

Then the displayed two-position data satisfy the factorisation clause

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Consequently, when the four displayed hypotheses hold for every measurable two-position event family B*, these data carry the finite factorisation row of Definition 187.3 for the two-element index family.*

Proof. Fix the measurable two-position event family B. Since I has exactly the certified positions i_0 and i_1, the universal clause in the product-cylinder ledger has no component other than those two positions. Hence the cylinder readback reduces to

$$s \in X^{-1}(\mathrm{Cyl}_X(B)) \iff \big(X_{i_0}(s) \in B_{i_0}\big) \wedge \big(X_{i_1}(s) \in B_{i_1}\big).$$

The joint endpoint hypothesis states that the displayed cylinder, after being read through the $\mathsf{Distribution}^{\uparrow}$ pushforward rows, is classified with the product of the two marginal masses:

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \mu_{i_0}(B_{i_0}) \cdot_R \mu_{i_1}(B_{i_1}).$$

The marginal spine has exactly the two certified entries

$$\mathsf{MargSp}_X(B)(i_0) = \mu_{i_0}(B_{i_0}), \qquad \mathsf{MargSp}_X(B)(i_1) = \mu_{i_1}(B_{i_1}).$$

By the two-element fold hypothesis, the finite $\mathsf{Real}^{\uparrow}$ product fold evaluates this spine as

$$\Pi_R(\mathsf{MargSp}_X(B)) \sim_R \mu_{i_0}(B_{i_0}) \cdot_R \mu_{i_1}(B_{i_1}).$$

Symmetry of the $\mathsf{Real}^{\uparrow}$ classifier from Definition 38.9 reverses the fold comparison and gives

$$\mu_{i_0}(B_{i_0}) \cdot_R \mu_{i_1}(B_{i_1}) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

Transitivity of the same classifier composes the joint-to-product comparison with this reversed fold comparison, yielding

$$\mu_X^J(\mathrm{Cyl}_X(B)) \sim_R \Pi_R(\mathsf{MargSp}_X(B)).$$

This is exactly the factorisation clause of Definition 187.3 at the two-position event family B.

If the displayed hypotheses hold for every measurable two-position event family, the same argument applies to each B. The finite-family carrier already supplies the product-cylinder ledger, marginal distribution spine, finite product fold, and joint endpoint required by Definition 187.3; therefore the two-element index family carries the stated finite factorisation row. □

Chapter 188

A Concrete Naming Certificate for $\mathsf{CondExp}^{\uparrow}$

The conditional-expectation interface $\mathsf{CondExp}^{\uparrow}$ packages the L^2 projection of an integrable random variable onto the subspace of functions measurable with respect to a sub-σ-algebra. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91).

188.1 Carrier definition

Theorem 188.1 (CondExp carrier obligation)**.** *Let* $\omega_T, \omega_S, \mathrm{pre}_X, x, p, r : \mathsf{BHist}$*. A total-readback certificate* $\mathsf{RandomVarTotalReadbackCertificate}(\omega_T, \omega_S, \mathrm{pre}_X)$*, a Hilbert singleton projection witness* $\mathsf{HilbertSingletonProjectionWitness}(x, p)$*, and a continuation ledger* $\mathsf{Cont}(p, r, x)$ *jointly form the carrier packet for the singleton* $\mathsf{CondExp}^{\uparrow}$ *interface. The visible packet is*

$$(\omega_T, \omega_S, \mathrm{pre}_X, x, p, r, \mathsf{RandomVarTotalReadbackCertificate}(\omega_T, \omega_S, \mathrm{pre}_X), \mathsf{HilbertSingletonProjectionWitness}(x, p), \mathsf{Co}$$

It records the target total event, the source total event, the chosen readback endpoint, the integrable-history endpoint, the projected endpoint, the residual history, the total-preimage certificate, the projection witness, and the residual continuation row.

Lean4 ✓ `BEDC.Derived.CondExpUp.CondExpCarrier packet`
↪ variant `BEDC.Derived.CondExpUp.CondExpCarrier obligation`

Proof. Unpack the total-readback certificate through Definition 185.14. Its carried-total bridge and chosen-readback clause place pre_X on the $\mathsf{RandomVar}^{\uparrow}$ source side over the displayed endpoints ω_T and ω_S.

Unpack the Hilbert projection witness through Definition 91.14. It supplies the carried endpoint x, the projected endpoint p, and the singleton classifier row for p.

The remaining field is not a hidden analytic object: it is the displayed kernel continuation $\mathsf{Cont}(p, r, x)$. Repacking these three rows gives exactly the carrier packet stated above. □

188.2 The certificate

Theorem 188.2 (CondExp classifier obligation)**.** *Consider two singleton* $\mathsf{CondExp}^{\uparrow}$ *carrier packets with histories* $(\omega_T, \omega_S, \mathrm{pre}_X, x, p, r)$ *and* $(\omega'_T, \omega'_S, \mathrm{pre}'_X, x', p', r')$*. If their source random-variable*

endpoints satisfy $\mathsf{hsame}(\omega_T, \omega'_T)$, $\mathsf{hsame}(\omega_S, \omega'_S)$, *and* $\mathsf{hsame}(\mathrm{pre}_X, \mathrm{pre}'_X)$, *and their projection endpoints satisfy* $\mathsf{hsame}(x, x')$ *and* $\mathsf{hsame}(p, p')$, *then the* $\mathsf{CondExp}^{\uparrow}$ *classifier identifies the two carrier packets. Carrier transport is exhausted by terminal readback uniqueness for the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint and by the Hilbert singleton projection transport and idempotence rows for the projected endpoint.*

Lean4 ✓ `BEDC.Derived.CondExpUp.CondExpClassifier obligation`

Proof. First compare the random-variable side. Apply Definition 185.16 to the two readback certificates over the total target endpoints. The source endpoints are therefore in the same hsame class after the total-preimage row is read through the carried certificate.

Next compare the Hilbert side. The projected endpoints are carried by the same singleton carrier from Definition 91.14, and Definition 91.16 says that projecting a projected endpoint stays in the same singleton classifier class.

Finally repack the two transported sides with the visible continuation rows. Since the carrier packet of Definition 188.1 contains no further hidden field, these transported rows are exactly the classifier comparison for the two $\mathsf{CondExp}^{\uparrow}$ packets. □

Theorem 188.3 (CondExp projection obligation)**.** *For every carried singleton Hilbert endpoint* $x : \mathsf{BHist}$, *the* $\mathsf{Hilbert}^{\uparrow}$ *rows Definition 91.15, Definition 91.16, and Definition 91.17 supply the singleton* $\mathsf{CondExp}^{\uparrow}$ *projection row, its idempotence row, and its residual orthogonality row. Thus the conditional-expectation projection surface is the Hilbert singleton projection surface restricted to the displayed* $\mathsf{RandomVar}^{\uparrow}$ *total-readback carrier packet.*

Lean4 ✓ `BEDC.Derived.CondExpUp.CondExpProjection obligation`

Proof. Start with the carried endpoint x. By Definition 91.15, it admits a projected endpoint in the Hilbert singleton carrier, with the projection witness needed by Definition 188.1.

The idempotence row is exactly Definition 91.16. It states that the projected endpoint remains in its singleton classifier class after a second projection.

The residual row is exactly Definition 91.17. Reading that row inside the carrier packet gives the residual orthogonality package required for the singleton conditional-expectation interface. □

Theorem 188.4 (CondExp ledger and exactness obligation)**.** *In the singleton* $\mathsf{CondExp}^{\uparrow}$ *carrier packet of Definition 188.1, the only hidden ledger fields are the* $\mathsf{RandomVar}^{\uparrow}$ *terminal-preimage exactness row and the* $\mathsf{Hilbert}^{\uparrow}$ *residual-zero exactness row. Consequently the visible packet gives coverage and exactness for the singleton conditional-expectation interface: the source event readback is classifier-exact over the source total event, the projected endpoint is the Hilbert singleton projection endpoint, and the residual orthogonality row is the constant-zero inner-product row.*

Lean4 ✓ `BEDC.Derived.CondExpUp.CondExpCarrierPacket ledger exactness`
↪ variant `BEDC.Derived.CondExpUp.CondExpLedgerExactness obligation`

Proof. The random-variable ledger is Definition 185.15. It gives source-measurability for the chosen total-target preimage and identifies that chosen preimage with the source total event by the source classifier.

The Hilbert ledger is Definition 91.17. It reads every residual inner-product against a carried endpoint as the constant RealUp zero row.

Combine these two ledger rows with the carrier packet. There is no extra field in the packet beyond the total-readback certificate, the Hilbert projection witness, and the displayed continuation row, so the same two exactness rows give the claimed coverage for the singleton $\mathsf{CondExp}^{\uparrow}$ interface. □

Theorem 188.5 (CondExp singleton standard bridge boundary)**.** *The singleton* $\mathsf{CondExp}^{\uparrow}$ *carrier packet of Definition 188.1, the classifier obligation of Definition 188.2, the projection obligation of Definition 188.3, and the ledger exactness theorem of Definition 188.4 give a conservative bridge from the BEDC packet* (RandomVarTotalReadbackCertificate, HilbertSingletonProjectionWitness, Cont) *to the standard singleton conditional-expectation interface. The bridge exports exactly the displayed total-readback source event, Hilbert singleton projection endpoint, residual continuation row, classifier transport, and ledger exactness rows. It does not export tower properties, non-total sub-σ-algebra measurability, or full conditional-expectation theory.*

Lean4 ✓ `BEDC.Derived.CondExpUp.CondExpSingleton standard bridge boundary`

Proof. Use the carrier packet to expose the three concrete BEDC inputs: the $\mathsf{RandomVar}^{\uparrow}$ total-readback certificate, the $\mathsf{Hilbert}^{\uparrow}$ singleton projection witness, and the continuation row.

The classifier obligation transports equality of displayed carrier data into the singleton $\mathsf{CondExp}^{\uparrow}$ classifier. The projection obligation supplies the singleton projection, idempotence, and residual rows from the Hilbert singleton surface.

Finally, the ledger exactness theorem identifies the only hidden bookkeeping with the terminal-preimage exactness row and the residual-zero row. Repacking only these rows yields the standard singleton interface, while every tower, non-total measurability, and unrestricted conditional-expectation clause remains outside the bridge boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$**.**

Constructive story: A conditional expectation in this scoped chapter is the singleton packet obtained by joining three displayed rows: a $\mathsf{RandomVar}^{\uparrow}$ total-readback certificate for the source and target total events, a $\mathsf{Hilbert}^{\uparrow}$ singleton projection witness for an integrable endpoint x and projected endpoint p, and a continuation row $\mathsf{Cont}(p, r, x)$ recording the residual history. The carrier accepts only packets with those three witnesses and their six visible histories, while the classifier identifies two packets when the total-readback histories, integrable endpoints, projected endpoints, and residual rows are transported by hsame and by the consumed $\mathsf{RandomVar}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ certificates. The source therefore covers only the total-readback endpoint of a carried random variable, the pattern is Hilbert singleton projection with idempotence and residual-zero orthogonality, and the ledger keeps the terminal-preimage exactness row and the residual-zero inner-product row as the hidden explanation for the visible projection. Since terminal readback uniqueness, singleton projection transport, and history-sameness are stable under the displayed comparisons, a carried singleton packet supplies the nonempty witness, the classifier is reflexive, symmetric, and transitive, and carrier membership survives classifier replacement. A packet h lies in the signature gap of a $\mathsf{CondExp}^{\uparrow}$ token p_0 when the source-domain accepts the total-readback and projection packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p_0; after unfolding, the projected endpoint is the Hilbert singleton projection of the displayed integrable endpoint and the residual row is the constant-zero orthogonality witness. The standard bridge is deliberately narrow, exporting only this singleton conditional-expectation boundary and not tower laws or non-total sub-σ-algebra measurability. The chapter registers this singleton-projection construction under the name $\mathsf{CondExp}^{\uparrow}$; conditional expectation is a RandomVarUp readback plus a Hilbert projection ledger, with no unrestricted host conditional-expectation operator introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the carrier packet, classifier obligation, projection obligation, ledger exactness rows, and singleton standard bridge boundary as the conservative singleton $\mathsf{CondExp}^{\uparrow}$ bridge.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CondExpUp.CondExpCarrierPacket\_ledger\_exactness`

Bridge status: paperBridge

Not claimed: This chapter does not claim tower properties, sub-σ-algebra measurability beyond the total-readback endpoint, Lean-side standard-bridge verification, or full conditional-expectation theory.

Upgrade path: No further theory-axis grade is above matureClosure; bridgeChecked verification remains a separate formal-axis target.

Chapter 189

A Concrete Naming Certificate for $\mathsf{MarkovChain}^{\uparrow}$

The Markov-chain interface $\mathsf{MarkovChain}^{\uparrow}$ packages a sequence of random variables in which the conditional distribution of each step depends only on the previous step. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186).

189.1 Carrier definition

Definition 189.1 (MarkovChain BHist transition carrier)**.** *A* $\mathsf{MarkovChain}^{\uparrow}$ *carrier is a finite* BHist *transition-row packet over a source* $\mathsf{ProbSpace}^{\uparrow}$ *carrier. It consists of a unary time ledger, carried* $\mathsf{RandomVar}^{\uparrow}$ *rows* $X_n : S \to E$ *for each displayed time row,* $\mathsf{Distribution}^{\uparrow}$ *rows for the laws of those random variables, transition rows* $K_n(e, -)$ *whose endpoints are probability distributions on the next-state event family, and the visible* hsame/Cont/Pkg *provenance tying each transition row to the adjacent pair* X_n, X_{n+1}.

The packet is accepted only when the Cont *ledger records that the next-step distribution read for* X_{n+1} *is sourced from the current-state endpoint of* X_n *and the displayed transition row* K_n*. The carrier contains no host stochastic-process object, no filtration, no conditional-expectation primitive, and no standard Markov-kernel bridge; those may be named only by later certificates that consume this finite transition ledger.*

Lean4 (definition) `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier`

Theorem 189.2 (MarkovChain kernel classifier stability)**.** *Let two* $\mathsf{MarkovChain}^{\uparrow}$ *transition packets have the same displayed unary time ledger. Suppose their* $\mathsf{ProbSpace}^{\uparrow}$ *sources,* $\mathsf{RandomVar}^{\uparrow}$ *rows,* $\mathsf{Distribution}^{\uparrow}$ *law rows, and transition rows are related componentwise by* hsame*, and suppose the transition* Cont *ledgers and* Pkg *provenance rows are transported along those comparisons. Then the transported rows form an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *carrier, and the transition classifier has the same current-state to next-law shape.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainKernelClassifier stability`

Proof. Use the classifier field of the roadmap ten-field template. The source rows are the $\mathsf{ProbSpace}^{\uparrow}$ carrier, the displayed $\mathsf{RandomVar}^{\uparrow}$ sequence, and the $\mathsf{Distribution}^{\uparrow}$ law rows; the carrier row is the finite BHist packet of Definition 189.1.

Transport each source row through the displayed hsame comparison. The $\mathsf{RandomVar}^{\uparrow}$ rows keep their domain in the same probability-space packet, and the $\mathsf{Distribution}^{\uparrow}$ rows keep the law endpoints attached to the transported random-variable rows.

Finally transport the transition Cont ledger and Pkg provenance. Since acceptance asks only that the next-step law be read through the adjacent carried rows and the displayed transition row, the transported packet satisfies the same carrier predicate without importing a host stochastic-process equality. □

Theorem 189.3 (MarkovChain transition ledger exactness)**.** *For every accepted* $\mathsf{MarkovChain}^{\uparrow}$ *carrier, the transition ledger exposes exactly the* BHist *rows for the* $\mathsf{ProbSpace}^{\uparrow}$ *source, the displayed* $\mathsf{RandomVar}^{\uparrow}$ *rows, the corresponding* $\mathsf{Distribution}^{\uparrow}$ *law rows, the transition rows* K_n*, the continuation row from the random-variable and transition rows, the provenance row from the probability-space and law rows, the endpoint row from provenance and transition-continuation rows, and the package provenance. It does not expose a host Markov process, a filtration, a conditional-expectation row, or a standard kernel bridge.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket ledger exactness`

Proof. Inspect the carrier definition. Its accepted rows are precisely the source probability-space row, the displayed random-variable rows, the displayed distribution rows, the transition rows, and the provenance entries that bind adjacent time steps.

The ledger field of the ten-field template is the finite Cont display explaining how the transition-continuation row is read from the displayed random-variable row and K_n, how the provenance row is read from the probability-space and law rows, and how the endpoint is read from the provenance and transition-continuation rows. Projecting the ledger therefore returns only those named rows and their package provenance.

No constructor in the carrier produces a filtration, a conditional-expectation object, or a host stochastic-process carrier. Hence none of those external stochastic objects is exported by the transition ledger. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier transition ledger exactness`

Theorem 189.4 (MarkovChain transition-kernel row factorization)**.** *Every transition-kernel read of an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *carrier factors through the displayed finite rows of the carrier: the* $\mathsf{ProbSpace}^{\uparrow}$ *event packet, the current and next* $\mathsf{RandomVar}^{\uparrow}$ *rows, the corresponding* $\mathsf{Distribution}^{\uparrow}$ *law rows, the transition row* K_n*, the* Cont *provenance from adjacent random-variable rows to the next-law endpoint, componentwise* hsame *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ *row. The row may be consumed as a finite transition-kernel datum inside the BEDC packet, but it is not a standard Markov-kernel object and does not carry a host stochastic process, filtration, conditional expectation, or standard kernel bridge.*

Proof. Start from the carrier definition Definition 189.1. It admits only the source probability-space row, displayed current and next random-variable rows, law rows, transition rows K_n, and the visible hsame/Cont/Pkg provenance tying each transition to its adjacent endpoint pair.

Apply transition ledger exactness from Definition 189.3. The ledger exposes exactly the transition BHist rows, the continuation row from the random-variable and transition rows, the provenance row from the probability-space and law rows, the endpoint row from those provenance and continuation rows, and package provenance.

Use the source-boundary theorem Definition 189.5. It keeps the probability source, random variables, and distribution laws inside the joint $\mathsf{ProbSpace}^{\uparrow}/\mathsf{RandomVar}^{\uparrow}/\mathsf{Distribution}^{\uparrow}$ packet, so

the transition row cannot replace them with external probability, random-variable, or distribution objects.

Thus every transition-kernel read factors through the displayed finite rows and their structural transport. Since none of those rows is a standard Markov-kernel object, and since the carrier contains no stochastic-process, filtration, conditional-expectation, or standard-kernel bridge coordinate, the factorized row remains a BEDC transition datum only. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionCarrier transition kernel row factorization`

Theorem 189.5 (MarkovChain probability-random-variable source boundary)**.** *The* $\mathsf{MarkovChain}^{\uparrow}$ *source boundary is the joint BEDC packet consisting of the underlying* $\mathsf{ProbSpace}^{\uparrow}$ *event-and-measure rows, the time-indexed* $\mathsf{RandomVar}^{\uparrow}$ *measurable-preimage rows, and the* $\mathsf{Distribution}^{\uparrow}$ *pushforward rows for the laws of the displayed random variables. Every transition row in an accepted carrier must read its source event through that packet, and no transition row may replace it with an external probability space, external random variable, or host distribution object.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket source boundary`
Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket semantic source boundary`

Proof. Project the source field of the ten-field template. The probability-space source is the displayed $\mathsf{ProbSpace}^{\uparrow}$ packet, and every random-variable row is carried over that same source packet.

Next project the law rows through $\mathsf{Distribution}^{\uparrow}$. Each law endpoint is a pushforward row for the corresponding $\mathsf{RandomVar}^{\uparrow}$ map, so the row remains inside the BEDC source boundary.

The transition row may consume those source, random-variable, and law rows through Cont and Pkg provenance only. This gives the stated boundary and excludes host probability spaces, host random variables, and host distribution objects. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier source boundary`
↪ variant `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier probspace randomvar source boundary`

Theorem 189.6 (MarkovChain NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ *obligation surface is the transition carrier of Definition 189.1, the componentwise kernel classifier of Definition 189.2, the transition ledger exactness row of Definition 189.3, and the probability-random-variable source boundary of Definition 189.5. The surface names only* BHist *rows,* hsame *comparisons,* Cont *transition displays, and* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*, and* $\mathsf{Distribution}^{\uparrow}$ *dependency certificates.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionLedgerSurface semantic name certificate`
↪ variant `BEDC.Derived.MarkovChainUp.MarkovChainTransitionCarrier semantic name certificate`

Proof. Assemble the obligation surface by the roadmap ten-field template. The traditional target is a discrete-time Markov chain; the BEDC source is the probability-space, random-variable, and distribution dependency packet; the carrier is the finite transition packet.

The classifier and stability fields are supplied by Definition 189.2. The ledger and exactness fields are supplied by Definition 189.3.

The source-boundary field is Definition 189.5. Together these rows give the named $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ obligation surface while keeping host stochastic-process structure outside the certificate. □

Theorem 189.7 (MarkovChain scoped kernel dependency package)**.** *The* $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ *obligation surface of Definition 189.6 is scoped exactly by the* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, *and* $\mathsf{Distribution}^{\uparrow}$ *dependency certificates, the finite unary time ledger, the transition rows* K_n, BHist *rows,* hsame *comparisons,* Cont *displays, and* Pkg *provenance. No filtration, conditional-expectation primitive, host stochastic process, or standard Markov kernel is read by this scoped package.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket dependency unary closure`
Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainKernelClassifier endpoint confluence`

Proof. Project the carrier row from Definition 189.1. Its constructors expose only the source probability-space packet, the carried random-variable rows, the law rows, the unary time ledger, transition rows, and the displayed provenance.

Next use Definition 189.2 and Definition 189.3. These identify the classifier transport and ledger reads with componentwise hsame, Cont, and Pkg rows over the same finite packet.

Finally apply Definition 189.5 together with Definition 189.6. The dependency rows consumed by those projections are exactly the $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{Distribution}^{\uparrow}$ certificates and the finite transition ledger, so the excluded stochastic-process, filtration, conditional-expectation, and standard-kernel objects remain outside the scoped package. □

Theorem 189.8 (MarkovChain transition-packet scope)**.** *The paper scope of* MarkovChainTransitionPacket *is the finite* BHist *packet consisting of* ProbSpacePublicEventPacket *source rows,* DistributionPushforwardSourceSpec *law rows,* RandomVarTotalReadbackCertificate *current-to-next readback rows,* Cont *provenance rows,* hsame *transport rows, and the visible* BHist *time, current-state, next-state, transition, provenance, and ledger endpoints. It introduces no host stochastic-process carrier, filtration, conditional-expectation primitive, or standard Markov-kernel object.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionCarrier transition packet scope`

Proof. Unfold the transition-packet rows. The source field is the public probability event packet, the law field is the distribution pushforward source specification, and the readback field is the random-variable total readback certificate.

The remaining rows are the two displayed Cont ledgers: one from the current state and transition row to the provenance endpoint, and one from that provenance endpoint and law row to the ledger endpoint. The only transport allowed across the packet is componentwise hsame on those displayed BHist rows, so the external stochastic objects listed above are not part of the scope. □

Theorem 189.9 (MarkovChain Lean target surface)**.** *The checked* MarkovChainTransitionPacket_kernel_classifier_ *theorem states that componentwise* hsame *transport of the source, time, current, next, law, and transition rows, together with transported* Cont *provenance and ledger rows, preserves the* MarkovChainTransitionPacket *carrier and returns* hsame *comparisons for the provenance and ledger endpoints.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket kernel classifier stability`

Proof. Read the checked theorem as the transition-packet classifier transport row. The inputs are precisely the rows isolated in Definition 189.8.

The theorem transports the probability source, unary time row, random-variable current and next endpoints, law row, and transition row by hsame. It then rebuilds the two Cont ledgers at the transported endpoints, producing both the transported carrier packet and the displayed provenance and ledger endpoint comparisons. □

Theorem 189.10 (MarkovChain scoped obligation package)**.** *The transition-packet stability theorem of Definition 189.9 supplies the scoped carrier/classifier transport row for* $\mathsf{MarkovChain}^{\uparrow}$*. Ledger exactness and the assembled* $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ *remain paper obligations until their corresponding rows are separately formalized.*

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionCarrier scoped obligation package`

Proof. Combine Definition 189.8 with Definition 189.9. The former fixes the visible carrier rows, and the latter proves that the classifier transport preserves that packet.

The remaining fields of the local name certificate are not consequences of the transport theorem alone. They require the ledger exactness row and the final assembly of the $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$ surface, so they stay as stated paper obligations. □

189.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$**.**

Constructive story: A Markov-chain packet is generated from a displayed $\mathsf{ProbSpace}^{\uparrow}$ source, a unary time ledger, carried $\mathsf{RandomVar}^{\uparrow}$ rows, $\mathsf{Distribution}^{\uparrow}$ law rows, transition rows, Cont displays, hsame transport, and Pkg provenance. The carrier accepts only finite transition ledgers whose next-law rows are sourced from the adjacent carried random-variable rows, so host stochastic-process objects, filtrations, conditional-expectation primitives, and standard kernel bridges remain outside the certificate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public surface consists of the finite-prefix restriction, transition-concat closure, splice transport, and finite-suffix carrier rows displayed in this chapter, over the same $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, hsame, Cont, and Pkg transition packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.MarkovChainUp.MarkovChainTransitionCarrier\_semantic\_name\_certificate`

Bridge status: none

Not claimed: No host stochastic-process carrier, filtration, conditional-expectation construction, stationary law, martingale property, or standard kernel bridge is claimed.

Upgrade path: No paper-axis strengthening remains; stronger status is purely formal or bridge-axis work.

Definition 189.11 (MarkovChain finite-prefix restriction)**.** *Let P be a displayed* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet over a unary time ledger with transition-time rows $0, \ldots, n+k-1$ and endpoint rows $0, \ldots, n+k$. The finite-prefix restriction $P_{\leq n}$ is the subpacket obtained by keeping the same* $\mathsf{ProbSpace}^{\uparrow}$ *source, the endpoint rows $X_0, \ldots, X_n$, the* $\mathsf{Distribution}^{\uparrow}$ *law rows read by the first n transitions, the transition rows $K_0, \ldots, K_{n-1}$, and exactly the* hsame/Cont/Pkg *provenance entries whose adjacent endpoints both lie in $X_0, \ldots, X_n$.*

No row with transition index at least n is copied into $P_{\leq n}$, and no row is reindexed except by the evident inclusion of the prefix ledger into the original unary ledger.

Lean4 (definition) `BEDC.Derived.MarkovChainUp.MarkovChainFinitePrefixRestriction`

Theorem 189.12 (MarkovChain finite-prefix restriction carrier)**.** *Let P be an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet in the sense of Definition 189.1. Suppose its unary time ledger has length $n+k$, with endpoint rows $X_0, \ldots, X_{n+k}$ and transition rows $K_0, \ldots, K_{n+k-1}$. Then the finite-prefix restriction $P_{\leq n}$ of Definition 189.11, consisting of the same* $\mathsf{ProbSpace}^{\uparrow}$ *source, the first $n+1$ displayed* $\mathsf{RandomVar}^{\uparrow}$ *endpoint rows, the first n next-law* $\mathsf{Distribution}^{\uparrow}$ *rows, the first n transition rows, and the inherited adjacent* hsame/Cont/Pkg *provenance, is again an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *carrier.*

Proof. The carrier predicate of Definition 189.1 is local along adjacent transition rows. It asks for a finite unary time ledger, carried $\mathsf{RandomVar}^{\uparrow}$ endpoint rows on that ledger, the displayed $\mathsf{Distribution}^{\uparrow}$ law rows read by the transition rows, and for each adjacent pair X_i, X_{i+1} a Cont ledger and Pkg provenance row recording that the next-step distribution for X_{i+1} is sourced from the current-state endpoint of X_i through the displayed transition row K_i.

The prefix ledger $0, \ldots, n$ is finite because it is obtained by induction on the first n successors of the original unary ledger. Its endpoint rows $X_0, \ldots, X_n$ are carried $\mathsf{RandomVar}^{\uparrow}$ rows because each is one of the carried endpoint rows of P. The retained next-law rows are $\mathsf{Distribution}^{\uparrow}$ rows for the same reason: each retained law row is one of the law rows already accepted in the original packet and is attached to the same adjacent transition where it is read.

It remains only to check the transition obligations. Fix a prefix transition index $i < n$. Since the original packet has length $n + k$, the same index also satisfies $i < n + k$. Acceptance of P therefore supplies the transition row K_i together with the displayed Cont ledger witnessing that the next-law row for X_{i+1} is read from the current-state endpoint of X_i through K_i, and the corresponding Pkg provenance. Both endpoints X_i and X_{i+1} lie in the retained endpoint list, and K_i lies among the retained transition rows, so this exact $\mathsf{Cont}/\mathsf{Pkg}$ entry is a well-formed adjacent transition entry of $P_{\leq n}$.

Every transition obligation of $P_{\leq n}$ is obtained in this way from a unique transition obligation of P, and the restriction introduces no additional transition row whose source, target, law, or provenance would need a separate witness. The source-boundary statement of Definition 189.5 also remains unchanged, because the restricted packet keeps the same $\mathsf{ProbSpace}^{\uparrow}$ source and only discards later rows. Thus all fields required by the MarkovChain transition carrier predicate are present in the restricted finite packet, so $P_{\leq n}$ is an accepted $\mathsf{MarkovChain}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier finite prefix restriction carrier`

Theorem 189.13 (MarkovChain transition-packet concatenation closure)**.** *Let P_1 be an accepted $\mathsf{MarkovChain}^{\uparrow}$ transition packet of length n over a unary time ledger T_1, with displayed random-variable rows*

$$X_0, X_1, \ldots, X_n$$

source law rows $\mu_0^X, \mu_1^X, \ldots, \mu_n^X$, transition rows $K_0^X, \ldots, K_{n-1}^X$, and transition provenance ledgers accepted by Definition 189.1. Let P_2 be an accepted $\mathsf{MarkovChain}^{\uparrow}$ transition packet of length m over a unary time ledger T_2, over the same displayed $\mathsf{ProbSpace}^{\uparrow}$ source, with displayed random-variable rows

$$Y_0, Y_1, \ldots, Y_m$$

source law rows $\mu_0^Y, \mu_1^Y, \ldots, \mu_m^Y$, transition rows $K_0^Y, \ldots, K_{m-1}^Y$, and accepted transition provenance ledgers. Suppose the shared endpoint is displayed by a $\mathsf{RandomVar}^{\uparrow}$ classifier row

$$\mathsf{hsame}(X_n, Y_0)$$

over that same $\mathsf{ProbSpace}^{\uparrow}$ source, and let the induced $\mathsf{Distribution}^{\uparrow}$ law comparison identify the endpoint law rows μ_n^X and μ_0^Y. If the unary ledgers are displayed with a concatenation row

$$\mathsf{Cont}(T_1, T_2, T_{12})$$

so that T_{12} is the unary ledger of length $n + m$, then the end-to-end displayed packet P_{12} is an accepted $\mathsf{MarkovChain}^{\uparrow}$ transition packet over T_{12}. Its random-variable row at time i is X_i for

$0 \le i \le n$ and Y_j for $i = n + j$ with $1 \le j \le m$; its transition rows are K_i^X for $0 \le i < n$ and the shifted rows K_j^Y for $0 \le j < m$, with the first shifted row read across the splice comparison $\mathsf{hsame}(X_n, Y_0)$.

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier transition concat closure`
Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainBHistTransitionCarrier adjacent transition transport`

Proof. Unpack the accepted carrier condition in Definition 189.1. The first packet supplies the common $\mathsf{ProbSpace}^{\uparrow}$ source, the random-variable rows X_i, the law rows μ_i^X, the transition rows K_i^X, and for every $0 \le i < n$ a displayed $\mathsf{Cont}/\mathsf{Pkg}$ provenance ledger certifying that the next-step law for X_{i+1} is read from the current-state endpoint of X_i through K_i^X. The second packet supplies the same kind of data for Y_j, μ_j^Y, and K_j^Y for $0 \le j < m$ over the same probability source.

Define the displayed concatenated row sequence by

$$Z_i := X_i \quad (0 \le i \le n), \qquad Z_{n+j} := Y_j \quad (1 \le j \le m).$$

Define its law rows by

$$\lambda_i := \mu_i^X \quad (0 \le i \le n), \qquad \lambda_{n+j} := \mu_j^Y \quad (1 \le j \le m).$$

The endpoint law at the splice is well-defined on the displayed carrier because the hypothesis $\mathsf{hsame}(X_n, Y_0)$ is a $\mathsf{RandomVar}^{\uparrow}$ comparison over the shared $\mathsf{ProbSpace}^{\uparrow}$ source, and the pushforward classifier of Definition 186.2 transports the law row of Y_0 to the law row already carried by X_n.

The transition rows of P_{12} are the visible list

$$K_0^X, \ldots, K_{n-1}^X, K_0^Y, \ldots, K_{m-1}^Y.$$

For indices $i < n$, the required adjacent provenance row from Z_i to Z_{i+1} is exactly the accepted provenance row of P_1. For indices $i = n + j$ with $1 \le j < m$, the required adjacent provenance row from $Z_{n+j} = Y_j$ to $Z_{n+j+1} = Y_{j+1}$ is exactly the accepted provenance row of P_2.

It remains to check the splice transition, the row using K_0^Y from time n to time $n + 1$. In P_2, acceptance says that the next-step law for Y_1 is read from the current-state endpoint Y_0 through K_0^Y, with its displayed $\mathsf{Cont}/\mathsf{Pkg}$ provenance. Transport the current-state endpoint and its law row along

$$\mathsf{hsame}(X_n, Y_0).$$

By Definition 189.2, this componentwise transport preserves acceptance of the transition row and preserves the current-state to next-law classifier shape. Therefore the transported splice row reads the same next-step law μ_1^Y from the concatenated current row $Z_n = X_n$ through K_0^Y.

The time ledger is finite and unary because T_1 and T_2 are unary and the displayed continuation row $\mathsf{Cont}(T_1, T_2, T_{12})$ is the finite BEDC concatenation of those two ledgers. The transition provenance ledgers concatenate in the same order: rows internal to P_1 and P_2 are inherited, and the single boundary row is the transported splice row just constructed. The carrier definition asks for no host stochastic process, filtration, conditional expectation, or standard kernel bridge; it asks only for these displayed random-variable rows, law rows, transition rows, and $\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}$ provenance. Hence the assembled packet satisfies the accepted $\mathsf{MarkovChain}^{\uparrow}$ carrier predicate over the unary ledger T_{12} of length $n + m$. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket splice transport closure`

Theorem 189.14 (MarkovChain filtered transition consumer catalogue)**.** *Every downstream finite stochastic-process consumer of an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet reads only the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source, unary time ledger, adjacent* $\mathsf{RandomVar}^{\uparrow}$ *rows,* $\mathsf{Distribution}^{\uparrow}$ *law rows, transition rows, finite-prefix restrictions, concatenation and splice rows,* hsame, Cont, *and* Pkg *provenance.*

The catalogue excludes host stochastic processes, filtrations, conditional expectation, stationary laws, martingale structure, and standard kernel bridges. A consumer may restrict to a finite window, concatenate compatible windows, or read across a transported splice, but every such read remains inside the displayed finite transition packet.

Proof. Project the accepted packet through Definition 189.1. The projection exposes only the $\mathsf{ProbSpace}^{\uparrow}$ source, the unary time ledger, adjacent $\mathsf{RandomVar}^{\uparrow}$ endpoints, $\mathsf{Distribution}^{\uparrow}$ law rows, transition rows, and the displayed hsame/Cont/Pkg provenance.

For local finite windows, apply Definition 189.12. The restriction keeps the same source row and copies only the visible endpoint, law, transition, and provenance rows whose indices lie in the chosen finite prefix.

For consumers that splice or extend windows, apply Definition 189.13 together with the adjacent transport row recorded after that theorem. These results rebuild the concatenated packet from the two displayed finite ledgers and the single transported splice row.

The semantic certificate row for $\mathsf{MarkovChain}^{\uparrow}$ is then the public boundary: it certifies that the visible carrier, classifier, stability, exactness, and provenance fields are the rows available to downstream consumers. No host stochastic process, filtration, conditional expectation, stationary law, martingale datum, or standard kernel bridge is a field of that packet. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainUp concrete to schema`

Definition 189.15 (MarkovChain finite-suffix restriction)**.** *Let P be a displayed* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet over a unary time ledger with transition-time rows $0, \ldots, n+k-1$ and endpoint rows $0, \ldots, n+k$. The finite-suffix restriction $P^k_{\geq n}$ is the packet obtained by keeping the same* $\mathsf{ProbSpace}^{\uparrow}$ *source, replacing the initial endpoint by the boundary row X_n, retaining the endpoint rows*

$$X_n, X_{n+1}, \ldots, X_{n+k},$$

retaining the law rows read at those endpoints, retaining the transition rows

$$K_n, K_{n+1}, \ldots, K_{n+k-1},$$

and retaining exactly the hsame/Cont/Pkg *provenance entries whose adjacent endpoints both lie in that displayed suffix.*

The suffix ledger is the unary ledger of length k whose endpoint row j is read as the original endpoint row $n+j$ and whose transition row $j < k$ is read as the original transition row $n+j$. No transition row before n is copied into $P^k_{\geq n}$.

Theorem 189.16 (MarkovChain finite-suffix restriction carrier)**.** *Let P be an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *transition packet in the sense of Definition 189.1. Suppose its unary time ledger has length $n+k$, with endpoint rows $X_0, \ldots, X_{n+k}$ and transition rows $K_0, \ldots, K_{n+k-1}$. Then the finite-suffix restriction $P^k_{\geq n}$ of Definition 189.15, consisting of the same* $\mathsf{ProbSpace}^{\uparrow}$ *source, the displayed endpoint rows $X_n, \ldots, X_{n+k}$, the corresponding* $\mathsf{Distribution}^{\uparrow}$ *law rows, the transition rows $K_n, \ldots, K_{n+k-1}$, and the inherited adjacent* hsame/Cont/Pkg *provenance, is again an accepted* $\mathsf{MarkovChain}^{\uparrow}$ *carrier.*

Proof. The carrier predicate of Definition 189.1 is local along adjacent transition rows. It requires a finite unary ledger, carried $\mathsf{RandomVar}^{\uparrow}$ endpoint rows, law rows supplied by $\mathsf{Distribution}^{\uparrow}$, transition rows, and for each adjacent pair a displayed Cont ledger and Pkg provenance row witnessing that the next-step distribution is read from the current-state endpoint through the displayed transition row.

The suffix ledger is finite because it is the displayed block of k successive transitions beginning at the boundary endpoint X_n. Its endpoint row j is X_{n+j} for $0 \le j \le k$, and its transition row $j < k$ is K_{n+j}. Each such endpoint and transition row is already a row of the accepted packet P, so the restricted packet obtains its $\mathsf{RandomVar}^{\uparrow}$ and transition data by projection from P rather than by introducing any outside stochastic-process object.

The boundary condition at the first suffix time is the only place where a prefix argument does not directly apply. In $P^k_{\ge n}$ the initial row is X_n, not X_0. This causes no additional obligation: the MarkovChain carrier does not require a distinguished initial law supplied by the original start row. It requires only that every displayed transition row have its adjacent source and target endpoints and its next-law row. For the first suffix transition, acceptance of P at the original index n supplies exactly the row

$$X_n \xrightarrow{K_n} X_{n+1}$$

with the displayed $\mathsf{Cont}/\mathsf{Pkg}$ provenance saying that the next-step law of X_{n+1} is read from the current-state endpoint of X_n through K_n. Thus the suffix packet has the required first adjacent transition ledger.

For any remaining suffix transition $j < k$, the corresponding original index is $n + j < n + k$. Acceptance of P supplies the transition row K_{n+j}, the adjacent endpoints X_{n+j} and X_{n+j+1}, the law row for the target endpoint, and the provenance witnessing the same current-state to next-law read. These are precisely the rows displayed at suffix positions j and $j + 1$, so the original provenance is well-typed after the suffix reindexing.

Every transition obligation of $P^k_{\ge n}$ is obtained from exactly one transition obligation of P with index between n and $n + k - 1$, and the suffix construction creates no additional transition. The source boundary of Definition 189.5 is preserved because the restricted packet keeps the same $\mathsf{ProbSpace}^{\uparrow}$ source and only discards rows outside the suffix. Hence all rows required by the accepted $\mathsf{MarkovChain}^{\uparrow}$ transition carrier predicate are present in $P^k_{\ge n}$. □

Lean4 ✓ `BEDC.Derived.MarkovChainUp.MarkovChainTransitionPacket finite suffix carrier`

Chapter 190

A Concrete Naming Certificate for $\mathsf{Martingale}^{\uparrow}$

The martingale interface $\mathsf{Martingale}^{\uparrow}$ packages an adapted sequence of integrable random variables whose conditional expectation at each step is the previous value. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ (chapter 188).

190.1 Carrier definition

Definition 190.1 (Adapted BHist carrier for MartingaleUp)**.** *A* $\mathsf{Martingale}^{\uparrow}$ *packet is a finite* BHist *row package over a displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source. It consists of:*

1. *a finite* UnaryHistory *time spine* T *whose entries are histories* $t : \mathsf{BHist}$*;*
2. *for each displayed time* $t \in T$*, a carried* $\mathsf{RandomVar}^{\uparrow}$ *row* $X_t : S \to R$ *with integrable* $\mathsf{Real}^{\uparrow}$ *endpoint history;*
3. *for each successor display* $t \rightsquigarrow t^+$ *in* T*, a carried* $\mathsf{CondExp}^{\uparrow}$ *row* E_t *whose source random-variable row is* X_{t^+} *and whose conditioning visibility row is the filtration row* $\mathcal{F}_t$*;*
4. *filtration visibility rows* $\mathcal{F}_t \hookrightarrow \mathcal{F}_{t^+}$ *saying that the readable events and the carried random-variable endpoint* X_t *are visible at time* t*;*
5. *a finite* Cont *ledger connecting the time row, the random-variable rows, the conditional-expectation rows, and the filtration visibility rows inside one shared* Pkg *provenance row.*

The displayed martingale row of the packet is the conditional-expectation comparison

$$E_t(X_{t^+} \mid \mathcal{F}_t) \sim_R X_t$$

for every successor display $t \rightsquigarrow t^+$ *in the finite time spine, where* $\sim_R$ *is the* $\mathsf{Real}^{\uparrow}$ *classifier read through the carried conditional-expectation endpoint.*

Definition 190.2 (Conditional-expectation classifier for MartingaleUp)**.** *Let* M *and* N *be two* $\mathsf{Martingale}^{\uparrow}$ *packets as in Definition 190.1. A conditional-expectation classifier from* M *to* N *consists of:*

1. hsame *transport between the two finite* UnaryHistory *time spines, preserving successor displays;*
2. *componentwise* hsame *transport between the matched* $\mathsf{RandomVar}^{\uparrow}$ *rows* X_t *and* Y_u*;*

3. *componentwise* hsame *transport between the matched* $\mathsf{CondExp}^{\uparrow}$ *rows* E_t *and* F_u*;*
4. *transport of filtration visibility rows along the matched time and random-variable histories;*
5. *a shared* Pkg *provenance comparison, with the finite* Cont *ledger of* M *transported to the finite* Cont *ledger of* N*.*

The classifier holds exactly when these transported rows identify the displayed conditional-expectation comparison of M *with the displayed conditional-expectation comparison of* N *at every matched successor display.*

Definition 190.3 (MartingaleUp adapted BHist stream carrier)**.** *Fix a displayed finite horizon* $N : \mathbb{N}$*. A finite adapted* $\mathsf{Martingale}^{\uparrow}$ *carrier packet is a BEDC row family indexed by unary histories* $\nu_0, \ldots, \nu_N : \mathsf{BHist}$*. For every* $n \leq N$ *the packet contains a visible value endpoint* $X_n : \mathsf{BHist}$ *and a* $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ *row witnessing that* X_n *is a carried* $\mathsf{RandomVar}^{\uparrow}$ *endpoint over the displayed source and target carriers. It also contains a filtration/provenance ledger row*

$$\mathsf{Fil}_n(\nu_n, X_n, \Gamma_n)$$

whose entries live only in BHist *and* Pkg *and record the displayed observable history* ν_n*, the value endpoint* X_n*, and the certificate package* Γ_n *consumed at that time.*

For each successor index $n < N$ *the packet contains displayed endpoints*

$$X_n, \quad X_{n+1}, \quad E_{n+1|n}, \quad R_{n+1|n} : \mathsf{BHist}$$

together with the continuation row $\mathsf{Cont}(E_{n+1|n}, R_{n+1|n}, X_n)$*. The endpoint* $E_{n+1|n}$ *is the named conditional-expectation endpoint supplied by the successor* $\mathsf{CondExp}^{\uparrow}$ *row, while* $R_{n+1|n}$ *is the residual history carried by that same row.*

Definition 190.4 (MartingaleUp conditional expectation step classifier)**.** *Two finite adapted* $\mathsf{Martingale}^{\uparrow}$ *carrier packets over the same displayed horizon are classifier-equal when their components match by the BEDC rows visible in Definition 190.3. For every index* $n \leq N$ *the observable histories, value endpoints, and provenance packages satisfy*

$$\mathsf{hsame}(\nu_n, \nu'_n), \qquad \mathsf{hsame}(X_n, X'_n), \qquad \mathsf{hsame}(\Gamma_n, \Gamma'_n).$$

The value-endpoint comparison is transported through the corresponding $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ *rows, so the classifier does not read a host-level sequence equality.*

For every successor $n < N$*, the step endpoints satisfy*

$$\mathsf{hsame}(E_{n+1|n}, E'_{n+1|n}), \qquad \mathsf{hsame}(R_{n+1|n}, R'_{n+1|n}),$$

and the displayed $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ *rows identify the transported conditional-expectation endpoint* $E_{n+1|n}$ *with the previous value endpoint* X_n *through the continuation row* $\mathsf{Cont}(E_{n+1|n}, R_{n+1|n}, X_n)$*. These componentwise comparisons are the full* $\mathsf{Martingale}^{\uparrow}$ *classifier.*

Definition 190.5 (MartingaleUp BHist adapted-sequence packet)**.** *A* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet is a finite BEDC tuple*

$$\mathcal{M} = (R, C, F, X, E, T, \rho, \lambda)$$

whose entries have the following roles. The row R *is the consumed* $\mathsf{RandomVar}^{\uparrow}$ *packet and the row* C *is the consumed* $\mathsf{CondExp}^{\uparrow}$ *packet. The row* F *is the displayed filtration and adaptation row. The row* X *is a finite sequence of* BHist *endpoint rows, each endpoint being a* $\mathsf{RandomVar}^{\uparrow}$ *row scoped*

by R and adapted to the displayed filtration row F. The row E is a finite sequence of $\mathsf{CondExp}^{\uparrow}$ *rows, one for each adjacent time face, whose source is the later endpoint row and whose target is the previous endpoint row. The row T is the tower-law boundary row recording compatibility of adjacent conditional-expectation faces. The row ρ records package provenance through the two consumed source packets, and λ is the finite* Cont *ledger that stores all displayed transport rows.*

The packet is accepted as a $\mathsf{Martingale}^{\uparrow}$ *carrier exactly when every row just named is present in this finite tuple, the X rows are random-variable endpoint rows over R, the E rows are conditional-expectation rows over C, and no carrier datum outside R, C, F, X, E, T, ρ, and λ is used to state the martingale condition.*

Definition 190.6 (MartingaleUp conditional-expectation ledger classifier)**.** *For two adapted-sequence packets*

$$\mathcal{M} = (R, C, F, X, E, T, \rho, \lambda) \quad \textit{and} \quad \mathcal{M}' = (R', C', F', X', E', T', \rho', \lambda'),$$

the $\mathsf{Martingale}^{\uparrow}$ *ledger classifier holds when the consumed* $\mathsf{RandomVar}^{\uparrow}$ *packets R and R' are related by the* $\mathsf{RandomVar}^{\uparrow}$ *classifier, the consumed* $\mathsf{CondExp}^{\uparrow}$ *packets C and C' are related by the* $\mathsf{CondExp}^{\uparrow}$ *classifier, and the remaining displayed rows are related componentwise by* hsame*:*

$$F\,\mathsf{hsame}\,F', \quad X\,\mathsf{hsame}\,X', \quad E\,\mathsf{hsame}\,E', \quad T\,\mathsf{hsame}\,T', \quad \rho\,\mathsf{hsame}\,\rho', \quad \lambda\,\mathsf{hsame}\,\lambda'.$$

Thus the classifier transports only the visible filtration, endpoint-sequence, conditional-expectation, tower-boundary, provenance, and ledger rows, while the two consumed dependency packets supply their own internal carrier and classifier obligations.

190.2 The certificate

Theorem 190.7 (Martingale filtration ledger exactness)**.** *For a* $\mathsf{Martingale}^{\uparrow}$ *packet of Definition 190.1, the* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *ledger exposes exactly the adaptedness rows, filtration visibility rows,* $\mathsf{RandomVar}^{\uparrow}$ *rows,* $\mathsf{CondExp}^{\uparrow}$ *rows,* $\mathsf{UnaryHistory}$ *time-index rows, shared* Pkg *provenance row, and finite* Cont *ledger listed in the carrier. The ledger is exact precisely when each successor display $t \rightsquigarrow t^{+}$ reads the displayed row*

$$E_t(X_{t^+} \mid \mathcal{F}_t) \sim_R X_t$$

from those entries and from no additional host equality, host function equality, or unlisted measurability field.

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket filtration ledger exactness`

Proof. Project the carrier packet of Definition 190.1. The time spine supplies only $\mathsf{UnaryHistory}$ indices and successor displays. The random-variable spine supplies the carried $\mathsf{RandomVar}^{\uparrow}$ rows, and the filtration part supplies exactly the visibility rows that make X_t readable at time t.

The conditional-expectation entry at a successor display is already a carried $\mathsf{CondExp}^{\uparrow}$ row over X_{t^+} and $\mathcal{F}_t$. Reading its endpoint through the shared $\mathsf{Real}^{\uparrow}$ classifier gives the displayed comparison with X_t.

The finite Cont ledger records how these rows sit in one Pkg provenance row. Since every value consumed by $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ is one of the projected carrier entries, and the theorem excludes every unlisted host field, the ledger is exact for the displayed martingale surface. □

Theorem 190.8 (Martingale conditional-expectation stability)**.** *Let M and N be* $\mathsf{Martingale}^{\uparrow}$ *packets related by the classifier of Definition 190.2. If M carries the displayed conditional-expectation comparison*

$$E_t(X_{t^+} \mid \mathcal{F}_t) \sim_R X_t$$

at a successor display $t \rightsquigarrow t^+$, then N carries the transported comparison

$$F_u(Y_{u^+} \mid \mathcal{G}_u) \sim_R Y_u$$

at the matched successor display $u \rightsquigarrow u^+$. Hence classifier-related packets preserve every displayed martingale equality row.

Proof. Use the time-spine clause of the classifier to transport the successor display $t \rightsquigarrow t^+$ to $u \rightsquigarrow u^+$. The same clause transports the filtration visibility row $\mathcal{F}_t$ to $\mathcal{G}_u$.

Transport the random-variable rows componentwise by hsame, sending X_t to Y_u and X_{t^+} to Y_{u^+}. Transport the conditional-expectation row componentwise by hsame, sending E_t to F_u over the matched source random variable and matched filtration row.

Finally use the shared Pkg provenance comparison and the transported finite Cont ledger. These rows identify the displayed conditional-expectation comparison in M with the displayed comparison in N, so the $\mathsf{Real}^{\uparrow}$ classifier row for M transports to the corresponding row for N. □

Theorem 190.9 (Martingale random-variable boundary)**.** *In every finite adapted carrier packet of Definition 190.3, each visible value row X_n is supplied by the displayed* $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ *component Γ_n. The* $\mathsf{Martingale}^{\uparrow}$ *packet never reads a host-level sequence or a host equality of values; all value comparisons are mediated by* hsame *and by the transported* $\mathsf{RandomVar}^{\uparrow}$ *classifier row recorded in Γ_n.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket randomvar boundary`

Proof. Fix an index $n \leq N$. By definition of the carrier packet, the value endpoint X_n appears only together with its package Γ_n and the ledger row $\mathsf{Fil}_n(\nu_n, X_n, \Gamma_n)$.

Unpack Γ_n through the public $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ surface of section 185.2. The measurable-preimage and endpoint classifier data used by the value row are exactly the rows carried by that certificate.

No other field of Definition 190.3 contains a function from natural numbers into host values. The only index is the displayed unary history label, and the only equality test is the componentwise hsame comparison of Definition 190.4. Thus every visible value row is supplied by $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ and no host sequence is read. □

Theorem 190.10 (Martingale conditional expectation step law)**.** *For every successor index $n < N$ in a finite adapted* $\mathsf{Martingale}^{\uparrow}$ *carrier packet, the row from X_{n+1} to X_n consumes a displayed* $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ *component whose conditional-expectation endpoint $E_{n+1|n}$ is identified with the previous value endpoint X_n. The identification is recorded inside the BEDC packet by the continuation row* $\mathsf{Cont}(E_{n+1|n}, R_{n+1|n}, X_n)$ *and by the* $\mathsf{CondExp}^{\uparrow}$ *classifier, not by an ambient analytic equality.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket condexp step law`

Proof. Fix a successor index $n < N$. The carrier packet supplies the displayed endpoints X_n, X_{n+1}, $E_{n+1|n}$, and $R_{n+1|n}$, together with the successor $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ row.

Read that row through Definition 188.1. Its visible fields are the random-variable total-readback component, the projected conditional-expectation endpoint, and the continuation ledger. In the martingale packet these fields are instantiated by X_{n+1}, $E_{n+1|n}$, $R_{n+1|n}$, and X_n.

Apply the classifier obligation of Definition 188.2. It transports the displayed endpoint comparison through hsame and the continuation row. Therefore the successor row states exactly that the conditional expectation of the next value endpoint over the displayed filtration history is classified as the previous endpoint X_n. □

Theorem 190.11 (MartingaleUp obligation closure package)**.** *The public* $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ *obligation set consists exactly of the finite time spine, the RandomVarUp endpoint rows, the CondExpUp successor rows, the filtration visibility rows, the conditional-expectation comparison rows, the shared* Pkg *provenance row, and the finite* Cont *ledger. These rows are precisely the data carried by Definition 190.3 and classified by Definition 190.4. The package exports no host sequence equality, ambient analytic equality, unlisted measurability field, or bridge status.*

Proof. Start with the adapted carrier. The definition supplies the finite time spine, the displayed value endpoints, the RandomVarUp endpoint rows, the CondExpUp successor rows, the filtration/provenance ledger rows, and the finite continuation rows.

Next attach the classifier. By Definition 190.4, comparison between two packets is only componentwise hsame transport of the finite time spine, RandomVarUp rows, CondExpUp rows, filtration visibility rows, provenance, and Cont ledger.

Use Definition 190.12 and Definition 190.9 to identify the public endpoint surface. The former packages the carrier and classifier rows as the public certificate family, while the latter says every visible value endpoint comes from the consumed RandomVarUp rows.

Finally apply Definition 190.10. It identifies each successor comparison row with the displayed CondExpUp endpoint comparison and the continuation row ending at the previous value endpoint. Combining these rows gives the listed obligation set exactly, so no host sequence equality, ambient analytic equality, unlisted measurability field, or bridge status is exported. □

Theorem 190.12 (Martingale public certificate row family)**.** *The finite adapted carrier packet of Definition 190.3, the classifier of Definition 190.4, the random-variable boundary of Definition 190.9, and the successor law of Definition 190.10 form the public* $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ *obligation family. Its exported rows are precisely the carrier rows, componentwise classifier transport, stability under* hsame *replacement, filtration/provenance ledger rows, conditional expectation exactness rows, and status rows naming the finite displayed horizon.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket public certificate row family`

Proof. Start with the carrier rows. They expose a finite family of displayed BHist endpoints, Pkg provenance entries, and $\mathsf{NameCert}_{\mathsf{RandomVar}^\uparrow}$ value packages, so the public object has a concrete BEDC carrier.

Next attach the classifier. By Definition 190.4, equality is componentwise hsame plus the transported $\mathsf{RandomVar}^\uparrow$ and $\mathsf{CondExp}^\uparrow$ classifier rows. This gives stability under replacement by displayed classifier witnesses.

The random-variable boundary theorem supplies the exact source of every visible value row. The successor law supplies the exact conditional-expectation row for every adjacent pair. Repacking these rows with the filtration/provenance ledger gives the public $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ family and leaves no unrecorded host sequence, ambient conditional expectation, or external status field. □

Theorem 190.13 (Martingale adapted-sequence carrier boundary)**.** *Carrier acceptance for a* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet exposes exactly the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint sequence rows adapted to the displayed filtration. In particular, every public endpoint row of an accepted packet is one of the rows listed in* X*, is scoped through the consumed* $\mathsf{RandomVar}^{\uparrow}$ *packet* R*, and is connected to the displayed filtration row* F *by the adaptation row already present in the packet.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket adapted sequence carrier boundary`

Proof. Unpack the accepted packet of Definition 190.5. The only endpoint component of the carrier is the finite row X, and the definition requires each entry of X to be a $\mathsf{RandomVar}^{\uparrow}$ endpoint row scoped by R.

The displayed filtration row is F. Since acceptance requires adaptation of the X rows to F, projecting the carrier gives the desired boundary without adding a further classifier, external sigma-algebra carrier, or host-level stochastic-process field. □

Theorem 190.14 (Martingale conditional-expectation ledger stability)**.** *If two accepted* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packets are related by the classifier of Definition 190.6, then the displayed conditional-expectation rows are preserved through the consumed* $\mathsf{CondExp}^{\uparrow}$ *classifier and through the finite* Cont *ledger. Consequently each adjacent martingale face recorded in* E *transports to the corresponding face in* E' *with the same previous-endpoint boundary.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket condexp ledger stability`

Proof. First use the $\mathsf{CondExp}^{\uparrow}$ classifier clause to transport the conditional-expectation source and target rows. This supplies the stability of the expectation faces themselves, including the row whose target is the previous endpoint.

Next use the hsame comparisons for X, E, and λ. The X comparison transports endpoint names, the E comparison identifies the displayed expectation rows, and the Cont ledger comparison records that the same finite transport boundary is being read in the two packets. Repacking these witnesses gives the stated adjacent-face stability. □

Theorem 190.15 (Martingale tower-law ledger surface)**.** *The tower-law row* T *of a* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet is a finite ledger boundary over adjacent conditional-expectation faces. It records only that composing two neighboring rows of* E *through the displayed* Cont *ledger has the same* hsame *boundary as the single conditional-expectation row supplied by the consumed* $\mathsf{CondExp}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket tower law ledger surface`

Proof. Read T as a boundary row, not as an analytic identity outside the packet. Its source data are two adjacent expectation faces from E, their endpoint rows from X, and the corresponding finite transport rows from λ.

The consumed $\mathsf{CondExp}^{\uparrow}$ packet supplies the conditional-expectation face classifier. The Cont ledger supplies composition of the two adjacent transports, and hsame records the equality of the composed boundary with the displayed single face. Therefore the tower-law surface is finite and ledger-scoped. □

Theorem 190.16 (Martingale NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *obligation surface is the package consisting of the adapted-sequence carrier of Definition 190.5, the ledger classifier of Definition 190.6, stability of the conditional-expectation rows, the tower-law ledger*

boundary, exactness of the displayed carrier rows, and the downstream consumer row used by $\mathsf{Brownian}^{\uparrow}$ *through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket namecert obligation surface`

Proof. The carrier and classifier clauses are exactly the two definitions in section 190.1. They name the finite packet and the componentwise transport relation over $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, BHist, hsame, and Cont.

Stability is supplied by Definition 190.14, and the tower-law ledger is supplied by Definition 190.15. Exactness follows by reading the accepted carrier backwards: every public martingale row must be one of R, C, F, X, E, T, ρ, or λ, or a row already scoped by the two consumed source certificates.

The Brownian-motion consumer uses only the resulting $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ row. Since that row is assembled from the finite carrier, classifier, stability, ledger, and exactness clauses just listed, the downstream dependency is a concrete BEDC certificate surface rather than an unstated martingale carrier assumption. □

Theorem 190.17 (Martingale finite-prefix restriction)**.** *Let*

$$\mathcal{M} = (R, C, F, X, E, T, \rho, \lambda)$$

be an accepted $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet in the sense of Definition 190.5. Restrict* $\mathcal{M}$ *to an initial finite unary-history prefix of its displayed time spine, keeping the same consumed* $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ *row* R *and* $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ *row* C*, the prefix-indexed rows of* X*,* E*,* T*,* ρ*, and* λ*, and the corresponding restricted* Cont *ledger. The resulting truncated tuple is again an accepted* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet.*

Lean4 ✓ `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket finite prefix restriction`

Proof. Project the rows named in Definition 190.5. The accepted packet contains exactly the consumed R and C certificates, the displayed filtration row F, the endpoint sequence X, the adjacent conditional-expectation rows E, the tower boundary T, the provenance row ρ, and the finite continuation ledger λ.

Restrict those displayed finite rows to the chosen initial time prefix. The consumed R and C rows are unchanged, while the retained X, E, T, ρ, and λ entries are precisely the rows whose indices remain in the prefix. The restricted Cont ledger is the subledger carried by those retained adjacent faces.

For each transported adjacent face, use Definition 190.14. It keeps the conditional-expectation row compatible with the consumed $\mathsf{CondExp}^{\uparrow}$ classifier and with the finite Cont ledger after the prefix restriction.

Finally repack the retained carrier, classifier, stability, ledger, and exactness rows through Definition 190.16. The truncated tuple therefore satisfies the same $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ obligation surface as the original finite packet, with no row outside the displayed prefix added. □

Theorem 190.18 (Martingale Brownian consumer projection scope)**.** *Any* $\mathsf{Brownian}^{\uparrow}$ *consumer reading the* $\mathsf{Martingale}^{\uparrow}$ *dependency row may access only the adapted-sequence carrier of Definition 190.5, the conditional-expectation ledger classifier of Definition 190.6, the finite endpoint sequence rows, the tower-law ledger, the shared* Pkg *provenance, and the finite* $\mathsf{Cont}/\mathsf{hsame}$ *transport rows named by Definition 190.16. It has no projection to host stochastic-process data, stopping-time data, convergence data, or standard bridge data through the* $\mathsf{Martingale}^{\uparrow}$ *row.*

Proof. First read the public row family from Definition 190.12. It exposes the adapted-sequence carrier, componentwise classifier transport, stability rows, filtration/provenance rows, conditional-expectation exactness rows, and the finite displayed horizon as the public $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ surface.

Next apply Definition 190.16. That theorem identifies the same public surface with the adapted-sequence packet, the conditional-expectation ledger classifier, the finite endpoint sequence rows, the tower-law boundary, the shared Pkg provenance, and the finite Cont/hsame transport rows.

Finally compare with the Brownian displayed dependency surface Definition 191.3. The Brownian packet reads its martingale input only through the consumed $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ row M, so composing the two Martingale projections with that Brownian projection gives exactly the listed consumer scope. Since neither the Martingale packet nor the Brownian dependency surface carries a host stochastic-process row, stopping-time row, convergence row, or standard bridge row, none of those data can be exported through this consumer projection. □

Closure status for $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$**.**

Constructive story: A martingale packet is assembled over a finite unary time spine by placing a carried $\mathsf{RandomVar}^\uparrow$ endpoint X_n at each displayed time, a carried $\mathsf{CondExp}^\uparrow$ row on each successor face, filtration visibility rows, shared Pkg provenance, and a finite Cont/hsame ledger into one adapted-sequence tuple. The carrier accepts exactly those tuples whose random-variable rows, conditional-expectation rows, tower-boundary row, provenance, and finite transport ledger are all present, and the classifier compares two packets componentwise through the consumed $\mathsf{RandomVar}^\uparrow$ and $\mathsf{CondExp}^\uparrow$ classifiers together with history-sameness on the remaining displayed rows. The source therefore covers finite adapted endpoint sequences over a displayed probability source, the pattern is the successor comparison $E_{n+1|n} \sim_R X_n$ plus filtration visibility and tower-boundary bookkeeping, and the ledger keeps the finite time row, endpoint sequence, conditional-expectation sequence, provenance, and transport rows visible. Because each dependency certificate already supplies classifier transport and the finite tuple classifier is componentwise, an admitted adapted sequence gives the carrier witness, the classifier is reflexive, symmetric, and transitive, and prefix restriction or row transport stays inside the carrier. A finite packet h lies in the signature gap of a $\mathsf{Martingale}^\uparrow$ token p when the source-domain accepts its time spine, endpoint rows, conditional-expectation faces, filtration rows, and finite ledger, the signature relation pairs that packet with an answer packet and evidence ledger, and token introduction names the answer as p; unfolded, every successor face says that the conditional expectation of the next endpoint is classified with the previous endpoint in the ledger encoded by p. There is no standard stochastic-process bridge at this closure level, so downstream Brownian consumers may read only the displayed packet rows. The chapter registers this finite adapted-sequence construction under the name $\mathsf{Martingale}^\uparrow$; the martingale law is a finite CondExpUp ledger over RandomVarUp endpoints, with no stopping-time, convergence, or host process primitive introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$, maturePackageClosed)

Scope closed: The finite adapted-sequence packet, conditional-expectation step law, RandomVarUp boundary, ledger stability, and public row family bind the finite time spine, endpoint rows, CondExpUp faces, filtration visibility rows, shared Pkg provenance, and finite Cont ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.MartingaleUp.MartingaleAdaptedSequencePacket\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: It does not close Brownian sample-path theory, measure-theoretic convergence, stopping-time results, or standard stochastic-process bridges.

Upgrade path: Mature closure requires examples and consumer coverage for the same finite adapted-sequence packet surface.

Definition 190.19 (Martingale finite-suffix restriction)**.** *Let*

$$\mathcal{M} = (R, C, F, X, E, T, \rho, \lambda)$$

be a displayed $\mathsf{Martingale}^\uparrow$ *adapted-sequence packet whose finite unary time spine has endpoint rows* $0, \ldots, n+k$*. The finite-suffix restriction* $\mathcal{M}^k_{\geq n}$ *is the tuple obtained by keeping the same consumed* $\mathsf{NameCert}_{\mathsf{RandomVar}^\uparrow}$ *row* R*, the same consumed* $\mathsf{NameCert}_{\mathsf{CondExp}^\uparrow}$ *row* C*, and the same displayed filtration and adaptation row* F*, while retaining only the suffix-indexed endpoint rows*

$$X_n, X_{n+1}, \ldots, X_{n+k},$$

the adjacent conditional-expectation rows

$$E_{n+1|n}, E_{n+2|n+1}, \ldots, E_{n+k|n+k-1},$$

the tower-boundary rows of T whose adjacent conditional-expectation faces both lie in that suffix, the corresponding provenance rows of ρ, and exactly the finite Cont *ledger entries of λ whose endpoints and conditional-expectation faces lie in the displayed suffix.*

The suffix time row is reindexed by unary histories so that suffix endpoint j is the original endpoint $n+j$ and suffix adjacent face $j<k$ is the original face $n+j+1|n+j$. No endpoint row before n, no conditional-expectation face before $n+1|n$, and no tower or ledger row crossing the boundary from $n-1$ to n is part of $\mathcal{M}^k_{\geq n}$.

Theorem 190.20 (Martingale finite-suffix restriction carrier)**.** *Let*

$$\mathcal{M} = (R, C, F, X, E, T, \rho, \lambda)$$

be an accepted Martingale$^{\uparrow}$ *adapted-sequence packet in the sense of Definition 190.5. Suppose its displayed finite unary time spine has endpoint rows $X_0, \ldots, X_{n+k}$ and adjacent conditional-expectation rows $E_{1|0}, \ldots, E_{n+k|n+k-1}$. Then the finite-suffix restriction $\mathcal{M}^k_{\geq n}$ of Definition 190.19, with the same consumed* RandomVar$^{\uparrow}$ *and* CondExp$^{\uparrow}$ *packets, the same displayed filtration and adaptation row, the suffix endpoint rows $X_n, \ldots, X_{n+k}$, the suffix conditional-expectation rows $E_{n+1|n}, \ldots, E_{n+k|n+k-1}$, and the inherited suffix rows of T, ρ, and λ, is again an accepted* Martingale$^{\uparrow}$ *adapted-sequence packet.*

Proof. Unpack the accepted packet by Definition 190.5. Acceptance is stated entirely by the presence of the finite tuple rows R, C, F, X, E, T, ρ, and λ: the X entries are RandomVar$^{\uparrow}$ endpoint rows over R adapted to F, the E entries are CondExp$^{\uparrow}$ rows over C whose source is the later retained endpoint and whose target is the earlier retained endpoint, T records tower compatibility of adjacent faces, ρ records provenance through the two consumed packets, and λ stores the displayed transport rows.

Form the suffix tuple. The consumed rows R and C are unchanged, so the source certificates for every retained RandomVar$^{\uparrow}$ endpoint and every retained CondExp$^{\uparrow}$ face are the same rows used in the parent packet. The filtration row F is also unchanged, and the suffix reads only the adaptation entries of F attached to the retained endpoints $X_n, \ldots, X_{n+k}$. By Definition 190.13, each retained public endpoint row is one of the original entries of X, is scoped through R, and is connected to the displayed filtration row by an adaptation row already present in the accepted packet.

Consider a suffix adjacent face $j<k$. Under the suffix reindexing it is the original face $n+j+1|n+j$. Acceptance of $\mathcal{M}$ supplies the conditional-expectation row over C whose source is X_{n+j+1} and whose target is X_{n+j}, together with the corresponding continuation ledger entry. By Definition 190.14, the displayed conditional-expectation row is preserved through the consumed CondExp$^{\uparrow}$ classifier and through the finite Cont ledger. Thus the same row is well typed as the suffix face from suffix endpoint $j+1$ to suffix endpoint j.

The first suffix face is not a separate initial-condition obligation. It is exactly the original face $E_{n+1|n}$, with boundary endpoint X_n as its target. The carrier definition does not require the suffix to remember the discarded predecessor X_{n-1} or any conditional-expectation row crossing into X_n from outside the suffix. It requires only the displayed endpoint rows, the adjacent conditional-expectation faces internal to the finite spine, and the finite ledger rows attached to those faces.

For tower compatibility, retain precisely those rows of T whose adjacent conditional-expectation faces both survive the suffix restriction. By Definition 190.15, each such row is only a finite ledger

boundary over adjacent faces from E and the corresponding rows of λ. Since the suffix retains both faces and the associated Cont entries, each retained tower-boundary row remains a valid tower row in the suffix packet. Rows of T crossing the discarded boundary are not obligations of the suffix because one of their adjacent faces is no longer displayed.

The provenance row ρ and ledger λ are finite rows, so restricting them to entries whose endpoints and faces lie in the suffix preserves exactly the displayed provenance and transport required by the retained X, E, and T rows. No row outside R, C, F, the suffix of X, the suffix of E, the retained tower rows, the retained provenance rows, and the retained ledger rows is introduced.

Repack the retained rows through Definition 190.16. The suffix tuple has the same form as the accepted adapted-sequence packet: consumed $\mathsf{RandomVar}^{\uparrow}$ and $\mathsf{CondExp}^{\uparrow}$ packets, displayed filtration and adaptation rows, finite endpoint rows, finite adjacent conditional-expectation faces, finite tower-boundary rows, provenance, and finite Cont ledger. Therefore $\mathcal{M}^{k}_{\geq n}$ is again an accepted $\mathsf{Martingale}^{\uparrow}$ adapted-sequence packet. $\square$

Chapter 191

A Concrete Naming Certificate for $\mathsf{Brownian}^{\uparrow}$

The Brownian-motion interface $\mathsf{Brownian}^{\uparrow}$ packages a continuous-time stochastic process with independent normally distributed increments and continuous sample paths. It depends on $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ (chapter 190) and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ (chapter 130).

191.1 Carrier definition

Definition 191.1 (Brownian BHist process packet)**.** *A* $\mathsf{Brownian}^{\uparrow}$ *process packet is a finite displayed* BHist *tuple*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda).$$

The row M is the consumed $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *dependency row for the adapted value process. The row C is the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *path row. The row T is a finite* UnaryHistory *time-index row. The row I is the step ledger recording the visible adjacent-time step faces over T. The row N is the normal-step display attached to those adjacent faces. The row P is the path-continuity display attached to the path row C. The row ρ is the shared* Pkg *provenance tying M, C, T, I, N, and P into one displayed packet, and λ is the finite* Cont/hsame *transport ledger for every row read by the classifier.*

The packet is admitted as a $\mathsf{Brownian}^{\uparrow}$ *carrier exactly when M supplies all adaptedness and martingale rows through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$*, C supplies all path-continuity rows through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, and the Gaussian-step, time-index, provenance, and transport rows are displayed inside this finite tuple. No host probability space, host stochastic-process object, or host path equality is part of the carrier.*

Definition 191.2 (Brownian step-continuity classifier)**.** *Let*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda) \quad \text{and} \quad \mathcal{B}' = (M', C', T', I', N', P', \rho', \lambda')$$

be two $\mathsf{Brownian}^{\uparrow}$ *process packets. The* $\mathsf{Brownian}^{\uparrow}$ *classifier holds exactly when the six displayed rows M, C, T, I, N, and P are compared rowwise by* hsame*, with the M comparison read through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *and the C comparison read through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*:*

$$\begin{aligned} &M\mathsf{hsame}M' \wedge C\mathsf{hsame}C' \wedge T\mathsf{hsame}T' \wedge I\mathsf{hsame}I' \\ &\quad \wedge N\mathsf{hsame}N' \wedge P\mathsf{hsame}P' \wedge \rho\mathsf{hsame}\rho' \wedge \lambda\mathsf{hsame}\lambda'. \end{aligned}$$

The Pkg *provenance and* Cont *ledger are shared under this comparison, so the classifier transports only the displayed martingale, continuous-path, time-index, Gaussian-step, and continuity rows. It explicitly excludes comparison of ambient probability spaces, host-level sample paths, and host path equality.*

191.2 The certificate

Theorem 191.3 (Brownian dependency surface)**.** *For every admitted* $\mathsf{Brownian}^{\uparrow}$ *process packet of Definition 191.1, every adaptedness and martingale row projects through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$*, and every path-continuity row projects through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*. The only dependency data exported to a downstream consumer are the displayed rows* M*,* C*, the* UnaryHistory *time row* T*, the Gaussian-step and continuity displays* I, N, P*, the shared* Pkg *provenance* ρ*, and the finite* Cont/hsame *ledger* λ*.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier dependency surface`

Proof. Unpack the carrier tuple. Its first coordinate M is declared to be the consumed $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ row, so all adaptedness, filtration, and martingale-step faces are obtained by projecting that dependency certificate.

The second coordinate C is declared to be the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ row. Thus the path-continuity display P is read through the continuous-map certificate surface, together with the transport rows recorded by λ.

The remaining coordinates do not create another dependency. The time row T, the adjacent-step ledger I, the normal-step display N, the provenance ρ, and the transport ledger λ only bind the two dependency certificates into one finite packet. Repacking these projections gives the stated dependency surface and no uncarried probability-space or path object. □

Theorem 191.4 (Brownian probability and martingale dependency route)**.** *Every adapted Gaussian-step read of an admitted* $\mathsf{Brownian}^{\uparrow}$ *process packet must expose the* $\mathsf{ProbSpace}^{\uparrow}$ *measure-total rows carried by chapter 184, the filtration and adaptedness rows read through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *of chapter 190, the finite time-index row* T*, the* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *path-continuity rows of chapter 130, componentwise transport, replay, provenance, and local naming before any stochastic-process consumer reads the packet. The route refuses host probability spaces, chosen sample paths, and ambient path equality.*

Proof. Start with the admitted packet $\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$. The martingale dependency row M is the only source for filtration, adaptedness, and measure-total reads, and those reads remain scoped by the ProbSpace and Martingale certificates named in the dependency route.

The finite time row T and the Gaussian-step displays I, N then bind the adapted read to the visible Brownian step ledger. The path face is separately read through C and the displayed continuity row P, so continuity comes from the ContinuousMap certificate rather than from a host path space.

Finally the provenance ρ and transport ledger λ keep these dependency rows in one finite packet. Since no coordinate names a host probability space, selected sample path, or ambient path equality, the stochastic-process consumer receives only the displayed route. □

Theorem 191.5 (Brownian classifier stability)**.** *If two admitted* $\mathsf{Brownian}^{\uparrow}$ *process packets are related by the classifier of Definition 191.2, then classifier transport preserves the displayed martingale row, continuous-map row, time row, Gaussian-step row, path-continuity row, provenance*

row, and finite Cont/hsame *ledger. In particular, the transported target packet still has the same public Brownian obligation surface over the displayed tuple* $(M, C, T, I, N, P, \rho, \lambda)$.

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier classifier stability`

Proof. Use Definition 191.3 to read the martingale and continuous-map rows only through the two dependency certificates. These projections account for the transported M and C rows without introducing a host stochastic-process object.

Use Definition 191.8 for the time-index, Gaussian-step, and continuity ledgers. The classifier supplies the displayed $T\mathsf{hsame}T'$, $I\mathsf{hsame}I'$, $N\mathsf{hsame}N'$, $P\mathsf{hsame}P'$, $\rho\mathsf{hsame}\rho'$, and $\lambda\mathsf{hsame}\lambda'$ comparisons, so every public row of the source packet has its matching target row. □

Theorem 191.6 (Brownian continuous-path projection)**.** *Every admitted* $\mathsf{Brownian}^{\uparrow}$ *process packet projects its path-continuity row through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *and its adaptedness row through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$. *These projections export only the displayed rows* C *and* M *and their certified dependency ledgers; they do not export host path equality, a host probability space, or a host stochastic-process object.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier continuous path projection`

Proof. Apply Definition 191.3. The theorem identifies M as the consumed $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ row and C as the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ row.

Then apply Definition 191.8 to the displayed transport ledger. The only path-continuity and adaptedness information that can move across a classifier comparison is the information already present in the two dependency certificates and in the finite Cont/hsame ledger. □

Theorem 191.7 (Brownian ledger exactness)**.** *Every public* $\mathsf{Brownian}^{\uparrow}$ *row is exactly one of the displayed rows* $M, C, T, I, N, P, \rho, \lambda$, *or a row projected from* M *through* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$, *or a row projected from* C *through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$. *No other host stochastic-process, probability, sample-path, or path-equality row belongs to the public Brownian ledger.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier ledger exactness`

Proof. First read the carrier packet. Its public coordinates are exactly $(M, C, T, I, N, P, \rho, \lambda)$, and Definition 191.3 says that the only dependency projections are the rows obtained from M and C.

Next read the classifier transport. By Definition 191.8, every row moved between admitted packets is one of the displayed time, step, continuity, provenance, or transport rows, again with martingale and continuous-map data delegated to the two dependency certificates.

Therefore coverage and soundness coincide: a public Brownian row is present exactly when it is displayed in the packet or projected from the named dependency certificates. □

Theorem 191.8 (Brownian step-ledger transport)**.** *If two admitted* $\mathsf{Brownian}^{\uparrow}$ *process packets are related by the classifier of Definition 191.2, then* hsame *transport of the classifier transports the independent-step ledger, the normal-step display, and the continuity ledger while leaving the visible* UnaryHistory *time-index row unchanged up to the displayed comparison* $T\mathsf{hsame}T'$.

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier step ledger transport`

Proof. Start with the time-index clause of the classifier. It supplies the displayed comparison $T\mathsf{hsame}T'$, so every adjacent-time face read by the source packet has a matched adjacent-time face in the target packet without introducing a host time set.

Next project the three ledger comparisons. The clauses $I\mathsf{hsame}I'$ and $N\mathsf{hsame}N'$ transport the independent-step and normal-step displays along the same time row, while $P\mathsf{hsame}P'$ transports the continuity display through the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ classifier.

Finally retain the shared provenance and transport boundary. The comparison $\rho\mathsf{hsame}\rho'$ keeps the dependency certificates in the same packet scope, and $\lambda\mathsf{hsame}\lambda'$ carries the finite Cont rows used by each display. Repacking these transported rows gives the target packet with the same visible time-index boundary. □

Theorem 191.9 (Brownian continuity-martingale joint classifier)**.** *If two admitted* $\mathsf{Brownian}^{\uparrow}$ *process packets are related by the classifier of Definition 191.2, then the classifier jointly transports the consumed* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *row* M*, the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *row* C*, the* $\mathsf{UnaryHistory}$ *time row* T*, the Gaussian-step display* N*, and the path-continuity display* P *by componentwise* hsame *rows. The transported packet remains a* $\mathsf{Brownian}^{\uparrow}$ *source packet and adds no host probability-space, host stochastic-process, or host path-equality comparison.*

Proof. Unpack the carrier of Definition 191.1. Its public rows are exactly $(M, C, T, I, N, P, \rho, \lambda)$, with the martingale surface read through M and the continuous-path surface read through C.

Unpack the classifier of Definition 191.2. It supplies componentwise hsame rows for M, C, T, I, N, P, ρ, and λ, so the martingale row, continuous-map row, time row, Gaussian-step display, and continuity display are transported in the same finite packet.

Repack the transported rows using Definition 191.3 and Definition 191.8. These two statements account for the dependency rows and the step-continuity ledger, while the carrier excludes every host-level comparison named in the conclusion. □

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier joint classifier transport`

Theorem 191.10 (Brownian Gaussian-step classifier stability)**.** *For classifier-related* $\mathsf{Brownian}^{\uparrow}$ *process packets, the normal-step display* N *and the adjacent-time ledger* I *remain attached to the transported* $\mathsf{UnaryHistory}$ *time row* T*. The transported Gaussian-step surface is read only through the componentwise rows* $T\mathsf{hsame}T'$*,* $I\mathsf{hsame}I'$*,* $N\mathsf{hsame}N'$*, the shared* Pkg *provenance row, and the finite* $\mathsf{Cont}/\mathsf{hsame}$ *ledger.*

Proof. Start with the time component of the classifier. The comparison $T\mathsf{hsame}T'$ is the only public transport of the displayed time row.

Project the adjacent-time and normal-step components. The comparisons $I\mathsf{hsame}I'$ and $N\mathsf{hsame}N'$ transport the step ledger and Gaussian display over the transported time row, and the provenance comparison keeps those rows inside the same packet.

Apply Definition 191.8. It states that the independent-step ledger, the normal-step display, and the continuity ledger move together under classifier transport, so the Gaussian-step surface remains attached to the transported time row and to no additional host-time object. □

Theorem 191.11 (Brownian path-continuity ledger exactness)**.** *Every public path-continuity row exported by a* $\mathsf{Brownian}^{\uparrow}$ *process packet is projected from the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *row* C *and the finite displayed ledger rows* P *and* λ*. The Brownian packet exports no host path equality, no separate path carrier, and no probability-space row beyond the displayed* BHist*,* hsame*,* Cont*, and* Pkg *support.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier path continuity ledger exactness`

Proof. Project the carrier of Definition 191.1. The only coordinates that mention path continuity are the consumed ContinuousMapUp row C, the continuity display P, and the finite transport ledger λ.

Use Definition 191.3. It identifies every path-continuity read with the continuous-map dependency row, while the remaining Brownian coordinates bind time, Gaussian steps, and provenance rather than a host path object.

Finally use Definition 191.8 to keep the continuity display stable under classifier transport. Since Definition 191.2 compares only displayed rows, the exported path-continuity ledger is exact and contains no host path equality. □

Theorem 191.12 (Brownian NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Brownian}^\uparrow}$ *obligation surface consists exactly of the carrier of Definition 191.1, the classifier of Definition 191.2, the two dependency certificates* $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$*, step-ledger transport, continuity transport, classifier stability, and ledger exactness for the displayed rows* $M, C, T, I, N, P, \rho, \lambda$*. It exports no host stochastic process, host probability space, or host path equality.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier namecert obligation surface`
↪ variant `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier semantic name certificate`

Proof. Collect the carrier and classifier rows. They list the finite tuple $(M, C, T, I, N, P, \rho, \lambda)$ and compare it componentwise by hsame, with the martingale and continuous-map faces delegated to their two named dependency certificates.

Use Definition 191.3 to account for the dependency projections. This covers all adaptedness, martingale, and path-continuity rows through the public surfaces of $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$.

Use Definition 191.8 for the transport surface. It preserves the independent-step, normal-step, and continuity ledgers by hsame and finite Cont transport, while the visible time row is kept as the same displayed $\mathsf{UnaryHistory}$ boundary.

Ledger exactness is the reverse reading of the packet: every public Brownian row must be one of $M, C, T, I, N, P, \rho, \lambda$ or a row already scoped by M or C. Since the carrier and classifier contain no additional host probability-space, stochastic-process, or path-equality coordinate, the public obligation surface is exactly the one stated. □

Theorem 191.13 (Brownian path-step boundary exhaustion)**.** *Every public* $\mathsf{Brownian}^\uparrow$ *consumer row mentioning either the Gaussian-step display or the path-continuity display factors through the displayed tuple* $(M, C, T, I, N, P, \rho, \lambda)$*. The martingale rows project only through* $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$*, path-continuity rows only through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$*, and step rows only through the common* $\mathsf{UnaryHistory}$ *time row, the* $\mathsf{Cont}/\mathsf{hsame}$ *ledger, and the* Pkg *provenance.*

The boundary exports no host probability-space equality, no host stochastic-process equality, and no host path equality. It is exactly the joint consumer boundary supplied by the Brownian packet carrier, classifier, dependency projections, and ledger rows.

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier path step boundary exhaustion`

Proof. Begin with the dependency surface. By Definition 191.3, the martingale and continuous-path faces of any admitted packet are read only through M and C, with the visible time row T, the Gaussian-step and continuity displays I, N, P, the provenance ρ, and the ledger λ remaining as displayed packet coordinates.

Next apply the step transport row. Theorem 191.8 states that Gaussian-step transport uses the common $\mathsf{UnaryHistory}$ time row, the adjacent-step and normal-step displays, the provenance

row, and the finite Cont/hsame ledger. Thus no additional host time object or stochastic process coordinate is exposed by a step consumer.

For path-continuity consumers, use Definition 191.11. It identifies every public path-continuity row with the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ row C together with the displayed continuity and transport rows P and λ.

Finally apply Definition 191.12. It exhausts the public $\mathsf{NameCert}_{\mathsf{Brownian}^{\uparrow}}$ surface by the carrier, classifier, two dependency certificates, step-ledger transport, continuity transport, classifier stability, and ledger exactness over $(M, C, T, I, N, P, \rho, \lambda)$. Combining these four surfaces gives the stated factorisation and rules out host probability-space, stochastic-process, and path-equality comparisons. □

Theorem 191.14 (Brownian time-step path interface)**.** *For every admitted* $\mathsf{Brownian}^{\uparrow}$ *process packet*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda),$$

the displayed BHist *tuple binds one public interface containing the martingale row M, the continuous-path row C, the unary time row T, the adjacent-step ledger I, the Gaussian display N, the path-continuity display P, the shared* Pkg *provenance ρ, and the finite* Cont/hsame *ledger λ. Every public row read by a time-step, path-continuity, or Brownian NameCert consumer factors through this interface.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier time step path interface`

Proof. Unpack the carrier of Definition 191.1. Its displayed coordinates are exactly $(M, C, T, I, N, P, \rho, \lambda)$, with M and C assigned to the martingale and continuous-map dependency rows.

Apply Definition 191.3. It keeps adaptedness and martingale data inside M, keeps path-continuity data inside C, and exposes only the displayed time, step, Gaussian, continuity, provenance, and transport rows beside those dependencies.

Use Definition 191.5 and Definition 191.8 to transport the same interface across hsame comparisons. The classifier moves $T, I, N, P, \rho, \lambda$ rowwise, so a time-step consumer and a path consumer read the same public packet boundary.

Finally apply Definition 191.12. Its exact surface is the carrier, classifier, dependency surface, step transport, continuity transport, classifier stability, and ledger exactness over the same tuple, which is precisely the asserted interface. □

Theorem 191.15 (Brownian Gaussian step ledger coverage)**.** *Every public* $\mathsf{Brownian}^{\uparrow}$ *consumer row mentioning an adjacent Gaussian step is exhausted by the unary time row T, the adjacent-step ledger I, the normal-step display N, the provenance row ρ, the transport ledger λ, the relevant* UnaryHistory *witnesses, and the public* Cont/hsame *transports. No Gaussian adjacent-step consumer obtains an additional host probability-space row, stochastic-process equality row, or path-equality row.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier gaussian step ledger coverage`

Proof. Start with the step row in the carrier. By Definition 191.1, the only Gaussian-step coordinates are the time row T, the adjacent-step ledger I, the normal display N, the provenance ρ, and the transport ledger λ.

Apply Definition 191.8. It states that transport of Gaussian-step information uses the common UnaryHistory time boundary, the adjacent-step and normal-step displays, the provenance row, and the finite Cont/hsame ledger.

Use Definition 191.10 to identify the same rows after classifier transport. The step classifier has no coordinate for ambient probability spaces, host stochastic-process equality, or host path-equality comparison.

Finish with Definition 191.13. It exhausts every step consumer by the displayed Brownian packet boundary, so the listed rows cover every Gaussian adjacent-step read. □

Theorem 191.16 (Brownian continuity martingale compatibility)**.** *If two public rows are read from an admitted Brownian packet through* $\mathsf{NameCert}_{\mathsf{Martingale}\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{ContinuousMap}\uparrow}$*, then transport by the same Brownian classifier keeps the martingale row and the continuous-path row compatible over the shared tuple* $(M, C, T, I, N, P, \rho, \lambda)$*. The common* Pkg *provenance and finite* Cont/hsame *ledger are the only public rows used to join the two dependency certificates.*

Proof. Project the two dependency rows with Definition 191.3. The martingale information is read through M, and the continuous-path information is read through C, while the shared time, step, continuity, provenance, and transport rows remain displayed coordinates of the Brownian packet.

Apply Definition 191.9. It reads the classifier rowwise over $M, C, T, I, N, P, \rho, \lambda$, so the martingale and continuous-map dependency comparisons move under one Brownian classifier rather than under two unrelated transports.

Use Definition 191.11 for the continuous path face and Definition 191.8 for the time-step ledger. These two rows share the same Pkg provenance and Cont/hsame ledger, so the transported dependency surfaces remain compatible.

Finally apply Definition 191.12. Since the public Brownian surface contains no host path equality, probability-space equality, or stochastic-process equality, compatibility is exactly the shared classifier transport of the displayed packet rows. □

Theorem 191.17 (Brownian step-path NameCert closure boundary)**.** *For every displayed Brownian packet* $\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$ *of Definition 191.1, the carrier packet, dependency surface, Gaussian-step ledger coverage, path-continuity ledger exactness, and continuity-martingale compatibility determine one public* $\mathsf{NameCert}_{\mathsf{Brownian}\uparrow}$ *boundary over the tuple* $(M, C, T, I, N, P, \rho, \lambda)$*. The boundary exports only the displayed martingale dependency row* M*, continuous-path dependency row* C*, time row* T*, adjacent-step ledger* I*, normal-step display* N*, path-continuity row* P*, shared provenance* ρ*, and finite* Cont/hsame *ledger* λ*; it does not export host stochastic-process equality, host path equality, or an ambient probability-space comparison.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier step path namecert closure boundary`

Proof. Begin with the carrier and dependency surface. Theorem 191.3 projects all martingale reads through M and all continuous-path reads through C, while leaving $T, I, N, P, \rho, \lambda$ as the visible Brownian rows.

Next attach the step and path ledgers. Theorem 191.8 transports the finite time-step rows, and Definition 191.15 exhausts every public adjacent Gaussian-step consumer by T, I, N, ρ, λ and the relevant UnaryHistory and Cont/hsame witnesses. The path face is fixed by the path-continuity ledger exactness already carried in the Brownian certificate surface.

Finally apply Definition 191.16. It joins the martingale and continuous-path dependency rows through the same classifier over $(M, C, T, I, N, P, \rho, \lambda)$. The exact public field list is then the one identified by Definition 191.12, so these rows determine the stated $\mathsf{NameCert}_{\mathsf{Brownian}\uparrow}$ boundary and nothing outside the displayed packet. □

Theorem 191.18 (Brownian carrier-classifier obligation package)**.** *For every displayed Brownian packet $\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$, the carrier of Definition 191.1, the classifier of Definition 191.2, the stability row of Definition 191.5, and the joint martingale–path classifier of Definition 191.9 form the carrier/classifier obligation row for* $\mathsf{Brownian}^{\uparrow}$*. This row binds only the displayed* BHist *carrier, the* $\mathsf{UnaryHistory}$ *time row, the two dependency certificates, the shared* Pkg *provenance, and the finite* $\mathsf{Cont/hsame}$ *transport ledger.*

Proof. Read the carrier from Definition 191.1. Its coordinates are precisely $M, C, T, I, N, P, \rho, \lambda$, with no additional host probability-space, stochastic-process, or path-equality coordinate.

Attach the classifier of Definition 191.2. The comparison is componentwise over the displayed rows, so the classifier has the same carrier boundary as the packet it compares.

Use Definition 191.5 for transport across hsame and Definition 191.9 for the joint martingale–continuous-map face. Together they show that M and C move under the same Brownian classifier as the time, step, continuity, provenance, and ledger rows. This is exactly the carrier/classifier obligation row. □

Theorem 191.19 (Brownian dependency-ledger obligation exactness)**.** *For every displayed Brownian packet $\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$, the dependency surface of Definition 191.3, the step transport of Definition 191.8, the path-continuity ledger exactness of Definition 191.11, and the Brownian ledger exactness of Definition 191.7 form the dependency-ledger obligation row for* $\mathsf{Brownian}^{\uparrow}$*. Every dependency read is through M or C, and every ledger read is through $T, I, N, P, \rho, \lambda$ and the displayed* $\mathsf{Cont/hsame}$ *witnesses.*

Proof. First project the dependency rows with Definition 191.3. The martingale surface is read through M, and the continuous-path surface is read through C.

Next attach the step ledger. Theorem 191.8 keeps adjacent-step and normal-step information on the displayed time, provenance, and finite transport rows.

For the path face, use Definition 191.11. It identifies the continuity display with C, P, and the same ledger λ.

Finally apply Definition 191.7. Since every public ledger coordinate is one of the displayed packet rows or a row already scoped by M or C, the dependency-ledger row is exact. □

Theorem 191.20 (Brownian obligation closure boundary)**.** *The Brownian obligation surface is exactly the union of the carrier/classifier row of Definition 191.18, the dependency-ledger row of Definition 191.19, the NameCert surface of Definition 191.12, and the step-path boundary of Definition 191.17. Thus the* $\mathsf{Brownian}^{\uparrow}$ *obligation boundary exports precisely the displayed $M, C, T, I, N, P, \rho, \lambda$ packet, its* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *dependencies, its* BHist *and* $\mathsf{UnaryHistory}$ *rows, its* Pkg *provenance, and its finite* $\mathsf{Cont/hsame}$ *ledger.*

Proof. The carrier/classifier package supplies the admitted packet and the comparison surface over the displayed rows. It fixes the public object language in which a Brownian consumer may read $M, C, T, I, N, P, \rho, \lambda$.

The dependency-ledger package supplies the exact dependency reads and the exact finite ledger reads. It prevents any extra host probability-space, stochastic-process, or path-equality row from entering through M, C, or the time-step and path-continuity ledgers.

The NameCert surface identifies the public Brownian certificate over these same rows, and the step-path boundary identifies the common consumer boundary for Gaussian steps and path continuity. Combining the four rows gives the stated obligation surface, and each cited row has the same displayed packet as its boundary. □

Theorem 191.21 (Brownian finite-prefix carrier restriction)**.** *Let*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$$

be an admitted $\mathsf{Brownian}^{\uparrow}$ *process packet in the sense of Definition 191.1. Suppose the displayed unary time row* T *has length* $n+k$*, and choose the initial finite subledger* $T_{\leq n}$ *containing exactly the first* n *adjacent time faces. Let* $I_{\leq n}$*,* $N_{\leq n}$*,* $P_{\leq n}$*,* $\rho_{\leq n}$*, and* $\lambda_{\leq n}$ *be the subrows of the adjacent-step ledger, normal-step display, path-continuity display, provenance row, and finite* $\mathsf{Cont/hsame}$ *ledger whose displayed faces lie entirely over* $T_{\leq n}$*. Let* $M_{\leq n}$ *be the finite-prefix restriction of the consumed* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *row* M*, and keep the same consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *path row* C*.*

Then

$$\mathcal{B}_{\leq n} = (M_{\leq n}, C, T_{\leq n}, I_{\leq n}, N_{\leq n}, P_{\leq n}, \rho_{\leq n}, \lambda_{\leq n})$$

is again an admitted $\mathsf{Brownian}^{\uparrow}$ *process packet. In particular, restricting the Brownian path-step carrier to the first* n *displayed time steps preserves the carrier boundary and introduces no host probability-space row, host stochastic-process row, or host path-equality row.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier finite prefix carrier restriction`

Proof. Unpack the Brownian carrier of Definition 191.1. The admitted packet is a finite displayed tuple whose public rows are exactly $M, C, T, I, N, P, \rho, \lambda$. The time row is a finite $\mathsf{UnaryHistory}$ row, so its initial subledger $T_{\leq n}$ is again a finite displayed unary time row. The rows $I_{\leq n}$ and $N_{\leq n}$ are obtained by retaining precisely the adjacent faces of I and N whose endpoints remain in $T_{\leq n}$; no boundary face crossing from the prefix into the omitted tail is retained.

For the martingale dependency, apply Definition 190.17 to M. It gives the accepted finite-prefix martingale packet $M_{\leq n}$ over the same displayed initial horizon. Thus all adaptedness, conditional-expectation, filtration, provenance, and finite Cont rows read through the Brownian martingale input remain licensed by $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ after the time-prefix restriction.

For the Gaussian-step face, use Definition 191.15. Every public adjacent Gaussian-step consumer reads only the unary time row, the adjacent-step ledger, the normal-step display, the provenance row, the finite transport ledger, and the corresponding $\mathsf{UnaryHistory}$ and $\mathsf{Cont/hsame}$ witnesses. The restriction keeps exactly those witnesses for the retained adjacent faces, so the Gaussian-step surface over $T_{\leq n}$ is still displayed inside the finite packet.

For the path-continuity face, use Definition 191.11. Every public path-continuity row is projected from the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ row C together with the displayed continuity row P and the finite ledger λ. The restricted row $P_{\leq n}$ contains the same modulus and certificate witnesses that the original packet displayed on the retained time faces, and $\lambda_{\leq n}$ contains exactly the transport entries needed by those retained witnesses. Since C is unchanged, the continuous-map dependency surface still supplies the path-continuity rows read by the prefix packet.

It remains to repack the provenance and exactness boundary. The rows $\rho_{\leq n}$ and $\lambda_{\leq n}$ are finite subledgers of the original shared Pkg provenance and $\mathsf{Cont/hsame}$ ledger, restricted to the retained martingale, time-step, Gaussian, and path-continuity faces. By Definition 191.7 and Definition 191.19, every public row of the restricted tuple is either one of its displayed coordinates or a row already scoped by $M_{\leq n}$ or C. Finally Definition 191.12 identifies these displayed carrier, dependency, classifier, and ledger rows as the public $\mathsf{NameCert}_{\mathsf{Brownian}^{\uparrow}}$ obligation surface. Therefore $\mathcal{B}_{\leq n}$ satisfies the Brownian carrier definition with no additional host-level data. □

Theorem 191.22 (Brownian finite-suffix carrier restriction)**.** *Let*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$$

be an admitted $\mathsf{Brownian}^{\uparrow}$ *process packet in the sense of Definition 191.1. Suppose the displayed unary time row* T *has length* $n+k$*. Let* $T^{\mathsf{sh}}_{[n,n+k]}$ *be the finite* k*-step unary row obtained by retaining the adjacent faces from the boundary endpoint at time* n *through the endpoint at time* $n+k$ *and then relabelling that boundary endpoint as the initial endpoint. Let*

$$I^{\mathsf{sh}}_{[n,n+k]}, \quad N^{\mathsf{sh}}_{[n,n+k]}, \quad P^{\mathsf{sh}}_{[n,n+k]}, \quad \rho^{\mathsf{sh}}_{[n,n+k]}, \quad \lambda^{\mathsf{sh}}_{[n,n+k]}$$

be the corresponding finite subrows of the adjacent-step ledger, normal-step display, path-continuity display, provenance row, and $\mathsf{Cont/hsame}$ *ledger whose displayed faces lie entirely in that closed suffix interval, with each time index translated by* $j \mapsto j-n$*. Let* $M^{\mathsf{sh}}_{[n,n+k]}$ *be the same translated suffix of the consumed* $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ *row, retaining exactly the endpoint rows and conditional-expectation faces with indices* $n \le j \le n+k$*. Let* $C^{\mathsf{sh}}_{[n,n+k]}$ *be the path-continuity subrow read from the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *row* C *on the retained path segment, with the time-source boundary reset to the displayed endpoint at time* n*.*

Then

$$\mathcal{B}^{\mathsf{sh}}_{[n,n+k]} = (M^{\mathsf{sh}}_{[n,n+k]}, C^{\mathsf{sh}}_{[n,n+k]}, T^{\mathsf{sh}}_{[n,n+k]}, I^{\mathsf{sh}}_{[n,n+k]}, N^{\mathsf{sh}}_{[n,n+k]}, P^{\mathsf{sh}}_{[n,n+k]}, \rho^{\mathsf{sh}}_{[n,n+k]}, \lambda^{\mathsf{sh}}_{[n,n+k]})$$

is again an admitted $\mathsf{Brownian}^{\uparrow}$ *process packet. Thus the Brownian path-step carrier is closed under finite suffix restriction from any displayed boundary endpoint, and the restricted carrier introduces no host probability-space row, no host stochastic-process row, and no host path-equality row.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier finite suffix carrier restriction`

Proof. Unpack the carrier of Definition 191.1. An admitted Brownian packet is a finite displayed tuple, and every public Brownian row is one of $M, C, T, I, N, P, \rho, \lambda$ or is read through the two consumed dependency certificates. Taking the interval from the boundary endpoint n to $n+k$ therefore selects a finite subtuple of already displayed rows. The translation $j \mapsto j-n$ changes only the displayed position names in the finite unary spine; it does not create a further time object.

For the martingale dependency, read M through the concrete $\mathsf{Martingale}^{\uparrow}$ packet surface. By Definition 190.5, the martingale carrier is a finite tuple of endpoint rows, adjacent conditional-expectation rows, tower-boundary rows, provenance, and finite Cont transport. The suffix row $M^{\mathsf{sh}}_{[n,n+k]}$ retains precisely those endpoint rows and adjacent faces whose indices lie in the displayed interval $[n, n+k]$, with the endpoint at n serving as the shifted initial value. Each retained successor face is one of the conditional-expectation faces already accepted in M, and Definition 190.16 identifies these retained finite rows as the public $\mathsf{NameCert}_{\mathsf{Martingale}^{\uparrow}}$ surface. Hence the martingale input needed by the shifted Brownian packet is still licensed by the consumed martingale certificate.

For the Gaussian-step surface, apply Definition 191.15. It says that every public adjacent Gaussian-step read uses only the unary time row, the adjacent-step ledger, the normal-step display, the provenance row, the finite transport ledger, and the corresponding $\mathsf{UnaryHistory}$ and $\mathsf{Cont/hsame}$ witnesses. The rows $T^{\mathsf{sh}}_{[n,n+k]}$, $I^{\mathsf{sh}}_{[n,n+k]}$, $N^{\mathsf{sh}}_{[n,n+k]}$, $\rho^{\mathsf{sh}}_{[n,n+k]}$, and $\lambda^{\mathsf{sh}}_{[n,n+k]}$ keep exactly those witnesses for the retained adjacent faces. No step crossing the cut between $n-1$ and n is retained, so the shifted packet has a closed k-step Gaussian ledger over its reset time boundary.

For the path-continuity surface, use Definition 191.11. Every public path-continuity row of a Brownian packet is projected from the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ row together with the displayed continuity row P and the finite ledger λ. The row $C^{\mathsf{sh}}_{[n,n+k]}$ is the retained continuous-map certificate read on the same path segment, while $P^{\mathsf{sh}}_{[n,n+k]}$ and $\lambda^{\mathsf{sh}}_{[n,n+k]}$ keep exactly the modulus, continuity, and transport entries whose source and target faces lie in the suffix interval. Resetting

the source boundary to the endpoint at time n is therefore a ledger relabelling inside the displayed segment, not a host path equality.

It remains to repack the exact public boundary. By Definition 191.3, the shifted martingale and path rows are still read only through the two dependency certificates, and the time, Gaussian-step, continuity, provenance, and transport rows remain displayed packet coordinates. By Definition 191.7 and Definition 191.19, every public row of the shifted tuple is either one of its displayed coordinates or a row scoped by the shifted martingale dependency or the shifted continuous-map dependency. Finally Definition 191.12 identifies exactly such a displayed carrier, dependency, classifier, and finite-ledger package as the $\mathsf{NameCert}_{\mathsf{Brownian}^\uparrow}$ obligation surface. Therefore $\mathcal{B}^{\mathsf{sh}}_{[n,n+k]}$ is an admitted Brownian process packet, with no added host-level probability, stochastic-process, or path-equality data. □

Theorem 191.23 (Brownian finite prefix suffix bridge boundary)**.** *The finite-prefix restriction of Definition 191.24 and the finite-suffix restriction of Definition 191.25 form the explicit* $\mathsf{Brownian}^\uparrow$ *bridge boundary over the consumed* $\mathsf{Martingale}^\uparrow$ *and* $\mathsf{ContinuousMap}^\uparrow$ *rows, the displayed* BHist *and* UnaryHistory *rows, the shared* Pkg *provenance, the finite* Cont *transport, and the* hsame *classifier rows. A downstream bridge read of the Brownian packet therefore factors through these finite prefix and suffix surfaces, and no host probability-space row, stochastic-process row, or path equality is added to the public boundary.*

Proof. The prefix theorem identifies the Brownian carrier after an initial finite time cut. Its retained rows are the consumed martingale prefix, the unchanged continuous-map dependency, the initial UnaryHistory time row, the retained Gaussian-step and continuity displays, the shared Pkg provenance, and the finite Cont/hsame transport ledger.

The suffix theorem identifies the same carrier after a finite shifted suffix cut. It resets the displayed boundary endpoint by relabelling the finite unary spine and retains only those martingale, continuous-map, Gaussian, provenance, and transport faces whose source and target lie in the closed suffix interval.

Together these two restrictions cover the bridge-facing boundary: every consumer read of a Brownian finite segment is either a prefix read from the initial endpoint or a suffix read from a displayed boundary endpoint. The dependency reads are still mediated by $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$, while the remaining rows are exactly the displayed BHist, UnaryHistory, Pkg, Cont, and hsame rows. Thus the bridge boundary is finite and visible, with no ambient stochastic object beyond the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Brownian}^\uparrow}$**.**

Constructive story: A Brownian packet is not a host path space; it is the finite displayed tuple $(M, C, T, I, N, P, \rho, \lambda)$ whose martingale row M is read through $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$, whose path-continuity row C is read through $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$, and whose time, adjacent-step, normal-step, provenance, and transport rows are all BHist data. The carrier accepts exactly those tuples for which the martingale dependency supplies adaptedness and step rows, the continuous-map dependency supplies path-continuity rows, and the normal-step and adjacent-time ledgers are displayed, while the classifier transports $M, C, T, I, N, P, \rho, \lambda$ rowwise by the two dependency certificates and by hsame. The source therefore covers only this finite step-continuity packet, the pattern is martingale adaptedness plus Gaussian adjacent-step data and continuous-path readback, and the ledger keeps every public coordinate either as one of the displayed tuple rows or as a row projected from M or C. Since the two dependency classifiers are equivalence relations and the remaining fields use componentwise history-sameness, an admitted Brownian packet supplies the carrier witness, the classifier is reflexive, symmetric, and transitive, and transported packets keep the same obligation surface. A packet h lies in the signature gap of a $\mathsf{Brownian}^\uparrow$ token p when the source-domain accepts its martingale row, continuous-path row, time spine, step ledger, normal-step display, and provenance, the signature relation pairs that tuple with an answer tuple and evidence ledger, and token introduction names the answer as p; unfolded, the token records exactly those adapted, continuous, Gaussian-step rows. The finite prefix and suffix restriction boundary exports the visible bridge surface over MartingaleUp, ContinuousMapUp, BHist, UnaryHistory, Pkg, Cont, and hsame rows. The chapter registers this finite step-continuity construction under

the name Brownian$^{\uparrow}$; Brownian motion is represented by dependency certificates and displayed ledgers, with no Gaussian measure construction, path-space completeness, stochastic calculus, or host stochastic-process object introduced.

Theory closure: TheoryGate(NameCert$_{\text{Brownian}\uparrow}$, maturePackageClosed)

Scope closed: The obligation scope binds the Brownian step-continuity packet to the MartingaleUp dependency row, ContinuousMapUp path row, time/path rows, normal-step row, shared provenance, finite ledger BHist rows, and the finite prefix and suffix restriction bridge surface already displayed for Brownian readers.

Formal verification: FormalStatus(NameCert$_{\text{Brownian}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: No standard Brownian motion, Gaussian measure construction, stochastic calculus, path-space completeness, or probability-space completeness theorem is claimed here.

Upgrade path: No further paper theory-axis upgrade is recorded for this Brownian surface; bridgeChecked remains a separate formal-axis matter.

Theorem 191.24 (Brownian finite-prefix carrier restriction)**.** *Let*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$$

be an admitted Brownian$^{\uparrow}$ *process packet in the sense of Definition 191.1. Suppose the displayed unary time row T has length $n+k$, and choose the initial finite subledger $T_{\leq n}$ containing exactly the first n adjacent time faces. Let $I_{\leq n}$, $N_{\leq n}$, $P_{\leq n}$, $\rho_{\leq n}$, and $\lambda_{\leq n}$ be the subrows of the adjacent-step ledger, normal-step display, path-continuity display, provenance row, and finite* Cont/hsame *ledger whose displayed faces lie entirely over $T_{\leq n}$. Let $M_{\leq n}$ be the finite-prefix restriction of the consumed* NameCert$_{\text{Martingale}\uparrow}$ *row M, and keep the same consumed* NameCert$_{\text{ContinuousMap}\uparrow}$ *path row C.*

Then

$$\mathcal{B}_{\leq n} = (M_{\leq n}, C, T_{\leq n}, I_{\leq n}, N_{\leq n}, P_{\leq n}, \rho_{\leq n}, \lambda_{\leq n})$$

is again an admitted Brownian$^{\uparrow}$ *process packet. In particular, restricting the Brownian path-step carrier to the first n displayed time steps preserves the carrier boundary and introduces no host probability-space row, host stochastic-process row, or host path-equality row.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier finite prefix carrier restriction`

Proof. Unpack the Brownian carrier of Definition 191.1. The admitted packet is a finite displayed tuple whose public rows are exactly $M, C, T, I, N, P, \rho, \lambda$. The time row is a finite UnaryHistory row, so its initial subledger $T_{\leq n}$ is again a finite displayed unary time row. The rows $I_{\leq n}$ and $N_{\leq n}$ are obtained by retaining precisely the adjacent faces of I and N whose endpoints remain in $T_{\leq n}$; no boundary face crossing from the prefix into the omitted tail is retained.

For the martingale dependency, apply Definition 190.17 to M. It gives the accepted finite-prefix martingale packet $M_{\leq n}$ over the same displayed initial horizon. Thus all adaptedness, conditional-expectation, filtration, provenance, and finite Cont rows read through the Brownian martingale input remain licensed by NameCert$_{\text{Martingale}\uparrow}$ after the time-prefix restriction.

For the Gaussian-step face, use Definition 191.15. Every public adjacent Gaussian-step consumer reads only the unary time row, the adjacent-step ledger, the normal-step display, the provenance row, the finite transport ledger, and the corresponding UnaryHistory and Cont/hsame witnesses. The restriction keeps exactly those witnesses for the retained adjacent faces, so the Gaussian-step surface over $T_{\leq n}$ is still displayed inside the finite packet.

For the path-continuity face, use Definition 191.11. Every public path-continuity row is projected from the consumed NameCert$_{\text{ContinuousMap}\uparrow}$ row C together with the displayed continuity row P and the finite ledger λ. The restricted row $P_{\leq n}$ contains the same modulus and certificate witnesses that the original packet displayed on the retained time faces, and $\lambda_{\leq n}$ contains exactly the

transport entries needed by those retained witnesses. Since C is unchanged, the continuous-map dependency surface still supplies the path-continuity rows read by the prefix packet.

It remains to repack the provenance and exactness boundary. The rows $\rho_{\le n}$ and $\lambda_{\le n}$ are finite subledgers of the original shared Pkg provenance and $\mathsf{Cont/hsame}$ ledger, restricted to the retained martingale, time-step, Gaussian, and path-continuity faces. By Definition 191.7 and Definition 191.19, every public row of the restricted tuple is either one of its displayed coordinates or a row already scoped by $M_{\le n}$ or C. Finally Definition 191.12 identifies these displayed carrier, dependency, classifier, and ledger rows as the public $\mathsf{NameCert}_{\mathsf{Brownian}^\uparrow}$ obligation surface. Therefore $\mathcal{B}_{\le n}$ satisfies the Brownian carrier definition with no additional host-level data. □

Theorem 191.25 (Brownian finite-suffix carrier restriction)**.** *Let*

$$\mathcal{B} = (M, C, T, I, N, P, \rho, \lambda)$$

be an admitted $\mathsf{Brownian}^\uparrow$ *process packet in the sense of Definition 191.1. Suppose the displayed unary time row T has length $n+k$. Let $T^{\mathsf{sh}}_{[n,n+k]}$ be the finite k-step unary row obtained by retaining the adjacent faces from the boundary endpoint at time n through the endpoint at time $n+k$ and then relabelling that boundary endpoint as the initial endpoint. Let*

$$I^{\mathsf{sh}}_{[n,n+k]}, \quad N^{\mathsf{sh}}_{[n,n+k]}, \quad P^{\mathsf{sh}}_{[n,n+k]}, \quad \rho^{\mathsf{sh}}_{[n,n+k]}, \quad \lambda^{\mathsf{sh}}_{[n,n+k]}$$

be the corresponding finite subrows of the adjacent-step ledger, normal-step display, path-continuity display, provenance row, and $\mathsf{Cont/hsame}$ *ledger whose displayed faces lie entirely in that closed suffix interval, with each time index translated by $j \mapsto j-n$. Let $M^{\mathsf{sh}}_{[n,n+k]}$ be the same translated suffix of the consumed* $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ *row, retaining exactly the endpoint rows and conditional-expectation faces with indices $n \le j \le n+k$. Let $C^{\mathsf{sh}}_{[n,n+k]}$ be the path-continuity subrow read from the consumed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ *row C on the retained path segment, with the time-source boundary reset to the displayed endpoint at time n.*

Then

$$\mathcal{B}^{\mathsf{sh}}_{[n,n+k]} = (M^{\mathsf{sh}}_{[n,n+k]}, C^{\mathsf{sh}}_{[n,n+k]}, T^{\mathsf{sh}}_{[n,n+k]}, I^{\mathsf{sh}}_{[n,n+k]}, N^{\mathsf{sh}}_{[n,n+k]}, P^{\mathsf{sh}}_{[n,n+k]}, \rho^{\mathsf{sh}}_{[n,n+k]}, \lambda^{\mathsf{sh}}_{[n,n+k]})$$

is again an admitted $\mathsf{Brownian}^\uparrow$ *process packet. Thus the Brownian path-step carrier is closed under finite suffix restriction from any displayed boundary endpoint, and the restricted carrier introduces no host probability-space row, no host stochastic-process row, and no host path-equality row.*

Lean4 ✓ `BEDC.Derived.BrownianUp.BrownianStepContinuityClassifier finite suffix carrier restriction`

Proof. Unpack the carrier of Definition 191.1. An admitted Brownian packet is a finite displayed tuple, and every public Brownian row is one of $M, C, T, I, N, P, \rho, \lambda$ or is read through the two consumed dependency certificates. Taking the interval from the boundary endpoint n to $n+k$ therefore selects a finite subtuple of already displayed rows. The translation $j \mapsto j-n$ changes only the displayed position names in the finite unary spine; it does not create a further time object.

For the martingale dependency, read M through the concrete $\mathsf{Martingale}^\uparrow$ packet surface. By Definition 190.5, the martingale carrier is a finite tuple of endpoint rows, adjacent conditional-expectation rows, tower-boundary rows, provenance, and finite Cont transport. The suffix row $M^{\mathsf{sh}}_{[n,n+k]}$ retains precisely those endpoint rows and adjacent faces whose indices lie in the displayed interval $[n, n+k]$, with the endpoint at n serving as the shifted initial value. Each retained successor face is one of the conditional-expectation faces already accepted in M, and Definition 190.16

identifies these retained finite rows as the public $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ surface. Hence the martingale input needed by the shifted Brownian packet is still licensed by the consumed martingale certificate.

For the Gaussian-step surface, apply Definition 191.15. It says that every public adjacent Gaussian-step read uses only the unary time row, the adjacent-step ledger, the normal-step display, the provenance row, the finite transport ledger, and the corresponding UnaryHistory and Cont/hsame witnesses. The rows $T^{\mathsf{sh}}_{[n,n+k]}$, $I^{\mathsf{sh}}_{[n,n+k]}$, $N^{\mathsf{sh}}_{[n,n+k]}$, $\rho^{\mathsf{sh}}_{[n,n+k]}$, and $\lambda^{\mathsf{sh}}_{[n,n+k]}$ keep exactly those witnesses for the retained adjacent faces. No step crossing the cut between $n-1$ and n is retained, so the shifted packet has a closed k-step Gaussian ledger over its reset time boundary.

For the path-continuity surface, use Definition 191.11. Every public path-continuity row of a Brownian packet is projected from the consumed $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ row together with the displayed continuity row P and the finite ledger λ. The row $C^{\mathsf{sh}}_{[n,n+k]}$ is the retained continuous-map certificate read on the same path segment, while $P^{\mathsf{sh}}_{[n,n+k]}$ and $\lambda^{\mathsf{sh}}_{[n,n+k]}$ keep exactly the modulus, continuity, and transport entries whose source and target faces lie in the suffix interval. Resetting the source boundary to the endpoint at time n is therefore a ledger relabelling inside the displayed segment, not a host path equality.

It remains to repack the exact public boundary. By Definition 191.3, the shifted martingale and path rows are still read only through the two dependency certificates, and the time, Gaussian-step, continuity, provenance, and transport rows remain displayed packet coordinates. By Definition 191.7 and Definition 191.19, every public row of the shifted tuple is either one of its displayed coordinates or a row scoped by the shifted martingale dependency or the shifted continuous-map dependency. Finally Definition 191.12 identifies exactly such a displayed carrier, dependency, classifier, and finite-ledger package as the $\mathsf{NameCert}_{\mathsf{Brownian}^\uparrow}$ obligation surface. Therefore $\mathcal{B}^{\mathsf{sh}}_{[n,n+k]}$ is an admitted Brownian process packet, with no added host-level probability, stochastic-process, or path-equality data. □

Theorem 191.26 (Brownian finite prefix suffix bridge boundary)**.** *The finite-prefix restriction of Definition 191.24 and the finite-suffix restriction of Definition 191.25 form the explicit* $\mathsf{Brownian}^\uparrow$ *bridge boundary over the consumed* $\mathsf{Martingale}^\uparrow$ *and* $\mathsf{ContinuousMap}^\uparrow$ *rows, the displayed* BHist *and* UnaryHistory *rows, the shared* Pkg *provenance, the finite* Cont *transport, and the* hsame *classifier rows. A downstream bridge read of the Brownian packet therefore factors through these finite prefix and suffix surfaces, and no host probability-space row, stochastic-process row, or path equality is added to the public boundary.*

Proof. The prefix theorem identifies the Brownian carrier after an initial finite time cut. Its retained rows are the consumed martingale prefix, the unchanged continuous-map dependency, the initial UnaryHistory time row, the retained Gaussian-step and continuity displays, the shared Pkg provenance, and the finite Cont/hsame transport ledger.

The suffix theorem identifies the same carrier after a finite shifted suffix cut. It resets the displayed boundary endpoint by relabelling the finite unary spine and retains only those martingale, continuous-map, Gaussian, provenance, and transport faces whose source and target lie in the closed suffix interval.

Together these two restrictions cover the bridge-facing boundary: every consumer read of a Brownian finite segment is either a prefix read from the initial endpoint or a suffix read from a displayed boundary endpoint. The dependency reads are still mediated by $\mathsf{NameCert}_{\mathsf{Martingale}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$, while the remaining rows are exactly the displayed BHist, UnaryHistory, Pkg, Cont, and hsame rows. Thus the bridge boundary is finite and visible, with no ambient stochastic object beyond the displayed packet. □

The Brownian process packet consumes chapter 190, chapter 184, and chapter 130 before its finite time row, Gaussian-step row, path-continuity row, provenance row, and transport rows are readable. The finite-prefix consumer Definition 191.24, the finite-suffix consumer Definition 191.25, and the combined boundary Definition 191.26 name the exposed restriction surface for martingale, probspace, and continuousmap siblings; the route refuses a host probability space, chosen sample paths, and ambient path equality.

Chapter 192

A Concrete Naming Certificate for $\mathsf{Entropy}^{\uparrow}$

The entropy interface $\mathsf{Entropy}^{\uparrow}$ packages the Shannon entropy $-\sum p_i \log p_i$ of a discrete distribution and the differential entropy $-\int f \log f$ of a continuous one. It depends on $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186) and $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ (chapter 123).

192.1 Carrier definition

Definition 192.1 (EntropyUp BHist measure source surface)**.** *An* $\mathsf{Entropy}^{\uparrow}$ *BHist measure source surface consists of a* $\mathsf{Distribution}^{\uparrow}$ *pushforward row, an* $\mathsf{Integral}^{\uparrow}$ *finite or integral readback row, a log-weight observation row treated as a certified BHist endpoint,* hsame *transport rows for the distribution, integral, and log-weight endpoints,* Cont *ledger rows connecting the displayed observations, and* Pkg *provenance for the whole package.*

Lean4 (definition) `BEDC.Derived.EntropyUp.EntropySourceSurface`

Definition 192.2 (EntropyUp log-partition BHist carrier)**.** *An* $\mathsf{Entropy}^{\uparrow}$ *log-partition BHist carrier is a finite source packet* $\mathcal{E}_{\log Z}$ *consisting of a* $\mathsf{Distribution}^{\uparrow}$ *pushforward row, a finite partition observation row, a log-weight endpoint row, an* $\mathsf{Integral}^{\uparrow}$ *or finite-sum readback row for the displayed partition, componentwise* hsame *transports for the distribution, partition, log-weight, and readback rows, finite* Cont *ledger routes connecting the observations, and* Pkg *provenance for the packet. A consumer reads a log-partition boundary only through these rows.*

Lean4 (definition) `BEDC.Derived.EntropyUp.EntropyLogPartitionBHistCarrier`
Lean4 (definition) `BEDC.Derived.EntropyUp.EntropyLogPartitionCarrier`

Theorem 192.3 (Entropy log-partition consumer exhaustion)**.** *Every accepted consumer of the carrier* $\mathcal{E}_{\log Z}$ *in Definition 192.2 reads exactly the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row, the finite partition observation row, the log-weight endpoint row, the* $\mathsf{Integral}^{\uparrow}$ *or finite-sum readback row, the componentwise* hsame *transports, the finite* Cont *ledger routes, and the* Pkg *provenance. It reads no analytic convergence row, no measure-completion row, and no host-level entropy equality.*

Proof. Project the finite carrier packet to its distribution, partition, log-weight, and readback coordinates. These are the only numeric observation rows displayed by the log-partition source.

Next project the componentwise hsame transports and finite Cont routes. Repacking those finite rows through Pkg supplies the consumer boundary and exhausts the visible carrier fields.

The packet has no field for analytic convergence, completed measures, or a host entropy identity. Such claims therefore cannot be read by a BEDC consumer of the log-partition carrier. □

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyLogPartitionCarrier consumer exhaustion`

Theorem 192.4 (Entropy log-partition transport closure)**.** *Let $\mathcal{E}_{\log Z}$ and $\mathcal{E}'_{\log Z}$ be two accepted* Entropy$^{\uparrow}$ *log-partition BHist carriers. If their* Distribution$^{\uparrow}$ *pushforward rows, finite partition rows, log-weight endpoint rows, and* Integral$^{\uparrow}$ *or finite-sum readback rows are related componentwise by the displayed* hsame *transports, then the transported packet exposes the same log-partition consumer boundary: the same partition readback, the same log-weight endpoint, the same finite* Cont *ledger routes, and the same* Pkg *provenance up to the displayed transport rows.*

The transport closure does not assert analytic convergence, measure completion, or host-level equality of entropy values.

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyLogPartitionBHistCarrier transport closure`

Proof. Begin with the exhaustion theorem for each accepted carrier. It reduces both consumer surfaces to the finite distribution, partition, log-weight, readback, transport, ledger, and provenance rows.

Use the displayed hsame rows component by component. They transport the distribution source, the partition observation, the log-weight endpoint, and the readback row without changing the finite ledger shape.

Finally repack the transported rows through the same Pkg provenance surface. Since analytic convergence, measure completion, and host entropy equality were excluded from the exhausted boundary, the transported packet is closed exactly at the finite log-partition surface. □

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyLogPartitionCarrier transport closure`

Theorem 192.5 (Entropy distribution-integral boundary)**.** *Consumers of the* Entropy$^{\uparrow}$ *source surface read exactly the* Distribution$^{\uparrow}$ *pushforward row, the* Integral$^{\uparrow}$ *readback row, the log-weight endpoint, the* hsame *transport rows, the* Cont *ledger rows, and the* Pkg *provenance from Definition 192.1. Analytic convergence, measure-theoretic completeness, and host-level entropy equality are not part of this boundary.*

Proof. Project the distribution row, integral readback row, and log-weight endpoint from the displayed BHist source surface. These are the only numerical observation rows exported by the package.

Next project the hsame transport rows and the Cont ledger rows. Repacking these rows through Pkg gives the consumer boundary, and no analytic completion claim can be derived without additional convergence or measure-theoretic rows. □

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyBHistMeasureSourceSurface distribution integral boundary`

192.2 The certificate

Theorem 192.6 (EntropyUp NameCert obligation surface)**.** *The* Entropy$^{\uparrow}$ *NameCert obligation surface for Definition 192.1 is the package containing the* Distribution$^{\uparrow}$ *pushforward row, the* Integral$^{\uparrow}$ *readback row, the certified log-weight endpoint, the* hsame *transport rows, the* Cont *ledger rows, and the* Pkg *provenance. It certifies the displayed BEDC source package without asserting analytic convergence or measure-theoretic completeness.*

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropySourceSurface namecert obligation surface`

Proof. Use Definition 192.5 to obtain the exact rows that consumers may inspect. The carrier field is the displayed log-weight BHist endpoint over the distribution and integral sources.

The dependency fields are the $\mathsf{Distribution}^{\uparrow}$ and $\mathsf{Integral}^{\uparrow}$ rows, the stability fields are the hsame transports, and the ledger fields are the Cont rows with their Pkg provenance. Repacking these finite rows supplies the NameCert obligation surface and leaves stronger analytic claims outside the certificate. □

Theorem 192.7 (Entropy log-weight transport determinacy)**.** *Let E and E' be two accepted* $\mathsf{Entropy}^{\uparrow}$ *source surfaces of the form displayed in Definition 192.1. If E and E' share the same* $\mathsf{Distribution}^{\uparrow}$ *pushforward row, the same* $\mathsf{Integral}^{\uparrow}$ *readback row, and the same transported log-weight endpoint through the displayed* hsame *rows, then every entropy consumer reads the same log-weight endpoint,* Cont *ledger rows, and* Pkg *provenance from E and E'.*

No entropy equality theorem, convergence theorem, measure completion, or host-level integral identity can distinguish the two packages within the displayed BEDC boundary.

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropySourceSurface log weight transport determinacy`

Proof. First project both packages through Definition 192.5. The only observable numerical rows are the shared $\mathsf{Distribution}^{\uparrow}$ pushforward row, the shared $\mathsf{Integral}^{\uparrow}$ readback row, and the transported log-weight endpoint carried by the displayed hsame rows.

Next project the certificate surface with Definition 192.6. Since the accepted surfaces have the same distribution, integral, and log-weight endpoints, the certificate can repack only the corresponding Cont ledger rows and the same Pkg provenance for the accepted source package.

Finally use the exclusions in the two boundary theorems. Analytic convergence, measure-theoretic completion, entropy equality, and host integral identity are not rows of the accepted $\mathsf{Entropy}^{\uparrow}$ surface. They therefore cannot supply a second consumer read or separate the two transported log-weight packages. □

Definition 192.8 (Entropy obligation inventory packet)**.** *The* $\mathsf{Entropy}^{\uparrow}$ *obligation inventory packet is the finite* BHist *packet consisting of the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row, the* $\mathsf{Integral}^{\uparrow}$ *readback row, the certified log-weight endpoint, the componentwise* hsame *transports for those three rows, the finite* Cont *ledgers carrying the displayed observations, and the* Pkg *provenance for the accepted entropy source. The carrier row is the accepted source surface of Definition 192.1; the classifier row is the componentwise comparison of the displayed distribution, integral, and log-weight endpoints; the stability and ledger rows are exactly the displayed* hsame, Cont, *and* Pkg *data.*

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyUpTasteGate single carrier alignment`

Theorem 192.9 (Entropy carrier-classifier obligation)**.** *Every accepted inventory packet of Definition 192.8 supplies the* $\mathsf{Entropy}^{\uparrow}$ *carrier row and the componentwise classifier obligations for the displayed* $\mathsf{Distribution}^{\uparrow}$ *pushforward row,* $\mathsf{Integral}^{\uparrow}$ *readback row, and certified log-weight endpoint. No analytic convergence row, measure-completion row, or host entropy equality participates in the carrier or classifier obligation.*

Proof. Project the inventory packet to its carrier row. By definition this row is the accepted source surface of Definition 192.1.

Next project the classifier coordinates. They are exactly the componentwise hsame transports on the distribution pushforward, integral readback, and log-weight endpoint, so they supply the classifier obligations named by the packet.

The remaining rows are continuation and provenance data. Since the inventory has no field for convergence, completed measures, or host entropy equality, those claims cannot enter the carrier-classifier obligation. □

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropyObligationInventoryPacket carrier classifier obligation`

Theorem 192.10 (Entropy ledger stability obligation)**.** *If an accepted* Entropy$^{\uparrow}$ *obligation inventory packet is transported componentwise along its displayed* hsame *rows, then the transported log-weight endpoint and transported* Integral$^{\uparrow}$ *readback row remain inside the same finite* Cont *ledger and* Pkg *provenance package. The transported package is again an* Entropy$^{\uparrow}$ *obligation inventory packet.*

Proof. Begin with Definition 192.7. It shows that transported packages with the same distribution, integral, and log-weight endpoints expose the same log-weight consumer boundary and provenance.

Apply Definition 192.6 to read the finite ledger rows of the transported package. The Cont rows and Pkg provenance are precisely the ledger fields listed in Definition 192.8.

Repack the transported distribution, integral, log-weight, hsame, Cont, and Pkg rows. This gives another accepted obligation inventory packet and keeps analytic convergence, measure completion, and host entropy equality outside the ledger. □

Lean4 ✓ `BEDC.Derived.EntropyUp.EntropySourceSurface ledger stability obligation`

Closure status for NameCert$_{\text{Entropy}^{\uparrow}}$.

Constructive story: An entropy source is assembled as a finite BHist packet whose displayed faces are a Distribution$^{\uparrow}$ pushforward row, an Integral$^{\uparrow}$ readback row, a certified log-weight endpoint, componentwise hsame transports, finite Cont ledgers, and Pkg provenance. The chapter binds these rows as the Entropy$^{\uparrow}$ obligation inventory while leaving analytic convergence, measure completion, and host entropy equality outside the packet.

Theory closure: TheoryGate(NameCert$_{\text{Entropy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 192.11 and Definition 192.12 bind the exported entropy readback bridge through the displayed BHist carrier, log-partition rows, transported log-weight and integral rows, componentwise classifier obligations, Cont ledgers, Pkg provenance, and consumer-exact boundary visible in this chapter.

Formal verification: FormalStatus(NameCert$_{\text{Entropy}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.EntropyUp.EntropySourceSurface\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: This chapter does not close convergence, measure completion, or host entropy equality.

Upgrade path: No further paper-axis theory closure remains; bridge verification remains separate if a machine bridge is later supplied.

Definition 192.11 (Entropy public readback bridge surface)**.** *The* Entropy$^{\uparrow}$ *public readback bridge surface is the conservative paper bridge whose BEDC side is the accepted source surface of Definition 192.1 together with the accepted log-partition carrier of Definition 192.2. Its bridge rows are exactly the displayed* Distribution$^{\uparrow}$ *pushforward row, finite partition observation row, transported log-weight endpoint row,* Integral$^{\uparrow}$ *or finite-sum readback row, componentwise* hsame *transport rows, finite* Cont *ledger rows,* Pkg *provenance, and the consumer-exact boundary supplied by Definition 192.3 and Definition 192.5.*

Theorem 192.12 (Entropy public readback bridge boundary)**.** *Let B be an* Entropy$^{\uparrow}$ *public readback bridge surface in the sense of Definition 192.11. Every conservative paper read through B factors through the displayed source carrier, the log-partition rows, the transported log-weight endpoint, the* Integral$^{\uparrow}$ *readback row, the* Cont *ledger rows, the* Pkg *provenance, and the consumer-exact boundary named in Definition 192.3 and Definition 192.5. Therefore B cannot import analytic convergence, measure completion, host entropy equality, or host integral equality.*

Proof. Project the source side of B. By Definition 192.11, this projection is the accepted source surface of Definition 192.1. Applying Definition 192.5 exhausts its public reads: a consumer sees only the $\mathsf{Distribution}^{\uparrow}$ pushforward row, the $\mathsf{Integral}^{\uparrow}$ readback row, the log-weight endpoint, the hsame transport rows, the Cont ledger rows, and the Pkg provenance.

Project the log-partition side of B. The same definition identifies this projection with the accepted carrier of Definition 192.2. Applying Definition 192.3 exhausts that side as the $\mathsf{Distribution}^{\uparrow}$ pushforward row, finite partition observation row, log-weight endpoint row, $\mathsf{Integral}^{\uparrow}$ or finite-sum readback row, componentwise hsame transports, finite Cont ledger routes, and Pkg provenance. Applying Definition 192.4 keeps those rows stable under the displayed componentwise transports and adds no analytic row.

It remains to check that the bridge has no second channel. The definition of the bridge surface lists its bridge rows as the union of the two exhausted consumer boundaries, with the transported log-weight and readback rows included only through their displayed hsame, Cont, and Pkg provenance. Thus any conservative paper read through B first lands in one of the rows already named by the source or log-partition exhaustion theorem, and reverse admission can accept only a package carrying those same rows.

Analytic convergence and measure completion are excluded by both the source boundary and the log-partition exhaustion. Host entropy equality is excluded by Definition 192.3, Definition 192.4, and Definition 192.5. Host integral equality is also excluded: the only integral information admitted by the bridge is the public $\mathsf{Integral}^{\uparrow}$ readback row, and Definition 123.4 keeps host function equality and hidden convergence choices outside the public $\mathsf{Integral}^{\uparrow}$ surface. Consequently the bridge boundary is exactly the displayed entropy readback surface and cannot import any of the four excluded host claims. □

Chapter 193

A Concrete Naming Certificate for $\mathsf{Ode}^{\uparrow}$

The ordinary-differential-equation interface $\mathsf{Ode}^{\uparrow}$ packages a system $\dot{x} = F(t, x)$ together with the local existence and uniqueness data on Lipschitz vector fields. It depends on $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ (chapter 122) and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90).

193.1 Carrier definition

Definition 193.1 (OdeUp BHist local-flow source)**.** *An* $\mathsf{Ode}^{\uparrow}$ *BHist local-flow source is a time-state carrier over the* $\mathsf{Banach}^{\uparrow}$ *carrier supplied by* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$*. Its points are pairs of histories* $(t, x) : \mathsf{BHist} \times \mathsf{BHist}$ *whose time component is carried by the scalar surface used by* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *and whose state component is carried by the Banach carrier. The vector-field row is a displayed operation*

$$F : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{BHist}$$

whose value $F(t, x)$ *is again a Banach-carried state history after transport along the time and state classifiers.*

A local-flow row consists of an initial time cell t_0*, an initial state cell* x_0*, an endpoint time cell* t_1*, an endpoint state cell* x_1*, a vector-field evaluation ledger* v*, a Picard-step history* p*, and a limit witness* ℓ *such that the step and limit are linked by displayed continuations*

$$\mathsf{Cont}(t_0, x_0, p), \qquad \mathsf{Cont}(p, v, \ell), \qquad \mathsf{Cont}(t_1, \ell, x_1).$$

The classifier is inherited from the Banach and derivative surfaces: two rows are equivalent when their time cells are identified by the derivative scalar classifier, their state cells by the Banach carrier classifier, and their displayed ledgers by hsame *transport through the same* NameCert *rows. The ledger records the time cell, state cell, vector-field evaluation, Picard-step continuation route, and limit witness. It does not assert equality in any host function space.*

Lean4 (definition) `BEDC.Derived.OdeUp.OdeLocalFlowRow`

193.2 The certificate

Theorem 193.2 (OdeUp root vector-field carrier stability)**.** *For an* $\mathsf{Ode}^{\uparrow}$ *BHist local-flow source, if* t *is carried by the derivative scalar surface and* x *is carried by the Banach state carrier, then*

the displayed vector-field value $F(t,x)$ is carried by the same Banach state carrier. This assignment is stable under classifier transport: replacing t and x by t' and x' with the corresponding $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *classifier witnesses gives* $F(t,x)\mathsf{hsame}F(t',x')$ *in the endpoint classifier recorded by* NameCert.

Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlowSource vector field carrier stability`

Proof. Unpack the local-flow source. The time cell is read through $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$, and the state cell is read through $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$. The vector-field row is part of the source data, so its evaluation ledger records a Banach-carried output whenever both input cells are carried.

For transport, move first along the derivative classifier on the time cell and then along the Banach carrier classifier on the state cell. The displayed evaluation ledger is functorial for these two classifier moves because it is stored as a NameCert row rather than as host equality. Repacking the transported output row gives the required Banach-carried value and its hsame endpoint classifier. □

Theorem 193.3 (OdeUp root Picard continuation scope)**.** *In an* $\mathsf{Ode}^{\uparrow}$ *BHist local-flow source, Picard-step packages compose inside the scope generated by* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$, *and* hsame *transport. If two adjacent Picard ledgers expose* $\mathsf{Cont}(t_0,x_0,p)$, $\mathsf{Cont}(p,v,\ell)$, *and* $\mathsf{Cont}(t_1,\ell,x_1)$ *together with a second compatible step from x_1 to x_2, then associativity of* Cont *builds the composite continuation route from (t_0,x_0) to (t_2,x_2) without adding any carrier outside those two certificate surfaces.*

Proof. Read the first Picard package as a chain of displayed continuations from the initial time-state cell to its limit witness and then to its endpoint. Read the second package in the same way, with the endpoint state of the first row as the carried input state of the second.

Use Cont associativity to reassociate the terminal continuation of the first package with the initial continuation of the second package. Whenever a shared endpoint is represented by a classifier-equal history rather than by the same syntax, transport across hsame using the Banach and derivative rows of $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$.

The resulting chain is again a list of BHist continuations whose endpoints are classified by the same NameCert surfaces. Thus the composite Picard route remains scoped to Banach states, derivative time cells, and hsame transport; no ambient function-space carrier is introduced. □

Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlowSource picard continuation scope`
Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlowRow picard continuation scope`

Theorem 193.4 (OdeUp root local-uniqueness classifier)**.** *For an* $\mathsf{Ode}^{\uparrow}$ *BHist local-flow source, two local-flow witnesses with the same initial time cell, the same initial state cell, and the same Lipschitz ledger are classified by the same endpoint classifier. Concretely, if their vector-field evaluation rows, Picard-step rows, and limit witnesses are transported through the shared* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, Cont, *and* hsame *ledgers, then their endpoint state histories are related by the Banach classifier recorded in* NameCert.

Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlow endpoint classifier deterministic`

Proof. Unpack the two local-flow witnesses into their common initial time-state cell, their vector-field evaluation ledgers, Picard-step histories, limit witnesses, and endpoint histories. The shared Lipschitz ledger classifies the vector-field rows through the derivative and Banach input classifiers, so the two Picard steps have the same continuation data up to hsame.

Transport the second Picard chain along these hsame witnesses and use determinism of the displayed Cont result at each continuation stage. This identifies the two limit witnesses in the classifier inherited from the Banach carrier.

Finally transport through the endpoint continuation rows $\mathsf{Cont}(t_1, \ell, x_1)$ and $\mathsf{Cont}(t_1, \ell', x_1')$. Since the time endpoint is the same cell and the limit witnesses have just been classified together, the Banach endpoint classifier relates x_1 and x_1'. This is precisely the endpoint classifier exposed by the $\mathsf{Ode}^{\uparrow}$ source through $\mathsf{NameCert}$. □

Theorem 193.5 (OdeUp local-flow concatenation endpoint determinacy)**.** *Let R_{01} and R_{12} be $\mathsf{Ode}^{\uparrow}$ BHist local-flow rows over the same $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ surfaces:*

$$R_{01} = (t_0, x_0, t_1, x_1; v_{01}, p_{01}, \ell_{01}), \qquad R_{12} = (t_1, x_1, t_2, x_2; v_{12}, p_{12}, \ell_{12}).$$

Assume the terminal time-state cell of R_{01} is the initial time-state cell of R_{12} in the derivative and Banach classifiers, so Definition 193.3 supplies a composite Picard route. Write the resulting composite local-flow row as

$$R_{02}^{\circ} = (t_0, x_0, t_2, x_2; v_{02}^{\circ}, p_{02}^{\circ}, \ell_{02}^{\circ}).$$

If

$$D_{02} = (t_0, x_0, t_2, y; v_{02}, p_{02}, \ell_{02})$$

is any direct $\mathsf{Ode}^{\uparrow}$ local-flow row whose Lipschitz vector-field ledger is classified with the composite ledger,

$$v_{02}^{\circ} \mathsf{hsame} v_{02},$$

then the concatenated endpoint and the direct endpoint are related by the Banach endpoint classifier:

$$x_2 \sim_{\mathsf{Banach}^{\uparrow}} y,$$

where $\sim_{\mathsf{Banach}^{\uparrow}}$ denotes the classifier supplied by $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlowSource concatenation route scope`

Proof. By Definition 193.3, the two adjacent Picard ledgers compose through the shared middle time-state cell inside the same $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ surfaces. Thus the composite row R_{02}° is a local-flow witness from the initial data (t_0, x_0) to the endpoint state x_2 at time t_2.

Unpacking Definition 193.1, the composite row exposes

$$\mathsf{Cont}(t_0, x_0, p_{02}^{\circ}), \qquad \mathsf{Cont}(p_{02}^{\circ}, v_{02}^{\circ}, \ell_{02}^{\circ}), \qquad \mathsf{Cont}(t_2, \ell_{02}^{\circ}, x_2).$$

The direct row exposes the parallel chain

$$\mathsf{Cont}(t_0, x_0, p_{02}), \qquad \mathsf{Cont}(p_{02}, v_{02}, \ell_{02}), \qquad \mathsf{Cont}(t_2, \ell_{02}, y).$$

The two witnesses have the same initial time cell, the same initial state cell, the same endpoint time cell, and the displayed Lipschitz vector-field ledger comparison $v_{02}^{\circ} \mathsf{hsame} v_{02}$. Therefore Definition 193.4 applies to the composite row and the direct row. It classifies their endpoint state histories by the Banach classifier carried by $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$. Those endpoint histories are exactly x_2 and y, giving $x_2 \sim_{\mathsf{Banach}^{\uparrow}} y$. □

Lean4 ✓ `BEDC.Derived.OdeUp.OdeLocalFlowRow concatenation endpoint determinacy`

Theorem 193.6 (OdeUp local-flow concatenation associativity)**.** *Let*

$$R_{01} = (t_0, x_0, t_1, x_1; v_{01}, p_{01}, \ell_{01}), \qquad R_{12} = (t_1, x_1, t_2, x_2; v_{12}, p_{12}, \ell_{12}), \qquad R_{23} = (t_2, x_2, t_3, x_3; v_{23}, p_{23}, \ell_{23})$$

be $\mathsf{Ode}^{\uparrow}$ *BHist local-flow rows over the same* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *surfaces. Assume the terminal time-state cell of* R_{01} *is classified with the initial time-state cell of* R_{12}*, and the terminal time-state cell of* R_{12} *is classified with the initial time-state cell of* R_{23}*, in the derivative and Banach classifiers. Let*

$$R^{\circ}_{02} = (t_0, x_0, t_2, y_{02}; v^{\circ}_{02}, p^{\circ}_{02}, \ell^{\circ}_{02}), \qquad R^{\circ}_{13} = (t_1, x_1, t_3, y_{13}; v^{\circ}_{13}, p^{\circ}_{13}, \ell^{\circ}_{13})$$

be the binary composites supplied by Definition 193.3. Let

$$L^{\circ}_{03} = (t_0, x_0, t_3, z_L; v_L, p_L, \ell_L)$$

be the composite of R°_{02} *with* R_{23} *supplied by the same theorem, and let*

$$R^{\circ}_{03} = (t_0, x_0, t_3, z_R; v_R, p_R, \ell_R)$$

be the composite of R_{01} *with* R°_{13} *supplied by the same theorem. Then the two grouped composites have Banach-classified endpoint state histories:*

$$z_L \sim_{\mathsf{Banach}^{\uparrow}} z_R.$$

Proof. Apply Definition 193.3 first to R_{01} and R_{12} and separately to R_{12} and R_{23}. This produces the two binary rows R°_{02} and R°_{13} inside the same $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ surfaces. Applying the same theorem again to the adjacent pairs (R°_{02}, R_{23}) and (R_{01}, R°_{13}) produces the two grouped rows L°_{03} and R°_{03}, both read as local-flow witnesses from the same initial time-state cell (t_0, x_0) to the same terminal time cell t_3.

Unpack Definition 193.1. Each grouped row is built from exactly the three displayed local-flow packages of R_{01}, R_{12}, and R_{23} together with the two classifier transports at the shared interfaces. The only difference is the parenthesization of the continuation chain. By Definition 4.56, reassociating the left-grouped chain and the right-grouped chain does not change the resulting continuation witness up to hsame. Therefore the direct row R°_{03} has its Lipschitz vector-field ledger classified with the outer composite ledger of L°_{03}:

$$v_L \mathsf{hsame} v_R.$$

Now apply Definition 193.5 to the adjacent pair R°_{02}, R_{23}, taking R°_{03} as the direct row from (t_0, x_0) to time t_3. Its hypotheses are exactly the shared initial data, the certified outer concatenation route, and the displayed ledger comparison $v_L \mathsf{hsame} v_R$. The conclusion of that theorem is precisely that the endpoint state history of the grouped composite $(R_{01} \circ R_{12}) \circ R_{23}$ is related by the Banach endpoint classifier to the endpoint state history of the grouped composite $R_{01} \circ (R_{12} \circ R_{23})$. Hence

$$z_L \sim_{\mathsf{Banach}^{\uparrow}} z_R.$$

□

Closure status for $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$**.**

Constructive story: An ordinary differential equation local-flow row is carried by a time-state source over $\mathsf{Derivative}^{\uparrow}$ and $\mathsf{Banach}^{\uparrow}$: a time cell, state cell, vector-field evaluation, Picard-step history, limit witness, and endpoint state are tied together by three displayed continuation rows. A row is admitted only when the time component is carried by the

derivative scalar surface, the state and vector-field value are carried by the Banach surface, and the Picard and limit ledgers are present; equality in a host function space, global flow data, and Banach fixed-point theory are not carrier fields. Two local-flow rows are classified by transporting time cells through the derivative classifier, state cells through the Banach classifier, and Picard, vector-field, limit, and endpoint ledgers through hsame and Cont determinacy, rather than by comparing host solutions. The source covers the local time-state carrier, vector-field row, Picard continuations, limit witness, endpoint row, and Lipschitz ledger; the pattern records vector-field carrier stability, Picard route composition, local endpoint determinacy, and concatenation endpoint comparison; the ledger keeps the three continuation routes and their classifier transports visible. The five name-certificate obligations are obligation-level rows here: upstream derivative and Banach certificates supply carried endpoints, hsame gives reflexive, symmetric, and transitive movement, and Picard continuation scope repacks transported or concatenated routes as the same kind of local-flow row. A history h lies in the signature gap of an $\mathsf{Ode}^{\uparrow}$ token p when the source-domain accepts this local-flow package, the signature relation pairs it with an answer flow row and evidence ledger, and token introduction names that answer as p; unfolded, the initial time-state cell, vector-field ledger, Picard chain, limit witness, endpoint time, and Banach-classified endpoint are the data encoded by p. The chapter registers this local Picard-continuation construction under the name $\mathsf{Ode}^{\uparrow}$; it records a scoped obligation boundary and does not introduce global existence, smooth flow groups, host solution-space equality, standard ODE models, or bridge verification.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature local-flow composition treatment records the BHist local-flow source carrier of Definition 193.1, the Picard continuation scope of Definition 193.3, endpoint classifier determinacy from Definition 193.4, concatenation route scope from Definition 193.5, endpoint determinacy for composed rows over the derivative and Banach NameCert surfaces, and the associativity theorem Definition 193.6.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.OdeUp.OdeLocalFlowRow\_concatenation\_endpoint\_determinacy`

Bridge status: paperBridge

Not claimed: This obligation boundary does not close global existence, Banach fixed-point theory, smooth flow groups, standard ODE models, or bridge verification.

Upgrade path: The remaining formal step is bridgeCheckedV when a machine-checked standard bridge is present.

Chapter 194

A Concrete Naming Certificate for $\mathsf{Pde}^{\uparrow}$

The partial-differential-equation interface $\mathsf{Pde}^{\uparrow}$ packages a relation among partial derivatives of a function on a manifold carrier together with classification by order, type, and boundary data. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ (chapter 122).

194.1 Carrier definition

Definition 194.1 (PdeUp carried source row)**.** *A carried source row for* $\mathsf{Pde}^{\uparrow}$ *consists of the following BEDC-facing data: a* $\mathsf{Manifold}^{\uparrow}$ *carrier row naming the underlying charted carrier; the* $\mathsf{Derivative}^{\uparrow}$ *partial-derivative observation rows whose endpoints are the named derivative observations used by the equation; a finite PDE relation row whose entries refer only to those derivative endpoints together with their order and type tags; a boundary-data classifier row over the same carrier and endpoint names; and a ledger recording hidden chart choices, coordinate choices, representative functions, and solver-side presentation data.*

The carried source row does not identify host functions, host real values, analytic solution spaces, or coordinate representatives. Those objects remain ledger entries unless they are re-expressed through the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, *relation, and boundary rows.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow source scope`

Theorem 194.2 (PdeUp carrier obligation)**.** *A carried* $\mathsf{Pde}^{\uparrow}$ *endpoint is a* BHist *row whose visible source data are exactly the* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *source, the* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *source, the PDE relation row, the boundary-data classifier row,* Pkg *provenance, and the probe bundles used to read the relation. Chart coordinates, derivative witnesses, and boundary presentations are ledger data for the certificate rather than additional carrier-equality coordinates.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow carrier obligation surface`

Proof. Read the endpoint through the ten-field certificate template. The traditional target is a partial-differential-equation endpoint on a certified manifold, while the BEDC source is the displayed BHist package assembled from the manifold certificate, derivative certificate, relation row, boundary classifier row, provenance, and probe bundles.

The carrier field therefore projects only those visible histories. The coordinate charts and derivative witnesses enter through the stability and ledger fields: they justify that the displayed

relation and boundary rows can be transported across the selected presentations, but they are not compared as carrier coordinates.

Exactness is readback from the same package. Any accepted carrier row must be one of the displayed BHist rows, and any hidden chart, derivative order, or boundary presentation is recoverable only as ledger evidence. □

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow carrier obligation`

Theorem 194.3 (PdeUp classifier obligation)**.** *The* $\mathsf{Pde}^{\uparrow}$ *classifier compares two carried endpoints exactly when the manifold source row, derivative-tuple row, PDE relation row, and boundary-data classifier row agree under the same* hsame *discipline and the same* Pkg *provenance. It does not compare endpoints by host function equality.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeRelationClassifier visible append surface`

Proof. Project the classifier slot from the ten-field template. Its source rows are the same source, carrier, classifier, stability, ledger, theorem-list, exactness, bridge, and boundary fields used by the carrier obligation.

The source field fixes the manifold row and the derivative tuple row. The carrier and classifier fields then compare the PDE relation and boundary classifier rows by hsame and by the provenance equality recorded in Pkg. Since each comparison is a comparison of displayed BHist rows, no host-level function equality is available as an extra rule.

Conversely, if these four rows and the provenance agree under the displayed hsame witnesses, the classifier field has exactly the data it requires. Repacking those witnesses gives the carried-endpoint comparison. □

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow classifier obligation`

Theorem 194.4 (PdeUp stability ledger exactness obligation)**.** *The* $\mathsf{Pde}^{\uparrow}$ *certificate is admitted only with transport stability for the derivative relation and boundary data, ledger entries for hidden chart/order/boundary choices, exactness rows for coverage, soundness, and completeness, and dependency rows citing the* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *sources.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow stability ledger exactness obligation surface`
Lean4 ✓ `BEDC.Derived.PdeUp.PdeStabilityLedger relation boundary append surface`

Proof. Separate the remaining certificate fields after the carrier and classifier rows have been fixed. Stability requires transport of the derivative-relation row and the boundary-data row along the displayed hsame comparisons of the manifold and derivative sources.

The ledger field records every chart, derivative-order, and boundary presentation choice used to witness that transport. These entries are evidence rows over BHist; they do not enlarge the carrier and do not create a second classifier.

The exactness field splits into coverage, soundness, and completeness rows: every admitted endpoint has displayed source and probe data, every accepted relation or boundary row is supported by the ledger, and every required source dependency cites $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$. These rows are precisely the non-bridge obligation surface claimed here. □

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow stability ledger exactness obligation`

194.2 The certificate

Theorem 194.5 (PdeUp carrier source obligation)**.** *For any carried source row of* $\mathsf{Pde}^{\uparrow}$*, the carrier obligation is discharged exactly by the named* $\mathsf{Manifold}^{\uparrow}$ *row, the named* $\mathsf{Derivative}^{\uparrow}$ *partial-derivative observation rows, the finite PDE relation row on those derivative endpoints, and the boundary-data classifier row. No host function carrier, host coordinate equality, analytic existence predicate, or solver object is part of the carrier source.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow visible source readback`

Proof. Unpack Definition 194.1. Its source entries are the $\mathsf{Manifold}^{\uparrow}$ carrier row, the $\mathsf{Derivative}^{\uparrow}$ observation rows, the relation row, and the boundary row.

The remaining listed host objects occur only in the ledger component of the same definition. Hence they can be recorded for provenance, but they cannot act as carrier-source dependencies for the $\mathsf{Pde}^{\uparrow}$ certificate. □

Theorem 194.6 (PdeUp relation classifier obligation)**.** *Let two* $\mathsf{Pde}^{\uparrow}$ *carried source rows have* $\mathsf{Manifold}^{\uparrow}$ *carrier rows related by the supplied* $\mathsf{Manifold}^{\uparrow}$ *classifier, and let each partial-derivative endpoint appearing in their PDE relation rows be related by the supplied* $\mathsf{Derivative}^{\uparrow}$ *classifier. Then the PDE relation rows are transported by replacing each relation entry with its transported derivative endpoint, order tag, and type tag; no further host equation equality is required.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeRelationClassifier endpoint transport`

Proof. The relation row in Definition 194.1 is finite and refers only to displayed derivative endpoints with their order and type tags. Transport the carrier name by the $\mathsf{Manifold}^{\uparrow}$ classifier and transport every endpoint name by the corresponding $\mathsf{Derivative}^{\uparrow}$ classifier.

After these transports, each relation entry has the same BEDC-facing shape as before: a finite relation entry over named derivative endpoints. Since host equations and coordinate representatives are ledger entries rather than relation entries, no host equation equality is introduced. □

Theorem 194.7 (PdeUp boundary-data obligation)**.** *For any* $\mathsf{Pde}^{\uparrow}$ *carried source row, boundary-data rows are transported by the same* $\mathsf{Manifold}^{\uparrow}$ *carrier classifier and* $\mathsf{Derivative}^{\uparrow}$ *endpoint classifiers used for the PDE relation row. The boundary-data obligation adds no host boundary equality and no host trace-space equality beyond those carried classifier rows.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeBoundaryData relation transport surface`
Lean4 ✓ `BEDC.Derived.PdeUp.PdeRelationClassifier boundary transport surface`

Proof. Unpack the boundary-data classifier row from Definition 194.1. Its carrier reference is the same $\mathsf{Manifold}^{\uparrow}$ carrier row, and its derivative references are among the displayed $\mathsf{Derivative}^{\uparrow}$ endpoint rows.

Therefore the boundary row uses the same two classifier families as the relation row: carrier transport from $\mathsf{Manifold}^{\uparrow}$ and endpoint transport from $\mathsf{Derivative}^{\uparrow}$. Host boundary representatives and trace-space presentations remain ledger data, so they do not become additional equality obligations. □

Theorem 194.8 (PdeUp ledger exactness obligation)**.** *The exactness data of a* $\mathsf{Pde}^{\uparrow}$ *carried source row is precisely the pair consisting of the carried PDE relation classifier and the carried boundary-data classifier. Host functions, host real equality, analytic existence of solutions, regularity theorems, numerical solvers, and solver data are ledger-external to this exactness claim.*

Lean4 ✓ `BEDC.Derived.PdeUp.PdeCarriedSourceRow ledger exactness surface`

Proof. The only classifier rows displayed in Definition 194.1 are the finite PDE relation classifier and the boundary-data classifier. Exactness is therefore the assertion that these two rows account for all BEDC-facing relation and boundary comparisons in the certificate.

Every analytic or solver-side object named in the statement is listed in the ledger component of the carried source row. Because ledger entries record presentation choices without adding classifier rows, they remain outside the exactness pair. □

Theorem 194.9 (PdeUp public interface export)**.** *The public* $\mathsf{NameCert}_{\mathsf{Pde}^\uparrow}$ *surface is exactly the* BHist *carried source row of Definition 194.1 together with the* $\mathsf{Manifold}^\uparrow$ *carrier row, the* $\mathsf{Derivative}^\uparrow$ *partial-derivative observation rows, the* Pkg *provenance row, the* ProbeBundle *rows used to read the finite PDE relation, and the following obligation surface:*

1. *the carrier obligation of Definition 194.2;*
2. *the classifier obligation of Definition 194.3;*
3. *the stability-ledger-exactness obligation of Definition 194.4;*
4. *the carrier-source obligation of Definition 194.5;*
5. *the relation classifier obligation of Definition 194.6;*
6. *the boundary-data obligation of Definition 194.7;*
7. *the ledger exactness obligation of Definition 194.8.*

No host function equality, host coordinate equality, analytic existence theorem, solution space, solver object, host real equality, regularity theorem, or numerical solver datum is part of this public interface.

Lean4 ✓ `BEDC.Derived.PdeUp.PdePublicInterface export semantic name certificate`

Proof. Start with the carried source row of Definition 194.1. Its visible BHist entries bind the $\mathsf{Manifold}^\uparrow$ carrier row, the $\mathsf{Derivative}^\uparrow$ derivative-observation rows, the finite PDE relation row, the boundary-data classifier row, Pkg provenance, and the ProbeBundle rows used to read the relation.

Repack those entries through the ten-field certificate template. The carrier, classifier, stability, ledger, theorem-list, exactness, bridge, and boundary slots are supplied by the seven cited obligation theorems, with the standard bridge slot absent and the dependency rows pointing back to the $\mathsf{Manifold}^\uparrow$ and $\mathsf{Derivative}^\uparrow$ source certificates.

Conversely, any public $\mathsf{NameCert}_{\mathsf{Pde}^\uparrow}$ endpoint must project through the same ten fields. The carrier and classifier projections recover the named source rows, the stability and ledger projections recover only transport and presentation evidence, and the exactness projection recovers the carried relation and boundary classifiers. Therefore the displayed surface is exact.

The excluded host objects occur only as ledger-side presentation data or analytic objects outside the BEDC-facing relation and boundary rows. Since no cited obligation introduces a classifier for them, they cannot enter the public $\mathsf{NameCert}_{\mathsf{Pde}^\uparrow}$ interface. □

Theorem 194.10 (PdeUp conservative standard bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{Pde}^\uparrow}$ *surface of Definition 194.9 has a conservative standard interpretation whose source is exactly the displayed* BHist *carried source row, the* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *carrier row, the* $\mathsf{NameCert}_{\mathsf{Derivative}^\uparrow}$ *partial-derivative observation rows, the finite PDE relation row, the boundary-data classifier row, the* Pkg *provenance row, and the* ProbeBundle *read rows. Conversely, any standard PDE presentation admitted through*

precisely those displayed manifold, derivative, relation, boundary, package, and probe rows repacks to the same public BEDC surface.

This bridge adds no analytic existence theorem, solution space, solver semantics, host function equality, host coordinate equality, host real equality, or regularity theory.

Proof. Apply Definition 194.9. It exposes the carried source row and the seven obligation rows: carrier, classifier, stability-ledger-exactness, carrier-source, relation-classifier, boundary-data, and ledger exactness. The standard reading is allowed to inspect only the displayed $\mathsf{Manifold}^{\uparrow}$ carrier row, $\mathsf{Derivative}^{\uparrow}$ partial-derivative rows, finite PDE relation row, boundary-data classifier row, Pkg provenance, and $\mathsf{ProbeBundle}$ read rows.

The ledger exactness theorem Definition 194.8 keeps chart selections, derivative-order witnesses, boundary presentations, representative functions, and solver-side presentation choices as provenance. Thus the standard interpretation is conservative: it reads the ordinary PDE presentation through the visible rows and does not turn hidden ledger choices into BEDC carrier coordinates or classifier clauses.

For the converse direction, start with a standard PDE presentation admitted through exactly those displayed rows. Repack its manifold carrier, derivative observations, relation entries, boundary classifier, package provenance, and probe reads into the carried source row of Definition 194.1. The seven obligations cited by Definition 194.9 then reconstruct the same public $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$ surface.

The excluded analytic and host-level objects remain outside the bridge because no row in the exported surface supplies classifiers for them. Hence the bridge certifies only conservative readback and repacking for the displayed PDE certificate surface. □

Lean4 ✓ `BEDC.Derived.PdeUp.PdeUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$.

Constructive story: A partial differential equation is carried here as a displayed row over already certified manifold and derivative data: the manifold carrier names the charted space, the derivative rows name the partial-derivative endpoints, and a finite relation row together with a boundary-data row says which endpoints are being compared. A row is admitted only when those visible pieces, the package provenance, and the probe bundles used to read the relation are present; chart choices, coordinate representatives, derivative witnesses, solver presentations, and host functions stay in the ledger, not in the carrier. Two carried endpoints are classified by transporting the manifold row, derivative tuple, finite relation row, boundary row, and provenance through the same hsame and upstream NameCert comparisons, rather than by equality of host equations or functions. The source therefore covers exactly the displayed manifold, derivative, relation, boundary, provenance, and probe rows; the pattern is the finite PDE-and-boundary shape on those derivative endpoints, and the ledger keeps hidden chart, order, representative, and boundary presentations visible only as evidence. The five name-certificate obligations are discharged because the upstream $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{Derivative}^{\uparrow}$ rows supply admitted endpoints, hsame gives reflexive, symmetric, and transitive classifier movement, and every transported relation or boundary row repacks as the same kind of carried source row. A history h lies in the signature gap of a $\mathsf{Pde}^{\uparrow}$ token p when the source-domain accepts this displayed PDE row, the signature relation pairs it with an answer row and evidence ledger, and token introduction names that answer as p; unfolded, the finite derivative relation and boundary classifier over the certified manifold must be the data encoded by p. The conservative standard bridge reads the visible rows as an ordinary PDE presentation and reads such a presentation back only when it uses precisely those rows, so the chapter registers this construction under the name $\mathsf{Pde}^{\uparrow}$ with no analytic existence theorem, solution space, solver semantics, or host-level function equality introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public paper-side surface is Definition 194.9, and the conservative standard bridge is Definition 194.10: the carried source row, the $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ carrier row, the $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ partial-derivative observation rows, the finite PDE relation row, the boundary-data classifier row, the Pkg provenance row, the ProbeBundle rows used to read the relation, and the seven public obligation rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.PdeUp.PdeUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim analytic existence, solution spaces, solver semantics, regularity theory, host function equality, host coordinate equality, host real equality, or Lean verification.

Upgrade path: No further upgrade – chapter at top of formal axis.

Theorem 194.11 (PdeUp finite relation subrow projection)**.** *Let S be a carried source row for* $\mathsf{Pde}^{\uparrow}$*. Write its displayed source data as a* $\mathsf{Manifold}^{\uparrow}$ *carrier row M, a finite row D of* $\mathsf{Derivative}^{\uparrow}$ *partial-derivative endpoints, a finite PDE relation row R, a boundary-data classifier row B, a* Pkg *provenance row P, and* ProbeBundle *rows Q reading the entries of R. Let R' be a displayed finite subrow of R, obtained by selecting relation entries from R, and suppose that each selected entry names only derivative endpoints occurring in D and carries only the order tags and type tags already displayed in R. Let Q' be the displayed restriction of Q to the reads of the selected entries. Then R' is again admitted as the relation coordinate of a carried* $\mathsf{Pde}^{\uparrow}$ *source row over the same M, D, B, P, and inherited probe-source rows Q'. This projection adds no host function equality, host coordinate equality, analytic solution predicate, trace-space equality, or solver object.*

Proof. Unpack the carried source row in Definition 194.1. The relation coordinate is finite, and every one of its entries is required to refer only to displayed $\mathsf{Derivative}^{\uparrow}$ endpoints together with displayed order and type tags. Since R' is obtained by selecting entries of R, no selected relation entry is created outside the original finite row. The endpoint, order, and type conditions for R' are therefore inherited entry by entry from the corresponding entries of R.

The source coordinates M, D, B, and P are not altered by the selection. The carrier-source obligation in Definition 194.5 says that these displayed manifold, derivative, relation, and boundary rows are exactly the carrier-source data for the certificate. Replacing the relation coordinate by the selected finite subrow keeps the same manifold carrier, the same derivative endpoint list, the same boundary-data classifier row, and the same package provenance.

The relation classifier obligation in Definition 194.6 permits relation rows only through their displayed derivative endpoints, order tags, and type tags. Here the transport is the identity on the retained entries: each selected entry keeps its original endpoint names and tags. Thus the selected relation row has the same BEDC-facing relation shape as the original row, with no appeal to host equation equality.

For the boundary coordinate, Definition 194.7 uses the same $\mathsf{Manifold}^{\uparrow}$ carrier classifier and $\mathsf{Derivative}^{\uparrow}$ endpoint classifiers as the relation row. Because the selected relation entries use only endpoints already present in D, the boundary-data classifier row remains compatible with the selected relation subrow. Restricting Q to Q' similarly changes only which displayed relation reads are consumed; it introduces no new probe source.

Finally apply the public export theorem Definition 194.9. The exported surface consists only of the carried source row, the manifold row, the derivative rows, the finite relation row, the boundary row, Pkg provenance, and the probe reads. The projected packet has exactly those same kinds of rows, with R' and Q' in place of the original relation reads. Since none of the cited obligations exports host functions, host coordinates, analytic solution data, trace-space equality, or solver objects, the projected packet exports none of them either. □

Lean4 ✓ `BEDC.Derived.PdeUp.PdeTasteGate single carrier alignment`

Chapter 195

A Concrete Naming Certificate for $\mathsf{DynSystem}^{\uparrow}$

The dynamical-system interface $\mathsf{DynSystem}^{\uparrow}$ packages a one-parameter family of self-maps of a phase-space carrier satisfying the flow axioms. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ (chapter 193).

195.1 Carrier definition

Definition 195.1 (DynSystemUp BHist flow carrier)**.** *A* $\mathsf{DynSystem}^{\uparrow}$ *flow packet is a displayed finite* BHist *tuple*

$$\mathcal{D} = (\mathcal{M}, \mathcal{O}, p, t, x, y, \phi, \omega, \rho, \lambda)$$

where $\mathcal{M}$ *is a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase-space packet,* $\mathcal{O}$ *is a* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *local-flow source packet,* $p : \mathsf{BHist}$ *is the phase carrier row,* $t : \mathsf{BHist}$ *is the elapsed-time row,* $x : \mathsf{BHist}$ *is the source point row,* $y : \mathsf{BHist}$ *is the target point row,* $\phi : \mathsf{BHist}$ *is the flow endpoint row,* $\omega : \mathsf{BHist}$ *is the ODE dependency witness read from* $\mathcal{O}$*,* ρ *records the* Pkg *provenance of the manifold and ODE dependency packets, and* λ *is the finite* Cont *ledger connecting the source point, elapsed time, local-flow row, and endpoint row.*

The packet is accepted only when the phase carrier and point rows are classified by the $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *surface, the local-flow source row is classified by the* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *surface, and the endpoint row* ϕ *is reached from* (t, x) *by the displayed* Cont *ledger. Two accepted packets classify the same* $\mathsf{DynSystem}^{\uparrow}$ *flow when their phase carrier, time row, source row, target row, ODE source witness, and flow endpoint row are transported by the inherited* hsame *comparisons from* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *while retaining the same* Pkg *provenance.*

Lean4 (definition) `BEDC.Derived.DynSystemUp.DynSystemFlowPacket`

Theorem 195.2 (DynSystemUp carrier flow surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase carrier and an* $\mathsf{Ode}^{\uparrow}$ *BHist local-flow source over that carrier. A* $\mathsf{DynSystem}^{\uparrow}$ *carrier cell is a* BHist *history whose displayed rows consist of the phase-carrier cell, the starting time cell, the starting state cell, the endpoint time cell, the endpoint state cell, the vector-field evaluation ledger, the Picard ledger, and the endpoint ledger of the corresponding* $\mathsf{Ode}^{\uparrow}$ *local-flow row. Thus every visible cell of the dynamical-system surface is read from* BHist *data already carried by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket flow route readback`

Proof. Read the phase row through the $\mathsf{Manifold}^{\uparrow}$ carrier. The source may be the singleton carrier of BEDC.Derived.ManifoldUp.ManifoldSingletonCarrier or any atlas row exposed by BEDC.Derived.ManifoldUp.ManifoldA in either case the displayed phase cell is a BHist cell with a certificate classifier.

Read the flow row through the $\mathsf{Ode}^{\uparrow}$ local-flow source of Definition 193.1. Its time, state, vector-field, Picard, limit, and endpoint entries are already arranged as BHist histories linked by Cont ledgers. Repacking these entries with the phase row gives the $\mathsf{DynSystem}^{\uparrow}$ carrier surface without introducing a host-level carrier of maps or trajectories. □

Theorem 195.3 (DynSystemUp classifier flow transport)**.** *Let two* $\mathsf{DynSystem}^{\uparrow}$ *carrier cells be displayed from two* $\mathsf{Ode}^{\uparrow}$ *local-flow rows over the same* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase carrier. If the displayed time cells, state cells, Picard histories, vector-field ledgers, and endpoint ledgers are related componentwise by the* hsame *transports supplied by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$*, then the two dynamical-system flow cells are classified as the same* $\mathsf{DynSystem}^{\uparrow}$ *flow.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket classifier flow transport`

Proof. Unpack each $\mathsf{DynSystem}^{\uparrow}$ cell into its phase row and its $\mathsf{Ode}^{\uparrow}$ local-flow row. The phase components are transported by the $\mathsf{Manifold}^{\uparrow}$ classifier, so the chart carrier visible to the flow row is unchanged up to hsame.

Transport the ODE row component by component: first the displayed time cells, then the state cells, then the vector-field ledger, the Picard history, the limit witness, and the endpoint ledger. The $\mathsf{Ode}^{\uparrow}$ classifier treats these as BHist rows and compares them by hsame rather than by equality of external functions.

After transport, both carrier cells expose the same ordered ledger surface. Repacking that transported surface is exactly the $\mathsf{DynSystem}^{\uparrow}$ classifier, so the two flow cells are classified together. □

Theorem 195.4 (DynSystemUp identity flow obligation)**.** *For every* $\mathsf{DynSystem}^{\uparrow}$ *phase-state cell* x *carried by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$*, the identity flow at empty time is represented by the* Cont *unit row*

$$\mathsf{Cont}(\mathsf{BHist.Empty}, x, x).$$

Its $\mathsf{DynSystem}^{\uparrow}$ *carrier ledger has empty displayed time, unchanged displayed state, the trivial Picard route, and endpoint row* x*; this row is classified by the same phase carrier and by the endpoint classifier inherited from* $\mathsf{Ode}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket identity flow obligation`

Proof. Start with the carried phase-state cell x. The empty time cell is the distinguished BHist unit, and the identity route is the empty-time continuation from x back to x.

The Picard part of the ledger is the unit continuation route. Since no nonempty vector-field evaluation is read, the vector-field component is only the displayed empty ledger accepted by the $\mathsf{Ode}^{\uparrow}$ local-flow source.

The endpoint row is again x. The $\mathsf{Manifold}^{\uparrow}$ classifier keeps the phase cell fixed, and the $\mathsf{Ode}^{\uparrow}$ endpoint classifier reads the unit continuation as an endpoint-preserving row. Repacking these rows gives the identity $\mathsf{DynSystem}^{\uparrow}$ flow obligation. □

Theorem 195.5 (DynSystemUp composition flow obligation)**.** *Let* R_{01} *and* R_{12} *be adjacent* $\mathsf{Ode}^{\uparrow}$ *local-flow source rows over the same* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase carrier, with the endpoint time-state*

cell of R_{01} classified with the initial time-state cell of R_{12}. Then $\mathsf{DynSystem}^{\uparrow}$ *represents the composite flow by concatenating the two source rows. The concatenated route is formed by* Cont *associativity on the adjacent Picard and endpoint ledgers, and its endpoint classifier is the* $\mathsf{Ode}^{\uparrow}$ *endpoint classifier transported through the shared phase carrier.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket composition flow obligation`

Proof. Read R_{01} as a local-flow row from (t_0, x_0) to (t_1, x_1), and read R_{12} as a local-flow row from (t_1, x_1) to (t_2, x_2). The adjacency hypothesis says that the middle time-state cell is shared in the $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ classifiers.

Use the Picard continuation scope of Definition 193.3 to concatenate the two Picard routes. The reassociation step is a Cont associativity move, with any syntactic mismatch at the middle cell handled by hsame transport.

The resulting source row has initial cell (t_0, x_0) and endpoint cell (t_2, x_2). Repacking it as a $\mathsf{DynSystem}^{\uparrow}$ flow row gives the composition obligation, and the endpoint determinacy is the classifier inherited from the concatenated $\mathsf{Ode}^{\uparrow}$ row. □

Theorem 195.6 (DynSystemUp Ode generator ledger)**.** *The* $\mathsf{DynSystem}^{\uparrow}$ *generator is hidden only through the* $\mathsf{Ode}^{\uparrow}$ *vector-field row and the* $\mathsf{Manifold}^{\uparrow}$ *chart carrier. Concretely, a* $\mathsf{DynSystem}^{\uparrow}$ *flow ledger records the displayed time cell, phase-state cell, Picard history, endpoint history, the* $\mathsf{Ode}^{\uparrow}$ *vector-field evaluation row, and the* $\mathsf{Manifold}^{\uparrow}$ *chart or singleton carrier row. It contains no additional generator oracle, no external trajectory equality, and no unclassified function-space comparison.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket ode generator ledger`

Proof. Inspect the carrier surface of Definition 195.2. Every time and state entry is a BHist row supplied by the $\mathsf{Ode}^{\uparrow}$ local-flow source, while the phase carrier is supplied by $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$.

The only place where the generator is read is the vector-field evaluation row of the $\mathsf{Ode}^{\uparrow}$ source. Its inputs are the displayed time cell and phase-state cell, and its output is transported by the endpoint classifier described in Definition 195.3.

The only place where coordinates or phase charts are read is the $\mathsf{Manifold}^{\uparrow}$ chart carrier. Therefore the $\mathsf{DynSystem}^{\uparrow}$ ledger hides the generator exactly behind those two certified rows and exposes only BHist, Cont, and hsame obligations to downstream flow-system consumers. □

Theorem 195.7 (DynSystemUp flow endpoint coverage)**.** *Every accepted flow endpoint row ϕ in a packet $\mathcal{D}$ of Definition 195.1 is reached by the displayed* Cont *ledger from the phase row p, elapsed-time row t, source point row x, and Ode local-flow witness ω. Conversely, whenever rows classified by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *supply a phase row, an elapsed-time row, a source point row, a local-flow witness, and a finite* Cont *route ending at ϕ over the same* Pkg *provenance, those rows repack as a* $\mathsf{DynSystem}^{\uparrow}$ *endpoint packet.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket endpoint coverage`

Proof. Project the tuple $\mathcal{D}$ to its fields p, t, x, ϕ, ω, ρ, and λ. By Definition 195.1, the field λ is exactly the finite continuation ledger that connects the source point, elapsed-time row, Ode source witness, and endpoint row inside the accepted packet.

The carrier theorem Definition 195.2 identifies the phase and Ode entries as displayed BHist rows rather than host objects. The generator theorem Definition 195.6 identifies ω as the only generator-facing row used by that endpoint route.

For the converse direction, start with the listed ManifoldUp and OdeUp classified rows together with the shared provenance and finite Cont route. Place them in the ten displayed fields of Definition 195.1; the same route supplies λ, and the classifier transport of Definition 195.3 supplies the inherited hsame comparisons. The resulting tuple is an accepted $\mathsf{DynSystem}^{\uparrow}$ endpoint packet whose public endpoint is ϕ. □

Theorem 195.8 (DynSystemUp phase-Ode classifier exactness)**.** *Two accepted* $\mathsf{DynSystem}^{\uparrow}$ *packets classify together exactly when their ManifoldUp phase-carrier and point rows, OdeUp local-flow rows, elapsed-time rows, endpoint rows, and carried* hsame *transports agree componentwise through the same* Pkg *provenance. No classifier clause may identify two packets by ignoring a displayed phase row, Ode row, elapsed-time row, endpoint row, transport witness, or provenance field.*

Proof. For the forward reading, unpack the public classifier through Definition 195.3. Its comparison first reads the ManifoldUp phase and point rows, then reads the OdeUp local-flow and elapsed-time rows, and finally transports the endpoint and ledger rows by the carried hsame witnesses.

The provenance field is not recomputed by the classifier. It is a projected field of Definition 195.1, so a classifier comparison that keeps the same public flow packet also keeps the same Pkg source for the ManifoldUp and OdeUp dependency packets.

Conversely, assume the listed componentwise comparisons and the shared provenance are already supplied. Apply the carrier surface theorem to read the two packets as the same ordered BHist flow surface, and apply the classifier transport theorem to repack those component comparisons as the public $\mathsf{DynSystem}^{\uparrow}$ classifier. The generator ledger theorem ensures that no hidden vector-field comparison is needed beyond the OdeUp local-flow row already in the list. □

Theorem 195.9 (DynSystemUp dependency provenance exhaustion)**.** *The only dependency rows exported by* $\mathsf{DynSystem}^{\uparrow}$ *are the ManifoldUp carrier and point rows, the OdeUp local-flow source row, the finite* Cont *endpoint ledger, and the shared* Pkg *provenance. The export surface contains no host trajectory space, no external vector-field equality row, and no global existence row.*

Proof. Inspect the carrier tuple in Definition 195.1. Its dependency-facing projections are the ManifoldUp packet $\mathcal{M}$, the OdeUp packet $\mathcal{O}$, the phase and point rows, the Ode witness ω, the continuation ledger λ, and the provenance row ρ.

The surface theorem Definition 195.2 shows that these projections are exactly the displayed BHist cells made available to a flow consumer. The endpoint coverage theorem above then accounts for the only endpoint-producing route, namely the finite Cont ledger from the certified phase, time, source, and Ode witness rows.

Finally apply Definition 195.6. It confines generator access to the OdeUp vector-field evaluation row and confines phase access to the ManifoldUp chart or singleton carrier. Therefore a downstream consumer receives no trajectory-space carrier, no vector-field equality outside the OdeUp row, and no global-existence dependency beyond the displayed provenance surface. □

195.2 The certificate

Theorem 195.10 (DynSystemUp zero-time flow ledger)**.** *For a packet of Definition 195.1, the zero-time row acts as the identity flow packet exactly when the displayed* Cont *ledger begins and ends at the same manifold point row after inherited* hsame *transport. In that case the source row* x*, target row* y*, and endpoint row* ϕ *are all classified with the same* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *point row, and the* Pkg *provenance remains the original dependency provenance.*

Proof. Unpack the flow packet and specialize the elapsed-time row to the zero-time cell carried by the ODE source surface. The local-flow source packet then exposes a trivial time-state route whose initial and terminal time rows are the same certified scalar row.

Project the point component through $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$. The source point is already a carried manifold row, and the zero-time route supplies no additional point movement beyond the displayed endpoint comparison.

Use the visible Cont ledger to identify the source, target, and endpoint rows by hsame. Since the tuple projection leaves ρ unchanged, the identity packet preserves the same Pkg provenance and is therefore the unit row of the dynamical-system certificate. □

Theorem 195.11 (DynSystemUp certified flow composition ledger)**.** *Let $\mathcal{D}_{01}$ and $\mathcal{D}_{12}$ be accepted $\mathsf{DynSystem}^\uparrow$ flow packets whose terminal point and time-state rows are classified with the initial point and time-state rows of the next packet. Certified time concatenation composes their accepted flow endpoint rows by associativity of Cont, and the resulting packet preserves the shared Pkg provenance over $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$.*

Proof. Read the first packet as a finite ledger from the source point x_0 through elapsed time t_{01} to endpoint x_1. Read the second packet as a finite ledger from the classified middle point x_1 through elapsed time t_{12} to endpoint x_2.

The compatibility hypothesis supplies the hsame transport needed to pass from the terminal row of the first packet to the initial row of the second. The ODE source classifier supplies the corresponding time concatenation row, so both pieces live over the same local-flow dependency surface.

Associate the two displayed Cont chains. The composite chain has source x_0, concatenated time row, endpoint x_2, and the same dependency witnesses $\mathcal{M}$ and $\mathcal{O}$.

Finally project the provenance row. Because neither component step changes the Pkg source certificates, the composite packet retains the shared provenance and is a valid $\mathsf{DynSystem}^\uparrow$ flow packet. □

Theorem 195.12 (DynSystemUp ODE source ledger)**.** *Every accepted $\mathsf{DynSystem}^\uparrow$ flow row projects to the $\mathsf{Ode}^\uparrow$ local-flow source row carried by ω. The dynamical-system packet contributes the phase carrier, manifold point classifier, elapsed-time ledger, endpoint row, and provenance binding, but it carries no additional host vector field beyond the $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ dependency row.*

Proof. Project the tuple of Definition 195.1 to its ODE component $\mathcal{O}$ and witness row ω. By acceptance of the packet, ω is already a row of the $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ local-flow source.

The phase component contributes only the carrier and point rows read through $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$. These rows select the manifold side on which the ODE source row is interpreted; they do not define a vector-field row.

The remaining fields t, x, y, ϕ, ρ, and λ are ledger and provenance data. Thus every vector-field evaluation, Picard route, and local endpoint determinacy claim is inherited from $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$, while $\mathsf{DynSystem}^\uparrow$ records how that source is packaged as a phase-space flow. □

Theorem 195.13 (DynSystemUp NameCert obligation surface)**.** *The public $\mathsf{NameCert}_{\mathsf{DynSystem}^\uparrow}$ obligation surface consists exactly of the flow carrier of Definition 195.1, the inherited classifier mediated by hsame over $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$, the zero-time unit ledger of Definition 195.10, the certified composition ledger of Definition 195.11, the ODE-source projection ledger of Definition 195.12, the stability rows obtained by transporting those fields through the same classifiers, and the ledger-exactness condition that every public flow endpoint is reached by the displayed Cont route and Pkg provenance.*

Proof. Collect the carrier fields from Definition 195.1. They provide the phase certificate, ODE source certificate, time row, source point, target point, endpoint, dependency witness, provenance, and finite ledger.

The classifier is componentwise. Manifold carrier and point rows are compared by $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, local-flow rows and elapsed-time rows by $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$, and endpoints by the transported hsame rows recorded in the same ledger.

The unit and composition rows are exactly the first two public laws needed by a flow certificate. The source theorem above supplies the dependency projection that prevents a downstream consumer from reading an uncarried vector field into the packet.

Conversely, every listed consumer reads one of these displayed rows: units read the zero-time ledger, composition reads two compatible ledgers, ODE projection reads ω, and stability reads classifier transport. Therefore the obligation surface is exact and contains no standard bridge, global-existence theorem, or host-level flow object beyond the carried BEDC rows. □

Definition 195.14 (DynSystemUp orbit-iterate packet)**.** *A* $\mathsf{DynSystem}^\uparrow$ *orbit-iterate packet is a finite* BHist *segment*

$$\mathcal{I} = (\mathcal{M}, \mathcal{O}, \nu, \rho, (R_i)_{0\le i<n}, (\alpha_i)_{0\le i<n}, (\beta_i)_{0\le i\le n}, (\lambda_i)_{0\le i<n})$$

displayed over one $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *phase carrier* $\mathcal{M}$ *and one* $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ *source packet* $\mathcal{O}$*. The row* $\nu : \mathsf{BHist}$ *is the unary index history for the finite segment, each* R_i *is an accepted* $\mathsf{DynSystem}^\uparrow$ *flow row of Definition 195.1 over the same phase carrier and ODE source, each* α_i *is the elapsed-time row of* R_i*, each* β_i *is the endpoint row at the* i*th public phase state, and each* λ_i *is the displayed* Cont *ledger from* β_i *through* α_i *to* β_{i+1}*.*

The packet is accepted when the segment provenance ρ *is the shared* Pkg *provenance of the manifold and ODE source packets, every adjacent endpoint pair is related by the inherited* hsame *witnesses needed to read* R_i *and* R_{i+1} *over the same public state, and the unary row* ν *indexes exactly the listed flow rows. Its public data are therefore a finite BHist segment, the two source certificates, the endpoint rows, the segment provenance, and the ordered* Cont *ledger list; it carries no external sequence object beyond these displayed histories.*

Theorem 195.15 (DynSystemUp orbit-iterate carrier closure)**.** *Let* $\mathcal{I}$ *be an accepted* $\mathsf{DynSystem}^\uparrow$ *orbit-iterate packet with terminal endpoint row* β_n*. Let* R_n *be an accepted* $\mathsf{DynSystem}^\uparrow$ *flow row over the same* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *phase carrier and* $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ *source packet, whose initial point and time-state rows are classified with* β_n *by the inherited* hsame *witnesses and whose provenance is the same* ρ*. Then appending* R_n *and its displayed* Cont *ledger to* $\mathcal{I}$ *produces an accepted* $\mathsf{DynSystem}^\uparrow$ *orbit-iterate packet with endpoint row* β_{n+1} *and the same carried phase, source, and segment provenance.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemOrbitIteratePacket carrier closure`

Proof. Unpack the accepted packet $\mathcal{I}$ into its unary index row, flow rows, endpoint rows, provenance row, and ordered Cont ledger list. The terminal endpoint row β_n is already a public BHist row of the packet.

Read R_n through Definition 195.1. The compatibility hypothesis supplies the hsame transport that identifies the initial point and time-state rows of R_n with the terminal public state of $\mathcal{I}$, and the provenance hypothesis keeps the same Pkg row.

Extend the unary index row by one successor cell and append the new flow row, elapsed-time row, endpoint row, and ledger. The only composition work is the reassociation of the existing

terminal Cont chain with the ledger of R_n, which is the same associativity move used in Definition 195.11.

Repack the displayed fields in the form of Definition 195.14. All acceptance clauses are inherited from the old packet or from R_n, so the carried packet remains over the same phase carrier, ODE source, and segment provenance. □

Theorem 195.16 (DynSystemUp invariant consumer projection)**.** *Let C be a downstream consumer predicate on accepted* $\mathsf{DynSystem}^{\uparrow}$ *orbit-iterate packets whose value is read only from the projected* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase row, the projected* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *source row, the endpoint rows β_i, the displayed* Cont *ledgers λ_i, the inherited* hsame *witnesses at adjacent endpoints, and the shared* Pkg *provenance ρ. If two accepted orbit-iterate packets are related by the* $\mathsf{DynSystem}^{\uparrow}$ *classifier transport on those displayed rows, then C has the same truth value on both packets.*

Proof. Project both orbit packets to the rows named in the statement. The consumer ignores every component outside this projection, so it cannot observe the syntactic presentation of the surrounding tuple.

Use the $\mathsf{DynSystem}^{\uparrow}$ classifier transport componentwise. The phase row is transported by $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, the ODE source row by $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$, and the endpoint and ledger rows by the hsame witnesses carried in the packets.

The provenance projection is unchanged because both packets expose the same Pkg source row after classifier transport. Therefore every argument supplied to C is transported to the corresponding displayed argument of the other packet.

Since C is defined only from those projected arguments, the two evaluations are the same. This proves that consumer-visible invariants are invariants of the DynSystem classifier rather than of a particular tuple presentation. □

Theorem 195.17 (DynSystemUp flow segment exhaustion)**.** *Every public finite segment of an accepted* $\mathsf{DynSystem}^{\uparrow}$ *orbit-iterate packet is obtained from its first endpoint row by repeated use of the zero-time ledger of Definition 195.10 and the certified composition ledger of Definition 195.11. Equivalently, no additional public segment operation is needed beyond selecting a finite consecutive sublist of displayed flow rows and composing their* Cont *ledgers in order.*

Proof. Proceed by induction on the length of the selected consecutive segment. For the empty segment, use the zero-time ledger of Definition 195.10; the source and endpoint rows are the same after the packet's inherited hsame transport.

For a one-row segment, read the selected row directly as a packet of Definition 195.1. Its public endpoint is already reached by the displayed Cont ledger in the orbit-iterate packet.

For the step from a length k segment to a length $k+1$ segment, apply the induction hypothesis to the first k rows. The next row is adjacent by the acceptance condition in Definition 195.14, so the terminal endpoint of the prefix is classified with the initial endpoint of the next accepted flow row.

Compose the prefix ledger with the next row using Definition 195.11. Reassociation of the finite Cont chain gives exactly the displayed ledger of the longer segment, and the shared Pkg provenance remains unchanged.

Thus every public finite segment is exhausted by repeated zero-time and composition moves already present in the DynSystem certificate surface. □

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket segment route exhaustion`

Theorem 195.18 (DynSystemUp time reindex stability)**.** *Let F_0 and F_1 be two* $\mathsf{DynSystem}^{\uparrow}$ *flow packets over the same* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase row, source state row, endpoint row, displayed* Cont *ledger, and* Pkg *provenance row. If their elapsed-time rows are related by the* $\mathsf{Ode}^{\uparrow}$ *time classifier and every fixed field is compared by the inherited* hsame *rows of the same packet surface, then F_0 and F_1 are classified by the public* $\mathsf{DynSystem}^{\uparrow}$ *classifier.*

Proof. Project both packets to their $\mathsf{Ode}^{\uparrow}$ local-flow rows by Definition 195.12. The hypotheses keep the phase carrier, source state, endpoint row, continuation ledger, and provenance fixed, so the only moving component is the elapsed-time row read by the $\mathsf{Ode}^{\uparrow}$ source classifier.

Apply the $\mathsf{Ode}^{\uparrow}$ time transport to that projected local-flow row. The transport produces the same source ledger with the reindexed elapsed-time cell and leaves the displayed endpoint comparison on the inherited ledger.

Repack the transported row through Definition 195.3. Since all non-time fields are already compared by the fixed hsame witnesses, the repacked packets are classified by $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$. □

Theorem 195.19 (DynSystemUp endpoint determinacy surface)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase carrier, an* $\mathsf{Ode}^{\uparrow}$ *local-flow source row, an initial time-state row, a displayed* Cont *ledger, and a* Pkg *provenance row. Any two accepted* $\mathsf{DynSystem}^{\uparrow}$ *endpoint rows over that surface are compared exactly by the endpoint classifier inherited from* $\mathsf{Ode}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket endpoint determinacy surface`

Proof. Use Definition 195.12 to project both accepted endpoint rows to the same $\mathsf{Ode}^{\uparrow}$ local-flow source. The fixed phase carrier, initial time-state row, continuation ledger, and provenance ensure that the two projections have the same source ledger.

Apply the $\mathsf{Ode}^{\uparrow}$ endpoint determinacy classifier on that shared source ledger. It compares the two projected endpoint rows without adding a host-level trajectory equality or any generator field outside the carried local-flow source.

Repack the comparison as a $\mathsf{DynSystem}^{\uparrow}$ endpoint comparison. The repacking uses only the fixed $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ phase carrier and the same Pkg provenance, so the public endpoint surface is precisely the inherited $\mathsf{Ode}^{\uparrow}$ endpoint surface. □

Theorem 195.20 (DynSystemUp flow provenance exactness)**.** *Every public* $\mathsf{DynSystem}^{\uparrow}$ *flow consumer row projects to one of the* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *carrier row,* $\mathsf{Ode}^{\uparrow}$ *local-flow row, elapsed-time row, endpoint row, finite* Cont *ledger, or* Pkg *provenance fields of Definition 195.1. Conversely, those displayed rows repack as the public* $\mathsf{DynSystem}^{\uparrow}$ *flow surface.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemFlowPacket flow provenance exactness`

Proof. Inspect the carrier tuple of Definition 195.1. Its projections are exactly the manifold carrier, ODE source row, elapsed-time row, endpoint row, dependency provenance, and finite continuation ledger made available to consumers.

The zero-time and composition consumers read the same tuple through Definition 195.10 and Definition 195.11. The ODE-source consumer reads the projected local-flow row through Definition 195.12, and endpoint consumers read the inherited classifier surface from the same source ledger.

Conversely, a public surface built from those rows is repacked componentwise by the classifier transport and obligation surface of Definition 195.13. No other row is exposed as public flow provenance, so the exported provenance surface is exact. □

Theorem 195.21 (DynSystemUp public flow endpoint exactness)**.** *For every accepted packet $\mathcal{D}$ of Definition 195.1, the public flow endpoint exposed by* $\mathsf{NameCert}_{\mathsf{DynSystem}^\uparrow}$ *is exactly the row ϕ reached by the displayed route*

$$\mathsf{Cont}(t, x, \phi).$$

The route is the λ component of $\mathcal{D}$, its source and endpoint comparisons are transported only by inherited hsame *rows from* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$*, and its dependency origin is the* Pkg *provenance recorded by ρ over the projected packets $\mathcal{M}$ and $\mathcal{O}$.*

Proof. Project the accepted tuple to the fields $\mathcal{M}$, $\mathcal{O}$, ρ, and λ. The first two fields name the manifold and ODE certificates through which the source point, time row, local-flow row, and endpoint row are read.

The ledger field λ is the finite Cont route connecting the displayed source row x, elapsed-time row t, and endpoint row ϕ. Any comparison needed to align the source or terminal rows is already supplied by the inherited hsame classifiers of $\mathcal{M}$ and $\mathcal{O}$.

Finally read ρ. Since the provenance field is projected from the same packet and is not recomputed by a downstream consumer, the exposed endpoint is the carried row ϕ with its original Pkg dependency binding. □

Theorem 195.22 (DynSystemUp flow adjacency classifier)**.** *Let $\mathcal{D}_{01}$ and $\mathcal{D}_{12}$ be accepted* $\mathsf{DynSystem}^\uparrow$ *packets over the same projected* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ *dependency surface. They compose as adjacent flow packets only when the terminal time-state rows of $\mathcal{D}_{01}$ and the initial time-state rows of $\mathcal{D}_{12}$ are classified together by the ManifoldUp point classifier and the OdeUp local-flow classifier. Under that classifier hypothesis, the composite packet is the certified composition row of Definition 195.11.*

Proof. Read the two packets through Definition 195.11. Its compatibility premise is not an external equality of trajectories; it is the displayed classification of the middle point and time-state rows.

Use Definition 195.3 to identify the classifiers that supply this middle comparison. The phase component is read by $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, while the elapsed-time and local-flow rows are read by $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$.

With those two classifier readings in place, the hypotheses of the composition ledger are exactly the adjacency hypotheses. The resulting concatenated Cont route is therefore the only certified composite exposed by $\mathsf{NameCert}_{\mathsf{DynSystem}^\uparrow}$. □

Theorem 195.23 (DynSystemUp ODE phase-scope separation)**.** *In every accepted* $\mathsf{DynSystem}^\uparrow$ *flow packet, phase rows are read only through* $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$ *and generator rows are read only through* $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$*. The fields p, x, y, and the phase part of ϕ belong to the ManifoldUp reading, while ω and every vector-field or local-flow generator row belong to the OdeUp reading.*

Proof. Project the packet as in Definition 195.12. The ODE source is carried by $\mathcal{O}$ and witnessed by ω, so generator information enters the dynamical-system packet only through the $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$ local-flow source row.

Next apply Definition 195.6. It identifies the chart or singleton carrier as the ManifoldUp side and the vector-field evaluation row as the OdeUp side.

Thus the two readings have disjoint scopes inside the packet: phase classification is routed through $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, and generator classification is routed through $\mathsf{NameCert}_{\mathsf{Ode}^\uparrow}$. The packet itself only records their BHist ledger and provenance binding. □

Theorem 195.24 (DynSystemUp flow consumer projection coverage)**.** *Every downstream consumer whose output is invariant under the exposed phase rows, elapsed-time rows, source and endpoint rows, ODE-source row, displayed* Cont *ledger, inherited* hsame *transports, and* Pkg *provenance of Definition 195.13 is invariant under the whole accepted* $\mathsf{DynSystem}^{\uparrow}$ *flow packet. No additional packet field can change such a consumer.*

Proof. Project an accepted packet to the obligation surface of Definition 195.13. This yields the phase certificate, ODE source certificate, time row, source point, target point, endpoint row, dependency witness, provenance, and finite continuation ledger.

Now compare two accepted packets whose listed rows are related by the exposed classifiers. The phase, source, target, and endpoint readings are transported by $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, while the elapsed-time, local-flow, and ODE-source readings are transported by $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$.

The consumer is invariant under exactly these projections, including the Cont route, hsame transports, and Pkg provenance. Since the displayed tuple of Definition 195.1 has no further public row beyond those projections, the consumer has the same value on the whole packet. □

Theorem 195.25 (DynSystemUp public NameCert export)**.** *The public* $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ *export surface consists of the carrier of Definition 195.1, the inherited classifier of Definition 195.3, the zero-time row of Definition 195.10, the composition row of Definition 195.11, the ODE-source row of Definition 195.12, the endpoint exactness row of Definition 195.21, the adjacency classifier of Definition 195.22, the phase-scope separation row of Definition 195.23, and the consumer projection coverage row of Definition 195.24.*

Lean4 ✓ `BEDC.Derived.DynSystemUp.DynSystemUp StdBridge`

Proof. Begin with the carrier and classifier. The carrier supplies the displayed BHist tuple, and the classifier transports each visible flow field through the ManifoldUp and OdeUp certificate surfaces.

Add the algebraic flow rows. The zero-time theorem supplies the unit packet, the composition theorem supplies certified concatenation, and the adjacency theorem identifies the classifier premise under which that concatenation is public.

Add the source and projection rows. The ODE-source theorem fixes the generator reading, the phase-scope theorem separates phase and generator access, and the endpoint exactness theorem identifies the exposed endpoint with the displayed Cont route and provenance.

The consumer coverage theorem then states that these rows exhaust the public projections of an accepted packet. Together they form the advertised $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ export surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$**.**

Constructive story: A dynamical-system row is built by taking an already certified $\mathsf{Manifold}^{\uparrow}$ phase packet and an already certified $\mathsf{Ode}^{\uparrow}$ local-flow packet, then displaying the phase row, elapsed-time row, source and target state rows, endpoint row, ODE dependency witness, provenance, and finite Cont route in one BHist tuple. The carrier accepts exactly those tuples whose phase and point rows are admitted by the manifold certificate, whose local-flow source is admitted by the ODE certificate, and whose endpoint is reached by the displayed route; a tuple missing either dependency surface, the endpoint route, or the shared provenance stays outside the carrier. Two accepted flow packets name the same dynamical-system step when the phase, time, source, target, ODE witness, endpoint, and provenance rows are transported componentwise by the inherited hsame comparisons and by the same visible continuation ledger, rather than by equality of external maps or trajectories. The source specification is the displayed manifold-plus-ODE flow surface, the pattern specification records zero-time identity, segment composition, adjacency, endpoint exactness, and phase-generator separation, and the ledger keeps every public coordinate aligned with the raw route that produced it. The five name-certificate obligations are inherited through that finite packet: an admitted flow supplies inhabitation, componentwise hsame gives reflexivity, symmetry, and transitivity, and classifier transport repacks the transported phase and ODE rows as another admitted flow packet. A packet h lies in the signature gap of a $\mathsf{DynSystem}^{\uparrow}$ token p when the source-domain accepts the displayed phase/ODE tuple, the signature relation pairs it with an answer flow packet and evidence ledger, and

the token-introduction relation names that answer as p; unfolded, the phase carrier, elapsed time, endpoint, ODE source, and continuation route are exactly the rows encoded by p. The chapter registers this routed-flow construction under the name $\mathsf{DynSystem}^{\uparrow}$; flow laws are certified concatenations and unit rows over existing manifold and ODE certificates, with no global existence theorem, invariant-measure object, ergodic package, or independent trajectory carrier introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public export surface of Definition 195.25 covers the displayed BHist flow packet, inherited Cont routes, hsame transports, Pkg provenance, the dependency surfaces of $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$, and the finite hsame-fixed orbit-iterate row of Definition 195.27.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.DynSystemUp.DynSystemFlowPacket\_endpoint\_coverage`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim global existence, uniqueness beyond the displayed OdeUp row, invariant-measure structure, or ergodic consequences.

Upgrade path: No further upgrade – chapter at top of formal axis.

Definition 195.26 (DynSystemUp hsame-fixed one-step row)**.** *Let $\mathcal{D}$ be an accepted* $\mathsf{DynSystem}^{\uparrow}$ *flow packet of Definition 195.1 over phase carrier $\mathcal{M}$, ODE source packet $\mathcal{O}$, elapsed-time row t, displayed source point row p, endpoint row ϕ, provenance row ρ, and ledger λ. The packet is* hsame*-fixed at p when its public endpoint is classified with the displayed source point,*

$$\mathsf{hsame}(\phi, p),$$

and the comparison is carried on the same $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *dependency surface as $\mathcal{D}$.*

Theorem 195.27 (DynSystemUp hsame-fixed orbit-iterate stability)**.** *Let R be an accepted* $\mathsf{DynSystem}^{\uparrow}$ *flow packet of Definition 195.1 over a fixed* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *phase carrier, a fixed* $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ *source packet, elapsed-time row t, displayed source point row p, endpoint row ϕ, provenance row ρ, and continuation ledger λ. Assume that R is* hsame*-fixed at p in the sense of Definition 195.26. Then, for every finite unary horizon ν admitted by Definition 195.14, the orbit-iterate packet obtained by reading R at each successor cell of ν is accepted, has every public endpoint row classified with p, and has an ordered* Cont *ledger list whose entries are copies of λ transported only by the supplied fixed-row comparison* $\mathsf{hsame}(\phi, p)$ *and by* Cont *associativity.*

Proof. Proceed by induction on the displayed unary horizon ν. For the empty horizon, the orbit segment has no successor flow row. Its public endpoint is the initial row p, and the required comparison with p is the reflexive inherited hsame row on the carried ManifoldUp point. The ledger list is empty, so there is no endpoint motion beyond the zero-time row of Definition 195.10.

For the successor step, assume that the prefix segment indexed by ν is an accepted orbit-iterate packet whose terminal public endpoint β_n is classified with p, and whose displayed ledger list consists of transported copies of λ. The fixed-row hypothesis supplies $\mathsf{hsame}(\phi, p)$. By symmetry and transitivity for the inherited endpoint classifier, the terminal comparison $\mathsf{hsame}(\beta_n, p)$ and the fixed comparison $\mathsf{hsame}(\phi, p)$ supply the transport needed to read another copy of R with initial point classified with β_n and terminal endpoint classified again with p.

Apply the carrier-closure theorem Definition 195.15 to append that transported copy of R to the prefix packet. The appended row has the same phase carrier, the same ODE source packet, the same elapsed-time row t, and the same provenance row ρ. Its continuation entry is the same ledger λ, with only the endpoint transports just described inserted at the boundary.

The only ledger operation introduced by the append is reassociation of the finite continuation chain, exactly as in Definition 195.11. Hence the successor packet is accepted by Definition 195.14, its terminal endpoint is classified with p, and its ledger list is the prefix list followed by one further transported copy of λ. Induction over the finite unary horizon gives the stated stability for every displayed iterate. □

Chapter 196

A Concrete Naming Certificate for $\mathsf{Ergodic}^{\uparrow}$

The ergodic-theory interface $\mathsf{Ergodic}^{\uparrow}$ packages a measure-preserving transformation together with the ergodic decomposition of its invariant subspaces. It depends on $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ (chapter 195) and $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92).

196.1 Carrier definition

Definition 196.1 (ErgodicUp BHist source surface)**.** *An* $\mathsf{Ergodic}^{\uparrow}$ *BHist source surface is a finite package consisting of* $\mathsf{DynSystem}^{\uparrow}$ *transformation rows,* $\mathsf{Measure}^{\uparrow}$ *measure rows, invariant-subspace classifier rows,* hsame *transport rows for the displayed dynamic and measure endpoints, finite* Cont *ledger routes, and* Pkg *provenance for the whole package. A consumer reads the ergodic interface only through these displayed rows.*

Definition 196.2 (ErgodicUp measure-preserving carrier)**.** *An* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier is an accepted Definition 196.1 whose displayed* $\mathsf{DynSystem}^{\uparrow}$ *transformation preserves the displayed* $\mathsf{Measure}^{\uparrow}$ *measure row through the listed finite* hsame, Cont, *and* Pkg *ledger. The carrier records the preservation claim as a finite BEDC row, not as an ambient sigma-algebra identity.*

196.2 The certificate

Theorem 196.3 (ErgodicUp invariant-subspace classifier)**.** *For an accepted* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier, invariant-subspace comparison is read only through the displayed* $\mathsf{DynSystem}^{\uparrow}$ *transformation rows,* $\mathsf{Measure}^{\uparrow}$ *measure rows,* BHist *endpoint rows, and* hsame *transport rows of Definition 196.1. No host sigma-algebra quotient, completed invariant subspace, or external equivalence relation contributes an additional classifier row.*

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingSurface invariant subspace classifier`
Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicBHistSourceSurface invariant subspace classifier`

Proof. Project the accepted carrier to its $\mathsf{DynSystem}^{\uparrow}$ transformation and $\mathsf{Measure}^{\uparrow}$ measure coordinates. These rows supply the only source endpoints for invariant-subspace comparison.

Next project the BHist endpoint and the displayed hsame transports. Repacking these finite rows gives the invariant-subspace classifier seen by a consumer.

The source package has no field for a host sigma-algebra quotient, completed invariant subspace, or external equivalence relation. Those data therefore cannot refine the displayed classifier. □

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingCarrier invariant subspace classifier`

Theorem 196.4 (ErgodicUp decomposition ledger)**.** *Every decomposition witness exposed by an accepted* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier is exhausted by the finite invariant-row ledger of Definition 196.1: the displayed invariant-subspace classifier rows, the* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance. The witness imports no host sigma-algebra quotient.*

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingSurface decomposition ledger`

Proof. First read the carrier through Definition 196.3. This reduces every decomposition comparison to the displayed transformation, measure, BHist, and transport rows.

Then project the finite Cont routes and Pkg provenance. These rows record how each invariant-row witness is carried through the package and give the full decomposition ledger available to downstream analysis consumers.

Since the carrier boundary contains no host sigma-algebra quotient, projecting and repacking the ledger cannot introduce one. The displayed finite invariant ledger therefore exhausts the decomposition witness. □

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingCarrier decomposition ledger`

Theorem 196.5 (ErgodicUp NameCert obligation surface)**.** *The* $\mathsf{Ergodic}^{\uparrow}$ *NameCert obligation surface packages the Definition 196.2, the invariant-subspace classifier of Definition 196.3, the transport stability rows, and the decomposition ledger of Definition 196.4. It exposes exactly the displayed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* Pkg *rows needed by downstream analysis consumers.*

Proof. Start with the accepted measure-preserving carrier. Its displayed preservation row supplies the carrier endpoint over the dynamic and measure source rows.

Use the invariant-subspace classifier theorem to identify the classifier rows available to a consumer, then use the decomposition ledger theorem to identify the finite witness ledger and provenance rows. These two projections supply the stability and ledger components of the NameCert obligation surface.

Finally repack the carrier, classifier, stability, and ledger rows into one finite BHist package. The package has no additional field beyond the displayed $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows, so it is the concrete $\mathsf{Ergodic}^{\uparrow}$ obligation surface. □

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingCarrier semantic name certificate`

Definition 196.6 (ErgodicUp invariant-row subledger restriction)**.** *Let* $\mathcal{E}$ *be an accepted* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier of Definition 196.2. An invariant-row subledger restriction of* $\mathcal{E}$ *is a finite packet* $\mathcal{E}|_I$ *formed as follows. It keeps the displayed* $\mathsf{DynSystem}^{\uparrow}$ *transformation rows, the displayed* $\mathsf{Measure}^{\uparrow}$ *measure rows, the* BHist *endpoint rows, the measure-preservation row, all* hsame *transports for the dynamic and measure endpoints, all finite* Cont *routes used by the preservation row, and the same* Pkg *provenance. Its invariant-subspace classifier ledger is a displayed finite subledger* I *of the invariant-subspace classifier rows of* $\mathcal{E}$*, with each retained row carrying its original endpoint witnesses and its original transport route. No row outside* I *is readable as an invariant-subspace row of* $\mathcal{E}|_I$*.*

Theorem 196.7 (ErgodicUp invariant subledger restriction stays carried)**.** *Let $\mathcal{E}$ be an accepted* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier, and let $\mathcal{E}|_I$ be an invariant-row subledger restriction in the sense of Definition 196.6. Then $\mathcal{E}|_I$ is an accepted* $\mathsf{Ergodic}^{\uparrow}$ *measure-preserving carrier. Its public decomposition ledger is exactly the retained finite invariant-row subledger I together with the unchanged* hsame, Cont, *and* Pkg *rows; in particular, the restriction adds no host sigma-algebra quotient and no completed invariant-subspace row.*

Proof. Project $\mathcal{E}$ to the source surface of Definition 196.1. The restriction keeps the dynamic rows, measure rows, endpoint rows, transport rows, continuation routes, and package provenance unchanged, and replaces the invariant-subspace classifier ledger by the displayed finite subledger I. A finite subledger of a finite ledger is again a finite ledger, so these retained rows form an $\mathsf{Ergodic}^{\uparrow}$ BHist source surface.

The measure-preserving carrier condition of Definition 196.2 is read from the displayed $\mathsf{DynSystem}^{\uparrow}$ transformation, the displayed $\mathsf{Measure}^{\uparrow}$ measure row, and the listed finite hsame, Cont, and Pkg ledger. None of those rows is removed or altered by Definition 196.6. The same preservation row therefore witnesses the carrier condition for $\mathcal{E}|_I$.

It remains to identify the public invariant data exposed by the restricted packet. By Definition 196.3, invariant-subspace comparison for $\mathcal{E}$ is read only through the displayed invariant rows and their hsame transports. The restricted packet displays exactly the rows in I, with the same endpoint witnesses and transport routes, so its classifier has no additional invariant row to read. By Definition 196.4, every decomposition witness is exhausted by the displayed finite invariant ledger, the hsame transports, the finite Cont routes, and Pkg provenance. After restriction this exhaustion ranges precisely over I and the unchanged route and provenance rows. Since neither the original surface nor the restriction has a coordinate for a host sigma-algebra quotient or completed invariant subspace, acceptance is preserved and no outside decomposition row is introduced. □

Lean4 ✓ `BEDC.Derived.ErgodicUp.ErgodicMeasurePreservingCarrier invariant subledger restriction stays carried`

Closure status for $\mathsf{NameCert}_{\mathsf{Ergodic}^{\uparrow}}$.

Constructive story: An $\mathsf{Ergodic}^{\uparrow}$ packet is generated from DynSystemUp transformation rows, MeasureUp measure rows, invariant-subspace classifier rows, measure-preservation evidence, finite BHist endpoint rows, componentwise hsame transport, Cont ledger routes, and Pkg provenance. The classifier reads ergodic decomposition only through displayed invariant rows and their transport routes; a subledger restriction is generated by retaining a finite invariant-row subledger and the unchanged preservation, transport, route, and provenance rows. The ledger story is finite invariant accounting: decomposition data is carried as displayed rows over the dynamic and measure sources, not as a host sigma-algebra quotient or completed invariant subspace.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Ergodic}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter binds the measure-preserving carrier, invariant-subspace classifier, transformation and invariant-measure rows, invariant subledger restriction, and the concrete BHist, hsame, Cont, Pkg, and NameCert rows consumed by the displayed proof routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Ergodic}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim host sigma-algebra quotients, completed invariant subspaces, off-ledger invariant rows, or ambient ergodic decomposition beyond the displayed finite ledger.

Upgrade path: Further closure requires a public bridge from the displayed DynSystemUp, MeasureUp, BHist, hsame, Cont, Pkg, and NameCert rows into a reusable ergodic decomposition interface.

Chapter 197

A Concrete Naming Certificate for $\mathsf{FirstOrder}^{\uparrow}$

The first-order-logic interface $\mathsf{FirstOrder}^{\uparrow}$ packages a signature of relation and function symbols, a set of formulas, and a deduction calculus together with the soundness and completeness data. It depends on $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ (chapter 111) and $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$ (chapter 119).

197.1 Carrier definition

Definition 197.1 (FirstOrderUp BHist syntax carrier)**.** *A carried* $\mathsf{FirstOrder}^{\uparrow}$ *syntax packet is a displayed* BHist *tuple* $(S, T, V, R, F, \tau, \varphi, \rho)$*. The row* S *is the* $\mathsf{Set}^{\uparrow}$ *symbol-source row,* T *is the* $\mathsf{Tree}^{\uparrow}$ *parse-tree row,* V *is the variable ledger,* R *and* F *are the relation-symbol and function-symbol rows,* τ *is the finite term/formula tree endpoint,* φ *is the displayed formula endpoint, and* ρ *records the* Pkg *provenance that ties the syntax packet to the visible naming certificate.*

Acceptance of the packet is read entirely through BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*, and the* $\mathsf{Set}^{\uparrow}$ *and* $\mathsf{Tree}^{\uparrow}$ NameCert *rows. A host grammar, a host set of strings, or an external syntax equality is not part of the carrier.*

Lean4 (definition) `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier`

Definition 197.2 (FirstOrderUp deduction ledger classifier)**.** *Two* $\mathsf{FirstOrder}^{\uparrow}$ *syntax/deduction packets agree when their symbol rows, parse-tree rows, formula endpoint rows, and deduction-step ledgers are transported by* hsame *through the same* $\mathsf{Set}^{\uparrow}$ *and* $\mathsf{Tree}^{\uparrow}$ *certificates. The comparison is rowwise: relation symbols and function symbols move through the* $\mathsf{Set}^{\uparrow}$ *source, parse subtrees and formula endpoints move through the* $\mathsf{Tree}^{\uparrow}$ *source, and every deduction step carries its premise and conclusion endpoints by the displayed* Cont *and* hsame *rows recorded in the ledger.*

197.2 The certificate

Theorem 197.3 (FirstOrderUp formula carrier obligation)**.** *Every accepted* $\mathsf{FirstOrder}^{\uparrow}$ *formula row projects to a* $\mathsf{Set}^{\uparrow}$ *symbol row, a* $\mathsf{Tree}^{\uparrow}$ *parse row, a finite* ProbeBundle *arity ledger, and a* SigRel *endpoint relation. These projections are exactly the visible carrier data of Definition 197.1; no formula is accepted by a host syntax predicate outside the displayed* BHist *packet.*

Proof. Invert the carried syntax packet of Definition 197.1. The first projections expose the $\mathsf{Set}^{\uparrow}$ symbol-source row, the $\mathsf{Tree}^{\uparrow}$ parse-tree row, the relation and function symbol rows, and the formula endpoint φ.

The arity information is read from the finite ProbeBundle ledger attached to the displayed relation and function rows. Each arity check is therefore a finite probe over the named symbol endpoints, not an inspection of an ambient syntax table.

Finally apply induction on the $\mathsf{Tree}^{\uparrow}$ syntax row. Leaf rows read variables and constants from the symbol and variable ledgers; internal rows read their subformula endpoints from the same finite tree ledger. The endpoint relation is the displayed SigRel row tying the resulting formula endpoint to the accepted packet. □

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier formula carrier obligation`

Theorem 197.4 (FirstOrderUp classifier transport obligation)**.** *Every classifier-related* $\mathsf{FirstOrder}^{\uparrow}$ *syntax and deduction packet transports its relation-symbol rows, function-symbol rows, formula endpoints, and deduction-step ledgers through the displayed* $\mathsf{Set}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, hsame, Cont, ProbeBundle, *and* SigRel *rows of* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$. *In particular, a transported classifier packet is obtained by projecting those visible rows from the source packet and repacking them as the corresponding target packet; no host syntax equality or external classifier relation is used.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier classifier transport obligation`

Proof. Start with an accepted classifier-related packet and invert the carried BHist tuple of Definition 197.1. The projections expose the $\mathsf{Set}^{\uparrow}$ symbol-source row, the $\mathsf{Tree}^{\uparrow}$ parse-tree row, the relation-symbol and function-symbol rows, the formula endpoint, and the deduction-step ledger.

Transport the symbol rows through the displayed $\mathsf{Set}^{\uparrow}$ certificate and the parse and formula rows through the displayed $\mathsf{Tree}^{\uparrow}$ certificate. The ProbeBundle rows carry the arity checks, while the SigRel rows carry the endpoint relation that identifies the source and target formula endpoints.

It remains to repack the deduction data. Use the Cont and hsame rows attached to each ledger step to move premises and conclusions through the same provenance. The repacked rows form another displayed $\mathsf{FirstOrder}^{\uparrow}$ classifier packet, and every field of that packet has been obtained from the public certificate surface. □

Theorem 197.5 (FirstOrderUp deduction endpoint exactness)**.** *The visible* $\mathsf{FirstOrder}^{\uparrow}$ *deduction endpoints are exact. Every accepted conclusion endpoint is produced by a finite displayed deduction ledger whose premises are already accepted formula endpoints, and every such finite ledger repacks as a* $\mathsf{FirstOrder}^{\uparrow}$ *deduction endpoint with the same conclusion row.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier deduction endpoint exactness`

Proof. For coverage, invert an accepted deduction endpoint. By Definition 197.8, the endpoint is visible only through the syntax carrier, the deduction classifier, and the displayed deduction-soundness ledger. Thus its conclusion row is the final conclusion of a finite ledger over formula endpoints already carried by Definition 197.3.

For soundness, induct over the finite ledger. The empty ledger exposes only initial accepted formula endpoints. In the step case, the induction hypothesis supplies accepted endpoints for the earlier premises, and Definition 197.7 supplies the visible Cont and hsame rows for the next rule step and its conclusion.

For completeness, start with a finite displayed ledger whose premises are accepted formula endpoints and whose steps carry the required transport rows. Repack the formula endpoints by the carrier obligation, repack the rule rows by the deduction-soundness obligation, and read the last conclusion row as the displayed $\mathsf{FirstOrder}^{\uparrow}$ deduction endpoint. □

Theorem 197.6 (FirstOrderUp ModelTheory consumer boundary)**.** *A downstream* $\mathsf{ModelTheory}^{\uparrow}$ *packet may consume from* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *only the symbol source, parse-tree source, formula endpoint rows, deduction endpoint rows, classifier transport rows, and provenance rows exposed by the* $\mathsf{FirstOrder}^{\uparrow}$ *certificate. The* $\mathsf{FirstOrder}^{\uparrow}$ *certificate alone supplies no host syntax equality, external proof-system validity predicate, semantic satisfaction relation, or model carrier.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier modeltheory consumer boundary`

Proof. Project the public surface described in Definition 197.8. The available rows are the syntax carrier, the deduction classifier, the formula and deduction obligation rows, and the $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ provenance rows.

Next apply Definition 197.4. It shows that classifier transport for relation symbols, function symbols, formula endpoints, and deduction ledgers is already accounted for by the displayed hsame, Cont, ProbeBundle, and SigRel rows.

Finally apply Definition 197.5. It makes the deduction endpoint rows exact, so a consumer can read precisely the finite ledgers and their accepted conclusion endpoints. None of the projected or repacked rows contains a semantic satisfaction relation or model carrier, and none asserts an ambient proof-system validity predicate outside the displayed BEDC ledger. □

Theorem 197.7 (FirstOrderUp deduction soundness ledger obligation)**.** *A* $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledger is accepted only when every rule step consumes formula endpoints already carried by Definition 197.1 and records its premise/conclusion transport by* Cont *and* hsame*. The ledger does not validate a rule by appeal to an external proof system; validity is the finite display of named endpoint consumption and transport.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier deduction soundness ledger obligation`

Proof. Induct over the finite displayed deduction ledger.

In the empty case there is no rule step to validate. The ledger acceptance condition reduces to the visibility of the initial formula endpoints already carried by the syntax packet.

In the step case, the induction hypothesis gives accepted carried endpoints for all earlier premises. The next rule row must cite only those endpoints, record the conclusion endpoint as another displayed formula row, and attach the Cont and hsame transport rows that move premises to the conclusion inside the same $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ provenance.

Appending the checked step preserves acceptance because the new ledger tail adds one finite row whose inputs, output, and transport are all visible BEDC rows. Thus the completed deduction remains a named ledger rather than a host derivation. □

Theorem 197.8 (FirstOrderUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *surface consists exactly of the syntax carrier of Definition 197.1, the deduction classifier of Definition 197.2, the formula carrier obligation of Definition 197.3, the deduction-soundness ledger of Definition 197.7, the* $\mathsf{Set}^{\uparrow}$ *symbol provenance, the* $\mathsf{Tree}^{\uparrow}$ *syntax provenance, stability under* hsame*, and exactness of displayed formula and deduction endpoints.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier namecert obligation surface`

Proof. Collect the carrier rows first. The syntax carrier supplies the displayed BHist packet, and the formula carrier obligation projects from it the symbol source, parse source, arity ledger, and endpoint relation.

Next collect the comparison rows. The deduction classifier supplies the rowwise hsame transport through the same $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ certificates, while the deduction-soundness ledger supplies the finite rule steps that consume already carried endpoints and produce displayed conclusion endpoints.

It remains to check public coverage. Any consumer of $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ asks for a syntax endpoint, a deduction endpoint, provenance in $\mathsf{Set}^{\uparrow}$ or $\mathsf{Tree}^{\uparrow}$, transport under hsame, or exact readback of a visible formula or deduction row. Each request is one of the rows just collected, so the public surface contains no additional carrier, classifier, or proof-system field. □

Definition 197.9 (FirstOrderUp deduction ledger concatenation source)**.** *Fix one carried* $\mathsf{FirstOrder}^{\uparrow}$ *syntax packet as in Definition 197.1. Let D_1 and D_2 be accepted finite* $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledgers read through the same* $\mathsf{Set}^{\uparrow}$ *and* $\mathsf{Tree}^{\uparrow}$ *certificates. They are composable when the final conclusion endpoint row displayed by D_1 is the same formula endpoint row that appears as the initial premise row of D_2, with that shared row compared only by the* hsame *transport already admitted by Definition 197.2.*

In that situation, the concatenation $D_1 \mathbin{++} D_2$ is the finite ledger obtained by listing all rule rows of D_1 followed by all rule rows of D_2, while reading the join through the shared formula endpoint row rather than through any extra host-level formula token or external proof-tree node.

Theorem 197.10 (FirstOrderUp deduction ledger concatenation closure)**.** *Let D_1 and D_2 be accepted finite* $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledgers over one carried syntax packet from Definition 197.1, and suppose that D_1 and D_2 are composable in the sense of Definition 197.9. Then the concatenated ledger $D_1 \mathbin{++} D_2$ is again an accepted* $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledger over the same* $\mathsf{Set}^{\uparrow}$ *and* $\mathsf{Tree}^{\uparrow}$ *dependency certificates.*

Moreover, every premise row, conclusion row, and transport row appearing in $D_1 \mathbin{++} D_2$ is already one of the visible rows carried by D_1, by D_2, or by their shared endpoint formula row. No extra deduction rule, external cut formula, or ambient proof-system datum enters at the join.

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderDeductionLedgerConcatenation closure`

Proof. Proceed by induction on the second ledger D_2, read as the finite displayed deduction ledger from Definition 197.7.

If D_2 is empty, then $D_1 \mathbin{++} D_2$ is just D_1, so acceptance is the given acceptance of D_1.

For the step case, write D_2 as a shorter accepted prefix E followed by one final rule row r. By the induction hypothesis, $D_1 \mathbin{++} E$ is an accepted deduction ledger over the same $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ certificates.

It remains to append r. Because E followed by r is accepted, Definition 197.7 says that r consumes only formula endpoints already carried by the displayed ledger before it, and that its premise/conclusion transport is recorded by visible Cont and hsame rows inside the same provenance. Every premise of r that lies wholly inside E is therefore already present in the accepted prefix $D_1 \mathbin{++} E$.

The only additional possibility is a premise of r that traces back to the initial premise row of D_2. By composability from Definition 197.9, that initial premise row is the shared endpoint formula row at the join with D_1. By Definition 197.3, this shared endpoint is already a carried $\mathsf{FirstOrder}^{\uparrow}$ formula row inside the same syntax packet, so the join introduces no extra carrier data. By Definition 197.2, any comparison needed at the join is exactly the admitted hsame transport through the same $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ certificates.

Thus all inputs required by r are available in the accepted prefix $D_1 \mathbin{+\!\!+} E$, and the output row of r is recorded by the same visible conclusion and transport data it had inside D_2. Appending r therefore preserves acceptance by the same step-case criterion used in Definition 197.7.

The induction closes, so $D_1 \mathbin{+\!\!+} D_2$ is an accepted finite $\mathsf{FirstOrder}^{\uparrow}$ deduction ledger over the original $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ certificates. The final claim follows because concatenation only reorders the already displayed rows by placing the rows of D_2 after those of D_1, with the join read through the shared endpoint formula row and no further datum. □

Theorem 197.11 (FirstOrderUp hsame stability obligation)**.** *For the public* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *surface, every classifier-related syntax or deduction packet transports its visible endpoints by the displayed* hsame *data. Formula endpoints, deduction-step premise and conclusion endpoints,* $\mathsf{Set}^{\uparrow}$ *symbol rows,* $\mathsf{Tree}^{\uparrow}$ *parse rows, and* Cont *ledger rows remain accepted after transport exactly when the rows are those projected from Definition 197.1 and compared by Definition 197.2.*

Proof. Project the deduction-ledger classifier of Definition 197.2. Its comparison is rowwise, so the relation-symbol and function-symbol rows move through the $\mathsf{Set}^{\uparrow}$ source, while parse subtrees and formula endpoints move through the $\mathsf{Tree}^{\uparrow}$ source.

For a deduction step, use the same classifier row on the displayed premise and conclusion endpoints. The step is accepted only with the attached Cont row and the corresponding hsame row, so transport does not create a new rule premise or a new conclusion outside the ledger.

The carrier rows then identify the endpoints being transported. By Definition 197.1, each formula endpoint and each deduction endpoint cited by the ledger is a row of the same displayed BHist packet. Thus stability is exactly the classifier transport applied to already carried rows. □

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier hsame stability obligation`

Theorem 197.12 (FirstOrderUp endpoint exactness obligation)**.** *Every public* $\mathsf{FirstOrder}^{\uparrow}$ *formula or deduction endpoint readback is exactly one of the endpoints projected from the syntax carrier and the deduction ledger. Conversely, every projected formula endpoint, premise endpoint, conclusion endpoint, and ledger endpoint repacks into the public* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *endpoint surface described by Definition 197.8.*

Proof. For the forward direction, read a public formula endpoint through the carrier of Definition 197.1. The readback exposes the displayed formula endpoint φ, together with its $\mathsf{Set}^{\uparrow}$ symbol row, $\mathsf{Tree}^{\uparrow}$ parse row, finite arity ledger, and SigRel endpoint relation.

A public deduction endpoint is read through the deduction ledger. By Definition 197.7, every premise and conclusion consumed by a rule step is already a carried formula endpoint, and the step records its own Cont and hsame transport rows. Hence no readback endpoint lies outside the projected carrier and ledger.

For the reverse direction, start with a projected carrier or ledger endpoint. The NameCert surface theorem, Definition 197.8, collects exactly these syntax endpoints, deduction endpoints, provenance rows, transport rows, and exact readback rows. Repacking the projection through that surface gives the corresponding public endpoint, and the two constructions expose the same displayed row. □

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier endpoint exactness obligation`

Theorem 197.13 (FirstOrderUp coverage obligation)**.** *Every accepted public* $\mathsf{FirstOrder}^{\uparrow}$ *request is covered by one of the listed certificate rows:* $\mathsf{Set}^{\uparrow}$ *symbol provenance,* $\mathsf{Tree}^{\uparrow}$ *parse provenance, finite* ProbeBundle *arity data, a* SigRel *endpoint relation, a* Cont *transport row, or a* Pkg *provenance*

row. No accepted request requires an additional carrier, classifier, stability, exactness, or coverage field outside the chapter surface.

Proof. Analyze an accepted public request by the part of the $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ surface it asks to read.

1. A syntax endpoint request is covered by Definition 197.1 and Definition 197.3. The former supplies the BHist packet and Pkg provenance; the latter projects the $\mathsf{Set}^{\uparrow}$ symbol row, $\mathsf{Tree}^{\uparrow}$ parse row, finite ProbeBundle arity rows, and SigRel endpoint relation.
2. A deduction endpoint request is covered by Definition 197.7. Each rule step consumes previously carried formula endpoints and records the displayed premise, conclusion, Cont, and hsame rows.
3. A transport request is covered by Definition 197.11. The classifier transports only the already projected symbol, parse, endpoint, and ledger rows, so the request stays inside the displayed carrier and deduction surface.
4. A provenance request is covered by the $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ NameCert rows named in Definition 197.8, together with the Pkg row carried in Definition 197.1.

These cases exhaust the public request forms listed by the NameCert surface theorem. Therefore each accepted request is accounted for by an existing displayed BEDC row. □

Definition 197.14 (FirstOrderUp satisfaction BHist carrier)**.** *A carried* $\mathsf{FirstOrder}^{\uparrow}$ *satisfaction packet is a displayed* BHist *tuple over five visible rows: the* $\mathsf{Set}^{\uparrow}$ *symbol row, the* $\mathsf{Tree}^{\uparrow}$ *formula row, the assignment and environment row, the interpretation row for relation and function symbols, and the finite provenance row. The provenance row records the* Cont*,* hsame*, and* Pkg *data that connect the symbol names, formula endpoint, variable assignment, and interpretation entries to the same visible syntax packet of Definition 197.1.*

The satisfaction predicate is the row classifier of this packet. It accepts only those reads whose formula endpoint is carried by the syntax row, whose free variables are read through the assignment/environment row, whose atomic relation and function entries are read through the interpretation row, and whose recursive connective and quantifier steps are displayed by finite Cont *and* hsame *transport. No host model carrier, ambient truth relation, or unrecorded semantic equality is part of the carrier.*

Lean4 (definition) `BEDC.Derived.FirstOrderUp.FirstOrderBHistSatisfactionCarrier`

Theorem 197.15 (FirstOrderUp soundness transport row)**.** *For every accepted* $\mathsf{FirstOrder}^{\uparrow}$ *deduction step, satisfaction rows transport from the displayed premise packets to the displayed conclusion packet using only the deduction ledger,* Cont*,* hsame*,* $\mathsf{Set}^{\uparrow}$*,* $\mathsf{Tree}^{\uparrow}$*, and* NameCert *provenance recorded in Definition 197.14. Equivalently, if every premise endpoint of a rule step carries a satisfaction packet over the same symbol, assignment/environment, and interpretation rows, then the conclusion endpoint carries the satisfaction packet produced by the rule row and its displayed transport data.*

Proof. Induct over the finite deduction ledger.

In the empty ledger there is no rule step to transport. The only available reads are the carried syntax and satisfaction rows already present in the visible packet.

For a nonempty ledger, apply the induction hypothesis to the earlier rule rows. This supplies satisfaction packets for every previously accepted premise endpoint, all over the same $\mathsf{Set}^{\uparrow}$ symbol row, $\mathsf{Tree}^{\uparrow}$ formula row, assignment/environment row, interpretation row, and provenance row.

The next displayed rule step names its premise endpoints and its conclusion endpoint inside the deduction ledger. Project the accompanying Cont and hsame rows. These rows transport the premise satisfaction packets across the carried formula endpoints and repack the result as the conclusion satisfaction packet.

Appending the transported conclusion preserves the ledger invariant because the new row cites only displayed premises, the displayed rule datum, and the same $\mathsf{Set}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, and NameCert provenance. Thus soundness is a finite row transport statement, not an appeal to an unrecorded proof system or model class. □

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSatisfactionCarrier soundness transport row`

Theorem 197.16 (FirstOrderUp completeness boundary)**.** *The* $\mathsf{FirstOrder}^{\uparrow}$ *completeness surface is a BEDC boundary: every public completeness readback exposes a carried syntax row, the satisfaction carrier row of Definition 197.14, a deduction ledger row, and the* Pkg *provenance tying those rows to* $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$*. A readback that requires a host model class, an unrecorded semantic relation, or an external proof system is outside the public* $\mathsf{FirstOrder}^{\uparrow}$ *certificate.*

Proof. Project a public completeness readback from the displayed $\mathsf{FirstOrder}^{\uparrow}$ packet. Its syntax component is one of the carried formula rows of Definition 197.1, and its derivation component is one of the finite rows of Definition 197.2.

Project the semantic component through Definition 197.14. This exposes the assignment/environment row, the interpretation row, and the finite Cont/hsame/Pkg provenance that connects semantic reads to the same displayed formula endpoint.

It remains to check that the readback is complete for the public surface and no larger. Induct over the finite deduction ledger. Initial rows are exactly the carried syntax and satisfaction packets; each rule row is handled by the transport statement of Definition 197.15; the final row is therefore a displayed conclusion packet with the same package provenance.

Any proposed readback field not obtained by these projections has no displayed BHist row, no Cont or hsame transport, and no Pkg route back to $\mathsf{NameCert}_{\mathsf{Set}^{\uparrow}}$ or $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$. Such a field is not admitted by the $\mathsf{FirstOrder}^{\uparrow}$ certificate. □

Theorem 197.17 (FirstOrderUp satisfaction endpoint soundness)**.** *A satisfaction endpoint row for a carried* $\mathsf{FirstOrder}^{\uparrow}$ *formula is sound only when the formula endpoint, the model/source row, and the semantic result row are all displayed* BHist *rows. The three rows must be transported through the same* hsame *and* Cont *ledgers, observed by finite* ProbeBundle *and* SigRel *data, and grounded in the* $\mathsf{Set}^{\uparrow}$*,* $\mathsf{Tree}^{\uparrow}$*, and* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *provenance exposed by Definition 197.8.*

Lean4 ✓ `BEDC.Derived.FirstOrderUp.FirstOrderBHistSatisfactionCarrier satisfaction endpoint soundness`

Proof. Project the carried syntax packet of Definition 197.1. The formula endpoint φ is already a displayed BHist row with $\mathsf{Set}^{\uparrow}$ symbol provenance and $\mathsf{Tree}^{\uparrow}$ parse provenance.

Next project the satisfaction packet. Its model/source row is accepted only as a named source row compatible with the same relation and function symbol rows, and its semantic result row is the endpoint reached from the formula and source rows by the displayed Cont ledger. The finite ProbeBundle probes record the arity and source observations used by that ledger, while SigRel records the endpoint relation being read.

Finally transport the three rows together by the carried hsame witnesses. Because the public surface of Definition 197.8 exports only these rows and their provenance, satisfaction soundness is exactly endpoint soundness inside the BEDC ledger and not a host-level semantic assertion. □

Theorem 197.18 (FirstOrderUp substitution transport obligation)**.** *Let a carried* $\mathsf{FirstOrder}^{\uparrow}$ *formula endpoint contain a variable row or subtree row that is replaced by another carried endpoint related by the deduction classifier of Definition 197.2. Formula acceptance and deduction-ledger acceptance are preserved precisely when every affected symbol, parse, premise, and conclusion row transports rowwise by the inherited* hsame *witnesses and the displayed* Cont *ledgers.*

Proof. Induct on the displayed $\mathsf{Tree}^{\uparrow}$ parse row of the carried formula.

In a leaf case, the replaced variable or constant endpoint is a row in the variable or symbol ledger. The classifier-related replacement supplies the same $\mathsf{Set}^{\uparrow}$ provenance and an hsame witness, so the leaf remains an accepted formula endpoint after the one-row Cont transport.

In an internal subtree case, apply the induction hypothesis to each affected child endpoint. The parent relation or function row is fixed by the $\mathsf{Set}^{\uparrow}$ classifier, and the $\mathsf{Tree}^{\uparrow}$ provenance repacks the transported children as the transported parent endpoint.

For a deduction ledger, induct over the finite rule list. Premise endpoints transport by the formula argument above, and each rule row records its conclusion through the same displayed Cont ledger. Appending the transported rule row preserves ledger acceptance because no step consumes an endpoint outside the rowwise hsame comparison supplied by the classifier. □

Theorem 197.19 (FirstOrderUp completeness boundary obligation)**.** *Any completeness datum exposed by* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *is a displayed BEDC ledger from semantic endpoint agreement back to a finite deduction ledger. It is not host model-theoretic completeness, and it is not an external proof-system oracle: the datum must begin with displayed satisfaction endpoint agreement and end with a finite deduction ledger accepted by Definition 197.7.*

Proof. Project the public surface of Definition 197.8. The available rows are the syntax carrier, the deduction classifier, the formula carrier obligation, the deduction-soundness ledger, and the $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ provenance.

Now read a completeness datum as a finite reverse ledger. Its input is not a class of all host models; it is semantic endpoint agreement between displayed satisfaction rows of Definition 197.17. Each reverse step must name the formula endpoint it consumes, the semantic row it reads, and the deduction endpoint it contributes.

Induct over this finite reverse ledger. The empty case returns the displayed initial deduction ledger. In the step case, the induction hypothesis supplies the earlier finite ledger, and the next reverse row appends one accepted deduction step with its carried premise and conclusion endpoints. The result is therefore a BEDC deduction ledger, which states the completeness boundary without importing an external model theory or proof-system oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$**.**

Constructive story: A first-order-logic package is built as a displayed syntax tuple whose histories name the $\mathsf{Set}^{\uparrow}$ symbol source, the $\mathsf{Tree}^{\uparrow}$ parse-tree source, variable, relation-symbol, and function-symbol ledgers, a finite term/formula endpoint, a formula endpoint, and the package provenance tying them to the certificate. The carrier accepts only formulas and deduction endpoints that can be read through those BHist, hsame, Cont, finite probe, and signature-relation rows, while host strings, host grammar tables, external proof-system validity, and semantic model carriers are excluded. Two syntax or deduction packets are classified by rowwise transport of symbol rows through $\mathsf{Set}^{\uparrow}$, parse and formula rows through $\mathsf{Tree}^{\uparrow}$, and every premise and conclusion through the displayed continuation and history-equality ledger. The source is the set-and-tree certificate pair, the pattern records formula construction, arity probes, substitution and deduction-step transport, satisfaction endpoint boundaries, and finite proof ledgers, and the ledger keeps the symbol provenance, parse provenance, rule steps, conclusion endpoints, and package rows visible. The name-certificate obligations follow because an accepted formula packet gives a carrier witness, rowwise hsame supplies reflexive, symmetric, and transitive classification, and classifier transport repacks every transported symbol, subtree, formula, and deduction step as another accepted packet. A packet h lies in the signature gap of a $\mathsf{FirstOrder}^{\uparrow}$ token p when the source-domain accepts its symbol, parse, formula, and deduction ledgers, the signature relation pairs that endpoint with an answer formula or deduction endpoint and evidence

ledger, and the token-introduction relation names the answer as p; unfolded, p records the same displayed syntax tree and finite deduction ledger. The chapter registers this syntax-and-deduction surface under the name $\mathsf{FirstOrder}^{\uparrow}$; it introduces no host syntax equality, model-theoretic completeness theorem, or external proof oracle.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirstOrder}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper package binds the syntax-and-deduction surface, satisfaction endpoint boundary, substitution transport, finite deduction ledger, and explicit exclusion of host syntax equality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FirstOrder}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FirstOrderUp.FirstOrderBHistSyntaxCarrier\_formula\_carrier\_obligation`

Bridge status: paperBridge

Not claimed: Host syntax equality and ModelTheoryUp completeness are not claimed by this chapter.

Upgrade path: No further paper-axis closure is claimed here; bridgeChecked remains a separate verification-axis target.

Chapter 198

A Concrete Naming Certificate for $\mathsf{ModelTheory}^{\uparrow}$

The model-theory interface $\mathsf{ModelTheory}^{\uparrow}$ packages a structure for a first-order signature, satisfaction of formulas, elementary equivalence, and elementary substructures. It depends on $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ (chapter 197).

198.1 Carrier definition

Definition 198.1 (ModelTheoryUp BHist structure packet)**.** *A* $\mathsf{ModelTheory}^{\uparrow}$ *source packet is a finite displayed* BHist *tuple* $(S, A, \mathsf{Val}, \mathsf{Sat}, \mathsf{Elem}, \rho, \lambda)$*. The row* S *is the* $\mathsf{FirstOrder}^{\uparrow}$ *signature and formula row,* A *is the displayed structure row,* Val *records variable-assignment observations,* Sat *records the satisfaction ledgers for named formulas and assignments, and* Elem *records elementary-equivalence and elementary-substructure faces. The row* ρ *is the* Pkg *provenance tying the packet to the visible certificate, and* λ *is the* Cont *and* hsame *transport ledger.*

Acceptance of the packet exports only these displayed rows. A host satisfaction relation, an ambient model equality, or an external comparison of structures is not part of the carrier.

Definition 198.2 (ModelTheoryUp satisfaction ledger classifier)**.** *Two* $\mathsf{ModelTheory}^{\uparrow}$ *source packets classify the same row exactly when* hsame *transports their* $\mathsf{FirstOrder}^{\uparrow}$ *signature rows, displayed carrier rows, valuation rows, satisfaction ledgers, elementary-equivalence rows, and* Pkg *provenance through the same* Cont *ledger. The comparison is therefore rowwise: formula and substitution reads move through the* $\mathsf{FirstOrder}^{\uparrow}$ *source, assignments move through* Val*, truth records move through* Sat*, and elementary faces move through* Elem*.*

Definition 198.3 (ModelTheoryUp satisfaction exactness row)**.** *The satisfaction exactness row of a* $\mathsf{ModelTheory}^{\uparrow}$ *packet is the finite* BHist *readback row whose entries are triples consisting of a formula endpoint, an assignment endpoint, and a truth ledger face. Such an entry is accepted exactly when the formula endpoint is read through the* $\mathsf{FirstOrder}^{\uparrow}$ *row* S*, the assignment endpoint is read through* Val*, and the truth ledger is one of the displayed* Sat *faces carried by the same packet.*

The row is exported only with its packet provenance ρ *and its comparison ledger* λ*. It does not contain a host satisfaction predicate, a semantic evaluation function, or an equality of ambient structures.*

Theorem 198.4 (ModelTheoryUp satisfaction exactness scope)**.** *Every readback accepted by Definition 198.3 is scoped by the displayed rows* S*,* Val*,* Sat*, the packet provenance* ρ*, the transport*

ledger λ, and the available hsame *comparisons. No accepted satisfaction readback requires a host satisfaction relation.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket satisfaction exactness scope`

Proof. Project an accepted readback from the exactness row. Its first coordinate is a formula endpoint, so the packet carrier and Definition 198.7 place it in the $\mathsf{FirstOrder}^{\uparrow}$ row S rather than in an independent syntax source.

Project the second coordinate. By the defining acceptance clause it is a displayed assignment read from Val, with the same packet provenance ρ and the same comparison ledger λ.

Project the third coordinate. It is one of the displayed Sat faces, and any comparison of that face is supplied by the rowwise hsame data of Definition 198.2. Repacking these three projections gives the stated scope and leaves no place for a host satisfaction relation. □

Theorem 198.5 (ModelTheoryUp elementary ledger coverage)**.** *The elementary-equivalence and elementary-substructure requests admitted by a* $\mathsf{ModelTheory}^{\uparrow}$ *packet are covered exactly by the displayed* Elem *faces whose formula endpoints are read through S and whose satisfaction references pass through Definition 198.3. No elementary request is accepted outside those faces.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket elementary ledger coverage`

Proof. For an admitted elementary request, project the packet carrier of Definition 198.1. The request must be one of the finite Elem faces, because elementary-equivalence and elementary-substructure data are exported only by that row.

Inside that face, each formula endpoint is a read through the first-order row S. Apply Definition 198.7 to identify that endpoint with the $\mathsf{FirstOrder}^{\uparrow}$ syntax surface, and apply Definition 198.3 to the corresponding satisfaction record.

Conversely, a displayed Elem face whose formula endpoints lie in S and whose satisfaction references are accepted by the exactness row is already one of the exported elementary ledger entries of the carrier packet. Repacking the face gives the elementary request, and no further row is available from the carrier. □

Theorem 198.6 (ModelTheoryUp carrier classifier stability)**.** *Classifier transport preserves both the satisfaction exactness row and the elementary ledger. The preservation is rowwise: formula endpoints, assignment endpoints, satisfaction faces, elementary faces, provenance, and comparison ledgers are transported by* hsame*, and entrywise* Cont *comparison identifies the transported ledger entries with the displayed target entries.*

Proof. Start from the classifier relation of Definition 198.2. It gives rowwise hsame transport for S, A, Val, Sat, Elem, ρ, and the Cont ledger λ.

For a satisfaction exactness entry, transport its formula endpoint along the S comparison, its assignment endpoint along the Val comparison, and its truth face along the Sat comparison. Entrywise Cont comparison identifies the transported provenance with the target packet, so the transported triple is accepted by the target exactness row.

For an elementary ledger entry, transport the displayed Elem face and then apply the previous paragraph to every satisfaction reference inside that face. The resulting face has the same formula scope and the same entrywise ledger comparison in the target packet, so the classifier preserves the elementary ledger as well. □

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket carrier classifier stability`

198.2 The certificate

Theorem 198.7 (ModelTheoryUp FirstOrder dependency surface)**.** *Every admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packet projects its signature row, term/formula grammar row, and substitution-read rows through* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$*. The structure, valuation, satisfaction, and elementary rows may consume those projected first-order endpoints, but they do not export an independent first-order syntax source.*

Proof. Invert the carried packet of Definition 198.1. The first projection is the row S, so the available symbols, terms, formulas, and substitution reads are exactly the rows already visible from $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$.

Project the displayed structure row A and the valuation row Val. Each assignment observation names variables and formula endpoints by referring back to S, so no separate grammar or substitution operation appears in the model-theory carrier.

Finally project the satisfaction and elementary ledgers. Their formula occurrences are read through the same S row, and their provenance is the same Pkg row ρ. Repacking these projections gives the admitted $\mathsf{ModelTheory}^{\uparrow}$ packet with the first-order dependency surface unchanged. □

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket firstorder dependency surface`

Theorem 198.8 (ModelTheoryUp elementary transport)**.** *If two admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packets are related by* hsame *transport of their* $\mathsf{FirstOrder}^{\uparrow}$ *rows and satisfaction ledgers, then their elementary-equivalence and elementary-substructure rows transport as displayed* Elem *rows. This transport uses no host model equality.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket elementary transport`

Proof. Start with the classifier relation of Definition 198.2. It supplies the hsame rows for the first-order source, the displayed structure row, the valuation ledger, the satisfaction ledger, the elementary ledger, and the Pkg provenance.

Use the transported $\mathsf{FirstOrder}^{\uparrow}$ row to move formula endpoints and substitution reads. Use the transported Sat row to move each satisfaction record for the corresponding assignment observation.

The elementary-equivalence and elementary-substructure faces are finite records in Elem. Project those records from the source packet, replace each formula and satisfaction reference by the transported row just obtained, and repack them as the target Elem row. No step compares ambient carriers by a host equality; every comparison is one of the displayed hsame rows. □

Theorem 198.9 (ModelTheoryUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ *obligation surface consists exactly of the carrier of Definition 198.1, the classifier of Definition 198.2, the first-order dependency surface of Definition 198.7, the elementary transport of Definition 198.8, stability transport, the satisfaction exactness ledger, the elementary-equivalence ledger, and no additional exported field.*

Proof. Collect the carrier rows first. The finite BHist packet provides S, A, Val, Sat, Elem, ρ, and λ, while the first-order dependency theorem identifies all syntax and substitution reads as projections through $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$.

Next collect the classifier rows. The satisfaction ledger classifier gives rowwise hsame transport for the first-order source, the structure carrier, valuation observations, satisfaction records, elementary faces, and provenance. The elementary transport theorem repacks the transported satisfaction data into elementary-equivalence and elementary-substructure rows.

It remains to check public coverage. A consumer of $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ can ask for a carrier row, a classifier row, stability under hsame, exact readback of satisfaction records, or the elementary

ledger. Each request is one of the collected rows, and none exposes a host satisfaction predicate, an ambient model equality, or another public field outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket namecert obligation surface`

Definition 198.10 (ModelTheoryUp assignment observation row)**.** *A* $\mathsf{ModelTheory}^{\uparrow}$ *assignment observation row is the displayed* BHist *entry*

$$(S, \varphi, x, \mathsf{Val}_{x,\varphi}, \rho_{\mathsf{Val}}, \lambda_{\mathsf{Val}})$$

whose variable endpoint x *and formula endpoint* φ *are read only through* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$*. The valuation component* $\mathsf{Val}_{x,\varphi}$ *is the visible assignment row for that endpoint pair,* λ_{Val} *records the* Cont *and* hsame *transport available for the observation, and* ρ_{Val} *is the* Pkg *provenance tying the row back to the admitted packet of Definition 198.1.*

Theorem 198.11 (ModelTheoryUp satisfaction exactness row)**.** *Every accepted* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction record is exactly the visible* Sat *row for a formula endpoint read through* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and an assignment observation of Definition 198.10. Conversely, every such visible* Sat *row repacks as the satisfaction record for the same admitted packet. No host satisfaction predicate is exported.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket satisfaction exactness row`

Proof. Unpack the finite packet of Definition 198.1. The projections expose the first-order row S, the valuation ledger Val, the satisfaction ledger Sat, and the provenance row ρ.

The formula endpoint is read from S through $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$, while the assignment endpoint is read from the displayed valuation row described in Definition 198.10. Thus the satisfaction record has no data source other than the visible pair of endpoints and the packet provenance.

Repack those projections as the same Sat row with its Pkg, Cont, and hsame witnesses. This gives the exact readback direction, and the converse is the same repacking step starting from the displayed row. At no point is a semantic satisfaction relation outside the BHist packet introduced. □

Theorem 198.12 (ModelTheoryUp elementary substructure boundary)**.** *The elementary-equivalence and elementary-substructure faces of an admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packet are finite displayed* Elem *rows transported through the satisfaction exactness row and* hsame*. Every formula occurrence in those faces is a* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *endpoint, and every assignment occurrence is a transported valuation row.*

Proof. Begin with the classifier of Definition 198.2. It supplies the rowwise hsame transports for the first-order source, valuation ledger, satisfaction ledger, elementary ledger, and Pkg provenance.

Apply Definition 198.11 to each satisfaction record referenced by an elementary face. This replaces the record by the visible Sat row for a named formula endpoint and a named assignment observation.

It remains to move the elementary face itself. Replace every formula reference by the transported first-order row and every assignment reference by the transported Val row, then repack the finite face as the target Elem row. The boundary is therefore a displayed ledger boundary, not an ambient elementary-substructure predicate. □

Theorem 198.13 (ModelTheoryUp downstream consumer row)**.** *A downstream consumer of* $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ *can ask only for the carrier packet, the satisfaction-ledger classifier, the assignment observation row, the satisfaction exactness row, the elementary substructure boundary, stability transport, and* Pkg *provenance. Any request for a host truth predicate, an ambient model equality, or an external elementary comparison lies outside the public obligation surface.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket downstream consumer row`

Proof. Project the public surface of Definition 198.9. The visible rows are the carrier packet, the classifier, first-order dependency rows, elementary transport rows, stability transport, satisfaction readback, and provenance.

Refine the satisfaction readback by Definition 198.11. Refine the assignment input by Definition 198.10, and refine the elementary request by Definition 198.12.

These refinements exhaust the consumer-facing rows. Since each row is a projection from the displayed packet or a repacking through Cont, hsame, and Pkg, no consumer receives an operation beyond the listed certificate surface. □

Definition 198.14 (ModelTheoryUp elementary readback ledger)**.** *The elementary readback ledger of an admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packet is the finite displayed* Elem *row of Definition 198.1. Its entries record only elementary-equivalence and elementary-substructure observations whose formula endpoints are read through the transported* $\mathsf{FirstOrder}^{\uparrow}$ *source row, whose truth witnesses are entries of the displayed* Sat *row, and whose provenance is the same* Pkg *row* ρ *carried by the packet.*

Theorem 198.15 (ModelTheoryUp satisfaction exactness ledger)**.** *For any admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packet, every satisfaction record in* Sat *reads back exactly through the displayed* $\mathsf{FirstOrder}^{\uparrow}$ *formula row and the displayed valuation row* Val*. The readback exports the transported formula endpoint, the transported valuation endpoint, the recorded truth entry, and the* Pkg *provenance, and it exports no host satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket satisfaction exactness ledger`

Proof. Begin with the packet of Definition 198.1. Its displayed rows include the first-order source S, the valuation ledger Val, the satisfaction ledger Sat, and the provenance row ρ.

Apply the classifier of Definition 198.2. A satisfaction entry is classified rowwise, so its formula endpoint is transported along the $\mathsf{FirstOrder}^{\uparrow}$ row and its assignment endpoint is transported along Val.

Use Definition 198.7 to identify the formula endpoint as a projection through $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$. The truth entry is then the displayed Sat record itself, and its source provenance is the carried Pkg row.

Finally apply Definition 198.8 and Definition 198.9 only to check that this projection lies on the public obligation surface. Those theorems add no host satisfaction predicate, so the readback is exactly the displayed row data. □

Theorem 198.16 (ModelTheoryUp elementary readback boundary)**.** *For any admitted* $\mathsf{ModelTheory}^{\uparrow}$ *packet, elementary-equivalence and elementary-substructure readback is exhausted by the elementary readback ledger of Definition 198.14. In particular, the boundary of the readback consists of the transported* $\mathsf{FirstOrder}^{\uparrow}$ *formula endpoints,* Val *assignment endpoints,* Sat *truth entries,* Elem *elementary faces, and* Pkg *provenance, and it uses no host model equality.*

Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket elementary readback boundary`
Lean4 ✓ `BEDC.Derived.ModelTheoryUp.ModelTheoryElementaryReadbackLedgerSpine normalized cont`

Proof. Project the elementary row from the carrier of Definition 198.1. By definition Definition 198.14, this row is the finite displayed Elem ledger attached to the packet.

The classifier of Definition 198.2 supplies the only allowed transport data: S, A, Val, Sat, Elem, and ρ. Hence every formula occurrence is read through the $\mathsf{FirstOrder}^{\uparrow}$ source row, every assignment occurrence through Val, and every truth occurrence through Sat.

Apply Definition 198.7 for the first-order projection and Definition 198.8 for repacking the elementary-equivalence and elementary-substructure faces. The public surface theorem Definition 198.9 then rules out any additional exported comparison field.

Thus the elementary readback boundary is exactly the displayed Elem ledger together with its named source rows, and no step requires an equality of ambient host models. □

Theorem 198.17 (ModelTheoryUp downstream consumer interface)**.** *Any downstream consumer of* $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ *can consume only the first-order source row, the displayed structure row, the valuation row, the satisfaction exactness row of Definition 198.15, the elementary readback row of Definition 198.14, and the* Pkg *provenance. No downstream interface can demand a host satisfaction predicate, host model equality, or an extra carrier outside those rows.*

Proof. Start from the public obligation surface of Definition 198.9. It lists the carrier, classifier, first-order dependency surface, elementary transport, stability, satisfaction exactness, elementary ledger, and no additional exported field.

Unpack the carrier using Definition 198.1. The consumable rows are S, A, Val, Sat, Elem, ρ, and the transport ledger. The first-order theorem Definition 198.7 identifies S as the only source of formula and substitution reads.

Use the classifier of Definition 198.2 to restrict transport to the displayed rows. The satisfaction exactness theorem gives the allowed truth-record readback, while Definition 198.8 and Definition 198.14 give the allowed elementary readback.

Therefore every downstream request factors through one of the listed rows or through Pkg provenance. Since the obligation surface has no remaining field, no consumer can obtain a host satisfaction predicate, host model equality, or another carrier from $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$**.**

Constructive story: A model-theory packet is generated by taking a $\mathsf{FirstOrder}^{\uparrow}$ signature-and-formula row, a displayed structure row, valuation observations, satisfaction faces, elementary-equivalence and elementary-substructure faces, package provenance, and a continuation/history-equality transport ledger. The carrier accepts only finite BHist tuples whose formulas and substitutions are read through the first-order source and whose valuation, satisfaction, and elementary rows are displayed in the same packet; host satisfaction relations, ambient model equality, and external structure comparisons are not admitted. Two packets are classified rowwise: formula and substitution reads move through $\mathsf{FirstOrder}^{\uparrow}$, assignments through the valuation row, truth records through the satisfaction ledger, elementary faces through the elementary row, and provenance through the same Cont ledger. The source is the first-order certificate surface, the pattern records structure, valuation, satisfaction exactness, elementary readback, and consumer handoff, and the ledger keeps the first-order row, assignment endpoints, truth faces, elementary faces, Pkg provenance, and continuation comparisons visible. The five name-certificate obligations hold because any accepted structure packet gives habitation, the rowwise history classifier is reflexive, symmetric, and transitive, and carrier-classifier stability transports satisfaction exactness and elementary ledgers into a repacked target packet. A packet h lies in the signature gap of a $\mathsf{ModelTheory}^{\uparrow}$ token p when the source-domain accepts its first-order, structure, valuation, satisfaction, and elementary rows, the signature relation pairs the packet with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records exactly those displayed truth and elementary-readback faces. The chapter registers this finite satisfaction-and-elementary ledger under the name $\mathsf{ModelTheory}^{\uparrow}$; it does not introduce completeness, compactness, Lowenheim–Skolem, host model equality, or a standard bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{ModelTheory}^{\uparrow}}$, maturePackageClosed)

Scope closed: The paper bridge boundary runs from FirstOrder$^{\uparrow}$ syntax rows to the ModelTheory$^{\uparrow}$ satisfaction and elementary-equivalence ledger through valuation, readback, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ModelTheory}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ModelTheoryUp.ModelTheoryBHistStructurePacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close compactness, Lowenheim-Skolem, host model equality, or bridgeChecked verification.

Upgrade path: BridgeChecked verification requires a checked standard bridge target.

Theorem 198.18 (ModelTheoryUp elementary-equivalence reflexivity row)**.** *Let* $(S, A, \mathsf{Val}, \mathsf{Sat}, \mathsf{Elem}, \rho, \lambda)$ *be an admitted* ModelTheory$^{\uparrow}$ *source packet in the sense of Definition 198.1. Then the displayed structure row* A *is classified with itself by the elementary-equivalence part of* Elem*. The diagonal witness is obtained by using* hsame*-reflexivity on the displayed* S*,* A*,* Val*,* Sat*,* ρ*, and* λ *rows, and by reading every formula-assignment truth comparison through Definition 198.3 with identical formula and assignment endpoints. No equality of host models is used.*

Proof. Project the admitted packet. The available rows are exactly S, A, Val, Sat, Elem, ρ, and λ by Definition 198.1; in particular, the carrier exports no ambient model equality and no host satisfaction predicate.

Use reflexivity of hsame from Definition 3.8 on each displayed row of the packet. This gives the trivial transport from S to S, from A to A, from Val to Val, from Sat to Sat, and from the provenance and comparison ledgers to themselves. Thus the two structure endpoints of the elementary-equivalence request are the same displayed row A, compared only by the packet's internal hsame ledger.

Fix any formula endpoint and assignment endpoint cited by the elementary face. Since the formula endpoint is read through S and the assignment endpoint is read through Val, the triple consisting of that formula endpoint, that assignment endpoint, and its displayed truth face is an entry of the satisfaction exactness row of Definition 198.3. Applying the same exactness row to the identical formula endpoint and identical assignment endpoint gives the matching truth readback on the other side of the diagonal comparison.

Now apply Definition 198.5. Its coverage criterion says that an elementary-equivalence request is accepted precisely through a displayed Elem face whose formula endpoints are read through S and whose satisfaction references pass through the satisfaction exactness row. The diagonal request just constructed has exactly those entries: the formula and assignment endpoints are unchanged, and each satisfaction reference is the exactness entry paired with itself.

Repacking these diagonal formula, assignment, satisfaction, provenance, and hsame rows gives the elementary-equivalence face in Elem classifying A with itself. Every component of the witness is a projected or reflexive row of the admitted packet, so the construction never invokes a host model identity. □

Chapter 199

A Concrete Naming Certificate for $\mathsf{Computable}^{\uparrow}$

The computable-function interface $\mathsf{Computable}^{\uparrow}$ packages partial functions $\mathbb{N} \to \mathbb{N}$ defined by Turing-machine simulation up to time-step bounds. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28) and $\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$ (chapter 120).

199.1 Carrier definition

199.2 The certificate

Definition 199.1 (Bounded simulation witness for $\mathsf{Computable}^{\uparrow}$)**.** *Fix a deterministic Turing-machine code P whose input and output tapes carry unary $\mathsf{Nat}^{\uparrow}$ histories. For unary histories n, B, and m, the judgment*

$$\mathsf{BoundedSim}(P, n, B, m)$$

means that there is a $\mathsf{Seq}^{\uparrow}$ trace C of configurations indexed by the unary prefixes of B such that:

1. *the zero-prefix entry is the initial configuration of P on input n;*
2. *every one-step extension in the prefix interval follows the transition relation of P, with halted configurations allowed to repeat;*
3. *the endpoint entry at bound B is halted and reads back the unary output m.*

The trace is a $\mathsf{Seq}^{\uparrow}$ object in the sense of chapter 120, while the bound and output are unary histories in the sense of Definition 28.1. Thus B is an upper bound, not an exact running time.

Definition 199.2 (Bounded $\mathsf{Computable}^{\uparrow}$ graph certificate)**.** *For a partial graph $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$, a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate consists of a deterministic machine code P_F and a $\mathsf{Seq}^{\uparrow}$ unary-bound modulus β_F such that, for all unary n and m,*

$$F(n, m) \iff \mathsf{BoundedSim}(P_F, n, \beta_F(n), m),$$

and any two bounded simulations for the same input return hsame*-classified output histories. The modulus value $\beta_F(n)$ is part of the certificate ledger, so each accepted graph value carries its displayed time-step bound.*

Lean4 (definition) `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate`

Theorem 199.3 (Computable carrier obligation)**.** *Let $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ carry a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate (P_F, β_F) in the sense of Definition 199.2. For every accepted graph row $F(n, m)$, the certificate exposes the unary source input n, the unary output m, the unary bound value $\beta_F(n)$, and a $\mathsf{Seq}^{\uparrow}$ trace carrier whose rows are configurations indexed by the unary prefixes of $\beta_F(n)$.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate carrier obligation`

Proof. Unpack the graph-equivalence clause in Definition 199.2. The accepted row $F(n, m)$ converts to a bounded simulation witness

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m).$$

Now read the witness data from Definition 199.1. Its input and endpoint clauses supply the unary source and unary output histories. Its bound argument is the displayed modulus value $\beta_F(n)$, and the witness itself is a $\mathsf{Seq}^{\uparrow}$ trace of configurations over the unary prefix interval of that bound. These are exactly the carrier rows visible at the public certificate surface. □

Theorem 199.4 (Computable classifier obligation)**.** *Let $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ carry a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate (P_F, β_F). Accepted outputs over the same unary input are classified by hsame: if $F(n, m)$ and $F(n, m')$, then $m \mathsf{hsame} m'$. Moreover, the identity graph $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$ uses exactly the unary $\mathsf{Nat}^{\uparrow}$ classifier, namely $n \mathsf{hsame} m$ between unary input and output histories.*

Proof. For the general certificate, apply Definition 199.13. It expands both accepted rows to bounded simulations at the same displayed bound and then applies the certificate's output-determinacy field, giving $m \mathsf{hsame} m'$.

For the identity graph, unfold Definition 199.14. Its graph predicate is precisely unary source membership, unary output membership, and the classifier clause $n \mathsf{hsame} m$. By Definition 28.27, this is the $\mathsf{Nat}^{\uparrow}$ unary classifier rather than host equality of history terms. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate graph output transport`

Theorem 199.5 (Computable stability obligation)**.** *The bounded $\mathsf{Computable}^{\uparrow}$ certificate interface is stable under identity and bounded-simulation composition. The identity surface is the empty-bound certificate for $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$, and two certified rows $F(n, m)$ and $G(m, k)$ compose to a bounded simulation from n to k with an explicit phase boundary.*

Proof. Identity stability is witnessed by Definition 199.17. Its graph-equivalence field is supplied by Definition 199.16, so the empty-bound halted code realizes exactly the unary classifier identity graph.

Composition stability is Definition 199.11. Given bounded simulations for $F(n, m)$ and $G(m, k)$, the proof concatenates their $\mathsf{Seq}^{\uparrow}$ traces along a displayed continuation witness and records the phase switch as part of the composite deterministic code. The resulting witness is again a bounded simulation, now from n to k. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim output hsame transport`

Theorem 199.6 (Computable ledger obligation)**.** *The visible ledger of a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate consists of the deterministic program code, the displayed unary bound value, the $\mathsf{Seq}^{\uparrow}$ trace, the phase boundary used for composition, and the empty bound used for identity.*

Proof. The deterministic program code and displayed bound are the two certificate fields (P_F, β_F) in Definition 199.2. For an accepted row $F(n, m)$, the graph-equivalence field exposes the concrete bound $\beta_F(n)$ and the bounded simulation witness at that bound.

The trace ledger is the $\mathsf{Seq}^{\uparrow}$ configuration trace named in Definition 199.1. Its zero-prefix row, transition rows, halted repetitions, and endpoint row are the observable rows carried by the witness.

For composition, Definition 199.11 records the phase machine $P_{G \circ F}$ and the boundary between the F trace and the G trace. For identity, Definition 199.15 fixes the displayed bound to $\beta_{\mathsf{id}}(n) = \mathsf{e}$, and Definition 199.17 reads that empty-bound ledger as the identity certificate. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim visible ledger rows`

Theorem 199.7 (Computable bounded graph source and observation carriers)**.** *Every bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate* (P_F, β_F) *for a partial graph* $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ *has the following carrier obligations. Its source carrier and output carrier are unary* $\mathsf{Nat}^{\uparrow}$ *histories in the sense of Definition 28.1. Its observation carrier is the* $\mathsf{Seq}^{\uparrow}$ *trace of configurations indexed by the unary prefixes of the displayed bound* $\beta_F(n)$*, as in chapter 120. Thus any accepted row* $F(n, m)$ *is represented by carrier data*

$$\mathsf{UnaryHistory}(n), \quad \mathsf{UnaryHistory}(\beta_F(n)), \quad \mathsf{SeqTrace}_{P_F}(n, \beta_F(n)), \quad \mathsf{UnaryHistory}(m).$$

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate source carrier obligation`

Proof. Unpack Definition 199.2. The certificate contains a deterministic code P_F and a displayed unary-bound modulus β_F. For a unary source history n, the modulus value $\beta_F(n)$ is a unary history, so the bound is an object of the same unary generation specified in Definition 28.1.

For an accepted row $F(n, m)$, the graph-exactness field gives the corresponding bounded simulation witness of Definition 199.1. That witness supplies a $\mathsf{Seq}^{\uparrow}$ trace of configurations over the unary prefixes of $\beta_F(n)$, with zero row, transition rows, and halted endpoint row. The endpoint readback is the output history m, so the output carrier is again unary by the same local witness and the $\mathsf{Nat}^{\uparrow}$ unary classifier. □

Theorem 199.8 (Computable bounded graph classifier stability)**.** *Let* (P_F, β_F) *be a bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate for* $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$*. If unary inputs* n *and* n' *are classified together by the* $\mathsf{Nat}^{\uparrow}$ *history classifier and* $F(n, m)$ *holds, then transporting the accepted graph row through that classifier preserves bounded simulation output classification. In particular, whenever* $F(n', m')$ *also holds, the two transported outputs satisfy*

$$m \mathsf{hsame} m'.$$

Consequently graph membership transported through the $\mathsf{Nat}^{\uparrow}$ *history classifier preserves the bounded simulation readback class and the single-valuedness of the certified graph.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate classifier stability obligation`

Proof. Convert the accepted row $F(n, m)$ to a bounded simulation by the exactness field of Definition 199.2. The source transport $n \mathsf{hsame} n'$ is a transport along the unary history classifier of Definition 28.1; the transported run uses the same deterministic code and the displayed modulus row assigned to the transported source.

If a second accepted row $F(n', m')$ is present, exactness converts it to a bounded simulation for the same certified graph over the transported source. Apply Definition 199.13 to the two accepted

rows after classifier transport. The conclusion is $m\mathsf{hsame}m'$, which is exactly stability of bounded simulation output classification rather than host equality of histories.

The same argument controls composition. When the transported row is used as a component of a composite run, Definition 199.11 composes the transported bounded trace with the next certified trace, and the output class remains governed by the same single-valuedness theorem. □

Theorem 199.9 (Computable bounded graph ledger exactness)**.** *For every bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate* (P_F, β_F) *for* $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$*, the ledger records the deterministic code* P_F*, the displayed bound modulus* β_F*, the* $\mathsf{Seq}^{\uparrow}$ *simulation trace, and the endpoint readback. Its exactness clause is the displayed equivalence*

$$F(n, m) \iff \mathsf{BoundedSim}(P_F, n, \beta_F(n), m)$$

at that bound, with continuation data recorded by Cont *whenever two certified traces are composed.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraph ledger exactness obligation`

Proof. The deterministic code and modulus are the two explicit fields of Definition 199.2. For each accepted graph row, the forward direction of the displayed equivalence gives the $\mathsf{Seq}^{\uparrow}$ trace and endpoint readback required by Definition 199.1.

Conversely, a bounded simulation at the displayed value $\beta_F(n)$ is converted back to graph membership by the reverse direction of the same equivalence. Thus ledger exactness is an iff between graph membership and bounded simulation at the displayed certificate bound, not an existential statement over hidden running times.

For composed certificates, Definition 4.10 supplies the continuation witness joining the two displayed bounds, and Definition 199.11 builds the composite trace. The endpoint readback of the composite ledger is the endpoint readback of the second trace, while single-valuedness is supplied by Definition 199.13. □

Theorem 199.10 (Computable bounded graph certificate scope)**.** *For every bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate* (P_F, β_F) *for a partial graph* $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$*, the primitive scope of the four certificate obligations is exactly the bounded graph certificate surface. The carrier obligation is scoped by the unary input history* n*, the unary output history* m*, the displayed unary bound* $\beta_F(n)$*, and the* $\mathsf{Seq}^{\uparrow}$ *simulation trace. The classifier obligation is scoped by* hsame *and the* $\mathsf{Nat}^{\uparrow}$ *unary classifier. The stability obligation is scoped by the identity certificate,* Cont *phase-boundary composition, and bounded-simulation single-valuedness. The ledger obligation is scoped by the program code* P_F*, the displayed bound value* $\beta_F(n)$*, the trace, the endpoint readback, and the continuation witness used for composed traces.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate scope`

Proof. Unpack the bounded graph certificate of Definition 199.2. Its fields are the deterministic code P_F, the unary-bound modulus β_F, the graph exactness equivalence with bounded simulation, and output determinacy. The witness format of Definition 199.1 then names the only carrier data used by an accepted row: unary source input, displayed bound, $\mathsf{Seq}^{\uparrow}$ trace, and unary endpoint readback.

The carrier and ledger clauses are therefore the obligations isolated in Definition 199.7 and Definition 199.9. They expose no hidden running-time object beyond $\beta_F(n)$ and no hidden observation object beyond the $\mathsf{Seq}^{\uparrow}$ trace and endpoint readback.

The classifier clause is Definition 199.8: transport is governed by hsame and the unary classifier of $\mathsf{Nat}^{\uparrow}$, and the accepted output class is controlled by Definition 199.13.

Finally, stability is supplied by Definition 199.17 for the identity surface and by Definition 199.11 for phase-boundary composition. Thus Cont records the continuation witness joining displayed bounds, while single-valuedness keeps each certified graph row functional up to the unary classifier. These ingredients are precisely the bounded $\mathsf{Computable}^{\uparrow}$ graph certificate scope. □

Theorem 199.11 (Computable composition by bounded simulation)**.** *Let F and G be partial $\mathsf{Nat}^{\uparrow}$ graphs carrying bounded $\mathsf{Computable}^{\uparrow}$ graph certificates (P_F, β_F) and (P_G, β_G) in the sense of Definition 199.2. Suppose $F(n, m)$ is witnessed by*

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m)$$

and $G(m, k)$ is witnessed by

$$\mathsf{BoundedSim}(P_G, m, \beta_G(m), k).$$

Then there is a unary history B and a deterministic composite machine code $P_{G\circ F}$ such that

$$\mathsf{BoundedSim}(P_{G\circ F}, n, B, k).$$

Consequently the composite graph

$$(G \circ F)(n, k) := \exists m : \mathsf{Nat}^{\uparrow},\ F(n, m) \text{ and } G(m, k)$$

has a certified bounded simulation returning k on input n.

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim composition`

Proof. Write $b_F := \beta_F(n)$ and $b_G := \beta_G(m)$. Since the two certificates are $\mathsf{Seq}^{\uparrow}$ unary-bound moduli over the $\mathsf{Nat}^{\uparrow}$ source, b_F and b_G are unary histories. Apply the continuation rules of Definition 4.10 to choose a history B with $\mathsf{Cont}(b_F, b_G, B)$. By unary closure under continuation from Definition 10.2, B is unary.

Define $P_{G\circ F}$ as the phase machine that first runs P_F on the input tape, reads the halted unary output as the next input tape, and then runs P_G. The phase boundary is exact: the endpoint row of the first bounded trace is halted with output m, which is precisely the initial input required by the second bounded trace.

Build a $\mathsf{Seq}^{\uparrow}$ trace over the unary prefixes of B by prefix recursion along the displayed continuation witness $\mathsf{Cont}(b_F, b_G, B)$. On the first segment, indexed by prefixes of b_F, use the trace supplied by $\mathsf{BoundedSim}(P_F, n, b_F, m)$. At the boundary b_F, the first trace is already halted with output m by Definition 199.1. On the second segment, indexed as the continuation of b_F by prefixes of b_G, use the trace supplied by $\mathsf{BoundedSim}(P_G, m, b_G, k)$.

The transition rows are valid in all cases. Inside the first segment they are the transition rows of P_F; inside the second segment they are the transition rows of P_G; at the boundary they are the phase switch of $P_{G\circ F}$. Halted-state repetition in Definition 199.1 permits the first bounded run to occupy its whole bound before the second phase begins. The endpoint of the composite bound corresponds to the endpoint of the G trace, so it is halted with output k. Hence the assembled trace is a witness of $\mathsf{BoundedSim}(P_{G\circ F}, n, B, k)$. □

Theorem 199.12 (Computable bounded graph certificate is functional)**.** *Let $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ be a partial graph carrying a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate (P_F, β_F) in the sense of Definition 199.2. For unary histories n, m, and m', if $F(n, m)$ and $F(n, m')$ both hold, then m and m' are classified as the same $\mathsf{Nat}^{\uparrow}$ output:*

$$m \mathsf{hsame} m'.$$

Proof. Apply the graph-equivalence clause of the bounded certificate to the first graph membership. It converts $F(n, m)$ into a bounded simulation witness

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m).$$

Apply the same clause to the second graph membership, obtaining

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m').$$

These are bounded simulations for the same deterministic machine code, the same input history, and the same displayed bound, because the bound is the certificate modulus value $\beta_F(n)$. The output-determinacy clause of Definition 199.2 therefore compares their endpoint readbacks and yields $m\mathsf{hsame}m'$. By the unary classifier specification of Definition 28.27, this is exactly sameness of the two $\mathsf{Nat}^{\uparrow}$ output classes. □

Theorem 199.13 (Computable bounded graph single-valuedness)**.** *Let $F \subseteq \mathsf{Nat}^{\uparrow} \times \mathsf{Nat}^{\uparrow}$ carry a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate (P_F, β_F) in the sense of Definition 199.2. For unary histories n, m, and m', if $F(n, m)$ and $F(n, m')$ both hold, then the two accepted rows expand to bounded simulations at the same displayed certificate bound,*

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m) \quad \textit{and} \quad \mathsf{BoundedSim}(P_F, n, \beta_F(n), m'),$$

and their output histories are classified together:

$$m\mathsf{hsame}m'.$$

Thus a bounded $\mathsf{Computable}^{\uparrow}$ graph certificate makes the accepted output over each unary input single-valued up to the $\mathsf{Nat}^{\uparrow}$ history classifier.

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim same bound output deterministic`

Proof. Unpack the bounded graph certificate of Definition 199.2. Its graph-equivalence field states, for every unary input and unary output, that graph membership is equivalent to bounded simulation by the certified machine at the certified bound:

$$F(n, x) \iff \mathsf{BoundedSim}(P_F, n, \beta_F(n), x).$$

Apply this field with $x = m$ to the accepted row $F(n, m)$, obtaining

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m).$$

Apply the same field with $x = m'$ to the accepted row $F(n, m')$, obtaining

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m').$$

The two simulations now have the same deterministic code P_F, the same unary input n, and the same displayed ledger bound $\beta_F(n)$. The output-determinacy field of the same bounded graph certificate says that any two bounded simulations for one input return hsame-classified output histories. Applying that field to the two displayed simulation witnesses gives $m\mathsf{hsame}m'$. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate single valuedness`

Definition 199.14 (Unary classifier identity graph)**.** *The identity graph on* $\mathsf{Nat}^{\uparrow}$ *histories is the partial graph*

$$\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m) :\Longleftrightarrow \mathsf{UnaryHistory}(n) \wedge \mathsf{UnaryHistory}(m) \wedge n\mathsf{hsame}m.$$

Its equality content is the unary classifier of Definition 28.27, not host equality of history terms.

Lean4 (definition) `BEDC.Derived.ComputableUp.ComputableUnaryIdentityGraph`

Definition 199.15 (Halted unary identity code with empty bound)**.** *Let* P_{id} *be the one-state deterministic machine code whose unique state is halted. On unary input* n*, its initial configuration already carries* n *on the output tape. Its halted readout is the* $\mathsf{Nat}^{\uparrow}$ *classifier readout: a unary output history* m *is read at that row exactly when* $n\mathsf{hsame}m$*. The halted-state repeat row is the only transition row. Define the displayed bound modulus by*

$$\beta_{\mathsf{id}}(n) := \mathsf{e}.$$

This is the constant-empty $\mathsf{Seq}^{\uparrow}$ *unary-bound modulus, since* e *is unary by Definition 28.1.*

Lean4 (definition) `BEDC.Derived.ComputableUp.ComputableHaltedUnaryIdentityCode`

Lemma 199.16 (Identity empty-bound simulation exactness)**.** *For unary histories* n *and* m*,*

$$\mathsf{BoundedSim}(P_{\mathsf{id}}, n, \beta_{\mathsf{id}}(n), m) \Longleftrightarrow \mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m).$$

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableUnaryIdentityGraph empty bound exactness`
↪ variant `BEDC.Derived.ComputableUp.ComputableUnaryIdentityGraph empty bound simulation exactness`

Proof. Assume first $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m)$. By Definition 199.14, the histories n and m are unary and $n\mathsf{hsame}m$. Use the one-entry $\mathsf{Seq}^{\uparrow}$ trace indexed by the empty bound $\beta_{\mathsf{id}}(n) = \mathsf{e}$. Its zero-prefix entry is the initial configuration of P_{id} on n, which is already halted by Definition 199.15. There are no nontrivial one-step extensions in the empty prefix interval. The endpoint is the same halted row, and its classifier readout accepts m because $n\mathsf{hsame}m$. Hence the trace satisfies Definition 199.1.

Conversely, assume $\mathsf{BoundedSim}(P_{\mathsf{id}}, n, \beta_{\mathsf{id}}(n), m)$. Since the displayed bound is e, the endpoint row is the zero-prefix row. By Definition 199.1, that row is the initial configuration of P_{id} on n and is halted with readout m. The readout clause of Definition 199.15 gives $n\mathsf{hsame}m$, and the simulation witness supplies the unary input and unary output histories. Therefore $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m)$ by Definition 199.14. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableUnaryIdentityGraph empty bound simulation iff`

Theorem 199.17 (Computable identity bounded graph certificate)**.** *The pair* $(P_{\mathsf{id}}, \beta_{\mathsf{id}})$ *is a bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate for* $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$*. In particular, for all unary histories* n *and* m*,*

$$\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m) \Longleftrightarrow \mathsf{BoundedSim}(P_{\mathsf{id}}, n, \beta_{\mathsf{id}}(n), m),$$

and any two bounded simulations for the same unary input return hsame*-classified output histories.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableUnaryIdentityGraph empty bound certificate`

Proof. The program field is the deterministic halted code P_{id} from Definition 199.15. The bound field is the constant-empty $\mathsf{Seq}^{\uparrow}$ unary-bound modulus β_{id}; its value is part of the certificate ledger and is unary for every unary input.

The graph-equivalence field is exactly Definition 199.16, read in both directions. For output determinacy, let

$$\mathsf{BoundedSim}(P_{\mathsf{id}}, n, \beta_{\mathsf{id}}(n), m) \quad \text{and} \quad \mathsf{BoundedSim}(P_{\mathsf{id}}, n, \beta_{\mathsf{id}}(n), m')$$

be two bounded simulations for the same unary input. Applying the reverse direction of Definition 199.16 gives $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m)$ and $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}(n, m')$. Unfolding Definition 199.14 gives $n\mathsf{hsame}m$ and $n\mathsf{hsame}m'$. Symmetry and transitivity of history-sameness from Definition 3.8 yield $m\mathsf{hsame}m'$. These are precisely the fields required by Definition 199.2. □

Theorem 199.18 (Computable bounded graph identity and composition interface)**.** *Bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificates provide an identity and composition interface for partial* $\mathsf{Nat}^{\uparrow}$ *graphs. The identity object is the classifier identity graph* $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$ *of Definition 199.14, certified by Definition 199.17. If* F *and* G *carry bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificates and accepted rows* $F(n, m)$ *and* $G(m, k)$ *are displayed by their certified bounded simulations, then Definition 199.11 supplies a unary bound* B *and a deterministic composite code* $P_{G\circ F}$ *with*

$$\mathsf{BoundedSim}(P_{G\circ F}, n, B, k).$$

Moreover, Definition 199.13 ensures that each certified component graph, and hence the composite graph assembled through its certified middle row, remains single-valued up to the $\mathsf{Nat}^{\uparrow}$ *history classifier rather than becoming a multivalued relation.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim identity composition interface`

Proof. The identity row is already a bounded graph certificate: Definition 199.17 packages the halted identity code P_{id}, the empty bound modulus β_{id}, the exactness equivalence with $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$, and output determinacy. Thus $\mathsf{Id}_{\mathsf{Nat}^{\uparrow}}$ is the neutral graph available at the certificate level, with equality interpreted through $n\mathsf{hsame}m$ by Definition 199.14.

For composition, start from certified rows

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m) \quad \text{and} \quad \mathsf{BoundedSim}(P_G, m, \beta_G(m), k).$$

Apply Definition 199.11. Its construction chooses a continuation witness $\mathsf{Cont}(\beta_F(n), \beta_G(m), B)$ for the displayed bounds, concatenates the two $\mathsf{Seq}^{\uparrow}$ traces along that witness, and runs the phase code $P_{G\circ F}$ first through P_F and then through P_G. The endpoint of the concatenated bound is the endpoint of the second trace, so it reads back k.

It remains to record that the interface is still a partial graph interface. If a certified component accepts two outputs over the same unary input, Definition 199.13 expands both accepted rows to simulations at the same ledger bound and applies output determinacy, yielding hsame-classified outputs. Therefore the middle row used by the composite is determined up to the $\mathsf{Nat}^{\uparrow}$ classifier, and the resulting accepted output row is controlled by the same single-valuedness clause instead of by an unconstrained relation. □

Theorem 199.19 (Computable bounded graph composition associativity)**.** *Let* F*,* G*, and* H *be partial* $\mathsf{Nat}^{\uparrow}$ *graphs carrying bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificates* (P_F, β_F)*,* (P_G, β_G)*, and*

(P_H, β_H). *Suppose accepted rows* $F(n,m)$, $G(m,k)$, *and* $H(k,l)$ *are displayed by bounded simulations*

$$\mathsf{BoundedSim}(P_F, n, \beta_F(n), m), \quad \mathsf{BoundedSim}(P_G, m, \beta_G(m), k), \quad \mathsf{BoundedSim}(P_H, k, \beta_H(k), l).$$

Then the two iterated composite rows $((H \circ G) \circ F)(n,l)$ *and* $(H \circ (G \circ F))(n,l)$ *are represented by bounded simulations whose phase order is* P_F *then* P_G *then* P_H. *Their displayed bounds are compared by the associativity row of* Cont, *and their endpoint output histories are classified by* hsame.

Proof. For the left bracketing, first apply Definition 199.11 to the displayed F and G simulations. This gives a composite code $P_{G \circ F}$, a unary bound B_{GF}, and a continuation witness $\mathsf{Cont}(\beta_F(n), \beta_G(m), B_{GF})$. Apply Definition 199.11 again to the resulting $G \circ F$ simulation and the displayed H simulation, obtaining a code for $(H \circ G) \circ F$ and a bound B_L with $\mathsf{Cont}(B_{GF}, \beta_H(k), B_L)$.

For the right bracketing, first apply Definition 199.11 to the displayed G and H simulations. This gives a composite code $P_{H \circ G}$, a unary bound B_{HG}, and a continuation witness $\mathsf{Cont}(\beta_G(m), \beta_H(k), B_{HG})$. Apply Definition 199.11 again to the displayed F simulation and the resulting $H \circ G$ simulation, obtaining a code for $H \circ (G \circ F)$ and a bound B_R with $\mathsf{Cont}(\beta_F(n), B_{HG}, B_R)$.

Both constructions are built from the same concrete $BHist$ traces. The $\mathsf{Seq}^{\uparrow}$ carrier clauses in Definition 199.1 supply the first trace segment for P_F, the second segment for P_G, and the third segment for P_H; only the parenthesizing of the continuation witnesses changes.

The bound comparison is exactly the associativity clause for the continuation relation of Definition 4.10, together with its unit rows for the empty-bound edge cases. Thus the left and right displayed bounds present the same three-phase continuation data, with phase order P_F, then P_G, then P_H.

Finally, the endpoint row of either iterated simulation is the endpoint row of the H trace. If two representatives for the same iterated composite row are produced through different middle histories, Definition 199.13 first classifies the G endpoints over m and then the H endpoints over k, yielding hsame-classified final output histories. This is a three-certificate composition law over concrete $BHist$ traces, not transport between two abstract source presentations. □

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedSim composition associativity`

Theorem 199.20 (Computable bounded graph scoped certificate)**.** *A bounded* $\mathsf{Computable}^{\uparrow}$ *graph certificate over a partial* $\mathsf{Nat}^{\uparrow}$ *graph is scoped exactly by the BEDC primitives* BHist, UnaryHistory, hsame, *the* $\mathsf{Seq}^{\uparrow}$ *bounded-simulation trace,* ComputableBoundedSim, *the displayed bound modulus, and* Cont *phase composition. For every certificate* (P_F, β_F) *and every accepted row* $F(n,m)$, *the scoped certificate packages the following four rows at the displayed bound* $\beta_F(n)$*:*

1. *the carrier and source row of Definition 199.7:* $\mathsf{UnaryHistory}(n)$, $\mathsf{UnaryHistory}(\beta_F(n))$, $\mathsf{UnaryHistory}(m)$, *and the* $\mathsf{Seq}^{\uparrow}$ *trace used by* $\mathsf{BoundedSim}(P_F, n, \beta_F(n), m)$*;*
2. *the classifier row of Definition 199.8: source transport through* hsame *preserves accepted output classes for the certified bounded simulation;*
3. *the stability row of Definition 199.18: the empty-bound identity certificate and* Cont*-mediated phase composition provide the identity and composition interface for certified partial graphs;*
4. *the ledger and exactness row of Definition 199.9: graph membership is equivalent to* $\mathsf{BoundedSim}(P_F, n, \beta_F(n), m$ *at the displayed certificate bound.*

Lean4 ✓ `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate scoped certificate`

Proof. For the carrier row, apply Definition 199.7. It exposes the unary source, displayed bound, output history, and $\mathsf{Seq}^{\uparrow}$ configuration trace from the bounded simulation witness, all at the certificate value $\beta_F(n)$. These are the local BHist objects seen by the certificate surface.

For the classifier row, use Definition 199.8. It says that transport through the $\mathsf{Nat}^{\uparrow}$ history classifier preserves the accepted bounded-simulation readback class, and Definition 199.13 supplies the same hsame control when two accepted rows share the certified source.

For the stability row, cite Definition 199.18. The neutral graph is the empty-bound identity certificate, while the composite row is built by concatenating two $\mathsf{Seq}^{\uparrow}$ traces along a Cont phase witness and returning a bounded simulation for the composite output.

For the ledger row, apply Definition 199.9. Its two directions identify the accepted graph row with $\mathsf{BoundedSim}(P_F, n, \beta_F(n), m)$ at the displayed bound, so the program code, modulus, trace, endpoint readback, and composition boundary are all recorded inside the scoped certificate rather than left as separate obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$.

Constructive story: A computable partial graph is carried by a deterministic machine code together with a displayed unary-bound modulus: for an accepted row $F(n, m)$, the input n, bound $\beta_F(n)$, and output m are unary histories, and a $\mathsf{Seq}^{\uparrow}$ trace of configurations runs along the unary prefixes of that bound. The carrier accepts exactly the bounded simulation package whose zero row is the initial configuration, whose one-step rows follow the transition relation with halted repetitions allowed, and whose endpoint is halted with unary readback m; hidden exact running times, universal machines, and external recursion-theoretic objects are not carrier fields. The classifier uses the $\mathsf{Nat}^{\uparrow}$ history classifier and hsame: accepted outputs over the same certified input and displayed bound are single-valued up to history-sameness, while the identity graph is the empty-bound halted code whose graph predicate is just unary input, unary output, and $n\mathsf{hsame}m$. The source covers unary inputs, unary bounds, unary outputs, deterministic program code, and $\mathsf{Seq}^{\uparrow}$ traces; the pattern records graph exactness, single-valuedness, identity, bounded-simulation composition, composition associativity, and classifier stability; the ledger keeps the code, modulus value, trace, endpoint readback, and Cont phase boundary visible. The five name-certificate obligations are supplied by the inhabited identity certificate, by reflexive, symmetric, and transitive hsame control on unary histories, and by transport and composition rows that repack bounded simulations after classifier movement. A history h lies in the signature gap of a $\mathsf{Computable}^{\uparrow}$ token p when the source-domain accepts the bounded graph row, the signature relation pairs it with an answer row and simulation ledger, and token introduction names that answer as p; unfolded, h is the graph value produced by the deterministic code at its displayed bound, with output matching the unary value encoded by p. The chapter registers this bounded-simulation construction under the name $\mathsf{Computable}^{\uparrow}$; computability here is an explicit time-bounded trace certificate, not a totality claim, halting decision procedure, recursion-theory library, or standard bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridged scope treats the bounded graph public interface of Definition 199.2 and the single-valuedness theorem Definition 199.13 as the bridge boundary, exposing identity, composition, associativity, and the scoped certificate rows of Definition 199.20.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ComputableUp.ComputableBoundedGraphCertificate\_single\_valuedness`

Bridge status: none

Not claimed: The chapter does not close total computability, universal machines, halting decidability, recursion theory, or standard bridge claims.

Upgrade path: The bounded-graph public interface is the mature theory boundary; the remaining verification work is a checked standard bridge for that boundary.

Chapter 200

A Concrete Naming Certificate for $\mathsf{LambdaCalc}^{\uparrow}$

The lambda-calculus interface $\mathsf{LambdaCalc}^{\uparrow}$ packages the inductive carrier of λ-terms together with β-reduction, α-equivalence, and a stability classifier under capture-avoiding substitution. It depends on $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$ (chapter 119) and $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28).

200.1 Carrier definition

Definition 200.1 (LambdaCalc BHist term carrier)**.** *The LambdaCalc BHist term carrier* LambdaCalcBHistTermCa *is the* $\mathsf{Tree}^{\uparrow}$*-backed syntax row whose visible spine is a* BHist *tree with three constructor tags. A variable row has tag* lamVar *and a* $\mathsf{Nat}^{\uparrow}$ *unary variable index* i *with* $\mathsf{UnaryHistory}(i)$*. An abstraction row has tag* lamAbs*, a unary binder index, and one body subtree. An application row has tag* lamApp *and two subtrees. In each case the spine is carried by a* TreeBHistCarrier *row, constructor readback is expressed by* hsame *between the visible tag-and-subtree endpoint and the displayed* BHist *syntax row, and the classifier compares carried terms only through the corresponding* Cont *witness on these visible spines.*

Theorem 200.2 (LambdaCalc variable constructor carrier)**.** *For histories* $i, h : \mathsf{Hist}$*, if* $\mathsf{UnaryHistory}(i)$*, the visible variable constructor row for* h *is a* TreeBHistCarrier *row, and the displayed constructor endpoint satisfies*

$$\mathsf{hsame}(h, \mathsf{lamVar}(i)),$$

then $\mathsf{LambdaCalcBHistTermCarrier}(h)$*. Moreover the carried row keeps the* $\mathsf{Nat}^{\uparrow}$ *index visible: any* Cont *witness transporting* h *to* h' *along a constructor-preserving* $\mathsf{hsame}(h, h')$ *transports the same unary index row and yields* $\mathsf{LambdaCalcBHistTermCarrier}(h')$*.*

Proof. Read the variable row through the visible spine supplied by TreeBHistCarrier. The tag readback identifies the root as lamVar, and the sole payload is the unary history i.

NatUp unary index closure keeps i in the unary fragment under the displayed hsame transport. TreeUp visible-spine transport then carries the tagged endpoint from h to h', so the transported row is again a LambdaCalc carrier row of the variable form. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermCarrier variable constructor carrier`

Theorem 200.3 (LambdaCalc abstraction and application carrier closure)**.** *For histories x, b, f, g, h : Hist, suppose the displayed syntax spines for b, f, and g are $\mathsf{TreeBHistCarrier}$ rows and*

$$\mathsf{LambdaCalcBHistTermCarrier}(b), \quad \mathsf{LambdaCalcBHistTermCarrier}(f), \quad \mathsf{LambdaCalcBHistTermCarrier}(g).$$

If $\mathsf{UnaryHistory}(x)$ *and*

$$\mathsf{hsame}(h, \mathsf{lamAbs}(x, b)),$$

then $\mathsf{LambdaCalcBHistTermCarrier}(h)$*. If instead*

$$\mathsf{hsame}(h, \mathsf{lamApp}(f, g)),$$

then $\mathsf{LambdaCalcBHistTermCarrier}(h)$*. Both conclusions are stable under any constructor-preserving* Cont *transport of the displayed* BHist *spine.*

Proof. For abstraction, use the TreeUp visible-spine row to expose the lamAbs root, the unary binder payload, and the single body subtree. The body is already a LambdaCalc carrier row, and NatUp unary index closure keeps the binder inside the admitted index surface.

For application, use the same TreeUp spine readback with root lamApp and the two carried subtrees f and g. No host syntax operation is used: the two arguments are exactly the displayed BHist subtrees, and the carrier conclusion repacks them through the LambdaCalc definition.

Constructor-preserving Cont transport acts componentwise on the visible spine. It preserves the root tag and transports each carried subtree by hsame, so both constructor clauses remain inside $\mathsf{LambdaCalcBHistTermCarrier}$. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermCarrier abstraction application closure`

Theorem 200.4 (LambdaCalc substitution ledger scope)**.** *Let x, s, t, u : Hist be histories with* $\mathsf{UnaryHistory}(x)$*,* $\mathsf{LambdaCalcBHistTermCarrier}(s)$*, and* $\mathsf{LambdaCalcBHistTermCarrier}(t)$*. A LambdaCalc substitution ledger row for substituting s for the unary variable x in t is in scope only when it is presented by a* BHist *visible-spine traversal whose nodes are carried by* $\mathsf{TreeBHistCarrier}$*, whose recursive calls are connected by* Cont *witnesses, and whose output endpoint u satisfies* $\mathsf{LambdaCalcBHistTermCarrier}(u)$ *together with constructorwise* hsame *readback. The row exports no host substitution function and no unmentioned capture-avoidance primitive.*

Proof. Traverse the displayed TreeUp spine of t. In the variable case, compare the visible NatUp unary index with x and choose either the carried substituent s or the original variable row, with the choice recorded as a BEDC ledger branch rather than a host-language function.

In the abstraction and application cases, use the constructor closure theorem above after recursively transporting the displayed child rows. The Cont witnesses record the recursive route through the visible BHist spine, and every endpoint comparison is an hsame comparison between displayed constructor rows.

Thus the substitution ledger remains a scoped LambdaCalc carrier obligation: it names only BHist, $\mathsf{UnaryHistory}$, $\mathsf{TreeBHistCarrier}$, Cont, hsame, and $\mathsf{LambdaCalcBHistTermCarrier}$ rows. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier substitution ledger scope`

Definition 200.5 (LambdaCalcUp term packet carrier)**.** *A* $\mathsf{LambdaCalc}^{\uparrow}$ *term packet carrier is a finite* BHist *packet*

$$\mathcal{L} = (T, I, a, b, s, \lambda, \rho)$$

over the source rows $\mathsf{NameCert}_{\mathsf{Tree}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$*. The row* T *is the displayed* $\mathsf{Tree}^\uparrow$ *syntax row for variables, abstraction nodes, and application nodes. The row* I *is the displayed* $\mathsf{Nat}^\uparrow$ *binder-index row used to name de Bruijn levels, binder depth, and free-variable positions. The row* a *is the alpha-equivalence ledger comparing syntax only through the carried tree and index rows. The row* b *is the beta-redex ledger naming application-abstraction pairs together with their substitution endpoint. The row* s *is the capture-avoidance* Cont *ledger recording renaming, binder shift, and substitution continuations. The row* λ *orders the finite term-construction and reduction steps, and* ρ *is the* Pkg *provenance row for the displayed syntax, index, alpha-equivalence, beta-redex, capture-avoidance, and continuation entries.*

200.2 The certificate

Definition 200.6 (LambdaCalc free-variable ledger packet)**.** *A LambdaCalc free-variable ledger packet is a finite* BHist *packet*

$$\mathcal{F} = (T, I, F, A, B, S, \lambda, \rho)$$

over $\mathsf{NameCert}_{\mathsf{Tree}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$*. The row* T *is the TreeUp syntax spine for variables, abstraction nodes, and application nodes. The row* I *is the NatUp binder-index ledger naming de Bruijn levels and binder depth. The row* F *records the free-variable endpoint rows exposed by the syntax traversal. The row* A *records alpha-renaming comparisons, and* B *records beta-redex source and endpoint rows. The row* S *is the finite* Cont *substitution provenance for renaming, binder shift, and capture-avoiding replacement. The row* λ *orders the displayed traversal, and* ρ *is the* Pkg *provenance tying the TreeUp syntax, NatUp indices, free-variable rows, alpha rows, beta-redex rows, and substitution ledger into a single LambdaCalc certificate packet.*

Theorem 200.7 (LambdaCalc free-variable ledger coverage)**.** *Every accepted free-variable endpoint of a* $\mathsf{LambdaCalc}^\uparrow$ *term is projected from a packet of Definition 200.6. More precisely, if* $v : \mathsf{Hist}$ *is accepted as a free variable of a carried* LambdaCalcBHistTermCarrier *term* t*, then there are rows* $T, I, F, A, B, S, \lambda, \rho$ *forming* $\mathcal{F}$ *such that* t *occurs in the TreeUp spine* T*,* v *occurs in the NatUp index ledger* I*, the endpoint row is listed in* F*, and the projection is justified only by the displayed* BHist*,* Cont*,* hsame*,* NameCert*, and* Pkg *rows. No host variable set is exported.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier free variable ledger coverage`

Proof. Induct on the displayed TreeUp spine of the carried LambdaCalc term. In the variable case, the NatUp unary index row is the exposed free-variable endpoint, and the packet records it in F with the same TreeUp and NatUp provenance rows.

For an abstraction node, read the binder index from I and compare it with the free-variable row by the carried hsame ledger. The binder branch removes only the displayed endpoint row from the packet; it does not form a host set of variables.

For an application node, use the two recursive packet projections and concatenate their finite Cont provenance through λ. Repacking the two child rows with the common Pkg provenance gives exactly the accepted endpoint rows in F, so every accepted free-variable endpoint is covered by the finite packet. □

Theorem 200.8 (LambdaCalc capture-avoidance Cont closure)**.** *Let* $x, s, t, u : \mathsf{Hist}$ *be histories with* $\mathsf{UnaryHistory}(x)$*,* $\mathsf{LambdaCalcBHistTermCarrier}(s)$*, and* $\mathsf{LambdaCalcBHistTermCarrier}(t)$*. If a packet of Definition 200.6 displays a capture-avoiding substitution* Cont *ledger from* t *with substituent* s *at the unary variable* x *to endpoint* u*, then* $\mathsf{LambdaCalcBHistTermCarrier}(u)$*. The closure statement is stable under the same TreeUp and NatUp* hsame *rows recorded by the packet.*

Proof. Induct on the TreeUp spine of t. The variable branch compares the visible NatUp index with x inside the finite ledger and chooses either the carried substituent s or the original variable constructor row.

For abstraction, use the binder row in I and the alpha ledger A to rename the binder when the displayed free-variable row would otherwise be captured. The substitution provenance remains a Cont row, and the body endpoint is obtained by the induction hypothesis.

For application, run the induction hypothesis on the two displayed child spines and combine their endpoints with the application constructor closure of Definition 200.3. The resulting endpoint u is therefore a LambdaCalc BHist term carrier, transported only through the listed hsame and Cont rows. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier capture avoidance cont closure`

Theorem 200.9 (LambdaCalc alpha-beta carrier transport)**.** *Suppose two beta-redex packets in Definition 200.6 have TreeUp syntax rows and NatUp binder-index rows related by the alpha ledger A. Then the source application, abstraction body, argument, substitution endpoint, and beta-contractum endpoints transport as* LambdaCalcBHistTermCarrier *rows along the displayed TreeUp and NatUp* hsame *comparisons. Thus alpha-related beta-redex packets determine equivalent carrier endpoints without exporting quotient alpha classes.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha beta carrier transport`

Proof. Read the two beta-redex packets through the TreeUp application spine. The alpha ledger supplies constructor-preserving hsame rows for the application root, the abstraction child, the binder index, and the argument subtree.

Transport the capture-avoidance ledger S along these rows. The preceding closure theorem keeps the substitution endpoint inside LambdaCalcBHistTermCarrier after binder renaming and NatUp index transport.

Finally repack the transported source and endpoint rows as the beta-redex carrier data. Since every comparison is a displayed TreeUp or NatUp hsame row and every route is recorded in Cont, the transported endpoints are carrier-equivalent in the LambdaCalc certificate surface. □

Theorem 200.10 (LambdaCalc carrier reflexivity)**.** *Every accepted* $\mathsf{LambdaCalc}^{\uparrow}$ *term packet over a* $\mathsf{Tree}^{\uparrow}$ *syntax row and a* $\mathsf{Nat}^{\uparrow}$ *binder ledger classifies with itself. The reflexive classifier is the componentwise* hsame *row on the syntax tree, binder indices, free-variable ledger, and package provenance.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier carrier reflexive`

Proof. 1. Traditional target: reflexivity of the lambda-term equality classifier. BEDC source: an accepted LambdaCalcBHistTermCarrier packet whose visible syntax row is supplied by $\mathsf{Tree}^{\uparrow}$ and whose binder indices are supplied by $\mathsf{Nat}^{\uparrow}$.

2. Carrier: keep the same BHist term packet and the same Pkg provenance row. Classifier: use reflexive hsame on the visible TreeUp skeleton, each unary binder index, the free-variable ledger, and the package provenance.

3. Stability and ledger: the identity Cont ledger transports no constructor payload and no binder row. The carried rows are therefore read back through the same visible spine.

4. Core theorem and exactness: the accepted packet classifies with itself, and no endpoint outside the displayed syntax, binder, free-variable, and provenance rows is read. Standard bridge: absent. Status: paper obligation for the future BEDC namespace Derived.LambdaCalcUp.

□

Theorem 200.11 (LambdaCalc alpha classifier symmetry)**.** *If an accepted alpha-classifier row relates two accepted* LambdaCalc$^{\uparrow}$ *term packets, then the reverse alpha-classifier row is accepted. The reversed row is obtained by reversing the carried* hsame *rows on the* Tree$^{\uparrow}$ *skeletons and the* Nat$^{\uparrow}$ *binder indices.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha classifier symmetric`

Proof. 1. Traditional target: symmetry of alpha equivalence. BEDC source: an accepted alpha-classifier row between two BHist lambda-term packets, with TreeUp skeleton certificates and NatUp binder-index certificates exposed as visible rows.

2. Carrier: both endpoints remain the original accepted LambdaCalc packets. Classifier: reverse each carried hsame comparison on the TreeUp skeleton, the unary binder indices, and the provenance rows that identify the two packets as LambdaCalc terms.

3. Stability and ledger: the alpha row already carries a constructorwise ledger for the binder-renaming route. Reading that ledger in reverse preserves the same constructor tags and consumes the same NatUp unary index rows.

4. Core theorem and exactness: the reversed row is an accepted alpha classifier and exports no quotient of alpha classes. Standard bridge: absent. Status: paper obligation for the future BEDC namespace Derived.LambdaCalcUp.

□

Theorem 200.12 (LambdaCalcUp substitution ledger obligation)**.** *Substitution in* LambdaCalc$^{\uparrow}$ *is admitted only from a carried Definition 200.5: the* Tree$^{\uparrow}$ *syntax row names the term spine, the* Nat$^{\uparrow}$ *binder-index row names the binder depth and free positions, and the capture-avoidance* Cont *ledger supplies the renaming, shift, and substitution continuations. No host syntax equality, ambient variable set, or quotient alpha-class is part of the substitution admission surface.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier substitution ledger obligation`

Proof. Project the syntax row T and binder-index row I from Definition 200.5. These rows give the only term spine and binder positions that the substitution endpoint may read.

Project the capture-avoidance ledger s and the finite continuation ledger λ. Their entries record the renaming and binder-shift steps needed to thread substitution through abstraction and application nodes.

Repack the substitution endpoint using exactly these projected rows and the Pkg provenance row ρ. Because the carrier contains no row for host syntax equality, ambient variable sets, or quotient alpha-classes, those data cannot be exported by the substitution obligation. □

Theorem 200.13 (LambdaCalcUp beta-reduction classifier obligation)**.** *Beta reduction in* LambdaCalc$^{\uparrow}$ *is classifier-stable under* hsame *transport of the carried* Tree$^{\uparrow}$ *syntax row, the carried* Nat$^{\uparrow}$ *binder-index row, and the substitution ledger of Definition 200.12. A transported beta-redex row therefore repacks as a beta-reduction row with the transported syntax, binder index, capture-avoidance ledger, and* Pkg *provenance; it does not identify terms by host syntax equality or by quotient alpha-classes.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier beta reduction classifier obligation`

Proof. Start from the beta-redex ledger b in Definition 200.5. Its source application, abstraction body, argument, and endpoint are named through the carried syntax row and binder-index row.

Transport the syntax and binder-index rows by hsame, then transport the substitution ledger supplied by Definition 200.12. The transported ledger still records renaming, shift, and capture-avoidance continuations inside the displayed BHist packet.

Repack the transported redex, substitution endpoint, continuation ledger, and provenance as the beta-reduction classifier row. Since each field came from a displayed carrier component or its hsame transport, no host equality of terms and no quotient alpha-class enters the classifier. □

Theorem 200.14 (LambdaCalcUp constructor case exhaustion)**.** *Every accepted* LambdaCalcBHistTermCarrier *row in a displayed* Hist *endpoint is witnessed by exactly one visible* TreeBHistCarrier *constructor tag among variable, abstraction, and application. More precisely, for any* BHist *packet whose* LambdaCalc$^{\uparrow}$ *carrier predicate and term-packet predicate are accepted, the carried* Tree$^{\uparrow}$ *spine supplies a* UnaryHistory *constructor tag whose* Cont *row and* hsame *endpoint rows select one of those three cases, and the constructor-tag ledger excludes the other two cases.*

Proof. Read the TreeBHistCarrier row from the LambdaCalcBHistTermCarrier packet. The row exposes the visible Hist spine, the BHist constructor ledger, and the unary tag used by the Tree$^{\uparrow}$ syntax endpoint.

Perform constructor inspection on that tag. The variable, abstraction, and application rows are the only admitted tags in the term-packet predicate, and each row carries its own Cont continuation data and hsame endpoint alignment.

The tag ledger is single-valued inside the displayed BHist packet. Thus an accepted row lands in exactly one constructor case, and the two remaining constructor tags are excluded by the same ledger field. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermCarrier constructor case exhaustion`

Theorem 200.15 (LambdaCalcUp alpha beta ledger separation)**.** *For every carried* Hist *endpoint satisfying* LambdaCalcBHistTermCarrier *and the LambdaCalc term-packet predicates, alpha-classifier rows and beta-redex rows consume disjoint visible* BHist *ledger fields except for the shared carried syntax row and the shared binder-index rows. The alpha side reads only the* TreeBHistCarrier *spine,* UnaryHistory *binder names,* Cont *renaming ledger, and* hsame *binder transport rows, while the beta side reads the application-abstraction redex row, the substitution packet, its* Cont *continuation ledger, and the output* hsame *endpoint row.*

Proof. Project the common syntax and binder-index rows from the LambdaCalcBHistTermCarrier packet. These are the only fields shared by alpha conversion and beta reduction, because both classifiers must address the same displayed Tree$^{\uparrow}$ term and binder coordinates.

For the alpha classifier, read the renaming and binder-transport ledger. Its continuation rows name the permitted capture-avoiding binder changes and do not mention an application redex or a substitution output endpoint.

For the beta-redex classifier, read the application, abstraction body, argument, substitution packet, and output endpoint rows. These fields are separate from the alpha renaming ledger, so the two classifiers can overlap only at carried syntax and binder-index data. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha beta ledger separation`

Theorem 200.16 (LambdaCalcUp substitution carrier preservation)**.** *Let a substitution ledger be admitted by Definition 200.12 for a carried input* Hist *endpoint satisfying* LambdaCalcBHistTermCarrier *and the LambdaCalc term-packet predicates. If the ledger records its output endpoint inside* BHist *with the displayed* TreeBHistCarrier *syntax row,* UnaryHistory *binder-index row,* Cont *substitution continuation, and* hsame *endpoint transport rows, then the output endpoint again satisfies* LambdaCalcBHistTermCar

Proof. Apply Definition 200.12 to obtain the syntax, binder-index, capture-avoidance, and provenance rows that the substitution endpoint is allowed to read.

The ledger transports the input tree spine and binder coordinates through the renaming, shift, and substitution continuations. Each transported row remains a displayed BHist row with the required hsame endpoint alignment.

Repack the transported TreeBHistCarrier row, unary binder row, substitution Cont ledger, and provenance row as the LambdaCalc carrier for the output endpoint. No host lambda syntax equality or quotient alpha-class is used in the preservation step. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier substitution carrier preservation`

Theorem 200.17 (LambdaCalcUp substitution output determinacy)**.** *Let two LambdaCalc substitution ledgers act on the same carried input endpoint and suppose they expose the same* $\mathsf{Tree}^{\uparrow}$ *syntax row, the same* $\mathsf{Nat}^{\uparrow}$ *binder index row, the same substituent carrier row, and the same capture-avoidance* Cont *provenance inside* BHist*. Then their output endpoints are classifier-equivalent LambdaCalc carrier endpoints through the displayed* hsame *and* Pkg *rows.*

Proof. Apply Definition 200.12 to both ledgers. The two readings expose the same TreeUp syntax row, NatUp binder index row, substituent carrier, capture-avoidance continuation, and provenance packet.

Use Definition 200.16 for each output. Both endpoints are repacked from the same TreeBHistCarrier syntax row, the same UnaryHistory binder-index row, and the same substitution Cont ledger.

The classifier comparison is therefore the displayed hsame endpoint transport between two outputs of one visible ledger packet. Repack this comparison with the common Pkg provenance row; no host substitution function, alpha quotient, or unlisted syntax equality is used. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier substitution output determinacy`

Theorem 200.18 (LambdaCalcUp alpha beta constructor transport)**.** *For an application-abstraction beta redex carried by* LambdaCalcBHistTermCarrier*, alpha transport along the displayed TreeUp syntax row, NatUp binder-index row,* Cont *renaming ledger, and* hsame *binder rows preserves the visible beta constructor row and its substitution endpoint. Thus alpha-related redex packets expose the same application constructor, abstraction body, argument row, and beta-contractum endpoint as LambdaCalc classifier data.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha beta constructor transport`

Proof. Read the redex through Definition 200.14. The application tag, abstraction tag, body row, and argument row are the visible constructor fields of one TreeBHistCarrier packet.

Apply Definition 200.15. Alpha transport may change only the renaming and binder-transport rows attached to the common syntax and binder coordinates; it does not replace the beta-redex constructor fields.

Finally apply Definition 200.16 to the transported beta substitution ledger. The substitution endpoint remains a LambdaCalc carrier endpoint over the same BHist packet, so the beta constructor row and its contractum read back unchanged at the public classifier surface. □

Theorem 200.19 (LambdaCalcUp free-variable ledger exactness)**.** *For a carried LambdaCalc term endpoint, the accepted free-variable endpoints are exactly the endpoints projected from a packet of Definition 200.6. The packet rows are the displayed* $\mathsf{Tree}^{\uparrow}$ *syntax spine,* $\mathsf{Nat}^{\uparrow}$ *binder-index ledger, free-variable row list, alpha row, beta-redex row, substitution* Cont *provenance, traversal order, and* Pkg *provenance; no host variable set or external syntax predicate is part of the accepted surface.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier free variable ledger exactness`

Proof. For soundness, apply Definition 200.7. Every accepted free-variable endpoint projects from the displayed packet rows $T, I, F, A, B, S, \lambda, \rho$, and the endpoint appears in the finite free-variable row list F.

For completeness, start from a row of Definition 200.6. The TreeUp syntax spine and NatUp binder-index ledger determine the variable, abstraction, and application branches of the traversal, while the Cont provenance records the finite route to each endpoint listed in F.

Use Definition 200.8 when the endpoint is transported through substitution provenance, and use Definition 200.9 when alpha or beta rows share the syntax endpoint. Repacking these rows with the common Pkg provenance gives exactly the accepted free-variable endpoints. □

Theorem 200.20 (LambdaCalcUp beta-redex packet exactness)**.** *Every admitted LambdaCalc beta-redex classifier endpoint is exposed by a finite packet consisting of a carried application row, carried abstraction body, carried argument row, substitution* Cont *ledger, output endpoint,* hsame *endpoint rows, and* Pkg *provenance. The packet is entirely a* Hist *and* BHist *object over* LambdaCalcBHistTermCarrier *and exports no host beta quotient.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBetaRedexPacket exactness`

Proof. Unfold the beta-redex classifier packet. The visible application row supplies the outer TreeBHistCarrier spine, while the abstraction and argument rows expose the two carried subterms as LambdaCalcBHistTermCarrier endpoints.

The substitution field is then read as a displayed Cont ledger from the abstraction body and argument packet to the output endpoint. Its endpoint alignment is recorded by the listed hsame rows, so the output is not identified by an ambient lambda quotient.

Finally, read the Pkg provenance row. Repacking these rows gives exactly the finite beta-redex packet: application, abstraction body, argument, substitution continuation, output endpoint, endpoint transports, and provenance, all inside the displayed Hist and BHist ledger. □

Theorem 200.21 (LambdaCalcUp alpha-renaming freshness transport)**.** *For any carried LambdaCalc endpoint satisfying* LambdaCalcBHistTermCarrier*, an alpha-renaming ledger that displays fresh binder transport preserves the carrier and the free-variable endpoint packet along the listed* TreeUp*,* NatUp*,* Cont*, and* hsame *rows.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha renaming freshness transport`

Proof. Project the source carrier as a TreeBHistCarrier syntax row together with its unary binder and free-variable endpoint packet. The freshness assumption is a visible ledger row, so it is read from the same BHist packet rather than from a host-side name convention.

Use the NatUp binder row to transport the selected binder index, and use the TreeUp row to transport the syntax spine through the renaming continuation. The displayed Cont ledger records the capture-avoiding renaming route.

The hsame endpoint rows align the transported syntax and free-variable packet with the output endpoint. Repacking those rows gives LambdaCalcBHistTermCarrier for the renamed term and preserves the free-variable endpoint packet under the displayed freshness transport. □

Theorem 200.22 (LambdaCalcUp substitution composition carrier closure)**.** *Let two consecutive substitution ledgers start from a carried* Hist *endpoint satisfying* LambdaCalcBHistTermCarrier*. If the first ledger records an intermediate carried endpoint and the second ledger records a final endpoint, with both ledgers displayed inside* BHist *by* TreeBHistCarrier *rows,* UnaryHistory *binder rows,* Cont *substitution routes,* hsame *endpoint rows, and* Pkg *provenance, then the composed substitution route has a final endpoint satisfying* LambdaCalcBHistTermCarrier*.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier substitution composition carrier closure`

Proof. Apply Definition 200.16 to the first substitution ledger. This yields a carried intermediate endpoint with the transported tree, binder, continuation, endpoint-alignment, and provenance rows.

Apply Definition 200.16 again to the second substitution ledger, now using the intermediate carrier as its input. The second application supplies the final carried endpoint and its displayed BHist packet.

The two Cont ledgers compose by continuation associativity, and the successive hsame endpoint rows transport along the same route. Repacking the final TreeBHistCarrier row, UnaryHistory binder row, composed substitution route, endpoint alignment, and Pkg provenance gives the claimed LambdaCalc carrier closure. □

Theorem 200.23 (LambdaCalc NameCert carrier obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$ *carrier is exactly the* LambdaCalcBHistTermCarrier *packet over* $\mathsf{Tree}^{\uparrow}$ *syntax,* $\mathsf{Nat}^{\uparrow}$ *unary binder rows, displayed* Cont *substitution provenance, and finite* Pkg *rows. It contains no host syntax equality, quotient alpha-class, or ambient variable-set carrier.*

Proof. Read the constructor rows through Definition 200.2 and Definition 200.3. These give the variable, abstraction, and application packets over $\mathsf{Tree}^{\uparrow}$ syntax and $\mathsf{Nat}^{\uparrow}$ binder rows.

Read the substitution part through Definition 200.16 and Definition 200.22. Their Cont rows keep every admitted substitution endpoint inside the same displayed BHist packet.

Finally attach the Pkg provenance from the constructor and substitution rows. The resulting packet is precisely LambdaCalcBHistTermCarrier, so the public carrier has no extra host-side syntax or alpha-equivalence field. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier namecert carrier obligation`

Theorem 200.24 (LambdaCalc NameCert classifier stability)**.** *Componentwise* hsame *on the* $\mathsf{Tree}^{\uparrow}$ *spine, unary binder indices, free-variable ledger, alpha ledger, beta-redex ledger, and provenance transports accepted LambdaCalc packets to accepted LambdaCalc packets.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier namecert classifier stability`

Proof. First transport the syntax and binder rows by Definition 200.9. This preserves the accepted constructor packet and aligns the visible $\mathsf{Tree}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ faces.

Next use Definition 200.19 for the free-variable rows and Definition 200.21 for the alpha ledger. The displayed transports keep the free-variable and freshness packets inside the LambdaCalc carrier.

For beta and substitution rows, apply Definition 200.18 and Definition 200.22. Repacking the transported rows with the shared Pkg provenance gives an accepted LambdaCalc packet at the target endpoint. □

Theorem 200.25 (LambdaCalc NameCert substitution ledger)**.** *Every admitted LambdaCalc substitution endpoint is read from the displayed* BHist *traversal and capture-avoidance* Cont *ledger, together with the listed* $\mathsf{Tree}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, hsame, *and* Pkg *rows. No host syntax equality, ambient variable set, or quotient alpha-class is used to identify the endpoint.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier namecert substitution ledger`

Proof. Start from Definition 200.4. It localizes the substitution endpoint to the displayed traversal, the binder rows, and the capture-avoidance continuation.

Then apply Definition 200.19 and Definition 200.8. These identify the free-variable rows that may pass through the ledger and show that the capture-avoidance route remains a LambdaCalc carrier route.

Finally use Definition 200.9 and Definition 200.22 to align successive endpoint transports. The endpoint is therefore read only from the displayed traversal and its Cont provenance. □

Theorem 200.26 (LambdaCalc root constructor scope)**.** *The variable, abstraction, and application constructor rows of* $\mathsf{LambdaCalc}^{\uparrow}$ *are exactly the* BHist, $\mathsf{Tree}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, hsame, *and* Pkg *rows exposed by the root* LambdaCalcBHistTermCarrier.

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier root constructor scope`

Proof. Apply Definition 200.14. It exhausts the accepted root carrier by the three visible constructor cases and reads their payloads from the displayed tree spine, unary index rows, and endpoint rows.

Definition 200.25 supplies the shared Pkg provenance, with no host syntax row or quotient alpha-class. □

Theorem 200.27 (LambdaCalc root substitution scope)**.** *Every root substitution endpoint is read only from the displayed traversal, the capture-avoidance* Cont *ledger, the binder rows, and the package provenance of the LambdaCalc root carrier.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier root substitution scope`

Proof. Apply Definition 200.4. The substitution endpoint is localized to the displayed traversal, unary binder coordinates, recursive Cont route, and hsame endpoint readback.

Then use Definition 200.25 to attach package provenance and exclude host substitution or quotient rows. □

Theorem 200.28 (LambdaCalc root beta ledger scope)**.** *Every beta-redex read at the* $\mathsf{LambdaCalc}^{\uparrow}$ *root factors through the constructor carrier, the substitution ledger, alpha transport, and the beta-redex packet exactness row, with no host equality or quotient alpha-class.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier root beta ledger scope`

Proof. Use Definition 200.14 to read the application and abstraction rows from the constructor carrier. Then use Definition 200.4 for the substitution endpoint.

Apply Definition 200.9 to carry the redex through alpha transport and Definition 200.20 to isolate the finite packet. Definition 200.25 supplies the package provenance, so no host equality or quotient row is read. □

Definition 200.29 (LambdaCalc root constructor readback packet)**.** *A LambdaCalc root constructor readback packet is the finite* BHist *packet consisting of the visible* $\mathsf{Tree}^{\uparrow}$ *syntax row, the* $\mathsf{Nat}^{\uparrow}$ *binder or variable-index row when the root tag carries one, the constructor tag* lamVar*,* lamAbs*, or* lamApp*, the constructor-preserving* Cont *provenance, the root* hsame *readback row, and the* Pkg *provenance tying those rows to* LambdaCalcBHistTermCarrier*. No host syntax tree, ambient variable set, or quotient alpha-class is part of the packet.*

Theorem 200.30 (LambdaCalc root constructor readback exhaustion)**.** *Every accepted variable, abstraction, or application row of* LambdaCalcBHistTermCarrier *projects to exactly one constructor case of Definition 200.29. The projection uses only the displayed* $\mathsf{Tree}^{\uparrow}$ *row, the visible* $\mathsf{Nat}^{\uparrow}$ *index rows,* Cont*,* hsame*, and* Pkg *provenance.*

Proof. For a variable row, apply Definition 200.2. The unary index is the visible $\mathsf{Nat}^{\uparrow}$ payload and the root readback row is the displayed lamVar tag.

For abstraction and application rows, use Definition 200.3. The theorem keeps the binder row, body row, and two application subtrees inside the visible $\mathsf{Tree}^{\uparrow}$ spine with constructor-preserving Cont transport.

The case split is exhausted by Definition 200.14. Repacking each case with the shared Pkg provenance gives exactly one readback packet and introduces no extra equality relation. □

Theorem 200.31 (LambdaCalc root beta-alpha consumer threshold)**.** *Every beta-redex, alpha-renaming, and substitution consumer for* $\mathsf{LambdaCalc}^{\uparrow}$ *reads its root data from Definition 200.29 together with the finite substitution ledger of Definition 200.25. Thus the consumer surface uses only* BHist *constructor packets,* Cont*,* hsame*, and* Pkg *provenance.*

Proof. First project each consumed root through Definition 200.30. This supplies the visible constructor tag, TreeUp syntax row, NatUp index row when present, and the root readback provenance.

Next attach the free-variable and capture-avoidance ledgers. Apply Definition 200.7 and Definition 200.8; these keep the substitution endpoint inside the displayed carrier and record the route as Cont data.

Finally transport alpha and beta endpoints by Definition 200.9. The finite ledger of Definition 200.25 supplies the remaining substitution provenance, so beta, alpha, and substitution consumers cross the root boundary only through the displayed packets. □

Definition 200.32 (LambdaCalc root normal form packet)**.** *A LambdaCalc root normal form packet is the finite* BHist *packet whose source is a root constructor readback packet of Definition 200.29, whose visible syntax row lies in* $\mathsf{Tree}^{\uparrow}$*, whose binder and variable-index rows lie in* $\mathsf{Nat}^{\uparrow}$ *when present, and whose reduction frontier records that the root is not a beta-redex visible through the displayed* Cont*,* hsame*, and* Pkg *provenance. The packet contains only constructor tags, finite subtree rows, NatUp index rows, capture-avoidance continuation rows, root readback rows, and the package provenance of* LambdaCalcBHistTermCarrier*.*

Theorem 200.33 (LambdaCalc root normal form readback)**.** *Every accepted* $\mathsf{LambdaCalc}^{\uparrow}$ *root whose constructor readback packet has no visible beta-redex exposure reads back to exactly one root normal form packet. The readback uses the same* $\mathsf{Tree}^{\uparrow}$ *syntax row, the same* $\mathsf{Nat}^{\uparrow}$ *binder or variable-index payload when present, and the same* Cont*,* hsame*, and* Pkg *provenance as the constructor readback packet.*

Proof. First project the accepted root through Definition 200.30. This gives the visible constructor tag and the finite TreeUp and NatUp rows, with no ambient syntax object.

Next use the beta ledger scope theorem Definition 200.28. The hypothesis that no visible beta-redex exposure is present excludes exactly the application-over- abstraction root case while leaving the variable, abstraction, and non-redex application cases inside the displayed constructor carrier.

Finally repack the surviving constructor row with the original continuation, root readback, and package provenance. The packet is unique because the constructor case exhaustion theorem gives exactly one root constructor case. □

Theorem 200.34 (LambdaCalc root substitution normal form consumer)**.** *Every substitution, alpha, or beta consumer that needs a LambdaCalc root normal form may read only the packet of Definition 200.32, the finite substitution ledger of Definition 200.25, and the alpha transport of Definition 200.9. It may not read a host substitution function, a quotient alpha-class, or a syntax tree outside* LambdaCalcBHistTermCarrier*.*

Proof. Begin with the consumer threshold Definition 200.31. It already restricts beta, alpha, and substitution consumers to constructor packets, finite substitution ledgers, Cont, hsame, and Pkg provenance.

Apply Definition 200.33 to the consumed root when the consumer asks for a normal-form surface. This replaces the root constructor packet by the corresponding normal form packet without changing the TreeUp syntax row, NatUp payload, continuation, or package provenance.

The remaining consumer data is exactly the finite substitution ledger and alpha transport already named by the threshold theorem. Since those rows live inside the displayed BHist carrier, no host substitution or quotient alpha object is available to the consumer. □

Theorem 200.35 (LambdaCalc constructor source disjointness)**.** *Let a carried* BHist *term row satisfy* LambdaCalcBHistTermCarrier*, and suppose its visible* $\mathsf{Tree}^{\uparrow}$ *root is one of* lamVar*,* lamAbs*, or* lamApp*. Then the public LambdaCalc classifier reads only the source fields of that root: a variable row exposes only its unary variable-index field, an abstraction row exposes only its binder-index and body fields, and an application row exposes only its left and right subterm fields. No accepted row can simultaneously be read as a different LambdaCalc constructor under the same classifier endpoint.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermCarrier constructor source disjointness`

Proof. Read the visible root tag through the $\mathsf{Tree}^{\uparrow}$ row in the carried BHist packet. The three LambdaCalc constructors are distinct TreeUp tags, so the tag readback fixes which source slots are present in the packet.

In the variable case, apply Definition 200.2. Its packet contains the unary variable-index row and no binder-body or binary subterm slots. Transport along the displayed hsame endpoint preserves that slot inventory.

In the abstraction case, apply Definition 200.3. The source packet consists of the binder-index row and the carried body row, and hsame transport cannot create the variable-only or application-only source slots.

In the application case, apply Definition 200.3. The packet consists of the two carried subterm rows, with no unary variable-index-only source and no binder-body abstraction source.

Since the classifier is read from the same $\mathsf{Tree}^{\uparrow}$ tag, the same hsame endpoint, and the same Pkg provenance in every case, two different constructor readings would require one visible root tag to be two distinct TreeUp constructors. The tag readback excludes this, giving constructor source disjointness. □

Theorem 200.36 (LambdaCalc substitution source coverage)**.** *Every accepted LambdaCalc substitution* Cont *ledger endpoint is sourced from exactly one variable, abstraction, or application branch of the displayed* $\mathsf{Tree}^{\uparrow}$ *syntax spine, together with the* $\mathsf{Nat}^{\uparrow}$ *unary index rows named by the free-variable packet. Equivalently, the substitution endpoint has no additional source outside the constructor branch chosen by the visible syntax spine, the binder-index ledger, the free-variable ledger, the* hsame *endpoint rows, and the* Pkg *provenance.*

Proof. Induct on the displayed $\mathsf{Tree}^{\uparrow}$ syntax spine of the carried source term. The base variable branch exposes the variable-index row and the matching free-variable packet endpoint, so the substitution continuation is sourced from that single variable branch.

For an abstraction spine, the visible constructor supplies the binder-index row and the body subterm. The $\mathsf{Nat}^{\uparrow}$ unary row records the binder index, while the free-variable packet names the rows that survive capture-avoiding transport through the body.

For an application spine, the visible constructor supplies the left and right subterm rows. The induction hypotheses cover the substitution endpoints of the two subspines, and the application packet repacks those endpoints into the single application branch selected by the root tag.

At each step, use Definition 200.19 to identify the free-variable rows and Definition 200.16 to transport the carried endpoint across the displayed Cont route.

The repacking step records the same hsame endpoint rows and the same Pkg provenance. Hence every accepted substitution endpoint is covered by exactly one constructor branch of the displayed syntax spine and the named $\mathsf{Nat}^{\uparrow}$ free-variable rows. □

Theorem 200.37 (LambdaCalc alpha-beta ledger determinacy)**.** *Let two accepted LambdaCalc packets expose the same* $\mathsf{Tree}^{\uparrow}$ *syntax row, the same* $\mathsf{Nat}^{\uparrow}$ *binder-index row, the same alpha ledger, the same beta-redex row, the same substitution* Cont *ledger, and the same* Pkg *provenance. Then their public LambdaCalc classifier endpoints agree by* hsame*.*

Proof. First transport the common syntax and binder rows through Definition 200.9. This aligns the carried LambdaCalc endpoints determined by the shared $\mathsf{Tree}^{\uparrow}$ syntax and $\mathsf{Nat}^{\uparrow}$ binder-index rows.

Next apply Definition 200.17 to the shared substitution Cont ledger. The two packets therefore expose the same substitution output endpoint after the displayed endpoint transports.

Finally apply Definition 200.15. The common alpha ledger and common beta-redex row cannot introduce a second public endpoint once the syntax, binder, substitution ledger, and provenance rows have been fixed.

The remaining comparison is componentwise hsame transport along the shared rows. Repacking those transports with the common Pkg provenance gives the same public LambdaCalc classifier endpoint for both packets. □

Theorem 200.38 (LambdaCalc root constructor frontier)**.** *The public LambdaCalc term frontier exposed by a carried* BHist *row is exhausted by the three visible root constructors* lamVar*,* lamAbs*, and* lamApp*. Each accepted root row reads exactly the* $\mathsf{Tree}^{\uparrow}$ *source slots,* $\mathsf{Nat}^{\uparrow}$ *index rows,* hsame *endpoint rows, and* Pkg *provenance assigned to that constructor branch, and no fourth public root constructor can be read from the same packet.*

Proof. Start with the carried BHist packet and read its visible $\mathsf{Tree}^{\uparrow}$ root. The constructor packet theorems cover the variable, abstraction, and application cases as the public constructors of the LambdaCalc carrier.

Apply Definition 200.35. It separates the source slots attached to the three visible roots and rules out a second constructor reading at the same endpoint.

For substitution or beta-visible rows attached to the root, use Definition 200.36. Those rows are still sourced from the constructor branch selected by the same syntax spine and the named $\mathsf{Nat}^{\uparrow}$, hsame, and Pkg rows.

Thus the public frontier is exactly the listed constructor inventory, with all readable rows anchored in the displayed BHist packet. □

Theorem 200.39 (LambdaCalc root substitution frontier)**.** *For every accepted LambdaCalc root substitution endpoint, the substitution provenance is preserved through the displayed* Cont *continuation and the carried* Pkg *rows. The endpoint remains attached to the same visible* BHist *constructor branch, the same* $\mathsf{Tree}^{\uparrow}$ *syntax spine, the same* $\mathsf{Nat}^{\uparrow}$ *binder and free-variable rows, and the same componentwise* hsame *transport used by the source packet.*

Proof. Begin with the accepted root substitution endpoint. Apply Definition 200.36 to identify the unique constructor branch, binder-index row, free-variable packet, and Pkg provenance that source the endpoint.

Next use Definition 200.25. It reads the endpoint from the displayed BHist traversal and capture-avoidance Cont ledger, so no ambient syntax equality or unrecorded variable set is available as a second source.

Finally apply Definition 200.22. Composition of the accepted continuations preserves the carried LambdaCalc endpoint and reuses the same Pkg provenance. The root substitution frontier is therefore stable under the displayed Cont and Pkg rows. □

Theorem 200.40 (LambdaCalc root beta-ledger frontier)**.** *Suppose two accepted LambdaCalc beta-ledger root packets expose the same* $\mathsf{Tree}^{\uparrow}$ *syntax row, the same* $\mathsf{Nat}^{\uparrow}$ *binder-index row, the same alpha ledger, the same beta-redex row, the same substitution* Cont *ledger, and the same* Pkg *provenance. Under the displayed alpha-beta classifier, the two root packets expose the same public LambdaCalc endpoint by componentwise* hsame*.*

Proof. First apply Definition 200.15. The alpha ledger and beta-redex row occupy disjoint readable positions once the syntax spine, binder row, substitution ledger, and provenance are fixed.

Next apply Definition 200.37. Its hypotheses are exactly the shared $\mathsf{Tree}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, alpha, beta, substitution, and Pkg rows displayed by the root packets.

The resulting hsame comparison is the endpoint comparison required by the public alpha-beta classifier. Hence the beta-ledger frontier has a deterministic root endpoint under the displayed classifier. □

Theorem 200.41 (LambdaCalc root obligation inventory)**.** *The displayed LambdaCalc root packet has the following obligation inventory: carrier rows for the visible* BHist *constructor branches, classifier rows for the* $\mathsf{Tree}^{\uparrow}$ *syntax spine and* $\mathsf{Nat}^{\uparrow}$ *binder and free-variable indices, substitution rows transported by* Cont*, alpha-beta rows transported by componentwise* hsame*, exactness rows for the readable constructor and substitution frontiers, stability rows for the beta-ledger classifier, and ledger rows recording* Pkg *provenance and* NameCert *source data. The scope of the inventory is exactly the* BHist*,* $\mathsf{Tree}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* Cont*,* hsame*,* Pkg*, and* NameCert *rows exposed by Definition 200.38, Definition 200.39, and Definition 200.40; it exports no host syntax equality, alpha quotient, beta quotient, or unrecorded variable-set primitive.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcUp StdBridge`

Proof. Project the root packet through Definition 200.38. This supplies the visible constructor branches, the $\mathsf{Tree}^{\uparrow}$ syntax spine, the $\mathsf{Nat}^{\uparrow}$ binder and free-variable rows, and the readable BHist carrier rows.

Next apply Definition 200.39. It accounts for the substitution endpoints, their capture-avoidance Cont ledger, and the carried Pkg provenance without adding a second syntax source.

Finally apply Definition 200.40. The alpha ledger, beta-redex row, substitution ledger, and provenance rows determine the public endpoint through componentwise hsame. Repacking these projections gives exactly the listed carrier, classifier, substitution, alpha-beta, exactness, stability, and ledger obligations. □

Theorem 200.42 (LambdaCalc root obligation carrier)**.** *The variable, abstraction, and application* BHist *constructor rows cover the* $\mathsf{LambdaCalc}^{\uparrow}$ *carrier under* TreeBHistCarrier*,* UnaryHistory*,* hsame*, and constructor-preserving* Cont*. No carrier row is read from host syntax equality or from an alpha-quotient representative.*

Proof. Apply Definition 200.2 to the variable case. The root packet exposes the visible $\mathsf{Tree}^{\uparrow}$ constructor tag and the $\mathsf{Nat}^{\uparrow}$ unary variable index.

Apply Definition 200.3 to the abstraction and application cases. These rows expose the binder index, body, left subterm, and right subterm through the same $\mathsf{Tree}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ visible spines.

Finally use Definition 200.14 and Definition 200.38. The constructor cases exhaust the carrier and transport through the displayed hsame and constructor-preserving Cont rows, so no host syntax row is added. □

Theorem 200.43 (LambdaCalc root obligation classifier)**.** *The beta and alpha classifier rows of the* $\mathsf{LambdaCalc}^{\uparrow}$ *root packet are stable under constructor-preserving* hsame *transport. They read only the visible* $\mathsf{Tree}^{\uparrow}$ *syntax spine,* $\mathsf{Nat}^{\uparrow}$ *binder and free-variable rows, alpha-beta ledger rows, substitution* Cont *rows, and* Pkg *provenance, and they do not use host syntax equality.*

Proof. First align the constructor endpoints by Definition 200.9. This transports the visible $\mathsf{Tree}^{\uparrow}$ syntax row and the $\mathsf{Nat}^{\uparrow}$ binder rows componentwise.

Next use Definition 200.37. Shared syntax, binder, alpha, beta, substitution, and provenance rows determine the same public endpoint by hsame.

The root beta-ledger frontier Definition 200.40 repacks the transported rows as the public classifier surface. Since the cited rows are all displayed constructor, ledger, transport, and provenance rows, host syntax equality is not consumed. □

Theorem 200.44 (LambdaCalc root obligation substitution ledger)**.** *Capture-avoiding substitution at the* $\mathsf{LambdaCalc}^{\uparrow}$ *root exposes only* $\mathsf{Nat}^{\uparrow}$ *index rows,* $\mathsf{Tree}^{\uparrow}$ *subtree rows,* BHist *constructor tags, displayed* Cont *substitution routes,* hsame *endpoint rows, and* Pkg *package provenance to downstream consumers.*

Proof. Use Definition 200.36. It identifies each accepted substitution endpoint with one visible constructor branch and the named $\mathsf{Nat}^{\uparrow}$ binder and free-variable rows.

Then apply Definition 200.25 and Definition 200.39. These theorems attach the displayed Cont capture-avoidance route, the endpoint hsame rows, and the same Pkg provenance to the substitution endpoint.

The downstream read is therefore exactly the NatUp, TreeUp, constructor-tag, continuation, endpoint-transport, and provenance surface. No host substitution function, variable-set primitive, or quotient class is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$**.**

Constructive story: A lambda-calculus term is generated by a visible $\mathsf{Tree}^{\uparrow}$ syntax spine whose root tag is one of variable, abstraction, or application, with $\mathsf{Nat}^{\uparrow}$ unary histories naming variable indices, binder indices, and free-variable

rows, and with Cont ledgers recording substitution, alpha, and beta routes. The carrier accepts exactly those BHist packets whose constructor tag, payload rows, subtrees, substitution output, alpha ledger, beta-redex row, and provenance are displayed; a host syntax tree, unrecorded variable set, alpha quotient, or beta quotient does not enter the carrier. Two accepted root packets are classified only by componentwise transport of the TreeUp root, NatUp indices, substitution continuation, beta row, alpha row, endpoint hsame comparisons, and Pkg provenance, not by ambient syntax conversion. The source specification is the TreeUp/NatUp constructor surface, the pattern specification records variable, abstraction, and application closure, constructor disjointness, substitution source coverage, capture-avoidance continuation closure, alpha-beta determinacy, and root beta-ledger exactness, and the ledger keeps the constructor branch and every recursive continuation route visible. At this seed frontier the five name-certificate obligations are not yet exported as a complete public certificate, but the rows show the mechanism: a displayed constructor packet inhabits the carrier, hsame gives reflexive, symmetric, and transitive endpoint movement, and constructor-preserving Cont transport keeps classified terms inside the same visible syntax spine. A packet h lies in the signature gap of a $\mathsf{LambdaCalc}^{\uparrow}$ token p when the source-domain accepts its constructor and ledger rows, the signature relation pairs it with an answer term packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the root tag, NatUp indices, substitution route, alpha row, beta-redex row, and provenance are exactly the rows encoded by p. The chapter registers this root-frontier construction under the name $\mathsf{LambdaCalc}^{\uparrow}$; it exposes only the BEDC syntax and reduction ledgers, with no host lambda syntax equality, alpha quotient, beta quotient, or complete standard bridge introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope binds the carrier, classifier, substitution, alpha-beta, exactness, stability, and ledger rows listed in Definition 200.41 for the displayed LambdaCalc root packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.LambdaCalcUp.LambdaCalcUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This frontier does not close host lambda syntax equality, alpha quotienting, beta quotienting, or analytic normalization.

Upgrade path: No further paper-side theory or verification grade remains for the displayed constructor, substitution, alpha, beta, root normal-form, and public NameCert consumer catalogue.

Theorem 200.45 (LambdaCalc alpha classifier transitivity)**.** *Let s, t, u : Hist be accepted $\mathsf{LambdaCalc}^{\uparrow}$ term packets whose visible term data are carried by the displayed $\mathsf{Tree}^{\uparrow}$ syntax skeletons, $\mathsf{Nat}^{\uparrow}$ binder-index ledgers, Cont alpha-renaming routes, endpoint hsame rows, and common Pkg provenance. If an accepted alpha-classifier row relates (s, t) and an accepted alpha-classifier row relates (t, u) over the same middle packet t, then the composed alpha-classifier row relates (s, u). The composed row is obtained by keeping the two endpoint carriers, reversing only those carried hsame rows whose displayed orientation points into the middle packet, concatenating the resulting $\mathsf{Tree}^{\uparrow}$ skeleton comparisons and $\mathsf{Nat}^{\uparrow}$ binder-index comparisons, and repacking the concatenated alpha ledger with the same Cont and Pkg provenance.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier alpha classifier transitive`

Proof. Project the two accepted alpha-classifier rows through Definition 200.48. Each row is a componentwise comparison of the visible $\mathsf{Tree}^{\uparrow}$ syntax skeleton, the visible $\mathsf{Nat}^{\uparrow}$ binder and variable-index rows, the alpha-renaming ledger, the endpoint hsame rows, and the carried Pkg provenance. No host syntax conversion or quotient alpha-class is available as an extra step.

Use Definition 200.11 only to orient the carried comparisons at the common middle packet. Thus every component has a row from the s-packet to the displayed middle t-packet and a row from that same middle packet to the u-packet. The middle syntax skeleton and middle binder ledger are the same displayed rows in the two hypotheses, so no additional matching datum is required.

Concatenate the two oriented hsame comparisons on the $\mathsf{Tree}^{\uparrow}$ skeletons. Do the same for each $\mathsf{Nat}^{\uparrow}$ binder or variable-index row named by the alpha ledger. The alpha-renaming routes are finite Cont rows attached to those same visible components, so their carried provenance concatenates along the middle packet while preserving constructor tags and binder positions.

Apply Definition 200.21 to the concatenated alpha route. It keeps the endpoint inside LambdaCalcBHistTermCarr preserves the free-variable packet, and records the transported syntax and binder rows by the concatenated hsame comparisons. Apply Definition 200.9 to read the resulting endpoints as LambdaCalc carrier data over the displayed syntax and binder ledgers.

Finally repack the concatenated syntax comparison, concatenated binder-index comparison, concatenated alpha-renaming Cont route, endpoint hsame rows, and common Pkg provenance. By Definition 200.49, these are exactly the rows an accepted public LambdaCalc alpha read may consume. Hence the repacked row is an accepted alpha-classifier row relating (s, u). □

Theorem 200.46 (LambdaCalc alpha classifier reflexivity)**.** *Let h : Hist be an accepted* $\mathsf{LambdaCalc}^{\uparrow}$ *term packet satisfying* LambdaCalcBHistTermCarrier*. Then the LambdaCalc alpha-classifier accepts a row relating (h, h). This reflexive row is read over the displayed* $\mathsf{Tree}^{\uparrow}$ *syntax skeleton of h, the displayed* $\mathsf{Nat}^{\uparrow}$ *binder-index and variable-index ledgers of h, the identity alpha-renaming* Cont *ledger, the componentwise reflexive* hsame *rows on each displayed endpoint, and the same* Pkg *provenance carried by h. It exports no host syntax equality, unrecorded variable set, or quotient alpha-class.*

Proof. Project the accepted packet through Definition 200.48. The public classifier boundary is componentwise: it contains the visible $\mathsf{Tree}^{\uparrow}$ syntax root, the $\mathsf{Nat}^{\uparrow}$ binder and variable rows, the substitution and alpha ledgers, endpoint hsame comparisons, and the carried Pkg provenance.

Use the carrier self-classification of Definition 200.10 on the same packet h. It supplies reflexive hsame rows for the visible syntax skeleton, every unary binder or variable index named by the packet, the free-variable ledger, and the package provenance. These are exactly the endpoint comparisons needed when both classifier endpoints are h.

Take the alpha-renaming ledger to be the identity Cont route on the displayed syntax and binder rows. Applying Definition 200.21 to this identity route preserves LambdaCalcBHistTermCarrier, preserves the free-variable endpoint packet, and records the transported syntax and binder rows by the reflexive hsame comparisons already supplied above.

Finally repack the syntax row, binder-index rows, identity alpha ledger, reflexive endpoint comparisons, and unchanged provenance. By Definition 200.49, these rows exhaust the public LambdaCalc alpha read. Hence the repacked row is an accepted alpha-classifier row relating (h, h), with no extra equality principle outside the displayed BHist packet. □

Theorem 200.47 (LambdaCalc NameCert public boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$ *boundary is exactly the constructor frontier, substitution ledger, beta ledger, alpha ledger,* hsame *endpoint comparisons,* Cont *continuation rows, and* Pkg *provenance exposed by the carried* BHist *term packets. Thus a public read of a LambdaCalc endpoint is sourced by the visible* $\mathsf{Tree}^{\uparrow}$ *constructor root, the relevant* $\mathsf{Nat}^{\uparrow}$ *binder or variable rows, the substitution and beta rows attached to that root, and the listed BEDC transport rows.*

Proof. Begin with the carried term packet. The carrier rows are supplied by Definition 200.2 and Definition 200.3, while Definition 200.38 exhausts the public root by the three visible $\mathsf{Tree}^{\uparrow}$ constructors.

For substitution reads, apply Definition 200.25 and Definition 200.39. These identify the displayed Cont ledger, the relevant $\mathsf{Nat}^{\uparrow}$ binder and free-variable rows, and the same Pkg provenance as the only substitution source.

For beta and alpha reads, apply Definition 200.9 and Definition 200.40. The public endpoint is transported componentwise by hsame after the alpha ledger, beta-redex ledger, substitution ledger, and constructor rows have been fixed.

Repacking these rows gives the stated public boundary. No host syntax equality, unrecorded variable set, or quotient alpha-class is available outside the displayed BHist, Cont, hsame, and Pkg rows. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier public boundary`

Theorem 200.48 (LambdaCalc NameCert classifier boundary)**.** *Classifier movement for* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$ *is componentwise over the* $\mathsf{Tree}^{\uparrow}$ *syntax root, the* $\mathsf{Nat}^{\uparrow}$ *binder and variable rows, the substitution* Cont *ledger, the beta-redex ledger, the alpha ledger, the endpoint* hsame *comparisons, and the carried* Pkg *provenance. Two accepted classifier endpoints therefore agree only by transporting each displayed component of this boundary; no ambient syntax conversion is admitted as an additional classifier step.*

Proof. First align the constructor root. The classifier stability supplied by Definition 200.9 transports the carried LambdaCalc endpoint over the visible $\mathsf{Tree}^{\uparrow}$ syntax row and the $\mathsf{Nat}^{\uparrow}$ binder rows.

Next align substitution. Use Definition 200.36 to isolate the constructor branch and Definition 200.39 to keep the substitution endpoint attached to the displayed Cont ledger and package provenance.

Finally align beta and alpha. The determinacy theorem Definition 200.37, together with Definition 200.40, says that shared syntax, binder, alpha, beta, substitution, and Pkg rows determine the same public endpoint by hsame.

Thus classifier movement is exactly the componentwise transport of the named boundary rows. □

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier public endpoint transport`

Theorem 200.49 (LambdaCalc NameCert ledger exactness)**.** *Every accepted public LambdaCalc read is exhausted by the constructor rows, substitution rows, beta rows, alpha rows,* hsame *endpoint comparisons,* Cont *continuation rows, and* Pkg *provenance displayed in the carried* BHist *packet. In particular, the public ledger contains no host syntax equality, no implicit capture-avoidance equality outside the substitution ledger, and no quotient alpha-class.*

Lean4 ✓ `BEDC.Derived.LambdaCalcUp.LambdaCalcBHistTermPacketCarrier namecert ledger exactness`

Proof. Project the accepted read to the visible constructor spine. By Definition 200.35 and Definition 200.38, the read chooses exactly one variable, abstraction, or application root and inherits only the slots assigned to that root.

The substitution part is exact by Definition 200.36. Every substitution endpoint is sourced from the chosen constructor branch, the named $\mathsf{Nat}^{\uparrow}$ rows, the displayed Cont ledger, and the same Pkg provenance.

The beta and alpha parts are exact by Definition 200.37 and Definition 200.40. Once the syntax, binder, alpha, beta, substitution, and provenance rows are fixed, the remaining public endpoint comparison is the recorded hsame row.

Combining these projections exhausts the public ledger and excludes any host-level syntax equality or quotient alpha-class from the accepted boundary. □

Chapter 201

A Concrete Naming Certificate for $\mathsf{Partition}^{\uparrow}$

The integer-partition interface $\mathsf{Partition}^{\uparrow}$ packages weakly decreasing sequences of positive integers summing to a fixed integer, together with the Young-diagram interpretation. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28) and $\mathsf{NameCert}_{\mathsf{List}^{\uparrow}}$ (chapter 36).

201.1 Carrier definition

Definition 201.1 (PartitionUp BHist carrier)**.** *A* $\mathsf{Partition}^{\uparrow}$ *carrier is a finite* BHist *packet consisting of a* $\mathsf{List}^{\uparrow}$ *row whose entries are* $\mathsf{Nat}^{\uparrow}$ *positive-part rows, a unary sum ledger row for the displayed entries, weak-decrease comparison rows between adjacent parts, endpoint rows for the total and each part, finite* Cont *routes tying the list, sum, and comparison rows into one ledger, and* Pkg *provenance for the whole packet. The packet is accepted only through these visible rows.*

The carrier records an integer partition as BEDC data. It does not include a host integer partition set, a symmetric-function basis, a representation theory object, or an external Young-diagram model.

Lean4 (definition) `BEDC.Derived.PartitionUp.PartitionBHistCarrier`

Definition 201.2 (PartitionUp Young-ledger classifier)**.** *The Young-ledger classifier compares two accepted* $\mathsf{Partition}^{\uparrow}$ *carriers by the displayed* hsame *transport of their* $\mathsf{List}^{\uparrow}$ *rows, their component* $\mathsf{Nat}^{\uparrow}$ *positive-part rows, their unary sum rows, their weak-decrease rows, and their boundary ledgers. A classifier witness must transport the same finite Young boundary through the list and sum ledgers and must preserve* Pkg *provenance for the packet.*

Lean4 (definition) `BEDC.Derived.PartitionUp.PartitionYoungLedgerClassifier`

201.2 The certificate

Theorem 201.3 (Partition carrier obligation)**.** *Every accepted* $\mathsf{Partition}^{\uparrow}$ *carrier is read from the packet of Definition 201.1: the* $\mathsf{List}^{\uparrow}$ *row of* $\mathsf{Nat}^{\uparrow}$ *positive parts, the unary sum ledger, the weak-decrease comparison rows, the Young boundary ledger, the finite* Cont *routes, and the* Pkg *provenance. No additional partition object is available to a consumer of the carrier.*

Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier obligation`

Proof. Project the finite BHist packet to its list, part, sum, comparison, boundary, route, and provenance coordinates. These are the only rows named by the carrier definition.

The $\mathsf{List}^{\uparrow}$ dependency supplies the finite row spine, and the $\mathsf{Nat}^{\uparrow}$ dependency supplies the positive unary part rows. The remaining rows are ordinary ledgers attached to that spine by Cont and packaged by Pkg.

Since the carrier has no coordinate for a host partition set or external Young-diagram model, every accepted consumer stops at the displayed ledger surface. □

Theorem 201.4 (Partition classifier stability)**.** *Let $\mathcal{P}$ and $\mathcal{P}'$ be accepted* $\mathsf{Partition}^{\uparrow}$ *carriers. If the classifier of Definition 201.2 relates their* $\mathsf{List}^{\uparrow}$ *rows, component* $\mathsf{Nat}^{\uparrow}$ *positive-part rows, unary sum rows, weak-decrease rows, and Young boundary ledgers by displayed* hsame *transport, then the transported rows repack as an accepted* $\mathsf{Partition}^{\uparrow}$ *carrier with the same finite certificate surface.*

Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier classifier stability`

Proof. Begin with the $\mathsf{List}^{\uparrow}$ row. Its classifier transports the finite spine and therefore fixes the positions at which the part rows are read.

Next transport the component $\mathsf{Nat}^{\uparrow}$ rows and the unary sum ledger along the displayed hsame witnesses. The positive-part condition and the total row remain visible because they are ledger entries, not external arithmetic objects.

Finally transport the weak-decrease and boundary rows and concatenate the finite Cont routes in the same order. The original Pkg provenance then repacks the transported rows as an accepted carrier. □

Theorem 201.5 (Partition ledger exactness)**.** *For every accepted* $\mathsf{Partition}^{\uparrow}$ *carrier, the readable ledger consists exactly of the* $\mathsf{List}^{\uparrow}$ *row, the* $\mathsf{Nat}^{\uparrow}$ *positive-part rows, the unary sum row, the weak-decrease comparison rows, the Young boundary ledger, the displayed* hsame *transports, the finite* Cont *routes, and the* Pkg *provenance. It exposes no host partition enumeration, hook formula, symmetric-function identity, or representation-theoretic datum.*

Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier ledger exactness`

Proof. Use Definition 201.3 to reduce a consumer readback to the visible packet rows.

The exactness direction is by projection: each listed row is a coordinate of the carrier packet and is therefore readable. The converse direction is by constructor inspection: the carrier definition has no constructor for any row outside the listed ledger.

The exclusions follow from the same inspection. A host enumeration, hook formula, symmetric-function identity, or representation-theoretic datum would require an additional coordinate, and none is present. □

Theorem 201.6 (Partition finite NameCert surface)**.** *The finite* $\mathsf{NameCert}_{\mathsf{Partition}^{\uparrow}}$ *surface consists of the* BHist *carrier of Definition 201.1, the Young-ledger classifier of Definition 201.2, classifier stability from Definition 201.4, ledger exactness from Definition 201.5, and the source coverage assertion that every accepted partition row is read through the displayed* $\mathsf{List}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *packet with* Cont *routes and* Pkg *provenance.*

Proof. Use the carrier as the source predicate. It displays the list spine, positive parts, sum row, weak-decrease rows, Young boundary row, route ledger, and package provenance in one finite packet.

Use the Young-ledger classifier as the comparison predicate. Its transport obligation is exactly Definition 201.4.

The exactness and coverage clauses are supplied by Definition 201.5 and Definition 201.3. These clauses cover the finite surface and exclude all host-level partition structures outside the packet. □

Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier semantic name certificate`
Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier finite namecert surface`

Theorem 201.7 (Partition Young boundary package)**.** *The Young-diagram boundary exported by* $\mathsf{Partition}^{\uparrow}$ *is the finite ledger whose rows are the positive-part list, the unary row sums, the adjacent weak-decrease comparisons, and the boundary route obtained by concatenating the displayed* Cont *rows. The boundary is stable under the classifier of Definition 201.2 and is packaged only through the carrier's* Pkg *provenance.*

Proof. Read the boundary rows from the carrier packet. The positive-part list gives the row lengths, the unary sum ledger gives the total size, and the weak-decrease comparisons give the Young boundary order.

Concatenate the displayed Cont routes from the list row to the sum row and from the comparison rows to the boundary ledger. This produces the visible boundary route without adding a host diagram.

Apply classifier stability to transport the same boundary route across hsame-related packets. The transported route is repackaged by the same Pkg provenance surface, so the exported Young boundary remains a finite BEDC ledger. □

Lean4 ✓ `BEDC.Derived.PartitionUp.PartitionBHistCarrier young boundary package`

Theorem 201.8 (Partition bridged closure route)**.** *Every public* $\mathsf{Partition}^{\uparrow}$ *consumer that reads the Young-boundary package factors through the standard Young-ledger surface: the positive-part* $\mathsf{List}^{\uparrow}$ *row, component* $\mathsf{Nat}^{\uparrow}$ *part rows, unary size ledger, weak-decrease comparisons, displayed boundary continuation, componentwise* hsame *transport, and* Pkg *provenance. The route preserves the finite* $\mathsf{NameCert}_{\mathsf{Partition}^{\uparrow}}$ *surface and exports no host partition enumeration, hook formula, symmetric-function identity, or representation-theoretic datum.*

Proof. Read the public certificate through Definition 201.6. The available surface is the finite carrier, its Young-ledger classifier, carrier obligation, classifier stability, ledger exactness, and the Young-boundary package.

Apply Definition 201.7. The exported boundary is computed from the positive-part list, unary sum rows, adjacent weak-decrease comparisons, and the displayed continuation route, then transported by the same classifier and provenance rows.

Thus every standard Young-ledger consumer receives exactly the public $\mathsf{Partition}^{\uparrow}$ NameCert rows. The absent host enumeration, hook formula, symmetric-function identity, and representation-theoretic data remain outside the bridged paper surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Partition}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Partition}^{\uparrow}}$, maturePackageClosed)

Scope closed: The visible seed scope is the finite BHist carrier, the Young-ledger classifier, the $\mathsf{List}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ source packet, and the displayed hsame, Cont, and Pkg rows.

Constructive story: A partition packet is generated from a finite list-source row, unary size rows, adjacent weak-decrease checks, displayed boundary continuations, hsame transport, and Pkg provenance; no host enumeration or representation-theoretic object is part of the carrier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Partition}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PartitionUp.PartitionBHistCarrier\_young\_boundary\_package`

Bridge status: paperBridge

Not claimed: No host partition enumeration, hook formula, symmetric-function identity, or representation-theoretic datum is claimed.

Upgrade path: No further paper-side theory upgrade remains short of a machine-checked standard bridge for the Young-ledger consumer surface.

Chapter 202

A Concrete Naming Certificate for $\mathsf{Matroid}^{\uparrow}$

The matroid interface $\mathsf{Matroid}^{\uparrow}$ packages a finite ground set with a family of subsets (independent sets) closed under taking subsets and satisfying the exchange axiom. It depends on $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112) and $\mathsf{NameCert}_{\mathsf{Lattice}^{\uparrow}}$ (chapter 55).

202.1 Carrier definition

202.2 The certificate

Definition 202.1 (FinSetUp subset and intersection over one ground)**.** *Fix a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. For predicates* $U, V, J : \mathsf{BHist} \to \mathsf{Prop}$*, define finite-set subset over this ground by*

$$\mathsf{FinSetSubset}_{E,\sim_E}(U, V) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs, U) \wedge \forall z : \mathsf{BHist}.\ U(z) \to V(z).$$

Define the $\mathsf{FinSet}^{\uparrow}$ *intersection predicate by*

$$\mathsf{FinSetIntersection}_{E,\sim_E}(J; U, V) \iff \exists zs : \mathsf{HistSpine}.\ \mathsf{FinSetSpineEnumerates}_{E,\sim_E}(zs, J) \wedge \forall z : \mathsf{BHist}.\ J(z) \iff U(z).$$

Thus the intersection row is an exact finite-spine presentation of pointwise membership conjunction in the common ground.

Definition 202.2 (MatroidUp independent-set certificate surface)**.** *A* $\mathsf{Matroid}^{\uparrow}$ *independent-set certificate over* $(E, \sim_E)$ *consists of a finite-set predicate family*

$$\mathsf{Ind}_M : (\mathsf{BHist} \to \mathsf{Prop}) \to \mathsf{Prop}$$

and the rows

$$\begin{gathered}
\mathsf{finite}_M : \forall I.\ \mathsf{Ind}_M(I) \to \exists xs : \mathsf{HistSpine}.\ \mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs, I), \\
\mathsf{ground}_M : \forall I\, z.\ \mathsf{Ind}_M(I) \to I(z) \to E(z), \\
\mathsf{empty}_M : \mathsf{Ind}_M(\lambda z.\ \bot), \\
\mathsf{hereditary}_M : \forall I\, J.\ \mathsf{Ind}_M(I) \to \mathsf{FinSetSubset}_{E,\sim_E}(J, I) \to \mathsf{Ind}_M(J).
\end{gathered}$$

The exchange row is the certified finite-spine exchange clause on these same finite predicates: independent finite sets with certified smaller cardinal admit a carried element of the larger set whose

$\mathsf{FinSet}^{\uparrow}$ *insertion into the smaller set remains independent. The hereditary row is the subset-closure field named in the* $\mathsf{Matroid}^{\uparrow}$ *certificate paragraph of section 202.2.*

Lemma 202.3 (FinSetUp intersection projects to the left factor)**.** *Fix a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. For predicates* $I, K, J : \mathsf{BHist} \to \mathsf{Prop}$*, if*

$$\mathsf{FinSetIntersection}_{E,\sim_E}(J; I, K)$$

holds, then

$$\mathsf{FinSetSubset}_{E,\sim_E}(J, I)$$

holds. This is the finite-set form of the left lower-bound role for meet in Definition 55.102, read here from the exact $\mathsf{FinSet}^{\uparrow}$ *membership row.*

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinsetIntersection left subset`
Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinSetIntersection left subset`

Proof. Unpack Definition 202.1. The intersection hypothesis supplies a finite spine zs with

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(zs, J)$$

and a pointwise equivalence

$$\forall z : \mathsf{BHist}.\ J(z) \iff I(z) \wedge K(z).$$

To prove $\mathsf{FinSetSubset}_{E,\sim_E}(J, I)$, keep the same finite spine zs as the finite-set witness for J. It remains only to prove the inclusion row. Fix $z : \mathsf{BHist}$ and assume $J(z)$. The forward half of the displayed pointwise equivalence gives $I(z) \wedge K(z)$, and projecting its first component gives $I(z)$. Thus every member of J is a member of I, with the finite-spine witness already supplied. □

Theorem 202.4 (Matroid intersection preserves independence)**.** *Fix a* $\mathsf{Matroid}^{\uparrow}$ *independent-set certificate over a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*, with independent-family row* Ind_M *as in Definition 202.2. Let* $I, K, J : \mathsf{BHist} \to \mathsf{Prop}$ *be finite-set predicates over the same ground. Suppose*

$$\mathsf{Ind}_M(I)$$

and suppose there are finite spines xs *and* ys *such that*

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs, I) \qquad \textit{and} \qquad \mathsf{FinSetSpineEnumerates}_{E,\sim_E}(ys, K).$$

If J *is the* $\mathsf{FinSet}^{\uparrow}$ *intersection of* I *with* K*,*

$$\mathsf{FinSetIntersection}_{E,\sim_E}(J; I, K),$$

then

$$\mathsf{Ind}_M(J).$$

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinsetIntersection independence preserved`
Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidIntersection preserves independence`

Proof. The finite-spine witnesses xs and ys record that I and K are finite predicates over the same ground $(E, \sim_E)$ in the sense of Definition 112.17. The independence hypothesis gives the $\mathsf{Matroid}^{\uparrow}$ independent set I.

Apply Definition 202.3 to the displayed intersection row. This gives

$$\mathsf{FinSetSubset}_{E,\sim_E}(J, I).$$

The hereditary field $\mathsf{hereditary}_M$ of the $\mathsf{Matroid}^{\uparrow}$ certificate in Definition 202.2 applies to $\mathsf{Ind}_M(I)$ and this finite-set subset row, and yields

$$\mathsf{Ind}_M(J).$$

The exchange row and all cardinal comparison data are irrelevant for this closure step; independence is preserved solely by the certified subset-closure field. □

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinSetIntersection preserves independence`

Definition 202.5 (Certified matroid restriction rows)**.** *Fix a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. For predicates* $I, I^+ : \mathsf{BHist} \to \mathsf{Prop}$ *and a history* $x : \mathsf{BHist}$*, define the certified* $\mathsf{FinSet}^{\uparrow}$ *insertion row by*

$$\mathsf{FinSetInsert}_{E,\sim_E}(I, x, I^+) \iff E(x) \wedge \exists xs^+ : \mathsf{HistSpine}.\ \mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs^+, I^+) \wedge \forall z : \mathsf{BHist}.\ I^+(z) \Longleftarrow$$

For a certified finite-cardinality comparison predicate $\mathsf{FinSetCardLt}_{E,\sim_E}$*, the ambient exchange row for a family* Ind *is*

$$\mathsf{MatroidExchangeRow}_{E,\sim_E,\mathsf{FinSetCardLt}}(\mathsf{Ind}) \iff \forall I\ J.\ \mathsf{Ind}(I) \to \mathsf{Ind}(J) \to \mathsf{FinSetCardLt}_{E,\sim_E}(I, J) \to \exists x : \mathsf{BHist}.\ \exists I^+$$

For a finite subset predicate K *of* E *and a candidate restricted family* $\mathsf{Ind}_{M|K}$*, define the restricted* $\mathsf{Matroid}^{\uparrow}$ *row package by*

$$\forall I$$

$$\forall I.\ \mathsf{Ind}_{M|}$$

$$\mathsf{MatroidRestrictionRows}_{E,\sim_E,K,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M, \mathsf{Ind}_{M|K}) \iff$$

$$\forall I\ J$$

$$\forall I\ J.\ \mathsf{Ind}_{M|K}(I) \to \mathsf{Ind}_{M|K}(J) \to \mathsf{FinSetCardLt}_{E,\sim_E}(I,$$

The rows use the same finite-spine and cardinal-comparison surface as the ambient $\mathsf{Matroid}^{\uparrow}$ *certificate of Definition 202.2, while the ground row is narrowed to* K*.*

Lemma 202.6 (Finite-spine subsets are classifier saturated)**.** *Let* $E : \mathsf{BHist} \to \mathsf{Prop}$ *have classifier* $\sim_E$*, let* $\mathsf{cert} : \mathsf{NameCert}(E, \sim_E)$*, and let* $K : \mathsf{BHist} \to \mathsf{Prop}$*. If*

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(ks, K),$$

then for all histories $x, y : \mathsf{BHist}$*,*

$$K(x) \wedge E(y) \wedge y \sim_E x \to K(y).$$

Proof. Unfold Definition 112.17. The enumeration hypothesis turns $K(x)$ into $\mathsf{EnumMem}_{E,\sim_E}(x, ks)$. Unfolding Definition 112.1, choose a position $i : \mathsf{Pos}(ks)$ with $E(\mathsf{entry}_{ks}(i))$ and

$$x \sim_E \mathsf{entry}_{ks}(i).$$

The transitivity field of cert from Definition 9.7, applied to $y \sim_E x$ and the displayed comparison, gives

$$y \sim_E \mathsf{entry}_{ks}(i).$$

Together with $E(y)$ and the same position i, this repacks as $\mathsf{EnumMem}_{E,\sim_E}(y, ks)$. The forward half of the enumeration equivalence for y yields $K(y)$. □

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinSetSpineEnumerates classifier saturation`

Theorem 202.7 (Matroid restriction certificate)**.** *Fix a* $\mathsf{Matroid}^{\uparrow}$ *independent-set certificate over a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$ *and naming certificate* $\mathsf{cert} : \mathsf{NameCert}(E, \sim_E)$*. Let* Ind_M *be its independent-family row, with fields* finite_M*,* ground_M*,* empty_M*, and* $\mathsf{hereditary}_M$ *as in Definition 202.2. Assume also the ambient exchange row*

$$\mathsf{MatroidExchangeRow}_{E,\sim_E,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M).$$

Let $K : \mathsf{BHist} \to \mathsf{Prop}$ *be a* $\mathsf{FinSet}^{\uparrow}$ *subset of* E*:*

$$\mathsf{FinSetSubset}_{E,\sim_E}(K, E).$$

If a family $\mathsf{Ind}_{M|K}$ *is exactly the family of ambient independent predicates contained in* K*,*

$$\forall I.\ \mathsf{Ind}_{M|K}(I) \iff \mathsf{Ind}_M(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, K),$$

then it satisfies the restricted $\mathsf{Matroid}^{\uparrow}$ *row package over* K*:*

$$\mathsf{MatroidRestrictionRows}_{E,\sim_E,K,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M, \mathsf{Ind}_{M|K}).$$

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidRestrictionRows certificate`
Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidRestrictionRows exchange support boundary`

Proof. Unpack the displayed K-subset row using Definition 202.1. Choose a finite spine ks with

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(ks, K),$$

and keep the inclusion row $K(z) \to E(z)$. Unfold Definition 202.5. The defining equivalence row for $\mathsf{Ind}_{M|K}$ is exactly the displayed hypothesis.

For the finite row, assume $\mathsf{Ind}_{M|K}(I)$. The defining equivalence gives $\mathsf{Ind}_M(I)$, so finite_M supplies a finite spine enumerating I over $(E, \sim_E)$.

For the ground row over K, assume $\mathsf{Ind}_{M|K}(I)$ and $I(z)$. The defining equivalence gives $\mathsf{FinSetSubset}_{E,\sim_E}(I, K)$. Unpacking that subset row gives the inclusion $I(z) \to K(z)$, hence $K(z)$.

For the empty row, empty_M gives $\mathsf{Ind}_M(\lambda z.\ \bot)$. Applying finite_M to this ambient independent set gives a finite spine enumerating the empty predicate. Pair that spine with the vacuous inclusion $(\lambda z.\ \bot)(z) \to K(z)$ to obtain

$$\mathsf{FinSetSubset}_{E,\sim_E}(\lambda z.\ \bot, K).$$

The defining equivalence for $\mathsf{Ind}_{M|K}$ then gives $\mathsf{Ind}_{M|K}(\lambda z.\ \bot)$.

For hereditary closure, assume $\mathsf{Ind}_{M|K}(I)$ and $\mathsf{FinSetSubset}_{E,\sim_E}(J, I)$. The defining equivalence gives $\mathsf{Ind}_M(I)$ and $\mathsf{FinSetSubset}_{E,\sim_E}(I, K)$. The ambient hereditary row gives $\mathsf{Ind}_M(J)$. Unpack the two subset rows: the row from J to I supplies a finite spine enumerating J and an inclusion $J(z) \to I(z)$, while the row from I to K supplies an inclusion $I(z) \to K(z)$. Composing the inclusions and keeping the finite spine for J gives

$$\mathsf{FinSetSubset}_{E,\sim_E}(J, K).$$

The defining equivalence repacks these two facts as $\mathsf{Ind}_{M|K}(J)$.

For exchange, assume $\mathsf{Ind}_{M|K}(I)$, $\mathsf{Ind}_{M|K}(J)$, and $\mathsf{FinSetCardLt}_{E,\sim_E}(I, J)$. The defining equivalence gives ambient independence of both I and J, plus subset rows

$$\mathsf{FinSetSubset}_{E,\sim_E}(I, K) \qquad \text{and} \qquad \mathsf{FinSetSubset}_{E,\sim_E}(J, K).$$

Apply the ambient exchange row to obtain histories x and I^+ such that

$$J(x) \wedge \neg I(x) \wedge \mathsf{FinSetInsert}_{E,\sim_E}(I, x, I^+) \wedge \mathsf{Ind}_M(I^+).$$

Unpacking the subset row from J to K gives the inclusion $\forall z : \mathsf{BHist}.\ J(z) \to K(z)$; applying this inclusion at x to the witness $J(x)$ gives $K(x)$. Unpack the insertion row: it supplies a finite spine enumerating I^+ and the pointwise law

$$I^+(z) \iff I(z) \vee (E(z) \wedge z \sim_E x).$$

To prove $\mathsf{FinSetSubset}_{E,\sim_E}(I^+, K)$, keep the finite spine from the insertion row. If $I^+(z)$ holds, the displayed pointwise law gives either $I(z)$ or $E(z) \wedge z \sim_E x$. In the first case, the subset row from I to K gives $K(z)$. In the second case, apply Definition 202.6 to the finite spine ks, the witness $K(x)$, the carried witness $E(z)$, and the comparison $z \sim_E x$; this gives $K(z)$. Thus I^+ is a finite subset of K.

Pair $\mathsf{Ind}_M(I^+)$ with this subset row and apply the defining equivalence for $\mathsf{Ind}_{M|K}$ to obtain $\mathsf{Ind}_{M|K}(I^+)$. The selected x, the predicate I^+, the witnesses $J(x)$, $\neg I(x)$, $K(x)$, the insertion row, and the restricted independence of I^+ are exactly the restricted exchange row. □

Lemma 202.8 (FinSetUp subset rows compose over one ground)**.** *Fix a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. For predicates* $A, B, C : \mathsf{BHist} \to \mathsf{Prop}$*, if*

$$\mathsf{FinSetSubset}_{E,\sim_E}(A, B) \qquad \textit{and} \qquad \mathsf{FinSetSubset}_{E,\sim_E}(B, C)$$

hold, then

$$\mathsf{FinSetSubset}_{E,\sim_E}(A, C)$$

holds.

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinsetSubset row composition`

Proof. Unpack Definition 202.1. The first subset row supplies a finite spine as with

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(as, A)$$

and an inclusion $A(z) \to B(z)$ for every $z : \mathsf{BHist}$. The second subset row supplies an inclusion $B(z) \to C(z)$ for every $z : \mathsf{BHist}$. Keep the same finite spine as as the enumeration witness for A. For the inclusion row, fix z and assume $A(z)$. The first inclusion gives $B(z)$, and the second gives $C(z)$. This is exactly $\mathsf{FinSetSubset}_{E,\sim_E}(A, C)$. □

Theorem 202.9 (Matroid restrictions compose to direct restriction)**.** *Fix a* $\mathsf{Matroid}^{\uparrow}$ *independent-set certificate over a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$ *and naming certificate* $\mathsf{cert} : \mathsf{NameCert}(E, \sim_E)$*. Let* Ind_M *be its independent-family row, with fields* finite_M*,* empty_M*, and* $\mathsf{hereditary}_M$ *as in Definition 202.2. Assume also the ambient exchange row*

$$\mathsf{MatroidExchangeRow}_{E,\sim_E,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M).$$

Let $K, L : \mathsf{BHist} \to \mathsf{Prop}$ *satisfy*

$$\mathsf{FinSetSubset}_{E,\sim_E}(K, E) \qquad \textit{and} \qquad \mathsf{FinSetSubset}_{E,\sim_E}(L, K).$$

Let $\mathsf{Ind}_{M|K}$ *and* $\mathsf{Ind}_{(M|K)|L}$ *be predicate families such that*

$$\forall I.\ \mathsf{Ind}_{M|K}(I) \iff \mathsf{Ind}_M(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, K)$$

and

$$\forall I.\ \mathsf{Ind}_{(M|K)|L}(I) \iff \mathsf{Ind}_{M|K}(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, L).$$

Then, for every predicate $I : \mathsf{BHist} \to \mathsf{Prop}$*,*

$$\mathsf{Ind}_{(M|K)|L}(I) \iff \mathsf{Ind}_M(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, L),$$

and the iterated restriction satisfies the direct restricted row package

$$\mathsf{MatroidRestrictionRows}_{E,\sim_E,L,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M, \mathsf{Ind}_{(M|K)|L}).$$

Proof. By Definition 202.8, the displayed subset rows for $L \subseteq K$ and $K \subseteq E$ compose to

$$\mathsf{FinSetSubset}_{E,\sim_E}(L, E).$$

Fix $I : \mathsf{BHist} \to \mathsf{Prop}$. For the forward implication, assume $\mathsf{Ind}_{(M|K)|L}(I)$. The second displayed defining equivalence gives

$$\mathsf{Ind}_{M|K}(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, L).$$

Applying the first displayed defining equivalence to the first component gives $\mathsf{Ind}_M(I)$. Pairing this ambient independence with the retained subset row $\mathsf{FinSetSubset}_{E,\sim_E}(I, L)$ gives the direct restriction condition.

For the reverse implication, assume

$$\mathsf{Ind}_M(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, L).$$

The subset row $\mathsf{FinSetSubset}_{E,\sim_E}(I, L)$ composes with $\mathsf{FinSetSubset}_{E,\sim_E}(L, K)$ by Definition 202.8, giving

$$\mathsf{FinSetSubset}_{E,\sim_E}(I, K).$$

The first defining equivalence therefore gives $\mathsf{Ind}_{M|K}(I)$ from $\mathsf{Ind}_M(I)$ and this subset row. Pairing $\mathsf{Ind}_{M|K}(I)$ with the original subset row $\mathsf{FinSetSubset}_{E,\sim_E}(I, L)$ and applying the second defining equivalence gives $\mathsf{Ind}_{(M|K)|L}(I)$.

Thus the displayed direct equivalence holds for every I. Apply Definition 202.7 with L as the finite restriction predicate, using the composed row $\mathsf{FinSetSubset}_{E,\sim_E}(L, E)$ and the direct equivalence just proved. The result is exactly

$$\mathsf{MatroidRestrictionRows}_{E,\sim_E,L,\mathsf{FinSetCardLt}}(\mathsf{Ind}_M, \mathsf{Ind}_{(M|K)|L}).$$

□

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidRestrictions compose direct restriction`

Lemma 202.10 (FinSetUp intersection projects to the right factor)**.** *Fix a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. For predicates* $I, K, J : \mathsf{BHist} \to \mathsf{Prop}$*, if*

$$\mathsf{FinSetIntersection}_{E,\sim_E}(J; I, K)$$

holds, then

$$\mathsf{FinSetSubset}_{E,\sim_E}(J, K)$$

holds. This is the finite-set form of the right lower-bound role for intersection over the common $\mathsf{FinSet}^{\uparrow}$ *ground, read from the exact membership row in Definition 202.1.*

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidFinsetIntersection right subset`

Proof. Unpack Definition 202.1. The intersection hypothesis supplies a finite spine zs with

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(zs, J)$$

and a pointwise equivalence

$$\forall z : \mathsf{BHist}.\ J(z) \iff I(z) \wedge K(z).$$

To prove $\mathsf{FinSetSubset}_{E,\sim_E}(J, K)$, keep the same finite spine zs as the finite-set witness for J. It remains only to prove the inclusion row. Fix $z : \mathsf{BHist}$ and assume $J(z)$. The forward half of the displayed pointwise equivalence gives $I(z) \wedge K(z)$, and projecting its second component gives $K(z)$. Thus every member of J is a member of K, with the finite-spine witness already supplied. □

Theorem 202.11 (Matroid restriction to the empty subset is singleton)**.** *Fix a* $\mathsf{Matroid}^{\uparrow}$ *independent-set certificate over a ground predicate* $E : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_E$*. Let* Ind_M *be its independent-family row, with fields* finite_M*,* empty_M*, and* $\mathsf{hereditary}_M$ *as in Definition 202.2. Let* $\mathsf{Ind}_{M|\varnothing}$ *be the restriction family defined by the empty restriction predicate:*

$$\forall I.\ \mathsf{Ind}_{M|\varnothing}(I) \iff \mathsf{Ind}_M(I) \wedge \mathsf{FinSetSubset}_{E,\sim_E}(I, \lambda z.\ \bot).$$

Then, for every predicate $I : \mathsf{BHist} \to \mathsf{Prop}$*,*

$$\mathsf{Ind}_{M|\varnothing}(I) \iff \forall z : \mathsf{BHist}.\ I(z) \iff \bot.$$

Thus the empty restriction accepts exactly the pointwise empty independent predicate.

Lean4 ✓ `BEDC.Derived.MatroidUp.MatroidRestriction empty singleton independent family`

Proof. Fix $I : \mathsf{BHist} \to \mathsf{Prop}$. For the forward implication, assume $\mathsf{Ind}_{M|\varnothing}(I)$. The displayed defining equivalence gives

$$\mathsf{FinSetSubset}_{E,\sim_E}(I, \lambda z.\ \bot).$$

Unpacking Definition 202.1, this subset row contains the inclusion

$$\forall z : \mathsf{BHist}.\ I(z) \to \bot.$$

For each z, this inclusion is the forward half of $I(z) \iff \bot$, while the reverse half follows from elimination of $\bot$. Hence I is pointwise empty.

For the reverse implication, assume

$$\forall z : \mathsf{BHist}.\ I(z) \iff \bot.$$

The ambient empty row gives $\mathsf{Ind}_M(\lambda z.\ \bot)$. Applying finite_M to this row chooses a finite spine xs with

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs, \lambda z.\ \bot).$$

Use this same finite spine as the ledger for I. By Definition 112.17, the enumeration row for the empty predicate says that, for every z,

$$\mathsf{EnumMem}_{E,\sim_E}(z, xs) \iff \bot.$$

Combining this equivalence with the assumed pointwise equivalence $I(z) \iff \bot$ gives

$$\mathsf{FinSetSpineEnumerates}_{E,\sim_E}(xs, I).$$

The assumed pointwise equivalence also gives the inclusion $I(z) \to \bot$ for every z. Therefore, by Definition 202.1, the same finite spine xs proves

$$\mathsf{FinSetSubset}_{E,\sim_E}(I, \lambda z.\ \bot).$$

The hereditary row $\mathsf{hereditary}_M$ applies to the ambient empty independent set and this finite-subset row, yielding $\mathsf{Ind}_M(I)$. Pairing $\mathsf{Ind}_M(I)$ with the displayed subset row and applying the defining equivalence for $\mathsf{Ind}_{M|\varnothing}$ gives

$$\mathsf{Ind}_{M|\varnothing}(I).$$

The two implications prove the claimed equivalence. □

Theorem 202.12 (Matroid finite bridge package)**.** *The displayed* $\mathsf{FinSet}^{\uparrow}$ *subset and intersection rows together with the* $\mathsf{Lattice}^{\uparrow}$ *dependency rows export the finite* $\mathsf{Matroid}^{\uparrow}$ *public interface as a paper-level bridge package. The bridge source is the finite ground row, the independent-set certificate of Definition 202.2, the restriction rows, the intersection rows, and the empty-restriction singleton row; its target reads the same finite matroid interface through subset, intersection, hereditary, and exchange data, with no host matroid oracle or off-ledger independence predicate.*

Proof. Start with the finite subset and intersection rows. The left and right subset theorems, the intersection-preservation theorems, and the subset-composition theorem expose the public finite-set operations only through $\mathsf{FinSet}^{\uparrow}$ rows over the displayed ground predicate.

Read the dependency rows through $\mathsf{Lattice}^{\uparrow}$. The restriction and composed-restriction theorems keep every retained-ground and empty-restriction read inside the same finite dependency surface, while the classifier-saturation row keeps the independent-family classifier tied to those displayed rows.

The bridge package is therefore the tuple of source, target, encode, decode, theorem-preservation, and no-host-leak data supplied by those finite rows. Encoding sends an accepted $\mathsf{Matroid}^{\uparrow}$ packet to its displayed finite ground, subset, intersection, hereditary, exchange, and restriction rows; decoding repacks the same rows as the public finite matroid interface. The preservation and no-host-leak fields are exactly the nonescape clauses already stated in the carrier and restriction proofs: a consumer receives only finite $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Lattice}^{\uparrow}$ dependency data, not a host independence oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{Matroid}^{\uparrow}}$.

Constructive story: A $\mathsf{Matroid}^{\uparrow}$ packet is generated from a finite ground-set row, displayed subset rows over that ground, independent-set certificate rows, restriction rows, intersection rows, singleton empty-restriction rows, componentwise hsame transport, finite Cont readbacks, package provenance, and local naming. The classifier reads independence only through the finite subset and certificate surface; restriction is generated by projecting to retained ground elements and repacking the same independence witnesses over the restricted carrier. The ledger story is finite matroid accounting: intersections, restrictions, composed restrictions, and the empty restriction are all produced from explicit FinSetUp rows and their transport routes, not from a host matroid object or off-ledger independence oracle.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Matroid}^{\uparrow}}, \mathsf{bridgedClosed})$

Scope closed: This chapter exports the public carrier route from finite independence, subset, intersection, restriction, classifier-saturation, and exchange rows to the downstream $\mathsf{Matroid}^{\uparrow}$ restriction surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Matroid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim host matroid equality, infinite ground sets, unlisted rank or closure operators, or independence data outside the displayed finite certificate rows.

Upgrade path: Bridge closure requires Lean verification and a standard bridge for the public finite-independence carrier route exposed here.

Chapter 203

A Concrete Naming Certificate for $\mathsf{Ramsey}^{\uparrow}$

The Ramsey-theory interface $\mathsf{Ramsey}^{\uparrow}$ packages, for every finite colouring of the r-element subsets of a sufficiently large finite set, a monochromatic subset of bounded size. It depends on $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112) and $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118).

203.1 Carrier definition

Definition 203.1 (Ramsey monochrome witness carrier)**.** *The Ramsey monochrome witness carrier is the finite* BHist *packet whose readable rows are the* $\mathsf{FinSet}^{\uparrow}$ *vertex spine, the displayed subset spine, the colour endpoint rows indexed by that subset spine, the chosen witness subset row, the component* hsame *transports, the finite* Cont *lookup routes from displayed subsets to colour endpoints, and the shared* Pkg *provenance for the packet. The carrier accepts a downstream witness only when each row is read through these finite packet coordinates and through the dependency certificates* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *where those projections are requested.*

Lean4 (definition) `BEDC.Derived.RamseyUp.RamseyMonochromeWitnessCarrier`

203.2 The certificate

Theorem 203.2 (Ramsey colouring carrier obligation)**.** *A carried* $\mathsf{Ramsey}^{\uparrow}$ *colouring row over a displayed* $\mathsf{FinSet}^{\uparrow}$ *source is a finite* BHist *packet consisting of the certified finite vertex spine, the certified r-subset spine read through* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$*, one colour endpoint row for each displayed subset, finite* Cont *routes tying the subset rows to their colour endpoints, and shared* Pkg *provenance for the table. The carrier accepts only these finite colouring rows.*

The carrier does not read a host finite-set enumeration, a host function from subsets to colours, or an ambient combinatorial universe. All subset and colour reads pass through the displayed BHist *rows and the finite ledger.*

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier obligation surface`

Proof. Project the $\mathsf{Ramsey}^{\uparrow}$ row to the finite vertex spine supplied by $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ and to the displayed subset spine built over it. The colour table is then a finite family of endpoint rows indexed by that subset spine, with each lookup routed by a visible Cont row and packaged by the same Pkg provenance.

Since the packet has no coordinate for a host enumeration or a host colouring function, a consumer can only read the listed BHist rows. Repacking those rows gives the stated carrier surface. □

Theorem 203.3 (Ramsey public projection package)**.** *The checked rows Definition 203.2, Definition 203.4, Definition 203.5, and Definition 203.6 assemble a public projection package for* $\mathsf{Ramsey}^{\uparrow}$*. FinSetUp and GraphUp consumers receive only the vertex spine, subset spine, colour endpoint rows, witness rows, component* hsame *transports, finite* Cont *lookup routes, and shared* Pkg *provenance displayed by the finite* BHist *ledger.*

The package exports no host finite-set enumeration, no host colouring function, no graph equality primitive, and no ambient graph object.

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier public projection package`

Proof. Project first through the carrier rows of Definition 203.2. This identifies the public source available to a FinSetUp or GraphUp consumer with the displayed vertex spine, subset spine, colour table, witness rows, finite lookup routes, transport rows, and provenance rows.

Apply the classifier stability row of Definition 203.4 to transport the monochrome classifier across component hsame witnesses. The transported classifier still reads the same finite colour endpoints through the same Cont routes.

Use the witness scope row Definition 203.5 for every downstream dependency read. It restricts finset and graph consumers to the displayed dependency certificates and prevents any read from leaving the finite packet.

Finally combine ledger exactness from Definition 203.6 with the downstream boundary of Definition 203.8. Together they give both inclusions: all exported public rows are the listed ledger rows, and the excluded host enumeration, colouring function, graph equality primitive, and ambient graph object have no carrier coordinate to project. □

Theorem 203.4 (Ramsey monochrome classifier stability)**.** *Let* $\mathcal{C}$ *and* $\mathcal{C}'$ *be carried* $\mathsf{Ramsey}^{\uparrow}$ *colouring rows over the same displayed* $\mathsf{FinSet}^{\uparrow}$ *source. If their vertex spines, subset spines, colour endpoint rows, finite* Cont *routes, and* Pkg *provenance are related componentwise by* hsame*, then the monochrome-subset classifier transports from* $\mathcal{C}$ *to* $\mathcal{C}'$*. In particular, any displayed witness subset whose* r*-subsets have one colour in* $\mathcal{C}$ *is carried to a displayed witness subset with the same classified colour in* $\mathcal{C}'$*.*

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier classifier stability`

Proof. Begin with the subset witness. The $\mathsf{FinSet}^{\uparrow}$ classifier transports the finite vertex entries and the displayed subset spine, so every r-subset read from the witness in $\mathcal{C}$ has a corresponding transported row in $\mathcal{C}'$.

Next transport the colour endpoints along the displayed hsame rows. The finite Cont routes preserve which table entry is read by each subset, and the shared Pkg provenance keeps the transported reads inside the same colouring packet.

The monochrome condition is pointwise equality through the colour classifier on the transported endpoint rows. Thus all transported r-subsets retain the same classified colour, and the witness subset remains monochrome without choosing a host subset representative. □

Theorem 203.5 (Ramsey monochrome witness scope)**.** *Every monochrome witness consumed downstream from the* $\mathsf{Ramsey}^{\uparrow}$ *surface is read only through the carrier of Definition 203.1, together with the* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *projections already exposed by the displayed packet.*

No downstream clause reads a host subset iterator, a host colouring function, or an ambient graph object outside those packet rows.

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier monochrome witness scope`

Proof. First project the witness to the finite packet of Definition 203.1. This gives the vertex spine, subset spine, colour endpoint rows, chosen witness subset, component hsame transports, finite lookup routes, and shared provenance as explicit BHist rows.

Next apply Definition 203.2. It identifies the accepted colouring surface with the same finite packet, so every colour read used by the witness is routed through a displayed Cont lookup rather than through a host colouring function.

The witness classifier is stable under the transported rows by Definition 203.4. The finite ledger exactness theorem then says that the readable rows are exactly the displayed carrier, lookup, transport, and provenance rows, with any finset or graph consumer receiving only the corresponding $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ or $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ projection.

Thus the downstream witness has no remaining coordinate from which a host subset iterator, host colouring function, or ambient graph object could be read. □

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier witness scope`

Theorem 203.6 (Ramsey finite ledger exactness)**.** *For every accepted* $\mathsf{Ramsey}^{\uparrow}$ *colouring row, the readable ledger consists exactly of the* $\mathsf{FinSet}^{\uparrow}$ *vertex spine, the finite* r*-subset spine, the colour endpoint table, the displayed monochrome witness subset rows, the component* hsame *transports, the finite* Cont *lookup routes, and the shared* Pkg *provenance. It exposes no host finite-set enumeration, host subset iterator, off-ledger colour function, or graph object beyond the displayed dependency rows.*

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier finite ledger exactness`

Proof. Use Definition 203.2 to reduce any consumer read to the carrier packet. Each listed ledger row is a coordinate of that packet or a finite lookup route attached to it.

For the reverse containment, inspect the carrier surface: every vertex, subset, colour, witness, transport, lookup, and provenance row appears in the finite ledger. The packet has no constructor for host enumeration, host subset iteration, or off-ledger colouring functions.

If a graph consumer is present, it can only receive the dependency rows already certified by $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ and the displayed finite ledger. No extra graph object is generated by the Ramsey ledger itself. □

Theorem 203.7 (Ramsey NameCert obligation assembly)**.** *The public* $\mathsf{NameCert}_{\mathsf{Ramsey}^{\uparrow}}$ *obligation surface is assembled from the finite colouring carrier of Definition 203.2, the monochrome-subset classifier stability of Definition 203.4, the ledger exactness of Definition 203.6, and the dependency certificates* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *when graph-shaped consumers read the same finite rows. The assembled surface binds carrier, classifier, stability, exactness, and source coverage over* BHist*,* hsame*,* Cont*, and* Pkg*.*

No clause of the assembled surface treats host finite-set enumeration, host function equality, or an ambient graph object as a primitive.

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier namecert obligation assembly`

Proof. Use the carried colouring row as the source predicate. Its finite vertex spine, subset spine, colour table, lookup routes, and provenance are exactly the carrier rows isolated in Definition 203.2.

Use the monochrome-subset classifier as the comparison predicate. Stability under transported vertex, subset, colour, route, and provenance rows is precisely Definition 203.4.

The exactness and source-coverage clauses are supplied by Definition 203.6. The $\mathsf{FinSet}^{\uparrow}$ dependency accounts for all finite spines, and the $\mathsf{Graph}^{\uparrow}$ dependency is only a consumer boundary for graph-shaped readings of those rows, so the assembled certificate remains a finite BHist ledger. □

Theorem 203.8 (Ramsey downstream boundary)**.** *Every graph, finset, or combinatorial consumer of the* $\mathsf{Ramsey}^{\uparrow}$ *surface reads only the assembled* $\mathsf{NameCert}_{\mathsf{Ramsey}^{\uparrow}}$ *rows of Definition 203.7: the displayed* $\mathsf{FinSet}^{\uparrow}$ *vertex spine, the finite subset spine, the colour endpoint table, the monochrome witness rows, the* hsame *transports, the finite* Cont *lookup routes, and the shared* Pkg *provenance. A consumer may project these rows into* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *or* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *interfaces, but it cannot read a host finite-set enumeration, host subset iterator, host colouring function, or graph equality primitive from* $\mathsf{Ramsey}^{\uparrow}$ *itself.*

Lean4 ✓ `BEDC.Derived.RamseyUp.RamseyColouringCarrier downstream boundary`

Proof. First reduce the consumer read to the assembled certificate surface by Definition 203.7. This fixes the carrier, classifier, stability, ledger, and source-coverage rows over the finite BHist packet.

For a finset consumer, project the vertex and subset spines through $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$. The projection preserves the displayed hsame transports and finite lookup routes, and it does not expose a host enumeration.

For a graph-shaped consumer, use $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ only on rows already present in the finite ledger. The graph interface receives carried vertex, edge, or incidence rows as projections, while the Ramsey table and witness remain the same Cont-routed finite packet.

Thus every downstream boundary is a projection of the public Ramsey ledger, not an additional primitive surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Ramsey}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Ramsey}^{\uparrow}$ packet is generated from a finite vertex spine, colour endpoint rows, monochrome-witness rows, dependency projections to FinSetUp and GraphUp, componentwise hsame transport, finite Cont lookup routes, package provenance, and local naming. The classifier reads monochromaticity only through the displayed finite colouring ledger and the transported colour endpoints for each listed subset. The ledger story is finite Ramsey projection: public consumers receive vertex, edge, subset, colour, witness, route, and provenance rows already present in the BHist packet, while host enumeration, off-ledger colouring functions, and graph objects beyond the displayed dependency certificates are not generated.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Ramsey}^{\uparrow}}$, obligation)

Scope closed: This chapter binds the Ramsey witness carrier and the colouring-carrier, public-projection, classifier-stability, witness-scope, finite-ledger-exactness, NameCert-assembly, and downstream-boundary theorems for $\mathsf{Ramsey}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Ramsey}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim host finite-set enumeration, host colouring functions, off-ledger subset iteration, or graph data beyond the displayed FinSetUp and GraphUp projections.

Upgrade path: Scoped closure requires binding the finite Ramsey ledger obligations to FinSetUp, GraphUp, BHist, hsame, Cont, Pkg, and NameCert scope.

Chapter 204

A Concrete Naming Certificate for $\mathsf{TopGroup}^{\uparrow}$

The topological-group interface $\mathsf{TopGroup}^{\uparrow}$ packages a group carrier with a topology making multiplication and inversion continuous. It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88).

204.1 Carrier definition

Definition 204.1 (TopGroupUp BHist carrier)**.** *A* $\mathsf{TopGroup}^{\uparrow}$ BHist *carrier is a finite packet*

$$\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$$

over the displayed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source rows. The row G is the group carrier and classifier surface. The row T is the topology carrier containing the public open-neighbourhood rows. The row m is the multiplication graph row, with endpoints read in G and neighbourhoods read in T. The row i is the inversion graph row with the same carrier and neighbourhood interface. The row n records the open neighbourhood transport rows used by multiplication and inversion. The row λ is the finite* Cont *ledger for continuity transport, and ρ is the* Pkg *provenance row for the group, topology, operation, neighbourhood, continuity, and ledger entries.*

The classifier between two carriers compares the G, T, m, i, n, and ρ rows by hsame *and compares the continuation ledgers by* Cont*. No external topological group object is part of the carrier.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupBHistCarrier`

204.2 The certificate

Theorem 204.2 (TopGroupUp multiplication continuity carrier)**.** *Let $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ be a* $\mathsf{TopGroup}^{\uparrow}$ BHist *carrier. If x and y are carried by the group row G, and if an open-neighbourhood row of T is displayed around the product endpoint $m(x, y)$, then the multiplication graph row keeps $m(x, y)$ in the topology carrier. Transporting x, y, the product row, and the target neighbourhood by the classifier uses only the displayed* hsame *rows and the finite* Cont *continuity ledger.*

Proof. Project the group row, topology row, multiplication graph row, neighbourhood row, continuation ledger, and provenance row from Definition 204.1. These projections name both operation endpoints and the target open-neighbourhood row.

The classifier transports the two group endpoints and the multiplication graph componentwise by hsame. It also transports the relevant open-neighbourhood row in T and records the continuity step in the finite Cont ledger.

Repack the transported multiplication graph with the same topology carrier and provenance row. The resulting product endpoint is still read through T, so the multiplication continuity row remains inside the $\mathsf{TopGroup}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage multiplication continuity carrier`

Theorem 204.3 (TopGroupUp inversion continuity carrier)**.** *Let $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ be a $\mathsf{TopGroup}^{\uparrow}$ BHist carrier. If x is carried by the group row G, and if an open-neighbourhood row of T is displayed around the inverse endpoint $i(x)$, then the inversion graph row keeps $i(x)$ in the topology carrier. Transporting x, its inverse row, and the target neighbourhood by the classifier uses only the displayed* hsame *rows and the same finite* Cont *continuity ledger.*

Proof. Project the inversion graph row from Definition 204.1, together with the group carrier, topology carrier, neighbourhood row, continuation ledger, and provenance row.

The endpoint transport is componentwise: x is transported inside G, the inverse graph row is transported by hsame, and the target neighbourhood is transported inside T. The continuity entry is recorded in the finite Cont ledger already carried by $\mathcal{G}$.

Repack these rows with the unchanged source certificates. Since the carrier contains no additional neighbourhood object or external topology, the inverse endpoint remains in the displayed $\mathsf{TopGroup}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage inversion continuity carrier`

Theorem 204.4 (TopGroupUp NameCert obligation surface)**.** *The public $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ obligation surface consists exactly of the* BHist *carrier of Definition 204.1, its componentwise classifier, the inherited $\mathsf{Group}^{\uparrow}$ operation rows, the $\mathsf{Topology}^{\uparrow}$ open-neighbourhood rows, the multiplication and inversion continuity rows, and the finite* Cont/Pkg *ledger scope. It exports no group-topology data beyond those displayed BEDC rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage namecert obligation surface`

Proof. Unpack the carrier packet. The projections expose the group source, topology source, multiplication graph, inversion graph, open-neighbourhood transport, continuation ledger, and provenance row.

Apply Definition 204.2 for the product continuity row and Definition 204.3 for the inverse continuity row. Both results use only the componentwise classifier and the finite Cont ledger.

Repack the projected rows as the public certificate surface. The source rows are the displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ certificates, while every operation and continuity row is carried over those sources, so the obligation surface is exhausted by the listed finite packet. □

Theorem 204.5 (TopGroupUp Cont ledger continuity exhaustion)**.** *Let $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ be a $\mathsf{TopGroup}^{\uparrow}$ packet with displayed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source rows. The finite* Cont *ledger λ exhausts exactly the multiplication and inversion continuity rows over those displayed sources: every public continuity obligation of $\mathcal{G}$ is one of the product rows of Definition 204.2 or one of the inverse rows of Definition 204.3, and those two families are carried by λ with no extra topology object.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage continuity ledger scope`

Proof. Project the ledger coordinate λ from the carrier packet of Definition 204.1. The same projection exposes the displayed $\mathsf{Group}^{\uparrow}$ source, $\mathsf{Topology}^{\uparrow}$ source, multiplication row, inversion row, neighbourhood row, and provenance row.

Apply Definition 204.2 to the product row and Definition 204.3 to the inverse row. These are the only operation-continuity rows exported by the public surface of Definition 204.4.

Repack the two projected families with the unchanged source certificates. Because the carrier has no further topological component, the ledger is exhausted by those two finite Cont families. □

Theorem 204.6 (TopGroupUp classifier continuity transport)**.** *For classifier-related* $\mathsf{TopGroup}^{\uparrow}$ *packets* $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ *and* $\mathcal{G}' = (G', T', m', i', n', \lambda', \rho')$*, the public classifier transports* m*,* i*,* n*,* λ*, and* ρ *componentwise by the carried* hsame *rows and the finite* Cont *ledger. In particular, continuity transport for the product and inverse rows is exactly the transport supplied by Definition 204.5.*

Proof. Unpack the two packets with Definition 204.1. The component projections give the group carrier, topology carrier, multiplication graph, inversion graph, neighbourhood row, ledger row, and provenance row on both sides.

Transport the operation rows and neighbourhood rows componentwise by the displayed hsame witnesses. The product and inverse continuity entries then transport through the same finite ledger by Definition 204.5.

The provenance row ρ is read only as the displayed source package, so it transports with the same source certificates. Repacking these transported coordinates gives the classifier-related $\mathsf{TopGroup}^{\uparrow}$ packet without adding an external continuity predicate. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage classifier continuity transport`

Theorem 204.7 (TopGroupUp public certificate boundary)**.** *The complete public* $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ *boundary is formed by Definition 204.1, Definition 204.2, Definition 204.3, Definition 204.5, and Definition 204.6. No host topological group and no external continuity predicate is read outside those displayed BEDC rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage public certificate boundary`

Proof. Combine the carrier definition with the two continuity carrier rows. The carrier supplies the finite packet, while the multiplication and inversion theorems identify the only operation-continuity rows over the displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ sources.

The ledger exhaustion theorem records that the finite Cont coordinate contains exactly those rows. The classifier transport theorem records that the same boundary is stable under componentwise hsame transport.

Thus the public certificate boundary is precisely the listed carrier, continuity, ledger, transport, and provenance data. Since each component is projected from the displayed packet, no host topological-group structure is available to the certificate. □

Theorem 204.8 (TopGroupUp source-coupled carrier scope)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *packet projects one shared* BHist *group source row, topology source row, multiplication row, inversion row,* Cont *ledger, and* Pkg *provenance row. These rows are the source of the public carrier and classifier scope; no independent host topological-group object is available outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source coupled carrier scope`

Proof. Unpack the accepted packet by Definition 204.1. The projection exposes the single group source G, topology source T, multiplication row m, inversion row i, neighbourhood row n, ledger λ, and provenance row ρ.

Apply Definition 204.2 and Definition 204.3 to read the operation rows over those same sources. Apply Definition 204.5 to read the finite Cont scope and Definition 204.6 for the carried classifier transport.

Finally use Definition 204.7. Since that boundary is exhausted by the projected rows and their provenance, there is no second carrier from which an external topological group could be read. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source rows empty coupled`

Theorem 204.9 (TopGroupUp operation-continuity ledger scope)**.** *For every accepted* $\mathsf{TopGroup}^{\uparrow}$ *packet, the product and inverse continuity rows are exactly the two operation-neighbourhood transport families carried by the same finite* Cont *ledger. Under endpoint* hsame *transport, both families remain in that ledger and no additional continuity family is introduced.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage operation continuity ledger scope`

Proof. Project the packet using Definition 204.1. The continuity coordinate is the single finite ledger λ, and its operation entries are attached to the displayed multiplication and inversion rows.

Apply Definition 204.5. It identifies the public continuity obligations with the product rows of Definition 204.2 and the inverse rows of Definition 204.3.

For transport, apply Definition 204.6. The endpoint hsame witnesses move the operation and neighbourhood endpoints componentwise, while the product and inverse continuity entries remain the same two ledger families. □

Theorem 204.10 (TopGroupUp public NameCert readback exactness)**.** *The public* $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ *readback is exhausted by the carrier, classifier, product continuity rows, inverse continuity rows, finite* Cont *ledger, and* Pkg *provenance. Reading a public certificate therefore returns exactly those displayed BEDC rows and no further topological-group datum.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage public namecert readback exactness`

Proof. Start from the public boundary theorem Definition 204.7. Its listed sources are the carrier definition, the product and inverse continuity carrier rows, the ledger exhaustion row, and the classifier transport row.

Use Definition 204.1 to identify the carrier, classifier, finite Cont ledger, and Pkg provenance coordinates. Use Definition 204.2 and Definition 204.3 for the two operation continuity families, and Definition 204.5 to see that the ledger contains no further public continuity obligations.

Thus every public readback component is one of the named coordinates or one of the two operation-continuity families. The classifier transport theorem Definition 204.6 gives stability of exactly those rows under hsame, so the readback has no additional datum. □

Theorem 204.11 (TopGroupUp product neighbourhood transport)**.** *Let* $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ *be a* $\mathsf{TopGroup}^{\uparrow}$ *packet with displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source rows. For carried* BHist *endpoints* x, x', y, y', z, z'*, suppose* $m(x, y, z)$ *is the displayed product row,* z *lies in a neighbourhood row of* T*, and* $\mathsf{hsame}(x, x')$*,* $\mathsf{hsame}(y, y')$*, and* $\mathsf{hsame}(z, z')$ *are the carried endpoint transports. Then the product neighbourhood row transports to the endpoint* z' *using only the componentwise* hsame *rows, the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *operation source, the* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *neighbourhood source, and the finite* Cont *ledger carried by* $\mathcal{G}$*.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage product neighborhood transport`

Proof. Project the packet $\mathcal{G}$. The projections expose the carried BHist endpoints, the $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ multiplication row, the $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ neighbourhood row around the product endpoint, and the relevant entry of the finite Cont ledger.

Transport the two input endpoints and the product endpoint componentwise by the displayed hsame rows. The group source transports the multiplication graph row, while the topology source transports the target neighbourhood row inside T.

Repack the transported multiplication graph with the transported neighbourhood row and the same continuity ledger entry. No outside topological-group datum is introduced, so the transported row remains a $\mathsf{TopGroup}^\uparrow$ product neighbourhood row over $\mathcal{G}$. □

Theorem 204.12 (TopGroupUp inverse neighbourhood transport)**.** *Let $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ be a $\mathsf{TopGroup}^\uparrow$ packet with displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ source rows. For carried BHist endpoints x, x', y, y', suppose $i(x, y)$ is the displayed inverse row, y lies in a neighbourhood row of T, and $\mathsf{hsame}(x, x')$ and $\mathsf{hsame}(y, y')$ are the carried endpoint transports. Then the inverse neighbourhood row transports to the endpoint y' using only the componentwise hsame rows, the $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ inverse source, the $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ neighbourhood source, and the finite Cont ledger carried by $\mathcal{G}$.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage inverse neighborhood transport`

Proof. Unpack $\mathcal{G}$ to obtain the group source, topology source, inverse graph row, neighbourhood row around the inverse endpoint, and the matching continuity entry in the Cont ledger.

The endpoint transport is componentwise. The $\mathsf{hsame}(x, x')$ row transports the input, the $\mathsf{hsame}(y, y')$ row transports the inverse endpoint, the $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ source carries the inverse graph across those rows, and the $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ source carries the neighbourhood row inside T.

Repack the transported inverse graph and transported neighbourhood with the same finite continuity entry. The result is still a row of the displayed $\mathsf{TopGroup}^\uparrow$ packet, because every field used came from $\mathcal{G}$. □

Theorem 204.13 (TopGroupUp continuity ledger coverage)**.** *For a $\mathsf{TopGroup}^\uparrow$ packet $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ with displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ sources, the finite Cont ledger covers exactly the product and inverse neighbourhood transport rows over carried BHist endpoints. Its rows are the transported operation neighbourhood entries of Definition 204.11, the transported inverse neighbourhood entries of Definition 204.12, and the source provenance rows already displayed by Definition 204.4. It exports no additional topological-group structure.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage continuity ledger coverage`

Proof. Start from the public obligation surface. Apply Definition 204.4 to project the BHist carrier, componentwise classifier, group source, topology source, multiplication and inversion rows, and finite $\mathsf{Cont}/\mathsf{Pkg}$ ledger.

The product rows are covered by Definition 204.11. Each entry uses the displayed carried endpoints, the componentwise hsame transport rows, and the same $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ sources.

The inverse rows are covered by Definition 204.12. These entries use the same packet projections and therefore sit in the same finite Cont ledger.

Repack the two transport families with the provenance rows of $\mathcal{G}$. Since the ledger contains only those product and inverse continuity rows, its coverage is exactly the displayed $\mathsf{TopGroup}^\uparrow$ continuity surface. □

Definition 204.14 (TopGroupUp source-coupled continuity packet)**.** *The public source-coupled continuity packet for* $\mathsf{TopGroup}^{\uparrow}$ *is the finite* BHist *packet consisting of the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row, the product graph row, the inverse graph row, the source and target neighbourhood rows, the finite* Cont *continuity ledger, and the* Pkg *provenance row connecting those displayed sources. A packet is accepted only when these rows are the rows projected by the* $\mathsf{TopGroup}^{\uparrow}$ *carrier* $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$.

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupSourceCoupledContinuityPacket`

Theorem 204.15 (TopGroupUp source-coupled continuity boundary)**.** *Every accepted source-coupled continuity packet projects exactly its* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, its* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source row, the product and inverse graph rows, the componentwise* hsame *endpoint transport rows, the finite* Cont *continuity rows, and the* Pkg *provenance row. It exports no ambient topological-group object, no external topology carrier equality, and no group operation beyond the carried graph rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source coupled continuity boundary`

Proof. Project the accepted packet of Definition 204.14. Its displayed coordinates are precisely the two source certificates, the product and inverse graph rows, the neighbourhood rows, the finite continuity ledger, and the provenance row.

The componentwise hsame rows are the endpoint transports used by those coordinates. They are already accounted for by the product transport theorem Definition 204.11 and the inverse transport theorem Definition 204.12.

Completeness is the reverse packaging. Given exactly the projected rows, repack them as the finite BHist packet of the definition and attach the same Cont and Pkg ledgers. No extra ambient object is available, because every public projection is one of the listed rows. □

Theorem 204.16 (TopGroupUp continuity consumer exhaustion)**.** *A downstream consumer of a* $\mathsf{TopGroup}^{\uparrow}$ *source-coupled continuity packet receives only the product-neighbourhood transport family of Definition 204.11, the inverse-neighbourhood transport family of Definition 204.12, and their finite ledger coverage in Definition 204.13. The consumer cannot read a topological-group object, an external continuity predicate, or any row outside the displayed* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *packet.*

Proof. Start with the boundary theorem Definition 204.15. It restricts the consumer input to the displayed source certificates, endpoint transports, finite continuity rows, and provenance row.

Apply Definition 204.13. The finite Cont ledger is exhausted by the product-neighbourhood and inverse-neighbourhood transport rows, with provenance supplied by the same source certificates.

Finally use the two transport theorems named in the statement to repack the consumer surface. Since every available row is either one of those transport families or its BHist provenance, the consumer has no access to an unlisted ambient continuity datum. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage continuity consumer exhaustion`

Theorem 204.17 (TopGroupUp root source compatibility)**.** *For every accepted* $\mathsf{TopGroup}^{\uparrow}$ *source-coupled continuity packet, the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row and displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row share the same* BHist *carrier endpoints of Definition 204.1. Classifier transport between the two source rows is supplied by the componentwise* hsame *endpoint rows together with the finite* Cont *continuity ledger.*

Proof. Unpack the carrier by Definition 204.1. The group source certificate and topology source certificate are projections of the same BHist packet, so their carrier endpoints are not independent objects; they are the shared endpoints carried by $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$.

The componentwise hsame rows are the classifier transports attached to those shared endpoints. The product and inverse transport theorems, Definition 204.11 and Definition 204.12, use those rows as the source and target classifiers for their continuity statements.

Apply Definition 204.13. The finite Cont ledger records exactly the product and inverse continuity rows over the shared carrier endpoints. Repack these rows to get the stated source compatibility. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root source compatibility`

Theorem 204.18 (TopGroupUp product-inverse obligation triad)**.** *The public* $\mathsf{TopGroup}^{\uparrow}$ *operation surface consists exactly of three obligations: carrier compatibility between* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$*, product-neighbourhood transport, and inverse-neighbourhood transport. These are the rows supplied by Definition 204.17, Definition 204.11, Definition 204.12, and Definition 204.13.*

Proof. Project the accepted packet through Definition 204.1. The carrier row gives the common source surface for the group and topology certificates, while the operation graph rows expose only multiplication and inversion.

Definition 204.17 identifies the carrier compatibility obligation. The product-neighbourhood transport obligation is exactly Definition 204.11, and the inverse-neighbourhood transport obligation is exactly Definition 204.12.

It remains to show that the list is exhaustive. Apply Definition 204.13; the Cont ledger contains only the product and inverse continuity rows. Together with the shared source carrier row, these three obligations are the whole public operation surface. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage product inverse obligation triad`

Theorem 204.19 (TopGroupUp public consumer minimality)**.** *Every downstream consumer of a* $\mathsf{TopGroup}^{\uparrow}$ *packet receives only the displayed source certificate rows, the product and inverse operation rows, the finite* Cont *rows, and the* Pkg *provenance. It receives no ambient topological-group object and no order, topology, or operation row outside the rows listed by Definition 204.18.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage public consumer minimality`

Proof. Start from the source-coupled packet of Definition 204.14 and unpack its carrier by Definition 204.1. Its projections are the two source certificates, the product and inverse graph rows, the finite Cont ledger, and the Pkg provenance row.

Definition 204.18 repacks the operation rows as carrier compatibility, product-neighbourhood transport, and inverse-neighbourhood transport. No fourth operation row is present in the finite ledger.

Finally apply Definition 204.16. It states that downstream readback is exhausted by the same transport families, their BHist provenance, hsame endpoint rows, Cont rows, and source NameCert rows. Thus the public consumer receives exactly the listed data and no ambient topological-group object. □

Definition 204.20 (TopGroupUp BHist continuity obligation triple)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ BHist *continuity obligation triple over the carrier of Definition 204.1 consists of exactly three public obligations:*

1. *the shared* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source compatibility over the same carried endpoints, classifier rows, and* Pkg *provenance;*
2. *the product* Cont*-neighbourhood transport row, where the two input endpoints, product endpoint, target neighbourhood, and componentwise* hsame *rows are read from the same carrier packet;*
3. *the inverse* Cont*-neighbourhood transport row, where the input endpoint, inverse endpoint, target neighbourhood, and componentwise* hsame *rows are read from the same carrier packet.*

No fourth operation obligation and no host topological-group primitive is part of this triple.

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupBHistContinuityObligationTriple`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupBHistContinuityObligationTriple sound`

Theorem 204.21 (TopGroupUp source projection NameCert compatibility)**.** *For every accepted* $\mathsf{TopGroup}^\uparrow$ *packet, the* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *source row and the* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source row project from the same* BHist *carrier endpoints, classifier rows, and* Pkg *provenance of Definition 204.1. Their compatibility is the first component of Definition 204.20.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source projection namecert compatibility`

Proof. Project the packet by Definition 204.1. The group row, topology row, operation rows, finite Cont ledger, and Pkg provenance are all coordinates of one carrier.

Apply Definition 204.17. It identifies the displayed group and topology source certificates as projections over the same carried endpoints, with classifier transport supplied by the componentwise hsame rows and the finite Cont ledger.

Finally compare this projected compatibility row with Definition 204.20. The source compatibility component is exactly the shared BHist endpoint, classifier, and provenance row just obtained, so no independent source certificate remains outside the packet. □

Theorem 204.22 (TopGroupUp Cont ledger operation exhaustion)**.** *For every accepted* $\mathsf{TopGroup}^\uparrow$ *packet, the finite* Cont *ledger contains exactly the product-neighbourhood continuity rows and the inverse-neighbourhood continuity rows listed by Definition 204.20. It contains no ambient topological-group operation row, no external continuity predicate, and no operation other than the carried product and inverse rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage cont ledger operation exhaustion`

Proof. Start with Definition 204.13. The ledger coverage theorem says that every public continuity row is either a product neighbourhood transport row or an inverse neighbourhood transport row over the carried BHist endpoints.

The product family is the second component of Definition 204.20, and it is supplied by Definition 204.11. The inverse family is the third component of the same definition, and it is supplied by Definition 204.12.

Apply Definition 204.19. A downstream consumer can read only the displayed source rows, operation rows, finite Cont rows, and Pkg provenance. Thus the ledger is exhausted by the two operation-continuity families and contains no ambient topological-group row. □

Theorem 204.23 (TopGroupUp downstream root unblock surface)**.** *The downstream* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *root-unblock surface is the* BHist *continuity obligation triple of Definition 204.20, together with the source projection compatibility of Definition 204.21, the finite* Cont *ledger exhaustion of Definition 204.22, and the product-inverse obligation triad of Definition 204.18. This is the complete public obligation surface consumed by downstream chapters.*

Proof. First take the triple of Definition 204.20. It names the shared source compatibility obligation, the product neighbourhood transport obligation, and the inverse neighbourhood transport obligation over the carrier of Definition 204.1.

Apply Definition 204.21 to bind the source obligation to the same BHist endpoints, classifier rows, and Pkg provenance. Apply Definition 204.22 to bind the two operation-continuity obligations to the finite Cont ledger and exclude any ambient operation row.

Finally use Definition 204.18. It repacks exactly the same carrier compatibility, product transport, and inverse transport obligations as the public operation surface. Therefore downstream chapters receive precisely this source-compatible, ledger-exhaustive $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage downstream root unblock surface`

Theorem 204.24 (TopGroupUp shared source-fiber product carrier)**.** *For every accepted* $\mathsf{TopGroup}^{\uparrow}$ *source-coupled continuity packet, the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row and the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row share one* BHist *carrier packet through Definition 204.1. The product graph, inverse graph, neighbourhood row, finite* Cont *ledger, and* Pkg *provenance are projections of that same packet, so neither source row supplies an independent carrier outside the* $\mathsf{TopGroup}^{\uparrow}$ *boundary.*

Proof. Unpack the accepted packet by Definition 204.1. Its coordinates are the shared group row G, topology row T, product graph m, inverse graph i, neighbourhood row n, finite ledger λ, and provenance row ρ.

Apply Definition 204.17. It identifies the two source certificates as projections over the same carried endpoints, with classifier transport supplied by the componentwise hsame rows and the finite Cont ledger.

Finally apply Definition 204.19. A downstream consumer receives only those displayed source, operation, ledger, and provenance rows. Hence the shared packet is the whole source fiber for product readback and no external carrier can be read. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source fiber product carrier`

Theorem 204.25 (TopGroupUp product Cont ledger soundness)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *packet, the product-neighbourhood continuity row is sound exactly when its two input endpoints, product endpoint, target neighbourhood row, componentwise* hsame *transports, and finite* Cont *ledger entry are all projected from the shared packet of Definition 204.1. The row has no additional topological-group source and no external continuity predicate.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage product cont ledger soundness`

Proof. For soundness, start with the projected shared packet. Theorem Definition 204.24 supplies the common $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source rows, together with the product graph, neighbourhood row, hsame transports, and finite Cont ledger.

Apply Definition 204.11. The product row transports the two input endpoints and the product endpoint through exactly those componentwise hsame rows, while the neighbourhood row is carried by the displayed topology source and the continuity step is recorded in the finite ledger.

For the converse direction, assume a public product-continuity row is accepted. Apply Definition 204.13; every product row in the public surface is one of the entries covered by the finite Cont ledger of the same packet.

The public consumer theorem Definition 204.19 then removes any remaining ambient reading. The accepted product row is therefore sound precisely as the projection of the shared packet's endpoints, neighbourhood row, hsame transports, and finite ledger entry. □

Theorem 204.26 (TopGroupUp obligation inventory)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *certificate has exactly the following public obligations: the shared carrier obligation for the displayed* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *rows, the classifier-transport obligation over their componentwise* hsame *rows, the multiplication-continuity obligation, the inversion-continuity obligation, the source-compatibility obligation, and the finite ledger obligation for the displayed* Cont *rows.*

No ambient topological-group object, no third operation-continuity family, and no operation row outside the carried product and inverse graphs belongs to the certificate boundary.

Proof. First bind the source side by Definition 204.21. It gives the shared $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ carrier endpoints, classifier rows, componentwise hsame transports, and one Pkg provenance packet.

Next bind the operation side by Definition 204.18. The operation surface has exactly carrier compatibility, product-neighbourhood transport, and inverse-neighbourhood transport.

Finally use Definition 204.22. The finite Cont ledger is exhausted by the product and inverse continuity families, and Definition 204.19 excludes every ambient row outside the displayed source, operation, ledger, and provenance surface. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation inventory bridge`

Theorem 204.27 (TopGroupUp carrier-classifier obligation)**.** *Carrier and classifier transport for a* $\mathsf{TopGroup}^{\uparrow}$ *certificate is exactly componentwise* hsame *over the shared* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *source rows, together with one* Pkg *provenance packet. The transport does not split into separate group and topology carriers and does not import any external topological-group classifier.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage carrier classifier obligation`

Proof. Unpack the accepted packet through Definition 204.1. The displayed group source, topology source, product graph, inverse graph, neighbourhood row, finite ledger, and provenance row are coordinates of one carried BHist packet.

Apply Definition 204.17. It identifies the $\mathsf{Group}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ source certificates as projections over the same carried endpoints, with classifier transport supplied by the componentwise hsame rows.

The same projection is recorded by Definition 204.21. Hence the carrier-classifier obligation is precisely componentwise hsame over the two shared source rows plus the single Pkg provenance packet, with no independent classifier source remaining. □

Theorem 204.28 (TopGroupUp inverse Cont ledger soundness)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *packet, the inverse-neighbourhood continuity row is sound exactly when its input endpoint, inverse endpoint, target neighbourhood row, componentwise* hsame *transports, and finite* Cont *ledger entry are all projected from the shared packet of Definition 204.1. The row has no additional topology carrier, group inverse source, or external continuity predicate.*

Proof. For soundness, unpack the shared packet using Definition 204.24. The projected rows include the group source, topology source, inverse graph, target neighbourhood row, endpoint hsame transports, and the finite Cont ledger.

Apply Definition 204.12. The input endpoint and inverse endpoint move componentwise along the displayed hsame rows, while the topology source carries the target neighbourhood and the ledger records the continuity step.

For the converse direction, start from an accepted public inverse-continuity row. By Definition 204.13, every such row is one of the inverse-neighbourhood entries in the same finite Cont ledger.

Apply Definition 204.19. Since the consumer surface exposes only the displayed source certificates, inverse graph, endpoint transports, finite ledger, and provenance row, the inverse-continuity row is sound exactly through those packet projections and no outside datum is present. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage inverse cont ledger soundness`

Theorem 204.29 (TopGroupUp source compatibility obligation)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface reads its displayed* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *source rows from the same* BHist *carrier, the same carrier-aware* hsame *endpoint classifier, and the same* Pkg *provenance packet of Definition 204.1.*

Proof. Apply Definition 204.24. It projects the group source, topology source, product graph, inverse graph, neighbourhood row, endpoint transports, finite continuity ledger, and provenance row from one shared carrier packet.

Use Definition 204.19 to read this packet from the public side. A downstream consumer sees only the displayed source certificates, hsame transports, finite Cont ledger, and Pkg provenance, so the two source rows cannot be separated into different carriers. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source compatibility obligation`

Theorem 204.30 (TopGroupUp operation continuity obligation)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface, the operation-continuity obligations exported by the carrier are exactly multiplication continuity and inversion continuity. They are the rows covered by the product and inverse neighbourhood-transport theorems, and no further operation-continuity row is exported by Definition 204.1.*

Proof. First apply Definition 204.25. It identifies the multiplication-continuity row with the product graph, target neighbourhood, endpoint hsame transports, and finite Cont ledger entry projected from the shared carrier.

Next apply Definition 204.28. It identifies the inversion-continuity row with the inverse graph, target neighbourhood, endpoint hsame transports, and its own finite Cont ledger entry.

Finally use Definition 204.19. Since the public surface exposes only these operation rows together with the source certificates, classifier transports, ledger, and provenance packet, there is no third operation-continuity obligation available to a consumer. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage operation continuity witness obligation`

Theorem 204.31 (TopGroupUp continuity ledger exactness obligation)**.** *The finite* Cont *ledger of an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface exhausts the continuity obligations for product and inverse rows: every accepted product-neighbourhood or inverse-neighbourhood continuity read is one of those two ledger families, and each listed ledger entry is sound through the shared* BHist *packet.*

Proof. Start with Definition 204.13. It states that the public continuity surface is covered by the finite ledger entries for product and inverse neighbourhood rows.

For soundness of the product entries, apply Definition 204.25; for soundness of the inverse entries, apply Definition 204.28. Both proofs project the relevant endpoints, neighbourhoods, hsame transports, and ledger rows from the same carrier packet.

Then use Definition 204.19 to exclude ambient continuity predicates or unstated operation rows. The finite Cont ledger is therefore exact for the product and inverse continuity obligations exported by $\mathsf{TopGroup}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage ledger exactness obligation`

Theorem 204.32 (TopGroupUp source-pair carrier obligation)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface binds its* $\mathsf{Group}^{\uparrow}$ *source row and its* $\mathsf{Topology}^{\uparrow}$ *source row to the same* BHist *carrier of Definition 204.1. The root obligation is therefore a source-pair obligation over one carried packet, not a pair of unrelated carrier certificates.*

Proof. Apply Definition 204.24. It identifies the group and topology certificates as projections from the shared packet of Definition 204.1.

The same theorem also records that product, inverse, neighbourhood, ledger, and provenance rows are packet projections. Hence a root consumer can read both source rows only through the common BHist carrier. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source pair carrier obligation`

Theorem 204.33 (TopGroupUp product continuity obligation)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface, product continuity is exactly the endpoint transport supplied by the packet's* hsame *rows together with the finite* Cont *ledger entry for multiplication. No extra product-continuity predicate is present outside the carrier of Definition 204.1.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage product continuity obligation`

Proof. First use Definition 204.24 to place the product graph, neighbourhood row, endpoint transports, and finite ledger over the shared source-pair carrier.

Then apply Definition 204.25. Its two directions say that a product-continuity row is sound exactly when it is this packet projection through the hsame endpoints and the finite Cont ledger. □

Theorem 204.34 (TopGroupUp inverse continuity obligation)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root surface, inverse continuity is the unary endpoint-transport row carried by the packet's* hsame *data and the finite* Cont *ledger entry for inversion. The obligation has no separate inverse carrier, topology source, or ambient continuity witness.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage inverse continuity obligation`

Proof. Apply Definition 204.32 to bind the unary inverse row to the same BHist carrier as the group and topology source certificates.

Now use Definition 204.28. It states that the accepted inverse-continuity row is sound precisely through the input endpoint, inverse endpoint, target neighbourhood, hsame transports, and finite Cont ledger entry projected from that packet. □

Theorem 204.35 (TopGroupUp operation ledger obligation)**.** *The multiplication and inverse continuity obligations of a* $\mathsf{TopGroup}^{\uparrow}$ *certificate are exactly the two operation-continuity ledger families supplied by the finite* Cont *ledger: product-neighbourhood transport and inverse-neighbourhood transport. There is no third operation row in the public certificate boundary.*

Proof. Use Definition 204.13. Every public continuity row over the accepted packet is covered by the finite Cont ledger and is one of the product or inverse neighbourhood transport rows.

Definition 204.22 identifies those two families with the product and inverse components of Definition 204.20. It also excludes ambient continuity predicates and any operation outside the carried product and inverse graphs.

Finally apply Definition 204.18. The public operation surface repacks the same two continuity families with the shared carrier compatibility row. Therefore the operation ledger obligation has exactly two operation families and no third operation row. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage operation ledger consumer boundary`

Theorem 204.36 (TopGroupUp fixed left-translation continuity row)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *packet and a carried group endpoint g in its source-fiber product, the row L_g defined on carried endpoints by the displayed product graph,*

$$L_g(h) = z \quad \textit{exactly when the packet displays } m(g, h, z),$$

is a sound continuity row through the same product graph, target neighbourhood row, componentwise hsame *transports, and finite* Cont *ledger entry used for multiplication. The fixed endpoint g does not create a separate unary operation source, a separate topology carrier, or an external translation-continuity predicate.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage fixed left translation continuity row`

Proof. Unpack the accepted packet by Definition 204.24. The shared carrier supplies the displayed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ source row, the displayed $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source row, the product graph m, the neighbourhood row, the componentwise hsame classifier rows, the finite Cont ledger, and the Pkg provenance row.

Fix a carried group endpoint g. For a carried endpoint h and a displayed product endpoint z with $m(g, h, z)$, regard the left-translation read $L_g(h) = z$ as the product-neighbourhood row whose first input has been fixed to the carried endpoint g. This specialization keeps the same product graph and the same target neighbourhood row; it merely restricts the two-input product read to the fiber over g.

Apply Definition 204.11 to the product row $m(g, h, z)$. The fixed endpoint g, the variable endpoint h, and the product endpoint z are transported only by the displayed componentwise hsame rows. The target neighbourhood is transported inside the displayed topology source, and the continuity step is recorded in the finite Cont ledger already carried by the packet.

Now apply Definition 204.25. It identifies soundness of the product-continuity row exactly with the packet projections just listed: the input endpoints, product endpoint, target neighbourhood, componentwise hsame transports, and finite ledger entry. Since the left-translation row is this product row with the first input fixed to g, its continuity soundness is inherited from the same ledger entry and no additional operation source is introduced.

For exhaustiveness, use Definition 204.13. Every accepted product-neighbourhood continuity read is already one of the product entries in the finite ledger. Finally, Definition 204.19 rules out any ambient translation predicate or detached topological-group carrier. Hence the fixed left-translation row is sound exactly through the displayed product graph, target neighbourhood, hsame transports, and finite Cont ledger of the accepted $\mathsf{TopGroup}^{\uparrow}$ packet. □

Definition 204.37 (TopGroupUp root shared source packet)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *root shared source packet is the projection of the carrier of Definition 204.1 consisting of one displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, one displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source row, the shared* BHist *carrier endpoints, the componentwise* hsame *classifier rows, the product and inverse operation rows, the finite* Cont *ledger, and the* Pkg *provenance row. A downstream root consumer reads the two source certificates only through this packet.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootSharedSourcePacket`

Theorem 204.38 (TopGroupUp root shared carrier obligation)**.** *For every accepted* TopGroup$^{\uparrow}$ *root shared source packet, the* Group$^{\uparrow}$ *and* Topology$^{\uparrow}$ *source rows share the same* BHist *carrier of Definition 204.1. Their endpoint classifiers are the same componentwise* hsame *rows, their operation-continuity transport is recorded in the same finite* Cont *ledger, and their provenance is the same* Pkg *row. No root consumer can split these fields into two independent source packets.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage shared source rows`

Proof. Begin with the shared packet of Definition 204.37. Its coordinates are exactly the group source, topology source, carried endpoints, classifier rows, finite continuity ledger, and provenance row projected from the TopGroup$^{\uparrow}$ carrier.

Apply Definition 204.24. It identifies the two source certificate rows, the product graph, inverse graph, neighbourhood row, finite Cont ledger, and Pkg provenance as projections of one BHist packet.

Finally apply Definition 204.19. Since a public consumer receives only those source, operation, ledger, classifier, and provenance rows, there is no available second carrier or second provenance packet. □

Theorem 204.39 (TopGroupUp root continuity classifier stability)**.** *For an accepted* TopGroup$^{\uparrow}$ *root shared source packet, product continuity and inverse continuity transport under the same componentwise classifier. The product row transports its two input endpoints, product endpoint, target neighbourhood, and* Cont *entry through the carried* hsame *rows, and the inverse row transports its input endpoint, inverse endpoint, target neighbourhood, and* Cont *entry through the same classifier packet.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage continuity classifier stability`

Proof. Project the root shared source packet. The projection exposes the common hsame endpoint classifier rows and the finite Cont ledger attached to the product and inverse operation rows.

Use Definition 204.25 for the product continuity row. It makes the row sound exactly through the two input endpoints, product endpoint, target neighbourhood row, componentwise hsame transports, and finite ledger entry projected from the shared packet.

Use Definition 204.28 for the inverse continuity row. It gives the same classifier form for the input endpoint, inverse endpoint, target neighbourhood row, and ledger entry.

The two rows therefore transport under one classifier packet: their endpoint lists differ by arity, but their carrier, hsame source, Cont ledger, and Pkg provenance are the shared rows of the root packet. □

Theorem 204.40 (TopGroupUp root ledger exactness obligation)**.** *For an accepted* TopGroup$^{\uparrow}$ *root shared source packet, the finite* Cont *ledger exhausts exactly the product-continuity rows and inverse-continuity rows. Every accepted continuity read is one of those two families, each listed entry is sound through the shared* BHist *packet, and no host topological-group continuity primitive is available outside the ledger.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root ledger semantic exactness`

Proof. Start from the shared carrier obligation of Definition 204.38. It binds both source certificates, the classifier rows, the continuity ledger, and the provenance row to the same BHist packet.

Apply Definition 204.31. It states that the finite Cont ledger covers every accepted product-neighbourhood and inverse-neighbourhood continuity read, and that the two listed families are sound through the shared packet.

Apply Definition 204.19. The public root consumer receives no continuity row beyond the displayed source certificates, operation rows, classifier rows, finite ledger, and provenance. Hence the ledger is exact for precisely the product and inverse continuity obligations. □

Theorem 204.41 (TopGroupUp root public obligation inventory)**.** *An accepted* $\mathsf{TopGroup}^{\uparrow}$ *root shared source packet exposes exactly the* $\mathsf{Group}^{\uparrow}$ *source row, the* $\mathsf{Topology}^{\uparrow}$ *source row, the product row, the inverse row, the neighbourhood row, the componentwise* hsame *classifier packet, the finite* Cont *ledger, and the shared* Pkg *provenance. This public inventory contains no ambient topological-group primitive and no second carrier or provenance source outside the displayed packet.*

```
Lean4 ✓  BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root public obligation inventory
  ↪ variant  BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage public obligation inventory
  ↪ variant  BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation inventory
```

Proof. Start with Definition 204.38. It fixes the $\mathsf{Group}^{\uparrow}$ source row, the $\mathsf{Topology}^{\uparrow}$ source row, the common BHist carrier endpoints, the componentwise classifier rows, the operation rows, the finite ledger, and the shared provenance as projections of one root packet.

Next apply Definition 204.39. The product and inverse continuity reads are transported only through the displayed componentwise hsame classifier packet and the same finite Cont ledger.

Finally use Definition 204.40. It exhausts the public continuity surface by the product and inverse ledger families and rules out any host topological-group continuity primitive beyond the rows already listed. The inventory is therefore exactly the displayed source, operation, neighbourhood, classifier, ledger, and provenance data. □

Theorem 204.42 (TopGroupUp root shared source exhaustion)**.** *Every downstream public read of the* $\mathsf{TopGroup}^{\uparrow}$ *root surface factors through the public obligation inventory of Definition 204.41. Conversely, each row of that inventory repacks into the same public root surface as a read of the accepted root shared source packet.*

```
Lean4 ✓  BEDC.Derived.TopGroupUp.TopGroupRootSharedSourcePacket public inventory exhaustion
```

Proof. For the forward direction, apply Definition 204.19. A downstream public consumer receives only the source certificate rows, operation rows, classifier rows, continuity ledger rows, and provenance row exposed by the root shared source packet.

Then apply Definition 204.41. Those are precisely the rows in the public inventory: the two source rows, product row, inverse row, neighbourhood row, componentwise classifier packet, finite Cont ledger, and shared Pkg provenance. Hence every downstream public read factors through the inventory.

For the reverse direction, read each inventory coordinate as the corresponding projection of the same accepted root shared source packet. The shared carrier obligation supplies the common packet, the classifier-stability theorem supplies the componentwise transport rows, and the ledger-exactness theorem supplies the finite product and inverse continuity rows.

Repacking those projections gives exactly the public root surface consumed by Definition 204.19. No additional carrier, classifier, ledger entry, or ambient topological-group primitive is introduced in the repacking. □

Theorem 204.43 (TopGroupUp obligation carrier-classifier surface)**.** *The obligation surface for the* $\mathsf{TopGroup}^{\uparrow}$ *carrier and classifier consists of one finite* BHist *packet carrying the displayed*

$\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *row, the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *row, the shared carrier endpoints, the componentwise* hsame *classifier rows, and the single* Pkg *provenance row. These rows are the carrier-classifier obligations visible to a public consumer, and no independent group carrier, topology carrier, or host topological-group classifier is part of the surface.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupObligation carrier classifier surface`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation carrier classifier surface`
↪ variant `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation surface certificate`

Proof. Start from Definition 204.37. Its displayed coordinates are the $\mathsf{Group}^\uparrow$ source, the $\mathsf{Topology}^\uparrow$ source, the shared BHist carrier endpoints, the componentwise classifier rows, and the provenance packet.

Apply Definition 204.38. It binds those coordinates to one accepted packet and rules out a split into two independent source packets.

Finally use Definition 204.27. The public carrier-classifier obligation is exactly componentwise hsame over the two displayed source rows together with the same Pkg provenance row. □

Theorem 204.44 (TopGroupUp obligation continuity ledger surface)**.** *The continuity-ledger obligation surface for* $\mathsf{TopGroup}^\uparrow$ *consists exactly of the product-continuity family, the inverse-continuity family, and the finite* Cont *ledger rows connecting them to the displayed* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *carrier rows. The surface contains no continuity row outside the carried product and inverse operation graphs.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation continuity ledger surface`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootObligationContinuityLedgerSurface exists`

Proof. Project the accepted shared packet through Definition 204.39. The product and inverse rows transport through the same componentwise classifier packet and the same finite Cont ledger.

Use Definition 204.40. It exhausts public continuity reads by the two operation families and makes each listed entry sound through the shared BHist packet.

The operation boundary is then Definition 204.35. It identifies the ledger surface with product-neighbourhood transport and inverse-neighbourhood transport, excluding any third operation-continuity family. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupObligationSurface continuity ledger inventory`

Theorem 204.45 (TopGroupUp obligation public consumer exhaustion)**.** *Every public consumer of the* $\mathsf{TopGroup}^\uparrow$ *obligation surface factors through the finite inventory consisting of the* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *row, the* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *row, the shared* BHist *carrier endpoints, the product row, the inverse row, the neighbourhood row, the componentwise* hsame *classifier packet, the finite* Cont *ledger, and the shared* Pkg *provenance. Conversely, each listed row repacks into the same public consumer surface.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupObligationSurface public consumer exhaustion`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage obligation public consumer exhaustion`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupUp StdBridge`

Proof. Apply Definition 204.41. It lists exactly the two source rows, operation rows, neighbourhood row, classifier packet, finite ledger, and provenance row exposed by an accepted root shared source packet.

Then apply Definition 204.42. Every downstream public read factors through that inventory, and each inventory row reconstructs the public root surface of the same accepted packet.

Combining the carrier-classifier surface with the continuity-ledger surface therefore gives the complete obligation surface visible to public consumers, with no ambient topological-group primitive outside the listed rows. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupObligation public consumer surface certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$.

Constructive story: A topological-group obligation packet is assembled by placing a certified $\mathsf{Group}^\uparrow$ source and a certified $\mathsf{Topology}^\uparrow$ source into one finite BHist surface, then displaying shared carrier endpoints, product rows, inverse rows, neighbourhood rows, continuity ledgers, and provenance. The carrier accepts only packets whose group, topology, product, inverse, neighbourhood, classifier, finite Cont, and Pkg rows are all present; an independent group carrier, independent topology carrier, host topological-group classifier, or third operation-continuity family is outside the obligation surface. Two accepted packets classify together by componentwise hsame on the group and topology source rows, operation rows, neighbourhood rows, and provenance, with the finite continuation ledger carrying product-continuity and inverse-continuity comparisons. The source specification is the paired group/topology certificate boundary, the pattern specification records product continuity, inverse continuity, carrier-classifier compatibility, and public-consumer exhaustion, and the ledger keeps product-neighbourhood and inverse-neighbourhood transport visible. The five name-certificate obligations are now bound at the obligation surface: an accepted packet inhabits the carrier, the inherited source classifiers give reflexivity, symmetry, and transitivity componentwise, and the public-consumer exhaustion theorem repacks classified replacements into the same finite inventory. A packet history h lies in the signature gap of a $\mathsf{TopGroup}^\uparrow$ token p when the source-domain accepts the displayed group/topology rows, the signature relation pairs the packet with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the product and inverse continuity ledgers are the operation families encoded by p. The chapter registers this finite continuity-obligation surface under the name $\mathsf{TopGroup}^\uparrow$; it introduces no standard bridge and no complete topological-group theory beyond the displayed public obligation rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$, maturePackageClosed)

Scope closed: The obligation scope binds the displayed $\mathsf{Group}^\uparrow$, $\mathsf{Topology}^\uparrow$, BHist, hsame, Cont, and Pkg rows to the carrier, classifier, product-continuity, inverse-continuity, and public-consumer obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.TopGroupUp.TopGroupObligation\_public\_consumer\_surface\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim a standard bridge or a complete topological-group theory beyond the displayed public obligation rows.

Upgrade path: No further upgrade – chapter at top of formal axis.

204.3 Root source-fiber obligation packet

Definition 204.46 (TopGroupUp root source-fiber obligation packet)**.** *The* $\mathsf{TopGroup}^\uparrow$ *root source-fiber obligation packet is the shared* BHist *packet*

$$\mathcal{F} = (G, T, m, i, n, \kappa, \lambda, \rho)$$

read over the displayed $\mathsf{Group}^\uparrow$ *source row* G *and* $\mathsf{Topology}^\uparrow$ *source row* T*. The row* m *is the product operation source,* i *is the inverse operation source,* n *is the neighbourhood source used by both continuity families,* κ *is the componentwise* hsame *classifier packet on the carried source rows and operation endpoints,* λ *is the finite* Cont *ledger listing the product and inverse continuity entries, and* ρ *is the shared* Pkg *provenance row. A root consumer may read the* $\mathsf{Group}^\uparrow$ *and* $\mathsf{Topology}^\uparrow$ *sources only through this packet.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiberObligationPacket`

Theorem 204.47 (TopGroupUp root source-fiber carrier obligation)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber obligation packet carries the* $\mathsf{Group}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *source rows on one* BHist *carrier and one classifier packet. The product row, inverse row, neighbourhood row, componentwise* hsame *classifier, finite* Cont *ledger, and* Pkg *provenance are projections from that carrier, so neither source row supplies a separate root carrier.*

Proof. Project the packet of Definition 204.46. Its coordinates are the two source rows, operation rows, neighbourhood row, classifier packet, continuation ledger, and provenance row.

Apply Definition 204.24. It identifies the group source, topology source, product graph, inverse graph, neighbourhood row, finite Cont ledger, and Pkg provenance as projections from one shared BHist carrier.

Then apply Definition 204.38. The root surface reads the two source certificates through exactly this shared packet, with their endpoint classifiers supplied by the componentwise hsame rows.

Finally use Definition 204.19. A public root consumer has no field for a second carrier, second classifier packet, or detached provenance row, so the displayed source fiber is the whole carrier obligation. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root source fiber carrier obligation`

Theorem 204.48 (TopGroupUp root source-fiber classifier determinacy)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber obligation packet, every classifier read accepted at the root is determined by the shared* BHist *carrier, the componentwise* hsame *classifier packet, the product and inverse endpoint rows, the finite* Cont *ledger, and the* Pkg *provenance row. Conversely, those rows repack the same root classifier read, so no detached group classifier, topology classifier, or ambient topological-group classifier is present at the root boundary.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThresholdPacket source fiber classifier determinacy`

Proof. First apply Definition 204.47. It places the group source, topology source, operation endpoints, componentwise classifier packet, ledger, and provenance over one shared BHist carrier.

Next use Definition 204.49. The product and inverse endpoint reads are transported through that same classifier packet, and their continuity evidence is read only through the finite Cont entries listed in the packet.

Finally apply Definition 204.50. The ledger covers exactly the accepted product-continuity and inverse-continuity families. Repacking the shared carrier, componentwise hsame packet, endpoint rows, ledger, and Pkg provenance therefore reconstructs the same classifier read, while any classifier outside those rows has no field in the accepted source-fiber packet. □

Theorem 204.49 (TopGroupUp root source-fiber continuity obligation)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber obligation packet, product continuity and inverse continuity transport through the same classifier packet and the same finite* Cont *ledger. The product row uses the two input endpoints, product endpoint, neighbourhood row, componentwise* hsame *transport, and its ledger entry; the inverse row uses the input endpoint, inverse endpoint, neighbourhood row, componentwise* hsame *transport, and its ledger entry. No continuity row is detached from the shared source fiber.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber continuity obligation`

Proof. Start from the carrier obligation of Definition 204.47. It places the product row, inverse row, neighbourhood row, hsame classifier packet, finite ledger, and provenance over the same carried source rows.

Apply Definition 204.33. The product continuity obligation is exactly endpoint transport through the packet's hsame rows together with the displayed finite Cont entry.

Apply Definition 204.34. The inverse continuity obligation has the same form for the unary inverse endpoint and its finite ledger entry.

Use Definition 204.39 to compare the two operation families through one classifier packet. Since both continuity rows read the same source fiber and ledger, classifier transport cannot introduce an ambient topological-group continuity predicate. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source fiber continuity obligation`

Theorem 204.50 (TopGroupUp root source-fiber ledger exactness)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber obligation packet, the finite* Cont *ledger exhausts exactly the product-continuity and inverse-continuity obligations. Every accepted operation-continuity read is one of those two families, and every listed ledger entry is sound through the shared* BHist *carrier, componentwise* hsame *classifier, and* Pkg *provenance of the packet.*

Proof. Begin with Definition 204.49. It binds both operation-continuity families to the same classifier packet and finite continuation ledger.

Apply Definition 204.31. It states that the finite Cont ledger covers every accepted product-neighbourhood and inverse-neighbourhood continuity read, with soundness through the shared carrier packet.

Use Definition 204.60 to rule out a third operation-continuity family at the root. The accepted root surface has only product continuity and inverse continuity over the displayed source pair.

Finally apply Definition 204.19. A public consumer receives no continuity primitive beyond the source rows, operation rows, classifier packet, finite ledger, and provenance, so the ledger is exact for precisely the two displayed obligation families. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber ledger exactness`

204.4 Root source-fiber export packet

Definition 204.51 (TopGroupUp root source-fiber export packet)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet is the public projection of Definition 204.46. It consists of the shared* BHist *source pair for* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$*, the product and inverse rows, the componentwise* hsame *classifier rows on the source and operation endpoints, the finite* Cont *ledger for the two continuity families, and the shared* Pkg *provenance. A downstream root consumer receives the* $\mathsf{TopGroup}^{\uparrow}$ *source fiber only through these rows.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiberExportPacket`

Theorem 204.52 (TopGroupUp root source-fiber export carrier)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet carries the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source rows over one shared* BHist *carrier. The product row, inverse row, componentwise* hsame *rows, finite* Cont *ledger, and* Pkg *provenance are projections from that carrier, so the export surface contains no detached group source, topology source, or ambient topological-group carrier.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export carrier`

Proof. Start with the carrier theorem Definition 204.47. It reads the two source rows and both operation rows from one root source-fiber obligation packet.

Apply Definition 204.50. The finite ledger and provenance rows are attached to that same source pair, so they cannot be exported as independent carrier data.

Finally compare with Definition 204.19. A public consumer receives only the displayed source certificate rows, operation rows, finite continuation rows, and provenance. This is precisely the carrier surface of Definition 204.51. □

Theorem 204.53 (TopGroupUp root source-fiber export continuity)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet, product continuity and inverse continuity are exported only as componentwise* hsame *transport through the shared source pair and the finite* Cont *ledger. The product and inverse rows have no continuity witness outside the displayed operation endpoints, classifier rows, ledger entries, and provenance.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export continuity`

Proof. Use Definition 204.49. It binds the product and inverse continuity rows to the same source-fiber carrier, componentwise classifier packet, and finite ledger.

Then apply Definition 204.47. The operation endpoints are already projections from the shared BHist source pair and cannot introduce a separate continuity carrier.

The public boundary is fixed by Definition 204.19. Since the consumer receives no ambient topological-group predicate, the two exported continuity rows are exactly the ledger-indexed product and inverse rows. □

Theorem 204.54 (TopGroupUp root source-fiber export ledger)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet, the finite* Cont *ledger is exact for the exported product and inverse continuity rows. Every exported operation-continuity read is one of those rows, and every listed ledger entry is sound through the shared* BHist *source pair, componentwise* hsame *classifier rows, and shared* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export ledger`

Proof. Begin with Definition 204.50. It states that the finite Cont ledger exhausts precisely the product-continuity and inverse-continuity obligation families over the root source fiber.

Next use Definition 204.49. This identifies the endpoints and classifier transports consumed by each listed ledger entry.

Finally apply Definition 204.19. A downstream consumer has no third operation-continuity family and no ledger row outside the exported source, operation, classifier, continuation, and provenance rows. □

Theorem 204.55 (TopGroupUp root source-fiber export common Cont ledger)**.** *In an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet, the product-continuity and inverse-continuity rows are indexed by the same finite* Cont *ledger and the same* Pkg *provenance over the shared* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source pair. Any downstream read of either continuity family factors through that common ledger, with no ambient topological-group object and no separate continuity carrier.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export common cont ledger`

Proof. Apply Definition 204.53. It binds both operation-continuity families to the displayed operation endpoints, classifier rows, finite Cont ledger, and provenance.

Next use Definition 204.54. The exported ledger is exact for the product and inverse continuity rows, so a downstream read of either family is one of its entries.

Finally, Definition 204.52 ties the ledger and provenance to the shared $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$–$\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source pair. This leaves no ambient topological-group carrier through which a second ledger could enter. □

Theorem 204.56 (TopGroupUp root source-fiber export consumer boundary)**.** *Every public consumer read of the* $\mathsf{TopGroup}^{\uparrow}$ *root source fiber factors through Definition 204.51. Conversely, each row of that export packet is a public row already forced by the shared* BHist *source pair for* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$*, the product and inverse rows, componentwise* hsame *transport, the finite* Cont *ledger, and shared* Pkg *provenance. No order row, topology row outside* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$*, operation row outside product and inverse, or ambient topological-group object is exported.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage export boundary certificate`

Proof. First assemble the export carrier by Definition 204.52. The source pair, operation rows, classifier packet, finite ledger, and provenance all live over one BHist source fiber.

Next attach the operational rows. Apply Definition 204.53 for the product and inverse continuity rows, and Definition 204.54 for the exact finite ledger that lists them.

For the forward direction, use Definition 204.19. It restricts every public consumer to the same source certificate rows, operation rows, finite continuation rows, and provenance, so the consumer read factors through the export packet.

For the reverse direction, each row of Definition 204.51 is already one of the public rows named by the carrier, continuity, and ledger theorems above. Thus the packet exports exactly the source-fiber surface and nothing outside it. □

Theorem 204.57 (TopGroupUp root source-fiber export exactness)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root source-fiber export packet is exact for public consumers: a consumer read factors through the packet if and only if it is one of the shared* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source rows, product rows, inverse rows, componentwise* hsame *rows, finite* Cont *ledger entries, or shared* Pkg *provenance rows. No order row, detached topology row, third operation, or ambient topological-group carrier is accepted by the export surface.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export exactness semantic certificate`
↪ variant `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiberExportPacket exactness`

Proof. First use Definition 204.52. It fixes the exported root source fiber over the shared $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$–$\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source rows, the product and inverse rows, componentwise hsame transport, and the shared package provenance.

Next apply Definition 204.53 and Definition 204.54. These theorems bind the product and inverse continuity rows to the displayed finite Cont ledger, so every accepted continuation read is one of the listed ledger entries.

The common ledger theorem, Definition 204.55, identifies the product and inverse rows as a single shared finite continuation ledger rather than two detached operation ledgers.

Finally apply Definition 204.56. The consumer boundary gives both directions of the factorization statement and excludes order rows, detached topology rows, any third operation, and any ambient topological-group carrier outside the exported packet. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootSourceFiber export exactness`

204.5 Root operation source packet

Definition 204.58 (TopGroupUp root operation-source packet)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *root operation-source packet is the shared* BHist *packet*

$$\mathcal{R} = (G, T, m, i, n, \lambda, \rho)$$

of Definition 204.1, read as a root-facing source for the operation rows. The coordinate G is the displayed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, T is the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row, m is the product row, i is the inverse row, n is the neighbourhood row used by both operation-continuity families, λ is the finite* Cont *ledger, and ρ is the* Pkg *provenance row tying the group, topology, operation, neighbourhood, classifier, and continuity entries to one packet.*

Two such packets are compared only by the componentwise hsame *rows already present in the carrier and by the displayed* Cont *ledger comparison. The packet contains no host topological-group object and no ambient continuity predicate outside the shown BEDC rows.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootOperationSourcePacket`

Theorem 204.59 (TopGroupUp root source-pair exactness)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root operation-source packet has exactly the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source pair. The product row, inverse row, neighbourhood row, finite* Cont *ledger, and* Pkg *provenance row are projections from the same* BHist *packet, so the root source cannot be read as two independent carrier certificates.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootOperationSourcePacket root source pair exactness`
Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootOperationSourcePacket source pair exactness`

Proof. Project the packet of Definition 204.58. Its coordinates are the source pair G, T together with the operation rows m, i, the neighbourhood row n, the ledger λ, and the provenance row ρ.

Apply Definition 204.24. It identifies the group source, topology source, product graph, inverse graph, neighbourhood row, finite Cont ledger, and Pkg provenance as projections from one shared BHist carrier.

Finally use Definition 204.38. A root consumer therefore receives the two source rows only through that shared packet, with no second carrier from which an independent source certificate could be assembled. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root source pair exactness`

Theorem 204.60 (TopGroupUp root operation-continuity exhaustion)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root operation-source packet, every operation-continuity row exported at the root is either a product-continuity row for m or an inverse-continuity row for i. Both families are carried by the displayed neighbourhood row n and finite* Cont *ledger λ, and there is no further operation-continuity predicate outside the packet.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootOperationSourcePacket operation continuity exhaustion`

Proof. Start with the exact source-pair packet supplied by Definition 204.59. It places the product row, inverse row, neighbourhood row, and ledger over the same displayed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ sources.

Apply Definition 204.30. It says that the exported operation-continuity obligations are exactly multiplication continuity and inversion continuity.

Apply Definition 204.31. The finite Cont ledger covers all accepted product-neighbourhood and inverse-neighbourhood continuity reads, and each listed entry is sound through the shared carrier packet. Thus the root operation-continuity surface is exhausted by those two families. □

Theorem 204.61 (TopGroupUp root classifier ledger stability)**.** *If two accepted* $\mathsf{TopGroup}^{\uparrow}$ *root operation-source packets are related by the root classifier, then the product row, inverse row, neighbourhood row, finite* Cont *ledger, and* Pkg *provenance row transport componentwise by the displayed* hsame *witnesses and the same ledger comparison. The transported packet still exposes only the displayed source pair and its two operation-continuity families.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThreshold classifier ledger transport packet`

Proof. Unpack the two root operation-source packets. The classifier compares the group source, topology source, product row, inverse row, neighbourhood row, ledger row, and provenance row componentwise through the carried hsame and Cont data.

Use Definition 204.39. It transports product continuity and inverse continuity through the same classifier packet, with the product and inverse arities read from the shared carrier rows.

Then apply Definition 204.60. Since the transported root surface has only the product and inverse continuity families listed in the finite ledger, classifier transport cannot introduce a host topological-group object or an ambient continuity predicate. □

Theorem 204.62 (TopGroupUp root operation-source continuity coupling)**.** *For an accepted root operation-source packet* $\mathcal{R} = (G, T, m, i, n, \lambda, \rho)$*, the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row* G *and displayed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row* T *determine the product-continuity and inverse-continuity families through the same neighbourhood row* n*, finite* Cont *ledger* λ*, and* Pkg *provenance row* ρ*. Classifier transport of* $\mathcal{R}$ *transports both continuity families through those same rows and cannot create an ambient continuity predicate outside the packet.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootOperationSourcePacket continuity coupling`

Proof. Apply Definition 204.59. It identifies G and T as the displayed source pair of a single BHist packet, with m, i, n, λ, and ρ read as projections from that same packet.

Next use Definition 204.60. The only operation-continuity rows over the displayed source pair are the product-continuity family for m and the inverse-continuity family for i, both indexed by the same neighbourhood row and finite Cont ledger.

Finally apply Definition 204.61. The classifier transports the product row, inverse row, neighbourhood row, ledger, and provenance componentwise, so it transports both continuity families together and has no row from which to synthesize an ambient continuity predicate. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root operation source continuity coupling`

204.6 Public threshold frontier

Definition 204.63 (TopGroupUp root public threshold packet)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *root public threshold packet is the finite* BHist *packet*

$$\Theta_{\mathsf{TopGroup}^{\uparrow}} = (G, T, m, i, n, \lambda, \rho, \tau)$$

whose first seven coordinates are the carrier $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ *of Definition 204.1. The row* G *is the displayed* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *source,* T *is the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source,* m *and* i *are the product and inverse operation rows,* n *is the neighbourhood transport row,* λ *is the finite* Cont *ledger, and* ρ *is the* Pkg *provenance row.*

The final row τ *is the public-threshold classifier row. It records that transport between two such packets is componentwise* hsame *on the source, operation, neighbourhood, and provenance rows, together with the displayed* Cont *comparison on the finite ledger. The packet exports a public* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *threshold over* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$*, but it does not add a host topological-group object or an ambient continuity predicate.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootPublicThresholdPacket`

Theorem 204.64 (TopGroupUp product-inverse joint continuity scope)**.** *For every accepted* $\mathsf{TopGroup}^\uparrow$ *root public threshold packet* $\Theta_{\mathsf{TopGroup}^\uparrow}$*, the public continuity scope is exactly the joint product-inverse scope carried by the shared packet: product continuity for* m*, inverse continuity for* i*, the common neighbourhood row* n*, the finite* Cont *ledger* λ*, and the* Pkg *provenance row* ρ*. No third operation-continuity family is exported at the root.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThreshold product inverse empty scope`

Proof. Project $\Theta_{\mathsf{TopGroup}^\uparrow}$ to the carrier packet $\mathcal{G} = (G, T, m, i, n, \lambda, \rho)$ of Definition 204.1. This exposes the displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ sources together with the product, inverse, neighbourhood, ledger, and provenance rows.

Use Definition 204.60. It identifies the root operation-continuity rows with precisely the product family for m and the inverse family for i.

Use Definition 204.40 to read both families through the same finite Cont ledger. The public threshold packet therefore has the listed joint continuity scope and no outside continuity row. □

Theorem 204.65 (TopGroupUp source ledger public threshold)**.** *Every accepted* $\mathsf{TopGroup}^\uparrow$ *root public threshold packet binds its* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *source,* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source, product row, inverse row, neighbourhood row, finite* Cont *ledger, and* Pkg *provenance into one public threshold surface. A downstream consumer can read the source pair and the ledger only as projections of that surface.*

Proof. Start from the threshold packet of Definition 204.63. Its carrier projection is the root operation-source packet of Definition 204.58.

Apply Definition 204.59. The displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ rows are the source pair of one shared BHist packet, not independent certificates.

Apply Definition 204.60. The ledger seen by the consumer is the same finite Cont ledger for product and inverse continuity, with the same Pkg provenance row. Thus the source pair and ledger form one public threshold surface. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage source ledger empty spine`

Theorem 204.66 (TopGroupUp NameCert frontier threshold)**.** *The* $\mathsf{TopGroup}^\uparrow$ *root public threshold packet is the frontier at which the displayed* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *rows become a public* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *surface. Its classifier transport is exactly the componentwise* hsame *transport of source, operation, neighbourhood, and provenance rows together with the finite* Cont *ledger comparison, and its provenance is exactly the displayed* Pkg *row.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThreshold frontier threshold`

Proof. Unpack the public-threshold classifier row τ. It contains only the componentwise hsame comparisons for the two sources, the product row, the inverse row, the neighbourhood row, and the provenance row, plus the finite Cont ledger comparison.

Use Definition 204.61. The same classifier transports the product, inverse, neighbourhood, ledger, and provenance rows while preserving the root source pair.

Finally apply Definition 204.7. The public certificate boundary is exhausted by the displayed BEDC rows. Hence the frontier threshold is exactly this finite packet over $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$. □

Definition 204.67 (TopGroupUp final public threshold packet)**.** *The final* $\mathsf{TopGroup}^\uparrow$ *public threshold packet is the finite* BHist *packet extending Definition 204.63 by making all public projections explicit:*

$$\Theta^{\mathsf{pub}}_{\mathsf{TopGroup}^\uparrow} = (G, T, m, i, n, \lambda, \tau, \rho).$$

The source projections are the displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *row* G *and the displayed* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *row* T*. The operation projections are the product row* m *and inverse row* i*. The neighbourhood projection is* n*, the finite ledger projection is* λ*, the classifier projection is* τ*, and the provenance projection is the* Pkg *row* ρ*.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootPublicThresholdFinalPacket`

Theorem 204.68 (TopGroupUp public threshold exhaustion)**.** *Every public threshold read of the root* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *surface projects to the source, operation, neighbourhood, finite ledger, classifier, and provenance rows of Definition 204.67. Conversely, those rows repack as the same final public threshold packet over* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThresholdPacket public threshold exhaustion`

Proof. Project the final packet to the root public threshold packet of Definition 204.63. Its carrier part supplies the two source rows, the product and inverse operation rows, the neighbourhood row, the finite Cont ledger, and the Pkg provenance row.

Apply Definition 204.64. The operation-facing public scope is exactly the joint product-inverse scope, so no third operation row or ambient continuity predicate is available.

Now use Definition 204.65 and Definition 204.66. These theorems bind the source pair, ledger, classifier, and provenance into the public $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ threshold. Repacking the listed projections gives the final packet and exhausts the public read. □

Theorem 204.69 (TopGroupUp public threshold transport)**.** *Componentwise* hsame *transport of the* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *source,* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source, product row, inverse row, neighbourhood row, classifier row, and provenance row, together with finite* Cont *transport of the ledger row, preserves the final* $\mathsf{TopGroup}^\uparrow$ *public threshold packet.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThreshold transport`

Proof. Unpack Definition 204.67. Each public projection is one displayed row of the finite BHist packet, so componentwise hsame transport acts on the source, operation, neighbourhood, classifier, and provenance projections without changing the packet shape.

Use Definition 204.65 for the ledger part. The finite Cont comparison is the same ledger comparison that binds the source pair and operation rows into one public threshold surface.

Finally apply Definition 204.66. Its classifier clause is precisely the componentwise transport just displayed, and therefore the transported rows repack as the same final public threshold packet. □

Theorem 204.70 (TopGroupUp public threshold NameCert surface)**.** *The rows of Definition 204.67 form the root public* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *surface over the two source certificates* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$*. This surface is exactly the source pair, product and inverse rows, neighbourhood row, finite* Cont *ledger, classifier row, and* Pkg *provenance row.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThreshold namecert surface`

Proof. Begin with the final threshold packet and project its source pair G, T. These are the displayed $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ rows, so the root certificate is explicitly over the two source certificates.

Apply Definition 204.64 to identify the product, inverse, neighbourhood, and ledger rows as the complete operation-continuity scope.

Then apply Definition 204.66. The frontier theorem packages the same componentwise classifier and provenance rows as the public $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ threshold, giving exactly the stated NameCert surface. □

Theorem 204.71 (TopGroupUp public threshold joint source determinacy)**.** *Two final public threshold packets over the same displayed* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *source rows, and with the same product, inverse, neighbourhood, finite* Cont *ledger, classifier, and* Pkg *provenance rows up to componentwise* hsame*, determine the same public* $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ *surface. The determined surface exposes the two source rows and the listed finite public threshold rows, and exposes no ambient continuity predicate.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootPublicThreshold joint source determinacy`

Proof. First apply Definition 204.65 to each packet. The common source rows, operation rows, neighbourhood row, finite Cont ledger, classifier row, and Pkg provenance row bind both reads to the same public threshold source ledger.

Next use Definition 204.66. The frontier theorem turns that shared source ledger into the public $\mathsf{NameCert}_{\mathsf{TopGroup}^\uparrow}$ threshold, so the two packets have the same NameCert-facing source boundary.

Apply Definition 204.69 to the componentwise hsame rows. This transports each displayed row of the first packet to the corresponding row of the second without adding any row beyond the final threshold packet.

Finally apply Definition 204.70. It identifies the transported threshold packet with exactly the source pair, product, inverse, neighbourhood, finite ledger, classifier, and provenance surface, so no ambient continuity predicate remains available to distinguish the two public reads. □

204.7 TopGroup root threshold downstream package

Definition 204.72 (TopGroupUp root threshold downstream package)**.** *The* $\mathsf{TopGroup}^\uparrow$ *root threshold downstream package is the finite* BHist *packet*

$$\Omega_{\mathsf{TopGroup}^\uparrow} = (G, T, m, i, n, \kappa, \lambda, \rho, \nu)$$

whose source pair G, T *is the displayed* $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$ *pair, whose operation rows* m, i *are the product and inverse rows, whose row* n *is the shared neighbourhood source, whose classifier packet* κ *is componentwise* hsame*, whose finite ledger* λ *is the product and inverse* Cont *ledger, whose provenance row* ρ *is the shared* Pkg *row, and whose threshold row* ν *records that a downstream public consumer reads only these NameCert-anchored coordinates.*

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootThresholdDownstreamPackage`

Theorem 204.73 (TopGroupUp root threshold carrier scope)**.** *Every accepted* $\mathsf{TopGroup}^{\uparrow}$ *root threshold downstream package carries the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *source row, the product row, the inverse row, the neighbourhood row, the componentwise* hsame *classifier packet, the finite* Cont *ledger, and the shared* Pkg *provenance as projections of one* BHist *carrier. No downstream root consumer may split the source pair into two carrier packages.*

Proof. Project the package of Definition 204.72. Its coordinates are the two NameCert source rows, the operation rows, the neighbourhood source, the classifier packet, the ledger, the provenance row, and the public threshold row.

Apply Definition 204.24. It identifies the source pair, product graph, inverse graph, neighbourhood row, finite Cont ledger, and Pkg provenance as projections of one BHist carrier.

Use Definition 204.19. A downstream public consumer receives only the carrier projections just listed, so the threshold surface has no second carrier or detached provenance row. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThreshold carrier scope`

Theorem 204.74 (TopGroupUp root threshold operation scope)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root threshold downstream package, the only operation-continuity rows exposed to a downstream consumer are the product continuity rows for* m *and the inverse continuity rows for* i*. Both families use the shared neighbourhood row* n*, the same componentwise* hsame *classifier packet* κ*, and the same finite* Cont *ledger* λ*.*

Proof. Begin with the carrier scope of Definition 204.73. It places the source pair, operation rows, neighbourhood row, classifier packet, ledger, and provenance over one BHist packet.

Apply Definition 204.25. The product continuity family is sound precisely through the two input endpoints, product endpoint, neighbourhood row, componentwise hsame transport, and finite ledger entry.

Apply Definition 204.28. The inverse continuity family has the same scope for its input endpoint, inverse endpoint, neighbourhood row, classifier transport, and ledger entry. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage operation scope`

Theorem 204.75 (TopGroupUp root threshold consumer exhaustion)**.** *Every downstream public consumer of a* $\mathsf{TopGroup}^{\uparrow}$ *root threshold package reads exactly the source pair, operation rows, neighbourhood row, componentwise* hsame *classifier packet, finite* Cont *ledger, and* Pkg *provenance of Definition 204.72. In particular, no ambient topological-group object, host continuity predicate, or detached NameCert source is available beyond those rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage consumer exhaustion`

Proof. First use Definition 204.73 to bind the source and provenance rows to the shared carrier. Then use Definition 204.74 to bind the product and inverse continuity rows to the same classifier packet and ledger.

Apply Definition 204.19. The public consumer surface contains only the displayed source, operation, classifier, continuity, and provenance rows. Hence every downstream read is exhausted by the root threshold package. □

Theorem 204.76 (TopGroupUp root threshold NameCert threshold)**.** *The rows of the* $\mathsf{TopGroup}^{\uparrow}$ *root threshold downstream package form a public* $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ *threshold over the displayed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *sources. The threshold classifier is exactly the componentwise* hsame *transport of source, operation, neighbourhood, and provenance rows together with finite* Cont *transport of the product and inverse ledger rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root threshold namecert threshold`

Proof. Unpack the threshold row ν in Definition 204.72. It records componentwise hsame transport for the NameCert source pair, product row, inverse row, neighbourhood row, and provenance row, plus finite Cont transport for the ledger.

Apply Definition 204.75. Since the downstream consumer reads no row outside the package, this classifier packet is the whole public threshold surface.

Finally use Definition 204.19. The public NameCert boundary is therefore exactly the displayed finite BHist packet over the two source certificates. □

Theorem 204.77 (TopGroupUp root threshold export boundary)**.** *Every downstream public export of the* $\mathsf{TopGroup}^{\uparrow}$ *root threshold factors through the single packet* $\Omega_{\mathsf{TopGroup}^{\uparrow}}$ *of Definition 204.72. The export exposes exactly the* $\mathsf{Group}^{\uparrow}$ *source, the* $\mathsf{Topology}^{\uparrow}$ *source, the product row, the inverse row, the shared neighbourhood row, the componentwise* hsame *classifier packet, the finite* Cont *ledger, the* Pkg *provenance, and the threshold row. It exposes no ambient topological-group object, no host continuity predicate, no detached source certificate, and no operation continuity row beyond the product and inverse ledger rows already carried by* $\Omega_{\mathsf{TopGroup}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root threshold export boundary`
↪ variant `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage export boundary`

Proof. Begin with Definition 204.73. It places the $\mathsf{Group}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ source certificates, the operation rows, the neighbourhood row, the classifier packet, the finite ledger, and the provenance row inside one BHist carrier.

Next use Definition 204.74. The product and inverse continuity data are precisely the ledger rows over the shared neighbourhood source and componentwise hsame classifier packet.

Apply Definition 204.75. A downstream public consumer reads only the displayed source, operation, classifier, continuity, and provenance rows, so an ambient topological-group object, host continuity predicate, or detached source certificate is outside the export.

Finally apply Definition 204.76. The threshold row is the public $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ boundary of the same packet. Repacking these projections gives the stated export boundary and leaves no extra operation-continuity row to expose. □

204.8 TopGroup root downstream obligation triad

Definition 204.78 (TopGroupUp root downstream obligation triad)**.** *The* $\mathsf{TopGroup}^{\uparrow}$ *root downstream obligation triad is the finite* BHist *packet*

$$\mathcal{D}_{\mathsf{TopGroup}^{\uparrow}} = (\eta_G, \eta_T, \mu, \iota, \mathcal{N}, C_\mu, C_\iota, H, \Pi)$$

whose source rows are the shared $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row* η_G*, the shared* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row* η_T*, the product row* μ*, the inverse row* ι*, the neighbourhood row family* $\mathcal{N}$*, the product and inverse continuity ledgers* C_μ *and* C_ι*, the displayed* hsame *comparison rows* H*, and the package provenance row* $\Pi : \mathsf{Pkg}$*.*

The packet is accepted exactly when these rows are the carrier, operation, continuity, history-sameness, and provenance projections of the root source packet of Definition 204.58, and when the public threshold row of Definition 204.72 authorizes no further downstream read.

Lean4 (definition) `BEDC.Derived.TopGroupUp.TopGroupRootDownstreamObligationTriad`

Theorem 204.79 (TopGroupUp root downstream carrier obligation)**.** *Every carrier-facing downstream read of an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root downstream obligation triad is exactly one of the shared* BHist *source rows* η_G*,* η_T*, or* $\mathcal{N}$*, together with its displayed* hsame *transport row from* H*. Conversely, those shared source rows repack as the carrier-facing boundary of the same triad.*

Proof. Project the triad to the root operation source packet. Apply Definition 204.59: it identifies the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ source, the $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ source, and the neighbourhood rows as the only carrier-facing source pair.

Then apply Definition 204.75. The threshold package allows a downstream consumer to read only the listed source rows and their displayed hsame comparisons, so no additional carrier row is available.

For the converse, insert those projected rows back into Definition 204.78. The acceptance clauses ask for exactly the root source pair and the threshold permission row, hence the same carrier-facing boundary is recovered. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root downstream carrier obligation`

Theorem 204.80 (TopGroupUp root downstream operation obligation)**.** *For an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root downstream obligation triad, the operation-facing boundary is exhausted by the product row* μ*, the inverse row* ι*, the product continuity ledger* C_μ*, the inverse continuity ledger* C_ι*, and their* hsame *transport rows. It includes no associativity, unit, inverse-law, quotient, completion, or standard-bridge datum beyond those operation and continuity rows.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root downstream operation transport`

Proof. First read the operation rows from the root packet. By Definition 204.60, the product and inverse rows are bound to the source pair only through the finite continuity ledgers C_μ and C_ι.

Next restrict the downstream view with Definition 204.75. A public downstream consumer receives the operation rows, their continuity ledgers, the source rows already isolated by Definition 204.79, and no further group law or host structure.

Repacking is componentwise: the product, inverse, and two continuity ledgers are coordinates of $\mathcal{D}_{\mathsf{TopGroup}^{\uparrow}}$, while the displayed hsame rows are exactly the transport rows recorded in H. This is the whole operation-facing boundary. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage operation ledger obligation`

Theorem 204.81 (TopGroupUp root downstream ledger obligation)**.** *The finite* Cont *ledger rows* C_μ *and* C_ι *together with the package provenance row* $\Pi : \mathsf{Pkg}$ *exhaust all downstream root reads of an accepted* $\mathsf{TopGroup}^{\uparrow}$ *root downstream obligation triad. Every such read is a carrier source row, an operation row, a displayed* hsame *row, a finite continuity-ledger row, or the package provenance row of the triad.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage downstream ledger obligation`

Proof. Begin with the carrier side. Apply Definition 204.79 to identify all source and neighbourhood reads as shared BHist rows over $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$.

Now use Definition 204.80 for the operation side. It says that product and inverse reads appear only through the displayed finite Cont ledgers and their hsame transport rows.

Finally apply Definition 204.75. The threshold package leaves no downstream root read outside the source pair, operation rows, continuity ledgers, history-sameness rows, and provenance package. Since those are exactly the coordinates of Definition 204.78, the finite ledger and Pkg provenance exhaust the root downstream surface. □

Theorem 204.82 (TopGroupUp root downstream continuity obligation)**.** *For an accepted* $\mathsf{TopGroup}^\uparrow$ *root downstream obligation triad, every product continuity read is exactly a row of the finite* Cont *ledger* C_μ *and every inverse continuity read is exactly a row of the finite* Cont *ledger* C_ι*, transported only by the displayed* hsame *rows in* H*. Conversely, the rows of* C_μ *and* C_ι *with their displayed* hsame *transports repack as the whole continuity-facing downstream surface of* $\mathcal{D}_{\mathsf{TopGroup}^\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage root downstream continuity obligation`

Proof. Project the accepted packet through Definition 204.78. Its continuity coordinates are precisely C_μ and C_ι, with no separate host continuity object.

Apply Definition 204.80. It identifies the product and inverse boundaries with the operation rows together with the finite continuity ledgers and their displayed hsame rows. Then apply Definition 204.81, which rules out any downstream root read outside the listed finite ledger, transport, source, and provenance rows.

For the converse, insert any row of C_μ or C_ι and its matching row of H back into the defining packet. The acceptance clauses are exactly the finite Cont ledger clauses of the root operation source packet, so the same rows recover the continuity-facing boundary. □

Theorem 204.83 (TopGroupUp root downstream provenance obligation)**.** *Every carrier, operation, continuity, and ledger read of an accepted* $\mathsf{TopGroup}^\uparrow$ *root downstream obligation triad is tied to the same package provenance row* $\Pi : \mathsf{Pkg}$*. In particular, no downstream read may exchange the carrier source, operation row,* Cont *ledger row, or displayed* hsame *transport for a row with a different provenance package.*

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootDownstreamPackage provenance transport`

Proof. Use Definition 204.79 for the carrier side: the visible carrier rows are the shared $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$, and neighbourhood rows, all projected from the same accepted packet.

Use Definition 204.80 and Definition 204.82 for the operation and continuity sides. Their reads are the product row, inverse row, finite Cont ledgers, and displayed hsame transports already present in $\mathcal{D}_{\mathsf{TopGroup}^\uparrow}$.

Finally apply Definition 204.81. It says that the finite ledger rows and the single package provenance row $\Pi : \mathsf{Pkg}$ exhaust the remaining downstream reads. Since all the cited rows are coordinates of the same packet, their provenance is the same Π. □

Theorem 204.84 (TopGroupUp root downstream threshold exactness)**.** *The public threshold row of the* $\mathsf{TopGroup}^\uparrow$ *root downstream package authorizes exactly the carrier rows, operation rows, continuity rows, displayed* hsame *rows, ledger rows, and package provenance row listed in* $\mathcal{D}_{\mathsf{TopGroup}^\uparrow}$*. It authorizes no group-law, completion, quotient, standard-bridge, or ambient topological-group datum outside those rows.*

Proof. First unpack $\mathcal{D}_{\mathsf{TopGroup}^{\uparrow}}$ by Definition 204.78. The displayed packet contains the two source certificates, product and inverse rows, neighbourhood rows, two finite Cont ledgers, the transport family H, and the provenance row Π.

The carrier boundary is exactly the one isolated in Definition 204.79. The operation boundary is exactly the one isolated in Definition 204.80. The continuity boundary is exactly the one isolated in Definition 204.82.

Apply Definition 204.81 to include the finite ledger and provenance rows. Then apply Definition 204.75: the public threshold package admits no downstream consumer row beyond this displayed finite family. Thus the threshold row is exact for the listed root downstream surface. □

Lean4 ✓ `BEDC.Derived.TopGroupUp.TopGroupRootThresholdPackage downstream threshold exactness`

Chapter 205

A Concrete Naming Certificate for $\mathsf{TopVecSpace}^{\uparrow}$

The topological-vector-space interface $\mathsf{TopVecSpace}^{\uparrow}$ packages a vector-space carrier with a Hausdorff topology making addition and scalar multiplication continuous. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88).

205.1 Carrier definition

Definition 205.1 (TopVecSpace BHist carrier)**.** *A* $\mathsf{TopVecSpace}^{\uparrow}$ *BHist carrier is a finite displayed packet*

$$(V, \mathcal{T}, A, S, \lambda, \rho)$$

whose fields are as follows.

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *carrier row, including the displayed zero, addition, negation, scalar-source, scalar-action, and vector-space classifier rows.*
2. *$\mathcal{T}$ is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row over the same displayed carrier, naming the open rows used to test continuity.*
3. *A is the finite addition-continuity ledger: each entry records source neighbourhood rows for two vector inputs, the displayed addition endpoint, and the target neighbourhood row.*
4. *S is the finite scalar-continuity ledger: each entry records a scalar-source row, a vector-source row, the displayed scalar-action endpoint, and the target neighbourhood row.*
5. *λ is the finite* Cont *ledger transporting the vector-space, topology, addition-continuity, and scalar-continuity rows.*
6. *ρ is the* Pkg *provenance row naming the* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *dependency certificates.*

Two packets classify together exactly when the vector-space row, topology row, continuity ledgers, continuation rows, and package provenance are transported componentwise by the available hsame *rows. The carrier exports no host topological-vector-space equality, ambient topology, or analytic continuity predicate outside the displayed ledgers.*

```
Lean4 ✓  BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier endpoint route readback
Lean4 ✓  BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier route transport boundary
```

Theorem 205.2 (TopVecSpace vector-space source obligation)**.** *Every accepted* $\mathsf{TopVecSpace}^{\uparrow}$ *packet projects its algebraic source exactly to the* $\mathsf{VecSpace}^{\uparrow}$ *row* V *of Definition 205.1. The zero, addition, negation, scalar-source, scalar-action, and vector-space classifier rows are consumed through* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and transported only by the carried* hsame, Cont, *and* Pkg *rows.*

Proof. Project an accepted packet to its first field V. By the carrier definition, V is the only row that exposes vector-space operations and the vector-space classifier.

The addition and scalar-continuity ledgers may refer to those operations, but they do so by naming the rows already present in V. Their references are routed through λ and the provenance row ρ.

Since no host vector space or ambient equality row is available in the packet, the algebraic source obligation is exactly the displayed $\mathsf{VecSpace}^{\uparrow}$ source row. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier vecspace source obligation`

Theorem 205.3 (TopVecSpace topology source obligation)**.** *Every accepted* $\mathsf{TopVecSpace}^{\uparrow}$ *packet projects its topological source exactly to the* $\mathsf{Topology}^{\uparrow}$ *row* $\mathcal{T}$ *of Definition 205.1. Open-neighbourhood membership, neighbourhood transport, and topological classifier data are consumed through* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *and remain tied to the vector-space carrier only through displayed* hsame, Cont, *and* Pkg *rows.*

Proof. Project the packet to $\mathcal{T}$. This row records the open-neighbourhood data used by the continuity ledgers, and it is the only field that consumes the topology certificate.

The vector-space row V supplies the carrier on which those opens are displayed. The connection between V and $\mathcal{T}$ is not host equality of underlying sets; it is the componentwise transport recorded by λ and named by ρ.

Thus every topological readback factors through $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ and the finite packet rows. No ambient topology or unlogged neighbourhood filter is exported. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier topology source obligation`

205.2 The certificate

Theorem 205.4 (TopVecSpace continuous addition obligation)**.** *For a* $\mathsf{TopVecSpace}^{\uparrow}$ *BHist carrier, every addition-continuity claim is a row of the finite ledger* A*. Each such row reads two vector-source neighbourhoods from* $\mathcal{T}$*, the displayed addition endpoint from* V*, and the target neighbourhood row from* $\mathcal{T}$*, with the route and provenance recorded by* λ *and* ρ*.*

Proof. Read an entry of the addition-continuity ledger A. Its input neighbourhoods and target neighbourhood are topology rows, while its operation endpoint is the addition row exposed by V.

The comparison data for transported packets are supplied componentwise by hsame. The finite continuation ledger λ records how the two input neighbourhoods, the addition endpoint, and the target neighbourhood remain aligned after transport.

Repacking these projections gives the addition-continuity obligation. Because the packet contains no host continuity predicate, no external topological-vector-space equality is used. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier continuous addition obligation`

Theorem 205.5 (TopVecSpace continuous scalar obligation)**.** *For a* $\mathsf{TopVecSpace}^{\uparrow}$ *BHist carrier, every scalar-continuity claim is a row of the finite ledger* S*. Each such row reads the scalar-source and vector-source coordinates from* V*, the relevant neighbourhood rows from* $\mathcal{T}$*, and the scalar-action endpoint from* V*, with transport and provenance recorded by* λ *and* ρ*.*

Proof. Read an entry of the scalar-continuity ledger S. The scalar coordinate, vector coordinate, and scalar-action endpoint are projected from the $\mathsf{VecSpace}^{\uparrow}$ row V.

Read the neighbourhood coordinates from $\mathcal{T}$. Their stability under replacement is witnessed by the carried hsame comparisons and the finite continuation route in λ.

The provenance row ρ records that the algebraic and topological reads come from the two named dependency certificates. Therefore the scalar-continuity obligation is internal to the displayed packet and does not invoke host topological-vector-space equality. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier continuous scalar obligation`

Theorem 205.6 (TopVecSpace classifier transport)**.** *Let $P = (V, \mathcal{T}, A, S, \lambda, \rho)$ and $P' = (V', \mathcal{T}', A', S', \lambda', \rho')$ be displayed $\mathsf{TopVecSpace}^{\uparrow}$ packets. If the $\mathsf{VecSpace}^{\uparrow}$ row, $\mathsf{Topology}^{\uparrow}$ row, addition-continuity ledger, scalar-continuity ledger, finite Cont ledger, and Pkg provenance are transported componentwise by the carried hsame rows from P to P', then accepted carrier status transports from P to P'. Under the same componentwise transport, every accepted addition-continuity ledger row of P transports to an accepted row of A', and every accepted scalar-continuity ledger row of P transports to an accepted row of S'.*

Proof. Project the two packets to their displayed rows. The carrier definition makes the vector-space row, topology row, the two continuity ledgers, the continuation ledger, and the package provenance the complete classifier surface.

Use the supplied hsame comparisons on each row. They replace V by V', $\mathcal{T}$ by $\mathcal{T}'$, A by A', S by S', λ by λ', and ρ by ρ' without introducing any coordinate outside the displayed packet.

The accepted carrier predicate is therefore rebuilt by the same finite tuple with primed rows. The addition and scalar obligations follow by applying Definition 205.4 and Definition 205.5 to the transported ledgers. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier classifier transport`

Theorem 205.7 (TopVecSpace NameCert boundary)**.** *The carrier definition of Definition 205.1, the vector-space and topology source obligations of Definition 205.2 and Definition 205.3, the continuous addition and scalar obligations of Definition 205.4 and Definition 205.5, and the classifier transport of Definition 205.6 assemble the $\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}$ boundary over BHist, hsame, Cont, and Pkg.*

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier namecert boundary`

Proof. Start with a displayed packet $(V, \mathcal{T}, A, S, \lambda, \rho)$ of Definition 205.1. The first two source obligations identify exactly the dependency-certificate rows consumed from $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$.

The two continuity obligations identify the only analytic rows exposed by the packet: addition is read from A, scalar multiplication is read from S, and both ledgers route through λ with provenance in ρ.

Finally apply Definition 205.6. It supplies the closure of the accepted packet surface under componentwise hsame replacement, so the carrier, source, continuity, continuation, and package rows repack as the displayed $\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}$ boundary. □

Theorem 205.8 (TopVecSpace consumer scope)**.** *Every downstream consumer of a $\mathsf{TopVecSpace}^{\uparrow}$ certificate reads only the displayed $\mathsf{VecSpace}^{\uparrow}$ row V, the displayed $\mathsf{Topology}^{\uparrow}$ row $\mathcal{T}$, the addition and scalar continuity ledgers A and S, the continuation ledger λ, and the provenance row ρ. Such a consumer cannot infer ambient topological-vector-space equality, host continuity of addition or scalar multiplication, or any topology or vector-space structure not present in the displayed packet.*

Proof. Inspect the NameCert boundary of Definition 205.7. Its exported rows are precisely the carrier tuple, the two source projections, the two continuity ledgers, the componentwise transport route, and the provenance row.

A consumer read is therefore a projection from those rows. If it asks for algebraic data, it lands in V; if it asks for topological data, it lands in $\mathcal{T}$; if it asks for continuity, it lands in A or S and follows λ and ρ.

No constructor in the boundary provides host equality of topological vector spaces, an ambient continuity predicate, or an external topology. Those conclusions are therefore outside the consumer scope of the certificate. □

Lean4 ✓ `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier consumer scope`

Closure status for $\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}$.

Constructive story: A topological vector-space packet is generated from a checked $\mathsf{VecSpace}^{\uparrow}$ source, a checked $\mathsf{Topology}^{\uparrow}$ source, displayed rows for continuous addition and continuous scalar action, componentwise hsame transport, backward Cont provenance, and Pkg links naming exactly those two sources. The consumer reads only that finite BHist surface, so algebraic vector-space data, topology data, operation-continuity rows, classifier transport, and downstream consumer scope are all projected from visible rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 205.8 gives the public consumer surface over the displayed vector-space row, topology row, continuity ledgers, classifier transport, and provenance packet for $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, and Pkg.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TopVecSpaceUp.TopVecSpaceBHistCarrier\_consumer\_scope`

Bridge status: none

Not claimed: No locally convex theorem, analytic topological-vector-space theorem, host continuity predicate, or standard bridge is claimed here.

Upgrade path: The paper-axis consumer treatment is mature for the displayed packet; formal bridge checking remains separate.

Chapter 206

A Concrete Naming Certificate for $\mathsf{AffineSpace}^{\uparrow}$

The affine-space interface $\mathsf{AffineSpace}^{\uparrow}$ packages a torsor over a vector-space carrier: a set with a free transitive action of vector translations. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41).

206.1 Carrier definition

Definition 206.1 (Affine history torsor carrier)**.** *An* $\mathsf{AffineSpace}^{\uparrow}$ *carried datum is a* BHist *package*

$$\mathcal{A} = (\mathcal{G}, \mathcal{V}, p, \tau, \eta_{\mathcal{G}}, \eta_{\mathcal{V}}, \lambda)$$

where $\mathcal{G}$ *is the* $\mathsf{Group}^{\uparrow}$ *additive-translation source,* $\mathcal{V}$ *is the* $\mathsf{VecSpace}^{\uparrow}$ *vector source,* $p : \mathsf{BHist}$ *is the point row,* $\tau : \mathsf{BHist}$ *is the vector-translation row,* $\eta_{\mathcal{G}}$ *and* $\eta_{\mathcal{V}}$ *are the displayed source* $\mathsf{NameCert}$ *witnesses, and* λ *records that the translation row is read through the vector carrier of* $\mathcal{V}$ *and the additive group operation exposed by* $\mathcal{G}$*.*

The affine classifier on two carried rows $\mathcal{A}$ *and* $\mathcal{A}'$ *is the componentwise relation saying that the point rows are* hsame*-compatible and that the translation rows are classified by the* $\mathsf{VecSpace}^{\uparrow}$ *vector classifier:*

$$\mathsf{AffCls}(\mathcal{A}, \mathcal{A}') := \mathsf{hsame}(p, p') \wedge \mathsf{VecCls}_{\mathcal{V}}(\tau, \tau').$$

The affine action of a certified vector translation $v : \mathsf{BHist}$ *on a carried point row* p *is the* Cont*-append row*

$$\mathsf{act}(p, v) := \mathsf{Cont}(p, \mathsf{trans}(v)),$$

where $\mathsf{trans}(v)$ *is the certified vector-translation packet scoped by the* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Group}^{\uparrow}$ *source certificates. Thus the carrier is a torsor surface over* $\mathsf{VecSpace}^{\uparrow}$ *whose visible rows are only the point row, the vector-translation row, the source* $\mathsf{NameCert}$ *witnesses, and the finite* Cont *action ledger.*

Lean4 (definition) `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier`

Definition 206.2 (AffineSpaceUp translation classifier)**.** *For point rows* $p, q : \mathsf{BHist}$ *in the carrier of Definition 206.1, the* $\mathsf{AffineSpace}^{\uparrow}$ *translation classifier declares*

$$p \sim_{\mathsf{Aff}} q$$

exactly when there are translation witnesses v, w : BHist *licensed by the* $\mathsf{VecSpace}^{\uparrow}$ *dependency certificate and the* $\mathsf{Group}^{\uparrow}$ *dependency certificate such that the displayed action routes from* p *to* q *are equal up to* hsame*. Equivalently, the classifier is equality of translation witnesses after transport through the source classifiers of* $\mathcal{V}$ *and* $\mathcal{G}$ *and through the* hsame *comparison on the* Cont *action rows.*

Lean4 (definition) `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier`

206.2 The certificate

Theorem 206.3 (AffineSpaceUp action closure obligation)**.** *Let* $\mathcal{V}$ *be a* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *dependency certificate and let* $\mathcal{G}$ *be a* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *dependency certificate. If* p : BHist *is carried by Definition 206.1 and* v : BHist *is a vector row accepted by* $\mathcal{V}$ *and by the translation row of* $\mathcal{G}$*, then every certified* Cont *action witness from* (p, v) *has an endpoint* q : BHist *carried by the same* $\mathsf{AffineSpace}^{\uparrow}$ *carrier, and the endpoint is stable under* hsame *transport of the action witness.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier action closure obligation`

Proof. Unpack the carrier row of p. It contains the $\mathsf{VecSpace}^{\uparrow}$ source certificate, the $\mathsf{Group}^{\uparrow}$ source certificate, and the finite ledger slot reserved for translation actions.

Insert the displayed Cont witness into that ledger. The $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ dependency scopes the vector endpoint, while the $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ dependency scopes the translation row that acts on the point. Therefore the endpoint q is generated by the same carrier predicate over the same two certificates.

If the action witness is transported by hsame, use the carrier transport fields of the two dependency certificates on the vector and group endpoints. Repacking the transported rows with the same point-source data gives the required carried endpoint. □

Theorem 206.4 (AffineSpaceUp action coverage obligation)**.** *For any carried point rows* p, q : BHist *in the* $\mathsf{AffineSpace}^{\uparrow}$ *carrier over* $\mathcal{V}$: $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathcal{G}$: $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$*, the certificate obligation surface contains a displayed vector row* v : BHist *and a* Cont *action witness carrying* p *to a row* hsame*-classified with* q*. The vector witness is accepted by the* $\mathsf{VecSpace}^{\uparrow}$ *dependency certificate, its translation reading is accepted by the* $\mathsf{Group}^{\uparrow}$ *dependency certificate, and the witness is unique only up to the classifier of Definition 206.2.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier action coverage obligation`
↪ variant `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier action coverage obligation`

Proof. Read the two point rows through the torsor carrier. The carrier records both rows against the same $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{Group}^{\uparrow}$ dependency certificates, so the only admissible route between them must be a translation row licensed by those dependencies.

The obligation is then stated by choosing the displayed vector row and the Cont action witness in the ledger. The endpoint comparison with q is not host equality of points; it is the hsame row required by the carrier and classifier definitions.

Transporting the chosen witness along either dependency classifier gives an equivalent witness. Thus coverage is recorded as existence of a BEDC translation route, while uniqueness is deferred to the translation classifier rather than exposed as an extra carrier field. □

Theorem 206.5 (AffineSpaceUp vector difference obligation)**.** *For carried point rows $p, q : \mathsf{BHist}$ over $\mathcal{V} : \mathsf{NameCert}_{\mathsf{VecSpace}^\uparrow}$ and $\mathcal{G} : \mathsf{NameCert}_{\mathsf{Group}^\uparrow}$, any two certified vector-difference witnesses from p to q are identified by the $\mathsf{AffineSpace}^\uparrow$ translation classifier. The comparison uses the $\mathsf{VecSpace}^\uparrow$ classifier on vector histories, the $\mathsf{Group}^\uparrow$ classifier on translation histories, and hsame transport along the two Cont action rows.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier witnesses identified`

Proof. Take two displayed difference witnesses. Each witness consists of a vector row, its translation-group reading, and a Cont action row from p to an endpoint classified with q.

Apply the source classifier field of $\mathcal{V}$ to compare the two vector rows and the source classifier field of $\mathcal{G}$ to compare their translation readings. The two endpoint comparisons to q are then concatenated by hsame symmetry and transitivity.

The resulting comparison is exactly the relation in Definition 206.2. Hence the difference row is determined at the $\mathsf{AffineSpace}^\uparrow$ boundary only up to the BEDC classifier, with all chosen action ledgers remaining internal. □

Theorem 206.6 (AffineSpaceUp separation obligation)**.** *If carried point rows $p, q : \mathsf{BHist}$ admit a certified zero-translation action from p to a row hsame-classified with q, then p and q are classified by the identity case of Definition 206.2. The zero row is the vector zero supplied by $\mathsf{NameCert}_{\mathsf{VecSpace}^\uparrow}$ and the identity translation is the group identity supplied by $\mathsf{NameCert}_{\mathsf{Group}^\uparrow}$.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier separation obligation`

Proof. Project the zero vector row from the $\mathsf{VecSpace}^\uparrow$ dependency certificate and the identity translation row from the $\mathsf{Group}^\uparrow$ dependency certificate. These rows are the two dependency endpoints of the zero-translation action.

The action ledger says that acting by this identity translation is a Cont row from p to an endpoint classified with q. The identity action law is therefore recorded as an hsame transport between the original point row and the endpoint row.

Combine that transport with the displayed endpoint comparison to q. Repacking the zero vector, identity translation, and composed hsame row gives the identity case of the translation classifier, so the carrier separates point rows exactly at the certified zero-translation boundary. □

Theorem 206.7 (Affine vector-action stability)**.** *Let $\mathcal{A}$ and $\mathcal{A}'$ be carried $\mathsf{AffineSpace}^\uparrow$ rows of Definition 206.1. If their point rows are related by hsame and their vector translations are related by the $\mathsf{VecSpace}^\uparrow$ classifier, then appending any certified vector-translation packet by Cont preserves the affine classifier. Equivalently, for carried point rows p, p' and vector rows v, v',*

$$\mathsf{hsame}(p, p') \wedge \mathsf{VecCls}_{\mathcal{V}}(v, v')$$

implies that the action endpoints $\mathsf{Cont}(p, \mathsf{trans}(v))$ and $\mathsf{Cont}(p', \mathsf{trans}(v'))$ are classified by AffCls.

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier vector action stability`

Proof. Use the $\mathsf{VecSpace}^\uparrow$ certificate to transport v to v' through its vector classifier. The $\mathsf{Group}^\uparrow$ dependency reads the transported vector row as the same certified additive-translation packet in the translation source.

Apply Cont endpoint transport to the point comparison $\mathsf{hsame}(p, p')$ and to the transported translation packet. The Cont associativity and unit rows keep the appended action ledger in the same finite-history surface, so the resulting endpoint pair satisfies the displayed affine classifier. □

Theorem 206.8 (Affine free-action obligation)**.** *For the carrier of Definition 206.1, the free-action obligation is the following BEDC row: if two certified vector translations v, w act on the same carried point row p and the resulting point endpoints are related by* hsame*, then the vector rows v and w are related by the* $\mathsf{VecSpace}^{\uparrow}$ *classifier. The only accepted witnesses are the shared point row, the two vector-translation packets, the two* Cont *action ledgers, and the source* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *rows.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier free action obligation`

Proof. Read both action endpoints as Cont appends from the same point row p. The left unit and right unit behavior of Cont identifies the empty translation packet as the neutral action row supplied by the additive $\mathsf{Group}^{\uparrow}$ source.

Cancel the shared point ledger by composing the first action with the inverse translation packet supplied by the $\mathsf{Group}^{\uparrow}$ certificate. Associativity of Cont moves the inverse packet next to the original packet, and the group inverse and unit rows reduce the visible endpoint comparison to a comparison of the two translation rows.

Finally, the $\mathsf{VecSpace}^{\uparrow}$ classifier is the public classifier for translation rows in this carrier. Transport through the $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ witness repacks the endpoint comparison as $\mathsf{VecCls}_{\mathcal{V}}(v, w)$, with no host-level point equality used. □

Theorem 206.9 (Affine transitive-action obligation)**.** *For any two carried point rows p and q in the* $\mathsf{AffineSpace}^{\uparrow}$ *carrier, the transitive-action obligation records a certified vector row v whose translation packet carries p to a point row classified with q. The witness ledger consists exactly of the source* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row for v, the additive* $\mathsf{Group}^{\uparrow}$ *row interpreting v as a translation packet, and the* Cont *action ledger from p to q.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier transitive action obligation`

Proof. Use the torsor source row in the carrier definition to read the difference from p to q as a vector-translation row in the $\mathsf{VecSpace}^{\uparrow}$ carrier. This is a named BHist row, not a quotient of host points.

The $\mathsf{Group}^{\uparrow}$ certificate interprets that vector row as an additive translation packet. Appending the packet to p by Cont gives the action endpoint, and the stored torsor ledger supplies the hsame comparison between this endpoint and q.

The witness is therefore the displayed vector row together with its two source certificates and the finite action ledger. Exactness of the ledger prevents any additional choice of ambient affine coordinate system from entering the obligation. □

Theorem 206.10 (Affine classifier transport)**.** *The affine classifier of Definition 206.1 is stable under* hsame *transport of point rows and under* $\mathsf{VecSpace}^{\uparrow}$ *classifier transport of vector-translation rows. If* $\mathsf{AffCls}(\mathcal{A}, \mathcal{A}')$ *and the source* NameCert *rows transport $\mathcal{A}$ to $\mathcal{B}$ and $\mathcal{A}'$ to $\mathcal{B}'$, then* $\mathsf{AffCls}(\mathcal{B}, \mathcal{B}')$ *holds.*

Proof. Unfold the affine classifier into its point and translation components. The point component is an hsame comparison, so reflexivity, symmetry, transitivity, and endpoint transport are exactly the corresponding hsame rows.

For the translation component, apply the $\mathsf{VecSpace}^{\uparrow}$ certificate transport field to the two vector rows. The $\mathsf{Group}^{\uparrow}$ source is used only to keep the transported vector rows accepted as additive translation packets.

Recombine the transported point comparison and transported vector comparison. Since the classifier contains no hidden field beyond those two components, the recombined pair is precisely the target affine classifier. □

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier classifier transport`
Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier transport`

Theorem 206.11 (Affine ledger exactness)**.** *The* $\mathsf{AffineSpace}^{\uparrow}$ *obligation ledger contains exactly the* $\mathsf{Group}^{\uparrow}$ *source row, the* $\mathsf{VecSpace}^{\uparrow}$ *source row, the carried point* BHist *row, the vector-translation* BHist *row, the* hsame *point classifier component, the* $\mathsf{VecSpace}^{\uparrow}$ *translation classifier component, the* Cont *action rows, and the two source* NameCert *witnesses. No coordinate chart, host torsor equality, quotient representative, or standard bridge row is part of this obligation surface.*

Proof. Soundness is by projection from the carrier tuple. Each accepted affine row exposes the two source certificates, the point row, the vector-translation row, and the two source NameCert witnesses. The classifier definition exposes only hsame on point rows and the $\mathsf{VecSpace}^{\uparrow}$ classifier on vector rows.

The action rows are sound because they are built by appending the certified translation packet with Cont. Associativity and the unit behavior of Cont account for changing parenthesization and for the neutral translation row, while the $\mathsf{Group}^{\uparrow}$ certificate supplies the additive inverse and unit packets used by the free-action ledger.

Exactness is the reverse reading. Any row used by stability, freeness, transitivity, or classifier transport is one of the projected carrier rows, one of the named source-certificate fields, or one finite Cont action row. Hence the public obligation surface is exactly the listed BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ data. □

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceTranslationClassifier ledger exactness`

Theorem 206.12 (AffineSpaceUp NameCert obligation surface)**.** *The* $\mathsf{AffineSpace}^{\uparrow}$ *certificate obligation surface consists exactly of the* BHist *affine history torsor carrier of Definition 206.1, the translation classifier of Definition 206.2, the action closure, action coverage, vector-difference, separation, vector-action stability, free-action, transitive-action, classifier-transport, and ledger-exactness obligations of Definition 206.3, Definition 206.4, Definition 206.5, Definition 206.6, Definition 206.7, Definition 206.8, Definition 206.9, Definition 206.10, and Definition 206.11, together with the two dependency certificates* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$*. No host affine-space carrier, chosen origin, coordinate chart, quotient representative, or external vector-difference function is exported as a public field.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier semantic name certificate`

Proof. Soundness is obtained by unpacking the carrier and classifier rows. Every visible point row is a BHist row scoped by the two dependency certificates, and every visible comparison is a translation witness read through Cont and hsame.

The obligation theorems supply the action rows consumed by the certificate: closure gives carried endpoints, coverage gives a route between carried points, vector difference identifies parallel witnesses, separation fixes the identity route, stability and transport keep classifier rows invariant under accepted motion, freeness and transitivity record the torsor laws, and ledger exactness fixes the public surface. Each row is constructed only from the dependency NameCert fields and the finite action ledger.

For exactness, read any accepted ledger row in reverse. It must be either a point row, a vector or group dependency row, a Cont action witness, or an hsame transport used by the classifier. Therefore the visible $\mathsf{NameCert}_{\mathsf{AffineSpace}^{\uparrow}}$ boundary is exactly the listed surface and no further public datum is claimed. □

Closure status for $\mathsf{NameCert}_{\mathsf{AffineSpace}^{\uparrow}}$.

Constructive story: An affine space in this chapter is generated by taking a carried point history, a vector-translation history certified by $\mathsf{VecSpace}^{\uparrow}$, the additive-translation reading supplied by $\mathsf{Group}^{\uparrow}$, and a finite Cont action ledger that appends the translation packet to the point. The carrier admits exactly rows with those two dependency certificates, the point row, the translation row, the source witnesses, and the action ledger; a row lacking the vector source, the group source, or the displayed action provenance stays outside the affine surface. Two carried rows are classified by point hsame together with the $\mathsf{VecSpace}^{\uparrow}$ classifier on translation histories, so vector differences are recognized through displayed action witnesses rather than through a chosen origin or host torsor equality. The source covered by the certificate is this torsor packet itself, the pattern records action closure, coverage between carried points, separation by the zero translation, freeness, transitivity, action additivity, and classifier transport, and the ledger keeps the raw point, translation, group reading, vector reading, action endpoint, and source-certificate rows visible. The five name-certificate obligations come from any accepted torsor row as a carrier witness, from hsame and the vector classifier for reflexivity, symmetry, and transitivity, and from the transport rows showing that classified point and translation replacements repack as accepted affine rows. A history h lies in the signature gap of an $\mathsf{AffineSpace}^{\uparrow}$ token p when the source-domain accepts it as such a torsor packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, h carries the same point, vector-translation, action, and dependency rows encoded by p. The chapter registers this torsor-action construction under the name $\mathsf{AffineSpace}^{\uparrow}$; affine structure is assembled from BHist, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$, with no chosen origin, coordinate chart, host affine object, or standard bridge introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AffineSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: The affine torsor carrier, classifier, translation action, public certificate, and abstract interface bridge are closed over the displayed BHist, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AffineSpace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier\_semantic\_name\_certificate`

Bridge status: none

Not claimed: No public affine object, coordinate-chart surface, host torsor equality, chosen origin, or standard bridge is claimed.

Upgrade path: The paper-axis treatment is mature; remaining coordination concerns bridge status and formal-axis reporting.

Theorem 206.13 (AffineSpaceUp action additivity)**.** *Let* $\mathcal{V} : \mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathcal{G} : \mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *be the dependency certificates of Definition 206.1. Let* $p : \mathsf{BHist}$ *be a carried point row, and let* $v, w : \mathsf{BHist}$ *be vector rows accepted by* $\mathcal{V}$ *whose translation readings are accepted by* $\mathcal{G}$*. Write* $v +_V w$ *for the vector-addition row supplied by* $\mathcal{V}$*. Then the one-step action endpoint* $\mathsf{act}(p, v +_V w)$ *and the iterated action endpoint* $\mathsf{act}(\mathsf{act}(p, v), w)$ *admit carried affine repackings over the same source certificates and the same cumulative translation row* $v +_V w$*, and those repackings are classified by* AffCls*.*

Lean4 ✓ `BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier action additivity`

Proof. Apply Definition 206.3 to the carried point row p and the accepted vector row $v +_V w$. This produces a carried affine endpoint whose point component is $\mathsf{act}(p, v +_V w)$.

Apply the same closure theorem first to (p, v) and then to the carried endpoint obtained from that first action together with w. This produces a carried affine endpoint whose point component is $\mathsf{act}(\mathsf{act}(p, v), w)$.

Unfold the action definition from Definition 206.1. The one-step endpoint is obtained by appending to p the translation packet read from $v +_V w$. The iterated endpoint is obtained by first appending the packet read from v and then appending the packet read from w.

Now unpack the field λ from Definition 206.1. It records that the translation row attached to the vector-sum $v +_V w$ is read through the vector carrier of $\mathcal{V}$ and the additive translation operation exposed by $\mathcal{G}$. Therefore the tail used by the one-step action and the composite tail used by the iterated action are two certified readbacks of the same cumulative translation row.

Apply Definition 21.29 to the continuation ledger starting at p with these two bracketings. It yields

$$\mathsf{hsame}(\mathsf{act}(p, v +_V w), \mathsf{act}(\mathsf{act}(p, v), w)).$$

Finally repack both endpoints as affine rows over the same two source certificates and the common cumulative translation row $v +_V w$. The point components are related by the displayed hsame row, and the translation components are the same vector-sum row, hence are classified by the $\mathsf{VecSpace}^{\uparrow}$ classifier. By the definition of AffCls in Definition 206.1, the two repackings are classified together. □

↪ variant BEDC.Derived.AffineSpaceUp.AffineSpaceHistoryTorsorCarrier action additivity

Theorem 206.14 (AffineSpaceUp zero action classifies the carried point)**.** *Let* $\mathcal{V} : \mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathcal{G} : \mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *be the dependency certificates of Definition 206.1. Let* $p : \mathsf{BHist}$ *be a carried point row, and let* $0_V : \mathsf{BHist}$ *be the vector zero row supplied by* $\mathcal{V}$ *whose translation reading is the identity translation row supplied by* $\mathcal{G}$*. Then the zero-action endpoint* $\mathsf{act}(p, 0_V)$ *and the original point row* p *admit carried affine repackings over the same two source certificates and the same zero translation row, and those repackings are classified by* AffCls*. Equivalently, the displayed zero action gives the identity case of the translation classifier between* p *and* $\mathsf{act}(p, 0_V)$*.*

Proof. Unfold the action clause in Definition 206.1. There exist histories $t, u : \mathsf{BHist}$ such that t is the certified translation packet read from the vector zero row 0_V through the $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ dependencies, u is the appended endpoint witnessing

$$\mathsf{Cont}(p, t, u),$$

and u is classified by hsame with $\mathsf{act}(p, 0_V)$. By the hypothesis on 0_V, the packet t is the identity translation row exposed by the $\mathsf{Group}^{\uparrow}$ dependency certificate.

Apply Definition 206.3 to the carried point row p and the accepted vector row 0_V. The endpoint u is therefore carried by the same $\mathsf{AffineSpace}^{\uparrow}$ carrier over $\mathcal{V}$ and $\mathcal{G}$.

The same displayed action ledger is a certified zero-translation action from p to the endpoint u. Its endpoint comparison with u is the reflexive hsame row. Thus the hypotheses of Definition 206.6 hold with $q = u$, and the separation obligation repacks the zero vector row, the identity translation row, and the endpoint comparison as the identity case of Definition 206.2.

It remains only to read this identity classification in the affine carrier language. Repack p and u as affine rows over the same source certificates and with the same zero translation row 0_V. The point component is the hsame comparison supplied by the identity action case, and the translation component is the reflexive $\mathsf{VecSpace}^{\uparrow}$ classifier comparison of 0_V with itself. By the definition of AffCls in Definition 206.1, the two carried repackings are classified together.

Finally transport the carried repacking of u along the recorded hsame comparison between u and $\mathsf{act}(p, 0_V)$. This gives the stated classification of the original point row p with the zero-action endpoint $\mathsf{act}(p, 0_V)$ over the same two source certificates and the same zero translation row. □

Theorem 206.15 (AffineSpaceUp inverse-vector action cancels to the carried point)**.** *Let* $\mathcal{V} : \mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *and* $\mathcal{G} : \mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *be the dependency certificates of Definition 206.1. Let* $p : \mathsf{BHist}$ *be a carried point row, and let* $v : \mathsf{BHist}$ *be a vector row accepted by* $\mathcal{V}$ *whose translation*

reading is accepted by $\mathcal{G}$. Suppose the vector inverse row $-_V v$ is the additive inverse supplied by $\mathcal{V}$, its translation reading is the inverse translation packet supplied by $\mathcal{G}$, and the dependency certificates classify the cumulative vector row by the zero row:

$$\mathsf{VecCls}_{\mathcal{V}}(v +_V (-_V v), 0_V).$$

Then the iterated inverse-action endpoint $\mathsf{act}(\mathsf{act}(p, v), -_V v)$ and the original point row p admit carried $\mathsf{AffineSpace}^{\uparrow}$ repackings over the same source certificates, and those repackings are classified by AffCls.

Proof. Apply Definition 206.13 to the accepted vector rows v and $-_V v$. It gives carried affine repackings of the one-step endpoint $\mathsf{act}(p, v +_V (-_V v))$ and the iterated endpoint $\mathsf{act}(\mathsf{act}(p, v), -_V v)$ over the same two source certificates, with cumulative translation row $v +_V (-_V v)$, and it classifies those repackings by AffCls.

The vector inverse hypothesis is read through the two dependency certificates. On the $\mathsf{VecSpace}^{\uparrow}$ side, $v +_V (-_V v)$ is classified with the zero vector row 0_V. On the $\mathsf{Group}^{\uparrow}$ side, the translation reading of $-_V v$ is the inverse packet for the translation reading of v, so the cumulative translation packet read from $v +_V (-_V v)$ is classified with the identity translation row.

Use the classifier transport theorem Definition 206.10 to transport the one-step repacking of $\mathsf{act}(p, v +_V (-_V v))$ along the displayed vector classifier comparison from $v +_V (-_V v)$ to 0_V. The transported repacking has point component classified with $\mathsf{act}(p, 0_V)$, keeps the same $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ sources, and carries the zero translation row.

Now apply Definition 206.14. It classifies the zero-action endpoint $\mathsf{act}(p, 0_V)$ with the original carried point row p after repacking both over the same source certificates and the same zero translation row.

Finally compose the affine classifier comparisons: the additivity comparison from the iterated endpoint to the cumulative one-step endpoint, the classifier-transport comparison from the cumulative endpoint to the zero-action endpoint, and the zero-action comparison from that endpoint to p. Since AffCls is componentwise hsame on point rows together with the $\mathsf{VecSpace}^{\uparrow}$ classifier on translation rows, its transitivity is inherited from those two component classifiers. Therefore $\mathsf{act}(\mathsf{act}(p, v), -_V v)$ and p admit carried affine repackings over the same two source certificates and are classified by AffCls. □

Chapter 207

A Concrete Naming Certificate for $\mathsf{ProjectiveSpace}^{\uparrow}$

The projective-space interface $\mathsf{ProjectiveSpace}^{\uparrow}$ packages the quotient of a punctured vector-space carrier by the scaling action of the underlying field's multiplicative group. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45).

207.1 Carrier definition

Theorem 207.1 (ProjectiveSpaceUp punctured carrier obligation)**.** *The* $\mathsf{ProjectiveSpace}^{\uparrow}$ *carrier is the* $\mathsf{VecSpace}^{\uparrow}$ *carrier of Definition 47.1 equipped with the* $\mathsf{Field}^{\uparrow}$ *nonzero-vector guard supplied by Definition 45.198 and the certified scalar-action rows required by Definition 47.21. A carried projective endpoint is therefore a vector-space row whose representative is guarded as nonzero over the field source, together with the ledger entries recording the scalar source, vector source, and scalar-action certificate used to read that row.*

Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpacePuncturedCarrier obligation`

Proof. Start with the vector carrier row from Definition 47.1. The certificate obligations in Definition 47.21 provide the scalar-action closure and congruence rows over the same scalar source.

The field source specification in Definition 45.198 supplies the nonzero guard and the proof-indexed inverse operations available for guarded scalars. Attaching that guard to the vector representative removes the zero row from the visible projective carrier while leaving the underlying $\mathsf{VecSpace}^{\uparrow}$ provenance unchanged.

Thus the carrier predicate is exactly the punctured vector-space row together with the field and scalar-action certificate rows. No host quotient carrier or external projective-point object is introduced. □

Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpacePuncturedCarrier empty boundary`
Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpaceSingletonPuncturedCarrier obligation`

Theorem 207.2 (ProjectiveSpaceUp scaling-orbit classifier obligation)**.** *Two carried* $\mathsf{ProjectiveSpace}^{\uparrow}$ *rows classify exactly when a certified nonzero* $\mathsf{Field}^{\uparrow}$ *scalar transports one guarded representative to the other by the* $\mathsf{VecSpace}^{\uparrow}$ *scalar action. The classifier is reflexive by scalar one, symmetric by the proof-indexed inverse of the scaling scalar, and transitive by field multiplication together with scalar-action associativity.*

Proof. For reflexivity, use the scalar one row from the field source and the $\mathsf{VecSpace}^{\uparrow}$ scalar unit law from Definition 47.21. Acting by one leaves the guarded representative in the same vector classifier row.

For symmetry, suppose a nonzero scalar transports representative v to representative w. The field source supplies the proof-indexed inverse for that scalar, and the vector-space scalar-action rows identify inverse action as the return route from w to v inside the same carried surface.

For transitivity, compose two certified nonzero scaling witnesses by field multiplication. The scalar-action associativity row in the vector-space certificate identifies the double action with action by the product scalar, so the resulting witness transports the first representative to the third. □

Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpaceSingletonScalingOrbitClassifier obligation`

Theorem 207.3 (ProjectiveSpaceUp ledger and exactness obligation)**.** *For* $\mathsf{ProjectiveSpace}^{\uparrow}$*, representative choice, nonzero scalar witnesses, and scaling-chain provenance remain ledger data. The visible exactness predicate is exactly the scaling-orbit classifier on punctured* $\mathsf{VecSpace}^{\uparrow}$ *rows: consumers may see the carrier of Definition 207.1 and the classifier of Definition 207.2, but they cannot inspect a chosen representative or a chosen scaling chain as public carrier data.*

Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpaceSingletonScalingOrbitClassifier ledger exactness obligation`
Lean4 ✓ `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpaceSingletonPuncturedCarrier action determinacy`

Proof. Project the certificate ledger into three hidden fields: the selected punctured representative, each nonzero scalar witness used to compare representatives, and the chain of scalar-action steps witnessing a composite comparison. These entries are consumed by the carrier and classifier rows but are not themselves part of the visible endpoint predicate.

Exactness is then read at the boundary. Soundness says every visible classification comes from a certified nonzero scalar-action route between punctured vector rows. Completeness says every such certified route repacks as the displayed projective classifier.

Hence the public surface exposes precisely punctured-carrier membership and scaling-orbit classification. All representative and route choices remain ledger provenance. □

207.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$**.**

Constructive story: A projective point begins as a vector-space history accepted by $\mathsf{VecSpace}^{\uparrow}$, then passes through the $\mathsf{Field}^{\uparrow}$ nonzero guard so that the zero representative is excluded before any scaling comparison is considered. The carrier therefore accepts punctured vector rows equipped with scalar-source, vector-source, and scalar-action ledger entries, while a zero row or an unguarded vector representative never becomes a projective endpoint. Two accepted representatives name the same projective point when a certified nonzero scalar transports one to the other by the $\mathsf{VecSpace}^{\uparrow}$ scalar action; scalar one gives reflexivity, the proof-indexed inverse gives symmetry, and field multiplication with scalar-action associativity gives transitivity. The source covered by the certificate is the punctured vector carrier, the pattern is the scaling-orbit presentation with nonzero scalar witnesses and action chains, and the ledger keeps representative choice, scalar witnesses, and scaling-chain provenance as evidence rather than as public carrier data. The five name-certificate obligations are supplied by an accepted punctured representative for nonemptiness, by the scaling classifier for reflexive, symmetric, and transitive comparison, and by scalar-action transport for closure of the punctured carrier under classified replacements. A history h lies in the signature gap of a $\mathsf{ProjectiveSpace}^{\uparrow}$ token p when the source-domain accepts h as a guarded vector representative, the signature relation pairs it with an answer representative and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is a nonzero vector row in the same certified scaling orbit as the representative encoded by p. The paper bridge reads this surface as the ordinary punctured-vector modulo nonzero-scaling presentation and reads back through the same representative and scalar-action ledger; the chapter registers it under the name $\mathsf{ProjectiveSpace}^{\uparrow}$, without closing full projective geometry or bridgeChecked verification.

Theory closure: TheoryGate(NameCert$_{\mathsf{ProjectiveSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public ProjectiveSpace$^{\uparrow}$ surface exports the punctured carrier, scaling-orbit classifier, ledger provenance, and exactness boundary from Theorems 207.1, 207.2, and 207.3.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProjectiveSpace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ProjectiveSpaceUp.ProjectiveSpaceSingletonScalingOrbitClassifier\_ledger\_exactness\_ob`

Bridge status: paperBridge

Not claimed: This chapter does not close full projective geometry or bridgeChecked verification.

Upgrade path: The paper-side theory axis is at the top grade; bridgeChecked verification remains a separate verification-axis obligation.

Chapter 208

A Concrete Naming Certificate for $\mathsf{ConvexSet}^{\uparrow}$

The convex-set interface $\mathsf{ConvexSet}^{\uparrow}$ packages a subset of a vector-space carrier closed under affine combinations with nonnegative coefficients summing to one. It depends on $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47) and $\mathsf{NameCert}_{\mathsf{AffineSpace}^{\uparrow}}$ (chapter 206).

208.1 Carrier definition

208.2 The certificate

Definition 208.1 (ConvexSet binary affine-combination row)**.** *For a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate over a scalar* $\mathsf{Field}^{\uparrow}(F)$ *and a vector carrier from* $\mathsf{VecSpace}^{\uparrow}(F, V)$*, write* C *for the convex carrier,* $\sim_F$ *for the scalar classifier,* $+_F$ *for scalar addition,* 1_F *for the scalar unit,* NonNeg_F *for the nonnegative-scalar predicate,* $\odot$ *for the scalar action, and* $+_V$ *for vector addition. The binary affine-combination row is the certificate field saying that, for all* $a, b : F$ *and all carried points* $x, y : V$*,*

$$C(x) \wedge C(y) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F \Longrightarrow C\big((a \odot x) +_V (b \odot y)\big).$$

This is the two-point instance of the convex closure advertised in the $\mathsf{ConvexSet}^{\uparrow}$ *carrier description, using the scalar and action data supplied by the* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{AffineSpace}^{\uparrow}$ *dependencies.*

Lean4 (definition) `BEDC.Derived.ConvexSetUp.ConvexSetBinaryAffineCombinationRow`

Theorem 208.2 (ConvexSet midpoint closure from affine-combination row)**.** *Let a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate be given with notation as in Definition 208.1. If* x *and* y *are carried by the convex carrier,* h *is a nonnegative scalar, and the scalar sum* $h +_F h$ *is classified as the scalar unit,*

$$C(x) \wedge C(y) \wedge \mathsf{NonNeg}_F(h) \wedge (h +_F h) \sim_F 1_F,$$

then the affine midpoint endpoint is carried by the same convex carrier:

$$C\big((h \odot x) +_V (h \odot y)\big).$$

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetSingletonAffineSpine midpoint closure`

Proof. Apply the binary affine-combination row of Definition 208.1 with both scalar coefficients equal to h. The two point-carrier hypotheses supply $C(x)$ and $C(y)$. The single nonnegative hypothesis supplies both coefficient nonnegativity clauses, since the row asks separately for $\mathsf{NonNeg}_F(h)$ on the left coefficient and on the right coefficient. The displayed classifier hypothesis is exactly the coefficient-sum clause $(h +_F h) \sim_F 1_F$. The row therefore returns $C\big((h \odot x) +_V (h \odot y)\big)$, which is the claimed midpoint carrier closure. □

Theorem 208.3 (ConvexSet affine-combination row order symmetry)**.** *Let a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate be given over a scalar* $\mathsf{Field}^{\uparrow}(F)$ *and a vector carrier from* $\mathsf{VecSpace}^{\uparrow}(F, V)$*, with notation as in Definition 208.1. For scalars* $a, b : F$ *and points* $x, y : V$*, assume the scalar additive commutativity row inherited from the scalar field certificate supplies the comparison*

$$(b +_F a) \sim_F (a +_F b).$$

If

$$C(x) \wedge C(y) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F,$$

then the swapped affine endpoint is carried by the same convex carrier:

$$C\big((b \odot y) +_V (a \odot x)\big).$$

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetSingletonAffineSpine order symmetric`

Proof. The scalar part of the field certificate retains the additive abelian-group rows listed through Definition 43.26 and Definition 42.8, so the displayed scalar additive commutativity comparison is an available classifier edge for these two coefficients. The scalar classifier is a naming-certificate classifier, and Definition 9.8 composes this comparison with the coefficient-sum hypothesis $(a +_F b) \sim_F 1_F$. Hence

$$(b +_F a) \sim_F 1_F.$$

Apply the binary affine-combination row of Definition 208.1 with left coefficient b, right coefficient a, left point y, and right point x. The carrier hypotheses provide $C(y)$ and $C(x)$ in the required order. The nonnegativity hypotheses provide $\mathsf{NonNeg}_F(b)$ and $\mathsf{NonNeg}_F(a)$ in the required order. The transitive scalar comparison above is exactly the reversed coefficient-sum premise. The row therefore returns $C\big((b \odot y) +_V (a \odot x)\big)$. □

Definition 208.4 (ConvexSet finite affine-spine data)**.** *Let a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate be given with notation as in Definition 208.1. Also write* 0_F *for the scalar zero and* $\cdot_F$ *for scalar multiplication. For a finite scalar spine* $\lambda : \mathsf{ListCarrier}(F)$*, write* $\mathsf{sum}_F(\lambda)$ *for the finite recursive scalar sum*

$$\mathsf{sum}_F(\mathsf{nil}) = 0_F, \qquad \mathsf{sum}_F(\mathsf{cons}(a, \lambda')) = a +_F \mathsf{sum}_F(\lambda').$$

For a finite point spine $xs : \mathsf{ListCarrier}(V)$*, a finite scalar-weight spine* $\lambda : \mathsf{ListCarrier}(F)$*, and an endpoint* $z : V$*, define* $\mathsf{ConvAffSpine}_C(xs, \lambda, z)$ *to mean the conjunction of the following two data.*

First, the global finite-weight readback says that the point spine and weight spine have the same finite positions, every displayed point is carried, every displayed scalar weight is nonnegative, and the finite scalar sum is classified as the scalar unit:

$$\exists \theta : \mathsf{Pos}(xs) \cong \mathsf{Pos}(\lambda).\ \Big(\forall i : \mathsf{Pos}(xs).\ C(\mathsf{entry}_{xs}(i)) \wedge \mathsf{NonNeg}_F(\mathsf{entry}_\lambda(\theta(i)))\Big) \wedge \mathsf{sum}_F(\lambda) \sim_F 1_F.$$

Second, the endpoint z *is obtained from those leaves by the least local binary bracketing relation* $\mathsf{Bracket}_C(xs, \lambda, z)$ *generated by the following clauses:*

1. *singleton: if* $C(x)$, $\mathsf{NonNeg}_F(a)$, *and* $a \sim_F 1_F$, *then*

$$\mathsf{Bracket}_C(\mathsf{cons}(x, \mathsf{nil}), \mathsf{cons}(a, \mathsf{nil}), x);$$

2. *binary extension: if* $C(x)$, $\mathsf{NonNeg}_F(a)$, $\mathsf{NonNeg}_F(b)$, $(a +_F b) \sim_F 1_F$, *and* $\mathsf{Bracket}_C(ys, \mu, y)$, *then*

$$\mathsf{Bracket}_C(\mathsf{cons}(x, ys), \mathsf{cons}(a, b \cdot_F \mu), (a \odot x) +_V (b \odot y)),$$

where $b \cdot_F \mu$ *denotes pointwise scalar multiplication of the tail weight spine.*

Thus $\mathsf{ConvAffSpine}_C(xs, \lambda, z)$ *records both the visible finite unit-sum weight spine and the recursively bracketed affine endpoint built from binary affine-combination rows.*

Lean4 (definition) `BEDC.Derived.ConvexSetUp.ConvexSetSingletonAffineSpine`

Theorem 208.5 (ConvexSet finite affine-spine closure)**.** *Let a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate be given over a scalar* $\mathsf{Field}^{\uparrow}(F)$ *and a vector carrier from* $\mathsf{VecSpace}^{\uparrow}(F, V)$, *with notation as in Definition 208.1. For every finite point spine* $xs : \mathsf{ListCarrier}(V)$, *finite scalar-weight spine* $\lambda : \mathsf{ListCarrier}(F)$, *and endpoint* $z : V$, *if*

$$\mathsf{ConvAffSpine}_C(xs, \lambda, z)$$

holds, then the recursively bracketed affine endpoint is carried by the convex carrier:

$$C(z).$$

Proof. Unpack Definition 208.4. The finite-weight readback supplies the advertised carried-point, nonnegative-weight, and unit-sum scalar spine. The carrier conclusion is proved by induction on the local bracketing witness $\mathsf{Bracket}_C(xs, \lambda, z)$.

In the singleton clause, the endpoint is the displayed point x, and the clause itself supplies $C(x)$. Hence the endpoint is carried.

In the binary-extension clause, the induction hypothesis applied to the tail bracketing witness gives $C(y)$. The current clause supplies $C(x)$, $\mathsf{NonNeg}_F(a)$, $\mathsf{NonNeg}_F(b)$, and the local unit-sum comparison $(a +_F b) \sim_F 1_F$. Applying the binary affine-combination row of Definition 208.1 to the carried points x, y and coefficients a, b gives

$$C((a \odot x) +_V (b \odot y)).$$

This endpoint is exactly the endpoint named by the binary-extension clause. The induction therefore covers every recursively bracketed finite affine-spine witness, so $C(z)$ holds. □

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetSingletonAffineSpine closure`

Definition 208.6 (ConvexSet pointwise intersection carrier)**.** *Let two* $\mathsf{ConvexSet}^{\uparrow}$ *certificates be given over the same scalar* $\mathsf{Field}^{\uparrow}(F)$ *and the same vector carrier from* $\mathsf{VecSpace}^{\uparrow}(F, V)$, *using the same scalar classifier* $\sim_F$, *scalar addition* $+_F$, *scalar unit* 1_F, *nonnegative predicate* NonNeg_F, *scalar action* $\odot$, *and vector addition* $+_V$. *Write* $C, D : V \to \mathsf{Prop}$ *for their convex carriers. The pointwise intersection carrier* $C \cap_{\mathsf{ConvexSet}^{\uparrow}} D$ *is the predicate on* V *defined by*

$$(C \cap_{\mathsf{ConvexSet}^{\uparrow}} D)(z) \iff C(z) \wedge D(z).$$

Lean4 (definition) `BEDC.Derived.ConvexSetUp.ConvexSetPointwiseIntersection`

Theorem 208.7 (ConvexSet pointwise intersection affine-combination closure)**.** *Let two* $\mathsf{ConvexSet}^{\uparrow}$ *certificates be given as in Definition 208.6. Assume each certificate supplies the binary affine-combination row of Definition 208.1 for its own convex carrier, and write* $K := C \cap_{\mathsf{ConvexSet}^{\uparrow}} D$*. For all* $a, b : F$ *and* $x, y : V$*, if*

$$K(x) \wedge K(y) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F,$$

then

$$K\big((a \odot x) +_V (b \odot y)\big).$$

Thus the pointwise intersection satisfies the same nonnegative unit-sum binary affine-combination closure row over the common $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *data.*

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetPointwiseIntersection affine combination closure`

Proof. Unfold Definition 208.6. From $K(x)$ obtain $C(x)$ and $D(x)$, and from $K(y)$ obtain $C(y)$ and $D(y)$. The scalar hypotheses are shared by the two input certificates because the scalar classifier, scalar addition, scalar unit, nonnegative predicate, scalar action, and vector addition are the common data named in the definition.

Apply the binary affine-combination row of Definition 208.1 for the carrier C to the coefficients a, b and the carried points x, y. The projected carrier hypotheses, the two nonnegativity hypotheses, and the unit-sum classifier hypothesis give

$$C\big((a \odot x) +_V (b \odot y)\big).$$

Apply the same row for the carrier D with the same coefficients and points. The other projected carrier hypotheses and the same scalar hypotheses give

$$D\big((a \odot x) +_V (b \odot y)\big).$$

Pairing these two carrier conclusions and folding Definition 208.6 gives

$$K\big((a \odot x) +_V (b \odot y)\big),$$

which is exactly the displayed binary affine-combination closure row for the intersection carrier. □

Definition 208.8 (ConvexSet classifier image carrier)**.** *Let* F *carry a* $\mathsf{Field}^{\uparrow}(F)$ *certificate, let* V *and* W *carry* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *and* $\mathsf{VecSpace}^{\uparrow}(F, W)$ *certificates over the same scalar source, and let* $C : V \to \mathsf{Prop}$ *be the convex carrier of a* $\mathsf{ConvexSet}^{\uparrow}$ *certificate on* V*. Write* $\sim_W$ *for the codomain vector classifier, and let* $f : V \to W$ *be a carried* $\mathsf{LinMap}^{\uparrow}(V, W)$ *map. The classifier image of* C *along* f *is the predicate* $f[C]_{\sim_W} : W \to \mathsf{Prop}$ *defined by*

$$f[C]_{\sim_W}(w) \iff \exists x : V.\ C(x) \wedge f(x) \sim_W w.$$

The witness records a source point carried by the convex carrier and a codomain classifier comparison from its f*-output to the displayed codomain endpoint.*

Lean4 (definition) `BEDC.Derived.ConvexSetUp.ConvexSetLinearImageCarrier`

Theorem 208.9 (ConvexSet linear image affine-combination closure)**.** *Let the data be as in Definition 208.8. Write $\sim_F$, $+_F$, 1_F, and NonNeg_F for the scalar classifier, scalar addition, scalar unit, and nonnegative predicate. Write $+_V$ and $\odot_V$ for the source vector addition and scalar action, and write $+_W$, $\odot_W$, and $\sim_W$ for the codomain vector addition, scalar action, and classifier. Assume the source $\mathsf{ConvexSet}^{\uparrow}$ certificate supplies the binary affine-combination row of Definition 208.1. For carried scalars $a, b : F$ and codomain endpoints $u, v : W$, if*

$$f[C]_{\sim_W}(u) \wedge f[C]_{\sim_W}(v) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F,$$

then

$$f[C]_{\sim_W}((a \odot_W u) +_W (b \odot_W v)).$$

Thus the pointwise classifier image $f[C]_{\sim_W}$ satisfies the same binary nonnegative unit-sum affine-combination closure row in the codomain vector-space carrier.

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetLinearImageCarrier affine combination closure`

Proof. Unpack Definition 208.8 for the two image hypotheses. Choose source points $x, y : V$ with

$$C(x), \qquad C(y), \qquad f(x) \sim_W u, \qquad f(y) \sim_W v.$$

Because the $\mathsf{ConvexSet}^{\uparrow}$ carrier is a subset of the source $\mathsf{VecSpace}^{\uparrow}$ carrier, x and y are carried source inputs. Set

$$s := (a \odot_V x) +_V (b \odot_V y).$$

The nonnegativity hypotheses and the scalar unit-sum comparison are exactly the coefficient hypotheses required by Definition 208.1. Applying that row to x, y and a, b gives $C(s)$.

It remains to compare the f-output of this source witness with the displayed codomain affine combination. The carried $\mathsf{LinMap}^{\uparrow}(V, W)$ map supplies the function-with-law rows of Definition 48.1; read through Definition 47.1 and Definition 48.59, these include output closure, additive preservation, and scalar compatibility. Source scalar-action closure and source additive closure make the two summands of s and s carried source inputs. Additive preservation gives

$$f(s) \sim_W f(a \odot_V x) +_W f(b \odot_V y).$$

Scalar compatibility gives

$$f(a \odot_V x) \sim_W a \odot_W f(x), \qquad f(b \odot_V y) \sim_W b \odot_W f(y).$$

The target additive congruence row from Definition 46.18 transports these two comparisons to

$$f(a \odot_V x) +_W f(b \odot_V y) \sim_W (a \odot_W f(x)) +_W (b \odot_W f(y)).$$

The image comparisons $f(x) \sim_W u$ and $f(y) \sim_W v$ are codomain classifier comparisons. Scalar-action congruence in the codomain vector-space module, using scalar reflexivity on the carried scalars a and b, gives

$$a \odot_W f(x) \sim_W a \odot_W u, \qquad b \odot_W f(y) \sim_W b \odot_W v.$$

A second use of target additive congruence gives

$$(a \odot_W f(x)) +_W (b \odot_W f(y)) \sim_W (a \odot_W u) +_W (b \odot_W v).$$

Classifier transitivity, as packaged by Definition 9.8, composes the three codomain comparisons and yields

$$f(s) \sim_W (a \odot_W u) +_W (b \odot_W v).$$

The source witness s, the convex-carrier proof $C(s)$, and this codomain classifier comparison repack by Definition 208.8 as

$$f[C]_{\sim_W}((a \odot_W u) +_W (b \odot_W v)).$$

□

Theorem 208.10 (ConvexSet linear image finite affine-spine closure)**.** *Let the data be as in Definition 208.8. Write $\sim_F$, $+_F$, 1_F, and NonNeg_F for the scalar classifier, scalar addition, scalar unit, and nonnegative predicate. Write $+_W$, $\odot_W$, and $\sim_W$ for the codomain vector addition, scalar action, and classifier. Assume the source $\mathsf{ConvexSet}^{\uparrow}$ certificate supplies the binary affine-combination row of Definition 208.1. For every finite codomain point spine $ys : \mathsf{ListCarrier}(W)$, finite scalar-weight spine $\lambda : \mathsf{ListCarrier}(F)$, and endpoint $z : W$, if*

$$\mathsf{ConvAffSpine}_{f[C]_{\sim_W}}(ys, \lambda, z)$$

holds in the sense of Definition 208.4 with carrier predicate $f[C]_{\sim_W}$, then

$$f[C]_{\sim_W}(z).$$

Thus the classifier image carrier is closed under every finite affine spine whose weights are nonnegative and whose finite scalar sum is classified as 1_F.

Lean4 ✓ `BEDC.Derived.ConvexSetUp.ConvexSetLinearImage finite affine spine closure`

Proof. By Definition 208.9, the image carrier $f[C]_{\sim_W}$ satisfies the codomain binary affine-combination row

$$f[C]_{\sim_W}(u) \wedge f[C]_{\sim_W}(v) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F \Longrightarrow f[C]_{\sim_W}((a \odot_W u) +_W (b \odot_W v)).$$

This row is obtained from the source convex row and the carried $\mathsf{LinMap}^{\uparrow}(V, W)$ data, whose additivity, scalar compatibility, and codomain congruence rows are read through Definition 48.1, Definition 48.59, and Definition 46.18.

Unpack the displayed hypothesis using Definition 208.4. The finite readback gives the common finite position data, nonnegative weights, and unit-sum scalar comparison. The endpoint carrier conclusion is proved by induction on the local bracketing witness $\mathsf{Bracket}_{f[C]_{\sim_W}}(ys, \lambda, z)$.

In the singleton clause, the endpoint is the displayed codomain point u, and the clause itself supplies $f[C]_{\sim_W}(u)$. Hence the endpoint lies in the image carrier.

In the binary-extension clause, the clause supplies $f[C]_{\sim_W}(u)$, the coefficient hypotheses $\mathsf{NonNeg}_F(a)$ and $\mathsf{NonNeg}_F(b)$, the local unit-sum comparison $(a +_F b) \sim_F 1_F$, and a tail bracketing witness with endpoint v. The induction hypothesis applied to that tail witness gives $f[C]_{\sim_W}(v)$. Applying the binary image row above to u, v and a, b gives

$$f[C]_{\sim_W}((a \odot_W u) +_W (b \odot_W v)),$$

which is exactly the endpoint named by the binary-extension clause. Therefore every local bracketing witness over $f[C]_{\sim_W}$ has its endpoint in the same classifier image carrier, and the original endpoint z satisfies $f[C]_{\sim_W}(z)$. □

Definition 208.11 (ConvexSet linear preimage carrier)**.** *Let F carry a $\mathsf{Field}^{\uparrow}(F)$ certificate, let V and W carry $\mathsf{VecSpace}^{\uparrow}(F,V)$ and $\mathsf{VecSpace}^{\uparrow}(F,W)$ certificates over the same scalar source, and let $D : W \to \mathsf{Prop}$ be the convex carrier of a $\mathsf{ConvexSet}^{\uparrow}$ certificate on W. Let $f : V \to W$ be a carried $\mathsf{LinMap}^{\uparrow}(V,W)$ map. The source-side preimage of D along f is the predicate $f^{-1}[D] : V \to \mathsf{Prop}$ on carried source points defined by*

$$f^{-1}[D](x) \iff D(f(x)).$$

Thus membership in the preimage carrier is read by evaluating the carried linear map and testing the codomain convex carrier at the displayed output.

Theorem 208.12 (ConvexSet linear preimage affine-combination closure)**.** *Let the data be as in Definition 208.11. Write $\sim_F$, $+_F$, 1_F, and NonNeg_F for the scalar classifier, scalar addition, scalar unit, and nonnegative predicate. Write $+_V$ and $\odot_V$ for the source vector addition and scalar action, and write $+_W$, $\odot_W$, and $\sim_W$ for the codomain vector addition, scalar action, and classifier. Assume the codomain $\mathsf{ConvexSet}^{\uparrow}$ certificate supplies the binary affine-combination row of Definition 208.1. For carried scalars $a, b : F$ and carried source points $x, y : V$, if*

$$f^{-1}[D](x) \wedge f^{-1}[D](y) \wedge \mathsf{NonNeg}_F(a) \wedge \mathsf{NonNeg}_F(b) \wedge (a +_F b) \sim_F 1_F,$$

then

$$f^{-1}[D]\big((a \odot_V x) +_V (b \odot_V y)\big).$$

Thus the source-side preimage carrier satisfies the same binary nonnegative unit-sum affine-combination closure row over the source vector-space carrier.

Proof. Unfold Definition 208.11 for the two preimage hypotheses. They are exactly

$$D(f(x)), \qquad D(f(y)).$$

Since the codomain convex carrier is a subset of the codomain $\mathsf{VecSpace}^{\uparrow}$ carrier, these hypotheses make $f(x)$ and $f(y)$ carried codomain inputs. Applying the codomain binary affine-combination row of Definition 208.1 to the points $f(x), f(y)$ and the coefficients a, b gives

$$D\big((a \odot_W f(x)) +_W (b \odot_W f(y))\big).$$

Set

$$s := (a \odot_V x) +_V (b \odot_V y).$$

Because x and y are carried source inputs, source scalar-action closure and source additive closure make the two summands of s and the endpoint s itself carried in V. The carried $\mathsf{LinMap}^{\uparrow}(V,W)$ map supplies additive preservation and scalar compatibility through Definition 48.1 and Definition 48.59. Therefore

$$f(s) \sim_W f(a \odot_V x) +_W f(b \odot_V y)$$

and

$$f(a \odot_V x) \sim_W a \odot_W f(x), \qquad f(b \odot_V y) \sim_W b \odot_W f(y).$$

The target additive congruence row from Definition 46.18 transports these two comparisons to

$$f(a \odot_V x) +_W f(b \odot_V y) \sim_W (a \odot_W f(x)) +_W (b \odot_W f(y)).$$

Classifier transitivity, as packaged by Definition 9.8, composes this with the additive-preservation comparison and yields

$$f(s) \sim_W (a \odot_W f(x)) +_W (b \odot_W f(y)).$$

Classifier symmetry from Definition 9.7 reverses this comparison, so

$$(a \odot_W f(x)) +_W (b \odot_W f(y)) \sim_W f(s).$$

The codomain convex carrier inherits carrier transport along its classifier from the naming-certificate core in Definition 9.7. Transporting the codomain carrier witness above across this reversed comparison gives

$$D(f(s)).$$

Substituting the definition of s and folding Definition 208.11 gives

$$f^{-1}[D]((a \odot_V x) +_V (b \odot_V y)),$$

which is the claimed source-side affine-combination closure. □

Theorem 208.13 (ConvexSet finite half-space intersection carrier)**.** *Let F carry the shared ordered-field surface, let V carry a* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *certificate, and let a finite family of carried affine half-spaces be presented by rows*

$$H_k(x) \iff \ell_k(x) \preceq_F r_k \qquad (k \in K),$$

where each ℓ_k is read through a carried affine or linear map and each comparison uses the $\mathsf{PositiveCone}^{\uparrow}$ *nonnegative classifier over F. Then the finite intersection*

$$\mathsf{Half}(x) \iff \forall k \in K.\ H_k(x)$$

is admitted as a $\mathsf{ConvexSet}^{\uparrow}$ *carrier. The same finite row supplies the face packet consumed by* $\mathsf{Polytope}^{\uparrow}$*: every face witness is a subfamily of the displayed half-space rows together with its positive-cone comparison ledger.*

Proof. Proceed by induction on the finite half-space row carrier K. In the empty case the carrier is the whole ambient carried vector-space surface, whose affine-combination closure is inherited from the $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{AffineSpace}^{\uparrow}$ rows used by the chapter's root convex carrier.

For the step, write the intersection as a tail carrier C together with one displayed half-space H_k. By the induction hypothesis, C is a $\mathsf{ConvexSet}^{\uparrow}$ carrier. The half-space row is convex because affine or linear readback transports binary affine combinations to scalar affine combinations, and the $\mathsf{PositiveCone}^{\uparrow}$ comparison ledger preserves the nonnegative ordered-field inequality under those combinations.

Apply the pointwise intersection closure of Definition 208.7 to C and H_k. This admits the enlarged finite intersection as a $\mathsf{ConvexSet}^{\uparrow}$ carrier, with no separation theorem or compactness row.

The face packet is read from the same finite ledger. A $\mathsf{Polytope}^{\uparrow}$ consumer selects a subfamily of active half-space rows, and the positive-cone classifier transports the corresponding comparison witnesses. Thus the convex carrier and the polytope face packet share the displayed half-space rows rather than appealing to ambient polyhedral geometry. □

Theorem 208.14 (ConvexSet pointwise intersection carrier commutativity)**.** *Let two* $\mathsf{ConvexSet}^{\uparrow}$ *certificates be given as in Definition 208.6. Write* $C, D : V \to \mathsf{Prop}$ *for their convex carriers, and write*

$$K_{CD} := C \cap_{\mathsf{ConvexSet}^{\uparrow}} D, \qquad K_{DC} := D \cap_{\mathsf{ConvexSet}^{\uparrow}} C.$$

For every endpoint $z : V$*, the two pointwise intersection carriers agree:*

$$K_{CD}(z) \Longleftrightarrow K_{DC}(z).$$

Equivalently, a carrier witness for $C \cap_{\mathsf{ConvexSet}^{\uparrow}} D$ *at* z *is exactly a pair consisting of a* $C(z)$ *witness and a* $D(z)$ *witness, and the corresponding carrier witness for* $D \cap_{\mathsf{ConvexSet}^{\uparrow}} C$ *is obtained by interchanging those two components. The scalar classifier* $\sim_F$*, scalar addition* $+_F$*, scalar unit* 1_F*, nonnegative predicate* NonNeg_F*, scalar action* $\odot$*, and vector addition* $+_V$ *are the common rows named in Definition 208.6, so this interchange does not alter any displayed scalar, addition, or affine-combination row.*

Proof. Fix an endpoint $z : V$. Unfold Definition 208.6 for the ordered pair of carriers (C, D). The hypothesis $K_{CD}(z)$ is precisely

$$C(z) \wedge D(z).$$

Projecting its two components and pairing them in the opposite order gives

$$D(z) \wedge C(z),$$

which is exactly $K_{DC}(z)$ after unfolding the same definition for the ordered pair (D, C).

Conversely, unfold the definition for (D, C). The hypothesis $K_{DC}(z)$ is precisely

$$D(z) \wedge C(z).$$

Projecting its two components and pairing them in the opposite order gives

$$C(z) \wedge D(z),$$

which is exactly $K_{CD}(z)$ after folding the definition for (C, D). The two folds use the same ambient $\mathsf{Field}^{\uparrow}(F)$ and $\mathsf{VecSpace}^{\uparrow}(F, V)$ data named in Definition 208.6; only the two carrier-support components at the fixed endpoint are swapped. □

Theorem 208.15 (ConvexSet pointwise intersection inclusion meet)**.** *Let two* $\mathsf{ConvexSet}^{\uparrow}$ *certificates be given as in Definition 208.6. Write* $C, D : V \to \mathsf{Prop}$ *for their convex carriers and write*

$$K := C \cap_{\mathsf{ConvexSet}^{\uparrow}} D.$$

On convex carriers over the same ambient $\mathsf{Field}^{\uparrow}(F)$ *and* $\mathsf{VecSpace}^{\uparrow}(F, V)$ *data, write*

$$E \preceq_{\mathsf{ConvexSet}^{\uparrow}} G \Longleftrightarrow \forall z : V.\ E(z) \to G(z)$$

for pointwise inclusion. Then K *is the greatest lower bound of* C *and* D *in this inclusion preorder:*

$$K \preceq_{\mathsf{ConvexSet}^{\uparrow}} C, \qquad K \preceq_{\mathsf{ConvexSet}^{\uparrow}} D,$$

and, for every third $\mathsf{ConvexSet}^{\uparrow}$ *convex carrier* $E : V \to \mathsf{Prop}$ *over the same ambient data,*

$$(E \preceq_{\mathsf{ConvexSet}^{\uparrow}} C) \wedge (E \preceq_{\mathsf{ConvexSet}^{\uparrow}} D) \Longrightarrow E \preceq_{\mathsf{ConvexSet}^{\uparrow}} K.$$

The carrier K *is therefore the meet of* C *and* D *among convex carriers ordered by inclusion, with its convex closure supplied by Definition 208.7.*

Proof. The pointwise intersection carrier is the predicate specified in Definition 208.6:

$$K(z) \iff C(z) \wedge D(z).$$

For the first projection bound, fix $z : V$ and assume $K(z)$. Unfolding the definition gives $C(z) \wedge D(z)$, and projecting the first component gives $C(z)$. Hence $K \preceq_{\mathsf{ConvexSet}^{\uparrow}} C$.

For the second projection bound, the same unfolding of $K(z)$ gives $C(z) \wedge D(z)$, and projecting the second component gives $D(z)$. Hence $K \preceq_{\mathsf{ConvexSet}^{\uparrow}} D$.

It remains to prove the universal lower-bound property. Let $E : V \to \mathsf{Prop}$ be the convex carrier of a third $\mathsf{ConvexSet}^{\uparrow}$ certificate over the same ambient data, and assume

$$E \preceq_{\mathsf{ConvexSet}^{\uparrow}} C, \qquad E \preceq_{\mathsf{ConvexSet}^{\uparrow}} D.$$

To prove $E \preceq_{\mathsf{ConvexSet}^{\uparrow}} K$, fix $z : V$ and assume $E(z)$. The first inclusion sends this witness to $C(z)$, and the second inclusion sends the same witness to $D(z)$. Pairing the two conclusions gives

$$C(z) \wedge D(z),$$

which is exactly $K(z)$ by Definition 208.6. Therefore every common lower bound E factors through K, and the two projection bounds make K the greatest lower bound of C and D for pointwise inclusion. □

Theorem 208.16 (ConvexSetUp public NameCert export)**.** *The carrier predicate, binary affine-combination row, midpoint and order-symmetry rows, finite affine spine, pointwise intersection, linear image, and linear preimage rows form the public* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *export over the displayed* $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* Pkg *surfaces. In this export, the public carrier accepts exactly the endpoints presented by the convex predicate, finite affine bracketing, intersection witnesses, linear image witnesses, or linear preimage witnesses; the public classifier is the ambient vector classifier transported through the carried package rows; and the public ledger records the visible points, weights, scalar comparisons, affine endpoints, linear-map rows, and package provenance.*

Proof. The carrier and binary affine-combination rows are the displayed $\mathsf{ConvexSet}^{\uparrow}$ certificate rows. The midpoint and order-symmetry rows are specialisations of the same binary affine row, with the scalar hypotheses and ambient classifier comparisons carried by the $\mathsf{Field}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ surfaces.

The finite affine spine is public because Definition 208.10 proves that any finite codomain spine over a carried linear image repacks into the same image carrier. Pointwise intersection is public by Definition 208.7, Definition 208.14, and Definition 208.15; these rows keep the two component carrier witnesses and no external set former.

Linear images are public by Definition 208.9 and the finite spine theorem above. Linear preimages are public by Definition 208.12; the source endpoint is accepted exactly when its carried image is accepted by the codomain convex carrier.

Together these rows expose the requested public $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ surface. The export contains no separation theorem, compactness theorem, finite-dimensional theory, or standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$**.**

Constructive story: A convex set here is generated inside already certified affine and vector-space data: start with carried points, attach nonnegative scalar histories from the field source whose finite sum is classified as the scalar unit, and build the endpoint by repeated affine-combination rows. The carrier accepts endpoints that are presented by this

convex predicate, by finite affine spines, by pointwise intersections, by classifier images along carried linear maps, or by classifier preimages along carried linear maps; data missing the affine source, scalar nonnegativity, unit-sum comparison, or carried linear-map witness remain outside the public surface. Two displayed endpoints are compared only through the ambient vector classifiers and the witness rows attached to the source convex predicate, so linear images use a source point plus a codomain classifier comparison rather than an external image set. The source is the displayed convex carrier predicate over $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$, the pattern records binary affine closure, midpoint closure, order symmetry, finite bracketing, intersection closure, linear image closure, and linear preimage closure, and the ledger keeps the raw points, weights, scalar comparisons, affine endpoints, linear-map outputs, and package rows visible. The name-certificate obligations at this public level are read from the accepted carrier rows: inhabited examples supply the carrier witness, the inherited classifiers give reflexivity, symmetry, and transitivity, and the closure theorems repack any classified affine combination, intersection member, image endpoint, or preimage endpoint back into the carrier. A history h lies in the signature gap of a $\mathsf{ConvexSet}^{\uparrow}$ token p when the source-domain accepts the displayed convex packet, the signature relation pairs it with an answer carrier row and evidence ledger, and the token-introduction relation names that answer as p; concretely, every affine-combination, finite-spine, image, and preimage endpoint recorded by p is carried by the same visible predicate. The chapter registers this affine-combination construction under the name $\mathsf{ConvexSet}^{\uparrow}$; it supplies a public convex surface, with no separation theorem, compactness theorem, finite-dimensional theory, or standard bridge introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public boundary is the carrier predicate, binary affine row, midpoint and order-symmetry rows, finite affine spine, pointwise intersection, linear image, linear preimage, and dependency inventory bound to $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, and Pkg.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ConvexSetUp.ConvexSetLinearImage\_finite\_affine\_spine\_closure`

Bridge status: none

Not claimed: This chapter does not close finite-dimensional convexity theory, separation theorem, compactness theorem, or standard bridge.

Upgrade path: The paper-axis treatment is mature; remaining work is bridge-status or formal-axis coordination.

Chapter 209

A Concrete Naming Certificate for $\mathsf{Polytope}^{\uparrow}$

The polytope interface $\mathsf{Polytope}^{\uparrow}$ packages a bounded convex set in a finite-dimensional vector-space carrier presented as the intersection of finitely many half-spaces, with vertex, edge, and face combinatorial structure. It depends on $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ (chapter 208) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

209.1 Carrier definition

Definition 209.1 (Polytope BHist face packet)**.** *A* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet is a finite certificate tuple*

$$\mathcal{P} = (\eta_C, \eta_F, H, V, E, \Phi, L, \pi)$$

over BHist *in which* $\eta_C : \mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *supplies the convex carrier rows,* $\eta_F : \mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *supplies the finite-index rows,* H *is the finite list of half-space rows,* V *is the finite vertex row ledger,* E *is the finite edge row ledger,* Φ *is the finite face row ledger,* L *is the collection of* Cont *ledgers connecting half-space, vertex, edge, and face rows, and* $\pi : \mathsf{Pkg}$ *records the provenance of those rows.*

The carrier accepts $\mathcal{P}$ *exactly when every half-space row is scoped by* η_C*, every vertex, edge, and face index is scoped by* η_F*, every incidence row is a* BHist *row transported by* hsame*, and each displayed face is generated by the finite half-space ledger together with the finite vertex and edge incidence ledgers named in* L*.*

Lean4 (definition) `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket`

Theorem 209.2 (Polytope half-space and face carrier stability)**.** *Let* $\mathcal{P}$ *be a* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet. If each half-space row, vertex row, edge row, face row, and incidence ledger row of* $\mathcal{P}$ *is transported along an* hsame *comparison to the corresponding row of a packet* $\mathcal{P}'$*, then* $\mathcal{P}'$ *is again a* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet. The transported packet keeps the same* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *dependency witnesses, the same finite* Cont *ledger shape, and the same* Pkg *provenance rows.*

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket halfspace face carrier stability`

Proof. Project the packet into its half-space, vertex, edge, face, incidence, dependency, ledger, and provenance components. Transport the half-space rows by the hsame stability supplied by the

convex-set source certificate, and transport the finite vertex, edge, and face indices by the finite-set source certificate.

Next transport the displayed incidence and containment rows componentwise. The Cont ledgers are finite, so each source, step, and result row is replaced by its transported hsame mate while the ledger order and package provenance are left unchanged.

Repack the transported rows into the tuple defining Definition 209.1. The acceptance clauses are exactly the transported clauses, so the target tuple is a carried $\mathsf{Polytope}^{\uparrow}$ packet. □

209.2 The certificate

Theorem 209.3 (Polytope vertex, edge, and face ledger readback)**.** *Every* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet exposes finite readback rows for its vertices, edges, and faces. More precisely, the packet determines a finite* BHist *vertex ledger, a finite edge-incidence ledger, and a finite face-incidence ledger, each indexed through the* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *source and each connected to the packet carrier through the displayed* Cont *rows.*

Proof. Unpack the packet of Definition 209.1. The components V, E, and Φ are already finite ledgers, and the finite-set source certificate records their index rows as $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ data.

For each ledger row, read the corresponding Cont entry from L. The entry names the source half-space or incidence row, the finite step row, and the resulting vertex, edge, or face row. Collecting these entries gives the three advertised finite readback ledgers, with no appeal to an ambient geometric carrier outside the packet. □

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket vertex edge ledger readback`

Theorem 209.4 (Polytope ConvexSet and FinSet dependency readback)**.** *Every* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet reads back exactly the two dependency rows* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *used to scope its half-space and finite combinatorial ledgers. The convex dependency accounts for half-space intersection and carried face rows, while the finite-set dependency accounts for the vertex, edge, face, and incidence indices.*

Proof. The first two components of the packet are the displayed dependency witnesses η_C and η_F. Projecting them gives the convex-set carrier and finite-set carrier rows available to the packet.

Conversely, every half-space, vertex, edge, face, and incidence clause in the packet definition is scoped through one of these two witnesses and a finite Cont row. Therefore the dependency readback is exactly $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ together with $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$, plus the kernel rows already named by the packet. □

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket convex finset dependency readback`

Theorem 209.5 (Polytope downstream boundary)**.** *Downstream convex consumers of a* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet may read only the* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *dependency, the half-space rows, the carried face rows, the* hsame *transports, and the finite* Cont *ledgers connecting them. Downstream combinatorial consumers may read only the* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *dependency, the vertex, edge, face, and incidence ledgers, the same* hsame *transports, and the same* Pkg *provenance rows. No downstream consumer receives an ambient Euclidean metric, a quotient carrier, an external face lattice, or a standard bridge from this packet.*

Proof. For a convex consumer, project the packet to η_C, the half-space list, the face rows, and the Cont ledgers that generate those faces. These are exactly the rows named in the carrier definition for convex readback.

For a combinatorial consumer, project the packet to η_F, the finite vertex, edge, face, and incidence ledgers, their hsame transport rows, and the provenance package. These rows determine the finite combinatorial surface without adding any external lattice object.

The two projections share only the kernel rows explicitly stored in the packet: BHist rows, hsame comparisons, Cont ledgers, and Pkg provenance. Hence the boundary grants precisely those read permissions and no metric, quotient, external lattice, or bridge data. □

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket downstream boundary`
↪ variant `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket downstream boundary transport`

Definition 209.6 (Polytope face classifier)**.** *The* $\mathsf{Polytope}^{\uparrow}$ *face classifier relates two BHist face packets when their half-space rows, vertex rows, edge rows, face rows,* Cont *ledgers, and* Pkg *provenance rows compare componentwise by* hsame*, and when the dependency rows supplied by* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *are the same accepted dependency rows on both sides.*

Thus classifier agreement is entirely a BEDC row relation: half-space agreement is read through the convex-set dependency, vertex, edge, and face agreement is read through the finite-set dependency, and every incidence or containment movement is witnessed by the displayed Cont *ledgers. No ambient face lattice or metric equality is part of the classifier.*

Lean4 (definition) `BEDC.Derived.PolytopeUp.PolytopeFaceClassifier`

Theorem 209.7 (Polytope face classifier transport)**.** *If two* $\mathsf{Polytope}^{\uparrow}$ *BHist face packets are related by the classifier of Definition 209.6, then any packet obtained by componentwise* hsame *transport of its half-space, vertex, edge, face,* Cont*, and* Pkg *rows is again accepted by the same classifier. The transported packet keeps the same* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *dependency rows and remains available to the downstream boundary of Definition 209.5.*

Proof. Project the classifier into its convex, finite, incidence, ledger, and provenance components. The convex component transports half-space and carried face rows through $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$, while the finite component transports vertex, edge, face, and incidence indices through $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$.

Next transport the finite Cont ledger entries. Each entry has the same source, step, and result shape after replacing endpoints by their hsame mates, and the Pkg row is carried along the same componentwise comparison.

Repack these transported rows using Definition 209.6. The acceptance clauses are exactly the transported classifier clauses, so the resulting packet remains classifier related and has the same downstream readback permissions. □

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeFaceClassifier transport`

Theorem 209.8 (Polytope obligation boundary exhaustion)**.** *The carrier block, face-classifier block, stability block, dependency readback block, and downstream boundary block exhaust the* $\mathsf{Polytope}^{\uparrow}$ NameCert *obligation surface. Every accepted obligation row is one of the finite* BHist *half-space, vertex, edge, face, incidence,* Cont*, or* Pkg *rows named in those blocks, together with the two dependency rows* $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket obligation boundary exhaustion`

Proof. Start with the carrier definition Definition 209.1. It lists all packet fields: dependency rows, half-space rows, vertex rows, edge rows, face rows, incidence ledgers, Cont ledgers, and provenance.

The classifier and stability rows are exactly Definition 209.6, Definition 209.7, and Definition 209.2. They move only the listed finite rows along hsame and do not add a metric, quotient, or external face lattice.

Finally combine dependency readback Definition 209.4 with the downstream boundary Definition 209.5. These projections account for every consumer-visible row, so the displayed blocks exhaust the obligation surface. □

Theorem 209.9 (PolytopeUp obligation surface)**.** *The existing* $\mathsf{Polytope}^{\uparrow}$ *BHist face packet supplies the roadmap obligation surface for this chapter. Its carrier row is Definition 209.1; its classifier and stability rows are the* hsame *transport surface of Definition 209.2; its finite vertex, edge, and face ledger readback is Definition 209.3; its dependency scope is Definition 209.4; and its downstream consumer boundary is Definition 209.5.*

Consequently the obligation inventory consists only of the $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *dependency rows, finite half-space, vertex, edge, face, and incidence* BHist *rows, displayed* hsame *transports, finite* Cont *ledger rows, and* Pkg *provenance. It contains no Euclidean metric, quotient carrier, external face lattice, or standard bridge.*

Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket public obligation surface`
Lean4 ✓ `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket obligation surface`

Proof. Project the packet to its carrier definition. The tuple of Definition 209.1 names exactly the convex dependency, finite-index dependency, half-space rows, vertex ledger, edge ledger, face ledger, continuity ledger, and provenance row.

Next transport the classifier surface. Apply Definition 209.2 to the displayed hsame comparisons; it keeps the dependency witnesses, ledger shape, and package provenance while replacing each carried row by its transported mate.

Read back the finite ledgers and dependencies. The vertex, edge, and face rows come from Definition 209.3, while Definition 209.4 isolates the two source certificates that scope every half-space and finite combinatorial row.

Finally apply Definition 209.5. It is precisely the consumer projection of the same packet, so repacking the carrier, transport, ledger, dependency, and boundary projections gives the stated obligation surface. The exclusion clause follows from the same downstream boundary: none of the packet projections exposes metric, quotient, external lattice, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{Polytope}^{\uparrow}}$**.**

Constructive story: A polytope packet is generated by placing a $\mathsf{ConvexSet}^{\uparrow}$ dependency beside a $\mathsf{FinSet}^{\uparrow}$ dependency, then listing finite half-space histories, vertex histories, edge histories, face histories, incidence rows, finite Cont ledgers, and one Pkg provenance row. The carrier accepts exactly those face packets whose half-spaces are scoped by the convex dependency, whose vertex, edge, face, and incidence indices are scoped by the finite-set dependency, and whose displayed faces are generated by the finite half-space and incidence ledgers; a metric space, quotient carrier, or external face lattice never enters the packet. Two packets are classified componentwise by hsame on half-space, vertex, edge, face, incidence, continuation, and provenance rows while keeping the two dependency certificates fixed. The source is this displayed finite face packet, the pattern records half-space stability, finite vertex-edge-face readback, dependency readback, downstream convex and combinatorial projections, and classifier transport, and the ledger keeps every raw row aligned with the consumer-visible face or incidence row produced from it. The name-certificate obligations are visible as a Lean-checked public certificate: an accepted packet supplies the carrier witness, componentwise hsame supplies reflexive, symmetric, and transitive comparison, and the stability theorem shows that transported components repack as accepted packets. A history h lies in the signature gap of a $\mathsf{Polytope}^{\uparrow}$ token p when the source-domain accepts h as this finite face packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p names the same finite half-space, vertex, edge, face, incidence, continuation, and provenance

rows. The chapter registers this finite face-ledger construction under the name Polytope$^{\uparrow}$ as a complete finite consumption surface, with no Euclidean metric, face-lattice completeness, quotient carrier, or standard bridge introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{Polytope}\uparrow}$, maturePackageClosed)

Scope closed: The mature paper scope is the finite face-packet carrier, classifier transport, convex and finite dependency readback, vertex-edge-face ledger readback, and downstream convex and combinatorial consumer boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{Polytope}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PolytopeUp.PolytopeBHistFacePacket\_public\_obligation\_surface`

Bridge status: none

Not claimed: This chapter does not close a Euclidean metric, face lattice completeness, quotient carrier, or standard bridge.

Upgrade path: No further theory-axis upgrade is claimed in this chapter; bridgeChecked remains a formal bridge-axis matter.

Chapter 210

A Concrete Naming Certificate for $\mathsf{VectorBundle}^{\uparrow}$

The vector-bundle interface $\mathsf{VectorBundle}^{\uparrow}$ packages a fibre bundle whose fibres are vector spaces and whose transition maps are linear. It depends on $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$ (chapter 97) and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47).

210.1 Carrier definition

Definition 210.1 (VectorBundle finite carrier)**.** *A* $\mathsf{VectorBundle}^{\uparrow}$ *carrier is the finite* BHist *packet* $\mathsf{VectorBundlePkt}(V)$ *with the following displayed rows.*

1. *A retained* $\mathsf{Bundle}^{\uparrow}$ *row for the base, total space, projection, fibre model, local-trivialization, transition, and source-ledger histories.*
2. *A retained* $\mathsf{VecSpace}^{\uparrow}$ *row for the vector-space structure carried by each displayed fibre history.*
3. *A local-trivialization row listing a finite* ProbeBundle *of trivializing neighborhoods and their trivialization histories.*
4. *A fibre carrier row assigning to each displayed local fibre a* BHist *carrier accepted by the retained* $\mathsf{VecSpace}^{\uparrow}$ *certificate.*
5. *A linear transition row recording, for each displayed overlap, the transition history and the vector-space linearity witnesses for addition, zero, negation, and scalar action.*
6. *Componentwise* hsame *rows for the base, total, projection, fibre, trivialization, transition, and vector-space operation histories.*
7. Cont *rows for restriction to overlaps, transition composition, and transport of linearity witnesses along classified fibre histories.*
8. *A* Pkg *provenance row tying the retained* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *certificates to the displayed finite vector-bundle packet.*

The carrier exposes only these finite BHist *rows and their displayed* hsame, Cont, *and* Pkg *witnesses; it does not import a host vector-bundle object or an unrecorded transition function.*

Lean4 (definition) `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier`

Definition 210.2 (VectorBundle classifier)**.** *Two* VectorBundlePkt *carriers classify together when the retained* Bundle$^{\uparrow}$ *base, total-space, projection, local-trivialization, and transition rows classify by the bundle certificate, the retained* VecSpace$^{\uparrow}$ *fibre rows classify by the vector-space certificate, and every fibre carrier, linearity, overlap,* Cont*, and* Pkg *row is related by the displayed componentwise* hsame *comparison. The classifier is finite row sameness over base, fibre, trivialization, and transition data, not equality of host vector-bundle structures.*

Lean4 (definition) `BEDC.Derived.VectorBundleUp.VectorBundleComponentwiseClassifier`
Lean4 (definition) `BEDC.Derived.VectorBundleUp.VectorBundleFiniteClassifier`
Lean4 (definition) `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrierClassifier`
Lean4 (definition) `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier componentwise classifier`

210.2 The certificate

Theorem 210.3 (VectorBundle carrier obligation)**.** *Every accepted* VectorBundlePkt(V) *row projects to one of the displayed* Bundle$^{\uparrow}$*,* VecSpace$^{\uparrow}$*, local-trivialization, fibre-carrier, linear-transition,* hsame*,* Cont*, or* Pkg *rows of Definition 210.1. There is no additional carrier case for a host vector bundle.*

Lean4 ✓ `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier carrier obligation`
↪ variant `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier obligation surface`

Proof. Unpack the carrier definition. The first two coordinates are retained dependency rows from Bundle$^{\uparrow}$ and VecSpace$^{\uparrow}$. The remaining coordinates are the local-trivialization probe bundle, fibre carrier rows, linear transition rows, componentwise hsame comparisons, Cont ledgers, and Pkg provenance.

Projection from an accepted packet can therefore read only one of those coordinates. The bundle dependency supplies base, total, projection, trivialization, and transition histories; the vector-space dependency supplies the fibre operation histories and their classifier.

Conversely, any displayed coordinate is repacked by the same finite packet constructor. Constructor inversion provides no branch for an ambient vector-bundle object, an implicit local trivialization, or an unlogged transition map. □

Theorem 210.4 (VectorBundle transition ledger exactness)**.** *The transition surface of* VectorBundle$^{\uparrow}$ *is exactly the finite overlap ledger in Definition 210.1: every transition comparison is recorded by displayed overlap rows, linearity rows,* Cont *composition rows, and* Pkg *provenance, and every such ledger row repacks as a public transition witness.*

Lean4 ✓ `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier transition ledger exactness`

Proof. Project the transition coordinate of a carried vector-bundle packet. It lists the overlap row, the source and target trivialization histories, the displayed transition history, and the vector-space linearity witnesses for the four operation families.

Exactness in the forward direction is projection. A public transition witness must cite the overlap row and the corresponding linearity rows, so the witness is represented inside the finite ledger.

For the reverse direction, start with a displayed overlap ledger. Repack its transition history, linearity witnesses, overlap restriction Cont rows, and source provenance into the transition coordinate of VectorBundlePkt(V). This produces the public transition witness and uses no host transition function. □

Theorem 210.5 (VectorBundle transition composition scope)**.** *Every composed transition witness in* $\mathsf{VectorBundle}^{\uparrow}$ *is consumed through the displayed overlap rows, the source and target trivialization histories, the linearity rows, the* Cont *composition rows, and the* Pkg *provenance of Definition 210.1. No host transition composition, ambient vector-bundle map, or unlogged fibre map may be invoked to build such a witness.*

Lean4 ✓ `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier transition composition scope`

Proof. Project the transition coordinate of the carried packet. A composable transition readback first exposes the two overlap ledgers and the three trivialization histories that supply the source, middle, and target endpoints.

The two ledger rows already contain the displayed linearity witnesses for the four vector-space operation families. Their endpoints are connected only by the listed transition-composition Cont rows, so the composition step is a consumer of those rows rather than an operation on a host bundle.

Repack the composed transition by carrying forward the same overlap ledgers, linearity rows, Cont composition witness, and Pkg provenance from Definition 210.1. Constructor inversion leaves no coordinate for an external transition function or an unlogged vector-bundle map. □

Theorem 210.6 (VectorBundle classifier stability obligation)**.** *The classifier of Definition 210.2 transports accepted vector-bundle packets: classified base, fibre, trivialization, and transition rows repack as a carried* $\mathsf{VectorBundle}^{\uparrow}$ *packet with the same finite transition ledger shape.*

Lean4 ✓ `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier classifier stability obligation`

Proof. Begin with a classifier witness between two displayed packets. Its retained $\mathsf{Bundle}^{\uparrow}$ component transports the base, total, projection, local-trivialization, and bundle transition rows by the existing bundle certificate.

Its retained $\mathsf{VecSpace}^{\uparrow}$ component transports the fibre carrier and the vector-space operation histories. Componentwise hsame then transports the linearity rows for addition, zero, negation, and scalar action.

The overlap-restriction and transition-composition Cont rows transport because their endpoint histories are among the same base, fibre, trivialization, and transition comparisons. The Pkg row transports the source provenance. Repacking these transported rows gives another accepted finite vector-bundle packet. □

Theorem 210.7 (VectorBundle NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{VectorBundle}^{\uparrow}}$ *fields determined by Definition 210.1 and Definition 210.2 are exactly the finite carrier rows, the componentwise classifier, carrier transport, the transition ledger exactness row, and the* Pkg *provenance row. The surface contains no host vector-bundle semantics beyond the displayed* $\mathsf{Bundle}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *dependency certificates.*

Lean4 ✓ `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier obligation surface`

Proof. First project the carrier by Definition 210.3. This accounts for the retained bundle and vector-space rows, the local-trivialization probe rows, the fibre carrier rows, the linear transition rows, the hsame comparisons, the Cont ledgers, and the provenance row.

Next apply Definition 210.4. The public transition surface is exhausted by the finite overlap ledger and the displayed linearity witnesses, and every such ledger repacks as a transition witness.

Finally apply Definition 210.6. It gives carrier transport and classifier stability over the same finite rows. Since all public fields have been projected from those rows or repacked into them, the certificate exposes no additional host vector-bundle semantics. □

Closure status for $\mathsf{NameCert}_{\mathsf{VectorBundle}^{\uparrow}}$.

Constructive story: A vector-bundle packet is generated from displayed $\mathsf{Bundle}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ dependency rows, local trivialization histories, finite transition ledgers, componentwise classifier rows, continuation ledgers, and package provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VectorBundle}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The scoped boundary binds the displayed $\mathsf{Bundle}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, carrier, classifier, and transition-ledger rows of the vector-bundle packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VectorBundle}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.VectorBundleUp.VectorBundleFiniteCarrier\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host vector-bundle semantics, ambient transition functions, fibrewise smooth or topological structure, or a standard bridge.

Upgrade path: Bridge closure requires a standard finite vector-bundle bridge over the displayed carrier, classifier, transition ledger, hsame, Cont, Pkg, and NameCert rows.

Chapter 211

A Concrete Naming Certificate for $\mathsf{CStarAlg}^{\uparrow}$

The C*-algebra interface $\mathsf{CStarAlg}^{\uparrow}$ packages a Banach algebra with an involution satisfying $\|a^*a\| = \|a\|^2$. It depends on $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90) and $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ (chapter 43), and is the substrate for noncommutative topology and operator-theoretic spectra.

211.1 Carrier definition

Definition 211.1 ($\mathsf{CStarAlg}^{\uparrow}$ BHist carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *row* B *and a* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *row* R *over the same* BHist *source. A* $\mathsf{CStarAlg}^{\uparrow}$ *carrier row is the predicate on a history* h : BHist *asserting that* h *carries exactly the following certificate-visible data: the Banach carrier row from* B*, the RingUp multiplication row from* R*, an involution row* s : BHist *on the same carried endpoints, a norm-square compatibility row for* s *and the multiplication row, the* hsame *transport rows for the carrier, multiplication, involution, and norm-square witnesses, and the* Pkg *provenance rows naming* B *and* R*.*

Lean4 (definition) `BEDC.Derived.CStarAlgUp.CstaralgebraBHistCarrier`

The predicate of Definition 211.1 treats multiplication, involution, norm, and provenance as BEDC rows. It does not expose an ambient host $*$-algebra object or a quotient of algebra presentations.

211.2 The certificate

Definition 211.2 ($\mathsf{CStarAlg}^{\uparrow}$ certificate surface)**.** *The* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *certificate surface consists of the carrier predicate of Definition 211.1, the classifier inherited from the common* hsame *transport of the BanachUp and RingUp rows, the involution transport row, the norm-square compatibility transport row, the* Cont *ledger recording multiplication and involution observations, and the* Pkg *provenance naming* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *as the only dependencies.*

Theorem 211.3 ($\mathsf{CStarAlg}^{\uparrow}$ NameCert obligation surface)**.** *The surface of Definition 211.2 is a complete paper-side obligation surface for* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$*. Every accepted* $\mathsf{CStarAlg}^{\uparrow}$ *row is read from a* BHist *carrier row, a BanachUp source row, a RingUp multiplication row, an involution row, a norm-square compatibility row, the corresponding* hsame *and* Cont *transport rows, and the* Pkg *provenance rows for the two dependency certificates.*

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraNameCert obligation surface`

Proof. Project an accepted carrier witness through Definition 211.1. The projection exposes the Banach row, the RingUp multiplication row, the involution row, the norm-square row, the transport rows, and the dependency provenance.

The classifier is then forced by the same hsame transport rows. Banach transport moves the norm carrier, RingUp transport moves multiplication, and the involution and norm-square rows are retained only when their displayed histories transport along the same carrier comparison.

Repack these projections as the certificate surface of Definition 211.2. No further row is available to the repacking step, so the listed rows are the whole obligation surface. □

Lean4 ✓ `BEDC.Derived.CStarAlgUp.CStarAlgCertificateSurface namecert obligation`
↪ variant `BEDC.Derived.CStarAlgUp.CStarAlgBHistCarrier namecert obligation surface`

Theorem 211.4 ($\mathsf{CStarAlg}^{\uparrow}$ Gelfand and von Neumann consumer boundary)**.** *Any* $\mathsf{GelfandDuality}^{\uparrow}$ *or* $\mathsf{VonNeumannAlg}^{\uparrow}$ *consumer citing* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *may consume only the* BHist *carrier, the BanachUp carrier and norm rows, the RingUp multiplication row, the involution row, the norm-square compatibility row, the* hsame *and* Cont *transport rows, and the* Pkg *provenance rows displayed in Definition 211.2. It may not infer a spectrum, weak-operator closure, representation on a Hilbert space, or commutative Gelfand transform from the root* $\mathsf{CStarAlg}^{\uparrow}$ *surface alone.*

Proof. Read the dependency citation through Definition 211.3. The citation yields only the certificate rows projected there, so a downstream consumer receives the carried Banach, RingUp, involution, norm-square, transport, ledger, and provenance data.

For $\mathsf{GelfandDuality}^{\uparrow}$, a spectrum and a commutative transform require rows not present in the $\mathsf{CStarAlg}^{\uparrow}$ root surface. For $\mathsf{VonNeumannAlg}^{\uparrow}$, weak-operator closure and Hilbert-space representation data also require rows not present in that surface.

Thus both consumers share the same BEDC source boundary: they may cite the $\mathsf{CStarAlg}^{\uparrow}$ certificate rows, while their own chapters must supply any spectrum, closure, representation, or transform rows they use. □

Lean4 ✓ `BEDC.Derived.CStarAlgUp.CStarAlgCertificateSurface gelfand vonneumann consumer boundary`

Theorem 211.5 ($\mathsf{CStarAlg}^{\uparrow}$ unital-commutative source boundary)**.** *A* $\mathsf{GelfandDuality}^{\uparrow}$ *consumer may treat a* $\mathsf{CStarAlg}^{\uparrow}$ *row as a unital commutative source only when the displayed* BHist *surface supplies, in addition to the root certificate, a unit row, multiplication row, commutativity classifier row, involution row, norm-square row,* hsame *transport,* Cont *ledger, and* Pkg *provenance over the same BanachUp and RingUp dependency rows. The root* $\mathsf{CStarAlg}^{\uparrow}$ *certificate alone does not export the Gelfand transform, the spectrum row, or a compact Hausdorff carrier.*

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraBHistCarrier unital commutative source boundary`

Proof. Unfold Definition 211.1. The root carrier exposes the Banach row, RingUp multiplication row, involution row, norm-square row, transport rows, and dependency provenance.

Unfold Definition 211.2. Its public classifier and ledger are the displayed hsame, Cont, and Pkg rows over those carrier components; unit and commutativity data must therefore be supplied as visible rows of the consumer surface when $\mathsf{GelfandDuality}^{\uparrow}$ asks for a unital commutative source.

Finally apply Definition 211.4. That boundary rules out hidden spectrum data and hidden transform rows, so the $\mathsf{GelfandDuality}^{\uparrow}$ source read is limited to the displayed unital, commutative, involutive, norm-square, transport, ledger, and provenance rows. □

Theorem 211.6 (No host star-algebra primitive in $\mathsf{CStarAlg}^{\uparrow}$)**.** *The* $\mathsf{CStarAlg}^{\uparrow}$ *obligation surface does not include a host* $*$*-algebra primitive. Equality and transport for* $\mathsf{CStarAlg}^{\uparrow}$ *are mediated only by* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *together with the displayed involution and norm-square rows.*

Proof. Unfold the carrier predicate of Definition 211.1. Its only source rows are the BanachUp and RingUp certificate rows, the involution row, the norm-square row, the transport rows, and the provenance rows.

Unfold the certificate surface of Definition 211.2. The surface records classifier and ledger data by hsame, Cont, and Pkg; it never adds an ambient equality predicate for host $*$-algebras.

Therefore replacing the displayed rows by a host $*$-algebra primitive would change the carrier. The BEDC object is precisely the row package described above. □

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraBHistCarrier no host star algebra primitive`

Theorem 211.7 ($\mathsf{CStarAlg}^{\uparrow}$ Banach–Ring source boundary)**.** *Every public source read of a* $\mathsf{CStarAlg}^{\uparrow}$ *certificate factors through the displayed BanachUp carrier row, the RingUp multiplication row, the involution row, the norm-square compatibility row, the* hsame *transports for those rows, the finite* Cont *ledger, and the* Pkg *provenance naming the BanachUp and RingUp dependency certificates. No source read exports a host* $*$*-algebra equality, an ambient Banach-ring object, or spectrum data.*

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraBHistCarrier banach ring source boundary`

Proof. Project the source read through Definition 211.1. The carrier predicate lists exactly the Banach row, RingUp multiplication row, involution row, norm-square row, transport rows, and dependency provenance.

Next read the certificate surface of Definition 211.2. Its public ledger is the Cont route recording multiplication and involution observations together with the same hsame and Pkg rows.

Finally apply Definition 211.6. Since the surface has no slot for host $*$-algebra equality, ambient Banach-ring structure, spectrum, or transform data, a public source read must factor through the displayed BEDC rows alone. □

Theorem 211.8 ($\mathsf{CStarAlg}^{\uparrow}$ classifier stability)**.** *Componentwise* hsame *transport of the BanachUp carrier row, the RingUp multiplication row, the involution row, the norm-square compatibility row, the* Cont *ledger, and the* Pkg *provenance preserves the* $\mathsf{CStarAlg}^{\uparrow}$ *classifier and the carrier predicate of Definition 211.1. The transported certificate remains a* $\mathsf{CStarAlg}^{\uparrow}$ *row precisely because each displayed component is transported inside the certificate surface.*

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraBHistCarrier classifier stability`

Proof. Unfold the carrier predicate in Definition 211.1. A transported carrier is accepted when the Banach, multiplication, involution, norm-square, and provenance rows are all transported along their displayed hsame witnesses.

Unfold Definition 211.2. Its classifier is inherited from those same componentwise transports, while the ledger rows are read only through Cont and the dependency provenance remains the displayed Pkg row.

Thus the transported packet satisfies the same carrier predicate and the same certificate classifier. No host star-algebra equality or spectrum datum is introduced, because neither appears among the transported rows. □

Theorem 211.9 ($\mathsf{CStarAlg}^{\uparrow}$ ledger exactness)**.** *The public* $\mathsf{CStarAlg}^{\uparrow}$ *ledger is exhausted by the multiplication row, the involution row, the norm-square compatibility row, the* hsame *transport rows, the finite* Cont *readbacks, and the* Pkg *provenance rows for* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$*. Every downstream* $\mathsf{GelfandDuality}^{\uparrow}$ *or* $\mathsf{VonNeumannAlg}^{\uparrow}$ *citation of* $\mathsf{CStarAlg}^{\uparrow}$ *reads one of those ledger rows, and every listed row is part of the certificate surface.*

Lean4 ✓ `BEDC.Derived.CstaralgebraUp.CstaralgebraBHistCarrier ledger exactness`

Proof. Start from the certificate surface in Definition 211.2. Its ledger records multiplication and involution observations, norm-square compatibility, componentwise transport, finite readback, and dependency provenance.

For downstream consumption, apply Definition 211.4. The consumer boundary exposes exactly the same carrier, source, transport, ledger, and provenance rows to $\mathsf{GelfandDuality}^{\uparrow}$ and $\mathsf{VonNeumannAlg}^{\uparrow}$.

The reverse inclusion is by projection from Definition 211.1: each listed ledger row is a field of the displayed carrier or certificate surface. Spectrum data, weak-operator closure, Hilbert-space representations, and host star-algebra equality are therefore outside the public ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$**.**

Constructive story: A C*-algebra row is assembled by placing a certified $\mathsf{Banach}^{\uparrow}$ row and a certified $\mathsf{Ring}^{\uparrow}$ multiplication row over the same BHist source, then adding the displayed involution row, norm-square compatibility row, transport rows, finite continuation readbacks, and dependency provenance. The carrier accepts only histories whose Banach, ring, involution, norm, and provenance fields are all present in that finite packet; host $*$-algebra equality, spectrum data, weak-operator closure, and Hilbert-space representation rows are not carrier fields. Two accepted rows classify together by componentwise hsame transport of the Banach carrier, multiplication, involution, norm-square witness, Cont ledger, and Pkg provenance rather than by equality in an ambient algebra. The source specification is the paired Banach and ring dependency surface, the pattern specification records multiplication, involution, and the norm-square law $\|a^*a\| = \|a\|^2$, and the ledger keeps multiplication and involution observations plus their finite readbacks visible. The five name-certificate obligations come from an accepted packet for inhabitation, the equivalence laws of the displayed component classifiers, and the stability theorem saying that every transported component rebuilds the same $\mathsf{CStarAlg}^{\uparrow}$ carrier row. A packet history h lies in the signature gap of a $\mathsf{CStarAlg}^{\uparrow}$ token p when the source-domain accepts its Banach-ring-involution surface, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the transported multiplication, involution, and norm-square rows are the ones encoded by p. The chapter registers this displayed Banach-ring-involution construction under the name $\mathsf{CStarAlg}^{\uparrow}$; it introduces no host star-algebra primitive, no spectrum, no Gelfand transform, and no standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$, obligation)

Scope closed: The obligation scope binds the $\mathsf{Banach}^{\uparrow}$ and $\mathsf{Ring}^{\uparrow}$ dependency rows, the involution row, norm-square compatibility, componentwise hsame, finite Cont readback, and Pkg provenance of the $\mathsf{CStarAlg}^{\uparrow}$ certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CStarAlgUp.CStarAlgCertificateSurface\_namecert\_obligation`

Bridge status: none

Not claimed: This chapter does not claim a spectrum, weak-operator closure, Hilbert-space representation, host star-algebra equality, or standard bridge.

Upgrade path: Scoped closure requires primitive binding for the displayed Banach, Ring, involution, norm-square, transport, ledger, and Gelfand or von-Neumann consumer rows.

Chapter 212

A Concrete Naming Certificate for $\mathsf{VonNeumannAlg}^{\uparrow}$

The von Neumann algebra interface $\mathsf{VonNeumannAlg}^{\uparrow}$ packages a unital $*$-subalgebra of $B(H)$ closed in the weak operator topology. It depends on $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ (chapter 211) and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91).

212.1 Carrier definition

Definition 212.1 (VonNeumannAlgUp BHist carrier)**.** *A* $\mathsf{VonNeumannAlg}^{\uparrow}$ *carrier is a finite* BHist *packet whose rows display the following data:*

a $\mathsf{CStarAlg}^{\uparrow}$ *source row recording the algebra, unit, multiplication, norm, and* $*$*-operation;*

a $\mathsf{Hilbert}^{\uparrow}$ *source row recording the Hilbert carrier and its vector-state probes;*

bounded-operator endpoint rows for each accepted algebra element over the displayed Hilbert carrier;

adjoint and multiplication rows comparing the operator endpoints with the $*$ *and product rows;*

finite weak-operator test rows, each logged as a probe ledger between displayed test vectors;

componentwise hsame *rows,* Cont *readback routes,* Pkg *provenance, and a* NameCert *provenance row.*

The carrier has no coordinate for completed operator topologies, host subalgebras of $B(H)$*, choices of projection lattices, unlogged weak closures, or external equality of von Neumann algebras.*

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier finite packet rows`

212.2 The certificate

Theorem 212.2 (VonNeumannAlgUp carrier obligation)**.** *The carrier of Definition 212.1 is a finite BEDC obligation surface for* $\mathsf{VonNeumannAlg}^{\uparrow}$*. Every exported algebraic, Hilbert, operator-endpoint, adjoint, multiplication, weak-test, transport, continuation, and provenance read is a projection of one displayed* BHist *row.*

Proof. Project the packet row by row. The $\mathsf{CStarAlg}^{\uparrow}$ source supplies the finite algebraic surface: carrier, unit, product, norm, and star rows, together with their hsame comparisons and package provenance.

Project the $\mathsf{Hilbert}^{\uparrow}$ source and the bounded-operator endpoint rows next. Each operator endpoint is indexed by a displayed algebra element and the displayed Hilbert carrier, so it has no independent host subalgebra coordinate.

The remaining rows are the weak-operator probes, Cont readbacks, componentwise hsame transports, Pkg provenance, and the NameCert provenance row. Since the definition lists exactly these coordinates, every consumer-visible read is exhausted by the finite packet. □

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier obligation surface`

Theorem 212.3 (VonNeumannAlgUp weak-operator ledger exactness)**.** *For an accepted* $\mathsf{VonNeumannAlg}^{\uparrow}$ *carrier, every weak-operator comparison visible to a consumer is a finite probe-ledger row between displayed* $\mathsf{Hilbert}^{\uparrow}$ *test-vector rows and displayed bounded-operator endpoint rows. The packet exports no completed weak operator topology and no unlogged limit closure.*

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonNeumannAlgBHistCarrier weak operator ledger exactness`
Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier weak operator ledger exactness`

Proof. Read a visible weak-operator comparison from the public packet. By Definition 212.1, the only weak-operator coordinates are finite probe rows, each naming its two test-vector rows through the $\mathsf{Hilbert}^{\uparrow}$ source and its operator endpoint through the bounded-operator ledger.

The Cont row records the readback route from that probe to the endpoint comparison, and componentwise hsame transports the same finite probe under classified replacement. There is no carrier coordinate for a completed weak topology or an unlogged net of operators, so the visible comparison is exactly the displayed probe ledger. □

Theorem 212.4 (VonNeumannAlgUp star-operation stability)**.** *If two accepted* $\mathsf{VonNeumannAlg}^{\uparrow}$ *carriers are related componentwise by their displayed* hsame *rows, then the bounded-operator endpoint assigned to the* $\mathsf{CStarAlg}^{\uparrow}$ *star row transports to the adjoint endpoint row of the classified carrier, and multiplication endpoints transport along the displayed product rows.*

Proof. Use the $\mathsf{CStarAlg}^{\uparrow}$ source row to read the algebraic star and product coordinates. The carrier stores adjoint and multiplication endpoint rows whose indices are those same displayed algebraic coordinates.

Apply componentwise hsame to the algebra row, Hilbert row, endpoint row, and operation row. The adjoint endpoint is transported along the star coordinate, while the multiplication endpoint is transported along the product coordinate.

The Cont readback route records that the transported endpoint is the one exported by the target packet. Thus classified replacement preserves the star and multiplication surfaces without invoking host equality of operator algebras. □

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier star operation stability`

Theorem 212.5 (VonNeumannAlgUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *obligation surface consists of the finite carrier, the componentwise classifier on its displayed rows, star and multiplication stability, weak-operator probe-ledger exactness,* Cont *readbacks, and* Pkg *provenance for the consumed* $\mathsf{CStarAlg}^{\uparrow}$ *and* $\mathsf{Hilbert}^{\uparrow}$ *certificates.*

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonNeumannAlgBHistCarrier namecert obligation surface`

Proof. Carrier inhabitance is the assertion that the rows listed in Definition 212.1 form a finite packet. The classifier is componentwise hsame on the $\mathsf{CStarAlg}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, endpoint, operation, probe, continuation, and provenance rows.

Use Definition 212.4 for operation transport and Definition 212.3 for weak-test exactness. The Pkg rows record the dependency certificates consumed by those two reads.

Repacking these rows yields the naming-certificate surface. No additional analytic closure, projection lattice, host subalgebra, or external von Neumann algebra equality is exported. □

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier namecert obligation surface`

Theorem 212.6 (VonNeumannAlgUp downstream boundary)**.** *Any downstream BEDC consumer of a* $\mathsf{VonNeumannAlg}^{\uparrow}$ *certificate can inspect only the finite algebra row, Hilbert row, bounded-operator endpoint rows, adjoint and multiplication rows, weak-operator probe ledgers,* hsame *transports,* Cont *readbacks,* Pkg *rows, and the* NameCert *provenance row displayed by the carrier.*

Proof. Let a downstream read be accepted by the $\mathsf{VonNeumannAlg}^{\uparrow}$ certificate. The read factors through the public carrier projection of Definition 212.2, so it must select one of the rows displayed in Definition 212.1.

If the read concerns weak-operator behavior, Definition 212.3 routes it through a finite probe ledger. If the read concerns algebraic operation stability, Definition 212.4 routes it through the $\mathsf{CStarAlg}^{\uparrow}$ star or product row and the corresponding bounded-operator endpoint.

All remaining reads are transport, continuation, package, or naming-provenance rows. Hence the public boundary is precisely the finite BEDC certificate surface, with no access to completed operator topologies, unlogged weak closures, projection choices, or host subalgebra equality. □

Lean4 ✓ `BEDC.Derived.VonneumannalgebraUp.VonneumannalgebraBHistCarrier downstream boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VonNeumannAlg}^{\uparrow}$ packet is generated from a CStarAlgUp algebraic source row, a HilbertUp carrier row, bounded-operator endpoint rows, adjoint and multiplication rows, finite weak-operator probe ledgers, componentwise hsame transport, Cont readback routes, Pkg provenance, and local NameCert provenance. The classifier reads operator-algebra structure only through these finite BEDC rows: weak comparison is a displayed probe ledger between Hilbert test vectors, and operation stability transports the CStar star and product rows to corresponding bounded-operator endpoints. The ledger story is weak-operator visibility without completed topology, projection-lattice choice, unlogged closure, or host subalgebra equality.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$, seed)

Scope closed: This chapter binds the BHist carrier definition and the carrier-obligation, weak-operator-ledger, star-stability, NameCert-surface, and downstream-boundary theorems for $\mathsf{VonNeumannAlg}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim completed operator topologies, host subalgebras of $B(H)$, choices of projection lattices, unlogged weak closures, or external equality of von Neumann algebras.

Upgrade path: To reach obligationClosure, the checked component lemmas must be assembled into a formal chapter-level closure theorem for the finite operator packet.

Chapter 213

A Concrete Naming Certificate for $\mathsf{OperatorIdeal}^{\uparrow}$

The operator-ideal interface $\mathsf{OperatorIdeal}^{\uparrow}$ packages a two-sided ideal of bounded operators on a Hilbert space carrier (compact, trace class, Hilbert-Schmidt). It depends on $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91) and $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81).

213.1 Carrier definition

Definition 213.1 (OperatorIdeal Hilbert source surface)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier supplied by $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$, with Hilbert carrier predicate C_H, classifier $\sim_H$, vector addition, additive inverse, zero, scalar action, and inner-product norm rows. Let $\mathsf{Op}_H(T)$ be a BHist predicate whose endpoints are bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ map-codes. The operator-ideal source surface over H is the ten-field $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ inventory with the following rows.*

1. *The traditional target is a two-sided ideal of bounded Hilbert operators on H.*
2. *The BEDC source is $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ together with $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ read over the bounded-operator composition ring on H.*
3. *The carrier is the predicate Op_H on bounded-operator rows $H \to H$.*
4. *The classifier is inherited from the Hilbert and linear-map classifier: $T \sim_{\mathsf{Op}(H)} S$ means pointwise $\sim_H$ agreement on every carried Hilbert endpoint after both map-codes are certified bounded and linear.*
5. *The stability rows are the bounded zero operator, addition, negation, scalar action where the Hilbert module row is consumed, and composition on the left and on the right by every carried bounded operator.*
6. *The ledger records the Hilbert source certificate, the $\mathsf{LinMap}^{\uparrow}(H, H)$ row, the ideal-membership witness for the selected predicate, and the two composition witnesses for left and right multiplication.*
7. *The core theorem rows are the carrier support, additive closure, two-sided absorption, and downstream trace-class boundary rows in section 213.2.*
8. *Exactness is row exhaustion: an accepted operator-ideal endpoint exposes only the Hilbert source, bounded linear-map row, ideal membership, classifier transport, additive rows, scalar rows, and*

the two bounded composition witnesses displayed here.

9. *The standard bridge row is absent at this paper site.*
10. *The boundary package is the trace-class-ready operator-ideal surface consumed by* $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, *and* $\mathsf{QuantumChannel}^{\uparrow}$.

Definition 213.2 (Trace-class operator-ideal carrier)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* Op_H *be the bounded-operator carrier from Definition 213.1. A trace-class carrier over H is the subcarrier*

$$\mathsf{TC}(H)(T) :\Longleftrightarrow \mathsf{Op}_H(T) \wedge \mathsf{TraceClassIdeal}_H(T),$$

where $\mathsf{TraceClassIdeal}_H$ *is an* $\mathsf{Ideal}^{\uparrow}$ *predicate over the bounded-operator composition ring carried by* Op_H*. Its classifier is the restriction of* $\sim_{\mathsf{Op}(H)}$*, and its ledger contains the bounded-operator witness for T together with the* $\mathsf{Ideal}^{\uparrow}$ *membership witness* $\mathsf{TraceClassIdeal}_H(T)$*.*

213.2 The certificate

Theorem 213.3 (OperatorIdeal carrier support row)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. If* $\mathsf{TC}(H)(T)$*, then T is carried by the bounded-operator carrier over H:*

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{Op}_H(T).$$

Proof. Unpack Definition 213.2. A trace-class witness is a pair consisting of the bounded-operator row $\mathsf{Op}_H(T)$ and an $\mathsf{Ideal}^{\uparrow}$ membership witness $\mathsf{TraceClassIdeal}_H(T)$.

Projecting the first component gives the displayed bounded-operator carrier support. No analytic trace row is used; the result is exactly the support projection from the carrier definition. □

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass support readback`

Theorem 213.4 (OperatorIdeal additive closure row)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. The* $\mathsf{OperatorIdeal}^{\uparrow}$ *certificate supplies the trace-class zero, additive, and negation rows:*

$$\mathsf{TC}(H)(0_H),$$

$$\mathsf{TC}(H)(T) \wedge \mathsf{TC}(H)(S) \Longrightarrow \mathsf{TC}(H)(T +_H S),$$

and

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(-_H T).$$

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass additive closure row`

Proof. Read $\mathsf{TC}(H)$ as the $\mathsf{Ideal}^{\uparrow}$ predicate $\mathsf{TraceClassIdeal}_H$ over the bounded-operator composition ring from Definition 213.2. The zero row of the $\mathsf{Ideal}^{\uparrow}$ predicate gives membership of the bounded zero operator; the bounded-operator surface of Definition 213.1 supplies its Op_H carrier witness.

For addition, two trace-class witnesses project to two ideal-membership witnesses. The additive closure row of $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ gives membership for $T +_H S$, while the bounded linear-map rows inherited from $\mathsf{Hilbert}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ carry the sum as a bounded operator.

For negation, project the additive inverse row of the same $\mathsf{Ideal}^{\uparrow}$ predicate and pair it with the bounded negation row from the operator surface. These paired rows are precisely the three displayed $\mathsf{TC}(H)$ clauses. □

Theorem 213.5 (OperatorIdeal two-sided absorption row)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. If A is a carried bounded operator on H and T is trace-class on H, then both composites are trace-class:*

$$\mathsf{Op}_H(A) \wedge \mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(A \circ T) \wedge \mathsf{TC}(H)(T \circ A).$$

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass two sided absorption row`

Proof. Unpack the trace-class witness for T. It supplies the bounded-operator row $\mathsf{Op}_H(T)$ and the ideal-membership row $\mathsf{TraceClassIdeal}_H(T)$.

The input $\mathsf{Op}_H(A)$ is exactly the ambient carried endpoint required by the two-sided absorption field of $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ over the bounded-operator composition ring. Projecting that field gives ideal membership for both $A \circ T$ and $T \circ A$.

The bounded composition rows of Definition 213.1 carry the two composites as bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ map-codes. Pairing these carrier witnesses with the two projected ideal-membership witnesses gives $\mathsf{TC}(H)(A \circ T)$ and $\mathsf{TC}(H)(T \circ A)$. □

Theorem 213.6 (DensityMatrix trace-class dependency boundary)**.** *The downstream* $\mathsf{DensityMatrix}^{\uparrow}$ *and* $\mathsf{QuantumChannel}^{\uparrow}$ *chapters may consume from* $\mathsf{OperatorIdeal}^{\uparrow}(H)$ *only the trace-class carrier* $\mathsf{TC}(H)$ *of Definition 213.2, its classifier transport from* $\sim_{\mathsf{Op}(H)}$*, the zero, addition, negation, and real scalar/module rows inherited through* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{LinMap}^{\uparrow}$*, and the two-sided absorption row of Definition 213.5. Positivity rows, trace rows, and trace-one rows belong to the downstream chapters themselves. This boundary does not claim compactness, a spectral theorem, trace existence, or a standard operator-algebra bridge.*

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass downstream boundary readback`

Proof. Inspect the ten-field surface in Definition 213.1. Its exactness field says that an operator-ideal endpoint exposes only the Hilbert source, bounded linear-map row, ideal membership, classifier transport, additive rows, scalar rows, and left and right bounded-composition witnesses.

The trace-class carrier in Definition 213.2 further restricts that surface to the $\mathsf{Ideal}^{\uparrow}$ predicate $\mathsf{TraceClassIdeal}_H$. Therefore the support, additive, and absorption theorems above are the available projections for downstream use.

The positivity predicate Pos_H, the trace functional Tr_H, and the trace-one equation are not fields of either displayed OperatorIdeal inventory. They are introduced only when $\mathsf{DensityMatrix}^{\uparrow}$ and $\mathsf{QuantumChannel}^{\uparrow}$ state their own carrier and certificate rows, so none of compactness, spectral decomposition, trace existence, or a standard bridge is exported by this chapter. □

Theorem 213.7 (OperatorIdeal trace-class scalar closure)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. Let a be a carried* $\mathsf{Real}^{\uparrow}$ *scalar endpoint for the Hilbert module action. Write $M_a : H \to H$ for the bounded Hilbert scalar-action endomorphism carried by the scalar row of Definition 213.1,*

$$M_a(x) := a \cdot_H x,$$

and let the induced trace-class scalar action on operators be

$$a \odot_H T := M_a \circ T.$$

Then trace-class membership is closed under this scalar action:

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(a \odot_H T).$$

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass scalar closure`

Proof. The carried scalar hypothesis gives a certified scalar-action row on the Hilbert module surface. Read as a map in the bounded-operator composition ring, that row carries the homothety M_a as a bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ endpoint; in the notation of Definition 213.1, this is the ambient witness $\mathsf{Op}_H(M_a)$.

Now assume $\mathsf{TC}(H)(T)$. By Definition 213.2, this witness consists of a bounded-operator carrier row $\mathsf{Op}_H(T)$ and an ideal-membership row $\mathsf{TraceClassIdeal}_H(T)$ inside the bounded-operator composition ring.

Apply Definition 213.5 with the carried ambient operator $A = M_a$. The left absorption component gives

$$\mathsf{TC}(H)(M_a \circ T),$$

while the right component gives $\mathsf{TC}(H)(T \circ M_a)$ and is not needed here.

The displayed definition of the inherited scalar action identifies $M_a \circ T$ with $a \odot_H T$. The classifier-transport row for the ideal predicate in Definition 81.16 carries the obtained trace-class witness across that endpoint identity, giving

$$\mathsf{TC}(H)(a \odot_H T).$$

□

Definition 213.8 (OperatorIdeal root BHist threshold source)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier and let $\mathsf{TC}(H)$ be the trace-class carrier of Definition 213.2. The root* BHist *threshold source for $\mathsf{OperatorIdeal}^{\uparrow}(H)$ is the ten-field package with the following rows.*

1. *The traditional target is the two-sided operator ideal surface on Hilbert carriers.*
2. *The BEDC source is $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, and the ambient* NameCert *discipline for row transport.*
3. *The carrier row is the bounded Hilbert-operator carrier Op_H together with its trace-class subcarrier $\mathsf{TC}(H)$.*
4. *The classifier row is $\sim_{\mathsf{Op}(H)}$, read as pointwise $\sim_H$ agreement after the two endpoints are certified as bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ codes.*
5. *The stability rows are bounded zero, addition, negation, real scalar action, left composition by a carried bounded operator, and right composition by a carried bounded operator.*
6. *The ledger rows are the Hilbert source certificate, the bounded linear-map certificate, the $\mathsf{Ideal}^{\uparrow}$ membership witness $\mathsf{TraceClassIdeal}_H$, and the two bounded-composition witnesses.*
7. *The core rows are the support, additive closure, two-sided absorption, scalar closure, and downstream-boundary theorems of section 213.2.*
8. *Exactness is row exhaustion: no endpoint accepted at this threshold exposes a field beyond the carrier, classifier, stability, ledger, and core rows displayed here.*
9. *The standard bridge row is absent.*
10. *The downstream consumers are $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, and $\mathsf{QuantumChannel}^{\uparrow}$.*

Theorem 213.9 (OperatorIdeal root carrier-classifier threshold)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier. The root* BHist *threshold source of Definition 213.8 exposes only bounded Hilbert-operator endpoints coded by* BHist*, pointwise classifier transport inherited from the Hilbert and* $\mathsf{LinMap}^{\uparrow}$ *rows, and trace-class support through* $\mathsf{TC}(H)$*. In particular,*

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{Op}_H(T),$$

and every classifier comparison at this threshold is a $\sim_{\mathsf{Op}(H)}$ *comparison between certified bounded* $\mathsf{LinMap}^{\uparrow}(H, H)$ *endpoints.*

Proof. Unpack the root threshold source in Definition 213.8. Its carrier row is exactly the bounded Hilbert-operator predicate Op_H plus the trace-class subcarrier $\mathsf{TC}(H)$, while its classifier row is the pointwise Hilbert classifier transported along certified bounded linear maps.

Projecting Definition 213.1 gives the BHist-coded bounded operator surface and its $\sim_{\mathsf{Op}(H)}$ classifier. Projecting Definition 213.2 gives the trace-class pair consisting of $\mathsf{Op}_H(T)$ and $\mathsf{TraceClassIdeal}_H(T)$.

The displayed support implication is precisely Definition 213.3. Since the root exactness row admits no further carrier or classifier field, these are the only carrier-classifier rows visible at the threshold. □

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass root carrier classifier threshold`

Theorem 213.10 (OperatorIdeal root ideal-absorption threshold)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier. Through the trace-class carrier* $\mathsf{TC}(H)$*, the root* BHist *threshold exposes exactly the* $\mathsf{Ideal}^{\uparrow}$ *rows needed for zero, addition, negation, real scalar closure, and two-sided absorption by carried bounded operators:*

$$\mathsf{TC}(H)(0_H),$$

$$\mathsf{TC}(H)(T) \wedge \mathsf{TC}(H)(S) \Longrightarrow \mathsf{TC}(H)(T +_H S),$$

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(-_H T),$$

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(a \odot_H T),$$

and

$$\mathsf{Op}_H(A) \wedge \mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(A \circ T) \wedge \mathsf{TC}(H)(T \circ A).$$

Proof. The trace-class carrier of Definition 213.2 packages a bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ endpoint together with the $\mathsf{Ideal}^{\uparrow}$ membership predicate $\mathsf{TraceClassIdeal}_H$. The root ledger of Definition 213.8 records that membership row and the left and right bounded-composition witnesses.

The zero, addition, and negation clauses are exactly the three projections supplied by Definition 213.4. They use the IdealUp membership row paired with the bounded operator rows from the Hilbert and linear-map source.

The scalar clause is Definition 213.7: the carried scalar action is read as a bounded endomorphism and then absorbed on the left. The final displayed clause is Definition 213.5, applied to the carried ambient operator A and the trace-class endpoint T.

The exactness row in the root source excludes any further ideal operation at this threshold. Thus the displayed additive, scalar, and absorption clauses are the full ideal surface available through $\mathsf{TC}(H)$. □

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass root ideal absorption threshold`

Theorem 213.11 (OperatorIdeal root downstream-face boundary)**.** *The downstream chapters* $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, *and* $\mathsf{QuantumChannel}^{\uparrow}$ *may consume from the root* BHist *threshold for* $\mathsf{OperatorIdeal}^{\uparrow}(H)$ *only the carrier* $\mathsf{TC}(H)$*, the classifier transport* $\sim_{\mathsf{Op}(H)}$*, the zero, addition, negation, scalar, and two-sided absorption rows, and the bounded Hilbert source ledger. They receive no positivity row, no trace row, no trace-one row, no compactness row, no spectral-theorem row, no trace-existence row, and no standard bridge row from this package.*

Proof. Start with the exactness field of Definition 213.8. It lists the carrier, classifier, stability, ledger, and core rows and states that no accepted endpoint exposes anything beyond those rows.

The carrier and classifier part is fixed by Definition 213.9. The ideal-operation part is fixed by Definition 213.10: zero, addition, negation, scalar closure, and two-sided absorption are the available trace-class rows.

The earlier dependency boundary Definition 213.6 already separates $\mathsf{DensityMatrix}^{\uparrow}$ and $\mathsf{QuantumChannel}^{\uparrow}$ from positivity, trace, trace-one, compactness, spectral, trace-existence, and standard bridge claims. The root source names $\mathsf{Nuclear}^{\uparrow}$ as a consumer of the same $\mathsf{TC}(H)$ face, so the identical boundary applies to that consumer as well. □

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass root downstream face boundary`

Theorem 213.12 (Trace-class binary linear-combination closure)**.** *Let* H *be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. Write* $+_{\mathsf{TC}(H)}$ *for the trace-class addition inherited from the operator-ideal additive row, and use the real trace-class scalar action* $\odot_H$ *of Definition 213.7. Let* T *and* S *be operator endpoints, and let* a *and* b *be carried* $\mathsf{Real}^{\uparrow}$ *scalar endpoints for the Hilbert module action. If*

$$\mathsf{TC}(H)(T) \wedge \mathsf{TC}(H)(S),$$

then

$$\mathsf{TC}(H)\big((a \odot_H T) +_{\mathsf{TC}(H)} (b \odot_H S)\big).$$

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass binary linear combination closure`

Proof. Assume the two trace-class witnesses. Applying Definition 213.7 to the carried scalar a and the endpoint T gives

$$\mathsf{TC}(H)(a \odot_H T).$$

Applying the same scalar-closure row to the carried scalar b and the endpoint S gives

$$\mathsf{TC}(H)(b \odot_H S).$$

The operation $+_{\mathsf{TC}(H)}$ is the restriction of the bounded-operator addition row named in Definition 213.1 to endpoints carrying the trace-class predicate of Definition 213.2. Therefore the additive closure row of Definition 213.4, applied to the two scalar-multiple witnesses displayed above, gives

$$\mathsf{TC}(H)\big((a \odot_H T) +_{\mathsf{TC}(H)} (b \odot_H S)\big).$$

Only the trace-class ideal membership, scalar closure, and additive closure rows are used; positivity, trace, and trace-one rows remain outside this operator-ideal claim. □

Definition 213.13 (OperatorIdeal trace-class consumer spine source)**.** *Let* H *be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. The trace-class consumer spine for* $\mathsf{OperatorIdeal}^{\uparrow}(H)$ *is the concrete downstream-facing package with the following rows.*

1. *The carrier row is* $\mathsf{TC}(H)$.
2. *The classifier row is the restricted operator classifier* $\sim_{\mathsf{Op}(H)}$ *on certified bounded* $\mathsf{LinMap}^{\uparrow}(H, H)$ *endpoints.*
3. *The algebra rows are the trace-class zero endpoint, addition, negation, real scalar action* $\odot_H$*, and binary linear-combination closure.*
4. *The absorption rows are left and right composition by any carried bounded Hilbert operator.*
5. *The ledger row is the bounded Hilbert source ledger: the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *carrier, the bounded* $\mathsf{LinMap}^{\uparrow}(H, H)$ *certificate, the trace-class* $\mathsf{Ideal}^{\uparrow}$ *membership witness, and the two bounded-composition witnesses.*
6. *The only consumer slots are* $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, *and* $\mathsf{QuantumChannel}^{\uparrow}$.

No positivity, trace, trace-one, nuclearity, complete-positivity, compactness, spectral, trace-existence, or standard bridge row is part of this source.

Theorem 213.14 (OperatorIdeal trace-class consumer spine coverage)**.** *Every row of the trace-class consumer spine of Definition 213.13 is covered by the root* BHist *threshold source of Definition 213.8, the carrier-classifier threshold of Definition 213.9, the ideal-absorption threshold of Definition 213.10, and the binary linear-combination closure of Definition 213.12.*

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass consumer spine coverage`

Proof. We first build the carrier and classifier part of the spine. The source in Definition 213.8 names $\mathsf{TC}(H)$ as the trace-class subcarrier over bounded Hilbert operators, and Definition 213.9 identifies the visible classifier as $\sim_{\mathsf{Op}(H)}$ on certified bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ endpoints.

Next we build the algebra and absorption rows. The zero, addition, negation, scalar action, and two-sided composition rows are exactly the displayed rows of Definition 213.10. The binary linear-combination row is Definition 213.12, which repacks scalar closure and addition into the two-term downstream form.

It remains to account for the ledger and consumer slots. The ledger row is one of the named rows of Definition 213.8: it records the Hilbert source certificate, the bounded linear-map certificate, the trace-class ideal-membership witness, and the two bounded-composition witnesses. The same source names $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, and $\mathsf{QuantumChannel}^{\uparrow}$ as the downstream consumers, so every spine row is a projection of the cited root package and its displayed threshold theorems. □

Theorem 213.15 (OperatorIdeal trace-class consumer row separation)**.** *The trace-class consumer spine of Definition 213.13 contains none of the following downstream rows:*

1. *positivity, trace, and trace-one rows for* $\mathsf{DensityMatrix}^{\uparrow}$*;*
2. *nuclearity and trace-existence rows for* $\mathsf{Nuclear}^{\uparrow}$*;*
3. *complete-positivity and standard bridge rows for* $\mathsf{QuantumChannel}^{\uparrow}$*;*
4. *compactness and spectral rows for analytic refinements of the trace-class face.*

Each listed row must be supplied by the corresponding consumer chapter rather than imported from $\mathsf{OperatorIdeal}^{\uparrow}(H)$.

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealBoundedContextAction row separation`

Proof. Inspect the source rows in Definition 213.13. The enumerated spine contains only $\mathsf{TC}(H)$, the restricted classifier $\sim_{\mathsf{Op}(H)}$, the zero, addition, negation, scalar, binary linear-combination, and two-sided absorption rows, together with the bounded Hilbert ledger and the three consumer slots.

The same absence is forced by the upstream boundary. Definition 213.11 states that the root threshold sends no positivity, trace, trace-one, compactness, spectral, trace-existence, or standard bridge row downstream. The definition of the spine is a subpackage of that boundary, so it cannot contain a row excluded by the root face.

For nuclearity and complete positivity, the separation is by row name and dependency. Nuclearity is not one of the ideal-membership, bounded-composition, additive, scalar, carrier, or classifier rows in the spine. Complete positivity is likewise absent because the spine contains no matrix-amplification or positive-cone preservation row. Therefore the named analytic and quantum rows must be supplied by $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, and $\mathsf{QuantumChannel}^{\uparrow}$ at their own paper sites. □

Theorem 213.16 (OperatorIdeal trace-class consumer exhaustion)**.** *Every allowed downstream use of* $\mathsf{OperatorIdeal}^{\uparrow}(H)$ *by* $\mathsf{DensityMatrix}^{\uparrow}$*,* $\mathsf{Nuclear}^{\uparrow}$*, or* $\mathsf{QuantumChannel}^{\uparrow}$ *factors through exactly the trace-class carrier and classifier rows, the additive, scalar, and absorption rows, the binary linear-combination closure, and the bounded Hilbert source ledger displayed in Definition 213.13.*

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass consumer exhaustion`

Proof. Let a downstream use be accepted by one of the three consumer slots in Definition 213.13. By the definition of the slot, the use must read an endpoint through $\mathsf{TC}(H)$, compare endpoints through $\sim_{\mathsf{Op}(H)}$, apply one of the displayed algebra or absorption rows, or inspect the bounded Hilbert ledger.

The factorization is obtained by cases on that row access. Carrier and classifier accesses factor through Definition 213.9. Zero, addition, negation, scalar action, and two-sided absorption factor through Definition 213.10. A two-term scalar linear combination factors through Definition 213.12. Ledger access factors through the ledger row of Definition 213.8.

No remaining case exists. The exactness clause of Definition 213.8 says that the root threshold exposes no row beyond its carrier, classifier, stability, ledger, and core rows, and Definition 213.15 removes the analytic and quantum rows from the spine. Thus every allowed downstream use is exhausted by the listed carrier, classifier, operation, absorption, closure, and ledger factors. □

Definition 213.17 (OperatorIdeal bounded-context action spine)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2. A bounded-context action spine over H is the finite freely generated context package* BCtx_H *with the following constructors:*

1. *the identity context* $\mathsf{id}_{\mathsf{BCtx}}$*;*
2. *if $K \in \mathsf{BCtx}_H$ and* $\mathsf{Op}_H(A)$*, the left-composition context* $\mathsf{L}_A(K)$*;*
3. *if $K \in \mathsf{BCtx}_H$ and* $\mathsf{Op}_H(A)$*, the right-composition context* $\mathsf{R}_A(K)$*.*

Its endpoint action on a trace-class input T is defined by structural recursion:

$$\mathsf{id}_{\mathsf{BCtx}}[T] = T,$$

$$\mathsf{L}_A(K)[T] = A \circ K[T],$$

and

$$\mathsf{R}_A(K)[T] = K[T] \circ A.$$

Thus BCtx_H *records exactly a finite list of certified bounded Hilbert-operator compositions around a trace-class endpoint, with no positivity, trace, nuclearity, complete-positivity, compactness, spectral, or bridge row.*

Theorem 213.18 (OperatorIdeal trace-class finite context closure)**.** *Let* H *be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, let* $\mathsf{TC}(H)$ *be the trace-class carrier of Definition 213.2, and let* $K \in \mathsf{BCtx}_H$ *be a bounded-context action spine of Definition 213.17. If* $\mathsf{TC}(H)(T)$*, then the endpoint obtained by applying* K *to* T *is again trace-class:*

$$\mathsf{TC}(H)(T) \Longrightarrow \mathsf{TC}(H)(K[T]).$$

Proof. Proceed by structural induction on the bounded-context action spine K.

In the identity case, the endpoint action is $\mathsf{id}_{\mathsf{BCtx}}[T] = T$. The required trace-class witness is exactly the input witness $\mathsf{TC}(H)(T)$.

For the left-composition step, write the context as $\mathsf{L}_A(K_0)$ with $\mathsf{Op}_H(A)$. The induction hypothesis gives $\mathsf{TC}(H)(K_0[T])$. Apply the left component of Definition 213.5 to the carried bounded operator A and the trace-class endpoint $K_0[T]$; the result is

$$\mathsf{TC}(H)(A \circ K_0[T]).$$

By the endpoint action defining $\mathsf{L}_A(K_0)[T]$, this is the desired witness for $\mathsf{TC}(H)(\mathsf{L}_A(K_0)[T])$.

For the right-composition step, write the context as $\mathsf{R}_A(K_0)$ with $\mathsf{Op}_H(A)$. The induction hypothesis again gives $\mathsf{TC}(H)(K_0[T])$. Apply the right component of Definition 213.5 to A and $K_0[T]$ to obtain

$$\mathsf{TC}(H)(K_0[T] \circ A).$$

The endpoint action for $\mathsf{R}_A(K_0)$ identifies this endpoint with $\mathsf{R}_A(K_0)[T]$. These three cases exhaust the finite spine constructors. □

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass finite context closure`

Theorem 213.19 (OperatorIdeal consumer context exhaustion)**.** *Every downstream use by* $\mathsf{DensityMatrix}^{\uparrow}$*,* $\mathsf{Nuclear}^{\uparrow}$*, or* $\mathsf{QuantumChannel}^{\uparrow}$ *that starts from a trace-class endpoint and repeatedly composes it with finitely many carried bounded Hilbert operators factors through the bounded-context action spine of Definition 213.17 and the finite context closure theorem of Definition 213.18.*

Lean4 ✓ `BEDC.Derived.OperatorIdealUp.OperatorIdealBoundedContextAction consumer context exhaustion`

Proof. Let the downstream use be accepted by one of the consumer slots of Definition 213.13, and assume that its operator action consists only of finitely many left or right compositions by carried bounded Hilbert operators.

First encode the action as a bounded-context spine. The empty list of compositions is the identity constructor. Adding a left composition by a carried $\mathsf{Op}_H(A)$ applies the constructor L_A, and adding a right composition by the same kind of carried operator applies the constructor R_A. Finite repetition therefore builds an element $K \in \mathsf{BCtx}_H$ whose endpoint action is the downstream endpoint.

Next apply Definition 213.18. Starting from the input witness $\mathsf{TC}(H)(T)$, the theorem gives $\mathsf{TC}(H)(K[T])$, so the finite bounded-context action preserves the trace-class carrier exposed to the consumer.

It remains to account for the consumer rows. Carrier, classifier, additive, scalar, binary linear-combination, absorption, and ledger accesses are exactly the rows exhausted by Definition 213.16. The repeated-composition part of such an access is the bounded-context spine just constructed,

while analytic rows remain absent by Definition 213.15. Hence no allowed $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, or $\mathsf{QuantumChannel}^{\uparrow}$ bounded-context use lies outside the displayed spine and closure package. □

Closure status for $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$.

Constructive story: An operator ideal is generated over a certified Hilbert carrier by first admitting bounded operator histories and then restricting to the trace-class subcarrier witnessed by an $\mathsf{Ideal}^{\uparrow}$ membership row inside the bounded-operator composition ring. The carrier accepts a trace-class endpoint only when a unary support history is displayed and continued by the empty tail to the endpoint, together with the Hilbert, linear-map, ideal-membership, scalar, and left/right composition rows; positivity, trace, trace-one, compactness, spectral data, and nuclearity are not accepted here. Two operator endpoints are classified through the Hilbert and linear-map classifiers, meaning pointwise agreement on carried Hilbert endpoints after both map-codes have bounded-operator evidence, with the trace-class witness transported along the same ideal row. The source is the $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, and $\mathsf{Ideal}^{\uparrow}$ surface, the pattern records zero, addition, negation, scalar closure, two-sided absorption, and finite bounded-context action, and the ledger keeps the bounded support, ideal membership, scalar action, composition witnesses, and consumer-row separation visible. The five obligations follow from the zero trace-class carrier, the equivalence behavior of the inherited operator classifier, and the finite-context closure theorem, which repacks every classified left or right composition as another trace-class endpoint. A history h lies in the signature gap of an $\mathsf{OperatorIdeal}^{\uparrow}$ token p when the source-domain accepts the Hilbert operator and ideal-membership rows, the signature relation pairs the trace-class endpoint with an answer endpoint and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records the same bounded support and finite ideal-action ledger. The chapter registers this trace-class operator-ideal surface under the name $\mathsf{OperatorIdeal}^{\uparrow}$; downstream analytic chapters may consume only these rows, with no trace functional, positivity predicate, CPTP row, or bridge data added.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature scope binds support, classifier transport, additive closure, scalar closure, two-sided absorption, finite bounded-context closure, consumer-row separation, exact consumer exhaustion, and the consumer-context exhaustion row supplied by Definition 213.19.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.OperatorIdealUp.OperatorIdealTraceClass\_consumer\_exhaustion`

Bridge status: none

Not claimed: Positivity, trace, trace-one, compactness, spectral theorem, nuclearity, CPTP rows, and bridge data remain outside this chapter.

Upgrade path: No further theory-axis upgrade remains for the displayed $\mathsf{OperatorIdeal}^{\uparrow}$ trace-class surface.

Chapter 214

A Concrete Naming Certificate for $\mathsf{SpectralTheorem}^{\uparrow}$

The spectral-theorem interface $\mathsf{SpectralTheorem}^{\uparrow}$ packages the diagonalisation data of a self-adjoint or normal bounded operator: spectrum, projection-valued measure, and functional calculus. It depends on $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91) and $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92).

214.1 Carrier definition

214.2 The certificate

Theorem 214.1 (SpectralTheoremUp carrier and classifier obligations)**.** *The* $\mathsf{SpectralTheorem}^{\uparrow}$ *certificate carrier is the* $\mathsf{Hilbert}^{\uparrow}$ *operator row restricted to displayed self-adjoint-or-normal operator witnesses. Its classifier row is read through the* $\mathsf{Hilbert}^{\uparrow}$ *operator classifier, together with* $\mathsf{Measure}^{\uparrow}$*-compatible spectral data: two carried operator histories are admissibly compared only when the operator classifier identifies them and the spectrum, projection-valued-measure, and functional-calculus rows are presented over the same measured spectral side.*

Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremEmptySpectrum calculus endpoint`
Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremOperatorSpectralData carrier classifier`

Proof. The obligation is the carrier-and-classifier portion of the roadmap ten-field template.

1. Traditional target: the spectral theorem for a self-adjoint or normal bounded operator.
2. BEDC source: the certified $\mathsf{Hilbert}^{\uparrow}$ operator package together with the certified $\mathsf{Measure}^{\uparrow}$ spectral-measure package.
3. Carrier: the $\mathsf{Hilbert}^{\uparrow}$ operator row, restricted by a displayed self-adjoint-or-normal witness.
4. Classifier: the $\mathsf{Hilbert}^{\uparrow}$ operator classifier, with comparison accepted only when the accompanying spectral rows are measured in the same $\mathsf{Measure}^{\uparrow}$-compatible fiber.
5. Stability: transport of the operator row and the self-adjoint-or-normal witness along the operator classifier.
6. Ledger: the hidden operator-domain witness and the measured spectral-side witness used by the comparison.

7. Core theorem list: carrier restriction, classifier readback, and compatibility of operator comparison with the spectral rows.
8. Exactness: coverage and soundness for precisely those displayed carrier and classifier rows.
9. Standard bridge: none.
10. Status: paper obligation.

Thus the carrier is not a host-level operator space and the classifier is not a host-level equality. Both are certificate rows inherited from the displayed $\mathsf{Hilbert}^{\uparrow}$ and $\mathsf{Measure}^{\uparrow}$ sources. □

Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremSingleton carrier classifier obligation`
Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremCarrierClassifier obligation`
Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremOperator measure row readback`

Theorem 214.2 (SpectralTheoremUp spectrum, projection measure, and functional-calculus obligations)**.** *The core theorem list for* $\mathsf{SpectralTheorem}^{\uparrow}$ *must supply spectrum rows, projection-valued-measure rows, and functional-calculus rows. Each row is scoped over the certified* $\mathsf{Hilbert}^{\uparrow}$ *operator carrier and the certified* $\mathsf{Measure}^{\uparrow}$ *measurable-side carrier, not over a host spectral theorem: the spectrum row displays the accepted spectral support, the projection-valued-measure row displays the measured projection assignment, and the functional-calculus row displays the calculus action compatible with both.*

Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremProjectionMeasure functional calculus empty`

Proof. The obligation is the spectral-data portion of the roadmap ten-field template.

1. Traditional target: spectrum, projection-valued measure, and bounded functional calculus for the carried operator.
2. BEDC source: $\mathsf{Hilbert}^{\uparrow}$ for the operator and projection endpoints, plus $\mathsf{Measure}^{\uparrow}$ for measurable spectral rows.
3. Carrier: the operator carrier of Definition 214.1, together with the displayed spectral-support carrier.
4. Classifier: operator comparison through the $\mathsf{Hilbert}^{\uparrow}$ classifier and measurable-row comparison through the $\mathsf{Measure}^{\uparrow}$ classifier.
5. Stability: transport of spectral support, projection rows, and calculus rows along accepted operator comparisons.
6. Ledger: the hidden spectrum witness, the hidden projection-valued-measure witness, and the hidden functional-calculus witness.
7. Core theorem list: spectrum coverage, projection-valued-measure assignment, calculus closure, and compatibility of calculus with the displayed projections.
8. Exactness: coverage, soundness, and completeness for the displayed spectrum, projection-valued-measure, and calculus rows.
9. Standard bridge: none.
10. Status: paper obligation.

The displayed rows are therefore obligations of the BEDC certificate itself. They do not import a completed external spectral theorem; they state the exact rows that later paper and Lean work must discharge inside the named sources. □

Theorem 214.3 (SpectralTheoremUp stability, ledger, and exactness obligations)**.** *For* $\mathsf{SpectralTheorem}^{\uparrow}$*, stability must transport the operator row, spectrum row, projection-valued-measure row, and functional-calculus row along the classifier of Definition 214.1. The ledger records the hidden spectrum, projection-valued-measure, and functional-calculus witnesses. Exactness is coverage, soundness, and completeness for those displayed rows, with dependency rows naming the* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{Measure}^{\uparrow}$ *sources.*

Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremCarrier classifier transport`

Proof. The obligation is the transport, provenance, and exactness portion of the roadmap ten-field template.

1. Traditional target: invariance of spectral data under equality of the carried operator.
2. BEDC source: $\mathsf{Hilbert}^{\uparrow}$ for operator and projection transport, plus $\mathsf{Measure}^{\uparrow}$ for measurable spectral transport.
3. Carrier: the restricted self-adjoint-or-normal operator carrier and the three spectral-data carriers.
4. Classifier: the operator classifier together with the compatible measurable-side classifier rows.
5. Stability: simultaneous transport of operator, spectrum, projection-valued-measure, and functional-calculus rows.
6. Ledger: hidden witnesses for spectral support, projection assignment, calculus action, and their dependency rows.
7. Core theorem list: transport rows, ledger readback rows, dependency rows, and rowwise exactness clauses.
8. Exactness: coverage, soundness, and completeness for the displayed operator, spectrum, projection-valued-measure, and functional-calculus rows.
9. Standard bridge: none.
10. Status: paper obligation.

Combining these rows with Definition 214.2 gives the full obligation boundary: every hidden spectral datum has a ledger row, every displayed row has an exactness clause, and every dependency is attached to either $\mathsf{Hilbert}^{\uparrow}$ or $\mathsf{Measure}^{\uparrow}$. □

Theorem 214.4 (SpectralTheoremUp public certificate)**.** *The carrier and classifier obligations of Definition 214.1, the spectrum, projection-valued-measure, and functional-calculus obligations of Definition 214.2, and the stability, ledger, exactness, and dependency obligations of Definition 214.3 assemble the public* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *export surface over the* $\mathsf{Hilbert}^{\uparrow}$ *operator source and the* $\mathsf{Measure}^{\uparrow}$ *spectral side. The exported surface is public as a BEDC naming certificate and does not claim a standard spectral theorem bridge.*

Lean4 ✓ `BEDC.Derived.SpectralTheoremUp.SpectralTheoremCarrier public certificate`

Proof. Apply Definition 214.1 to fix the carried operator row, the self-adjoint-or-normal witness, and the classifier row through the certified $\mathsf{Hilbert}^{\uparrow}$ operator source. This determines the operator side of the public export surface.

Apply Definition 214.2 to attach the spectrum, projection-valued-measure, and functional-calculus rows over the certified $\mathsf{Measure}^{\uparrow}$ spectral side. These rows provide the visible spectral data exported by $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$.

Finally apply Definition 214.3 to pack the transport, ledger, exactness, and dependency rows around the same carrier and classifier. The result is the public certificate surface for $\mathsf{SpectralTheorem}^{\uparrow}$; since no StandardBridge row is part of the package, the construction remains a public BEDC certificate and not a standard spectral theorem bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$.

Constructive story: A spectral-theorem package starts from a certified $\mathsf{Hilbert}^{\uparrow}$ operator row restricted by a displayed self-adjoint-or-normal witness, then appends measured spectral-side rows for spectrum, projection-valued measure, and functional calculus through the cited $\mathsf{Measure}^{\uparrow}$ source. The carrier accepts only histories whose operator, spectral support, projection, calculus, and provenance rows are displayed in this Hilbert-measure package, while a completed host spectral theorem or bridge-checked standard model is not part of the carrier. Two carried packages are classified when the Hilbert operator classifier identifies the operator rows and the spectral data live over the same measured side, so the comparison transports spectrum, projection-valued-measure, and calculus ledgers rather than equating ambient operators externally. The source is the paired $\mathsf{Hilbert}^{\uparrow}$ and $\mathsf{Measure}^{\uparrow}$ certificate surface, the pattern records the self-adjoint-or-normal restriction plus spectrum, projection, and calculus obligations, and the ledger keeps hidden spectrum, projection-valued-measure, calculus, dependency, and provenance witnesses visible. Since the restricted operator carrier is inhabited in the displayed singleton boundary, the inherited Hilbert and measure classifiers are equivalence rows, and the transport theorem repacks the spectral data over a classified endpoint, the five name-certificate obligations reduce to those componentwise rows. A package h lies in the signature gap of a $\mathsf{SpectralTheorem}^{\uparrow}$ token p when the source-domain accepts its Hilbert operator and measured spectral rows, the signature relation pairs them with an answer spectral package and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records the same operator together with its displayed spectrum, projection measure, and calculus rows. The chapter registers this Hilbert-measure spectral-data package under the name $\mathsf{SpectralTheorem}^{\uparrow}$; it closes the BEDC certificate surface without importing the full external spectral theorem.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature package is exactly the $\mathsf{Hilbert}^{\uparrow}$ source, the $\mathsf{Measure}^{\uparrow}$ source, the carrier and classifier rows of Definition 214.1, the spectral-data rows of Definition 214.2, the stability, ledger, exactness, and dependency rows of Definition 214.3, the public certificate theorem of Definition 214.4, and the standard-boundary rows for the exported spectral data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SpectralTheoremUp.SpectralTheoremCarrier\_classifier\_transport`

Bridge status: none

Not claimed: This chapter does not claim the full external spectral theorem or bridge-checked verification.

Upgrade path: Only bridge-checked verification remains outside the paper-side mature treatment.

Chapter 215

A Concrete Naming Certificate for $\mathsf{Nuclear}^{\uparrow}$

The nuclear-operator interface $\mathsf{Nuclear}^{\uparrow}$ packages compact operators between Banach spaces representable as absolutely convergent rank-one sums. It depends on $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90) and $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ (chapter 213).

215.1 Carrier definition

Theorem 215.1 (NuclearUp carrier obligation)**.** *The carrier for* $\mathsf{Nuclear}^{\uparrow}$ *is a* BHist *predicate packaging a* $\mathsf{Banach}^{\uparrow}$ *source endpoint, an* $\mathsf{OperatorIdeal}^{\uparrow}$ *compact-operator surface, and a unary rank-index ledger. The carried endpoint records the visible* BHist *rows, their* Pkg *provenance, and the rank-index observations used by the nuclear expansion; hidden representatives of the Banach spaces, compact operators, and rank-one terms are not carrier equality.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearCarrier obligation`

Proof. Unfold the carrier field of the ten-field certificate template. The named source object is $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, the operator surface is $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$, and the rank index is represented as a unary BHist ledger row.

Packing these three rows through BHist and Pkg gives the visible carrier predicate. Since all remaining data are placed behind the ledger row, the carrier equality is exactly the displayed BEDC package equality and not equality of host representatives. □

Theorem 215.2 (NuclearUp compact-operator obligation)**.** *The compactness row for* $\mathsf{Nuclear}^{\uparrow}$ *is read through certified* $\mathsf{OperatorIdeal}^{\uparrow}$ *approximation data and* $\mathsf{Banach}^{\uparrow}$ *classifier transport. It is not an appeal to external norm compactness: compactness is visible only when the* $\mathsf{OperatorIdeal}^{\uparrow}$ *surface supplies the approximation row and the* $\mathsf{Banach}^{\uparrow}$ *endpoints transport it along the displayed* hsame *classifier.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearCompactOperator prefix transport obligation`
Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearCompactOperator banach classifier transport`
↪ variant `BEDC.Derived.NuclearUp.NuclearCompactOperator banach transport surface`

Proof. Unfold the compact-operator field of the certificate template after Definition 215.1. The available objects are the $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ approximation row and the $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ classifier on source and target histories.

The construction transports approximation observations along the $\mathsf{Banach}^{\uparrow}$ classifier and then repacks them as the compact row of the carried BHist endpoint. No norm-compactness principle is introduced, because the only compactness witness consumed by the template is the certified operator-ideal approximation row. □

215.2 The certificate

Theorem 215.3 (NuclearUp rank-one-sum obligation)**.** *Rank-one expansion data for* $\mathsf{Nuclear}^{\uparrow}$ *supplies unary* BHist *indices, coefficient and vector observations, a tail modulus, and* Cont*-compatible partial sums. The partial-sum rows are stable under* hsame*, so changing the displayed Banach histories by the certified classifier preserves the carried nuclear expansion package.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearRankOnePrefixLedger semantic name certificate`

Proof. Unfold the expansion field of the ten-field template. Each unary index names one rank-one row, with its scalar coefficient, source vector observation, target vector observation, and tail-modulus entry recorded as BHist data.

The Cont row is then applied to the finite partial sums generated by those indices. Stability follows by transporting the coefficient and vector observations along hsame before repacking the same tail-modulus ledger, so the expansion remains a carried nuclear package. □

Theorem 215.4 (NuclearUp ledger exactness obligation)**.** *The* $\mathsf{Nuclear}^{\uparrow}$ *ledger hides exactly rank indices, rank-one projections, and tail-modulus evidence. Its exactness row must supply coverage for every visible nuclear package, soundness for every ledger entry projected to the visible carrier, and completeness saying that no visible compact rank-one-sum package is lost by the hidden ledger representation.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearLedgerExactness obligation`

Proof. Unfold the ledger and exactness fields of the certificate template. The hidden rows are precisely the unary rank index, the rank-one projection data named by that index, and the tail-modulus witness used by the expansion row.

Coverage reads a visible nuclear package back into such ledger rows. Soundness projects any ledger row to the BHist carrier of Definition 215.1, and completeness states that the projection captures all visible compact rank-one-sum packages admitted by Definition 215.2 and Definition 215.3. □

Theorem 215.5 (NuclearUp dependency boundary)**.** *Downstream consumers of* $\mathsf{Nuclear}^{\uparrow}$ *may use only the* $\mathsf{Banach}^{\uparrow}$ *interface, the* $\mathsf{OperatorIdeal}^{\uparrow}$ *interface, and the rank-one ledger specified in this chapter. They may not use a trace-class principle, a spectral principle, choice, or host compactness as part of the* $\mathsf{Nuclear}^{\uparrow}$ *certificate.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearDependencyBoundary obligation`

Proof. Unfold the dependency field of the ten-field template. The declared dependencies are exactly $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$, and the unary rank-one ledger rows fixed by Definition 215.3.

Every public observation is obtained by projecting those named dependencies through BHist, Pkg, hsame, and Cont. Since trace-class, spectral, choice, and host compactness principles are not dependency rows, they cannot be consumed by the certified $\mathsf{Nuclear}^{\uparrow}$ surface. □

Theorem 215.6 (NuclearUp carrier compact-operator obligation)**.** *A* $\mathsf{Nuclear}^{\uparrow}$ *carrier must expose a* $\mathsf{Banach}^{\uparrow}$ *source pair, an* $\mathsf{OperatorIdeal}^{\uparrow}$ *compact-operator row, and a carrier predicate over* BHist *for operators represented by certificate-visible finite prefixes of compact maps. The source pair names the domain and codomain Banach carriers, the compact-operator row records that each visible prefix endpoint is compact in the cited operator-ideal surface, and the carrier predicate admits only histories whose operator evidence factors through those finite prefixes.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearCompactPrefixCarrier banach operator rows`

Proof. Project the $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ row named in the introduction to obtain the source and target Banach carriers and their classifier transport fields.

Next project the $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ dependency row. Its compact-operator entry is the only operator-ideal evidence exposed to the nuclear carrier at this level, so every admitted operator history must be witnessed by a finite prefix of compact maps visible in that row.

The carrier predicate is then the BEDC history predicate that stores exactly these three projections: Banach source, Banach target, and compact-prefix evidence. No host operator or infinite expansion is admitted without an accompanying finite prefix row. □

Theorem 215.7 (NuclearUp rank-one-sum ledger obligation)**.** *The* $\mathsf{Nuclear}^{\uparrow}$ *pattern row records each rank-one summand, coefficient-bound witness, finite-prefix certificate, and convergence ledger entry. Consequently no unrecorded infinite sum or host sequence is used by the certificate surface.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearRankOnePrefixLedger tail readback`

Proof. Read the pattern row as a finite BEDC ledger. The empty prefix carries no summand and no endpoint beyond the zero prefix certified by the dependency rows.

For a nonempty prefix, project its head row. It contains the rank-one summand, its coefficient-bound witness, the finite-prefix endpoint, and the ledger entry that relates the endpoint to the previous prefix.

Apply induction over the finite prefix spine. The induction hypothesis accounts for every earlier summand and bound, while the head projection accounts for the current row. Since every accepted prefix is built by these constructors, an infinite host sequence can be referenced only through the displayed convergence ledger and never as unrecorded data. □

Theorem 215.8 (NuclearUp OperatorIdeal and Banach dependency obligation)**.** *Stability and exactness for* $\mathsf{Nuclear}^{\uparrow}$ *are scoped only to the cited* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ *rows: carrier transport uses Banach classifier transport, ideal closure uses the* $\mathsf{OperatorIdeal}^{\uparrow}$ *compactness and ideal-law rows, and exactness is the visible rank-one ledger iff the nuclear classifier.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearOperatorIdealBanachDependency rank rows`

Proof. For carrier transport, use only the two Banach classifiers named by the source pair. Transport of a carried nuclear operator changes the visible source or target history only along those $\mathsf{Banach}^{\uparrow}$ classifier witnesses.

For ideal closure, project the compactness and ideal-law rows from $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$. Composition or finite linear recombination is accepted exactly when the cited operator-ideal laws produce the next compact-prefix witness.

For exactness, unfold the nuclear classifier as equality of the visible rank-one ledger modulo the Banach and operator-ideal transports just named. The forward direction projects the ledger equivalence from the classifier witness. The reverse direction repacks the same ledger equivalence, together with the Banach transport and compact-prefix witnesses, as the nuclear classifier. □

Theorem 215.9 (NuclearUp scoped BEDC boundary)**.** *The carrier, compact-prefix, rank-one-ledger, and dependency rows of Definition 215.6, Definition 215.7, and Definition 215.8 are scoped exactly to* BHist, Pkg, hsame, Cont, $\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^\uparrow}$, *unary rank-index rows, rank-one projection rows, compact-prefix witnesses, coefficient-bound witnesses, and tail-modulus ledger rows. No trace-class row, spectral row, choice principle, host compactness datum, or host infinite-sequence datum is part of this boundary.*

Lean4 ✓ `BEDC.Derived.NuclearUp.NuclearScopedBHistBoundary rows`

Proof. Project the carrier and compact-prefix rows of Definition 215.6. The accepted data are the BHist carrier predicate, its Pkg provenance, Banach classifier transport through hsame, the $\mathsf{OperatorIdeal}^\uparrow$ compact row, and the compact-prefix witnesses exposed by that operator-ideal surface.

Project the pattern row of Definition 215.7. Each admitted prefix contributes only a unary rank-index row, a rank-one projection row, a coefficient-bound witness, a compact-prefix endpoint, and a tail-modulus ledger row, with Cont used only for the finite partial-sum observation named by those rows.

Finally project the dependency rows of Definition 215.8. They name exactly $\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^\uparrow}$ as external certificate inputs, so every accepted dependency is already one of the displayed BEDC rows. Since trace-class, spectral, choice, host compactness, and host infinite-sequence data are never produced by these projections, they are outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Nuclear}^\uparrow}$.

Constructive story: A nuclear operator is generated from a pair of certified $\mathsf{Banach}^\uparrow$ source and target histories, a compact-operator row supplied by $\mathsf{OperatorIdeal}^\uparrow$, and a unary rank-index ledger that records finite rank-one prefixes. The carrier accepts only BHist endpoints whose operator evidence factors through those visible compact prefixes and whose expansion ledger displays coefficients, vector observations, rank-one projections, partial sums, and tail-modulus rows; trace-class principles, spectral principles, choice, host compactness, and unrecorded infinite sequences are excluded. Two nuclear packages are classified by Banach source and target transport together with the visible rank-one ledger and compact-prefix witnesses, so changing representatives means repacking the same finite expansion rows under hsame rather than comparing host operators. The source is the $\mathsf{Banach}^\uparrow$ and $\mathsf{OperatorIdeal}^\uparrow$ certificate pair, the pattern records compact-prefix support and rank-one-sum ledger exactness, and the ledger keeps unary rank indices, coefficient bounds, vector rows, compact-prefix endpoints, partial-sum continuations, and tail moduli visible. The five obligations hold because an admitted compact-prefix package supplies habitation, the Banach and operator-ideal classifiers give reflexive, symmetric, and transitive transport, and rank-ledger exactness proves that classified finite prefixes remain admitted. A package h lies in the signature gap of a $\mathsf{Nuclear}^\uparrow$ token p when the source-domain accepts the Banach endpoints, compact-operator row, and rank-one ledger, the signature relation pairs the visible nuclear package with an answer package and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records the same compact prefix and coefficient-tail modulus data. The chapter registers this compact-prefix rank-one construction under the name $\mathsf{Nuclear}^\uparrow$ as a paper-side bridge boundary for the exported public surface, while trace-class principles, spectral principles, and machine-checked standard-reference comparison remain outside the chapter.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Nuclear}^\uparrow}$, maturePackageClosed)

Scope closed: The public surface exports the $\mathsf{Banach}^\uparrow$ source pair, $\mathsf{OperatorIdeal}^\uparrow$ compact-operator row, rank-index ledger, BHist carrier, hsame transport, Cont-compatible prefix rows, and NameCert boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Nuclear}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.NuclearUp.NuclearCompactPrefixCarrier\_banach\_operator\_rows`

Bridge status: paperBridge

Not claimed: Machine-checked standard bridge, trace-class theorem, spectral principle, choice principle, and host compactness theorem are not closed here.

Upgrade path: Only a machine-checked standard bridge can raise the formal bridge axis; trace-class and spectral principles remain outside the chapter.

Chapter 216

A Concrete Naming Certificate for $\mathsf{Factor}^{\uparrow}$

The factor interface $\mathsf{Factor}^{\uparrow}$ packages a von Neumann algebra with trivial centre, classified into types I, II_1, II_∞, III. It depends on $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ (chapter 212).

216.1 Carrier definition

Definition 216.1 (Factor BHist source packet)**.** *A* $\mathsf{Factor}^{\uparrow}$ *source packet is a* BHist *row carrying a* $\mathsf{VonNeumannAlg}^{\uparrow}$ *dependency row, a displayed centre row, a trivial-centre witness, and a factor-type classifier row with value in* I, II_1, II_∞, *or* III. *The carrier accepts exactly those packets whose algebra component is accepted by* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$, *whose centre witness is transported by the same* hsame *evidence, and whose type row is recorded in the visible ledger.*

Theorem 216.2 (Factor carrier trivial-centre obligation)**.** *Every accepted* $\mathsf{Factor}^{\uparrow}$ *carrier row projects to a* $\mathsf{VonNeumannAlg}^{\uparrow}$ *carrier row together with a centre witness whose only visible carrier is the scalar endpoint of the source algebra. Conversely, a* $\mathsf{VonNeumannAlg}^{\uparrow}$ *carrier row, a displayed centre row, and a trivial-centre witness transported through the same* hsame *comparison repack as a* $\mathsf{Factor}^{\uparrow}$ *carrier row. No host equality of algebras or ambient Hilbert-space realization is read by this obligation.*

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket trivial centre obligation`
Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket carrier trivial centre obligation`

Proof. Project the dependency row from the source packet of Definition 216.1. The $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ row supplies the algebra carrier and the classifier used to compare its visible history.

The trivial-centre field is then read as a displayed witness over that same history. Transport along hsame preserves the centre row because the packet stores the centre witness beside the algebra dependency rather than as a host property.

For the converse direction, place the algebra dependency, centre row, and trivial-centre witness into the packet constructor and apply Cont to the shared history comparison. The result is exactly the factor carrier surface. □

Theorem 216.3 (Factor type classifier stability obligation)**.** *The factor-type classifier row of a* $\mathsf{Factor}^{\uparrow}$ *packet is stable under accepted* hsame *replacement of the underlying* $\mathsf{VonNeumannAlg}^{\uparrow}$ *history. If two source packets compare through the algebra classifier, the transported trivial-centre witnesses determine the same displayed type row among* I, II_1, II_∞, *and* III.

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket type classifier stability obligation`

Proof. Start with the algebra comparison supplied by $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$. It transports the carrier, operation, and classifier rows of the source von Neumann algebra through the accepted hsame evidence.

The factor packet keeps the type classifier in the same ledger as the trivial-centre witness. Therefore the transported centre witness is read before the type row is compared, so the classification cannot change without leaving the displayed packet.

Repacking the transported algebra row, centre witness, and type row gives the same $\mathsf{Factor}^{\uparrow}$ classifier surface for the target packet. □

Theorem 216.4 (Factor ledger exactness obligation)**.** *The* $\mathsf{Factor}^{\uparrow}$ *ledger consists exactly of the* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency row, the* BHist *algebra carrier row, the centre row, the trivial-centre witness row, the factor-type classifier row, and their* hsame *and* Cont *transport evidence. Coverage, soundness, and completeness hold for precisely those rows; spectral decomposition theorems, representation theorems, measure-class data, and standard bridge rows are excluded.*

Proof. Coverage projects an accepted packet into its visible ledger entries: the von Neumann algebra dependency, the algebra carrier, the centre row, the trivial-centre witness, the factor-type row, and the transport witnesses.

Soundness reads those entries back through the packet constructor. The dependency certificate justifies the algebra component, while hsame and Cont carry the centre and type rows across the accepted history comparison.

Completeness follows because the packet has no ledger coordinate for spectral decomposition, representation, measure class, or bridge data. Any accepted row therefore exposes exactly the listed entries and no larger analytic surface. □

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket ledger exactness obligation`

Theorem 216.5 (Factor root threshold unblock surface)**.** *The rows of Definition 216.1, Definition 216.2, Definition 216.3, and Definition 216.4 form the dependency-ready surface for* $\mathsf{Factor}^{\uparrow}$*. A downstream certificate may consume the* BHist *carrier,* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency, trivial-centre witness, factor-type classifier, and exact ledger exclusions, but it must supply any standard bridge or analytic classification theorem separately.*

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket root threshold unblock surface`

Proof. The definition fixes the source packet and names every row that can enter the factor certificate. The carrier obligation recovers the algebra dependency and the trivial-centre witness from any accepted packet.

The stability obligation gives the classifier transport needed by downstream NameCert rows. The ledger exactness obligation then closes the readback by excluding every row outside the displayed packet.

Combining these three obligations gives a concrete finite-history interface over BHist rather than an empty chapter shell. Later bridges may attach to this surface, but they are not part of the factor packet itself. □

216.2 The certificate

Theorem 216.6 (Factor type row no-confusion)**.** *An accepted* $\mathsf{Factor}^{\uparrow}$ *packet cannot expose two distinct visible factor-type rows under the same transported trivial-centre witness. If the packet carries a* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency, a* BHist *algebra carrier, a centre row, a trivial-centre witness transported by the displayed* hsame *evidence, and visible type rows* T *and* T'*, then the certificate ledger identifies* T *and* T' *as the same row among* I, II_1, II_∞, *and* III.

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket type row no confusion`

Proof. Project the accepted packet through Definition 216.2. The projection gives one von Neumann algebra dependency and one transported trivial-centre witness over the displayed centre row.

Now read the type coordinates through Definition 216.3. The classifier row is stored in the same ledger as the transported centre witness, so two visible type rows under that witness must be compared by the same hsame evidence.

Finally apply Definition 216.4. Since the ledger has exactly one factor-type coordinate and no hidden representation or spectral coordinate, the two displayed type rows are the same certificate row. □

Theorem 216.7 (Factor centre ledger transport determinacy)**.** *For a* $\mathsf{Factor}^{\uparrow}$ *packet,* hsame *transport of the* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency, the* BHist *algebra carrier, the centre row, the trivial-centre witness, and the* Cont *provenance determines the transported* $\mathsf{Factor}^{\uparrow}$ *centre ledger. No additional representation row, spectral row, or ambient Hilbert-space equality can change the transported ledger.*

Proof. Start from the dependency row named by Definition 216.1. Its $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ component transports the algebra carrier through the displayed hsame evidence.

Transport the centre row and the trivial-centre witness along the same history comparison. The Cont provenance records that these rows move together as one factor packet rather than as independent host data.

The ledger exactness theorem Definition 216.4 then leaves no further centre coordinate to choose. The transported dependency, carrier, centre, trivial-centre witness, and Cont provenance therefore determine the transported centre ledger. □

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket centre ledger transport determinacy`

Theorem 216.8 (Factor VonNeumann dependency consumer boundary)**.** *Every downstream consumer of the* $\mathsf{Factor}^{\uparrow}$ *certificate may read only the* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency, the* BHist *algebra carrier, the centre row, the trivial-centre witness, the factor-type classifier, and the* $\mathsf{hsame}/\mathsf{Cont}$ *ledger. It may not read representation data, spectral decomposition data, measure-class data, ambient Hilbert-space equality, or any row not displayed by the factor source packet.*

Proof. The source packet of Definition 216.1 lists the complete visible input surface: dependency, carrier, centre, trivial-centre witness, type classifier, and transport provenance.

Apply Definition 216.4 to the consumer read. The coverage direction supplies exactly those visible rows, while soundness and completeness repack only rows of the same listed kinds.

The excluded analytic data have no coordinate in the packet and no entry in the ledger. Hence a consumer that remains inside the $\mathsf{Factor}^{\uparrow}$ certificate can read the displayed dependency and centre surface, but cannot inspect representation or spectral information. □

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket vonneumann dependency consumer boundary`

Theorem 216.9 (FactorUp public NameCert export)**.** *The source packet of Definition 216.1, the trivial-centre carrier obligation of Definition 216.2, the classifier stability obligation of Definition 216.3, the ledger exactness obligation of Definition 216.4, and the root threshold surface of Definition 216.5 together export the public* $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$ *surface.*

Consumers of this certificate may read only the $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency row, the centre row, the trivial-centre witness, the factor-type classifier, the* hsame/Cont *transport rows, and the ledger exclusions listed by the packet. Representation theory, spectral decomposition, measure-class data, and standard bridge rows remain outside the certificate.*

Proof. The source packet fixes the only rows available to $\mathsf{Factor}^{\uparrow}$: a $\mathsf{VonNeumannAlg}^{\uparrow}$ dependency, a displayed centre row, a trivial-centre witness, a factor-type classifier, and the shared transport ledger.

Apply the trivial-centre carrier obligation to project an accepted packet to the von Neumann algebra carrier and its scalar-centre witness. Apply the classifier stability obligation to transport the displayed factor type along the same hsame comparison.

The ledger exactness obligation then says that the visible rows are exactly the dependency, centre, trivial-centre, classifier, hsame, and Cont entries. The root threshold surface packages these rows for downstream consumers, so the public $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$ surface contains precisely those entries and no representation-theoretic, spectral, measure-class, or bridge datum. □

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket public namecert export`

Theorem 216.10 (FactorUp public bridge boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$ *surface is exactly the BEDC-side boundary available to any later standard bridge packet. It exposes the* $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ *dependency, the centre row, the trivial-centre witness, the factor-type classifier, the* hsame/Cont *transport rows, and the ledger exclusions.*

It exposes no representation data, spectral decomposition, measure-class data, ambient Hilbert-space equality, or bridge row unless those data are supplied by a separate bridge packet.

Proof. Apply Definition 216.9. The exported public surface already packages the von Neumann algebra dependency, centre row, trivial-centre witness, classifier, transport provenance, and ledger exclusions.

Next apply Definition 216.8 to any consumer that reads the certificate as the input face of a standard bridge. The consumer boundary permits exactly the displayed dependency and centre surface and blocks representation, spectral, measure-class, and Hilbert-space equality data.

Finally apply Definition 216.4. Ledger exactness says that no additional public row is available inside the $\mathsf{Factor}^{\uparrow}$ source packet. Hence bridge data can enter only through a separate bridge packet, not through the public $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$ surface itself. □

Lean4 ✓ `BEDC.Derived.FactorUp.FactorBHistSourcePacket public bridge boundary`

Theorem 216.11 (Factor bridge consumer completeness)**.** *Any finite consumer of the displayed* $\mathsf{Factor}^{\uparrow}$ *bridge surface reads exactly the dependency* $\mathsf{VonNeumannAlg}^{\uparrow}$ *row, algebra carrier, centre row, transported trivial-centre witness, factor-type classifier, and shared* hsame/Cont/Pkg *ledger already exported by Definition 216.9 and Definition 216.10.*

It cannot read a host von Neumann algebra, central decomposition, trace theorem, or ambient factor classification.

Proof. Apply Definition 216.8. It fixes the consumer boundary at the displayed $\mathsf{VonNeumannAlg}^{\uparrow}$ dependency, algebra carrier, centre row, and trivial-centre witness.

Next apply Definition 216.9. The public certificate adds the factor-type classifier and the shared hsame/Cont/Pkg ledger, and ledger exactness prevents any hidden classifier or representation row from being read.

Finally apply Definition 216.10. The bridge-facing surface is precisely the same public packet, so a finite consumer receives no host von Neumann algebra, central decomposition, trace theorem, or ambient factor classification. □

Closure status for $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Factor}^{\uparrow}$ row is built as a finite BHist packet whose visible coordinates are the $\mathsf{NameCert}_{\mathsf{VonNeumannAlg}^{\uparrow}}$ dependency, the algebra carrier, the centre row, the transported trivial-centre witness, the factor-type classifier, and the shared hsame/Cont ledger. The carrier theorem reads the dependency and trivial-centre witness directly from that packet, while the surrounding certificate theorems record classifier stability, ledger exactness, consumer boundaries, and the public bridge-facing surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface is the listed BHist source packet, trivial-centre carrier, classifier, $\mathsf{VonNeumannAlg}^{\uparrow}$ consumer rows, exported public $\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$ certificate, and the cited Lean target.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Factor}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FactorUp.FactorBHistSourcePacket\_public\_bridge\_boundary`

Bridge status: paperBridge

Not claimed: This chapter does not close spectral decomposition, representation theorems, measure-class data, or standard bridge rows.

Upgrade path: No stronger paper theory grade remains; bridgeChecked is a separate verification-axis target.

The $\mathsf{QuantumState}^{\uparrow}$ hub consumes the Hilbert source boundary of chapter 91 and the ProjectiveSpace phase classifier of chapter 207, then exposes readback boundaries to chapter 218, chapter 220, and chapter 221.

Chapter 217

A Concrete Naming Certificate for $\mathsf{QuantumState}^{\uparrow}$

The quantum-state interface $\mathsf{QuantumState}^{\uparrow}$ packages a unit vector in a Hilbert space carrier together with the equivalence under global phase. It depends on $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91) and $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$ (chapter 207).

217.1 Carrier definition

Definition 217.1 (QuantumStateUp BHist carrier)**.** *A* $\mathsf{QuantumState}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho).$$

The row $\mathcal{H}$ *is a consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *row. The row* $\mathcal{P}$ *is a consumed* $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$ *row over the displayed projective line determined by* $\mathcal{H}$*. The endpoint* v *is the unit-vector row projected from* $\mathcal{H}$*,* n *is the norm-one ledger row,* ϕ *is the global-phase row, and* p *is the projective-line endpoint row projected from* $\mathcal{P}$*.*

The rows λ_H *and* λ_P *are the inherited Hilbert and projective classifier rows, respectively. The row* ρ *is the* Pkg *provenance record witnessing that the two consumed certificates and all displayed endpoint rows belong to the same finite packet. No carrier datum outside* $\mathcal{H}$*,* $\mathcal{P}$*,* v*,* n*,* ϕ*,* p*,* λ_H*,* λ_P*, and* ρ *is part of the public* $\mathsf{QuantumState}^{\uparrow}$ *row.*

217.2 The certificate

Theorem 217.2 (QuantumStateUp Hilbert source boundary)**.** *For every carrier packet* $\mathcal{Q}$ *of Definition 217.1, all vector, norm, and inner-product rows used by the* $\mathsf{QuantumState}^{\uparrow}$ *interface are projected from the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *row* $\mathcal{H}$*. In particular, the visible vector endpoint* v*, the norm-one ledger* n*, and every inner-product row used to test unit length are read through* $\mathcal{H}$ *and the inherited classifier* λ_H*, not through an ambient Hilbert-space carrier.*

Proof. The traditional target is a Hilbert-space unit vector carrying the usual norm and inner-product data. The BEDC source is the consumed $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ row $\mathcal{H}$, and the carrier choice is the finite packet $\mathcal{Q}$ of Definition 217.1.

Unpack the definition of $\mathcal{Q}$. The only fields naming Hilbert-side data are $\mathcal{H}$, v, n, and λ_H, together with the provenance row ρ. Thus the classifier choice is the inherited Hilbert classifier λ_H, stability is hsame transport through that classifier, and the ledger is the norm-one row n.

The core theorem list for this boundary consists of the visible endpoint projection, the norm-one ledger projection, and the inner-product projection. Exactness says that every Hilbert-side row visible to $\mathsf{QuantumState}^{\uparrow}$ is one of those projected rows. There is no standard bridge supplied here, and the status is a paper obligation for the $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ Hilbert source boundary. □

Theorem 217.3 (QuantumStateUp projective phase classifier)**.** *The classified* $\mathsf{QuantumState}^{\uparrow}$ *rows are exactly the Hilbert unit-vector rows* v *whose projective endpoints* p *are related by the global-phase row* ϕ *and the* $\mathsf{ProjectiveSpace}^{\uparrow}$ *classifier* λ_P*. Equivalently, two carrier packets*

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho)$$

and

$$\mathcal{Q}' = (\mathcal{H}', \mathcal{P}', v', n', \phi', p', \lambda'_H, \lambda'_P, \rho')$$

classify the same quantum state precisely when the Hilbert endpoint rows are unit-vector rows and the projective endpoints are identified by the displayed global-phase/projective-space classifier rows. The classifier records no host vector equality, no independent Hilbert-space quotient, and no primitive quotient object beyond the projective scaling classifier supplied by $\mathcal{P}$*.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier projective phase classifier`

Proof. The traditional target is equality of pure quantum states modulo global phase. The BEDC source is the pair of consumed certificates $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$. The carrier choice is the packet $\mathcal{Q}$, while the classifier choice is the pair formed by the global-phase row ϕ and the projective classifier λ_P.

For the forward reading, a classified $\mathsf{QuantumState}^{\uparrow}$ row first exposes a Hilbert endpoint v and its norm-one ledger n through Definition 217.2. It then exposes the projective endpoint p through $\mathcal{P}$. The row ϕ records the allowed phase change, and λ_P records the projective classifier that accepts the resulting endpoint comparison.

For the reverse reading, a Hilbert unit-vector endpoint with a norm-one ledger and a projective endpoint accepted by ϕ and λ_P supplies all fields of Definition 217.1. Stability is componentwise hsame transport through λ_H and λ_P, the ledger is n together with the phase row ϕ, and exactness says that no host quotient is read. The core theorem list is this classifier row together with the Hilbert source boundary. The standard bridge is absent, and the status is a paper obligation for the $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ classifier. □

Theorem 217.4 (QuantumStateUp global phase stability)**.** *Let* $\mathcal{Q}$ *and* $\mathcal{Q}'$ *be two carrier packets of Definition 217.1. If the Hilbert rows* $\mathcal{H}, v, n, \lambda_H$ *and the projective rows* $\mathcal{P}, p, \lambda_P$ *are transported componentwise by* hsame*, then the global-phase classifier row is preserved and the norm-one ledger remains a valid* $\mathsf{QuantumState}^{\uparrow}$ *ledger row. Thus* hsame *transport of the consumed Hilbert and projective certificates preserves the phase classifier and the norm-one condition.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier global phase stability`

Proof. The traditional target is invariance of a pure state under change of certified presentation. The BEDC source is the pair of source certificates $\mathcal{H}$ and $\mathcal{P}$, the carrier choice is the packet $\mathcal{Q}$, and the classifier choice is λ_H, λ_P, and the phase row ϕ.

Transport the Hilbert side first. The hsame witnesses for $\mathcal{H}$, v, n, and λ_H move the unit-vector endpoint and the norm-one ledger through the Hilbert classifier without opening any host vector equality.

Transport the projective side next. The hsame witnesses for $\mathcal{P}$, p, and λ_P move the projective endpoint and its classifier row. The phase row ϕ is preserved because its source and target endpoints are exactly the transported Hilbert and projective endpoints.

The ledger consists of the norm-one row, the phase row, and the provenance row. The core theorem list is the Hilbert source boundary, the projective phase classifier, and this transport row. Exactness says that every preserved row is one of the displayed packet fields. No standard bridge is supplied, and the status is a paper obligation for $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ stability. □

Theorem 217.5 (QuantumStateUp public obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *surface consists exactly of the carrier packet of Definition 217.1, the projective phase classifier of Definition 217.3, the stability row of Definition 217.4, the norm-one ledger, the projective exactness predicate, and the hidden global-phase rows needed to witness the endpoint comparison. These rows form the carrier, classifier, stability, ledger, exactness, and hidden global-phase obligations exported by* $\mathsf{QuantumState}^\uparrow$*. The surface contains no density-matrix row, measurement row, channel row, or spectral row.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier namecert obligation surface`

Proof. The traditional target is the pure-state naming certificate. The BEDC source is the pair $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^\uparrow}$, and the carrier choice is the displayed packet $\mathcal{Q}$. The classifier choice is the global-phase/projective classifier, while stability is Definition 217.4.

The ledger is finite: it contains the norm-one row n, the phase row ϕ, the two inherited classifier rows λ_H and λ_P, and the provenance row ρ. The core theorem list is the Hilbert source boundary, the projective phase classifier, and the global-phase stability row.

Exactness says that every public $\mathsf{QuantumState}^\uparrow$ row is assembled from those fields and that every hidden phase row is used only to classify projective endpoints. The named dependency certificates expose no density-matrix, measurement, channel, or spectral row at this site. The standard bridge field is empty. The status is a paper obligation for the $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ public surface. □

Theorem 217.6 (QuantumStateUp observable measurement input boundary)**.** *An* $\mathsf{Observable}^\uparrow$ *expectation row may consume a* $\mathsf{QuantumState}^\uparrow$ *input only through the Hilbert row* $\mathcal{H}$*, the projective endpoint* p*, the phase classifier* ϕ*, the norm-one ledger* n*, the package provenance* ρ*, and the displayed* hsame/Cont *transport rows exported by Definition 217.5. The expectation row cannot inspect host vector equality, an ambient quotient object, or a phase witness except through the* $\mathsf{QuantumState}^\uparrow$ *classifier used to compare the projective endpoint.*

Proof. The observable expectation source is the row E in Definition 218.1. By Definition 218.5, that row is read through $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ and is transported only by the displayed classifier ledger of the observable packet.

Now read the public surface of Definition 217.5. Its exported state input rows are exactly the Hilbert source $\mathcal{H}$, the projective endpoint p, the phase classifier ϕ, the norm-one ledger n, the provenance row ρ, and the hsame/Cont transport rows attached to those fields.

The exclusion clauses follow from exactness of the same surface. Host vector equality is not a field of Definition 217.1, quotient equality is replaced by the projective classifier, and hidden phase data are visible only through the classifier row. Thus an observable measurement input has precisely the displayed BHist boundary. □

Theorem 217.7 (QuantumStateUp density-matrix pure-state boundary)**.** *Any* $\mathsf{DensityMatrix}^\uparrow$ *pure-state row sourced from a* $\mathsf{QuantumState}^\uparrow$ *carrier packet must factor through the projective*

endpoint p and the norm-one ledger n of that packet. The hidden phase data of the source state are visible only through the $\mathsf{QuantumState}^{\uparrow}$ *phase classifier ϕ and projective classifier λ_P, and no mixed-state ledger, affine-mixture spine, trace-class classifier, spectral row, or channel row is imported back into the* $\mathsf{QuantumState}^{\uparrow}$ *carrier.*

Proof. Use Definition 217.3 to identify the source pure state by its unit Hilbert endpoint and projective endpoint rather than by host vector equality. The norm side is fixed by Definition 217.4: hsame transport of the Hilbert and projective rows preserves the phase classifier and the norm-one ledger.

A density-matrix row then reads only the target density obligations. The carrier and ledger clauses of Definition 220.1 and Definition 220.3 describe trace-class membership, positivity, trace one, and the restricted trace-class comparison on the density side.

Those density rows are consumer rows, not source-state fields. The public $\mathsf{QuantumState}^{\uparrow}$ surface of Definition 217.5 contains no density-matrix row or affine-mixture ledger, so the only source information available to a pure-state density row is the projective endpoint together with the norm-one and phase-classifier data already carried by $\mathsf{QuantumState}^{\uparrow}$. □

Theorem 217.8 (QuantumStateUp quantum-channel pure input boundary)**.** $\mathsf{QuantumChannel}^{\uparrow}$ *may act on a* $\mathsf{QuantumState}^{\uparrow}$ *pure input only after the associated density or projective input row has been read through the same public* $\mathsf{QuantumState}^{\uparrow}$ *surface described in Definition 217.5. The channel output is a density-matrix endpoint produced by the channel carrier and density-image rows; it does not identify source states by host Hilbert-vector equality or by any equality outside the* $\mathsf{QuantumState}^{\uparrow}$ *projective classifier and the density-matrix classifier it consumes.*

Proof. First pass the pure input through the two preceding boundaries. The observable boundary fixes the permitted $\mathsf{QuantumState}^{\uparrow}$ input rows, while the density-matrix pure-state boundary says that a density input sourced from a pure state factors through the projective endpoint and norm-one ledger.

Now apply the channel surface. By Definition 221.2, a channel acts on trace-class density endpoints, and Definition 221.3 sends carried density inputs to carried density outputs. For finite density mixtures, Definition 221.8 preserves the density affine-spine rows without opening the source state carrier.

Therefore a channel may read the pure input only after it has been presented as the permitted density or projective input row. The output comparison lives in the density-matrix and channel ledgers, while source-state comparison remains the $\mathsf{QuantumState}^{\uparrow}$ projective classifier transported through $\mathsf{hsame}/\mathsf{Cont}$ rows. □

Theorem 217.9 (Observable expectation readback)**.** *Every expectation row in an* $\mathsf{Observable}^{\uparrow}$ *packet that consumes a* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *state input is read from a displayed* BHist *and* $\mathsf{NameCert}$ *packet consisting only of the Hilbert source row $\mathcal{H}$, the projective endpoint row p, the global-phase classifier ϕ, the norm ledger n, and the provenance row ρ of Definition 217.1. The observable expectation site may pair those rows with the observable endpoint rows of chapter 218, but it receives no host Hilbert-vector equality, no quotient object, and no density-matrix or channel row from* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$.

Proof. The downstream observable input is the expectation-source row named in chapter 218. Read its state component through the public $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ surface of Definition 217.5. That surface exposes exactly the carrier packet of Definition 217.1, the projective classifier, the norm ledger, and the provenance row.

Now apply Definition 217.5. It gives the complete consumer frontier for $\mathsf{Observable}^{\uparrow}$: expectation evaluation may use the Hilbert source row, projective endpoint, phase classifier, norm ledger, and provenance. The observable endpoint itself is supplied by the observable chapter, so no extra state equality, quotient representative, density row, or channel row is imported from the state certificate. □

Theorem 217.10 (DensityMatrix pure-state readback)**.** *Every pure-state density row assembled by* $\mathsf{DensityMatrix}^{\uparrow}$ *from a* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *input is assembled only from the same displayed unit-vector, projective-phase, norm, and provenance packet of Definition 217.1, together with the density-matrix carrier boundary of chapter 220. The readback does not expose a host vector equality, a quotient representative, or a spectral decomposition as an additional state-side input.*

Proof. The source packet is the $\mathsf{QuantumState}^{\uparrow}$ carrier of Definition 217.1. Its public certificate surface is fixed by Definition 217.5: the available state rows are the Hilbert source, projective endpoint, phase classifier, norm ledger, and provenance rows, with hidden phase data readable only through the classifier.

The density-matrix side supplies its own trace-class, positivity, and trace-one carrier rows in chapter 220. Combining that carrier boundary with Definition 217.5 leaves only the unit-vector/projective-phase packet as the state-side readback. Thus the pure-state density row is assembled from the certified state packet plus the density carrier boundary, not from host vector equality, quotient representatives, or spectral data. □

Theorem 217.11 (QuantumChannel input readback)**.** *A* $\mathsf{QuantumChannel}^{\uparrow}$ *input may consume a* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *packet only through the certified density and expectation boundary determined by Definition 217.1, Definition 217.5, and Definition 217.5. Its channel-side rows in chapter 221 may refer to the state only through those certified readbacks; they may not appeal to host Hilbert-vector equality, quotient equality, or an unrecorded representative.*

Proof. The channel input is downstream of the density-matrix and observable interfaces named in chapter 220 and chapter 218. Those interfaces can read the state only through the public rows of Definition 217.5, namely the carrier packet of Definition 217.1 and its classifier, ledger, and provenance rows.

Apply Definition 217.5 at the $\mathsf{QuantumChannel}^{\uparrow}$ site of chapter 221. The certified density and expectation readbacks exhaust the permitted state access. Since host vector equality, quotient equality, and unrecorded representatives are not rows of the public state packet, they cannot be consumed by the channel input. □

Theorem 217.12 (QuantumStateUp density-matrix readback boundary)**.** *A* $\mathsf{DensityMatrix}^{\uparrow}$ *consumer that reads back a rank-one density row from a public* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *packet may read only the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the norm ledger* n*, the phase classifier* ϕ*, and the package provenance* ρ *exported by Definition 217.1. Any rank-one density row sourced from the state must factor through the projective classifier of Definition 217.3; it may not use host Hilbert-vector equality as a public input.*

Proof. Project the state packet $\mathcal{Q}$ through Definition 217.1. The readable rows are exactly the Hilbert source, projective endpoint, norm ledger, phase classifier, and provenance fields already exposed by Definition 217.5.

The rank-one readback step is then forced through the projective classifier. Apply Definition 217.3 to convert the hidden phase witness into the public endpoint comparison, and use Definition 217.4 to keep that comparison stable under the same classified packet.

Finally, package the allowed rows through BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ provenance. No density row adds a host vector equality: the only equality available to the consumer is the projective classifier row exported by the public surface. □

Theorem 217.13 (QuantumStateUp observable expectation source)**.** *An* $\mathsf{Observable}^{\uparrow}$ *expectation row sourced from a public* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *packet uses the observable packet together with exactly the* $\mathsf{QuantumState}^{\uparrow}$ *Hilbert source row* $\mathcal{H}$ *and projective endpoint row* p*. The phase witness remains hidden behind the classifier* ϕ*, and the expectation source may read no host vector equality, density-matrix row, measurement outcome row, or ambient function-space equality from* $\mathsf{QuantumState}^{\uparrow}$*.*

Proof. Begin with the observable packet and the public state packet $\mathcal{Q}$. The state side contributes only the Hilbert and projective rows displayed in Definition 217.1, together with the norm ledger and provenance fields named by Definition 217.5.

The phase data is not passed as a separate witness. Use Definition 217.3 to expose only the classified endpoint row, and use Definition 217.4 to show that the observable source is insensitive to changing representatives inside the same classified state packet.

The expectation source is therefore repackaged through BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ provenance using only those rows. Since no other $\mathsf{QuantumState}^{\uparrow}$ field is projected, no density, measurement, or ambient function-space equality is introduced. □

Theorem 217.14 (QuantumStateUp channel input exactness)**.** $\mathsf{QuantumChannel}^{\uparrow}$ *input transport is exact for the public* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *packet. A classified input packet exposes the same Hilbert source row* $\mathcal{H}$*, projective endpoint row* p*, norm ledger* n*, phase row* ϕ*, and package provenance* ρ *as the source packet, and it adds no density-matrix row, measurement row, or ambient function-space equality.*

Proof. Project the source packet through Definition 217.1. The transported input is classified by the same public rows: Hilbert source, projective endpoint, norm ledger, phase classifier, and provenance.

Exactness follows by using Definition 217.5 as the complete list of exported state rows. Apply Definition 217.3 to keep the projective endpoint as the channel-visible state classifier, and apply Definition 217.4 to preserve the phase row under classified transport.

The channel input is then repacked through BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ provenance. Because the repacking uses only the displayed packet fields, the input surface is exact and no density-matrix, measurement, or ambient function-space equality is added. □

Theorem 217.15 (QuantumStateUp density-matrix consumer exclusion)**.** *A* $\mathsf{DensityMatrix}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *may read only the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the global-phase classifier* ϕ*, the norm ledger* n*, and the package provenance* ρ *from the pure-state certificate. The trace-class row, positivity row, and trace-one row required for a density matrix must be supplied by* $\mathsf{DensityMatrix}^{\uparrow}$ *itself. They are not fields of the finite* BHist *packet* $\mathcal{Q}$ *and are not consequences of the pure-state package provenance alone.*

Proof. Unpack the carrier packet of Definition 217.1. Its public rows are $\mathcal{H}$, $\mathcal{P}$, v, n, ϕ, p, λ_H, λ_P, and ρ, with the consumer-readable boundary restricted by Definition 217.5.

Compare this finite list with the public obligation surface of Definition 217.5. The only rows exported to a downstream construction are the Hilbert source, projective endpoint, phase classifier, norm ledger, and provenance rows. None of the displayed fields is a trace-class carrier, a positivity witness, or a trace-one ledger.

Thus a $\mathsf{DensityMatrix}^{\uparrow}$ construction may use the pure-state certificate as its input boundary, but its density rows must enter through its own BHist packet and Pkg provenance. Reading those rows from $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ would require a field absent from the carrier and from the obligation surface. □

Theorem 217.16 (QuantumStateUp observable measurement readback boundary)**.** *An* $\mathsf{Observable}^{\uparrow}$ *measurement readback over a* $\mathsf{QuantumState}^{\uparrow}$ *input factors through the Hilbert classifier row* λ_H *and the projective phase classifier formed by* ϕ *and* λ_P*. The outcome row, spectral row, and observable package provenance are supplied by* $\mathsf{Observable}^{\uparrow}$ *rather than by the state certificate, and cannot be read as hidden rows of* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$*.*

Proof. Project the consumer-readable rows described in Definition 217.5. The state input gives a Hilbert source row, a projective endpoint row, a phase classifier, a norm ledger, and the package provenance for the pure-state packet.

Measurement readback uses these rows only to classify the input state. The Hilbert part is read through λ_H, while the projective part is read through ϕ and λ_P as in Definition 217.3.

The observable rows have different provenance. Outcome and spectral data are introduced by the observable packet and its own Pkg row, so they are separate from the BHist fields listed in Definition 217.1. Therefore the readback boundary factors through the state classifiers and leaves observable output rows to $\mathsf{Observable}^{\uparrow}$. □

Theorem 217.17 (QuantumStateUp channel phase-classifier preservation)**.** *A* $\mathsf{QuantumChannel}^{\uparrow}$ *consumer transporting a pure-state input by* hsame *and* Cont *ledgers must preserve the displayed projective phase classifier of* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$*. It may repack the Hilbert source, projective endpoint, phase classifier, norm ledger, and provenance rows as its channel input boundary, but it may not replace the phase classifier by host vector equality or by equality in an ambient quotient object.*

Proof. Start with the hsame transport row of Definition 217.4. It preserves the Hilbert source rows, the projective endpoint rows, the norm-one ledger, and the global-phase classifier through the displayed carrier fields.

The classifier being preserved is the one characterized by Definition 217.3. That theorem identifies the public comparison with the projective endpoint row and the phase classifier, not with host Hilbert-vector equality and not with an ambient quotient equality.

Finally, repack the allowed rows using Definition 217.5. The channel side may attach its own Cont ledger to transport the input boundary, but the incoming phase row remains the displayed $\mathsf{QuantumState}^{\uparrow}$ classifier. Any replacement by host equality would add a row absent from the carrier packet and from the public obligation surface. □

Theorem 217.18 (QuantumStateUp observable expectation input exactness)**.** *An* $\mathsf{Observable}^{\uparrow}$ *expectation consumer that receives* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *may read exactly the state rows* $\mathcal{H}$*,* v*,* n*,* p*,* ϕ*,* λ_H*,* λ_P*, and* ρ *from the carrier packet of Definition 217.1. The expectation input is the* BHist *endpoint* v *scoped by the Hilbert row* $\mathcal{H}$*, the norm-one ledger* n*, the projective endpoint* p*, the global-phase row* ϕ*, the inherited classifier rows* λ_H *and* λ_P*, and the* Pkg *provenance row* ρ*. It may not read an observable spectrum row, measurement-result row, host-level vector equality, or state data outside the displayed* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *classifier.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier observable expectation input exactness`

Proof. Unpack the carrier packet of Definition 217.1. The rows available to any consumer are the two source certificates, the Hilbert endpoint, the norm-one ledger, the phase and projective endpoint data, the two inherited classifiers, and the finite provenance package. The observable expectation site uses the state side of this packet, so its state input is the listed eight rows

$$\mathcal{H}, v, n, p, \phi, \lambda_H, \lambda_P, \rho.$$

The classifier is fixed by Definition 217.3. Thus v is not read as a raw host vector: it is read as a Hilbert BHist endpoint whose admissible comparisons pass through λ_H, ϕ, and λ_P. The stability available to the expectation consumer is exactly Definition 217.4, so changing certified presentations transports the same input rows by hsame rather than by a host quotient.

Exactness follows from Definition 217.5 together with Definition 217.5. The public surface contains no spectral, measurement, density-matrix, or channel row, and the consumer boundary forbids equality outside the Hilbert and projective classifiers. Therefore the observable expectation input is precisely the displayed BHist and Pkg row family. □

Theorem 217.19 (QuantumStateUp density-matrix pure ray readback)**.** *A* $\mathsf{DensityMatrix}^{\uparrow}$ *pure-state consumer reading* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *may recover only the projective ray endpoint* p *together with the Hilbert source row* $\mathcal{H}$*, the endpoint row* v*, the norm-one ledger* n*, the global-phase row* ϕ*, the inherited classifiers* λ_H, λ_P*, and the provenance row* ρ*. Its readback is a certified pure ray in* $\mathsf{BHist}/\mathsf{hsame}$ *terms, not a host vector quotient and not an already supplied density-matrix row.*

Proof. Start with the carrier packet

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho).$$

The pure-density consumer needs a ray source and a unit ledger. These are already the projective endpoint p and the norm-one Hilbert ledger n, with v serving only as the certified Hilbert endpoint from which the ray row is projected.

Apply Definition 217.3 to identify the ray comparison used by the consumer. The global-phase row ϕ and the projective classifier λ_P determine sameness of pure rays, while λ_H scopes the Hilbert endpoint and norm ledger. Apply Definition 217.4 to transport this readback along hsame without changing the rows that the consumer may inspect.

The exclusion is the constructive content. By Definition 217.5, no density-matrix row is exported by $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$. By Definition 217.5, no host vector quotient or equality outside the displayed classifiers is available. Thus a pure-state density-matrix consumer can build its downstream row only from the certified ray endpoint, the norm-one Hilbert ledger, and the same Pkg provenance record. □

Theorem 217.20 (QuantumStateUp projective ray non-escape)**.** *Every pure-state downstream consumer reading* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *factors through the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the norm-one ledger* n*, the global-phase row* ϕ*, the inherited classifiers* λ_H, λ_P*, and the provenance row* ρ*. The state packet exports no host vector quotient, ambient projective equality, density-matrix row, observable spectrum row, or channel row.*

Proof. Start with the projective phase classifier of Definition 217.3. It identifies pure-state sameness through the displayed projective endpoint and global-phase classifier, not through a host quotient of vectors or an ambient projective equality.

Next use global phase stability from Definition 217.4. Transport along hsame preserves the Hilbert source, projective endpoint, norm-one ledger, phase row, inherited classifiers, and provenance row as the same finite packet data.

Apply the public obligation surface of Definition 217.5. Its exactness clause lists the exported $\mathsf{QuantumState}^{\uparrow}$ rows and does not include density-matrix data, observable spectra, or channel rows.

Finally use the density-matrix pure ray readback of Definition 217.19. Even a density-matrix consumer of a pure ray receives only the projective ray endpoint, norm-one Hilbert ledger, classifiers, and provenance supplied by the state packet. Thus every pure-state consumer remains inside the displayed ray route, and all excluded rows must be supplied by separate downstream certificates. □

Theorem 217.21 (QuantumStateUp channel consumer input boundary)**.** *A* $\mathsf{QuantumChannel}^{\uparrow}$ *state-input consumer of* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *is stable under the same global-phase/projective classifier as* $\mathsf{QuantumState}^{\uparrow}$*. Its input rows are limited to* $\mathcal{H}$*,* v*,* n*,* p*,* ϕ*,* λ_H*,* λ_P*, and* ρ *from the carrier packet; it carries no independent channel row, no density-matrix row, and no classifier other than the inherited* BHist*,* hsame*,* Pkg*, and* NameCert *data already exported by the state certificate.*

Proof. Unpack Definition 217.1 and read the state input as a finite BHist package. The endpoint supplied to a channel is v together with its norm-one ledger n, its projective endpoint p, the phase row ϕ, the inherited classifiers λ_H, λ_P, and the provenance row ρ tying these rows to the consumed source certificates.

The classifier for this input is not chosen by the channel consumer. It is the classifier of Definition 217.3: Hilbert endpoint transport is scoped by λ_H, and ray transport is scoped by ϕ and λ_P. The stability required by the channel input is exactly Definition 217.4, which keeps the same finite row family under componentwise hsame transport.

Finally apply Definition 217.5 and Definition 217.5. The public surface exports no channel datum and no density-matrix datum, while the consumer boundary forbids a separate quotient or host equality. Repacking the displayed rows therefore gives the whole channel state-input boundary in BHist/hsame/Pkg/NameCert terms. □

Theorem 217.22 (QuantumStateUp observable state boundary)**.** *An* $\mathsf{Observable}^{\uparrow}$ *expectation row may read a* $\mathsf{QuantumState}^{\uparrow}$ BHist *packet only through the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the global-phase classifier* ϕ*, the norm ledger* n*, and the package provenance* ρ *exposed by Definition 217.5, together with the observable endpoint, self-adjoint witness, spectral-data row, and expectation row displayed by* $\mathsf{Observable}^{\uparrow}$*. It may not replace the certified state input by host vector equality, a raw quotient representative, or an expectation row whose Hilbert source is not the shared* $\mathcal{H}$ *endpoint.*

Proof. Project the accepted quantum-state packet through Definition 217.5. The only state-side rows visible to a downstream consumer are $\mathcal{H}$, p, ϕ, n, and ρ.

Now apply the expectation boundary of Definition 218.5. The observable packet contributes its operator endpoint, self-adjoint witness, spectral-data row, and expectation row, and that theorem requires the expectation input to be compatible with the same Hilbert source.

Combining the two projections gives the displayed boundary. Any host vector equality, quotient representative, or Hilbert-mismatched expectation row would have to enter through a field not present in one of the two cited packet surfaces, so it is not readable by the consumer. □

Theorem 217.23 (QuantumStateUp density-matrix boundary)**.** *A* $\mathsf{QuantumState}^{\uparrow}$ BHist *packet does not itself supply a* $\mathsf{DensityMatrix}^{\uparrow}$ *row. A density-matrix consumer of a pure projective state*

must add, as separate displayed rows, trace-class membership, positivity, trace-one evidence, and any rank-one projection bridge that turns the projective endpoint into a positive trace-class trace-one operator. These rows are not part of the Hilbert source, projective endpoint, phase classifier, norm ledger, or provenance exposed by Definition 217.5.

Proof. Read the quantum-state packet fields from Definition 217.5. They certify the Hilbert source, projective endpoint, phase classifier, norm ledger, and package provenance, but no field has the form of a trace-class operator on the Hilbert carrier.

Compare this with the density carrier obligation of Definition 220.1. A $\mathsf{DensityMatrix}^{\uparrow}$ endpoint is accepted only when trace-class membership, positivity, and trace-one evidence are all displayed over the Hilbert carrier.

Therefore a pure-state-to-density-matrix use must introduce the trace-class, positivity, trace-one, and rank-one projection bridge rows at the consumer site. The quantum-state packet can supply the named source and projective endpoint for such a bridge, but it does not supply the density row itself. □

Theorem 217.24 (QuantumStateUp quantum-channel input boundary)**.** *A* $\mathsf{QuantumChannel}^{\uparrow}$ *consumer may use a* $\mathsf{QuantumState}^{\uparrow}$ *packet as an input only after a displayed* $\mathsf{DensityMatrix}^{\uparrow}$*-compatible input row has been supplied for the same Hilbert source. Channel image closure can then read the* $\mathsf{DensityMatrix}^{\uparrow}$ *row and the CPTP carrier of the channel, but it cannot read host vector equality, a raw pure-state quotient, or a direct channel action on the projective endpoint.*

Proof. First apply Definition 217.23. The quantum-state packet alone has no density-matrix row, so a channel input must pass through a separately displayed trace-class, positivity, trace-one, and rank-one projection bridge package.

Next use the CPTP carrier of Definition 221.2. Its source type is the trace-class carrier over the input Hilbert space, and its density-state action is stated over $\mathsf{DensityMatrix}^{\uparrow}$ inputs rather than over projective representatives.

Finally apply the density-matrix image lemma Definition 221.3. Once the compatible density row is present, the channel image closure reads exactly that row and the CPTP channel rows. The excluded host equality, quotient representative, and direct projective action are outside the displayed density and channel carrier surfaces. □

Theorem 217.25 (ObservableUp expectation-row boundary)**.** *When* $\mathsf{Observable}^{\uparrow}$ *forms expectation or eigenvalue rows from a* $\mathsf{QuantumState}^{\uparrow}$ *source, it may read only the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the global-phase classifier* ϕ*, the norm ledger* n*, and the provenance row* ρ *of Definition 217.1. The observable consumer may combine those rows with its own observable certificate, but it receives no density-matrix row, measurement row, channel row, spectral row, quotient object, or host Hilbert-vector equality from* $\mathsf{QuantumState}^{\uparrow}$ *itself.*

Proof. Unpack the carrier packet $\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho)$ of Definition 217.1. The rows available to an observable construction are the Hilbert source row, the projected endpoint, the phase classifier, the norm ledger, and the provenance row.

Apply the Hilbert source boundary of Definition 217.2 to locate all vector, norm, and inner-product inputs inside $\mathcal{H}$ and λ_H. Apply Definition 217.3 and Definition 217.4 to replace any phase choice by the projective endpoint classifier.

The public surface theorem, Definition 217.5, identifies the complete exported row list. Expectation and eigenvalue rows therefore require a separate $\mathsf{Observable}^{\uparrow}$ certificate for observable, measurement, or spectral data. Since none of those rows occurs in $\mathcal{Q}$, and since quotient and

host-vector equalities are excluded by the classifier boundary, the displayed rows are the whole $\mathsf{QuantumState}^{\uparrow}$ input boundary for this consumer. □

Theorem 217.26 (DensityMatrixUp rank-one row boundary)**.** *When* $\mathsf{DensityMatrix}^{\uparrow}$ *forms a rank-one projector row from a* $\mathsf{QuantumState}^{\uparrow}$ *source, it may read only the projective endpoint row* p *and the norm-one ledger* n*, together with the provenance row* ρ *that records their common packet. The density-matrix consumer may not read a host quotient, an ambient vector equality, a measurement row, a channel row, a spectral row, or any density-matrix row unless that row is supplied by its own downstream certificate.*

Proof. Unpack Definition 217.1. The rank-one projector input is not a new field of $\mathcal{Q}$; the packet exposes only the projective endpoint p, the norm-one ledger n, the inherited classifiers, and the provenance row ρ.

Use Definition 217.2 to read n through the Hilbert source row rather than through an ambient vector equality. Use Definition 217.3 to read p through the projective endpoint classifier, and use Definition 217.4 to ensure that replacing a unit-vector representative by a certified phase-equivalent representative does not change the exposed projector input.

Finally apply Definition 217.5. Its exactness clause contains no density-matrix, measurement, channel, or spectral row, and it contains no quotient object or host-vector equality. Thus any matrix row, operator row, or spectral decomposition must come from the $\mathsf{DensityMatrix}^{\uparrow}$ certificate itself, while $\mathsf{QuantumState}^{\uparrow}$ contributes only the projective endpoint, norm ledger, and provenance needed to form the rank-one boundary. □

Theorem 217.27 (QuantumChannelUp pure-state input boundary)**.** *When* $\mathsf{QuantumChannel}^{\uparrow}$ *consumes pure-state input or output rows from a* $\mathsf{QuantumState}^{\uparrow}$ *source, it may use the exported* $\mathsf{QuantumState}^{\uparrow}$ *classifier and provenance only: the Hilbert source row* $\mathcal{H}$*, the projective endpoint row* p*, the global-phase classifier* ϕ*, the norm ledger* n*, and the provenance row* ρ*. Channel action, density-matrix data, measurement data, spectral data, quotient objects, and host Hilbert-vector equalities must be supplied by the channel-side certificate rather than by* $\mathsf{QuantumState}^{\uparrow}$*.*

Proof. Unpack the packet of Definition 217.1. A pure-state input or output row can name the state only through the Hilbert source row, the projective endpoint, the phase classifier, the norm ledger, and the provenance row.

Apply Definition 217.2 to keep vector and norm data inside the consumed Hilbert certificate. Apply Definition 217.3 to expose equality of pure states only through the projective/global-phase classifier, and apply Definition 217.4 to transport that classifier across certified presentations.

The public obligation surface of Definition 217.5 is exact. It exports no channel map, no density-matrix row, no measurement row, no spectral row, and no quotient or host-vector equality. Therefore $\mathsf{QuantumChannel}^{\uparrow}$ may attach its channel certificate to the exported classifier and provenance, but all channel action and output certification remains a downstream obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$**.**

Constructive story: A quantum-state row is built by consuming a $\mathsf{Hilbert}^{\uparrow}$ packet and a $\mathsf{ProjectiveSpace}^{\uparrow}$ packet, then displaying the Hilbert unit-vector endpoint, norm-one ledger, global-phase row, projective endpoint, inherited classifier rows, and shared provenance in one finite BHist carrier. The carrier accepts only packets whose vector and norm data are projected from the Hilbert source and whose projective endpoint and phase comparison are projected from the projective-space source; density matrices, observables, channel actions, spectral data, host vector equality, and quotient objects are outside this public row. Two accepted packets name the same quantum state when the unit-vector endpoints are related through the displayed global-phase/projective classifier while Hilbert and projective rows transport componentwise by hsame, not by equality of vectors in an ambient Hilbert space. The source specification is the Hilbert-plus-projective dependency surface, the pattern specification records norm-one projection, projective phase classification, global-phase

stability, provenance exactness, observable and density readback boundaries, and channel input boundaries, and the ledger keeps the Hilbert row, projective endpoint, phase row, norm row, and provenance visible. The five name-certificate obligations are read from the consumed classifiers: an admitted packet inhabits the carrier, the projective phase classifier is reflexive, symmetric, and transitive through its dependency rows, and classified Hilbert/projective transports repack as another admitted quantum-state packet. A packet h lies in the signature gap of a $\mathsf{QuantumState}^{\uparrow}$ token p when the source-domain accepts its Hilbert and projective rows, the signature relation pairs it with an answer state packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, the unit vector, norm-one row, phase comparison, projective endpoint, and provenance must be the entries encoded by p. The chapter registers this projective-unit-vector construction under the name $\mathsf{QuantumState}^{\uparrow}$; downstream observable, density, and channel consumers may read only the exported classifier and provenance, with no density-matrix carrier, measurement spectrum, channel row, host vector equality, or primitive quotient introduced here.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$, maturePackageClosed)

Scope closed: The carrier, classifier, stability, ledger, exactness, observable, density, and channel consumer-readback obligations are enumerated by Definition 217.5, Definition 217.16, Definition 217.19, and Definition 217.27.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier\_projective\_phase\_classifier`

Bridge status: none

Not claimed: The chapter does not close density-matrix rows, observable spectra, channel rows, host vector equality, or quotient primitives.

Upgrade path: Broader density-matrix, observable-spectrum, and channel packages require separate certificates beyond the exported $\mathsf{QuantumState}^{\uparrow}$ surface.

Theorem 217.28 (QuantumStateUp provenance row exactness)**.** *For every carrier packet*

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho)$$

of Definition 217.1, the provenance row ρ partitions the public rows of $\mathcal{Q}$ exactly into the consumed $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *packet $\mathcal{H}$, the consumed* $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$ *packet $\mathcal{P}$, the displayed norm row n, phase row ϕ, projective endpoint row p, Hilbert endpoint row v, and inherited classifier rows λ_H, λ_P. No observable row, density-matrix row, channel row, spectral row, host-vector-equality row, or quotient row belongs to the public* BHist *packet witnessed by ρ.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier provenance row exactness`

Proof. Use the carrier display in Definition 217.1 as the row inventory. The packet has the two consumed NameCert rows $\mathcal{H}$ and $\mathcal{P}$, the endpoint and ledger rows v, n, ϕ, p, the inherited classifiers λ_H, λ_P, and the provenance row ρ. The row ρ is therefore the Pkg record tying exactly those rows into one finite BHist packet.

The Hilbert portion is supplied by Definition 217.5: it exposes the carrier, classifier, stability, ledger, exactness, and hidden phase rows but no downstream observable object. The projective portion is supplied by Definition 217.3: it reads phase classification through ϕ and λ_P, not through a host quotient.

Finally apply the consumer boundary in Definition 217.5. A downstream row may inspect only the named Hilbert, projective, phase, norm, and provenance fields. Together with the global-phase stability theorem Definition 217.4, this proves that ρ is exact for the displayed NameCert package and excludes every observable, density-matrix, channel, spectral, host-vector-equality, and quotient row from the public surface. □

Theorem 217.29 (QuantumStateUp phase ledger exhaustion)**.** *Every hidden row used to classify two* $\mathsf{QuantumState}^{\uparrow}$ *carrier packets is one of the displayed phase row ϕ, inherited Hilbert classifier λ_H, inherited projective classifier λ_P, norm-one ledger n, endpoint* hsame *transport row, or provenance row ρ from the carrier of Definition 217.1. Conversely, those rows reconstruct exactly the projective phase classifier of Definition 217.3 inside the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{ProjectiveSpace}^{\uparrow}}$ *packets.*

Lean4 ✓ `BEDC.Derived.QuantumStateUp.QuantumStateBHistCarrier phase ledger exhaustion`

Proof. Start from two displayed packets

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho)$$

and

$$\mathcal{Q}' = (\mathcal{H}', \mathcal{P}', v', n', \phi', p', \lambda'_H, \lambda'_P, \rho').$$

The carrier definition lists the only rows available to the classifier. Thus a hidden classifier row must be read from the phase row, one of the inherited classifiers, the norm ledger, the endpoint hsame transport data, or the Pkg provenance record.

For exhaustion, use Definition 217.5 to identify the public $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ obligations and Definition 217.5 to rule out consumer access to a stronger equality. The stability row in Definition 217.4 accounts for every endpoint transport through hsame, while the provenance exactness of Definition 217.28 keeps those transports inside the same finite BHist package.

For reconstruction, read the Hilbert endpoint through λ_H, read the norm ledger through n, and read the projective endpoint comparison through ϕ and λ_P. These are precisely the inputs named by Definition 217.3; repacking them with ρ recovers the same projective phase classifier and adds no hidden row beyond the displayed ledger. □

Theorem 217.30 (QuantumStateUp consumer readback completeness)**.** *Every downstream* $\mathsf{Observable}^\uparrow$, $\mathsf{DensityMatrix}^\uparrow$, *and* $\mathsf{QuantumChannel}^\uparrow$ *consumer row reading* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *factors through the five public fields named in Definition 217.5: the Hilbert source row* $\mathcal{H}$*, projective endpoint row* p*, global-phase classifier* ϕ*, norm ledger* n*, and package provenance* ρ*. Conversely, any consumer depending only on those five fields is stable under Definition 217.4.*

Proof. The forward direction is a readback claim. By Definition 217.1, a consumer begins with a finite BHist packet carried by Pkg provenance. By Definition 217.5, the public $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ surface supplies the carrier, classifier, stability, ledger, exactness, and hidden phase obligations, but it supplies no observable, density-matrix, channel, or spectral row.

Apply Definition 217.5. The rows that remain readable by $\mathsf{Observable}^\uparrow$, $\mathsf{DensityMatrix}^\uparrow$, and $\mathsf{QuantumChannel}^\uparrow$ consumers are exactly $\mathcal{H}$, p, ϕ, n, and ρ. The projective classifier theorem Definition 217.3 explains how the endpoint and phase comparison behind those five fields is classified without exposing host vector equality or a quotient primitive.

For the converse direction, suppose a consumer depends only on $\mathcal{H}$, p, ϕ, n, and ρ. Transport the Hilbert and projective source data by the hsame rows governed by λ_H and λ_P. The global-phase stability theorem Definition 217.4 preserves the phase row and norm ledger, and Definition 217.28 preserves the finite package boundary. Therefore the consumer readback is unchanged under certified global-phase transport. □

Theorem 217.31 (QuantumStateUp phase-provenance scoped readback)**.** *Every* $\mathsf{Observable}^\uparrow$, $\mathsf{DensityMatrix}^\uparrow$, *or* $\mathsf{QuantumChannel}^\uparrow$ *consumer row reading* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *has a unique scoped* BHist *read surface. This surface consists exactly of the Hilbert source row* $\mathcal{H}$*, projective endpoint row* p*, global-phase classifier* ϕ*, norm ledger* n*, package provenance* ρ*, and the inherited Hilbert and projective classifier witnesses* λ_H, λ_P *from the carrier*

$$\mathcal{Q} = (\mathcal{H}, \mathcal{P}, v, n, \phi, p, \lambda_H, \lambda_P, \rho)$$

of Definition 217.1. Conversely, these rows reconstruct the same consumer-visible projective phase classifier and exclude observable, density-matrix, channel, spectral, host-vector-equality, and quotient rows from the scoped readback surface.

Proof. Begin with the provenance boundary. By Definition 217.28, the BEDC source BHist packet carried by ρ is exactly the finite package surface of $\mathcal{Q}$: the carrier is $\mathsf{QuantumStateBHistCarrier}$, the classifier and projective phase row are ϕ together with λ_P, stability is read through hsame, and the ledger is packaged by Pkg.

Next apply Definition 217.29. It identifies all phase and hidden-classifier rows available to a consumer with the displayed rows $\phi, n, \lambda_H, \lambda_P$ and the endpoint transport governed by hsame. Thus the Hilbert and projective classifier witnesses are not extra public objects; they are the inherited witnesses needed to read the projective phase classifier.

Finally use Definition 217.30. For each downstream consumer, the per-object template is the same: the source is the BHist packet above, the carrier is $\mathsf{QuantumStateBHistCarrier}$, the classifier is the projective phase row, stability is by hsame, the ledger is by Pkg provenance, and the downstream readers are $\mathsf{Observable}^{\uparrow}$, $\mathsf{DensityMatrix}^{\uparrow}$, and $\mathsf{QuantumChannel}^{\uparrow}$. These three consumers therefore see exactly the same projective phase classifier and no observable, density-matrix, channel, spectral, host-vector-equality, or quotient row outside the scoped surface. □

Chapter 218

A Concrete Naming Certificate for $\mathsf{Observable}^{\uparrow}$

The observable interface $\mathsf{Observable}^{\uparrow}$ packages a self-adjoint operator on a Hilbert space carrier together with its spectrum and expectation pairing with quantum states. It depends on $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91), $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ (chapter 214), and $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ (chapter 217).

218.1 Carrier definition

Definition 218.1 (ObservableUp BHist operator carrier)**.** *An* $\mathsf{Observable}^{\uparrow}$ *packet is a finite displayed* BHist *row package*

$$\mathcal{O} = (H, A, S, E, W, \rho, \lambda).$$

The row H is the consumed $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *carrier row. The endpoint $A : \mathsf{BHist}$ is the visible operator endpoint over H. The row W records the self-adjointness witness for that endpoint. The row S is the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *spectrum row attached to (H, A, W). The row E records the expectation-source endpoints read through* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *and paired with the observable endpoint A. The row ρ is the shared finite* Pkg *provenance tying H, A, S, E, and W into one certificate packet, and λ is the finite* $\mathsf{Cont/hsame}$ *ledger used by every transported row.*

The packet is admitted as an $\mathsf{Observable}^{\uparrow}$ *carrier exactly when all Hilbert space structure is projected from H, all spectral data is projected from S, all expectation sources are projected from E, and the only observable endpoint outside the dependency rows is the displayed* BHist *row A with its witness row W. No host operator equality, ambient Hilbert-space equality, or external spectral primitive is part of the carrier.*

Lean4 (definition) `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier`

Definition 218.2 (ObservableUp spectral expectation classifier)**.** *Let*

$$\mathcal{O} = (H, A, S, E, W, \rho, \lambda) \quad \textit{and} \quad \mathcal{O}' = (H', A', S', E', W', \rho', \lambda')$$

be admitted $\mathsf{Observable}^{\uparrow}$ *packets. The* $\mathsf{Observable}^{\uparrow}$ *classifier holds exactly when* hsame *transports the Hilbert carrier row, the operator endpoint row, the self-adjointness witness row, the spectral row, the expectation-source endpoints, and the shared package ledger:*

$$H\mathsf{hsame}H' \wedge A\mathsf{hsame}A' \wedge W\mathsf{hsame}W' \wedge S\mathsf{hsame}S' \wedge E\mathsf{hsame}E' \wedge \rho\mathsf{hsame}\rho' \wedge \lambda\mathsf{hsame}\lambda'.$$

The H comparison is read through $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$*, the S comparison is read through* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$*, and the expectation endpoint comparison is read through* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$*. The classifier therefore identifies only displayed rows and their common* Pkg *provenance. It does not compare host operators by equality.*

Lean4 (definition) `BEDC.Derived.ObservableUp.ObservableSpectralExpectationClassifier`

218.2 The certificate

Theorem 218.3 (ObservableUp Hilbert source boundary)**.** *For every admitted* $\mathsf{Observable}^\uparrow$ *packet of Definition 218.1, every operator row, spectral row, and expectation row reads its Hilbert source only through the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ *row H and* hsame *transport. In particular, the operator endpoint A, the spectral-data row S, and the expectation-source row E cannot inspect an ambient Hilbert carrier apart from the certified source row and its displayed classifier transport.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier hilbert source readback`
Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier hilbert source boundary`

Proof. Unpack the packet $\mathcal{O} = (H, A, S, E, W, \rho, \lambda)$. The only field that names a Hilbert source is H, and the only transport field available to move Hilbert-side data is the Cont ledger λ recording hsame-compatible rows.

Project the operator endpoint A and its self-adjointness witness row W from the packet. Their source is required to be the same H because the endpoint is displayed over the Hilbert source field rather than over a separate carrier. Project the spectral row S next; by construction it is the spectral-data row for that same displayed endpoint and witness.

Finally project the expectation row E. Its state input is constrained by $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$, while its vector-space and inner-product source is still the consumed Hilbert row H. Repacking these projections with ρ gives the stated source boundary: all Hilbert reads factor through H and the displayed hsame transport rows. □

Theorem 218.4 (ObservableUp spectral data ledger)**.** *The spectral rows consumed by* $\mathsf{Observable}^\uparrow$ *are exactly the* $\mathsf{SpectralTheorem}^\uparrow$ *spectrum, projection-measure, and functional-calculus rows for the displayed self-adjoint operator endpoint (A, W), all with the same package provenance row ρ. No spectral support, projection assignment, or calculus action is accepted by the observable packet unless it is one of the rows carried by the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$ *entry S.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier spectral data ledger`

Proof. Read the spectral-data field S of Definition 218.1. It is not an independent host spectral theorem; it is the consumed $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$ row for the operator endpoint field A and witness field W.

The forward projection exposes precisely the spectral entries supplied by that certificate: the spectrum row, the projection-measure row, and the functional-calculus row. The provenance row ρ attaches these entries to the same finite packet as H, A, and W.

For the reverse reading, any displayed spectrum, projection-measure, and functional-calculus rows with the shared ρ provenance repack as the spectral field S only when they are accepted by $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$. Thus the ledger is exact for those spectral rows and contains no further spectral datum. □

Theorem 218.5 (ObservableUp quantum-state expectation boundary)**.** *Expectation evaluation in an* $\mathsf{Observable}^{\uparrow}$ *packet may consume only* $\mathsf{QuantumState}^{\uparrow}$*-compatible input rows over the same Hilbert source* H*. If the Hilbert source, operator endpoint, witness row, spectral row, and quantum-state expectation row are transported by the displayed classifiers in* λ*, then the expectation readback transports to the corresponding expectation row over the transported packet.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier expectation transport readback`

Proof. Project the expectation-source row E from the packet. By definition E is read through $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$, so every state input used by the expectation pairing is a certified quantum-state row rather than an arbitrary vector row.

Use Definition 218.3 to identify the Hilbert source of the operator endpoint, witness row, spectral row, and expectation row with the same consumed row H. The quantum-state boundary then supplies only state inputs whose Hilbert field is compatible with that source.

Transport a classified packet through λ. The hsame rows move H, A, W, S, and the expectation-source componentwise. Since the expectation readback is displayed as the pairing of the transported operator endpoint with the transported state input, it repacks as the transported expectation row and remains inside the $\mathsf{Observable}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier quantumstate expectation boundary`

Theorem 218.6 (ObservableUp Hilbert and spectral boundary)**.** *For every admitted* $\mathsf{Observable}^{\uparrow}$ *packet of Definition 218.1, carrier acceptance exposes exactly the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *row* H*, the operator endpoint* A*, the self-adjointness witness row* W*, the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *row* S*, the expectation-source row* E*, the shared* Pkg *provenance row* ρ*, and the finite* Cont *ledger* λ*. The exposed boundary is stable only by the displayed* hsame *transport rows in* λ*.*

Proof. Unpack the carrier packet of Definition 218.1. The definition lists exactly the consumed Hilbert certificate, the operator endpoint, its witness row, the consumed spectral certificate, the expectation-source row, the provenance row, and the continuation ledger.

Acceptance is therefore a row-readback property. A consumer may inspect H, A, W, S, E, ρ, and λ, and every stability witness must be one of the hsame rows named by λ.

There is no additional carrier whose fields could expose an ambient Hilbert space, an ambient spectral theorem, a host operator equality, or an external state primitive. The boundary is precisely the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier hilbert spectral boundary`

Theorem 218.7 (ObservableUp expectation ledger exactness)**.** *For every admitted* $\mathsf{Observable}^{\uparrow}$ *packet of Definition 218.1, each public expectation row of* $\mathsf{Observable}^{\uparrow}$ *is read from the expectation-source endpoint row* E*, the operator endpoint* A*, the self-adjointness witness row* W*, the spectral row* S*, the shared* Pkg *provenance row* ρ*, and the finite* Cont *ledger* λ*. The readback uses the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *packet* H*, the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *packet* S*, and the consumed* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *rows inside* E *only through their displayed rows and their* hsame *transport entries in* λ*, not through host operator equality.*

Proof. The expectation interface is the row E. Its observable source is the endpoint A with witness row W, its spectral support is the consumed row S, and its package membership is recorded by ρ.

Exactness follows by restricting public expectation readback to those fields. The ledger λ supplies the only permitted hsame transport data for moving the Hilbert source, the operator row, the witness row, the spectral row, and the expectation row across admitted presentations.

Thus equality of expectation rows is not recovered from an external operator identity. It is the certificate-local relation determined by E, A, W, S, ρ, λ, and the displayed rows of the consumed dependency certificates. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier expectation ledger exactness`

Theorem 218.8 (ObservableUp spectral transport stability)**.** *Let $\mathcal{O} = (H, A, S, E, W, \rho, \lambda)$ and $\mathcal{O}' = (H', A', S', E', W', \rho', \lambda')$ be admitted* $\mathsf{Observable}^{\uparrow}$ *packets of Definition 218.1. If the two packets are related by the classifier of Definition 218.2, then the transported packet is an admitted* $\mathsf{Observable}^{\uparrow}$ *packet. The displayed expectation boundary of Definition 218.7 is preserved with the transported* Pkg *provenance and finite* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier spectral transport stability`

Proof. Transport the Hilbert dependency first. The classifier moves the Hilbert carrier row from H to H', and the relevant hsame witness is recorded in the continuation ledger.

Transport the observable endpoint and witness next. The rows A and W move to A' and W', so the spectral row S transports to S' over the transported endpoint and witness, again using only the hsame entries named by the ledger.

The expectation-source row is then transported as a dependent endpoint over the already transported Hilbert, operator, witness, and spectral rows. The provenance row is carried from ρ to ρ', and the ledger changes from λ to λ'.

All fields of the target tuple are now present, so the target tuple is admitted by Definition 218.1. The expectation boundary is unchanged in form because the public rows are still exactly E', A', W', S', ρ', and λ' over H'. □

Theorem 218.9 (Observable Hilbert-spectral ledger exactness)**.** *For every admitted* $\mathsf{Observable}^{\uparrow}$ *packet of Definition 218.1, the public* $\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$ *ledger exposes exactly the Hilbert carrier row H, the operator endpoint A, the self-adjointness witness row W, the* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *spectrum row S, the* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *expectation-source row E, the shared* Cont/hsame *transport ledger λ, and the shared* Pkg *provenance ρ. No other row is exported by the certificate.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier hilbert spectral ledger exactness`

Proof. Unpack the carried tuple $(H, A, S, E, W, \rho, \lambda)$. The first projection is the consumed $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ row, so all inner-product, vector, and scalar structure available to the observable is read through H.

Project the spectral and state rows. The spectrum row S is consumed through $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ for the displayed endpoint and witness (A, W), while the expectation row E supplies the quantum-state endpoints through $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$.

The remaining projections are the visible operator endpoint A, the self-adjointness witness row W, the package provenance ρ, and the finite transport ledger λ. Repacking these projections gives exactly the displayed public ledger. Since the carrier contains no host operator equality and no external spectral primitive, no additional ledger row can be read from the certificate. □

Theorem 218.10 (Observable classifier stability)**.** *If two admitted* $\mathsf{Observable}^{\uparrow}$ *packets are related by the classifier of Definition 218.2, then* hsame *transport preserves the carried Hilbert row, operator*

endpoint row, self-adjointness row, spectral row, expectation endpoint row, shared Pkg *provenance, and finite* Cont *ledger.*

Proof. Start from the classifier relation. It supplies the displayed comparisons $H\mathsf{hsame}H'$, $A\mathsf{hsame}A'$, $W\mathsf{hsame}W'$, $S\mathsf{hsame}S'$, $E\mathsf{hsame}E'$, $\rho\mathsf{hsame}\rho'$, and $\lambda\mathsf{hsame}\lambda'$.

Transport the dependency rows first. The Hilbert comparison moves all carrier reads through $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$, and the spectral comparison moves the spectrum read through $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ with the transported endpoint and witness.

Transport the expectation row through $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ and keep the package boundary fixed by $\rho\mathsf{hsame}\rho'$. The ledger comparison $\lambda\mathsf{hsame}\lambda'$ then identifies every finite Cont row used by the source packet with the corresponding row in the target packet. Thus the target packet has the same visible carrier and spectral-expectation rows, up to the displayed classifier. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier classifier stability`

Theorem 218.11 (Observable expectation pairing boundary)**.** *Every expectation value exported by an admitted* $\mathsf{Observable}^{\uparrow}$ *packet is read through the carried* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *expectation-source row E and the observable endpoint ledger (A, W, ρ, λ). It is not justified by host-level inner-product equality, host operator equality, or an external spectral primitive.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier expectation pairing boundary`

Proof. Take an exported expectation value. By ledger exactness it must appear in the expectation-source row E of the carried packet, because no other row in the packet is allowed to export expectation endpoints.

The observable side of the pairing is the displayed endpoint A together with its self-adjointness witness row W. The spectrum row S may be used only as the consumed $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ dependency row attached to those displayed observable data.

Finally attach the provenance and transport boundary. The Pkg row ρ places the state endpoint and observable endpoint in the same finite packet, and the Cont/hsame ledger λ supplies the only admissible transport of the readback. Therefore the expectation pairing is a packet-level readback, not a claim about equality of host inner products or host operators. □

Theorem 218.12 (ObservableUp public row family)**.** *The public* $\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$ *family consists exactly of the finite* BHist *carrier packet of Definition 218.1, the classifier of Definition 218.2, the Hilbert source boundary, the spectral data ledger, the quantum-state expectation boundary, the Hilbert-spectral boundary, the expectation ledger exactness rows, the spectral transport rows, the classifier stability rows, and the expectation pairing boundary. Its public fields are the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *packet, the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *packet, the consumed* $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ *expectation-source row, the observable endpoint, the self-adjointness witness row, the* Pkg *provenance row, the finite* Cont *ledger, and the displayed* hsame *transport family.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier public row family`

Proof. Start with the carrier packet. It supplies the Hilbert dependency, observable endpoint, spectral dependency, expectation-source dependency, self-adjointness witness, package provenance, and continuation ledger.

The classifier identifies the admitted packet coordinates by hsame. The boundary theorems identify the rows that carrier acceptance exposes. The exactness theorems then state that public spectral and expectation readback is confined to the displayed dependency, observable, provenance, and ledger rows.

The transport and classifier stability theorems supply stability under the displayed classifiers of the dependency packets. Combining these rows gives the carrier, classifier, transport, ledger, and exactness pieces of the public $\mathsf{NameCert}_{\mathsf{Observable}^\uparrow}$ obligation family. No host-level observable, operator-equality primitive, independent state primitive, or extra spectral primitive is part of this surface. □

Theorem 218.13 (Observable NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Observable}^\uparrow}$ *obligation surface consists exactly of the carrier of Definition 218.1, the classifier of Definition 218.2, Hilbert source boundary, spectral data ledger, quantum-state expectation boundary, Hilbert-spectral ledger exactness, classifier stability, expectation-pairing boundary, and the shared* Cont/Pkg *ledger clauses for the displayed rows* $H, A, S, E, W, \rho, \lambda$*. It consumes the existing* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$*,* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *dependency surfaces and consumes no host operator equality or independent spectral primitive.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableNameCert obligation surface`
Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier semanticNameCert`
Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier namecert obligation surface`

Proof. Collect the carrier rows first. The finite tuple $(H, A, S, E, W, \rho, \lambda)$ displays the Hilbert dependency, observable endpoint, spectral dependency, expectation-source dependency, self-adjointness witness, package provenance, and transport ledger.

Collect the classifier rows next. The spectral expectation classifier compares exactly those displayed coordinates by hsame, with dependency reads delegated to the Hilbert, spectral-theorem, and quantum-state certificates named by the chapter dependencies.

Use Definition 218.3 for the Hilbert source boundary, Definition 218.4 for spectral ledger exactness, Definition 218.5 for the state-input boundary, Definition 218.9 for public ledger coverage, Definition 218.10 for transported stability, and Definition 218.11 for the expectation readback boundary. These clauses cover every public request admitted by the carrier and classifier. Since none of the collected rows contains host operator equality or an external spectral primitive, the stated clauses form the full paper-side $\mathsf{Observable}^\uparrow$ obligation surface. □

Definition 218.14 (ObservableUp expectation interface packet)**.** *The downstream expectation interface exported by* $\mathsf{Observable}^\uparrow$ *is the finite* BHist *packet*

$$\mathcal{I}_{\mathsf{Observable}^\uparrow} = (H, A, W, S, E, \rho, \lambda).$$

Here H *is the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ *carrier row,* A *is the displayed observable endpoint,* W *is its self-adjointness witness row,* S *is the consumed* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$ *spectral row for* (H, A, W)*, and* E *is the consumed* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *expectation-source row over the same Hilbert source. The row* ρ *is the shared finite* Pkg *provenance for these entries, and* λ *is the shared finite* Cont/hsame *transport ledger.*

The packet is exact as a consumer boundary: a downstream quantum-facing chapter may read Hilbert structure, spectral structure, state endpoints, expectation readback, provenance, and transport only through the seven rows displayed above. It contains no host operator equality and no ambient spectral or state primitive.

Lean4 (definition) `BEDC.Derived.ObservableUp.ObservableExpectationInterfacePacket`

Theorem 218.15 (ObservableUp spectral-state source coherence)**.** *For every admitted expectation interface packet* $\mathcal{I}_{\mathsf{Observable}^\uparrow} = (H, A, W, S, E, \rho, \lambda)$ *of Definition 218.14, the spectral row* S *and the*

expectation row E share the same Hilbert source H, the same operator endpoint A, the same self-adjointness witness W, the same package provenance ρ, and the same transport ledger λ. Their coherence is a displayed BHist *coherence: it is witnessed by projection from the carrier packet and by* hsame *rows in λ, not by host equality of Hilbert spaces, operators, spectra, or states.*

Proof. Project the expectation interface packet to the carrier tuple of Definition 218.1. The rows are the same rows, only ordered for consumer readback as $(H, A, W, S, E, \rho, \lambda)$.

Use Definition 218.3 to identify H as the sole Hilbert source available to the operator endpoint, witness row, spectral row, and expectation row. Use Definition 218.4 to read S as the consumed spectral certificate for the displayed endpoint and witness (A, W).

Use Definition 218.5 to read E through certified quantum-state endpoints over the same source H. The rows ρ and λ are then the shared package and transport rows supplied by the packet itself, so repacking these projections gives the stated spectral-state coherence without adding any host equality. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier spectral state source coherence`

Theorem 218.16 (ObservableUp expectation transport exactness)**.** *Let $\mathcal{I}_{\mathsf{Observable}^\uparrow} = (H, A, W, S, E, \rho, \lambda)$ and $\mathcal{I}'_{\mathsf{Observable}^\uparrow} = (H', A', W', S', E', \rho', \lambda')$ be admitted expectation interface packets. If the two packets are related by the* $\mathsf{Observable}^\uparrow$ *classifier, then classifier transport preserves the whole expectation interface packet:*

$$H\mathsf{hsame}H', \quad A\mathsf{hsame}A', \quad W\mathsf{hsame}W', \quad S\mathsf{hsame}S', \quad E\mathsf{hsame}E', \quad \rho\mathsf{hsame}\rho', \quad \lambda\mathsf{hsame}\lambda'.$$

The transported packet is again an admitted downstream interface, and this transport produces no host operator equality, ambient Hilbert equality, or external equality of spectral or state primitives.

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier expectation transport exactness`

Proof. Start from the classifier of Definition 218.2. It supplies exactly the displayed hsame comparisons for H, A, W, S, E, ρ, and λ.

Apply Definition 218.10 to transport the carried Hilbert row, endpoint row, witness row, spectral row, expectation endpoint row, provenance row, and continuation ledger. Apply Definition 218.8 to repack the target rows as an admitted $\mathsf{Observable}^\uparrow$ carrier.

Finally read the target carrier through Definition 218.14. The packet components are exactly $(H', A', W', S', E', \rho', \lambda')$, so the consumer interface is preserved componentwise. Since the classifier relation contains only displayed hsame rows, no host operator equality or ambient equality is created by the transport. □

Theorem 218.17 (ObservableUp public consumer exhaustion)**.** *Every downstream consumer of the public* $\mathsf{Observable}^\uparrow$ *interface reads only the finite expectation interface packet of Definition 218.14 together with the public* $\mathsf{NameCert}_{\mathsf{Observable}^\uparrow}$ *obligation surface of Definition 218.13. Thus every permitted consumer request factors through the rows $H, A, W, S, E, \rho, \lambda$, the displayed* Cont/hsame *transport ledger, and the consumed* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{QuantumState}^\uparrow}$ *surfaces. No downstream request can require a row outside that finite* BHist *packet or a host operator equality.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier public consumer exhaustion`

Proof. Begin with an arbitrary public consumer request. The obligation surface of Definition 218.13 says that public requests are exhausted by the carrier, classifier, Hilbert source boundary, spectral data ledger, quantum-state expectation boundary, Hilbert-spectral ledger exactness, classifier stability, expectation-pairing boundary, and the shared Cont/Pkg ledger clauses.

The carrier and classifier rows are exactly the rows repacked by Definition 218.14. The Hilbert source boundary, spectral data ledger, and expectation boundary reduce the dependency reads to H, S, and E with endpoint and witness rows A and W.

The remaining public clauses are ledger clauses. Use Definition 218.9 for finite public row coverage, Definition 218.10 for transported stability, and Definition 218.11 for expectation readback. These rows are all contained in $(H, A, W, S, E, \rho, \lambda)$ and the named dependency certificates, so the consumer has no additional public source to inspect. □

Theorem 218.18 (ObservableUp dependency NameCert exactness)**.** *For every admitted* $\mathsf{Observable}^{\uparrow}$ *packet* $\mathcal{O} = (H, A, S, E, W, \rho, \lambda)$ *of Definition 218.1, every* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{SpectralTheorem}^{\uparrow}$, *and* $\mathsf{QuantumState}^{\uparrow}$ *dependency read performed by the packet factors through the displayed rows* H, S, *and* E *together with the finite* hsame/Cont *ledger* λ. *No dependency read is justified by an ambient Hilbert carrier, a host operator equality, an external spectral theorem, or an independent state endpoint outside those rows.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier dependency namecert exactness`

Proof. Project the admitted packet as $(H, A, S, E, W, \rho, \lambda)$. The carrier definition names H as the only Hilbert source, S as the only spectral dependency row, and E as the only quantum-state expectation source.

Apply Definition 218.3 to the Hilbert reads, Definition 218.4 to the spectral reads, and Definition 218.5 to the state-input and expectation reads.

The remaining admissible movement is the finite ledger λ. Its hsame rows are the only transport witnesses exposed by Definition 218.9, so every dependency read factors through H, S, E, and λ. □

Theorem 218.19 (ObservableUp operator-row classifier determinacy)**.** *Let* $\mathcal{O} = (H, A, S, E, W, \rho, \lambda)$ *and* $\mathcal{O}' = (H', A', S', E', W', \rho', \lambda')$ *be admitted* $\mathsf{Observable}^{\uparrow}$ *packets related by the classifier of Definition 218.2. Then the two packets determine the same public operator endpoint row, self-adjointness witness row, spectral row, expectation row, package provenance, and transport ledger up to the displayed* hsame *comparisons. The determinacy is a row-level certificate claim and does not assert equality of host operators.*

Proof. Unpack the classifier. It supplies exactly the comparisons $H\mathsf{hsame}H'$, $A\mathsf{hsame}A'$, $W\mathsf{hsame}W'$, $S\mathsf{hsame}S'$, $E\mathsf{hsame}E'$, $\rho\mathsf{hsame}\rho'$, and $\lambda\mathsf{hsame}\lambda'$.

Repack the public rows through these comparisons. The endpoint and witness are determined by $A\mathsf{hsame}A'$ and $W\mathsf{hsame}W'$, the spectral row by $S\mathsf{hsame}S'$, the expectation row by $E\mathsf{hsame}E'$, and the provenance and ledger by $\rho\mathsf{hsame}\rho'$ and $\lambda\mathsf{hsame}\lambda'$.

Use Definition 218.10 to identify the transported public surface. Since the classifier contains no host operator equality field, the resulting determinacy is precisely the displayed row determinacy. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier operator row classifier determinacy`

Theorem 218.20 (ObservableUp spectral row consumer coverage)**.** *Every downstream consumer of the public* $\mathsf{Observable}^{\uparrow}$ *surface can read only the consumed Hilbert row* H, *the operator endpoint*

A, the self-adjointness witness row W, the consumed $\mathsf{SpectralTheorem}^{\uparrow}$ *row S, the* $\mathsf{QuantumState}^{\uparrow}$ *expectation row E, the* Pkg *provenance row ρ, and the finite* Cont/hsame *ledger λ. In particular, spectral consumers receive the displayed spectral row and its certified dependencies, not an extra host-level spectral object.*

Proof. Start from Definition 218.9. It lists exactly the Hilbert, operator, witness, spectral, expectation, provenance, and ledger rows exported by an admitted packet.

Then apply Definition 218.12. The public family is assembled from the carrier, classifier, boundary, transport, and ledger clauses already named in the chapter, so a downstream consumer has no additional row source to inspect.

Thus a spectral consumer reads S together with H, A, W, E, ρ, and λ only through the public certificate family. The coverage claim is therefore exact for the displayed $\mathsf{Observable}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier spectral row consumer coverage`

Theorem 218.21 (ObservableUp expectation row transport composition)**.** *For admitted packets $\mathcal{O} = (H, A, S, E, W, \rho, \lambda)$ and $\mathcal{O}' = (H', A', S', E', W', \rho', \lambda')$ related by the classifier of Definition 218.2, classifier transport followed by expectation readback agrees with expectation readback over the transported packet. The composite uses only the rows E, A, W, S, ρ, and λ together with their displayed transported counterparts.*

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier expectation row transport composition`

Proof. Apply Definition 218.5 to identify the expectation readback as a certified state-input row paired with the observable endpoint and witness over the common Hilbert source.

Apply Definition 218.8 to transport the packet. The classifier moves A, W, S, E, ρ, and λ to their displayed target rows, and the transported tuple is again admitted as an $\mathsf{Observable}^{\uparrow}$ packet.

Finally use Definition 218.11. It confines the readback to E, A, W, S, ρ, and λ, so reading before transport and then transporting gives the same displayed row as reading from the transported packet. □

Theorem 218.22 (ObservableUp complete paper obligation surface)**.** *The complete paper-side* $\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$ *obligation inventory is formed by the carrier of Definition 218.1, the classifier of Definition 218.2, the ledger exactness rows, the public row family, the obligation surface of Definition 218.13, dependency* NameCert *exactness, operator-row classifier determinacy, spectral consumer coverage, and expectation transport composition. This inventory supplies the downstream read boundary for* $\mathsf{QuantumState}^{\uparrow}$ *and* $\mathsf{SpectralTheorem}^{\uparrow}$ *consumers of* $\mathsf{Observable}^{\uparrow}$.

Proof. Collect the existing carrier and classifier definitions. They give the finite packet $(H, A, S, E, W, \rho, \lambda)$ and the displayed hsame comparisons that control admitted presentations.

Collect the existing boundary rows through Definition 218.9, Definition 218.12, and Definition 218.13. These rows specify the public carrier, dependency, transport, provenance, and readback clauses.

The four preceding theorems make the dependency exactness, classifier determinacy, consumer coverage, and transport-composition obligations explicit. Together they cover every public dependency and readback used by downstream $\mathsf{QuantumState}^{\uparrow}$ and $\mathsf{SpectralTheorem}^{\uparrow}$ consumers, while excluding host operator equality and external spectral primitives from the paper surface. □

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier namecert obligation surface complete`

Theorem 218.23 (ObservableUp public surface bridge boundary)**.** *The complete paper-side* $\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$ *inventory of Definition 218.22 determines the bridge boundary to the standard observable schema by reading the Hilbert source row, operator endpoint, self-adjointness witness, spectral row, expectation input rows,* hsame *transports,* Cont *routes, and* Pkg *provenance from the concrete* BHist *packet* $(H, A, S, E, W, \rho, \lambda)$*.*

This boundary records exactly the standard-observable input faces exposed by the public surface. It does not claim a checked standard bridge, host operator equality, ambient Hilbert equality, or an external spectral primitive.

Lean4 ✓ `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier public surface bridge boundary`

Proof. Project the complete obligation surface of Definition 218.22. The projection exposes the carrier packet, classifier, public row family, dependency rows, transport rows, provenance row, and readback rows of the concrete BHist presentation.

Use Definition 218.3, Definition 218.4, Definition 218.5, and Definition 218.9. These rows make the Hilbert source, spectral dependency, expectation pairing, and dependency exactness ledger explicit.

Finally use Definition 218.11, Definition 218.20, and Definition 218.21. They exhaust the public consumer faces and show that transport and readback remain inside the displayed rows $H, A, S, E, W, \rho, \lambda$.

The resulting boundary is therefore the standard-observable input surface and nothing beyond it. Since each cited row is stated without host operator equality, ambient Hilbert equality, or an external spectral primitive, the boundary also preserves those exclusions and leaves the checked bridge as a separate obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$**.**

Constructive story: An observable is displayed as the finite packet $(H, A, S, E, W, \rho, \lambda)$: the Hilbert source H is consumed from Hilbert$^{\uparrow}$, the operator endpoint A and self-adjointness witness W are visible histories over that source, the spectral row S is consumed from SpectralTheorem$^{\uparrow}$, and the expectation inputs E are read through QuantumState$^{\uparrow}$. The carrier predicate encoded here admits only packets whose Hilbert, spectral, expectation, self-adjointness, provenance, and transport rows are all displayed, and the classifier identifies two packets by transporting exactly those rows through hsame and through the three dependency certificates rather than by comparing host operators. The source therefore covers this Hilbert-operator packet, the pattern is self-adjoint operator plus spectrum, projection-measure, functional-calculus, and state-expectation readback, and the ledger keeps $H, A, S, E, W, \rho, \lambda$ as the exact public inventory. Within this obligation surface, an admitted packet provides the carrier witness, componentwise history-sameness and dependency classifiers give reflexivity, symmetry, and transitivity, and the carrier remains stable when each visible row is replaced by a classified row with the same provenance. A packet h lies in the signature gap of an Observable$^{\uparrow}$ token p when the source-domain accepts the Hilbert row, operator endpoint, self-adjointness witness, spectral data, expectation rows, and finite ledger, the signature relation pairs that packet with an answer packet and evidence ledger, and token introduction names the answer as p; unfolded, p records the same displayed operator together with its certified spectral and expectation rows. No standard bridge or full observable certificate is exported by this chapter. The chapter registers this encoded operator packet under the name Observable$^{\uparrow}$; an observable is a HilbertUp-sourced BHist operator with spectral and state-expectation ledgers, with no host operator equality, ambient Hilbert equality, or external spectral primitive introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper-side treatment is the displayed spectral-data ledger, expectation transport ledger, public consumer exhaustion row, dependency exactness row, and their Pkg, Cont, and hsame readback boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Observable}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.ObservableUp.ObservableBHistOperatorCarrier`

Bridge status: none

Not claimed: This chapter does not claim a Lean standard bridge or host-operator equality beyond the exported observable packet.

Upgrade path: The remaining upgrade path is Lean-side bridge verification for the already enumerated public surface.

Chapter 219

A Concrete Naming Certificate for $\mathsf{UnitaryGroup}^{\uparrow}$

The unitary-group interface $\mathsf{UnitaryGroup}^{\uparrow}$ packages the Lie group of inner-product-preserving automorphisms of a Hilbert space carrier. It depends on $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91) and $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140).

219.1 Carrier definition

Theorem 219.1 (UnitaryGroup source classifier obligation)**.** *The source carrier for* $\mathsf{UnitaryGroup}^{\uparrow}$ *is a* $\mathsf{Hilbert}^{\uparrow}$ *endpoint equipped with a* $\mathsf{LieGroup}^{\uparrow}$ *operation row and a displayed inner-product-preservation witness. Its classifier is the combined row obtained from the* $\mathsf{Hilbert}^{\uparrow}$ *endpoint classifier, the* $\mathsf{LieGroup}^{\uparrow}$ *operation classifier, and transport of the preservation row along the accepted histories.*

Proof. Unpack the two dependency certificates named in the introduction. The $\mathsf{Hilbert}^{\uparrow}$ source supplies the carried endpoint row and the Hilbert classifier, while the $\mathsf{LieGroup}^{\uparrow}$ source supplies the multiplication, inverse, and unit operation rows over the same visible history surface.

The unitary carrier is then the BHist package that stores those two source projections together with a preservation witness for the Hilbert inner product. A comparison is accepted only when the Hilbert endpoint classifier, the Lie-group operation classifier, and the preservation witness are all transported through the same hsame evidence.

Repacking these rows through Cont and $\mathsf{NameCert}$ gives the required classifier surface. No host equality of operators is consumed; the accepted comparison is exactly the displayed BEDC row transport. □

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupSourceClassifier obligation`

Theorem 219.2 (UnitaryGroup operation preservation obligation)**.** *The multiplication, inverse, and unit rows of* $\mathsf{UnitaryGroup}^{\uparrow}$ *are inherited from the* $\mathsf{LieGroup}^{\uparrow}$ *source certificate, and each inherited row preserves the* $\mathsf{Hilbert}^{\uparrow}$ *inner-product classifier on every carried endpoint. In particular, a product, inverse, or unit endpoint is admitted by the unitary carrier only when the corresponding* $\mathsf{LieGroup}^{\uparrow}$ *operation row repacks with the displayed Hilbert inner-product-preservation witness.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupOperation preservation obligation`

Proof. Project the operation rows from $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$. The product row supplies a carried composite endpoint, the inverse row supplies a carried inverse endpoint, and the unit row supplies the identity endpoint of the same source group package.

Next project the Hilbert classifier and inner-product rows from $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$. For each Lie-group operation, transport the input endpoint histories through hsame, apply the operation row through Cont, and carry the inner-product-preservation witness along the resulting endpoint comparison.

The resulting endpoint is repacked as a $\mathsf{UnitaryGroup}^\uparrow$ row precisely because it contains both pieces: the inherited Lie-group operation evidence and the Hilbert preservation witness. Thus operation closure does not extend the source group structure; it restricts it to the inner-product-preserving subcarrier. □

Theorem 219.3 (UnitaryGroup stability ledger exactness obligation)**.** *The* $\mathsf{UnitaryGroup}^\uparrow$ *certificate has coverage, soundness, and completeness for exactly the Hilbert carrier row, the* $\mathsf{LieGroup}^\uparrow$ *operation rows, the inner-product preservation rows, the classifier transport rows, and the dependency ledger naming* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$*. No additional operator, spectral, measure, or channel row is part of this ledger.*

Proof. Coverage reads any carried unitary endpoint back into its visible components: the $\mathsf{Hilbert}^\uparrow$ carrier row, the $\mathsf{LieGroup}^\uparrow$ operation rows, the preservation witness, the classifier transport evidence, and the two source NameCert dependencies.

Soundness is the forward projection. Given those ledger entries, the BHist carrier repacks the endpoint, hsame supplies the classifier transport, and Cont supplies the operation-composition rows needed by multiplication, inverse, and unit.

Completeness is the reverse reading. Any visible row accepted by the unitary carrier must be one of the listed Hilbert, Lie-group, preservation, transport, or dependency rows, because the certificate exposes no other ledger field. Hence the exactness boundary is precisely the displayed ledger and not a larger analytic operator surface. □

219.2 The certificate

Theorem 219.4 (UnitaryGroup carrier and classifier obligation)**.** *The carrier for* $\mathsf{UnitaryGroup}^\uparrow$ *is the* $\mathsf{Hilbert}^\uparrow$ *carrier together with the* $\mathsf{LieGroup}^\uparrow$ *automorphism rows whose source and target Hilbert carriers are classifier-compatible. Its classifier is inherited componentwise from the* $\mathsf{Hilbert}^\uparrow$ *carrier classifier and the* $\mathsf{LieGroup}^\uparrow$ *automorphism classifier, restricted to histories that preserve the displayed Hilbert source and target rows.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupCarrier classifier obligation`
↪ variant `BEDC.Derived.UnitaryGroupUp.UnitaryGroupCarrierClassifier obligation`

Proof. Unpack the carrier, classifier, source, and dependency fields of the ten-field roadmap template. The introduction supplies exactly the two certificate dependencies: $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ for Hilbert carriers and classifiers, and $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$ for automorphism rows and their group surface.

The carrier is obtained by pairing a Hilbert carrier row with a Lie-group automorphism row whose source and target projections land back in the Hilbert carrier and are related by the Hilbert classifier. Repacking these projections gives the displayed BHist carrier predicate for $\mathsf{UnitaryGroup}^\uparrow$.

The classifier field is then the restricted componentwise classifier. The Hilbert component is transported by $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$, the automorphism component by $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$, and the

restriction records that both transported endpoints remain classifier-compatible Hilbert carriers. □

Theorem 219.5 (UnitaryGroup operation stability obligation)**.** *The product, inverse, and identity rows of* $\mathsf{UnitaryGroup}^{\uparrow}$ *are the* $\mathsf{LieGroup}^{\uparrow}$ *operation rows restricted to the carrier of Definition 219.4. Product, inverse, and identity witnesses are stable under the displayed classifier because their source, target, and result rows remain Hilbert-carrier-preserving automorphisms.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupOperation stability obligation`

Proof. Project the operation fields of the ten-field roadmap template after the carrier has been fixed. The only available product, inverse, and identity operations are the rows exposed by $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$.

For product, compose two accepted automorphism rows through the $\mathsf{LieGroup}^{\uparrow}$ product row and then project the Hilbert source and target rows. Because each factor preserves the Hilbert carrier, the composite also lands in the same classifier-compatible Hilbert carrier rows.

For inverse and identity, use the corresponding $\mathsf{LieGroup}^{\uparrow}$ rows and apply the same restriction. The inverse row swaps source and target while preserving classifier compatibility, and the identity row is the Hilbert-carrier-preserving automorphism already exposed by the restricted carrier.

Stability follows by transporting every source, target, and result row along the classifier built in Definition 219.4. Repacking those transported rows gives the same product, inverse, and identity witnesses in the restricted $\mathsf{UnitaryGroup}^{\uparrow}$ package. □

Theorem 219.6 (UnitaryGroup inner-product preservation obligation)**.** *Every automorphism carried by* $\mathsf{UnitaryGroup}^{\uparrow}$ *transports the* $\mathsf{Hilbert}^{\uparrow}$ *inner-product classifier to itself. The preservation row is stable under classifier replacement: if the carried automorphism is replaced by an equivalent restricted* $\mathsf{LieGroup}^{\uparrow}$ *automorphism and the Hilbert carrier is replaced by an equivalent* $\mathsf{Hilbert}^{\uparrow}$ *carrier, the transported inner-product row is the same* $\mathsf{Hilbert}^{\uparrow}$ *classifier row.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupInnerProduct preservation obligation`

Proof. Read the inner-product field of the ten-field roadmap template through the $\mathsf{Hilbert}^{\uparrow}$ dependency. The Hilbert certificate supplies the inner-product classifier row, while the restricted automorphism carrier supplies the source and target histories on which that classifier is evaluated.

For a carried automorphism, apply its endpoint-preservation row to the Hilbert source and target projections. The result says that transporting an inner-product observation along the automorphism returns an observation classified by the same $\mathsf{Hilbert}^{\uparrow}$ inner-product row.

For classifier replacement, first transport the Hilbert carrier by the $\mathsf{Hilbert}^{\uparrow}$ classifier and the automorphism by the restricted $\mathsf{LieGroup}^{\uparrow}$ classifier. Then apply the preservation row to the transported endpoints. Repacking gives the identical inner-product-preservation witness for the classified $\mathsf{UnitaryGroup}^{\uparrow}$ row. □

Theorem 219.7 (UnitaryGroup ledger exactness obligation)**.** *The* $\mathsf{UnitaryGroup}^{\uparrow}$ *ledger consists exactly of the* $\mathsf{Hilbert}^{\uparrow}$ *carrier and classifier rows, the* $\mathsf{LieGroup}^{\uparrow}$ *operation rows, inverse, product, and identity witnesses, inner-product-preservation witnesses, classifier-transport rows, and dependency rows. Coverage, soundness, and completeness hold for precisely that package: every accepted unitary-group row exposes those entries, every such entry projects to a valid carried row, and every package with exactly those entries is accepted. No spectral theorem, operator-algebra theorem, or standard bridge claim is part of this ledger.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupLedger exactness obligation`

Proof. Unpack the ledger, exactness, and dependency fields of the ten-field roadmap template. The dependency rows are exactly $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$, and the visible ledger entries are the carrier, classifier, operation, preservation, and transport rows specified in Definition 219.4, Definition 219.5, and Definition 219.6.

Coverage projects an accepted $\mathsf{UnitaryGroup}^\uparrow$ row into those entries: the Hilbert carrier and classifier, the restricted Lie-group automorphism row, the product, inverse, and identity witnesses, the inner-product-preservation witness, and the classifier-transport rows.

Soundness reads any displayed ledger entry back through the corresponding BEDC primitive. Hilbert entries are justified by $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$, operation entries by $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$, and the preservation and transport entries by the restrictions already carried in the unitary-group package.

Completeness repacks a package with exactly these entries as the public obligation surface. Since spectral principles, operator-algebra theorems, and standard bridges are not among the dependency or ledger rows, they cannot be consumed by this certificate. □

Theorem 219.8 (UnitaryGroupUp public NameCert export)**.** *The carrier-classifier row of Definition 219.4, the product, inverse, and identity stability rows of Definition 219.5, the inner-product preservation row of Definition 219.6, and the ledger exactness row of Definition 219.7 assemble the public* $\mathsf{NameCert}_{\mathsf{UnitaryGroup}^\uparrow}$ *export over the source certificates* $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$*. The exported surface consists exactly of restricted Hilbert-carrier-preserving Lie-group automorphisms, their inherited group-operation rows, the displayed Hilbert inner-product-preservation witness, classifier transport for those rows, and the two dependency rows. It exports no spectral theorem, operator-algebra theorem, quantum-channel row, or standard bridge claim.*

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupPublicNameCert export`

Proof. Apply Definition 219.4. This fixes the public carrier as the $\mathsf{Hilbert}^\uparrow$ carrier together with the restricted $\mathsf{LieGroup}^\uparrow$ automorphism row whose source and target remain classifier-compatible Hilbert carriers. It also fixes the public classifier componentwise from the Hilbert carrier classifier and the Lie-group automorphism classifier, with the Hilbert source and target restrictions retained as visible rows.

Apply Definition 219.5. The only product, inverse, and identity rows available to the exported object are the inherited $\mathsf{LieGroup}^\uparrow$ operation rows after restriction to the carrier just fixed. Their stability witnesses say precisely that each operation result remains a Hilbert-carrier-preserving automorphism and that the operation witnesses transport through the public classifier.

Apply Definition 219.6. This supplies the additional unitary condition: every carried automorphism transports the $\mathsf{Hilbert}^\uparrow$ inner-product classifier to itself, and the preservation row is stable when the Hilbert carrier and the restricted Lie-group automorphism are replaced by classifier-equivalent representatives. Thus the public group operations do not merely preserve the carrier; they preserve the displayed Hilbert inner-product row that defines the unitary subcarrier.

Apply Definition 219.7. Coverage projects any accepted $\mathsf{UnitaryGroup}^\uparrow$ row to the Hilbert carrier and classifier rows, the restricted Lie-group operation rows, the product, inverse, identity, inner-product-preservation, classifier-transport, and dependency rows. Soundness repacks exactly those rows as an accepted unitary-group package. Completeness excludes every row not listed in that ledger. Therefore the four obligation rows assemble the public $\mathsf{NameCert}_{\mathsf{UnitaryGroup}^\uparrow}$ export surface over $\mathsf{NameCert}_{\mathsf{Hilbert}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$, with no additional analytic or bridge component. □

Theorem 219.9 (UnitaryGroupUp public standard bridge boundary)**.** *The restricted Hilbert-preserving* $\mathsf{LieGroup}^{\uparrow}$ *automorphism packet exported in Definition 219.8 is the public bridge surface for the standard unitary-group schema. Its bridge-facing rows are exactly the* $\mathsf{Hilbert}^{\uparrow}$ *carrier and classifier rows, the restricted* $\mathsf{LieGroup}^{\uparrow}$ *automorphism row, product, inverse, and identity rows, the displayed inner-product-preservation witness, classifier transport, and the two dependency rows. The boundary adds no spectral theorem, operator-algebra row, quantum-channel row, or host-level equality of unitary operators.*

Proof. Start from the public export theorem. It assembles the carrier, classifier, operation stability, inner-product preservation, and ledger exactness rows over $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$.

The standard unitary-group schema reads exactly the Hilbert carrier together with the group automorphism rows that preserve the displayed inner-product classifier. Those are the rows exported by the restricted packet, and their transport is already supplied by the classifier and ledger obligations.

The ledger exactness theorem excludes all rows outside this list. Therefore the exported packet is a bridge boundary for the standard schema while leaving spectral, operator-algebraic, channel, and host-equality claims outside the chapter. □

Lean4 ✓ `BEDC.Derived.UnitaryGroupUp.UnitaryGroupUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{UnitaryGroup}^{\uparrow}}$.

Constructive story: A unitary-group element is built by pairing a $\mathsf{Hilbert}^{\uparrow}$ carrier row with a $\mathsf{LieGroup}^{\uparrow}$ automorphism row and then requiring the displayed automorphism to preserve the Hilbert inner-product classifier. The carrier accepts only those restricted automorphism packets whose source and target Hilbert rows are classifier-compatible and whose product, inverse, and identity witnesses come from the same $\mathsf{LieGroup}^{\uparrow}$ operation surface; the classifier compares the Hilbert endpoint, automorphism endpoint, operation rows, and preservation witness through the two dependency certificates and hsame transport. The source therefore covers Hilbert-carrier-preserving automorphism rows, the pattern records multiplication, inverse, identity, inner-product preservation, and operation stability under the restricted classifier, and the ledger exposes exactly the Hilbert rows, Lie-group rows, preservation witnesses, classifier transports, and dependency rows. Because the underlying Hilbert and Lie-group classifiers are equivalence relations and the preservation row is transported with the accepted histories, an admitted restricted automorphism supplies the carrier witness, the classifier is reflexive, symmetric, and transitive, and products, inverses, identities, and classified replacements remain in the carrier. A packet h lies in the signature gap of a $\mathsf{UnitaryGroup}^{\uparrow}$ token p when the source-domain accepts its Hilbert row, restricted automorphism row, operation witnesses, preservation witness, and ledger, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names the answer as p; unfolded, the automorphism represented by h is exactly the inner-product-preserving group element encoded by p. The chapter registers this restricted-automorphism construction under the name $\mathsf{UnitaryGroup}^{\uparrow}$; the public bridge boundary is the standard unitary-group schema surface, with no spectral theorem, operator-algebra package, quantum-channel row, or host unitary group introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UnitaryGroup}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public scope binds exactly the carrier and classifier, operation stability, inner-product preservation, ledger exactness, public NameCert export, and standard bridge boundary rows in Definition 219.4, Definition 219.5, Definition 219.6, Definition 219.7, Definition 219.8, and Definition 219.9.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UnitaryGroup}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.UnitaryGroupUp.UnitaryGroupCarrier\_classifier\_obligation`

Bridge status: bridgeChecked

Not claimed: This chapter does not close a group-law completion, spectral theorem, operator-algebra package, or quantum-channel row.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 220

A Concrete Naming Certificate for $\mathsf{DensityMatrix}^{\uparrow}$

The density-matrix interface $\mathsf{DensityMatrix}^{\uparrow}$ packages a positive trace-class operator of trace one on a Hilbert space carrier (a mixed quantum state). It depends on $\mathsf{NameCert}_{\mathsf{OperatorIdeal}^{\uparrow}}$ (chapter 213) and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91).

220.1 Carrier definition

220.2 The certificate

Theorem 220.1 (DensityMatrix carrier obligation)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier supplied by $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$. The carrier row of $\mathsf{DensityMatrix}^{\uparrow}(H)$ is exactly the positive trace-class trace-one subcarrier*

$$\mathsf{Dens}_H(\rho) :\Longleftrightarrow \rho \in \mathsf{TC}(H) \wedge \mathsf{Pos}_H(\rho) \wedge \mathrm{Tr}_H(\rho) = 1.$$

The trace-class membership row $\rho \in \mathsf{TC}(H)$ is read from the $\mathsf{OperatorIdeal}^{\uparrow}$ trace-class surface of Definition 221.1, while the Hilbert carrier, positivity predicate, and trace functional are read over the $\mathsf{Hilbert}^{\uparrow}$ endpoint named in chapter 91. No object is carried by $\mathsf{DensityMatrix}^{\uparrow}(H)$ unless all three displayed rows are present, and any endpoint with all three rows is a carried density matrix.

Lean4 (definition) `BEDC.Derived.DensityMatrixUp.DensityMatrixCarrier`
Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixCarrier carrier obligation rows`
Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixTraceLedgerCarrier rows`

Proof. Unpack the density carrier named in Definition 221.1. Its carrier field is the intersection of the trace-class carrier with the positivity row and the trace-one equation, so the forward direction projects the three displayed witnesses.

For the reverse direction, repack the same three witnesses in the order required by the carrier field. The trace-class witness is supplied by the $\mathsf{OperatorIdeal}^{\uparrow}$ dependency surface, while positivity and trace are rows over the underlying $\mathsf{Hilbert}^{\uparrow}$ carrier. □

Theorem 220.2 (DensityMatrix restricted classifier obligation)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier, and write $\sim_{\mathsf{TC}(H)}$ for the trace-class classifier consumed from the $\mathsf{OperatorIdeal}^{\uparrow}$ boundary. The classifier row of $\mathsf{DensityMatrix}^{\uparrow}(H)$ is the restricted classifier of Definition 220.7:*

$$T \sim_{\mathsf{Dens}(H)} S :\Longleftrightarrow \mathsf{Dens}_H(T) \wedge \mathsf{Dens}_H(S) \wedge T \sim_{\mathsf{TC}(H)} S.$$

Thus every density classifier comparison carries both density endpoints and the inherited trace-class comparison. Its reflexivity, symmetry, transitivity, and transport rows are inherited from the trace-class classifier together with transport of the carrier rows of Definition 220.1.

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixTraceLedgerCarrier positivity trace ledger obligation`
Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixTraceLedgerCarrier restricted classifier transport`

Proof. Unpack Definition 220.7. The comparison is a restricted trace-class comparison, so projection gives the two density endpoints and the underlying trace-class classifier witness.

Reflexivity uses density-carrier reflexivity together with trace-class classifier reflexivity. Symmetry and transitivity keep the endpoint carrier witnesses and apply the corresponding inherited trace-class classifier rows.

For transport, use the carrier description of Definition 220.1. Trace-class membership is transported by the $\mathsf{OperatorIdeal}^{\uparrow}$ classifier row, while positivity and trace-one evidence are transported over the Hilbert endpoint; repacking gives the transported density-carrier endpoint required by the restricted classifier. □

Theorem 220.3 (DensityMatrix positivity and trace ledger obligation)**.** *For a* $\mathsf{Hilbert}^{\uparrow}$ *carrier* H*, the ledger hidden by the* $\mathsf{DensityMatrix}^{\uparrow}(H)$ *seal consists exactly of the trace-class membership witness* $\rho \in \mathsf{TC}(H)$*, positivity evidence* $\mathsf{Pos}_H(\rho)$*, trace-one evidence* $\mathrm{Tr}_H(\rho) = 1$*, the restricted trace-class classifier comparison of Definition 220.7, and the finite affine-mixture weight and bracketing data of Definition 220.4 when that operation is invoked. The seal contains no spectral decomposition, purification, channel action, dilation, or measurement-interface data.*

Proof. The carrier ledger is fixed by Definition 220.1: one trace-class witness, one positivity witness, and one trace-one witness. The classifier ledger is fixed by Definition 220.2: two carried density endpoints together with the inherited trace-class comparison.

When an affine mixture is used, Definition 220.4 adds only the finite list of density entries, the finite real-weight list, nonnegative unit-sum evidence, and the local bracketing endpoint. These data are read over the trace-class vector surface supplied by $\mathsf{OperatorIdeal}^{\uparrow}$ and the scalar rows supplied through the $\mathsf{Hilbert}^{\uparrow}$ dependency.

The named dependency surfaces do not expose spectral decompositions, purifications, channel actions, dilations, or measurement rows at this paper site. Therefore none of those structures can be part of the hidden ledger of the $\mathsf{DensityMatrix}^{\uparrow}(H)$ seal. □

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixTraceLedgerCarrier trace one unit readback`

Definition 220.4 (DensityMatrix affine mixture spine)**.** *Let* H *be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier, and let* $\mathsf{TC}(H)$*,* Pos_H*,* Tr_H*, and* Dens_H *be as in Definition 221.1. Use the real trace-class scalar action* $\odot_H$ *and trace-class addition* $+_{\mathsf{TC}(H)}$ *inherited from the* $\mathsf{OperatorIdeal}^{\uparrow}$ *and* $\mathsf{Hilbert}^{\uparrow}$ *dependency surfaces of chapter 220. For a finite trace-class spine* $xs : \mathsf{ListCarrier}(\mathsf{TC}(H))$*, a finite real-weight spine* $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$*, and an endpoint* $\rho_{\mathrm{mix}} \in \mathsf{TC}(H)$*, define*

$$\mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}}) :\Longleftrightarrow \mathsf{ConvAffSpine}_{\mathsf{Dens}_H}(xs, \lambda, \rho_{\mathrm{mix}})$$

where the right side is the finite affine-spine datum of Definition 208.4 read with carrier $C = \mathsf{Dens}_H$*, scalar action* $\odot_H$*, and vector addition* $+_{\mathsf{TC}(H)}$*. Thus the datum records a position match between density-matrix entries and real weights, nonnegative weights whose finite sum is classified as* $1_{\mathsf{Real}^{\uparrow}}$*, and a recursive bracketing endpoint for the displayed affine mixture.*

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine unary endpoint`

Lemma 220.5 (DensityMatrix binary convex closure)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier. Let $\rho, \sigma \in \mathsf{TC}(H)$ and let a, b be real scalar coefficients. If*

$$\mathsf{Dens}_H(\rho) \wedge \mathsf{Dens}_H(\sigma) \wedge \mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(a) \wedge \mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(b) \wedge a + b = 1,$$

then the binary affine mixture is a carried density matrix:

$$\mathsf{Dens}_H\big((a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \sigma)\big).$$

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine binary convex closure`

Proof. Unpack Dens_H by Definition 221.1. The two input hypotheses give

$$\rho, \sigma \in \mathsf{TC}(H), \qquad \mathsf{Pos}_H(\rho), \mathsf{Pos}_H(\sigma), \qquad \mathrm{Tr}_H(\rho) = \mathrm{Tr}_H(\sigma) = 1.$$

The trace-class carrier is closed under real scalar action and addition by the vector-space surface inherited from the $\mathsf{OperatorIdeal}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ dependencies, so

$$(a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \sigma) \in \mathsf{TC}(H).$$

The positivity cone on trace-class operators is closed under multiplication by nonnegative real scalars and under addition. The two nonnegativity hypotheses therefore give

$$\mathsf{Pos}_H\big((a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \sigma)\big).$$

Trace linearity over the same trace-class vector carrier gives

$$\mathrm{Tr}_H\big((a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \sigma)\big) = a\,\mathrm{Tr}_H(\rho) + b\,\mathrm{Tr}_H(\sigma) = a + b = 1.$$

These are exactly the trace-class, positivity, and trace-one clauses of Definition 221.1 for the displayed affine mixture. □

Theorem 220.6 (DensityMatrix finite affine mixture closure)**.** *Let H be a* $\mathsf{Hilbert}^{\uparrow}$ *carrier. For every finite trace-class spine $xs : \mathsf{ListCarrier}(\mathsf{TC}(H))$, finite real-weight spine $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$, and endpoint $\rho_{\mathrm{mix}} \in \mathsf{TC}(H)$, if*

$$\mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}}),$$

then

$$\mathsf{Dens}_H(\rho_{\mathrm{mix}}).$$

Equivalently, any finite affine mixture of carried $\mathsf{DensityMatrix}^{\uparrow}(H)$ *entries with nonnegative real weights whose finite sum is classified as one is again a carried* $\mathsf{DensityMatrix}^{\uparrow}(H)$ *entry.*

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine finite closure`

Proof. Unpack Definition 220.4. The hypothesis is the finite affine-spine datum of Definition 208.4 specialized to the concrete carrier $C = \mathsf{Dens}_H$ on the trace-class vector carrier $\mathsf{TC}(H)$. It contains both the finite readback of carried entries and nonnegative unit-sum weights, and a local bracketing witness for the endpoint ρ_{mix}.

Induct on that local bracketing witness. In the singleton clause, the endpoint is the displayed trace-class operator ρ, and the clause itself supplies $\mathsf{Dens}_H(\rho)$.

In the binary clause, the induction hypothesis gives that the tail endpoint τ is carried by Dens_H. The same clause supplies a carried head ρ, nonnegative real coefficients a, b, and the local unit-sum identity $a + b = 1$. Applying Definition 220.5 to ρ, τ, a, and b gives

$$\mathsf{Dens}_H\big((a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \tau)\big),$$

which is exactly the endpoint named by the binary bracketing clause.

The induction covers every recursive affine bracketing packaged by $\mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}})$. Hence the finite mixture endpoint ρ_{mix} satisfies $\mathsf{Dens}_H(\rho_{\mathrm{mix}})$. □

Definition 220.7 (DensityMatrix restricted classifier)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier. Write $\sim_{\mathsf{TC}(H)}$ for the trace-class classifier inherited from the $\mathsf{OperatorIdeal}^{\uparrow}$ boundary of Definition 213.6. The density-matrix classifier is the restricted trace-class classifier*

$$T \sim_{\mathsf{Dens}(H)} S :\Longleftrightarrow \mathsf{Dens}_H(T) \wedge \mathsf{Dens}_H(S) \wedge T \sim_{\mathsf{TC}(H)} S.$$

Thus a density classifier comparison records both density-carrier endpoints and the underlying trace-class classifier comparison.

Lean4 (definition) `BEDC.Derived.DensityMatrixUp.DensityMatrixRestrictedClassifier`

Lemma 220.8 (DensityMatrix binary constant affine exactness)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier, and use the notation of Definition 220.7. Let $\rho, \sigma, \rho_0 \in \mathsf{TC}(H)$ and let a, b be real scalar coefficients. Put*

$$E := (a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \sigma).$$

If

$$\mathsf{Dens}_H(E) \wedge \rho \sim_{\mathsf{Dens}(H)} \rho_0 \wedge \sigma \sim_{\mathsf{Dens}(H)} \rho_0 \wedge (a + b) \sim_{\mathsf{Real}^{\uparrow}} 1_{\mathsf{Real}^{\uparrow}},$$

then

$$E \sim_{\mathsf{Dens}(H)} \rho_0.$$

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine binary constant exactness`

Proof. Unpack Definition 220.7 for the two input density-classifier comparisons. They give the density carrier of ρ_0 and trace-class classifier comparisons

$$\rho \sim_{\mathsf{TC}(H)} \rho_0, \qquad \sigma \sim_{\mathsf{TC}(H)} \rho_0.$$

By Definition 213.6, the trace-class carrier inherits the real scalar action, vector addition, and trace-class classifier from the operator-ideal and Hilbert module surface. Reading that surface through Definition 47.19 and Definition 46.18, scalar-action congruence and additive congruence give

$$E \sim_{\mathsf{TC}(H)} (a \odot_H \rho_0) +_{\mathsf{TC}(H)} (b \odot_H \rho_0).$$

The scalar-distributivity row over scalar addition gives the reverse comparison from the displayed sum to the single scalar action:

$$(a \odot_H \rho_0) +_{\mathsf{TC}(H)} (b \odot_H \rho_0) \sim_{\mathsf{TC}(H)} ((a + b) \odot_H \rho_0).$$

Scalar-action congruence applied to $(a + b) \sim_{\mathsf{Real}^{\uparrow}} 1_{\mathsf{Real}^{\uparrow}}$ and classifier reflexivity of the carried endpoint ρ_0 gives

$$((a + b) \odot_H \rho_0) \sim_{\mathsf{TC}(H)} (1_{\mathsf{Real}^{\uparrow}} \odot_H \rho_0).$$

The scalar unit law in the same module surface gives

$$(1_{\mathsf{Real}^{\uparrow}} \odot_H \rho_0) \sim_{\mathsf{TC}(H)} \rho_0.$$

Transitivity of the trace-class classifier composes the four displayed comparisons into

$$E \sim_{\mathsf{TC}(H)} \rho_0.$$

Pair this comparison with the assumed carrier $\mathsf{Dens}_H(E)$ and the carrier of ρ_0 obtained from either input density-classifier comparison. Folding Definition 220.7 gives $E \sim_{\mathsf{Dens}(H)} \rho_0$. □

Theorem 220.9 (DensityMatrix constant affine-spine exactness)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier, and use the notation of Definition 220.7. Let $xs : \mathsf{ListCarrier}(\mathsf{TC}(H))$ be a finite trace-class spine, let $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$ be a finite real-weight spine, and let $\rho_{\mathrm{mix}}, \rho_0 \in \mathsf{TC}(H)$. If*

$$\mathsf{Dens}_H(\rho_0) \wedge \mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}}) \wedge \Big(\forall i : \mathsf{Pos}(xs).\ \mathsf{entry}_{xs}(i) \sim_{\mathsf{Dens}(H)} \rho_0\Big),$$

then

$$\rho_{\mathrm{mix}} \sim_{\mathsf{Dens}(H)} \rho_0.$$

Proof. Unpack $\mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}})$ by Definition 220.4. This is the finite affine-spine datum of Definition 208.4 specialized to the concrete density carrier Dens_H, the trace-class scalar action $\odot_H$, and trace-class addition $+_{\mathsf{TC}(H)}$. It contains a local bracketing witness $\mathsf{Bracket}_{\mathsf{Dens}_H}(xs, \lambda, \rho_{\mathrm{mix}})$.

Prove the stronger statement for every displayed sub-bracketing: if every leaf displayed in that sub-spine is density-classifier equal to ρ_0, then the sub-bracketing endpoint is density-classifier equal to ρ_0. Induct on the local bracketing witness.

In the singleton clause, the endpoint is the displayed leaf ρ. The finite-position hypothesis applied to the unique displayed position gives

$$\rho \sim_{\mathsf{Dens}(H)} \rho_0,$$

which is exactly the endpoint claim.

In the binary-extension clause, write the endpoint as

$$E = (a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \tau).$$

The head position of the displayed cons spine gives

$$\rho \sim_{\mathsf{Dens}(H)} \rho_0.$$

The tail positions inherit the same leafwise hypothesis, so the induction hypothesis gives

$$\tau \sim_{\mathsf{Dens}(H)} \rho_0.$$

The binary bracketing clause supplies $\mathsf{Dens}_H(\rho)$, the nonnegative coefficient rows, the local unit-sum comparison $(a + b) \sim_{\mathsf{Real}^{\uparrow}} 1_{\mathsf{Real}^{\uparrow}}$, and the tail bracketing witness. Since the induction hypothesis includes $\mathsf{Dens}_H(\tau)$ through Definition 220.7, the specialized binary affine-combination row of Definition 208.1, concretely proved for density matrices in Definition 220.5, gives

$$\mathsf{Dens}_H(E).$$

Now apply Definition 220.8 to ρ, τ, ρ_0, a, and b. The result is

$$E \sim_{\mathsf{Dens}(H)} \rho_0,$$

which is the endpoint claim for the binary node.

The induction applies to the full bracketing witness packaged by the original affine-spine datum. Therefore the endpoint ρ_{mix} is density-classifier equal to the fixed carried density matrix ρ_0. □

Lean4 ✓ `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine restricted constant exactness`

Closure status for $\mathsf{NameCert}_{\mathsf{DensityMatrix}^{\uparrow}}$.

Constructive story: A density matrix over a carried Hilbert source H is obtained by reading the trace-class carrier from OperatorIdeal$^{\uparrow}$, then keeping only those endpoints ρ that also display positivity and a trace classified with the real unit. The carrier accepts exactly positive trace-class trace-one endpoints, and the classifier is the inherited trace-class classifier restricted to pairs that both remain in that density carrier, so equality never comes from a host matrix identity outside the displayed rows. The source therefore covers Hilbert-indexed trace-class endpoints, the pattern records positivity, trace-one readback, binary and finite affine-mixture closure, and constant-mixture exactness, and the ledger keeps the trace-class witness, positivity witness, trace-one witness, restricted classifier comparison, real weights, list positions, and affine bracketing data visible when mixtures are formed. Since the trace-class classifier is reflexive, symmetric, and transitive and the carrier rows transport through the Hilbert$^{\uparrow}$, OperatorIdeal$^{\uparrow}$, and Real$^{\uparrow}$ surfaces, an admitted density endpoint supplies the carrier witness and classified replacements remain positive, trace-class, and trace one. A row h lies in the signature gap of a DensityMatrix$^{\uparrow}$ token p when the source-domain accepts its Hilbert source, trace-class endpoint, positivity row, trace-one row, and any finite affine ledger, the signature relation pairs that row with an answer endpoint and evidence ledger, and token introduction names the answer as p; unfolded, the endpoint is precisely the positive trace-one trace-class operator encoded by p, with finite mixtures evaluated by the displayed affine spine. The finite affine-mixture certificate is the paper bridge from the concrete density-matrix construction to the abstract density-matrix schema. The chapter registers this trace-class trace-one construction under the name DensityMatrix$^{\uparrow}$; mixed quantum states are certified by positivity and trace ledgers over OperatorIdealUp and HilbertUp, with no spectral decomposition, purification, channel action, or host density-matrix type introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DensityMatrix}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature finite affine-mixture and constant affine-spine boundary is closed for the carrier, restricted classifier, positivity and trace-one ledger, affine-mixture spine, finite closure, constant affine-spine exactness rows, and paper bridge through the finite affine-mixture certificate over the Hilbert$^{\uparrow}$, OperatorIdeal$^{\uparrow}$, Real$^{\uparrow}$, and ListCarrier surfaces used by the cited proofs.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DensityMatrix}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DensityMatrixUp.DensityMatrixAffineMixtureSpine\_finite\_closure`

Bridge status: paperBridge

Not claimed: This chapter does not close the full quantum-state interface, spectral decomposition, purification, channel interaction, or standard-model density-matrix claims.

Upgrade path: No higher paper-side theory grade remains; standard bridge checking is a separate formal-axis target beyond the finite affine-mixture paper bridge.

Chapter 221

A Concrete Naming Certificate for $\mathsf{QuantumChannel}^{\uparrow}$

The quantum-channel interface $\mathsf{QuantumChannel}^{\uparrow}$ packages a completely positive trace-preserving (CPTP) linear map between density-matrix spaces. It depends on $\mathsf{NameCert}_{\mathsf{DensityMatrix}^{\uparrow}}$ (chapter 220) and $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ (chapter 48).

221.1 Carrier definition

221.2 The certificate

Definition 221.1 (DensityMatrix trace-class carrier)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier, and let $\mathsf{TC}(H)$ be the trace-class operator-ideal carrier supplied for H by the $\mathsf{OperatorIdeal}^{\uparrow}$ dependency of chapter 220. Write $\mathsf{Pos}_H(T)$ for positivity of a trace-class operator on H and $\mathrm{Tr}_H(T)$ for its trace. The concrete $\mathsf{DensityMatrix}^{\uparrow}(H)$ carrier is*

$$\mathsf{Dens}_H(\rho) :\Longleftrightarrow \rho \in \mathsf{TC}(H) \wedge \mathsf{Pos}_H(\rho) \wedge \mathrm{Tr}_H(\rho) = 1.$$

Thus a carried density matrix is exactly positive trace-class trace-one data on the Hilbert carrier named by section 220.1.

Definition 221.2 (QuantumChannel CPTP carrier)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers. A map $\Phi : \mathsf{TC}(H) \to \mathsf{TC}(K)$ is carried by $\mathsf{QuantumChannel}^{\uparrow}(H, K)$, written $\mathsf{QChan}_{H,K}(\Phi)$, when it carries the $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K))$ map-code of Definition 48.1 and satisfies the following two operator rows:*

1. *complete positivity: for every finite ancilla Hilbert carrier A and every positive trace-class operator X on $A \otimes H$, the amplified operator $(\mathrm{id}_A \otimes \Phi)(X)$ is positive trace-class on $A \otimes K$;*
2. *trace preservation: for every $T \in \mathsf{TC}(H)$,*

$$\mathrm{Tr}_K(\Phi(T)) = \mathrm{Tr}_H(T).$$

The carrier is the CPTP linear-map interface announced in chapter 221; its middle space for composition is the displayed trace-class carrier $\mathsf{TC}(K)$.

Lemma 221.3 (QuantumChannel density-matrix image)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers. If*

$$\mathsf{QChan}_{H,K}(\Phi) \qquad \textit{and} \qquad \mathsf{Dens}_H(\rho),$$

then

$$\mathsf{Dens}_K(\Phi(\rho)).$$

Proof. Unpack $\mathsf{Dens}_H(\rho)$ by Definition 221.1. It gives that ρ is trace-class on H, positive, and has trace one. Unpack $\mathsf{QChan}_{H,K}(\Phi)$ by Definition 221.2. The carried $\mathsf{LinMap}^{\uparrow}$ row supplies output closure, so $\Phi(\rho)$ is trace-class on K.

Apply complete positivity with the one-dimensional ancilla. The amplification by the one-dimensional identity is the original map on trace-class operators, hence positivity of ρ gives positivity of $\Phi(\rho)$ on K. Trace preservation gives

$$\mathrm{Tr}_K(\Phi(\rho)) = \mathrm{Tr}_H(\rho) = 1.$$

The three rows are exactly the Dens_K carrier clauses in Definition 221.1. □

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelSingleton densitymatrix image`

Theorem 221.4 (QuantumChannel composition closure)**.** *Let H, K, and L be* $\mathsf{Hilbert}^{\uparrow}$ *carriers with trace-class spaces* $\mathsf{TC}(H)$*,* $\mathsf{TC}(K)$*, and* $\mathsf{TC}(L)$*. If*

$$\mathsf{QChan}_{H,K}(\Phi) \qquad \textit{and} \qquad \mathsf{QChan}_{K,L}(\Psi),$$

then the composite map $\Psi \circ \Phi : \mathsf{TC}(H) \to \mathsf{TC}(L)$ *is carried by* $\mathsf{QuantumChannel}^{\uparrow}(H, L)$*:*

$$\mathsf{QChan}_{H,L}(\Psi \circ \Phi).$$

Moreover every $\mathsf{DensityMatrix}^{\uparrow}(H)$ *input is sent to a* $\mathsf{DensityMatrix}^{\uparrow}(L)$ *output:*

$$\mathsf{Dens}_H(\rho) \Longrightarrow \mathsf{Dens}_L((\Psi \circ \Phi)(\rho)).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine composition closure`

Proof. Unpack the two channel carriers by Definition 221.2. Their $\mathsf{LinMap}^{\uparrow}$ rows are maps

$$\Phi : \mathsf{TC}(H) \to \mathsf{TC}(K) \qquad \text{and} \qquad \Psi : \mathsf{TC}(K) \to \mathsf{TC}(L)$$

with the matching middle trace-class carrier $\mathsf{TC}(K)$. The trace-class carriers are module and vector-space carriers inherited from the operator-ideal and Hilbert surfaces cited in chapter 220. Therefore the composition row of Definition 48.50, concretely supplied by Definition 48.60, gives the carried $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(L))$ map-code for $\Psi \circ \Phi$.

For complete positivity, let A be a finite ancilla Hilbert carrier and let X be a positive trace-class operator on $A \otimes H$. Complete positivity of Φ gives that

$$Y := (\mathrm{id}_A \otimes \Phi)(X)$$

is positive trace-class on $A \otimes K$. Complete positivity of Ψ applied to Y gives that

$$(\mathrm{id}_A \otimes \Psi)(Y)$$

is positive trace-class on $A \otimes L$. The amplified-map ledger is functorial in the operator argument, so

$$(\mathrm{id}_A \otimes \Psi)((\mathrm{id}_A \otimes \Phi)(X)) = (\mathrm{id}_A \otimes (\Psi \circ \Phi))(X).$$

Thus $\Psi \circ \Phi$ is completely positive.

For trace preservation, let $T \in \mathsf{TC}(H)$. Output closure for Φ gives $\Phi(T) \in \mathsf{TC}(K)$. Trace preservation for Ψ, followed by trace preservation for Φ, gives

$$\mathrm{Tr}_L((\Psi \circ \Phi)(T)) = \mathrm{Tr}_L(\Psi(\Phi(T))) = \mathrm{Tr}_K(\Phi(T)) = \mathrm{Tr}_H(T).$$

The linear-map row, complete positivity row, and trace-preservation row are precisely the three clauses of Definition 221.2; hence $\mathsf{QChan}_{H,L}(\Psi \circ \Phi)$ holds.

Finally assume $\mathsf{Dens}_H(\rho)$. Applying Definition 221.3 to the composite channel just constructed gives

$$\mathsf{Dens}_L((\Psi \circ \Phi)(\rho)),$$

which is the asserted $\mathsf{DensityMatrix}^{\uparrow}$ output closure. □

Definition 221.5 (QuantumChannel affine mixture spine)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers, and write*

$$\mathsf{Map}_{H,K} := \mathsf{TC}(H) \to \mathsf{TC}(K).$$

Use the pointwise real scalar action and pointwise trace-class addition on maps,

$$(a \odot_{H,K} \Phi)(T) := a \odot_K \Phi(T), \qquad (\Phi \oplus_{H,K} \Psi)(T) := \Phi(T) +_{\mathsf{TC}(K)} \Psi(T).$$

For a finite map spine $\Phi_s : \mathsf{ListCarrier}(\mathsf{Map}_{H,K})$, a finite real-weight spine $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$, and an endpoint $\Phi_{\mathrm{mix}} : \mathsf{Map}_{H,K}$, define

$$\mathsf{QChanAffSpine}_{H,K}(\Phi_s, \lambda, \Phi_{\mathrm{mix}}) :\Longleftrightarrow \mathsf{ConvAffSpine}_{\mathsf{QChan}_{H,K}}(\Phi_s, \lambda, \Phi_{\mathrm{mix}}),$$

where the right side is the finite affine-spine datum of Definition 208.4 read with carrier $C = \mathsf{QChan}_{H,K}$, scalar action $\odot_{H,K}$, and addition $\oplus_{H,K}$. Thus the datum records carried channel entries, nonnegative real weights with total weight classified as one, and a recursive pointwise affine endpoint.

Lemma 221.6 (QuantumChannel binary convex mixture closure)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers. Let $\Phi, \Psi : \mathsf{TC}(H) \to \mathsf{TC}(K)$ and let a, b be real scalar coefficients. Define $\Phi_{\mathrm{mix}} : \mathsf{TC}(H) \to \mathsf{TC}(K)$ pointwise by*

$$\Phi_{\mathrm{mix}}(T) = (a \odot_K \Phi(T)) +_{\mathsf{TC}(K)} (b \odot_K \Psi(T)).$$

If

$$\mathsf{QChan}_{H,K}(\Phi) \wedge \mathsf{QChan}_{H,K}(\Psi) \wedge \mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(a) \wedge \mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(b) \wedge a + b = 1,$$

then

$$\mathsf{QChan}_{H,K}(\Phi_{\mathrm{mix}}).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelSingleton binary convex mixture closure`

Proof. Unpack the two channel hypotheses by Definition 221.2. Their first rows carry the two $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K))$ map-codes of Definition 48.1. The trace-class carriers are real vector-space and module carriers inherited from the $\mathsf{OperatorIdeal}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ dependency surfaces cited in chapter 220.

The pointwise formula for Φ_{mix} gives a carried linear-map code. Indeed, output closure follows from output closure for Φ and Ψ together with scalar-action and addition closure in $\mathsf{TC}(K)$. Source-classifier descent follows by applying the descent rows for Φ and Ψ, then scalar-action congruence and additive congruence in the target trace-class module. Additivity follows from the additivity rows for Φ and Ψ, scalar distributivity over target addition, and the middle-four additive

law in the target module. Scalar compatibility follows from the scalar rows for Φ and Ψ, scalar associativity and commutativity in the real field, and target scalar distributivity over addition. Zero preservation follows from the zero rows for Φ and Ψ, the target zero-action laws, and the target additive unit law. These are exactly the rows of Definition 48.59; equivalently, they are the pointwise-sum mechanism of Definition 48.66 applied after checking the two pointwise scalar multiples. Hence Φ_{mix} carries the required $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K))$ map-code.

For complete positivity, let A be a finite ancilla Hilbert carrier and let X be a positive trace-class operator on $A \otimes H$. Complete positivity of Φ and Ψ gives positive trace-class operators

$$Y_\Phi := (\mathrm{id}_A \otimes \Phi)(X), \qquad Y_\Psi := (\mathrm{id}_A \otimes \Psi)(X)$$

on $A \otimes K$. The amplified-map ledger is pointwise affine in the channel argument, so

$$(\mathrm{id}_A \otimes \Phi_{\mathrm{mix}})(X) = (a \odot_{A\otimes K} Y_\Phi) +_{\mathsf{TC}(A\otimes K)} (b \odot_{A\otimes K} Y_\Psi).$$

The positivity cone on trace-class operators is closed under nonnegative real scalar action and addition, and the hypotheses supply $\mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(a)$ and $\mathsf{NonNeg}_{\mathsf{Real}^{\uparrow}}(b)$. Therefore $(\mathrm{id}_A \otimes \Phi_{\mathrm{mix}})(X)$ is positive trace-class on $A \otimes K$. Since A and X were arbitrary, Φ_{mix} is completely positive.

For trace preservation, let $T \in \mathsf{TC}(H)$. Trace linearity over the trace-class vector carrier and trace preservation for Φ and Ψ give

$$\begin{aligned}
\mathrm{Tr}_K(\Phi_{\mathrm{mix}}(T)) &= a\,\mathrm{Tr}_K(\Phi(T)) + b\,\mathrm{Tr}_K(\Psi(T)) \\
&= a\,\mathrm{Tr}_H(T) + b\,\mathrm{Tr}_H(T) \\
&= (a+b)\,\mathrm{Tr}_H(T) \\
&= \mathrm{Tr}_H(T).
\end{aligned}$$

Thus the linear-map row, complete-positivity row, and trace-preservation row required by Definition 221.2 all hold for Φ_{mix}. □

Theorem 221.7 (QuantumChannel finite convex mixture closure)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers. For every finite map spine $\Phi_s : \mathsf{ListCarrier}(\mathsf{Map}_{H,K})$, finite real-weight spine $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$, and endpoint $\Phi_{\mathrm{mix}} : \mathsf{Map}_{H,K}$, if*

$$\mathsf{QChanAffSpine}_{H,K}(\Phi_s, \lambda, \Phi_{\mathrm{mix}}),$$

then the pointwise affine mixture is carried by $\mathsf{QuantumChannel}^{\uparrow}(H, K)$:

$$\mathsf{QChan}_{H,K}(\Phi_{\mathrm{mix}}).$$

Equivalently, any finite pointwise affine mixture of carried $\mathsf{QChan}_{H,K}$ entries with nonnegative real weights whose finite sum is classified as one is again a carried quantum channel from H to K.

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine finite closure`

Proof. Unpack Definition 221.5. The hypothesis is the finite affine-spine datum of Definition 208.4 specialized to the concrete carrier $C = \mathsf{QChan}_{H,K}$ on the pointwise map carrier $\mathsf{Map}_{H,K}$. It contains the visible finite readback of carried channel entries and nonnegative unit-sum weights, together with a local bracketing witness for the endpoint Φ_{mix}.

Induct on the local bracketing witness. In the singleton clause, the endpoint is the displayed map Φ, and the clause itself supplies $\mathsf{QChan}_{H,K}(\Phi)$.

In the binary-extension clause, the induction hypothesis gives that the tail endpoint Θ is carried by $\mathsf{QChan}_{H,K}$. The same clause supplies a carried head map Φ, nonnegative real coefficients a, b, and the local unit-sum identity $a + b = 1$. Its endpoint is the pointwise map

$$T \longmapsto (a \odot_K \Phi(T)) +_{\mathsf{TC}(K)} (b \odot_K \Theta(T)).$$

Applying Definition 221.6 to Φ, Θ, a, and b gives that this endpoint is carried by $\mathsf{QChan}_{H,K}$.

The induction covers every recursive affine bracketing packaged by $\mathsf{QChanAffSpine}_{H,K}(\Phi_s, \lambda, \Phi_{\mathrm{mix}})$. Hence $\mathsf{QChan}_{H,K}(\Phi_{\mathrm{mix}})$ holds. □

Theorem 221.8 (QuantumChannel finite density-mixture preservation)**.** *Let H and K be $\mathsf{Hilbert}^{\uparrow}$ carriers, let $\Phi : \mathsf{TC}(H) \to \mathsf{TC}(K)$, let $xs : \mathsf{ListCarrier}(\mathsf{TC}(H))$ be a finite trace-class spine, let $\lambda : \mathsf{ListCarrier}(\mathsf{Real}^{\uparrow})$ be a finite real-weight spine, and let $\rho_{\mathrm{mix}} \in \mathsf{TC}(H)$. Write $\Phi_* xs$ for the entrywise image spine defined by*

$$\Phi_*(\mathsf{nil}) := \mathsf{nil}, \qquad \Phi_*(\mathsf{cons}(\rho, xs')) := \mathsf{cons}(\Phi(\rho), \Phi_*(xs')).$$

If

$$\mathsf{QChan}_{H,K}(\Phi) \qquad \textit{and} \qquad \mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}}),$$

then

$$\mathsf{DensAffSpine}_K(\Phi_* xs, \lambda, \Phi(\rho_{\mathrm{mix}})) \qquad \textit{and} \qquad \mathsf{Dens}_K(\Phi(\rho_{\mathrm{mix}})).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine density mixture preservation`

Proof. Unpack $\mathsf{QChan}_{H,K}(\Phi)$ by Definition 221.2. Its carried $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K))$ row, read through Definition 48.1, supplies output closure, additivity, and scalar compatibility on the trace-class vector carriers. Hence for carried trace-class endpoints T, U and real scalars a, b,

$$\Phi\big((a \odot_H T) +_{\mathsf{TC}(H)} (b \odot_H U)\big) = (a \odot_K \Phi(T)) +_{\mathsf{TC}(K)} (b \odot_K \Phi(U)).$$

Unpack $\mathsf{DensAffSpine}_H(xs, \lambda, \rho_{\mathrm{mix}})$ by Definition 220.4. This is the affine-spine datum of Definition 208.4 for the concrete carrier Dens_H. The global readback gives a position match between xs and λ, carried source entries, nonnegative weights, and unit finite weight sum. The recursive definition of $\Phi_* xs$ has the same finite positions as xs, so the same position match applies to $\Phi_* xs$ and λ. For each displayed source entry ρ, the readback gives $\mathsf{Dens}_H(\rho)$; applying Definition 221.3 gives $\mathsf{Dens}_K(\Phi(\rho))$. The weight nonnegativity clauses and the finite unit-sum clause are unchanged, since the weight spine is still λ.

It remains to rebuild the local bracketing witness over K. Induct on the source bracketing witness packaged by Definition 208.4. In the singleton clause, the source clause supplies $\mathsf{Dens}_H(\rho)$, nonnegativity of the displayed weight, and classification of that weight as the scalar unit. The density-image lemma gives $\mathsf{Dens}_K(\Phi(\rho))$, and the singleton bracketing clause over K gives the target bracket for $\mathsf{cons}(\Phi(\rho), \mathsf{nil})$ with endpoint $\Phi(\rho)$.

In the binary-extension clause, write the source endpoint as

$$z = (a \odot_H \rho) +_{\mathsf{TC}(H)} (b \odot_H \tau).$$

The induction hypothesis gives the target tail bracket with endpoint $\Phi(\tau)$. The source clause supplies $\mathsf{Dens}_H(\rho)$, nonnegativity of a and b, and the same local unit-sum comparison. By Definition 221.3, $\mathsf{Dens}_K(\Phi(\rho))$. The binary bracketing clause over K therefore builds the endpoint

$$(a \odot_K \Phi(\rho)) +_{\mathsf{TC}(K)} (b \odot_K \Phi(\tau)).$$

The displayed linearity identity rewrites this target endpoint as $\Phi(z)$. Thus the binary target bracket matches the entrywise image spine and has endpoint $\Phi(z)$.

The induction supplies the target bracketing witness for the whole source witness, and the global readback was already rebuilt. Repacking these two rows by Definition 220.4 gives

$$\mathsf{DensAffSpine}_K(\Phi_* xs, \lambda, \Phi(\rho_{\mathrm{mix}})).$$

Applying Definition 220.6 to this target density affine spine gives $\mathsf{Dens}_K(\Phi(\rho_{\mathrm{mix}}))$. □

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine density preservation`

Theorem 221.9 (QuantumChannel identity channel is CPTP)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier, and let $\mathrm{id}_{\mathsf{TC}(H)} : \mathsf{TC}(H) \to \mathsf{TC}(H)$ be the identity function on the trace-class carrier supplied by the $\mathsf{OperatorIdeal}^{\uparrow}$ dependency of chapter 220. Then*

$$\mathsf{QChan}_{H,H}(\mathrm{id}_{\mathsf{TC}(H)}).$$

Moreover every carried density matrix is sent to the same carried density-matrix endpoint:

$$\mathsf{Dens}_H(\rho) \Longrightarrow \mathrm{id}_{\mathsf{TC}(H)}(\rho) = \rho \wedge \mathsf{Dens}_H(\mathrm{id}_{\mathsf{TC}(H)}(\rho)).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelSingleton identity channel cptp`

Proof. The trace-class carrier $\mathsf{TC}(H)$ is a module carrier over the scalar certificate used by the trace-class operator-ideal surface. Applying Definition 48.61 with $M = \mathsf{TC}(H)$ gives the module identity certificate for $\mathrm{id}_{\mathsf{TC}(H)}$. Read through Definition 48.1, this is exactly the carried $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(H))$ map-code required as the first row of Definition 221.2.

For complete positivity, let A be a finite ancilla Hilbert carrier and let X be a positive trace-class operator on $A \otimes H$. The amplified identity map is the identity on the trace-class carrier over $A \otimes H$:

$$(\mathrm{id}_A \otimes \mathrm{id}_{\mathsf{TC}(H)})(X) = X.$$

The input X is already positive trace-class on $A \otimes H$, and the target channel also has Hilbert carrier H. Hence the complete-positivity row of Definition 221.2 holds.

For trace preservation, let $T \in \mathsf{TC}(H)$. Since the map is the identity on $\mathsf{TC}(H)$,

$$\mathrm{Tr}_H(\mathrm{id}_{\mathsf{TC}(H)}(T)) = \mathrm{Tr}_H(T).$$

Thus the linear-map row, complete-positivity row, and trace-preservation row are precisely the three clauses of Definition 221.2, so

$$\mathsf{QChan}_{H,H}(\mathrm{id}_{\mathsf{TC}(H)})$$

holds.

Now assume $\mathsf{Dens}_H(\rho)$. The identity equation gives $\mathrm{id}_{\mathsf{TC}(H)}(\rho) = \rho$. Applying Definition 221.3 to the channel just constructed gives

$$\mathsf{Dens}_H(\mathrm{id}_{\mathsf{TC}(H)}(\rho)),$$

which is the asserted density-matrix output clause. □

Theorem 221.10 (QuantumChannel composition associativity law)**.** *Let H, K, L, and J be* $\mathsf{Hilbert}^{\uparrow}$ *carriers with trace-class spaces* $\mathsf{TC}(H)$, $\mathsf{TC}(K)$, $\mathsf{TC}(L)$, *and* $\mathsf{TC}(J)$. *Write* $\sim_{\mathsf{TC}(J)}$ *for the classifier on the final trace-class module inherited from the operator-ideal certificate. If*

$$\mathsf{QChan}_{H,K}(\Phi), \qquad \mathsf{QChan}_{K,L}(\Psi), \qquad \mathsf{QChan}_{L,J}(\Theta),$$

then both bracketings of the triple composite are carried by $\mathsf{QuantumChannel}^{\uparrow}(H, J)$*:*

$$\mathsf{QChan}_{H,J}((\Theta \circ \Psi) \circ \Phi) \qquad \textit{and} \qquad \mathsf{QChan}_{H,J}(\Theta \circ (\Psi \circ \Phi)).$$

Their underlying map-code endpoints are related by the pointwise $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(J))$ *classifier:*

$$(\Theta \circ \Psi) \circ \Phi \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(J))} \Theta \circ (\Psi \circ \Phi).$$

Equivalently, every carried trace-class endpoint $T : \mathsf{TC}(H)$ *satisfies*

$$((\Theta \circ \Psi) \circ \Phi)(T) \sim_{\mathsf{TC}(J)} (\Theta \circ (\Psi \circ \Phi))(T).$$

Moreover, every carried density-matrix input is sent to carried density-matrix outputs by both bracketings:

$$\mathsf{Dens}_H(\rho) \Longrightarrow \mathsf{Dens}_J(((\Theta \circ \Psi) \circ \Phi)(\rho)) \wedge \mathsf{Dens}_J((\Theta \circ (\Psi \circ \Phi))(\rho)).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine composition assoc`

Proof. Apply Definition 221.4 to Φ and Ψ. This gives

$$\mathsf{QChan}_{H,L}(\Psi \circ \Phi).$$

Applying the same theorem to $\Psi \circ \Phi$ and Θ gives

$$\mathsf{QChan}_{H,J}(\Theta \circ (\Psi \circ \Phi)).$$

In the other bracketing, first apply Definition 221.4 to Ψ and Θ, obtaining

$$\mathsf{QChan}_{K,J}(\Theta \circ \Psi).$$

Applying the same theorem to Φ and $\Theta \circ \Psi$ gives

$$\mathsf{QChan}_{H,J}((\Theta \circ \Psi) \circ \Phi).$$

By Definition 221.2, these two carrier conclusions include the carried $\mathsf{LinMap}^{\uparrow}$ map-code row, complete positivity for every finite ancilla, and trace preservation on $\mathsf{TC}(H)$.

It remains to compare the two carried map-code endpoints. Unpack the three channel hypotheses by Definition 221.2. Their first rows are carried $\mathsf{LinMap}^{\uparrow}$ map-codes, read concretely as the module-linear packages of Definition 48.59 through Definition 48.1. The trace-class carriers are module carriers over the common scalar source inherited from the $\mathsf{OperatorIdeal}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ dependency surfaces cited in chapter 220. Applying Definition 48.62 with

$$M = \mathsf{TC}(H), \qquad N = \mathsf{TC}(K), \qquad P = \mathsf{TC}(L), \qquad Q = \mathsf{TC}(J),$$

and with $f = \Phi$, $g = \Psi$, and $k = \Theta$, gives the pointwise $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(J))$ classifier comparison

$$(\Theta \circ \Psi) \circ \Phi \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(J))} \Theta \circ (\Psi \circ \Phi).$$

This is exactly the classifier specified in Definition 48.49: it is pointwise comparison in the final trace-class classifier, together with the two carried map-code endpoints already supplied by the two channel-carrier conclusions above.

Concretely, let T be a carried trace-class endpoint of $\mathsf{TC}(H)$. The output-closure rows of the three input channel map-codes make $\Phi(T)$ carried in $\mathsf{TC}(K)$, $\Psi(\Phi(T))$ carried in $\mathsf{TC}(L)$, and $\Theta(\Psi(\Phi(T)))$ carried in $\mathsf{TC}(J)$. Both bracketed composites evaluate to this same endpoint:

$$((\Theta \circ \Psi) \circ \Phi)(T) = \Theta(\Psi(\Phi(T))) = (\Theta \circ (\Psi \circ \Phi))(T).$$

Classifier reflexivity for the final trace-class module therefore gives

$$((\Theta \circ \Psi) \circ \Phi)(T) \sim_{\mathsf{TC}(J)} (\Theta \circ (\Psi \circ \Phi))(T),$$

which is the displayed pointwise form.

Finally assume $\mathsf{Dens}_H(\rho)$. Applying Definition 221.3 to the channel $\mathsf{QChan}_{H,J}((\Theta \circ \Psi) \circ \Phi)$ gives

$$\mathsf{Dens}_J(((\Theta \circ \Psi) \circ \Phi)(\rho)).$$

Applying the same lemma to $\mathsf{QChan}_{H,J}(\Theta \circ (\Psi \circ \Phi))$ gives

$$\mathsf{Dens}_J((\Theta \circ (\Psi \circ \Phi))(\rho)).$$

These are the two asserted density-output clauses. □

Definition 221.11 (Unitary trace-class conjugation map)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier and let $U : H \to H$ be an endpoint carried by the $\mathsf{UnitaryGroup}^{\uparrow}(H)$ carrier of section 219.1. Write $U^{-1} : H \to H$ for the inverse endpoint supplied by the same unitary automorphism row. For $T \in \mathsf{TC}(H)$ define the unitary trace-class conjugation expression by*

$$\mathrm{Ad}_U^{\mathsf{TC}}(T) := U \circ T \circ U^{-1}.$$

The expression is read in the bounded-operator composition ring over H from Definition 213.1; its trace-class output closure is the first carrier row proved in Definition 221.12.

Theorem 221.12 (Unitary conjugation is a QuantumChannel)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier. If $U : H \to H$ is carried by the $\mathsf{UnitaryGroup}^{\uparrow}(H)$ automorphism carrier and U^{-1} is its carried inverse, then the conjugation map of Definition 221.11 is carried by $\mathsf{QuantumChannel}^{\uparrow}(H, H)$:*

$$\mathsf{QChan}_{H,H}(\mathrm{Ad}_U^{\mathsf{TC}}).$$

Consequently every carried density matrix is sent to a carried density matrix:

$$\mathsf{Dens}_H(\rho) \Longrightarrow \mathsf{Dens}_H(\mathrm{Ad}_U^{\mathsf{TC}}(\rho)).$$

Proof. Unpack the $\mathsf{UnitaryGroup}^{\uparrow}$ carrier cited in section 219.1. The unitary witness supplies bounded $\mathsf{LinMap}^{\uparrow}(H, H)$ endpoints U and U^{-1}, the two inverse equations against the Hilbert identity, and the inner-product preservation row identifying U^{-1} with the adjoint-side action of U. Hence both endpoints are carried bounded operators in the ambient operator surface Op_H of Definition 213.1.

First prove trace-class output closure. Let $T \in \mathsf{TC}(H)$. Apply the right absorption half of Definition 213.5 to the carried bounded operator U^{-1} and the trace-class operator T. This gives

$T \circ U^{-1} \in \mathsf{TC}(H)$. Apply the left absorption half of the same theorem to the carried bounded operator U and this trace-class operator. The result is

$$U \circ T \circ U^{-1} \in \mathsf{TC}(H),$$

which is output closure for $\mathrm{Ad}_U^{\mathsf{TC}}$.

The same bounded-composition rows give the linear-map package. Source-classifier descent follows from the operator-ideal classifier transport row, since composing on the left by the fixed carried bounded operator U and on the right by the fixed carried bounded operator U^{-1} preserves the pointwise bounded-operator classifier. For carried $T, S \in \mathsf{TC}(H)$ and a carried real scalar a, bilinearity of bounded-operator composition over the trace-class module gives

$$\mathrm{Ad}_U^{\mathsf{TC}}(T +_{\mathsf{TC}(H)} S) = \mathrm{Ad}_U^{\mathsf{TC}}(T) +_{\mathsf{TC}(H)} \mathrm{Ad}_U^{\mathsf{TC}}(S)$$

and

$$\mathrm{Ad}_U^{\mathsf{TC}}(a \odot_H T) = a \odot_H \mathrm{Ad}_U^{\mathsf{TC}}(T).$$

The zero row is the same calculation at the trace-class zero operator. These are the output-closure, descent, additive, scalar, and zero rows of Definition 48.59, read through the concrete $\mathsf{LinMap}^{\uparrow}$ carrier in Definition 48.1. Thus $\mathrm{Ad}_U^{\mathsf{TC}}$ carries the required linear-map row in Definition 221.2.

For complete positivity, let A be a finite ancilla Hilbert carrier and let X be a positive trace-class operator on $A \otimes H$. Put

$$V_A := \mathrm{id}_A \otimes U, \qquad V_A^{-1} := \mathrm{id}_A \otimes U^{-1}.$$

The tensor row for Hilbert carriers makes V_A a carried unitary automorphism of $A \otimes H$ with inverse V_A^{-1}. The amplified conjugation map is the corresponding bounded conjugation:

$$(\mathrm{id}_A \otimes \mathrm{Ad}_U^{\mathsf{TC}})(X) = V_A \circ X \circ V_A^{-1}.$$

Trace-class membership of this amplified output follows from Definition 213.5 on $A \otimes H$. For positivity, evaluate the quadratic-form predicate at an arbitrary carried vector z of $A \otimes H$. The unitary inverse-adjoint row gives

$$\langle V_A \circ X \circ V_A^{-1} z, z\rangle = \langle X(V_A^{-1} z), V_A^{-1} z\rangle.$$

Since X is positive and $V_A^{-1} z$ is again a carried Hilbert endpoint, the right side is nonnegative. Therefore the amplified output is positive trace-class on $A \otimes H$. As A and X were arbitrary, $\mathrm{Ad}_U^{\mathsf{TC}}$ is completely positive.

For trace preservation, let $T \in \mathsf{TC}(H)$. The cyclic trace row for one trace-class factor and one carried bounded factor gives

$$\begin{aligned}
\mathrm{Tr}_H(\mathrm{Ad}_U^{\mathsf{TC}}(T)) &= \mathrm{Tr}_H(U \circ (T \circ U^{-1})) \\
&= \mathrm{Tr}_H((T \circ U^{-1}) \circ U) \\
&= \mathrm{Tr}_H(T \circ (U^{-1} \circ U)) \\
&= \mathrm{Tr}_H(T \circ \mathrm{id}_H) \\
&= \mathrm{Tr}_H(T).
\end{aligned}$$

The last two equalities use the inverse equation supplied by the unitary automorphism row and the bounded-operator identity law.

The carried linear-map row, complete-positivity row, and trace-preservation row are precisely the clauses of Definition 221.2; hence

$$\mathsf{QChan}_{H,H}(\mathrm{Ad}_U^{\mathsf{TC}})$$

holds. If $\mathsf{Dens}_H(\rho)$, applying Definition 221.3 to this channel gives

$$\mathsf{Dens}_H(\mathrm{Ad}_U^{\mathsf{TC}}(\rho)),$$

which is the asserted density-matrix endpoint preservation. □

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelUnitaryConjugation channel`

Theorem 221.13 (QuantumChannel identity composition unit laws)**.** *Let H and K be* $\mathsf{Hilbert}^{\uparrow}$ *carriers with trace-class spaces* $\mathsf{TC}(H)$ *and* $\mathsf{TC}(K)$*. Let* $\mathrm{id}_{\mathsf{TC}(H)}$ *and* $\mathrm{id}_{\mathsf{TC}(K)}$ *be the identity functions on these trace-class carriers, and write* $\sim_{\mathsf{TC}(K)}$ *for the target trace-class classifier inherited from the* $\mathsf{OperatorIdeal}^{\uparrow}$ *dependency surface. If*

$$\mathsf{QChan}_{H,K}(\Phi),$$

then both unit composites are carried by $\mathsf{QuantumChannel}^{\uparrow}(H,K)$*:*

$$\mathsf{QChan}_{H,K}(\Phi \circ \mathrm{id}_{\mathsf{TC}(H)}) \qquad \textit{and} \qquad \mathsf{QChan}_{H,K}(\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi).$$

Moreover both unit composites are related to Φ by the pointwise $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))$ *classifier:*

$$\Phi \circ \mathrm{id}_{\mathsf{TC}(H)} \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))} \Phi \qquad \textit{and} \qquad \mathrm{id}_{\mathsf{TC}(K)} \circ \Phi \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))} \Phi.$$

Equivalently, every carried trace-class endpoint $T \in \mathsf{TC}(H)$ satisfies

$$(\Phi \circ \mathrm{id}_{\mathsf{TC}(H)})(T) \sim_{\mathsf{TC}(K)} \Phi(T) \qquad \textit{and} \qquad (\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi)(T) \sim_{\mathsf{TC}(K)} \Phi(T).$$

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelSingleton identity composition unit laws`
Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelAffineMixtureSpine unit laws`

Proof. Apply Definition 221.9 to H and to K. This gives

$$\mathsf{QChan}_{H,H}(\mathrm{id}_{\mathsf{TC}(H)}) \qquad \text{and} \qquad \mathsf{QChan}_{K,K}(\mathrm{id}_{\mathsf{TC}(K)}).$$

Applying Definition 221.4 first to $\mathrm{id}_{\mathsf{TC}(H)}$ and Φ gives

$$\mathsf{QChan}_{H,K}(\Phi \circ \mathrm{id}_{\mathsf{TC}(H)}).$$

Applying the same theorem to Φ and $\mathrm{id}_{\mathsf{TC}(K)}$ gives

$$\mathsf{QChan}_{H,K}(\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi).$$

These are the two carried quantum-channel clauses.

It remains to compare the underlying map-code endpoints. Unpack $\mathsf{QChan}_{H,K}(\Phi)$ by Definition 221.2. Its first row carries the $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))$ map-code of Definition 48.1. The trace-class carriers are module carriers over the scalar certificate inherited from the $\mathsf{OperatorIdeal}^{\uparrow}$ and $\mathsf{Hilbert}^{\uparrow}$ dependency surfaces cited in chapter 220. Applying Definition 48.63 with

$$M = \mathsf{TC}(H), \qquad N = \mathsf{TC}(K), \qquad f = \Phi,$$

gives the two pointwise $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K))$ classifier comparisons

$$\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))} \Phi \qquad \text{and} \qquad \Phi \circ \mathrm{id}_{\mathsf{TC}(H)} \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(K))} \Phi.$$

For the displayed pointwise form, let T be a carried trace-class endpoint of $\mathsf{TC}(H)$. The output-closure row of the carried $\mathsf{LinMap}^{\uparrow}$ map-code for Φ gives that $\Phi(T)$ is carried in $\mathsf{TC}(K)$. The identity equations give

$$(\Phi \circ \mathrm{id}_{\mathsf{TC}(H)})(T) = \Phi(T) \qquad \text{and} \qquad (\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi)(T) = \Phi(T).$$

Classifier reflexivity for the target trace-class module therefore gives

$$(\Phi \circ \mathrm{id}_{\mathsf{TC}(H)})(T) \sim_{\mathsf{TC}(K)} \Phi(T) \qquad \text{and} \qquad (\mathrm{id}_{\mathsf{TC}(K)} \circ \Phi)(T) \sim_{\mathsf{TC}(K)} \Phi(T).$$

By Definition 48.49, these pointwise target-classifier comparisons are exactly the underlying $\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H), \mathsf{TC}(K$ classifier relations. □

Theorem 221.14 (Unitary conjugation channel composition law)**.** *Let H be a $\mathsf{Hilbert}^{\uparrow}$ carrier. Let $U, V : H \to H$ be endpoints carried by the $\mathsf{UnitaryGroup}^{\uparrow}(H)$ automorphism carrier, with carried inverses U^{-1} and V^{-1}. Let $W := U \circ V$ be their unitary-group product endpoint, and let $W^{-1} := V^{-1} \circ U^{-1}$ be the inverse endpoint supplied by the product row. Then the composite conjugation map and the product-unitary conjugation map are both carried quantum channels:*

$$\mathsf{QChan}_{H,H}(\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}}) \qquad \textit{and} \qquad \mathsf{QChan}_{H,H}(\mathrm{Ad}_W^{\mathsf{TC}}).$$

Moreover the composite channel endpoint is classified by the product-unitary endpoint:

$$\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}} \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(H))} \mathrm{Ad}_W^{\mathsf{TC}}.$$

Equivalently, writing $\sim_{\mathsf{TC}(H)}$ for the trace-class classifier inherited from the operator-ideal surface, every carried trace-class endpoint $T \in \mathsf{TC}(H)$ satisfies

$$(\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}})(T) \sim_{\mathsf{TC}(H)} \mathrm{Ad}_{U \circ V}^{\mathsf{TC}}(T).$$

Consequently every carried density matrix is sent to carried density-matrix endpoints by both representatives:

$$\mathsf{Dens}_H(\rho) \Longrightarrow \mathsf{Dens}_H((\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}})(\rho)) \wedge \mathsf{Dens}_H(\mathrm{Ad}_W^{\mathsf{TC}}(\rho)).$$

Proof. The $\mathsf{UnitaryGroup}^{\uparrow}(H)$ automorphism carrier cited in section 219.1 is a group of Hilbert automorphisms. Thus its product row carries $W = U \circ V$, and the inverse row identifies the carried inverse of W with $V^{-1} \circ U^{-1}$. Applying Definition 221.12 to U, to V, and to W gives

$$\mathsf{QChan}_{H,H}(\mathrm{Ad}_U^{\mathsf{TC}}), \qquad \mathsf{QChan}_{H,H}(\mathrm{Ad}_V^{\mathsf{TC}}), \qquad \mathsf{QChan}_{H,H}(\mathrm{Ad}_W^{\mathsf{TC}}).$$

Apply Definition 221.4 with $\Phi = \mathrm{Ad}_V^{\mathsf{TC}}$ and $\Psi = \mathrm{Ad}_U^{\mathsf{TC}}$. This gives the carried composite channel

$$\mathsf{QChan}_{H,H}(\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}}).$$

Together with the displayed W clause, this proves the two quantum-channel carrier conclusions.

It remains to identify the endpoints. Let T be a carried endpoint of $\mathsf{TC}(H)$. By Definition 221.11,

$$(\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}})(T) = U \circ (V \circ T \circ V^{-1}) \circ U^{-1}.$$

The bounded-operator composition ring over H from Definition 213.1 is associative on carried endpoints, and Definition 213.5 supplies the trace-class carrier rows for each intermediate conjugation. Hence the displayed endpoint is classified in the trace-class carrier with

$$(U \circ V) \circ T \circ (V^{-1} \circ U^{-1}).$$

The product inverse row gives $W^{-1} = V^{-1} \circ U^{-1}$, while $W = U \circ V$ by definition. Therefore

$$(U \circ V) \circ T \circ (V^{-1} \circ U^{-1}) = W \circ T \circ W^{-1} = \mathrm{Ad}_W^{\mathsf{TC}}(T).$$

Classifier reflexivity and classifier transport in the trace-class operator-ideal carrier give

$$(\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}})(T) \sim_{\mathsf{TC}(H)} \mathrm{Ad}_W^{\mathsf{TC}}(T).$$

Since T was arbitrary, Definition 48.49 packages these pointwise trace-class comparisons, together with the two carried linear-map rows included in the two channel carrier conclusions by Definition 221.2, as

$$\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}} \sim_{\mathsf{LinMap}^{\uparrow}(\mathsf{TC}(H),\mathsf{TC}(H))} \mathrm{Ad}_W^{\mathsf{TC}}.$$

Finally assume $\mathsf{Dens}_H(\rho)$. Applying Definition 221.3 to the composite channel gives

$$\mathsf{Dens}_H((\mathrm{Ad}_U^{\mathsf{TC}} \circ \mathrm{Ad}_V^{\mathsf{TC}})(\rho)).$$

Applying the same lemma to $\mathsf{QChan}_{H,H}(\mathrm{Ad}_W^{\mathsf{TC}})$ gives

$$\mathsf{Dens}_H(\mathrm{Ad}_W^{\mathsf{TC}}(\rho)).$$

These are the asserted density-matrix endpoint clauses. □

Lean4 ✓ `BEDC.Derived.QuantumChannelUp.QuantumChannelUnitaryConjugation composition law`

Closure status for $\mathsf{NameCert}_{\mathsf{QuantumChannel}^{\uparrow}}$.

Constructive story: A quantum channel from a carried Hilbert source H to a carried Hilbert target K is built by taking a $\mathsf{LinMap}^{\uparrow}$ code between the two trace-class carriers and adding the two CPTP rows: complete positivity under every finite ancilla and trace preservation on every trace-class endpoint. The carrier accepts only maps with that linear row, complete-positivity row, and trace-preservation row, while the classifier compares channels pointwise through the trace-class and linear-map classifiers and keeps the same density-matrix image, composition, affine-mixture, identity, and unitary-conjugation ledgers. The source therefore covers trace-class linear maps between $\mathsf{DensityMatrix}^{\uparrow}$ surfaces, the pattern records density-matrix image closure, composition closure and associativity, finite convex mixture closure, identity-channel unit laws, and unitary conjugation, and the ledger keeps the underlying $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, and $\mathsf{UnitaryGroup}^{\uparrow}$ rows visible. Since the dependency classifiers and pointwise trace-class comparisons are equivalence relations and CPTP rows are stable under composition, convex combination, identity, and unitary conjugation as displayed, an admitted channel supplies the carrier witness, the classifier is reflexive, symmetric, and transitive, and classified replacements remain carried channels. A map row h lies in the signature gap of a $\mathsf{QuantumChannel}^{\uparrow}$ token p when the source-domain accepts its linear trace-class map and CPTP ledger, the signature relation pairs it with an answer channel and evidence ledger, and token introduction names that answer as p; unfolded, h sends every carried density matrix to the carried density matrix and channel behavior encoded by p. No standard quantum-information bridge is exported here. The chapter registers this CPTP-map construction under the name $\mathsf{QuantumChannel}^{\uparrow}$; a channel is a displayed linear trace-class map with positivity and trace ledgers, with no Stinespring dilation, external channel category, or host quantum-information primitive introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuantumChannel}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The obligation scope displayed in this chapter consists of the CPTP carrier, density-matrix image, composition closure and associativity, finite convex mixture closure, finite density-mixture preservation, identity channel, identity unit laws, unitary conjugation channel, and unitary-conjugation composition law over the $\mathsf{DensityMatrix}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, and BHist surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuantumChannel}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.QuantumChannelUp.QuantumChannelUnitaryConjugation\_composition\_law`

Bridge status: none

Not claimed: This chapter does not claim Stinespring dilation, full quantum information theory, a standard quantum-information bridge, or bridge verification.

Upgrade path: No further paper-axis upgrade is required for the mature $\mathsf{QuantumChannel}^{\uparrow}$ package displayed here.

Chapter 222

A Concrete Naming Certificate for $\mathsf{Entanglement}^{\uparrow}$

The entanglement interface $\mathsf{Entanglement}^{\uparrow}$ packages the structure obstruction of a bipartite quantum state to factor as a tensor product of pure states, certified via Schmidt rank or entanglement entropy. It depends on $\mathsf{NameCert}_{\mathsf{QuantumState}^{\uparrow}}$ (chapter 217) and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87).

222.1 Carrier definition

Definition 222.1 (EntanglementUp BHist source surface)**.** *An* $\mathsf{Entanglement}^{\uparrow}$ *BHist source surface consists of a* $\mathsf{QuantumState}^{\uparrow}$ *row for the displayed bipartite state,* $\mathsf{TensorProduct}^{\uparrow}$ *factor rows for the two displayed factors, a bipartite state endpoint, a factorization-obstruction witness for that endpoint,* hsame *transport rows for the state and factor endpoints, and* Pkg *provenance for all displayed rows. The obstruction row is part of the certified source surface; it is not a host-level negation over all possible decompositions.*

Lean4 (definition) `BEDC.Derived.EntanglementUp.EntanglementSourceSurface`

Theorem 222.2 (Entanglement source factor boundary)**.** *Consumers of the* $\mathsf{Entanglement}^{\uparrow}$ *source surface may read only the two displayed factor rows, the bipartite endpoint row, the factorization-obstruction witness, the* hsame *transport rows, and the* Pkg *provenance from Definition 222.1. No Schmidt-rank completeness theorem, entropy criterion, or classification of all bipartite states is included in this boundary.*

Lean4 ✓ `BEDC.Derived.EntanglementUp.EntanglementSourceSurface factor boundary`

Proof. Unpack the source surface as a finite BHist package. The $\mathsf{QuantumState}^{\uparrow}$ row supplies the carried bipartite endpoint, and the two $\mathsf{TensorProduct}^{\uparrow}$ rows name the displayed factor histories read by downstream tensor consumers.

Project the obstruction row and the hsame transport rows from the same package. Repacking these projections through Pkg gives exactly the visible consumer boundary, while completeness principles about Schmidt rank or entropy would require rows absent from the displayed source surface. □

Lean4 ✓ `BEDC.Derived.EntanglementUp.EntanglementBHistSourceSurface factor boundary`

Theorem 222.3 (Entanglement obstruction transport stability)**.** *Let two* Entanglement$^{\uparrow}$ *BHist source surfaces have* hsame*-related bipartite endpoints,* hsame*-related factor endpoints, the same displayed factorization-obstruction row transported through the package, and the same* Pkg *provenance. Then consumers of the two surfaces read the same obstruction boundary.*

The stable boundary is only the displayed obstruction row together with its transport and provenance data. It does not classify Schmidt rank, compute entanglement entropy, or decide completeness of all tensor-product factorizations.

Lean4 ✓ `BEDC.Derived.EntanglementUp.EntanglementBHistSourceSurface obstruction transport stability`

Proof. Unpack the two source surfaces using Definition 222.1. Their visible data consist of the bipartite endpoint row, the two factor rows, the obstruction row, hsame transports for the state and factor endpoints, and Pkg provenance.

Use the assumed hsame rows to transport the bipartite and factor endpoints inside the same package provenance. Because the displayed obstruction row is also transported through that package, the consumer sees one obstruction boundary rather than two independent host-level negations.

Reapply Definition 222.2. That boundary permits consumers to read exactly the transported obstruction, endpoint, factor, transport, and provenance rows. Since Schmidt-rank completeness and entropy classification are absent from the source surface, the stability claim does not depend on them. □

222.2 The certificate

Theorem 222.4 (EntanglementUp NameCert obligation surface)**.** *The* Entanglement$^{\uparrow}$ *NameCert obligation surface for Definition 222.1 is the package containing its* QuantumState$^{\uparrow}$ *endpoint row, the two* TensorProduct$^{\uparrow}$ *factor rows, the factorization-obstruction row, the* hsame *transport rows, and the* Pkg *provenance. Certificate consumers may demand those rows and no stronger entanglement invariant.*

Lean4 ✓ `BEDC.Derived.EntanglementUp.EntanglementBHistSourceSurface namecert obligation surface`

Proof. Start from the source factor boundary of Definition 222.2. It already separates the state endpoint, factor rows, obstruction witness, transport rows, and provenance rows.

Repack those rows into the NameCert fields: carrier from the state endpoint, dependencies from QuantumState$^{\uparrow}$ and TensorProduct$^{\uparrow}$, stability from the hsame transports, and ledger provenance from Pkg. Since no row supplies a complete Schmidt decomposition or entropy comparison, the certificate closes only this displayed obligation surface. □

Closure status for NameCert$_{\text{Entanglement}^{\uparrow}}$.

Constructive story: An Entanglement$^{\uparrow}$ packet is generated from a QuantumStateUp bipartite endpoint row, TensorProductUp factor endpoint rows, a displayed factorization-obstruction witness, componentwise hsame transport for state and factors, finite route rows, Pkg provenance, and local naming. The classifier reads entanglement as an obstruction surface carried by the same finite BHist packet: source factors are exposed, the obstruction transports along the displayed state and factor rows, and the NameCert surface repacks carrier, classifier, stability, and ledger provenance. The ledger story is quantum-structural but finite: no complete Schmidt decomposition, entropy comparison, Hilbert-space quotient, or host tensor equality is added beyond the displayed obstruction route.

Theory closure: TheoryGate(NameCert$_{\text{Entanglement}^{\uparrow}}$, seed)

Scope closed: This chapter binds the BHist source-surface definition and the source-factor-boundary, obstruction-transport-stability, and NameCert-obligation theorems for Entanglement$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{Entanglement}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim complete Schmidt decompositions, entropy comparisons, host tensor-factor equality, or quantum-state data beyond the displayed source and obstruction rows.

Upgrade path: To reach obligationClosure, the obstruction carrier and transport-stability theorem must be assembled into a checked packet-level closure statement.

Chapter 223

A Concrete Naming Certificate for $\mathsf{NewtonIteration}^{\uparrow}$

The Newton-iteration interface $\mathsf{NewtonIteration}^{\uparrow}$ packages the iterative scheme $x_{n+1} = x_n - F(x_n)/F'(x_n)$ for finding zeros of a differentiable map, with quadratic convergence on a contraction neighbourhood. It depends on $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ (chapter 122) and $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90).

223.1 Carrier definition

Definition 223.1 (Newton-iteration finite step carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *derivative source row and a* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *normed-source row for the same displayed point context. A* $\mathsf{NewtonIteration}^{\uparrow}$ *carrier row is the finite* BHist *ledger recording the source point answer, the derivative answer row at that point, an inverse-row witness for the displayed derivative answer, the next-step answer row, and the* Cont *witnesses that place the step computation inside the derivative and Banach source ledgers.*

The carrier records one finite Newton step or a finite list of such displayed steps. It does not assert convergence, completeness of the ambient space, a contraction theorem, or a radius of validity.

Lean4 (definition) `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier`

Definition 223.2 (Newton-iteration contraction classifier)**.** *Two rows of Definition 223.1 are classified as the same* $\mathsf{NewtonIteration}^{\uparrow}$ *step exactly when they have the same source point up to* hsame*, the derivative rows are transported through* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$*, the inverse-row witnesses are matched as visible answer histories, and the next-step answer rows are related by* hsame *inside the same* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *source context.*

The classifier is generated by derivative-source transport, Banach-source transport, inverse-row transport, next-step history transport, and finite concatenation of displayed step rows. It contains no convergence or completeness generator.

Lean4 (definition) `BEDC.Derived.NewtonIterationUp.NewtonIterationContractionClassifier`

223.2 The certificate

Theorem 223.3 (Newton-iteration Banach and derivative source scope)**.** *Every* $\mathsf{NewtonIteration}^{\uparrow}$ *carrier row has exactly two source certificates: the derivative source row from* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$

and the normed-source row from $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$*. The point, derivative, inverse-row, and next-step data are finite answer histories attached to those sources by* hsame*,* Cont*, and* Pkg *witnesses.*

Proof. Unpack Definition 223.1. The derivative and Banach certificates are explicit source fields, while the source point, derivative answer, inverse-row witness, and next-step answer are the finite displayed step ledger.

The classifier of Definition 223.2 can transport only those fields. Its source moves use the two named NameCert classifiers, and its computational moves use the displayed hsame, Cont, and Pkg witnesses, so no convergence or completeness datum enters the scope. □

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier banach derivative source scope`

Theorem 223.4 (Newton-iteration step stability)**.** *Classified* $\mathsf{NewtonIteration}^{\uparrow}$ *step rows remain classified after transporting the derivative source through* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$*, transporting the Banach source through* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$*, and replacing the visible source point, inverse-row witness, or next-step answer by an* hsame*-related row.*

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier step endpoint transport` ↪ variant `BEDC.Derived.NewtonIterationUp.NewtonIterationStep stability`

Proof. First transport the source certificates. The derivative source gives the derivative-row comparison, and the Banach source gives the point and next-step context in which the row is read.

Next transport the displayed step data. The source point and next-step rows use hsame endpoint replacement; the inverse-row witness is carried as a visible answer history; and Cont repacks the transported fields into the same finite computation ledger. The generated classifier then supplies the classified step row. □

Theorem 223.5 (Newton-iteration obligation package)**.** *The* $\mathsf{NewtonIteration}^{\uparrow}$ *chapter exports a finite obligation package consisting of the step carrier, the contraction classifier, the derivative and Banach source certificates, and ledger exactness for the displayed Newton step row. It does not claim convergence or completeness.*

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier semantic name certificate`
Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationCarrier obligation package`

Proof. Build the package from a carried step row and a classifier witness. The carrier supplies the source point, derivative answer, inverse-row witness, next-step answer, and the Cont rows placing them under the two source certificates.

Ledger exactness follows by inverting the carrier and classifier constructors. Every accepted datum is one of the two source certificates, one of the finite step answer histories, or one of the named hsame, Cont, Pkg, or NameCert transport witnesses. Hence the package exposes only the displayed finite Newton step obligation. □

Theorem 223.6 (Newton-iteration scoped kernel dependency boundary)**.** *The* $\mathsf{NewtonIteration}^{\uparrow}$ *obligation package is scoped exactly by the* $\mathsf{Derivative}^{\uparrow}$ *source row, the* $\mathsf{Banach}^{\uparrow}$ *source row, the finite* BHist *step ledger, the inverse-row witness, the next-step answer row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *transport witnesses.*

Equivalently, every obligation row exported by Definition 223.5 is read from the carrier of Definition 223.1, classified by Definition 223.2, and bounded by the source-scope and stability rows of Definition 223.3 and Definition 223.4. Convergence, completeness, radius data, and host Newton data are outside this carrier.

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier scoped kernel dependency boundary`

Proof. Start with an exported obligation row. By Definition 223.5, it is either one of the two source certificates, one of the finite step answer histories, or one of the displayed transport witnesses used to read that finite step.

Apply Definition 223.1 to identify the only carrier fields: the derivative source row, the Banach source row, the source point answer, the derivative answer, the inverse-row witness, the next-step answer, and the finite BHist ledger connecting those rows by Cont. Apply Definition 223.2 to identify the only classifier moves: source transport, inverse-row transport, next-step transport, and finite concatenation of displayed step rows.

The source boundary is then fixed by Definition 223.3; there are no further source certificates beyond $\mathsf{Derivative}^{\uparrow}$ and $\mathsf{Banach}^{\uparrow}$. The admissible replacements are fixed by Definition 223.4; they use only hsame, Cont, Pkg, and NameCert transport witnesses already visible in the carrier.

Therefore the dependency boundary is exactly the named kernel surface. Any convergence theorem, ambient completeness field, contraction radius, or host-level Newton recurrence would require data not present in the carrier or classifier, so those rows are not part of the scoped obligation package. □

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonIteration}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NewtonIteration}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridged scope is the finite step ledger over $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$, the inverse-row witness, next-step history, BHist rows, hsame, Cont, Pkg, displayed source certificate rows, and the semantic name-certificate package.

Constructive story: A $\mathsf{NewtonIteration}^{\uparrow}$ packet is built by choosing the displayed derivative and Banach source certificates, reading one finite Newton step through the point, derivative, inverse-row, and next-step histories, and repacking those rows through the visible hsame, Cont, Pkg, and NameCert witnesses.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NewtonIteration}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not claim convergence, ambient completeness, a contraction theorem, or a radius of validity.

Upgrade path: No further paper-side theory-axis upgrade remains for the finite Newton-step packet.

Theorem 223.7 (Newton-iteration finite-step concatenation closure)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *derivative source row D and a* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *normed-source row B for the same displayed point context. Let R_1 and R_2 be accepted* $\mathsf{NewtonIteration}^{\uparrow}$ *carrier rows of Definition 223.1 over this same source pair. Write the source and next-step answer rows of R_1 as x_0 and x_1, and those of R_2 as y_1 and x_2. Let Λ_1 and Λ_2 be their displayed finite step ledgers, including the derivative-answer rows, inverse-row witnesses, and the* Cont *witnesses placing each displayed Newton step inside D and B.*

If $\mathsf{hsame}(x_1, y_1)$ *and* $\Lambda_{12} : \mathsf{BHist}$ *is displayed with*

$$\mathsf{Cont}(\Lambda_1, \Lambda_2, \Lambda_{12}),$$

then the displayed concatenated row R_{12} over D and B is an accepted $\mathsf{NewtonIteration}^{\uparrow}$ *carrier row. Its source-point answer is x_0, its next-step answer is x_2, and its finite step ledger is exactly the concatenated ledger Λ_{12}. The derivative-answer and inverse-witness subledgers of R_{12} are the visible finite lists obtained by keeping the R_1 subledgers before the R_2 subledgers after transporting the splice endpoint along* $\mathsf{hsame}(x_1, y_1)$.

Proof. Unpack the carrier of Definition 223.1. The row R_1 supplies a finite displayed Newton-step ledger from x_0 to x_1 under the derivative source D and the Banach source B. The row R_2 supplies a finite displayed Newton-step ledger from y_1 to x_2 under the same two sources. Both ledgers are finite BHist rows, and their derivative-answer rows, inverse-row witnesses, next-step rows, and source-containment witnesses are already among the accepted carrier fields.

Use the splice comparison $\mathsf{hsame}(x_1, y_1)$ to read the source endpoint of R_2 as the next-step endpoint of R_1. This is exactly the endpoint replacement allowed by Definition 223.4: the derivative and Banach source certificates remain D and B, while the visible source point of the second row is transported through the displayed hsame witness. No analytic Newton equation, convergence datum, completeness datum, or radius condition is introduced by this transport.

Now concatenate the two finite ledgers by the displayed continuation row $\mathsf{Cont}(\Lambda_1, \Lambda_2, \Lambda_{12})$. Since Cont is the finite continuation relation on BHist, the result is again a finite BHist ledger. The first segment records exactly the step data already accepted in R_1, and the second segment records exactly the transported step data already accepted in R_2. At the splice, the outgoing answer of the first segment and the incoming answer of the second segment are identified by the given $\mathsf{hsame}(x_1, y_1)$ witness, so the displayed list has a single source endpoint x_0 and a single final next-step endpoint x_2.

The internal Cont witnesses remain valid after concatenation. Where a witness is wholly inside one segment, it is inherited from the corresponding accepted carrier row. Where a witness crosses the displayed grouping of the two ledgers, Definition 4.56 identifies the two possible groupings up to hsame, and continuation result transport repacks the same finite history as Λ_{12}. Thus every derivative-source and Banach-source containment row required by the carrier definition is present in the concatenated display.

Finally apply the finite-concatenation generator in the classifier of Definition 223.2. It classifies the two accepted displayed step rows, after the splice transport, as one accepted finite Newton-iteration row. Repacking these fields gives R_{12} with source point x_0, next-step answer x_2, and finite step ledger Λ_{12}, as claimed. □

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier finite step concatenation closure`

↪ variant `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier finite step ledger concat closure`

Theorem 223.8 (Newton-iteration splice source determinacy)**.** *Let R_{12} and R'_{12} be accepted concatenated* $\mathsf{NewtonIteration}^{\uparrow}$ *carrier rows over the same* $\mathsf{NameCert}_{\mathsf{Derivative}^{\uparrow}}$ *source row D, the same* $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ *source row B, the same initial endpoint x_0, the same final endpoint x_2, and the same displayed splice witness $\sigma : \mathsf{hsame}(x_1, y_1)$. Suppose their concatenation ledgers Λ_{12} and Λ'_{12} are related by* hsame *and arise from displayed continuation rows*

$$\mathsf{Cont}(\Lambda_1, \Lambda_2, \Lambda_{12}) \qquad \textit{and} \qquad \mathsf{Cont}(\Lambda'_1, \Lambda'_2, \Lambda'_{12})$$

whose two segments are accepted Newton step ledgers over D and B.

Then R_{12} and R'_{12} have the same source-boundary projection. Their derivative source is D, their Banach source is B, their inverse-row subledger lists are read as the same displayed finite lists up to the hsame *transport of the concatenation ledgers, their next-step endpoint is x_2, and their finite ledger provenance is classified by the Newton-iteration contraction classifier of Definition 223.2.*

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier splice source determinacy`

Proof. Apply Definition 223.7 to both displayed concatenations. Each accepted concatenated row is therefore obtained by repacking the same derivative source D, the same Banach source B, the

same initial endpoint x_0, the same final endpoint x_2, and the same splice comparison σ into the carrier of Definition 223.1.

The carrier definition exposes no hidden source boundary. Its source projection consists only of the derivative source, the Banach source, the inverse-row answer histories, the next-step endpoint, and the finite Cont provenance attached to those rows. Thus two rows with the same displayed sources and endpoints can differ at the source boundary only by the listed ledger rows.

Use the hypothesis $\mathsf{hsame}(\Lambda_{12}, \Lambda'_{12})$ to transport the continuation result. The continuation result transport repacks the Cont witnesses for the two displayed concatenations into the same finite history boundary, and Definition 223.4 transports the segment data through the shared splice witness σ.

Finally read the transported rows through the classifier of Definition 223.2. Its generators identify derivative-source transport, Banach-source transport, inverse-row transport, next-step history transport, and finite concatenation, so the derivative source, Banach source, inverse-row subledger list, next-step endpoint, and finite ledger provenance have the same classified source-boundary projection.

No convergence datum, completeness datum, contraction radius, or analytic Newton theorem is introduced in this comparison. The statement is only a determinacy result for the finite BHist source boundary already present in the accepted carrier rows. □

Theorem 223.9 (Newton-iteration finite-prefix restriction carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Derivative}^\uparrow}$ *derivative source row D and a* $\mathsf{NameCert}_{\mathsf{Banach}^\uparrow}$ *normed-source row B for the same displayed point context. Let R_{n+k} be an accepted* $\mathsf{NewtonIteration}^\uparrow$ *carrier row of Definition 223.1 over D and B whose finite Newton ledger $\Lambda_{n+k} : \mathsf{BHist}$ is displayed as a prefix ledger followed by a tail ledger:*

$$\mathsf{Cont}(\Lambda_n, \Lambda_k, \Lambda_{n+k}).$$

Assume that the displayed prefix ledger Λ_n contains exactly the first n source-point rows, next-step rows, function-value rows, derivative-answer rows, inverse-row witnesses, and derivative-nonzero denominator witnesses read from R_{n+k}, with endpoint rows x_0 and x_n. Then the row R_n obtained by keeping D, B, x_0, x_n, and the displayed prefix subledgers is an accepted $\mathsf{NewtonIteration}^\uparrow$ *carrier row. In particular, finite restriction to the first n Newton recurrence steps preserves the iteration rows, the function-value rows, and the derivative-nonzero ledger without adding convergence, completeness, a contraction theorem, or a radius datum.*

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier finite prefix restriction`

Proof. Unpack the accepted carrier row R_{n+k} using Definition 223.1. Its public data are the two source certificates D and B, the visible source-point and next-step histories, the derivative answer rows, the inverse-row witnesses for the displayed derivative answers, and the Cont witnesses placing the displayed Newton computations inside the derivative and Banach source ledgers. The accepted row contains no analytic convergence field and no ambient completeness field.

The displayed continuation

$$\mathsf{Cont}(\Lambda_n, \Lambda_k, \Lambda_{n+k})$$

exhibits Λ_n as a finite initial subledger of the accepted finite ledger. Since Cont is generated only by the finite right-recursive continuation rules of Definition 4.10, every row of Λ_n is one of the visible rows already appearing in Λ_{n+k} before the tail Λ_k is read. Thus the first n source-point rows, next-step rows, function-value rows, derivative-answer rows, inverse-row witnesses, and derivative-nonzero denominator witnesses are inherited as finite BHist rows from the accepted packet.

The source certificates do not change under restriction. The derivative rows in the prefix are still scoped by D, because they are the same derivative-answer and nonzero-denominator rows that R_{n+k} placed under the derivative source. The point and next-step rows in the prefix are still scoped by B, because the Banach source containment witnesses are among the same retained Cont rows. If an endpoint of the retained prefix is represented by an hsame-transported history, Definition 223.4 repacks the same visible source point, inverse-row witness, and next-step answer without changing the two source certificates.

It remains only to check that the retained ledger is a Newton-iteration carrier ledger rather than an arbitrary subhistory. By the hypothesis on Λ_n, each retained step is one of the first n displayed Newton recurrence steps from R_{n+k}: it carries its source point, function value, derivative answer, inverse-row witness, nonzero-denominator witness, and next-step answer. The containment witnesses for those rows are inherited from the longer carrier, and no witness from the discarded tail Λ_k is required to read a retained recurrence step. Repacking these retained fields gives exactly the carrier row described in Definition 223.1 over the same pair D, B.

The classifier of Definition 223.2 permits only source transport, inverse-row transport, next-step history transport, and finite concatenation of displayed step rows. Restricting to the first displayed segment uses none of the tail-specific data and introduces no additional generator. Therefore the prefix row R_n is accepted as a $\mathsf{NewtonIteration}^{\uparrow}$ carrier row, with finite ledger Λ_n and endpoints x_0, x_n. □

Lean4 ✓ `BEDC.Derived.NewtonIterationUp.NewtonIterationBHistCarrier finite prefix restriction carrier`

Chapter 224

A Concrete Naming Certificate for $\mathsf{Interpolation}^{\uparrow}$

The polynomial-interpolation interface $\mathsf{Interpolation}^{\uparrow}$ packages, for a finite set of selected nodes and target values, the polynomial row whose displayed evaluations hit those targets. It depends on $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

224.1 Carrier definition

Definition 224.1 (Interpolation BHist node carrier)**.** *An* $\mathsf{Interpolation}^{\uparrow}$ *carrier row is a finite* BHist *package consisting of:*

1. *a* $\mathsf{FinSet}^{\uparrow}$ *finite-spine membership ledger selecting the interpolation nodes;*
2. *for every selected node, a carried target-value row with its* BHist *endpoint;*
3. *a* $\mathsf{Poly}^{\uparrow}$ *candidate row whose coefficient spine is carried by the polynomial certificate;*
4. *for every selected node, a polynomial-evaluation row comparing the candidate row at that node with the displayed target value; and*
5. *a sample ledger recording the* hsame*,* Cont*, and* Pkg *provenance used by those membership, target-value, polynomial, and evaluation rows.*

The carrier is the finite package itself; it is not a host polynomial function, a quotient representative, a field-completeness witness, or a uniqueness claim over an external field.

Theorem 224.2 (Interpolation sample ledger surface)**.** *Every accepted sample in the carrier of Definition 224.1 is readable from only the* $\mathsf{FinSet}^{\uparrow}$ *membership row for the selected node, the* $\mathsf{Poly}^{\uparrow}$ *evaluation row at that node, the* BHist *endpoints for the node, target value, and polynomial candidate, the displayed* hsame *comparisons, the displayed* Cont *routes, and the displayed* Pkg *provenance.*

No accepted sample exposes a host evaluation function, a quotient polynomial class, field completeness, a global interpolation algorithm, or uniqueness over an external field.

Lean4 ✓ `BEDC.Derived.InterpolationUp.InterpolationSampleLedger surface`

Proof. Unpack the carrier row of Definition 224.1. A sample is accepted only when its node occurs in the displayed finite-spine membership ledger and when the candidate polynomial has the displayed evaluation row at that same node.

Project the two dependency certificates. The finite-set side contributes only membership in the selected finite spine, while the polynomial side contributes only the named evaluation row and the coefficient-spine carrier row already present in the package.

The remaining data are bookkeeping rows: the node endpoint, target-value endpoint, candidate-polynomial endpoint, hsame comparisons, Cont routes, and Pkg provenance. Since the carrier contains no other constructor, no host evaluation function, quotient row, completeness witness, global algorithm, or external uniqueness field can be read from an accepted sample. □

Theorem 224.3 (Interpolation evaluation stability)**.** *Let two interpolation carrier rows share the same* $\mathsf{FinSet}^{\uparrow}$ *selected-node spine and the same* $\mathsf{Poly}^{\uparrow}$ *candidate surface up to displayed* hsame *transport. If each selected node endpoint, target-value endpoint, and polynomial endpoint row is transported by the displayed* hsame *comparisons and the corresponding* Cont/Pkg *routes, then the interpolation evaluation ledger is preserved sample by sample.*

Equivalently, every transported selected node reads the same membership row and the same target-hit evaluation row through the transported polynomial endpoint.

Lean4 ✓ `BEDC.Derived.InterpolationUp.InterpolationEvaluationLedger sample transport`

Proof. Fix an accepted sample in the first carrier row. By Definition 224.2, the sample is completely described by finite-set membership, a polynomial evaluation row, the three visible endpoints, and the displayed transport provenance.

Transport the selected node through the finite-set classifier rows. The membership ledger is stable because the node endpoint is carried along the displayed hsame comparison and the finite-spine row is the same selected surface.

Transport the candidate polynomial and target value through their endpoint comparisons. The $\mathsf{Poly}^{\uparrow}$ evaluation row is stable under the displayed polynomial endpoint transport and target-value transport, so the target-hit comparison is rebuilt at the transported sample.

The converse transport uses the same rows in the opposite direction, using symmetry of the displayed hsame comparisons and the reverse listed provenance routes. Hence the evaluation ledger is preserved sample by sample. □

Theorem 224.4 (Interpolation NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Interpolation}^{\uparrow}}$ *surface consists exactly of the carrier row of Definition 224.1, the classifier that compares two packages by the same selected-node membership rows, candidate polynomial rows, target-value rows, sample ledgers, and evaluation rows, the stability row of Definition 224.3, the sample ledger surface of Definition 224.2, and the exactness rows saying that every accepted package has all displayed selected samples and every displayed selected sample is read from those rows.*

This is a finite BHist *obligation surface over* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.InterpolationUp.InterpolationNameCert obligation surface`

Proof. Start with the carrier row. Its fields list exactly the finite selected-node spine, target values, candidate polynomial row, sample ledger, and evaluation rows, so it supplies the source predicate for the local certificate.

Define the classifier by fieldwise comparison of those visible rows. Reflexive, symmetric, and transitive classifier witnesses are obtained by the corresponding hsame ledger laws on endpoints and by the already displayed classifier rows from the $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Poly}^{\uparrow}$ dependency certificates.

The stability obligation is precisely Definition 224.3. The ledger and exactness obligations are precisely Definition 224.2 together with the carrier definition: an accepted package contains no

sample except one read through the finite membership and evaluation rows, and each listed sample is accepted by those rows. □

Theorem 224.5 (Interpolation downstream threshold package)**.** Poly$^{\uparrow}$ *and* FinSet$^{\uparrow}$ *consumers may cite the interpolation carrier, classifier, stability, sample-ledger, and exactness rows of Definition 224.4. Such a citation gives only the finite* BHist *package of selected nodes, target values, candidate polynomial rows, evaluation rows, and visible* hsame/Cont/Pkg *provenance.*

It gives no host polynomial function, no quotient representative, no field completeness principle, and no uniqueness statement over external fields.

```
Lean4 ✓  BEDC.Derived.InterpolationUp.InterpolationDownstreamThresholdPackage boundary
  ↪ variant  BEDC.Derived.InterpolationUp.InterpolationDownstreamThresholdPackage threshold surface
  ↪ variant  BEDC.Derived.InterpolationUp.InterpolationDownstreamThreshold surface
```

Proof. Unpack the obligation surface from Definition 224.4. The rows available to a downstream consumer are the five displayed certificate rows: carrier, classifier, stability, sample ledger, and exactness.

Project those rows to a Poly$^{\uparrow}$ consumer. The consumer sees the candidate polynomial endpoint and the evaluation rows already carried by the polynomial certificate, together with their hsame/Cont/Pkg provenance.

Project the same rows to a FinSet$^{\uparrow}$ consumer. The consumer sees the selected finite-spine membership ledger and the node endpoints, again with only the displayed provenance rows.

Since neither projection adds a constructor to the interpolation carrier, the consumer receives no host function, quotient representative, completeness principle, or external-field uniqueness statement. The package is exactly the finite BHist threshold surface stated above. □

224.2 The certificate

Closure status for NameCert$_{\text{Interpolation}^{\uparrow}}$**.**

Constructive story: InterpolationUp is a finite selected-node spine over FinSet$^{\uparrow}$, target-value rows, a Poly$^{\uparrow}$ candidate row, hsame, Cont, and Pkg provenance. The displayed sample, evaluation, NameCert, and downstream-threshold rows bind that spine to BEDC histories only, so the chapter exports no host polynomial function and no field-completeness claim.

Theory closure: TheoryGate(NameCert$_{\text{Interpolation}^{\uparrow}}$, publicClosed)

Scope closed: The visible scope is the finite selected-node spine, target-value rows, Poly$^{\uparrow}$ candidate row, FinSet$^{\uparrow}$ membership ledger, hsame, Cont, and Pkg provenance recorded by the local carrier and sample ledger.

Formal verification: FormalStatus(NameCert$_{\text{Interpolation}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InterpolationUp.InterpolationNameCert\_obligation\_surface`

Bridge status: none

Not claimed: The chapter does not close host polynomial functions, quotient representatives, field completeness, global interpolation algorithms, or external uniqueness.

Upgrade path: The next bridge obligation is a checked Interpolation$^{\uparrow}$ standard bridge over the same finite-node and polynomial-evaluation rows.

Theorem 224.6 (Interpolation empty selection has empty evaluation ledger)**.** *Let I be an* Interpolation$^{\uparrow}$ *carrier row in the sense of Definition 224.1. Suppose that the displayed* FinSet$^{\uparrow}$ *selected-node membership ledger of I has no entries: no history n :* BHist *occurs as a selected node in that finite-spine ledger. Then I has no accepted sample. Consequently, its polynomial-evaluation ledger has no sample-indexed row, and the sample-ledger surface of I is empty.*

Proof. Assume, toward contradiction, that I has an accepted sample at a node n. By Definition 224.2, every accepted sample is read through the $\mathsf{FinSet}^{\uparrow}$ membership row for its selected node and the $\mathsf{Poly}^{\uparrow}$ evaluation row at that same node. In particular, the accepted sample exposes a selected-node membership entry for n in the displayed finite-spine ledger of I.

This contradicts the empty-ledger hypothesis. Hence no accepted sample exists. The only polynomial-evaluation rows admitted as sample rows by Definition 224.1 are the rows indexed by selected nodes and read together with the corresponding membership rows. Since there is no selected-node membership entry, there is no sample index from which such an evaluation row can be read. Therefore the polynomial-evaluation ledger contains no sample-indexed row, and the sample-ledger surface is empty. □

Lean4 ✓ `BEDC.Derived.InterpolationUp.InterpolationBHistNodeCarrier empty selection empty evaluation ledger`

Chapter 225

A Concrete Naming Certificate for $\mathsf{Approximation}^{\uparrow}$

The best-approximation interface $\mathsf{Approximation}^{\uparrow}$ packages, for a continuous function on a compact carrier, the polynomial of bounded degree minimising the supremum-norm error (Weierstrass / Chebyshev). It depends on $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ (chapter 130) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

225.1 Carrier definition

Theorem 225.1 (ApproximationUp carrier obligation surface)**.** *The* $\mathsf{Approximation}^{\uparrow}$ *carrier surface is a* BHist *packet whose continuous-source row is certified by* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, whose polynomial-candidate row is certified by* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*, whose comparison row is* hsame*, whose transport rows are* Cont *ledgers, and whose finite error ledger records the displayed approximation error data. The carrier does not expose a supremum object or any choice data.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationFiniteErrorLedger carrier surface`

Proof. Read the carrier through the ten-field roadmap template:

1. source: the BHist input carrying the compact continuous-map source;
2. carrier: the accepted continuous endpoint supplied by $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$;
3. pattern: the polynomial candidate supplied by $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$;
4. classifier: componentwise hsame between source, candidate, and error endpoints;
5. stability: Cont transport for every exposed endpoint;
6. ledger: the finite error ledger;
7. exactness: each error row is attached to one displayed candidate and one displayed source;
8. consumer: polynomial approximation consumers read only these rows;
9. boundary: no supremum witness is exported;
10. provenance: no choice data is added beyond the two cited NameCert sources.

These ten rows are all carried by named histories and finite ledgers. Therefore the carrier surface is a BEDC packet rather than an external analytic minimisation object. □

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket obligation surface`

Theorem 225.2 (ApproximationUp polynomial consumer window)**.** *Every* $\mathsf{Approximation}^{\uparrow}$ *polynomial consumer reads a* BHist *source through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, a candidate row through* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*,* hsame *comparison rows,* Cont *transport rows, and the finite error ledger. Its permitted window is the finite candidate, degree, endpoint, and error data displayed by the packet, with no supremum row and no choice data.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket polynomial consumer window`

Proof. The consumer first projects the continuous map from $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ and the candidate polynomial from $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$. Both projections remain histories in the same BHist packet.

It then reads the hsame classifier and the Cont transport ledgers. These rows allow the candidate and error endpoints to move together without replacing the source certificate.

Finally the consumer reads the finite error ledger. Since the ledger is finite, the consumer can inspect recorded error rows directly; no supremum witness, selection functional, or choice data is part of the exposed window. □

225.2 The certificate

Theorem 225.3 (ApproximationUp compact continuous source boundary)**.** *The* $\mathsf{Approximation}^{\uparrow}$ *source boundary for a compact continuous input is exactly the* BHist *packet combining* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*,* Cont *transport,* hsame *comparison, and the finite error ledger. Compactness and continuity are consumed only through the continuous-map certificate row, and the boundary exports no supremum or choice data.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket compact continuous source boundary`

Proof. Use the ten-field roadmap template with the continuous certificate as the source side: source BHist, continuous carrier, polynomial pattern, hsame classifier, Cont stability, finite error ledger, endpoint exactness, polynomial consumer, no-supremum boundary, and no-choice provenance.

The compact continuous hypothesis is therefore not a separate analytic carrier. It is the continuous-map certificate row already present in $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$, transported by Cont and compared by hsame.

The polynomial side is read through $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$. The finite error ledger relates the two sides at recorded endpoints only, so the source boundary remains finite and certificate-local. □

Theorem 225.4 (ApproximationUp error ledger exactness)**.** *For* $\mathsf{Approximation}^{\uparrow}$*, the finite error ledger is exact relative to the* BHist *source certified by* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, the candidate certified by* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*, the* hsame *classifier, and the* Cont *transport rows: every ledger entry names one source endpoint, one polynomial endpoint, and one error endpoint, and every exposed error comparison is one of these finite entries. No supremum row or choice data is part of the exactness claim.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket error ledger exactness`

Proof. Project the source, candidate, classifier, and transport rows from the same certificate packet. The ten fields are the source, carrier, pattern, classifier, stability, ledger, exactness, consumer, no-supremum boundary, and no-choice provenance rows.

For an error entry already in the finite ledger, exactness is immediate from its recorded endpoints: it contains the source endpoint, the polynomial endpoint, and the named error endpoint.

Conversely, an exposed approximation error comparison is visible to consumers only by passing through the finite ledger row. Thus there is no hidden supremum-derived error endpoint and no unrecorded choice data. □

Theorem 225.5 (ApproximationUp finite error-ledger consumer determinacy)**.** *For* $\mathsf{Approximation}^{\uparrow}$*, two consumer reads of the same finite error-ledger entry over the same* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *source row and the same* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *candidate row classify to the same source endpoint, polynomial endpoint, and error endpoint under* hsame*. The* Cont *transport rows carry the visible source and candidate rows to those endpoints, and the consumer surface exposes no supremum row or choice row.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationFiniteErrorLedger consumer determinacy`
↪ variant `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket finite error ledger consumer determinacy`

Proof. Apply Definition 225.4 to the first consumer read. The finite ledger entry names a source endpoint, a polynomial endpoint, and an error endpoint, and the read exposes no endpoint outside that recorded triple.

Apply the same exactness theorem to the second consumer read. Because the ledger entry is the same recorded finite entry, the second read has the same three named endpoints.

Use the shared ledger-entry identity to identify the source endpoint, the polynomial endpoint, and the error endpoint under hsame. The visible Cont rows transport only the already recorded source and candidate rows, so no supremum-derived value or choice-selected witness can be read by the consumer. □

Theorem 225.6 (ApproximationUp no-choice polynomial window stability)**.** *Transport the displayed* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *polynomial-candidate row, and one finite error-ledger entry by the chapter's* hsame *classifier rows and finite* Cont *transport rows. The transported read preserves the same* $\mathsf{Approximation}^{\uparrow}$ *polynomial consumer window. It still exposes only the source endpoint, polynomial endpoint, error endpoint, and finite ledger entry, with no supremum object and no choice row.*

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket no choice polynomial window stability`

Proof. Apply Definition 225.4 before transport. The original consumer window is therefore the displayed finite ledger entry together with its source endpoint, polynomial endpoint, and error endpoint.

Apply the same exactness theorem after transporting the continuous-source row, polynomial-candidate row, and finite ledger entry along the chapter's hsame and Cont rows. Exactness again reads only the transported source endpoint, polynomial endpoint, error endpoint, and ledger entry.

Finally use Definition 225.5. Since both reads are over the same finite error-ledger entry up to the displayed transport rows, determinacy identifies the exposed endpoints under hsame. The polynomial consumer window is stable, and no supremum object or choice-selected row is introduced. □

Theorem 225.7 (ApproximationUp NameCert obligation inventory)**.** *The* $\mathsf{Approximation}^{\uparrow}$ NameCert *obligation inventory consists of the* BHist *source row,* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$ *source certificate,* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *candidate certificate,* hsame *classifier,* Cont *stability transport, finite error ledger,*

ledger exactness, polynomial consumer window, no-supremum boundary, and no-choice provenance. These are the only obligations exported by the certificate surface.

Lean4 ✓ `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket namecert obligation inventory`

Proof. List the ten obligations in order: source row, continuous carrier row, polynomial pattern row, hsame classifier row, Cont stability row, finite error ledger row, exactness row, consumer row, no-supremum boundary row, and no-choice provenance row.

The first two rows come from $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$, and the polynomial row comes from $\mathsf{NameCert}_{\mathsf{Poly}^\uparrow}$. The classifier and stability rows are BEDC transport data over the same BHist endpoints.

The final four rows restrict consumers to finite recorded error data. Hence the inventory closes the approximation certificate surface without adding a supremum object, a best-candidate selector, or any choice data. □

Theorem 225.8 (ApproximationUp public certificate export)**.** *Every approximation consumer reads exactly the finite error-ledger surface: the* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^\uparrow}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Poly}^\uparrow}$ *candidate row, the* BHist *source packet, the* hsame *classifier, finite* Cont *transports, source endpoint, polynomial endpoint, error endpoint, finite ledger entry, and no-choice provenance.*

The public certificate exports no supremum object, best-candidate selector, host function equality, or choice row.

Proof. Project the accepted packet through Definition 225.1. The carrier exposes the continuous-map source certificate, polynomial-candidate certificate, BHist packet, classifier, transport, endpoint, ledger, and provenance rows.

Apply Definition 225.2 and Definition 225.3. These keep the consumer window on the displayed polynomial endpoint and continuous source endpoint, without importing a host function-equality row.

Next use Definition 225.4 and Definition 225.5. They identify every readable approximation value with the finite ledger entry and its source, polynomial, and error endpoints.

Finally apply Definition 225.6 and Definition 225.7. The exported certificate is therefore exactly the displayed finite error-ledger surface, and there is no coordinate for a supremum object, best-candidate selector, host function equality, or choice row. □

Closure status for $\mathsf{NameCert}_{\mathsf{Approximation}^\uparrow}$**.**

Constructive story: An approximation packet is generated by taking a compact continuous-map source through $\mathsf{ContinuousMap}^\uparrow$, a bounded-degree polynomial candidate through $\mathsf{Poly}^\uparrow$, and a finite error ledger that names the source endpoint, candidate endpoint, and error endpoint being compared. The carrier accepts only this finite BHist packet with its hsame comparison and Cont transport rows; a supremum object, best-candidate selector, or choice-produced minimizer has no slot in the accepted data. Two approximation packets classify together when the continuous source, polynomial candidate, and recorded error endpoints transport componentwise through the displayed hsame and continuation rows. The source specification is the compact continuous-map certificate surface, the pattern specification is the polynomial-candidate row together with recorded approximation-error comparisons, and the ledger keeps every finite error entry, consumer window, endpoint exactness row, and no-choice provenance row visible. The five name-certificate obligations follow from an admitted packet for carrier inhabitation, the component classifiers inherited from the two dependency certificates, and ledger exactness, which repacks transported source, polynomial, and error endpoints as the same approximation surface. A packet history h lies in the signature gap of an $\mathsf{Approximation}^\uparrow$ token p when the source-domain accepts its continuous source and polynomial-candidate rows, the signature relation pairs it with an answer approximation packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, p records only the candidate and finite error rows actually present in the ledger. The chapter registers this finite error-ledger construction under the name $\mathsf{Approximation}^\uparrow$; it introduces no supremum attainment theorem, no best-polynomial selection functional, and no choice data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Approximation}^\uparrow}$, publicClosed)

Scope closed: The public surface binds ContinuousMapUp, PolynomialUp, BHist, hsame, Cont, the endpoint rows, no-choice provenance, and the finite error ledger displayed by the approximation certificate surface.

Formal verification: FormalStatus(NameCert$_{\text{Approximation}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.ApproximationUp.ApproximationCarrierPacket\_obligation\_surface`

Bridge status: none

Not claimed: This chapter does not close supremum attainment, best-candidate selection, or choice data.

Upgrade path: Bridged closure requires a bridge theorem for the exported finite error-ledger certificate.

Chapter 226

A Concrete Naming Certificate for $\mathsf{Fft}^{\uparrow}$

The Fast-Fourier-transform interface $\mathsf{Fft}^{\uparrow}$ packages the $O(n \log n)$ divide-and-conquer evaluation of the discrete Fourier transform on length-n data when n has small prime factors. It depends on $\mathsf{NameCert}_{\mathsf{Fourier}^{\uparrow}}$ (chapter 124) and $\mathsf{NameCert}_{\mathsf{Complex}^{\uparrow}}$ (chapter 39).

226.1 Carrier definition

Definition 226.1 (FFT BHist source packet)**.** *An* $\mathsf{Fft}^{\uparrow}$ *source packet is a finite* BHist *schedule carrying a* $\mathsf{Complex}^{\uparrow}$ *scalar row, a* $\mathsf{Fourier}^{\uparrow}$ *transform row, a finite* ProbeBundle *of butterfly stages,* InBundle *membership witnesses for every scheduled stage, and a factorization ledger naming the split of the transform length. The carrier accepts exactly those packets whose complex scalar rows and Fourier rows are accepted by their dependency certificates and whose stage rows transport through the same* hsame *and* Cont *evidence.*

Theorem 226.2 (FFT carrier butterfly obligation)**.** *Every accepted* $\mathsf{Fft}^{\uparrow}$ *carrier row projects to a finite butterfly schedule: each stage is a* BHist *row in the displayed* ProbeBundle*, each membership witness is an* InBundle *row, and each butterfly continuation is carried by* Cont *from the* $\mathsf{Complex}^{\uparrow}$ *scalar dependency and the* $\mathsf{Fourier}^{\uparrow}$ *transform dependency. Conversely, those rows repack as the FFT carrier.*

```
Lean4 ✓  BEDC.Derived.FftUp.FftBHistSourcePacket butterfly schedule obligation
Lean4 ✓  BEDC.Derived.FftUp.FftBHistSourcePacket carrier butterfly obligation
```

Proof. Project the $\mathsf{Complex}^{\uparrow}$ and $\mathsf{Fourier}^{\uparrow}$ dependency rows from the packet. They supply the scalar and transform histories used by every scheduled stage.

Read the finite schedule through the ProbeBundle field. Each butterfly row is certified by an InBundle witness, so the schedule is finite at the BEDC surface and does not depend on a host array traversal.

For the converse direction, place the dependency rows, stage membership rows, and butterfly continuations into the packet constructor. The common hsame and Cont evidence transports the stage rows and yields the carried $\mathsf{Fft}^{\uparrow}$ endpoint. □

Theorem 226.3 (FFT factorization classifier stability obligation)**.** *The factorization classifier row of an* $\mathsf{Fft}^{\uparrow}$ *packet is stable under accepted replacement of the complex scalar, Fourier transform,*

and finite butterfly schedule rows. If two packets compare by the displayed hsame *evidence, then the split-length ledger, the stage membership witnesses, and the butterfly continuations classify the same FFT factorization surface.*

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket factorization classifier stability`

Proof. Transport the scalar row through $\mathsf{NameCert}_{\mathsf{Complex}^{\uparrow}}$ and the transform row through $\mathsf{NameCert}_{\mathsf{Fourier}^{\uparrow}}$. These two dependency transports fix the source histories on which the finite schedule is read.

Next transport every scheduled butterfly stage along the same packet comparison. The InBundle witnesses keep the stage list inside the finite ProbeBundle, while Cont transports the butterfly output rows.

The factorization ledger is stored with those stage rows. Repacking the transported dependencies, schedule, and ledger gives the same classifier surface for the target FFT packet. □

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket factorization classifier stability obligation`

Theorem 226.4 (FFT ledger exactness obligation)**.** *The* $\mathsf{Fft}^{\uparrow}$ *ledger consists exactly of the* $\mathsf{NameCert}_{\mathsf{Complex}^{\uparrow}}$ *dependency row, the* $\mathsf{NameCert}_{\mathsf{Fourier}^{\uparrow}}$ *dependency row, the finite* ProbeBundle *schedule, the* InBundle *stage witnesses, the butterfly continuation rows, the factorization ledger rows, and their* hsame *and* Cont *transports. It excludes analytic error estimates, floating-point bounds, asymptotic runtime claims, and standard bridge rows.*

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket ledger exactness obligation`

Proof. Coverage reads any accepted FFT packet into the two dependency rows, the finite schedule, the stage membership witnesses, the butterfly continuations, and the factorization ledger.

Soundness repacks exactly those rows. The scalar and transform dependencies justify the input histories, while InBundle and Cont justify the finite schedule and every butterfly output.

Completeness is the absence of further ledger coordinates. Error estimates, floating-point bounds, asymptotic analysis, and standard bridge data are not fields of the packet, so they cannot be consumed by this certificate. □

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket factorization ledger transport determinacy`

Theorem 226.5 (FFT root threshold unblock surface)**.** *The rows of Definition 226.1, Definition 226.2, Definition 226.3, and Definition 226.4 form the dependency-ready finite-history surface for* $\mathsf{Fft}^{\uparrow}$*. Downstream numerical-analysis certificates may consume the complex dependency, Fourier dependency, finite butterfly schedule, factorization ledger, and exact exclusion of analytic error or bridge data, but must provide any such estimates or bridges outside the FFT packet.*

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket root threshold unblock surface`

Proof. The definition fixes the finite packet and the two dependency certificates. The butterfly obligation turns the packet into carried BHist stage rows over a certified finite schedule.

The factorization stability obligation supplies the classifier transport for the split-length ledger and butterfly continuations. The ledger exactness obligation then states that no analytic estimate or standard bridge row is available inside the packet.

Together these rows expose the concrete FFT source surface needed by later certificates while keeping numerical-analysis obligations separate. □

226.2 The certificate

Theorem 226.6 (FFT schedule public NameCert export)**.** *The source packet of Definition 226.1, the butterfly carrier obligation of Definition 226.2, the factorization classifier stability obligation of Definition 226.3, the ledger exactness obligation of Definition 226.4, and the root threshold surface of Definition 226.5 together export the public* $\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$ *finite schedule interface.*

Consumers may read the $\mathsf{NameCert}_{\mathsf{Complex}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Fourier}^\uparrow}$ *dependencies, the finite* ProbeBundle *schedule, the* InBundle *stage witnesses, the butterfly continuation rows, the factorization ledger, and the* hsame/Cont *transport rows. Analytic error estimates, floating-point bounds, asymptotic runtime claims, and standard bridge rows are not part of this certificate.*

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket schedule public namecert export`

Proof. The source packet fixes the two dependency certificates and the finite ProbeBundle schedule. Its membership rows and butterfly continuations are the only stage data exposed to the BEDC surface.

Apply the butterfly carrier obligation to recover the finite stage witnesses and their Cont rows from any accepted packet. Apply the factorization classifier stability obligation to transport the split-length ledger and the butterfly rows through the displayed hsame comparison.

The ledger exactness obligation states that the readable ledger consists exactly of the two dependency rows, the schedule, the stage witnesses, the butterfly continuations, the factorization ledger, and their transport evidence. The root threshold surface repacks those rows as the public $\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$ interface, while analytic estimates, floating-point bounds, asymptotic runtime claims, and bridge rows remain unavailable. □

Lean4 ✓ `BEDC.Derived.FftUp.FftBHistSourcePacket public namecert export`

Theorem 226.7 (FFT standard schema bridge boundary)**.** *The public finite FFT schedule interface of Definition 226.6 supplies the standard FFT schema bridge boundary exactly through the* $\mathsf{Complex}^\uparrow$ *and* $\mathsf{Fourier}^\uparrow$ *dependency rows, the finite* ProbeBundle *schedule,* InBundle *stage witnesses, butterfly continuation rows, factorization ledger, and* hsame/Cont *transport rows.*

The bridge boundary exports no analytic error estimate, floating-point bound, asymptotic runtime claim, or operation-optimality claim.

Proof. Project the public schedule interface through Definition 226.6. The consumer-visible rows are the two dependency certificates, the finite schedule, stage witnesses, butterfly continuations, factorization ledger, and transport evidence.

Apply Definition 226.2 to read every scheduled stage as a finite ProbeBundle member with an InBundle witness. Then use Definition 226.3 to transport the split-length ledger and butterfly rows along the displayed hsame and Cont surface.

Finally apply Definition 226.4 and Definition 226.5. These rows exhaust the standard FFT schema boundary supplied by the packet, while analytic error estimates, floating-point bounds, asymptotic runtime claims, and operation-optimality claims remain outside the bridge boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$**.**

Constructive story: An FFT packet is presented by the ComplexUp and FourierUp dependency certificates, a finite probe-bundle schedule, displayed stage membership witnesses, butterfly continuation rows, factorization ledger rows, and the transport rows that make those finite rows readable as the public $\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$ surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$, maturePackageClosed)

Scope closed: The mature surface is the finite FFT schedule, public $\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$ export rows, standard schema bridge boundary, factorization classifier, ledger exactness rows, transport determinacy rows, length-one identity surface, schedule-stage exhaustion, and length-one input-classifier theorem over the ComplexUp and FourierUp dependencies.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Fft}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.FftUp.FftBHistSourcePacket\_public\_namecert\_export`

Bridge status: paperBridge

Not claimed: The chapter does not close analytic error bounds, floating-point bounds, asymptotic runtime claims, operation-optimality claims, or schedule completeness beyond the displayed finite bridge.

Upgrade path: No further paper-side theory grade remains for this chapter; bridgeChecked status is a separate verification-axis obligation.

Definition 226.8 (FFT length-one identity surface)**.** *For an accepted* $\mathsf{Fft}^\uparrow$ *source packet of Definition 226.1, write* c *for its* $\mathsf{Complex}^\uparrow$ *scalar row,* F *for its* $\mathsf{Fourier}^\uparrow$ *transform row,* x *for the carrier row produced from* c *and* F*,* β *for the scheduled butterfly row,* ν *for the factorization row,* λ *for the ledger row,* y *for the endpoint row,* σ *for the stage name, and* Π *for the finite schedule. The predicate* $\mathsf{FftLenOneId}(P, x, y)$ *holds when the following rows are displayed:*

- *the schedule has length one,* $\mathsf{BLEN}(\Pi) = 1$*;*
- *the carrier membership row of Definition 226.2 reads* σ *as the unique scheduled stage and reads that stage back as the input row* x*;*
- *the singleton butterfly row supplied by* $\mathsf{NameCert}_{\mathsf{Fourier}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{Complex}^\uparrow}$ *classifies the butterfly ledger with the input row,* $\lambda \sim_{\mathsf{Complex}^\uparrow} x$*;*
- *the length-one factorization ledger is neutral on that ledger, so the endpoint row classifies with the ledger row,* $y \sim_{\mathsf{Complex}^\uparrow} \lambda$*.*

This surface contains no analytic FFT equality, host array equality, floating-point estimate, asymptotic claim, or standard bridge row.

Lemma 226.9 (FFT length-one schedule stage exhaustion)**.** *Let* P *be an accepted* $\mathsf{Fft}^\uparrow$ *source packet of Definition 226.1 with finite schedule* Π*. If* $\mathsf{BLEN}(\Pi) = 1$*, then every scheduled butterfly stage read by the carrier obligation is the single stage named by the packet membership row. In particular, the packet exposes no second butterfly stage and no tail-stage membership row.*

Proof. Apply Definition 226.2 to the accepted packet. This gives the displayed schedule membership row and the butterfly continuation rows inside the finite $\mathsf{ProbeBundle}$ schedule.

The length-one hypothesis and Definition 6.6 identify Π with a singleton bundle. Then Definition 6.7 identifies the member supplied by the carrier obligation with that singleton head. A tail membership would be a membership row in Bnil, which is impossible by the same singleton reading. Hence the carrier has exactly the one scheduled stage named by the packet membership row. □

Theorem 226.10 (FFT length-one schedule classifies with the input row)**.** *Let* P *be an accepted* $\mathsf{Fft}^\uparrow$ *source packet of Definition 226.1. If* $\mathsf{FftLenOneId}(P, x, y)$ *holds for its input row* x *and endpoint row* y*, then the FFT output endpoint is classified with the input row:* $y \sim_{\mathsf{Complex}^\uparrow} x$*.*

Proof. Read the accepted packet through Definition 226.2. The packet yields the $\mathsf{Complex}^\uparrow$ scalar dependency, the $\mathsf{Fourier}^\uparrow$ transform dependency, the finite schedule membership row, the singleton butterfly continuation, and the continuation rows that lead to the endpoint.

By the first row of Definition 226.8, the schedule has length one. The exhaustion lemma Definition 226.9 therefore removes every possible tail stage: the butterfly continuation seen by the carrier obligation is the unique scheduled row, and its stage readback is the input row x.

The singleton butterfly row in Definition 226.8 classifies the butterfly ledger with the input row, giving $\lambda \sim_{\mathsf{Complex}^{\uparrow}} x$. Next apply Definition 226.4. The ledger contains exactly the two dependency rows, the singleton schedule membership row, the butterfly continuation, the factorization row, and their displayed transports; there is no further stage or hidden analytic row that can change the endpoint classification.

The neutral length-one factorization row in Definition 226.8 gives $y \sim_{\mathsf{Complex}^{\uparrow}} \lambda$. Transitivity of the $\mathsf{Complex}^{\uparrow}$ classifier supplied by the dependency certificate then composes $y \sim_{\mathsf{Complex}^{\uparrow}} \lambda$ with $\lambda \sim_{\mathsf{Complex}^{\uparrow}} x$, yielding $y \sim_{\mathsf{Complex}^{\uparrow}} x$. □

Chapter 227

A Concrete Naming Certificate for $\mathsf{Quadrature}^{\uparrow}$

The numerical-quadrature interface $\mathsf{Quadrature}^{\uparrow}$ packages a finite weighted sum approximating an integral on a compact interval, classified by exactness degree (trapezoid, Simpson, Gauss-Legendre). It depends on $\mathsf{NameCert}_{\mathsf{Integral}^{\uparrow}}$ (chapter 123) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

227.1 Carrier definition

Definition 227.1 (Quadrature carried rule source row)**.** *A carried* $\mathsf{Quadrature}^{\uparrow}$ *rule source row is the* BHist*-visible tuple*

$$(xs, \alpha, \omega, \mathcal{I}_Q)$$

where xs *is the* $\mathsf{FinSet}^{\uparrow}$ *node spine, each node endpoint* α_i *and weight endpoint* ω_i *is carried by the scalar endpoint rows, and* $\mathcal{I}_Q$ *is the endpoint map supplied by the* $\mathsf{Integral}^{\uparrow}$ *polynomial row on the fixed compact interval. The polynomial inputs to the row are finite* $\mathsf{ListCarrier}$ *coefficient codes. Ordering choices for the node spine, numerical approximation choices, and convergence data are not carrier fields of the rule; when they are used, they are recorded only by ledger rows attached to the source tuple.*

227.2 The certificate

Definition 227.2 (Quadrature finite weighted exactness degree)**.** *Fix a scalar carrier* R *with classifier* $\sim_R$*, zero endpoint* 0_R*, addition* $+$*, multiplication* $\cdot$*, canonical polynomial trim* trim_R*, and Horner evaluation* $\mathsf{Eval}_{\alpha,R}$ *as in Definition 50.133. A finite weighted* $\mathsf{Quadrature}^{\uparrow}$ *rule over* R *is a tuple*

$$Q = (xs, \alpha, \omega, \mathcal{I}_Q)$$

where xs *is the finite* $\mathsf{FinSet}^{\uparrow}$ *index spine for the nodes,* α_i *and* ω_i *are carried scalar endpoints for each* $i : \mathsf{Pos}(xs)$*, and* $\mathcal{I}_Q(p)$ *is the scalar endpoint supplied by the* $\mathsf{Integral}^{\uparrow}$ *row for the polynomial code* p *on the fixed compact interval. Its quadrature sum on a finite coefficient spine* $p : \mathsf{ListCarrier}(R)$ *is*

$$\mathsf{QSum}_Q(p) := \mathsf{fold}_{+,\mathsf{Pos}(xs)}(i \mapsto \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p)).$$

The pointwise exactness predicate is

$$\mathsf{QExact}_Q(p) \iff \mathsf{QSum}_Q(p) \sim_R \mathcal{I}_Q(p).$$

For a unary degree bound $n : \mathsf{Nat}^{\uparrow}$*, define*

$$\mathsf{PolyDegLe}_R(p, n) \iff \forall k : \mathsf{Nat}^{\uparrow}.\ \mathsf{NatUnaryStrictPrefix}(n, k) \implies \mathsf{trim}_R(p)[k] \sim_R 0_R.$$

The quadrature exactness-degree predicate is

$$\mathsf{QuadratureExactUpTo}_R(Q, n) \iff \forall p : \mathsf{ListCarrier}(R).\ \mathsf{PolyDegLe}_R(p, n) \implies \mathsf{QExact}_Q(p).$$

Finally, the degree-bound comparison used by this $\mathsf{Quadrature}^{\uparrow}$ *classifier is*

$$\mathsf{DegBoundLe}(e, d) \iff \forall k : \mathsf{Nat}^{\uparrow}.\ \mathsf{NatUnaryStrictPrefix}(d, k) \implies \mathsf{NatUnaryStrictPrefix}(e, k).$$

This is the unary BEDC reading of the ordinary inequality $e \le d$*: every coefficient index above* d *is also above* e*.*

Theorem 227.3 (Quadrature carrier scope obligation)**.** *For the* $\mathsf{Quadrature}^{\uparrow}$ *certificate described by Definition 227.1 and Definition 227.2, the carrier scope is exactly the* $\mathsf{FinSet}^{\uparrow}$ *node spine* xs*, the scalar endpoint rows carrying* α_i *and* ω_i*, the* $\mathsf{Integral}^{\uparrow}$ *endpoint map* $\mathcal{I}_Q$*, the* ListCarrier *polynomial code* p*, and the* $\mathsf{Nat}^{\uparrow}$ *degree bound* n*. No host ordering of nodes, host analytic function, host real equality, convergence assumption, or approximation scheme is part of the carried source row.*

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureCarriedRuleSourceRow scope`

Proof. The carrier row of Definition 227.1 names only the tuple $(xs, \alpha, \omega, \mathcal{I}_Q)$, with polynomial inputs supplied as finite ListCarrier codes. The exactness-degree definition then adds only the unary bound $n : \mathsf{Nat}^{\uparrow}$ and the derived rows QSum_Q, QExact_Q, $\mathsf{PolyDegLe}_R$, and $\mathsf{QuadratureExactUpTo}_R$.

Each displayed row is built from the listed source objects: folding over $\mathsf{Pos}(xs)$, scalar multiplication and addition, Horner evaluation on a list code, the integral endpoint map, trimming, and strict-prefix comparison on unary naturals. Therefore any datum outside these rows can enter only as a ledger entry and not as carrier scope. □

Theorem 227.4 (Quadrature classifier scope obligation)**.** *The exactness-degree classifier for the carried* $\mathsf{Quadrature}^{\uparrow}$ *rule is the preorder row* DegBoundLe *on* $\mathsf{Nat}^{\uparrow}$ *bounds together with the weakening action on* $\mathsf{QuadratureExactUpTo}_R$*. Its source predicates are precisely* NatUnaryStrictPrefix *and* $\mathsf{PolyDegLe}_R$ *as displayed in Definition 227.2.*

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureExactUpTo classifier scope`

Proof. The classifier comparison is the displayed predicate $\mathsf{DegBoundLe}(e, d)$. By Definition 227.6, this comparison transports a polynomial degree window from the smaller bound e to the larger bound d using only NatUnaryStrictPrefix and the coefficient condition in $\mathsf{PolyDegLe}_R$.

Applying Definition 227.7 to that transported window gives the classifier action on $\mathsf{QuadratureExactUpTo}_R$. The reflexive and transitive rows recorded in Definition 227.11 identify the comparison surface as a preorder on the same unary bounds, so the classifier has no additional source objects. □

Theorem 227.5 (Quadrature ledger exactness obligation)**.** *For a carried* $\mathsf{Quadrature}^{\uparrow}$ *rule, the ledger exactness surface records the node spine, weights, integral endpoint provenance, polynomial trim and evaluation rows, and the pointwise predicate* QExact_Q*. The ledger does not assert host real equality or host function equality; exactness is the carried scalar endpoint relation*

$$\mathsf{QSum}_Q(p) \sim_R \mathcal{I}_Q(p)$$

on finite polynomial codes.

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureLedgerExactness surface`

Proof. Unfold Definition 227.2. The quadrature sum is a finite fold over $\mathsf{Pos}(xs)$ using the carried weights ω_i and Horner evaluation at the carried node endpoints α_i. The integral side is the carried endpoint $\mathcal{I}_Q(p)$ supplied by the $\mathsf{Integral}^{\uparrow}$ polynomial row.

The polynomial side is controlled by trim_R and $\mathsf{PolyDegLe}_R$, while exactness itself is exactly the predicate $\mathsf{QExact}_Q(p)$. The window inclusion lemma, the weakening theorem, and the preorder rows cited above transport this same predicate between unary degree bounds without replacing it by host equality of real numbers or functions. □

Lemma 227.6 (Polynomial degree windows are nested)**.** *With the notation of Definition 227.2, if* $\mathsf{DegBoundLe}(e, d)$ *holds, then every polynomial code with degree at most* e *also has degree at most* d*:*

$$\mathsf{PolyDegLe}_R(p, e) \Longrightarrow \mathsf{PolyDegLe}_R(p, d).$$

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadraturePolynomialDegreeWindow inclusion`

Proof. Unfold $\mathsf{PolyDegLe}_R$ and $\mathsf{DegBoundLe}$. Assume $\mathsf{PolyDegLe}_R(p, e)$ and fix $k : \mathsf{Nat}^{\uparrow}$ with $\mathsf{NatUnaryStrictPrefix}(d, k)$. The degree-bound comparison sends this strict-prefix witness to $\mathsf{NatUnaryStrictPrefix}(e, k)$. Applying the degree-at-most-e hypothesis at this k gives

$$\mathsf{trim}_R(p)[k] \sim_R 0_R.$$

This is precisely the coefficient condition required for $\mathsf{PolyDegLe}_R(p, d)$. □

Theorem 227.7 (Quadrature exactness degree weakening)**.** *Let* Q *be a finite weighted* $\mathsf{Quadrature}^{\uparrow}$ *rule over the scalar carrier* R *in the sense of Definition 227.2. For unary degree bounds* $e, d : \mathsf{Nat}^{\uparrow}$*, if*

$$\mathsf{QuadratureExactUpTo}_R(Q, d) \qquad \textit{and} \qquad \mathsf{DegBoundLe}(e, d)$$

hold, then the same rule is exact on every polynomial of degree at most e*:*

$$\mathsf{QuadratureExactUpTo}_R(Q, e).$$

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureExactUpTo weakening`

Proof. Unfold the target exactness predicate. Fix a finite coefficient spine $p : \mathsf{ListCarrier}(R)$ and assume $\mathsf{PolyDegLe}_R(p, e)$. By Definition 227.6, the degree-bound comparison gives $\mathsf{PolyDegLe}_R(p, d)$. The exactness hypothesis at degree d applies to this same polynomial code and returns

$$\mathsf{QExact}_Q(p).$$

Unfolding QExact_Q identifies this row as

$$\mathsf{QSum}_Q(p) \sim_R \mathcal{I}_Q(p),$$

which is the required exactness row for p under the degree-e predicate. Since p was arbitrary, $\mathsf{QuadratureExactUpTo}_R(Q, e)$ holds. The finite node spine, the weights, and the integral endpoint map are the original components of Q throughout. □

Theorem 227.8 (Quadrature exactness degree equivalence stability)**.** *Let Q be a finite weighted* $\mathsf{Quadrature}^{\uparrow}$ *rule over the scalar carrier R in the sense of Definition 227.2. For unary degree bounds $e, d : \mathsf{Nat}^{\uparrow}$, if*

$$\mathsf{DegBoundLe}(e, d) \qquad \textit{and} \qquad \mathsf{DegBoundLe}(d, e)$$

hold, then the exactness classifiers at d and e determine the same row:

$$\mathsf{QuadratureExactUpTo}_R(Q, d) \Longleftrightarrow \mathsf{QuadratureExactUpTo}_R(Q, e).$$

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureExactUpTo degree equivalence stability`

Proof. We prove the two directions separately.

For the forward direction, assume $\mathsf{QuadratureExactUpTo}_R(Q, d)$. Apply Definition 227.7 to this degree-d exactness row and to the comparison $\mathsf{DegBoundLe}(e, d)$. The result is $\mathsf{QuadratureExactUpTo}_R(Q, e)$.

For the reverse direction, assume $\mathsf{QuadratureExactUpTo}_R(Q, e)$. Apply Definition 227.7 to this degree-e exactness row and to the comparison $\mathsf{DegBoundLe}(d, e)$. The result is $\mathsf{QuadratureExactUpTo}_R(Q, d)$. □

Theorem 227.9 (Quadrature degree-bound comparison is transitive)**.** *With the notation of Definition 227.2, the unary degree-bound comparison used by the* $\mathsf{Quadrature}^{\uparrow}$ *exactness-degree classifier is transitive. For unary degree bounds $e, d, c : \mathsf{Nat}^{\uparrow}$, if*

$$\mathsf{DegBoundLe}(e, d) \qquad \textit{and} \qquad \mathsf{DegBoundLe}(d, c)$$

hold, then

$$\mathsf{DegBoundLe}(e, c)$$

holds.

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureDegBoundLe trans`

Proof. Unfold $\mathsf{DegBoundLe}$. To prove $\mathsf{DegBoundLe}(e, c)$, fix $k : \mathsf{Nat}^{\uparrow}$ and assume

$$\mathsf{NatUnaryStrictPrefix}(c, k).$$

The comparison $\mathsf{DegBoundLe}(d, c)$ sends this witness to

$$\mathsf{NatUnaryStrictPrefix}(d, k).$$

Applying $\mathsf{DegBoundLe}(e, d)$ to that witness gives

$$\mathsf{NatUnaryStrictPrefix}(e, k).$$

This is exactly the strict-prefix condition required at the index k for $\mathsf{DegBoundLe}(e, c)$. Since k was arbitrary, the comparison from e to c holds. □

Theorem 227.10 (Quadrature degree-bound comparison is reflexive)**.** *With the notation of Definition 227.2, the exactness-degree comparison used by the* $\mathsf{Quadrature}^{\uparrow}$ *classifier is reflexive. For every unary degree bound $d : \mathsf{Nat}^{\uparrow}$,*

$$\mathsf{DegBoundLe}(d, d)$$

holds.

Proof. Unfold DegBoundLe at the pair (d, d). The required statement is

$$\forall k : \mathsf{Nat}^{\uparrow}.\ \mathsf{NatUnaryStrictPrefix}(d, k) \Longrightarrow \mathsf{NatUnaryStrictPrefix}(d, k).$$

Fix $k : \mathsf{Nat}^{\uparrow}$ and assume $\mathsf{NatUnaryStrictPrefix}(d, k)$. The conclusion is the same strict-prefix witness. Since k was arbitrary, the displayed universal clause holds, and therefore $\mathsf{DegBoundLe}(d, d)$ holds. □

Proposition 227.11 (Quadrature degree-bound comparison preorder rows)**.** *With the notation of Definition 227.2, the unary degree-bound comparison attached to the* $\mathsf{Quadrature}^{\uparrow}$ *exactness-degree classifier supplies the reflexive and transitive rows on unary bounds:*

$$\forall d : \mathsf{Nat}^{\uparrow}.\ \mathsf{DegBoundLe}(d, d)$$

and

$$\forall e, d, c : \mathsf{Nat}^{\uparrow}.\ \mathsf{DegBoundLe}(e, d) \Longrightarrow \mathsf{DegBoundLe}(d, c) \Longrightarrow \mathsf{DegBoundLe}(e, c).$$

Proof. The reflexive row is Definition 227.10. The transitive row is Definition 227.9. Both rows use the same concrete predicate DegBoundLe from Definition 227.2, so together they give the stated preorder data for the $\mathsf{Quadrature}^{\uparrow}$ degree-bound classifier. □

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureDegBoundLe preorder rows`

Closure status for $\mathsf{NameCert}_{\mathsf{Quadrature}^{\uparrow}}$**.**

Constructive story: A quadrature rule is generated from a finite $\mathsf{FinSet}^{\uparrow}$ node spine, carried scalar endpoints for each node and weight, an $\mathsf{Integral}^{\uparrow}$ endpoint map for polynomial codes on the fixed compact interval, and a unary $\mathsf{Nat}^{\uparrow}$ degree bound. A row is admitted only through this finite weighted tuple together with finite list polynomial codes, Horner evaluation, trimming, scalar addition and multiplication, and the displayed exactness predicate; host node orderings, analytic functions, convergence data, numerical approximations, and host real equality are not carrier fields. The classifier on exactness degree is the unary preorder DegBoundLe, which says that every coefficient index above one bound is also above the other, and it transports exactness by narrowing the polynomial window rather than by comparing real-valued functions externally. The source covers the node spine, scalar node and weight endpoints, integral endpoint map, polynomial code, and degree bound; the pattern records the finite weighted sum, polynomial degree window, pointwise exactness, weakening, equivalence stability, and preorder rows; the ledger keeps the nodes, weights, Horner rows, trim rows, integral provenance, and scalar endpoint comparisons visible. The five name-certificate obligations are supplied by the inhabited finite-rule surface and the reflexive and transitive degree-bound comparison, with carrier closure coming from reusing the same finite tuple while transporting exactness through the displayed scalar classifier. A history h lies in the signature gap of a $\mathsf{Quadrature}^{\uparrow}$ token p when the source-domain accepts the finite rule and degree-bound row, the signature relation pairs it with an answer rule and evidence ledger, and token introduction names that answer as p; unfolded, the weighted sum is exact on precisely the polynomial codes within the unary degree window encoded by p. The chapter registers this finite weighted exactness construction under the name $\mathsf{Quadrature}^{\uparrow}$ and provides a standard finite-rule bridge without convergence claims, analytic error estimates, host function equality, or host real equality.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Quadrature}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridged boundary is the finite-rule carrier, the DegBoundLe classifier, exactness weakening and degree-equivalence stability, the ledger exactness surface, the exactness-degree preorder rows, and the external finite-rule bridge visible to downstream numerical-analysis chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Quadrature}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.QuadratureUp.QuadratureExactUpTo\_degree\_equivalence\_stability`

Bridge status: paperBridge

Not claimed: This chapter does not close convergence of numerical schemes, analytic error estimates, host real equality, host function equality, or machine-checked bridge evidence.

Upgrade path: No further paper theory-axis step is claimed for this chapter; bridge verification remains a separate status.

Lemma 227.12 (Quadrature empty position fold)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$ whose finite additive scalar fold is the fold of Definition 49.80.*

Let xs be a finite $\mathsf{FinSet}^{\uparrow}$ *node spine. If the position carrier* $\mathsf{Pos}(xs)$ *has no positions, then every scalar summand family* $s : \mathsf{Pos}(xs) \to R$ *satisfies*

$$\mathsf{fold}_{+,\mathsf{Pos}(xs)}(i \mapsto s(i)) \sim_R 0_R.$$

Proof. The finite-position ledger of Definition 112.27 reads a fold over $\mathsf{Pos}(xs)$ as the finite additive scalar fold over the scalar spine obtained by enumerating those positions. If $\mathsf{Pos}(xs)$ has no positions, that scalar spine is nil; no value of s is read into the ledger.

By the nil clause of Definition 49.80, equivalently by Definition 49.79, the fold over this empty scalar spine is classified with 0_R. This gives the displayed classifier comparison. □

Theorem 227.13 (Quadrature empty-node sum is zero)**.** *Let $Q = (xs, \alpha, \omega, \mathcal{I}_Q)$ be a finite weighted* $\mathsf{Quadrature}^{\uparrow}$ *rule over the scalar carrier R in the sense of Definition 227.2. Assume the* $\mathsf{FinSet}^{\uparrow}$ *node spine is empty, equivalently* $\mathsf{Pos}(xs)$ *has no positions. Then for every finite polynomial-code spine* $p : \mathsf{ListCarrier}(R)$*,*

$$\mathsf{QSum}_Q(p) \sim_R 0_R.$$

Proof. Fix a finite coefficient spine $p : \mathsf{ListCarrier}(R)$. Unfolding Definition 227.2 gives

$$\mathsf{QSum}_Q(p) := \mathsf{fold}_{+,\mathsf{Pos}(xs)}\big(i \mapsto \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p)\big).$$

The empty-node hypothesis says that the indexing carrier of this fold has no positions. Apply Definition 227.12 to the summand family

$$s(i) := \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p).$$

The result is

$$\mathsf{fold}_{+,\mathsf{Pos}(xs)}\big(i \mapsto \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p)\big) \sim_R 0_R.$$

Substituting the displayed definition of $\mathsf{QSum}_Q(p)$ gives $\mathsf{QSum}_Q(p) \sim_R 0_R$. The endpoint map $\mathcal{I}_Q$ and the pointwise exactness predicate do not enter this boundary calculation, because the claim concerns only the actual quadrature fold. □

Lemma 227.14 (Quadrature singleton position fold)**.** *Let R be a scalar carrier with classifier $\sim_R$, zero endpoint 0_R, and addition $+$ whose finite additive scalar fold is the fold of Definition 49.80. Let xs be a finite* $\mathsf{FinSet}^{\uparrow}$ *node spine. If the position carrier* $\mathsf{Pos}(xs)$ *enumerates exactly the one position i_0, then every scalar summand family* $s : \mathsf{Pos}(xs) \to R$ *satisfies*

$$\mathsf{fold}_{+,\mathsf{Pos}(xs)}(i \mapsto s(i)) \sim_R s(i_0).$$

Proof. The finite-position ledger of Definition 112.27 reads a fold over $\mathsf{Pos}(xs)$ as the finite additive scalar fold over the scalar spine obtained by enumerating those positions. Under the singleton-position hypothesis, that scalar spine is exactly $\mathsf{singleton}(s(i_0))$; no other value of s is read into the ledger.

Thus the displayed position fold is the same folded endpoint as

$$\mathsf{fold}_{+,R}(\mathsf{singleton}(s(i_0))).$$

By the singleton clause of Definition 49.80, equivalently by Definition 49.78, this singleton scalar fold is classified with $s(i_0)$. Transporting along the finite-position readback gives

$$\mathsf{fold}_{+,\mathsf{Pos}(xs)}(i \mapsto s(i)) \sim_R s(i_0).$$

□

Theorem 227.15 (Quadrature singleton-node sum is weighted evaluation)**.** *Let $Q = (xs, \alpha, \omega, \mathcal{I}_Q)$ be a finite weighted* $\mathsf{Quadrature}^{\uparrow}$ *rule over the scalar carrier R in the sense of Definition 227.2. Assume the* $\mathsf{FinSet}^{\uparrow}$ *node spine enumerates exactly one position i_0, equivalently* $\mathsf{Pos}(xs)$ *has the single position i_0. Then for every finite polynomial-code spine $p : \mathsf{ListCarrier}(R)$,*

$$\mathsf{QSum}_Q(p) \sim_R \omega_{i_0} \cdot \mathsf{Eval}_{\alpha_{i_0},R}(p).$$

Proof. Fix a finite coefficient spine $p : \mathsf{ListCarrier}(R)$. Unfolding Definition 227.2 gives

$$\mathsf{QSum}_Q(p) := \mathsf{fold}_{+,\mathsf{Pos}(xs)}\big(i \mapsto \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p)\big).$$

Apply Definition 227.14 to the summand family

$$s(i) := \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p).$$

The singleton-node hypothesis identifies the finite-position scalar spine as the singleton spine whose sole entry is

$$\omega_{i_0} \cdot \mathsf{Eval}_{\alpha_{i_0},R}(p).$$

Therefore the quadrature fold is classified with that sole entry:

$$\mathsf{fold}_{+,\mathsf{Pos}(xs)}\big(i \mapsto \omega_i \cdot \mathsf{Eval}_{\alpha_i,R}(p)\big) \sim_R \omega_{i_0} \cdot \mathsf{Eval}_{\alpha_{i_0},R}(p).$$

Substituting the displayed definition of $\mathsf{QSum}_Q(p)$ gives

$$\mathsf{QSum}_Q(p) \sim_R \omega_{i_0} \cdot \mathsf{Eval}_{\alpha_{i_0},R}(p).$$

The endpoint map $\mathcal{I}_Q$ and the exactness predicate QExact_Q do not enter this boundary calculation, because the claim concerns only the finite weighted quadrature fold. □

Lemma 227.16 (Finite additive fold decomposes over spine append)**.** *Let R be a scalar* $\mathsf{Ring}^{\uparrow}$ *carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, and finite additive fold* $\mathsf{fold}_{+,R}$ *from Definition 49.80. Assume the classifier equivalence rows, additive congruence, additive associativity, and additive left-zero row retained in Definition 43.26. For finite carried scalar spines $xs, ys : \mathsf{ListCarrier}(R)$,*

$$\mathsf{fold}_{+,R}(xs{+}{+}ys) \sim_R \mathsf{fold}_{+,R}(xs) +_R \mathsf{fold}_{+,R}(ys).$$

Lean4 ✓ `BEDC.Derived.QuadratureUp.QuadratureFiniteAdditiveFold append decomposition`

Proof. Induct on the finite spine xs.

If $xs = \mathsf{nil}$, then $xs{+}{+}ys = ys$ and Definition 49.80 identifies $\mathsf{fold}_{+,R}(xs)$ with 0_R. The additive left-zero row classifies $0_R +_R \mathsf{fold}_{+,R}(ys)$ with $\mathsf{fold}_{+,R}(ys)$, and classifier symmetry reverses that comparison. Hence

$$\mathsf{fold}_{+,R}(ys) \sim_R 0_R +_R \mathsf{fold}_{+,R}(ys),$$

which is the displayed claim in the nil case.

For the cons case, write $xs = \mathsf{cons}(a, xs_0)$. The fold clauses give

$$\mathsf{fold}_{+,R}(\mathsf{cons}(a, xs_0){+}{+}ys) = a +_R \mathsf{fold}_{+,R}(xs_0{+}{+}ys)$$

and

$$\mathsf{fold}_{+,R}(\mathsf{cons}(a, xs_0)) = a +_R \mathsf{fold}_{+,R}(xs_0).$$

The induction hypothesis supplies

$$\mathsf{fold}_{+,R}(xs_0{+}{+}ys) \sim_R \mathsf{fold}_{+,R}(xs_0) +_R \mathsf{fold}_{+,R}(ys).$$

Additive congruence, using classifier reflexivity on a, classifies

$$a +_R \mathsf{fold}_{+,R}(xs_0{+}{+}ys)$$

with

$$a +_R (\mathsf{fold}_{+,R}(xs_0) +_R \mathsf{fold}_{+,R}(ys)).$$

Additive associativity then classifies this endpoint with

$$(a +_R \mathsf{fold}_{+,R}(xs_0)) +_R \mathsf{fold}_{+,R}(ys),$$

which is exactly

$$\mathsf{fold}_{+,R}(\mathsf{cons}(a, xs_0)) +_R \mathsf{fold}_{+,R}(ys).$$

Classifier transitivity composes these comparisons and proves the cons case. □

Theorem 227.17 (Quadrature node-list append sum decomposition)**.** *Let R be a scalar $\mathsf{Ring}^{\uparrow}$ carrier with classifier $\sim_R$, zero endpoint 0_R, addition $+_R$, multiplication $\cdot_R$, canonical trim, and Horner evaluation as in Definition 227.2. Assume the additive rows required by Definition 227.16. Let A and B be finite $\mathsf{FinSet}^{\uparrow}$ node spines, and set*

$$L = A{+}{+}B.$$

Let

$$Q_L = (L, \alpha, \omega, \mathcal{I}_Q)$$

be a finite weighted $\mathsf{Quadrature}^{\uparrow}$ rule. Let Q_A and Q_B be the restricted rules obtained from the same node, weight, and integral endpoint data along the left and right position injections of the finite-position append ledger:

$$Q_A = (A, \alpha|_{\mathsf{Pos}(A)}, \omega|_{\mathsf{Pos}(A)}, \mathcal{I}_Q), \qquad Q_B = (B, \alpha|_{\mathsf{Pos}(B)}, \omega|_{\mathsf{Pos}(B)}, \mathcal{I}_Q).$$

Then every finite polynomial-code spine $p : \mathsf{ListCarrier}(R)$ satisfies

$$\mathsf{QSum}_{Q_L}(p) \sim_R \mathsf{QSum}_{Q_A}(p) +_R \mathsf{QSum}_{Q_B}(p).$$

Proof. Fix $p : \mathsf{ListCarrier}(R)$. For a node position i, write the quadrature summand as

$$s_p(i) := \omega_i \cdot_R \mathsf{Eval}_{\alpha_i,R}(p).$$

Unfolding Definition 227.2 gives

$$\mathsf{QSum}_{Q_L}(p) := \mathsf{fold}_{+_R,\mathsf{Pos}(L)}(i \mapsto s_p(i)),$$

with the analogous formulas for Q_A and Q_B over $\mathsf{Pos}(A)$ and $\mathsf{Pos}(B)$.

By the finite-position ledger of Definition 112.27, together with the append split underlying Definition 112.18, the positions of $L = A{+}{+}B$ are read as the left positions of A followed by the right positions of B. Therefore the scalar spine enumerated by the summand family s_p over $\mathsf{Pos}(L)$ is the concatenation

$$S_A(p){+}{+}S_B(p),$$

where

$$S_A(p) = \big(i \in \mathsf{Pos}(A) \mapsto \omega_i \cdot_R \mathsf{Eval}_{\alpha_i,R}(p)\big)$$

and

$$S_B(p) = \big(j \in \mathsf{Pos}(B) \mapsto \omega_j \cdot_R \mathsf{Eval}_{\alpha_j,R}(p)\big).$$

The same position readback identifies

$$\mathsf{QSum}_{Q_A}(p) \sim_R \mathsf{fold}_{+,R}(S_A(p))$$

and

$$\mathsf{QSum}_{Q_B}(p) \sim_R \mathsf{fold}_{+,R}(S_B(p)),$$

while the appended rule is read as

$$\mathsf{QSum}_{Q_L}(p) \sim_R \mathsf{fold}_{+,R}(S_A(p) {+}{+} S_B(p)).$$

Applying Definition 227.16 to the two carried scalar spines $S_A(p)$ and $S_B(p)$ gives

$$\mathsf{fold}_{+,R}(S_A(p) {+}{+} S_B(p)) \sim_R \mathsf{fold}_{+,R}(S_A(p)) +_R \mathsf{fold}_{+,R}(S_B(p)).$$

Classifier transitivity and additive congruence transport the two restricted-rule readbacks through the right-hand side, yielding

$$\mathsf{QSum}_{Q_L}(p) \sim_R \mathsf{QSum}_{Q_A}(p) +_R \mathsf{QSum}_{Q_B}(p).$$

The endpoint map $\mathcal{I}_Q$ is shared by the three rules but is not used in this fold identity; the calculation concerns only the finite weighted quadrature sum. □

Theorem 227.18 (Quadrature external finite-rule bridge)**.** *The finite node spine, node and weight endpoints, Horner polynomial-code evaluation, endpoint integral map, degree-bound classifier, exactness weakening, degree-equivalence stability, and append-sum decomposition assemble the standard finite-rule bridge for* $\mathsf{Quadrature}^{\uparrow}$*. The bridge reads exactly the public* $\mathsf{NameCert}_{\mathsf{Quadrature}^{\uparrow}}$ *finite-rule certificate and exports it as a standard finite quadrature-rule boundary.*

The bridge adds no convergence theorem, analytic error estimate, host real equality, host function equality, infinite node family, or external ordering of nodes.

Proof. First read the public certificate rows from Definition 227.3, Definition 227.4, and Definition 227.5. These rows provide the finite node spine, scalar node and weight endpoints, endpoint integral map, Horner polynomial-code evaluation, degree-bound classifier, and exactness ledger.

The boundary cases of the finite-rule algebra are Definition 227.13 and Definition 227.15. They identify the empty and singleton node reads as finite weighted-sum rows of the same certificate.

For composite node spines, apply Definition 227.17. Together with Definition 227.7 and Definition 227.8, this gives the standard bridge read of the finite rule without introducing any analytic or host-level data beyond the public certificate. □

Chapter 228

A Concrete Naming Certificate for $\mathsf{FisherInfo}^{\uparrow}$

The Fisher-information interface $\mathsf{FisherInfo}^{\uparrow}$ packages, for a parametric family of distributions, the Riemannian metric whose components are expectations of squared score functions. It depends on $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186) and $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ (chapter 133).

228.1 Carrier definition

Definition 228.1 (FisherInfoUp BHist score carrier)**.** *A carried* $\mathsf{FisherInfo}^{\uparrow}$ *score packet is an eight-tuple*

$$(\mathcal{D}, \mathcal{G}, \theta, s, \eta, g, \rho, \lambda).$$

Here $\mathcal{D}$ *is a displayed* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *packet for the parametric distribution family,* $\mathcal{G}$ *is a displayed* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ *metric packet,* θ *is the parameter row,* s *is the score-observable row over the family,* η *is the expectation row reading the squared score through the distribution endpoint,* g *is the metric-component row,* ρ *is the shared* Pkg *provenance row, and* λ *is a finite* Cont *ledger connecting these displayed rows. The packet is accepted exactly when each coordinate is a* BHist *row, all comparisons used by the ledger are* hsame *comparisons,* λ *is a finite* Cont *composite,* ρ *is the common* Pkg *provenance, and the only dependency rows used by the carrier are the displayed* $\mathsf{Distribution}^{\uparrow}$ *and* $\mathsf{RiemannianMetric}^{\uparrow}$ *rows.*

Theorem 228.2 (FisherInfoUp distribution source obligation)**.** *Every accepted packet of Definition 228.1 projects a* $\mathsf{Distribution}^{\uparrow}$ *source row whose pushforward endpoint is the only probability-distribution data available to the score and expectation ledger. In particular, the packet consumes the pushforward carrier of Definition 186.1, the classifier of Definition 186.2, and the pushforward row of Definition 186.3; it does not import an ambient probability space outside the displayed* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *surface.*

Proof. Unpack the accepted tuple. Its first field is the displayed source certificate $\mathcal{D}$, and the provenance row ρ binds every later score, expectation, and metric-component row to that same source.

Project $\mathcal{D}$ through the $\mathsf{Distribution}^{\uparrow}$ pushforward carrier and classifier. The target measurable-event rows and RealUp endpoint rows are therefore read from Definition 186.1 and Definition 186.2.

The score and expectation ledger λ has no independent probability carrier. It may use the pushforward row of Definition 186.3 as source data, but any probability-space, measurable-preimage, or measure-row information is accessed only through the displayed $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ packet. □

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier distribution source obligation`

Theorem 228.3 (FisherInfoUp score row carrier obligation)**.** *For an accepted* $\mathsf{FisherInfo}^{\uparrow}$ *score packet, the parameter row* θ*, score row* s*, and squared-score expectation row* η *form a finite* BHist *score-expectation spine over the same* $\mathsf{Distribution}^{\uparrow}$ *source. The score row is accepted only as a history over the displayed parameter and distribution rows, while the expectation row is accepted only as the* Cont *readback of the squared score through the source distribution ledger.*

Proof. Project the parameter, score, expectation, and ledger fields from Definition 228.1. The parameter row θ supplies the displayed family index, and the score row s is carried over that index rather than over an external coordinate object.

Next read the finite continuation ledger λ. It records the passage from the displayed distribution endpoint and parameter row to the score row, and then from the score row to the squared-score expectation row η.

Thus the carrier admits the score and expectation rows only as a connected BHist spine. Different score representatives or numerical expectation displays may appear as ledger witnesses, but the public row is the carried endpoint η together with its source and provenance. □

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier score row carrier obligation`

Theorem 228.4 (FisherInfoUp RiemannianMetric comparison obligation)**.** *The metric-component row* g *of an accepted* $\mathsf{FisherInfo}^{\uparrow}$ *score packet is consumable by the public* $\mathsf{RiemannianMetric}^{\uparrow}$ *source-fibre surface. Under this consumption, the squared-score expectation row* η *supplies the displayed metric endpoint on the parameter fibre, while the fibre carrier, classifier stability, and positivity or chart-compatibility obligations are read from Definition 133.4.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier riemannianmetric comparison obligation`

Proof. Project the metric-component row g from the Fisher-information packet. By construction it is the row presented through $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$, so its base point is the parameter row and its fibre endpoint is the squared-score expectation row.

Apply the public $\mathsf{RiemannianMetric}^{\uparrow}$ source-fibre surface of Definition 133.4. That surface provides the manifold point row, tangent-fibre row, fibrewise inner-product row, and classifier transport needed to read g as a metric endpoint.

The comparison is therefore between the Fisher-information expectation endpoint and the displayed Riemannian-metric fibre endpoint. It is not a claim that a host coordinate formula, a differentiability structure, or an external metric tensor has been exported by the $\mathsf{FisherInfo}^{\uparrow}$ carrier. □

Theorem 228.5 (FisherInfoUp distribution score row coverage)**.** *Every accepted packet of Definition 228.1 projects exactly the* $\mathsf{Distribution}^{\uparrow}$ *family row* $\mathcal{D}$*, the score-observable row* s*, the expectation ledger* η*, and the shared provenance row* ρ*. These projections are the complete public distribution-side rows of the* $\mathsf{FisherInfo}^{\uparrow}$ *carrier: no host density, ambient probability law, or analytic sample-space object is part of the accepted packet.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier distribution score row coverage`

Proof. Project the accepted eight-tuple onto its distribution-side coordinates. The first projection is the carried $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ packet $\mathcal{D}$, so the family row is read through the pushforward carrier and probability-package rows already exposed by $\mathsf{Distribution}^{\uparrow}$.

The score observable is the displayed row s over the same parameter row θ. It is not reconstructed from a host density; it is one of the accepted BHist coordinates.

The expectation row η is read only through the carried $\mathsf{Distribution}^{\uparrow}$ endpoint and the finite Cont ledger λ. The provenance row ρ is shared by all displayed coordinates, so repacking the four projections gives exactly the distribution family, score, expectation, and provenance surface stated in the theorem. □

Theorem 228.6 (FisherInfoUp Riemannian metric row transport)**.** *Let $(\mathcal{D}, \mathcal{G}, \theta, s, \eta, g, \rho, \lambda)$ and $(\mathcal{D}', \mathcal{G}', \theta', s', \eta', g', \rho', \lambda')$ be accepted* $\mathsf{FisherInfo}^{\uparrow}$ *score packets. If $\mathcal{D}$ transports to $\mathcal{D}'$ by* hsame *and the* $\mathsf{Distribution}^{\uparrow}$ *classifier, and s transports to s' by the displayed score-row classifier over the transported parameter row, then the metric component g transports to g' through the* $\mathsf{RiemannianMetric}^{\uparrow}$ *classifier carried by $\mathcal{G}$ and $\mathcal{G}'$.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoScorePacket metric component transport`

Proof. Use the carried $\mathsf{Distribution}^{\uparrow}$ classifier first. It transports the distribution endpoint used by the expectation row from $\mathcal{D}$ to $\mathcal{D}'$, and the score-row hypothesis transports the displayed score observable from s to s' over the transported parameter row.

The expectation endpoint is therefore read at the transported pair of distribution and score rows. Because η and η' are ledger rows in the accepted packets, the finite Cont ledgers λ and λ' record only this distribution-to-expectation readback.

It remains to compare the displayed metric components. The metric coordinate is carried by the $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$ packets $\mathcal{G}$ and $\mathcal{G}'$. Applying the RiemannianMetricUp classifier-stability row transports the fibrewise metric endpoint, so the transported expectation component is classified as g'. □

Theorem 228.7 (FisherInfoUp expectation ledger exactness)**.** *In an accepted* $\mathsf{FisherInfo}^{\uparrow}$ *score packet, the expectation of the squared score is hidden only by the displayed* $\mathsf{Distribution}^{\uparrow}$ *pushforward ledger, the* $\mathsf{Real}^{\uparrow}$ *classifier endpoint, and the finite* Cont *readback λ. No integration primitive, host density, or analytic probability space is a public coordinate of the carrier.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier expectation ledger exactness`

Proof. Inspect the carrier coordinates of Definition 228.1. The only distribution-side source is $\mathcal{D}$, and the only observable-side source is the displayed score row s over θ.

The expectation coordinate η is the public endpoint of reading the squared score through the $\mathsf{Distribution}^{\uparrow}$ pushforward ledger. Its endpoint comparison is a $\mathsf{Real}^{\uparrow}$ classifier comparison, and its construction is recorded by the finite Cont ledger λ.

Every public endpoint of the carrier is therefore one of the displayed rows $\mathcal{D}$, $\mathcal{G}$, θ, s, η, g, ρ, or the finite ledger λ. Since no coordinate names an integration primitive, host density, or external probability object, those objects are not part of the public carrier. □

Theorem 228.8 (FisherInfoUp Riemannian metric component row)**.** *In every accepted packet of Definition 228.1, the endpoint of the expected squared score η is exactly the metric-component* BHist *row g consumed by the inherited* $\mathsf{RiemannianMetric}^{\uparrow}$ *fibre surface $\mathcal{G}$. Thus the Fisher information component is not an external coordinate metric; it is the displayed expectation endpoint repacked as the metric fibre component.*

Proof. Project the accepted packet onto the expectation row η and the metric-component row g. By the carrier definition, both are displayed BHist rows in the same finite Cont ledger and share the provenance row ρ.

The distribution source obligation identifies η as the expectation endpoint of s^2 through the carried Distribution$^\uparrow$ probability row. The RiemannianMetricUp dependency $\mathcal{G}$ consumes a displayed metric fibre component rather than a host coordinate tensor.

The accepted FisherInfoUp packet equates these two displayed endpoints inside the finite ledger: the expected squared score endpoint is repacked as g. Consequently the inherited fibre surface reads the same BHist row that the expectation source row produces. □

Theorem 228.9 (FisherInfoUp score expectation metric confluence)**.** *For every accepted* FisherInfo$^\uparrow$ *score packet* $(\mathcal{D}, \mathcal{G}, \theta, s, \eta, g, \rho, \lambda)$*, the* Distribution$^\uparrow$ *pushforward row, the score row, the expectation row, and the* RiemannianMetric$^\uparrow$ *metric row share one* Pkg *provenance row* ρ *and one finite* Cont *ledger* λ*. Under that shared ledger, the public metric component* g *is exactly the displayed expected squared-score endpoint* η *consumed through* $\mathcal{G}$*.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier score expectation metric confluence`

Proof. Apply Definition 228.2. It projects the packet to the Distribution$^\uparrow$ source row, the parameter row θ, the score row s, and the expectation endpoint η.

Apply Definition 228.7. The expectation endpoint is read only through the displayed pushforward and the finite Cont route λ, with provenance recorded by the same row ρ.

Apply Definition 228.4 to read the metric dependency $\mathcal{G}$, and then Definition 228.8 to identify the metric component g with the expected squared-score endpoint. Repacking these four rows gives the confluence statement: the probability face and metric face consume the same displayed endpoint under the same ledger. □

Theorem 228.10 (FisherInfoUp classifier transport obligation)**.** *Let* $\mathcal{I}$ *and* $\mathcal{I}'$ *be accepted* FisherInfo$^\uparrow$ *score packets whose source distribution, parameter, score, squared-score expectation, and metric-component rows are related by the componentwise* hsame *witnesses available to their finite ledgers. Then the* FisherInfo$^\uparrow$ *classifier transports the score-expectation spine of* $\mathcal{I}$ *to the score-expectation spine of* $\mathcal{I}'$ *and preserves the target* RiemannianMetric$^\uparrow$ *fibre comparison.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier classifier transport obligation`

Proof. Start with the two accepted carrier tuples and project their componentwise classifier witnesses. The source distribution rows are transported by the Distribution$^\uparrow$ classifier, while the parameter, score, expectation, and metric-component rows are transported by the listed hsame comparisons.

Transport the finite continuation ledger component by component. The source-to-score step moves along the transported distribution and parameter rows; the score-to-expectation step moves along the transported score row; the expectation-to-metric-component step moves along the transported metric row.

Finally apply the RiemannianMetric$^\uparrow$ classifier stability supplied by Definition 133.4. The transported expectation endpoint is still the metric endpoint of the transported metric-component row, so the public FisherInfo$^\uparrow$ classifier is preserved. □

228.2 The certificate

Theorem 228.11 (FisherInfoUp classifier transport row)**.** *Let* $(\mathcal{D}, \mathcal{G}, \theta, s, \eta, g, \rho, \lambda)$ *and* $(\mathcal{D}', \mathcal{G}', \theta', s', \eta', g', \rho', \lambda')$ *be accepted* $\mathsf{FisherInfo}^{\uparrow}$ *score packets. If* hsame *transport carries the distribution family row* $\mathcal{D}$ *to* $\mathcal{D}'$*, the parameter row* θ *to* θ'*, the score row* s *to* s'*, and the metric fibre row* $\mathcal{G}$ *to* $\mathcal{G}'$*, then it transports the FisherInfoUp metric endpoint componentwise from* g *to* g'*.*

Proof. Use the carried $\mathsf{Distribution}^{\uparrow}$ classifier first. It transports the pushforward and probability row that supplies the expectation source from $\mathcal{D}$ to $\mathcal{D}'$, while the parameter and score hypotheses transport the displayed score-function row from s to s' over θ'.

The expected squared score endpoint therefore transports from η to η' along the same finite Cont readback pattern. No analytic integration or derivative primitive enters this step because both endpoints are displayed BHist rows in accepted packets.

Apply the metric component row theorem on both packets. It identifies η with g and η' with g' as the metric fibre components consumed by $\mathcal{G}$ and $\mathcal{G}'$. The $\mathsf{RiemannianMetric}^{\uparrow}$ classifier then transports the component row, giving the stated componentwise FisherInfoUp metric endpoint transport. □

Theorem 228.12 (FisherInfoUp public certificate surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ *obligation surface consists exactly of the score packet carrier of Definition 228.1, the distribution source row of Definition 228.2, the metric component row of Definition 228.8, the classifier transport row of Definition 228.11, the finite* Cont *ledger* λ*, the shared* Pkg *ledger* ρ*, the exactness boundary excluding host density, integral, derivative, and coordinate metric primitives, and an absent bridge row. These data form the public* $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ *surface over the dependency certificates* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier public certificate surface`

Proof. The carrier component is exactly the displayed eight-coordinate packet of Definition 228.1. Its accepted coordinates expose the $\mathsf{Distribution}^{\uparrow}$ dependency, the $\mathsf{RiemannianMetric}^{\uparrow}$ dependency, the parameter history, the score-function history, the squared-score expectation history, the metric-component history, the shared Pkg provenance, and the finite Cont ledger.

The distribution-side public rows are supplied by Definition 228.2. The metric endpoint identified from the expected squared score is supplied by Definition 228.8.

The classifier transport row is supplied by Definition 228.11. Repacking these rows with the finite Cont ledger, the Pkg provenance ledger, the explicit exactness boundary, and the absent bridge row gives the stated ten-field $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ obligation surface. □

Theorem 228.13 (FisherInfoUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ *obligation surface consists exactly of the* BHist *score carrier of Definition 228.1, the distribution score row coverage of Definition 228.5, the Riemannian metric transport row of Definition 228.6, the expectation ledger exactness row of Definition 228.7, the finite* Cont *ledger* λ*, and the shared dependency provenance* ρ *from* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^{\uparrow}}$*. It exports no host density, analytic integration primitive, external probability law, or host Riemannian metric object.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier semantic name certificate`

Proof. The carrier component is exactly the eight-coordinate packet of Definition 228.1. Its accepted coordinates expose the $\mathsf{Distribution}^{\uparrow}$ dependency, the $\mathsf{RiemannianMetric}^{\uparrow}$ dependency, the

parameter row, the score row, the expectation row, the metric-component row, the shared Pkg provenance, and the finite Cont ledger.

The distribution-side public rows are supplied by Definition 228.5. The classifier transport row is supplied by Definition 228.6, which composes hsame transport, DistributionUp endpoint transport, score-row transport, and the RiemannianMetricUp classifier.

The exactness boundary is supplied by Definition 228.7. Repacking these rows under the ten-field NameCert pattern yields the stated $\mathsf{NameCert}_{\mathsf{FisherInfo}^\uparrow}$ surface, and the same projections exclude every host density, analytic integration primitive, external probability law, and host metric object from the public certificate. □

Theorem 228.14 (FisherInfoUp ledger exactness boundary)**.** *The public* $\mathsf{NameCert}_{\mathsf{FisherInfo}^\uparrow}$ *obligation surface consists exactly of the* $\mathsf{Distribution}^\uparrow$ *source row, the* $\mathsf{RiemannianMetric}^\uparrow$ *metric packet, the parameter row, the score row, the squared-score expectation row, the metric-component row, the* Pkg *provenance row, the componentwise* hsame *classifier witnesses used by the finite ledger, and the finite* Cont *ledger of Definition 228.1. It exports no additional row for score choice, expectation approximation, coordinate chart, host differentiability, ambient statistical model equality, or standard bridge.*

Lean4 ✓ `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier ledger exactness boundary`

Proof. Soundness is by projection from the accepted packet. Each accepted public row is one of the named fields of Definition 228.1: distribution source, metric packet, parameter, score, expectation, metric-component, provenance, or ledger, together with the hsame comparisons admitted by that ledger.

Completeness is the reverse packaging. Given exactly those rows, the eight-coordinate packet is reconstructed as a finite BHist packet whose score and expectation endpoints are connected by the displayed Cont ledger and whose metric endpoint is presented to $\mathsf{NameCert}_{\mathsf{RiemannianMetric}^\uparrow}$.

The exclusions follow from the same exactness reading. Score representatives, expectation approximations, coordinate choices, and differentiability evidence can justify ledger steps, but no such datum is a separate public carrier equality or a row available to downstream consumers outside the listed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FisherInfo}^\uparrow}$.

Constructive story: A Fisher-information packet is carried as eight displayed rows: a certified distribution family, a certified Riemannian-metric packet, a parameter row, a score row, the expected squared-score endpoint, the metric-component endpoint, shared package provenance, and a finite continuation ledger tying the rows together. The carrier accepts exactly those BHist coordinates when the score and expectation are read through the $\mathsf{Distribution}^\uparrow$ pushforward surface and the metric component is consumed through $\mathsf{RiemannianMetric}^\uparrow$; host densities, ambient probability spaces, integration primitives, score choices, and coordinate metric tensors remain outside the carrier. Two packets are classified by transporting the distribution, parameter, score, expectation, and metric rows through componentwise hsame, the $\mathsf{Distribution}^\uparrow$ classifier, and the $\mathsf{RiemannianMetric}^\uparrow$ fibre classifier, so sameness is a ledgered endpoint comparison rather than equality of analytic formulas. The source covers the distribution, score, expectation, metric, provenance, and finite continuation rows, the pattern records the confluence between the expected squared score and the metric fibre component, and the ledger keeps the pushforward, RealUp endpoint, provenance, and continuation readback visible. The five name-certificate obligations follow because the dependency certificates supply admitted distribution and metric endpoints, hsame and their classifiers are equivalence-like transport mechanisms, and the expectation-to-metric row repacks transported endpoints back into an accepted score packet. A history h lies in the signature gap of a $\mathsf{FisherInfo}^\uparrow$ token p when the source-domain accepts this packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names the answer as p; unfolded, h presents the distribution family, score, expected squared-score row, metric component, shared provenance, and finite ledger encoded by p. The chapter registers this score-expectation metric construction under the name $\mathsf{FisherInfo}^\uparrow$; Fisher information is the displayed expectation endpoint reused as a metric fibre row, with no host density, analytic integral, external probability law, coordinate metric object, or standard bridge introduced.

Theory closure: TheoryGate(NameCert$_{\mathsf{FisherInfo}\uparrow}$, maturePackageClosed)

Scope closed: The public boundary exports exactly the distribution, metric, parameter, score, expectation, metric-component, provenance, and finite ledger rows displayed in the FisherInfoUp certificate surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{FisherInfo}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.FisherInfoUp.FisherInfoBHistScoreCarrier\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not close host density, analytic integration, coordinate metric, standard bridge, or completeness claims for Fisher information.

Upgrade path: No further paper theory-axis upgrade is pending for the finite score-expectation metric packet.

Chapter 229

A Concrete Naming Certificate for $\mathsf{StatManifold}^{\uparrow}$

The statistical-manifold interface $\mathsf{StatManifold}^{\uparrow}$ packages a parameter-space manifold whose points label probability distributions, equipped with the Fisher information metric and dual affine connections. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ (chapter 228).

229.1 Carrier definition

Theorem 229.1 (StatManifoldUp carrier classifier obligation)**.** *A carried* $\mathsf{StatManifold}^{\uparrow}$ *packet is a* BHist *row tuple*

$$(\mathcal{M}, \mathcal{I}, \theta, p, g, \nabla, \nabla^*, \rho, \lambda).$$

Here $\mathcal{M}$ *is the displayed* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *parameter carrier,* $\mathcal{I}$ *is the displayed* $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$ *score and metric packet,* θ *and* p *are parameter and distribution-endpoint rows,* g *is the Fisher metric row consumed from* $\mathcal{I}$*,* ∇ *and* ∇^* *are the dual affine connection ledger rows,* ρ *is the shared* Pkg *provenance, and* λ *is the finite* Cont *ledger joining these displayed rows. The classifier is the componentwise* hsame *transport that preserves the* $\mathsf{Manifold}^{\uparrow}$ *carrier row, the* $\mathsf{FisherInfo}^{\uparrow}$ *metric row, both connection ledgers, and the shared provenance.*

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldCarrier classifier transport`

Proof. Unpack the displayed tuple as a finite list of BHist rows. The parameter carrier is not an abstract transported manifold; it is the concrete $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ source row $\mathcal{M}$.

Read the Fisher metric coordinate through $\mathcal{I}$. The dependency chapter chapter 228 supplies the score, expectation, metric-component, and provenance rows, so g is accepted only as the FisherInfoUp metric endpoint.

Finally inspect the classifier. Each coordinate moves by the visible hsame comparison and the finite Cont route already stored in λ. Repacking the transported coordinates gives another accepted tuple with the same dependency sources and the same public row inventory. □

Theorem 229.2 (StatManifoldUp FisherInfo and Manifold source scope)**.** *Every accepted* $\mathsf{StatManifold}^{\uparrow}$ *packet projects exactly two dependency sources: the parameter carrier* $\mathcal{M}$ *from* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ *and the Fisher metric packet* $\mathcal{I}$ *from* $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$*. The parameter point row* θ*, distribution endpoint row* p*, and Fisher metric row* g *are read through these two dependency certificates, not through a host smooth manifold, ambient probability family, or external metric tensor.*

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldFisherManifoldSource scope transport`

Proof. Project the first two coordinates of the packet. The row $\mathcal{M}$ is the only carrier for parameter-space chart and tangent information, and chapter 96 is the only source chapter used for that carrier.

Project the Fisher-information coordinate $\mathcal{I}$. By chapter 228, the metric component is a displayed expectation endpoint over the distribution family, so g is inherited as a BHist metric row.

The remaining parameter and distribution rows are ledger coordinates tied to these two sources by ρ and λ. Since no coordinate of the tuple names a host manifold, probability family, or metric tensor, those objects are not exported by the source scope. □

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldSourceScope dependency boundary`
Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldCarrier dependency source scope transport`

Theorem 229.3 (StatManifoldUp ledger exactness)**.** *The public ledger of a* $\mathsf{StatManifold}^{\uparrow}$ *packet consists exactly of the parameter carrier row* $\mathcal{M}$*, the Fisher packet* $\mathcal{I}$*, the parameter and distribution endpoint rows* θ, p*, the Fisher metric row* g*, the dual affine connection rows* ∇, ∇^**, the shared* Pkg *row* ρ*, and the finite* Cont *ledger* λ*. No coordinate-level Christoffel symbol, density formula, or ambient affine connection object is available outside this list.*

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldBHistPacket ledger exactness`
Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldCarrierPacket ledger exactness`

Proof. Inspect the coordinates displayed in Definition 229.1. They already list the complete accepted packet, including both dependency certificates, the metric row, the two connection rows, and the shared provenance.

The connection data are ledger rows rather than host formulas. They can be transported only along the carried Cont route and compared only by hsame against the same dependency sources.

Conversely, every public readback from the packet must project one of the listed rows. Repacking these projections gives exactly the same finite ledger, so there is no hidden coordinate connection or analytic density primitive to read. □

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldDependencyLedgerPacket public projection`

Theorem 229.4 (StatManifoldUp NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$ *obligation surface is generated by the carrier-classifier row of Definition 229.1, the dependency source scope of Definition 229.2, and the ledger exactness row of Definition 229.3. Its carrier witness is the displayed packet, its classifier obligations are reflexivity, symmetry, and transitivity of the componentwise* hsame *row, and its stability obligation is preservation of the same packet shape under the finite* Cont *transports recorded by* λ*.*

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldPacket namecert obligation surface`

Proof. Use the carrier theorem to obtain the displayed packet and its classifier. The carrier witness is the packet itself; the classifier relation is the componentwise hsame relation over the visible coordinates.

Reflexivity, symmetry, and transitivity are inherited coordinate by coordinate from hsame while the dependency rows remain those of $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$. The shared Pkg row ensures that the transported coordinates still belong to the same certificate packet.

Apply the ledger exactness theorem to close the public surface. It says that no extra carrier or classifier field can be read outside the tuple, so the displayed packet supplies the whole NameCert obligation surface. □

Theorem 229.5 (StatManifoldUp downstream consumption boundary)**.** *Any downstream consumer of* $\mathsf{StatManifold}^{\uparrow}$ *can read only the* BHist *parameter carrier, Fisher metric, dual affine connection, provenance, and finite continuation rows enumerated in Definition 229.3. In particular, consumers such as information-geometric curvature, statistical geodesic, or divergence interfaces must factor their inputs through the displayed* NameCert *rows and cannot demand an external smooth model or analytic probability space.*

Lean4 ✓ `BEDC.Derived.StatManifoldUp.StatManifoldCarrier downstream consumption endpoint transport`

Proof. Start from an accepted consumer readback. By the NameCert obligation surface, the readback first factors through the carried packet and the componentwise classifier of Definition 229.1.

Apply the source-scope theorem. It restricts all manifold and metric inputs to the two dependency certificates $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$.

Apply ledger exactness. The remaining consumable rows are precisely $\theta, p, g, \nabla, \nabla^*, \rho, \lambda$ together with the two dependency packets, so every downstream interface must consume those rows and no others. □

229.2 The certificate

Theorem 229.6 (StatManifoldUp public certificate export)**.** *The carrier-classifier obligation of Definition 229.1, the FisherInfo/Manifold source scope of Definition 229.2, the ledger exactness row of Definition 229.3, the* $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$ *obligation surface of Definition 229.4, and the downstream consumption boundary of Definition 229.5 together export the public* $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$ *surface. The exported rows are exactly the* BHist *parameter carrier, Fisher metric packet, dual affine connection ledgers,* Pkg *provenance, and finite* Cont *ledger, with no host manifold or analytic probability primitive.*

Proof. First project the carrier-classifier obligation and the source-scope row. They identify the public packet as the displayed parameter carrier from $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ and the Fisher metric packet from $\mathsf{NameCert}_{\mathsf{FisherInfo}^{\uparrow}}$.

Next apply ledger exactness. It says that the remaining public rows are the parameter endpoint, distribution endpoint, Fisher metric, dual affine connection ledgers, shared provenance, and finite continuation ledger, with no extra coordinate-level analytic data.

Finally use the NameCert obligation surface and the downstream consumption boundary. These two rows repack the displayed carrier, classifier, stability, ledger, and consumer-facing rows as the public $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$ surface, and they exclude any host manifold or analytic probability primitive outside the listed BHist packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation surface names the carrier-classifier row, Fisher and manifold source scope, ledger exactness row, $\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$ obligation row, downstream boundary, and public export row already displayed in this chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StatManifold}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.StatManifoldUp.StatManifoldPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Constructive story: The statistical-manifold packet is carried by a finite BHist ledger whose source rows come from $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{FisherInfo}^{\uparrow}$, whose visible rows expose parameter and distribution endpoints, Fisher metric data, dual affine connection data, package provenance, and continuation evidence, and whose classifier transports only those displayed rows through hsame and Cont.

Not claimed: This chapter does not close host manifolds, analytic probability primitives, coordinate Christoffel symbols, or standard bridges.

Upgrade path: No further paper theory-axis upgrade is recorded for this chapter; bridge verification remains on the formal axis.

Chapter 230

A Concrete Naming Certificate for $\mathsf{Estimator}^{\uparrow}$

The estimator interface $\mathsf{Estimator}^{\uparrow}$ packages a sequence of measurable functions of i.i.d. samples taking values in a parameter space, with consistency, bias, and variance certificates. It depends on $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$ (chapter 185) and $\mathsf{NameCert}_{\mathsf{Independence}^{\uparrow}}$ (chapter 187).

230.1 Carrier definition

Definition 230.1 (EstimatorUp BHist source surface)**.** *The* $\mathsf{Estimator}^{\uparrow}$ BHist *source surface consists of displayed* $\mathsf{RandomVar}^{\uparrow}$ *sample rows, an* $\mathsf{Independence}^{\uparrow}$ *sample-family row, estimator endpoint rows, bias and variance witness rows, componentwise* hsame *transport, finite* Cont *readback ledgers, and* Pkg *provenance naming the source certificates. A carried estimator packet is accepted only through these rows: the sample rows expose the random-variable inputs, the independence row records the finite sample-family boundary, the endpoint rows name the estimator outputs, and the bias and variance witnesses are read by finite continuation ledgers from the same packet.*

Lean4 (definition) `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface`

230.2 The certificate

Theorem 230.2 (EstimatorUp source obligation)**.** *Every* $\mathsf{Estimator}^{\uparrow}$ BHist *source packet is sourced exactly by the* $\mathsf{RandomVar}^{\uparrow}$ *sample rows, the* $\mathsf{Independence}^{\uparrow}$ *sample-family row, the estimator endpoint rows, the bias and variance witness rows, finite* Cont *readback, and* Pkg *provenance of Definition 230.1. It includes no host probability-space equality, no asymptotic normality row, and no completed statistical convergence claim.*

Proof. Unpack the source surface definition. Its sample coordinates are displayed $\mathsf{RandomVar}^{\uparrow}$ rows, and its family coordinate is the single $\mathsf{Independence}^{\uparrow}$ row that relates those samples inside BHist.

The estimator endpoint, bias, and variance rows are then read by the listed finite Cont ledgers. The Pkg provenance names exactly the certificates used for those reads, so there is no separate source from an ambient probability space.

Repacking those rows gives the whole source packet. Since the definition contains no row for host probability-space equality, asymptotic normality, or completed statistical convergence, none is exported by the source obligation. □

Lean4 ✓ `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface source obligation`

Theorem 230.3 (EstimatorUp transport obligation)**.** *The classifier transport obligation for* $\mathsf{Estimator}^{\uparrow}$ *is componentwise* hsame *transport of the displayed* $\mathsf{RandomVar}^{\uparrow}$ *samples, the* $\mathsf{Independence}^{\uparrow}$ *sample-family row, the estimator endpoint rows, the bias and variance witness rows, and the finite* Cont *ledgers of Definition 230.1. Transport preserves the same* Pkg *provenance and does not identify host probability spaces, derive asymptotic normality, or complete any statistical convergence argument.*

Proof. Start with two displayed estimator packets related componentwise by hsame. The sample rows transport through their $\mathsf{RandomVar}^{\uparrow}$ certificates, while the sample-family row transports through the $\mathsf{Independence}^{\uparrow}$ certificate.

Next transport the endpoint, bias, and variance rows along the listed component comparisons. The finite Cont ledgers are stable under the same endpoint replacement, so the readback packet remains a BHist estimator packet with unchanged Pkg provenance.

The transported witness is therefore still built only from the source surface. No step introduces host probability-space equality, asymptotic normality, or a completed convergence theorem. □

Lean4 ✓ `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface transport obligation`

Theorem 230.4 (EstimatorUp finite Cont-readback determinacy)**.** *Let* E_0 *and* E_1 *be accepted* $\mathsf{Estimator}^{\uparrow}$ BHist *packets. Suppose they display the same* $\mathsf{RandomVar}^{\uparrow}$ *sample rows, the same* $\mathsf{Independence}^{\uparrow}$ *sample-family row, matching estimator endpoint, bias, and variance witness rows, componentwise* hsame *transport, the same finite* Cont *ledgers, and the same* Pkg *provenance from Definition 230.1. Then every public estimator readback row is identical after the displayed transports. No asymptotic normality row and no completed convergence row can mediate this readback.*

Proof. Project both accepted packets through Definition 230.2. The only source rows exposed are the displayed sample rows, sample-family row, endpoint rows, bias and variance witness rows, finite continuation ledgers, and package provenance listed in Definition 230.1.

Next apply Definition 230.3 componentwise. The matching hsame transports identify the sample, independence, endpoint, bias, and variance rows while preserving the same finite Cont ledgers and the same Pkg provenance.

Compare the finite continuation ledgers row-by-row. Since the ledgers are the same and their endpoints have already been transported into the same displayed rows, every public readback entry of E_0 agrees with the corresponding entry of E_1.

Repack the common rows using Definition 230.6. The public $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ surface has no field for asymptotic normality or completed statistical convergence, so neither can act as an intermediate readback witness. □

Lean4 ✓ `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface finite cont readback determinacy`

Theorem 230.5 (EstimatorUp bias and variance ledger exclusion)**.** *Let* E *be an accepted* $\mathsf{Estimator}^{\uparrow}$ BHist *packet. If* E *exposes a public bias witness row or a public variance witness row, that row factors through the bias or variance coordinate of Definition 230.1, the finite* Cont *readback ledgers displayed there, and the same* Pkg *provenance naming the source certificates. No host probability-space equality, asymptotic normality row, completed convergence theorem, or external statistical model supplies a public bias or variance witness for* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$.

Proof. Project the accepted packet through Definition 230.2. The projection lists the displayed sample rows, the sample-family row, estimator endpoint rows, bias and variance witness rows, finite continuation ledgers, and package provenance as the complete source surface for E.

Transport this projected packet by Definition 230.3. Componentwise hsame movement keeps the bias and variance coordinates inside the same displayed packet and preserves the finite Cont ledgers together with their Pkg provenance.

Now compare public readback rows using Definition 230.4. Any exposed bias or variance witness row is determined by the displayed finite ledger row after the listed transports, so it cannot be replaced by a row outside the source surface.

The source surface has no coordinate for host probability-space equality, asymptotic normality, completed convergence, or an external statistical model. Hence none of those data can enter the public $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ witness boundary. □

Lean4 ✓ `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface bias variance ledger exclusion`

Theorem 230.6 (EstimatorUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ *obligation surface is precisely the* BHist *source surface of Definition 230.1, the source obligation of Definition 230.2, and the transport obligation of Definition 230.3. Its public reads expose only sample, independence, endpoint, bias, variance,* hsame*,* Cont*, and* Pkg *rows, and exclude host probability-space equality, asymptotic normality, and completed statistical convergence.*

Proof. The carrier row is the displayed source surface. By Definition 230.2, every accepted estimator packet unpacks into the named sample, independence, endpoint, bias, variance, continuation, and provenance rows.

The classifier and stability rows are supplied by Definition 230.3. Componentwise hsame moves the visible rows and preserves the finite ledgers, so the transported packet reassembles under the same $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ interface.

Combining these two obligations gives the exact public surface. The surface is anchored in BHist and contains no field for host probability-space equality, asymptotic normality, or completed statistical convergence. □

Lean4 ✓ `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface namecert obligation surface`

Closure status for $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$.

Constructive story: An estimator row is seeded by consuming $\mathsf{RandomVar}^{\uparrow}$ and $\mathsf{Independence}^{\uparrow}$ source packets, then displaying the sample rows, sample-family row, estimator endpoint rows, bias and variance witness rows, finite continuation ledgers, and shared Pkg provenance in one BHist source surface. The checked obligation theorem records this source surface as the visible $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ boundary; host probability equality, asymptotic normality, completed convergence, and external statistical-model data are not fields of the seed packet.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, maturePackageClosed)

Scope closed: The exported $\mathsf{Estimator}^{\uparrow}$ public surface binds the estimator source rows, cited $\mathsf{FisherInfo}^{\uparrow}$ and $\mathsf{StatManifold}^{\uparrow}$ dependency rows, classifier rows, hsame transport, Cont readback, and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.EstimatorUp.EstimatorBHistSourceSurface\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: This chapter does not close host probability equality, asymptotic normality, or completed convergence.

Upgrade path: No further paper theory-axis grade is recorded for the displayed estimator schema.

Chapter 231

A Concrete Naming Certificate for $\mathsf{HypothesisTest}^{\uparrow}$

The hypothesis-testing interface $\mathsf{HypothesisTest}^{\uparrow}$ packages a decision procedure between null and alternative distributional hypotheses certified by Type I and Type II error bounds. It depends on $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186) and $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ (chapter 184).

The null and alternative rows consumed from $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186) supply the classifier surface for hypothesis alternatives, while the event rows consumed from $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ (chapter 184) supply the reject and accept-null event surfaces read by the Type I and Type II ledgers. The $\mathsf{HypothesisTest}^{\uparrow}$ packet adds only the decision row, threshold comparison row, and finite error-budget rows; it does not add probability completion, asymptotic optimality, or external statistical semantics.

231.1 Carrier definition

Definition 231.1 (HypothesisTestUp decision carrier)**.** *A* $\mathsf{HypothesisTest}^{\uparrow}$ *decision carrier is a finite* BHist *decision row over a displayed null hypothesis, a displayed alternative hypothesis, and a finite threshold ledger supplied by* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$*. The carrier records the sample row, the statistic row, the threshold comparison row, the accept/reject decision row, and the finite Type I and Type II error-budget rows. It does not contain an asymptotic limit, optimality principle, or classical probability-completion field.*

231.2 The certificate

Theorem 231.2 (HypothesisTestUp Type I error ledger)**.** *For a* $\mathsf{HypothesisTest}^{\uparrow}$ *carrier, the Type I error row is a visible* $\mathsf{ProbSpace}^{\uparrow}/\mathsf{Distribution}^{\uparrow}$ *ledger entry: under the displayed null distribution row, the reject-decision event is bounded by the finite threshold budget recorded in Definition 231.1. The theorem asserts only this finite ledger bound and no optimality or asymptotic claim.*

Lean4 ✓ `BEDC.Derived.HypothesisUp.HypothesisDecisionCarrier type one error ledger`

Proof. Project the null distribution row, sample row, statistic row, and threshold comparison row from the decision carrier. The reject-decision event is a finite BHist event row in the consumed $\mathsf{ProbSpace}^{\uparrow}$ surface.

The Type I budget is one ledger coordinate of the same carrier. Reading the event through the displayed $\mathsf{Distribution}^{\uparrow}$ row gives exactly the recorded finite bound, with no appeal to external completeness or limiting frequency data. □

Lean4 ✓ `BEDC.Derived.HypothesisUp.HypothesisTestDecisionCarrier type one error ledger`

Theorem 231.3 (HypothesisTestUp Type II error ledger)**.** *For a* $\mathsf{HypothesisTest}^{\uparrow}$ *carrier, the Type II error row is a visible* $\mathsf{ProbSpace}^{\uparrow}/\mathsf{Distribution}^{\uparrow}$ *ledger entry: under the displayed alternative distribution row, the accept-null decision event is bounded by the finite threshold budget recorded in Definition 231.1. The theorem asserts only this finite ledger bound and no power optimality, asymptotic consistency, or classical probability-completion claim.*

Proof. Project the alternative distribution row together with the sample, statistic, threshold, and accept-null decision rows. These are all finite rows in the decision carrier, so the consumed $\mathsf{ProbSpace}^{\uparrow}$ event is the displayed accept-null event.

The Type II budget row is recorded beside that event in the finite threshold ledger. Evaluation through the displayed $\mathsf{Distribution}^{\uparrow}$ row returns the listed bound and nothing stronger than the finite BEDC evidence row. □

Lean4 ✓ `BEDC.Derived.HypothesisUp.HypothesisTestDecisionCarrier type two error ledger`

Theorem 231.4 (HypothesisTestUp distribution consumer boundary)**.** *The* $\mathsf{HypothesisTest}^{\uparrow}$ *consumer boundary reads only the displayed* $\mathsf{Distribution}^{\uparrow}$ *null and alternative rows, the* $\mathsf{ProbSpace}^{\uparrow}$ *event rows, the sample and statistic rows, and the finite threshold ledger. It exports no statistical optimality theorem, asymptotic decision principle, or classical probability-completeness row.*

Lean4 ✓ `BEDC.Derived.HypothesisTestUp.HypothesisTestDecisionCarrier distribution consumer boundary`

Proof. Unpack the consumer side of the decision carrier. Every probability statement is routed through a displayed $\mathsf{ProbSpace}^{\uparrow}$ event row and a displayed $\mathsf{Distribution}^{\uparrow}$ hypothesis row, and every decision statement is routed through the sample, statistic, and threshold comparison rows.

Constructor inversion on the carrier leaves no additional consumer coordinate. The boundary can therefore repack the finite error ledgers, but it cannot export an optimality, asymptotic, or completion field because no such field is present in the carrier. □

Lean4 ✓ `BEDC.Derived.HypothesisUp.HypothesisTestDecisionCarrier distribution boundary`

Theorem 231.5 (HypothesisTestUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{HypothesisTest}^{\uparrow}}$ *obligation surface is the repacking of the decision carrier, the null and alternative distribution classifiers, decision stability under* hsame *transport, Type I and Type II error ledger exactness, and the finite threshold ledger. Its claims remain finite-row* BHist *claims over* $\mathsf{Distribution}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$*.*

Proof. First take the carrier from Definition 231.1. It supplies the hypotheses, sample, statistic, threshold comparison, decision row, and two finite error-budget rows.

Next use Definition 231.2 and Definition 231.3 for the two error exactness fields. The distribution consumer boundary of Definition 231.4 fixes the allowed source rows and prevents external probability-completion data from entering the certificate.

Finally repack these rows with the visible classifier and hsame transport fields. Since every component is a finite row from the displayed decision carrier or its two consumed distribution certificates, the resulting $\mathsf{NameCert}_{\mathsf{HypothesisTest}^{\uparrow}}$ surface contains no statistical optimality, asymptotic, or classical completeness assertion. □

Lean4 ✓ `BEDC.Derived.HypothesisUp.HypothesisTestDecisionCarrier namecert obligation surface`

Theorem 231.6 (HypothesisTestUp finite-risk boundary)**.** *For every accepted* $\mathsf{HypothesisTest}^{\uparrow}$ *carrier, any finite risk read factors through the displayed null and alternative* $\mathsf{Distribution}^{\uparrow}$ *rows, the consumed* $\mathsf{ProbSpace}^{\uparrow}$ *event rows, the sample and statistic rows, the statistic-threshold comparison row, the decision row, and the Type I and Type II error-budget rows. The boundary exports only these finite ledgers and no asymptotic consistency, optimality principle, or probability-completion datum.*

Proof. Project the decision carrier of Definition 231.1. The carrier displays the null and alternative hypotheses, the sample row, statistic row, statistic-threshold comparison, decision row, and the two finite error-budget rows.

Apply the Type I and Type II ledger theorems, Definition 231.2 and Definition 231.3. They identify the only reject-under-null and accept-under-alternative risk reads with the displayed $\mathsf{ProbSpace}^{\uparrow}/\mathsf{Distribution}^{\uparrow}$ event ledgers.

Finally use the distribution consumer boundary and the NameCert obligation surface. Theorems 231.4 and 231.5 keep every public risk read inside the displayed decision rows, so no hidden asymptotic, optimality, or probability-completion field can be exported. □

Theorem 231.7 (HypothesisTestUp decision stability under transport)**.** *Let two accepted* $\mathsf{HypothesisTest}^{\uparrow}$ *carriers be related componentwise by* hsame *on their null and alternative* $\mathsf{Distribution}^{\uparrow}$ *rows,* $\mathsf{ProbSpace}^{\uparrow}$ *event rows, statistic-threshold ledgers, decision rows, and Type I/Type II budget rows. Then the transported carrier preserves the decision row and both error-ledger reads through the same finite* $\mathsf{NameCert}_{\mathsf{HypothesisTest}^{\uparrow}}$ *surface. The transport adds no asymptotic consistency claim, optimality datum, or probability-completion row.*

Proof. Start from the componentwise hsame relation. The carrier projection gives matching null and alternative distribution rows, event rows, statistic and threshold comparisons, decision rows, and finite Type I/Type II budget rows.

Transport the error reads through the already checked ledgers. Theorem 231.2 preserves the reject-under-null row, and Definition 231.3 preserves the accept-under-alternative row, because each is read from the displayed carrier coordinates.

Repack the transported rows using Definition 231.5. Since the NameCert surface contains only the decision carrier, visible classifiers, transport field, and finite ledgers, the transported decision cannot acquire an optimality, asymptotic, or probability-completion coordinate. □

Theorem 231.8 (HypothesisTestUp finite-risk public route)**.** *For every accepted* $\mathsf{HypothesisTest}^{\uparrow}$ *carrier, any public finite-risk read factors through the displayed* $\mathsf{Distribution}^{\uparrow}$ *null and alternative rows, the consumed* $\mathsf{ProbSpace}^{\uparrow}$ *event rows, the statistic-threshold comparison row, the decision row, the Type I/II error ledgers, and componentwise* hsame *transport. The route exports a finite decision surface over the named carrier and no statistical optimality, asymptotic consistency, probability-completion datum, or standard statistical bridge.*

Proof. Start with the finite-risk boundary of Definition 231.6. It routes every public risk read through the displayed null and alternative distributions, consumed probability events, statistic row, threshold comparison, decision row, and both finite error budgets.

Use Definition 231.7 to make the same route stable under componentwise hsame transport. The transported read still sees only the displayed carrier coordinates and the two Type I/II ledgers.

Finally repack the read through Definition 231.5. The public surface is therefore exactly the finite carrier route through $\mathsf{Distribution}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, statistic-threshold comparison, decision rows, error ledgers, and hsame transport, with no bridge or asymptotic field added. □

Closure status for $\mathsf{NameCert}_{\mathsf{HypothesisTest}^\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HypothesisTest}^\uparrow}$, maturePackageClosed)

Scope closed: The bridge-facing scope names the finite decision carrier, Type I/II ledgers, distribution consumer boundary, decision transport, and public finite-risk route over displayed DistributionUp and ProbSpaceUp rows, with the checked decision-carrier obligation surface supplying the packet boundary.

Constructive story: A $\mathsf{HypothesisTest}^\uparrow$ token names a finite decision carrier with null and alternative $\mathsf{Distribution}^\uparrow$ inputs, consumed $\mathsf{ProbSpace}^\uparrow$ event rows, a displayed statistic-threshold comparison, and two visible finite error-budget ledgers. The classifier is hsame on the decision history, transport repacks the statistic, threshold, decision row, and error ledgers through that history comparison, and the source covers only the displayed distribution, event, threshold, and finite budget rows. A history h lies in the signature gap of a $\mathsf{HypothesisTest}^\uparrow$ token p when the finite carrier accepts h, the signature relation pairs it with the decision verdict and error-budget rows, and token introduction names that finite verdict as p.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HypothesisTest}^\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.HypothesisUp.HypothesisTestDecisionCarrier\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: The chapter does not claim statistical optimality, asymptotic consistency, classical probability completion, or a standard statistical bridge.

Upgrade path: The remaining bridge work is standard statistical bridge verification for the same decision-carrier obligation surface and its Type I/II ledger route, not additional paper closure.

Closure status for $\mathsf{NameCert}_{\mathsf{Hypothesis}^\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Hypothesis}^\uparrow}$, seed)

Scope closed: Definition 231.1, Definition 231.5, and Definition 231.8 expose the finite carrier, decision obligation rows, Type I/II ledgers, and public finite-risk route for the hypothesis-testing surface.

Constructive story: A $\mathsf{Hypothesis}^\uparrow$ token is recorded here through the displayed hypothesis-testing decision carrier: null and alternative distribution rows, probability-event rows, a finite statistic-threshold comparison, a decision row, Type I/II budget rows, componentwise transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Hypothesis}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This seed record does not claim Lean verification, bridge checking, statistical optimality, asymptotic consistency, probability completion, or any global hypothesis-selection principle.

Upgrade path: Obligation closure requires a local HypothesisUp carrier boundary that separates the general hypothesis token from the displayed HypothesisTestUp decision carrier and its finite risk ledgers.

Chapter 232

A Concrete Naming Certificate for $\mathsf{Bayesian}^{\uparrow}$

The Bayesian-update interface $\mathsf{Bayesian}^{\uparrow}$ packages the conditional-probability rule transforming a prior distribution and likelihood into a posterior distribution. It depends on $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ (chapter 186) and $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ (chapter 188).

232.1 Carrier definition

232.2 The certificate

Theorem 232.1 (Bayesian posterior source obligation)**.** *Let $\mathcal{B}$ be a Bayesian update packet over* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$*. The packet exposes, as* BHist *data, the prior distribution row π, the likelihood row λ, the evidence row e, the posterior endpoint ω, the classifier comparison* $\mathsf{hsame}(\mathsf{Bayes}(\pi, \lambda, e), \omega)$*, and the normalisation row* $\mathsf{Cont}(\pi, \lambda, e; \omega)$*. These are the only public source obligations of the Bayesian posterior surface; no host probability kernel, measurable-space quotient, or external conditional-density equality is exported by the certificate.*

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorSurface source obligation`

Proof. Project the displayed update packet. Its $\mathsf{Distribution}^{\uparrow}$ source supplies the prior distribution row and the posterior endpoint as distribution-side BHist witnesses. Its $\mathsf{CondExp}^{\uparrow}$ source supplies the conditioning row used to read the likelihood and evidence ledger.

The classifier projection is the displayed history-sameness row comparing the Bayesian update expression with the posterior endpoint. The continuation projection is the normalisation row, recording that the prior, likelihood, and evidence data are continued to the visible posterior endpoint.

Repacking these projected rows gives exactly the stated public surface. Since the packet inventory consists only of the two dependency certificates, the evidence ledger, the posterior endpoint, the classifier row, the continuation row, and provenance, there is no field from which a host probability kernel or external conditional-density equality can be read. □

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorPacket source obligation`

Theorem 232.2 (Bayesian update ledger exactness)**.** *For any accepted Bayesian update packet, every public posterior read factors through the displayed* $\mathsf{Distribution}^{\uparrow}$ *source, the* $\mathsf{CondExp}^{\uparrow}$ *conditioning row, the evidence ledger, the posterior endpoint, and the provenance row. A consumer can therefore read the posterior only by projecting those rows and repacking them into the displayed classifier and normalisation obligations.*

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorSurface update ledger exactness`

Proof. Unpack the accepted packet into its dependency rows. The $\mathsf{Distribution}^{\uparrow}$ component is the sole source for the prior distribution row and the posterior endpoint. The $\mathsf{CondExp}^{\uparrow}$ component is the sole source for the conditioning row that binds the likelihood to the evidence ledger.

The public posterior read then has only two operations available. First, project the classifier comparison $\mathsf{hsame}(\mathsf{Bayes}(\pi, \lambda, e), \omega)$ from the displayed packet rows. Second, project the continuation row $\mathsf{Cont}(\pi, \lambda, e; \omega)$ that records the normalised endpoint.

Finally, the provenance row records the Pkg origin of the distribution, conditioning, evidence, endpoint, classifier, and normalisation rows. Repacking only these projected rows reconstructs the public posterior read, while any read requiring a host kernel, an unstated probability space, or an external density equality has no source field in the packet. □

Definition 232.3 (Bayesian posterior packet carrier)**.** *A* $\mathsf{Bayesian}^{\uparrow}$ *posterior packet is a finite* BHist *packet*

$$\mathcal{B} = (\mathcal{D}_\pi, \mathcal{D}_\omega, \mathcal{C}, \pi, \lambda, e, \omega, \beta, H, N, P)$$

with the following displayed rows. The rows $\mathcal{D}_\pi$ *and* $\mathcal{D}_\omega$ *are* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *source certificates carrying, respectively, the prior distribution row* π *and the posterior distribution endpoint* ω *as distribution data in the sense of Definition 186.1. The row* $\mathcal{C}$ *is a* $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ *source certificate carrying the likelihood row* λ *and evidence row* e *through the singleton conditional-expectation packet of Definition 188.1. The row* β *is the Bayesian update history computed from* π*,* λ*, and* e*. The classifier row is*

$$H = \mathsf{hsame}(\beta, \omega),$$

and the normalisation row is

$$N = \mathsf{Cont}(\pi, \lambda, e; \omega).$$

The provenance row $P : \mathsf{Pkg}$ *names exactly the two distribution certificates, the conditional-expectation certificate, the prior row, the likelihood row, the evidence row, the posterior endpoint, the classifier row, and the normalisation row. The packet is accepted precisely when all these rows are present with this provenance and no additional posterior-source coordinate is exposed.*

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianTasteGate single carrier alignment`

Theorem 232.4 (Bayesian posterior packet carrier introduction)**.** *Suppose the following data are supplied under the* $\mathsf{Bayesian}^{\uparrow}$ *setup: a* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *prior source carrying* π*, a* $\mathsf{NameCert}_{\mathsf{Distribution}^{\uparrow}}$ *posterior source carrying* ω*, a* $\mathsf{NameCert}_{\mathsf{CondExp}^{\uparrow}}$ *row carrying the likelihood and evidence rows* λ *and* e*, a Bayesian update history* β *read from* π*,* λ*, and* e*, a comparison row* $H = \mathsf{hsame}(\beta, \omega)$*, a normalisation row* $N = \mathsf{Cont}(\pi, \lambda, e; \omega)$*, and a provenance row* $P : \mathsf{Pkg}$ *naming exactly these sources and rows. Then these rows assemble an accepted* $\mathsf{Bayesian}^{\uparrow}$ *posterior packet in the carrier of Definition 232.3. Its public posterior source is exactly*

$$(\mathcal{D}_\pi, \mathcal{D}_\omega, \mathcal{C}, \pi, \lambda, e, \omega, H, N, P).$$

In particular, every public posterior read factors through the displayed prior, likelihood, evidence, posterior, classifier, normalisation, and provenance rows, and no host probability kernel, ambient measurable-space quotient, or external conditional-density equality is part of the accepted packet.

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorPacket carrier introduction`

Proof. Form the tuple

$$\mathcal{B} = (\mathcal{D}_\pi, \mathcal{D}_\omega, \mathcal{C}, \pi, \lambda, e, \omega, \beta, H, N, P).$$

The two distribution certificates license the prior and posterior distribution rows by the carrier discipline of Definition 186.1. The conditional-expectation certificate licenses the likelihood and evidence rows by the packet discipline of Definition 188.1.

The supplied comparison row H identifies the computed Bayesian update history β with the displayed posterior endpoint ω through hsame. The supplied normalisation row N records that the visible prior, likelihood, and evidence rows continue to the visible posterior endpoint. The provenance row P names exactly these dependency, classifier, and continuation rows.

These are precisely the acceptance clauses of Definition 232.3, so the tuple is an accepted $\mathsf{Bayesian}^\uparrow$ posterior packet. Projection of an accepted packet returns only its displayed coordinates. Hence every public posterior read is obtained from the two distribution sources, the conditional-expectation source, π, λ, e, ω, H, N, and P.

Conversely, each row in the displayed source list is a coordinate of the accepted tuple, so the list is not missing any public posterior-source row. Since the carrier definition contains no coordinate for a host probability kernel, an ambient measurable-space quotient, or an external conditional-density equality, no such datum can be read from the accepted packet. □

Theorem 232.5 (Bayesian posterior classifier transport)**.** *Let $\mathcal{B}$ and $\mathcal{B}'$ be accepted Bayesian update packets over $\mathsf{NameCert}_{\mathsf{Distribution}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{CondExp}^\uparrow}$. Write the public rows of $\mathcal{B}$, with its explicit Bayesian-update witness history b, as*

$$(\pi, \lambda, e, b, \omega, n, \rho, \mathsf{hsame}(b, \omega), \mathsf{Cont}(\pi, \lambda, e; \omega))$$

and the corresponding public rows of $\mathcal{B}'$, with its explicit Bayesian-update witness history b', as

$$(\pi', \lambda', e', b', \omega', n', \rho', \mathsf{hsame}(b', \omega'), \mathsf{Cont}(\pi', \lambda', e'; \omega')).$$

Assume the displayed prior, likelihood, evidence, Bayesian-update witness, posterior, normalisation, and provenance rows are matched componentwise by hsame *rows, with the prior, Bayesian-update witness, and posterior comparisons read through the* $\mathsf{Distribution}^\uparrow$ *classifier and the likelihood and evidence comparisons read through the* $\mathsf{CondExp}^\uparrow$ *classifier. If the* Cont *normalisation row and the* Pkg *provenance row are transported along these displayed comparisons, then the transported rows form an accepted* $\mathsf{Bayesian}^\uparrow$ *posterior packet, and the transported posterior endpoint ω' is classified with the original posterior endpoint ω. This is only a componentwise classifier-transport and row-repackaging statement for the Bayesian packet surface; it asserts no concrete closure, inversion, coverage, determinacy, uniqueness, bridge, or obstruction claim about* $\mathsf{Bayesian}^\uparrow$*.*

Proof. Project $\mathcal{B}$ by Definition 232.1. The projected surface consists exactly of the prior distribution row π, the likelihood row λ, the evidence row e, the explicit Bayesian-update witness history b, the posterior endpoint ω, the classifier comparison $\mathsf{hsame}(b, \omega)$, the normalisation row $\mathsf{Cont}(\pi, \lambda, e; \omega)$, and the provenance row. By Definition 232.2, every public posterior read of an accepted packet factors through this displayed inventory and through no other row.

Use the supplied componentwise hsame comparisons to move the dependency rows. By Definition 26.3 and Definition 26.4, the $\mathsf{Distribution}^{\uparrow}$ classifier transports the prior row, the Bayesian-update witness history, and the posterior endpoint, so π' remains a distribution-side prior row, b' remains the displayed Bayesian-update witness history, and ω' remains a distribution-side posterior endpoint. By Definition 26.3 and Definition 26.4, the $\mathsf{CondExp}^{\uparrow}$ classifier transports the likelihood and evidence rows, so the conditioning ledger that reads λ against e is replaced by the displayed ledger that reads λ' against e'. Thus the dependency part of the Bayesian packet has the same accepted shape after transport.

It remains to account for the Bayesian-specific rows. The transported normalisation row is, by hypothesis, the visible continuation $\mathsf{Cont}(\pi', \lambda', e'; \omega')$. The transported provenance row is the same Pkg-named source row read at the transported endpoint. Since the accepted Bayesian carrier exposes no row besides the two dependency certificates, the prior, likelihood, evidence, Bayesian-update witness, posterior, classifier, normalisation, and provenance rows, these transported coordinates repack as an accepted $\mathsf{Bayesian}^{\uparrow}$ posterior packet.

Finally compare endpoints. One of the componentwise hypotheses is precisely the posterior transport $\mathsf{hsame}(\omega, \omega')$ supplied through the $\mathsf{Distribution}^{\uparrow}$ endpoint classifier. By symmetry of the displayed history-sameness classifier, the transported endpoint also satisfies $\mathsf{hsame}(\omega', \omega)$. Hence the posterior endpoint of the transported packet is classified with the original posterior endpoint, and the classifier read remains the same visible Bayesian posterior read described by Definition 232.2. □

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorPacket classifier transport`

Theorem 232.6 (Bayesian posterior NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$ *posterior surface is exactly the finite packet formed by the* $\mathsf{Distribution}^{\uparrow}$ *prior and posterior rows, the* $\mathsf{CondExp}^{\uparrow}$ *likelihood and evidence rows, the posterior comparison row* $\mathsf{hsame}(\mathsf{Bayes}(\pi, \lambda, e), \omega)$*, the normalisation row* $\mathsf{Cont}(\pi, \lambda, e; \omega)$*, and the* Pkg *provenance naming those dependency rows. A posterior read is accepted only when those rows are present in the carrier of Definition 232.3.*

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorPacket namecert obligation surface`

Proof. Project the carrier of Definition 232.3. Its dependency coordinates are the two $\mathsf{Distribution}^{\uparrow}$ rows and the $\mathsf{CondExp}^{\uparrow}$ row, and these are the only sources for the visible prior, likelihood, evidence, and posterior endpoint.

The Bayesian-specific coordinates are the displayed posterior comparison, the normalisation continuation, and the package provenance. Together with the dependency rows, these coordinates are exactly the public posterior surface described by the source and ledger exactness theorems above.

No further obligation is hidden behind the classifier. Acceptance is therefore the presence of these finite rows in the packet and their readback through the displayed dependency certificates. □

Theorem 232.7 (Bayesian posterior non-escape boundary)**.** *An accepted* $\mathsf{Bayesian}^{\uparrow}$ *posterior packet exports no host probability kernel, no external density equality, no ambient measurable-space quotient, and no choice-selected posterior. Every public posterior read is mediated by the displayed* $\mathsf{Distribution}^{\uparrow}$ *rows,* $\mathsf{CondExp}^{\uparrow}$ *rows,* hsame *comparison,* Cont *normalisation,* Pkg *provenance, and local packet carrier.*

Proof. Inspect the accepted packet. Each coordinate is either a dependency certificate row, a displayed Bayesian-update history, a posterior endpoint, a classifier comparison, a continuation row, or provenance.

The carrier has no coordinate for a host kernel, a density equality outside the packet, a quotient of measurable spaces, or a selected posterior obtained by choice. Transport can only move the displayed coordinates along their hsame and Cont rows.

Thus a downstream posterior read factors through the finite packet boundary. Any read requiring one of the excluded objects has no source coordinate in the accepted $\mathsf{Bayesian}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.BayesianUp.BayesianPosteriorPacket non escape boundary`

Theorem 232.8 (Bayesian posterior public source exhaustion)**.** *Every public posterior read from* $\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$ *factors through the finite rows listed in Definition 232.3. In particular, the read is exhausted by* $\mathcal{D}_\pi$*,* $\mathcal{D}_\omega$*,* $\mathcal{C}$*,* π*,* λ*,* e*,* ω*,* β*,* H*,* N*, and* P*, with no additional posterior-source coordinate.*

Proof. Start with a public posterior read. By Definition 232.1, the read first projects the dependency rows and the displayed posterior endpoint.

Apply Definition 232.2. The ledger forces the read to pass through the visible Bayesian-update history, classifier row, normalisation row, and provenance row, and it gives no operation for introducing an external probability object.

The carrier definition lists exactly these coordinates. Repacking them reconstructs the public posterior read, while any attempted read outside this list is blocked by Definition 232.7. □

Definition 232.9 (Bayesian posterior obligation inventory)**.** *The* $\mathsf{Bayesian}^{\uparrow}$ *posterior obligation inventory is the finite* BHist *list*

$$\mathcal{I}_{\mathsf{Bayesian}^{\uparrow}} = (\mathcal{K}, \mathcal{T}, \mathcal{L}, \mathcal{S}, \mathcal{E})$$

where $\mathcal{K}$ *is the carrier clause of Definition 232.3,* $\mathcal{T}$ *is the classifier-transport clause of Definition 232.5,* $\mathcal{L}$ *is the ledger-exactness clause of Definition 232.2,* $\mathcal{S}$ *is the posterior-source exhaustion clause of Definition 232.8, and* $\mathcal{E}$ *is the non-escape clause of Definition 232.7. The inventory is accepted only when each clause is read through the displayed* $\mathsf{Distribution}^{\uparrow}$ *rows,* $\mathsf{CondExp}^{\uparrow}$ *row,* hsame *classifier row,* Cont *normalisation row,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$ *surface.*

Theorem 232.10 (Bayesian posterior obligation package)**.** *The inventory of Definition 232.9 assembles the paper-side* $\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$ *posterior obligation package. Its five public clauses are carrier admission, classifier transport, ledger exactness, posterior source exhaustion, and non-escape from host probability kernels or measurable-space quotients, witnessed respectively by Definition 232.3, Definition 232.5, Definition 232.2, Definition 232.8, and Definition 232.7.*

Proof. First project the carrier clause. It supplies exactly the distribution sources, conditional-expectation source, prior row, likelihood row, evidence row, posterior endpoint, Bayesian-update history, classifier comparison, normalisation row, and provenance row named in the inventory.

Next use classifier transport and ledger exactness. The classifier clause moves the displayed rows componentwise through hsame, while the ledger clause says that every posterior read factors through those same dependency, classifier, continuation, and provenance rows.

Finally apply posterior source exhaustion and non-escape. Exhaustion says the listed rows are all public posterior-source coordinates, and non-escape says there is no coordinate for a host kernel, external density equality, measurable-space quotient, or choice-selected posterior. These five clauses therefore form the advertised finite obligation package. □

Theorem 232.11 (Bayesian posterior obligation non-escape)**.** *Every downstream consumer of the* $\mathsf{Bayesian}^{\uparrow}$ *posterior obligation package receives only the rows in* $\mathcal{I}_{\mathsf{Bayesian}^{\uparrow}}$*. In particular, the package cannot be extended by reading a host probability kernel, an ambient measurable-space quotient, an external conditional-density equality, or a posterior selected outside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with the obligation package of Definition 232.10. Its clauses are the carrier, classifier transport, ledger exactness, posterior source exhaustion, and non-escape rows.

The source-exhaustion clause fixes the public read boundary: a consumer can project $\mathcal{D}_\pi$, $\mathcal{D}_\omega$, $\mathcal{C}$, π, λ, e, ω, β, H, N, and P, and can transport these rows only by the displayed classifier and continuation data.

The non-escape clause then blocks the excluded host objects. Since the inventory contains no coordinate for probability kernels, measurable-space quotients, external density equalities, or selected posterior witnesses, no downstream consumer can recover one from the package. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Bayesian}^{\uparrow}$ posterior packet is generated from the $\mathsf{Distribution}^{\uparrow}$ prior row, the $\mathsf{Distribution}^{\uparrow}$ posterior row, the $\mathsf{CondExp}^{\uparrow}$ likelihood and evidence row, the Bayesian update history, the displayed hsame posterior comparison, the Cont normalisation row, and Pkg provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 232.9, Definition 232.10, and Definition 232.11 bind the carrier, classifier transport, ledger exactness, posterior source exhaustion, and non-escape clauses of the public posterior package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Bayesian}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BayesianUp.BayesianPosteriorPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host probability kernels, measurable-space quotients, conditional-density equality, or bridge checking.

Upgrade path: Public closure requires an exported posterior consumer surface for the scoped obligation package.

Chapter 233

A Concrete Naming Certificate for $\mathsf{NetworkFlow}^{\uparrow}$

The network-flow interface $\mathsf{NetworkFlow}^{\uparrow}$ packages a directed graph carrier with edge capacities together with the maximum-flow / minimum-cut duality. It depends on $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118) and $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28).

233.1 Carrier definition

233.2 The certificate

Definition 233.1 (Finite NetworkFlowUp flow and cut data)**.** *Fix a directed* $\mathsf{Graph}^{\uparrow}$ *carrier* G*. Write* $\mathsf{Vert}_G(v)$ *for vertex membership,* $\mathsf{Edge}_G(e)$ *for edge membership,* $\mathsf{tail}_G(e)$ *and* $\mathsf{head}_G(e)$ *for the directed endpoints of an edge, and* $\sim_E$ *for the edge classifier used by the finite-edge presentation. On unary histories write*

$$x \preceq_{\mathsf{Nat}^{\uparrow}} y \iff \mathsf{PreorderPrefixLE}(x, y),$$

the concrete $\mathsf{Nat}^{\uparrow}$ *below relation from the unary-prefix preorder of Definition 52.3.*

For a finite edge spine xs *and a unary edge assignment* $a : \mathsf{BHist} \to \mathsf{BHist}$*, define the finite unary fold by*

$$\mathsf{USum}_{\mathsf{nil}}(a) := \mathsf{e}, \qquad \mathsf{USum}_{\mathsf{cons}(e,xs')}(a) := \mathsf{append}(a(e), \mathsf{USum}_{xs'}(a)).$$

This is the finite-spine form of the unary continuation fold licensed by $\mathsf{Add}^{\uparrow}$ *in Definition 29.7.*

A finite s*-*t *cut record* C *in* G *consists of a vertex predicate* S_C*, witnesses* $\mathsf{Vert}_G(s)$*,* $\mathsf{Vert}_G(t)$*,* $S_C(s)$*, and* $\neg S_C(t)$*, together with finite edge spines* C^+ *and* C^- *enumerating the forward and backward crossing predicates:*

$$\mathsf{Forward}_C(e) \iff \mathsf{Edge}_G(e) \wedge S_C(\mathsf{tail}_G(e)) \wedge \neg S_C(\mathsf{head}_G(e)),$$

$$\mathsf{Backward}_C(e) \iff \mathsf{Edge}_G(e) \wedge \neg S_C(\mathsf{tail}_G(e)) \wedge S_C(\mathsf{head}_G(e)),$$

by the finite-set enumeration predicate of Definition 112.17. For a capacity assignment u *and a flow assignment* f*, set*

$$\mathsf{CutCap}_{G,u}(C) := \mathsf{USum}_{C^+}(u), \qquad \mathsf{CutForw}_{G,C}(f) := \mathsf{USum}_{C^+}(f), \qquad \mathsf{CutBack}_{G,C}(f) := \mathsf{USum}_{C^-}(f).$$

A feasible $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow* F *over the capacity assignment* u *consists of a unary edge-flow assignment* f_F*, a unary value history witness* V_F*, and the notation*

$$\mathsf{FlowValue}(F) := V_F,$$

a capacity row

$$\forall e : \mathsf{BHist}.\ \mathsf{Edge}_G(e) \to f_F(e) \preceq_{\mathsf{Nat}^{\uparrow}} u(e),$$

and, for every finite s-t cut record C *in the same* G*, history witnesses* $B_{F,C}$ *and* $X_{F,C}$ *with identification rows*

$$B_{F,C} = \mathsf{CutBack}_{G,C}(f_F), \qquad X_{F,C} = \mathsf{CutForw}_{G,C}(f_F),$$

together with the finite conservation-accounting row

$$\mathsf{Cont}(V_F, B_{F,C}, X_{F,C}).$$

The accounting row records the cut form of flow conservation; the capacity row is separate.

Lean4 (definition) `BEDC.Derived.NetworkFlowUp.NetworkFlowFiniteFlowCutData`

Lemma 233.2 (Unary finite-fold monotonicity)**.** *Let* xs *be a finite spine and let* $a, b : \mathsf{BHist} \to \mathsf{BHist}$ *be unary assignments on every entry of* xs*. If every position of* xs *satisfies*

$$a(\mathsf{entry}_{xs}(i)) \preceq_{\mathsf{Nat}^{\uparrow}} b(\mathsf{entry}_{xs}(i)),$$

then

$$\mathsf{USum}_{xs}(a) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{USum}_{xs}(b).$$

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlowUSum prefix monotone`
Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlowUSum monotonicity`

Proof. Induct on the finite spine xs. In the nil case, both folds are e, so reflexivity of the unary-prefix preorder from Definition 52.4 gives the displayed comparison.

For the cons case, write $xs = \mathsf{cons}(e, xs')$. The pointwise hypothesis at the head gives $a(e) \preceq_{\mathsf{Nat}^{\uparrow}} b(e)$, and the induction hypothesis gives

$$\mathsf{USum}_{xs'}(a) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{USum}_{xs'}(b).$$

Right-context monotonicity of the concrete prefix relation, Definition 52.37, applied to the head comparison and the unary tail $\mathsf{USum}_{xs'}(a)$, gives

$$\mathsf{append}(a(e), \mathsf{USum}_{xs'}(a)) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{append}(b(e), \mathsf{USum}_{xs'}(a)).$$

Left-context monotonicity, Definition 52.36, applied to the tail-fold comparison with left context $b(e)$, gives

$$\mathsf{append}(b(e), \mathsf{USum}_{xs'}(a)) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{append}(b(e), \mathsf{USum}_{xs'}(b)).$$

Transitivity from Definition 52.28 composes these two comparisons. After unfolding the cons clause of Definition 233.1, this is exactly the finite-fold comparison for $\mathsf{cons}(e, xs')$. □

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow unary finite fold monotonicity`
Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlowUSum monotone`
Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlowUnaryFold pointwise prefix monotonicity`

Theorem 233.3 (Network-flow weak duality for cuts)**.** *Let G be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let u be a unary capacity assignment on its edges, and let s, t be carried vertices. If F is a feasible* $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow over u and C is a finite s-t cut record in the same directed graph, then the value of F is* $\mathsf{Nat}^{\uparrow}$*-classified below the capacity of C:*

$$\mathsf{FlowValue}(F) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(C).$$

Proof. Unpack Definition 233.1. Put $f = f_F$ and take the flow-value history witness V_F, together with the cut-history witnesses $B_{F,C}$ and $X_{F,C}$ for the chosen cut C. Their identification rows are

$$\mathsf{FlowValue}(F) = V_F, \qquad B_{F,C} = \mathsf{CutBack}_{G,C}(f), \qquad X_{F,C} = \mathsf{CutForw}_{G,C}(f).$$

The conservation-accounting row of feasibility for the cut C supplies

$$\mathsf{Cont}(V_F, B_{F,C}, X_{F,C}).$$

Since the backward crossing witness $B_{F,C}$ is unary, this continuation witness is precisely a unary-tail prefix witness from V_F to $X_{F,C}$. Hence

$$V_F \preceq_{\mathsf{Nat}^{\uparrow}} X_{F,C}.$$

Using the identification rows, this is the comparison

$$\mathsf{FlowValue}(F) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutForw}_{G,C}(f).$$

It remains to compare $\mathsf{CutForw}_{G,C}(f)$ with $\mathsf{CutCap}_{G,u}(C)$. Every entry of the forward cut spine C^+ is an edge of G by its $\mathsf{FinSet}^{\uparrow}$ enumeration witness. The capacity row of feasibility therefore gives, at each position i of C^+,

$$f(\mathsf{entry}_{C^+}(i)) \preceq_{\mathsf{Nat}^{\uparrow}} u(\mathsf{entry}_{C^+}(i)).$$

Applying Definition 233.2 to the finite spine C^+ gives

$$\mathsf{CutForw}_{G,C}(f) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(C).$$

Finally compose $\mathsf{FlowValue}(F) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutForw}_{G,C}(f)$ with $\mathsf{CutForw}_{G,C}(f) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(C)$ by Definition 52.28. This proves

$$\mathsf{FlowValue}(F) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(C).$$

□

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow weak duality cuts`

Theorem 233.4 (Network-flow cut accounting from vertex conservation)**.** *Let G be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let C be a finite s-t cut record in G, and let f be a unary edge-flow assignment with unary value history V. Let $\sim_V$ be the vertex classifier carried by G. Suppose W_C is a finite vertex spine enumerating the nonterminal cut-side predicate*

$$\mathsf{CutSide}_{C,s,t}(v) \iff \mathsf{Vert}_G(v) \wedge S_C(v) \wedge \neg(v \sim_V s) \wedge \neg(v \sim_V t)$$

in the finite-spine sense of Definition 112.17. Let I_s and O_s be finite edge spines enumerating the incoming and outgoing edges at s, and for every entry v of W_C let I_v and O_v be finite edge

spines enumerating the incoming and outgoing edges at v. Let $\mathsf{VCat}_{W_C}(I)$ and $\mathsf{VCat}_{W_C}(O)$ denote the concatenations of these edge spines in the order of W_C. Finally let J_C be a finite edge spine enumerating the internal cut-side edges

$$\mathsf{Internal}_C(e) \iff \mathsf{Edge}_G(e) \wedge S_C(\mathsf{tail}_G(e)) \wedge S_C(\mathsf{head}_G(e)).$$

Assume the finite incidence readback rows

$$\mathsf{USum}_{I_s ++ \mathsf{VCat}_{W_C}(I)}(f)\mathsf{hsameappend}(\mathsf{CutBack}_{G,C}(f), \mathsf{USum}_{J_C}(f))$$

and

$$\mathsf{USum}_{O_s ++ \mathsf{VCat}_{W_C}(O)}(f)\mathsf{hsameappend}(\mathsf{CutForw}_{G,C}(f), \mathsf{USum}_{J_C}(f))$$

are supplied by the certified finite-position accounting of Definition 112.2. If the source-value row

$$\mathsf{Cont}(V, \mathsf{USum}_{I_s}(f), \mathsf{USum}_{O_s}(f))$$

holds and every nonterminal cut-side vertex satisfies vertex conservation

$$\forall v : \mathsf{BHist.\ SpineMember}(v, W_C) \to \mathsf{USum}_{I_v}(f)\mathsf{hsameUSum}_{O_v}(f),$$

then the cut supplies the finite conservation-accounting row

$$\mathsf{Cont}(V, \mathsf{CutBack}_{G,C}(f), \mathsf{CutForw}_{G,C}(f)).$$

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow cut accounting suffix cancellation`

Proof. First aggregate the local vertex rows over the finite spine W_C. Induct through W_C, carrying the invariant that the source row together with the conservation rows for the vertices already visited supplies a continuation from V by the accumulated incoming-side sum to the accumulated outgoing-side sum. The empty-spine case is the source-value row, using the nil clause in the finite fold of Definition 233.1. In the cons step, the induction hypothesis supplies the accumulated continuation and the head occurrence supplies

$$\mathsf{USum}_{I_v}(f)\mathsf{hsameUSum}_{O_v}(f).$$

Associativity for unary continuation from Definition 29.6 appends this head block to the accumulated incoming and outgoing ledgers, and the commutative unary additive field from Definition 29.10, concretely Definition 9194.6, permits the block to be read in the vertex-spine order used by VCat_{W_C}. Thus the induction yields

$$\mathsf{Cont}(V, \mathsf{USum}_{I_s ++ \mathsf{VCat}_{W_C}(I)}(f), \mathsf{USum}_{O_s ++ \mathsf{VCat}_{W_C}(O)}(f)).$$

Put $J = \mathsf{USum}_{J_C}(f)$. Transport this aggregate continuation along the two displayed incidence readback rows. This gives

$$\mathsf{Cont}(V, \mathsf{append}(\mathsf{CutBack}_{G,C}(f), J), \mathsf{append}(\mathsf{CutForw}_{G,C}(f), J)).$$

The forward direction of the continuation suffix equivalence Definition 18.22, with $a = V$, $f = \mathsf{CutBack}_{G,C}(f)$, $b = \mathsf{CutForw}_{G,C}(f)$, and $p = J$, cancels the common internal suffix and yields

$$\mathsf{Cont}(V, \mathsf{CutBack}_{G,C}(f), \mathsf{CutForw}_{G,C}(f)).$$

This is exactly the finite conservation-accounting row from Definition 233.1. □

Definition 233.5 (Residual-cut augmenting-path exhaustion certificate)**.** *Fix the finite* $\mathsf{NetworkFlow}^{\uparrow}$ *setup of Definition 233.1: a directed* $\mathsf{Graph}^{\uparrow}$ *carrier* G*, a unary capacity assignment* u*, carried vertices* s, t*, and a feasible* s*-*t *flow* F *with edge-flow assignment* f_F*. A residual-cut augmenting-path exhaustion certificate* Ξ *for* F *consists of a finite* s*-*t *cut record* C_Ξ *in* G*, written*

$$\mathsf{ResCut}_{F,u}(\Xi) := C_\Xi,$$

and the following residual boundary rows. For each $i : \mathsf{Pos}(C_\Xi^+)$*, write*

$$U_{\Xi,i} := u(\mathsf{entry}_{C_\Xi^+}(i)), \qquad E_{\Xi,i} := f_F(\mathsf{entry}_{C_\Xi^+}(i)).$$

Then

$$U_{\Xi,i} \preceq_{\mathsf{Nat}^{\uparrow}} E_{\Xi,i}.$$

There is also a named backward residual history B_Ξ *with*

$$B_\Xi := \mathsf{CutBack}_{G,C_\Xi}(f_F), \qquad B_\Xi \mathsf{hsamee}.$$

The first row is the finite-spine saturation of every forward edge of the induced residual cut. The second row is the finite-spine absence of backward flow across that cut.

Lemma 233.6 (Empty backward accounting reflects cut flow below value)**.** *Let* G *be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let* u *be a unary capacity assignment, let* F *be a feasible finite* $\mathsf{NetworkFlow}^{\uparrow}$ s*-*t *flow over* u *with edge-flow assignment* f_F*, and let* C *be a finite* s*-*t *cut record in* G*. Write*

$$B_C := \mathsf{CutBack}_{G,C}(f_F), \qquad X_C := \mathsf{CutForw}_{G,C}(f_F), \qquad V_F := \mathsf{FlowValue}(F).$$

If

$$B_C \mathsf{hsamee},$$

then

$$X_C \preceq_{\mathsf{Nat}^{\uparrow}} V_F.$$

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlowEmptyBackwardAccounting cut flow below value`
Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow empty backward accounting cut flow below value`

Proof. Unpack Definition 233.1 for the chosen cut C. The feasibility data supply histories $B_{F,C}$ and $X_{F,C}$ with

$$B_{F,C} = B_C, \qquad X_{F,C} = X_C,$$

and with the accounting row

$$\mathsf{Cont}(V_F, B_{F,C}, X_{F,C}).$$

The displayed backward-empty hypothesis and the identification row for $B_{F,C}$ give $B_{F,C}\mathsf{hsamee}$. The identification row for $X_{F,C}$ gives $X_{F,C}\mathsf{hsame}X_C$.

Transport the accounting row along these endpoint classifications using Definition 4.41. This gives

$$\mathsf{Cont}(V_F, \mathsf{e}, X_C).$$

The right-unit clause of Definition 4.38 identifies the result with the source, so

$$X_C \mathsf{hsame} V_F.$$

Applying Definition 52.17 to this history-sameness gives the displayed $\mathsf{Nat}^{\uparrow}$ prefix comparison. □

Theorem 233.7 (Network-flow max-flow min-cut equality from residual exhaustion)**.** *Let G be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let u be a unary capacity assignment, let s, t be carried vertices, and let F be a feasible finite* $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow over u with edge-flow assignment f_F. If Ξ is a residual-cut augmenting-path exhaustion certificate for F, write*

$$C_\Xi := \mathsf{ResCut}_{F,u}(\Xi), \qquad V_F := \mathsf{FlowValue}(F), \qquad K_\Xi := \mathsf{CutCap}_{G,u}(C_\Xi).$$

Then the value of F is $\mathsf{Nat}^{\uparrow}$*-classified equal to the capacity of the residual cut induced by Ξ:*

$$V_F \mathsf{hsame} K_\Xi.$$

Moreover, the equality is supported by the two prefix comparisons

$$V_F \preceq_{\mathsf{Nat}^{\uparrow}} K_\Xi \qquad \textit{and} \qquad K_\Xi \preceq_{\mathsf{Nat}^{\uparrow}} V_F.$$

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow maxflow mincut equality from residual exhaustion`

Proof. Use the displayed names and put

$$X_\Xi := \mathsf{CutForw}_{G,C_\Xi}(f_F).$$

The comparison from flow value to cut capacity is exactly Definition 233.3 applied to the feasible flow F and the finite cut record C_Ξ:

$$V_F \preceq_{\mathsf{Nat}^{\uparrow}} K_\Xi.$$

Unpack Definition 233.5. Its forward saturation row gives, for every $i : \mathsf{Pos}(C_\Xi^+)$, with $U_{\Xi,i}$ and $E_{\Xi,i}$ as named there,

$$U_{\Xi,i} \preceq_{\mathsf{Nat}^{\uparrow}} E_{\Xi,i}.$$

Apply Definition 233.2 to the finite spine C_Ξ^+ with assignments u and f_F. By the definitions of K_Ξ and X_Ξ in Definition 233.1, this yields

$$K_\Xi \preceq_{\mathsf{Nat}^{\uparrow}} X_\Xi.$$

The backward-empty row of the certificate and Definition 233.6 give

$$X_\Xi \preceq_{\mathsf{Nat}^{\uparrow}} V_F.$$

Transitivity of the concrete unary prefix preorder from Definition 52.28 composes these two comparisons and gives

$$K_\Xi \preceq_{\mathsf{Nat}^{\uparrow}} V_F.$$

The two displayed prefix comparisons now face in opposite directions. Applying Definition 52.30 gives

$$V_F \mathsf{hsame} K_\Xi.$$

Both endpoints are unary: the source is the flow-value witness of feasibility, and the target is the finite unary fold defining cut capacity. Therefore Definition 28.27 reads this history-sameness as the claimed $\mathsf{Nat}^{\uparrow}$ classification equality. □

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow residual exhaustion value capacity hsame`
Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow residual exhaustion cut value hsame`

Theorem 233.8 (Network-flow equality implies optimality)**.** *Let G be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let u be a unary capacity assignment, and let s, t be carried vertices. Let F be a feasible finite* $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow over u, and let C be a finite s-t cut record in G. If*

$$\mathsf{FlowValue}(F)\mathsf{hsameCutCap}_{G,u}(C),$$

then F is maximal among feasible finite s-t flows and C is minimal among finite s-t cuts: for every feasible finite $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow F' over u and every finite s-t cut record D in G,*

$$\mathsf{FlowValue}(F') \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{FlowValue}(F) \qquad \textit{and} \qquad \mathsf{CutCap}_{G,u}(C) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(D).$$

Proof. Write

$$V := \mathsf{FlowValue}(F), \qquad K := \mathsf{CutCap}_{G,u}(C).$$

The displayed equality gives $V\mathsf{hsame}K$. By symmetry of history-sameness from Definition 3.8, this also gives $K\mathsf{hsame}V$. Applying Definition 52.17 yields

$$K \preceq_{\mathsf{Nat}^{\uparrow}} V.$$

Let F' be any feasible finite $\mathsf{NetworkFlow}^{\uparrow}$ s-t flow over u. Applying weak duality, Definition 233.3, to F' and the fixed cut C gives

$$\mathsf{FlowValue}(F') \preceq_{\mathsf{Nat}^{\uparrow}} K.$$

Compose this comparison with $K \preceq_{\mathsf{Nat}^{\uparrow}} V$ by Definition 52.28. Since $V = \mathsf{FlowValue}(F)$, the result is

$$\mathsf{FlowValue}(F') \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{FlowValue}(F).$$

Let D be any finite s-t cut record in G. Applying weak duality to the fixed feasible flow F and the cut D gives

$$V \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(D).$$

Compose $K \preceq_{\mathsf{Nat}^{\uparrow}} V$ with this comparison by Definition 52.28. Since $K = \mathsf{CutCap}_{G,u}(C)$, the result is

$$\mathsf{CutCap}_{G,u}(C) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(D).$$

The two comparisons are exactly the displayed maximality and minimality clauses. □

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow equality implies optimality`

Corollary 233.9 (Network-flow residual exhaustion optimality)**.** *Let G be a directed* $\mathsf{Graph}^{\uparrow}$ *carrier, let u be a unary capacity assignment, and let s, t be carried vertices. Let F be a feasible finite* $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow over u with edge-flow assignment f_F. If Ξ is a residual-cut augmenting-path exhaustion certificate for F, write*

$$C_{\Xi} := \mathsf{ResCut}_{F,u}(\Xi).$$

Then F is maximal among feasible finite s-t flows and C_{Ξ} is minimal among finite s-t cuts: for every feasible finite $\mathsf{NetworkFlow}^{\uparrow}$ *s-t flow F' over u and every finite s-t cut record D in G,*

$$\mathsf{FlowValue}(F') \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{FlowValue}(F) \qquad \textit{and} \qquad \mathsf{CutCap}_{G,u}(C_{\Xi}) \preceq_{\mathsf{Nat}^{\uparrow}} \mathsf{CutCap}_{G,u}(D).$$

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow residual exhaustion optimality`

Proof. Put

$$V_F := \mathsf{FlowValue}(F), \qquad K_\Xi := \mathsf{CutCap}_{G,u}(C_\Xi).$$

By Definition 233.7, the residual exhaustion certificate supplies

$$V_F \mathsf{hsame} K_\Xi.$$

Apply Definition 233.8 to the feasible flow F and the finite cut record C_Ξ. For each feasible finite s-t flow F' over u and each finite s-t cut record D in G, it gives

$$\mathsf{FlowValue}(F') \preceq_{\mathsf{Nat}^\uparrow} \mathsf{FlowValue}(F) \qquad \text{and} \qquad \mathsf{CutCap}_{G,u}(C_\Xi) \preceq_{\mathsf{Nat}^\uparrow} \mathsf{CutCap}_{G,u}(D).$$

These are exactly the maximal-flow and minimal-cut clauses for the residual cut induced by Ξ. □

Lemma 233.10 (Finite unary fold of the constant empty assignment)**.** *Fix a finite spine xs and define*

$$a_{\mathsf{e}}(e) := \mathsf{e} \qquad \textit{for every } e : \mathsf{BHist}.$$

Then

$$\mathsf{USum}_{xs}(a_{\mathsf{e}}) = \mathsf{e}.$$

Proof. Induct on the finite spine xs. In the nil case, the defining nil clause of Definition 233.1 gives

$$\mathsf{USum}_{\mathsf{nil}}(a_{\mathsf{e}}) = \mathsf{e}.$$

For the cons case, write $xs = \mathsf{cons}(e, xs')$. Unfolding the cons clause of Definition 233.1 gives

$$\mathsf{USum}_{\mathsf{cons}(e,xs')}(a_{\mathsf{e}}) = \mathsf{append}(a_{\mathsf{e}}(e), \mathsf{USum}_{xs'}(a_{\mathsf{e}})).$$

By definition of a_{e} and the induction hypothesis, this becomes

$$\mathsf{USum}_{\mathsf{cons}(e,xs')}(a_{\mathsf{e}}) = \mathsf{append}(\mathsf{e}, \mathsf{e}).$$

By the append presentation of continuation from Definition 4.10, the left-unit clause of Definition 4.38 identifies $\mathsf{append}(\mathsf{e}, \mathsf{e})$ with e. Hence

$$\mathsf{USum}_{\mathsf{cons}(e,xs')}(a_{\mathsf{e}}) = \mathsf{e}.$$

This closes the induction. □

Theorem 233.11 (Zero edge-flow is feasible at zero value)**.** *Let G be a directed $\mathsf{Graph}^\uparrow$ carrier, let u be a unary capacity assignment on its edges, and let s, t be carried vertices. Define the zero edge-flow assignment by*

$$f_0(e) := \mathsf{e} \qquad \textit{for every } e : \mathsf{BHist},$$

and define the value witness by

$$V_{F_0} := \mathsf{e}.$$

Then the flow datum F_0 with edge-flow assignment f_0 and value history V_{F_0} is a feasible $\mathsf{NetworkFlow}^\uparrow$ s-t flow over u. More precisely, for every finite s-t cut record C in G, one may take

$$B_{F_0,C} := \mathsf{e}, \qquad X_{F_0,C} := \mathsf{e},$$

and all rows required by Definition 233.1 hold.

Lean4 ✓ `BEDC.Derived.NetworkFlowUp.NetworkFlow zero edge flow feasible`

Proof. Unfold feasibility from Definition 233.1. We must verify the capacity row and the cut-accounting row.

For the capacity row, fix $e : \mathsf{BHist}$ and assume $\mathsf{Edge}_G(e)$. Because u is a unary capacity assignment on edges, the history $u(e)$ lies in the concrete unary history preorder carrier. Therefore Definition 52.19 gives

$$\mathsf{e} \preceq_{\mathsf{Nat}^\uparrow} u(e).$$

Since $f_0(e) = \mathsf{e}$, this is exactly

$$f_0(e) \preceq_{\mathsf{Nat}^\uparrow} u(e).$$

So the capacity row holds for every carried edge.

Now fix a finite s-t cut record C in G. Apply Definition 233.10 to the backward and forward cut spines with the constant assignment f_0. Using the definitions of $\mathsf{CutBack}_{G,C}$ and $\mathsf{CutForw}_{G,C}$ from Definition 233.1, this gives

$$\mathsf{CutBack}_{G,C}(f_0) = \mathsf{e}, \qquad \mathsf{CutForw}_{G,C}(f_0) = \mathsf{e}.$$

Hence the chosen witnesses satisfy the required identification rows

$$B_{F_0,C} = \mathsf{CutBack}_{G,C}(f_0), \qquad X_{F_0,C} = \mathsf{CutForw}_{G,C}(f_0).$$

It remains to verify the conservation-accounting row. With the chosen value and cut histories, this row is

$$\mathsf{Cont}(V_{F_0}, B_{F_0,C}, X_{F_0,C}) = \mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e}).$$

The left-unit clause of Definition 4.38 gives exactly

$$\mathsf{Cont}(\mathsf{e}, \mathsf{e}, \mathsf{e}).$$

Thus the required accounting row holds for every finite cut record C.

Both the capacity row and the cut-accounting row are therefore satisfied, so F_0 is a feasible $\mathsf{NetworkFlow}^\uparrow$ s-t flow over u with zero edge-flow, zero value witness, and trivial cut-history witnesses. □

Theorem 233.12 (Zero feasible network flow has minimum value)**.** *Let G be a directed $\mathsf{Graph}^\uparrow$ carrier, let u be a unary capacity assignment on its edges, and let s, t be carried vertices. Define the zero edge-flow assignment by*

$$f_0(e) := \mathsf{e} \qquad \textit{for every } e : \mathsf{BHist},$$

and let F_0 be the feasible $\mathsf{NetworkFlow}^\uparrow$ s-t flow over u supplied by Definition 233.11, with value witness

$$V_{F_0} := \mathsf{e}.$$

Then F_0 has minimum value among feasible finite s-t flows over u: for every feasible finite $\mathsf{NetworkFlow}^\uparrow$ s-t flow F over u,

$$\mathsf{FlowValue}(F_0) \preceq_{\mathsf{Nat}^\uparrow} \mathsf{FlowValue}(F).$$

Proof. Unpack the feasible-flow datum of Definition 233.1 for the arbitrary feasible flow F. Its value witness is a unary history, and by notation

$$\mathsf{FlowValue}(F) = V_F.$$

Since V_F lies in the concrete unary history preorder carrier, Definition 52.19 gives

$$\mathsf{e} \preceq_{\mathsf{Nat}^\uparrow} V_F.$$

Using the value-witness definition of the zero flow from Definition 233.11,

$$\mathsf{FlowValue}(F_0) = V_{F_0} = \mathsf{e}.$$

Transporting the left endpoint along this identification and the right endpoint along $\mathsf{FlowValue}(F) = V_F$ gives

$$\mathsf{FlowValue}(F_0) \preceq_{\mathsf{Nat}^\uparrow} \mathsf{FlowValue}(F).$$

The quantified flow F was arbitrary among feasible finite s-t flows over u, so the zero feasible flow is minimum for the concrete $\mathsf{Nat}^\uparrow$ prefix order on flow values. □

Closure status for $\mathsf{NameCert}_{\mathsf{NetworkFlow}^\uparrow}$**.**

Constructive story: A $\mathsf{NetworkFlow}^\uparrow$ packet is generated from finite BHist rows for a directed graph, unary capacity assignment, source and target endpoints, feasible-flow witnesses, and finite cut witnesses. The flow value and cut-capacity reads are not external graph-optimization objects; they are Cont folds over displayed edge spines whose summands are unary $\mathsf{Nat}^\uparrow$ histories. The classifier compares accepted packets by hsame on graph, capacity, flow, cut, and value rows, while Pkg records the consumed $\mathsf{Graph}^\uparrow$, $\mathsf{Nat}^\uparrow$, and additive-fold dependencies. Thus the max-flow/min-cut surface is a finite ledger of histories and routes, with no hidden residual network or host optimization oracle.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NetworkFlow}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 233.1, Definition 233.2, Definition 233.3, Definition 233.4, Definition 233.7, Definition 233.11, and Definition 233.12 bind the public finite flow-cut export: residual-exhaustion max-flow/min-cut equality, zero edge-flow feasibility, zero feasible minimum value, BHist ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows displayed in the chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NetworkFlow}^\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.NetworkFlowUp.NetworkFlow\_maxflow\_mincut\_equality\_from\_residual\_exhaustion`

Bridge status: paperBridge

Origin: human

Not claimed: It does not claim a host graph algorithm, residual-network completeness, ambient linear programming duality, or equality of external flow objects.

Upgrade path: No further theory-axis upgrade remains beyond maintaining bridge and readback audits for the finite residual-exhaustion and cut-certificate surface.

Chapter 234

A Concrete Naming Certificate for $\mathsf{Matching}^{\uparrow}$

The graph-matching interface $\mathsf{Matching}^{\uparrow}$ packages a subset of edges in a graph carrier with no shared vertex, together with the maximum-matching / minimum-vertex-cover duality on bipartite graphs. It depends on $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

234.1 Carrier definition

234.2 The certificate

Definition 234.1 (Matching carrier and finite edge subset)**.** *Fix a* $\mathsf{Graph}^{\uparrow}$ *carrier* G*. Write* $\mathsf{Vert}_G(v)$ *for vertex membership,* $\mathsf{Edge}_G(e)$ *for edge membership,* $\mathsf{Inc}_G(v, e)$ *for incidence of a vertex with an edge, and* $\sim_E$ *for the edge classifier used by the* $\mathsf{FinSet}^{\uparrow}$ *presentation of graph edges. For an edge predicate* $M : \mathsf{BHist} \to \mathsf{Prop}$*, define*

$$\mathsf{MatchingEdgeSet}_G(M) \iff (\forall e : \mathsf{BHist}.\ M(e) \to \mathsf{Edge}_G(e)) \wedge (\forall e, e', v : \mathsf{BHist}.\ M(e) \wedge M(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{I}$$

Thus a matching edge predicate is a graph-edge subset whose selected edges have no shared incident vertex except along the edge classifier. For edge predicates N *and* M*, define*

$$\mathsf{FinSetEdgeSubset}_{G,\sim_E}(N, M) \iff \exists xs : \mathsf{HistSpine}.\ \mathsf{FinSetSpineEnumerates}_{\mathsf{Edge}_G,\sim_E}(xs, N) \wedge \forall e : \mathsf{BHist}.\ N(e) \to M($$

The first component is the finite enumeration witness supplied by the $\mathsf{FinSet}^{\uparrow}$ *machinery of Definition 112.17; the second component is ordinary edge-subset inclusion into* M*.*

Lean4 (definition) `BEDC.Derived.MatchingUp.MatchingEdgeSet`
Lean4 (definition) `BEDC.Derived.MatchingUp.FinSetEdgeSubset`

Theorem 234.2 (Matching finite-subset closure)**.** *Let* G *be a* $\mathsf{Graph}^{\uparrow}$ *carrier with edge classifier* $\sim_E$*. If* $M, N : \mathsf{BHist} \to \mathsf{Prop}$ *satisfy*

$$\mathsf{MatchingEdgeSet}_G(M) \qquad \textit{and} \qquad \mathsf{FinSetEdgeSubset}_{G,\sim_E}(N, M),$$

then

$$\mathsf{MatchingEdgeSet}_G(N).$$

Equivalently, every $\mathsf{FinSet}^{\uparrow}$ *edge subset of a* $\mathsf{Matching}^{\uparrow}$ *edge set is again a* $\mathsf{Matching}^{\uparrow}$ *edge set in the same* $\mathsf{Graph}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet finite subset closed`

Proof. Unpack Definition 234.1. The matching hypothesis for M supplies two rows: first, every M-edge is a graph edge of G; second, any two M-edges incident to the same carried vertex are classified by $\sim_E$ as the same edge. The finite-subset hypothesis supplies a finite spine xs with

$$\mathsf{FinSetSpineEnumerates}_{\mathsf{Edge}_G,\sim_E}(xs, N)$$

and an inclusion row

$$\forall e : \mathsf{BHist}.\ N(e) \to M(e).$$

The spine witness records that N is the $\mathsf{FinSet}^{\uparrow}$ edge subset; the hereditary argument uses the displayed inclusion row.

To prove the graph-edge row for N, fix e and assume $N(e)$. The inclusion row gives $M(e)$, and the graph-edge row for M gives $\mathsf{Edge}_G(e)$.

To prove the no-shared-vertex row for N, fix e, e', v and assume

$$N(e) \wedge N(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e').$$

The inclusion row gives $M(e)$ and $M(e')$. Keeping the same vertex and incidence witnesses, apply the no-shared-vertex row for M to obtain

$$e \sim_E e'.$$

These two rows are exactly $\mathsf{MatchingEdgeSet}_G(N)$ by Definition 234.1. □

Theorem 234.3 (Matching compatible union closure)**.** *Let G be a $\mathsf{Graph}^{\uparrow}$ carrier with edge classifier $\sim_E$. Let $M, N : \mathsf{BHist} \to \mathsf{Prop}$ be edge predicates, and define the pointwise union predicate $U : \mathsf{BHist} \to \mathsf{Prop}$ by*

$$U(e) \iff M(e) \vee N(e).$$

Assume

$$\mathsf{MatchingEdgeSet}_G(M) \qquad \textit{and} \qquad \mathsf{MatchingEdgeSet}_G(N),$$

and assume the cross-incident compatibility condition

$$\forall e, e', v : \mathsf{BHist}.\ ((M(e) \wedge N(e')) \vee (N(e) \wedge M(e'))) \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e') \to e \sim_E e'.$$

Then

$$\mathsf{MatchingEdgeSet}_G(U).$$

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet compatible union closed`

Proof. Unpack Definition 234.1. The matching hypotheses for M and N each supply a graph-edge row and a no-shared-vertex row.

For the graph-edge row of U, fix $e : \mathsf{BHist}$ and assume $U(e)$. By the defining display for U, either $M(e)$ or $N(e)$ holds. In the first case, the graph-edge row for M gives $\mathsf{Edge}_G(e)$. In the second case, the graph-edge row for N gives $\mathsf{Edge}_G(e)$.

For the no-shared-vertex row of U, fix $e, e', v : \mathsf{BHist}$ and assume

$$U(e) \wedge U(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e').$$

The two U-memberships split into four cases. If $M(e)$ and $M(e')$ hold, the no-shared-vertex row for M, together with the same vertex and incidence witnesses, gives $e \sim_E e'$. If $N(e)$ and $N(e')$ hold, the no-shared-vertex row for N gives $e \sim_E e'$.

It remains to handle the two cross cases. If $M(e)$ and $N(e')$ hold, the left disjunct in the cross-incident compatibility hypothesis applies to the same e, e', v and gives $e \sim_E e'$. If $N(e)$ and $M(e')$ hold, the right disjunct applies directly and gives $e \sim_E e'$. Thus every pair of U-selected edges incident to the same carried vertex is classified by $\sim_E$ as the same edge.

The graph-edge row and the no-shared-vertex row are precisely $\mathsf{MatchingEdgeSet}_G(U)$ by Definition 234.1. □

Theorem 234.4 (Empty edge predicate matching)**.** *Let G be a $\mathsf{Graph}^\uparrow$ carrier with vertex predicate Vert_G, edge predicate Edge_G, incidence predicate Inc_G, and edge classifier $\sim_E$. Let $Z : \mathsf{BHist} \to \mathsf{Prop}$ be an edge-selection predicate satisfying*

$$\forall e : \mathsf{BHist}.\ Z(e) \to \bot.$$

Then

$$\mathsf{MatchingEdgeSet}_G(Z).$$

In particular, the empty predicate $M_\varnothing(e) \iff \bot$ is a $\mathsf{Matching}^\uparrow$ edge set in G.

Proof. Unpack Definition 234.1. The matching claim has two rows: graph-edge support for every selected edge, and the no-shared-vertex implication for pairs of selected edges.

For graph-edge support, fix $e : \mathsf{BHist}$ and assume $Z(e)$. The displayed emptiness row gives $\bot$, and hence gives $\mathsf{Edge}_G(e)$.

For the no-shared-vertex row, fix $e, e', v : \mathsf{BHist}$ and assume

$$Z(e) \wedge Z(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e').$$

The first component gives $Z(e)$, so the displayed emptiness row again gives a contradiction. Therefore the required conclusion $e \sim_E e'$ follows. These two vacuous rows are exactly $\mathsf{MatchingEdgeSet}_G(Z)$.

Taking $Z = M_\varnothing$, the defining equivalence $M_\varnothing(e) \iff \bot$ supplies $M_\varnothing(e) \to \bot$ for every e, so the displayed result applies to the empty edge predicate. □

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet e0 empty absurd predicate`

Theorem 234.5 (Singleton edge predicate matching)**.** *Let G be a $\mathsf{Graph}^\uparrow$ carrier with edge classifier $\sim_E$, and let $\mathsf{cert}_E : \mathsf{NameCert}(\mathsf{Edge}_G, \sim_E)$ be the edge naming certificate in the sense of Definition 9.7. Fix a carried edge $e_\star : \mathsf{BHist}$ such that*

$$\mathsf{Edge}_G(e_\star).$$

Define the edge predicate $M_{e_\star} : \mathsf{BHist} \to \mathsf{Prop}$ by

$$M_{e_\star}(e) \iff e \sim_E e_\star.$$

Then

$$\mathsf{MatchingEdgeSet}_G(M_{e_\star}).$$

In particular, the $\sim_E$-class of any carried edge is a $\mathsf{Matching}^\uparrow$ edge set in G.

Proof. Unpack Definition 234.1. To prove $\mathsf{MatchingEdgeSet}_G(M_{e_\star})$, it is enough to prove the graph-edge row and the no-shared-vertex row.

For the graph-edge row, fix $e : \mathsf{BHist}$ and assume $M_{e_\star}(e)$. By definition of $M_{e_\star}$, this means

$$e \sim_E e_\star.$$

The symmetry field of the naming certificate from Definition 9.7 reverses this comparison and gives

$$e_\star \sim_E e.$$

Apply the carrier-transport field of the same certificate to the carried-edge witness $\mathsf{Edge}_G(e_\star)$. This transports edge membership along $e_\star \sim_E e$ and yields

$$\mathsf{Edge}_G(e).$$

Thus every $M_{e_\star}$-selected edge is a graph edge of G.

For the no-shared-vertex row, fix $e, e', v : \mathsf{BHist}$ and assume

$$M_{e_\star}(e) \wedge M_{e_\star}(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e').$$

Project the first two components. By the definition of $M_{e_\star}$, they give

$$e \sim_E e_\star \qquad \text{and} \qquad e' \sim_E e_\star.$$

Use symmetry from Definition 9.7 on the second comparison to obtain

$$e_\star \sim_E e'.$$

Now apply classifier transitivity from Definition 9.8 to the comparisons $e \sim_E e_\star$ and $e_\star \sim_E e'$. This yields

$$e \sim_E e'.$$

The vertex and incidence witnesses are retained from the hypothesis but are not needed further, because every selected edge lies in the same classifier class of $e_\star$.

The graph-edge row and the no-shared-vertex row are exactly the two clauses of $\mathsf{MatchingEdgeSet}_G(M_{e_\star})$ in Definition 234.1. □

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet singleton edge predicate`
Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet singleton classifier closed`

Theorem 234.6 (Matching augmentation symmetric-difference decomposition)**.** *Let G be a $\mathsf{Graph}^\uparrow$ carrier with vertex predicate Vert_G, edge predicate Edge_G, incidence predicate Inc_G, and edge classifier $\sim_E$. For two finite $\mathsf{Matching}^\uparrow$ edge predicates M, N in G, the symmetric-difference ledger decomposes into carried edge rows: each displayed edge e has $\mathsf{Edge}_G(e)$, an endpoint-incidence witness row through Inc_G, and one of the two finite membership certificates $M(e)$ or $N(e)$ but not both. The ledger contains no unrecorded host edge set beyond these BHist rows.*

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingAugmentation symmetric difference decomposition`

Proof. Read the finite edge-subset rows for M and N from Definition 234.1. Each selected edge is a carried $\mathsf{Graph}^{\uparrow}$ edge, and each pair of selected incident edges is governed by the no-shared-vertex row.

Construct the symmetric-difference ledger by scanning the two finite membership ledgers. An edge in exactly one ledger is copied with its carried edge witness; an edge in both ledgers is omitted from the toggled surface and retained only as the classifier comparison explaining the cancellation.

The endpoint-incidence witnesses are the incidence rows already stored by the $\mathsf{Graph}^{\uparrow}$ carrier. Because the decomposition stores membership side, carried edge status, incidence, and classifier comparisons explicitly, no external host edge set is introduced. □

Theorem 234.7 (Matching augmenting edge toggle closure)**.** *Let M be a $\mathsf{Matching}^{\uparrow}$ edge predicate in a $\mathsf{Graph}^{\uparrow}$ carrier G, and let $e_\star : \mathsf{BHist}$ be a carried edge. Suppose the singleton predicate $M_{e_\star}(e) \iff e \sim_E e_\star$ is incident-disjoint from M except along the edge classifier $\sim_E$. Then toggling the singleton predicate into M gives another $\mathsf{MatchingEdgeSet}_G$: the resulting predicate is the compatible union of M with the singleton edge predicate, with classifier-equal overlap cancelled by the same finite toggle ledger.*

Proof. First apply Definition 234.5 to the carried edge $e_\star$. This gives the singleton matching predicate whose selected edges are exactly the $\sim_E$-class of $e_\star$.

Next use the incident-disjointness hypothesis to verify compatibility with M. Whenever a vertex is incident to one edge selected by M and one edge selected by the singleton predicate, the hypothesis collapses the two edges by $\sim_E$, which is precisely the compatibility condition needed by Definition 234.3.

Finally apply the compatible-union closure theorem and read the finite toggle ledger from Definition 234.6. Edges in the classifier-equal overlap are cancelled by the toggle ledger, and all remaining selected edges keep their $\mathsf{Graph}^{\uparrow}$ carried-edge and incidence rows. □

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet toggle singleton compatible union closed`

Theorem 234.8 (Matching bipartite cover consumer boundary)**.** *A bipartite cover consumer of the finite augmenting surface may read only the finite edge subset, the no-shared-vertex row of Definition 234.1, the incidence* BHist *rows of* $\mathsf{Graph}^{\uparrow}$*, the carried graph rows, and the* $\mathsf{FinSet}^{\uparrow}$ *provenance of the finite ledger. It may not read an ambient host edge set, an uncarried vertex quotient, or any matching predicate not presented by the displayed finite* BHist *ledger.*

Proof. The symmetric-difference and toggle theorems expose the finite matching data as listed rows: carried edges, incidence witnesses, membership side, classifier comparisons, and no-shared-vertex certificates.

A bipartite cover construction uses exactly those rows to decide which endpoints are covered by selected edges and which alternating steps are available. The $\mathsf{FinSet}^{\uparrow}$ provenance bounds the search to the displayed finite ledger, while the $\mathsf{Graph}^{\uparrow}$ incidence rows provide all endpoint reads.

Thus the consumer boundary is the finite BEDC ledger itself. Since every edge and vertex access is mediated by BHist and the graph certificate, there is no permission to inspect an ambient host graph or an unrecorded edge collection. □

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet bipartite cover consumer boundary`

Theorem 234.9 (Matching abstract sub-predicate closure)**.** *Let G be a $\mathsf{Graph}^{\uparrow}$ carrier with edge classifier $\sim_E$. Let $M, N : \mathsf{BHist} \to \mathsf{Prop}$ be edge predicates. If*

$$\mathsf{MatchingEdgeSet}_G(M)$$

and

$$\forall e : \mathsf{BHist}.\ N(e) \to M(e),$$

then

$$\mathsf{MatchingEdgeSet}_G(N).$$

Thus the matching condition is hereditary under ordinary predicate inclusion; no finite-spine enumeration of N is required.

Lean4 ✓ `BEDC.Derived.MatchingUp.MatchingEdgeSet abstract subpredicate closed`

Proof. Unpack Definition 234.1. The matching hypothesis for M supplies the graph-edge row

$$\forall e : \mathsf{BHist}.\ M(e) \to \mathsf{Edge}_G(e)$$

and the no-shared-vertex row

$$\forall e, e', v : \mathsf{BHist}.\ M(e) \wedge M(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e') \to e \sim_E e'.$$

The second hypothesis supplies the inclusion row from N into M.

For the graph-edge row of N, fix $e : \mathsf{BHist}$ and assume $N(e)$. The inclusion row gives $M(e)$, and the graph-edge row for M gives $\mathsf{Edge}_G(e)$.

For the no-shared-vertex row of N, fix $e, e', v : \mathsf{BHist}$ and assume

$$N(e) \wedge N(e') \wedge \mathsf{Vert}_G(v) \wedge \mathsf{Inc}_G(v, e) \wedge \mathsf{Inc}_G(v, e').$$

Projecting the first two components gives $N(e)$ and $N(e')$. The inclusion row turns these into $M(e)$ and $M(e')$. Keeping the same vertex and incidence witnesses, apply the no-shared-vertex row for M to obtain

$$e \sim_E e'.$$

The graph-edge row and the no-shared-vertex row are exactly $\mathsf{MatchingEdgeSet}_G(N)$ by Definition 234.1. □

Theorem 234.10 (Matching standard finite-edge bridge)**.** *The public finite-edge-subset matching certificate bridges to the standard finite matching interface exactly through the displayed* $\mathsf{Graph}^{\uparrow}$ *carrier,* $\mathsf{FinSet}^{\uparrow}$ *finite ledger,* BHist *edge predicates, incidence rows, no-shared-vertex classifier, augmentation and toggle rows, and bipartite-cover consumer boundary.*

The bridge exposes finite selected edges, carried incidence witnesses, classifier comparisons for shared vertices, compatible augmentation toggles, and the public cover-side consumer boundary. It exposes no ambient host edge set, maximum-matching optimality theorem, vertex-cover duality theorem, or external graph representation.

Proof. Start with Definition 234.2. It binds the selected edge predicate to the displayed $\mathsf{Graph}^{\uparrow}$ carrier and the $\mathsf{FinSet}^{\uparrow}$ finite ledger, so every selected edge read is a carried graph-edge read.

Next combine the closure rows. Use Definition 234.3, Definition 234.4, and Definition 234.5. These rows give the finite empty, singleton, and compatible-union entry points for the standard finite matching interface.

Now add the consumer operations. Apply Definition 234.6, Definition 234.7, and Definition 234.8. They keep alternating augmentation, finite toggling, and the bipartite-cover read boundary inside the same edge predicate, incidence, and classifier ledger.

Finally apply Definition 234.9. It shows that sub-predicate reads remain matching reads without adding an unrecorded host edge set. The listed rows are therefore exactly the finite-edge standard interface available from the public certificate, and the excluded graph-theory objects have no coordinate in the packet. □

Theorem 234.11 (Matching alternating-cover consumer example)**.** *Let G be a displayed* $\mathsf{Graph}^{\uparrow}$ *carrier, and let M be a finite* $\mathsf{FinSet}^{\uparrow}$ *edge predicate satisfying* $\mathsf{MatchingEdgeSet}_G(M)$*. An alternating finite edge path together with a bipartite-cover side row consumes only the selected edge predicate, the displayed incidence rows, the edge classifier, the augmentation toggle rows, and the* Pkg/NameCert *provenance already exposed by the public matching certificate.*

The resulting consumer packet is again a finite BHist *surface. It records the selected path edges, incidence witnesses, membership-side rows, classifier comparisons, toggle ledger, cover-side rows, and provenance rows. It does not claim maximum matching optimality or vertex-cover duality.*

Proof. First bind the selected edge predicate to the public finite carrier. Apply Definition 234.2. Every edge read by the alternating path is therefore a carried $\mathsf{Graph}^{\uparrow}$ edge with a finite $\mathsf{FinSet}^{\uparrow}$ provenance row.

Next expose the alternating path ledger. Use Definition 234.6 to split the path steps into selected and unselected membership-side reads, and then apply Definition 234.7 to keep the toggle surface inside the same matching predicate and edge classifier.

Now add the cover-side consumer. Apply Definition 234.8. It restricts the cover read to selected edges, incidence rows, no-shared-vertex classifier comparisons, carried graph rows, and finite ledger provenance.

Finally apply Definition 234.10. The bridge packages the alternating path, toggle, and cover-side reads as one finite BHist consumer surface. Since the packet contains no coordinate for an optimality theorem or a duality theorem, neither conclusion is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{Matching}^{\uparrow}}$**.**

Constructive story: A matching is generated inside a displayed $\mathsf{Graph}^{\uparrow}$ carrier by choosing an edge predicate and then bounding the chosen edges with a $\mathsf{FinSet}^{\uparrow}$ finite-spine witness. The carrier accepts a predicate only when every selected history is a graph edge and any two selected edges incident to the same carried vertex collapse through the graph edge classifier; predicates with uncarried edges or unclassified shared vertices are excluded. Two matching presentations are compared through the edge classifier and the finite-subset or compatible-union ledgers, so a singleton edge class, a finite subpredicate, and a compatible toggle all remain inside the same graph-certified surface rather than becoming host edge sets. The source is the finite edge subset over the graph carrier, the pattern records hereditary subset closure, compatible union closure, empty and singleton edge predicates, augmentation toggles, and the bipartite-cover consumer boundary, and the ledger keeps selected edges, incidence witnesses, membership side, finite enumeration, and classifier comparisons visible. The five name-certificate obligations are supplied by the empty or singleton carried edge examples for nonemptiness, by the edge certificate for reflexivity, symmetry, and transitivity, and by finite-subset and compatible-union closure for repacking classified selected edges as matchings. A history h lies in the signature gap of a $\mathsf{Matching}^{\uparrow}$ token p when the source-domain accepts its finite edge predicate, the signature relation pairs it with an answer matching and evidence ledger, and the token-introduction relation names that answer as p; concretely, every edge selected by h is a carried graph edge and any shared-vertex conflict is resolved exactly as p records. The chapter registers this finite incidence construction under the name $\mathsf{Matching}^{\uparrow}$; maximum matching, vertex-cover duality, external graph-theory bridges, and ambient host edge sets remain outside the scoped certificate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Matching}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public finite-edge-subset matching certificate and its standard finite-edge bridge bind the finite edge subset, incidence, compatibility, augmentation, toggle, and bipartite-cover consumer rows to the displayed $\mathsf{Graph}^{\uparrow}$ carrier, $\mathsf{FinSet}^{\uparrow}$ finite-subset witness, BHist edge predicates, and edge classifier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Matching}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MatchingUp.MatchingEdgeSet\_finite\_subset\_closed`

Bridge status: paperBridge

Not claimed: Maximum matching, vertex-cover duality, and ambient host edge sets are not claimed by this chapter.

Upgrade path: No further paper-side theory closure upgrade remains for this chapter; formal bridge checking remains a separate verification-axis task.

Chapter 235

A Concrete Naming Certificate for $\mathsf{LPDuality}^{\uparrow}$

The linear-programming-duality interface $\mathsf{LPDuality}^{\uparrow}$ packages a primal/dual pair of linear programs together with strong-duality and complementary-slackness certificates over an ordered field. It depends on $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ (chapter 208) and $\mathsf{NameCert}_{\mathsf{Polytope}^{\uparrow}}$ (chapter 209).

235.1 Carrier definition

235.2 The certificate

Definition 235.1 (Finite $\mathsf{LPDuality}^{\uparrow}$ ordered-field feasibility row)**.** *An* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field consists of finite row-position and column-position carriers* I *and* J*, a scalar carrier* F *equipped with the* $\mathsf{Field}^{\uparrow}$ *operations and classifier of chapter 45, and a* $\mathsf{TotalOrder}^{\uparrow}$ *comparison on the same scalar carrier and classifier as in chapter 54. Write* $+_F$*,* $\cdot_F$*,* 0_F*,* $\sim_F$*, and* $\preceq_F$ *for these data, with* $u \preceq_F v$ *meaning that* u *is less than or classifier-equal to* v*. The nonnegative scalar predicate is*

$$\mathsf{NonNeg}_F(a) \iff 0_F \preceq_F a.$$

The scalar order rows used by the certificate are classifier stability of $\preceq_F$*,*

$$u \sim_F u' \wedge v \sim_F v' \wedge u \preceq_F v \Longrightarrow u' \preceq_F v',$$

left and right multiplication monotonicity by nonnegative scalars,

$$\mathsf{NonNeg}_F(a) \wedge u \preceq_F v \Longrightarrow (a \cdot_F u \preceq_F a \cdot_F v) \wedge (u \cdot_F a \preceq_F v \cdot_F a),$$

and finite-sum monotonicity over every finite position carrier K*,*

$$\Big(\forall k \in K.\ r_k \preceq_F s_k\Big) \Longrightarrow \sum_{k \in K} r_k \preceq_F \sum_{k \in K} s_k.$$

For coefficients $A : I \times J \to F$*, right-hand side* $b : I \to F$*, and cost vector* $c : J \to F$*, define*

$$\alpha_i(x) := \sum_{j \in J} A_{ij} \cdot_F x_j, \qquad \beta_j(y) := \sum_{i \in I} y_i \cdot_F A_{ij}.$$

The finite distributivity and interchange row records the classifier comparison

$$\sum_{j\in J} \beta_j(y) \cdot_F x_j \sim_F \sum_{i\in I} y_i \cdot_F \alpha_i(x).$$

The primal and dual feasible carriers, read as the row descriptions of the corresponding $\mathsf{Polytope}^{\uparrow}$ *feasible sets over the convex substrate of chapter 208 and chapter 209, are*

$$\mathsf{PrFeas}_{A,b}(x) \iff \Big(\forall j \in J.\ \mathsf{NonNeg}_F(x_j)\Big) \wedge \Big(\forall i \in I.\ \alpha_i(x) \preceq_F b_i\Big)$$

and

$$\mathsf{DuFeas}_{A,b,c}(y) \iff \Big(\forall i \in I.\ \mathsf{NonNeg}_F(y_i)\Big) \wedge \Big(\forall j \in J.\ c_j \preceq_F \beta_j(y)\Big).$$

The objective endpoints are

$$\mathsf{PrObj}_{A,c}(x) := \sum_{j\in J} c_j \cdot_F x_j, \qquad \mathsf{DuObj}_b(y) := \sum_{i\in I} y_i \cdot_F b_i.$$

Theorem 235.2 (LPDuality root obligation surface)**.** *The root certificate surface for* $\mathsf{LPDuality}^{\uparrow}$ *consists of a* $\mathsf{ConvexSet}^{\uparrow}$ *feasible-region carrier,* $\mathsf{Field}^{\uparrow}$ *scalar objective rows,* $\mathsf{Polytope}^{\uparrow}$ *face and witness rows,* $\mathsf{TotalOrder}^{\uparrow}$ *comparison rows,* hsame *classifier transport,* Cont *ledger movement, and* Pkg *provenance. Every admitted dual-certificate row projects to these fields: primal and dual feasibility are read from the feasible-region and polytope rows, scalar objective endpoints are read from the ordered-field rows, comparisons are read from the total-order rows, and all transport is recorded by the displayed* hsame*,* Cont*, and* Pkg *ledgers. No host linear-programming primitive, ambient matrix equality, or quotient vector space is exported by this surface.*

Proof. Project an admitted dual-certificate row through Definition 235.1. The finite row and column carriers, feasible predicates, scalar sums, and objective endpoints are exactly the displayed certificate fields.

The feasible-region and face witnesses are read through the cited $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$ certificates, while scalar operations and objective comparisons are read through the $\mathsf{Field}^{\uparrow}$ and $\mathsf{TotalOrder}^{\uparrow}$ rows. Classifier movement uses the explicit hsame stability rows, and ledger movement uses the displayed Cont and Pkg provenance rows.

Repacking these projections gives a finite BHist surface with no additional constructor. Since the definition exposes no host optimization solver, matrix equality primitive, or quotient-space row, every admitted dual-certificate row is exhausted by the listed BEDC fields. □

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityFiniteOrderedFieldFeasibilityRow root obligation surface`

Theorem 235.3 (LPDuality feasible weak duality)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. If* x *is primal feasible and* y *is dual feasible,*

$$\mathsf{PrFeas}_{A,b}(x) \wedge \mathsf{DuFeas}_{A,b,c}(y),$$

then the primal objective value at x *is less than or classifier-equal to the dual objective value at* y*:*

$$\mathsf{PrObj}_{A,c}(x) \preceq_F \mathsf{DuObj}_b(y).$$

Equivalently, the $\mathsf{TotalOrder}^{\uparrow}$ *comparison between the two objective endpoints cannot select the strict-right branch.*

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityFeasibleWeakDuality prefix order`

Proof. Unpack the feasibility predicates of Definition 235.1. From primal feasibility, for each row position $i \in I$ there is a constraint comparison

$$\alpha_i(x) \preceq_F b_i,$$

and from dual feasibility there is a nonnegativity witness $\mathsf{NonNeg}_F(y_i)$. Multiplication monotonicity by the nonnegative scalar y_i gives

$$y_i \cdot_F \alpha_i(x) \preceq_F y_i \cdot_F b_i.$$

Finite-sum monotonicity over I yields

$$\sum_{i \in I} y_i \cdot_F \alpha_i(x) \preceq_F \sum_{i \in I} y_i \cdot_F b_i.$$

The right-hand side is $\mathsf{DuObj}_b(y)$ by definition.

From dual feasibility, for each column position $j \in J$ there is a comparison

$$c_j \preceq_F \beta_j(y),$$

and from primal feasibility there is a nonnegativity witness $\mathsf{NonNeg}_F(x_j)$. Right multiplication monotonicity by the nonnegative scalar x_j gives

$$c_j \cdot_F x_j \preceq_F \beta_j(y) \cdot_F x_j.$$

Finite-sum monotonicity over J yields

$$\sum_{j \in J} c_j \cdot_F x_j \preceq_F \sum_{j \in J} \beta_j(y) \cdot_F x_j.$$

The left-hand side is $\mathsf{PrObj}_{A,c}(x)$ by definition.

The finite distributivity and interchange row gives the classifier comparison

$$\sum_{j \in J} \beta_j(y) \cdot_F x_j \sim_F \sum_{i \in I} y_i \cdot_F \alpha_i(x).$$

Classifier stability of $\preceq_F$, applied to reflexivity at the left endpoint of this comparison, turns it into the order comparison

$$\sum_{j \in J} \beta_j(y) \cdot_F x_j \preceq_F \sum_{i \in I} y_i \cdot_F \alpha_i(x).$$

Composing the three displayed comparisons by transitivity of the $\mathsf{TotalOrder}^{\uparrow}$ order gives

$$\mathsf{PrObj}_{A,c}(x) \preceq_F \mathsf{DuObj}_b(y).$$

This is exactly the weak-duality comparison for the feasible primal and dual witnesses. □

Corollary 235.4 (LPDuality weak-duality equality gives optimality)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Suppose x is primal feasible and y is dual feasible, and suppose their objective endpoints are equal under the scalar classifier:*

$$\mathsf{PrFeas}_{A,b}(x) \wedge \mathsf{DuFeas}_{A,b,c}(y), \qquad \mathsf{PrObj}_{A,c}(x) \sim_F \mathsf{DuObj}_b(y).$$

Then x is primal optimal and y is dual optimal in the row sense:

$$\forall x'.\ \mathsf{PrFeas}_{A,b}(x') \Longrightarrow \mathsf{PrObj}_{A,c}(x') \preceq_F \mathsf{PrObj}_{A,c}(x),$$

and

$$\forall y'.\ \mathsf{DuFeas}_{A,b,c}(y') \Longrightarrow \mathsf{DuObj}_b(y) \preceq_F \mathsf{DuObj}_b(y').$$

Proof. For the primal optimality clause, fix x' with $\mathsf{PrFeas}_{A,b}(x')$. Applying Definition 235.3 to x' and the given dual feasible witness y gives

$$\mathsf{PrObj}_{A,c}(x') \preceq_F \mathsf{DuObj}_b(y).$$

The scalar classifier is reflexive at $\mathsf{PrObj}_{A,c}(x')$, and symmetry of the displayed classifier equality gives

$$\mathsf{DuObj}_b(y) \sim_F \mathsf{PrObj}_{A,c}(x).$$

The classifier-stability row for $\preceq_F$ in Definition 235.1 transports the right endpoint of the weak-duality comparison, yielding

$$\mathsf{PrObj}_{A,c}(x') \preceq_F \mathsf{PrObj}_{A,c}(x).$$

Since x' was arbitrary, the first universal clause follows.

For the dual optimality clause, fix y' with $\mathsf{DuFeas}_{A,b,c}(y')$. Applying Definition 235.3 to the given primal feasible witness x and y' gives

$$\mathsf{PrObj}_{A,c}(x) \preceq_F \mathsf{DuObj}_b(y').$$

The displayed classifier equality gives

$$\mathsf{PrObj}_{A,c}(x) \sim_F \mathsf{DuObj}_b(y),$$

and reflexivity of the scalar classifier holds at $\mathsf{DuObj}_b(y')$. The same classifier-stability row transports the left endpoint of the weak-duality comparison, yielding

$$\mathsf{DuObj}_b(y) \preceq_F \mathsf{DuObj}_b(y').$$

Since y' was arbitrary, the second universal clause follows. □

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityWeakDualityEquality optimality`

Theorem 235.5 (LPDuality complementary slackness gives objective equality)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Suppose x is primal feasible and y is dual feasible,*

$$\mathsf{PrFeas}_{A,b}(x) \wedge \mathsf{DuFeas}_{A,b,c}(y),$$

and suppose the two complementary-slackness product families are classifier-equal termwise:

$$\forall j \in J.\ c_j \cdot_F x_j \sim_F \beta_j(y) \cdot_F x_j, \qquad \forall i \in I.\ y_i \cdot_F \alpha_i(x) \sim_F y_i \cdot_F b_i.$$

Then the primal and dual objective endpoints are classifier-equal,

$$\mathsf{PrObj}_{A,c}(x) \sim_F \mathsf{DuObj}_b(y).$$

Moreover, x is primal optimal and y is dual optimal in the row sense:

$$\forall x'.\ \mathsf{PrFeas}_{A,b}(x') \Longrightarrow \mathsf{PrObj}_{A,c}(x') \preceq_F \mathsf{PrObj}_{A,c}(x),$$

and

$$\forall y'.\ \mathsf{DuFeas}_{A,b,c}(y') \Longrightarrow \mathsf{DuObj}_b(y) \preceq_F \mathsf{DuObj}_b(y').$$

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityComplementarySlackness objective equality`

Proof. For each column position $j \in J$, order reflexivity gives

$$c_j \cdot_F x_j \preceq_F c_j \cdot_F x_j \qquad \text{and} \qquad \beta_j(y) \cdot_F x_j \preceq_F \beta_j(y) \cdot_F x_j.$$

Transporting these reflexive comparisons by the classifier-stability row for $\preceq_F$ and by the displayed column slack equality, in both directions, gives

$$c_j \cdot_F x_j \preceq_F \beta_j(y) \cdot_F x_j \qquad \text{and} \qquad \beta_j(y) \cdot_F x_j \preceq_F c_j \cdot_F x_j.$$

Finite-sum monotonicity over J therefore yields the opposing comparisons

$$\sum_{j \in J} c_j \cdot_F x_j \preceq_F \sum_{j \in J} \beta_j(y) \cdot_F x_j, \qquad \sum_{j \in J} \beta_j(y) \cdot_F x_j \preceq_F \sum_{j \in J} c_j \cdot_F x_j.$$

Antisymmetry of the ordered-field order gives the classifier equality

$$\mathsf{PrObj}_{A,c}(x) \sim_F \sum_{j \in J} \beta_j(y) \cdot_F x_j.$$

The same argument over the row positions uses the displayed row slack equality. For each $i \in I$, classifier stability transports the two reflexive order comparisons into

$$y_i \cdot_F \alpha_i(x) \preceq_F y_i \cdot_F b_i \qquad \text{and} \qquad y_i \cdot_F b_i \preceq_F y_i \cdot_F \alpha_i(x).$$

Finite-sum monotonicity over I, followed by antisymmetry, gives

$$\sum_{i \in I} y_i \cdot_F \alpha_i(x) \sim_F \mathsf{DuObj}_b(y).$$

The finite distributivity and interchange row in Definition 235.1 supplies

$$\sum_{j \in J} \beta_j(y) \cdot_F x_j \sim_F \sum_{i \in I} y_i \cdot_F \alpha_i(x).$$

Transitivity of the scalar classifier composes the three displayed classifier equalities into

$$\mathsf{PrObj}_{A,c}(x) \sim_F \mathsf{DuObj}_b(y).$$

The feasibility hypotheses and this objective equality are precisely the hypotheses of Definition 235.4. Applying that corollary gives the stated primal and dual optimality clauses. □

Theorem 235.6 (LPDuality dual feasibility binary convex closure)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $y, y' : I \to F$ *be dual feasible witnesses, and let* $\lambda, \mu : F$ *be scalar coefficients such that*

$$\mathsf{DuFeas}_{A,b,c}(y) \wedge \mathsf{DuFeas}_{A,b,c}(y') \wedge \mathsf{NonNeg}_F(\lambda) \wedge \mathsf{NonNeg}_F(\mu) \wedge (\lambda +_F \mu) \sim_F 1_F.$$

Define $\eta : I \to F$ *componentwise by*

$$\eta_i := \lambda \cdot_F y_i +_F \mu \cdot_F y'_i \qquad (i \in I).$$

Then

$$\mathsf{DuFeas}_{A,b,c}(\eta).$$

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityDualFeasibility binary convex closure`

Proof. Unpack the two dual-feasibility hypotheses from Definition 235.1. They give

$$\forall i \in I.\ \mathsf{NonNeg}_F(y_i), \qquad \forall i \in I.\ \mathsf{NonNeg}_F(y'_i),$$

and

$$\forall j \in J.\ c_j \preceq_F \beta_j(y), \qquad \forall j \in J.\ c_j \preceq_F \beta_j(y').$$

We verify the two clauses of $\mathsf{DuFeas}_{A,b,c}(\eta)$.

First fix $i \in I$. Since $\mathsf{NonNeg}_F(y_i)$ and $\mathsf{NonNeg}_F(y'_i)$ mean $0_F \preceq_F y_i$ and $0_F \preceq_F y'_i$, left multiplication monotonicity by the nonnegative scalars λ and μ gives

$$\lambda \cdot_F 0_F \preceq_F \lambda \cdot_F y_i, \qquad \mu \cdot_F 0_F \preceq_F \mu \cdot_F y'_i.$$

Apply finite-sum monotonicity over a two-point carrier to these two comparisons. This yields

$$(\lambda \cdot_F 0_F) +_F (\mu \cdot_F 0_F) \preceq_F (\lambda \cdot_F y_i) +_F (\mu \cdot_F y'_i).$$

The field rows from chapter 45 identify the left endpoint with 0_F, using zero multiplication and the additive unit law. By the definition of η_i, the right endpoint is η_i. Classifier stability of $\preceq_F$ therefore gives

$$0_F \preceq_F \eta_i.$$

Thus $\mathsf{NonNeg}_F(\eta_i)$ holds for every $i \in I$.

Next fix $j \in J$. Dual feasibility of y and y' gives

$$c_j \preceq_F \beta_j(y), \qquad c_j \preceq_F \beta_j(y').$$

Left multiplication monotonicity by the nonnegative scalars λ and μ yields

$$\lambda \cdot_F c_j \preceq_F \lambda \cdot_F \beta_j(y), \qquad \mu \cdot_F c_j \preceq_F \mu \cdot_F \beta_j(y').$$

Finite-sum monotonicity over a two-point carrier gives

$$(\lambda \cdot_F c_j) +_F (\mu \cdot_F c_j) \preceq_F (\lambda \cdot_F \beta_j(y)) +_F (\mu \cdot_F \beta_j(y')).$$

On the left, distributivity and the coefficient-sum hypothesis give the classifier chain

$$(\lambda \cdot_F c_j) +_F (\mu \cdot_F c_j) \sim_F (\lambda +_F \mu) \cdot_F c_j \sim_F 1_F \cdot_F c_j \sim_F c_j.$$

On the right, expand the definition of β_j and use distributivity, associativity, and finite-sum compatibility with the scalar classifier from chapter 45:

$$(\lambda \cdot_F \beta_j(y)) +_F (\mu \cdot_F \beta_j(y')) \sim_F \sum_{i \in I} (\lambda \cdot_F y_i) \cdot_F A_{ij} +_F \sum_{i \in I} (\mu \cdot_F y'_i) \cdot_F A_{ij}$$

and hence

$$(\lambda \cdot_F \beta_j(y)) +_F (\mu \cdot_F \beta_j(y')) \sim_F \sum_{i \in I} (\lambda \cdot_F y_i +_F \mu \cdot_F y'_i) \cdot_F A_{ij} = \beta_j(\eta).$$

Classifier stability of $\preceq_F$ transports the displayed order comparison along these left and right classifier equalities, yielding

$$c_j \preceq_F \beta_j(\eta).$$

Thus the dual constraint clause holds for every $j \in J$.

We have shown both

$$\forall i \in I.\ \mathsf{NonNeg}_F(\eta_i)$$

and

$$\forall j \in J.\ c_j \preceq_F \beta_j(\eta).$$

By the definition of dual feasibility in Definition 235.1, this is exactly

$$\mathsf{DuFeas}_{A,b,c}(\eta).$$

□

Theorem 235.7 (B-516: $\mathsf{LPDuality}^{\uparrow}$ primal feasibility binary convex closure)**.** *Let F be the ordered-field scalar interface carried by a finite $\mathsf{LPDuality}^{\uparrow}$ row certificate as in Definition 235.1. Let J be the finite column index object, let I_0 be any finite family of certified primal rows, let*

$$A : I_0 \times J \to F, \qquad b : I_0 \to F$$

be the row-coefficient and right-hand-side data, and let

$$x, x' : J \to F$$

be two primal vectors. Write

$$\mathsf{Row}_i(z) := \sum_{j \in J} A_{ij} \cdot_F z_j$$

for the finite row evaluation, and define

$$\mathsf{PrFeas}^{I_0}_{A,b}(z) \quad :\Longleftrightarrow \quad \forall i \in I_0,\ \mathsf{Row}_i(z) \preceq_F b_i.$$

Assume

$$\mathsf{PrFeas}^{I_0}_{A,b}(x), \qquad \mathsf{PrFeas}^{I_0}_{A,b}(x'),$$

and let $\alpha, \beta : F$ satisfy

$$\mathsf{NonNeg}_F(\alpha), \qquad \mathsf{NonNeg}_F(\beta), \qquad \alpha +_F \beta \sim_F 1_F.$$

Define the componentwise binary affine mixture

$$\lambda_j := \alpha \cdot_F x_j +_F \beta \cdot_F x'_j, \qquad j \in J.$$

The finite row certificate records the row-affine-linearity comparison for this mixture:

$$\mathsf{Row}_i(\lambda) \sim_F \alpha \cdot_F \mathsf{Row}_i(x) +_F \beta \cdot_F \mathsf{Row}_i(x') \qquad (i \in I_0).$$

The scalar order portion of the same certificate records binary order-addition monotonicity, the two-point finite-sum monotonicity instance of Definition 235.1:

$$p \preceq_F q \wedge r \preceq_F s \Longrightarrow p +_F r \preceq_F q +_F s.$$

Then

$$\mathsf{PrFeas}^{I_0}_{A,b}(\lambda).$$

Moreover, the closure is row-local and coefficient-sign-free: no hypothesis of the form $\mathsf{NonNeg}_F(A_{ij})$ is required for any row coefficient.

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityPrimalFeasibilityBinaryConvexClosure`

Proof. Fix a row $i \in I_0$. By the definition of λ, field distributivity, commutativity of scalar multiplication, and the finite-sum linearity certificate over the finite column object J recorded by the row-affine-linearity field displayed in the theorem statement,

$$\begin{aligned}\mathsf{Row}_i(\lambda) &= \sum_{j\in J} A_{ij} \cdot_F (\alpha \cdot_F x_j +_F \beta \cdot_F x'_j)\\ &\sim_F \sum_{j\in J} (\alpha \cdot_F (A_{ij} \cdot_F x_j) +_F \beta \cdot_F (A_{ij} \cdot_F x'_j))\\ &\sim_F \alpha \cdot_F \sum_{j\in J} A_{ij} \cdot_F x_j +_F \beta \cdot_F \sum_{j\in J} A_{ij} \cdot_F x'_j\\ &= \alpha \cdot_F \mathsf{Row}_i(x) +_F \beta \cdot_F \mathsf{Row}_i(x').\end{aligned}$$

The two feasibility assumptions give

$$\mathsf{Row}_i(x) \preceq_F b_i, \qquad \mathsf{Row}_i(x') \preceq_F b_i.$$

Since α and β are nonnegative, the multiplication-monotonicity row of Definition 235.1 yields

$$\alpha \cdot_F \mathsf{Row}_i(x) \preceq_F \alpha \cdot_F b_i, \qquad \beta \cdot_F \mathsf{Row}_i(x') \preceq_F \beta \cdot_F b_i.$$

The binary order-addition monotonicity row displayed in the theorem statement gives

$$\alpha \cdot_F \mathsf{Row}_i(x) +_F \beta \cdot_F \mathsf{Row}_i(x') \preceq_F \alpha \cdot_F b_i +_F \beta \cdot_F b_i.$$

By right distributivity and the convex-weight certificate,

$$\alpha \cdot_F b_i +_F \beta \cdot_F b_i \sim_F (\alpha +_F \beta) \cdot_F b_i \sim_F 1_F \cdot_F b_i \sim_F b_i.$$

The classifier-stability row for $\preceq_F$ recorded in Definition 235.1, applied to the row-evaluation classifier comparison above and to the final scalar classifier comparison, yields

$$\mathsf{Row}_i(\lambda) \preceq_F b_i.$$

Because $i \in I_0$ was arbitrary, $\mathsf{PrFeas}^{I_0}_{A,b}(\lambda)$ follows. The proof used only finite-sum linearity, commutative-field distributivity, ordered-field monotonicity under nonnegative scalar multiplication, additive monotonicity of the order through the displayed two-point finite-sum row, and classifier stability for $\preceq_F$ from Definition 235.1. The entries A_{ij} were never compared with 0_F. Hence the closure is local in the chosen row ledger I_0 and does not require any coefficient-sign side condition. □

Theorem 235.8 (LPDuality feasible-region face exhaustion)**.** *Every displayed feasible-region face row read by an* $\mathsf{LPDuality}^{\uparrow}$ *consumer is generated by either the primal binary convex closure row of Definition 235.7 or the dual binary convex closure row of Definition 235.6, together with the ordered-field prefix order and the finite* BHist *ledger of Definition 235.1. The consumer surface has no coordinate for host convex geometry, ambient polyhedra, topological faces, separation theorems, or external linear-programming objects.*

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityFeasibleRegion face exhaustion`

Proof. Start with a displayed feasible-region face row in the $\mathsf{LPDuality}^{\uparrow}$ consumer surface. The finite ordered-field row of Definition 235.1 presents such a row only through the primal feasibility predicate, the dual feasibility predicate, the ordered-field prefix comparison, and the finite BHist ledger.

For a primal face row, apply Definition 235.7. It shows that binary convex motion inside any certified finite family of primal rows is generated by the carried field operations, order monotonicity, classifier stability, and the displayed finite-sum ledger.

For a dual face row, apply Definition 235.6. The same finite ordered-field rows generate the nonnegative dual coordinates and the dual constraint comparisons, again without leaving the displayed ledger.

The remaining comparison between primal and dual consumer rows is the weak-duality prefix-order theorem Definition 235.3. It relates the two generated surfaces only through the ordered-field prefix order. Since every cited row is drawn from the finite BHist ledger, no host convex geometry or external linear-programming object is available to generate an additional feasible-region face. □

Lemma 235.9 (LPDuality primal objective binary affine readback)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $x, x' : J \to F$ *be primal vectors, let* $\alpha, \beta : F$ *be scalar coefficients, and define* $\lambda : J \to F$ *componentwise by*

$$\lambda_j := \alpha \cdot_F x_j +_F \beta \cdot_F x'_j \qquad (j \in J).$$

Then the primal objective is affine over this binary mixture:

$$\mathsf{PrObj}_{A,c}(\lambda) \sim_F \alpha \cdot_F \mathsf{PrObj}_{A,c}(x) +_F \beta \cdot_F \mathsf{PrObj}_{A,c}(x').$$

Proof. Expand the objective endpoint from Definition 235.1. By the definition of λ, distributivity in the carried $\mathsf{Field}^{\uparrow}$ row, commutativity and associativity of multiplication, and finite-sum compatibility with the scalar classifier, one has

$$\begin{aligned}
\mathsf{PrObj}_{A,c}(\lambda) &= \sum_{j\in J} c_j \cdot_F (\alpha \cdot_F x_j +_F \beta \cdot_F x'_j) \\
&\sim_F \sum_{j\in J} \big(\alpha \cdot_F (c_j \cdot_F x_j) +_F \beta \cdot_F (c_j \cdot_F x'_j)\big) \\
&\sim_F \alpha \cdot_F \sum_{j\in J} c_j \cdot_F x_j +_F \beta \cdot_F \sum_{j\in J} c_j \cdot_F x'_j \\
&= \alpha \cdot_F \mathsf{PrObj}_{A,c}(x) +_F \beta \cdot_F \mathsf{PrObj}_{A,c}(x').
\end{aligned}$$

This is the asserted classifier equality. □

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityFiniteOrderedFieldFeasibilityRow primal objective binary affine readback`

Theorem 235.10 (LPDuality Farkas finite-alternative handoff)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Suppose the same finite coefficient rows* A, b, c *are equipped with a* $\mathsf{FarkasLemma}^{\uparrow}$ *packet from chapter 6185. Then the Farkas finite-alternative row supplies the nonnegative multiplier ledger consumed by the weak-duality comparison of Definition 235.3: feasibility branches become primal or dual feasible rows, and obstruction branches become the displayed multiplier comparison rows whose finite sums witness objective separation in the ordered-field ledger.*

Proof. Project the $\mathsf{LPDuality}^{\uparrow}$ row through Definition 235.1. The weak-duality theorem consumes exactly finite row and column carriers, ordered-field scalar operations, nonnegative scalar predicates, finite-sum monotonicity, and the distributivity-interchange ledger.

Project the $\mathsf{FarkasLemma}^{\uparrow}$ packet through Definition 6185.1. Its feasibility branch is already a finite row of inequalities over the same coefficient system, so classifier transport repacks it as the corresponding primal or dual feasibility row in the linear-programming surface.

In the obstruction branch, the Farkas packet supplies a nonnegative multiplier row and the coefficient-annihilation equations. The $\mathsf{PositiveCone}^{\uparrow}$ nonnegative witnesses give the scalar monotonicity hypotheses used in weak duality, while the finite-sum equations identify the multiplier sum with the objective endpoint comparison read by the $\mathsf{LPDuality}^{\uparrow}$ ledger.

Thus the handoff is row-for-row. The Farkas certificate contributes no optimizer, separation theorem, or strong-duality existence claim; it only names the finite multiplier ledger that the weak-duality comparison already consumes. □

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityFarkasAlternativeHandoff`

Theorem 235.11 (LPDuality optimal primal face binary convex closure)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $x, x' : J \to F$ *be primal feasible witnesses, let* $\tau : F$ *be a scalar endpoint, and suppose both witnesses lie in the same primal objective fiber:*

$$\mathsf{PrFeas}_{A,b}(x), \qquad \mathsf{PrFeas}_{A,b}(x'),$$

$$\mathsf{PrObj}_{A,c}(x) \sim_F \tau, \qquad \mathsf{PrObj}_{A,c}(x') \sim_F \tau.$$

Let $\alpha, \beta : F$ *satisfy*

$$\mathsf{NonNeg}_F(\alpha), \qquad \mathsf{NonNeg}_F(\beta), \qquad \alpha +_F \beta \sim_F 1_F,$$

and define the componentwise binary affine mixture

$$\lambda_j := \alpha \cdot_F x_j +_F \beta \cdot_F x'_j \qquad (j \in J).$$

Then λ *is primal feasible and remains in the same primal objective fiber:*

$$\mathsf{PrFeas}_{A,b}(\lambda), \qquad \mathsf{PrObj}_{A,c}(\lambda) \sim_F \tau.$$

Consequently, when τ *is the scalar endpoint certified as the primal optimum of the row, the certified primal optimal face at* τ *is closed under binary affine combinations with nonnegative coefficients whose sum is classifier-equal to* 1_F*.*

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityOptimalPrimalFace binary convex closure`

Proof. Unpack the two primal-feasibility hypotheses using Definition 235.1. They give

$$\forall j \in J.\ \mathsf{NonNeg}_F(x_j), \qquad \forall j \in J.\ \mathsf{NonNeg}_F(x'_j),$$

and

$$\forall i \in I.\ \alpha_i(x) \preceq_F b_i, \qquad \forall i \in I.\ \alpha_i(x') \preceq_F b_i.$$

First fix $j \in J$. Since $\mathsf{NonNeg}_F(x_j)$ and $\mathsf{NonNeg}_F(x'_j)$ mean $0_F \preceq_F x_j$ and $0_F \preceq_F x'_j$, multiplication monotonicity by the nonnegative scalars α and β gives

$$\alpha \cdot_F 0_F \preceq_F \alpha \cdot_F x_j, \qquad \beta \cdot_F 0_F \preceq_F \beta \cdot_F x'_j.$$

The two-point finite-sum monotonicity instance in Definition 235.1 yields

$$\alpha \cdot_F 0_F +_F \beta \cdot_F 0_F \preceq_F \alpha \cdot_F x_j +_F \beta \cdot_F x'_j.$$

The $\mathsf{Field}^{\uparrow}$ zero laws identify the left endpoint with 0_F, and the definition of λ_j identifies the right endpoint with λ_j. Classifier stability of $\preceq_F$ therefore gives

$$0_F \preceq_F \lambda_j.$$

Thus $\mathsf{NonNeg}_F(\lambda_j)$ holds for every $j \in J$.

For the row inequalities, apply Definition 235.7 to the finite row family I, with $\mathsf{Row}_i(z) = \alpha_i(z)$. The displayed row-constraint clauses for x and x', together with the same nonnegative coefficients and the classifier equality $\alpha +_F \beta \sim_F 1_F$, give

$$\forall i \in I.\ \alpha_i(\lambda) \preceq_F b_i.$$

Combining this row-constraint clause with the nonnegativity clause just proved is exactly

$$\mathsf{PrFeas}_{A,b}(\lambda)$$

by Definition 235.1.

It remains to read back the objective endpoint. By Definition 235.9,

$$\mathsf{PrObj}_{A,c}(\lambda) \sim_F \alpha \cdot_F \mathsf{PrObj}_{A,c}(x) +_F \beta \cdot_F \mathsf{PrObj}_{A,c}(x').$$

Scalar classifier congruence transports the two objective-fiber hypotheses through multiplication and addition, giving

$$\alpha \cdot_F \mathsf{PrObj}_{A,c}(x) +_F \beta \cdot_F \mathsf{PrObj}_{A,c}(x') \sim_F \alpha \cdot_F \tau +_F \beta \cdot_F \tau.$$

Distributivity, the coefficient-sum classifier equality, and the multiplicative unit law give

$$\alpha \cdot_F \tau +_F \beta \cdot_F \tau \sim_F (\alpha +_F \beta) \cdot_F \tau \sim_F 1_F \cdot_F \tau \sim_F \tau.$$

Transitivity of the scalar classifier composes the displayed classifier equalities into

$$\mathsf{PrObj}_{A,c}(\lambda) \sim_F \tau.$$

The final sentence is the specialization of the same statement to a scalar endpoint already certified as optimal for the row. □

Lemma 235.12 (LPDuality dual objective binary affine readback)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $y, y' : I \to F$ *be dual vectors, let* $\alpha, \beta : F$ *be scalar coefficients, and define* $\mu : I \to F$ *componentwise by*

$$\mu_i := \alpha \cdot_F y_i +_F \beta \cdot_F y'_i \qquad (i \in I).$$

Then the dual objective is affine over this binary mixture:

$$\mathsf{DuObj}_b(\mu) \sim_F \alpha \cdot_F \mathsf{DuObj}_b(y) +_F \beta \cdot_F \mathsf{DuObj}_b(y').$$

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityDualObjective binary affine readback`

Proof. Expand the dual objective endpoint from Definition 235.1. By the definition of μ, distributivity in the carried $\mathsf{Field}^{\uparrow}$ row, associativity and commutativity of multiplication, and finite-sum compatibility with the scalar classifier, one has

$$\begin{aligned}\mathsf{DuObj}_b(\mu) &= \sum_{i\in I}(\alpha \cdot_F y_i +_F \beta \cdot_F y'_i) \cdot_F b_i \\ &\sim_F \sum_{i\in I}\big(\alpha \cdot_F (y_i \cdot_F b_i) +_F \beta \cdot_F (y'_i \cdot_F b_i)\big) \\ &\sim_F \alpha \cdot_F \sum_{i\in I} y_i \cdot_F b_i +_F \beta \cdot_F \sum_{i\in I} y'_i \cdot_F b_i \\ &= \alpha \cdot_F \mathsf{DuObj}_b(y) +_F \beta \cdot_F \mathsf{DuObj}_b(y').\end{aligned}$$

This is the asserted classifier equality. □

Theorem 235.13 (LPDuality optimal dual face binary convex closure)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $y, y' : I \to F$ *be dual feasible witnesses, let* $\tau : F$ *be a scalar endpoint, and suppose both witnesses lie in the same dual objective fiber:*

$$\mathsf{DuFeas}_{A,b,c}(y), \qquad \mathsf{DuFeas}_{A,b,c}(y'),$$

$$\mathsf{DuObj}_b(y) \sim_F \tau, \qquad \mathsf{DuObj}_b(y') \sim_F \tau.$$

Let $\alpha, \beta : F$ *satisfy*

$$\mathsf{NonNeg}_F(\alpha), \qquad \mathsf{NonNeg}_F(\beta), \qquad \alpha +_F \beta \sim_F 1_F,$$

and define the componentwise binary affine mixture

$$\mu_i := \alpha \cdot_F y_i +_F \beta \cdot_F y'_i \qquad (i \in I).$$

Then μ *is dual feasible and remains in the same dual objective fiber:*

$$\mathsf{DuFeas}_{A,b,c}(\mu), \qquad \mathsf{DuObj}_b(\mu) \sim_F \tau.$$

Consequently, when τ *is the scalar endpoint certified as the dual optimum of the row, the certified dual optimal face at* τ *is closed under binary affine combinations with nonnegative coefficients whose sum is classifier-equal to* 1_F*.*

Proof. The feasibility statement is exactly the dual-feasibility binary convex closure of Definition 235.6, applied to the two feasible witnesses y, y' and to the coefficients α, β. Hence

$$\mathsf{DuFeas}_{A,b,c}(\mu).$$

It remains to read back the objective endpoint. By Definition 235.12,

$$\mathsf{DuObj}_b(\mu) \sim_F \alpha \cdot_F \mathsf{DuObj}_b(y) +_F \beta \cdot_F \mathsf{DuObj}_b(y').$$

Scalar classifier congruence transports the two objective-fiber hypotheses through multiplication and addition, giving

$$\alpha \cdot_F \mathsf{DuObj}_b(y) +_F \beta \cdot_F \mathsf{DuObj}_b(y') \sim_F \alpha \cdot_F \tau +_F \beta \cdot_F \tau.$$

Distributivity, the coefficient-sum classifier equality, and the multiplicative unit law give

$$\alpha \cdot_F \tau +_F \beta \cdot_F \tau \sim_F (\alpha +_F \beta) \cdot_F \tau \sim_F 1_F \cdot_F \tau \sim_F \tau.$$

Transitivity of the scalar classifier composes the displayed classifier equalities into

$$\mathsf{DuObj}_b(\mu) \sim_F \tau.$$

The final sentence is the specialization of the same statement to a scalar endpoint already certified as optimal for the dual row. □

Theorem 235.14 (LPDuality objective fiber sandwich closure)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $x, x' : J \to F$ *be primal feasible witnesses, let* $y, y' : I \to F$ *be dual feasible witnesses, and let* $\tau : F$ *be a scalar endpoint. Suppose weak duality gives the objective comparison for every displayed primal-dual pair, and suppose all four displayed objective endpoints lie in the same scalar fiber:*

$$\mathsf{PrObj}_{A,c}(x) \sim_F \tau, \qquad \mathsf{PrObj}_{A,c}(x') \sim_F \tau,$$

$$\mathsf{DuObj}_b(y) \sim_F \tau, \qquad \mathsf{DuObj}_b(y') \sim_F \tau.$$

Then the primal optimal-face closure row at τ *and the dual optimal-face closure row at* τ *are consumed by the same* τ*-fiber sandwich: every binary affine primal mixture of* x, x' *and every binary affine dual mixture of* y, y' *with nonnegative coefficients whose sum is classifier-equal to* 1_F *remains feasible, remains classifier-equal to* τ *on its own objective side, and sits between weak-duality comparisons whose endpoints are classifier-stable copies of* τ*.*

Lean4 ✓ `BEDC.Derived.LPDualityUp.LPDualityObjectiveFiber sandwich closure`

Proof. First apply Definition 235.3 to each displayed primal-dual pair. Classifier stability of $\preceq_F$ transports the primal and dual objective endpoints along the four fiber hypotheses, so each weak duality comparison becomes a comparison from a scalar endpoint classifier-equal to τ to a scalar endpoint classifier-equal to τ.

Next apply Definition 235.11 to the primal witnesses x, x' at the fiber endpoint τ. This gives primal feasibility and primal objective readback at τ for every allowed binary affine primal mixture.

Apply Definition 235.13 to the dual witnesses y, y' at the same fiber endpoint. This gives dual feasibility and dual objective readback at τ for every allowed binary affine dual mixture.

Finally use weak duality once more on the transported primal and dual mixtures. The two closure rows provide the objective endpoint equalities to τ, and classifier stability of the ordered-field order transports the comparison endpoints into the common τ fiber. Thus the primal and dual closure rows are consumed by one objective-fiber sandwich rather than by separate scalar certificates. □

Theorem 235.15 (LPDuality primal-dual affine fiber product closure)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let* $x, x' : J \to F$ *be primal feasible witnesses, let* $y, y' : I \to F$ *be dual feasible witnesses, and let* $\tau : F$ *be a scalar endpoint. Suppose all four objective endpoints lie in the same scalar fiber:*

$$\mathsf{PrObj}_{A,c}(x) \sim_F \tau, \qquad \mathsf{PrObj}_{A,c}(x') \sim_F \tau,$$

$$\mathsf{DuObj}_b(y) \sim_F \tau, \qquad \mathsf{DuObj}_b(y') \sim_F \tau.$$

Let $\alpha, \beta : F$ be nonnegative coefficients with $\alpha +_F \beta \sim_F 1_F$, and form the paired binary affine mixture

$$\lambda_j := \alpha \cdot_F x_j +_F \beta \cdot_F x'_j \qquad (j \in J), \qquad \mu_i := \alpha \cdot_F y_i +_F \beta \cdot_F y'_i \qquad (i \in I).$$

Then (λ, μ) is a feasible primal-dual pair, both objective endpoints remain classifier-equal to τ, and the weak-duality comparison between the mixed pair is a comparison between classifier-stable copies of τ.

Proof. Apply Definition 235.11 to the primal witnesses x, x' with endpoint τ and coefficients α, β. This gives the primal feasibility of λ and the readback $\mathsf{PrObj}_{A,c}(\lambda) \sim_F \tau$.

Apply Definition 235.13 to the dual witnesses y, y' with the same endpoint and coefficients. This gives the dual feasibility of μ and the readback $\mathsf{DuObj}_b(\mu) \sim_F \tau$.

Finally use Definition 235.14 on the same four witnesses and the same scalar fiber. The two closure rows give the paired feasible mixture, and weak duality for the mixed pair is transported along the two objective readbacks to a comparison whose endpoints are classifier-stable copies of τ. □

Theorem 235.16 (LPDuality weak-duality slackness optimality packet)**.** *Let an* $\mathsf{LPDuality}^{\uparrow}$ *row over an ordered field be given as in Definition 235.1. Let $x : J \to F$ be a primal feasible witness and let $y : I \to F$ be a dual feasible witness. Suppose the two complementary-slackness product families are classifier-equal termwise:*

$$\forall j \in J.\ c_j \cdot_F x_j \sim_F \beta_j(y) \cdot_F x_j, \qquad \forall i \in I.\ y_i \cdot_F \alpha_i(x) \sim_F y_i \cdot_F b_i.$$

Then the row carries a certified optimality packet: the objectives $\mathsf{PrObj}_{A,c}(x)$ and $\mathsf{DuObj}_b(y)$ are classifier-equal, x is primal optimal, and y is dual optimal. Moreover, writing $\tau := \mathsf{PrObj}_{A,c}(x)$, any further primal feasible optimal witness $x' : J \to F$ and any further dual feasible optimal witness $y' : I \to F$ whose objective endpoints are classifier-equal to τ enter the common objective-fiber sandwich at τ: every binary affine primal mixture of x, x' and every binary affine dual mixture of y, y' with nonnegative coefficients whose sum is classifier-equal to 1_F remains feasible, remains in its own τ-objective fiber, and is bounded by the weak-duality comparisons between classifier-stable copies of τ.

Proof. Apply Definition 235.5 to the feasible witnesses x and y and to the two displayed slackness families. This gives the scalar objective equality together with the primal optimality of x and the dual optimality of y.

The objective equality identifies $\mathsf{DuObj}_b(y)$ with the same fiber endpoint $\tau := \mathsf{PrObj}_{A,c}(x)$. Together with the assumed fiber readbacks for the feasible optimal witnesses x' and y', the four endpoints x, x', y, y' have weak-duality comparisons by Definition 235.3. Hence they satisfy the hypotheses of Definition 235.14. Applying that theorem gives the shared primal and dual binary-affine closure statement and the weak-duality sandwich over the common τ fiber. □

Closure status for $\mathsf{NameCert}_{\mathsf{LPDuality}^{\uparrow}}$.

Constructive story: A linear-programming duality row begins with finite row and column carriers, a $\mathsf{Field}^{\uparrow}$ scalar surface equipped with a $\mathsf{TotalOrder}^{\uparrow}$ comparison, and displayed coefficient, right-hand-side, and cost rows from which primal feasibility, dual feasibility, and objective endpoints are computed by finite sums. The carrier accepts only packets whose feasible-region rows come through $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$, whose scalar operations and comparisons come through the ordered-field surface, and whose weak-duality, complementary-slackness, face, and objective-fiber data are kept as finite BHist rows. Two accepted packets classify together by componentwise hsame transport of scalar endpoints, feasible witnesses, objective values, finite-sum ledgers, face witnesses, and consumer rows, never by ambient matrix

equality or quotient vector-space equality. The source specification is the finite ordered-field feasibility row plus the convex and polytope dependency rows, the pattern specification records weak-duality comparison and complementary-slackness equality, and the ledger keeps nonnegativity, finite-sum monotonicity, objective readbacks, convex closure, and face-fiber rows visible. The five name-certificate obligations are supplied by an admitted feasible packet, the inherited equivalence and order-stability rows, and the finite transport proofs that rebuild each primal, dual, objective, and ledger row after classified replacement. A packet history h lies in the signature gap of an LPDuality$^{\uparrow}$ token p when the source-domain accepts its ordered-field primal-dual surface, the signature relation pairs it with an answer duality packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the feasibility rows, weak-duality comparison, and any complementary-slackness equality are exactly those encoded by p. The chapter registers this finite ordered-field duality construction under the name LPDuality$^{\uparrow}$; it introduces no optimization solver, no strong-duality existence theorem, no separation theorem, and no standard bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{LPDuality}\uparrow}$, maturePackageClosed)

Scope closed: The mature boundary binds the finite ordered-field feasibility row, weak-duality comparison, complementary-slackness equality, primal and dual convex closure rows, feasible-region face exhaustion, primal objective binary affine readback, PolytopeUp face and witness rows, FieldUp scalar operations, TotalOrderUp comparison rows, classifier transport, finite-sum ledgers, and consumer rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{LPDuality}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.LPDualityUp.LPDualityFeasibleWeakDuality\_prefix\_order`

Bridge status: none

Not claimed: This chapter does not close full strong duality, existence of optima, separation theorems, or a standard bridge.

Upgrade path: No further theory-axis upgrade remains; bridgeChecked remains a formal-axis standard-bridge task.

Chapter 236

A Concrete Naming Certificate for $\mathsf{ShortestPath}^{\uparrow}$

The shortest-path interface $\mathsf{ShortestPath}^{\uparrow}$ packages, on a weighted graph carrier with nonnegative weights, the path of minimum total weight between two vertices together with Dijkstra and Bellman-Ford correctness certificates. It depends on $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118) and $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38).

236.1 Carrier definition

Definition 236.1 (Shortest-path weighted graph carrier)**.** *A* $\mathsf{ShortestPath}^{\uparrow}$ *carrier is a finite* BHist *weighted-graph packet. Its visible rows consist of vertex rows, directed or undirected edge rows, nonnegative* $\mathsf{Real}^{\uparrow}$ *weight rows attached to the displayed edges, source and target endpoint rows, a finite path ledger, and the* Pkg *provenance row that records which graph and weight packet the ledger reads from.*

The carrier does not include a completed metric space, an infinite graph, an unstated host ordering on vertices, or an external minimization oracle. Every comparison row is read from the finite graph packet, the $\mathsf{Real}^{\uparrow}$ *weight certificates, the endpoint rows, and the displayed path ledger.*

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier endpoint expanded boundary`
Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier source rows`

236.2 The certificate

Theorem 236.2 (Shortest-path visible path ledger)**.** *For any accepted* $\mathsf{ShortestPath}^{\uparrow}$ *packet, the readable path data are exactly the finite vertex sequence in the path ledger, the edge-incidence rows certifying that consecutive vertices are connected, the nonnegative* $\mathsf{Real}^{\uparrow}$ *weight rows for those edges, the source and target endpoint rows, and the* Pkg *provenance row tying the ledger to the weighted-graph carrier.*

No row of the certificate asserts an unseen edge, a hidden lower-weight path, a completed real metric, or a host-index equality between vertices.

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier visible path ledger`
↪ variant `BEDC.Derived.ShortestPathUp.ShortestPathVisiblePathLedger`

Proof. Unpack the weighted-graph carrier of Definition 236.1. The only graph rows it exports are the finite vertex rows, edge rows, endpoint rows, weight rows, path-ledger rows, and package provenance.

Read the path ledger pointwise. Each adjacent pair in the displayed vertex sequence is checked by one visible edge-incidence row, and each such edge contributes its displayed nonnegative $\mathsf{Real}^{\uparrow}$ weight row.

The source and target constraints are the endpoint rows at the first and last ledger positions. Since the carrier has no generator for unseen edges, completed metrics, host-index equalities, or external minimization oracles, the listed rows are exactly the visible path ledger. □

Theorem 236.3 (Shortest-path relaxation certificate scope)**.** *Dijkstra and Bellman-Ford readback for* $\mathsf{ShortestPath}^{\uparrow}$ *remains a finite* BHist *ledger obligation. The certificate surface consists of the weighted-graph carrier, the source and target endpoint rows, nonnegative* $\mathsf{Real}^{\uparrow}$ *edge-weight rows, finite relaxation rows comparing displayed path weights, predecessor or path-ledger rows, and the* Pkg *provenance row.*

The certificate does not assert an ambient choice of shortest path, a total ordering of all host vertices, an infinite relaxation process, or a completed real-analysis object.

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier relaxation certificate scope`

↪ variant `BEDC.Derived.ShortestPathUp.ShortestPathRelaxationCertificate scope`

Proof. Start from the carrier definition. It fixes a finite graph packet, a finite family of nonnegative $\mathsf{Real}^{\uparrow}$ weight rows, and the package provenance used by the path ledger.

For a Dijkstra readback, each settled-step claim is a row comparing visible candidate path weights over the finite frontier named by the packet. For a Bellman-Ford readback, each relaxation claim is a finite edge row together with the displayed before-and-after path-weight comparison.

In both readings the predecessor or path-ledger row supplies the reconstructed path, and Definition 236.2 identifies the readable endpoint and incidence rows. Thus the certificate surface is a finite BHist ledger obligation over the named graph and weight rows, with no ambient choice principle, host-vertex ordering, infinite process, or completed real object. □

Definition 236.4 (Shortest-path relaxation ledger carrier)**.** *A* $\mathsf{ShortestPath}^{\uparrow}$ *relaxation ledger carrier is a finite* BHist *packet over the weighted-graph carrier of Definition 236.1. Its displayed rows are the* $\mathsf{Graph}^{\uparrow}$ *vertex and edge rows, nonnegative* $\mathsf{Real}^{\uparrow}$ *edge-weight rows, before-and-after path-weight rows, predecessor rows, visible* Cont *relaxation rows, the* hsame *comparisons recording the transported packet equalities, and the* Pkg *provenance tying the relaxation ledger to the named graph and weight packet.*

The carrier records only finite relaxation evidence. It does not contain a host minimizer, a hidden vertex order, an infinite relaxation stream, or a completed metric assumption.

Lean4 (definition) `BEDC.Derived.ShortestPathUp.ShortestPathRelaxationLedgerCarrier`

Theorem 236.5 (Shortest-path relaxation continuation closure)**.** *Let a* $\mathsf{ShortestPath}^{\uparrow}$ *weighted-graph carrier display finite vertex rows, edge rows, nonnegative* $\mathsf{Real}^{\uparrow}$ *weight rows, endpoint rows, and path-ledger rows. If a transported relaxation packet also displays the before-and-after path-weight rows, predecessor rows, visible* Cont *relaxation rows, matching* hsame *comparisons, and the same* Pkg *provenance, then those rows repack as an accepted relaxation ledger carrier over the same weighted graph.*

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier relaxation cont closure`

Proof. Project the weighted-graph carrier rows. They give the finite vertex, edge, weight, endpoint, path-ledger, and provenance rows named in Definition 236.1.

Next read the transported relaxation packet. Its visible Cont rows produce the before-and-after path-weight rows and the predecessor rows, while the hsame comparisons identify the transported packets with the displayed source graph and weight rows.

The required carrier is obtained by repacking exactly these projected and transported rows into Definition 236.4. The Pkg provenance is unchanged, so the relaxation ledger remains attached to the original weighted-graph packet. □

Theorem 236.6 (Shortest-path visible relaxation NameCert surface)**.** *The visible path ledger of Definition 236.2 together with an accepted relaxation ledger carrier of Definition 236.4 supplies the finite* $\mathsf{NameCert}_{\mathsf{ShortestPath}^{\uparrow}}$ *surface for displayed Dijkstra or Bellman-Ford readback. The surface consists of endpoint visibility, edge-incidence visibility, nonnegative weight visibility, finite before-and-after path-weight comparisons, predecessor visibility, relaxation* Cont *rows,* hsame *transport comparisons, and* Pkg *provenance.*

It does not expose a host minimization principle, a hidden vertex order, an infinite relaxation process, or a completed metric-space assumption.

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier visible relaxation namecert surface`

Proof. Use Definition 236.2 to obtain the endpoint, incidence, nonnegative weight, path-ledger, and provenance rows from the visible path certificate.

Apply Definition 236.5 to the transported relaxation rows. This supplies the finite before-and-after path-weight rows, predecessor rows, visible Cont relaxation rows, and hsame transport comparisons over the same weighted graph.

Combining the two finite ledgers gives exactly the displayed $\mathsf{NameCert}_{\mathsf{ShortestPath}^{\uparrow}}$ surface. Since neither carrier has a generator for host minimizers, hidden vertex orderings, infinite relaxation streams, or completed metric objects, none of those assumptions is part of the certificate surface. □

Theorem 236.7 (Shortest-path finite edge exhaustion)**.** *Every accepted* $\mathsf{ShortestPath}^{\uparrow}$ *path-ledger step is exhausted by visible finite data: a displayed edge-incidence row, the source and target endpoint rows it connects, a finite* Cont *route through the carrier rows, and the* Pkg *provenance row that ties the step to the weighted-graph packet.*

Consequently the certificate has no hidden edge source. Any edge used by the accepted path is read from the finite BHist *carrier of Definition 236.1, not from a host graph outside the certificate.*

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier finite edge exhaustion`

Proof. Inspect the carrier constructors in Definition 236.1. The carrier exports only vertex rows, edge rows, nonnegative weight rows, endpoint rows, path-ledger rows, and the package provenance row.

Apply Definition 236.2 to a single adjacent ledger step. The step supplies its two displayed vertices, the incidence row between them, and the endpoint rows used by the ledger.

It remains to account for routing and provenance. The finite Cont route is the row-by-row traversal through those displayed carrier rows, and the Pkg row records that the traversal reads this weighted-graph packet. Since the carrier has no constructor for an unseen edge source, the listed rows exhaust the edge obligation. □

Theorem 236.8 (Shortest-path weight readback)**.** *For any accepted* $\mathsf{ShortestPath}^{\uparrow}$ *packet, the total weight of the displayed path is read only from the visible nonnegative* $\mathsf{Real}^{\uparrow}$ *edge-weight rows and the finite path-ledger summation rows.*

No total path weight may be supplied by an ambient metric, an unlisted edge weight, a host real-number accumulator, or an implicit minimization oracle.

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier weight readback`

Proof. Start with the visible ledger theorem. By Definition 236.2, every adjacent pair in the accepted vertex sequence has one displayed edge-incidence row and one displayed nonnegative $\mathsf{Real}^{\uparrow}$ weight row.

The summation rows are finite because the path ledger is finite in the weighted-graph carrier. Reading them in ledger order gives the total weight as the certificate's displayed finite sum of those edge-weight rows.

The carrier described in Definition 236.1 contains no completed metric object, no unlisted edge-weight generator, and no host accumulator. Thus the readback of total weight is exactly the finite $\mathsf{Real}^{\uparrow}$ row summation named by the certificate. □

Theorem 236.9 (Shortest-path relaxation soundness)**.** *A Dijkstra or Bellman-Ford relaxation row in an accepted* $\mathsf{ShortestPath}^{\uparrow}$ *packet can lower or preserve a displayed path-weight candidate only through visible rows: the relaxed edge row, its predecessor row, its endpoint rows, the relevant* hsame *identifications, the finite* Cont *route through the ledger, and the* Pkg *provenance row for the weighted-graph carrier.*

The relaxation certificate therefore cannot justify a candidate change by a hidden predecessor, a host-side vertex equality, an unrecorded edge weight, or an infinite relaxation process.

Lean4 ✓ `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier relaxation soundness`

Proof. Use Definition 236.3. It identifies a relaxation claim as a finite row comparing displayed before-and-after path-weight candidates over the named graph packet.

The row can mention an edge only through the carrier of Definition 236.1. Its predecessor and endpoint data are therefore the displayed predecessor row, endpoint rows, and edge-incidence row already present in the finite BHist packet.

Apply Definition 236.2 to reconstruct the candidate path segment. The required hsame identifications and finite Cont route are the visible transports that keep the predecessor, edge, and endpoint rows in the same ledger packet, while the Pkg row supplies provenance.

No remaining constructor can lower or preserve the displayed candidate: the certificate has no host-side equality source, no hidden predecessor row, no unrecorded weight row, and no infinite relaxation process. The relaxation is therefore sound exactly over the visible finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ShortestPath}^{\uparrow}}$**.**

Constructive story: A shortest-path row is generated from a finite weighted graph packet by keeping the vertex rows, edge rows, nonnegative $\mathsf{Real}^{\uparrow}$ weight rows, endpoint rows, path ledger, relaxation rows, hsame transports, Cont routes, and Pkg provenance visible in the same BHist carrier. The obligation surface records only the finite ledger data available to Dijkstra or Bellman-Ford readback; it introduces no ambient minimizer, hidden vertex order, infinite relaxation stream, or completed metric-space object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ShortestPath}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped surface binds the weighted-graph carrier rows, endpoint rows, nonnegative $\mathsf{Real}^{\uparrow}$ weight rows, finite path-ledger rows, relaxation Cont rows, hsame transports, and Pkg provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ShortestPath}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ShortestPathUp.ShortestPathWeightedGraphCarrier\_visible\_path\_ledger`

Bridge status: none

Not claimed: The obligation surface does not close optimality over infinite graphs or negative-cycle completeness.
Upgrade path: The paper-axis treatment is mature; the remaining top formal upgrade is bridgeChecked verification.

Chapter 237

A Concrete Naming Certificate for $\mathsf{SpanningTree}^{\uparrow}$

The spanning-tree interface $\mathsf{SpanningTree}^{\uparrow}$ packages a maximal acyclic subgraph reaching every vertex of a connected graph carrier, with minimum-weight selection in the weighted case (Kruskal / Prim). It depends on $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ (chapter 118) and $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$ (chapter 119).

237.1 Carrier definition

Theorem 237.1 (SpanningTreeUp carrier obligation surface)**.** *The carrier of* $\mathsf{SpanningTree}^{\uparrow}$ *exposes only finite* BHist *rows over the two source certificates* $\mathsf{NameCert}_{\mathsf{Graph}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Tree}^{\uparrow}}$*. An accepted packet carries displayed vertex rows, graph-edge rows, tree-edge rows, a root row or root-free witness inherited from* $\mathsf{Tree}^{\uparrow}$*, incidence rows tying each tree edge to graph endpoints, reachability rows from the root branch when present, acyclicity rows, and* Pkg *provenance for the two dependency certificates.*

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrierPacket dependency surface`

Proof. Read the carrier through the named $\mathsf{Graph}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ certificates. The graph certificate supplies the visible vertex and edge rows, while the tree certificate supplies the acyclicity, root-branch, and tree-edge rows.

The spanning-tree carrier adds no ambient graph object. It records the incidence and reachability witnesses as finite BHist rows with their Pkg provenance, so every consumer-visible field is already present in the displayed carrier packet. □

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrier obligation surface`

Theorem 237.2 (SpanningTreeUp classifier obligation surface)**.** *The classifier for* $\mathsf{SpanningTree}^{\uparrow}$ *is componentwise* hsame *on the visible vertex, graph-edge, tree-edge, root, incidence, reachability, acyclicity, and provenance rows named in Definition 237.1. Transport along the classifier may use the* $\mathsf{Graph}^{\uparrow}$ *and* $\mathsf{Tree}^{\uparrow}$ *source classifiers, but it does not compare hidden host graphs, hidden edge sets, or unrecorded root choices.*

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeBHistCarrier classifier obligation`
Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrierPacket classifier obligation`

Proof. Project the two source classifiers from $\mathsf{NameCert}_{\mathsf{Graph}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Tree}^\uparrow}$. These transport the graph-edge rows and tree rows by the same hsame witnesses carried in the spanning-tree packet.

The remaining rows are finite ledger coordinates. Transporting incidence, reachability, acyclicity, root, and provenance rows componentwise preserves the carrier because each transported row still points to the same displayed dependency certificates. □

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeClassifier obligation surface`

Theorem 237.3 (SpanningTreeUp acyclic maximality obligation surface)**.** *Maximal acyclic reachability for* $\mathsf{SpanningTree}^\uparrow$ *is read only from the displayed tree-edge ledger, the finite incidence rows, the root or root-free branch inherited from* $\mathsf{Tree}^\uparrow$*, and finite* Cont *routes witnessing reachability. The obligation excludes any maximality proof that inspects a hidden graph equality, an unlisted edge, or an infinite search over ambient edges.*

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeAcyclicMaximality surface`
Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeAcyclicMaximality obligation surface`

Proof. Use the tree-edge ledger as the only candidate subgraph. Acyclicity is the visible row imported from the $\mathsf{Tree}^\uparrow$ certificate, and reachability is checked by finite Cont routes whose endpoints are the displayed vertex rows.

Maximality is therefore a ledger-readback statement. If an edge participates in the claim, it must already occur as a graph-edge row with incidence data and as a candidate tree-edge comparison row; otherwise the carrier has no constructor through which the edge can be read. □

Theorem 237.4 (SpanningTreeUp weight readback obligation surface)**.** *In the weighted case,* $\mathsf{SpanningTree}^\uparrow$ *compares spanning-tree candidates only through displayed* $\mathsf{Real}^\uparrow$ *edge-weight rows attached to the graph-edge ledger and through finite summation ledgers attached to the chosen tree-edge rows. No comparison may use an external minimizer, a hidden weight function, or an infinite family of candidate trees.*

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrierPacket weight readback obligation`

Proof. Read each weight from the visible graph-edge row. The $\mathsf{Real}^\uparrow$ source supplies the carrier for the displayed edge-weight value, and the spanning-tree packet records which tree-edge row consumes that value.

The total weight of a candidate is a finite summation ledger over those displayed rows. Comparing two candidates is therefore a comparison of two finite ledgers and their $\mathsf{Real}^\uparrow$ readbacks, with no extra optimization object outside the packet. □

Theorem 237.5 (SpanningTreeUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SpanningTree}^\uparrow}$ *surface is assembled from the carrier rows of Definition 237.1, the classifier transport of Definition 237.2, the acyclic-reachability ledger of Definition 237.3, and the weighted readback rows of Definition 237.4. It explicitly excludes host graph equality, hidden edges, external minimization, and infinite graph data.*

Lean4 ✓ `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrier semantic name certificate`

Proof. Collect the four preceding obligation surfaces into one certificate packet. The carrier and classifier determine which histories are admissible and how they transport under hsame.

The maximality and weight clauses determine which ledger rows can be read by downstream algorithms such as Kruskal and Prim. Since each clause names only visible BHist, Cont, hsame, Pkg, $\mathsf{NameCert}_{\mathsf{Graph}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Tree}^\uparrow}$, and $\mathsf{Real}^\uparrow$ rows, the assembled certificate has no entry point for the excluded host-level data. □

237.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{SpanningTree}^{\uparrow}}$.

Constructive story: A spanning-tree packet is read from the displayed graph and tree certificates by projecting vertex rows, graph-edge rows, tree-edge rows, incidence witnesses, root or root-free data, finite reachability routes, acyclicity evidence, and provenance. The classifier transports only those visible rows through hsame and the source certificate classifiers. In the weighted case, edge weights are read through displayed $\mathsf{Real}^{\uparrow}$ rows and finite summation ledgers over the chosen tree edges. The certificate therefore seeds the spanning-tree surface as a finite ledger package over $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no hidden host graph, unlisted edge family, external minimizer, or infinite graph object.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpanningTree}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The displayed $\mathsf{Graph}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, finite BHist edge and vertex ledgers, Cont reachability routes, hsame classifier transport, and Pkg provenance form the seeded paper scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpanningTree}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SpanningTreeUp.SpanningTreeCarrier\_semantic\_name\_certificate`

Bridge status: none

Not claimed: Hidden graph equality, hidden edge sets, external minimizers, and infinite graph data are not claimed.

Upgrade path: Bridged closure requires a finite graph/tree bridge over edge-weight, summation-ledger, and $\mathsf{Real}^{\uparrow}$ consumer surfaces.

Chapter 238

A Concrete Naming Certificate for $\mathsf{SimplicialComplex}^{\uparrow}$

The simplicial-complex interface $\mathsf{SimplicialComplex}^{\uparrow}$ packages a finite collection of simplices closed under taking faces, with face-incidence and dimension-grading data. It depends on $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112) and $\mathsf{NameCert}_{\mathsf{Poset}^{\uparrow}}$ (chapter 53).

238.1 Carrier definition

238.2 The certificate

Definition 238.1 (SimplicialComplexUp finite face carrier)**.** *Fix a finite vertex source* $V : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_V$*. A* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carrier* K *over* $(V, \sim_V)$ *consists of a simplex-code source* $S_K : \mathsf{BHist} \to \mathsf{Prop}$ *with classifier* $\sim_K$*, a finite spine* $ks : \mathsf{HistSpine}$*, a carried-simplex predicate* $\mathsf{Simplex}_K : \mathsf{BHist} \to \mathsf{Prop}$*, a face-incidence relation*

$$\mathsf{Face}_K : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop},$$

and a dimension-grading row on carried simplices.

The finite-simplex row is

$$\mathsf{FinSetSpineEnumerates}_{S_K, \sim_K}(ks, \mathsf{Simplex}_K),$$

using the $\mathsf{FinSet}^{\uparrow}$ *enumeration predicate of Definition 112.17. The face-incidence row is the* $\mathsf{Poset}^{\uparrow}$ *relation on simplex codes supplied by Definition 53.2; in particular it includes the transitivity field*

$$\forall \rho\, \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Face}_K(\rho, \tau) \wedge \mathsf{Face}_K(\tau, \sigma) \to \mathsf{Face}_K(\rho, \sigma).$$

The simplicial closure row says that faces of carried simplices are carried:

$$\forall \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Simplex}_K(\sigma) \wedge \mathsf{Face}_K(\tau, \sigma) \to \mathsf{Simplex}_K(\tau).$$

Thus the carrier uses the finite inventory from chapter 112 and the transitive face-order row from chapter 53.

Lean4 (definition) `BEDC.Derived.SimplicialComplexUp.SimplicialComplexFiniteFaceCarrier`

Theorem 238.2 (Simplicial face-chain closure)**.** *Let K be a* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carrier over $(V, \sim_V)$ as in Definition 238.1. For histories $\rho, \tau, \sigma : \mathsf{BHist}$, if*

$$\mathsf{Simplex}_K(\sigma) \wedge \mathsf{Face}_K(\tau, \sigma) \wedge \mathsf{Face}_K(\rho, \tau),$$

then

$$\mathsf{Simplex}_K(\tau) \wedge \mathsf{Simplex}_K(\rho) \wedge \mathsf{Face}_K(\rho, \sigma).$$

In particular, a face of a face of a carried simplex is again carried by K and is a face of the original simplex.

```
Lean4 ✓  BEDC.Derived.SimplicialComplexUp.SimplicialComplexFace chain closure
  ↪variant  BEDC.Derived.SimplicialComplexUp.SimplicialFaceCarrier face chain closure
```

Proof. Unpack the carrier rows from Definition 238.1. The displayed hypotheses give the carried upper simplex $\mathsf{Simplex}_K(\sigma)$, the first face incidence $\mathsf{Face}_K(\tau, \sigma)$, and the second face incidence $\mathsf{Face}_K(\rho, \tau)$.

Apply the simplicial closure row to $\mathsf{Simplex}_K(\sigma)$ and $\mathsf{Face}_K(\tau, \sigma)$. This gives

$$\mathsf{Simplex}_K(\tau).$$

Apply the same closure row again to this carried simplex and $\mathsf{Face}_K(\rho, \tau)$. This gives

$$\mathsf{Simplex}_K(\rho).$$

The face-incidence relation is the $\mathsf{Poset}^{\uparrow}$ row of the carrier. Its transitivity field, cited in Definition 53.2, composes $\mathsf{Face}_K(\rho, \tau)$ with $\mathsf{Face}_K(\tau, \sigma)$ and yields

$$\mathsf{Face}_K(\rho, \sigma).$$

Combining the carried intermediate simplex, the carried lower simplex, and the composite face-incidence witness gives the displayed conjunction. □

Definition 238.3 (Simplicial face dimension grading row)**.** *Let K be a* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carrier over $(V, \sim_V)$ as in Definition 238.1. A face dimension grading row for K consists of a grading map*

$$\dim_K : \mathsf{BHist} \to \mathbb{N}$$

and the carried-face monotonicity field

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_K(\alpha) \wedge \mathsf{Simplex}_K(\beta) \wedge \mathsf{Face}_K(\alpha, \beta) \to \dim_K(\alpha) \leq \dim_K(\beta).$$

Thus the dimension comparison is asserted on the carried endpoints of the face relation supplied by the $\mathsf{Poset}^{\uparrow}$ *row.*

Theorem 238.4 (Simplicial face-chain dimension monotonicity)**.** *Let K be a* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carrier over $(V, \sim_V)$ as in Definition 238.1, equipped with a face dimension grading row $\dim_K$ in the sense of Definition 238.3. For histories $\rho, \tau, \sigma : \mathsf{BHist}$, if*

$$\mathsf{Simplex}_K(\sigma) \wedge \mathsf{Face}_K(\rho, \tau) \wedge \mathsf{Face}_K(\tau, \sigma),$$

then

$$\dim_K(\rho) \leq \dim_K(\tau) \wedge \dim_K(\tau) \leq \dim_K(\sigma) \wedge \dim_K(\rho) \leq \dim_K(\sigma).$$

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexFace dimension chain monotonicity`

Proof. Unpack the displayed hypothesis. Apply Definition 238.2, whose proof uses the simplicial closure row of Definition 238.1 and the transitive $\mathsf{Poset}^{\uparrow}$ face row of Definition 53.2. This gives

$$\mathsf{Simplex}_K(\tau) \wedge \mathsf{Simplex}_K(\rho) \wedge \mathsf{Face}_K(\rho, \sigma).$$

Apply the carried-face monotonicity field of Definition 238.3 to ρ and τ, using $\mathsf{Simplex}_K(\rho)$, $\mathsf{Simplex}_K(\tau)$, and $\mathsf{Face}_K(\rho, \tau)$. This yields

$$\dim_K(\rho) \leq \dim_K(\tau).$$

Apply the same field to τ and σ, using $\mathsf{Simplex}_K(\tau)$, $\mathsf{Simplex}_K(\sigma)$, and $\mathsf{Face}_K(\tau, \sigma)$. This yields

$$\dim_K(\tau) \leq \dim_K(\sigma).$$

Finally apply the same field to the composite face incidence $\mathsf{Face}_K(\rho, \sigma)$ together with the carried endpoints $\mathsf{Simplex}_K(\rho)$ and $\mathsf{Simplex}_K(\sigma)$. This yields

$$\dim_K(\rho) \leq \dim_K(\sigma).$$

The three displayed comparisons are exactly the claimed conjunction. □

Definition 238.5 (Simplicial intersection carrier rows)**.** *Let K and L be $\mathsf{SimplicialComplex}^{\uparrow}$ finite face carriers over the same finite vertex source $(V, \sim_V)$ as in Definition 238.1. Assume that both carriers use the same simplex-code source $S : \mathsf{BHist} \to \mathsf{Prop}$ with classifier $\sim_S$, and that their face-incidence rows are read through one displayed relation*

$$\mathsf{Face} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}.$$

The simplicial intersection predicate is

$$\mathsf{Simplex}_{K \cap L}(z) := \mathsf{Simplex}_K(z) \wedge \mathsf{Simplex}_L(z).$$

For a finite spine $js : \mathsf{HistSpine}$, the finite enumeration row for the intersection is

$$\mathsf{FinSetSpineEnumerates}_{S, \sim_S}(js, \mathsf{Simplex}_{K \cap L}),$$

in the sense of Definition 112.17. The inherited face relation is

$$\mathsf{Face}_{K \cap L}(\rho, \sigma) := \mathsf{Face}(\rho, \sigma).$$

Lean4 (definition) `BEDC.Derived.SimplicialComplexUp.SimplicialComplexIntersectionCarrier`

Theorem 238.6 (Simplicial intersection carrier is face closed)**.** *Let K and L share the simplex-code source, classifier, and face relation as in Definition 238.5. Let $js : \mathsf{HistSpine}$ satisfy*

$$\mathsf{FinSetSpineEnumerates}_{S, \sim_S}(js, \mathsf{Simplex}_{K \cap L}).$$

Then the intersection rows carry the finite enumeration, face closure, and inherited face transitivity:

$$\mathsf{FinSetSpineEnumerates}_{S, \sim_S}(js, \mathsf{Simplex}_{K \cap L}),$$

$$\forall \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Simplex}_{K \cap L}(\sigma) \wedge \mathsf{Face}_{K \cap L}(\tau, \sigma) \to \mathsf{Simplex}_{K \cap L}(\tau),$$

and

$$\forall \rho\, \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Face}_{K \cap L}(\rho, \tau) \wedge \mathsf{Face}_{K \cap L}(\tau, \sigma) \to \mathsf{Face}_{K \cap L}(\rho, \sigma).$$

Moreover, every two-step face chain in the intersection satisfies

$$\mathsf{Simplex}_{K \cap L}(\sigma) \wedge \mathsf{Face}_{K \cap L}(\tau, \sigma) \wedge \mathsf{Face}_{K \cap L}(\rho, \tau) \to \mathsf{Simplex}_{K \cap L}(\tau) \wedge \mathsf{Simplex}_{K \cap L}(\rho) \wedge \mathsf{Face}_{K \cap L}(\rho, \sigma).$$

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexIntersectionCarrier face chain closure`

Proof. The finite-enumeration row is exactly the displayed hypothesis, with the predicate named in Definition 238.5 and the enumeration predicate supplied by Definition 112.17.

For face closure, fix $\tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cap L}(\sigma) \wedge \mathsf{Face}_{K\cap L}(\tau, \sigma).$$

Unfolding the intersection predicate gives $\mathsf{Simplex}_K(\sigma)$ and $\mathsf{Simplex}_L(\sigma)$, and unfolding the inherited relation gives $\mathsf{Face}(\tau, \sigma)$. The simplicial closure row of K from Definition 238.1 gives $\mathsf{Simplex}_K(\tau)$. The same closure row for L gives $\mathsf{Simplex}_L(\tau)$. Pairing these two conclusions folds back to

$$\mathsf{Simplex}_{K\cap L}(\tau).$$

For inherited face transitivity, fix $\rho, \tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Face}_{K\cap L}(\rho, \tau) \wedge \mathsf{Face}_{K\cap L}(\tau, \sigma).$$

Unfold the inherited relation to read these as $\mathsf{Face}(\rho, \tau)$ and $\mathsf{Face}(\tau, \sigma)$. The face-incidence row of K is the $\mathsf{Poset}^{\uparrow}$ relation row from Definition 53.2 packaged by Definition 238.1; its transitivity field composes the two displayed face incidences and yields $\mathsf{Face}(\rho, \sigma)$. Folding the inherited relation gives $\mathsf{Face}_{K\cap L}(\rho, \sigma)$.

For the two-step chain statement, start with the displayed chain hypothesis. The face-closure row just proved gives $\mathsf{Simplex}_{K\cap L}(\tau)$ from the upper face incidence. Applying the same row again to $\mathsf{Simplex}_{K\cap L}(\tau)$ and $\mathsf{Face}_{K\cap L}(\rho, \tau)$ gives $\mathsf{Simplex}_{K\cap L}(\rho)$. The inherited transitivity row gives $\mathsf{Face}_{K\cap L}(\rho, \sigma)$. Combining these three conclusions proves the final conjunction. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexIntersection face chain closure`

Theorem 238.7 (Simplicial intersection inherits face dimension grading)**.** *Let K and L be* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carriers over the same finite vertex source $(V, \sim_V)$, sharing the simplex-code source, classifier, and face relation as in Definition 238.5. Let $d : \mathsf{BHist} \to \mathbb{N}$ be a common face-dimension grading map: its K-row and L-row are*

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_K(\alpha) \wedge \mathsf{Simplex}_K(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \le d(\beta),$$

and

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_L(\alpha) \wedge \mathsf{Simplex}_L(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \le d(\beta).$$

Then the intersection predicate and inherited face relation satisfy the face-dimension grading row

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_{K\cap L}(\alpha) \wedge \mathsf{Simplex}_{K\cap L}(\beta) \wedge \mathsf{Face}_{K\cap L}(\alpha, \beta) \to d(\alpha) \le d(\beta).$$

Moreover, every two-step face chain in the intersection satisfies

$$\mathsf{Simplex}_{K\cap L}(\sigma) \wedge \mathsf{Face}_{K\cap L}(\tau, \sigma) \wedge \mathsf{Face}_{K\cap L}(\rho, \tau) \to d(\rho) \le d(\tau) \wedge d(\tau) \le d(\sigma) \wedge d(\rho) \le d(\sigma).$$

Proof. For the one-step grading row, fix histories $\alpha, \beta : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cap L}(\alpha) \wedge \mathsf{Simplex}_{K\cap L}(\beta) \wedge \mathsf{Face}_{K\cap L}(\alpha, \beta).$$

Unfolding the intersection predicate from Definition 238.5 gives $\mathsf{Simplex}_K(\alpha)$, $\mathsf{Simplex}_L(\alpha)$, $\mathsf{Simplex}_K(\beta)$, and $\mathsf{Simplex}_L(\beta)$. Unfolding the inherited face relation gives $\mathsf{Face}(\alpha, \beta)$. The K-row of the common grading map, which is the carried-face monotonicity field of Definition 238.3 read through the shared relation, gives

$$d(\alpha) \leq d(\beta).$$

The L-row applies to the same unfolded face incidence and the L-endpoints, giving the identical comparison for the same map d. Hence the displayed intersection row holds.

For the two-step assertion, fix $\rho, \tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cap L}(\sigma) \wedge \mathsf{Face}_{K\cap L}(\tau, \sigma) \wedge \mathsf{Face}_{K\cap L}(\rho, \tau).$$

Unfolding gives $\mathsf{Simplex}_K(\sigma)$ and $\mathsf{Simplex}_L(\sigma)$, together with $\mathsf{Face}(\tau, \sigma)$ and $\mathsf{Face}(\rho, \tau)$. The simplicial closure row of Definition 238.1 gives $\mathsf{Simplex}_K(\tau)$ and $\mathsf{Simplex}_L(\tau)$, hence $\mathsf{Simplex}_{K\cap L}(\tau)$. Applying the same closure rows to $\mathsf{Face}(\rho, \tau)$ gives $\mathsf{Simplex}_K(\rho)$ and $\mathsf{Simplex}_L(\rho)$, hence $\mathsf{Simplex}_{K\cap L}(\rho)$. The transitivity field of the shared $\mathsf{Poset}^{\uparrow}$ face row from Definition 53.2, packaged by Definition 238.1, composes $\mathsf{Face}(\rho, \tau)$ with $\mathsf{Face}(\tau, \sigma)$ and gives $\mathsf{Face}(\rho, \sigma)$, which folds back to $\mathsf{Face}_{K\cap L}(\rho, \sigma)$.

Apply the inherited one-step grading row to (ρ, τ), to (τ, σ), and to (ρ, σ). These three applications give

$$d(\rho) \leq d(\tau),$$

$$d(\tau) \leq d(\sigma),$$

and

$$d(\rho) \leq d(\sigma).$$

Combining the three comparisons gives the claimed chain conjunction. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexIntersection face dimension grading`

Definition 238.8 (Simplicial union carrier rows)**.** *Let K and L be* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carriers over the same finite vertex source $(V, \sim_V)$ as in Definition 238.1. Assume that both carriers use the same simplex-code source $S : \mathsf{BHist} \to \mathsf{Prop}$ with classifier $\sim_S$, and that their face-incidence rows are read through one displayed relation*

$$\mathsf{Face} : \mathsf{BHist} \to \mathsf{BHist} \to \mathsf{Prop}.$$

The simplicial union predicate is

$$\mathsf{Simplex}_{K\cup L}(z) := \mathsf{Simplex}_K(z) \vee \mathsf{Simplex}_L(z).$$

For a finite spine $js : \mathsf{HistSpine}$, the finite enumeration row for the union is

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(js, \mathsf{Simplex}_{K\cup L}),$$

in the sense of Definition 112.17. The inherited face relation is

$$\mathsf{Face}_{K\cup L}(\rho, \sigma) := \mathsf{Face}(\rho, \sigma).$$

Theorem 238.9 (Simplicial union carrier is face closed)**.** *Let K and L share the simplex-code source, classifier, and face relation as in Definition 238.8. Let $js : \mathsf{HistSpine}$ satisfy*

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(js, \mathsf{Simplex}_{K\cup L}).$$

Then the union rows carry the finite enumeration, face closure, and inherited face transitivity:

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(js, \mathsf{Simplex}_{K\cup L}),$$

$$\forall \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Simplex}_{K\cup L}(\sigma) \wedge \mathsf{Face}_{K\cup L}(\tau,\sigma) \to \mathsf{Simplex}_{K\cup L}(\tau),$$

and

$$\forall \rho\, \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Face}_{K\cup L}(\rho,\tau) \wedge \mathsf{Face}_{K\cup L}(\tau,\sigma) \to \mathsf{Face}_{K\cup L}(\rho,\sigma).$$

Moreover, every two-step face chain in the union satisfies

$$\mathsf{Simplex}_{K\cup L}(\sigma) \wedge \mathsf{Face}_{K\cup L}(\tau,\sigma) \wedge \mathsf{Face}_{K\cup L}(\rho,\tau) \to \mathsf{Simplex}_{K\cup L}(\tau) \wedge \mathsf{Simplex}_{K\cup L}(\rho) \wedge \mathsf{Face}_{K\cup L}(\rho,\sigma).$$

Proof. The finite-enumeration row is exactly the displayed hypothesis, with the predicate named in Definition 238.8 and the enumeration predicate supplied by Definition 112.17.

For face closure, fix $\tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cup L}(\sigma) \wedge \mathsf{Face}_{K\cup L}(\tau,\sigma).$$

Unfolding the union predicate gives either $\mathsf{Simplex}_K(\sigma)$ or $\mathsf{Simplex}_L(\sigma)$, and unfolding the inherited relation gives $\mathsf{Face}(\tau,\sigma)$. In the K case, the simplicial closure row of Definition 238.1 gives $\mathsf{Simplex}_K(\tau)$, hence $\mathsf{Simplex}_{K\cup L}(\tau)$. In the L case, the same closure row for L gives $\mathsf{Simplex}_L(\tau)$, hence again $\mathsf{Simplex}_{K\cup L}(\tau)$.

For inherited face transitivity, fix $\rho, \tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Face}_{K\cup L}(\rho,\tau) \wedge \mathsf{Face}_{K\cup L}(\tau,\sigma).$$

Unfold the inherited relation to read these as $\mathsf{Face}(\rho,\tau)$ and $\mathsf{Face}(\tau,\sigma)$. The face-incidence row of K is the $\mathsf{Poset}^{\uparrow}$ relation row from Definition 53.2 packaged by Definition 238.1; its transitivity field composes the two displayed face incidences and yields $\mathsf{Face}(\rho,\sigma)$. Folding the inherited relation gives $\mathsf{Face}_{K\cup L}(\rho,\sigma)$.

For the two-step chain statement, start with the displayed chain hypothesis. The face-closure row just proved gives $\mathsf{Simplex}_{K\cup L}(\tau)$ from the upper face incidence. Applying the same row again to $\mathsf{Simplex}_{K\cup L}(\tau)$ and $\mathsf{Face}_{K\cup L}(\rho,\tau)$ gives $\mathsf{Simplex}_{K\cup L}(\rho)$. The inherited transitivity row gives $\mathsf{Face}_{K\cup L}(\rho,\sigma)$. Combining these three conclusions proves the final conjunction. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion face chain closure`

Theorem 238.10 (Simplicial union append enumeration)**.** *Let K and L share the simplex-code source and classifier used for the simplicial union predicate in Definition 238.8. Let ks, ls :* $\mathsf{HistSpine}$ *be finite BEDC-history spines. If*

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(ks, \mathsf{Simplex}_K) \qquad \textit{and} \qquad \mathsf{FinSetSpineEnumerates}_{S,\sim_S}(ls, \mathsf{Simplex}_L)$$

hold, then the appended spine enumerates the simplicial union predicate:

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(ks{+}{+}ls, \mathsf{Simplex}_{K\cup L}).$$

Proof. Apply Definition 112.19 with the shared simplex-code source S, classifier $\sim_S$, finite spines ks and ls, and predicates $\mathsf{Simplex}_K$ and $\mathsf{Simplex}_L$. The two displayed enumeration hypotheses are exactly its two inputs, so it gives

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(ks{+}{+}ls, \lambda z.\ \mathsf{Simplex}_K(z) \vee \mathsf{Simplex}_L(z)).$$

By Definition 238.8, the predicate on the right is precisely $\mathsf{Simplex}_{K\cup L}$. Rewriting the displayed conclusion by this definition gives

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(ks{+}{+}ls, \mathsf{Simplex}_{K\cup L}),$$

which is the claimed appended enumeration row. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion append enumeration`

Theorem 238.11 (Simplicial union inherits face dimension grading)**.** *Let K and L be* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carriers over the same finite vertex source $(V, \sim_V)$, sharing the simplex-code source, classifier, and face relation as in Definition 238.8. Let $d : \mathsf{BHist} \to \mathbb{N}$ be a common face-dimension grading map: its K-row and L-row are*

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_K(\alpha) \wedge \mathsf{Simplex}_K(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \le d(\beta),$$

and

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_L(\alpha) \wedge \mathsf{Simplex}_L(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \le d(\beta).$$

Then the union predicate and inherited face relation satisfy the face-dimension grading row

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_{K\cup L}(\alpha) \wedge \mathsf{Simplex}_{K\cup L}(\beta) \wedge \mathsf{Face}_{K\cup L}(\alpha, \beta) \to d(\alpha) \le d(\beta).$$

Moreover, every two-step face chain in the union satisfies

$$\mathsf{Simplex}_{K\cup L}(\sigma) \wedge \mathsf{Face}_{K\cup L}(\tau, \sigma) \wedge \mathsf{Face}_{K\cup L}(\rho, \tau) \to$$

$$d(\rho) \le d(\tau) \wedge d(\tau) \le d(\sigma) \wedge d(\rho) \le d(\sigma).$$

Proof. The face-closure and inherited-transitivity rows for the union surface are the rows isolated in Definition 238.9. For the grading comparison, keep the branch of the upper simplex visible.

For the one-step grading row, fix histories $\alpha, \beta : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cup L}(\alpha) \wedge \mathsf{Simplex}_{K\cup L}(\beta) \wedge \mathsf{Face}_{K\cup L}(\alpha, \beta).$$

Unfolding Definition 238.8 gives $\mathsf{Face}(\alpha, \beta)$ and a branch for the upper endpoint β. If $\mathsf{Simplex}_K(\beta)$ holds, the simplicial closure row of K from Definition 238.1 gives $\mathsf{Simplex}_K(\alpha)$ from $\mathsf{Face}(\alpha, \beta)$. The K-grading row then gives $d(\alpha) \le d(\beta)$. If $\mathsf{Simplex}_L(\beta)$ holds, the same argument with the closure row of L gives $\mathsf{Simplex}_L(\alpha)$, and the L-grading row gives $d(\alpha) \le d(\beta)$. Thus the displayed union grading row holds.

For the two-step chain statement, fix $\rho, \tau, \sigma : \mathsf{BHist}$ and assume

$$\mathsf{Simplex}_{K\cup L}(\sigma) \wedge \mathsf{Face}_{K\cup L}(\tau, \sigma) \wedge \mathsf{Face}_{K\cup L}(\rho, \tau).$$

Unfolding the inherited relation gives $\mathsf{Face}(\tau, \sigma)$ and $\mathsf{Face}(\rho, \tau)$. Case-analyze the upper simplex branch $\mathsf{Simplex}_{K\cup L}(\sigma)$.

If $\mathsf{Simplex}_K(\sigma)$ holds, the closure row of K gives $\mathsf{Simplex}_K(\tau)$, and applying it again gives $\mathsf{Simplex}_K(\rho)$. The transitivity field of the shared $\mathsf{Poset}^{\uparrow}$ face row from Definition 53.2, packaged by Definition 238.1, composes the two face incidences and gives $\mathsf{Face}(\rho, \sigma)$. Applying the K-grading row to (ρ, τ), (τ, σ), and (ρ, σ) gives

$$d(\rho) \le d(\tau),$$

$$d(\tau) \le d(\sigma),$$

and

$$d(\rho) \leq d(\sigma).$$

If $\mathsf{Simplex}_L(\sigma)$ holds, the identical branch argument using the closure row of L gives $\mathsf{Simplex}_L(\tau)$ and $\mathsf{Simplex}_L(\rho)$; the same shared transitivity field gives $\mathsf{Face}(\rho, \sigma)$; and the L-grading row applied to the same three face incidences gives the same three displayed comparisons. Combining the three comparisons in either upper branch proves the chain conjunction. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion face dimension grading`

Lemma 238.12 (Simplicial union finite face-closed carrier inherits face dimension grading)**.** *Let K and L be* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face carriers over the same finite vertex source $(V, \sim_V)$, sharing the simplex-code source, classifier, and face relation as in Definition 238.8. Let $d : \mathsf{BHist} \to \mathbb{N}$ be a common face-dimension grading map: its K-row and L-row are*

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_K(\alpha) \wedge \mathsf{Simplex}_K(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \leq d(\beta),$$

and

$$\forall \alpha\, \beta : \mathsf{BHist}.\ \mathsf{Simplex}_L(\alpha) \wedge \mathsf{Simplex}_L(\beta) \wedge \mathsf{Face}(\alpha, \beta) \to d(\alpha) \leq d(\beta).$$

Let js : $\mathsf{HistSpine}$ *satisfy*

$$\mathsf{FinSetSpineEnumerates}_{S,\sim_S}(js, \mathsf{Simplex}_{K \cup L}),$$

and assume face closure and inherited transitivity:

$$\forall \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Simplex}_{K \cup L}(\sigma) \wedge \mathsf{Face}_{K \cup L}(\tau, \sigma) \to \mathsf{Simplex}_{K \cup L}(\tau),$$

$$\forall \rho\, \tau\, \sigma : \mathsf{BHist}.\ \mathsf{Face}_{K \cup L}(\rho, \tau) \wedge \mathsf{Face}_{K \cup L}(\tau, \sigma) \to \mathsf{Face}_{K \cup L}(\rho, \sigma).$$

For histories α, β : BHist*, if*

$$\mathsf{Simplex}_{K \cup L}(\alpha) \wedge \mathsf{Simplex}_{K \cup L}(\beta) \wedge \mathsf{Face}_{K \cup L}(\alpha, \beta),$$

then

$$d(\alpha) \leq d(\beta).$$

Moreover, for histories ρ, τ, σ : BHist*, if*

$$\mathsf{Simplex}_{K \cup L}(\sigma) \wedge \mathsf{Face}_{K \cup L}(\tau, \sigma) \wedge \mathsf{Face}_{K \cup L}(\rho, \tau),$$

then

$$d(\rho) \leq d(\tau) \wedge d(\tau) \leq d(\sigma) \wedge d(\rho) \leq d(\sigma).$$

Proof. Apply Definition 238.11. Its one-step and two-step conclusions are exactly the two graded comparison claims for the union predicate and inherited face relation over the same shared carrier rows from Definition 238.8. The finite and face-closed hypotheses identify the finite face-closed context in which this application is intended, and therefore the two displayed implications are the target inequalities. □

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion face dimension grading rows`

Lean4 ✓ `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion face dimension grading surface`

Closure status for $\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}$.

Constructive story: A simplicial complex is generated from a finite simplex-code spine together with a face relation supplied by the $\mathsf{Poset}^{\uparrow}$ row and finite enumeration supplied by $\mathsf{FinSet}^{\uparrow}$. The carrier accepts simplex histories listed by the finite spine and closes downward along displayed face incidences, while a code outside the enumeration or a face relation without the shared poset transitivity row is not part of this seed surface. Two presentations are compared through the shared simplex-code classifier, inherited face relation, and common dimension-grading row, so intersections and unions are built by pointwise conjunction or disjunction of carried simplex predicates rather than by an external geometric realization. The source is the finite face carrier over a fixed vertex source, the pattern records face-chain closure, dimension monotonicity, intersection closure, union closure, appended finite enumeration, and union grading rows, and the ledger keeps simplex codes, finite spines, face incidences, branches, and grading comparisons visible. The name-certificate obligations at this seed level are visible in the same rows: an accepted finite carrier gives the inhabited surface, the shared classifiers provide reflexive, symmetric, and transitive comparison, and face-chain, union, and intersection closure repack transported faces back into carried simplices. A history h lies in the signature gap of a $\mathsf{SimplicialComplex}^{\uparrow}$ token p when the source-domain accepts its finite face carrier, the signature relation pairs it with an answer carrier and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h lists the same finite simplex inventory, face-chain closure, and dimension-grading data encoded by p. The chapter registers this finite face-poset construction under the name $\mathsf{SimplicialComplex}^{\uparrow}$; homology, geometric realization, a full public simplicial-complex object, and a standard bridge are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature $\mathsf{SimplicialComplex}^{\uparrow}$ surface is the finite face-poset union and dimension-grading consumer interface over the carried simplex codes, face incidences, finite spines, BHist, hsame, Pkg, and NameCert rows displayed in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SimplicialComplexUp.SimplicialComplexUnion\_face\_dimension\_grading\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close a full public simplicial-complex object, homology, geometric realization, or standard bridge.

Upgrade path: Only Lean-side bridge checking remains beyond the mature finite face-poset union and dimension-grading consumer surface.

Chapter 239

A Concrete Naming Certificate for $\mathsf{PersistentHom}^{\uparrow}$

The persistent-homology interface $\mathsf{PersistentHom}^{\uparrow}$ packages the bigraded module of homology groups of a filtered simplicial complex together with the persistence-diagram / barcode summary. It depends on $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ (chapter 98) and $\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}$ (chapter 238).

239.1 Carrier definition

Theorem 239.1 (PersistentHomUp carrier obligation surface)**.** *An accepted* $\mathsf{PersistentHom}^{\uparrow}$ *row is a finite* BHist *filtration packet over a displayed* $\mathsf{SimplicialComplex}^{\uparrow}$ *carrier together with* $\mathsf{Homology}^{\uparrow}$ *rows for the filtration stages. Its visible data consist of the finite index spine, the simplicial-complex packet at each index, the boundary maps between adjacent stages, the homology rows read from those complexes, the persistence maps induced by the boundary ledger, and the* Pkg *provenance row binding all of these entries into one named packet.*

Proof. Inspect the accepted BHist packet. The carrier row contains only the finite filtration spine, the stagewise $\mathsf{SimplicialComplex}^{\uparrow}$ packets, the $\mathsf{Homology}^{\uparrow}$ readback rows, the displayed persistence-map routes, and their common provenance.

Each projected row is already a BEDC history component rather than an ambient topological object. The finite spine supplies the index boundary, the simplicial-complex rows supply the face carriers at each stage, and the homology rows supply the cycle-boundary readbacks.

Repacking these projections gives exactly the carrier obligation surface: a finite filtration packet whose accepted rows are the displayed histories and no additional geometric realization, chain-complex library, or barcode object. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier obligation surface`

Theorem 239.2 (PersistentHomUp filtration classifier stability)**.** *If two accepted* $\mathsf{PersistentHom}^{\uparrow}$ *packets are related by the displayed classifier, then transport along their* hsame *rows and* Cont *routes preserves the finite filtration-index rows, the stagewise* $\mathsf{SimplicialComplex}^{\uparrow}$ *rows, the* $\mathsf{Homology}^{\uparrow}$ *readback rows, the persistence maps between stages, and the shared* Pkg *provenance.*

Proof. Unpack the classifier as componentwise history comparison. Its index component is an hsame row on the finite filtration spine, so transported endpoints keep the same displayed index boundary.

For each carried index, the classifier also contains the transported $\mathsf{SimplicialComplex}^{\uparrow}$ and $\mathsf{Homology}^{\uparrow}$ rows. Their source certificates remain the same named rows after transport because the classifier stores the endpoint histories rather than replacing them by external complexes.

Finally, inspect the Cont routes and Pkg row. The route entries identify the persistence maps between the transported stage rows, while the provenance entry is carried unchanged as the common package witness. These projections are precisely the advertised stability clauses. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier classifier stability`

Theorem 239.3 (PersistentHomUp simplicial homology source boundary)**.** *A downstream consumer of an accepted* $\mathsf{PersistentHom}^{\uparrow}$ *row may read only the source certificates for the displayed* $\mathsf{SimplicialComplex}^{\uparrow}$ *stages, the source certificates for the displayed* $\mathsf{Homology}^{\uparrow}$ *rows, and the finite filtration ledger showing how those rows are ordered and connected. It cannot read an unstated complex, an unstated boundary quotient, or a barcode row outside the displayed ledger.*

Proof. Project the consumer interface from the accepted carrier. The only named source certificates in the packet are the $\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}$ rows for the stages and the $\mathsf{NameCert}_{\mathsf{Homology}^{\uparrow}}$ rows for the associated homology readbacks.

The filtration ledger supplies the order relation on the finite index spine and the Cont routes connecting adjacent stages. Therefore every consumer-visible map is traced through a displayed history row and its package provenance.

Any row not listed in those projections has no constructor in the accepted packet. It is therefore outside the source boundary for the $\mathsf{PersistentHom}^{\uparrow}$ interface. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier source boundary`

Theorem 239.4 (PersistentHomUp filtration ledger exactness)**.** *The filtration ledger of an accepted* $\mathsf{PersistentHom}^{\uparrow}$ *row exposes exactly the finite filtration indices, the boundary maps between adjacent stages, the* $\mathsf{Homology}^{\uparrow}$ *rows computed from the displayed* $\mathsf{SimplicialComplex}^{\uparrow}$ *stages, the barcode summary rows read from those persistence maps, the* Cont *routes that witness stage transport, and the* Pkg *provenance that binds the packet.*

Proof. Read the ledger by constructor inspection. Its index entries are the finite filtration spine, and its stage entries are the displayed simplicial-complex and homology histories attached to those indices.

The boundary-map and persistence-map rows are exactly the Cont entries whose endpoints are adjacent stage histories. The barcode summary rows are ledger summaries of those same routes, so they do not introduce a separate external barcode carrier.

The remaining displayed row is the Pkg provenance. Combining these projections gives both directions of exactness: every listed component is visible in the ledger, and every ledger component is one of the listed rows. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier ledger exactness`

Theorem 239.5 (PersistentHomUp local NameCert surface)**.** *The local* NameCert *obligation surface for* $\mathsf{PersistentHom}^{\uparrow}$ *is assembled from the finite filtration carrier, the filtration-classifier stability row, the simplicial and homology source boundary, and the filtration-ledger exactness row. Thus a local persistent-homology name certificate records a carried* BHist *packet, classifier transport by* hsame *and* Cont*, source readback through* $\mathsf{SimplicialComplex}^{\uparrow}$ *and* $\mathsf{Homology}^{\uparrow}$*, and package provenance through* Pkg*.*

Proof. Use Definition 239.1 for the carried packet. It supplies the finite filtration spine, stage rows, homology rows, persistence maps, and package provenance.

Use Definition 239.2 for classifier transport and Definition 239.3 for the consumer-visible source rows. These two projections give the hsame and Cont movement required by the local certificate.

Use Definition 239.4 for the ledger field. Repacking the carrier, classifier, source, ledger, and provenance rows into the NameCert template gives the local obligation surface and does not assert a separate Lean theorem or standard bridge. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier local namecert surface`

239.2 The certificate

Theorem 239.6 (PersistentHomUp empty filtration spine ledger collapse)**.** *Let P be an accepted* $\mathsf{PersistentHom}^{\uparrow}$ *packet. Suppose that the finite filtration-index spine of P is empty, in the sense that no displayed index entry of P is carried by that spine. Then the filtration ledger of P has no stagewise* $\mathsf{SimplicialComplex}^{\uparrow}$ *packet row, no stagewise* $\mathsf{Homology}^{\uparrow}$ *readback row, no adjacent-stage boundary-map row, no persistence-map row, and no barcode-summary row. Apart from the* Pkg *provenance row, every persistence-indexed ledger component of P is empty.*

Proof. Use Definition 239.1 to read the accepted carrier. Its visible non-provenance data are the finite index spine, the $\mathsf{SimplicialComplex}^{\uparrow}$ packet at each carried index, the boundary maps between adjacent stages, the $\mathsf{Homology}^{\uparrow}$ rows read from those displayed complexes, and the persistence maps induced by the boundary ledger.

Use Definition 239.4 for the ledger side. It says that the filtration ledger exposes exactly the finite filtration indices, the adjacent-stage boundary maps, the $\mathsf{Homology}^{\uparrow}$ rows computed from the displayed $\mathsf{SimplicialComplex}^{\uparrow}$ stages, the barcode summary rows read from the persistence maps, the Cont routes witnessing stage transport, and the Pkg provenance binding the packet.

If a stagewise $\mathsf{SimplicialComplex}^{\uparrow}$ packet row were present, the carrier description would attach it to a displayed filtration index. This contradicts emptiness of the index spine. A $\mathsf{Homology}^{\uparrow}$ readback row is computed from such a displayed stage, so it cannot occur either. An adjacent-stage boundary-map row has adjacent stage endpoints, hence displayed stage rows and their carried indices; these are absent. A persistence-map row is induced by that adjacent-stage boundary ledger, so it has no source row once the boundary-map rows are absent. Finally, every barcode-summary row is read from the persistence maps by ledger exactness, and therefore no barcode-summary row remains.

Thus all index-dependent filtration, stage, homology, persistence-map, and barcode entries are empty. The only row not eliminated by the empty-spine hypothesis is the package provenance row, which records that the accepted packet is named but supplies no additional persistence-indexed content. □

Lean4 ✓ `BEDC.Derived.PersistentHomUp.PersistentHomFiltrationCarrier empty spine ledger collapse`

Closure status for $\mathsf{NameCert}_{\mathsf{PersistentHom}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PersistentHom}^{\uparrow}$ packet is generated from a finite BHist filtration spine, stagewise $\mathsf{SimplicialComplex}^{\uparrow}$ rows, $\mathsf{Homology}^{\uparrow}$ readback rows, displayed persistence-map routes, and a barcode-facing ledger. Each stage is read as a history component, and each adjacent map is replayed through Cont rather than imported from an ambient chain-complex library. The classifier transports the index spine, stage packets, homology rows, and persistence routes by componentwise hsame, while Pkg names the simplicial and homology dependencies. Empty-spine collapse is therefore a visible packet degeneration: all indexed rows disappear and only provenance remains.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PersistentHom}^{\uparrow}}, \mathsf{obligation})$

Scope closed: This chapter binds the filtration carrier obligation surface, classifier stability, source-boundary, barcode ledger obligation, and empty-spine ledger collapse for $\mathsf{PersistentHom}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PersistentHom}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim geometric realization, quotient homology classes beyond the displayed rows, persistence stability over metric barcodes, or external barcode equality.

Upgrade path: Scoped closure requires binding the listed obligations to their kernel primitives and exposing the resulting verification target for $\mathsf{PersistentHom}^{\uparrow}$.

Chapter 240

A Concrete Naming Certificate for $\mathsf{Mapper}^{\uparrow}$

The Mapper interface $\mathsf{Mapper}^{\uparrow}$ packages, for a continuous function from a topological space to a parameter range, the simplicial-complex skeleton built from clusterings of preimages of a finite cover. It depends on $\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}$ (chapter 238) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88).

240.1 Carrier definition

Definition 240.1 (Mapper cover-preimage carrier)**.** *A* $\mathsf{Mapper}^{\uparrow}$ *carrier is a finite* BHist *packet consisting of a* $\mathsf{Topology}^{\uparrow}$ *open-cover row, displayed preimage rows for the cover elements, clustering ledger rows on those preimages, incidence rows recording when clusters meet, and a* $\mathsf{SimplicialComplex}^{\uparrow}$ *skeleton row whose simplices are read from the incidence ledger. The packet also carries the visible endpoint data and* hsame/Cont/Pkg *provenance for the cover, preimage, cluster, and simplex rows.*

The carrier is the finite cover-preimage packet. It contains no host clustering algorithm, no metric convergence assertion, no arbitrary compactness principle for open covers, and no standard bridge object.

Lean4 ✓ `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier source readback`

Theorem 240.2 (Mapper cluster classifier stability)**.** *Let two* $\mathsf{Mapper}^{\uparrow}$ *carrier packets have open-cover rows, displayed preimage rows, and clustering ledger rows related by displayed* hsame *classifiers. If the incidence rows and simplicial skeleton row are read only from those transported cover, preimage, and cluster rows, then the transported packet is again an accepted* $\mathsf{Mapper}^{\uparrow}$ *carrier with the same cluster-incidence shape.*

Lean4 ✓ `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier cluster classifier stability`

Proof. Project the $\mathsf{Topology}^{\uparrow}$ dependency certificate. It transports the open-cover row and the displayed preimage rows attached to the cover elements.

Transport the clustering ledger rows along the same endpoint comparisons. The cluster rows retain their source preimages because the preimage endpoints are part of the transported cover packet.

Replay the incidence ledger on the transported clusters. Since the simplicial skeleton row is read from incidence and simplex rows already in the finite packet, it repacks over the transported data with the same shape. □

Theorem 240.3 (Mapper simplicial ledger exactness)**.** *For every accepted* $\mathsf{Mapper}^{\uparrow}$ *carrier, the simplicial skeleton ledger exposes exactly the cover row, preimage rows, cluster rows, incidence rows, and simplex rows listed in the finite packet. It does not expose a host clustering algorithm, metric convergence theorem, arbitrary open-cover compactness principle, or standard bridge claim.*

Lean4 ✓ `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier simplicial ledger exactness`

Proof. Inspect the carrier of Definition 240.1. Its accepted rows are cover, preimage, cluster, incidence, and skeleton rows, with their displayed provenance.

Project the skeleton row through the $\mathsf{SimplicialComplex}^{\uparrow}$ dependency certificate. The simplices are exactly those read from the incidence rows already present in the packet.

Since the packet has no constructor for algorithmic clustering, metric limits, compactness, or a bridge object, none of those host claims is exported by the simplicial ledger. □

Theorem 240.4 (Mapper NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Mapper}^{\uparrow}}$ *surface consists of the cover-preimage carrier of Definition 240.1, the cluster classifier over displayed* $\mathsf{Topology}^{\uparrow}$ *rows, the stability clause of Definition 240.2, the simplicial ledger exactness clause of Definition 240.3, and the source coverage assertion that every accepted Mapper row is read from the finite cover, preimage, cluster, incidence, and simplex rows.*

Lean4 ✓ `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier namecert obligation surface`

Proof. Use the cover-preimage carrier as the source predicate. It lists precisely the rows that may enter the local naming certificate.

The classifier is the displayed comparison of cover, preimage, and cluster rows. Its transport obligation is Definition 240.2.

The ledger exactness and source coverage obligations are supplied by Definition 240.3 and the carrier definition. Together they expose the Mapper skeleton as a finite BEDC packet and rule out host analytic or algorithmic assertions. □

Theorem 240.5 (Mapper topology boundary exclusions)**.** *The* $\mathsf{Mapper}^{\uparrow}$ *boundary supplies only the BEDC cover-preimage and simplicial skeleton surface over* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{SimplicialComplex}^{\uparrow}}$*. It does not assert host clustering algorithms, metric convergence, arbitrary open-cover compactness, or standard bridge claims.*

Proof. Read the boundary through Definition 240.4. The exported data are the carrier, classifier, stability, skeleton ledger, exactness, and source coverage rows.

Each exported row factors through the finite carrier of Definition 240.1. That carrier records only cover, preimage, cluster, incidence, and simplex rows with their displayed provenance.

Therefore the topology boundary is exactly the stated BEDC packet and carries none of the excluded host claims. □

Lean4 ✓ `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier topology boundary exclusions`

Theorem 240.6 (Mapper public export)**.** *The cluster-classifier stability, simplicial-ledger exactness,* $\mathsf{NameCert}_{\mathsf{Mapper}^{\uparrow}}$ *obligation surface, and topology-boundary exclusions form the public* $\mathsf{Mapper}^{\uparrow}$ *export over the cover-preimage carrier rows.*

The export consists only of the finite cover, preimage, cluster, incidence, simplex, transport, continuation, provenance, and local naming rows. It does not export persistent-homology completeness, an analytic cover-refinement theorem, or a host topological quotient.

Proof. Project the finite carrier of Definition 240.1. Its rows are the cover, preimage, clustering, incidence, simplicial skeleton, transport, continuation, provenance, and local naming rows.

Apply Definition 240.2 and Definition 240.3. These identify the accepted cluster transport and the exact simplicial ledger available from the packet.

Use Definition 240.4 and Definition 240.5 to repack those rows as the public naming boundary. Since the four cited rows all project from the same finite carrier, the excluded completeness, analytic refinement, and quotient claims are not exported. □

240.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{Mapper}^{\uparrow}}$.

Constructive story: MapperUp is presented by the cover-preimage carrier, the topology cover row, the simplicial-complex ledger, hsame, Cont, and Pkg provenance. These rows record finite cover membership and simplicial incidence over BEDC histories only, so the chapter does not export a host cover object or a persistent-homology completeness theorem.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Mapper}^{\uparrow}}$, publicClosed)

Scope closed: Definition 240.6 binds the cover-preimage carrier, cluster classifier, simplicial ledger exactness, NameCert obligation surface, and topology-boundary exclusions over $\mathsf{Topology}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, and Pkg.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Mapper}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MapperUp.MapperCoverPreimageCarrier\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: The chapter does not close host covers, persistent-homology completeness, or standard bridge data.

Upgrade path: Bridged closure requires a standard $\mathsf{Mapper}^{\uparrow}$ bridge over the finite cover-preimage carrier and simplicial ledger.

Chapter 241

A Concrete Naming Certificate for $\mathsf{ErrorCode}^{\uparrow}$

The error-correcting-code interface $\mathsf{ErrorCode}^{\uparrow}$ packages a subset of $\mathbb{F}_q^n$ at minimum Hamming distance d together with encoder, syndrome, and decoder data correcting up to $\lfloor (d-1)/2 \rfloor$ errors. It depends on $\mathsf{NameCert}_{\mathsf{Field}^{\uparrow}}$ (chapter 45) and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47).

241.1 Carrier definition

241.2 The certificate

Definition 241.1 (ErrorCode minimum-distance decoding interface)**.** *An* $\mathsf{ErrorCode}^{\uparrow}$ *certificate over the field certificate of chapter 45 and the vector-space certificate of chapter 47 carries the following metric data. Its ambient words are histories of the certified vector space* $\mathbb{F}_q^n$*. It has a codeword predicate* $\mathsf{Code}_{\mathcal{C}}(x)$*, a code classifier* $\rho_{\mathcal{C}}(x, y)$ *on codeword histories, and a Hamming-distance row*

$$\Delta_{\mathcal{C}}(x, y) \in \mathbb{N}.$$

The Hamming row is symmetric and satisfies the triangle inequality on ambient word histories. The minimum-distance field for a positive natural number d *is recorded in the constructive direction*

$$\mathsf{Code}_{\mathcal{C}}(x) \wedge \mathsf{Code}_{\mathcal{C}}(y) \wedge \Delta_{\mathcal{C}}(x, y) < d \Longrightarrow \rho_{\mathcal{C}}(x, y).$$

The decoding-radius field records the natural number

$$t = \lfloor (d-1)/2 \rfloor.$$

A word r *is within decoding radius* t *of a codeword* c *when* $\Delta_{\mathcal{C}}(c, r) \leq t$*.*

Lean4 (definition) `BEDC.Derived.ErrorCodeUp.ErrorCodeMinimumDistanceInterface`

Lemma 241.2 (Half minimum-distance radius bound)**.** *Let* d *be a positive natural number and let* $t = \lfloor (d-1)/2 \rfloor$*. Then* $2t < d$*.*

Proof. The floor relation gives $t \leq (d-1)/2$. Multiplying by 2 gives $2t \leq d-1$. Positivity of d gives $d-1 < d$. Thus $2t < d$. □

Theorem 241.3 (ErrorCode unique decoding radius)**.** *Let $\mathcal{C}$ be an* $\mathsf{ErrorCode}^{\uparrow}$ *certificate with Hamming-distance row $\Delta_{\mathcal{C}}$, codeword predicate* $\mathsf{Code}_{\mathcal{C}}$*, code classifier $\rho_{\mathcal{C}}$, positive minimum distance d, and decoding radius $t = \lfloor (d-1)/2 \rfloor$ as in Definition 241.1. Let c_1, c_2, and r be ambient word histories. If* $\mathsf{Code}_{\mathcal{C}}(c_1)$*,* $\mathsf{Code}_{\mathcal{C}}(c_2)$*, and*

$$\Delta_{\mathcal{C}}(c_1, r) \leq t \quad \textit{and} \quad \Delta_{\mathcal{C}}(c_2, r) \leq t,$$

then c_1 and c_2 are code-classifier equal:

$$\rho_{\mathcal{C}}(c_1, c_2).$$

Lean4 ✓ `BEDC.Derived.ErrorCodeUp.ErrorCode unique decoding radius`

Proof. The Hamming triangle row gives

$$\Delta_{\mathcal{C}}(c_1, c_2) \leq \Delta_{\mathcal{C}}(c_1, r) + \Delta_{\mathcal{C}}(r, c_2).$$

By symmetry, $\Delta_{\mathcal{C}}(r, c_2) = \Delta_{\mathcal{C}}(c_2, r)$, so the two radius bounds give

$$\Delta_{\mathcal{C}}(c_1, c_2) \leq t + t = 2t.$$

By Definition 241.2, $2t < d$. Hence

$$\Delta_{\mathcal{C}}(c_1, c_2) < d.$$

The minimum-distance field in Definition 241.1 sends two carried codeword histories at Hamming distance below d to a code-classifier witness. Applying that field to the codeword witnesses for c_1 and c_2 yields $\rho_{\mathcal{C}}(c_1, c_2)$. □

Lean4 ✓ `BEDC.Derived.ErrorCodeUp.ErrorCodeMetricReceivedWithinZero unique decoding empty`
Lean4 ✓ `BEDC.Derived.ErrorCodeUp.ErrorCodeMinimumDistanceInterface unique decoding radius`
Lean4 ✓ `BEDC.Derived.ErrorCodeUp.ErrorCodeSingleton unique decoding radius`

Corollary 241.4 (ErrorCode smaller-radius uniqueness)**.** *Let $\mathcal{C}$ be an* $\mathsf{ErrorCode}^{\uparrow}$ *certificate with Hamming-distance row $\Delta_{\mathcal{C}}$, codeword predicate* $\mathsf{Code}_{\mathcal{C}}$*, code classifier $\rho_{\mathcal{C}}$, positive minimum distance d, and decoding radius $t = \lfloor (d-1)/2 \rfloor$ as in Definition 241.1. Let s be a natural number with $s \leq t$. For ambient word histories c_1, c_2, and r, suppose* $\mathsf{Code}_{\mathcal{C}}(c_1)$ *and* $\mathsf{Code}_{\mathcal{C}}(c_2)$*, and suppose*

$$\Delta_{\mathcal{C}}(c_1, r) \leq s \quad \textit{and} \quad \Delta_{\mathcal{C}}(c_2, r) \leq s.$$

Then c_1 and c_2 are code-classifier equal:

$$\rho_{\mathcal{C}}(c_1, c_2).$$

Proof. Transitivity of the natural-number order composes the first radius bound with $s \leq t$ and gives

$$\Delta_{\mathcal{C}}(c_1, r) \leq t.$$

The same order-transitivity step composes the second radius bound with $s \leq t$ and gives

$$\Delta_{\mathcal{C}}(c_2, r) \leq t.$$

The two codeword witnesses and these two enlarged radius bounds are exactly the hypotheses of Definition 241.3. Applying that theorem yields $\rho_{\mathcal{C}}(c_1, c_2)$. □

Lean4 ✓ `BEDC.Derived.ErrorCodeUp.ErrorCode smaller radius uniqueness`

Closure status for $\mathsf{NameCert}_{\mathsf{ErrorCode}^{\uparrow}}$.

Constructive story: An $\mathsf{ErrorCode}^{\uparrow}$ packet is generated from BHist word rows over the displayed field and vector-space certificates, a codeword predicate, a code classifier, a Hamming-distance row, and a decoding-radius ledger. The minimum-distance and radius comparisons are read as finite natural-history inequalities, and the decoding conclusion is a classifier row rather than a host codeword equality. Transport by hsame preserves the carried words, distance rows, and received-word witnesses, while Cont records the route from received data to the classifier conclusion and Pkg records the field and vector-space sources. The unique-decoding surface is therefore the finite minimum-distance packet.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ErrorCode}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the minimum-distance decoding interface, half-radius lemma, unique-decoding theorem, zero-radius specialization, and smaller-radius uniqueness for $\mathsf{ErrorCode}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ErrorCode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim an encoder implementation, syndrome decoding algorithm, probabilistic channel model, host vector equality, or full coding-theory optimality.

Upgrade path: Obligation closure requires Lean encodings of the metric interface, radius arithmetic, classifier conclusion, and the displayed uniqueness variants.

Chapter 242

A Concrete Naming Certificate for $\mathsf{Hash}^{\uparrow}$

The hash-function interface $\mathsf{Hash}^{\uparrow}$ packages a deterministic finite-output map with collision-resistance, preimage-resistance, and second-preimage-resistance certificates against a bounded adversary. It depends on $\mathsf{NameCert}_{\mathsf{Func}^{\uparrow}}$ (chapter 132) and $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ (chapter 112).

242.1 Carrier definition

242.2 The certificate

Definition 242.1 (Hash evaluation and success transcripts)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate as in section 242.1 and section 242.2. Its carrier data consist of a message carrier $\mathsf{Msg}_{\mathcal{H}}$ with message classifier $\sigma_{\mathcal{H}}$, a finite digest carrier $\mathsf{Dig}_{\mathcal{H}}$ with digest classifier $\rho_{\mathcal{H}}$ from the finite-output $\mathsf{FinSet}^{\uparrow}$ presentation of chapter 112, and a deterministic function row supplied by the $\mathsf{Func}^{\uparrow}$ substrate of chapter 132. Write*

$$\mathsf{HashEval}_{\mathcal{H}}(m, d)$$

for the row witness that m is a carried message, d is a carried digest, and the hash row of $\mathcal{H}$ evaluates m to the digest representative d.

For message histories x and x', define successful second-preimage transcript by

$$\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x') \iff \exists d, d' : \mathsf{BHist}.\ \mathsf{HashEval}_{\mathcal{H}}(x, d) \wedge \mathsf{HashEval}_{\mathcal{H}}(x', d') \wedge \neg\sigma_{\mathcal{H}}(x, x') \wedge \rho_{\mathcal{H}}(d', d).$$

Define successful collision transcript for the ordered pair (x, x') by

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \iff \exists d, d' : \mathsf{BHist}.\ \mathsf{HashEval}_{\mathcal{H}}(x, d) \wedge \mathsf{HashEval}_{\mathcal{H}}(x', d') \wedge \neg\sigma_{\mathcal{H}}(x, x') \wedge \rho_{\mathcal{H}}(d, d').$$

The two predicates use the same evaluation row; the digest-classifier orientation is the only difference.

Lean4 (definition) `BEDC.Derived.HashUp.HashEvalTranscript`

Theorem 242.2 (Hash evaluation transcript determinacy)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose hash row is the deterministic $\mathsf{Func}^{\uparrow}$ row carried by the certificate and whose digest endpoint classifier is the finite-output classifier $\rho_{\mathcal{H}}$ supplied by the $\mathsf{FinSet}^{\uparrow}$ dependency. For any message history m and digest histories d and d', two carried evaluations over the same hash row determine the same digest class:*

$$\mathsf{HashEval}_{\mathcal{H}}(m, d) \wedge \mathsf{HashEval}_{\mathcal{H}}(m, d') \Longrightarrow \rho_{\mathcal{H}}(d, d').$$

Proof. Unfold the evaluation row of Definition 242.1. The two hypotheses expose the same carried message row m, the same deterministic $\mathsf{Func}^{\uparrow}$ function row, and two digest endpoint rows d and d' produced by that row.

Project determinacy from the $\mathsf{Func}^{\uparrow}$ certificate. Since both evaluations read the same input history through the same finite Cont ledger and the same Pkg provenance, the function-row determinacy comparison identifies the two produced endpoint histories at the digest boundary.

Now transport that endpoint comparison through the finite-output $\mathsf{FinSet}^{\uparrow}$ classifier. The digest carrier row of $\mathcal{H}$ is exactly the finite-output row named in Definition 242.1, so classifier transport yields $\rho_{\mathcal{H}}(d, d')$.

Repack the evidence without changing the transcript: the message row, digest endpoint rows, BHist witnesses, Cont readback, and Pkg provenance are the rows already carried by the two $\mathsf{HashEval}_{\mathcal{H}}$ hypotheses. □

Lean4 ✓ `BEDC.Derived.HashUp.HashEvalTranscript determinacy`

Theorem 242.3 (Hash second-preimage success induces collision)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose finite-output digest classifier is supplied by its $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ component. For message histories x and x', every successful second-preimage transcript for base message x and returned message x' induces a successful collision transcript for the ordered pair (x, x') over the same hash row:*

$$\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x').$$

Lean4 ✓ `BEDC.Derived.HashUp.HashSecondPreimageSuccess induces collision`

Proof. Unfold Definition 242.1. A second-preimage success transcript gives digest histories d and d' with the two evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x', d'),$$

the message distinction witness $\neg\sigma_{\mathcal{H}}(x, x')$, and the digest-classifier comparison $\rho_{\mathcal{H}}(d', d)$. The digest classifier is the classifier of the finite-output naming certificate from chapter 112, so the classifier-symmetry field of Definition 9.7 reverses the displayed comparison and gives

$$\rho_{\mathcal{H}}(d, d').$$

Using the same digest witnesses d, d', the same two evaluation row witnesses, and the same message distinction witness, this is exactly the collision predicate over the same $\mathsf{HashEval}_{\mathcal{H}}$ row in Definition 242.1. □

Theorem 242.4 (Hash collision success is symmetric)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose message classifier $\sigma_{\mathcal{H}}$ and finite-output digest classifier $\rho_{\mathcal{H}}$ have the classifier-symmetry fields supplied by their naming-certificate components. For message histories x and x', every successful collision transcript for the ordered pair (x, x') induces a successful collision transcript for the ordered pair (x', x) over the same hash-evaluation row:*

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x).$$

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionSuccess symmetric`

Proof. Unfold Definition 242.1. A collision transcript for (x, x') gives digest histories d and d' with evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x', d'),$$

the message distinction witness $\neg\sigma_{\mathcal{H}}(x, x')$, and the digest comparison $\rho_{\mathcal{H}}(d, d')$.

The digest-classifier symmetry field from Definition 9.7 reverses the digest comparison to

$$\rho_{\mathcal{H}}(d', d).$$

For the reversed message distinction, suppose $\sigma_{\mathcal{H}}(x', x)$ held. The message-classifier symmetry field would give $\sigma_{\mathcal{H}}(x, x')$, contradicting the displayed witness $\neg\sigma_{\mathcal{H}}(x, x')$. Hence

$$\neg\sigma_{\mathcal{H}}(x', x).$$

Using the swapped digest witnesses d' and d, the two original evaluation row witnesses read in the opposite order, the reversed message distinction, and the reversed digest comparison, the definition repacks exactly as $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x)$ over the same $\mathsf{HashEval}_{\mathcal{H}}$ row. □

Theorem 242.5 (Hash compression round carrier)**.** *Each hash evaluation transcript in Definition 242.1 is a* BHist *packet over the* $\mathsf{Func}^{\uparrow}$ *deterministic row, the message carrier, the finite digest carrier, the* Cont *readback row, and the* Pkg *provenance row. The transcript exports no separate host bitstring computation.*

Lean4 ✓ `BEDC.Derived.HashUp.HashEvalTranscript compression round carrier`

Proof. Unfold Definition 242.1. A row $\mathsf{HashEval}_{\mathcal{H}}(m, d)$ contains a carried message endpoint, a carried digest endpoint, and the hash-function row supplied by the $\mathsf{Func}^{\uparrow}$ substrate.

Project determinacy from Definition 242.2. The two digest rows produced by the same message and function row are compared only through the finite digest classifier inherited from $\mathsf{FinSet}^{\uparrow}$.

Finally read the evaluation route through the displayed Cont readback and Pkg provenance. Repacking these rows gives the compression-round carrier as a BEDC transcript rather than as an external bitstring computation. □

Theorem 242.6 (Hash digest classifier stability)**.** *Transporting digest endpoints by the finite-output* $\mathsf{FinSet}^{\uparrow}$ *digest classifier preserves* $\mathsf{HashEval}_{\mathcal{H}}$*,* $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}$*, and* $\mathsf{HashCollisionSuccess}_{\mathcal{H}}$*. The preservation is a classifier transport of digest rows, not an equality of host strings.*

Lean4 ✓ `BEDC.Derived.HashUp.HashEvalTranscript digest classifier stability`

Proof. Start with an evaluation transcript and a digest-classifier edge supplied by the finite-output $\mathsf{FinSet}^{\uparrow}$ certificate. The digest endpoint remains in the digest carrier, and the same $\mathsf{Func}^{\uparrow}$ row and message carrier row still witness the evaluation.

For second-preimage success, unfold Definition 242.1. The transported digest row replaces the original digest witness while the two evaluation rows and the message-distinction witness remain the same.

For collision success, use the same unpacking together with the digest-classifier symmetry used in Definition 242.4. Repacking the transported digest witnesses preserves the two success predicates through the classifier boundary. □

Theorem 242.7 (Hash collision ledger boundary)**.** *The collision, second-preimage, and preimage-resistance ledgers of a* $\mathsf{Hash}^{\uparrow}$ *certificate may read only message-carrier rows, digest-carrier rows, function determinacy rows, classifier comparisons, and bounded adversary transcripts. They may not read external probability spaces, host bitstring equality, or unrecorded randomness.*

Proof. Unfold the transcript definition of Definition 242.1. Collision and second-preimage success are built from two evaluation rows, one message-classifier distinction, and one digest-classifier comparison.

Apply Definition 242.3 and Definition 242.4. These theorem blocks use only the evaluation rows, classifier symmetry, function determinacy, and the existing transcript witnesses.

The bounded adversary surface therefore has no additional ledger row to consult. Repacking the available witnesses gives the stated boundary for collision, second-preimage, and preimage-resistance ledgers. □

Theorem 242.8 (Hash collision transcript symmetry)**.** *Let* $\mathcal{H}$ *be a* $\mathsf{Hash}^{\uparrow}$ *certificate whose message classifier* $\sigma_{\mathcal{H}}$ *and finite-output digest classifier* $\rho_{\mathcal{H}}$ *have the classifier-symmetry fields supplied by their naming-certificate components. For message histories* x *and* x'*, every successful collision transcript for the ordered pair* (x, x') *induces a successful collision transcript for the ordered pair* (x', x) *over the same two hash-evaluation rows:*

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x).$$

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionTranscript symmetric`

Proof. Unfold Definition 242.1. A collision transcript for (x, x') gives digest histories d and d' with evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x', d'),$$

the message distinction witness $\neg\sigma_{\mathcal{H}}(x, x')$, and the digest comparison $\rho_{\mathcal{H}}(d, d')$.

The digest-classifier symmetry field from Definition 9.7, applied through the finite-output $\mathsf{FinSet}^{\uparrow}$ component cited in chapter 112, reverses the digest comparison to

$$\rho_{\mathcal{H}}(d', d).$$

For the reversed message distinction, suppose $\sigma_{\mathcal{H}}(x', x)$ held. The message-classifier symmetry field from the same naming-certificate discipline would give $\sigma_{\mathcal{H}}(x, x')$, contradicting $\neg\sigma_{\mathcal{H}}(x, x')$. Hence

$$\neg\sigma_{\mathcal{H}}(x', x).$$

Using the swapped digest witnesses d' and d, the two original evaluation row witnesses read in the opposite order, the reversed message distinction, and the reversed digest comparison, the definition repacks exactly as $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x)$ over the same two $\mathsf{HashEval}_{\mathcal{H}}$ rows. □

Theorem 242.9 (Hash collision success induces reversed second-preimage)**.** *Let* $\mathcal{H}$ *be a* $\mathsf{Hash}^{\uparrow}$ *certificate whose message classifier* $\sigma_{\mathcal{H}}$ *has the classifier-symmetry field supplied by its naming-certificate component. For message histories* x *and* x'*, every successful collision transcript for the ordered pair* (x, x') *induces a successful second-preimage transcript for base message* x' *and returned message* x *over the same two hash-evaluation rows:*

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x', x).$$

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionSuccess induces reversed second preimage`

Proof. Unfold Definition 242.1. A collision transcript for (x, x') gives digest histories d and d' with the two evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x', d'),$$

the message distinction witness $\neg\sigma_{\mathcal{H}}(x, x')$, and the digest comparison

$$\rho_{\mathcal{H}}(d, d').$$

To build the second-preimage transcript for (x', x), take the digest witness for the base message x' to be d' and the digest witness for the returned message x to be d. The two displayed evaluation row witnesses then occur in the required reversed order, and the required digest condition is still

$$\rho_{\mathcal{H}}(d, d'),$$

which is the collision digest comparison itself.

It remains only to reverse the message distinction. Suppose $\sigma_{\mathcal{H}}(x', x)$ held. The message-classifier symmetry field from Definition 9.7 would give $\sigma_{\mathcal{H}}(x, x')$, contradicting $\neg\sigma_{\mathcal{H}}(x, x')$. Hence

$$\neg\sigma_{\mathcal{H}}(x', x).$$

Using the swapped digest witnesses d' and d, the same two hash-evaluation rows, the reversed message distinction, and the unchanged digest comparison, the definition repacks exactly as $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x', x)$. □

Definition 242.10 (Hash collision-freeness)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate as in section 242.1 and section 242.2. For histories x and x', define*

$$\mathsf{CollFreeH}_{\mathcal{H}}(x, x') \iff \neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x').$$

Lean4 (definition) `BEDC.Derived.HashUp.HashCollisionFree`

Theorem 242.11 (Hash collision-freeness excludes second-preimage success)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose finite-output digest classifier is supplied by its $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ component. For message histories x and x', if*

$$\mathsf{CollFreeH}_{\mathcal{H}}(x, x') \wedge \mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$$

holds, then

$$\bot.$$

Proof. Unfold $\mathsf{CollFreeH}_{\mathcal{H}}(x, x')$, so it is enough to show $\neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$. By Definition 242.3, the assumed second-preimage success gives

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x').$$

This directly contradicts the displayed collision-freeness hypothesis, so the conjunction is inconsistent. □

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionFree excludes second preimage success`

Theorem 242.12 (Hash collision-freeness is symmetric)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose message classifier $\sigma_{\mathcal{H}}$ and finite-output digest classifier $\rho_{\mathcal{H}}$ have the classifier-symmetry fields supplied by their naming-certificate components. For message histories x and x', if*

$$\mathsf{CollFreeH}_{\mathcal{H}}(x, x')$$

holds, then

$$\mathsf{CollFreeH}_{\mathcal{H}}(x', x).$$

Proof. Assume

$$\mathsf{CollFreeH}_{\mathcal{H}}(x, x').$$

By Definition 242.10, this means

$$\neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x').$$

To prove $\mathsf{CollFreeH}_{\mathcal{H}}(x', x)$, it is enough by Definition 242.10 to show

$$\neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x).$$

Suppose instead that

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x)$$

held. Then Definition 242.4, applied with the ordered pair (x', x) in place of (x, x'), gives

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x').$$

This contradicts the displayed collision-freeness hypothesis. Therefore

$$\neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x', x),$$

and another use of Definition 242.10 yields

$$\mathsf{CollFreeH}_{\mathcal{H}}(x', x).$$

□

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionFree symmetric`

Theorem 242.13 (Hash collision success is irreflexive on the message)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose message classifier $\sigma_{\mathcal{H}}$ carries the reflexivity field supplied by its naming-certificate component. For every message history x, if*

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x)$$

holds, then

$$\bot.$$

Proof. Unfold Definition 242.1. A collision transcript for the diagonal pair (x, x) gives digest histories d and d' with the two evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x, d'),$$

the message distinction witness

$$\neg\sigma_{\mathcal{H}}(x, x),$$

and the digest comparison

$$\rho_{\mathcal{H}}(d, d').$$

The first evaluation row already exhibits x as a carried message of $\mathcal{H}$. Since the message classifier is supplied by a naming certificate, the reflexivity field from Definition 9.7 therefore gives

$$\sigma_{\mathcal{H}}(x, x).$$

This contradicts the displayed witness $\neg\sigma_{\mathcal{H}}(x, x)$. Hence no diagonal collision-success transcript can exist, and the assumption implies $\bot$. □

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionSuccess irreflexive on message`

Theorem 242.14 (Hash collision success induces same-direction second-preimage)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose finite-output digest classifier $\rho_{\mathcal{H}}$ carries the classifier-symmetry field supplied by its $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ component. For message histories x and x', every successful collision transcript for the ordered pair (x, x') induces a successful second-preimage transcript with base message x and returned message x' over the same two hash-evaluation rows:*

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x').$$

Lean4 ✓ `BEDC.Derived.HashUp.HashCollisionSuccess induces same direction second preimage`

Proof. Unfold Definition 242.1. A collision transcript for (x, x') gives digest histories d and d' with evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x', d'),$$

the message distinction witness

$$\neg\sigma_{\mathcal{H}}(x, x'),$$

and the digest comparison

$$\rho_{\mathcal{H}}(d, d').$$

The digest classifier is the classifier of the finite-output naming certificate from chapter 112. Its classifier-symmetry field, as recorded in Definition 9.7, reverses the displayed digest comparison and gives

$$\rho_{\mathcal{H}}(d', d).$$

Keep the same digest witnesses d and d'. The two hash-evaluation rows and the message distinction witness already occur in the order required for a second-preimage transcript with base message x and returned message x', and the reversed digest comparison is exactly the second-preimage digest condition in Definition 242.1. Repacking these witnesses gives

$$\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x').$$

□

Theorem 242.15 (Hash resistance success boundary)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose finite-output digest classifier $\rho_{\mathcal{H}}$ and message classifier $\sigma_{\mathcal{H}}$ are symmetric classifier fields supplied by the naming-certificate components of the certificate. Assume the digest classifier has the symmetry*

field used by Definition 242.14. For message histories x and x', the two transcript predicates over the same two evaluation rows are equivalent:

$$\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$$

and

$$\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x') \Longrightarrow \mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x').$$

Thus collision-freeness and second-preimage resistance share the same transcript boundary under the BEDC classifiers.

Proof. For the forward direction, assume $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$. Apply Definition 242.3. It uses the same two $\mathsf{HashEval}_{\mathcal{H}}$ rows, the same message-classifier distinction, and the digest comparison already carried by the second-preimage transcript, so it returns $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$ without changing the transcript boundary.

For the reverse direction, assume $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$. Apply Definition 242.14. The only transport it needs is the finite-output digest-classifier symmetry field from the $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ component, while the message classifier and the two evaluation rows remain the same ordered pair. This returns $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$.

The two implications identify the successful transcript boundary. Negating the collision side or the second-preimage side therefore excludes the same successful pair of BEDC evaluation rows under the certificate classifiers. □

Theorem 242.16 (Hash resistance negated boundary equivalence)**.** *Let $\mathcal{H}$ be a $\mathsf{Hash}^{\uparrow}$ certificate whose finite-output digest classifier $\rho_{\mathcal{H}}$ and message classifier $\sigma_{\mathcal{H}}$ are symmetric classifier fields supplied by the naming-certificate components of the certificate. Assume the digest classifier has the symmetry field used by Definition 242.14. For message histories x and x', collision-freeness for the ordered pair is equivalent to the negation of second-preimage success:*

$$\mathsf{CollFreeH}_{\mathcal{H}}(x, x') \Longleftrightarrow \neg\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x').$$

Proof. Unfold Definition 242.10. The displayed claim becomes the equivalence between $\neg\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x$ and $\neg\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$ under the same certificate hypotheses.

For the forward direction, assume $\mathsf{CollFreeH}_{\mathcal{H}}(x, x')$ and suppose $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$ held. By Definition 242.3, the second-preimage transcript induces $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$, contradicting the unfolded collision-freeness hypothesis.

For the reverse direction, assume $\neg\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$. To show $\mathsf{CollFreeH}_{\mathcal{H}}(x, x')$, unfold Definition 242.10 and suppose $\mathsf{HashCollisionSuccess}_{\mathcal{H}}(x, x')$ held. By Definition 242.14, that collision transcript induces $\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x')$, contradicting the assumed negation. □

Theorem 242.17 (Hash transcript predicate mature coverage)**.** *For a $\mathsf{Hash}^{\uparrow}$ certificate $\mathcal{H}$, evaluation, collision success, second-preimage success, collision-freeness, resistance success, irreflexivity, and negated-boundary reads are all projections of the displayed message carrier, finite digest carrier, deterministic function row, message and digest classifiers,* Cont *readback, and* Pkg *provenance.*

The mature transcript surface exports these predicate rows and no probabilistic adversary, host bitstring computation, external complexity assumption, or randomized oracle model.

Proof. The evaluation rows are fixed by the deterministic carrier and readback surface of Definition 242.2. Collision reads are bounded by Definition 242.7 and transported by Definition 242.4.

The second-preimage and collision boundaries are related by Definition 242.3 and Definition 242.14. Collision-freeness is the negated transcript predicate of Definition 242.10, and the shared resistance boundary is the surface of Definition 242.15.

The negated boundary is recorded by Definition 242.16, while Definition 242.18 excludes the diagonal second-preimage transcript from the same message classifier. These results mention only the displayed message rows, finite digest rows, deterministic function row, classifiers, continuation readback, and provenance, so the transcript predicates have no exported coordinate for external cryptographic or computational assumptions. □

Closure status for $\mathsf{NameCert}_{\mathsf{Hash}^\uparrow}$**.**

Constructive story: A hash certificate is generated by combining a message carrier, a finite digest carrier from $\mathsf{FinSet}^\uparrow$, and a deterministic function row supplied by $\mathsf{Func}^\uparrow$, then recording each evaluation as a visible transcript from message history to digest history. The carrier accepts evaluation, collision, second-preimage, and collision-freeness rows only when the message and digest endpoints are carried, the function row supplies the evaluation ledger, and the message and digest classifiers supply the distinction or equality comparisons; unrecorded randomness, host bitstrings, and probability spaces are excluded. Two transcripts are compared by the message classifier, the finite digest classifier, and the deterministic function row, so reversing a collision or translating a second preimage into a collision uses classifier symmetry rather than external string equality. The source is the displayed evaluation transcript surface, the pattern records evaluation determinacy, collision symmetry, second-preimage-to-collision implications, collision-freeness, irreflexivity on the message, and shared resistance boundaries, and the ledger keeps both evaluations, both digest endpoints, the message-distinction witness, digest comparison, Cont readback, and Pkg provenance visible. The five name-certificate obligations are supplied by accepted transcript rows for carrier witnesses, by the message and digest naming certificates for reflexivity, symmetry, and transitivity, and by classifier-stability rows showing that transported digest or message endpoints preserve the success predicates. A history h lies in the signature gap of a $\mathsf{Hash}^\uparrow$ token p when the source-domain accepts its evaluation or resistance transcript, the signature relation pairs it with an answer transcript and evidence ledger, and the token-introduction relation names that answer as p; concretely, p records the same carried messages, finite digests, evaluation rows, and collision or second-preimage boundary. The chapter registers this deterministic transcript construction under the name $\mathsf{Hash}^\uparrow$; cryptographic reductions, probabilistic adversary models, external complexity assumptions, and host bitstring computation are not introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Hash}^\uparrow}$, maturePackageClosed)

Scope closed: The mature surface binds the inherited $\mathsf{Func}^\uparrow$ and $\mathsf{FinSet}^\uparrow$ rows, message and digest classifiers, BHist evaluation rows, transcript predicates for collision and second-preimage success, collision-freeness, resistance, irreflexivity, and the negated-boundary coverage of Definition 242.17.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Hash}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.HashUp.HashCollisionFree\_excludes\_second\_preimage\_success`

Bridge status: paperBridge

Not claimed: The chapter does not close cryptographic security reductions, probabilistic adversary models, or external complexity assumptions.

Upgrade path: No further paper theory-axis upgrade is recorded for the deterministic transcript-predicate surface; bridgeChecked remains a separate formal-axis matter.

Theorem 242.18 (Hash second-preimage success is irreflexive on the message)**.** *Let $\mathcal{H}$ be a* $\mathsf{Hash}^\uparrow$ *certificate whose message classifier $\sigma_{\mathcal{H}}$ carries the reflexivity field supplied by its naming-certificate component. For every message history x, if*

$$\mathsf{HashSecondPreimageSuccess}_{\mathcal{H}}(x, x)$$

holds, then

$$\bot.$$

Lean4 ✓ `BEDC.Derived.HashUp.HashSecondPreimageSuccess irreflexive on message`

Proof. Unfold Definition 242.1. A second-preimage transcript for the diagonal pair (x, x) gives digest histories d and d' with the two evaluation row witnesses

$$\mathsf{HashEval}_{\mathcal{H}}(x, d) \qquad \text{and} \qquad \mathsf{HashEval}_{\mathcal{H}}(x, d'),$$

the message distinction witness

$$\neg\sigma_{\mathcal{H}}(x, x),$$

and the digest comparison

$$\rho_{\mathcal{H}}(d', d).$$

The first evaluation row exhibits x as a carried message of $\mathcal{H}$. Since the message classifier is supplied by a naming certificate, the reflexivity field from Definition 9.7 gives

$$\sigma_{\mathcal{H}}(x, x).$$

This contradicts the displayed witness $\neg\sigma_{\mathcal{H}}(x, x)$. Hence no diagonal second-preimage-success transcript can exist, and the assumption implies $\perp$. $\square$

Chapter 243

A Concrete Naming Certificate for $\mathsf{PublicKey}^{\uparrow}$

The public-key-cryptosystem interface $\mathsf{PublicKey}^{\uparrow}$ packages a key-generation, encryption, and decryption triple together with security reductions to a hardness assumption (factoring, discrete logarithm, lattice). It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168).

243.1 Carrier definition

243.2 The certificate

Definition 243.1 (PublicKey decrypt-encrypt transcript interface)**.** *Let $\mathcal{P}$ be a $\mathsf{PublicKey}^{\uparrow}$ certificate over the group-side naming certificate of chapter 41 and the number-field plaintext certificate of chapter 168. Write $\mathsf{Msg}_{\mathcal{P}}(m)$ for its plaintext carrier and $\mu_{\mathcal{P}}(m, m')$ for its plaintext classifier. The plaintext certificate is a carrier naming certificate in the sense of Definition 9.7; in the concrete rational number-field surface it is supplied by Definition 168.5.*

Write $\mathsf{Pub}_{\mathcal{P}}(pk)$, $\mathsf{Sec}_{\mathcal{P}}(sk)$, and $\mathsf{Ciph}_{\mathcal{P}}(c)$ for the public-key, secret-key, and ciphertext carrier rows. The group-side rows named by Definition 41.55 type the key-generation and encryption witnesses. Write

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk)$$

for the row witness that pk and sk are a generated public-key pair, write

$$\mathsf{PKEncrypt}_{\mathcal{P}}(pk, m, c)$$

for the row witness that c is the encryption transcript of plaintext m under public key pk, and write

$$\mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d)$$

for the row witness that secret key sk decrypts ciphertext c to plaintext endpoint d.

A certified encryption transcript is the predicate

$$\mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c) \iff \mathsf{Pub}_{\mathcal{P}}(pk) \wedge \mathsf{Msg}_{\mathcal{P}}(m) \wedge \mathsf{Ciph}_{\mathcal{P}}(c) \wedge \mathsf{PKEncrypt}_{\mathcal{P}}(pk, m, c).$$

The decrypt-encrypt exactness row of $\mathcal{P}$ is

$$\mathsf{PKDecryptEncryptExact}_{\mathcal{P}} \iff \forall pk, sk, m, c : \mathsf{BHist}.\ \mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \wedge \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c) \implies \exists d : \mathsf{BHist}.$$

Lean4 (definition) `BEDC.Derived.PublicKeyUp.PKCertifiedEnc`
Lean4 (definition) `BEDC.Derived.PublicKeyUp.PKDecryptEncryptExact`

Theorem 243.2 (PublicKey decrypt-encrypt correctness)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1, and assume* $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$*. For histories pk, sk, m, and c, if*

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \qquad \textit{and} \qquad \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c),$$

then decrypting c with sk returns a carried plaintext endpoint in the message-classifier class of m:

$$\exists d : \mathsf{BHist}.\ \mathsf{Msg}_{\mathcal{P}}(d) \wedge \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d) \wedge \mu_{\mathcal{P}}(d, m).$$

Lean4 ✓ `BEDC.Derived.PublicKeyUp.PublicKeyDecryptEncrypt correctness`

Proof. Unfold Definition 243.1 only as far as the certified-encryption predicate and the decrypt-encrypt exactness row. The certified-encryption hypothesis gives the carried plaintext fact $\mathsf{Msg}_{\mathcal{P}}(m)$ together with the encryption transcript under pk.

Apply $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$ to the generated keypair row and the certified-encryption transcript. This gives a history d with

$$\mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d) \qquad \text{and} \qquad \mu_{\mathcal{P}}(d, m).$$

The plaintext classifier is supplied by a carrier naming certificate. Its symmetry field reverses $\mu_{\mathcal{P}}(d, m)$ to $\mu_{\mathcal{P}}(m, d)$, and its carrier-transport field carries $\mathsf{Msg}_{\mathcal{P}}(m)$ along that comparison to obtain $\mathsf{Msg}_{\mathcal{P}}(d)$.

Using the same witness d, the carried plaintext fact, the decryption row, and the classifier comparison prove the displayed existential statement. □

Lean4 ✓ `BEDC.Derived.PublicKeyUp.PublicKey decrypt encrypt correctness`

Definition 243.3 (PublicKey decryption output determinacy row)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1. The decrypt-output determinacy row of $\mathcal{P}$ is the predicate*

$$\mathsf{PKDecryptOutputDet}_{\mathcal{P}} \iff \forall pk, sk, c, d_1, d_2 : \mathsf{BHist}.\ \mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \wedge \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d_1) \wedge \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c$$

It records that decryption for a generated secret key and a fixed ciphertext has a plaintext endpoint determined up to the plaintext classifier of the certificate.

Lean4 (definition) `BEDC.Derived.PublicKeyUp.PKDecryptOutputDetRow`

Theorem 243.4 (PublicKey decryption output determinacy)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1, and assume* $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$*. For histories pk, sk, c, d_1, and d_2, if*

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \qquad \textit{and} \qquad \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d_1) \qquad \textit{and} \qquad \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d_2),$$

then the two decryption endpoints are plaintext-classifier equal:

$$\mu_{\mathcal{P}}(d_1, d_2).$$

Proof. By Definition 243.3, the row $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$ is the universal BEDC-history statement that any two decryption endpoints for the same generated keypair row and ciphertext are related by $\mu_{\mathcal{P}}$. Instantiate that universal statement at the displayed histories pk, sk, c, d_1, and d_2. The three hypotheses are exactly its antecedent, so the instantiated implication gives $\mu_{\mathcal{P}}(d_1, d_2)$. □

Theorem 243.5 (PublicKey certified encryption arbitrary decryption)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1. Assume* $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$ *and* $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$*. For histories pk, sk, m, c, and d, if*

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \quad \textit{and} \quad \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c) \quad \textit{and} \quad \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d),$$

then the displayed decryption endpoint is carried by the plaintext carrier and lies in the message-classifier class of m:

$$\mathsf{Msg}_{\mathcal{P}}(d) \wedge \mu_{\mathcal{P}}(d, m).$$

Proof. Apply Definition 243.2 to $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$, the generated keypair row, and the certified encryption transcript. This gives a history $e : \mathsf{BHist}$ with

$$\mathsf{Msg}_{\mathcal{P}}(e) \wedge \mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, e) \wedge \mu_{\mathcal{P}}(e, m).$$

Apply Definition 243.4 to $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$, the same generated keypair row, the displayed decryption row for d, and the decryption row for e. This yields

$$\mu_{\mathcal{P}}(d, e).$$

The plaintext classifier of $\mathcal{P}$ is supplied by a carrier naming certificate in the sense of Definition 9.7; in the concrete rational number-field surface this is the certificate cited in Definition 168.5. Its transitivity field, equivalently Definition 9.8, composes $\mu_{\mathcal{P}}(d, e)$ with $\mu_{\mathcal{P}}(e, m)$ and gives

$$\mu_{\mathcal{P}}(d, m).$$

It remains only to record carrier membership for the arbitrary endpoint. Unfolding $\mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c)$ through Definition 243.1 gives $\mathsf{Msg}_{\mathcal{P}}(m)$. The classifier symmetry field from the same carrier naming certificate reverses $\mu_{\mathcal{P}}(d, m)$ to $\mu_{\mathcal{P}}(m, d)$, and its carrier-transport field carries $\mathsf{Msg}_{\mathcal{P}}(m)$ along that comparison to obtain $\mathsf{Msg}_{\mathcal{P}}(d)$. Pairing this carrier fact with $\mu_{\mathcal{P}}(d, m)$ proves the claim. □

Lean4 ✓ `BEDC.Derived.PublicKeyUp.PublicKeyCertifiedEncryption arbitrary decryption`

Theorem 243.6 (PublicKey certified ciphertext plaintext uniqueness)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1, and assume* $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$ *and* $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$*. For histories pk, sk, m, n, and c, if*

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk) \quad \textit{and} \quad \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c) \quad \textit{and} \quad \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, n, c),$$

then the two certified plaintext endpoints are classifier-equal:

$$\mu_{\mathcal{P}}(m, n).$$

Proof. Apply Definition 243.2 to $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$, the generated keypair row, and the first certified-encryption hypothesis. This gives a history $d : \mathsf{BHist}$ with

$$\mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, d) \wedge \mu_{\mathcal{P}}(d, m).$$

Apply the same theorem to the second certified-encryption hypothesis to obtain $e : \mathsf{BHist}$ with

$$\mathsf{PKDecrypt}_{\mathcal{P}}(sk, c, e) \wedge \mu_{\mathcal{P}}(e, n).$$

Apply Definition 243.4 to $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$, the generated keypair row, and the two decrypt rows. This yields

$$\mu_{\mathcal{P}}(d, e).$$

By the symmetry field of Definition 9.7, reverse $\mu_{\mathcal{P}}(d, m)$ to get $\mu_{\mathcal{P}}(m, d)$.

Compose $\mu_{\mathcal{P}}(m, d)$ with $\mu_{\mathcal{P}}(d, e)$ using Definition 9.8 to obtain

$$\mu_{\mathcal{P}}(m, e).$$

Then compose $\mu_{\mathcal{P}}(m, e)$ with $\mu_{\mathcal{P}}(e, n)$, again by Definition 9.8, to conclude

$$\mu_{\mathcal{P}}(m, n).$$

□

Lean4 ✓ `BEDC.Derived.PublicKeyUp.PublicKeyCertifiedCiphertext plaintext uniqueness`

Theorem 243.7 (PublicKey shared-public-key decryption consistency)**.** *Let $\mathcal{P}$ be a* $\mathsf{PublicKey}^{\uparrow}$ *certificate with transcript rows as in Definition 243.1, and assume* $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$ *and* $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$*. For histories pk, sk_1, sk_2, m, c, d_1, and d_2, if*

$$\mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk_1) \quad \textit{and} \quad \mathsf{PKKeyGen}_{\mathcal{P}}(pk, sk_2) \quad \textit{and} \quad \mathsf{PKCertifiedEnc}_{\mathcal{P}}(pk, m, c),$$

and if

$$\mathsf{PKDecrypt}_{\mathcal{P}}(sk_1, c, d_1) \quad \textit{and} \quad \mathsf{PKDecrypt}_{\mathcal{P}}(sk_2, c, d_2),$$

then the two decryption endpoints are in the same plaintext classifier class:

$$\mu_{\mathcal{P}}(d_1, d_2).$$

Lean4 ✓ `BEDC.Derived.PublicKeyUp.PublicKeySharedPublicKey decryption consistency`

Proof. Apply Definition 243.5 to $\mathsf{PKDecryptEncryptExact}_{\mathcal{P}}$, $\mathsf{PKDecryptOutputDet}_{\mathcal{P}}$, the first generated keypair row, the certified encryption transcript, and the displayed decryption row for d_1. This gives

$$\mathsf{Msg}_{\mathcal{P}}(d_1) \wedge \mu_{\mathcal{P}}(d_1, m).$$

Apply the same theorem to the second generated keypair row, the same certified encryption transcript, and the displayed decryption row for d_2. This gives

$$\mathsf{Msg}_{\mathcal{P}}(d_2) \wedge \mu_{\mathcal{P}}(d_2, m).$$

The plaintext classifier of $\mathcal{P}$ is supplied by the carrier naming certificate recorded in Definition 243.1, hence by Definition 9.7. Use its symmetry field on $\mu_{\mathcal{P}}(d_2, m)$ to obtain

$$\mu_{\mathcal{P}}(m, d_2).$$

Classifier transitivity, equivalently Definition 9.8, composes $\mu_{\mathcal{P}}(d_1, m)$ with $\mu_{\mathcal{P}}(m, d_2)$ and yields

$$\mu_{\mathcal{P}}(d_1, d_2).$$

□

Closure status for $\mathsf{NameCert}_{\mathsf{PublicKey}^{\uparrow}}$.

Constructive story: A $\mathsf{PublicKey}^{\uparrow}$ packet is generated from finite BHist rows for public keys, secret keys, plaintexts, ciphertexts, encryption transcripts, decryption transcripts, and the plaintext classifier inherited from the number-field source. Correctness is read bottom-up: key generation and certified encryption expose displayed rows, Cont routes the decryption endpoint, and the result is compared by the carried classifier rather than by host message equality. Componentwise hsame transports key, ciphertext, plaintext, and transcript rows, while Pkg records the group and number-field dependencies. The security-facing surface remains a finite transcript certificate, not a cryptographic hardness theorem.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PublicKey}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the decrypt-encrypt transcript interface, decrypt-encrypt correctness, and classifier-facing stability results for $\mathsf{PublicKey}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PublicKey}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim semantic security, hardness reductions, randomness soundness, host plaintext equality, or a concrete encryption implementation.

Upgrade path: Obligation closure requires Lean encodings of the transcript carrier, decrypt-encrypt exactness row, and classifier transport for decrypted endpoints.

Chapter 244

A Concrete Naming Certificate for $\mathsf{ZeroKnowledge}^{\uparrow}$

The zero-knowledge-proof interface $\mathsf{ZeroKnowledge}^{\uparrow}$ packages an interactive prover-verifier protocol that is complete, sound, and simulation-paradigm zero-knowledge. It depends on $\mathsf{NameCert}_{\mathsf{Hash}^{\uparrow}}$ (chapter 242) and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

244.1 Carrier definition

A $\mathsf{ZeroKnowledge}^{\uparrow}$ carrier is a finite BHist packet. It displays a prover row P, a verifier row V, a challenge transcript row c, a response transcript row r, a hash commitment row k, a computable verifier row q, a simulator transcript row s, and an evidence ledger row ℓ. The hash row is sourced from $\mathsf{NameCert}_{\mathsf{Hash}^{\uparrow}}$ and the computable verifier row is sourced from $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ through the packet provenance Pkg; neither dependency is read through host bitstrings, probability spaces, or external machines.

The classifier compares two accepted packets only through displayed hsame rows: prover with prover, verifier with verifier, challenge transcript with challenge transcript, response transcript with response transcript, commitment with commitment, computable-verifier row with computable-verifier row, simulator transcript with simulator transcript, and ledger with ledger. The continuation ledger Cont records the transcript routes from prover and verifier to challenge, from challenge and prover to response, from response and hash source to commitment, from challenge and computable verifier to acceptance, from simulator and verifier to simulated transcript, and from the visible transcript rows to the evidence ledger.

244.2 The certificate

Theorem 244.1 (ZeroKnowledgeUp carrier obligation)**.** *For a* $\mathsf{ZeroKnowledge}^{\uparrow}$ *packet with displayed rows* P*,* V*,* c*,* r*,* k*,* q*,* s*, and* ℓ*, the carrier obligation is exactly the finite* BHist *surface consisting of the prover row, verifier row, challenge transcript row, response transcript row, hash commitment row, computable verifier row, simulator transcript row, the* Cont *transcript routes among those rows, and the* Pkg *provenance rows naming the consumed* $\mathsf{Hash}^{\uparrow}$ *and* $\mathsf{Computable}^{\uparrow}$ *certificates. No cryptographic hardness predicate, negligible bound, host randomness source, or external protocol machine is part of this carrier.*

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeCarrier obligation`

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeFinitePacket carrier obligation`

Proof. Project the packet to its displayed histories. The prover, verifier, challenge, response, commitment, computable-verifier, simulator, and evidence rows are the only carrier coordinates.

Next project the continuation evidence. Each visible read is one of the listed Cont routes, and each dependency read is one of the two Pkg provenance rows. Repacking those projections gives the same finite carrier surface, so no unlisted host object is available to the carrier. □

Theorem 244.2 (ZeroKnowledgeUp classifier obligation)**.** *Two accepted* ZeroKnowledge$^{\uparrow}$ *packets classify together only when their prover rows, verifier rows, challenge transcript rows, response transcript rows, hash commitment rows, computable verifier rows, simulator transcript rows, continuation ledgers, and package provenance rows are related by the displayed componentwise* hsame *comparisons. The classifier does not compare hidden witnesses, probability distributions, bitstring encodings, or machine implementations outside the accepted* BHist *packet.*

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeCarrier classifier obligation`

Proof. Read the classifier by projection to the accepted packet coordinates. Each coordinate has a named hsame comparison row, and the dependency rows are compared through the same Pkg provenance that supplied Hash$^{\uparrow}$ and Computable$^{\uparrow}$.

For reflexivity, symmetry, and transitivity, apply the corresponding hsame operation componentwise to the displayed rows. The continuation and provenance rows are then repacked with those transported coordinates, giving a classifier on accepted packets and nothing beyond them. □

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgePacket classifier transport`

Theorem 244.3 (ZeroKnowledgeUp completeness ledger obligation)**.** *The completeness ledger of a* ZeroKnowledge$^{\uparrow}$ *packet is a BEDC obligation over the displayed prover row P, verifier row V, challenge row c, response row r, computable verifier row q, transcript continuation rows, and provenance rows for* Hash$^{\uparrow}$ *and* Computable$^{\uparrow}$*. It states only that the accepted transcript route records a verifier-acceptance row in the packet ledger; it does not assert a probability of acceptance or a completeness error bound.*

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeFiniteCarrier completeness ledger obligation`

Proof. Project the completeness evidence to the transcript route from prover and verifier into challenge, then from challenge and prover into response. The computable verifier row reads that response through the displayed Cont evidence.

The verifier-acceptance row is then packed into the ledger ℓ together with the dependency provenance. Since every object used in this packing is one of the carrier rows of Definition 244.1, the completeness ledger is a finite BEDC row statement rather than a probabilistic claim. □

Theorem 244.4 (ZeroKnowledgeUp soundness ledger obligation)**.** *The soundness ledger of a* ZeroKnowledge$^{\uparrow}$ *packet is a finite BEDC obligation over challenge rows, response rows, hash commitment rows, computable verifier rows,* hsame *comparisons,* Cont *transcript routes, and* Pkg *provenance for* Hash$^{\uparrow}$ *and* Computable$^{\uparrow}$*. It records only the visible rows needed to compare an accepting transcript with the commitment and verifier rows; it does not claim cryptographic hardness, extractor existence, or a negligible cheating bound.*

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeCarrier soundness ledger obligation`

Proof. First project the accepting transcript to its challenge, response, commitment, and computable-verifier rows. The hash dependency is read through the carried $\mathsf{Hash}^{\uparrow}$ provenance, and the verifier computation is read through the carried $\mathsf{Computable}^{\uparrow}$ provenance.

The ledger comparison is then obtained by transporting those rows through their displayed hsame comparisons and repacking them with the transcript Cont routes. This proves only that the packet exposes the finite soundness surface claimed in the statement. □

Theorem 244.5 (ZeroKnowledgeUp simulation ledger obligation)**.** *The simulation ledger of a* $\mathsf{ZeroKnowledge}^{\uparrow}$ *packet is a finite BEDC obligation over the simulator transcript row s, verifier row V, challenge row c, response row r, commitment row k, computable verifier row q, componentwise* hsame *comparisons, transcript* Cont *routes, and* Pkg *provenance for* $\mathsf{Hash}^{\uparrow}$ *and* $\mathsf{Computable}^{\uparrow}$*. It states that the simulated transcript row is compared to the displayed interaction transcript by packet rows alone; it does not assert distributional indistinguishability, zero-knowledge error bounds, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeCarrier simulation ledger obligation`
Lean4 ✓ `BEDC.Derived.ZeroKnowledgeUp.ZeroKnowledgeCarrier simulation transcript boundary`

Proof. Project the simulator evidence to the simulator transcript row and the verifier-facing transcript rows. The comparison with the interaction transcript is a row-by-row hsame comparison between the displayed challenge, response, commitment, and verifier-computation coordinates.

Repack the simulator route, the verifier route, and the dependency provenance into the ledger ℓ. Because the proof uses only accepted carrier rows and their Cont routes, the result is a BEDC simulation surface and not a claim about external distributions. □

Definition 244.6 (ZeroKnowledgeUp public finite-transcript surface)**.** *For a* $\mathsf{ZeroKnowledge}^{\uparrow}$ *packet, the public finite-transcript surface is the following finite* BHist *package: the prover row P, verifier row V, challenge transcript row c, response transcript row r, hash commitment row k, computable verifier row q, simulator transcript row s, evidence ledger row ℓ, the componentwise* hsame *comparisons on those rows, the* Cont *routes from prover and verifier to challenge, from challenge and prover to response, from response and the* $\mathsf{Hash}^{\uparrow}$ *source to commitment, from challenge and the* $\mathsf{Computable}^{\uparrow}$ *source to verifier acceptance, from simulator and verifier to simulated transcript, and from the visible transcript rows to ℓ, together with the* Pkg *provenance rows naming the consumed* $\mathsf{NameCert}_{\mathsf{Hash}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *certificates.*

Theorem 244.7 (ZeroKnowledgeUp public finite-transcript surface)**.** *Let Z be a* $\mathsf{ZeroKnowledge}^{\uparrow}$ *packet satisfying the carrier, classifier, completeness-ledger, soundness-ledger, and simulation-ledger obligations of Definition 244.1, Definition 244.2, Definition 244.3, Definition 244.4, and Definition 244.5. Then the public* $\mathsf{NameCert}_{\mathsf{ZeroKnowledge}^{\uparrow}}$ *surface of Z is exactly the finite-transcript surface of Definition 244.6. In particular, every public consumer read is one of the displayed transcript, classifier, continuation, ledger, or provenance rows, and the packet exports no hidden witness, probability distribution, hardness predicate, extractor, negligible bound, or standard bridge.*

Proof. Project the public certificate to its carrier field. By Definition 244.1, the only carrier coordinates are the rows P, V, c, r, k, q, s, and ℓ, together with the named Cont transcript routes and the two Pkg provenance rows for $\mathsf{Hash}^{\uparrow}$ and $\mathsf{Computable}^{\uparrow}$. This gives the row inventory of Definition 244.6. The commitment coordinate k is read only through the consumed $\mathsf{NameCert}_{\mathsf{Hash}^{\uparrow}}$ row, whose local public use here is the finite hash-evaluation transcript boundary of Definition 242.5; the verifier coordinate q is read only through the consumed $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ row, whose local public use here is the bounded-simulation ledger boundary of Definition 199.6.

Project next to the classifier field. By Definition 244.2, accepted packets classify only by componentwise hsame comparisons on the displayed prover, verifier, challenge, response, commitment, computable-verifier, simulator, continuation-ledger, and provenance rows. Thus a consumer cannot distinguish or identify packets through a witness value, a distribution law, a machine implementation, or any object not carried by those comparisons.

The three ledger obligations exhaust the remaining public reads. By Definition 244.3, completeness contributes only the finite accepted-transcript route through P, V, c, r, q, the transcript continuation rows, and the dependency provenance; it contributes no acceptance probability or error bound. By Definition 244.4, soundness contributes only the visible comparison between accepting transcript rows, commitment rows, verifier rows, hsame transport, Cont routes, and Pkg provenance; it contributes no cryptographic hardness predicate, extractor, or negligible cheating bound. By Definition 244.5, simulation contributes only the simulator transcript row s, the verifier-facing transcript rows, their componentwise hsame comparisons, and the same continuation and provenance rows; it contributes no distributional indistinguishability claim, zero-knowledge error bound, or standard bridge.

Combining the carrier projection, classifier projection, and the three ledger projections gives every row listed in Definition 244.6, so the finite-transcript surface is public. Conversely, the same five obligations leave no public field outside those rows: each public read is either a carrier coordinate, a componentwise classifier comparison, a transcript continuation route, one of the completeness, soundness, or simulation ledger rows, or one of the two dependency provenance rows. Therefore the public $\mathsf{NameCert}_{\mathsf{ZeroKnowledge}^{\uparrow}}$ surface is exactly the displayed finite BHist/Cont/Pkg transcript surface and exports none of the excluded host-level data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroKnowledge}^{\uparrow}}$.

Constructive story: A $\mathsf{ZeroKnowledge}^{\uparrow}$ packet is generated from finite BHist rows for prover, verifier, challenge, response, commitment, computable-verifier, simulator transcript, and evidence ledger data. Completeness, soundness, and simulation appear only as public transcript ledgers: each accepted read is routed by Cont through displayed transcript rows, not through probability spaces or external machines. The classifier is componentwise hsame on prover, verifier, transcript, commitment, simulator, and ledger rows, while Pkg records the $\mathsf{Hash}^{\uparrow}$ and $\mathsf{Computable}^{\uparrow}$ sources. The resulting object is a finite public transcript surface with explicit refusal of hidden cryptographic semantics.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeroKnowledge}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite transcript carrier, classifier projection, completeness, soundness, simulation ledger obligations, and public finite-transcript surface for $\mathsf{ZeroKnowledge}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeroKnowledge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim negligible bounds, distributional indistinguishability, extractor existence, host randomness, or a standard protocol bridge.

Upgrade path: Obligation closure requires Lean encodings of the finite transcript carrier, componentwise classifier, and the three public ledger obligations.

Chapter 245

A Concrete Naming Certificate for $\mathsf{CharacterTheory}^{\uparrow}$

The character-theory interface $\mathsf{CharacterTheory}^{\uparrow}$ packages, for a finite group, the trace functions of its irreducible representations together with the orthogonality relations and class-function basis. It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47).

245.1 Carrier definition

Definition 245.1 (CharacterTheoryUp root BHist surface)**.** *The root* $\mathsf{CharacterTheory}^{\uparrow}$ *surface is a finite* BHist *packet*

$$\mathcal{X}_{\mathsf{CharacterTheory}^{\uparrow}} := (G, V, \rho, \tau, \Omega, \Pi)$$

whose rows have the following meanings:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row for the finite group endpoint.*
2. *V is the consumed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *source row for the representation space endpoint.*
3. *ρ is the displayed representation-action row, read as a finite* Cont *ledger from the group row and vector-space row to an endomorphism row of V.*
4. *τ is the trace row of the displayed action endpoint, carried as a* BHist *readback row and not as a host matrix trace.*
5. *Ω is the orthogonality and class-function ledger, consisting only of the finite rows that record class-function basis reads, trace-pairing reads, and the* hsame *comparisons used to transport those reads.*
6. *$\Pi : \mathsf{Pkg}$ is the provenance row tying G, V, ρ, τ, and Ω to the same accepted package.*

No row of $\mathcal{X}_{\mathsf{CharacterTheory}^{\uparrow}}$ is a quotient of conjugacy classes, a complex inner-product space, or an ambient trace operator outside this finite BHist *packet.*

Lean4 (definition) `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface`

Theorem 245.2 (CharacterTheoryUp root carrier obligation)**.** *An accepted root packet $\mathcal{X}_{\mathsf{CharacterTheory}^{\uparrow}}$ exposes only the* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row G, the* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row V, the representation-action row ρ, the trace row τ, and the provenance row $\Pi : \mathsf{Pkg}$ as carrier-facing data. It exposes no host group element set, no host vector space, no matrix algebra, and no detached trace carrier beyond these displayed rows.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootCarrier obligation`
Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface root carrier obligation`

Proof. Unpack the packet of Definition 245.1. Its carrier-facing coordinates are exactly G, V, ρ, τ, and Π; the orthogonality ledger Ω is a downstream readback ledger rather than a new carrier coordinate.

For each accepted carrier read, project to the corresponding coordinate of the finite packet. The group source is G, the vector source is V, the action is ρ, the trace readback is τ, and the provenance is Π.

Conversely, repack those five rows into the displayed tuple and leave Ω available only as ledger data. Since the definition lists no other constructor for accepted root packets, there is no additional host carrier or trace object to expose. □

Theorem 245.3 (CharacterTheoryUp root classifier obligation)**.** *If two accepted root packets have* hsame*-transported group source rows,* hsame*-transported vector-space source rows, and* hsame*-transported trace rows, then they determine the same* $\mathsf{CharacterTheory}^{\uparrow}$ *character classifier. The classifier transport uses only those source and trace transports together with the representation-action row of Definition 245.1.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface root classifier obligation`

Proof. Project the two accepted packets to their G, V, ρ, and τ rows. The displayed hsame transport on G identifies the group source endpoints, and the displayed hsame transport on V identifies the vector-space endpoints that carry the representation row.

Transport the action row ρ along the two source comparisons. The trace row τ is then transported along the given trace comparison, so the character readback obtained from the transported action has the same BHist classifier endpoint as the original readback.

Repack the transported source rows, action row, trace row, ledger row, and provenance row as an accepted packet. The character classifier reads only this transported trace endpoint, so the classifier is preserved. □

Theorem 245.4 (CharacterTheoryUp root ledger obligation)**.** *Every downstream orthogonality read and every class-function-basis read of an accepted* $\mathsf{CharacterTheory}^{\uparrow}$ *root packet factors through the finite* Cont *and* Pkg *ledger rows of* Ω *and* Π*. Such reads introduce no host trace, quotient class object, or complex inner product beyond the rows displayed in Definition 245.1.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface root ledger obligation`

Proof. Unpack the ledger coordinate Ω. Its entries are finite rows for trace-pairing reads, class-function basis reads, and the hsame comparisons that transport those reads across accepted source endpoints.

Any downstream orthogonality query first projects to the trace row τ and then follows a finite Cont row inside Ω. Any downstream class-function-basis query projects to the same packet and follows the basis read rows recorded in Ω.

The provenance row $\Pi : \mathsf{Pkg}$ ties those ledger entries back to the same accepted group source, vector-space source, action, and trace rows. Repacking the query through Ω and Π leaves no remaining coordinate from which a host trace, quotient class object, or complex inner product could be read. □

245.2 The certificate

Theorem 245.5 (CharacterTheoryUp source compatibility obligation)**.** *Let* $\mathcal{X}_{\mathsf{CharacterTheory}^{\uparrow}} = (G, V, \rho, \tau, \Omega, \Pi)$ *be an accepted packet of Definition 245.1. If the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row* G *and consumed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row* V *move along displayed* hsame *rows, and if the representation-action row* ρ *is transported by the finite* Cont *ledger recorded with the packet, then the resulting source-compatible trace row and its provenance remain in the same finite* BHist *and* Pkg *packet. No host trace operator, quotient class object, or detached carrier is introduced by this transport.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface source compatibility obligation`

Proof. Project the six coordinates G, V, ρ, τ, Ω, and Π from Definition 245.1. The displayed hsame rows supply the group and vector-space source comparisons, and the finite Cont ledger supplies the action comparison for ρ.

Apply Definition 245.3 to the transported source and action data. The classifier comparison is therefore read through the transported trace endpoint rather than through an ambient trace object.

Finally repack the transported source rows, action row, trace row, and ledger entries with the same provenance coordinate $\Pi : \mathsf{Pkg}$. Since each coordinate is projected from and returned to the same finite BHist packet, no host trace or quotient class object is available. □

Theorem 245.6 (CharacterTheoryUp trace transport stability)**.** *For an accepted packet* $\mathcal{X}_{\mathsf{CharacterTheory}^{\uparrow}} = (G, V, \rho, \tau, \Omega, \Pi)$*, every orthogonality read and every class-function-basis read recorded in* Ω *transports along the source-compatible trace comparison of Definition 245.5. Each transported read factors through finite* Cont*,* hsame*, and* Pkg *rows of the displayed* BHist *packet.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface trace transport stability`

Proof. Apply Definition 245.4 to project the orthogonality and class-function-basis reads from Ω. The theorem confines those reads to finite Cont rows, the relevant hsame comparisons, and the provenance row $\Pi : \mathsf{Pkg}$.

Use Definition 245.5 to transport the trace endpoint supplied by the group and vector-space source comparisons. The ledger reads then follow the transported trace row inside the same finite packet.

Repack the transported read rows with Ω and Π. Because the reads have no endpoint outside the displayed BHist coordinates, stability is exactly preservation of those finite ledger rows. □

Theorem 245.7 (CharacterTheoryUp NameCert obligation surface)**.** *The root packet definition of Definition 245.1, together with Definition 245.2, Definition 245.3, Definition 245.4, Definition 245.5, and Definition 245.6, supplies the carrier, classifier, source-compatibility, stability, ledger, and no-host-boundary fields required of the* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface carrier obligation`

Proof. Enumerate the six packet fields G, V, ρ, τ, Ω, and $\Pi : \mathsf{Pkg}$ from Definition 245.1. The carrier face is Definition 245.2, and the classifier face is Definition 245.3.

The source-compatibility field is Definition 245.5. The stability field is Definition 245.6, and the ledger field is Definition 245.4.

The remaining no-host-boundary field follows by finite constructor projection and repacking. Every field just listed reads from the displayed BHist, Cont, hsame, and Pkg rows, so the

$\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ surface receives no quotient class carrier, host trace, or external representation object. □

Theorem 245.8 (CharacterTheoryUp scoped source route)**.** *Every accepted* $\mathsf{CharacterTheory}^{\uparrow}$ *packet factors through exactly the finite source rows displayed in this chapter: the group-action carrier row, trace readback row, class-function classifier row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *row. The route exports no external character table, host trace equality, quotient conjugacy-class enumeration, or ambient representation category outside those rows.*

Lean4 ✓ `BEDC.Derived.CharacterTheoryUp.CharacterTheoryRootBHistSurface scoped source route`

Proof. Project the accepted root carrier of Definition 245.1. The group source, vector-space source, representation action, trace readback, orthogonality and class-function ledger, and provenance rows are the only carrier coordinates available.

Use the root carrier, classifier, and ledger obligations, Definition 245.2, Definition 245.3, and Definition 245.4. They force carrier reads, classifier comparison, and ledger reads to pass through the displayed finite rows rather than through host trace equality or an external character table.

Finally apply source compatibility, trace transport, and the NameCert obligation surface, Definition 245.5, Definition 245.6, and Definition 245.7. These results repack all transported reads through hsame, Cont, Pkg, and the local naming row, leaving no exported quotient conjugacy-class enumeration or ambient representation category. □

Theorem 245.9 (CharacterTheoryUp scoped source induction)**.** *The root carrier, classifier, ledger, source compatibility, trace transport, NameCert obligation surface, and scoped source route jointly bind every* $\mathsf{CharacterTheory}^{\uparrow}$ *read to the displayed* $\mathsf{Group}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *rows. Induction over accepted reads therefore has no step that leaves the group source row, vector-space source row, representation-action row, trace readback row, class-function ledger, componentwise transport, replay, provenance, and local naming surface.*

Proof. Start with the root carrier of Definition 245.1. Its rows display the group source, vector-space source, representation action, trace readback, class-function ledger, and provenance surface.

Use the carrier, classifier, and ledger obligations, Definition 245.2, Definition 245.3, and Definition 245.4. These bind every carrier read, classifier comparison, and ledger read to the displayed source rows and their finite transport data.

Finally apply source compatibility, trace transport stability, the NameCert obligation surface, and the scoped source route. They close the induction step under hsame, Cont, Pkg, and local naming, so no read can escape to a host trace operator, quotient conjugacy-class enumeration, or external representation category. □

Closure status for $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CharacterTheory}^{\uparrow}$ packet is generated from finite BHist rows for a representation source, class-function or trace readbacks, root or character ledgers, source-compatibility rows, and the local naming row. The trace-facing data are displayed histories transported by hsame, so equality of characters is not imported as host class-function equality. The Cont rows replay how representation rows feed the trace and root ledgers, and Pkg records the representation-theoretic dependencies that the packet consumes. The public surface is therefore the finite character ledger and its stability obligations, with quotient classes and ambient representation objects left outside.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 245.9 supplies the bridge route through $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows while keeping trace, class-function, and representation reads inside the finite CharacterTheory packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{CharacterTheory}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: It does not claim full character-table computation, orthogonality relations, quotient conjugacy classes, host trace equality, or external representation categories.
Upgrade path: The paper closure is complete; the remaining work is Lean verification and bridge evidence for the displayed finite character-theory rows.

Chapter 246

A Concrete Naming Certificate for $\mathsf{RepresentationRing}^{\uparrow}$

The representation-ring interface $\mathsf{RepresentationRing}^{\uparrow}$ packages the Grothendieck ring of finite-dimensional representations of a group under direct sum and tensor product. It depends on $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ (chapter 41) and $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ (chapter 43).

246.1 Carrier definition

Definition 246.1 (RepresentationRingUp BHist representation packet)**.** *A* $\mathsf{RepresentationRing}^{\uparrow}$ *representation packet is a finite* BHist *tuple*

$$\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda).$$

The row G is the consumed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *packet, and R is the consumed* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *packet. The row V is the finite list of representation endpoints carried as* BHist *rows: each endpoint has its group action source read through G and its additive and multiplicative target rows read through R. The row S records the displayed direct-sum faces, and T records the displayed tensor-product faces. Both are finite* Cont *ledgers whose endpoints are members of V.*

The row P is the provenance packet listing the Pkg *rows consumed from G and R and the package rows attached to S and T. The row ρ is the Grothendieck classifier row naming formal direct-sum balance among the displayed representation endpoints. The finite ledger λ is the* Cont *list containing exactly the consumed GroupUp rows, consumed RingUp rows, representation rows, direct-sum rows, tensor rows, classifier row, and provenance rows exposed by the public packet.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingTasteGate single carrier alignment`

Definition 246.2 (RepresentationRingUp Grothendieck ledger classifier)**.** *For two representation packets*

$$\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda) \qquad \text{and} \qquad \mathcal{K}' = (G', R', V', S', T', P', \rho', \lambda'),$$

the Grothendieck ledger classifier $\mathcal{K} \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}'$ *is the finite componentwise transport datum consisting of:*

1. *an* hsame *transport row between the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *packets G and G';*
2. *an* hsame *transport row between the consumed* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *packets R and R';*

3. *an entrywise* hsame *transport row between the displayed representation endpoint lists V and V';*
4. *entrywise* hsame *transport rows between the direct-sum ledgers S and S' and between the tensor-product ledgers T and T';*
5. *an* hsame *transport row between the provenance packets P and P';*
6. *an* hsame *transport row between the Grothendieck classifier rows ρ and ρ';*
7. *a finite* Cont *ledger comparison between λ and λ', entry by entry.*

Thus equality in the public representation-ring carrier is not host equality of representation categories. It is the displayed hsame *transport of the consumed GroupUp packet, consumed RingUp packet, representation rows, direct-sum rows, tensor rows, provenance, and finite continuation ledger.*

Lean4 (definition) `BEDC.Derived.RepresentationRingUp.RepresentationRingGrothendieckClassifier`

Theorem 246.3 (RepresentationRingUp Grothendieck classifier reflexivity)**.** *Every representation packet $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ satisfies $\mathcal{K} \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}$. The row is the componentwise reflexive* hsame *datum on G, R, V, S, T, P, ρ, together with the identical finite* Cont *ledger comparison on λ.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket semantic name certificate`

Proof. Apply the classifier definition in Definition 246.2 to the pair $(\mathcal{K}, \mathcal{K})$. For the consumed GroupUp and RingUp packets use the reflexive hsame rows on G and R.

For the displayed representation endpoints, direct-sum rows, tensor rows, provenance packet, and Grothendieck classifier row, choose the entrywise reflexive hsame rows on V, S, T, P, ρ. These are exactly the component slots required by the displayed classifier list.

For the ledger slot choose the identity comparison of the finite Cont ledger λ with itself. Repacking these component rows gives $\mathcal{K} \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}$. □

Theorem 246.4 (RepresentationRingUp Grothendieck classifier symmetry)**.** *For representation packets $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ and $\mathcal{K}' = (G', R', V', S', T', P', \rho', \lambda')$, a classifier row $\mathcal{K} \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}'$ determines a classifier row $\mathcal{K}' \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}$ by reversing each displayed* hsame *row and reversing the finite* Cont *ledger comparison.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket grothendieck classifier symmetric`

Proof. Unpack the classifier row using Definition 246.2. It consists of component transports from G to G', from R to R', entrywise from V to V', from S to S', from T to T', from P to P', and from ρ to ρ', together with the finite ledger comparison from λ to λ'.

Reverse the two dependency rows by symmetry of hsame, giving rows from G' to G and from R' to R. Reverse the entrywise rows on the endpoint list, direct-sum ledger, tensor ledger, provenance packet, and classifier row in the same componentwise manner.

Reverse the finite Cont ledger comparison so that each entry of λ' is matched with the corresponding entry of λ. The reversed component family is precisely the classifier datum from $\mathcal{K}'$ to $\mathcal{K}$. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingGrothendieckClassifier symmetric`

Theorem 246.5 (RepresentationRingUp Grothendieck classifier transitivity)**.** *For representation packets* $\mathcal{K}_0 = (G_0, R_0, V_0, S_0, T_0, P_0, \rho_0, \lambda_0)$*,* $\mathcal{K}_1 = (G_1, R_1, V_1, S_1, T_1, P_1, \rho_1, \lambda_1)$*, and* $\mathcal{K}_2 = (G_2, R_2, V_2, S_2, T_2, P_2, \rho_2, \lambda_2)$*, classifier rows* $\mathcal{K}_0 \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}_1$ *and* $\mathcal{K}_1 \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}_2$ *compose to a classifier row* $\mathcal{K}_0 \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}_2$*. The composite is componentwise* hsame *transitivity on* G, R, V, S, T, P, ρ*, together with composition of the finite* Cont *ledger comparisons from* λ_0 *through* λ_1 *to* λ_2*.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket grothendieck classifier transitive`

Proof. Unpack both classifier rows by Definition 246.2. The first row supplies component transports from the subscript 0 packet to the subscript 1 packet, and the second supplies the corresponding component transports from the subscript 1 packet to the subscript 2 packet.

Compose the consumed GroupUp and RingUp rows by transitivity of hsame, obtaining rows from G_0 to G_2 and from R_0 to R_2. Compose the endpoint-list rows entry by entry, and do the same for the direct-sum ledger, tensor ledger, provenance packet, and Grothendieck classifier row.

For the continuation ledger, concatenate the comparison from λ_0 to λ_1 with the comparison from λ_1 to λ_2, and then read it as the finite entrywise comparison from λ_0 to λ_2. Repacking the composed components gives the required classifier row from $\mathcal{K}_0$ to $\mathcal{K}_2$. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingGrothendieckClassifier transitive`

246.2 The certificate

Theorem 246.6 (RepresentationRingUp carrier BHist boundary)**.** *Every public endpoint of a* $\mathsf{RepresentationRing}^{\uparrow}$ *carrier packet of Definition 246.1 is one of the displayed* BHist *rows in* $G, R, V, S, T, P, \rho, \lambda$*. In particular, every representation endpoint is scoped through the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *certificates, and no additional group carrier, ring carrier, module carrier, or host representation object is exported at the boundary.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket carrier boundary`

Proof. Project an accepted packet to the tuple $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$. The carrier definition lists all public rows in that tuple, so a public endpoint can only be the consumed GroupUp row, the consumed RingUp row, a representation endpoint from V, a direct-sum row from S, a tensor row from T, a provenance row from P, the Grothendieck classifier row ρ, or an entry of the finite ledger λ.

For a representation endpoint in V, the same projection reads its action source through G and its target algebra rows through R. The direct-sum and tensor rows are ledger faces whose endpoints are members of V, so they also remain scoped by those two consumed certificates.

Conversely, any row from the displayed tuple repacks into the public carrier surface by the constructor pattern of Definition 246.1. No external representation object is needed, because the boundary is exactly the finite BHist packet. □

Theorem 246.7 (RepresentationRingUp direct-sum tensor ledger stability)**.** *If two packets of Definition 246.1 are related by the classifier of Definition 246.2, then the direct-sum faces and tensor-product faces of the first packet transport to the corresponding faces of the second packet through* hsame *and the finite* Cont *ledger comparison. The transported packet preserves the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *sources and the Grothendieck classifier row.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket direct sum tensor ledger stability`

Proof. Unpack the classifier between $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ and $\mathcal{K}' = (G', R', V', S', T', P', \rho', \lambda')$. It supplies transported GroupUp and RingUp dependency rows, entrywise endpoint transport between V and V', entrywise ledger transport between S and S' and between T and T', provenance transport, classifier-row transport, and a finite comparison between λ and λ'.

Read a direct-sum face from S. Its endpoints are displayed members of V, so the endpoint comparison in the classifier carries them to the matching members of V'. The ledger comparison then carries the Cont row witnessing the direct-sum face to the corresponding entry of S'.

The tensor-product case is identical except that the ledger entry is read from T. The transported endpoints still read their action source through G' and their target rows through R', because the classifier transported the consumed GroupUp and RingUp packets at the same time.

Finally transport ρ to ρ' and P to P'. Repacking these rows gives a representation-ring packet with stable direct-sum and tensor faces over the transported finite ledger. □

Theorem 246.8 (RepresentationRingUp NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *obligation surface is the package consisting of the carrier packet of Definition 246.1, the Grothendieck ledger classifier of Definition 246.2, the carrier boundary of Definition 246.6, the direct-sum and tensor stability row of Definition 246.7, exactness of the finite* Cont *and* Pkg *ledger rows, and the consumer boundary used by* $\mathsf{InducedRep}^{\uparrow}$ *through* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$*. The consumer may read the consumed GroupUp packet, consumed RingUp packet, representation endpoints, direct-sum faces, tensor faces, Grothendieck classifier row, provenance rows, and finite ledger rows, but it receives no additional induction functor, restriction functor, character row, or host-level representation category.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket namecert obligation surface`

Proof. Start with the carrier and classifier definitions. The carrier supplies the finite row family, while the classifier specifies componentwise hsame transport for each dependency, endpoint list, operation face, provenance packet, classifier row, and ledger entry.

Apply the carrier boundary theorem to identify the public endpoints. This shows that every exported representation-ring row is already one of the displayed BHist rows and that the representation endpoints remain scoped by the consumed GroupUp and RingUp certificates.

Next apply the direct-sum and tensor stability theorem. It transports the two operation faces through the same endpoint comparisons and the same finite Cont ledger comparison, so the Grothendieck classifier row remains attached to the transported packet.

Exactness follows by inspecting the displayed rows. The only provenance rows are the Pkg rows in P, and the only continuation rows are the entries of λ together with the displayed direct-sum and tensor faces. Thus every public row is accounted for by projection from the packet.

For the downstream boundary, an $\mathsf{InducedRep}^{\uparrow}$ consumer reads precisely this public surface: representation endpoints together with direct-sum, tensor, classifier, provenance, and dependency rows. Induction, restriction, Frobenius reciprocity, and character-theoretic data are not fields of $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ and must be supplied by their own certificates. □

Theorem 246.9 (RepresentationRingUp classifier ledger exhaustion)**.** *Every accepted Grothendieck ledger classifier row for* $\mathsf{RepresentationRing}^{\uparrow}$ *is exhausted by the finite rows displayed in Definition 246.2. Concretely, for packets*

$$\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda) \qquad \textit{and} \qquad \mathcal{K}' = (G', R', V', S', T', P', \rho', \lambda'),$$

an accepted classifier between them consists only of BHist *direct-sum rows from* S *and* S'*, tensor-product rows from* T *and* T'*, provenance rows from* P *and* P'*,* hsame *transport rows for the consumed packets, endpoints, operation ledgers, provenance packets, and classifier rows, and the finite* Cont *comparison between* λ *and* λ'*. There is no further classifier row outside that displayed ledger.*

Proof. Project the accepted classifier to the seven entries listed in Definition 246.2. Each entry is either an hsame transport row or a finite Cont comparison row, and the operation entries are exactly the displayed direct-sum and tensor-product ledgers.

The carrier boundary theorem identifies every public representation-ring endpoint as one of the rows of Definition 246.1. Hence any classifier row with public effect must land on one of G, R, V, S, T, P, ρ, or λ.

Apply the obligation-surface theorem to the accepted NameCert packet. It permits no additional host representation category, quotient object, character row, or functorial equality row. The seven displayed classifier entries therefore exhaust the accepted Grothendieck ledger classifier. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket classifier ledger exhaustion`

Theorem 246.10 (RepresentationRingUp tensor-product descent)**.** *Let* $\mathcal{K}$ *and* $\mathcal{K}'$ *be packets of Definition 246.1 related by the classifier of Definition 246.2. If a tensor-product representation packet is read from two endpoints of* V *together with a tensor ledger row of* T*, then the transported endpoints in* V' *and the corresponding tensor ledger row of* T' *form a tensor-product representation packet related to the first one by the same classifier. Thus tensor products descend along the finite representation-ring classifier relation.*

Proof. Unpack the classifier between $\mathcal{K}$ and $\mathcal{K}'$. It supplies entrywise hsame transport for the representation endpoints and entrywise transport between the tensor ledgers T and T'.

Read the chosen tensor-product packet from its two endpoint rows in V and its tensor ledger row in T. The endpoint transports carry those two rows to the matching endpoint rows in V', and the tensor-ledger transport carries the operation row to the corresponding member of T'.

Apply the direct-sum tensor ledger stability theorem to preserve the consumed GroupUp and RingUp sources, the provenance row, the Grothendieck classifier row, and the finite Cont ledger comparison. Repacking the transported rows gives the tensor-product packet over $\mathcal{K}'$, related by the same displayed classifier data. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket tensor product descent`

Theorem 246.11 (RepresentationRingUp induced consumer boundary)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *consumes exactly the finite boundary rows consisting of the group source row, the subgroup boundary row supplied by its own group-side certificate, the representation packet of Definition 246.1, the induction ledger rows supplied by the consumer certificate, and the Grothendieck classifier rows of Definition 246.2. It does not receive a host quotient of representation categories, an external representation-equality oracle, or any equality relation beyond the displayed* hsame *and* Cont *rows.*

Proof. Start from the public representation-ring packet. Its exposed rows are the consumed GroupUp and RingUp sources, representation endpoints, direct-sum and tensor ledgers, provenance rows, classifier row, and continuation ledger.

An $\mathsf{InducedRep}^{\uparrow}$ consumer adds its own subgroup boundary and induction ledger, both as finite certificate rows. The representation-ring side contributes only the representation packet and the Grothendieck classifier needed to transport that packet across accepted classifier-related inputs.

The obligation-surface theorem excludes any additional induction functor, restriction functor, character row, or host representation category from $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$. The carrier boundary and classifier exhaustion results then leave exactly the named rows, with equality mediated only by hsame and finite Cont comparison. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket induced consumer boundary`

Theorem 246.12 (RepresentationRingUp InducedRep consumer boundary)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *may read only the finite* BHist *rows displayed in the carrier packet* $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ *of Definition 246.1: the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row* G*, the consumed* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *row* R*, the representation endpoint list* V*, the direct-sum ledger* S*, the tensor-product ledger* T*, the* Pkg *provenance packet* P*, the Grothendieck classifier row* ρ*, and the finite* Cont *ledger* λ*. Any induction, restriction, Frobenius, subgroup-inclusion, or character-theoretic row required by* $\mathsf{InducedRep}^{\uparrow}$ *lies outside this RepresentationRingUp boundary unless it is separately supplied by the* $\mathsf{InducedRep}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket consumer boundary`

Proof. Project the consumer request to the carrier packet $\mathcal{K}$. The carrier definition lists exactly the rows $G, R, V, S, T, P, \rho, \lambda$, and each representation endpoint in V reads its action source through G and its algebra target through R.

The operation rows are also already named. A direct-sum request lands in the finite Cont ledger S, a tensor-product request lands in the finite Cont ledger T, and every public continuation read is an entry of λ. Provenance is read through the displayed Pkg packet P.

Repack these projections as the representation-ring part of an $\mathsf{InducedRep}^{\uparrow}$ input. The repacking gives a concrete BEDC source boundary: the downstream packet receives the rows named above and no host representation category, induction functor, restriction functor, Frobenius adjunction, or character row from $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$. □

Theorem 246.13 (RepresentationRingUp classifier exactness)**.** *For packets* $\mathcal{K}$ *and* $\mathcal{K}'$ *of Definition 246.1, the Grothendieck ledger classifier* $\mathcal{K} \sim^{\mathrm{gr}}_{\mathsf{RepresentationRing}^{\uparrow}} \mathcal{K}'$ *of Definition 246.2 is exact for the public carrier: its projections are precisely the* hsame *rows for the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *packets, the entrywise representation rows, the direct-sum and tensor* Cont *ledgers, the* Pkg *provenance packet, the Grothendieck row* ρ*, and the finite ledger comparison for* λ*. Conversely, those displayed projections repack into the classifier without using host equality of representation categories.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket classifier exactness`

Proof. Unpack the classifier between $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ and $\mathcal{K}' = (G', R', V', S', T', P', \rho', \lambda')$. Its fields are the two dependency comparisons, endpoint-list comparison, operation-ledger comparisons, provenance comparison, Grothendieck-row comparison, and finite ledger comparison.

Each field is a projection of a public row of the carrier packet. The GroupUp and RingUp comparisons account for the dependency rows; the endpoint and operation comparisons account for V, S, T; the provenance comparison accounts for P; the Grothendieck comparison accounts for ρ; and the finite ledger comparison accounts for λ.

For the converse direction, begin with exactly those displayed projections. Place them in the same ordered slots required by Definition 246.2. This constructs the classifier row by repacking finite BEDC data, so no source-level equivalence of representation categories is invoked. □

Theorem 246.14 (RepresentationRingUp operation ledger coverage)**.** *Every public operation row of a* $\mathsf{RepresentationRing}^{\uparrow}$ *carrier packet is covered by the finite ledgers S, T, and λ of Definition 246.1. Direct-sum operations are exactly the rows of S, tensor-product operations are exactly the rows of T, their endpoints are members of the representation endpoint list V, and their dependency and provenance reads factor through the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row G, the consumed* $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ *row R, and the* Pkg *packet P.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket operation ledger coverage`

Proof. Project a public operation row from the packet $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$. The carrier definition has only two operation ledgers: S for direct sums and T for tensor products. Thus the row is read from one of those finite Cont ledgers.

In the direct-sum case, the endpoints of the selected row are displayed members of V. Their group-action source is obtained by projecting G, and their additive target rows are obtained by projecting R. The relevant continuation entry is also listed in λ, and its package provenance is listed in P.

The tensor-product case is parallel. The selected row lies in T, its endpoints are members of V, its dependency rows are again G and R, and its public continuation and package reads are entries of λ and P. Repacking these projections covers every operation row and leaves no unlisted operation at the boundary. □

Theorem 246.15 (RepresentationRingUp finite BHist source exhaustion)**.** *The finite* BHist *source of a* $\mathsf{RepresentationRing}^{\uparrow}$ *public packet is exhausted by the displayed tuple $G, R, V, S, T, P, \rho, \lambda$. Every public carrier, classifier, operation, ledger, provenance, and downstream-consumer row is a projection of that tuple or a row named by one of its finite* Cont *ledgers. No ambient group object, ambient ring object, module category, character table, or representation category is a separate source row of the public certificate.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket finite source exhaustion`

Proof. Inspect the carrier packet. Its dependency sources are exactly G and R, its representation endpoints are exactly the finite list V, and its operation sources are exactly the finite ledgers S and T. The classifier source is the row ρ, provenance is the packet P, and public continuation data is the ledger λ.

Now consider any public read. A carrier read is one of those fields. A classifier read is a projection of Definition 246.2. An operation read is covered by Definition 246.14. A consumer read for $\mathsf{InducedRep}^{\uparrow}$ is covered by Definition 246.12.

These cases exhaust the finite packet. Repacking the selected projections returns a row of the same BHist source, and there is no remaining slot where an ambient host object or representation category could enter as a public source. □

Theorem 246.16 (RepresentationRingUp public NameCert surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *surface consists exactly of the* BHist *carrier packet of Definition 246.1, the Grothendieck ledger classifier of Definition 246.2, the consumer boundary of Definition 246.12, the classifier exactness of Definition 246.13, the operation-ledger coverage of Definition 246.14, and the finite source exhaustion of Definition 246.15. This surface exports the GroupUp and RingUp dependency rows, finite representation endpoint rows, finite* Cont *ledgers,* Pkg *provenance, and the Grothendieck classifier row, and it exports nothing outside those rows.*

Proof. Begin with the carrier packet and classifier. They provide the finite BEDC rows $G, R, V, S, T, P, \rho, \lambda$ and the componentwise comparisons between two such packets. The classifier exactness theorem shows that these comparisons are precisely the public classifier fields.

Next use the operation-ledger coverage theorem. It accounts for all direct sum and tensor product rows through S, T, λ, and P, with endpoints in V and dependencies through G and R.

The consumer boundary theorem supplies the downstream side: an $\mathsf{InducedRep}^{\uparrow}$ packet may consume the same finite surface but receives no induction, restriction, Frobenius, subgroup-inclusion, or character data from the representation-ring certificate.

Finally apply finite source exhaustion. Every public row named by the certificate is already a projection of the displayed BHist packet or of one of its finite Cont ledgers. Repacking those projections gives the public $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket public namecert surface`

Definition 246.17 (RepresentationRingUp obligation row inventory)**.** *The* $\mathsf{RepresentationRing}^{\uparrow}$ *obligation row inventory is the finite list of public rows that a* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *packet must expose:*

1. *the carrier packet row of Definition 246.1, including the GroupUp packet, RingUp packet, finite representation endpoints, direct-sum ledger, tensor-product ledger, provenance packet, and finite* Cont *ledger;*
2. *the Grothendieck classifier row of Definition 246.2;*
3. *the direct-sum ledger row, consisting of source endpoint rows, target endpoint rows, and the finite* Cont *records witnessing the displayed direct sum construction;*
4. *the tensor-product ledger row, consisting of source endpoint rows, target endpoint rows, and the finite* Cont *records witnessing the displayed tensor construction;*
5. *the provenance packet row, namely the* Pkg *row that records that each operation endpoint is read from the displayed GroupUp and RingUp dependencies;*
6. *the finite* Cont *ledger row for all carrier, classifier, direct-sum, tensor-product, and provenance continuations in the public packet;*
7. *the consumer readback rows exposed to* $\mathsf{InducedRep}^{\uparrow}$ *and no other downstream consumer.*

Lean4 (definition) `BEDC.Derived.RepresentationRingUp.RepresentationRingObligationRowInventory`

Theorem 246.18 (RepresentationRingUp obligation inventory soundness)**.** *Every row listed in Definition 246.17 is supplied by the displayed* $\mathsf{RepresentationRing}^{\uparrow}$ *public packet: carrier rows come from Definition 246.6, classifier rows from Definition 246.13, semantic certificate rows from Definition 246.8, direct-sum and tensor rows from Definition 246.14, provenance and finite* Cont *rows from Definition 246.15, and consumer readback rows from Definition 246.12.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket obligation inventory sound`

Proof. Start with the carrier packet. The carrier boundary theorem projects the finite GroupUp packet, RingUp packet, representation endpoint rows, operation ledgers, provenance packet, and finite Cont records from the displayed BHist representation packet.

Next read the classifier and semantic certificate rows. Classifier exactness identifies the Grothendieck classifier with the finite ledger comparison, and the obligation-surface theorem places that classifier inside the public $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ surface.

The operation rows are covered by the operation-ledger theorem. It handles the direct-sum ledger and tensor-product ledger by reading their source endpoints, target endpoints, provenance packet, and finite Cont records from the same packet.

Finally apply finite source exhaustion and the consumer boundary theorem. Finite source exhaustion supplies the remaining provenance and Cont rows, and the consumer boundary theorem supplies exactly the $\mathsf{InducedRep}^{\uparrow}$ readback rows named by the inventory. □

Theorem 246.19 (RepresentationRingUp obligation inventory exhaustion)**.** *Every public* $\mathsf{RepresentationRing}^{\uparrow}$ *obligation projects to one of the rows in Definition 246.17. In particular, the public surface never reads a host representation category, an ambient Grothendieck quotient, character theory, induction-restriction calculus, or any operation not displayed by the finite carrier packet.*

Lean4 ✓ `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket obligation inventory exhaustive`

Proof. Let a public $\mathsf{RepresentationRing}^{\uparrow}$ obligation be given. By the public NameCert surface theorem, it is a carrier, classifier, operation, provenance, finite Cont, or consumer row of the displayed packet.

Carrier obligations project to the carrier packet row. Classifier obligations project to the Grothendieck classifier row. Operation obligations are either direct-sum rows or tensor-product rows, hence project to the corresponding ledger row.

Provenance and finite-continuation obligations project to the provenance packet row or to the finite Cont ledger row. Consumer obligations project to the $\mathsf{InducedRep}^{\uparrow}$ readback row and receive no additional downstream surface.

These cases exhaust the public packet by finite source exhaustion. Since the exhaustion theorem has no branch for a host representation category, Grothendieck quotient object, character row, or induction-restriction calculus, none of those structures is read by a public $\mathsf{RepresentationRing}^{\uparrow}$ obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$**.**

Constructive story: A representation ring is generated as a finite packet $\mathcal{K} = (G, R, V, S, T, P, \rho, \lambda)$ whose G row consumes $\mathsf{Group}^{\uparrow}$, whose R row consumes $\mathsf{Ring}^{\uparrow}$, whose V row lists representation endpoints, and whose S and T ledgers display direct-sum and tensor-product faces. The carrier accepts exactly the consumed group and ring packets, the finite representation endpoint list, the operation ledgers, the provenance packet, the Grothendieck classifier row, and the finite Cont ledger; it excludes a host representation category, module category, character table, induction or restriction functor, and ambient Grothendieck quotient object. Two packets classify together when every displayed component is transported

by hsame and the continuation ledger is compared entry by entry, so Grothendieck equality is a finite ledger balance among direct-sum rows rather than an external quotient. The source specification is the packet of dependency rows and representation endpoints, the pattern records direct-sum and tensor ledger coverage, classifier exactness, consumer boundary, and finite source exhaustion, and the ledger keeps operation endpoints, provenance, classifier rows, and continuation entries visible. The five naming-certificate obligations are discharged by the reflexive packet classifier for inhabitance and reflexivity, componentwise reversal for symmetry, componentwise composition for transitivity, and direct-sum/tensor ledger stability for closure under classifier transport. A history h lies in the signature gap of a $\mathsf{RepresentationRing}^{\uparrow}$ token p when the source domain accepts the finite packet, the signature relation pairs it with an answer packet and evidence ledger, and token introduction names that answer as p; concretely, the consumed group and ring rows, representation endpoints, operation ledgers, Grothendieck balance row, and provenance of h are the packet encoded by p. The chapter registers this finite Grothendieck-ledger construction under $\mathsf{RepresentationRing}^{\uparrow}$; it exports the rows needed by induced-representation consumers without adding character theory, induction-restriction calculus, a standard bridge, or a host representation-ring primitive.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature scope is the displayed GroupUp packet, RingUp packet, representation endpoints, direct-sum and tensor ledgers, Grothendieck classifier, public NameCert surface, finite BHist source exhaustion, consumer obligations, induced-representation readback rows, and obligation inventory soundness and exhaustion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RepresentationRingUp.RepresentationRingBHistRepresentationPacket\_public\_namecert\_sur`

Bridge status: none

Not claimed: Full representation theory, character theory, induction-restriction calculus, and standard bridge data are not claimed by this chapter.

Upgrade path: No further paper theory-axis upgrade is recorded for this chapter; bridgeChecked remains a separate formal-axis target.

Chapter 247

A Concrete Naming Certificate for $\mathsf{InducedRep}^{\uparrow}$

The induced-representation interface $\mathsf{InducedRep}^{\uparrow}$ records a finite induction–restriction packet with consumed $\mathsf{Subgroup}^{\uparrow}$ and $\mathsf{RepresentationRing}^{\uparrow}$ certificate rows, Frobenius rows, classifier rows, continuation ledgers, and package provenance. The induction-restriction packet consumes the subgroup carrier of chapter 80 and the character-ledger surface of chapter 246 before the local Frobenius rows are named.

247.1 Carrier definition

Definition 247.1 (InducedRepUp BHist carrier packet)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda).$$

The row S is the consumed $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *packet naming the subgroup inclusion. The row R is the consumed* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *packet naming the source and target representation-ring rows. The rows i and r are the displayed induction and restriction endpoints, each scoped by S for the subgroup inclusion and by R for the representation-ring source and target rows. The row F is the finite Frobenius reciprocity ledger whose entries compare the displayed induction and restriction endpoints without exporting a host adjunction.*

The rows η and ϵ are the Frobenius unit and counit rows in the same finite ledger. The row ρ is the Pkg *provenance packet recording exactly the consumed package rows from S and R together with the package rows attached to i, r, F, η, and ϵ. The finite row λ is the* Cont *ledger containing exactly the consumed SubgroupUp rows, consumed RepresentationRingUp rows, induction and restriction endpoint rows, Frobenius rows, provenance rows, and public continuation rows exposed by the packet.*

Definition 247.2 (InducedRepUp induction–restriction packet)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction packet is a finite* BHist *packet*

$$\mathcal{P}_{\mathsf{InducedRep}^{\uparrow}} = (S_{\leq}, R_{\mathrm{rep}}, I_{\uparrow}, R_{\downarrow}, \Phi, H_{\leq}, H_{\mathrm{rep}}, H_{\uparrow}, H_{\downarrow}, C_{\mathrm{src}}, C_{\mathrm{fr}}, \Pi).$$

The row $S_{\leq}$ is the consumed subgroup-inclusion row supplied by $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$*. The row R_{rep} is the consumed representation-ring row supplied by* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$*. The rows $I_{\uparrow}$ and $R_{\downarrow}$ are the displayed induction and restriction rows over that same subgroup source and representation-ring source.*

The row Φ is the finite Frobenius comparison row. It compares the displayed $I_\uparrow$ and $R_\downarrow$ rows as BEDC data and does not export a host adjunction. The classifier rows $H_\leq, H_{\mathrm{rep}}, H_\uparrow, H_\downarrow$ are the corresponding hsame *rows for subgroup inclusion, representation-ring transport, induction transport, and restriction transport. The ledgers C_{src} and C_{fr} are finite* Cont *ledgers for the source-certificate rows and the Frobenius comparison rows. The row Π is the* Pkg *provenance row recording exactly the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ *package, the consumed* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^\uparrow}$ *package, and the displayed induction, restriction, classifier, Frobenius, and continuation rows.*

Definition 247.3 (InducedRepUp restriction-induction classifier)**.** *For two carrier packets*

$$\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda) \qquad \text{and} \qquad \mathcal{I}' = (S', R', i', r', F', \eta', \epsilon', \rho', \lambda'),$$

the restriction-induction classifier $\mathcal{I} \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^\uparrow} \mathcal{I}'$ is the finite componentwise transport datum consisting of:

1. *an* hsame *transport row between the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ *packets S and S';*
2. *an* hsame *transport row between the consumed* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^\uparrow}$ *packets R and R';*
3. *entrywise* hsame *transport of the subgroup-inclusion rows read from S and the representation-ring rows read from R;*
4. hsame *transport rows between the induction endpoints i and i' and between the restriction endpoints r and r';*
5. *entrywise* hsame *transport of the Frobenius ledger rows F, η, ϵ to F', η', ϵ';*
6. *an* hsame *transport row between the provenance packets ρ and ρ';*
7. *a finite* Cont *ledger comparison between λ and λ', entry by entry.*

This classifier transports only displayed BEDC rows: subgroup inclusion rows, representation-ring rows, induction and restriction endpoints, Frobenius unit and counit rows, provenance, and continuation entries. It explicitly does not require equality of host induction functors, host restriction functors, representation categories, or adjunction data outside the finite packet.

247.2 The certificate

Theorem 247.4 (InducedRepUp subgroup source boundary)**.** *For every* $\mathsf{InducedRep}^\uparrow$ *packet of Definition 247.2, the subgroup source visible to downstream consumers is exactly the finite* BHist *row $S_\leq$ consumed from* $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$*, together with its* hsame *classifier row $H_\leq$, its* Cont *entries inside C_{src}, and its* Pkg *provenance inside Π. No additional subgroup object, ambient group, inclusion witness, or host restriction datum is part of the public source boundary.*

Lean4 ✓ `BEDC.Derived.InducedRepUp.InducedRepSubgroupSource boundary`

Proof. The traditional target is the subgroup inclusion along which induction and restriction are displayed. The BEDC source is not an ambient group object but the consumed $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ row $S_\leq$ in the packet.

Project the carrier. The only subgroup-source row in $\mathcal{P}_{\mathsf{InducedRep}^\uparrow}$ is $S_\leq$, and the matching classifier entry is $H_\leq$. Stability is therefore componentwise hsame transport through $H_\leq$ together with the source continuation entries recorded in C_{src}.

The ledger and exactness clauses are read from Π and C_{src}. Every exported subgroup-source entry occurs in those rows, and every such entry is named by the packet. Since no StandardBridge row is carried, the standard bridge is absent at this boundary. □

Theorem 247.5 (InducedRepUp representation-ring action boundary)**.** *For every* $\mathsf{InducedRep}^{\uparrow}$ *packet of Definition 247.2, the representation-ring action visible at the induction and restriction interface is exactly the finite* BHist *row* R_{rep} *consumed from* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$*, the displayed rows* $I_{\uparrow}$ *and* $R_{\downarrow}$*, the* hsame *classifier rows* $H_{\mathrm{rep}}, H_{\uparrow}, H_{\downarrow}$*, the continuation rows in* C_{src} *and* C_{fr}*, and the* Pkg *provenance row* Π*.*

Proof. The traditional target is an action of representation rings on induction and restriction. The BEDC source is the consumed $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ row R_{rep}, and the carrier exposes only the two displayed action endpoints $I_{\uparrow}$ and $R_{\downarrow}$.

Classifier stability is componentwise. The row H_{rep} transports the representation-ring source, while $H_{\uparrow}$ and $H_{\downarrow}$ transport the displayed induction and restriction rows. The Cont ledgers record the public continuation entries attached to those rows and to the Frobenius comparison.

Exactness is obtained by checking the provenance row Π. Any action row available to a consumer must be one of $R_{\mathrm{rep}}, I_{\uparrow}, R_{\downarrow}$ or a continuation entry named by the displayed ledgers. No host representation category, functor equality, or standard bridge is exported. □

Theorem 247.6 (InducedRepUp Frobenius reciprocity classifier)**.** *The Frobenius reciprocity data of an* $\mathsf{InducedRep}^{\uparrow}$ *packet is classified by the finite comparison row* Φ*, the endpoint classifier rows* $H_{\uparrow}$ *and* $H_{\downarrow}$*, the source classifier rows* $H_{\leq}$ *and* H_{rep}*, the* Cont *ledger* C_{fr}*, and the* Pkg *provenance row* Π*. Thus* hsame *equivalence of Frobenius rows is a BEDC comparison between displayed induction and restriction rows, not equality of a host adjunction.*

Proof. The traditional target is Frobenius reciprocity for induction and restriction. The BEDC carrier names this target by the row Φ, whose endpoints are the displayed rows $I_{\uparrow}$ and $R_{\downarrow}$ over the source rows $S_{\leq}$ and R_{rep}.

The classifier is the tuple of hsame rows. Source transport uses $H_{\leq}$ and H_{rep}; endpoint transport uses $H_{\uparrow}$ and $H_{\downarrow}$. Stability of the Frobenius comparison is the finite Cont transport recorded in C_{fr}.

The ledger clause is exactly the combination of C_{fr} and Π. Exactness follows because every Frobenius entry consumed by the public interface is named by Φ or by that ledger. The packet carries no $\mathsf{StandardBridge}$ row, so the standard bridge is absent and the classifier remains inside BHist. □

Theorem 247.7 (InducedRepUp public obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ *obligation surface consists of the* BHist *packet of Definition 247.2, the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *rows, the displayed induction and restriction rows, the Frobenius comparison row, all listed* hsame *classifier rows, the* Cont *ledgers* C_{src} *and* C_{fr}*, and the* Pkg *provenance row* Π*. No obligation outside those displayed rows is public.*

Proof. The traditional target is the public certificate for an induced representation interface. The BEDC source is the pair of consumed certificates $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$, read through $S_{\leq}$ and R_{rep}.

The carrier is the finite packet $\mathcal{P}_{\mathsf{InducedRep}^{\uparrow}}$. Its classifier rows are $H_{\leq}, H_{\mathrm{rep}}, H_{\uparrow}, H_{\downarrow}$, and stability is the corresponding componentwise hsame transport together with the two finite Cont ledgers.

The ledger and exactness conditions are public precisely because they are recorded in C_{src}, C_{fr}, and Π. Every row in the surface is a projection of the packet, and every projection repacks into the same surface. With no $\mathsf{StandardBridge}$ row present, the standard bridge is absent. □

Theorem 247.8 (InducedRepUp consumer boundary)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *consumer may inspect exactly the source rows* $S_{\leq}$ *and* R_{rep}*, the displayed induction and restriction rows* $I_{\uparrow}$ *and* $R_{\downarrow}$*, the Frobenius row* Φ*, the* hsame *classifier rows* $H_{\leq}, H_{\mathrm{rep}}, H_{\uparrow}, H_{\downarrow}$*, the* Cont *ledgers* C_{src} *and* C_{fr}*, and the*

Pkg *row* Π. *Any use of* $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ *or* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^\uparrow}$ *by that consumer factors through those rows.*

Proof. The traditional target is a downstream use of induction, restriction, and Frobenius reciprocity. The BEDC source data are the two consumed certificates, and the carrier presents them only through $S_\leq$ and R_{rep}.

For classifier stability, the consumer must use the displayed hsame rows. Subgroup and representation-ring requests pass through $H_\leq$ and H_{rep}; induction and restriction requests pass through $H_\uparrow$ and $H_\downarrow$; Frobenius requests pass through Φ and C_{fr}.

The ledger is finite and explicit: C_{src}, C_{fr}, and Π list exactly what the consumer may read. Exactness is the converse projection from those listed rows back to the carrier. No host carrier, external representation category, adjunction object, or standard bridge is available beyond that boundary. □

Theorem 247.9 (InducedRepUp Frobenius ledger boundary)**.** *Every Frobenius row exported by an* $\mathsf{InducedRep}^\uparrow$ *carrier of Definition 247.1 is one of the displayed finite rows* $F, \eta, \epsilon, \lambda$*. Its endpoints are the displayed induction row* i *and restriction row* r*, scoped through the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^\uparrow}$ *packets. No host adjunction, host functor equality, naturality square, or external representation category is exported at the Frobenius boundary.*

Lean4 ✓ `BEDC.Derived.InducedRepUp.InducedRepFrobeniusLedger boundary`

Proof. Project an accepted packet to $\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda)$. The carrier definition lists the Frobenius part as the finite ledger F together with the unit row η, the counit row ϵ, and the public continuation entries in λ.

The displayed endpoints of those rows are i and r. Their subgroup source is read through S, and their representation-ring source and target rows are read through R. The source certificates are precisely the SubgroupUp and RepresentationRingUp NameCert rows cited in the packet.

Exactness is the reverse projection. A row outside $F, \eta, \epsilon, \lambda$ is not a Frobenius row of the carrier, and a row outside S, R, i, r, ρ supplies no public endpoint or provenance for that Frobenius display. Thus the Frobenius boundary is exactly the finite ledger shown in the packet. □

Theorem 247.10 (InducedRepUp NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{InducedRep}^\uparrow}$ *obligation surface is the package consisting of the carrier packet of Definition 247.1, the restriction-induction classifier of Definition 247.3, the Frobenius ledger boundary of Definition 247.9, componentwise* hsame *and* Cont *stability for each displayed row, exactness of the finite provenance and continuation ledgers, and no standard bridge. The surface exports no row outside the carrier, classifier, and provenance ledger.*

Lean4 ✓ `BEDC.Derived.InducedRepUp.InducedRepCarrierPacket namecert obligation surface`
Lean4 ✓ `BEDC.Derived.InducedRepUp.InducedRepBHistCarrier namecert obligation surface`

Proof. Start with the carrier and classifier. The carrier supplies the finite row family $S, R, i, r, F, \eta, \epsilon, \rho, \lambda$, while the classifier supplies componentwise hsame transport for the consumed certificates, subgroup inclusion rows, representation-ring rows, induction and restriction endpoints, Frobenius rows, provenance packet, and continuation ledger.

The ten-field reading is therefore fixed. The traditional target is the restriction-induction interface with Frobenius reciprocity; the BEDC sources are $\mathsf{NameCert}_{\mathsf{Subgroup}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{RepresentationRing}^\uparrow}$; the carrier is the displayed finite BHist packet; the classifier is the restricted componentwise relation; stability is the listed hsame and Cont transport; the ledger is λ together with the Frobenius

rows; the core theorem list is this surface and the Frobenius ledger boundary; exactness is projection from the displayed rows; the standard bridge is none; and the status is a paper-side obligation surface for $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$.

For stability, transport each component of a packet along the corresponding classifier row. The SubgroupUp source certificate transports the subgroup inclusion rows, the RepresentationRingUp source certificate transports the representation-ring rows, and the endpoint and Frobenius rows are transported by their displayed hsame comparisons and the finite Cont comparison.

Exactness follows because every exported row is named in the carrier, classifier, or provenance ledger. Conversely, each named row repacks into the obligation surface by the displayed packet, so no additional host functor, host category, adjunction, character table, or bridge row is part of the certificate. □

Theorem 247.11 (InducedRepUp Subgroup and RepresentationRing consumer boundary)**.** *An* $\mathsf{InducedRep}^{\uparrow}$ *consumer of the source certificates may read exactly the SubgroupUp rows exposed by* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *and the representation-ring rows exposed by* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$*, together with the induction, restriction, Frobenius, provenance, and continuation rows displayed in Definition 247.1. It may not read a subgroup carrier, representation ring, functor, category, or Frobenius adjunction row unless that row is present in the carrier, classifier, or provenance ledger of the* $\mathsf{InducedRep}^{\uparrow}$ *packet.*

Proof. Project the consumer request through the carrier packet. A SubgroupUp request lands in S, and a RepresentationRingUp request lands in R. The source certificates named by those two rows supply their own public carrier, classifier, stability, exactness, and ledger boundaries.

Next project the $\mathsf{InducedRep}^{\uparrow}$-specific request. The induction endpoint is i, the restriction endpoint is r, the Frobenius rows are F, η, ϵ, the provenance packet is ρ, and the continuation ledger is λ. Transport between two such consumer surfaces is exactly the componentwise hsame and Cont transport listed in the restriction-induction classifier.

There are no remaining exported rows. Any row not found in the two source certificates or in $i, r, F, \eta, \epsilon, \rho, \lambda$ is outside the consumer boundary, so the downstream interface receives a concrete BEDC packet rather than equality of host-level induction and restriction constructions. □

Definition 247.12 (InducedRepUp stability obligation packet)**.** *For carrier packets*

$$\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda) \qquad \text{and} \qquad \mathcal{I}' = (S', R', i', r', F', \eta', \epsilon', \rho', \lambda'),$$

the $\mathsf{InducedRep}^{\uparrow}$ *stability obligation packet is the finite datum consisting of the two displayed carriers, the componentwise* hsame *rows from the restriction-induction classifier of Definition 247.3, and the finite* Cont *comparison ledger from* λ *to* λ'*. Its fields are:*

1. hsame *rows for the consumed certificate rows* S *and* R*;*
2. hsame *rows for the induction and restriction endpoints* i *and* r*;*
3. *entrywise* hsame *rows for the Frobenius display* F, η, ϵ*;*
4. *an* hsame *row for the provenance packet* ρ*;*
5. *the finite* Cont *comparison ledger relating the source, endpoint, Frobenius, provenance, and public continuation entries recorded by* λ*.*

The packet has no field for equality of host categories, host induction functors, host restriction functors, functorial naturality, or adjunction objects outside the displayed BHist *rows.*

Theorem 247.13 (InducedRepUp classifier stability)**.** *The restriction-induction classifier of Definition 247.3 transports every carrier component of Definition 247.1. For any two packets $\mathcal{I}$ and $\mathcal{I}'$ related by that classifier, the transport repacks as the stability obligation packet of Definition 247.12 and preserves the public* $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ *obligation surface of Definition 247.10.*

Proof. Project the two carriers as $\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda)$ and $\mathcal{I}' = (S', R', i', r', F', \eta', \epsilon', \rho', \lambda')$. The classifier is already componentwise: it gives the hsame rows for the consumed SubgroupUp and RepresentationRingUp packets, the displayed subgroup and representation-ring rows read from them, the induction and restriction endpoints, the Frobenius rows, and the provenance packet.

It remains to repack the continuation part. The same classifier includes the finite Cont comparison between λ and λ', entry by entry, so the source, endpoint, Frobenius, provenance, and public continuation reads are transported inside the displayed ledger.

The obligation surface in Definition 247.10 asks for no more data than those projected rows. Repacking the projected hsame rows and the finite Cont comparison gives the stability obligation packet, and the surface is preserved because no host category, functor equality, adjunction row, or bridge row is part of the transported data. □

Theorem 247.14 (InducedRepUp continuation ledger closure)**.** *The source reads, induction and restriction endpoint reads, Frobenius reads, provenance reads, and public continuation reads of an* $\mathsf{InducedRep}^{\uparrow}$ *carrier close under finite* Cont *composition inside the ledger λ of Definition 247.1. Consequently every finite continuation needed by the obligation surface is again a row recorded by λ.*

↪ variant `BEDC.Derived.InducedRepUp.InducedRepBHistCarrier namecert obligation surface`

Proof. Project the carrier to the finite packet $\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda)$. By definition, λ contains the consumed SubgroupUp rows, consumed RepresentationRingUp rows, induction and restriction endpoint rows, Frobenius rows, provenance rows, and public continuation rows exposed by the packet.

Now compose only rows already listed in λ. A source read followed by an endpoint read, an endpoint read followed by a Frobenius read, or a Frobenius read followed by a public continuation read is still a finite read through the same displayed ledger, because the ledger is the complete Cont record for the carrier.

Thus the closure statement is a projection-and-repacking argument. The projection chooses the listed rows of λ, and the repacking records their finite composite as another continuation row of the same public ledger. Nothing in the argument introduces a host functor, host category, or adjunction object outside the carrier. □

Theorem 247.15 (InducedRepUp package provenance exactness)**.** *Every row exported by the* $\mathsf{InducedRep}^{\uparrow}$ *package provenance surface is exactly one of the rows recorded by ρ or λ in Definition 247.1: the consumed SubgroupUp rows, the consumed RepresentationRingUp rows, the induction row, the restriction row, and the Frobenius rows F, η, ϵ. No host category, host functor equality, or adjunction row is exported by the provenance surface.*

Proof. Project the package row ρ. Its definition records precisely the consumed package rows from S and R together with the package rows attached to i, r, F, η, and ϵ. These are exactly the SubgroupUp, RepresentationRingUp, induction, restriction, and Frobenius rows named in the theorem.

Project the continuation ledger λ next. It records the corresponding public reads: source rows, endpoint rows, Frobenius rows, provenance rows, and public continuation rows. Therefore every public continuation witness for a provenance row is already present in the same finite carrier.

For the converse, take any row recorded by ρ or λ with one of the listed roles. The carrier definition repacks it as part of the $\mathsf{InducedRep}^{\uparrow}$ public provenance surface, and the consumer boundary of Definition 247.11 excludes every row not so recorded. Hence no host category, equality of induction or restriction functors, or Frobenius adjunction row is exported. □

Theorem 247.16 (InducedRepUp source-Frobenius exact consumption)**.** *A downstream row consumed from an* $\mathsf{InducedRep}^{\uparrow}$ *packet is visible exactly when it is one of the following finite public rows: a SubgroupUp source row exposed through S, a RepresentationRingUp source row exposed through R, an induction or restriction endpoint row, one of the Frobenius rows F, η, ϵ, or a provenance and continuation row recorded by ρ or λ. No host subgroup object, host representation category, host functor equality, or Frobenius adjunction object is visible outside this finite source–Frobenius ledger.*

Proof. Project a consumed row through the carrier packet $\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda)$. The source rows are exactly the SubgroupUp and RepresentationRingUp rows allowed by Definition 247.11; the endpoint rows are the displayed induction and restriction rows i and r.

Project the Frobenius part next. By Definition 247.9, every Frobenius read is one of F, η, ϵ and has endpoints in the displayed induction and restriction rows. By Definition 247.14, any finite continuation needed to consume those rows remains inside λ.

The provenance projection is fixed by Definition 247.15: every public provenance row is recorded by ρ or by the corresponding public continuation row in λ. Therefore any downstream row consumed from the packet is on the listed source, endpoint, Frobenius, provenance, or continuation surface.

Conversely, take any row on that finite list. The source rows repack through the consumer boundary, the endpoint and Frobenius rows repack through the Frobenius ledger boundary, and the provenance and continuation rows repack through ρ and λ. The four cited boundaries together reconstruct the same $\mathsf{InducedRep}^{\uparrow}$ public packet, so every listed row is visible to the consumer and no outside host-level datum is added. □

Theorem 247.17 (InducedRepUp Frobenius source-row determinacy)**.** *Let $\mathcal{I}_0 = (S, R, i, r, F, \eta, \epsilon, \rho_0, \lambda_0)$ and $\mathcal{I}_1 = (S, R, i, r, F, \eta, \epsilon, \rho_1, \lambda_1)$ be two* $\mathsf{InducedRep}^{\uparrow}$ *packets with the same SubgroupUp source row, the same RepresentationRingUp source row, the same induction and restriction endpoint rows, the same Frobenius rows F, η, ϵ, the same public* Cont *ledger rows, and the same* Pkg *provenance rows. Then the public classifier endpoints exported by $\mathcal{I}_0$ and $\mathcal{I}_1$ are related by* hsame.

Proof. Project both packets through the restriction-induction classifier. The source rows are the common SubgroupUp and RepresentationRingUp rows, and the endpoint rows are the common induction and restriction histories i and r.

Apply Definition 247.16. It rules out any consumed public row outside the common source rows, endpoint rows, Frobenius rows, Cont ledger, and Pkg provenance. Thus both packets expose the same finite public ledger.

Finally repack that common source-Frobenius ledger through Definition 247.9 and Definition 247.15. The classifier endpoint is therefore read from the same carried rows in both packets, so the two exported endpoints are related by hsame. □

Theorem 247.18 (InducedRepUp source classifier obligation)**.** *For an* $\mathsf{InducedRep}^{\uparrow}$ *carrier packet* $\mathcal{I} = (S, R, i, r, F, \eta, \epsilon, \rho, \lambda)$*, the public classifier is determined by the SubgroupUp source row* S*, the RepresentationRingUp source row* R*, the induction endpoint* i*, the restriction endpoint* r*, and the displayed* hsame *transport rows of Definition 247.3. No classifier read may use a host subgroup, representation category, or functor equality outside those carried rows.*

Proof. Project the carrier through Definition 247.11. The consumer sees the SubgroupUp and RepresentationRingUp source rows only through S and R, together with the induction and restriction endpoints.

Apply Definition 247.8. It identifies the visible classifier rows as $H_{\leq}, H_{\mathrm{rep}}, H_{\uparrow}$, and $H_{\downarrow}$ together with the public Frobenius and continuation rows.

Finally apply Definition 247.17. Once the source rows, endpoints, Frobenius rows, Cont ledger, and Pkg provenance are fixed, the exported classifier endpoints are related by hsame, so no outside classifier datum is available. □

Theorem 247.19 (InducedRepUp ledger exactness obligation)**.** *The visible Frobenius ledger reads of an* $\mathsf{InducedRep}^{\uparrow}$ *carrier are exactly the rows recorded by the* Cont *ledger* λ *and the* Pkg *provenance packet* ρ *for the consumed source certificates, induction endpoint, restriction endpoint, and Frobenius rows* F, η, ϵ*. Every visible read is on that finite ledger, and every listed row repacks into the public carrier.*

Proof. Begin with Definition 247.9. It restricts Frobenius reads to $F, \eta, \epsilon, \lambda$ with endpoints i and r over the consumed source packets S and R.

Use Definition 247.14. It shows that finite continuations among source reads, endpoint reads, Frobenius reads, provenance reads, and public continuation reads remain inside λ.

Apply Definition 247.15. The provenance rows are precisely those recorded by ρ or λ for the source, endpoint, and Frobenius roles. The reverse projection repacks each listed row into the carrier, so the ledger is exact. □

Theorem 247.20 (InducedRepUp NameCert threshold witness)**.** *The dependency-ready* $\mathsf{InducedRep}^{\uparrow}$ *obligation surface is the finite* BHist *packet consisting of the source classifier obligation of Definition 247.18, the ledger exactness obligation of Definition 247.19, and the source-Frobenius consumption boundary of Definition 247.16. It supplies downstream consumers exactly the carried source, endpoint, Frobenius, classifier,* Cont*, and* Pkg *rows.*

Proof. First use Definition 247.18. It fixes the public classifier from the two source certificates, the induction endpoint, the restriction endpoint, and the displayed hsame transport rows.

Next use Definition 247.19. It fixes the visible ledger reads as the rows in λ and ρ for the source, endpoint, and Frobenius roles.

Finally apply Definition 247.16. It states that downstream consumption is exactly the finite list of source, endpoint, Frobenius, provenance, and continuation rows. Repacking those three surfaces gives the dependency-ready $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ threshold, with no host category, functor equality, adjunction object, or external bridge row. □

Theorem 247.21 (InducedRepUp restriction-induction classifier equivalence)**.** *Assume that the consumed SubgroupUp classifier rows, the consumed RepresentationRingUp classifier rows, the induction and restriction endpoint* hsame *rows, the Frobenius rows* F, η, ϵ*, the provenance row, and the finite* Cont *ledger comparisons are each stable under reflexive, reversed, and composable comparison on their own displayed rows. Then the restriction-induction classifier of Definition 247.3*

is an equivalence relation on accepted $\mathsf{InducedRep}^{\uparrow}$ *carrier packets of Definition 247.1. Moreover every reflexive, reversed, or composite witness has exactly the same support roles*

$$S, R, i, r, F, \eta, \epsilon, \rho, \lambda$$

as the carriers it compares; no off-ledger host category, functor equality, adjunction object, character formula, or bridge row can be introduced by the equivalence operations.

Proof. Let

$$\mathcal{I}_a = (S_a, R_a, i_a, r_a, F_a, \eta_a, \epsilon_a, \rho_a, \lambda_a), \qquad \mathcal{I}_b = (S_b, R_b, i_b, r_b, F_b, \eta_b, \epsilon_b, \rho_b, \lambda_b),$$

and

$$\mathcal{I}_c = (S_c, R_c, i_c, r_c, F_c, \eta_c, \epsilon_c, \rho_c, \lambda_c)$$

be accepted carriers. The classifier of Definition 247.3 has seven visible coordinates: the two consumed source certificate comparisons, the entrywise source-row comparisons read from those certificates, the induction and restriction endpoint comparisons, the Frobenius comparisons for F, η, ϵ, the provenance comparison, and the finite Cont ledger comparison.

For reflexivity of $\mathcal{I}_a$, choose the reflexive comparison supplied by the assumed stability of each of those seven coordinates. This gives the SubgroupUp row S_a compared with itself, the RepresentationRingUp row R_a compared with itself, the source rows read from them compared with themselves, the endpoint rows i_a and r_a compared with themselves, the Frobenius rows F_a, η_a, ϵ_a compared with themselves, the provenance row ρ_a compared with itself, and the identity finite continuation comparison on λ_a. These rows are precisely the fields required by the restriction-induction classifier, so they form a witness $\mathcal{I}_a \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^{\uparrow}} \mathcal{I}_a$.

For symmetry, suppose $\mathcal{I}_a \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^{\uparrow}} \mathcal{I}_b$. Project the witness to the same seven coordinates. By the assumed reversed comparison for each coordinate, reverse the SubgroupUp and RepresentationRingUp source certificate comparisons, reverse the entrywise source-row comparisons, reverse the endpoint comparisons for i and r, reverse the Frobenius comparisons for F, η, ϵ, reverse the provenance comparison, and reverse the finite Cont comparison from λ_a to λ_b. Repacking these reversed rows gives a witness $\mathcal{I}_b \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^{\uparrow}} \mathcal{I}_a$.

For transitivity, suppose $\mathcal{I}_a \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^{\uparrow}} \mathcal{I}_b$ and $\mathcal{I}_b \sim^{\mathrm{ri}}_{\mathsf{InducedRep}^{\uparrow}} \mathcal{I}_c$. Project both witnesses coordinate by coordinate. Compose the two SubgroupUp comparisons through S_b, compose the two RepresentationRingUp comparisons through R_b, compose the entrywise source-row comparisons through the rows read from S_b and R_b, compose the endpoint comparisons through i_b and r_b, compose the Frobenius comparisons through F_b, η_b, ϵ_b, compose the provenance comparisons through ρ_b, and compose the finite continuation comparisons through λ_b. The assumed composability of each coordinate supplies exactly the rows required for a classifier witness from $\mathcal{I}_a$ to $\mathcal{I}_c$.

It remains to check that these operations do not enlarge the carrier support. The consumer boundary of Definition 247.11 limits source consumption to the SubgroupUp and RepresentationRingUp rows carried by S and R, together with the displayed endpoint, Frobenius, provenance, and continuation rows. The Frobenius ledger boundary of Definition 247.9 limits Frobenius reads to $F, \eta, \epsilon, \lambda$ with endpoints i and r. The ledger exactness obligation of Definition 247.19 says that visible ledger reads are exactly the rows recorded by ρ and λ for those source, endpoint, and Frobenius roles.

Therefore the reflexive witness, the reversed witness, and the composite witness all live on the same finite support $S, R, i, r, F, \eta, \epsilon, \rho, \lambda$. Any proposed comparison using a host representation category, functor equality, adjunction object, character formula, bridge row, or other off-ledger

datum is rejected before the classifier is formed, because such a row is not one of the seven coordinates listed in Definition 247.3 and is not visible through the exact consumer ledger. Thus the restriction-induction classifier is reflexive, symmetric, and transitive on accepted carrier packets, with no additional support beyond the displayed BEDC packet. □

Theorem 247.22 (InducedRepUp obligation closure package)**.** *The explicit obligation set for* $\mathsf{InducedRep}^{\uparrow}$ *is the conjunction of the carried* BHist *packet of Definition 247.1, the SubgroupUp source boundary, the RepresentationRingUp action boundary, the Frobenius classifier, the source classifier obligation, the ledger exactness obligation, and the consumer boundary. Each row is scoped through the displayed* hsame *transport rows, the finite* Cont *ledger, and the shared* Pkg *provenance packet. This package exports no host representation category, host functor equality, adjunction object, character formula, or standard bridge row.*

Proof. First read the carrier and source boundaries. The carrier packet lists exactly S, R, i, r, F, η, ϵ, ρ, and λ, while Definition 247.11 fixes the SubgroupUp source row and the RepresentationRingUp action row consumed by the package.

Next attach the classifier rows. Use Definition 247.8 for the Frobenius classifier surface and Definition 247.18 for the source classifier obligation. These rows are componentwise hsame transport over the displayed subgroup, representation-ring, induction, restriction, and Frobenius endpoints.

Now use Definition 247.19. It says that the visible reads are exactly the rows recorded by ρ and λ for the consumed source certificates, endpoints, Frobenius rows, provenance rows, and continuation rows.

Finally apply Definition 247.16 and Definition 247.20. Together they repack the listed carrier, classifier, source, ledger, and consumer rows into the public $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ obligation package. Since every exported row has already been accounted for in that finite package, no host category, functor equality, adjunction object, character formula, or standard bridge row is exported. □

Theorem 247.23 (InducedRepUp Subgroup source-consumer scope)**.** *Every downstream consumer of the SubgroupUp source used by* $\mathsf{InducedRep}^{\uparrow}$ *reads that source only through the displayed subgroup row of Definition 247.1, the source classifier of Definition 247.18, the continuation rows carried by* Cont*, and the provenance rows carried by* Pkg*. In particular, the consumer surface exposes no independent coset carrier, subgroup inclusion object, orbit classifier, or host-group equality beyond the source row already bounded by Definition 247.4.*

Proof. Start with the finite carrier packet of Definition 247.1. Its SubgroupUp input is the single source row S, together with the endpoint rows that connect S to the induction and restriction data.

Apply Definition 247.4. It identifies the subgroup source consumed by $\mathsf{InducedRep}^{\uparrow}$ as the displayed row, not as a second subgroup certificate or an external coset object. Then use Definition 247.18 to read the classifier attached to that source row.

Finally apply Definition 247.19 and Definition 247.22. The ledger says that the visible reads are exactly the recorded Cont and Pkg rows, and the closure package accounts for every exported source obligation. Hence a consumer can read the SubgroupUp source only through the listed source, classifier, continuation, and provenance rows. □

Theorem 247.24 (InducedRepUp RepresentationRing action scope)**.** *Every downstream consumer of the RepresentationRingUp action used by* $\mathsf{InducedRep}^{\uparrow}$ *reads that action only through the displayed representation row of Definition 247.1, the induction and restriction endpoints, the Frobenius row, and the finite ledger rows carried by* ρ *and* λ*. The consumer surface does not expose*

a separate representation category, action functor equality, adjunction object, or character formula beyond the boundary fixed by Definition 247.5.

Proof. Read first the representation component of Definition 247.1. The relevant data are the row R, the induction endpoint i, the restriction endpoint r, and the Frobenius row F; no ambient representation category is introduced by the carrier.

Apply Definition 247.5. It fixes the RepresentationRingUp action source as the displayed representation row together with the induction and restriction endpoints. Then apply Definition 247.18 to keep the classifier read tied to those endpoint rows.

Use Definition 247.19 for the ledger. The ledger projection permits only the reads recorded by ρ and λ for the representation, endpoint, Frobenius, continuation, and provenance rows. Combining this with Definition 247.22 exhausts the action surface visible to consumers. □

Theorem 247.25 (InducedRepUp consumer obligation exhaustion)**.** *The downstream consumer interface exported by* $\mathsf{InducedRep}^{\uparrow}$ *is exactly the union of the SubgroupUp source rows scoped in Definition 247.23, the RepresentationRingUp action rows scoped in Definition 247.24, and the displayed Frobenius, continuation, and package rows of the obligation closure package. No consumer receives an additional host category, coset carrier, action transport, adjunction, character formula, or standard bridge row.*

Proof. The SubgroupUp contribution is exactly the source row, source classifier, Cont rows, and Pkg provenance supplied by Definition 247.23. The RepresentationRingUp contribution is exactly the representation row, induction endpoint, restriction endpoint, Frobenius row, and ledger projection supplied by Definition 247.24.

It remains to compare this union with the chapter's finite obligation package. Apply Definition 247.18 for the classifier rows and Definition 247.19 for the recorded ledger reads. These two obligations identify the classifier and ledger pieces with rows already present in Definition 247.1.

Finally apply Definition 247.22. It repacks the source, action, Frobenius, continuation, and provenance rows into the single exported $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$ obligation surface. Since every row in either scoped source theorem is a row of that package, and every row of the package is one of those scoped rows or a displayed Frobenius, continuation, or provenance row, the downstream consumer interface is exactly the stated union. □

Theorem 247.26 (InducedRepUp standard bridge scope)**.** *The public* $\mathsf{InducedRep}^{\uparrow}$ *packet gives a conservative bridge to the standard induction-restriction reading only through the displayed SubgroupUp row, RepresentationRingUp row, induction endpoint, restriction endpoint, Frobenius ledger, componentwise* hsame *rows, finite* Cont *rows, and* Pkg *provenance. The bridge reads no host representation category, functor equality, adjunction object, character formula, or source row outside that packet.*

Proof. Start from the public consumer interface of Definition 247.25. It exhausts downstream reads by the SubgroupUp source rows, RepresentationRingUp action rows, Frobenius rows, continuation rows, and package provenance already present in the displayed carrier.

Interpret the standard induction-restriction reading by projecting only those rows: the subgroup source supplies the source boundary, the representation-ring row supplies the action boundary, and the induction and restriction endpoints supply the two displayed transport directions. The Frobenius ledger is then the sole comparison row available to the bridge.

Finally use Definition 247.22 to repack the projected rows into the same public obligation surface. Since the repacking has no coordinates beyond hsame, Cont, and Pkg on the displayed packet, the bridge is conservative over the public certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}$.

Constructive story: An induced-representation packet starts with two consumed certificates: a $\mathsf{Subgroup}^{\uparrow}$ row naming the inclusion and a $\mathsf{RepresentationRing}^{\uparrow}$ row naming the representation-ring source and target, then displays induction, restriction, Frobenius comparison, unit, counit, continuation, and provenance rows in one finite BHist package. The carrier admits exactly those displayed rows and excludes any host representation category, coset carrier, functor equality, adjunction object, character formula, or bridge row that is not present in the packet. Two accepted packets classify together by componentwise hsame transport on the subgroup source, representation-ring source, induction endpoint, restriction endpoint, Frobenius ledger, provenance packet, and finite Cont ledger, so the source specification covers the two dependency certificates and the pattern specification is the displayed induction–restriction/Frobenius shape. The ledger keeps the raw and visible sides aligned by retaining the subgroup row, representation-ring row, induction and restriction endpoints, Frobenius row, unit and counit rows, source and Frobenius continuation ledgers, and Pkg provenance as inspectable histories. With one such packet in hand, the five name-certificate obligations are componentwise: the displayed ledger inhabits the carrier, inherited and endpoint hsame rows provide reflexivity, symmetry, and transitivity, and transported dependency and Frobenius rows repack into the same accepted carrier surface. A history h lies in the signature gap of an $\mathsf{InducedRep}^{\uparrow}$ token p when the source-domain accepts h as this packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h must present exactly the subgroup, representation-ring, induction, restriction, Frobenius, continuation, and provenance rows that p encodes. The chapter registers this finite induction–restriction construction under the name $\mathsf{InducedRep}^{\uparrow}$; Frobenius reciprocity is a displayed ledger comparison inside BEDC, with no imported functor category, host adjunction, standard bridge, or extra source.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 247.25 and Definition 247.26 bind the mature finite consumer surface and bridge boundary to the carried BHist packet, SubgroupUp source rows, RepresentationRingUp action rows, Frobenius rows, hsame, Cont, and Pkg provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InducedRep}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.InducedRepUp.InducedRepCarrierPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This does not close external representation categories, functor equality, adjunction objects, character formulas, or rows outside the displayed induction-restriction packet.

Upgrade path: No further paper theory-axis work is recorded for this chapter; audit and bridge-status changes remain on the verification axis.

Chapter 248

A Concrete Naming Certificate for $\mathsf{VermaModule}^{\uparrow}$

The Verma-module interface $\mathsf{VermaModule}^{\uparrow}$ packages, for a semisimple Lie algebra and a highest weight, the universal highest-weight module obtained by inducing from a Borel subalgebra. It depends on $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ (chapter 141) and $\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}$ (chapter 144).

248.1 Carrier definition

Theorem 248.1 (Verma-module finite carrier obligation)**.** *A* $\mathsf{VermaModule}^{\uparrow}$ *packet is a finite* BHist *row package containing a displayed* $\mathsf{LieAlgebra}^{\uparrow}$ *source row,* $\mathsf{RootSystem}^{\uparrow}$ *positive-root rows, a highest-weight row, Borel action rows, induced-module generator rows, and* Pkg *provenance naming the displayed Lie-algebra and root-system sources. The packet contains no host quotient module, no selected PBW basis, and no ambient category of modules beyond these finite rows.*

Lean4 ✓ `BEDC.Derived.VermaModuleUp.VermaModuleFiniteCarrier obligation surface`

Proof. Unpack the packet as a finite BEDC history. The Lie-algebra source, positive root data, highest weight, Borel action, induced generator, and provenance coordinates are the displayed rows of that history package.

Each consumer-visible coordinate is obtained by projecting one of those rows or transporting it along the displayed hsame and Cont data. Since the packet has no coordinate for a host quotient module or selected basis, neither object is introduced by the carrier obligation. □

Lean4 ✓ `BEDC.Derived.VermaModuleUp.VermaModuleFiniteCarrier carrier obligation`

248.2 The certificate

Theorem 248.2 (Verma-module highest-weight ledger)**.** *Every accepted* $\mathsf{VermaModule}^{\uparrow}$ *packet exposes exactly the highest-weight row, lowering-operator rows from the displayed positive-root data, Borel action rows,* hsame *comparison rows,* Cont *provenance rows, and* Pkg *dependency rows consumed by downstream representation certificates. It exports no quotient by a maximal submodule, no irreducible-module classifier, and no chosen basis for weight spaces.*

Proof. Project an accepted packet to the rows named in Definition 248.1. The highest-weight coordinate, lowering-operator coordinates, Borel action, transport, continuation, and package provenance are all finite BHist entries.

Downstream representation certificates can only read through those finite entries. Componentwise transport preserves the highest-weight and action rows, and continuation rows record the finite lowering operations already present in the packet.

The carrier contains no row for a maximal-submodule quotient, irreducibility classifier, or basis choice. The accepted ledger therefore exhausts exactly the displayed highest-weight surface. □

Lean4 ✓ `BEDC.Derived.VermaModuleUp.VermaModuleFiniteCarrier highest weight ledger`
↪ variant `BEDC.Derived.VermaModuleUp.VermaModuleHighestWeightLedger boundary`

Theorem 248.3 (Verma-module NameCert obligation surface)**.** *The carrier obligation of Definition 248.1, the componentwise classifier transport over the displayed highest-weight and action rows, the highest-weight ledger of Definition 248.2, and the source dependencies on* $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RootSystem}^{\uparrow}}$ *provide the local* $\mathsf{NameCert}_{\mathsf{VermaModule}^{\uparrow}}$ *obligation surface. The surface is entirely* BHist*-anchored and does not introduce a host quotient module or a choice of basis.*

Proof. Begin with the finite carrier obligation. It supplies the source rows, positive-root rows, highest-weight row, Borel action rows, induced-generator rows, and package provenance needed for the local naming surface.

Transport the displayed classifier componentwise along hsame. The highest-weight and action rows move through the same finite coordinates, while the Cont rows record only the lowering operations already carried by the packet.

Finally apply the highest-weight ledger to identify the exact downstream readback surface. Together with the displayed $\mathsf{LieAlgebra}^{\uparrow}$ and $\mathsf{RootSystem}^{\uparrow}$ provenance, these rows provide the local $\mathsf{NameCert}_{\mathsf{VermaModule}^{\uparrow}}$ obligation surface without adding a quotient module, irreducible classifier, or basis choice. □

Lean4 ✓ `BEDC.Derived.VermaModuleUp.VermaModuleFiniteCarrier obligation ledger surface`
Lean4 ✓ `BEDC.Derived.VermaModuleUp.VermaModuleFiniteCarrier semanticNameCert`

Closure status for $\mathsf{NameCert}_{\mathsf{VermaModule}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VermaModule}^{\uparrow}$ packet is generated from finite BHist rows for a Lie-algebra source, a highest-weight row, a generator row, an action ledger, and the displayed module-facing carrier rows. The Verma surface is assembled by replaying the generator and action routes through Cont, so consumers see finite obligation rows rather than a host universal module. The classifier transports the highest-weight, generator, and action rows by hsame, and Pkg records the consumed Lie-algebra and module dependencies. The resulting certificate names the finite highest-weight ledger and its semantic carrier boundary without exporting irreducible quotients or basis choices.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VermaModule}^{\uparrow}}$, seed)

Scope closed: This chapter binds the finite carrier obligation ledger surface and semantic NameCert surface for $\mathsf{VermaModule}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{VermaModule}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim universal-module construction, irreducible quotient classification, character formulae, host module equality, or a chosen PBW basis.

Upgrade path: Obligation closure requires Lean encodings of the finite carrier, highest-weight action ledger, semantic NameCert rows, and non-escape boundary.

Chapter 249

A Concrete Naming Certificate for $\mathsf{Hypergeometric}^{\uparrow}$

The hypergeometric-function interface $\mathsf{Hypergeometric}^{\uparrow}$ packages the analytic continuation of ${}_pF_q$ series defined by ascending-factorial coefficient ratios. It depends on $\mathsf{NameCert}_{\mathsf{CplxAnalytic}^{\uparrow}}$ (chapter 76) and $\mathsf{NameCert}_{\mathsf{Gamma}^{\uparrow}}$ (chapter 75).

249.1 Carrier definition

Definition 249.1 (HypergeometricUp BHist source packet)**.** *A* $\mathsf{Hypergeometric}^{\uparrow}$ *source packet is a finite* BHist *tuple* $(C, G, A, B, R, S, \rho, \lambda)$*. The row* C *is the consumed* $\mathsf{CplxAnalytic}^{\uparrow}$ *dependency packet, and the row* G *is the consumed* $\mathsf{Gamma}^{\uparrow}$ *dependency packet. The rows* A *and* B *record the visible numerator and denominator parameter lists for the* ${}_pF_q$ *series, the row* R *records the ascending-factorial coefficient-ratio boundary, and the row* S *records the finite series-readback boundary over the displayed complex-analytic carrier. The row* ρ *is the* Pkg *provenance linking the two dependency packets to the hypergeometric name, and* λ *is the finite* Cont *ledger storing the* hsame *transports among parameters, coefficient ratios, and readback rows.*

Acceptance is exactly this finite packet. It carries no analytic continuation theorem, no host power-series object, no monodromy data, no differential equation, and no external equality of special functions.

Theorem 249.2 (HypergeometricUp root carrier)**.** *Every accepted* $\mathsf{Hypergeometric}^{\uparrow}$ *carrier row projects to the* $\mathsf{CplxAnalytic}^{\uparrow}$ *dependency row, the* $\mathsf{Gamma}^{\uparrow}$ *dependency row, the parameter rows* A *and* B*, the coefficient-ratio row* R*, the finite readback row* S*, the* Pkg *provenance* ρ*, and the* Cont *ledger* λ *of Definition 249.1. Conversely, those rows repack as an accepted carrier packet when their displayed* hsame *transports are the ones recorded by* λ*.*

Proof. Project an accepted packet by its displayed tuple coordinates. The first two coordinates expose the complex-analytic and Gamma dependency packets, while the parameter, coefficient-ratio, readback, provenance, and ledger coordinates are read directly from the finite BHist packet.

For the converse, place the same displayed rows back into the tuple constructor and use the stored Cont ledger as the acceptance evidence. The construction does not inspect a host series or an analytic continuation theorem, because no such coordinate occurs in the packet. □

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket root carrier`

Theorem 249.3 (HypergeometricUp root classifier)**.** *For two* Hypergeometric$^{\uparrow}$ *source packets, the root classifier compares only the consumed* CplxAnalytic$^{\uparrow}$ *and* Gamma$^{\uparrow}$ *dependency rows, the numerator and denominator parameter rows, the coefficient-ratio row, the finite readback row, the* Pkg *provenance row, and the finite* Cont *and* hsame *ledger. Transport along those rows preserves the accepted carrier and supplies no analytic-continuation classifier outside the displayed packet.*

Proof. Start with the dependency comparisons. The carried CplxAnalytic$^{\uparrow}$ and Gamma$^{\uparrow}$ rows supply the only classifiers available for the analytic carrier and Gamma-derived parameter faces.

Next transport the parameter rows, the coefficient-ratio row, and the finite series-readback row through the displayed hsame comparisons stored in λ. The Pkg row records that the transported faces belong to the same hypergeometric packet.

Repack the transported rows by Definition 249.2. Since the classifier has compared every packet coordinate and no continuation classifier is present, the result is exactly the root classifier surface. □

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket root classifier`

Theorem 249.4 (HypergeometricUp root ledger)**.** *The* Hypergeometric$^{\uparrow}$ *root ledger is exact: its rows are precisely the* CplxAnalytic$^{\uparrow}$ *dependency, the* Gamma$^{\uparrow}$ *dependency, the two parameter lists, the coefficient-ratio boundary, the finite* Cont *readback, the* hsame *transports, and the* Pkg *provenance of Definition 249.1. No analytic continuation, singularity classification, monodromy representation, differential-equation solution, or host-function equality is part of this ledger.*

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket root ledger`

Proof. Coverage follows by projecting the accepted source packet. Each ledger entry is one of the displayed tuple coordinates or one of the transport rows stored in the finite Cont ledger.

Soundness is the reverse projection. Given exactly those rows, the source packet constructor rebuilds the same accepted Hypergeometric$^{\uparrow}$ carrier and the same classifier surface.

Completeness follows from the packet inventory. There is no coordinate for analytic continuation, singularity data, monodromy, a host differential equation, or external function equality, so none of those rows can be consumed from the root ledger. □

Theorem 249.5 (HypergeometricUp root stability)**.** *If a* Hypergeometric$^{\uparrow}$ *source packet is transported by the displayed* hsame *rows on its* CplxAnalytic$^{\uparrow}$ *dependency,* Gamma$^{\uparrow}$ *dependency, parameter lists, coefficient-ratio boundary, finite readback row, and* Pkg *provenance, then the transported packet is again accepted by the same root carrier and classified by the same root classifier.*

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket root stability`

Proof. Read the carrier rows using Definition 249.2. The hypotheses give an hsame transport for every visible coordinate of the packet.

Use the finite Cont ledger to move the coefficient-ratio and readback rows with the parameter transports. The dependency rows move through their own displayed NameCert classifiers, and the Pkg provenance remains attached to the transported packet.

Apply Definition 249.3 to the transported row inventory. It compares exactly the rows just transported, so the accepted packet and its classifier are stable without using analytic continuation as an uncarried host theorem. □

Theorem 249.6 (HypergeometricUp endpoint ledger determinacy)**.** *Let two* $\mathsf{Hypergeometric}^{\uparrow}$ *source packets share the* $\mathsf{CplxAnalytic}^{\uparrow}$ *dependency row and the* $\mathsf{Gamma}^{\uparrow}$ *dependency row. Suppose their numerator-parameter, denominator-parameter, finite readback, and* Pkg *provenance rows are compared rowwise by* hsame*, and suppose their coefficient-ratio, finite ledger, and endpoint rows are carried by compatible* Cont *rows over those comparisons. Then the transported endpoint, the coefficient-ratio row, and the finite ledger are determined up to* hsame *by the common* BHist *packet rows.*

Equivalently, any downstream consumer that sees only those dependency rows, parameter comparisons, readback rows, provenance rows, and compatible Cont *entries cannot distinguish two different transported endpoints, coefficient-ratio rows, or finite ledgers inside the root surface.*

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket endpoint ledger determinacy`

Proof. First apply Definition 249.3. The classifier compares exactly the $\mathsf{CplxAnalytic}^{\uparrow}$ dependency, the $\mathsf{Gamma}^{\uparrow}$ dependency, the numerator and denominator parameter rows, the readback rows, and the Pkg provenance rows, so the two packets have the same root-facing classification data.

Next apply Definition 249.5. The compatible Cont rows transport the coefficient-ratio boundary, finite ledger, and endpoint through the already matched parameter and dependency rows, keeping the transported packet inside the same root carrier and classifier surface.

Finally apply Definition 249.4. Ledger exactness says that no further row is available to decide an endpoint, coefficient-ratio boundary, or finite ledger. The conclusion therefore uses only the finite BHist packet, Cont, hsame, and Pkg rows; it does not use analytic continuation or host special-function equality. □

Theorem 249.7 (HypergeometricUp root consumer boundary)**.** *A downstream consumer of the* $\mathsf{Hypergeometric}^{\uparrow}$ *root may read exactly the* BHist *source packet of Definition 249.1, the carrier projection of Definition 249.2, the classifier stability of Definition 249.3, the exact ledger of Definition 249.4, and the transport stability of Definition 249.5. It may consume the displayed complex-analytic dependency, Gamma dependency, parameter rows, coefficient-ratio row, finite readback,* Cont*,* hsame*, and* Pkg *provenance; it may not consume analytic continuation as an uncarried host theorem.*

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket root consumer boundary`

Proof. First project the source packet. The carrier theorem lists every row available to the consumer and repacks exactly those rows as the accepted $\mathsf{Hypergeometric}^{\uparrow}$ root.

Next apply the classifier and stability theorems. They show that all comparisons and transports remain inside the same dependency, parameter, coefficient-ratio, readback, provenance, and ledger surface.

Finally apply ledger exactness. Since the ledger has no coordinate for analytic continuation or for any external special-function equality, the consumer boundary is precisely the displayed finite BHist packet. □

249.2 The certificate

Theorem 249.8 (HypergeometricUp NameCert obligation surface)**.** *The source packet of Definition 249.1 together with Definition 249.2, Definition 249.3, Definition 249.4, Definition 249.5, and Definition 249.7 supplies the paper-side* $\mathsf{NameCert}_{\mathsf{Hypergeometric}^{\uparrow}}$ *obligation surface. Its rows are*

exactly the BHist *source packet,* Cont, hsame, Pkg, *the* CplxAnalytic$^{\uparrow}$ *dependency row, the* Gamma$^{\uparrow}$ *dependency row, the parameter rows, the coefficient-ratio row, and the finite readback row.*

Proof. Start with the source packet definition. It displays the carrier coordinates: the two dependency packets, numerator and denominator parameter rows, the coefficient-ratio row, the finite readback row, package provenance, and the finite continuation ledger.

Apply the root carrier, classifier, ledger, and stability theorems. They show that these displayed coordinates are enough for habitation, rowwise classification, ledger exactness, and transport stability of the Hypergeometric$^{\uparrow}$ packet.

Finally use the root consumer-boundary theorem. It states that downstream reads factor through the same displayed rows, so the five local theorems assemble the paper-side NameCert$_{\text{Hypergeometric}^{\uparrow}}$ obligation surface without adding an analytic-continuation row. □

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket namecert obligation surface`

Theorem 249.9 (HypergeometricUp signature-gap boundary)**.** *Any* Hypergeometric$^{\uparrow}$ *token gap may expose only the rows of Definition 249.8: the finite* BHist *packet,* Cont, hsame, Pkg, *the* CplxAnalytic$^{\uparrow}$ *and* Gamma$^{\uparrow}$ *dependency rows, parameter rows, the coefficient-ratio row, and the finite readback row. It never exposes analytic continuation, monodromy, singularity classification, host differential equations, or host special-function equality.*

Proof. Use the NameCert obligation surface to enumerate the only rows accepted and transported by the certificate. The carrier and classifier theorems keep token gap reads inside those finite packet coordinates.

The ledger and stability theorems exclude additional coordinates from the transport surface. In particular, analytic continuation, monodromy, singularity classification, host differential equations, and host special-function equality are not fields of the packet and cannot be recovered by hsame or Cont.

The consumer-boundary theorem gives the downstream direction. Any exposed token gap row must be consumed through the finite packet surface already listed, so the signature gap has exactly that boundary. □

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket signature gap boundary`

Theorem 249.10 (HypergeometricUp public certificate handoff)**.** *The public* NameCert$_{\text{Hypergeometric}^{\uparrow}}$ *handoff consists exactly of the finite* BHist *source packet of Definition 249.1,* Cont, hsame, Pkg, *the* CplxAnalytic$^{\uparrow}$ *dependency row, the* Gamma$^{\uparrow}$ *dependency row, the parameter rows, the coefficient-ratio row, the finite readback row, and the signature-gap exclusion of Definition 249.9. It exports no analytic continuation, monodromy, singularity classification, host differential equation, or host special-function equality.*

Lean4 ✓ `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket public certificate handoff`

Proof. First apply Definition 249.8. It packages the finite source packet with exactly the continuation, transport, provenance, dependency, parameter, coefficient-ratio, and readback rows required by the public name certificate.

Next apply Definition 249.9. The same finite rows are the only rows visible through a token gap, and the excluded analytic objects have no coordinate in the accepted packet.

Combining the obligation surface with the gap boundary gives the consumer handoff. A downstream chapter may cite this theorem to receive the displayed finite BHist packet and the stated exclusions without importing any host analytic continuation or special-function equality. □

Theorem 249.11 (HypergeometricUp finite consumer completeness)**.** *Every downstream finite consumer of the public* Hypergeometric$^{\uparrow}$ *handoff factors through exactly the finite* BHist *source packet, the* CplxAnalytic$^{\uparrow}$ *and* Gamma$^{\uparrow}$ *dependency rows, the parameter rows, the coefficient-ratio row, the finite readback row, the signature-gap boundary, componentwise* hsame *transport, finite* Cont *routes,* Pkg *provenance, and the local* NameCert *surface displayed in this chapter. The factorization exposes no analytic continuation, monodromy, singularity classification, host differential equation, asymptotic package, or host special-function equality.*

Proof. Begin with the certificate surface. By Definition 249.8, the accepted packet contains only the finite source packet, the two dependency rows, parameter rows, coefficient-ratio row, readback row, continuation ledger, transport rows, package provenance, and local name-certificate data.

Next apply Definition 249.9. A token-gap read can expose only those same finite coordinates, so a finite consumer has no separate channel for analytic continuation, monodromy, singularity classification, host differential equations, or host special-function equality.

Finally use Definition 249.10. The public handoff is the consumer-facing form of the same packet. Therefore every finite downstream consumer factors through that displayed surface, and the listed exclusions remain outside the consumer boundary. □

Closure status for NameCert$_{\text{Hypergeometric}^{\uparrow}}$.

Constructive story: A hypergeometric packet is assembled from the finite BHist source packet, the CplxAnalytic$^{\uparrow}$ and Gamma$^{\uparrow}$ dependency rows, the parameter rows, the coefficient-ratio row, the finite readback row, and the displayed NameCert obligation surface. The carrier accepts only these rows together with Cont, hsame, and Pkg provenance, so downstream consumers receive the public certificate handoff while analytic continuation, differential-equation semantics, asymptotics, and standard special-function bridges remain outside the displayed packet.

Theory closure: TheoryGate(NameCert$_{\text{Hypergeometric}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge boundary is the displayed finite hypergeometric source, signature-gap, ledger, hsame, Cont, Pkg, and NameCert surface.

Formal verification: FormalStatus(NameCert$_{\text{Hypergeometric}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.HypergeometricUp.HypergeometricBHistSourcePacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close analytic continuation, differential equations, asymptotics, standard hypergeometric identities, or standard bridge rows.

Upgrade path: No further paper theory-axis grade remains; standard analytic identities and bridge checking stay outside this closure block.

Chapter 250

A Concrete Naming Certificate for $\mathsf{Bessel}^{\uparrow}$

The Bessel-function interface $\mathsf{Bessel}^{\uparrow}$ packages the holomorphic solutions to Bessel's ODE $x^2y'' + xy' + (x^2 - n^2)y = 0$, with their integral representations and recurrence laws. It depends on $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ (chapter 193) and $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ (chapter 68).

250.1 Carrier definition

Definition 250.1 (BesselUp BHist carrier packet)**.** *The* $\mathsf{Bessel}^{\uparrow}$ *carrier is the* BHist *packet*

$$\mathcal{B} = (O, H, \nu, \sigma, \tau, \rho, \kappa, \pi)$$

where O is an $\mathsf{Ode}^{\uparrow}$ *source row for the displayed Bessel differential equation, H is a* $\mathsf{Holomorphic}^{\uparrow}$ *source row for the corresponding disk-local solution histories, ν is the order and index row, σ and τ are the solution endpoint rows, ρ is the finite recurrence ledger, κ is the family of* hsame *transport rows connecting the ODE, holomorphic, order, and endpoint entries, and π is the shared* Pkg *provenance row. A* $\mathsf{Bessel}^{\uparrow}$ *consumer reads solution data only through this packet.*

250.2 The certificate

Theorem 250.2 (BesselUp root ODE source obligation)**.** *Every accepted* $\mathsf{Bessel}^{\uparrow}$ *root packet projects an* $\mathsf{Ode}^{\uparrow}$ *source row for the Bessel equation, its order and index row, the solution endpoint rows, the ODE-side endpoint classifier, and the* Pkg *provenance recorded in Definition 250.1. The root packet exports no host analytic solution operator and no equality of ordinary functions.*

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket root ode source obligation`

Proof. Project the carrier packet of Definition 250.1 to its O, ν, σ, τ, κ, and π coordinates. These are exactly the ODE source, order data, endpoint data, transport rows, and provenance rows named in the statement.

Repack the projected rows through the $\mathsf{Ode}^{\uparrow}$ naming surface. Since the carrier stores endpoint comparison as hsame rows and package provenance as Pkg data, the projection never requires a host solution operator or function equality. □

Theorem 250.3 (BesselUp holomorphic consumer boundary)**.** *Every holomorphic read accepted by the* $\mathsf{Bessel}^{\uparrow}$ *root packet is determined by the* $\mathsf{Holomorphic}^{\uparrow}$ *source row, the solution endpoint rows, the order/index row, the* hsame *transport packet, and the shared* Pkg *provenance of Definition 250.1. It does not add a separate complex-analysis carrier or a standard holomorphic function bridge.*

Proof. Read the H coordinate of the carrier packet together with the endpoint rows, order/index row, transport packet, and provenance. These are the only rows exposed to the holomorphic consumer side.

Transport the endpoint histories along the componentwise hsame rows in κ. Repacking those rows through the $\mathsf{Holomorphic}^{\uparrow}$ naming surface gives the same holomorphic consumer boundary, while the packet contains no field for a separate carrier or standard bridge. □

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket holomorphic consumer boundary`

Theorem 250.4 (BesselUp recurrence ledger exactness)**.** *The recurrence ledger* ρ *in a* $\mathsf{Bessel}^{\uparrow}$ *carrier packet exhausts the accepted recurrence reads for the carried order/index row and solution endpoint rows. Each recurrence read is a finite* Cont *route transported by the packet's* hsame *rows and justified by the same* Pkg *provenance; conversely each ledger entry repacks as one accepted recurrence read.*

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket recurrence ledger exactness`

Proof. Project the recurrence ledger ρ from Definition 250.1. Each entry has source endpoints, target endpoints, order/index data, and a displayed finite Cont route.

Use the hsame packet κ to transport the endpoints and order row inside the shared BHist carrier. The Pkg row π records that the transported route belongs to the same Bessel packet.

For the converse direction, start from any accepted recurrence read. Its fields are exactly a source endpoint row, target endpoint row, order/index row, finite continuation route, transport row, and provenance row, so it repackages as an entry of ρ. □

Theorem 250.5 (BesselUp name-certificate obligation surface)**.** *The* $\mathsf{Bessel}^{\uparrow}$ *root naming obligation surface consists exactly of the carrier packet of Definition 250.1, the* $\mathsf{Ode}^{\uparrow}$ *source obligation of Definition 250.2, the* $\mathsf{Holomorphic}^{\uparrow}$ *consumer boundary of Definition 250.3, and the recurrence ledger exactness of Definition 250.4. This surface contains no host analytic primitive, no standard Bessel model, and no function-space quotient row.*

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket namecert obligation surface`

Proof. Assemble the packet by projection from Definition 250.1. The ODE source, holomorphic source, order/index row, endpoints, recurrence ledger, transport rows, and provenance are all coordinates of the same BHist carrier.

Apply Definition 250.2 to identify the ODE source side and Definition 250.3 to identify the holomorphic consumer side. Both projections use the same transport packet and provenance row.

Finally apply Definition 250.4. The finite recurrence ledger covers exactly the accepted recurrence reads, so repacking the three projections gives the whole name-certificate obligation surface without adding analytic primitives or quotient data. □

Theorem 250.6 (BesselUp carrier, equation, and order obligation)**.** *An accepted* $\mathsf{Bessel}^{\uparrow}$ *name-certificate packet binds the Bessel ODE source row* O*, the order and index row* ν*, the two solution endpoint rows, the ODE-side endpoint classifier, the* hsame *transport packet, and the shared* Pkg *provenance to one* BHist *carrier packet* $\mathcal{B}$ *of Definition 250.1. No consumer can read the equation*

row or the order row through a second carrier, a host analytic solution operator, or an equality of ordinary functions.

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket carrier equation order obligation`

Proof. Project the accepted root packet through Definition 250.5. Its carrier coordinate is the packet $\mathcal{B}$ of Definition 250.1, whose O, ν, endpoint, transport, and provenance coordinates are the displayed ODE source, order and index, endpoint, hsame, and Pkg rows.

Apply Definition 250.2. This projection already identifies the ODE row, the order and index row, the endpoint classifiers, and the provenance as rows of the same root packet. Since the obligation surface has no field for a second carrier, host solution operator, or ordinary-function equality, those data are not exported. □

Theorem 250.7 (BesselUp recurrence Cont ledger obligation)**.** *Every recurrence read accepted by a* Bessel$^{\uparrow}$ *name-certificate packet is exactly one of the finite* Cont *ledger rows already exposed by Definition 250.4, transported along the packet's* hsame *rows and justified by the same* Pkg *provenance. Conversely, each such finite ledger row repacks as an accepted recurrence read for the same order and endpoint rows.*

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket recurrence cont ledger obligation`

Proof. Start from an accepted recurrence read of the name-certificate packet. Apply Definition 250.5 to place the read inside the carrier packet, with the same order row, endpoint rows, transport packet, and provenance as the root surface.

Now apply Definition 250.4. It says that the recurrence ledger ρ exhausts accepted recurrence reads and that each read is a finite Cont route transported by the packet's hsame rows. The reverse direction in the same theorem repacks each finite ledger entry as an accepted recurrence read, so the exposed ledger is exact. □

Theorem 250.8 (BesselUp asymptotic ledger scope)**.** *Any holomorphic or asymptotic consumer read of a* Bessel$^{\uparrow}$ *packet is scoped to the displayed Bessel ODE source row, the* Holomorphic$^{\uparrow}$ *source row, the order and index row, the solution endpoint rows, the recurrence ledger, the* hsame *transport packet, and the shared* Pkg *provenance. The read exposes no standard analytic primitive, no host Bessel function model, no function-space quotient, and no separate asymptotic carrier.*

Lean4 ✓ `BEDC.Derived.BesselUp.BesselRootPacket asymptotic ledger scope`

Proof. For holomorphic reads, apply Definition 250.3. It restricts the consumer surface to the Holomorphic$^{\uparrow}$ source row, endpoint rows, order and index row, transport packet, and provenance of the shared carrier.

For recurrence and endpoint movement, apply Definition 250.4. The only routes available to the consumer are the finite recurrence Cont rows transported by the same hsame packet and justified by the same provenance.

Finally compose these two projections with Definition 250.5. The resulting surface is the ODE row, HolomorphicUp row, order row, endpoints, recurrence ledger, transport packet, and Pkg row of one BHist carrier. Because the root surface excludes host analytic primitives, standard Bessel models, function-space quotients, and separate carriers, the asymptotic read has no additional row to expose. □

Theorem 250.9 (BesselUp public certificate export)**.** *The public* $\mathsf{Bessel}^{\uparrow}$ *certificate is exactly the finite packet assembled from the Bessel ODE source row, the* $\mathsf{Holomorphic}^{\uparrow}$ *source row, the order and index rows, the solution endpoint rows, the recurrence ledger, the asymptotic ledger scope, componentwise* hsame *transport, finite* Cont *routes, and the* Pkg *provenance already fixed in this chapter. It exports analytic existence, analytic uniqueness, host Bessel functions, asymptotic expansions as functions, and a standard bridge nowhere in the public packet.*

Proof. Start with the ODE and holomorphic source boundary. Apply Definition 250.2 to fix the displayed Bessel ODE row and Definition 250.3 to bind every holomorphic consumer read to the same source row, endpoint rows, order and index rows, transport packet, and provenance.

Next project the recurrence data. By Definition 250.4, each recurrence read is one of the finite ledger entries transported by the packet's componentwise hsame rows and justified by the same Pkg provenance.

Finally combine those rows with Definition 250.5 and Definition 250.8. The first theorem supplies the carrier, classifier, transport, and name-certificate obligations; the second restricts holomorphic and asymptotic-facing reads to the displayed finite packet. No coordinate remains for analytic existence, uniqueness, host Bessel functions, functional asymptotic expansions, or a standard bridge. □

Theorem 250.10 (BesselUp standard root bridge boundary)**.** *The public root packet exported by Definition 250.9 is exactly the bridge-facing surface for* $\mathsf{Bessel}^{\uparrow}$*. Its visible rows are the* $\mathsf{Ode}^{\uparrow}$ *source row, the* $\mathsf{Holomorphic}^{\uparrow}$ *source row, the order and index histories, the solution endpoints, the recurrence ledgers, componentwise* hsame *transport, finite* Cont *routes, the shared* Pkg *provenance, and the local* NameCert *rows. No bridge-facing coordinate is available for analytic existence, asymptotic expansions as functions, host Bessel functions, or a standard analytic model.*

Proof. First take the public packet of Definition 250.9. It already identifies the ODE, holomorphic, order, endpoint, recurrence, transport, continuation, and provenance rows that a public consumer may receive.

Next compose the packet with the root and recurrence boundaries. The ODE source is fixed by Definition 250.2, the holomorphic reads by Definition 250.3, and the recurrence reads by Definition 250.4. The name-certificate obligations of Definition 250.5 place these rows under the same local NameCert and Pkg surface.

Finally use coordinate exhaustion. The carrier packet has coordinates only for the displayed ODE row, holomorphic row, order and index histories, endpoints, recurrence ledgers, hsame, Cont, Pkg, and local NameCert data. Since analytic existence, asymptotic functions, host Bessel functions, and a standard analytic model are not coordinates of that packet, the public export is precisely the bridge-facing boundary and nothing larger. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bessel}^{\uparrow}}$**.**

Constructive story: A Bessel packet is generated by placing an $\mathsf{Ode}^{\uparrow}$ row for the displayed Bessel differential equation beside a $\mathsf{Holomorphic}^{\uparrow}$ row for the disk-local solution histories, then adding order and index histories, solution endpoints, recurrence entries, transport rows, and package provenance. The carrier accepts only this finite packet $(O, H, \nu, \sigma, \tau, \rho, \kappa, \pi)$; a host analytic solution operator, ordinary-function equality, function-space quotient, or standard Bessel model is not part of the surface. Two accepted packets classify together by componentwise hsame transport of the ODE source, holomorphic source, order and index rows, solution endpoints, recurrence ledgers, and provenance, with finite Cont routes carrying each recurrence readback. The source specification is the paired ODE and holomorphic certificate boundary, the pattern specification is the Bessel-equation solution surface with its recurrence laws, and the ledger keeps the order/index row, solution endpoints, recurrence routes, endpoint classifiers, and shared Pkg row visible. The five name-certificate obligations are witnessed by an accepted root packet, by the equivalence laws of the component classifiers, and by recurrence-ledger exactness, which says every transported recurrence read repacks as one of the same finite ledger entries. A packet history

h lies in the signature gap of a Bessel$^{\uparrow}$ token p when the source-domain accepts its ODE and holomorphic rows, the signature relation pairs it with an answer Bessel packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the order, solution endpoints, and recurrence ledger are precisely the rows encoded by p. The chapter registers this ODE-and-recurrence packet under the name Bessel$^{\uparrow}$; it introduces no analytic existence or uniqueness theorem, no asymptotic expansion package, no host Bessel function, and no standard bridge.

Theory closure: TheoryGate(NameCert$_{\text{Bessel}\uparrow}$, maturePackageClosed)

Scope closed: The obligation scope binds the Bessel ODE row, Holomorphic$^{\uparrow}$ source row, order and index rows, solution endpoints, recurrence ledger, componentwise hsame transport, finite Cont routes, and Pkg provenance of the root packet.

Formal verification: FormalStatus(NameCert$_{\text{Bessel}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.BesselUp.BesselRootPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not claim analytic existence or uniqueness, asymptotic expansions, host Bessel functions, function-space quotients, or a standard bridge.

Upgrade path: No further paper-side theory-axis upgrade remains for the displayed ODE, holomorphic, recurrence-ledger, classifier, provenance, and bridge consumer surface.

Chapter 251

A Concrete Naming Certificate for $\mathsf{ThetaFunction}^{\uparrow}$

The theta-function interface $\mathsf{ThetaFunction}^{\uparrow}$ packages the holomorphic functions $\theta(z, \tau)$ on $\mathbb{C} \times \mathbb{H}$ satisfying the heat equation and the lattice-quasiperiodicity laws. It depends on $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ (chapter 68) and $\mathsf{NameCert}_{\mathsf{EllipticCurve}^{\uparrow}}$ (chapter 175).

251.1 Carrier definition

Definition 251.1 (ThetaFunction carrier source)**.** *A* $\mathsf{ThetaFunction}^{\uparrow}$ *carrier source is a finite* BHist *packet* $\Theta = (E, H, Q, \Pi, C)$ *whose rows consist of an* $\mathsf{EllipticCurve}^{\uparrow}$ *period ledger* E*, a* $\mathsf{Holomorphic}^{\uparrow}$ *local-function row* H *on the displayed* (z, τ) *chart, a finite* q*-expansion coefficient window* Q *compatible with the* $\mathsf{CplxSer}^{\uparrow}$ *coefficient ledger,* Pkg *provenance* Π *for the elliptic-curve, holomorphic, and complex-series dependencies, and finite* Cont *readback rows* C *tying the period, chart, and coefficient observations to the same packet.*

Lean4 (definition) `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrier`

Definition 251.2 (ThetaFunction classifier)**.** *Two carrier sources* $\Theta = (E, H, Q, \Pi, C)$ *and* $\Theta' = (E', H', Q', \Pi', C')$ *are classified by the* $\mathsf{ThetaFunction}^{\uparrow}$ *source classifier when the* $\mathsf{EllipticCurve}^{\uparrow}$ *period rows are transported by the same* hsame*-stable period comparison, the finite coefficient windows* Q *and* Q' *agree under the displayed* $\mathsf{CplxSer}^{\uparrow}$ *coefficient-window classifier, the* $\mathsf{Holomorphic}^{\uparrow}$ *chart rows* H *and* H' *agree on the same local-function source, and the* Pkg *and* Cont *provenance rows are componentwise the displayed rows for those transports.*

Lean4 (definition) `BEDC.Derived.ThetaFunctionUp.ThetaFunctionSourceClassifier`
Lean4 (definition) `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource classifier`

251.2 The certificate

Theorem 251.3 (ThetaFunction hsame stability)**.** *The classifier of Definition 251.2 is stable under rowwise* hsame *transport of the period ledger, coefficient window, holomorphic chart row,* Pkg *provenance, and* Cont *readback rows.*

Proof. Unpack the classifier into its displayed coordinates. The period coordinate is transported by the $\mathsf{EllipticCurve}^{\uparrow}$ period row, the chart coordinate by the $\mathsf{Holomorphic}^{\uparrow}$ local-function row, and the coefficient coordinate by the $\mathsf{CplxSer}^{\uparrow}$ window row.

Transport each coordinate along the given hsame witness and keep the same Cont readback route and Pkg provenance. Repacking these transported rows gives a classifier witness for the target packet, with no appeal to host function equality or analytic continuation. □

Lean4 ✓ `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrier hsame stability`

Theorem 251.4 (ThetaFunction ledger exactness)**.** *Every accepted* ThetaFunction$^{\uparrow}$ *carrier source exposes exactly the* EllipticCurve$^{\uparrow}$ *period rows, finite q-expansion coefficient rows,* Holomorphic$^{\uparrow}$ *chart rows,* Pkg *provenance, and* Cont *readback rows of Definition 251.1. It exposes no analytic continuation datum, completed modular form, host holomorphic-function equality, or infinite coefficient object.*

Lean4 ✓ `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource ledger exactness`

Proof. Project an accepted carrier source to the five components in Definition 251.1. These projections are the period ledger, holomorphic chart, coefficient window, provenance, and finite readback rows.

Conversely, any ledger read admitted by the carrier must enter through one of those five coordinates. The EllipticCurve$^{\uparrow}$, Holomorphic$^{\uparrow}$, and CplxSer$^{\uparrow}$ dependency anchors account for the mathematical rows, while Pkg and Cont account for provenance and finite readback.

There is no constructor field for analytic continuation, completed modular forms, host equality of functions, or an infinite coefficient object. Therefore none of those objects is exported by the ledger. □

Theorem 251.5 (ThetaFunction NameCert boundary)**.** *The carrier source of Definition 251.1, the classifier of Definition 251.2, the stability theorem Definition 251.3, and the ledger exactness theorem Definition 251.4 assemble the* ThetaFunction$^{\uparrow}$ NameCert *boundary over* BHist, hsame, Cont, *and* Pkg. *The boundary consumes only the displayed* EllipticCurve$^{\uparrow}$, Holomorphic$^{\uparrow}$, *and* CplxSer$^{\uparrow}$ *source rows and supplies exactly the finite period, chart, and coefficient observation surface.*

Lean4 ✓ `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource namecert boundary`

Proof. The carrier definition supplies the source object for the certificate. The classifier definition supplies the comparison relation, and Definition 251.3 gives transport of accepted rows along the componentwise hsame comparisons.

The exact ledger is supplied by Definition 251.4. Its projections are precisely the period, chart, coefficient, provenance, and readback rows required by the displayed NameCert surface.

Combining these rows gives the advertised boundary: a finite BHist packet with dependency provenance through Pkg, readback through Cont, and no additional analytic-continuation or completed-modular-form field. □

Theorem 251.6 (ThetaFunction modular-shift ledger)**.** *Let* Θ *be a* ThetaFunction$^{\uparrow}$ *packet whose* CplxSer$^{\uparrow}$ *source row is transported by a certified integer or unary period-shift row. The transported packet preserves the displayed q-expansion carrier, classifier, and ledger rows through the same* hsame *comparisons,* Cont *provenance, and* Pkg *entries. The shift is recorded as a finite package-ledger row rather than as an external modular-group action.*

Lean4 ✓ `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource modular shift ledger`

Proof. Inspect the carrier packet of Definition 251.1. The period row, holomorphic chart row, coefficient window, provenance row, and readback row are the only available coordinates.

Use the $\mathsf{CplxSer}^{\uparrow}$ pointwise append and transport rows on each finite coefficient window touched by the certified shift. The $\mathsf{Holomorphic}^{\uparrow}$ source row supplies stability for the analytic carrier coordinate, while the $\mathsf{EllipticCurve}^{\uparrow}$ period ledger supplies the displayed period source.

Repack the transported coefficient, period, chart, continuation, and package rows. Since the shift appears only as a Pkg ledger entry, the resulting packet preserves the carrier, classifier, and ledger without exporting a modular-group action. □

Theorem 251.7 (ThetaFunction period-lattice scope)**.** *Every consumer-visible period-lattice readback of a* $\mathsf{ThetaFunction}^{\uparrow}$ *packet factors through the* $\mathsf{CplxSer}^{\uparrow}$ *coefficient stream, the* $\mathsf{EllipticCurve}^{\uparrow}$ *period source, the* $\mathsf{Holomorphic}^{\uparrow}$ *carrier row, displayed* hsame *comparisons, the* Cont *route, and the* Pkg *ledger of Definition 251.1. The packet exports no analytic continuation object, quotient torus, host lattice object, or completed infinite coefficient stream.*

Lean4 ✓ `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource period lattice scope`

Proof. Begin with the carrier source, classifier, hsame stability, and ledger exactness rows. These rows already identify the finite period ledger, local holomorphic chart, coefficient window, provenance, and readback coordinates.

For a period-lattice readback, first project the $\mathsf{EllipticCurve}^{\uparrow}$ period source and the $\mathsf{CplxSer}^{\uparrow}$ coefficient stream rows. Then transport the chart and coefficient windows along the displayed hsame comparisons and follow the recorded Cont route.

The NameCert boundary of Definition 251.5 exhausts the consumer-visible surface. Since the boundary has no constructor field for analytic continuation, quotient tori, host lattices, or completed coefficient streams, none of those objects is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{ThetaFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ThetaFunction}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The seed scope names the displayed $\mathsf{EllipticCurve}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{CplxSer}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows of the theta-function packet.

Constructive story: A theta-function packet records finite period, holomorphic chart, coefficient-window, continuation, and package rows as its visible certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ThetaFunction}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ThetaFunctionUp.ThetaFunctionCarrierSource\_namecert\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic continuation, quotient tori, completed modular forms, host lattice objects, or standard bridges.

Upgrade path: Bridge closure requires a standard bridge boundary for the displayed theta-function public packet.

Theorem 251.8 (ThetaFunction coefficient-prefix restriction carrier)**.** *Let* $\Theta = (E, H, Q, \Pi, C)$ *be an accepted* $\mathsf{ThetaFunction}^{\uparrow}$ *carrier source of Definition 251.1. Let* $Q_{\preceq n}$ *be a displayed finite prefix window of the same q-expansion coefficient ledger. Suppose the* $\mathsf{CplxSer}^{\uparrow}$ *coefficient-window classifier accepts* $Q_{\preceq n}$ *under the same coefficient source that accepts* Q*, and suppose the displayed prefix readback is one of the finite coefficient observations already routed by the* Cont *rows in* C*. Then*

$$\Theta_{\preceq n} = (E, H, Q_{\preceq n}, \Pi, C)$$

is again an accepted $\mathsf{ThetaFunction}^{\uparrow}$ *carrier source. The replacement keeps the period ledger, holomorphic chart row, package provenance, and continuation rows fixed, and changes only the finite coefficient-window coordinate from* Q *to the classified prefix* $Q_{\preceq n}$*.*

Proof. Invert the accepted carrier source by Definition 251.1. Its exposed rows are exactly the period ledger E, the holomorphic local-function row H, the finite coefficient window Q, the package provenance Π, and the finite readback rows C tying the period, chart, and coefficient observations to the same packet.

The prefix hypothesis supplies a finite coefficient row $Q_{\preceq n}$ over the same displayed $\mathsf{CplxSer}^{\uparrow}$ coefficient ledger. Since it is accepted by the same coefficient-window classifier as Q, replacing the coefficient coordinate does not introduce a different coefficient source, a completed series, or an unrecorded equality of analytic functions. The coefficient row remains a finite window row in the exact sense required by the carrier definition.

The period coordinate E and the chart coordinate H are not altered. The package row Π is also unchanged, because the dependency provenance still names the same elliptic-curve, holomorphic, and complex-series sources. The continuation row C remains valid by the displayed prefix-readback assumption: its period and chart components are identical, and its coefficient component already routes the prefix observation $Q_{\preceq n}$ as a finite readback under the same packet.

Repacking the five displayed coordinates gives $(E, H, Q_{\preceq n}, \Pi, C)$, which satisfies the carrier clauses of Definition 251.1. Finally, Definition 251.4 shows that this operation has no hidden effect on the public ledger: the result still exposes only the elliptic-curve period rows, the holomorphic chart row, the finite coefficient window, Pkg provenance, and Cont readbacks, with no analytic continuation datum, completed modular form, host holomorphic-function equality, or infinite coefficient object. □

Chapter 252

A Concrete Naming Certificate for $\mathsf{ModularForm}^{\uparrow}$

The modular-form interface $\mathsf{ModularForm}^{\uparrow}$ packages a holomorphic function on the upper half plane invariant under a congruence subgroup of $SL_2(\mathbb{Z})$ up to a fixed weight factor. It depends on $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ (chapter 68) and $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ (chapter 104).

252.1 Carrier definition

Definition 252.1 (Modular-form q-expansion carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ *source row and a* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *source row for the same displayed weight and congruence-subgroup context. A* $\mathsf{ModularForm}^{\uparrow}$ *carrier row is the finite* BHist *ledger consisting of the holomorphic source answer row, the automorphic transformation answer row, a finite q-expansion coefficient observation row, and the* Cont *witnesses that place the coefficient observations under both sources.*

The carrier records finite q-expansion and transformation data only. It does not assert analytic continuation, dimension formulas, Hecke operators, cusp expansions at every cusp, or a host space of modular forms.

Lean4 (definition) `BEDC.Derived.ModularFormUp.ModularFormQExpansionCarrier`

Definition 252.2 (Modular-form congruence classifier)**.** *Two rows of Definition 252.1 are classified as the same* $\mathsf{ModularForm}^{\uparrow}$ *observation exactly when their holomorphic source rows are transported by* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$*, their automorphic transformation rows are transported by* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$*, and their visible finite q-expansion coefficient packages are matched by* hsame*,* Cont*, and* Pkg*.*

The classifier is generated by source transport, finite coefficient package transport, and congruence-subgroup transformation transport. It contains no generator for analytic continuation, dimension counting, or Hecke stability.

252.2 The certificate

Theorem 252.3 (Modular-form holomorphic source scope)**.** *Every* $\mathsf{ModularForm}^{\uparrow}$ *carrier row contains one* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ *source row, one* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *transformation source row, and finite q-expansion coefficient answer histories attached to those sources by* Cont *and* Pkg *witnesses.*

Lean4 ✓ `BEDC.Derived.ModularFormUp.ModularFormQExpansionCarrier holomorphic source scope`

Proof. Invert Definition 252.1. The holomorphic source and automorphic source are explicit fields, while the coefficient data are finite answer rows recorded under those fields.

The classifier of Definition 252.2 can change representatives only through the two source certificates and the displayed finite coefficient package witnesses. Thus the source scope is exactly the holomorphic row, the automorphic row, and the finite q-expansion ledger. □

Theorem 252.4 (Modular-form automorphic transport stability)**.** *If two* $\mathsf{ModularForm}^{\uparrow}$ *rows are classified by Definition 252.2, then transporting the automorphic transformation row through* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *and transporting any visible q-expansion coefficient row through* hsame*,* Cont*, or* Pkg *preserves the classified carrier row.*

Lean4 ✓ `BEDC.Derived.ModularFormUp.ModularFormAutomorphicTransport stability`

Proof. Transport the automorphic source first. The $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ classifier supplies the new transformation answer row, and the carrier retains the same displayed weight and congruence-subgroup context.

For the coefficient row, use the stored hsame endpoint comparison and the Pkg package comparison, with Cont placing the transported coefficient back under the holomorphic and automorphic source rows. Since the coefficient list is finite, concatenating these transports yields a classifier witness for the whole row. □

Theorem 252.5 (Modular-form NameCert obligation surface)**.** *The* $\mathsf{ModularForm}^{\uparrow}$ *chapter exports a NameCert obligation surface consisting of the finite q-expansion carrier, the congruence classifier, the holomorphic and automorphic source rows, and ledger exactness for the displayed coefficient and transformation observations.*

Lean4 ✓ `BEDC.Derived.ModularFormUp.ModularFormQExpansionCarrier semantic name certificate`

Proof. Assemble the surface from a carried row and a classifier witness. The carried row supplies the holomorphic source, the automorphic source, and the finite q-expansion observations; the classifier supplies only source transport, coefficient transport, and transformation transport.

Ledger exactness follows by inverting those constructors. Every accepted field is a source row, a coefficient answer, a transformation answer, or a named hsame, Cont, or Pkg witness. Therefore the obligation surface stays within finite BEDC data and makes no analytic-continuation, dimension-formula, or Hecke-theoretic claim. □

Lean4 ✓ `BEDC.Derived.ModularFormUp.ModularFormQExpansionCarrier namecert obligation surface`

Closure status for $\mathsf{NameCert}_{\mathsf{ModularForm}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModularForm}^{\uparrow}}$, publicClosed)

Scope closed: The obligation surface records the finite q-expansion carrier of Definition 252.1, the congruence classifier of Definition 252.2, the $\mathsf{Holomorphic}^{\uparrow}$ and $\mathsf{Automorphic}^{\uparrow}$ source rows, and the BHist, hsame, Cont, and Pkg rows exposed by Definition 252.5.

Constructive story: A modular-form packet consumes one $\mathsf{Holomorphic}^{\uparrow}$ source row and one $\mathsf{Automorphic}^{\uparrow}$ source row over the same displayed weight and congruence-subgroup context, then attaches a finite q-expansion coefficient ledger through Cont, Pkg, and hsame rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModularForm}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ModularFormUp.ModularFormAutomorphicTransport\_stability`

Bridge status: none

Not claimed: This obligation boundary does not close analytic continuation, dimension formulas, Hecke operators, all-cusp expansions, or a host modular-form space.

Upgrade path: The next paper-grade step is a bridge boundary for the public q-expansion and automorphic-source package.

Theorem 252.6 (Modular-form empty coefficient classifier)**.** *Let R and R' be two* $\mathsf{ModularForm}^{\uparrow}$ *carrier rows of Definition 252.1 over the same displayed weight and congruence-subgroup context. Suppose that both rows use the same* $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ *source row H, the same* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *source row A, and the visible q-expansion coefficient observation row is* e *in both carriers. Then R and R' are classified as the same* $\mathsf{ModularForm}^{\uparrow}$ *observation by the modular-form congruence classifier of Definition 252.2.*

Lean4 ✓ `BEDC.Derived.ModularFormUp.ModularFormQExpansionCarrier empty coefficient classifier`

Proof. Unpack the two carrier rows by Definition 252.1. Their source part is the common pair (H, A), so the holomorphic source component is transported by the reflexive classifier row of $\mathsf{NameCert}_{\mathsf{Holomorphic}^{\uparrow}}$ and the automorphic component is transported by the reflexive classifier row of $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$. The displayed weight and congruence-subgroup context is also common, so no context replacement is required.

It remains only to compare the finite q-expansion coefficient ledger. Both visible coefficient rows are e. The empty history supplies the endpoint comparison $\mathsf{ehsamee}$, and the right-unit continuation rows place the empty coefficient observation under the already shared holomorphic and automorphic source rows:

$$\mathsf{Cont}(H, \mathsf{e}, H) \qquad \text{and} \qquad \mathsf{Cont}(A, \mathsf{e}, A).$$

The finite coefficient package therefore has no nonempty coefficient entry to match. Its package comparison is the empty package comparison, with the displayed hsame and Cont witnesses as its whole ledger.

The generators listed in Definition 252.2 are exactly source transport, finite coefficient package transport, and congruence-subgroup transformation transport. Applying the two reflexive source transports, the empty coefficient package comparison, and the shared automorphic transformation context gives a classifier witness from R to R'. Since no generator for analytic continuation, dimension counting, Hecke stability, or all-cusp data participates in that classifier, the conclusion is confined to the two empty-coefficient carrier rows. □

Chapter 253

A Concrete Naming Certificate for $\mathsf{ClebschGordan}^{\uparrow}$

The Clebsch-Gordan interface $\mathsf{ClebschGordan}^{\uparrow}$ packages, for a pair of irreducible representations of a compact Lie group, the decomposition of their tensor product into irreducibles together with the explicit coupling coefficients. It depends on $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140) and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87).

253.1 Carrier definition

253.2 The certificate

Definition 253.1 (ClebschGordanUp BHist coupling packet)**.** *A* $\mathsf{ClebschGordan}^{\uparrow}$ *coupling packet is a finite* BHist *tuple*

$$\mathcal{C} = (G, T, R, V_1, V_2, \tau, D, \Gamma, \theta, P, \lambda).$$

The rows G, T, and R are consumed certificates $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$. *The rows V_1 and V_2 are the two irreducible source-representation histories, each read through G and R. The row τ is the tensor-product endpoint history for $V_1 \otimes V_2$ read through T.*

The finite decomposition ledger D lists irreducible output histories W_j with multiplicity histories m_j and representation-ring summand rows in R. The coefficient ledger Γ lists the displayed coupling-coefficient histories $\gamma^{j,q}_{a,b}$ that relate source basis histories for V_1 and V_2 to output basis histories for each W_j and multiplicity slot q. The row θ is the componentwise hsame *transport datum for $G, T, R, V_1, V_2, \tau, D, \Gamma$.*

The row P is the Pkg *provenance recording exactly the consumed LieGroupUp, TensorProductUp, RepresentationRingUp, decomposition, and coefficient rows. The finite ledger λ is the* Cont *route containing only those displayed rows and their endpoint readbacks.*

Theorem 253.2 (ClebschGordanUp carrier source obligation)**.** *Every public carrier row of a* $\mathsf{ClebschGordan}^{\uparrow}$ *coupling packet is one of the displayed rows of Definition 253.1. In particular, the two irreducible source rows are read only through* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$, *and the tensor endpoint is read only through* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ *together with those two source rows.*

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket carrier source obligation`

Proof. Project an accepted packet to the tuple $\mathcal{C} = (G, T, R, V_1, V_2, \tau, D, \Gamma, \theta, P, \lambda)$. The carrier definition lists all public rows in that tuple, so no additional carrier row can be read by a downstream consumer.

The two source rows V_1 and V_2 carry their group action through G and their representation-ring membership through R. The tensor endpoint τ is then the row supplied by T on those same two source histories.

Repacking these projections gives the carrier obligation: source irreducibles, tensor endpoint, decomposition entries, coefficients, transport, provenance, and continuation are all BHist rows of the same displayed packet. □

Theorem 253.3 (ClebschGordanUp classifier transport obligation)**.** *Let $\mathcal{C}$ and $\mathcal{C}'$ be two Definition 253.1 packets. A public* $\mathsf{ClebschGordan}^{\uparrow}$ *classifier row from $\mathcal{C}$ to $\mathcal{C}'$ is the componentwise* hsame *transport of the consumed* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *rows, the two irreducible source rows, the tensor endpoint row, the finite decomposition ledger, the coefficient histories, the* Pkg *provenance row, and the finite* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket classifier transport obligation`

Proof. Unpack the transport row θ in the source packet and the corresponding row θ' in the target packet. Their displayed fields compare the three consumed certificates, the two source irreducibles, and the tensor endpoint by hsame.

The same componentwise comparison transports the decomposition ledger D: each output irreducible history, multiplicity history, and representation-ring summand row is matched with its transported target row. The coefficient ledger Γ is transported entry by entry over the same source and output histories.

Finally compare P and λ by the displayed provenance and continuation rows. These projections exhaust the public classifier, so no host equality of representations, basis vectors, or coefficient arrays is available outside the packet. □

Theorem 253.4 (ClebschGordanUp ledger exactness obligation)**.** *The exact ledger of a* $\mathsf{ClebschGordan}^{\uparrow}$ *coupling packet consists precisely of the tensor endpoint τ, the finite decomposition ledger D, the coefficient-history ledger Γ, the componentwise* hsame *transport θ, the provenance packet P, and the finite* Cont *route λ. Every accepted coupling read projects to these rows, and every family of these rows repacks as the public coupling ledger.*

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket ledger exactness obligation`

Proof. For the projection direction, take an accepted coupling read and inspect its public packet. The tensor product endpoint is the row τ, the finite list of irreducible summands and multiplicities is D, and the coupling coefficients are the histories in Γ.

The transport, provenance, and continuation fields are then the displayed rows θ, P, and λ. Because the packet definition contains no other public field, the projection cannot expose a host decomposition theorem or an unrecorded coefficient table.

For the reverse direction, start with rows of exactly these six forms over the same consumed G, T, R, V_1, V_2 source data. Insert them into the tuple of Definition 253.1. The resulting tuple is the public ledger, so exactness is projection followed by this repacking. □

Theorem 253.5 (ClebschGordanUp source NameCert boundary)**.** *The source boundary of a* $\mathsf{ClebschGordan}^{\uparrow}$ *coupling packet is exactly the triple of consumed certificates* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{RepresentationRing}^{\uparrow}}$ *together with the two irreducible source rows*

and their tensor endpoint. It does not export a separate compact-group carrier, representation category, basis convention, or ambient Hilbert-space object.

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket source namecert boundary`

Proof. Project the dependency rows G, T, R from the packet. The irreducible source histories V_1 and V_2 are scoped by G and R, while the tensor row τ is scoped by T and those same two source histories.

The decomposition ledger and coefficient histories are downstream rows over this source boundary. They refer back to G, T, R, V_1, V_2, τ through Pkg provenance and through the displayed Cont route.

Since no other source row occurs in the tuple, any consumer requesting a Clebsch-Gordan source can receive only those certificates and histories. Thus the boundary is a BEDC source boundary, not a hidden representation theory environment. □

Theorem 253.6 (ClebschGordanUp consumer readback boundary)**.** *Every downstream consumer of a* ClebschGordan$^{\uparrow}$ *coupling packet reads only the two irreducible source histories, the tensor endpoint, the finite decomposition ledger, the coefficient histories, the componentwise* hsame *transport rows, the* Pkg *provenance, and the finite* Cont *readback route of Definition 253.1. If two packets agree on those rows by the public classifier, then every such consumer has the same readback.*

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket consumer readback boundary`

Proof. Let a consumer be invariant under the displayed rows named in the statement. Project both packets to their source histories, tensor endpoint, decomposition ledger, coefficient histories, transport rows, provenance, and continuation route.

The public classifier supplies componentwise hsame comparisons for those projections. The consumer therefore sees the same source inputs, the same tensor endpoint, the same finite decomposition entries, and the same coefficient histories after transport.

By the ledger exactness obligation of Definition 253.4, there is no further public row that could affect the consumer. Repacking the transported projections gives identical readback at the consumer boundary. □

Theorem 253.7 (ClebschGordanUp coefficient slot determinacy)**.** *Let two accepted reads of coefficient histories in the coefficient ledger Γ of a* ClebschGordan$^{\uparrow}$ *coupling packet have the same transported source rows V_1 and V_2, the same tensor endpoint τ, the same output irreducible row W_j, the same multiplicity history m_j, and the same source and output basis-index histories. Then the public classifier identifies the two coefficient rows by the componentwise* hsame *transport field of the coupling packet of Definition 253.1.*

Equivalently, a displayed coefficient history accepted in Γ belongs to a unique public slot determined by $(V_1, V_2, \tau, W_j, m_j)$ together with its basis-index histories; no second public coefficient table can be read at that same slot.

Proof. Project the two accepted coefficient reads through Definition 253.1. Each read is a row of the same displayed coefficient ledger Γ, so its source rows, tensor endpoint, output row, multiplicity row, and basis-index histories are among the packet fields.

Use the classifier transport obligation of Definition 253.3. Under the shared slot hypotheses, the public classifier compares the two coefficient rows by the componentwise hsame entry assigned to Γ in the transport row θ.

It remains to exclude a second public coefficient source for that slot. Apply the ledger exactness obligation of Definition 253.4: every accepted coefficient read is projected from Γ, and the packet has no additional coefficient table. Thus the transported Γ entry is the unique public coefficient row at the displayed slot. □

Lean4 ✓ `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket coefficient slot determinacy`

Closure status for $\mathsf{NameCert}_{\mathsf{ClebschGordan}^{\uparrow}}$.

Constructive story: A coupling packet is generated by consuming a LieGroup$^{\uparrow}$ row, a TensorProduct$^{\uparrow}$ row, and a RepresentationRing$^{\uparrow}$ row, then displaying two irreducible source-representation histories, their tensor endpoint, a finite decomposition ledger, coefficient histories, transport data, provenance, and a continuation route. The carrier accepts only finite packets of the form $(G, T, R, V_1, V_2, \tau, D, \Gamma, \theta, P, \lambda)$; no representation category, Hilbert-space basis convention, host decomposition theorem, or external coefficient table is available outside those rows. Two packets classify together by componentwise hsame transport of the three consumed certificates, the two sources, the tensor endpoint, decomposition entries, coefficient slots, provenance, and finite Cont ledger. The source specification is the Lie-group, tensor-product, and representation-ring boundary with the two irreducible source rows, the pattern specification is tensor-product decomposition into irreducible outputs with multiplicities and coupling coefficients, and the ledger keeps D, Γ, θ, P, and λ visible. The five name-certificate obligations are inherited from the three dependency classifiers and from the exactness rows: an accepted packet inhabits the carrier, component transport is reflexive, symmetric, and transitive, and any transported coefficient or decomposition row repacks into the same public coupling ledger. A packet history h lies in the signature gap of a ClebschGordan$^{\uparrow}$ token p when the source-domain accepts its representation-coupling surface, the signature relation pairs it with an answer coupling packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the tensor endpoint, summand ledger, multiplicities, and coefficient slots are exactly those encoded by p. The chapter registers this finite coupling-ledger construction under the name ClebschGordan$^{\uparrow}$; it introduces no full representation-category bridge and no standard Clebsch–Gordan coefficient table.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClebschGordan}^{\uparrow}}$, publicClosed)

Scope closed: The scoped boundary binds the carrier source, classifier transport, ledger exactness, source NameCert boundary, consumer readback boundary, and coefficient slot determinacy obligations for the BHist coupling packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClebschGordan}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ClebschGordanUp.ClebschGordanCouplingPacket\_coefficient\_slot\_determinacy`

Bridge status: none

Not claimed: This chapter does not close a full representation-category bridge or standard Clebsch-Gordan coefficient tables.

Upgrade path: Bridged closure requires a standard coupling bridge over the displayed LieGroup$^{\uparrow}$, TensorProduct$^{\uparrow}$, RepresentationRing$^{\uparrow}$, decomposition, coefficient, transport, provenance, and finite Cont rows.

Chapter 254

A Concrete Naming Certificate for $\mathsf{Knot}^{\uparrow}$

The knot interface $\mathsf{Knot}^{\uparrow}$ packages a diagrammatic BEDC surface whose traditional reading is a closed one-dimensional source carried through an ambient deformation ledger. It depends on $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ (chapter 58) and $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$ (chapter 156); those dependencies supply the endpoint rows and deformation rows, while the knot certificate records only finite BHist, Cont, hsame, and Pkg data.

254.1 Carrier definition

Definition 254.1 (KnotUp diagram BHist carrier)**.** *A* $\mathsf{Knot}^{\uparrow}$ *diagram packet is a finite carried tuple*

$$\mathcal{K} = (\mathcal{S}, \mathcal{A}, d, r, h, e_0, e_1, \rho, \lambda)$$

whose rows are all BEDC rows. The source row $\mathcal{S}$ *is a consumed* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ *packet, read only through its exposed source carrier, component classifier, endpoint readback, and ledger surface. The ambient row* $\mathcal{A}$ *is the target row in which the diagram lives; it is a* BHist *endpoint family classified by the same package provenance as the deformation rows and not by an external host embedding.*

The diagram row d *is a finite crossing ledger. Each crossing entry is a* BHist *face whose incident source endpoints are read from* $\mathcal{S}$ *and whose over-under comparison is a displayed* hsame *or* Cont *row. The row* r *is the Reidemeister trace: a finite* Cont *list whose entries are exactly the local crossing changes, endpoint-preserving moves, and package-provenance updates used to pass from one diagram ledger to the next. The row* h *is the consumed* $\mathsf{Homotopy}^{\uparrow}$ *deformation row, read as an ambient-deformation history compatible with the same endpoints and with the same finite trace.*

The endpoint rows e_0 *and* e_1 *expose the initial and final embedded-source interfaces as* BHist *rows over* $\mathcal{S}$ *and* $\mathcal{A}$*. The row* ρ *records the* Pkg *provenance of the source certificate, ambient target row, diagram ledger, Reidemeister trace, and HomotopyUp deformation row. The finite ledger* λ *is the* Cont *display containing precisely those source, ambient, crossing, move-trace, deformation, endpoint, classifier, provenance, and exactness rows accepted by the public* $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$ *carrier.*

Definition 254.2 (KnotUp Reidemeister ledger classifier)**.** *Two* $\mathsf{Knot}^{\uparrow}$ *diagram packets* $\mathcal{K} = (\mathcal{S}, \mathcal{A}, d, r, h, e_0, e_1, \rho, \lambda)$ *and* $\mathcal{K}' = (\mathcal{S}', \mathcal{A}', d', r', h', e'_0, e'_1, \rho', \lambda')$ *are related by the* $\mathsf{Knot}^{\uparrow}$ *Reidemeister ledger classifier when the following finite rows are supplied:*

1. *the* $\mathsf{S}^{1\uparrow}$ *source packets* $\mathcal{S}$ *and* $\mathcal{S}'$ *are transported by the* hsame *classifier of* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$*;*

2. *the crossing ledgers d and d' are matched entrywise by* hsame *on their incident source endpoint rows and by* Cont *transport on their over-under comparison rows;*
3. *the Reidemeister traces r and r' are related by a finite* Cont *refinement whose entries are again Reidemeister trace rows, not external diagram equivalences;*
4. *the HomotopyUp deformation rows h and h' are related by the ambient-deformation classifier exposed by* $\mathsf{NameCert}_{\mathsf{Homotopy}^\uparrow}$*;*
5. *the endpoint rows e_0, e_1, e_0', e_1' and the provenance rows ρ, ρ' remain in the same* Pkg *scope, and the resulting ledgers λ, λ' list exactly the transported source, diagram, deformation, endpoint, and provenance faces.*

This classifier is componentwise on the visible BEDC rows. It does not identify packets by equality of host embeddings, by a quotient of knots, or by treating ambient isotopy as an undecomposed primitive.

254.2 The certificate

Theorem 254.3 (KnotUp SOne source boundary)**.** *For a* $\mathsf{Knot}^\uparrow$ *diagram packet $\mathcal{K}$ of Definition 254.1, the public source boundary exposed to* $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ *is exactly the* $\mathsf{S}^{1\uparrow}$ *source carrier row $\mathcal{S}$, its component classifier rows, the endpoint rows e_0 and e_1, and the package-provenance row ρ linking those endpoints to the ambient target row. No carrier clause may read an unclassified source point, an external circle object, or a host-level embedding in place of these rows.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket sone source boundary`

Proof. Project $\mathcal{K}$ to its source field $\mathcal{S}$. The source certificate is already a $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ packet, so its carrier, component classifier, endpoint readback, and ledger rows are exactly the rows available to the knot packet.

Next project the endpoint fields e_0 and e_1. By Definition 254.1, each endpoint row is a BHist interface over the source packet and ambient target row, and the provenance field ρ records that these endpoints are part of the same Pkg package.

Conversely, given precisely those source, endpoint, and provenance rows, place them in the displayed fields of $\mathcal{K}$ and leave the diagram, trace, deformation, and ledger rows to the remaining certificate clauses. This reconstructs the public source boundary without adding any host source object. □

Theorem 254.4 (KnotUp Homotopy ledger boundary)**.** *For a* $\mathsf{Knot}^\uparrow$ *diagram packet $\mathcal{K}$ of Definition 254.1, the HomotopyUp and Reidemeister rows exposed to* $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ *are exactly the crossing ledger d, the Reidemeister trace r, the HomotopyUp deformation row h, and the finite* Cont *ledger λ. If two packets are related by the classifier of Definition 254.2, these four rows are stable under the displayed* hsame *and* Cont *transports.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket homotopy ledger boundary`

Proof. First read the crossing ledger d. Its entries name only incident $\mathsf{S}^{1\uparrow}$ endpoint rows, over-under comparison rows, and finite BHist faces. Transporting the source endpoints by hsame therefore transports every crossing entry componentwise.

Now read the Reidemeister trace r and the deformation row h. The trace is a finite Cont list of local moves, while h is the corresponding $\mathsf{Homotopy}^\uparrow$ deformation row. The classifier requires these

rows to be matched by finite trace refinement and by the $\mathsf{NameCert}_{\mathsf{Homotopy}^\uparrow}$ ambient-deformation classifier.

Finally inspect λ. It contains exactly the source, crossing, Reidemeister, deformation, endpoint, provenance, and exactness faces. Since each listed face is transported by the classifier clauses above, the ledger is stable under the same hsame and Cont transports. Projecting a classifier witness gives no additional ambient relation beyond these rows. □

Theorem 254.5 (KnotUp public NameCert obligation surface)**.** *For a* $\mathsf{Knot}^\uparrow$ *diagram packet* $\mathcal{K}$ *of Definition 254.1, the public* $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ *obligation surface is exactly the following finite row family: the* $\mathsf{S}^{1\uparrow}$ *source packet* $\mathcal{S}$*, the ambient target row* $\mathcal{A}$*, the crossing ledger* d*, the Reidemeister trace* r*, the* $\mathsf{Homotopy}^\uparrow$ *deformation row* h*, the endpoint rows* e_0 *and* e_1*, the* Pkg *provenance row* ρ*, the finite* Cont *ledger* λ*, the classifier of Definition 254.2, the source boundary of Definition 254.3, and the Homotopy ledger boundary of Definition 254.4. These rows supply the carrier, classifier, stability, ledger, and exactness fields consumed by downstream knot-facing chapters.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket namecert obligation surface`

Proof. Project the packet to the fields listed in Definition 254.1. The source boundary theorem supplies the exact $\mathsf{S}^{1\uparrow}$ rows and endpoint package provenance. The Homotopy ledger boundary theorem supplies the exact crossing, Reidemeister, deformation, and continuation rows together with their classifier stability.

The classifier field is the relation of Definition 254.2. Its clauses transport the source, diagram, trace, deformation, endpoint, provenance, and ledger rows componentwise, so it is the stability surface required by the public $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ package.

Conversely, every row named in the obligation surface is a field of $\mathcal{K}$ or a projection theorem for those fields. Repacking them gives the carrier, classifier, stability, ledger, and exactness surface visible to downstream consumers. No additional host embedding, external knot quotient, or undecomposed ambient-isotopy witness is part of the public interface. □

Theorem 254.6 (KnotUp projection stability obligation)**.** *Every carried knot projection row over* BHist *is stable under* hsame *transport of the displayed* $\mathsf{S}^{1\uparrow}$ *source row and the ambient* $\mathsf{Homotopy}^\uparrow$ *row. The transported row has the same visible knot packet, and its hidden projection data are carried only through the* Cont *ledger and the two cited NameCert source rows.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket projection stability obligation`

Proof. Read the carried projection row as a BHist packet whose visible boundary consists of the $\mathsf{S}^{1\uparrow}$ source row, the ambient $\mathsf{Homotopy}^\uparrow$ row, and the finite projection ledger. The BEDC source field is therefore $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ together with $\mathsf{NameCert}_{\mathsf{Homotopy}^\uparrow}$; the carrier field is the projection packet; the classifier field is the row displayed by the packet; and the stability field is the given hsame transport.

Transport first along the $\mathsf{S}^{1\uparrow}$ source row and then along the ambient $\mathsf{Homotopy}^\uparrow$ row. The Cont ledger carries exactly the hidden projection witnesses, so no additional diagram equality or ambient quotient is introduced.

The ledger field is the transported finite projection ledger, the core theorem list is this stability row together with the two dependency rows, and exactness says that the accepted packet is precisely the transported packet. The standard bridge field is none, and the status field is a paper obligation for $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$. □

Theorem 254.7 (KnotUp Reidemeister-move classifier obligation)**.** *The* $\mathsf{Knot}^{\uparrow}$ *classifier is generated by finite* ProbeBundle *ledgers for Reidemeister I, II, and III moves, together with endpoint* hsame *transport. Reflexive, symmetric, and transitive classifier rows are supplied respectively by the empty move ledger, the reversed move ledger, and concatenation of finite move ledgers.*

Proof. The BEDC source field is the projection carrier from Definition 254.6 with endpoints classified by $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$ and ambient rows classified by $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$. The carrier field is a finite move ledger whose entries are tagged Reidemeister I, Reidemeister II, or Reidemeister III probes inside ProbeBundle.

For the classifier field, accept a row exactly when such a finite ledger connects the two visible projection endpoints after endpoint hsame transport. Stability follows by transporting the endpoints first and then reusing the same tagged move probes over the transported boundary.

The ledger field records the tagged move list. The core theorem list consists of the empty-ledger row, the reverse-ledger row, and the concatenation row, giving reflexivity, symmetry, and transitivity of the classifier. Exactness states that every accepted classifier row has one of these finite ledger presentations; the standard bridge field is none, and the status field is a paper obligation. □

Theorem 254.8 (KnotUp empty Reidemeister ledger reflexivity)**.** *For every* $\mathsf{Knot}^{\uparrow}$ *diagram packet, the empty finite* ProbeBundle *Reidemeister ledger over its displayed* $\mathsf{S}^1\uparrow$ *endpoints and ambient* $\mathsf{Homotopy}^{\uparrow}$ *row supplies the* $\mathsf{Knot}^{\uparrow}$ *classifier row from the packet to itself.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotReidemeisterLedgerClassifier empty reflexive`

Proof. Project the packet to its visible boundary: the two $\mathsf{S}^1\uparrow$ endpoint rows, the ambient $\mathsf{Homotopy}^{\uparrow}$ row, the packet provenance in Pkg, and the hidden diagram data carried by Cont. These are exactly the endpoint fields read by the classifier obligation of Definition 254.7.

Choose the empty ProbeBundle ledger. It has the same starting and ending endpoint boundary, so the endpoint hsame transport is the identity row and no hidden move witness is introduced.

Applying Definition 254.7 to this boundary and empty ledger gives the accepted classifier row from the displayed packet back to itself. The ledger exactness surface is the empty presentation named in Definition 254.13. □

Theorem 254.9 (KnotUp reversed Reidemeister ledger symmetry)**.** *If a finite* ProbeBundle *Reidemeister ledger over the displayed* $\mathsf{S}^1\uparrow$ *endpoints and ambient* $\mathsf{Homotopy}^{\uparrow}$ *row supplies an accepted* $\mathsf{Knot}^{\uparrow}$ *classifier row from one diagram packet to another, then the reversed finite ledger supplies the opposite classifier row over the same visible endpoint data and the same* Pkg *scope.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotReidemeisterLedgerClassifier reversal symmetry`

Proof. Start with the accepted classifier row and read it through Definition 254.7. The row exposes a finite tagged Reidemeister ledger, endpoint hsame transport, the shared $\mathsf{S}^1\uparrow$ endpoint source rows, and the ambient $\mathsf{Homotopy}^{\uparrow}$ row.

Reverse the finite ledger by reversing the order of its ProbeBundle entries and replacing each Reidemeister probe with the same move read in the opposite direction. The endpoints are unchanged as visible BEDC rows: the former target boundary becomes the source boundary, the former source boundary becomes the target boundary, and the Pkg provenance remains the same package scope.

The reversed ledger is therefore a finite presentation over the same visible endpoints. Applying Definition 254.7 again gives the opposite classifier row, and Definition 254.13 records that no extra ambient-isotopy witness is admitted beyond that finite reversal. □

Theorem 254.10 (KnotUp concatenated Reidemeister ledger transitivity)**.** *If one accepted* $\mathsf{Knot}^{\uparrow}$ *classifier row is carried by a finite Reidemeister ledger from a first diagram packet to a middle packet, and a second accepted row is carried by a finite Reidemeister ledger from that middle packet to a third packet, then the concatenated* ProbeBundle *ledger supplies a classifier row from the first packet to the third.*

Proof. Read the two accepted rows through Definition 254.7. The first row supplies a finite ledger from the first boundary to the middle boundary, and the second supplies a finite ledger from the middle boundary to the third boundary.

Use the Cont and Pkg provenance of the middle packet to identify the two displayed middle boundaries as the same BEDC boundary data. The shared $\mathsf{S}^{1\uparrow}$ endpoint rows and the shared ambient $\mathsf{Homotopy}^{\uparrow}$ row give the endpoint compatibility needed before the ledgers can be joined.

Append the two finite ProbeBundle ledgers. The appended ledger starts at the first packet boundary, passes through the shared middle boundary, and ends at the third packet boundary, while preserving every tagged Reidemeister probe already accepted in the two input rows.

Applying Definition 254.7 to the appended ledger gives the composite classifier row. Exactness is inherited from Definition 254.13: the accepted composite row is represented precisely by the concatenated finite Reidemeister ledger over the displayed endpoints. □

Theorem 254.11 (KnotUp Homotopy source obligation)**.** *Every ambient-isotopy row consumed by the* $\mathsf{Knot}^{\uparrow}$ *certificate factors through the displayed* $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$ *source, and it consumes only endpoints supplied by the* $\mathsf{S}^{1\uparrow}$ *carrier boundary.*

Proof. The BEDC source field is the pair of source certificates $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$. The carrier field is the knot packet whose source boundary exposes the circle endpoints and whose ambient boundary exposes the homotopy row.

Factor the ambient row through $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$ before it is used by any Reidemeister ledger. The endpoint checks are then read from the $\mathsf{S}^{1\uparrow}$ carrier boundary, so the homotopy row cannot introduce a new endpoint source or an untracked ambient endpoint.

The classifier and stability fields are inherited from the factorized homotopy row and the endpoint hsame transport. The ledger field records only the homotopy source witness and the endpoint boundary witnesses; exactness says that these are the only inputs consumed by the ambient-isotopy row. The standard bridge field is none, and the status field is a paper obligation. □

Theorem 254.12 (KnotUp SOne boundary ledger)**.** *Every diagram endpoint, arc endpoint, and crossing endpoint used by a* $\mathsf{Knot}^{\uparrow}$ *packet has an* $\mathsf{S}^{1\uparrow}$ *boundary source. The boundary ledger records the hidden projection and crossing data while exposing only the* $\mathsf{S}^{1\uparrow}$ *endpoint rows and the ambient* $\mathsf{Homotopy}^{\uparrow}$ *row.*

Proof. The BEDC source field is the displayed $\mathsf{S}^{1\uparrow}$ boundary together with the ambient $\mathsf{Homotopy}^{\uparrow}$ boundary. The carrier field is the knot packet with diagram endpoints, arc endpoints, crossing endpoints, hidden projection data, and hidden crossing data.

For each endpoint kind, choose its boundary witness from the $\mathsf{S}^{1\uparrow}$ carrier row. The classifier field then reads endpoint compatibility through those source witnesses, while the stability field transports them by hsame and leaves hidden projection and crossing witnesses inside the finite ledger.

The ledger field is the list of endpoint witnesses plus the hidden projection and crossing records. The core theorem list is the diagram-endpoint, arc-endpoint, and crossing-endpoint boundary rows.

Exactness states that no endpoint used by the packet is outside the $\mathsf{S}^{1\uparrow}$ source boundary; the standard bridge field is none, and the status field is a paper obligation. □

Theorem 254.13 (KnotUp ambient-isotopy ledger exactness)**.** *If two knot packets are related by the* $\mathsf{Knot}^{\uparrow}$ *classifier, then they have compatible visible* $\mathsf{S}^{1\uparrow}$ *endpoints and compatible visible* $\mathsf{Homotopy}^{\uparrow}$ *endpoints, and a finite Reidemeister ledger over those same visible endpoints supplies the accepted classifier row.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket ambient isotopy ledger exactness`

Proof. The BEDC source field is the pair of endpoint sources from $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$. The carrier field is the two knot packets together with their displayed boundaries, and the classifier field is the accepted finite Reidemeister ledger from Definition 254.7.

Project the accepted classifier row to its endpoint boundary. The $\mathsf{S}^{1\uparrow}$ projection gives compatible circle endpoints, and the $\mathsf{Homotopy}^{\uparrow}$ projection gives compatible ambient endpoints; both projections use the same visible endpoints as the finite Reidemeister ledger.

The stability field is endpoint hsame transport, and the ledger field is the shared finite Reidemeister ledger. The core theorem list is projection stability, the Reidemeister classifier, homotopy source factorization, and the $\mathsf{S}^{1\uparrow}$ boundary ledger. Exactness says that the accepted classifier row contains exactly those visible endpoints and that ledger; the standard bridge field is none, and the status field is a paper obligation. □

Theorem 254.14 (KnotUp endpoint source exhaustion)**.** *For every* $\mathsf{Knot}^{\uparrow}$ *diagram packet* $\mathcal{K}$ *of Definition 254.1, every endpoint row appearing as a diagram endpoint, arc endpoint, crossing endpoint, or Reidemeister-move endpoint factors through the displayed* $\mathsf{S}^{1\uparrow}$ *source row and the ambient* $\mathsf{Homotopy}^{\uparrow}$ *row. The public packet has no host embedding endpoint source and no undecomposed ambient-isotopy endpoint source outside those two rows.*

Proof. Project $\mathcal{K}$ to its source field $\mathcal{S}$, ambient field $\mathcal{A}$, endpoint rows e_0, e_1, Reidemeister trace r, HomotopyUp row h, provenance row ρ, and finite ledger λ as displayed in Definition 254.1. These projections are the only places where endpoint data enter the packet.

For diagram endpoints, arc endpoints, and crossing endpoints, apply Definition 254.12. It supplies an $\mathsf{S}^{1\uparrow}$ boundary source for each endpoint kind and leaves hidden projection or crossing data inside the finite ledger rather than in a separate source object.

For endpoints consumed by Reidemeister moves, first read the finite $\mathsf{ProbeBundle}$ move ledger from Definition 254.7. Then apply Definition 254.11 to the ambient row used by that ledger. The ambient row factors through $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$ and consumes only endpoints already supplied by the $\mathsf{S}^{1\uparrow}$ carrier boundary.

Thus each endpoint row factors through the displayed $\mathsf{S}^{1\uparrow}$ and $\mathsf{Homotopy}^{\uparrow}$ sources, with package provenance recorded by ρ and with all remaining witnesses carried by λ. No host embedding source or primitive ambient-isotopy source remains available to the public packet. □

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket endpoint source exhaustion`

Theorem 254.15 (KnotUp Reidemeister ledger composition closure)**.** *Let two accepted finite* $\mathsf{ProbeBundle}$ *Reidemeister ledgers over* $\mathsf{Knot}^{\uparrow}$ *packets have matching visible endpoint rows after the endpoint* hsame *transports required by Definition 254.2. Their* Cont *concatenation is again an accepted* $\mathsf{Knot}^{\uparrow}$ *classifier ledger over the same public source and ambient rows, and the empty ledger is the unit accepted classifier row.*

Proof. Apply Definition 254.7 to the first accepted ledger. It presents the classifier row as a finite ProbeBundle list whose entries are Reidemeister I, II, or III moves together with endpoint hsame transport.

Apply the same theorem to the second accepted ledger. The hypothesis that the visible endpoints match supplies the endpoint transport needed to place the second list after the first one without changing the displayed $\mathsf{S}^{1\uparrow}$ and $\mathsf{Homotopy}^{\uparrow}$ boundary rows.

Concatenate the two finite lists by the Cont ledger operation. Each entry is still one of the accepted Reidemeister probe rows, and the middle endpoint is discharged by the supplied hsame transport. Therefore Definition 254.2 accepts the concatenated ledger as a classifier row.

For the unit, take the empty finite ProbeBundle list. The empty ledger has the same visible endpoints on both sides and is the reflexive row named by Definition 254.7, so it is accepted and acts as the Cont unit. □

Lean4 ✓ `BEDC.Derived.KnotUp.KnotReidemeisterLedgerClassifier composition closure`

Theorem 254.16 (KnotUp classifier ledger completeness)**.** *Every accepted* $\mathsf{Knot}^{\uparrow}$ *classifier row is exactly a finite ledger generated by endpoint* hsame *transport and Reidemeister I, II, and III* ProbeBundle *rows. Its source and ambient endpoints are the visible endpoints of Definition 254.13, and every finite ledger with those generators repacks as an accepted classifier row of Definition 254.2.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotReidemeisterLedgerClassifier ledger completeness`

Proof. For the forward direction, start from an accepted classifier row. Project it through Definition 254.13; the result has compatible visible $\mathsf{S}^{1\uparrow}$ endpoints, compatible visible $\mathsf{Homotopy}^{\uparrow}$ endpoints, and a finite Reidemeister ledger over those same endpoints.

Next apply Definition 254.7. The accepted ledger is generated by endpoint hsame transport and Reidemeister I, II, and III ProbeBundle rows. The exhaustion result Definition 254.14 identifies the endpoint sources with the displayed $\mathsf{S}^{1\uparrow}$ and $\mathsf{Homotopy}^{\uparrow}$ rows.

For the reverse direction, assume such a finite generated ledger over the visible source and ambient endpoints. Place the endpoint transports, move probes, provenance row, and continuation ledger into the clauses of Definition 254.2.

The classifier definition accepts exactly those component rows, so the repacked finite ledger is an accepted $\mathsf{Knot}^{\uparrow}$ classifier row. This gives both directions and leaves no additional quotient or ambient-isotopy primitive in the classifier surface. □

Theorem 254.17 (KnotUp NameCert field coverage)**.** *The public* $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$ *carrier, classifier, stability, ledger, and exactness fields are covered by Definition 254.1, Definition 254.2, Definition 254.3, Definition 254.4, Definition 254.5, Definition 254.6, Definition 254.7, Definition 254.11, Definition 254.12, Definition 254.13, Definition 254.14, Definition 254.15, and Definition 254.16. Downstream consumers may cite these rows without assuming a knot quotient, host embedding equality, or external ambient-isotopy primitive.*

Proof. Enumerate the public certificate surface in the ten-field roadmap form. The BEDC source field is the pair of consumed $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$ packets; the carrier field is Definition 254.1; and the classifier field is Definition 254.2.

The stability field is supplied by Definition 254.6 and by the endpoint hsame transport in Definition 254.7. The ledger field is the finite Cont and ProbeBundle surface identified by Definition 254.4, Definition 254.15, and Definition 254.16.

The source-boundary and endpoint-coverage fields are Definition 254.3, Definition 254.11, Definition 254.12, and Definition 254.14. The exactness field is Definition 254.13 together with the classifier completeness theorem. The standard bridge field is none, and the status field is the public paper surface described by Definition 254.5.

Every item in the roadmap is therefore a displayed BEDC row or a theorem about those displayed rows. Repacking those rows gives the public $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ coverage surface and supplies downstream knot-facing work with citations that do not import a knot quotient, host embedding equality, or external ambient-isotopy primitive. □

Definition 254.18 (KnotUp downstream consumer boundary packet)**.** *For an accepted* $\mathsf{Knot}^\uparrow$ *diagram packet* $\mathcal{K}$ *of Definition 254.1, its downstream consumer boundary packet is the finite BEDC row family*

$$\mathsf{KnotConsumer}(\mathcal{K})$$

consisting of the projected $\mathsf{S}^{1\uparrow}$ *source packet* $\mathcal{S}$ *and its endpoint rows, the ambient* $\mathsf{Homotopy}^\uparrow$ *target and deformation rows, the two endpoint* BHist *rows, the finite Reidemeister* ProbeBundle *ledger, the finite* Cont *continuation ledger, and the* Pkg *provenance row linking all of these rows in the same package scope. The packet exposes only these rows and the public classifier surface of Definition 254.2.*

Theorem 254.19 (KnotUp consumer boundary extraction)**.** *Every accepted* $\mathsf{Knot}^\uparrow$ *diagram packet projects to exactly the downstream consumer boundary packet of Definition 254.18. The projection contains the visible* $\mathsf{S}^{1\uparrow}$ *source rows, ambient* $\mathsf{Homotopy}^\uparrow$ *rows, endpoint family, finite Reidemeister* ProbeBundle *ledger, finite* Cont *ledger, and* Pkg *provenance, and it contains no additional host embedding equality, knot quotient, or undecomposed ambient-isotopy primitive.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket consumer boundary extraction`

Proof. Start with an accepted diagram packet $\mathcal{K}$ and project its carrier fields through Definition 254.1. This gives the displayed source packet, ambient packet, endpoint rows, crossing ledger, Reidemeister trace, deformation row, continuation ledger, and provenance row.

The certificate field coverage theorem Definition 254.17 identifies these same carrier and classifier fields as the public $\mathsf{NameCert}_{\mathsf{Knot}^\uparrow}$ surface. The source and ambient boundary theorems then place the visible source and deformation rows in the same finite row family.

Repack the projected rows in the order named by Definition 254.18. Since every listed component is a carrier field or a field-coverage projection, the extracted family is exactly the consumer boundary packet and no hidden host datum is available to add. □

Theorem 254.20 (KnotUp consumer classifier determinacy)**.** *If two accepted* $\mathsf{Knot}^\uparrow$ *classifier rows have the same downstream consumer boundary packet, then they have the same public classifier surface. In particular, their visible endpoint transports agree by* hsame*, and their Reidemeister and continuation ledgers agree as the finite ledger surface specified by Definition 254.16.*

Proof. Assume two accepted classifier rows share the same consumer boundary packet. The shared packet identifies the projected $\mathsf{S}^{1\uparrow}$ endpoint family and the ambient $\mathsf{Homotopy}^\uparrow$ rows, so the endpoint transports used by the classifier are the same hsame rows.

Apply Definition 254.16 to each classifier row. Both rows decompose into finite ledgers generated by endpoint hsame transport and Reidemeister I, II, and III ProbeBundle rows over the visible endpoints supplied by the shared packet.

The shared ProbeBundle and Cont ledgers therefore give the same generator list and the same package provenance. Repacking those generators through Definition 254.2 yields the same public classifier surface for both accepted rows. □

Theorem 254.21 (KnotUp consumer continuation composition)**.** *Composable downstream consumer boundary packets compose by concatenating their finite* Cont *continuation ledgers and their finite Reidemeister* ProbeBundle *ledgers. The composed packet preserves the endpoint source exhaustion of Definition 254.14 and remains accepted by the classifier composition theorem Definition 254.15.*

Lean4 ✓ `BEDC.Derived.KnotUp.KnotDiagramPacket consumer cont composition`

Proof. Take two consumer boundary packets whose target endpoint family of the first matches the source endpoint family of the second. The shared middle endpoint is one of the projected $\mathsf{S}^{1\uparrow}$ and $\mathsf{Homotopy}^{\uparrow}$ rows named by the packets, so Definition 254.14 supplies the same source identification on both sides.

Concatenate the two finite Cont ledgers and concatenate the two finite Reidemeister ProbeBundle ledgers. The package provenance rows are composable because both packets expose the same middle endpoint and ambient scope.

Finally apply Definition 254.15. It accepts exactly this concatenated ledger as a classifier row, while the endpoint-source exhaustion theorem keeps the visible source rows fixed at the composed boundary. Repacking the rows gives the composed consumer boundary packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$**.**

Constructive story: A knot packet is generated from a consumed $\mathsf{S}^{1\uparrow}$ source packet, an ambient $\mathsf{Homotopy}^{\uparrow}$ deformation row, finite crossing data, a finite Reidemeister trace, endpoint rows, a Pkg provenance row, and a Cont ledger tying those rows together. The carrier accepts exactly diagram packets whose endpoints factor through the displayed circle source and ambient deformation source, whose crossings and moves are finite BHist or ProbeBundle rows, and whose hidden projection data stays inside the continuation ledger; host embeddings and undecomposed ambient isotopy are excluded. Two packets are classified by transporting the circle source, crossing entries, Reidemeister ledgers, homotopy rows, endpoints, provenance, and continuation ledger componentwise by hsame and finite Cont refinements. The source is the displayed diagram packet over $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Homotopy}^{\uparrow}}$, the pattern records source-boundary extraction, Homotopy ledger boundary, projection stability, finite Reidemeister reflexivity, reversal, concatenation, endpoint exhaustion, and classifier completeness, and the ledger keeps source endpoints, ambient rows, crossings, local moves, deformation rows, and package provenance visible. The five name-certificate obligations are supplied by any accepted diagram packet for carrier inhabitation, by the empty, reversed, and concatenated Reidemeister ledgers for reflexivity, symmetry, and transitivity, and by endpoint-source exhaustion plus ledger composition for closure under classified transport. A history h lies in the signature gap of a $\mathsf{Knot}^{\uparrow}$ token p when the source-domain accepts its diagram packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h has the same $\mathsf{S}^{1\uparrow}$ endpoints, HomotopyUp deformation row, Reidemeister trace, classifier ledger, and provenance encoded by p. The chapter registers this finite diagram-and-move construction under the name $\mathsf{Knot}^{\uparrow}$; ambient-isotopy quotients, host knot types, and general knot invariants beyond the displayed certificate rows are not introduced.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature $\mathsf{Knot}^{\uparrow}$ surface is the finite diagram consumer boundary given by the consumer boundary extraction row and closed under consumer continuation-composition over the displayed $\mathsf{S}^{1\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, Pkg, Cont, and hsame rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.KnotUp.KnotDiagramPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close ambient-isotopy quotients, host knot types, or general knot invariants beyond the displayed finite certificate rows.

Upgrade path: Only Lean-side bridge checking remains beyond the mature finite diagram consumer surface.

Chapter 255

A Concrete Naming Certificate for $\mathsf{JonesPolynomial}^{\uparrow}$

The Jones-polynomial interface $\mathsf{JonesPolynomial}^{\uparrow}$ packages the Laurent polynomial knot invariant in $\mathbb{Z}[t^{1/2}, t^{-1/2}]$ defined by the skein relation. It depends on $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$ (chapter 254) and $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50).

255.1 Carrier definition

255.2 The certificate

Definition 255.1 (JonesPolynomial skein ledger packet)**.** *A* $\mathsf{JonesPolynomial}^{\uparrow}$ *skein ledger packet is a finite BEDC row family*

$$\mathcal{J} = (\mathcal{K}, \mathcal{K}_+, \mathcal{K}_-, \mathcal{K}_0, J, \sigma, \rho, \lambda)$$

with the following visible fields.

1. *$\mathcal{K}$ is an accepted* $\mathsf{Knot}^{\uparrow}$ *diagram packet of Definition 254.1; it is the diagram row whose polynomial endpoint is being named.*
2. *$\mathcal{K}_+$, $\mathcal{K}_-$, and $\mathcal{K}_0$ are accepted* $\mathsf{Knot}^{\uparrow}$ *diagram packets over the same displayed* $\mathsf{S}^{1\uparrow}$ *source, ambient* $\mathsf{Homotopy}^{\uparrow}$ *row, endpoint family, and* Pkg *scope, except for one local skein window tagged respectively as the positive crossing, negative crossing, and oriented smoothing boundary row.*
3. *J is a Laurent-polynomial endpoint row whose underlying normalized finite coefficient row is accepted by* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *through Definition 50.3, Definition 50.21, and Definition 50.24. The Laurent bookkeeping is carried as displayed endpoint data and is not read as host polynomial equality.*
4. *σ is a finite* Bundle*-indexed skein ledger. Each entry names one local skein face, the three* $\mathsf{Knot}^{\uparrow}$ *boundary rows $\mathcal{K}_+, \mathcal{K}_-, \mathcal{K}_0$, the endpoint row J, the polynomial classifier rows consumed at that face, and the* $\mathsf{Knot}^{\uparrow}$ *classifier rows showing that Reidemeister transport of the surrounding diagram commutes with reading the same local face.*
5. *ρ records the shared* Pkg *provenance of the four knot packets, the polynomial endpoint, and the finite skein ledger. The continuation ledger λ lists exactly the* Cont, hsame, Bundle, *polynomial-classifier, knot-classifier, and exactness rows accepted by the public* $\mathsf{JonesPolynomial}^{\uparrow}$ *surface.*

The packet exports only these rows. It does not export a host knot quotient, external diagram equality, or a complete Jones-polynomial invariant beyond the supplied finite skein ledger.

Lean4 (definition) `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket`

Definition 255.2 (JonesPolynomial skein ledger classifier)**.** *Two packets* $\mathcal{J} = (\mathcal{K}, \mathcal{K}_+, \mathcal{K}_-, \mathcal{K}_0, J, \sigma, \rho, \lambda)$ *and* $\mathcal{J}' = (\mathcal{K}', \mathcal{K}'_+, \mathcal{K}'_-, \mathcal{K}'_0, J', \sigma', \rho', \lambda')$ *of Definition 255.1 are related by* $\mathsf{JonesSkeinCls}$ *when the following finite rows are supplied.*

1. *The main diagram rows* $\mathcal{K}$ *and* $\mathcal{K}'$ *are related by the* $\mathsf{Knot}^{\uparrow}$ *Reidemeister ledger classifier of Definition 254.2.*
2. *The three skein boundary pairs* $(\mathcal{K}_+, \mathcal{K}'_+)$*,* $(\mathcal{K}_-, \mathcal{K}'_-)$*, and* $(\mathcal{K}_0, \mathcal{K}'_0)$ *are related by the same* $\mathsf{Knot}^{\uparrow}$ *classifier surface, with matching local-window tags and the same transported endpoint rows.*
3. *The polynomial endpoints* J *and* J' *are related by the normalized structural classifier of* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ *from Definition 50.21, with stability read through Definition 50.22.*
4. *The ledgers* σ *and* σ' *are matched by finite* Bundle *induction: empty skein ledgers compare, and a cons step compares the displayed local skein face after applying the four* $\mathsf{Knot}^{\uparrow}$ *transports and the* $\mathsf{Poly}^{\uparrow}$ *endpoint classifier row.*
5. *The provenance rows* ρ, ρ' *and continuation ledgers* λ, λ' *list exactly the transported rows above and no further host-level witness.*

Lean4 (definition) `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerClassifier`
Lean4 (definition) `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket classifier`

Lemma 255.3 (JonesPolynomial skein ledger Reidemeister transport)**.** *Let* $\mathcal{J}$ *be a* $\mathsf{JonesPolynomial}^{\uparrow}$ *skein ledger packet. Transport* $\mathcal{K}$ *and the three boundary rows* $\mathcal{K}_+, \mathcal{K}_-, \mathcal{K}_0$ *by accepted finite* $\mathsf{Knot}^{\uparrow}$ *Reidemeister ledgers whose displayed endpoint transports preserve the local skein window. Transport the endpoint* J *by an accepted* $\mathsf{Poly}^{\uparrow}$ *classifier row. Then the transported rows repack as a* $\mathsf{JonesPolynomial}^{\uparrow}$ *skein ledger packet* $\mathcal{J}'$*, and* $\mathsf{JonesSkeinCls}(\mathcal{J}, \mathcal{J}')$ *holds.*

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket reidemeister transport`

Proof. Project the four knot rows of $\mathcal{J}$ to downstream consumer boundary packets by Definition 254.19. Each projection contains exactly the displayed $\mathsf{S}^{1\uparrow}$ source rows, ambient $\mathsf{Homotopy}^{\uparrow}$ rows, finite Reidemeister Bundle ledger, finite Cont ledger, and Pkg provenance.

Apply Definition 254.15 to compose any Reidemeister transport already present in the packet with the supplied transport ledgers. Then apply Definition 254.16 to read the result again as a finite ledger generated by endpoint hsame transport and Reidemeister move rows. The hypothesis that the local skein window is preserved identifies the positive, negative, and smoothing tags after transport, so every skein face in σ has a transported face over the same displayed boundary pattern.

The polynomial endpoint transport is accepted by the normalized structural classifier of Definition 50.21; the stability rows of Definition 50.22 keep the endpoint inside the $\mathsf{Poly}^{\uparrow}$ carrier. Induct over the finite Bundle ledger σ. The empty ledger transports to the empty ledger. In the cons case, the head face is transported by the four knot classifier rows and the polynomial endpoint classifier row, while the induction hypothesis transports the remaining finite ledger.

Collecting the transported skein faces gives a finite ledger σ'. The transported provenance and continuation rows give ρ' and λ'. These fields satisfy the clauses of Definition 255.1, and the same construction is exactly the classifier relation of Definition 255.2. □

Theorem 255.4 (JonesPolynomial skein ledger obligation surface)**.** *Let $\mathcal{J}$ be a* $\mathsf{JonesPolynomial}^{\uparrow}$ *skein ledger packet of Definition 255.1. Assume its finite skein ledger is compatible with Reidemeister transport in the sense of Definition 255.3. Then the finite row family consisting of the main* $\mathsf{Knot}^{\uparrow}$ *diagram row, the three skein-related* $\mathsf{Knot}^{\uparrow}$ *boundary rows, the* $\mathsf{Poly}^{\uparrow}$ *Laurent-polynomial endpoint row, the finite* Bundle *skein ledger, the shared* Pkg *provenance, the finite* Cont *continuation ledger, and the classifier* JonesSkeinCls *forms an accepted carrier/classifier surface for* $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ *over those rows.*

The exactness boundary of this surface is the displayed packet itself: every accepted $\mathsf{JonesPolynomial}^{\uparrow}$ *row projects to those listed fields, and every listed field repacks into the same packet. No full Jones-polynomial invariant, unknot normalization, global skein recursion, host polynomial quotient, or external ambient-isotopy primitive is part of this obligation surface.*

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket obligation surface`

Proof. Start with the packet fields displayed in Definition 255.1. The main diagram row and the three skein boundary rows are accepted $\mathsf{Knot}^{\uparrow}$ packets, so Definition 254.19 exposes their public boundary packets and Definition 254.20 fixes the classifier surface read by any downstream consumer.

The endpoint J is accepted by the polynomial carrier and classifier rows of Definition 50.3, Definition 50.21, and Definition 50.24. Thus the polynomial side of the surface is a normalized finite coefficient endpoint carried by $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$, with the Laurent tags retained only as displayed endpoint data.

The finite skein ledger σ ties these two sides together. Each ledger entry reads the same three $\mathsf{Knot}^{\uparrow}$ boundary rows, the same polynomial endpoint row, and the same package provenance. By the Reidemeister transport lemma, transporting the diagram and boundary rows through accepted $\mathsf{Knot}^{\uparrow}$ classifier ledgers transports the corresponding skein face and keeps the polynomial endpoint inside the $\mathsf{Poly}^{\uparrow}$ classifier. Hence the surface has a stable classifier, namely Definition 255.2.

For exactness, project an accepted surface row to its knot fields, polynomial endpoint field, skein ledger, provenance row, and continuation ledger. These are precisely the fields of Definition 255.1. Conversely, given those fields, the definition repacks them as a $\mathsf{JonesPolynomial}^{\uparrow}$ skein ledger packet, and the classifier definition supplies the accepted comparison rows. The public carrier/classifier surface is therefore exactly the supplied finite skein-ledger row family and contains no additional invariant or bridge row. □

Theorem 255.5 (JonesPolynomial skein endpoint classifier exhaustion)**.** *Every accepted* JonesSkeinCls *witness between skein ledger packets $\mathcal{J}$ and $\mathcal{J}'$ projects exactly to the main* $\mathsf{Knot}^{\uparrow}$ *row, the three skein-boundary* $\mathsf{Knot}^{\uparrow}$ *rows, the* $\mathsf{Poly}^{\uparrow}$ *endpoint classifier row, the finite* Bundle *skein ledger, the shared* Pkg *provenance, and the finite* Cont *continuation ledger displayed in Definition 255.2. Conversely, any finite row family with precisely those fields repacks as the same* JonesSkeinCls *witness.*

Thus endpoint readback for $\mathsf{JonesPolynomial}^{\uparrow}$ *is exhausted by generated classifier data: the endpoint comparison reads the listed knot, polynomial, ledger, provenance, and continuation rows, not host Laurent-polynomial equality or a knot quotient.*

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket endpoint classifier exhaustion`

Proof. Project an accepted classifier witness by the constructor clauses of Definition 255.2. The first four clauses give the main $\mathsf{Knot}^{\uparrow}$ row, the three skein-boundary $\mathsf{Knot}^{\uparrow}$ rows, the $\mathsf{Poly}^{\uparrow}$ endpoint classifier row, and the finite Bundle ledger comparison.

For the ledger component, induct over the finite Bundle skein ledger. The empty ledger projects to the empty displayed row family. In the cons step, the head face projects to its four knot classifier rows and its polynomial endpoint classifier row, while the induction hypothesis projects the tail ledger.

Apply Definition 255.3 at each head face to keep Reidemeister transport inside the same local skein window. The polynomial classifier specification from Definition 50.21 keeps the Laurent endpoint readback inside $\mathsf{NameCert}_{\mathsf{Poly}^\uparrow}$.

The final classifier clauses project the shared Pkg provenance and finite Cont continuation ledger. These are exactly the rows listed in the theorem statement, and no further host witness is present in the classifier definition.

Conversely, given the listed rows, rebuild the witness by the same constructor clauses. The finite Bundle ledger is rebuilt by the empty and cons cases just used in projection, the knot rows are accepted by their displayed $\mathsf{Knot}^\uparrow$ classifier clauses, and the polynomial endpoint row is accepted by the polynomial classifier specification. Repacking these fields gives precisely an accepted JonesSkeinCls witness. □

Theorem 255.6 (JonesPolynomial local skein-window composition)**.** *Let $\mathcal{J}_1$ and $\mathcal{J}_2$ be finite* $\mathsf{JonesPolynomial}^\uparrow$ *skein ledger packets over* BHist *as in Definition 255.1. Suppose the terminal local skein-window row of $\mathcal{J}_1$ and the initial local skein-window row of $\mathcal{J}_2$ are the same displayed* $\mathsf{Knot}^\uparrow$ *boundary row under* hsame*, and suppose the adjacent Laurent endpoints carry compatible* $\mathsf{Poly}^\uparrow$ *endpoint classifier rows. Then the finite* Bundle *skein ledgers of $\mathcal{J}_1$ and $\mathcal{J}_2$ concatenate to a finite* $\mathsf{JonesPolynomial}^\uparrow$ *skein ledger packet whose classifier projects exactly to the two original classifiers, the shared* $\mathsf{Knot}^\uparrow$ *boundary row, the compatible* $\mathsf{Poly}^\uparrow$ *endpoint row, the inherited* Cont *ledgers, and the shared* Pkg *provenance.*

The composed packet is a local finite-window construction. It introduces no host Laurent-polynomial equality, no knot quotient, and no global invariant row beyond the displayed BHist*,* Bundle*,* hsame*,* Cont*,* Pkg*,* $\mathsf{Knot}^\uparrow$*, and* $\mathsf{Poly}^\uparrow$ *data.*

Proof. Unpack the two packets by Definition 255.1. Each packet supplies four visible $\mathsf{Knot}^\uparrow$ rows, one $\mathsf{Poly}^\uparrow$ endpoint row, a finite Bundle skein ledger, a Pkg provenance row, and a finite Cont continuation ledger.

Concatenate the two finite Bundle skein ledgers at the shared boundary face. The hypothesis identifies that face as the same displayed $\mathsf{Knot}^\uparrow$ local-window row under hsame, and the compatible $\mathsf{Poly}^\uparrow$ endpoint classifier row gives the polynomial endpoint comparison needed at the join.

Apply Definition 255.3 to the shared boundary face. The Reidemeister transport stays inside the same local skein window, so the concatenated ledger still has the positive, negative, and smoothing boundary rows required by Definition 255.1.

The provenance rows and continuation ledgers concatenate as finite Pkg and Cont rows over the same public source certificates. Repacking these fields therefore gives a $\mathsf{JonesPolynomial}^\uparrow$ skein ledger packet.

Finally apply Definition 255.5. The classifier of the composed packet projects exactly to the two original classifier surfaces plus the shared boundary row and the compatible polynomial endpoint row. Since endpoint exhaustion lists no host polynomial equality or knot quotient, the composition does not introduce either one. □

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket local skein window composition`

Theorem 255.7 (JonesPolynomial public NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ *obligation surface consists exactly of the skein ledger packet, skein classifier, Reidemeister transport row,* $\mathsf{Poly}^{\uparrow}$ *endpoint classifier,* $\mathsf{Knot}^{\uparrow}$ *boundary rows, finite* Bundle *skein ledger,* Cont *ledger, and shared* Pkg *provenance displayed in this section. It exports no full ambient-isotopy quotient, no host Laurent-polynomial equality, and no complete-knot-invariant claim.*

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket namecert obligation surface`

Proof. Start with the packet of Definition 255.1. Its fields already list the main knot diagram, the three skein boundary diagrams, the polynomial endpoint, the finite skein ledger, the continuation ledger, and the shared provenance row.

The classifier is Definition 255.2. It supplies the accepted comparison rows for the knot faces, the polynomial endpoint, the ledger, and the provenance component. The Reidemeister stability needed by that classifier is exactly Definition 255.3.

Apply Definition 255.4. That theorem packages the displayed packet and classifier into the accepted finite carrier/classifier surface for $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ over the knot, polynomial, bundle, continuation, and provenance rows.

Finally read the roadmap ten-field template field by field. Each required public field is one of the rows just named, and every row in the public surface projects back to the same packet. Hence the public NameCert obligation surface is exactly the displayed skein-ledger construction, with no additional quotient, host-polynomial equality, or invariant row. □

Theorem 255.8 (JonesPolynomial carrier obligation)**.** *The carrier obligation for* $\mathsf{JonesPolynomial}^{\uparrow}$ *is exactly the finite* JonesSkeinLedgerPacket *over* BHist*,* Bundle*,* Cont*,* hsame*,* Pkg*,* $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{S}^{1\uparrow}}$*. Its rows are the displayed main knot row, the three skein-boundary rows, the polynomial endpoint row, the finite skein* Bundle*, the continuation ledger, and the shared provenance row. It contains no host knot quotient, no host Laurent-polynomial equality, and no complete-invariant row.*

Proof. Unpack Definition 255.1. The packet definition lists precisely the BHist knot histories, polynomial endpoint history, finite Bundle ledger, Cont continuation entries, hsame comparison rows, and Pkg provenance field named in the statement.

Apply Definition 255.3 to the three boundary rows. Transport stays inside the same displayed skein window, so the carrier does not acquire an ambient quotient or an external polynomial comparison.

Finally use Definition 255.4 together with Definition 255.7. These identify the public carrier part of the $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ surface with the packet fields just enumerated and with no additional host row. □

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket carrier obligation`

Theorem 255.9 (JonesPolynomial classifier obligation)**.** *The classifier obligation for* $\mathsf{JonesPolynomial}^{\uparrow}$ *is exactly* JonesSkeinCls*. It consists of the main knot classifier row, the positive-crossing, negative-crossing, and smoothing boundary classifier rows, the* $\mathsf{Poly}^{\uparrow}$ *endpoint classifier, finite* Bundle *comparison,* Pkg *provenance comparison, and* Cont *continuation rows attached to the packet of Definition 255.1.*

Proof. Start from Definition 255.2. Its fields are the accepted comparisons for the main $\mathsf{Knot}^{\uparrow}$ row, the three skein boundary rows, the polynomial endpoint row, the finite skein Bundle, the provenance component, and the continuation ledger.

Use Definition 255.3 for the boundary comparisons and Definition 255.5 for the endpoint comparison. Together they show that every accepted classifier witness projects to exactly those rows.

The reverse packing is supplied by Definition 255.4. The public surface theorem Definition 255.7 then reads the same classifier rows as the $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ classifier obligation, with no extra equality principle. □

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket classifier obligation`

Theorem 255.10 (JonesPolynomial ledger exactness obligation)**.** *Every accepted public* $\mathsf{JonesPolynomial}^{\uparrow}$ *row projects to the carrier packet, classifier, Reidemeister-transport, endpoint, finite* Bundle*,* Cont*, and* Pkg *ledger fields of Definition 255.8 and Definition 255.9. Conversely, those fields repack as the accepted public* $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ *row. The exactness surface uses no host knot quotient and no host Laurent-polynomial equality.*

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket ledger exactness obligation`

Proof. For the projection direction, apply Definition 255.7. It states that the public row consists exactly of the skein ledger packet, classifier, Reidemeister transport row, endpoint classifier, finite Bundle ledger, Cont ledger, and shared Pkg provenance.

Refine the projected rows by Definition 255.8 and Definition 255.9. The carrier theorem names the packet fields, and the classifier theorem names the JonesSkeinCls comparison fields.

For the repacking direction, use Definition 255.6 to compose the local skein window inside the displayed finite ledger, then apply Definition 255.5 to recover the endpoint row.

Finally invoke Definition 255.4 and Definition 255.7. They repack exactly the named carrier, classifier, transport, endpoint, ledger, continuation, and provenance rows as the public certificate, while excluding host quotient and host polynomial equality rows. □

Theorem 255.11 (JonesPolynomial skein classifier determinacy)**.** *Fix a* $\mathsf{JonesPolynomial}^{\uparrow}$ *skein ledger packet whose main* $\mathsf{Knot}^{\uparrow}$ *diagram row, three skein boundary rows,* $\mathsf{Poly}^{\uparrow}$ *endpoint classifier row, finite skein* Bundle*, finite* Cont *ledger, and* Pkg *provenance are displayed as in Definition 255.1. Then any accepted* JonesSkeinCls *witness over those fixed rows determines the same displayed* $\mathsf{JonesPolynomial}^{\uparrow}$ *endpoint row by* hsame *and* Cont *transport. The determinacy is internal to the skein classifier surface and asserts no host Laurent-polynomial equality, knot quotient, or complete invariant.*

Proof. Start with two accepted JonesSkeinCls witnesses over the fixed packet rows. The packet fixes the main knot diagram, positive crossing, negative crossing, smoothing boundary rows, the polynomial endpoint row, the finite skein bundle, the continuation ledger, and the provenance row.

Apply Definition 255.3. The Reidemeister row transports only the displayed $\mathsf{Knot}^{\uparrow}$ local-window boundary, so both classifier witnesses read the same boundary diagrams through hsame.

Next use Definition 255.5. It projects each accepted classifier witness exactly to the fixed knot rows, the fixed polynomial endpoint classifier, the finite skein ledger, the Cont ledger, and the Pkg provenance.

Apply Definition 255.6 to the fixed local window. The composition theorem keeps the skein bundle and the polynomial endpoint comparison at the same displayed boundary, so the two classifier readings cannot produce distinct displayed endpoint rows.

Finally cite Definition 255.7. The public surface contains exactly these packet and classifier rows. Repacking the exhausted rows therefore gives one $\mathsf{JonesPolynomial}^{\uparrow}$ endpoint row determined by hsame and Cont transport. □

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket skein classifier determinacy`

Theorem 255.12 (JonesPolynomial skein boundary no-confusion)**.** *Fix a* JonesSkeinCls *packet as in Definition 255.1. An accepted classifier witness cannot interchange the positive-crossing boundary row, the negative-crossing boundary row, and the smoothing boundary row unless the displayed local-window tag and the corresponding finite* Bundle *ledger row are the same packet entry. The separation is internal to the skein ledger and does not assert any host knot quotient or Laurent-polynomial equality.*

Proof. Unpack the packet of Definition 255.1. Its skein boundary fields are three displayed local-window rows, each paired with its own tag and its own position in the finite skein Bundle ledger.

Proceed by closed finite Bundle induction over the skein ledger. In the base case there is no ledger row available, so no accepted classifier witness can manufacture a boundary tag from outside the packet.

For the induction step, inspect the head ledger row. If the displayed tag is the positive-crossing tag, the classifier reads the positive boundary entry and the remaining bundle is handled by the induction hypothesis. The negative and smoothing tag cases are identical, with their own displayed boundary entries.

Apply Definition 255.5. Every accepted witness projects back to the fixed knot rows, the finite skein ledger, the Cont ledger, and the Pkg provenance. Thus an attempted interchange would require changing either the displayed local-window tag or the ledger row selected by that tag.

The tag fields in Definition 255.1 are distinct packet coordinates. Hence two boundary rows can be identified by an accepted witness only when the displayed tag and the finite Bundle ledger row already agree as the same coordinate of the fixed packet. No quotient of host knots and no Laurent-polynomial equality is used. □

Lean4 ✓ `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket skein boundary no confusion`

Theorem 255.13 (Jones polynomial mature finite skein consumer example)**.** *Let C be a downstream knot or diagram consumer whose input is a finite skein-ledger packet read through the bridged* $\mathsf{JonesPolynomial}^{\uparrow}$ *surface of this chapter. Then the visible data available to C are exactly the endpoint classifier, local skein-window composition, carrier and classifier obligations, ledger exactness, skein classifier determinacy, and no-confusion boundary displayed above.*

The consumer may therefore inspect the finite packet rows named by JonesSkeinLedgerPacket*,* JonesSkeinCls*,* $\mathsf{Knot}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* Bundle*,* Cont*,* hsame*,* Pkg*, and the inherited* NameCert *surfaces. It cannot read a quotient of diagrams, a full Jones invariant, or a Reidemeister-complete standard model.*

Proof. Start with the checked skein-ledger transport of Definition 255.3. It keeps a finite local skein packet inside the same displayed boundary rows and transports only the knot, polynomial endpoint, continuation, and provenance entries already stored in the packet.

Next apply the obligation and endpoint surfaces: Definition 255.4, Definition 255.5, and Definition 255.6. These identify the endpoint classifier, the finite local-window composition, and the displayed skein ledger as the consumer's available row family.

Now read the public $\mathsf{NameCert}_{\mathsf{JonesPolynomial}^{\uparrow}}$ obligations through Definition 255.7, Definition 255.8, Definition 255.9, and Definition 255.10. The carrier, classifier, and ledger fields are exhausted by the finite packet coordinates and their Cont, hsame, and Pkg provenance.

Finally use Definition 255.11 and Definition 255.12. They prevent a consumer from separating two endpoint reads or interchanging the positive, negative, and smoothing boundary entries except

through the displayed finite ledger coordinates. Since every cited row belongs to that finite surface, no diagram quotient, full invariant, or Reidemeister-complete standard model is available to C. □

Closure status for $\mathsf{NameCert}_{\mathsf{JonesPolynomial}\uparrow}$.

Constructive story: A Jones-polynomial packet is generated by taking one accepted $\mathsf{Knot}^{\uparrow}$ diagram row, its positive, negative, and smoothing boundary rows in a local skein window, a Laurent-polynomial endpoint carried by $\mathsf{Poly}^{\uparrow}$, a finite Bundle skein ledger, a Pkg provenance row, and a finite Cont continuation ledger. The carrier accepts only packets in which all four knot rows share the displayed source, ambient, endpoint, and package scope, and in which the polynomial endpoint is read through the normalized finite coefficient carrier; a host knot quotient, host Laurent-polynomial equality, or complete invariant row is excluded. Two packets are classified by the $\mathsf{Knot}^{\uparrow}$ Reidemeister ledger classifier on the main and boundary diagrams, the $\mathsf{Poly}^{\uparrow}$ endpoint classifier on the polynomial row, and finite induction over the skein ledger, with local tags preventing confusion between positive, negative, and smoothing entries. The source is this finite skein ledger over $\mathsf{NameCert}_{\mathsf{Knot}\uparrow}$, $\mathsf{NameCert}_{\mathsf{Poly}\uparrow}$, and $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$, the pattern records Reidemeister transport, skein endpoint exhaustion, local-window composition, classifier determinacy, and no-confusion, and the ledger keeps every knot boundary, polynomial endpoint, local face, continuation row, and provenance row visible. The name-certificate obligations at this seed surface are carried by the accepted packet for inhabitation, by the inherited knot and polynomial classifiers plus finite bundle induction for reflexive, symmetric, and transitive comparison, and by the transport and exhaustion rows for closure under classified Reidemeister and polynomial endpoint movement. A history h lies in the signature gap of a $\mathsf{JonesPolynomial}^{\uparrow}$ token p when the source-domain accepts its skein packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, p names the same local skein faces, four knot rows, polynomial endpoint, and finite ledger. The chapter registers this finite skein-ledger construction under the name $\mathsf{JonesPolynomial}^{\uparrow}$; it does not introduce a full Jones invariant, diagram quotient, Reidemeister-complete standard model, or bridgeChecked verification.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JonesPolynomial}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The obligation scope is the finite skein ledger over BHist, Cont, hsame, Pkg, $\mathsf{NameCert}_{\mathsf{Knot}\uparrow}$, $\mathsf{NameCert}_{\mathsf{Poly}\uparrow}$, and $\mathsf{NameCert}_{\mathsf{S}^1\uparrow}$ displayed by the carrier, classifier, transport, exhaustion, determinacy, and no-confusion blocks in this chapter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JonesPolynomial}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.JonesPolynomialUp.JonesSkeinLedgerPacket\_namecert\_obligation\_surface`

Bridge status: none

Not claimed: This chapter does not claim a full Jones invariant, a quotient of diagrams, Reidemeister-complete standard model, or bridgeChecked verification.

Upgrade path: No further paper-axis upgrade is pending; bridgeChecked remains a formal-axis matter.

Chapter 256

A Concrete Naming Certificate for $\mathsf{ThreeManifold}^{\uparrow}$

The 3-manifold interface $\mathsf{ThreeManifold}^{\uparrow}$ packages a closed orientable manifold of dimension three together with the JSJ and geometric decompositions licensed by Thurston-Perelman geometrisation. It depends on $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$ (chapter 96) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88).

256.1 Carrier definition

Definition 256.1 (ThreeManifold finite carrier)**.** *A* $\mathsf{ThreeManifold}^{\uparrow}$ *carrier is a finite* BHist *packet displaying a closed orientable three-dimensional* $\mathsf{Manifold}^{\uparrow}$ *row, a* $\mathsf{Topology}^{\uparrow}$ *source row, a decomposition ledger row for the visible JSJ and geometric pieces, componentwise* hsame *comparisons,* Cont *rows connecting the decomposition ledger to the source rows, and* Pkg *provenance for the manifold and topology certificates. The carrier exports no checked geometrisation bridge, no host smooth manifold, and no unlogged decomposition object.*

Theorem 256.2 (ThreeManifold carrier obligation)**.** *Every accepted* $\mathsf{ThreeManifold}^{\uparrow}$ *packet is generated by the finite* BHist *carrier of Definition 256.1: its manifold row, topology row, decomposition ledger,* hsame *classifier rows,* Cont *continuation rows, and* Pkg *provenance are all displayed, and the dependency rows from* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *are the only source rows available to downstream consumers.*

Lean4 ✓ `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier obligation surface`

Proof. Project the carrier definition. The packet records the $\mathsf{Manifold}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ dependencies as explicit source rows and stores the decomposition data as a finite ledger attached to them.

The classifier rows are componentwise hsame comparisons over the displayed coordinates. The continuation rows tie those coordinates to the packet readback, and the package rows identify the two dependency certificates.

Since no other coordinate is present in the BHist packet, every accepted readback is confined to the displayed carrier rows. □

Theorem 256.3 (ThreeManifold topology source scope)**.** *The topology source of a* $\mathsf{ThreeManifold}^{\uparrow}$ *packet is exactly the displayed* $\mathsf{Topology}^{\uparrow}$ *row together with its* BHist *data,* hsame *transport,* Cont *ledger,* Pkg *provenance, and the paired* $\mathsf{Manifold}^{\uparrow}$ *dependency row. It cannot be replaced by a host topological space, an external atlas, or an unlogged equivalence relation.*

Lean4 ✓ `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier topology source scope`

Proof. Read the topology coordinate from the carrier. It is paired with the displayed manifold row and with the finite continuation ledger that carries the topology data into the decomposition readback.

Transport along the classifier uses only the named hsame rows. The provenance of the topology source is recorded by Pkg and is not inferred from an ambient topological model.

Thus the topology source scope is exhausted by the displayed Topology$^{\uparrow}$ and Manifold$^{\uparrow}$ dependency rows. □

256.2 The certificate

Theorem 256.4 (ThreeManifold JSJ ledger exactness)**.** *Every JSJ or geometric-decomposition entry visible in a* ThreeManifold$^{\uparrow}$ *packet is read from the finite* BHist *decomposition ledger, the* Manifold$^{\uparrow}$ *source row, the* Topology$^{\uparrow}$ *source row, componentwise* hsame *transport, the* Cont *continuation rows, and* Pkg *provenance. The ledger does not assert a checked geometrisation bridge and does not expose a host decomposition theorem.*

Lean4 ✓ `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier jsj ledger exactness`

Proof. Inspect a decomposition entry. Its inputs are the displayed manifold and topology rows, and its output is a finite ledger row inside the same packet.

Transport of that entry follows the componentwise classifier. The hsame rows move the manifold, topology, decomposition, continuation, and package coordinates together.

The readback therefore has no route to a host JSJ theorem or checked geometrisation bridge; it contains exactly the displayed ledger entry and its dependency provenance. □

Theorem 256.5 (ThreeManifold chart-transition ledger obligation)**.** *Every chart-transition row or piece-boundary row consumed by the visible JSJ/geometric decomposition ledger of a* ThreeManifold$^{\uparrow}$ *packet is read through the displayed* Manifold$^{\uparrow}$ *chart row, the* Topology$^{\uparrow}$ *open-set row, componentwise* hsame *transport, finite* Cont *routing, and* Pkg *provenance. The ledger cannot introduce a host atlas, unlogged triangulation, or external geometrisation object.*

Lean4 ✓ `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier chart transition ledger obligation`

Proof. Project the finite carrier. A chart-transition or piece-boundary row appears only as a coordinate of the displayed decomposition ledger, and the carrier places that ledger beside the Manifold$^{\uparrow}$ and Topology$^{\uparrow}$ source rows.

Route the transition row through those two displayed sources. The manifold row supplies the chart coordinate, the topology row supplies the open-set coordinate, and the finite Cont rows carry the resulting readback into the visible decomposition ledger.

Componentwise hsame transport compares only the displayed coordinates, while Pkg records the provenance of the two dependency certificates. Row exhaustion from Definition 256.6 then rules out any host atlas, unlogged triangulation, or external geometrisation coordinate. □

Theorem 256.6 (ThreeManifold certificate boundary)**.** *Every downstream consumer of* ThreeManifold$^{\uparrow}$ *reads only the displayed* BHist *carrier,* Manifold$^{\uparrow}$ *dependency row,* Topology$^{\uparrow}$ *dependency row, decomposition ledger,* hsame *classifier rows,* Cont *continuation rows, and* Pkg *provenance of Definition 256.1. The certificate boundary excludes unlogged smooth, topological, or geometrisation data.*

Lean4 ✓ `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier certificate boundary`

Proof. Begin with a downstream readback. The carrier obligation theorem confines the readback to the displayed finite packet rows.

Use the topology source theorem for the topology coordinate and the JSJ ledger theorem for the decomposition coordinate. Together they account for every dependency and ledger row that the consumer can inspect.

Because the packet contains no additional smooth, topological, or geometrisation coordinate, the downstream boundary is exactly the displayed BEDC surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ThreeManifold}^{\uparrow}}$.

Constructive story: The three-manifold packet is a finite BHist carrier over displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, decomposition-ledger, classifier, continuation, and package rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ThreeManifold}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The scoped surface is the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, Pkg, carrier, decomposition-ledger, classifier, continuation, and boundary rows of the three-manifold packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ThreeManifold}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ThreeManifoldUp.ThreeManifoldFiniteCarrier\_certificate\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close checked geometrisation, host JSJ decomposition, smooth manifold semantics, or a standard bridge.

Upgrade path: Bridged closure requires a standard three-manifold bridge over the finite-carrier certificate boundary.

Chapter 257

A Concrete Naming Certificate for $\mathsf{Iwasawa}^{\uparrow}$

The Iwasawa-theory interface $\mathsf{Iwasawa}^{\uparrow}$ packages the projective limit of class groups of cyclotomic $\mathbb{Z}_p$-extensions of a number field together with the Iwasawa main conjecture relating it to a p-adic L-function. It depends on $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ (chapter 174) and $\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$ (chapter 170).

257.1 Carrier definition

Definition 257.1 (Iwasawa cyclotomic tower carrier)**.** *An* $\mathsf{Iwasawa}^{\uparrow}$ *cyclotomic tower carrier is a finite* BHist *packet whose rows are:*

1. *the cyclotomic layer rows read from* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$ *by Definition 174.29;*
2. *the ideal-class endpoint rows read from* $\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$ *by Definition 170.4;*
3. *transition rows between successive displayed levels, recorded as* Cont *ledgers and compared by* hsame*;*
4. *unary level indices identifying the finite window of tower levels;*
5. *one* Pkg *provenance row tying the cyclotomic source, ideal-class source, and transition ledger to the same carried packet.*

The carrier records only these finite BEDC rows. It does not contain a host projective limit, an analytic p-adic L-function, or an equality asserted outside the displayed BHist*,* hsame*,* Cont*, and* Pkg *surface.*

257.2 The certificate

Definition 257.2 (Iwasawa classgroup limit classifier)**.** *For two carriers of Definition 257.1, the* IwaCls *classifier holds when their unary level windows agree by* hsame*, each displayed level has matching* $\mathsf{IdealClass}^{\uparrow}$ *classifier rows, the cyclotomic source rows agree through the classifier exported by* $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$*, and the transition ledgers are compatible under* hsame *after levelwise transport. The agreement is levelwise and ledgerwise; it is not quotient equality of a host inverse limit.*

Theorem 257.3 (Iwasawa transition ledger stability)**.** *If two* $\mathsf{Iwasawa}^{\uparrow}$ *carriers are related by* IwaCls*, then* hsame *transport preserves every accepted transition ledger in Definition 257.1. The*

transported ledger is again accepted over the transported cyclotomic rows and the transported ideal-class rows, with the same unary level index window and Pkg *provenance scope.*

Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger cons transport stability`
Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger transition hsame transport`

Proof. Project the cyclotomic component of the classifier. By Definition 174.29, the downstream Iwasawa packet may read only the displayed source rows, primitive-root endpoint, factorization ledger, and provenance package from $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$.

Project the ideal-class component next. The scoped source certificate of Definition 170.4 transports carried fractional-ideal endpoints, principal-classifier witnesses, and quotient operation rows through the same BHist, hsame, and Pkg surface.

For each adjacent pair of displayed unary levels, compose these transported source projections with the carried Cont transition row. Since IwaCls requires transition-ledger compatibility under hsame, the transported transition row is accepted at the corresponding transported levels. The Pkg provenance row is unchanged except for the same componentwise transport, so no external tower object is introduced. □

Theorem 257.4 (Iwasawa source obligation surface)**.** *The carrier of Definition 257.1, the classifier of Definition 257.2, and the stability theorem Definition 257.3 form the source obligation surface for* $\mathsf{Iwasawa}^{\uparrow}$*. The surface consists exactly of cyclotomic source rows, ideal-class endpoint and classifier rows, unary level indices, transition* Cont *ledgers,* hsame *transport rows, and one* Pkg *provenance packet. It closes carrier formation, classifier agreement, transition stability, source exactness, and finite ledger readback for this packet, but it does not claim the Iwasawa main conjecture or analytic p-adic L-function equality.*

Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger source obligation surface`

Proof. Carrier formation is the packet definition itself: every accepted row is one of the cyclotomic, ideal-class, level-index, transition, or provenance rows listed in Definition 257.1.

Classifier agreement is supplied by Definition 257.2. It compares the same finite rows level by level and records transition compatibility as carried hsame data, so it never appeals to an ambient inverse-limit quotient.

Transition stability is exactly Definition 257.3. It shows that accepted transition ledgers remain accepted after componentwise carrier transport through the cyclotomic and ideal-class sources.

Exactness and ledger readback follow by the reverse projection. Any $\mathsf{Iwasawa}^{\uparrow}$ source row permitted by the surface must be one of the displayed packet rows or a row projected by the cited $\mathsf{Cyclotomic}^{\uparrow}$ and $\mathsf{IdealClass}^{\uparrow}$ source certificates. Repacking these rows gives the stated obligation surface and excludes analytic main-conjecture data from the certificate. □

Theorem 257.5 (Iwasawa window classifier obligations)**.** *The classifier* IwaCls *transports exactly the finite-window rows of Definition 257.1: unary level windows, cyclotomic source rows, ideal-class endpoint and classifier rows, transition* Cont *ledgers,* hsame *comparisons, and the shared* Pkg *provenance packet. No classifier obligation refers to a host projective limit or analytic p-adic L-function row.*

Proof. Project IwaCls through Definition 257.2. The classifier compares unary level windows by hsame, compares cyclotomic source rows through $\mathsf{NameCert}_{\mathsf{Cyclotomic}^{\uparrow}}$, compares ideal-class rows through $\mathsf{NameCert}_{\mathsf{IdealClass}^{\uparrow}}$, and checks transition-ledger compatibility.

Apply Definition 257.3 to the transition rows. It preserves the finite Cont ledger entries over the transported cyclotomic and ideal-class endpoints with the same provenance packet.

Thus every classifier obligation is one of the displayed finite-window rows, and the packet has no slot for host inverse-limit or analytic equality data. □

Theorem 257.6 (Iwasawa finite level exactness)**.** *Every accepted finite* Iwasawa$^{\uparrow}$ *tower window reads exactly the finite level rows displayed in Definition 257.1. The accepted window exposes cyclotomic rows, ideal-class rows, unary indices, adjacent-level transition ledgers,* hsame *transports, and one provenance packet, but it does not expose a host projective limit.*

Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaFiniteLevel exactness`

Proof. Unpack an accepted carrier. Its rows are precisely the five carrier entries: cyclotomic source rows, ideal-class endpoint rows, transition ledgers, unary level indices, and provenance.

Use Definition 257.4 to read exactness in the reverse direction. Any accepted source row must be projected from those finite entries or from the cited cyclotomic and ideal-class source certificates.

Therefore finite-level exactness is bounded by the displayed window itself. No projective-limit quotient is reconstructed from the accepted finite carrier. □

Theorem 257.7 (Iwasawa ledger readback obligations)**.** *Downstream number-theory consumers of the finite* Iwasawa$^{\uparrow}$ *window may read only the cyclotomic source, ideal-class source, transition ledgers, unary indices,* hsame *transports, and provenance packet named by Definition 257.4. These are the complete ledger readback obligations for the finite source surface.*

Proof. Project the consumer read through Definition 257.4. The theorem lists the finite carrier, classifier, stability, exactness, and ledger-readback rows available to the source surface.

The cyclotomic part is bounded by Definition 174.29, and the ideal-class part is bounded by Definition 170.4. The transition part is bounded by Definition 257.3.

Combining these projections leaves exactly the displayed finite-window ledger rows. Analytic main-conjecture rows, host projective limits, and p-adic L-function equalities are outside the readback surface. □

Theorem 257.8 (Iwasawa transition ledger exactness)**.** *For the carrier of Definition 257.1, the accepted transition rows are exactly the adjacent-level* Cont *ledger entries between the displayed cyclotomic rows and the displayed ideal-class rows. Soundness means that every accepted transition row has matching unary endpoints, compatible* hsame *transport, and the* Pkg *provenance tying it to the same cyclotomic and ideal-class packet. Completeness means that every adjacent pair of displayed levels carrying those cyclotomic rows, ideal-class endpoint rows,* hsame *transports, and* Pkg *provenance determines an accepted transition ledger row and no additional transition source.*

Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger finite window exactness`

Proof. For soundness, start with an accepted transition row. By Definition 257.1, it is carried as a Cont ledger between successive displayed unary levels and is packaged with the same cyclotomic source, ideal-class source, and provenance row.

Apply Definition 257.3. The transported ledger remains accepted over the transported cyclotomic and ideal-class rows, and the same Pkg provenance scope is preserved.

For completeness, begin with adjacent displayed levels equipped with the cyclotomic source rows of Definition 174.29, the ideal-class rows of Definition 170.4, compatible hsame transport, and one provenance packet. Repack these rows using Definition 257.4; the only transition slot in

that surface is the adjacent-level Cont ledger. Hence the resulting row is accepted, and the source inventory leaves no second transition kind. □

Theorem 257.9 (Iwasawa downstream consumer boundary)**.** *Every downstream consumer of the finite* $\mathsf{Iwasawa}^{\uparrow}$ *tower window may read only the cyclotomic source rows, ideal-class endpoint and classifier rows, unary level indices, accepted adjacent-level transition ledgers,* hsame *transport rows, and* Pkg *provenance of Definition 257.4. No downstream consumer row is a projective-limit quotient, an analytic main-conjecture row, or a direct p-adic L-function equality row.*

Proof. First project any downstream read through Definition 257.4. The theorem lists exactly the finite carrier, classifier, stability, exactness, and ledger-readback rows that belong to the source surface.

The cyclotomic portion is bounded by Definition 174.29. It supplies only the displayed source rows, primitive-root endpoint, factorization ledger, and provenance package consumed by the tower packet.

The ideal-class portion is bounded by Definition 170.4. It supplies the endpoint and classifier rows needed at the displayed levels, again through the finite BHist, hsame, and Pkg surface.

Finally use Definition 257.8. Transition reads are exactly adjacent-level Cont ledger rows between the already displayed sources. Therefore the consumer boundary is exhausted by the listed finite rows and cannot contain projective-limit quotient data, analytic main-conjecture data, or p-adic L-function equality data. □

Theorem 257.10 (Iwasawa finite source inventory)**.** *Every accepted finite* $\mathsf{Iwasawa}^{\uparrow}$ *source packet contains only the visible* $\mathsf{Cyclotomic}^{\uparrow}$ *tower rows, the* $\mathsf{IdealClass}^{\uparrow}$ *endpoint and classifier rows, the finite* BHist *packet holding the unary level window, the* hsame *comparison rows between transported levels, the adjacent-level* Cont *transition ledger, the* Pkg *provenance row, and the displayed* $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$ *row. In particular, the inventory has no projective limit object, no analytic p-adic L-function row, and no main-conjecture equality row.*

Proof. Project the accepted source packet through Definition 257.4. The carrier, classifier, stability, exactness, and ledger clauses listed there are all finite rows over the displayed unary tower window.

The cyclotomic part is exactly the source boundary of Definition 174.29, and the ideal-class part is exactly the scoped source certificate of Definition 170.4. These two projections supply the named $\mathsf{Cyclotomic}^{\uparrow}$ and $\mathsf{IdealClass}^{\uparrow}$ rows, while the same BHist, hsame, Cont, Pkg, and NameCert rows account for the finite tower packet. Since no projected clause has a projective-limit or analytic slot, the inventory is exhausted by the displayed rows. □

Theorem 257.11 (Iwasawa classifier transport surface)**.** *Let two finite* $\mathsf{Iwasawa}^{\uparrow}$ *source packets be connected by displayed* hsame *transport over their cyclotomic tower rows, ideal-class endpoint rows, finite* BHist *windows, transition* Cont *ledgers, and* Pkg *provenance. Then the transported packet carries the same* $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$ *source inventory: the* $\mathsf{Cyclotomic}^{\uparrow}$ *rows move to the transported tower rows, the* $\mathsf{IdealClass}^{\uparrow}$ *classifier rows move to the transported ideal-class rows, and the adjacent transition row remains the displayed finite* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger provenance hsame transport`

Proof. Begin with the hsame transport rows supplied for the tower window. Apply Definition 257.3; it carries the accepted transition ledger across the transported cyclotomic and ideal-class rows while keeping the same Pkg provenance scope.

Now transport the two source faces separately. The cyclotomic face is stable by the source boundary in Definition 174.29, and the ideal-class face is stable by the classifier rows in Definition 170.4. Repacking these transported rows gives the same finite $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$ inventory, with no new source family introduced by transport. □

Theorem 257.12 (Iwasawa transition source coverage)**.** *Every accepted* $\mathsf{Iwasawa}^{\uparrow}$ *transition row projects to one of the finite source rows of Definition 257.10: either a* $\mathsf{Cyclotomic}^{\uparrow}$ *tower source row, an* $\mathsf{IdealClass}^{\uparrow}$ *endpoint or classifier row, a finite* BHist *window row, an* hsame *transport row, an adjacent-level* Cont *ledger row, a* Pkg *provenance row, or the displayed* $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$ *row. No accepted transition row projects to an external projective-limit, analytic, or quotient source.*

Proof. Take an accepted transition row and project it through Definition 257.8. The projection identifies the transition as an adjacent-level Cont ledger between displayed source rows, not as an ambient tower limit.

Perform the finite case split over the rows listed in Definition 257.10. Cyclotomic cases are read by Definition 174.29; ideal-class cases are read by Definition 170.4; window, transport, provenance, and certificate cases are the corresponding BHist, hsame, Pkg, and NameCert projections of the same packet.

The classifier transport theorem Definition 257.11 keeps these cases stable under displayed hsame movement. Hence every accepted transition row lands in the finite source inventory and no external source remains. □

Closure status for $\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$**.**

Constructive story: An Iwasawa tower window is assembled by taking the already certified cyclotomic rows, the already certified ideal-class endpoints, a finite unary list of levels, and adjacent Cont transition ledgers tying each displayed level to the next. The carrier admits exactly that finite packet, together with one Pkg provenance row binding the cyclotomic source, ideal-class source, level indices, and transition rows; a host projective limit, an analytic p-adic L-function, and any unshown equality are left outside the carrier. Two accepted windows count as the same only when their unary level windows are related by hsame, their cyclotomic and ideal-class rows classify through the dependency certificates, and every transition ledger is transported level by level, so the source specification is the same finite tower packet and the pattern specification is the displayed cyclotomic-endpoint/ideal-class/transition shape. The ledger keeps the raw rows visible as the visible tower window: levels, endpoints, transition continuations, classifier transports, and provenance are all retained rather than collapsed into an inverse-limit object. Once one such finite window is displayed, the five name-certificate obligations follow from the dependency classifiers and finite transport: the carrier is inhabited by the packet, the classifier is reflexive, symmetric, and transitive componentwise, and an accepted packet transported through the classifier remains a packet of the same finite shape. A history h lies in the signature gap of an $\mathsf{Iwasawa}^{\uparrow}$ token p when the source-domain accepts h as this finite tower window, the signature relation pairs it with an answer window and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h must expose exactly the cyclotomic source rows, ideal-class endpoints, unary level indices, adjacent transition ledgers, hsame transports, and provenance encoded by p. The chapter registers this finite-window construction under the name $\mathsf{Iwasawa}^{\uparrow}$; no Iwasawa main conjecture, projective-limit quotient, analytic p-adic L-function, standard bridge, or additional tower primitive is introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$, maturePackageClosed)

Scope closed: The finite tower public surface is bridged to the abstract $\mathsf{Iwasawa}^{\uparrow}$ schema through the finite tower source inventory, classifier transport surface, transition source coverage, cyclotomic source rows, ideal-class endpoint and classifier rows, unary level indices, transition Cont ledgers, hsame transports, Pkg provenance, and finite ledger exactness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Iwasawa}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.IwasawaUp.IwasawaTransitionLedger\_finite\_window\_exactness`

Bridge status: none

Not claimed: The Iwasawa main conjecture, analytic p-adic L-functions, projective-limit quotient, and standard bridge are not closed by this chapter.

Upgrade path: The paper theory axis is complete for this finite-window surface; standard bridge verification remains a separate formal-axis matter.

Chapter 258

A Concrete Naming Certificate for $\mathsf{ClassField}^{\uparrow}$

The class-field-theory interface $\mathsf{ClassField}^{\uparrow}$ packages the maximal abelian extension of a number field together with the Artin map sending an idele class to a Frobenius automorphism. It depends on $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168) and $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$ (chapter 101).

258.1 Carrier definition

258.2 The certificate

Theorem 258.1 (ClassField carrier and classifier obligations)**.** *A* $\mathsf{ClassField}^{\uparrow}$ *certificate must provide a* BHist *carrier packaging a* $\mathsf{NumField}^{\uparrow}$ *base source, an* $\mathsf{Adele}^{\uparrow}$ *idele-class source, an abelian-extension history row, and a classifier stable under the source* NameCert *classifiers and* hsame*.*

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldSourceCarrier semantic name certificate`

Proof. Unpack the roadmap ten-field template for $\mathsf{ClassField}^{\uparrow}$. The visible carrier row names the base source, the idele-class source, and the abelian-extension history row.

Then read the $\mathsf{NumField}^{\uparrow}$ and $\mathsf{Adele}^{\uparrow}$ certificates as the two named dependency certificates. Their classifiers supply source replacement, while the $\mathsf{ClassField}^{\uparrow}$ row records the remaining classifier as an obligation rather than as a closed field. □

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldCarrierClassifier obligation`

Theorem 258.2 (ClassField Artin and Frobenius stability obligations)**.** *The Artin-map relation and Frobenius endpoint rows of a* $\mathsf{ClassField}^{\uparrow}$ *certificate must transport along the* $\mathsf{Adele}^{\uparrow}$ *classifier, the* $\mathsf{NumField}^{\uparrow}$ *classifier, and* BHist hsame*, and they must be compatible with the chosen extension classifier.*

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldArtinFrobeniusRows stability`

Proof. Carrier transport in the dependency NameCert objects supplies source replacement for the $\mathsf{NumField}^{\uparrow}$ and $\mathsf{Adele}^{\uparrow}$ rows. The BHist comparison supplies the remaining history replacement.

The Artin relation and the Frobenius endpoints are then listed as stability obligations for the class-field classifier. Compatibility with the extension classifier is part of that obligation surface, not a reciprocity theorem. □

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldArtinFrobenius stability obligation`

Theorem 258.3 (ClassField ledger and exactness obligations)**.** *The* $\mathsf{ClassField}^{\uparrow}$ *ledger records the hidden idele representative, the local-prime and Frobenius representative, and the extension presentation. Exactness consists of coverage, soundness, and completeness for the Artin-Frobenius classifier, without claiming reciprocity.*

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldArtinFrobeniusRows ledger exactness`
Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldLedgerExactnessRows transport`

Proof. Separate visible classifier data from hidden representative data. The visible side contains the classifier row and the Artin-Frobenius comparison; the hidden side records the idele representative, local-prime representative, Frobenius representative, and extension presentation.

Identify the three ExactGlobalize-style rows that must be supplied: coverage of the classifier, soundness of accepted Artin-Frobenius rows, and completeness for the visible rows that the classifier is meant to cover. These rows state the obligation surface and do not assert reciprocity. □

Theorem 258.4 (ClassField scoped carrier classifier)**.** *The carrier and classifier obligations of* $\mathsf{ClassField}^{\uparrow}$ *are scoped exactly over* BHist *rows for the number-field source, adele source, abelian-extension carrier, classifier, and ledger, the* $\mathsf{UnaryHistory}$ *witnesses on the source rows,* Cont *composition from the* $\mathsf{NumField}^{\uparrow}$ *row through the* $\mathsf{Adele}^{\uparrow}$ *row to the extension row,* hsame *transport from the classifier row to the extension row, and the two dependency certificates* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldSourceCarrier scoped carrier classifier`

Proof. Unpack Definition 258.1. Its source side names exactly two dependency certificates: $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ for the base number-field row and $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$ for the idele-class row. The class-field carrier is therefore not a free ambient object; it is a BHist package whose visible projections are supplied by those two certificates.

The remaining carrier row is the abelian-extension history. It is still a BHist row, so its comparison data is not an external equality relation. The classifier obligation is the restricted hsame transport that keeps the base row, idele-class row, and extension row inside the same carried package.

Thus every carrier/classifier witness used by the certificate is named by $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$, BHist, or hsame. No extra classifier or host-level carrier is part of the scoped surface. □

Theorem 258.5 (ClassField scoped Artin-Frobenius stability)**.** *The Artin-map and Frobenius stability obligations of* $\mathsf{ClassField}^{\uparrow}$ *are scoped exactly over the predicate* $\mathsf{ClassFieldArtinFrobeniusRows}$ *on base, idele, extension, Artin, and Frobenius* BHist *rows, the* Cont *transports from idele to Artin to Frobenius and from base to Frobenius to extension,* hsame *transport of idele, Artin, Frobenius, and extension rows, and the dependency classifiers supplied by* $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldArtinFrobenius scope`

Proof. Begin with Definition 258.2. The Artin relation reads its source through the $\mathsf{Adele}^{\uparrow}$ classifier and its base through the $\mathsf{NumField}^{\uparrow}$ classifier. Transport along either source is therefore the transport field already exposed by the corresponding dependency $\mathsf{NameCert}$.

The Frobenius endpoint and the abelian-extension presentation are carried histories. Their replacement step is not a separate Galois-theoretic principle; it is the hsame transport on the named BHist endpoint rows.

After these transports are named, the stability row has a fixed dependency profile: $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$ for idele-class replacement, $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ for base-field replacement, and hsame for endpoint and extension-history replacement. Compatibility with the extension classifier is precisely this repackaging of transported rows. □

Theorem 258.6 (ClassField scoped ledger exactness)**.** *The ledger exactness obligations of* $\mathsf{ClassField}^{\uparrow}$ *are scoped exactly over hidden idele representatives, local-prime representatives, Frobenius representatives, extension-presentation representatives, and the* ExactGlobalizeBase *coverage, soundness, and completeness rows for the visible Artin-Frobenius classifier.*

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassFieldScopedLedger exactness`

Proof. Unpack Definition 258.3. The visible side of the ledger is the Artin-Frobenius classifier over the scoped source rows. The hidden side consists of the idele representative, local-prime representative, Frobenius representative, and extension presentation.

Coverage says that each accepted visible Artin-Frobenius row has one such hidden witness tuple. Soundness says that any row produced from the hidden tuple is accepted by the visible classifier. Completeness says that the accepted visible rows are exhausted by those witness tuples.

These are exactly the three ExactGlobalize rows used here: coverage, soundness, and completeness. They bind the ledger to the named source certificates and BHist representatives without asserting reciprocity or maximal abelian exactness beyond the displayed classifier surface. □

Theorem 258.7 (ClassField reciprocity boundary)**.** *Coverage, soundness, and completeness for the Artin-Frobenius classifier do not assert class-field reciprocity, uniqueness of the maximal abelian extension, or a standard bridge for* $\mathsf{ClassField}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassField reciprocity boundary`

Proof. Read the ledger exactness statement in Definition 258.3 as an ExactGlobalize-style certificate. It names the visible rows covered by the classifier and the hidden representatives used to witness those rows.

Coverage, soundness, and completeness only relate accepted classifier rows to the recorded representatives. They do not identify the classifier with the classical reciprocity map, prove uniqueness of the maximal abelian extension, or supply the bridge object required for a standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$**.**

Constructive story: A class field is carried by a source packet whose visible rows are a $\mathsf{NumField}^{\uparrow}$ base, an $\mathsf{Adele}^{\uparrow}$ idele-class source, an abelian-extension history, a chosen classifier row, and a ledger row, all connected by unary history evidence and Cont composition. The carrier accepts exactly those histories in which the base and idele dependency certificates supply the first two rows and the extension endpoint is the continuation result, while reciprocity maps, maximal abelian extension exactness, and standard bridges are not fields of the packet. Its classifier is hsame on the carried extension endpoint together with the dependency classifiers for the number-field and adele rows, so Artin and Frobenius data transport only by the named source classifiers and endpoint history comparisons. The source specification is the NumField-plus-Adele carrier surface, the pattern specification is the Artin-Frobenius row with idele, local-prime, Frobenius, and extension-presentation representatives, and the ledger separates visible classifier rows from those hidden representatives through coverage, soundness, and completeness obligations. The five naming-certificate obligations are supplied by the semantic source carrier: the displayed extension endpoint inhabits the carrier, hsame gives reflexivity, symmetry, and transitivity, and result transport along hsame rebuilds the same source carrier at a classified endpoint. A history h lies in the signature GAP of a $\mathsf{ClassField}^{\uparrow}$ token p when the source accepts h, the signature relation pairs it with an answer extension row and evidence ledger, and token introduction names that answer as p; here that means the base, idele, extension, Artin,

Frobenius, classifier, hidden representatives, and ExactGlobalize rows all line up inside the displayed packet. The chapter registers this construction as $\mathsf{ClassField}^{\uparrow}$ without closing Artin-map laws, class-field reciprocity, maximal abelian extension uniqueness, or a standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope binds the public $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$ interface, carrier-classifier row, Artin-Frobenius stability row, ledger exactness rows, and reciprocity boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ClassFieldUp.ClassFieldSourceCarrier\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not close the Artin map laws, maximal abelian extension exactness, reciprocity, or a standard bridge.

Upgrade path: The paper theory axis is mature; bridgeChecked remains a separate formal-axis status for a standard bridge theorem.

Theorem 258.8 (ClassFieldUp public NameCert interface)**.** *The carrier-classifier row of Definition 258.1, the Artin-Frobenius stability row of Definition 258.2, and the ledger exactness row of Definition 258.3 assemble the public* $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$ *interface over the source certificates* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Adele}^{\uparrow}}$*. The exported surface consists exactly of the base number-field source row, the idele-class source row, the abelian-extension history row, the extension classifier, the Artin-map relation row, the Frobenius endpoint rows, the hidden idele representative, the local-prime representative, the Frobenius representative, the extension-presentation representative, and the ExactGlobalize coverage, soundness, and completeness rows for the visible Artin-Frobenius classifier. It exports no class-field reciprocity theorem, no uniqueness theorem for the maximal abelian extension, and no standard bridge.*

Proof. Apply Definition 258.1. This fixes the public source side of the certificate: a $\mathsf{NumField}^{\uparrow}$ base row, an $\mathsf{Adele}^{\uparrow}$ idele-class row, an abelian-extension history row, and the classifier comparing extension presentations after replacement through the two dependency certificates and hsame. Therefore the carrier and classifier slots of $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$ are not ambient host objects; they are the displayed BHist rows together with the two named source certificates.

Apply Definition 258.2. The Artin relation and Frobenius endpoints are the only operation-like rows added to that carrier surface. Their stability is exactly the transport through the $\mathsf{Adele}^{\uparrow}$ classifier on the idele side, the $\mathsf{NumField}^{\uparrow}$ classifier on the base side, and hsame on the extension, Artin, and Frobenius history rows. Thus the stability slot of the public certificate is filled by those displayed Artin-Frobenius rows, with compatibility to the chosen extension classifier and no additional reciprocity law.

Apply Definition 258.3. Coverage projects every accepted public Artin-Frobenius classifier row to a hidden tuple consisting of an idele representative, a local-prime representative, a Frobenius representative, and an extension-presentation representative. Soundness repacks any such displayed tuple as an accepted public row. Completeness says that the accepted public rows are exhausted by those tuples. Hence the ledger and exactness slots of $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$ are precisely the listed ExactGlobalize rows.

The forward projection from any public $\mathsf{NameCert}_{\mathsf{ClassField}^{\uparrow}}$ read therefore lands in one of the carrier, classifier, stability, representative, or ExactGlobalize rows just named. Conversely, a packet containing exactly those rows repacks through the three cited obligations into the same public interface. Since none of the three obligations introduces a classifier for reciprocity, maximal-abelian uniqueness, or a standard bridge, those objects are outside the exported surface. □

Lean4 ✓ `BEDC.Derived.ClassFieldUp.ClassField public namecert interface`

Chapter 259

A Concrete Naming Certificate for $\mathsf{Regulator}^{\uparrow}$

The regulator interface $\mathsf{Regulator}^{\uparrow}$ packages the determinant of logarithms of fundamental units (for number fields) and its Beilinson generalisation to higher K-theory. It depends on $\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$ (chapter 171) and $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168).

259.1 Carrier definition

Definition 259.1 (Regulator root input packet)**.** *A* $\mathsf{Regulator}^{\uparrow}$ *root input packet is a displayed* BHist *tuple*

$$\mathcal{R} = (\mathcal{D}, \mathcal{N}, u, \iota, \nu, \beta, \delta, \rho)$$

whose source rows are the dependency certificates $\mathcal{D} : \mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$ *and* $\mathcal{N} : \mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$*. The unit-facing rows* u *and* ι *are the visible unit and inverse/unit-product rows read from* $\mathcal{D}$*; the unary row* ν *is the Dirichlet-unit rank witness; the row* β *is the finite basis and logarithmic ledger supplied by* $\mathcal{D}$*; the row* δ *is the displayed determinant-layout ledger linking* ν*,* β*, and the* $\mathsf{NumField}^{\uparrow}$ *source rows; and* ρ *is the* Pkg *provenance tying the two dependency certificates to the regulator packet.*

The packet also carries finite Cont *readbacks from* $\mathcal{D}$ *to the unit, inverse, rank, and logarithmic ledger rows, from* $\mathcal{N}$ *to the number-field carrier, embedding, singleton-basis, and ledger rows, and from* ρ *to every exported dependency slot. It contains no analytic regulator value, no host logarithm map, no determinant evaluation in a host field, and no standard bridge row.*

Lean4 ✓ `BEDC.Derived.RegulatorUp.RegulatorTasteGate single carrier alignment`

Theorem 259.2 (Regulator DirichletUnit input boundary)**.** *Every accepted packet of Definition 259.1 projects to exactly the* $\mathsf{DirichletUnit}^{\uparrow}$ *rows needed by a root regulator source: visible unit rows, inverse/unit-product rows, the unary rank witness, the finite basis/logarithmic ledger row, finite* Cont *readbacks, and the* Pkg *provenance attached to* $\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$*. It does not project a determinant, analytic regulator value, host logarithm equality, or standard bridge from the* $\mathsf{DirichletUnit}^{\uparrow}$ *dependency.*

Lean4 ✓ `BEDC.Derived.RegulatorUp.RegulatorRootInputPacket dirichletunit input boundary`

Proof. Project the source row $\mathcal{D}$ from Definition 259.1. The available unit, inverse, rank, and logarithmic rows are precisely the rows named by Definition 171.7 and Definition 171.16.

The packet adds no second Dirichlet-unit source. Its Cont readbacks point from $\mathcal{D}$ to u, ι, ν, and β, and its provenance row ρ records those projections as the same package boundary.

Inspecting the packet fields gives the exclusion clause. Determinant layout is recorded only as the separate regulator-side row δ, while analytic regulator values, host logarithm equality, and bridge rows are not fields of $\mathcal{R}$. □

Theorem 259.3 (Regulator NumField carrier boundary)**.** *Every accepted packet of Definition 259.1 projects to the* $\mathsf{NumField}^{\uparrow}$ *source rows through* $\mathcal{N}$*: the RatUp history carrier, the RatUp endpoint classifier, the identity finite-extension row, the singleton-basis row, the scalar-action row, exact endpoint classification, and the inherited ledger provenance of the public* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *surface. No ambient host number field, quotient of embeddings, or analytic place map is exported by the regulator root packet.*

```
Lean4 ✓  BEDC.Derived.RegulatorUp.RegulatorRootInputPacket numfield finite basis boundary
```

Proof. Project the source row $\mathcal{N}$ from the regulator packet. It is a $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ dependency row, so its public carrier and classifier surface is read through Definition 168.27.

The exactness and ledger clauses are the rows isolated by Definition 168.31. They give the identity finite-extension row, singleton basis, scalar action, exact endpoint classification, and ledger provenance available to a downstream packet.

Repack these projections with the Cont readbacks and Pkg provenance of Definition 259.1. Since no host number-field carrier, embedding quotient, or analytic place map is a component of $\mathcal{R}$, none is exported at the root boundary. □

Theorem 259.4 (Regulator ledger exactness)**.** *The determinant and logarithm data visible in a Definition 259.1 are finite displayed ledger rows: the Dirichlet-unit logarithmic row* β*, the unary rank row* ν*, the number-field source ledger read from* $\mathcal{N}$*, and the determinant-layout row* δ*. These rows specify the shape of the regulator input matrix; they do not assert an analytic regulator value or equality of host logarithms.*

```
Lean4 ✓  BEDC.Derived.RegulatorUp.RegulatorLedgerSpine rows unary
Lean4 ✓  BEDC.Derived.RegulatorUp.RegulatorRootInputPacket ledger exactness
```

Proof. First project the Dirichlet-unit side by Definition 259.2. This gives ν and β as finite BHist ledger rows, read through Cont and package provenance.

Next project the number-field side by Definition 259.3. The source rows available to the regulator are the displayed carrier, classifier, finite-extension, singleton-basis, exactness, and ledger rows of $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$.

The remaining determinant-facing row is δ. By the packet definition it records only a finite layout over ν, β, and the source ledger of $\mathcal{N}$. Repacking these three sources exhausts the regulator ledger, and there is no field from which an analytic regulator value or host logarithm equality could be read. □

259.2 The certificate

Theorem 259.5 (Regulator root NameCert threshold)**.** *The root* $\mathsf{NameCert}_{\mathsf{Regulator}^{\uparrow}}$ *threshold is assembled from the packet of Definition 259.1: the* $\mathsf{NameCert}_{\mathsf{DirichletUnit}^{\uparrow}}$ *source rows, the* $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ *source rows, the visible unit and inverse rows, the unary rank witness, the finite logarithmic ledger, the determinant-layout ledger, the finite* Cont *readbacks, the* hsame *transport rows on displayed*

packet entries, and the Pkg *provenance tying the two dependency certificates to the packet. The threshold exports no analytic regulator value, no host determinant evaluation, no host logarithm equality, and no standard bridge.*

Lean4 ✓ `BEDC.Derived.RegulatorUp.RegulatorRootInputPacket root namecert threshold`

Proof. Build the carrier source from Definition 259.1. Its entries provide the two dependency certificates, the Dirichlet-unit visible rows, the number-field source rows, the determinant-layout row, and the finite readback and provenance rows.

Use Definition 259.2 for the exact Dirichlet-unit projections and Definition 259.3 for the exact number-field projections. These two boundaries bind every dependency row to BHist, Cont, hsame, NameCert, and Pkg data.

Finally use Definition 259.4. It says that the logarithm and determinant-facing entries are finite displayed ledger rows rather than analytic values. Repacking the carrier, dependency, classifier, ledger, readback, and provenance rows gives the stated root threshold, with no additional host determinant, logarithm, or bridge row to export. □

Theorem 259.6 (Regulator dependency-ledger factorization)**.** *Every public read of the root* $\mathsf{NameCert}_{\mathsf{Regulator}^{\uparrow}}$ *threshold factors through exactly three rows: the* $\mathsf{DirichletUnit}^{\uparrow}$ *projection of Definition 259.2, the* $\mathsf{NumField}^{\uparrow}$ *projection of Definition 259.3, and the finite determinant/logarithm layout ledger of Definition 259.4. The factorization exports the finite* BHist*,* Cont*,* hsame*,* NameCert*, and* Pkg *rows of those three sources, and it excludes analytic regulator values, host determinant evaluation, host logarithm equality, and standard bridge rows.*

Lean4 ✓ `BEDC.Derived.RegulatorUp.RegulatorRootInputPacket dependency ledger factorization`

Proof. Project the packet fields from Definition 259.1. The root threshold reads the two dependency certificates, the visible unit-facing entries, the number-field source entries, the determinant-layout entry, the finite readbacks, and the package provenance.

Apply Definition 259.2 and Definition 259.3. These two boundary theorems identify the dependency-facing rows that can be read from the packet: the $\mathsf{DirichletUnit}^{\uparrow}$ unit, inverse, rank, logarithmic, readback, and provenance rows, and the $\mathsf{NumField}^{\uparrow}$ carrier, classifier, finite-extension, singleton-basis, scalar-action, exactness, ledger, and provenance rows.

Now apply Definition 259.4. It isolates the remaining determinant/logarithm information as the finite displayed layout over the Dirichlet-unit rank and logarithmic rows and the number-field source ledger.

Repack the three projected sources as the public threshold surface. Since Definition 259.1 has no field for an analytic regulator value, host determinant evaluation, host logarithm equality, or standard bridge, no public read of $\mathsf{NameCert}_{\mathsf{Regulator}^{\uparrow}}$ can factor through such a row. □

Closure status for $\mathsf{NameCert}_{\mathsf{Regulator}^{\uparrow}}$**.**

Constructive story: A regulator root packet is built by placing a certified $\mathsf{DirichletUnit}^{\uparrow}$ source beside a certified $\mathsf{NumField}^{\uparrow}$ source, then adding the visible unit and inverse rows, the unary rank witness, the finite logarithmic-basis row, and a determinant-layout row connected by finite Cont readbacks. The carrier accepts only packets whose unit-facing rows come from the Dirichlet-unit certificate, whose number-field rows come from the NumField certificate, and whose determinant-facing row is a displayed layout ledger; analytic regulator values, host logarithm maps, determinant evaluation, Beilinson data, and bridge rows have no slot in the packet. Two accepted packets are classified together only by the inherited dependency classifiers and hsame transport on the displayed rank, basis, layout, readback, and provenance rows, so the source specification covers these root inputs and the pattern specification is the finite determinant/logarithm layout assembled over them. The ledger keeps the raw dependency rows and visible regulator rows aligned: a consumer can see the Dirichlet-unit projection, the NumField projection, the rank, the logarithmic ledger, the determinant-layout

ledger, the readbacks, and the Pkg provenance, but not a computed analytic number. Given such a packet, the five name-certificate obligations close on the displayed data: the packet witnesses nonemptiness, the component classifiers are reflexive, symmetric, and transitive, and transport through them preserves the same finite root-input shape. A history h lies in the signature gap of a Regulator$^{\uparrow}$ token p when the source-domain accepts h as this root packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, the dependency rows, rank, logarithmic basis, determinant layout, continuation readbacks, and provenance in h must match the packet encoded by p. The chapter registers this finite determinant-ledger construction under the name Regulator$^{\uparrow}$; it introduces no host regulator value, logarithm equality, determinant computation, standard bridge, or independent analytic carrier.

Theory closure: TheoryGate(NameCert$_{\text{Regulator}^{\uparrow}}$, maturePackageClosed)

Scope closed: The paper bridge is the public root NameCert threshold together with the dependency-ledger factorization surface over the DirichletUnitUp input, NumField carrier boundary, determinant/logarithm ledger, finite Cont readbacks, hsame transport, and Pkg provenance.

Formal verification: FormalStatus(NameCert$_{\text{Regulator}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegulatorUp.RegulatorRootInputPacket\_root\_namecert\_threshold`

Bridge status: paperBridge

Not claimed: It does not close analytic regulator values, determinant evaluation, host logarithms, Beilinson regulator data, or standard bridges.

Upgrade path: No further paper theory-axis upgrade is recorded for this chapter; bridge verification remains a separate verification-axis obligation.

Definition 259.7 (Regulator rank-zero displayed layout)**.** *Let*

$$\mathcal{R} = (\mathcal{D}, \mathcal{N}, u, \iota, \nu, \beta, \delta, \rho)$$

be a Regulator$^{\uparrow}$ *root input packet in the sense of Definition 259.1. The packet is rank-zero displayed when the unary rank witness is the empty unary row,*

$$\nu\mathsf{hsamee},$$

and the determinant-layout row δ is read only as the finite displayed layout indexed by positions licensed by ν, the Dirichlet-unit ledger β, and the public NumField$^{\uparrow}$ *source ledger of $\mathcal{N}$.*

Theorem 259.8 (Regulator rank-zero layout degeneracy)**.** *Let*

$$\mathcal{R} = (\mathcal{D}, \mathcal{N}, u, \iota, \nu, \beta, \delta, \rho)$$

be an accepted Regulator$^{\uparrow}$ *root input packet. If $\mathcal{R}$ is rank-zero displayed in the sense of Definition 259.7, then the determinant-layout ledger δ is the empty finite layout over the* DirichletUnit$^{\uparrow}$ *and* NumField$^{\uparrow}$ *projections. Consequently the rank-zero packet exports no analytic regulator value, no host determinant evaluation, and no host logarithm equality.*

Lean4 ✓ `BEDC.Derived.RegulatorUp.RegulatorRootInputPacket rank zero layout degeneracy`

Proof. Project the accepted packet through Definition 259.1. Its determinant-facing entries are only the unary rank witness ν, the finite Dirichlet-unit basis and logarithmic ledger β, the NumField$^{\uparrow}$ source ledger read from $\mathcal{N}$, and the displayed determinant-layout ledger δ linking those rows.

Apply Definition 259.2. The DirichletUnit$^{\uparrow}$ side available to the regulator consists of visible unit rows, inverse and unit-product rows, the unary rank witness, the finite basis/logarithmic ledger, finite Cont readbacks, and package provenance. In particular, the Dirichlet-unit dependency contributes no determinant row, no analytic regulator value, and no host logarithm equality.

Apply Definition 259.3 to the source $\mathcal{N}$. The NumField$^{\uparrow}$ side contributes the RatUp reflexive carrier, classifier, identity finite-extension row, singleton-basis row, scalar-action row, exactness

row, and ledger provenance. It contributes no ambient host number field, quotient of embeddings, analytic place map, determinant evaluation, or logarithm equality.

It remains to identify the determinant-layout row under the rank-zero hypothesis. By Definition 259.7, the rank row satisfies $\nu\mathsf{hsamee}$, and δ is indexed only by positions licensed by ν together with the displayed $\mathsf{DirichletUnit}^{\uparrow}$ ledger β and the displayed $\mathsf{NumField}^{\uparrow}$ source ledger. The empty unary row has no positive basis position to license. Hence there is no row at which a logarithmic column, determinant minor, or host evaluation entry can be placed. The only finite determinant-layout compatible with the displayed index set is therefore the empty layout over the two dependency projections.

Finally use Definition 259.4. It says that all regulator determinant and logarithm data visible at the packet boundary are finite displayed ledger rows shaping the input matrix, not analytic values. Since the rank-zero layout is empty and the packet definition contains no field for an analytic regulator value, host determinant evaluation, host logarithm equality, or bridge row, none of those rows can be exported from $\mathcal{R}$. □

Chapter 260

A Concrete Naming Certificate for $\mathsf{Stack}^{\uparrow}$

The algebraic-stack interface $\mathsf{Stack}^{\uparrow}$ packages a sheaf of groupoids on the étale site of schemes satisfying descent and representability axioms (Artin / Deligne-Mumford). It depends on $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ (chapter 151) and $\mathsf{NameCert}_{\mathsf{Scheme}^{\uparrow}}$ (chapter 153).

260.1 Carrier definition

Theorem 260.1 (Stack BHist carrier obligation)**.** *A* $\mathsf{Stack}^{\uparrow}$ *carrier is a finite* BHist *packet containing a displayed* $\mathsf{Scheme}^{\uparrow}$ *site row, a displayed* $\mathsf{Sheaf}^{\uparrow}$ *groupoid-valued presheaf row over that site,* hsame *transport rows for object and arrow comparisons,* Cont *restriction routes for cover refinement, and* Pkg *provenance for the consumed dependency certificates. It contains no host stack object, no host quotient stack, and no external algebraic-space equality.*

Lean4 ✓ `BEDC.Derived.StackUp.StackBHistCarrier obligation surface`

Proof. Read the site coordinate through the $\mathsf{Scheme}^{\uparrow}$ certificate of chapter 153. The site row is a visible BEDC ledger row built from scheme carrier and cover data, so it is already a BHist-scoped input rather than a host site primitive.

Read the groupoid-valued presheaf coordinate through the locality and gluing surface of chapter 151. Its object rows, arrow rows, restriction rows, and classifier comparisons are the rows transported by hsame and routed by Cont over the displayed scheme site.

The remaining coordinates are provenance and transport rows: Pkg records which dependency packets are consumed, while hsame and Cont record replacement and restriction behavior. Since every coordinate has been listed in this finite packet, there is no carrier slot for a host stack object, a quotient stack, or external algebraic-space equality. □

Theorem 260.2 (Stack descent obligation)**.** *For a* $\mathsf{Stack}^{\uparrow}$ *carrier packet, accepted cover-compatible local object rows and arrow rows glue through the* $\mathsf{Sheaf}^{\uparrow}$ *locality and gluing ledger over the displayed* $\mathsf{Scheme}^{\uparrow}$ *site row, and the glued object and arrow rows remain inside the same* BHist *carrier packet.*

Lean4 ✓ `BEDC.Derived.StackUp.StackCarrierPacket descent rows`
Lean4 ✓ `BEDC.Derived.StackUp.StackCarrier descent obligation`
Lean4 ✓ `BEDC.Derived.StackUp.StackCarrierPacket descent transport`
Lean4 ✓ `BEDC.Derived.StackUp.StackBHistCarrier descent obligation`
Lean4 ✓ `BEDC.Derived.StackUp.StackCarrierPacket descent obligation`

Proof. Use the cover rows supplied by the $\mathsf{Scheme}^{\uparrow}$ site surface of chapter 153. They determine the visible cover indices and refinement routes on which the stack packet is allowed to restrict object and arrow rows.

Use the $\mathsf{Sheaf}^{\uparrow}$ locality and gluing ledger of chapter 151 on the groupoid-valued presheaf coordinate. Cover-compatible local object rows glue to a global object row, and compatible local arrow rows glue to a global arrow row, with the readback comparisons expressed by the displayed section classifiers.

The resulting rows are repacked with the original site, hsame transports, Cont routes, and Pkg provenance. Thus descent closes inside the same BHist carrier and does not appeal to host quotient stacks or external algebraic-space equality. □

Lean4 ✓ `BEDC.Derived.StackUp.StackDescent obligation surface`

Theorem 260.3 (Stack representability obligation)**.** *Every visible diagonal row or atlas row of a* $\mathsf{Stack}^{\uparrow}$ *carrier packet is readable only through displayed* $\mathsf{Scheme}^{\uparrow}$ *carrier rows and* hsame *transport over the site. Representability is recorded as a ledger predicate on those rows, not as an external algebraic-space primitive and not as host equality of quotient stacks.*

Lean4 ✓ `BEDC.Derived.StackUp.StackCarrierPacket representability surface`
Lean4 ✓ `BEDC.Derived.StackUp.StackRepresentability boundary`
Lean4 ✓ `BEDC.Derived.StackUp.StackRepresentability scheme site transport`

Proof. Project the diagonal or atlas read through the site coordinate supplied by chapter 153. Each visible source, target, overlap, and cover entry is a scheme-carrier or scheme-cover row, so the read has no independent host algebraic-space endpoint.

Project the presheaf and descent comparisons through the $\mathsf{Sheaf}^{\uparrow}$ rows of chapter 151. The relevant object and arrow comparisons are transported by hsame and restricted along the Cont routes already displayed in the stack carrier.

The representability witness is therefore a ledger predicate saying that the listed diagonal or atlas row is covered by the displayed scheme rows after the named transports. It is not an imported algebraic-space object and it does not identify quotient-stack presentations by external equality. □

Theorem 260.4 (Stack ledger obligation)**.** *The public ledger of a* $\mathsf{Stack}^{\uparrow}$ *carrier packet exports exactly the site rows, cover rows, groupoid-object rows, arrow rows, descent witnesses, representability witnesses,* Cont *routes, and dependency* Pkg *rows displayed in the packet.*

Lean4 ✓ `BEDC.Derived.StackUp.StackLedger obligation surface`

Proof. The site and cover exports are the rows inherited from the $\mathsf{Scheme}^{\uparrow}$ surface of chapter 153. They are the only rows that expose scheme carriers, affine or site cover entries, refinements, and the restriction routes consumed by the stack packet.

The groupoid-object, arrow, and descent exports are the rows inherited from the $\mathsf{Sheaf}^{\uparrow}$ surface of chapter 151. They account for the groupoid-valued presheaf coordinate and for the locality, gluing, and readback rows used by descent.

Representability rows, Cont routes, and Pkg provenance are the remaining coordinates of the carrier packet. Since the ledger is the ordered projection of these displayed coordinates, it exports no host stack object, no quotient-stack equality, and no external algebraic-space equality. □

Theorem 260.5 (Stack NameCert obligation surface)**.** *The preceding four obligations supply the* $\mathsf{Stack}^{\uparrow}$ NameCert *surface. Carrier inhabitance is witnessed by the displayed trivial-cover packet, classifier equivalence is componentwise* hsame*, carrier transport is governed by the descent and representability rows, and ledger exactness is the exact projection asserted by Definition 260.4.*

Lean4 ✓ `BEDC.Derived.StackUp.StackCarrierPacket namecert obligation surface`
Lean4 ✓ `BEDC.Derived.StackUp.StackNameCert obligation surface`

Proof. Carrier inhabitance uses the trivial cover available in the $\mathsf{Scheme}^{\uparrow}$ surface of chapter 153 together with the identity local object and arrow rows read through the $\mathsf{Sheaf}^{\uparrow}$ surface of chapter 151. These rows form the finite packet required by Definition 260.1.

Classifier equivalence is componentwise hsame on the site, object, arrow, transport, restriction, and provenance rows. Reflexive, symmetric, and transitive behavior is inherited row by row from the consumed scheme and sheaf classifiers, with Cont only recording the displayed restriction route.

Carrier transport follows by applying the descent obligation to glued object and arrow rows and the representability obligation to diagonal and atlas rows. Ledger exactness then follows from Definition 260.4, which lists every exported row and excludes host quotient stacks and external algebraic-space equality from the public surface. □

260.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{Stack}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Stack}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The public finite $\mathsf{Scheme}^{\uparrow}/\mathsf{Sheaf}^{\uparrow}$ stack packet binds the displayed stack carrier, descent rows, representability rows, ledger rows, NameCert obligation surface, hsame transports, Cont restriction routes, and Pkg provenance for the consumed dependency certificates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Stack}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.StackUp.StackCarrierPacket\_namecert\_obligation\_surface`

Bridge status: none

Constructive story: A $\mathsf{Stack}^{\uparrow}$ packet is assembled from displayed scheme-site rows, groupoid-valued sheaf rows, descent and restriction routes, and dependency provenance; every public read is a projection from this finite carrier packet.

Not claimed: This chapter does not close host quotient stacks or external algebraic-space equality.

Upgrade path: A standard bridge would have to identify this public stack packet with the acceptance-bridge schema without adding host quotient stacks or external algebraic-space equality.

Chapter 261

A Concrete Naming Certificate for $\mathsf{StoneDuality}^{\uparrow}$

The Stone-duality interface $\mathsf{StoneDuality}^{\uparrow}$ packages the contravariant equivalence between the category of Boolean algebras and the category of totally disconnected compact Hausdorff spaces. It depends on $\mathsf{NameCert}_{\mathsf{Lattice}^{\uparrow}}$ (chapter 55) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88), and is a cross-domain bridge witnessing the unified-kernel principle: order theory and topology meet through the same classifier discipline.

261.1 Carrier definition

Theorem 261.1 (StoneDualityUp Boolean-algebra source)**.** *A* $\mathsf{StoneDuality}^{\uparrow}$ *source row is a finite* BHist *packet whose Boolean-algebra side consists only of* $\mathsf{Lattice}^{\uparrow}$ *observations: zero and one rows, meet and join rows, complement rows, and the distributive ledgers that classify those rows by* hsame*. If two such packets are* hsame*-equivalent as* $\mathsf{Lattice}^{\uparrow}$ *packets, then their induced* Pkg *rows for* $\mathsf{StoneDuality}^{\uparrow}$ *have the same Boolean source classifier. No point of a Stone space, ultrafilter, or external compact Hausdorff carrier is part of the source data.*

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityBooleanSource transport`

Proof. Read the carrier from the finite BHist ledger. The zero, one, meet, join, complement, and distributive rows are already $\mathsf{Lattice}^{\uparrow}$-typed observations, so the $\mathsf{StoneDuality}^{\uparrow}$ source packet is obtained by repacking those rows into a single Pkg coordinate.

The classifier statement is just transport along hsame. Since each Boolean operation row is classified by the same finite history relation before and after repacking, the $\mathsf{StoneDuality}^{\uparrow}$ source classifier has no additional semantic input.

The last clause is a boundary condition on the carrier. The construction names only the finite observation packet and therefore never asks for an ultrafilter witness or a completed Stone-space object. □

Theorem 261.2 (StoneDualityUp Stone-space classifier)**.** *For every accepted* $\mathsf{StoneDuality}^{\uparrow}$ *Boolean source packet B, the topology-facing classifier is the finite* BHist *packet of clopen observation rows generated from the Boolean rows of B: a Boolean element b is read as a clopen row C_b, meet is read as intersection, join as union, complement as topological complement, zero as the empty clopen row, and one as the full clopen row. If b and b' are* hsame *on the Boolean side, then C_b and $C_{b'}$ are* hsame *as* $\mathsf{Topology}^{\uparrow}$ *clopen rows.*

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityStoneSpaceClassifier source semanticNameCert`

Proof. Construct the clopen packet by structural traversal of the finite Boolean source ledger. Each generator contributes one topology row, and each Boolean operation contributes the matching $\mathsf{Topology}^{\uparrow}$ operation row.

Classifier stability follows by induction over the source ledger. A generator step is copied directly; meet, join, complement, zero, and one steps use the corresponding $\mathsf{Topology}^{\uparrow}$ classifier rows and preserve the same hsame witness.

No representation by points is used. The topology packet is the clopen ledger itself, so the theorem is a classifier statement about finite rows rather than an existence theorem for a separate space. □

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityStoneSpaceClassifier clopen transport`

261.2 The certificate

Theorem 261.3 (StoneDualityUp clopen ledger)**.** *An accepted* $\mathsf{StoneDuality}^{\uparrow}$ *certificate carries a* Cont *ledger from Boolean source rows to clopen topology rows. For each finite Boolean observation row* r *in the source* Pkg*, the ledger records the clopen row* C_r*, its* $\mathsf{Topology}^{\uparrow}$ *classifier, and the* hsame *transport showing that Boolean equality of rows implies topological equality of the corresponding clopen rows. The ledger is exhausted by the rows generated in Theorem 261.2.*

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityClopenLedger transport closure`

Proof. Build the Cont ledger in parallel with the clopen classifier packet. Each source row contributes exactly one ledger entry, and each operation row contributes the corresponding continuity-preserving clopen operation entry.

Exhaustion is by inspection of the finite source traversal. The traversal has no branch for arbitrary open sets or arbitrary points; every ledger row is generated from a Boolean observation row already present in the source packet.

The transport clause is inherited from hsame stability of the Boolean rows and the $\mathsf{Topology}^{\uparrow}$ clopen classifier. Thus the Cont ledger records the constructive bridge without adding a new carrier. □

Theorem 261.4 (StoneDualityUp ultrafilter-free soundness)**.** *The* $\mathsf{StoneDuality}^{\uparrow}$ *soundness obligation is satisfied at the certificate surface when every accepted Boolean source row is read through the finite clopen ledger and every* hsame *comparison is transported by the recorded* Cont *rows. This soundness statement does not assert that every Boolean homomorphism has an ultrafilter point, and it does not invoke an external Stone-space existence theorem.*

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityUltrafilterFreeSoundness surface`

Proof. Start with an accepted Boolean source packet and its clopen ledger. Soundness means that each Boolean operation row has the matching clopen operation row and that classifier equality is preserved by the ledger.

The required checks are finite. Zero, one, meet, join, and complement are handled by their recorded clopen rows; distributive compatibility is handled by the copied $\mathsf{Lattice}^{\uparrow}$ classifier rows and the $\mathsf{Topology}^{\uparrow}$ operation classifiers.

There is no choice step. The proof never selects an ultrafilter extending a filter and never builds a point space; it only verifies that the displayed BHist, Pkg, NameCert, and Cont rows commute under hsame. □

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityUltrafilterFreeSoundness`

Theorem 261.5 (StoneDualityUp continuous-map readback)**.** *Let F be a finite* StoneDuality$^{\uparrow}$ *morphism row between accepted Boolean source packets. If F preserves zero, one, meet, join, and complement in the Boolean* BHist *classifier, then the readback row sends each clopen generator C_b in the target topology packet to the clopen generator $C_{F(b)}$ in the source topology packet, and the resulting* Cont *ledger is stable under* hsame*. Hence finite Boolean-algebra observations determine the contravariant continuous-map observation surface recorded by* NameCert*.*

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityContinuousMapReadback transport`

Proof. Read the morphism row contravariantly. A target clopen generator is named by a Boolean row b, and the Boolean morphism row supplies the source Boolean row $F(b)$.

Operation preservation gives continuity on the clopen basis represented in the ledger. The preimage of zero, one, meet, join, and complement is again recorded by the corresponding Boolean operation row, so the Topology$^{\uparrow}$ clopen classifier remains inside the generated packet.

Finally, hsame stability follows from the morphism classifier and the clopen ledger transport. The readback is therefore a finite Cont row in the StoneDuality$^{\uparrow}$ certificate, not an appeal to a completed dual space. □

Lean4 ✓ `BEDC.Derived.StoneDualityUp.StoneDualityContinuousMapReadback contravariant endpoint`

Closure status for NameCert$_{\text{StoneDuality}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{StoneDuality}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature certificate binds the Lattice$^{\uparrow}$ Boolean rows, Topology$^{\uparrow}$ clopen rows, finite BHist packets, hsame transport, Cont ledgers, Pkg provenance, NameCert rows, and the Stone-space classifier bridge through $BEDC.Derived.StoneDualityUp.StoneDualityStoneSpaceClassifier_source_semanticNameCert$.

Constructive story: The construction starts from finite Boolean-operation rows, traverses them into clopen topology rows, records the contravariant readback as Cont ledger entries, and transports every comparison through the displayed hsame, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{StoneDuality}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.StoneDualityUp.StoneDualityStoneSpaceClassifier\_source\_semanticNameCert`

Bridge status: paperBridge

Not claimed: This chapter does not close ultrafilter existence, completed compact Hausdorff spaces, choice-based point construction, or standard bridge closure.

Upgrade path: Standard bridge closure requires a bridge-checked transport from the finite Boolean-row to clopen-topology classifier surface into the acceptance-bridge schema.

Chapter 262

A Concrete Naming Certificate for $\mathsf{GelfandDuality}^{\uparrow}$

The Gelfand-duality interface $\mathsf{GelfandDuality}^{\uparrow}$ packages the contravariant equivalence between unital commutative C*-algebras and compact Hausdorff spaces, identifying a space with the spectrum of its continuous-function algebra. It depends on $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ (chapter 211) and $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ (chapter 88), and witnesses how operator-algebraic and topological certificates share a common spectrum substrate.

262.1 Carrier definition

Definition 262.1 (GelfandDuality BHist spectrum-pairing carrier)**.** *A* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum-pairing carrier is a finite* BHist *packet*

$$\mathcal{G} = (A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda).$$

The row A is a consumed $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *source row for a unital commutative C*-algebra history. The row X is a consumed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *compact-spectrum row. The finite row family χ lists the visible character histories from A to the scalar endpoint, and the finite row family ε lists the displayed character-evaluation histories relating elements of A to points of X.*

The rows ρ_A and ρ_X are the componentwise hsame *transport rows for the algebraic and topological faces of the same spectrum pairing. The row P is the* Pkg *provenance recording exactly the consumed* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *row, the consumed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row, the character rows, and the evaluation rows. The finite ledger λ is the* Cont *route containing only those displayed rows and their endpoint readbacks.*

Lean4 (definition) `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier`

Definition 262.2 (GelfandDuality spectrum transport package)**.** *A* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum transport package for Definition 262.1 is the displayed* BHist *packet*

$$\mathcal{T}_{\mathcal{G}} = (A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda).$$

The rows A and X are the transported algebra and compact-spectrum faces, χ and ε are the transported character and evaluation rows, ρ_A and ρ_X are the two componentwise hsame *transport rows, P is the* Pkg *provenance for the shared spectrum pairing, and λ is the* Cont *route carrying exactly those transported readbacks. The package exposes transport only through these displayed rows.*

Lean4 (definition) `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumTransportPackage`

262.2 The certificate

Theorem 262.3 (GelfandDuality spectrum source obligation)**.** *Every public source row of a* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum-pairing carrier is a displayed* BHist *row of Definition 262.1. In particular, the source boundary consists of the consumed* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *row* A*, the consumed* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row* X*, the character-evaluation rows, the componentwise* hsame *rows* ρ_A, ρ_X*, the* Pkg *row* P*, and the finite* Cont *route* λ*.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrum source obligation`
↪ variant `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier source obligation`

Proof. Project an accepted carrier to the tuple $\mathcal{G} = (A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda)$. The carrier definition names each public source row, so projection cannot expose a second algebra object, a second spectrum object, or an unrecorded evaluation row.

The algebraic source is the $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ row A, and the topological source is the $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ row X. The character and evaluation histories are rows over those two sources, while ρ_A, ρ_X, P, and λ provide transport, provenance, and readback.

Repacking these displayed projections gives the source obligation. The carrier therefore exports only the listed BHist rows and no hidden host-level Gelfand duality theorem. □

Theorem 262.4 (GelfandDuality CStarAlg consumer boundary)**.** *A downstream* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *consumer of a* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum-pairing carrier reads the C*-algebra face only through the* BHist *row* A*, the character-evaluation rows* χ *and* ε*, the algebra-side* hsame *row* ρ_A*, the shared* Pkg *row* P*, and the finite* Cont *route* λ*; the* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *row* X *is available to that consumer only as the recorded compact-spectrum dependency of the same packet.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier cstaralgebra consumer boundary`

Proof. Take a consumer whose input is the algebraic face of the carrier. Projecting the carrier exposes A as the only consumed $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ row and exposes χ and ε as the only visible character and evaluation-history rows.

The algebra-side transport available to the consumer is ρ_A. The provenance row P records that these algebraic rows are paired with the topological dependency X, and the continuation route λ records the readbacks that the consumer may follow.

No separate algebra carrier occurs in Definition 262.1. Thus the consumer boundary is exactly the projected algebraic face together with its recorded $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ dependency. □

Theorem 262.5 (GelfandDuality Topology consumer boundary)**.** *A downstream* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *consumer of a* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum-pairing carrier reads the compact-spectrum face only through the* BHist *row* X*, the character-evaluation rows* χ *and* ε*, the topology-side* hsame *row* ρ_X*, the shared* Pkg *row* P*, and the finite* Cont *route* λ*; the* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *row* A *is available to that consumer only as the recorded C*-algebra dependency of the same packet.*

Proof. Take a consumer whose input is the topological face of the carrier. Projection exposes X as the only consumed $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ compact spectrum row. The visible character and evaluation histories are the rows χ and ε, now read as the connection from the spectrum points back to the algebraic source.

The topology-side transport available to the consumer is ρ_X. The provenance row P records the dependence on A, and the continuation route λ records exactly the endpoint readbacks that remain public.

Since the carrier has no further topology row, no additional compact Hausdorff space or open-set basis can be imported by the consumer. The boundary is the displayed BHist spectrum face and its recorded $\mathsf{NameCert}_{\mathsf{CStarAlg}^\uparrow}$ dependency. □

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier topology consumer boundary`

Theorem 262.6 (GelfandDuality spectrum transport exactness)**.** *Every accepted transport of the spectrum pairing of Definition 262.2 projects exactly to the algebra face A, the topology face X, the character rows χ, the evaluation rows ε, the two* hsame *rows ρ_A and ρ_X, the* Pkg *provenance row P, and the continuation route λ. Conversely, those displayed rows repack as the transported* $\mathsf{GelfandDuality}^\uparrow$ *spectrum-pairing surface, with no second spectrum carrier, no hidden transform row, and no host compact-Hausdorff theorem.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumTransportPackage exactness`

Proof. Project the accepted transport to the tuple $\mathcal{T}_{\mathcal{G}} = (A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda)$. By Definition 262.2, these are exactly the rows carried by the transport package.

The algebraic projection is checked against Definition 262.4, and the topological projection is checked against Definition 262.5. These two boundaries force the character and evaluation rows to remain attached to the shared algebra-topology packet.

For the reverse direction, repack the displayed projections into the transport package. The provenance and continuation rows are already P and λ, so there is no remaining public slot for an external spectrum construction. □

Theorem 262.7 (GelfandDuality evaluation ledger determinacy)**.** *Let two accepted evaluation reads in a* $\mathsf{GelfandDuality}^\uparrow$ *spectrum transport package have the same transported algebra row A, the same topology point row inside X, the same character row in χ, and the same endpoint readback in ε. Then the public spectrum-pairing classifier identifies the two evaluation rows through the displayed* hsame *transport rows ρ_A, ρ_X and the finite* Cont *route λ. No second evaluation ledger can be exposed at that same transported slot.*

Proof. Project both evaluation reads through Definition 262.2. The shared algebra row, topology point row, character row, and endpoint readback place both reads in the same displayed ε slot.

Apply Definition 262.6. Every accepted evaluation read is one of the evaluation rows in the transported package, and the only available comparison rows are ρ_A, ρ_X, and the readback route λ.

Repacking those comparisons gives the public classifier identification. A second evaluation ledger would require another public row outside $\mathcal{T}_{\mathcal{G}}$, which the exactness theorem excludes. □

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier evaluation ledger determinacy`

Theorem 262.8 (GelfandDuality NameCert obligation surface)**.** *The public* $\mathsf{GelfandDuality}^\uparrow$ *obligation surface is exactly the coverage of the* BHist *rows A, X, χ, ε, ρ_A, ρ_X, P, and λ in Definition 262.1: A is covered by* $\mathsf{NameCert}_{\mathsf{CStarAlg}^\uparrow}$*, X is covered by* $\mathsf{NameCert}_{\mathsf{Topology}^\uparrow}$*, ρ_A and ρ_X are the only* hsame *transports, P is the only* Pkg *provenance, and λ is the only* Cont *readback route.*

Proof. Start with an accepted public obligation for the certificate. Projection to the carrier tuple determines whether the obligation is algebraic, topological, evaluational, transport, provenance, or readback data.

The algebraic case lands in A and is covered by $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$. The topological case lands in X and is covered by $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$. The evaluation case lands in χ or ε, so it remains tied to both displayed source rows.

The remaining cases are exact by inspection of the packet: ρ_A and ρ_X are the available hsame transports, P is the available Pkg provenance, and λ is the available Cont route. Repacking those projections gives the stated obligation surface, with no parameter-transport equivalence and no extra public row. □

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier namecert obligation surface`

Theorem 262.9 (GelfandDuality spectrum-pairing classifier stability)**.** *Componentwise* hsame *transport of the rows A, X, χ, ε, ρ_A, ρ_X, P, and λ in Definition 262.1 preserves the public* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum-pairing classifier. The transported packet reads the algebraic face through the same* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *dependency, the topological face through the same* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *dependency, and the evaluation face through the displayed character and evaluation rows.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier classifier stability`

Proof. Project the accepted packet to $\mathcal{G} = (A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda)$ using Definition 262.1. Transport acts on each displayed component and therefore cannot create a second algebra row, topology row, character family, evaluation family, provenance row, or readback route.

Apply Definition 262.4 to the algebraic face. The transported algebra consumer still reads only A, χ, ε, ρ_A, P, and λ.

Apply Definition 262.5 to the topological face. The transported topology consumer still reads only X, χ, ε, ρ_X, P, and λ.

Combining the two transported faces gives the same public spectrum-pairing classifier. No external Gelfand transform, compact-Hausdorff theorem, or parameter-transport equivalence is imported by the transport. □

Theorem 262.10 (GelfandDuality spectrum ledger exactness)**.** *The public* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum ledger is exactly the* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *source row A, the* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *compact-spectrum row X, the character rows χ, the evaluation rows ε, the algebra and topology transport rows ρ_A and ρ_X, the* Pkg *provenance row P, and the finite* Cont *readback route λ displayed in Definition 262.1.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier spectrum ledger exactness`

Proof. First use Definition 262.8. Every public obligation projects to one of the packet components A, X, χ, ε, ρ_A, ρ_X, P, and λ.

The algebraic side is exact by Definition 262.4; it permits only the CStarAlgUp source row, character and evaluation rows, algebra transport, provenance, and finite readback. The topological side is exact by Definition 262.5; it permits only the TopologyUp compact-spectrum row, the same character and evaluation rows, topology transport, provenance, and finite readback.

Conversely, each listed component is a field of the carrier packet, so it is a public ledger row of the certificate. The ledger therefore has no remaining slot for an external Gelfand transform, an imported compact-Hausdorff theorem, or any host-level spectrum construction. □

Theorem 262.11 (GelfandDuality obligation closure inventory)**.** *The* $\mathsf{GelfandDuality}^{\uparrow}$ *obligation inventory consists exactly of the* BHist *spectrum-pairing carrier rows* A, X, χ, ε, ρ_A, ρ_X, P, *and* λ, *the two consumer boundaries for the* $\mathsf{NameCert}_{\mathsf{CStarAlg}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Topology}^{\uparrow}}$ *faces, spectrum transport exactness, classifier stability, ledger exactness, and the NameCert obligation surface. All entries are mediated by* BHist, hsame, Pkg, *and* Cont, *and the inventory excludes host Gelfand transforms and compact-Hausdorff imports.*

Lean4 ✓ `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrumPairingCarrier obligation closure inventory`

Proof. The carrier rows are the displayed tuple of Definition 262.1. The source boundary is Definition 262.3, so every public source row is one of A, X, χ, ε, ρ_A, ρ_X, P, or λ.

The two consumer sides are exactly Definition 262.4 and Definition 262.5. Transport contributes Definition 262.6, which repacks only the displayed rows through the two hsame transports, the provenance row, and the finite continuation route.

The remaining public obligations are supplied by Definition 262.8, Definition 262.9, and Definition 262.10. Together these blocks cover the NameCert surface, classifier stability, and ledger exactness, while each block rules out any public slot for a host Gelfand transform, compact-Hausdorff import, or external spectrum construction. □

Theorem 262.12 (GelfandDuality standard spectrum bridge boundary)**.** *The standard bridge boundary for the public* $\mathsf{GelfandDuality}^{\uparrow}$ *spectrum packet is exactly the displayed pairing between the* $\mathsf{CStarAlg}^{\uparrow}$ *source row* A, *the compact-spectrum* $\mathsf{Topology}^{\uparrow}$ *row* X, *the character rows* χ, *the evaluation rows* ε, *the transport rows* ρ_A *and* ρ_X, *the* Pkg *provenance* P, *and the finite* Cont *route* λ.

This bridge boundary is only a conservative readback surface. It does not provide a host Gelfand transform, a compact-Hausdorff theorem, a second spectrum carrier, or any host-level spectrum equality.

Proof. Start with Definition 262.3. It fixes the source boundary of the packet as the displayed A, X, χ, ε, ρ_A, ρ_X, P, and λ rows.

The algebraic and topological consumer faces are Definition 262.4 and Definition 262.5. These two faces show that the $\mathsf{CStarAlg}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ consumers read the same finite pairing surface, each through its own source row and the shared character, evaluation, transport, provenance, and continuation rows.

Apply Definition 262.6 and Definition 262.10. Transport repacks only the displayed rows, and ledger exactness says that no extra spectrum component is present in the public packet.

Finally use Definition 262.11. The inventory lists precisely the obligations that the bridge boundary may expose, so the readback surface remains conservative and contains no host Gelfand transform, compact-Hausdorff theorem, second spectrum carrier, or host-level spectrum equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{GelfandDuality}^{\uparrow}}$**.**

Constructive story: A spectrum-pairing packet is generated by placing a unital commutative $\mathsf{CStarAlg}^{\uparrow}$ source row A beside a compact-spectrum $\mathsf{Topology}^{\uparrow}$ row X, then displaying character rows, evaluation rows, algebra-side and topology-side transports, provenance, and a finite continuation readback route. The carrier accepts only packets $(A, X, \chi, \varepsilon, \rho_A, \rho_X, P, \lambda)$ with those rows present; an external Gelfand transform, host compact-Hausdorff theorem, second spectrum carrier, or host-level spectrum equality is not a field of the packet. Two accepted packets classify together by componentwise hsame transport of the algebra face, topology face, character family, evaluation family, both transport rows, the Pkg provenance, and the finite Cont route. The source specification is the paired $\mathsf{CStarAlg}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ dependency surface, the pattern specification is the character-and-evaluation spectrum pairing, and the ledger keeps χ,

ε, ρ_A, ρ_X, P, and λ visible to algebraic and topological consumers. The five name-certificate obligations come from an accepted packet for inhabitation, the inherited component equivalences for reflexivity, symmetry, and transitivity, and the transport-exactness theorem that repacks every transported spectrum read into the same displayed rows. A packet history h lies in the signature gap of a GelfandDuality$^{\uparrow}$ token p when the source-domain accepts its algebra-topology spectrum surface, the signature relation pairs it with an answer spectrum packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, the character and evaluation rows are exactly the pairing data encoded by p. The chapter registers this spectrum-pairing packet under the name GelfandDuality$^{\uparrow}$; it introduces no standard compact-Hausdorff bridge, no Gelfand-transform bridge, and no host spectrum equality.

Theory closure: TheoryGate(NameCert$_{\text{GelfandDuality}\uparrow}$, maturePackageClosed)

Scope closed: The obligation scope binds Definition 262.12 together with the spectrum-pairing carrier rows A, X, χ, ε, ρ_A, ρ_X, P, and λ, the two consumer boundaries, transport exactness, classifier stability, ledger exactness, and the NameCert obligation surface.

Formal verification: FormalStatus(NameCert$_{\text{GelfandDuality}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.GelfandDualityUp.GelfandDualitySpectrum\_source\_obligation`

Bridge status: paperBridge

Not claimed: This chapter does not claim a standard compact-Hausdorff bridge, a Gelfand-transform bridge, or host-level spectrum equality.

Upgrade path: No further paper theory-axis step is recorded; bridge verification remains separate.

Chapter 263

A Concrete Naming Certificate for $\mathsf{PontryaginDuality}^{\uparrow}$

The Pontryagin-duality interface $\mathsf{PontryaginDuality}^{\uparrow}$ packages the contravariant self-equivalence on locally compact abelian groups via the Hom-into-circle functor, with the double-dual canonical isomorphism. It depends on $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ (chapter 204) and $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ (chapter 42), and bridges harmonic-analytic and group-theoretic certificates through the same character classifier.

263.1 Carrier definition

Definition 263.1 (Pontryagin-duality character carrier)**.** *A* $\mathsf{PontryaginDuality}^{\uparrow}$ *carrier is a finite* BHist *character-ledger packet over* $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$*, together with a circle-valued target row read through* $\mathsf{S}^{1\uparrow}$*-style component data. Its rows are the displayed topological-group source row, the displayed abelian-group operation row, finite character rows from the shared source to the circle target, pointwise product and inverse rows for those characters,* hsame *comparisons between corresponding source, target, and character rows,* Cont *provenance explaining how each character row is read from the displayed source and circle rows, and* Pkg *rows naming the two dependency certificates and the circle component packet.*

The carrier records only the BEDC character ledger. It does not include Haar measure, Fourier inversion, local compactness theorems, a completed dual group topology, or a full duality equivalence.

263.2 The certificate

Theorem 263.2 (Pontryagin-duality character carrier obligation)**.** *Every accepted* $\mathsf{PontryaginDuality}^{\uparrow}$ *packet is read from the character carrier of Definition 263.1: the* $\mathsf{TopGroup}^{\uparrow}/\mathsf{AbGroup}^{\uparrow}$ *source rows, the circle-valued target row, the finite character rows, the pointwise product and inverse rows, the displayed* hsame *comparisons, the* Cont *provenance, and the* Pkg *source rows. No additional analytic or host duality object is exposed.*

```
Lean4 ✓  BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier obligation
  ↪ variant  BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier source boundary
```

Proof. Project an accepted packet to the finite character ledger. The carrier definition lists exactly the source, target, character, algebraic operation, comparison, continuation, and package rows

available to a consumer.

Constructor inspection gives the converse readback boundary: every row named in the theorem is a coordinate of the finite BHist carrier, and the carrier has no coordinate for Haar measure, Fourier inversion, local compactness theorems, or a completed duality object. □

Theorem 263.3 (Pontryagin-duality character classifier stability)**.** *Let P and P' be accepted* $\mathsf{PontryaginDuality}^{\uparrow}$ *character-ledger packets. If their displayed* $\mathsf{TopGroup}^{\uparrow}$ *rows,* $\mathsf{AbGroup}^{\uparrow}$ *rows, circle-valued target rows, character rows, pointwise product and inverse rows,* Cont *provenance, and* Pkg *source rows are related componentwise by* hsame*, then the transported rows form another accepted character carrier with the same visible classifier surface.*

Lean4 ✓ `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier stability`

Proof. Transport first the dependency rows from $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ and the circle target row. This preserves the finite source and target boundary through displayed hsame witnesses.

Next transport each character row and its pointwise product and inverse rows. The operations are carrier rows in the same finite ledger, so their readback is preserved by projection along the transported row positions.

Finally transport the Cont and Pkg rows. The acceptance predicate only asks that characters be read through those visible provenance rows from the displayed source and circle target, so the transported packet is accepted again. □

Theorem 263.4 (Pontryagin-duality dual ledger exactness)**.** *The dual ledger exported by an accepted* $\mathsf{PontryaginDuality}^{\uparrow}$ *packet consists exactly of finite character rows into the circle-valued target, their pointwise product and inverse rows, the displayed source and target comparisons, and the visible* Cont/Pkg *provenance. It exports no Haar integration row, Fourier transform row, local compactness theorem, or double-dual equivalence.*

Lean4 ✓ `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier visible dual ledger boundary`
Lean4 ✓ `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier dual ledger rows`

Proof. Use Definition 263.2 to reduce readback to carrier projection. Every listed dual-ledger row is a coordinate of the finite packet.

For exactness, inspect the constructors of the packet. They generate character rows and their pointwise algebraic rows, together with source, target, comparison, continuation, and package provenance. They do not generate an analytic measure row, transform row, compactness theorem row, or equivalence row. □

Theorem 263.5 (Pontryagin-duality source boundary)**.** *The dependency boundary of* $\mathsf{PontryaginDuality}^{\uparrow}$ *is exactly the displayed* $\mathsf{NameCert}_{\mathsf{TopGroup}^{\uparrow}}$ *source row, the displayed* $\mathsf{NameCert}_{\mathsf{AbGroup}^{\uparrow}}$ *algebraic row, and the circle-valued target component read in the style of* $\mathsf{S}^{1\uparrow}$*. Every accepted character row is sourced through that boundary by visible* Cont *provenance and packaged by* Pkg*.*

Lean4 ✓ `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier dependency source boundary`

Proof. Read the source rows from Definition 263.1. The carrier names only the topological-group source, the abelian-group operation surface, and the circle target packet as external dependency data.

Each character row is accepted only when the displayed Cont row connects it to that boundary and the Pkg row records the same source package. There is no constructor that sources a character from a host locally compact abelian group object or an external harmonic-analysis structure. □

Theorem 263.6 (Pontryagin-duality NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{PontryaginDuality}^{\uparrow}}$ *obligation surface consists of the character carrier of Definition 263.1, carrier readback from Definition 263.2, classifier stability from Definition 263.3, dual-ledger exactness from Definition 263.4, and the dependency-source boundary of Definition 263.5.*

Proof. Use the character carrier as the source predicate and its displayed hsame rows as the classifier surface. Carrier readback supplies the visible packet rows, and classifier stability shows that this surface is preserved by componentwise transport.

Use dual-ledger exactness to identify the complete exported dual character ledger. Use the source-boundary theorem to bind every character row to the displayed $\mathsf{TopGroup}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, and circle target rows.

These clauses assemble the finite local NameCert surface. The assembly is deliberately limited to the BEDC packet and therefore excludes Haar measure, Fourier inversion, local compactness theorems, and a full Pontryagin-duality equivalence. □

Lean4 ✓ `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier namecert obligation surface`

Theorem 263.7 (Pontryagin-duality public character export)**.** *The scoped carrier, classifier-stability, dual-ledger exactness, source-boundary, and* NameCert *obligation-surface rows assemble the public* $\mathsf{NameCert}_{\mathsf{PontryaginDuality}^{\uparrow}}$ *export for downstream consumers. The export exposes only the displayed character rows, target-circle rows,* hsame, Cont, Pkg, *and* NameCert *provenance.*

It exposes no locally compact group theorem, Haar measure, biduality isomorphism, or host dual group.

Proof. Apply Definition 263.1 and Definition 263.2. These identify the character carrier and the displayed character rows available for readback.

Next apply Definition 263.3 and Definition 263.4. They transport the classifier surface and exhaust the dual character ledger by the displayed hsame, Cont, and Pkg rows.

Finally use Definition 263.5 and Definition 263.6. The source rows and local NameCert packet repack exactly the target-circle and provenance fields already named above, leaving no constructor for Haar measure, biduality, locally compact group theorems, or a host dual group. □

Closure status for $\mathsf{NameCert}_{\mathsf{PontryaginDuality}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PontryaginDuality}^{\uparrow}}$, publicClosed)

Scope closed: The scoped boundary binds the $\mathsf{TopGroup}^{\uparrow}$ source row, $\mathsf{AbGroup}^{\uparrow}$ source row, $\mathsf{S}^{1\,\uparrow}$-style circle target, finite character rows, pointwise product and inverse rows, hsame, Cont, and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PontryaginDuality}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PontryaginDualityUp.PontryaginDualityCharacterCarrier\_namecert\_obligation\_surface`

Bridge status: none

Constructive story: A finite character packet is assembled from the displayed topological-group and abelian-group rows, a circle-valued target row, finite character rows, transport witnesses, and one provenance packet; every accepted read remains inside that ledger.

Not claimed: This chapter does not claim Haar measure, Fourier inversion, completed dual topology, or double-dual equivalence.

Upgrade path: Bridged closure next requires a standard comparison bridge for the displayed finite character ledger.

Chapter 264

A Concrete Naming Certificate for $\mathsf{TannakaKrein}^{\uparrow}$

The Tannaka-Krein interface $\mathsf{TannakaKrein}^{\uparrow}$ packages the reconstruction of a compact group from its tensor category of finite-dimensional representations together with a fibre functor to vector spaces. It depends on $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140) and $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ (chapter 181), and bridges representation theory and category theory through the same fibre-functor classifier.

264.1 Carrier definition

Definition 264.1 (Tannaka-Krein fibre-functor carrier)**.** *A* $\mathsf{TannakaKrein}^{\uparrow}$ *carrier is a finite* BHist *reconstruction packet over the displayed dependency rows* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$*. Its visible rows are the Lie-group source certificate, the monoidal-category source certificate, a fibre-functor row from representation-object histories into vector-space histories, representation-object rows in the monoidal source, the tensor-unit comparison row, tensor-product comparison rows,* hsame *witnesses comparing transported group, category, functor, unit, product, and reconstruction-ledger endpoints,* Cont *provenance for the displayed reads, and* Pkg *provenance naming the two source certificates and the fibre-functor ledger.*

The carrier records only this BEDC reconstruction packet. It does not include Haar measure, analytic compactness, an external compact-group equality, or a full Tannaka duality theorem.

Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier dependency endpoint readback`
Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier endpoint expanded boundary`
Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier endpoint deterministic`
Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier reconstruction provenance endpoint confluence`

Definition 264.2 (Tannaka-Krein monoidal classifier)**.** *Two* $\mathsf{TannakaKrein}^{\uparrow}$ *reconstruction packets are identified by the local monoidal classifier exactly when the displayed* hsame *transport relates their Lie-group rows, monoidal-category rows, fibre-functor rows, representation-object rows, tensor-unit comparison rows, tensor-product comparison rows, reconstruction-ledger rows,* Cont *provenance rows, and* Pkg *source rows componentwise.*

No further equality of compact groups, representation categories, vector spaces, or analytic reconstruction objects is part of the classifier.

Lean4 (definition) `BEDC.Derived.TannakaKreinUp.TannakaKreinMonoidalClassifier`

264.2 The certificate

Theorem 264.3 (Tannaka-Krein source boundary)**.** *Every downstream consumer of a* $\mathsf{TannakaKrein}^{\uparrow}$ *packet may read only the displayed* $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ *source certificate, the displayed* $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ *source certificate, and the fibre-functor ledger of Definition 264.1. In particular, the consumer receives no Haar-measure row, no analytic compactness row, and no host compact-group object beyond those visible histories.*

Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier source boundary`

Proof. Project the packet to the carrier rows. The carrier definition lists the Lie-group source, monoidal-category source, fibre-functor row, representation-object rows, tensor-unit and tensor-product comparison rows, continuation rows, and package provenance.

The consumer boundary is therefore the displayed dependency pair together with the fibre-functor reconstruction ledger. There is no carrier coordinate for Haar measure, analytic compactness, or a host compact-group equality, so those objects cannot be read from the packet. □

Theorem 264.4 (Tannaka-Krein fibre-functor classifier stability)**.** *If two accepted* $\mathsf{TannakaKrein}^{\uparrow}$ *packets are related by the monoidal classifier of Definition 264.2, then transport along the displayed* hsame *rows preserves the carrier, the fibre-functor row, the representation-object rows, the tensor-unit comparison row, the tensor-product comparison rows, the reconstruction ledger, and the* Cont/Pkg *provenance rows.*

Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier classifier stability`

Proof. Transport the dependency rows first. The classifier supplies componentwise hsame witnesses for the $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ row and the $\mathsf{NameCert}_{\mathsf{MonoidalCat}^{\uparrow}}$ row, so the source boundary remains inside the same displayed pair of certificates.

Next transport the fibre-functor row, representation-object rows, tensor-unit comparison, tensor-product comparisons, and reconstruction-ledger row. These are exactly the remaining carrier coordinates, and each is related by a displayed hsame witness.

Finally transport the Cont and Pkg rows. Since acceptance asks only for the visible carrier rows and their provenance, the transported packet is again an accepted carrier with the same tensor comparison surface. □

Theorem 264.5 (Tannaka-Krein reconstruction ledger exactness)**.** *The reconstructed group token exported by a* $\mathsf{TannakaKrein}^{\uparrow}$ *packet is determined by the displayed fibre-functor row, representation-object rows, tensor-unit comparison row, tensor-product comparison rows, reconstruction ledger,* Cont *provenance, and* Pkg *source rows. It is not determined by, and does not expose, host compact-group equality or a full analytic Tannaka-duality equivalence.*

Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier reconstruction ledger exactness`

Proof. Unfold the carrier rows. The reconstruction token is read from the same finite BHist packet that records the fibre functor, its representation objects, the tensor-unit comparison, tensor-product comparisons, continuation routes, and package provenance.

The exactness claim is componentwise: changing any visible reconstruction row changes the packet boundary, while componentwise hsame transport of those rows preserves the token by Definition 264.4.

The carrier has no coordinate for host compact-group equality and no row for full Tannaka duality. Hence the exported token is determined by the displayed ledger rather than by an external analytic reconstruction theorem. □

Theorem 264.6 (Tannaka-Krein NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{TannakaKrein}^{\uparrow}}$ *obligation surface consists of the carrier of Definition 264.1, the monoidal classifier of Definition 264.2, classifier stability from Definition 264.4, the source boundary from Definition 264.3, and reconstruction-ledger exactness from Definition 264.5.*

Lean4 ✓ `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier semantic name certificate`

Proof. Use the carrier as the accepted-packet predicate and the monoidal classifier as the visible hsame comparison relation. The carrier supplies exactly the Lie-group source row, monoidal-category source row, fibre-functor row, representation-object rows, tensor comparison rows, continuation rows, and package provenance.

Classifier stability shows that the carrier and tensor comparison rows are preserved under componentwise transport. The source-boundary theorem limits all consumer reads to the two dependency certificates and the displayed fibre-functor ledger.

Reconstruction-ledger exactness identifies the exported group token with the visible ledger. The assembled obligation surface is therefore constructive and finite, using only Cont, Pkg, NameCert, $\mathsf{LieGroup}^{\uparrow}$, and $\mathsf{MonoidalCat}^{\uparrow}$ rows, while excluding Haar measure, analytic compactness, and full Tannaka duality. □

Closure status for $\mathsf{NameCert}_{\mathsf{TannakaKrein}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TannakaKrein}^{\uparrow}}$, publicClosed)

Scope closed: The public surface binds the $\mathsf{LieGroup}^{\uparrow}$ source rows, $\mathsf{MonoidalCat}^{\uparrow}$ tensor rows, fibre-functor rows, representation rows, unit and tensor-product rows, reconstruction ledger, hsame classifier transport, Cont reconstruction routes, and Pkg provenance into one finite dependency packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TannakaKrein}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TannakaKreinUp.TannakaKreinFiberFunctorCarrier\_semantic\_name\_certificate`

Bridge status: none

Constructive story: A $\mathsf{TannakaKrein}^{\uparrow}$ packet is assembled from the two source certificates, the fibre-functor row, representation-object rows, tensor-unit and tensor-product comparison rows, reconstruction ledger endpoints, and visible provenance rows; consumers read only this finite reconstruction surface.

Not claimed: This chapter does not claim Haar measure, analytic compactness, host compact-group equality, or full Tannaka duality.

Upgrade path: The next paper-side step is a bridge theorem connecting the public reconstruction export to the standard Tannaka-Krein schema.

Chapter 265

A Concrete Naming Certificate for $\mathsf{CurryHoward}^{\uparrow}$

The Curry-Howard interface $\mathsf{CurryHoward}^{\uparrow}$ packages the bijective correspondence between intuitionistic propositions and types, between proofs and programs, and between cut-elimination and reduction. It depends on $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ (chapter 197) and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$ (chapter 200), and bridges logic and computation through the same inductive carrier.

265.1 Carrier definition

265.2 The certificate

Definition 265.1 (CurryHowardUp shared cut-beta packet)**.** *A shared cut-beta packet for* $\mathsf{CurryHoward}^{\uparrow}$ *is a displayed* BHist *tuple*

$$\mathcal{H} = (P, A, B, D_1, D_2, D_{\mathsf{cut}}, x, a, b, u, L, r, s, \chi, \rho)$$

with the following rows. The histories P, A, B are the shared proof-program, cut-formula/type, and conclusion-type endpoints. The rows D_1 and D_2 are accepted finite $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledgers over one carried syntax packet from Definition 197.1; D_1 has final endpoint A, and D_2 has A as the displayed initial premise row, so the pair is composable in the sense of Definition 197.9. The row D_{cut} is an accepted* $\mathsf{FirstOrder}^{\uparrow}$ *deduction ledger with the same open-premise rows as D_1 followed by D_2 and final endpoint B, and the comparison between $D_1 +\!\!+ D_2$ and D_{cut} is carried only by the* $\mathsf{FirstOrder}^{\uparrow}$ *deduction classifier of Definition 197.2.*

The row L is an accepted $\mathsf{LambdaCalc}^{\uparrow}$ *term packet in the sense of Definition 200.5. The history x is a displayed unary binder row, a is the lambda-term endpoint classified with the proof ledger D_1,* $\mathsf{lamAbs}(x, b)$ *is the lambda-term endpoint classified with the proof ledger D_2, and u is the lambda-term endpoint classified with D_{cut}. The row r is the displayed beta-redex source*

$$\mathsf{lamApp}(\mathsf{lamAbs}(x, b), a),$$

and s is the capture-avoiding substitution ledger that substitutes a for the unary binder x in b with output endpoint u. The classifier row χ is shared by the two sides: it relates D_1 with a, D_2 with $\mathsf{lamAbs}(x, b)$*, and D_{cut} with u by the same displayed* hsame, Cont*, and provenance rows. The provenance row ρ ties these rows to* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*. No host proof tree, host lambda syntax equality, or quotient of alpha classes is a field of the packet.*

Theorem 265.2 (CurryHowardUp cut-beta bridge obligation)**.** *Let $\mathcal{H}$ be an accepted shared cut-beta packet for* $\mathsf{CurryHoward}^{\uparrow}$ *as in Definition 265.1. If the* $\mathsf{FirstOrder}^{\uparrow}$ *component displays the cut-elimination step from $D_1 +\!\!+ D_2$ to D_{cut}, then the* $\mathsf{LambdaCalc}^{\uparrow}$ *component displays an accepted beta-reduction row whose source is* $\mathsf{lamApp}(\mathsf{lamAbs}(x, b), a)$ *and whose substitution endpoint is u. Under the shared classifier χ, this beta/substitution endpoint is the same proof-program endpoint as the cut-eliminated deduction ledger D_{cut}.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket bridge obligation`

Proof. Unpack the accepted packet $\mathcal{H}$. The $\mathsf{FirstOrder}^{\uparrow}$ side supplies two accepted ledgers D_1 and D_2 over the same syntax carrier, with the final endpoint of D_1 displayed as the initial premise row of D_2. By Definition 197.9, this is exactly the composability datum for the concatenated ledger $D_1 +\!\!+ D_2$.

Apply Definition 197.10. The concatenation is an accepted finite deduction ledger over the same $\mathsf{Set}^{\uparrow}$ and $\mathsf{Tree}^{\uparrow}$ dependency certificates, and the join reads no extra cut formula outside the shared endpoint row A. The displayed cut-elimination row in $\mathcal{H}$ then compares this concatenated ledger with D_{cut} through the deduction classifier. By Definition 197.5 and Definition 197.4, every premise endpoint, conclusion endpoint, and transport row of D_{cut} is therefore a visible $\mathsf{FirstOrder}^{\uparrow}$ carrier or ledger row.

Now read the $\mathsf{LambdaCalc}^{\uparrow}$ side. The term packet L exposes the carried syntax row, binder-index row, beta-redex ledger, substitution ledger, continuation ledger, and provenance listed in Definition 200.5. The redex row r has the displayed source $\mathsf{lamApp}(\mathsf{lamAbs}(x, b), a)$, where the shared classifier χ identifies a with D_1 and $\mathsf{lamAbs}(x, b)$ with D_2.

Apply Definition 200.12 to the substitution row s. It may read only the displayed $\mathsf{Tree}^{\uparrow}$ syntax row, the $\mathsf{Nat}^{\uparrow}$ binder-index row, and the capture-avoidance Cont ledger, so substituting a for x in b produces the displayed endpoint u without importing a host substitution function. By Definition 200.16, the output endpoint u remains a carried LambdaCalc term endpoint.

Finally apply Definition 200.13. The redex row r, the substitution ledger s, the endpoint u, and the provenance row ρ repack as an accepted beta-reduction classifier row, stable under the displayed hsame transport. The shared classifier χ identifies this endpoint u with the proof-program endpoint carried by D_{cut}, while its other two clauses identify the redex components with D_1 and D_2. Hence the displayed deduction-side cut-elimination step is carried to the displayed beta-reduction/substitution ledger under the shared $\mathsf{CurryHoward}^{\uparrow}$ classifier, and no host proof equality or host lambda-term equality enters the bridge. □

Definition 265.3 (CurryHowardUp endpoint classifier)**.** *For accepted shared cut-beta packets $\mathcal{H}$ and $\mathcal{H}'$ as in Definition 265.1, the CurryHowardUp endpoint classifier relates $\mathcal{H}$ and $\mathcal{H}'$ when the following rows hold simultaneously. The* $\mathsf{FirstOrder}^{\uparrow}$ *deduction classifier relates the displayed deduction ledgers $D_1, D_2, D_{\mathsf{cut}}$ of $\mathcal{H}$ with the corresponding ledgers $D'_1, D'_2, D'_{\mathsf{cut}}$ of $\mathcal{H}'$. The* $\mathsf{LambdaCalc}^{\uparrow}$ *beta/substitution classifier relates the displayed term packets, beta-redex rows, substitution ledgers, and output endpoints u, u'. The shared endpoint rows satisfy the displayed* hsame *clauses for the proof-program endpoint P, the cut-formula/type endpoint A, and the conclusion-type endpoint B. The finite provenance row ρ records the common dependence on* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*, with no host proof equality, host lambda equality, or quotient of alpha classes.*

Theorem 265.4 (CurryHoward cut-beta source row separation)**.** *For every accepted shared cut-beta packet $\mathcal{H}$ for* $\mathsf{CurryHoward}^{\uparrow}$*, the* $\mathsf{FirstOrder}^{\uparrow}$ *deduction rows $D_1, D_2, D_{\mathsf{cut}}$ and the* $\mathsf{LambdaCalc}^{\uparrow}$ *beta/substitution rows L, r, s, u project into disjoint source slots. The only common public rows are*

the proof-program BHist *endpoint, the displayed* hsame *endpoint clauses, the finite* Cont *provenance, and the two source rows* $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$.

Consequently the cut-beta bridge identifies neither host proof trees with host lambda terms nor host deduction syntax with host term syntax. It relates the two source rows only through the displayed CurryHoward endpoint and provenance surface.

Proof. Unpack the shared cut-beta packet of Definition 265.1. Its $\mathsf{FirstOrder}^\uparrow$ rows are the deduction ledgers $D_1, D_2, D_{\mathsf{cut}}$, while its $\mathsf{LambdaCalc}^\uparrow$ rows are the term packet, beta-redex row, substitution ledger, and output endpoint. These rows occupy different named fields of the displayed BHist tuple.

Now unfold the endpoint classifier of Definition 265.3. The classifier has one $\mathsf{FirstOrder}^\uparrow$ deduction clause, one $\mathsf{LambdaCalc}^\uparrow$ beta/substitution clause, shared hsame endpoint clauses, and the finite provenance row over the two source certificates. It contains no clause equating a host proof tree with a host lambda term.

Apply Definition 265.5. The common information transported by the bridge is the proof-program endpoint under the displayed hsame and Cont rows.

Finally use Definition 265.8. It says that the classifier surface is exhausted by those displayed deduction, beta/substitution, endpoint, transport, provenance, and source-certificate rows. Hence the source rows remain separated except at the stated public endpoint and provenance surface. □

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket source row separation`

Theorem 265.5 (CurryHowardUp classifier endpoint transport)**.** *Let $\mathcal{H}$ and $\mathcal{H}'$ be accepted shared cut-beta packets for* $\mathsf{CurryHoward}^\uparrow$*. If the CurryHowardUp endpoint classifier of Definition 265.3 relates $\mathcal{H}$ and $\mathcal{H}'$, then the cut-eliminated deduction endpoint and the beta-substitution endpoint transport to the same proof-program endpoint under the shared* hsame *rows.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket classifier endpoint transport`

Proof. Unpack the endpoint classifier. Its $\mathsf{FirstOrder}^\uparrow$ row gives deduction-classifier agreement for the two cut sources, the two cut targets, and the two cut-eliminated ledgers. Apply Definition 197.5 to read the displayed endpoints of D_{cut} and D'_{cut}, and then apply Definition 197.4 to transport those endpoints through the deduction classifier.

Next use the $\mathsf{LambdaCalc}^\uparrow$ row. Apply Definition 200.16 to the two substitution ledgers, so both substitution outputs remain carried LambdaCalc term endpoints. Apply Definition 200.13 to the related beta-redex and substitution rows; the classifier transports u and u' through the same displayed beta/substitution endpoint relation.

It remains to compare the two sides of the bridge. The shared hsame clauses in the endpoint classifier identify the proof-program endpoint, the cut-formula/type endpoint, and the conclusion-type endpoint on the two packets, while the finite Cont provenance row ρ keeps both transports over $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$. Repacking these clauses gives one proof-program endpoint for the cut-eliminated deduction side and the beta-substitution side. □

Theorem 265.6 (CurryHowardUp public NameCert boundary)**.** *The shared cut-beta packet of Definition 265.1, the endpoint classifier of Definition 265.3, the endpoint transport theorem Definition 265.5, and the no-host-equality ledger in the packet definition form the public CurryHowardUp certificate boundary over* $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$.

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket public namecert boundary`

Proof. The carrier part of the boundary is the accepted shared cut-beta packet. It lists the $\mathsf{FirstOrder}^{\uparrow}$ deduction ledgers, the $\mathsf{LambdaCalc}^{\uparrow}$ term packet, the beta-redex row, the substitution ledger, the shared classifier row, and the provenance row as displayed BHist data.

The classifier part is Definition 265.3. It requires agreement on the deduction classifier, beta/substitution classifier, shared hsame endpoint rows, and the finite Cont provenance row. These clauses are all stated over the two source certificates $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$.

The stability and endpoint-exactness part is Definition 265.5. The theorem transports the cut-eliminated deduction endpoint and the beta-substitution endpoint to the same proof-program endpoint, so the public boundary has a certificate-level endpoint row rather than a host equality assertion.

Finally, the packet definition explicitly excludes host proof trees, host lambda syntax equality, and quotients of alpha classes. This no-host-equality ledger is the remaining boundary clause, and the four clauses together give the stated public CurryHowardUp certificate boundary. □

Theorem 265.7 (CurryHoward cut-beta proof-program determinacy)**.** *Let $\mathcal{H}$ and $\mathcal{H}'$ be accepted shared cut-beta packets for* $\mathsf{CurryHoward}^{\uparrow}$ *as in Definition 265.1. Suppose their* $\mathsf{FirstOrder}^{\uparrow}$ *cut-elimination rows,* $\mathsf{LambdaCalc}^{\uparrow}$ *beta-substitution rows, shared* hsame *endpoint rows, and finite* Cont *provenance rows are the same displayed rows under the CurryHoward endpoint classifier of Definition 265.3. Then the cut-eliminated deduction endpoint and the beta-substitution endpoint determine one proof-program endpoint:*

$$(\mathsf{BHist}_{D_{\mathsf{cut}}}, \mathsf{BHist}_u, \mathsf{hsame}_P, \mathsf{Cont}_\rho).$$

This endpoint is a displayed CurryHoward certificate row, not a host equality between proof trees or lambda terms.

Proof. Unpack the two accepted packets by Definition 265.1. The displayed cut-elimination rows fix the $\mathsf{FirstOrder}^{\uparrow}$ ledgers $D_1, D_2, D_{\mathsf{cut}}$ and $D'_1, D'_2, D'_{\mathsf{cut}}$, while the displayed beta-substitution rows fix the redex, substitution ledger, and output endpoints u, u'.

Apply Definition 197.5 to the two cut-eliminated ledgers. Their endpoints are read from the visible deduction rows, so no external proof-tree equality is available or needed. Apply Definition 200.13 to the two beta-substitution rows. Their substitution outputs are accepted beta-reduction classifier endpoints, not host lambda-term quotients.

Now apply Definition 265.5. The CurryHoward endpoint classifier transports the deduction endpoint and the beta-substitution endpoint through the same shared hsame proof-program row and the same finite Cont provenance over $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$.

The common endpoint is therefore obtained by repacking the displayed rows as

$$(\mathsf{BHist}_{D_{\mathsf{cut}}}, \mathsf{BHist}_u, \mathsf{hsame}_P, \mathsf{Cont}_\rho).$$

This is the single proof-program endpoint determined by both readings. The conclusion is a statement about the CurryHoward classifier data itself, so it does not assert equality in the host proof language. □

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket proof program determinacy`

Theorem 265.8 (CurryHoward cut-beta classifier exhaustion)**.** *Every accepted CurryHoward cut-beta endpoint classifier witness of Definition 265.3 projects exactly to the* $\mathsf{FirstOrder}^{\uparrow}$ *deduction classifier rows, the* $\mathsf{LambdaCalc}^{\uparrow}$ *beta/substitution classifier rows, the shared* BHist *proof-program*

endpoint rows, the displayed hsame *transports, finite* Cont *provenance, and the two source certificates* $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$. *Conversely, those rows repack into the same accepted CurryHoward endpoint classifier witness.*

Thus the cut-beta classifier surface is exhausted by displayed deduction, program, endpoint, transport, provenance, and source-certificate rows. It does not read host proof equality, host lambda-term equality, or an alpha-quotient.

Proof. Project an accepted witness through Definition 265.3. Its first clause is precisely the $\mathsf{FirstOrder}^\uparrow$ deduction classifier comparison for $D_1, D_2, D_{\mathsf{cut}}$ and the corresponding rows of the second packet. Its second clause is the $\mathsf{LambdaCalc}^\uparrow$ beta/substitution classifier comparison for the term packets, redex rows, substitution ledgers, and output endpoints.

The remaining clauses of the same definition expose the shared BHist proof-program endpoint rows, the displayed hsame transports for P, A, B, and the finite Cont provenance row ρ. The provenance row names exactly $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$ as the source certificates for the comparison.

Apply Definition 265.5. The projected deduction and beta/substitution rows transport to the same proof-program endpoint under those displayed hsame and Cont rows.

Apply Definition 265.7. The transported endpoint is the displayed CurryHoward proof-program row, so the classifier has no further host equality component to project.

Conversely, given exactly the rows listed above, rebuild the witness by the clauses of Definition 265.3. The transport theorem and determinacy theorem verify that the repacked deduction-side and program-side endpoints are the same displayed proof-program endpoint, which closes the classifier without adding any external quotient. □

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket classifier exhaustion`

Theorem 265.9 (CurryHoward cut-beta endpoint exactness)**.** *Every accepted shared cut-beta packet for* $\mathsf{CurryHoward}^\uparrow$ *projects the cut-eliminated* $\mathsf{FirstOrder}^\uparrow$ *deduction endpoint and the* $\mathsf{LambdaCalc}^\uparrow$ *beta-substitution endpoint to the same displayed proof-program* BHist *row under the shared* hsame *endpoint clauses and finite* Cont *provenance. Conversely, exactly those displayed deduction, beta-substitution, proof-program,* hsame, Cont*, and source-certificate rows repack into the CurryHoward endpoint classifier of Definition 265.3.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket endpoint exactness`

Proof. For the projection direction, start with an accepted shared cut-beta packet. Apply Definition 265.2. It identifies the cut-eliminated deduction ledger D_{cut} with the accepted beta-reduction and substitution endpoint u through the shared classifier row.

Next apply Definition 265.5. The deduction-side endpoint and the program-side endpoint transport along the same displayed hsame proof-program row, with the finite Cont provenance tied to $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$.

Now apply Definition 265.8. It projects the endpoint classifier witness exactly to the deduction classifier rows, beta/substitution classifier rows, shared BHist endpoint rows, hsame transports, finite Cont provenance, and the two source certificates.

For the converse direction, assume exactly those rows are displayed. The exhaustion theorem repacks them by the clauses of Definition 265.3, and the bridge and transport theorems verify that the repacked deduction endpoint and beta-substitution endpoint are the same CurryHoward proof-program row. □

Definition 265.10 (CurryHoward cut-beta obligation package)**.** *The cut-beta obligation package for* $\mathsf{CurryHoward}^{\uparrow}$ *is the finite row family*

$$\mathcal{O}_{\mathsf{cut}\beta} = (\mathcal{H}, \Xi_{\mathsf{src}}, \Xi_{\mathsf{cls}}, \Xi_{\mathsf{end}}, \Lambda_{\mathsf{pp}}, \rho),$$

where $\mathcal{H}$ *is the shared cut-beta packet of Definition 265.1,* Ξ_{src} *is the source surface consisting of the* $\mathsf{FirstOrder}^{\uparrow}$ *ledgers* $D_1, D_2, D_{\mathsf{cut}}$*, the* $\mathsf{LambdaCalc}^{\uparrow}$ *rows* L, r, s, u*, the shared* BHist *endpoints* P, A, B*, and the source certificates* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*,* Ξ_{cls} *is the endpoint classifier of Definition 265.3,* Ξ_{end} *is the cut-beta endpoint row supplied by Definition 265.9,* Λ_{pp} *is the proof-program ledger consisting of the displayed* hsame*,* Cont*, and* Pkg *rows, and* ρ *records the finite provenance over the two source certificates.*

The package exports only displayed BHist *source rows, classifier rows, endpoint rows, and ledger rows. It does not export host proof equality, host lambda-term equality, an alpha quotient, normalization, or categorical semantics.*

Theorem 265.11 (CurryHoward cut-beta source obligation surface)**.** *Every accepted package* $\mathcal{O}_{\mathsf{cut}\beta}$ *of Definition 265.10 projects exactly to the* BHist *source surface consisting of* $D_1, D_2, D_{\mathsf{cut}}$*,* L, r, s, u*, the shared proof-program, cut-formula/type, and conclusion-type endpoints* P, A, B*, the displayed* hsame *and* Cont *rows, and the two source certificates* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*. Conversely, those source rows repack as the source component of the package.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket source obligation surface`

Proof. Project the package to its shared packet coordinate $\mathcal{H}$. By Definition 265.1, the first-order side supplies exactly the deduction ledgers $D_1, D_2, D_{\mathsf{cut}}$, while the lambda side supplies the term packet L, the beta-redex row r, the substitution ledger s, and the output endpoint u.

Apply Definition 265.4. It separates the two source families and states that their only common public rows are the proof-program endpoint, displayed hsame endpoint clauses, finite Cont provenance, and the two source $\mathsf{NameCert}$ rows.

Now apply Definition 265.2. The bridge obligation confirms that the cut-elimination source and beta-reduction source meet at the displayed endpoint surface, not at a host proof tree or host lambda syntax equality.

For the converse direction, assume exactly the listed source rows. The packet definition repacks them as $\mathcal{H}$ and then as Ξ_{src} inside $\mathcal{O}_{\mathsf{cut}\beta}$, with no additional source coordinate. □

Theorem 265.12 (CurryHoward endpoint classifier obligation surface)**.** *The classifier component of every accepted cut-beta obligation package is exhausted by the* $\mathsf{FirstOrder}^{\uparrow}$ *deduction classifier rows, the* $\mathsf{LambdaCalc}^{\uparrow}$ *beta/substitution classifier rows, the shared* BHist *endpoint rows, the displayed* hsame *transports, finite* Cont *provenance, and the source certificates* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*. These rows repack into the same endpoint classifier and determine the common cut-beta endpoint.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket endpoint classifier obligation surface`

Proof. Start from the classifier component Ξ_{cls}. Unfold Definition 265.3. Its clauses are exactly the $\mathsf{FirstOrder}^{\uparrow}$ deduction classifier, the $\mathsf{LambdaCalc}^{\uparrow}$ beta/substitution classifier, the shared endpoint hsame rows, and the finite provenance over the two source certificates.

Apply Definition 265.8. It proves that there are no further classifier fields beyond those deduction, beta/substitution, endpoint, transport, provenance, and source-certificate rows.

Next apply Definition 265.5 and Definition 265.9. These blocks transport the cut-eliminated deduction endpoint and the beta-substitution endpoint to the same displayed proof-program row.

The reverse packaging is the converse half of the classifier exhaustion and endpoint exactness blocks: the displayed rows rebuild the endpoint classifier, and the transport rows determine the common cut-beta endpoint without adding host equality or an alpha quotient. □

Theorem 265.13 (CurryHoward proof-program ledger obligation surface)**.** *The proof-program ledger component of Definition 265.10 consists exactly of the displayed* BHist *proof-program endpoint, the cut-eliminated* $\mathsf{FirstOrder}^{\uparrow}$ *endpoint, the* $\mathsf{LambdaCalc}^{\uparrow}$ *beta-substitution endpoint, the shared* hsame *endpoint clauses, the finite* Cont *provenance, the* Pkg *row, and the source certificates* $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$*. This ledger determines one CurryHoward proof-program endpoint and exports no normalization theorem, host proof equality, host lambda equality, or categorical semantics.*

Proof. Project the ledger component Λ_{pp} from the obligation package. The displayed rows are the proof-program endpoint, the cut-eliminated deduction endpoint, the beta-substitution endpoint, the shared hsame clauses, the finite Cont provenance, and the Pkg row over the two source certificates.

Apply Definition 265.7. With the cut-elimination rows, beta-substitution rows, shared hsame endpoint rows, and finite Cont provenance fixed, the deduction endpoint and the beta-substitution endpoint determine one displayed proof-program endpoint.

Use Definition 265.9 to identify this endpoint as the exact endpoint of the accepted shared cut-beta packet. Then use Definition 265.6 to keep the ledger inside the certificate boundary over $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$.

Since the public boundary and the package definition name no normalization operator, host equality, alpha quotient, or categorical semantic object, the ledger surface is exhausted by the displayed proof-program rows. □

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket proof program ledger obligation surface`

Closure status for $\mathsf{NameCert}_{\mathsf{CurryHoward}^{\uparrow}}$**.**

Constructive story: A Curry-Howard packet is generated by placing $\mathsf{FirstOrder}^{\uparrow}$ deduction ledgers beside $\mathsf{LambdaCalc}^{\uparrow}$ term, beta-redex, and substitution ledgers, then tying them to shared proof-program, cut-formula/type, and conclusion-type endpoint histories. The carrier accepts only shared cut-beta packets whose first-order side displays composable deduction ledgers and a cut-eliminated endpoint, whose lambda side displays the corresponding application of an abstraction to an argument and the capture-avoiding substitution output, and whose common endpoint and provenance are carried by hsame, finite Cont, and the two source certificates. Two packets are classified by the first-order deduction classifier, the lambda beta/substitution classifier, and the shared endpoint rows, so the bridge never identifies host proof trees with host lambda terms or quotients alpha classes. The source is the displayed cut-beta packet over $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^{\uparrow}}$, the pattern records the cut-beta bridge obligation, source-row separation, endpoint transport, public boundary, proof-program determinacy, classifier exhaustion, endpoint exactness, and proof-program ledger surface, and the ledger keeps deduction endpoints, program endpoints, substitution output, shared hsame, Cont, and Pkg provenance visible. The five name-certificate obligations at this seed level are represented by an accepted shared packet for carrier inhabitation, by the two inherited classifiers for reflexivity, symmetry, and transitivity, and by endpoint exactness and classifier exhaustion for closure under classified transport. A history h lies in the signature gap of a $\mathsf{CurryHoward}^{\uparrow}$ token p when the source-domain accepts its cut-beta packet, the signature relation pairs it with an answer packet and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h carries the same deduction ledgers, beta/substitution rows, proof-program endpoint, and provenance encoded by p. The chapter registers this shared cut-beta construction under the name $\mathsf{CurryHoward}^{\uparrow}$; normalization, full Curry-Howard equivalence, host proof equality, host lambda equality, alpha quotient, categorical semantics, and standard bridge data are not introduced.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CurryHoward}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope is the shared cut-beta packet, cut-beta bridge obligation, public NameCert boundary, endpoint exactness rows, proof-program ledger surface, Definition 265.14 as the displayed consumer classifier

row, $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$ source certificates, BHist proof-program endpoints, hsame endpoint rows, finite Cont provenance, and Pkg rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CurryHoward}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket\_endpoint\_classifier\_reflexive`

Bridge status: none

Not claimed: This chapter does not close a full Curry-Howard equivalence, normalization theorem, host proof equality, host lambda-term equality, alpha quotient, categorical semantics, or standard bridge data.

Upgrade path: No further paper theory-axis grade remains; bridgeChecked status is a formal-axis matter for the displayed cut-beta surface.

Theorem 265.14 (CurryHoward cut-beta endpoint classifier reflexivity)**.** *Let*

$$\mathcal{H} = (P, A, B, D_1, D_2, D_{\mathsf{cut}}, x, a, b, u, L, r, s, \chi, \rho)$$

be an accepted shared cut-beta packet for $\mathsf{CurryHoward}^\uparrow$ *as in Definition 265.1. Then the CurryHowardUp endpoint classifier of Definition 265.3 relates* $\mathcal{H}$ *to itself. The witnessing classifier row is obtained from the rowwise reflexive* hsame *comparisons on* $P, A, B, D_1, D_2, D_{\mathsf{cut}}, L, r, s, u$ *together with the unchanged finite* Cont *and provenance row* ρ*.*

Lean4 ✓ `BEDC.Derived.CurryHowardUp.CurryHowardCutBetaPacket endpoint classifier reflexive`

Proof. Unpack the accepted packet of Definition 265.1. Its first-order coordinates are the accepted deduction ledgers $D_1, D_2, D_{\mathsf{cut}}$ over the same carried syntax packet, and its lambda coordinates are the accepted term packet L, the displayed beta-redex row r, the substitution ledger s, and the endpoint u.

For the first-order classifier clause, apply the rowwise classifier of Definition 197.2 to the pairs (D_1, D_1), (D_2, D_2), and $(D_{\mathsf{cut}}, D_{\mathsf{cut}})$. The symbol rows, parse-tree rows, formula endpoints, premise endpoints, conclusion endpoints, and deduction-step ledgers are the same displayed rows on the two sides. Hence the required transports are the componentwise reflexive hsame rows, with the same $\mathsf{Set}^\uparrow$ and $\mathsf{Tree}^\uparrow$ certificate provenance and the same finite Cont ledgers.

For the lambda classifier clause, use the accepted term packet L of Definition 200.5. The source redex is the same displayed row

$$\mathsf{lamApp}(\mathsf{lamAbs}(x, b), a),$$

and the substitution ledger is the same ledger s with output endpoint u. Apply Definition 200.13 with reflexive hsame transport on the carried $\mathsf{Tree}^\uparrow$ syntax row, the carried $\mathsf{Nat}^\uparrow$ binder-index row, the redex row, the substitution endpoint, and the Pkg provenance. The transported beta/substitution row therefore repacks as the original beta/substitution row.

The shared endpoint clauses of Definition 265.3 are filled by the reflexive hsame rows on the proof-program endpoint P, the cut-formula/type endpoint A, and the conclusion-type endpoint B. The provenance clause is the same finite row ρ over $\mathsf{NameCert}_{\mathsf{FirstOrder}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{LambdaCalc}^\uparrow}$.

Repacking these four families of rows gives exactly the CurryHoward endpoint classifier witness from $\mathcal{H}$ to $\mathcal{H}$. Since every component is projected from the accepted packet and returned through the displayed hsame, Cont, and provenance fields, the reflexive classifier row exports no host proof equality, host lambda equality, or quotient of alpha classes. □

Chapter 266

A Concrete Naming Certificate for $\mathsf{RiemannHilbert}^{\uparrow}$

The Riemann-Hilbert interface $\mathsf{RiemannHilbert}^{\uparrow}$ packages the equivalence between regular holonomic D-modules on a complex manifold and perverse sheaves of $\mathbb{C}$-vector spaces. It depends on $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$ (chapter 177) and $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ (chapter 151), and bridges differential and topological certificates through a common derived classifier.

266.1 Carrier definition

Definition 266.1 (Riemann-Hilbert BHist bridge packet)**.** *A* $\mathsf{RiemannHilbert}^{\uparrow}$ *carrier is a finite* BHist *bridge packet whose rows are the following data.*

1. *A derived-functor source row from* $\mathsf{NameCert}_{\mathsf{DerivedFunctor}^{\uparrow}}$*, including the displayed source object, derived endpoint, degree or cohomological index row, and exact-triangle provenance carried by the derived-functor ledger.*
2. *A sheaf target row from* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$*, read as a perverse-sheaf or constructible-sheaf target over the same displayed base histories, with its section, restriction, locality, and gluing rows visible as* BHist *data.*
3. *A regular-holonomic source classifier row saying that the differential source lies on the regular-holonomic branch of the derived-functor packet and can be compared only through the displayed classifier and* hsame *transport rows.*
4. *Transport rows matching the derived source, sheaf target, de Rham readback, local-system branch, and regular-holonomic classifier by* hsame*.*
5. Cont *provenance rows explaining how the derived source is read to the sheaf target, how local horizontal sections are read as local-system rows, and how gluing reads back through the* $\mathsf{Sheaf}^{\uparrow}$ *exactness ledger.*
6. Pkg *rows naming the* $\mathsf{DerivedFunctor}^{\uparrow}$ *and* $\mathsf{Sheaf}^{\uparrow}$ *dependency certificates, together with the public dependency boundary consumed by the bridge packet.*

The carrier records only this displayed BEDC packet. It does not contain an external derived category, analytic D-module model, sheaf quotient, monodromy equivalence, or host-level Riemann-Hilbert theorem.

Lean4 (definition) `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket`

266.2 The certificate

Theorem 266.2 (Riemann-Hilbert derived-sheaf source)**.** *Every accepted* RiemannHilbert$^{\uparrow}$ *packet exposes a derived-functor source row and a sheaf target row exactly as in Definition 266.1. The source side is the displayed* DerivedFunctor$^{\uparrow}$ *row with its exact-triangle and degree provenance, and the target side is the displayed* Sheaf$^{\uparrow}$ *row with its section, restriction, locality, and gluing ledger. No external derived category, analytic solution functor, or host sheaf quotient is projected by the source boundary.*

Lean4 ✓ `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket derived sheaf source`
Lean4 ✓ `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket source boundary`

Proof. Project the accepted packet to its first two carrier coordinates. The definition gives the DerivedFunctor$^{\uparrow}$ source row, including the derived endpoint and exact-triangle provenance, and the Sheaf$^{\uparrow}$ target row, including section, restriction, locality, and gluing data.

The remaining carrier coordinates are comparison, continuation, and package rows attached to those two projections. Since the carrier has no coordinate for an ambient derived category, analytic solution functor, or host sheaf quotient, the source boundary is exactly the displayed pair of BEDC rows. □

Lean4 ✓ `BEDC.Derived.RiemannHilbertUp.RiemannHilbertDerivedSheafSource composite boundary`

Theorem 266.3 (Riemann-Hilbert flat-connection classifier)**.** *The flat-connection branch of a* RiemannHilbert$^{\uparrow}$ *packet is classified only through the regular-holonomic source classifier, the local-system sheaf row, and the displayed* hsame *transports between the derived source, de Rham readback, and sheaf target. If two accepted packets agree on those rows, then they have the same visible flat-connection classifier surface.*

Lean4 ✓ `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket flat connection classifier`

Proof. Read the regular-holonomic classifier coordinate from Definition 266.1. It is the only row that marks the differential source branch and it is already tied to the derived-functor packet.

Next project the hsame rows matching the derived source, de Rham readback, local-system branch, and sheaf target. Agreement on these rows forces the same visible comparison surface after transport, because every local-system read is mediated by the displayed Cont provenance rather than by a host flat-connection object. □

Theorem 266.4 (Riemann-Hilbert local-system ledger)**.** *The local-system ledger exported by a* RiemannHilbert$^{\uparrow}$ *packet consists exactly of the sheaf target row, the local horizontal-section read rows, the* hsame *comparisons tying them to the derived source and de Rham readback, and the* Cont/Pkg *provenance naming the* DerivedFunctor$^{\uparrow}$ *and* Sheaf$^{\uparrow}$ *dependencies. It exports no monodromy representation, covering-space quotient, or external equivalence row.*

Proof. Project the local-system branch from the sheaf target and continuation coordinates of the carrier. The carrier definition lists the horizontal-section reads, their comparison rows, and the dependency packages that make the read visible to downstream consumers.

Constructor inspection of the displayed packet gives the exclusion claim. The rows generate local-system information only through Sheaf$^{\uparrow}$ sections, restrictions, locality, gluing, and Cont provenance; there is no constructor slot for a monodromy representation, covering-space quotient, or external equivalence. □

Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket local system ledger
Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertLocalSystemLedger continuation closure

Theorem 266.5 (Riemann-Hilbert regular-holonomic soundness)**.** *The regular-holonomic soundness boundary of* $\mathsf{RiemannHilbert}^{\uparrow}$ *is the conjunction of the derived-functor source row, the regular-holonomic source classifier row, the de Rham readback transport, and the sheaf target classifier row. A downstream consumer can read a regular-holonomic conclusion only by projecting these rows from the bridge packet.*

Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket regular holonomic soundness

Proof. Use Definition 266.2 to identify the source and target rows available to the packet. The source contributes the derived-functor endpoint and exact-triangle provenance, while the target contributes the sheaf classifier and gluing ledger.

Use Definition 266.3 for the classifier transport. The regular-holonomic branch is stable only along the displayed hsame and de Rham readback rows, so any consumer conclusion must be a projection from those rows together with the source and target packets. □

Theorem 266.6 (Riemann-Hilbert sheaf-gluing exactness)**.** *For a* $\mathsf{RiemannHilbert}^{\uparrow}$ *bridge packet, compatible local-system rows glue exactly through the* $\mathsf{Sheaf}^{\uparrow}$ *target ledger carried in Definition 266.1. The resulting global sheaf row remains tied to the same derived-functor source, regular-holonomic classifier,* hsame *transport rows,* Cont *provenance, and* Pkg *dependency rows.*

Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket sheaf gluing exactness

Proof. Project the $\mathsf{Sheaf}^{\uparrow}$ locality and gluing coordinates from the target row. Compatible local-system rows are sheaf rows in this packet, so gluing is performed by the displayed sheaf exactness ledger rather than by an external sheafification operation.

After gluing, project the comparison and provenance coordinates again. They still name the same derived-functor source, regular-holonomic classifier, hsame transports, continuation routes, and dependency packages, so the glued target remains inside the original $\mathsf{RiemannHilbert}^{\uparrow}$ carrier. □

Theorem 266.7 (Riemann-Hilbert de Rham readback scope)**.** *Every de Rham readback row in an accepted* $\mathsf{RiemannHilbert}^{\uparrow}$ *bridge packet is sourced only from the displayed* $\mathsf{DerivedFunctor}^{\uparrow}$ *source row, the regular-holonomic classifier row, the* $\mathsf{Sheaf}^{\uparrow}$ *target row, the local-system ledger, the displayed* hsame *transports, the* Cont *provenance, and the* Pkg *dependency rows. It exports no analytic solution functor, monodromy equivalence, host derived-category quotient, or external perverse-sheaf equivalence.*

Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket de rham readback scope
Lean4 ✓ BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket deRham readback scope
↪ variant BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket de Rham readback scope

Proof. Use Definition 266.2 to identify the dependency rows available to the packet: the $\mathsf{DerivedFunctor}^{\uparrow}$ source row and the $\mathsf{Sheaf}^{\uparrow}$ target row with their displayed provenance. Use Definition 266.3 for the regular-holonomic classifier and its de Rham transport row.

Next use Definition 266.4 to bind the local-system ledger to the same sheaf target, horizontal-section read rows, hsame comparisons, and Cont/Pkg provenance. Use Definition 266.6 to keep those rows stable under sheaf gluing.

Finally inspect the constructors of Definition 266.1. The carrier has slots only for the displayed source, classifier, target, transport, continuation, and package rows, so a de Rham readback row cannot export an analytic solution functor, monodromy equivalence, host derived-category quotient, or external perverse-sheaf equivalence. □

Theorem 266.8 (Riemann-Hilbert public certificate export)**.** *The bridge packet of Definition 266.1 exports exactly the derived-functor source row, the sheaf target row, the regular-holonomic classifier row, the local-system ledger, the de Rham readback scope, the displayed* hsame *transports, the* Cont *provenance, and the* Pkg *dependency rows listed in Definition 266.2, Definition 266.5, Definition 266.4, Definition 266.6, and Definition 266.7. These rows form the public* $\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$ *surface. No host derived-category quotient, analytic solution functor, monodromy equivalence, or external perverse-sheaf equivalence is exported.*

Lean4 ✓ `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket public certificate export`

Proof. Project the bridge packet to the derived-functor and sheaf coordinates. By Definition 266.2, these projections give the displayed source row and target row with their exact-triangle, section, restriction, locality, and gluing provenance.

Next project the classifier, local-system, and gluing rows. The regular-holonomic classifier is the one isolated by Definition 266.5; the local-system ledger is the one isolated by Definition 266.4; and the gluing stability is the row supplied by Definition 266.6.

Finally add the de Rham readback projection from Definition 266.7. Repacking these projections gives the public $\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$ surface, and constructor inspection of the same packet shows that no additional host derived-category quotient or analytic solution functor has a row to export. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannHilbert}^{\uparrow}$ packet is assembled from the displayed $\mathsf{DerivedFunctor}^{\uparrow}$ source row, $\mathsf{Sheaf}^{\uparrow}$ target row, regular-holonomic classifier, local-system ledger, de Rham readback, hsame transports, Cont routes, and Pkg provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 266.8 gives the public exported surface over the displayed $\mathsf{DerivedFunctor}^{\uparrow}$ source row, $\mathsf{Sheaf}^{\uparrow}$ target row, regular-holonomic classifier, local-system ledger, de Rham readback, hsame transport, Cont provenance, and Pkg dependency rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RiemannHilbertUp.RiemannHilbertBHistBridgePacket\_public\_certificate\_export`

Bridge status: none

Not claimed: This chapter does not close external derived categories, analytic $\mathcal{D}$-module models, monodromy equivalence, or a host-level Riemann-Hilbert theorem.

Upgrade path: Paper-axis maturity is complete for the displayed public certificate export; only machine bridge checking remains on the formal axis.

Chapter 267

A Concrete Naming Certificate for $\mathsf{KoszulDuality}^{\uparrow}$

The Koszul-duality interface $\mathsf{KoszulDuality}^{\uparrow}$ packages the contravariant derived equivalence between graded algebras of finite global dimension and their Koszul-dual algebras. It depends on $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ (chapter 160) and $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$ (chapter 87), and bridges algebraic and homological certificates through the same Ext-classifier.

267.1 Carrier definition

Definition 267.1 (KoszulDuality BHist Ext carrier)**.** *A* $\mathsf{KoszulDuality}^{\uparrow}$ *carrier is a finite* BHist *package consisting of four visible rows. The first row is a* $\mathsf{DerivedCat}^{\uparrow}$ *object row naming the source graded algebra object and its displayed derived-category endpoint. The second row is a* $\mathsf{TensorProduct}^{\uparrow}$ *graded tensor row naming the tensor surface used to form the dual algebra side. The third row is the displayed* Ext*-classifier row comparing the derived object row with the graded tensor row through their certificate dependencies. The fourth row is a variance ledger recording that the dual row reverses source and target order contravariantly.*

The carrier is this finite packet of rows and endpoint data. It contains no quotient derived category, no chosen representative, no external equivalence object, and no host-level model of Koszul duality outside the displayed BHist *ledger.*

Lean4 (definition) `BEDC.Derived.KoszulDualityUp.KoszulDualityBHistExtCarrier`

Theorem 267.2 (KoszulDuality Ext classifier stability)**.** *Let two* $\mathsf{KoszulDuality}^{\uparrow}$ *carrier packets have derived-category rows and graded tensor rows related by displayed* hsame *classifiers from* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{TensorProduct}^{\uparrow}}$*. If the* Ext*-classifier row in the first packet is read only through those two dependency certificates and the visible variance ledger, then the same* Ext*-classifier package repacks over the transported rows, and the contravariant dual row is accepted in the transported packet.*

Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityExtCarrier classifier stability`

Proof. Project the derived-category dependency certificate from the source packet. It transports the source object endpoint and all displayed derived rows through the listed hsame comparison.

Project the graded tensor dependency certificate in the same way. The tensor surface is transported as a certificate row, so the target-side endpoint used by the dual algebra surface is still available after transport.

Replay the Ext-classifier ledger with these two transported rows. The classifier has no independent carrier beyond the two projected dependency rows and the variance tag, so it repacks over the transported endpoints.

Finally read the contravariant tag. Since the tag records only reversal of source and target order, the transported dual row is accepted by the same finite ledger with the order reversed. □

Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityBHistExtCarrier ext classifier stability`

Theorem 267.3 (KoszulDuality derived tensor ledger exactness)**.** *For every accepted* $\mathsf{KoszulDuality}^{\uparrow}$ *carrier, the ledger exposes exactly the* $\mathsf{DerivedCat}^{\uparrow}$ *object row, the* $\mathsf{TensorProduct}^{\uparrow}$ *graded tensor row, the displayed* Ext*-classifier row, and the variance tag recording contravariant source-target reversal. No quotient category, chosen representative, external equivalence object, triangulated enhancement, or model-category localization is exported by the carrier.*

Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityDerivedTensorLedger exactness`

Proof. Inspect the carrier of Definition 267.1. Its accepted rows are the derived object row, graded tensor row, Ext-classifier row, and variance ledger.

The dependency certificates may be projected only through the rows named in the packet. The derived side contributes the object endpoint, and the tensor side contributes the graded tensor endpoint.

The Ext row is then a displayed classifier between those projected endpoints. Since no further constructor is present in the packet, none of the excluded host objects can be read from the accepted carrier. □

Theorem 267.4 (KoszulDuality NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{KoszulDuality}^{\uparrow}}$ *surface is the package consisting of the carrier of Definition 267.1, the displayed* Ext *classifier, the stability clause of Definition 267.2, the exact ledger of Definition 267.3, and the source coverage assertion that every accepted duality row is read from the displayed derived-category row, graded tensor row, classifier row, and variance tag.*

Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityBHistExtCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityExtCarrier semantic name certificate`

Proof. Use the carrier definition as the source predicate for the local certificate. It supplies precisely the finite BHist rows that the certificate is allowed to classify.

The classifier row is the displayed Ext row. Its transport obligation is supplied by Definition 267.2, which repacks the same classifier after the two dependency certificates transport their rows.

The ledger and source-coverage obligations are supplied by Definition 267.3. Together they say that every accepted row is visible in the finite package and that no host duality structure is added by the certificate boundary. □

Theorem 267.5 (KoszulDuality bridge boundary exclusions)**.** *The* $\mathsf{KoszulDuality}^{\uparrow}$ *boundary supplies only the BEDC* Ext*-classifier correspondence between the displayed* $\mathsf{DerivedCat}^{\uparrow}$ *and* $\mathsf{TensorProduct}^{\uparrow}$ *rows. It does not assert a full derived equivalence, a triangulated enhancement, a model-category localization, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.KoszulDualityUp.KoszulDualityBHistExtCarrier bridge boundary exclusions`

Proof. Read the public boundary through Definition 267.4. The available data are carrier, classifier, stability, ledger exactness, and source coverage.

Each of those rows factors through the finite Definition 267.1 packet. That packet contains the dependency rows and the variance tag, but it contains no object representing a host derived equivalence or any localization apparatus.

Therefore the boundary exports the stated BEDC classifier correspondence and nothing beyond the displayed rows. □

Theorem 267.6 (KoszulDuality standard derived-tensor bridge)**.** *Any standard bridge for the public* $\mathsf{KoszulDuality}^{\uparrow}$ *packet is restricted to the displayed* $\mathsf{DerivedCat}^{\uparrow}$ *source row, the* $\mathsf{TensorProduct}^{\uparrow}$ *graded tensor row, the* Ext*-classifier row, the variance ledger, the* hsame *transport, the* Cont *routes, the* Pkg *provenance, and the* NameCert *surface already isolated by the public certificate. It does not supply a full derived equivalence, a triangulated enhancement, a model-category localization, or a host quotient.*

Proof. Combine Definition 267.4 with Definition 267.5. The obligation surface lists the carrier, classifier, stability, ledger exactness, and source coverage rows, while the exclusions identify which host structures are absent from that public packet.

Repack the same finite BHist and Ext rows as the standard bridge boundary. The bridge has no independent coordinate beyond the displayed derived-category row, tensor row, classifier row, variance ledger, transports, routes, provenance, and naming surface.

Therefore a full derived equivalence, a triangulated enhancement, a model-category localization, and a host quotient remain outside the exported bridge boundary. □

Theorem 267.7 (KoszulDuality mature Ext-bridge boundary)**.** *Every finite consumer of the bridged* $\mathsf{KoszulDuality}^{\uparrow}$ *packet factors through the displayed* $\mathsf{DerivedCat}^{\uparrow}$ *source row,* $\mathsf{TensorProduct}^{\uparrow}$ *graded tensor row,* Ext*-classifier row, variance ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and* NameCert *surface isolated by Definition 267.6 and Definition 267.4.*

The boundary contains no full derived equivalence, triangulated enhancement, model-category localization, host quotient data, or machine-checked bridge evidence.

Proof. First apply Definition 267.4. It exposes the carrier, classifier, stability, ledger exactness, and source coverage rows as one finite $\mathsf{NameCert}_{\mathsf{KoszulDuality}^{\uparrow}}$ surface.

Next use Definition 267.5. The exclusion boundary removes full derived equivalence data, triangulated enhancement, model-category localization, and host quotients from the available packet rows.

Finally apply Definition 267.6. Any finite bridged consumer therefore reads only the displayed derived source, graded tensor, Ext classifier, variance, transport, route, provenance, and naming rows already present in the public packet. □

267.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{KoszulDuality}^{\uparrow}}$.

Constructive story: A $\mathsf{KoszulDuality}^{\uparrow}$ seed packet is generated from a displayed $\mathsf{DerivedCat}^{\uparrow}$ source row, a displayed $\mathsf{TensorProduct}^{\uparrow}$ graded tensor row, an Ext-classifier row tying those dependencies together, and a variance ledger recording contravariant source-target reversal. The carrier reads only this finite BHist packet; classifier stability, ledger exactness, source coverage, and the standard derived-tensor bridge boundary are paper-side obligations over the same rows, while quotient derived categories, host equivalences, triangulated enhancements, and model-category localizations are absent from the exported surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KoszulDuality}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridged semantic NameCert surface of Definition 267.4 and the standard derived-tensor bridge of Definition 267.6 bind the BHist/Ext carrier, classifier stability, derived tensor ledger exactness, source coverage, and bridge-boundary exclusions over the displayed DerivedCat$^{\uparrow}$ and TensorProduct$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KoszulDuality}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.KoszulDualityUp.KoszulDualityExtCarrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not claim a full derived equivalence, a triangulated or model-category presentation, host quotient data, or machine-checked bridge evidence for KoszulDuality$^{\uparrow}$.

Upgrade path: No further theory-axis upgrade is available for this chapter; bridgeChecked status remains a verification-axis upgrade.

Chapter 268

A Concrete Naming Certificate for $\mathsf{Langlands}^{\uparrow}$

The Langlands-correspondence interface $\mathsf{Langlands}^{\uparrow}$ packages the conjectural bijection between n-dimensional Galois representations of a global field and irreducible automorphic representations of GL_n matching local L-factors at almost all places. It depends on $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ (chapter 164) and $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ (chapter 104), and bridges Galois and automorphic certificates through a common L-factor classifier.

268.1 Carrier definition

Definition 268.1 (Langlands correspondence ledger carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *source row, a* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *source row, and a finite list of observed places at which both sources expose local factor answers. A* $\mathsf{Langlands}^{\uparrow}$ *correspondence carrier row is the* BHist *ledger row consisting of the two source certificates, the visible Galois and automorphic answer histories at the selected places, the local L-factor package rows read from those answers, and the* Cont *witnesses that place the paired local rows inside one finite observation packet.*

The carrier records only the observed packet. It does not assert an all-place correspondence, irreducibility, analytic continuation, functoriality, or uniqueness beyond the displayed finite BHist *ledger.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsBHistCorrespondenceCarrier source rows`

Definition 268.2 (Langlands local factor classifier)**.** *Two rows of Definition 268.1 are classified as the same* $\mathsf{Langlands}^{\uparrow}$ *packet exactly when their visible local L-factor packages are matched place by place, their source certificate rows are transported through the corresponding* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *classifiers, and their answer histories are related by the available* hsame, Cont, *and* Pkg *witnesses.*

Equivalently, the classifier is the least relation generated by source-certificate transport, answer-history transport, local package agreement, and finite concatenation of the observed place rows. No hidden global correspondence row is a generator of the classifier.

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsLFactorClassifier observed packet rows`

268.2 The certificate

Theorem 268.3 (Langlands source certificate scope)**.** *Every carried* $\mathsf{Langlands}^{\uparrow}$ *packet of Definition 268.1 has exactly two external source certificate dependencies: one* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *row and one* $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ *row. All remaining data in the packet are finite answer histories, local L-factor package rows, and the* hsame*,* Cont*, and* Pkg *witnesses that connect those rows.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsCorrespondenceLedger source certificate scope`

Proof. Unpack the carrier row. Its first two fields are the Galois and automorphic source certificates, and each observed place contributes only the two visible answer histories and the local package rows read from them.

The classifier of Definition 268.2 transports these fields through the two source NameCert classifiers and through the displayed history and package witnesses. Since no constructor introduces a third source certificate or a global host theorem, the stated scope is exact. □

Theorem 268.4 (Langlands local factor stability)**.** *If two carried* $\mathsf{Langlands}^{\uparrow}$ *packets are classified by Definition 268.2, then replacing either source representative by a* NameCert*-classified representative and replacing any visible local answer by an* hsame*-related answer preserves the classified local L-factor packet.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsLocalFactor stability`

Proof. First transport the source side. The $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Automorphic}^{\uparrow}}$ rows supply the certificate-level replacement witnesses, and the carrier keeps the transported rows inside the same finite observation packet by Cont.

Next transport the local factor rows. At each observed place, the classifier stores the package comparison as a Pkg row and the answer-history comparison as an hsame row. Finite concatenation of these rows gives a classifier witness for the whole packet after the replacement. □

Theorem 268.5 (Langlands correspondence obligation package)**.** *The* $\mathsf{Langlands}^{\uparrow}$ *chapter exports a finite correspondence obligation package: a carrier row, a local factor classifier row, the two source certificate rows, and the ledger exactness statement that the packet hides only the finite observed matching row. It does not claim a global Langlands correspondence theorem.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsCorrespondenceCarrier semantic name certificate`
Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsLFactorClassifier endpoint confluence`

Proof. Build the package from a carried row and its classifier witness. The carrier supplies the observed Galois answer rows, automorphic answer rows, local package rows, and Cont links. The classifier supplies the source transport and the visible place-by-place package comparisons.

Ledger exactness is obtained by inverting these two displayed structures. Every accepted datum is either one of the two source certificates, a finite answer history, a local factor package row, or a transport witness already named by hsame, Cont, Pkg, or NameCert. Hence the package contains no hidden all-place bijection or analytic assertion. □

Theorem 268.6 (Langlands finite local-factor public surface)**.** *The public surface exported by* $\mathsf{Langlands}^{\uparrow}$ *is exactly the finite local L-factor packet of Definition 268.5. A downstream consumer may read the two source certificates, the observed Galois and automorphic answer histories, the local L-factor package rows, and the displayed* hsame*,* Cont*, and* Pkg *witnesses connecting those rows. The same consumer cannot read an all-place bijection, irreducibility, analytic continuation, functoriality, or uniqueness from this surface.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsCorrespondenceLedger public endpoint boundary`
Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsCorrespondenceLedger finite local factor public surface`
Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsBHistCorrespondenceCarrier finite local factor public surface`
Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsBHistCorrespondenceCarrier public surface`

Proof. Start with Definition 268.3. It projects any accepted packet to the two named source certificates and to the finite answer and package rows carried at the observed places.

Next apply Definition 268.4. Source replacement and visible local-answer transport preserve the classified L-factor packet through the displayed hsame, Cont, and Pkg rows.

Finally use Definition 268.5. Its ledger exactness clause says that every readable row is already one of the listed certificate, history, local package, or witness rows, so the excluded global and analytic assertions have no carrier coordinate in the public surface. □

Theorem 268.7 (Langlands generated rows are unary)**.** *Every generated row of* LanglandsBHistCorrespondenceCarr *is a finite unary* BHist *row. The two source certificate rows, the visible Galois and automorphic answer rows, the local factor package rows, and the* Cont/Pkg *ledger rows are all read through the displayed finite observation packet of Definition 268.1.*

Lean4 ✓ `BEDC.Derived.LanglandsUp.LanglandsBHistCorrespondenceCarrier generated rows unary`

Proof. Unpack Definition 268.1. The carrier has two source certificate entries and then a finite list of observed-place entries. By Definition 268.3, those source entries are exactly the $\mathsf{NameCert}_{\mathsf{GaloisGroup}^\uparrow}$ and $\mathsf{NameCert}_{\mathsf{Automorphic}^\uparrow}$ rows, while each observed place contributes only the displayed answer histories and the local package rows read from them.

Apply Definition 268.5 to the same packet. The obligation package exposes no further generator: its remaining entries are the hsame, Cont, and Pkg rows already named by the finite ledger. Projecting each ledger entry therefore gives one unary BHist row, and finite ledger projection exhausts the generated carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{Langlands}^\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Langlands}^\uparrow}$, publicClosed)

Scope closed: The obligation scope is the finite local L-factor packet over $\mathsf{NameCert}_{\mathsf{GaloisGroup}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Automorphic}^\uparrow}$, visible BHist answer histories, hsame, Cont, Pkg, and the displayed source certificate rows.

Constructive story: A Langlands$^\uparrow$ packet is built by pairing one displayed Galois source certificate with one displayed automorphic source certificate, reading the visible local L-factor rows place by place, and recording only the finite history, continuation, package, and certificate-transport witnesses named in the carrier and classifier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Langlands}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.LanglandsUp.LanglandsBHistCorrespondenceCarrier\_finite\_local\_factor\_public\_surface`

Bridge status: none

Not claimed: This chapter does not claim a global Langlands correspondence, an all-place bijection, irreducibility, analytic continuation, functoriality, or uniqueness.

Upgrade path: Bridge closure requires a standard finite local-factor bridge for the exported public packet.

Chapter 269

A Concrete Naming Certificate for $\mathsf{NoetherSymmetry}^{\uparrow}$

The Noether-symmetry interface $\mathsf{NoetherSymmetry}^{\uparrow}$ packages the bijection between continuous symmetries of a Lagrangian and conserved currents along solutions of the Euler-Lagrange equations. It depends on $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ (chapter 141) and $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$ (chapter 194), and bridges algebraic and analytic certificates through the same conservation-law ledger.

269.1 Carrier definition

Definition 269.1 (Noether-symmetry conservation-ledger carrier)**.** *A* $\mathsf{NoetherSymmetry}^{\uparrow}$ *conservation-ledger carrier is a finite* BHist *row containing a* $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ *infinitesimal symmetry source, a* $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$ *Euler-Lagrange source, a displayed current history, and named* hsame, Cont, *and* Pkg *witnesses placing the current under both sources. The carrier also records the finite source-domain token that says the Lie-algebra row acts on the same displayed field history whose Euler-Lagrange equation is carried by the PDE row.*

The carrier is only this finite certificate surface. It does not assert a global Noether bijection, smooth variational calculus, an analytic action functional, or a standard model of conservation laws.

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryTasteGate single carrier alignment`

Definition 269.2 (Noether-symmetry current classifier)**.** *Two rows of Definition 269.1 are classified as the same* $\mathsf{NoetherSymmetry}^{\uparrow}$ *observation when their* $\mathsf{LieAlgebra}^{\uparrow}$ *infinitesimal-symmetry sources are transported by* $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$*, their* $\mathsf{Pde}^{\uparrow}$ *Euler-Lagrange sources are transported by* $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$*, and their displayed current histories and dependency ledgers are matched by* hsame, Cont, *and* Pkg.

The classifier is generated by source transport and finite current-ledger transport. It has no generator for equality of smooth fields, integration by parts in a host calculus, or a bijective correspondence between all symmetries and all conserved currents.

Lean4 (definition) `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryCurrentClassifier`

269.2 The certificate

Theorem 269.3 (Noether-symmetry Lie-PDE source boundary)**.** *Every row of Definition 269.1 exposes a finite* $\mathsf{LieAlgebra}^{\uparrow}$ *infinitesimal-symmetry source, a finite* $\mathsf{Pde}^{\uparrow}$ *Euler-Lagrange source, and the displayed current row attached to their shared field-history boundary by* Cont *and* Pkg *witnesses.*

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier source boundary`

Proof. Invert the carrier definition. The Lie-algebra source, the PDE source, the current history, and the source-domain token are explicit fields of the finite BHist row.

The only boundary comparisons available in the carrier are the named hsame, Cont, and Pkg witnesses. Therefore the shared field-history boundary is read from those finite rows, not from a smooth variational model or from global conservation-law semantics. □

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier lie pde source boundary`

Theorem 269.4 (Noether-symmetry conservation-ledger stability)**.** *If two* $\mathsf{NoetherSymmetry}^{\uparrow}$ *rows are classified by Definition 269.2, then transport of the* $\mathsf{LieAlgebra}^{\uparrow}$ *infinitesimal-symmetry source, transport of the* $\mathsf{Pde}^{\uparrow}$ *Euler-Lagrange source, and transport of the displayed current through* hsame*,* Cont*, and* Pkg *preserve the same conservation-ledger carrier.*

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier stability`

Proof. Transport the two source rows first. The $\mathsf{NameCert}_{\mathsf{LieAlgebra}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Pde}^{\uparrow}}$ classifiers supply transported source histories while keeping the displayed field-history boundary visible in the finite ledger.

Then transport the current row and its dependency witnesses through hsame, Cont, and Pkg. These are exactly the generators listed in Definition 269.2, so the transported packet is again a row of Definition 269.1. □

Theorem 269.5 (Noether-symmetry NameCert obligation surface)**.** *The* $\mathsf{NoetherSymmetry}^{\uparrow}$ *chapter exports a NameCert obligation surface consisting of the conservation-ledger carrier of Definition 269.1, the current classifier of Definition 269.2, the finite* $\mathsf{LieAlgebra}^{\uparrow}$ *infinitesimal-symmetry source, the finite* $\mathsf{Pde}^{\uparrow}$ *Euler-Lagrange source, the displayed conservation-current row, and the* BHist*,* hsame*,* Cont*,* Pkg*, and NameCert rows that connect those fields.*

The surface does not claim a global Noether bijection, smooth variational calculus, equality in a standard analytical model, or completeness of all conserved currents.

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier namecert obligation surface`

Proof. Assemble the surface from one carried row and one classifier witness. The carried row supplies the Lie-algebra source, PDE source, current history, source-domain token, and dependency ledger; the classifier supplies only source transport and finite current-ledger transport.

Conversely, every exported row appears in one of those fields or in the transport witnesses named by the classifier. Thus a downstream chapter may consume the finite BHist and NameCert obligation surface, but it cannot read off a global Noether correspondence or an analytic variational calculus from this certificate alone. □

Theorem 269.6 (Noether-symmetry finite-current public export)**.** *The accepted finite conservation-current package for* $\mathsf{NoetherSymmetry}^{\uparrow}$ *exports exactly the* BHist *conservation-ledger carrier of Definition 269.1, the current classifier of Definition 269.2, the Lie-PDE source boundary of Definition 269.3, the stability row of Definition 269.4, and the* NameCert *obligation surface of Definition 269.5.*

The exported public row exposes only Cont*,* hsame*,* Pkg*,* NameCert*, the finite Lie source, the finite PDE field row, and the conservation-current ledger. It does not expose variational calculus, analytic PDE regularity, smooth manifold data, or a standard Noether theorem.*

Lean4 ✓ `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier public export`

Proof. Project the five displayed rows. The carrier projection gives the finite BHist conservation ledger, the classifier projection gives the current comparison rule, and the source-boundary projection gives the finite Lie source together with the finite PDE field row.

Next use the stability theorem to keep the projected carrier fixed under the allowed hsame, Cont, and Pkg transports. Use the obligation surface theorem to retain exactly the NameCert rows and dependency ledgers already named by the package.

Repack these projections as the public consumer boundary. Since every component of the repacked boundary is one of the five projected rows, and since no projected row contains variational calculus, analytic PDE regularity, smooth manifold data, or a standard Noether theorem, the public export has precisely the stated finite-current surface. □

Theorem 269.7 (Noether-symmetry mature finite-current boundary)**.** *The mature package boundary for* $\mathsf{NoetherSymmetry}^{\uparrow}$ *is exhausted by the conservation-ledger carrier, current classifier, Lie-PDE source boundary, stability row, NameCert obligation surface, and finite-current public export. Every consumer reads only the finite Lie source, finite PDE field row, conservation-current ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already displayed by that surface.*

Proof. Project the conservation-ledger carrier and current classifier. These rows provide the finite current packet and the finite comparison rule used by every accepted $\mathsf{NoetherSymmetry}^{\uparrow}$ read.

Apply the Lie-PDE source boundary and the stability row. They keep the Lie source, PDE field row, conservation-current row, hsame transport, Cont route, and Pkg provenance inside the same finite package boundary.

Use the NameCert obligation surface and the finite-current public export to repack those projections as the consumer-facing package. The public export already lists the exact rows available to consumers, so the mature boundary adds no independent coordinate.

The boundary explicitly excludes a global Noether correspondence, smooth variational calculus, analytic PDE regularity, smooth manifold data, and a standard Noether theorem. Those structures are not read from the finite current package. □

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherSymmetry}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherSymmetry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 269.6 is the public boundary over the finite Lie source, PDE field row, conservation-current ledger, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherSymmetry}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.NoetherSymmetryUp.NoetherSymmetryConservationLedgerCarrier\_public\_export`

Bridge status: none

Constructive story: A finite conservation-ledger packet is assembled from the displayed Lie-algebra row, PDE row, current classifier row, transport witnesses, and one provenance packet; accepted consumers read only those finite current rows.

Not claimed: This chapter does not claim a full Noether theorem, variational calculus, analytic PDE regularity, smooth manifold data, or standard bridge.

Upgrade path: The paper axis is mature; further progress is Lean and bridge verification for the same finite-current package boundary.

Chapter 270

A Concrete Naming Certificate for $\mathsf{GeomQuantization}^{\uparrow}$

The geometric-quantization interface $\mathsf{GeomQuantization}^{\uparrow}$ packages the construction of a Hilbert space from a symplectic manifold via a prequantum line bundle, polarisation, and metaplectic correction. It depends on $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ (chapter 138) and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91), and bridges geometric and operator-algebraic certificates through a common polarisation classifier.

270.1 Carrier definition

Definition 270.1 (GeomQuantization BHist source packet)**.** *A* $\mathsf{GeomQuantization}^{\uparrow}$ *source packet is a finite* BHist *package*

$$\mathcal{G} = (\mathcal{S}, \mathcal{H}, L, \Pi, \mu, \rho, \lambda, \tau)$$

where $\mathcal{S}$ *is a dependency row certified by* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$*,* $\mathcal{H}$ *is a dependency row certified by* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$*,* $L : \mathsf{BHist}$ *is the prequantum line-bundle row over the symplectic source,* $\Pi : \mathsf{BHist}$ *is the polarisation classifier row,* $\mu : \mathsf{BHist}$ *is the metaplectic-correction ledger row,* ρ *records the* Pkg *provenance of the symplectic and Hilbert dependency packets,* λ *is the finite* Cont *readback from* $(\mathcal{S}, L, \Pi, \mu)$ *to the Hilbert carrier row, and* τ *is the family of* hsame *transports preserving the source, line-bundle, polarisation, metaplectic, and Hilbert endpoint rows.*

The packet is accepted only when all listed rows project from one displayed Pkg *provenance package. Thus* $\mathsf{GeomQuantization}^{\uparrow}$ *carries a finite history surface from a certified symplectic source to a certified Hilbert target; it does not export an ambient quantization functor, a host Hilbert construction, or uncarried symplectic geometry.*

Lean4 (definition) `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket`

Theorem 270.2 (GeomQuantization source dependency surface)**.** *For every packet of Definition 270.1, the* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *projection and the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *projection are rows of the same* BHist *package. The prequantum line-bundle row, polarisation classifier row, metaplectic ledger row, finite* Cont *readback,* hsame *transports, and* Pkg *provenance are therefore shared dependency data rather than separate certificates.*

Proof. Project the tuple $\mathcal{G}$. Its first two components are the symplectic and Hilbert dependency rows, and its provenance component ρ records both rows inside one package.

The remaining components are read from that package: L gives the prequantum line-bundle row, Π gives the polarisation classifier, μ gives the metaplectic ledger, λ gives the finite continuation to the Hilbert endpoint, and τ gives the history transports.

Repacking these projections returns the same source packet. Since the construction never leaves the displayed BHist package, the two dependency projections share one BEDC carrier surface. □

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket source dependency surface`

Theorem 270.3 (GeomQuantization polarisation classifier stability)**.** *If two packets of Definition 270.1 are related by the carried* hsame *transports on their* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source rows and by the same* Pkg *provenance, then transport of the polarisation classifier preserves the prequantum line-bundle row and the polarisation row, and the finite* Cont *readback lands in the transported* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *row.*

Proof. Begin with the transported symplectic source rows. The packet lists τ as the transport family, so the source comparison is a BEDC history comparison rather than an equality of external symplectic manifolds.

Apply the same transport family to the line-bundle row L and the polarisation row Π. Because both rows are components of the same package, their transported versions remain attached to the transported source.

Finally read the ledger λ. It is the finite Cont path from the transported source, line-bundle, polarisation, and metaplectic rows to the Hilbert endpoint. Repacking that endpoint through the $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ dependency row proves classifier stability. □

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket polarisation classifier stability`

Theorem 270.4 (GeomQuantization ledger exactness)**.** *The* $\mathsf{GeomQuantization}^{\uparrow}$ *ledger is exact for exactly the rows listed in Definition 270.1: the* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *endpoint row, the prequantum line-bundle row, the polarisation classifier row, the metaplectic-correction ledger row, the finite* Cont *readback, the* hsame *transports, and the* Pkg *provenance. Coverage, soundness, and completeness range only over those rows and exclude host Hilbert-space construction and uncarried symplectic geometry.*

Proof. Coverage is projection from an accepted packet. Each accepted packet displays the two dependency rows, the line-bundle row, the polarisation row, the metaplectic ledger, the continuation readback, the transport family, and the provenance row.

For soundness, take exactly such displayed rows with the dependency certificates attached. Repack them as the tuple $\mathcal{G}$, and the carrier definition accepts the resulting package.

For completeness, suppose a public $\mathsf{GeomQuantization}^{\uparrow}$ row is accepted. The carrier definition has no projection for a host Hilbert construction and no projection for uncarried symplectic data. Hence every public row is generated by the listed source, classifier, ledger, transport, continuation, and provenance rows. □

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket ledger exactness`
Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket line ledger transport determinacy`

Theorem 270.5 (GeomQuantization NameCert obligation surface)**.** *The paper-side* $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ *obligation surface consists of the carrier packet of Definition 270.1, the shared dependency surface of Definition 270.2, the polarisation classifier stability row of Definition 270.3, and the ledger exactness row of Definition 270.4. Downstream consumers may project the symplectic dependency, Hilbert dependency, prequantum, polarisation, metaplectic,* Cont*,* hsame*, and* Pkg *rows, but no stronger geometric-quantization bridge is part of this surface.*

Proof. Assemble the carrier first. The definition supplies a finite BHist packet with the two dependency certificates, the prequantum row, the polarisation row, the metaplectic row, continuation readback, transport data, and provenance.

Next use Definition 270.2 to bind the $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ projections to the same package. Use Definition 270.3 to keep the classifier stable under the carried hsame transports.

Finally apply Definition 270.4. Its coverage, soundness, and completeness clauses say that the public certificate is exhausted by the displayed rows. Repacking these rows gives the $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ obligation surface and rules out any unlisted bridge data. □

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket namecert obligation surface`

Theorem 270.6 (GeomQuantization shared-provenance readback exactness)**.** *Any public readback of a* $\mathsf{GeomQuantization}^{\uparrow}$ *packet that consumes both the* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source row and the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *endpoint row factors through the same displayed* Pkg *provenance row, finite* Cont *readback, and carried* hsame *transport family of Definition 270.1. The readback may also project the prequantum line-bundle row, polarisation classifier row, and metaplectic-correction ledger row named by that packet, but it cannot read an ambient quantization functor or host Hilbert construction.*

Proof. Start with the accepted source packet. By Definition 270.2, the symplectic source row and Hilbert endpoint row are projections of one finite BHist package, and their provenance is the single displayed Pkg row ρ.

Next transport the classifier data. The polarisation classifier stability theorem keeps the prequantum line-bundle and polarisation rows attached to the same carried hsame transport family, and it sends the finite Cont ledger to the transported $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ endpoint row.

Finally apply Definition 270.4 together with Definition 270.5. These results exhaust the public certificate by the displayed source, endpoint, classifier, ledger, continuation, transport, and provenance rows. Since no accepted packet field names a host quantization functor or an ambient Hilbert construction, the public readback has no route to either structure. □

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket shared provenance readback exactness`

Theorem 270.7 (GeomQuantization public certificate handoff)**.** *A public* $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ *consumer may read exactly the shared* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *endpoint row, the prequantum line-bundle row, the polarisation classifier row, the metaplectic ledger row, the finite* Cont *readback, the* hsame *transport family, and the* Pkg *provenance supplied by Definition 270.1 and exhausted by Definition 270.5. The handoff exports no host quantization functor, no ambient Hilbert construction, and no uncarried symplectic geometry.*

Lean4 ✓ `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket public certificate handoff`

Proof. Project the accepted source packet. Its displayed coordinates supply the symplectic dependency row, Hilbert endpoint row, prequantum line-bundle row, polarisation classifier, metaplectic ledger, finite continuation readback, transport family, and package provenance.

Apply Definition 270.2 to bind the source and endpoint rows to one provenance package. Then use Definition 270.3 for the classifier transport and Definition 270.4 for the exact public ledger rows.

Finally repack exactly those projected fields as the consumer handoff. The obligation surface of Definition 270.5 exhausts the public projections, so there is no coordinate from which to read a host quantization functor, an ambient Hilbert construction, or uncarried symplectic geometry. □

270.2 The certificate

Closure status for $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$.

Constructive story: A geometric-quantization packet is assembled from one certified $\mathsf{Symplectic}^{\uparrow}$ source row and one certified $\mathsf{Hilbert}^{\uparrow}$ endpoint row, together with the displayed prequantum line-bundle row, polarisation classifier, metaplectic ledger, finite continuation readback, history-sameness transport, and package provenance. The source packet accepts only those rows, so a public consumer may project the shared dependencies and the carried classifier and ledger faces, while ambient quantization functors, host Hilbert construction, uncarried symplectic geometry, and standard bridges remain outside the displayed carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 270.7 gives the public export surface over the scoped $\mathsf{Symplectic}^{\uparrow}$ dependency row, $\mathsf{Hilbert}^{\uparrow}$ endpoint row, prequantum line-bundle row, polarisation classifier, metaplectic ledger, finite Cont readback, hsame transports, and Pkg provenance carried by the source packet, and Definition 270.9 gives the mature public-surface standard presentation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.GeomQuantizationUp.GeomQuantizationBHistSourcePacket\_public\_certificate\_handoff`

Bridge status: none

Not claimed: This chapter does not close ambient quantization functors, host Hilbert construction, uncarried symplectic geometry, or standard bridges.

Upgrade path: No further paper-axis theory closure remains for the public-surface packet; Lean bridge checking remains a separate verification-axis task.

Definition 270.8 (GeomQuantization public-surface standard presentation)**.** *A standard geometric-quantization presentation is public-surface over a* $\mathsf{GeomQuantization}^{\uparrow}$ *source packet* $\mathcal{G}$ *when its admitted BEDC readback is a tuple*

$$\mathcal{R} = (S, H, L, \Pi, \mu, \lambda, \tau, \rho)$$

whose entries are, respectively, the $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *endpoint row, the prequantum line-bundle row, the polarisation classifier row, the metaplectic-correction ledger row, the finite* Cont *readback, the carried* hsame *transport family, and the* Pkg *provenance row projected from* $\mathcal{G}$ *in the sense of Definition 270.1.*

The readback is conservative when every consumer-visible row factors through those eight projections and their public certificate handoff in Definition 270.7. In particular, the presentation may name the ordinary geometric-quantization words attached to S*,* H*,* L*,* Π*, and* μ*, but it has no separate coordinate for an ambient quantization functor, no separate coordinate for a host Hilbert construction, and no separate coordinate for uncarried symplectic geometry.*

Theorem 270.9 (GeomQuantization public-surface standard bridge)**.** *Every public-surface standard presentation over a* $\mathsf{GeomQuantization}^{\uparrow}$ *source packet conservatively bridges to the public* $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ *certificate. More explicitly, if a standard geometric-quantization presentation is read only through the* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *endpoint row,*

the prequantum line-bundle row, the polarisation classifier row, the metaplectic-correction ledger row, the finite Cont *readback, the carried* hsame *transport family, and the* Pkg *provenance row of Definition 270.8, then its BEDC export is exactly the public handoff of Definition 270.7. Consequently the bridge cannot expose an ambient quantization functor, a host Hilbert construction, or uncarried symplectic geometry.*

Proof. Let $\mathcal{R} = (S, H, L, \Pi, \mu, \lambda, \tau, \rho)$ be the admitted readback of a public-surface standard presentation over a source packet $\mathcal{G}$. By the definition of the public-surface presentation, each coordinate of $\mathcal{R}$ is a projection from the same packet named in Definition 270.1.

The source and endpoint rows are not independent standard data. By Definition 270.2, S and H are rows of the same finite BHist package, with the same displayed Pkg provenance row ρ. The line-bundle, polarisation, metaplectic, continuation, and transport entries are therefore shared dependency data rather than a second certificate.

The classifier part is stable under the carried history transport. By Definition 270.3, transporting the polarisation classifier keeps the line-bundle and polarisation rows attached to the transported source and sends the finite Cont readback to the transported $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ endpoint row. Thus the standard presentation has no additional route from the symplectic row to the Hilbert row beyond the displayed public transport and continuation rows.

It remains to identify the public export. By Definition 270.4, the ledger of $\mathsf{GeomQuantization}^{\uparrow}$ is exact for precisely the listed rows. By Definition 270.5, those rows are the local $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ obligation surface. Finally, Definition 270.7 says that a public consumer may read exactly those eight rows and nothing stronger.

Repacking $\mathcal{R}$ through that handoff gives the public $\mathsf{NameCert}_{\mathsf{GeomQuantization}^{\uparrow}}$ certificate. Since the public-surface presentation has no coordinate outside S, H, L, Π, μ, λ, τ, and ρ, there is no BEDC projection from which to read an ambient quantization functor, a host Hilbert construction, or uncarried symplectic geometry. This is exactly the claimed conservative bridge. □

Chapter 271

A Concrete Naming Certificate for $\mathsf{DeformQuantization}^{\uparrow}$

The deformation-quantization interface $\mathsf{DeformQuantization}^{\uparrow}$ packages the formal star product $\star_\hbar$ on a Poisson manifold whose first-order term is the Poisson bracket (Kontsevich formality). It depends on $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ (chapter 138) and $\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$ (chapter 180), and bridges Poisson and noncommutative certificates through the same formal-deformation classifier.

271.1 Carrier definition

271.2 The certificate

Definition 271.1 (DeformQuantization first-order star-product ledger)**.** *Fix a* $\mathsf{Symplectic}^{\uparrow}$ *obligation boundary* Σ *as in Definition 138.2. Fix a* $\mathsf{HopfAlg}^{\uparrow}$ *row* $H = (a, t, s, u, e)$ *whose carrier and classifier are read through Definition 180.1 and Definition 180.2. A first-order star-product ledger for* $\mathsf{DeformQuantization}^{\uparrow}$ *over* (Σ, H) *is a displayed* BHist *row*

$$\mathsf{DQStarLedger}_{\Sigma,H}(\mu_\hbar, \mu_0, \mu_1, \mu_1^{\mathrm{op}}, \kappa, \pi, \Lambda_\star)$$

with the following fields.

1. *The row* $\mu_\hbar$ *is a formal-product row whose multiplication provenance is read through the* $\mathsf{HopfAlg}^{\uparrow}$ *multiplication and tensor rows of* H*.*
2. *The row* μ_0 *is the degree-zero product endpoint of* $\mu_\hbar$ *and is classified with the ordinary commutative product row on the function algebra over the carrier exposed by* Σ*.*
3. *The rows* μ_1 *and* μ_1^{op} *are the degree-one coefficient endpoints of* $\mu_\hbar$ *and of the opposite formal product.*
4. *The row* κ *is the degree-one commutator endpoint, displayed by the continuation row*

$$\mathsf{Cont}(\mu_1, \mu_1^{\mathrm{op}}, \kappa).$$

5. *The row* π *is the Poisson-bracket obligation row attached by the* $\mathsf{DeformQuantization}^{\uparrow}$ *packet to the same symplectic boundary* Σ*.*

6. *The formal-deformation classifier attached to $\Lambda_\star$ contains the bracket exactness comparison*

$$\mathsf{hsame}(\kappa, \pi),$$

and its coverage and soundness clauses say that every bracket read of the formal-deformation classifier factors through π and every accepted degree-one commutator endpoint factors through κ before this comparison is applied.

Theorem 271.2 (DeformQuantization first-order bracket ledger)**.** *Let Σ be a* $\mathsf{Symplectic}^\uparrow$ *obligation boundary and let $H = (a, t, s, u, e)$ be a* $\mathsf{HopfAlg}^\uparrow$ *formal-product row as in Definition 271.1. Let Q be a* $\mathsf{DeformQuantization}^\uparrow$ *packet over (Σ, H) containing the visible ledger row*

$$\mathsf{DQStarLedger}_{\Sigma,H}(\mu_\hbar, \mu_0, \mu_1, \mu_1^{\mathrm{op}}, \kappa, \pi, \Lambda_\star).$$

If μ_0 is read as the degree-zero product endpoint of the star product and κ is read as the degree-one commutator endpoint, then the formal-deformation classifier of Q reads the same degree-one endpoint as the displayed Poisson-bracket obligation row:

$$\mathsf{PoissonBracketOb}_{\Sigma,H}(\pi) \wedge \mathsf{hsame}(\kappa, \pi).$$

Thus the bracket endpoint belongs to the $\mathsf{DeformQuantization}^\uparrow$ *ledger over the supplied* $\mathsf{Symplectic}^\uparrow$ *boundary and* $\mathsf{HopfAlg}^\uparrow$ *formal-product row, not to the* $\mathsf{Symplectic}^\uparrow$ *boundary alone.*

Lean4 ✓ `BEDC.Derived.DeformQuantizationUp.DeformQuantizationStarLedger first order bracket`

Proof. Unpack Definition 271.1. The $\mathsf{HopfAlg}^\uparrow$ side supplies the formal product substrate through the multiplication and tensor provenance rows recorded in Definition 180.5. This places $\mu_\hbar$ inside the same ledger spine as the degree-zero endpoint μ_0 and the two degree-one coefficient endpoints μ_1 and μ_1^{op}.

The commutator endpoint is not an additional classifier choice. It is the row κ displayed by the continuation face

$$\mathsf{Cont}(\mu_1, \mu_1^{\mathrm{op}}, \kappa)$$

inside the same star-product ledger. The degree-zero clause records that the zeroth coefficient is the ordinary product endpoint, while the degree-one clause records that the first antisymmetric endpoint is κ.

The Poisson-bracket clause of the ledger then supplies the row π and the exactness comparison

$$\mathsf{hsame}(\kappa, \pi).$$

By the coverage clause, any bracket read made by the formal-deformation classifier factors through this displayed π row. By the soundness clause, the accepted degree-one commutator read factors through κ before it is compared with π. Therefore the endpoint read from the degree-one commutator is classified with the displayed Poisson-bracket obligation row, giving

$$\mathsf{PoissonBracketOb}_{\Sigma,H}(\pi) \wedge \mathsf{hsame}(\kappa, \pi).$$

The separation from the symplectic dependency is exactly the consumer frontier of Definition 138.11: the $\mathsf{Symplectic}^\uparrow$ boundary supplies carrier, two-form, closedness, nondegeneracy, pairing, and transport rows, but it does not by itself export a Poisson bracket. The bracket endpoint used here is supplied only after the $\mathsf{DeformQuantization}^\uparrow$ packet adds the first-order star-product ledger over the Hopf formal-product row. □

Theorem 271.3 (DeformQuantization star-ledger carrier obligation)**.** *Let Σ be a* $\mathsf{Symplectic}^{\uparrow}$ *obligation boundary and let $H = (a, t, s, u, e)$ be a* $\mathsf{HopfAlg}^{\uparrow}$ *row satisfying Definition 180.1 and Definition 180.2. A* $\mathsf{DeformQuantization}^{\uparrow}$ *packet Q over (Σ, H) has its first-order carrier exactly when it exposes a displayed row*

$$\mathsf{DQStarLedger}_{\Sigma,H}(\mu_\hbar, \mu_0, \mu_1, \mu_1^{\mathrm{op}}, \kappa, \pi, \Lambda_\star)$$

of Definition 271.1 whose formal product provenance is the Pkg *row read from the multiplication and tensor faces of H, whose commutator face is the continuation row* $\mathsf{Cont}(\mu_1, \mu_1^{\mathrm{op}}, \kappa)$*, and whose carrier comparisons are the displayed* hsame *rows of the packet. Thus no separate carrier for Q is admitted beyond the* BHist *ledger row, the* Cont *face, the* hsame *comparisons, and the* Pkg *provenance supplied by Σ and H.*

Proof. Project the $\mathsf{HopfAlg}^{\uparrow}$ side first. By Definition 180.5, the row H supplies the carrier, classifier, tensor-row stability, multiplication and comultiplication provenance through Cont, and the antipode ledger rows. The formal-product provenance of $\mu_\hbar$ is therefore the same Pkg row read from those multiplication and tensor faces.

Next project the $\mathsf{Symplectic}^{\uparrow}$ boundary. The consumer frontier of Definition 138.11 supplies the carrier boundary and transport surface needed by a Poisson consumer, while withholding any independent bracket export. Hence the deformation packet must attach its bracket-facing carrier through the displayed star ledger rather than through a hidden symplectic row.

Finally repack the projections into Definition 271.1. The row $\mu_\hbar$ carries the formal product, $\mu_0, \mu_1, \mu_1^{\mathrm{op}}$ are its coefficient endpoints, κ is displayed by $\mathsf{Cont}(\mu_1, \mu_1^{\mathrm{op}}, \kappa)$, and the packet comparisons are the recorded hsame rows. These fields are exactly the advertised BHist carrier for Q, so any accepted first-order carrier read is covered by that row and by no additional carrier datum. □

Lean4 ✓ `BEDC.Derived.DeformQuantizationUp.DeformQuantizationStarLedger carrier obligation`

Theorem 271.4 (DeformQuantization bracket classifier obligation)**.** *Let Q and Q' be* $\mathsf{DeformQuantization}^{\uparrow}$ *packets over transported rows (Σ, H) and (Σ', H'), with displayed star-product ledger rows*

$$\mathsf{DQStarLedger}_{\Sigma,H}(\mu_\hbar, \mu_0, \mu_1, \mu_1^{\mathrm{op}}, \kappa, \pi, \Lambda_\star)$$

and

$$\mathsf{DQStarLedger}_{\Sigma',H'}(\mu'_\hbar, \mu'_0, \mu'_1, (\mu'_1)^{\mathrm{op}}, \kappa', \pi', \Lambda'_\star).$$

If the $\mathsf{Symplectic}^{\uparrow}$ *and* $\mathsf{HopfAlg}^{\uparrow}$ *carrier-classifier rows transport Σ to Σ' and H to H', then the* $\mathsf{DeformQuantization}^{\uparrow}$ *classifier transports the degree-zero product endpoint, the two first-order coefficient endpoints, the commutator endpoint, and the Poisson-bracket comparison row:*

$$\mathsf{hsame}(\mu_0, \mu'_0) \wedge \mathsf{hsame}(\mu_1, \mu'_1) \wedge \mathsf{hsame}(\mu_1^{\mathrm{op}}, (\mu'_1)^{\mathrm{op}}) \wedge \mathsf{hsame}(\kappa, \kappa') \wedge \mathsf{hsame}(\pi, \pi').$$

Lean4 ✓ `BEDC.Derived.DeformQuantizationUp.DeformQuantizationStarLedger bracket classifier obligation`

Proof. Begin with the transported $\mathsf{HopfAlg}^{\uparrow}$ row. The classifier surface in Definition 180.5 carries the multiplication and tensor provenance rows through the transport from H to H', so the formal-product row $\mu_\hbar$ and the transported row $\mu'_\hbar$ have the same Pkg provenance for the product spine.

The endpoint clauses are then read directly from the two displayed star ledgers. The degree-zero endpoint is transported as the ordinary product endpoint over the transported symplectic

carrier. The two first-order coefficient endpoints are transported as coefficient faces of the same formal-product spine.

It remains to transport the bracket-facing rows. The continuation faces

$$\mathsf{Cont}(\mu_1, \mu_1^{\mathrm{op}}, \kappa)$$

and

$$\mathsf{Cont}(\mu_1', (\mu_1')^{\mathrm{op}}, \kappa')$$

identify the commutator endpoints after the two coefficient endpoints have been transported. The consumer frontier of Definition 138.11 supplies the transported Poisson consumer boundary, and the ledger exactness clause records the corresponding $\mathsf{hsame}(\pi, \pi')$ comparison. Repacking these five transported rows gives the displayed classifier obligation. □

Theorem 271.5 (DeformQuantization first-order ledger exactness obligation)**.** *For every* $\mathsf{DeformQuantization}^{\uparrow}$ *packet* Q *over* (Σ, H) *with displayed ledger*

$$\mathsf{DQStarLedger}_{\Sigma,H}(\mu_\hbar, \mu_0, \mu_1, \mu_1^{\mathrm{op}}, \kappa, \pi, \Lambda_\star),$$

every accepted first-order bracket readback of Q *is covered by the ledger row of Definition 271.1 and is sound for the checked bracket statement of Definition 271.2. Equivalently, an accepted readback factors through the degree-one commutator endpoint* κ*, the Poisson-bracket obligation row* π*, and the exactness comparison* $\mathsf{hsame}(\kappa, \pi)$ *before it is reported as a* $\mathsf{DeformQuantization}^{\uparrow}$ *bracket endpoint.*

Lean4 ✓ `BEDC.Derived.DeformQuantizationUp.DeformQuantizationStarLedger first order ledger exactness`

Proof. Coverage is the first projection. Unpack Definition 271.1: the formal-deformation classifier attached to $\Lambda_\star$ records that every bracket read factors through the displayed row π, and every accepted degree-one commutator endpoint factors through κ.

Soundness is the second projection. The checked ledger theorem Definition 271.2 identifies the endpoint read from the degree-one commutator with the displayed Poisson-bracket obligation row by the comparison

$$\mathsf{PoissonBracketOb}_{\Sigma,H}(\pi) \wedge \mathsf{hsame}(\kappa, \pi).$$

The Hopf provenance and the symplectic frontier rule out the two possible escapes. By Definition 180.5, the formal product and its coefficient endpoints are read through the Pkg and Cont faces supplied by H. By Definition 138.11, Σ supplies the consumer boundary but does not export an independent Poisson bracket. Therefore every accepted first-order readback is both covered by the displayed ledger row and sound for the checked bracket exactness statement. □

Definition 271.6 (DeformQuantization formal-deformation obligation inventory)**.** *For a* $\mathsf{DeformQuantization}^{\uparrow}$ *packet* Q *over a* $\mathsf{Symplectic}^{\uparrow}$ *boundary* Σ *and a* $\mathsf{HopfAlg}^{\uparrow}$ *row* H*, the formal-deformation obligation inventory is the finite list of public rows consisting of the carrier row of Definition 271.3, the bracket classifier row of Definition 271.4, the* Cont *and* hsame *stability rows for the coefficient and commutator endpoints, the first-order ledger row of Definition 271.1, the exactness row of Definition 271.5, and the public certificate rows that expose these data through* Pkg *provenance. The inventory permits reads from* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{HopfAlg}^{\uparrow}}$*,* Cont*,* hsame*,* BHist*, and* Pkg *only.*

Theorem 271.7 (DeformQuantization scoped primitive inventory)**.** *Every accepted first-order obligation row of the* $\mathsf{DeformQuantization}^{\uparrow}$ *inventory of Definition 271.6 factors through exactly the displayed* $\mathsf{Symplectic}^{\uparrow}$ *boundary* Σ*, the* $\mathsf{HopfAlg}^{\uparrow}$ *formal-product row* H*, the coefficient and commutator faces carried by* Cont*, the endpoint comparisons carried by* hsame*, the first-order* BHist *ledger row, and the* Pkg *provenance of that inventory. Conversely, those rows repack into the public first-order obligation surface of* $\mathsf{NameCert}_{\mathsf{DeformQuantization}^{\uparrow}}$*, with no additional bracket source, analytic convergence witness, or formality datum.*

Proof. For the carrier part, apply Definition 271.3. It identifies the accepted first-order carrier with the displayed BHist ledger, the Cont commutator face, the packet hsame comparisons, and the Pkg provenance supplied by the $\mathsf{Symplectic}^{\uparrow}$ and $\mathsf{HopfAlg}^{\uparrow}$ dependencies.

For classifier movement, apply Definition 271.4. Transport of the degree-zero product endpoint, the two first-order coefficient endpoints, the commutator endpoint, and the Poisson-bracket row is componentwise over the transported Σ and H rows.

For readback exactness, apply Definition 271.5. It forces every accepted first-order bracket read through the degree-one commutator endpoint κ, the Poisson-bracket obligation row π, and the comparison $\mathsf{hsame}(\kappa, \pi)$ before the public endpoint is reported.

The remaining source check is Definition 271.8. It rules out any source outside Σ, H, Cont, hsame, BHist, and Pkg. Repacking these projections through Definition 271.9 gives the public first-order obligation surface and no larger primitive boundary. □

Theorem 271.8 (DeformQuantization source-boundary obligation)**.** *Every source read accepted by the inventory of Definition 271.6 factors through the supplied* $\mathsf{Symplectic}^{\uparrow}$ *boundary* Σ *and the supplied* $\mathsf{HopfAlg}^{\uparrow}$ *formal-product row* H*. In particular, a bracket-facing read reaches the public packet only through the symplectic consumer boundary, the Hopf formal-product provenance, the coefficient continuation rows, and the first-order exactness comparison of Definition 271.5.*

Proof. Project the read into the finite inventory. The carrier case is exactly Definition 271.3: it exposes the star-product carrier through the displayed BHist ledger, the Hopf Pkg provenance, the Cont face, and the packet hsame comparisons.

The classifier and stability cases use Definition 271.4. Transport of the degree-zero endpoint, first-order coefficient endpoints, commutator endpoint, and Poisson comparison is therefore controlled by the transported $\mathsf{Symplectic}^{\uparrow}$ and $\mathsf{HopfAlg}^{\uparrow}$ rows, not by an independent deformation source.

For the bracket readback, apply Definition 271.5. It factors the read through κ, π, and $\mathsf{hsame}(\kappa, \pi)$ after the Hopf formal-product row has supplied the coefficient spine and after the symplectic boundary has supplied only the Poisson consumer frontier. Thus every source read is bounded by the two supplied dependencies and the named BEDC transport rows. □

Theorem 271.9 (DeformQuantization public certificate obligation surface)**.** *The inventory of Definition 271.6, together with the source-boundary factorization of Definition 271.8, forms the public* $\mathsf{NameCert}_{\mathsf{DeformQuantization}^{\uparrow}}$ *obligation surface at first order. The surface contains carrier, classifier, stability, ledger, exactness, and public certificate rows over* $\mathsf{Symplectic}^{\uparrow}$*,* $\mathsf{HopfAlg}^{\uparrow}$*,* Cont*,* hsame*,* BHist*, and* Pkg*, and it does not claim Kontsevich formality, analytic convergence, or a full star-product construction beyond the displayed first-order ledger.*

Proof. Assemble the carrier and classifier fields first. The carrier field is supplied by Definition 271.3; the classifier and transported endpoint fields are supplied by Definition 271.4.

Next add stability and exactness. The stability rows are the displayed Cont and hsame fields in the ledger, and the exactness row is Definition 271.5. These rows cover the first-order commutator endpoint and its comparison with the Poisson-bracket obligation row.

Finally apply Definition 271.8. It ensures that every public read is sourced by Σ, H, and the named BEDC rows in the inventory. Repacking those fields gives the stated public obligation surface while leaving full formality and convergence outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DeformQuantization}^{\uparrow}}$.

Constructive story: A deformation-quantization packet is generated over an accepted Symplectic$^{\uparrow}$ boundary and an accepted HopfAlg$^{\uparrow}$ formal-product row by displaying the formal product endpoint, its degree-zero product, two degree-one coefficient endpoints, their commutator endpoint, the Poisson-bracket obligation row, and the exactness comparison between them. The carrier accepts only this first-order star-product ledger with its Cont commutator face, hsame comparisons, coefficient rows, and Pkg provenance from the two dependencies; Kontsevich formality, analytic convergence, and a full star-product construction beyond the visible first-order ledger are not carrier data. Two accepted packets classify together when the symplectic and Hopf rows transport and the degree-zero endpoint, the two first-order endpoints, the commutator endpoint, and the Poisson row all transport by the displayed hsame comparisons, so the source specification is the supplied dependency pair and the pattern specification is the formal-product ledger whose antisymmetric first-order endpoint is compared with the bracket row. The ledger keeps the raw coefficient and commutator histories visible beside the public bracket endpoint: every bracket read factors through the commutator row, the Poisson row, and the exactness comparison before being reported. Once such a ledger is displayed, the five name-certificate obligations are supplied by the inherited dependency certificates and the componentwise transport rows: the ledger gives a carrier witness, the classifier is reflexive, symmetric, and transitive on accepted endpoints, and transported coefficient and bracket rows remain inside the same first-order surface. A history h lies in the signature gap of a DeformQuantization$^{\uparrow}$ token p when the source-domain accepts its symplectic boundary, Hopf formal-product row, coefficient ledger, and exactness row, the signature relation pairs it with an answer ledger and evidence, and the token-introduction relation names that answer as p; unfolded, the degree-one commutator in h must be hsame to the Poisson-bracket row encoded by p. The chapter registers this first-order ledger construction under the name DeformQuantization$^{\uparrow}$; it adds no analytic deformation object, formality theorem, convergence witness, standard bridge, or independent Poisson-bracket source.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DeformQuantization}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope is exactly the carrier row, bracket classifier, coefficient and commutator stability rows, first-order ledger, exactness row, source-boundary factorization, and public certificate rows over Symplectic$^{\uparrow}$, HopfAlg$^{\uparrow}$, Cont, hsame, BHist, and Pkg.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DeformQuantization}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DeformQuantizationUp.DeformQuantizationStarLedger\_first\_order\_bracket`

Bridge status: paperBridge

Not claimed: This seed does not claim Kontsevich formality, full deformation quantization, analytic convergence, a machine-checked standard bridge, or closure beyond the displayed first-order ledger.

Upgrade path: No further paper-axis theory grade remains; bridgeChecked status remains a separate formal-axis upgrade.

Chapter 272

A Concrete Naming Certificate for $\mathsf{ChernWeil}^{\uparrow}$

The Chern-Weil interface $\mathsf{ChernWeil}^{\uparrow}$ packages the cohomology-class invariants of a principal bundle as polynomial expressions in the curvature form of any chosen connection. It depends on $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ (chapter 135) and $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ (chapter 137), and bridges differential and topological certificates through the same characteristic-class classifier.

272.1 Carrier definition

Definition 272.1 (Chern-Weil source envelope)**.** *A* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope is a finite* BHist *packet*

$$\mathcal{W} = (\mathcal{R}, \mathcal{D}, p, \gamma, \lambda)$$

where $\mathcal{R}$ *is a public* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *packet carrying the curvature boundary row and provenance exported by Definition 135.45,* $\mathcal{D}$ *is a* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *packet carrying the cohomology row into which the characteristic class is read,* $p : \mathsf{BHist}$ *is the shared* Cont *provenance row connecting the curvature source to the de Rham projection,* $\gamma : \mathsf{BHist}$ *is the ledger face recording the chosen connection used to read the curvature polynomial, and* $\lambda : \mathsf{BHist}$ *is the characteristic-class row returned by the de Rham classifier after the* Cont *projection.*

The envelope is admitted only when the curvature boundary, the de Rham cohomology row, the shared Cont *provenance, the chosen-connection ledger face, and the characteristic-class row all occur as rows of the same finite history packet. Thus* $\mathsf{ChernWeil}^{\uparrow}$ *carries the concrete curvature-to-cohomology passage rather than an ambient characteristic-class operation.*

Lean4 (definition) `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope`

Definition 272.2 (Chern-Weil characteristic classifier)**.** *Two* $\mathsf{ChernWeil}^{\uparrow}$ *source envelopes* $\mathcal{W}$ *and* $\mathcal{W}'$ *classify the same Chern-Weil object exactly when* hsame *preserves their* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *source rows and the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *classifier returns the same characteristic-class row after projecting both packets through their carried* Cont *provenance rows. Equivalently, the classifier ignores only the hidden connection ledger face* γ *and compares the exported curvature source row together with the de Rham row* λ *read from it.*

Lean4 (definition) `BEDC.Derived.ChernWeilUp.ChernWeilEnvelopeCharacteristicClassifier`

Definition 272.3 (Chern-Weil curvature-to-de Rham source packet)**.** *A* $\mathsf{ChernWeil}^{\uparrow}$ *source packet is a finite BEDC package*

$$\mathcal{W} = (\mathcal{R}, \mathcal{D}, p, \chi, c, \epsilon, \lambda, \pi)$$

where $\mathcal{R}$ *is a* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *carrier packet,* $\mathcal{D}$ *is a* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *carrier packet,* $p : \mathsf{BHist}$ *is the invariant-polynomial row,* $\chi : \mathsf{BHist}$ *is the evaluated curvature-polynomial row,* $c : \mathsf{BHist}$ *is the de Rham cohomology-class endpoint,* ϵ *is the exact-boundary row witnessing connection replacement,* λ *is the finite* Cont *ledger from the curvature Bianchi row to the closed de Rham row, and* π *records the* Pkg *provenance of the curvature, de Rham, and polynomial rows.*

The packet is accepted only when the source curvature rows, the de Rham endpoint rows, the hsame *comparisons, the* Cont *ledgers, the* $\mathsf{ProbeBundle}$ *observations, and the* Pkg *provenance all project from the same displayed history package. Thus the Chern-Weil source is a CurvatureUp-to-DeRhamUp row package, not a host equality between differential forms.*

Definition 272.4 (Chern-Weil characteristic-class classifier)**.** *Two packets of Definition 272.3 have the same Chern-Weil characteristic class when their invariant-polynomial rows classify together, their evaluated curvature rows transport by the carried* hsame *and* Cont *ledgers, and the transported endpoints determine the same de Rham cohomology-class row in the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *classifier. The classifier is therefore agreement of invariant-polynomial curvature rows after de Rham cohomology-class transport; it is not equality of ambient forms in a host calculus.*

Lean4 (definition) `BEDC.Derived.ChernWeilUp.ChernWeilCharacteristicClassClassifier`

Theorem 272.5 (Chern-Weil invariant-polynomial carrier obligation)**.** *Every accepted Chern-Weil row exposes only the source curvature packet, the invariant-polynomial evaluation history, the de Rham class endpoint, the finite transport ledgers, the* $\mathsf{ProbeBundle}$ *observations, and the* Pkg *provenance. In particular, the public carrier does not expose a coordinate connection form, an ambient polynomial algebra, or an external cohomology quotient beyond the displayed* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *rows.*

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope carrier obligation`

Proof. Unpack an accepted packet of Definition 272.3. The tuple projections give exactly the source curvature certificate, the source de Rham certificate, the polynomial row, the evaluated curvature-polynomial row, the class endpoint, the exact-boundary row, the finite ledger, and the provenance row.

The classifier of Definition 272.4 consumes only these projections. It compares the polynomial and evaluated rows by hsame, transports them through the finite Cont ledger, and then reads the resulting endpoint through the de Rham classifier row.

Repacking those projections gives the stated public carrier surface. Since no projection returns a host connection form, an external polynomial algebra, or an uncarried quotient, no accepted Chern-Weil row can expose such data. □

Theorem 272.6 (Chern-Weil connection-change cocycle obligation)**.** *If two packets of Definition 272.3 differ only by replacing the* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *representative inside their* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *source while preserving the carried base, fibre, section, invariant-polynomial, and provenance rows, then the two evaluated curvature-polynomial endpoints are related in* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *by the exact-boundary row* ϵ*. Consequently the characteristic-class classifier transports across the connection change.*

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope connection change cocycle obligation`

Proof. Project the two Chern-Weil packets to their curvature sources. The curvature tensorial transport row carries the section and tangent replacement data through hsame, while preserving the shared base, fibre, invariant-polynomial, and provenance projections.

Evaluate the invariant-polynomial row against the transported curvature boundary row. The evaluation history is part of the Chern-Weil packet, so the two evaluated endpoints are connected by the displayed finite Cont ledger rather than by an unstated analytic homotopy.

The exact-boundary row ϵ is then projected as a de Rham boundary witness from the first endpoint to the second. Reading this witness through the de Rham classifier gives the same cohomology-class endpoint, and repacking the transported rows proves classifier stability under the connection change. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierPacket connection change cocycle obligation`

Theorem 272.7 (Chern-Weil closed-form de Rham class obligation)**.** *For every accepted Chern-Weil packet, the Bianchi-style curvature ledger supplies a closed de Rham class endpoint consumable by the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *classifier. The closedness row is the finite* Cont *consequence of the curvature cyclic ledger, the invariant-polynomial evaluation row, and the transported de Rham endpoint row.*

Proof. Start from the curvature component of the source packet. Its Bianchi ledger is a finite cyclic Cont row among displayed curvature boundaries, and the Chern-Weil packet records the invariant-polynomial evaluation of those boundaries.

Transport the evaluated row along the carried hsame comparisons into the de Rham source packet. The ledger λ records this transport as a BEDC continuation from the Bianchi row to the closed de Rham endpoint.

The de Rham classifier consumes the transported endpoint as a closed class row. Thus closedness is not asserted as an external differential identity; it is the class endpoint projected from the finite curvature ledger and repacked inside $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope closed form derham class obligation`
Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope closed form derham class obligation`

Theorem 272.8 (Chern-Weil NameCert obligation surface)**.** *The* $\mathsf{ChernWeil}^{\uparrow}$ *obligation surface consists exactly of the source packet of Definition 272.3, the characteristic-class classifier of Definition 272.4, the invariant-polynomial carrier row of Definition 272.5, the connection-change exact-boundary row of Definition 272.6, the closed de Rham endpoint row of Definition 272.7, the finite ledgers, and the dependency provenance over* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$.

Proof. Collect the two definitions and the three obligation rows above. They provide the carrier packet, classifier, exposed public fields, connection-change transport, closedness endpoint, ledgers, exact-boundary row, and dependency provenance.

Each field has a consumer: the carrier exposes the source histories, the classifier reads the transported de Rham endpoint, the connection-change row supplies exact-boundary transport, and the closedness row supplies the de Rham class endpoint. Conversely, every consumer reads one of these displayed fields.

Therefore the surface is exact. A downstream $\mathsf{NameCert}_{\mathsf{ChernWeil}^{\uparrow}}$ consumer may project the listed CurvatureUp-to-DeRhamUp rows, but it may not assume a stronger equality of forms, a coordinate curvature model, or a standard bridge absent from the packet. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope namecert obligation surface`

Definition 272.9 (Chern-Weil carrier packet)**.** *A* $\mathsf{ChernWeil}^{\uparrow}$ *carrier packet is a displayed* BHist *tuple*

$$\mathcal{W} = (\mathcal{R}, \mathcal{D}, b, E, \kappa, p, \pi, \omega, [\omega], \rho, \lambda)$$

where $\mathcal{R}$ *is a source* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *packet,* $\mathcal{D}$ *is a source* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *packet,* $b : \mathsf{BHist}$ *is the shared base row,* $E : \mathsf{BHist}$ *is the bundle row,* $\kappa : \mathsf{BHist}$ *is the curvature-boundary row projected from* $\mathcal{R}$*,* $p : \mathsf{BHist}$ *is the invariant-polynomial row,* $\pi : \mathsf{BHist}$ *is the polynomial endpoint row obtained by applying* p *to* κ*,* $\omega : \mathsf{BHist}$ *is the closed differential-form representative row read by* $\mathcal{D}$*,* $[\omega] : \mathsf{BHist}$ *is the de Rham class row,* ρ *records the* Pkg *provenance of the curvature and de Rham dependency packets, and* λ *is the finite* Cont *ledger from the boundary and polynomial rows to the endpoint, representative, and class rows.*

The packet is accepted only when the curvature source row comes from $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$*, the cohomology representative row comes from* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$*, and the displayed* Pkg *provenance binds both sources to the same characteristic-class classifier. Thus the carrier is the finite BEDC history packet for a Chern-Weil class, not an equality between host-level connections, polynomials, forms, or cohomology classes.*

Theorem 272.10 (Chern-Weil curvature polynomial stability obligation)**.** *For packets of Definition 272.9, transporting the* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *boundary row* κ *by inherited* hsame *and by the finite* Cont *ledger transports the polynomial endpoint row* π *under the same characteristic-class classifier. The invariant-polynomial row* p*, the dependency* Pkg *provenance* ρ*, and the de Rham source packet* $\mathcal{D}$ *are retained as ledger data rather than recomputed.*

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierPacket curvature polynomial stability obligation`

Proof. Read the carrier through the roadmap ten-field template of section 9467.18: the source fields are $\mathcal{R}$ and $\mathcal{D}$, the carrier field is the displayed BHist tuple, the classifier field is the characteristic-class classifier, and the ledger field is λ.

Apply Definition 135.45 to project the curvature-boundary row from $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$. The allowed transport of that row is exactly the inherited hsame comparison together with the finite Cont readback exposed by the curvature certificate.

Append the polynomial row p to that transported boundary inside the same finite ledger. Since π is the endpoint row of the displayed ledger, transporting κ transports π with no appeal to host equality of forms or connections. The Pkg provenance and the $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ dependency row remain fixed inputs to the same classifier surface. □

Theorem 272.11 (Chern-Weil de Rham class exactness obligation)**.** *For packets of Definition 272.9, the de Rham class row* $[\omega]$ *is accepted exactly when the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *dependency row reads the polynomial endpoint* π *as a closed cohomology representative* ω*. No additional Chern-Weil class row is available unless it is reached by the displayed* Cont *ledger and classified by the same* $\mathsf{NameCert}$ *dependency surface.*

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope derham class exactness obligation`

Proof. Use the exactness field of the roadmap ten-field template in section 9467.18. Exactness asks that every public row be generated from the listed source, carrier, classifier, stability, and ledger data.

The source row for closed cohomology representatives is the $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ dependency chapter chapter 137. Feed it the polynomial endpoint row π carried by the Chern-Weil packet. Its accepted output is the representative row ω together with the class row $[\omega]$.

Conversely, if $[\omega]$ is displayed as a Chern-Weil class row, the packet definition lists no source other than $\mathcal{D}$ and no path other than λ from π to ω and then to $[\omega]$. Therefore class acceptance is precisely de Rham acceptance of the polynomial endpoint as a closed representative, mediated by BHist, Cont, Pkg, and NameCert rows. □

Theorem 272.12 (Chern-Weil characteristic-class ledger obligation)**.** *In a* $\mathsf{ChernWeil}^{\uparrow}$ *carrier packet, the choice of connection, invariant-polynomial presentation, representative differential form, and class representative is ledger data. Carrier equality is controlled only by the shared* BHist *source rows, inherited* hsame *transport, the finite* Cont *ledger, and the characteristic-class classifier; it is not equality of the chosen connection, polynomial syntax, form representative, or cohomology representative in an external theory.*

Proof. Project the connection-dependent curvature boundary from $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ using Definition 135.45. That theorem exposes the boundary row, cyclic ledger, tensorial transport row, and provenance rows, but it does not export a host-level curvature form or connection equality.

Project the cohomology representative side from the $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ dependency chapter chapter 137. The representative and class rows enter the Chern-Weil packet as BHist rows reached by λ, with their source provenance recorded in ρ.

Thus changing a connection, a polynomial presentation, a representative form, or a representative class can affect the packet only by changing ledger entries. The classifier compares the carried rows through hsame and finite Cont transport, so the choices are part of the certificate ledger rather than part of carrier equality. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope characteristic ledger transport`
Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope characteristic ledger exactness`

Theorem 272.13 (Chern-Weil root NameCert obligation surface)**.** *The minimum* $\mathsf{NameCert}_{\mathsf{ChernWeil}^{\uparrow}}$ *obligation surface consists of the carrier packet of Definition 272.9, the curvature-polynomial stability row of Definition 272.10, the de Rham class exactness row of Definition 272.11, and the characteristic-class ledger row of Definition 272.12. Downstream bridge chapters may consume the curvature boundary, invariant-polynomial endpoint, de Rham representative, class row, provenance row, and finite ledger, but may not assume a standard bridge or a stronger analytic Chern-Weil object from this root surface alone.*

Proof. Assemble the four rows by the roadmap ten-field template of section 9467.18. The traditional target is a Chern-Weil characteristic class; the BEDC sources are $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$; the carrier is the displayed BHist packet; the classifier is the characteristic-class classifier; the stability, ledger, and exactness fields are the three theorem rows above.

The curvature component is licensed by Definition 135.45. The cohomology component is licensed by the dependency chapter chapter 137. The shared primitives are exactly BHist for rows, hsame for transported endpoint comparison, Cont for finite readback, Pkg for provenance, and NameCert for dependency admission.

Therefore every downstream consumer receives a finite obligation surface with visible carrier, classifier, stability, ledger, and exactness rows. Anything beyond those rows, including a standard bridge or an analytic equality of characteristic classes, must be supplied by a later bridge certificate rather than read from $\mathsf{ChernWeil}^{\uparrow}$ itself. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope semantic name certificate`

272.2 The certificate

Theorem 272.14 (Chern-Weil curvature-de Rham obligation surface)**.** *The public obligation surface of a* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope consists of the* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *export row of Definition 135.45, the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *cohomology row carried by the de Rham certificate, the shared* Cont *provenance row, the chosen-connection ledger face, and the characteristic-class row returned by the de Rham classifier. No connection-dependent differential form is exported except through the curvature source row and the hidden ledger face.*

Proof. Project the envelope of Definition 272.1. Its curvature component exposes the public boundary, Bianchi, transport, and provenance rows supplied by $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$, and Definition 135.45 is the export rule for those rows.

Project the de Rham component next. The de Rham packet supplies the cohomology row into which the characteristic-class row λ is read, while the carried Cont provenance row records that this readout is made from the same finite history packet as the curvature source.

The remaining row is the connection ledger face γ. It is listed as a hidden face because the envelope needs it to witness the chosen connection, but the classifier of Definition 272.2 never exports it as part of the public characteristic class. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope public rows`
Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope projection rows`

Theorem 272.15 (Chern-Weil connection-choice stability)**.** *Changing the selected connection in a* $\mathsf{ChernWeil}^{\uparrow}$ *source envelope changes only the hidden ledger face* γ *when the transported curvature source rows are related by* hsame *and the de Rham classifier returns the same characteristic-class row after* Cont *projection. Under those hypotheses the two envelopes classify the same* $\mathsf{ChernWeil}^{\uparrow}$ *object.*

Proof. Begin with two envelopes whose visible curvature source rows are related by hsame. The public curvature certificate transports the boundary and provenance rows through the same $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ export surface used in Definition 272.14.

Apply the shared Cont projections to the transported rows. By hypothesis the de Rham classifier returns the same characteristic-class row after those projections, so the visible cohomology row λ is unchanged.

The chosen connection can therefore differ only in the ledger face γ. Since Definition 272.2 compares the curvature source and the de Rham class while ignoring that hidden face, the two envelopes have the same classifier value. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope connection choice stability`

Theorem 272.16 (Chern-Weil characteristic ledger exactness)**.** *The* $\mathsf{ChernWeil}^{\uparrow}$ *ledger is exact: every admitted source envelope is covered by a curvature source row, a de Rham cohomology row, a shared* Cont *provenance row, a selected-connection ledger face, and a characteristic-class row; and every public classifier comparison is sound exactly when it is generated by* hsame *on the curvature source and equality of the de Rham characteristic-class row after* Cont *projection.*

Proof. Coverage is by projection from the displayed envelope. Each admitted packet contains the curvature certificate row, the de Rham certificate row, the shared provenance row, the connection ledger face, and the characteristic-class row as finite BHist data.

For soundness, take a public classifier comparison. The definition requires hsame on the curvature source row and agreement of the de Rham characteristic-class row after the carried Cont projections, so every accepted comparison is generated by listed envelope data.

Conversely, if those listed rows are present and the two classifier conditions hold, repack them as the comparison of Definition 272.2. No unlisted host connection, differential-form representative, or external characteristic-class operator is needed. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilSourceEnvelope characteristic ledger exactness`

Theorem 272.17 (Chern-Weil standard bridge boundary)**.** *The standard bridge boundary for the public* ChernWeil$^{\uparrow}$ *certificate reads only the displayed curvature source row, the de Rham cohomology row, the invariant-polynomial endpoint, the characteristic-class classifier, the finite* Cont *projections,* hsame *transport, and* Pkg *provenance already listed in the carrier envelope. Analytic Chern-Weil equality and external cohomology quotients are outside this bridge boundary.*

Proof. Start from the carrier packet of Definition 272.9. Its displayed rows are the curvature source, de Rham target, invariant polynomial, connection ledger, characteristic endpoint, and provenance packet.

Apply Definition 272.13 to restrict the root certificate to the carrier, stability, de Rham exactness, and characteristic-ledger rows. Apply Definition 272.14 to read the curvature source row and de Rham cohomology row through the same finite Cont provenance.

Finally use Definition 272.16. It says that every public classifier comparison is generated by hsame on the curvature source and equality of the de Rham characteristic-class row after the listed Cont projections. Since no step introduces analytic equality of differential forms or an external cohomology quotient, those objects remain outside the standard bridge boundary. □

Lean4 ✓ `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope standard bridge boundary`

Closure status for NameCert$_{\text{ChernWeil}^{\uparrow}}$**.**

Constructive story: A Chern-Weil packet is assembled from a certified Curvature$^{\uparrow}$ source and a certified DeRham$^{\uparrow}$ target by displaying the curvature boundary, an invariant-polynomial row, the evaluated curvature-polynomial endpoint, the closed representative and class rows, the chosen-connection ledger face, finite Cont projections, and one Pkg provenance packet. The carrier accepts only this curvature-to-de Rham history package; coordinate connection forms, ambient polynomial algebras, external cohomology quotients, and host equalities of differential forms are not accepted unless they appear as the listed BEDC rows. Two accepted packets classify together when the curvature source transports by hsame and the de Rham classifier returns the same characteristic-class row after the carried Cont projections, while the chosen connection remains ledger data rather than part of public equality. Thus the source specification covers the CurvatureUp-to-DeRhamUp packet, the pattern specification is invariant-polynomial evaluation followed by de Rham class readback and connection-change exactness, and the ledger keeps the raw curvature, polynomial, representative, class, connection, projection, and provenance rows aligned with the visible characteristic class. The five name-certificate obligations close because an accepted packet inhabits the carrier, inherited hsame and de Rham classification give reflexivity, symmetry, and transitivity, and connection or representative transport repacks the same finite rows into another accepted packet. A history h lies in the signature gap of a ChernWeil$^{\uparrow}$ token p when the source-domain accepts h as this curvature-to-cohomology package, the signature relation pairs it with an answer characteristic-class packet and evidence ledger, and the token-introduction relation names that answer as p; concretely, h must expose the curvature row, invariant polynomial, de Rham endpoint, exact-boundary or connection ledger, Cont projections, and provenance encoded by p. The chapter registers this finite characteristic-class ledger and its paper bridge boundary under the name ChernWeil$^{\uparrow}$; it introduces no analytic Chern-Weil theorem, external polynomial calculus, or independent cohomology quotient.

Theory closure: TheoryGate(NameCert$_{\text{ChernWeil}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public ChernWeilUp carrier envelope, semantic NameCert certificate, standard bridge boundary of Definition 272.17, displayed characteristic-class ledger examples, and completeness treatment are exported over the displayed curvature, characteristic-form, BHist, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus(NameCert$_{\text{ChernWeil}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ChernWeilUp.ChernWeilCarrierEnvelope\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not close the standard analytic Chern-Weil theorem, external polynomial calculus, or an independent cohomology quotient.

Upgrade path: No further paper-axis upgrade is recorded; machine bridge checking remains on the separate bridge and formal axes.

Chapter 273

A Concrete Naming Certificate for $\mathsf{AtiyahSinger}^{\uparrow}$

The Atiyah-Singer-index interface $\mathsf{AtiyahSinger}^{\uparrow}$ packages the equality between the analytic index of an elliptic operator on a closed manifold and a topological index defined via Chern characters and Todd classes. It depends on $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ (chapter 214) and $\mathsf{NameCert}_{\mathsf{ChernWeil}^{\uparrow}}$ (chapter 272), and bridges spectral and topological certificates through the same index pairing.

273.1 Carrier definition

273.2 The certificate

Definition 273.1 (Atiyah-Singer index-pairing carrier packet)**.** *An* $\mathsf{AtiyahSinger}^{\uparrow}$ *index-pairing carrier packet is a finite* BHist *tuple*

$$\mathcal{A} = (m, D, \sigma_D, \mathcal{S}, \alpha, \mathcal{W}, \theta, \tau, \eta, \rho)$$

with the following rows. The row $m : \mathsf{BHist}$ *is the closed-manifold endpoint. The row* $D : \mathsf{BHist}$ *is the elliptic-operator endpoint over* m*, and* $\sigma_D : \mathsf{BHist}$ *is its symbol row. The packet* $\mathcal{S}$ *is a* $\mathsf{NameCert}_{\mathsf{SpectralTheorem}^{\uparrow}}$ *source whose operator row is* D *and whose spectral-data ledger is read over the same endpoint* m*. The row* $\alpha : \mathsf{BHist}$ *is the analytic-index endpoint projected from that spectral source. The packet* $\mathcal{W}$ *is a* $\mathsf{ChernWeil}^{\uparrow}$ *source packet over the same endpoint* m*. The row* $\theta : \mathsf{BHist}$ *is the Chern-character and Todd characteristic-class row projected from* $\mathcal{W}$*, and* $\tau : \mathsf{BHist}$ *is the topological-index endpoint obtained by the displayed characteristic-class pairing with* σ_D *over* m*. The row* $\eta : \mathsf{BHist}$ *is the index-pairing equality obligation, and* ρ *records the* Pkg *provenance of the spectral, Chern-Weil, symbol, endpoint, and pairing rows.*

The packet is admitted only when the analytic side and the topological side are displayed over the identical closed-manifold endpoint row m*. Its public classifier row is the index-pairing predicate*

$$\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau) :\Longleftrightarrow \mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha) \wedge \mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau) \wedge \mathsf{hsame}(\alpha, \tau).$$

Here $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha)$ *names the analytic-index projection from the* $\mathsf{SpectralTheorem}^{\uparrow}$ *dependency surface, while* $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau)$ *names the Chern-Weil characteristic-class projection and its displayed pairing endpoint. Thus the packet carries an equality obligation between two named index endpoints, not a host-level Fredholm theorem or an ambient equality of characteristic forms.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerTasteGate single carrier alignment`

Theorem 273.2 (Atiyah-Singer index-pairing carrier surface)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. If its* $\mathsf{SpectralTheorem}^{\uparrow}$ *component supplies the analytic-index row α for the elliptic-operator row D over the closed-manifold endpoint m, and its* $\mathsf{ChernWeil}^{\uparrow}$ *component supplies the characteristic-class row θ and topological-index endpoint τ for the symbol row σ_D over the same endpoint m, then $\mathcal{A}$ exposes the public index-pairing classifier row η exactly as*

$$\eta = \mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau).$$

The equality obligation in this row is scoped exactly to the two dependency reads

$$\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha) \quad \textit{and} \quad \mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau),$$

together with the endpoint comparison $\mathsf{hsame}(\alpha, \tau)$*. No further analytic spectrum row, coordinate connection form, ambient cohomology quotient, external symbol class, or completed Atiyah-Singer theorem is exported by this carrier surface.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket carrier surface`

Proof. Unpack the accepted packet of Definition 273.1. Projection gives the closed-manifold endpoint row m, the elliptic-operator row D, the symbol row σ_D, the spectral packet $\mathcal{S}$, the analytic-index endpoint α, the Chern-Weil packet $\mathcal{W}$, the characteristic-class row θ, the topological-index endpoint τ, the equality-obligation row η, and the provenance row ρ.

The spectral component is bounded by the $\mathsf{SpectralTheorem}^{\uparrow}$ surface. By Definition 214.1, the carried operator row and its classifier are read through the certified operator side rather than through host equality. By Definition 214.2, the spectrum, projection-valued-measure, and functional-calculus rows are the visible spectral-data rows. By Definition 214.3 and Definition 214.4, a downstream consumer of $\mathcal{S}$ may use only the displayed carrier, classifier, spectral-data, stability, ledger, exactness, and dependency rows. The Atiyah-Singer packet consumes this surface only through the analytic-index projection named by $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha)$.

The topological component is bounded by the $\mathsf{ChernWeil}^{\uparrow}$ surface. The source packet of Definition 272.3 supplies the curvature-to-de Rham rows, invariant-polynomial row, evaluated characteristic row, endpoint row, exact-boundary row, ledger, and provenance. The classifier of Definition 272.4 compares characteristic classes only after the displayed de Rham transport. The carrier and exactness theorems Definition 272.5, Definition 272.7, and Definition 272.13 exclude coordinate connection forms, ambient polynomial algebras, and uncarried cohomology quotients from the public surface. The Atiyah-Singer packet consumes this surface only through the characteristic-class row θ and the topological-index endpoint τ named by $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau)$.

The shared endpoint condition is part of the admission rule in Definition 273.1. Both dependency reads cite the identical row m before the equality obligation is formed. Therefore there is no hidden transport from one manifold endpoint to another inside η; any such transport would have to appear as an additional displayed ledger before a packet of this definition could be admitted.

It remains to identify the public classifier row. The definition assigns η to the predicate $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau)$, whose three clauses are exactly the spectral analytic-index read, the Chern-Weil topological-index read, and the comparison $\mathsf{hsame}(\alpha, \tau)$. Coverage follows by projecting η from the accepted packet. Soundness follows because any public index-pairing comparison must unpack this displayed η and hence must supply those two dependency reads and that endpoint comparison. Exactness follows from the tuple inventory: after the listed projections and the provenance row ρ, the packet contains no field from which an additional analytic, topological, or host-theoretic equality could be read. □

Theorem 273.3 (Atiyah-Singer index-pairing provenance exactness)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. Every public read of the analytic-index endpoint α, the Chern-Weil topological endpoint τ, the symbol row σ_D, the shared closed-manifold endpoint m, and the comparison row η factors through the displayed provenance row ρ, the dependency packets $\mathcal{S}$ and $\mathcal{W}$, and the endpoint comparison* $\mathsf{hsame}(\alpha, \tau)$.

Equivalently, the exact public inventory of the packet is the spectral analytic-index read $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S},$ *the Chern-Weil topological-index read* $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau)$*, the shared symbol and closed-manifold rows, the displayed comparison* $\mathsf{hsame}(\alpha, \tau)$*, and the* Pkg *provenance recorded by ρ. The packet exports no host Fredholm index equality, coordinate characteristic form equality, ambient cohomology quotient, or uncarried equality between analytic and topological models.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket provenance exactness`

Proof. Project the accepted tuple. Its fields give exactly m, D, σ_D, $\mathcal{S}$, α, $\mathcal{W}$, θ, τ, η, and ρ, with the shared endpoint condition imposed before the classifier row is read.

Apply Definition 273.2. It identifies the classifier row η with $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau)$, so every public comparison read must pass through the spectral analytic-index read, the Chern-Weil topological-index read, and $\mathsf{hsame}(\alpha, \tau)$.

The endpoint and symbol reads are then covered by the same tuple inventory. The analytic endpoint α is read only through the dependency packet $\mathcal{S}$, the topological endpoint τ only through $\mathcal{W}$, and the shared rows m and σ_D are the displayed coordinates used by both dependency clauses.

Finally, the provenance row ρ records precisely the Pkg source of these dependency, endpoint, symbol, and comparison rows. Since the packet has no field beyond the projected inventory, no host Fredholm equality, coordinate form equality, ambient cohomology quotient, or uncarried analytic-topological identity can be exported by a public read of $\mathcal{A}$. □

Theorem 273.4 (Atiyah-Singer index-pairing classifier coverage)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. Its accepted public classifier surface is exactly the coverage, soundness, and completeness table whose rows are the spectral analytic-index read* $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha)$*, the Chern-Weil topological-index read* $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau)$*, the shared closed-manifold row m, the shared symbol row σ_D, the displayed comparison* $\mathsf{hsame}(\alpha, \tau)$*, and the provenance row ρ recording the relevant* Pkg *sources.*

Thus every accepted public classifier read is covered by one of these rows, each such row is sound for the displayed index-pairing packet, and the table is complete: no host Fredholm equality, ambient characteristic-class quotient, or uncarried analytic-topological comparison is part of the public classifier surface.

Proof. Apply Definition 273.2. It identifies the public classifier row η with the spectral analytic-index read, the Chern-Weil topological-index read, and the comparison $\mathsf{hsame}(\alpha, \tau)$ over the displayed endpoint and symbol rows.

Apply Definition 273.3. It identifies every public read of α, τ, σ_D, m, and η with the dependency packets, the endpoint comparison, and the provenance row ρ.

Coverage follows by projecting those rows from the accepted tuple $\mathcal{A}$. Soundness follows because each projected row is one of the admission rows of Definition 273.1. Completeness follows by repacking only those projected rows: the tuple has no additional field carrying a host Fredholm equality, an ambient cohomology quotient, or a separate analytic-topological comparison. □

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket classifier coverage`

Theorem 273.5 (Atiyah-Singer shared index-source boundary)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. The spectral analytic-index read* $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S},\alpha)$ *and the Chern-Weil topological-index read* $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W},\theta,\tau)$ *are admitted only over the same closed-manifold endpoint row m. Consequently the comparison* $\mathsf{hsame}(\alpha,\tau)$ *in the public classifier row never compares index endpoints whose dependency reads were formed over distinct manifold endpoints.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket shared index source boundary`

Proof. Unpack the accepted packet of Definition 273.1. Its admission clause states that the analytic side and the topological side are displayed over the identical closed-manifold endpoint row m before the classifier row is formed.

The spectral read is bounded by the $\mathsf{SpectralTheorem}^{\uparrow}$ public surface cited in Definition 273.2: the operator row D and the analytic endpoint α are read through the displayed spectral dependency packet $\mathcal{S}$. The Chern-Weil read is bounded by the $\mathsf{ChernWeil}^{\uparrow}$ surface cited in the same theorem: the symbol row σ_D, the characteristic row θ, and the topological endpoint τ are read through the displayed packet $\mathcal{W}$.

Apply Definition 273.3. Every public read of α, τ, σ_D, m, and η factors through the two dependency packets, the endpoint comparison, and the provenance row ρ. Since the only endpoint row available to both dependency reads is the projected row m, a comparison formed from distinct manifold endpoints cannot be read from an accepted packet. □

Theorem 273.6 (Atiyah-Singer root endpoint exactness frontier)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. Every public Atiyah-Singer endpoint read is one of the carried closed-manifold endpoint m, elliptic-operator row D, symbol row σ_D, spectral analytic-index row α, Chern-Weil topological-index row τ together with its characteristic-class row θ, endpoint comparison* $\mathsf{hsame}(\alpha,\tau)$*, or* Pkg *provenance row ρ from that packet. Conversely, these rows repack into the public index-pairing classifier*

$$\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha,\theta,\tau).$$

No host Fredholm equality, coordinate characteristic-form equality, or cohomology quotient equality is exported by this frontier.

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket root endpoint exactness frontier`

Proof. Project the accepted tuple of Definition 273.1. Its visible endpoint inventory is exactly m, D, σ_D, $\mathcal{S}$, α, $\mathcal{W}$, θ, τ, η, and ρ.

Apply Definition 273.2. It identifies the public classifier row η with the displayed predicate $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha,\theta,\tau)$, whose clauses are the spectral analytic-index read, the Chern-Weil topological-index read, and the comparison $\mathsf{hsame}(\alpha,\tau)$.

Apply Definition 273.3. It restricts every public read of α, τ, σ_D, m, and η to the dependency packets, the endpoint comparison, and the provenance row ρ.

Finally use Definition 273.5. Both dependency reads are admitted over the same closed-manifold endpoint m, so the endpoint frontier has no second manifold row and no host-level equality between analytic and topological models. Repacking the listed rows yields the public classifier and excludes Fredholm equality, coordinate characteristic-form equality, and cohomology quotient equality. □

Theorem 273.7 (Atiyah-Singer spectral Chern-Weil consumer exhaustion)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. Any downstream consumer of the public Atiyah-Singer bridge surface sees exactly the spectral analytic-index read, the Chern-Weil topological-index read, the shared symbol row σ_D, the shared endpoint row m, the comparison* $\mathsf{hsame}(\alpha, \tau)$*, and the* Pkg *provenance row ρ. No additional consumer-visible datum is supplied by the carrier packet.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket spectral chernweil consumer exhaustion`

Proof. Project the accepted packet. The tuple inventory gives the endpoint row m, the elliptic row D, the symbol row σ_D, the dependency packets $\mathcal{S}$ and $\mathcal{W}$, the endpoints α and τ, the characteristic row θ, the classifier row η, and the provenance row ρ.

Apply Definition 273.2. It identifies η with the public predicate whose clauses are precisely $\mathsf{SpecAnalyticIndex}_{m,D}(\mathcal{S}, \alpha)$, $\mathsf{CWTopologicalIndex}_{m,\sigma_D}(\mathcal{W}, \theta, \tau)$, and $\mathsf{hsame}(\alpha, \tau)$.

Apply Definition 273.3. It states that every public read of the analytic endpoint, topological endpoint, symbol row, endpoint row, and comparison row factors through those dependency reads and the Pkg provenance ρ.

The $\mathsf{SpectralTheorem}^{\uparrow}$ citations in Definition 273.2 bound the spectral consumer to the carrier, classifier, spectral-data, stability, ledger, exactness, and dependency rows exposed by $\mathcal{S}$. The $\mathsf{ChernWeil}^{\uparrow}$ citations there bound the topological consumer to the displayed curvature-to-de Rham, invariant-polynomial, characteristic-class, ledger, and provenance rows exposed by $\mathcal{W}$. Thus the listed reads exhaust the public bridge surface. □

Theorem 273.8 (Atiyah-Singer index-pairing host exclusion)**.** *Let $\mathcal{A}$ be an accepted packet of Definition 273.1. The bridge surface exported by $\mathcal{A}$ does not export a host Fredholm equality, coordinate form equality, cohomology quotient, or ambient analytic-topological identity. Its only equality-bearing public row is the displayed comparison* $\mathsf{hsame}(\alpha, \tau)$ *inside the classifier* $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau)$*.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket host exclusion`

Proof. Unpack the carrier packet. The displayed classifier in Definition 273.1 contains the spectral analytic-index read, the Chern-Weil topological-index read, and $\mathsf{hsame}(\alpha, \tau)$; it does not contain a host Fredholm row, a coordinate form row, a quotient row, or an ambient identity row.

Apply Definition 273.2 to identify the public classifier row η with that displayed predicate. Apply Definition 273.3 to identify the complete public inventory as the two dependency reads, the shared symbol and endpoint rows, the endpoint comparison, and the provenance row ρ.

The $\mathsf{SpectralTheorem}^{\uparrow}$ surface used by the spectral dependency excludes host operator equality from the public spectral read. The $\mathsf{ChernWeil}^{\uparrow}$ surface used by the topological dependency excludes coordinate connection forms, ambient polynomial algebras, and uncarried cohomology quotients from the public characteristic-class read. Since $\mathcal{A}$ exports only those two bounded reads and their displayed comparison, none of the excluded host-level objects can be recovered from the bridge surface. □

Theorem 273.9 (Atiyah-Singer NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{AtiyahSinger}^{\uparrow}}$ *obligation surface for the Atiyah-Singer index-pairing certificate consists exactly of the carrier packet of Definition 273.1, the classifier row* $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha, \theta, \tau)$*, the shared endpoint and source-compatibility rows, the spectral and Chern-Weil dependency reads, the endpoint comparison* $\mathsf{hsame}(\alpha, \tau)$*, and the* Pkg *provenance row. The surface excludes host Fredholm equality, coordinate forms, cohomology quotients, and uncarried analytic-topological identities.*

Lean4 ✓ `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket namecert obligation surface`

Proof. Start with the carried packet. Apply Definition 273.2 to identify the public classifier as $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha,\theta,\tau)$ and to bound the equality-bearing row by the displayed spectral read, Chern-Weil read, and $\mathsf{hsame}(\alpha,\tau)$.

Next apply Definition 273.3. This records that every public endpoint, symbol, comparison, and dependency read factors through the two dependency packets and the Pkg provenance row.

The endpoint and source boundary is supplied by Definition 273.5. The dependency reads are then exhausted by Definition 273.7, which leaves only the spectral analytic-index read and the Chern-Weil topological-index read as public consumers of the two source certificates.

Finally apply Definition 273.8. It removes host Fredholm equality, coordinate forms, cohomology quotients, and uncarried analytic-topological identities from the public surface. The remaining rows are precisely the carrier packet, the classifier, the shared compatibility rows, the two dependency reads, $\mathsf{hsame}(\alpha,\tau)$, and Pkg provenance. □

Closure status for $\mathsf{NameCert}_{\mathsf{AtiyahSinger}^{\uparrow}}$.

Constructive story: An Atiyah-Singer index-pairing packet is assembled by reading one spectral-theorem source row and one Chern-Weil source row over a shared manifold endpoint and symbol row, then carrying the analytic-index endpoint, topological-index endpoint, the displayed comparison $\mathsf{hsame}(\alpha,\tau)$, and the Pkg provenance row in a single finite packet. The classifier is only the displayed $\mathsf{ASIndexPair}_{m,D,\sigma_D}(\alpha,\theta,\tau)$ row, so consumer equality is routed through the two dependency reads and the endpoint comparison rather than through a host Fredholm theorem or a cohomology quotient. The source covers exactly the spectral dependency, the Chern-Weil dependency, the shared manifold and symbol data, and the finite package provenance; the packet exports no ambient analytic-topological identity. The chapter registers this index-pairing obligation surface under the name $\mathsf{AtiyahSinger}^{\uparrow}$ while leaving the full external Atiyah-Singer theorem outside the carried surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AtiyahSinger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped surface binds the carried index-pairing carrier, classifier, provenance, shared source rows, consumer-exhaustion rows, host-exclusion boundary, and public $\mathsf{NameCert}_{\mathsf{AtiyahSinger}^{\uparrow}}$ obligation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AtiyahSinger}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.AtiyahSingerUp.AtiyahSingerIndexPairingCarrierPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close the full external Atiyah-Singer theorem, Fredholm-index equality, ambient cohomology quotient, or standard bridge.

Upgrade path: No further paper-side theory closure upgrade remains for this chapter; formal bridge checking remains on the verification axis.

Chapter 274

A Concrete Naming Certificate for $\mathsf{HodgeBridge}^{\uparrow}$

The Hodge-bridge interface $\mathsf{HodgeBridge}^{\uparrow}$ packages the Hodge decomposition of de Rham cohomology of a compact Kähler manifold into harmonic (p, q)-forms together with the Lefschetz / Hard-Lefschetz operators. It depends on $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ (chapter 137) and $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ (chapter 99), and bridges complex-analytic and topological certificates through the harmonic projector.

274.1 Carrier definition

Definition 274.1 (HodgeBridge BHist source packet)**.** *A* $\mathsf{HodgeBridge}^{\uparrow}$ *source packet is a finite* BHist *package*

$$\mathcal{H} = (\mathcal{D}, \mathcal{C}, P, \beta, \ell, \rho, \lambda, \tau)$$

where $\mathcal{D}$ *is a dependency row certified by* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$*,* $\mathcal{C}$ *is a dependency row certified by* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$*,* $P : \mathsf{BHist}$ *is the harmonic projector row,* $\beta : \mathsf{BHist}$ *is the bidegree ledger row,* $\ell : \mathsf{BHist}$ *records the Lefschetz and Hard-Lefschetz operation rows carried by the packet,* ρ *records the* Pkg *provenance binding the de Rham and cohomology dependency packets,* λ *is the finite* Cont *readback from de Rham representatives through harmonic projection to cohomology rows, and* τ *is the family of* hsame *transports preserving the source representative, harmonic, bidegree, Lefschetz, and cohomology endpoint rows.*

The packet is accepted only when these rows occur in one displayed provenance package. Thus $\mathsf{HodgeBridge}^{\uparrow}$ *carries the BEDC history surface relating de Rham and cohomology certificates; it does not export uncarried Kähler geometry, a host Hodge decomposition, or an ambient analytic projector.*

Lean4 (definition) `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket`

Theorem 274.2 (HodgeBridge source dependency surface)**.** *For every packet of Definition 274.1, the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *projection and the* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *projection are rows of the same* BHist *package. The harmonic projector row, bidegree ledger, Lefschetz ledger, finite* Cont *readback,* hsame *transports, and* Pkg *provenance are shared dependency data.*

Proof. Project the tuple $\mathcal{H}$. The first two components are the de Rham and cohomology dependency rows, while ρ records their common provenance package.

The projector row P, bidegree row β, Lefschetz ledger ℓ, continuation readback λ, and transport family τ are further projections from the same finite package.

Repacking those projections gives the original carrier. Hence the de Rham and cohomology sides are not separate host objects; they are dependency rows of one BEDC history packet. □

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket source dependency surface`

Theorem 274.3 (HodgeBridge harmonic projector classifier stability)**.** *If two packets of Definition 274.1 are related by the carried* hsame *transports on their de Rham source rows and by the same* Pkg *provenance, then transport of the harmonic projector preserves the bidegree ledger and carries the finite* Cont *readback to the transported* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *endpoint row.*

Proof. Start with the de Rham source representatives and their carried hsame comparison. This comparison is one of the transport rows in τ, so it remains internal to the displayed history package.

Transport the harmonic projector row P and the bidegree ledger β along the same family. Since P and β are components of the packet, their transported rows remain paired with the transported source representative.

Read the continuation ledger λ after transport. It carries the harmonic row to the cohomology endpoint, and repacking through the $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ dependency row proves stability of the classifier. □

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket harmonic projector classifier stability`

Theorem 274.4 (HodgeBridge ledger exactness)**.** *The* $\mathsf{HodgeBridge}^{\uparrow}$ *ledger is exact for exactly the rows listed in Definition 274.1: the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *source row, the* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *endpoint row, the harmonic projector row, the bidegree ledger, the Lefschetz ledger, the finite* Cont *readback, the* hsame *transports, and the* Pkg *provenance. Coverage, soundness, and completeness range only over those rows and exclude uncarried Kähler geometry and host Hodge decomposition.*

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket ledger exactness`

Proof. Coverage follows by projecting an accepted packet. The accepted tuple displays the two dependency rows, projector row, bidegree row, Lefschetz row, continuation readback, transport family, and provenance row.

For soundness, begin with exactly those displayed rows and the two dependency certificates. Repack them as $\mathcal{H}$, and the carrier definition accepts the package because every row is present in the same provenance surface.

For completeness, take any accepted public $\mathsf{HodgeBridge}^{\uparrow}$ row. The carrier definition contains no field for ambient Kähler geometry and no field for a host Hodge decomposition. Therefore the public row is exhausted by the listed dependency, projector, ledger, transport, continuation, and provenance data. □

Theorem 274.5 (HodgeBridge NameCert obligation surface)**.** *The paper-side* $\mathsf{NameCert}_{\mathsf{HodgeBridge}^{\uparrow}}$ *obligation surface consists of the carrier packet of Definition 274.1, the shared dependency surface of Definition 274.2, the harmonic projector classifier stability row of Definition 274.3, and the ledger exactness row of Definition 274.4. Downstream consumers may project the de Rham dependency, cohomology dependency, harmonic, bidegree, Lefschetz,* Cont, hsame, *and* Pkg *rows, but no stronger Hodge bridge is part of this surface.*

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket namecert obligation surface`

Proof. Assemble the carrier packet from Definition 274.1. It provides the de Rham source row, cohomology endpoint row, harmonic projector, bidegree ledger, Lefschetz ledger, continuation readback, transport rows, and provenance.

Use Definition 274.2 to bind the two dependency projections to one package. Use Definition 274.3 to transport the harmonic classifier and its bidegree ledger across hsame replacement.

Finally apply Definition 274.4. Its coverage, soundness, and completeness clauses exhaust the public rows, so repacking the listed data gives the $\mathsf{NameCert}_{\mathsf{HodgeBridge}^{\uparrow}}$ obligation surface and excludes unlisted analytic bridge data. □

Theorem 274.6 (HodgeBridge shared-projector readback exactness)**.** *Any public readback of a* $\mathsf{HodgeBridge}^{\uparrow}$ *packet that consumes both the* $\mathsf{NameCert}_{\mathsf{DeRham}^{\uparrow}}$ *source row and the* $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ *endpoint row factors through the single displayed provenance package, harmonic projector row, bidegree ledger, Lefschetz ledger, finite* Cont *readback, and carried* hsame *transport family of Definition 274.1. It exports no uncarried Kähler geometry, ambient analytic projector, or host Hodge decomposition.*

Proof. First project the accepted packet. By Definition 274.2, the de Rham dependency row and the cohomology endpoint row are rows of one finite BHist package, and the provenance coordinate ρ binds them to the same displayed source.

Next read the projector data through the carried transport family. Definition 274.3 keeps the harmonic projector and bidegree ledger stable under the packet's hsame rows, and it carries the finite Cont readback to the transported $\mathsf{NameCert}_{\mathsf{Cohomology}^{\uparrow}}$ endpoint row.

Finally combine Definition 274.4 with Definition 274.5. The public certificate is exhausted by the displayed dependency, projector, bidegree, Lefschetz, continuation, transport, and provenance rows. Because the carrier has no field for uncarried Kähler geometry, an ambient analytic projector, or a host Hodge decomposition, none of those structures is exported by the readback. □

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket shared projector readback exactness`

Theorem 274.7 (HodgeBridge public certificate handoff)**.** *A public* $\mathsf{NameCert}_{\mathsf{HodgeBridge}^{\uparrow}}$ *consumer may read exactly the de Rham dependency row, the cohomology dependency row, the harmonic projector row, the bidegree ledger, the Lefschetz ledger, the finite* Cont *readback, the* hsame *transport family, and the* Pkg *provenance supplied by Definition 274.1 and exhausted by Definition 274.5. The handoff exports no host Hodge decomposition, no ambient analytic projector, and no uncarried Kähler geometry.*

Proof. Project the accepted source packet. Its displayed coordinates supply the de Rham dependency row, cohomology dependency row, harmonic projector, bidegree ledger, Lefschetz ledger, finite continuation readback, transport family, and package provenance.

Apply Definition 274.2 to bind the two dependency rows to one provenance package. Then use Definition 274.3 for the harmonic classifier transport and Definition 274.4 for the exact public ledger rows.

Finally repack exactly those projected fields as the consumer handoff. The obligation surface of Definition 274.5 exhausts the public projections, so there is no coordinate from which to read a host Hodge decomposition, an ambient analytic projector, or uncarried Kähler geometry. □

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket public certificate handoff`

Theorem 274.8 (HodgeBridgeUp bridge-status surface)**.** *The checked public handoff of Definition 274.7 exposes only the* BHist *source packet, the* Cohomology$^{\uparrow}$ *and* DeRham$^{\uparrow}$ *dependency rows, the harmonic-projector classifier, the shared projector readback,* Cont, hsame, *and the* Pkg *ledger rows. Analytic Hodge decomposition, ambient harmonic analysis, and Kähler geometry are not part of this bridge surface.*

Proof. Start with Definition 274.5. It gives the finite source packet, the two dependency rows, the projector and bidegree ledger, continuation readback, transport, and package provenance as the public obligation surface.

Then apply Definition 274.6. The shared projector readback is exhausted by the carried projector row and the displayed dependency endpoints; no ambient analytic projector is introduced by the readback.

Finally use Definition 274.7. Its exported rows are exactly the public packet rows just listed. Since the handoff has no field for analytic Hodge decomposition, ambient harmonic analysis, or Kähler geometry, those structures remain outside the bridge-status surface. □

Lean4 ✓ `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket bridge status surface`

Closure status for NameCert$_{\text{HodgeBridge}^{\uparrow}}$**.**

Constructive story: A HodgeBridge packet is built from a finite BHist source packet, the displayed DeRham$^{\uparrow}$ and Cohomology$^{\uparrow}$ dependency rows, a harmonic-projector classifier row, a shared projector readback row, and the Cont, hsame, and Pkg ledger rows that carry those dependencies through the certificate. The source accepts only this finite packet surface; analytic Hodge decomposition, ambient harmonic analysis, and Kähler geometry remain outside the carrier. The classifier transports the finite source packet and projector readback through the named dependency rows and the displayed history-sameness and continuation ledgers. The public certificate handoff is therefore the visible packet assembled by the chapter, with no bridgeChecked row or ambient Hodge-theoretic object introduced.

Theory closure: TheoryGate(NameCert$_{\text{HodgeBridge}^{\uparrow}}$, publicClosed)

Scope closed: The public certificate handoff exports exactly the finite BHist packet, DeRham$^{\uparrow}$ and Cohomology$^{\uparrow}$ dependency rows, harmonic-projector classifier, shared projector readback, Cont, hsame, and Pkg ledger rows carried by this chapter.

Formal verification: FormalStatus(NameCert$_{\text{HodgeBridge}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.HodgeBridgeUp.HodgeBridgeBHistSourcePacket\_public\_certificate\_handoff`

Bridge status: none

Not claimed: Ambient Hodge theory, Kähler geometry, analytic harmonic analysis, and bridgeChecked status are not closed.

Upgrade path: The next step is bridgeChecked coverage for the finite HodgeBridge packet surface and its displayed dependency rows.

274.2 The certificate

Chapter 275

A Concrete Naming Certificate for $\mathsf{MirrorSymmetry}^{\uparrow}$

The mirror-symmetry interface $\mathsf{MirrorSymmetry}^{\uparrow}$ packages, for a Calabi-Yau pair $(X, X^{\vee})$, the conjectural equivalence of the A-model on one side with the B-model on the other (Hodge numbers, Fukaya category vs derived category). It depends on $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ (chapter 138) and $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ (chapter 160), and bridges symplectic and complex-algebraic certificates through a common categorical classifier.

275.1 Carrier definition

Definition 275.1 (Mirror-symmetry paired history carrier)**.** *Fix a* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *A-model source row and a* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ *B-model source row. A* $\mathsf{MirrorSymmetry}^{\uparrow}$ *pair carrier row is the finite* BHist *ledger that records the two source certificates, the visible A-model categorical answer rows, the visible B-model categorical answer rows, and the* Cont *witnesses that align each displayed A-row with the corresponding displayed B-row inside one observed pair packet.*

The carrier is a pairing surface for finite categorical observations. It does not include a full Fukaya category, a derived-equivalence theorem, Hodge-number symmetry beyond the observed rows, or a host Calabi-Yau moduli statement.

Definition 275.2 (Mirror-symmetry categorical classifier)**.** *Two rows of Definition 275.1 are classified as the same* $\mathsf{MirrorSymmetry}^{\uparrow}$ *packet exactly when their A-model source rows are transported by* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$*, their B-model source rows are transported by* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$*, and each paired categorical answer row is matched through* hsame*,* Cont*, and* Pkg *witnesses.*

The classifier is generated only by source transport, categorical answer transport, package transport for the displayed finite ledger, and concatenation of observed pair rows. It contains no generator asserting an external mirror equivalence.

275.2 The certificate

Theorem 275.3 (Mirror-symmetry source certificate scope)**.** *Every* $\mathsf{MirrorSymmetry}^{\uparrow}$ *carrier row has exactly one* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *A-model source and one* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ *B-model source, and every other accepted field is a finite categorical answer history or a transport witness named by* hsame*,* Cont*, or* Pkg*.*

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier source certificate scope`

Proof. Invert the carrier definition. The first source field is the symplectic certificate, the second source field is the derived-category certificate, and the remaining fields are indexed by the finite observed pairing ledger.

The classifier of Definition 275.2 can replace source representatives only through the two named NameCert classifiers. It can replace answer rows only through the displayed hsame, Cont, and Pkg witnesses, so no additional source or host category is introduced. □

Theorem 275.4 (Mirror-symmetry categorical stability)**.** *Classified* $\mathsf{MirrorSymmetry}^{\uparrow}$ *packets remain classified after replacing either source certificate by a NameCert-classified representative and after transporting any displayed categorical answer row through* hsame*,* Cont*, or* Pkg*.*

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryCategorical stability`

Proof. Transport the A-model and B-model sources first. The source certificates give the two NameCert-level comparison rows, and the carrier repacks the transported representatives as the same finite pair packet.

For each displayed categorical answer, use the stored hsame row to replace endpoints, the Cont row to keep the answer in the paired ledger, and the Pkg row to preserve the package reading. Concatenating the finite list of transported pair rows gives the required classifier witness. □

Theorem 275.5 (Mirror-symmetry pairing obligation package)**.** *The* $\mathsf{MirrorSymmetry}^{\uparrow}$ *chapter exports a finite pairing obligation package consisting of the paired* BHist *carrier, the categorical classifier, the two source* NameCert *rows, and the statement that the ledger hides only the observed A-model and B-model matching rows.*

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier semantic name certificate`
Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryCategoricalClassifier endpoint confluence`

Proof. Start from a carrier row and a classifier witness. The carrier supplies the two source certificates and the finite list of paired categorical answer rows; the classifier supplies only the source transports and answer transports allowed in Definition 275.2.

Ledger exactness follows by induction over the finite observed pair list. Each item is either an A-model answer, a B-model answer, or one of the named transport witnesses. Therefore the obligation package records a BEDC pairing surface and does not assert a full external mirror-equivalence theorem. □

Theorem 275.6 (Mirror-symmetry HMS source boundary)**.** *The only external source rows available to a* $\mathsf{MirrorSymmetry}^{\uparrow}$ *packet are* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$*. Every A-model or B-model answer exported by the packet is a finite* BHist *row transported by the displayed* hsame*,* Cont*, and* Pkg *witnesses.*

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier hms source boundary`

Proof. Project the source certificate scope of Definition 275.3. It exposes one symplectic source certificate and one derived-category source certificate.

The classifier of Definition 275.2 then lists the only movements allowed after those source rows are chosen. Source representatives move by their two NameCert classifiers, while answer rows move only by hsame, Cont, and Pkg.

Thus the HMS-facing boundary is a finite BEDC ledger of observed categorical answer rows. It does not add a derived equivalence, a complete Fukaya category, a Hodge theorem, or a standard bridge. □

Theorem 275.7 (Mirror-symmetry local pairing ledger exactness)**.** *The local pairing ledger of a* $\mathsf{MirrorSymmetry}^{\uparrow}$ *packet exposes exactly the two source certificates, the paired A-model and B-model categorical answer rows, the source-transport rows, the answer-transport rows, and the package provenance row. No other row is readable from the packet.*

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier local pairing ledger exactness`

Proof. Unpack the paired history carrier of Definition 275.1. Its fields are the two source certificates, the visible paired categorical answers, and the continuation witnesses aligning each displayed pair.

Next unpack the categorical classifier of Definition 275.2. It contributes only source transport, answer transport, package transport, and concatenation of the observed finite pair rows.

Combining these projections gives the exact ledger. Since every exported item is one of those fields or one of those transport witnesses, there is no hidden row for global mirror equivalence, category completeness, or host Hodge theory. □

Theorem 275.8 (Mirror-symmetry NameCert obligation inventory)**.** *The finite* $\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$ *obligation inventory consists of the paired carrier, the categorical classifier, the HMS source boundary, the categorical stability row, and the local pairing ledger exactness row. These rows give the complete obligation package for the displayed finite pairing surface.*

Proof. Start with the carrier and classifier definitions. They provide the finite packet to be named and the admissible comparison rows between packets.

Apply Definition 275.6 to fix the only external source rows. Apply Definition 275.4 to keep source and answer replacement inside the displayed classifier. Apply Definition 275.7 to identify the full readable ledger.

Together these rows are exactly the obligation inventory asserted by Definition 275.5. The inventory is a finite BEDC pairing surface and leaves derived equivalence, Fukaya category completeness, Hodge theory, and standard bridge data unclaimed. □

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier namecert obligation inventory`

Theorem 275.9 (Mirror-symmetry scoped kernel surface)**.** *The carrier, categorical classifier, HMS source boundary, stability row, and local pairing ledger of* $\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$ *are scoped entirely by the two source* NameCert *rows, the finite* BHist *paired answer rows,* hsame*,* Cont*, and* Pkg*.*

This scoped surface contains no full Fukaya category, derived equivalence, Hodge theorem, Calabi-Yau moduli datum, or standard bridge.

Proof. Start with Definition 275.3. It fixes the only external sources as the symplectic and derived-category NameCert rows, and it leaves every remaining accepted field inside the finite observed pairing ledger.

Apply Definition 275.4. The only admissible movements of source and answer rows are the displayed NameCert transports and the hsame, Cont, and Pkg witnesses, so the classifier remains within the same kernel surface.

Next use Definition 275.7. It reads the local ledger as exactly the two source certificates, paired categorical answer rows, source transports, answer transports, and package provenance.

Finally apply Definition 275.8. The inventory contains precisely the carrier, categorical classifier, HMS source boundary, stability row, and local ledger exactness row. Hence the scoped kernel surface is the listed finite BEDC packet and excludes full Fukaya categories, derived equivalence, Hodge theory, Calabi-Yau moduli, and standard bridges. □

Lean4 ✓ `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier scoped kernel scope`

Theorem 275.10 (Mirror-symmetry finite HMS bridge boundary)**.** *The public HMS source boundary and the local pairing ledger define a paper bridge from the finite* $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$ *source rows to the abstract* $\mathsf{MirrorSymmetry}^{\uparrow}$ *pairing schema. Every exported A-model or B-model answer row factors through the displayed* BHist *paired ledger,* hsame*,* Cont*,* Pkg*, and the two source certificates.*

This bridge boundary does not claim a full Fukaya category, derived equivalence, Hodge-number theorem, Calabi-Yau moduli theorem, or machine-checked standard bridge.

Proof. Begin with Definition 275.6. It restricts the available source data to one symplectic certificate, one derived-category certificate, and finite A-model and B-model answer rows transported by the displayed hsame, Cont, and Pkg witnesses.

Apply Definition 275.7. The ledger therefore exposes exactly the two source certificates, paired categorical answer rows, source transports, answer transports, and package provenance.

Use Definition 275.8 to identify these rows as the local $\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$ obligation package, and then apply Definition 275.9 to keep every consumer read inside the same finite kernel surface.

Repacking those four displayed rows gives the paper bridge from the two finite source certificates to the $\mathsf{MirrorSymmetry}^{\uparrow}$ pairing schema. Since no cited row contains a full Fukaya category, derived equivalence, Hodge-number theorem, Calabi-Yau moduli theorem, or machine-checked standard bridge, the boundary exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope is the finite paired BHist ledger over $\mathsf{NameCert}_{\mathsf{Symplectic}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DerivedCat}^{\uparrow}}$, visible A-model and B-model categorical rows, hsame, Cont, Pkg, and the displayed source certificate rows.

Constructive story: A $\mathsf{MirrorSymmetry}^{\uparrow}$ packet is built from one displayed symplectic source certificate, one displayed derived-category source certificate, and a finite ledger of paired categorical answer rows whose admissible movements are exactly the source-certificate, history, continuation, and package transports named by the classifier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MirrorSymmetry}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MirrorSymmetryUp.MirrorSymmetryPairCarrier\_hms\_source\_boundary`

Bridge status: paperBridge

Not claimed: This chapter does not claim a full Fukaya category, derived equivalence, Hodge-number theorem, Calabi-Yau moduli theorem, or standard bridge.

Upgrade path: The finite A-side and B-side source rows and displayed bridge packet are the mature paper surface, while machine bridge checking remains governed by the paperBridge bridge status.

Chapter 276

A Concrete Naming Certificate for $\mathsf{AxisNat}^{\uparrow}$

This chapter gives a concrete naming-certificate construction for $\mathsf{AxisNat}^{\uparrow}$ at the paper-proof level. The interface $\mathsf{AxisNat}^{\uparrow}$ is a conservative parallel to $\mathsf{Nat}^{\uparrow}$ (Chapter 28): it is a finite-repetition counting seed whose carrier is the $\mathsf{E0}$-spine, not the $\mathsf{E1}$-spine that $\mathsf{Nat}^{\uparrow}$ uses. The two interfaces are kernel-distinct; their relation is mediated by the bridge of Chapter 9344.

276.1 The $\mathsf{E0}$-spine carrier

Definition 276.1 (Zero-spine history judgment)**.** *The zero-spine history judgment* $\mathsf{ZeroSpine}(h)$ *is generated by:*

1. $\mathsf{ZeroSpine}(\mathsf{e})$*;*
2. *if* $\mathsf{ZeroSpine}(h)$*, then* $\mathsf{ZeroSpine}(\mathsf{E0}(h))$*.*

No rule generates $\mathsf{ZeroSpine}(\mathsf{E1}(h))$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Spine.ZeroSpine`

Lemma 276.2 (Zero-spine empty base)**.** $\mathsf{ZeroSpine}(\mathsf{e})$ *holds.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Spine.zeroSpine empty`

Proof. First generator of the zero-spine judgment. □

Lemma 276.3 (Zero-spine closure under zero-extension)**.** *If* $\mathsf{ZeroSpine}(h)$*, then* $\mathsf{ZeroSpine}(\mathsf{E0}(h))$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Spine.zeroSpine e0 closed`

Proof. Second generator of the zero-spine judgment. □

Lemma 276.4 (Zero-spine zero-extension inversion)**.** *If* $\mathsf{ZeroSpine}(\mathsf{E0}(h))$*, then* $\mathsf{ZeroSpine}(h)$*.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Spine.zeroSpine e0 inversion`

Proof. Proceed by cases on the closed generation of $\mathsf{ZeroSpine}(\mathsf{E0}(h))$.

The empty constructor cannot produce an $\mathsf{E0}$ endpoint. The zero-extension constructor is the only matching generated shape, and it exposes exactly the predecessor witness $\mathsf{ZeroSpine}(h)$. □

Lemma 276.5 (Zero-spine no one-extension)**.** *No judgment* ZeroSpine(E1(h)) *is generated.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Spine.zeroSpine no e1 extension`

Proof. By closed generation for ZeroSpine, the only extension rule uses E0. No E1-extension constructor is displayed. □

Lemma 276.6 (Zero-spine hsame transport)**.** *If* ZeroSpine(h) *and* hsame(h, k)*, then* ZeroSpine(k)*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Spine.zeroSpine hsame transport`

Proof. hsame is propositional history identity, so the witness transports along the equality. □

276.2 The certificate

Definition 276.7 (Zero-spine source specification)**.** *The source specification* AxisNatSource *consists of histories h with* ZeroSpine(h)*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNatSourceSpec`

Definition 276.8 (Zero-spine pattern specification)**.** *The pattern specification* AxisNatPattern *is repeated generation by the zero-extension constructor* E0 *from* e*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNatPatternSpec`

Definition 276.9 (Zero-spine classifier specification)**.** *The classifier specification* AxisNatClassifier *classifies zero-spine histories pairwise by* hsame *restricted to the zero-spine carrier.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNatClassifierSpec`

Definition 276.10 (Zero-spine stability certificate)**.** *The stability certificate* AxisNatStability *consists of:*

1. *the base derivation* ZeroSpine(e)*;*
2. *closure under* E0*;*
3. *no one-extension, recording that* E1 *is not part of the zero-spine generation discipline.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNatStabilityCert`
↪ variant `BEDC.Derived.AxisZeckendorf.AxisNat.axisNatStabilityCert`

Definition 276.11 (Zero-spine ledger policy)**.** *The ledger policy* AxisNatLedger *records the source zero-spine history behind any visible zero-spine class. It does not erase the emitted history; it licenses a derived interface while retaining provenance.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNatLedgerPolicy`

Theorem 276.12 (Concrete $\mathsf{AxisNat}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}} := (\mathsf{AxisNatSource}, \mathsf{AxisNatPattern}, \mathsf{AxisNatClassifier}, \mathsf{AxisNatStability}, \mathsf{AxisNatLedger})$$

forms a valid naming certificate for the derived interface $\mathsf{AxisNat}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisNat.axisNat namecert`

Proof. The source, pattern, classifier, stability, and ledger fields are explicitly supplied by the preceding definitions. The stability field is justified by zero-spine base generation, zero-spine closure under $\mathsf{E0}$, and zero-spine no-one-extension. Therefore the naming protocol is satisfied. □

Corollary 276.13 ($\mathsf{AxisNat}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{AxisNat}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$*. It is not a primitive object of the finite kernel.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNat semantic name certificate`

276.3 Distinction from $\mathsf{Nat}^{\uparrow}$

Theorem 276.14 (Zero-spine and unary spine are kernel-distinct)**.** *The* $\mathsf{E0}$*-spine carrier* $\mathsf{ZeroSpine}$ *and the* $\mathsf{E1}$*-spine carrier* Unary *intersect, as history predicates, only at the empty history* e*.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisNat.ZeroSpine unaryHistory intersection empty`

Proof. A zero-spine history is either e or has the form $\mathsf{E0}(h)$; a unary history is either e or has the form $\mathsf{E1}(h)$. The two non-empty shapes are $\mathsf{E0}$-headed and $\mathsf{E1}$-headed respectively, so a non-empty history cannot satisfy both judgments. □

Remark 276.15 (Bridge instead of identification)**.** *The kernel-distinct theorem above ensures that any identification of* $\mathsf{AxisNat}^{\uparrow}$ *with* $\mathsf{Nat}^{\uparrow}$ *must come from a separately certified bridge, not from internal kernel reasoning. Such a bridge is the subject of Chapter 9344.*

Theorem 276.16 (Zero-spine source shape exhaustion)**.** *For every history* $h : \mathsf{BHist}$*, if* $\mathsf{ZeroSpine}(h)$*, then*

$$\mathsf{hsame}(h, \mathsf{e}) \vee \exists h_{\mathrm{pred}} : \mathsf{BHist}.\ \mathsf{hsame}(h, \mathsf{E0}(h_{\mathrm{pred}})) \wedge \mathsf{ZeroSpine}(h_{\mathrm{pred}}).$$

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisNat.zeroSpine source shape exhaustion`

Proof. Induct on the displayed generation of $\mathsf{ZeroSpine}(h)$ from Definition 276.1.

In the empty-generator case, h is e. The left disjunct follows from the empty generator for $\mathsf{hsame}(\mathsf{e}, \mathsf{e})$.

In the zero-extension generator case, the derivation has the displayed form $h = \mathsf{E0}(p)$ together with a predecessor derivation $\mathsf{ZeroSpine}(p)$. Choose $h_{\mathrm{pred}} := p$ in the right disjunct. The history-sameness component is $\mathsf{hsame}(\mathsf{E0}(p), \mathsf{E0}(p))$, obtained by applying the $\mathsf{E0}$ history-sameness constructor to reflexivity on p, and the remaining component is exactly the predecessor derivation $\mathsf{ZeroSpine}(p)$. □

Theorem 276.17 (AxisNat unary bridge consumer example)**.** *A package-level consumer of the* $\mathsf{AxisNat}^{\uparrow}$*-to-*$\mathsf{Nat}^{\uparrow}$ *bridge reads only the zero-spine source, the unary-direction bridge boundary, the* $\mathsf{Nat}^{\uparrow}$ *comparison row, the displayed* hsame *transports, the* Cont *provenance, and the local* $\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *rows. The consumer cannot identify the two spines internally except at the empty history* e*.*

Proof. Start from the zero-spine source and Definition 276.16. This gives the consumer only the empty history case or a displayed $\mathsf{E0}$ predecessor case, with no access to a unary constructor inside the $\mathsf{AxisNat}^{\uparrow}$ carrier.

Next use Definition 276.14. It states that the zero-spine and unary-spine history predicates meet only at e, so any comparison with $\mathsf{Nat}^{\uparrow}$ must pass through the separately displayed unary-direction bridge boundary.

The bridge consumer then reads the $\mathsf{Nat}^{\uparrow}$ comparison row, the displayed hsame transports, the Cont provenance, and the two local naming surfaces. Since the bridge boundary is the only route between the spines, no internal identification of $\mathsf{AxisNat}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ is produced beyond the empty-history overlap. □

Lean4 ✓ `BEDC.Derived.AxisNatUp.AxisNatTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$.

Constructive story: A zero-axis natural number begins at e and grows only by repeated E0-extension: e is admitted, E0(h) is admitted whenever h is already admitted, and the first E1-headed history is excluded from the carrier. The classifier is hsame restricted to this zero-spine carrier, so two accepted histories are the same only as histories on the E0-spine, while the source specification covers exactly zero-spine histories and the pattern specification is their closed generation from e by E0. The ledger keeps the raw zero-spine history behind every visible $\mathsf{AxisNat}^{\uparrow}$ class, preserving the fact that this carrier is kernel-distinct from the E1-spine used by $\mathsf{Nat}^{\uparrow}$. The five name-certificate obligations are the same closed-generation facts in certificate form: e inhabits the carrier, restricted hsame is reflexive, symmetric, and transitive, and if an admitted zero-spine history is transported along hsame then the transported history is admitted as well. A history h lies in the signature gap of an $\mathsf{AxisNat}^{\uparrow}$ token p when the source-domain accepts h as a zero-spine history, the signature relation pairs it with an answer history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is a finite stack of E0 marks whose zero-depth matches the spine encoded by p. The only bridge mentioned here is negative: any comparison with $\mathsf{Nat}^{\uparrow}$ must pass through a separate bridge, because the two spines meet internally only at e. The chapter registers this zero-spine counting construction under the name $\mathsf{AxisNat}^{\uparrow}$; it adds no arithmetic induction, recursion principle, RealUp completion, infinite limit object, or new primitive beyond empty history and E0-stacking.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation inventory binds the zero-spine constructor, source, pattern, classifier, stability, ledger, semantic $\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$, the source-exhaustion rows of Definition 276.16, the NatUp-separation rows of Definition 276.14, the unary bridge consumer example of Definition 276.17, and the paper bridge boundary of chapter 9344.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisNat}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AxisZeckendorf.AxisNat.AxisNat\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: It does not identify AxisNatUp with NatUp, arithmetic induction, recursion, RealUp completeness, or an infinite limit object.

Upgrade path: No further paper-side theory grade remains for this chapter.

Chapter 277

A Concrete Naming Certificate for $\mathsf{AxisAdd}^{\uparrow}$

The additive seed on the $\mathsf{E0}$-spine is licensed by continuation of zero-spine histories. It is not a primitive operation. The zero-spine obligation surface for $\mathsf{AxisAdd}^{\uparrow}$ is the source, pattern, classifier, stability, ledger, and checked algebra rows recorded below.

277.1 Continuation on the zero-spine

Definition 277.1 (Additive source specification (zero-spine))**.** *The source specification* AxisAddSource *consists of ordered pairs* (h, k) *of zero-spine histories.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddSourceSpec`

Definition 277.2 (Additive pattern specification (zero-spine))**.** *The pattern specification* AxisAddPattern *is the continuation predicate*

$$\mathsf{Cont}(h, k, r)$$

together with $\mathsf{ZeroSpine}(h)$ *and* $\mathsf{ZeroSpine}(k)$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddPatternSpec`

Definition 277.3 (Additive classifier specification (zero-spine))**.** *The classifier specification* AxisAddClassifier *identifies two additive results* r_1 *and* r_2 *of* (h, k) *when both share the same zero-spine pattern witness and* $\mathsf{hsame}(r_1, r_2)$ *holds.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddClassifierSpec`

Theorem 277.4 (Left unit on the zero-spine)**.** *For every zero-spine history* h*, continuation by the empty history on the left produces* h*:*

$$\mathsf{Cont}(\mathsf{e}, h, h).$$

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont left unit zeroSpine`

Proof. This is the kernel left-unit theorem of Cont restricted to a zero-spine input. □

Definition 277.5 (Additive stability certificate (zero-spine))**.** *The stability certificate* AxisAddStability *records the checked zero-spine rows for left unit, result zero-spine closure, determinacy, right unit, associativity, and commutativity.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddStabilityCert`
↪ variant `BEDC.Derived.AxisZeckendorf.AxisAdd.axisAddStabilityCert`

Definition 277.6 (Additive ledger policy (zero-spine))**.** *The ledger policy* AxisAddLedger *records the input pair* (h, k) *and the continuation witness behind any visible additive result.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddLedgerPolicy`

277.2 The certificate

Theorem 277.7 (Concrete $\mathsf{AxisAdd}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{AxisAdd}^{\uparrow}} := (\mathsf{AxisAddSource}, \mathsf{AxisAddPattern}, \mathsf{AxisAddClassifier}, \mathsf{AxisAddStability}, \mathsf{AxisAddLedger})$$

forms a valid naming certificate for the derived additive interface $\mathsf{AxisAdd}^{\uparrow}$ *on the* E0*-spine.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.AxisAdd.axisAdd namecert`

Proof. Source, pattern, classifier, stability, and ledger fields are supplied by the preceding definitions. The stability field includes the checked zero-spine rows for left unit, result closure, determinacy, right unit, associativity, and commutativity. □

Corollary 277.8 ($\mathsf{AxisAdd}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{AxisAdd}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{AxisAdd}^{\uparrow}}$*. It is not a primitive operation of the finite kernel.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAdd semantic name certificate`

Theorem 277.9 (AxisAdd dependency on the AxisNat zero-spine)**.** *The* $\mathsf{AxisAdd}^{\uparrow}$ *source, pattern, classifier, and stability rows are read over the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine carrier of chapter 276 before they become an additive continuation certificate. Thus an accepted* $\mathsf{AxisAdd}^{\uparrow}$ *certificate is a consumer of the AxisNat zero-spine source surface, followed by the continuation row* $\mathsf{Cont}(h, k, r)$ *and the checked additive rows for determinacy, unit laws, associativity, and commutativity.*

Lean4 ✓ `BEDC.Derived.AxisAddUp.AxisAddTasteGate single carrier alignment`

Proof. Begin with the source and pattern specifications. They require the ordered input pair (h, k) to satisfy the zero-spine carrier condition before the continuation witness $\mathsf{Cont}(h, k, r)$ can serve as an additive pattern.

Apply zero-spine preservation for continuation. This keeps the result row on the same AxisNat carrier surface, so the additive classifier is still read over the zero-spine source rather than over an ambient arithmetic carrier.

Use result determinacy to identify any two displayed continuation results over the same zero-spine source pair. Then apply the left-unit, right-unit, associativity, and commutativity rows to package the stable additive law surface.

The resulting certificate therefore depends on the sibling AxisNat zero-spine chapter for its carrier and source discipline, and only then exposes the $\mathsf{AxisAdd}^{\uparrow}$ continuation certificate. □

Lean4 ✓ `BEDC.Derived.AxisAddUp.AxisAddUp single carrier alignment`

277.3 Horizon obligations

The following theorems record the checked zero-spine law surface.

Theorem 277.10 (Continuation preserves zero-spine)**.** *If* $\mathsf{ZeroSpine}(h)$*,* $\mathsf{ZeroSpine}(k)$*, and* $\mathsf{Cont}(h, k, r)$*, then* $\mathsf{ZeroSpine}(r)$*.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont result zeroSpine`
↪ variant `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAdd cont result zeroSpine`

Theorem 277.11 (Result determinacy on the zero-spine)**.** *If* $\mathsf{ZeroSpine}(h)$*,* $\mathsf{ZeroSpine}(k)$*,* $\mathsf{Cont}(h, k, r)$*, and* $\mathsf{Cont}(h, k, r')$*, then* $\mathsf{hsame}(r, r')$*. Equivalently, two displayed* AxisAddPattern *witnesses over the same ordered source* (h, k) *determine an* AxisAddClassifier *witness between their results.*

Proof. Apply the finite-kernel continuation determinacy theorem to the two continuation witnesses over the same ordered source. It yields $\mathsf{hsame}(r, r')$ for the displayed results.

Unfold the AxisAddPattern and AxisAddClassifier specifications. The zero-spine source witnesses and the two continuation witnesses supply the pattern fields, and the determinacy result supplies the classifier equality row. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont result determinacy`

Theorem 277.12 (Right unit on the zero-spine)**.** *For every zero-spine history* h*,* $\mathsf{Cont}(h, \mathsf{e}, h)$*.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont right unit zeroSpine`

Theorem 277.13 (Associativity on the zero-spine)**.** $\mathsf{AxisAdd}^{\uparrow}$ *is associative up to* hsame *on zero-spine histories.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont associative zeroSpine`

Theorem 277.14 (Commutativity on the zero-spine)**.** $\mathsf{AxisAdd}^{\uparrow}$ *is commutative up to* hsame *on zero-spine histories.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont commutative zeroSpine`

Remark 277.15 (Distinct from $\mathsf{Add}^{\uparrow}$)**.** $\mathsf{AxisAdd}^{\uparrow}$ *and* $\mathsf{Add}^{\uparrow}$ *act on different carriers: the first uses* ZeroSpine*, the second uses* Unary*. The two are kernel-distinct, by Theorem 276.14; identification across the two carriers requires the bridge of Chapter 9344.*

Theorem 277.16 (Zero-spine result inversion for $\mathsf{AxisAdd}^{\uparrow}$)**.** *For histories* $h, k, r : \mathsf{BHist}$*, if* $\mathsf{Cont}(h, k, r)$ *and* $\mathsf{ZeroSpine}(r)$*, then*

$$\mathsf{ZeroSpine}(h) \wedge \mathsf{ZeroSpine}(k).$$

Thus a displayed $\mathsf{AxisAdd}^{\uparrow}$ *continuation whose result lies on the* E0*-spine has both its left prefix and right suffix on the same spine.*

Proof. Induct on the displayed derivation of $\mathsf{Cont}(h, k, r)$.

In the empty-tail generator, the continuation is $\mathsf{Cont}(h, \mathsf{e}, h)$ and the result is h. The assumed result witness is exactly $\mathsf{ZeroSpine}(h)$, while $\mathsf{ZeroSpine}(\mathsf{e})$ is supplied by Definition 276.2. Thus $\mathsf{ZeroSpine}(h) \wedge \mathsf{ZeroSpine}(\mathsf{e})$ holds.

In the zero-tail step, the derivation has predecessor $\mathsf{Cont}(h, k_0, r_0)$ and conclusion $\mathsf{Cont}(h, \mathsf{E0}(k_0), \mathsf{E0}(r_0))$. The result assumption is $\mathsf{ZeroSpine}(\mathsf{E0}(r_0))$. By Definition 276.4, obtain $\mathsf{ZeroSpine}(r_0)$. The induction hypothesis applied to the predecessor continuation gives $\mathsf{ZeroSpine}(h) \wedge \mathsf{ZeroSpine}(k_0)$. Applying Definition 276.3 to the second component gives $\mathsf{ZeroSpine}(\mathsf{E0}(k_0))$. Hence the step conclusion is $\mathsf{ZeroSpine}(h) \wedge \mathsf{ZeroSpine}(\mathsf{E0}(k_0))$.

In the one-tail step, the derivation has result $\mathsf{E1}(r_0)$, so the result assumption is $\mathsf{ZeroSpine}(\mathsf{E1}(r_0))$. This contradicts Definition 276.5. Therefore no one-tail continuation can have a zero-spine result, and the implication holds in that branch. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont zeroSpine result inversion`

Corollary 277.17 (Zero-spine source/result equivalence for $\mathsf{AxisAdd}^{\uparrow}$)**.** *For histories $h, k, r : \mathsf{BHist}$ with $\mathsf{Cont}(h, k, r)$,*

$$\mathsf{ZeroSpine}(r) \iff \mathsf{ZeroSpine}(h) \wedge \mathsf{ZeroSpine}(k).$$

Proof. The forward direction is Definition 277.16. The reverse direction is Definition 277.10 applied to the displayed continuation witness. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddCont zeroSpine result sources iff`

Theorem 277.18 (AxisAdd zero-spine standard bridge boundary)**.** *The public zero-spine result inversion and source/result equivalence identify the standard bridge boundary for the* $\mathsf{AxisAdd}^{\uparrow}$ *zero-spine addition surface. The boundary is over* ZeroSpine*,* Cont*,* hsame*, and the displayed ledger for the two source histories, the continuation witness, and the result history. It does not identify this surface with full* $\mathsf{Add}^{\uparrow}$ *and does not add arithmetic outside the zero-spine carrier.*

Proof. The result-inversion theorem reads a displayed $\mathsf{Cont}(h, k, r)$ witness from the result side: when r lies on the zero spine, it recovers the two zero-spine source witnesses. The source/result equivalence supplies the converse direction over the same continuation witness.

Together these two public rows give the bridge boundary. A consumer may move between the zero-spine source pair and the zero-spine result exactly through the displayed Cont ledger and the endpoint classifier hsame.

The carrier is still ZeroSpine rather than the ordinary $\mathsf{Add}^{\uparrow}$ carrier. Hence the boundary exports only the zero-spine surface and leaves full addition identification outside the claim. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAddUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{AxisAdd}^{\uparrow}}$**.**

Constructive story: Zero-axis addition is built by taking two already admitted zero-spine histories and continuing the first by the second with the kernel continuation relation Cont. The carrier for a fixed input pair accepts exactly those result histories displayed by a $\mathsf{Cont}(h, k, r)$ witness with both h and k on the E0-spine; results lacking the two zero-spine inputs or the continuation witness are outside the surface. Two accepted results are classified together when they sit over the same zero-spine input pair and are related by hsame, so the source specification is the ordered zero-spine pair and the pattern specification is the displayed continuation result. The ledger keeps the raw inputs and the continuation witness visible behind the public result, which lets the checked rows state left and right units at e, result closure on the zero-spine, determinacy, associativity, and commutativity without turning this operation into the ordinary $\mathsf{Add}^{\uparrow}$ carrier. For the five name-certificate obligations, the canonical continuation result inhabits the carrier, endpoint hsame gives reflexivity, symmetry, and transitivity, and determinacy plus transport keep any recognized result inside the same accepted pattern. A history r lies in the signature gap of an $\mathsf{AxisAdd}^{\uparrow}$ token p when the source-domain accepts its ordered zero-spine inputs, the signature relation pairs the displayed continuation result with an answer result and evidence ledger, and the token-introduction relation names that answer as p; concretely, r is the Cont sum of the two zero-spine histories encoded by p, up to hsame. The chapter registers this continuation-on-zero-spines construction under the name $\mathsf{AxisAdd}^{\uparrow}$; it introduces no full $\mathsf{Add}^{\uparrow}$ identification, public arithmetic beyond the zero-spine, or independent addition primitive.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisAdd}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The closed scope is the zero-spine source and pattern over ZeroSpine, continuation witnesses Cont, endpoint classifier hsame, the AxisAdd$^{\uparrow}$ NameCert tuple, the displayed ledger policy, the public zero-spine result inversion surface, and Definition 277.18.

Formal verification: FormalStatus(NameCert$_{\mathsf{AxisAdd}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.AxisZeckendorf.AxisAdd.AxisAdd\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Not claimed: It does not claim full Add$^{\uparrow}$ identification or public arithmetic beyond the zero-spine.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 278

A Concrete Naming Certificate for ZeckendorfNormal$^{\uparrow}$

This chapter introduces a Zeckendorf-style normal-form classifier on histories: no two adjacent E1-extensions. The classifier is structural; it does not depend on hsame and does not assert any arithmetic interpretation. The Fibonacci value-preservation reading is a horizon target queued for the formalize pipeline. The carry-classifier chapter of chapter 9345 consumes this chapter's normal-form judgment, the displayed `word_011` rejection row, and the displayed `word_100` admission row as its source/target normality surface.

278.1 The normal-form predicate

Definition 278.1 (Zeckendorf normal-form judgment)**.** *The Zeckendorf normal-form judgment* ZNormal(h) *is generated by:*

1. ZNormal(e)*;*
2. *if* ZNormal(h)*, then* ZNormal(E0(h))*;*
3. ZNormal(E1(e))*;*
4. *if* ZNormal(E0(h))*, then* ZNormal(E1(E0(h))).

No rule attaches E1 *directly on top of another* E1*-extension.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal`

Definition 278.2 (Display words `011` and `100`)**.** *We fix two reference histories used throughout this chapter and Chapter 9345:*

$$\texttt{word_011} := \mathsf{E1}(\mathsf{E1}(\mathsf{E0}(\mathsf{e}))), \qquad \texttt{word_100} := \mathsf{E0}(\mathsf{E0}(\mathsf{E1}(\mathsf{e}))).$$

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.word_011`
↪ variant `BEDC.Derived.AxisZeckendorf.Zeckendorf.word_100`

Theorem 278.3 (`word_100` is normal)**.** ZNormal(`word_100`) *holds.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.znormal_word_100`

Proof. By two applications of the E0-closure rule on top of the E1-after-empty base. □

Theorem 278.4 (`word_011` is not normal)**.** ZNormal(`word_011`) *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.znormal word 011 absurd`

Proof. By cases on a hypothetical derivation of ZNormal(E1(E1(E0(e)))). The only E1-headed generators require either an empty argument or an E0-headed argument; the actual argument is E1(E0(e)), which has neither shape. So no rule generates the judgment. □

278.2 The certificate

Definition 278.5 (Normal-form source specification)**.** *The source specification* ZNormalSource *consists of histories h with* ZNormal(h)*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormalSourceSpec`

Definition 278.6 (Normal-form pattern specification)**.** *The pattern specification* ZNormalPattern *is structural normal-form generation:* E0*-extensions of any normal-form history, and* E1*-extensions only above* e *or above an* E0*-extension.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormalPatternSpec`

Definition 278.7 (Normal-form classifier specification)**.** *The classifier specification* ZNormalClassifier *classifies normal-form histories pairwise by* hsame *restricted to the normal-form carrier.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormalClassifierSpec`

Definition 278.8 (Normal-form stability certificate)**.** *The stability certificate* ZNormalStability *consists of:*

1. *the base derivation* ZNormal(e)*;*
2. *closure under* E0*;*
3. *the* E1*-after-empty base,* ZNormal(E1(e))*;*
4. *closure under* E1 *above an* E0*-extension.*

The exclusion of E1*-on-*E1 *is enforced by closed generation.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormalStabilityCert`
↪ variant `BEDC.Derived.AxisZeckendorf.Zeckendorf.zNormalStabilityCert`

Definition 278.9 (Normal-form ledger policy)**.** *The ledger policy* ZNormalLedger *records the source normal-form history behind any visible normal-form class.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormalLedgerPolicy`

Theorem 278.10 (Concrete $\mathsf{ZeckendorfNormal}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}} := (\mathsf{ZNormalSource}, \mathsf{ZNormalPattern}, \mathsf{ZNormalClassifier}, \mathsf{ZNormalStability}, \mathsf{ZNormalLedger})$$

forms a valid naming certificate for the derived interface $\mathsf{ZeckendorfNormal}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.Zeckendorf.zNormal namecert`

Proof. The five fields are supplied by the preceding definitions. The stability field is justified by closed generation of the ZNormal judgment together with the no-one-on-one exclusion. □

Theorem 278.11 (ZNormal adjacent-one inversion)**.** *If a derivation of* $\mathsf{ZNormal}(\mathsf{E1}(h))$ *is displayed, then either* $h = \mathsf{e}$*, or there is a history* k *such that* $h = \mathsf{E0}(k)$ *and* $\mathsf{ZNormal}(\mathsf{E0}(k))$*. In particular, no closed derivation of* $\mathsf{ZNormal}(\mathsf{E1}(h))$ *has an* $\mathsf{E1}$*-headed tail.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal adjacent one inversion`

Proof. Proceed by cases on the displayed derivation of $\mathsf{ZNormal}(\mathsf{E1}(h))$. The $\mathsf{E1}$-after-empty constructor gives the left alternative with $h = \mathsf{e}$.

The $\mathsf{E1}$-above-zero constructor has input $\mathsf{ZNormal}(\mathsf{E0}(k))$ and conclusion $\mathsf{ZNormal}(\mathsf{E1}(\mathsf{E0}(k)))$. This is exactly the right alternative, with $h = \mathsf{E0}(k)$.

The empty constructor and the $\mathsf{E0}$-extension constructor have conclusions whose top constructor is not $\mathsf{E1}$. They cannot match the displayed target. Thus the two alternatives exhaust all closed generators, with no arithmetic interpretation used. □

Theorem 278.12 (ZNormal zero-extension inversion)**.** *Every zero-headed normal form descends to a normal tail: if* $\mathsf{ZNormal}(\mathsf{E0}(h))$ *is derived, then* $\mathsf{ZNormal}(h)$ *is derived.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal zero extension inversion`

Proof. Proceed by cases on the derivation of $\mathsf{ZNormal}(\mathsf{E0}(h))$. The only closed generator whose conclusion has top constructor $\mathsf{E0}$ is the $\mathsf{E0}$-extension rule.

That rule is applied to a predecessor derivation of $\mathsf{ZNormal}(h)$. Project this predecessor derivation as the desired normal tail. The remaining constructors conclude at e or at an $\mathsf{E1}$-headed history, so they do not match the zero-headed target. □

Theorem 278.13 (ZNormal normal-form coverage)**.** *The* $\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}$ *source, pattern, classifier, stability, and ledger rows exhaust exactly the closed constructors of* $\mathsf{ZNormal}$*: the empty row, the* $\mathsf{E0}$*-extension row, the* $\mathsf{E1}$*-after-empty row, and the* $\mathsf{E1}$*-above-zero row. No arithmetic value row, Fibonacci interpretation, or extra classifier constructor is part of the covered surface.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal normal form coverage`

Proof. Unpack the certificate of Definition 278.10. Its source is the normal-form carrier of Definition 278.5, and its pattern row is the closed structural generation of Definition 278.6.

The stability row of Definition 278.8 lists the four generators: empty, zero extension, one after empty, and one above a zero-headed normal form. The ledger row of Definition 278.9 records the source normal-form history behind each visible class.

Apply Definition 278.11 to exclude any hidden $\mathsf{E1}$-on-$\mathsf{E1}$ constructor, and apply Definition 278.12 to read zero-headed judgments back to their normal tails. Hence the displayed certificate covers exactly the closed constructors and nothing from an arithmetic interpretation. □

Corollary 278.14 ($\mathsf{ZeckendorfNormal}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}$*. It is not a primitive object of the finite kernel.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal namecert licensed not primitive`

Theorem 278.15 (Zeckendorf normal carry-window determinacy)**.** *For a displayed* $\mathsf{ZNormal}$ *derivation, any finite carry window whose local source word is* `word_011` *is rejected by Definition 278.4, while the corresponding structural target window* `word_100` *is accepted by Definition 278.3. Consequently the Zeckendorf structural carry boundary has a unique normal target window under the displayed* BHist *constructors and the chapter's* hsame *transport. This is only a structural normal-form statement and makes no Fibonacci value-preservation claim.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Carry.ZCarry window determinacy`

Proof. Inspect the source side of the carry window. If it reads `word_011`, then it has an E1-headed tail under another E1-extension. Apply Definition 278.11 and then Definition 278.4; no displayed ZNormal(`word_011`) derivation can be accepted.

Inspect the target side. The word `word_100` is generated by the E1-after-empty base followed by two E0-extensions, so Definition 278.3 supplies the accepted normal target window.

The remaining possible ambiguity is hidden transport along hsame. Apply Definition 278.12 to peel zero-headed target judgments and Definition 278.19 to move accepted normal windows along the chapter's history-sameness comparisons. These operations keep the same displayed BHist constructor shape, so a carry boundary over `word_011` has only the displayed `word_100` normal target.

No arithmetic reading of the two words is used. The argument separates the structural source rejection from the structural target acceptance inside the normal-form classifier alone. □

Theorem 278.16 (ZNormal scoped classifier boundary)**.** *Every accepted* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *classifier row is exactly an* hsame *row between* BHist *witnesses that lie in the closed* ZNormal *source. The only scope inputs are the* BHist *constructors, the closed* ZNormal *generation rules,* hsame *transport, and the* $\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}$ *ledger. No Fibonacci value semantics, arithmetic weight, uniqueness-of-value row, or external number interpretation is read by the classifier boundary.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Zeckendorf.ZNormal semantic name certificate`

Proof. Unfold the source, pattern, classifier, stability, and ledger rows of Definition 278.10. The source row is exactly the closed ZNormal carrier over BHist, the pattern and stability rows are the four closed generators, and the classifier row is hsame restricted to witnesses recorded by the ledger.

For an accepted row, the ledger supplies the two displayed source witnesses and the classifier supplies the hsame comparison between them. Transport along Definition 278.19 keeps the accepted history inside the same closed normal-form source, so the row is an hsame row between BHist witnesses and nothing wider.

For the converse direction, take an hsame row between two displayed ZNormal source witnesses. The classifier specification reads exactly such restricted history-sameness rows, and the ledger records the source witness behind each visible class, so the row is accepted by the $\mathsf{ZeckendorfNormal}^{\uparrow}$ classifier.

The only constructor boundary that can hide an extra case is an E1-headed history. Apply Definition 278.11: an accepted E1 row is either above e or above an E0-headed normal source, and no E1-on-E1 source can be generated. Thus the scoped classifier boundary is exhausted by the displayed primitives and does not use Fibonacci value semantics. □

Theorem 278.17 (ZeckendorfNormal public normal-form boundary)**.** *The public* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *boundary consists exactly of the closed* ZNormal *source over* BHist*, the four constructor rows for empty, zero-extension, one-after-empty, and one-above-zero normal forms, the restricted* hsame *classifier between displayed normal-form witnesses, and the* $\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}$ *ledger of displayed normal-form witnesses. It excludes Fibonacci value preservation, uniqueness of arithmetic value, and any external number interpretation.*

Lean4 ✓ `BEDC.Derived.ZeckendorfNormalUp.ZNormal public normal form boundary`

Proof. Start from the certificate in Definition 278.10. Its source row is the closed ZNormal predicate, its pattern and stability rows are the four displayed normal-form constructors, and its ledger records the normal-form witness behind each visible class. Definition 278.13 shows that the constructor rows cover the visible normal-form surface. The empty, zero-extension, one-after-empty, and one-above-zero cases are therefore the only public source shapes admitted by the certificate.

The classifier boundary is supplied by Definition 278.16. It identifies every accepted classifier row with an hsame comparison between two histories already lying in the closed ZNormal source.

Finally, Definition 278.19 keeps the restricted classifier inside the same source when a displayed history is moved along hsame. Thus the public boundary is exactly the closed source, the four constructor rows, the restricted classifier, and the displayed-witness ledger.

No step reads an arithmetic value. Fibonacci value preservation, uniqueness of arithmetic value, and external number interpretation are outside this normal-form boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}$.

Constructive story: A Zeckendorf normal-form history is generated by four structural moves: admit e, extend any normal history by E0, admit the single E1(e) case, and extend by E1 only when the tail is already E0-headed and normal. The carrier therefore accepts exactly histories made by those rules and excludes any hidden E1-on-E1 step, as witnessed by the rejection of the displayed word `word_011` while the displayed word `word_100` is accepted. The classifier is hsame restricted to accepted normal histories, so the source specification is the closed normal-form predicate, the pattern specification is the same constructor discipline, and no Fibonacci value or arithmetic weight participates in equality. The ledger keeps the raw normal-form history and its visible class aligned, allowing zero-extension inversion, adjacent-one inversion, and carry-window rejection or acceptance to be read directly from the constructor surface. The five name-certificate obligations are immediate from closed generation and restricted hsame: e inhabits the carrier, the classifier is reflexive, symmetric, and transitive on accepted histories, and transported histories remain normal because transport cannot create an unlisted constructor. A history h lies in the signature gap of a $\mathsf{ZNormal}^{\uparrow}$ token p when the source-domain accepts h as structurally normal, the signature relation pairs it with an answer normal history and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is built by the four normal constructors and is hsame to the normal word encoded by p. The chapter registers this structural normal-form construction under the name $\mathsf{ZNormal}^{\uparrow}$; it introduces no Fibonacci value preservation, uniqueness of arithmetic value, external number interpretation, or additional classifier constructor.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The seed scope is the closed BHist generation of ZNormal together with the exclusion of adjacent one-extensions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeckendorfNormal}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.AxisZeckendorf.Zeckendorf.zNormal\_namecert`

Bridge status: none

Not claimed: This chapter does not claim Fibonacci value preservation, uniqueness of arithmetic value, or any arithmetic interpretation of the structural normal form.

Upgrade path: The next public step is a public exported normal-form certificate boundary.

278.3 Horizon: Fibonacci value reading

The classifier above is purely structural. The arithmetic reading—each normal-form history denotes a unique non-negative integer via the Zeckendorf decomposition into Fibonacci numbers—is a horizon target. It must be licensed by a separate $\mathsf{AxisAdd}^{\uparrow}$-compatible bridge before being used in any downstream chapter.

Theorem 278.18 (Zeckendorf value-reading is well-defined, horizon)**.** *There is a function from normal-form histories to non-negative integers that is determined by the recurrence* $W_{n+2} = W_{n+1} + W_n$ *on Fibonacci-indexed positions.*

Lean4 ✓ `BEDC.Derived.ZeckendorfNormalUp.ZNormal fibonacci value horizon`

Theorem 278.19 (ZNormal hsame transport)**.** *If* $\mathsf{ZNormal}(h)$ *and* $\mathsf{hsame}(h, k)$*, then* $\mathsf{ZNormal}(k)$*.*

Lean4 ✓ `BEDC.Derived.ZeckendorfNormalUp.ZNormal hsame transport`

Proof. Proceed by induction on the closed derivation of $\mathsf{ZNormal}(h)$ from Definition 278.1. In each case, invert the displayed hsame comparison by Definition 3.4.

In the empty case, $h = \mathsf{e}$. The comparison $\mathsf{hsame}(\mathsf{e}, k)$ has only the empty aligned branch, so $k = \mathsf{e}$. The empty generator gives $\mathsf{ZNormal}(k)$.

In the zero-extension case, the derivation has predecessor $\mathsf{ZNormal}(h_0)$ and conclusion $\mathsf{ZNormal}(\mathsf{E0}(h_0))$. From $\mathsf{hsame}(\mathsf{E0}(h_0), k)$, constructor inversion gives a history k_0 such that $k = \mathsf{E0}(k_0)$ and $\mathsf{hsame}(h_0, k_0)$. The induction hypothesis applied to the predecessor derivation gives $\mathsf{ZNormal}(k_0)$. Applying the zero-extension generator yields $\mathsf{ZNormal}(\mathsf{E0}(k_0))$, hence $\mathsf{ZNormal}(k)$.

In the one-after-empty case, the derivation concludes $\mathsf{ZNormal}(\mathsf{E1}(\mathsf{e}))$. From $\mathsf{hsame}(\mathsf{E1}(\mathsf{e}), k)$, one-constructor inversion gives a history k_0 such that $k = \mathsf{E1}(k_0)$ and $\mathsf{hsame}(\mathsf{e}, k_0)$. Empty no-confusion forces $k_0 = \mathsf{e}$. The one-after-empty generator therefore gives $\mathsf{ZNormal}(k)$.

In the one-above-zero case, the derivation has predecessor $\mathsf{ZNormal}(\mathsf{E0}(h_0))$ and conclusion $\mathsf{ZNormal}(\mathsf{E1}(\mathsf{E0}(h_0)))$. From $\mathsf{hsame}(\mathsf{E1}(\mathsf{E0}(h_0)), k)$, one-constructor inversion gives a history t such that $k = \mathsf{E1}(t)$ and $\mathsf{hsame}(\mathsf{E0}(h_0), t)$. Zero-constructor inversion then gives a history k_0 such that $t = \mathsf{E0}(k_0)$ and $\mathsf{hsame}(h_0, k_0)$. Apply the induction hypothesis to the predecessor derivation $\mathsf{ZNormal}(\mathsf{E0}(h_0))$ and the comparison $\mathsf{hsame}(\mathsf{E0}(h_0), t)$; this gives $\mathsf{ZNormal}(t)$, hence $\mathsf{ZNormal}(\mathsf{E0}(k_0))$. The one-above-zero generator gives $\mathsf{ZNormal}(\mathsf{E1}(\mathsf{E0}(k_0)))$, which is $\mathsf{ZNormal}(k)$.

These four cases exhaust the closed generators of ZNormal, so the predicate is closed under hsame transport. □

Chapter 279

A Concrete Naming Certificate for $\mathsf{FullAxis}^{\uparrow}$

This chapter records a prefix-thread seal interface over the E0-spine. $\mathsf{FullAxis}^{\uparrow}$ is NOT a metric or Cauchy completion: its boundary token E1(E0(e)) is a finite history, not a limit object, and the chapter explicitly disclaims any reading of the boundary as an infinity. The Cauchy / Real / Topology readings remain in the existing $\mathsf{CompleteMetric}^{\uparrow}$ and related chapters; $\mathsf{FullAxis}^{\uparrow}$ is a structural seal, not a metric completion.

279.1 The prefix thread

Definition 279.1 (Prefix thread)**.** *A prefix thread T is a history-membership predicate T.*member *that contains* e *and is closed under* E0*-extension.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.PrefixThread`

Definition 279.2 (Zero-spine prefix thread)**.** *The zero-spine prefix thread* ZSpineThread *takes T.*member := ZeroSpine*, with empty membership from Lemma 276.2 and* E0*-step from Lemma 276.3.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.zeroSpinePrefixThread`

279.2 Seal certificate

Definition 279.3 (Seal certificate over a prefix thread)**.** *A seal certificate* SealCert(T) *records the structural seal data for a prefix thread T: directedness, monotonicity, prefix-closure, and a ledger policy. The first three fields are skeleton-grade; the ledger field is a horizon obligation queued for the formalize pipeline.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.SealCert`
↪ variant `BEDC.Derived.AxisZeckendorf.FullAxis.zeroSpineSealCert`

279.3 The boundary marker

Definition 279.4 (Boundary marker `boundary_01`)**.** *The boundary marker is the finite history*

$$\texttt{boundary_01} := \mathsf{E1}(\mathsf{E0}(\mathsf{e})).$$

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.boundary 01`

Theorem 279.5 (Boundary marker is not on the zero-spine)**.** ZeroSpine(`boundary_01`) *does not hold.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.boundary 01 not zeroSpine`

Proof. `boundary_01` has top constructor E1; apply zero-spine no-one-extension. □

Remark 279.6 (Boundary marker is finite, not an infinity)**.** *The boundary marker is a finite history of length two. It does NOT denote a limit, an infinity, or a transfinite stage. Any reading that promotes* `boundary_01` *to an infinity is explicitly rejected (cf. Chapter 9350).*

279.4 The certificate

Definition 279.7 (Full-axis source specification)**.** *For a prefix thread T, the source specification* FullAxisSource(T) *consists of histories h with T.*member(h)*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSourceSpec`

Definition 279.8 (Full-axis pattern specification)**.** *For a prefix thread T, the pattern specification* FullAxisPattern(T) *is repeated* E0*-extension within T.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisPatternSpec`

Definition 279.9 (Full-axis classifier specification)**.** *For a prefix thread T, the classifier specification* FullAxisClassifier(T) *classifies pairs of T-members by joint thread membership.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisClassifierSpec`

Definition 279.10 (Full-axis stability certificate)**.** *For a prefix thread T, the stability certificate* FullAxisStability(T) *bundles the seal certificate* SealCert(T) *with the explicit declaration that boundary tokens are finite, not limits.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisStabilityCert`

Definition 279.11 (Full-axis ledger policy)**.** *The ledger policy* FullAxisLedger *retains the prefix thread and every visible thread-member as part of the construction history of the seal.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisLedgerPolicy`

Theorem 279.12 (Concrete $\mathsf{FullAxis}^{\uparrow}$ naming certificate)**.** *The tuple*

$$\mathsf{NameCert}_{\mathsf{FullAxis}^{\uparrow}} := (\mathsf{FullAxisSource}, \mathsf{FullAxisPattern}, \mathsf{FullAxisClassifier}, \mathsf{FullAxisStability}, \mathsf{FullAxisLedger})$$

specialised to the zero-spine prefix thread forms a valid naming certificate for the derived seal interface $\mathsf{FullAxis}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.AxisZeckendorf.FullAxis.fullAxis namecert`

Proof. The five fields are supplied by the preceding definitions, instantiated at $T = \mathsf{ZSpineThread}$. Stability includes the seal certificate together with the boundary-finite declaration. □

Corollary 279.13 ($\mathsf{FullAxis}^{\uparrow}$ is licensed, not primitive)**.** *The interface* $\mathsf{FullAxis}^{\uparrow}$ *is licensed by* $\mathsf{NameCert}_{\mathsf{FullAxis}^{\uparrow}}$*. It is a structural seal, not a metric completion, and it is not a primitive object of the finite kernel.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxis semantic name certificate`

Theorem 279.14 (FullAxis completion sibling separation)**.** *The* $\mathsf{FullAxis}^{\uparrow}$ *prefix-thread seal is separated from the completion routes of chapter 128 and chapter 38. Its carrier is finite zero-spine membership together with the finite excluded boundary marker* `boundary_01`*, while the completion and* $\mathsf{Real}^{\uparrow}$ *routes consume Cauchy,* $\mathsf{StreamName}^{\uparrow}$*, metric, or seal data. Consequently the* $\mathsf{FullAxis}^{\uparrow}$ *seal exports no metric limit, real-completion row, Cauchy witness, or host-level completion coordinate.*

Lean4 ✓ `BEDC.Derived.FullAxisUp.FullAxisUp completion sibling separation`

Proof. Start with Definition 279.15. The boundary marker `boundary_01` is finite and outside the zero-spine thread, so it cannot serve as a limit witness or completion point.

Apply Definition 279.16. The source, pattern, classifier, stability, ledger, and finite boundary rows exhaust the $\mathsf{FullAxis}^{\uparrow}$ seal surface over the prefix thread.

Now compare the sibling routes. The complete-metric and real chapters consume Cauchy, $\mathsf{StreamName}^{\uparrow}$, metric, or seal rows, none of which is present in the exhausted prefix-thread inventory. Therefore the FullAxis seal remains a structural finite boundary and exports no metric limit or real-completion row. □

Theorem 279.15 (FullAxis prefix-thread boundary disjointness)**.** *For the zero-spine prefix thread of Definition 279.2, the finite boundary marker* `boundary_01` *is outside the thread carrier. Consequently it cannot serve as a thread-member witness, a prefix-thread membership row, or a limit witness for the structural seal.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxis e1 e0 empty boundary exclusion`

Proof. Unfold the zero-spine prefix thread of Definition 279.2. Its membership predicate is exactly $\mathsf{ZeroSpine}$, closed from e by repeated $\mathsf{E0}$-extension.

Apply Definition 279.5. The finite marker `boundary_01` has an $\mathsf{E1}$ head and therefore is not a member of the zero-spine carrier.

Since Definition 279.7 reads source rows only from thread membership, the boundary marker cannot be repacked as a source member. The seal certificate therefore treats it only as a finite excluded boundary token, not as a completion point or infinity. □

The $\mathsf{FullAxis}^{\uparrow}$ seal consumes the $\mathsf{AxisNat}^{\uparrow}$ zero-spine carrier of chapter 276 together with the finite-boundary refusal surface of chapter 554. Thus `boundary_01` is a boundary over the AxisNat zero-spine and not a metric, Cauchy, completion, or Real-limit row.

Theorem 279.16 (FullAxis seal source exhaustion)**.** *The* $\mathsf{FullAxis}^{\uparrow}$ *source, pattern, classifier, stability, and ledger rows are exhausted by zero-spine prefix-thread membership together with the finite boundary exclusion of Definition 279.15. They contain no completion witness, no metric limit, and no thread-member row for* `boundary_01`*.*

Proof. Start with the zero-spine source. By Definition 279.7, the source rows are exactly the members of the chosen prefix thread, and by Definition 279.2 those members are precisely the $\mathsf{ZeroSpine}$ histories.

The pattern and classifier rows add no outside carrier. The pattern row of Definition 279.8 is repeated E0-extension inside the same thread, and the classifier row of Definition 279.9 classifies only pairs already present by joint thread membership.

Apply Definition 279.15. The finite marker `boundary_01` is outside that carrier, so it cannot be read as a thread member or as a limit witness. The licensing corollary Definition 279.13 then repacks the displayed source, pattern, classifier, stability, and ledger rows as the whole structural seal surface. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSeal source exhaustion`

279.5 Horizon obligations

Theorem 279.17 (Boundary-finite witness, horizon)**.** *For every prefix thread T, the seal does not equate any thread-member with a non-thread history that purports to be a limit object.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSourceSpec boundary 01 thread separation`

Theorem 279.18 (Distinct from $\mathsf{Real}^{\uparrow}$, horizon)**.** $\mathsf{FullAxis}^{\uparrow}$ *does not entail any of the algebraic operations of* $\mathsf{Real}^{\uparrow}$ *(Cauchy, limit uniqueness, field structure).*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSeal not real horizon`

Remark 279.19 (Where Real is licensed)**.** *The Cauchy / Real route is licensed inside the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *chapters.* $\mathsf{FullAxis}^{\uparrow}$ *does not enter that route; any attempt to identify the two interfaces requires a separate bridge.*

Theorem 279.20 (FullAxis prefix-thread obligation inventory)**.** *The* $\mathsf{FullAxis}^{\uparrow}$ *horizon obligation surface consists only of the* BHist*-facing rows displayed by the prefix-thread seal:*

1. *the carrier row supplied by the zero-spine prefix thread;*
2. *the classifier row on jointly visible thread members;*
3. *the stability row recording directedness, monotonicity, prefix-closure, and finite-boundary exclusion;*
4. *the ledger row retaining the visible prefix-thread provenance;*
5. *the zero-spine thread rows generated by* e *and repeated* E0*-extension;*
6. *the finite boundary marker row for* `boundary_01` *as a non-thread history;*
7. *the source-exhaustion row stating that no source outside thread membership is exposed.*

No row of this inventory is a RealUp completion, metric limit, compactness witness, or host-level infinity.

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisPrefixThread obligation inventory`

Proof. Use the prefix-thread definitions Definition 279.1 and Definition 279.2 for the carrier and zero-spine thread rows. The source, classifier, stability, and ledger rows are exactly the displayed fields of Definition 279.7, Definition 279.9, Definition 279.10, and Definition 279.11.

The boundary marker enters only through Definition 279.4. Apply Definition 279.5 and then Definition 279.17: the marker is finite and is not a thread member.

Finally apply Definition 279.16. It repacks the prefix-thread carrier, classifier, stability, ledger, zero-spine thread, boundary marker, and source-exhaustion rows as the whole $\mathsf{FullAxis}^{\uparrow}$ horizon surface, with no completion or limit row exported. □

Theorem 279.21 (FullAxis boundary marker finite obligation)**.** *The marker* `boundary_01` *remains a finite non-thread* BHist *marker in the* $\mathsf{FullAxis}^{\uparrow}$ *obligation surface. It cannot be read as a limit object, completion point, compactness witness, or point at infinity.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisBoundaryMarker finite obligation`

Proof. The definition Definition 279.4 presents `boundary_01` as the finite history $\mathsf{E1}(\mathsf{E0}(\mathsf{e}))$.

Apply Definition 279.5. Since the zero-spine prefix thread exposes only ZeroSpine histories, the boundary marker has no thread-member row.

Now use Definition 279.17. It states that the seal does not equate a thread member with a non-thread history presented as a limit object. The boundary marker is therefore retained only as the finite excluded marker named by Definition 279.15. □

Theorem 279.22 (FullAxis Real route separation obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$*, Cauchy, metric-completion, compactness, or limit reading of a* $\mathsf{FullAxis}^{\uparrow}$ *consumer must factor through the Real, Completion, Metric, or Compact chapters. Such a reading is not projected from the* $\mathsf{FullAxis}^{\uparrow}$ *prefix-thread seal, whose public rows are only the carrier, classifier, stability, ledger, zero-spine thread, finite boundary marker, and source-exhaustion rows listed in Definition 279.20.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSeal real route separation obligation`

Proof. Start from the structural seal surface. By Definition 279.20, the exposed rows are all BHist-facing prefix-thread rows and finite boundary rows.

Apply Definition 279.21. The only named boundary marker is finite and non-thread, so it cannot supply a Cauchy object, metric completion point, compact witness, or limit.

Finally use Definition 279.18. It separates the $\mathsf{FullAxis}^{\uparrow}$ horizon from $\mathsf{Real}^{\uparrow}$ algebraic and completion operations. Therefore any RealUp, Cauchy, metric-completion, compactness, or limit reading must enter through the chapters where those structures are licensed, not through projection from $\mathsf{FullAxis}^{\uparrow}$. □

Theorem 279.23 (FullAxis seal object name alignment)**.** *The critical-path horizon name* $\mathsf{FullAxisSeal}^{\uparrow}$ *is the seal-facing registry name for the same prefix-thread certificate exported here as* $\mathsf{FullAxis}^{\uparrow}$*. The registry alias does not add an object row: the only exported rows remain the zero-spine prefix thread, finite boundary marker, source, pattern, classifier, stability, ledger, and RealUp-separation rows displayed in this chapter.*

Proof. Read the certificate through Definition 279.12. The exported tuple is the source, pattern, classifier, stability, and ledger certificate for the zero-spine prefix thread.

Apply Definition 279.20. It lists the public horizon rows as the prefix-thread carrier, finite boundary marker, classifier, stability, ledger, and boundary-exclusion data, all over the same zero-spine source.

Finally use Definition 279.22. It keeps RealUp, Cauchy, metric-completion, compactness, and limit readings outside the seal-facing registry alias. Thus $\mathsf{FullAxisSeal}^{\uparrow}$ is only the registry name for this exported structural seal surface. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxisSeal object name alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{FullAxis}^{\uparrow}}$.

Constructive story: A full-axis seal starts from a prefix thread: a history-membership predicate containing e and closed under E0-extension, specialized here to the zero-spine thread already generated from the empty history. The carrier admits exactly thread members, while the finite marker `boundary_01` $= \mathsf{E1}(\mathsf{E0}(\mathsf{e}))$ is deliberately kept outside the zero-spine carrier because its head is an E1 rather than an E0. Two visible seal entries classify together by joint membership in

the same thread, so the source specification is prefix-thread membership, the pattern specification is repeated E0-extension inside that thread, and no RealUp, Cauchy, compactness, metric, or limit row is accepted as a source. The ledger keeps the prefix thread, each visible thread member, the finite boundary marker, the boundary-exclusion row, and the public seal alias visible, rather than turning the excluded marker into a completion point. The five name-certificate obligations close on membership itself: e inhabits the carrier, joint-membership classification is reflexive, symmetric, and transitive on thread members, and any classified visible target is already a member of the same admitted thread. A history h lies in the signature gap of a FullAxis$^{\uparrow}$ token p when the source-domain accepts h as a member of the zero-spine prefix thread, the signature relation pairs it with an answer seal row and evidence ledger, and the token-introduction relation names that answer as p; unfolded, h is a finite E0-prefix member and not the finite excluded boundary marker unless p names that marker only as an outside-boundary row. The chapter registers this structural seal under the name FullAxis$^{\uparrow}$; it introduces no RealUp completeness, metric completion, compactness, limit object, infinity reading, or independent seal primitive.

Theory closure: TheoryGate(NameCert$_{\mathsf{FullAxis}\uparrow}$, scopedClosed)

Scope closed: The scoped surface binds the prefix-thread carrier, zero-spine boundary marker, finite-boundary row, ledger policy, public seal alias, and RealUp-route separation rows cited by Definition 279.20, Definition 279.21, Definition 279.22, and Definition 279.23.

Formal verification: FormalStatus(NameCert$_{\mathsf{FullAxis}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.AxisZeckendorf.FullAxis.FullAxis\_semantic\_name\_certificate`

Bridge status: none

Not claimed: It does not close RealUp completeness, metric completion, compactness, limits, or any reading of the boundary marker as infinity.

Upgrade path: Public closure requires a single exported public FullAxis$^{\uparrow}$ certificate theorem.

Chapter 280

A Concrete Naming Certificate for $\mathsf{TuringMachine}^{\uparrow}$

The Turing-machine interface $\mathsf{TuringMachine}^{\uparrow}$ packages a deterministic transition function on a finite alphabet together with a BHist tape carrier and a finite head-position witness. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$ (chapter 120), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

280.1 Carrier definition

A $\mathsf{TuringMachine}^{\uparrow}$ token carries: a finite state alphabet, a finite tape alphabet with a designated blank, a deterministic transition function, an initial-state name, and a halting-state predicate. The carrier accepts a configuration history h when there is a $\mathsf{Seq}^{\uparrow}$ trace of $(\mathsf{state}, \mathsf{tape}, \mathsf{head})$ rows whose zero entry is the initial configuration on a given input, every one-step extension matches the transition function, and halted rows repeat. The classifier is hsame on the configuration trace; the ledger keeps the deterministic transition table, head movement, and tape readback visible; the source covers finite alphabets, configurations, and traces; the pattern records initial-row, transition-step, halted-repetition, and tape-readback rows.

Theorem 280.1 (TuringMachine configuration trace carrier)**.** *A configuration trace accepted by the* $\mathsf{TuringMachine}^{\uparrow}$ *carrier is a* $\mathsf{Seq}^{\uparrow}$ *row of* $(\mathsf{state}, \mathsf{tape}, \mathsf{head})$ *histories over* BHist *with a unary time index, an initial configuration row, and the displayed finite state and tape alphabets. The source rows consumed by the carrier are exactly the* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *unary time and head-position rows, the* $\mathsf{NameCert}_{\mathsf{Seq}^{\uparrow}}$ *configuration trace row, and the* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *bounded deterministic-trace row.*

Proof. Unpack the carrier prose above. The accepted history is not a host-level run; it is the displayed $\mathsf{Seq}^{\uparrow}$ trace whose entries are triples of BEDC histories, and whose index row is supplied by $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$.

The initial row is one entry of the same trace. The finite alphabets and configuration components are therefore visible source rows, while $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ supplies the bounded deterministic trace format that keeps the program code, bound, and endpoint readback in the ledger. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineConfigurationTraceCarrier configuration trace surface`

Theorem 280.2 (TuringMachine deterministic step transport)**.** *Let two current configuration rows of a* $\mathsf{TuringMachine}^{\uparrow}$ *trace be related by* hsame*, and let their deterministic transition-table rows be*

transported by the same classifier data. Then the next-step configuration rows produced by the transition table are again related by hsame*. The transport uses the* Cont *trace-extension row and the displayed* $\mathsf{NameCert}_{\mathsf{Seq}^\uparrow}$ *classifier on configuration traces.*

Proof. Read the current configuration comparison through the hsame classifier on the configuration trace. The deterministic transition table is a visible ledger row, so transporting it carries the state update, tape-cell write, and head move to the transported current configuration.

Apply the Cont trace-extension row to append the transported one-step result to the $\mathsf{Seq}^\uparrow$ trace. Since the transition relation is deterministic, there is only one next configuration row over the transported current row, and the $\mathsf{NameCert}_{\mathsf{Seq}^\uparrow}$ classifier records it as the same trace step up to hsame. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineDeterministicStepTransport step stability`

Theorem 280.3 (TuringMachine halted repetition ledger exactness)**.** *Once the halting predicate is displayed for a configuration row in a* $\mathsf{TuringMachine}^\uparrow$ *trace, every later accepted trace-extension ledger row records only repetitions of that halted configuration. No transition-table branch, oracle row, or fresh tape update is available after the halted row.*

Proof. Use the carrier clause that halted rows repeat. The halting predicate is a visible ledger entry, so a later trace extension must cite the halted configuration as its source row rather than reopen the deterministic transition table.

The Cont row then extends the trace by the same configuration history. The classifier for the repeated suffix is the ambient hsame on configuration traces, and the exactness ledger records only the halted configuration, its tape, and its head-position witness. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineHaltedTrace ledger exactness`

Theorem 280.4 (TuringMachine finite tape readback carrier)**.** *Every visible tape readback of a* $\mathsf{TuringMachine}^\uparrow$ *accepted trace is a finite* BHist *row indexed by the* $\mathsf{Nat}^\uparrow$ *head-position witness. The readback surface is carried by the* $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$ *head row, the* $\mathsf{NameCert}_{\mathsf{Seq}^\uparrow}$ *configuration trace row, and the visible tape component of the current configuration; it does not expose an infinite tape object.*

Proof. Project a carried configuration row to its tape and head components. The head is classified by the consumed $\mathsf{NameCert}_{\mathsf{Nat}^\uparrow}$ row, while the configuration is one entry of the consumed $\mathsf{NameCert}_{\mathsf{Seq}^\uparrow}$ trace.

The tape readback therefore names only the finite BEDC row addressed by that head-position witness. Transport along hsame preserves the same finite readback row, and the trace ledger records the accessed cell without adding an unbounded external tape carrier. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineFiniteTapeReadbackCarrier finite readback surface`

Theorem 280.5 (TuringMachine obligation surface)**.** *The complete* $\mathsf{TuringMachine}^\uparrow$ *obligation surface over* BHist *is formed by the configuration trace carrier, the* hsame *classifier on configuration traces, the deterministic transition row, the halted-repetition row, the finite tape readback row, and the bounded* $\mathsf{Computable}^\uparrow$ *simulation trace rows. These rows consume exactly the displayed* $\mathsf{Nat}^\uparrow$ *unary time and head-position witnesses, the* $\mathsf{Seq}^\uparrow$ *configuration trace,* Cont *trace-extension data, and the ledger rows for initial configurations, transition steps, halted repetitions, and bounded readback.*

Proof. Collect the carrier rows from Definition 280.1 and the finite readback rows from Definition 280.4. These identify the accepted histories as BHist configuration rows indexed by the consumed $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Seq}^{\uparrow}$ sources.

Next add the deterministic and halted parts. The step transport theorem supplies the deterministic transition row and its Cont/hsame stability, while the halted-repetition theorem supplies the exact repeated suffix.

Finally include the bounded $\mathsf{Computable}^{\uparrow}$ trace row. Since the carrier exposes only the displayed initial row, transition step, halted repetition, finite readback, and bounded-simulation ledger entries, the listed rows exhaust the obligation surface. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineObligationSurface bounded trace ledger`

Theorem 280.6 (TuringMachine source-classifier obligation)**.** *The* $\mathsf{TuringMachine}^{\uparrow}$ *source reads exactly the finite state alphabet, finite tape alphabet with blank row, unary time rows, unary head-position rows,* $\mathsf{Seq}^{\uparrow}$ *configuration trace, deterministic transition table, and* $\mathsf{Computable}^{\uparrow}$ *bounded trace. Classifier transport is* hsame *on configuration trace rows, with the deterministic transition table and finite readback rows transported only through the displayed* Cont *trace-extension surface.*

Proof. Project a token to its source rows. The alphabets, transition table, time indices, head positions, configuration trace, and bounded trace are explicit coordinates of the accepted package.

The classifier has no separate host-machine equality. It compares configuration trace rows by hsame, then transports transition and readback rows along the same displayed Cont extension data.

Because no other source coordinate is available in the package, every admitted classifier transport is obtained from those rows and no oracle, universal machine, or external computation predicate can enter the source. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineSourceClassifierObligation source classifier surface`
Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineSourceClassifierObligation cont transport`

Theorem 280.7 (TuringMachine ledger exactness obligation)**.** *The* $\mathsf{TuringMachine}^{\uparrow}$ *ledger is exhausted by initial-row, transition-step, halted-repetition, finite tape readback, and bounded-simulation rows. It contains no oracle transcript, nondeterministic branch row, universal-machine row, or halting-decision data.*

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineLedgerExactnessObligation rows and boundary`

Proof. Read the ledger fields of an accepted $\mathsf{TuringMachine}^{\uparrow}$ token. The initial row starts the $\mathsf{Seq}^{\uparrow}$ trace, transition-step rows append deterministic successors, and halted-repetition rows append the fixed halted configuration.

Finite tape readback is the projection of a displayed configuration row through the head-position witness, and bounded simulation is the consumed $\mathsf{Computable}^{\uparrow}$ trace prefix.

These are all ledger constructors named by the carrier. Since the package has no field for oracles, branch choice, universal-machine coding, or halting decision, those data are excluded from ledger exactness. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineLedgerExactnessObligation no external rows`

280.2 The certificate

Theorem 280.8 (TuringMachine trace source obligation)**.** *The* $\mathsf{TuringMachine}^{\uparrow}$ *trace source is a* $\mathsf{Seq}^{\uparrow}$ *configuration carrier over* BHist *whose rows are finite triples* $(\mathsf{state}, \mathsf{tape}, \mathsf{head})$*. Its public source surface is the* $\mathsf{Nat}^{\uparrow}$ *unary time row, the* $\mathsf{Nat}^{\uparrow}$ *head-position row, the* $\mathsf{Seq}^{\uparrow}$ *configuration trace row, the finite state and tape alphabets, and the* $\mathsf{Computable}^{\uparrow}$ *bounded deterministic-trace row.*

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineTraceSourceObligation source rows`

Proof. Project the accepted machine token to its trace coordinate. The trace coordinate is already a $\mathsf{Seq}^{\uparrow}$ row, and each entry is a triple of BEDC histories, so the carrier is a finite BHist configuration sequence rather than a host-level execution.

The visible indices are supplied by the consumed $\mathsf{Nat}^{\uparrow}$ rows. The finite alphabets and the bounded deterministic-trace row are ledger coordinates of the same token, so every source row in the theorem is obtained by projection from the displayed carrier package. □

Theorem 280.9 (TuringMachine deterministic step stability)**.** *For a* $\mathsf{TuringMachine}^{\uparrow}$ *trace, one-step transition rows are stable under the displayed* Cont *and* hsame *transport. If current configurations, transition table rows, and head-position witnesses are transported by the public classifier, then the next configurations produced by the deterministic step row are again* hsame*.*

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineStepRow hsame transport`

Proof. Read the current configurations through the hsame classifier on the $\mathsf{Seq}^{\uparrow}$ trace. The transition table is a visible ledger row, so its transported copy determines the same state update, tape write, and head movement over the transported current configuration.

Apply the Cont row for trace extension. Determinism gives a single next row for the transported data, and the $\mathsf{Seq}^{\uparrow}$ classifier records the two extended traces as hsame at the next time index. □

Theorem 280.10 (TuringMachine halted repeat obligation)**.** *If a* $\mathsf{TuringMachine}^{\uparrow}$ *configuration row is marked halted, every accepted later trace row repeats that halted configuration up to* hsame*. The repeated suffix exposes only the halted configuration, its tape row, its head-position witness, and the halted ledger entry.*

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineHaltedTrace repeat obligation`

Proof. Use the halted-row clause of the carrier. Once the halted predicate is visible, the accepted trace-extension field cites the halted configuration as its source instead of reopening the transition table.

The repeated suffix is then built by the same Cont extension row. Since each later entry is classified with the halted configuration by hsame, the ledger contains no additional branch, oracle query, or tape update beyond the repeated halted row. □

Theorem 280.11 (TuringMachine bounded readback soundness)**.** *Every* $\mathsf{Computable}^{\uparrow}$ *bounded readback consumed by* $\mathsf{TuringMachine}^{\uparrow}$ *is read from the displayed finite trace prefix. The readback endpoint is the tape value addressed by the* $\mathsf{Nat}^{\uparrow}$ *head-position witness inside the current configuration row, and no infinite tape carrier or halting-decision row is exported.*

Proof. Project the bounded readback row from the consumed $\mathsf{Computable}^{\uparrow}$ surface. Its bound is a finite trace prefix, and each trace entry is a $\mathsf{Seq}^{\uparrow}$ configuration row with a visible tape component and head-position witness.

The endpoint readback therefore evaluates the displayed tape row at the displayed finite head witness. Transport by hsame preserves that bounded endpoint, while the absence of an infinite tape coordinate prevents the readback from exporting any stronger host computation claim. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineBoundedReadbackSoundness endpoint transport`
Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineBoundedReadback soundness`

Theorem 280.12 (TuringMachine NameCert surface obligation)**.** *The public* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *surface is obtained by repacking the configuration trace carrier, deterministic transition classifier,* Cont *and* hsame *stability rows, halted-repeat exactness, bounded readback exactness, and deterministic transition ledger. This surface binds only the listed* BHist*,* $\mathsf{Seq}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* Cont*,* hsame*, and ledger rows.*

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineNameCertSurface obligation`
Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineNameCertSurfaceObligation public surface`

Proof. First collect the carrier rows from Definition 280.8. These provide the finite configuration trace, time index, head-position witness, alphabets, and bounded deterministic-trace source.

Next add the stability and exactness rows. The deterministic-step row is stable by Definition 280.9, halted suffixes are exact by Definition 280.10, and bounded readback is exact by Definition 280.11.

Finally repack those rows as the naming certificate ledger. The package has no field for nondeterminism, oracle queries, parallel execution, a universal machine theorem, or a halting-decision classifier, so the public surface is exactly the listed finite-row obligation surface. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineObligationSurface rows`

Theorem 280.13 (TuringMachine public certificate export)**.** *The configuration-trace carrier, deterministic step transport, halted repeat obligation, finite tape readback carrier, bounded readback soundness, source-classifier obligation, and ledger exactness obligation repack into one public* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *certificate export over* BHist*,* hsame*,* Cont*,* Pkg*,* UnaryHistory*, and the bounded* $\mathsf{Computable}^{\uparrow}$ *verdict. The exported packet has the same finite transition table, configuration trace, halted suffix, bounded tape readback, and ledger rows as the component obligations.*

This public certificate export does not claim unbounded halting, a universal machine theorem, nondeterministic execution, oracle rows, or host computation.

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachinePublicCertificate export`

Proof. First collect the visible certificate rows. Use Definition 280.1, Definition 280.2, Definition 280.3, and Definition 280.4 to obtain the finite trace carrier, deterministic step route, halted suffix, and tape readback carrier.

Next add the public stability and exactness rows. Apply Definition 280.8, Definition 280.6, Definition 280.7, and Definition 280.11. These rows supply the BHist source, hsame classifier, Cont transport, Pkg provenance, UnaryHistory time bound, and bounded $\mathsf{Computable}^{\uparrow}$ verdict.

Finally invoke Definition 280.12. The component rows are already exactly the rows required by the public naming surface, so repacking them exports one $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ certificate. Since no component theorem contains an unbounded halting row, universal-machine row, nondeterministic branch, oracle coordinate, or host computation field, the exported certificate contains none either. □

Theorem 280.14 (TuringMachine finite trace standard bridge)**.** *The finite deterministic trace surface exported in Definition 280.13 has a standard bridge to the ordinary finite-trace deterministic machine reading. On the BEDC side the data are the finite state alphabet, finite tape alphabet with blank, transition table, input row, configuration trace, halted suffix, and bounded readback rows. On the ordinary finite-trace side the data are the corresponding state, tape, head position, time bound, configuration trace, halted suffix, and bounded readback rows.*

The bridge preserves these rows both ways: a BEDC accepted finite trace reads as an ordinary bounded deterministic run with the same state, tape, head, bound, trace, halted suffix, and readback values, and an ordinary bounded deterministic run with those rows repacks as the BEDC public certificate surface. The bridge does not add halting decidability, a universal machine, oracle-machine rows, nondeterministic branches, or unbounded computation.

Proof. Read the BEDC-to-standard direction from the public certificate export. The configuration carrier supplies the state, tape, head, and time-bound rows; the deterministic step transport supplies the one-step transition rows; the halted repeat obligation supplies the halted suffix; and the bounded readback theorem supplies the visible tape values.

For the standard-to-BEDC direction, take an ordinary bounded deterministic run with the same finite alphabets, transition table, initial row, trace, halted suffix, and readback rows. These are exactly the rows required by the source, classifier, transport, and ledger obligations in the public certificate export, so the run repacks as the $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ surface.

No extra computational principle is introduced in either direction. Every row used by the bridge is already finite and bounded, so the bridge has no field for halting decidability, universal machines, oracle machines, nondeterminism, or unbounded computation. □

Lean4 ✓ `BEDC.Derived.TuringMachineUp.TuringMachineFiniteTrace standard bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TuringMachine}^{\uparrow}$ token names a deterministic finite-state transition table over a finite tape alphabet with a designated blank symbol. A configuration history is built from a $\mathsf{Seq}^{\uparrow}$ trace of (state, tape, head-position) rows whose initial row is the boot configuration on a $\mathsf{Nat}^{\uparrow}$-encoded input, whose one-step rows follow the transition function, and whose halted rows repeat the halted configuration. The classifier is hsame on the configuration trace, transport repacks the trace through the deterministic transition table, and the source binds finite-alphabet, finite-state, finite-trace, and finite-head-displacement rows. The pattern records the initial-row, transition-step, halted-repetition, and finite-tape-readback rows; the ledger keeps the deterministic transition table, head movement, and tape values visible. The five name-certificate obligations follow: the inhabited identity certificate is the boot trace at length zero; classifier stability is reflexive, symmetric, and transitive hsame on traces; transport repacks configurations through the transition function; ledger exactness identifies the accepted history with the bounded simulation row in the sense of chapter 199; the extension boundary fixes the halted-repetition rule. A history h lies in the signature gap of a $\mathsf{TuringMachine}^{\uparrow}$ token p when the source-domain accepts a configuration trace with the right transition table, the signature relation pairs it with the halted output, and token introduction names that output as p. The chapter registers this finite-trace machine surface under the name $\mathsf{TuringMachine}^{\uparrow}$; it is a deterministic-transition-table object with no oracle, nondeterminism, parallelism, halting decision, or universal-machine claim.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridged scope is the certificate export in Definition 280.13 together with the finite-trace standard bridge in Definition 280.14, grounded by Definition 280.5, Definition 280.6, Definition 280.7, Definition 280.1, Definition 280.2, Definition 280.3, and Definition 280.4.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TuringMachineUp.TuringMachineNameCertSurfaceObligation\_public\_surface`

Bridge status: paperBridge

Not claimed: The chapter does not close halting decidability, universal-machine theorems, oracle Turing machines, parallel/nondeterministic machines, or any unbounded standard-model computability theorem beyond the finite-trace deterministic bridge.

Upgrade path: Any machine-checked bridge-status uplift is recorded on the verification axis.

Chapter 281

A Concrete Naming Certificate for $\mathsf{RecursiveFn}^{\uparrow}$

The partial-recursive interface $\mathsf{RecursiveFn}^{\uparrow}$ packages partial functions $\mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$ generated by zero, successor, projection, composition, primitive recursion, and bounded minimisation. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{List}^{\uparrow}}$ (chapter 36), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

281.1 Carrier definition

A $\mathsf{RecursiveFn}^{\uparrow}$ token carries an inductive constructor history of generation steps: zero, successor, projection π_i^k, composition row, primitive-recursion row with explicit base and step subterms, and bounded-minimisation row with an explicit unary bound. The carrier accepts a partial-graph value $F(n, m)$ when the constructor-history evaluation on the $\mathsf{Nat}^{\uparrow}$-input row n returns $\mathsf{Nat}^{\uparrow}$-output m within the displayed bound, and the evaluation trace is a $\mathsf{Seq}^{\uparrow}$ object. The classifier is hsame on output rows after the constructor history is fixed; the ledger keeps the constructor sequence, intermediate evaluation rows, and bounded-minimisation witness visible.

Theorem 281.1 (RecursiveFn zero successor projection obligation)**.** *The* $\mathsf{RecursiveFn}^{\uparrow}$ *constructor-history surface contains visible zero, successor, and projection rows over* $\mathsf{Nat}^{\uparrow}$ *inputs. For a displayed finite constructor history, the zero row returns the displayed* $\mathsf{Nat}^{\uparrow}$ *zero output, the successor row transports the input to its displayed successor output, and the projection row* π_i^k *selects the* i*th* $\mathsf{Nat}^{\uparrow}$ *input from the finite input tuple. In each case the accepted graph row is classified by* hsame *on the output once the constructor history is fixed.*

These base constructor rows are finite BHist *and* $\mathsf{Seq}^{\uparrow}$ *rows only. They do not assert unbounded* μ*-recursion totality, oracle queries, or a Church-Turing bridge.*

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnZeroSuccessorProjection obligation surface`

Proof. Inspect the head constructor of the displayed finite constructor history.

In the zero case, the ledger contains no recursive subtrace; the accepted output is the visible $\mathsf{Nat}^{\uparrow}$ zero row, and hsame compares that output row with itself.

In the successor case, the ledger contains the input $\mathsf{Nat}^{\uparrow}$ row and its successor output row. The accepted graph row is obtained by the constructor route from the input to the displayed successor.

In the projection case, the finite input tuple supplies the selected coordinate π_i^k. The ledger records the coordinate choice and the selected $\mathsf{Nat}^{\uparrow}$ row, so no oracle or host computation is introduced. □

Theorem 281.2 (RecursiveFn composition and primitive-recursion obligation)**.** *For a displayed* $\mathsf{RecursiveFn}^{\uparrow}$ *constructor history, composition and primitive recursion are compatible with the visible* $\mathsf{Seq}^{\uparrow}$ *evaluation traces. A composition row accepts exactly when the inner constructor histories evaluate on the displayed* $\mathsf{Nat}^{\uparrow}$ *inputs to intermediate* $\mathsf{Nat}^{\uparrow}$ *rows and the outer history evaluates on those rows. A primitive-recursion row accepts exactly when the base row and each displayed step row appear in order in the finite* $\mathsf{Seq}^{\uparrow}$ *trace, with* hsame *comparing only the final* $\mathsf{Nat}^{\uparrow}$ *output after the constructor history is fixed.*

The obligation is a constructor-history compatibility statement over finite BHist *and* $\mathsf{Seq}^{\uparrow}$ *ledgers. It excludes unbounded μ-recursion totality, oracle queries, and Church-Turing bridge claims.*

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnCompositionPrimitiveRecursion obligation surface`

Proof. Proceed by inspecting the displayed constructor history.

For composition, first read the inner histories. Each one consumes the same visible $\mathsf{Nat}^{\uparrow}$ input tuple and produces an intermediate output row in the $\mathsf{Seq}^{\uparrow}$ evaluation trace. The outer constructor history then consumes exactly those intermediate rows, so the composed graph row is the final row already present in the trace.

For primitive recursion, read the base row at the displayed zero input and then inspect each successor step in the finite recursion trace. The step row uses the previous output row and the current $\mathsf{Nat}^{\uparrow}$ index already recorded in the ledger. The final accepted output is therefore the final visible trace row.

No branch of the inspection adds an unbounded search, an oracle row, or a standard recursion-theory bridge; all movement is through the displayed finite constructor history. □

Theorem 281.3 (RecursiveFn bounded minimisation ledger exactness)**.** *For a displayed bounded-minimisation constructor in* $\mathsf{RecursiveFn}^{\uparrow}$*, the accepted output row is exactly the first witness row recorded in the finite* BHist *ledger under the visible* UnaryHistory *bound, or the displayed failure row when the bounded trace records no witness. The* $\mathsf{Seq}^{\uparrow}$ *evaluation trace contains every tested* $\mathsf{Nat}^{\uparrow}$ *input up to the bound, and* hsame *compares only the resulting bounded output row after the constructor history is fixed.*

The exactness statement is bounded and ledger-local. It does not claim unbounded μ-recursion totality, oracle access, or a Church-Turing bridge.

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnBoundedMinimisationLedger exactness`

Proof. Inspect the bounded-minimisation constructor row. The UnaryHistory coordinate gives the finite bound, and the BHist ledger lists the tested $\mathsf{Nat}^{\uparrow}$ inputs and their $\mathsf{Seq}^{\uparrow}$ evaluation rows in order.

If a witness row appears, take the first such displayed row. Constructor inspection shows that every earlier tested row is recorded as a non-witness, so ledger exactness identifies the accepted output with that first witness.

If no witness row appears, the same finite inspection reaches the displayed failure row at the end of the bounded trace. Since the trace is bounded by the visible unary history, no unbounded search or oracle query is used. □

Theorem 281.4 (RecursiveFn Nat input stability obligation)**.** *Let x and y be displayed* $\mathsf{Nat}^{\uparrow}$ *input rows related by the* $\mathsf{Nat}^{\uparrow}$ *classifier, and fix a finite* $\mathsf{RecursiveFn}^{\uparrow}$ *constructor history. If the displayed* BHist *and* $\mathsf{Seq}^{\uparrow}$ *evaluation trace evaluates the history at x to an accepted output row, then the transported trace evaluates the same history at y to an output row classified by* hsame*. The*

obligation uses only BHist *constructor rows,* $\mathsf{Seq}^{\uparrow}$ *evaluation rows,* $\mathsf{Nat}^{\uparrow}$ *input and output rows, and the fixed constructor ledger.*

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnNatInputStability obligation`

Proof. Inspect the fixed constructor history. For zero, successor, and projection, apply Definition 281.1; the transported input row changes only the displayed $\mathsf{Nat}^{\uparrow}$ input component, and the output comparison is the same hsame row recorded by the constructor ledger.

For composition and primitive recursion, use Definition 281.2. The inner traces, base row, and step rows are already visible $\mathsf{Seq}^{\uparrow}$ rows, so classifier transport repacks each intermediate $\mathsf{Nat}^{\uparrow}$ output and leaves the final output classified by hsame.

For bounded minimisation, apply Definition 281.3. The transported trace uses the same visible bound and the same finite list of tested rows, so the first displayed witness, or the displayed failure row, is transported without adding any unbounded search. □

Theorem 281.5 (RecursiveFn constructor ledger coverage obligation)**.** *Every accepted* $\mathsf{RecursiveFn}^{\uparrow}$ *graph row is covered by exactly one visible constructor case among zero, successor, projection, composition, primitive recursion, and bounded minimisation. In that case the ledger exposes the corresponding finite subtrace: the direct base-constructor row, the inner and outer composition rows, the primitive-recursion base and step rows, or the bounded-minimisation test rows under the displayed unary bound.*

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnConstructorLedger coverage obligation`

Proof. Proceed by the head constructor of the accepted constructor history.

The zero, successor, and projection heads are covered by Definition 281.1. Each head has a distinct constructor tag, so its accepted row cannot also be a composition, primitive-recursion, or bounded-minimisation row.

The composition and primitive-recursion heads are covered by Definition 281.2. Their visible $\mathsf{Seq}^{\uparrow}$ subtraces distinguish the inner-and-outer composition case from the base-and-step recursion case.

The bounded-minimisation head is covered by Definition 281.3. Its UnaryHistory bound and finite test ledger are part of the displayed row, so no other constructor case supplies the same ledger shape. □

Theorem 281.6 (RecursiveFn bounded search scope obligation)**.** *For a displayed bounded-minimisation constructor in* $\mathsf{RecursiveFn}^{\uparrow}$*, the search reads only the* UnaryHistory *bound, the listed* $\mathsf{Nat}^{\uparrow}$ *test rows, and their* $\mathsf{Seq}^{\uparrow}$ *evaluation rows. The accepted row has no unbounded search coordinate, no oracle row, and no hidden host predicate beyond the finite constructor ledger.*

Lean4 ✓ `BEDC.Derived.RecursiveFnUp.RecursiveFnBoundedSearchScope obligation`

Proof. Start with the bounded-minimisation constructor and apply Definition 281.3. The theorem identifies the accepted output with the first visible witness row, or with the displayed failure row when the finite test ledger contains no witness.

The tested rows themselves are ordinary $\mathsf{Nat}^{\uparrow}$ rows in the same finite constructor history. The base-constructor, composition, and primitive-recursion obligations, Definition 281.1 and Definition 281.2, account for the evaluations of those rows inside the listed $\mathsf{Seq}^{\uparrow}$ trace.

Thus every read used by the bounded search is already named in the UnaryHistory, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Seq}^{\uparrow}$ ledger, and there is no exported row for an unbounded search or oracle. □

Theorem 281.7 (RecursiveFn public constructor-history export)**.** *The* $\mathsf{RecursiveFn}^{\uparrow}$ *public export is the finite* $\mathsf{NameCert}_{\mathsf{RecursiveFn}^{\uparrow}}$ *constructor-history surface assembled from the zero, successor, and projection obligation; the composition and primitive-recursion trace obligation; bounded-minimisation ledger exactness;* $\mathsf{Nat}^{\uparrow}$ *input stability; constructor-ledger coverage; and bounded-search scope obligation.*

Its public rows are exactly the displayed BHist *constructor history,* UnaryHistory *bound rows,* $\mathsf{Seq}^{\uparrow}$ *evaluation traces,* $\mathsf{Nat}^{\uparrow}$ *input and output rows, visible* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The export contains no unbounded* μ*-recursion totality assertion, no oracle row, and no Church–Turing bridge claim.*

Proof. Build the export from the constructor obligations. The base constructor rows come from Definition 281.1, and the composition and primitive-recursion traces come from Definition 281.2.

Add the bounded-search data next. Apply Definition 281.3 for the finite UnaryHistory bound and tested $\mathsf{Seq}^{\uparrow}$ rows, then use Definition 281.6 to keep every search read inside that displayed bounded ledger.

Now attach transport and coverage. The $\mathsf{Nat}^{\uparrow}$ input stability row is supplied by Definition 281.4, and the constructor-ledger coverage row is supplied by Definition 281.5. Together they show that each accepted public graph row has one displayed constructor case and transports only through visible hsame, Cont, Pkg, and local NameCert data.

The six cited obligations exhaust the public export. Since none contains a row for unbounded μ-recursion totality, oracle evaluation, or a Church–Turing bridge, those claims remain outside $\mathsf{NameCert}_{\mathsf{RecursiveFn}^{\uparrow}}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RecursiveFn}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RecursiveFn}^{\uparrow}$ token names a finite constructor history built from zero, successor, projection π_i^k, composition, primitive recursion with base/step subterms, and bounded minimisation with a displayed unary bound. The carrier accepts a graph row $F(n, m)$ when evaluating the constructor history on the $\mathsf{Nat}^{\uparrow}$-input row n over a $\mathsf{Seq}^{\uparrow}$ trace yields $\mathsf{Nat}^{\uparrow}$-output m within the displayed bound. Classifier control is hsame on outputs once the constructor history is fixed; the source covers $\mathsf{Nat}^{\uparrow}$ inputs and outputs, constructor histories, and $\mathsf{Seq}^{\uparrow}$ evaluation traces; the pattern records constructor-history admissibility, composition row, primitive-recursion base/step row, and bounded-minimisation row; the ledger keeps the constructor sequence, intermediate evaluation rows, and minimisation witness visible. The five name-certificate obligations follow from the inductive constructor-history evaluation: the inhabited identity certificate is the zero constructor; classifier stability is reflexive, symmetric, and transitive hsame on outputs; transport repacks evaluations through composition and recursion; ledger exactness identifies the graph row with the displayed evaluation; the extension boundary fixes the bounded-minimisation rule. A history h lies in the signature gap of a $\mathsf{RecursiveFn}^{\uparrow}$ token p when the source-domain accepts a constructor-history evaluation, the signature relation pairs it with the graph output, and token introduction names that output as p. The chapter registers this finite-constructor-history surface under the name $\mathsf{RecursiveFn}^{\uparrow}$; it is a partial-recursive function object with no general μ-recursion totality claim, no oracle, no Church-Turing bridge, and no standard recursion-theory library.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RecursiveFn}^{\uparrow}}$, publicClosed)

Scope closed: Definition 281.7 binds the constructor-history export over zero, successor, projection, composition, primitive recursion, bounded minimisation, BHist, UnaryHistory, $\mathsf{Seq}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RecursiveFn}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RecursiveFnUp.RecursiveFnBoundedMinimisationLedger\_exactness`

Bridge status: none

Not claimed: The chapter does not close unbounded μ-recursion totality, the Church-Turing thesis, oracle recursive functions, or a standard recursion-theory bridge.

Upgrade path: Bridged closure requires a standard recursive-function bridge over the same constructor-history, trace-compatibility, and bounded-ledger dependencies.

Chapter 282

A Concrete Naming Certificate for $\mathsf{ComplexityClass}^{\uparrow}$

The complexity-class interface $\mathsf{ComplexityClass}^{\uparrow}$ packages a resource-bounded acceptance predicate parametrised by an explicit unary modulus on input length. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ (chapter 280), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

282.1 Carrier definition

A $\mathsf{ComplexityClass}^{\uparrow}$ token carries: a deterministic $\mathsf{TuringMachine}^{\uparrow}$ acceptor, a unary modulus β that bounds the configuration-trace length as a function of the unary input length, and a resource type tag (time, space, or alternation depth). The carrier accepts an input history h when the bounded simulation of the acceptor on h within $\beta(|h|)$ rows reaches a halted accepting configuration. The classifier is hsame on input histories of the same unary length; the ledger keeps the modulus value, the trace length, and the halting verdict visible.

Theorem 282.1 (ComplexityClassUp bounded acceptance carrier)**.** *The* $\mathsf{ComplexityClass}^{\uparrow}$ *bounded acceptance carrier is a finite* BHist *package consisting of a* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *unary input-length row, a* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *deterministic acceptor row, a* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *bounded-simulation row, the unary resource modulus* β*, and the accepted halted verdict. The classifier is* hsame *on input histories of the same unary length, and the carrier exposes no nondeterminism, randomness, oracle computation, average-case complexity, or* P *versus* NP *row.*

Lean4 ✓ `BEDC.Derived.ComplexityClassUp.ComplexityClassBoundedAcceptanceCarrier bounded surface`

Proof. Project the finite input-length row from the $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ coordinate and the bounded configuration trace from the $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ coordinate. These rows determine the BHist source on which the unary modulus β is evaluated.

Read the halted verdict through the $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ ledger. The carrier accepts exactly when that bounded simulation row reaches an accepting halted configuration within the displayed modulus value.

The classifier uses hsame on transported inputs of the same unary length. Since the package has no row for nondeterministic branches, random bits, oracle calls, average-case sampling, or class separation, none of those data can be projected from the carrier. □

↪ variant BEDC.Derived.ComplexityClassUp.ComplexityClassBoundedAcceptanceCarrier bounded acceptance surface

↪ variant BEDC.Derived.ComplexityClassUp.ComplexityClassBoundedAcceptanceCarrier bounded trace surface

Theorem 282.2 (ComplexityClassUp resource modulus monotonicity)**.** *For* $\mathsf{ComplexityClass}^{\uparrow}$*, the unary resource modulus is a visible monotone ledger row: if two* BHist *inputs have unary lengths related by the* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *order row and the smaller input is accepted within* $\beta(n)$ *bounded* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *trace rows, then the larger bound* $\beta(m)$ *is an admissible* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *simulation budget for the* hsame *transported trace. The statement is only a bounded-resource monotonicity claim and does not assert a classical complexity-class collapse.*

Lean4 ✓ BEDC.Derived.ComplexityClassUp.ComplexityClassResourceModulusMonotonicity budget transport

Lean4 ✓ BEDC.Derived.ComplexityClassUp.ComplexityClassResourceModulusMonotonicity bound transport

Proof. Read the two unary input-length rows through $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$. The monotonicity coordinate of the modulus ledger transports the comparison of lengths to the comparison of bounds $\beta(n)$ and $\beta(m)$.

Transport the accepted trace by hsame. The $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ trace rows remain the same deterministic bounded computation after transport, while the enlarged bound only adds unused budget rows.

The $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ bounded-simulation ledger therefore reads the same accepting verdict under the larger budget. Nothing in the argument adds nondeterministic, randomised, oracle, average-case, or separation data. □

Lean4 ✓ BEDC.Derived.ComplexityClassUp.ComplexityClassResourceModulusMonotonicity bound enlargement

Theorem 282.3 (ComplexityClassUp bounded simulation soundness)**.** *Every accepted* $\mathsf{ComplexityClass}^{\uparrow}$ *row has a sound bounded-simulation readback: the* BHist *input, the unary length row, the* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *trace, the unary resource modulus, and the* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *verdict determine the same halted-accepting configuration after* hsame *transport. No unbounded halting predicate or universal-machine theorem is read from this surface.*

Lean4 ✓ BEDC.Derived.ComplexityClassUp.ComplexityClassBoundedSimulation soundness

Proof. Begin with the accepted carrier row. It supplies the input history, unary length, displayed modulus value, deterministic trace, and bounded-simulation verdict.

Use hsame to transport the input and trace rows. The deterministic $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ component fixes each finite transition row, so the transported trace has the same halted endpoint inside the transported budget.

Read the verdict through the $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ ledger. The result is soundness of the bounded accepting simulation only; the carrier contains no coordinate for an unbounded halting decision or a universal-machine simulation theorem. □

Theorem 282.4 (ComplexityClassUp resource ledger exactness)**.** *The* $\mathsf{ComplexityClass}^{\uparrow}$ *resource ledger is exact: for an accepted row, the recorded* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *input length, unary modulus value, realised deterministic* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *trace length, bounded* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *verdict, and* hsame *transport row are precisely the rows used by the carrier. Enlarging a bound uses the monotone modulus row, but the accepted verdict remains the displayed finite* BHist *ledger entry.*

Lean4 ✓ BEDC.Derived.ComplexityClassUp.ComplexityClassResourceLedger membership exactness

Proof. Project the resource ledger from the carrier. Its coordinates are the modulus value, realised trace length, halted verdict, and transport evidence, all over finite BHist rows.

Apply the monotone modulus row only when the budget is enlarged. The underlying deterministic trace and bounded-simulation verdict are not recomputed by an external model; they are the same rows already present in the carrier.

Thus exactness is the equality of the carrier's accepted row with its visible ledger coordinates. There is no additional resource interpretation outside the NatUp, TuringMachineUp, ComputableUp, and hsame surfaces. □

Lean4 ✓ `BEDC.Derived.ComplexityClassUp.ComplexityClassResourceLedgerExactness visible transport`

Theorem 282.5 (ComplexityClassUp NameCert obligation surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{ComplexityClass}^{\uparrow}}$ *obligation surface is obtained by repacking four rows: the bounded acceptance carrier, unary resource-modulus monotonicity, bounded-simulation soundness, and resource-ledger exactness. It depends only on finite* BHist *rows from* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$, *the classifier* hsame*, and the unary resource modulus. It does not claim nondeterminism, randomness, oracle computation, average-case complexity, or* P *versus* NP*.*

Lean4 ✓ `BEDC.Derived.ComplexityClassUp.ComplexityClassNameCertObligationSurface public rows`

Proof. First take the bounded acceptance carrier from Definition 282.1. It supplies the finite input-length, acceptor, bounded trace, modulus, classifier, and verdict rows.

Next add the monotonicity and soundness rows from Definition 282.2 and Definition 282.3. These transport accepted traces by hsame and enlarge resource bounds only through the displayed unary modulus.

Finally use Definition 282.4 to identify the accepted verdict with the visible ledger row. Repacking these four rows gives the public naming-certificate obligation surface, and constructor inversion shows that no nondeterministic, randomised, oracle, average-case, or class-separation coordinate is present. □

Theorem 282.6 (ComplexityClassUp deterministic resource bridge)**.** *The deterministic bounded-resource certificate of* $\mathsf{ComplexityClass}^{\uparrow}$ *bridges to the standard finite resource-class schema exactly through the* $\mathsf{Nat}^{\uparrow}$ *unary input-length row, the* $\mathsf{TuringMachine}^{\uparrow}$ *deterministic trace row, the* $\mathsf{Computable}^{\uparrow}$ *bounded verdict row, the monotone unary resource modulus, and* hsame *transport. The bridge exports no nondeterministic, randomized, oracle, average-case, interactive-proof, or* P*-versus-*NP *row.*

Proof. Begin with the bounded acceptance carrier of Definition 282.1. It supplies the unary length row, deterministic machine trace, bounded simulation verdict, modulus value, and classifier transport needed by the finite resource-class schema.

Next compose resource-modulus monotonicity with bounded simulation soundness. By Definition 282.2 and Definition 282.3, enlarging the unary input bound uses only the monotone modulus row, and the accepted verdict remains the displayed deterministic bounded computation after hsame transport.

Finally apply resource-ledger exactness and the NameCert obligation surface, Definition 282.4 and Definition 282.5. These identify the accepted resource row with the public finite ledger and exclude every nondeterministic, randomized, oracle, average-case, interactive-proof, or class-separation coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComplexityClass}^{\uparrow}}$.

Constructive story: A $\mathsf{ComplexityClass}^{\uparrow}$ token names a deterministic $\mathsf{TuringMachine}^{\uparrow}$ acceptor together with a displayed unary modulus β and a resource-type tag (time, space, or alternation depth). The carrier accepts an input history h when the bounded simulation of the acceptor on h, in the sense of $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$, halts in an accepting configuration within $\beta(|h|)$ trace rows; the resource-type tag selects whether β bounds trace length, head-cell footprint, or alternation depth. The classifier is hsame on input histories of the same unary length; the source covers unary input lengths, modulus values, configuration-trace lengths, head-cell footprints, and alternation rows; the pattern records halted-acceptance, modulus monotonicity, footprint admissibility, and alternation-depth admissibility rows; the ledger keeps the modulus $\beta(|h|)$, the realised trace length or footprint, the alternation depth, and the halting verdict visible. The five name-certificate obligations are supplied by the bounded-simulation certificate of $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ together with monotonicity of β: the inhabited identity certificate is the trivial accepter at $\beta \equiv 0$; classifier stability is reflexive, symmetric, and transitive hsame on inputs of fixed length; transport repacks bounded simulations through input transport; ledger exactness identifies the accepted row with the bounded-simulation verdict; the extension boundary fixes monotonicity of β in the unary input length. A history h lies in the signature gap of a $\mathsf{ComplexityClass}^{\uparrow}$ token p when the source-domain accepts the bounded-resource simulation, the signature relation pairs it with the halting verdict, and token introduction names that verdict as p. The chapter registers this resource-bounded surface under the name $\mathsf{ComplexityClass}^{\uparrow}$; it is a deterministic-resource-class object with no nondeterministic, randomised, average-case, oracle, or interactive-proof bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComplexityClass}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature bounded-resource surface is the finite deterministic schema given by the bounded acceptance carrier, resource modulus monotonicity, bounded simulation soundness, resource ledger exactness, public $\mathsf{NameCert}_{\mathsf{ComplexityClass}^{\uparrow}}$ obligation surface, and deterministic resource bridge of Definition 282.1, Definition 282.2, Definition 282.3, Definition 282.4, Definition 282.5, and Definition 282.6. Nondeterministic, probabilistic, and oracle resources are excluded.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComplexityClass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close P-versus-NP, randomised classes, oracle separations, average-case complexity, or interactive-proof systems.

Upgrade path: A checked standard bridge remains a separate formal-axis target for the deterministic finite resource-class schema.

Chapter 283

A Concrete Naming Certificate for $\mathsf{Kolmogorov}^{\uparrow}$

The Kolmogorov-complexity interface $\mathsf{Kolmogorov}^{\uparrow}$ packages a description-length predicate over a deterministic prefix-free decoder, parametrised by a displayed unary length budget. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ (chapter 280), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

283.1 Carrier definition

A $\mathsf{Kolmogorov}^{\uparrow}$ token carries a deterministic prefix-free decoder $\mathsf{TuringMachine}^{\uparrow}$, a unary length budget ℓ, and a target output history h. The carrier accepts a witness when there is a prefix-free description d of unary length at most ℓ that, decoded by the decoder, halts and reads back h. The classifier is hsame on output histories at fixed budget ℓ; the ledger keeps the description d, the unary budget, the decoded trace, and the readback row visible; the pattern records prefix-freeness, budget admissibility, decoded-readback exactness, and budget monotonicity.

Theorem 283.1 (Kolmogorov prefix-free carrier obligation)**.** *The* $\mathsf{Kolmogorov}^{\uparrow}$ *carrier obligation is the finite* BHist *surface whose accepted row consists of a prefix-free description history, a displayed* $\mathsf{UnaryHistory}$ *length row, a fixed-budget* hsame *classifier on decoded output histories, and the dependency rows supplied by* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$*. At a fixed unary budget* ℓ*, the carrier accepts exactly those description rows whose decoder trace is a displayed halted* $\mathsf{TuringMachine}^{\uparrow}$ *trace consumed by the bounded* $\mathsf{Computable}^{\uparrow}$ *readback.*

This obligation records prefix-freeness and budget admissibility only as visible finite rows. It does not assert universal-decoder optimality, randomness extraction, the invariance theorem, or any standard information-theoretic bridge.

```
Lean4 ✓  BEDC.Derived.KolmogorovUp.KolmogorovPrefixFreeCarrier obligation surface
Lean4 ✓  BEDC.Derived.KolmogorovUp.KolmogorovPrefixFreeCarrierObligation finite surface
```

Proof. Inspect the displayed witness row. Its BHist component supplies the description history, its $\mathsf{UnaryHistory}$ component supplies the length budget, and the fixed-budget hsame row supplies the classifier on decoded output histories.

The $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ dependency reads the unary length and budget rows, the $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ dependency reads the deterministic decoder trace, and the $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ dependency reads

the bounded halted output. Since each component is carried in the same finite ledger, the carrier contains exactly the displayed prefix-free and budget rows and no external optimality or information-theoretic structure. □

Theorem 283.2 (Kolmogorov budget monotonicity obligation)**.** *Let a* $\mathsf{Kolmogorov}^{\uparrow}$ *witness over* BHist *accept a description under a unary budget* ℓ *with fixed-budget* hsame *decoded output classifier. If* ℓ' *is obtained from* ℓ *by a displayed* $\mathsf{UnaryHistory}$ *budget extension, then the same description, the same* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *length row, the same* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *halted trace, and the same* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *bounded readback row form an accepted* $\mathsf{Kolmogorov}^{\uparrow}$ *witness at budget* ℓ'*.*

The transported row preserves the decoded readback and enlarges only the visible unary admissibility bound. It does not introduce universal-decoder optimality, randomness extraction, or a standard information-theoretic bridge.

Lean4 ✓ `BEDC.Derived.KolmogorovUp.KolmogorovBudgetMonotonicity budget transport`

Proof. Start with the accepted witness at budget ℓ. The description history, decoder trace, halted readback, and output classifier are already visible in the BHist ledger.

Apply constructor inspection to the displayed $\mathsf{UnaryHistory}$ extension from ℓ to ℓ'. The length-admissibility row for the old budget embeds into the larger unary budget, while the fixed-budget hsame classifier on the decoded output is reused unchanged at the new displayed budget.

The $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ row witnesses the unary bound, and the $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ rows are not rerun. Therefore the accepted readback is preserved exactly, with only the budget-admissibility coordinate enlarged. □

Lean4 ✓ `BEDC.Derived.KolmogorovUp.KolmogorovBudgetMonotonicity budget extension`

Theorem 283.3 (Kolmogorov decoded readback exactness)**.** *For every accepted* $\mathsf{Kolmogorov}^{\uparrow}$ *witness over* BHist*, the ledger identifies the accepted output row with the displayed* $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ *halted decoder trace as read through the bounded* $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ *verdict, while the unary description and budget are the rows certified by* $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ *and* $\mathsf{UnaryHistory}$*. The* hsame *classifier therefore compares only the decoded output histories at the fixed displayed budget.*

The exactness statement closes only the finite decoded-readback ledger. It does not close universal-decoder optimality, randomness extraction, the invariance theorem, or any standard information-theoretic bridge.

Lean4 ✓ `BEDC.Derived.KolmogorovUp.KolmogorovDecodedReadback exactness`

Proof. Read the accepted witness as a finite ledger. The description and budget rows are the $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ and $\mathsf{UnaryHistory}$ coordinates, and the decoder run is the displayed deterministic $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ trace.

The bounded $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ verdict consumes that halted trace and exports a single readback row. Ledger exactness then identifies the accepted $\mathsf{Kolmogorov}^{\uparrow}$ output with that readback row, so the fixed-budget hsame classifier has no additional coordinate to compare.

The proof never replaces the displayed decoder by an optimal universal decoder, never samples a randomness row, and never invokes an information-theoretic bridge; all equalities are inside the finite decoded trace ledger. □

Theorem 283.4 (Kolmogorov public NameCert export)**.** *The prefix-free carrier obligation, budget monotonicity obligation, and decoded readback exactness assemble the public* $\mathsf{NameCert}_{\mathsf{Kolmogorov}^{\uparrow}}$

export over BHist, UnaryHistory, Nat$^{\uparrow}$, TuringMachine$^{\uparrow}$, Computable$^{\uparrow}$, *the fixed-budget* hsame *classification, and the visible prefix-free decoder ledger.*

The exported public surface consists exactly of the carrier row for accepted prefix-free descriptions, the fixed-budget output classifier, the stability rows for transporting the same decoded output across displayed unary budget extensions, the budget transport row, and the decoded-readback exactness row. It does not assert universal-decoder optimality, randomness extraction, an invariance theorem, resource-bounded complexity, or an information-theoretic standard bridge.

Proof. First project Definition 283.1. This supplies the finite carrier: description histories in BHist, displayed UnaryHistory budgets, $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ length rows, $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ halted decoder traces, and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ bounded readback rows.

Next use Definition 283.2. The budget transport keeps the description, decoder trace, readback row, and fixed-budget hsame classifier stable while only enlarging the visible unary admissibility bound.

Finally apply Definition 283.3. The accepted output row is the decoded readback from the displayed halted trace, so the public certificate repacks the carrier, classifier, stability, budget transport, and exactness rows listed in the statement. No optimal universal decoder, randomness row, or standard information-theoretic bridge is added. □

Closure status for $\mathsf{NameCert}_{\mathsf{Kolmogorov}^{\uparrow}}$.

Constructive story: A Kolmogorov$^{\uparrow}$ token names a deterministic prefix-free decoder TuringMachine$^{\uparrow}$ together with a displayed unary length budget ℓ and a target output history h. The carrier accepts a witness when there is a prefix-free description history d of unary length at most ℓ such that running the decoder on d, in the sense of $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$, halts and reads back h. The classifier is hsame on output histories at fixed unary budget; the source covers unary description lengths, budget values, decoder configurations, prefix-freeness rows, and decoded readback histories; the pattern records prefix-freeness admissibility, budget admissibility, decoded-readback exactness, budget monotonicity, and decoder-determinism rows; the ledger keeps the description d, the budget ℓ, the decoded trace, and the readback row visible. The five name-certificate obligations are supplied by the bounded-simulation certificate plus the prefix-free description structure: the inhabited identity certificate is the empty-description trivial decoder; classifier stability is reflexive, symmetric, and transitive hsame at fixed budget; transport repacks descriptions through prefix-extension; ledger exactness identifies the accepted row with the decoded readback at the displayed budget; the extension boundary fixes budget monotonicity. A history h lies in the signature gap of a Kolmogorov$^{\uparrow}$ token p when the source-domain accepts the prefix-free decoded readback, the signature relation pairs it with the decoded output, and token introduction names that output as p. The chapter registers this prefix-free description-budget surface under the name Kolmogorov$^{\uparrow}$; it is a description-length object with no universal-decoder optimality, no resource-bounded variant (K^t, K^s), no randomness extraction, and no information-theoretic standard bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Kolmogorov}^{\uparrow}}$, publicClosed)

Scope closed: The public surface binds BHist, UnaryHistory, Nat$^{\uparrow}$, TuringMachine$^{\uparrow}$, Computable$^{\uparrow}$, prefix-free description rows, bounded decoder traces, budget monotonicity, fixed-budget hsame output classification, and the exported $\mathsf{NameCert}_{\mathsf{Kolmogorov}^{\uparrow}}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Kolmogorov}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close universal-decoder optimality, resource-bounded Kolmogorov complexity (K^t, K^s), randomness extraction, the invariance theorem, or a standard information-theoretic bridge.

Upgrade path: Bridged closure requires a standard information-theoretic bridge theorem for the exported prefix-free decoder, budget, readback, classifier, and ledger surface.

Chapter 284

A Concrete Naming Certificate for $\mathsf{BusyBeaver}^{\uparrow}$

The busy-beaver interface $\mathsf{BusyBeaver}^{\uparrow}$ packages a finite-state-bounded halting maximum predicate parametrised by a displayed state count and a candidate output bound. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{TuringMachine}^{\uparrow}}$ (chapter 280), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

284.1 Carrier definition

A $\mathsf{BusyBeaver}^{\uparrow}$ token carries a unary state-count n, a finite enumeration of n-state two-symbol $\mathsf{TuringMachine}^{\uparrow}$ codes, a unary candidate output bound B, and a halted-step ledger. The carrier accepts a witness when, for every machine code in the enumeration that halts on the empty tape, its halted output and step count are bounded by B in the sense of $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$. The classifier is hsame on output bounds at fixed unary state count; the ledger keeps the enumeration, the halted bound B, the per-machine readback, and the non-halting witness rows visible.

Theorem 284.1 (BusyBeaver halted-bound admissibility)**.** *For a* $\mathsf{BusyBeaver}^{\uparrow}$ *token with unary state-count* n*, displayed finite enumeration* E *of* n*-state two-symbol* $\mathsf{TuringMachine}^{\uparrow}$ *codes, unary bound* B*, and halted-step ledger* L*, every code* M *in* E *whose empty-input row is recorded in* L *as halted has a* $\mathsf{Computable}^{\uparrow}$ *bounded-trace row in* L*. That row starts from the displayed empty tape, follows the finite* $\mathsf{TuringMachine}^{\uparrow}$ *transition trace for* M*, reads back the halted output row, records the halted step-count row, and compares both rows with* B *through the* $\mathsf{Nat}^{\uparrow}$ *order surface.*

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverHaltedBound admissibility`
Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverHaltedBoundAdmissibility halted trace surface`

Proof. Project the accepted $\mathsf{BusyBeaver}^{\uparrow}$ packet to its finite enumeration and halted-step ledger. Membership of M in E is a displayed finite row, and the halted case is exactly the ledger branch that carries a finite $\mathsf{TuringMachine}^{\uparrow}$ configuration trace from the empty input to a halted row.

Read that halted branch through the $\mathsf{TuringMachine}^{\uparrow}$ finite trace and tape readback rows. They supply the endpoint configuration, output readback, and unary step count without appealing to a host run or to a universal-machine coding theorem.

The carrier clause for $\mathsf{BusyBeaver}^{\uparrow}$ requires this halted branch to be repacked as a $\mathsf{Computable}^{\uparrow}$ bounded trace. The same ledger row supplies the bound comparisons against B in the $\mathsf{Nat}^{\uparrow}$ order, so the accepted witness is admissible exactly for the halted machines already displayed in E. □

Theorem 284.2 (BusyBeaver enumeration completeness)**.** *The finite enumeration surface of a* $\mathsf{BusyBeaver}^{\uparrow}$ *token exhausts the displayed n-state two-symbol* $\mathsf{TuringMachine}^{\uparrow}$ *code rows for its fixed unary state count n. Every admitted machine row is either one of the listed enumeration entries with its corresponding ledger branch, or it is outside the token's carrier surface; no host universal-machine theorem, ambient coding theorem, or external halting search is imported into the enumeration claim.*

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverEnumerationCompleteness finite ledger surface`
Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverEnumerationCompleteness append coverage`

Proof. Unpack the source coordinate of the token. The enumeration is a finite BEDC row whose entries are already typed as n-state two-symbol $\mathsf{TuringMachine}^{\uparrow}$ codes, and the carrier consumes only entries addressed by that row.

For a displayed code row at the same state count and alphabet, enumeration completeness is the finite coverage statement internal to the token: the row is found at one listed position, with the halted or non-halting ledger branch attached there. If no listed position is supplied, the row has no carrier membership witness for this $\mathsf{BusyBeaver}^{\uparrow}$ token.

Thus the proof is finite projection from the enumeration surface, not an argument that a universal simulator can decide which machines halt. The only readback used downstream is the per-entry $\mathsf{TuringMachine}^{\uparrow}$ or $\mathsf{Computable}^{\uparrow}$ ledger row already attached to the finite list. □

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverEnumerationCompleteness finite coverage`

Theorem 284.3 (BusyBeaver bound monotonicity)**.** *Let a* $\mathsf{BusyBeaver}^{\uparrow}$ *token with enumeration E, unary state-count n, bound B, and halted-step ledger L be accepted. If B' is another unary* $\mathsf{Nat}^{\uparrow}$ *bound with $B \leq B'$ in the displayed order surface, then replacing B by B' preserves the accepted witness with the same enumeration and the same halted-step ledger. The transport changes only the bound rows; it does not add a halting-decision row and does not assert computability of the busy-beaver function.*

Proof. Start with an accepted witness at bound B. Each halted entry in the finite enumeration has, by Definition 284.1, a $\mathsf{Computable}^{\uparrow}$ bounded trace whose output and step-count rows are bounded by B.

Transport those two comparisons along the supplied $\mathsf{Nat}^{\uparrow}$ order proof $B \leq B'$. The enumeration row, halted trace, output readback, and step-count ledger are unchanged; only the terminal bound comparison is replaced by the larger unary bound.

Non-halting ledger branches are also unchanged, since they carry no halted output comparison. Repacking the same finite ledger with the transported comparisons yields an accepted $\mathsf{BusyBeaver}^{\uparrow}$ witness at B', with no new halting oracle and no computability claim about the global busy-beaver function. □

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverBoundMonotonicity bound transport`

Theorem 284.4 (BusyBeaver non-halting exclusion obligation)**.** *For a* $\mathsf{BusyBeaver}^{\uparrow}$ *token with unary state-count n, finite enumeration E, unary bound B, and ledger L, a non-halting branch for an entry $M \in E$ is only a visible finite ledger row attached to that enumeration position. It records that the accepted packet has no halted output row for M inside L; it does not supply a halting-decision oracle, a complement enumerator for halting rows, or a method for deciding whether arbitrary n-state* $\mathsf{TuringMachine}^{\uparrow}$ *codes halt.*

Proof. Project the token to the finite enumeration surface of Definition 284.2. Each enumeration position has exactly the ledger branch displayed at that position, so a non-halting row is read as part of the finite packet rather than as the result of a search over machine runs.

The halted branch has a different shape: by Definition 284.1, it carries a finite $\mathsf{TuringMachine}^{\uparrow}$ trace, a halted output row, a step-count row, and $\mathsf{Computable}^{\uparrow}$ bound comparisons. The non-halting branch lacks precisely those halted endpoint fields.

Therefore the row excludes only a halted readback inside this accepted ledger. It gives downstream consumers a visible finite boundary for this token and no procedure for classifying machines outside the displayed branch. □

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverNonhaltingExclusion obligation surface`

Theorem 284.5 (BusyBeaver per-machine readback exactness)**.** *For every halted enumeration entry of an accepted* $\mathsf{BusyBeaver}^{\uparrow}$ *token, the per-machine readback is exactly the bounded* $\mathsf{TuringMachine}^{\uparrow}$ *and* $\mathsf{Computable}^{\uparrow}$ *endpoint already attached to that entry. The readback does not introduce a second simulator, a host computation trace, or a detached output bound; it is the finite trace endpoint, halted output row, step-count row, and B-comparison row carried by the same enumeration position.*

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverPerMachineReadback membership exactness`

Proof. Start from a halted entry M of the finite enumeration. By Definition 284.1, the ledger branch attached to M already contains the $\mathsf{TuringMachine}^{\uparrow}$ transition trace from the empty tape to the halted configuration and the corresponding $\mathsf{Computable}^{\uparrow}$ bounded trace.

The finite projection from the enumeration position supplies the readback key. It points to the same branch of L, so the output, step count, and bound comparison are the endpoint rows already present there.

If the bound is transported as in Definition 284.3, the machine trace and halted readback remain unchanged. Only the terminal $\mathsf{Nat}^{\uparrow}$ comparison is replaced, which preserves exactness of the per-machine endpoint. □

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverPerMachineReadback endpoint exactness`

Theorem 284.6 (BusyBeaver scoped name-certificate surface)**.** *The scoped* $\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$ *surface over a displayed unary state count n, finite enumeration E, unary bound B, and ledger L is exactly the package consisting of halted-bound admissibility, enumeration completeness, bound monotonicity, non-halting exclusion, and per-machine readback exactness. Its public read exposes the enumeration, the halted bound, the halted per-entry* $\mathsf{TuringMachine}^{\uparrow}$ *and* $\mathsf{Computable}^{\uparrow}$ *rows, and the non-halting ledger rows, but it does not expose busy-beaver computability or halting decidability.*

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverHaltingLedger scoped namecert surface`

Proof. First assemble the halted side. Apply Definition 284.1 to obtain the bounded $\mathsf{Computable}^{\uparrow}$ row for each halted entry, and apply Definition 284.5 to identify each consumer read with the endpoint attached to its enumeration position.

Next assemble the enumeration and transport side. Apply Definition 284.2 to keep every admitted machine row inside the displayed finite list, and apply Definition 284.3 to show that increasing the unary bound changes only the terminal comparison rows.

Finally handle the negative branch. Apply Definition 284.4: the non-halting rows are finite ledger projections, not a decision procedure. Repacking these five rows gives the scoped $\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$ surface and preserves the stated non-claims. □

Theorem 284.7 (BusyBeaver per-machine halted-readback determinacy)**.** *Let W and W' be accepted* $\mathsf{BusyBeaver}^{\uparrow}$ *witnesses with the same unary state-count n and the same displayed finite enumeration E of n-state two-symbol* $\mathsf{TuringMachine}^{\uparrow}$ *codes. Let M be one entry of E. Suppose the ledger of W records for M a halted branch from the displayed empty tape with finite transition trace T, halted output row o, and halted step-count row s, and the ledger of W' records for the same entry M a halted branch from the displayed empty tape with finite transition trace T', halted output row o', and halted step-count row s'. Then the carrier ledger classifies the two readbacks componentwise:*

$$T\mathsf{hsame}T', \qquad o\mathsf{hsame}o', \qquad s\mathsf{hsame}s'.$$

Thus, at fixed enumeration entry, the halted readback of an accepted $\mathsf{BusyBeaver}^{\uparrow}$ *witness is determined by the deterministic machine code and the empty-tape initial row, not by the choice of accepted witness packet.*

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverPerMachineReadback halted determinacy`

Proof. Project both accepted witnesses to their common enumeration row. Since the entry is the same displayed $M \in E$, both halted branches use the same finite-state code, the same deterministic transition table, the same blank-tape initial configuration, and the same halting predicate. By Definition 284.5, each public readback is the endpoint already attached to the ledger branch at that enumeration position; no detached simulator or host computation trace is introduced.

Compare the two finite transition traces from the empty-tape initial row. Their zero configurations are classified together because they are the displayed initial configuration of the same machine on the same input. If two compared current configurations are classified by hsame, then Definition 280.9 transports the common transition-table row and classifies the next configurations by hsame again. Induction over the finite trace rows therefore identifies the two deterministic runs up to the first halted row exposed by the halted-step ledger.

It remains to rule out a mismatch of the displayed halted index. Once either trace reaches a halted configuration, the halted-repeat clause of Definition 280.10 says that later accepted trace rows only repeat that halted configuration. The ledger exactness clause of Definition 280.7 gives no second branch choice or fresh transition row after that point. Hence the two halted branches for the same M expose the same halted-step row in the $\mathsf{Nat}^{\uparrow}$ classifier, so $s\mathsf{hsame}s'$.

At that common halted row the configuration traces have already been compared, so the visible tape components and head-position readbacks are compared by the finite readback surface of Definition 280.11. The halted output rows read from those compared endpoints are therefore classified together, $o\mathsf{hsame}o'$. Combining the trace comparison, the halted-step comparison, and the output readback comparison gives $T\mathsf{hsame}T'$, $s\mathsf{hsame}s'$, and $o\mathsf{hsame}o'$ as claimed. □

Lean4 ✓ `BEDC.Derived.BusyBeaverUp.BusyBeaverPerMachineReadback halted readback determinacy`

Theorem 284.8 (BusyBeaver public enumeration-bound export)**.** *For a fixed unary state count and displayed enumeration, the finite enumeration rows, bounded halted-readback rows, non-halting ledger rows, monotone bound transport, and per-machine determinacy assemble a public* $\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$ *export. The public rows are exactly the unary state count, the finite* $\mathsf{TuringMachine}^{\uparrow}$ *enumeration, the unary candidate bound, the halted trace and readback rows, the non-halting branch rows, the* $\mathsf{Nat}^{\uparrow}$ *order comparisons, and the visible* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance.*

The export asserts no computability of the global busy-beaver function, no halting decidability, no asymptotic growth theorem, and no standard recursion-theory bridge.

Proof. Start from Definition 284.6. It packages the fixed unary state count, displayed finite enumeration, candidate unary bound, halted-step ledger, non-halting witness rows, BHist, hsame, Cont, Pkg, and local NameCert data as the scoped certificate surface.

For each displayed enumeration entry, use Definition 284.7. The halted trace, halted output row, and halted step-count row are determined by the displayed machine code and empty-tape initial row, so the public export does not depend on an additional readback choice.

Use Definition 284.3 to transport the unary candidate bound along the displayed $\mathsf{Nat}^{\uparrow}$ order comparisons. Use Definition 284.4 to keep off-table branches recorded as non-halting ledger rows rather than as missing public data.

Combining these rows gives the public $\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$ export for the fixed enumeration and bound. Since the construction only repacks finite enumeration, bounded readback, non-halting, order-comparison, and provenance rows, it does not export global busy-beaver computability, halting decidability, asymptotic growth, or a standard recursion-theory bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$.

Constructive story: A $\mathsf{BusyBeaver}^{\uparrow}$ token names a unary state-count n together with a finite enumeration of n-state two-symbol $\mathsf{TuringMachine}^{\uparrow}$ codes, a candidate unary output bound B, and a halted-step ledger row. The carrier accepts a witness when every code in the enumeration that halts on the empty tape, in the sense of $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$, has its halted output and step count bounded by B; non-halting codes are recorded in the non-halting witness row. The classifier is hsame on output bounds at fixed unary state count; the source covers the unary state count, the finite machine enumeration, the unary output bound, the halted-step ledger, and the non-halting witness; the pattern records halted bound admissibility, finite enumeration completeness, non-halting exclusion, bound monotonicity, and per-machine-readback exactness rows; the ledger keeps the enumeration, the halted bound B, the per-machine readback, and the non-halting witness rows visible. The five name-certificate obligations are supplied by the bounded-simulation certificate plus enumeration-finiteness: the inhabited identity certificate is the trivial $n = 0$ certificate; classifier stability is reflexive, symmetric, and transitive hsame at fixed state count; transport repacks bounded simulations through machine enumeration; ledger exactness identifies the accepted row with the displayed halted bound; the extension boundary fixes bound monotonicity in the unary state count. A history h lies in the signature gap of a $\mathsf{BusyBeaver}^{\uparrow}$ token p when the source-domain accepts the enumeration-bounded readback, the signature relation pairs it with the halted maximum, and token introduction names that maximum as p. The chapter registers this enumeration-bounded halting surface under the name $\mathsf{BusyBeaver}^{\uparrow}$; it is a finite-state-bounded maximum object with no actual computability claim about the busy-beaver function, no asymptotic growth-rate claim, and no halting-decidability bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$, publicClosed)

Scope closed: The public surface binds the unary state count n, the finite enumeration of n-state two-symbol machines, halted-bound admissibility, enumeration completeness, bound monotonicity, non-halting exclusion, per-machine readback exactness, halted-readback determinacy, the halted-step ledger, and the non-halting witness rows through Definition 284.6 and Definition 284.8.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BusyBeaver}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close busy-beaver-function computability, asymptotic growth-rate theorems, halting decidability, or a standard recursion-theory bridge.

Upgrade path: Standard bridge closure requires a recursion-theory bridge for the same finite enumeration and bounded halted-readback surface.

Chapter 285

A Concrete Naming Certificate for $\mathsf{AbelRuffini}^{\uparrow}$

The Abel-Ruffini interface $\mathsf{AbelRuffini}^{\uparrow}$ packages a non-solvability-by-radicals certificate for a degree-five polynomial over a $\mathsf{NumField}^{\uparrow}$ base. It depends on $\mathsf{NameCert}_{\mathsf{Poly}^{\uparrow}}$ (chapter 50), $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ (chapter 164), $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$ (chapter 117), and $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168).

285.1 Carrier definition

An $\mathsf{AbelRuffini}^{\uparrow}$ token carries: a separable monic polynomial P of degree five over a $\mathsf{NumField}^{\uparrow}$ base K together with three irreducibility, non-trivial-rational-root, and Galois-group-isomorphic-to-S_5 rows. The carrier accepts a non-solvability witness when $\mathsf{Gal}(P/K)$ is isomorphic to S_5 and the derived series of S_5 does not reach the trivial subgroup. The classifier is hsame on the Galois group rows; the ledger keeps the irreducibility witness, the S_5 identification, and the derived-series ledger visible.

Theorem 285.1 (AbelRuffini polynomial Galois obligation)**.** *The* $\mathsf{AbelRuffini}^{\uparrow}$ *source rows are the* $\mathsf{Poly}^{\uparrow}$, $\mathsf{NumField}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, *and* $\mathsf{SymGroup}^{\uparrow}$ BHist *packages for a displayed degree-five polynomial, its base field row, its splitting-field row, and its* S_5 *identification. The carrier reads the polynomial coefficient history, separability and irreducibility rows, the non-trivial-rational-root row, the Galois group package over the splitting field, and the symmetric-group identification row.*

Proof. Project the accepted token to its polynomial and base-field coordinates. The coefficient history is supplied by $\mathsf{Poly}^{\uparrow}$, while the base and splitting field data are supplied by $\mathsf{NumField}^{\uparrow}$ rows.

Next project the group coordinates. The Galois action row is a $\mathsf{GaloisGroup}^{\uparrow}$ package, and the displayed comparison with S_5 is the consumed $\mathsf{SymGroup}^{\uparrow}$ identification.

The carrier has no additional source coordinate for a host field extension or an external classification theorem, so these four package families are exactly the polynomial Galois source surface. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniPolynomialGaloisObligation source surface`

Theorem 285.2 (AbelRuffini derived-series ledger obligation)**.** *The* $\mathsf{AbelRuffini}^{\uparrow}$ *derived-series ledger is a finite* BHist *ledger. Each step exposes the current subgroup row, the next commutator subgroup row, and the* hsame *transport relating the displayed* S_5 *identification to the Galois-group package. The ledger does not import a host solvable-group theorem.*

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniDerivedSeriesLedger obstruction endpoint boundary`

Proof. Read the derived-series field of the accepted package as a finite list of displayed subgroup rows. Each row is a BEDC history, and each successor row is recorded by the displayed commutator-step ledger.

Transport is internal to the same list. The hsame classifier relates the Galois identification row to the corresponding S_5 subgroup row, then carries the step witness to the next displayed subgroup.

Since every step is recorded as an explicit finite ledger row, non-solvability is read from this ledger rather than from an imported abstract solvability theorem. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniDerivedSeriesLedger assoc surface`

Theorem 285.3 (AbelRuffini derived-series finite transport)**.** *A displayed Galois row, S_5 row, subgroup row, commutator row, obstruction row, and continuation chain repack into a finite* $\mathsf{AbelRuffini}^{\uparrow}$ *BEDC transport ledger. The ledger has a unary next row for each displayed commutator step and an endpoint row for the obstruction, with* hsame *comparisons for the subgroup row, the unary next row, and the endpoint row.*

Proof. Start with the finite derived-series ledger from Definition 285.2. It supplies the displayed subgroup rows, the commutator successor rows, and the obstruction row as finite BHist data.

Project unary closure through the three continuation rows: first from the Galois row to the displayed S_5 row, then from the current subgroup row to the unary next commutator row, and finally from the obstruction row to the endpoint row.

The three hsame comparisons are read from these same projections. Repacking the subgroup comparison, next-row comparison, and endpoint comparison gives the finite transport ledger without importing a separate radical-tower or solvable-group theorem. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniDerivedSeriesLedger finite transport`

Theorem 285.4 (AbelRuffini non-radical boundary obligation)**.** *The* $\mathsf{AbelRuffini}^{\uparrow}$ *non-solvability-by-radicals boundary is read only from the polynomial and Galois identification rows together with the finite derived-series ledger. It excludes general degree-n classification, an A_5 simplicity proof, and radical-tower bridge data.*

Proof. Start with the polynomial Galois obligation. It supplies the displayed degree-five polynomial, the splitting-field row, the Galois-group row, and the S_5 identification.

Then apply the derived-series ledger obligation. The finite subgroup ledger records the obstruction row that prevents the displayed series from reaching the trivial subgroup.

The boundary statement is therefore only a readback from these rows. No field of the package names a degree-n classification, an independent A_5 simplicity argument, or a standard radical-tower bridge, so those data remain outside the chapter scope. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniNonradicalBoundary obstruction surface`
↪ variant `BEDC.Derived.AbelRuffiniUp.AbelRuffiniNonradicalBoundary obligation`

Theorem 285.5 (AbelRuffini public certificate export)**.** *The polynomial Galois obligation, derived-series ledger obligation, and non-radical boundary obligation repack into the public* $\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}$ *certificate over the displayed* $\mathsf{Poly}^{\uparrow}$, $\mathsf{NumField}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{SymGroup}^{\uparrow}$, hsame, *and finite derived-series rows. The exported certificate exposes the degree-five polynomial source, base-field row, Galois-group row, S_5 identification, derived-series ledger, and non-solvability-by-radicals boundary,*

while keeping general degree-n Galois solvability, an A_5 simplicity proof, transcendental methods, abstract solvable-group theory, and a standard radical-tower bridge outside its scope.

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniPublicCertificate export`

Proof. First apply Definition 285.1. This supplies the displayed polynomial, number-field base, splitting-field and Galois-group rows, and the consumed $\mathsf{SymGroup}^{\uparrow}$ comparison with S_5.

Next apply Definition 285.2. The derived series is a finite BEDC ledger whose rows are transported through the same hsame comparison, so the public certificate reads only the displayed subgroup steps.

Finally apply Definition 285.4. It identifies the non-solvability boundary as a readback from the polynomial, Galois identification, and finite derived-series rows. Repacking these three obligations yields the public $\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}$ export and preserves all stated non-claims. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniPublicCertificateExport namecert surface`

Theorem 285.6 (AbelRuffini standard radical-tower bridge boundary)**.** *The standard radical-tower bridge for the displayed degree-five* $\mathsf{AbelRuffini}^{\uparrow}$ *certificate may read only the* $\mathsf{Poly}^{\uparrow}$ *coefficient rows, the* $\mathsf{NumField}^{\uparrow}$ *base row, the* $\mathsf{GaloisGroup}^{\uparrow}$ *and* $\mathsf{SymGroup}^{\uparrow}$ *S_5 identification rows, the finite derived-series obstruction ledger, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the* NameCert *rows already exported by Definition 285.5. It adds no general degree-n solvability theorem, no A_5 simplicity proof, no resolvent reduction, and no transcendental method.*

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniStandardRadicalTower bridge boundary`
↪ variant `BEDC.Derived.AbelRuffiniUp.AbelRuffiniPublicCertificate radical tower bridge boundary`

Proof. Project the public certificate export through Definition 285.5. This exposes the coefficient rows, base-field row, Galois-group row, S_5 comparison, derived-series ledger, transport routes, package provenance, and local naming rows as the whole visible packet.

Use Definition 285.1 for the polynomial-Galois coordinates and Definition 285.2 for the finite derived-series obstruction. These obligations account for every row the radical-tower bridge is allowed to consume.

Finally repack the same finite rows as the bridge boundary. Since the packet has no coordinate for a general degree-n theorem, an A_5 simplicity proof, a resolvent reduction, or a transcendental method, none of those objects can be exported by the bridge. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniRadicalTowerBridge standard boundary`

Theorem 285.7 (AbelRuffini degree-five consumer examples)**.** *Finite downstream consumers for polynomial coefficient readback, Galois-group identification, derived-series obstruction, and radical-tower exclusion all read the same displayed degree-five packet exported by the public certificate and the standard radical-tower bridge boundary. The shared packet consists of the* $\mathsf{Poly}^{\uparrow}$ *coefficient rows,* $\mathsf{NumField}^{\uparrow}$ *base row,* $\mathsf{GaloisGroup}^{\uparrow}$ *row,* $\mathsf{SymGroup}^{\uparrow}$ *S_5 identification row, finite derived-series ledger, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

These consumers do not assert a general degree-n solvability theorem, an A_5 simplicity proof, a resolvent reduction, transcendental methods, or machine-checked radical-tower bridge evidence.

Proof. Apply Definition 285.5. It exposes the degree-five polynomial source, base-field row, Galois-group row, S_5 identification, derived-series ledger, non-radical boundary, transports, continuations, package provenance, and local naming rows as one public packet.

The polynomial coefficient and Galois-identification consumers read the polynomial-Galois rows supplied by Definition 285.1. The obstruction consumer reads the finite derived-series rows supplied by Definition 285.2.

Finally apply Definition 285.6. It identifies the radical-tower exclusion consumer as a read of the same public packet rather than as a new general solvability surface. Since the packet has no coordinate for the excluded objects, the four consumers add none of them. □

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniDegreeFiveConsumerExamples`

Theorem 285.8 (AbelRuffini finite consumer completeness)**.** *Every finite downstream consumer of the displayed degree-five non-radical boundary reads only the* $\mathsf{Poly}^{\uparrow}$ *coefficient rows,* $\mathsf{NumField}^{\uparrow}$ *base row,* $\mathsf{GaloisGroup}^{\uparrow}$ *and* $\mathsf{SymGroup}^{\uparrow}$ S_5 *rows, finite derived-series obstruction ledger, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}$ *provenance. Conversely, those displayed rows repack as the same finite consumer surface.*

The consumer surface contains no general degree-n solvability theorem, A_5 simplicity proof, resolvent reduction, transcendental method, or hidden radical-tower assumption.

Lean4 ✓ `BEDC.Derived.AbelRuffiniUp.AbelRuffiniFiniteConsumer completeness`

Proof. Project the polynomial and Galois rows through Definition 285.1. This supplies the degree-five coefficient rows, the number-field base row, the Galois-group row, and the displayed S_5 comparison through $\mathsf{SymGroup}^{\uparrow}$.

Next apply Definition 285.2. The finite obstruction is read as the displayed derived-series ledger, transported through the same hsame rows and package provenance.

Use Definition 285.5 to package these polynomial, base-field, group, obstruction, transport, continuation, package, and naming rows as the public $\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}$ surface.

For the converse direction, start from exactly those displayed rows and repack them by the same public export. Then apply Definition 285.6; it identifies the finite downstream radical-tower consumer as a read of that surface and not as an additional algebraic assumption.

Since no cited row names a general degree-n theorem, an A_5 simplicity proof, a resolvent reduction, a transcendental method, or a hidden radical-tower assumption, every finite consumer is exhausted by the displayed surface and the displayed surface repacks every such consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelRuffini}^{\uparrow}$ token names a separable monic polynomial P of degree five over a $\mathsf{NumField}^{\uparrow}$ base K together with three witnesses: an irreducibility row, a non-trivial-rational-root row, and a Galois-group identification row $\mathsf{Gal}(P/K) \cong S_5$. The carrier accepts a non-solvability-by-radicals certificate when $\mathsf{Gal}(P/K)$ is isomorphic to S_5 in the sense of $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$, and the derived series of S_5 stops at A_5 without reaching the trivial subgroup. The classifier is hsame on the Galois group identification rows; the source covers polynomial coefficient histories, splitting field carrier rows, Galois group elements, derived-series subgroups, and rational-root exclusion rows; the pattern records irreducibility admissibility, S_5 identification, derived-series exactness, non-radical-tower exclusion, and degree-five admissibility rows; the ledger keeps the polynomial coefficients, the irreducibility witness, the S_5 identification, and the derived-series ledger visible. The five name-certificate obligations are supplied by the $\mathsf{GaloisGroup}^{\uparrow}$ chain plus solvable-group structure: the inhabited identity certificate is a fixed polynomial $x^5 - x - 1$ over $\mathbb{Q}$; classifier stability is reflexive, symmetric, and transitive hsame on Galois group identifications; transport repacks Galois actions through polynomial transport; ledger exactness identifies the carrier row with the S_5 derived-series ledger; the extension boundary fixes the non-radical-tower row. A history h lies in the signature gap of an $\mathsf{AbelRuffini}^{\uparrow}$ token p when the source-domain accepts the polynomial-coefficient history, the signature relation pairs it with the non-solvable Galois identification, and token introduction names that identification as p. The chapter registers this finite Galois-group-based non-solvability surface under the name $\mathsf{AbelRuffini}^{\uparrow}$; it is a degree-five S_5-Galois object whose standard radical-tower bridge boundary reads only the displayed finite rows and does not add general degree-n classification, an A_5 simplicity proof, resolvent reduction, or transcendental method.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridged surface binds the displayed $\mathsf{Poly}^{\uparrow}$, $\mathsf{NumField}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{SymGroup}^{\uparrow}$, hsame, and finite derived-series ledger rows through Definition 285.5, the standard radical-tower bridge boundary of Definition 285.6, the degree-five consumer examples of Definition 285.7, and the finite consumer completeness theorem of Definition 285.8.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelRuffini}^{\uparrow}}, \text{unformalized})$

Bridge status: paperBridge

Not claimed: The chapter does not close general degree-n Galois solvability, the A_5 simplicity proof, transcendental methods, abstract solvable-group theory, or machine-checked radical-tower bridge evidence.

Upgrade path: No paper-side theory grade remains for this chapter; formal bridge verification remains on the separate verification axis.

Chapter 286

A Concrete Naming Certificate for $\mathsf{SolvableRadicals}^{\uparrow}$

The radical-tower interface $\mathsf{SolvableRadicals}^{\uparrow}$ packages a tower of radical extensions $K = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_r$ where each $K_{i+1} = K_i(\alpha_i)$ with $\alpha_i^{n_i} \in K_i$. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161), $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ (chapter 164), and $\mathsf{NameCert}_{\mathsf{NumField}^{\uparrow}}$ (chapter 168).

286.1 Carrier definition

A $\mathsf{SolvableRadicals}^{\uparrow}$ token carries a finite tower depth r, a $\mathsf{Seq}^{\uparrow}$-typed sequence of intermediate field rows $K_0, \ldots, K_r$, a sequence of root indices $n_0, \ldots, n_{r-1}$, and a sequence of radicand witnesses α_i with $\alpha_i^{n_i} \in K_i$. The carrier accepts a polynomial P when P splits in K_r and each step is genuinely radical. The classifier is hsame on the tower depth and intermediate-field rows; the ledger keeps the depth r, the root indices, the radicand witnesses, and the splitting verdict visible.

Theorem 286.1 (SolvableRadicals tower-step obligation)**.** *Every accepted* $\mathsf{SolvableRadicals}^{\uparrow}$ *radical-tower row exposes the finite* $\mathsf{Seq}^{\uparrow}$ *tower depth, the root-index rows, the radicand witnesses, and the* $\mathsf{FieldExt}^{\uparrow}$ *radical-step certificate for each adjacent pair of intermediate field rows. For each displayed step* $K_i \subseteq K_{i+1}$*, the ledger records the row* $\alpha_i^{n_i} \in K_i$ *and the corresponding extension row for* $K_{i+1} = K_i(\alpha_i)$*.*

The obligation is only the finite tower-step surface. It does not assert a Galois-correspondence theorem, abstract solvable-group classification, or a radical-closure bridge.

Lean4 ✓ `BEDC.Derived.SolvableRadicalsUp.SolvableRadicalsTowerStepLedger assoc surface`

Proof. Project the accepted packet to its tower-depth coordinate and intermediate field sequence. These rows give the finite $\mathsf{Seq}^{\uparrow}$ spine $K_0, \ldots, K_r$.

For each adjacent pair, read the root-index and radicand coordinates. The $\mathsf{FieldExt}^{\uparrow}$ certificate attached to that pair records the radical-step witness and the equation $\alpha_i^{n_i} \in K_i$ in the same ledger.

Repacking the displayed adjacent-step rows gives the tower-step obligation. No classification of solvable groups or radical closures is invoked. □

Lean4 ✓ `BEDC.Derived.SolvableRadicalsUp.SolvableRadicalsTowerStep obligation`

Theorem 286.2 (SolvableRadicals intermediate-field stability)**.** *For* $\mathsf{SolvableRadicals}^{\uparrow}$*,* hsame *transport of fixed-depth intermediate field rows preserves the displayed radical-step ledger. If two tower*

presentations have the same finite depth and their corresponding intermediate field rows are related by the fixed classifier, then each root-index, radicand, and $\mathsf{FieldExt}^{\uparrow}$ *radical-step certificate transports to the matching adjacent field row.*

The statement is a fixed-depth stability obligation. It does not identify abstract Galois groups or claim a full Galois correspondence.

Proof. Start with two accepted tower rows of the same displayed depth. The classifier relates corresponding intermediate field rows by hsame and keeps the finite index positions fixed.

At each adjacent position, transport the root-index, radicand, and $\mathsf{FieldExt}^{\uparrow}$ radical-step certificate along the corresponding field-row classifiers. Since the positions are fixed by the tower depth, no new step is created and no step is omitted.

The transported packet therefore has the same radical-step ledger at the matching positions. The argument is finite ledger transport rather than a Galois-correspondence or solvable-group classification result. □

Lean4 ✓ `BEDC.Derived.SolvableRadicalsUp.SolvableRadicalsTowerPacket intermediate field stability`

Theorem 286.3 (SolvableRadicals splitting ledger exactness)**.** *In an accepted* $\mathsf{SolvableRadicals}^{\uparrow}$ *packet, the accepted polynomial row is identified with the displayed splitting verdict in the top field* K_r*. The ledger reads the polynomial over* K_0*, transports it through the finite tower, and records the top-field factor rows exactly as the carrier's splitting verdict.*

This exactness statement does not assert a Galois-correspondence theorem, abstract solvable-group classification, transcendental methods, or a standard radical-closure bridge.

Proof. Read the polynomial coordinate over the base field row K_0 and the finite tower coordinate ending at K_r. Both are part of the accepted packet.

Next inspect the splitting verdict. Its factor rows live in the displayed top field, and the ledger ties them to the original polynomial row through the finite tower already exposed by the carrier.

Thus the accepted polynomial row and the top-field splitting verdict are the same ledger surface. No additional correspondence theorem, solvable-group classification, or transcendental method is projected from the packet. □

Lean4 ✓ `BEDC.Derived.SolvableRadicalsUp.SolvableRadicalsTowerPacket splitting ledger exactness`

Theorem 286.4 (SolvableRadicals public NameCert export)**.** *The tower-step obligation, intermediate-field stability row, and splitting-ledger exactness row assemble the public* $\mathsf{NameCert}_{\mathsf{SolvableRadicals}^{\uparrow}}$ *export consumed by* $\mathsf{AbelRuffini}^{\uparrow}$ *and Galois-side users. The export contains only finite tower packets, displayed* $\mathsf{FieldExt}^{\uparrow}$ *source rows, radical-step ledgers,* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

This public surface does not claim the Galois-correspondence theorem, abstract solvable-group classification, transcendental methods, or a solvability-by-radicals standard schema.

Proof. Project Definition 286.1. It supplies the finite tower depth, adjacent intermediate-field rows, root-index rows, radicand witnesses, and displayed $\mathsf{FieldExt}^{\uparrow}$ radical-step certificates.

Use Definition 286.2 to transport the same fixed-depth tower surface through the visible hsame rows. The transport keeps the radical-step ledger attached to the matching adjacent field rows and does not add a group-classification coordinate.

Apply Definition 286.3 to identify the accepted polynomial row with the top-field splitting verdict in the same finite tower ledger.

Repacking the tower-step, stability, and splitting rows gives the exported $\mathsf{NameCert}_{\mathsf{SolvableRadicals}^{\uparrow}}$ surface used by the downstream polynomial consumers. The repacked packet contains only the fields listed in the statement and leaves the standard solvability-by-radicals schema outside the chapter boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SolvableRadicals}^{\uparrow}}$.

Constructive story: A $\mathsf{SolvableRadicals}^{\uparrow}$ token names a finite tower depth r, a $\mathsf{Seq}^{\uparrow}$-typed sequence of intermediate field rows $K_0, \ldots, K_r$, a sequence of root indices $n_0, \ldots, n_{r-1}$, and a sequence of radicand witnesses $\alpha_i \in K_{i+1}$ with $\alpha_i^{n_i} \in K_i$. The carrier accepts a separable polynomial P over K_0 when each step $K_{i+1} = K_i(\alpha_i)$ is genuinely radical in the sense of $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ and P splits in K_r. The classifier is hsame on the tower depth and intermediate-field rows; the source covers field carriers, root indices, radicand witnesses, splitting witnesses, and tower-depth rows; the pattern records radical-step admissibility, splitting admissibility, tower-depth admissibility, root-of-unity availability, and intermediate-field-stability rows; the ledger keeps the depth r, the root indices, the radicand witnesses, and the splitting verdict visible. The five name-certificate obligations are supplied by the $\mathsf{FieldExt}^{\uparrow}$ tower together with $\mathsf{GaloisGroup}^{\uparrow}$ solvability: the inhabited identity certificate is the trivial $r = 0$ tower; classifier stability is reflexive, symmetric, and transitive hsame on towers of fixed depth; transport repacks radical extensions through field-isomorphism transport; ledger exactness identifies the accepted polynomial with its splitting in K_r; the extension boundary fixes the radical-step rule. A history h lies in the signature gap of a $\mathsf{SolvableRadicals}^{\uparrow}$ token p when the source-domain accepts the radical-tower-splitting witness, the signature relation pairs it with the splitting verdict, and token introduction names that verdict as p. The chapter registers this radical-tower surface under the name $\mathsf{SolvableRadicals}^{\uparrow}$; it is a finite-tower polynomial-splitting object with no Galois-correspondence claim, no general solvable-group identification, no transcendental methods, and no standard radical-closure bridge.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SolvableRadicals}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The obligation scope binds the radical tower-step rows, fixed-depth intermediate-field stability under hsame, and top-field splitting ledger exactness over the displayed $\mathsf{Seq}^{\uparrow}$, $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{NumField}^{\uparrow}$, and polynomial rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SolvableRadicals}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SolvableRadicalsUp.SolvableRadicalsTowerPacket\_intermediate\_field\_stability`

Bridge status: none

Not claimed: The chapter does not close the Galois-correspondence theorem, abstract solvable-group classification, transcendental methods, or a standard radical-closure bridge.

Upgrade path: Bridged closure requires a bridge from the exported radical-tower splitting surface to the standard solvability-by-radicals schema.

Chapter 287

A Concrete Naming Certificate for $\mathsf{DiffGalois}^{\uparrow}$

The differential-Galois interface $\mathsf{DiffGalois}^{\uparrow}$ packages the Picard-Vessiot Galois group of a linear differential equation over a $\mathsf{NumField}^{\uparrow}$-base differential field. It depends on $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ (chapter 161), $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ (chapter 164), $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$ (chapter 193), and $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ (chapter 140).

287.1 Carrier definition

A $\mathsf{DiffGalois}^{\uparrow}$ token carries: a differential field (K, ∂) over a $\mathsf{NumField}^{\uparrow}$ subfield, a linear differential operator $L = \sum a_i \partial^i$ with coefficients in K, a $\mathsf{Seq}^{\uparrow}$-typed solution-space row V of dimension equal to the order of L, and the differential-Galois group G acting on V that fixes K^{∂} pointwise. The carrier accepts a Picard-Vessiot certificate when the solution space exists, the differential-Galois group G acts faithfully on V, and the constants K^{∂} are exactly the G-fixed elements. The classifier is hsame on the solution-space rows; the ledger keeps the differential operator, the solution-space row, the Galois action, and the constant-field witness visible.

Theorem 287.1 (DiffGalois Picard-Vessiot obligation)**.** *An accepted* $\mathsf{DiffGalois}^{\uparrow}$ *Picard-Vessiot row exposes the solution-space, differential-operator, and fundamental-solution rows as finite* BHist *data over* $\mathsf{Seq}^{\uparrow}$, $\mathsf{FieldExt}^{\uparrow}$, *and* $\mathsf{Ode}^{\uparrow}$. *The differential field* (K, ∂) *is read through the displayed* $\mathsf{FieldExt}^{\uparrow}$ *row, the coefficient rows of the linear operator* $L = \sum a_i \partial^i$ *are read through the displayed* $\mathsf{Ode}^{\uparrow}$ *row, and the solution-space basis and fundamental-solution matrix are read through the* $\mathsf{Seq}^{\uparrow}$ *row.*

The obligation is local to the displayed Picard-Vessiot ledger. It does not assert algebraic-group classification, Liouvillian solvability, or an integration-by-quadratures bridge.

Proof. Project the accepted packet to its differential-field and operator coordinates. These rows provide (K, ∂) and the coefficient list of L as finite source data.

Next read the solution-space coordinate. The $\mathsf{Seq}^{\uparrow}$ basis row and the fundamental-solution row are the only witnesses consumed by the carrier, and the classifier compares solution-space histories by the fixed hsame row.

Repacking these coordinates gives the stated obligation surface. No step adds an algebraic-group classification row, a Liouvillian criterion, or an external analytic solution method. □

Lean4 ✓ `BEDC.Derived.DiffGaloisUp.DiffGaloisPicardVessiotPacket solution space obligation`

Theorem 287.2 (DiffGalois faithful action and constant-field exactness)**.** *For an accepted* $\mathsf{DiffGalois}^{\uparrow}$ *packet, the displayed* $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ *action row is faithful on the visible solution-space basis and transports along the fixed solution-space classifier. Two action histories that agree on every displayed basis row are identified by the same* hsame *solution-space classifier. The constant-field row is exactly the fixed-point ledger for this action: a displayed row is accepted as a constant precisely when the ledger records it as fixed, and every displayed fixed row is read back through the same* K^{∂} *coordinate.*

This obligation is a finite action-and-fixed-row statement. It does not classify differential-Galois groups, assert Liouvillian solvability, or supply a standard model-theoretic bridge.

Lean4 ✓ `BEDC.Derived.DiffGaloisUp.DiffGaloisFaithfulActionLedger basis fold surface`

Proof. Start from the action coordinate of the accepted packet. It is already typed as a $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ row acting on the visible solution-space basis.

Faithfulness is read at the displayed basis rows: if two action histories have the same effect on each basis coordinate, the ledger identifies their action rows through the fixed classifier. Transport along hsame changes only the solution-space presentation and keeps the action row attached to the same basis surface.

Now read the constant-field coordinate from the same packet. A displayed constant row is fixed by the action ledger, and a displayed fixed row is read back through K^{∂}, so exactness is an hsame transport comparison between two visible projections of the same Picard-Vessiot data.

Thus the obligation is witnessed by finite projection from the accepted Picard-Vessiot packet rather than by a classification theorem for the ambient group. □

Theorem 287.3 (DiffGalois public certificate export)**.** *The Picard-Vessiot obligation and the faithful-action constant-field exactness obligation repack into the public* $\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$ *boundary. The exported certificate exposes the* $\mathsf{FieldExt}^{\uparrow}$ *differential-field row, the* $\mathsf{Ode}^{\uparrow}$ *operator row, the* $\mathsf{Seq}^{\uparrow}$ *solution-space and fundamental-solution rows, the faithful* $\mathsf{GaloisGroup}^{\uparrow}$ *action row, and the constant-field fixed-row ledger. It excludes Liouvillian solvability, algebraic-group classification, integration-by-quadratures, and standard model-theoretic bridge claims.*

Lean4 ✓ `BEDC.Derived.DiffGaloisUp.DiffGaloisPublicCertificate export`

Proof. Apply Definition 287.1. This supplies the differential-field, operator, solution-space, and fundamental-solution rows as the visible Picard-Vessiot ledger.

Next apply Definition 287.2. This adds the faithful action row and identifies the constant-field coordinate with the fixed-row ledger by hsame transport.

Repacking these rows gives the public $\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$ boundary. No field of the export supplies a Liouvillian criterion, an algebraic-group classification theorem, integration-by-quadratures, or a standard model-theoretic bridge. □

Theorem 287.4 (DiffGalois scoped certificate surface)**.** *The Picard-Vessiot packet, faithful action ledger, constant-field fixed-point exactness, solution-space row, and displayed Galois action rows form the closed scoped* $\mathsf{DiffGalois}^{\uparrow}$ *certificate surface over* $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, BHist, hsame, Cont, *and* Pkg*. Its carrier is the displayed differential-field, operator, solution-space, fundamental-solution, action, and constant-field ledger; its classifier is* hsame *on the solution-space row together with transported Galois action rows; and its visible ledger is exactly the public* $\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$ *boundary of Definition 287.3.*

This scoped surface closes only the finite Picard-Vessiot certificate rows. It does not add algebraic-group classification, Liouvillian solvability, integration-by-quadratures, or a standard model-theoretic differential-Galois bridge.

Lean4 ✓ `BEDC.Derived.DiffGaloisUp.DiffGaloisPicardVessiotPacket scoped certificate surface`

Proof. Start with Definition 287.1. It supplies the differential-field row, operator row, solution-space row, and fundamental-solution row as finite BHist data over the displayed $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, and $\mathsf{Seq}^{\uparrow}$ sources.

Next apply Definition 287.2. This adds the faithful $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ action ledger, transports it through the solution-space classifier, and identifies the constant-field coordinate with the fixed-point ledger.

Finally use Definition 287.3. It repacks the Picard-Vessiot rows and faithful-action rows into the public $\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$ boundary. Reading that export as a scoped carrier/classifier/ledger package gives the stated surface over $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, BHist, hsame, Cont, and Pkg, with no additional classification, solvability, quadrature, or model-theoretic bridge row. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$.

Constructive story: A $\mathsf{DiffGalois}^{\uparrow}$ token names a differential field (K, ∂) over a $\mathsf{NumField}^{\uparrow}$ subfield, a linear differential operator $L = \sum a_i \partial^i$ with coefficients in K, a $\mathsf{Seq}^{\uparrow}$-typed solution-space row V of dimension equal to the order of L, and the differential-Galois group G acting on V and fixing the constants K^{∂} pointwise. The carrier accepts a Picard-Vessiot certificate when V is closed under ∂ and contains a fundamental solution row, G acts faithfully on V in the sense of $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$, and $K^{\partial} = V^G$ in the sense that the G-fixed solutions are exactly the constants. The classifier is hsame on the solution-space rows; the source covers differential-field carrier rows, differential-operator coefficients, solution-space basis rows, Galois action rows, and constant-field rows; the pattern records solution-space admissibility, faithful-action admissibility, constant-field exactness, fundamental-solution row, and operator-order admissibility rows; the ledger keeps the differential operator, the solution-space basis, the Galois action, and the constant-field witness visible. The five name-certificate obligations are supplied by $\mathsf{NameCert}_{\mathsf{FieldExt}^{\uparrow}}$ together with $\mathsf{NameCert}_{\mathsf{GaloisGroup}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{Ode}^{\uparrow}}$: the inhabited identity certificate is the order-zero operator; classifier stability is reflexive, symmetric, and transitive hsame on solution-space rows; transport repacks Galois actions through field-isomorphism transport; ledger exactness identifies the accepted certificate with the solution-space basis; the extension boundary fixes the constant-field-fixed-points equation. A history h lies in the signature gap of a $\mathsf{DiffGalois}^{\uparrow}$ token p when the source-domain accepts the Picard-Vessiot certificate, the signature relation pairs it with the differential-Galois group identification, and token introduction names that group as p. The chapter registers this Picard-Vessiot Galois surface under the name $\mathsf{DiffGalois}^{\uparrow}$; it is a finite-order linear-operator object with no algebraic-group classification, no Liouvillian solvability theorem, no integration-by-quadratures bridge, and no standard model-theoretic differential-Galois bridge.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$, publicClosed)

Scope closed: The scoped certificate surface is closed over $\mathsf{FieldExt}^{\uparrow}$, $\mathsf{GaloisGroup}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, BHist, hsame, Cont, and Pkg through the displayed Picard-Vessiot ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiffGalois}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DiffGaloisUp.DiffGaloisPublicCertificate\_export`

Bridge status: none

Not claimed: The chapter does not close algebraic-group classification of differential-Galois groups, Liouvillian solvability theorems, integration-by-quadratures, or a standard model-theoretic differential-Galois bridge.

Upgrade path: A standard bridge would have to identify this public $\mathsf{DiffGalois}^{\uparrow}$ certificate export with the acceptance-bridge schema without adding algebraic-group classification, solvability, quadrature, or model-theoretic bridge data.

Chapter 288

A Concrete Naming Certificate for $\mathsf{GoedelIncompleteness}^{\uparrow}$

The Goedel-incompleteness interface $\mathsf{GoedelIncompleteness}^{\uparrow}$ packages an arithmetised provability predicate over a recursively-axiomatised first-order theory of arithmetic, together with a self-referential fixed-point row. It depends on $\mathsf{NameCert}_{\mathsf{Nat}^{\uparrow}}$ (chapter 28), $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ (chapter 197), $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ (chapter 198), and $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ (chapter 199).

288.1 Carrier definition

A $\mathsf{GoedelIncompleteness}^{\uparrow}$ token carries: a recursively-enumerated axiom set T over the language of arithmetic, an arithmetised provability predicate $\mathsf{Prov}_T(x)$ via a $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ proof-checking decoder, a Goedel numbering $\ulcorner\cdot\urcorner$ on formulas, and a fixed-point row G with $T \vdash G \leftrightarrow \neg\mathsf{Prov}_T(\ulcorner G\urcorner)$. The carrier accepts an incompleteness witness when T is consistent and G is neither provable nor refutable in T. The classifier is hsame on the fixed-point rows; the ledger keeps the axiom enumeration, the proof-checker, the Goedel numbering, and the fixed-point row visible.

Theorem 288.1 (GoedelIncompleteness proof-checker obligation)**.** *An accepted* $\mathsf{GoedelIncompleteness}^{\uparrow}$ *witness exposes a recursively enumerated axiom row, a* $\mathsf{Computable}^{\uparrow}$ *proof-checker decoder, and the finite proof-verdict ledger over* BHist*. The carrier reads a proof row only through the displayed decoder and records its verdict as a visible* $\mathsf{FirstOrder}^{\uparrow}$ *syntactic row.*

The obligation is syntactic and finite-ledger local. It does not assert semantic completeness, second incompleteness, or a set-theoretic foundations bridge.

Lean4 ✓ `BEDC.Derived.GoedelIncompletenessUp.GoedelIncompletenessProofCheckerLedger verdict nonempty surface`

Proof. Project the accepted packet to its axiom-enumeration coordinate. That row supplies the recursively enumerated theory data in the displayed $\mathsf{FirstOrder}^{\uparrow}$ language.

Read the proof-checker coordinate through the $\mathsf{Computable}^{\uparrow}$ decoder. Each candidate proof is consumed by this finite decoder row, and the result is the visible proof-verdict ledger over BHist.

Repacking these coordinates gives the proof-checker obligation. No semantic model argument, second-incompleteness claim, or external foundations bridge is used. □

Lean4 ✓ `BEDC.Derived.GoedelIncompletenessUp.GoedelIncompletenessWitnessPacket proof checker obligation`

Theorem 288.2 (GoedelIncompleteness fixed-point obligation)**.** *The fixed-point row in an accepted* $\mathsf{GoedelIncompleteness}^{\uparrow}$ *witness carries the displayed* $\mathsf{FirstOrder}^{\uparrow}$ *formula, its Goedel-number row, and the provability-predicate row for the fixed proof-checker decoder. The classifier* hsame *compares fixed-point histories only at the displayed numbering and does not identify formulas across a different coding surface.*

This obligation records the finite fixed-point ledger. It does not assert the Loeb conditions, semantic completeness, or a second-incompleteness theorem.

Proof. Start with the fixed-point coordinate of the accepted packet. It exposes the $\mathsf{FirstOrder}^{\uparrow}$ formula row and its Goedel-number row in the same finite ledger.

Next read the provability-predicate coordinate. It is tied to the displayed proof-checker decoder, so the predicate row refers to the fixed numbering rather than to an ambient coding theorem.

The hsame classifier is then applied only at that fixed numbering. This gives the stated fixed-point obligation without importing semantic or metatheoretic closure beyond the visible rows. □

Lean4 ✓ `BEDC.Derived.GoedelIncompletenessUp.GoedelIncompletenessFixedPointLedger obligation`

Theorem 288.3 (GoedelIncompleteness undecidable row obligation)**.** *Conditionally on the consistency row displayed by an accepted* $\mathsf{GoedelIncompleteness}^{\uparrow}$ *witness, the witness ledger contains both the no-proof row and the no-refutation row for the displayed fixed-point formula. The no-proof row is checked against* $\mathsf{Prov}_T(\ulcorner G \urcorner)$*, and the no-refutation row is checked against the displayed negation row in the same* $\mathsf{FirstOrder}^{\uparrow}$ *syntax.*

The statement is only the accepted finite undecidability ledger. It does not claim second incompleteness, semantic completeness, ω-consistency, or the Loeb conditions.

Lean4 ✓ `BEDC.Derived.GoedelIncompletenessUp.GoedelIncompletenessWitnessPacket undecidable row obligation`

Proof. Read the consistency coordinate, the fixed-point formula, and the proof-checker verdict ledger from the accepted packet. They share the same Goedel numbering and the same $\mathsf{Computable}^{\uparrow}$ decoder.

For the no-proof side, inspect the verdict rows attached to $\mathsf{Prov}_T(\ulcorner G \urcorner)$. The accepted packet records that no displayed proof row verifies the fixed-point formula.

For the no-refutation side, inspect the verdict rows attached to the displayed negation of G. The same proof-checker ledger records that no displayed proof row verifies the refutation. Repacking the two rows gives the conditional undecidable-row obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{GoedelIncompleteness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GoedelIncompleteness}^{\uparrow}$ token names a recursively-enumerated axiom set T over the language of arithmetic, an arithmetised provability predicate $\mathsf{Prov}_T(x)$ realised by a $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ proof-checking decoder, a Goedel numbering $\ulcorner \cdot \urcorner$ on formulas, and a self-referential fixed-point row G such that $T \vdash G \leftrightarrow \neg\mathsf{Prov}_T(\ulcorner G \urcorner)$. The carrier accepts an incompleteness witness when T is consistent in the sense that no proof-checker run yields $\bot$, and G is neither provable nor refutable in T. The classifier is hsame on the fixed-point rows at fixed Goedel numbering; the source covers the axiom enumeration, the proof-checker decoder, Goedel-number rows, fixed-point rows, and provability witnesses; the pattern records consistency admissibility, fixed-point admissibility, derivability admissibility, refutability admissibility, and recursive-axiomatisation admissibility rows; the ledger keeps the axiom enumeration, the proof-checker, the Goedel numbering, and the fixed-point row visible. The five name-certificate obligations are supplied by $\mathsf{NameCert}_{\mathsf{Computable}^{\uparrow}}$ for the proof-checker plus $\mathsf{NameCert}_{\mathsf{FirstOrder}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{ModelTheory}^{\uparrow}}$ for the syntactic carrier: the inhabited identity certificate is the empty axiom system over the language of arithmetic; classifier stability is reflexive, symmetric, and transitive hsame on fixed-point rows; transport repacks proof-checks through Goedel-number transport; ledger exactness identifies the accepted witness with the fixed-point identity; the extension boundary fixes the recursive-axiomatisation rule. A history h lies in the signature gap of a $\mathsf{GoedelIncompleteness}^{\uparrow}$ token p when the source-domain accepts the consistent recursively-axiomatised

theory, the signature relation pairs it with the undecidable fixed-point row, and token introduction names that fixed point as p. The chapter registers this self-referential incompleteness surface under the name GoedelIncompleteness$^{\uparrow}$; it is a finite-axiomatisation incompleteness object with no semantic-completeness theorem, no second-incompleteness theorem ($\neg\mathsf{Prov}_T(\bot)$ unprovability), no ω-consistency claim, and no standard set-theoretic foundations bridge.

Theory closure: TheoryGate(NameCert$_{\mathsf{GoedelIncompleteness}^{\uparrow}}$, publicClosed)

Scope closed: The obligation scope binds the proof-checker decoder row, fixed-point formula and numbering row, and conditional no-proof plus no-refutation rows over the displayed FirstOrder$^{\uparrow}$, ModelTheory$^{\uparrow}$, Computable$^{\uparrow}$, and hsame surfaces.

Formal verification: FormalStatus(NameCert$_{\mathsf{GoedelIncompleteness}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GoedelIncompletenessUp.GoedelIncompletenessFixedPointLedger\_obligation`

Bridge status: none

Not claimed: The chapter does not close the second-incompleteness theorem, ω-consistency, semantic-completeness, the Loeb conditions, or a standard set-theoretic foundations bridge.

Upgrade path: Bridged closure requires the standard bridge from the exported fixed-point ledger to FirstOrder$^{\uparrow}$, ModelTheory$^{\uparrow}$, and Computable$^{\uparrow}$ consumers.

Chapter 289

A Concrete Naming Certificate for $\mathsf{Belief}^{\uparrow}$

The $\mathsf{Belief}^{\uparrow}$ interface is an AI-proposed BEDC chapter. Its carrier sketch and dependency anchors come from the proposal at `papers/bedc/proposals/accepted/`; the seed-stub here records the closurestatus block. Its dependency rows are the finite index spine of chapter 28, the update trace surface of chapter 120, and the probability certificate surface of chapter 184.

Falsifiable prediction: If a public $\mathsf{Belief}^{\uparrow}$ consumer deletes the prior row, observation row, update trace, probability row, or evidence ledger, then it cannot reconstruct the public belief consumer surface of Definition 289.8.

Independence witness: Nearest siblings: nat, seq, probspace, modeltheory, hypothesis-test. The carrier is not bijective to any of them because it binds five simultaneous rows: a finite index spine, an observation row, an update trace, a probability certificate, and an evidence ledger; each named sibling supplies at most a proper projection of that packet rather than the whole observation-update ledger.

289.1 Carrier definition

A finite $\mathsf{Belief}^{\uparrow}$ packet is the observation-update carrier suggested by `BEDC.Derived.BeliefUp.taste_gate`; the Lean touchpoint fixes the chapter name and TasteGate carrier, while the paper surface below records the certificate obligations without marking them as checked. On the Lean side, any future AI-origin upgrade that relies on non-vacuity must keep the existing `FieldFaithful` instance and add a `Nontrivial BeliefUp` instance.

Lean4 ✓ `BEDC.Derived.BeliefUp.taste gate`

Definition 289.1 (Belief observation-update carrier)**.** *A* $\mathsf{Belief}^{\uparrow}$ *observation-update carrier is a finite packet over* BHist *consisting of the following rows.*

1. *A prior row naming the displayed belief state before the observation.*
2. *An observation row naming the finite evidence read by the update.*
3. *An update trace recording the visible constructor route from the prior row and observation row to the posterior row.*
4. *A probability row supplied by* $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ *and classified only through the displayed probability certificate.*

5. *An evidence ledger that exposes exactly the prior, observation, update trace, and probability rows used by the packet.*

The packet is accepted when those rows are present in the same finite BHist *ledger, the update trace is a* $\mathsf{Seq}^{\uparrow}$ *row, and the posterior belief row is the row named by that trace.*

Lean4 (definition) `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier`

Theorem 289.2 (Belief classifier transport obligation)**.** *Let p be an accepted* $\mathsf{Belief}^{\uparrow}$ *packet with prior row, observation row, update trace, and probability row displayed in its evidence ledger. If* hsame *transports each of those four rows to corresponding displayed rows, then the transported rows repack as an accepted* $\mathsf{Belief}^{\uparrow}$ *packet with the same update-trace shape and a posterior row classified by the transported probability certificate.*

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier classifier transport`

Proof. Read the accepted packet through its finite BHist ledger. The prior and observation rows are transported first, so the update trace has the same source rows after transport.

Next transport the update trace as a $\mathsf{Seq}^{\uparrow}$ row. Since the trace records a visible constructor route, the posterior row is obtained by replaying the same finite route on the transported prior and observation rows.

Finally transport the probability row through the displayed $\mathsf{NameCert}_{\mathsf{ProbSpace}^{\uparrow}}$ certificate. The four transported rows fill the same carrier fields, so the packet is accepted with no hidden evidence row. □

Theorem 289.3 (Belief ledger exactness obligation)**.** *For every accepted* $\mathsf{Belief}^{\uparrow}$ *packet, the evidence ledger exposes exactly the prior row, observation row, update trace, and probability row admitted by the carrier. A downstream consumer may read those four rows and the posterior row named by the update trace, but no additional hidden evidence, prior source, or probability witness is part of the packet.*

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier ledger exactness`

Proof. Inspect the carrier definition. Acceptance requires the prior row, observation row, update trace, probability row, and evidence ledger to be displayed in the same finite BHist packet.

The ledger exactness clause then identifies every exported evidence entry with one of those four carrier rows. The posterior is readable only as the endpoint of the displayed update trace, so it does not add an independent hidden evidence coordinate. □

Theorem 289.4 (Belief TasteGate carrier recognition)**.** *Every accepted* $\mathsf{Belief}^{\uparrow}$ *packet is constrained by the TasteGate carrier named by the Lean definition* `BEDC.Derived.BeliefUp.taste_gate`*. In particular, the accepted packet may expose only the prior row, observation row, update trace, probability row, posterior endpoint, evidence ledger, and displayed provenance rows admitted by that carrier boundary.*

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier tastegate carrier recognition`
↪ variant `BEDC.Derived.BeliefUp.BeliefTasteGate carrier recognition`

Proof. Read the accepted packet through the TasteGate carrier boundary named above. The carrier admits the observation-update rows as a finite BHist packet and supplies no constructor for a host belief state, hidden likelihood object, or external update operation.

Projecting the packet therefore leaves only the displayed prior, observation, update trace, probability, posterior endpoint, evidence ledger, and provenance rows. Since every accepted packet must pass through the same carrier boundary, consumer recognition is restricted to those rows. □

Theorem 289.5 (Belief TasteGate carrier obligation scope)**.** *An accepted* $\mathsf{Belief}^{\uparrow}$ *packet leaves seed-level TasteGate recognition only through the carrier rows of Definition 289.1, the classifier transport obligation of Definition 289.2, ledger exactness from Definition 289.3, posterior trace transport from Definition 289.6, finite evidence ledger coverage from Definition 289.7, the public* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$ *surface of Definition 289.8, and the sibling dependency lattice of Definition 289.11.*

The checked TasteGate marker displayed in section 289.1, named `BEDC.Derived.BeliefUp.taste_gate`*, recognizes only the carrier boundary. Scope beyond seed recognition is therefore a paper-side obligation over the displayed prior, observation, update trace, probability, posterior endpoint, evidence ledger,* hsame*,* Cont*,* Pkg*, and sibling rows, not a hidden likelihood object or host belief-update operation.*

Proof. Start with Definition 289.4. The TasteGate boundary admits the finite observation-update rows and no external update primitive.

Project the carrier through Definition 289.1. The prior, observation, update trace, probability row, evidence ledger, and posterior endpoint are the only rows available for scope growth.

Apply Definition 289.2, Definition 289.3, and Definition 289.6. These clauses keep transport, ledger readback, and posterior trace movement attached to the same finite rows.

Finally use Definition 289.7, Definition 289.8, and Definition 289.11. They expose exactly the finite evidence ledger, public naming surface, and $\mathsf{Nat}^{\uparrow}/\mathsf{Seq}^{\uparrow}/\mathsf{ProbSpace}^{\uparrow}$ sibling route. Since the cited TasteGate marker supplies only recognition of the carrier boundary, the stated rows exhaust the obligation scope. □

Theorem 289.6 (Belief posterior trace transport obligation)**.** *Let p be an accepted* $\mathsf{Belief}^{\uparrow}$ *packet. If its prior row, observation row, update trace, and probability row are transported through displayed* hsame *comparisons and* Cont *routes, then the transported rows repack as a* $\mathsf{Belief}^{\uparrow}$ *packet whose posterior endpoint is the endpoint named by the transported update trace.*

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier posterior trace transport`

Proof. Start with the accepted packet and project its four transported input rows. The prior and observation endpoints move through the displayed hsame comparisons, while the update trace moves along the displayed Cont route.

The posterior endpoint is not an independent input. It is read as the endpoint named by the update trace after transport, so replaying the transported trace with the transported prior and observation rows determines the posterior row.

Transport the probability row through the displayed probability certificate and place it with the transported trace data in the same finite ledger. These rows fill the carrier fields again, so the packet repacks under hsame/Cont. □

Theorem 289.7 (Belief finite evidence ledger coverage)**.** *For every accepted* $\mathsf{Belief}^{\uparrow}$ *packet, the finite evidence ledger covers exactly the prior row, observation row, update trace, probability row, and posterior endpoint. No additional likelihood row, hidden prior source, external observation stream, or host update primitive is exported by the ledger.*

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefFiniteEvidenceLedger coverage`

Proof. Inspect the carrier rows. Acceptance places the prior, observation, update trace, probability row, and posterior endpoint in one finite BHist ledger with their displayed provenance.

The ledger coverage clause identifies each readable evidence entry with one of those five rows. Since the posterior endpoint is already the trace endpoint, it does not create a second evidence source.

Thus a downstream consumer can cite exactly the listed rows and no hidden belief-update data. Every permitted read is a projection of the displayed finite ledger. □

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier finite evidence ledger coverage`

Theorem 289.8 (Belief public NameCert surface)**.** *The observation-update carrier, classifier transport, ledger exactness, TasteGate carrier recognition, posterior trace transport, and finite evidence ledger coverage assemble the public* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$ *surface. Its consumer-readable rows are exactly the prior, observation, update trace, probability, posterior endpoint, evidence ledger,* hsame, Cont, *and* Pkg *rows.*

This surface exports no hidden prior source, external observation stream, host update primitive, likelihood object outside the probability certificate, or bridge datum.

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier public namecert surface`

Proof. Project the packet through Definition 289.1. The carrier supplies the prior, observation, update trace, probability, evidence ledger, and posterior endpoint rows inside one finite BHist packet.

Apply Definition 289.2 and Definition 289.3. These obligations give the displayed classifier movement and ledger readback, so a consumer reads only the carrier rows transported through hsame, Cont, and Pkg.

Next use Definition 289.4. The TasteGate marker `BEDC.Derived.BeliefUp.taste_gate` fixes the accepted carrier boundary and contributes no additional evidence row.

Finally apply Definition 289.6 and Definition 289.7. The posterior endpoint is read from the update trace, and the finite ledger covers exactly the listed evidence rows.

The cited clauses exhaust the public $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$ surface. Since no clause exports a hidden prior source, external observation stream, host update primitive, likelihood object outside the probability certificate, or bridge datum, no consumer can read one from the public surface. □

Theorem 289.9 (Belief mature consumer exhaustion)**.** *Every downstream consumer of the public* $\mathsf{Belief}^{\uparrow}$ *packet reads only the prior row, observation row, update trace, probability certificate row, posterior endpoint, finite evidence ledger,* hsame, Cont, Pkg, *and the TasteGate carrier boundary named by* `BEDC.Derived.BeliefUp.taste_gate`*. Every such read factors through Definition 289.8.*

No hidden likelihood object, host update primitive, external observation stream, or extra probability witness is exported by the mature packet surface.

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefObservationUpdateCarrier mature consumer exhaustion`

Proof. Project the consumer read through Definition 289.8. The public surface is already the assembled observation-update carrier with its classifier transport, ledger exactness, TasteGate recognition, posterior trace, and finite evidence coverage rows.

The six source clauses give the exhaustive list. Apply Definition 289.2, Definition 289.3, Definition 289.4, Definition 289.6, Definition 289.7, and Definition 289.8 to reduce each consumer read to one displayed row of the packet.

The checked TasteGate marker displayed in section 289.1, named `BEDC.Derived.BeliefUp.taste_gate`, supplies only the carrier boundary. It contributes no hidden likelihood object, host update primitive, external observation stream, or extra probability witness. □

Theorem 289.10 (Belief finite evidence consumer completeness)**.** *Every finite consumer of an accepted* $\mathsf{Belief}^{\uparrow}$ *packet reads exactly the prior row, observation row, posterior endpoint, finite evidence ledger, classifier transport rows, posterior trace transport rows, and the TasteGate carrier named by* `BEDC.Derived.BeliefUp.taste_gate`*. Conversely, those rows repack as the public belief consumer surface.*

The packet exposes no hidden likelihood object, host update primitive, external observation stream, or extra probability witness beyond those rows.

Lean4 ✓ `BEDC.Derived.BeliefUp.BeliefFiniteEvidenceConsumerCompleteness`

Proof. Apply Definition 289.4. The named TasteGate carrier fixes the accepted $\mathsf{Belief}^{\uparrow}$ boundary without adding an evidence row.

Use Definition 289.6 to route the posterior endpoint and posterior trace through the displayed hsame, Cont, and Pkg coordinates.

Next apply Definition 289.7. This exhausts the finite evidence ledger available to any accepted packet consumer.

Finally use Definition 289.8. The prior, observation, posterior, evidence, classifier-transport, posterior-trace, and TasteGate rows repack as exactly the public belief consumer surface, with no remaining coordinate for a hidden likelihood object, host update primitive, external observation stream, or extra probability witness. □

Theorem 289.11 (Belief sibling dependency lattice)**.** *Every public* $\mathsf{Belief}^{\uparrow}$ *consumer reads* $\mathsf{Nat}^{\uparrow}$ *as the finite index spine,* $\mathsf{Seq}^{\uparrow}$ *as the update trace, and* $\mathsf{ProbSpace}^{\uparrow}$ *as the probability certificate before the TasteGate carrier boundary is opened. The packet exports no hidden likelihood object, host update primitive, or extra probability witness outside those sibling rows and the displayed public* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$ *surface.*

Proof. First apply Definition 289.2. The transported prior, observation, update trace, and probability rows must remain visible finite rows of the same accepted packet, so the finite index spine is read through the displayed $\mathsf{Nat}^{\uparrow}$ dependency before the consumer opens the carrier boundary.

Next use Definition 289.6. The posterior endpoint is determined by the transported update trace, and that trace is the $\mathsf{Seq}^{\uparrow}$ row recorded by the packet rather than a host update primitive.

Apply Definition 289.7. The evidence ledger covers the prior, observation, update trace, probability row, and posterior endpoint, so the probability contribution is exactly the $\mathsf{ProbSpace}^{\uparrow}$ certificate row already present in the finite ledger.

Finally use Definition 289.8. The public $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$ surface repacks these sibling rows and the TasteGate carrier boundary, leaving no coordinate for a hidden likelihood object, host update primitive, or extra probability witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Belief}^{\uparrow}$ packet is generated by assembling a finite BHist ledger whose visible rows are the prior, observation, update trace, probability certificate, posterior endpoint, and evidence ledger. The prior and observation rows are the sources; the $\mathsf{Seq}^{\uparrow}$ update trace is the displayed Cont route that carries them to the posterior row; the $\mathsf{ProbSpace}^{\uparrow}$ row supplies the probability coordinate without adding a hidden likelihood object. The classifier reads only rowwise hsame transport of the prior, observation, trace, and probability rows, so the transported packet has the same posterior-trace shape. Ledger exactness and the source specification coincide because every evidence entry is one of the displayed rows.

Nonemptiness is the presence of such a finite accepted ledger, and reflexivity, symmetry, and transitivity are inherited componentwise from the displayed hsame comparisons.

Theory closure: TheoryGate(NameCert$_{\mathsf{Belief}\uparrow}$, maturePackageClosed)

Scope closed: Definition 289.10 and the public NameCert$_{\mathsf{Belief}\uparrow}$ surface bind the observation-update carrier definition together with the classifier-transport and ledger-exactness obligations, TasteGate carrier recognition, posterior trace transport, and finite evidence coverage for the displayed prior, observation, update trace, probability, posterior endpoint, evidence ledger, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{Belief}\uparrow}$, encodedDef)

Lean target: `BEDC.Derived.BeliefUp.taste\_gate`

Bridge status: paperBridge

Origin: ai

Not claimed: Machine-checked standard bridge acceptance remains outside this paper-axis closure claim.

Upgrade path: There is no further paper-axis work for the mature packet surface; machine bridge acceptance remains a separate formal-axis task.

Chapter 290

A Concrete Naming Certificate for $\mathsf{Policy}^{\uparrow}$

The $\mathsf{Policy}^{\uparrow}$ interface is an AI-proposed BEDC chapter. Its carrier sketch and dependency anchors come from the proposal at `papers/bedc/proposals/accepted/`; the seed-stub here records the closurestatus block. Dependencies cited by the proposal: $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$. The substantive sibling route is chapter 289, chapter 189, chapter 185, and chapter 230: $\mathsf{Policy}^{\uparrow}$ consumes belief-state rows, Markov transition rows, random-variable observation rows, and estimator-score rows before selecting an action ledger.

Lean4 (definition) `BEDC.Derived.PolicyUp.taste gate`

290.1 Carrier definition

Definition 290.1 (Policy action-ledger carrier)**.** *A* $\mathsf{Policy}^{\uparrow}$ *carrier is a finite* BHist *action-ledger packet over* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$. *Its rows are belief-state rows, Markov transition rows, estimator-score rows, selected action rows, the* hsame *transport rows comparing corresponding ledger entries, the* Cont *provenance rows explaining how each selected action is read from the displayed belief, transition, and estimator rows, and the* Pkg *source rows naming the four dependency certificates.*

The carrier accepts only packets whose selected-action row is sourced from the displayed dependency rows through the visible Cont *ledger. It contains no host Markov decision process, utility function, optimizer, or external policy object.*

Theorem 290.2 (Policy TasteGate source scope)**.** *Every accepted* $\mathsf{Policy}^{\uparrow}$ *action-ledger packet exposes exactly the four dependency certificates* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ *together with the finite action ledger of Definition 290.1. No host Markov decision process, utility function, optimizer, or external policy object is exported by the packet.*

Proof. Project the carrier rows of Definition 290.1. The source rows are precisely the four named dependency certificates and the finite BHist action ledger whose entries display belief states, transition rows, estimator scores, selected actions, hsame comparisons, Cont provenance, and Pkg source rows.

The acceptance condition reads a selected action only through those displayed rows. Since the carrier definition has no constructor for a host Markov decision process, utility function, optimizer, or external policy object, no accepted packet can expose such a row. □

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedger source scope`

Theorem 290.3 (Policy local action transport)**.** *Let two finite* Policy$^{\uparrow}$ *action-ledger packets have componentwise* hsame *comparisons between their belief-state rows, Markov transition rows, estimator-score rows, and selected-action rows. If the* Cont *provenance and* Pkg *source rows are transported along those comparisons, then the transported rows again form an accepted local* Policy$^{\uparrow}$ *carrier packet over the same four dependency certificates.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier local action transport`

Proof. Transport the four dependency rows componentwise through the displayed hsame comparisons. The belief, transition, estimator, and selected-action entries remain inside the same finite BHist ledger shape.

Next transport the Cont provenance rows and the Pkg source rows. The carrier acceptance condition only asks that the selected-action row be read from the transported belief, transition, and estimator rows through that visible provenance ledger. Therefore the transported packet satisfies the same local carrier predicate. □

Theorem 290.4 (Policy action classifier stability)**.** *Let P and P' be accepted* Policy$^{\uparrow}$ *action-ledger packets over the same four displayed dependency certificates. Suppose the belief-state rows, Markov transition rows, random-variable rows, estimator-score rows, selected-action rows,* Cont *provenance rows, and* Pkg *source rows of P are transported componentwise to the corresponding rows of P' by displayed* hsame *comparisons. Then the transported packet is again accepted by the PolicyUp action-ledger carrier of Definition 290.1, and the selected-action classifier reads the same visible BEDC ledger surface.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier action classifier stability`

Proof. Inspect the finite action-ledger carrier. Its acceptance predicate names only the displayed dependency certificates, the belief, transition, random-variable, estimator, and selected-action rows, and the visible hsame/Cont/Pkg provenance attached to those rows.

Transport each listed coordinate along the supplied hsame rows. The transport preserves the finite BHist ledger shape and leaves the source certificates inside the same four dependency packets.

The selected-action classifier has no access to any row outside that carrier. After the Cont and Pkg rows are transported, the classifier therefore reads the selected action through the transported visible ledger and accepts the packet again. □

Theorem 290.5 (Policy visible continuation provenance coverage)**.** *Every selected-action row in an accepted* Policy$^{\uparrow}$ *packet is read through the visible* Cont *provenance from the displayed* Belief$^{\uparrow}$, MarkovChain$^{\uparrow}$, RandomVar$^{\uparrow}$, *and* Estimator$^{\uparrow}$ *source rows. The packet exposes no host Markov decision process, optimizer, utility function, or external policy object.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier visible cont provenance coverage`

Proof. Project an accepted packet to the rows named by Definition 290.1. The selected-action coordinate is accepted only when a visible Cont row relates it to the displayed belief, transition, random-variable, and estimator rows, with Pkg provenance naming the four source certificates.

No other source is present in the carrier. A host Markov decision process, optimizer, utility function, or external policy object would require an additional readable coordinate, but the carrier has no such coordinate. □

Theorem 290.6 (Policy NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$ *obligation surface consists of the action-ledger carrier of Definition 290.1, the TasteGate source scope of Definition 290.2, action classifier stability from Definition 290.4, visible continuation provenance coverage from Definition 290.5, and local action transport from Definition 290.3.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier semantic name certificate`
↪ variant `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier namecert obligation surface`

Proof. Use the action-ledger carrier as the source predicate. It fixes the finite BHist rows and the four displayed dependency certificates.

Use the TasteGate source-scope theorem to bound the exported sources. Use classifier stability to transport the accepted action classifier along componentwise hsame rows, and use visible provenance coverage to identify the exact Cont rows through which selected actions are read.

Finally apply local action transport to repack transported rows as another accepted carrier. These clauses provide exactly the local BEDC surface named by $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$, without adding host MDP, optimizer, utility, or external policy data. □

Theorem 290.7 (Policy selected-action determinacy)**.** *Let P and Q be accepted* $\mathsf{Policy}^{\uparrow}$ *action-ledger packets over the same transported* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$ *rows. Suppose their belief-state, Markov transition, random-variable, estimator,* Cont, *and* Pkg *rows are matched componentwise by displayed* hsame *transport, and suppose each selected-action row is accepted only through those displayed rows. Then the selected-action classifier assigns P and Q to the same visible selected-action row.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier selected action determinacy`

Proof. Project both accepted packets to the finite action-ledger carrier of Definition 290.1. The carrier exposes the four dependency rows, the local ledger rows, and the hsame/Cont/Pkg transport data, so these are the only coordinates available to the classifier.

Use the supplied componentwise hsame comparisons to transport the displayed belief-state, Markov transition, random-variable, estimator, Cont, and Pkg rows from P to Q. By Definition 290.4, the transported packet remains accepted by the same local carrier and the selected-action classifier reads the same visible ledger surface.

It remains to rule out a second action source. The carrier contains no host MDP, optimizer, utility, or external policy coordinate, and Definition 290.5 makes the visible Cont route exhaustive. Hence two accepted packets with the same transported dependency and provenance rows classify the same selected-action row. □

Theorem 290.8 (Policy dependency source exactness)**.** *Every consumer-visible row of an accepted* $\mathsf{Policy}^{\uparrow}$ *action-ledger packet is one of the four dependency certificate rows* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, *the selected-action row, an* hsame *transport row, a* Cont *provenance row, or a* Pkg *source row naming one of the four displayed certificates.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier dependency source exactness`

Proof. Unfold the carrier in Definition 290.1. Its public packet is a finite BHist ledger whose readable coordinates are the four dependency rows, selected-action entries, hsame comparisons, Cont provenance, and Pkg source rows.

Apply Definition 290.2 to the source side. The source projection exposes exactly the four named dependency certificates and no host MDP, optimizer, utility, or external policy object.

Apply Definition 290.5 to the action side. Each selected action is read through visible Cont provenance from the displayed dependency rows, with Pkg rows naming those sources. Therefore every consumer-visible row belongs to the listed inventory. □

Theorem 290.9 (Policy TasteGate scoped package)**.** *The action-ledger carrier, TasteGate source scope, action classifier stability, visible continuation provenance coverage, local action transport, and* `BEDC.Derived.PolicyUp.taste_gate` *together form the scoped* $\mathsf{Policy}^{\uparrow}$ *BEDC package. This package excludes host Markov decision processes, optimizers, utility functions, and external policy objects from the exported surface.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier tastegate scoped package`

Proof. Start with the five local obligations enumerated in Definition 290.6. They provide the carrier, the bounded source inventory, classifier stability, Cont coverage, and transport closure for accepted action-ledger packets.

Use the TasteGate object `BEDC.Derived.PolicyUp.taste_gate` as the chapter-level admission gate. It packages the local carrier obligations without adding a new consumer-visible row, so the available projections remain the rows already listed in Definition 290.8.

Finally combine source exactness with selected-action determinacy. The transported dependency, Cont, and Pkg rows determine the visible selected action, while the carrier has no constructor for a host MDP, optimizer, utility, or external policy object. Thus the TasteGate package is a scoped BEDC surface and nothing beyond that surface is exported. □

Theorem 290.10 (Policy kernel scope boundary)**.** *The five local obligations of* $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$ *are scoped exactly to the finite* BHist *action-ledger carrier of Definition 290.1, the four dependency certificates* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, *and the displayed* hsame, Cont, *and* Pkg *rows. No host Markov decision process, optimizer, utility function, external policy object, hidden state space, or unlogged action source is readable by those obligations.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier kernel scope boundary`

Proof. Start from the action-ledger carrier of Definition 290.1. Its rows are the finite BHist ledger, the four dependency certificate rows, the visible hsame transport rows, the Cont provenance rows, and the Pkg source rows.

The source side is bounded by Definition 290.2. The classifier and transported carrier reads are bounded by Definition 290.3 and Definition 290.4. The selected-action provenance is bounded by Definition 290.5.

Finally, Definition 290.6 says that these are exactly the five local obligations named by $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$. Since each source, classifier, transport, and ledger read has already been accounted for by the listed rows, no host MDP, optimizer, utility, external policy object, hidden state space, or unlogged action source is exposed. □

Theorem 290.11 (Policy public export surface)**.** *The public* $\mathsf{Policy}^{\uparrow}$ *surface consists exactly of the finite action-ledger carrier of Definition 290.1, the four dependency certificate rows* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, *selected-action determinacy, dependency source exactness, the TasteGate scoped package Definition 290.9, and the kernel-scope boundary Definition 290.10. It exports no host Markov decision process, optimizer, utility function, or external policy row.*

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier public export surface`

Proof. The TasteGate evidence is the chapter's displayed `BEDC.Derived.PolicyUp.taste_gate` row together with the checked scoped package recorded at Definition 290.9. That package admits the action-ledger carrier without adding a consumer-visible row beyond the listed finite carrier, dependency certificates, hsame, Cont, and Pkg provenance.

Apply Definition 290.8 to the dependency side. The only readable sources are the four displayed certificate rows for $\mathsf{Belief}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{Estimator}^{\uparrow}$. Apply Definition 290.7 to the action side. Once those source rows and visible continuation provenance are fixed, the selected action row is fixed.

Finally apply Definition 290.10. The kernel boundary has no constructor or projection for a host MDP, optimizer, utility function, or external policy object. Therefore the public export is precisely the action-ledger surface listed in the statement. □

Theorem 290.12 (Policy action-ledger finite consumer coverage)**.** *Every consumer of the* $\mathsf{Policy}^{\uparrow}$ *bridge reads only the finite action-ledger carrier, the four dependency certificate rows* $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$, *selected-action determinacy, the TasteGate scoped package, the kernel-scope boundary, the displayed* hsame, Cont, *and* Pkg *rows, and the local* $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$ *rows. The bridge surface exports no host Markov decision process, optimizer, utility function, external policy object, hidden state space, or unlogged action source.*

Proof. Start from the public surface of Definition 290.11. It lists the finite action-ledger carrier, the four dependency certificate rows, selected-action determinacy, dependency source exactness, the TasteGate scoped package, and the kernel-scope boundary as the exported $\mathsf{Policy}^{\uparrow}$ rows.

Use Definition 290.9 together with the displayed `BEDC.Derived.PolicyUp.taste_gate` row. The TasteGate admission packages the action-ledger carrier without adding any consumer-visible source beyond the finite ledger, dependency certificates, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$ rows.

Finally apply Definition 290.10. That boundary rules out host MDPs, optimizers, utility functions, external policy objects, hidden state spaces, and unlogged action sources. Hence every finite bridge consumer is covered by the displayed action-ledger surface. □

Lean4 ✓ `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier finite consumer coverage`

Closure status for $\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Policy}^{\uparrow}$ packet is generated as a finite action ledger over the four displayed dependency certificates $\mathsf{NameCert}_{\mathsf{Belief}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{MarkovChain}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RandomVar}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{Estimator}^{\uparrow}}$. Its visible coordinates are belief-state rows, Markov transition rows, random-variable observation rows, estimator-score rows, selected-action rows, rowwise hsame comparisons, Cont provenance explaining how the selected action is read, and Pkg source rows naming the dependencies. The classifier is componentwise: after transporting those rows by hsame, the same visible continuation ledger still selects the same action coordinate. The ledger and source specification align because every public read factors through one of those dependency rows and no host decision process, optimizer, utility function, or external policy object is present. Nonemptiness is an accepted finite ledger; reflexivity, symmetry, and transitivity are inherited row by row from hsame.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface is the action-ledger carrier with the public TasteGate scoped package, TasteGate source scope, action classifier stability, visible continuation provenance coverage, local action transport, finite consumer coverage, and the kernel boundary over the four displayed dependency certificates.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Policy}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.PolicyUp.PolicyActionLedgerCarrier\_kernel\_scope\_boundary`

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.PolicyUp.taste gate`

Not claimed: The chapter does not claim a host Markov decision process, optimizer, utility function, external policy object, or public standard bridge.

Upgrade path: BridgeChecked verification requires a machine-checked standard bridge no-host-leak field.

DyadicRatCore$^{\uparrow}$ exposes the L10 sibling dependency relation: chapter 31 and chapter 37 supply the signed mantissa and denominator ledger, while chapter 126, chapter 301, and chapter 38 consume the finite dyadic tolerance ledger.

Chapter 291

A Concrete Naming Certificate for $\mathsf{DyadicRatCore}^{\uparrow}$

The $\mathsf{DyadicRatCore}^{\uparrow}$ surface names the dyadic rational source layer used by the Phase Real and Completion interface in section 9467.12. Following the Bishop–Bridges constructive-analysis treatment of dyadic rational approximants, the chapter consumes the signed integer and rational certificate surfaces of chapter 31 and chapter 37 through its mantissa and denominator ledger. It keeps each denominator exponent as visible unary history data and never identifies dyadics by a quotient carrier.

291.1 Carrier and classifier

Definition 291.1 (DyadicRatCore signed mantissa carrier)**.** *A* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier row is a finite* BHist *packet* (m, e, ℓ, P) *where* m *is a signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row,* e *is a positive* $\mathsf{UnaryHistory}$ *exponent row,* ℓ *is the denominator ledger displaying the unary tower for* 2^e*, and* P *is the* Pkg *provenance row naming the* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *source certificates. The intended value of the packet is the dyadic rational* $m \cdot 2^{-e}$*, but the carrier stores only the visible mantissa row, positive unary exponent row, denominator ledger, and provenance.*

The carrier contains no quotient class of rational pairs, no hidden integer normal form, and no imported real-number endpoint. A consumer can read a dyadic value only by projecting the displayed $\mathsf{Rat}^{\uparrow}$ *mantissa and unary denominator rows through the visible* Cont *ledger.*

Theorem 291.2 (DyadicRatCore Int/Rat sibling route)**.** *Every public read of an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet* $D = (m, e, \ell, P)$ *factors the signed mantissa through the consumed* $\mathsf{Rat}^{\uparrow}$ *row, the sign and integer provenance through the consumed* $\mathsf{Int}^{\uparrow}$ *row, and the exponent and denominator ledger through the visible unary-history route. The read has no coordinate for a quotient rational, host integer normal form, or* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. Project the four carrier fields in Definition 291.1. The mantissa field m is already a signed $\mathsf{Rat}^{\uparrow}$ row, while the provenance field P records the attached $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ source certificates cited at the start of this chapter.

The exponent field e and ledger field ℓ give the unary-history route for the displayed denominator. A public read can therefore inspect the dyadic packet only by following the mantissa, integer-provenance, rational-provenance, exponent, and ledger rows that are already part of D.

Finally use the carrier exclusion in Definition 291.1. Since the carrier contains no quotient class of rational pairs, no host integer normal form, and no imported real-number endpoint, the sibling route through $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ is the complete public route for the finite dyadic source layer. □

Definition 291.3 (DyadicRatCore common-exponent classifier)**.** *For two accepted dyadic packets $D = (m, e, \ell, P)$ and $D' = (m', e', \ell', P')$, the predicate* $\mathsf{DyadicRatCoreClassifier}(D, D')$ *is the common-exponent scaling classifier: it supplies a positive unary exponent row E above both e and e', transported denominator ledgers from ℓ and ℓ' to E, and a* $\mathsf{Rat}^{\uparrow}$ *classifier row between the two scaled mantissas $m \cdot 2^{E-e}$ and $m' \cdot 2^{E-e'}$.*

The classifier is a BEDC predicate over displayed scaling rows. It is not quotient equality of rational pairs, and it does not collapse the two packets to a host rational number.

Lean4 (definition) `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreClassifier`

Definition 291.4 (DyadicRatCore common-exponent scale packet)**.** *For accepted dyadic packets $D = (m, e, \ell, P)$ and $D' = (m', e', \ell', P')$, a* $\mathsf{DyadicRatCoreCommonExponentScale}(D, D')$ *packet is the finite scale tuple*

$$(E, s, s', r, P_E).$$

Here E is the displayed positive unary exponent row above both e and e', s transports the mantissa and denominator ledger of D to the common denominator 2^E, s' transports the corresponding rows of D' to 2^E, r is the $\mathsf{Rat}^{\uparrow}$ *classifier row comparing the transported mantissas, and P_E is the* Pkg *provenance row binding the same* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *sibling certificates already cited by Definition 291.2. The packet has no coordinate for a quotient rational representative, host denominator selector, host integer normalization, or* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Theorem 291.5 (DyadicRatCore common-exponent classifier stability)**.** *Let $D = (m, e, \ell, P)$ and $D' = (m', e', \ell', P')$ be accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets, and let (E, s, s', r, P_E) be a* $\mathsf{DyadicRatCoreCommonExponent}$ *packet. If D, D', E, s, s', r, P_E are transported componentwise through displayed* hsame, Cont, *and* Pkg *rows that preserve the visible* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *sibling provenance, then the transported tuple is again a common-exponent scale packet and it witnesses the same* $\mathsf{DyadicRatCoreClassifier}$ *comparison at the transported common exponent.*

The classifier read remains confined to the transported mantissa rows, positive unary exponent rows, denominator ledgers, $\mathsf{Rat}^{\uparrow}$ *comparison row, and provenance rows. It does not acquire a quotient rational carrier, host denominator selector, host integer normalization, or* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. Project the carrier rows of D and D' using Definition 291.1. The only source rows available for scaling are the signed mantissas, positive unary exponent histories, denominator ledgers, and Pkg rows carrying the $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ provenance.

Project the common-exponent scale packet of Definition 291.4. The rows s and s' are precisely the displayed transports from the two original exponent ledgers to E, and r is precisely the $\mathsf{Rat}^{\uparrow}$ classifier row between the scaled mantissas.

Transport these rows componentwise through the supplied hsame and displayed Cont data. The transported exponent remains the common positive unary row, the transported ledgers remain the two denominator ledgers at that row, and the transported $\mathsf{Rat}^{\uparrow}$ comparison is still attached to the same sibling provenance.

Finally apply the classifier definition Definition 291.3. It reads exactly the common exponent, transported denominator ledgers, scaled mantissas, and $\mathsf{Rat}^{\uparrow}$ classifier row. Since no other carrier coordinate exists, the transported tuple witnesses the same classifier comparison and cannot import any refused host object. □

Theorem 291.6 (DyadicRatCore tolerance halving closure)**.** *Let $D = (m, e, \ell, P)$ be an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance packet, and suppose a displayed successor unary exponent row e^+ extends e by one denominator-halving step. If the mantissa, denominator ledger, endpoint comparison, distance row, radius row,* hsame *transport, displayed* Cont *replay, and* Pkg *provenance are transported along this successor row, then the resulting packet is again accepted by the dyadic tolerance ledger.*

Equivalently, one displayed unary halving step preserves the finite DyadicRatCore tolerance surface: the accepted rows are still the $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ *mantissa provenance, positive unary exponent history, denominator ledger, dyadic comparison rows, transport rows, continuation rows, and package provenance, with no quotient rational equality, host normalization, selected real endpoint, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from the accepted carrier row of Definition 291.1. The exponent data are stored as positive unary history, so adjoining one displayed successor row gives another visible positive unary exponent history rather than a hidden denominator selector.

Transport the denominator ledger along the successor row. The ledger for 2^e is replayed into the ledger for 2^{e^+}, and the mantissa provenance remains attached to the same $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ certificates.

Read the tolerance rows after transport. Endpoint comparisons, distance rows, and radius rows are all part of the same finite dyadic packet, so halving the displayed denominator scale repacks them as the next accepted tolerance window.

Finally use the exclusion clause of the carrier definition and the common-exponent classifier discipline of Definition 291.3. The halved packet still exposes only displayed mantissa, exponent, ledger, comparison, transport, continuation, and provenance rows, so the dyadic tolerance ledger remains accepted without adding quotient rational equality, host normalization, selected endpoints, or ambient real completeness. □

Theorem 291.7 (DyadicRatCore denominator ledger transport)**.** *Let $D = (m, e, \ell, P)$ be an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet, and let $D' = (m', e', \ell', P')$ be obtained from D by displayed* hsame *transport of the mantissa, exponent, denominator ledger, and provenance rows. Then e' is again a positive unary exponent row, ℓ' is the transported denominator ledger for $2^{e'}$, and every finite consumer window that read the denominator of D can read the denominator of D' through the transported* Cont *route.*

```
Lean4 ✓  BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier denominator ledger transport
Lean4 ✓  BEDC.Derived.DyadicRatCoreUp.DyadicRatCorePacket denominator ledger transport
Lean4 ✓  BEDC.Derived.DyadicRatCoreUp.DyadicRatCore denominator ledger transport window
Lean4 ✓  BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier denominator transport
```

Proof. Project the accepted carrier row of Definition 291.1. The positivity of e is a displayed unary-history row, so transporting it along the supplied hsame comparison gives the corresponding positive row for e'.

The denominator ledger is finite and is indexed by the same unary exponent tower. Transport each ledger entry along the displayed exponent comparison and repack the entries with the transported Pkg provenance.

Finally transport the finite Cont route used by a consumer window. The window reads only the visible denominator ledger entries, so the transported route reads exactly the denominator rows of D' and introduces no quotient or real endpoint. □

Theorem 291.8 (DyadicRatCore common-exponent window exactness)**.** *Let $D = (m, e, \ell, P)$ and $D' = (m', e', \ell', P')$ be accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets. Any displayed common-exponent witness*

for DyadicRatCoreClassifier(D, D') *determines a finite* BHist *window whose rows are exactly the common unary exponent row, the two transported denominator ledgers, the two scaled mantissa rows, the* Rat$^{\uparrow}$ *classifier row comparing* $m \cdot 2^{E-e}$ *with* $m' \cdot 2^{E-e'}$*, and the attached* Cont *and* Pkg *provenance. Conversely, every consumer-visible classifier read of the two packets factors through such a window. It has no row from which to read a quotient rational, a host integer normal form, or a* Real$^{\uparrow}$ *endpoint.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCommonExponentWindow exactness`
Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreClassifier common exponent transport`
Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier common exponent window exactness`

Proof. Unpack Definition 291.3. A classifier witness displays the common unary exponent E, the two scaling rows from e and e' to E, and the two denominator ledgers transported along those scaling rows.

The same witness also displays the scaled mantissas and the single Rat$^{\uparrow}$ classifier row comparing them. Repack these rows with the displayed Cont routes and the Pkg provenance inherited from the two accepted packets.

For the reverse direction, inspect the carrier inventory in Definition 291.1. A consumer-visible classifier read can project only the packet rows named there and the classifier rows named in Definition 291.3. Those rows are precisely the common exponent, denominator-ledger, scaled-mantissa, Rat$^{\uparrow}$, Cont, and Pkg rows above, so the read factors through the same finite window and cannot access any quotient rational, host integer normal form, or Real$^{\uparrow}$ endpoint. □

Theorem 291.9 (DyadicRatCore common-exponent e0 exclusion)**.** *Let* $D = (m, e, \ell, P)$ *and* $D' = (m', e', \ell', P')$ *be accepted* DyadicRatCore$^{\uparrow}$ *packets whose classifier is witnessed by a displayed common exponent row* E*. Then* E *cannot be an* e0 *denominator row. Equivalently, any classifier read that tries to use* e0 *as the common exponent lies outside the accepted dyadic classifier surface.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreClassifier common exponent e0 exclusion`

Proof. By Definition 291.3, a classifier witness displays the common exponent row together with the two denominator-ledger scaling rows and the Rat$^{\uparrow}$ comparison row. The exactness window of Definition 291.8 identifies this common exponent as part of the same finite positive-unary denominator ledger transported with the accepted packets.

The carrier discipline of Definition 291.1 admits denominator rows only through positive unary denominator data. The row e0 is excluded by that positive-unary discipline, so no accepted classifier witness can present e0 as its common exponent. □

Theorem 291.10 (DyadicRatCore distance-window stability)**.** *Let* D_0, D_1, D'_0, D'_1 *be accepted* DyadicRatCore$^{\uparrow}$ *packets such that* D_i *is transported to* D'_i *by displayed* hsame*,* Cont*, and* Pkg *rows for* $i = 0, 1$*. Any finite observation window that computes the dyadic distance* $|D_0 - D_1|$ *or the dyadic absolute value* $|D_0|$ *from common-exponent mantissa scaling is stable under this transport: the transported window computes* $|D'_0 - D'_1|$ *or* $|D'_0|$ *from the transported mantissa, exponent, denominator, and provenance rows.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier distance window stability`
Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier distance window transport`
Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier distance window transport closure`

Proof. Choose the common exponent displayed by the finite observation window. The classifier of Definition 291.3 makes the scaling rows visible, so the mantissa difference and sign split are ordinary $\mathsf{Rat}^{\uparrow}$ rows inside the same window.

Transport the two input packets componentwise through the supplied hsame rows and transport the scaling ledger through the corresponding Cont routes. The Pkg rows keep the same $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ source certificates attached to the transported packet.

The distance and absolute-value observations therefore recompute the same finite rational expression at the transported exponent. Since the window has no access to a quotient carrier or real completion endpoint, stability is exactly the repacking of those displayed rows. □

291.2 Certificate surface

Theorem 291.11 (DyadicRatCore arithmetic-window stability)**.** *Let D_0, D_1, D_0', D_1' be accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets such that each D_i is transported to D_i' by displayed* hsame*,* Cont*, and* Pkg *rows. Any finite common-exponent window that reads $D_0 + D_1$, $-D_0$, $D_0 - D_1$, or $D_0 D_1$ is stable under this transport: the transported window reads the corresponding operation on D_0', D_1' using only the displayed* $\mathsf{Rat}^{\uparrow}$ *mantissa rows,* UnaryHistory *exponent rows, denominator ledgers, common-exponent scaling rows,* hsame *comparisons,* Cont *routes, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreArithmeticWindow transport stability`

Proof. Choose the displayed common exponent carried by the arithmetic window. The window first projects the two mantissas to that exponent through the visible denominator ledgers and then applies the corresponding $\mathsf{Rat}^{\uparrow}$ row for sum, negation, difference, or product.

Transport the mantissa, exponent, denominator, and provenance coordinates of each input packet along the supplied hsame rows. The same finite Cont route transports the common-exponent scaling ledger, so the transported window has exactly the rows needed to repeat the arithmetic readback.

No step identifies rational pairs by a quotient or by host equality. The operation row is therefore recovered by recomposing the transported mantissa row with the transported unary denominator ledger and the same Pkg source certificates. □

Theorem 291.12 (DyadicRatCore monotone-radius obligation)**.** *For an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet $D = (m, e, \ell, P)$, the visible dyadic radius row 2^{-e} refines monotonically under any displayed unary extension $e \preceq E$. The refined radius row 2^{-E} is obtained by appending the finite denominator ledger entries from e to E, transporting the mantissa observation through the same* Cont *route, and retaining the same* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier scoped source boundary`

Proof. Project the positive unary exponent row from the carrier. A displayed extension $e \preceq E$ is a finite unary-history continuation, so the additional denominator entries form a visible tail of the same ledger.

Append that tail to the denominator ledger ℓ. The resulting row displays 2^E and hence the radius row 2^{-E}, with the original 2^{-e} radius as the coarser prefix read by the shorter window.

The mantissa row is not changed; only the denominator window is refined. Transporting the readback through the appended Cont route shows monotone radius refinement entirely inside the displayed unary ledger. □

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier monotone radius obligation`

Theorem 291.13 (DyadicRatCore scale-window obligation surface)**.** *The canonical common-exponent scale window, monotone radius obligation, denominator ledger, and arithmetic-window stability together provide the exact dyadic tolerance row consumed by* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity checks. The consumed row is the visible dyadic packet formed from the signed mantissa, positive unary exponent, denominator ledger, common-exponent scaling row, monotone radius refinement, arithmetic stability row,* BHist *carrier data,* hsame *transport,* Cont *replay, and* Pkg *provenance; it is not a quotient rational class, host rational equality, or host real distance.*

Proof. First use monotone-radius obligation, Definition 291.12. It refines the dyadic radius by appending visible unary denominator entries and keeps the radius row inside the displayed carrier and continuation surface.

Next use common-exponent exactness, Definition 291.15, together with the denominator ledger transport of Definition 291.7. These rows put the compared packets on one common exponent while preserving the visible mantissa and denominator-ledger sources.

Then apply arithmetic-window stability, Definition 291.11, and the RegSeqRat window-radius handoff, Definition 291.19. The arithmetic and radius windows are the same finite dyadic tolerance packet consumed by StreamName finite windows and RegSeqRat regularity checks.

The cited rows mention only displayed dyadic data, carrier histories, transports, continuations, and provenance. None provides quotient rational classes, host equality, or a host real distance coordinate, so the obligation surface is exact. □

Theorem 291.14 (DyadicRatCore arithmetic-monotone window package)**.** *Let* D_0, D_1, D_0', D_1' *be accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets satisfying the arithmetic-window transport hypotheses of Definition 291.11, and suppose the visible radius rows of the transported packet are refined along displayed unary exponent extensions as in Definition 291.12. Then the transported arithmetic endpoint and the refined radius ledger repack into one public finite window consumed by* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$*. This window reads only mantissa, exponent, denominator, common-exponent, distance and arithmetic,* BHist*,* hsame*,* Cont*, and* Pkg *rows, and it exposes no quotient rational, host equality, completion endpoint, or trichotomy row.*

Proof. First apply Definition 291.11. The transported arithmetic endpoint is read from the common-exponent scaling ledger, the displayed mantissas, the denominator rows, and the transported Pkg provenance.

Next apply Definition 291.12 to the visible unary extension of the radius row. The refined denominator ledger keeps the radius bound inside the same BHist and Cont surface.

The two ledgers have the same public coordinates: mantissa, exponent, denominator, common exponent, arithmetic or distance readback, history comparison, continuation, and package provenance. Repack those coordinates as the shared finite-window surface required by the RegSeqRat and $\mathsf{Real}^{\uparrow}$ consumers. Since neither input theorem exports quotient rational equality, host equality, a completion endpoint, or a trichotomy classifier, the combined window does not introduce any such row. □

Theorem 291.15 (DyadicRatCore common-exponent exactness)**.** *Suppose two classifier witnesses compare accepted packets* D *and* D' *by displayed common exponents* E *and* F*. Then there is a displayed unary exponent* G *extending both* E *and* F *such that transporting each witness to* G *yields the same* $\mathsf{Rat}^{\uparrow}$ *classifier row between the scaled mantissas of* D *and* D'*. The exactness claim is a statement about the displayed scaling ledgers and not about host rational equality.*

Proof. Project the two classifier witnesses of Definition 291.3. Each witness consists of a common unary exponent, two finite scaling ledgers from the packet exponents, and a $\mathsf{Rat}^{\uparrow}$ classifier row between the scaled mantissas.

Take the displayed unary exponent obtained by appending the two finite extension ledgers to a shared exponent G. Transport the E-witness and the F-witness along their denominator-ledger continuations into the same exponent row.

At G both witnesses read the identical scaled mantissa rows through the same $\mathsf{Rat}^{\uparrow}$ source certificate. Constructor inspection of the transported classifier rows therefore gives the same visible classifier row, with no appeal to quotient equality of dyadic pairs. □

Definition 291.16 (DyadicRatCore canonical comparison window)**.** *For accepted packets D and D' of* $\mathsf{DyadicRatCore}^{\uparrow}$*, the canonical comparison window is the finite BEDC packet*

$$(G,\ \Delta_D^G,\ \Delta_{D'}^G,\ M_D^G,\ M_{D'}^G,\ R_G,\ H_G,\ S_G,\ C_G,\ P_G,\ N_G).$$

Here G is a chosen common positive unary exponent, Δ_D^G and $\Delta_{D'}^G$ are the transported denominator ledgers, M_D^G and $M_{D'}^G$ are the scaled mantissa rows, and R_G is the displayed $\mathsf{Rat}^{\uparrow}$ *classifier row between those scaled mantissas. The remaining rows H_G, S_G, C_G, P_G, and N_G are the attached* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance carried by the accepted packets and their transport ledgers. The window is a finite BEDC comparison surface; it is not a quotient equality of dyadic pairs and not a host rational representative.*

Theorem 291.17 (DyadicRatCore canonical-window uniqueness)**.** *If two canonical comparison windows compare the same accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets D and D', then after displayed unary exponent refinement to a common terminal exponent they expose the same transported denominator ledgers, scaled mantissa rows,* $\mathsf{Rat}^{\uparrow}$ *classifier row, and support rows. No quotient rational, host equality, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or trichotomy row can distinguish the two windows.*

Proof. Unfold Definition 291.16. Each window consists of a common positive unary exponent, two transported denominator ledgers, two scaled mantissa rows, one $\mathsf{Rat}^{\uparrow}$ classifier row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Apply Definition 291.15 to the two classifier witnesses. Refining both displayed exponents to a common terminal exponent puts the two comparison rows on the same scaled mantissa surface.

Use Definition 291.7 to identify the transported denominator rows after the same exponent refinement. Use Definition 291.10 to keep the finite observation rows stable under the resulting hsame transport.

It remains only to read the carrier inventory. The available rows are exactly the mantissa, exponent, denominator, classifier, history, sameness, continuation, package, and name-certificate rows listed in Definition 291.1. The exclusions in Definition 291.22 remove quotient rational equality, host equality, completed real endpoints, and trichotomy classifiers, so no additional row can separate the two refined windows. □

Theorem 291.18 (DyadicRatCore real-phase source coverage)**.** *An accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet with transported mantissa and displayed* Cont *rows for the real row and the ledger window supplies exactly the real-phase source rows used here: the* $\mathsf{Real}^{\uparrow}$ *constant-history row at the transported mantissa head, the positive unary denominator row, the unary ledger row, the unary transported window row, the displayed readback* hsame *from the real row to the mantissa-head/exponent append, and the displayed readback* hsame *from the transported window to the real-row/ledger append.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier real phase source coverage`

Proof. Start with the accepted $\mathsf{DyadicRatCore}^{\uparrow}$ packet. Its carrier clause gives a $\mathsf{Rat}^{\uparrow}$ mantissa carrier, a positive unary denominator row, and a unary ledger row. Transport the mantissa along the displayed hsame row; the transported mantissa remains a $\mathsf{Rat}^{\uparrow}$ history carrier.

Taking the head of the transported mantissa yields the $\mathsf{Real}^{\uparrow}$ constant-history carrier. The displayed Cont row from that head and the exponent is read back as the real-row append, and the displayed Cont row from the real row and the ledger is read back as the window append.

Finally transport the displayed window along its hsame row. Unary closure of Cont gives the unary transported window, while the two append readbacks are exactly the two displayed hsame rows named in the statement. □

Theorem 291.19 (DyadicRatCore RegSeqRat window radius handoff)**.** *Every downstream* RegSeqRat *finite-window handoff that consumes a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation reads its radius premise through the displayed positive unary exponent, denominator ledger, monotone-radius refinement, distance-window stability,* hsame*,* Cont*, and* Pkg *rows of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *public envelope. Consequently the handoff cannot introduce a quotient rational, host epsilon equality, or* $\mathsf{Real}^{\uparrow}$ *completion endpoint.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreRegSeqRatWindow radius handoff`

Proof. Start with the downstream handoff and apply Definition 291.18 to obtain the real constant-history row, the positive unary denominator row, the unary ledger and window rows, and the two append readbacks.

Use Definition 291.12 to identify the radius premise as a displayed positive-unary refinement over the denominator ledger. Use Definition 291.10 to keep the finite distance window stable under the same hsame transport.

Finally apply Definition 291.23. Its public tuple supplies exactly the exponent, denominator, radius, distance, Cont, and Pkg coordinates consumed by RegSeqRat, so there is no remaining coordinate through which host epsilon equality or a $\mathsf{Real}^{\uparrow}$ completion endpoint could enter the handoff. □

Theorem 291.20 (DyadicRatCore NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *obligation surface consists of the signed mantissa carrier of Definition 291.1, the common-exponent scaling classifier of Definition 291.3, denominator ledger transport from Definition 291.7, dyadic distance and absolute-value window stability from Definition 291.10, arithmetic-window stability from Definition 291.11, monotone radius refinement from Definition 291.12, common-exponent exactness from Definition 291.15, real-phase source coverage from Definition 291.18, and exactness of the visible mantissa, exponent, denominator,* hsame*,* Cont*, and* Pkg *rows. It supplies a named dyadic source surface for* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*, and* $\mathsf{Real}^{\uparrow}$*, but it supplies no standard bridge and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier namecert obligation surface`

Proof. The carrier and classifier clauses are exactly Definition 291.1 and Definition 291.3. They determine which finite BEDC rows may be read as a dyadic packet and how two such packets are compared by common-exponent scaling.

The ledger clause is Definition 291.7; it keeps the positive unary denominator tower visible under hsame transport. The observation clause is Definition 291.10; it keeps distance and absolute-value windows inside the displayed $\mathsf{Rat}^{\uparrow}$ and unary ledger rows.

The arithmetic, radius, common-exponent, and real-phase source clauses are Definition 291.11, Definition 291.12, Definition 291.15, and Definition 291.18. Together they identify the finite arithmetic rows, the radius refinement rows, the common-exponent comparison rows, and the real-row/window readbacks used by downstream real-construction chapters.

Exactness follows by inspection of the listed rows: every readable component of the dyadic source is one of the mantissa, exponent, denominator, provenance, transport, or continuation entries. Thus the surface is a seed source for later real-construction chapters and not a bridge to host rationals or completed reals. □

Theorem 291.21 (DyadicRatCore source-boundary SemanticNameCert)**.** *For any accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier with mantissa* m*, exponent* e*, ledger* ℓ*, and provenance packet* p*, the source-boundary predicate*

$$\mathsf{row} \in \mathsf{Source}_{m,e,p} \iff \exists \ell'.\, \mathsf{DyadicRatCore}^{\uparrow}(m, e, \ell', p) \wedge \mathsf{hsame}(\mathsf{row}, \ell')$$

carries a SemanticNameCert *whose carrier, pattern, and ledger predicates are this same boundary predicate and whose equality relation is* hsame*. The same certificate exposes the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa carrier, positive* UnaryHistory *exponent, unary ledger, and unary provenance rows. It does not identify dyadic packets by quotient rational equality and does not introduce a completed* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreSourceBoundary semanticNameCert`

Proof. Begin with the accepted carrier of Definition 291.1. Its visible coordinates give the signed $\mathsf{Rat}^{\uparrow}$ mantissa, the positive unary exponent, the unary ledger, and the unary provenance packet. The source-boundary row is then the ledger component transported by hsame inside the same mantissa, exponent, and provenance packet.

The certificate fields are the rows already packaged by Definition 291.20 and the scoped boundary inventory below. Reflexivity, symmetry, transitivity, and carrier transport are exactly the displayed hsame rows; pattern soundness and ledger soundness are the same witness read in the pattern and ledger positions.

Thus the SemanticNameCert is a certificate on the finite source boundary alone. All data used by the proof are the displayed carrier, hsame, ledger, and provenance rows, so no quotient rational equality or completed $\mathsf{Real}^{\uparrow}$ endpoint enters the construction. □

Theorem 291.22 (DyadicRatCore scoped source boundary)**.** *The scoped* $\mathsf{DyadicRatCore}^{\uparrow}$ *source boundary is exactly the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, the positive* UnaryHistory *exponent row, the denominator ledger, the common-exponent scaling classifier, the arithmetic-window and distance-window rows, monotone-radius refinement, common-exponent exactness, real-phase source coverage,* BHist *carrier data,* hsame *transport,* Cont *readback, and* Pkg *provenance listed in this chapter. It exports no quotient of dyadic pairs, host rational equality, standard bridge, or completed* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. The carrier and classifier boundaries are the displayed rows of Definition 291.1 and Definition 291.3. They supply the signed mantissa, positive unary exponent, denominator tower, BHist tuple, hsame transport, Cont readback, and Pkg provenance that can be read from an accepted dyadic packet.

The denominator, arithmetic, distance, radius, and common-exponent clauses are respectively Definition 291.7, Definition 291.11, Definition 291.10, Definition 291.12, and Definition 291.15. These rows say how the same visible packet supports finite dyadic computation under displayed unary denominator transport.

The real-phase and NameCert clauses are Definition 291.18 and Definition 291.20. The first contributes the real constant-history row, unary ledger/window rows, and append readbacks; the second packages those rows with the separate arithmetic, distance, radius, classifier, and provenance obligations. Hence the listed primitives are precisely the scoped boundary exported by this chapter. □

Theorem 291.23 (DyadicRatCore public readback envelope)**.** *The public* DyadicRatCore$^{\uparrow}$ *consumer interface is exactly the tuple consisting of the signed mantissa carrier, common-exponent classifier, denominator ledger transport, arithmetic-window stability, distance-window stability, monotone-radius refinement, common-exponent exactness, real-phase source coverage,* BHist *carrier rows,* hsame *transport,* Cont *readback, and* Pkg *provenance. Every downstream* StreamName$^{\uparrow}$*,* RegSeqRat*, or* Real$^{\uparrow}$ *dyadic read factors through this envelope and cannot read a quotient rational, host rational equality, standard bridge, or completed real endpoint.*

Proof. Start with an arbitrary public consumer read of a DyadicRatCore$^{\uparrow}$ packet. By Definition 291.1 and Definition 291.3, the only carrier and comparison rows available to that read are the signed mantissa, positive unary exponent, denominator ledger, common-exponent scaling witness, BHist data, hsame rows, Cont routes, and Pkg provenance.

The finite denominator, arithmetic, distance, radius, and exactness reads are then forced through the named certificate rows. Use Definition 291.7, Definition 291.11, Definition 291.10, Definition 291.12, and Definition 291.15 to identify the visible rows that remain available under transport and denominator refinement.

Finally apply Definition 291.18 for the real constant-history row, unary ledger/window rows, and append readbacks, and apply Definition 291.22 for the full public dyadic source surface. Since the carrier inventory contains no quotient rational, host equality row, standard bridge, or completed real endpoint, none of those objects can be exported by the public consumer certificate. □

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier public readback envelope`

Theorem 291.24 (DyadicRatCore public bridge consumer exactness)**.** *Every standard dyadic-source consumer admitted through the public bridge reads exactly the signed* Rat$^{\uparrow}$ *mantissa row, positive* UnaryHistory *exponent row, denominator ledger, common-exponent classifier, arithmetic window, distance window, radius window,* BHist *carrier data,* hsame *transport,* Cont *readback, and* Pkg *provenance already enumerated by Definition 291.23. Conversely, every public envelope row with precisely those coordinates supplies the finite observation interface consumed by* StreamName$^{\uparrow}$*,* RegSeqRat*, and* Real$^{\uparrow}$*. The bridge exports no quotient rational, host equality row, completion object, or* Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier public bridge consumer exactness`

Proof. For the consumer-to-envelope direction, apply Definition 291.23. It lists exactly the mantissa, exponent, denominator, classifier, finite-window, carrier, transport, readback, and provenance rows visible to a public DyadicRatCore$^{\uparrow}$ read.

The bridge admission cannot enlarge that inventory. Use Definition 291.22 to restrict the downstream StreamName$^{\uparrow}$, RegSeqRat, and Real$^{\uparrow}$ reads to the displayed dyadic source rows, and use Definition 38.272 for the standard dyadic-source shape consumed at the bridge boundary.

For the envelope-to-consumer direction, start with one of the public envelope rows. Its signed mantissa, positive unary exponent, denominator ledger, common-exponent classifier, finite arithmetic and distance windows, radius refinement, BHist rows, hsame transport, Cont readback,

and Pkg provenance are exactly the coordinates requested by the standard dyadic-source bridge. Repacking them gives the finite observation interface used by $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Real}^{\uparrow}$.

The exclusion clauses are constructor inspections of the same public envelope and bridge boundary. Neither theorem supplies a quotient-rational row, a host equality row, a completion object, or a real-completeness field, so no admitted consumer can read one. □

Theorem 291.25 (DyadicRatCore standard source bridge)**.** *The public envelope of Definition 291.23 determines a bridge packet whose BEDC side is exactly the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, positive* UnaryHistory *exponent row, denominator ledger, common-exponent classifier, arithmetic and distance windows, monotone-radius refinement,* BHist *data,* hsame *transport,* Cont *readback, and* Pkg *provenance carried by the envelope. Its standard side is the usual dyadic source interface* (m, e) *with positive unary exponent and common-exponent comparison. The bridge exports no quotient rational, host rational equality, completed real, or real completeness claim.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier standard source bridge`

Proof. Apply Definition 291.23 to obtain the public tuple. Its visible coordinates give the signed mantissa, positive unary exponent, denominator ledger, finite arithmetic and distance windows, BHist data, hsame transport, Cont readback, and dependency provenance required on the BEDC side.

Use Definition 291.15 to read the standard comparison row. It identifies comparison of two displayed dyadic packets with comparison after lifting their mantissas to a common positive unary exponent, which is exactly the common-exponent comparison expected by the usual source schema.

Use Definition 291.12 to keep radius and distance observations stable under denominator refinement. Then apply Definition 291.22 to check that the downstream $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Real}^{\uparrow}$ boundary consumes only this dyadic source surface. Constructor inspection of the same envelope shows that there is no row for a quotient rational, host rational equality, completed real, or real completeness theorem. □

Theorem 291.26 (DyadicRatCore standard bridge denominator-refinement stability)**.** *Let* P *and* P' *be public* $\mathsf{DyadicRatCore}^{\uparrow}$ *bridge packets. Suppose that* P' *is obtained from* P *by a displayed positive-unary denominator refinement: the exponent row is extended inside* UnaryHistory*, the denominator ledger is the corresponding tower for the refined power of* 2*, and the public* hsame *and* Cont *rows exhibit the same dyadic source after the displayed rescaling. Then the standard dyadic source bridge of Definition 291.25 transports exactly the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, common-exponent classifier, distance window, monotone-radius row,* BHist *carrier data,* hsame *transport,* Cont *readback, and* Pkg *provenance from* P *to* P'*. The transported bridge packet exports no quotient rational, host equality, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or real-completeness row.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier standard bridge denominator refinement stability`

Proof. Start with the public envelope of Definition 291.23. It exposes precisely the mantissa, positive unary exponent, denominator ledger, classifier, finite window, carrier, transport, readback, and provenance rows available to a public $\mathsf{DyadicRatCore}^{\uparrow}$ packet.

Apply Definition 291.15 to compare P and P' after lifting their mantissas to the displayed common positive unary exponent. The hypothesis that the refinement is shown inside UnaryHistory supplies the same common-exponent classifier row for the refined packet.

Next use Definition 291.12. The positive-unary denominator refinement can only pass through the displayed monotone-radius row, so the finite distance window and radius observation are read at the refined denominator without introducing a new ambient rational object.

The public bridge consumer exactness theorem, Definition 291.24, transports the remaining classifier windows through the named hsame and Cont rows. Thus the carrier data, readback, and dependency provenance seen by the consumer of P' are the same public rows repackaged at the refined exponent.

Finally repackage those rows by Definition 291.25. Its exclusion clause is unchanged under the refinement: none of the cited public rows contains a quotient-rational representative, a host equality witness, a completed $\mathsf{Real}^{\uparrow}$ endpoint, or a real-completeness field. □

Theorem 291.27 (DyadicRatCore standard bridge denominator-refinement composition)**.** *Let P, P', and P'' be public* $\mathsf{DyadicRatCore}^{\uparrow}$ *bridge packets. Suppose that P' is obtained from P by a displayed positive-unary denominator refinement, and that P'' is obtained from P' by another displayed positive-unary denominator refinement. Then the standard dyadic source bridge of Definition 291.25 transports from P to P'' by composing the two displayed* UnaryHistory *exponent extensions, denominator ledgers, common-exponent classifier rows, distance and radius windows,* BHist *carrier data,* hsame *transports,* Cont *readbacks, and* Pkg *provenance rows. The composed transport exports no quotient rational, host equality, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier standard bridge denominator refinement composition`

Proof. Apply Definition 291.26 to the refinement from P to P'. This gives the transported signed mantissa, common-exponent classifier, distance window, monotone-radius row, carrier data, transport, readback, and provenance at the first refined positive-unary exponent.

Apply the same theorem to the refinement from P' to P''. The second application transports exactly the same public bridge inventory through the second displayed exponent extension and denominator ledger.

Use Definition 291.15 to identify the classifier row for the two-step exponent extension with the classifier row obtained by composing the two displayed common-exponent lifts. Use Definition 291.12 to read the finite distance and radius windows through the composed denominator refinement.

The public rows obtained after the two transports are still the rows listed by Definition 291.24. Repacking them through Definition 291.25 gives the standard dyadic source bridge from P directly to P''.

The exclusion clause is inherited from the two single-step bridge packets and from the public source bridge. Neither step introduces a quotient-rational representative, host equality witness, completed $\mathsf{Real}^{\uparrow}$ endpoint, or real-completeness field, so the composed bridge cannot export one. □

Theorem 291.28 (DyadicRatCore RegSeqRat bridge handoff exactness)**.** *Every* RegSeqRat *finite-window handoff that consumes a* $\mathsf{DyadicRatCore}^{\uparrow}$ *public envelope through the standard source bridge reads exactly the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, positive* UnaryHistory *exponent row, denominator ledger, common-exponent classifier, arithmetic window, distance window, radius window,* BHist *carrier data,* hsame *transport,* Cont *readback, and* Pkg *provenance listed by Definition 291.23. Conversely, every public envelope row with those visible coordinates supplies the dyadic observation interface*

admitted by the RegSeqRat *handoff. The handoff exports no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier regseqrat bridge handoff exactness`

Proof. For the forward direction, apply Definition 291.19. It identifies the downstream handoff boundary as a finite read of the displayed dyadic source rows and excludes quotient rationals, host epsilon equality, and completed $\mathsf{Real}^{\uparrow}$ endpoints from that boundary.

Next apply Definition 291.23 and Definition 291.24. These results give the exact public inventory: the signed mantissa, positive unary exponent, denominator ledger, classifier, arithmetic, distance, and radius windows, BHist data, hsame transport, Cont readback, and Pkg provenance. Use Definition 291.25 to read the same inventory as the standard dyadic source packet consumed at the bridge boundary.

For the converse direction, start with a public envelope row carrying precisely those coordinates. Repack the signed mantissa, positive unary exponent, denominator ledger, common-exponent classifier, finite windows, carrier data, transport, readback, and provenance as the dyadic observation interface already admitted by the standard source bridge and by the RegSeqRat handoff boundary.

The exclusion clauses are the same carrier-inventory inspection. None of the referenced rows is a quotient-rational representative, a host equality proof, a completed $\mathsf{Real}^{\uparrow}$ endpoint, or a real-completeness field, so neither direction can export one. □

Theorem 291.29 (DyadicRatCore RegSeqRat radius monotone-window extraction)**.** *Every* RegSeqRat *finite-window handoff consuming a* $\mathsf{DyadicRatCore}^{\uparrow}$ *public envelope through the standard source bridge exposes a monotone unary radius window whose denominator ledger, common-exponent classifier, distance window,* BHist *carrier row,* hsame *transport,* Cont *readback, and* Pkg *provenance are exactly the rows listed by Definition 291.12, Definition 291.10, Definition 291.25, and Definition 291.28. The extraction exports no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or standard bridge data beyond the finite bridge packet already displayed.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier regseqrat radius monotone window extraction`

Proof. Project the RegSeqRat handoff through the standard source bridge of Definition 291.25. The bridge exposes the signed mantissa row, positive unary exponent row, denominator ledger, common-exponent classifier, distance window, carrier data, transport rows, readback rows, and package provenance of the public envelope.

Read the distance and radius windows from that projection. The distance component is the finite row of Definition 291.10, while the radius component is the positive-unary denominator row refined by Definition 291.12.

Use the monotonicity clause to refine the requested unary radius window. Appending the displayed denominator tail preserves the same mantissa observation, common-exponent classifier, BHist carrier row, hsame transport, Cont readback, and Pkg provenance.

Finally apply Definition 291.28. Its exclusion clause rules out quotient rationals, host equality, completed real endpoints, and any standard bridge data not present in the finite window, so the extracted radius window is exactly the displayed dyadic-to-RegSeqRat window. □

Theorem 291.30 (DyadicRatCore RegSeqRat radius-window composition)**.** *Two successive* RegSeqRat *finite-window handoffs that consume* $\mathsf{DyadicRatCore}^{\uparrow}$ *public envelopes through the standard source bridge compose to a single monotone radius-window extraction. The visible rows of the composed handoff are exactly the denominator ledger, common-exponent classifier, distance window,* BHist

carrier row, hsame *transport,* Cont *readback, and* Pkg *provenance from the first and second windows, read through the same standard dyadic source bridge. The composed extraction exports no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier regseqrat radius window composition`

Proof. First compose the two dyadic bridge packets by Definition 291.27. This identifies the denominator refinement produced by the first handoff with the source denominator row consumed by the second, and it keeps the common-exponent classifier and distance window inside the displayed dyadic bridge inventory.

Next apply Definition 291.28 to each handoff. The only rows visible to the downstream RegSeqRat consumer are the denominator ledger, classifier, finite distance and radius windows, carrier data, transport rows, readback rows, and package provenance.

Finally apply Definition 291.29 to the composed packet. Monotonicity turns the two finite radius reads into one radius-window extraction, and the inherited exclusion clauses remove quotient rationals, host equality, completed real endpoints, and real-completeness data from the composed boundary. □

Theorem 291.31 (DyadicRatCore directed refinement common window)**.** *Let P, P', and P'' be public* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets. Suppose two displayed denominator-refinement routes through the standard source bridge carry P to P' and carry P' to P''. Then any two* RegSeqRat *finite-window handoffs consuming the resulting refined packets factor through one common monotone unary radius window.*

The rows of this common window are exactly the signed mantissa row, refined exponent row, refined denominator ledger, common-exponent classifier, distance-window rows, BHist *carrier rows,* hsame *transport rows,* Cont *readback rows, and* Pkg *provenance rows named by Definition 291.25, Definition 291.27, Definition 291.28, and Definition 291.30. The common window exports no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier directed refinement common window`

Proof. First compose the two displayed denominator-refinement routes by Definition 291.27. The composed bridge route identifies the refined denominator ledger carried by P' with the source denominator ledger consumed by the route into P'', while leaving the signed mantissa, refined exponent, common-exponent classifier, distance rows, carrier rows, transport rows, readback rows, and package provenance inside the displayed standard source inventory.

Next apply Definition 291.28 to the two RegSeqRat handoffs. Each handoff reads only the finite dyadic bridge rows: signed mantissas, refined unary exponents, denominator ledgers, common-exponent classifiers, distance and radius windows, BHist carrier data, hsame transport, Cont readback, and Pkg provenance.

Now use Definition 291.30 for the two handoff windows, and then Definition 291.29 for the composed packet. The resulting monotone unary radius window is common to both routes because both are read through the same composed denominator-refinement ledger.

Finally inspect the exclusion clauses of Definition 291.23 and Definition 291.25. Neither envelope has a row for quotient rationals, host equality, completed real endpoints, or real completeness, so no RegSeqRat handoff factored through the common window can export such data. □

Theorem 291.32 (DyadicRatCore directed refinement radius handoff)**.** *Every directed-refinement common-window read of* $\mathsf{DyadicRatCore}^{\uparrow}$ *that feeds a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exposes the common exponent, transported denominator ledgers, dyadic radius row, monotone radius witness,*

and $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before any* $\mathsf{Real}^{\uparrow}$ *seal can consume it. The handoff exports no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier directed refinement radius handoff`

Proof. Apply Definition 291.31. It places both refined packets on one common monotone unary radius window with the shared exponent, denominator ledger, classifier, distance rows, transport, readback, and provenance already visible.

Use Definition 291.29 to read the dyadic radius row and monotone radius witness from that common window.

Finally apply Definition 291.19. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives the finite radius handoff before any $\mathsf{Real}^{\uparrow}$ seal can inspect it, and the displayed carrier rows contain no quotient rational, host equality row, completed real endpoint, or real-completeness coordinate. □

Theorem 291.33 (DyadicRatCore finite radius-window cofinality)**.** *Every finite family of public* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets whose* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumers read them through displayed directed-refinement radius handoffs factors through one common monotone unary radius window. The common window carries the shared refined exponent, denominator ledger, common-exponent classifier, distance rows,* BHist *carrier rows,* hsame *transports,* Cont *readbacks, and* Pkg *provenance for the whole finite family.*

The common window exports no quotient rational, host equality row, completed $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice selector, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreFiniteRadiusWindowCofinality`

Proof. Induct on the finite family of displayed directed-refinement handoffs. For the empty family, take the vacuous finite window with no consumer rows. For a singleton, apply Definition 291.29 to the single public handoff and read its monotone unary radius window.

For the induction step, assume the first finite subfamily has already been placed on one common monotone unary radius window. Compose that accumulated window with the next directed-refinement handoff by Definition 291.27 and Definition 291.30. The composition keeps the refined exponent, denominator ledger, classifier, distance rows, carrier data, transports, readbacks, and provenance inside the same displayed dyadic bridge inventory.

Apply Definition 291.31 to identify the composed bridge packet with a common monotone unary radius window for the accumulated subfamily and the new handoff. The induction invariant is preserved because the window is enlarged only by displayed denominator-refinement rows and finite $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows.

Finally use Definition 291.32 at the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer boundary. Its exclusion clause, together with the exclusions in the bridge and radius-window composition results, rules out quotient rationals, host equality rows, completed real endpoints, countable-choice selectors, and real-completeness fields from the cofinal window. □

Theorem 291.34 (DyadicRatCore finite radius-window directed meet)**.** *Let two displayed common monotone unary radius windows be obtained from the same finite family of public* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets and directed-refinement handoffs. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *terminal consumer of either window factors through their displayed directed meet. The meet carries exactly the shared refined exponent, denominator ledger, common-exponent classifier, distance rows,* BHist *carrier rows,* hsame *transports,* Cont *readbacks,* Pkg *provenance, and local NameCert row.*

No quotient rational, host equality row, selected $\mathsf{Real}^{\uparrow}$ *limit, countable-choice selector, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row distinguishes the two displayed windows.*

Proof. Apply Definition 291.33 to each finite-family presentation. Both presentations therefore expose the same finite public packet family through a common monotone unary radius window with the refined exponent, denominator ledger, classifier, distance rows, carrier rows, transports, readbacks, and provenance already displayed.

Compose the two denominator-refinement routes by Definition 291.27. Then apply Definition 291.31 to identify the composed handoff as a common directed-refinement window for the two displays. This common handoff is the displayed directed meet because both terminal routes are read through the same finite family and the same refined denominator ledger.

Finally use Definition 291.32 and Definition 291.37 at the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary. Their non-escape clauses leave only the displayed dyadic rows available to terminal consumers, so no quotient rational, host equality row, selected real limit, countable-choice selector, metric-completeness datum, or ambient real-completeness row can distinguish the two windows after passage to the meet. □

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier finite radius window directed meet`

Theorem 291.35 (DyadicRatCore finite radius-window filtered meet)**.** *Let a finite family of public* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets be equipped with displayed directed-refinement radius handoffs into* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *terminal consumers, and let a retained finite subfamily of those terminal consumers be selected. The retained consumers factor through the displayed directed meet of the restricted monotone unary radius windows obtained from the retained handoffs. The visible rows of that restricted meet are exactly the refined exponent, denominator ledger, common-exponent classifier, distance rows,* BHist *carrier rows,* hsame *transports,* Cont *readbacks,* Pkg *provenance, and local* NameCert *rows inherited from the ambient finite-family meet.*

The restricted meet exports no quotient rational, selected $\mathsf{Real}^{\uparrow}$ *limit, countable-choice selector, metric-completeness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, or aggregate bridge/status row.*

Proof. Apply Definition 291.33 to the retained finite family of displayed handoffs. This supplies a common monotone unary radius window whose rows are restricted to the retained consumers while keeping the refined exponent, denominator ledger, classifier, distance rows, carrier rows, transports, readbacks, provenance, and local certificate rows inside the displayed dyadic inventory.

Compare this restricted window with the ambient finite-family window by Definition 291.34. Since the retained handoffs are a subfamily of the ambient presentation, the directed meet is obtained by forgetting only the consumer rows outside the retained subfamily and by keeping the common denominator-refinement rows already present in the ambient meet.

Finally use Definition 291.32 and Definition 291.37 at the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary. These results force every retained terminal consumer to read through the finite radius handoff and exclude quotient rationals, selected real limits, countable-choice selectors, metric-completeness data, ambient real-completeness rows, and aggregate bridge/status rows from the restricted meet. □

Theorem 291.36 (DyadicRatCore finite radius-window meet associativity)**.** *Let* W_0*,* W_1*, and* W_2 *be three displayed common monotone unary radius windows over the same finite family of public* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets and directed-refinement handoffs. The displayed meet obtained by first meeting* W_0 *with* W_1 *and then meeting that result with* W_2 *exposes the same finite BEDC rows as the displayed meet obtained by first meeting* W_1 *with* W_2 *and then meeting* W_0 *with that result.*

The common row inventory is the refined exponent, denominator ledger, common-exponent classifier, distance rows, BHist *carrier rows,* hsame *transports,* Cont *readbacks,* Pkg *provenance, local* NameCert *row, and the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *terminal faces.*

No quotient rational, host equality row, selected $\mathsf{Real}^{\uparrow}$ *limit, countable-choice selector, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row distinguishes the two parenthesizations.*

Proof. Apply Definition 291.33 to the finite family underlying W_0, W_1, and W_2. Each displayed window is therefore a presentation of the same finite dyadic packet family through the same kind of monotone unary radius handoff, with the refined exponent, denominator ledger, classifier, distance rows, carrier rows, transports, readbacks, provenance, and local certificate row already visible.

Next apply Definition 291.34 to the pair W_0, W_1 and then to its displayed result with W_2. Apply the same directed-meet theorem to the pair W_1, W_2 and then to W_0 with that displayed result. In both orders the meet is computed over the identical finite family of public packets and directed-refinement handoffs, so the displayed row inventory is the directed meet of the same finite data rather than a datum depending on parentheses.

The retained consumer faces are checked by Definition 291.35. Filtering either parenthesized meet to the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ terminal faces forgets the same non-retained rows and preserves the same refined denominator, classifier, carrier, transport, readback, provenance, and certificate rows.

Finally use Definition 291.37 at the terminal boundary. Since both parenthesizations restrict to the same displayed directed meet of the same finite family, their terminal consumers cannot be separated by quotient rationals, host equality rows, selected limits, countable-choice selectors, metric-completeness data, or ambient real completeness. □

Theorem 291.37 (DyadicRatCore RegSeqRat terminal radius-window boundary)**.** *Every Phase Real and Completion terminal handoff whose dyadic face is a public* $\mathsf{DyadicRatCore}^{\uparrow}$ *envelope and whose downstream consumer is a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal factors through the common monotone unary radius window supplied by Definition 291.33. The visible rows of that terminal boundary are exactly the shared refined exponent, denominator ledger, common-exponent classifier, distance rows,* BHist *carrier rows,* hsame *transports,* Cont *readbacks, and* Pkg *provenance read through Definition 291.28 and Definition 291.32.*

The boundary exports no quotient rational, host equality row, selected real limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, or bridge row beyond the displayed dyadic source bridge.*

Proof. First place the finite family of terminal dyadic handoffs on the common monotone unary radius window of Definition 291.33. This identifies one shared refined exponent, denominator ledger, common-exponent classifier, distance inventory, BHist carrier inventory, transport inventory, readback inventory, and provenance inventory for the dyadic faces consumed at the terminal boundary.

Next apply Definition 291.32. The directed-refinement handoff exposes the dyadic radius row, monotone radius witness, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff before any $\mathsf{Real}^{\uparrow}$ seal can consume the packet, so the terminal read is forced to pass through the displayed finite radius window.

Use Definition 291.30 to compose the finite radius window with the downstream readback route. Then apply Definition 291.28 to read the consumer-visible rows from the public bridge inventory rather than from an external rational quotient or completed real object.

Finally inspect the exclusion clauses of the finite cofinality theorem, directed-refinement radius handoff, and bridge handoff exactness theorem. Their shared inventory contains only displayed dyadic source bridge rows and finite $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, hence it has no quotient rational, host equality row, selected real limit, countable-choice selector, ambient real-completeness coordinate, metric-completeness coordinate, or extra bridge row. □

Theorem 291.38 (DyadicRatCore open-phase exit criterion)**.** *A current Phase Real and Completion exit read whose dyadic face is a public* DyadicRatCore$^{\uparrow}$ *envelope is admissible if and only if its visible data factor through the displayed dyadic source rows, the* StreamName$^{\uparrow}$ *finite-window face, the* RegSeqRat$^{\uparrow}$ *readback face, and the local* Real$^{\uparrow}$ *seal face. The only support rows carried through this route are* BHist *carrier rows,* hsame *transports,* Cont *readbacks,* Pkg *provenance, and* NameCert *certificate rows already attached to the public dyadic envelope.*

The exit route exposes no quotient rational representative, host equality row, selected limit, countable-choice selector, ambient Real$^{\uparrow}$ *completeness theorem, metric-completeness row, or standard bridge row beyond the displayed dyadic source rows. The* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and local* Real$^{\uparrow}$ *faces are downstream consumer contexts for the same finite dyadic inventory, not additional dyadic coordinates.*

Proof. For the forward direction, begin with the public dyadic envelope and apply Definition 291.24. This identifies the consumer-visible carrier, transport, readback, package, and certificate rows without introducing a host rational equality or completed real endpoint.

Next apply Definition 291.25 to read the same envelope through its displayed dyadic source rows. The bridge is used only to name the finite signed mantissa, exponent, denominator, classifier, and distance inventory; no additional standard bridge row is exported to the exit read.

Then apply Definition 291.28 and Definition 291.37. These two results place the RegSeqRat$^{\uparrow}$ readback and the local Real$^{\uparrow}$ seal on the same terminal finite radius window, so the StreamName$^{\uparrow}$ finite-window face, RegSeqRat$^{\uparrow}$ readback face, and local Real$^{\uparrow}$ seal face consume one displayed dyadic inventory.

For the reverse direction, any exit read already factored through those four faces has exactly the public dyadic support rows named above. Repacking these rows through the public bridge consumer exactness theorem gives an admissible Phase Real and Completion exit read, while the exclusion clauses of the source bridge, handoff exactness, and terminal boundary theorems rule out quotient rational representatives, host equality, selected limits, countable choice, ambient real completeness, metric-completeness rows, and any extra standard bridge coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreOpenPhaseExitCriterion`

Theorem 291.39 (DyadicRatCore RegSeqRat handoff failure localization)**.** *For a current Phase Real and Completion exit read whose dyadic face is a public* DyadicRatCore$^{\uparrow}$ *envelope, the* RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *handoff fails exactly when the finite public route has an obstruction at one of the displayed sites:*

1. *a dyadic source row named by the public bridge is absent, or its signed mantissa, positive exponent, denominator ledger, common-exponent classifier, or distance component is not transported componentwise;*
2. *the* StreamName$^{\uparrow}$ *finite-window face does not carry the same finite radius and source-row window as the dyadic bridge;*
3. *the* RegSeqRat$^{\uparrow}$ *readback face is absent, or it reads a row outside the terminal radius window;*
4. *the local* Real$^{\uparrow}$ *seal face is absent, or it seals data not read back from the terminal* RegSeqRat$^{\uparrow}$ *face;*
5. *one of the support rows* BHist, hsame, Cont, Pkg, *or* NameCert *is absent, or is not transported componentwise along the public envelope.*

Equivalently, once all five finite sites are present and componentwise transported, the open Phase Real and Completion exit read is admissible and the handoff has no further obstruction. No quotient

rational representative, host equality row, selected limit, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness row, or bridge row beyond the displayed dyadic source rows can witness failure.*

Proof. For the failure direction, apply Definition 291.38. A non-admissible exit read cannot factor through the displayed dyadic source rows, the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, the local $\mathsf{Real}^{\uparrow}$ seal face, and the listed support rows at once.

Use Definition 291.24 to inspect the public envelope. If the carrier, transport, readback, package, or certificate support is missing, the obstruction is the fifth listed site; otherwise these rows are present and componentwise transported.

Next use Definition 291.25 to inspect the displayed dyadic source inventory. If one of its source components is absent or not transported componentwise, the obstruction is the first listed site. Otherwise the signed mantissa, positive exponent, denominator ledger, common-exponent classifier, and distance rows are the public source rows of the same envelope.

It remains to locate any downstream obstruction. Apply Definition 291.28 and Definition 291.37. These results force the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the local $\mathsf{Real}^{\uparrow}$ seal face to consume the same terminal finite radius window, so a remaining failure is exactly one of the second, third, or fourth listed sites.

For the converse direction, assume one listed site is absent or not componentwise transported. Then the exit read cannot factor through all four finite faces and all support rows, so Definition 291.38 rejects admissibility of the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff.

If none of the five listed sites occurs, the same criterion gives admissibility. The exclusion clauses in Definition 291.24, Definition 291.25, Definition 291.28, and Definition 291.37 rule out quotient rational representatives, host equality, selected limits, countable choice, ambient real completeness, metric completeness, and extra bridge rows as possible obstruction witnesses. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$.

Constructive story: A dyadic packet is generated by placing a signed $\mathsf{Rat}^{\uparrow}$ mantissa beside a positive unary exponent row, displaying the finite denominator tower for 2^e, and attaching Pkg provenance for the $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ sources. The classifier compares two packets by lifting them to a displayed common unary exponent and reading a $\mathsf{Rat}^{\uparrow}$ classifier row between the scaled mantissas. Distance and absolute-value observations are finite common-exponent windows over the same displayed rows. The surface records the dyadic source needed by $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, and $\mathsf{Real}^{\uparrow}$ without forming quotient equality, importing host rationals, or claiming real completeness.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped paper boundary binds the signed $\mathsf{Rat}^{\uparrow}$ mantissa row, positive UnaryHistory exponent row, denominator ledger, common-exponent scaling classifier, arithmetic and distance windows, monotone-radius refinement, common-exponent exactness, real-phase source coverage, BHist data, hsame transport, Cont readback, and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier\_standard\_source\_bridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not claim a host rational quotient, arithmetic completeness, completed real endpoint, or any $\mathsf{Real}^{\uparrow}$ completeness theorem.

Upgrade path: No further upgrade – chapter at top of formal axis.

Definition 291.40 (DyadicRatCore scale-window packet)**.** *The DyadicRatCore scale-window packet is the finite dyadic observation packet consisting of a signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, a positive* UnaryHistory *exponent row, the comparison window at that exponent, the scale transport row to any displayed wider exponent window, the componentwise* hsame *rows comparing the transported dyadic observations, the displayed* Cont *replay row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *naming row.*

The packet is consumed by $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *as a finite dyadic source. It carries no host rational order and no quotient equality coordinate.*

Theorem 291.41 (DyadicRatCore scale-window monotonicity)**.** *Widening the positive unary exponent window of a DyadicRatCore scale-window packet preserves the dyadic comparison row consumed by* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*. The preserved row is transported only by the displayed scale transport, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows of Definition 291.40.*

No host rational order, quotient equality, completed Real endpoint, or ambient completeness row is imported by the widened window.

Proof. Start with Definition 291.26. It transports the signed mantissa, common-exponent classifier, distance window, radius row, carrier data, hsame transport, Cont readback, and provenance across a displayed positive-unary denominator refinement.

Next apply Definition 291.27. Repeated widening of the exponent window is therefore the composite of the same displayed finite transports, not a comparison in an ambient host order.

Use Definition 291.28. It identifies the dyadic comparison rows consumed by $\mathsf{RegSeqRat}^{\uparrow}$ with exactly the public DyadicRatCore envelope rows, and the same finite source surface is available to $\mathsf{StreamName}^{\uparrow}$ through the public bridge consumer boundary.

The rows left by these references are precisely the rows of Definition 291.40. Their exclusion clauses remove host rational order, quotient equality, completed Real endpoints, and ambient completeness from the widened window. □

Theorem 291.42 (DyadicRatCore scale-window cofinality)**.** *Any finite chain of positive-unary exponent-window widenings of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet remains on the same cofinal dyadic comparison route consumed by* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*. The terminal packet of the chain is reached only through displayed scale transports, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *data.*

The cofinal route imports no host rational order, quotient equality, completed $\mathsf{Real}^{\uparrow}$ *endpoint, or ambient completeness row.*

Proof. First apply scale-window monotonicity, Definition 291.41, to each displayed widening in the finite chain. Each step preserves the dyadic comparison row consumed by $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$.

Next use Definition 291.27. It composes the finite positive-unary denominator refinements into one displayed scale-transport route from the initial packet to the terminal packet.

Now apply Definition 291.28. The terminal comparison row remains the same public DyadicRatCore envelope read by $\mathsf{RegSeqRat}^{\uparrow}$, with the same finite source surface available to $\mathsf{StreamName}^{\uparrow}$.

Therefore the whole finite widening chain is cofinal in the displayed dyadic comparison route, and the exclusion clauses from the cited theorems remove host rational order, quotient equality, completed Real endpoints, and ambient completeness. □

Theorem 291.43 (DyadicRatCore common-exponent comparison closure)**.** *Any finite positive-unary exponent-window widening chain of an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet preserves the common-exponent comparison row consumed by* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*. At every stage the comparison row is carried only by the displayed scale transport, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows of Definition 291.40.*

The closure statement exports no host rational order, quotient equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, hidden denominator selector, or ambient completeness row.*

Proof. Proceed by finite induction on the displayed positive-unary exponent-window widening chain. The empty chain leaves the initial packet unchanged, so the common-exponent comparison row is the original displayed row of the accepted packet.

For the step, factor the chain as a shorter widening chain followed by one displayed widening. The induction hypothesis preserves the common-exponent comparison row at the intermediate packet, with only the displayed transport, hsame, Cont, Pkg, and local NameCert support rows in scope.

Apply scale-window monotonicity, Definition 291.41, to the last widening. It transports the same comparison row across the wider exponent window and imports no ambient order or quotient equality.

Finally use scale-window cofinality, Definition 291.42, to identify the terminal packet as lying on the same cofinal dyadic comparison route consumed by $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. The exclusion clauses from monotonicity and cofinality remove host rational order, quotient equality, selected Real endpoints, hidden denominator selectors, and ambient completeness. □

Theorem 291.44 (DyadicRatCore finite widening-chain induction)**.** *Let I be a BEDC-visible invariant on accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packets. Suppose I holds for an initial accepted packet, and suppose one displayed positive-unary exponent-window widening preserves I whenever the widening is witnessed only by the scale transport, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows of Definition 291.40. Then I holds for the terminal packet of any finite positive-unary widening chain starting at the initial packet.*

The induction surface imports no host rational order, quotient equality, hidden denominator selector, selected $\mathsf{Real}^{\uparrow}$ *endpoint, or ambient completeness row.*

Proof. Proceed by finite induction on the displayed widening chain. The empty chain has the initial packet as its terminal packet, so the base assumption gives I directly.

For the step, write the chain as a shorter displayed widening chain followed by one positive-unary exponent-window widening. The induction hypothesis gives I on the intermediate accepted packet. Apply scale-window monotonicity, Definition 291.41, to the last displayed widening; its output supplies exactly the transported scale row, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows required by the one-step preservation hypothesis.

The one-step preservation hypothesis therefore transfers I from the intermediate packet to the terminal packet. Repeating this finite argument reaches the terminal packet of the original chain, and the exclusion clause is inherited from the accepted packet rows and from scale-window monotonicity. □

Theorem 291.45 (DyadicRatCore successor scale-window nesting)**.** *Let P be an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, and let P^{+} be obtained from P by one displayed positive-unary successor widening of the exponent window. Then P nests inside P^{+} as the same finite dyadic source surface: the signed mantissa row, denominator ledger, common-exponent comparison row, distance and radius rows, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support of P are carried into the corresponding rows of P^{+}.*

This one-successor nesting imports no host rational order, quotient equality, hidden denominator selector, selected $\mathsf{Real}^{\uparrow}$ *endpoint, or ambient completeness row.*

Proof. Apply scale-window monotonicity, Definition 291.41, to the displayed successor widening. It gives the transported dyadic comparison row and identifies the allowed support as displayed

scale transport, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

View the successor widening as the one-step finite chain. The finite widening-chain induction theorem, Definition 291.44, then specializes its preservation surface to the invariant saying that the original signed mantissa, denominator ledger, comparison row, distance and radius rows, and displayed support rows remain visible inside the terminal packet.

Finally use RegSeqRat bridge handoff exactness, Definition 291.28. The widened packet is still the public DyadicRatCore envelope consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and exposed to $\mathsf{StreamName}^{\uparrow}$, so the nesting is a finite packet inclusion rather than a passage through host rational order, quotient equality, hidden denominator selectors, selected $\mathsf{Real}^{\uparrow}$ endpoints, or ambient completeness. □

Theorem 291.46 (DyadicRatCore scale-window terminal fusion)**.** *If two finite chains of positive-unary exponent-window widenings of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet have a common terminal exponent, then they fuse into the same terminal DyadicRatCore comparison packet for* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers. The fused packet is reached only through the displayed scale transports, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows of Definition 291.40.*

No host rational order, quotient equality, completed $\mathsf{Real}^{\uparrow}$ *endpoint, hidden denominator selector, or ambient completeness row is imported by either chain.*

Proof. Apply scale-window cofinality, Definition 291.42, to both finite widening chains. Each chain reaches its terminal comparison row through the same displayed scale transports, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local NameCert rows.

Use denominator-refinement composition, Definition 291.27. Since the terminal exponent is common, the two composed refinement routes identify the same terminal positive-unary denominator window rather than two ambient host-order comparisons.

Now apply RegSeqRat bridge handoff exactness, Definition 291.28. The terminal DyadicRatCore comparison packet handed to $\mathsf{RegSeqRat}^{\uparrow}$ is exactly the public envelope row, and the same finite source packet is available to $\mathsf{StreamName}^{\uparrow}$.

The non-escape exclusions in the packet and the cited bridge theorems remove host rational order, quotient equality, completed Real endpoints, hidden denominator selectors, and ambient completeness. Therefore the two chains fuse at one terminal comparison packet for both consumers. □

Theorem 291.47 (DyadicRatCore scale-window consumer face determinacy)**.** *Let an accepted* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer read expose its dyadic face through a* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet and a finite chain of positive-unary exponent-window widenings. Then the consumer-facing dyadic face is determined by the terminal scale-window packet: the signed mantissa row, exponent window, denominator ledger, common-exponent comparison, distance and radius window, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row.*

A second widening chain, $\mathsf{Real}^{\uparrow}$ *seal read, or* $\mathsf{RegSeqRat}^{\uparrow}$ *finite request cannot introduce another dyadic face with different displayed rows over the same terminal packet. The consumer boundary therefore reads one dyadic source surface through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, without quotient stream equality, selected limits, host rational order, hidden denominator selectors, or ambient completeness.*

Proof. Apply scale-window cofinality, Definition 291.42, to the finite positive-unary widening chain used by the consumer read. The read reaches its terminal packet only through displayed scale trans-

ports, componentwise hsame rows, Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ row.

For two displayed widening chains, apply terminal fusion, Definition 291.46. A common terminal packet fuses the signed mantissa row, exponent window, denominator ledger, common-exponent comparison, distance and radius window, and the displayed support rows into one terminal DyadicRatCore comparison packet.

For the $\mathsf{RegSeqRat}^{\uparrow}$ face, use Definition 291.28. The finite request consumes exactly the public DyadicRatCore envelope delivered by that terminal packet, so it has no independent dyadic row from which to read a different face.

For the $\mathsf{Real}^{\uparrow}$ face, apply the dependency route Definition 38.738. The Real seal-visible read factors through the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumer faces, so it inherits the same terminal DyadicRatCore packet rather than adding a separate dyadic source.

The exclusion clauses in the scale-window packet, terminal fusion, bridge handoff, and dependency route remove quotient stream equality, selected limits, host rational order, hidden denominator selectors, and ambient completeness. Hence every accepted consumer read over the terminal scale-window packet exposes the same dyadic face. □

Theorem 291.48 (DyadicRatCore tail-radius monotone readback)**.** *Let an accepted* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer read its dyadic tolerance face through a* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet and a positive-unary tail-radius widening chain. Then the radius row read by the consumer is monotone along the displayed widening route: every later window contains the same signed mantissa, exponent ledger, common-exponent comparison, distance row, radius row, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support as the corresponding earlier window, transported through the displayed scale route.*

Thus the $\mathsf{StreamName}^{\uparrow}$ *finite-window read, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal all see one dyadic radius ledger on the cofinal terminal packet. The readback refuses host rational order, quotient equality, hidden denominator selectors, selected Real endpoints, and ambient completeness.*

Proof. Apply scale-window monotonicity, Definition 291.41, to each displayed positive-unary tail-radius widening. This preserves the dyadic comparison and radius rows through the permitted scale transport, hsame, Cont, Pkg, and local NameCert support.

Use scale-window cofinality, Definition 291.42, to place the finite widening chain on one terminal dyadic comparison route. The terminal route is reached without adding a host order or quotient coordinate.

Next apply terminal fusion, Definition 291.46. Any two displayed subchains with the same terminal radius window fuse into the same terminal packet, so no hidden denominator selector can choose a different ledger.

Finally use consumer face determinacy, Definition 291.47. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers all read their dyadic face from that terminal packet, and the cited exclusions remove host rational order, quotient equality, selected Real endpoints, and ambient completeness. □

Theorem 291.49 (DyadicRatCore RegSeqRat radius-window normal form)**.** *Let an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback expose its dyadic face through a finite positive-unary widening chain of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Then, on BEDC-visible rows, that readback is equivalent to the normal form consisting of the terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet together with the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the local* $\mathsf{Real}^{\uparrow}$ *seal face, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

The normal form contains no quotient rational, quotient stream equality, selected limit, countable-choice selector, metric-completeness row, host rational order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First apply finite widening-chain induction, Definition 291.44. Use as invariant the assertion that the accepted readback is presented by the visible signed mantissa, exponent window, denominator ledger, common-exponent comparison, distance and radius rows, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows of the current scale-window packet.

Next apply terminal fusion, Definition 291.46. Any two finite routes to the same terminal positive-unary exponent therefore present one terminal $\mathsf{DyadicRatCore}^{\uparrow}$ comparison packet, so the readback cannot retain a route-dependent dyadic face.

Now use consumer face determinacy, Definition 291.47. The terminal packet determines the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the local $\mathsf{Real}^{\uparrow}$ seal face from the same displayed dyadic rows.

Finally apply the RegSeqRat terminal radius-window boundary, Definition 291.37. It forces the accepted terminal readback through the finite radius window attached to the public DyadicRatCore envelope. Combining this boundary with the previous steps gives exactly the stated terminal packet plus the three consumer faces and shared support rows.

The exclusion clauses in the induction surface, terminal fusion, consumer determinacy, and terminal radius-window boundary remove quotient rationals, quotient stream equality, selected limits, countable-choice selectors, metric-completeness rows, host rational order, and ambient Real completeness coordinates. Thus the accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback has precisely the displayed BEDC-visible normal form. □

Theorem 291.50 (DyadicRatCore directed radius-refinement L10 request support)**.** *Let an accepted* $\mathsf{StreamName}^{\uparrow}$ *finite-window request, an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback request, or an accepted* $\mathsf{Real}^{\uparrow}$ *seal request present its dyadic face through a directed finite family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-unary radius-window refinements. Then every such request has one common terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window support packet. That packet consists only of the signed mantissa row, exponent window, denominator ledger, common-exponent comparison, distance and radius rows, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows.*

The support packet excludes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness rows, host rational order, and ambient $\mathsf{Real}^{\uparrow}$ *completeness. Thus the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite request routes cannot split into separate dyadic radius ledgers once their faces are presented by the same directed finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement family.*

Proof. Apply scale-window cofinality, Definition 291.42, to each branch of the directed finite refinement family. Every branch reaches its visible terminal dyadic comparison route through displayed scale transports, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

For two branches with the displayed common terminal window, use terminal fusion, Definition 291.46. Directedness supplies the finite comparison point, and fusion identifies the terminal signed mantissa row, exponent window, denominator ledger, common-exponent comparison, distance and radius rows, and displayed support rows as one $\mathsf{DyadicRatCore}^{\uparrow}$ scale-window packet.

Next apply consumer face determinacy, Definition 291.47. The accepted $\mathsf{StreamName}^{\uparrow}$ finite-window request, $\mathsf{RegSeqRat}^{\uparrow}$ readback request, and $\mathsf{Real}^{\uparrow}$ seal request read the same terminal dyadic face from that packet, so none of the three consumers carries an independent dyadic support route.

Finally use the RegSeqRat radius-window normal form, Definition 291.49. It repacks the $\mathsf{RegSeqRat}^{\uparrow}$ readback and local $\mathsf{Real}^{\uparrow}$ seal face into the terminal $\mathsf{DyadicRatCore}^{\uparrow}$ packet together with the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. The exclusion clauses from the cited results remove quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness rows, host rational order, and ambient $\mathsf{Real}^{\uparrow}$ completeness, leaving exactly the common terminal support packet stated above. □

Theorem 291.51 (DyadicRatCore terminal scale-window realizer)**.** *Let D be an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-exponent classifier window whose visible rows are the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, the positive-unary exponent row, the denominator ledger for the displayed power of two, the scaled-mantissa* $\mathsf{Rat}^{\uparrow}$ *comparison row, the componentwise* hsame *transport row, the displayed* Cont *route, and the* Pkg *provenance row. Then D determines a terminal finite scale window*

$$T_D = (m, e, \ell, r, H, C, P, N)$$

whose mantissa m, exponent e, denominator ledger ℓ, $\mathsf{Rat}^{\uparrow}$ *comparison row r, transport H, continuation route C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row are all read from the common-exponent classifier window.*

The realizer is terminal among finite dyadic scale-window reads with the same classifier data: any $\mathsf{StreamName}^{\uparrow}$ *finite-window consumer,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity consumer, or downstream handoff that preserves the same mantissa, exponent, denominator ledger,* $\mathsf{Rat}^{\uparrow}$ *comparison,* Cont *route, and* Pkg *provenance factors through T_D. The realizer exports no completed* $\mathsf{Real}^{\uparrow}$ *endpoint, selected limit, quotient rational equality, quotient stream equality, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the common-exponent classifier window. Its visible rows already contain the signed $\mathsf{Rat}^{\uparrow}$ mantissa, the positive-unary exponent, the denominator ledger, and the scaled-mantissa $\mathsf{Rat}^{\uparrow}$ comparison row.

Apply the scale-window obligation surface of Definition 291.13. That theorem identifies the finite dyadic tolerance row consumed by $\mathsf{StreamName}^{\uparrow}$ finite windows and $\mathsf{RegSeqRat}^{\uparrow}$ regularity checks, using only the displayed mantissa, exponent, denominator, comparison, transport, replay, and provenance data.

Next apply common-scale terminal refinement, Definition 291.95. It refines the displayed classifier window to a terminal finite scale packet while preserving the same signed mantissa row, denominator ledger, endpoint comparison rows, hsame transport, Cont replay, Pkg provenance, and local DyadicRatCore naming rows.

For terminality, use the uniqueness clause of Definition 291.96. Any competing finite scale-window read with the same visible classifier data and the same downstream $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ faces must determine the same terminal tolerance row.

Finally use the non-escape clauses of Definition 291.37. The terminal handoff is read before any $\mathsf{Real}^{\uparrow}$ seal can consume it, so the constructed packet has no completed real endpoint or hidden completion coordinate. □

Definition 291.52 (DyadicRatCore signed-exponent carrier)**.** *The signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier refines the carrier of Definition 291.1 to the finite packet*

$$Q = (m, e, \ell, W, R, H, C, P, N)$$

where m is the signed $\mathsf{Rat}^{\uparrow}$ *mantissa row, e is the positive unary exponent row, ℓ is the denominator ledger for 2^e, W is the common-exponent comparison window of Definition 291.3, R is the dyadic*

tolerance-radius row, H records componentwise hsame *transport, C records displayed* Cont *replay, P records the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers, and N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row. The packet has no quotient rational, host equality row, completed* $\mathsf{Real}^{\uparrow}$ *endpoint, or ambient completeness coordinate.*

Theorem 291.53 (DyadicRatCore common-exponent stability)**.** *Let Q and Q' be signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carriers whose mantissa, exponent, denominator ledger, tolerance-radius row, transport, replay, provenance, and local name are related by displayed* hsame *and* Cont *rows. Any common-exponent comparison window for Q factors through the same finite rows as the transported common-exponent comparison window for Q', namely the signed mantissa rows, positive unary exponent rows, transported denominator ledgers, scaled* $\mathsf{Rat}^{\uparrow}$ *mantissas,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows.*

Proof. Start from the signed-exponent carrier of Definition 291.52. Its comparison row W is already the common-exponent classifier window of Definition 291.3.

Apply denominator ledger transport, Definition 291.7, to the exponent and denominator rows. This transports the positive unary exponent and the ledger for 2^e without introducing a quotient rational or host normal form.

Use common-exponent window exactness, Definition 291.8. The transported classifier is determined by the transported exponent, ledgers, scaled mantissas, $\mathsf{Rat}^{\uparrow}$ classifier row, Cont replay, and Pkg provenance. The displayed hsame rows therefore carry the same common-exponent comparison surface from Q to Q'. □

Theorem 291.54 (DyadicRatCore tolerance-radius monotonicity)**.** *For a signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier Q, every displayed positive-unary refinement of the exponent row induces a monotone refinement of the tolerance-radius row R. The refined radius window remains a finite BEDC row over the denominator ledger, common-exponent comparison window, distance window,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *data, and it exports no host epsilon equality or completed* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. Read the radius row R through the signed-exponent carrier. It is attached to the same positive unary exponent and denominator ledger that define the common-exponent comparison window.

Apply the monotone-radius obligation of Definition 291.12. A positive-unary refinement widens the denominator scale and transports the dyadic radius through displayed ledger rows.

Then apply the distance-window stability of Definition 291.10. The distance and absolute-value observations remain inside the transported signed mantissa, common exponent, denominator ledger, hsame, Cont, and Pkg rows. Hence the tolerance-radius refinement is monotone on the displayed finite packet and cannot read host epsilon equality or a completed real endpoint. □

Theorem 291.55 (DyadicRatCore signed-exponent denominator-chain determinacy)**.** *Let $Q = (m, e, \ell, W, R, H, C, P, N)$ be a signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier. For any finite chain $Q = Q_0, Q_1, \ldots, Q_n$ in which each adjacent step is a displayed positive-unary refinement of the exponent row and transports the signed mantissa, denominator ledger, common-exponent comparison window, tolerance-radius row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row, the terminal denominator ledger, common-exponent comparison row, tolerance-radius row, and L10 dyadic face are determined by Q alone. The same terminal surface is obtained for every subdivision and parenthesization of the finite refinement chain. This determinacy is a statement only about the finite BEDC rows of the signed-exponent carrier; it excludes quotient rational*

equality, host normalization, selected $\mathsf{Real}^{\uparrow}$ *endpoints, countable choice, metric completeness, and ambient Real completeness.*

Proof. Proceed by induction on the finite refinement chain. In the empty chain the terminal rows are exactly ℓ, W, R, and the displayed L10 dyadic face of the carrier in Definition 291.52.

For a successor step, apply Definition 291.53 to the adjacent carriers. This identifies the transported common-exponent comparison row with the same finite mantissa, exponent, denominator-ledger, transport, replay, provenance, and local naming rows.

Apply Definition 291.54 to the same adjacent refinement. The radius row is transported monotonically along the displayed positive-unary exponent refinement and remains attached to the denominator ledger and common-exponent window already fixed by the induction hypothesis.

Associativity of finite chain concatenation is therefore harmless at the paper level: subdividing a step or reparenthesizing consecutive steps only repeats the same adjacent comparison and radius transports. The terminal denominator ledger, comparison row, and radius row remain the rows forced by the original carrier.

Finally use Definition 291.59 and Definition 291.58. They identify the downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ L10 dyadic face with exactly the displayed signed mantissa, positive unary exponent, denominator ledger, comparison, radius, transport, replay, provenance, and local NameCert rows, leaving no quotient rational, host normalization, selected endpoint, countable choice, metric-completeness, or ambient-completeness coordinate. □

Theorem 291.56 (DyadicRatCore signed-exponent denominator clearance)**.** *Let* $Q = (m, e, \ell, W, R, H, C, P, N)$ *be an accepted signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet. Then the positive unary exponent row* e *and denominator ledger* ℓ *are determined before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer may read a tolerance radius from* R*. Every accepted consumer route must therefore expose the denominator ledger as a finite BEDC row upstream of the L10 tolerance read.*

No host rational normal form, quotient equality, completed endpoint, selected limit, metric-completeness datum, or ambient completeness coordinate can replace the denominator ledger.

Proof. Start with Definition 291.55. It fixes the terminal denominator ledger, common-exponent comparison row, and tolerance-radius row from the accepted signed-exponent carrier and from its displayed finite refinement chain.

Next apply Definition 291.58. The NameCert surface lists the exponent, denominator ledger, comparison window, radius row, transport, replay, provenance, and local naming as the exact rows available to downstream consumers.

Finally use Definition 291.59. A StreamName, RegSeqRat, or RealUp source read determines precisely the displayed signed mantissa, positive unary exponent, denominator ledger, comparison, radius, transport, replay, provenance, and local NameCert rows. Hence the tolerance radius is read only after denominator clearance, with no quotient or host-normal-form shortcut. □

Theorem 291.57 (DyadicRatCore refinement-chain induction)**.** *Let* $Q = (m, e, \ell, W, R, H, C, P, N)$ *be a signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier, and let* $\mathcal{R}$ *be the class of finite displayed chains that start at* Q *and whose adjacent steps are positive-unary refinements of the exponent row transporting the signed mantissa* m*, exponent row* e*, denominator ledger* ℓ*, common-exponent window* W*, tolerance-radius row* R*, componentwise* hsame *row* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row* N*. If a predicate* Φ *on such finite chains holds for the empty chain at* Q*, and if* $\Phi(\Gamma)$ *implies* $\Phi(\Gamma')$ *whenever* Γ' *is obtained from* Γ *by appending one displayed positive-unary refinement step whose transported surface is accepted by the same* hsame*,*

Cont, Pkg, *and local naming rows and whose downstream face is the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *L10 dyadic face of the carrier, then* $\Phi(\Gamma)$ *holds for every finite chain* $\Gamma \in \mathcal{R}$.

Proof. Induct on the finite displayed chain. The empty chain is the base clause in the hypothesis, and its visible surface is precisely the carrier rows $m, e, \ell, W, R, H, C, P, N$ of Definition 291.52.

For the successor step, write the chain as a shorter displayed chain followed by one positive-unary exponent refinement. By the induction hypothesis Φ holds for the shorter chain.

The appended step is stable by Definition 291.53: the transported common-exponent window still reads only the signed mantissa, positive unary exponent, denominator ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Apply Definition 291.54 to the same step. The tolerance-radius row is monotone along the displayed refinement and remains attached to the denominator and comparison rows already carried by the shorter chain. The successor premise for Φ therefore applies, including the downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ L10 face.

This proves the induction principle for every finite displayed refinement chain without adding quotient rational equality, host normalization, selected endpoints, countable choice, metric completeness, or ambient Real completeness. □

Theorem 291.58 (DyadicRatCore signed-exponent NameCert obligation surface)**.** *For the signed-exponent carrier of Definition 291.52, the local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *obligation surface consists exactly of signed mantissa admission through* m*, positive unary exponent admission through* e*, denominator-ledger exposure through* ℓ*, common-exponent comparison through* W*, tolerance-radius monotonicity through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. This surface is the finite obligation layer consumed by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *when they read dyadic source data.*

Proof. Project the carrier $Q = (m, e, \ell, W, R, H, C, P, N)$. The mantissa, exponent, and denominator ledger rows are the signed dyadic source data of Definition 291.1, while W is the common-exponent classifier window and R is the tolerance-radius row.

Use Definition 291.53 for the comparison part of the surface. It keeps common-exponent reads stable under the displayed transport and replay rows.

Use Definition 291.54 for the radius part. It keeps all tolerance refinements on positive unary exponent data and finite dyadic distance rows.

Finally compare with the public bridge and handoff clauses, Definition 291.24, Definition 291.28, and Definition 291.20. The downstream StreamName, RegSeqRat, and Real consumers receive exactly the displayed signed mantissa, exponent, denominator, comparison, radius, transport, replay, provenance, and local NameCert rows, with no quotient rational, host equality, completed endpoint, or ambient completeness coordinate. □

Theorem 291.59 (DyadicRatCore signed-exponent source determinacy)**.** *Let* $Q = (m, e, \ell, W, R, H, C, P, N)$ *be a signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier as in Definition 291.52. Any accepted downstream dyadic source read for* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *determines exactly the displayed mantissa row* m*, positive unary exponent row* e*, denominator ledger* ℓ*, common-exponent window* W*, tolerance-radius row* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row* N*. Conversely, any finite packet exposing precisely those nine rows repacks as the same signed-exponent dyadic source surface for those consumers.*

Proof. Project the signed-exponent carrier of Definition 291.52. The projection names the only rows available to a downstream source read: m, e, ℓ, W, R, H, C, P, and N.

The exponent and denominator coordinates are fixed by Definition 291.7. The comparison and radius coordinates are fixed by Definition 291.53 and Definition 291.54. Together these clauses leave no additional source coordinate through which a consumer could distinguish two accepted reads.

The naming surface is then fixed by Definition 291.58. It identifies the local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ obligations with the same mantissa, positive unary exponent, denominator ledger, comparison window, radius, transport, replay, provenance, and local naming rows.

Apply the public bridge and handoff clauses, Definition 291.24 and Definition 291.28. They say that the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads consume exactly the public dyadic source rows already displayed, so an accepted downstream read determines exactly those rows.

For the converse, start with any finite packet exposing precisely m, e, ℓ, W, R, H, C, P, and N. The carrier projection repacks these rows as the signed-exponent surface, and the NameCert obligation theorem supplies the same public source boundary for the three consumers.

The correspondence exports no quotient rational, host equality row, completed $\mathsf{Real}^{\uparrow}$ endpoint, ambient completeness principle, or selected host limit. □

Theorem 291.60 (DyadicRatCore signed-exponent no-shortcut boundary)**.** *Let $Q = (m, e, \ell, W, R, H, C, P, N)$ be a signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier. Any accepted L10 consumer route for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *that exposes the signed-exponent* $\mathsf{DyadicRatCore}^{\uparrow}$ *face must pass through the displayed rows m, e, ℓ, W, R, H, C, P, and N. In particular, such a route cannot replace the positive unary exponent row or denominator ledger by host rational normalization, quotient rational equality, quotient stream equality, selected limits, countable choice, ambient completeness, metric completeness, or a direct* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Project the carrier of Definition 291.52. The projection exposes only the signed mantissa row, positive unary exponent row, denominator ledger, common-exponent window, tolerance-radius row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ row.

Apply Definition 291.58 and Definition 291.59. These identify the accepted source surface with exactly the nine displayed rows, so no consumer can keep the signed-exponent face while substituting a host rational normal form, quotient equality, selected limit, completeness principle, or terminal seal coordinate for either e or ℓ.

The L10 boundary is then routed through Definition 291.80, Definition 126.442, and Definition 301.69. These clauses carry the same dyadic rows through the current $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff, so every accepted L10 consumer route reaches the dyadic face through the displayed carrier rows rather than through an ambient shortcut. □

Theorem 291.61 (DyadicRatCore finite-tail window factorization)**.** *Every finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 window read whose* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is obtained from a displayed DyadicRatCore packet by a finite positive-unary exponent widening factors through the scale-window packet of Definition 291.40. The factorization first reads the widened* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, then the fused* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet supplied by Definition 291.46, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section supplied by Definition 291.28, and finally the* $\mathsf{Real}^{\uparrow}$ *L10 finite-window boundary of Definition 38.464.*

The visible route is exactly the four-face finite-tail packet

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

with the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. It exports no host rational order, quotient equality, selected limit, ambient completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the displayed finite positive-unary widening on the DyadicRatCore face. By scale-window cofinality, Definition 291.42, the widened observation remains on the same cofinal dyadic comparison route carried by the scale-window packet.

Next apply terminal fusion, Definition 291.46. It identifies the terminal widened comparison packet as one fused $\mathsf{StreamName}^{\uparrow}$ finite-window source, using only displayed scale transports, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

Now use RegSeqRat bridge handoff exactness, Definition 291.28. The fused terminal packet is read by $\mathsf{RegSeqRat}^{\uparrow}$ as the same public DyadicRatCore envelope, so the finite request section does not add an ambient order coordinate or a quotient-equality coordinate.

Finally pass to the Real L10 finite-window determinacy surface, Definition 38.464. That theorem consumes exactly the fused $\mathsf{StreamName}^{\uparrow}$ packet, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, and the local $\mathsf{Real}^{\uparrow}$ seal over the displayed support rows.

The exclusion clauses of the cited DyadicRatCore, RegSeqRat, and Real L10 theorems are stable under this composition. Hence the finite-tail read has the stated four-face factorization and cannot read host rational order, quotient equality, selected limits, ambient completeness, or Real completeness. □

Theorem 291.62 (DyadicRatCore finite-tail window confluence)**.** *Let two finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 window reads have* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces obtained from displayed DyadicRatCore packets by finite positive-unary exponent widenings. Assume the two widened DyadicRatCore faces fuse to the same terminal scale-window packet of Definition 291.40. Then the two four-face routes*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

determine the same terminal finite pullback through the DyadicRatCore, StreamName, RegSeqRat, and Real boundary.

The equality is only equality over the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. It asserts no quotient stream equality, no host rational equality, no selected limit, no ambient completeness, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First use terminal fusion, Definition 291.46. The two finite positive-unary widenings have the same terminal scale-window packet, so their DyadicRatCore comparison faces expose one common terminal dyadic packet with the same displayed scale transports, hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows.

Apply the finite-tail factorization theorem, Definition 291.61, to each read. Both reads therefore pass through the same four displayed faces: the common widened $\mathsf{DyadicRatCore}^{\uparrow}$ packet, the fused $\mathsf{StreamName}^{\uparrow}$ finite-window source, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, and the $\mathsf{Real}^{\uparrow}$ L10 finite-window boundary.

Next use the StreamName terminal pullback, Definition 126.504. Since the StreamName, DyadicRatCore, RegSeqRat, and Real faces and the displayed support rows agree after terminal fusion, the two accepted routes factor through one terminal finite BEDC pullback.

Finally apply the classifier pullback, Definition 126.514, together with the Real L10 finite-window witness determinacy theorem, Definition 38.464. The public classifier and the Real L10 boundary are determined by exactly the common terminal StreamName packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The exclusion clauses in the cited DyadicRatCore, StreamName, RegSeqRat, and Real L10 boundary theorems are stable under this common terminal pullback. Thus confluence holds only on the displayed BEDC rows and cannot be read as quotient stream equality, selected limits, ambient completeness, metric completeness, host rational equality, or Real completeness. □

Theorem 291.63 (DyadicRatCore finite-tail window terminality)**.** *Every* $\mathsf{Real}^{\uparrow}$*-and-completion consumer whose visible rows consist of a finite positive-unary widened* $\mathsf{DyadicRatCore}^{\uparrow}$ *face, the fused* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *finite-window boundary, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows factors uniquely through the terminal four-face packet of Definition 291.61. The forced packet has route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

If two such consumers agree componentwise on those visible rows, then they have the same public L10 read. The statement exports no quotient stream equality, no host rational equality, no selected limit, no ambient completeness row, no metric-completeness datum, no standard bridge datum, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply the finite-tail factorization theorem, Definition 291.61, to the consumer read. The visible DyadicRatCore widening, fused StreamName finite window, RegSeqRat finite-request section, local Real boundary, and displayed support rows place the consumer on exactly the four faces named in the theorem.

For uniqueness, compare two consumers with componentwise agreement on those rows. The finite-tail confluence theorem, Definition 291.62, identifies their presentations once both have been routed through the same terminal scale-window packet.

Now invoke the StreamName terminal pullback, Definition 126.504. It says that a downstream Real-and-completion read exposing the four faces and the displayed BHist, hsame, Cont, Pkg, and NameCert rows has a single terminal finite BEDC pullback through which the read must pass.

Finally use Real L10 finite-window witness determinacy, Definition 38.464. The terminal pullback determines the public L10 boundary from the fused StreamName packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, and the local Real seal, so componentwise agreement gives the same public L10 read and leaves no additional quotient, selected-limit, completeness, bridge, metric-completeness, host-equality, or Real-completeness coordinate. □

Theorem 291.64 (DyadicRatCore finite-tail phase-entry determinacy)**.** *Let two current Phase Real or* $\mathsf{Completion}^{\uparrow}$ *reads have first visible face a finite positive-unary widening of a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Suppose their remaining visible faces are the fused* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *finite-window boundary, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then the two reads determine the same entry* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet exactly when their terminal four-face packets agree componentwise over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

This is only entry determinacy for the finite BEDC phase boundary. It supplies no host rational order, quotient stream equality, selected limit, countable choice, ambient completeness, metric completeness, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. First apply scale-window cofinality, Definition 291.42, and terminal fusion, Definition 291.46. They force every displayed finite positive-unary widening of the first visible face onto one terminal

DyadicRatCore comparison packet, using only displayed scale transports, componentwise hsame rows, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^\uparrow}$ data.

Next route each Phase Real or $\mathsf{Completion}^\uparrow$ read through finite-tail factorization, Definition 291.61. The entry packet is therefore read only through the ordered four faces $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$, together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

For the forward implication, if the entry $\mathsf{DyadicRatCore}^\uparrow$ packets are the same, terminal fusion gives the same fused StreamName finite window. Finite-tail confluence, Definition 291.62, and finite-tail terminality, Definition 291.63, then identify the $\mathsf{RegSeqRat}^\uparrow$ finite-request section, the local $\mathsf{Real}^\uparrow$ finite-window boundary, and the resulting terminal four-face packet.

For the reverse implication, assume the terminal four-face packets agree componentwise. Finite-tail terminality says that every accepted Phase Real or $\mathsf{Completion}^\uparrow$ read exposing those four faces has a single terminal finite BEDC pullback, while confluence identifies any two presentations routed to that terminal packet. The first face of that terminal packet is exactly the entry $\mathsf{DyadicRatCore}^\uparrow$ scale-window packet, so the entry packets agree.

Finally use Real L10 finite-window witness determinacy, Definition 38.464. The Real boundary can read only the fused StreamName packet, DyadicRatCore endpoint and radius ledger, RegSeqRat finite-request section, local Real seal, and displayed support rows. Thus no off-table coordinate, including host rational order, quotient stream equality, selected limits, countable choice, ambient completeness, metric completeness, or $\mathsf{Real}^\uparrow$ completeness, can distinguish two entries after the terminal four-face packet agrees. □

Theorem 291.65 (DyadicRatCore finite-tail consumer-order square)**.** *For any current Phase Real and* $\mathsf{Completion}^\uparrow$ *finite-tail read whose first visible face is a finite positive-unary widening of a displayed* $\mathsf{DyadicRatCore}^\uparrow$ *packet, the construction-source projection*

$$\mathsf{DyadicRatCore}^\uparrow \to \mathsf{StreamName}^\uparrow \to \mathsf{RegSeqRat}^\uparrow \to \mathsf{Real}^\uparrow$$

and the consumer-readback projection

$$\mathsf{StreamName}^\uparrow \to \mathsf{DyadicRatCore}^\uparrow \to \mathsf{RegSeqRat}^\uparrow \to \mathsf{Real}^\uparrow$$

are two edge readings of the same displayed four-face certificate.

The square commutes only as equality of the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream equality, selected limit, host rational equality, host real equality, countable choice, metric-completeness datum, standard bridge row, aggregate formalstatus row, or* $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. First apply finite-tail window factorization, Definition 291.61. It gives the construction-source reading of the widened DyadicRatCore packet through the ordered $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$ faces over the displayed support rows.

Next use finite-tail phase-entry determinacy, Definition 291.64. The first visible DyadicRatCore entry is determined exactly by agreement of the terminal four-face packet, so changing from the construction-source edge to the consumer-facing edge cannot introduce a second entry record.

Now read the same packet from the StreamName side. The open-phase four-face exit readback, Definition 126.450, and the Real-completion terminal pullback, Definition 126.504, say that every current consumer read is consumed through the same StreamName, DyadicRatCore, RegSeqRat, and Real faces with the same visible BHist, hsame, Cont, Pkg, and NameCert rows.

Finally compare with the Real-side construction-consumer square, Definition 38.799. That theorem reads the same two orderings from the $\mathsf{Real}^\uparrow$ boundary, and its exclusion clauses leave only the

displayed BEDC rows as the commuting data. Hence the DyadicRatCore construction-source projection and the StreamName-facing consumer-readback projection are the two boundary readings of one four-face certificate, with no quotient, selected-limit, host-equality, choice, completeness, bridge, aggregate-status, or $\mathsf{Real}^{\uparrow}$ completeness coordinate added. □

Theorem 291.66 (DyadicRatCore finite-tail scale-window pullback exactness)**.** *Let two finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 reads have* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces obtained from displayed DyadicRatCore packets by finite positive-unary exponent widenings. Suppose the two reads share the same terminal scale-window packet of Definition 291.40 and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section. Then their* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces determine the same finite pullback route over* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

The exactness is only the finite BEDC pullback exactness of the displayed route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It exports no quotient stream equality, selected limit, host rational equality, ambient completeness row, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First apply scale-window terminal fusion, Definition 291.46. The common terminal scale-window packet identifies the widened DyadicRatCore comparison face through displayed scale transports, componentwise hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

Next apply finite-tail window factorization, Definition 291.61. Each finite-tail $\mathsf{Real}^{\uparrow}$ L10 read factors through the same ordered four-face packet: the terminal $\mathsf{DyadicRatCore}^{\uparrow}$ face, the fused $\mathsf{StreamName}^{\uparrow}$ finite-window packet, the displayed $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, and the $\mathsf{Real}^{\uparrow}$ L10 finite-window boundary.

Use finite-tail window confluence, Definition 291.62, on the two factorizations. The shared terminal scale-window packet already fixes the DyadicRatCore and StreamName faces, while the hypothesis fixes the RegSeqRat finite-request section.

Finally invoke the StreamName and Real completion pullback rows, Definition 126.504, Definition 126.514, and Definition 38.464. These rows identify the Real-facing boundary exactly from the common four faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

The cited exclusion clauses remove quotient stream equality, selected limits, host rational equality, ambient completeness, and Real completeness from each step. Therefore the two reads determine the same finite pullback route and nothing outside the displayed BEDC rows. □

Theorem 291.67 (DyadicRatCore finite-tail common scale-window reduction)**.** *Let two finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 reads have* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces obtained from displayed DyadicRatCore packets by finite positive-unary exponent widenings. Suppose both widened faces are read through the same displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet and share a terminal scale-window packet of Definition 291.40. Then the two reads determine the same visible* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section.*

Consequently the hypotheses of Definition 291.66 reduce to the common displayed scale-window route together with the common $\mathsf{StreamName}^{\uparrow}$ *finite-window packet. The reduction exports no quotient stream equality, selected limit, host rational equality, ambient completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First apply scale-window terminal fusion, Definition 291.46. The common terminal packet identifies the widened $\mathsf{DyadicRatCore}^{\uparrow}$ comparison face for both reads using only displayed scale transports, componentwise hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

Next use finite-tail window factorization, Definition 291.61, on each read. Since both reads pass through the same $\mathsf{StreamName}^{\uparrow}$ finite-window packet, the factorization exposes one downstream finite request surface for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff rather than two independent request sections.

Now apply RegSeqRat bridge handoff exactness, Definition 291.28. The handoff consumes exactly the public DyadicRatCore envelope visible through the shared StreamName packet, so the displayed signed mantissa row, positive unary exponent row, denominator ledger, common-exponent classifier, arithmetic window, distance window, radius window, BHist carrier data, hsame transport, Cont readback, and Pkg provenance are the same on both sides.

Therefore the visible $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section is common. Feeding this derived section match, together with the already common terminal scale-window packet, into Definition 291.66 gives the same finite pullback route over BHist, hsame, Cont, Pkg, and the local $\mathsf{NameCert}$ rows.

The exclusion clauses of terminal fusion, factorization, and bridge handoff are preserved by this reduction. Hence the argument does not assert quotient stream equality, selected limits, host rational equality, ambient completeness, or any $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 291.68 (DyadicRatCore finite-tail cofinal section)**.** *Let a finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 read have a* $\mathsf{DyadicRatCore}^{\uparrow}$ *face lying on a finite positive-unary cofinal scale-window route. Then the visible cofinal route determines one* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section and one* $\mathsf{Real}^{\uparrow}$ *seal-visible source section through*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The determined sections use only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows of the route.*

No quotient stream equality, selected limit, host rational equality, ambient completion row, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ *completeness principle is part of the section data.*

Proof. Start from the finite positive-unary cofinal scale-window route. By scale-window cofinality, Definition 291.42, the visible $\mathsf{DyadicRatCore}^{\uparrow}$ face reaches the terminal scale-window packet through displayed scale transports, componentwise hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ data.

Next apply finite-tail window factorization, Definition 291.61. The finite-tail $\mathsf{Real}^{\uparrow}$ read is therefore routed through the fused $\mathsf{StreamName}^{\uparrow}$ finite-window packet, then through the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, and finally through the $\mathsf{Real}^{\uparrow}$ L10 finite-window boundary.

Use the common scale-window reduction, Definition 291.67, to identify the $\mathsf{RegSeqRat}^{\uparrow}$ section as the one forced by the visible cofinal route. The reduction removes any choice of a second request section once the displayed scale-window route and the displayed $\mathsf{StreamName}^{\uparrow}$ finite-window packet are fixed.

Finally apply StreamName finite-request section budget exactness, Definition 126.556, and StreamName finite-request section pullback, Definition 126.577. These rows identify the Real-facing section as the same displayed packet over the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

The cited nonescape clauses exclude quotient stream equality, selected limits, host rational equality, ambient completion, metric-completeness data, and RealUp completeness from every step. Thus the cofinal finite-tail route determines exactly the finite-request and seal-visible source sections described above. □

Theorem 291.69 (DyadicRatCore RegSeqRat Real pullback composition)**.** *Let a finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 read have a* $\mathsf{DyadicRatCore}^{\uparrow}$ *face lying on a finite positive-unary scale-window route. Suppose the route displays its terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the induced* $\mathsf{StreamName}^{\uparrow}$

finite-window packet, the visible $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, and the* $\mathsf{Real}^{\uparrow}$ *seal-visible source section determined by the finite L10 boundary. Then these four faces compose as one BEDC pullback route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over the displayed BHist, hsame, Cont, Pkg, *and local* NameCert *rows.*

The composed route is exactly the finite pullback handoff of the displayed packets. It does not identify quotient streams, select a limit, assert host rational equality, add ambient completion, import metric completeness, or prove $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite positive-unary scale-window route. By Definition 291.68, the visible route determines a single $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section and a single $\mathsf{Real}^{\uparrow}$ seal-visible source section through the four displayed faces.

Next apply the common scale-window reduction, Definition 291.67. It turns the common terminal $\mathsf{DyadicRatCore}^{\uparrow}$ packet and the induced $\mathsf{StreamName}^{\uparrow}$ finite-window packet into the same visible $\mathsf{RegSeqRat}^{\uparrow}$ request section, so the middle face is not a second choice point.

Use StreamName finite-request section budget exactness, Definition 126.556, to bind the StreamName packet to the same finite request budget. The request budget is therefore the one consumed by the RegSeqRat section already forced by the scale-window route.

Now invoke finite-tail scale-window pullback exactness, Definition 291.66. Its hypotheses are exactly the terminal scale-window packet, the induced StreamName packet, and the visible RegSeqRat section built above, so it packages the DyadicRatCore, StreamName, RegSeqRat, and Real faces as one finite pullback route over BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally apply Real L10 finite-window witness determinacy, Definition 38.464. This fixes the seal-visible Real source section reached by the composed route and prevents a second Real-facing witness from entering the handoff.

The exclusion clauses in the cited results remove quotient stream equality, selected limits, host rational equality, ambient completion, metric completeness, and $\mathsf{Real}^{\uparrow}$ completeness at each step. The conclusion is therefore only the composed finite BEDC pullback route. □

Theorem 291.70 (DyadicRatCore finite-tail endpoint invariance)**.** *Let two finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 reads have* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces on finite positive-unary scale-window routes. Suppose the two presentations display the same terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the same induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, and the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source section. Suppose also that their endpoint-facing rows share the same* BHist, hsame, Cont, Pkg, *and local* NameCert *support over these four faces. Then every endpoint-facing* $\mathsf{Real}^{\uparrow}$ *consumer reads the same finite BEDC endpoint route from the two presentations.*

The invariant endpoint is only the finite route exposed by the four L10 faces. It is not quotient stream equality, a selected limit, host rational equality, ambient completion, metric completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply finite-tail scale-window pullback exactness, Definition 291.66. The shared terminal $\mathsf{DyadicRatCore}^{\uparrow}$ packet and shared $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section identify the same finite pullback route over the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Next apply the common scale-window reduction, Definition 291.67. Since the two presentations also share the induced $\mathsf{StreamName}^{\uparrow}$ finite-window packet, the RegSeqRat request section is forced by the visible scale-window route rather than supplied as an independent endpoint choice.

Now use the RegSeqRat–Real pullback composition, Definition 291.69. It composes the shared DyadicRatCore packet, StreamName finite-window packet, RegSeqRat finite-request section, and Real seal-visible source section as one BEDC pullback route through the endpoint-facing rows.

Finally invoke Real L10 finite-window witness determinacy, Definition 38.464. Any $\mathsf{Real}^{\uparrow}$ consumer that only sees the finite L10 endpoint boundary must therefore read the same seal-visible endpoint route from either presentation.

The cited nonescape clauses exclude quotient stream equality, selected limits, host rational equality, ambient completion, metric completeness, and $\mathsf{Real}^{\uparrow}$ completeness. Hence the conclusion is endpoint invariance of the displayed finite BEDC route, not equality in any ambient completion. □

Theorem 291.71 (DyadicRatCore finite-request Real-seal uniqueness)**.** *Let two finite-tail* $\mathsf{Real}^{\uparrow}$ *L10 reads have* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces on finite positive-unary scale-window routes. Suppose the two presentations share the same terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet, the same induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window packet, the same visible* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, and the same seal-visible* $\mathsf{Real}^{\uparrow}$ *source section. Suppose also that the displayed consumer rows expose the same* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support over these four faces. Then every Real-seal consumer reads one unique finite request route through*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over the displayed rows.

The unique route is only the finite Real-seal request route visible to the consumer. It does not identify quotient streams, choose a selected limit, assert host rational equality, add ambient completion, import metric completeness, or prove $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the finite-tail common scale-window reduction, Definition 291.67. The shared terminal $\mathsf{DyadicRatCore}^{\uparrow}$ packet and shared induced $\mathsf{StreamName}^{\uparrow}$ packet force one visible $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, so the middle request surface cannot split into two consumer routes.

Next use the RegSeqRat–Real pullback composition, Definition 291.69. It composes the terminal DyadicRatCore packet, StreamName finite-window packet, RegSeqRat finite-request section, and seal-visible Real source section as one finite BEDC pullback route over the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally apply finite-tail endpoint invariance, Definition 291.70. Since the two presentations expose the same endpoint-facing support over the same four faces, any Real-seal consumer that reads only the displayed finite boundary must read the same finite request route from both presentations.

The nonescape clauses in the three cited results exclude quotient stream equality, selected limits, host rational equality, ambient completion, metric completeness, and $\mathsf{Real}^{\uparrow}$ completeness. Therefore the uniqueness is exactly Real-seal finite-request uniqueness over the displayed BEDC rows. □

Theorem 291.72 (DyadicRatCore Real-completion obstruction ledger)**.** *For a current Phase Real and Completion exit read whose dyadic face is displayed by a* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet, failure is localized exactly at one of the following finite sites:*

1. *the dyadic source row or its componentwise transport is absent;*
2. *the induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window face is absent or does not read the same finite window;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section is absent or leaves the displayed request surface;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal is absent or does not seal the displayed finite request;*
5. *one of the support rows* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *is absent or is not transported with the four faces.*

Equivalently, once the dyadic source and transport row, the $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the listed support rows are all present over the same displayed packet, the exit read has no further obstruction.*

The ledger is only a finite obstruction account for the displayed BEDC route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness principle, metric-completeness row, or bridge row beyond the displayed dyadic-facing packet.

Proof. For the forward direction, apply Definition 291.39. It localizes any failed public dyadic handoff to the dyadic source inventory, the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ read-back face, the local $\mathsf{Real}^{\uparrow}$ seal, or the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next apply Definition 126.523. This keeps the StreamName-facing Real-completion read on the same four-object exit surface and rules out moving the failure into an undisplayed completion object or choice of a limit.

Use Definition 301.77 to align the four faces. The StreamName schedule rows, DyadicRatCore endpoint and radius rows, RegSeqRat classifier and regularity rows, and local Real seal row are necessary and sufficient for the same finite BHist route.

Finally use Definition 38.464. Once the local $\mathsf{Real}^{\uparrow}$ seal-visible finite window is fixed by those faces, no second Real-facing witness remains from which a distinct failure site could be read.

For the reverse direction, suppose one of the five listed finite sites is missing or not transported componentwise. The corresponding local theorem blocks the route at exactly that face: the DyadicRatCore handoff theorem blocks source and support defects, the StreamName localization theorem blocks finite-window defects, the RegSeqRat boundary theorem blocks finite-request defects, and the Real L10 determinacy theorem blocks local seal defects.

Therefore absence of any listed site is an obstruction, while presence of all five sites composes the same finite exit route by Definition 291.71. The exclusion clauses of the four cited results remove quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, and extra bridge rows from the ledger. □

Theorem 291.73 (DyadicRatCore finite-tail obstruction minimality)**.** *For a current Phase Real and Completion exit read whose dyadic face is displayed by a* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet, each finite site in Definition 291.72 is independently necessary. The dyadic source and componentwise transport row, the induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support cannot be supplied by any combination of the other four sites.*

When all five sites are present over the same displayed packet, the obstruction ledger and Definition 291.71 leave no further finite obstruction for the displayed route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

This minimality claim is only about the finite BEDC exit route. It does not identify quotient streams, select a limit, assert host rational or host real equality, add an ambient completion row, import a metric-completeness row, or add a bridge row.

Proof. The obstruction ledger, Definition 291.72, gives the five-site classification. It says that every failed exit read is localized at exactly the dyadic source and transport row, the StreamName

finite-window face, the RegSeqRat finite-request section, the local Real seal, or the shared BHist, hsame, Cont, Pkg, and NameCert support.

Independence follows by testing one missing site at a time. Without the dyadic source and componentwise transport row, the DyadicRatCore handoff cannot present the signed mantissa, positive unary exponent, denominator ledger, and common-exponent classifier consumed downstream, so the remaining four sites have no source inventory to transport.

Without the $\mathsf{StreamName}^{\uparrow}$ finite-window face, finite-tail common scale-window reduction, Definition 291.67, has no displayed finite window through which the dyadic packet can expose the RegSeqRat request surface. Supplying the other four sites would have to replace that face by quotient stream equality or a selected limit, both excluded by the route theorems.

Without the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, RegSeqRat–Real pullback composition, Definition 291.69, has no middle request row to compose with the StreamName packet and the Real seal. Replacing it would require host rational or host real equality outside the displayed BEDC rows.

Without the local $\mathsf{Real}^{\uparrow}$ seal, Real L10 finite-window witness determinacy cannot bind the endpoint-facing witness. The other four sites therefore do not create a Real-facing seal by themselves, and any substitute would amount to an ambient completion or metric-completeness row excluded from the finite route.

Without the shared BHist, hsame, Cont, Pkg, and NameCert support, the four visible faces are not transported over one packet. Replacing those rows by the faces alone would require an extra bridge row rather than the displayed local support.

Conversely, when all five sites are present over the same displayed packet, Definition 291.72 removes every finite obstruction listed in the classification, and Definition 291.71 gives one finite request route through the displayed faces. Hence none of the five sites is replaceable by the other four, and together they are sufficient for the finite exit route claimed here. □

Theorem 291.74 (DyadicRatCore finite-tail obstruction classifier determinacy)**.** *Let two current Phase Real and Completion exit reads have first visible face a finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Suppose the two reads expose the same terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet, the same induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the same local* $\mathsf{Real}^{\uparrow}$ *seal-facing section, and the same transported* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows over those four faces. Then the two obstruction classifiers agree exactly when the two reads fail at the same one of the five finite sites listed in Definition 291.72 over that shared support surface.*

The classifier comparison is only the finite BEDC comparison attached to

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It excludes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, apply the obstruction ledger, Definition 291.72, to each failed read. Each read is classified by exactly one displayed finite site: the dyadic source and transport row, the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, the local $\mathsf{Real}^{\uparrow}$ seal, or the transported support rows.

Now compare the two ledger classifications over the shared four-face support. If the classifiers agree while the sites differed, one of the five sites would be replaceable by the other four together with a different failed site. This contradicts finite-tail obstruction minimality, Definition 291.73, which makes each listed site independently necessary.

For the reverse direction, assume the two reads fail at the same listed finite site over the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. Replay the local blocking theorem for that site on both reads: the DyadicRatCore handoff block for source and transport defects, the StreamName localization block for finite-window defects, the RegSeqRat boundary block for finite-request defects, the Real L10 determinacy block for local seal defects, and the support-transport block for shared row defects.

The same local blocking theorem reads the same failed coordinate in both presentations, so the obstruction classifiers agree. The nonescape clauses of the ledger and minimality theorem keep the result finite: it is a BEDC classifier determinacy statement, not a quotient-equality, selected-limit, choice, bridge, or completeness theorem. □

Theorem 291.75 (DyadicRatCore finite-tail exit-certificate equivalence)**.** *Let a current Phase Real and Completion exit read have first visible face a finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Then the read satisfies the objectwise exit certificate exactly when the five finite sites of Definition 291.72 are present over the same four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and the four faces $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$ *each supply their local scoped-closure and audit-clean threshold rows. The support surface is exactly the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows transported with those four faces.*

This equivalence exports no quotient streams, selected limits, countable-choice selector, metric-completeness data, aggregate formalstatus row, aggregate bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness theorem. The local* $\mathsf{Real}^{\uparrow}$ *face remains a face of the finite packet, not an ambient completeness object.*

Proof. For necessity, start with an accepted objectwise exit certificate. Apply the Real L10 objectwise face-status exit theorem, Definition 38.792, to read the certificate as four local face records over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$, together with the finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Now use the DyadicRatCore obstruction ledger, Definition 291.72. Since the read has first visible face a finite-tail $\mathsf{DyadicRatCore}^{\uparrow}$ scale-window packet, any missing dyadic source row, StreamName finite-window face, RegSeqRat finite-request section, local Real seal, or transported support row would be exactly one of the ledger obstructions. Therefore an accepted objectwise certificate contains all five finite sites over the same four-face packet.

For sufficiency, assume the five ledger sites are present over the same displayed packet and that each of the four faces supplies its local scoped-closure and audit-clean threshold rows. The ledger removes every finite obstruction, while finite-request Real-seal uniqueness, Definition 291.71, gives a single route through

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Apply the Real L10 current-phase exit local threshold theorem, Definition 38.796. It identifies precisely those four local threshold records, read with the displayed support rows, as the current Phase Real and Completion objectwise exit read.

Minimality comes from Definition 291.73. Removing any one of the five finite sites leaves a ledger obstruction that the other four sites cannot replace, so the equivalence cannot be shortened by substituting a different face or support coordinate.

Finally, the Real L10 threshold theorem keeps the closure and verification axes facewise, and the DyadicRatCore nonescape clauses keep the route finite. Hence the accepted roadmap exit read

is exactly the stated four-face certificate, with no quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 291.76 (DyadicRatCore finite-tail exit-certificate classifier reflection)**.** *Let a current Phase Real and Completion exit read have dyadic face a finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Then satisfying the objectwise exit certificate of Definition 291.75 is equivalent to its obstruction classifier reading the non-failing class over the five finite sites of Definition 291.72 on the same four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The classifier reflection uses only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows transported with those four faces.*

It exports no quotient stream, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness theorem. The non-failing classifier class is therefore a finite BEDC readback of the objectwise exit certificate, not an ambient completion principle.*

Proof. Use Definition 291.75 to identify the objectwise exit certificate with the simultaneous presence of the five finite sites over the displayed four-face route. This gives the sufficiency and necessity sides of the certificate without adding any quotient stream, selected limit, choice row, metric-completeness datum, bridge row, or $\mathsf{Real}^{\uparrow}$ completeness theorem.

Next apply obstruction classifier determinacy, Definition 291.74. It says that two reads have the same obstruction classifier exactly when they fail at the same one of the five displayed sites over the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

For a read satisfying the exit certificate, the equivalence theorem leaves no failed ledger site. Hence the determinacy theorem reads its classifier as the single non-failing class attached to that complete finite ledger.

Conversely, suppose the obstruction classifier reads the non-failing class over the five sites on the displayed route. By determinacy, there is no failed dyadic source and transport row, no failed $\mathsf{StreamName}^{\uparrow}$ finite-window face, no failed $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, no failed local $\mathsf{Real}^{\uparrow}$ seal, and no failed transported support row.

Finally use finite-tail obstruction minimality, Definition 291.73. It excludes replacing any one site by the other four, so the non-failing classifier class cannot be produced by a hidden substitute outside the displayed ledger. The five-site finite ledger is therefore exactly the objectwise exit certificate. □

Theorem 291.77 (DyadicRatCore finite-tail exit-certificate refinement invariance)**.** *Let* $\mathcal{E}$ *and* $\mathcal{E}'$ *be current Phase Real and Completion exit reads whose first visible face is a finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packet. Suppose* $\rho : \mathcal{E} \to \mathcal{E}'$ *is a certified finite refinement whose visible action preserves the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$*, and preserves the displayed support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Then* $\mathcal{E}$ *and* $\mathcal{E}'$ *determine the same objectwise exit certificate of Definition 291.75.*

Conversely, if the objectwise exit certificates determined by $\mathcal{E}$ *and* $\mathcal{E}'$ *are equal, then on the displayed four-face section this equality recovers only a certified finite refinement preserving the four displayed faces and the displayed support rows above. It recovers no quotient stream equality, selected limit, host rational equality, host real equality, countable-choice selector, ambient completion*

row, metric-completeness row, aggregate formalstatus row, aggregate bridge row, standard bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. For invariance under a certified finite refinement, read both exit reads through Definition 291.75. The certificate is exactly the finite-tail $\mathsf{DyadicRatCore}^{\uparrow}$ source and transport row, the induced $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section, the local $\mathsf{Real}^{\uparrow}$ seal, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support over one four-face packet.

Apply the StreamName terminal pullback, Definition 126.504. Since ρ preserves the four displayed faces and the support rows, the two reads factor through the same terminal finite pullback rather than through two independently chosen completion surfaces.

Now apply Real L10 refinement functoriality, Definition 38.805. It transports the current Phase Real and Completion certificate across exactly such a support-row-preserving finite refinement. The local threshold theorem, Definition 38.796, identifies the transported certificate as the objectwise product of the same four local threshold records and the same displayed support rows.

For the converse, assume the two objectwise exit certificates are equal. By Definition 291.75, equality of certificates identifies precisely the five finite sites listed in the DyadicRatCore obstruction ledger: the dyadic source and transport row, the StreamName finite-window face, the RegSeqRat finite-request section, the local Real seal, and the displayed support rows.

Use obstruction minimality, Definition 291.73. None of those five sites can be replaced by the other four, so the equality cannot recover a hidden quotient stream, selected limit, host equality, choice row, completion row, bridge row, or Real-completeness coordinate. The only recoverable comparison is the finite support-row-preserving refinement on the displayed four-face section.

Finally, the Real local threshold and the StreamName terminal pullback keep the recovered comparison inside the same finite terminal packet. Therefore finite refinements preserving the displayed faces and support rows are exactly the presentation changes under which the DyadicRatCore finite-tail exit certificate remains invariant. □

Theorem 291.78 (DyadicRatCore finite-tail exit-certificate cofinal-meet stability)**.** *Let* $\mathcal{E}_1$ *and* $\mathcal{E}_2$ *be current Phase Real and Completion exit reads whose first visible faces are finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packets. Suppose each read is related by a certified finite refinement to a common cofinal finite scale-window meet* $\mathcal{M}$*, and that both refinement legs preserve the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$*, together with the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Then* $\mathcal{E}_1$*,* $\mathcal{E}_2$*, and their common meet* $\mathcal{M}$ *determine the same objectwise exit certificate of Definition 291.75.*

Conversely, equality of the objectwise exit certificates after passage to such a common cofinal meet reflects only the displayed finite rows: the four faces $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$*, and the transported* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support. It reflects no quotient stream equality, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. First apply the finite-tail cofinal section theorem, Definition 291.68, to the visible cofinal route into $\mathcal{M}$. The route forces one $\mathsf{RegSeqRat}^{\uparrow}$ finite-request section and one local $\mathsf{Real}^{\uparrow}$ seal-visible section through the displayed four-face packet.

Next apply finite-tail scale-window pullback exactness, Definition 291.66. The common meet supplies the terminal $\mathsf{DyadicRatCore}^{\uparrow}$ scale-window packet, the shared $\mathsf{StreamName}^{\uparrow}$ finite-window face, the forced $\mathsf{RegSeqRat}^{\uparrow}$ section, and the local $\mathsf{Real}^{\uparrow}$ boundary, all over the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Transport the certificate from $\mathcal{E}_1$ to $\mathcal{M}$ by Definition 291.77. Apply the same theorem to the refinement leg from $\mathcal{E}_2$ to $\mathcal{M}$. Both transports land on the objectwise certificate determined by the common four-face packet, so the two original reads determine the same certificate.

For the converse, inspect the equality after transport to $\mathcal{M}$ using Definition 291.75. The certificate consists exactly of the five finite sites in the obstruction ledger: the dyadic source and transport row, the StreamName finite-window face, the RegSeqRat finite-request section, the local Real seal, and the transported support rows.

Finally use obstruction minimality, Definition 291.73. None of those finite sites can be replaced by quotient streams, selected limits, choice data, metric completeness, aggregate status, bridge rows, or Real-completeness coordinates. Hence equality after the cofinal meet reflects only the displayed finite rows and no hidden completion coordinate. □

Theorem 291.79 (DyadicRatCore finite-tail refinement support necessity)**.** *Let $\mathcal{E}$ and $\mathcal{E}'$ be current Phase Real and Completion exit reads whose dyadic faces are finite-tail* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packets. If a certified finite refinement between them preserves the objectwise exit certificate, then it preserves exactly the dyadic source and transport row, the induced* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the visible* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request section, the local* $\mathsf{Real}^{\uparrow}$ *seal, and the transported* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows.*

Conversely, dropping any one of those rows prevents preservation of the objectwise exit certificate. The loss cannot be repaired by quotient stream equality, selected limits, host equality, choice data, metric completeness, aggregate status, bridge rows, standard bridge rows, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Expand the preserved certificate with Definition 291.75. It identifies the certificate with the five finite sites over the four-face route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus preservation of the objectwise exit certificate already reads the dyadic source and transport row, the induced StreamName finite-window face, the RegSeqRat finite-request section, the local Real seal, and the transported support rows as the whole certificate surface.

For the preservation direction, use finite-tail exit-certificate refinement invariance, Definition 291.77. A certified finite refinement that preserves the displayed faces and the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows transports the certificate and recovers no additional quotient, limit, host-equality, choice, metric, bridge, or completeness coordinate.

For common refinements, apply Definition 291.78. Passing both reads to a common cofinal finite scale-window meet preserves the same displayed rows and reflects equality of certificates back only along those rows.

It remains to show necessity of each support row. Use finite-tail obstruction minimality, Definition 291.73. If any one of the dyadic source and transport row, StreamName finite-window face, RegSeqRat finite-request section, local Real seal, or transported support rows is dropped, the missing site is an independent ledger obstruction and cannot be replaced by the other sites.

The nonescape clauses in the equivalence and minimality results exclude every ambient repair listed in the statement. Therefore certificate preservation for finite-tail $\mathsf{DyadicRatCore}^{\uparrow}$ exit reads is precisely facewise and support-row preserving, and no completion, quotient, choice, aggregate-status, or bridge coordinate can masquerade as finite refinement data. □

$\mathsf{DyadicRatCore}^{\uparrow}$ reads its signed mantissa row and denominator ledger through the integer and rational certificate surfaces of chapter 31 and chapter 37. It then supplies the finite dyadic tolerance

ledger consumed by chapter 126, chapter 301, and chapter 38, while exporting no host rational quotient, selected real limit, or $\mathsf{Real}^{\uparrow}$ completeness theorem.

Theorem 291.80 (DyadicRatCore L10 sibling dependency totality)**.** *Every accepted Phase Real or completion-facing consumer that reaches* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *through a dyadic tolerance face consumes exactly the signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, positive* UnaryHistory *exponent row, denominator ledger, common-exponent classifier, radius/distance windows, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*. Conversely, those rows supply the dyadic face required by the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal. The total route exports no quotient rational equality, quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. For the consumer-to-dyadic direction, apply Definition 291.24. It identifies the public dyadic consumer inventory with the signed mantissa, positive unary exponent, denominator ledger, common-exponent classifier, finite windows, displayed history and sameness rows, readback, package provenance, and certificate support already supplied by $\mathsf{DyadicRatCore}^{\uparrow}$.

The standard-source bridge of Definition 291.25 keeps that inventory fixed when the dyadic face is presented to sibling consumers. Then Definition 291.28 gives the RegSeqRat handoff exactness for the same endpoint, radius, and finite-window rows.

For the reverse direction, start with precisely those $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The StreamName terminal pullback of Definition 126.504 places the DyadicRatCore face, the finite StreamName schedule, the RegSeqRat regular source, and the local Real seal in one terminal finite packet.

Finally apply Definition 38.738. It reads the ordered route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and excludes quotient rational or stream equality, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, and $\mathsf{Real}^{\uparrow}$ completeness from the route. These exclusions are exactly the non-rows listed in the theorem statement. □

Theorem 291.81 (DyadicRatCore StreamName finite-window radius transport)**.** *Whenever a* $\mathsf{StreamName}^{\uparrow}$ *finite-window consumer reads a* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint/radius ledger and then hands the read to a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer or to a* $\mathsf{Real}^{\uparrow}$ *seal route, the transported finite window preserves the same signed* $\mathsf{Rat}^{\uparrow}$ *mantissa row, positive* UnaryHistory *exponent row, common-exponent classifier, radius/distance window, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The transport introduces no point carrier, quotient rational equality, quotient stream equality, selected limit, host real equality, countable-choice row, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the endpoint/radius ledger supplied by Definition 291.80. That theorem identifies the dyadic consumer surface with the signed mantissa, positive unary exponent, denominator, classifier, finite radius and distance windows, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Apply Definition 126.442. The StreamName finite-window read is anchored to this dyadic observation ledger before any downstream consumer can inspect it, so the projection keeps the same rows rather than replacing them by a detached stream or host-real witness.

For the RegSeqRat handoff, use Definition 291.28. It reads the same endpoint/radius packet as the regular-rational finite window, preserving the classifier and radius/distance rows already visible through the StreamName projection.

For the Real seal route, apply Definition 38.738. Its ordered route reads the DyadicRatCore face through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before the local $\mathsf{Real}^{\uparrow}$ seal, so the Real-facing packet inherits the same finite support rows.

The exclusion clauses in these three route theorems rule out every additional point carrier, quotient rational or stream equality, selected limit, host real equality, countable-choice row, metric-completeness row, and $\mathsf{Real}^{\uparrow}$ completeness coordinate. Hence the finite-window transport is exactly the displayed DyadicRatCore radius ledger throughout the StreamName, RegSeqRat, and Real consumers. □

Theorem 291.82 (DyadicRatCore common scale-window triangle ledger)**.** *Let three accepted L10 consumers among the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *routes expose dyadic faces that each factor through pairwise* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window packets. Then the three faces admit one common finite directed-meet scale window. That window carries the signed mantissa row, refined positive-unary exponent row, denominator ledger, common-exponent comparison row, distance and radius rows, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows for all three consumer faces.*

The triangle ledger exports no quotient rational, host epsilon equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice selector, metric-completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First place each pairwise dyadic face on a terminal scale-window packet by applying Definition 291.46. The pairwise packets therefore expose the same kind of signed mantissa, positive-unary exponent, denominator, comparison, transport, readback, provenance, and local NameCert rows.

Next apply Definition 291.34 to the finite family consisting of the three pairwise terminal packets. The directed meet supplies one common finite monotone radius window carrying the shared refined exponent, denominator ledger, common-exponent classifier, distance rows, carrier rows, hsame transports, Cont readbacks, Pkg provenance, and local NameCert row.

Use Definition 291.32 at each consumer boundary. It forces the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read to pass through the displayed radius handoff before any terminal seal can consume the dyadic packet, so the three L10 faces cannot split into separate ambient dyadic sources.

Finally apply Definition 291.80. It reads the same finite dyadic rows through the route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and identifies those rows as the dyadic source consumed by the L10 real-and-completion boundary.

The exclusion clauses of the terminal fusion, directed meet, directed-refinement handoff, and L10 route leave no row for quotient rationals, host epsilon equality, selected real endpoints, countable-choice selectors, metric-completeness data, or $\mathsf{Real}^{\uparrow}$ completeness. Hence the common window is exactly the finite triangle ledger stated above. □

Theorem 291.83 (DyadicRatCore four-face pullback correspondence)**.** *Let three accepted L10 consumers expose dyadic faces satisfying Definition 291.82. Suppose the same visible rows are also consumed by a terminal four-face route through*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the common scale-window triangle ledger and the terminal four-face pullback determine one another exactly on the signed mantissa, refined positive-unary exponent, denominator ledger, common-exponent comparison, distance and radius rows, StreamName finite window, RegSeqRat classifier and readback rows, Real seal row, componentwise hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

This correspondence exports no quotient rationals, quotient streams, host ε equality, selected $\mathsf{Real}^{\uparrow}$ *endpoints, countable choice, metric completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge data.*

Proof. Begin with the three accepted L10 dyadic faces. Apply Definition 291.82 to obtain the common directed-meet scale window over the three consumers. The result supplies the signed mantissa row, refined exponent row, denominator ledger, comparison row, distance and radius rows, transports, continuations, package provenance, and local DyadicRatCore naming surface.

Next place that same finite window on the downstream route. Use Definition 126.356 to read the DyadicRatCore window as the StreamName finite window, RegSeqRat classifier and readback surface, and Real seal boundary consumed by the ordered route. Then apply Definition 38.455 and Definition 38.458. These identify the terminal presentation with the displayed four faces and leave no hidden terminal coordinate beyond the listed rows.

For the reverse direction, start from the terminal four-face pullback. Read its DyadicRatCore face together with the displayed StreamName, RegSeqRat, and Real rows. The pullback uniqueness fixes the same BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ coordinates, so the dyadic face restricts to the common finite scale window supplied by the triangle ledger.

The cited pullback and terminal results add no quotient rational carrier, quotient stream equality, host ε equality, selected real endpoint, countable-choice selector, completeness datum, metric-completeness row, or standard bridge witness. Hence the two presentations determine exactly the same BEDC-visible finite surface. □

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreCarrier common scale window four face pullback correspondence`

Theorem 291.84 (DyadicRatCore terminal Real exit source uniqueness)**.** *Let two accepted Phase Real and Completion terminal exits expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, local* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *coordinates. Suppose both exits are accepted through the terminal four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then their $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window sources agree as the same common finite triangle ledger. In particular there is no independent dyadic source branch behind the shared terminal* $\mathsf{Real}^{\uparrow}$ *seal: the signed mantissa, refined positive-unary exponent, denominator ledger, common-exponent comparison, distance and radius rows, componentwise transports, displayed continuations, package provenance, and local DyadicRatCore naming rows are the same BEDC-visible source data for both exits.*

This uniqueness exports no quotient rational, quotient stream, host ε equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable choice, metric completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge row.*

Proof. Apply Definition 291.83 to each accepted exit. Each terminal presentation is therefore determined by a common finite triangle ledger together with the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use Definition 126.351. Since the two exits expose the same downstream $\mathsf{StreamName}^{\uparrow}$ finite-window surface and the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows, both routes pass through the same four ordered faces. The four-face source lattice leaves no extra source edge between the StreamName window and the Real seal.

Finally apply Definition 38.458 to the shared terminal boundary. The two accepted terminal pullbacks over the same four faces and support coordinates are identified, so any difference in signed mantissa, exponent, denominator, distance, or radius rows would create a second four-face packet over the same terminal seal.

That second packet is impossible by the shared terminal pullback. Hence both exits read the same $\mathsf{DyadicRatCore}^{\uparrow}$ scale-window source as the common finite triangle ledger, and the exclusion clause follows from the corresponding four-face and terminal uniqueness inventories. □

Theorem 291.85 (DyadicRatCore four-face window-refinement cocycle)**.** *Let three accepted finite positive-unary scale-window refinements expose pairwise* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces, and suppose each pairwise face maps through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the two ways of passing from the pairwise common scale windows to a terminal four-face pullback agree on the signed mantissa, refined positive-unary exponent, denominator ledger, common-exponent comparison, distance and radius rows, $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows.*

The cocycle exports no quotient rational or quotient stream equality, selected limit, choice row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 291.82 to the three accepted pairwise scale-window refinements. This supplies one common finite dyadic window carrying the signed mantissa, refined exponent, denominator ledger, comparison, distance, radius, transport, continuation, package, and local NameCert rows for the three faces.

Next apply Definition 291.83. The common dyadic window is identified with the terminal four-face pullback over the route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so any comparison between two refinement orders is reduced to a comparison of the same four displayed faces.

Use Definition 126.550 on the downstream triple overlap. It makes the $\mathsf{StreamName}^{\uparrow}$ finite window and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier/readback seal independent of the order in which the three finite overlaps are assembled.

Finally apply Definition 301.69. The handoff exactness carries the downstream cocycle back across the $\mathsf{DyadicRatCore}^{\uparrow}$ source and forward to the $\mathsf{Real}^{\uparrow}$ seal row, preserving the componentwise hsame, displayed Cont, Pkg, and local NameCert coordinates. Thus both refinement paths determine the same terminal four-face surface, and the exclusion clause is inherited from the triangle ledger, four-face correspondence, triple-overlap seal, and handoff exactness results. □

Theorem 291.86 (DyadicRatCore common-scale choice-free triangle)**.** *Let three terminal L10 dyadic faces expose pairwise* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale windows over the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The common finite scale window obtained for the three faces is determined by directed finite refinement of the displayed signed mantissa rows, refined positive-unary exponent rows, denominator ledgers, common-exponent comparison rows, distance and radius rows, componentwise hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support. It is not determined by choosing rational representatives, endpoints, host epsilon data, quotient streams, or selected RealUp limits.*

Thus every downstream StreamName, RegSeqRat, or Real terminal read of the triangle consumes the same common finite dyadic ledger.

Proof. First apply Definition 291.82. It constructs the common scale window as a directed finite meet of the three displayed dyadic faces, carrying exactly the mantissa, exponent, denominator, comparison, distance, radius, transport, replay, provenance, and local naming rows.

Next use the directed meet and refinement handoff invoked in that ledger. Each refinement step keeps the same finite row inventory and has no coordinate for a selected rational representative, endpoint selector, host epsilon witness, quotient stream, or selected Real limit.

Now apply Definition 291.83. The common dyadic window is the source read by the downstream StreamName, RegSeqRat, and Real faces, so any alternate choice of representatives would have to produce a second four-face pullback over the same visible rows.

Finally use Definition 291.85. The cocycle identifies the terminal four-face surface for all orders of finite refinement. Therefore the common-scale triangle is choice-free and is determined only by the directed finite refinement ledger displayed above. □

Theorem 291.87 (DyadicRatCore terminal Real exit has no fifth source)**.** *Let an accepted Phase Real and Completion terminal* $\mathsf{Real}^{\uparrow}$ *exit be accepted through the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the visible $\mathsf{DyadicRatCore}^{\uparrow}$ *scale window, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and classifier surface, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support exhaust the source inventory of the exit.*

No fifth source, aggregate formalstatus row, bridge row, quotient stream, selected limit, countable-choice row, metric-completeness row, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate can compensate for a missing one of those four faces or its displayed BEDC support rows.*

Proof. Apply Definition 291.83. It identifies the terminal route with exactly the four visible faces: the dyadic scale-window face, the StreamName finite-window face, the RegSeqRat readback and classifier face, and the local Real seal face, together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 291.84. Once the downstream StreamName, RegSeqRat, Real, and support rows are fixed, the dyadic source is forced to be the same common finite triangle ledger. Thus a different dyadic source would be a second terminal pullback over the same four-face surface.

Finally apply Definition 38.730 and Definition 38.729. They make the L10 exit facewise: each of the four route positions must be present with its own support and local status rows, and no aggregate token or off-table Real-completeness datum can supply an absent face.

The exclusion inventories in the DyadicRatCore four-face correspondence, the terminal Real exit source uniqueness theorem, and the Real L10 route theorems list exactly the rows outside the route: quotient streams, selected limits, countable choice, metric completeness, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Since none of those rows belongs to the four-face source surface, the displayed inventory is exhausted and there is no fifth compensating source. □

Theorem 291.88 (DyadicRatCore common scale regular-name interface)**.** *Let an accepted regular-name route expose a* $\mathsf{StreamName}^{\uparrow}$ *finite-window face and a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity face, and suppose both faces consume the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *common scale window along the L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the two regular-name consumers read the same signed mantissa row, positive-unary exponent row, denominator ledger, common-exponent comparison, distance and radius rows, componentwise hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The*

StreamName finite-window face and the RegSeqRat regularity face therefore form one BEDC-visible regular-name interface over the common DyadicRatCore scale window.

This interface exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable choice, metric-completeness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 291.82 to the accepted route faces. The common finite scale window supplies the signed mantissa, positive-unary exponent, denominator ledger, comparison, distance, radius, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows before either downstream consumer reads the dyadic packet.

Next apply Definition 291.83. It identifies that dyadic window with the ordered four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the StreamName finite-window face and the RegSeqRat classifier and readback face are projections of the same pullback surface rather than independent regular-name sources.

Use the StreamName and RegSeqRat handoff rows Definition 126.356, Definition 301.25, and Definition 301.69. These rows force the finite-window consumer and the regularity consumer to preserve the displayed DyadicRatCore rows across the regular-name boundary, including the componentwise transports, continuations, package provenance, and local naming coordinates.

Finally apply Definition 291.87. Since the terminal L10 exit has only the dyadic scale window, the StreamName finite window, the RegSeqRat regularity/readback surface, the Real seal, and the displayed support rows, none of the excluded objects can supply another regular-name coordinate. The two consumers therefore read exactly the same common-scale interface, with no quotient rational, quotient stream, selected endpoint, choice row, metric-completeness row, or Real-completeness coordinate. □

Theorem 291.89 (DyadicRatCore common scale comparison)**.** *Let two accepted common scale windows expose the same signed mantissa row, positive-unary exponent row, denominator ledger, distance and radius rows,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows. Suppose both windows are consumed by the regular-name and Real seal route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the two windows compare by the same finite directed-refinement ledger. Consequently every downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer reads the same tolerance order from the common scale surface.*

The comparison exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First place the two accepted windows in the common triangle ledger of Definition 291.82. The shared signed mantissa, exponent, denominator, distance, radius, transport, replay, provenance, and local naming rows determine the finite directed-refinement surface on which both windows are read.

Next apply Definition 291.83. It identifies the directed-refinement ledger with the terminal four-face route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the comparison cannot split into separate dyadic and regular-name sources.

Use Definition 291.84 at the terminal Real seal. Since the two windows expose the same downstream StreamName, RegSeqRat, Real, hsame, Cont, Pkg, and NameCert rows, terminal uniqueness forces the same DyadicRatCore source ledger behind both exits.

Finally apply Definition 291.88. The regular-name consumer reads the common dyadic scale window before the RegSeqRat readback and Real seal, so the directed-refinement order transported to RegSeqRat and RealUp is the same order for both windows. The exclusion clause is inherited from the triangle ledger, four-face pullback, terminal source uniqueness, and regular-name interface. □

Theorem 291.90 (DyadicRatCore common scale window composition coherence)**.** *Let $W_0 \leq W_1 \leq W_2$ be accepted nested* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale windows whose visible rows feed the ordered L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume each window displays the signed mantissa row, positive-unary exponent row, denominator ledger, common-exponent comparison row, distance and radius rows, StreamName finite window, RegSeqRat classifier and readback rows, Real seal row, and the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Then the direct radius handoff from W_0 to W_2 and the two-step handoff from W_0 through W_1 to W_2 determine the same finite four-face exit certificate over the displayed route.*

The common certificate exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 291.41 to the two accepted inclusions $W_0 \leq W_1$ and $W_1 \leq W_2$. The visible mantissa, positive-unary exponent, denominator, comparison, distance, radius, BHist, hsame, Cont, Pkg, and local NameCert rows are therefore retained along both legs.

Next apply Definition 291.46. It identifies the direct W_0 to W_2 scale window with the terminal scale window obtained by first refining to W_1 and then refining to W_2, so the two radius handoff surfaces have the same finite dyadic row inventory.

Use Definition 291.77. The finite-tail exit certificate transported along the direct refinement agrees with the certificate transported along the nested refinement, because both are read from the same terminal fused dyadic window and the same displayed support rows.

Finally apply Definition 291.83. The equality of dyadic exit certificates pushes through the StreamName finite-window face, the RegSeqRat classifier and readback face, and the Real seal face. The exclusions in Definition 291.87 rule out any fifth source that could distinguish the direct and two-step certificates, so the two radius handoffs determine one finite four-face exit certificate. □

Theorem 291.91 (DyadicRatCore common scale distance-window invariant)**.** *Let two accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint/radius ledgers be transported to a common exponent scale, and suppose their visible distance-window comparison feeds the ordered L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the displayed distance-window comparison is invariant under the common scale-window triangle ledger and composes through the terminal four-face pullback route. The invariant preserves the signed mantissa rows, refined positive-unary exponent rows, denominator ledgers, endpoint distance rows, radius rows, StreamName finite window, RegSeqRat classifier and readback rows, Real seal row, componentwise hsame*, displayed* Cont*,* Pkg*, and local* NameCert *coordinates.*

The invariant exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First place the two endpoint/radius ledgers inside a three-face comparison by adding the common-scale comparison face. Apply Definition 291.82. It supplies one finite directed-meet scale window carrying the mantissa, exponent, denominator, distance, radius, transport, replay, provenance, and local naming rows for the displayed comparison.

Next read that finite window through the terminal route. Apply Definition 291.83. The dyadic distance-window comparison is identified with the StreamName finite window, RegSeqRat classifier and readback rows, and Real seal row of the same four-face pullback.

Finally use Definition 291.90. Any refinement from one common exponent scale to another composes with the same direct radius handoff, so the distance-window comparison cannot depend on the chosen common-scale presentation.

The nonescape inventory is inherited from the triangle ledger, the four-face pullback correspondence, and the composition coherence theorem. None of those rows contains quotient rationals, quotient streams, selected endpoints, choice data, metric-completeness data, bridge rows, or ambient Real-completeness coordinates, so the stated invariant is exactly the displayed finite BEDC surface. □

Theorem 291.92 (DyadicRatCore common-scale order refinement)**.** *Let two accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison windows be lifted to a common exponent scale, and suppose their displayed comparison row is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *finite window along the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

If that finite window is refined to any finer common scale carrying the same signed mantissa rows, refined positive-unary exponent rows, denominator ledgers, distance-window row, radius row, componentwise hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support, then the displayed dyadic order row is stable on the finer window. The RegSeqRat consumer reads the transported comparison row; it does not recompute order by selecting a new rational representative, quotient stream, host real comparison, or terminal* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Proof. Begin with the common exponent presentation. Apply Definition 291.82 to place the two comparison windows and the RegSeqRat finite-window consumer on one directed-meet scale ledger.

Next use Definition 291.91. The displayed distance-window comparison is invariant under the common scale ledger and through the terminal four-face route, so refining the scale preserves the comparison data already visible to the RegSeqRat window.

Apply Definition 291.94 on the finer common scale. It recovers the same tolerance request from the refined radius and distance rows, which prevents the finer window from introducing a different comparison source.

Finally transport the order row by the componentwise hsame coordinates carried by the refinement. Since the mantissa, exponent, denominator, distance, radius, continuation, provenance, and local naming rows are all retained, the RegSeqRat finite window consumes the transported order row rather than a new selector or host comparison. The exclusion clause follows from the same finite row inventory. □

Theorem 291.93 (DyadicRatCore four-face scale-window chain induction)**.** *Let*

$$W_0 \leq W_1 \leq \cdots \leq W_n$$

be a finite nonempty nested chain of accepted $\mathsf{DyadicRatCore}^{\uparrow}$ *scale windows whose visible rows feed the ordered L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume each W_i displays the signed mantissa row, refined positive-unary exponent row, denominator ledger, common-exponent comparison row, distance and radius rows, StreamName finite window, RegSeqRat classifier and readback rows, Real seal row, and the visible BHist, hsame, Cont, Pkg, *and local* NameCert *support. Then the direct radius handoff from W_0 to W_n and every parenthesized finite composite of adjacent handoffs along the chain determine the same finite four-face exit certificate over the displayed route.*

The common certificate exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct on the finite chain length. In the terminal base case the chain has one window, and the direct certificate is the reflexive terminal fusion of that accepted $\mathsf{DyadicRatCore}^{\uparrow}$ scale window with its displayed support rows.

In the one-step case, apply Definition 291.41 to the adjacent inclusion. The visible mantissa, exponent, denominator, comparison, distance, radius, BHist, hsame, Cont, Pkg, and local NameCert rows are retained, so the unique adjacent handoff is already the direct handoff.

For the induction step, split the chain into the prefix

$$W_0 \leq W_1 \leq \cdots \leq W_{n-1}$$

and the last adjacent inclusion $W_{n-1} \leq W_n$. By the induction hypothesis, every parenthesized composite inside the prefix determines the same four-face certificate as the direct handoff from W_0 to W_{n-1}.

Now compose that prefix certificate with the final adjacent handoff. Apply Definition 291.90 to the three-window comparison $W_0 \leq W_{n-1} \leq W_n$. It identifies the direct W_0 to W_n handoff with the certificate obtained by first using the prefix handoff and then using the last adjacent inclusion.

Finally apply Definition 291.83 to push this equality through the StreamName finite-window face, the RegSeqRat classifier and readback face, and the Real seal face. The nonescape inventory of Definition 291.87 leaves no quotient rational, quotient stream equality, selected endpoint, countable-choice row, metric-completeness row, bridge row, or ambient Real-completeness row that could distinguish a different parenthesization. Thus every finite adjacent composite and the direct handoff determine one four-face exit certificate. □

Theorem 291.94 (DyadicRatCore common scale tolerance inversion)**.** *Let W_0 and W_1 be terminal four-face* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows over the ordered L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Assume the two windows share a common exponent scale after refinement and agree on the displayed dyadic radius row and distance-window row at that scale. Then the earlier tolerance request that generated the terminal comparison is recovered by the same common-scale ledger. The recovered request is read from the signed mantissa rows, positive-unary exponent rows, denominator ledgers, common-scale comparison row, radius row, distance row, StreamName finite window, RegSeqRat classifier and readback rows, Real seal row, and visible BHist, hsame, Cont, Pkg, *and* NameCert *support.*

The inversion exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, host real comparison, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First read the two terminal windows through Definition 291.91. The distance-window invariant places both presentations on the same common-scale ledger and preserves the displayed mantissa, exponent, denominator, radius, distance, transport, replay, provenance, and local naming rows.

Next use Definition 291.90. Since the radius rows agree after refinement, composing the earlier tolerance request into the terminal common scale gives the same radius handoff on both windows.

Apply Definition 291.93 to the finite chain from the earlier request to the terminal refined windows. The induction says that every parenthesized refinement path determines the same four-face exit certificate, so the common terminal radius and distance rows recover the unique earlier tolerance request along that chain.

Finally push the recovered request through Definition 291.83. The StreamName, RegSeqRat, and Real faces read the same finite dyadic ledger, while Definition 291.87 excludes quotient rationals, quotient streams, selected endpoints, host real comparisons, choice data, metric completeness, bridge rows, and ambient Real completeness as alternate recovery coordinates. □

Theorem 291.95 (DyadicRatCore common-scale terminal refinement)**.** *Let P and Q be two finite dyadic tolerance windows accepted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier. Suppose their signed mantissa rows are read from the same* $\mathsf{Int}^{\uparrow}$ *ledger, and suppose each window presents its denominator row through a finite positive-unary exponent history. Then P and Q admit one common* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale window T with the following BEDC-visible rows:*

1. *the endpoint comparisons of P and Q transport to T by componentwise* hsame *and displayed* Cont *replay;*
2. *the denominator route of T factors through the same unary exponent tower used by the two input histories, with no host rational order or hidden denominator selector;*
3. *the terminal tolerance row of T is exactly the row consumed by the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity face of chapter 301, and the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

The refinement exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct first on the finite positive-unary exponent history for P. The empty case leaves P at its displayed scale-window packet. The successor case applies Definition 291.41 to widen the endpoint comparison, denominator, distance, radius, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows by one displayed unary exponent step.

Induct in the same way on the finite positive-unary exponent history for Q. At each successor step the signed mantissa row remains the same $\mathsf{Int}^{\uparrow}$ ledger row by hypothesis, while the denominator row is transported only through the displayed unary tower. Thus both histories reach accepted terminal packets whose endpoint comparisons are still comparable by componentwise hsame over the common signed mantissa ledger.

Now apply Definition 291.82 to the two terminal dyadic packets together with the downstream L10 face. The common directed-meet window supplies one finite scale window carrying the shared signed mantissa row, refined positive-unary exponent row, denominator ledger, endpoint comparison rows, distance and radius rows, displayed continuations, package provenance, and local DyadicRatCore naming rows.

Finally use Definition 291.123. It identifies the terminal tolerance row of that common window with the four-face route

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

so the same row is the one consumed by the StreamName finite-window chapter, the RegSeqRat regularity chapter, and the Real seal chapter cited in the statement.

The exclusion clauses of scale-window monotonicity, the common scale-window triangle ledger, and terminal four-face tolerance exhaustion leave no quotient rational carrier, quotient stream equality, selected endpoint, countable-choice row, metric-completeness row, bridge row, or ambient Real-completeness coordinate. Hence the terminal refinement is a finite dyadic induction product rather than a prose route through external completion data. □

Theorem 291.96 (DyadicRatCore common-scale terminal refinement uniqueness)**.** *Let P and Q be finite dyadic tolerance windows accepted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier, and let T and T' be two common* $\mathsf{DyadicRatCore}^{\uparrow}$ *terminal refinements of those same windows. Suppose T and T' preserve the same signed mantissa row, the same positive-unary exponent histories, the same denominator ledger, the same endpoint comparison rows, the same distance and radius rows, the same* hsame *and displayed* Cont *rows, the same* Pkg *provenance, the same local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows, and the same downstream four-face rows through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Then T and T' determine the same terminal tolerance row and the same finite four-face exit certificate over the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The uniqueness statement uses no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 291.95 to the two presentations. Both T and T' are placed on common finite scale-window packets whose visible data are the signed mantissa ledger, positive-unary exponent histories, denominator ledger, endpoint comparisons, distance and radius rows, hsame, Cont, Pkg, and local DyadicRatCore naming rows.

Next apply Definition 291.83. The comparison of T and T' is therefore a comparison through the displayed four-face route, with the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity surface, the $\mathsf{Real}^{\uparrow}$ seal row, and the DyadicRatCore tolerance ledger all read from the same finite support.

Now use Definition 291.84. Since the downstream StreamName, RegSeqRat, Real, and support rows agree, there is only one dyadic scale-window source for the exit. Hence the two terminal refinements have the same terminal tolerance row.

Finally apply Definition 291.87. With the dyadic source and the three downstream faces fixed, no additional route coordinate can distinguish the exits. The common tolerance row therefore determines one finite four-face exit certificate, and the excluded objects cannot supply a second certificate. □

Theorem 291.97 (DyadicRatCore finite directed common-scale terminal refinement)**.** *Let $\mathcal{F}$ be a finite family of dyadic tolerance windows accepted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier. Suppose every member of $\mathcal{F}$ reads its signed mantissa row from the same displayed* $\mathsf{Int}^{\uparrow}$ *ledger and presents its denominator row through a finite positive-unary exponent history. Then repeated application of*

common-scale terminal refinement produces one accepted terminal $\mathsf{DyadicRatCore}^{\uparrow}$ *window* $T_{\mathcal{F}}$ *with the following properties:*

1. *every window in* $\mathcal{F}$ *maps to* $T_{\mathcal{F}}$ *by componentwise endpoint transport, denominator replay, distance and radius replay,* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows;*
2. *the terminal tolerance row of* $T_{\mathcal{F}}$ *is independent of the parenthesization used to combine the finite family;*
3. *the finite exit certificate determined by* $T_{\mathcal{F}}$ *is the same four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The construction exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct on the finite family $\mathcal{F}$. In the singleton case, the given accepted window is already the terminal packet for the one-member family after replaying its displayed positive-unary exponent history. The terminal four-face tolerance exhaustion theorem identifies its tolerance row with the route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

For the induction step, let S be the terminal window supplied for the preceding finite subfamily and let P be the next accepted tolerance window. Apply Definition 291.95 to S and P. The hypotheses are stable under the induction construction: the signed mantissa ledger remains the displayed $\mathsf{Int}^{\uparrow}$ ledger, and the denominator rows still factor through finite positive-unary exponent histories.

It remains to check that the chosen bracketing has not introduced extra data. Apply Definition 291.96 to two terminal windows obtained from two parenthesizations of the same finite family. Their visible signed mantissa, exponent, denominator, endpoint, distance, radius, hsame, Cont, Pkg, local DyadicRatCore naming, and downstream four-face rows agree by the induction hypothesis and by the pairwise refinement step. Hence the two bracketings determine the same terminal tolerance row.

Finally apply Definition 291.123. The terminal row is therefore the finite support consumed by the StreamName finite-window face, the RegSeqRat regularity face, and the Real seal face. The exclusion clauses are inherited from the pairwise refinement and uniqueness theorems, so no quotient carrier, quotient stream equality, selected endpoint, choice row, metric-completeness row, bridge row, or ambient Real-completeness coordinate enters the directed finite meet. □

Theorem 291.98 (DyadicRatCore common-scale terminal refinement permutation invariance)**.** *Let* $\mathcal{E}$ *and* $\mathcal{E}'$ *be two finite enumerations of the same finite family of dyadic tolerance windows accepted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *carrier. Suppose every enumerated window reads its signed mantissa row from the same displayed* $\mathsf{Int}^{\uparrow}$ *ledger and presents its denominator row through a finite positive-unary exponent history. Suppose also that* $\mathcal{E}$ *and* $\mathcal{E}'$ *preserve the same signed mantissa ledger, the same finite exponent histories, the same denominator rows, the same endpoint comparison rows, the same distance and radius rows, the same* hsame *and displayed* Cont *rows, the same* Pkg *provenance, and the same local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows. Then the repeated common-scale terminal refinements determined by* $\mathcal{E}$ *and by* $\mathcal{E}'$ *determine the same terminal tolerance row and the same finite four-face exit certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

This permutation-invariance statement exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. It is enough to compare adjacent swaps of finite enumerations. The usual finite enumeration argument decomposes any reordering of the same family into a finite chain of adjacent transpositions, and each intermediate list preserves the same signed mantissa ledger, exponent histories, denominator rows, endpoint comparisons, distance and radius rows, hsame, Cont, Pkg, and local DyadicRatCore naming rows.

For one adjacent swap, first apply Definition 291.97 to the finite family underlying the two neighboring presentations. Both orders produce accepted terminal DyadicRatCore windows for the same finite support, and the directed-system theorem supplies the common terminal refinement row for that support.

Next apply Definition 291.96. The two terminal windows have the same displayed signed mantissa, exponent, denominator, endpoint, distance, radius, hsame, Cont, Pkg, local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$, and downstream four-face rows, so the adjacent swap cannot change the terminal tolerance row.

Induct along the finite chain of adjacent swaps from $\mathcal{E}$ to $\mathcal{E}'$. The base case is the identical enumeration. The step composes the equality of terminal tolerance rows supplied by the adjacent-swap comparison with the induction hypothesis, so the initial and final enumerations determine the same terminal row.

Finally apply Definition 291.123. The shared terminal tolerance row is exactly the finite support consumed by the StreamName finite-window face, the RegSeqRat regularity face, and the Real seal face. Since the support rows are fixed throughout the adjacent-swap induction, no quotient rational carrier, quotient stream equality, selected endpoint, countable-choice row, metric-completeness row, bridge row, or ambient Real-completeness coordinate can distinguish a second four-face exit certificate. □

Theorem 291.99 (DyadicRatCore directed common-scale normal form)**.** *Let* $\mathcal{F}$ *be a finite family of accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance windows whose signed mantissa rows are read from one displayed* $\mathsf{Int}^{\uparrow}$ *ledger and whose denominator rows are presented by finite positive-unary exponent histories. For every iterated common-scale refinement route through* $\mathcal{F}$*, there is one consumer-readable normal form* $N_{\mathcal{F}}$ *with the following rows:*

1. *each member of* $\mathcal{F}$ *maps to* $N_{\mathcal{F}}$ *by endpoint transport, denominator replay, distance and radius replay,* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *rows;*
2. *every* $\mathsf{StreamName}^{\uparrow}$ *finite-window face,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity face, and* $\mathsf{Real}^{\uparrow}$ *seal read from* $\mathcal{F}$ *factors through* $N_{\mathcal{F}}$*;*
3. *two constructions of* $N_{\mathcal{F}}$ *from different parenthesizations or refinement routes agree on the terminal tolerance row and on the four-face exit certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The normal form exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness data.*

Proof. Apply Definition 291.97 to the finite family $\mathcal{F}$. It supplies an accepted terminal window $T_{\mathcal{F}}$ whose visible rows are exactly the transported endpoint comparisons, denominator histories,

distance and radius rows, hsame, displayed Cont, Pkg, local DyadicRatCore naming rows, and terminal four-face exit support. Define $N_{\mathcal{F}}$ to be this displayed terminal packet together with those rows.

Next compare two iterated refinement routes through the same family. The directed-system theorem gives a terminal packet for each route, and Definition 291.96 identifies the two packets because their signed mantissa ledger, exponent histories, denominator ledger, endpoint comparisons, support rows, and downstream four-face rows are the same finite data. Thus parenthesization and route choices cannot change the terminal tolerance row.

It remains to justify the consumer-readable factorization. The terminal four-face certificate of the directed system is the route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Applying Definition 291.123 reads the StreamName finite-window face, the RegSeqRat regularity face, and the Real seal from that same terminal support, so each downstream consumer factors through $N_{\mathcal{F}}$ instead of through a separate pairwise refinement argument.

Finally, the exclusion clause follows from the same three inputs. The directed-system construction uses only finite accepted DyadicRatCore windows and positive-unary histories; uniqueness adds no hidden carrier; and terminal four-face exhaustion lists exactly the visible downstream support. Hence quotient rational carriers, quotient stream equality, selected Real endpoints, countable-choice rows, metric-completeness rows, bridge rows, and ambient Real-completeness data have no coordinate in $N_{\mathcal{F}}$. □

Theorem 291.100 (DyadicRatCore terminal refinement absorption)**.** *Let $\mathcal{F}$ be a finite family of accepted $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance windows whose signed mantissa rows are read from one displayed $\mathsf{Int}^{\uparrow}$ ledger and whose denominator rows are presented by finite positive-unary exponent histories. Let $T_{\mathcal{F}}$ be its terminal common refinement, and let P be another accepted $\mathsf{DyadicRatCore}^{\uparrow}$ window. Suppose P already maps to $T_{\mathcal{F}}$ by the displayed signed mantissa row, positive-unary exponent row, denominator row, endpoint comparison, distance and radius rows, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows. Then the terminal refinement of $\mathcal{F}\cup\{P\}$ determines the same terminal tolerance row as $T_{\mathcal{F}}$ and the same four-face exit certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The absorption statement exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ endpoint, countable-choice row, metric-completeness row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Proof. Apply Definition 291.99 to the family $\mathcal{F}$. It presents $T_{\mathcal{F}}$ as the consumer-readable normal form for the signed mantissa ledger, exponent histories, denominator rows, endpoint comparisons, distance and radius rows, hsame, Cont, Pkg, local naming rows, and downstream four-face support.

Now apply Definition 291.95 to $T_{\mathcal{F}}$ and the additional window P. The displayed map from P into $T_{\mathcal{F}}$ already provides every row the pairwise refinement may inspect, so the pairwise terminal packet carries the same signed mantissa, exponent, denominator, endpoint, distance, radius, transport, continuation, package, and naming support as $T_{\mathcal{F}}$.

Use Definition 291.96 to compare this pairwise terminal packet with $T_{\mathcal{F}}$ itself. Since the visible DyadicRatCore rows and the downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces agree, uniqueness identifies the terminal tolerance row.

Finally apply Definition 291.123. The shared terminal tolerance row is exactly the finite four-face exit certificate, so adjoining P adds no second terminal packet and no external quotient, selected endpoint, choice, metric-completeness, bridge, or ambient Real-completeness coordinate. □

Theorem 291.101 (DyadicRatCore common-scale terminal dichotomy)**.** *Let $\mathcal{R}$ be an accepted terminal tolerance route for a finite family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows whose signed mantissa rows are read from one displayed* $\mathsf{Int}^{\uparrow}$ *ledger and whose denominator rows are presented by finite positive-unary exponent histories. Then exactly one of the following BEDC-visible alternatives holds:*

1. *$\mathcal{R}$ factors through the directed common-scale normal form $N_{\mathcal{F}}$ of Definition 291.99, and its terminal tolerance row is the finite four-face exit certificate over*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow};$$

2. *the route omits at least one displayed signed mantissa row, exponent row, denominator row, endpoint row, tolerance row,* hsame *row,* Cont *row,* Pkg *row, or local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *row required by that normal form.*

The dichotomy exports no quotient rational carrier, quotient stream equality, selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable-choice row, metric-completeness row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First suppose the route displays every listed signed mantissa, exponent, denominator, endpoint, tolerance, hsame, Cont, Pkg, and local naming row. Apply Definition 291.99. The route is repacked into the consumer-readable normal form $N_{\mathcal{F}}$, and every downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ face factors through that packet.

Use Definition 291.96 to remove route choices. Any second accepted route with the same displayed rows has the same terminal tolerance row and the same finite four-face exit certificate, so the first alternative is determined by the visible data rather than by a hidden refinement choice.

Now suppose one of the displayed rows is absent. The normal-form theorem cannot be applied, because each factorization coordinate of $N_{\mathcal{F}}$ is a visible row: signed mantissa ledger, exponent history, denominator ledger, endpoint comparison, tolerance support, transport, continuation, package provenance, and local NameCert data.

Finally apply Definition 291.123. A successful terminal route exhausts exactly the four faces through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Therefore a route that lacks one required displayed row has no fifth source from which to recover it, and it falls into the failure alternative. □

Theorem 291.102 (DyadicRatCore terminal radius is choice-free)**.** *Let P and Q be accepted L10 terminal exits whose displayed packets share the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *face, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window face, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback face, the same local* $\mathsf{Real}^{\uparrow}$ *seal face, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Assume moreover that both exits present their dyadic tolerance data by finite positive-unary scale histories over the same signed mantissa and denominator ledgers. Then the terminal radius row displayed at the $\mathsf{Real}^{\uparrow}$ *seal is uniquely determined by the finite common-scale directed refinement of those rows.*

In particular, the terminal radius cannot be replaced by quotient rational equality, quotient stream equality, a selected $\mathsf{Real}^{\uparrow}$ *endpoint, countable choice, metric completeness, a bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. It is the finite DyadicRatCore radius row read through the displayed four-face support.*

Proof. First put the two dyadic presentations on a common finite scale. Apply Definition 291.95 to the two accepted finite tolerance windows. The result supplies one terminal $\mathsf{DyadicRatCore}^{\uparrow}$ window carrying the signed mantissa, denominator, endpoint, distance, radius, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ rows.

Now compare possible terminal presentations of that same window. The uniqueness theorem Definition 291.96 identifies any two common terminal refinements that preserve the same finite rows and the same downstream four-face surface. Hence their terminal tolerance rows agree.

The common finite packet is the packet seen by the terminal route. Apply Definition 291.83 to read the DyadicRatCore face together with the StreamName finite window, the RegSeqRat classifier and readback rows, the Real seal row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally use Definition 291.84 and Definition 291.123. The first result rules out a second dyadic source behind the same downstream surface, and the second forces every public tolerance read at the terminal route through the displayed four faces. Since the radius row is one of the finite DyadicRatCore tolerance rows in that surface, no quotient, selected endpoint, choice, completeness, bridge, or ambient Real-completeness datum can determine a different terminal radius. □

Theorem 291.103 (DyadicRatCore common-scale terminal radius normal form)**.** *For every accepted L10 terminal exit whose visible surface is the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support, the following presentations determine the same terminal radius row:*

1. *the public four-face terminal packet carrying the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal row;*
2. *the common-scale normal form obtained by finite directed refinement of the dyadic tolerance windows over their signed mantissa and denominator ledgers.*

Thus the terminal four-face route already presents the radius in common-scale normal form. The normal form exports no quotient rational equality, quotient stream equality, selected endpoint, countable-choice row, metric-completeness row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted four-face terminal packet. By Definition 291.123, every public tolerance read factors through the DyadicRatCore tolerance ledger together with the StreamName finite-window face, the RegSeqRat classifier and readback face, the Real seal face, and the displayed support rows.

Build the common-scale presentation from the same finite dyadic data. Apply Definition 291.97 to the finite family of accepted dyadic tolerance windows visible in the packet. Then apply Definition 291.99; this gives one consumer-readable normal form through which every downstream StreamName, RegSeqRat, and Real face factors.

It remains to identify the radius row across the two presentations. Use Definition 291.96 to compare the common-scale terminal packet with the DyadicRatCore face recovered from the four-face route. Their signed mantissa, denominator, endpoint, distance, radius, hsame, Cont, Pkg, local NameCert, and downstream rows are the same displayed finite rows, so the terminal tolerance row agrees.

Finally apply Definition 291.102. Since the terminal radius row is already forced by the finite common-scale directed refinement, the four-face packet cannot read a second radius from quotient rational equality, quotient stream equality, a selected endpoint, countable choice, metric

completeness, a bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The terminal four-face route and the common-scale normal form therefore present one and the same radius row. □

Theorem 291.104 (DyadicRatCore terminal radius half-step inversion)**.** *If a terminal four-face packet exposes a common-scale normal form and a public half-radius read, then the half-radius row is determined by finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-scale refinement before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal consumer opens. No quotient rational equality, selected endpoint, countable choice, metric completeness, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness can alter the half-radius row.*

Proof. Apply Definition 291.103. It identifies the public four-face terminal packet with the common-scale normal form obtained by finite directed refinement of the displayed dyadic tolerance windows.

Next apply Definition 291.102. The terminal radius row is forced by the finite DyadicRatCore tolerance data already visible in that common-scale packet.

Finally use Definition 291.83. The public half-radius read is a downstream read of the same four-face surface, so it is pulled back to the finite DyadicRatCore common-scale refinement before any StreamName, RegSeqRat, or Real seal consumer can add data. □

Theorem 291.105 (DyadicRatCore choice-free halving cover)**.** *Let a terminal four-face packet expose a finite unary schedule of terminal half-radius reads along the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Then the whole schedule is determined by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *common-scale normal form together with the iterated halving budget. Each halving step preserves the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Consequently no quotient rational equality, selected endpoint, countable choice, bridge row, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle supplies an alternate terminal radius at any finite stage of the schedule.*

Proof. Begin with the terminal radius choice-free theorem. It makes the first terminal radius row a finite DyadicRatCore read through the displayed common-scale refinement and the four-face downstream surface.

Apply the common-scale terminal radius normal form and the half-step inversion theorem. Together they identify each public half-radius read with a finite refinement step before any StreamName, RegSeqRat, or Real consumer can add data.

Now use the iterated halving budget, Definition 291.138. A finite unary schedule is obtained by repeating the same half-step over the displayed dyadic ledger, so every stage remains a common-scale DyadicRatCore row transported by the same BHist, hsame, Cont, Pkg, and NameCert support.

The schedule has no remaining coordinate from which to read a second radius. Quotient rational equality, selected endpoints, countable choice, bridge data, metric completeness, and ambient Real completeness are absent from the normal form and therefore cannot alter the finite halving cover. □

Theorem 291.106 (DyadicRatCore StreamName RegSeqRat Real L10 exit ledger)**.** *Every accepted L10 exit read whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *has a unique BEDC-visible dyadic source ledger. The forced ledger consists exactly of the signed*

mantissa row, refined positive-unary exponent row, denominator ledger, common-exponent comparison, distance and radius rows, $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, local* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows.*

The exit cannot be supplied by quotient rational equality, quotient stream equality, host real equality, a selected limit, countable choice, metric-completeness rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 291.82 to the dyadic source faces visible in the L10 exit read. It supplies the common finite scale-window packet with signed mantissa, refined positive-unary exponent, denominator, comparison, distance, radius, transport, continuation, package, and local DyadicRatCore naming rows.

Next use Definition 291.83. The same finite packet is read as the four displayed faces

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

so the StreamName finite window, RegSeqRat classifier and readback surface, Real seal row, and displayed support coordinates are not additional sources.

Finally apply Definition 291.84 and Definition 291.87. Once the four visible faces and the displayed BHist, hsame, Cont, Pkg, and NameCert rows agree, the dyadic scale-window source is forced to be the same common finite triangle ledger, and the exit inventory has no fifth source.

The non-escape clause is the combined exclusion inventory of the cited triangle ledger, four-face correspondence, source-uniqueness theorem, and no-fifth-source theorem. Those clauses leave no row for quotient rational equality, quotient stream equality, host real equality, selected limits, countable choice, metric completeness, standard bridges, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.107 (DyadicRatCore L10 exit NameCert interface)**.** *An accepted Phase Real and Completion exit packet is a local* NameCert *interface exactly when the following facewise rows are present on one BEDC-visible support surface: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows.*

Equivalently, the four faces $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *jointly present the certificate surface consumed by the Phase Real and Completion exit criterion. Deleting any face, aggregating two faces into one status row, or replacing a face by a global equality datum fails to present that interface. The interface has no coordinate for quotient rational equality, quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, standard bridge rows, aggregate formalstatus rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreL10ExitNameCertInterface`

Proof. For sufficiency, start with the displayed four-face inventory and support rows. Apply Definition 291.106 to read the dyadic source surface as the unique BEDC-visible ledger of the L10 exit.

Next apply Definition 126.504. It repacks the same DyadicRatCore, StreamName, RegSeqRat, and Real rows as the terminal four-object boundary seen by every accepted Phase Real and Completion consumer.

Use Definition 301.60 to identify the RegSeqRat classifier and readback face with that terminal boundary. The classifier is therefore not an external parameter transported into the packet; it is one of the four exposed faces of the same local certificate surface.

Finally apply Definition 38.793. Minimality says that each of the four faces and each displayed support row is indispensable for the current Real L10 exit certificate, so the assembled packet is precisely the local NameCert interface.

For necessity, assume an accepted packet presents the local NameCert interface. The StreamName terminal pullback forces the packet to factor through the four-object boundary, and the RegSeqRat correspondence reads the classifier surface facewise inside that boundary.

The DyadicRatCore ledger theorem then recovers the signed mantissa, exponent, denominator, comparison, distance, and radius rows as the unique dyadic source ledger. Real-side face minimality prevents deletion or aggregation of any of the four faces, and it also rules out replacing a face by quotient rational equality, quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, standard bridge rows, aggregate formalstatus rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.108 (DyadicRatCore L10 exit public bridge route)**.** *Every public* $\mathsf{DyadicRatCore}^{\uparrow}$ *bridge consumer that is read by a Phase Real and Completion exit packet factors through one common monotone unary radius window and then through the four visible faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over the shared BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. This factorization is exactly the L10 exit interface: the public bridge supplies the dyadic source rows, the finite radius-window cofinality supplies the common monotone window, and the four-face exit packet supplies the local certificate surface.*

The route has no coordinate for quotient rational equality, quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public bridge consumer. Apply Definition 291.24. The consumer reads exactly the signed mantissa, positive-unary exponent, denominator ledger, common-exponent classifier, arithmetic window, distance window, radius window, BHist carrier rows, hsame transport, Cont readback, and Pkg provenance already visible at the public dyadic bridge.

Now apply Definition 291.33 to the finite family of downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads of those public packets. It places the family on one common monotone unary radius window with the shared refined exponent, denominator ledger, classifier, distance rows, BHist rows, hsame transports, Cont readbacks, and Pkg provenance.

Next use Definition 291.107. The same rows, together with the $\mathsf{StreamName}^{\uparrow}$ finite-window rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback rows, local $\mathsf{Real}^{\uparrow}$ seal rows, and local NameCert support, form precisely the interface consumed by the accepted Phase Real and Completion exit packet.

Finally apply Definition 126.504, Definition 301.60, and Definition 38.793. These results identify the common radius-window packet with the terminal four-face boundary and make each displayed face and support row indispensable. The exclusion clause is inherited from the public bridge exactness, finite cofinality, NameCert interface, terminal pullback, RegSeqRat correspondence, and Real face-minimality inventories, none of which exposes quotient rational equality, quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.109 (DyadicRatCore L10 exit ledger facewise noncollapse)**.** *An accepted Phase Real and Completion exit ledger presents the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face if and only if the signed mantissa, exponent, denominator, comparison, distance, and radius rows remain a distinct face alongside*

the $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier readback,* $\mathsf{Real}^{\uparrow}$ *seal, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

In this correspondence no aggregate formalstatus row, bridge row, quotient rational equality, quotient stream equality, selected limit, metric-completeness datum, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row can replace the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face.*

Proof. For the forward direction, assume the accepted exit ledger presents the $\mathsf{DyadicRatCore}^{\uparrow}$ face. Apply Definition 291.106. The dyadic source surface is then forced to be the unique BEDC-visible ledger consisting of the signed mantissa, exponent, denominator, comparison, distance, and radius rows together with the displayed support rows.

Next apply Definition 291.107. Since the packet presents the local NameCert interface, those dyadic rows must appear facewise with the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows rather than as a collapsed status coordinate.

For the reverse direction, start with the distinct dyadic rows, the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and the displayed support rows. Use Definition 291.107 to read this inventory as the local NameCert interface for the accepted Phase Real and Completion exit packet.

Apply Definition 126.504. The four exposed faces form the terminal finite packet seen by downstream Real-and-completion consumers, so the distinct dyadic rows are presented as the $\mathsf{DyadicRatCore}^{\uparrow}$ face of that packet.

It remains to rule out substitute rows. Apply Definition 38.793. Each of the four faces and each displayed support row is indispensable for the current exit certificate; hence no aggregate formalstatus row, bridge row, quotient rational equality, quotient stream equality, selected limit, metric-completeness datum, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness row can stand in for the $\mathsf{DyadicRatCore}^{\uparrow}$ face. □

Theorem 291.110 (DyadicRatCore Phase Real and Completion objectwise exit equivalence)**.** *An accepted L10 exit packet satisfies the current Phase Real and Completion objectwise exit criterion if and only if the four facewise rows are simultaneously present on one BEDC-visible support surface: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows.*

No quotient rational equality, quotient stream equality, selected limit, countable choice, metric-completeness datum, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness can replace any of the four faces or any shared support row in this criterion.*

Proof. For the forward direction, assume the accepted L10 packet satisfies the current objectwise exit criterion. Apply Definition 38.796. The threshold reads the criterion through local Real exit rows, so the packet must expose the Real seal together with the support rows consumed by the Phase Real and Completion surface.

Apply Definition 126.504. The same local Real surface is the terminal four-object boundary with visible $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. Use Definition 301.60 to read the RegSeqRat classifier and readback rows as one of those exposed faces.

It remains to recover the dyadic source rows and the local certificate interface. Apply Definition 291.107. Then apply Definition 291.109. These two results force the signed mantissa, exponent, denominator, comparison, distance, and radius rows to remain present as the distinct $\mathsf{DyadicRatCore}^{\uparrow}$ face, with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the reverse direction, start with the simultaneous four-face inventory and the shared support rows. Apply Definition 291.107 to package this inventory as the local NameCert interface for the accepted L10 exit packet.

Next apply Definition 291.109. The dyadic rows do not collapse into an aggregate status coordinate. Apply Definition 126.504 and Definition 301.60 to transport the same inventory to the terminal four-object boundary.

Finally apply Definition 38.796. Since the local Real threshold sees exactly that four-object boundary and the displayed support surface, the packet satisfies the current Phase Real and Completion objectwise exit criterion.

The exclusion sentence follows from the noncollapse theorem together with the terminal pullback, the RegSeqRat correspondence, and the Real local threshold. None of those cited results exposes a coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, bridge rows, aggregate formalstatus rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none can replace any face or shared support row. □

Theorem 291.111 (DyadicRatCore four-face exit facewise certificate)**.** *An accepted Phase Real and Completion exit packet is certificate-complete exactly when it exposes four separate local certificate faces over one shared support surface: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed-mantissa, exponent, denominator, comparison, distance, and radius ledger; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, readback, and regularity rows; and the local* $\mathsf{Real}^{\uparrow}$ *seal, all over the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

No quotient rational equality, quotient stream equality, selected limit, countable choice, metric-completeness row, aggregate formalstatus row, aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row may replace any of the four certificate faces.*

Proof. For the forward direction, apply Definition 291.110. It identifies an accepted certificate-complete packet with the simultaneous four-face inventory on one support surface, including the dyadic ledger, StreamName window, RegSeqRat rows, local Real seal, and shared BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 291.107. The same inventory is not merely an objectwise criterion; it is the local certificate package consumed by the accepted Phase Real and Completion exit packet.

The StreamName and RegSeqRat faces are read locally. Apply Definition 126.657 for the finite-window schedule and Definition 301.239 for the classifier, readback, and regularity rows. Use Definition 301.246 to keep those rows as a finite local cover rather than a quotient stream or selected limit.

For the reverse direction, start with the four exposed certificate faces and the displayed support rows. The NameCert interface theorem packages them as the local certificate surface, and the objectwise exit equivalence reads that surface as the accepted Phase Real and Completion exit packet.

Finally apply Definition 38.746. Together with the objectwise equivalence and the NameCert interface, it rules out aggregation and off-table replacements. Hence quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness rows cannot stand in for any certificate face. □

Theorem 291.112 (DyadicRatCore L10 face status pullback)**.** *Let P be an accepted Phase Real and Completion exit packet whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Then P pulls back exactly the four local status records from the four corresponding face chapters' own* closurestatus *and* formalstatus *records. The* $\mathsf{DyadicRatCore}^{\uparrow}$ *face contributes the signed*

mantissa, exponent, denominator, comparison, distance, and radius ledger; the $\mathsf{StreamName}^{\uparrow}$ *face contributes the finite-window status; the* $\mathsf{RegSeqRat}^{\uparrow}$ *face contributes the classifier and readback status; and the* $\mathsf{Real}^{\uparrow}$ *face contributes the local seal status.*

This pullback has no aggregate formalstatus *row, aggregate bridge token, route-level Lean target, quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Apply Definition 291.110. It identifies the accepted packet with the simultaneous four-face inventory on one BEDC-visible support surface: the dyadic source ledger, the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback rows, the $\mathsf{Real}^{\uparrow}$ seal rows, and the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply Definition 291.109. The signed mantissa, exponent, denominator, comparison, distance, and radius rows remain the distinct $\mathsf{DyadicRatCore}^{\uparrow}$ face. They therefore cannot be re-read as a route status, bridge token, equality quotient, selected limit, choice datum, completeness datum, or ambient real-completion row.

Now apply Definition 126.507. Each visible face imports closure grade, verification token, Lean target, and bridge row only by reference to that face chapter's own closurestatus block. Hence the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ entries remain their own finite-window, classifier-readback, and seal status records, while the dyadic ledger is the local record supplied by the distinct DyadicRatCore face.

Finally apply Definition 38.494. The Phase Real and Completion threshold is read object by object on these four local records. Thus the packet can be consumed as a finite paper-side boundary only through the four pulled-back face records, and never through aggregate formalstatus, aggregate bridge, route-level Lean target, quotient rational equality, quotient stream equality, selected limit, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.113 (DyadicRatCore L10 exit Lean touchpoint completeness)**.** *Let P be an accepted Phase Real and Completion exit packet whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Then the machine-facing support of P is exhausted by the three route touchpoints named by Definition 126.452 and Definition 126.453: the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff. These three route sites are read together with the four local face records pulled back by Definition 291.112.*

Thus the accepted four-face exit has no additional route-level Lean target, aggregate formalstatus *row, bridge row, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply Definition 291.112. The accepted packet is read through the four local records carried by the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, and the pullback already excludes aggregate status, aggregate bridge, route-level Lean targets, quotient stream equality, selected limits, countable choice, metric completeness, and ambient real completeness.

Next apply Definition 126.453 together with Definition 126.452. These theorems identify the only machine-facing route sites for the same open-phase exit as the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff.

Use Definition 301.60 to identify the RegSeqRat classifier and readback rows with the terminal four-face boundary. The readback face is therefore one of the local records already pulled back from the accepted packet, not a separate quotient stream equality or selected limit.

Finally apply Definition 38.494. The open-phase threshold is consumed object by object on the four local face records, while the three named Lean touchpoints merely expose the existing finite-window, dyadic-handoff, and seal-handoff route. Hence no machine support remains outside those sites, and none of the excluded rows can be added as a companion route coordinate. □

Theorem 291.114 (DyadicRatCore L10 exit ledger source determinacy)**.** *Let P and Q be accepted Phase Real and Completion objectwise audit reads. Suppose both reads expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the same local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Then P and Q determine the same BEDC-visible dyadic source ledger and the same four-face exit packet.*

This determinacy has no coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 291.112 to both accepted reads. The hypotheses give the same four local status records on the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, so both reads pull back the same facewise status packet.

Next use Definition 291.110. Since the objectwise audit criterion is exactly the simultaneous four-face inventory with the displayed support rows, P and Q have the same accepted objectwise exit record rather than two independent route credentials.

Now apply Definition 291.109. The common signed mantissa, exponent, denominator, comparison, distance, and radius rows remain the distinct dyadic face inside each packet. Thus the dyadic source ledger read from P is the same BEDC-visible ledger read from Q.

Finally apply Definition 126.504 and Definition 38.494. The common four faces and support rows factor through one terminal finite Real-completion boundary, and the threshold is consumed object by object on that boundary. Hence the two reads determine the same four-face exit packet.

The exclusions follow from the same citations: the pulled-back face-status packet, objectwise exit equivalence, facewise noncollapse, terminal pullback, and objectwise threshold expose no row for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.115 (DyadicRatCore L10 exit ledger restriction functoriality)**.** *Let P be an accepted Phase Real and Completion four-face packet, and let P' be a sub-exit packet obtained by deleting only downstream consumer rows. Suppose the restriction preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Then the DyadicRatCore ledger read from P' is the restriction of the DyadicRatCore ledger read from P. The restriction introduces no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply Definition 291.114 to the accepted read P and to the restricted read P'. The preservation hypotheses give the same dyadic source rows, StreamName window rows, RegSeqRat classifier-readback rows, Real seal rows, and displayed support rows on the common part, so the two reads determine the same BEDC-visible dyadic source ledger there.

Next apply Definition 291.112. Both packets read their closure and verification data face by face from the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ records. Deleting downstream consumer rows therefore restricts the visible face-status packet; it does not create a new route-level status coordinate.

Now use Definition 291.110 together with Definition 126.504. These theorems identify the shared source as the same terminal four-face boundary with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. Hence the restricted read factors through the restriction of the terminal boundary already read by P.

Finally apply Definition 291.109 and Definition 38.494. The dyadic rows remain the distinct $\mathsf{DyadicRatCore}^{\uparrow}$ face, and the Phase Real and Completion threshold is consumed objectwise on the four local records. Thus no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness row appears in the restriction. □

Theorem 291.116 (DyadicRatCore L10 exit ledger common restriction compatibility)**.** *Let P and Q be accepted Phase Real and Completion four-face packets, and let P' and Q' be sub-exit packets obtained from them by deleting only downstream consumer rows. Suppose P' and Q' preserve the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the same local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the same shared* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support.*

Then the DyadicRatCore ledger read from P' is the same BEDC-visible ledger read from Q', and the two sub-exit packets determine the same restricted four-face exit packet. This compatibility has no coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 291.115 to the pair P, P' and again to the pair Q, Q'. The hypotheses say that both restrictions preserve the same dyadic rows, StreamName finite-window rows, RegSeqRat classifier-readback rows, Real seal rows, and displayed support rows, so the ledger read from each restricted packet is the corresponding restricted ledger of its accepted source packet.

Next apply Definition 291.114 to the two accepted source reads on the common displayed rows. The common BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support forces the BEDC-visible dyadic source ledger read from P and the one read from Q to agree on that support. Hence their restricted ledgers agree, and the ledger read from P' is the same ledger read from Q'.

It remains to identify the restricted exit packet. Use Definition 291.110 to read each restricted packet through the same simultaneous four-face inventory. Then apply Definition 126.504; the shared DyadicRatCore, StreamName, RegSeqRat, Real, and support rows factor through one terminal finite Real-completion boundary. Thus P' and Q' determine the same restricted four-face exit packet.

The exclusion clause is inherited from the restriction functoriality theorem, source determinacy, objectwise exit equivalence, and terminal pullback. None of those cited surfaces exposes a row for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.117 (DyadicRatCore L10 exit ledger restriction composition)**.** *Let P be an accepted Phase Real and Completion four-face packet, let P' be obtained from P by deleting only downstream*

consumer rows, and let P'' be obtained from P' by the same kind of downstream-consumer restriction. Suppose both restrictions preserve the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Then the DyadicRatCore ledger read from P'' is exactly the direct restriction of the DyadicRatCore ledger read from P. Equivalently, the two-step restricted four-face exit packet determined by $P \mapsto P' \mapsto P''$ is the same restricted four-face exit packet obtained by deleting the combined downstream consumer rows from P in one step.

This composition law introduces no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply Definition 291.115 to P and P'. The ledger read from P' is the restriction of the ledger read from P along the preserved DyadicRatCore, StreamName, RegSeqRat, Real, and support rows.

Apply Definition 291.115 again to P' and P''. The second read is therefore the restriction of the P' ledger along the rows still visible after the second downstream deletion.

Now use Definition 291.116 on the intermediate restriction and the direct restriction of P to the rows visible in P''. Both routes preserve exactly the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows, so the ledger obtained in two steps agrees with the ledger obtained in one step on the common support.

It remains to identify the four-face packet, not only the ledger rows. Apply Definition 291.114 to the two reads on the shared displayed rows, and then apply Definition 126.504. The common DyadicRatCore, StreamName, RegSeqRat, Real, and support surface factors through one terminal finite Real-completion boundary. Hence the two-step restricted packet and the direct restricted packet are the same four-face exit packet.

The exclusion clause follows from the two uses of restriction functoriality, common-restriction compatibility, source determinacy, and the terminal pullback. None of these cited surfaces exposes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.118 (DyadicRatCore L10 exit ledger consumer normal form)**.** *Let C be a Phase Real and Completion consumer of an accepted four-face packet with visible* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces. Then the rows readable by C are exactly the DyadicRatCore signed mantissa, exponent, denominator, comparison, distance, and radius ledger; the StreamName finite-window rows; the RegSeqRat classifier and readback rows; the local Real seal rows; and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Conversely, any Phase Real and Completion consumer exposing precisely those rows determines the same L10 exit ledger and the same four-face exit packet. The normal form has no row for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 126.504. The accepted packet is read through one terminal finite Real-completion boundary whose visible faces are exactly the DyadicRatCore, StreamName, RegSeqRat, and Real records. Apply Definition 38.494; the Phase Real and Completion threshold is consumed

object by object on that boundary, so a downstream consumer can read only the listed local rows and the displayed support rows.

Next apply Definition 291.115. Deleting rows outside the displayed inventory is a downstream-consumer restriction, and the DyadicRatCore ledger read after restriction is the restriction of the ledger read from the accepted source packet. Apply Definition 291.116 to two such consumer restrictions; if they preserve the same displayed rows, they determine the same restricted ledger and the same restricted four-face packet.

For the converse, apply Definition 291.114. A consumer exposing precisely the dyadic ledger rows, StreamName finite-window rows, RegSeqRat classifier-readback rows, Real seal rows, and shared BHist, hsame, Cont, Pkg, and NameCert support has the full BEDC-visible source data used by the accepted objectwise read. Source determinacy therefore identifies its L10 exit ledger with the ledger read by any other consumer exposing precisely the same rows.

Finally combine the terminal pullback and objectwise threshold equivalence with the exclusion clauses in source determinacy, restriction functoriality, and common restriction compatibility. None of the cited surfaces adds quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the displayed inventory is both sufficient for consumption and maximal among the rows exposed by the current Phase Real and Completion objectwise exit. □

Theorem 291.119 (DyadicRatCore L10 exit ledger finite-cover descent)**.** *Let $\{P_i\}_{i\in I}$ be a finite cover of accepted Phase Real and Completion four-face packets. Suppose every P_i exposes the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the same local* $\mathsf{Real}^{\uparrow}$ *seal rows; and shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support on all displayed overlaps. Suppose also that the overlap restrictions preserve those rows and that pairwise common restrictions agree on the displayed support.*

Then the finite cover descends to one BEDC-visible DyadicRatCore ledger and one accepted four-face exit packet with visible $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces. The descended packet has no coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First normalize each cover member. Apply Definition 291.118 to every P_i. Each accepted packet is therefore read through the same displayed DyadicRatCore ledger rows, StreamName finite-window rows, RegSeqRat classifier-readback rows, local Real seal rows, and shared support surface.

Now compare overlaps. Use Definition 291.116 on every pairwise overlap. The common restriction hypotheses identify the restricted DyadicRatCore ledger and the restricted four-face exit packet on the overlap, so no overlap introduces an additional equality, choice, completeness, status, or bridge coordinate.

Next paste the four visible faces. Apply Definition 38.771 to the local Real and Completion four-face packets, and apply Definition 126.552 to exhaust the StreamName finite-window and RegSeqRat seal rows over the same finite cover. The pasted surface still carries the displayed DyadicRatCore rows and the same BHist, hsame, Cont, Pkg, and NameCert support.

It remains to pass from compatible local ledgers to one global ledger. Induct over the finite cover. The empty stage is the empty displayed restriction, and the first nonempty stage is normalized by the consumer normal form. At the adjoining step, Definition 291.115 transports the already

pasted ledger to the common part, while source determinacy from Definition 291.114 identifies the adjoining member with that same ledger on the preserved rows.

The induction produces one BEDC-visible DyadicRatCore ledger. The pasted Real finite-cover packet and the StreamName-RegSeqRat seal exhaustion supply the corresponding four-face exit packet. The exclusion clause follows from the normal form, common restriction compatibility, Real finite-cover face pasting, and StreamName-RegSeqRat finite-cover seal exhaustion: none of these cited surfaces exposes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.120 (DyadicRatCore L10 exit ledger finite-cover refinement invariance)**.** *Let $\{P_i\}_{i\in I}$ be a finite cover of accepted Phase Real and Completion four-face packets, and let $\{Q_j\}_{j\in J}$ be a finite cover refining it by a finite assignment $j \mapsto i(j)$. Suppose each Q_j lies over $P_{i(j)}$, and every refinement map preserves the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Suppose also that refined overlaps preserve the same displayed rows and agree with the corresponding coarse overlap restrictions on the shared support.*

Then applying finite-cover descent to the coarse cover $\{P_i\}_{i\in I}$ or to the refined cover $\{Q_j\}_{j\in J}$ yields the same BEDC-visible DyadicRatCore ledger and the same accepted four-face exit packet with visible $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces. The invariant comparison has no coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 291.119 to the coarse cover and to the refined cover. Both outputs are built only from the displayed dyadic ledger rows, StreamName finite-window rows, RegSeqRat classifier-readback rows, local Real seal rows, and shared support rows.

For a refined member Q_j, use Definition 291.115 along the refinement map $Q_j \to P_{i(j)}$. The preservation hypotheses identify the ledger read from Q_j with the restriction of the ledger read from its coarse source on exactly the displayed rows.

On refined overlaps, apply Definition 291.116. The refined common restrictions and the corresponding coarse common restrictions agree on the same BHist, hsame, Cont, Pkg, and NameCert support, so every refined overlap carries the same restricted dyadic ledger as its coarse image.

Now use Definition 291.114. Since the coarse and refined descent outputs have the same displayed DyadicRatCore, StreamName, RegSeqRat, Real, and support rows on every finite member and overlap, source determinacy identifies the BEDC-visible DyadicRatCore ledger produced by the two descents.

It remains to compare the four visible faces. Invoke Definition 38.771 for the Real and Completion faces, and Definition 126.552 for the StreamName finite-window and RegSeqRat seal rows. These two finite pasting surfaces are computed from the same preserved rows, hence the coarse and refined covers determine the same four-face exit packet.

The exclusion clause follows from the descent theorem, restriction functoriality, common-restriction compatibility, source determinacy, Real finite-cover face pasting, and StreamName-RegSeqRat finite-cover seal exhaustion. None of those cited surfaces exposes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 291.121 (DyadicRatCore L10 exit ledger finite-support coverage)**.** *Let P be an accepted Phase Real and Completion four-face packet with visible* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces. Suppose the finite support of P is covered by displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-observation rows; displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail classifier and readback rows; displayed local* $\mathsf{Real}^{\uparrow}$ *seal rows; and shared* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Suppose also that the displayed support rows cover every overlap used by the Phase Real and Completion acceptance packet and that any two displayed finite covers admit a common finite refinement preserving the same rows.*

Then the resulting DyadicRatCore L10 exit ledger is completely covered by those finite support rows. In particular, every BEDC-visible ledger coordinate of the accepted four-face exit packet is read from the displayed DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, *and* NameCert *rows, and no further support row is required.*

This finite-support coverage theorem has no coordinate for quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the StreamName side through finite observations. Apply Definition 126.527 to the displayed StreamName finite-observation rows. This gives a finite section for exactly the observable stream windows carried by the Phase Real and Completion packet, with no appeal to quotient stream equality or selected limits.

Next read the RegSeqRat side through common tails. Apply Definition 301.193 to the displayed RegSeqRat classifier and readback rows over the same finite support. The induction supplies the common-tail support needed for the finite window rows and keeps the shared BHist, hsame, Cont, Pkg, and NameCert rows fixed on every retained overlap.

Now form the cover whose members are the displayed dyadic ledger rows together with the StreamName finite section, the RegSeqRat common-tail support, the local Real seal rows, and the shared support rows. Apply Definition 291.119. The descent theorem produces one BEDC-visible DyadicRatCore ledger and one accepted four-face exit packet from exactly that finite inventory.

It remains to show that the coverage does not depend on the chosen displayed finite cover. Given another cover of the same finite support, pass to the stipulated common finite refinement and apply Definition 291.120. The refinement comparison identifies the ledger and the four-face exit packet computed from the two covers, because both covers preserve the same DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

The exclusion clause is inherited from the StreamName finite-observation section, the RegSeqRat common-tail support induction, finite-cover descent, and finite-cover refinement invariance. None of those cited surfaces exposes quotient rational equality, quotient stream equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the accepted DyadicRatCore L10 exit ledger is completely covered by the displayed finite support inventory and by nothing outside it. □

Theorem 291.122 (DyadicRatCore Phase Real and Completion exit support minimality)**.** *An accepted Phase Real and Completion four-face packet is a valid exit certificate only if all four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

remain present over the shared BHist, hsame, Cont, Pkg, *and* NameCert *support rows. None of these four faces and none of the displayed support rows may be deleted, collapsed into an aggregate status coordinate, or replaced by quotient rational equality, quotient stream equality, selected limits,*

countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 291.107. A valid accepted packet must present the local NameCert interface, and that interface is explicitly the four-face inventory over the shared BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 291.110 and Definition 291.111. These theorems identify the objectwise exit criterion and the certificate-complete packet with the same four separate faces, so a packet missing one face or replacing a face by an aggregate coordinate no longer has the accepted exit surface.

The StreamName, RegSeqRat, and Real rows are forced facewise. Apply Definition 126.657, Definition 301.239, and Definition 301.246. These results keep the finite-window, classifier, readback, regularity, and seal rows as local BEDC rows rather than quotient stream equality, selected limits, countable choice, or metric-completeness data.

It remains to rule out replacement by route-level status or ambient completeness data. Apply Definition 38.746; then apply Definition 291.121. The first citation separates the objectwise support route from aggregate formalstatus and bridge rows, while the finite-support coverage theorem says every BEDC-visible ledger coordinate is already read from the displayed DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Hence deleting any listed face or support row leaves no theorem in the cited chain that can supply the missing coordinate, and every proposed substitute lies outside the finite support inventory. The accepted four-face packet is therefore a valid exit certificate only with exactly those indispensable faces and shared support rows present. □

Theorem 291.123 (DyadicRatCore terminal four-face tolerance exhaustion)**.** *Let P be an accepted L10 terminal route whose four visible faces are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and classifier row, and the local* $\mathsf{Real}^{\uparrow}$ *seal row over the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Then every public tolerance read of P factors through exactly that $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger together with the three downstream faces and the displayed support rows.*

No quotient rational, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is an additional tolerance coordinate for the terminal route.*

Lean4 ✓ `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreTerminalFourFaceToleranceExhaustion`

Proof. First apply Definition 291.83. It identifies the terminal route with exactly the four visible faces: the DyadicRatCore tolerance ledger, the StreamName finite window, the RegSeqRat classifier and readback surface, and the Real seal row, all over the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Next use Definition 291.87. Once the four faces and support rows are fixed, the exit inventory has no fifth source from which another tolerance coordinate could be read.

The StreamName side keeps the tolerance coordinate inside the same finite packet. Apply Definition 126.471 and Definition 126.472; these results say that the accepted open-phase read exposes the ledger through the four faces themselves, not through quotient stream equality, selected limits, ambient completion, or standard bridge data.

Now read the RegSeqRat and Real faces. The non-escape theorem Definition 301.169 confines the readback consumer to the displayed dyadic observation rows, StreamName window rows, RegSeqRat budget rows, transports, continuations, provenance, and local naming rows.

Finally apply Definition 38.464. The terminal Real boundary determines exactly the fused StreamName packet, the DyadicRatCore endpoint and radius ledger, the RegSeqRat finite-request section, the local Real seal, and the displayed support. Hence every public tolerance read of P factors through the DyadicRatCore ledger already present in the four-face packet, and none of the excluded quotient, selected-limit, choice, completeness, metric-completeness, bridge, or ambient $\mathsf{Real}^{\uparrow}$ rows can be added as a separate tolerance coordinate. □

Theorem 291.124 (DyadicRatCore terminal tolerance finite-support readback)**.** *Let P be an accepted L10 terminal route whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *face over the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Then any public tolerance read of P is determined by the displayed finite support rows: the $\mathsf{DyadicRatCore}^{\uparrow}$ *signed mantissa, exponent, denominator, comparison, distance, and radius rows; the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows; the local* $\mathsf{Real}^{\uparrow}$ *seal row; and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

The readback exports no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 291.123. Every public tolerance read of P factors through the four visible faces and the shared support rows, so it is enough to identify the finite support carried by those faces.

The DyadicRatCore support is finite. Apply Definition 291.121; the accepted four-face packet reads the signed mantissa, exponent, denominator, comparison, distance, radius, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows from the displayed finite inventory.

Now read the three downstream faces on the same inventory. The StreamName face is supplied by Definition 126.527. The RegSeqRat classifier and readback face is supplied by Definition 301.193. The local Real seal is supplied by Definition 38.464.

Finally use the no-fifth-source clause from Definition 291.123. Since all public tolerance reads already factor through the displayed four faces and those faces have just been read from the finite support rows, none of the excluded quotient, selected-limit, countable-choice, metric-completeness, bridge, aggregate formalstatus, or ambient $\mathsf{Real}^{\uparrow}$ completeness rows can be exported by the readback. □

Theorem 291.125 (DyadicRatCore terminal tolerance projection)**.** *Let P and Q be accepted L10 terminal routes whose visible four-face packets expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and classifier row, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Then the public tolerance reads of P and Q agree if and only if their $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger rows agree.*

No quotient rational, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row can mediate a different tolerance read between the two terminal routes.*

Proof. Apply Definition 291.123 to P and to Q. Each public tolerance read is forced through the displayed four-face packet: the DyadicRatCore tolerance ledger, the StreamName finite window, the RegSeqRat classifier and readback row, the local Real seal row, and the shared BHist, hsame, Cont, Pkg, and NameCert support.

For the forward direction, suppose the public tolerance reads agree. Use Definition 291.83. Since the StreamName, RegSeqRat, Real, and support faces are already the same, the pullback has only one remaining tolerance coordinate; the agreement of the public reads therefore identifies the DyadicRatCore tolerance ledger row.

For the reverse direction, suppose the DyadicRatCore tolerance ledger rows agree. The other three faces and all displayed support rows agree by hypothesis, so the same pullback theorem gives one common four-face packet for both routes. Reading the tolerance face through that common packet gives the same public tolerance read.

Finally apply Definition 291.87. The exit inventory has no fifth source that could alter the tolerance coordinate after the four visible faces and support rows have been identified. The StreamName rows Definition 126.471 and Definition 126.472, together with the RegSeqRat nonescape row Definition 301.169, rule out external completion, quotient stream, quotient rational, selected-limit, countable-choice, bridge, metric-completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness coordinates. □

Theorem 291.126 (DyadicRatCore terminal tolerance pullback initiality)**.** *Let P be an accepted L10 terminal route whose visible four-face packet exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row, a local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support. Let Q be any accepted L10 terminal route presenting the same $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support. Then Q factors uniquely through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of P if and only if the public tolerance read of Q agrees with the terminal projection of P supplied by Definition 291.125.*

This initiality exports no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read both terminal routes through the same visible support. Apply Definition 291.123 to P and Q; every public tolerance read is forced through the displayed four-face packet and the shared BHist, hsame, Cont, Pkg, and NameCert rows. Apply Definition 291.124 to replace each such read by the finite DyadicRatCore ledger, StreamName window, RegSeqRat readback, local Real seal, and support rows.

For the forward direction, assume Q factors through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of P. The StreamName face is already the same finite section, so Definition 126.586 identifies the two finite-window pullbacks. The RegSeqRat and Real faces are then locked by Definition 301.239 and Definition 38.845. With all four faces fixed, Definition 291.125 gives agreement with the terminal projection of P.

For the reverse direction, assume the public tolerance read of Q agrees with the terminal projection of P. By Definition 291.125, agreement of public tolerance reads over the already shared StreamName, RegSeqRat, Real, and support faces is equivalent to agreement of the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger rows. The finite-support readback theorem identifies the only ledger rows available to Q, so the factorization through the DyadicRatCore ledger of P is unique.

It remains to rule out external witnesses for a second factorization. The exhaustion theorem supplies no fifth source beyond the four visible faces, the finite-support readback supplies no hidden row outside the displayed inventory, the StreamName pullback exactness supplies no quotient stream coordinate, the RegSeqRat-Real terminal exactness supplies no quotient rational or selected-limit coordinate, and the Real terminal pullback coherence supplies no ambient completeness coordinate. Hence quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot mediate another terminal tolerance factorization. □

Theorem 291.127 (DyadicRatCore terminal tolerance cofinal-refinement finality)**.** *Let P be an accepted L10 terminal route whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *face over the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Let R be any accepted cofinal refinement of P preserving the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support, and the public terminal status of the route. Then the public tolerance read of R factors through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger as P.*

Equivalently, no accepted cofinal refinement with those preserved visible faces can introduce a new tolerance coordinate. The finality statement exports no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read R as a terminal route over the preserved visible support. Since the refinement keeps the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback row, the local $\mathsf{Real}^{\uparrow}$ seal row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support, Definition 291.126 applies to P and R.

The StreamName preservation clause identifies the finite-window side of the comparison. Apply Definition 126.586; the refinement has the same finite StreamName section and therefore cannot supply a second window through which a tolerance row could be read.

Next lock the RegSeqRat and Real faces. Theorem 301.239 reads the preserved classifier, readback, and local Real rows as the same four-face terminal surface, and Definition 38.845 rules out an ambient $\mathsf{Real}^{\uparrow}$ completion coordinate.

With all non-dyadic faces fixed, use Definition 291.125. Equality of public tolerance reads is now exactly equality of the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger rows. The pullback initiality theorem then gives the desired factorization of the public tolerance read of R through the ledger of P.

Finally apply the non-export clauses in the exhaustion, projection, and initiality results. Quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not visible tolerance coordinates for the terminal route or for its accepted cofinal refinements. □

Theorem 291.128 (DyadicRatCore terminal tolerance common-refinement confluence)**.** *Let P be an accepted L10 terminal route whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{Real}^{\uparrow}$ *face over the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Let R and S be accepted cofinal refinements of P preserving the $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* BHist*,*

hsame, Cont, Pkg, *and* NameCert *support, and the public terminal status of the route. Suppose R and S admit a common finite* DyadicRatCore$^{\uparrow}$ *scale-window refinement T preserving those same finite-window, classifier, readback, seal, history, sameness, continuity, package, and NameCert rows. Then the public tolerance reads of R and S agree exactly on the same* DyadicRatCore$^{\uparrow}$ *tolerance ledger of P.*

The confluence statement exports no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Apply Definition 291.127 first to R and then to S. Both accepted cofinal refinements preserve the displayed StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows, so each public tolerance read factors through the DyadicRatCore$^{\uparrow}$ tolerance ledger already carried by P.

Now use the common finite scale-window refinement. By Definition 291.99, the two refinement routes through R and S have one consumer-readable finite normal form whose endpoint transport, denominator replay, distance and radius rows, hsame, displayed Cont, Pkg, and local DyadicRatCore naming rows are the common rows read by T.

It remains to identify the public reads with the same ledger coordinate. Apply Definition 291.125; once the StreamName finite window, the RegSeqRat classifier and readback row, the local Real seal row, and the displayed support rows agree, agreement of public tolerance reads is exactly agreement of the DyadicRatCore$^{\uparrow}$ tolerance ledger rows.

Finally apply Definition 291.123 and Definition 291.126. Exhaustion leaves no fifth tolerance source outside the four visible faces and support rows, while pullback initiality prevents a second terminal factorization over those same faces. Hence the common refinement proves confluence on the finite DyadicRatCore ledger and not through quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, or ambient Real$^{\uparrow}$ completeness. □

Theorem 291.129 (DyadicRatCore terminal tolerance facewise support minimality)**.** *Let P be an accepted L10 terminal route whose visible faces are* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and the local* Real$^{\uparrow}$ *face over the same displayed*

BHist, hsame, Cont, Pkg, NameCert

support. Then the terminal tolerance read is facewise support-minimal. Deleting any DyadicRatCore$^{\uparrow}$ *tolerance-ledger row, or deleting any one of the displayed support rows* BHist, hsame, Cont, Pkg, *and* NameCert, *destroys the public terminal tolerance read.*

Moreover, equality of terminal tolerance reads recovers exactly the DyadicRatCore$^{\uparrow}$ *tolerance-ledger rows together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. It recovers no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. First identify the route inventory. By Definition 291.123, every public tolerance read factors through the four visible faces and the displayed support. By Definition 291.124, that factorization is read from the finite DyadicRatCore$^{\uparrow}$ tolerance ledger, the StreamName$^{\uparrow}$ finite-window rows, the RegSeqRat$^{\uparrow}$ classifier and readback rows, the local Real$^{\uparrow}$ seal row, and the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Now delete one listed row. If the deleted row is a DyadicRatCore tolerance-ledger row, the finite readback theorem no longer supplies the finite tolerance inventory. If the deleted row is one

of BHist, hsame, Cont, Pkg, or NameCert, the same theorem no longer supplies the shared support on which all four visible faces are read. In either case the public terminal tolerance read has lost a required finite support coordinate.

Next compare two terminal routes whose public tolerance reads agree. Apply Definition 291.125; over fixed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, and displayed support faces, equality of public tolerance reads is exactly equality of the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-ledger rows. Apply Definition 126.586 to keep the StreamName finite section inside the same support packet.

The RegSeqRat and Real faces do not enlarge the recovered support. Theorem 301.239 locks the classifier, readback, and local Real rows to the same four-face terminal surface, while Definition 38.793 identifies the Real exit certificate by the visible face rows it carries. Thus agreement of terminal tolerance reads recovers the DyadicRatCore ledger and the displayed BHist, hsame, Cont, Pkg, and NameCert support, rather than a hidden completion witness.

Finally apply Definition 291.126 and Definition 291.127. Pullback initiality prevents a second terminal factorization over the same non-dyadic faces, and cofinal-refinement finality prevents an accepted refinement from adding a new tolerance coordinate. Therefore quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not recovered by equality of terminal tolerance reads. □

Theorem 291.130 (DyadicRatCore terminal tolerance support minimality)**.** *Let P be an accepted L10 terminal route whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *face over the same displayed*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support. Then the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and each displayed support row* BHist, hsame, Cont, Pkg, *and* NameCert *are necessary for the public tolerance read of P. Deleting the DyadicRatCore tolerance ledger, or deleting any one of those displayed support rows, destroys the public tolerance read.*

The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *faces cannot replace the deleted row. In particular, quotient streams, selected limits, countable choice, metric completeness, bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness do not supply an alternate public tolerance read.*

Proof. First identify the finite inventory. By Definition 291.124, every public tolerance read of P is determined by the displayed DyadicRatCore signed mantissa, exponent, denominator, comparison, distance, and radius rows, the downstream StreamName, RegSeqRat, and Real faces, and the shared BHist, hsame, Cont, Pkg, and NameCert support. Hence a route that deletes the DyadicRatCore ledger or any one of the displayed support rows no longer contains the finite readback inventory named by the theorem.

Next isolate the dyadic coordinate. Apply Definition 291.125. Once the StreamName finite window, the RegSeqRat classifier and readback row, the local Real seal, and the displayed support are fixed, agreement of public tolerance reads is equivalent to agreement of the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger rows. A packet with no DyadicRatCore ledger therefore cannot present the same public tolerance read through the three downstream faces alone.

Now rule out replacement by refinement. The pullback initiality theorem Definition 291.126 says that any accepted terminal route over the same non-dyadic faces factors uniquely through the DyadicRatCore ledger exactly when its public tolerance read agrees with the terminal projection. The cofinal-refinement finality theorem Definition 291.127 then prevents an accepted cofinal

refinement from adding a new tolerance coordinate after the visible faces and support rows have been preserved.

Finally exclude the three downstream faces as substitutes for the missing row. The StreamName exactness rows Definition 126.471 and Definition 126.472 keep the open-phase ledger inside the same finite four-face packet, not in quotient stream equality or selected limits. The RegSeqRat non-escape row Definition 301.169 confines readback to the displayed dyadic observation rows and their transported support, not to countable choice or a separate metric-completeness row. The Real boundary theorem Definition 38.464 determines the local Real seal from the fused finite packet and supplies no bridge row or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Thus deleting the DyadicRatCore ledger, BHist, hsame, Cont, Pkg, or NameCert destroys the public tolerance read rather than moving it to another face. □

Theorem 291.131 (DyadicRatCore terminal tolerance finite-cover descent)**.** *Let $\{P_i\}_{i\in I}$ be a finite cover of an accepted L10 terminal route by accepted terminal-route members whose visible faces are* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *face. Suppose every member and every nonempty overlap preserves the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

support. Then the descended public tolerance read on the covered route is determined exactly by the descended $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

If the descended DyadicRatCore tolerance ledger is deleted, or if two cover members or an overlap disagree on that ledger while the displayed support rows remain fixed, the public tolerance read does not descend. The obstruction localizes to a cover member missing the ledger or to an overlap on which the two local ledgers disagree. No quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness supplies a substitute descent datum.*

Proof. First apply Definition 126.669. The finite cover has a single descent support exactly when the finite StreamName window and the displayed BHist, hsame, Cont, Pkg, and NameCert rows agree on every cover member and on every nonempty overlap. Thus the cover-level support packet is a finite packet, not a quotient stream or a selected limiting object.

Next apply Definition 291.127 on each member and on each accepted overlap route. Since the visible StreamName, RegSeqRat, local Real, history, sameness, continuity, package, and NameCert rows are preserved, no accepted cofinal refinement can add an extra terminal tolerance coordinate beyond the DyadicRatCore ledger already carried by the member.

The local public reads therefore compare through the DyadicRatCore ledger. By Definition 301.239, the RegSeqRat classifier, RegSeqRat readback, and local Real seal are locked to the same four-face terminal surface. They transport the local route rows across the cover, but they do not create a second tolerance source.

It remains to identify the obstruction when descent fails. Apply Definition 291.130. Deleting the DyadicRatCore tolerance ledger on a member deletes a necessary support coordinate for that member's public read. If two members carry different DyadicRatCore ledgers on an overlap, the same minimality statement and the cofinal-refinement finality statement prevent the disagreement from being hidden in the StreamName, RegSeqRat, Real, or shared support rows.

Consequently the only descended public tolerance read compatible with all cover members is the one read from the descended DyadicRatCore tolerance ledger. The failure cases are precisely

memberwise absence of that ledger or overlapwise disagreement of that ledger; quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the terminal descent datum. □

Theorem 291.132 (DyadicRatCore terminal four-face normal-form uniqueness)**.** *Let P and Q be accepted L10 terminal routes. Suppose their public terminal packets expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support. Then the public terminal four-face normal form of P is identical to the public terminal four-face normal form of Q.

Conversely, equality of the public terminal four-face normal forms of P and Q recovers exactly those rows: the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The recovery exports no quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read both routes through the visible terminal inventory. Apply Definition 291.123; every public tolerance coordinate of either route is forced through the DyadicRatCore ledger, the StreamName finite window, the RegSeqRat classifier and readback rows, the local Real seal, and the displayed support.

Next project the public tolerance reads. By Definition 291.125, once the StreamName, RegSeqRat, Real, and support faces are fixed, equality of public terminal tolerance reads is exactly equality of the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger rows. The assumed agreement of all displayed rows therefore identifies the four-face terminal packets of P and Q.

The terminal comparison is unique over that packet. Apply Definition 291.126; any accepted terminal route with the same non-dyadic faces and the same displayed support factors uniquely through the same DyadicRatCore tolerance ledger. Hence the two public terminal four-face normal forms coincide.

For the converse, assume the public terminal four-face normal forms coincide. The finite-cover descent theorem Definition 291.131 says that a descended public tolerance read is determined exactly by the descended DyadicRatCore ledger and that absence or overlapwise disagreement of that ledger is the precise obstruction to descent. Thus equality of normal forms recovers the DyadicRatCore tolerance ledger.

The remaining faces are then read back from the same public packet. Use Definition 291.123 and Definition 291.128; exhaustion leaves no fifth source outside the visible faces and support rows, while confluence identifies the common finite route through the StreamName window, RegSeqRat classifier and readback, local Real seal, and displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply the non-export clauses from the exhaustion, projection, initiality, confluence, and finite-cover descent results. Quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate formalstatus rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not terminal normal-form coordinates. The equality therefore recovers exactly the listed rows and no hidden completion, choice, bridge, or aggregate-status data. □

Theorem 291.133 (DyadicRatCore terminal four-face support)**.** *An accepted L10 terminal route has a public tolerance read exactly when it displays the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Deleting any one of those four faces or any displayed support row prevents the public tolerance read.

No quotient rational equality, quotient stream equality, selected limit, countable-choice row, metric-completeness row, bridge row, aggregate formalstatus row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness can replace a missing face or support row.*

Proof. First combine Definition 291.123 with Definition 291.124. These theorems identify every public tolerance read with the DyadicRatCore ledger, the StreamName finite window, the RegSeqRat readback surface, the local Real seal, and the displayed support rows.

For sufficiency, use Definition 291.132. Once the four faces and support rows are present, the terminal normal form is fixed, so the public tolerance read is the one read from that finite packet.

For necessity, delete one listed face or support row. If the deleted row is part of the DyadicRatCore ledger or the shared support, Definition 291.130 blocks the read. If the deleted row is the StreamName, RegSeqRat, or Real face, Definition 126.471, Definition 301.239, and Definition 38.464 remove the corresponding downstream route.

Finally apply the projection and minimality clauses, Definition 291.125 and Definition 291.129. They leave no replacement coordinate outside the displayed finite packet. Hence quotient rational equality, quotient stream equality, selected limits, countable choice, metric completeness, bridge rows, aggregate status rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot restore the public read after a face or support row is absent. □

Theorem 291.134 (DyadicRatCore tolerance-halving cover)**.** *For every positive dyadic tolerance row accepted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite scale-window ledger, there is a halved positive dyadic tolerance row in a displayed finite refinement window whose doubled radius is classifier-below the original tolerance row. The refinement preserves the signed mantissa row, denominator ledger, common-exponent comparison, distance and radius rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

Consequently every $\mathsf{StreamName}^{\uparrow}$ *finite-window request and every* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity check that consumes a DyadicRatCore tolerance row may replace it by a carried half-tolerance row without invoking host rationals, quotient equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted positive tolerance row in the finite scale-window packet. Apply scale-window monotonicity, Definition 291.41, to pass to the displayed successor exponent window while keeping the same signed mantissa, denominator ledger, comparison row, radius row, hsame, Cont, Pkg, and local NameCert support.

Next use the common-scale triangle ledger, Definition 291.82. It compares the original radius row and the successor-window radius row inside one common finite denominator scale, so the doubled successor radius is classifier-below the original tolerance rather than compared by host rational order.

Finally apply the terminal four-face tolerance exhaustion theorem, Definition 291.123. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces have no fifth tolerance source outside the displayed DyadicRatCore ledger and support rows. The halved row is therefore the finite-window handoff

consumed by StreamName requests and RegSeqRat regularity checks, with the stated transport and nonescape clauses preserved. □

Theorem 291.135 (DyadicRatCore halving-cover RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity check consuming a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row may replace that row by the halved finite refinement supplied by Definition 291.134. The replacement preserves the* $\mathsf{StreamName}^{\uparrow}$ *request rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *support.*

The refined read is still a finite regularity read over the displayed DyadicRatCore ledger. It does not invoke host rational order, quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 291.134 to the positive tolerance row consumed by the regularity check. It supplies the halved positive row in a displayed finite refinement window while preserving the dyadic ledger, transport, replay, provenance, and local naming rows.

Then use the RegSeqRat common-window compatibility supplied by Definition 301.36. The original regularity read and the halved refinement are compared on one accepted common finite window, so the $\mathsf{StreamName}^{\uparrow}$ request rows and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows are unchanged as public source data.

Finally the terminal four-face tolerance exhaustion from Definition 291.134 rules out a second tolerance source. The RegSeqRat check therefore consumes the halved row through the same finite BEDC support rather than through host order, quotient streams, selected limits, choice, host equality, or ambient completeness. □

Theorem 291.136 (DyadicRatCore halving-cover StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *finite-window request consuming a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row may replace that row by the halved finite refinement supplied by Definition 291.134. The replacement preserves the finite window schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *support.*

The refined StreamName read remains a finite-window read over the displayed DyadicRatCore ledger. It does not invoke host rational order, quotient streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a separate completion coordinate.*

Proof. Apply Definition 291.134 to the positive tolerance row consumed by the StreamName request. It supplies a halved positive dyadic row in a displayed finite refinement window while preserving the DyadicRatCore comparison ledger, transport, replay, provenance, and local naming rows.

Next read the StreamName finite-window request rows. The request schedule is indexed by displayed windows and tolerance demands, so replacing the tolerance row by its carried half-tolerance refinement changes no public schedule face and adds no host rational comparison.

The same support rows transport the replacement through hsame, replay it through Cont, source it through Pkg, and name it through the local NameCert. Thus the StreamName handoff consumes the halved row through the finite BEDC support.

Finally the nonescape clause in Definition 291.134 rules out host rational order, quotient streams, selected limits, choice, host equality, ambient Real completeness, and separate completion data as alternative tolerance sources. □

Theorem 291.137 (DyadicRatCore common-exponent transport)**.** *The halved tolerance refinement supplied by Definition 291.134 preserves the common-exponent comparison ledger under componentwise* hsame *transport and displayed* Cont *replay. If a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity check or a*

$\mathsf{StreamName}^{\uparrow}$ *finite-window request consumes the original positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, then replacing that row by the carried half-tolerance row keeps the signed mantissa, denominator scale, doubled-radius comparison, and common-exponent classifier in the same finite BEDC ledger.*

The comparison is BEDC classifier data: it is replayed through displayed common-exponent rows, not reopened as host rational order, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the positive tolerance row and its halved refinement from Definition 291.134. The refinement already carries the signed mantissa row, denominator ledger, common-exponent comparison, transport, replay, provenance, and local naming rows.

Use Definition 291.41 to move the read to the displayed successor scale window without changing the public dyadic components. Then apply Definition 291.82; it keeps the original row, the halved row, and the doubled-radius comparison inside one common finite exponent ledger.

The downstream handoff is stable on both consuming faces. Definition 291.135 carries the same ledger to $\mathsf{RegSeqRat}^{\uparrow}$ regularity checks, while Definition 291.136 carries it to $\mathsf{StreamName}^{\uparrow}$ finite-window requests. Componentwise hsame transport and displayed Cont replay therefore repeat the common-exponent classifier rather than creating a second comparison source.

Since every comparison row used by the replacement is one of the displayed DyadicRatCore classifier rows, the construction never appeals to host rational order, quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness. □

Theorem 291.138 (DyadicRatCore iterated halving budget)**.** *For every positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row and every finite unary schedule of halving requests, there is a displayed finite budget whose final row refines the original tolerance row and whose every predecessor doubles classifier-below the previous row. The budget preserves the signed mantissa row, denominator ledger, common-exponent comparison, distance and radius rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support at each displayed step.*

Every $\mathsf{StreamName}^{\uparrow}$ *finite-window request and every* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity check may consume this iterated budget through the same finite dyadic ledger. The construction does not invoke host rational order, quotient stream equality, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a separate completion coordinate.*

Proof. Induct over the finite unary schedule. In the empty case, the budget is the original positive tolerance row together with its displayed signed mantissa, denominator, common-exponent, distance, radius, transport, replay, provenance, and local naming rows.

For the successor step, apply Definition 291.134 to the current positive tolerance row. It supplies the next halved row in a finite refinement window and records that doubling the new row is classifier-below the previous row inside the DyadicRatCore comparison ledger.

Use Definition 291.135 and Definition 291.136 to transport the accumulated budget to RegSeqRat regularity checks and StreamName finite-window requests. The induction preserves only displayed dyadic rows and support data, so no host rational order, quotient stream equality, selected limit, choice, host equality, ambient Real completeness, or completion coordinate is introduced. □

Theorem 291.139 (DyadicRatCore halving-budget cofinality)**.** *For any finite family of* $\mathsf{StreamName}^{\uparrow}$ *finite-window requests and* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity checks consuming a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the iterated halving budget of Definition 291.138 supplies a common cofinal tolerance window. The final budget row preserves the signed mantissa, denominator ledger, common-exponent comparison, distance and radius rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

Every listed consumer factors through that common row by the same finite dyadic ledger. The cofinal window does not use host rational order, quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a separate completion coordinate.*

Proof. Enumerate the finite family of StreamName requests and RegSeqRat checks. Choose the unary halving schedule long enough to meet the largest displayed tolerance demand among those finite rows, and apply Definition 291.138 to the original positive tolerance row.

The resulting budget carries each predecessor and the final row inside one displayed dyadic ledger. Signed mantissas, denominator rows, common-exponent comparisons, distance and radius rows, transport, replay, provenance, and local naming are preserved at every step of the schedule.

Next apply Definition 291.136 to the StreamName requests and Definition 291.135 to the RegSeqRat checks. These handoffs let each consumer replace its original tolerance by the carried final row while keeping the same finite support rows.

Finally use Definition 291.137. It keeps all comparisons in the common-exponent ledger, so the common row is cofinal for the finite family without reopening the comparison as host rational order or importing quotient streams, selected limits, choice, host equality, ambient Real completeness, or completion data. □

Theorem 291.140 (DyadicRatCore terminal halving-cover selection-free budget)**.** *For any finite family of* $\mathsf{StreamName}^{\uparrow}$ *finite-window requests and* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity checks consuming a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, there is one displayed terminal halving-cover budget over the common-exponent ledger through which every listed consumer factors. The terminal budget preserves the signed mantissa row, denominator ledger, common-exponent comparison, doubled-radius comparisons, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

The budget is selection-free: it does not use host rational order, quotient stream equality, selected limits, countable choice, host $\mathsf{Real}^{\uparrow}$ *equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden completion coordinate.*

Proof. Start from the finite family of StreamName requests and RegSeqRat regularity checks. Apply the cofinality theorem Definition 291.139 to obtain a common final tolerance row for the whole displayed family.

The cofinal row is built from iterated halving closure. By Definition 291.138, each predecessor and the terminal row remain inside the same finite DyadicRatCore ledger, with the signed mantissa, denominator scale, common-exponent comparison, doubled-radius comparisons, transport, replay, provenance, and local naming rows preserved.

The two consumer faces are then routed through the existing handoffs. Use Definition 291.136 for finite-window requests and Definition 291.135 for regularity checks.

Finally apply Definition 291.137. It keeps every tolerance comparison in the displayed common-exponent ledger, so the terminal budget is one finite BEDC object rather than a selected limiting tolerance. The construction therefore refuses host rational order, quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, and completion coordinates. □

Theorem 291.141 (DyadicRatCore terminal halving-cover exhaustion)**.** *Every terminal four-face L10 tolerance read using* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *factors through the selection-free terminal halving-cover budget of Definition 291.140. No such consumer can introduce a separate tolerance source, host rational order, quotient stream equality, selected limit, countable choice, host* $\mathsf{Real}^{\uparrow}$ *equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

The exhaustion route is the finite ledger

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the common-exponent transport and the two halving-cover handoffs. The terminal read therefore sees only the displayed dyadic budget, its common-exponent comparisons, the finite-window consumer face, the regular rational consumer face, and the downstream Real seal.

Proof. Start with a terminal four-face tolerance read. Apply Definition 291.140 to obtain one displayed terminal budget through which the finite-window and regularity consumers factor.

Use Definition 291.137 on that budget. It keeps every comparison in the same common-exponent ledger, so the terminal row is not replaced by host rational order or by an independent tolerance source.

Route the consumer faces through the two handoff theorems. The StreamName face uses Definition 291.136, and the RegSeqRat face uses Definition 291.135; the Real-facing read is downstream of those finite tolerance rows and cannot bypass them.

The accepted packet has no row for quotient stream equality, selected limits, countable choice, host Real equality, or ambient completeness. Hence every terminal four-face L10 read exhausts the available tolerance data at the selection-free halving-cover budget. □

Theorem 291.142 (DyadicRatCore halving-cover Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *terminal seal read consuming a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row factors through the terminal halving-cover budget before the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *faces are sealed. The handoff preserves the common-exponent transport, the finite StreamName request window, the regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

The Real-facing handoff introduces no host rational order, quotient stream equality, selected limit, countable-choice schedule, host $\mathsf{Real}^{\uparrow}$ *equality, ambient completeness coordinate, or separate completion coordinate.*

Proof. Start with a Real terminal seal read that consumes a DyadicRatCore tolerance row. Apply terminal halving-cover exhaustion, Definition 291.141, to force the terminal four-face read through the selection-free halving-cover budget.

Use common-exponent transport, Definition 291.137, on that budget. The tolerance row, its halved refinements, and the doubled-radius comparisons remain inside one displayed DyadicRatCore ledger before any StreamName, RegSeqRat, or Real seal face is exposed.

Route the two analytic faces through their handoffs. The finite StreamName request face is supplied by Definition 291.136, and the regular rational face is supplied by Definition 291.135.

Only after those two faces are sealed does the Real terminal read consume the tolerance row. Since each cited route preserves the displayed support rows and has no coordinate for host rational order, quotient stream equality, selected limits, countable choice, host Real equality, ambient completeness, or separate completion coordinates, the Real-facing handoff has none. □

The tolerance-halving cover sequence displayed by Definition 291.141 is the local public-boundary witness consumed by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ terminal tolerance reads: later L10 routes cite this finite ledger, not a new top-level closure claim.

Theorem 291.143 (DyadicRatCore tolerance-halving cover composition)**.** *Let two displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *halving-cover packets occupy adjacent dyadic tolerance windows, with the terminal row of the first packet serving as the positive tolerance row consumed by the second. Then the two packets compose into a single displayed halving-cover packet whose common-exponent transport,* $\mathsf{StreamName}^{\uparrow}$

handoff, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal budget, and nonselection clauses are inherited from the two inputs. The composite packet refines the original tolerance by the concatenated finite halving schedule and still exposes only signed mantissa rows, denominator ledgers, doubled-radius comparisons, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

The composed cover does not select a radius, reopen host rational order, identify quotient streams, choose limits, use countable choice, assert host $\mathsf{Real}^{\uparrow}$ *equality, import ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or add a separate completion coordinate.*

Proof. Start with the two adjacent halving-cover packets. Apply common-exponent transport, Definition 291.137, to both packets so the terminal row of the first and the initial row of the second are read inside the same finite comparison ledger.

Concatenate the two finite halving schedules. The iterated halving budget theorem, Definition 291.138, packages that concatenation as one displayed budget, preserving signed mantissas, denominator rows, doubled-radius comparisons, transport, replay, provenance, and local naming at each step.

Use halving-budget cofinality, Definition 291.139, to route the finite StreamName requests and RegSeqRat regularity checks through the final row of the composite budget. The StreamName and RegSeqRat handoff clauses are therefore inherited from the two input covers through the same finite support rows.

Finally apply terminal halving-cover exhaustion, Definition 291.141. It rules out a second tolerance source at the terminal four-face read, so the composite cover remains selection-free and does not reopen host order, quotient streams, selected limits, choice, host equality, ambient Real completeness, or completion coordinates. □

Theorem 291.144 (DyadicRatCore common-scale halving-cover factorization)**.** *Every composed* $\mathsf{DyadicRatCore}^{\uparrow}$ *halving-cover packet factors through the common exponent scale before it is consumed by a* $\mathsf{StreamName}^{\uparrow}$ *finite-window request or a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity check. More precisely, after two adjacent halving-cover packets are composed, the resulting tolerance route is first read in the common-exponent ledger and only then handed to the StreamName and RegSeqRat faces. The factorization preserves signed mantissa rows, denominator ledgers, doubled-radius comparisons, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRatCore}^{\uparrow}}$ *support.*

The common-scale factorization does not select a radius, reopen host rational order, identify quotient streams, choose limits, use countable choice, assert host $\mathsf{Real}^{\uparrow}$ *equality, import ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or add a separate completion coordinate.*

Proof. Start from the composed halving-cover packet of Definition 291.143. Its terminal row and every predecessor row are inherited from the adjacent covers with the same finite dyadic support.

Apply common-exponent transport, Definition 291.137, to the composite packet. This places the original tolerance, the halved rows, and the doubled-radius comparisons in one displayed exponent ledger before any analytic consumer can read them.

The consumer faces then factor through the same ledger. StreamName finite-window requests use Definition 291.136, while RegSeqRat regularity checks use Definition 291.135. Both handoffs therefore consume the common-scale row rather than a separate tolerance source.

Since the composition and the two handoffs all preserve only the displayed dyadic rows, transport, replay, provenance, and local naming data, the factorization exports no host rational order, quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or completion coordinate. □

Chapter 292

A Concrete Naming Certificate for $\mathsf{CauchyModulus}^{\uparrow}$

The $\mathsf{CauchyModulus}^{\uparrow}$ interface records the explicit rate data used by the Bishop-style regular-sequence construction of $\mathsf{RegSeqRat}$ and $\mathsf{Real}^{\uparrow}$. It sits over $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$: natural-number rows index observations, rational tolerance rows state requested bounds, stream-name rows identify the scheduled observations, and a finite ledger witnesses that the displayed modulus bound has been consumed by the downstream regular-sequence and real-number carriers.

The finite threshold ledger is the upstream rate surface consumed by chapter 641. That route reads the tolerance request, finite budget, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}$ schedule row, dyadic comparison ledger, and $\mathsf{Real}^{\uparrow}$ handoff only after the modulus packet has exposed its $\mathsf{Nat}^{\uparrow}$ threshold, $\mathsf{Rat}^{\uparrow}$ tolerance, $\mathsf{StreamName}^{\uparrow}$ schedule, finite Cont window, and Pkg provenance rows.

292.1 Carrier and classifier

Definition 292.1 (Cauchy-modulus carrier)**.** *A* $\mathsf{CauchyModulus}^{\uparrow}$ *carrier packet is a finite* BHist *ledger with four visible groups of rows.*

1. *Unary index rows supplied by* $\mathsf{Nat}^{\uparrow}$*, including the requested precision index and the threshold index returned by the modulus.*
2. *Dyadic or rational tolerance rows supplied by* $\mathsf{Rat}^{\uparrow}$*, including the positive tolerance budget and the displayed comparison to the tolerance read by the regular-sequence carrier.*
3. *Scheduled observation rows supplied by* $\mathsf{StreamName}^{\uparrow}$*, naming the stream entries whose late pairwise distance is being bounded.*
4. *A finite consumption ledger stating that every displayed observation pair past the threshold uses the same modulus row, the same tolerance row, and the same* Pkg *provenance before it is passed to* $\mathsf{RegSeqRat}$ *or* $\mathsf{Real}^{\uparrow}$*.*

The carrier accepts only packets in which the index, tolerance, schedule, and consumption rows are present in the same finite history and are connected by displayed Cont *routes. It has no quotient of streams, no ambient limit object, and no hidden choice of a threshold outside the ledger.*

Definition 292.2 (Cauchy-modulus classifier)**.** *The* $\mathsf{CauchyModulus}^{\uparrow}$ *classifier compares two accepted packets by componentwise* hsame *on their precision-index rows, threshold-index rows, tolerance rows, scheduled observation rows, consumption-ledger rows,* Cont *routes, and* Pkg *provenance.*

Two packets classify together only when the same visible modulus schedule can be replayed through the transported finite ledger.

Lean4 (definition) `BEDC.Derived.CauchyModulusUp.CauchyModulusClassifier`

Theorem 292.3 (Cauchy-modulus hsame stability)**.** *Let P and Q be accepted* CauchyModulus$^{\uparrow}$ *packets. If every index, tolerance, scheduled-observation, consumption-ledger,* Cont*, and* Pkg *row of P is transported to the corresponding row of Q by displayed* hsame *comparisons, then Q is accepted by the carrier of Definition 292.1 and the classifier of Definition 292.2 identifies the two packets.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket hsame stability`
Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusCarrier hsame stability`

Proof. Project P to the four carrier groups. The supplied hsame rows transport the natural indices and rational tolerance rows first, so the threshold still answers the same displayed precision request.

Transport the scheduled observation rows through the StreamName$^{\uparrow}$ surface. The finite consumption ledger then replays the same Cont routes from the transported threshold and tolerance rows to the transported late-observation claims.

Finally repack the transported rows with the transported Pkg provenance. These are exactly the fields required by the carrier and by the componentwise classifier, so Q is accepted and the two packets classify together. □

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusLedgerPacket hsame stability`

Theorem 292.4 (Cauchy-modulus finite-window closure)**.** *For every accepted* CauchyModulus$^{\uparrow}$ *packet and every displayed precision request in its* Nat$^{\uparrow}$ *index rows, the packet exposes a finite* Cont *window containing the threshold row, the rational tolerance row, the scheduled late observation rows, and the consumption ledger read by* RegSeqRat *and* Real$^{\uparrow}$*. No consumer may read a convergence rate except through that finite window.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusContWindow closure`
Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket cont window closure`

Proof. Inspect the accepted carrier packet at the requested precision row. The threshold is present as a visible Nat$^{\uparrow}$ row, and the tolerance budget is present as a visible Rat$^{\uparrow}$ row in the same finite history.

The scheduled observation rows are linked to those two rows by the displayed Cont routes. Restrict the packet to the rows mentioned by those routes and to the corresponding consumption-ledger entries.

This restricted packet is finite and contains every row used by the downstream regular-sequence and real-number readbacks. Since the carrier has no other constructor for rate data, every convergence-rate read factors through the finite window just described. □

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusCarrier finite cont window closure`

Theorem 292.5 (Cauchy-modulus finite-window composition handoff)**.** *Suppose two accepted* CauchyModulus$^{\uparrow}$ *precision windows are composable through the displayed* Cont *routes of their carrier packets. Then their composition exposes a single finite handoff packet whose rows are exactly the* Nat$^{\uparrow}$ *precision and threshold rows, the* Rat$^{\uparrow}$ *tolerance rows, the* StreamName$^{\uparrow}$ *scheduled observation rows, the* RegSeqRat *late-pair evidence rows, the* Real$^{\uparrow}$ *classifier transport rows, the displayed* hsame *comparisons, and the* Pkg *provenance rows already present in the two windows. The handoff packet introduces no selected limit, quotient stream, or hidden threshold selector outside those rows.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket window composition handoff`

Proof. Project the two accepted windows by Definition 292.4. Each projection is finite and contains its displayed precision request, threshold row, tolerance row, scheduled observations, and consumption ledger.

Compose the two Cont ledgers along their shared carrier rows. The resulting ledger has only the rows named by the projections: the natural-number precision and threshold rows, rational tolerance rows, scheduled observation rows, regular-sequence late-pair evidence, real-classifier transport rows, hsame comparisons, and Pkg provenance.

Finally apply Definition 292.3 to transport the shared classifier comparisons across the composed packet. Since Definition 292.4 already exhausts every consumer-visible rate read, the composed packet has no constructor for a selected limit, quotient stream, or threshold selector beyond the displayed finite ledger. □

Theorem 292.6 (Monotone tail obligation)**.** *Let M be an accepted* CauchyModulus$^{\uparrow}$ *carrier packet. If a tolerance row is transported to a smaller positive dyadic tolerance through the displayed comparison ledger, then the threshold selected by the modulus row can be transported to a threshold that works for the smaller tolerance, and the late-pair bound rows remain accepted after* hsame *transport of the observation indices.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusTailWindow smaller tolerance transport`
Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulus monotone tail refinement window`

Proof. Unpack the carrier packet of Definition 292.1. The selected threshold and every late-pair bound are visible rows of the same finite BHist ledger, so decreasing the tolerance asks only for a refined dyadic or rational comparison row, not for a new stream object.

Transport the observation endpoints by the displayed hsame rows and compose the old bound with the smaller-tolerance comparison. The provenance ledger supplies the Cont and Pkg rows needed to repack the same late-pair evidence under the transported threshold. □

Theorem 292.7 (RegSeqRat regularity obligation)**.** *Every accepted* CauchyModulus$^{\uparrow}$ *carrier packet whose source stream is a* RegSeqRat *regular rational sequence supplies the explicit regularity data required by the* RegSeqRat *phase: for displayed indices m, n past the threshold selected by the modulus row, the bound ledger gives the estimate required by the regular-sequence classifier, and this estimate is stable under componentwise* hsame *transport of m, n, and the scheduled observations.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusRegularityWindow hsame transport`
Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket regseqrat regularity handoff`

Proof. Read the RegSeqRat regularity row as a request for a visible late-pair estimate. The carrier already stores exactly that estimate in the consumption ledger, indexed by the threshold and by the two displayed stream observations.

For stability, transport the two indices and observations through the classifier of Definition 292.2. The comparison row is part of the ledger, so the transported packet again contains the regularity estimate demanded by RegSeqRat. □

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusLedgerPacket regseqrat regularity rows`

Theorem 292.8 (Real classifier transport obligation)**.** *Assume the* $\mathsf{Real}^{\uparrow}$ *classifier is read from* RegSeqRat *representatives by rational tolerance tests. A* $\mathsf{CauchyModulus}^{\uparrow}$ *carrier packet transports across the* $\mathsf{Real}^{\uparrow}$ *classifier by sending its stream observations through the* RegSeqRat *classifier, transporting the modulus threshold rows by* hsame*, and reusing the tail-bound ledger for the transported representatives.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusLedgerPacket real classifier transport`

Proof. The only analytic datum used by the real classifier is an explicit tolerance test on sufficiently late representatives. The $\mathsf{CauchyModulus}^{\uparrow}$ packet exposes this datum as a threshold row together with the late-pair ledger.

Apply the RegSeqRat transport to the representative stream and then repack the threshold, comparison, and provenance rows. Since no equivalence class of real numbers is formed, the transport is a row-by-row certificate operation inside the displayed BHist packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusCarrierPacket real classifier transport`

Theorem 292.9 (Cauchy-modulus scoped dependency certificate)**.** *The carrier, classifier,* hsame *stability, finite* Cont *window closure, monotone tail obligation,* RegSeqRat *regularity handoff, and* $\mathsf{Real}^{\uparrow}$ *classifier transport theorem bind every exported* $\mathsf{CauchyModulus}^{\uparrow}$ *row to the following dependency stack:* $\mathsf{Nat}^{\uparrow}$ *index rows,* $\mathsf{Rat}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *scheduled observations,* RegSeqRat *regularity rows,* $\mathsf{Real}^{\uparrow}$ *classifier transport,* hsame *transport, finite* Cont *windows, and* Pkg *provenance. No exported row uses a selected limit, a stream quotient, or an unrecorded threshold selector.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusCarrier dependency boundary`

Proof. Start with the carrier and classifier of Definition 292.1 and Definition 292.2. They list the visible precision, tolerance, scheduled-observation, comparison, Cont, and Pkg rows, so every consumer-visible coordinate begins as a displayed finite BHist entry.

Next project the transport rows in order. The hsame stability theorem keeps the classifier componentwise, the finite Cont window theorem bounds the readback window, the monotone tail theorem records stronger-tolerance reuse, and the RegSeqRat handoff reads the regularity estimate from the same late pair ledger.

Finally apply the $\mathsf{Real}^{\uparrow}$ classifier transport theorem. It reuses the representative-stream, threshold, comparison, and provenance rows already named above, so the scoped dependency certificate closes inside the displayed dependency stack and introduces no selected limit, stream quotient, or hidden threshold choice. □

Theorem 292.10 (Cauchy-modulus NameCert obligation surface)**.** *The* $\mathsf{CauchyModulus}^{\uparrow}$ *carrier and classifier of Definition 292.1 and Definition 292.2 supply the paper-side obligation surface for an eventual* $\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$*. The source rows are the displayed* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *packets; the classifier is componentwise* hsame *transport of the finite modulus ledger; the ledger exactness condition exports only the threshold, tolerance, scheduled observation,* Cont*, and* Pkg *rows consumed by* RegSeqRat *and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket namecert obligation surface`

Proof. The carrier definition names the source rows and the finite consumption ledger. The classifier definition states exactly how those rows move under componentwise hsame transport.

Apply Definition 292.3 for classifier stability and Definition 292.4 for the finite read window. Use Definition 292.6, Definition 292.7, and Definition 292.8 to identify the visible tail-control

rows consumed by stronger tolerances, regular-sequence readback, and real-classifier transport. Together they expose the naming-certificate fields at the paper surface without adding hidden limits or unrecorded threshold choices. □

Theorem 292.11 (Cauchy-modulus public consumer boundary)**.** *The public* $\mathsf{CauchyModulus}^{\uparrow}$ *consumer boundary is exactly the finite* BHist *package consisting of the carrier rows of Definition 292.1, the classifier rows of Definition 292.2, the* hsame *stability rows, the finite* Cont *window-closure rows, monotone tail transport, the* RegSeqRat *regularity handoff,* $\mathsf{Real}^{\uparrow}$ *classifier transport, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$ *obligation surface. Thus any* RegSeqRat *consumer or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer reads only visible* $\mathsf{Nat}^{\uparrow}$ *precision rows,* $\mathsf{Rat}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *scheduled observation rows,* RegSeqRat *late-pair rows,* $\mathsf{Real}^{\uparrow}$ *classifier-transport rows,* hsame *rows,* Cont *rows, and* Pkg *provenance. It reads no selected limit, stream quotient, host index equality, or completion row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket public consumer boundary`

Proof. Start with the carrier and classifier of Definition 292.1 and Definition 292.2. These rows display the precision, tolerance, scheduled observation, comparison, continuation, and provenance coordinates available to a consumer.

Apply Definition 292.3 to keep the classifier rowwise, and apply Definition 292.4 to bound the finite readback window. Use Definition 292.6 for stronger-tolerance transport, and use Definition 292.7 for the regular-sequence late-pair handoff.

Now apply Definition 292.8 to identify the finite real-classifier transport rows. The dependency stack is then the one certified by Definition 292.9 and exposed by Definition 292.10.

Finite ledger exhaustion gives the boundary. Every row named in the statement is a displayed BHist coordinate or a displayed transport of one, and the carrier has no coordinate for a selected limit, a stream quotient, a host index equality, or a completion row. □

Theorem 292.12 (Cauchy-modulus refinement consumer boundary)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *consumer read whose shared-meet row, selector-budget row, dyadic tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row, and* $\mathsf{Real}^{\uparrow}$ *seal row are admitted factors through an accepted* $\mathsf{CauchyModulus}^{\uparrow}$ *finite-window packet of Definition 292.4 and through the public consumer boundary of Definition 292.11. The read exports only the visible* $\mathsf{Nat}^{\uparrow}$ *precision and threshold rows,* $\mathsf{Rat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *scheduled observation rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *late-pair schedule rows,* $\mathsf{Real}^{\uparrow}$ *seal-transport rows,* hsame *rows,* Cont *rows, and* Pkg *provenance consumed by the downstream carrier of chapter 885. It exports no quotient stream, selected limit, host real equality, countable-choice row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusCarrier refinement consumer boundary`

Proof. Begin with the carrier rows of Definition 292.1. They display the precision request, threshold, tolerance, scheduled observations, finite consumption ledger, continuation routes, and provenance needed by any rate consumer.

Apply the classifier of Definition 292.2 to keep those rows componentwise under hsame transport. Then use Definition 292.4 to restrict the accepted read to the finite window containing the threshold, tolerance, scheduled observation, and regularity rows.

Next apply Definition 292.11. It identifies the exact public package seen by $\mathsf{RegSeqRat}^{\uparrow}$ and finite-window $\mathsf{Real}^{\uparrow}$ consumers, including the dyadic tolerance, regular readback, real seal transport, continuation, and provenance rows.

Finally read the refinement chapter as a downstream consumer of that package: chapter 885 admits its shared meet, selector budget, dyadic tolerance, regular schedule, and real seal rows only after this public finite-window route is present. Since the upstream package has no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness, neither can the accepted refinement consumer read one from the $\mathsf{CauchyModulus}^{\uparrow}$ side. □

Theorem 292.13 (CauchyModulus explicit-modulus standard bridge)**.** *The public consumer boundary of Definition 292.11 is exactly the standard constructive explicit-modulus reading of a regular rational name. It consists of a visible precision row, a visible tolerance row, a finite threshold row, a scheduled observation window, a late-pair comparison ledger, and provenance through* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Under this reading, the $\mathsf{CauchyModulus}^{\uparrow}$ *packet exports no selected limit, hidden threshold choice, stream quotient, host index equality, or completion row.*

Proof. Project the carrier and classifier rows of Definition 292.1 and Definition 292.2. These rows display the precision, tolerance, threshold, scheduled observation, late-pair comparison, transport, and provenance coordinates used by the public consumer boundary.

Apply the monotone-tail obligation of Definition 292.6 and the regularity obligation of Definition 292.7. They identify the late-index threshold and the late-pair comparison ledger as finite rows, not as hidden choices or quotient data.

Use Definition 292.11 to collect the same rows into the public $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer surface. The source certificates through $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are already displayed in that boundary.

Repack this finite ledger as the standard explicit-rate schema. Since every exported coordinate is a visible row of the original boundary, there is no coordinate from which to read a selected limit, hidden threshold choice, stream quotient, host index equality, or completion row. □

Lean4 ✓ `BEDC.Derived.CauchyModulusUp.CauchyModulusUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulus}^{\uparrow}$ packet is generated from visible $\mathsf{Nat}^{\uparrow}$ precision and threshold rows, visible $\mathsf{Rat}^{\uparrow}$ tolerance rows, scheduled $\mathsf{StreamName}^{\uparrow}$ observation rows, and a finite consumption ledger showing how the displayed bound is read by RegSeqRat and $\mathsf{Real}^{\uparrow}$. This is the Bishop-style explicit-rate object needed for regular rational sequences and real-number convergence, with no stream quotient, ambient limit object, hidden threshold choice, limit selector, or completed real object.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope is the carrier and classifier, finite-window closure, monotone tail, $\mathsf{RegSeqRat}^{\uparrow}$ regularity, $\mathsf{Real}^{\uparrow}$ classifier transport, public consumer boundary, and explicit-modulus bridge.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchyModulusUp.CauchyModulusPacket\_public\_consumer\_boundary`

Bridge status: bridgeChecked

Origin: human

Not claimed: The chapter does not close the formal Lean namespace, the full $\mathsf{NameCert}_{\mathsf{CauchyModulus}^{\uparrow}}$ theorem, $\mathsf{Real}^{\uparrow}$ completeness, a Cauchy completion, quotient equality of real names, or a selected limit.

Upgrade path: No further formal-axis work remains for the explicit-modulus packet; the standard bridge is checked in Lean.

Chapter 293

A Concrete Naming Certificate for $\mathsf{DyadicAbsoluteValue}^{\uparrow}$

The $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ interface isolates the finite rows needed when a real-completion tolerance ledger reads the nonnegative size of a dyadic rational source. It factors the absolute-value operation through the visible $\mathsf{DyadicRatCore}^{\uparrow}$ source packet, sign split, common-exponent mantissa row, nonnegative endpoint, and tolerance ledger, so $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers can cite a named certificate without extending the near-capacity dyadic core chapter.

The source row D, common-exponent mantissa row M, endpoint row E, and tolerance row V are read from the $\mathsf{DyadicRatCore}^{\uparrow}$ source surface of chapter 291 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees the endpoint. This records the finite dyadic-core dependency used by the carrier and the consumer proof sketches below.

Definition 293.1 (Dyadic absolute-value carrier)**.** *A* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, M, E, V, H, C, P, N)$$

with the following visible coordinates:

1. *D is the source* $\mathsf{DyadicRatCore}^{\uparrow}$ *row whose mantissa, unary exponent, denominator ledger,* $\mathsf{Int}^{\uparrow}$ *provenance, and* $\mathsf{Rat}^{\uparrow}$ *provenance are read from the dyadic core surface;*
2. *S is the sign-split row deciding which displayed branch of the scaled mantissa is used for the absolute-value read;*
3. *M is the common-exponent mantissa row that exposes the source mantissa at the same denominator ledger used by dyadic distance-window observations;*
4. *E is the nonnegative output endpoint row, recorded as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint whose mantissa is the displayed nonnegative branch of M;*
5. *V is the tolerance and absolute-value ledger recording the radius window consumed by* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *routes;*
6. *H stores componentwise* hsame *transports for the source, sign split, mantissa row, nonnegative endpoint, tolerance ledger, provenance, and local name;*
7. *C stores the displayed* Cont *routes from D to S and M, from S and M to E, and from E to the tolerance ledger V;*

8. *P records* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient rationals, selected real values, trichotomy, host integer normal forms, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 293.2 (Dyadic absolute-value NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}$ *surface for Definition 293.1 consists exactly of source* $\mathsf{DyadicRatCore}^{\uparrow}$ *admission, sign-split exposure, common-exponent mantissa readback, nonnegative endpoint construction, tolerance-ledger exposure, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet A. The coordinates list the dyadic source, sign split, common-exponent mantissa, endpoint, tolerance ledger, transport, continuation, provenance, and local name.

The source and mantissa reads are inherited from the visible $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The sign split chooses the displayed nonnegative branch of the scaled mantissa, and the endpoint row repacks that branch with the same denominator ledger and provenance.

The remaining obligations are routing and naming obligations. The rows H, C, P, N transport the listed coordinates, replay the reads, preserve the $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ source ledgers, and name the packet without introducing any hidden quotient or real-completion coordinate. □

Theorem 293.3 (Dyadic absolute-value transport stability)**.** *For accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carriers whose displayed source, sign-split, mantissa, endpoint, tolerance-ledger, provenance, and naming rows are related componentwise by the stored* hsame *rows, the absolute-value endpoint and tolerance ledger remain readable after transport. The transported carrier computes the nonnegative endpoint from the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *source through the transported common-exponent mantissa row.*

Lean4 ✓ `BEDC.Derived.DyadicAbsoluteValueUp.DyadicAbsoluteValuePacket transport stability`

Proof. Project the two accepted carriers. The rows that determine the absolute-value read are exactly the source D, sign split S, common-exponent mantissa row M, endpoint E, and tolerance ledger V.

Use the componentwise transport entries in H to move those rows to the transported surface. The continuation rows in C then replay the read from D to S and M, from S and M to E, and from E to V.

The dyadic core distance-window stability theorem, Definition 291.10, supplies the underlying common-exponent absolute-value stability for the source row. Repacking that stable window with the displayed endpoint and ledger rows gives the claimed transport stability. □

Theorem 293.4 (Dyadic absolute-value nonnegative bridge)**.** *Every accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier exposes a nonnegative dyadic endpoint E whose source is the sign-split common-exponent mantissa row of D. Any* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of this endpoint reads only the displayed endpoint row, tolerance ledger, transports, continuation routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}$ *row.*

Proof. Start from the accepted carrier and read the sign-split row S. The common-exponent mantissa row M places the source mantissa of D on the same denominator ledger used by the dyadic core distance and absolute-value window.

The nonnegative endpoint E is then the displayed branch chosen by S and repacked as a dyadic endpoint with the visible $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ provenance. No separate order oracle or trichotomy row is needed, because the branch is already part of the finite packet.

A downstream consumer reaches E through the continuation routes in C and receives the tolerance ledger V as the only real-facing output surface. Thus the bridge exports a nonnegative dyadic endpoint without exporting quotient rationals, selected real values, host integer normal forms, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 293.5 (Dyadic absolute-value ledger exactness)**.** *For an accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier, the finite ledger is exact for the route from the source* $\mathsf{DyadicRatCore}^{\uparrow}$ *row through sign split, common-exponent mantissa readback, nonnegative endpoint construction, and tolerance readback to* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers. No ledger entry records a quotient rational, selected real value, trichotomy proof, host integer normal form, or real-completeness witness.*

Proof. Inspect the carrier of Definition 293.1. The only absolute-value data are D, S, M, E, V together with their transport, continuation, provenance, and naming rows.

Apply Definition 293.2 to bind every accepted read to those rows. Transport stability then shows that a transported read cannot replace the common-exponent mantissa window or tolerance ledger with an external absolute-value oracle.

Finally apply Definition 293.4. Every completion-facing use factors through the same endpoint and tolerance ledger, so the ledger is exact for the displayed finite route and contains no unlisted analytic or host-normalization coordinate. □

Theorem 293.6 (Dyadic absolute-value obligation closure package)**.** *The carrier, NameCert obligations, transport stability, nonnegative bridge, and ledger exactness of Definition 293.1, Definition 293.2, Definition 293.3, Definition 293.4, and Definition 293.5 assemble the finite* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *obligation package over* $\mathsf{DyadicRatCore}^{\uparrow}$, *the* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *sign rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance consumers,* BHist, hsame, Cont, Pkg, *and* NameCert. *The package exports only the displayed source, sign split, common-exponent mantissa, nonnegative endpoint, tolerance ledger, transport, continuation, provenance, and local naming rows.*

Proof. The carrier theorem supplies the finite packet and lists exactly the rows available to the package. The NameCert obligations bind admission, sign-split exposure, common-exponent mantissa readback, endpoint construction, tolerance-ledger exposure, transport, continuation, provenance, and naming to those coordinates.

Transport stability then shows that accepted reads remain inside the same displayed surface after componentwise hsame movement. The nonnegative bridge identifies the downstream endpoint read consumed by $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ routes.

Ledger exactness supplies the final packaging boundary. It rules out any hidden coordinate for a host absolute value, trichotomy, quotient normalization, or analytic completeness, so the five cited rows form one finite $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ obligation package rather than an appeal to an external absolute-value operation. □

Theorem 293.7 (Dyadic absolute-value endpoint determinacy)**.** *Accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carriers with the same transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, sign-split row, and common-exponent mantissa row expose the same nonnegative endpoint row. The endpoint is determined by the displayed branch row S and the mantissa row M, not by a host absolute-value operation or an external order oracle.*

Proof. Project the accepted carriers through Definition 293.1. The only rows that determine the endpoint are the source D, sign split S, common-exponent mantissa row M, and endpoint row E.

Apply Definition 293.3 to move the compared packets to the common displayed surface. On that surface the same sign-split row selects the same nonnegative branch of the same mantissa row.

Finally apply Definition 293.4. The endpoint row is the repacking of that displayed branch with the visible dyadic provenance, so two accepted carriers with the same transported source, split, and mantissa rows expose the same endpoint row. □

Theorem 293.8 (Dyadic absolute-value tolerance-window handoff)**.** *Every* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the absolute-value endpoint of an accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier reads the displayed tolerance ledger* V*. No such consumer can bypass* V *by reading a separate host absolute-value, trichotomy, quotient rational, selected real value, or completeness coordinate.*

Proof. Begin with a downstream consumer route. By Definition 293.4, the consumer reaches the endpoint only through the displayed endpoint row E together with the tolerance ledger V.

The route from source to endpoint is fixed by the carrier and by Definition 293.3. It passes through D, S, M, E and then through the continuation route from E to V.

Apply Definition 293.5. The finite ledger is exact for the route from source row through endpoint construction to completion-facing consumers, so the tolerance window V is part of every accepted handoff. □

Theorem 293.9 (Dyadic absolute-value sign-split exclusion)**.** *The* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *packet excludes trichotomy and host integer normal forms. Its only sign data are the displayed branch row* S*, the common-exponent mantissa row* M*, the nonnegative endpoint row* E*, and the tolerance ledger* V *routed by the finite carrier.*

Proof. Inspect Definition 293.1. The coordinate list includes the source row, sign-split row, mantissa row, endpoint row, tolerance ledger, transport, continuation, provenance, and local name, and it explicitly omits trichotomy and host integer normal forms.

Definition 293.2 binds the local naming surface to those displayed rows. Endpoint determinacy then shows that the endpoint is fixed by the displayed branch row and mantissa row rather than by any hidden order datum.

The exclusion is stable for downstream use by Definition 293.8. A consumer receives only the endpoint and tolerance ledger routed through the finite packet, so no trichotomy or host normalization row can be extracted from the certificate. □

Theorem 293.10 (Dyadic absolute-value scoped certificate boundary)**.** *The* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *scoped certificate boundary is exactly the package formed by the carrier, local obligation surface, transport stability, nonnegative bridge, ledger exactness, endpoint determinacy, tolerance-window handoff, and sign-split exclusion of Definition 293.1, Definition 293.2, Definition 293.3, Definition 293.4, Definition 293.5, Definition 293.7, Definition 293.8, and Definition 293.9. Every consumer read of* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *is bound to the displayed rows*

$$D, S, M, E, V, H, C, P, N$$

over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*. The boundary contains no quotient rational, selected real value, trichotomy witness, host integer normal form, external absolute-value operation, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First collect the visible inventory. The carrier lists exactly the rows $D, S, M, E, V, H, C, P, N$, and the local NameCert obligations identify those rows as the admitted source, sign-split, common-exponent mantissa, endpoint, tolerance ledger, transport, continuation, provenance, and naming surface.

Next fix the route followed by every absolute-value read. Transport stability keeps accepted reads on the transported displayed surface, while endpoint determinacy forces the endpoint through the same sign-split row and common-exponent mantissa row. The nonnegative bridge and tolerance-window handoff then route every $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer through the endpoint row E and tolerance ledger V.

Finally apply ledger exactness together with sign-split exclusion. Those theorems rule out quotient rationals, selected real values, trichotomy, host integer normal forms, external absolute-value rows, and $\mathsf{Real}^{\uparrow}$ completeness, so the scoped boundary is precisely the displayed finite packet. □

Theorem 293.11 (Dyadic absolute-value scoped public handoff)**.** *The scoped certificate boundary of* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *exposes a public consumer surface for* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consisting exactly of the rows*

$$D, S, M, E, V, H, C, P, N.$$

Every accepted public handoff reads the nonnegative endpoint through E, reads tolerance information through V, and cites the scoped boundary before any completion-facing consumer can use the packet.

Proof. Begin with the scoped certificate boundary. It binds every consumer read to the finite rows $D, S, M, E, V, H, C, P, N$ over the dyadic source, sign split, mantissa row, endpoint, tolerance ledger, transport, route, provenance, and local naming surface.

Apply the nonnegative bridge and tolerance-window handoff. The bridge identifies the endpoint row E as the only nonnegative dyadic endpoint exported to downstream consumers, and the handoff forces every completion-facing read to include the tolerance ledger V.

Finally apply ledger exactness and sign-split exclusion. These results remove quotient rationals, selected real values, trichotomy, host integer normal forms, external absolute-value operations, and $\mathsf{Real}^{\uparrow}$ completeness from the public surface, so the cited handoff is exactly the scoped finite boundary. □

Theorem 293.12 (Dyadic absolute-value RegSeqRat consumer exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance read of an accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier factors through the displayed rows*

$$D, S, M, E, V, H, C, P, N.$$

In particular, a regular-Cauchy consumer can inspect the source dyadic row, sign split, common-exponent mantissa, nonnegative endpoint, tolerance ledger, transports, routes, provenance, and local name, but it cannot inspect any absolute-value datum outside those rows.

Proof. Take a $\mathsf{RegSeqRat}^{\uparrow}$ consumer route. By the tolerance-window handoff, the route reaches the absolute-value endpoint only through E and then reads the tolerance window through V.

The source of that endpoint is fixed by transport stability and endpoint determinacy. Together they force the endpoint to be the transported sign-split branch of the common-exponent mantissa row, so the consumer also factors through D, S, M, H, C.

The remaining rows are packaging rows. The NameCert obligations and ledger exactness bind provenance and local naming to P, N and rule out any extra host absolute-value row, quotient rational, selected real, trichotomy witness, host normal form, or completeness witness. Thus the regular-Cauchy consumer surface is exhausted by the displayed packet. □

Theorem 293.13 (Dyadic absolute-value Real tolerance non-escape)**.** *No* $\mathsf{Real}^{\uparrow}$ *tolerance read of an accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *carrier can export a selected real value, trichotomy witness, quotient rational, host integer normal form, or* $\mathsf{Real}^{\uparrow}$ *completeness row. The real-facing read is limited to the nonnegative dyadic endpoint E, the tolerance ledger V, and the transport, route, provenance, and local naming rows already present in*

$$D, S, M, E, V, H, C, P, N.$$

Proof. Start with a real-facing endpoint read. The nonnegative bridge routes the read through the displayed endpoint E, and the tolerance-window handoff requires the tolerance ledger V before any $\mathsf{Real}^{\uparrow}$ consumer can use the endpoint.

The sign data are already exhausted by the sign-split exclusion theorem. It permits only the branch row S, mantissa row M, endpoint row E, and tolerance row V, and it excludes trichotomy and host integer normal forms.

Apply ledger exactness and the scoped certificate boundary. They rule out quotient rationals, selected real values, external absolute-value operations, and $\mathsf{Real}^{\uparrow}$ completeness rows, leaving only the displayed finite packet for the real tolerance read. □

Theorem 293.14 (Dyadic absolute-value Bishop induction gap)**.** *The* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *packet supplies finite Bishop-style induction over the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows: sign-split induction through S, endpoint induction through E, and tolerance induction through V. This induction route projects the source row D, common-exponent mantissa row M, nonnegative endpoint row E, tolerance ledger V, componentwise transport H, route C, provenance P, and local name N, but it does not construct quotient rationals, selected real values, trichotomy, host integer normal forms, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 293.1. The finite induction data are the source $\mathsf{DyadicRatCore}^{\uparrow}$ row D, the sign-split row S, the mantissa row M, the endpoint row E, and the tolerance ledger V.

Use endpoint determinacy and sign-split exclusion. The sign branch determines the endpoint by the displayed mantissa row, while the exclusion theorem keeps trichotomy and host integer normal forms outside the packet.

Finally use tolerance-window handoff and Real tolerance non-escape. Every completion-facing consumer reads the tolerance ledger V before it reaches a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ use, and the route refuses quotient rationals, selected real values, and ambient real completeness. Thus the available induction is exactly the finite Bishop route through the displayed dyadic rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ packet is generated from a $\mathsf{DyadicRatCore}^{\uparrow}$ source row, a sign-split row, a common-exponent mantissa row, a nonnegative endpoint row, a tolerance and absolute-value ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 293.1, Definition 293.2, Definition 293.3, Definition 293.4, Definition 293.5, Definition 293.10, Definition 293.11, Definition 293.12, and Definition 293.13 bind the mature packet boundary consisting of the finite dyadic absolute-value carrier, local NameCert obligations, transport stability, nonnegative bridge, ledger exactness, scoped certificate boundary, public handoff, RegSeqRat consumer exhaustion, and Real tolerance non-escape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DyadicAbsoluteValueUp.DyadicAbsoluteValuePacket\_transport\_stability`
Bridge status: none
Origin: human
Not claimed: This chapter does not close quotient rationals, selected real values, trichotomy, host integer normal forms, $\mathrm{Real}^{\uparrow}$ completeness, or bridge checking.
Upgrade path: The theory axis is at mature closure; the remaining upgrade is bridge checking for the displayed dyadic absolute-value packet.

Chapter 294

A Concrete Naming Certificate for $\mathsf{ControlObservability}^{\uparrow}$

The $\mathsf{ControlObservability}^{\uparrow}$ interface records the finite observation data of a finite-dimensional linear system. Its BEDC surface is the observation-row part of the standard control-theory observability test, read through $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$. The textbook reference is Kailath, *Linear Systems* (1980).

The finite observation matrix is consumed by the reachability-dual ledger of chapter 297 and by the finite controller surface of chapter 296. This chapter exports the finite output-trace ledger and kernel-separation row; those siblings consume the same matrix and vector-space rows when forming reachability and control certificates.

294.1 Carrier definition

Definition 294.1 (Control-observability carrier)**.** *A* $\mathsf{ControlObservability}^{\uparrow}$ *carrier is a finite* BHist *observation packet over* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$*. Its rows are a finite-dimensional state-space row, a state-transition row* A *read as a* $\mathsf{LinMap}^{\uparrow}$ *endomorphism, an output row* C *read as a* $\mathsf{Mat}^{\uparrow}$ *or* $\mathsf{LinMap}^{\uparrow}$ *map to an output space, the finite observation rows*

$$C, \quad CA, \quad CA^2, \quad \ldots, \quad CA^{n-1},$$

the stacked observation-matrix row $\mathcal{O}(A, C)$*, the finite output-trace ledger, the* hsame *comparisons between corresponding observation rows, the* Cont *provenance rows explaining how each trace is read from the displayed system rows, and the* Pkg *rows naming the four dependency certificates.*

The classifier identifies two accepted packets exactly when their finite observable output traces are matched by displayed hsame *rows, with the same* Cont *and* Pkg *provenance. The carrier exports no hidden host state-space quotient, analytic trajectory space, unlogged infinite trace, or external control-system object.*

Lean4 (definition) `BEDC.Derived.ControlObservabilityUp.ControlObservabilityFiniteObservationPacket`

294.2 The certificate

Theorem 294.2 (Control-observability NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ControlObservability}^{\uparrow}}$ *obligation surface consists of the following five rows.*

1. *The carrier row of Definition 294.1, including the displayed finite-dimensional* DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$ *source rows.*
2. *The classifier row saying that accepted packets are compared only by shared finite observable output traces under* hsame.
3. *The observation-matrix rank/kernel separation row for* $\mathcal{O}(A, C)$.
4. *The finite trace ledger readback row, which reads every consumer-visible output trace from the displayed observation rows.*
5. *The transport-stability row saying that* hsame *transport of the state, transition, output, observation, ledger,* Cont, *and* Pkg *rows remains an accepted* ControlObservability$^{\uparrow}$ *packet.*

```
Lean4 ✓  BEDC.Derived.ControlObservabilityUp.ControlObservabilityFiniteObservationPacket classifier
transport
Lean4 ✓  BEDC.Derived.ControlObservabilityUp.ControlObservabilityCarrierPacket obligation surface
Lean4 ✓  BEDC.Derived.ControlObservabilityUp.ControlObservationPacket namecert obligation surface
```

Proof. Start from the carrier definition. It already lists the finite source rows, the observation-matrix rows, the trace ledger, and the provenance rows, so the carrier obligation is a projection from a displayed BHist packet.

The classifier obligation is the declared comparison rule on observable traces. Because the carrier has no coordinate for hidden state quotients or infinite trace objects, equality of the visible output ledger is the only comparison surface available to consumers.

The remaining obligations are the finite matrix test, the ledger readback, and transport stability. Each is stated over rows already present in the carrier, so the obligation surface is local to the displayed BEDC packet and its four dependency certificates. □

Theorem 294.3 (Control-observability kernel separation)**.** *Let* P *be an accepted* ControlObservability$^{\uparrow}$ *packet with state-space dimension row* n*, transition row* A*, output row* C*, and observation matrix row* $\mathcal{O}(A, C)$*. If the visible kernel row of* $\mathcal{O}(A, C)$ *is the zero state row and the rank row of* $\mathcal{O}(A, C)$ *is the displayed dimension row* n*, then any two initial-state rows with the same finite output-trace ledger are classified by the state-space* hsame *row.*

```
Lean4 ✓  BEDC.Derived.ControlObservabilityUp.ControlObservabilityPacket kernel separation
Lean4 ✓  BEDC.Derived.ControlObservabilityUp.ControlObservabilityKernelSeparation state rows
hsame
```

Proof. Read the finite output traces through the observation rows of Definition 294.1. Agreement of the trace ledgers says that the displayed difference of the two initial-state rows is sent to the zero output row by every row $C, CA, \ldots, CA^{n-1}$.

Stacking those rows gives a single $\mathcal{O}(A, C)$ ledger equation. The zero-kernel row says that a state row annihilated by the whole observation matrix is the displayed zero state row.

The rank row names the same condition in finite-dimensional matrix form. Repacking the zero difference through the state-space classifier gives the required hsame comparison of the two initial-state rows. □

Theorem 294.4 (Control-observability finite trace ledger readback)**.** *Every consumer-visible output trace of an accepted* ControlObservability$^{\uparrow}$ *packet is read from the finite observation rows* $C, CA, \ldots, CA^{n-1}$*, the stacked observation-matrix row* $\mathcal{O}(A, C)$*, the displayed* hsame *comparisons,*

and the Cont/Pkg *provenance naming* DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$. *No unbounded trajectory object, hidden observer state, host controllability dual, or external realization is exported.*

Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservabilityPacket trace ledger readback`
Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservability finite trace ledger readback`
Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservabilityCarrier finite trace ledger readback`

Proof. Project an accepted packet to its finite trace ledger. Each ledger entry is attached to one of the displayed observation rows and is read through the Cont provenance of the carrier.

Project the dependency rows next. The Pkg entries name exactly the dynamical-system, matrix, vector-space, and linear-map certificates that supply the displayed transition and output rows.

Constructor inspection of the carrier excludes any additional trace source. Since no unbounded trajectory object, hidden observer state, host controllability dual, or external realization has a row in the packet, every visible trace must read back to the finite ledger just described. □

Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservabilityFiniteTraceLedger readback`

Closure status for NameCert$_{\text{ControlObservability}^{\uparrow}}$.

Constructive story: The chapter builds a finite observation packet over DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, and LinMap$^{\uparrow}$: it displays the observation-matrix row, separates rank and kernel obligations, records the finite output-trace ledger, and keeps host trajectory and full control-system data outside the carrier.

Theory closure: TheoryGate(NameCert$_{\text{ControlObservability}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped boundary is the finite BHist observation packet, the hsame trace classifier, the observation-matrix rank and kernel rows, the Cont trace ledger, and the Pkg dependencies on DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, and LinMap$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{ControlObservability}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ControlObservabilityUp.ControlObservationPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host control-system category, analytic trajectory semantics, Kalman decomposition, controllability duality, or a public standard bridge.

Upgrade path: The paper theory axis is mature; only bridgeChecked formal bridge status remains separate from this paper-axis closure.

Theorem 294.5 (Control-observability zero-dimensional observation ledger)**.** *Let P be an accepted* ControlObservability$^{\uparrow}$ *packet as in Definition 294.1. Suppose its displayed finite-dimensional state-space row has dimension $n = 0$, with transition row A, output row C, finite observation-row ledger*

$$(CA^k)_{0 \le k < n},$$

and stacked observation-matrix row $\mathcal{O}(A, C)$ formed by stacking exactly those displayed observation rows. Then the observation-row ledger is the empty finite sequence, and the stacked row $\mathcal{O}(A, C)$ has no entries.

Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservationPacket zero dimensional observation ledger`

Proof. By Definition 294.1, the only observation rows available to an accepted ControlObservability$^{\uparrow}$ packet are the finite rows

$$C, \quad CA, \quad CA^2, \quad \ldots, \quad CA^{n-1}.$$

Equivalently, these rows are indexed by the finite interval $0 \le k < n$ and have the displayed form CA^k.

If $n = 0$, there is no index k satisfying $0 \leq k < 0$. Hence the finite interval indexing the displayed observation rows is empty. The carrier has no additional observation-row coordinate beyond this finite ledger, so the observation-row sequence of P is empty.

The stacked observation-matrix row $\mathcal{O}(A, C)$ is, by the same carrier definition, the row obtained from exactly the displayed finite observation rows. Stacking over the empty finite sequence contributes no row. Therefore $\mathcal{O}(A, C)$ has no entries in the zero-dimensional state-space case. □

Lean4 ✓ `BEDC.Derived.ControlObservabilityUp.ControlObservabilityFiniteObservationPacket zero dimensional observation ledger`

Chapter 295

A Concrete Naming Certificate for $\mathsf{KalmanFilter}^{\uparrow}$

Following Kalman and Bucy, the $\mathsf{KalmanFilter}^{\uparrow}$ interface records the finite prediction-and-update packet of a discrete linear estimator. Its BEDC surface consumes $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ for transition rows, $\mathsf{NameCert}_{\mathsf{ControlObservability}^{\uparrow}}$ for observation rows, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ for covariance and gain updates, $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ for state and measurement rows, and $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ for the displayed linear maps. Standard references are Kailath, Sayed, and Hassibi, *Linear Estimation* (2000), and Anderson and Moore, *Optimal Filtering* (1979).

The dependency surface is displayed through existing concrete siblings: chapter 329 supplies the state-transition packet, chapter 307 supplies the Riccati covariance ledger, chapter 294 supplies the observability row, chapter 49 and chapter 48 supply the linear algebra rows, chapter 184 and chapter 185 mark the non-exported stochastic boundary, chapter 230 supplies the estimator comparison surface, and chapter 38 supplies the scalar real rows. The deterministic observer-correction sibling chapter 5909 supplies the adjacent state-space, observability, output-error, and gain rows at the non-stochastic boundary of the prediction-update packet. $\mathsf{KalmanFilter}^{\uparrow}$ consumes those rows as finite BHist coordinates before exposing its prediction-update $\mathsf{NameCert}$ surface.

295.1 Finite carrier

Definition 295.1 (KalmanFilterUp BHist carrier)**.** *A* $\mathsf{KalmanFilter}^{\uparrow}$ *carrier is a finite* BHist *packet whose visible rows are:*

1. *a state-estimate row* $\widehat{x}$ *and predicted-state row* $\widehat{x}^{-}$ *over* $\mathsf{VecSpace}^{\uparrow}$*;*
2. *an error and covariance ledger row* P^{-}, P, K, P^{+} *over* $\mathsf{Mat}^{\uparrow}$*;*
3. *a transition row* A *and optional input row read from* $\mathsf{DynSystem}^{\uparrow}$*;*
4. *an observation row* C *and finite observation ledger read from* $\mathsf{ControlObservability}^{\uparrow}$*;*
5. *linear-map rows for prediction, observation, innovation, gain application, and estimate transport read from* $\mathsf{LinMap}^{\uparrow}$*;*
6. *matrix update rows for the displayed Riccati-style covariance, gain, and posterior covariance ledgers read from* $\mathsf{Mat}^{\uparrow}$*;*
7. Cont *and* Pkg *provenance rows naming the consumed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$ *certificates.*

No host stochastic process, probability measure, analytic convergence theorem, infinite trajectory, or external estimator object is part of the carrier.

Lean4 (definition) `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier`
Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterTasteGate single carrier alignment`

Theorem 295.2 (KalmanFilterUp prediction and update obligation)**.** *Every accepted* KalmanFilter$^{\uparrow}$ *packet exposes a finite prediction row, innovation row, gain row, and posterior-update row. The prediction row reads the transition and prior estimate through* DynSystem$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$*; the update row reads the observation, innovation, gain, and posterior estimate through* ControlObservability$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$*. Each row remains a* BHist *row with its displayed* Cont/Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier prediction update obligation`

Proof. Inspect the carrier of Definition 5102.1. The prediction rows are exactly the transition, state-estimate, and linear-map rows, while the update rows are exactly the observation, innovation, gain, and posterior-estimate rows.

The matrix rows supply the covariance and gain ledgers, the vector rows supply the state and observation endpoints, and the provenance rows name the certificates consumed by each read. Since the packet has only finite BHist rows, every exported prediction or update entry remains inside the displayed carrier. □

Theorem 295.3 (KalmanFilterUp observation innovation ledger)**.** *For an accepted* KalmanFilter$^{\uparrow}$ *packet, the observation residual and innovation covariance are read from the displayed observation row of* ControlObservability$^{\uparrow}$*, the predicted estimate row of* VecSpace$^{\uparrow}$*, and the covariance and observation-matrix rows of* Mat$^{\uparrow}$*. The innovation ledger is exact: no unrecorded measurement stream, hidden residual object, or host probabilistic equality contributes to the consumer-visible innovation row.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier observation innovation ledger`

Proof. Read the observation row first. By Definition 5102.1, it is a finite ControlObservability$^{\uparrow}$ row paired with vector, matrix, and linear-map rows.

The residual is the displayed comparison between the observation endpoint and the predicted observation endpoint. The innovation covariance is the displayed matrix update over the covariance and observation rows. Because the carrier excludes host stochastic and trajectory objects, exactness is only the projection from these finite rows. □

Theorem 295.4 (KalmanFilterUp prediction-update ledger exactness)**.** *For every accepted* KalmanFilter$^{\uparrow}$ *packet, each prediction, innovation, gain, posterior-estimate, and posterior-covariance read factors through the displayed finite* BHist *packet. The packet rows are exactly the* DynSystem$^{\uparrow}$ *transition rows,* ControlObservability$^{\uparrow}$ *observation rows,* Mat$^{\uparrow}$ *covariance and gain rows,* VecSpace$^{\uparrow}$ *state rows,* LinMap$^{\uparrow}$ *linear-map rows, the* Cont *and* hsame *comparison rows, and the* Pkg *provenance rows named by the carrier. No consumer-visible prediction-update read uses a stochastic process, probability space, unrecorded random variable, or external estimator semantics.*

Proof. Project the accepted carrier of Definition 5102.1. Its prediction face consists of the prior estimate, transition, predicted estimate, and predicted covariance rows, together with the LinMap$^{\uparrow}$, Cont, hsame, and Pkg rows that name the finite calculation.

Split the update face into observation, innovation, gain, posterior-estimate, and posterior-covariance reads. The observation and innovation reads are exact by Definition 5102.3, while the gain and posterior rows are the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ entries transported by the same packet.

Estimate transport stability is supplied by Definition 5102.5. Thus any representative change remains inside the same finite ledger, and the excluded stochastic and probability-space objects cannot be recovered from the certificate surface. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier prediction update endpoint transport`

Theorem 295.5 (KalmanFilterUp estimate transport stability)**.** *If two accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *packets have* hsame*-matched state-estimate, covariance, transition, observation, linear-map,* Cont*, and* Pkg *rows, then prediction, innovation, gain, posterior-estimate, and posterior-covariance reads transport to accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *rows. The classifier is therefore stable under componentwise* hsame *transport of the displayed carrier.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier estimate transport stability`

Proof. Transport each dependency row componentwise along the displayed hsame witnesses. The transition and observation rows remain accepted by their consumed certificates, while the vector, matrix, and linear-map rows remain inside their carrier surfaces.

Apply the same componentwise transport to the prediction, innovation, gain, and posterior rows. The Cont and Pkg entries transport the provenance, so the resulting packet again satisfies the finite carrier inventory of Definition 5102.1. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier estimate transport`

Theorem 295.6 (KalmanFilterUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *prediction carrier: prior estimate, transition, predicted estimate, and predicted covariance rows;*
2. *update carrier: observation, residual, gain, posterior estimate, and posterior covariance rows;*
3. *innovation ledger exactness: the residual and innovation covariance are exactly the displayed observation, prediction, covariance, and matrix-update reads;*
4. *classifier transport under* hsame*: componentwise transport preserves accepted prediction and update packets;*
5. *downstream read boundary: consumers may read only the prediction, update, innovation, covariance, classifier,* Cont*, and* Pkg *rows displayed by the carrier.*

Proof. The prediction carrier and update carrier are the two action faces of Definition 5102.1. Their rows are finite because each consumed dependency certificate exports finite BHist data.

The innovation row is exact by Definition 5102.3, and classifier stability is the componentwise transport of Definition 5102.5. The consumer boundary is the remaining projection from the same finite inventory. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier namecert obligation surface`

Theorem 295.7 (KalmanFilterUp downstream consumer boundary)**.** *Any algebraic or control-theory consumer of* $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$ *may read only the finite state-estimate, covariance, transition, observation, innovation, gain, posterior-update,* hsame, Cont, *and* Pkg *rows of the carrier. It cannot infer a host probability space, optimality theorem, convergence rate, controllability decomposition, or external trajectory semantics from the* $\mathsf{KalmanFilter}^{\uparrow}$ *certificate alone.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier endpoint closed generation`
Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier downstream consumer boundary`

Proof. Project the exported rows from Definition 5102.1. These are precisely the finite rows named by the prediction-update obligation and the NameCert obligation surface.

All excluded objects are absent from the carrier inventory. Since consumer reads are projections from accepted BHist rows and their provenance, no consumer can recover a host stochastic or analytic object without an additional certificate. □

Theorem 295.8 (KalmanFilterUp standard prediction-update bridge)**.** *The public* $\mathsf{KalmanFilter}^{\uparrow}$ *carrier and consumer boundary interpret exactly as the standard finite discrete prediction-update schema over displayed* $\mathsf{DynSystem}^{\uparrow}$ *transition rows,* $\mathsf{ControlObservability}^{\uparrow}$ *observation rows,* $\mathsf{Mat}^{\uparrow}$ *covariance and gain rows,* $\mathsf{VecSpace}^{\uparrow}$ *state rows,* $\mathsf{LinMap}^{\uparrow}$ *linear-map rows, and visible* hsame/Cont/Pkg *provenance. No stochastic process, probability-space semantics, convergence theorem, or external estimator object is exported.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterUp StdBridge`

Proof. Start from the carrier of Definition 5102.1. Its rows are the finite state-estimate, covariance, transition, observation, innovation, gain, posterior-update, linear-map, transport, continuation, and package rows.

The prediction-update face is exactly Definition 5102.2. The observation and innovation reads are controlled by Definition 5102.3, and the full ledger is exact by Definition 5102.4.

Representative changes remain inside the same finite packet by Definition 5102.5. The $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$ surface is the finite obligation surface of Definition 5102.6, and downstream consumers are bounded by Definition 5102.7.

Repack those finite rows as the standard prediction-update schema. Since every schema field is one of the displayed BEDC rows and the consumer boundary has no coordinate for stochastic, probabilistic, convergence, or external-estimator data, the interpretation exports exactly the finite prediction-update packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5102.8 binds the public finite Kalman prediction-update packet to $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows.

Constructive story: The construction starts from the finite observation packet and dynamic-system transition row, adds matrix and vector rows for prediction, covariance propagation, innovation, gain, and posterior estimate transport, and records the whole calculation as BHist rows with Cont and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier\_downstream\_consumer\_boundary`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close stochastic optimality, analytic convergence, probability-space semantics, or controllability decomposition.

Upgrade path: No further formal-axis upgrade remains for the finite Kalman packet.

Theorem 295.9 (KalmanFilterUp zero-gain posterior estimate collapse)**.** *Let $\mathcal{K}$ be an accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *packet in the sense of Definition 5102.1. Write*

$$\widehat{x}^{-}$$

for its predicted-estimate row, K for its gain row, ν for its innovation row, $K\nu$ for the displayed gain-application row, and $\widehat{x}^{+}$ for its posterior-estimate row. Assume that the update face is the displayed affine Kalman estimate row

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} K\nu,$$

with the gain application $K\nu$ read through the $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{LinMap}^{\uparrow}$ *rows exposed by Definition 5102.2. If the gain row is classified with the zero matrix row,*

$$K \sim_{\mathsf{Mat}^{\uparrow}} Z,$$

where Z is the zero matrix datum of Definition 49.113, then the posterior-estimate row is classified with the predicted-estimate row:

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-}.$$

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier zero gain posterior estimate collapse`

Proof. Project the accepted packet by Definition 5102.2. The projection supplies the finite prediction row, innovation row, gain row, posterior-estimate row, and the provenance rows saying that the gain application is read only through the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ certificates. By Definition 5102.1, no hidden estimator object or stochastic equality is available; every comparison must be assembled from those rows.

The gain hypothesis identifies the carried gain endpoint with the zero matrix datum Z. Matrix classifier stability transports the gain-application row along this comparison, so the displayed product $K\nu$ is classified with the zero-gain product $Z\nu$. The finite-fold zero matrix multiplication law, Definition 49.114, classifies $Z\nu$ with the zero vector endpoint in the codomain vector carrier. Thus the gain-application contribution satisfies

$$K\nu \sim_{\mathsf{VecSpace}^{\uparrow}} 0.$$

The use of Definition 49.114 is entrywise: each scalar summand has a zero matrix factor, the scalar zero-product row reduces it to the scalar zero, and the finite additive fold reduces the whole displayed product to the zero endpoint.

Now apply vector-space addition congruence to the posterior affine row. The comparison $K\nu \sim_{\mathsf{VecSpace}^{\uparrow}} 0$ transports

$$\widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} K\nu$$

to

$$\widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} 0.$$

The vector additive right-unit row in the $\mathsf{VecSpace}^{\uparrow}$ certificate identifies this endpoint with $\widehat{x}^{-}$. Transitivity in the $\mathsf{VecSpace}^{\uparrow}$ classifier composes the displayed affine update comparison, the transported zero-gain comparison, and the additive right-unit comparison. Hence

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-}.$$

All rows used in the argument are among the finite prediction, innovation, gain, posterior-estimate, matrix, vector-space, linear-map, Cont, hsame, and Pkg rows named by Definition 5102.1; the consumer boundary of Definition 5102.7 excludes any additional analytic or probabilistic premise. □

Chapter 296

A Concrete Naming Certificate for $\mathsf{LQR}^{\uparrow}$

The $\mathsf{LQR}^{\uparrow}$ interface records the finite certificate surface of a linear-quadratic regulator: state and control rows, a displayed linear transition row, a quadratic cost ledger, a finite horizon index, and a backward dynamic-programming ledger. The control-theory source is the standard finite-horizon linear-quadratic regulator construction as presented in Bertsekas, *Dynamic Programming and Optimal Control* (2017), or Anderson and Moore, *Optimal Control: Linear Quadratic Methods* (2007). The finite $\mathsf{LQR}^{\uparrow}$ packet consumes the displayed dynamics rows of chapter 195, matrix rows of chapter 49, vector-space rows of chapter 47, real cost rows of chapter 38, and estimator/probability rows of chapter 184, chapter 185, and chapter 230.

296.1 Carrier definition

Definition 296.1 (LQR finite control carrier)**.** *An* $\mathsf{LQR}^{\uparrow}$ *carrier is a finite* BHist *packet whose rows are a state row, a control row, a linear transition row, a quadratic cost row, a finite horizon unary index, a Riccati or backward dynamic-programming ledger, displayed estimator-input rows,* hsame *comparisons between corresponding rows,* Cont *provenance rows for every backward step, and* Pkg *rows naming* $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, *and* $\mathsf{Estimator}^{\uparrow}$.

The classifier accepts exactly those packets whose transition, cost, horizon, estimator, and backward-update rows are read through the displayed dependency certificates and finite provenance ledger. It exports no host stochastic process, analytic optimal-control problem, unbounded horizon, external Riccati solver, or hidden policy object.

Lean4 (definition) `BEDC.Derived.LQRUp.LQRFiniteControlCarrier`

296.2 The certificate

Theorem 296.2 (LQR NameCert seed obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LQR}^{\uparrow}}$ *seed surface consists of the following five obligation rows.*

1. *The carrier and classifier row of Definition 296.1, including the displayed state, control, transition, cost, horizon, estimator, and provenance rows.*
2. *The transition-stability row saying that* hsame *transport of state, control, linear transition, horizon,* Cont, *and* Pkg *rows remains inside the same finite* $\mathsf{LQR}^{\uparrow}$ *packet.*

3. *The quadratic-cost ledger exactness row saying that every visible cost contribution is read from the displayed* $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows and from no ambient cost functional.*
4. *The backward-update row stating that each finite horizon step in the Riccati or dynamic-programming ledger consumes only the previous displayed value row, the transition row, the control row, the quadratic cost row, and the visible provenance rows.*
5. *The estimator/control consumption boundary saying that* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, *and* $\mathsf{Estimator}^{\uparrow}$ *inputs may be consumed only through the displayed estimator rows and cannot introduce a host stochastic control model.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket namecert seed obligation surface`

Proof. Start from the carrier of Definition 296.1. It lists the finite BHist rows that make the state, control, transition, cost, horizon, estimator, update, and provenance data visible.

Transport stability is a componentwise statement over those rows. Since the classifier reads only displayed hsame comparisons and the same finite Cont ledger, transporting the rows preserves the accepted packet shape.

For the cost and backward-update rows, inspect the finite ledger one horizon index at a time. Each step consumes matrix, vector-space, real-valued cost, transition, control, and previous value rows that already appear in the packet, so no analytic optimizer or unbounded horizon can be projected.

Finally project the estimator boundary. The probability, random-variable, and estimator certificates appear only as named source rows and finite estimator-input ledgers; therefore any control consumer must read them through the displayed packet rather than through an external stochastic model. □

Theorem 296.3 (LQR dynamic-programming row)**.** *In an accepted* $\mathsf{LQR}^{\uparrow}$ *packet, every finite backward-update row is a* Cont *step from the displayed successor value row, linear transition row, control row, quadratic cost row, estimator-input row, and* Pkg *provenance to the displayed predecessor value row. The update ledger has no coordinate for an external Riccati solver or analytic value function.*

Proof. Project the backward ledger from the carrier. Its entries are indexed by the finite horizon unary row and each entry names its predecessor and successor value rows.

Read one entry. The input coordinates are precisely the transition, control, quadratic cost, estimator, and provenance rows displayed in the packet, and the output coordinate is the next predecessor value row in the same finite ledger.

Because the carrier contains no row for an external solver or analytic value function, the only accepted dynamic-programming update is the displayed Cont step. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQR dynamic programming cont determinacy`

Theorem 296.4 (LQR estimator-control consumption boundary)**.** *Every downstream consumer of* $\mathsf{LQR}^{\uparrow}$ *can read estimator-dependent control data only through the displayed* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Estimator}^{\uparrow}$, *state, control, transition, quadratic cost, horizon, backward-update,* Cont, *and* Pkg *rows of Definition 296.1. No consumer can demand an unlogged stochastic process, hidden controller state, infinite-horizon policy, or ambient optimization problem from the certificate.*

Proof. Begin with an accepted downstream readback. The carrier theorem restricts the readback to the finite packet rows and their visible classifier.

The estimator-dependent rows are source rows named by the probability, random-variable, and estimator certificates. Their consumption route is the displayed Cont ledger that also supplies the state, control, transition, cost, and backward-update rows.

Apply the dynamic-programming row theorem to the control readback. Since each backward step is internal to the displayed finite ledger, no downstream consumer can project an unlogged stochastic process, hidden controller state, infinite-horizon policy, or ambient optimization problem. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket estimator control consumption boundary`

Theorem 296.5 (LQR transition stability)**.** *Let L and L' be accepted* $\mathsf{LQR}^{\uparrow}$ *packets. If the state, control, linear transition, horizon,* Cont*, and* Pkg *rows of L are transported to the corresponding rows of L' by displayed* hsame *comparisons, then the transported packet remains inside the same finite* $\mathsf{LQR}^{\uparrow}$ *carrier surface. The transition readback of L' is exactly the transported transition row together with the transported state, control, horizon, continuation, and provenance rows.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlCarrier transition transport`

Proof. Project the finite carrier of Definition 296.1. Its transition component is not a host map; it is a displayed row tied to the state row, control row, horizon row, finite continuation ledger, and package provenance.

Transport those coordinates componentwise along the supplied hsame comparisons. The classifier for an accepted packet reads exactly those displayed comparisons, so the transported tuple has the same carrier shape.

Finally repack the transported transition row with the transported horizon and provenance rows. Since no external dynamics object is present in the carrier, the transition readback of L' is precisely this finite BEDC row. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlCarrier transition stability`
↪ variant `BEDC.Derived.LQRUp.LQRFiniteControlPacket transition stability`
↪ variant `BEDC.Derived.LQRUp.LQRFiniteControlPacket transition transport`

Theorem 296.6 (LQR quadratic-cost exactness)**.** *Every visible cost contribution in an accepted* $\mathsf{LQR}^{\uparrow}$ *packet reads only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, state, control, horizon,* Cont*, and* Pkg *rows of Definition 296.1. The packet exports no ambient quadratic functional, analytic norm, host expectation, or hidden optimizer that can alter the finite cost ledger.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlCarrier quadratic cost exactness`

Proof. Read a cost entry from the finite packet. The carrier exposes the matrix row, vector-space rows for state and control coordinates, real-valued coefficient rows, the horizon index, and the package provenance naming the dependency certificates.

The cost contribution is then reconstructed as a displayed finite ledger entry over those rows. Its scalar output is a $\mathsf{Real}^{\uparrow}$ row reached by the packet's Cont route, not by an ambient cost functional.

Because the carrier has no coordinate for a host expectation, analytic norm, or optimizer, every downstream cost readback is exhausted by the displayed matrix, vector-space, real, horizon, continuation, and provenance rows. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket cost endpoint determinacy`
Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket quadratic cost exactness`
Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket quadratic cost transport exactness`

Theorem 296.7 (LQR finite-horizon Riccati closure)**.** *Every backward dynamic-programming or Riccati ledger step in an accepted* $\mathsf{LQR}^{\uparrow}$ *packet is a finite* Cont *row from the displayed successor value, transition, control, quadratic cost, estimator-input, horizon, and* Pkg *rows to the displayed predecessor value row. The certificate exports no infinite-horizon policy, limiting algebraic Riccati equation, external solver, or host optimal-control object.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlCarrier endpoint closed generation`
Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket finite horizon riccati closure`

Proof. Unpack the backward ledger indexed by the finite horizon row. Each entry has a successor value row and a predecessor value row, and its input coordinates are the transition, control, quadratic cost, estimator, and provenance rows already listed in the carrier.

Apply Definition 296.3 to a single entry. The update is a Cont step internal to the finite packet, and iterating along the unary horizon merely repeats the same displayed ledger pattern.

The terminal row is therefore reached by a finite chain of continuation entries. No row remains from which a consumer could read an infinite-horizon policy, a limiting Riccati equation, an external solver, or a host optimal-control object. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket finite horizon riccati transport`

Theorem 296.8 (LQR scoped dependency certificate)**.** *For the finite-control carrier of Definition 296.1, the seed obligation surface, dynamic-programming row, estimator-control boundary, transition stability row, quadratic-cost exactness row, and finite-horizon Riccati closure bind every exported* $\mathsf{LQR}^{\uparrow}$ *row to the displayed* $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Estimator}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and dependency-certificate rows. The resulting certificate exports no host stochastic process, infinite-horizon policy, ambient optimizer, or external Riccati solver.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket scoped dependency certificate`

Proof. Project the rows of the carrier in Definition 296.1. This gives the state, control, transition, cost, horizon, estimator, backward-update, hsame, Cont, and Pkg coordinates as displayed finite BHist rows.

Apply the transition stability theorem and the quadratic-cost exactness theorem to the displayed transition and cost coordinates. These two rows bind the dynamics and cost readbacks to the named $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependency packets.

Next apply the dynamic-programming row theorem, the estimator-control boundary, and the finite-horizon Riccati closure theorem. The backward ledger then consumes only the displayed successor value, transition, control, quadratic-cost, estimator-input, horizon, continuation, and provenance rows.

It remains to check the exported packet fields. Exhausting the displayed carrier shows that the only remaining public coordinates are the named dependency certificates and their hsame, Cont, and Pkg rows, so no host stochastic process, infinite-horizon policy, ambient optimizer, or external Riccati solver is exported. □

Theorem 296.9 (LQR empty-horizon backward ledger)**.** *Let L be an accepted* $\mathsf{LQR}^{\uparrow}$ *packet. If the finite horizon unary index row of L is the empty unary row, then the Riccati or backward dynamic-programming ledger of L contains no backward-update entries. Equivalently, there is no displayed* Cont *row in L whose role is a finite-horizon backward update from a successor value row to a predecessor value row.*

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket empty horizon predecessor update hsame`

Proof. Project the carrier of Definition 296.1. Its backward dynamic-programming ledger is not an ambient list of solver events; it is the displayed finite ledger whose entries are indexed by the finite horizon unary row and whose accepted entries must be read through the packet's visible provenance rows.

Assume for contradiction that the backward ledger contains an update entry. By Definition 296.3, that entry is a Cont step from the displayed successor value row, transition row, control row, quadratic cost row, estimator-input row, and Pkg provenance to the displayed predecessor value row. The same theorem also identifies the entry as one finite backward-update row indexed by the finite horizon unary row.

The empty unary row has no finite horizon index position. Hence it supplies no coordinate at which such a backward-update entry can be placed. This contradicts the existence of the displayed update entry, so the Riccati or backward dynamic-programming ledger contains no update entries. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlCarrier empty horizon backward ledger`
Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket empty horizon backward ledger`

Theorem 296.10 (LQRUp public NameCert export)**.** *The finite-control carrier of Definition 296.1, together with the seed obligation surface, transition stability, quadratic-cost exactness, estimator-control boundary, finite-horizon Riccati closure, and empty-horizon ledger, assembles the public* $\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$ *over the displayed* $\mathsf{DynSystem}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, $\mathsf{RandomVar}^\uparrow$, $\mathsf{Estimator}^\uparrow$, BHist, hsame, Cont, *and* Pkg *rows. This public surface exports no host stochastic process, infinite-horizon policy, ambient optimizer, or external Riccati solver.*

Proof. Project the finite-control carrier. Its visible rows are exactly the state, control, transition, cost, estimator, horizon, backward-ledger, hsame, Cont, and Pkg coordinates already bound in the scoped dependency certificate.

Apply Definition 296.8 to bind those rows to the displayed $\mathsf{DynSystem}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, $\mathsf{RandomVar}^\uparrow$, and $\mathsf{Estimator}^\uparrow$ sources. The transition-stability, quadratic-cost exactness, estimator-control, and finite-horizon Riccati rows supply the public certificate fields without importing an ambient control model.

Use Definition 296.9 for the terminal boundary. It rules out hidden predecessor updates at the empty horizon, so the public packet cannot read an infinite-horizon policy or an external Riccati solver from a missing ledger row.

Repack the displayed rows as the public $\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$ surface. Every exported coordinate is one of the named carrier, dependency, classifier, continuation, or provenance rows, and every non-displayed stochastic or optimizer object remains outside the certificate. □

Lean4 ✓ `BEDC.Derived.LQRUp.LQRFiniteControlPacket public namecert export`

Theorem 296.11 (LQR public schema bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$ *surface of Definition 296.10 interprets exactly as the standard finite-horizon LQR schema. The displayed state and control rows, transition row, quadratic-cost row, estimator-input rows, finite horizon row, and backward dynamic-programming ledger are precisely the public schema fields of the exported certificate. The surface exports no infinite-horizon policy, host Riccati solver, stochastic process, or ambient optimizer.*

Proof. Begin with the scoped dependency certificate of Definition 296.8. It binds the state, control, transition, quadratic cost, estimator, horizon, continuation, provenance, and dependency rows to the displayed finite-control carrier.

The finite-horizon part is the Riccati closure of Definition 296.7. It identifies each backward dynamic-programming ledger entry as a finite Cont row from the displayed successor value, transition, control, quadratic-cost, estimator-input, horizon, and package rows to the predecessor value row.

The estimator side is separated by Definition 296.4. Thus estimator input is a public certificate row consumed by the finite ledger, not a hidden stochastic process or ambient controller.

Finally apply Definition 296.10. Repacking those rows as the public $\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$ surface gives exactly the standard finite-horizon LQR schema fields, while the excluded infinite-horizon policy, host Riccati solver, stochastic process, and ambient optimizer remain outside the export. □

Closure status for $\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$**.**

Constructive story: An $\mathsf{LQR}^\uparrow$ packet is generated by laying out a finite BHist control ledger: state and control rows, a linear transition row, a quadratic-cost row, a unary horizon row, estimator-input rows, backward dynamic-programming rows, rowwise hsame comparisons, Cont provenance for each horizon step, and Pkg rows for the displayed dynamics, matrix, vector-space, real, probability, random-variable, and estimator certificates. The classifier accepts exactly this finite shape after componentwise transport, so a transported transition, cost, or backward-update row remains part of the same packet. The ledger and source specification align because each cost and predecessor-value coordinate is reached through the displayed continuation route from the named dependency rows. Nonemptiness is the finite empty-horizon or finite-horizon ledger; reflexivity, symmetry, and transitivity are supplied rowwise by hsame, with no hidden stochastic process, solver, or optimizer coordinate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$, maturePackageClosed)

Scope closed: The public certificate binds the finite linear-quadratic control carrier, seed obligation surface, dynamic-programming row, estimator-control boundary, transition stability, quadratic-cost exactness, finite-horizon Riccati closure, empty-horizon ledger, and public finite-horizon LQR schema bridge to the displayed $\mathsf{DynSystem}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{ProbSpace}^\uparrow$, $\mathsf{RandomVar}^\uparrow$, $\mathsf{Estimator}^\uparrow$, BHist, hsame, Cont, Pkg, and dependency-certificate rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LQR}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.LQRUp.LQRFiniteControlPacket\_public\_namecert\_export`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close an analytic stochastic-control model, infinite-horizon regulator, host Riccati solver, projective limit, or stochastic-process semantics.

Upgrade path: No further theory-axis upgrade is asserted here; bridgeChecked remains a formal-axis matter for the standard bridge audit.

Chapter 297

A Concrete Naming Certificate for ControlControllability$^{\uparrow}$

The ControlControllability$^{\uparrow}$ interface records the finite reachability surface of a linear control system. Its textbook source is the Kalman controllability rank-style data of Kalman, *On the General Theory of Control Systems* (1960), but the BEDC chapter states only the finite reachability ledger and NameCert surface. The dependency anchors are NameCert$_{\text{DynSystem}^{\uparrow}}$, NameCert$_{\text{Mat}^{\uparrow}}$, NameCert$_{\text{VecSpace}^{\uparrow}}$, NameCert$_{\text{LinMap}^{\uparrow}}$, NameCert$_{\text{ControlObservability}^{\uparrow}}$, NameCert$_{\text{KalmanFilter}^{\uparrow}}$, and NameCert$_{\text{LQR}^{\uparrow}}$.

The finite reachability rows consume the observation-matrix sibling surface of chapter 294 as the dual ledger for the same displayed state, transition, matrix, and finite-horizon rows before KalmanFilter$^{\uparrow}$ and LQR$^{\uparrow}$ consumers read the packet.

297.1 Carrier definition

Definition 297.1 (ControlControllability finite reachability carrier)**.** *A* ControlControllability$^{\uparrow}$ *carrier is a finite* BHist *packet whose rows are a state-carrier row, an input-carrier row, a transition-matrix row A, a control-matrix row B, a finite horizon unary row n, the displayed reachability columns $B, AB, \ldots, A^{n-1}B$, a reachability-matrix row collecting those columns,* hsame *comparisons between corresponding finite rows,* Cont *provenance for each displayed multiplication and column append, and* Pkg *rows naming* DynSystem$^{\uparrow}$*,* Mat$^{\uparrow}$*,* VecSpace$^{\uparrow}$*,* LinMap$^{\uparrow}$*,* ControlObservability$^{\uparrow}$*,* KalmanFilter$^{\uparrow}$*, and* LQR$^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket`
Lean4 (definition) `BEDC.Derived.ControlControllabilityUp.ControlControllabilityCarrier`

Definition 297.2 (ControlControllability classifier)**.** *Two* ControlControllability$^{\uparrow}$ *carriers are classified as the same finite reachability certificate exactly when their state rows, input rows, A rows, B rows, horizon rows, reachability-column rows, reachability-matrix rows,* Cont *routes, and* Pkg *rows agree under displayed* hsame *transports and the* Mat$^{\uparrow}$ *column-transport ledger. The classifier does not quotient by a host reachable subspace and does not identify packets by an ambient controllability predicate.*

Lean4 (definition) `BEDC.Derived.ControlControllabilityUp.ControlControllabilityClassifier`
Lean4 (definition) `BEDC.Derived.ControlControllabilityUp.ControlControllabilityCarrier classifier`

297.2 The certificate

Theorem 297.3 (ControlControllability reachability ledger)**.** *Every accepted* ControlControllability$^{\uparrow}$ *packet exposes exactly the state-carrier row, input-carrier row, transition row A, control row B, finite horizon row n, displayed reachability columns $B, AB, \ldots, A^{n-1}B$, reachability-matrix row,* hsame *transports,* Cont *provenance, and* Pkg *dependency rows of Definition 297.1. It exports no host reachable subspace, rank theorem, Kalman decomposition, analytic trajectory semantics, or infinite-horizon object.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket reachability ledger`

Proof. Project the accepted carrier to the rows listed in Definition 297.1. The state, input, transition, control, horizon, column, reachability-matrix, transport, continuation, and provenance rows are visible as finite BHist coordinates.

Conversely, every consumer-visible reachability read enters through one of those coordinates. Matrix multiplication and column append are recorded as Mat$^{\uparrow}$ and LinMap$^{\uparrow}$ ledger rows, while the state and input endpoints are recorded through VecSpace$^{\uparrow}$ and DynSystem$^{\uparrow}$ rows.

The carrier has no coordinate for a host reachable subspace, rank theorem, Kalman decomposition, analytic trajectory, or infinite horizon. Therefore those objects cannot be exported by the certificate. □

Theorem 297.4 (ControlControllability reachability-column transport obligation)**.** *Every displayed reachability column $B, AB, \ldots, A^{n-1}B$ in a* ControlControllability$^{\uparrow}$ *packet is generated by finite* Mat$^{\uparrow}$, LinMap$^{\uparrow}$, Cont, *and* BHist *rows. Transport along the displayed* hsame *comparisons preserves the reachability-matrix packet by transporting each carrier column and the recorded column-append route, without invoking a host rank predicate or a host reachable subspace.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket column transport obligation`

Proof. Project the accepted packet to its $B, AB, \ldots, A^{n-1}B$ rows. Each projected column is attached to a finite multiplication route, and the reachability-matrix row is the ordered append of those displayed columns.

Now transport the source rows along hsame. The multiplication rows remain the same finite Mat$^{\uparrow}$ and LinMap$^{\uparrow}$ routes after transport, and the append route remains a Cont coordinate of the same packet shape.

The preserved object is therefore the displayed reachability-matrix packet itself. No coordinate in the construction names a rank predicate or host reachable subspace, so no such predicate is used by the transport obligation. □

Theorem 297.5 (ControlControllability classifier stability obligation)**.** *The* ControlControllability$^{\uparrow}$ *classifier is reflexive, symmetric, and transitive componentwise over the state row, input row, A row, B row, horizon row, reachability-column rows, reachability-matrix row,* Cont *rows, and* Pkg *rows of the carrier packet.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityCarrier classifier stability obligation`

Proof. For reflexivity, apply the rowwise reflexivity of hsame to the state, input, matrix, horizon, reachability, continuation, and package coordinates, then repack the resulting finite row comparisons as a classifier witness.

For symmetry, reverse each displayed row comparison. The carrier has only finitely many named coordinates, so the reversed state, input, A, B, horizon, column, reachability-matrix, Cont, and Pkg comparisons assemble into the symmetric classifier witness.

For transitivity, compose the corresponding row comparisons coordinate by coordinate. The composed rows have the same packet shape as the original classifier data, and repacking them gives the transitive classifier witness. □

Theorem 297.6 (ControlControllability dependency scope obligation)**.** *All public reachability reads from a* $\mathsf{ControlControllability}^{\uparrow}$ *packet factor through* $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{ControlObservability}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{LQR}^{\uparrow}}$ *rows together with the displayed* BHist, hsame, Cont, *and* Pkg *coordinates. They do not factor through host reachable subspaces, Kalman decomposition objects, or analytic trajectories.*

Proof. Inspect the package provenance row of the carrier. Its named dependencies are exactly the displayed $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, and $\mathsf{LQR}^{\uparrow}$ certificate rows, together with the kernel rows that record history, transport, continuation, and packaging.

Every public reachability read is a projection from that finite provenance packet or from the reachability rows transported by it. The dependency factorization is therefore exhausted by the named NameCert rows and the displayed BHist, hsame, Cont, and Pkg coordinates.

The packet contains no coordinate for a host reachable subspace, no Kalman decomposition object, and no analytic trajectory semantics. Those objects are outside the dependency scope of the public surface. □

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket dependency scope`

Theorem 297.7 (ControlControllability NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ControlControllability}^{\uparrow}}$ *seed surface consists of the carrier and classifier rows of Definition 297.1 and Definition 297.2, the finite reachability-ledger row of Definition 297.3, the reachability-column transport obligation of Definition 297.4, the classifier stability obligation of Definition 297.5, and the dependency scope obligation of Definition 297.6.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityCarrier namecert obligation surface`

Proof. Start with the finite carrier and classifier. They determine the visible packet shape and the rowwise comparison relation for all state, input, transition, control, horizon, column, reachability-matrix, continuation, and provenance coordinates.

The reachability-ledger theorem identifies the exact exported rows. The matrix column-transport row is then a finite componentwise transport of the displayed $B, AB, \ldots, A^{n-1}B$ columns, not an appeal to a host reachable-space quotient.

The remaining dependency rows are package provenance. They allow consumers to compare the controllability packet with observability, Kalman-filter, and regulator packets only through named finite rows, so the resulting surface is a seed NameCert obligation rather than a standard control-theory bridge. □

Theorem 297.8 (ControlControllability consumer boundary)**.** *Every downstream consumer of* ControlControllability$^{\uparrow}$ *can read controllability data only through the displayed* DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, LinMap$^{\uparrow}$, ControlObservability$^{\uparrow}$, KalmanFilter$^{\uparrow}$, LQR$^{\uparrow}$, BHist, hsame, Cont, *and* Pkg *rows of Definition 297.1. No consumer can require analytic trajectory semantics, a host reachable subspace, Kalman decomposition, stochastic optimality, or a standard bridge from this chapter.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket consumer boundary`

Proof. Take a consumer-visible readback from an accepted packet. By the classifier, the readback is invariant only under the displayed hsame transports and the Mat$^{\uparrow}$ column-transport ledger.

Apply the reachability-ledger theorem to exhaust the visible rows. The readback is therefore a finite route through state, input, transition, control, horizon, reachability-column, reachability-matrix, continuation, and package coordinates.

Since none of the excluded host objects appears as a carrier coordinate, a consumer cannot project one from the packet. The boundary is exactly the finite BEDC reachability surface. □

Theorem 297.9 (ControlControllability Kalman reachability source)**.** *The prediction-update and observation-innovation rows imported from* KalmanFilter$^{\uparrow}$ *can be consumed by a* ControlControllability$^{\uparrow}$ *packet only through the displayed state row, transition-matrix row A, control row B, finite horizon row n, reachability columns $B, AB, \ldots, A^{n-1}B$,* Mat$^{\uparrow}$ *and* LinMap$^{\uparrow}$ *column ledger,* hsame *transports,* Cont *provenance, and* Pkg *dependency rows of Definition 297.1. The packet exports no rank theorem, host reachable subspace, stochastic filter semantics, or analytic trajectory.*

Proof. Project the accepted carrier of Definition 297.1. Its Kalman dependency is a finite package row, so any read from the KalmanFilter$^{\uparrow}$ side must pass through the displayed Pkg coordinate and the finite carrier rows named by that coordinate.

Route the prediction-update and observation-innovation rows through the same finite name-certificate surface. The only shared rows available to ControlControllability$^{\uparrow}$ are the state, transition, control, horizon, reachability-column, matrix-ledger, linear-map-ledger, transport, and continuation coordinates.

Apply Definition 297.3 to identify those coordinates as the complete reachability ledger. Apply Definition 297.8 to exclude any readback that would require a rank theorem, a host reachable subspace, stochastic filter semantics, or an analytic trajectory. □

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket kalman reachability source`

Theorem 297.10 (ControlControllability finite continuation closure)**.** *Let two accepted finite reachability-column ledgers for* ControlControllability$^{\uparrow}$ *be displayed over the same* DynSystem$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, LinMap$^{\uparrow}$, ControlObservability$^{\uparrow}$, KalmanFilter$^{\uparrow}$, *and* LQR$^{\uparrow}$ Pkg *rows. If the terminal endpoint row of the first ledger and the initial endpoint row of the second ledger are matched by a displayed* hsame *comparison, then their concatenation repacks as an accepted* ControlControllability$^{\uparrow}$ *reachability packet over those same package rows. The repacked packet asserts no infinite-horizon reachability theorem and no host reachable-subspace quotient.*

Proof. Unpack the two accepted ledgers into their finite reachability-column rows, reachability-matrix rows, Cont routes, and package provenance. Because both ledgers use the same displayed dependency rows, the state, transition, control, horizon, matrix, linear-map, observability, Kalman-filter, and regulator endpoints are read through the same package surface.

Transport the meeting endpoint along the displayed hsame comparison. The first continuation route then lands at the source row expected by the second route, and the two finite Cont records compose as a single displayed continuation ledger.

Reuse the $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ column-transport ledger to append the transported columns in order. Repacking those columns, the composed continuation route, the endpoint transports, and the unchanged package rows gives the accepted carrier shape of Definition 297.1.

The construction is finite at every step. It records only the concatenated displayed reachability packet, so it does not introduce an infinite-horizon object or quotient the packet by a host reachable subspace. □

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket continuation closure`

Theorem 297.11 (ControlControllability mature consumer coverage)**.** *Every mature consumer read from a* $\mathsf{ControlControllability}^{\uparrow}$ *packet factors through exactly the finite carrier and classifier rows of Definition 297.1 and Definition 297.2, the reachability ledger, the column-transport obligation, the classifier-stability obligation, the dependency scope, the consumer boundary, the Kalman reachability source, and the finite continuation closure. The mature catalogue exports no analytic trajectory semantics, rank-completeness theorem, host reachable-subspace quotient, Kalman decomposition, stochastic optimality, or infinite-horizon object.*

Lean4 ✓ `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket mature consumer coverage`

Proof. Project the finite packet to the carrier and classifier rows. The state, input, transition, control, horizon, reachability-column, reachability-matrix, transport, continuation, and package coordinates are the only displayed coordinates available to a mature consumer.

Apply Definition 297.3 and Definition 297.6 to restrict every visible read to those rows and their named dependency certificates. The column-transport and classifier-stability obligations then account for transported reachability columns and componentwise classifier comparisons without introducing an ambient control-theory predicate.

Use Definition 297.8 and Definition 297.9 for downstream reads through the observability, Kalman-filter, and regulator package rows. Use Definition 297.10 for concatenated finite ledgers over the same package surface.

No carrier coordinate names analytic trajectories, rank completeness, a host reachable-subspace quotient, Kalman decomposition, stochastic optimality, or an infinite-horizon object. Since every mature read is already a projection from the finite packet rows above, none of those excluded objects is exported by the mature catalogue. □

Closure status for $\mathsf{NameCert}_{\mathsf{ControlControllability}^{\uparrow}}$.

Constructive story: A $\mathsf{ControlControllability}^{\uparrow}$ packet is generated from a finite BHist reachability ledger. The visible coordinates are the state-carrier row, input-carrier row, transition matrix A, control matrix B, unary horizon row n, reachability columns $B, AB, \ldots, A^{n-1}B$, the reachability-matrix row, rowwise hsame comparisons, Cont provenance for multiplication and column append, and Pkg rows for the dynamics, matrix, vector-space, linear-map, observability, Kalman-filter, and LQR certificates. The classifier compares exactly these rows under displayed transport and the matrix column ledger, so it never quotients by a host reachable subspace. Ledger exactness and source scope coincide because every public reachability read is one of the listed coordinates or a continuation from them. Nonemptiness is an accepted finite reachability packet, and reflexivity, symmetry, and transitivity are assembled componentwise from hsame.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ControlControllability}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridged reachability surface records the finite carrier, classifier, reachability ledger, NameCert obligation surface, consumer boundary, and continuation closure for $\mathsf{ControlControllability}^{\uparrow}$ over displayed $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows.

Formal verification: FormalStatus(NameCert$_{\text{ControlControllability}\uparrow}$, encodedDef)
Lean target: `BEDC.Derived.ControlControllabilityUp.ControlControllabilityReachabilityPacket`
Bridge status: none
Origin: human
Not claimed: This chapter does not close analytic trajectory semantics, Kalman decomposition, stochastic optimality, host reachable subspace quotients, rank-completeness, or a standard bridge.
Upgrade path: No paper-side theory grade remains; Lean verification and any standard bridge remain on the formal axis.

Chapter 298

A Concrete Naming Certificate for $\mathsf{SpectralMeasure}^{\uparrow}$

Following von Neumann and standard functional analysis, the $\mathsf{SpectralMeasure}^{\uparrow}$ interface records the finite BEDC certificate surface for a projection-valued spectral measure over a Hilbert carrier, a bounded-operator row, a projection-family ledger, and finite scalar measure windows. Its traditional target is the projection-valued measure used in the spectral theorem, as in Reed and Simon, *Methods of Modern Mathematical Physics I: Functional Analysis* (1980), or Conway, *A Course in Functional Analysis* (1990). The chapter states only the finite certificate surface consumed by $\mathsf{SpectralTheorem}^{\uparrow}$ and $\mathsf{Observable}^{\uparrow}$: it does not prove a spectral theorem, host sigma-additivity, quotient measure spaces, unrestricted functional calculus, or choice-selected spectral decompositions.

The packet consumes the Hilbert carrier of chapter 91, the bounded operator surface of chapter 1644, the operator-ideal projection boundary of chapter 213, the self-adjoint observable row of chapter 218, the event-carrier and finite-additivity surface of chapter 92, and the spectral-data consumer of chapter 214. These are substantive carrier and ledger dependencies, not background references: without them the spectral-measure packet has no Hilbert endpoint for projections, no bounded-operator row whose projections can act, no operator-ideal surface for projection-family transport, no observable row to measure, and no finite event ledger on which projection additivity can be recorded.

298.1 Carrier definition

Definition 298.1 (SpectralMeasureUp carrier)**.** *A* $\mathsf{SpectralMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (H, B, \Pi, \mu, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row over which all projection endpoints are typed.*
2. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row carrying the displayed self-adjoint or normal operator endpoint whose spectral windows are being named.*
3. *Π is the projection-family ledger: each visible event row is assigned a projection endpoint over the same H, with orthogonality and finite-additivity rows for disjoint displayed events.*

4. *μ is the scalar* $\mathsf{Real}^{\uparrow}$ *measure-window row read from finite event probes, recording only finite scalar windows attached to the same projection family.*
5. *T is the classifier-transport row, comparing Hilbert, bounded-operator, event, projection, and scalar-window endpoints only by displayed* hsame *rows.*
6. *C records displayed* Cont *replay from event rows through projection endpoints and scalar* $\mathsf{Real}^{\uparrow}$ *measure windows.*
7. *P is the* Pkg *provenance over* $\mathsf{SpectralMeasure}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{SpectralTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *row.*

The carrier exports no ambient Hilbert-space equality, host sigma-additivity, quotient measure space, unrestricted functional calculus, choice-selected spectral decomposition, countable family selector, eigenbasis, or external spectral-theorem object.

Definition 298.2 (SpectralMeasureUp classifier)**.** *For accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packets S and S', the classifier $S \sim_{\mathsf{SpectralMeasure}^{\uparrow}} S'$ holds exactly when the displayed* $\mathsf{Hilbert}^{\uparrow}$ *carrier rows,* $\mathsf{Observable}^{\uparrow}$ *rows,* $\mathsf{Measure}^{\uparrow}$ *event rows, projection endpoints, orthogonality and finite-additivity ledgers,* Cont *routes, and* Pkg *provenance compare componentwise by their recorded* hsame *rows. The classifier does not compare completed measurable spaces or host projection-valued functions; it compares only the finite rows visible in the certificate.*

Theorem 298.3 (SpectralMeasureUp public consumer boundary)**.** *Every downstream* $\mathsf{SpectralTheorem}^{\uparrow}$ *or* $\mathsf{Observable}^{\uparrow}$ *consumer of an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet may read only the finite* $\mathsf{Hilbert}^{\uparrow}$ *carrier, the observable row, the* $\mathsf{Measure}^{\uparrow}$ *event row, the projection endpoints, the orthogonality and finite-additivity ledger, the displayed* hsame *and* Cont *rows, and the* Pkg *provenance. Such a consumer cannot read countable additivity, completed sigma-algebras, host projection-valued functions, eigenbasis choices, or a spectral theorem proof.*

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFinitePacket public consumer boundary`

Proof. Project the carrier of Definition 298.1. Its exported coordinates are exactly the Hilbert carrier, observable row, MeasureUp event row, projection endpoints, finite ledger, transport rows, and provenance rows.

Then project the classifier of Definition 298.2. It compares only those displayed rows by their recorded hsame witnesses and does not introduce a completed sigma-algebra, host projection-valued function, eigenbasis selector, or spectral theorem object.

The public consumer boundary is therefore the finite list of exported rows. Every downstream $\mathsf{SpectralTheorem}^{\uparrow}$ or $\mathsf{Observable}^{\uparrow}$ read factors through that list and has no coordinate for the excluded host-level data. □

Theorem 298.4 (SpectralMeasureUp finite functional-calculus consumer route)**.** *Every functional-calculus consumer of an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet must expose the* $\mathsf{Hilbert}^{\uparrow}$ *source,* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *endpoint,* $\mathsf{Observable}^{\uparrow}$ *row,* $\mathsf{Measure}^{\uparrow}$ *event row, finite projection ledger, scalar* $\mathsf{Real}^{\uparrow}$ *windows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *row before it reads spectral-theorem data. The route is finite and cannot bypass through host sigma-additivity, completed measurable spaces, unrestricted functional calculus, quotient measure equality, selected decompositions, or eigenbasis choices.*

Proof. Start with the carrier and classifier of Definition 298.1 and Definition 298.2. They expose only the Hilbert source, observable row, Measure event row, projection endpoints, orthogonality and finite-additivity ledgers, transport, replay, provenance, and local naming coordinates.

Apply the public consumer boundary of Definition 298.3. A calculus-facing read must therefore enter through those displayed rows before it can use any spectral-theorem-facing packet.

Use the finite-additivity ledger of Definition 298.7. It keeps the projection endpoint for each finite displayed event union tied to the same $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, and $\mathsf{Measure}^{\uparrow}$ provenance rows.

Finally apply the projection-ledger nonescape theorem, Definition 298.10. Since the packet contains no completed sigma-algebra, host projection-valued function, quotient measure equality, selected decomposition, or eigenbasis coordinate, the functional-calculus consumer route is exhausted by the finite BEDC rows named above. □

298.2 The certificate

Theorem 298.5 (SpectralMeasureUp hsame stability)**.** *If accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packets S and S' are related by the classifier of Definition 298.2, then transporting the Hilbert, observable, event, projection, orthogonality, finite-additivity,* Cont*, and* Pkg *rows of S along the displayed* hsame *witnesses yields the corresponding accepted rows of S'. In particular, downstream spectral-theorem and observable consumers may replace a carried spectral-measure packet by a classified packet without reading any unlogged measurable or operator data.*

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureCarrier hsame stability`

Proof. Project the finite carrier of Definition 298.1. Each accepted row is one of the displayed Hilbert, observable, event, projection, ledger, continuation, or provenance coordinates.

Apply the classifier definition componentwise. The hsame witnesses transport each source coordinate to its partner coordinate, and the Cont rows record the same event-to-projection readback after transport.

Since the carrier has no coordinate for completed measurable spaces, host projection-valued functions, or eigenbasis choices, the transported packet is exhausted by the displayed rows and remains an accepted $\mathsf{SpectralMeasure}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasurePacket hsame stability`

Theorem 298.6 (SpectralMeasureUp orthogonal finite additivity)**.** *Let S be an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet and let E and F be displayed event rows in its* $\mathsf{Measure}^{\uparrow}$ *event carrier. If the packet ledger records E and F as disjoint displayed events, then the projection endpoint row for the displayed union event is the finite-additivity endpoint obtained from the two projection endpoint rows, and the orthogonality row records that the two projections are compatible as Hilbert projections over the same* $\mathsf{Hilbert}^{\uparrow}$ *carrier. This is a finite ledger statement and not a countable-additivity theorem over a completed sigma-algebra.*

Proof. Read the two event rows and their disjointness witness from the $\mathsf{Measure}^{\uparrow}$ side of the carrier. The projection endpoint rows for E and F are displayed Hilbert projection rows with the same $\mathsf{Hilbert}^{\uparrow}$ provenance.

The orthogonality ledger supplies the compatibility row for the two projection endpoints. The finite-additivity ledger then names the union-event projection endpoint as the endpoint obtained from those two displayed projection rows.

No countable family is introduced. The proof sketch uses only the finite event pair present in the packet and the ledger rows already displayed by the carrier. □

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureOrthogonalFiniteAdditivity row`

Theorem 298.7 (SpectralMeasureUp projection-ledger finite additivity)**.** *Let S be an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet and let $E_1, \ldots, E_n$ be a finite displayed family of pairwise-disjoint event rows in its* $\mathsf{Measure}^{\uparrow}$ *event carrier. If the packet also displays the finite-union event row $E_1 \cup \cdots \cup E_n$, then every projection endpoint row assigned to that union is obtained by iterating the packet's orthogonality and finite-additivity ledger over the same* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, *and* $\mathsf{Measure}^{\uparrow}$ *provenance rows. The read uses no countable additivity, completed sigma-algebra data, eigenbasis choice, or host projection-valued function.*

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasurePacket projection ledger finite additivity`
Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFinitePacket projection ledger transport`

Proof. Inspect the carrier and classifier of Definition 298.1 and Definition 298.2. The only available event-to-projection data are the displayed event rows, projection endpoint rows, orthogonality ledger, finite-additivity ledger, continuation routes, and source packages for $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, and $\mathsf{Measure}^{\uparrow}$.

Proceed by induction over the finite ledger for the displayed union. The empty or singleton ledger reads back the projection endpoint already stored for the displayed event row.

For the binary step, apply Definition 298.6 to the next disjoint event row and the accumulated displayed union row. The orthogonality row supplies compatibility over the same Hilbert carrier, and the finite-additivity row names the next projection endpoint without changing the observable or measure provenance.

The packet has no coordinate for countable families, completed measurable spaces, eigenvectors, or external projection-valued functions. Thus the finite-union endpoint is exhausted by the projection ledger recorded in the accepted $\mathsf{SpectralMeasure}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFiniteAdditivityLedger projection chain`
Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFinitePacket projection ledger additivity`

Theorem 298.8 (SpectralMeasureUp functional-calculus handoff)**.** *Every functional-calculus or spectral-theorem consumer of an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet reaches projection-valued-measure data only through the displayed route*

$$H, B, \Pi, \mu, T, C, P, N.$$

Thus the consumer must pass through the $\mathsf{Hilbert}^{\uparrow}$ *row, the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row, the projection-family ledger, the finite scalar* $\mathsf{Real}^{\uparrow}$ *measure windows, classifier transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *row before it reads calculus data. It may not read host sigma-additivity, quotient measure spaces, unrestricted functional calculus, selected spectral decompositions, or eigenbasis choices.*

Proof. Start with the carrier of Definition 298.1. Its projection-valued-measure surface is not a host measure object; it is the finite row package $H, B, \Pi, \mu, T, C, P, N$ typed over the Hilbert source, bounded-operator endpoint, projection ledger, scalar windows, transport, replay, provenance, and local name certificate.

Apply the NameCert obligation surface of Definition 298.9. That theorem identifies the rows that every $\mathsf{SpectralMeasure}^{\uparrow}$ consumer may inspect, including the projection-family ledger and finite scalar windows, and it supplies no coordinate for completed sigma-algebras, quotient measure spaces, selected decompositions, or eigenbasis data.

The downstream chapters use the same boundary. The Dunford functional-calculus packet of chapter 3740 records contour and resolvent routes before any spectral projection or spectral measure is public, while the spectral-theorem packet of chapter 214 reads spectrum, projection-valued-measure, and calculus rows over certified Hilbert and measure sources.

Therefore a calculus-facing or spectral-theorem-facing read of a $\mathsf{SpectralMeasure}^{\uparrow}$ packet can only enter through the displayed rows. Since none of the excluded host objects appears in that route, the handoff delivers finite BEDC calculus data without importing host sigma-additivity, unrestricted functional calculus, choice-selected spectral decompositions, quotient measure spaces, or eigenbasis choices. □

Theorem 298.9 (SpectralMeasureUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *obligation surface consists of the carrier of Definition 298.1, the classifier of Definition 298.2, the* hsame *stability theorem, the orthogonal finite-additivity ledger, the projection-family ledger, the scalar* $\mathsf{Real}^{\uparrow}$ *measure windows, and the provenance rows naming* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *and* $\mathsf{SpectralTheorem}^{\uparrow}$. *Every consumer of* $\mathsf{SpectralMeasure}^{\uparrow}$ *can read only those finite* BHist, hsame, Cont, Pkg, *and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFinitePacket namecert obligation surface`

Proof. Start with the carrier and classifier definitions. They expose the Hilbert source, observable row, event carrier, projection endpoints, finite ledger, transport rows, and source provenance as the whole packet.

Use Definition 298.5 for classified replacement of a packet and Definition 298.6 for the finite disjoint-event ledger. These two rows are the visible stability and exactness obligations needed by spectral-data consumers.

Finally repack the displayed rows as the local naming-certificate surface. Because no spectral theorem proof, completed sigma-algebra, countable additivity row, or eigenbasis row is part of the carrier, the obligation surface stops at the finite BEDC certificate boundary. □

Theorem 298.10 (SpectralMeasureUp projection-ledger nonescape)**.** *Every* $\mathsf{SpectralTheorem}^{\uparrow}$, *bounded-operator, or observable consumer of an accepted* $\mathsf{SpectralMeasure}^{\uparrow}$ *packet reads the projection-family ledger only through*

$$H, B, \Pi, \mu, T, C, P, N.$$

The read therefore passes through the displayed $\mathsf{Hilbert}^{\uparrow}$ *source,* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row, projection-family ledger, scalar* $\mathsf{Real}^{\uparrow}$ *measure windows, classifier transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *row before it exports spectral data. It cannot expose host sigma-additivity, quotient measure spaces, unrestricted functional calculus, choice-selected spectral decompositions, host projection-valued functions, or eigenbasis choices.*

Proof. Project the carrier of Definition 298.1. The only projection-facing coordinates are $H, B, \Pi, \mu, T, C, P, N$.

The projection-family ledger Π is typed over the same Hilbert source H and bounded-operator row B, and the scalar measure windows μ are read only through finite $\mathsf{Real}^{\uparrow}$ windows attached to visible event probes. Thus the consumer receives finite projection and scalar-window data, not a completed host measure space.

The classifier and replay rows close the boundary. The row T transports only the displayed Hilbert, operator, event, projection, and scalar-window endpoints, while C, P, N replay, source, and name that finite route. Since no excluded object appears among these rows, the projection-ledger read has no route to host sigma-additivity, quotient measure spaces, unrestricted functional

calculus, choice-selected spectral decompositions, host projection-valued functions, or eigenbasis choices. □

Theorem 298.11 (SpectralMeasureUp finite-packet standard bridge)**.** *Every standard projection-valued spectral-measure presentation admitted into BEDC only through the displayed finite rows*

$$H, B, \Pi, \mu, T, C, P, N$$

bridges conservatively to the public $\mathsf{SpectralMeasure}^{\uparrow}$ *packet and back. The bridge preserves the Hilbert source, observable row,* $\mathsf{Measure}^{\uparrow}$ *event row, projection-family endpoints, orthogonality ledger, finite-additivity ledger, scalar* $\mathsf{Real}^{\uparrow}$ *measure windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ *row. Conversely, every public* $\mathsf{SpectralMeasure}^{\uparrow}$ *consumer reads the standard presentation only through those rows.*

Proof. Start with the finite carrier and classifier of Definition 298.1 and Definition 298.2. They expose exactly the Hilbert, bounded-operator, projection-family, scalar-window, classifier, replay, provenance, and local naming rows used by the public packet.

Apply the public consumer boundary of Definition 298.3. It says that consumers of the public packet can inspect only the displayed BHist, hsame, Cont, Pkg, and NameCert rows, so transporting a standard presentation into BEDC cannot add a second projection-valued surface.

Use the finite-additivity ledger of Definition 298.7 and the projection-ledger nonescape theorem Definition 298.10. These identify the orthogonality, finite-additivity, event, projection, and scalar-window data as finite rows and block any route to countable additivity, completed sigma-algebras, host projection-valued functions, eigenbasis selection, or a spectral theorem proof.

The reverse direction repacks the same rows as the public $\mathsf{SpectralMeasure}^{\uparrow}$ packet. Since both directions read and write only $H, B, \Pi, \mu, T, C, P, N$, the bridge is conservative over the displayed finite packet rather than a host spectral-measure construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$, publicClosed)

Scope closed: The scoped packet names the finite BHist carrier, classifier, hsame stability, orthogonal finite-additivity ledger, Cont transport, Pkg provenance, $\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$ obligation surface, and the public consumer boundary of Definition 298.3 over $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Observable}^{\uparrow}$, and $\mathsf{Measure}^{\uparrow}$.

Constructive story: A spectral-measure packet starts from a displayed Hilbert carrier, observable row, and measurable-event carrier, then records finite event-to-projection endpoints, orthogonality and finite-additivity ledgers, componentwise hsame transport, and continuation provenance as the visible certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpectralMeasure}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SpectralMeasureUp.SpectralMeasureFinitePacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close the spectral theorem, countable additivity over completed sigma-algebras, host projection-valued measures, eigenbasis selection, or standard bridges.

Upgrade path: Bridged closure requires a standard bridge theorem from the finite spectral-measure public surface to the usual projection-valued spectral-measure boundary.

Chapter 299

A Concrete Naming Certificate for $\mathsf{Holonomy}^{\uparrow}$

The $\mathsf{Holonomy}^{\uparrow}$ interface records the finite certificate surface of parallel transport around displayed loops in a certified bundle with a certified connection. The traditional geometric target is the holonomy construction of Kobayashi and Nomizu, *Foundations of Differential Geometry*, read here only through $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ rows. The finite transport ledger consumes the certified connection surface of chapter 134, the curvature boundary of chapter 135, and the endpoint-composition rows of chapter 140 before any holonomy packet is read.

299.1 Carrier definition

Definition 299.1 (Holonomy BHist transport carrier)**.** *A* $\mathsf{Holonomy}^{\uparrow}$ *carrier is a finite* BHist *transport packet*

$$(\mathcal{B}, \mathcal{C}, \ell, e, \kappa, \lambda, \rho)$$

whose rows are as follows.

1. $\mathcal{B}$ *is a certified bundle row supplied by* $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$ *over a base row supplied by* $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$.
2. $\mathcal{C}$ *is a certified connection row supplied by* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$ *on the same displayed bundle rows.*
3. ℓ *is a loop row whose source and target base endpoint are the same displayed* $\mathsf{Manifold}^{\uparrow}$ *row after the carried* hsame *comparison.*
4. e *is the transported fibre endpoint row obtained by reading the connection transport ledger along* ℓ.
5. κ *records the curvature and control provenance rows supplied through* $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ *and the finite control ledger governing the displayed transport route.*
6. λ *is the finite* Cont *ledger for loop subdivision, transport composition, endpoint readback, and provenance projection.*
7. ρ *is the* Pkg *provenance row naming the bundle, connection, curvature, manifold, and Lie-group dependency certificates.*

Two such carriers classify together when their base loop, connection row, transported fibre endpoint, curvature-control provenance, continuation ledger, and package rows are related componentwise by the available hsame *transports. The carrier exports no host parallel-transport functor, analytic path space, smooth loop manifold, or ambient holonomy group outside these rows.*

Definition 299.2 (Holonomy finite transport classifier)**.** *The* $\mathsf{Holonomy}^{\uparrow}$ *classifier relates two accepted packets exactly when their base loop, connection, transported fibre endpoint, curvature-control provenance, bundle provenance, manifold provenance,* Cont*, and* Pkg *coordinates are related componentwise by displayed* hsame *rows. The classifier exports only this finite componentwise comparison.*

Lean4 (definition) `BEDC.Derived.HolonomyUp.HolonomyFiniteTransportClassifier`

299.2 The certificate

Theorem 299.3 (Holonomy loop source obligation)**.** *Every accepted* $\mathsf{Holonomy}^{\uparrow}$ *packet reads its loop source from the displayed* $\mathsf{Manifold}^{\uparrow}$ *and* $\mathsf{Bundle}^{\uparrow}$ *rows of Definition 299.1. The loop row* ℓ *names its base endpoint, its return endpoint, the carried* hsame *comparison between those endpoints, and the bundle fibre row over that base; no ambient loop space or unlogged base path is part of the source obligation.*

Proof. Project the carrier packet to the rows $\mathcal{B}$ and ℓ. The bundle row $\mathcal{B}$ already exposes the base row through $\mathsf{NameCert}_{\mathsf{Bundle}^{\uparrow}}$, and that base row is read through $\mathsf{NameCert}_{\mathsf{Manifold}^{\uparrow}}$.

The loop row ℓ then names the source endpoint, return endpoint, and endpoint hsame comparison over the same base row. These projections are finite BHist data, so the source obligation is exhausted by the displayed manifold and bundle certificate rows.

Since the carrier contains no coordinate for an analytic path space or a host loop object, a consumer cannot read anything beyond those projected loop and bundle rows. □

Theorem 299.4 (Holonomy continuation transport stability)**.** *If two* $\mathsf{Holonomy}^{\uparrow}$ *packets have componentwise* hsame *transports of their base loop rows, certified connection rows, transported fibre endpoints, curvature-control provenance rows, and finite continuation ledgers, then the transported packet remains accepted by the carrier of Definition 299.1. The fibre endpoint readback is stable under that transport and remains tied to the same* $\mathsf{Connection}^{\uparrow}$ *source row.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket loop source obligation`
Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyBHistTransportCarrier continuation transport stability`

Proof. Unpack the two packets and read the supplied componentwise comparisons. The loop comparison transports the displayed base endpoints, while the connection comparison transports the row that supplies the parallel-transport ledger.

Next transport the fibre endpoint and curvature-control provenance rows. These fields are already listed in the carrier, so the transported endpoint and provenance remain inside the same finite packet shape.

Finally compare the Cont ledgers. The continuation rows identify the same subdivision and readback routes after transport, and the package provenance rows match the dependency certificates. Repacking the transported fields gives an accepted $\mathsf{Holonomy}^{\uparrow}$ packet with the stated stable endpoint readback. □

Theorem 299.5 (Holonomy parallel-transport stability)**.** *Let H and H' be accepted* Holonomy$^{\uparrow}$ *packets. If the base loop, connection, transported fibre endpoint, curvature-control provenance,* Cont*, and* Pkg *rows of H are transported to the corresponding rows of H' by displayed* hsame *comparisons, then the transported packet remains inside the same finite* Holonomy$^{\uparrow}$ *carrier surface. The endpoint readback of H' is exactly the transported finite parallel-transport loop row.*

Proof. Project the carrier of Definition 299.1. The parallel-transport component is a finite spine read from Connection$^{\uparrow}$ together with the base, fibre, loop, continuation, and package rows.

Transport those coordinates componentwise along the displayed hsame comparisons. The classifier of Definition 299.2 accepts precisely that componentwise transport, so the resulting tuple has the same carrier shape.

Finally read back the endpoint row. Since the carrier has no coordinate for an ambient path space or external connection object, the endpoint readback is exhausted by the transported finite loop spine. □

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket parallel transport stability`

Theorem 299.6 (Holonomy curvature-loop boundary)**.** *Every curvature contribution visible in an accepted* Holonomy$^{\uparrow}$ *packet is read through the displayed certified loop row, the* Curvature$^{\uparrow}$ *boundary row, the* Connection$^{\uparrow}$ *parallel-transport spine, the* Bundle$^{\uparrow}$ *provenance row, the* Manifold$^{\uparrow}$ *provenance row,* hsame *transport, and the finite* Cont *ledger. No downstream consumer can insert an unlogged curvature tensor, ambient surface integral, or host smooth-loop family.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket curvature loop boundary`
Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportCarrier curvature loop boundary`

Proof. Read the curvature coordinate from the packet. It is attached to a certified loop row and to the parallel-transport spine already named by the connection source.

Use the displayed Curvature$^{\uparrow}$ boundary row to account for the curvature data along that loop. The bundle and manifold provenance rows identify the source of the fibre and base coordinates, while the Cont ledger records the finite route to the endpoint readback.

Because the packet contains no host tensor, surface integral, or smooth-loop family coordinate, the curvature contribution is exactly the displayed boundary row together with its loop and transport provenance. □

Theorem 299.7 (Holonomy composition ledger obligation)**.** *Composition of displayed holonomy transports is recorded only by finite* Cont *rows. For two accepted loop rows whose endpoint ledgers concatenate inside the same bundle and connection packet, the composed fibre endpoint is the endpoint reached by the concatenated continuation ledger, with dependency provenance retained in ρ.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket composition ledger obligation`

Proof. Start with the two accepted loop rows and their transport ledgers. Each ledger entry names a finite source endpoint, target endpoint, connection row, and fibre endpoint readback.

Concatenate the displayed continuation routes using the Cont ledger carried by the packet. The intermediate endpoint is matched by the endpoint hsame row, so the second transport can read the fibre endpoint produced by the first.

The composed endpoint is therefore just the endpoint of the concatenated finite ledger. The package row ρ records that the bundle, connection, manifold, curvature, and Lie-group dependencies are the same named certificate rows throughout the composition. □

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportCarrier composition ledger obligation`

Theorem 299.8 (Holonomy NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{Holonomy}^\uparrow}$ *seed obligation surface consists of five rows over* $\mathsf{Connection}^\uparrow$, $\mathsf{Curvature}^\uparrow$, $\mathsf{Bundle}^\uparrow$, $\mathsf{Manifold}^\uparrow$, *and* $\mathsf{LieGroup}^\uparrow$.

1. *The loop-source row supplied by Definition 299.3.*
2. *The connection-transport row supplied by* $\mathsf{NameCert}_{\mathsf{Connection}^\uparrow}$ *and stabilized by Definition 299.4.*
3. *The composition-ledger row supplied by Definition 299.7.*
4. *The curvature-compatibility row saying that curvature and control provenance are read through* $\mathsf{NameCert}_{\mathsf{Curvature}^\uparrow}$ *and never through an external curvature operator.*
5. *The finite naming ledger row saying that* $\mathsf{NameCert}_{\mathsf{Bundle}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{Manifold}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{LieGroup}^\uparrow}$ *are consumed only through displayed* BHist, hsame, Cont, *and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket namecert obligation surface`

Proof. Read the carrier packet of Definition 299.1. Its fields list the bundle, connection, loop, transported endpoint, curvature-control provenance, continuation, and package rows.

Apply Definition 299.3 to bind the loop source to the displayed manifold and bundle rows. Apply Definition 299.4 to bind transport and endpoint readback to the connection row.

Apply Definition 299.7 to record composition as a finite continuation route. The curvature-compatibility and naming rows are then projections from κ, λ, and ρ: every curvature, control, Lie-group, bundle, and manifold dependency appears as a named row in the packet.

These five rows exhaust the public seed surface. No host parallel-transport functor, analytic path space, ambient holonomy group, or unrecorded curvature operator has a projection from the carrier. □

Theorem 299.9 (Holonomy certificate boundary)**.** *The seed* $\mathsf{NameCert}_{\mathsf{Holonomy}^\uparrow}$ *obligation surface consists of five finite rows: carrier inhabitation for Definition 299.1, reflexivity, symmetry, and transitivity for the classifier of Definition 299.2, transport stability of finite parallel-transport loops, exactness of the curvature-boundary ledger, and a downstream readback boundary restricting consumers to the displayed loop, transport, curvature, bundle, manifold,* hsame, Cont, *and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyBHistTransportCarrier certificate boundary`

Proof. Carrier inhabitation is the constructor that assembles the base, fibre, loop, transport, curvature, provenance, and continuation rows into one finite BHist packet.

The classifier laws are componentwise. Reflexivity uses the displayed row identities, symmetry reverses each hsame comparison, and transitivity composes the corresponding comparisons while preserving the same finite continuation ledger.

The transport and curvature obligations are supplied by the preceding transport and curvature-boundary theorems. The final readback boundary follows by projecting the carrier coordinates: no accepted packet exposes an ambient holonomy group, host smooth-loop space, external transport solver, or extra curvature source. □

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket certificate boundary`

Theorem 299.10 (Holonomy standard boundary scope)**.** *The standard geometric boundary of* $\mathsf{Holonomy}^{\uparrow}$ *is limited in this chapter to the displayed loop, connection-transport, fibre endpoint, curvature-control, composition, and dependency-provenance rows of the carrier. A classical holonomy group may motivate the target name, but the BEDC packet exposes only a finite NameCert ledger and does not claim a host Lie subgroup, smooth parallel-transport action, or analytic path-space quotient.*

Lean4 ✓ `BEDC.Derived.HolonomyUp.HolonomyTransportPacket standard boundary scope`

Proof. Project a consumer-visible holonomy readback from the carrier. The available fields are exactly the loop row, connection row, transported endpoint, curvature-control provenance, continuation ledger, and package provenance.

The Lie-group dependency is present only through the named $\mathsf{NameCert}_{\mathsf{LieGroup}^{\uparrow}}$ row in ρ. It can classify displayed endpoint-composition data, but it does not create an ambient group of all loops or a smooth action on a host fibre.

Thus the standard boundary remains a scope statement about which classical rows can be consumed from the finite packet. Anything requiring a host path space, host parallel transport, or Lie-subgroup theorem lies outside this seed chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{Holonomy}^{\uparrow}}$**.**

Constructive story: The holonomy packet is a finite BHist transport ledger over certified bundle, connection, loop, transported fibre endpoint, curvature-control, continuation, and package rows. Classification transports the loop, connection, endpoint, curvature-control provenance, and ledger rows componentwise by hsame, while composition is a finite Cont route rather than a host path operation.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Holonomy}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope names only the displayed $\mathsf{Connection}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{LieGroup}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows of the transport packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Holonomy}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.HolonomyUp.HolonomyTransportPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: This chapter does not close host parallel transport, analytic path spaces, a holonomy Lie subgroup theorem, or a standard bridge.

Upgrade path: No paper-axis theory grade remains; bridgeChecked verification is a separate formal-axis upgrade.

Chapter 300

A Concrete Naming Certificate for $\mathsf{KKT}^{\uparrow}$

The $\mathsf{KKT}^{\uparrow}$ interface records the finite certificate surface of Karush–Kuhn–Tucker optimality data for a displayed constrained convex program. Its standard optimization source is Boyd and Vandenberghe, *Convex Optimization* (2004). The BEDC dependencies are $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ for the feasible-region row, $\mathsf{NameCert}_{\mathsf{LPDuality}^{\uparrow}}$ for primal-dual comparison rows, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ and $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ for stationarity and residual rows, and $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ for scalar slack and objective rows. It consumes the feasible-region carrier of chapter 208 and the primal-dual comparison ledger of chapter 235 before packaging stationarity, feasibility, and complementary-slackness rows.

300.1 Carrier definition

Definition 300.1 (KKTUp BHist carrier)**.** *A* $\mathsf{KKT}^{\uparrow}$ *carrier is a finite* BHist *packet whose visible rows are a primal variable row, a dual multiplier row, a constraint residual row, a stationarity row, feasibility rows, a complementary-slackness ledger,* hsame *comparison rows between corresponding displayed entries,* Cont *provenance rows for the stationarity and ledger readbacks,* Pkg *rows naming* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{LPDuality}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and a* NameCert *provenance row for the accepted packet.*

The classifier accepts exactly those packets whose primal, dual, residual, stationarity, feasibility, slackness, comparison, continuation, and package rows are read through the displayed dependency certificates. It exports no nonlinear optimizer, differentiability oracle, Slater witness, analytic duality theorem, or host real-valued minimization semantics.

Lean4 (definition) `BEDC.Derived.KKTUp.KKTCarrierPacket`

300.2 The certificate

Theorem 300.2 (KKTUp primal-dual row obligations)**.** *Every accepted* $\mathsf{KKT}^{\uparrow}$ *packet exposes a finite primal row, dual multiplier row, constraint residual row, stationarity row, and feasibility row. The primal and feasibility rows are read through* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$*; the dual and residual rows are read through* $\mathsf{LPDuality}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; and every row carries its displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualCarrier primal dual row obligations`

Proof. Project the carrier of Definition 300.1. Its accepted packet lists exactly the primal, multiplier, residual, stationarity, feasibility, comparison, continuation, package, and naming rows.

Read each row through its named dependency certificate. The feasible-region row supplies the convex-set face, the linear and vector rows supply the displayed stationarity and residual endpoints, the real rows supply scalar slack and objective endpoints, and the duality row supplies the primal-dual comparison.

No other row is present in the packet. The accepted primal-dual surface is therefore exhausted by the finite BHist rows and their visible provenance. □

Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualPacket row obligations`

Theorem 300.3 (KKTUp stationarity and feasibility hsame transport)**.** *Let K and K' be finite* $\mathsf{KKT}^{\uparrow}$ *packets whose primal variable, dual multiplier, residual, stationarity, feasibility, and slackness rows are related componentwise by the displayed* hsame, Cont, Pkg, *and* NameCert *rows of the* $\mathsf{KKT}^{\uparrow}$ *classifier. Suppose the same transport also respects the dependency rows over* $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *If K is accepted, then K' is accepted on the same stationarity and feasibility surface, and the complementary-slackness ledger remains confined to the finite boundary of Definition 300.5.*

The transported packet introduces no analytic differentiability row, Slater data, host minimization predicate, or solver semantics beyond the displayed BEDC rows.

Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualCarrier stationarity feasibility hsame transport`
Lean4 ✓ `BEDC.Derived.KKTUp.KKTCarrierPacket stationarity feasibility hsame transport`

Proof. Unfold Definition 300.1. The carrier names exactly the primal, dual multiplier, residual, stationarity, feasibility, slackness, comparison, continuation, package, and naming rows that can participate in the classifier.

Apply Definition 300.2. The primal and feasibility rows are read through $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$, while the multiplier, residual, stationarity, slackness, and scalar rows are read through $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Transport each displayed row through the given componentwise hsame, Cont, Pkg, and NameCert data, then repack the resulting finite rows in the same carrier positions. Since the dependency rows are transported with the visible packet, the stationarity and feasibility surface of K' is accepted by the same classifier shape.

Finally apply Definition 300.5. The slackness ledger remains only the displayed residual, multiplier, scalar slack, comparison, continuation, and package rows, so the transport cannot add analytic differentiability, Slater data, host minimization semantics, or solver evidence. □

Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualPacket stationarity feasibility hsame transport`

Theorem 300.4 (KKTUp stationarity and feasibility stability)**.** *Let K and K' be accepted* $\mathsf{KKT}^{\uparrow}$ *packets whose displayed primal variable, dual multiplier, residual, matrix and vector stationarity rows, real scalar rows, finite* Cont *routes, and* Pkg *provenance compare componentwise by the* hsame *rows of the* $\mathsf{KKT}^{\uparrow}$ *classifier. Transporting along those rows preserves the accepted stationarity and feasibility rows of the packet. The transport introduces no analytic differentiability oracle, Slater witness, nonlinear optimizer, or host minimization semantics.*

Lean4 ✓ `BEDC.Derived.KKTUp.KKTCarrierPacket stationarity feasibility stability`

Proof. Unpack the carrier of Definition 300.1. The stationarity and feasibility coordinates are displayed rows in the same finite BHist packet as the primal variable, multiplier, residual, scalar, continuation, package, and naming rows.

Apply Definition 300.2 to read the dependency provenance of those coordinates. The primal and feasibility rows are routed through the convex-set and vector-space certificates, while the dual, residual, stationarity, and scalar rows are routed through the displayed linear, vector, real, and duality certificates.

Componentwise hsame transport replaces each displayed row by its classified partner and keeps the finite Cont routes and Pkg provenance fixed at the readback boundary. Therefore the stationarity and feasibility rows remain accepted rows of K'.

The carrier has no field for a differentiability oracle, Slater witness, nonlinear optimizer, or host real minimization semantics. The stability claim is therefore confined to the displayed BEDC packet rows. □

Theorem 300.5 (KKTUp complementarity ledger exactness)**.** *In an accepted* $\mathsf{KKT}^{\uparrow}$ *packet, the complementary-slackness ledger is exactly the finite list of displayed residual, multiplier, scalar slack,* hsame*,* Cont*, and* Pkg *rows carried by Definition 300.1. A downstream read of complementarity cannot use an unrecorded inequality family, host multiplication law, analytic limit, or external optimality predicate.*

Lean4 ✓ `BEDC.Derived.KKTUp.KKTComplementarityLedger exactness`

Proof. Inspect the complementary-slackness coordinate of the carrier. It is a finite ledger, and each entry points to the displayed residual row, multiplier row, scalar row, comparison row, continuation row, and package row.

The ledger readback is obtained by following those entries and no ambient arithmetic table is introduced. The real-valued component is the named $\mathsf{Real}^{\uparrow}$ endpoint already present in the packet.

Since the carrier has no coordinate for an external inequality family, analytic limit, or host optimality predicate, complementarity is exact at the displayed ledger boundary. □

Theorem 300.6 (KKTUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{KKT}^{\uparrow}}$ *obligation surface consists of five finite rows: the carrier and classifier row of Definition 300.1; the primal-dual dependency row over* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{LPDuality}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; the stationarity and feasibility stability row under componentwise* hsame *transport; the complementary-slackness ledger exactness row; and the downstream consumer boundary that restricts all reads to the displayed* BHist*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.KKTUp.KKTPrimalDualPacket namecert obligation surface`

Proof. Start from the finite carrier and classifier. These give the packet inventory and the accepted readback relation for every visible row.

Apply Definition 300.2 to bind the primal, dual, residual, stationarity, and feasibility rows to their dependency certificates. Componentwise hsame transport preserves the same row inventory because the classifier reads only those displayed comparisons and the same provenance rows.

Apply Definition 300.5 for the slackness face. Repacking the dependency rows, transport rows, ledger row, and consumer restriction gives exactly the five local $\mathsf{NameCert}_{\mathsf{KKT}^{\uparrow}}$ obligations. □

Theorem 300.7 (KKTUp downstream consumer boundary)**.** *Any optimization, algebraic, or control consumer of* $\mathsf{NameCert}_{\mathsf{KKT}^\uparrow}$ *may read only the finite primal, multiplier, residual, stationarity, feasibility, slackness,* hsame, Cont, Pkg, *and* NameCert *rows of the carrier. It cannot infer full nonlinear optimization semantics, Slater existence, analytic differentiability, host real minimization, or a solver certificate from the* $\mathsf{KKT}^\uparrow$ *packet alone.*

Lean4 ✓ `BEDC.Derived.KKTUp.KKTCarrierPacket downstream consumer boundary`

Proof. Take an arbitrary downstream read of the certificate. By the local obligation surface, the read must factor through the displayed carrier and its finite provenance rows.

The only optimization-facing rows in that carrier are the primal, multiplier, residual, stationarity, feasibility, and complementary-slackness entries. Each is already tied to a dependency certificate and to a visible continuation route.

All excluded analytic objects require rows absent from the carrier. The downstream boundary therefore stops at the finite BEDC packet and cannot project host optimization semantics or solver evidence. □

Theorem 300.8 (KKTUp public finite-packet export)**.** *The public export for* $\mathsf{KKT}^\uparrow$ *is exactly the finite primal, multiplier, residual, stationarity, feasibility, complementary-slackness,* hsame, Cont, Pkg, *and* NameCert *rows of Definition 300.1. Its dependency reads pass only through* $\mathsf{ConvexSet}^\uparrow$, $\mathsf{LPDuality}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, *and* $\mathsf{Real}^\uparrow$ *rows already recorded in the carrier and its local certificate surface.*

The export contains no full nonlinear optimization semantics, Slater existence, analytic differentiability, host minimization predicate, or solver certificate.

Lean4 ✓ `BEDC.Derived.KKTUp.KKTTasteGate single carrier alignment`

Proof. Start with Definition 300.2. It binds the primal, multiplier, residual, stationarity, and feasibility rows to the displayed $\mathsf{ConvexSet}^\uparrow$, $\mathsf{LPDuality}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, and $\mathsf{Real}^\uparrow$ dependency reads.

Use Definition 300.3 for the componentwise hsame routes. These routes preserve the public row inventory because they transport only the displayed stationarity and feasibility certificates.

Apply Definition 300.5 for the complementary-slackness ledger. Then use Definition 300.7 to restrict every downstream read to the finite carrier rows, Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{KKT}^\uparrow}$ surface.

The carrier has no coordinate for nonlinear optimization semantics, Slater existence, analytic differentiability, host minimization, or solver evidence. The listed finite rows are therefore the whole public packet export. □

Closure status for $\mathsf{NameCert}_{\mathsf{KKT}^\uparrow}$**.**

Constructive story: The construction starts from a finite constrained-convex-program packet, records primal, dual, residual, stationarity, feasibility, and complementary-slackness rows, and routes every read through displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KKT}^\uparrow}$, maturePackageClosed)

Scope closed: By Definition 300.7, Definition 300.8, Definition 300.9, and Definition 300.10, the scoped boundary binds the primal, dual, residual, stationarity, feasibility, slackness, hsame, Cont, Pkg, and NameCert rows of the finite KKT packet and its standard finite-packet bridge.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KKT}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.KKTUp.KKTCarrierPacket\_downstream\_consumer\_boundary`

Bridge status: paperBridge

Not claimed: This chapter does not close full nonlinear optimization, Slater existence, analytic differentiability, host real optimization semantics, or checked bridge status.

Upgrade path: The paper-side finite KKT packet is mature; the remaining obligation is Lean bridge checking without adding analytic solver semantics.

Definition 300.9 (Standard finite KKT packet presentation)**.** *A standard finite KKT packet presentation for* $\mathsf{KKT}^{\uparrow}$ *is a displayed finite table whose entries are exactly a primal variable row, a dual multiplier row, a constraint residual row, a stationarity row, feasibility rows, a complementary-slackness row, componentwise* hsame *rows between the displayed entries,* Cont *rows for stationarity and slackness readback,* Pkg *rows naming* $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *a* $\mathsf{NameCert}_{\mathsf{KKT}^{\uparrow}}$ *row, and the dependency rows through those five named certificates. Its standard reading is only this finite KKT table: primal data, multipliers, residuals, stationarity, feasibility, and complementary slackness as displayed rows.*

Theorem 300.10 (KKTUp finite-packet standard bridge)**.** *Let* S *be a standard finite KKT packet presentation in the sense of Definition 300.9. If* S *is admitted solely through the primal, multiplier, residual, stationarity, feasibility, complementary-slackness,* hsame, Cont, Pkg, NameCert, *and dependency rows exported by* $\mathsf{KKT}^{\uparrow}$, *then* S *has a conservative standard bridge through the public* $\mathsf{KKT}^{\uparrow}$ *finite packet. The forward read sends* S *to the public export of Definition 300.8. The reverse read sends that public export back to the same displayed finite KKT presentation, up to the componentwise* hsame *rows already present in* S*. Both composites preserve the primal, multiplier, residual, stationarity, feasibility, complementary-slackness, continuation, package, naming, and dependency rows.*

The bridge adds no Slater condition, differentiability oracle, solver certificate, analytic duality theorem, nonlinear optimizer, host minimization predicate, or host real-valued optimization semantics.

Lean4 ✓ `BEDC.Derived.KKTUp.KKTUp finite packet standard bridge`

Proof. Read S through Definition 300.9. Its entries are exactly the finite KKT rows listed there, together with the dependency rows over $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

By Definition 300.1, these are precisely the visible carrier rows accepted by $\mathsf{KKT}^{\uparrow}$. The classifier therefore admits S only by placing its primal, multiplier, residual, stationarity, feasibility, slackness, comparison, continuation, package, naming, and dependency entries in the corresponding finite BHist packet positions.

Apply Definition 300.2. The primal, multiplier, residual, stationarity, and feasibility entries of S are read through the displayed dependency certificates, and every such entry carries its visible hsame, Cont, Pkg, and NameCert provenance. Apply Definition 300.5 to the slackness entry. The complementary-slackness row is exactly the finite ledger over the displayed residual, multiplier, scalar slack, comparison, continuation, and package rows.

The forward standard read is now forced: it forgets no row and adds no row, but regards this accepted finite packet as the standard finite KKT table consisting of the displayed primal data, multipliers, residuals, stationarity, feasibility, and complementary slackness. This is the public export described in Definition 300.8, because that theorem identifies the whole public $\mathsf{KKT}^{\uparrow}$ surface with exactly these finite rows and their five dependency reads.

For the reverse read, start from the public export just obtained. Definition 300.8 says that the export contains only the listed finite rows of Definition 300.1. Reinsert those rows into the table positions of Definition 300.9. The stationarity and feasibility entries are stable under the componentwise comparisons by Definition 300.3, and the slackness entry is fixed at the ledger boundary by Definition 300.5. Hence the returned table is the same displayed finite KKT presentation, with equality understood through the hsame rows already carried by S.

The two composites therefore preserve every visible row of the presentation: primal, multiplier, residual, stationarity, feasibility, complementary slackness, hsame, Cont, Pkg, NameCert, and the named dependency rows. Finally, Definition 300.7 rules out every downstream read not present in that finite carrier. Since there is no carrier coordinate for Slater data, differentiability, solver evidence, analytic duality, nonlinear optimization, host minimization, or host real-valued optimizer semantics, the bridge is conservative at exactly the finite packet boundary. □

Theorem 300.11 (KKTUp finite slackness optimality extraction)**.** *Let K be an accepted* $\mathsf{KKT}^{\uparrow}$ *packet. Suppose the displayed primal variable row, dual multiplier row, feasibility rows, and complementary-slackness ledger of K instantiate a finite* $\mathsf{LPDuality}^{\uparrow}$ *ordered-field row as in Definition 235.1. Write the displayed primal and dual witnesses as $x : J \to F$ and $y : I \to F$. Suppose the projected feasibility rows give*

$$\mathsf{PrFeas}_{A,b}(x) \wedge \mathsf{DuFeas}_{A,b,c}(y),$$

and suppose the projected slackness ledger gives the two finite product families

$$\forall j \in J.\ c_j \cdot_F x_j \sim_F \beta_j(y) \cdot_F x_j, \qquad \forall i \in I.\ y_i \cdot_F \alpha_i(x) \sim_F y_i \cdot_F b_i.$$

Then the $\mathsf{KKT}^{\uparrow}$ *packet exports the* $\mathsf{LPDuality}^{\uparrow}$ *weak-duality slackness optimality packet for the displayed rows. In particular,*

$$\mathsf{PrObj}_{A,c}(x) \sim_F \mathsf{DuObj}_b(y),$$

$$\forall x'.\ \mathsf{PrFeas}_{A,b}(x') \Longrightarrow \mathsf{PrObj}_{A,c}(x') \preceq_F \mathsf{PrObj}_{A,c}(x),$$

and

$$\forall y'.\ \mathsf{DuFeas}_{A,b,c}(y') \Longrightarrow \mathsf{DuObj}_b(y) \preceq_F \mathsf{DuObj}_b(y').$$

With $\tau := \mathsf{PrObj}_{A,c}(x)$, every further feasible optimal primal witness and feasible optimal dual witness whose objective endpoints are classifier-equal to τ also enters the common objective-fiber sandwich supplied by Definition 235.16. The extraction uses only the finite carrier rows of K and adds no Slater condition, differentiability oracle, analytic strong-duality theorem, nonlinear optimizer, host minimization predicate, or solver certificate.

Proof. Project the accepted packet K through Definition 300.1. By Definition 300.2, the primal row, dual multiplier row, residual row, stationarity row, feasibility rows, comparison rows, continuation rows, package rows, and naming provenance are all visible finite rows. By hypothesis, the primal row and dual multiplier row are read through the same finite $\mathsf{LPDuality}^{\uparrow}$ ordered-field surface of Definition 235.1; hence they determine the displayed witnesses $x : J \to F$ and $y : I \to F$ together with the projected feasibility statement

$$\mathsf{PrFeas}_{A,b}(x) \wedge \mathsf{DuFeas}_{A,b,c}(y).$$

Apply Definition 300.5 to the slackness coordinate of K. The ledger is exhausted by the displayed residual, multiplier, scalar slack, hsame, Cont, and Pkg rows. Under the stated $\mathsf{LPDuality}^{\uparrow}$ instantiation, those displayed ledger rows are precisely the two product-family equalities

$$\forall j \in J.\ c_j \cdot_F x_j \sim_F \beta_j(y) \cdot_F x_j, \qquad \forall i \in I.\ y_i \cdot_F \alpha_i(x) \sim_F y_i \cdot_F b_i.$$

The stationarity row remains part of the accepted $\mathsf{KKT}^{\uparrow}$ provenance, but the present extraction does not use it to introduce any analytic optimality semantics.

The hypotheses are exactly the inputs of Definition 235.16. Applying that theorem to the projected $\mathsf{LPDuality}^{\uparrow}$ row gives the scalar objective equality, the primal optimality clause for x, the dual optimality clause for y, and the common objective-fiber sandwich at $\tau := \mathsf{PrObj}_{A,c}(x)$ for any further feasible optimal witnesses whose objective endpoints lie in that scalar fiber.

It remains only to identify the consumer boundary. By Definition 300.8, the public $\mathsf{KKT}^{\uparrow}$ export consists exactly of the finite primal, multiplier, residual, stationarity, feasibility, complementary-slackness, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows and their dependency reads through $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. By Definition 300.7, a downstream consumer cannot read any solver certificate, host minimization predicate, Slater witness, differentiability oracle, or full nonlinear optimization semantics from that packet. Therefore the optimality packet just obtained is a conservative finite-row export through the displayed $\mathsf{LPDuality}^{\uparrow}$ slackness theorem and no stronger optimization principle is introduced. □

The $\mathsf{RegSeqRat}^{\uparrow}$ hub reads its finite observation schedules through chapter 126, its dyadic tolerance ledger through chapter 291, and its terminal seal through chapter 38. The child files below keep those three sibling rows in the finite L10 four-face route and export no quotient stream equality, selected limit, host real equality, or ambient completeness coordinate.

Chapter 301

A Concrete Naming Certificate for $\mathsf{RegSeqRat}^{\uparrow}$

The $\mathsf{RegSeqRat}^{\uparrow}$ interface names Bishop-style regular rational streams over the dyadic observation layer. It sits between $\mathsf{DyadicRatCore}^{\uparrow}$, which supplies finite dyadic endpoints and radius ledgers, and $\mathsf{Real}^{\uparrow}$, which consumes accepted regular-sequence packets as the finite source surface of a real-name seal.

The carrier consumes the finite observation schedules supplied by $\mathsf{StreamName}^{\uparrow}$ in chapter 126 and the endpoint and radius rows supplied by $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291. Its accepted packets are then consumed by the $\mathsf{Real}^{\uparrow}$ seal surface in chapter 38.

301.1 Carrier and classifier

Definition 301.1 (RegSeqRat stream carrier)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carrier is a finite* BHist *packet naming a stream* $x : \mathbb{N} \to \mathsf{BHist}$ *together with the visible observation schedule supplied by* $\mathsf{StreamName}^{\uparrow}$*. For every finite window requested by the schedule, the packet carries:*

1. *a unary index row naming the requested stream position;*
2. *a dyadic endpoint row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. *the dyadic radius row* 2^{-n} *attached to the requested precision;*
4. *a regularity ledger comparing any two displayed endpoints* x_n *and* x_m *by the bound* $2^{-n} + 2^{-m}$*;*
5. *the* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance rows for the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *reads.*

The carrier stores only these scheduled finite observations. It contains no quotient carrier of Cauchy sequences, no ambient metric completion, and no host real endpoint.

Lean4 (definition) `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier`

Definition 301.2 (RegSeqRat dyadic-tolerance classifier)**.** *For accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carriers* x *and* y*, the predicate* $\mathsf{RegSeqRatClassifier}(x, y)$ *records dyadic tolerance agreement on scheduled finite windows. A classifier witness supplies, for each requested unary precision window, dyadic observation rows for* x_n *and* y_m*, a common finite refinement requested through* $\mathsf{StreamName}^{\uparrow}$*, and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger proving that the displayed dyadic distance is bounded by the requested dyadic radius.*

The classifier is a predicate over finite dyadic ledgers and visible stream-name rows. It does not identify streams by a quotient relation and does not import any external real equality.

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatClassifier window pairing`

301.2 Certificate obligations

Theorem 301.3 (RegSeqRat regularity obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the scheduled stream carrier of Definition 301.1;*
2. *classifier: the dyadic-tolerance classifier of Definition 301.2;*
3. *regularity ledger: every displayed pair of endpoints* x_n, x_m *is accompanied by a* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound row for* $2^{-n} + 2^{-m}$*;*
4. *dyadic-radius observation: each precision request exposes the radius row, endpoint row, and finite dyadic distance row consumed by the classifier;*
5. *finite-window transport: restricting, refining, or transporting a window through* hsame *and* Cont *repacks the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows without adding a quotient or completion object.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier regularity obligation surface`

Proof. Project the carrier definition. Its accepted packet already lists the scheduled unary index rows, dyadic endpoint rows, radius rows, regularity comparisons, continuation routes, package provenance, and naming rows.

The classifier definition supplies the second face: every comparison between two accepted packets is a finite dyadic-tolerance ledger over the same scheduled observation rows. The regularity and dyadic-radius faces are the two distinguished projections from that ledger.

For finite-window transport, follow the displayed hsame and Cont rows attached to the schedule. A restriction or refinement only changes which finite indices are read; each selected endpoint and radius remains a $\mathsf{DyadicRatCore}^{\uparrow}$ row with the same visible provenance. Repacking these projections gives exactly the five obligation rows and no additional carrier. □

Theorem 301.4 (RegSeqRat window regularity ledger)**.** *Every scheduled finite window in an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carrier carries the unary index rows requested by* $\mathsf{StreamName}^{\uparrow}$*, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for the displayed endpoints* x_n *and* x_m*, the two dyadic radius rows* 2^{-n} *and* 2^{-m}*, and the regularity ledger row displaying the bound*

$$2^{-n} + 2^{-m}.$$

The window carries these rows together with their hsame, Cont, Pkg, *and* NameCert *provenance, and it carries no quotient sequence, metric-completion object, or host real endpoint.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier window regularity ledger`

Proof. Project the stream carrier of Definition 301.1. A scheduled window is generated by the visible $\mathsf{StreamName}^{\uparrow}$ request, so its finite rows include the unary indices selected by that request and the dyadic endpoint rows accepted by $\mathsf{DyadicRatCore}^{\uparrow}$.

The regularity row in Definition 301.3 is the distinguished comparison row for any displayed endpoint pair. Reading it at the selected indices gives the two radius rows and the displayed ledger bound $2^{-n} + 2^{-m}$.

The same carrier projection lists the hsame, Cont, Pkg, and NameCert provenance rows attached to those observations. Since the carrier has no coordinate for quotient sequences, completions, or host real endpoints, the window ledger is exhausted by the displayed finite rows. □

Theorem 301.5 (RegSeqRat dyadic-radius observation)**.** *Every classifier precision request in* RegSeqRatClassifier(x, y) *exposes only the* StreamName$^{\uparrow}$ *schedule rows,* DyadicRatCore$^{\uparrow}$ *endpoint rows, dyadic radius rows, finite dyadic distance rows, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and* NameCert *rows of the accepted* RegSeqRat$^{\uparrow}$ *packets.*

Proof. Unpack the classifier of Definition 301.2. A precision request first selects a finite stream-name window and then reads dyadic endpoints for the two accepted packets through DyadicRatCore$^{\uparrow}$.

The tolerance witness is a finite dyadic ledger. It records the requested radius and the displayed dyadic distance, together with the common refinement supplied by the same StreamName$^{\uparrow}$ schedule.

All comparison and routing data are the componentwise hsame rows, displayed Cont routes, package provenance, and naming rows already present in the packets. No external real equality, quotient carrier, or completion datum can be projected from the classifier definition. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatClassifier dyadic radius observation`

Theorem 301.6 (RegSeqRat finite-window transport obligation)**.** *Restricting, refining, or transporting a finite* RegSeqRat$^{\uparrow}$ *window preserves the carrier and classifier rows through componentwise* hsame *comparisons and the displayed* Cont *routes. The transported window retains the* StreamName$^{\uparrow}$ *schedule,* DyadicRatCore$^{\uparrow}$ *endpoint rows, dyadic radius rows, finite dyadic distance rows, regularity ledgers,* Pkg *provenance, and* NameCert *rows, and introduces no additional carrier.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier finite window transport obligation`

Proof. Start with the window ledger of Definition 301.4. It identifies the unary index rows, endpoint rows, radius rows, and regularity bound carried by the source window.

Use Definition 301.5 for the classifier side. Every precision request reads the same schedule, endpoint, radius, distance, comparison, continuation, package, and naming rows.

Now apply the transport statement of Definition 301.7. Its hypotheses are exactly componentwise hsame rows and displayed Cont routes preserving the schedule, endpoints, radius rows, and regularity ledgers. Repacking the transported rows gives the target finite window with the same carrier and classifier surface, and there is no coordinate from which an additional carrier could be introduced. □

Theorem 301.7 (RegSeqRat classifier transport)**.** *Let x, y, x', y' be accepted* RegSeqRat$^{\uparrow}$ *stream carriers. Suppose x is transported to x' and y is transported to y' by componentwise* hsame *rows and displayed* Cont *routes that preserve the* StreamName$^{\uparrow}$ *schedule, the* DyadicRatCore$^{\uparrow}$ *endpoint rows, the dyadic-radius rows, and the finite regularity ledgers. If* RegSeqRatClassifier(x, y) *holds, then* RegSeqRatClassifier(x', y') *holds on the transported finite windows.*

The transported classifier reads only the carried stream-name, dyadic endpoint, radius, distance, regularity, hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier classifier transport`
Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatClassifier transport`

Proof. Start with a finite classifier witness for x and y. It consists of scheduled observation indices, endpoint rows, radius rows, common finite refinements, and dyadic distance ledgers.

Transport the two endpoint families along the supplied componentwise hsame rows. The Cont routes preserve the schedule, so every requested finite window for x' and y' has the same shape as the source window and reads the corresponding transported $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows.

Transport the dyadic-radius and distance ledgers through the same provenance. Since the regularity rows are carried with the endpoints, the transported witness satisfies the dyadic-tolerance classifier for x' and y' on the selected windows. No quotient sequence or ambient real equality is introduced. □

Theorem 301.8 (RegSeqRat classifier symmetric finite-window)**.** *Let x and y be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carriers. Suppose a finite* $\mathsf{StreamName}^{\uparrow}$ *scheduled window witnesses* $\mathsf{RegSeqRatClassifier}(x, y)$ *by displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic-radius rows, finite dyadic distance ledgers, regularity ledgers, componentwise* hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and* NameCert *rows. Then the reversed scheduled window witnesses* $\mathsf{RegSeqRatClassifier}(y, x)$ *using the same finite rows with the endpoint-pair order swapped.*

Proof. Project the witness through the classifier definition of Definition 301.2. The only visible coordinates are the scheduled endpoint rows, radius rows, dyadic distance ledgers, regularity rows, comparison rows, continuation routes, package provenance, and naming rows.

Reverse the scheduled window by swapping the two endpoint rows in every displayed pair. The dyadic distance ledger is read symmetrically by Definition 301.13, so the same tolerance rows certify the reversed endpoint comparison.

Transport the swapped rows through the finite-window transport obligation of Definition 301.6. The componentwise hsame comparisons and displayed Cont routes preserve the schedule, endpoint rows, radii, distance rows, and regularity ledgers.

Apply classifier transport from Definition 301.7 to the repacked reversed window. The resulting finite packet has exactly the rows required for $\mathsf{RegSeqRatClassifier}(y, x)$, and no quotient stream equality or host real equality is introduced. □

Theorem 301.9 (RegSeqRat common-refinement classifier determinacy)**.** *Let x and y be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carriers. Suppose two finite* $\mathsf{StreamName}^{\uparrow}$ *schedules both refine to a common window and carry the same transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic-radius rows, regularity ledgers, classifier ledgers,* hsame *comparisons,* Cont *routes, and* Pkg *provenance. Then the two schedules determine the same* $\mathsf{RegSeqRatClassifier}(x, y)$ *read on that common window.*

The common read is the one exposed by the $\mathsf{Real}^{\uparrow}$ *seal handoff through the refined finite rows. It contains no quotient stream equality, host real equality, or metric-completion comparison.*

Proof. Project the carrier of Definition 301.1 and the classifier of Definition 301.2. Each schedule reads only unary index rows, dyadic endpoints, radius rows, regularity ledgers, dyadic distance ledgers, and the displayed provenance attached to those rows.

Move both schedules to the stated common window. The finite-window transport obligation of Definition 301.6 carries the endpoints, radii, regularity rows, and provenance to the same refined rows on that window.

Apply classifier transport from Definition 301.7 to both transported reads. Since the hypotheses identify the transported endpoint, radius, regularity, classifier, hsame, Cont, and Pkg rows, the two classifier witnesses have the same finite ledger on the common window.

Finally read the $\mathsf{Real}^{\uparrow}$ seal boundary through Definition 301.22. That handoff consumes only the refined finite rows just identified, so it cannot distinguish the two source schedules and yields the same RegSeqRatClassifier read. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier common refinement classifier determinacy`

Theorem 301.10 (RegSeqRat classifier common-refinement transitivity)**.** *Let x, y, and z be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets. Suppose one finite* $\mathsf{StreamName}^{\uparrow}$ *schedule witnesses* $\mathsf{RegSeqRatClassifier}(x, y)$ *and another finite* $\mathsf{StreamName}^{\uparrow}$ *schedule witnesses* $\mathsf{RegSeqRatClassifier}(y, z)$*. If the two witnesses admit a common refinement on which the middle y endpoint rows, dyadic-radius rows, regularity ledgers, classifier ledgers,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance agree, then the common refinement witnesses* $\mathsf{RegSeqRatClassifier}(x, z)$ *using only the displayed finite rows.*

Proof. Project both classifier witnesses to their finite scheduled windows. The finite-window transport obligation of Definition 301.6 moves the x–y and y–z endpoint families, radius rows, regularity ledgers, classifier ledgers, hsame rows, Cont routes, package rows, and naming rows to the stated common refinement.

On that common refinement, apply Definition 301.9 to the two transported reads of the middle y side. The agreement hypotheses identify the displayed y endpoint rows and all provenance rows that the classifier can inspect, so the two source schedules have one middle finite surface.

Read the two dyadic tolerance ledgers through Definition 301.13. Since the radius and distance data are displayed on the same refined finite window, the x endpoint rows and the z endpoint rows compose across the common y rows inside the dyadic ledger surface.

Finally use classifier transport from Definition 301.7 to repack the composed finite surface as a witness for $\mathsf{RegSeqRatClassifier}(x, z)$. The construction introduces no quotient stream equality, host real equality, metric-completion object, or $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 301.11 (RegSeqRat diagonal window extraction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S be a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule whose two classifier witnesses for displayed reads of P admit a common refinement. At the shared unary index of that refinement, the diagonal observation is a displayed finite window consisting only of* BHist *rows, the* UnaryHistory *witness for the index, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, regularity ledgers,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance.*

The same diagonal row is exactly the finite source row consumed by the $\mathsf{Real}^{\uparrow}$ *seal handoff. Thus the extraction exports no quotient stream, host real equality, metric-completion point,* $\mathsf{Real}^{\uparrow}$ *completeness datum, or additional bridge coordinate.*

Proof. Project the two scheduled reads through Definition 301.12. This exposes the common-refinement schedule together with the displayed unary index, endpoint rows, radius rows, regularity ledgers, comparison rows, continuation routes, package provenance, and naming provenance.

Read the radius and distance components by Definition 301.13. The diagonal row at the shared unary index is therefore a finite $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint-and-radius observation inside the same displayed BHist packet.

Use Definition 301.9 to identify the two transported classifier reads on the common refinement. Since the classifier can inspect only the displayed rows already projected, both witnesses present the same diagonal finite-window row.

Finally apply Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal consumes exactly the finite regular-rational source surface exposed by that diagonal row, so the handoff reads the extracted finite window itself and not any quotient, host-real, completion, or bridge datum. □

Theorem 301.12 (RegSeqRat scheduled-window projection)**.** *For every accepted finite scheduled window of a* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carrier, projection exposes only the unary index row, the*

DyadicRatCore$^{\uparrow}$ *endpoint row, the dyadic-radius row, the regularity ledger rows, and the associated* hsame, Cont, Pkg, *and* NameCert *rows. No projected window contains a quotient sequence coordinate, a host real coordinate, or a completion object.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier scheduled window projection`

Proof. Project Definition 301.1. Each scheduled window is stored as a finite BHist packet, and the carrier list contains precisely the unary index, dyadic endpoint, radius, regularity, continuation, package, and naming rows.

Apply Definition 301.3 to read the same packet as a local obligation surface. The theorem identifies the carrier, regularity ledger, dyadic-radius observation, and finite-window transport faces as projections of those displayed rows.

Finally use Definition 301.2 and Definition 301.7 to check that classifier reads do not add further coordinates. They consume the projected dyadic ledgers and transport them along the displayed hsame and Cont rows. □

Theorem 301.13 (RegSeqRat dyadic-radius ledger readback)**.** *Every radius or distance comparison used by* RegSeqRatClassifier *is read back from displayed* DyadicRatCore$^{\uparrow}$ *radius and distance ledgers in the accepted finite window. The classifier never appeals to quotient equality of streams, host real equality, or an ambient metric-completion comparison.*

Proof. Unfold the classifier of Definition 301.2. A witness is a finite list of scheduled dyadic observations, a common refinement row, and a DyadicRatCore$^{\uparrow}$ tolerance ledger for the displayed distance against the requested radius.

The available radius and regularity rows are exactly the dyadic-radius observation and regularity faces of Definition 301.3. Thus every comparison used in a classifier witness has a displayed ledger source inside the finite packet.

Transport stability follows from Definition 301.7. When the window is carried through hsame and Cont, the radius and distance ledgers are carried as the same DyadicRatCore$^{\uparrow}$ rows, so no quotient or host-real equality can enter the readback. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier dyadic radius ledger readback`

Theorem 301.14 (RegSeqRat finite-window transport closure)**.** *Restriction, refinement, and componentwise* hsame *or* Cont *transport of an accepted finite* RegSeqRat$^{\uparrow}$ *window repack the same* StreamName$^{\uparrow}$ *schedule rows and* DyadicRatCore$^{\uparrow}$ *endpoint, radius, distance, and regularity rows. The repacked window remains within the displayed carrier and classifier surface of the same stream-name certificate.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier finite window transport closure`

Proof. Begin with the finite-window transport row singled out by Definition 301.3. Restriction selects fewer scheduled unary indices, while refinement requests a larger finite schedule through StreamName$^{\uparrow}$ and keeps each selected endpoint and radius as a displayed DyadicRatCore$^{\uparrow}$ row.

For componentwise transport, apply Definition 301.7. Its hypotheses preserve the stream schedule, endpoint rows, radius rows, and regularity ledgers, and its conclusion rebuilds the dyadic-tolerance classifier on the transported finite windows.

The carrier of Definition 301.1 and the classifier of Definition 301.2 therefore close under these finite operations by repacking existing rows. The resulting packet is still a RegSeqRat$^{\uparrow}$ source surface, not a quotient or completion object. □

Theorem 301.15 (RegSeqRat finite-window classifier reflexivity)**.** *Every accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *scheduled window over its displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, regularity ledger,* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *row carries a* RegSeqRatClassifier *witness from that finite window to itself.*

The self-classifier uses the same schedule rows, endpoint rows, radius rows, dyadic distance ledgers, regularity ledgers, package provenance, and naming row as the input window. It introduces no quotient stream, host real equality, metric-completion datum, or $\mathsf{Real}^{\uparrow}$ *seal claim.*

Proof. Project the accepted window through Definition 301.3. This exposes the finite schedule, endpoint rows, radius rows, regularity ledger, displayed hsame comparisons, Cont routes, Pkg provenance, and NameCert row that already belong to the accepted packet.

Use identity hsame rows on each displayed coordinate and reuse the same Cont routes. With the source and target window equal, the endpoint, radius, distance, and regularity ledgers are exactly the rows already read by the dyadic-radius classifier surface.

Apply Definition 301.14 to the identity transport of this finite packet. The resulting repackaged surface is a RegSeqRatClassifier witness from the window to itself, built only from the displayed finite-window rows. □

Theorem 301.16 (RegSeqRat refined-window gluing)**.** *Let* W_0 *and* W_1 *be accepted finite scheduled windows of the same* $\mathsf{RegSeqRat}^{\uparrow}$ *stream carrier. Suppose* $\mathsf{StreamName}^{\uparrow}$ *supplies a finite common refinement schedule* $W_{\cup}$ *whose source rows restrict to the two input schedules, and suppose the overlapping unary index rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic radius rows, regularity ledgers, and classifier ledgers of* W_0 *and* W_1 *agree along the displayed* hsame *comparisons and* Cont *routes. Then the refined union repacks as an accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window over* $W_{\cup}$*.*

The restrictions of this refined window back to W_0 *and* W_1 *recover the original classifier witnesses on the two input windows. The gluing operation introduces no quotient sequence, host real equality, metric-completion row, or additional* $\mathsf{Real}^{\uparrow}$ *seal datum.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier refined window gluing`

Proof. Project the two accepted windows by Definition 301.12. This exposes the two finite lists of unary indices, dyadic endpoints, radius rows, regularity ledgers, classifier ledgers, and their displayed hsame, Cont, Pkg, and NameCert provenance rows.

Use the common-refinement schedule supplied by $\mathsf{StreamName}^{\uparrow}$ to choose the joint finite source window. On rows that come from only one input window, keep the corresponding projected row. On overlapping rows, use the stated hsame comparisons and Cont routes to identify the unary index, endpoint, radius, regularity, and classifier ledger entries as the same displayed BEDC data.

The overlap classifier is stable by Definition 301.7. Thus the two classifier witnesses restrict to the same transported witness wherever the schedules meet, and the non-overlap rows remain the witnesses already accepted in the two input windows.

Apply Definition 301.14 to repack the joint schedule, transported overlap rows, and retained non-overlap rows as one finite $\mathsf{RegSeqRat}^{\uparrow}$ carrier and classifier surface. Restricting the repacked window back along either source schedule discards only the other side's rows, so it recovers the original classifier witness for that input window. Since every row used in the construction came from the two projected finite packets and their displayed refinement routes, no quotient sequence, host real equality, completion object, or $\mathsf{Real}^{\uparrow}$ seal row is created. □

Theorem 301.17 (RegSeqRat prefix-window monotonicity)**.** *Let* W *be an accepted finite scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *window over the dyadic-tolerance classifier of Definition 301.2. Suppose* W' *is a finite*

certified $\mathsf{StreamName}^{\uparrow}$ *prefix subwindow of* W*, obtained by restricting the displayed schedule to an initial certified prefix. Then* W' *repacks as an accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window over the same dyadic-tolerance classifier.*

The endpoint, radius, regularity, classifier, hsame*,* Cont*,* Pkg*, and* NameCert *rows of the repacked prefix are exactly the projections of the corresponding rows of* W*. The prefix restriction introduces no quotient sequence, host real equality, completion object, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Proof. Project W by Definition 301.12. This exposes the visible finite schedule, endpoint rows, radius rows, regularity ledgers, classifier ledgers, hsame comparisons, Cont routes, Pkg provenance, and NameCert rows.

Use the certified $\mathsf{StreamName}^{\uparrow}$ prefix to select the initial sublist of scheduled rows. Since the subwindow is a prefix restriction, every endpoint, radius, regularity, classifier, hsame, Cont, Pkg, and NameCert row visible in W' is the corresponding projected row already visible in W.

Apply Definition 301.14 to repack those selected rows as a finite accepted $\mathsf{RegSeqRat}^{\uparrow}$ carrier and classifier surface over the original dyadic-tolerance classifier. No new coordinate is available in this restriction step, so quotient sequences, host real equality, completion objects, and $\mathsf{Real}^{\uparrow}$ seal rows cannot enter the prefix packet.

Finally, when the prefix is compared back with the larger window, use Definition 301.16 on the overlap consisting of the prefix rows. The gluing restriction recovers the same classifier witness on that overlap, because both sides read the same projected endpoint, radius, regularity, classifier, hsame, Cont, Pkg, and NameCert rows. □

Theorem 301.18 (RegSeqRat shifted-tail regularity closure)**.** *Let* P *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet whose stream carrier, dyadic-tolerance classifier, regularity ledger, scheduled windows, and dyadic-radius rows are those of Definition 301.1 and Definition 301.2. For every certified unary tail index* k*, shifting the schedule of* P *by* k *and then reading any finite* $\mathsf{StreamName}^{\uparrow}$ *window produces another accepted regular-rational packet over the same dyadic-tolerance classifier.*

The shifted packet exposes only the transported BHist *rows, componentwise* hsame *comparisons, visible* Cont *routes, the retained* Pkg *row, and the local* NameCert *provenance. It exports no quotient sequence, host real equality, ambient metric-completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Use Definition 301.12 on the shifted finite tail window. The projection exposes exactly the unary tail indices, dyadic endpoint rows, regularity ledgers, radius rows, classifier ledgers, and the visible provenance already carried by P.

Read the transported radius and distance comparisons through Definition 301.13. Since the tail shift only selects later scheduled rows, each tolerance comparison is still a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row in the same source packet.

Apply Definition 301.14 to repack the selected tail rows as a finite $\mathsf{RegSeqRat}^{\uparrow}$ window over the original dyadic-tolerance classifier. When two shifted finite windows overlap, use Definition 301.16 to identify the common tail rows and retain the same classifier witness on the common refinement.

All consumed coordinates are therefore finite BEDC rows of the original accepted packet. The tail construction has no coordinate from which to read quotient sequences, host real equality, metric completion, or a $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 301.19 (RegSeqRat tail-shift classifier stability)**.** *Let* P *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet whose scheduled stream carrier, dyadic-tolerance classifier, regularity ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, and* $\mathsf{Real}^{\uparrow}$ *seal source are the displayed rows of Definition 301.1 and Definition 301.2. If a certified unary tail shift selects a later finite window and the shared tail window is transported by the recorded* hsame *comparisons and* Cont *routes, then*

the shifted packet preserves the same classifier route through the $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal.*

The shifted read exports no quotient sequence equality, host real equality, metric-completion object, selected limit, ambient completeness row, or additional seal datum.

Proof. Start with Definition 301.18. It repacks the shifted finite window as an accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet over the original dyadic-tolerance classifier, using only transported rows from the source packet.

Use Definition 301.15 on the shared tail window. The finite $\mathsf{StreamName}^{\uparrow}$ schedule, endpoint observations, $\mathsf{DyadicRatCore}^{\uparrow}$ radius rows, regularity ledgers, classifier ledgers, hsame comparisons, Cont routes, Pkg provenance, and NameCert rows are the same displayed finite surface before and after the tail shift.

Apply Definition 301.20. It keeps the shifted $\mathsf{Real}^{\uparrow}$ seal source attached to the same transported classifier and finite-window rows whenever the shared tail window agrees under the recorded transport and replay data.

Finally invoke Definition 301.22. The seal consumes exactly the finite regular-rational source surface already identified, so the classifier route through stream windows, dyadic radii, readback, and seal rows is stable under the tail shift and cannot acquire any hidden quotient, host-real, completion, or seal coordinate. □

Theorem 301.20 (RegSeqRat shifted-tail seal determinacy)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let k be a certified unary tail index for its* $\mathsf{StreamName}^{\uparrow}$ *schedule. For any finite* $\mathsf{StreamName}^{\uparrow}$ *window read after shifting P by k, the* $\mathsf{Real}^{\uparrow}$ *seal source read from that window is determined by the shifted scheduled window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic-radius rows, regularity ledgers, classifier ledgers, componentwise* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *rows already present in P.*

If two selected shifted windows admit a common finite refinement and agree there on those displayed rows, then the shifted $\mathsf{Real}^{\uparrow}$ *seal source read is invariant under restriction to that refinement. The invariant source surface exports no quotient sequence, host real equality, metric-completion object,* $\mathsf{Real}^{\uparrow}$ *completeness row, or seal datum beyond those finite rows.*

Proof. Apply Definition 301.18 to the certified tail shift and selected finite window. This repacks the shifted read as an accepted finite $\mathsf{RegSeqRat}^{\uparrow}$ window whose visible coordinates are precisely the scheduled tail rows, dyadic endpoints, dyadic-radius rows, regularity ledgers, classifier ledgers, hsame comparisons, Cont routes, Pkg provenance, and NameCert rows inherited from P.

For two selected windows with a common finite refinement, use Definition 301.16 to repack the refinement as one accepted finite window. The classifier witnesses on the overlap are fixed by Definition 301.9, so restriction back to either shifted window reads the same displayed classifier and regularity surface.

Finally apply Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal boundary consumes exactly the finite regular-rational source rows exposed above, so common refinement cannot create a new seal coordinate. The result is a determinacy statement for finite BEDC rows only, with no quotient sequence, host real equality, metric-completion object, completeness row, or further seal datum in the export surface. □

Theorem 301.21 (RegSeqRat tail-cofinal RealUp seal invariance)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Suppose T_0 and T_1 are two certified unary tail shifts of P whose finite* $\mathsf{StreamName}^{\uparrow}$ *windows admit a common cofinal refinement. Assume that, on the displayed refinement, the two*

shifted windows agree on the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic radius rows, regularity ledgers, classifier witnesses,* hsame *comparisons,* Cont *routes, and* Pkg *provenance rows.*

Then the two tail reads determine the same $\mathsf{Real}^{\uparrow}$ *seal source surface: every row consumed by the* $\mathsf{Real}^{\uparrow}$ *seal from* T_0 *is the corresponding displayed row consumed from* T_1*, after restriction to the common cofinal refinement. This invariance is only a statement about finite BEDC source rows. It does not identify quotient streams, host real values, metric-completion points, or any* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 301.18 separately to the two certified tail shifts. Each shifted finite window is repacked as an accepted $\mathsf{RegSeqRat}^{\uparrow}$ window over the original dyadic-tolerance classifier, and each exposes only transported BHist rows, displayed hsame comparisons, visible Cont routes, the retained Pkg row, and local NameCert provenance.

Use the common cofinal refinement as the joint schedule. The hypotheses identify the endpoint, radius, regularity, classifier, hsame, Cont, and Pkg rows on the overlap, so Definition 301.16 repacks the two tail windows into one accepted finite refinement whose restrictions recover the two original tail classifier witnesses.

Finally apply Definition 301.22. A $\mathsf{Real}^{\uparrow}$ seal read of a regular rational stream consumes exactly the scheduled stream-carrier rows, dyadic-tolerance classifier rows, regularity ledger rows, dyadic-radius rows, finite-window transport rows, and their displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and NameCert provenance. Since both tail reads restrict to the same refined finite packet, their seal source surfaces are the same displayed BEDC surface, with no quotient stream, host real equality, completion object, or completeness row available. □

Theorem 301.22 (RegSeqRat RealUp seal handoff)**.** *An accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet is exactly the finite source surface consumed by the existing* $\mathsf{Real}^{\uparrow}$ *seal rows. The handoff surface consists of the scheduled stream carrier, the dyadic-tolerance classifier, the regularity ledger, the dyadic-radius observation rows, the finite-window transport rows, and their* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Thus a $\mathsf{Real}^{\uparrow}$ *seal may read a regular rational stream only through those finite source rows. The handoff does not provide RealUp completeness, quotient carriers, metric completeness, or standard bridge data.*

Proof. Read the accepted packet through Definition 301.3. The projection exposes precisely the carrier, classifier, regularity, dyadic-radius, and finite-window transport rows.

The existing $\mathsf{Real}^{\uparrow}$ seal rows consume regular rational-stream data by finite windows: a stream-name schedule selects dyadic endpoint observations, the classifier supplies dyadic tolerance agreement, and the regularity ledger supplies the source bounds needed at the seal boundary.

Conversely, any source read used by the seal is one of those finite rows, because the $\mathsf{RegSeqRat}^{\uparrow}$ carrier has no coordinate for quotient sequences, ambient metric completion, host real equality, or standard bridge data. The accepted packet and the seal source surface therefore coincide at the displayed finite boundary. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier real seal handoff`

Theorem 301.23 (RegSeqRat public source export)**.** *Any consumer of* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *may read exactly the scheduled stream carrier, the dyadic-tolerance classifier, the regularity ledger, dyadic-radius observations, finite-window transport and gluing rows, the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, and their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The exported surface contains no quotient sequences, host real equality, ambient metric completion, or extra seal data beyond those finite BEDC rows.

Proof. The regularity obligation surface of Definition 301.3 exposes the scheduled carrier, classifier, regularity ledger, dyadic-radius rows, and their displayed provenance. The finite-window transport theorem Definition 301.14 keeps reads inside the same finite source rows under restriction and refinement.

The refined-window gluing theorem Definition 301.16 adds only common-refinement rows and transported overlap witnesses, again using the visible BHist, hsame, Cont, Pkg, and NameCert provenance already present in the two windows.

Finally, Definition 301.22 identifies the $\mathsf{Real}^{\uparrow}$ seal boundary with that same regular-rational source surface. Since each cited theorem exposes only finite BEDC rows, their composition exhausts the public export surface and leaves no coordinate through which a consumer could read quotient sequences, host real equality, ambient metric completion, or additional seal data. □

Theorem 301.24 (RegSeqRat standard bridge boundary)**.** *An accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet determines the finite standard-bridge boundary consumed by the* $\mathsf{Real}^{\uparrow}$*-facing regular rational stream interface. The boundary consists exactly of the scheduled stream carrier, the dyadic-tolerance classifier, the regularity ledger, dyadic-radius observation rows, finite-window transport and gluing rows, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The bridge boundary exports no quotient sequence, host real equality, ambient metric completion, or $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Start with the public source export of Definition 301.23. It names the scheduled carrier, classifier, regularity ledger, radius observations, finite-window operations, seal handoff, and provenance as the consumer-visible rows of an accepted packet.

The seal-facing part is supplied by Definition 301.22. The finite-window part is supplied by Definition 301.14, Definition 301.16, and Definition 301.12; the dyadic-radius readback is supplied by Definition 301.13.

Composing these rows gives the finite boundary read by the regular rational stream interface. Since the cited statements expose only the carrier, classifier, ledger, window, seal, and provenance coordinates just listed, there is no extra coordinate from which a quotient sequence, host real equality, ambient metric completion, or $\mathsf{Real}^{\uparrow}$ completeness datum could be exported. □

Theorem 301.25 (RegSeqRat StreamName Dyadic finite-window lock)**.** *Let P be an accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *scheduled window. The same finite packet simultaneously exposes the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger extracted by the dyadic-radius window theorem, and the exact* $\mathsf{Real}^{\uparrow}$ *seal source surface consumed by Definition 301.22.*

If two refinements of P agree on the displayed $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic-radius rows, regularity ledgers, classifier rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *rows, then they induce the same* $\mathsf{Real}^{\uparrow}$ *seal source surface. The lock exports no quotient stream, host real equality, ambient completion point,* $\mathsf{Real}^{\uparrow}$ *completeness datum, or additional standard-bridge coordinate.*

Proof. Project P by Definition 301.12. This exposes the finite $\mathsf{StreamName}^{\uparrow}$ schedule row together with the endpoint, regularity, classifier, transport, provenance, and local naming rows carried by the same accepted packet.

Next read the regularity and radius data through Definition 301.4 and Definition 301.13. These two facts identify the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows and radius ledger as rows of the same finite scheduled window.

For two refinements, apply Definition 301.16. Agreement on the displayed schedule, endpoint, radius, regularity, classifier, hsame, Cont, Pkg, and NameCert rows repacks the overlap as one accepted finite $\mathsf{RegSeqRat}^{\uparrow}$ window whose restrictions recover both refinements.

Finally use Definition 301.22. Since the seal reads only the finite regular-rational source rows just locked together, the two agreeing refinements present the same $\mathsf{Real}^{\uparrow}$ seal source surface and no hidden quotient, completion, equality, or bridge coordinate is available. □

Theorem 301.26 (RegSeqRat StreamName Dyadic seal refinement lock)**.** *Let W_0 and W_1 be accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *windows that refine to a common* $\mathsf{StreamName}^{\uparrow}$ *schedule. Suppose the two windows agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, regularity ledgers, classifier witnesses,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *rows. Then their* $\mathsf{Real}^{\uparrow}$ *seal source reads are the same displayed finite BEDC surface.*

The refinement lock exports no quotient stream equality, host real equality, ambient completion point, $\mathsf{Real}^{\uparrow}$ *completeness row, or extra bridge coordinate.*

Proof. First glue the two windows along the common schedule using Definition 301.16. The hypotheses identify exactly the schedule, endpoint, radius, regularity, classifier, transport, provenance, and naming rows needed to repack the common refinement as one accepted finite $\mathsf{RegSeqRat}^{\uparrow}$ window.

Next apply Definition 301.21. It says that cofinal refinement does not change the finite source surface read by the $\mathsf{Real}^{\uparrow}$ seal, provided the displayed carrier, classifier, ledger, and provenance rows agree on the refinement.

Finally combine Definition 301.22 with Definition 301.25. These theorems identify the seal-facing read with the same StreamName schedule and DyadicRatCore ledger rows already locked by the common refinement, so the result is a finite BEDC surface equality and not a host-level equality or completion claim. □

Theorem 301.27 (RegSeqRat tail seal diagonal-window coherence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0 and T_1 be certified unary tail shifts of P. Suppose the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows for T_0 and T_1 admit a common refinement. Then the diagonal finite window supplied by Definition 301.25 is exactly the finite* BHist *surface consumed by Definition 301.22 for both shifted reads.*

Consequently the two $\mathsf{Real}^{\uparrow}$ *seal source surfaces agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, regularity rows, classifier rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance. The coherence exports no quotient stream equality, host real equality, completion object, or completeness datum.*

Proof. Apply Definition 301.18 to each certified tail shift. Each shifted read is thereby presented as an accepted finite regular-rational window carrying only the displayed schedule, dyadic, regularity, classifier, transport, package, and naming rows of P.

Use Definition 301.16 to glue the two selected windows along the common refinement. The overlap classifier is determined by Definition 301.9, so both restrictions read the same classifier and regularity rows on the refined window.

Now apply Definition 301.25. The lock identifies the shared $\mathsf{StreamName}^{\uparrow}$ schedule rows with the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers inside the diagonal finite packet.

Finally use Definition 301.22. The seal consumes exactly that finite BHist surface, so the two shifted reads present the same displayed source rows to $\mathsf{Real}^{\uparrow}$ and still expose no quotient stream, host equality, completion object, or completeness datum. □

Theorem 301.28 (RegSeqRat cofinal tail seal selected-window uniqueness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0 and T_1 be two certified cofinal tail shifts of P. Suppose the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by T_0 and T_1 have a common cofinal refinement on which their selected endpoints, dyadic radii, regularity ledgers, classifier ledgers,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance agree. Then the two selected windows determine the same finite* $\mathsf{Real}^{\uparrow}$ *seal source surface.*

The conclusion is only a uniqueness statement for the displayed finite BHist *rows: it exports no quotient stream, host real equality, metric-completion point, completeness row, or standard bridge coordinate.*

Proof. First apply Definition 301.21. The common cofinal refinement identifies the two tail surfaces on exactly the selected endpoint, dyadic-radius, regularity, classifier, hsame, Cont, Pkg, and NameCert rows named in the hypotheses.

Next use Definition 301.27. The diagonal selected window is the finite BHist surface through which both cofinal tail reads pass, so the two selected windows cannot supply separate seal-facing source data.

Finally apply Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal reads precisely that diagonal finite regular-rational source surface. Since the handoff consumes no quotient stream, host-real equality, metric-completion point, completeness row, or standard bridge coordinate, the two selected windows have the same finite seal source surface and nothing stronger is exported. □

Theorem 301.29 (RegSeqRat finite-window bridge source exhaustion)**.** *For every accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, the standard-bridge source rows of Definition 301.24 are exactly the finite-window rows locked by Definition 301.25, refined by Definition 301.26, and stabilized by Definition 301.27. Equivalently, the bridge-facing rows and the finite-window rows are the same displayed BEDC surface over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This exhaustion statement does not identify quotient streams, host real equality, ambient completion points, metric completeness, $\mathsf{Real}^{\uparrow}$ *completeness rows, or bridge coordinates beyond the displayed finite source surface.*

Proof. Project the accepted packet through Definition 301.24. The bridge boundary exposes only the scheduled stream carrier, dyadic-tolerance classifier, regularity ledger, radius observations, finite transport and gluing rows, the $\mathsf{Real}^{\uparrow}$ seal handoff, and the displayed provenance rows.

Next apply Definition 301.25. The same packet projects the $\mathsf{StreamName}^{\uparrow}$ schedule rows together with the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers, and the lock identifies those rows with the finite $\mathsf{Real}^{\uparrow}$ seal source surface read by Definition 301.22.

For overlapping finite windows, use Definition 301.26. It glues agreement on the displayed schedule, endpoint, radius, regularity, classifier, hsame, Cont, Pkg, and NameCert rows into one finite surface, so the bridge read cannot split from the window read on a common refinement.

Finally use Definition 301.27 and Definition 301.22. The diagonal window is exactly the finite BHist surface consumed by the $\mathsf{Real}^{\uparrow}$ seal, so every seal row named by the standard bridge boundary is already one of the displayed finite-window source rows. The carrier has no coordinate for quotient streams, host real equality, ambient completion, metric completeness, or any further bridge datum, which gives equality of the two displayed source surfaces. □

Theorem 301.30 (RegSeqRat refined-window gluing RealUp seal factorization)**.** *Let W_0 and W_1 be accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *windows whose displayed* $\mathsf{StreamName}^{\uparrow}$ *schedules admit a common refined window $W_{\cup}$ as in Definition 301.16. Suppose the overlap hypotheses identify the shared unary index rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, dyadic-radius rows, regularity ledgers, classifier*

ledgers, hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance on* $W_\cup$*. Then the common refined window* $W_\cup$ *is exactly the finite* BHist *source surface consumed by the* $\mathsf{Real}^\uparrow$ *seal handoff of Definition 301.22.*

Equivalently, composing refined-window gluing with the $\mathsf{Real}^\uparrow$ *seal handoff factors through the displayed finite BEDC packet* $W_\cup$*. The composition introduces no quotient stream, host real equality, metric completeness, completion point, or seal datum beyond the finite rows already present in* $W_\cup$*.*

Proof. Apply Definition 301.16 to the two accepted windows. The result is one accepted finite $\mathsf{RegSeqRat}^\uparrow$ window over $W_\cup$ whose union schedule, unary indices, endpoint rows, dyadic-radius rows, regularity ledgers, provenance rows, and classifier readback restrict to the two source windows.

Use the classifier window-pairing row of Definition 301.8 on the refined window. It identifies the shared endpoint observations and radius observations as the same displayed $\mathsf{DyadicRatCore}^\uparrow$ rows inside the common finite packet, so the two source reads cannot supply separate seal-facing data.

Finally apply Definition 301.22. The handoff consumes exactly the accepted finite regular-rational source surface just produced by the gluing theorem, and therefore reads $W_\cup$ itself as the $\mathsf{Real}^\uparrow$ finite seal source. Since every consumed coordinate is already a displayed BHist, $\mathsf{StreamName}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, hsame, Cont, Pkg, or NameCert row of $W_\cup$, the composition has no route to a quotient stream, host real equality, metric-completion object, completion point, or extra seal datum. □

Theorem 301.31 (RegSeqRat common-window RealUp seal transitivity)**.** *Let* W_0*,* W_1*, and* W_2 *be accepted finite* $\mathsf{RegSeqRat}^\uparrow$ *windows. Suppose the* W_0/W_1 *and* W_1/W_2 *refined-window gluing surfaces identify the middle unary index rows,* $\mathsf{DyadicRatCore}^\uparrow$ *endpoint rows, dyadic-radius rows, regularity ledgers, classifier ledgers,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance on a shared common refinement. Then the composed* W_0/W_2 *handoff factors through the same finite* BHist *source surface consumed by the* $\mathsf{Real}^\uparrow$ *seal.*

Equivalently, the two common-window legs determine one seal-facing finite packet: its source rows are exactly the shared unary observations, DyadicRatCore endpoints, radius rows, regularity and classifier ledgers, hsame*,* Cont*,* Pkg*, and* NameCert *provenance read on the common refinement. The composed handoff exports no quotient stream, host real equality, metric completeness, completion point, or seal datum beyond that finite source surface.*

Proof. Project both legs with Definition 301.16. The W_0/W_1 leg gives one accepted refined packet whose restrictions recover W_0 and W_1, and the W_1/W_2 leg gives one accepted refined packet whose restrictions recover W_1 and W_2.

Use Definition 301.9 on the middle window. The middle unary observations, endpoint rows, radius rows, regularity ledgers, classifier ledgers, and provenance rows are the same finite read, so the two legs share one common packet rather than two seal-facing sources.

Apply Definition 301.27 and Definition 301.22. The diagonal finite read is the BHist source surface consumed by the $\mathsf{Real}^\uparrow$ seal, and the seal handoff reads only that finite regular-rational packet.

Finally invoke Definition 301.30 on the composed common packet. The W_0/W_2 handoff therefore factors through the same finite source surface, with no coordinate for quotient streams, host real equality, metric completeness, completion points, or additional seal data. □

Theorem 301.32 (RegSeqRat refined-window chain exhaustion)**.** *Let $W_0, W_1, \ldots, W_n$ be a finite chain of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *scheduled windows. Suppose every consecutive overlap agrees on the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, regularity ledgers, classifier ledgers,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}$ *provenance. Then the chain repacks into one accepted common finite window, and every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from the chain factors through that common packet.*

The common packet exposes only the displayed regular-rational source rows just listed. It does not introduce quotient sequence equality, host real equality, metric completion, or completeness data.

Proof. Induct on the finite chain of scheduled windows. The first adjacent link is handled by Definition 301.16, which repacks two accepted windows into one accepted finite refinement over the shared schedule, endpoint, radius, ledger, transport, and provenance rows.

In the step, the induction hypothesis supplies one accepted common packet for the initial segment. Apply Definition 301.9 on the overlap with the next scheduled window. The determinacy row identifies the shared $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius data, regularity ledger, classifier ledger, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows as one finite read.

Use Definition 301.16 again to glue the next window to the common packet. This preserves acceptedness and produces the next common finite window without adding any off-surface sequence, host-real, completion, or completeness coordinate.

Finally apply Definition 301.22. Since the $\mathsf{Real}^{\uparrow}$ terminal seal consumes only the accepted finite regular-rational packet, every seal read from the original chain factors through the repacked common window. □

Theorem 301.33 (RegSeqRat finite-window regular closure)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let W' be any finite-window restriction or refinement of a displayed window of P. If the restriction or refinement is carried by the visible* $\mathsf{StreamName}^{\uparrow}$ *schedule rows and componentwise* hsame *and* Cont *routes, then W' preserves the displayed regularity ledger, dyadic-radius rows, classifier ledger,* Pkg *provenance, and local* $\mathsf{NameCert}$ *surface of P.*

The repacked finite window remains a $\mathsf{RegSeqRat}^{\uparrow}$ *source surface for later* $\mathsf{Real}^{\uparrow}$ *and completion consumers. It introduces no quotient stream equality, host real equality, selected limit, ambient completion object, or new Real seal coordinate.*

Proof. Apply Definition 301.3. The regularity ledger, dyadic-radius observation rows, and finite-window transport face are already listed as projections of the accepted packet.

Use Definition 301.14 for restrictions and refinements. It repacks the same $\mathsf{StreamName}^{\uparrow}$ schedule rows and $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, distance, and regularity rows inside the displayed carrier and classifier surface.

For common refinements, apply Definition 301.16. The gluing theorem keeps overlap rows on the same endpoint, radius, regularity, classifier, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface. Thus later $\mathsf{Real}^{\uparrow}$ or completion consumers receive the same finite regularity closure, with no coordinate from which quotient streams, host real equality, selected limits, ambient completion objects, or extra seal data could be read. □

Theorem 301.34 (RegSeqRat StreamName cover-refinement seal handoff)**.** *Let $\mathcal{W}$ be a finite cover family of public* $\mathsf{StreamName}^{\uparrow}$ *windows over one observation map, and let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet whose scheduled reads consume every member of $\mathcal{W}$. If a finite-window* $\mathsf{Real}^{\uparrow}$ *seal handoff is read from the same packet, then that seal-facing read factors through exactly the public* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface together with the displayed* $\mathsf{InBundle}$ *rows, pointwise*

$\mathsf{Rat}^{\uparrow}$ *classifier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator ledger,* hsame*,* Cont*, and* Pkg *provenance exposed by* P*.*

Thus the cover-refinement handoff supplies no quotient stream, host real equality, hidden limit selector, metric-completeness datum, or $\mathsf{Real}^{\uparrow}$ *completeness row beyond the finite public window interface and its displayed regular-rational ledgers.*

Proof. First apply Definition 126.334. The finite cover family over the common observation map is exhausted by public $\mathsf{StreamName}^{\uparrow}$ finite-window rows, so every covered read is already visible through the public window interface and its InBundle membership rows.

Next use Definition 301.26. For each covered window consumed by P, the public schedule rows lock to the same pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows and to the $\mathsf{DyadicRatCore}^{\uparrow}$ denominator ledger carried by the accepted regular-rational packet.

Now apply Definition 301.27. The finite cover refinements and the shifted diagonal reads present one coherent finite BHist surface, with the same hsame, Cont, and Pkg rows wherever the cover windows overlap.

Finally invoke Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal consumes only that finite regular-rational source surface, so the handoff factors through the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface and cannot read a quotient stream, host real equality, hidden limit selector, metric completion, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 301.35 (RegSeqRat tail-stable window cover exhaustion)**.** *Let* $\mathcal{W}$ *be a finite cover family of public* $\mathsf{StreamName}^{\uparrow}$ *windows over one accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. If every member of* $\mathcal{W}$ *is stable under shifted-tail regularity closure and the overlaps of* $\mathcal{W}$ *glue by refined-window gluing, then every finite-window* $\mathsf{Real}^{\uparrow}$ *seal read over that packet factors through exactly the public* $\mathsf{StreamName}^{\uparrow}$ *windows, the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator and radius ledgers, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The factorization exports no quotient stream, host real equality, hidden limit selector, metric-completeness datum, completion point, or additional $\mathsf{Real}^{\uparrow}$ *seal coordinate beyond the finite tail-stable cover surface.*

Proof. Start with the cover-refinement handoff of Definition 301.34. It reads the $\mathsf{Real}^{\uparrow}$ seal only through the public $\mathsf{StreamName}^{\uparrow}$ finite-window interface, the displayed InBundle rows, pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, dyadic denominator ledger, and the stated provenance rows.

Apply Definition 301.18 to each window in the cover. Tail stability keeps the scheduled regularity rows inside the same accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet after the shifted-tail read, so the seal cannot leave the finite public cover.

Use Definition 301.16 on every overlap. The overlap rows glue the public windows into one finite source surface while preserving the pointwise classifier rows, the dyadic denominator and radius ledgers, and the displayed hsame, Cont, Pkg, and NameCert provenance.

Finally invoke Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal handoff consumes only the finite regular-rational packet just assembled, so the tail-stable cover exhausts the whole seal-facing source surface and leaves no off-table quotient, host-equality, completion, or extra seal row to read. □

Theorem 301.36 (RegSeqRat tail-stable cover common-window lock)**.** *Let* $\mathcal{W}$ *be a finite cover family of public* $\mathsf{StreamName}^{\uparrow}$ *windows over one accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Suppose the cover is tail-stable, every overlap glues by refined-window gluing, and two* $\mathsf{Real}^{\uparrow}$ *seal consumers read through members of* $\mathcal{W}$*. Then the two seal reads factor through one accepted common finite window.*

The common window exposes exactly the $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledgers, and the displayed* hsame*,*

Cont, Pkg, *and* NameCert *provenance. It exposes no quotient stream, host real equality, selected limit, metric-completeness datum, completion point, or additional* Real$^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 301.35 to the cover. Each seal consumer is forced onto the finite tail-stable cover surface, so neither read can leave the public StreamName$^{\uparrow}$ windows and their displayed regular-rational ledgers.

Next connect the two selected cover members by the finite overlap chain and apply Definition 301.32. The overlap gluing hypotheses repack that chain into one accepted common finite window whose rows are the displayed schedule, endpoint, radius, classifier, regularity, transport, package, and naming rows.

Use Definition 301.9 on the overlapping classifier reads. The determinacy row identifies the source rows seen by the two consumers, so their apparent cover-window origins are two restrictions of the same accepted common packet.

Finally invoke Definition 301.22. Since the seal handoff consumes exactly that finite regular-rational source surface, both consumers factor through the single common window and no quotient-stream, host-equality, selected-limit, completion, metric-completeness, or extra seal coordinate remains available. □

Theorem 301.37 (RegSeqRat finite-cover nerve descent)**.** *Let $\mathcal{W}$ be a finite cover family of public* StreamName$^{\uparrow}$ *windows over an accepted* RegSeqRat$^{\uparrow}$ *packet P. Suppose every nonempty finite intersection of members of $\mathcal{W}$ glues by Definition 301.16. Suppose also that the pairwise tail-stable common-window locks agree on the displayed* DyadicRatCore$^{\uparrow}$ *endpoint and radius rows, the* RegSeqRat$^{\uparrow}$ *classifier and regularity ledgers, the* hsame *and* Cont *routes, and the* Pkg *and* NameCert *provenance rows carried by P. Then all* Real$^{\uparrow}$ *seal consumers over the cover descend to one finite seal-facing source packet.*

The descended packet is a public finite StreamName$^{\uparrow}$ *source surface whose seal rows are exactly the displayed regular-rational rows named above. It does not introduce quotient streams, host real equality, selected limits, metric completeness, completion points, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. First apply Definition 301.34. Every seal-facing read over the cover is already a read from the public StreamName$^{\uparrow}$ windows and the finite regular-rational rows exposed by P.

Next use Definition 301.35. The tail-stable cover surface exhausts the available seal-facing source rows, so no seal consumer can leave the displayed schedule, classifier, dyadic, transport, package, or naming data.

For each pair of cover members, apply Definition 301.36. The pairwise locks identify the common finite-window source rows seen by the two local seal consumers. The stated agreement on endpoint, radius, classifier, regularity, hsame, Cont, Pkg, and NameCert rows makes these pairwise packets compatible along every nonempty finite intersection.

Now order the finite nerve by any enumeration of its nonempty intersections and apply Definition 301.32. The intersection gluing hypotheses repack the compatible pairwise common windows into one accepted finite source packet for the whole cover.

Finally invoke Definition 301.22. Since the terminal Real$^{\uparrow}$ seal consumes only the accepted finite regular-rational packet, all cover-local seal consumers descend through that single seal-facing source packet, with no quotient-stream, host-equality, selected-limit, completion, metric-completeness, or ambient-completeness coordinate available. □

Theorem 301.38 (RegSeqRat finite-cover nerve induction)**.** *Let $\mathcal{W}$ be a finite cover family of public* StreamName$^{\uparrow}$ *windows over an accepted* RegSeqRat$^{\uparrow}$ *packet P. Suppose every nonempty finite intersection of members of $\mathcal{W}$ glues by Definition 301.16, and suppose each lower dimensional*

face satisfies the pairwise common-window lock of Definition 301.36. Then induction over the finite nerve of $\mathcal{W}$ produces one global seal-facing source packet.

The global packet preserves exactly the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledgers, the* hsame *and* Cont *routes, and the* Pkg *and* NameCert *provenance rows carried by P.*

It introduces no quotient stream, host real equality, selected limit, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. For the base case, take a singleton nonempty intersection in the nerve and apply Definition 301.34. The single local seal read is already a public $\mathsf{StreamName}^{\uparrow}$ finite-window read from the displayed regular-rational rows of P.

For the induction step, adjoin one nonempty intersection to a nerve segment whose seal-facing packet has already been assembled. Apply Definition 301.35 to keep the new intersection inside the tail-stable public cover surface.

On each overlap between the new intersection and the assembled segment, use Definition 301.36. The lower dimensional face hypotheses identify the same endpoint, radius, classifier, regularity, hsame, Cont, Pkg, and NameCert rows on the common finite window.

Now apply Definition 301.32. The refined overlap chain repacks the previously assembled packet and the new intersection packet into one accepted finite source packet with the same displayed rows.

Finally invoke Definition 301.37. The descent theorem identifies the terminal global seal-facing source for the whole finite cover, and the construction leaves no quotient stream, host real equality, selected limit, metric completion, completion point, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate to read. □

Theorem 301.39 (RegSeqRat RealUp seal uniqueness boundary)**.** *Two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets with the same displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic-radius ledger, common refinement window, and* hsame *and* Cont *provenance determine the same finite* $\mathsf{Real}^{\uparrow}$ *seal read.*

This uniqueness is only the finite seal-boundary read. It does not assert quotient Cauchy equality, host stream equality, metric-completion equality, or completed $\mathsf{Real}^{\uparrow}$ *equality.*

Proof. Apply Definition 301.31 to compare the two seal-facing reads through one common finite source surface.

Use Definition 301.28 to identify the selected cofinal window rows, then apply Definition 301.34. The latter forces both reads through the same public $\mathsf{StreamName}^{\uparrow}$ cover-refinement boundary and leaves no quotient or completed-real coordinate to distinguish. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatUp StdBridge`

Theorem 301.40 (RegSeqRat RealUp seal route exclusion package)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read obtained from an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet factors through exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the regularity ledgers, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *provenance. Conversely, any seal-facing read assembled from precisely those rows repacks as the same* $\mathsf{Real}^{\uparrow}$ *seal route.*

This route package contains no quotient stream, no host real equality, no metric-completion point, and no $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatRealSealRouteExclusionPackage`

Proof. For the forward direction, project the accepted packet by Definition 301.12. This exposes the $\mathsf{StreamName}^{\uparrow}$ schedule rows, unary indices, endpoint rows, regularity ledgers, displayed transport, continuation routes, package provenance, and local naming provenance.

Next read the dyadic endpoint, radius, and distance data through Definition 301.13. The seal-facing distance comparisons are therefore finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rows in the same scheduled packet.

Apply Definition 301.22. It consumes exactly the regular-rational source surface assembled from the schedule, dyadic, regularity, transport, continuation, package, and naming rows just exposed.

For the converse direction, start with a read assembled from exactly those rows. Repack the rows as the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ source surface and apply Definition 301.22 to obtain the corresponding $\mathsf{Real}^{\uparrow}$ seal route.

Finally use Definition 301.39. Any two assembled reads with the same displayed route package give the same finite seal read. Since the package rows do not include quotient streams, host real equality, metric-completion points, or completeness data, none of those objects can be read by the route. □

Theorem 301.41 (RegSeqRat cover normal form for RealUp seal)**.** *Let $\mathcal{W}$ be any finite cover family of public* $\mathsf{StreamName}^{\uparrow}$ *windows consumed by an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and read by the* $\mathsf{Real}^{\uparrow}$ *seal. Then the seal read determines one canonical finite* BHist *packet consisting exactly of the public* $\mathsf{StreamName}^{\uparrow}$ *window rows, the displayed* $\mathsf{InBundle}$ *rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, regularity ledgers, classifier ledgers,* hsame *transport,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}$ *provenance.*

This normal form contains no quotient stream, host real equality, hidden limit selector, metric-completion point, $\mathsf{Real}^{\uparrow}$ *completeness row, or standard-bridge coordinate.*

Proof. Start with Definition 301.34. It restricts the seal-facing read of the finite cover to the public $\mathsf{StreamName}^{\uparrow}$ windows, the displayed $\mathsf{InBundle}$ rows, pointwise $\mathsf{Rat}^{\uparrow}$ classifier rows, dyadic ledgers, and the visible transport and provenance rows.

Apply Definition 301.31 to compare all covered seal reads through a single common finite window. The common window keeps the regularity and classifier ledgers inside the same accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet.

Use Definition 301.30 to factor the refined windows into one finite BHist packet carrying the endpoint and radius ledgers, hsame transport, Cont routes, Pkg provenance, and $\mathsf{NameCert}$ provenance.

Finally invoke Definition 301.39. Any two packets with the same displayed cover-normal rows give the same finite $\mathsf{Real}^{\uparrow}$ seal read, and the excluded quotient, host-real, completion, completeness, and bridge coordinates have no row in the canonical packet. □

Theorem 301.42 (RegSeqRat RealUp seal transport fusion)**.** *Let P and P' be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets. Suppose P' is obtained from P by transporting the visible* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, dyadic endpoint and radius ledgers, finite-window readback, regularity ledgers, and* Pkg *provenance along displayed* hsame *and* Cont *rows. Then P and P' determine the same* $\mathsf{Real}^{\uparrow}$ *finite seal route through the canonical finite* BHist *packet of Definition 301.41.*

The fused route is only the displayed finite seal boundary. It does not assert quotient-stream equality, host real equality, existence of a completion point, or equality of selected limits.

Proof. Project both seal-facing reads through Definition 301.40. The only rows available to either route are the schedule, dyadic ledger, regularity, transport, continuation, package, and naming rows.

Use Definition 301.41 to name the canonical finite BHist packet carrying those rows. The transported packet has the same normal-form coordinates because the hypotheses preserve the schedule, endpoint and radius ledgers, finite-window readback, regularity ledgers, and provenance.

Apply Definition 301.14 to move the window rows along the displayed hsame and Cont routes, and then use Definition 301.16 to repack the original and transported reads through one common finite refinement.

Finally invoke Definition 301.39. Since both reads now use the same canonical finite packet, the resulting $\mathsf{Real}^{\uparrow}$ finite seal route is the same, with no quotient stream, host real equality, completion point, or selected limit available as an extra coordinate. □

Theorem 301.43 (RegSeqRat Real seal classifier factorization)**.** *Let P and Q be displayed finite* BHist *packets carrying* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-stream readback rows, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. Suppose two* $\mathsf{Real}^{\uparrow}$ *seal routes are accepted over the same regular-stream boundary and read from P and Q. Then the two seal routes classify together exactly through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *stream classifier and the transported finite-window package obtained from these rows.*

The classifier has no coordinate for quotient Cauchy classes, host real equality, completed metric limits, or selected limits. Equality consumed by the $\mathsf{Real}^{\uparrow}$ *seal is therefore only the finite classifier factorization through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and the displayed local* NameCert *rows.*

Proof. Use Definition 301.40 on both accepted seal routes. This removes every non-displayed route component and leaves only finite packets carrying the stream schedule, dyadic ledgers, regular readback, transport, continuation, package, and naming rows.

Apply Definition 301.42 to move the endpoint and radius rows, together with the regular readback and package provenance, to a common finite window. The transported rows still present the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier boundary because the transport is entirely through displayed hsame and Cont data.

Choose the shared finite packet by Definition 301.41. In that packet the classifier row and the finite-window package are the only data available to the seal consumer.

Finally invoke Definition 301.39. The two accepted routes now read the same normal-form packet, so their $\mathsf{Real}^{\uparrow}$ seal classification is exactly the displayed $\mathsf{RegSeqRat}^{\uparrow}$ classifier transported through that finite window, with no quotient, host-real, completion, or choice coordinate introduced. □

Theorem 301.44 (RegSeqRat seal-route classifier stability)**.** *Let P and Q be accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *packets determining* $\mathsf{Real}^{\uparrow}$ *seal routes. Suppose their route packages are related by displayed* hsame *transport and* Cont *containment routes preserving the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, the regularity rows, the classifier ledgers,* Pkg *provenance, and the local* NameCert *rows. Then the two* $\mathsf{Real}^{\uparrow}$ *seal classifiers agree exactly through the transported* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface.*

The agreement is a statement about the finite route package only: no quotient stream, host real equality, completion point, selected limit, or external representative can change the classifier read at the $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Proof. Apply Definition 301.40 to the two accepted seal routes. This reduces both reads to displayed finite rows: the schedule, dyadic endpoint and radius ledgers, regularity rows, classifier ledgers, package provenance, local name rows, and the hsame/Cont transport data.

Next use Definition 301.42. The displayed transport moves both route packages to one canonical finite packet while preserving the stream schedule, dyadic ledgers, regularity rows, and classifier surface.

Apply Definition 301.43 in that canonical packet. The seal equality is identified with the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier and not with any extra real-number coordinate.

Finally invoke Definition 301.39. Since the canonical read has no quotient stream, host equality, completion point, or selected limit field, the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier is the unique stable seal-route classifier. □

Theorem 301.45 (RegSeqRat Real seal classifier exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal classifier read between two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets is exactly the finite route package consisting of the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, the regularity rows, the transported* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface, the displayed* hsame *transports, the* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *rows.*

Conversely, any packet carrying precisely those rows supplies the same finite $\mathsf{Real}^{\uparrow}$ *seal classifier read. The exhaustion statement has no coordinate for quotient streams, host real equality, completion points, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First reduce the read to its visible route package. By Definition 301.40, an accepted seal read exposes only the schedule, dyadic ledger, regularity, transport, continuation, package, and local naming rows, and excludes the non-displayed real-number coordinates named in the statement.

Next put the exposed rows into normal form. Apply Definition 301.41 to obtain the canonical finite BHist packet whose rows are exactly the public $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint and radius ledgers, regularity and classifier ledgers, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

The classifier component is then forced by Definition 301.43. In the normal-form packet the $\mathsf{Real}^{\uparrow}$ seal classifier reads only the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface and the finite-window package.

Finally use Definition 301.44. Transport along the displayed hsame and Cont rows preserves the classifier read, so any packet with precisely the displayed rows gives the same finite seal classifier. Since none of the four cited boundaries contains quotient streams, host real equality, completion points, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, the exhaustion has no further read channel. □

Theorem 301.46 (RegSeqRat Real seal route replay)**.** *Let a finite route package list exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, regularity rows, transported* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface, displayed* hsame *rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows named by Definition 301.45. Replaying those rows through the exclusion package, the cover normal form, the displayed transport fusion, and the classifier exhaustion reconstructs the same finite* $\mathsf{Real}^{\uparrow}$ *seal route.*

The replay has no coordinate for a quotient stream, host real equality, completion point, selected limit, or $\mathsf{Real}^{\uparrow}$ *completeness. Hence any consumer that receives the displayed package receives exactly the same finite seal route as the original accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. First project the input package by Definition 301.40. The row inventory is precisely the finite schedule, dyadic ledger, regularity, transport, continuation, package, and local naming surface, with no hidden real-number coordinate available to a replaying consumer.

Next apply Definition 301.41. The displayed rows are repacked as the canonical finite BHist packet whose windows, ledgers, classifier rows, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance are the rows of the input package.

Use Definition 301.42 to preserve the route under the displayed endpoint and classifier transports. The transport step does not add a stream quotient or a completion coordinate; it only relabels the finite rows already present in the normal-form packet.

Finally invoke Definition 301.45. The classifier read is exhausted by the transported $\mathsf{RegSeqRat}^{\uparrow}$ surface and the finite route package, so replaying the same displayed rows reconstructs the same $\mathsf{Real}^{\uparrow}$ seal route and no additional limit-like datum. □

Theorem 301.47 (RegSeqRat Real seal replay ledger lock)**.** *Any replay supplied by Definition 301.46 factors through the same finite* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, unary index rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, regularity rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, displayed* InBundle *rows, transported classifier surface,* Pkg *provenance, local* NameCert *rows, displayed* hsame *transports, and visible* Cont *routes named by Definition 301.43 and Definition 301.45. Consequently replay cannot change the finite regularity, radius, provenance, bundle, package, or classifier ledger rows of the accepted* $\mathsf{Real}^{\uparrow}$ *seal route.*

The ledger lock contains no coordinate for quotient-real equality, host equality, a completion point, trichotomy, a selected representative, or choice data.

Proof. Start with the replay package of Definition 301.46. That theorem reconstructs the $\mathsf{Real}^{\uparrow}$ seal route only from the displayed schedule, dyadic, regularity, transport, continuation, package, and local naming rows.

Apply Definition 301.43. The classifier part of the replay factors through the transported $\mathsf{RegSeqRat}^{\uparrow}$ stream classifier and the finite-window package over the same endpoint, radius, regular readback, hsame, Cont, Pkg, and NameCert rows.

Now apply Definition 301.45. The visible classifier read is exhausted by precisely those schedule, endpoint, radius, regularity, transported classifier, hsame, Cont, package, and local name rows. There is therefore no second row inventory through which replay could alter the regularity, radius, provenance, bundle, package, or classifier ledgers.

The cited theorem surfaces expose finite route rows only. None contains quotient-real equality, host equality, a completion point, trichotomy, a selected representative, or choice data, so replay has no channel by which to replace the locked ledger rows. □

Theorem 301.48 (RegSeqRat Real seal schedule coverage)**.** *Every finite schedule accepted as a* $\mathsf{Real}^{\uparrow}$ *seal read from accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets is consumed by exactly the public* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, regularity rows, transported* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface, displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Conversely, any route package containing exactly those rows replays to the same finite* $\mathsf{Real}^{\uparrow}$ *seal route.*

The coverage assertion does not add quotient real carriers, host real equality, selected limits, metric-completion points, or $\mathsf{Real}^{\uparrow}$ *completeness. It only states that the accepted finite schedule has no read channel outside the displayed package rows.*

Proof. First apply Definition 301.40. An accepted $\mathsf{Real}^{\uparrow}$ seal read from accepted $\mathsf{RegSeqRat}^{\uparrow}$ packets has only the finite schedule, dyadic ledger, regularity, transport, continuation, package, and local naming rows available to the route.

Next apply Definition 301.41. The exposed rows are repacked as the canonical finite BHist packet whose row inventory is exactly the public $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint and radius ledgers, regularity rows, transported classifier rows, hsame, Cont, Pkg, and NameCert provenance.

Then use Definition 301.45. The classifier component is fully read from the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface together with the finite route package, so no further classifier coordinate remains to be consumed.

For the converse, apply Definition 301.46. A package containing exactly the displayed rows reconstructs the same finite $\mathsf{Real}^{\uparrow}$ seal route, and the exclusion boundary prevents replay from reading quotient reals, host equality, selected limits, completion points, or completeness data. □

Theorem 301.49 (RegSeqRat Real tail classifier seal)**.** *For accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets whose public* $\mathsf{StreamName}^{\uparrow}$ *tail windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, transported* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, displayed* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows agree, the* $\mathsf{Real}^{\uparrow}$ *seal consumer reads exactly the same tail classifier on both sides.*

The boundary is only the finite seal-classifier boundary. It excludes quotient streams, host real equality, selected limits, completion points, and $\mathsf{Real}^{\uparrow}$ *completeness rows as classifier coordinates.*

Proof. First apply Definition 301.45 to both accepted reads. Each read is reduced to the finite route package consisting of the public tail windows, dyadic endpoint and radius ledgers, transported classifier rows, hsame, Cont, Pkg, and local NameCert provenance.

Next use Definition 301.50. The shared public rows lock the consumer schedule and replay ledger, so the two reads cannot choose different finite handoff packages for the same displayed tail surface.

Finally invoke Definition 301.51. The locked schedules replay to one terminal finite $\mathsf{Real}^{\uparrow}$ seal route, and the exhaustion row identifies its readback classifier with the transported $\mathsf{RegSeqRat}^{\uparrow}$ tail classifier. Since the cited boundaries contain no quotient stream, host real equality, selected limit, completion point, or $\mathsf{Real}^{\uparrow}$ completeness row, the common transported tail classifier is exactly the classifier read by the $\mathsf{Real}^{\uparrow}$ seal consumer. □

Theorem 301.50 (RegSeqRat Real seal public handoff determinacy)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *stream packet factors through the finite schedule-coverage row of Definition 301.48, the replay-ledger lock of Definition 301.47, the classifier exhaustion row of Definition 301.45, and the handoff row* BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_real_seal_handoff*. If two accepted public handoff routes have the same finite schedule and the same replay ledger, then they expose the same* $\mathsf{Real}^{\uparrow}$ *readback classifier.*

The conclusion is a deterministic consumer boundary for the displayed finite seal package. It does not introduce quotient-real equality, host equality, selected limits, metric-completion points, or $\mathsf{Real}^{\uparrow}$ *completeness as readback coordinates.*

Proof. First use Definition 301.48. The public consumer has only the finite $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint and radius ledgers, regularity rows, transported classifier rows, BHist packet rows, hsame, Cont, Pkg, and local NameCert rows available for a seal read.

Next apply Definition 301.47. Equality of the schedule and replay ledger locks the accepted route package, so the consumer cannot choose a second route through a different dyadic-radius, regularity, continuation, package, or local naming row.

Then invoke Definition 301.45. The visible classifier is exhausted by the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface together with the locked route package, leaving no extra seal case for the readback classifier.

Finally use the displayed handoff row BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_real_seal_hand That row is the finite $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ seal handoff already named by the chapter closure target, so the common schedule and replay ledger yield one exposed $\mathsf{Real}^{\uparrow}$ readback classifier. □

Theorem 301.51 (RegSeqRat Real seal schedule confluence)**.** *Let S and T be finite schedules accepted as* $\mathsf{Real}^{\uparrow}$ *seal reads from accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets. Suppose the two accepted reads expose the same public* $\mathsf{StreamName}^{\uparrow}$ *window rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius*

ledgers, the same transported $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface, the same visible* BHist, hsame, Cont, *and* Pkg *rows, and the same local* $\mathsf{NameCert}$ *rows. Then* S *and* T *replay to one and the same terminal finite* $\mathsf{Real}^{\uparrow}$ *seal route.*

This confluence is only the finite seal-route assertion over the displayed packet rows. It excludes quotient reals, host equality, selected limits, completion points, and $\mathsf{Real}^{\uparrow}$ *completeness as route coordinates.*

Proof. First apply Definition 301.48 to both accepted schedules. Each accepted read is consumed by exactly the displayed $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint and radius ledgers, transported classifier surface, BHist packet rows, hsame, Cont, Pkg provenance, and local $\mathsf{NameCert}$ rows named in the statement.

Next use Definition 301.46. Since the two row inventories are the same, the replay construction applied to S and to T reconstructs seal routes from the same finite route package.

Apply Definition 301.45 to the common package. The transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface exhausts the visible classifier read, so the two replays cannot diverge by choosing a second classifier, package, or local naming ledger.

Finally invoke Definition 301.39. The two replayed routes now have the same displayed schedule, dyadic-radius ledger, common finite window, hsame provenance, and Cont provenance, hence they determine one terminal finite $\mathsf{Real}^{\uparrow}$ seal route. The cited boundaries contain no quotient-real carrier, host equality, selected limit, completion point, or $\mathsf{Real}^{\uparrow}$ completeness datum, so confluence has no hidden real-number coordinate. □

Theorem 301.52 (RegSeqRat Real seal endpoint coverage)**.** *Every accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal endpoint factors through exactly the displayed endpoint package with four open-phase faces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and transported classifier rows;*
4. *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

The carrier data of the package are only BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. Conversely, any accepted seal endpoint with those four faces and those carrier rows has no additional route coordinate beyond the terminal finite* $\mathsf{Real}^{\uparrow}$ *seal endpoint named by the schedule.*

In particular, the endpoint package excludes quotient streams, host equality, selected limits, completion points, $\mathsf{Real}^{\uparrow}$ *completeness, and metric-completeness rows from the finite endpoint surface.*

Proof. First apply Definition 301.48. It gives the finite schedule inventory for any accepted endpoint, so the visible rows are the $\mathsf{StreamName}^{\uparrow}$ windows, dyadic endpoint and radius ledgers, regularity rows, transported classifier rows, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Next apply Definition 301.40. This locks the route coordinates to the displayed finite package and removes quotient streams, host equality, selected limits, completion points, $\mathsf{Real}^{\uparrow}$ completeness, and metric-completeness data from the accepted endpoint read.

Now invoke Definition 301.45. The transported $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier rows exhaust the visible classifier surface, so there is no second classifier face, package row, or local naming coordinate that can refine the endpoint.

Finally use Definition 301.51. Any two accepted reads with the same four displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert rows replay to one terminal finite Real$^{\uparrow}$ seal route. Hence the four-face package is exactly the finite endpoint coverage package. □

Theorem 301.53 (RegSeqRat Real seal endpoint phase boundary)**.** *Every accepted finite* RegSeqRat$^{\uparrow}$*-to-*Real$^{\uparrow}$ *seal endpoint covered by Definition 301.52 is precisely the* RegSeqRat$^{\uparrow}$ *face of the four-face Phase Real and Completion package of Definition 38.290. Conversely, the* RegSeqRat$^{\uparrow}$ *face of that phase package is readable at the local seal endpoint only through the same finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledgers,* RegSeqRat$^{\uparrow}$ *regularity and classifier rows, terminal* Real$^{\uparrow}$ *seal row, and* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows.*

Thus the local seal endpoint is a boundary face of the phase package, not a separate quotient-real object. It carries no host equality, selected limit, completion point, Real$^{\uparrow}$ *completeness row, metric-completeness datum, or standard bridge.*

Proof. First apply Definition 301.52. It identifies the accepted local endpoint with exactly four displayed faces: StreamName$^{\uparrow}$ finite-window rows, DyadicRatCore$^{\uparrow}$ endpoint and radius ledger rows, RegSeqRat$^{\uparrow}$ regularity and transported classifier rows, and the terminal finite Real$^{\uparrow}$ seal row.

Next use Definition 38.290. Its four public packages are the DyadicRatCore observation package, the StreamName$^{\uparrow}$ countable-name and finite-window packages, the RegSeqRat stream-name scoped interface, and the Real$^{\uparrow}$ public certificate export through the StreamName$^{\uparrow}$ interface. The third item is exactly the regular rational stream face through which the local endpoint coverage reads its RegSeqRat$^{\uparrow}$ rows.

For the converse, read that third phase item back through the same public finite windows and dyadic ledgers. The phase boundary exposes no independent RegSeqRat$^{\uparrow}$ consumer coordinate beyond the transported regularity and classifier rows, so the local read is forced to use the finite StreamName$^{\uparrow}$ windows, dyadic endpoint and radius ledgers, terminal Real$^{\uparrow}$ seal row, and the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Finally combine the exclusions from both cited boundaries. The local endpoint coverage excludes quotient streams, host equality, selected limits, completion points, Real$^{\uparrow}$ completeness, and metric-completeness rows; the phase boundary also withholds quotient, completion, arithmetic, order, metric-completeness, Classical, and standard-bridge data. Hence the RegSeqRat$^{\uparrow}$ seal endpoint is only that finite phase face and carries none of the excluded data as a hidden coordinate. □

Theorem 301.54 (RegSeqRat Real seal finite-window witness induction)**.** *Every accepted finite* RegSeqRat$^{\uparrow}$*-to-*Real$^{\uparrow}$ *seal endpoint covered by Definition 301.52 admits induction over the finite request window. The empty window is the terminal* Real$^{\uparrow}$ *seal row. If a witness package is available for a finite request window W, then adjoining one visible request extends it by the* StreamName$^{\uparrow}$ *finite-window transport row, the* DyadicRatCore$^{\uparrow}$ *radius-ledger readback, and the* RegSeqRat$^{\uparrow}$ *classifier transport row.*

The induction is only a finite-window construction for the endpoint face of Definition 301.53. It introduces no quotient stream equality, selected limit, host real equality, choice, Real$^{\uparrow}$ *completeness, or metric-completeness datum.*

Proof. First read the endpoint package through Definition 301.48 and Definition 301.52. These two results expose the finite request inventory: StreamName$^{\uparrow}$ windows, dyadic endpoint and radius ledgers, transported RegSeqRat$^{\uparrow}$ regularity and classifier rows, the terminal Real$^{\uparrow}$ seal row, and the visible BHist, hsame, Cont, Pkg, and NameCert provenance.

For the base window, take the terminal Real$^{\uparrow}$ seal row from the endpoint coverage package. With no visible request to adjoin, the witness package has no dyadic-radius readback or classifier-transport obligation beyond that terminal row and its displayed provenance.

For the step, suppose the witness has been built over a finite window W. Adjoin one visible request through Definition 301.6; this repacks the same $\mathsf{StreamName}^{\uparrow}$ finite-window rows, hsame comparisons, Cont routes, Pkg rows, and local NameCert provenance on the enlarged window.

Next read the new radius and endpoint rows by Definition 301.13. Then apply Definition 301.7 to carry the transported regularity and classifier surface to the enlarged window. This gives exactly the witness package required after adjoining the request.

Finally use Definition 301.53. The endpoint is the $\mathsf{RegSeqRat}^{\uparrow}$ face of the Phase Real and Completion package, so the induction cannot pick up any coordinate withheld by that boundary. Together with the exclusions in the endpoint coverage theorem, this rules out quotient stream equality, selected limits, host real equality, choice, $\mathsf{Real}^{\uparrow}$ completeness, and metric-completeness data at every finite stage. □

Theorem 301.55 (RegSeqRat paperBridge audit-threshold locality)**.** *Let $\mathcal{R}$ be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source satisfying the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read. Then $\mathcal{R}$ remains admissible for that read exactly when the four face records*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

are read objectwise at the scoped and audit-clean threshold on the same finite source, even if the $\mathsf{RegSeqRat}^{\uparrow}$ *closurestatus bridge row is* `paperBridge`*. The pending bridge row is local to the* $\mathsf{RegSeqRat}^{\uparrow}$ *face. It exports no aggregate bridge token, quotient stream, selected limit, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 301.219. It identifies the accepted audit-threshold read on $\mathcal{R}$ with the RegSeqRat four-face exit-threshold equivalence together with the local closurestatus and formalstatus records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 301.222. The status interface of the read contains only the four face records and their own closurestatus, formalstatus, leantarget, and bridgestatus rows. Thus a `paperBridge` row on the $\mathsf{RegSeqRat}^{\uparrow}$ face is read as that face's local bridge row, not as a route credential and not as a failed aggregate bridge row.

The same objectwise attachment is the status pullback square of Definition 301.223. Both routes through the square pass through the same four faces, so the pending bridge row cannot move to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, or the Phase Real and Completion route as a separate aggregate coordinate.

Finally compare this local reading with the downstream gate in Definition 1327.28. That gate asks for the four local status records at the scoped and audit-clean threshold and for the same finite support rows; it does not ask for componentwise bridgeChecked status. Hence the RegSeqRat `paperBridge` row is local bridge debt only, while quotient streams, selected limits, metric-completeness data, aggregate bridge tokens, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the accepted read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The obligation boundary consists of the stream carrier, dyadic classifier, regularity rows, finite-window transport rows, and $\mathsf{Real}^{\uparrow}$ seal-handoff rows over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, and Pkg.

Constructive story: Construct the packet by finite scheduled reads: $\mathsf{StreamName}^{\uparrow}$ supplies the visible windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies endpoint and radius ledgers, and the regularity and classifier rows are projections of those finite observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier\_real\_seal\_handoff`

Bridge status: bridgeChecked

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient carriers, metric completeness, or standard bridge data.

Upgrade path: No further upgrade is pending because this chapter is already at the top formal bridge axis.

Theorem 301.56 (RegSeqRat seal-route visible source normal form)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *seal route consumed from an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet factors through one visible finite source surface. That surface consists exactly of the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows and transported classifier rows, the finite* BHist *packet rows, the displayed* hsame *transports, the visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows.*

Conversely, any finite route package carrying exactly those rows replays to the same terminal finite $\mathsf{Real}^{\uparrow}$ *seal route. The normal form has no coordinate for quotient stream equality, host real equality, a selected limit, a metric-completion point,* $\mathsf{Real}^{\uparrow}$ *completeness, or standard-bridge data.*

Proof. Begin with the accepted seal route and apply Definition 301.40. The route can read only the displayed schedule, dyadic, regularity, transport, continuation, package, and local naming rows, and it has no channel for the excluded real-number or bridge coordinates.

Next apply Definition 301.41. The exposed rows are repacked as one canonical finite BHist packet carrying the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance ledgers, regularity and classifier rows, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Use Definition 301.45 inside that packet. It identifies the classifier component with the transported $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface and shows that no quotient stream, host equality, completion point, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness row remains to be read.

Finally invoke Definition 301.51. Any two accepted seal routes exposing the same displayed rows replay through one terminal finite $\mathsf{Real}^{\uparrow}$ seal route. Thus the common finite packet is the unique visible source surface consumed by the $\mathsf{Real}^{\uparrow}$ seal handoff, and every accepted route factors through precisely that surface. □

Theorem 301.57 (RegSeqRat seal-route ordered face determinacy)**.** *Let an accepted terminal seal route carry a* $\mathsf{RegSeqRat}^{\uparrow}$ *packet to the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and suppose its visible source rows are exactly the normal-form rows of Definition 301.56. Then the route consumes those rows in the ordered four-face route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

An accepted reindexing gives the same visible source section only when it preserves that order and preserves the same BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *support rows.*

The determinacy statement has no coordinate for quotient streams, selected limits, host equality, metric completion, $\mathsf{Real}^{\uparrow}$ *completeness, or standard bridge rows.*

Proof. Apply Definition 301.56. The accepted terminal seal route exposes one finite source surface, and that surface is exactly the StreamName schedule, DyadicRatCore endpoint and ledger, RegSeqRat regularity and classifier, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ inventory.

Next use Definition 301.65. The same inventory is consumed as a four-face terminal boundary rather than as an ambient completion or verification-axis token. Thus the visible section must read first the $\mathsf{StreamName}^{\uparrow}$ schedule face, then the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, then the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier face, and finally the $\mathsf{Real}^{\uparrow}$ seal face.

It remains to compare an accepted reindexing. Apply Definition 126.576 to the StreamName-facing seal handoff and then return to the normal form. A reindexed section reaches the same

terminal seal section precisely by preserving the displayed ordered faces together with the same BHist, hsame, Cont, Pkg, and local NameCert rows.

The exclusion clause follows from the coordinate inventories of the cited routes. Those inventories contain no quotient stream row, selected limit, host-equality row, metric-completion coordinate, $\mathsf{Real}^{\uparrow}$ completeness row, or standard bridge row, so no accepted reindexing can use such data to change the face order while keeping the same visible source section. □

Theorem 301.58 (RegSeqRat four-face terminal pullback)**.** *Every accepted Phase Real terminal read and every completion-facing terminal read obtained from a* $\mathsf{RegSeqRat}^{\uparrow}$ *packet factors through one finite* BHist *pullback with exactly four source faces:*

1. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *scheduled window rows and displayed bundle rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows, classifier rows, and provenance rows;*
4. *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

If two terminal reads expose the same four faces, then they classify together through the displayed hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows of that pullback.*

The terminal pullback exports no quotient stream equality, host sequence equality, selected limit, host real equality, ambient completeness, metric-completion point, or standard bridge.

Proof. Start with Definition 301.25. It places the $\mathsf{StreamName}^{\uparrow}$ schedule rows and the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers inside the same accepted finite $\mathsf{RegSeqRat}^{\uparrow}$ window.

Next apply Definition 301.56. The terminal $\mathsf{Real}^{\uparrow}$ seal route can read only the visible finite source surface formed from the schedule, dyadic ledgers, regularity rows, classifier rows, BHist packet rows, hsame, Cont, Pkg, and local NameCert provenance.

Use Definition 301.40 to remove every off-surface route component. The remaining route package is exactly the finite inventory from which the four faces of the pullback are read.

Apply Definition 301.43. Agreement on the four faces transports the two terminal reads through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface together with the visible hsame, Cont, Pkg, and local NameCert rows.

Finally invoke Definition 301.51. The common displayed inventory replays to one terminal finite $\mathsf{Real}^{\uparrow}$ seal route, and the cited exclusion package leaves no coordinate for quotient stream equality, host sequence equality, selected limits, host real equality, ambient completeness, metric-completion points, or standard bridge data. □

Theorem 301.59 (RegSeqRat four-face terminal route uniqueness)**.** *Let two accepted terminal reads from* $\mathsf{RegSeqRat}^{\uparrow}$ *packets expose the same four-face pullback inventory: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *scheduled-window and bundle rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. Then the two terminal routes classify together through the same displayed* hsame, Cont, Pkg, *and local* NameCert *rows of that inventory.*

This route uniqueness introduces no quotient stream equality, selected host limit, metric-completion point, $\mathsf{Real}^{\uparrow}$ *completeness datum, or standard bridge.*

Proof. Apply Definition 301.58. The shared four faces are already one finite pullback surface, so both terminal reads are read through the same displayed endpoint ledgers, schedules, regularity rows, classifier rows, provenance rows, and terminal seal row.

Use Definition 301.40. It removes all off-surface coordinates from the terminal route and leaves only the visible hsame, Cont, Pkg, and local NameCert rows for the accepted read.

Next apply Definition 301.43. Since the two reads agree on the four faces, the classifier comparison factors through the same displayed rows rather than through an external completion or quotient coordinate.

Finally use Definition 301.51. The common scheduled-window inventory has a single terminal replay route to the $\mathsf{Real}^{\uparrow}$ seal row, so the terminal routes classify together through the same finite RegSeqRat evidence, with no quotient stream equality, selected host limit, metric-completion point, $\mathsf{Real}^{\uparrow}$ completeness datum, or standard bridge. □

Theorem 301.60 (RegSeqRat terminal four-face correspondence)**.** *Two accepted terminal reads from* $\mathsf{RegSeqRat}^{\uparrow}$ *packets classify together exactly when their visible four-face inventories agree. The four faces are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *scheduled-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

The agreement is transported only by the displayed BHist *rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The correspondence exports no quotient stream equality, host sequence equality, selected limit, host real equality, metric-completion point, completeness datum, or standard bridge row.*

Proof. For the forward direction, apply Definition 301.58. Any accepted terminal read from a $\mathsf{RegSeqRat}^{\uparrow}$ packet already factors through one finite surface with the DyadicRatCore ledger face, StreamName schedule face, RegSeqRat regularity and classifier face, and terminal $\mathsf{Real}^{\uparrow}$ seal face.

Next use Definition 301.59. Once two accepted terminal reads classify together on that finite surface, their classifier route is forced to be the displayed route through the same BHist, hsame, Cont, Pkg, and local NameCert rows. Thus the only visible inventory on which the shared classification can be read is the four-face inventory named in the statement.

For the reverse direction, suppose the two terminal reads expose agreeing four-face inventories with the displayed transport rows. Apply Definition 301.58 to repack the two inventories as one finite pullback surface.

Finally use Definition 301.40 and Definition 301.51. The exclusion package removes quotient streams, selected limits, host equality, metric-completion points, completeness rows, and standard bridge rows from the terminal route; schedule confluence leaves a single terminal replay to the $\mathsf{Real}^{\uparrow}$ seal row. The remaining route is exactly the displayed BHist, hsame, Cont, Pkg, and local NameCert route, so the two terminal reads classify together. □

Theorem 301.61 (RegSeqRat four-face terminal classifier completeness)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal classifier is complete for accepted terminal reads exactly at the four visible faces: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *schedules; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Thus two accepted terminal reads classify together precisely when these four faces agree through the displayed BHist, hsame, Cont, Pkg, *and local* NameCert *rows. No quotient stream equality, host sequence equality, selected limit, host real equality, metric-completion point, ambient completeness datum, or standard bridge row is part of the classifier completion surface.*

Proof. First assemble the classifier surface. Apply Definition 301.58; every accepted terminal read factors through one finite pullback whose visible faces are exactly the DyadicRatCore ledger face, StreamName schedule face, RegSeqRat regularity/classifier/provenance face, and terminal $\mathsf{Real}^{\uparrow}$ seal face.

Next use route uniqueness. By Definition 301.59, agreement on those four faces forces the two terminal routes to classify through the same displayed hsame, Cont, Pkg, and local NameCert rows.

For the converse, apply Definition 301.60. If two accepted terminal reads classify together, their visible four-face inventories agree, and if the four visible faces agree, the accepted terminal reads classify together.

The exclusions in the cited pullback and correspondence statements remove quotient stream equality, host sequence equality, selected limits, host real equality, metric-completion points, ambient completeness, and standard bridge rows from the terminal classifier. Hence the four-face surface is complete for accepted terminal reads. □

Theorem 301.62 (RegSeqRat dyadic-tail Real-seal status exhaustion)**.** *Every accepted terminal L10 read through* $\mathsf{RegSeqRat}^{\uparrow}$ *is exhausted by the status ledger formed from exactly four faces: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *scheduled-window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. Transport across this ledger uses only the displayed* BHist *rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The ledger contains no quotient stream equality, host equality, selected limit, ambient completeness, metric-completion point, countable-choice row, verification-axis promotion, or standard bridge row.

Proof. First apply Definition 301.58. It places each accepted terminal L10 read on the single finite pullback whose faces are the DyadicRatCore ledger face, StreamName schedule face, RegSeqRat regularity and classifier face, and terminal $\mathsf{Real}^{\uparrow}$ seal face.

Next use Definition 301.61. The classifier is complete exactly on those four visible faces, so every accepted terminal read is accounted for by the displayed status rows rather than by an external completion surface.

Finally combine Definition 301.40 and Definition 301.51. The exclusion package removes quotient stream equality, host equality, selected limits, metric-completion points, countable-choice rows, verification-axis promotion, and standard bridge rows; schedule confluence keeps the replay on the same displayed BHist, hsame, Cont, Pkg, and local NameCert inventory. Thus the four-face status ledger is exhaustive. □

Theorem 301.63 (RegSeqRat finite-cover terminal classifier descent)**.** *Let* $\{W_i\}_{i\in I}$ *be a finite family of certified* $\mathsf{StreamName}^{\uparrow}$ *windows for a* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal read. Suppose the pairwise and triple overlaps satisfy the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *source cocycle and seal cocycle hypotheses, and suppose the induced handoffs consume the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers on overlaps. Then the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal classifier surface is exactly the pasted four-face inventory consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers; the* $\mathsf{StreamName}^{\uparrow}$ *window rows; the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows; and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

The descent transport uses only the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows. It introduces no quotient streams, selected host limits, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness rows, or standard bridge rows.*

Proof. First paste the finite StreamName cover. Apply Definition 126.551 to the certified windows and their pairwise and triple source cocycle data. Apply Definition 126.552 to the same cover with the seal cocycle data. These two finite-cover statements keep the pasted source and seal on the displayed BHist, hsame, Cont, Pkg, and NameCert rows, and they keep the DyadicRatCore ledgers shared on overlaps.

Next identify the pasted cover with the terminal classifier surface. By Definition 301.61, an accepted terminal $\mathsf{RegSeqRat}^{\uparrow}$ read is classified completely by the four visible faces: the DyadicRatCore ledger face, the StreamName window face, the RegSeqRat regularity/classifier/provenance face, and the terminal $\mathsf{Real}^{\uparrow}$ seal face. The pasted finite cover supplies exactly those rows, so no extra classifier coordinate is available.

Finally keep the terminal seal on the same four-face route. Use Definition 301.62 to exhaust the $\mathsf{RegSeqRat}^{\uparrow}$ terminal status by the displayed four-face ledger, and use Definition 38.455 to read the terminal $\mathsf{Real}^{\uparrow}$ seal as the same four-face packet. The exclusions in these route statements remove quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric-completeness rows, and standard bridge rows. Hence the descended classifier is precisely the pasted terminal four-face inventory. □

Theorem 301.64 (RegSeqRat finite-cover terminal refinement invariance)**.** *Let $\{W_i\}_{i\in I}$ and $\{V_j\}_{j\in J}$ be two finite certified* $\mathsf{StreamName}^{\uparrow}$ *cover presentations of the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal read. Suppose both covers satisfy the finite source and seal cocycle hypotheses of Definition 301.63. Suppose also that they admit a certified finite common refinement whose refinement maps preserve the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *support.*

Then the two cover presentations descend to the same four-face terminal classifier surface. The invariant surface is exactly the visible $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face,* $\mathsf{StreamName}^{\uparrow}$ *window face,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity/classifier/provenance face, and terminal* $\mathsf{Real}^{\uparrow}$ *seal face. It contains no quotient stream, host sequence equality, selected limit, host real equality, countable-choice row, ambient or metric completeness datum, or standard bridge row.*

Proof. Apply Definition 301.63 to the $\{W_i\}_{i\in I}$ cover. The first presentation descends to the pasted four-face classifier surface carried by its DyadicRatCore ledgers, StreamName windows, RegSeqRat rows, terminal $\mathsf{Real}^{\uparrow}$ seal, and displayed BHist, hsame, Cont, Pkg, and local NameCert support.

Apply the same descent theorem to the $\{V_j\}_{j\in J}$ cover. The second presentation also descends to a pasted four-face classifier surface with the same kind of visible support and with the same exclusion clauses.

Use the certified common refinement to compare the two pasted surfaces. By hypothesis, every refinement row preserves the DyadicRatCore endpoint, radius, and distance ledgers, the StreamName window rows, the RegSeqRat regularity, classifier, and provenance rows, the terminal $\mathsf{Real}^{\uparrow}$ seal row, and the displayed BHist, hsame, Cont, Pkg, and local NameCert support. Thus the two descended presentations have identical visible four-face inventories.

Now apply Definition 301.61 and Definition 301.60. Completeness says that accepted terminal reads are classified exactly by those four visible faces, and the correspondence identifies equality of the visible inventories with equality of the terminal classifier surface.

The exclusion clauses are inherited from the two descent applications and from the terminal pullback route. Hence refinement invariance is witnessed inside the displayed BEDC rows only: no quotient stream, host sequence equality, selected limit, host real equality, countable choice, ambient or metric completeness, or standard bridge is introduced. □

301.3 RegSeqRat Terminal Four-Face Axis Separation

Theorem 301.65 (RegSeqRat terminal four-face axis separation)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal four-face read through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

supplies only the paper-side finite boundary whose support rows are

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The read cannot raise a formal-status token for any of the four faces, create a standard bridge row, assert ambient $\mathsf{Real}^{\uparrow}$ *completeness, introduce quotient stream equality, or supply metric-completeness data.*

Thus downstream Phase Real and Completion consumers may use the terminal four-face packet only as a finite manuscript boundary. Verification status, bridge status, quotient identification, and completeness data must still be read from the relevant face's own closurestatus, Lean target, and bridge boundary.

Proof. Start with Definition 301.60. It identifies terminal RegSeqRat agreement exactly with the visible four-face inventory and transports that agreement only through the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Next apply Definition 301.68. The route is not an isolated RegSeqRat coordinate: it passes through the DyadicRatCore, StreamName, and Real sibling chapters before it is consumed as a Real-facing handoff, while retaining the same finite support rows.

Now invoke Definition 126.500. It reads the Phase Real and Completion exit packet as exactly the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row.

Finally use Definition 38.429. The Real-facing completion interface consumes the same finite window, regular-rational readback, dyadic observation ledger, seal row, componentwise hsame transport, displayed Cont route, package provenance, and local NameCert row. Since the cited routes contain no verification-axis token, standard bridge row, ambient completeness principle, quotient stream equality, or metric-completeness coordinate, the terminal four-face read separates the paper-side boundary from those independent axes. □

Theorem 301.66 (RegSeqRat terminal four-face status pullback)**.** *An accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal four-face read satisfies the current Phase Real and Completion objectwise status boundary if and only if it exposes the four local face records for*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

where each face is read from its own closurestatus*,* formalstatus*,* leantarget*, and* bridgestatus *rows, together with the shared support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The pullback contains no aggregate formalstatus *row, aggregate bridge token, selected limit, quotient stream equality, metric-completeness datum, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. For necessity, begin with Definition 301.65. It separates the terminal four-face paper boundary from verification status, bridge status, quotient identification, selected limits, metric-completeness data, countable choice, and ambient real completeness.

Next apply Definition 301.60. The accepted RegSeqRat terminal read has exactly the visible four-face inventory, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback rows are one of those local faces rather than an aggregate route token.

Now use Definition 126.500. The current Phase Real and Completion route consumes precisely the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

For sufficiency, start with the four local face records and the shared support rows. Apply Definition 291.112 to read the local $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ status rows from their own closurestatus, formalstatus, leantarget, and bridgestatus entries.

Finally invoke Definition 38.792. The Real-side criterion consumes exactly those objectwise face records together with the displayed finite support, so the accepted RegSeqRat terminal read satisfies the current Phase Real and Completion objectwise status boundary.

The exclusions follow from the same citations. The axis-separation theorem and the dyadic pullback theorem expose no aggregate formalstatus, aggregate bridge, selected limit, quotient stream equality, metric-completeness datum, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, and the four-object route plus the Real objectwise criterion provide no further row in which such data could be stored. □

Theorem 301.67 (RegSeqRat terminal four-face finite-request restriction)**.** *Let $\mathcal{T}$ be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal four-face read satisfying the current Phase Real and Completion objectwise status boundary. Let $\mathcal{R}$ be any finite request restriction of $\mathcal{T}$ preserving the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local* $\mathsf{Real}^{\uparrow}$ *face rows, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows. Then $\mathcal{R}$ satisfies the same objectwise boundary.*

Conversely, if such a finite request restriction satisfies that boundary, then the boundary recovers exactly those displayed face and support rows of the ambient read. It recovers no aggregate $\mathsf{formalstatus}$ *row, aggregate bridge token, selected limit, quotient stream equality, metric-completeness datum, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, or fifth public coordinate.*

Proof. Start with Definition 301.66. It reads the ambient accepted terminal read through exactly the four local face records and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Restrict those rows along the finite request. Since the restriction preserves the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ faces and the displayed support rows, the same objectwise boundary remains visible on $\mathcal{R}$.

For the reverse direction, suppose the finite request restriction satisfies the objectwise boundary. Apply Definition 126.500 to identify the ordered four-face route, and then apply Definition 38.792 to read the boundary as precisely those objectwise face records with the displayed finite support.

Finally use Definition 301.65. It prevents verification status, bridge status, selected limits, quotient stream equality, metric-completeness data, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and any fifth public coordinate from leaking into the restricted read. Hence the restriction recovers exactly the displayed rows of the ambient read and no aggregate completion or verification datum. □

Theorem 301.68 (RegSeqRat four-face hub sibling route)**.** *Every accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal handoff used by current L10 consumers factors through the sibling chapters chapter 291, chapter 126, and chapter 38 before it is read as a Real-facing route. The route is the finite four-face inventory consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *scheduled-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

The handoff exports no quotient stream equality, selected host limit, host real equality, ambient completeness datum, metric-completion point, or additional Real seal coordinate.

Proof. Start with Definition 301.69. It identifies the public handoff route as the ordered passage through $\mathsf{StreamName}^{\uparrow}$ scheduled windows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Next apply Definition 301.58. The terminal read is one finite pullback with exactly the DyadicRatCore face, StreamName face, RegSeqRat face, and Real seal face, transported by the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally use Definition 301.60. Agreement of terminal reads is detected only on that four-face inventory, so the hub route is a substantive sibling route through the named DyadicRatCore, StreamName, and Real chapters rather than an isolated local theorem. The cited exclusion clauses leave no quotient stream equality, selected host limit, host equality, ambient completeness datum, metric-completion point, or extra seal coordinate to export. □

Theorem 301.69 (RegSeqRat StreamName Dyadic Real handoff exactness)**.** *For every accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, any public* $\mathsf{Real}^{\uparrow}$ *seal consumer of that carrier factors exactly through the finite route*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows selecting the visible finite windows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows read from those windows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows over the same scheduled dyadic observations;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal rows consuming that finite classifier surface.*

The factorization preserves the displayed hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows at every stage. No quotient stream, host real equality, selected limit, metric-completion object, ambient completeness datum, or extra seal coordinate can enter the handoff.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamNameDyadicRealHandoffExactness`

Proof. Project the carrier of Definition 301.1 and the classifier of Definition 301.2. Their visible rows are the scheduled $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius reads, the finite dyadic-distance rows, regularity rows, classifier rows, and the attached hsame, Cont, Pkg, and local NameCert provenance.

Apply Definition 301.4 to the schedule side and Definition 301.5 to the classifier side. Together they identify the endpoint, radius, distance, and regularity rows as rows of the same finite scheduled observation package.

Transport that package through Definition 301.6. The transport keeps the schedule, endpoint, radius, distance, regularity, classifier, hsame, Cont, Pkg, and NameCert rows componentwise, so the consumer has no alternative finite source surface available.

Finally use Definition 301.22, Definition 301.43, and Definition 301.50. The first theorem routes the accepted regular-rational source to the $\mathsf{Real}^{\uparrow}$ seal, the second makes the classifier factorization explicit, and the third locks the public consumer to the same finite schedule and replay ledger.

Since the cited surfaces expose only the displayed finite rows, there is no coordinate for quotient streams, host real equality, selected limits, completion objects, ambient completeness data, or an additional seal route. □

Theorem 301.70 (RegSeqRat StreamName Dyadic Real handoff single-source lock)**.** *Let two public* $\mathsf{Real}^{\uparrow}$ *seal consumers read the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier through displayed* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers. Then both seal reads factor through one finite source route: the schedule rows, dyadic endpoint rows, dyadic radius rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, regularity rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows attached to that carrier.*

The lock exposes no quotient stream, host real equality, selected limit, metric-completeness datum, or extra $\mathsf{Real}^{\uparrow}$ *seal coordinate. A consumer that requests any of those objects has left the displayed source route rather than reading a second seal source.*

Proof. Begin with Definition 301.69. It routes each public $\mathsf{Real}^{\uparrow}$ seal read through the same four-stage StreamName, DyadicRatCore, RegSeqRat, and Real seal path, preserving the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next apply Definition 301.103. The L10 source lattice records the finite schedule rows, dyadic endpoint and radius rows, classifier rows, regularity rows, transports, routes, provenance, and naming rows as one visible source packet. Since both consumers start from the same accepted carrier, their displayed schedules and dyadic ledgers point into that same packet.

Finally use Definition 301.22, Definition 301.43, and Definition 301.50. These seal handoff results identify the public read with the regular-rational classifier surface and rule out a second public seal source. The excluded quotient, host-equality, selected-limit, completeness, and extra-coordinate data have no row in the route just assembled, so the two seal consumers are locked to one finite source route. □

Theorem 301.71 (RegSeqRat StreamName Dyadic Real source-seal correspondence)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier read through displayed* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, the public* $\mathsf{Real}^{\uparrow}$ *seal route is equivalent, as a finite BEDC route, to the four-stage route*

1. $\mathsf{StreamName}^{\uparrow}$ *finite-window rows selecting the scheduled observations;*
2. $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows read from those windows;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows over the same finite dyadic observations;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal row that consumes that classifier surface.*

The correspondence is bidirectional at the level of displayed rows: every public seal read factors through the four-stage route, and every such four-stage route determines the same public seal read for the accepted carrier. It introduces no quotient stream, host real equality, selected limit, ambient completeness datum, or hidden seal coordinate.

Proof. For the forward direction, apply Definition 301.69. A public $\mathsf{Real}^{\uparrow}$ seal read is routed through StreamName schedules, DyadicRatCore endpoint and radius ledgers, RegSeqRat classifier rows, and the public seal, while preserving the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next lock the source by Definition 301.70. The same accepted carrier supplies one finite source packet for all public seal reads, so the route cannot branch into a second completion object or an unstated equality of streams.

For the reverse direction, read the four displayed stages through the Lean-facing anchors `RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff`, `DyadicRatCoreCarrier_regseqrat_` and `RegSeqRatStreamCarrier_real_seal_handoff`. These anchors match the paper route from finite window rows to dyadic radius rows, from dyadic ledgers to the RegSeqRat classifier, and from that classifier to the public Real seal.

The two directions have the same displayed coordinates. Since neither the paper handoff theorems nor the Lean-facing handoff anchors expose quotient streams, host real equality, selected limits, ambient completeness, or hidden seal data, the correspondence is exactly the finite source-seal route stated above. □

Theorem 301.72 (RegSeqRat StreamName Dyadic Real triple exactness)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, the public* $\mathsf{Real}^{\uparrow}$ *seal route is exactly the common finite triple surface consisting of the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, with the* $\mathsf{Real}^{\uparrow}$ *seal as the consumer of that surface. The certificate preserves precisely the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows.*

The triple surface contains no quotient stream, host real equality, selected limit, metric-completion object, ambient completeness datum, choice datum, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Apply Definition 301.69. It gives the four-stage public seal route and identifies the StreamName, DyadicRatCore, RegSeqRat, and RealUp rows read by the consumer.

Next apply Definition 301.70. The schedule, dyadic ledger, classifier, regularity, transport, continuation, package, and naming rows all come from one finite source route attached to the accepted carrier.

Use Definition 301.71 to read the public Real seal route and the displayed finite source-seal route in both directions. Finally apply Definition 301.74 to place cofinal tail reads on the same dyadic ledger whenever a tail refinement is presented.

These four inputs leave exactly the triple surface named in the statement, with the $\mathsf{Real}^{\uparrow}$ seal as its consumer. Their exclusion clauses remove quotient streams, host real equality, selected limits, metric-completion objects, ambient completeness data, choice data, and second seal coordinates. □

Theorem 301.73 (RegSeqRat StreamName Dyadic Real four-stage consumer minimality)**.** *For any accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier whose public* $\mathsf{Real}^{\uparrow}$ *seal is read through displayed* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, every valid public seal consumer passes through all four displayed surfaces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows;*
4. *the public* $\mathsf{Real}^{\uparrow}$ *seal row.*

If a consumer route omits any one of these four surfaces, then that route is outside the displayed BEDC handoff and cannot claim the public $\mathsf{Real}^{\uparrow}$ *seal of the accepted carrier.*

The minimality certificate exports no quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or second seal coordinate.

Proof. First apply Definition 301.69. It gives the forward public route and lists exactly the StreamName schedule rows, DyadicRatCore endpoint, radius, and distance ledgers, RegSeqRat classifier and regularity rows, and RealUp seal row consumed by the public seal read.

Next use Definition 301.70. The single-source lock prevents a consumer from replacing a missing displayed surface by a second source packet, a host equality row, or a hidden completion coordinate.

Now apply Definition 301.71. The source route and the public seal route are the same finite BEDC route in both directions, so omitting one displayed surface removes a coordinate needed by the seal route itself.

Finally use Definition 301.72. The schedule, dyadic ledger, and RegSeqRat classifier exhaust the source side of the seal, with the RealUp row as its consumer. Thus a route lacking any one of the four surfaces is not a public Real seal consumer for the displayed handoff; the excluded quotient stream, host equality, selected-limit, completeness, choice, and second-seal data have no coordinate from which to repair the omission. □

Theorem 301.74 (RegSeqRat common dyadic tail ledger)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier. Suppose two finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window reads of P feed public* $\mathsf{Real}^{\uparrow}$ *seal consumers, and suppose the two reads admit a common cofinal dyadic refinement. Then the two seal routes factor through one common finite ledger. This ledger consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows on the common refinement, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows attached to P.*

The common ledger is a displayed finite BHist *route. It is not a quotient stream equality, host real equality, selected limit, choice datum, ambient completeness witness, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Apply Definition 301.69 to each of the two tail-window reads. Each public seal consumer is thereby routed through the same four surfaces: the $\mathsf{StreamName}^{\uparrow}$ schedule rows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows, and the $\mathsf{Real}^{\uparrow}$ seal rows consuming that finite source surface.

Next apply Definition 301.70. Since both reads start from the same accepted carrier P, the two public seal consumers have one displayed source route, with the same componentwise hsame, Cont, Pkg, and local NameCert rows.

It remains to identify the shared cofinal refinement as finite ledger data. Use Definition 301.103 to place the schedule, endpoint, radius, classifier, regularity, transport, route, provenance, and naming rows inside one visible L10 source packet.

Finally apply Definition 301.4, Definition 301.5, and Definition 301.13. These facts read the cofinal dyadic refinement as displayed endpoint, radius, distance, and regularity rows of the same finite packet. No step creates a host completion object, selected limit, choice datum, quotient stream equality, host real equality, or second seal coordinate, so the common tail route is exactly the finite ledger stated above. □

Theorem 301.75 (RegSeqRat Real seal window triple correspondence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier read through a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule. Suppose the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows for that schedule feed the public* $\mathsf{Real}^{\uparrow}$ *seal consumer of P. Then the following four finite BEDC surfaces mutually determine one another:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows and membership rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows;*
4. *the public* $\mathsf{Real}^{\uparrow}$ *seal consumer row.*

Equivalently, any one of these displayed finite-window surfaces determines the same BHist *route through the other three, with the same componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The correspondence contains no quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or hidden seal coordinate.*

Proof. Start with Definition 301.69. It gives the finite route from the accepted carrier through the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance rows, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface, and the public $\mathsf{Real}^{\uparrow}$ seal consumer.

Apply Definition 301.71 to read the route in both directions. The public seal read determines the finite source surface, and the finite source surface determines the same public seal read, without adding completion data or external equality.

Next use Definition 301.72. It identifies the schedule, dyadic ledger, and RegSeqRat classifier/regularity surfaces as one exact triple surface whose consumer is the public $\mathsf{Real}^{\uparrow}$ seal row.

Finally apply Definition 301.74. Whenever the finite schedule is viewed through a common cofinal dyadic refinement, the same schedule rows, endpoint rows, radius rows, distance rows, classifier rows, regularity rows, transports, continuations, package provenance, and naming rows are read back as one common BHist ledger. Thus each of the four displayed surfaces recovers the same route through the other three.

All cited inputs expose only finite rows of the BEDC route. They provide no coordinate for quotient streams, host real equality, selected limits, ambient completeness, choice data, or hidden seal data, so the correspondence is exactly the finite Real seal route stated above. □

Theorem 301.76 (RegSeqRat L10 four-face necessity)**.** *A current Phase Real and Completion exit read for an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier requires all four displayed faces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule and membership rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledgers;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows;*
4. *the public* $\mathsf{Real}^{\uparrow}$ *seal row.*

Removing any one of these faces breaks the finite BEDC route from the L10 source lattice to the completion-facing seal. Conversely, the four faces together suffice for the exit read by the handoff exactness, source-seal correspondence, triple exactness, window triple correspondence, and L10 source lattice pointer results.

Proof. Apply Definition 301.103. It places the accepted $\mathsf{RegSeqRat}^{\uparrow}$ carrier on the finite source lattice whose public route must pass through scheduled windows, dyadic ledger rows, regular sequence classifier rows, and the completion-facing seal.

Use Definition 301.69. The handoff theorem displays the same route as an ordered finite passage through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. If the schedule face is removed, the window route has no finite source index. If the dyadic face is removed, the endpoint, radius, and distance ledgers cannot be read. If the RegSeqRat face is removed, the classifier and regularity rows are not available. If the Real face is removed, the public seal consumer has no finite target row.

Now apply Definition 301.71 and Definition 301.72. The first identifies the source and seal reads as the same finite public route, while the second binds the schedule, dyadic ledger, and RegSeqRat classifier faces as one exact triple before the seal is consumed.

Finally use Definition 301.75. It states that each of the four displayed faces recovers the same BHist route through the other three. Thus each face is necessary for the finite exit route, and

all four together are sufficient without adding quotient streams, host real equality, selected limits, ambient completeness, choice data, or hidden seal coordinates. □

Theorem 301.77 (StreamName RegSeqRat Real Dyadic completion-face boundary)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, a Phase Real and Completion exit read is admissible exactly when the same finite* BHist *route contains all four completion-facing rows:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule and membership rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal row.*

The route preserves the displayed hsame *rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows, and it has no completion-facing source outside the shared finite route.*

Proof. For admissibility from the four rows, apply Definition 301.69. It routes the accepted carrier through the StreamName schedule, DyadicRatCore endpoint and radius data, RegSeqRat classifier and regularity rows, and the public Real seal.

Use Definition 301.71 to identify the public seal read with the same finite source route. Then apply Definition 301.72 to bind the schedule, dyadic ledger, and RegSeqRat classifier as one exact triple before the seal row is consumed.

For necessity, apply Definition 301.75. Each of the four displayed faces recovers the same BHist route through the other three.

Finally use Definition 301.76. It states that removing any one of the schedule, dyadic, RegSeqRat, or Real seal faces breaks the Phase Real and Completion exit read. Hence the exit read is admissible exactly for the shared finite route containing all four rows. □

Theorem 301.78 (StreamName RegSeqRat Real completion-face non-escape)**.** *The completion-facing route of Definition 301.77 exports no quotient stream, host real equality, selected limit, metric completion object, ambient completeness datum, countable-choice schedule, second* $\mathsf{Real}^{\uparrow}$ *seal coordinate, standard bridge row, or aggregate verification token.*

Proof. Start from Definition 301.77. The exit read is the shared finite route containing the StreamName schedule rows, DyadicRatCore endpoint and radius rows, RegSeqRat classifier and regularity rows, and the Real seal row.

Apply Definition 301.69 and Definition 301.71. These identify the public seal read with displayed finite rows and exclude quotient streams, host real equality, selected limits, completion objects, ambient completeness, and hidden seal data.

Next apply Definition 301.72 and Definition 301.75. The triple and window correspondences keep the four faces inside one BHist route with the same support rows.

Finally use Definition 301.76. Since the exit read has exactly the required four faces, neither a standard bridge row nor an aggregate verification token is among the displayed faces, and there is no fifth coordinate from which a countable-choice schedule, metric completion object, ambient completeness witness, selected limit, host equality, quotient stream, second seal coordinate, standard bridge row, or aggregate verification token could be exported. □

Theorem 301.79 (RegSeqRat Real seal finite-window saturation)**.** *Every accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read that exits as a* $\mathsf{Real}^{\uparrow}$ *seal for Phase Real and Completion is saturated by exactly four finite faces:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule and membership rows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledgers;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows;*
4. *the public* $\mathsf{Real}^{\uparrow}$ *seal row.*

No fifth face is available to the exit read. In particular, the saturated route contains no quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or hidden seal coordinate beyond the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier real seal finite window saturation`

Proof. Begin with Definition 301.76. It shows that a Phase Real and Completion exit read requires the schedule, dyadic ledger, RegSeqRat classifier, and public seal faces, and that those four faces suffice for the L10 exit route.

Use Definition 301.75. It identifies each displayed face with the same finite BHist route through the other three, so any accepted public seal read already recovers the schedule, dyadic, classifier, transport, and seal rows without an extra coordinate.

Next apply Definition 301.74. It keeps common cofinal dyadic refinements on the same endpoint, radius, distance, classifier, regularity, transport, continuation, provenance, and local naming ledger. Thus refinements do not create an additional face.

Finally use Definition 301.71. It ties the finite source read to the public seal read in both directions. The only surfaces exported to the seal are the four listed faces, and the excluded quotient, host equality, selected-limit, completeness, choice, and hidden-coordinate data have no row in the accepted route. □

Theorem 301.80 (RegSeqRat finite-cover Real seal gluing uniqueness)**.** *Let* $\mathcal{W}$ *be a finite cover of* $\mathsf{StreamName}^{\uparrow}$ *windows for an accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read in Phase Real and Completion. Suppose that on every nonempty overlap the two local window routes agree on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledgers, the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows, and the displayed public* $\mathsf{Real}^{\uparrow}$ *seal rows, with the agreement witnessed by the same componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows in a finite* BHist *route.*

Then the finite family of local window routes glues to at most one Real-facing public seal route. Equivalently, any two finite BHist *routes assembled from the same cover and the same overlap ledgers have the same public* $\mathsf{Real}^{\uparrow}$ *seal row and the same StreamName, DyadicRatCore, and RegSeqRat support rows. The glued seal exports no quotient stream, selected limit, host equality, countable-choice schedule, ambient completeness datum, or standard bridge row.*

Proof. First apply Definition 301.79. Each local window route has exactly the four displayed faces: the StreamName schedule and membership rows, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat classifier-regularity-transport rows, and the public Real seal row.

Next use Definition 301.75. On an overlap, agreement of the endpoint and radius ledger, classifier and regularity rows, and seal row identifies the same finite BHist route through the remaining face. Thus adjacent windows in the cover cannot choose distinct Real-facing exit routes while keeping the displayed overlap data fixed.

We now pass across the cover. Since $\mathcal{W}$ is finite, repeat the overlap identification along the finite adjacency chain inside the displayed route data. The result is a single compatible StreamName,

DyadicRatCore, RegSeqRat, and RealUp four-face route, with no appeal to an infinite selection schedule or to an ambient completion object.

Finally apply Definition 301.77 and Definition 301.78. The boundary theorem identifies the completion-facing exit with the same four finite faces, while the nonescape theorem excludes quotient streams, host real equality, selected limits, ambient completeness witnesses, extra seal coordinates, standard bridge rows, and aggregate verification tokens. Hence two glued Real-facing public seals built from the cover must be the same finite BHist route. □

Theorem 301.81 (RegSeqRat StreamName Real three-surface route determinacy)**.** *Let R be an accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read. Suppose two public presentations of R display the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and membership rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledger, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows. Then the two presentations determine the same BEDC route to the* $\mathsf{Real}^{\uparrow}$ *seal exactly when they reach the same displayed public* $\mathsf{Real}^{\uparrow}$ *seal row.*

Equivalently, the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces are not four independent choices. Once the first three faces are fixed, the displayed public seal row is the whole remaining route coordinate, and two seal reads agree precisely by sharing that row in the same finite BHist *route. The statement uses no quotient stream, host real equality, selected limit, ambient completeness object, choice schedule, or hidden seal coordinate.*

Proof. First apply Definition 301.69. It places each accepted presentation on the finite handoff route through the displayed StreamName schedule rows, dyadic endpoint-radius-distance ledger, RegSeqRat classifier and regularity rows, and public Real seal row.

Next use Definition 301.71. This identifies the source read and the seal read as the same public finite route. Therefore any agreement of the displayed seal row is agreement of the route endpoint reached by the fixed source rows.

Now apply Definition 301.72 together with Definition 301.75. The triple exactness theorem binds the shared StreamName, DyadicRatCore, and RegSeqRat faces, while the window-triple correspondence recovers the same route through the Real seal face. Thus the reverse direction also holds: if the two presentations determine the same BEDC route, the displayed public seal row is the same row of that route.

Finally invoke Definition 301.79. The saturated route has exactly the four public faces listed in the statement. There is no fifth coordinate from which quotient streams, host real equality, selected limits, ambient completeness, choice schedules, or hidden seal data could alter the route after the first three faces and the displayed seal row have been fixed. □

Theorem 301.82 (RegSeqRat StreamName Dyadic Real budgeted request determinacy)**.** *Let R be an accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read, and let $B = (E, W, D, Q, S, H, C, P, N)$ be an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose B presents the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, and the same public* $\mathsf{Real}^{\uparrow}$ *seal row as the accepted read R. Then the budgeted request determines the same finite BEDC route as R exactly when it agrees on the displayed rows $E, W, D, Q, S, H, C, P, N$.*

Equivalently, the packet rows are the whole request-side coordinate of the four-face route. Agreement on those rows fixes the StreamName window surface, the DyadicRatCore metric ledger, the RegSeqRat handoff, the RealUp seal, and the transport, continuation, provenance, and local naming rows. The budgeted request exports no quotient stream, host real equality, selected limit, ambient completeness witness, choice schedule, or hidden seal coordinate.

Proof. First place the accepted read on the finite handoff route. Apply Definition 301.69 to identify the StreamName window rows, the DyadicRatCore endpoint-radius-distance ledger, the RegSeqRat handoff, and the public Real seal row as one finite BEDC route.

Next record that no smaller public surface can carry the same exit. By Definition 301.76, the L10 handoff needs the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces together, and Definition 301.81 says that once the first three faces are fixed, agreement of the displayed Real seal row is exactly agreement of the route endpoint.

Now attach the explicit observation packet. Apply Definition 611.32 to read the packet route through

$$E \longmapsto W \longmapsto D \longmapsto Q \longmapsto S$$

with the rows H, C, P, N carried as transport, continuation, provenance, and local naming data. Then use Definition 611.33 to lock every RealUp seal read admitted by S behind the same finite StreamName windows, DyadicRatCore rows, and RegSeqRat handoff.

The forward direction follows because agreement on $E, W, D, Q, S, H, C, P, N$ therefore supplies exactly the packet rows needed to reproduce the finite four-face route already determined by R. The reverse direction follows because the same finite BEDC route has no extra packet coordinate: if the budgeted request and R determine the same route, the displayed request, window, dyadic, handoff, seal, transport, continuation, provenance, and naming rows must be the rows of that route.

The non-escape clauses in the cited handoff and budget theorems exclude quotient stream equality, host real equality, selected limits, ambient completeness, choice schedules, and hidden seal coordinates. Hence no additional analytic or ambient datum can distinguish two budgeted requests after the displayed rows agree. □

Theorem 301.83 (RegSeqRat StreamName Real tail-window exhaustion)**.** *Every accepted tail-window consumer of the* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *handoff factors only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *scheduled windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, finite* Cont *continuation,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. The consumer surface has no coordinate for quotient streams, host real equality, selected limits, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 301.74. It places the shared tail read on the common dyadic ledger, so the endpoint, radius, and tail-window entries seen by a consumer are already the displayed finite rows of the handoff.

Next use Definition 301.79. The RealUp seal contributes only the finite windows saturated by the StreamName schedule, the dyadic ledger, and the RegSeqRat handoff rows; it does not add ambient completeness or a host-level limit object.

Then invoke Definition 301.81. Once the StreamName, DyadicRatCore, and RegSeqRat surfaces are fixed, the displayed RealUp seal row is the only remaining public route coordinate.

Finally attach the request-side rows by Definition 301.82. Transport, continuation, provenance, and local naming are exactly the displayed $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ support rows, so the tail-window consumer has no route through quotient streams, selected limits, metric completeness, or ambient RealUp completeness. □

Theorem 301.84 (RegSeqRat StreamName Real audit-clean route triad)**.** *Any RegSeqRat-side consumer of the L10 handoff reaches machine audit support through exactly the StreamName singleton finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSe-*

qRat Real seal handoff. The route has no quotient stream, selected limit, ambient Real completeness, metric-completeness row, countable-choice row, or bridge row.

Proof. First apply Definition 301.69. It places the RegSeqRat-side consumer on the finite handoff route through the displayed StreamName window, DyadicRatCore ledger, RegSeqRat readback, and Real seal.

Next use Definition 301.72. The triple exactness theorem binds the StreamName, DyadicRatCore, and RegSeqRat faces as one finite route before the Real seal is read.

Now invoke Definition 301.81. Once the first three surfaces are fixed, the displayed Real seal row is the whole remaining route coordinate, so a consumer cannot introduce another machine-facing support site.

Finally apply Definition 301.82. The budgeted request theorem confirms that even an explicit observation request reuses the same StreamName, DyadicRatCore, RegSeqRat, and RealUp rows. Therefore the audit route is exactly the three touchpoints listed in the statement, with no quotient stream, selected limit, ambient completeness, metric-completeness row, countable choice, or bridge row. □

Theorem 301.85 (RegSeqRat audit-clean route sufficiency)**.** *Let H be an accepted RegSeqRat-side L10 handoff whose visible rows are exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the* $\mathsf{Real}^{\uparrow}$ *seal row. Then the three Lean-facing anchors* RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff*,* DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness*, and* RegSeqRatStreamCarrier_real_seal_handoff *are sufficient for the Phase Real and Completion audit-clean route surface determined by H.*

No aggregate formalstatus row, bridge row, quotient stream, selected limit, countable-choice row, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate is part of that route surface.*

Proof. Apply Definition 301.84. It names exactly the StreamName finite-window handoff, the DyadicRatCore RegSeqRat bridge handoff exactness, and the RegSeqRat Real seal handoff as the machine-facing support sites for any RegSeqRat-side L10 consumer.

Next use Definition 301.69. This identifies the visible rows of H as one finite route through the StreamName schedule, DyadicRatCore endpoint-radius-distance ledger, RegSeqRat classifier and regularity rows, and public RealUp seal row. Hence the four concrete-instance faces named in the statement are the whole carrier surface seen by the route.

Then invoke Definition 301.79. The RealUp seal is saturated by the same finite windows and ledger rows, so it adds no ambient completeness coordinate, selected limit, quotient stream, or hidden analytic witness beyond the displayed handoff data.

Finally apply Definition 301.82. Even when the handoff is read through an explicit budgeted request, agreement on the displayed StreamName, DyadicRatCore, RegSeqRat, and RealUp rows determines the same route. Therefore the three Lean-facing anchors listed in the theorem are sufficient for the current audit-clean surface, and no aggregate formalstatus row, bridge row, countable-choice row, metric-completeness row, or ambient Real completeness coordinate is imported. □

Theorem 301.86 (RegSeqRat StreamName Dyadic Real Lean anchor audit surface)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier whose public seal route is saturated by the four faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, the Lean-facing anchors* `RatStreamNameFiniteWindowClassifier_singl` `DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness`*, and* `RegSeqRatStreamCarrier_real_seal_h` *expose exactly the same finite audit surface. Concretely, they expose the StreamName finite-window membership surface, the DyadicRatCore endpoint, radius, and distance handoff surface, and the*

RegSeqRat-to-Real public seal handoff surface. No additional audit coordinate is supplied by a quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or hidden seal coordinate.

Proof. Apply Definition 301.71. It identifies the accepted carrier's finite source route with the public seal route and already names the three Lean-facing anchors as the route from finite-window rows to dyadic rows, from dyadic rows to the RegSeqRat classifier, and from that classifier to the public Real seal.

Next use Definition 301.76. It shows that a Phase Real and Completion exit read requires exactly the StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier, and Real seal faces for the L10 route. Thus the first two anchors cover the first-three-face passage through StreamName, DyadicRatCore, and RegSeqRat, while the third anchor covers the terminal seal passage into $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 301.79. The saturated route has no fifth face, and the excluded quotient stream, host real equality, selected limit, ambient completeness, choice datum, and hidden seal coordinate have no displayed row. Therefore the three named anchors are exactly the finite audit surface that remains to be formalized for the four-face route. □

Theorem 301.87 (RegSeqRat L10 exit route order lock)**.** *For any accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read whose terminal output is a* $\mathsf{Real}^{\uparrow}$ *seal, the L10 exit route is ordered as follows: first the* $\mathsf{StreamName}^{\uparrow}$ *finite-window membership face, second the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger face, third the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport face, and last the public* $\mathsf{Real}^{\uparrow}$ *seal face.*

No later face may be read before its predecessor. In particular the public seal row is not available before the RegSeqRat face has been read, the RegSeqRat face is not available before the DyadicRatCore ledger has been read, and the dyadic ledger is not available before the StreamName membership face has fixed the finite window. The route excludes quotient streams, host real equality, selected limits, ambient completeness, choice rows, metric-completeness data, aggregate formalstatus rows, aggregate bridge rows, and a second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Begin with Definition 301.86. It identifies the three Lean-facing anchors as the finite-window handoff, the dyadic endpoint-radius-distance handoff, and the RegSeqRat-to-Real seal handoff, with no extra audit coordinate.

Next apply Definition 301.76. Necessity says that the L10 public exit needs exactly the StreamName, DyadicRatCore, RegSeqRat, and Real faces. Thus removing or bypassing any predecessor would leave a later face without the finite source row on which it is certified.

Use Definition 301.72 to bind the first three faces as one ordered finite route. The triple exactness step keeps the StreamName membership rows upstream of the DyadicRatCore ledger and keeps the dyadic ledger upstream of the RegSeqRat classifier, regularity, and transport rows.

Finally combine Definition 301.71 with Definition 301.79. The correspondence places the terminal public seal on the same finite source route, while saturation says that the route has no fifth face and no hidden second seal coordinate. Hence the four public faces are read in the displayed order, and the excluded quotient, host-equality, limit, completeness, choice, formalstatus, bridge, and second-seal data remain outside the route. □

Theorem 301.88 (RegSeqRat Real seal finite-cover gluing)**.** *Let* $\{\mathcal{S}_i\}_{i\in I}$ *be a finite family of accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads whose members exit as* $\mathsf{Real}^{\uparrow}$ *seals. Suppose each* $\mathcal{S}_i$ *is saturated by exactly the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Suppose also that every displayed pairwise overlap agrees on the shared BHist*, componentwise* hsame*,* Cont*,* Pkg*,* NameCert*, dyadic-ledger, classifier, regularity, and seal rows. Then the finite family glues to one finite four-face L10 packet over the finite union of the requests, with the same four faces and only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The glued packet contains no quotient stream equality, no host real equality, no selected limit, no ambient or metric-completeness datum, no choice data, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. For each family member, apply Definition 301.79. This places the member on one finite L10 exit route whose only faces are the StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier and regularity surface, and Real seal row.

Glue the StreamName windows first. The overlap hypotheses identify the displayed BHist, hsame, Cont, Pkg, NameCert, dyadic, RegSeqRat, and Real rows seen from adjacent members, so Definition 126.562 gives one finite StreamName-side request packet over the finite union.

Next keep the Real consumer on that same finite section. Use Definition 38.698 to repack each seal request through the displayed finite BHist packet, and use Definition 38.701 to force every completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer to factor through that section.

It remains to identify the dyadic and RegSeqRat faces on overlaps. Apply Definition 301.74 to keep common cofinal dyadic refinements on the same endpoint, radius, distance, classifier, regularity, transport, continuation, provenance, and local naming ledger. Then apply Definition 301.75 to recover each of the four faces from the other three inside the same finite route.

Finish by finite induction over the request family. The empty cover gives the empty displayed packet. The adjoining move is exactly Definition 301.89, applied with the packet already built as the glued finite family and with the next accepted saturated member as the outside request. At each step the cited saturation, StreamName gluing, Real fusion, consumer section, dyadic ledger, and window correspondence theorems export only the four displayed faces and their support rows, so quotient stream equality, host real equality, selected limits, ambient or metric completeness, choice data, and a second seal coordinate remain absent. □

Theorem 301.89 (RegSeqRat Real seal finite-cover induction step)**.** *Let* $\mathcal{F}$ *be a finite family of accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads that has already been glued to a finite four-face L10 packet. Let* $\mathcal{T}$ *be one further accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read whose public exit is a* $\mathsf{Real}^{\uparrow}$ *seal. Suppose* $\mathcal{T}$ *is saturated by exactly the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

and suppose that its displayed overlap with the glued packet agrees on BHist*, componentwise* hsame*,* Cont*,* Pkg*,* NameCert*, dyadic-ledger, classifier, regularity, and seal rows. Then adjoining* $\mathcal{T}$ *produces one glued finite four-face L10 packet over* $\mathcal{F} \cup \{\mathcal{T}\}$*.*

The enlarged packet has exactly the same four public faces $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and exactly the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It contains no quotient stream, no host real equality, no selected limit, no ambient or metric-completeness datum, no choice data, and no second seal coordinate.*

Proof. First saturate the outside member. Apply Definition 301.79 to $\mathcal{T}$. This gives a finite L10 route whose only faces are the StreamName request window, DyadicRatCore ledger, RegSeqRat classifier and regularity surface, and the terminal Real seal row.

Next enlarge the request cover. The glued packet over $\mathcal{F}$ already has a StreamName finite request packet, and the overlap hypotheses identify the displayed BHist, hsame, Cont, Pkg, NameCert, dyadic-ledger, classifier, regularity, and seal rows with those of $\mathcal{T}$. Hence Definition 126.562 attaches the outside request to the same finite cover.

Now keep the Real consumer on that finite section. Use Definition 38.698 to express the outside Real seal through the enlarged finite BHist packet, and use Definition 38.701 to keep the completion-facing consumer on that section rather than on an ambient completion or choice row.

It remains to preserve the dyadic and RegSeqRat faces on the shared boundary. Apply Definition 301.74 to carry common cofinal dyadic refinements through the same endpoint, radius, distance, classifier, regularity, transport, continuation, provenance, and local naming ledger. Then apply Definition 301.75 to recover the four-face route from any three of its StreamName, DyadicRatCore, RegSeqRat, and Real windows.

The repacked object is therefore the glued packet over $\mathcal{F} \cup \{\mathcal{T}\}$. Every cited route exports only the four displayed faces and the named support rows, so the excluded quotient stream, host real equality, selected limit, ambient or metric-completeness datum, choice data, and second seal coordinate have no coordinate in the enlarged packet. □

Theorem 301.90 (RegSeqRat Real seal finite-cover associativity)**.** *Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be finite families of accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads whose members exit as* $\mathsf{Real}^{\uparrow}$ *seals. Assume that every member is saturated by the same four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

and that every pairwise overlap among the three families satisfies the four-face agreement used in Definition 301.88. Then gluing $\mathcal{A} \cup \mathcal{B}$ first and then gluing with $\mathcal{C}$ displays the same finite four-face L10 packet as gluing $\mathcal{B} \cup \mathcal{C}$ first and then gluing with $\mathcal{A}$.

The common packet has exactly the StreamName schedule and membership face, the DyadicRatCore endpoint, radius, and distance face, the RegSeqRat classifier, regularity, and transport face, and the RealUp seal face, with the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It contains no quotient stream equality, no host real equality, no selected limit, no ambient or metric-completeness datum, no choice data, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Apply Definition 301.88 to $\mathcal{A} \cup \mathcal{B}$ and then to $(\mathcal{A} \cup \mathcal{B}) \cup \mathcal{C}$. Apply the same theorem to $\mathcal{B} \cup \mathcal{C}$ and then to $\mathcal{A} \cup (\mathcal{B} \cup \mathcal{C})$. The overlap hypotheses are exactly the hypotheses of the binary gluing theorem for each of these steps.

For every member used in either parenthesization, apply Definition 301.79. This identifies the member with the same StreamName schedule and membership row, DyadicRatCore endpoint, radius, and distance row, RegSeqRat classifier, regularity, and transport row, and Real seal row.

The only remaining issue is overlap agreement after the intermediate gluing step. Use Definition 301.75 to read any three of the four faces as the same fourth face inside the finite route, and use Definition 301.74 to keep the dyadic and RegSeqRat ledger fixed on common cofinal tails. Thus both intermediate packets present the same finite BHist, hsame, Cont, Pkg, NameCert, endpoint, radius, distance, classifier, regularity, transport, and seal rows on every overlap.

Conclude by finite induction over the members of $\mathcal{C}$ after the first parenthesization, and symmetrically over the members of $\mathcal{A}$ after the second parenthesization. The base case is the already glued binary packet. The adjoining step attaches one saturated member along the overlap rows just identified, so it cannot introduce a new face or a different support row. Hence both parenthesizations expose the same finite BEDC rows and no extra route. □

Theorem 301.91 (RegSeqRat Real seal common-refinement cover completeness)**.** *Let $\{\mathcal{S}_i\}_{i \in I}$ be a finite family of accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads whose members exit as* $\mathsf{Real}^{\uparrow}$ *seals. Suppose each $\mathcal{S}_i$ is saturated by exactly the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Suppose further that every finite subfamily $J \subseteq I$ has a certified $\mathsf{StreamName}^{\uparrow}$ *common-refinement packet $\mathcal{R}_J$ whose restrictions to the members of J agree on the dyadic endpoint, radius, distance, and tail ledger rows, on the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, on the* $\mathsf{Real}^{\uparrow}$ *public seal row, and on the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

Then the glued family is covered by one finite four-face pullback surface over the union of the accepted reads. This surface restricts to each $\mathcal{R}_J$ on every finite subfamily and displays only the $\mathsf{StreamName}^{\uparrow}$ *schedule and membership face, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport face, the* $\mathsf{Real}^{\uparrow}$ *seal face, and the listed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It exports no quotient stream, no host equality, no selected limit, no ambient or metric-completeness datum, no choice data, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Begin with the common-refinement packets. By Definition 126.240, each $\mathcal{R}_J$ exposes only the public StreamName source tuple and forces all downstream $\mathsf{RegSeqRat}^{\uparrow}$ and finite-window $\mathsf{Real}^{\uparrow}$ reads to factor through that tuple.

Next apply Definition 126.405. For every finite subfamily, the StreamName window, RegSeqRat readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row form one finite pullback with exactly the displayed BHist, hsame, Cont, Pkg, and local NameCert support.

Now pass from subfamily pullbacks to the whole finite family. Apply Definition 301.88 to glue the accepted reads along their shared dyadic ledger, classifier, regularity, seal, and support rows. The hypotheses give the overlap agreement required by that gluing theorem because every overlap is the restriction of some $\mathcal{R}_J$.

Finally use Definition 301.90. Reordering or reparenthesizing the finite subfamilies does not change the resulting four-face packet, so the cover obtained from the common-refinement packets is complete for the finite family. The cited results list all exported rows, and none supplies quotient streams, host equality, selected limits, ambient or metric completeness, choice data, or a second Real seal coordinate. □

Theorem 301.92 (RegSeqRat Real seal finite-cover refinement invariance)**.** *Let $\{\mathcal{S}_i\}_{i\in I}$ be a finite family of accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *reads saturated by the four displayed faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Let a second finite family refine it by certified* $\mathsf{StreamName}^{\uparrow}$ *common-refinement packets whose restrictions preserve the dyadic endpoint, radius, distance, and tail ledger rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *public seal row, and the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

Then the original finite cover and its finite certified refinement determine the same four-face L10 exit packet: the $\mathsf{StreamName}^{\uparrow}$ *schedule and membership face, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport face, and the* $\mathsf{Real}^{\uparrow}$ *seal face with the listed support rows. The packet contains no quotient stream, no host equality, no selected limit, no choice datum, no ambient or metric-completeness datum, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Apply Definition 301.91 to the original finite family. It gives one finite four-face pullback surface and records agreement of the endpoint, radius, distance, classifier, regularity, seal, and support rows.

Apply Definition 301.88 to the refined finite family. The restriction hypotheses give the overlap rows required by gluing, because each refined member lands in the same StreamName common-refinement packet described by Definition 126.405.

It remains only to compare the packet obtained by first gluing the refined family with the packet obtained by gluing the original family. Use Definition 301.90: finite regrouping and reparenthesizing of saturated members preserves the StreamName, dyadic ledger, RegSeqRat, Real seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Thus the refined and unrefined finite covers expose the same L10 exit packet. The cited theorems also list the absent data, so the comparison cannot add a quotient stream, host equality, selected limit, choice, ambient or metric completeness, or a second Real seal coordinate. □

Theorem 301.93 (RegSeqRat StreamName Dyadic Real audit-clean exit readiness)**.** *For any accepted public L10 exit read of a* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, audit-clean readiness for the Phase Real and Completion quartet is exactly the finite four-face route through the* $\mathsf{StreamName}^{\uparrow}$ *finite schedules, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic endpoint, radius, and distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and transport rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal row.*

The readiness surface contains only the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows for those four faces. It contains no Real completeness theorem, no quotient representative, no host sequence equality, no trichotomy principle, no open-cover compactness object, and no second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Start with Definition 301.69. It routes the accepted public seal read through the StreamName schedule face, the DyadicRatCore ledger face, the RegSeqRat classifier and regularity face, and the public Real seal face while preserving the displayed hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 301.70 and Definition 301.71. The single-source lock prevents a second completion source, and the correspondence identifies the finite source read with the public seal read in both directions. Thus the exit criterion can read only the same finite rows from either side.

Use Definition 301.76 together with Definition 301.79. These results say that all four faces are required and that no fifth face is available to an accepted Phase Real and Completion exit read.

Finally apply Definition 301.88 and Definition 301.90. Finite families of accepted reads glue and reassociate without changing the StreamName, DyadicRatCore, RegSeqRat, or Real seal rows, so audit-clean Lean work should target this finite route. Stronger off-table claims about Real completeness, quotient representatives, host sequence equality, trichotomy, open-cover compactness, or a second Real seal coordinate are not part of the exit-readiness surface. □

Theorem 301.94 (RegSeqRat Real seal terminal-section exhaustion)**.** *Every terminal public* $\mathsf{Real}^{\uparrow}$ *seal section of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier is exhausted by the same finite L10 route through the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The exhausted section consists exactly of the displayed BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows, together with the StreamName finite-window schedules, the DyadicRatCore endpoint, radius, and distance ledgers, the RegSeqRat classifier, regularity, and transport rows, and the public Real seal row.*

No terminal section produced by this route contains a Real completeness theorem, quotient representative, host equality, trichotomy principle, open-cover compactness datum, choice datum, or second seal coordinate.

Proof. Begin with Definition 301.69. It presents the accepted carrier's public seal read as a finite passage from StreamName finite-window schedules through DyadicRatCore ledgers and RegSeqRat classifier data to the public Real seal row.

Next use Definition 301.71. The source side and the terminal seal side expose the same displayed section, so a terminal $\mathsf{Real}^{\uparrow}$ seal section cannot carry rows absent from the finite source route.

Apply Definition 301.76 and Definition 301.79. Necessity gives the four required faces, and saturation says the accepted exit read has no fifth face beyond $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

For finite terminal sections, apply Definition 301.88 and Definition 301.90. Gluing turns overlapping accepted seal reads into one finite packet over the displayed support rows, and associativity shows that the packet is independent of the order in which those finite pieces are attached.

Finally apply Definition 301.93. Its negative boundary is exactly the asserted terminal boundary: no Real completeness theorem, quotient representative, host equality, trichotomy principle, open-cover compactness datum, choice datum, or second seal coordinate is part of the terminal public section. Hence every terminal section is exhausted by the finite four-face route and by no additional BEDC row. □

Theorem 301.95 (RegSeqRat terminal descent uniqueness)**.** *Let S and T be terminal public* $\mathsf{Real}^{\uparrow}$ *seal sections of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carriers. Suppose S and T display the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

the same displayed BHist *rows, and the same componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows. Then S and T are the same terminal finite L10 section.*

Equivalently, descent along the RegSeqRat terminal route is unique at the public $\mathsf{Real}^{\uparrow}$ *seal: the displayed StreamName schedules, DyadicRatCore ledgers, RegSeqRat classifier and regularity rows, and public Real seal row determine one terminal section and admit no quotient stream, host sequence equality, selected limit, ambient completeness witness, choice datum, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 301.94 to both S and T. Each terminal public $\mathsf{Real}^{\uparrow}$ seal section is exhausted by the same finite L10 route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no extra BEDC row beyond the displayed support.

Next apply Definition 301.90. The finite pieces of the two sections may be regrouped without changing any displayed face or support row, so the comparison is independent of the order in which StreamName, DyadicRatCore, RegSeqRat, and Real seal pieces are attached.

Use Definition 301.91. The two exhausted finite covers have a common accepted refinement, and the hypothesis identifies every displayed BHist, hsame, Cont, Pkg, and local NameCert row on that refinement. Thus both terminal reads descend to the same finite packet.

Finally apply Definition 301.93. The audit-clean boundary refuses quotient streams, host sequence equality, selected limits, ambient completeness, choice data, and any second $\mathsf{Real}^{\uparrow}$ seal coordinate. Since no off-route datum remains available to separate S from T, the common refined packet is the unique terminal finite L10 section determined by the four displayed faces. □

Theorem 301.96 (RegSeqRat Real seal terminal-section observation lock)**.** *For every terminal public* $\mathsf{Real}^{\uparrow}$ *seal section of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, the exhausted four-face section determines exactly the same finite* $\mathsf{StreamName}^{\uparrow}$ *observation interface named by Definition 126.316.*

Every readback of that terminal section preserves the $\mathsf{StreamName}^{\uparrow}$ *finite-window carrier and classifier rows, the* ProbeBundle *and* InBundle *membership rows, UnaryHistory observations, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal row. It cannot add a quotient stream, host equality, selected limit, ambient completeness witness, trichotomy principle, open-cover compactness datum, choice datum, or second seal coordinate.*

Proof. First apply Definition 301.94. It identifies a terminal public $\mathsf{Real}^{\uparrow}$ seal section with the finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and it excludes every row outside that route.

Next combine this exhaustion with Definition 301.79. Saturation locks the four displayed faces as the whole accepted section, so a readback cannot replace the StreamName face or attach another completion-facing coordinate.

Use Definition 126.316 to identify the first face. That theorem supplies exactly the finite-window carrier, classifier, bundle-membership, and UnaryHistory observation interface consumed by the Real seal boundary.

Finally apply Definition 301.93 and Definition 38.684. They keep the Real-facing read on the same L10 lattice route through the DyadicRatCore ledger, the RegSeqRat classifier and regularity rows, and the public Real seal row. Thus terminal readback preserves the finite StreamName observation interface and introduces none of the off-route data listed in the statement. □

Theorem 301.97 (RegSeqRat StreamName open-phase Real exit correspondence)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, a terminal public* $\mathsf{Real}^{\uparrow}$ *seal section is an open-phase Phase Real and Completion exit boundary exactly when it exposes the same four finite faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

On this correspondence the StreamName face is the finite observation boundary locked by Definition 301.96, and the remaining faces are the DyadicRatCore ledger, the RegSeqRat classifier and regularity rows, and the public Real seal row named by Definition 301.94. The correspondence contains no quotient stream, no host sequence equality, no selected limit, no ambient $\mathsf{Real}^{\uparrow}$ *completeness row, no metric-completeness datum, no countable-choice row, and no second seal coordinate.*

Proof. For the forward direction, start with Definition 301.94. It exhausts a terminal public $\mathsf{Real}^{\uparrow}$ seal section by the four finite faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed history, sameness, continuity, package, and local certificate rows. Then apply Definition 301.96 to fix the StreamName face as the finite observation interface, and use Definition 126.451 to read that same packet as sufficient for the current Phase Real and Completion boundary.

For the reverse direction, use Definition 126.450. Any open-phase Phase Real and Completion exit boundary with the four displayed faces reads back through exactly those support rows. Combine this with Definition 301.79 to identify the readback with a saturated terminal public Real seal section, and then apply Definition 301.93 to preserve the negative boundary. Hence the two descriptions name the same finite section and exclude quotient streams, host sequence equality, selected limits, ambient completeness, metric-completeness data, countable choice, and second seal coordinates. □

Theorem 301.98 (StreamName RegSeqRat Real observation interface)**.** *For any accepted public L10 exit read of a* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *finite-window seal boundary of Definition 126.316 is exactly the input face of the four-face route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *recorded by Definition 301.79 and Definition 301.93. Thus the public observation interface consumed by the RegSeqRat-to-Real exit is the finite StreamName window package: the* RatStreamNameCarrier *and* RatStreamNameClassifier *rows, finite-window classifier rows, endpoint readbacks in* $\mathsf{Rat}^{\uparrow}$*, empty-window boundary, and constant/reindexing stability rows.*

This identification introduces no quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Begin with the StreamName boundary theorem, Definition 126.316. It names exactly the finite observation rows exposed by the $\mathsf{StreamName}^{\uparrow}$ public interface and states that those rows are available to later $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}$ seal sites without carrying regularity or completion data.

Next apply the RegSeqRat handoff exactness theorem, Definition 301.69. The handoff uses the same finite StreamName schedules as its first face before it reads the DyadicRatCore ledger, the RegSeqRat classifier and regularity rows, and the public Real seal row.

The four-face necessity and saturation results, Definition 301.76 and Definition 301.79, then force that first face to be precisely the public finite-window observation interface. A different StreamName input would either omit a required face or add a fifth face, both excluded by the saturated finite route.

Finally use the Real sibling lattice summary, Definition 38.684, together with the audit-clean exit readiness theorem Definition 301.93. They place the Real-facing exit on the same horizontal lattice edge and preserve the negative boundary: no quotient stream, host real equality, selected limit, ambient completeness witness, choice datum, or second $\mathsf{Real}^{\uparrow}$ seal coordinate is introduced. □

Theorem 301.99 (RegSeqRat Dyadic StreamName Real public seal non-escape)**.** *Every accepted public L10 exit read whose four visible faces are the* $\mathsf{StreamName}^{\uparrow}$ *finite observation interface, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the* $\mathsf{Real}^{\uparrow}$ *public seal is exhausted by those four faces together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Consequently the route exports no quotient stream, host real equality, selected limit, ambient completeness witness, trichotomy branch, open-cover compactness datum, choice datum, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate. Any consumer requesting one of those data is no longer reading the public seal through the displayed L10 exit route.*

Proof. First identify the input face by Definition 301.98. The theorem fixes the StreamName side as the finite-window observation package used by the RegSeqRat-to-Real exit, rather than a quotient stream or a host-level equality of sequences.

Next apply Definition 301.79. The saturation result says that the terminal public Real seal read has exactly the StreamName observation face, the DyadicRatCore ledger face, the RegSeqRat classifier and regularity face, and the Real seal face, carrying only the displayed history, sameness, continuity, package, and local certificate rows.

Use Definition 301.93 to preserve the negative boundary through the exit audit. It rules out the off-route completion and choice data at the same public read, so no selected limit, ambient completeness witness, trichotomy branch, open-cover compactness datum, or choice datum is supplied by the exit.

Finally locate the read on the sibling lattice route by Definition 38.684. Since the lattice edge has the same four faces and the same displayed provenance rows, finite route exhaustion leaves no coordinate for a second public seal or for any of the off-route data listed above. □

Theorem 301.100 (RegSeqRat four-face finite-observation completeness)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *Phase Real and Completion exit read, admission to the public L10 exit surface is equivalent to finite observation data exposing exactly the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

On the forward reading, the accepted exit read presents the finite $\mathsf{StreamName}^{\uparrow}$ *observation interface, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal. On the reverse reading, any finite observation packet with exactly those four faces and displayed rows is admitted as the same RegSeqRat-to-Real public exit read.*

The complete four-face surface exports no quotient stream, selected limit, host equality, countable-choice row, ambient completeness row, metric-completeness row, or standard bridge row.

Proof. For the forward direction, apply Definition 301.97. It identifies an accepted terminal public Real seal section with the open Phase Real and Completion boundary whose visible support is precisely the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed history, sameness, continuity, package, and certificate rows.

Next use Definition 301.98. This pins the first face to the finite $\mathsf{StreamName}^{\uparrow}$ window data consumed by the RegSeqRat-to-Real route, so the accepted read is described by finite observation data rather than by a quotient stream or a host-level equality.

For the reverse direction, start again from the correspondence theorem and read the same finite packet back as a Phase Real and Completion exit boundary. Then apply Definition 301.93 to place the packet on the audit-clean L10 route through the DyadicRatCore ledger, the RegSeqRat classifier and regularity rows, and the public Real seal row.

Finally apply Definition 301.99. Since the public seal is exhausted by the four displayed faces and provenance rows, no coordinate remains for a quotient stream, selected limit, host equality, countable-choice row, ambient completeness row, metric-completeness row, or standard bridge row. Thus the admitted exit reads are exactly the finite four-face observation packets described in the statement. □

Theorem 301.101 (RegSeqRat four-face status locality)**.** *Let* $\mathcal{E}$ *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *Phase Real and Completion exit read through the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Let F *be one of these four faces, and form* $\mathcal{E}_F^+$ *by replacing only one face-local* formalstatus*,* leantarget*, or* bridgestatus *row attached to* F *by a stronger row attached to the same face.*

Assume that the other three face records and the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows are unchanged. Then* $\mathcal{E}_F^+$ *presents the same finite four-face observation packet as* $\mathcal{E}$*: the StreamName finite observation face, the DyadicRatCore ledger face, the RegSeqRat classifier and regularity face, and the public Real seal face.*

This preservation introduces no aggregate route-level status row, quotient stream, selected limit, countable-choice row, metric-completeness datum, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, or second seal coordinate.*

Proof. First apply Definition 301.100. It identifies acceptance of $\mathcal{E}$ with a finite observation packet whose visible content is exactly the four displayed faces and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 301.93. The audit-clean route reads the same StreamName, DyadicRatCore, RegSeqRat, and Real faces and exports no off-route completion datum. Since the replacement is attached to the same face F and leaves the other face records and support rows fixed, the audit-clean route still reads the same four-face packet.

Apply Definition 126.426. The StreamName-side terminal theorem supplies the local rule for strengthening a formalstatus, leantarget, or bridgestatus row: the strengthened row remains a face-local row and does not become a route-level aggregate credential.

Finally apply Definition 38.815. The Real terminal locality theorem keeps every status, Lean-target, and bridge coordinate attached to one of the four local faces rather than to the route as a whole. Hence $\mathcal{E}_F^+$ has the same finite observation packet as $\mathcal{E}$, and none of the excluded quotient, limit, choice, completeness, aggregate-status, or second-seal data is introduced. □

Theorem 301.102 (RegSeqRat Real seal support-removal obstruction)**.** *Let P be an accepted public* $\mathsf{RegSeqRat}^{\uparrow}$ *read whose public exit is a* $\mathsf{Real}^{\uparrow}$ *seal through the L10 route displayed by Definition 301.69. Remove from that route any one of the four displayed faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

or remove any displayed support row among the BHist *route rows, componentwise* hsame *rows,* Cont *rows,* Pkg *provenance rows, or local* NameCert *rows. Then the remaining data do not form a valid public* $\mathsf{Real}^{\uparrow}$ *seal route for P.*

No quotient stream, host real equality, selected limit, ambient or metric-completeness datum, choice datum, aggregate formalstatus *row, aggregate bridge row, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate compensates for the missing face or support row.*

Proof. First remove one displayed face. By Definition 301.73, every valid public seal consumer for the accepted read passes through the StreamName schedule face, the DyadicRatCore endpoint and radius face, the RegSeqRat classifier and regularity face, and the Real seal face. Omitting any one of these four surfaces therefore places the attempted consumer outside the displayed public route.

Next remove a displayed support row. Apply Definition 301.69 to identify the route as one finite BEDC handoff, with the visible BHist rows, componentwise hsame rows, Cont rows, Pkg provenance rows, and local NameCert rows carrying the same accepted read from source schedules to the terminal seal. Without one of those support rows, the four faces no longer sit in a single certified handoff route for P.

It remains to rule out replacement data. The finite-cover results Definition 301.88, Definition 301.89, and Definition 301.90 keep all glued and regrouped public seal packets inside the same four displayed faces and the same BHist, hsame, Cont, Pkg, and NameCert support rows. Their exclusion clauses leave no coordinate for quotient streams, host real equality, selected limits, ambient or metric completeness, choice data, aggregate formalstatus rows, aggregate bridge rows, or second seal coordinates. Thus none of these objects can repair the missing displayed face or support row, and no valid public $\mathsf{Real}^{\uparrow}$ seal route remains. □

Theorem 301.103 (RegSeqRat L10 source lattice pointer)**.** *Every accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier consumes the finite observation schedules of* $\mathsf{StreamName}^{\uparrow}$ *from chapter 126 and the endpoint and radius ledgers of* $\mathsf{DyadicRatCore}^{\uparrow}$ *from chapter 291 before it can be consumed by the* $\mathsf{Real}^{\uparrow}$ *seal surface of chapter 38. The source lattice is therefore the visible route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Along this route the accepted packet carries only scheduled finite stream windows, dyadic endpoint rows, dyadic radius rows, finite dyadic-distance ledgers, regularity ledgers, classifier rows, hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It does not expose quotient streams, host real equality, selected limits, ambient completeness, or a hidden completion carrier.*

Proof. Start from the carrier definition in Definition 301.1. Its stored coordinates are the scheduled finite windows supplied by $\mathsf{StreamName}^{\uparrow}$ together with the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius rows attached to those windows.

Next read the classifier in Definition 301.2. The classifier adds only finite dyadic-tolerance rows over the same scheduled observations: endpoint rows, radius rows, finite distance ledgers, regularity rows, and the displayed provenance needed to transport them.

The exact handoff theorem Definition 301.69 identifies the complete route from the stream-name schedule and dyadic ledgers through the $\mathsf{RegSeqRat}^{\uparrow}$ classifier to the $\mathsf{Real}^{\uparrow}$ seal rows. Applying Definition 301.22 then shows that a seal consumer reads this regular-rational source surface itself.

Since all cited surfaces are finite BEDC rows, no quotient stream, host real equality, selected limit, ambient completeness datum, or hidden completion carrier can be consumed along the route. □

Theorem 301.104 (RegSeqRat L10 source lattice exhaustion)**.** *Every accepted* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal read on the L10 route has exactly four row owners.* $\mathsf{StreamName}^{\uparrow}$ *owns the finite observation schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *owns endpoint rows, radius rows, and finite dyadic-distance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *owns regularity rows and classifier rows, and* $\mathsf{Real}^{\uparrow}$ *owns the terminal seal row. The accepted handoff has no row supplied by a quotient stream, host real equality, selected limit, completion witness, or ambient completeness principle.*

Proof. Start with the source lattice pointer, Definition 301.103. It fixes the route as $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ and lists the visible source rows before the seal can be read.

Apply the exact handoff theorem, Definition 301.69. It identifies the StreamName schedules, dyadic endpoint and radius ledgers, finite dyadic-distance ledgers, regularity rows, classifier rows, and seal rows as one finite handoff rather than as separate ambient services.

Finally use the Real seal handoff rows. They make the $\mathsf{Real}^{\uparrow}$ seal the terminal row of the same route, so the ownership list is exhausted by the four displayed chapters and cannot be enlarged by quotient streams, host equality, selected limits, completion witnesses, or ambient completeness. □

Theorem 301.105 (RegSeqRat StreamName Real handoff exhaustion)**.** *Every accepted L10 handoff that starts from* $\mathsf{StreamName}^{\uparrow}$ *schedules and is admitted to the* $\mathsf{Real}^{\uparrow}$ *seal through* $\mathsf{RegSeqRat}^{\uparrow}$ *is exhausted by exactly four row owners.* $\mathsf{StreamName}^{\uparrow}$ *owns the finite stream schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *owns the dyadic endpoint and radius rows together with finite dyadic-distance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *owns the regularity and classifier rows, and* $\mathsf{Real}^{\uparrow}$ *owns the terminal seal row. The handoff has no row owned by a quotient stream, host real equality, selected limit, completion witness, ambient completeness principle, or choice datum.*

Proof. Begin with the source lattice pointer and source lattice exhaustion, Definition 301.103 and Definition 301.104. Together they identify the visible L10 route

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

and assign the finite schedule, dyadic, regularity, classifier, and seal rows to the four displayed owners.

Apply the exact StreamName–dyadic–Real handoff theorem, Definition 301.69. It keeps the scheduled StreamName windows, dyadic endpoint and radius rows, finite dyadic-distance ledgers, RegSeqRat regularity rows, classifier rows, and Real seal row in one finite handoff rather than in separate ambient services.

The Lean-facing touchpoints are `RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff`, `DyadicRatCoreCarrier_regseqrat_bridge_handoff_exactness`, and `RegSeqRatStreamCarrier_real_seal_ha` They are precisely the schedule, dyadic bridge, and Real seal pieces named by the paper route.

Since the two lattice theorems already exclude quotient streams, host real equality, selected limits, completion witnesses, and ambient completeness, and since the exact handoff introduces no

extra owner, adding a choice datum or any other off-route owner would require a row absent from the displayed packet. □

Theorem 301.106 (RegSeqRat L10 row owner noncollapse)**.** *Every accepted L10 handoff from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *through* $\mathsf{RegSeqRat}^{\uparrow}$ *to the public* $\mathsf{Real}^{\uparrow}$ *seal has four pairwise noncollapsible row owners. The schedule rows belong to* $\mathsf{StreamName}^{\uparrow}$*; the endpoint, radius, and finite dyadic-distance ledgers belong to* $\mathsf{DyadicRatCore}^{\uparrow}$*; the regularity, classifier, and transport rows belong to* $\mathsf{RegSeqRat}^{\uparrow}$*; and the terminal seal row belongs to* $\mathsf{Real}^{\uparrow}$*.*

Removing any owner, or identifying any owner with another, breaks the accepted finite route or introduces one of the excluded coordinates: quotient streams, host real equality, selected limits, completion witnesses, ambient completeness data, or choice data.

Proof. The source lattice pointer Definition 301.103 fixes the visible route from $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. The source lattice exhaustion theorem Definition 301.104 assigns the route rows to exactly these four owners and excludes off-route owners.

The exact StreamName–dyadic–Real handoff Definition 301.69 then reads the schedule rows, dyadic endpoint and radius rows, dyadic-distance ledgers, regularity rows, classifier rows, transport rows, and public seal row as one finite route. Thus the four owners are not alternative names for one store: each contributes a different face of the displayed packet.

Apply the four-face necessity theorem Definition 301.76. If the schedule owner is removed, there is no finite source index; if the dyadic owner is removed, there are no endpoint, radius, or distance ledgers; if the RegSeqRat owner is removed, the classifier and regularity rows disappear; and if the Real owner is removed, the public seal has no terminal row. Identifying any two owners therefore erases one of these required faces rather than preserving the same accepted route.

Finally use finite-window saturation, Definition 301.79. It says that the accepted Real seal exit is already saturated by the four displayed finite faces. A collapse of owners can only be repaired by importing an off-route coordinate, and those coordinates are precisely the quotient streams, host real equality, selected limits, completion witnesses, ambient completeness data, and choice data excluded by the handoff. □

Theorem 301.107 (RegSeqRat L10 owner touchpoint coverage)**.** *On the L10 route from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *through* $\mathsf{RegSeqRat}^{\uparrow}$ *to the public* $\mathsf{Real}^{\uparrow}$ *seal, the Lean-facing handoff touchpoints* `RatStreamNameFiniteWindowClassifier_singleton_diagonal_handoff`*,* `DyadicRatCoreCarrier_regse` *and* `RegSeqRatStreamCarrier_real_seal_handoff` *cover exactly the finite StreamName schedule rows, the DyadicRatCore endpoint, radius, and finite dyadic-distance rows, the RegSeqRat regularity and classifier rows, and the RealUp terminal seal row.*

They do not supply quotient streams, host equality, selected limits, ambient completeness, choice data, standard bridge data, or any upgrade of the verification axis.

Proof. The source lattice pointer Definition 301.103 fixes the public route as

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

It lists the finite schedules, dyadic rows, regularity rows, classifier rows, and seal rows as the visible packet consumed by the handoff.

The StreamName–Real handoff exhaustion theorem Definition 301.105 identifies the same packet as exhausted by the four row owners. The row-owner noncollapse theorem Definition 301.106 then prevents any two owners from being read as alternative names for the same source.

The three Lean-facing touchpoints are the schedule component, the dyadic bridge component, and the terminal seal component of this packet. The StreamName touchpoint supplies the finite

schedule rows, the DyadicRatCore touchpoint supplies endpoint, radius, and finite dyadic-distance rows, and the RegSeqRat-to-Real touchpoint supplies the regularity, classifier, and terminal seal passage read by the public seal.

The roadmap discipline keeps the theory and verification axes separate. Thus these touchpoints report the Lean-facing handoff surface for the four-owner route, but they do not create quotient streams, host equality, selected limits, ambient completeness, choice data, standard bridges, or any stronger verification-axis status. □

Theorem 301.108 (RegSeqRat L10 face-order initiality)**.** *Every accepted L10 handoff from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *through* $\mathsf{RegSeqRat}^{\uparrow}$ *to the public* $\mathsf{Real}^{\uparrow}$ *seal reads the four faces in the initial order*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The finite stream schedules are read first, the dyadic endpoint, radius, and distance ledger is read second, the $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows are read third, and the public* $\mathsf{Real}^{\uparrow}$ *seal row is read last.*

Permuting any face, deleting any face, or inserting a fifth face fails to be an accepted read by a $\mathsf{Real}^{\uparrow}$ *seal consumer. The only possible repairs would introduce one of the excluded coordinates: quotient streams, host real equality, selected limits, completion witnesses, ambient completeness data, choice data, bridge rows, or verification-axis upgrades.*

Proof. The source lattice pointer Definition 301.103 fixes the public route from $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. The route already places the finite stream schedules before the dyadic ledger, the dyadic ledger before the regularity and classifier rows, and those rows before the terminal seal.

The source lattice exhaustion theorem Definition 301.104 says that the accepted route has exactly the four displayed owners. Thus an order change cannot be explained by appealing to a hidden owner between two faces or behind the seal.

Apply row-owner noncollapse, Definition 301.106. A permutation would make a later row available before its source face is present; a deletion would erase a required owner; and an insertion would require an additional owner. Each case breaks the finite route rather than producing the same accepted handoff.

Finally use owner touchpoint coverage, Definition 301.107. The Lean-facing rows seen by the seal consumer are exactly the schedule rows, the dyadic bridge rows, the regularity and classifier rows, and the terminal seal row. Any consumer that attempts another order must import quotient streams, host real equality, selected limits, completion witnesses, ambient completeness data, choice data, bridge rows outside the handoff, or a verification-axis upgrade, all of which are outside the accepted L10 packet. □

Theorem 301.109 (RegSeqRat tail-budget meet handoff)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier and let a tail-budget consumer of P request a completion-facing read. The consumer first meets the displayed* $\mathsf{StreamName}^{\uparrow}$ *pointwise schedule from chapter 126, then reads the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance row from chapter 291, then passes through the regular-rational classifier rows of P, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38. Thus every tail-budget read factors through one regular-Cauchy route before any completion-facing surface is visible.*

Proof. Start with the L10 source lattice pointer of Definition 301.103. It places the accepted regular-rational packet on the displayed route $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$.

Next apply the common dyadic tail ledger of Definition 301.74. A tail-budget consumer must read the cofinal finite ledger consisting of the stream schedule, dyadic endpoint and radius rows, distance rows, regularity rows, classifier rows, transport rows, continuation routes, provenance, and local naming.

Finally use the exact handoff of Definition 301.69. The completion-facing read is therefore downstream of the same finite regular-Cauchy route and cannot bypass the StreamName schedule, the DyadicRatCore tolerance row, or the RealUp seal row. □

Theorem 301.110 (RegSeqRat StreamName dyadic source lock)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, the displayed* $\mathsf{StreamName}^{\uparrow}$ *pointwise name and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row are locked to the same finite source. Any two consumers that present the same tail budget, the same stream schedule row, and the same dyadic endpoint and radius ledger must read the same regular-rational classifier route before reaching a* $\mathsf{Real}^{\uparrow}$ *seal. No separate quotient stream, host real equality, selected limit, ambient completion datum, or second dyadic source can be inserted between those reads.*

Proof. Use Definition 301.70. It identifies public seal reads through displayed StreamName schedules and DyadicRatCore ledgers as reads of one finite source route.

Then apply Definition 301.103. The source route already contains the schedule rows, endpoint rows, radius rows, distance rows, classifier rows, regularity rows, hsame transports, Cont routes, Pkg provenance, and local NameCert rows.

The same route is the input to the RealUp handoff by Definition 301.22. Since the carrier exposes no row for the excluded quotient, host equality, selected limit, completion datum, or second dyadic source, the two consumers remain locked to the displayed StreamName and DyadicRatCore source. □

Theorem 301.111 (RegSeqRat Real seal consumer factorization)**.** *Every completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet factors through the three visible rows named by the regular-Cauchy route: the* $\mathsf{StreamName}^{\uparrow}$ *pointwise schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the* $\mathsf{Real}^{\uparrow}$ *seal row that consumes the regular-rational classifier surface. The factorization preserves the displayed* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *row, and it has no route to a hidden completion carrier or an ambient completeness principle.*

Proof. Apply Definition 301.109. It orders every tail-budget read through StreamName, DyadicRatCore, RegSeqRat, and RealUp before any completion-facing surface is visible.

Apply Definition 301.110. The StreamName schedule and DyadicRatCore tolerance row cannot be separated into two sources once the accepted regular-rational packet is fixed.

Finally use Definition 301.50. The public RealUp consumer is determined by the finite regular-rational source surface and its displayed transport, continuation, package, and naming rows. Thus the consumer factorization is exactly the regular-Cauchy route stated above, with no hidden completion or ambient completeness read. □

Theorem 301.112 (RegSeqRat tail-budget Real seal nonescape)**.** *A completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet cannot bypass the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface, or the displayed* $\mathsf{Real}^{\uparrow}$ *seal row at the tail-budget boundary. Every such consumer factors through the same regular-Cauchy route with its displayed* hsame, Cont, Pkg, *and local* NameCert *rows.*

The boundary has no hidden completion carrier, selected limit, quotient stream, ambient completeness principle, or second dyadic source.

Proof. Apply Definition 301.109. It orders the completion-facing read through the StreamName schedule, DyadicRatCore tolerance row, RegSeqRat classifier rows, and RealUp seal row before any completion-facing surface is visible.

Next apply Definition 301.110. The stream schedule and dyadic tolerance row are locked to one finite source attached to the accepted regular-rational packet.

Finally use Definition 301.111. It identifies the completion-facing consumer with the displayed regular-Cauchy route and its transport, continuation, package, and naming rows. Since that route has no coordinate for a hidden completion carrier, selected limit, quotient stream, ambient completeness principle, or second dyadic source, the consumer cannot escape the stated boundary. □

Theorem 301.113 (RegSeqRat tail-budget window-seal pullback)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let R be any completion-facing tail-budget consumer of P. The visible pullback of R along the public* $\mathsf{Real}^{\uparrow}$ *seal is exactly the finite packet consisting of the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance row, and the local* NameCert *row.*

Consequently every completion-facing tail-budget read reaches $\mathsf{Real}^{\uparrow}$ *only through that same finite* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *route. The pullback exposes no quotient stream, selected host limit, ambient completeness carrier, second dyadic source, or additional Real seal coordinate.*

Proof. Begin with Definition 301.109. It orders the consumer route through the displayed StreamName schedule row, DyadicRatCore tolerance ledger, RegSeqRat classifier surface, and RealUp seal before any completion-facing read is visible.

Next apply Definition 301.110. The schedule row and dyadic endpoint, radius, and distance rows are locked to one finite source attached to the accepted packet. Thus the pullback cannot split into a second dyadic source or a quotient stream route.

Now use Definition 301.111. The RealUp consumer factors through the same finite packet, preserving the displayed hsame transports, Cont routes, Pkg provenance, and local NameCert row.

Finally apply Definition 301.112. The nonescape boundary removes the remaining excluded coordinates: selected host limits, ambient completeness carriers, and additional Real seal coordinates. Hence the visible pullback is precisely the stated finite route. □

Theorem 301.114 (RegSeqRat tail-budget terminal Real seal)**.** *Every completion-facing tail-budget consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the single finite route consisting of the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance row, local* NameCert *naming, and the public* $\mathsf{Real}^{\uparrow}$ *seal row. This terminal surface exports no quotient stream, selected host limit, ambient completeness carrier, choice data, or second seal coordinate.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatTailBudgetTerminalRealSeal`

Proof. First build the tail-budget route. Apply Definition 301.109 to force the completion-facing read through the displayed StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier surface, and RealUp seal.

Next lock the source rows. By Definition 301.110, the schedule row and the endpoint, radius, and distance ledger belong to one finite source attached to the accepted packet, so a quotient stream or second dyadic source cannot enter.

Now identify the public consumer surface. Use Definition 301.111 to preserve the displayed hsame transports, Cont routes, Pkg provenance, and local NameCert row on the way to the public RealUp seal.

Finally combine Definition 301.112 with Definition 301.113. These rows remove selected host limits, ambient completeness carriers, choice data, hidden completion coordinates, and additional Real seal coordinates, leaving exactly the finite route stated above. □

Theorem 301.115 (RegSeqRat tail-budget composition correspondence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Suppose two completion-facing tail-budget consumer routes of P are composable and suppose a certified composite consumer route is displayed for their two-step read. The two-step read and the certified composite determine the same finite terminal* $\mathsf{Real}^{\uparrow}$ *seal surface exactly when they preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

No quotient stream, selected host limit, countable choice datum, ambient completeness carrier, metric-completeness row, or second seal coordinate is part of this correspondence.

Proof. First put the component consumers on the common route. Apply Definition 301.109 to both component routes, so each read is displayed on the four-face path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next compare the two finite-window presentations. The shared-window reindex composition law of Definition 301.162 identifies the two-step presentation with the certified composite exactly on the displayed StreamName schedule and DyadicRatCore ledger rows.

Now pull the Real seal read back along that finite window. Use Definition 301.113 to preserve the RegSeqRat classifier and regularity rows together with the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply Definition 301.114. The terminal seal surface is therefore the same for the two-step route and the certified composite. The nonescape theorem Definition 301.112 removes quotient streams, selected host limits, countable choice data, ambient and metric completeness rows, and second seal coordinates from the comparison. □

Theorem 301.116 (RegSeqRat dyadic-stream tail-budget common refinement)**.** *Let P and Q be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget presentations whose* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers admit a displayed common finite window. The two presentations determine the same terminal* $\mathsf{Real}^{\uparrow}$ *seal surface exactly when that common window preserves the* $\mathsf{StreamName}^{\uparrow}$ *pointwise classifier, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, and the* $\mathsf{Real}^{\uparrow}$ *source factorization rows.*

The common-refinement certificate is only this finite BEDC route through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It contains no quotient stream, selected host limit, countable choice datum, ambient completeness carrier, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First view the two accepted presentations as composable tail-budget consumer routes through the displayed four-face path. Apply Definition 301.115; it reduces agreement of the terminal $\mathsf{Real}^{\uparrow}$ seal surface to preservation of the StreamName schedule row, DyadicRatCore ledger rows, RegSeqRat classifier and regularity rows, the terminal RealUp seal row, and the displayed support rows.

Next identify the common finite window. The shared-window reindex composition law of Definition 301.162 supplies the finite comparison of the two StreamName schedules and DyadicRatCore

ledgers. The pullback theorem Definition 301.113 transports the RegSeqRat classifier rows and the Real-facing support rows along that window.

Finally apply Definition 301.114. The terminal seal row is therefore determined by exactly the four preserved surfaces named in the statement. The later Lean route is represented by `BEDC.Derived.RegSeqRatUp.RegSeqRatStreamCarrier_common_refinement_classifier_determinacy`, `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreRegSeqRatWindow_radius_handoff`, `BEDC.Derived.StreamNa` and `BEDC.Derived.RealUp.RealNameSeal_source_factorization`; no Lean marker is asserted here. □

Theorem 301.117 (RegSeqRat finite tail-budget family common refinement)**.** *Let F be a finite family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget presentations. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *schedules and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers for every pair of members of F admit displayed common finite windows. Then F determines one terminal* $\mathsf{Real}^{\uparrow}$ *seal surface exactly when the common finite refinement preserves, for every member of F, the* $\mathsf{StreamName}^{\uparrow}$ *pointwise classifier, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *source factorization rows, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

The finite-family certificate is only the common BEDC refinement route. It contains no quotient stream, selected host limit, countable choice datum, ambient completeness carrier, metric-completeness row, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Induct on the finite family F. For the empty and singleton families, the claimed terminal seal surface is the displayed accepted presentation itself, and the listed support rows are exactly the rows already carried by its tail-budget certificate.

For the induction step, separate the last accepted presentation from the refined family already produced by the induction hypothesis. The pairwise window assumptions give common finite windows between the last presentation and each member of the refined family, while the induction hypothesis supplies the already preserved StreamName classifiers, DyadicRatCore radius handoffs, RegSeqRat classifier and regularity rows, RealUp source factorization rows, and displayed support.

Apply Definition 301.115 to the refined family route and the last presentation. This is the binary step: it keeps the terminal $\mathsf{Real}^{\uparrow}$ seal read on the same four-face path and records that no extra route data is introduced by composing the two tail-budget certificates.

Next use Definition 291.93. It identifies the nested $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance handoffs across the finite chain of common windows, so the composite window is coherent with the earlier dyadic radius handoff rows.

Finally apply Definition 301.116 to compare the terminal seal surface of the composite refinement with the seal surface of each accepted family member. The forward direction reads preservation of the listed rows from the common finite refinement; the reverse direction repacks those preserved rows into the same terminal $\mathsf{Real}^{\uparrow}$ seal surface.

The nonescape clauses remain facewise throughout the induction. Since every step uses only the displayed BHist, hsame, Cont, Pkg, and NameCert support, the finite-family route has no quotient streams, selected host limits, countable choice data, ambient or metric-completeness rows, or second $\mathsf{Real}^{\uparrow}$ seal coordinates. □

Theorem 301.118 (RegSeqRat tail-refinement handoff)**.** *Let F be a finite family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget windows with a common refinement in the sense of Definition 301.117. Then every completion-facing consumer receives the same* $\mathsf{Real}^{\uparrow}$ *seal handoff after the StreamName window transport and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparison have been displayed. The*

handoff preserves the $\mathsf{StreamName}^{\uparrow}$ *schedule classifier,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support.*

The handoff exports no quotient stream equality, selected host limit, countable-choice row, ambient completeness carrier, metric-completeness row, duplicate $\mathsf{Real}^{\uparrow}$ *seal coordinate, aggregate formalstatus token, or aggregate bridge row.*

Proof. Start with the common refinement supplied by Definition 301.117. It places all members of the finite family on one visible support surface preserving the StreamName schedules, DyadicRatCore endpoint/radius/distance rows, RegSeqRat classifier and regularity rows, RealUp source-factorization rows, and displayed structural support.

Next compare one family member with that common support. Apply Definition 301.116. The StreamName window transport and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance comparison are therefore read before the RegSeqRat classifier and terminal Real seal can be consumed.

Finally apply Definition 301.111. The Real-facing consumer sees the same source factorization rows after the window transport and dyadic tolerance comparison, so each member of F hands off to the same terminal $\mathsf{Real}^{\uparrow}$ seal surface.

The exclusions are inherited from the finite-family refinement, the dyadic-stream common refinement, and the Real seal consumer factorization. These theorems keep the route inside the displayed finite rows and provide no quotient stream, selected host limit, countable-choice, completeness, duplicate-seal, aggregate-status, or aggregate-bridge coordinate. □

Theorem 301.119 (RegSeqRat tail-budget facewise status pullback)**.** *Every accepted tail-budget* $\mathsf{Real}^{\uparrow}$*-seal consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet reads the four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with each face's own closurestatus, formalstatus, leantarget, and bridgestatus rows. The consumer route reads those rows from the local face records and not from an aggregate formalstatus token.

Conversely, a tail-budget packet carrying exactly those four local status records together with the displayed

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support rows gives the same accepted tail-budget terminal $\mathsf{Real}^{\uparrow}$*-seal route. This pullback has no fifth face, quotient stream equality, selected host limits, choice data, ambient completeness carrier, metric-completeness datum, standard-bridge row, or aggregate audit certificate.*

Proof. First locate the finite route. Apply Definition 301.114. It confines every completion-facing tail-budget consumer to the StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier and regularity rows, displayed hsame transports, Cont routes, Pkg provenance, local $\mathsf{NameCert}$ naming, and public $\mathsf{Real}^{\uparrow}$ seal.

Next read the status data face by face. Apply Definition 38.818. It identifies the terminal seal status read with the four local closurestatus, formalstatus, leantarget, and bridgestatus rows and rejects a route-level formalstatus token or aggregate bridge row.

For the phase boundary, apply Definition 38.471. The current Phase Real and Completion exit read is the product of the four local face records, so the tail-budget route cannot contribute a fifth face or cached completion status.

Finally reuse the nonescape clauses of Definition 301.114. They exclude quotient stream equality, selected host limits, choice data, ambient completeness, metric-completeness data, standard-bridge rows, and any additional Real seal coordinate. The converse is then the same finite packet read with exactly the four local status records and the displayed support rows. □

Theorem 301.120 (RegSeqRat tail-budget support refinement invariance)**.** *Let an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget route terminate in a completion-facing* $\mathsf{Real}^{\uparrow}$ *seal row. Suppose this route is replaced by a finite certified support refinement whose finite windows preserve the* $\mathsf{StreamName}^{\uparrow}$ *schedule classifier, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support rows.

Then the original route and the refined route determine the same terminal completion-facing $\mathsf{Real}^{\uparrow}$ *seal surface. The common seal is read through the same facewise closurestatus, formalstatus, leantarget, and bridgestatus rows, and not through quotient streams, selected host limits, countable choice, ambient completeness, metric completeness, a second* $\mathsf{Real}^{\uparrow}$ *seal coordinate, or an aggregate formalstatus token.*

Proof. First compare a single replacement step. Apply Definition 301.116. It supplies the common finite window in which the $\mathsf{StreamName}^{\uparrow}$ classifier, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance rows, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows are preserved on both sides of the handoff.

Next pass from one replacement step to the finite certified support refinement. Apply Definition 301.117. The finite family certificate keeps the same displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows while threading the preserved dyadic and regular-sequence rows through the whole family.

Now read the terminal surface. Apply Definition 301.114. Since both routes have the same finite tail-budget support and the same terminal $\mathsf{Real}^{\uparrow}$ seal row, the completion-facing seal surface determined by the original route is the seal surface determined by the refined route.

Finally apply Definition 301.119. The status data used in the conclusion is read objectwise from the four faces and from the displayed support rows. Hence the refinement cannot introduce quotient streams, selected host limits, countable choice, ambient or metric completeness, a second seal coordinate, or any aggregate formalstatus token. □

Theorem 301.121 (RegSeqRat tail-budget chain refinement exhaustion)**.** *Let an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget route terminate in a completion-facing* $\mathsf{Real}^{\uparrow}$ *seal row, and let* $R_0, \ldots, R_n$ *be a finite nonempty chain of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget routes with* R_0 *the original route and* R_n *the terminal route. Suppose every adjacent replacement* (R_i, R_{i+1}) *is a certified support refinement preserving the* $\mathsf{StreamName}^{\uparrow}$ *schedule classifier, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}$$

support rows.

Then every parenthesized finite refinement chain from R_0 *to* R_n *and the direct first-to-last comparison determine the same terminal completion-facing* $\mathsf{Real}^{\uparrow}$ *seal surface. The common surface is exactly the terminal seal read from the preserved tail-budget support rows, with the facewise closurestatus, formalstatus, leantarget, and bridgestatus rows pulled back from the displayed faces. It exports no quotient streams, selected host limits, countable choice, ambient completeness, metric completeness, duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates, aggregate formalstatus tokens, or aggregate bridge rows.*

Proof. The base case is the one-route chain. The comparison from R_0 to R_0 is reflexive on the displayed tail-budget support, so the completion-facing $\mathsf{Real}^{\uparrow}$ seal surface is the terminal surface already carried by that route.

For a one-step chain, apply Definition 301.120. The certified support refinement preserves the StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier and regularity rows, displayed support rows, and terminal $\mathsf{Real}^{\uparrow}$ seal row, hence the two routes read the same terminal seal surface.

For a longer finite chain, first forget the parenthesization and read the chain as a finite request support-face chain. Apply Definition 301.150. It makes all adjacent support-face composites and the direct first-to-last comparison land on one terminal $\mathsf{Real}^{\uparrow}$ seal-facing support face over the displayed rows.

Next build the common finite support needed by the tail-budget route. Apply Definition 301.117. The common refinement keeps the finite StreamName windows, the DyadicRatCore endpoint, radius, and distance ledger, the RegSeqRat classifier and regularity rows, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows on one finite support family.

Now read the terminal completion-facing surface. Apply Definition 301.114. Since the direct comparison and every parenthesized chain have the same finite tail-budget support and the same terminal $\mathsf{Real}^{\uparrow}$ seal row, they determine the same completion-facing $\mathsf{Real}^{\uparrow}$ seal surface.

Finally apply Definition 301.119. The status and bridge data are read objectwise from the preserved faces and not from an aggregate token. Therefore the finite chain cannot introduce quotient streams, selected host limits, countable choice, ambient or metric completeness, duplicate $\mathsf{Real}^{\uparrow}$ seal coordinates, aggregate formalstatus tokens, or aggregate bridge rows. □

Theorem 301.122 (RegSeqRat tail-budget fusion)**.** *Let B_0 and B_1 be compatible finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget rows sharing the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then B_0 and B_1 fuse to one terminal budget route whose visible surface is the shared StreamName schedule, dyadic tolerance ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local* NameCert *rows. The theory-status and bridge-status axes remain objectwise rows of that terminal route.*

Proof. Start from the common-tail cocycle descent theorem Definition 301.190. It identifies compatible finite tail windows on common overlaps, so the two budget rows can be compared through the same StreamName schedule and dyadic tolerance ledger.

Apply the tail-budget handoff theorem Definition 301.109. The shared schedule, tolerance ledger, and RegSeqRat readback are therefore read before the terminal Real seal, and the displayed hsame, Cont, Pkg, and local NameCert rows preserve the same finite route.

The fused route is terminal because every Real-facing budget consumer has already passed through the common-tail descent and the meet handoff. Since status and bridge data are carried only as displayed objectwise rows, the fusion does not identify axes globally, add quotient streams, select limits, invoke countable choice, or import ambient completeness. □

Theorem 301.123 (RegSeqRat tail-selector RealUp seal exhaustion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let a completion-facing* $\mathsf{Real}^{\uparrow}$ *seal consumer use a RegSeqRat tail selector to choose its finite readout. Then the consumer factors through exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, and the local* $\mathsf{Real}^{\uparrow}$ *seal provenance attached to P.*

Equivalently, the tail-selector route exposes only the same finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *source surface as the terminal tail-budget handoff. It exports no quotient stream,*

host real equality, selected limit, completion point, metric-completeness datum, or extra seal coordinate.

Proof. First apply Definition 301.114. It places every completion-facing tail-budget consumer on the single finite route whose rows are the displayed StreamName schedule, DyadicRatCore endpoint, radius, and distance ledger, RegSeqRat classifier and regularity ledgers, transport, continuation, package provenance, local naming, and public RealUp seal row.

Next use Definition 301.35. The tail-selector read selects only a finite public window cover of that same accepted packet, and the cover exhaustion identifies each selected window with the displayed schedule, pointwise classifier, dyadic ledger, and provenance rows.

Now apply Definition 301.36. Any two selector-visible seal reads over the cover repack through one accepted common finite window, so the selected tail window cannot introduce a second quotient stream, selected limit, completion point, or metric-completeness coordinate.

Finally invoke Definition 301.22. The RealUp seal consumes exactly the regular-rational source surface assembled above. Since that surface has no projection for host real equality or an additional seal coordinate, the tail-selector consumer is exhausted by the displayed schedule, ledger, regularity, classifier, and local seal-provenance rows. □

Theorem 301.124 (RegSeqRat tail-selector window-seal lock)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let a* $\mathsf{Real}^{\uparrow}$ *seal consumer use the RegSeqRat tail selector carried by P. The consumer must read, in order, the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, the RegSeqRat tail-budget handoff, and the local* $\mathsf{Real}^{\uparrow}$ *seal row. It cannot replace the selected window by quotient-stream equality, host real equality, a selected limit, a completion point, or an ambient completeness datum.*

Proof. Start from tail-selector RealUp seal exhaustion in Definition 301.123. It places the consumer on the finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route and excludes any detached seal coordinate.

Apply the tail-budget meet handoff of Definition 301.109. The selected window is not optional data; it is the finite readout through which the tail-budget route reaches the regular-rational meet surface.

Next use Definition 301.69. The StreamName window and dyadic radius ledger travel together into the Real-facing handoff, so a consumer cannot keep the seal while substituting a quotient stream, host equality, selected limit, or completion point for the selected finite window.

Finally invoke Definition 301.22. The Real seal consumes exactly the preceding finite handoff surface, and therefore the local lock is the ordered route through the selected window, dyadic ledger, tail-budget handoff, and Real seal row. □

Theorem 301.125 (RegSeqRat tail-selector cofinal request determinacy)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be public finite requests to P whose readouts use certified cofinal RegSeqRat tail selectors. Suppose both requests preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite dyadic-distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the componentwise* hsame *rows tying those data to the request packet, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Then S and T determine the same* $\mathsf{Real}^{\uparrow}$ *seal-visible source section.*

The determinacy is only for the seal-visible source section. It does not identify quotient streams, host real equality, selected limits, completion points, ambient completeness data, choice data, or second seal coordinates.

Proof. First apply Definition 301.123. Each cofinal tail-selector request is exhausted by the displayed $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ source surface, so neither request has a detached completion coordinate from which extra seal rows could be read.

Next use the window-seal lock, Definition 301.124. The certified cofinal selector must pass through the selected finite $\mathsf{StreamName}^{\uparrow}$ window, the dyadic radius ledger, the RegSeqRat tail-budget handoff, and the local Real seal row. Since S and T preserve those displayed rows, their selector routes reach the same locked finite window surface.

Now factor both requests through Definition 301.139. The source section of each request consists exactly of the schedule rows, dyadic endpoint, radius, distance, classifier, regularity, componentwise hsame, displayed Cont, package provenance, and local naming rows named in the statement. The hypotheses identify those rows component by component, so the two finite source sections are the same public BHist section.

Finally apply Definition 301.141 and Definition 301.114. The correspondence restricts the Real seal read to the finite source section just identified, and the terminal tail-budget theorem prevents a second selector, limit, completion-point, completeness, choice, or seal-coordinate channel from entering the route. Therefore the two certified cofinal requests determine the same $\mathsf{Real}^{\uparrow}$ seal-visible source section and nothing beyond that section. □

Theorem 301.126 (RegSeqRat tail-selector cofinal refinement monotonicity)**.** *Let P be an accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, let S be a public finite request served by a certified cofinal RegSeqRat tail selector, and let S' be a certified cofinal refinement of S. Suppose S' preserves the displayed $\mathsf{StreamName}^{\uparrow}$ window rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and finite dyadic-distance ledgers, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows, the componentwise hsame rows, the displayed Cont routes, the Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows of S. Then S and S' determine the same $\mathsf{Real}^{\uparrow}$ seal-visible source section.*

Conversely, any certified cofinal refinement of S that determines the same $\mathsf{Real}^{\uparrow}$ seal-visible source section must preserve exactly those displayed rows. The monotonicity claim is only for the finite L10 $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route. It does not expose quotient streams, selected limits, choice data, metric-completeness rows, aggregate status rows, bridge rows, second seal coordinates, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. First apply Definition 301.124. Both S and S' are forced through the selected finite $\mathsf{StreamName}^{\uparrow}$ window, the dyadic radius ledger, the RegSeqRat tail-budget handoff, and the local Real seal row. Since S' is a certified cofinal refinement preserving the displayed rows of S, the refinement cannot change the locked window-seal route.

Next use Definition 301.125. Regard S and S' as the two certified cofinal requests in that theorem. The preserved schedule, dyadic endpoint, radius, distance, classifier, regularity, hsame, Cont, package, and local naming rows identify their seal-visible request surfaces, so the two requests determine the same $\mathsf{Real}^{\uparrow}$ source section.

Read that source section through Definition 301.139. This identifies the section with exactly the displayed rows named in the statement, not with a quotient stream, selected limit, choice datum, metric-completeness row, aggregate status row, bridge row, or ambient completion claim.

For the converse, suppose a certified cofinal refinement determines the same $\mathsf{Real}^{\uparrow}$ seal-visible source section. The finite-request source-section description forces equality on each displayed row of the section. Finally apply Definition 301.114: the terminal seal has no additional channel through which a changed endpoint, radius, distance, classifier, regularity, hsame, Cont, package, or local NameCert row could be hidden. Hence same-section cofinal refinements are precisely the refinements that preserve those displayed rows. □

Theorem 301.127 (RegSeqRat tail-selector bundle induction Real seal)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let $\mathcal{P}$:* ProbeBundle *be a finite bundle of certified unary observations, and suppose one certified cofinal RegSeqRat tail selector serves every bundled observation row equipped with its* UnaryHistory *and* InBundle *witnesses. Let $\mathcal{F}$ be a finite family of* $\mathsf{Real}^{\uparrow}$ *seal requests over P whose members preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite dyadic-distance ledgers, the RegSeqRat classifier and regularity rows, the componentwise* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Then the entire family factors through one BEDC-visible* $\mathsf{Real}^{\uparrow}$ *seal source section.*

The factorization is exactly the finite source section assembled from the bundle rows, the shared tail selector, the common dyadic scale-window ledger, the RegSeqRat classifier and regularity data, componentwise hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It exports no quotient stream, host real equality, selected limit, completion point, countable-choice selector, metric-completeness datum, aggregate status row, bridge row, second seal coordinate, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First collect the finite observations. Apply Definition 126.547 to the shared tail-selector route and the finite ProbeBundle, and then use Definition 126.548. These results place every bundled seal request on one displayed StreamName packet whose rows are the unary observation certificates, the bundle-membership witnesses, the shared finite-window budget, pointwise RegSeqRat handoff rows, and the support rows hsame, Cont, Pkg, and NameCert.

Next attach the dyadic face to that packet. Apply Definition 291.82 to the finite dyadic faces and, equivalently at the terminal boundary, use Definition 291.83. The common window supplies the endpoint, radius, distance, scale, transport, continuation, package, and local DyadicRatCore naming rows consumed by the StreamName and RegSeqRat faces.

Now run induction over the finite family $\mathcal{F}$. The empty family is served by the collected bundle packet itself. For the step adjoining one seal request, apply Definition 301.124 to place the new request on the selected StreamName window, dyadic ledger, tail-budget handoff, and local Real seal row. Then compare it with the already collected family through Definition 301.125; the preserved schedule, dyadic, classifier, regularity, support, provenance, and local naming rows force the same seal-visible source section.

Finally apply Definition 301.114 and Definition 301.22. The terminal tail-budget result keeps the finite family on the single regular-rational source route, and the Real handoff identifies that route with the public $\mathsf{Real}^{\uparrow}$ seal source surface. Since none of the cited packets contains rows for quotient streams, host real equality, selected limits, completion points, countable choice, metric completeness, aggregate status, bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the family factors through exactly one BEDC-visible seal source section. □

Theorem 301.128 (RegSeqRat tail-selector finite-cover gluing)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{U}$ be a finite certified cover of* $\mathsf{Real}^{\uparrow}$ *seal requests over P. Suppose every member of $\mathcal{U}$ is served by a certified cofinal RegSeqRat tail selector, and suppose every overlap in the cover preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite dyadic-distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier rows, the componentwise* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Then the finite cover glues through one BEDC-visible* $\mathsf{Real}^{\uparrow}$ *seal source section.*

The glued section is only the finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal route displayed by those requests. It does not identify quotient streams, host real equality, selected limits, completion points, countable-choice selectors, metric-completeness rows, aggregate status rows, bridge rows, second seal coordinates, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First treat the finite cover as a finite family of seal requests over the accepted packet P. Apply Definition 301.127 to collect the cover members into one seal-visible source section whose rows are the displayed StreamName windows, dyadic endpoint, radius, and distance ledgers, RegSeqRat classifier and regularity rows, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ data.

Next check agreement on overlaps. For two cover members, use Definition 301.125. The overlap hypotheses identify exactly the displayed schedule, dyadic, classifier, regularity, support, provenance, and local naming rows, so the two members determine the same $\mathsf{Real}^{\uparrow}$ seal-visible source section on their common subrequest.

Certified refinements do not change the glued section. Apply Definition 301.126 whenever an overlap is represented by a smaller cofinal tail-selector request. The refinement keeps the same displayed rows, and the converse direction in that theorem rules out a hidden change in endpoint, radius, classifier, support, package, or local naming data.

Now attach the dyadic face uniformly across the cover. Use Definition 291.83 to read the common scale-window endpoint, radius, distance, transport, continuation, package, and local naming rows as one four-face dyadic pullback ledger. This is the dyadic component consumed by the RegSeqRat tail-selector route on every cover member.

Finally pass from the glued regular-rational section to the Real seal route. Apply Definition 301.22 together with Definition 301.114. These results keep the glued cover on the finite Real seal source surface and leave no BEDC-visible row for quotient streams, host real equality, selected limits, completion points, countable choice, metric completeness, aggregate status, bridge data, a second seal coordinate, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 301.129 (RegSeqRat tail-selector diagonal cover source coherence)**.** *Let P be an accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, and let $\mathcal{U}$ be a finite certified cover of $\mathsf{Real}^{\uparrow}$ seal requests over P. Suppose every member of $\mathcal{U}$ is served by a certified cofinal RegSeqRat tail selector. Suppose also that every pairwise overlap preserves the displayed $\mathsf{StreamName}^{\uparrow}$ window rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and finite dyadic-distance ledgers, the RegSeqRat classifier and regularity rows, the componentwise hsame rows, the displayed Cont routes, the Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows. Then all cover members determine one diagonal BEDC-visible $\mathsf{Real}^{\uparrow}$ seal source section.*

The diagonal source section is the single finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ source section seen simultaneously by every cover member and every overlap. It contains no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness row, aggregate status row, bridge row, second seal coordinate, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum.

Proof. First glue the finite cover at the visible seal surface. Apply Definition 301.128. This produces one BEDC-visible $\mathsf{Real}^{\uparrow}$ seal source section whose displayed rows are exactly the StreamName windows, dyadic endpoint, radius, and distance ledgers, RegSeqRat classifier and regularity rows, componentwise hsame, displayed Cont, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows carried by the cover.

Next make the gluing independent of the chosen cover member. For each member, apply Definition 301.125. The overlap hypotheses identify the same schedule, dyadic ledger, classifier, regularity, support, provenance, and local naming rows, so the section read from that member is the section read from the glued cover.

Now compare different cofinal tail selectors that serve the same overlap. Apply Definition 301.126. Refining a request may shrink the finite tail window, but it preserves the displayed rows that feed the seal source section. Hence all refinements still read the same diagonal section.

The remaining coherence is the common finite window. Use Definition 126.547 to lock the cofinal StreamName windows across the cover, and use Definition 291.83 to read the endpoint, radius, distance, transport, continuation, package, and local naming rows through the same four-face dyadic pullback.

These two locks leave only the displayed finite source section. Since the cited packets export no quotient streams, selected limits, host real equality, countable choice, metric completeness, aggregate status, bridge data, second seal coordinate, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no extra BEDC-visible row can separate one cover member from another. The cover members therefore determine one diagonal $\mathsf{Real}^{\uparrow}$ seal source section. □

Definition 301.130 (RegSeqRat constant dyadic source)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packet for a displayed dyadic endpoint* d *is the finite packet obtained by reading one* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row for* d*, the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule that returns that same row at every requested unary window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity ledger that compares any two displayed positions by the same endpoint row. Its visible rows are the dyadic endpoint row, the dyadic radius row requested by each window, the zero dyadic-distance row for equal endpoints, the regularity bound row, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance.*

The packet is only a finite BEDC source description. It is not a quotient stream, host real equality, selected limit, metric completion object, Real completeness row, or ambient embedding principle.

Theorem 301.131 (RegSeqRat constant dyadic regularity)**.** *Every finite window of a* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source is regular. For requested precisions* n *and* m*, both displayed endpoints are the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* d*, the dyadic-distance row is the equal-endpoint row, and the regularity ledger is the displayed bound*

$$0 \leq 2^{-n} + 2^{-m}.$$

Thus the window supplies exactly the RegSeqRat regularity row over the constant StreamName schedule, together with its hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

Proof. Project the constant dyadic source of Definition 301.130. The two endpoint projections in any finite window are the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, because the $\mathsf{StreamName}^{\uparrow}$ schedule is constant.

Read the equal-endpoint dyadic-distance row from chapter 291. The radius ledger for the same window supplies the two displayed radii 2^{-n} and 2^{-m}, so the regularity comparison is bounded by the displayed sum.

Finally apply the RegSeqRat regularity obligation surface of Definition 301.3 and the finite-window transport closure of Definition 301.6. They repack the constant StreamName schedule of chapter 126 and the dyadic endpoint rows as the required RegSeqRat regularity ledger, without adding quotient or completion data. □

Theorem 301.132 (RegSeqRat constant RealUp seal handoff)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the four visible faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The source first reads the constant StreamName schedule, then the shared DyadicRatCore endpoint and radius rows, then the RegSeqRat regularity and classifier ledger of Definition 301.131, and then the RealUp seal row of chapter 38. The handoff exports no quotient stream equality, host real

equality, completion object, selected limit, ambient embedding theorem, or RealUp completeness row.

Proof. Start with the constant schedule and endpoint packet of Definition 301.130. Its source rows are already the StreamName and DyadicRatCore rows named by chapter 126 and chapter 291.

Use Definition 301.131 to place every finite window on the RegSeqRat regularity surface. The cited theorem supplies the same endpoint, radius, distance, transport, route, package, and naming rows in each requested window.

Finally apply the existing RegSeqRat seal route: Definition 301.69, Definition 301.109, and Definition 301.22. These route the finite regular-rational packet to the RealUp seal through the four visible faces. Since none of those faces contains a coordinate for quotient stream equality, host real equality, completion objects, selected limits, embedding theorems, or RealUp completeness, the constant packet exports no such row. □

Theorem 301.133 (RegSeqRat constant dyadic classifier reflection)**.** *For two* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packets over displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints d and e, the* $\mathsf{RegSeqRat}^{\uparrow}$ *dyadic-tolerance classifier read between their constant schedules is exactly the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *equality-and-tolerance row consumed by the* $\mathsf{Real}^{\uparrow}$ *seal classifier. Thus agreement of the two constant sources at the* $\mathsf{Real}^{\uparrow}$ *seal factors through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedules, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal, with no quotient stream equality, host real equality, selected limit, completion datum, or RealUp completeness row.*

Proof. Project both constant source packets using Definition 301.130. Their visible schedules are constant $\mathsf{StreamName}^{\uparrow}$ schedules, and every finite comparison therefore reads the same displayed endpoint rows d and e at the requested windows.

Apply Definition 301.131 to both constant sources. The resulting finite windows carry the endpoint rows, radius rows, endpoint-distance row, hsame transport rows, Cont routes, Pkg provenance, and local NameCert provenance needed by the RegSeqRat classifier ledger.

Now use the general seal classifier route: Definition 301.43 identifies the classifier seen by the RealUp seal with the RegSeqRat dyadic-tolerance classifier, and Definition 301.45 says that this classifier consumes only the displayed finite rows just listed. In the constant-source case those rows are precisely the DyadicRatCore equality-and-tolerance row for d and e.

Finally close with the handoff of Definition 301.132. That theorem supplies the four-face route from the two constant source descriptions to the RealUp seal, so the classifier reflection contains no additional coordinate for quotient stream equality, host real equality, selected limits, completion data, or RealUp completeness. □

Theorem 301.134 (RegSeqRat constant tail seal invariance)**.** *Let C be a* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packet over a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint d. For every certified unary tail shift of the* $\mathsf{StreamName}^{\uparrow}$ *schedule of C, the shifted source presents the same constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, the same endpoint-distance ledger at d, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier surface to the* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently any two certified unary tail shifts of the constant source have the same $\mathsf{Real}^{\uparrow}$ *seal source read after restriction to their common cofinal finite window. This invariance is only a statement about the displayed constant dyadic source rows, the RegSeqRat regularity and classifier surface, and the RealUp seal-facing finite packet. It exports no quotient stream equality, host real equality, selected limit, completion datum, or RealUp completeness row.*

Proof. Project the source by Definition 301.130. A certified unary tail shift changes only which schedule rows are requested; every requested row still returns the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint d and the same constant $\mathsf{StreamName}^{\uparrow}$ schedule entry.

Apply Definition 301.131. The endpoint rows on the shifted windows are equal to d, so the endpoint-distance ledger is the same equal-endpoint row and the dyadic-radius rows are read from the same constant packet.

Use Definition 301.18 to repack each shifted finite window as an accepted $\mathsf{RegSeqRat}^{\uparrow}$ window. Since the source is constant, the repacked window carries the same regularity ledger and the same dyadic-tolerance classifier surface as the unshifted constant source.

Now apply Definition 301.21 to two certified tail reads of the constant source. Their common cofinal finite window identifies the displayed schedule rows, endpoint rows, endpoint-distance ledger, classifier witness, hsame comparisons, Cont routes, Pkg provenance, and local NameCert provenance.

Finally close with Definition 301.132. The handoff says that the constant source reaches the $\mathsf{Real}^{\uparrow}$ seal only through the visible $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, so the tail comparison has no coordinate for quotient stream equality, host real equality, selected limits, completion data, or RealUp completeness. □

Theorem 301.135 (RegSeqRat constant tail readback)**.** *For a constant dyadic* $\mathsf{RegSeqRat}^{\uparrow}$ *source, every tail readback consumed by* $\mathsf{Real}^{\uparrow}$ *factors through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity ledger,* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. No quotient stream equality, selected limit, metric-completion row, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can alter the readback.*

Proof. Begin with the constant source of Definition 301.130. Every requested tail window reads the same constant $\mathsf{StreamName}^{\uparrow}$ schedule and the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, with only the requested radius rows varying inside the finite packet.

Use Definition 301.131. The regularity ledger for each tail window is rebuilt from the shared endpoint, the displayed radius rows, the equal-endpoint distance row, and the local transport, replay, package, and naming rows.

Apply Definition 301.132 and Definition 301.133. These results route the constant packet to the $\mathsf{Real}^{\uparrow}$ seal through the same StreamName, DyadicRatCore, RegSeqRat, and Real faces, and identify the seal-facing classifier with the finite dyadic-tolerance row already carried by the packet.

Finally use Definition 301.134. Tail shifts preserve the same constant schedule, endpoint-distance ledger, regularity surface, classifier surface, hsame transports, Cont routes, Pkg provenance, and local NameCert row. The excluded quotient, limit, completion, and completeness coordinates are absent from every cited surface, so none can change the tail readback consumed by $\mathsf{Real}^{\uparrow}$. □

Theorem 301.136 (RegSeqRat constant dyadic finite-request exhaustion)**.** *Let C be a* $\mathsf{RegSeqRat}^{\uparrow}$ *constant dyadic source packet over a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint d. Every finite* $\mathsf{Real}^{\uparrow}$*-facing request made of C factors through exactly the following displayed rows: the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row d, the requested dyadic-radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier ledger, the common cofinal finite window for the requested tails, the displayed* hsame *comparisons, the displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance.*

The request has no remaining coordinate for quotient stream equality, host real equality, selected limits, completion data, an embedding principle, or a $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the source packet using Definition 301.130. Each finite request first sees the same constant $\mathsf{StreamName}^{\uparrow}$ schedule and the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row d.

For every requested finite window, apply Definition 301.131. This supplies the requested radius rows, the equal-endpoint distance row, the regularity bound, the displayed hsame comparisons, the Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ provenance.

Next apply Definition 301.133. The classifier row read by the $\mathsf{Real}^{\uparrow}$ seal is therefore the finite dyadic-tolerance row already carried by the constant $\mathsf{RegSeqRat}^{\uparrow}$ packet.

Use Definition 301.134 for any requested tail reads. It replaces the tail requests by their common cofinal finite window while preserving the same schedule rows, endpoint row, regularity row, classifier row, transport rows, route rows, package row, and naming row.

Finally close with Definition 301.132. The handoff permits the constant packet to reach the $\mathsf{Real}^{\uparrow}$ seal only through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. Consequently the finite request is exhausted by the listed BEDC rows and cannot export quotient stream equality, host real equality, selected limits, completion data, embedding principles, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 301.137 (RegSeqRat constant dyadic finite-request union seal)**.** *Let C be a $\mathsf{RegSeqRat}^{\uparrow}$ constant dyadic source packet over a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint d. Suppose a finite family of $\mathsf{Real}^{\uparrow}$-facing finite requests to C has requested tail windows admitting a common cofinal finite window. Then the family factors through one displayed union packet whose rows are exactly the constant $\mathsf{StreamName}^{\uparrow}$ schedule, the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row d, the union of the requested dyadic-radius rows, the equal-endpoint distance rows, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier ledger, the terminal $\mathsf{Real}^{\uparrow}$ seal row, the componentwise hsame comparisons, the displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ provenance.*

The union packet exports no quotient stream equality, host real equality, selected limit, completion datum, countable-choice selector, embedding principle, or $\mathsf{Real}^{\uparrow}$ completeness row.

Proof. Apply Definition 301.136 to each request in the finite family. Every component request is already exhausted by the constant schedule, the endpoint row d, its requested radius rows, the regularity and classifier ledgers, the transport rows, the route rows, Pkg provenance, and local $\mathsf{NameCert}$ provenance.

Use Definition 301.134 to move all requested tail reads to the chosen common cofinal finite window. This preserves the constant $\mathsf{StreamName}^{\uparrow}$ schedule, the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, the equal-endpoint distance rows, the componentwise hsame comparisons, the displayed Cont routes, and the provenance rows while replacing the separate tail windows by the single finite window seen by the union packet.

Next apply Definition 301.133. The classifier rows for all component requests are identified with the same dyadic endpoint ledger, so their finite union only records the union of the requested radius rows and does not introduce a second classifier surface.

Finally close with Definition 301.132. The seal route passes through exactly the visible $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. Hence the displayed union packet has the listed BEDC rows and no coordinate for quotient stream equality, host real equality, selected limits, completion data, countable-choice selectors, embedding principles, or $\mathsf{Real}^{\uparrow}$ completeness. □

The $\mathsf{RegSeqRat}^{\uparrow}$ readback row is a consumed coordinate of the reciprocal-window route in chapter 703. That consumer may read the regular rational sequence only after the finite $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger have been displayed, so the reciprocal budget receives the row as part of the finite route $W \longmapsto D \longmapsto R$ rather than as a host inverse or quotient-real equality surface.

Theorem 301.138 (RegSeqRat reciprocal-budget visible route exhaustion)**.** *Every reciprocal-facing consumer of a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row inside an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet factors through the displayed finite route. The route first exposes the apartness budget* A *and modulus row* M*, then selects the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and reciprocal ledger* D*, then consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and finally reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* S *through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Consequently the reciprocal-facing consumer receives only the finite route

$$A, M, W, D, R, S, H, C, P, N.$$

It receives no host inverse, quotient stream equality, total reciprocal, trichotomy, selected limit, countable choice datum, or ambient completeness principle.

Proof. Start from the route paragraph above: it places the RegSeqRat readback behind the finite window and dyadic ledger as $W \longmapsto D \longmapsto R$.

Apply Definition 703.7. It expands that local RegSeqRat handoff to the CauchyInverseBudget carrier rows X, A, M, W, D, R, with the apartness budget and modulus exposed before the dyadic reciprocal ledger can be consumed.

Apply Definition 703.8. The downstream $\mathsf{Real}^{\uparrow}$ seal is then reached only after the same readback row R, so the seal consumer cannot bypass the displayed finite route.

Finally apply Definition 703.17. It adds the visible hsame, Cont, Pkg, and local NameCert rows to the same packet, exhausting the reciprocal-facing consumer surface and excluding host inverse, quotient stream equality, total reciprocal, trichotomy, selected limits, countable choice, and ambient completeness. □

Theorem 301.139 (RegSeqRat finite-request source section)**.** *Let* R *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let* S *be a public finite request to the accepted packet. Then* S *factors through one displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source section carried by a finite* BHist *packet. The visible rows of that section are exactly the following rows:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows selected by the request;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite dyadic-distance rows read at those schedule positions;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row and the regularity rows certifying the accepted packet;*
4. *the componentwise* hsame *rows tying the schedule, dyadic endpoint, radius, distance, classifier, and regularity rows to the same request packet;*
5. *the displayed* Cont *routes from the StreamName schedule through the dyadic rows and the RegSeqRat classifier surface;*
6. *the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*
7. *the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows for the request.*

No other row is part of the RegSeqRat source section.

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatFiniteRequestSourceSection`

Proof. Begin with the StreamName finite-observation section of Definition 126.527. It gives the displayed schedule rows named by the finite request and places them in one finite source packet.

Apply the RegSeqRat tail-budget section lock, Definition 126.530. The lock prevents the request from replacing the displayed schedule by an off-packet tail selector, so every later RegSeqRat row is read over the same finite window.

Next use the exact RegSeqRat handoff, Definition 301.69. It supplies the dyadic endpoint, radius, finite distance, classifier, and regularity rows over the displayed schedule.

Now apply Definition 301.74. The common ledger identifies the componentwise hsame rows, Cont routes, package provenance, and local naming rows as part of the same finite RegSeqRat packet. This gives one BHist source section and no second source coordinate. □

Theorem 301.140 (RegSeqRat Real request section nonescape)**.** *Let R be an accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, and let S be a public finite request to R that is consumed by a $\mathsf{Real}^{\uparrow}$ seal. Then the same displayed source section of Definition 301.139 is the entire source section seen by the seal. In particular the section cannot export quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, choice data, or a second $\mathsf{Real}^{\uparrow}$ seal coordinate.*

Proof. First factor the request through Definition 301.139. The factorization has only the StreamName schedule rows, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier and regularity rows, and the displayed transport, route, package, and local naming rows.

Apply the Real finite-request fusion theorem, Definition 38.698. It says that a Real seal which consumes the same request reads through one finite BHist packet with the same schedule, dyadic, classifier, regularity, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally compare the two packets through Definition 301.69 and Definition 301.74. These theorems keep the StreamName, DyadicRatCore, and RegSeqRat faces as the only source faces available to the seal-facing request. Since those faces carry no row for quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, choice data, or a second Real seal coordinate, the consumed section cannot export any such datum. □

Theorem 301.141 (RegSeqRat finite-request Real seal correspondence)**.** *Let R be an accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, and let S be a public finite request to R that is consumed by a $\mathsf{Real}^{\uparrow}$ seal. The finite request source section of Definition 301.139 determines the seal-visible source section of Definition 301.140: its rows restrict exactly to the StreamName schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and dyadic-distance rows, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows, the componentwise hsame rows, the displayed Cont routes, the Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows. Conversely any seal-visible section for the same request restricts to exactly those rows of the finite request source section. The correspondence carries no quotient stream equality, host sequence equality, selected limits, ambient completeness, metric-completeness data, choice data, or second seal coordinate.*

Proof. First build the finite source packet with Definition 301.139. This gives the unique displayed request section whose visible rows are the StreamName schedule rows, the dyadic endpoint, radius, and finite distance rows, the RegSeqRat classifier and regularity rows, and the displayed hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Next route the same public request to the Real consumer by Definition 38.698. The fusion theorem forces the seal to read through the finite BHist packet already named by the request, so the consumer has no independent source packet from which additional rows could be drawn.

The common visible rows are then fixed by Definition 301.69 and Definition 301.74. These theorems identify the schedule, dyadic, classifier, regularity, hsame, Cont, package, and naming

rows on both sides of the handoff, giving the restriction from the finite source section to the seal-visible section.

For the converse, start with a seal-visible section for the same request and apply Definition 301.140. Nonescape says that the seal-visible section is already the displayed RegSeqRat finite source section, so restricting it back along the request recovers exactly the same StreamName schedule, dyadic endpoint, radius, distance, classifier, regularity, hsame, Cont, Pkg, and local NameCert rows. The excluded rows are absent on both sides, so the correspondence introduces no quotient stream equality, host sequence equality, selected limit, ambient or metric completeness, choice datum, or second seal coordinate. □

Theorem 301.142 (RegSeqRat finite-request terminal budget factorization)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S be a public finite request to R that is consumed by a* $\mathsf{Real}^{\uparrow}$ *seal. Every terminal budget consumer of the finite seal request factors through exactly the rows of the finite request source section:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows selected by S;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite dyadic-distance rows read at those schedule positions;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row and regularity rows;*
4. *the componentwise* hsame *rows for the schedule, dyadic endpoint, radius, distance, classifier, regularity, and seal-facing request packet;*
5. *the displayed* Cont *routes from the schedule through the dyadic rows, classifier surface, regularity rows, and seal read;*
6. *the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
7. *the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows from the finite request source section.*

The factorization exports no quotient stream equality, host real equality, host sequence equality, selected limit, choice datum, ambient completeness principle, metric-completeness datum, or second seal coordinate.

Proof. Start with the finite request source section of Definition 301.139. It displays the StreamName schedule, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier and regularity rows, componentwise hsame, displayed Cont, package provenance, and local RegSeqRat naming rows as one finite BHist source section.

Next use the request nonescape theorem, Definition 301.140. It says that a $\mathsf{Real}^{\uparrow}$ seal consuming the same public finite request sees no source section beyond those displayed rows, so quotient streams, host equality rows, selected limits, completeness principles, choice data, and second seal coordinates remain outside the request surface.

The correspondence theorem Definition 301.141 identifies the seal-visible section with the finite request source section in both directions. Thus any consumer that reaches the seal-facing request must first restrict to the same StreamName, DyadicRatCore, RegSeqRat, transport, route, provenance, and local naming rows.

Finally apply the Real finite-request selector-budget lock, Definition 38.699, and the completion-consumer section exactness theorem, Definition 38.701. The lock prevents a terminal budget consumer from replacing the displayed finite section by a second selector or off-packet tail route, and

section exactness says that completion-facing acceptance is exactly factorization through that section. Repacking these two Real-side facts through the RegSeqRat correspondence gives the stated terminal factorization and no additional exported coordinate. □

Theorem 301.143 (RegSeqRat finite-request audit-threshold source lock)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S be a public finite request to R consumed by a* $\mathsf{Real}^{\uparrow}$ *seal. Suppose the finite request source section of Definition 301.139 is the same section through which the terminal budget factorization of Definition 301.142 consumes S.*

Then the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read on S is locked to exactly that finite source section. The read is the objectwise four-face threshold on*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

over the displayed support rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

It has no coordinate for quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus*, aggregate bridge rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the terminal budget factorization, Definition 301.142. It says that every terminal budget consumer of S factors through the displayed finite request source section, with the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces carried by the local BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply the four-face audit-threshold factorization, Definition 301.219. Since the terminal budget factorization has already fixed the finite source section, the current Phase $\mathsf{Real}^{\uparrow}$ and Completion threshold read can only be read through the same four local faces and the same displayed support rows.

The threshold remains objectwise. Use Definition 126.509 to read the StreamName-side threshold from the four local face records, and use Definition 291.107 to keep the DyadicRatCore, StreamName, RegSeqRat, and Real rows on the shared BHist, hsame, Cont, Pkg, and NameCert interface.

Finally invoke Definition 38.471. The Phase Real and Completion audit is pulled back from the local face records rather than from a cached route-level status. Therefore quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot move the read away from the finite request source section fixed above. □

Theorem 301.144 (RegSeqRat finite-request source-section intersection lock)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be public finite requests to R consumed by local* $\mathsf{Real}^{\uparrow}$ *seals. Suppose the displayed source sections of S and T agree on the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the componentwise* hsame *rows, the displayed* Cont *rows, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows.*

Then the common restriction of the two seal-facing requests is exactly one finite $\mathsf{RegSeqRat}^{\uparrow}$ *source section over those rows. No quotient stream, host equality, selected limit, choice datum, ambient or metric completeness principle, or second seal coordinate can distinguish S from T on that restriction.*

Proof. First apply the finite request source section theorem, Definition 301.139, to S and to T. The agreement hypotheses identify the displayed schedule, dyadic, classifier, regularity, transport, continuation, provenance, and local naming rows of the two finite source packets.

Next pass to the seal-visible side with Definition 301.141. The local $\mathsf{Real}^{\uparrow}$ seals see only the restriction of those finite source packets, so the same displayed rows give one seal-facing restriction for both requests.

Use the support composition theorem, Definition 301.149, to compose the two agreed support faces through their common displayed rows. The composed support face still contains exactly the StreamName, DyadicRatCore, RegSeqRat, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows named in the statement.

Finally apply the terminal budget factorization theorem, Definition 301.142, to each seal-facing request. Every terminal consumer factors through the finite source section already identified above. Since that section carries no quotient stream, host equality, selected limit, choice datum, completeness principle, or second seal coordinate, no such external datum can separate the two requests on the common restriction. □

Theorem 301.145 (RegSeqRat finite-request intersection chain stability)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $S_0, \ldots, S_n$ be a finite nonempty chain of accepted public finite requests to R consumed by local* $\mathsf{Real}^{\uparrow}$ *seals. Suppose each adjacent pair (S_i, S_{i+1}) agrees on the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the componentwise* hsame *rows, the displayed* Cont *rows, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Suppose also that every adjacent pair agrees on its support subpacket and on the classifier-visible finite request section over those rows.*

Then every pairwise common restriction determined by the chain has the same Real-seal-facing finite $\mathsf{RegSeqRat}^{\uparrow}$ *source section. The direct restriction from S_0 to S_n and every iterated adjacent restriction determine that same section. The statement does not identify quotient stream equality, host real equality, selected limits, choice data, ambient or metric completeness, duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates, aggregate formalstatus rows, or bridge rows.*

Proof. First lock each adjacent intersection. Apply Definition 301.144 to every adjacent pair (S_i, S_{i+1}). The hypotheses give exactly the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows required by that lock, so each adjacent common restriction has one Real-seal-facing finite source section.

Next compose the support faces along the chain. Use Definition 301.149 for each two-leg composite and then Definition 301.150 for the finite nonempty chain. These results identify all parenthesized adjacent support-face composites with the direct S_0 to S_n support face, while keeping only the displayed StreamName, DyadicRatCore, RegSeqRat, hsame, Cont, Pkg, and local naming rows.

Finally compare the source sections read from those support faces. The section-refinement composition theorem, Definition 301.244, says that the direct restriction and the iterated adjacent restriction have the same finite request section and the same local $\mathsf{Real}^{\uparrow}$ seal on the displayed rows. Substituting the adjacent intersection locks into this comparison gives one Real-seal-facing finite $\mathsf{RegSeqRat}^{\uparrow}$ source section for every pairwise common restriction determined by the chain.

The cited lock, support-composition, chain-induction, and section-refinement routes are all finite BEDC routes over displayed request rows. None supplies a quotient stream equality, host real equality, selected limit, choice datum, completeness principle, duplicate Real seal coordinate, aggregate formalstatus row, or bridge row. Hence the chain stability result does not claim any of those objects. □

Theorem 301.146 (RegSeqRat finite-request classifier alignment)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be public finite requests to P consumed by the public* $\mathsf{Real}^{\uparrow}$ *seal route. The following data determine one another:*

1. *agreement of the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows of* S *and* T*;*
2. *agreement of the classifier-visible finite request section of* S *and* T *inside the source section of Definition 301.139.*

Thus the rows consumed by the Real seal route are exactly the classifier rows visible in the finite request section. The alignment has no coordinate for quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness data, or second seal coordinates.

Proof. For the forward direction, start with the finite request source section of Definition 301.139. Agreement of the displayed rows in the first item names the same StreamName schedule, dyadic endpoint, radius, distance, RegSeqRat classifier, regularity, transport, continuation, package, and local naming rows in that finite BHist section.

Next apply Definition 301.69. It keeps the schedule rows, dyadic rows, RegSeqRat classifier surface, and terminal $\mathsf{Real}^{\uparrow}$ seal row on one ordered handoff rather than allowing a separate seal-facing source packet. Hence the agreed displayed rows are precisely the classifier-visible finite request section.

For the reverse direction, suppose the classifier-visible finite request sections agree. Use Definition 301.160 to identify that agreement with agreement of the shared selector surface on the same schedule, dyadic, classifier, regularity, terminal seal, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally apply Definition 301.159. The terminal pullback reads the same displayed selector surface as the same $\mathsf{Real}^{\uparrow}$ seal route, so the first list of rows agrees component by component. Since the source section and selector surface contain no rows for quotient stream equality, host real equality, selected limits, choice data, ambient or metric completeness, or a second seal coordinate, the equivalence cannot transport any of those objects. □

Theorem 301.147 (RegSeqRat finite-request support transport)**.** *Let* P *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let* S *and* T *be public finite requests to* P *consumed by the public* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose the classifier-visible finite request sections of* S *and* T *agree. Then the support rows transported through that section are the same support packet: the componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows of* S *and* T *agree component by component. Conversely, equality of that transported support packet recovers agreement of the support part of the classifier-visible finite request section. This support transport has no coordinate for quotient stream equality, host sequence equality, selected limits, choice data, ambient completeness, metric-completeness rows, or duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates.*

Proof. Begin with the source section theorem, Definition 301.139. It displays the support part of each finite request as the componentwise hsame rows, displayed Cont rows, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows in one finite BHist source packet.

For the forward direction, apply the classifier alignment theorem, Definition 301.146. Agreement of the classifier-visible sections forces agreement on every row of the displayed source section that the classifier can see. The four support families listed in the statement are part of that displayed section, so transporting them through the agreed section yields one support packet.

Next use the StreamName-Dyadic-Real handoff exactness theorem, Definition 301.69. It keeps the schedule, dyadic, classifier, regularity, and terminal seal rows on the same finite handoff, so the support packet cannot detach from the section or acquire a second seal-facing coordinate.

Now apply the Real finite-request fusion theorem, Definition 38.698. The $\mathsf{Real}^{\uparrow}$ consumer reads through the same finite BHist packet, with the same hsame, Cont, Pkg, and local naming rows. Thus the support packet transported through the RegSeqRat section is the support packet consumed by the Real seal route.

For the converse, assume the transported support packets are equal. Reading them back through the source section identifies exactly the support sublist of the classifier-visible finite request section. The classifier alignment theorem then recovers agreement of that support part and no larger object, because the cited source, handoff, and fusion routes contain no rows for quotient stream equality, host sequence equality, selected limits, choice data, ambient or metric completeness, or duplicate Real seal coordinates. □

Theorem 301.148 (RegSeqRat Real seal support face uniqueness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be public finite requests to P consumed by the public* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose the classifier-visible finite request sections of S and T agree on the support subpacket. Then the Real-seal-facing support face is unique: the displayed* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows form the same support face for S and T. Every accepted* $\mathsf{Real}^{\uparrow}$ *seal consumer that exposes those rows factors through that face. Conversely, equality of the Real-seal support face recovers equality of the support part of the classifier-visible* $\mathsf{RegSeqRat}^{\uparrow}$ *finite request section. The uniqueness has no coordinate for quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, or duplicate* $\mathsf{Real}^{\uparrow}$ *seal coordinates.*

Proof. Start with the support transport theorem, Definition 301.147. It transports agreement of the support subpacket through the componentwise hsame rows, displayed Cont rows, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows, so the two requests present one support packet on the RegSeqRat side.

Next apply the classifier alignment theorem, Definition 301.146. The classifier-visible section sees exactly the displayed support rows named above. Thus equality of the support subpacket is equality of the support part of the finite request section, not equality of any hidden host or quotient object.

Now use the Real finite-request fusion theorem, Definition 38.698. The public $\mathsf{Real}^{\uparrow}$ consumer reads the same finite BHist packet and the same displayed support rows. Hence any accepted seal consumer exposing hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ data must factor through that single Real-seal-facing support face.

For the converse, assume the Real-seal support faces agree. Reading the agreed face back through the fusion packet and the classifier-visible section returns the same support part of the RegSeqRat finite request section. The source, transport, alignment, and fusion routes expose no rows for quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, or duplicate Real seal coordinates, so no external coordinate can distinguish two support faces with the same displayed rows. □

Theorem 301.149 (RegSeqRat finite-request support face composition)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S_0, S_1, and S_2 be accepted public finite requests to P consumed along the public* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose the classifier-visible finite request sections of (S_0, S_1) and (S_1, S_2) agree on their support subpackets. Suppose also that the middle support face exposes the same* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows on both legs.*

Then the composed route from S_0 to S_2 has one Real-seal-facing support face, and that face contains exactly the displayed $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows exposed by the middle face. It exports no quotient stream equality, host real equality, selected limit, countable-choice data, metric-completeness rows, duplicate Real seal, aggregate formalstatus row, or bridge row.*

Proof. First identify the two classifier-visible sections. Apply Definition 301.146 to the pair (S_0, S_1) and again to the pair (S_1, S_2). The hypotheses say that the two legs display the same support subpacket at S_1, so the classifier sees one finite support surface through the middle request.

Next transport the displayed support rows along each leg. By Definition 301.147, the shared componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows determine the same transported support packet on the RegSeqRat side for (S_0, S_1) and for (S_1, S_2). The listed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ classifier rows keep that support packet attached to the same finite request face.

Now pass to the Real-seal-facing side. Use Definition 301.148 on the transported support packet to pin the unique Real-facing support face exposed by each leg. Since both legs expose the same middle face, uniqueness identifies their Real-facing support faces as one displayed face.

Finally compare the direct and two-step finite request routes. The section refinement composition theorem, Definition 301.244, says that the one-step route from S_0 to S_2 and the route through S_1 have the same finite request section and local Real seal on the displayed rows. Therefore the composed route has exactly the support face already pinned above.

All cited routes are finite classifier, support-transport, uniqueness, and section-refinement routes over the displayed BEDC rows. None carries quotient stream equality, host real equality, selected-limit data, countable-choice data, metric-completeness rows, duplicate Real seal data, aggregate formalstatus rows, or bridge rows, so the composed support face exports none of those objects. □

Theorem 301.150 (RegSeqRat finite-request support face chain induction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $S_0, \ldots, S_n$ be a finite nonempty chain of accepted public finite requests to P consumed along the public* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose each adjacent pair (S_i, S_{i+1}) has agreeing classifier-visible finite request sections on the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Suppose also that every adjacent pair agrees on its support subpacket.*

Then every parenthesized adjacent support-face composite along the chain and the direct comparison from S_0 to S_n determine the same terminal $\mathsf{Real}^{\uparrow}$ *seal-facing support face. That face contains exactly the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows carried by the adjacent finite request sections, and it exports no quotient stream equality, host real equality, selected limit, choice data, ambient completeness, metric-completeness rows, duplicate* $\mathsf{Real}^{\uparrow}$ *seal, aggregate formalstatus row, or bridge row.*

Proof. The base case is the one-object chain. The direct comparison from S_0 to S_0 is reflexive on the displayed finite request section, so the terminal support face is the same Real-seal-facing face read from that section. No composition data are introduced.

For a one-step chain, apply Definition 301.147 to the adjacent pair and then apply Definition 301.148. The agreed classifier-visible section and the agreed support subpacket pin one Real-facing support face for the two requests.

For the induction step, assume the prefix from S_0 to S_k has a unique terminal support face independent of parenthesization. Compose that prefix face with the last adjacent face from S_k to S_{k+1} using Definition 301.149. The theorem keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows on the same Real-seal-facing support face, so the result is independent of whether the last leg is attached before or after the prefix composite.

It remains to compare the iterated composite with the direct first-to-last section. The section-refinement composition route used in Definition 301.149 identifies the direct S_0 to S_{k+1} classifier-visible section with the composed finite section on the displayed rows. Applying support-face uniqueness once more gives the same terminal $\mathsf{Real}^{\uparrow}$ seal-facing support face.

Every step uses only classifier alignment, support transport, support-face uniqueness, and finite section composition over the listed BEDC rows. These routes contain no rows for quotient streams, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, duplicate Real seals, aggregate formalstatus rows, or bridge rows, so the chain induction cannot export any of those objects. □

Theorem 301.151 (RegSeqRat Real seal support face classifier readback)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be public finite requests to P consumed by the public* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose the terminal Real-seal-facing support faces of S and T are equal, and suppose the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows agree component by component. Then the classifier-visible finite request sections of S and T are equal.*

The readback has no coordinate for quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, duplicate $\mathsf{Real}^{\uparrow}$ *seal coordinates, aggregate formalstatus rows, or bridge rows.*

Proof. First read the terminal support face back through Definition 301.148. Equality of the Real-seal-facing support faces recovers equality of the support part of the classifier-visible $\mathsf{RegSeqRat}^{\uparrow}$ finite request section.

Next identify that recovered support part with the finite request support rows. Apply Definition 301.147: the componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows are exactly the transported support packet seen by the classifier-visible section.

Now combine the recovered support rows with the displayed non-support rows. The hypotheses already identify the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and dyadic-distance rows, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows. Hence the four visible faces of the finite request section agree.

Finally apply Definition 301.146. Agreement of the displayed four-face data recovers equality of the full classifier-visible finite request sections. The nonescape argument is the same one used in Definition 301.150: the classifier, support-transport, and Real-seal uniqueness routes contain only the displayed BEDC rows. They contain no rows for quotient stream equality, host real equality, selected limits, choice data, ambient completeness, metric-completeness rows, duplicate Real seal coordinates, aggregate formalstatus rows, or bridge rows, so the readback cannot export any of those objects. □

Theorem 301.152 (RegSeqRat StreamName Real scope triad)**.** *For every accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier P, every public* $\mathsf{Real}^{\uparrow}$ *seal read of P is exposed through the three visible scope gates*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule selecting the requested windows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and tail-budget rows read from those windows;*

3. the $\mathsf{Real}^{\uparrow}$ *seal row that consumes the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy classifier surface.*

The triad preserves the displayed hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It contains no quotient stream equality, host real equality, selected limit, completion object, ambient completeness datum, or second seal coordinate.*

Proof. Start with the exact handoff of Definition 301.69. It routes public seal reads through StreamName schedules, DyadicRatCore ledgers, RegSeqRat classifier rows, and the public RealUp seal.

Next apply the source lock of Definition 301.110. The StreamName schedule and the dyadic endpoint, radius, and distance rows are one finite source surface attached to the accepted carrier P.

Finally use Definition 301.111. It identifies the downstream consumer as the same finite regular-Cauchy route with its displayed transports, continuations, package provenance, and local name row. Since those cited surfaces expose only the listed rows, the excluded quotient, host, completion, and second-seal data cannot enter the triad. □

Theorem 301.153 (RegSeqRat dyadic tail-budget totality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, and let a tail-budget consumer request any completion-facing comparison of P. The comparison is total over the displayed dyadic budget rows: it reads a* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the requested dyadic radius rows, the dyadic distance rows, the regularity bound rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance. No comparison step has a separate source outside those rows.*

Consequently every tail comparison that can feed the $\mathsf{Real}^{\uparrow}$ *seal is a read of the same finite dyadic budget surface, not a quotient-stream comparison, host real comparison, selected-limit comparison, completion comparison, or ambient completeness comparison.*

Proof. Project the common dyadic tail ledger from Definition 301.74. It lists the schedule, endpoint, radius, distance, classifier, regularity, transport, continuation, package, and naming rows available to tail reads.

Use Definition 301.109 to order a completion-facing request through that ledger before any RealUp-facing surface is visible. The request therefore has no earlier coordinate from which a completion or host comparison could be read.

Now apply Definition 301.114. The terminal surface preserves exactly the same finite dyadic budget rows all the way to the RealUp seal. Thus the comparison is total over the displayed budget surface and nowhere else. □

Theorem 301.154 (RegSeqRat consumer seal unblock surface)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier P, the downstream* $\mathsf{Real}^{\uparrow}$ *handoff surface is unblocked precisely by the two rows supplied by Definition 301.152 and Definition 301.153: the StreamName–RegSeqRat–RealUp scope triad and the total dyadic tail-budget surface. Once these rows are present, every public RealUp consumer reads the finite regular-Cauchy packet through the visible* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces, preserving* hsame, Cont, Pkg, *and* NameCert *provenance.*

Moreover, two downstream $\mathsf{Real}^{\uparrow}$ *handoff surfaces exposing the same* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and tail-budget rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* hsame, *displayed* Cont, Pkg *provenance, and local* NameCert *provenance determine the same public consumer seal surface.*

The unblock surface is only this downstream handoff. It does not provide a quotient stream, host real equality, selected limit, metric-completion object, ambient completeness theorem, aggregate

formalstatus, aggregate bridge status, route-level leantarget, route-level completeness coordinate, or bridge-checked RealUp consumer.

Proof. First assemble the scope row using Definition 301.152. This places every public seal read behind the StreamName schedule, the dyadic observation budget, and the RealUp seal.

Next assemble the budget row using Definition 301.153. It says that every tail-budget comparison available to a completion-facing consumer is already a read of the displayed dyadic ledger.

Finally combine the public consumer factorization of Definition 301.111 with the terminal seal surface of Definition 301.114. These facts package the scope row and the budget row as the downstream RealUp handoff.

The determinacy of that package is the finite-row determinacy of Definition 301.164: the dyadic ledger, the RegSeqRat selector presentation, and the terminal Real seal read are three presentations of the same displayed surface. Thus two handoff surfaces with the same schedule, dyadic, classifier, readback, seal, transport, continuation, package, and local naming rows determine the same public consumer seal surface.

Finally apply Definition 38.837. It keeps formalstatus, leantarget, bridgestatus, and completeness reports attached to their displayed faces rather than turning their conjunction into a route-level coordinate. Since the package exposes no quotient, host, selected-limit, completion, completeness, aggregate-status, route-target, or bridge-checked coordinate, the consumer is unblocked exactly at the stated finite surface. □

Theorem 301.155 (RegSeqRat consumer seal defect localization)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, and suppose a downstream* $\mathsf{Real}^{\uparrow}$ *handoff from P does not read through the consumer seal surface of Definition 301.154. Then the defect is visible on at least one displayed row of that surface: the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, or tail-budget rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier or readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, or the support rows* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

No quotient stream, host real equality, selected limit, metric-completeness object, aggregate formalstatus row, bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness datum repairs such a failed read. Any repair must restore one of the displayed rows above.*

Proof. Apply Definition 301.152. It restricts every public RealUp read of an accepted RegSeqRat carrier to the StreamName schedule, the dyadic observation rows, and the terminal RealUp seal, with the displayed transport, continuation, package, and name rows preserved.

Apply Definition 301.153. It makes each completion-facing comparison that can feed the terminal seal a read of the finite dyadic budget surface, including endpoint, radius, distance, regularity, classifier, readback, transport, continuation, package, and local naming data.

Now use Definition 301.154. The theorem says that the positive consumer handoff is exactly the finite package formed from those rows. Therefore a failed downstream read of that handoff must remove, alter, or fail to present at least one row in the displayed package.

Finally apply Definition 38.837. It keeps formalstatus, bridge status, leantarget reports, and completeness reports on their own faces rather than letting them act as hidden RealUp repair coordinates. The excluded quotient, host-real, selected-limit, completion, aggregate-status, bridge, and completeness objects therefore cannot repair the failed read; the obstruction is localized on the stated finite rows. □

Theorem 301.156 (RegSeqRat scope unblock)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *face of the current L10 route is unblocked exactly by the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance,*

and tail-budget rows, the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

When those rows are present, the RegSeqRat consumer-facing route is public; when any one is missing, no quotient stream, host real equality, selected limit, metric-completion object, aggregate formalstatus row, bridge row, or $\mathsf{Real}^{\uparrow}$ *completeness datum repairs the route.*

Proof. First apply Definition 301.152. It places every public RealUp-facing read behind the StreamName schedule, the dyadic endpoint, radius, distance, and tail-budget rows, the RegSeqRat classifier surface, and the terminal RealUp seal.

Next apply Definition 301.153. This makes the completion-facing comparison total over the displayed dyadic budget and support rows, so the RegSeqRat face has no earlier source outside that finite ledger.

For sufficiency, use Definition 301.154. The theorem packages the scope triad and the dyadic budget totality as the consumer-facing unblock surface, including classifier, readback, seal, transport, continuation, package, and local naming rows.

For necessity, apply Definition 301.155. If the route does not read through the unblock surface, the defect is visible on one of the displayed schedule, dyadic, classifier, readback, seal, or support rows. The same theorem and Definition 38.837 exclude quotient streams, host equality, selected limits, metric completion, aggregate status, bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness as repair coordinates. □

Definition 301.157 (RegSeqRat shared-window selector surface)**.** *For an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet* P*, the shared-window selector surface is the finite* BHist *surface consisting of the following displayed rows:*

1. $\mathsf{StreamName}^{\uparrow}$ *schedule rows selecting the public finite windows read by the request;*
2. $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows attached to those windows;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and tail-budget rows for the same selected windows;*
4. *componentwise* hsame *rows, displayed* Cont *replay rows,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *row;*
5. *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consuming the displayed regular-rational source surface.*

The surface contains only the rows visible in the $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *handoff. It has no quotient stream equality, host real equality, selected limit, completion point, ambient completeness datum, hidden selector, or second seal coordinate.*

Theorem 301.158 (RegSeqRat shared-window selector determinacy)**.** *Let* P *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let* S *and* T *be accepted finite requests whose displayed selector surfaces are the shared-window surface of Definition 301.157. If* S *and* T *display the same* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and tail-budget rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, local* $\mathsf{NameCert}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row, then* S *and* T *determine the same RegSeqRat tail-window route up to the displayed* hsame *transport and* Cont *replay.*

The determinacy is internal to the shared-window surface. It does not identify quotient streams, host real values, selected limits, completion points, ambient completeness data, hidden selector choices, or any seal outside the displayed $\mathsf{Real}^{\uparrow}$ *row.*

Proof. First use the scope triad of Definition 301.152. It places every public $\mathsf{Real}^{\uparrow}$ seal read behind the visible StreamName schedule, the dyadic endpoint and radius rows, and the RegSeqRat classifier surface before the seal row is consumed.

Next apply the finite request source section of Definition 301.139. The two finite requests have the same displayed source rows, so their schedule, dyadic ledger, classifier, regularity, transport, continuation, package, and naming coordinates are one public source section.

Now use the tail-selector RealUp seal exhaustion theorem, Definition 301.123. It prevents either request from using a quotient stream, hidden selector, selected limit, completion point, or second seal to distinguish a separate tail-window route.

Finally cite the StreamName-Dyadic-Real handoff exactness of Definition 301.69. Transport through hsame and replay through Cont preserve the same handoff rows, so the two accepted requests reach the same RegSeqRat tail-window route and then the same displayed $\mathsf{Real}^{\uparrow}$ seal row. □

Theorem 301.159 (RegSeqRat shared-window terminal pullback equivalence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be accepted finite requests to P whose displayed selector surfaces are the shared-window surface of Definition 301.157. The following data determine one another:*

1. *equality of the shared-window selector surface displayed by S and T;*
2. *equality of the terminal* $\mathsf{Real}^{\uparrow}$ *seal reads obtained from S and T after both requests factor through the ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Equivalently, every terminal-budget consumer accepted by S is accepted by T with the same visible dyadic, stream, regular-rational, terminal seal, BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows, and conversely. The correspondence exposes no quotient stream equality, host real equality, selected limit, hidden selector, completion carrier, ambient completeness datum, or second terminal seal.*

Proof. For the forward direction, apply Definition 301.158. Agreement on the shared-window selector surface identifies the finite StreamName schedule rows, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier and tail-budget rows, componentwise transports, continuations, package provenance, local naming row, and displayed terminal seal coordinate.

Next use Definition 301.142 and Definition 38.713. The two accepted finite requests therefore factor through the same ordered four-face packet from $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, with the same support rows. The seal uniqueness theorem, Definition 38.714, then identifies their visible terminal $\mathsf{Real}^{\uparrow}$ seal reads.

For the reverse direction, suppose the terminal $\mathsf{Real}^{\uparrow}$ seal reads agree after both requests have been read through the four-face packet. The exhaustion theorem places every visible terminal-budget row of each request on that packet, and seal uniqueness prevents a second terminal read from separating requests with the same displayed faces and support rows.

It remains to recover the selector surface. The finite-request factorization returns exactly the StreamName, DyadicRatCore, RegSeqRat, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows listed in Definition 301.157. Hence equality of the terminal seal reads on the factored packet is equality of the shared-window selector surface.

For the consumer form, take any terminal-budget consumer accepted by S. By terminal-budget exhaustion it reads only the four displayed faces and support rows, so the already established

surface equality repacks the same visible rows as an accepted consumer for T; the converse is symmetric. Finally, Definition 301.167 excludes quotient streams, host real equality, selected limits, hidden selectors, completion carriers, ambient completeness data, and second seals from both sides of the correspondence. □

Theorem 301.160 (RegSeqRat tail-selector section pullback)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be accepted finite requests to P whose displayed selector surfaces are the shared-window surface of Definition 301.157. Agreement of the shared-window selector surface displayed by S and T is equivalent to agreement of the finite-request source section of Definition 301.139, restricted exactly to the* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and tail-budget rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows.*

The pullback has no coordinates for quotient stream equality, host real equality, selected limits, completion points, ambient completeness data, choice data, or second seal coordinates. Thus the selector surface and the finite-request source section name the same finite request data over the ordered handoff

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Proof. For the forward direction, begin with Definition 301.158. Agreement of the shared-window selector surface identifies the visible schedule, dyadic ledger, classifier, regularity, tail-budget, transport, continuation, package, naming, and terminal seal rows.

Apply the finite-request source section theorem, Definition 301.139. It records exactly those visible rows as the finite source section of each accepted request. The StreamName-Dyadic-Real handoff exactness theorem, Definition 301.69, keeps the schedule rows, dyadic endpoint and radius rows, regular-rational classifier rows, and terminal seal row on the same finite handoff, so the two source sections agree on precisely the displayed rows named in the statement.

For the reverse direction, suppose the finite-request source sections agree on those rows. Use Definition 301.142 to factor both accepted requests through the common terminal-budget section. Then Definition 38.713 places every visible Real-facing consumer on that same four-face packet, and Definition 38.714 identifies the terminal seal row.

It remains only to rule out hidden coordinates. The selector surface definition contains no row for quotient streams, host equality, selected limits, completion points, ambient completeness, choice data, or a second seal coordinate. Applying Definition 301.167 removes those candidates from the Real-facing boundary, so agreement of the finite-request source section repacks exactly as agreement of the shared-window selector surface. □

Theorem 301.161 (RegSeqRat shared-window reindex pullback coherence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be accepted finite requests to P whose displayed selector surfaces are the shared-window surface of Definition 301.157. Suppose R_S and R_T are certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes over the displayed selector surfaces of S and T. Assume that the reindex routes preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and tail-budget rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows.*

Then the RegSeqRat tail-selector section pullback of Definition 301.160 before applying R_S and R_T is equivalent to the tail-selector section pullback after applying them. Moreover, both terminal

$\mathsf{Real}^{\uparrow}$ *seal reads factor through the same ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The coherence is internal to the displayed finite surface. It introduces no quotient stream equality, host real equality, selected limit, completion point, choice datum, ambient completeness row, or second seal coordinate.

Proof. First compare the StreamName windows before and after reindexing. Apply Definition 126.639 to the certified routes R_S and R_T. The hypotheses name the same schedule, endpoint, BHist, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows, so the public finite observation section is unchanged by the reindex routes.

Next place that preserved section on the terminal side. By Definition 126.642, the reindexed reads expose the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal. The four-face exit naturality theorem, Definition 126.647, keeps this terminal read natural in the certified reindex route rather than in any hidden host coordinate.

Now return to the RegSeqRat selector surface. Apply Definition 301.160 before reindexing and again after reindexing. Since the preserved rows are exactly the StreamName schedule rows, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier, regularity, and tail-budget rows, terminal seal row, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows, the two pullback readings are the same finite source-section comparison.

Finally use Definition 301.159 to identify the terminal seal read with that same selector surface, and use Definition 301.167 to exclude extra coordinates. Thus the pre-reindex and post-reindex pullbacks transport through the same four-face route and cannot be separated by quotient streams, host real equality, selected limits, completion points, choice data, ambient completeness rows, or second seals. □

Theorem 301.162 (RegSeqRat shared-window reindex composition law)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S and T be accepted finite requests to P whose displayed selector surfaces are the shared-window surface of Definition 301.157. Suppose R_1 and R_2 are composable certified public* $\mathsf{StreamName}^{\uparrow}$ *finite-window reindex routes on those displayed surfaces, and suppose their certified composite route is $R_2 \circ R_1$. Assume that R_1, R_2, and $R_2 \circ R_1$ preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and tail-budget rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows.*

Then applying R_1 and then R_2 gives the same RegSeqRat tail-selector section pullback, and the same terminal $\mathsf{Real}^{\uparrow}$ *seal read, as applying the certified composite route $R_2 \circ R_1$ once. The equality is only equality of the displayed finite selector surface over the ordered handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It does not identify quotient stream equality, host real equality, selected limits, completion points, choice data, ambient completeness rows, or second seal coordinates.

Proof. First work inside the public StreamName window. The reindex composition law of Definition 126.20 identifies the two-step certified route R_1 followed by R_2 with the single certified route $R_2 \circ R_1$ on the finite StreamName observation section.

Next transport the displayed rows through this identification. Apply Definition 126.639 to the two certified presentations, and use Definition 126.647 to keep the StreamName window, DyadicRatCore ledger, RegSeqRat readback, and terminal Real seal on the same four-face exit route.

Now return to the RegSeqRat selector section. The hypotheses say that each route preserves exactly the rows listed in Definition 301.157: schedule, dyadic endpoint, radius, distance, classifier, regularity, tail budget, terminal seal, hsame, Cont, Pkg, and local NameCert. Applying Definition 301.161 to the first route, then to the second route, and then to the composite route gives the same tail-selector section pullback on those rows.

Finally identify the terminal seal boundary. By Definition 301.159, equality of the displayed selector pullback is equality of the terminal $\mathsf{Real}^{\uparrow}$ seal read through the same ordered handoff. The nonescape theorem, Definition 301.167, removes quotient streams, host real equality, selected limits, completion points, choice data, ambient completeness rows, and second seal coordinates from the comparison, so no hidden coordinate can distinguish the two-step route from its certified composite. □

Theorem 301.163 (RegSeqRat shared-window triple-overlap cocycle)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S, T, and U be accepted finite requests to P. Suppose every pair among S, T, and U displays the shared-window selector surface of Definition 301.157. Pairwise agreement on the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and dyadic-distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, regularity, and tail-budget rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* NameCert *rows determines one triple-overlap selector surface over the ordered handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Conversely, one such triple-overlap selector surface restricts to the three pairwise pullback equivalences between (S,T), (T,U), and (S,U).

The triple-overlap surface contains no quotient stream equality, host real equality, selected limit, completion point, ambient completeness datum, choice data, or second seal coordinate. It is exactly the common finite BHist *surface on the displayed StreamName, DyadicRatCore, RegSeqRat, terminal Real,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Proof. Start with the three pairwise selector surfaces. On each pair, apply Definition 301.158 to identify the displayed schedule, dyadic ledger, classifier, regularity, tail-budget, transport, continuation, package, naming, and terminal seal rows.

Next pass from pairwise selector agreement to finite-request source sections using Definition 301.160. The three source-section comparisons have the same visible rows, and Definition 126.549 identifies their StreamName–RegSeqRat source restrictions as one source surface on the common triple overlap.

Now attach the terminal Real row. The terminal pullback equivalence of Definition 301.159 and the Real finite-window cocycle Definition 38.191 make the three terminal seal restrictions the restrictions of one $\mathsf{Real}^{\uparrow}$ boundary over the same four-face handoff.

This gives the forward direction: the common source surface and the common terminal boundary are precisely the shared-window selector rows listed in Definition 301.157, so they determine one triple-overlap selector surface.

For the reverse direction, restrict a single triple-overlap selector surface to the three displayed pairwise faces. Each restriction has the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows, so Definition 301.160 returns the finite-request source-section equivalence and Definition 301.159 returns the terminal pullback equivalence for each pair.

Finally apply Definition 301.167. Since the shared-window surface has no row for quotient streams, host real equality, selected limits, completion points, ambient completeness data, choice

data, or second seals, none of those objects can distinguish the triple-overlap surface from its three pairwise restrictions. □

Theorem 301.164 (RegSeqRat dyadic triangle selector equivalence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S, T, and U be accepted finite requests to P whose pairwise faces display the shared-window selector surface of Definition 301.157. The following presentations determine one another on exactly the displayed finite rows:*

1. *pairwise agreement of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *common scale-window triangle ledger on the signed mantissa row, refined positive-unary exponent row, denominator ledger, common-exponent comparison row, and dyadic distance/radius rows;*
2. *agreement of the* $\mathsf{RegSeqRat}^{\uparrow}$ *shared-window triple-overlap selector surface on the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier/readback/tail-budget rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows;*
3. *equality of the terminal* $\mathsf{Real}^{\uparrow}$ *seal read after the three requests are read through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus the dyadic triangle ledger and the RegSeqRat selector surface are the same BEDC-visible finite data at the terminal $\mathsf{Real}^{\uparrow}$ *boundary. The equivalence adds no quotient rational, quotient stream, host real equality, selected limit, completion point, choice data, ambient completeness datum, or second seal.*

Proof. Begin with the dyadic presentation. Apply Definition 291.83 to pass from pairwise agreement of the common scale-window triangle ledger to the terminal four-face pullback over the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Next identify that four-face pullback with the RegSeqRat selector surface. The terminal pullback equivalence of Definition 301.159 says that equality of the shared-window selector surface and equality of the terminal $\mathsf{Real}^{\uparrow}$ seal read through the same four faces determine one another on the listed StreamName, dyadic, regular-rational, transport, continuation, package, naming, and seal rows.

For three requests, use Definition 301.163. The three pairwise selector faces restrict from one triple-overlap selector surface, and conversely that surface restricts back to the three pairwise pullbacks. Hence the dyadic triangle ledger, the RegSeqRat triple-overlap surface, and the terminal seal read are three presentations of the same finite boundary data.

Finally apply Definition 301.167. Since the selector surface has no row for quotient streams, host real equality, selected limits, completion points, choice data, ambient completeness rows, or second seals, the correspondence cannot be enlarged by any of those objects. □

Theorem 301.165 (RegSeqRat finite four-face current-phase exit coverage)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let C be a finite current-phase RegSeqRat consumer reading P through the Phase Real and Completion exit surface. The consumer is accepted by that exit surface if and only if it displays the same four visible faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

over one finite BHist *surface, with the same componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows as the shared-window selector surface of Definition 301.157.*

Equivalently, the current-phase exit test is exactly the finite four-face coverage test that reads the dyadic common-scale ledger, the public StreamName common refinement, the RegSeqRat selector/readback surface, and the terminal Real finite-cover face as one BEDC-visible boundary. It does not accept or compare quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, start with an accepted current-phase consumer. Apply Definition 126.240 to read its StreamName schedule as a public finite common refinement, with the displayed BHist, hsame, Cont, Pkg, and local NameCert rows preserved.

Next apply Definition 301.164. This identifies the dyadic common-scale triangle ledger, the RegSeqRat shared-window selector surface, and the terminal Real seal read as three presentations of the same finite four-face boundary. Finally use Definition 38.771 to paste the terminal $\mathsf{Real}^{\uparrow}$ finite-cover face onto that boundary without adding any completion datum.

For the reverse direction, assume the consumer displays exactly those four faces over the same finite rows. The StreamName common-refinement theorem supplies the public refinement face, the RegSeqRat dyadic-triangle equivalence supplies the selector/readback face, and the Real finite-cover pasting theorem supplies the terminal finite-cover face. These three certificates repack the displayed rows as the Phase Real and Completion exit surface.

The exclusions are part of the same packaging. None of the three cited surfaces has a row for quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Therefore acceptance by the current-phase exit surface is exactly acceptance of the four displayed finite faces and no stronger comparison. □

Theorem 301.166 (RegSeqRat selector-composition visible boundary)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S, T, and U be accepted finite requests to P whose pairwise faces display the shared-window selector surface of Definition 301.157. The route that first reads the $S \to T$ selector and then reads the $T \to U$ selector determines the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary as the direct $S \to U$ selector if and only if the two-step and direct presentations agree on exactly the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-scale rows,* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and tail-budget rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *rows, displayed* Cont *rows,* Pkg *provenance, and local* NameCert *rows.*

The comparison is only a comparison of the BEDC-visible terminal boundary over the ordered handoff

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It has no coordinate for quotient streams, host real equality, selected limits, completion points, choice data, ambient completeness, or any second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. For the forward direction, apply Definition 301.163. The two-step read through $S \to T$ and $T \to U$ and the direct read through $S \to U$ are restrictions of one triple-overlap selector surface, so their visible StreamName, DyadicRatCore, RegSeqRat, terminal Real, hsame, Cont, Pkg, and local NameCert rows are the same rows.

Next use Definition 301.164. It identifies the common-scale dyadic triangle ledger with the RegSeqRat selector surface and the terminal $\mathsf{Real}^{\uparrow}$ seal read on the same ordered four-face route. Thus agreement of the displayed rows is exactly agreement of the terminal visible boundary for the two-step and direct selector presentations.

For the reverse direction, suppose the terminal $\mathsf{Real}^{\uparrow}$ seal boundary exposed by the two-step route agrees with the boundary exposed by the direct $S \to U$ route. Apply Definition 301.159 to recover the shared-window selector surface from the terminal boundary on each compared route.

The recovered selector surfaces have the same displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. Applying Definition 301.163 repacks those rows as one triple-overlap surface, and Definition 301.164 presents the same data as the dyadic triangle ledger.

Finally, Definition 301.167 excludes quotient streams, host real equality, selected limits, completion points, choice data, ambient completeness rows, and second seal coordinates from both routes. Hence no hidden coordinate can separate the two-step selector read from the direct selector read once the displayed terminal boundary agrees. □

Theorem 301.167 (RegSeqRat Real seal selector non-escape)**.** *A* $\mathsf{Real}^{\uparrow}$ *seal consumer of the RegSeqRat shared-window selector surface receives only the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and tail-budget rows, componentwise* hsame *rows, displayed* Cont *replay rows,* Pkg *provenance, local* $\mathsf{NameCert}$ *provenance, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. The consumer receives no quotient stream, host real equality, selected limit, hidden selector, detached completeness row, or second* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with Definition 301.123. It exhausts a selector-facing seal read by the finite StreamName, DyadicRatCore, RegSeqRat, and RealUp rows of the accepted packet.

Apply the finite request Real seal correspondence of Definition 301.141. A finite request reaches the seal only through its displayed source section, not through an external real equality or selected limit.

Use the terminal tail-budget seal theorem Definition 301.114. It keeps the tail budget and seal route on the same finite dyadic ledger and removes detached completion and completeness coordinates from the consumer inventory.

The handoff exactness theorem Definition 301.69 supplies the remaining source-lock: the StreamName schedule and dyadic rows travel together into the Real-facing seal. Since the shared-window selector surface has no row for the excluded objects, the consumer cannot export any of them. □

Definition 301.168 (RegSeqRat tail-budget packet)**.** *The RegSeqRat tail-budget packet is the finite packet displayed to a* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier. Its rows are the dyadic observation rows from* $\mathsf{DyadicRatCore}^{\uparrow}$*, the StreamName window rows naming the finite request, the regularity-budget rows of the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier, the tail-selector rows used by the request, the componentwise transport rows, displayed* Cont *continuations,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *naming row.*

The packet has no hidden completion carrier. Its visible data are exactly the tail-budget source rows that feed the public $\mathsf{Real}^{\uparrow}$ *seal.*

Theorem 301.169 (RegSeqRat tail-budget non-escape)**.** *A* $\mathsf{Real}^{\uparrow}$ *seal consumer of a RegSeqRat tail-budget packet can read only the displayed dyadic observation rows, StreamName window rows, regularity-budget rows, tail-selector rows, transport, continuation, provenance, and local naming rows of Definition 301.168. It cannot obtain quotient sequence equality, a selected limit, countable choice, or ambient completeness from that packet.*

Proof. Begin with Definition 301.114. It places every completion-facing tail-budget consumer on the finite route through the StreamName schedule, DyadicRatCore ledger, RegSeqRat classifier and regularity rows, displayed support rows, and the public RealUp seal.

Next apply Definition 301.158. The shared-window selector is determined by the displayed finite source rows and by the local transport, continuation, provenance, and naming data; hidden selectors and selected limits do not distinguish a second route.

Use Definition 301.167 to remove the excluded coordinates at the Real-facing boundary. It prevents quotient streams, selected limits, detached completion rows, and second seal data from being exported by the selector surface.

The rows remaining after these references are precisely the packet rows of Definition 301.168. Since no cited boundary supplies quotient sequence equality, a selected limit, countable choice, or ambient completeness, a RealUp seal consumer cannot obtain them from the tail-budget packet. □

Theorem 301.170 (RegSeqRat tail-budget sibling route)**.** *Every RegSeqRat tail-budget packet used by a* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through the sibling chapter surfaces chapter 126, chapter 291, and chapter 38 before Definition 301.169 applies. The StreamName surface supplies the finite windows, the DyadicRatCore surface supplies the dyadic tolerances, and the RealUp surface supplies the terminal seal reads for the same displayed packet.*

Proof. Use Definition 301.169. Its route inventory already identifies the dyadic observation rows, StreamName window rows, regularity-budget rows, tail-selector rows, transports, continuations, provenance, and local naming rows that can be read by a RealUp seal consumer.

The three sibling surfaces provide the named external faces of that inventory. The StreamName chapter supplies the finite-window reads, the DyadicRatCore chapter supplies the dyadic tolerance rows, and the Real chapter supplies the terminal RealUp seal reads. Once those faces are displayed, the local non-escape theorem applies to the same tail-budget packet. □

Theorem 301.171 (RegSeqRat tail-budget four-face uniqueness)**.** *Let two* $\mathsf{Real}^{\uparrow}$ *seal consumers of RegSeqRat tail-budget packets expose the same four visible faces: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity, classifier, and provenance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. Then they determine the same BEDC-visible terminal seal route through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

This uniqueness statement uses only the visible tail-budget packet. It does not identify quotient streams, choose a limit, assert host real equality, use countable choice, assume ambient completeness, or add standard bridge rows.

Proof. Begin with Definition 301.169. It fixes the packet inventory available to a $\mathsf{Real}^{\uparrow}$ seal consumer: the finite StreamName rows, dyadic observation rows, regularity-budget rows, selector rows, transport, continuation, provenance, and local naming data. The same theorem excludes quotient stream equality, selected limits, countable choice, and ambient completeness from that inventory.

Next use Definition 301.170. The visible rows are read through the StreamName, DyadicRatCore, and Real sibling surfaces, so the two consumers have the same three external supports for the same tail-budget packet.

Now apply Definition 301.61. Agreement on the StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity/classifier/provenance face, and terminal $\mathsf{Real}^{\uparrow}$ seal face is complete for the accepted terminal classifier surface.

Finally invoke Definition 301.59. That theorem identifies the terminal route once the four faces and the displayed support rows agree. The StreamName terminality theorem Definition 126.449, the DyadicRatCore refinement uniqueness theorem Definition 291.96, and the Real L10 terminal correspondence Definition 38.455 name the same external faces of the L10 route. Therefore the two consumers determine one BEDC-visible terminal seal route through the displayed BHist, hsame, Cont, Pkg, and local NameCert rows, with no quotient stream equality, selected limit, host real equality, countable-choice row, ambient-completeness row, or standard bridge row added. □

Theorem 301.172 (RegSeqRat tail-budget exhaustion)**.** *Every finite dyadic request admitted by the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail selector is exhausted by a displayed tail budget and cannot escape the* $\mathsf{Real}^{\uparrow}$ *seal route. The displayed budget contains the StreamName window row, DyadicRatCore tolerance row, RegSeqRat regularity row, selected tail row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Thus no admitted finite dyadic request can be consumed as a Real-facing request without passing through the same tail-budget packet and the same public seal boundary.

Proof. Begin with the tail-budget handoff of Definition 301.109. It names the finite StreamName window, DyadicRatCore observation rows, RegSeqRat regularity rows, and local support rows that any Real-facing tail-budget request must display.

Apply the tail selector Real-seal exhaustion theorem, Definition 301.114. The selector cannot export a finite dyadic request to the Real seal unless the displayed budget rows already carry that request.

Now use the non-escape result Definition 301.169. It removes quotient sequence equality, selected limits, countable choice, and ambient completeness from the tail-budget consumer route.

The remaining data are exactly the displayed tail budget and the public RealUp seal route. Therefore every admitted finite dyadic request is exhausted by that budget and cannot escape the seal boundary. □

Theorem 301.173 (RegSeqRat tail-budget exit status product)**.** *Every accepted finite dyadic request admitted by the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail selector and consumed as a Phase Real and Completion endpoint is audit-readable only as the componentwise status product of four local face records: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. Each face contributes its own* closurestatus *and* formalstatus *record, while the shared transport is only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

The exit read contains no quotient stream, host equality, selected limit, countable-choice principle, ambient completeness datum, metric-completion datum, bridge row, aggregate formalstatus *token, or promotion from one status axis to the other.*

Proof. Start with Definition 301.172. It puts every admitted finite dyadic request on the displayed tail-budget packet and keeps the Real-facing consumer inside the same public $\mathsf{Real}^{\uparrow}$ seal boundary.

Next apply Definition 301.58 and Definition 301.62. These results identify the terminal read as the four-face ledger consisting of the DyadicRatCore tolerance face, StreamName finite-window face, RegSeqRat regularity and classifier face, and terminal Real seal face, transported only by the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally read the ledger through the two-axis discipline of chapter 9203. A face's closure record and verification record remain local to that face, so their product is only a componentwise audit surface. It cannot add a quotient stream, host equality, selected limit, countable-choice principle, completeness datum, bridge row, aggregate verification token, or implication from one axis to the other. □

Theorem 301.174 (RegSeqRat tail-budget status product reverse)**.** *Any finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget packet exposing exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows presents the same public Phase Real and Completion exit status product.*

The packet has no aggregate formalstatus *coordinate, bridge coordinate, route-level completeness coordinate, quotient stream, host real equality, selected limit, or countable-choice selector. Its public read is therefore exactly the componentwise four-face status product of Definition 301.173.*

Proof. Apply Definition 301.172. Every admitted finite dyadic request has already been forced onto the displayed tail-budget packet before the public $\mathsf{Real}^{\uparrow}$ seal can consume it.

Next use Definition 301.58. The exposed DyadicRatCore tolerance ledger, StreamName finite-window row, RegSeqRat regularity and classifier row, and local Real seal row form the same terminal four-face pullback over the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Now apply Definition 301.169. No quotient stream, host equality, selected limit, countable-choice principle, ambient completeness datum, or metric-completion datum can enter the consumer route from outside those rows.

Finally use Definition 301.177. Since the displayed face rows and support rows are exactly the support-minimal packet, adding a route-level formalstatus token, bridge row, or completeness coordinate would not be a component of the accepted public read.

Read the result through the two-axis discipline of chapter 9203. The same face-local rows reconstruct the same componentwise status product, and no route-level token is introduced. □

Theorem 301.175 (RegSeqRat four-face exit consumer normal form)**.** *Every Phase Real and Completion consumer that reads an accepted finite dyadic request from the* $\mathsf{RegSeqRat}^{\uparrow}$ *selector factors through exactly four displayed faces: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The transport between these faces is only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Conversely, any consumer exposing these four faces with the same displayed transport has the same public L10 exit read. The normal form exports no quotient stream, host real equality, selected limit, countable-choice selector, metric-completeness datum, aggregate formalstatus *row, or bridge row.*

Proof. Start with Definition 301.173. It places every accepted finite dyadic request consumed at a Phase Real and Completion endpoint on the four componentwise status faces and records the admissible transport rows.

Next apply Definition 301.58. The terminal pullback identifies the same four faces as a single finite BHist surface and shows that the route is carried only by the displayed hsame, Cont, Pkg, and local NameCert rows.

Use Definition 126.372 and Definition 291.110 to align the consumer-side finite-window read with the objectwise DyadicRatCore, StreamName, RegSeqRat, and Real exit records. These identifications make the forward factorization and the reverse same-read criterion the same public L10 read.

Finally apply the two-axis discipline of section 9467.2. It keeps closure rows, verification rows, and bridge rows local to their own faces, so the consumer normal form cannot introduce quotient streams, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, or bridge rows. □

Theorem 301.176 (RegSeqRat tail-budget finite-window refusal localization)**.** *Let a finite dyadic request be addressed to a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget route. If the request is not accepted as a Phase Real and Completion four-face exit consumer, then the defect is localized to at least one displayed face row or one displayed support row.*

The possible face rows are exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The possible support rows are exactly* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, aggregate* formalstatus *tokens, aggregate bridge rows, and* $\mathsf{Real}^{\uparrow}$ *completeness cannot serve as the defect carrier.*

Proof. Apply Definition 301.172. Any finite dyadic request that reaches the tail selector is exhausted by the displayed tail budget before the public $\mathsf{Real}^{\uparrow}$ seal can read it, so a non-accepted request cannot be explained by an off-budget object.

Next use Definition 301.173. The accepted endpoint is a componentwise status product over the four listed faces, transported only by BHist, hsame, Cont, Pkg, and NameCert. Hence a missing acceptance witness must remove one of those component rows or one of those transport rows.

Now apply Definition 301.175. That normal form is also the reverse public-read criterion: exposing the same four faces with the same displayed transport gives the same L10 exit read. Therefore any request that does not expose such a read has its defect on the named face or support rows.

Finally read the conclusion through the two-axis discipline of section 9467.2. Since closure rows, verification rows, and bridge rows do not promote into one another, no quotient equality, host equality, selected limit, choice principle, completeness datum, aggregate status token, aggregate bridge row, or $\mathsf{Real}^{\uparrow}$ completeness claim can replace the missing displayed row. □

Theorem 301.177 (RegSeqRat tail-budget support minimality)**.** *Every accepted finite dyadic request consumed by the Phase Real and Completion* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget route has a support-minimal four-face packet. Its face rows are exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. Its shared support rows are exactly the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Deleting any one of these face or support rows destroys the accepted public read. Quotient stream equality, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus *tokens, aggregate bridge rows, and* $\mathsf{Real}^{\uparrow}$ *completeness cannot replace the missing row.*

Proof. Apply Definition 301.175. The forward direction factors every accepted finite dyadic request through the four displayed faces and the listed transport support. The reverse direction says that the same four faces with the same support rows give the same public L10 exit read.

Now delete one displayed face row or one displayed support row. The resulting packet no longer satisfies the reverse same-read criterion, so it is not an accepted four-face exit consumer for the same request.

Apply Definition 301.176 to the failed read. Its defect must localize to one of the displayed face rows or support rows, and the localization list is exactly the list removed from the packet above.

Finally use Definition 301.173 together with the two-axis discipline of section 9467.2. Aggregate verification tokens and bridge rows are not component rows of the status product, and neither quotient equality, host equality, selected limits, choice principles, completeness data, nor $\mathsf{Real}^{\uparrow}$ completeness can promote into the missing face or support row. Thus no displayed row can be dropped. □

Theorem 301.178 (RegSeqRat tail-window stability)**.** *Let an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget packet expose a finite tail window as a Phase Real and Completion four-face exit consumer. If that finite tail window is replaced by a certified finer tail window with the same displayed request and support rows, then the public exit read preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal row.*

The stability read exports no quotient stream equality, selected limit, countable-choice selector, $\mathsf{Real}^{\uparrow}$ *completeness datum, metric-completion datum, aggregate* formalstatus *token, or bridge row.*

Proof. Start with Definition 301.172. The original accepted request is already exhausted by the displayed tail budget before the public $\mathsf{Real}^{\uparrow}$ seal can consume it.

Apply Definition 301.173. The public read is componentwise: the DyadicRatCore tolerance ledger, StreamName finite-window row, RegSeqRat regularity and classifier row, and Real seal row remain separate faces over the same BHist, hsame, Cont, Pkg, and NameCert support.

Now use Definition 301.175. A certified finer tail window with the same displayed support rows exposes the same four-face consumer normal form, so it has the same L10 public exit read as the original window.

Finally apply Definition 301.177. Since the four face rows and shared support rows are exactly the support-minimal packet, the refinement can neither drop a displayed row nor replace one by quotient stream equality, a selected limit, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, an aggregate status token, or a bridge row. □

Theorem 301.179 (RegSeqRat tail-budget common-threshold exhaustion)**.** *Let q_0 and q_1 be accepted finite dyadic requests for the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget route. Suppose they share the same displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, local* $\mathsf{Real}^{\uparrow}$ *seal row, and displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Then the two requests admit one common displayed threshold packet. Every Real-facing consumer of either request factors through that packet and the same four-face exit surface. The common-threshold read exports no quotient stream, selected limit, host equality, countable choice, metric-completion datum, aggregate formalstatus *token, or bridge row capable of distinguishing the two requests.*

Proof. Start with Definition 301.178. Since the requests share the same schedule, tolerance, classifier, regularity, Real-seal, and support rows, refining either finite tail window along the common support preserves the same four displayed faces.

Next apply Definition 301.175. A Real-facing consumer can read an accepted request only through the DyadicRatCore tolerance face, StreamName finite-window face, RegSeqRat classifier face, Real seal face, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Use Definition 301.177. The shared face and support rows are exactly the support-minimal packet, so a common displayed threshold can be chosen inside that finite support and no omitted row can be recovered from outside it.

Finally the normal-form non-escape clauses remove quotient streams, selected limits, host equality, countable choice, metric-completion data, aggregate verification tokens, and bridge rows from the consumer route. Thus no Real-facing consumer distinguishes q_0 from q_1 except through the common displayed threshold packet, and both requests factor through it. □

Definition 301.180 (RegSeqRat common-tail window packet)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail window packet is a finite* BHist *packet over one accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source P with the following displayed rows:*

1. *two certified cofinal tail shifts T_0 and T_1 of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P;*
2. *one common* $\mathsf{StreamName}^{\uparrow}$ *refinement window W_* through which the selected tail windows of T_0 and T_1 both restrict;*
3. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read on W_*, together with the displayed radius rows and regularity bounds*
$$2^{-n} + 2^{-m}$$
for the selected unary indices;
4. *the* RegSeqRatClassifier *rows for the two tail-window reads and for their transported read on W_*;*

5. *the* $\mathsf{Real}^{\uparrow}$ *seal-source rows consumed from the same common finite surface by Definition 301.22;*
6. *the componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance attached to the StreamName, DyadicRatCore, RegSeqRat, and RealUp-facing rows.*

The packet is only a finite common-tail source surface. It stores no quotient stream, host real value, selected limit, metric-completion point, $\mathsf{Real}^{\uparrow}$ *completeness datum, or standard-bridge coordinate beyond the displayed rows.*

Theorem 301.181 (RegSeqRat common-tail union classifier determinacy)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the two shifted tail-window reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. Then the union window determined by that common-tail packet determines one accepted* RegSeqRatClassifier *read for both shifted tail windows, over exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows,* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance.*

The determinacy statement is only a finite BEDC classifier read on the common union window. It exports no quotient stream equality, host real equality, selected limit, ambient completeness row, metric-completeness row, hidden completion carrier, or additional seal coordinate.

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatCommonTailUnionClassifierDeterminacy`

Proof. Project the common-tail window packet of Definition 301.180. Its rows provide the two certified cofinal tail shifts, the common $\mathsf{StreamName}^{\uparrow}$ refinement W_*, the transported dyadic endpoint and radius ledgers, the two tail classifier rows, the common transported classifier row, and the displayed hsame, Cont, Pkg, and NameCert provenance.

Apply Definition 301.9 to the two transported tail reads on W_*. The hypotheses identify the transported $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, radius rows, regularity ledgers, classifier ledgers, componentwise comparisons, continuation routes, and package provenance, so the two shifted schedules expose one RegSeqRatClassifier read on the common union window.

Use Definition 301.25. The common $\mathsf{StreamName}^{\uparrow}$ refinement locks the dyadic endpoint, radius, distance, and regularity rows to the same finite ledger, so the accepted classifier read cannot branch through a separate schedule coordinate or a separate dyadic coordinate.

Finally route the common classifier through Definition 301.182 and Definition 301.22. The bridge identifies the shared classifier and seal-source surface already present on W_*, and the seal handoff consumes exactly those finite regular-rational rows. Hence quotient stream equality, host real equality, selected limits, ambient completeness, metric-completeness rows, hidden completion carriers, and extra seal data have no displayed row from which to distinguish the two shifted tail windows. □

Theorem 301.182 (RegSeqRat common-tail window classifier bridge)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source, and let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule of P. Suppose two tail-window classifier reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. Then the two reads determine the same* $\mathsf{Real}^{\uparrow}$ *seal-source surface and the same* RegSeqRatClassifier *read after restriction to W_*.*

The bridge conclusion is a statement about the displayed finite packet only. It exports no quotient stream, host real equality, selected limit, metric-completion point, $\mathsf{Real}^{\uparrow}$ *completeness datum, or additional seal coordinate.*

Proof. First use Definition 301.36. The two seal consumers carried by the cofinal tail shifts factor through one accepted common finite window, and the visible rows of that window are exactly the StreamName schedule rows, DyadicRatCore endpoint and radius ledgers, RegSeqRat classifier and regularity ledgers, and the displayed provenance.

Next read the common-tail packet rows componentwise. The definition lists the cofinal tail shifts, the common $\mathsf{StreamName}^{\uparrow}$ refinement W_*, the DyadicRatCore endpoint and radius ledgers, the classifier rows, the $\mathsf{Real}^{\uparrow}$ seal-source rows, and the hsame, Cont, Pkg, and NameCert provenance. Thus each tail-window read restricts to the same finite source surface on W_*.

Apply Definition 301.30. The refined-window gluing surface factors the RealUp-facing handoff through the displayed finite packet W_*, so the two tail-window origins cannot supply separate seal-source surfaces once their rows have been restricted to the common refinement.

Use Definition 301.25 to lock the $\mathsf{StreamName}^{\uparrow}$ schedule rows to the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers inside the common window. The classifier reads inspect only those locked rows together with the regularity ledgers and provenance, so their transported RegSeqRat classifier read on W_* is the same displayed read.

Finally invoke Definition 301.22. The $\mathsf{Real}^{\uparrow}$ seal consumes exactly the accepted regular-rational finite source surface already identified, and the carrier has no coordinate for quotient streams, host real equality, selected limits, metric completions, completeness data, or further seal coordinates. □

Theorem 301.183 (RegSeqRat common-tail window terminality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the two shifted tail reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. Every completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer whose visible rows are exactly the two shifted reads, the common* $\mathsf{StreamName}^{\uparrow}$ *window W_*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and regularity rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, the displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance factors through that common-tail packet.*

The terminal factorization exports no quotient stream, host real equality, selected limit, metric-completion point, $\mathsf{Real}^{\uparrow}$ *completeness datum, or additional seal coordinate.*

Proof. First apply Definition 301.36 to the two cofinal tail shifts. The tail-stable cover places both completion-facing seal consumers on one accepted common finite window with the StreamName schedule rows, DyadicRatCore endpoint and radius ledgers, RegSeqRat classifier and regularity ledgers, and displayed provenance.

Next use Definition 301.30. The common refinement W_* repacks the two shifted tail reads into one RealUp-facing seal source, so every consumer whose visible rows are limited to the displayed shifts and the common window is already routed through the common-tail packet.

Apply Definition 301.25. It locks the common $\mathsf{StreamName}^{\uparrow}$ refinement to the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and regularity rows, leaving the consumer no separate dyadic or schedule coordinate outside W_*.

Now invoke Definition 301.182. The transported classifier read and the transported seal-source rows are the same rows listed by the common-tail packet, including the displayed hsame, Cont, Pkg, and NameCert provenance.

Finally apply Definition 301.22. Since the $\mathsf{Real}^{\uparrow}$ seal consumes exactly this accepted regular-rational finite source surface, the consumer factors through the common-tail packet. Quotient streams, host real equality, selected limits, metric-completion points, $\mathsf{Real}^{\uparrow}$ completeness data, and extra seal coordinates have no displayed carrier row from which to be read. □

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatCommonTailWindowTerminality`

Theorem 301.184 (RegSeqRat common-tail finite-request Real seal pullback)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, and let the two shifted tail reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. For any finite-request* $\mathsf{Real}^{\uparrow}$ *seal consumer whose request is visible along that common-tail packet, the visible pullback of the consumer is exactly the finite source surface listed in Definition 301.180: the* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *provenance.*

Equivalently, the common-tail window source surface and the finite-request $\mathsf{Real}^{\uparrow}$ *seal pullback present the same BEDC-visible rows. The equivalence exports no quotient stream, selected limit, host real equality, ambient completeness row, hidden completion carrier, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First apply Definition 301.183. Any completion-facing consumer whose visible rows are the shifted tail reads, the common window W_*, the dyadic ledger, classifier, seal-source rows, transports, continuations, package provenance, and local naming must factor through the common-tail packet.

Next use Definition 301.113. A finite-request $\mathsf{Real}^{\uparrow}$ seal consumer has visible pullback exactly the StreamName schedule row, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier and regularity rows, displayed hsame, Cont, Pkg provenance, and local NameCert row.

Now apply Definition 301.159. It identifies equality of the shared selector surface with equality of the terminal $\mathsf{Real}^{\uparrow}$ seal reads after both requests have been read through the ordered finite packet from $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. Thus the common-tail presentation and the finite-request pullback presentation determine one another on the same displayed rows.

Finally invoke Definition 301.167. The selector boundary removes the excluded coordinates from both descriptions: quotient streams, selected limits, host real equality, ambient completeness data, hidden completion carriers, and second seal data have no row in either surface. The remaining rows are precisely the StreamName schedule and refinement rows, DyadicRatCore endpoint, radius, and distance rows, RegSeqRat classifier and regularity rows, RealUp seal-source rows, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert provenance listed in Definition 301.180. □

Theorem 301.185 (RegSeqRat common-tail visible packet assembly)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose a finite-request* $\mathsf{Real}^{\uparrow}$ *seal consumer exposes exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows,* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *provenance for the two shifted tail reads. Then those rows assemble a common-tail window packet of Definition 301.180 with common refinement W_*.*

The assembled packet is unique at the displayed-row level. It exports no quotient stream, selected limit, host real equality, ambient completeness, metric-completeness row, hidden completion carrier, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. First use Definition 301.36. The two certified cofinal tail shifts and the finite-request seal consumer are forced onto one common finite window, so the candidate refinement W_* is the shared support on which the displayed rows may be read.

Next apply Definition 301.25. The exposed $\mathsf{StreamName}^{\uparrow}$ schedule and refinement rows lock the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance rows to the same finite ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows are therefore read on that locked window.

Use Definition 301.45 and Definition 301.48. These results say that the visible finite-request $\mathsf{Real}^{\uparrow}$ seal boundary has no further source rows beyond the listed schedule, dyadic, classifier, seal-source, hsame, Cont, Pkg, and NameCert data. Hence the displayed rows are exactly the rows required by Definition 301.180.

Finally invoke Definition 301.184. The pullback theorem identifies the assembled common-tail packet with the finite-request seal boundary on the same visible rows. Uniqueness follows because two assemblies with the same W_* and the same displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows have no remaining BEDC coordinate on which to differ, and the excluded quotient, limit, equality, completeness, carrier, and second-seal coordinates are not part of the packet. □

Theorem 301.186 (RegSeqRat common-tail Real boundary equivalence)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, and suppose the shifted tail reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. Then the common-tail packet and the finite-request* $\mathsf{Real}^{\uparrow}$ *seal boundary determine one another exactly on the* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *provenance.*

The public boundary is only this finite BEDC surface. It excludes quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, and any completion row not displayed by the common-tail packet.

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatCommonTailRealBoundaryEquivalence`

Proof. First apply Definition 301.182. The two cofinal tail reads restrict to one classifier and one seal-source surface on W_*, so the common-tail packet already determines the RegSeqRat and RealUp-facing rows of the boundary.

Next use Definition 301.183. Every completion-facing consumer whose visible rows are the two shifts, the common StreamName window, the dyadic endpoint-radius-distance ledger, the classifier and regularity ledger, and the displayed structural rows factors through the common-tail packet. Thus a finite-request $\mathsf{Real}^{\uparrow}$ seal boundary cannot add a second route outside that packet.

Now apply Definition 301.185. It assembles any finite-request boundary exposing exactly the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows back into the common-tail packet on the same refinement W_*.

Then use Definition 301.184. It identifies the visible pullback of any finite-request seal consumer with exactly the rows listed in Definition 301.180. Hence the finite-request boundary and the common-tail packet determine one another on the same displayed rows.

Finally compose the Real L10 four-face route results. The canonical route and exactness theorems, Definition 38.450 and Definition 38.457, place the accepted Real-and-Completion read on the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. The terminal correspondence and pullback uniqueness theorems, Definition 38.455 and Definition 38.458, then repack the same displayed rows as one terminal finite boundary. Since none of the cited packets carries quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, or hidden completion data, the two presentations determine one another exactly on the stated BEDC-visible surface. □

Theorem 301.187 (RegSeqRat common-tail support descent nonescape)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, and suppose the shifted tail reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. A* $\mathsf{Real}^{\uparrow}$*-facing terminal consumer or finite-cover consumer descends through the common-tail packet if and only if its visible support rows are exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont, Pkg*, and local* $\mathsf{NameCert}$ *provenance.*

This descent surface excludes quotient stream equality, selected limits, host real equality, countable choice, metric-completeness data, ambient $\mathsf{Real}^{\uparrow}$ *completeness, aggregate route-level* $\mathsf{formalstatus}$ *rows, and bridge rows. No such coordinate can be used to escape the finite support displayed by the common-tail packet.*

Proof. For the forward direction, suppose the terminal or finite-cover consumer descends through the common-tail packet. Apply Definition 301.186. The descended consumer has the same finite boundary as the common-tail packet, hence its visible rows are confined to the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

Use Definition 301.184. It identifies the visible pullback of any finite $\mathsf{Real}^{\uparrow}$ seal request through the packet with exactly those source rows. Then Definition 126.558 keeps the $\mathsf{StreamName}^{\uparrow}$ section rows terminal under restriction to the cofinal tail window, so a finite-cover refinement cannot introduce a second schedule coordinate.

Finally apply Definition 38.846. The L10 terminal cover exhausts the Real-facing coordinate list on the four displayed faces, and the common-tail equivalence has already identified those faces with the packet rows. Thus the descended consumer exposes precisely the stated support and nothing else.

For the reverse direction, suppose the consumer exposes exactly the displayed support rows. The $\mathsf{StreamName}^{\uparrow}$ schedule and refinement rows, together with terminal section coherence, place the request on the same finite section window W_*. The DyadicRatCore endpoint-radius-distance ledger and the RegSeqRat classifier and regularity rows are therefore read on the same locked common-tail surface.

Apply Definition 301.184 to read the consumer as the finite $\mathsf{Real}^{\uparrow}$ seal pullback over that surface. Then use Definition 301.186 to identify this pullback with the common-tail packet boundary. The terminal-cover coordinate exhaustion theorem supplies the finite-cover case by showing that every retained cover coordinate is already one of the displayed rows.

The nonescape clause is the row inventory of these four applications. The common-tail equivalence, finite-request pullback, StreamName terminal coherence, and Real L10 coordinate exhaustion expose no quotient stream equality, selected limit, host real equality, countable-choice principle, metric-completeness datum, ambient $\mathsf{Real}^{\uparrow}$ completeness row, aggregate route-level status row, or bridge row. Hence the descent criterion cannot factor through any support outside the displayed packet. □

Theorem 301.188 (RegSeqRat common-tail retained-window induction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{W} = \{W_i\}_{i\in I}$ be a nonempty finite retained family of common-tail window packets over certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the retained family is closed under the displayed finite refinements used by Definition 301.185 and that every retained pair has the classifier bridge and finite-request* $\mathsf{Real}^{\uparrow}$ *seal pullback of Definition 301.182 and Definition 301.184. Then finite induction over $\mathcal{W}$ preserves exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger,*

the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *provenance.*

The induction invariant is the finite common-tail packet surface itself. At the base window, at each retained refinement step, and after adjoining any further retained window, the resulting packet is equivalent to its finite-request $\mathsf{Real}^{\uparrow}$ *boundary in the sense of Definition 301.186. The induction exports no quotient stream, selected limit, host real equality, countable-choice principle, metric-completeness row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, hidden completion carrier, or additional seal coordinate.*

Proof. For the base window, read the chosen member of $\mathcal{W}$ through Definition 301.180. Its visible rows are precisely the StreamName schedule and refinement rows, DyadicRatCore endpoint-radius-distance ledger, RegSeqRat classifier and regularity rows, RealUp seal-source rows, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ provenance.

For a retained refinement step, apply Definition 301.182. The classifier and seal-source rows restrict to the common refinement without changing the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, or $\mathsf{NameCert}$ surface.

Next apply Definition 301.184. It identifies the visible pullback of every finite $\mathsf{Real}^{\uparrow}$ seal request with exactly the same packet rows, so the induction step cannot add a quotient stream, selected limit, host real equality, metric-completeness row, ambient completeness datum, or second seal coordinate.

When adjoining another retained window, use Definition 301.185. The displayed rows assemble again into a common-tail packet over the retained refinement, and the nonempty finite hypothesis lets this step iterate through the retained family in the usual finite induction order.

Finally invoke Definition 301.186. At every stage the current retained packet and the finite-request $\mathsf{Real}^{\uparrow}$ boundary determine one another on the same BEDC-visible rows. Thus the induction preserves exactly the stated packet surface and refuses quotient streams, selected limits, host real equality, countable choice, metric-completeness data, ambient $\mathsf{Real}^{\uparrow}$ completeness, hidden completion carriers, and additional seal coordinates. □

Theorem 301.189 (RegSeqRat common-tail four-face cocycle determinacy)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose each pair (T_i, T_j) lands in a common-tail window packet of Definition 301.180, and suppose the three pairwise packets restrict to one triple* $\mathsf{StreamName}^{\uparrow}$ *refinement W_* with the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger and the same* $\mathsf{Real}^{\uparrow}$ *seal-source row.*

Then the three pairwise common-tail $\mathsf{Real}^{\uparrow}$ *boundaries satisfy one finite four-face cocycle exactly when the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows agree on the triple refinement W_*. The cocycle surface has no coordinate for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard-bridge data, or hidden completion carriers.*

Proof. Apply Definition 301.186 to the three pairs (T_0, T_1), (T_1, T_2), and (T_0, T_2). Each pairwise boundary is thereby identified with the same kind of finite common-tail source surface: StreamName schedule and refinement rows, DyadicRatCore endpoint-radius-distance rows, RegSeqRat classifier and regularity rows, RealUp seal-source rows, and the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

On the triple refinement W_*, the definition of the common-tail window packet identifies those rows componentwise. The hypotheses give one StreamName refinement, one dyadic ledger, and one

Real seal-source row, so the RegSeqRat classifier and regularity rows and the support provenance are the only remaining displayed coordinates to compare.

For the forward direction, a finite four-face cocycle already pastes the three pairwise Real boundaries along the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. The Real L10 canonical route and exactness results, Definition 38.450 and Definition 38.457, force that pasted surface to read through the displayed rows on W_*, so disagreement on any listed row would give two different finite boundaries for the same face.

For the reverse direction, assume the displayed rows agree on W_*. Use the Real L10 terminal correspondence and pullback uniqueness results, Definition 38.455 and Definition 38.458, to repack the shared rows as one terminal four-face surface. The three pairwise common-tail boundaries are the restrictions of that surface, hence they satisfy the finite cocycle.

No additional boundary coordinate remains to choose. The preceding common-tail Real boundary equivalence and finite-request seal pullback show that quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridges, and hidden completion data are not rows of the packet, so they cannot split the cocycle after the displayed surface agrees. □

Theorem 301.190 (RegSeqRat common-tail triple-overlap cocycle)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose each shifted pair (T_i, T_j) lands in a common-tail window packet of Definition 301.180, and suppose the three packets restrict to one triple refinement W_* carrying the same* $\mathsf{StreamName}^{\uparrow}$ *schedule rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the same* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*, displayed* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance.*

Then the three pairwise $\mathsf{Real}^{\uparrow}$ *boundary equivalences agree on the triple refinement if and only if W_* exposes exactly that same finite common-tail surface. This triple-overlap cocycle exports no quotient stream, host real equality, selected limit, countable choice, ambient completeness, metric-completeness row, standard-bridge coordinate, or completion datum outside the displayed packet.*

Proof. First apply Definition 301.182 to the three pairs (T_0, T_1), (T_1, T_2), and (T_0, T_2). Each pairwise shifted read is transported to the same kind of common-tail classifier and seal-source surface on its refinement face.

Use Definition 301.183 next. Every completion-facing consumer whose visible rows are the shifted reads, the common StreamName refinement, the dyadic endpoint-radius-distance ledger, the classifier and regularity rows, the seal-source rows, componentwise hsame, displayed Cont, Pkg provenance, and local $\mathsf{NameCert}$ provenance factors through the corresponding common-tail packet.

For the forward direction, assume the three pairwise $\mathsf{Real}^{\uparrow}$ boundary equivalences agree on W_*. Apply Definition 301.186 on each face. The Real L10 canonical route and exactness results cited there, Definition 38.450 and Definition 38.457, identify a unique BEDC-visible boundary on the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. Therefore any row visible to the triple refinement must be one of the displayed common-tail rows, and each such row has the same value on the three faces.

For the reverse direction, suppose W_* exposes exactly the common finite surface listed in the statement. The three pairwise boundaries are then restrictions of one displayed packet. Apply Definition 301.186 on each pair and use the Real L10 terminal correspondence and pullback uniqueness results, Definition 38.455 and Definition 38.458, to identify the same terminal finite boundary on all three restrictions.

The exclusions are forced by the carrier rows already used. Neither Definition 301.180 nor the Real L10 four-face route supplies quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard-bridge coordinates, or hidden completion data, so no such coordinate can distinguish the triple-overlap cocycle once the displayed common-tail surface agrees. □

Theorem 301.191 (RegSeqRat common-tail cocycle descent)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose each shifted pair (T_i, T_j) lands in a common-tail window packet of Definition 301.180, and suppose the three pairwise common-tail windows restrict to one triple refinement W_* carrying the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows, componentwise* hsame, *displayed* Cont, *displayed* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance.*

If the three pairwise $\mathsf{Real}^{\uparrow}$ *boundary equivalences agree on W_*, then the three boundaries descend to a single finite four-face exit boundary over exactly the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces, with precisely the same componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. Conversely, any single displayed four-face boundary with exactly those rows restricts to the three pairwise common-tail boundary equivalences. The descent boundary contains no quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge data, or hidden completion carriers.*

Proof. Assume first that the three pairwise $\mathsf{Real}^{\uparrow}$ boundary equivalences agree on W_*. Apply Definition 301.190. The triple refinement exposes exactly the common finite tail surface on the StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now use Definition 301.189. Since the displayed rows agree on the triple refinement, the three pairwise common-tail boundaries satisfy the finite four-face cocycle criterion. This turns pairwise agreement into one cocycle surface rather than three unrelated boundary presentations.

Package that surface by applying Definition 38.455 and Definition 38.458. The terminal Real L10 pullback identifies the shared StreamName, DyadicRatCore, RegSeqRat, and RealUp rows as one finite four-face exit boundary, and uniqueness prevents any second boundary with the same displayed rows.

For the converse direction, start with a single displayed four-face boundary over exactly those four faces and support rows. Restrict it along the three pairwise common-tail windows. Each restriction has precisely the rows required by Definition 301.186, so it recovers the corresponding pairwise $\mathsf{Real}^{\uparrow}$ boundary equivalence.

The exclusions follow from the same displayed-row accounting. The common-tail packet, the cocycle criterion, and the Real L10 terminal pullback provide no coordinates for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, standard bridge data, or hidden completion carriers, so none can appear in the descended boundary or in its pairwise restrictions. □

Theorem 301.192 (RegSeqRat common-tail cofinal-bundle four-face descent)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let $\mathcal{P}$:* $\mathsf{ProbeBundle}$ *be a finite bundle of certified unary observations served by the shared tail-compatible* $\mathsf{StreamName}^{\uparrow}$ *selector, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the bundled observations have* $\mathsf{InBundle}$ *witnesses, a shared finite-window budget, and pointwise* $\mathsf{RatHistoryClassifier}$ *handoff rows through the selector; suppose each shifted pair (T_i, T_j) lands in a RegSeqRat common-tail window packet of Definition 301.180; and suppose the three pairwise packets restrict to one triple refinement carrying the same bundled StreamName rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius,*

and distance ledger, the same $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the same* $\mathsf{Real}^{\uparrow}$ *seal-source rows, and the same support rows* hsame, Cont, Pkg, *and* NameCert.

Then the bundled StreamName unary rows, their InBundle *witnesses, the shared tail selector, the finite-window budget, the pointwise* RatHistoryClassifier *handoff rows, the dyadic endpoint-radius-distance ledger, the RegSeqRat classifier and regularity rows, the RealUp seal-source rows, and the support rows descend to one finite four-face boundary over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *This boundary contains no quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, standard bridge coordinate, or hidden completion carrier.*

Proof. First apply Definition 126.548 to the finite bundle $\mathcal{P}$. The bundled unary rows, their InBundle witnesses, the shared selector, the budget surface, the pointwise rational-history classifier handoffs, and the support rows are exhausted by one displayed StreamName packet.

Next restrict the three RegSeqRat common-tail packets to the stated triple refinement. By Definition 301.190, the three pairwise Real boundary equivalences agree exactly when that refinement exposes the common finite surface on the StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Now use Definition 301.191. The pairwise common-tail surfaces therefore descend to a single displayed four-face exit boundary with the same StreamName rows, dyadic ledger, RegSeqRat classifier and regularity rows, RealUp seal-source rows, and support provenance.

Finally apply Definition 38.455 and Definition 38.458. Terminality repacks the displayed boundary over the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and prevents a second boundary with the same rows.

The nonescape clauses in Definition 126.548, Definition 301.190, and Definition 301.191 remove the remaining possible coordinates. Quotient streams, selected limits, host real equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, standard bridge data, and hidden completion carriers are not rows of the bundled packet or of the common-tail descent boundary. □

Theorem 301.193 (RegSeqRat common-tail support induction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{P}$:* ProbeBundle *be a finite bundle of certified unary observations over P. A common-tail four-face boundary over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is accepted for $\mathcal{P}$ exactly when the displayed finite support of $\mathcal{P}$ admits an induction with the following constructors:*

1. *the empty support exposes only the accepted packet rows of P and no boundary beyond the visible common-tail packet;*
2. *a singleton observation exposes the same* BHist, *componentwise* hsame, Cont, Pkg, *and local* NameCert *rows on the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces, and the observation restricts through the common-tail window packet of Definition 301.180;*
3. *a union support is accepted exactly when its two accepted sub-supports have matching displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *rows on their overlap, each added observation restricts through the common-tail packet, and the triple-overlap cocycle descends to the same four-face boundary.*

The induced boundary is exhausted by the bundled StreamName unary rows, their InBundle *witnesses, the shared tail selector, the finite-window budget, the pointwise* RatHistoryClassifier *handoff rows, the dyadic endpoint-radius-distance ledger, the RegSeqRat classifier and regularity rows, the RealUp seal-source rows, and the support rows* BHist, hsame, Cont, Pkg, *and* NameCert.

No quotient stream, selected limit, host real equality, countable choice, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, standard bridge coordinate, or hidden completion carrier has a support row in this induction principle.*

Proof. For the empty support, the accepted packet P supplies the only rows to inspect. Apply Definition 301.180 and Definition 301.185. They identify the vacuous support surface with the finite common-tail packet whose coordinates are the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

For a singleton support, restrict the certified unary observation along the shared tail-compatible selector. The cofinal exhaustion theorem Definition 126.548 places the observation, its InBundle witness, the finite-window budget, and the rational-history handoff row on one displayed StreamName packet. Then Definition 301.184 identifies the same observation with the common-tail Real seal pullback.

Now suppose two finite supports have already been accepted and are being joined. The induction hypotheses give the same displayed BHist, hsame, Cont, Pkg, and local NameCert rows on each accepted sub-support. On the overlap, apply Definition 301.181 to obtain one classifier read for the union window.

The boundary comparison for the joined support is then forced by Definition 301.186. It says that the common-tail packet and the finite-request Real boundary determine one another on exactly the displayed support rows, so no separate boundary can appear after the union classifier has been fixed.

It remains to check coherence for a further observation added to the joined support. Apply Definition 301.190 on the two old support pieces and the new observation. The triple refinement exposes the same common finite surface, and Definition 301.191 descends the pairwise restrictions to one finite four-face boundary.

Conversely, assume a finite four-face boundary for $\mathcal{P}$ is accepted. Restrict it to each constructor layer of the displayed finite support. The empty and singleton restrictions recover the packet and visible observation rows by the first two readings above, while every union restriction recovers matching overlap rows by union classifier determinacy and Real boundary equivalence. The triple-overlap cocycle theorem recovers the required descent datum for each added observation.

Induction on the displayed finite support therefore gives precisely the accepted common-tail four-face boundary stated in the theorem. The row inventory of the cited packet, exhaustion, pullback, determinacy, boundary-equivalence, and cocycle-descent results contains no quotient stream, selected limit, host real equality, countable choice, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ completeness datum, standard bridge coordinate, or hidden completion carrier, so none can become a support row of the induction. □

Theorem 301.194 (RegSeqRat common-tail face-status objectwise pullback)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let T_0, T_1, and T_2 be three certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the shifted pairs land in accepted RegSeqRat common-tail triple-overlap packets whose pairwise boundaries agree on the shared* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Then the only status data pulled back by the resulting common-tail four-face cocycle are the four local face records read objectwise from the $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *chapters' own* closurestatus *blocks. The cocycle creates no aggregate* formalstatus*, no aggregate bridge token, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, and no metric-completeness or countable-choice coordinate.*

Proof. First apply Definition 301.190. The accepted triple-overlap surface is exactly the finite common-tail surface on the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows, with no hidden completion or bridge coordinate.

Next use Definition 301.186. The common-tail packet and the finite-request $\mathsf{Real}^{\uparrow}$ boundary determine one another on the same visible rows. Thus any status read available to the cocycle must pass through the same four visible faces rather than through a separate Real-completion carrier.

Now apply Definition 126.504. The terminal four-object boundary is built from the ordered $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces plus their displayed support rows. Terminality leaves no fifth status object in which an aggregate verification grade, bridge datum, completeness theorem, or choice principle could be stored.

Finally use Definition 126.507. That result reads closure grades, formalstatus tokens, Lean targets, and bridge rows only from the corresponding face chapters' own closurestatus records. Pulling those four records through the RegSeqRat common-tail cocycle therefore preserves them object by object, while metric completeness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and aggregate bridge or verification data remain absent from the packet. □

Theorem 301.195 (RegSeqRat common-tail terminal section cover)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\{T_i\}_{i\in I}$ be a finite family of certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose every pair (T_i, T_j) lands in a RegSeqRat common-tail window packet, every triple overlap restricts to one common refinement carrying the same displayed rows, and these pairwise and triple-overlap rows satisfy the common-tail cocycle conditions of Definition 301.189 and Definition 301.191. Then the family determines exactly one terminal finite section cover over the ordered faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

The visible rows of this section cover are precisely the $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source rows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *provenance. The cover contains no quotient streams, host real equality, selected limits, countable choice, ambient completeness row, metric-completeness row, aggregate* formalstatus *token, standard-bridge data, or completion coordinate outside those displayed rows.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatCommonTailTerminalSectionCover`

Proof. First apply Definition 301.183 to each pairwise packet. Each completion-facing consumer whose visible rows are the shifted reads, the common StreamName refinement, the dyadic endpoint-radius-distance ledger, the classifier and regularity ledger, the RealUp seal-source rows, componentwise hsame, displayed Cont, Pkg, and local NameCert provenance factors through the corresponding common-tail packet.

Next use Definition 301.186. Pairwise common-tail packets and finite-request $\mathsf{Real}^{\uparrow}$ seal boundaries determine one another on exactly the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows, so the local sections have the same boundary vocabulary on every overlap.

The triple-overlap compatibility is supplied by Definition 301.189 and Definition 301.191. The determinacy theorem turns agreement of the displayed rows on a triple refinement into the finite four-face cocycle condition, and the descent theorem packages the pairwise boundaries as one displayed four-face boundary whenever those cocycle rows agree.

Now compose the Real L10 terminal correspondence with pullback uniqueness. The theorems Definition 38.455 and Definition 38.458 identify the pasted StreamName, DyadicRatCore, RegSeqRat, and RealUp rows as one terminal finite boundary. Applying Definition 301.194 then keeps

the four face-status records objectwise, so the section cover does not acquire an aggregate status object while it is being pasted.

Uniqueness follows from the same pullback calculation. Any second section cover with the same visible local sections restricts to the same pairwise common-tail packets, hence to the same Real L10 terminal boundary on every four-face overlap. Since the listed packets provide no coordinates for quotient streams, host real equality, selected limits, countable choice, ambient or metric completeness, aggregate formalstatus, standard-bridge data, or hidden completion rows, there is no remaining BEDC-visible row on which two terminal section covers could differ. □

Theorem 301.196 (RegSeqRat common-tail finite-support gluing exhaustion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let* $\mathcal{P} : \mathsf{ProbeBundle}$ *be a finite bundle of certified unary observations over P served by the shared tail-compatible* $\mathsf{StreamName}^{\uparrow}$ *selector. Suppose every pair of observations in the displayed support lands in a RegSeqRat common-tail window packet, every triple overlap exposes the common-tail cocycle rows, and the pairwise packets are compatible with the finite support constructors of Definition 301.193.*

Then the finite support induction data determine exactly one terminal four-face boundary over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Its visible rows are precisely the* $\mathsf{StreamName}^{\uparrow}$ *bundle rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal-source row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support.*

This finite-support boundary has no quotient stream, selected limit, host real equality, countable-choice principle, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, aggregate status row, standard bridge datum, or completion coordinate outside those rows.*

Proof. First apply Definition 301.193 to the displayed support of $\mathcal{P}$. The empty, singleton, and union constructors give finite support acceptance for exactly the bundled observations served by the shared tail-compatible selector, with the same BHist, hsame, Cont, Pkg, and local NameCert rows on every retained overlap.

Next use Definition 126.548. The shared selector exhausts the StreamName finite bundle rows, their InBundle witnesses, the finite-window budget, and the pointwise rational-history handoff rows on one displayed StreamName surface.

The dyadic face is supplied by Definition 291.83. It reads the same finite support through the endpoint, radius, and distance ledger and places that ledger on the common four-face packet without adding a separate completion coordinate.

Now apply Definition 301.195. The accepted support, the pairwise common-tail packets, and the triple-overlap cocycle rows assemble into one terminal finite section cover over the ordered $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

Finally use Definition 38.455 together with pullback uniqueness. The Real L10 terminal correspondence identifies the section cover with the terminal four-face boundary over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and uniqueness prevents a second boundary with the same visible finite support rows.

The exclusion list follows from the row inventories of the cited support induction, StreamName exhaustion, dyadic pullback, terminal section cover, and Real L10 terminality results. None of those packets contains a quotient stream, selected limit, host real equality, countable-choice principle, metric-completeness row, ambient $\mathsf{Real}^{\uparrow}$ completeness datum, aggregate status row, standard bridge datum, or hidden completion carrier, so none is a row of this boundary. □

Theorem 301.197 (RegSeqRat common-tail terminal section Real seal-source uniqueness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\{T_i\}_{i\in I}$ be a finite family of certified cofinal tail shifts*

of the $\mathsf{StreamName}^{\uparrow}$ *schedule carried by* P*. Suppose the pairwise common-tail window packets and triple-overlap refinements satisfy the hypotheses of Definition 301.195, so that they determine a terminal finite section cover over the ordered faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Let $\mathcal{R}$ *and* $\mathcal{S}$ *be two* $\mathsf{Real}^{\uparrow}$ *seal-source reads exposed over this cover. Suppose their visible rows are the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and refinement rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the same* $\mathsf{Real}^{\uparrow}$ *seal row, and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then* $\mathcal{R}$ *and* $\mathcal{S}$ *expose the same finite* $\mathsf{Real}^{\uparrow}$ *seal-source surface.*

This uniqueness statement exports no quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness row, standard bridge data, or hidden completion carrier.

Proof. Apply Definition 301.195 to the accepted RegSeqRat packet and the finite common-tail family. The output is the terminal finite section cover whose visible vocabulary is exactly the four-face row inventory listed in the statement.

Next use Definition 301.198. Since $\mathcal{R}$ and $\mathcal{S}$ agree on the pairwise common-tail packets, triple-overlap refinements, and displayed support rows through the section cover, they determine the same BEDC-visible terminal four-face boundary.

Now apply the Real-side uniqueness results. By Definition 38.410, two finite $\mathsf{Real}^{\uparrow}$ seal-source reads with the same displayed StreamName, DyadicRatCore, RegSeqRat, Real seal, BHist, hsame, Cont, Pkg, and NameCert rows have the same finite seal-source surface. By Definition 38.692, this surface is the terminal Real name-seal source seen by any downstream finite readback through the same packet.

The final row inventory is the nonescape clause. The common-tail cover, the four-face cocycle uniqueness theorem, and the Real seal-source uniqueness theorems provide no coordinate for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric completeness, standard bridge data, or hidden completion carriers. There is therefore no remaining displayed BEDC row on which the two Real seal-source surfaces can differ. □

Theorem 301.198 (RegSeqRat terminal four-face cocycle uniqueness)**.** *Let* P *be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let* $\mathcal{U}$ *and* $\mathcal{V}$ *be two finite common-tail terminal section covers over the same ordered faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Suppose the two covers have identical pairwise common-tail window packets, identical triple-overlap cocycle rows, and the same displayed support rows* hsame*,* Cont*,* Pkg*, and* NameCert *on every finite refinement. Then* $\mathcal{U}$ *and* $\mathcal{V}$ *are the same BEDC-visible terminal four-face boundary.*

Conversely, if $\mathcal{U}$ *and* $\mathcal{V}$ *are equal as terminal four-face boundaries, then their restrictions have the same pairwise common-tail packets, the same triple-overlap cocycle rows, and the same displayed* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. This uniqueness statement exports no quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, aggregate* formalstatus *token, standard bridge data, or completion coordinate outside the displayed four-face packet.*

Proof. Assume first that the pairwise packets, triple-overlap rows, and displayed support rows of $\mathcal{U}$ and $\mathcal{V}$ agree. Apply Definition 301.195 to each cover. Both covers are terminal finite section covers over the same ordered StreamName, DyadicRatCore, RegSeqRat, and RealUp faces and with the same visible local sections.

Use Definition 301.191 on the common pairwise and triple-overlap data. The shared cocycle rows descend to one displayed four-face exit boundary carrying exactly the common $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Now apply Definition 38.455 and Definition 38.458. The Real L10 terminal pullback repacks the shared displayed packet as a single terminal boundary and forbids a second BEDC-visible boundary with the same four faces and support rows. By Definition 301.194, this identification keeps the four face-status records objectwise and adds no aggregate status coordinate.

For the converse direction, start from equality of the terminal four-face boundaries. Restrict the common boundary along each pairwise common-tail window and along each triple refinement. Terminal equality preserves exactly the displayed rows on those restrictions, so the pairwise packets, triple-overlap cocycle rows, and support rows hsame, Cont, Pkg, and NameCert agree componentwise.

The exclusions follow from the same row inventory. The terminal section cover, the cocycle descent, the objectwise face-status pullback, and the Real L10 terminal pullback provide no coordinates for quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, aggregate formalstatus, standard bridge data, or hidden completion carriers. Therefore such data cannot split two boundaries after the displayed four-face packet has been fixed, nor can it appear in the restrictions read back from an equal terminal boundary. □

Theorem 301.199 (RegSeqRat common-tail terminal section restriction exactness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\{T_i\}_{i\in I}$ be a finite family of certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose every pair (T_i, T_j) lands in a RegSeqRat common-tail window packet, every triple overlap carries the common-tail cocycle rows, and these rows determine the terminal finite section cover of Definition 301.195. For any finite subfamily $J \subseteq I$, restrict the pairwise packets and triple-overlap cocycle rows to the indices in J.*

Then the restriction of the terminal section cover on I to J is exactly the terminal finite section cover obtained by applying Definition 301.195 to the restricted pairwise and triple-overlap rows. The restricted cover has precisely the visible $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows inherited from the original family, and it introduces no quotient stream, selected limit, host real equality, countable choice, ambient completeness, metric-completeness row, aggregate* formalstatus *token, standard bridge datum, or completion coordinate outside those rows.*

Proof. Project the finite family cover along the inclusion $J \subseteq I$. The projection keeps exactly the pairwise common-tail packets indexed by pairs in J, the triple-overlap cocycle rows indexed by triples in J, and the displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows on their finite refinements.

Apply Definition 301.195 to the restricted data. The hypotheses are inherited by finite restriction: every retained pair still lands in the same common-tail window packet, every retained triple still exposes the same cocycle rows, and the row vocabulary is still the displayed four-face vocabulary of the original cover.

It remains to identify the projected cover with the independently assembled subcover. Use Definition 301.198. The projected cover and the subcover have identical restricted pairwise packets, identical restricted triple-overlap cocycle rows, and the same displayed support rows on every retained finite refinement, so they are the same BEDC-visible terminal four-face boundary.

Finally apply the row inventory from Definition 301.194 together with the Real L10 terminal correspondence and pullback uniqueness results already used in Definition 301.195. Terminal

restriction preserves the four objectwise face records and leaves no additional coordinate through which quotient streams, selected limits, host real equality, choice principles, ambient or metric completeness, aggregate status data, bridge rows, or hidden completion carriers could enter the finite subfamily cover. □

Theorem 301.200 (RegSeqRat four-face terminal section functoriality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\{T_i\}_{i\in I}$ be a finite family of certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Let $J \subseteq I$ and $K \subseteq J$ be finite request subfamilies. Suppose every pair in I lands in a RegSeqRat common-tail window packet, every triple overlap carries the common-tail cocycle rows, and these rows determine the terminal finite section cover of Definition 301.195.*

Then the terminal section cover obtained by restricting the common-tail four-face cover directly from I to K is the same finite BEDC packet as the cover obtained by first restricting from I to J and then restricting from J to K. Both routes display exactly the inherited $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, and neither route introduces quotient streams, selected limits, host real equality, countable choice, ambient completeness, aggregate* formalstatus *data, standard bridge data, or any non-displayed completion coordinate.*

Proof. Apply Definition 301.199 to the refinement $J \subseteq I$. The first restriction is exactly the terminal finite section cover assembled from the pairwise common-tail packets and triple-overlap cocycle rows whose indices lie in J.

Apply the same restriction exactness theorem to the refinement $K \subseteq J$. The second restriction keeps only the rows indexed by K and again produces the terminal section cover determined by those restricted pairwise and triple-overlap rows.

Now apply Definition 301.199 directly to $K \subseteq I$. The direct route has the same restricted pairwise packets, the same restricted triple-overlap cocycle rows, and the same displayed support rows as the two-step route.

It remains to identify the two displayed packets. Use Definition 301.198 together with the descent calculation in Definition 301.195. Since both routes expose the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows on every retained finite refinement, they are the same BEDC-visible terminal four-face boundary.

The exclusion list follows from the row inventory already used in the restriction exactness theorem. Neither direct restriction nor two-step restriction contains a coordinate for quotient streams, selected limits, host real equality, countable choice, ambient completeness, aggregate formalstatus data, standard bridge data, or a hidden completion carrier, so functoriality is witnessed entirely inside the displayed finite BEDC packet. □

Theorem 301.201 (RegSeqRat common-tail terminal section intersection exactness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Let $\{T_i\}_{i\in I}$ and $\{S_j\}_{j\in J}$ be two finite common-tail terminal section covers of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, both over the ordered faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Suppose every pairwise packet, every triple-overlap cocycle row, and every displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *support row of the first cover agrees with the corresponding row of the second cover on each retained index in the finite intersection $I \cap J$.*

Then the restriction of the first terminal section cover to $I \cap J$ and the restriction of the second terminal section cover to $I \cap J$ are the same BEDC-visible terminal four-face boundary. The common boundary is determined only by the retained common-tail packets and their displayed four-face support rows, and it contains no quotient stream, selected limit, host real equality, countable choice, ambient completeness, aggregate formalstatus *data, standard bridge row, or hidden completion carrier.*

Proof. Apply Definition 301.199 to the first cover along the inclusion $I \cap J \subseteq I$. Apply the same theorem to the second cover along $I \cap J \subseteq J$. The two restricted covers are terminal finite section covers built from exactly the pairwise common-tail packets, triple-overlap cocycle rows, and displayed support rows retained by the finite intersection.

The hypothesis identifies those retained rows component by component. Hence the two restricted covers have identical pairwise packets, identical triple-overlap cocycle rows, and the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows on every overlap that remains in $I \cap J$.

Use Definition 301.198. Since the restricted covers expose the same retained finite data, uniqueness identifies them as one BEDC-visible terminal four-face boundary.

The descent and Real terminality inputs make the identification intrinsic to the displayed packet. By Definition 301.191, the common retained pairwise packets descend to one four-face boundary. By Definition 38.455 and Definition 38.458, that descended boundary is the unique terminal pullback over the ordered StreamName, DyadicRatCore, RegSeqRat, and RealUp faces.

The exclusions follow from the same row inventory. The restriction theorem, cocycle uniqueness, cocycle descent, and Real L10 pullback uniqueness provide no coordinate for quotient streams, selected limits, host real equality, countable choice, ambient completeness, aggregate formalstatus data, standard bridge rows, or hidden completion carriers. Thus the meet boundary is exact on the retained finite intersection and nowhere else. □

Theorem 301.202 (RegSeqRat common-tail terminal four-face sheaf exactness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\{T_i\}_{i\in I}$ be a finite family of common-tail terminal section covers of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, all over the ordered faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Suppose the family carries compatible pairwise common-tail packets and triple-overlap cocycle rows on the finite intersections of the index set.*

Then those data determine a unique BEDC-visible terminal four-face boundary if and only if the covers agree on all finite intersections. The visible rows of the boundary are exactly the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *together with* BHist, hsame, Cont, Pkg, *and* NameCert. *No quotient stream, host real equality, selected limit, countable-choice principle, ambient completeness row, metric-completeness row, aggregate* formalstatus *token, standard bridge row, or hidden completion carrier is part of the glued boundary.*

Proof. Assume first that the covers agree on all finite intersections. Apply Definition 301.195 to each local common-tail family. Each local packet is a terminal finite section cover with the same displayed StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows on its finite window.

Use Definition 301.199 to pass from a cover to any retained finite subfamily, and use Definition 301.201 on every overlap of two such subfamilies. These two exactness statements identify the retained pairwise packets and triple-overlap cocycle rows on the intersections, so the local terminal boundaries have one common displayed packet wherever they meet.

The compatible local packets now have one global boundary. Apply Definition 301.198: any two terminal four-face presentations with the same pairwise packets, triple-overlap rows, and support rows are the same BEDC-visible boundary. Thus the glued packet is unique on the displayed rows.

Finally keep the glued packet on the finite Real L10 route. The theorems Definition 38.455 and Definition 38.458 repack the StreamName, DyadicRatCore, RegSeqRat, and RealUp rows as the terminal four-face pullback and forbid a second boundary with the same visible support data.

Conversely, suppose a unique BEDC-visible terminal four-face boundary is already determined by the family. Restrict that boundary to any finite intersection of covers. By Definition 301.199,

each restricted cover is the terminal finite section cover obtained from the retained pairwise and triple-overlap data.

Apply Definition 301.201 to the two restricted presentations. Since both are restrictions of the same terminal boundary, their displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows agree on the retained finite intersection.

The exclusion clauses follow from the row inventory in the cited cover, restriction, intersection, uniqueness, and Real L10 pullback results. Those results expose only the four named faces and the listed support rows, so quotient streams, host real equality, selected limits, countable choice, ambient or metric completeness, aggregate status data, standard bridge rows, and hidden completion carriers cannot appear in the sheaf boundary or distinguish two glued presentations. □

Definition 301.203 (RegSeqRat common-tail descent atlas)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. A RegSeqRat common-tail descent atlas for P is a finite family $\mathcal{A} = \{U_i\}_{i\in I}$ of common-tail terminal section covers of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, together with the following BEDC-visible rows:*

1. *for every retained pair (i, j), a RegSeqRat common-tail window packet on $U_i \cap U_j$ with the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces;*
2. *for every retained triple (i, j, k), the triple-overlap cocycle rows asserting that the three pairwise packets restrict to the same common-tail boundary on $U_i \cap U_j \cap U_k$;*
3. *restriction maps from each U_i to every finite subfamily and intersection agreements for every finite meet of subfamilies;*
4. *support rows* BHist, hsame, Cont, Pkg, *and* NameCert *witnessing that all local faces, pairwise packets, triple rows, restrictions, and intersection agreements are carried by the same finite BEDC packet.*

The atlas is accepted when each local cover is terminal in the sense of Definition 301.195, each finite restriction satisfies Definition 301.199, and each finite meet satisfies Definition 301.201. The atlas contains no quotient stream, host real equality, selected limit, countable-choice principle, ambient completeness row, metric-completeness row, aggregate formalstatus *token, standard bridge row, or hidden completion carrier outside the displayed four-face and support rows.*

Theorem 301.204 (RegSeqRat common-tail descent atlas Real seal exactness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A} = \{U_i\}_{i\in I}$ be an accepted RegSeqRat common-tail descent atlas for P. Then $\mathcal{A}$ determines a unique Real seal-visible terminal four-face boundary over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *if and only if its finite section covers agree on all finite intersections.*

The boundary displays exactly the atlas rows BHist, hsame, Cont, Pkg, *and* NameCert *together with the four named faces. It exports no quotient stream, host real equality, selected limit, countable-choice principle, ambient completeness row, metric-completeness row, aggregate* formalstatus *token, standard bridge row, or hidden completion carrier.*

Proof. Assume first that the finite section covers in $\mathcal{A}$ agree on all finite intersections. By the atlas acceptance clauses and Definition 301.195, every U_i supplies a terminal common-tail section cover over the same ordered StreamName, DyadicRatCore, RegSeqRat, and RealUp faces. Its support rows are the displayed BHist, hsame, Cont, Pkg, and NameCert rows of the atlas.

Use Definition 301.199 for finite subfamilies and Definition 301.201 for finite meets. These two exactness results identify the pairwise packets, triple-overlap cocycle rows, restriction maps, and intersection agreements on every retained overlap.

Now apply Definition 301.202. The compatible finite atlas rows glue to one BEDC-visible terminal four-face boundary, and the same theorem gives uniqueness among terminal presentations with the displayed rows.

Finally pass from the RegSeqRat common-tail boundary to the Real seal-visible boundary. The Real L10 terminal four-face correspondence and pullback uniqueness theorems, Definition 38.455 and Definition 38.458, identify the glued StreamName, DyadicRatCore, RegSeqRat, and RealUp packet with the unique terminal Real seal boundary seen by downstream Phase Real and Completion reads.

Conversely, suppose $\mathcal{A}$ determines a unique Real seal-visible terminal four-face boundary. Restrict that boundary to any finite intersection of atlas covers. By Definition 301.199, the restriction is the terminal finite section cover assembled from the retained pairwise packets and triple-overlap cocycle rows.

Apply Definition 301.201 to any two such restricted presentations. Since both are restrictions of the same Real seal-visible terminal boundary, their displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows agree on the finite meet.

The nonescape statement follows from the same inventory. The atlas definition, the terminal cover theorem, the restriction and intersection exactness theorems, the sheaf exactness theorem, and the Real L10 terminal pullback provide no coordinate outside the named faces and support rows. Hence no quotient stream, host real equality, selected limit, choice principle, ambient or metric completeness row, aggregate status token, standard bridge row, or hidden completion carrier can distinguish two descended Real seal boundaries. □

Theorem 301.205 (RegSeqRat common-tail descent atlas finite-request coverage)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A} = \{U_i\}_{i\in I}$ be an accepted RegSeqRat common-tail descent atlas for P. Every finite-request* $\mathsf{Real}^{\uparrow}$ *seal consumer whose visible rows are carried by $\mathcal{A}$ factors through a unique atlas member or finite intersection. The pullback exposed to that consumer is exactly the four faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

No quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, metric-completeness row, aggregate formalstatus *token, standard bridge row, or hidden completion carrier is exported by this finite-request factorization.*

Proof. Let C be a finite-request $\mathsf{Real}^{\uparrow}$ seal consumer whose visible rows are carried by $\mathcal{A}$. The request reads only finitely many displayed rows, so its support is contained in one retained atlas member or in a finite meet of retained members.

If the support is contained in one member U_i, apply Definition 301.199. The consumer factors through the terminal section cover on U_i, and terminality gives uniqueness of the factorization over the displayed StreamName, DyadicRatCore, RegSeqRat, and RealUp rows.

If the support uses several members, apply Definition 301.201. The finite meet carries the same retained pairwise packets, triple-overlap cocycle rows, and support rows, so the request factors through that finite intersection with a unique displayed pullback.

Now use Definition 301.184. It identifies the local member or intersection factorization with the finite $\mathsf{Real}^{\uparrow}$ seal pullback seen by C, and it exposes exactly the four named faces together with BHist, hsame, Cont, Pkg, and NameCert.

The local pullbacks glue to one atlas boundary by Definition 301.202. Finally, Definition 301.204 identifies this glued boundary with the unique Real seal-visible terminal boundary of the accepted atlas. Therefore every finite request carried by $\mathcal{A}$ is covered by a unique member or finite-intersection factorization.

The exclusion statement follows from the same row inventory. The cited pullback, restriction, intersection, sheaf, and atlas exactness theorems contain no coordinate for quotient streams, host real equality, selected limits, countable choice, ambient or metric completeness, aggregate status data, standard bridge data, or hidden completion carriers. □

Theorem 301.206 (RegSeqRat common-tail filtered colimit normal form)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A} = \{U_i\}_{i\in I}$ be a finite directed atlas of accepted common-tail windows for P. Suppose every refinement map in $\mathcal{A}$ preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Then the filtered colimit read of $\mathcal{A}$ has a unique finite BEDC normal form over exactly the four faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with those support rows.*

The normal form is the retained finite atlas display: each finite consumer factors through one retained common-tail member or retained finite intersection, and two such displays are equal precisely when their four face rows and support rows agree there. It exports no quotient stream equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First reduce the directed atlas to its finite support. Apply Definition 301.193 to the retained common-tail observations. The empty, singleton, and union constructors keep every schedule row, dyadic endpoint-radius ledger, RegSeqRat classifier and readback row, local Real seal row, and support row inside the same finite BEDC packet.

Next glue the retained windows. The finite-support gluing exhaustion theorem Definition 301.196 says that pairwise packets, triple-overlap cocycle rows, and compatible support rows assemble to one finite four-face boundary. The descent-atlas coverage theorem Definition 301.205 then places every finite $\mathsf{Real}^{\uparrow}$ seal request on a retained member or retained finite intersection.

Now identify the four faces read by the L10 route. Use Definition 301.69 to keep the StreamName schedule, DyadicRatCore ledger, RegSeqRat readback, and Real seal rows on the same handoff surface. The Real L10 terminal correspondence and pullback uniqueness theorems, Definition 38.455 and Definition 38.458, repack that surface as the unique terminal finite boundary over the four named faces.

It remains to compare two proposed colimit reads. Restrict both reads to any retained member or retained finite intersection. By finite-request coverage they have the same visible local factorization, and by the terminal pullback uniqueness theorem a second normal form with the same displayed rows has no BEDC coordinate on which to differ. Hence the colimit read is the unique finite normal form carried by the atlas.

The excluded data have no route into the construction. The cited support induction, finite-support gluing, descent coverage, handoff exactness, and Real terminal boundary theorems expose only the four face rows and the shared BHist, hsame, Cont, Pkg, and NameCert support. Quotient stream equality, selected limits, countable choice, metric completeness, aggregate status records, aggregate bridge records, and ambient $\mathsf{Real}^{\uparrow}$ completeness therefore cannot be exported by the normal form. □

Theorem 301.207 (RegSeqRat common-tail descent atlas classifier completeness)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A} = \{U_i\}_{i\in I}$ be an accepted RegSeqRat common-tail descent atlas for P. For any two finite* $\mathsf{Real}^{\uparrow}$ *seal consumers visible through $\mathcal{A}$, the consumers determine the same Real seal-visible terminal boundary exactly when their displayed* RegSeqRatClassifier *rows agree on every retained finite section and every retained finite intersection of the atlas.*

The complete finite witness for that boundary is precisely the atlas display over the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. No quotient stream, host real equality, selected limit, choice principle, ambient completeness row, metric-completeness row, standard bridge row, or hidden completion carrier is exported by the classifier-complete Real seal boundary.*

Proof. Assume first that the two visible $\mathsf{Real}^{\uparrow}$ seal consumers determine the same Real seal-visible terminal boundary through $\mathcal{A}$. Apply Definition 301.204. The shared boundary restricts to the same accepted finite section cover on every retained section and every finite intersection of the atlas.

Use Definition 301.202. Since the restricted presentations are one terminal four-face boundary, their displayed StreamName, DyadicRatCore, RegSeqRat, and RealUp rows agree on each retained finite meet. Applying Definition 301.182 on those common-tail windows identifies the transported RegSeqRatClassifier rows seen by the two consumers.

Conversely, suppose the displayed RegSeqRatClassifier rows of the two consumers agree on every retained finite section and finite intersection of $\mathcal{A}$. The bridge theorem Definition 301.182 turns each common-tail agreement into one shared classifier and seal-source surface on the corresponding finite window.

Apply Definition 301.202. The matching classifier rows, together with the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert support rows, glue to one BEDC-visible terminal four-face boundary. Then Definition 301.204 identifies that glued boundary with the unique Real seal-visible terminal boundary determined by the accepted atlas.

Finally use the Real L10 terminal correspondence and pullback uniqueness theorems, Definition 38.455 and Definition 38.458. They say that a terminal presentation over the ordered StreamName, DyadicRatCore, RegSeqRat, and RealUp faces is uniquely read as the Real seal boundary. Thus the classifier rows are complete for the finite boundary witnessed by $\mathcal{A}$.

The nonescape clause is just the row inventory of the construction. The cited atlas exactness, classifier bridge, sheaf exactness, and Real L10 terminal pullback results expose only the four named faces and the listed support rows, so the excluded quotient, equality, limit, choice, completeness, bridge, and hidden-carrier data have no displayed coordinate in the classifier-complete boundary. □

Theorem 301.208 (RegSeqRat common-tail descent atlas boundary naturality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ and $\mathcal{B}$ be accepted RegSeqRat common-tail descent atlases for P. Suppose a finite boundary-preserving atlas morphism sends every retained member, retained pairwise packet, retained triple-overlap cocycle row, and displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *support row of $\mathcal{A}$ to the corresponding displayed row of $\mathcal{B}$. Then the Real seal-visible terminal boundary induced by $\mathcal{A}$ maps to the Real seal-visible terminal boundary induced by $\mathcal{B}$.*

Conversely, for any such finite atlas morphism, equality of the induced Real seal-visible terminal boundary forces agreement of the displayed RegSeqRatClassifier *rows on every retained member and every retained finite intersection of $\mathcal{A}$ and $\mathcal{B}$.*

No quotient stream equality, selected limit, host real equality, countable-choice principle, ambient completeness row, metric-completeness row, aggregate status row, standard bridge row, or hidden completion carrier is exported or used by this naturality statement.

Proof. First consider the forward transport. By Definition 301.204, the Real seal-visible terminal boundary of each accepted atlas is identified by its displayed common-tail section covers, finite

intersections, and four named faces. The morphism preserves exactly those retained rows, so it carries the boundary presentation of $\mathcal{A}$ to a boundary presentation over the displayed rows of $\mathcal{B}$.

Use Definition 301.205 to factor every finite Real seal request through a retained member or retained finite intersection. On those factors the morphism is already row-preserving by hypothesis. The restriction and intersection exactness theorems, Definition 301.199 and Definition 301.201, keep the transported packet inside the same finite L10 display.

The transported local packets glue in $\mathcal{B}$ by Definition 301.202. Since the glued packet has the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows of $\mathcal{B}$, another application of Definition 301.204 identifies it with the Real seal-visible terminal boundary induced by $\mathcal{B}$.

Now assume the induced Real seal-visible terminal boundaries are equal along the same morphism. Restrict the equality to any retained member or retained finite intersection. The finite-request coverage theorem places that restriction in the displayed finite section cover, and the restriction and intersection exactness theorems identify the common-tail window on which both atlas presentations are compared.

Apply Definition 301.207. Equality of the Real seal-visible terminal boundary on each retained finite window recovers equality of the displayed $\mathsf{RegSeqRatClassifier}$ rows there. Hence the morphism forces classifier agreement on every retained member and every retained finite intersection.

The nonescape clause follows from the same inventory. The atlas definition, the Real seal exactness theorem, finite-request coverage, classifier completeness, restriction and intersection exactness, and four-face sheaf exactness expose only the four named faces and the listed support rows. Therefore quotient stream equality, selected limits, host real equality, countable choice, ambient or metric completeness, aggregate status rows, standard bridge rows, and hidden completion carriers have no coordinate through which this naturality result can factor. □

Theorem 301.209 (RegSeqRat audit-threshold atlas naturality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ and $\mathcal{B}$ be accepted RegSeqRat common-tail descent atlases for P. Suppose a finite boundary-preserving atlas morphism sends every retained member, retained pairwise packet, retained triple-overlap cocycle row, and displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support row of $\mathcal{A}$ to the corresponding displayed row of $\mathcal{B}$. Then the morphism transports the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold coordinate induced by $\mathcal{A}$ to the audit-threshold coordinate induced by $\mathcal{B}$.*

Conversely, for any such finite atlas morphism, equality of the induced Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold coordinates forces agreement of the displayed* $\mathsf{RegSeqRatClassifier}$ *rows on every retained member and every retained finite intersection of $\mathcal{A}$ and $\mathcal{B}$.*

No quotient stream, selected limit, host real equality, countable-choice principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness row, aggregate* $\mathsf{formalstatus}$ *row, aggregate bridge row, or hidden completion carrier is exported or used by this audit-threshold naturality statement.*

Proof. First transport the underlying atlas boundary. Apply Definition 301.208. The finite morphism preserves the retained members, finite intersections, four displayed faces, and support rows, so it carries the Real seal-visible terminal boundary induced by $\mathcal{A}$ to the one induced by $\mathcal{B}$.

Next read the audit-threshold coordinate on the transported boundary. Apply Definition 301.219. The Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read is the RegSeqRat four-face exit-threshold equivalence together with the local closurestatus and formalstatus records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

It remains to identify those local records after transport. Use Definition 126.509 for the StreamName-side threshold readback, Definition 38.471 for the Phase $\mathsf{Real}^{\uparrow}$ and Completion audit

pullback, and Definition 291.107 for the shared NameCert interface over BHist, hsame, Cont, Pkg, and NameCert. Since the morphism preserves each of these displayed rows, the audit-threshold coordinate read from $\mathcal{A}$ is the coordinate read from $\mathcal{B}$.

For the converse, assume the induced audit-threshold coordinates are equal along the same finite morphism. The factorization theorem reads both coordinates through the same RegSeqRat four-face exit-threshold surface and the same four local face records, while the StreamName readback, Real audit pullback, and DyadicRatCore interface theorem keep the support rows objectwise.

Apply Definition 301.208 to the boundary left by these equal threshold reads. Its converse clause recovers agreement of the displayed RegSeqRatClassifier rows on every retained member and every retained finite intersection.

The nonescape clause is the row inventory of the construction. The cited boundary naturality, audit-threshold factorization, StreamName readback, Real audit pullback, and DyadicRatCore interface theorems expose only the four named faces and the listed support rows. Therefore quotient streams, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, aggregate status data, aggregate bridge data, and hidden completion carriers have no coordinate through which this naturality result can factor. □

Theorem 301.210 (RegSeqRat common-tail atlas finality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ be an accepted common-tail descent atlas for P in the sense of Definition 301.203. Let $\mathcal{T}$ be any finite L10 terminal presentation over the same four faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$. *Suppose every finite* $\mathsf{Real}^{\uparrow}$ *seal request of $\mathcal{T}$ factors through a retained member or retained finite intersection of $\mathcal{A}$, and suppose the displayed* RegSeqRatClassifier *rows of $\mathcal{T}$ agree with the rows of $\mathcal{A}$ on all such overlaps. Then there is a unique boundary-preserving morphism from $\mathcal{T}$ into $\mathcal{A}$.*

Equivalently, $\mathcal{A}$ is terminal among finite common-tail presentations inducing the same Real seal-visible boundary and the same Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold coordinate. This terminal surface exports no quotient streams, selected limits, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness rows, aggregate* formalstatus *rows, aggregate bridge rows, or hidden completion carriers.*

Proof. First build the local comparison map. By Definition 301.205, each finite $\mathsf{Real}^{\uparrow}$ seal request of $\mathcal{T}$ is read inside a retained member or retained finite intersection of $\mathcal{A}$. The hypotheses therefore place every displayed StreamName, DyadicRatCore, RegSeqRat, local RealUp, BHist, hsame, Cont, Pkg, and NameCert row of $\mathcal{T}$ on a retained atlas window.

On each retained window, apply Definition 301.207. Agreement of the displayed classifier rows on overlaps identifies the Real seal-visible boundary carried by $\mathcal{T}$ with the boundary of the corresponding retained part of $\mathcal{A}$. The Real seal exactness theorem Definition 301.204 then supplies the unique local factorization into the accepted atlas boundary.

The local factorizations glue because they agree on every retained finite intersection. This is exactly the boundary-preserving content of Definition 301.208: a finite atlas morphism preserving the retained rows transports the Real seal-visible terminal boundary and, conversely, boundary equality recovers classifier agreement on the overlaps. Hence the locally defined comparison maps assemble into one boundary-preserving morphism $\mathcal{T} \to \mathcal{A}$.

Uniqueness follows by applying the same two readings in reverse. Any boundary-preserving morphism from $\mathcal{T}$ into $\mathcal{A}$ has the same finite $\mathsf{Real}^{\uparrow}$ seal requests by Definition 301.204; it has the same overlap classifier rows by Definition 301.207; and it has the same Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate by Definition 301.209. Thus no second morphism can differ on any retained member, finite intersection, boundary row, or audit-threshold coordinate.

The nonescape clause is the displayed row inventory. The atlas definition and the cited exactness, coverage, classifier, boundary-naturality, and audit-threshold-naturality theorems expose only the four named faces and the support rows BHist, hsame, Cont, Pkg, and NameCert. There is no coordinate for quotient streams, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness rows, aggregate formal-status rows, aggregate bridge rows, or hidden completion carriers. □

Theorem 301.211 (RegSeqRat common-tail atlas refinement invariance)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let $\mathcal{A}$ be an accepted common-tail descent atlas for P, and let $\mathcal{B}$ be a finite certified refinement of $\mathcal{A}$. Suppose the refinement preserves every retained member of $\mathcal{A}$ by a finite cover in $\mathcal{B}$, preserves the retained pairwise packets and triple-overlap cocycle rows, and preserves the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then the Real seal-visible terminal boundary induced by $\mathcal{B}$ factors through the boundary induced by $\mathcal{A}$, and the induced Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold coordinate of $\mathcal{B}$ agrees with the coordinate induced by $\mathcal{A}$ under this factorization.*

Conversely, any finite certified refinement with the same retained-row preservation and with the same induced Real seal-visible terminal boundary and audit-threshold coordinate is already equivalent to $\mathcal{A}$ on every retained common-tail member, pairwise packet, and triple-overlap cocycle row.

This refinement-invariance surface exports no quotient streams, selected limits, host equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness rows, aggregate* formalstatus *rows, aggregate bridge rows, or hidden completion carriers.*

Proof. First identify the support packet visible to the refinement. Apply Definition 126.662: the refined finite cover cannot introduce a second $\mathsf{StreamName}^{\uparrow}$ support surface, so every retained StreamName coordinate of $\mathcal{B}$ reads through the support packet already displayed by $\mathcal{A}$.

Next perform the finite-cover comparison on the RegSeqRat and Real faces. The refinement hypotheses give exactly the retained members, pairwise packets, triple-overlap cocycle rows, and support rows required by Definition 301.64. Applying Definition 301.92 then identifies the refined Real seal-visible terminal boundary with the boundary obtained from $\mathcal{A}$ through the certified finite cover.

Now pass from finite covers to atlas boundaries. Use Definition 301.208 to view the refinement as a boundary-preserving atlas morphism. By Definition 301.210, the resulting finite common-tail presentation has the unique factorization into $\mathcal{A}$, so the boundary induced by $\mathcal{B}$ factors through the boundary induced by $\mathcal{A}$.

It remains to compare the audit-threshold coordinate. Apply Definition 301.209 to the same boundary-preserving refinement morphism. Since the refinement preserves the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows, the Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate read from $\mathcal{B}$ is the one read from $\mathcal{A}$.

For the converse, assume the refined atlas has the same boundary and audit-threshold coordinate under the retained-row preservation hypotheses. The converse clauses of boundary naturality and audit-threshold atlas naturality recover agreement of the displayed RegSeqRatClassifier rows on each retained member and finite intersection. The retained pairwise packets and triple-overlap cocycle rows are therefore the same certified finite data as those of $\mathcal{A}$.

The nonescape statement is again a row inventory. The support-minimality, finite-cover refinement, Real-seal refinement, boundary-naturality, atlas-finality, and audit-threshold-naturality theorems use only finite retained covers and the displayed support rows, leaving no coordinate for

quotient streams, selected limits, host equality, choice, ambient or metric completeness, aggregate status data, aggregate bridge data, or hidden completion carriers. □

Theorem 301.212 (RegSeqRat descent atlas audit-threshold fusion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ be an accepted common-tail descent atlas for P in the sense of Definition 301.203. The atlas presents the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read if and only if the following three conditions hold:*

1. *its finite section covers satisfy common-tail terminal four-face sheaf exactness;*
2. *its displayed* $\mathsf{RegSeqRatClassifier}$ *rows satisfy descent-atlas classifier completeness on every retained member and finite intersection;*
3. *its Real seal-visible boundary is final among finite common-tail presentations inducing the same four local face records.*

When these conditions hold, the audit read is exhausted exactly by the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$, *together with* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

No quotient stream, selected limit, host equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness row, aggregate* $\mathsf{formalstatus}$ *row, aggregate bridge row, or hidden completion carrier is part of the presented audit-threshold read.*

Proof. For the forward direction, assume that $\mathcal{A}$ presents the current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. Apply Definition 301.219. The threshold read factors through the RegSeqRat four-face exit surface, so the only local face records available to the presentation are the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ records.

The exactness and classifier clauses are then read back on the atlas itself. Use Definition 301.209 to transport the threshold coordinate through boundary-preserving atlas maps, and use Definition 301.210 to compare any finite common-tail presentation with $\mathcal{A}$. The StreamName readback, Real audit pullback, and DyadicRatCore interface theorems, Definition 126.509, Definition 38.471, and Definition 291.107, identify the support rows as BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

It remains, in the forward direction, to record that the three displayed conditions are precisely the atlas witnesses used by the presentation. The sheaf condition is Definition 301.202; the classifier condition is Definition 301.207; and the finality condition is Definition 301.210. Thus a presentation of the audit-threshold read exposes all three witnesses and no additional carrier.

Conversely, assume the three conditions. Common-tail terminal four-face sheaf exactness glues the finite section covers to one BEDC-visible terminal boundary over the four named faces. Classifier completeness identifies the same Real seal-visible boundary for every finite consumer whose displayed classifier rows agree on the retained members and finite intersections.

Now apply atlas finality. Any finite common-tail presentation inducing the same four local face records has a unique boundary-preserving morphism into $\mathcal{A}$. Audit-threshold atlas naturality then identifies the Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate along that unique morphism, so the atlas carries a single objectwise audit read.

Finally apply the StreamName, Real, and Dyadic L10 interface theorems cited above. They read the unique threshold coordinate through exactly the four local faces and the support rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since none of the cited exactness, classifier, finality, naturality, or L10 pullback results exports quotient streams, selected limits, host equality, countable choice, ambient or metric completeness, aggregate status data, aggregate bridge data, or hidden completion carriers, the presented read is exhausted by the displayed rows. □

Theorem 301.213 (RegSeqRat common-tail descent atlas support minimality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ be an accepted common-tail descent atlas for P in the sense of Definition 301.203. Suppose $\mathcal{A}$ presents the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read. If one deletes any retained member, retained pairwise packet, retained triple-overlap cocycle row, or any displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *or* $\mathsf{NameCert}$ *support row, the remaining finite surface no longer presents that audit-threshold read.*

Conversely, any accepted common-tail descent atlas carrying all retained members, retained pairwise packets, retained triple-overlap cocycle rows, all displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *local* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows, and the three fusion conditions of Definition 301.212, presents the same current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read.*

This support-minimality statement exports no quotient streams, selected limits, host equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness rows, aggregate* $\mathsf{formalstatus}$ *rows, aggregate bridge rows, or hidden completion carriers.*

Proof. First read the accepted atlas through the fusion theorem. By Definition 301.212, presenting the audit-threshold read is equivalent to the common-tail terminal four-face sheaf condition, descent-atlas classifier completeness, and finality among finite common-tail presentations with the same four local face records. The same theorem identifies the displayed support as exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Now delete one retained member, retained pairwise packet, or retained triple-overlap cocycle row. The finite-support gluing exhaustion theorem Definition 301.196 says that every finite Real seal request is exhausted by retained members and finite intersections of $\mathcal{A}$. Removing one of those rows leaves a request or overlap without its displayed factorization, so the sheaf and classifier clauses required by the fusion theorem can no longer be witnessed.

If instead one deletes one displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, or $\mathsf{NameCert}$ support row, apply Definition 301.209. Audit-threshold transport is defined only along boundary-preserving atlas morphisms that preserve precisely those rows. With any one of them absent, the transported Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate has no objectwise support coordinate, so the remaining surface cannot be the same presented read.

Finality gives the necessity conclusion. By Definition 301.210, every finite common-tail presentation of the same boundary and audit-threshold coordinate factors uniquely through $\mathcal{A}$. A surface obtained by deleting a retained row would either omit a finite request required by gluing exhaustion or fail to preserve a support coordinate required by naturality. Such a surface therefore cannot receive the unique final comparison as a presentation of the same read.

For the converse, assume an accepted atlas keeps all listed rows and satisfies the three fusion conditions. Apply Definition 301.212 to obtain a presented audit-threshold read. Then use Definition 301.211 to compare it with $\mathcal{A}$ along the retained-row refinement surface, and use Definition 301.209 to transport the coordinate. The result is the same current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read.

The exclusions are part of the same inventory. The proof uses only threshold fusion, atlas finality, refinement invariance, audit-threshold naturality, and finite-support gluing exhaustion; none of these results supplies a coordinate for quotient streams, selected limits, host equality, countable choice, ambient or metric completeness, aggregate status data, aggregate bridge data, or hidden completion carriers. □

Theorem 301.214 (RegSeqRat terminal support carrier inversion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $\mathcal{A}$ be an accepted common-tail descent atlas for P in the sense of Definition 301.203. Suppose $\mathcal{A}$ presents the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read. Then every downstream L10 read of the Real seal-visible terminal boundary presented by $\mathcal{A}$, when read as the same Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold coordinate, projects exactly the four carrier and classifier faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$, *together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

Conversely, if two downstream L10 reads of that presented terminal boundary have the same projected $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and local* $\mathsf{Real}^{\uparrow}$ *face rows and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows, then they determine the same Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read.*

The carrier-inversion surface exports no quotient streams, selected limits, host equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness rows, aggregate* formalstatus *rows, aggregate bridge rows, or hidden completion carriers.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatCommonTailTerminalSupportCarrierInversion`

Proof. First read the terminal boundary through the fusion theorem. By Definition 301.212, the hypothesis that $\mathcal{A}$ presents the audit-threshold read is equivalent to the common-tail terminal four-face sheaf condition, descent-atlas classifier completeness, and atlas finality over the same four local face records. The same theorem identifies the complete support inventory as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Now apply support minimality. Theorem 301.213 says that deleting any retained atlas row or displayed support row destroys the presented audit-threshold read. Hence a downstream L10 read of the same presented boundary cannot project through a smaller carrier surface, and any visible carrier row it uses must already be one of the listed rows.

It remains to identify the readback as objectwise, not merely by absence. Use Definition 301.210: every finite common-tail presentation of the same Real seal-visible boundary and audit-threshold coordinate factors uniquely into $\mathcal{A}$. Then apply Definition 301.209 to the resulting boundary-preserving comparison. The StreamName readback, Real audit pullback, and DyadicRatCore interface theorems, Definition 126.509, Definition 38.471, and Definition 291.107, read that comparison through exactly the four face rows and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the converse, take two downstream reads with the same projected face rows and support rows. Atlas finality gives each read the same unique comparison into $\mathcal{A}$, because there is no retained member, finite intersection, boundary row, or support coordinate on which the comparisons can differ. Audit-threshold atlas naturality then transports both reads to the same Phase $\mathsf{Real}^{\uparrow}$ and Completion coordinate.

The exclusions follow from the same support inventory. The fusion, support-minimality, finality, naturality, StreamName readback, Real pullback, and Dyadic interface results cited above expose no coordinate for quotient streams, selected limits, host equality, countable choice, ambient or metric completeness, aggregate status rows, aggregate bridge rows, or hidden completion carriers, so no downstream readback can recover such a carrier from the terminal atlas boundary. □

Theorem 301.215 (RegSeqRat local-threshold exit descent)**.** *Let $\mathcal{E}$ be an accepted objectwise Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit-threshold read whose public classifier coordinate is read on the same visible terminal packet. Suppose the visible source of $\mathcal{E}$ is an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal four-face packet with rows over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the local* $\mathsf{Real}^{\uparrow}$ *seal. Then the exit-threshold coordinate descends to that same finite source record.*

The descended record carries exactly the $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The descent uses only the displayed finite BHist *packet, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *support. It exports no quotient streams, selected limits, host real equality, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, aggregate* formalstatus *row, or aggregate bridge row.*

Proof. First read the threshold and classifier coordinates from one visible packet. Apply Definition 38.789. The objectwise exit-threshold read and the public classifier coordinate are two readings of the same finite four-face record over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, local $\mathsf{Real}^{\uparrow}$, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next identify the terminal pullback. Apply Definition 38.458 to the accepted Phase Real and Completion terminal read. The terminal boundary has only one pullback packet with those four visible faces and those support rows, so the exit-threshold coordinate cannot be read from a second completion-facing source.

Now descend through the RegSeqRat side. Use Definition 301.237. Since the visible source is an accepted $\mathsf{RegSeqRat}^{\uparrow}$ terminal four-face packet and the StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and local NameCert rows agree with the terminal pullback, the RegSeqRat source section is exactly the same finite record.

Finally compare the common-tail boundary rows. The common-tail bridge and terminal section results, Definition 301.186, Definition 301.195, and Definition 301.202, keep the RegSeqRat regularity and common-tail rows on the same four-face surface. The four-face exactness theorem Definition 301.239 then reads the local $\mathsf{Real}^{\uparrow}$ seal from that displayed packet.

Thus the threshold coordinate descends to the same finite record over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. None of the cited packets contains a coordinate for quotient streams, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, aggregate verification status, or bridge rows, so no such data can appear in the descended source. □

Theorem 301.216 (RegSeqRat local-threshold four-face sufficiency)**.** *Let* $\mathcal{R}$ *be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record. Suppose* $\mathcal{R}$ *carries the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the displayed support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

If these rows satisfy the four-face exactness criterion of Definition 301.239, then the objectwise Phase $\mathsf{Real}^{\uparrow}$ *and Completion exit-threshold coordinate is accepted on the same record* $\mathcal{R}$*.*

The accepted coordinate exports no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness data, aggregate formalstatus *row, aggregate bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the displayed four-face data on $\mathcal{R}$. Apply Definition 301.239. The StreamName finite window, DyadicRatCore endpoint and radius ledger, RegSeqRat classifier and common-tail face, and local Real rows therefore present a Real terminal seal on exactly the same finite source record.

Next read the classifier and threshold coordinates from that terminal seal. Apply Definition 38.789. The public classifier coordinate and the objectwise exit-threshold coordinate are two visible readings of the same four-face packet, with support only from the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally compare the accepted threshold read with the original RegSeqRat source. Use Definition 301.215. Since the visible source already is $\mathcal{R}$ and the terminal four-face exactness step supplies the local Real seal on $\mathcal{R}$, the descended source of the accepted exit-threshold coordinate is $\mathcal{R}$ itself.

The construction has used only the finite rows named in the statement. No cited component introduces quotient streams, selected limits, host real equality, countable choice, metric-completeness structure, aggregate verification status, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the accepted coordinate contains none of those excluded fields. □

Theorem 301.217 (RegSeqRat local-threshold terminal-window fusion)**.** *Let T be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source presented to the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit. An accepted local-threshold coordinate for the Phase* $\mathsf{Real}^{\uparrow}$ *and Completion read exists on T if and only if T carries one fused terminal-window packet whose visible faces are*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and whose support is only

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Under this equivalence, the local-threshold coordinate and the fused terminal-window packet determine the same BEDC-visible terminal source.

The fused packet refuses quotient streams, host real equality, selected limits, countable choice, metric-completeness data, aggregate status rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, start with an accepted Phase $\mathsf{Real}^{\uparrow}$ and Completion local-threshold coordinate on the finite source T. Apply Definition 301.215. The coordinate descends to the same RegSeqRat terminal source and exposes only the StreamName, DyadicRatCore, RegSeqRat, local Real, and displayed support rows named in the statement.

Next apply Definition 301.239. The descended source is therefore the exact four-face terminal surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Then use Definition 126.626 to repack that surface as one fused terminal-window packet without changing the source T.

Finally apply Definition 291.107. It identifies the fused four-face packet with the local NameCert interface consumed by the L10 exit, so the packet carries precisely the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

For the reverse direction, assume T carries one fused terminal-window packet over the four displayed faces and the five displayed support rows. The source-exhaustion theorem Definition 126.626 reads this packet as the same public finite-request section, and Definition 291.107 reads it as the local NameCert interface for the Phase $\mathsf{Real}^{\uparrow}$ and Completion exit.

Now apply Definition 301.239 to obtain the local $\mathsf{Real}^{\uparrow}$ terminal seal on T. With that exact four-face surface in place, Definition 301.216 supplies the accepted local-threshold coordinate on the same source.

The non-escape clause is inherited in both directions. The descent theorem, the four-face sufficiency theorem, the RegSeqRat terminal exactness theorem, the StreamName source-exhaustion

theorem, and the DyadicRatCore local NameCert interface all expose only finite BHist, hsame, Cont, Pkg, and NameCert support. None supplies a coordinate for quotient streams, host real equality, selected limits, countable choice, metric-completeness data, aggregate status rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the equivalence cannot export any of those fields. □

Theorem 301.218 (RegSeqRat four-face exit-threshold equivalence)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record whose visible rows are exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support, and no other completion-facing coordinates. Then $\mathcal{R}$ satisfies the real terminal four-face exactness criterion of Definition 301.239 if and only if the objectwise Phase* $\mathsf{Real}^{\uparrow}$ *and Completion exit-threshold coordinate is accepted on the same record $\mathcal{R}$.*

The equivalence is terminal over the displayed four faces: it exports no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness row, aggregate formalstatus *row, aggregate bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, assume the displayed rows of $\mathcal{R}$ satisfy the four-face exactness criterion. Apply Definition 301.216. Since the source record already has exactly the StreamName finite-window rows, DyadicRatCore endpoint, radius, distance, and regularity rows, RegSeqRat classifier and common-tail rows, local Real seal rows, and the displayed support rows, the objectwise Phase $\mathsf{Real}^{\uparrow}$ and Completion exit-threshold coordinate is accepted on $\mathcal{R}$ itself.

For the reverse direction, assume the objectwise Phase $\mathsf{Real}^{\uparrow}$ and Completion exit-threshold coordinate is accepted on $\mathcal{R}$. Apply Definition 301.215 to descend that accepted coordinate to the same finite RegSeqRat terminal source. The descent identifies the accepted source with the four visible faces and the same BHist, hsame, Cont, Pkg, and NameCert support carried by $\mathcal{R}$.

It remains to read the descended source as the terminal four-face record. Use Definition 38.796 to identify the accepted exit coordinate with the objectwise product of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and local $\mathsf{Real}^{\uparrow}$ threshold records. Then apply Definition 301.239 to read those four rows as the RegSeqRat real terminal four-face exactness certificate on $\mathcal{R}$.

All steps are performed on the finite source record named in the statement. The cited criteria mention only the displayed four faces and support rows, so quotient streams, selected limits, countable choice, metric-completeness rows, aggregate verification or bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are excluded from both directions. □

Theorem 301.219 (RegSeqRat four-face audit-threshold factorization)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record whose exposed rows are exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then $\mathcal{R}$ satisfies the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read exactly when the RegSeqRat four-face exit-threshold equivalence of Definition 301.218 holds on $\mathcal{R}$ and each of the four face chapters* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *supplies its own local closurestatus and formalstatus record at the roadmap threshold.*

The factorization is objectwise. It preserves no quotient stream, no host real equality, no selected limit, no countable-choice coordinate, no metric-completeness row, no aggregate formalstatus *row, no aggregate bridge row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, assume $\mathcal{R}$ satisfies the current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. Apply Definition 38.471. The audit boundary is the product of the four local face records, with each closurestatus, formalstatus, leantarget, and bridgestatus row read from its own face chapter.

Next apply Definition 301.218. Since the source is the displayed finite $\mathsf{RegSeqRat}^{\uparrow}$ terminal record, the accepted exit-threshold coordinate is equivalent to the RegSeqRat four-face exactness criterion on that same record.

It remains to read the roadmap threshold objectwise. Use Definition 126.509 for the StreamName-side threshold readback, and use Definition 291.107 to identify the shared $\mathsf{NameCert}$ surface consumed by the dyadic ledger, StreamName finite window, RegSeqRat readback, and local Real seal. Thus the audit read factors through the four local closurestatus and formalstatus records rather than through a route-level credential.

For the reverse direction, assume the RegSeqRat four-face exit-threshold equivalence holds on $\mathcal{R}$ and the four face chapters supply their own local closurestatus and formalstatus records at the roadmap threshold. The equivalence gives the accepted Phase $\mathsf{Real}^{\uparrow}$ and Completion exit-threshold coordinate on $\mathcal{R}$.

Apply Definition 126.509 and Definition 38.471. Together they read that accepted threshold as the objectwise product of the four local audit-clean face records, with no cached aggregate status. The DyadicRatCore interface theorem Definition 291.107 supplies the same displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support surface, so the product is exactly the current audit-threshold read on $\mathcal{R}$.

The excluded boundary is preserved in both directions. The cited exit equivalence, StreamName readback, Real audit pullback, and DyadicRatCore interface mention only the four local faces and the displayed finite BEDC support rows. None introduces quotient streams, host real equality, selected limits, countable choice, metric-completeness data, aggregate formalstatus, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 301.220 (RegSeqRat audit-threshold missing-face obstruction)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record accepted by the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read. Then the accepted read requires the four local face records*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

on that same finite source. If any one of these four local face records is absent, the read is inadmissible at the RegSeqRat audit threshold.

No aggregate $\mathsf{formalstatus}$ *row, aggregate bridge row, selected limit, quotient stream, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row can compensate for the absent face. In particular, the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows certify the displayed local interface only after all four faces are present.*

Proof. Start with the audit-threshold factorization of Definition 301.219. It identifies any accepted current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read on $\mathcal{R}$ with the RegSeqRat four-face exit-threshold equivalence together with local closurestatus and formalstatus records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now apply Definition 301.218. The accepted threshold coordinate is equivalent to the exact four-face terminal surface on the same finite $\mathsf{RegSeqRat}^{\uparrow}$ record, so removing one displayed face destroys the surface on which the equivalence is stated. The read therefore has no RegSeqRat-local source after any face is absent.

It remains to exclude replacements. The DyadicRatCore interface theorem Definition 291.107 supplies only the local NameCert interface over the displayed BHist, hsame, Cont, Pkg, and NameCert support rows, and Definition 126.509 reads the audit-clean threshold through the four local faces objectwise. Finally, Definition 38.471 reads the current Phase Real and Completion audit from those local records rather than from an aggregate row. Hence selected limits, quotient streams, countable choice, metric-completeness data, aggregate verification or bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot supply the missing local face. □

Theorem 301.221 (RegSeqRat audit-threshold objectwise face necessity)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record. The record $\mathcal{R}$ satisfies the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read if and only if $\mathcal{R}$ carries the four objectwise local records*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

on the same terminal source, with support only from

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The four local records are exactly the StreamName finite-window rows, the DyadicRatCore endpoint, radius, distance, and regularity rows, the RegSeqRat classifier and common-tail rows, and the local Real seal rows.

The equivalence excludes quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate formalstatus *rows, aggregate bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. None of these excluded coordinates is part of the objectwise source required by the audit threshold.*

Proof. For the forward direction, assume $\mathcal{R}$ satisfies the current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. Apply Definition 301.219. The audit read factors through the RegSeqRat four-face exit-threshold equivalence and through the four local face records for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 301.218. The accepted exit-threshold coordinate is placed on the same finite RegSeqRat terminal source record, so the four local records are not external witnesses. They are the objectwise faces of $\mathcal{R}$.

It remains to identify the local records read by the factorization. Use Definition 126.509 to read the StreamName finite-window threshold objectwise. Use Definition 291.107 to read the DyadicRatCore endpoint, radius, distance, regularity, and shared NameCert interface over the displayed BHist, hsame, Cont, Pkg, and NameCert support. Use Definition 38.471 to read the Phase Real and Completion audit as the product of the four local face records rather than as an aggregate route row.

For the reverse direction, assume $\mathcal{R}$ carries the four displayed local records on the same terminal source and only the displayed support rows. Apply Definition 301.219 with those four records supplied locally. The factorization reduces the audit-threshold read to the RegSeqRat four-face exit-threshold equivalence on $\mathcal{R}$.

Finally apply Definition 301.218. Since the StreamName, DyadicRatCore, RegSeqRat, and local Real rows are already present on $\mathcal{R}$, the equivalence supplies the accepted Phase $\mathsf{Real}^{\uparrow}$ and Completion exit-threshold coordinate, and the factorization reads it as the accepted audit-threshold coordinate.

The boundary exclusions follow from the same objectwise path. The factorization theorem, the exit-threshold equivalence, the StreamName readback, the DyadicRatCore interface, and the

Real audit pullback mention only the four local records and the displayed finite BEDC support rows. Therefore quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not available to witness either direction. □

Theorem 301.222 (RegSeqRat audit-threshold face-status minimality)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record satisfying the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read. The BEDC-visible status interface of this read is exactly the four local records*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

where each record contributes only its own closurestatus*,* formalstatus*,* leantarget*, and* bridgestatus *rows.*

No proper subfamily of these four local status records witnesses the audit threshold on $\mathcal{R}$. No aggregate formalstatus *row, aggregate bridge row, selected limit, quotient stream, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row witnesses it either.*

Conversely, if the same finite source $\mathcal{R}$ carries the four displayed local status records and the RegSeqRat four-face exit-threshold equivalence of Definition 301.218, then $\mathcal{R}$ satisfies the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read.*

Proof. For necessity, apply Definition 301.219 to the accepted audit-threshold read on $\mathcal{R}$. The read factors through the RegSeqRat four-face exit-threshold equivalence together with the local status records of $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The records are read objectwise. By Definition 301.221, the four faces are the StreamName finite-window rows, the DyadicRatCore endpoint, radius, distance, and regularity rows, the RegSeqRat classifier and common-tail rows, and the local Real seal rows on the same terminal source. The facewise status pullback Definition 301.119 and the common-tail status pullback Definition 301.194 then keep each closurestatus, formalstatus, leantarget, and bridgestatus row attached to its own face chapter.

It remains to prove minimality. Apply Definition 301.220. If any one of the four local face records is absent, the audit-threshold read has no RegSeqRat-local source. Since every proper subfamily omits at least one displayed face, no proper subfamily can witness the read.

The same obstruction excludes replacements. The factorization theorem has no coordinate for an aggregate verification row or aggregate bridge row, and the objectwise face pullbacks expose no selected limit, quotient stream, countable-choice row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness row. Those data therefore cannot replace a missing local status record.

For sufficiency, assume the four local status records and the RegSeqRat four-face exit-threshold equivalence on the same finite source $\mathcal{R}$. Apply Definition 301.219 in the reverse direction. The supplied local status rows give exactly the status side of the factorization, and the supplied exit-threshold equivalence gives the threshold side. Hence $\mathcal{R}$ satisfies the current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. □

Theorem 301.223 (RegSeqRat audit-threshold status pullback square)**.** *Let $\mathcal{R}$ be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source record whose visible faces are*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and whose support rows are only

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Read from the four faces the local closurestatus*,* formalstatus*,* leantarget*, and* bridgestatus *rows. Then the square*

$$\begin{array}{ccc} \mathsf{FaceStatus}(\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}) & \longrightarrow & \mathsf{RealL10TerminalPullback} \\ \downarrow & & \downarrow \\ \mathsf{RegSeqRatFourFaceExitThreshold} & \longrightarrow & \mathsf{BEDCVisibleStatusInterface}(\mathcal{R}) \end{array}$$

commutes: reading the four local status records and then applying the $\mathsf{Real}^{\uparrow}$ *L10 terminal pullback gives the same BEDC-visible status interface as first applying the RegSeqRat four-face exit-threshold equivalence and then reading the face records objectwise.*

The commutative square is exactly objectwise over the four displayed faces. It excludes quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate formalstatus *rows, aggregate bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 301.219. The audit read on $\mathcal{R}$ is identified with the RegSeqRat four-face exit-threshold equivalence together with the local status records on $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next use Definition 301.222. The only status records available to the audit threshold are the four local closurestatus, formalstatus, leantarget, and bridgestatus rows, so the upper-left corner of the displayed square has no aggregate status or bridge coordinate.

The vertical attachment is objectwise. By Definition 301.194, each RegSeqRat common-tail status row remains attached to its own face chapter. By Definition 126.509, the StreamName audit-clean threshold is read back from the same four visible faces rather than from a route-level row.

Finally apply Definition 38.471 and Definition 38.458. Both paths in the square pass through the same four-face terminal packet, so the resulting BEDC-visible status interface on $\mathcal{R}$ is unique. Since none of the cited pullbacks supplies quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those coordinates are outside the square. □

Theorem 301.224 (RegSeqRat audit-threshold determinacy)**.** *Let* $\mathcal{R}_0$ *and* $\mathcal{R}_1$ *be finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source records. Suppose both records expose the same four visible face rows*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the same support rows

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

If $\mathcal{R}_0$ *and* $\mathcal{R}_1$ *both satisfy the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read, then the two accepted audit reads determine the same BEDC-visible terminal source record.*

The determined source is only the displayed four-face RegSeqRat terminal record. The determinacy claim excludes quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus *rows, aggregate bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 301.219 separately to the accepted audit-threshold reads on $\mathcal{R}_0$ and $\mathcal{R}_1$. Each accepted read factors through the RegSeqRat four-face exit-threshold equivalence and through the local closurestatus and formalstatus records of $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 301.218 to both factorizations. Since the two records expose the same StreamName finite-window rows, DyadicRatCore endpoint, radius, distance, and regularity rows, RegSeqRat classifier and common-tail rows, local Real rows, and displayed support, the exit-threshold coordinate in each read is placed on the same finite four-face source surface.

It remains to rule out a hidden route-level distinction between the two accepted reads. The StreamName threshold readback Definition 126.509 reads the audit-clean threshold from the visible four faces, Definition 38.471 pulls the current Phase Real and Completion audit back to the local face records, and Definition 291.107 identifies the shared $\mathsf{NameCert}$ interface over the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Thus every coordinate used by either accepted audit read is already fixed by the common four visible faces and common support rows. None of the cited components supplies quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate verification status, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Those absent coordinates cannot distinguish $\mathcal{R}_0$ from $\mathcal{R}_1$, so the two reads determine the same BEDC-visible terminal source record. □

Theorem 301.225 (RegSeqRat audit-threshold finite-refinement invariance)**.** *Let $\{\mathcal{R}_i\}_{i\in I}$ be a finite family of $\mathsf{RegSeqRat}^{\uparrow}$ terminal source records whose displayed faces are*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Suppose a finite cover refinement replaces this family by $\{\mathcal{R}'_j\}_{j\in J}$ and preserves, on every refined member, the four displayed face rows together with the shared

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

rows. If the original family carries the current Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate objectwise, then the refined family carries the same coordinate objectwise. Conversely, any refined family with these preserved rows determines the same coordinate on the original finite family.

The invariant coordinate is only the four-face audit-threshold read of the displayed RegSeqRat terminal source. The comparison excludes quotient streams, selected limits, countable choice, metric-completeness rows, aggregate $\mathsf{formalstatus}$ *rows, aggregate bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness.*

Proof. First compare the common-tail part of the two presentations. Apply Definition 301.211. It identifies the RegSeqRat classifier, common-tail atlas, and overlap rows read from the original finite family with the rows read from the refined family, because the refinement preserves the displayed $\mathsf{RegSeqRat}^{\uparrow}$ face and the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next compare the dyadic and Real exit data. The DyadicRatCore finite-cover theorem, Definition 291.120, keeps the endpoint, radius, distance, regularity, and ledger rows fixed under the same finite cover comparison. The Real functoriality theorem, Definition 38.805, transports the local $\mathsf{Real}^{\uparrow}$ exit certificate across the support-row-preserving finite refinement.

Now apply the status pullback square Definition 301.223. Since the StreamName, DyadicRatCore, RegSeqRat, and local Real rows agree after the preceding comparisons, both routes through the square read the same BEDC-visible status interface and the same Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate.

Finally use Definition 301.224. Any two objectwise audit reads obtained from the original family and from the refined family expose the same four face rows and the same displayed support

rows, so they determine one BEDC-visible terminal source record. The reverse direction is the same argument applied to the common finite refinement that maps back to the original finite cover.

Every cited comparison is facewise and support-row preserving. None supplies quotient streams, selected limits, countable choice, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no excluded coordinate can alter the invariant audit-threshold read. □

Theorem 301.226 (RegSeqRat four-face audit source uniqueness)**.** *Let $\mathcal{R}_0$ and $\mathcal{R}_1$ be finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal source records equipped with accepted current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold reads. Suppose the two reads expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, distance, and regularity rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the same displayed support*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then the two accepted reads determine the same RegSeqRat four-face source and the same accepted audit-threshold coordinate.

The uniqueness is local to the displayed four faces. It excludes quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate formalstatus *rows, aggregate bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 301.219 to the accepted audit-threshold reads on $\mathcal{R}_0$ and $\mathcal{R}_1$. Each read factors through the RegSeqRat four-face exit-threshold equivalence and through the local closurestatus and formalstatus records of $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply Definition 301.218 to the two factorizations. Because the StreamName finite window, DyadicRatCore endpoint, radius, distance, and regularity rows, RegSeqRat classifier and common-tail rows, local Real seal rows, and displayed BHist, hsame, Cont, Pkg, and NameCert support are the same, the accepted exit-threshold coordinate is read on the same four-face source surface.

Now use Definition 301.224. The theorem identifies the BEDC-visible terminal source determined by the two accepted audit reads, so the source left by the two factorizations is a single RegSeqRat four-face source.

Finally read the objectwise face records. The $\mathsf{StreamName}^{\uparrow}$ finite-window rows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, distance, and regularity rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier and common-tail rows, and local $\mathsf{Real}^{\uparrow}$ seal rows are exactly the four records used by the factorization and exit-threshold equivalence. Since the displayed BHist, hsame, Cont, Pkg, and NameCert support is also shared, no second accepted audit-threshold coordinate remains.

The cited records are objectwise local. They do not provide quotient streams, selected limits, host real equality, countable choice, metric-completeness data, aggregate verification status, aggregate bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of those excluded coordinates can distinguish a second completion-facing source or a route-level status credential. □

Theorem 301.227 (RegSeqRat audit-threshold finite-cover gluing)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections for a Phase* $\mathsf{Real}^{\uparrow}$ *and Completion consumer. Assume that every nonempty overlap agrees on the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared support*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Assume also that the overlap status-pullback squares of Definition 301.223 agree under these displayed identifications.

Then the glued terminal section satisfies the Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read if and only if every local section satisfies the objectwise audit-threshold read and the overlap status-pullback squares agree. The glued read is objectwise over the finite four-face packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with only the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support.*

The gluing theorem exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatAuditThresholdFiniteCoverGluing`

Proof. For the forward direction, restrict the glued Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read to a local section S_i. The restriction preserves the displayed StreamName finite window, DyadicRatCore endpoint, radius, and distance ledger, RegSeqRat classifier and common-tail rows, local Real seal rows, and the shared BHist, hsame, Cont, Pkg, and NameCert support.

Apply Definition 301.218 on each restricted terminal packet. The local threshold coordinate is therefore the same objectwise four-face read as the RegSeqRat terminal exactness certificate. Then apply Definition 301.223 to read the local closurestatus, formalstatus, leantarget, and bridgestatus rows through the same four faces.

The overlap agreement is inherited from the glued packet. By Definition 301.224 and Definition 301.226, two local restrictions with the same displayed overlap rows determine one BEDC-visible terminal source and one accepted audit-threshold coordinate. Hence every local section satisfies the objectwise audit-threshold read, and the status-pullback squares agree on overlaps.

For the reverse direction, assume each local section satisfies the objectwise audit-threshold read and the overlap status-pullback squares agree. Apply Definition 301.240 to the underlying four-face local packets. The StreamName windows, DyadicRatCore ledgers, RegSeqRat classifier and common-tail rows, local Real seal rows, and support rows glue to one accepted $\mathsf{Real}^{\uparrow}$ terminal seal over the finite cover.

Next apply Definition 38.796. It reads the glued terminal seal as the objectwise Phase $\mathsf{Real}^{\uparrow}$ and Completion threshold certificate over the four local records. The overlap status-pullback hypothesis supplies the facewise status interface required by Definition 301.223, so the threshold certificate is read as the Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read on the glued terminal section.

It remains to identify the glued audit source. Use Definition 301.224 on any two local presentations carried into the glued packet, and then use Definition 301.226 to remove any second audit-threshold coordinate over the same displayed faces. The resulting source is the single finite four-face $\mathsf{RegSeqRat}^{\uparrow}$ terminal section obtained by gluing the cover.

All cited constructions use only the finite rows named in the statement. None introduces quotient streams, selected limits, countable choice, metric-completeness data, aggregate verification status, aggregate bridge data, standard bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the glued audit-threshold read contains none of those excluded coordinates. □

Theorem 301.228 (RegSeqRat four-face status cofinal descent)**.** *Let* $\{S_i\}_{i\in I}$ *be a compatible finite cover satisfying the hypotheses of Definition 301.227, and let* $\{T_j\}_{j\in J}$ *be a cofinal finite cover refining it. Assume each refinement map sends every* T_j *into some* S_i *and preserves the*

$\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, the shared support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

and the overlap status-pullback squares of Definition 301.223. Assume moreover that the refinement is cofinal on displayed observations: every row of the original four-face packet is the pullback of a finite family of rows on the refined cover, and compatible refined overlap rows descend to the original overlap row.

Then the refined cover glues to the same Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read if and only if the original cover glues to that read. The common glued read is determined by the same finite four-face packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with exactly the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support.*

The descent exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate $\mathsf{formalstatus}$ *row, aggregate bridge row, standard bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Suppose first that the original cover glues to the displayed Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. Pull this glued packet along each refinement map. The preservation hypotheses keep the StreamName windows, DyadicRatCore ledgers, RegSeqRat classifier and common-tail rows, Real seal rows, and support rows unchanged on every refined member.

The overlap status-pullback square is stable under these pullbacks by hypothesis. Apply Definition 301.227 to the refined cover. The refined local reads therefore glue to an audit-threshold section over the same four faces and support.

It remains to identify this refined glued section with the original one. Use Definition 301.224 on the two glued presentations over the shared finite rows, and then apply Definition 301.226 to rule out a second audit source over the same StreamName, DyadicRatCore, RegSeqRat, and Real faces. Thus the refined cover glues to the same read.

Conversely assume that the refined cover glues to that read. Cofinality on displayed observations descends the refined StreamName windows, DyadicRatCore ledgers, RegSeqRat classifier and common-tail rows, local Real seals, support rows, and compatible overlap status-pullback squares to the original finite cover.

Apply Definition 301.227 to the descended original cover. Its glued read has the same four-face data as the refined glued read. The local status-pullback square transports the closurestatus, formalstatus, leantarget, and bridgestatus rows through the same finite faces, so determinacy again identifies the two presentations.

All objects used in both directions are rows of the displayed finite covers or their status-pullback squares. The argument never selects a limit, forms a quotient stream, invokes countable choice, uses metric completeness, aggregates status or bridge rows, adds a standard bridge row, or appeals to ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 301.229 (RegSeqRat audit-threshold refinement composition)**.** *Let* $\{S_i\}_{i\in I}$, $\{T_j\}_{j\in J}$, *and* $\{U_k\}_{k\in K}$ *be finite covers of compatible local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections. Suppose* $\{T_j\}_{j\in J}$ *cofinally refines* $\{S_i\}_{i\in I}$ *and* $\{U_k\}_{k\in K}$ *cofinally refines* $\{T_j\}_{j\in J}$. *Assume all refinement maps preserve the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and*

distance ledgers, the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, the support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

and the status-pullback squares of Definition 301.223. Assume also that the composite refinement maps from $\{U_k\}_{k\in K}$ *to* $\{S_i\}_{i\in I}$ *are cofinal on the same displayed observations.*

Then the Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read obtained by descending along*

$$\{U_k\}_{k\in K} \longrightarrow \{T_j\}_{j\in J} \longrightarrow \{S_i\}_{i\in I}$$

is identical to the read obtained by the direct cofinal refinement

$$\{U_k\}_{k\in K} \longrightarrow \{S_i\}_{i\in I}.$$

Both reads are the finite four-face packet

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

The composition exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 301.228 to the refinement from $\{T_j\}_{j\in J}$ to $\{S_i\}_{i\in I}$. The hypotheses identify the audit-threshold read on the T cover with the read on the S cover through the same StreamName, DyadicRatCore, RegSeqRat, Real, and support rows.

Apply Definition 301.228 again to the refinement from $\{U_k\}_{k\in K}$ to $\{T_j\}_{j\in J}$. The two-step descent therefore carries the U cover to the same T read and then to the same S read, without changing any displayed status-pullback square.

For the direct route, use the cofinality of the composite maps from $\{U_k\}_{k\in K}$ to $\{S_i\}_{i\in I}$. A third application of Definition 301.228 identifies the directly descended U read with the S read over the same four visible faces.

It remains to compare the witnesses supplied by the two routes. Apply Definition 301.227 on each of the three covers to identify the glued read determined by its compatible local packets. Then use Definition 301.224 and Definition 301.226 to rule out a second audit source over the same StreamName, DyadicRatCore, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

All data used by the comparison are rows already preserved by the refinement maps or status-pullback squares already present in the covers. No quotient stream, selected limit, countable choice, metric-completeness datum, aggregate status row, aggregate bridge row, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness is introduced. □

Theorem 301.230 (RegSeqRat finite-cover overlap no-confusion)**.** *Let* $\{S_i\}_{i\in I}$ *and* $\{T_j\}_{j\in J}$ *be compatible finite covers of local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections for the same Phase* $\mathsf{Real}^{\uparrow}$ *and Completion consumer. Suppose both covers satisfy the hypotheses of Definition 301.227. Assume that, after passing to the common finite overlap refinement, the two presentations expose the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and distance ledger rows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and common-tail rows, the same local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the same support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the glued Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold reads supplied by the two covers are identical.*

Conversely, if the glued Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold reads of the two covers are identical, the common read recovers exactly the displayed overlap rows: the four L10 faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. It recovers no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate* formalstatus *row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 301.227 to $\{S_i\}_{i\in I}$ and to $\{T_j\}_{j\in J}$. Each cover produces a glued audit-threshold read whose source data are precisely the displayed StreamName, DyadicRatCore, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

The common overlap refinement identifies these displayed rows on the two covers. Use Definition 301.224 to compare the two glued presentations over that shared finite packet, and use Definition 301.226 to rule out a second BEDC-visible audit source over the same four faces and support. This proves equality of the glued Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold reads.

For the converse, start from equality of the glued reads and restrict the common read back along the finite covers. The finite-cover gluing theorem says that every local and overlap row used by either glued read is one of the displayed four faces or one of the displayed support rows. Thus equality of the read identifies exactly those rows on the common overlap surface.

Cofinal changes of the finite cover do not add a new recovery coordinate. Apply Definition 301.228 to compare either cover with any common finite overlap refinement, and apply Definition 301.229 to identify the direct and two-step descent routes. The recovered rows are therefore independent of the chosen overlap refinement.

It remains to exclude hidden substitutes. Invoke Definition 301.231: deleting any displayed face row or support row prevents presentation of the same glued read, while quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not replacement coordinates for the deleted row. Hence equality of the glued audit-threshold read recovers exactly the four L10 faces and BEDC support rows, and nothing outside that surface. □

Theorem 301.231 (RegSeqRat audit-threshold support-removal descent)**.** *Let* $\{S_i\}_{i\in I}$ *be a finite cover satisfying the hypotheses of Definition 301.227, and let* G *be the glued terminal section supplied by that theorem. Delete from one displayed local section or overlap one row among the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, or distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier or common-tail row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, or one of the shared support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

Then the resulting cover cannot present the same glued Phase $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read on* G*.*

The obstruction is support-local and descends to the glued threshold. Quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus *rows, aggregate bridge rows, standard bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are not replacement coordinates for the deleted row.*

Proof. First read the original cover through Definition 301.227. The glued audit-threshold section is determined by the finite four-face packet

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and by the displayed BHist, hsame, Cont, Pkg, and NameCert support. Thus every row named in the statement is part of the terminal presentation used by the glued read.

If the deleted row is one of the four visible local faces, apply Definition 301.220 to the affected local terminal packet. The RegSeqRat audit-threshold read has no admissible local source after the face is absent, so the finite cover no longer satisfies the local hypotheses required by the gluing theorem.

If the deleted row is one of the shared support rows, apply Definition 126.665 on the StreamName–DyadicRatCore–RegSeqRat–Real route carried by the same finite cover. The cover-descent terminal support packet then cannot present the same $\mathsf{Real}^{\uparrow}$ seal-visible source. The status-pullback square of Definition 301.223 transports this defect to the RegSeqRat audit-threshold read, because the status rows are read through the same displayed faces and support.

It remains to rule out replacement data. The finite-cover gluing theorem already exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. The missing-face obstruction and the StreamName support-removal obstruction exclude the same objects as substitutes for a deleted local row. Therefore the altered cover cannot present the same glued Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. □

Theorem 301.232 (RegSeqRat four-face support induction)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections satisfying the hypotheses of Definition 301.227. Then the glued Phase* $\mathsf{Real}^{\uparrow}$ *and Completion audit-threshold read is generated by finite induction over the cover from exactly the four visible L10 faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed support

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The base packet is a singleton local terminal section. The induction step glues one compatible local terminal packet to an already glued finite subcover by the overlap status-pullback square, and the resulting audit-threshold read is independent of the order of adjoining local packets and of cofinal finite refinements.

The induction exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim. Removing any displayed face row or support row destroys the possibility of presenting the same glued read.*

Proof. For the base case, take a singleton local terminal packet S_i. The hypotheses of Definition 301.227 reduce to the local objectwise audit-threshold read and the displayed support rows, so the generated packet is precisely the four-face $\mathsf{RegSeqRat}^{\uparrow}$ terminal section with BHist, hsame, Cont, Pkg, and NameCert support.

For the induction step, assume a finite subcover has already been glued from the same displayed faces and support. Adjoin one further compatible local section. The overlap hypotheses identify the StreamName windows, DyadicRatCore ledgers, RegSeqRat classifier and common-tail rows,

Real seal rows, and support rows on the common overlap. Apply Definition 301.227 to glue the enlarged finite cover.

The induction step has no hidden branch. Use Definition 301.230 on the overlap between the previously glued subcover and the adjoined local section. Equality of glued reads recovers exactly the four visible faces and support rows, so no second audit source or extra coordinate can enter at the adjoining boundary.

Cofinal presentations of the same finite induction are identified by Definition 301.228. When two adjoining schedules are compared through a common refinement, Definition 301.229 identifies the direct descent with the two-step descent. Hence the finite induction does not depend on the order of adjoining local packets or on the chosen finite refinement.

Finally apply Definition 301.231. If any StreamName, DyadicRatCore, RegSeqRat, Real, BHist, hsame, Cont, Pkg, or $\mathsf{NameCert}$ row is removed from a local member or overlap, the altered cover cannot present the same glued Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read. Since the gluing, descent, composition, no-confusion, and support-removal steps all exclude quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness, the finite induction derives the L10 audit-threshold read exactly from the stated four-face support surface. □

Theorem 301.233 (RegSeqRat finite-cover status-gluing completeness)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections satisfying the hypotheses of Definition 301.227. Suppose moreover that every local member and every overlap displays the four Phase* $\mathsf{Real}^{\uparrow}$ *and Completion faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

with support

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},$$

and that the overlap maps carry the corresponding status-pullback squares.

Then the Phase $\mathsf{Real}^{\uparrow}$ *and Completion status-gluing interface for this finite cover is complete exactly when the finite induction of Definition 301.232 supplies the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows together with the overlap status-pullback squares. Equivalently, no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate* $\mathsf{formalstatus}$ *row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can be part of the same status-gluing read.*

Proof. First apply Definition 301.227. The compatible local terminal sections glue to one Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold read over exactly the displayed four faces and support rows.

The resulting read is independent of presentation. Cofinal descent in Definition 301.228 identifies cofinal finite covers of the same local data, and refinement composition in Definition 301.229 identifies a direct descent with any two-step descent through a common finite refinement.

Uniqueness of the status-gluing read is forced on overlaps. By Definition 301.230, two glued reads agreeing as terminal status objects agree row-by-row on the four visible faces, on BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, and on the overlap status-pullback squares. Thus no second source of status data can be hidden inside the glued object.

Necessity of every displayed row follows from Definition 301.231. Removing any named face row, support row, or overlap square prevents the altered cover from presenting the same glued Phase $\mathsf{Real}^{\uparrow}$ and Completion read.

Finally, Definition 301.232 says that the whole finite cover is generated by adjoining local terminal packets through those overlap squares. Combining the existence, refinement-independence,

no-confusion, support-removal, and finite-induction clauses proves the exact completeness assertion and excludes quotient streams, selected limits, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the same status-gluing interface. □

Theorem 301.234 (RegSeqRat zero-source finite-cover gluing)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections satisfying the hypotheses of Definition 301.227. Suppose each local section also displays the same* $\mathsf{RegularCauchyZero}^{\uparrow}$ *source boundary through the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero endpoint and radius ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared support*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then the glued terminal section carries the same $\mathsf{RegularCauchyZero}^{\uparrow}$ *zero-source boundary. Its terminal read is the finite four-face packet*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

specialized to the constant zero source, and it exports no quotient stream, host real equality, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus *row, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 301.227 to the underlying compatible terminal sections. The result is one glued audit-threshold read over the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with only the listed BHist, hsame, Cont, Pkg, and NameCert support.

Now identify the zero source on the glued packet. The local zero-source hypotheses say that every finite-window face is the constant $\mathsf{StreamName}^{\uparrow}$ zero row, every dyadic face is the same zero endpoint and radius ledger, every regular face is the same $\mathsf{RegSeqRat}^{\uparrow}$ zero readback row, and every seal face is the local $\mathsf{Real}^{\uparrow}$ seal row of the $\mathsf{RegularCauchyZero}^{\uparrow}$ packet. Apply Definition 126.551 to paste the StreamName and RegSeqRat source rows, and apply Definition 126.552 to paste the local seal rows over the same finite cover.

The pasted four-face packet is still the zero packet. Use Definition 624.3 to read the constant StreamName and dyadic ledger faces as the sibling route of the accepted $\mathsf{RegularCauchyZero}^{\uparrow}$ source, and use Definition 624.8 to identify the glued read with the same terminal zero-source boundary on every downstream consumer.

Finally apply Definition 624.6 and Definition 624.7. Their boundary excludes quotient streams, host real equality, selected limits, nonzero witnesses, and ambient completeness from the zero packet. The additional exclusions for countable choice, metric-completeness data, aggregate formalstatus rows, and bridge rows follow from Definition 301.177 and the audit-threshold gluing theorem, because those objects are not among the displayed four faces or support rows. □

Theorem 301.235 (RegSeqRat zero-source deleted-row obstruction)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover satisfying the hypotheses of Definition 301.234, and let G be the glued terminal section carrying the resulting* $\mathsf{RegularCauchyZero}^{\uparrow}$ *boundary. Delete from one local section or overlap one row among the constant* $\mathsf{StreamName}^{\uparrow}$ *zero-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero endpoint or radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback row, the local* $\mathsf{Real}^{\uparrow}$ *seal row, or one of the shared support rows*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Then the altered finite cover cannot present the same glued $\mathsf{RegularCauchyZero}^{\uparrow}$ *zero-source boundary on G.*

The obstruction is terminal-consumer visible. Quotient streams, host real equality, selected limits, countable-choice rows, metric-completeness data, aggregate formalstatus *rows, bridge rows, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness do not replace the deleted zero-source row.*

Proof. First apply Definition 301.234 to the original cover. The glued read is the zero specialization of the four-face packet

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with only the displayed BHist, hsame, Cont, Pkg, and NameCert support. Hence every row named in the statement is part of the source boundary used by the terminal consumer.

If the deleted row is one of the displayed zero faces, apply Definition 301.220 to the affected local terminal packet. The missing constant StreamName window, zero dyadic ledger, RegSeqRat zero readback, or local Real seal prevents the local packet from supplying the same audit-threshold source. The zero-source gluing theorem therefore has no compatible local zero packet from which to recover the same boundary on G.

If the deleted row is one of the shared support rows, first apply Definition 301.231 to transport the support loss to the glued RegSeqRat audit-threshold read. Then apply Definition 126.665 on the StreamName–DyadicRatCore–RegSeqRat–Real route. The altered cover no longer carries the global support needed to present the same zero-source seal-visible boundary.

It remains to exclude external substitutes. Apply Definition 624.8 to identify any alleged presentation of the same glued boundary with the terminal zero-source consumer. Then use Definition 624.6 and Definition 624.7 to rule out quotient streams, host real equality, selected limits, nonzero witnesses, and ambient completeness as alternate witnesses.

The countable-choice, metric-completeness, aggregate status, and bridge rows are also outside the finite source packet. By Definition 301.177 and the finite-cover gluing theorem, none of them is among the four displayed zero faces or the shared BHist, hsame, Cont, Pkg, and NameCert support. Thus the altered finite cover cannot present the same glued $\mathsf{RegularCauchyZero}^{\uparrow}$ zero-source boundary. □

Theorem 301.236 (RegSeqRat stream-dyadic real section coherence)**.** *Let R be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let S be an accepted finite request section whose displayed rows include the* $\mathsf{StreamName}^{\uparrow}$ *pointwise window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and the* $\mathsf{Real}^{\uparrow}$ *finite-section readback. If those rows are consumed by a Phase Real and Completion reader, then the StreamName window, dyadic ledger, regular-rational readback, and Real finite-section readback factor through one common source section carried by the same finite* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

The coherent section is only the finite L10 route

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

It exports no quotient stream equality, selected limit, host real equality, ambient completeness row, metric-completion point, countable-choice selector, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Begin with the finite request source section of Definition 301.139. It places the accepted request on one displayed BHist packet with the StreamName schedule rows, DyadicRatCore endpoint, radius, and finite-distance rows, RegSeqRat classifier and regularity rows, componentwise hsame, displayed Cont routes, Pkg provenance, and local naming.

Next attach the dyadic and Real faces. The exact handoff of Definition 301.69 keeps the StreamName window, DyadicRatCore ledger, and RegSeqRat classifier on the same finite route,

while Definition 301.132 records the constant dyadic case as a special Real-facing handoff through the same four visible faces.

Now use the common-tail bridge. The classifier bridge of Definition 301.182 and the finite-request pullback of Definition 301.184 identify common-tail and finite-request presentations with the same StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Finally compare the sibling finite-section surfaces. The StreamName finite-section pullback and lock, Definition 126.586 and Definition 126.589, identify the pointwise window as a finite section over the same dyadic and RegSeqRat rows. The Real terminal finite-section pullback and four-face exactness, Definition 38.720 and Definition 38.457, read the terminal Real section from that same ordered packet.

Thus the StreamName pointwise window, DyadicRatCore ledger, RegSeqRat readback, and Real finite-section readback are four projections of one common finite source section. The sibling chapters chapter 126, chapter 291, and chapter 38 supply only those displayed finite rows, so the excluded quotient, limit, host-equality, completeness, choice, and second-seal coordinates have no row from which to be read. □

Theorem 301.237 (RegSeqRat terminal four-face source uniqueness)**.** *Let R_0 and R_1 be accepted Phase Real and Completion terminal consumers of finite* $\mathsf{RegSeqRat}^{\uparrow}$ *sections. Suppose that they display the same* $\mathsf{StreamName}^{\uparrow}$ *finite-section window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the same local* $\mathsf{Real}^{\uparrow}$ *finite-section readback, and the same* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows. Then R_0 and R_1 determine the same common finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source section.*

The unique source is the four-face packet over

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

There is no independent $\mathsf{RegSeqRat}^{\uparrow}$ *readback branch, second* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream, selected limit, countable-choice row, host equality, ambient completeness row, or aggregate bridge or status row available to distinguish the two terminal consumers.*

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatTerminalFourFaceSourceUniqueness`

Proof. Apply Definition 301.236 to both terminal consumers. Each read is thereby placed on the same four visible projections: the StreamName finite window, the DyadicRatCore ledger, the RegSeqRat classifier and regularity surface, and the local Real finite-section readback.

Next identify the StreamName and dyadic coordinates. The StreamName finite-section pullback Definition 126.586 and the real, RegSeqRat, and dyadic lock Definition 126.589 force the two window coordinates to be the same finite section. The dyadic source uniqueness theorem Definition 291.84 then identifies the endpoint, radius, and distance rows as one dyadic source over that shared window.

Now keep the Real readback terminal. By Definition 38.720, each accepted terminal consumer reads its local Real section from the displayed finite packet. The four-face exactness theorem Definition 38.457 keeps that packet on the ordered $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with only the listed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 38.458. A second RegSeqRat source over the same displayed support rows would give a second four-face terminal packet for the same Phase Real and Completion boundary. Terminal pullback uniqueness identifies those packets, so the common finite $\mathsf{RegSeqRat}^{\uparrow}$ source section is determined uniquely. Since the cited four-face packet has no coordinate for

quotient streams, selected limits, choice rows, host equality, ambient completeness, a second Real seal, or an aggregate bridge or status surface, none of those rows can separate the two consumers. □

Theorem 301.238 (RegSeqRat four-face refinement functoriality)**.** *Let R be an accepted Phase Real and Completion terminal consumer over a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *section, and let R' be an accepted cofinal refinement of R. Suppose the refinement preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-section window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, the local* $\mathsf{Real}^{\uparrow}$ *finite-section readback, and the same* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows. Then R and R' determine the same common finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source section.*

Equivalently, accepted cofinal refinement acts functorially on the terminal four-face packet

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

with identity action on the common finite source. It creates no quotient stream, selected limit, choice row, host equality, ambient completeness row, or second $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Apply Definition 301.236 to the terminal consumer R and to its accepted refinement R'. Both reads are therefore represented by four visible faces over the same displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ support rows.

The preserved StreamName window is fixed by Definition 126.586. The preserved dyadic endpoint, radius, and finite-distance ledger are fixed by Definition 291.84. The local Real terminal readback is fixed by Definition 38.458.

It remains only to compare the RegSeqRat source face. Since the refinement preserves the RegSeqRat classifier and regularity rows and the other three faces have already been identified, both terminal reads satisfy the hypotheses of Definition 301.237. That theorem identifies the common finite $\mathsf{RegSeqRat}^{\uparrow}$ source section for R and R'.

The same packet supplies no coordinate for quotient streams, selected limits, choice rows, host equality, ambient completeness, a second Real seal, or any independent aggregate status surface. Hence the refinement has no additional row along which it could move the accepted terminal consumer away from the common finite source. □

Theorem 301.239 (RegSeqRat real terminal four-face exactness)**.** *Let S be a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal section presented to the Phase Real and Completion exit. Then S is accepted by the* $\mathsf{Real}^{\uparrow}$ *terminal seal if and only if its finite ledger contains one compatible four-face packet consisting of the* $\mathsf{StreamName}^{\uparrow}$ *finite-request section, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *common scale-window witness, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity witness, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal row, all carried by the same* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

The exact packet is the concrete terminal surface

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

It asserts no quotient stream equality, no selected limit existence, and no completeness property of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegSeqRatUp.RegSeqRatRealTerminalFourFaceExactness`

Proof. For the forward direction, start with the accepted $\mathsf{Real}^{\uparrow}$ terminal seal. By Definition 301.236, the terminal read factors through a single finite source section whose visible coordinates are

the StreamName finite-request window, the DyadicRatCore endpoint, radius, and finite-distance ledger, the RegSeqRat classifier and regularity rows, and the Real finite-section readback.

The compatibility of those faces is already part of the same finite ledger: the StreamName rows are tied to the displayed dyadic rows by Definition 126.589, the dyadic common scale-window rows are locked by Definition 291.84, and the terminal Real readback is pulled from the displayed packet by Definition 38.720. Thus the accepted seal supplies exactly the four required finite faces over one BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

For the reverse direction, suppose that those four compatible faces are present in the finite ledger. The StreamName face gives the finite-request section, the DyadicRatCore face gives the common scale-window witness, the RegSeqRat face gives the regularity witness, and the Real face gives the terminal seal row. Applying Definition 301.236 packages these rows as one common terminal section.

It remains only to identify the terminal consumer. The source uniqueness result Definition 301.237 shows that any accepted Phase Real and Completion consumer with these four faces must come from this same finite $\mathsf{RegSeqRat}^{\uparrow}$ source. The Real terminal pullback uniqueness theorem Definition 38.458 then identifies the displayed Real seal row as the terminal acceptance surface for S.

Every step used only rows visible in the finite BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ ledger. There is no coordinate for quotient stream equality, no selector for an external limit, and no ambient completeness field for $\mathsf{Real}^{\uparrow}$, so none of those stronger claims is produced by the exactness criterion. □

Theorem 301.240 (RegSeqRat four-face local gluing covers the Real seal)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local $\mathsf{RegSeqRat}^{\uparrow}$ terminal sections for a Phase Real and Completion consumer. Assume that on every nonempty overlap the two local presentations agree on the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and finite-distance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback, the $\mathsf{Real}^{\uparrow}$ finite-section readback, and the shared BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ support rows. Then the local sections glue to one accepted $\mathsf{Real}^{\uparrow}$ terminal seal if and only if each local section satisfies the four-face exactness criterion of Definition 301.239.*

The glued seal is covered only by the finite terminal faces

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

It contains no quotient stream coordinate, selected limit, host equality, ambient completeness row, countable-choice selector, standard bridge row, or second terminal seal.

Proof. For the forward direction, suppose the finite cover glues to an accepted $\mathsf{Real}^{\uparrow}$ terminal seal. Restrict the glued terminal consumer to any local piece S_i. The restriction keeps exactly the same displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Apply Definition 301.239 to this local restriction. The accepted terminal seal therefore supplies the local StreamName finite-request section, DyadicRatCore common scale-window witness, RegSeqRat regularity witness, and Real terminal row as one compatible four-face packet. Since the argument is local in i, every member of the finite cover satisfies the exactness criterion.

For the reverse direction, assume each S_i satisfies the four-face criterion. Each local packet has a common source section by Definition 301.236. On overlaps, Definition 126.587 identifies the StreamName classifier read from the shared finite window, and Definition 301.237 identifies the RegSeqRat terminal source because all DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows agree.

Thus the overlap data satisfy the finite-section descent hypothesis. Apply Definition 126.594 to glue the finite StreamName windows, carrying along the already identified dyadic ledger, RegSeqRat readback, Real finite-section readback, and support rows. The result is a single compatible four-face terminal packet over the finite cover.

Finally apply Definition 301.239 to the glued packet. The four compatible faces give exactly one accepted $\mathsf{Real}^{\uparrow}$ terminal seal for the covered $\mathsf{RegSeqRat}^{\uparrow}$ terminal section. The construction used only finite overlap agreement and the displayed ledger rows, so it cannot produce quotient streams, selected limits, host equality, ambient completeness, countable choice, standard bridge rows, or an additional terminal seal. □

Theorem 301.241 (RegSeqRat four-face gluing uniqueness)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local $\mathsf{RegSeqRat}^{\uparrow}$ terminal sections, and let T and T' be accepted $\mathsf{Real}^{\uparrow}$ terminal seals obtained from this same cover. Assume that both gluing presentations use the same $\mathsf{StreamName}^{\uparrow}$ finite windows, the same $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and finite-distance ledgers, the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier and readback rows, the same $\mathsf{Real}^{\uparrow}$ finite-section readbacks, and the same* BHist*,* hsame*,* Cont*,* Pkg*, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ support rows on every cover member and every nonempty overlap. Then T and T' are the same BEDC-visible four-face terminal packet.*

The comparison is internal to the four displayed faces

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

It introduces no quotient stream, selected limit, host equality, ambient completeness row, countable-choice selector, standard bridge row, or second terminal seal.

Proof. Apply Definition 301.240 separately to the presentations of T and T'. For each cover member S_i, both seals therefore restrict to local packets satisfying the four-face exactness criterion of Definition 301.239 with the same StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

On each local member and each nonempty overlap, use Definition 301.237. Since the two presentations have identical finite windows, endpoint, radius, distance, classifier, readback, and support ledgers, the local $\mathsf{RegSeqRat}^{\uparrow}$ terminal source sections coincide as BEDC-visible records.

The overlap ledger is now the same for the two presentations. Apply Definition 126.594 to the common StreamName finite windows. The glued StreamName section obtained from T and the glued StreamName section obtained from T' are the same finite descent object, carrying the already identified dyadic, RegSeqRat, RealUp, and support rows.

It remains to compare the terminal Real row. The common local four-face packet is read by Definition 301.239, and terminal pullback uniqueness Definition 38.458 identifies the accepted $\mathsf{Real}^{\uparrow}$ terminal row over that packet. Hence T and T' present the same BEDC-visible four-face terminal packet.

All identifiers used in the comparison come from the finite cover and its overlap ledger. There is no field in the argument for a quotient stream, selected limit, host equality, ambient completeness principle, countable choice, standard bridge row, or independent second terminal seal, so none is introduced by the uniqueness comparison. □

Theorem 301.242 (RegSeqRat four-face finite-section restriction functoriality)**.** *Let $\{S_i\}_{i\in I}$ be a finite cover of compatible local $\mathsf{RegSeqRat}^{\uparrow}$ terminal sections whose accepted $\mathsf{Real}^{\uparrow}$ terminal seal is presented by the four faces*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Restrict the cover to a finite subcover, or replace the local sections by a finite family obtained through support-row-preserving $\mathsf{StreamName}^{\uparrow}$ *reindex routes. Assume that the reindex routes preserve the finite request windows, tail-window rows, dyadic endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the* $\mathsf{Real}^{\uparrow}$ *finite-section readback, and the shared* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

Then the terminal packet over the restricted or reindexed family is exactly the restriction of the original four-face boundary. Conversely, any four-face terminal packet over such a restricted family is read from precisely the restricted local sections and the corresponding support rows.

This object-level comparison is confined to the finite terminal faces displayed above. It does not identify quotient streams, select limits, compare host reals, assume countable choice, add ambient or metric completeness, aggregate formal-status rows, or introduce standard bridge rows.

Proof. First restrict the accepted terminal seal to the chosen finite subcover or to the support-row-preserving reindexed family. Apply Definition 301.240 to the original cover. Each retained or reindexed local section is therefore covered by a four-face local packet satisfying the exactness criterion, with only the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows present.

For the $\mathsf{StreamName}^{\uparrow}$ face, use Definition 126.648 to transport the terminal four-face boundary along the reindex route. Use Definition 126.649 to identify the tail-window terminal rows after restriction. Thus the retained StreamName request windows and terminal rows are exactly the old ones read on the smaller finite support.

For the dyadic and real faces, apply Definition 291.84 to the endpoint, radius, and finite-distance ledger, and apply Definition 38.458 to the terminal $\mathsf{Real}^{\uparrow}$ seal over the same four-face packet. These two uniqueness results leave no freedom to create a different dyadic source or a different terminal Real row after finite restriction.

It remains to compare the whole packet. The restricted presentation and the packet read directly from the retained local sections have identical StreamName, DyadicRatCore, RegSeqRat, RealUp, and support rows on every retained member and overlap. Apply Definition 301.241. The two presentations are the same BEDC-visible four-face terminal packet.

Conversely, suppose a four-face terminal packet is given over the restricted or support-row-preserving reindexed family. Local acceptance follows from Definition 301.240, and the same StreamName, dyadic, and Real uniqueness references identify each face with the corresponding restricted row of the original cover. Hence the packet is read from exactly the retained local sections and support rows.

Every step only restricts finite indices, transports finite request windows, or invokes terminal uniqueness over the displayed four faces. The argument has no constructor or field for quotient stream equality, selected limits, host real equality, countable choice, ambient or metric completeness, aggregate formal-status data, or standard bridge rows, so none is introduced. □

Theorem 301.243 (RegSeqRat finite-request section cocycle Real seal)**.** *Let* S_0, S_1, *and* S_2 *be three compatible finite-request terminal sections over the same accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Suppose every pairwise overlap agrees on the* $\mathsf{StreamName}^{\uparrow}$ *finite request window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

Then the three pairwise Real-seal-visible packets satisfy the finite section cocycle condition: their restrictions to each triple overlap are the same terminal four-face packet. Consequently the three sections determine one BEDC-visible terminal packet over

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The conclusion is only about the finite request section and its displayed terminal Real seal. It does not introduce quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate status rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First place each pairwise overlap on the local Real-seal surface. Apply Definition 301.240 to the three pairwise families. Each overlap is therefore represented by the same four visible faces and by the listed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next identify the common finite request spine. The pairwise hypotheses give the same $\mathsf{StreamName}^{\uparrow}$ finite windows on all double overlaps, so Definition 126.594 glues the three windows to one finite descent object. The dyadic endpoint, radius, and finite-distance ledger are then forced by Definition 291.84; no second dyadic source can sit over the same finite request data.

Now compare the Real-seal-visible rows on a triple overlap. By Definition 301.242, restricting a pairwise packet to the triple overlap reads exactly the restricted StreamName, DyadicRatCore, RegSeqRat, RealUp, and support rows. Those rows are common by hypothesis and by the preceding descent step, so Definition 38.458 identifies the three restricted Real terminal rows as one terminal pullback.

The cocycle condition follows from this triple-overlap identification. To obtain the single terminal packet, apply Definition 301.241 to the three pairwise Real-seal-visible presentations. Since their restrictions agree on every overlap, the glued four-face packet is independent of the chosen pairwise route and is the unique BEDC-visible terminal packet determined by S_0, S_1, and S_2.

All data used in the construction are finite request windows, dyadic ledger rows, classifier and readback rows, the local Real seal, and the named support coordinates. The cited gluing, restriction, and terminal-uniqueness theorems have no coordinate for quotient streams, selected limits, host real equality, countable choice, metric-completeness rows, aggregate status rows, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the cocycle packet does not claim any of them. □

Theorem 301.244 (RegSeqRat finite-request section refinement composition)**.** *Let S_0, S_1, and S_2 be accepted finite-request terminal sections over the four-face route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Suppose $\rho_{01} : S_0 \to S_1$ and $\rho_{12} : S_1 \to S_2$ are compatible certified refinements preserving the finite $\mathsf{StreamName}^{\uparrow}$ *request windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-scale endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows. Suppose also that the direct certificate $\rho_{02} : S_0 \to S_2$ is accepted on the same visible support and has the same facewise restriction as the composite $\rho_{12} \circ \rho_{01}$.*

Then refining S_0 to S_2 in one certified step by ρ_{02} or in two certified steps by ρ_{01} followed by ρ_{12} determines the same terminal four-face packet and the same local $\mathsf{Real}^{\uparrow}$ *seal. The comparison is confined to finite request sections and their displayed support rows; it does not identify quotient streams, select limits, compare host reals, use countable choice, add metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, or assert* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read both refinement routes on their visible finite supports. By Definition 301.242, each restriction of the one-step route and each restriction of the two-step route is exactly the packet obtained by restricting the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and support rows.

Next compare the request spines. Apply Definition 126.597 to identify the finite $\mathsf{StreamName}^{\uparrow}$ windows carried by the direct certificate with the windows obtained after the two successive re-

strictions. Apply Definition 291.90 to the dyadic endpoint, radius, and finite-distance ledger. The common-scale window seen by ρ_{02} is therefore the same dyadic window seen by $\rho_{12} \circ \rho_{01}$.

The local Real seal is then forced on every overlap. Use Definition 301.243 on the three sections S_0, S_1, and S_2 to identify the pairwise Real-seal-visible packets on triple overlaps. Finally apply Definition 38.796 to read that common overlap packet as the local $\mathsf{Real}^{\uparrow}$ threshold seal for the same four visible faces and support rows.

It remains only to repack the terminal object. The direct refinement route and the two-step refinement route now have identical StreamName request windows, dyadic common-scale rows, RegSeqRat classifier and readback rows, local Real seal rows, and support coordinates on every finite member and overlap. Applying Definition 301.242 once more, both routes determine the same terminal four-face packet.

All cited inputs are finite restriction, finite cocycle, common-scale composition, and local Real threshold statements over the displayed BEDC rows. None of these constructions carries quotient stream equality, selected-limit data, host real equality, countable-choice data, metric-completeness rows, aggregate formalstatus rows, aggregate bridge rows, or a $\mathsf{Real}^{\uparrow}$ completeness field, so the composition law does not claim any such structure. □

Theorem 301.245 (RegSeqRat four-face finite-cover induction)**.** *Let $\mathcal{S}$ be a finite family of compatible accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections over the visible four-face route*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Assume that the empty and singleton subfamilies are the displayed local packets of Definition 301.240. Assume also that the successor clause glues any previously accepted subcover to one additional accepted section along overlap rows preserving the finite $\mathsf{StreamName}^{\uparrow}$ *request windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

Then the whole finite cover $\mathcal{S}$ determines exactly one accepted $\mathsf{Real}^{\uparrow}$ *terminal seal over the same four visible faces. Moreover, every accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal whose visible support is the cover $\mathcal{S}$ is obtained by this finite-cover induction from the local empty, singleton, and successor clauses.*

The claim is only the finite descent principle for the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *section cover. It does not identify quotient streams, select limits, compare host real equality, use countable choice, assume ambient or metric completeness, aggregate formalstatus rows, or introduce standard bridge rows.*

Proof. Put the finite cover into a section-by-section descent order. Apply Definition 126.600 to replace the cover by its finite request normal form. The empty and singleton entries are then exactly the local four-face packets supplied by Definition 301.240.

For the successor step, suppose a subcover has already produced an accepted terminal $\mathsf{Real}^{\uparrow}$ seal. Glue this seal to the next section along the stated overlap rows. The preservation hypotheses give the same BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ rows on the common support, so Definition 301.242 reads both sides on the same restricted four-face packet.

The overlap condition is coherent because the pairwise restrictions satisfy Definition 301.243. Hence the new section and the previously accepted subcover have a common terminal overlap packet. Applying Definition 301.241 gives one accepted $\mathsf{Real}^{\uparrow}$ terminal seal for the enlarged subcover, and uniqueness prevents any second seal from being carried by the same visible rows.

Iterating this finite successor construction reaches the full cover $\mathcal{S}$. Refining the descent order does not change the resulting seal: adjacent regroupings are identified by Definition 301.244, while

all restrictions back to smaller subcovers are controlled by Definition 301.242. Thus the terminal seal is independent of the chosen finite normal form.

Finally let T be any accepted $\mathsf{Real}^{\uparrow}$ terminal seal over the same cover. Restrict T to every empty, singleton, and successor subcover in the normal form. The local gluing theorem supplies the base packets, the restriction theorem identifies the visible rows, the cocycle theorem supplies the overlap compatibility, and gluing uniqueness identifies each successor seal with the induction seal. Therefore T is obtained by the same finite-cover induction.

Every step uses only finite request normal forms, local four-face gluing, restriction functoriality, cocycle compatibility, uniqueness, and certified refinement composition. None of these cited inputs contains a coordinate for quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, aggregate formalstatus rows, or standard bridge rows, so the induction theorem does not claim any of that structure. □

Theorem 301.246 (RegSeqRat four-face finite-cover normal form)**.** *A finite family $\mathcal{S}$ of compatible accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections presents an accepted* $\mathsf{Real}^{\uparrow}$ *terminal seal precisely when its visible support is the following finite row data:*

1. *the finite* $\mathsf{StreamName}^{\uparrow}$ *request windows;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledgers;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows;*
4. *the local* $\mathsf{Real}^{\uparrow}$ *seal rows; and*
5. *the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support rows.*

These rows must satisfy the local-gluing hypotheses of Definition 301.240, the restriction hypotheses of Definition 301.242, the cocycle hypotheses of Definition 301.243, the refinement-composition hypotheses of Definition 301.244, and the finite-cover induction hypotheses of Definition 301.245. No additional row for quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric completeness, aggregate formalstatus data, aggregate bridge data, or standard bridge data is present or necessary.

Equivalently, those finite rows are necessary and sufficient for the $\mathsf{Real}^{\uparrow}$ *terminal seal determined by the RegSeqRat-to-Real four-face cover, and any other accepted terminal seal with exactly those visible rows is the same seal.*

Proof. For necessity, start with an accepted $\mathsf{Real}^{\uparrow}$ terminal seal presented by the finite compatible family $\mathcal{S}$. Apply Definition 301.245 to express the seal by empty, singleton, and successor cover clauses. Project the empty and singleton clauses through Definition 301.240; project the successor restrictions through Definition 301.242; and project the overlap rows through Definition 301.243. The visible support is therefore exactly the finite StreamName, DyadicRatCore, RegSeqRat, Real, and shared BEDC support rows listed in the statement.

For sufficiency, suppose the displayed finite rows and the named compatibility hypotheses are given. Package the empty, singleton, successor, cocycle, and refinement-composition data as the input cover for Definition 301.245. That theorem constructs an accepted $\mathsf{Real}^{\uparrow}$ terminal seal over the four visible faces.

It remains to identify possible alternatives. If T is any other accepted terminal seal with exactly the same visible rows, then its restrictions to the finite request windows have the same local gluing rows, the same finite restriction rows, and the same cocycle rows. Apply Definition 301.241 on each successor gluing step and then along the whole finite-cover induction order. The resulting seal is the one constructed from the displayed rows.

The excluded structures cannot enter the construction because every cited input theorem is stated over finite visible support rows. Thus quotient stream identification, selected-limit data, host real equality, countable-choice data, ambient or metric completeness, aggregate formalstatus rows, aggregate bridge rows, and standard bridge rows are neither consumed nor produced. □

Theorem 301.247 (RegSeqRat finite-cover normal form StreamName support refinement)**.** *Let $\mathcal{S}$ be a finite compatible family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections whose visible rows satisfy the normal-form clauses of Definition 301.246. Let $\mathcal{S}'$ be obtained from $\mathcal{S}$ by refining only the terminal* $\mathsf{StreamName}^{\uparrow}$ *request windows and by preserving, on every refined support row, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, the local* $\mathsf{Real}^{\uparrow}$ *seal rows, and the shared* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support.*

Assume the refined terminal windows satisfy Definition 126.657 and that transport from $\mathcal{S}$ to $\mathcal{S}'$ satisfies Definition 126.658. Then $\mathcal{S}$ and $\mathcal{S}'$ determine the same accepted $\mathsf{Real}^{\uparrow}$ *terminal seal exactly when the preserved* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger rows agree with the local* $\mathsf{Real}^{\uparrow}$ *seal rows on the shared support.*

Equivalently, the finite-cover normal form factors through the StreamName terminal support surface: refining finite terminal windows changes no accepted $\mathsf{Real}^{\uparrow}$ *terminal seal unless it changes the shared DyadicRatCore ledger or the local Real seal row read on that support. The refinement surface has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, ambient or metric completeness, aggregate formalstatus rows, aggregate bridge rows, or standard bridge rows.*

Proof. First normalize the original cover. Apply Definition 301.246. The accepted $\mathsf{Real}^{\uparrow}$ terminal seal is carried by exactly the finite StreamName request windows, the DyadicRatCore endpoint, radius, and finite-distance ledgers, the RegSeqRat classifier-readback rows, the local Real seal rows, and the shared BEDC support rows.

Next read the refined windows through the StreamName terminal support surface. By Definition 126.657, every accepted support coordinate of $\mathcal{S}'$ is one of the same four visible faces or one of the same displayed support rows. By Definition 126.658, the componentwise transport from $\mathcal{S}$ to $\mathcal{S}'$ preserves that support packet whenever those rows agree after replay.

Now compare the consumer ledger. Apply Definition 291.118 to the consumer read of each refined terminal packet. The consumer can read exactly the DyadicRatCore ledger rows, the StreamName finite-window rows, the RegSeqRat classifier-readback rows, the local Real seal rows, and the shared support. Therefore agreement of the DyadicRatCore endpoint and radius ledger with the local Real seal rows on the shared support is sufficient to make the refined consumer packet expose the same finite inventory as the original packet.

With the same inventory in hand, use Definition 301.240 on each refined local section and Definition 301.242 on the overlaps. The transported support rows give the same pairwise restrictions as before, so Definition 301.243 supplies the same finite cocycle. Applying Definition 301.241 along the finite-cover induction order identifies the accepted terminal seal produced by $\mathcal{S}'$ with the seal produced by $\mathcal{S}$.

Conversely, suppose the two covers determine the same accepted $\mathsf{Real}^{\uparrow}$ terminal seal. Project that common seal back through Definition 301.246 and then through Definition 291.118. Since the StreamName support exhaustion leaves no additional consumer coordinate, any accepted difference between the two finite presentations must be visible in the DyadicRatCore ledger rows or in the local Real seal rows. Equality of the terminal seal therefore forces the endpoint and radius ledger and the local Real seal rows to agree on the shared support.

All implications use only finite StreamName support exhaustion, support transport stability, the DyadicRatCore consumer normal form, local RegSeqRat gluing, finite restriction functoriality, cocycle compatibility, and gluing uniqueness. None of these cited surfaces supplies quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, aggregate formalstatus rows, aggregate bridge rows, or standard bridge rows, so the refinement theorem does not claim any of that structure. □

Theorem 301.248 (RegSeqRat finite-window map preservation)**.** *Let S be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carried by a finite* BHist *packet with its displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and finite-distance ledger, its* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, and its* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support. Let M be a uniformly controlled finite-window map row on the same support. Assume M sends every displayed StreamName window to a displayed StreamName window, transports the DyadicRatCore tolerance ledger by the stated finite modulus row, and preserves the RegSeqRat classifier and regularity rows on each finite observation window.*

Then the readback row obtained by applying M to S is again an accepted $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over a finite four-face packet*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

with the transported BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support. The construction is finite-window inductive and introduces no quotient stream equality, selected limit, countable-choice row, host function equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or second* $\mathsf{Real}^{\uparrow}$ *seal coordinate.*

Proof. Start from the finite request source section of Definition 301.139. It places the source readback on one finite packet carrying the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat classifier and regularity rows, and the displayed support.

Apply the map row on the StreamName and dyadic faces. The StreamName pullback and lock results, Definition 126.586 and Definition 126.589, keep the transported windows as finite observed rows. The dyadic source uniqueness theorem Definition 291.84 keeps the transported endpoint, radius, and distance ledger on the same finite tolerance surface.

Now perform induction over the finite observation window family. The base window is the transported singleton read supplied by the map row. In the induction step, the preservation hypotheses identify the new overlap rows with the already transported StreamName and DyadicRatCore rows, while the RegSeqRat classifier and regularity rows are preserved by assumption.

The induction is glued by Definition 301.240 and Definition 301.227. Those theorems turn compatible local four-face packets into one accepted terminal readback without adding a quotient stream, selected limit, choice row, host equality, ambient completeness datum, or second Real seal.

Finally use Definition 301.236 and Definition 301.238. They identify the transported readback as a single RegSeqRat source section over the same StreamName, DyadicRatCore, RegSeqRat, and Real faces, so the output of M is accepted as a $\mathsf{RegSeqRat}^{\uparrow}$ readback row. □

Theorem 301.249 (RegSeqRat uniformly controlled Real seal handoff)**.** *Let S be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose terminal consumer is a Phase* $\mathsf{Real}^{\uparrow}$ *and Completion seal, and let M be a uniformly controlled finite-window map row satisfying the hypotheses of Definition 301.248. Then the* $\mathsf{Real}^{\uparrow}$*-facing seal read of the transported row factors through exactly the four finite faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the map-control row M and the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *support.*

This handoff is the RegSeqRat-side input needed by the $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *route: it reads the mapped regular Cauchy data through StreamName windows, DyadicRatCore tolerances, the RegSeqRat accepted readback, and the terminal Real seal. It exports no quotient stream, host function equality, selected limit, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or independent Real completion coordinate.*

Proof. Apply Definition 301.248 to the source readback and the map-control row. The result is an accepted transported $\mathsf{RegSeqRat}^{\uparrow}$ readback over the same finite support and the same four visible faces.

Read the transported row at the Real boundary. The terminal exactness theorem Definition 301.239 identifies its accepted Real seal precisely with the displayed StreamName, DyadicRatCore, RegSeqRat, and Real rows. The source uniqueness theorem Definition 301.237 rules out a second RegSeqRat source over those rows.

Next compare this handoff with the local cover surface. The finite-cover gluing theorem Definition 301.227 and the local four-face coverage theorem Definition 301.240 keep the seal read invariant under finite cover refinement of the transported windows. Thus every Real-facing read of the mapped row is recovered from the same finite four-face packet and the same map-control row.

Finally point the packet toward the completion-extension consumer. The $\mathsf{CauchyContinuousExtension}^{\uparrow}$ regular source handoff of Definition 1632.3 in chapter 1632 consumes regular source data only through StreamName finite windows, a DyadicRatCore tolerance ledger, a Cauchy-continuous map row, and the resulting completion surface. The handoff above supplies exactly the RegSeqRat source side of that route before any completed extension is read.

Since each cited step is a finite BEDC row calculation, no quotient stream, host function equality, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ completeness, or independent completion coordinate is available to enter the seal read. □

Theorem 301.250 (RegSeqRat uniform-continuity window modulus composition)**.** *Let S be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row with its displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Let M_0 and M_1 be uniformly controlled finite-window map rows, each satisfying the hypotheses of Definition 301.248. Assume that the output four-face packet of M_0 is the input four-face packet of M_1, with the same StreamName, DyadicRatCore, RegSeqRat, Real, history, continuity, package, and local NameCert rows exposed on the shared boundary.*

Then the composed finite-window row $M_1 \circ M_0$ preserves the accepted $\mathsf{RegSeqRat}^{\uparrow}$ *readback of S and exports a single Phase* $\mathsf{Real}^{\uparrow}$ *and Completion handoff through exactly the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. The composed route introduces no quotient stream, selected limit, countable-choice row, host function equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or second Real seal.*

Proof. Apply Definition 301.248 to S and M_0. This gives an accepted intermediate RegSeqRat readback over the output packet of M_0, with the transported StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat classifier, Real face, BHist, hsame, Cont, Pkg, and local NameCert rows still displayed as finite BEDC data.

Use the boundary hypothesis to read that intermediate output packet as the input packet of M_1. A second application of Definition 301.248 therefore produces the accepted readback for $M_1 \circ M_0$. The four-face refinement functoriality theorem Definition 301.238 identifies the two finite transports with one composed route rather than two unrelated readback choices.

The cover bookkeeping is finite. Apply Definition 301.227 to glue the modulus windows used by M_0 and M_1 into the single observed finite cover seen by the composed row. This keeps the overlap checks inside the displayed StreamName, DyadicRatCore, RegSeqRat, and Real faces and does not add a new ambient equality principle.

Finally apply Definition 301.249 to the composed accepted readback. The handoff theorem reads the terminal seal only through the displayed four-face packet and its BHist, hsame, Cont, Pkg, and NameCert support. Since both preservation steps and the finite-cover gluing are internal row calculations, there is no quotient stream, selected limit, countable-choice row, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or second Real seal available for the composed route. □

Theorem 301.251 (RegSeqRat uniform-continuity modulus fusion)**.** *In the setting of Definition 301.250, the dyadic modulus schedules of the two uniformly controlled finite-window rows fuse into one displayed finite support row consumed by the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The fused row is read over the shared*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

boundary together with the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. It exports no selected limit, quotient stream, host function equality, ambient completeness coordinate, or second Real seal.*

Proof. Begin with Definition 301.250. It identifies the composite route $M_1 \circ M_0$ as one accepted finite-window transport over the shared StreamName, DyadicRatCore, RegSeqRat, and Real faces.

The modulus bookkeeping is the finite cover bookkeeping. Apply Definition 301.227 to the two observed modulus-window families. The gluing row records the common finite support on which the output windows of M_0 are the input windows of M_1, so the two dyadic schedules are consumed as one displayed finite support row.

Finally use Definition 301.249. The terminal Real seal reads the composed accepted row only through the four visible faces and the structural support rows. Since the fused support row is already the support used by that seal, there is no source for a selected limit, quotient stream, host function equality, ambient completeness coordinate, or second Real seal. □

Theorem 301.252 (RegSeqRat tail-modulus positive cofinality)**.** *Let S be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carried by displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the supporting* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows. Suppose the displayed modulus schedule for S dominates a later positive dyadic tolerance request ε. Then there is a finite later window whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback refines the earlier ε request before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The refinement is read through the same four faces

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the same structural support rows. It introduces no quotient stream, selected limit, countable-choice row, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or independent Real seal.*

Proof. Start with the displayed positive dyadic request ε. Since it is dominated by the modulus schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger contains a finite later tolerance row whose request is at most ε on the same StreamName support.

Use the common-window transport route for $\mathsf{RegSeqRat}^{\uparrow}$ readbacks. The later window is transported along the displayed hsame row and replayed through Cont, so the classifier remains on the same StreamName, DyadicRatCore, and RegSeqRat faces.

Apply the finite-cover gluing used in Definition 301.251. The gluing row places the later window and the earlier request on one finite support, and the dominance relation makes the later readback a refinement of the earlier positive request.

Finally use the Real-seal handoff Definition 301.249. The terminal seal consumes only the displayed four-face route, so the cofinal later window is available before any Real seal read and no quotient stream, selected limit, choice schedule, ambient completeness, or second seal can enter the route. □

Theorem 301.253 (RegSeqRat uniform-continuity composition terminal seal)**.** *Two displayed uniform-continuity packets over* $\mathsf{RegSeqRat}^{\uparrow}$ *compose to a single terminal* $\mathsf{Real}^{\uparrow}$*-seal handoff exactly through the shared* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budgets,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The composed consumer receives one finite route, not two independently selected limit reads.*

The terminal seal contains no selected limit, quotient stream equality, countable choice, ambient Real completeness, or standard bridge coordinate.

Proof. Start with preservation, Definition 301.248, for the two displayed uniform-continuity packets. The first packet produces the finite output windows that the second packet consumes, all inside the visible RegSeqRat readback surface.

Apply the Real-seal handoff theorem Definition 301.249. It forces every terminal consumer to pass through the displayed StreamName, DyadicRatCore, RegSeqRat, and Real faces together with their support rows.

The window-modulus composition theorem Definition 301.250 identifies the two packet routes as one composed finite-window route. Then Definition 301.251 fuses the two dyadic modulus schedules into the single support row consumed by the terminal seal.

Therefore the composed Real-seal handoff is terminal and finite. Since the route has already exhausted its visible windows, tolerance budgets, classifier and readback rows, and Real seal, none of the excluded quotient, choice, completeness, selected-limit, or bridge coordinates can supply another composition read. □

Theorem 301.254 (RegSeqRat finite-cover induction diagonal)**.** *Let* $\{S_i\}_{i\in I}$ *be a finite family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections for one Phase* $\mathsf{Real}^{\uparrow}$ *and Completion consumer. Assume every* S_i *is read through a common* $\mathsf{StreamName}^{\uparrow}$ *window family, a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledger, a local* $\mathsf{Real}^{\uparrow}$ *seal row, and the shared support*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Assume also that the accepted pairwise overlap rows glue by the finite-cover gluing theorem, that adjoining one more section preserves the induction-step rows, that the gluing operation is associative on displayed overlaps, and that every two local presentations admit the common-refinement rows of the RegSeqRat cover surface. Then finite induction over I *produces a single diagonal terminal window through which all local* $\mathsf{Real}^{\uparrow}$ *seal reads factor.*

The diagonal terminal window exposes exactly the common $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledgers, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and displayed support rows named above. It exports no quotient stream equality, host real equality, selected limit,*

countable-choice schedule, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct on the finite cover. In the empty boundary there is no terminal seal read to compare, and in the singleton case the accepted section itself supplies the diagonal terminal window with its displayed StreamName, DyadicRatCore, RegSeqRat, Real, and support rows.

For the induction step, suppose a finite subcover has already been packed into one diagonal terminal window. Adjoin the next accepted section S_j. The common-refinement rows place the old diagonal window and S_j over one common $\mathsf{StreamName}^{\uparrow}$ window family and the same displayed dyadic tolerance ledger.

Apply the RegSeqRat finite-cover gluing row to the old diagonal window and the new local section. The induction-step hypotheses preserve the classifier rows, regularity ledgers, local Real seal rows, and the shared BHist, hsame, Cont, Pkg, and NameCert support while the two packets are glued.

Associativity identifies the result with any parenthesized order of adjoining the same finite sections. Thus the terminal window obtained after the last induction step is independent of the chosen bracketing and is the single diagonal window for the whole family.

Finally use the existing common-refinement determinacy and Real seal handoff rows. They force every local seal consumer to factor through the diagonal window just constructed, and they leave no coordinate for quotient streams, host real equality, selected limits, countable choice, metric completeness, completion points, or ambient Real completeness. □

Theorem 301.255 (RegSeqRat finite-cover diagonal Real-seal exhaustion)**.** *Let* $\{S_i\}_{i\in I}$ *be a finite family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections satisfying the common* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity, local* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support hypotheses of Definition 301.254. Then every* $\mathsf{Real}^{\uparrow}$*-facing read from any local section* S_i *factors through the diagonal terminal window produced by Definition 301.254.*

The factorization is exhaustive for the displayed family: no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle is exported by a local seal read outside the diagonal packet.*

Proof. Invoke Definition 301.254 for the displayed finite family. It supplies one diagonal terminal window carrying the same StreamName, DyadicRatCore, RegSeqRat, Real, and support rows as the local sections.

Now read an arbitrary $\mathsf{Real}^{\uparrow}$-facing consumer from a chosen local section S_i. The existing common-refinement determinacy rows in the RegSeqRat cover surface place that local read and the diagonal window over the same finite StreamName window family and the same dyadic tolerance ledger.

Use the Real seal handoff rows of the RegSeqRat chapter. They route the local Real seal through the displayed RegSeqRat classifier and regularity ledger before any terminal read is exposed, so the consumer cannot bypass the diagonal packet once the common-refinement row has been fixed.

Since i was arbitrary, every local Real-facing seal read factors through the same diagonal terminal window. The carrier has no coordinate for quotient stream equality, selected limits, countable-choice schedules, metric-completeness data, completion points, or ambient Real completeness, so none of those objects is exported by the exhaustion route. □

Theorem 301.256 (RegSeqRat finite-cover diagonal tail-budget preservation)**.** *Let* $\{S_i\}_{i\in I}$ *be a finite family of accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *terminal sections satisfying the hypotheses of Definition 301.254. The diagonal terminal window produced by finite-cover induction preserves every local tail-budget*

row when read through the common $\mathsf{StreamName}^{\uparrow}$ *window family and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Consequently no local section may contribute a separate selected limit, quotient stream, countable-choice schedule, host sequence equality, metric-completeness row, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate to the terminal Real seal route.*

Proof. Invoke Definition 301.254. The induction constructs a single diagonal terminal window over the common StreamName window family, the same DyadicRatCore tolerance ledger, the RegSeqRat classifier and regularity rows, the local Real seal rows, and the shared support rows.

Now apply Definition 301.255. Every Real-facing read from a local section factors through that diagonal window, so the local tail-budget row is read through the same StreamName and DyadicRatCore data rather than through a private terminal coordinate.

The StreamName finite-request pullback and exactness rows identify the public request section used by the diagonal window, while the DyadicRatCore tolerance-halving handoff rows keep tolerance refinements inside the displayed ledger. Thus finite-cover induction preserves the tail-budget rows already visible in the common window family.

Since the diagonal packet has no coordinate for selected limits, quotient streams, countable choice, host sequence equality, metric completeness, completion points, or ambient Real completeness, no local section can export one of those objects along the tail-budget route. □

Theorem 301.257 (RegSeqRat tail-shift stability for scheduled finite windows)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let t be a displayed unary tail index for the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the shifted finite window exposes exactly the endpoint rows, dyadic-radius rows, regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance obtained by transporting the original window through the shifted* $\mathsf{StreamName}^{\uparrow}$ *schedule. Then the shifted window preserves the* $\mathsf{RegSeqRatClassifier}$ *read of P.*

Equivalently, every accepted classifier read on the original scheduled finite window factors through the shifted StreamName window and returns the same RegSeqRat finite source surface before any $\mathsf{Real}^{\uparrow}$ *seal row can be consumed. The stability statement exports no quotient stream equality, host real equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the original scheduled-window rows of the accepted packet. The visible source data are the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius rows, the regularity ledgers, the classifier row, and the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Use the finite-window transport obligation for the displayed unary tail index t. It transports endpoint rows, dyadic-radius comparisons, and regularity ledgers componentwise along the shifted StreamName window, preserving exactly the rows that the classifier is allowed to inspect.

Apply classifier transport to the shifted window. Since the transported hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows are part of the same finite packet, the classifier read cannot add a second schedule, quotient stream, selected limit, or host real coordinate.

Finally close by common-refinement determinacy. The original scheduled window and the shifted window restrict to the same displayed finite RegSeqRat source surface on their common refinement, so their classifier reads agree before any terminal $\mathsf{Real}^{\uparrow}$ seal is opened. □

Theorem 301.258 (RegSeqRat two-shift common-refinement compatibility)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let t_0 and t_1 be two displayed unary tail indices for the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the two shifted finite windows admit one displayed common finite refinement on which they expose the same transported StreamName schedule, the same* $\mathsf{DyadicRatCore}^{\uparrow}$

endpoint and radius rows, the same regularity ledgers, the same componentwise hsame *rows, the same displayed* Cont *routes, the same* Pkg *provenance, and the same local* NameCert *rows. Then the two shifted reads determine the same* RegSeqRatClassifier *value before any* $\mathsf{Real}^{\uparrow}$ *seal row is consumed.*

Equivalently, whenever two displayed tail shifts of an accepted RegSeqRat packet are compared through a common finite refinement carrying the full StreamName, DyadicRatCore, regularity, transport, continuation, package, and naming support, the classifier has only one finite source row to read. The theorem exports no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 301.257 to the t_0-shifted window. Its classifier read is the original RegSeqRat finite source surface transported along the first displayed StreamName tail index.

Apply the same theorem to the t_1-shifted window. Its classifier read is likewise the original finite source surface transported along the second displayed tail index.

Use the common finite refinement to compare the two transported supports. By hypothesis the StreamName schedule, DyadicRatCore endpoint and radius rows, regularity ledgers, hsame, Cont, Pkg, and local NameCert rows coincide on that refinement, so common-refinement determinacy gives one visible finite source row for both shifted windows.

The classifier can inspect only that visible finite RegSeqRat source surface. Since neither shifted read opens a quotient stream, selected limit, choice schedule, metric-completeness datum, or ambient Real-completeness coordinate, no further row remains that could distinguish the two classifier values before the $\mathsf{Real}^{\uparrow}$ seal is consumed. □

Theorem 301.259 (RegSeqRat tail-shift finite-window stability)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let t_0 and t_1 be two displayed unary tail indices over the same* $\mathsf{StreamName}^{\uparrow}$ *source and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Suppose both shifted finite windows expose accepted endpoint rows, dyadic-radius rows, regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows for P. Then the two shifts admit a common finite window whose readback preserves the* RegSeqRatClassifier *value and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal route.*

The common window exports only the finite StreamName source surface, dyadic tolerance ledger, regular-rational readback, transport, replay, provenance, and local naming rows. It does not expose quotient stream equality, selected limits, countable choice, metric-completeness data, completion points, or ambient Real-completeness coordinates.

Proof. Apply Definition 301.257 to the t_0-shifted window. The classifier read is the original finite RegSeqRat source surface transported along the first displayed StreamName tail index.

Apply the same theorem to the t_1-shifted window. The second read is transported from the same original source and the same dyadic tolerance budget, so it has the same kind of visible support.

Now use Definition 301.258. The two accepted shifted windows have a common finite refinement on which the StreamName schedule, DyadicRatCore rows, regularity ledgers, hsame, Cont, Pkg, and local NameCert rows determine one visible classifier read.

Finally project the finite carrier rows. Since the classifier is already fixed on the common window, the downstream $\mathsf{Real}^{\uparrow}$ seal route is read through the same finite support and no quotient stream, selected limit, choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate remains available. □

Theorem 301.260 (RegSeqRat tail-shift diagonal budget)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let t be a displayed unary tail index for the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. If the scheduled tail shift is read through a common finite window, then the shifted read preserves the diagonal dyadic tolerance budget consumed by the downstream* $\mathsf{Real}^{\uparrow}$ *seal route. The preserved budget is the same componentwise ledger used by the finite-cover diagonal induction of Definition 301.254; it is transported through the StreamName schedule, the DyadicRatCore endpoint and radius rows, the RegSeqRat regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The statement exports no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate.

Proof. Induct over the displayed finite window. In the initial window, the diagonal dyadic tolerance row is one of the visible DyadicRatCore budget rows consumed by the accepted RegSeqRat packet.

For the tail step, apply Definition 301.257. It transports the StreamName schedule, endpoint rows, dyadic-radius rows, regularity ledgers, and support rows along the displayed unary tail index.

Compare two shifted reads by Definition 301.258 and then restrict to the common finite window using Definition 301.259. The common refinement keeps the classifier read and downstream RealUp seal route on the same finite support.

Finally invoke Definition 301.254. The displayed diagonal budget rows are transported componentwise along the same finite window, so the Real seal consumes the preserved dyadic tolerance budget rather than an ambient limit or completeness coordinate. □

Theorem 301.261 (RegSeqRat tail-window cofinality)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let W_0 and W_1 be accepted scheduled tail windows over the same* $\mathsf{StreamName}^{\uparrow}$ *source and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Then W_0 and W_1 admit a common finite cofinal window whose* RegSeqRatClassifier *read agrees and whose downstream* $\mathsf{Real}^{\uparrow}$ *seal route is the same displayed route.*

The common cofinal window exposes only the shifted StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat regularity ledgers, componentwise hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected limits, countable choice, metric-completeness data, completion points, or ambient Real completeness.*

Proof. Apply tail-shift stability, Definition 301.257, to both scheduled tail windows. Each accepted window reads the same original finite RegSeqRat source surface through its displayed StreamName tail.

Use two-shift common-refinement compatibility, Definition 301.258. Since the two windows share the StreamName source and DyadicRatCore tolerance budget, their displayed supports admit a common finite refinement with one classifier read.

Next apply finite-window stability, Definition 301.259. It identifies the common refinement as a window on which the classifier read and downstream Real seal route are preserved together.

Finally invoke the diagonal budget theorem, Definition 301.260. The dyadic tolerance budget consumed by the downstream Real seal is the same componentwise ledger on the common cofinal window, leaving no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate. □

Theorem 301.262 (RegSeqRat tail-shift synchronized readback)**.** *Let P and Q be accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packets whose displayed tail-shift schedules share the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and the same*

dyadic tail budget. If both schedules expose accepted endpoint rows, regularity ledgers, componentwise hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows on that shared window, then both schedules read back through one RegSeqRat row before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

The synchronized readback exports only the shared finite StreamName window, the dyadic tail budget, the regular rational classifier row, and the displayed transport, replay, provenance, and naming rows. It does not expose quotient stream equality, selected limits, countable choice, metric-completeness data, completion points, or ambient Real-completeness coordinates.

Proof. Apply Definition 301.257 to each displayed schedule. Each accepted tail shift transports the original finite source surface through the shared StreamName window and preserves the classifier rows allowed by the RegSeqRat packet.

Use Definition 301.258. The two schedules already share the finite StreamName window and dyadic tail budget, so their common refinement has the same endpoint rows, regularity ledgers, transport rows, replay rows, provenance, and local naming.

Apply Definition 301.259 on that common window and then Definition 301.260. The classifier read and the diagonal dyadic budget are synchronized on one finite RegSeqRat row before the downstream Real seal route is opened.

Since the proof has used only displayed finite rows, no quotient stream, selected limit, choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate remains available to distinguish the two readbacks. □

Theorem 301.263 (RegSeqRat tail-shift finite-section induction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let $t_0, \ldots, t_k$ be a finite displayed family of unary tail indices for the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P. Suppose the shifted StreamName windows share accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows on their displayed common windows. Then induction over the finite family gives one synchronized finite readback route for all displayed shifts before the downstream* $\mathsf{Real}^{\uparrow}$ *seal is opened.*

The synchronized route exposes only the finite StreamName windows, dyadic budgets, regular-rational classifier rows, transport, replay, provenance, and local naming rows. It does not expose quotient stream equality, selected limits, countable choice, metric completeness, completion points, or ambient Real completeness.

Proof. Induct over the displayed finite family of tail indices. In the empty or singleton case, apply Definition 301.257 to read the shifted window through the original finite RegSeqRat source surface.

For the induction step, assume a synchronized readback route has already been built for the initial family. Compare the next shifted window with that route by Definition 301.258; the hypotheses provide the shared StreamName windows, DyadicRatCore budgets, regularity ledgers, hsame, Cont, Pkg, and local NameCert rows needed for the common refinement.

Apply Definition 301.259 to keep the extended family on one finite window, and then use Definition 301.262 to repack the accumulated family and the next shifted window as one synchronized readback. No proof step introduces quotient stream equality, selected limits, choice, metric completeness, completion points, or ambient Real completeness. □

Theorem 301.264 (RegSeqRat tail-shift Real-seal classifier lock)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let $t_0, \ldots, t_k$ be a finite displayed family of tail shifts admitted by Definition 301.263. Before any Real consumer may read the shifted sequence, the synchronized route must preserve the same* $\mathsf{StreamName}^{\uparrow}$ *finite section, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal classifier.*

Equivalently, every admitted tail-shift family is locked to the four displayed objects already visible in the finite-section route. The lock exports no quotient stream equality, selected limit, countable choice, metric completeness, completion point, or ambient Real completeness.

Proof. Apply Definition 301.263 to the displayed family. It gives one synchronized finite readback route whose available rows are the finite StreamName windows, dyadic budgets, regular-rational classifier rows, transport, replay, provenance, and local naming.

Next route that synchronized tail through the dyadic tolerance ledger. The same $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the budget consumed by each shifted finite section, so no shifted read can replace the ledger by a selected limit or by a host Cauchy object.

The readback and seal are downstream of that ledger. The $\mathsf{RegSeqRat}^{\uparrow}$ row is the only regular-rational classifier available to the synchronized route, and the terminal $\mathsf{Real}^{\uparrow}$ seal is opened only after the same finite section, tolerance row, and readback row have been displayed.

Thus a Real consumer receives the shifted family only through the locked four-object surface $\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$. Since the proof never adds quotient stream equality, selected limits, countable choice, metric completeness, completion points, or ambient Real completeness, no additional classifier coordinate remains. □

Theorem 301.265 (RegSeqRat tail-shift cofinal fusion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Let $t_0, \ldots, t_k$ and $u_0, \ldots, u_m$ be two finite displayed families of unary tail shifts over the same* $\mathsf{StreamName}^{\uparrow}$ *source and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Suppose both families expose the transported endpoint rows, dyadic-radius rows, regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of P. Then the two synchronized RegSeqRat readback routes admit one finite cofinal fusion route preserving the* RegSeqRatClassifier *value and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

The fusion route exports only the shared finite StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat regularity readback, transport, replay, provenance, and local naming rows. It exports no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate.

Proof. Apply Definition 301.263 to the first displayed family and again to the second. This gives two synchronized finite readback routes over the same StreamName source and DyadicRatCore tolerance budget.

Next use Definition 301.262 to compare the two synchronized routes on their shared finite support. The visible rows are exactly the transported endpoint rows, regularity ledgers, componentwise hsame rows, displayed Cont routes, Pkg provenance, and local NameCert rows, so a common cofinal window can read both routes as one RegSeqRat row.

Finally apply Definition 301.264. The fused route remains locked to the same $\mathsf{StreamName}^{\uparrow}$ finite section, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal classifier before any Real-facing consumer opens the seal.

All rows used by the fusion are finite BEDC rows already displayed by the packet. Hence the fusion introduces no quotient stream equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient Real-completeness coordinate. □

Theorem 301.266 (RegSeqRat tail-shift regularity refinement)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let $t_0, \ldots, t_k$ be a finite displayed family of unary tail shifts over the same* $\mathsf{StreamName}^{\uparrow}$ *source and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Suppose every shifted window exposes the*

transported endpoint rows, dyadic-radius rows, regularity ledgers, componentwise hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of P. Then the finite-section synchronized route preserves the regularity ledger on a common refinement before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The refinement is internal to the displayed RegSeqRat source surface. It does not identify quotient streams, select limits, use choice, appeal to metric completeness, or import ambient Real completeness.

Proof. Apply Definition 301.263 to the finite family of displayed tail shifts. This gives one synchronized finite readback route whose visible rows are the StreamName windows, dyadic budgets, regular-rational classifier rows, transport, replay, provenance, and local naming rows.

Next apply Definition 301.260 along each shifted window as it enters the synchronized route. The dyadic tolerance row used by the downstream seal is the same componentwise budget row carried by the RegSeqRat regularity ledger, so synchronization does not replace the ledger with a selected limiting object.

Finally compare the resulting common refinement with Definition 301.254. The finite-cover diagonal induction reads the same displayed endpoint, radius, and regularity rows on every finite section, and therefore preserves the regularity ledger before the Real seal is opened.

All rows used in the argument are finite BEDC rows of the packet. Thus no quotient stream equality, selected limit, choice principle, metric-completeness datum, completion point, or ambient Real-completeness coordinate is introduced. □

Theorem 301.267 (RegSeqRat tail-shift choice-free finite window)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let $t_0, \ldots, t_k$ be a finite displayed family of tail shifts over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Suppose the shifted windows expose the transported endpoint rows, dyadic-radius rows, regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of P. Then the family has a synchronized regularity readback whose downstream* $\mathsf{Real}^{\uparrow}$ *seal consumes only the displayed finite window.*

The readback exports no selected limit, quotient stream equality, countable-choice schedule, completion point, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 301.263 to the displayed finite family. This supplies one synchronized finite readback route for all tail shifts before the downstream Real seal is opened.

Use Definition 301.266 to preserve the regularity ledger on the common refinement. Then apply Definition 301.261 to keep the synchronized readback inside the same finite StreamName schedule and DyadicRatCore tolerance budget.

When the terminal radius is read by a Real-facing consumer, apply Definition 291.104. The half-radius row is already determined by the finite DyadicRatCore common-scale refinement, so the RegSeqRat route has no need for a selected limit, quotient stream equality, countable-choice schedule, completion point, metric-completeness datum, or ambient Real completeness. □

Theorem 301.268 (RegSeqRat tail-shift regularity)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let $t_0, \ldots, t_k$ be a finite displayed family of tail shifts over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. Suppose every shifted finite window preserves the scheduled endpoint rows, dyadic-radius rows, regularity ledgers, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of P. Then every inspected tail window preserves the regular Cauchy inequality witnessed by the RegSeqRat regularity ledger before any downstream* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The regularity preservation is choice-free: it uses only the displayed finite windows and does not introduce a new selector, quotient stream equality, terminal Real seal, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 301.263 to the finite family of scheduled tail shifts. This builds one synchronized finite readback route whose rows are the shifted StreamName windows, DyadicRatCore tolerance budgets, regular-rational classifier rows, transport, replay, provenance, and local naming.

Next use Definition 301.262. The synchronized route keeps every inspected tail window on the same RegSeqRat readback row, so the regularity ledger is read from one finite source surface rather than from a selected limit.

Finally apply Definition 301.267. The choice-free finite-window theorem keeps the downstream Real-facing read inside the displayed StreamName schedule and DyadicRatCore tolerance budget. Therefore the regular Cauchy inequality carried by the RegSeqRat ledger is preserved on all inspected tail windows, with no new selector or terminal Real seal. □

Theorem 301.269 (RegSeqRat tail-shift budget stability)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and let $t_0, \ldots, t_k$ be a finite displayed family of admitted tail shifts over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule. Suppose the shifted windows preserve the synchronized StreamName finite section, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, the regularity ledger, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegSeqRat}^{\uparrow}}$ *rows of P. Then every admitted finite tail shift preserves the same DyadicRatCore tolerance ledger and the same RegSeqRat regularity ledger before any downstream* $\mathsf{Real}^{\uparrow}$ *seal is opened.*

The preservation exports no quotient stream equality, selected limit, countable-choice schedule, terminal seal, metric-completeness datum, or ambient Real completeness.

Proof. Apply Definition 301.263 to the finite family of admitted shifts. This produces one synchronized finite section whose visible rows are the StreamName windows, dyadic tolerance budget, regularity ledger, transport, replay, provenance, and local naming rows.

Use Definition 301.267 to keep the readback inside that finite section. The theorem rules out replacing the displayed window by a selector or by a terminal Real-facing object.

Finally apply Definition 291.104. The terminal-radius read is forced by the displayed DyadicRatCore denominator and half-step rows, so each admitted tail shift consumes the same tolerance ledger that the RegSeqRat regularity ledger already names. □

Theorem 301.270 (RegSeqRat tail-halving closure)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let n be a displayed unary scale whose dyadic tolerance window is valid for the tail budget carried by P. Then there is a displayed successor route at the halved scale through the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows: the next finite StreamName window is read first, the DyadicRatCore tolerance ledger refines the visible radius from scale n to its halved successor, the RegSeqRat classifier reads back the same regular-rational source surface, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal consumes only that displayed finite route.*

The route exports no quotient stream equality, selected limit, host real equality, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted tail-budget rows of P. The visible data are the $\mathsf{StreamName}^{\uparrow}$ finite schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance rows, the RegSeqRat regularity ledger, the classifier row, the displayed hsame and Cont supports, Pkg provenance, local NameCert naming, and the downstream $\mathsf{Real}^{\uparrow}$ seal-source row.

Use the StreamName finite schedule to choose the next visible window after the displayed scale n. This is a window read inside the packet, not a choice of an infinite subsequence or a selected limit.

Apply the DyadicRatCore tolerance rows on that next window. Their monotonicity and halving budget refine the radius ledger from the scale n window to the halved successor window while preserving the same endpoint and regularity rows available to the classifier.

Now repackage the successor window through Definition 301.257. The shifted finite window has the transported StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat regularity rows, classifier support, hsame, Cont, Pkg, and local NameCert rows required for the same public classifier read.

Finally read the $\mathsf{Real}^{\uparrow}$ seal through the existing regular-rational handoff. Since the successor route has only displayed StreamName, DyadicRatCore, RegSeqRat, and RealUp-facing rows, quotient stream equality, selected limits, host real equality, completion data, and ambient completeness have no carrier coordinate from which to enter. □

Theorem 301.271 (RegSeqRat tail-halving induction)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let*

$$n_0 \to n_1 \to \cdots \to n_k$$

be a finite displayed unary ledger whose adjacent arrows are accepted tail-halving steps for the tail budget carried by P. Then the whole ledger composes into one route through the same $\mathsf{StreamName}^{\uparrow}$ *finite schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, and downstream* $\mathsf{Real}^{\uparrow}$ *seal. Every intermediate window is read by shifting the finite schedule already carried by P, and the terminal route exports no quotient stream equality, selected limit, countable choice, metric-completeness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Induct on the displayed finite halving ledger.

In the base case the ledger has no successor step. The route is the original accepted $\mathsf{RegSeqRat}^{\uparrow}$ packet, whose StreamName schedule, DyadicRatCore tolerance ledger, classifier and readback rows, and RealUp seal are already displayed.

For the successor step, apply Definition 301.270 to the current accepted route. It supplies the next halved-scale window through the same StreamName, DyadicRatCore, RegSeqRat, and RealUp rows.

Use Definition 301.257 to shift the finite window before applying the induction hypothesis to the remaining ledger. The shifted route preserves the displayed schedule, tolerance, classifier, readback, hsame, Cont, Pkg, and local NameCert rows needed by the next step.

Composing the successor route with the induction hypothesis gives one finite route through the same packet rows. Since each adjacent step excludes quotient stream equality, selected limits, countable choice, metric completeness, and ambient RealUp completeness, the finite composite has no row from which any of those objects can enter. □

Theorem 301.272 (RegSeqRat terminal finite-window Real-seal)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let*

$$n_0 \to n_1 \to \cdots \to n_k$$

be a finite displayed unary halving ledger whose adjacent arrows are accepted tail-halving steps for the tail budget carried by P. Then the terminal halved window supplied by Definition 301.271 is the only finite input consumed by the downstream $\mathsf{Real}^{\uparrow}$ *seal: the route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows, and the selection-free Real-seal input is exactly that terminal displayed window.*

The route exports no quotient stream equality, selected limit, host real equality, countable choice, metric-completeness row, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 301.271 to the displayed ledger. That theorem already packages the whole finite route through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and downstream $\mathsf{Real}^{\uparrow}$ seal rows, with the terminal window determined by the last halving step.

Now read the nonescape wording of Definition 301.270. The seal boundary is only a row inventory: it keeps the displayed finite window and the listed support rows, and it removes any coordinate for quotient streams, selected limits, host real equality, countable choice, metric-completeness data, completion points, or ambient completeness.

Therefore the terminal halved window is exactly the selection-free Real-seal input. No other finite window can enter the seal, because the displayed route contains no additional row from which a different finite input could be selected. □

Theorem 301.273 (RegSeqRat tail-halving stability)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let P' be a finite tail-halving ledger over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *support. If P' is obtained from P by a displayed finite halving ledger, then replacing P by P' preserves the downstream* $\mathsf{Real}^{\uparrow}$ *seal handoff.*

The preserved handoff is still the finite terminal-window route and exports no quotient stream equality, selected limit, host real equality, countable choice, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 301.271 to the displayed finite halving ledger from P to P'. It composes the halving steps through the same StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat classifier and readback rows, and downstream RealUp seal.

Now apply Definition 301.272 to the terminal ledger. The theorem identifies the terminal halved window as the only finite input consumed by the downstream $\mathsf{Real}^{\uparrow}$ seal.

The support rows named in the statement are precisely the rows retained by those two theorems: hsame, Cont, Pkg, and local $\mathsf{NameCert}$ remain componentwise over the same public packet. Hence the replacement by P' preserves the seal handoff and cannot add quotient stream equality, selected limits, host real equality, choice, completion data, or ambient completeness. □

Theorem 301.274 (RegSeqRat tail-halving shift fusion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet. Suppose a displayed tail-shift row over the* $\mathsf{StreamName}^{\uparrow}$ *schedule and a displayed tail-halving row over the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger are both available for P. Then the two rows compose to a common finite tail window whose StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, and downstream* $\mathsf{Real}^{\uparrow}$ *seal are the same public read.*

The fused window is a finite L10 four-object route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It exports no quotient stream equality, selected limit, host real equality, metric-completeness datum, completion point, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the existing tail-shift route of Definition 301.257. The shifted window exposes only the transported StreamName schedule, DyadicRatCore endpoint and radius rows, regularity ledgers, classifier support, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Apply the tail-halving closure route of Definition 301.270 to that shifted window. The halved successor refines the displayed dyadic tolerance ledger while keeping the same finite StreamName source and the same regular-rational classifier surface.

Now take the common finite tail window given by the shifted window and its halved successor. On that window the StreamName schedule is already fixed by the tail-shift row, and the DyadicRatCore tolerance ledger is fixed by the halving row. The RegSeqRat classifier can therefore read only one finite source surface.

Finally read the downstream $\mathsf{Real}^{\uparrow}$ seal through the same four-object route. Since the carrier rows have been exhausted by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local naming, no quotient stream equality, selected limit, host real equality, metric-completeness datum, completion point, countable-choice schedule, or ambient completeness coordinate remains available. □

Theorem 301.275 (RegSeqRat tail-shift halving fusion is choice-free)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let a finite displayed family of tail-shift rows and tail-halving rows be indexed over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. Iterating Definition 301.274 over the family yields one finite common window with the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal.*

The finite-family route preserves the four-object path through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. It does not select a limiting tail, invoke quotient stream equality, use host real equality, ask for countable choice, expose a completion point, or assume ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct over the displayed finite family. In the empty case, the original accepted packet supplies the common window, readback, and Real seal through its StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat rows, transport, replay, provenance, and local naming data.

For the successor step, apply Definition 301.274 to the current common window and the next displayed pair of tail-shift and tail-halving rows. The theorem fuses that pair into a finite tail window whose StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal are the same public read.

Repack the successor window as the induction invariant. The same four-object route is preserved because each step reads only $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Thus the finite family has one common window without selecting a limiting tail. Since every step is finite and row-local, no quotient stream equality, host real equality, countable-choice schedule, completion point, or ambient Real completeness coordinate is introduced. □

Theorem 301.276 (RegSeqRat tail-halving classifier fusion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let a finite family of tail-shift and tail-halving rows share the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. Then the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier read is the same shared-window selector surface, and the family reaches one terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

The classifier route is the finite L10 route through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. It does not introduce quotient stream equality, a selected limit, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 301.275 to the displayed finite family. It gives one common finite window with the shared StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, and downstream Real seal.

Use the shared-window selector determinacy route for the same accepted packet. Since all rows are indexed over the same schedule and tolerance ledger, the classifier can read only the shared-window selector surface already exposed by the fused family.

Finally apply tail-shift regularity to keep the RegSeqRat row on the same regular-rational carrier. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached through that finite route, and no quotient stream equality, selected limit, metric-completeness datum, or ambient Real completeness coordinate is introduced. □

Theorem 301.277 (RegSeqRat tail-halving common refinement)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, and let two finite families of tail-shift and tail-halving rows be indexed over the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. Then the two families admit a common finite refinement whose* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier read and terminal* $\mathsf{Real}^{\uparrow}$ *seal agree.*

The refinement is the displayed BEDC route through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It does not introduce quotient streams, selected limits, metric completeness, countable choice, or ambient Real completeness.*

Proof. Apply Definition 301.275 separately to the two displayed finite families. Each family is reduced to one common finite window over the same StreamName schedule, dyadic tolerance ledger, RegSeqRat readback, and terminal Real seal route.

Compare the two resulting windows by the shared-window selector surface. Since both fused families are indexed over the same schedule and tolerance ledger, Definition 301.276 presents both classifier reads as the same shared-window selector surface.

Repack the comparison through the displayed hsame, Cont, Pkg, and local NameCert rows of the accepted packet. The terminal $\mathsf{Real}^{\uparrow}$ seal is read only after the StreamName, DyadicRatCore, and RegSeqRat rows agree on the common refinement, and the carrier has no quotient-stream, selected-limit, metric-completeness, countable-choice, or ambient-completeness coordinate that could supply a second route. □

Theorem 301.278 (RegSeqRat finite tail-window exhaustion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier row, let B be a finite displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row for P, and let S be a* $\mathsf{StreamName}^{\uparrow}$ *pointwise certificate for the same scheduled source. For every finite family of requested dyadic radii carried by B, there is a unary threshold t and a finite tail-window ledger such that every request beyond t is witnessed by a pair of* $\mathsf{StreamName}^{\uparrow}$ *observations and by* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-scale comparison rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or downstream* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The ledger is a finite BEDC route through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream equality, host real equality, selected limit, countable-choice schedule, metric-completeness datum, completion point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Induct over the displayed finite family of requested dyadic radii. In the empty case, the accepted packet already supplies the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ tail-budget row, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier surface, and the structural BHist, hsame, Cont, Pkg, and NameCert support.

For the successor step, assume the previous requests have been placed behind a unary threshold and a finite tail-window ledger. Use the pointwise $\mathsf{StreamName}^{\uparrow}$ certificate to expose the next pair of scheduled observations beyond that threshold.

Next refine the dyadic comparison. The terminal-scale rows of $\mathsf{DyadicRatCore}^{\uparrow}$ place the new radius and the accumulated budget on one common finite scale, so the next request is represented by displayed endpoint, radius, and comparison rows rather than by a hidden modulus or a selected subsequence.

Apply the tail-halving shift fusion route of Definition 301.274. It fuses the transported tail window and the dyadic refinement into a common finite window with the same $\mathsf{RegSeqRat}^{\uparrow}$ readback surface.

Repack the successor ledger by increasing the threshold to cover the new observation pair. The induction invariant is preserved because every step uses only finite StreamName observations, DyadicRatCore common-scale comparisons, the existing RegSeqRat classifier route, and displayed structural rows.

Thus the finite request family is exhausted by a tail-window ledger after a unary threshold. Since no step introduces quotient stream equality, host real equality, selected limits, countable choice, metric completeness, completion points, or ambient Real completeness, the downstream $\mathsf{Real}^{\uparrow}$ seal can read only the displayed finite route. □

Theorem 301.279 (RegSeqRat finite tail-window shift exhaustion)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier row, let B be a finite displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row for P, and let S be a* $\mathsf{StreamName}^{\uparrow}$ *pointwise certificate for the same scheduled source. If P is shifted by a displayed finite unary tail, then every retained request is still exhausted by a finite tail-window ledger whose* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows are the retained rows of the original ledger.*

The shift exports no quotient stream equality, host real equality, selected subsequence, countable-choice schedule, metric-completeness datum, completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the finite tail-window exhaustion theorem, Definition 301.278. It supplies a unary threshold and a finite ledger for every displayed family of dyadic radii, with StreamName observations and DyadicRatCore common-scale comparison rows visible before any RegSeqRat or Real handoff.

Shift the regular Cauchy name by the displayed finite unary tail. Retained requests are still beyond a displayed threshold, so the same finite StreamName windows remain the observation rows after dropping the shifted prefix.

Use the StreamName finite-section surface to keep the retained windows inside the same certified source, and use the DyadicRatCore tolerance handoff surface to keep the common-scale tolerance ledger attached to those windows. The RegSeqRat readback and Real seal handoff are therefore replayed through the same H, C, P, N support rows on every retained request.

Repack the shifted route as a finite tail-window ledger. Since the operation discards only the displayed finite prefix and preserves the retained request rows, it cannot add quotient stream equality, a selected subsequence, host equality, choice, metric completeness, completion points, or ambient Real completeness. □

Definition 301.280 (RegSeqRat tail-refinement carrier)**.** *A finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-refinement packet is a* BHist *row*

$$Q = (S, D, T, H, C, P, N)$$

whose displayed components are: a $\mathsf{StreamName}^{\uparrow}$ *finite request row S, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-window row T, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and a local* NameCert *row N. The classifier reads only these seven rows and exports no quotient stream equality, selected limit, host real equality, metric-completeness datum, completion point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 301.281 (RegSeqRat tail-refinement stability)**.** *Let $Q = (S, D, T, H, C, P, N)$ be an accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-refinement packet. Any retained tail request transported by componentwise* hsame *and replayed by the displayed* Cont *row is represented by the same* $\mathsf{StreamName}^{\uparrow}$

request row, $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-window row. Thus tail refinement is stable under the displayed transport and replay surface of* Q*.*

Proof. Project the carrier of Definition 301.280. The mathematical data rows are exactly S, D, T, while H records componentwise transport for those rows.

Replay the retained request through C. Since C is displayed as part of the same packet, it can only replay the finite request, tolerance, and tail-window rows already named by S, D, T.

The provenance and naming rows P, N do not add new tail data. Hence every transported retained request remains represented by the same finite tail-refinement surface. □

Theorem 301.282 (RegSeqRat tail-refinement Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-refinement packet*

$$Q = (S, D, T, H, C, P, N)$$

may read only the finite $\mathsf{RegSeqRat}^{\uparrow}$ *tail window* T *together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *and the supporting* $\mathsf{StreamName}^{\uparrow}$ *request row* S*. The handoff deliberately exposes finite tail-window and dyadic tolerance data, not quotient stream equality, a selected limit, host real equality, or a completion point.*

Proof. Start with a Real-facing read. The request row S displays the finite StreamName window being inspected, and the tolerance row D gives the dyadic scale at which the retained tail request is compared.

Read the tail-window row T. It is the only RegSeqRat row exported by the packet, so the Real-facing consumer receives a finite tail-window route rather than a quotient stream object or selected limit.

Finally transport and replay through H and C, with P, N recording provenance and local naming. These rows preserve the finite handoff and do not add host real equality, completion points, or ambient Real completeness. □

Theorem 301.283 (RegSeqRat tail-refinement composition)**.** *Let*

$$Q_0 = (S_0, D_0, T_0, H_0, C_0, P_0, N_0)$$

and

$$Q_1 = (S_1, D_1, T_1, H_1, C_1, P_1, N_1)$$

be adjacent accepted finite $\mathsf{RegSeqRat}^{\uparrow}$ *tail-refinement packets. If the retained* $\mathsf{StreamName}^{\uparrow}$ *request row of* Q_0 *is the source request row of* Q_1 *by componentwise* hsame *transport, and the terminal* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of* Q_0 *is the tolerance row read by* Q_1*, then the two packets compose into one accepted finite tail-refinement packet preserving the* T, H, C, P, N *rows and the Real handoff surface.*

The composite packet exposes only finite $\mathsf{StreamName}^{\uparrow}$ *request data, displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, finite* $\mathsf{RegSeqRat}^{\uparrow}$ *tail windows, transport, replay, provenance, and local naming; it adds no selected subsequence, quotient stream equality, selected limit, host real equality, completion point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the two carriers using Definition 301.280. Their public rows are exactly the finite request rows S_i, tolerance rows D_i, tail-window rows T_i, transport rows H_i, replay rows C_i, provenance rows P_i, and local naming rows N_i.

Identify the shared retained request. The hypothesis says that the retained $\mathsf{StreamName}^{\uparrow}$ request from Q_0 is transported by hsame to the source request of Q_1, so the two packets meet on a displayed finite request row rather than on a quotient stream.

Compose the displayed Cont replay. First replay the retained request through C_0, then replay the second tail refinement through C_1; the shared request row and shared tolerance row keep the composite inside the same finite DyadicRatCore and RegSeqRat support.

Package the result as the tail window obtained by following T_0 into T_1, with the inherited transport, replay, provenance, and local naming rows. Stability, Definition 301.281, shows that every retained request remains represented by those displayed rows.

Finally apply the Real handoff theorem, Definition 301.282, to the composite packet. A Real-facing consumer therefore reads only the composed finite tail window, the shared dyadic tolerance route, and the supporting StreamName request row, with no selected subsequence, quotient stream equality, selected limit, host equality, completion point, or ambient Real completeness. □

Chapter 302

A Concrete Naming Certificate for $\mathsf{IntervalArithmetic}^{\uparrow}$

The $\mathsf{IntervalArithmetic}^{\uparrow}$ interface names certified interval arithmetic over the real-name layer already present in $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Interval}^{\uparrow}$. In the constructive numerical-analysis sense represented by Moore, Kearfott, and Cloud's interval-analysis presentation, an interval computation is acceptable when the endpoint and enclosure ledgers are finite and inspectable. The BEDC surface therefore treats every operation as a row over dyadic endpoint histories and regular-sequence observations, not as a host interval object or a quotient-real interval.

The endpoint and enclosure rows consume chapter 291, chapter 126, chapter 301, and chapter 38; those chapters supply, respectively, dyadic endpoint histories, finite observation windows, regular-sequence readback, and the real-name seal used by the interval packet.

302.1 Endpoint carrier

Definition 302.1 (IntervalArithmetic BHist carrier)**.** *An* $\mathsf{IntervalArithmetic}^{\uparrow}$ *endpoint packet is a finite* BHist *row*

$$(\ell, u, \lambda, \upsilon, \epsilon, P)$$

with the following displayed coordinates:

1. ℓ *is a lower dyadic endpoint history certified by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. u *is an upper dyadic endpoint history certified by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. λ *and* υ *are the lower and upper observation rows read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*;*
4. ϵ *is the enclosure predicate saying that the represented* $\mathsf{Real}^{\uparrow}$ *observation lies between the two endpoint histories at every displayed finite observation window;*
5. P *is the* Pkg *row naming the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Interval}^{\uparrow}$ *dependency certificates.*

The carrier predicate is $\mathsf{IntervalArithmeticPacket}(\ell, u, \lambda, \upsilon, \epsilon, P)$*. Its classifier compares two accepted packets by endpoint* hsame *for* ℓ *and* u*, plus the finite-window* Cont *rows that transport* λ*,* υ*, and* ϵ *across the named dependency certificates.*

302.2 Operation ledgers

Theorem 302.2 (Outward-rounded addition ledger)**.** *Let $A = (\ell_A, u_A, \lambda_A, \upsilon_A, \epsilon_A, P_A)$ and $B = (\ell_B, u_B, \lambda_B, \upsilon_B, \epsilon_B, P_B)$ be accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packets. An outward-rounded addition ledger consists of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *addition rows*

$$\ell_C \leq_{\mathsf{dyad}} \ell_A + \ell_B, \qquad u_A + u_B \leq_{\mathsf{dyad}} u_C,$$

the transported $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *observation rows for the two endpoints, and a* Pkg *row tying these data to P_A and P_B. The result packet $C = (\ell_C, u_C, \lambda_C, \upsilon_C, \epsilon_C, P_C)$ is an accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packet whose enclosure predicate contains every finite* $\mathsf{Real}^{\uparrow}$ *observation obtained by adding an observation enclosed by A to an observation enclosed by B.*

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticPacket outward rounded addition`
↪ variant `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket outward rounded addition`

Proof. Project the two endpoint packets. Their carrier rows expose only dyadic endpoint histories, observation histories, enclosure predicates, and package provenance.

Use the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ addition rows to form the candidate lower and upper endpoints. The outward inequalities are stored as rows of the ledger, so the result endpoint pair is not read from host rounding behavior.

Transport the lower and upper observation rows through the finite-window Cont data supplied by the two packages. At each finite window, the lower outward row lies below the sum of lower observations and the upper outward row lies above the sum of upper observations.

Repack the transported observation rows and the endpoint inequalities as ϵ_C. The resulting packet is exhausted by the displayed dyadic, stream, regular-sequence, real-name, and package rows, so the addition ledger exports no host real interval. □

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket outward addition closure`

Theorem 302.3 (IntervalArithmetic classifier transport)**.** *Suppose A and A' are accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packets whose lower and upper endpoint histories are related by endpoint* hsame*, and whose observation and enclosure rows are related by the displayed finite-window* Cont *transports. Then every outward-rounded operation ledger accepted for A transports to an accepted ledger for A', and the result packets are classified by the* $\mathsf{IntervalArithmetic}^{\uparrow}$ *classifier.*

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket transport`

Proof. Unpack the classifier. It gives endpoint hsame rows for the lower and upper dyadic histories and continuation rows for the observation and enclosure data.

Apply these rows to each coordinate of the operation ledger. The lower and upper dyadic bounds transport by endpoint hsame, and the observation windows transport by the corresponding Cont rows.

Finally repack the transported coordinates as the result ledger. Since the classifier for the result packet is built from the same endpoint and finite-window transports, the transported result is classified with the original result packet. □

Theorem 302.4 (RealUp enclosure soundness)**.** *For every accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packet $A = (\ell, u, \lambda, \upsilon, \epsilon, P)$, any* $\mathsf{Real}^{\uparrow}$ *observation row admitted by ϵ is enclosed by the displayed lower and upper endpoint histories at each finite observation window named by P. In particular, the soundness*

claim is a finite ledger claim over $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *observations, not a completeness or compactness claim about external real intervals.*

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket realup enclosure soundness`

Proof. Read the enclosure coordinate ϵ from the carrier. It is already indexed by the finite observation windows named in the package row.

For a chosen window, project the lower endpoint observation, the upper endpoint observation, and the admitted $\mathsf{Real}^{\uparrow}$ observation. The definition of ϵ supplies the two displayed order comparisons between them.

Because the projection uses only rows present in P, the conclusion is stable under the same finite-window transport used by the classifier. No open-cover compactness or quotient-real interval is invoked. □

302.3 Public surface

Theorem 302.5 (IntervalArithmetic public certificate surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{IntervalArithmetic}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *the endpoint carrier of Definition 302.1;*
2. *the endpoint and finite-window classifier over* hsame *and* Cont*;*
3. *the stability transport row of Definition 302.3;*
4. *the outward-rounded operation ledger of Definition 302.2;*
5. *the public enclosure surface of Definition 302.4.*

These rows are the complete public surface presently claimed for certified interval arithmetic over the existing BEDC real-name infrastructure.

Proof. The carrier row supplies the lower endpoint, upper endpoint, observation, enclosure, and package coordinates. The classifier row is the restricted endpoint hsame plus finite-window Cont transport on those coordinates.

The stability row says that accepted ledgers survive classifier transport. The operation row supplies outward-rounded addition as a finite endpoint and enclosure ledger.

The public enclosure row is exactly the real-name readback permitted to downstream consumers. Since the five rows list every exported coordinate, the certificate surface has no coordinate for completeness, Heine–Borel, external real intervals, or bridge data. □

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket public certificate surface`

Theorem 302.6 (IntervalArithmetic monotone enclosure transport)**.** *Let* $A = (\ell, u, \lambda, \upsilon, \epsilon, P)$ *and* $A' = (\ell', u', \lambda', \upsilon', \epsilon', P')$ *be accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packets. Suppose the lower endpoint of* A' *tightens the lower endpoint of* A*, the upper endpoint of* A' *tightens the upper endpoint of* A*, and the endpoint rows are related by the displayed* hsame *and* Cont *transports of the classifier. Then every finite* $\mathsf{Real}^{\uparrow}$ *observation row enclosed by* A' *transports to an observation row enclosed by* A*, and the transported enclosure predicate is read from the same endpoint carrier and package rows.*

Proof. Project the two packets through Definition 302.1. The only rows available for the enclosure comparison are the lower endpoint, upper endpoint, observation, enclosure, and Pkg coordinates.

Use Definition 302.3 to move the endpoint and observation rows along the displayed hsame and Cont data. Tightening the lower row and tightening the upper row preserve the order comparisons used by the enclosure predicate at each named finite window.

Finally apply Definition 302.4. The admitted $\mathsf{Real}^{\uparrow}$ observation is read only through the transported finite-window endpoint rows, so the enclosure predicate for A' maps into the enclosure predicate for A without adding an external interval object. □

Theorem 302.7 (IntervalArithmetic width-bound stability)**.** *Let an accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packet carry a displayed dyadic width bound between its lower and upper endpoint histories. If its endpoint packet is transported by the classifier and then used in an outward-rounded addition ledger, the resulting packet carries the transported width bound obtained from the same endpoint rows and outward inequalities.*

The width row is therefore stable under endpoint classifier transport and outward rounding: it remains a finite dyadic endpoint ledger row, not a host metric estimate over external intervals.

Lean4 ✓ `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket width bound stability`

Proof. Read the width row from the endpoint carrier of Definition 302.1. It is a finite dyadic comparison between the displayed lower and upper endpoint histories.

Apply Definition 302.3. The lower and upper endpoints, the finite observation rows, and the enclosure predicate move componentwise along the displayed hsame and Cont rows, so the same dyadic width comparison is read after transport.

Apply Definition 302.2. The outward inequalities used by the addition ledger are stored as endpoint rows, and therefore compose with the transported width comparison inside the dyadic endpoint ledger.

The repacked result has only the endpoint, observation, enclosure, and package rows named by the carrier. Hence the width bound is stable as a finite certificate row and does not depend on an external interval metric. □

Theorem 302.8 (IntervalArithmetic refined-endpoint addition soundness)**.** *Let A, B, and C be accepted* $\mathsf{IntervalArithmetic}^{\uparrow}$ *packets such that C is produced from A and B by the outward-rounded addition ledger of Definition 302.2. If either input is replaced by a refined endpoint packet classified with it by the displayed endpoint* hsame *and finite-window* Cont *rows, then outward-rounded addition remains sound for the transported result packet.*

In particular, every finite $\mathsf{Real}^{\uparrow}$ *observation obtained by adding observations enclosed by the refined inputs is enclosed by the transported outward-rounded result packet.*

Proof. Start with the original outward-rounded ledger. By Definition 302.2, its lower and upper result endpoints are justified by displayed dyadic endpoint inequalities, and its enclosure predicate is built from the finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ observation rows.

Transport the replaced input through Definition 302.3. The endpoint hsame rows and finite-window Cont rows move the lower endpoint, upper endpoint, observation, and enclosure coordinates to the refined packet while preserving the accepted ledger shape.

Now reuse the outward-rounded addition ledger with the transported endpoint rows. The lower and upper inequalities remain valid in the displayed dyadic ledger, and the observation rows remain finite rows over the same named dependency certificates.

Finally apply Definition 302.4. The transported result packet encloses the finite $\mathsf{Real}^{\uparrow}$ observations obtained from the refined inputs, because the proof reads only the displayed endpoint, classifier, operation, and enclosure rows. □

Theorem 302.9 (IntervalArithmetic obligation enclosure package)**.** *The obligation package for* $\mathsf{IntervalArithmetic}^{\uparrow}$ *is the finite endpoint carrier, endpoint classifier, outward-rounded addition ledger, classifier transport,* $\mathsf{Real}^{\uparrow}$ *enclosure soundness row, monotone enclosure transport, width-bound stability row, and refined-endpoint addition soundness row.*

All coordinates are displayed over $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Interval}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*; no host interval carrier, external real equality, compactness principle, or completion object is part of the package.*

Proof. Start from Definition 302.1. It supplies the finite endpoint carrier, the endpoint classifier coordinates, and the displayed observation rows over the dyadic, stream, regular-sequence, real, and interval certificates.

Apply Definition 302.3 to account for the classifier transport rows, and apply Definition 302.2 to add the operation ledger. The latter contributes only outward-rounded dyadic endpoint evidence and finite observation provenance.

The enclosure obligations are then supplied by Definition 302.4 and Definition 302.6. These rows keep the $\mathsf{Real}^{\uparrow}$ observations and monotone transport inside the displayed endpoint and finite-window surface.

Finally use Definition 302.7 and Definition 302.8. They give the width-bound stability and refined-endpoint addition soundness rows, so the listed data form the full finite obligation package and introduce no additional host-level interval or real object. □

Theorem 302.10 (IntervalArithmetic standard bridge)**.** *The public endpoint packet of* $\mathsf{IntervalArithmetic}^{\uparrow}$ *bridges to the standard outward-rounded interval-arithmetic reading exactly through the rational endpoint rows,* $\mathsf{Real}^{\uparrow}$ *enclosure soundness row, monotone transport row, width-bound row, refined endpoint addition row, visible* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

The bridge withholds multiplication, division, dependency-aware narrowing, compactness, Heine–Borel, and completed-real equality. It is only the finite outward-rounded endpoint and enclosure surface displayed by the public packet.

Proof. Start with the endpoint packet and public certificate surface. The carrier in Definition 302.1 supplies the lower and upper dyadic endpoint histories, finite observation rows, enclosure predicate, and package provenance, while Definition 302.5 repacks these rows as the local $\mathsf{NameCert}_{\mathsf{IntervalArithmetic}^{\uparrow}}$ surface.

Add the outward-rounded operation row. Apply Definition 302.2 for the displayed endpoint addition ledger, and apply Definition 302.3 to keep that ledger stable under visible endpoint hsame and finite-window Cont transport.

The enclosure side is supplied by Definition 302.4 and Definition 302.6. These two rows keep every $\mathsf{Real}^{\uparrow}$ observation inside the displayed endpoint and monotone transport surface.

Finally use Definition 302.7, Definition 302.8, and Definition 302.9. Together they identify the bridge as the finite outward-rounded endpoint package with its width and refined-addition rows. None of the cited rows contains multiplication, division, dependency-aware narrowing, compactness, Heine–Borel, completed-real equality, or host interval semantics. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalArithmetic}^{\uparrow}}$**.**

Constructive story: Certified interval arithmetic is represented by finite lower and upper dyadic endpoint histories, finite stream and regular-sequence observation rows, an enclosure predicate over $\mathsf{Real}^{\uparrow}$ observations, outward-rounded operation ledgers, and endpoint-plus-window classifier transport.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntervalArithmetic}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 302.10 binds the endpoint carrier, endpoint classifier, outward-rounded addition ledger, classifier transport, $\mathsf{Real}^{\uparrow}$ enclosure soundness, monotone enclosure transport, width-bound stability, refined-endpoint addition soundness, and obligation enclosure package over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Interval}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntervalArithmetic}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.IntervalArithmeticUp.IntervalArithmeticEndpointPacket\_public\_certificate\_surface`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close multiplication, division, dependency-aware narrowing, completeness, compactness, Heine–Borel, or completed-real equality.

Upgrade path: No further paper-side theory grade remains for the displayed outward-rounded endpoint and $\mathsf{Real}^{\uparrow}$ enclosure consumer package; formal bridge verification remains separate.

Chapter 303

A Concrete Naming Certificate for $\mathsf{ModularArithmetic}^{\uparrow}$

The $\mathsf{ModularArithmetic}^{\uparrow}$ interface names residue arithmetic over certified integer histories. In the standard sense of modular arithmetic developed in Ireland–Rosen and Hardy–Wright, a modulus determines residue classes and addition and multiplication are read by reducing integer operations modulo that modulus. The BEDC surface keeps the modulus and every residue representative as finite rows over $\mathsf{Int}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$; it does not introduce a quotient primitive.

The residue carrier consumes the signed-unary representative and modulus rows supplied by chapter 31. It then routes the $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, and $\mathsf{Order}^{\uparrow}$ reads over those displayed $\mathsf{Int}^{\uparrow}$ rows before handing the reduced residue boundary to $\mathsf{Rat}^{\uparrow}$ consumers.

303.1 Residue source

Definition 303.1 (ModularArithmetic residue source)**.** *A* $\mathsf{ModularArithmetic}^{\uparrow}$ *residue source is a finite* BHist *packet*

$$(m, u, r, \rho, P)$$

whose rows have the following shape:

1. *m is an accepted* $\mathsf{Int}^{\uparrow}$ *modulus row;*
2. *u is a positive* UnaryHistory *witness naming the nonzero modulus boundary;*
3. *r is an accepted* $\mathsf{Int}^{\uparrow}$ *representative row;*
4. *ρ is a finite residue ledger recording that r is classified by the remainder window selected by m and u;*
5. *P is the* Pkg *provenance row naming the* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Mul}^{\uparrow}$*,* $\mathsf{Order}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *dependency certificates used by the packet.*

The carrier is the predicate $\mathsf{ModResidue}(m, u, r, \rho, P)$*. Its classifier compares two accepted residue packets exactly when their modulus rows and unary modulus witnesses transport by* hsame *and their representatives reduce to the same displayed remainder window through the carried ρ ledgers.*

Theorem 303.2 (ModularArithmetic residue carrier stability)**.** *Let $D = (m, u, r, \rho, P)$ be an accepted* $\mathsf{ModularArithmetic}^{\uparrow}$ *residue source and let $D' = (m', u', r', \rho', P')$ be obtained from D by displayed* hsame *transport of the modulus, unary witness, representative, residue ledger, and*

provenance rows. Then D' is again an accepted residue source, and its classifier reads the same remainder window through the transported $\mathsf{Int}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, *and* Pkg *rows.*

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource carrier stability`
Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueCarrier stability`

Proof. Project the residue source of Definition 303.1. Each coordinate is a displayed row rather than a quotient class, so an hsame transport of the coordinate gives the corresponding coordinate in D'.

The positivity and nonzero-modulus information is carried by the unary witness u. Transporting this witness gives u', and the remainder window selected by the transported modulus is read by the same finite order ledger.

Finally transport the residue ledger ρ and the package row P. Repacking the transported modulus, unary witness, representative, ledger, and provenance gives $\mathsf{ModResidue}(m', u', r', \rho', P')$, and the classifier has exactly the transported remainder-window rows available. □

Theorem 303.3 (ModularArithmetic addition and multiplication ledger)**.** *For accepted residue sources over the same transported modulus witness, the* $\mathsf{ModularArithmetic}^{\uparrow}$ *addition and multiplication reads consist of the following finite rows: the two* $\mathsf{Int}^{\uparrow}$ *representative rows, the shared modulus and unary witness, the* $\mathsf{Add}^{\uparrow}$ *sum row or* $\mathsf{Mul}^{\uparrow}$ *product row on representatives, the* $\mathsf{Order}^{\uparrow}$ *reduction window selecting the remainder, the resulting* $\mathsf{Int}^{\uparrow}$ *residue representative, and the* Pkg *provenance tying these rows to the same dependency certificates.*

The ledger exports no host quotient ring, no set of residue classes, and no equality outside the displayed hsame *and classifier rows.*

Proof. Start with two accepted sources and project their common modulus data. The carrier gives the two representative rows and the same transported nonzero-modulus witness.

For addition, read the displayed $\mathsf{Add}^{\uparrow}$ row on the two representatives. The order and modulus rows then select the finite remainder window, which is repacked as a new residue source by Definition 303.1.

For multiplication, use the same construction with the displayed $\mathsf{Mul}^{\uparrow}$ product row. In both cases the resulting packet is exhausted by the representative operation row, the reduction ledger, and the shared provenance, so no quotient-ring object is introduced. □

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource add mul ledger`

Theorem 303.4 (ModularArithmetic residue operation congruence)**.** *Let A, A', B, B' be accepted* $\mathsf{ModularArithmetic}^{\uparrow}$ *residue sources in the sense of Definition 303.1, all over the same transported modulus witness. Suppose that A classifies with A' and B classifies with B' by the residue classifier, using only the displayed* $\mathsf{Int}^{\uparrow}$ *representative rows,* $\mathsf{Order}^{\uparrow}$ *remainder windows,* BHist *packets,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *dependency rows. Then the* $\mathsf{Add}^{\uparrow}$ *ledgers for $A + B$ and $A' + B'$ classify the two reduced sum packets, and the* $\mathsf{Mul}^{\uparrow}$ *ledgers for $A \cdot B$ and $A' \cdot B'$ classify the two reduced product packets. The resulting packets expose the same* $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ *handoff boundary and no quotient-ring equality beyond the displayed residue classifier.*

Proof. Project the four residue sources through Definition 303.1. The hypotheses identify the two A representatives through one transported remainder window and the two B representatives through another, while the modulus and unary modulus witness are shared by the same hsame and Cont routes.

For addition, read the displayed $\mathsf{Add}^{\uparrow}$ rows on the two representative pairs. Because the input representatives classify through the transported remainder windows, the resulting sum representatives are reduced through the same $\mathsf{Order}^{\uparrow}$ modulus window and give classified reduced sum packets.

For multiplication, repeat the same construction with the displayed $\mathsf{Mul}^{\uparrow}$ rows. The product representatives pass through the same modulus witness and the same finite reduction window, so the two reduced product packets are classifier-related as residue sources.

Finally repack both operation outputs using the carrier of Definition 303.1. The BHist rows, Pkg provenance, NameCert dependencies, and $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ handoff rows are exactly the rows already carried by the projected packets and operation ledgers, so the congruence is a statement about the displayed classifier only. □

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource operation congruence`

Theorem 303.5 (ModularArithmetic Int/Rat handoff)**.** *Every accepted* $\mathsf{ModularArithmetic}^{\uparrow}$ *residue packet exposes its representative and modulus through* $\mathsf{Int}^{\uparrow}$ *rows, and every rational normalization or denominator-side consumer may read the packet only through the corresponding* $\mathsf{Rat}^{\uparrow}$ *handoff rows, the modulus unary witness, the residue ledger,* hsame *transport,* Cont *routes, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource int rat handoff`

Proof. The source definition stores the representative and modulus as $\mathsf{Int}^{\uparrow}$ rows. Projecting the packet therefore gives the integer data needed by the residue classifier without invoking any external arithmetic carrier.

A $\mathsf{Rat}^{\uparrow}$ consumer can only receive the integer-to-rational handoff rows named by the same provenance package. The modulus witness and residue ledger remain visible, so the consumer sees which finite reduction window produced the representative.

Because the packet has no coordinate for a rational quotient of residue classes, the handoff is exactly the displayed $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ boundary with its hsame, Cont, and Pkg rows. □

Theorem 303.6 (ModularArithmetic NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}$ *obligation surface consists of five rows: the residue source carrier of Definition 303.1; the remainder-window classifier over transported modulus witnesses; the addition ledger of Definition 303.3; the multiplication ledger of the same theorem; and the* $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ *handoff boundary of Definition 303.5. These rows are the full public surface for residue arithmetic over certified integer histories.*

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource namecert obligation surface`

Proof. The carrier and classifier are the two projections of Definition 303.1. They provide the accepted residue packets and the visible equality test by common remainder windows.

The operation rows are supplied by Definition 303.3. Addition and multiplication both use the same modulus witness, representative rows, finite reduction window, and package provenance.

The handoff row is supplied by Definition 303.5. Together these projections list every coordinate readable from the packet, so the $\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}$ surface is exhausted by the five displayed rows and no quotient primitive is exported. □

Theorem 303.7 (ModularArithmetic scoped dependency boundary)**.** *The five local* $\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}$ *rows are scoped exactly by the displayed* $\mathsf{Int}^{\uparrow}$ *modulus and representative rows, the* $\mathsf{Add}^{\uparrow}$ *and* $\mathsf{Mul}^{\uparrow}$

operation rows, the $\mathsf{Order}^{\uparrow}$ *reduction-window row, the* $\mathsf{Rat}^{\uparrow}$ *handoff row, the positive* UnaryHistory *modulus witness, the* BHist *residue packet,* hsame *transports,* Cont *routes,* Pkg *provenance, and the dependency certificates named by* NameCert. *No quotient ring, host congruence class, finite-field structure, or external equality is exported from this scoped boundary.*

Proof. Project Definition 303.1. Its carrier and classifier coordinates give the representative row, modulus row, positive unary witness, residue packet, transported remainder window, continuation routes, package provenance, and dependency certificates.

Use Definition 303.3 for the operation surface. The addition and multiplication rows consume only the displayed $\mathsf{Add}^{\uparrow}$ and $\mathsf{Mul}^{\uparrow}$ dependencies before repacking the reduced residue through the same modulus witness and $\mathsf{Order}^{\uparrow}$ reduction window.

Use Definition 303.5 for the consumer boundary. The handoff exposes exactly the integer representative and modulus rows to $\mathsf{Rat}^{\uparrow}$ consumers, together with the same hsame, Cont, and Pkg ledgers.

These three projections exhaust the concrete BEDC rows visible in the packet. Since none of the named rows contains a coordinate for quotient-ring equality, host congruence classes, finite-field structure, or external equality, the scoped surface exports no such object. □

Lean4 ✓ `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource scoped dependency boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}$.

Constructive story: Residue arithmetic is generated by choosing a certified integer modulus, a positive unary modulus witness, an integer representative, and a finite reduction ledger; addition and multiplication read the existing $\mathsf{Add}^{\uparrow}$ and $\mathsf{Mul}^{\uparrow}$ rows before reducing through the displayed $\mathsf{Order}^{\uparrow}$ window.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature finite residue package binds the residue carrier and classifier, addition and multiplication ledgers, $\mathsf{Int}^{\uparrow}/\mathsf{Rat}^{\uparrow}$ handoff rows, scoped dependency boundary, and downstream $\mathsf{Divisibility}^{\uparrow}$ consumer surface over certified integer histories.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModularArithmetic}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ModularArithmeticUp.ModularArithmeticResidueSource\_scoped\_dependency\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient rings, Chinese-remainder structure, primality-specific arithmetic, finite-field laws, or a standard bridge.

Upgrade path: No further paper-side theory grade remains for the finite residue arithmetic package; bridgeChecked verification remains on the formal axis.

Chapter 304

A Concrete Naming Certificate for $\mathsf{ApartnessReal}^{\uparrow}$

The $\mathsf{ApartnessReal}^{\uparrow}$ interface names constructive apartness evidence between displayed real-name presentations. Following the Bishop–Bridges 1985 constructive-analysis treatment, it records positive finite separation witnesses in place of a trichotomy principle. It is a companion to $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, not a quotient construction and not a completeness theorem.

Its certificate consumes the sealed real-name surface of chapter 38, the regular Cauchy source route of chapter 301, the dyadic positive-radius rows of chapter 291, and the finite observation windows of chapter 126. These four sibling chapters supply, respectively, the endpoint names, source classifier, positive radius, and finite window rows used by the apartness packet, so the positive-separation certificate is routed through the real-completion lattice rather than through an isolated kernel packet.

304.1 Carrier and classifier

Definition 304.1 (ApartnessReal carrier)**.** *An* $\mathsf{ApartnessReal}^{\uparrow}$ *carrier is a finite* BHist *packet containing two displayed* $\mathsf{Real}^{\uparrow}$ *names, or equivalently two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source packets at the real-seal boundary, together with the finite rows needed to witness positive separation. The packet carries:*

1. *the left and right real-name rows, with their visible* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *provenance;*
2. *a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* ρ*;*
3. *a finite* $\mathsf{StreamName}^{\uparrow}$ *observation window shared by the two presentations;*
4. *two endpoint readbacks* a *and* b *from that window;*
5. *a separation ledger displaying that the readbacks are separated by the radius* ρ*.*

The carrier accepts only the displayed positive-separation packet. It does not decide whether arbitrary real names are equal or apart, does not assert trichotomy, and does not add quotient classes of real-name presentations.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealTasteGate single carrier alignment`

Definition 304.2 (ApartnessReal classifier)**.** *For accepted* $\mathsf{ApartnessReal}^{\uparrow}$ *packets* P *and* Q*, the classifier* $\mathsf{ApartnessRealClassifier}(P, Q)$ *holds when every visible component of the separation packet*

transports componentwise through the available classifiers. The two real names transport by the existing $\mathsf{Real}^{\uparrow}$ *classifier, or by the* $\mathsf{RegSeqRat}^{\uparrow}$ *source classifier before the real-seal handoff; the radius row transports by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *classifier; the finite observation window transports by the* $\mathsf{StreamName}^{\uparrow}$ *schedule classifier; the two endpoint readbacks transport through the same* hsame *and* Cont *routes; and the separation ledger is repacked after those componentwise transports.*

Thus the classifier identifies two presentations only as presentations of the same finite apartness evidence. It supplies no host real inequality, no quotient equality of Cauchy streams, and no law deciding equality versus apartness for arbitrary real names.

304.2 Certificate obligations

Theorem 304.3 (ApartnessReal obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApartnessReal}^{\uparrow}}$ *obligation surface has five finite rows:*

1. *carrier acceptance for the packet of Definition 304.1;*
2. *classifier stability for the componentwise transport relation of Definition 304.2;*
3. *symmetry, obtained by swapping the two endpoint readbacks and reversing the displayed separation ledger;*
4. *exactness, read as the finite-window separation witness exposed by the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;*
5. *ledger visibility, asserting that the positive radius, window, endpoint readbacks, and separation ledger remain visible through the displayed* Cont *and* Pkg *rows.*

The surface supplies apartness evidence only. It does not supply trichotomy of constructive reals, completeness of real names, a metric-completion object, or a quotient carrier.

```
Lean4 ✓  BEDC.Derived.ApartnessRealUp.ApartnessRealNameCertObligationSurface rows
Lean4 ✓  BEDC.Derived.ApartnessRealUp.ApartnessRealCarrier namecert obligation surface
```

Proof. Project the carrier packet. Its rows are exactly the two real-name presentations, the positive dyadic radius, the finite stream window, the two endpoint readbacks, and the displayed separation ledger.

Project the classifier next. Componentwise transport of the two real-name rows, radius row, stream window, endpoint rows, and ledger is precisely the stability face of the certificate.

For symmetry, swap the left and right endpoint readbacks while keeping the same positive radius and observation window. The separation ledger is read in the reverse direction, and the componentwise hsame and Cont routes repack the swapped packet.

Exactness and visibility are finite-readback claims: every apartness witness consumed downstream is one of the displayed finite rows, and every such row remains inspectable through its Cont and Pkg provenance. Since the packet has no coordinate for trichotomy, completeness, or quotient classes, those principles cannot be exported. □

```
Lean4 ✓  BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket namecert obligation surface
```

Theorem 304.4 (ApartnessReal symmetry stability)**.** *If an accepted* $\mathsf{ApartnessReal}^{\uparrow}$ *packet witnesses positive separation from a displayed real name* x *to a displayed real name* y*, then swapping the two real-name rows and the two endpoint readbacks produces an accepted* $\mathsf{ApartnessReal}^{\uparrow}$ *packet witnessing positive separation from* y *to* x*. The swapped packet preserves the same positive*

DyadicRatCore$^{\uparrow}$ *radius row, the same finite* StreamName$^{\uparrow}$ *observation window, and the same* hsame, Cont, Pkg, *and* NameCert *provenance.*

This symmetry is stability of the displayed apartness evidence. It is not a decision procedure for whether x and y are equal or apart.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealCarrier symmetry stability`

Proof. Start from the accepted carrier of Definition 304.1. It contains the two real-name rows, the shared finite observation window, the positive radius, the two endpoint readbacks, and the separation ledger.

Exchange the left and right real-name rows and exchange the two endpoint readbacks. The finite observation window and the positive radius row are unchanged.

The ledger is repacked by reading the separation row in the opposite endpoint order and carrying along the same hsame, Cont, Pkg, and NameCert provenance. The classifier of Definition 304.2 accepts this componentwise transport, so the swapped packet is again an accepted apartness witness. No equality-or-apartness case split is introduced. □

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket symmetry stability`

Theorem 304.5 (ApartnessReal metric handoff)**.** *An accepted* ApartnessReal$^{\uparrow}$ *packet supplies exactly the finite positive-separation row needed by a later* Met$^{\uparrow}$ *or real-order consumer: two displayed real-name endpoints, a positive* DyadicRatCore$^{\uparrow}$ *radius, a finite* StreamName$^{\uparrow}$ *observation window, endpoint readbacks, and a separation ledger visible through* Cont *and* Pkg*.*

The handoff permits a downstream metric or order construction to consume apartness evidence as data. It does not provide real trichotomy, total order, completeness, quotient classes, or a standard bridge for constructive reals.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket metric handoff`
Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealMetricHandoffPacket transport`

Proof. Read the packet through Definition 304.3. The carrier and exactness rows expose the two endpoints, positive radius, finite observation window, endpoint readbacks, and separation ledger.

A metric or order consumer may use only those exposed rows. The classifier row transports the same evidence across accepted presentations, and Definition 304.4 supplies the reversible endpoint orientation expected of apartness.

Because the exported surface is a finite separation witness, it gives no way to select a separation witness for arbitrary distinct names and no way to decide equality against apartness. The handoff is therefore constructive apartness data only. □

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealPositiveSeparationCarrier metric handoff`

Definition 304.6 (ApartnessReal finite-window separation row)**.** *An* ApartnessReal$^{\uparrow}$ *finite-window separation row is the displayed* BHist *tuple consisting of:*

1. *a left endpoint row and a right endpoint row, each read either as a displayed* Real$^{\uparrow}$ *name or as its* RegSeqRat$^{\uparrow}$ *source row at the real-seal boundary;*
2. *one positive* DyadicRatCore$^{\uparrow}$ *radius row ρ;*
3. *one finite* StreamName$^{\uparrow}$ *observation window shared by the two endpoint presentations;*
4. *two endpoint readbacks from that window, one for each endpoint row;*

5. one separation ledger witnessing that the two readbacks are separated by the radius ρ.

The tuple is accepted only when these six rows are visible through the same hsame, Cont, Pkg, *and* NameCert *provenance used by Definition 304.1 and Definition 304.2. It is a finite certificate row, not a host-level inequality between real numbers and not a quotient class of real-name presentations.*

Theorem 304.7 (ApartnessReal finite-window separation transport)**.** *Let R and S be accepted* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation rows in the sense of Definition 304.6. If the two endpoint rows, the positive radius row, the finite observation window, the two endpoint readbacks, and the separation ledger of R transport componentwise to the corresponding rows of S through the displayed* hsame, Cont, Pkg, *and* NameCert *routes, then S repacks as an accepted finite-window separation row.*

This transport adds no trichotomy principle, quotient carrier, total order, or completeness datum for constructive reals.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket finite window transport`

Proof. Project the accepted carrier data from Definition 304.1. The only coordinates present are the two endpoint presentations, the positive dyadic radius, the finite stream window, the two endpoint readbacks, and the separation ledger.

Use Definition 304.2 to transport those coordinates one by one. The real-name endpoints transport either through their displayed $\mathsf{Real}^{\uparrow}$ classifiers or through their $\mathsf{RegSeqRat}^{\uparrow}$ source classifiers before the real-seal boundary; the radius and window rows transport through their own classifiers; the endpoint readbacks follow the same hsame and Cont routes.

It remains to repack the ledger. Since the ledger is assumed to transport componentwise through the same visible Pkg and NameCert provenance, the target tuple has exactly the six rows required by Definition 304.6. No additional coordinate can be read from this repacking, so no excluded-order, quotient, or completion principle is exported. □

Theorem 304.8 (ApartnessReal metric consumer separation boundary)**.** *A later* $\mathsf{Met}^{\uparrow}$ *or real-order consumer may consume exactly the* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row of Definition 304.6, together with the existing handoff rows of Definition 304.5: the two displayed endpoint rows, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window, the two endpoint readbacks, and the visible separation ledger.*

Such a consumer cannot read a host real inequality, total order, trichotomy decision, quotient carrier, or completion datum from the $\mathsf{ApartnessReal}^{\uparrow}$ *certificate surface.*

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealMetricHandoffPacket metric consumer separation boundary`
Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket metric consumer separation boundary`
Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealCarrier metric consumer separation boundary`

Proof. Apply Definition 304.3. Its carrier, classifier, exactness, and ledger-visibility rows expose precisely the finite endpoint, radius, window, readback, and separation-ledger data named in Definition 304.6.

Apply Definition 304.4 to account for the reversible orientation required by apartness consumers. The symmetry operation swaps only the two endpoint rows and the two endpoint readbacks while preserving the same positive radius, finite window, and provenance rows.

Finally apply Definition 304.5. That handoff is already limited to finite positive-separation evidence for downstream metric or order constructions. Composing these three surfaces leaves the

consumer with the finite-window row and no additional coordinate from which host inequality, total order, quotient, or completion data could be extracted. □

Theorem 304.9 (ApartnessReal located-window positive gap handoff)**.** *Let L and R be displayed* $\mathsf{LocatedReal}^{\uparrow}$ *packets. Suppose they share a finite* $\mathsf{StreamName}^{\uparrow}$ *observation window and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers display two disjoint enclosures with positive dyadic gap row ρ. If the two endpoint real-name rows, the common finite window, the two endpoint readbacks, and the visible gap ledger are all carried by the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *provenance of the two packets, then the tuple consisting of those two endpoint real-name rows, ρ, the common finite window, the two endpoint readbacks, and the visible separation ledger repacks as an accepted* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row in the sense of Definition 304.6.*

The handoff exports finite positive-gap evidence only. It does not decide equality of real names, does not supply trichotomy or a total order, and does not identify located real packets by a quotient or host real equality.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket located window positive gap handoff`

Proof. Project both located packets through Definition 309.2. The carrier surface exposes the $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the $\mathsf{StreamName}^{\uparrow}$ schedule row, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint ledger, and the locatedness classifier rows, all tied to the visible provenance.

Apply Definition 309.12 to read the shared observation window as a finite scheduled window. Apply Definition 309.10 to identify the two endpoint readbacks with rows of the displayed dyadic endpoint ledgers.

The disjoint-enclosure hypothesis supplies the positive dyadic gap row ρ. Repack that row as the separation ledger, using the same common window, endpoint readbacks, and provenance rows already projected from the two located packets.

Finally apply Definition 304.7 and then Definition 304.8. The first theorem identifies the repacked tuple as a stable finite-window separation row, and the second theorem restricts every later metric or order consumer to exactly that finite apartness evidence. □

Theorem 304.10 (Localized Cauchy apartness bridge)**.** *Let L and R be displayed* $\mathsf{LocatedReal}^{\uparrow}$ *packets whose finite* $\mathsf{StreamName}^{\uparrow}$ *observation window,* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row, endpoint readbacks, and* hsame, Cont, Pkg, *and* NameCert *provenance agree componentwise. Suppose the displayed endpoint ledgers exhibit a positive located-window gap in the sense of Definition 304.9. Then the resulting* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row is stable under the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier routes and may be consumed by* $\mathsf{Real}^{\uparrow}$ *metric or order interfaces only as positive apartness evidence.*

The bridge exports no host inequality, quotient equality, trichotomy decision, total order, or completeness datum over the real or completion substrate.

Proof. Apply Definition 304.9 to the two displayed located-real packets. The shared finite window, regular Cauchy source rows, positive dyadic gap row, endpoint readbacks, and visible provenance repack as the finite separation row of Definition 304.6.

Next read the packet through Definition 304.2. The componentwise agreement hypotheses are exactly the classifier coordinates needed to transport the two endpoint rows, the common window, the gap row, the two readbacks, and the separation ledger along the displayed $\mathsf{RegSeqRat}^{\uparrow}$ routes.

Apply Definition 304.7 to make that transport stable. The transported packet is still the same finite positive-gap row, now presented over the regular Cauchy classifiers feeding the real-name substrate.

Finally apply Definition 304.8. It restricts every $\mathsf{Real}^{\uparrow}$ metric or order consumer to the finite positive-apartness row and excludes any readback of host inequality, quotient equality, trichotomy, total order, or completion data. □

Theorem 304.11 (ApartnessReal real separation scope)**.** *The scoped separation surface consists exactly of the two displayed* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows, one positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, one finite* $\mathsf{StreamName}^{\uparrow}$ *observation window, two endpoint readbacks, the separation ledger, and their* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Any $\mathsf{Met}^{\uparrow}$ *or real-order consumer may read those rows as finite positive-separation data. The surface provides no host inequality, trichotomy decision, quotient carrier, total order, or completeness datum.*

Proof. Combine the obligation surface of Definition 304.3 with the finite-window row of Definition 304.6. Together they expose exactly the two endpoint rows, positive radius, finite window, endpoint readbacks, separation ledger, and visible provenance.

Apply Definition 304.7 to keep that tuple stable under the displayed hsame, Cont, Pkg, and NameCert routes. Then apply Definition 304.8 to identify the same tuple as the downstream metric or order consumer surface.

The carrier and classifier coordinate lists contain only those finite rows. Therefore a consumer has no coordinate from which to read host inequality, trichotomy, quotient-carrier, total-order, or completeness data. □

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket real separation scope`

Theorem 304.12 (ApartnessReal mature consumer coverage)**.** *Every metric, order, or* $\mathsf{RealAlgOrder}^{\uparrow}$ *consumer of the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-separation packet may read only the finite carrier, classifier, metric-consumer separation boundary, bridge boundary,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already exposed by this chapter.*

The mature consumer surface does not export trichotomy, total order, completeness, quotient construction, metric completion, or standard constructive-real bridge data.

Lean4 ✓ `BEDC.Derived.ApartnessRealUp.ApartnessRealSeparationPacket mature consumer coverage`

Proof. Project the packet through Definition 304.6 and Definition 304.2. These rows expose the two endpoint names, positive radius, finite observation window, endpoint readbacks, separation ledger, and componentwise classifier transports.

Next apply Definition 304.7 and Definition 304.8. They restrict metric and order consumers to the same finite positive-separation carrier, classifier, hsame, Cont, Pkg, and NameCert rows.

Use Definition 304.9 and Definition 304.11 for the bridge boundary from located or sealed real-name packets to the displayed separation surface. These theorems repack the same finite rows and add no quotient, completion, or total-order coordinate.

The cited boundary theorems exhaust the rows available to metric, order, and $\mathsf{RealAlgOrder}^{\uparrow}$ consumers. Since the carrier, classifier, and bridge boundary contain no trichotomy, total order, completeness, quotient construction, metric completion, or standard constructive-real bridge datum, none is exported by the consumer coverage theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessReal}^{\uparrow}}$**.**

Constructive story: An apartness-real packet is generated from two displayed $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ names, a positive $\mathsf{DyadicRatCore}^{\uparrow}$ radius, a finite $\mathsf{StreamName}^{\uparrow}$ observation window, two endpoint readbacks, and a visible separation ledger transported componentwise by hsame and Cont.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessReal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public boundary binds exactly the finite positive-separation carrier, located-window positive gap handoff, metric consumer separation boundary, real-separation scope, and their $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert provenance for $\mathsf{ApartnessReal}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessReal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ApartnessRealUp.ApartnessRealCarrier\_metric\_consumer\_separation\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close trichotomy, total order, completeness, quotient construction, or metric-completion data.

Upgrade path: No further paper theory-axis grade is recorded; formal bridge work remains on the verification axis.

Chapter 305

A Concrete Naming Certificate for $\mathsf{BraidGroup}^{\uparrow}$

The braid-group interface $\mathsf{BraidGroup}^{\uparrow}$ packages the finite Artin-word presentation of the braid group as a BEDC naming surface. Its textbook source is the Artin generator presentation and braid-closure calculus appearing in Birman's braid monograph and in the Kassel–Turaev treatment of braid groups. The BEDC packet keeps the presentation finite: every generator letter, inverse letter, braid relation, inverse cancellation, far-commutation move, Weyl-action read, and knot-closure read is a displayed BHist, hsame, Cont, or Pkg row.

The finite Artin-word packet consumes the concatenation and inverse certificate surfaces of chapter 40 and chapter 41. It exports the symmetric shadow to chapter 117, the root-action ledger to chapter 145, and the closure ledger to chapter 254; these are dependency-lattice reads of the same displayed word, move, provenance, and naming rows rather than ambient presentation quotients.

305.1 Carrier And Ledger

Definition 305.1 (BraidGroup finite Artin-word carrier)**.** *A* $\mathsf{BraidGroup}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$\mathcal{B} = (n, w, L, C, \rho, \lambda)$$

whose source row n is a positive unary strand-count ledger, whose word row w is a finite list of signed adjacent-strand letters σ_i^{ϵ} with $1 \leq i < n$ and $\epsilon \in \{-1, +1\}$, and whose classifier ledger L is generated by visible Artin moves:

1. *inverse cancellation rows $\sigma_i\sigma_i^{-1} \rightsquigarrow \mathsf{e}$ and $\sigma_i^{-1}\sigma_i \rightsquigarrow \mathsf{e}$;*
2. *braid rows $\sigma_i\sigma_{i+1}\sigma_i \rightsquigarrow \sigma_{i+1}\sigma_i\sigma_{i+1}$ and their signed inverse transports;*
3. *far-commutation rows $\sigma_i\sigma_j \rightsquigarrow \sigma_j\sigma_i$ when the displayed unary gap records $|i - j| > 1$.*

The row C is the classifier witness obtained by composing such ledger entries. The row ρ records the consumed $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{WeylGroup}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$ *package sources used by downstream consumers. The finite ledger λ lists exactly the word letters, move rows, classifier transports, action readbacks, closure readbacks, and provenance rows accepted by the packet.*

305.2 Certificate Surface

Theorem 305.2 (BraidGroup Artin ledger stability)**.** *Let $\mathcal{B}$ and $\mathcal{B}'$ be* $\mathsf{BraidGroup}^{\uparrow}$ *packets over the same displayed strand-count row. If their Artin words are related by a finite classifier ledger generated by braid relations, inverse cancellation, and far-commutation moves, then every transported word row remains a* $\mathsf{BraidGroup}^{\uparrow}$ *word row, and the composed ledger remains visible as* BHist *carrier data with the same* hsame*,* Cont*, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket artin ledger stability`

Proof. Read the classifier ledger L entry by entry. Each generator of L rewrites a finite subword whose indices are already checked against the unary strand-count row, so the target subword is again a finite list of signed adjacent-strand letters over the same row.

For inverse cancellation, delete only the displayed adjacent inverse pair and keep the surrounding word prefix and suffix unchanged. For the braid and far-commutation rows, replace the named subword by the displayed target subword and attach the corresponding Cont route to the same prefix and suffix.

Composing the entries gives the transported word row and classifier witness C. Since all entries come from the finite ledger and every transport is recorded by hsame, Cont, and Pkg rows, the result is still a $\mathsf{BraidGroup}^{\uparrow}$ packet rather than a quotient of Artin words. □

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupPacket artin ledger stability`
Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket ledger stability`

Theorem 305.3 (BraidGroup NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{BraidGroup}^{\uparrow}}$ *obligation surface consists of the finite Artin-word carrier of Definition 305.1, the classifier ledger generated by braid relations, inverse cancellation, and far-commutation moves, the inherited monoid concatenation and group inverse rows from* $\mathsf{NameCert}_{\mathsf{Monoid}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$*, the permutation boundary consumed by* $\mathsf{NameCert}_{\mathsf{SymGroup}^{\uparrow}}$*, the braid-row action boundary consumed by* $\mathsf{NameCert}_{\mathsf{WeylGroup}^{\uparrow}}$*, the closure-ledger boundary consumed by* $\mathsf{NameCert}_{\mathsf{Knot}^{\uparrow}}$*, and the exact* BHist*,* hsame*,* Cont*, and* Pkg *rows listed by the packet ledger.*

Proof. Project the packet of Definition 305.1. The carrier supplies the strand count, signed adjacent-strand word, classifier ledger, classifier witness, dependency provenance, and finite ledger rows.

The monoid and group faces are the visible concatenation, empty-word, and inverse-word operations on the same finite word lists. Their stability is the Artin ledger stability of Definition 305.2, because concatenating a displayed move with fixed prefixes and suffixes keeps the same finite move visible.

The consumer faces are projections of the same packet: $\mathsf{SymGroup}^{\uparrow}$ reads the adjacent transposition shadow, $\mathsf{WeylGroup}^{\uparrow}$ reads the root-action braid rows, and $\mathsf{Knot}^{\uparrow}$ reads braid closure as a finite closure ledger. No face adds host word equality, a presentation quotient, or an unrecorded isotopy class. □

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupPacket namecert obligation surface`
Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket namecert obligation surface`

Theorem 305.4 (BraidGroup Weyl root-action handoff)**.** *Every* $\mathsf{WeylGroup}^{\uparrow}$ *braid-row consumer of a* $\mathsf{BraidGroup}^{\uparrow}$ *packet reads exactly the signed Artin letters, the adjacent-strand index rows, the braid and far-commutation ledger entries, the symmetric-group shadow row, and the* BHist*,*

hsame, Cont, *and* Pkg *provenance exposed by Definition 305.1. The handoff supplies a finite root-action ledger and supplies no Coxeter completion, host reflection equality, or Artin-representation faithfulness claim.*

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket weyl root action handoff`
Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupPacket weyl root action handoff`

Proof. Project the Artin word to its adjacent-strand letters. Each letter has a displayed index row, so the $\mathsf{WeylGroup}^{\uparrow}$ consumer can read it as the corresponding adjacent simple-reflection action row only through the packet ledger.

Use Definition 305.2 for transport along braid and far-commutation moves. These are precisely the moves that a finite Weyl braid row consumes when comparing two displayed reduced-action ledgers.

The symmetric-group shadow is read from the same adjacent transposition sequence, while provenance remains the Pkg row of the packet. Because the packet contains no host reflection equality or faithfulness field, the handoff is only the finite root-action ledger just described. □

Theorem 305.5 (BraidGroup knot-closure ledger boundary)**.** *Every* $\mathsf{Knot}^{\uparrow}$ *closure consumer of a* $\mathsf{BraidGroup}^{\uparrow}$ *packet reads the strand-count row, the finite signed Artin-word row, the crossing ledger induced by those letters, the Artin move ledger, the closure endpoint rows, and the* BHist, hsame, Cont, *and* Pkg *provenance exposed by the braid packet. The boundary records finite braid closures as knot-diagram ledgers and does not claim Markov theorem, knot isotopy completeness, braid group faithfulness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket knot closure empty boundary`

Proof. Read the signed Artin word from the carrier. Each positive or negative adjacent-strand letter determines one finite crossing entry with its incident strand indices and over-under orientation recorded in the closure ledger.

Transport of closure diagrams follows the Artin move ledger. Braid relations and far-commutations rewrite local crossing subledgers, while inverse cancellation removes a displayed opposite crossing pair; the closure endpoint rows remain attached to the same strand-count ledger.

The resulting object is the $\mathsf{Knot}^{\uparrow}$ crossing ledger read through the braid packet's Cont and Pkg rows. Since the packet does not contain a Markov-move classifier, an isotopy-completeness theorem, a faithfulness theorem, or a standard bridge row, none of those claims is exported by the closure boundary. □

Theorem 305.6 (BraidGroup scoped Artin source package)**.** *Every consumer-facing* $\mathsf{BraidGroup}^{\uparrow}$ *read is exhausted by the finite strand-count row, signed Artin word row, Artin move ledger, symmetric-group shadow, Weyl root-action handoff, knot-closure ledger, and the* BHist, hsame, Cont, *and* Pkg *provenance from the carrier. The package exports no Markov theorem, braid-group faithfulness, knot isotopy completeness, or standard bridge data.*

Proof. Project the carrier of Definition 305.1. It supplies the positive unary strand count, signed adjacent-strand word, classifier ledger, dependency provenance, and finite packet ledger.

Apply Definition 305.2 to identify every transported word read with the visible Artin move ledger. Then apply Definition 305.3 to account for the monoid, group, symmetric-group, Weyl-group, and knot dependency rows in the NameCert surface.

The two downstream faces are the finite projections already isolated by Definition 305.4 and Definition 305.5. Their exclusion clauses remove Markov moves, Artin-representation faithfulness,

knot isotopy completeness, and standard bridge data, leaving exactly the scoped Artin source package stated above. □

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket scoped artin source package`

Theorem 305.7 (BraidGroup public Artin interface)**.** *The finite signed Artin-word carrier, Artin ledger stability, NameCert obligation surface, Weyl root-action handoff, knot-closure ledger boundary, and scoped Artin source package assemble the public* $\mathsf{NameCert}_{\mathsf{BraidGroup}^{\uparrow}}$ *interface. It is readable by* $\mathsf{Group}^{\uparrow}$, $\mathsf{SymGroup}^{\uparrow}$, $\mathsf{WeylGroup}^{\uparrow}$, *and* $\mathsf{Knot}^{\uparrow}$ *consumers as strand-count rows, signed adjacent-strand word rows, braid/inverse/far-commutation ledgers, symmetric-group shadow rows, Weyl action rows, knot-closure ledger rows,* BHist, hsame, Cont, *and* Pkg *provenance.*

The interface exports no Markov theorem, braid-group faithfulness, knot isotopy completeness, presentation quotient, or standard bridge data.

Proof. Start with the finite carrier of Definition 305.1. It provides the strand count, signed Artin word, classifier ledger, dependency provenance, and packet rows that all four consumer directions read.

Apply Definition 305.2 to keep braid, inverse-cancellation, and far-commutation transport inside the displayed Artin ledger. Then use Definition 305.3 to expose the NameCert rows shared by the group, symmetric-group, Weyl-group, and knot readers.

The two non-group faces are exactly the finite handoffs isolated by Definition 305.4 and Definition 305.5. Finally Definition 305.6 gathers these rows into a single scoped Artin source package.

Since the cited package has no coordinate for Markov moves, representation faithfulness, isotopy completeness, presentation quotients, or standard bridge rows, repacking it as a public $\mathsf{NameCert}_{\mathsf{BraidGroup}^{\uparrow}}$ interface cannot export any of that data. □

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket public artin interface`

Theorem 305.8 (BraidGroup public NameCert export)**.** *The finite Artin-word carrier, Artin ledger stability, NameCert obligation surface, Weyl root-action handoff, knot-closure ledger boundary, and scoped Artin source package assemble the public* $\mathsf{NameCert}_{\mathsf{BraidGroup}^{\uparrow}}$ *export. Every consumer reads only signed Artin words, strand-count rows, braid/inverse/far-commutation ledgers, symmetric-group shadow, Weyl action handoff, knot closure ledger,* BHist, hsame, Cont, *and* Pkg *provenance, with no Markov theorem, braid faithfulness, knot-isotopy completeness, presentation quotient, or standard bridge.*

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket public namecert export`

Proof. Begin with the finite carrier of Definition 305.1. It displays the strand count, signed Artin word, classifier ledger, dependency provenance, and packet ledger that every public consumer can read.

Use Definition 305.2 and Definition 305.3. These theorems keep the braid, inverse, and far-commutation moves inside visible ledgers and identify the NameCert rows inherited by the group, monoid, symmetric-group, Weyl-group, and knot consumers.

Next apply Definition 305.4, Definition 305.5, and Definition 305.6. Their exclusion clauses remove Markov moves, braid faithfulness, knot-isotopy completeness, presentation quotients, and standard bridge data.

The cited carrier and theorem blocks exhaust the consumer-facing rows. Repacking them gives the public $\mathsf{NameCert}_{\mathsf{BraidGroup}^{\uparrow}}$ export with exactly the finite surface stated above. □

Theorem 305.9 (BraidGroup public certificate export)**.** *The finite Artin-word carrier, Artin ledger stability,* NameCert *obligation surface, Weyl root-action handoff, knot-closure ledger boundary, and scoped Artin source package assemble the public* $\mathsf{NameCert}_{\mathsf{BraidGroup}\uparrow}$ *export. The exported rows are exactly signed Artin words, strand counts, braid-relation ledgers, inverse-cancellation ledgers, far-commutation ledgers, symmetric-group shadow rows, Weyl root-action rows, knot-closure ledger rows,* BHist*,* hsame*,* Cont*, and* Pkg *provenance.*

This public certificate still withholds Markov theorem, braid group faithfulness, knot isotopy completeness, and standard bridge data.

Lean4 ✓ `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket public certificate export`

Proof. Project the carrier through Definition 305.1. The carrier provides only the finite strand-count row, signed Artin-word row, classifier ledger, dependency provenance, and packet ledger.

Next use Definition 305.2 and Definition 305.3. These results keep braid relations, inverse cancellation, far commutation, and the shared NameCert obligation rows inside the displayed finite ledger.

Apply Definition 305.4 and Definition 305.5 to account for the Weyl and knot-closure faces. Then apply Definition 305.6 to gather the group, symmetric-group, Weyl, and knot readers into one scoped Artin package.

Those cited theorem blocks exhaust the public certificate rows. None contains a Markov theorem, braid-group faithfulness theorem, knot-isotopy completeness theorem, or standard bridge datum, so the public export withholds exactly those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{BraidGroup}\uparrow}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BraidGroup}\uparrow}$, maturePackageClosed)

Scope closed: The scoped boundary binds finite signed Artin words, visible braid-relation, inverse-cancellation, and far-commutation ledgers, the NameCert obligation surface, the Weyl root-action handoff, the knot-closure ledger boundary, and the BHist, hsame, Cont, and Pkg rows consumed by symmetric-group, Weyl-group, and knot-closure readers.

Constructive story: Construct the packet by listing a unary strand count, a finite signed adjacent-strand word, and a finite move ledger generated by the Artin relations; downstream group, symmetric-group, Weyl-action, and knot-closure rows are projections of that same ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BraidGroup}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.BraidGroupUp.BraidGroupArtinPacket\_public\_certificate\_export`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not claim Markov theorem, braid group faithfulness, knot isotopy completeness, or a standard bridge.

Upgrade path: Formal bridge checking remains separate from the mature paper treatment of the finite Artin packet bridge surface.

Chapter 306

A Concrete Naming Certificate for $\mathsf{DyadicInterval}^{\uparrow}$

The $\mathsf{DyadicInterval}^{\uparrow}$ interface names finite dyadic interval packets in the sense of Moore–Kearfott–Cloud interval analysis. It sits between $\mathsf{DyadicRatCore}^{\uparrow}$, which supplies visible dyadic endpoint and radius ledgers, and $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, which consume finite approximation packets without importing quotient reals or completeness principles.

306.1 Carrier and refinement ledger

The endpoint-order, midpoint, radius, and width rows consume the dyadic arithmetic carrier of chapter 291, while the finite tolerance and readback rows align with chapter 322. Thus $\mathsf{DyadicInterval}^{\uparrow}$ is the interval packet assembled from those sibling dyadic ledgers before $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumes the finite approximation surface.

Definition 306.1 (DyadicInterval BHist carrier)**.** *A* $\mathsf{DyadicInterval}^{\uparrow}$ *carrier row is a finite* BHist *packet*

$$I = (L, R, W, O, M, \rho, H, C, P, N)$$

with the following visible rows:

1. *left and right endpoint rows L and R accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *a unary nonnegative width row W naming the displayed dyadic width bound;*
3. *an endpoint-order ledger O proving $L \leq R$ by a* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-exponent comparison;*
4. *midpoint and radius rows M and ρ computed from the same endpoint window;*
5. *componentwise* hsame *rows H and displayed* Cont *routes C for endpoint, order, midpoint, radius, and width transport;*
6. *a* Pkg *provenance row P naming the consumed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *source rows;*
7. *a* $\mathsf{NameCert}$ *row N recording that the public interval packet is classified only by endpointwise finite dyadic tolerance.*

The carrier has no coordinate for Dedekind cuts, open-cover compactness, quotient intervals, host real endpoints, or real-completeness data.

Theorem 306.2 (DyadicInterval nested refinement ledger)**.** *Let $I = (L, R, W, O, M, \rho, H, C, P, N)$ and $I' = (L', R', W', O', M', \rho', H', C', P', N')$ be accepted* DyadicInterval$^{\uparrow}$ *carriers. Suppose I' is obtained from I by a nested refinement: the transported endpoint rows satisfy $L \leq L' \leq R' \leq R$ in the displayed* DyadicRatCore$^{\uparrow}$ *common-exponent ledger, the width row W' is bounded by the transported width row W, and the midpoint and radius rows are recomputed from the refined endpoint window. Then I' repacks the same kind of endpoint, radius, width, order,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows as I.*

The finite source consumed by RegSeqRat$^{\uparrow}$ *is preserved: a scheduled regular-stream window may read the refined endpoint rows, radius row, width row, and provenance through the same visible finite dyadic interval surface, and it receives no quotient sequence, real-completion object, or hidden host-real endpoint.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket nested refinement ledger`

Proof. Project the carrier of Definition 306.1. The endpoint rows, width row, order ledger, midpoint row, radius row, componentwise hsame data, continuation routes, package provenance, and naming row are all visible finite rows.

Apply the nested-refinement hypotheses to the endpoint-order ledger. The refined endpoints are still accepted DyadicRatCore$^{\uparrow}$ rows, and the order comparisons are transported through the same common-exponent dyadic ledger.

Recompute the midpoint and radius rows from the refined endpoint window and transport the width row along the displayed nonnegative dyadic bound. These operations only repack the finite endpoint and arithmetic ledgers already named by the carrier.

Finally restrict the consumer view to the rows a RegSeqRat$^{\uparrow}$ scheduled window can read. That view consists exactly of endpoint, radius, width, regular dyadic comparison, hsame, Cont, Pkg, and NameCert provenance rows, so refinement preserves the finite source surface. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalEndpointPacket nested refinement ledger`

Theorem 306.3 (DyadicInterval nested window composition)**.** *Let $I_0 = (L_0, R_0, W_0, O_0, M_0, \rho_0, H_0, C_0, P_0, N_0)$, $I_1 = (L_1, R_1, W_1, O_1, M_1, \rho_1, H_1, C_1, P_1, N_1)$, and $I_2 = (L_2, R_2, W_2, O_2, M_2, \rho_2, H_2, C_2, P_2, N_2)$ be accepted* DyadicInterval$^{\uparrow}$ *carriers. Suppose I_1 is a nested refinement of I_0 and I_2 is a nested refinement of I_1 in the endpoint-order, width, midpoint, and radius ledgers of Definition 306.2. Then I_2 is a nested refinement of I_0.*

Moreover, the seal-source packet obtained by refining first from I_0 to I_1 and then from I_1 to I_2 exposes exactly the same endpoint, width, radius, order, hsame*,* Cont*,* Pkg*, and* NameCert *rows as the direct refinement from I_0 to I_2. It exposes no Dedekind cut, quotient interval, host-real endpoint, open-cover compactness witness, completeness datum, or standard bridge.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket nested window composition`

Proof. Project the three accepted packets through Definition 306.1. The only coordinates available for comparison are the endpoint rows, width row, order ledger, midpoint row, radius row, componentwise hsame rows, displayed Cont routes, Pkg provenance, and NameCert row.

Compose the endpoint-order ledgers. The hypotheses give $L_0 \leq L_1 \leq R_1 \leq R_0$ and $L_1 \leq L_2 \leq R_2 \leq R_1$ in the displayed DyadicRatCore$^{\uparrow}$ common-exponent comparisons, so transitivity of the finite comparison ledger gives $L_0 \leq L_2 \leq R_2 \leq R_0$. The width and radius transports compose in the same visible dyadic arithmetic rows, and the midpoint row is recomputed from the final endpoint window.

Apply Definition 306.2 to the first refinement and then to the second refinement. The two applications repack the same endpoint, width, radius, order, hsame, Cont, Pkg, and NameCert row kinds that the direct I_0 to I_2 refinement uses.

Finally apply Definition 306.7. That boundary exhausts the rows visible to a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ finite-window consumer, so the two-step seal source and the direct seal source have the same consumer-visible rows. Since the carrier has no coordinate for Dedekind cuts, quotient intervals, host-real endpoints, open-cover compactness, completeness data, or standard bridges, neither refinement path can expose one. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket nested window seal source composition`
Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket nested window name certificate`

Theorem 306.4 (DyadicInterval NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicInterval}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *interval packet of Definition 306.1;*
2. *classifier: endpointwise dyadic tolerance, comparing left endpoints and right endpoints by* $\mathsf{DyadicRatCore}^{\uparrow}$ *common-exponent windows while preserving the displayed width and radius rows;*
3. *endpoint-order stability: componentwise* hsame *and* Cont *transport preserve the visible* $L \leq R$ *ledger;*
4. *nested refinement transport: the refinement ledger of Definition 306.2 repacks the same finite endpoint, midpoint, radius, width, provenance, and naming rows;*
5. $\mathsf{Real}^{\uparrow}$ *seal handoff boundary: a real-name seal may consume only the finite interval endpoint, radius, width, order, and provenance rows and gains no real-completeness, quotient-interval, or standard-bridge datum.*

Proof. The carrier row is Definition 306.1. It lists the only public coordinates exposed by an accepted $\mathsf{DyadicInterval}^{\uparrow}$ packet.

The classifier row is endpointwise because the carrier has separate left and right $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows. A dyadic-tolerance witness therefore uses two common-exponent comparisons, one for each endpoint, together with the displayed width and radius rows.

Endpoint-order stability follows by transporting the order ledger through the componentwise hsame and Cont rows named in the carrier. Nested refinement transport is exactly the repacking statement of Definition 306.2.

At the $\mathsf{Real}^{\uparrow}$ seal boundary, the available source rows are exhausted by the carrier, classifier, order, and refinement rows above. Since none of those rows contains a Dedekind cut, an open-cover compactness object, a quotient interval, or completeness data, the handoff remains a finite BEDC source surface. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket namecert obligation surface`

Theorem 306.5 (DyadicInterval endpoint classifier transport)**.** *Let* $I = (L, R, W, O, M, \rho, H, C, P, N)$ *and* $I' = (L', R', W', O', M', \rho', H', C', P', N')$ *be accepted* $\mathsf{DyadicInterval}^{\uparrow}$ *carriers. Suppose the displayed* H *rows provide componentwise* hsame *witnesses and the displayed* C *rows provide* Cont *routes from the coordinates of* I *to the corresponding coordinates of* I'*. Then every endpointwise dyadic-tolerance witness for the left and right endpoint rows of* I *transports componentwise to* I'*, and the same transport repacks the order, midpoint, radius, width,* Pkg*, and* NameCert *rows.*

Proof. Project the two carriers through Definition 306.1. The left endpoint, right endpoint, width, order, midpoint, radius, package, and naming coordinates are separate finite rows, and the carrier exposes the matching hsame and Cont rows for each coordinate.

Transport the classifier in the endpoint coordinates first. The dyadic-tolerance witness is endpointwise, so the left-endpoint comparison follows the left endpoint route and the right-endpoint comparison follows the right endpoint route.

It remains to repack the non-endpoint rows. The order, midpoint, radius, and width data are read from the same displayed carrier coordinates, while the Pkg and NameCert rows are transported by their own visible routes. This gives the componentwise target classifier without adding any extra source datum. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalEndpointPacket endpoint classifier transport`
↪ variant `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket endpoint classifier transport`

Theorem 306.6 (DyadicInterval width-radius ledger exactness)**.** *Every width, midpoint, and radius readback used by the* DyadicInterval$^{\uparrow}$ *endpoint classifier is projected from the carrier rows of Definition 306.1 and, under nested refinement, from the transported rows supplied by Definition 306.2. No classifier readback may use an external host-real radius, quotient-interval width, or completion-side midpoint.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket width radius ledger exactness`

Proof. The carrier projection exposes exactly the width row W, midpoint row M, and radius row ρ. These are the only rows named in Definition 306.1 for width, midpoint, and radius readback.

Under nested refinement, apply Definition 306.2. The refined endpoint window repacks the transported width row and recomputes the midpoint and radius rows from that finite window.

Thus every classifier readback is a projection from either the original carrier or its nested-refinement transport. Since neither source contains a host-real, quotient-interval, or completion coordinate, no such coordinate can participate in the readback. □

Theorem 306.7 (DyadicInterval RealUp seal source boundary)**.** *A* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *consumer of a* DyadicInterval$^{\uparrow}$ *packet may read only the finite endpoint, radius, width, order, refinement,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in Definition 306.1, Definition 306.2, and Definition 306.4. The consumer receives no Dedekind cut, quotient interval, open-cover compactness witness, host real equality, or completeness datum.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket real seal source boundary`

Proof. Start with the carrier projection. Its finite rows provide the endpoint, radius, width, order, hsame, Cont, Pkg, and NameCert coordinates, and the carrier statement explicitly excludes Dedekind, quotient, host-real, and completeness coordinates.

Add the refinement view from Definition 306.2. Refinement repacks the same finite rows after endpoint narrowing; it does not create a new completion-side object.

Finally use the obligation surface of Definition 306.4. The seal handoff row is therefore bounded by the carrier, refinement, and naming projections, so a Real$^{\uparrow}$ or RegSeqRat$^{\uparrow}$ consumer can read only the listed finite DyadicInterval rows. □

Theorem 306.8 (DyadicInterval regular-window seal handoff)**.** *Let* $I = (L, R, W, O, M, \rho, H, C, P, N)$ *be an accepted* DyadicInterval$^{\uparrow}$ *packet. For any finite regular-stream observation window requested by a* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer, nested refinement transports the endpoint, width, radius, order,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of* I *into a finite seal-source packet. The consumer*

receives no Dedekind cut, quotient interval, host-real equality, open-cover compactness witness, or completeness datum.

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket regular window seal handoff`

Proof. Project the accepted packet through Definition 306.1. The available rows are exactly the finite endpoint, width, order, midpoint, radius, hsame, Cont, Pkg, and NameCert coordinates, with no completion-side coordinate in the carrier.

Apply Definition 306.2 to the requested regular-stream observation window. Its endpoint narrowing and width-radius readback repack the same finite rows after refinement, so the window source is again a dyadic interval packet rather than a quotient or host-real object.

Finally use Definition 306.7. The $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer boundary can read only the carrier and refinement rows just identified. Hence every regular-window read is exhausted by the finite seal-source rows transported from I. □

Theorem 306.9 (DyadicInterval scoped dependency package)**.** *The carrier of Definition 306.1, endpoint classifier transport, nested-refinement ledger, width-radius ledger exactness,* $\mathsf{Real}^{\uparrow}$ *seal source boundary, and regular-window seal handoff jointly form the scoped dependency package for* $\mathsf{DyadicInterval}^{\uparrow}$*. Its readable rows are exactly the* BHist *endpoint, width, midpoint, radius, order, refinement,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$*.*

The package exports no Dedekind cut, host real endpoint, quotient interval, open-cover compactness, real-completeness, or standard bridge datum.

Proof. Project the accepted carrier through Definition 306.1. This gives the finite endpoint, width, midpoint, radius, order, refinement, BHist, hsame, Cont, Pkg, and NameCert coordinates, with the arithmetic sources read through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Int}^{\uparrow}$ rows.

Next use Definition 306.5 for endpoint classifier movement, Definition 306.6 for the width-radius row, and Definition 306.2 for finite nested refinement. These results repack the same dyadic endpoint, observation-window, and history rows under displayed hsame and Cont transport.

Finally apply Definition 306.7 and Definition 306.8. The $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumer boundaries read the same finite package through the visible Pkg, NameCert, and $\mathsf{StreamName}^{\uparrow}$ schedule rows.

Since the carrier and the cited handoff rows exhaust every accepted coordinate, there is no source from which a Dedekind cut, host real endpoint, quotient interval, open-cover compactness, real-completeness, or standard bridge datum can enter the scoped dependency package. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket scoped dependency package`

Theorem 306.10 (DyadicInterval refined-width enclosure package)**.** *The carrier of Definition 306.1, endpoint classifier transport, nested refinement ledger, width-radius exactness,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and regular-window seal handoff assemble the public* $\mathsf{NameCert}_{\mathsf{DyadicInterval}^{\uparrow}}$ *refined-width enclosure package. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of this package may read only finite endpoint, radius, width, order, refinement,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The package exports no Dedekind cut, host real endpoint, quotient interval, open-cover compactness witness, completeness datum, or standard bridge datum.

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket refined width enclosure package`
↪ variant `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket refined width radius transport`

Proof. Project an accepted public certificate through Definition 306.1. The carrier exposes the finite dyadic endpoint rows, midpoint, width, radius, order, BHist, hsame, Cont, Pkg, and NameCert coordinates, and it has no coordinate for a host real endpoint or a quotient interval.

Next read the certificate rows through Definition 306.4. Its endpointwise classifier and order clauses are the public naming tests for the carrier, so they add no object beyond the displayed finite endpoint and order ledgers.

Apply Definition 306.2, Definition 306.5, and Definition 306.6. These results transport and refine the same endpoint, width, radius, order, hsame, and Cont rows, with exact width-radius readback inside the dyadic packet.

Finally apply Definition 306.7 and Definition 306.8. The $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ boundaries read only the finite package just identified: endpoint, radius, width, order, refinement, hsame, Cont, Pkg, and NameCert rows.

The cited projections exhaust the public refined-width package. Since none of those projections supplies Dedekind cuts, host real endpoints, open-cover compactness, quotient intervals, completeness data, or standard bridge data, no consumer-facing interval certificate can export such a row. □

Theorem 306.11 (DyadicInterval public RegSeq Real bridge)**.** *The public refined-width enclosure package of Definition 306.10 bridges to every finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *observation-window consumer by exposing exactly dyadic endpoint rows, radius rows, width rows, order rows, nested refinement transport, endpoint classifier transport, width-radius exactness,* $\mathsf{StreamName}^{\uparrow}$ *schedule provenance,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The bridge surface exports no Dedekind cuts, host real endpoints, quotient intervals, open-cover compactness, real completeness, or standard bridge data.

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket public regseq real bridge`

Proof. Project the package through Definition 306.1. The carrier contains the finite dyadic endpoint, radius, width, order, BHist, hsame, Cont, Pkg, and NameCert coordinates, and no Dedekind-cut, host-real, quotient-interval, compactness, completeness, or standard-bridge coordinate.

Apply Definition 306.2 to transport the finite endpoint and width rows along the observation-window refinement ledger. Use Definition 306.5 for endpoint classifier movement and Definition 306.6 for the exact width-radius row.

Next read the consumer boundary through Definition 306.7 and Definition 306.8. These theorems attach the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ finite-window reads to the same $\mathsf{StreamName}^{\uparrow}$ schedule provenance and the same dyadic packet rows.

Finally repack the rows using Definition 306.10. The consumer bridge is therefore only the finite readback package just listed, so none of the excluded standard interval or real-completeness data is available. □

Theorem 306.12 (DyadicInterval standard interval bridge boundary)**.** *The standard interval-analysis reading of a public* $\mathsf{DyadicInterval}^{\uparrow}$ *refined-width enclosure may consume exactly the finite endpoint, radius, width, order, nested-refinement, endpoint-transport, width-radius exactness,* $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows isolated by Definition 306.10. It may not read Dedekind cuts, quotient intervals, host real endpoints, open-cover compactness, real completeness, or external interval libraries.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket standard interval bridge boundary`

Proof. Project the enclosure through Definition 306.1. This gives the finite endpoint, radius, width, order, hsame, Cont, Pkg, and NameCert coordinates, with no coordinate for a host real endpoint, Dedekind cut, or quotient interval.

Use Definition 306.2 for the finite refinement rows, Definition 306.5 for endpoint transport, and Definition 306.6 for the exact width-radius readback. These rows keep the interval-analysis data inside the same dyadic packet.

Finally apply Definition 306.7, Definition 306.8, and Definition 306.10. The public refined-width theorem already exhausts the rows readable by $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers, so a standard interval-analysis bridge has only the finite rows listed above. □

Theorem 306.13 (DyadicInterval regular-window bridge boundary)**.** *The refined-width package of Definition 306.10, together with the nested-refinement ledger, width-radius exactness row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and regular-window seal handoff, forms the paper bridge from finite* $\mathsf{DyadicInterval}^{\uparrow}$ *packets to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *finite observation windows. The consumer-visible rows are exactly endpoint, radius, width, order, refinement,* hsame*,* Cont*,* Pkg*,* NameCert*, and* $\mathsf{StreamName}^{\uparrow}$ *schedule rows.*

The bridge boundary still excludes Dedekind cuts, quotient intervals, host real endpoints, open-cover compactness, real completeness, and standard bridge data.

Lean4 ✓ `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket regular window bridge boundary`

Proof. Project the public package through Definition 306.10. This exposes the finite endpoint, radius, width, order, refinement, hsame, Cont, Pkg, and NameCert rows, while withholding host endpoints, quotient intervals, Dedekind cuts, compactness witnesses, completeness data, and standard bridge rows.

Next apply Definition 306.2 and Definition 306.6. These projections identify the refinement row and the exact width-radius readback as rows already present in the dyadic packet rather than as external real-analysis structure.

Finally use Definition 306.7 and Definition 306.8. They attach the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ finite-window reads to the same $\mathsf{StreamName}^{\uparrow}$ schedule rows and exhaust every row available at the seal boundary.

The cited projections account for every consumer-visible coordinate. Since no named component contains any of the excluded data, the paper bridge has exactly the finite regular-window surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicInterval}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicInterval}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public refined-width enclosure package is exactly the finite BHist interval carrier, endpoint-wise classifier, endpoint-order ledger, nested-refinement transport, endpoint transport, width-radius exactness row, and $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ seal boundary over $\mathsf{DyadicRatCore}^{\uparrow}$.

Constructive story: Construct the interval packet by finite dyadic endpoint reads: $\mathsf{DyadicRatCore}^{\uparrow}$ supplies endpoints and arithmetic ledgers, the order and width rows record finite bounds, and refinement repacks the same visible rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicInterval}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DyadicIntervalUp.DyadicIntervalPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Dedekind cuts, open-cover compactness, real completeness, quotient intervals, or standard bridge data.

Upgrade path: No further paper theory-axis step is claimed for this chapter; formal bridge verification remains a separate status.

Chapter 307

A Concrete Naming Certificate for $\mathsf{Riccati}^{\uparrow}$

The $\mathsf{Riccati}^{\uparrow}$ interface records the finite control-theory ledger used by finite-horizon regulator and filtering certificates. Its textbook object is the Riccati recursion for linear-quadratic control and covariance propagation, as in Lancaster and Rodman, *The Algebraic Riccati Equation* (1995), or Anderson and Moore, *Optimal Control: Linear Quadratic Methods* (1990). The BEDC certificate is only the named finite packet consumed by $\mathsf{LQR}^{\uparrow}$ and $\mathsf{KalmanFilter}^{\uparrow}$.

The Riccati packet reads its matrix rows through chapter 49, its vector-space rows through chapter 47, its real-scalar rows through chapter 38, and its dynamic-system rows through chapter 195. Its downstream readers are chapter 296 and chapter 5102.

Definition 307.1 (RiccatiUp control packet carrier)**.** *A* $\mathsf{Riccati}^{\uparrow}$ *carrier is a finite* BHist *packet whose displayed rows are matrix state rows, matrix control rows, symmetric quadratic-cost rows, a finite horizon unary index, Riccati value-row ledger entries, backward* Cont *routes from each successor value row to the preceding value row,* hsame *transports for the state, control, cost, horizon, and value rows, and* Pkg *provenance naming* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*, and* $\mathsf{KalmanFilter}^{\uparrow}$*.*

The classifier accepts exactly the packets whose finite value-row ledger is read through these displayed matrix, vector-space, real-scalar, dynamic-system, continuation, transport, and provenance rows. It exports no host analytic operator equation, infinite-horizon limit, stabilizing solution, or external solver state.

Theorem 307.2 (RiccatiUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ *obligation surface has five rows:*

1. *the carrier row of Definition 307.1;*
2. *the classifier row accepting exactly the finite Riccati value-ledger packets with displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DynSystem}^{\uparrow}$ *sources;*
3. *the stability row saying componentwise* hsame *transport of state, control, symmetric-cost, horizon, value,* Cont*, and* Pkg *rows remains inside the same carrier surface;*
4. *the finite ledger-closure row saying each backward value step consumes only the displayed successor value, transition, control, symmetric-cost, horizon, continuation, and provenance rows;*
5. *the consumer-boundary row saying* $\mathsf{LQR}^{\uparrow}$ *and* $\mathsf{KalmanFilter}^{\uparrow}$ *may read only the displayed finite Riccati rows and cannot demand an algebraic infinite-horizon equation or external solver from the certificate.*

Lean4 ✓ `BEDC.Derived.RiccatiUp.RiccatiControlPacketCarrier component transport`

Proof. Construct the surface by enumerating the finite carrier coordinates. The carrier row supplies the visible state, control, symmetric-cost, horizon, value, continuation, transport, and provenance fields.

The classifier is the same finite membership test restricted to those rows. Transport stability follows by replacing each visible row by its hsame match and reassembling the packet with the same horizon index and provenance names.

For ledger closure, inspect one backward horizon position. Its predecessor value row is generated from the displayed successor value, transition, control, symmetric-cost, continuation, and package rows, so no host Riccati equation or solver coordinate is available.

The consumer boundary is then a projection statement. $\mathsf{LQR}^{\uparrow}$ reads the finite value recursion as its dynamic-programming ledger, while $\mathsf{KalmanFilter}^{\uparrow}$ reads the covariance-style matrix ledger; both reads are confined to the displayed BHist coordinates. □

Theorem 307.3 (RiccatiUp finite ledger step closure)**.** *In an accepted* $\mathsf{Riccati}^{\uparrow}$ *packet, every finite backward ledger step is a* Cont *row from the displayed successor value row, state-transition row, control row, symmetric-cost row, finite horizon row, and* Pkg *provenance to the displayed predecessor value row. No accepted step reads an unlogged analytic value function, limiting algebraic Riccati solution, or external solver state.*

Lean4 ✓ `BEDC.Derived.RiccatiUp.RiccatiControlPacket finite ledger step closure`

Proof. Fix a backward position in the finite horizon unary index. By Definition 307.1, the only rows available at that position are the successor value row, transition and control rows, symmetric-cost row, continuation row, transport rows, and provenance rows.

The classifier in Definition 307.2 accepts the predecessor row only when it is assembled from those displayed coordinates. Since the carrier has no coordinate for a limiting solution or solver state, such data cannot contribute to the accepted step. □

Theorem 307.4 (RiccatiUp LQR and Kalman consumer boundary)**.** *Every* $\mathsf{LQR}^{\uparrow}$ *or* $\mathsf{KalmanFilter}^{\uparrow}$ *consumer of* $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ *may read only the finite value-row ledger, covariance-style matrix rows, horizon rows,* hsame *transports,* Cont *routes, and* Pkg *provenance of Definition 307.1. It cannot infer an infinite-horizon algebraic Riccati limit, a stabilizing solution theorem, or an external numerical solver from the* $\mathsf{Riccati}^{\uparrow}$ *certificate alone.*

Lean4 ✓ `BEDC.Derived.RiccatiUp.RiccatiControlPacket lqr kalman consumer boundary`

Proof. Read the consumer as a projection from the accepted carrier. The $\mathsf{LQR}^{\uparrow}$ side projects the state, control, symmetric-cost, horizon, and value-ledger rows needed for finite dynamic programming.

The $\mathsf{KalmanFilter}^{\uparrow}$ side projects the covariance-style matrix value ledger, transport rows, continuation rows, and provenance rows. These are the same finite BHist coordinates, interpreted by the filtering packet as prediction-update covariance data.

All remaining objects named in the theorem are absent from the carrier. Hence they are not readable through the $\mathsf{Riccati}^{\uparrow}$ certificate. □

Theorem 307.5 (RiccatiUp state-space control dependency)**.** *Every accepted* $\mathsf{Riccati}^{\uparrow}$ *finite value ledger must first read the state-transition and control rows through chapter 329. Only after those*

rows are displayed may $\mathsf{LQR}^{\uparrow}$ *and* $\mathsf{KalmanFilter}^{\uparrow}$ *consumers read the finite value and covariance-style rows through chapter 296 and chapter 5102. The dependency introduces no solver row and no infinite-horizon row.*

Proof. Project the carrier of Definition 307.1. Its visible fields include matrix state rows, matrix control rows, symmetric-cost rows, a finite horizon row, value ledger entries, continuation rows, transports, and provenance.

The state and control fields are the state-space read. They are therefore consumed through chapter 329 before any downstream regulator or filter view can read the value ledger.

The $\mathsf{LQR}^{\uparrow}$ consumer then reads the finite cost, horizon, control, and value rows through chapter 296. The $\mathsf{KalmanFilter}^{\uparrow}$ consumer reads the covariance-style matrix value rows through chapter 5102.

All three reads are projections from the same finite packet. Since the carrier has no solver coordinate and no infinite-horizon coordinate, the dependency row records only the displayed state-space, control, value, and covariance ledger. □

Theorem 307.6 (RiccatiUp seed NameCert surface)**.** *The carrier of Definition 307.1, together with the obligation surface of Definition 307.2, the finite ledger step closure of Definition 307.3, and the consumer boundary of Definition 307.4, forms the seed* $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ *surface over* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{KalmanFilter}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* Pkg*.*

Proof. Assemble the carrier and obligation rows first. The carrier names every row that can be exported, and the obligation surface supplies the classifier, transport, ledger, and consumer-boundary fields.

Apply the finite ledger step closure to each horizon position in the unary index. Each backward row is therefore a displayed continuation between finite value rows with visible matrix, cost, horizon, and provenance inputs.

Finally, apply the LQR and Kalman consumer boundary. The resulting $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ surface is exactly the displayed packet and has no field for an infinite-horizon algebraic Riccati limit or an external solver. □

Theorem 307.7 (RiccatiUp finite horizon tail restriction)**.** *Let P be an accepted* $\mathsf{Riccati}^{\uparrow}$ *packet and let T be a displayed tail of its finite unary horizon. Restricting P to T repacks the same matrix state rows, matrix control rows, symmetric-cost rows, value ledger, backward* Cont *routes,* hsame *transports, and* Pkg *provenance over the tail horizon. Every* $\mathsf{LQR}^{\uparrow}$ *or* $\mathsf{KalmanFilter}^{\uparrow}$ *consumer of the restricted packet factors through these displayed rows. The restricted packet exports no infinite-horizon algebraic Riccati limit, stabilizing solution theorem, or external solver.*

Lean4 ✓ `BEDC.Derived.RiccatiUp.RiccatiControlPacketCarrier finite horizon tail restriction`

Proof. Apply Definition 307.2 to the accepted packet. The carrier, classifier, stability, ledger, and consumer-boundary rows are already listed as finite BHist coordinates with a unary horizon index.

Restrict to the displayed tail by projecting the horizon positions and keeping the same state, control, symmetric-cost, value, continuation, transport, and provenance coordinates on those positions. The classifier row from the obligation surface accepts the restricted packet because it is assembled from the same displayed finite coordinates.

Use Definition 307.3 for each retained backward position. Each predecessor value row is still generated only from the displayed successor value, transition, control, symmetric-cost, continuation, horizon, and package rows, so tail restriction does not introduce a new ledger source.

Finally apply Definition 307.4. The $\mathsf{LQR}^{\uparrow}$ and $\mathsf{KalmanFilter}^{\uparrow}$ consumers of the restricted packet can read only the displayed tail rows just described. Since the carrier has no coordinate for an infinite-horizon algebraic Riccati limit, a stabilizing solution theorem, or an external solver, none of those objects is exposed by the restricted packet. □

Theorem 307.8 (RiccatiUp public finite-ledger certificate)**.** *The carrier of Definition 307.1, the obligation surface of Definition 307.2, the finite ledger step closure of Definition 307.3, the LQR and Kalman consumer boundary of Definition 307.4, the seed NameCert surface of Definition 307.6, and the finite horizon tail restriction of Definition 307.7 assemble the public* $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ *certificate over* $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, BHist, hsame, Cont, *and* Pkg.

The certificate exposes only the finite control ledger. It exports no infinite-horizon algebraic Riccati limit, stabilizing solution theorem, convergence rate, or external solver.

Proof. Start with the seed surface of Definition 307.6. It packages the carrier, classifier, finite value ledger, component transport, and consumer boundary as the visible finite $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$ packet.

Apply the tail restriction of Definition 307.7. The same displayed matrix, vector-space, real-scalar, dynamic-system, continuation, transport, and provenance rows remain stable under the finite horizon views that downstream consumers may request.

Finally use the LQR and Kalman consumer boundary of Definition 307.4. It identifies the public consumers and excludes infinite-horizon limits, stabilizing solution theorems, convergence data, and external solvers. Repacking these displayed rows gives the stated public certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$**.**

Constructive story: A Riccati row is seeded by a finite BHist packet of matrix state and control rows, symmetric-cost rows, a finite horizon unary index, value-row ledger entries, backward Cont routes, hsame transports, and Pkg provenance naming $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, and $\mathsf{KalmanFilter}^{\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public Riccati finite-ledger export is scoped by the component-transport, finite-ledger-step, $\mathsf{LQR}^{\uparrow}$/$\mathsf{KalmanFilter}^{\uparrow}$ consumer-boundary, finite-horizon tail-restriction, and public certificate rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Riccati}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RiccatiUp.RiccatiControlPacketCarrier\_component\_transport`

Bridge status: none

Origin: human

Not claimed: This chapter does not close infinite-horizon algebraic Riccati limits, stabilizing solution theorems, convergence rates, or external solvers.

Upgrade path: Only a separate standard bridge verification target can raise the independent formal bridge status.

Chapter 308

A Concrete Naming Certificate for $\mathsf{CofibrantReplacement}^{\uparrow}$

The $\mathsf{CofibrantReplacement}^{\uparrow}$ interface names the finite model-category row that displays one cofibrant replacement witness for one object. Hovey's *Model Categories* treats cofibrant replacement as a standard model-category construction; the BEDC surface below records only a finite displayed factorization packet, not a selection of replacements for every object.

The dependency boundary is $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, and $\mathsf{Cat}^{\uparrow}$. The model-category row supplies cofibration, weak-equivalence, and factorization surfaces; the simplicial-set row supplies a concrete displayed source family when the model surface is read in that setting; the homotopy row records the weak-equivalence ledger; and the category row supplies the object and arrow typing rows.

The factorization row is consumed concretely through chapter 158: $\mathsf{ModelCat}^{\uparrow}$ supplies the cofibration, weak-equivalence, and factorization faces that the $\mathsf{CofibrantReplacement}^{\uparrow}$ packet repackages as one displayed witness, without a global replacement choice or homotopy-category quotient.

308.1 Finite factorization source

Definition 308.1 (CofibrantReplacement BHist source)**.** *A* $\mathsf{CofibrantReplacement}^{\uparrow}$ *source is a finite* BHist *packet*

$$(X, Q, i, p, F, \rho, \lambda)$$

whose visible rows have the following shape:

1. *X is the displayed object row read from* $\mathsf{ModelCat}^{\uparrow}$ *and typed through* $\mathsf{Cat}^{\uparrow}$*;*
2. *Q is the displayed cofibrant object row over the same model-category surface;*
3. *$i : Q \to X$ is the displayed arrow row whose model-category ledger marks it as a cofibration and as a weak equivalence;*
4. *p is the factorization provenance row recording the finite factorization step that produced Q and i;*
5. *F is a finite list of lifting-test rows attached to the displayed cofibration face;*
6. *ρ is the package row naming the* $\mathsf{ModelCat}^{\uparrow}$*,* $\mathsf{SimplicialSet}^{\uparrow}$*,* $\mathsf{Homotopy}^{\uparrow}$*, and* $\mathsf{Cat}^{\uparrow}$ *dependency certificates used by the packet;*

7. *λ is the continuation ledger that records the* Cont *routes and componentwise* hsame *comparisons used to read the packet downstream.*

The carrier predicate accepts exactly such packets. Its classifier compares accepted packets componentwise by the object, cofibrant-object, arrow, factorization, lifting-test, package, and continuation-ledger hsame *rows, and it does not identify replacements by a homotopy-category quotient.*

308.2 Certificate obligations

Theorem 308.2 (CofibrantReplacement factorization obligation)**.** *Every accepted* $\mathsf{CofibrantReplacement}^{\uparrow}$ *source exposes a finite factorization row over* $\mathsf{ModelCat}^{\uparrow}$*: the displayed object X, the displayed cofibrant object Q, the arrow $i : Q \to X$, the cofibration ledger for i, the weak-equivalence ledger for i, and the provenance row p that records this factorization as part of the displayed model-category surface.*

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementBHistSource factorization obligation`

Proof. Project the source packet of Definition 308.1. The first four coordinates are exactly X, Q, i, and p, and the arrow coordinate is stored with both model-category marks.

The category typing of the arrow is read through the $\mathsf{Cat}^{\uparrow}$ rows named in ρ. The model-category marks are read through the $\mathsf{ModelCat}^{\uparrow}$ row, while the provenance coordinate records that these rows come from a finite factorization step.

Since the carrier has no coordinate for a family of replacements, the obligation is exhausted by the displayed packet for this one object. □

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementBHistSource factorization readback`

Theorem 308.3 (CofibrantReplacement weak-equivalence ledger)**.** *In an accepted* $\mathsf{CofibrantReplacement}^{\uparrow}$ *source, every weak-equivalence read of $i : Q \to X$ factors through the finite homotopy and model-category ledger carried by the tuple $(X, Q, i, p, F, \rho, \lambda)$. Transporting the packet along its displayed* hsame *rows preserves the weak-equivalence ledger and introduces no localization, derived mapping space, or homotopy-category quotient.*

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementPacket weak equivalence ledger transport`

Proof. Read the arrow coordinate i in Definition 308.1. Its weak-equivalence mark is one of the displayed rows of the accepted packet, and ρ names the $\mathsf{Homotopy}^{\uparrow}$ and $\mathsf{ModelCat}^{\uparrow}$ certificates through which that mark is interpreted.

Transport uses the componentwise hsame comparisons stored in λ. These comparisons carry the source object, cofibrant object, arrow row, and weak-equivalence mark together, so the target packet reads the same weak-equivalence ledger.

No localization or quotient row is available in the source tuple. Therefore all weak-equivalence information remains inside the displayed finite ledger. □

Theorem 308.4 (CofibrantReplacement NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CofibrantReplacement}^{\uparrow}}$ *obligation surface consists of five finite rows: the source carrier of Definition 308.1; the componentwise classifier over X, Q, i, p, F, ρ, and λ; the finite factorization obligation of Definition 308.2;*

the weak-equivalence ledger of Definition 308.3; and the dependency-boundary row over $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, *and* $\mathsf{Cat}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementNameCertObligationSurface dependency transport`

Proof. The carrier and classifier are the two direct projections of the source definition. They list the accepted finite packets and the componentwise comparison relation used between them.

The factorization row is supplied by Definition 308.2. It records the object, cofibrant object, arrow, cofibration mark, weak-equivalence mark, and factorization provenance.

The weak-equivalence row is supplied by Definition 308.3. The remaining dependency-boundary row is the package coordinate ρ, which names exactly the four certificates used by the construction. □

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementBHistSource namecert obligation surface`

Theorem 308.5 (CofibrantReplacement dependency boundary)**.** *Any consumer of* $\mathsf{NameCert}_{\mathsf{CofibrantReplacement}^{\uparrow}}$ *may read only the displayed* $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, *and* $\mathsf{Cat}^{\uparrow}$ *dependency rows, the factorization packet* $(X, Q, i, p, F, \rho, \lambda)$*, and the associated* hsame, Cont, Pkg, *and* NameCert *provenance. The packet does not export functorial cofibrant replacement, a global replacement assignment, localization data, derived-category objects, or homotopy-category quotients.*

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementPacket dependency boundary`
Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementBHistSource dependency boundary`

Proof. Project the package coordinate ρ. It names precisely the model-category, simplicial-set, homotopy, and category certificates used by the source.

Project the continuation ledger λ. It contains the visible Cont routes and componentwise hsame rows by which downstream consumers may transport the factorization packet.

All excluded objects require coordinates absent from the source tuple. A consumer can therefore read the single displayed factorization witness but cannot promote it to a functor, a global assignment, a localization, or a quotient category. □

Theorem 308.6 (CofibrantReplacement scoped kernel dependency package)**.** *Every accepted finite* $\mathsf{CofibrantReplacement}^{\uparrow}$ *factorization packet* $(X, Q, i, p, F, \rho, \lambda)$ *exposes its consumer-readable scope exactly through the displayed* BHist *packet rows, componentwise* hsame *comparisons, visible* Cont *routes, the dependency package* ρ*, and the local* NameCert *surface over* $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, *and* $\mathsf{Cat}^{\uparrow}$.

The scoped package exports no global choice of replacements, functorial replacement operation, localization, derived mapping space, or homotopy-category quotient.

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementBHistSource scoped kernel dependency package`

Proof. Project the dependency package ρ and the continuation ledger λ. These coordinates name the four dependency certificates and the visible routes by which the finite packet may be transported componentwise.

Apply Definition 308.2 to read the object row X, the cofibrant object row Q, the arrow i, the factorization provenance p, and the finite lifting-test row F as displayed packet data. Apply

Definition 308.3 to keep the weak-equivalence read of i inside the same homotopy and model-category ledger.

Finally use Definition 308.5. It identifies the consumer boundary with the factorization packet, ρ, λ, and their visible provenance over the four named dependencies.

Since the source tuple contains no coordinate for a global replacement function, a functorial assignment, a localization, a derived mapping object, or a homotopy-category quotient, none of those objects can be exported by the scoped package. □

Theorem 308.7 (CofibrantReplacement five-row NameCert sketch)**.** *The* $\mathsf{CofibrantReplacement}^{\uparrow}$ *source supplies the five expected* NameCert *checks at the displayed finite boundary:*

1. *carrier inhabitance is witnessed by a displayed finite factorization row* $(X, Q, i, p, F, \rho, \lambda)$*;*
2. *classifier reflexivity, symmetry, and transitivity are componentwise* hsame *checks on* X*,* Q*,* i*,* p*,* F*,* ρ*, and* λ*;*
3. *carrier transport along the factorization ledger repacks the same object, cofibrant-object, arrow, factorization, and lifting-test rows;*
4. *weak-equivalence ledger stability follows from the transported* $\mathsf{Homotopy}^{\uparrow}$ *and* $\mathsf{ModelCat}^{\uparrow}$ *rows for* i*;*
5. *dependency-boundary non-leakage follows because* ρ *names only* $\mathsf{ModelCat}^{\uparrow}$*,* $\mathsf{SimplicialSet}^{\uparrow}$*,* $\mathsf{Homotopy}^{\uparrow}$*, and* $\mathsf{Cat}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CofibrantReplacementUp.CofibrantReplacementPacket five row namecert surface`

Proof. For carrier inhabitance, use the displayed finite factorization row itself. It supplies the object, cofibrant object, arrow, factorization provenance, lifting tests, package row, and continuation ledger.

For the classifier, compare accepted packets componentwise. Reflexivity, symmetry, and transitivity are exactly the corresponding hsame moves on each coordinate, repacked into the tuple classifier.

For transport, follow the continuation ledger λ. It carries each coordinate of the factorization row through the displayed routes and repacks the target tuple with the same shape.

The weak-equivalence and dependency claims are the two boundary projections of the transported packet. The weak-equivalence mark remains attached to i through the $\mathsf{Homotopy}^{\uparrow}$ and $\mathsf{ModelCat}^{\uparrow}$ rows, and ρ contains no dependency certificate beyond the four named rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofibrantReplacement}^{\uparrow}}$**.**

Constructive story: A cofibrant-replacement packet is generated by displaying one object row X, one cofibrant object row Q, one arrow $i : Q \to X$ marked as a cofibration and weak equivalence, one factorization provenance row, finite lifting tests, dependency certificates, and continuation routes.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofibrantReplacement}^{\uparrow}}$, publicClosed)

Scope closed: The scoped boundary binds the finite factorization packet, weak-equivalence ledger, dependency package, and scoped kernel dependency theorem over BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofibrantReplacement}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close functorial cofibrant replacement, global choice of replacements, localization, derived mapping spaces, or homotopy category quotients.

Upgrade path: Bridge closure requires a standard bridge boundary for the displayed cofibrant-replacement packet.

Chapter 309

A Locatedness Route Surface for $\mathsf{LocatedReal}^{\uparrow}$

The $\mathsf{LocatedReal}^{\uparrow}$ surface names constructive located real-name packets between $\mathsf{RegSeqRat}^{\uparrow}$ and later metric or topological real constructions. Following the Bishop–Bridges constructive-analysis treatment of located real data, the chapter keeps interval endpoints, radii, and observation schedules as visible finite rows; it does not identify real names by a quotient or by host real equality.

Theorem 309.1 (LocatedReal L10 sibling dependency route)**.** *The concrete* $\mathsf{LocatedReal}^{\uparrow}$ *carrier route passes through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence source of chapter 301, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation schedule of chapter 126, the* DyadicRatCore *endpoint and radius ledger of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of chapter 38.*

Thus every public $\mathsf{LocatedReal}^{\uparrow}$ *consumer receives only this finite L10 route: scheduled regular rational windows, dyadic endpoint and radius rows, enclosure evidence, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The route contains no quotient real, host equality, trichotomy, open-cover compactness, or completion datum.*

Proof. Project the carrier surface of Definition 309.2. Its first and third components identify the $\mathsf{RegSeqRat}^{\uparrow}$ source and the $\mathsf{StreamName}^{\uparrow}$ schedule as the only route by which a located real packet reads regular rational observations.

The second component and the locatedness classifier identify the DyadicRatCore ledger as the only source of dyadic endpoints, dyadic radii, and enclosure rows. Therefore every locatedness read factors through the displayed finite endpoint/radius ledger rather than through a completed real object.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal compatibility row of Definition 309.5. It keeps the public handoff at the finite source-and-ledger surface, so downstream consumers receive exactly the L10 route stated above and none of the excluded quotient, equality, order-decision, compactness, or completion structure. □

309.1 Carrier and classifier

Definition 309.2 (LocatedReal carrier surface)**.** *A* $\mathsf{LocatedReal}^{\uparrow}$ *packet consists of four displayed components:*

1. *a* $\mathsf{RegSeqRat}^{\uparrow}$ *real-name source carrying the scheduled regular rational stream, dyadic-radius rows, regularity ledgers, and finite-window transport provenance;*

2. *a* DyadicRatCore *interval ledger naming dyadic endpoint pairs* (a_r, b_r) *for each displayed radius row* r*;*
3. *a* StreamName$^{\uparrow}$ *finite-observation schedule naming the finite rows at which the regular-sequence window is read;*
4. *a locatedness classifier which, for every displayed dyadic radius row, records a certified endpoint pair enclosing the current regular-sequence window.*

The enclosure row says that each endpoint read selected by the schedule lies inside the displayed dyadic interval for that radius, with the two endpoints read through the same DyadicRatCore *ledger and the same* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The packet stores only these finite rows. It contains no Dedekind cut, quotient real, trichotomy decision, open-cover compactness datum, or host real equality.

Theorem 309.3 (LocatedReal dyadic interval obligation)**.** *The local interval obligation for a* LocatedReal$^{\uparrow}$ *packet is the finite row that, for every displayed dyadic radius, exposes the* StreamName$^{\uparrow}$ *schedule row, the current* RegSeqRat$^{\uparrow}$ *regular-sequence window, the two* DyadicRatCore *endpoint rows, and the enclosure ledger proving that all displayed window endpoints lie between them.*

The obligation is exhausted by these dyadic endpoint, radius, schedule, regularity, hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project the carrier surface of Definition 309.2. The schedule determines the finite regular-sequence window, and the interval ledger determines the displayed dyadic endpoint pair for the requested radius.

The locatedness classifier is a finite enclosure row over those same projections. Reading it at the selected radius gives exactly the endpoint pair and the proof that every displayed regular-sequence endpoint lies in the interval.

Since the carrier has no coordinate for a cut, quotient, trichotomy decision, compactness cover, or host real equality, no further row can be read by the interval obligation. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface dyadic interval obligation`

Theorem 309.4 (LocatedReal RegSeqRat classifier stability)**.** *Let* L *and* L' *be accepted* LocatedReal$^{\uparrow}$ *packets whose* RegSeqRat$^{\uparrow}$ *sources are related by the dyadic-tolerance classifier and whose finite-observation schedules, interval endpoint ledgers, and enclosure rows are transported componentwise by* hsame *and the displayed* Cont *routes. Then the locatedness classifier of* L *transports to the locatedness classifier of* L' *on every displayed dyadic radius row.*

The transported classifier reads only the scheduled RegSeqRat$^{\uparrow}$ *windows, the* DyadicRatCore *endpoint ledgers, and the visible provenance rows.*

Proof. Start with a displayed radius row of L. The carrier projection gives a finite StreamName$^{\uparrow}$ window and a certified dyadic endpoint pair enclosing the selected RegSeqRat$^{\uparrow}$ observations.

Transport the regular-sequence window along the stated RegSeqRat$^{\uparrow}$ classifier row. Transport the endpoint pair and radius ledger along the componentwise hsame rows, using the displayed Cont routes to keep the finite schedule aligned.

Repack the transported endpoint pair with the transported regular-sequence observations. The same finite enclosure row now witnesses the locatedness classifier for L' at that radius, and the construction has used only rows already named in the two packets. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface regseqrat classifier stability`

Theorem 309.5 (LocatedReal RealUp seal compatibility)**.** *Every accepted* $\mathsf{LocatedReal}^{\uparrow}$ *packet presents a* $\mathsf{Real}^{\uparrow}$ *seal-compatible source surface by forgetting only the locatedness interval rows and retaining the underlying* $\mathsf{RegSeqRat}^{\uparrow}$ *real-name source,* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic-radius observations, regularity ledgers, finite-window transport rows, and provenance.*

Conversely, the locatedness rows may be attached to such a $\mathsf{Real}^{\uparrow}$ *seal source only as finite dyadic endpoint enclosures over the already displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *windows; they do not add a completed real object or an equality principle for real names.*

Proof. Use the $\mathsf{RegSeqRat}^{\uparrow}$ component of the carrier surface. It already contains the scheduled stream carrier, dyadic-radius rows, regularity ledger, classifier source rows, and finite-window transport provenance consumed at the $\mathsf{Real}^{\uparrow}$ seal boundary.

Forgetting the interval ledger and locatedness classifier leaves exactly that source surface. No seal consumer can read the forgotten rows, because the retained source has no coordinate for interval endpoints beyond the dyadic observations carried by $\mathsf{RegSeqRat}^{\uparrow}$.

For the converse attachment, choose a displayed seal source window and add only the DyadicRatCore endpoint pair and enclosure ledger required by Definition 309.2. This enriches the finite packet with locatedness data while leaving the $\mathsf{Real}^{\uparrow}$ seal source unchanged. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface realup seal compatibility`

Theorem 309.6 (LocatedReal NameCert boundary)**.** *The seed* $\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$ *obligation sketch consists of five rows:*

1. *carrier inhabitance is witnessed by a constant dyadic* $\mathsf{StreamName}^{\uparrow}$ *stream whose* $\mathsf{RegSeqRat}^{\uparrow}$ *source is the constant regular sequence and whose interval ledger uses the same endpoint on both sides;*
2. *classifier reflexivity, symmetry, and transitivity are witnessed by interval refinement of displayed dyadic endpoint pairs;*
3. *carrier transport along* hsame *repacks the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, schedule, endpoint ledger, and enclosure classifier componentwise;*
4. *ledger exactness says that every interval endpoint used by the locatedness classifier is one of the displayed* DyadicRatCore *endpoint rows;*
5. $\mathsf{Real}^{\uparrow}$ *seal compatibility is the handoff of Definition 309.5.*

The boundary claims neither Dedekind cuts, quotient reals, trichotomy, open-cover compactness, nor host real equality.

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealNameCertBoundary rows`

Proof. For inhabitance, take the constant dyadic stream. Its $\mathsf{StreamName}^{\uparrow}$ schedule reads the same dyadic endpoint at every displayed window, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity ledger is the constant-window ledger, and each located interval can use that endpoint as both sides.

For classifier laws, refine finite interval ledgers. Reflexivity keeps the same endpoint pair, symmetry swaps the two classified packets while keeping the common refinement row, and transitivity composes two finite refinements by taking the displayed common endpoint ledger supplied by DyadicRatCore.

For transport, apply componentwise hsame to the regular-sequence source, observation schedule, endpoint rows, and enclosure classifier. The transported rows satisfy the same finite carrier definition.

Ledger exactness follows by projection from Definition 309.2: every classifier endpoint is drawn from the stored DyadicRatCore interval ledger. The $\mathsf{Real}^{\uparrow}$ seal row is then exactly Definition 309.5, so the five rows form the seed boundary without adding any excluded real-number principle. □

Theorem 309.7 (LocatedReal scoped source boundary)**.** *The scoped* $\mathsf{LocatedReal}^{\uparrow}$ *boundary is exactly the four carrier components of Definition 309.2 together with the dyadic interval obligation, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier-stability row, the* $\mathsf{Real}^{\uparrow}$ *seal-compatibility row, and the* NameCert *boundary rows. All rows are read through* $\mathsf{RegSeqRat}^{\uparrow}$*,* DyadicRatCore*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

This scoped boundary contains no Dedekind cut, quotient real, trichotomy decision, open-cover compactness datum, host real equality, or standard bridge.

Proof. Project the carrier definition of Definition 309.2. It exposes exactly the $\mathsf{RegSeqRat}^{\uparrow}$ source, the dyadic interval ledger, the finite observation schedule, and the locatedness classifier.

Apply Definition 309.3 to obtain the endpoint and radius rows. Apply Definition 309.4 to transport the scheduled regular-sequence reads, and apply Definition 309.5 to identify the seal-facing source surface.

Finally apply Definition 309.6 to bind inhabitance, classifier laws, transport, ledger exactness, and seal compatibility to the same displayed provenance. Since the carrier and the cited rows have no coordinate for cuts, quotient reals, trichotomy, compactness, host equality, or a standard bridge, the scoped source boundary exports none of those principles. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface scoped source boundary`

Theorem 309.8 (LocatedReal common refinement gluing)**.** *Let two* $\mathsf{LocatedReal}^{\uparrow}$ *packets be displayed through the carrier surface of Definition 309.2. If their* $\mathsf{StreamName}^{\uparrow}$ *schedules admit a common finite observation window and their* DyadicRatCore *endpoint ledgers transport along the visible* hsame *and* Cont *rows to that window, then the two packets determine one repacked locatedness classifier row over the common window.*

The repacked row uses only the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *source data, the common schedule, the transported dyadic endpoint pairs, the enclosure classifier, and the visible* BHist*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface common refinement tail empty`
Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrier common refinement gluing`

Proof. Project both packets to the surface of Definition 309.2. The hypotheses identify a finite $\mathsf{StreamName}^{\uparrow}$ window receiving the two schedules and give the endpoint transports needed to view both dyadic interval ledgers over that window.

Use Definition 309.3 to keep the transported endpoint pairs inside the displayed DyadicRatCore interval ledger. Use Definition 309.4 to transport the regular-sequence source rows without changing the consumer-visible source surface.

Finally repack the common window, transported endpoint pairs, enclosure classifier, and provenance rows into the locatedness classifier coordinate of Definition 309.2. The repacking does not introduce any host real equality or completion datum, so it yields exactly one classifier row over the displayed common refinement. □

Theorem 309.9 (LocatedReal common-refinement consumer exactness)**.** *Let two accepted* $\mathsf{LocatedReal}^{\uparrow}$ *packets have a common finite* $\mathsf{StreamName}^{\uparrow}$ *observation window and transported* DyadicRatCore

endpoint ledgers as in Definition 309.8. Every later Met$^{\uparrow}$ *or real-order consumer read of their locatedness data factors through the repacked common-window row.*

The consumer can read only the RegSeqRat$^{\uparrow}$ *source, the schedule row, the dyadic endpoint pair, the enclosure classifier, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It cannot read quotient reals, host real equality, trichotomy, open-cover compactness, metric completion, or standard bridge data.*

Proof. Use Definition 309.8 to repack the two accepted packets over the common finite window. The resulting row has exactly the common schedule, transported endpoint pairs, classifier coordinate, and visible provenance named in the carrier surface.

Apply Definition 309.10 to the endpoint side of the repacked row. This exhausts every dyadic endpoint consumed by the locatedness classifier by projection from the displayed DyadicRatCore interval ledger.

Finally apply Definition 309.11. Its downstream projection bound says that a metric or order consumer reads only the displayed source, schedule, ledger, classifier, and provenance coordinates. Since none of those coordinates contains quotient reals, host real equality, trichotomy, compactness, completion, or standard bridge data, the consumer read is exhausted by the common-window row. □

Theorem 309.10 (LocatedReal interval ledger exactness)**.** *Every dyadic endpoint consumed by the* LocatedReal$^{\uparrow}$ *locatedness classifier is the projection of an endpoint row from the* DyadicRatCore *interval ledger displayed in Definition 309.2.*

Consequently the classifier has no hidden endpoint source beyond the stored dyadic interval ledger, the finite StreamName$^{\uparrow}$ *observation schedule, and the transported* RegSeqRat$^{\uparrow}$ *source rows.*

Proof. Unfold the carrier surface of Definition 309.2. Its locatedness classifier coordinate is attached to the displayed endpoint ledger, so each endpoint read used in a classifier row is obtained by projection from that ledger.

Apply Definition 309.3 to the projected endpoints. The theorem records that the interval rows remain dyadic and are transported through the visible hsame and Cont provenance rather than through an ambient real object.

The stability row of Definition 309.4 accounts for the regular-sequence source data used beside the endpoints. Thus every consumed endpoint is ledger-visible, and no unrecorded endpoint coordinate is available to the classifier. □

Theorem 309.11 (LocatedReal metric consumer handoff)**.** *Any later* Met$^{\uparrow}$ *or real-order consumer of a* LocatedReal$^{\uparrow}$ *packet may read only the* RegSeqRat$^{\uparrow}$ *source, the* StreamName$^{\uparrow}$ *schedule, the dyadic endpoint and radius rows, the enclosure classifier, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Such a consumer cannot read a quotient-real carrier, host real equality, trichotomy decision, open-cover compactness datum, or metric-completion object from the LocatedReal$^{\uparrow}$ *certificate surface.*

```
Lean4 ✓  BEDC.Derived.LocatedRealUp.LocatedRealCarrier metric consumer empty boundary
Lean4 ✓  BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface metric consumer handoff
Lean4 ✓  BEDC.Derived.LocatedRealUp.LocatedRealCarrier metric consumer handoff
```

Proof. Begin with the source projection supplied by Definition 309.5. It identifies the Real$^{\uparrow}$ seal-facing part of the packet with the displayed RegSeqRat$^{\uparrow}$ source, finite schedule, dyadic radius data, and classifier source rows.

Next use Definition 309.10 for endpoint reads and Definition 309.8 for finite-window repacking. These two rows show that all interval and classifier data consumed downstream are already present in the carrier surface and remain stable under common refinements.

The handoff is therefore the projection of the named finite coordinates from Definition 309.2. Since that surface has no coordinate for quotient reals, host equality, trichotomy, compactness, or completion, no later metric or order consumer can extract such data from the certificate. □

Theorem 309.12 (LocatedReal scoped RegSeqRat dyadic window)**.** *Every scoped locatedness read from a* $\mathsf{LocatedReal}^{\uparrow}$ *packet factors through a displayed finite window: a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source window, a* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the observations, and a* DyadicRatCore *endpoint ledger naming the dyadic enclosure endpoints. The factorization carries the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance used by the packet.*

Thus the scoped read exposes only finite regular-sequence observations, schedule entries, dyadic endpoint rows, enclosure ledgers, and their named provenance.

Proof. Project the carrier surface of Definition 309.2. It gives the $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ schedule, DyadicRatCore interval ledger, and locatedness classifier as displayed coordinates.

Apply Definition 309.4 to keep the scheduled regular-sequence reads stable under the visible hsame and Cont transports. Apply Definition 309.10 to identify every consumed endpoint with a stored dyadic ledger row.

Finally use Definition 309.8 to repack the finite window when the consumer asks for a common refinement, and use Definition 309.5 to keep the seal-facing source equal to the same regular rational surface. No coordinate outside the named carrier surface is introduced. □

Theorem 309.13 (LocatedReal scoped carrier transport)**.** *Let L be an accepted* $\mathsf{LocatedReal}^{\uparrow}$ *packet. If its* $\mathsf{RegSeqRat}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* DyadicRatCore *endpoint ledger, and located enclosure row are transported componentwise by the displayed* hsame *and* Cont *routes, then the transported rows repack as an accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carrier.*

The repacking uses only the transported finite rows and their BHist*,* Pkg*, and* NameCert *provenance; it does not use quotient-real data, host real equality, or an ambient completion object.*

Proof. Unpack L through Definition 309.2. The four carrier coordinates are the regular-sequence source, schedule, endpoint ledger, and locatedness classifier.

Transport the source and classifier by Definition 309.4. Transport the endpoint reads by Definition 309.10, and use Definition 309.8 when the transported schedule is read over a common finite window.

The transported rows satisfy the same carrier definition, so they repack as a $\mathsf{LocatedReal}^{\uparrow}$ packet. The $\mathsf{Real}^{\uparrow}$ seal compatibility of Definition 309.5 shows that the repacked source surface is still finite regular rational data rather than a quotient real or host equality principle. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface scope carrier transport`

Theorem 309.14 (LocatedReal scoped RealUp seal consumer)**.** *A* $\mathsf{Real}^{\uparrow}$ *seal consumer of a scoped* $\mathsf{LocatedReal}^{\uparrow}$ *packet may read exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *source surface together with the displayed located interval rows. Concretely, it may read the scheduled regular rational windows, dyadic radius and endpoint ledgers, enclosure classifier rows, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

It may not read a Dedekind cut, trichotomy decision, open-cover compactness witness, completion object, quotient-real carrier, or host real equality from the $\mathsf{LocatedReal}^{\uparrow}$ *seal boundary.*

Proof. Apply Definition 309.5. It identifies the seal-facing surface with the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, finite schedule, dyadic-radius observations, regularity ledgers, and transport provenance.

The additional located rows are bounded by Definition 309.10: every endpoint consumed by the classifier is drawn from the displayed DyadicRatCore ledger. The common-refinement row of Definition 309.8 keeps later reads inside the same finite window surface.

Together with Definition 309.4 and Definition 309.2, these rows exhaust the seal consumer boundary. Since none of the cited surfaces contains cuts, trichotomy, compactness, completion, quotient carriers, or host real equality, the consumer cannot read such data. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealCarrierSurface scope real seal consumer`

Theorem 309.15 (LocatedReal public RegSeqRat interval handoff)**.** *Every public consumer of* $\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$ *receives a single finite package consisting of the* $\mathsf{RegSeqRat}^{\uparrow}$ *source window, the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* DyadicRatCore *endpoint ledger, the locatedness enclosure row, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

No Dedekind cut, quotient real, trichotomy decision, open-cover compactness witness, host real equality, or completion object is exported by this public handoff.

Proof. Use Definition 309.12 to obtain the finite regular-sequence source window, finite schedule row, and dyadic endpoint ledger. The endpoint visibility needed by a public consumer is exactly the ledger projection of Definition 309.10.

Next apply Definition 309.13. It shows that any componentwise transport of the source, schedule, endpoint ledger, and located enclosure rows remains inside the accepted $\mathsf{LocatedReal}^{\uparrow}$ carrier surface, with only displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Finally apply Definition 309.14. The $\mathsf{Real}^{\uparrow}$ seal-facing consumer reads the same finite source and located interval rows, so the public package is the stated single handoff package. The excluded objects have no coordinate in the cited carrier, ledger, transport, or seal-consumer surfaces, hence they are not exported. □

Theorem 309.16 (LocatedReal standard bridge boundary)**.** *A* $\mathsf{LocatedReal}^{\uparrow}$ *packet whose* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* DyadicRatCore *interval endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *observation schedule,* $\mathsf{Real}^{\uparrow}$ *seal row, metric handoff row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, and classifier-transport rows are exactly the public rows displayed in Definition 309.15 admits the standard bridge surface used by downstream* $\mathsf{LocatedReal}^{\uparrow}$ *consumers.*

Conversely, any such standard bridge packet repacks to the public $\mathsf{LocatedReal}^{\uparrow}$ *boundary without adding quotient reals, trichotomy, open-cover compactness, host real equality, or completion data.*

Proof. First apply Definition 309.6. It binds the carrier, classifier transport, interval obligation, seal compatibility, and certificate provenance to the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 309.11. The metric and order consumer surface can read only the regular rational source, schedule, dyadic endpoint and radius rows, enclosure classifier, and visible provenance. Then apply Definition 309.15 to package those rows with the $\mathsf{Real}^{\uparrow}$ seal handoff as the public boundary.

The reverse direction forgets the bridge wrapper and keeps the same finite rows. The exclusions recorded in the closure status show that quotient reals, trichotomy, open-cover compactness, host real equality, and completion data have no coordinate in the bridge surface, so no off-table object enters during either packing. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$, maturePackageClosed)

Constructive story: A $\mathsf{LocatedReal}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ regular rational source window, a $\mathsf{StreamName}^{\uparrow}$ observation schedule, DyadicRatCore endpoint and radius rows, a locatedness enclosure row, $\mathsf{Real}^{\uparrow}$ seal provenance, visible BHist, hsame, Cont, and Pkg routes, and local NameCert rows.

Scope closed: The closed scope is the finite $\mathsf{RegSeqRat}^{\uparrow}$/$\mathsf{StreamName}^{\uparrow}$/DyadicRatCore/$\mathsf{Real}^{\uparrow}$ route through BHist, hsame, Cont, Pkg, and NameCert, covering the carrier surface, dyadic interval obligation, $\mathsf{RegSeqRat}^{\uparrow}$ classifier stability, $\mathsf{Real}^{\uparrow}$ seal compatibility, NameCert boundary, common-refinement gluing, metric consumer handoff, scoped carrier transport, public $\mathsf{RegSeqRat}^{\uparrow}$ interval handoff, and standard bridge boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LocatedRealUp.LocatedRealCarrier\_realup\_regseqrat\_boundary`

Bridge status: paperBridge

Origin: human

Not claimed: The chapter does not close Dedekind cuts, quotient reals, trichotomy, open-cover compactness, host real equality, metric completion, or a checked standard bridge.

Upgrade path: The remaining formal work is bridgeChecked verification for the same finite locatedness route, with no further theory-closure upgrade.

Chapter 310

A Concrete Naming Certificate for $\mathsf{SemidefiniteCone}^{\uparrow}$

The $\mathsf{SemidefiniteCone}^{\uparrow}$ interface names the finite certificate surface used by semidefinite programming over symmetric matrix data. In the external optimization presentation of Boyd–Vandenberghe 2004, the semidefinite cone is read through symmetric matrices whose quadratic form is nonnegative on every vector. The BEDC packet below keeps only the displayed finite rows required to name that surface: matrix rows, vector-space rows, convex-combination ledgers, bilinear-form pairings, order witnesses, and package provenance.

The symmetric-matrix row is read through chapter 49, the vector row through chapter 47, and the convex ledger through chapter 208. Thus $\mathsf{SemidefiniteCone}^{\uparrow}$ is a consumer of those sibling carriers rather than an isolated kernel packet: it binds their displayed matrix, vector, and convex-combination surfaces into the cone certificate while the bilinear-form and order rows remain explicit finite BEDC evidence.

310.1 Carrier packet

Definition 310.1 (SemidefiniteCone carrier packet)**.** *A* $\mathsf{SemidefiniteCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(M, V, C, B, N, P)$$

with the following rows:

1. *M is a symmetric-matrix row read through* $\mathsf{Mat}^{\uparrow}$*;*
2. *V is a vector-space row read through* $\mathsf{VecSpace}^{\uparrow}$ *over the same scalar boundary as M;*
3. *C is a convex-combination ledger from* $\mathsf{ConvexSet}^{\uparrow}$ *for displayed matrix rows;*
4. *B is a bilinear-form pairing row from* $\mathsf{BilinForm}^{\uparrow}$ *that reads $v^T M v$ as a scalar row for displayed vectors v;*
5. *N is an order and nonnegativity witness row saying that every vector row explicitly carried by the packet has nonnegative paired scalar value through* $\mathsf{Order}^{\uparrow}$*;*
6. *P is the* Pkg *provenance tying these rows to the cited* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{BilinForm}^{\uparrow}$*, and* $\mathsf{Order}^{\uparrow}$ *certificate boundaries.*

The carrier predicate is SDConePacket(M, V, C, B, N, P). *Two accepted packets classify together only when their matrix, vector-space, convex-ledger, bilinear-pairing, nonnegativity, and provenance rows transport by displayed* hsame *and by the same* Cont *routes named in* P.

Theorem 310.2 (SemidefiniteCone carrier stability)**.** *Let* $D = (M, V, C, B, N, P)$ *be an accepted* SemidefiniteCone$^{\uparrow}$ *carrier and let* $D' = (M', V', C', B', N', P')$ *be obtained by displayed* hsame *transport of all six rows, with the same* Cont *routes through the cited certificate boundaries. Then* D' *is again an accepted carrier, and its classifier reads exactly the transported matrix, vector, convex, bilinear, nonnegativity, and provenance rows.*

Lean4 ✓ `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket carrier stability`

Proof. Project the six coordinates of Definition 310.1. The matrix and vector-space coordinates are ordinary displayed rows, so hsame transport gives the rows M' and V' with the same scalar boundary.

Next transport the convex-combination ledger and the bilinear-form row. The stated Cont routes identify the same matrix and vector slots on the transported packet, so the pairing row remains readable as the displayed scalar value attached to M' and V'.

Finally transport the order/nonnegativity witness and the package row. Repacking these transported rows gives SDConePacket(M', V', C', B', N', P'), and the classifier has no coordinate other than the transported finite rows. □

Theorem 310.3 (SemidefiniteCone cone-operation ledger)**.** *For accepted* SemidefiniteCone$^{\uparrow}$ *carriers* D_0 *and* D_1 *over a common transported vector-space and scalar boundary, the cone-operation surface consists of finite rows for displayed matrix addition, displayed nonnegative scalar multiplication, the convex-combination ledger inherited from* ConvexSet$^{\uparrow}$, *the bilinear pairing rows inherited from* BilinForm$^{\uparrow}$, *the order witness rows inherited from* Order$^{\uparrow}$, *and* Pkg *provenance tying the operation packet to* D_0 *and* D_1.

The operation ledger exports no spectral decomposition, no eigenvalue existence claim, no analytic closure of the cone, and no positivity predicate from a host matrix library.

Lean4 ✓ `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket cone operation ledger`

Proof. Read the two input carriers through Definition 310.1. Their matrix rows and common vector-space row supply the finite endpoints of the operation packet.

For addition, use the displayed Mat$^{\uparrow}$ addition row and the ConvexSet$^{\uparrow}$ ledger that records the resulting matrix row as a convex or conic combination with visible coefficients. For nonnegative scalar multiplication, use the scalar row together with the Order$^{\uparrow}$ witness that the coefficient is nonnegative.

The bilinear-form row then reads each displayed vector test against the resulting matrix row. Because each nonnegativity assertion is an explicit order row and all dependencies are named in Pkg, the ledger is exhausted by finite BEDC rows and does not appeal to spectral or analytic facts. □

Theorem 310.4 (SemidefiniteCone dual-pairing boundary)**.** *Every accepted* SemidefiniteCone$^{\uparrow}$ *packet exposes its dual-facing read only through the displayed* BilinForm$^{\uparrow}$ *pairing row and the transported* Order$^{\uparrow}$ *nonnegativity witness. A consumer may compare a matrix row with a displayed vector or test row through the finite scalar value named by the packet,* hsame *transport,* Cont *routes, and* Pkg *provenance.*

The packet does not assert self-duality of the cone, closedness under analytic limits, or equality with any host-space positive-semidefinite predicate.

Lean4 ✓ `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket dual pairing empty boundary`

Proof. Project the bilinear-form row B and the nonnegativity row N from the carrier. These are the only coordinates that connect a matrix row to scalar comparison data.

A dual-facing consumer therefore receives a scalar pairing value only after following the displayed $\mathsf{BilinForm}^{\uparrow}$ row and the package routes named by P. The comparison of that scalar value with zero is an $\mathsf{Order}^{\uparrow}$ row, transported by hsame when the underlying matrix or vector row is replaced.

No coordinate in the carrier contains a host dual cone, a topological closure assertion, or a spectral certificate. Thus the boundary is exactly the finite pairing-and-order surface stated in the theorem. □

Theorem 310.5 (SemidefiniteCone NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet of Definition 310.1;*
2. *classifier and stability: the displayed transport statement of Definition 310.2;*
3. *cone-operation ledger: the addition and nonnegative scalar-multiplication rows of Definition 310.3;*
4. *dual-pairing boundary: the bilinear pairing and order rows of Definition 310.4;*
5. *provenance: the* Pkg *row tying the packet to* $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, *and* $\mathsf{Order}^{\uparrow}$.

These rows are the full public seed surface for the semidefinite-cone name.

Proof. The carrier row supplies the displayed matrix, vector-space, convex-ledger, bilinear-pairing, order, and provenance coordinates. This fixes the source domain of the name.

The stability row supplies the classifier behavior under hsame and Cont transport. The cone-operation row records the only operation reads available to downstream optimization packets, and the dual-pairing boundary records the only scalar comparison reads.

The provenance row names every dependency certificate used by the packet. Since the construction withholds spectral decomposition, eigenvalue existence, analytic closure, and host matrix positivity, the listed rows exhaust the $\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}$ obligation surface. □

Lean4 ✓ `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket namecert obligation surface`

Theorem 310.6 (SemidefiniteCone scoped provenance package)**.** *Every accepted* $\mathsf{SemidefiniteCone}^{\uparrow}$ *packet exposes its public scope only through* $\mathsf{Mat}^{\uparrow}$ *symmetric-matrix rows,* $\mathsf{VecSpace}^{\uparrow}$ *vector rows,* $\mathsf{ConvexSet}^{\uparrow}$ *cone-combination rows,* $\mathsf{BilinForm}^{\uparrow}$ *pairing rows,* $\mathsf{Order}^{\uparrow}$ *nonnegativity rows, and* Pkg *provenance, transported by displayed* hsame *and* Cont *routes. The scoped package exports no spectral decomposition, eigenvalue existence theorem, analytic closure of the cone, self-duality assertion, host matrix positivity predicate, or standard bridge data.*

Lean4 ✓ `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket scoped provenance package`

Proof. Project the carrier of Definition 310.1. Its six coordinates are the symmetric-matrix row, vector-space row, convex-combination ledger, bilinear-pairing row, order/nonnegativity witness, and package provenance, so the carrier has no dependency outside the displayed BEDC rows.

Use Definition 310.2 for classifier transport. The only permitted motion of an accepted packet is componentwise hsame transport along the same Cont routes named by the package row.

Use Definition 310.3 for the operation surface and Definition 310.4 for the scalar-facing boundary. These two rows consume precisely the displayed cone-operation, pairing, and order dependencies, then return to the same finite packet surface.

Finally apply Definition 310.5. The NameCert surface has no remaining coordinate from which a consumer could read spectral decomposition, eigenvalue existence, analytic closure, self-duality, host matrix positivity, or standard bridge data, so the scoped package is exactly the listed BEDC provenance boundary. □

Theorem 310.7 (SemidefiniteCone public packet certificate)**.** *Whenever the carrier stability, cone-operation ledger, dual-pairing boundary,* $\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}$ *obligation surface, and scoped provenance package rows of Definition 310.2, Definition 310.3, Definition 310.4, Definition 310.5, and Definition 310.6 hold, the displayed* BHist *matrix row, vector-space row, convex-ledger row, bilinear-pairing row, order row, provenance row, and package row form the exported public* $\mathsf{SemidefiniteCone}^{\uparrow}$ *packet. This public packet is the consumer-facing certificate surface for the semidefinite-cone name, and it exports no row beyond the ten-field roadmap entries already supplied by those finite BEDC dependencies.*

Proof. Use Definition 310.2 to fix the transported matrix, vector-space, convex-ledger, bilinear-pairing, order, and provenance coordinates of the accepted packet.

Use Definition 310.3 and Definition 310.4 to expose the only operation and scalar-comparison rows available to a consumer. These two rows are the cone-operation and dual-facing entries of the public packet.

Finally apply Definition 310.5 and Definition 310.6. Together they identify the carrier, classifier, stability, ledger, exactness, provenance, scope, absence, package, and bridge-boundary slots of the ten-field roadmap template, so the public packet is exactly their conjunction over the displayed BEDC rows. □

Theorem 310.8 (SemidefiniteCone standard bridge boundary)**.** *Any standard bridge for the public* $\mathsf{SemidefiniteCone}^{\uparrow}$ *packet must read and write exactly the displayed symmetric* $\mathsf{Mat}^{\uparrow}$ *row,* $\mathsf{VecSpace}^{\uparrow}$ *row,* $\mathsf{ConvexSet}^{\uparrow}$ *cone-operation ledger,* $\mathsf{BilinForm}^{\uparrow}$ *pairing row,* $\mathsf{Order}^{\uparrow}$ *nonnegativity witnesses,* hsame *and* Cont *transport rows, and* Pkg *provenance of Definition 310.7. The bridge boundary exports no spectral decomposition, eigenvalue existence, analytic closure, self-duality theorem, or host matrix positivity predicate.*

Proof. Project the public packet theorem. Its carrier coordinates are the symmetric matrix row, compatible vector-space row, convex cone-operation ledger, bilinear-pairing row, order nonnegativity witnesses, transport rows, and provenance rows.

The standard-facing bridge may therefore read those coordinates and may write back only the same rows after hsame and Cont transport. This is the full boundary because the public packet certificate has no additional output slot.

Spectral decomposition, eigenvalue existence, analytic closure, self-duality, and host matrix positivity are not among the projected rows. They cannot be exported by a bridge whose domain and codomain are restricted to the displayed public packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}$**.**

Constructive story: A semidefinite-cone packet is generated by a symmetric $\mathsf{Mat}^{\uparrow}$ row, a compatible $\mathsf{VecSpace}^{\uparrow}$ row, a $\mathsf{ConvexSet}^{\uparrow}$ cone-operation ledger, a $\mathsf{BilinForm}^{\uparrow}$ pairing row, displayed $\mathsf{Order}^{\uparrow}$ nonnegativity witnesses, and Pkg provenance for those finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 310.7, Definition 310.8, Definition 310.3, and Definition 310.4 bind the public packet certificate, standard bridge boundary, cone-operation ledger, and dual-pairing boundary as the mature packet surface over the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, hsame, Cont, and Pkg rows visible to $\mathsf{SemidefiniteCone}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SemidefiniteConeUp.SemidefiniteConePacket\_dual\_pairing\_empty\_boundary`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close spectral decomposition, eigenvalue existence, analytic closure, self-duality, host matrix positivity, or a machine-checked standard bridge.

Upgrade path: No further paper-side theory grade remains; formal bridge verification remains on the verification axis.

Chapter 311

A Concrete Naming Certificate for $\mathsf{DyadicBall}^{\uparrow}$

Following the Bishop–Bridges constructive-analysis treatment of rational balls, the $\mathsf{DyadicBall}^{\uparrow}$ interface names a finite center-radius enclosure packet for L10 real-name observations. It sits between the dyadic arithmetic surface of chapter 291, the regular rational stream surface of chapter 301, and the real-name seal surface of chapter 38.

311.1 Finite center-radius carrier

Definition 311.1 (DyadicBall finite enclosure carrier)**.** *A* $\mathsf{DyadicBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (c, r, k, o, \eta, H, C, P, N)$$

with the following visible rows:

1. *c is a center row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *r is a positive dyadic radius row, displayed by the same dyadic arithmetic surface and readable as a one-sided* $\mathsf{DyadicInterval}^{\uparrow}$ *tolerance window;*
3. *k is a precision and schedule row supplied by* $\mathsf{StreamName}^{\uparrow}$*, with the requested tolerance coordinated by* $\mathsf{CauchyModulus}^{\uparrow}$ *when a regular source provides an explicit rate;*
4. *o is the scheduled observation row, either a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window endpoint or the finite source read of a* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *η is the containment ledger proving that the displayed observation lies within the radius row around the center row;*
6. *H and C are the componentwise* hsame *witnesses and displayed* Cont *routes for the center, radius, schedule, observation, and containment rows;*
7. *P is the* Pkg *provenance naming the consumed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source surfaces;*
8. *N is the local* NameCert *row recording that classification is by componentwise finite transport of the displayed rows.*

The carrier has no quotient ball space, no host real equality, no completed metric object, and no hidden real-radius coordinate. Its classifier compares two accepted packets only by transporting the listed center, radius, schedule, observation, containment, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallTasteGate single carrier alignment`

Theorem 311.2 (DyadicBall classifier laws)**.** *For accepted* $\mathsf{DyadicBall}^{\uparrow}$ *carriers* $B = (c, r, k, o, \eta, H, C, P, N)$ *and* $B' = (c', r', k', o', \eta', H', C', P', N')$*, the componentwise classifier on the center, radius, schedule, observation, containment,* hsame, Cont, Pkg, *and* NameCert *rows is reflexive, symmetric, and transitive. Whenever it relates* B *to* B'*, the displayed finite routes transport carrier acceptance from* B *to* B'*.*

These laws act only on the rows exposed by Definition 311.1. They introduce no quotient ball, host real equality, hidden metric point, or completed real.

↪variant `BEDC.Derived.DyadicBallUp.DyadicBallPacket classifier transport`
Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallPacket classifier laws`

Proof. Project the accepted packets through Definition 311.1. The only rows available to the classifier are the center, radius, schedule, observation, containment, routing, provenance, and naming rows.

Reflexivity uses the componentwise reflexive hsame witness on each visible row and the displayed identity Cont routes. Symmetry reverses the same componentwise witnesses and routes. Transitivity composes the center, radius, schedule, observation, containment, hsame, Cont, Pkg, and NameCert comparisons row by row.

For carrier transport, move the accepted rows of B along the classifier routes and keep the stated containment ledger in the target packet. Repacking the transported rows gives the accepted carrier B'. Since the repacking uses only the finite rows of the carrier, it cannot create a quotient ball, host real equality, hidden metric point, or completed real. □

Theorem 311.3 (DyadicBall radius refinement closure)**.** *Let* $B = (c, r, k, o, \eta, H, C, P, N)$ *be an accepted* $\mathsf{DyadicBall}^{\uparrow}$ *carrier, and let* r' *be a positive dyadic radius row whose displayed comparison ledger proves* $r' \leq r$*. If the same center row* c*, schedule row* k*, observation row* o*, and containment ledger are transported through finite* Cont *rows so that the observation remains inside the smaller displayed radius, then the repacked tuple*

$$B' = (c, r', k, o, \eta', H', C', P', N')$$

is again an accepted $\mathsf{DyadicBall}^{\uparrow}$ *carrier.*

The refinement changes only the visible radius and containment rows. It does not form a quotient of balls and does not compare centers by host real equality.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteCarrier radius refinement closure`

Proof. Project the carrier of Definition 311.1. The center, radius, schedule, observation, containment, routing, provenance, and naming coordinates are all explicit finite BHist rows.

Use the displayed comparison $r' \leq r$ to replace the radius coordinate. The extra hypothesis supplies the finite containment ledger η' for the same center and observation at the smaller radius.

Transport the remaining coordinates by the componentwise hsame witnesses and Cont routes. Repacking these rows gives the accepted packet B', and the construction has no coordinate from which a quotient ball space or host real equality could be read. □

Theorem 311.4 (DyadicBall RegSeqRat window handoff)**.** *Every accepted* DyadicBall$^{\uparrow}$ *packet supplies exactly the finite dyadic rows that a scheduled* RegSeqRat$^{\uparrow}$ *window may consume: the* StreamName$^{\uparrow}$ *precision row, the* DyadicRatCore$^{\uparrow}$ *center row, the positive dyadic radius row, the scheduled endpoint row, the containment ledger, and the associated* hsame, Cont, Pkg, *and* NameCert *provenance.*

The handoff may be read as a dyadic endpoint-radius window for the regular stream, but it exports no quotient stream, no completion object, and no external real value.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallPacket regseqrat window handoff`
↪ variant `BEDC.Derived.DyadicBallUp.DyadicBallFiniteEnclosure regseqrat window handoff`

Proof. Read the packet through Definition 311.1. The schedule row is already a StreamName$^{\uparrow}$ finite request, and the center and radius rows are accepted dyadic rows.

For a RegSeqRat$^{\uparrow}$ consumer, keep exactly the scheduled observation row and the containment ledger that compares it to the displayed center and radius. These are the endpoint and tolerance rows consumed by a finite regular-stream window.

The componentwise hsame, Cont, Pkg, and NameCert rows are the same rows listed in the carrier. Since the carrier contains no stream quotient or completed limit, the handoff cannot export one. □

Theorem 311.5 (DyadicBall finite-window packet RegSeqRat handoff)**.** *A finite-window* DyadicBall$^{\uparrow}$ *packet with* BHist *rows* $(center, radius, schedule, observation, containment, route, provenance, certRow, handoff, sealBoundary)$ *exposes exactly the* RegSeqRat$^{\uparrow}$ *handoff rows: unary center, radius, schedule, observation, containment, route, and handoff rows; the append-readback rows* hsame *containment* $(append\ center\ radius)$, hsame *route* $(append\ schedule\ observation)$, *and* hsame *handoff* $(append\ containment\ route)$*; and the package row* PkgSig$(bundle, handoff, pkg)$.

The finite-window packet exports no quotient stream, completed real point, host real equality, or external ball object.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket regseqrat window handoff`

Proof. Project the finite-window packet rows. The unary rows for center, radius, schedule, and observation are stored directly, and the displayed Cont routes build the containment, route, and handoff rows over those sources.

The three displayed Cont routes are precisely the append readbacks: containment reads append$(center, radius)$, route reads append$(schedule, observation)$, and handoff reads append$(containment, route)$. Keeping the package row PkgSig$(bundle, handoff, pkg)$ gives the full regular-rational handoff surface.

The remaining packet coordinates are provenance, certificate, and seal-boundary rows. They certify and delimit the finite packet, but they do not add a quotient stream, a completed real point, a host real equality, or an external ball object. □

Theorem 311.6 (DyadicBall Real seal boundary)**.** *A* Real$^{\uparrow}$ *seal reading an accepted* DyadicBall$^{\uparrow}$ *packet may inspect only the center row, the positive radius row, the scheduled observation row, the containment ledger, the componentwise* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

The seal boundary does not add a real-equality classifier, a quotient of regular streams, a completeness witness, or a host real ball. It merely records that the displayed finite observation lies in the displayed dyadic center-radius enclosure.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket real seal boundary`

Proof. Start with the public coordinates of Definition 311.1. These coordinates already include the finite source read that a $\mathsf{Real}^{\uparrow}$ seal may consume: center, radius, schedule, observation, containment, routing, provenance, and naming data.

Apply the handoff statement of Definition 311.4 when the seal source is read through a regular-rational scheduled window. The same finite dyadic rows are then the only source rows available to the real-name seal.

No carrier coordinate stores a real quotient, a completed metric point, or a host ball. Therefore the seal boundary is exactly the listed finite row surface. □

Theorem 311.7 (DyadicBall intersection refinement handoff)**.** *Let an accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packet be read through the finite center-radius carrier of Definition 311.1. If the packet is refined to a smaller positive dyadic radius by Definition 311.3, then the refined packet exports the same scheduled* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite-boundary rows as the original packet, except for the transported radius row and the containment ledger supplied for that smaller radius.*

Concretely, the BHist *center, schedule, scheduled observation, componentwise* hsame *transports, displayed* Cont *windows,* Pkg *provenance, and local* NameCert *rows remain the same rows of the carrier. The* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint-radius window of Definition 311.4 and the* $\mathsf{Real}^{\uparrow}$ *seal surface of Definition 311.6 are read from those rows after replacing only the radius and containment coordinates. The handoff does not introduce quotient balls, host real equality, completed metric points, or selected limits.*

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket intersection refinement handoff`

Proof. Unpack the carrier of Definition 311.1. Its public rows are the finite BHist center, positive radius, schedule, scheduled observation, containment ledger, hsame transports, Cont windows, Pkg provenance, and NameCert certificate.

Apply Definition 311.3. The theorem repacks the accepted packet at the smaller displayed dyadic radius, using the supplied containment ledger for that radius and leaving the center, schedule, observation, transport, window, provenance, and certificate rows in place.

Project the repacked packet through Definition 311.4 and Definition 311.6. Both projections inspect only the carrier rows just listed, so their finite consumer surfaces agree with the original packet except at the transported radius and containment entries.

Since no quotient-ball coordinate, host real equality, completed metric point, or selected limit occurs in the carrier, none is created by the repacking or by either outgoing handoff. □

Theorem 311.8 (DyadicBall NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$ *obligation surface consists of five finite requirements:*

1. *carrier: the finite center-radius packet of Definition 311.1;*
2. *classifier: the componentwise classifier laws of Definition 311.2;*
3. *radius-refinement stability: Definition 311.3 repacks a smaller displayed positive radius when the containment ledger is supplied for that radius;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: Definition 311.4 exposes exactly the finite endpoint-radius rows consumed by a scheduled regular-rational window;*
5. $\mathsf{Real}^{\uparrow}$ *seal boundary: Definition 311.6 restricts real-name reads to the center, radius, observation, containment, routing, package, and naming rows.*

These requirements are the full public obligation surface for $\mathsf{DyadicBall}^{\uparrow}$*.*

Proof. The carrier requirement is Definition 311.1. It lists every coordinate that can appear in an accepted packet.

The classifier requirement is Definition 311.2. It records reflexivity, symmetry, transitivity, and carrier transport for exactly the componentwise rows named by the carrier.

Radius-refinement stability, regular-window handoff, and real-seal boundary are the three theorem rows just stated. Together they cover the local ways a dyadic ball packet is refined, consumed by a regular stream window, and read by a real-name seal. □

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallPacket semantic name certificate`

Theorem 311.9 (DyadicBall obligation closure)**.** *The carrier, classifier, radius-refinement stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal boundary listed in Definition 311.8 exhaust the local obligation set for* $\mathsf{DyadicBall}^{\uparrow}$*.*

Proof. The carrier row is the finite center-radius packet of Definition 311.1. Its classifier row compares exactly the center, radius, schedule, scheduled observation, containment ledger, Cont routes, Pkg rows, and NameCert rows named by that carrier.

The three outgoing obligation rows are the radius-refinement stability theorem of Definition 311.3, the scheduled regular-rational handoff of Definition 311.4, and the real-seal boundary of Definition 311.6. These are the only local operations that refine, export, or seal a displayed dyadic-ball packet.

No quotient ball space, host real equality, metric topology, or completeness coordinate is present in the carrier. Hence there is no additional local obligation row beyond the five rows enumerated in Definition 311.8. □

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteCarrier obligation closure`

Theorem 311.10 (DyadicBall center-radius public interface)**.** *The carrier, classifier laws, radius-refinement closure,* $\mathsf{RegSeqRat}^{\uparrow}$ *window handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and obligation-closure theorem assemble the public* $\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$ *interface over the displayed center, positive radius, schedule, observation, containment,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers may cite this single public interface instead of reopening the scoped center-radius packet.*

Proof. Assemble the public certificate by projecting the scoped packet into the rows listed by Definition 311.1. The exported object has only the center, positive radius, schedule, observation, containment, transport, provenance, and naming coordinates already present in that packet.

The classifier row is supplied by Definition 311.2. Its hsame stability transports accepted packets across the displayed componentwise rows, and the associated Cont routes are precisely the routes allowed to downstream consumers.

The remaining certificate fields are the refinement, regular-window, and real-seal rows. Apply Definition 311.3 for radius ledger transport, Definition 311.4 for the $\mathsf{RegSeqRat}^{\uparrow}$ consumer row, and Definition 311.6 for the $\mathsf{Real}^{\uparrow}$ seal boundary.

Finally use Definition 311.9 to see that these rows exhaust the local obligation surface. Repacking them yields the public $\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$ interface, with no quotient ball, host real equality, metric topology, or extra completeness coordinate. □

Theorem 311.11 (DyadicBall common observation overlap)**.** *Let* $B = (c, r, k, o, \eta, H, C, P, N)$ *and* $B' = (c', r', k, o, \eta', H', C', P', N')$ *be accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packets with the same scheduled observation row* o *and the same schedule row* k*. Suppose the displayed center and radius rows*

admit the componentwise hsame *and* Cont *transports required by Definition 311.2, and suppose a finite dyadic positive radius row ρ is supplied with containment ledgers into both packets. Then the common observation determines a single finite* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer package whose visible rows are exactly o, k, the transported center rows, the transported radius row ρ, the two containment ledgers, the displayed* hsame *rows, the visible* Cont *routes, the* Pkg *provenance rows, and the two* NameCert *rows.*

This common package exports no quotient ball, host real equality, metric completeness claim, or selected real point.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket common observation overlap`

Proof. Project both accepted packets through the carrier of Definition 311.1. The two projections expose only their displayed centers, radii, common schedule row, common scheduled observation, containment ledgers, hsame rows, Cont routes, package provenance, and local naming rows.

Use Definition 311.3 on each side with the finite radius row ρ and its displayed containment ledger. This repacks both packets at the same radius row while retaining only the transported center, schedule, observation, containment, transport, package, and naming coordinates.

The shared schedule and observation rows now feed Definition 311.4. That handoff identifies one finite regular-rational window surface, because the visible scheduled rows are the same and the remaining differences have already been transported into the displayed center, radius, and containment ledgers.

Apply Definition 311.6 to the same repacked rows. The $\mathsf{Real}^{\uparrow}$ seal can read only the finite carrier coordinates named above, so it cannot distinguish the two packets by any quotient ball, host real equality, metric-completeness datum, or selected point.

Finally cite Definition 311.10 to package the common finite surface as the public $\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$ consumer interface. The resulting $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ package has exactly the rows listed in the statement. □

Theorem 311.12 (DyadicBall refinement overlap transitivity)**.** *Let $B_0 = (c_0, r_0, k, o, \eta_0, H_0, C_0, P_0, N_0)$, $B_1 = (c_1, r_1, k, o, \eta_1, H_1, C_1, P_1, N_1)$, and $B_2 = (c_2, r_2, k, o, \eta_2, H_2, C_2, P_2, N_2)$ be accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packets with the same schedule row k and the same scheduled observation row o. Suppose B_0 and B_1 admit a common refined positive dyadic radius row ρ with displayed containment ledgers into both packets, and suppose B_1 and B_2 admit a common refined positive dyadic radius row σ with displayed containment ledgers into both packets.*

If a positive dyadic radius row τ is displayed together with refinement ledgers from τ to both ρ and σ, and with containment ledgers from τ into all three packets, then the three packets determine one finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer package. Its visible rows are exactly the shared schedule k, the shared observation o, the transported center rows, the positive radius row τ, the three containment ledgers, the displayed* hsame *rows, the visible* Cont *routes, the* Pkg *provenance rows, and the three* NameCert *rows.*

The package introduces no quotient ball, host real equality, completeness datum, or selected real point.

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket refinement overlap transitivity`

Proof. Project the three accepted packets through Definition 311.1. This exposes only center rows, radius rows, the shared schedule, the shared observation, containment ledgers, transport rows, package provenance, and local naming rows.

Use Definition 311.3 separately for B_0, B_1, and B_2 at the displayed radius row τ. The refinement ledgers from τ to ρ and σ, together with the containment ledgers into the three packets, supply the radius and containment coordinates required for each repacked packet.

Apply Definition 311.11 to the pair (B_0, B_1) using the row ρ, and again to the pair (B_1, B_2) using the row σ. The two overlap packages have the same schedule and scheduled observation, and their middle packet rows are transported through the same carrier coordinates.

Now read both overlap packages through Definition 311.4. The handoff identifies the single finite regular-rational consumer surface carried by the shared schedule, shared observation, transported center rows, radius row τ, and the three containment ledgers.

Finally apply Definition 311.6 to the same finite rows. The real-name seal has no coordinate for a quotient ball, host real equality, completeness datum, or selected point, so the composite overlap remains a finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ package with precisely the rows listed above. □

Theorem 311.13 (DyadicBall public bridge package)**.** *Let B be an accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packet whose public center-radius interface is the one of Definition 311.10. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *window handoff of Definition 311.4 and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 311.6 assemble a single finite bridge package. Its visible rows are exactly the center row, positive radius row, schedule row, scheduled observation row, containment ledger, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row displayed by the public interface.*

Under the overlap hypotheses of Definition 311.11 and Definition 311.12, the same bridge package is stable under common refinement. No quotient ball, host real equality, metric topology, completeness witness, or selected limit is introduced.

Proof. Unpack the public center-radius interface. It exposes exactly the finite center, radius, schedule, observation, containment, transport, provenance, and local naming rows of Definition 311.1.

Project the consumer rows through the RegSeqRat handoff and the RealUp seal boundary. Both consumer surfaces read the same displayed rows, so their joint package has no coordinate beyond the center-radius interface listed in the statement.

For refinement stability, apply the common-observation overlap theorem and then the refinement-overlap transitivity theorem. These results transport the displayed center, positive radius, schedule, scheduled observation, containment, hsame, Cont, Pkg, and NameCert rows through common refinement without adding a quotient ball or host real datum.

The exclusions follow by inspection of the assembled finite rows: neither the handoff nor the seal boundary introduces host real equality, metric topology, completeness data, or a selected limit. □

Theorem 311.14 (DyadicBall mature observation envelope)**.** *The public center-radius interface of Definition 311.10, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 311.4, the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 311.6, the common-observation overlap of Definition 311.11, the refinement-overlap transitivity of Definition 311.12, and the public bridge package of Definition 311.13 assemble a mature observation envelope for* $\mathsf{DyadicBall}^{\uparrow}$.

Every consumer-visible row in this envelope is one of the finite center, radius, schedule, observation, containment, transport, provenance, and NameCert *rows displayed by the public interface and the bridge package. The envelope has no row for quotient balls, host real equality, metric topology, completeness, or selected limits.*

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket mature observation envelope`

Proof. Start with the public center-radius interface. It lists the finite center, positive radius, schedule, scheduled observation, containment, transport, provenance, and local naming rows exported by an accepted $\mathsf{DyadicBall}^{\uparrow}$ packet.

Next pass the same rows through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal boundary. These two consumer surfaces read the displayed packet data and do not add a host equality coordinate, a quotient ball, a topology row, a completion datum, or a selected point.

For stability under shared observations, use the common-observation overlap and refinement-overlap transitivity theorems. They transport the same finite rows through common refinements, so the consumer envelope is unchanged when two visible packets are compared through a shared scheduled observation.

Finally apply the public bridge package. It already collects the interface, handoff, seal boundary, overlap, and refinement rows into one finite bridge surface, and its exclusions are exactly the non-rows named in the statement. □

Theorem 311.15 (DyadicBall finite intersection basis package)**.** *Let $B_0, \ldots, B_n$ be a finite displayed chain of accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packets sharing one schedule row k and one scheduled observation row o. Suppose the chain carries a terminal positive dyadic radius row τ, containment ledgers from τ into every packet, and refinement ledgers from τ into each adjacent overlap row displayed by the chain.*

Then the chain determines one finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer package. Its visible rows are exactly the shared schedule k, the shared observation o, the transported center rows, the terminal radius row τ, the containment ledgers into the displayed packets, the adjacent-overlap refinement ledgers, the displayed* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance rows, and the local* NameCert *rows.*

This package is a finite-intersection basis surface for the displayed $\mathsf{DyadicBall}^{\uparrow}$ *packets. It has no row for quotient balls, metric topology, host real equality, selected limits, or completeness.*

Proof. Read each accepted packet through the carrier of Definition 311.1. The shared schedule row, shared observation row, transported center row, positive radius row, containment ledger, transport rows, package provenance, and local naming row are the only coordinates exposed to the chain.

For each adjacent pair, apply Definition 311.11. The displayed refinement ledger from τ into the adjacent overlap row supplies the common positive radius witness, while the containment ledgers into the two packets supply the two containment coordinates. This produces one local overlap package for each adjacent step.

Move along the finite chain by repeatedly using Definition 311.12. Each step keeps the same schedule and scheduled observation rows, replaces the local overlap radius by the terminal radius row τ, and records only the containment, hsame, Cont, Pkg, and NameCert rows already displayed by the chain.

Finally apply Definition 311.13 to the resulting common-refinement surface. The bridge package reads the displayed finite ledger and repacks it as one $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer package, so no quotient ball, metric-topology row, host real equality, selected limit, or completeness datum is invoked. □

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket finite intersection basis package`

Theorem 311.16 (DyadicBall radius-window monotonicity)**.** *Let B be an accepted* $\mathsf{DyadicBall}^{\uparrow}$ *finite enclosure. If its displayed positive dyadic radius is narrowed or its finite observation window is restricted along displayed refinement rows, then the resulting finite enclosure is again an accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packet.*

The narrowed packet preserves the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and every consumer-visible read factors through the original* hsame, Cont, Pkg, *and local* NameCert *rows. No quotient ball, metric-topology row, host real equality, selected limit, or completeness datum is introduced by the narrowing.*

Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteEnclosure radius window monotonicity`
Lean4 ✓ `BEDC.Derived.DyadicBallUp.DyadicBallFiniteWindowPacket radius window monotonicity`

Proof. Start from the accepted center-radius carrier of B and apply Definition 311.3. Narrowing the displayed radius or observation window keeps the finite center, positive-radius, containment, transport, provenance, and naming coordinates inside the $\mathsf{DyadicBall}^{\uparrow}$ carrier.

Use Definition 311.11 for the common visible schedule row. It identifies the narrowed observation as a shared finite overlap of the original packet and the restricted packet.

Apply Definition 311.12 to compose the radius and observation refinements. The composed refinement still reads through the same displayed hsame, Cont, Pkg, and NameCert rows.

Finally invoke Definition 311.15. The finite basis package repacks the narrowed enclosure as the same $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer surface, so the handoff is preserved and no excluded host or quotient coordinate is available. □

Theorem 311.17 (DyadicBall terminal Real seal boundary)**.** *Let $B_0, \ldots, B_n$ be a finite displayed chain of accepted* $\mathsf{DyadicBall}^{\uparrow}$ *packets sharing one schedule row k and one scheduled observation row o. Suppose the chain carries a terminal positive dyadic radius row τ, containment ledgers from τ into every displayed packet, and adjacent-overlap refinement ledgers as in Definition 311.15.*

Then every terminal $\mathsf{Real}^{\uparrow}$ *seal read of this chain factors through exactly the finite package named by Definition 311.15: the shared schedule row, the shared observation row, the transported center rows, the terminal radius row, the containment ledgers, the adjacent-overlap refinement ledgers, the displayed* hsame *rows, the displayed* Cont *routes, the* Pkg *provenance rows, and the local* NameCert *rows.*

The seal has no readable row for quotient balls, host real equality, metric topology, completion, selected limits, or any real-completeness datum.

Proof. First apply Definition 311.15 to the displayed chain. This assembles the terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer package over the shared schedule and observation rows, the transported centers, the terminal radius, the containment ledgers, the adjacent-overlap refinement ledgers, and the visible transport, provenance, and naming rows.

Next apply Definition 311.6 to the resulting consumer package. The $\mathsf{Real}^{\uparrow}$ seal reads only the rows exported by that package, so no host real equality, quotient ball, topology coordinate, completion row, selected limit, or real-completeness witness can be inspected.

Finally use Definition 311.14. It identifies the same terminal package as the mature observation envelope for finite center-radius chains, so every terminal seal read remains confined to the named finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicBall}^{\uparrow}$ packet is generated by a finite dyadic center row, a positive dyadic radius row, a scheduled observation row, a containment ledger, componentwise hsame and Cont transport, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 311.14 and Definition 311.15 bridge the public center-radius packet to the shared $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer surface and close the finite common-refinement basis package over center, radius, schedule, observation, containment, transport, provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicBall}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicBallUp.DyadicBallPacket\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close real completeness, quotient stream equality, host real balls, metric topology, or a standard bridge.

Upgrade path: The mature paper surface is the public center-radius bridge plus the finite common-refinement consumer package; formal bridge checking remains a separate Lean-side axis.

Chapter 312

A Concrete Naming Certificate for $\mathsf{Monodromy}^{\uparrow}$

The $\mathsf{Monodromy}^{\uparrow}$ interface records the finite certificate surface of carrying a fibre or local-system row around a displayed loop and reading back the resulting transport row. The loop transport packet consumes the finite holonomy transport surface of chapter 299 before it packages the returned-row ledger, while the $\mathsf{RiemannHilbert}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, and $\mathsf{Curvature}^{\uparrow}$ rows remain provenance sources. The classical reference point is the regular-singular monodromy and Riemann-Hilbert tradition of Deligne, read here only through finite $\mathsf{NameCert}_{\mathsf{Holonomy}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{RiemannHilbert}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$, and $\mathsf{NameCert}_{\mathsf{Curvature}^{\uparrow}}$ rows.

312.1 Carrier definition

Definition 312.1 (Monodromy BHist transport carrier)**.** *A* $\mathsf{Monodromy}^{\uparrow}$ *carrier is a finite* BHist *transport packet*

$$(\gamma, b, \mathcal{L}, F, \lambda, \tau, \rho)$$

with the following displayed rows.

1. γ *is a loop row with named source endpoint, return endpoint, and endpoint* hsame *comparison.*
2. b *is the base endpoint row over which the loop is read.*
3. $\mathcal{L}$ *is a local-system, sheaf, or fibre row supplied through* $\mathsf{NameCert}_{\mathsf{Sheaf}^{\uparrow}}$ *together with the connection row supplied through* $\mathsf{NameCert}_{\mathsf{Connection}^{\uparrow}}$.
4. F *is the returned fibre or stalk row obtained by following the displayed loop.*
5. λ *is the finite continuation ledger for loop subdivision, transport concatenation, endpoint readback, and returned-row comparison.*
6. τ *records componentwise* hsame *transports for the loop row, base endpoint, local-system or fibre row, returned row, and continuation ledger.*
7. ρ *is the* Pkg *provenance row over* $\mathsf{Holonomy}^{\uparrow}$, $\mathsf{RiemannHilbert}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, *and* $\mathsf{Curvature}^{\uparrow}$.

The carrier exports no analytic fundamental group, no host representation category, and no unbounded continuation procedure beyond these displayed rows.

Lean4 ✓ `BEDC.Derived.MonodromyUp.taste gate`

Definition 312.2 (Monodromy finite transport classifier)**.** *The* $\mathsf{Monodromy}^{\uparrow}$ *classifier relates two accepted packets exactly when the loop rows, base endpoint rows, local-system or fibre rows, returned rows, finite continuation ledgers, componentwise transport rows, and package provenance rows are related by the displayed* hsame *comparisons. The classifier is therefore a finite rowwise comparison on the packet of Definition 312.1; it does not classify host paths, analytic continuation germs, or representations outside the packet.*

312.2 The certificate

Theorem 312.3 (Monodromy loop-continuation obligation)**.** *Every accepted* $\mathsf{Monodromy}^{\uparrow}$ *packet binds its loop-continuation row to the displayed base endpoint, local-system or fibre row, connection row, returned row, and finite continuation ledger of Definition 312.1. If two accepted packets have componentwise* hsame *transports of those rows, then the transported returned row is again accepted by the classifier of Definition 312.2.*

Lean4 ✓ `BEDC.Derived.MonodromyUp.MonodromyTransportCarrier loop continuation stability`
Lean4 ✓ `BEDC.Derived.MonodromyUp.MonodromyTransportPacket loop continuation obligation`

Proof. Project the accepted packet to its loop row γ, base endpoint b, local-system or fibre row $\mathcal{L}$, and connection source. These rows name the only source data used by the continuation route.

Next read the finite continuation ledger λ. Its entries record loop subdivision, transport concatenation, and endpoint readback, so the returned row F is determined by a displayed route rather than by an ambient analytic continuation object.

Finally apply the componentwise hsame transports stored in τ. The classifier accepts exactly those rowwise transports, and repacking the transported loop, base, fibre, returned row, ledger, and provenance rows gives another accepted $\mathsf{Monodromy}^{\uparrow}$ packet. □

Theorem 312.4 (Monodromy returned-row provenance exactness)**.** *For any accepted* $\mathsf{Monodromy}^{\uparrow}$ *packet, every returned fibre row or returned stalk row accepted by the classifier is obtained from the displayed loop row, base endpoint row, local-system or fibre row, finite continuation ledger, componentwise* hsame *transports, and* Pkg *provenance named in Definition 312.1. A transported returned row is accepted only through these same rows. No host path equality, analytic continuation germ, or external representation row supplies a returned-row source for the packet.*

Lean4 ✓ `BEDC.Derived.MonodromyUp.MonodromyTransportPacket returned row provenance exactness`

Proof. Project the packet to γ, b, $\mathcal{L}$, F, λ, τ, and ρ. These coordinates name the loop, endpoint, local-system or fibre source, returned row, continuation ledger, rowwise transports, and package provenance visible to the classifier.

Apply Definition 312.3 to the transport step. The obligation binds the returned row to the displayed loop route and to the finite continuation ledger, and the transported returned row is repacked by the same componentwise hsame transports.

The classifier of Definition 312.2 has no coordinate for host paths, analytic germs, or representation rows outside the packet. Hence every accepted returned row, including a transported one, has exactly the displayed provenance listed in the carrier. □

Theorem 312.5 (Monodromy NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$ *obligation surface consists of five finite rows.*

1. *The carrier row supplied by Definition 312.1.*

2. *The classifier row supplied by Definition 312.2.*
3. *The loop-continuation stability row supplied by Definition 312.3.*
4. *The returned-row provenance row of Definition 312.4, naming the consumed* $\mathsf{Holonomy}^{\uparrow}$, $\mathsf{RiemannHilbert}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, *and* $\mathsf{Curvature}^{\uparrow}$ *certificates through* Pkg.
5. *The standard-boundary exclusion row saying that no analytic fundamental group, representation category, unbounded continuation solver, or regular-singular comparison theorem is exported by the seed packet.*

Proof. Read the carrier fields. The loop, base endpoint, local-system or fibre, returned row, continuation ledger, componentwise transport rows, and package provenance supply the carrier and classifier obligations directly.

Use Definition 312.3 for the stability row and Definition 312.4 for the returned-row provenance row. Their proofs record that transported packets remain inside the same finite row shape, so no extra continuation principle is needed.

The provenance and boundary rows are projections from ρ and from the absence of any host path, representation, or analytic solver coordinate in the carrier. These five rows exhaust the seed obligation surface for $\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$. □

Theorem 312.6 (Monodromy standard boundary scope)**.** *The standard mathematical boundary of* $\mathsf{Monodromy}^{\uparrow}$ *is limited in this chapter to Deligne-style regular-singular monodromy and Riemann-Hilbert motivation as a naming source. The BEDC packet itself exposes only the displayed loop row, base endpoint, local-system or fibre row, finite continuation ledger, returned-row transport, componentwise* hsame *comparisons, and* Pkg *provenance over* $\mathsf{Holonomy}^{\uparrow}$, $\mathsf{RiemannHilbert}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, *and* $\mathsf{Curvature}^{\uparrow}$.

Proof. Project any consumer-visible monodromy readback from the carrier. The only available coordinates are the loop, base, local-system or fibre, returned row, finite ledger, transport comparisons, and dependency provenance rows.

The external regular-singular and Riemann-Hilbert traditions justify the human-known name, but they are not fields of the packet. In particular, the carrier has no coordinate for a fundamental-group representation, analytic continuation theorem, regular-singular classification, or host category equivalence.

Thus the boundary is a finite scope statement about what the certificate rows expose. Standard analytic or categorical monodromy results remain outside this seed chapter until they are represented by additional displayed BEDC rows. □

Theorem 312.7 (Monodromy public loop readback export)**.** *The public* $\mathsf{Monodromy}^{\uparrow}$ *readback exports exactly the loop row, base endpoint row, local-system or fibre row, returned row, finite continuation ledger, componentwise* hsame *transports, and* Pkg *provenance over* $\mathsf{Holonomy}^{\uparrow}$, $\mathsf{RiemannHilbert}^{\uparrow}$, $\mathsf{Sheaf}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, *and* $\mathsf{Curvature}^{\uparrow}$.

The export does not include analytic fundamental groups, host representation categories, unbounded continuation solvers, or a Riemann-Hilbert equivalence.

Proof. Start from the carrier definition Definition 312.1. Its coordinates are precisely the loop row, base endpoint, local-system or fibre row, returned row, finite continuation ledger, componentwise transports, and dependency provenance.

Apply the finite transport classifier of Definition 312.2. It permits only rowwise comparison of the displayed loop, endpoint, fibre or local-system, returned, ledger, transport, and provenance rows.

Next use Definition 312.3 and Definition 312.4. The first theorem binds the returned row to the finite loop-continuation ledger, and the second shows that no returned row has a source outside the displayed packet.

Finally apply Definition 312.5 together with Definition 312.6. The local $\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$ surface and the standard boundary scope repack the same rows as the public readback while keeping analytic fundamental groups, host representation categories, unbounded continuation solvers, and Riemann-Hilbert equivalence outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$.

Constructive story: A monodromy packet is a finite BHist transport ledger over a displayed loop, base endpoint, local-system or fibre row, returned row, finite continuation ledger, componentwise hsame transports, and package provenance. Classification is rowwise transport of those finite coordinates, with Deligne-style regular-singular monodromy and Riemann-Hilbert theory serving only as the human-known naming source.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$, publicClosed)

Scope closed: The public boundary binds the loop-continuation rows, returned-row provenance, local NameCert surface, public loop readback export, and RiemannHilbert, connection, and sheaf sources through BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Monodromy}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MonodromyUp.MonodromyTransportPacket\_loop\_continuation\_obligation`

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic continuation, a fundamental-group representation theorem, regular-singular classification, Riemann-Hilbert equivalence, or a standard bridge.

Upgrade path: The next paper-grade step is a standard bridge boundary for the public monodromy readback packet.

Lemma 312.8 (Monodromy constant-loop returned-row identity)**.** *Let*

$$(\gamma, b, \mathcal{L}, F, \lambda, \tau, \rho)$$

be an accepted $\mathsf{Monodromy}^{\uparrow}$ *transport packet. Suppose that* γ *is the displayed constant loop at* b*: its source endpoint row, return endpoint row, and endpoint comparison are the reflexive* b *row. Suppose also that* λ *is the empty finite continuation ledger. Then the returned fibre or stalk row* F *is in the same finite* $\mathsf{Monodromy}^{\uparrow}$ *transport class as the input local-system or fibre row* $\mathcal{L}$*. More explicitly, the classifier of Definition 312.2 accepts the comparison whose loop, base, ledger, transport, and provenance coordinates are the displayed reflexive rows, and whose returned-row coordinate is the no-step comparison* $F\mathsf{hsame}\mathcal{L}$ *read from the identity-route packet.*

Proof. Project the accepted packet through Definition 312.1. The loop row starts at b, returns to b, and carries the displayed endpoint comparison $b\mathsf{hsame}b$. The ledger row λ has no subdivision entry, no transport-concatenation entry, and no nontrivial endpoint-readback entry.

Apply Definition 312.3. It binds the returned row F to the same displayed base endpoint, the same input row $\mathcal{L}$, and the same finite ledger. In the present hypotheses the bound route is the empty identity route, so there is no intermediate loop segment and no later continuation row from which a different returned row could be sourced.

Now use Definition 312.4. Any classifier-accepted returned row must come from the displayed loop row, base endpoint row, input local-system or fibre row, empty ledger, componentwise hsame transports, and package provenance already present in the packet. Since the ledger is empty and both endpoints are the same displayed base row, the returned-row provenance has only the no-step readback of $\mathcal{L}$ at b. The componentwise transport row τ therefore supplies the comparison $F\mathsf{hsame}\mathcal{L}$ inside the packet, not from an analytic continuation germ or an outside representation row.

Finally repack this comparison into the finite classifier of Definition 312.2. The loop, base, ledger, transport, and provenance coordinates are reflexive copies of the displayed packet rows, while the returned coordinate is the no-step comparison $F\mathsf{hsame}\mathcal{L}$. Hence F and $\mathcal{L}$ are classified together by the finite $\mathsf{Monodromy}^{\uparrow}$ transport classifier. $\square$

Chapter 313

A Concrete Naming Certificate for $\mathsf{Lyapunov}^{\uparrow}$

The $\mathsf{Lyapunov}^{\uparrow}$ interface records the finite stability ledger used by Lyapunov arguments for displayed dynamical systems and quadratic matrix certificates. In the classical presentation of Khalil, *Nonlinear Systems* (2002), the proof reads a positive function and a one-step or derivative decrease condition to conclude stability in a stated neighborhood. The BEDC certificate below keeps only the finite rows needed to name that surface: state rows, transition rows, quadratic certificate rows, positive-definite witnesses, one-step decrease rows, transports, continuation routes, package provenance, and a local NameCert row.

Definition 313.1 (Lyapunov finite stability carrier)**.** *A* $\mathsf{Lyapunov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, T, Q, P, D, H, C, G, N)$$

with the following rows:

1. *X is a displayed state-row packet read through* $\mathsf{DynSystem}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$*;*
2. *T is a displayed transition-row packet for the sampled or one-step state motion, with its source and target slots named by X;*
3. *Q is a quadratic certificate row read through* $\mathsf{Mat}^{\uparrow}$ *and the scalar boundary of* RealAlgOrder*;*
4. *P is a positive-definite witness row for the displayed quadratic certificate on the state rows carried by the packet;*
5. *D is a one-step decrease ledger saying that the displayed successor row has a nonincreasing, or strictly decreasing when named, quadratic value relative to the displayed predecessor row;*
6. *H records the componentwise* hsame *transports for state, transition, quadratic, positivity, and decrease rows;*
7. *C records the* Cont *routes from predecessor states to successor states and from both states to the quadratic-value comparison row;*
8. *G is the* Pkg *provenance naming* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* RealAlgOrder*, and* $\mathsf{Riccati}^{\uparrow}$ *as the only imported certificate boundaries;*
9. *N is the local* NameCert *row for the resulting finite stability surface.*

The carrier predicate is $\mathsf{LyapunovLedger}(X, T, Q, P, D, H, C, G, N)$*. Two accepted packets classify together only when all nine rows transport by the displayed* hsame *data and follow the same* Cont *routes named in* G*.*

Lean4 ✓ `BEDC.Derived.LyapunovUp.taste gate`

Theorem 313.2 (Lyapunov decrease ledger)**.** *Let* $L = (X, T, Q, P, D, H, C, G, N)$ *be an accepted* $\mathsf{Lyapunov}^{\uparrow}$ *carrier. The decrease information exported by* L *is exactly the finite row* D*: a displayed comparison between the quadratic value of the predecessor state row and the quadratic value of the successor state row, routed through* C *and typed by* RealAlgOrder*. The packet exports no differentiability assumption, no infinite-time convergence assertion, no analytic limit, and no external optimization solver semantics.*

Proof. Project the carrier of Definition 313.1. The predecessor and successor state rows are the visible endpoints of the transition row T.

Use the continuation routes in C to read the two quadratic values from the same displayed certificate row Q. The order comparison between those two values is precisely the row D, and the positive-definite row P supplies only the local sign witness for Q on the named state rows.

The remaining coordinates are transport, provenance, and the local name row. None of them contains a derivative, an infinite trajectory, a limiting convergence theorem, or a solver state, so the decrease surface is exhausted by the finite comparison row described above. □

Theorem 313.3 (Lyapunov NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet of Definition 313.1;*
2. *classifier and stability: the componentwise* hsame *transport rule for state, transition, quadratic, positivity, decrease, continuation, provenance, and local name rows;*
3. *certificate ledger: the positive-definite witness row* P *and the one-step decrease row* D *of Definition 313.2;*
4. *consumer boundary: the finite stability read permitted to downstream control packets by Definition 313.4;*
5. *provenance: the* Pkg *row tying the packet to* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* RealAlgOrder*, and* $\mathsf{Riccati}^{\uparrow}$*.*

These rows form the full public seed surface for the Lyapunov certificate name.

Lean4 ✓ `BEDC.Derived.LyapunovUp.LyapunovLedger namecert obligation surface`

Proof. The carrier row fixes every coordinate that can be named by the certificate. The classifier is the finite membership test on those coordinates, and the stability clause is componentwise hsame transport along the same Cont routes.

The certificate ledger consists only of the positive-definite row and the decrease row. The consumer boundary then determines which downstream reads are available from those rows.

Finally, the package row names every imported BEDC certificate boundary. Since the carrier has no coordinate for analytic trajectory convergence, smooth derivatives, basin computation, or solver output, the five listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$ obligation surface. □

Theorem 313.4 (Lyapunov consumer boundary)**.** *Every consumer of* $\mathsf{NameCert}_{\mathsf{Lyapunov}^\uparrow}$ *may read only the displayed state rows, transition rows, quadratic certificate row, positive-definite witness, one-step decrease ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row of Definition 313.1. A* $\mathsf{DynSystem}^\uparrow$ *consumer may use those rows as a finite stability certificate, and a* $\mathsf{Riccati}^\uparrow$ *or matrix-control consumer may reuse the quadratic and order rows, but no consumer obtains a basin-of-attraction computation, asymptotic convergence theorem, differentiable flow, or external synthesis solver from the certificate alone.*

Proof. Read a consumer map as a projection from the accepted carrier. The $\mathsf{DynSystem}^\uparrow$ projection returns the displayed state and transition rows together with the decrease comparison.

The matrix-control projection returns the quadratic row, its positive-definite witness, and the RealAlgOrder comparison row. The $\mathsf{Riccati}^\uparrow$ projection can reuse the same finite matrix and order coordinates as a named quadratic ledger, without adding any Riccati recursion step.

All other objects named in the statement are absent from the carrier and from the package provenance. They therefore cannot be read through $\mathsf{NameCert}_{\mathsf{Lyapunov}^\uparrow}$. □

Lean4 ✓ `BEDC.Derived.LyapunovUp.LyapunovPacket consumer boundary`

Theorem 313.5 (Lyapunov DynSystem sibling route)**.** *Every accepted* $\mathsf{Lyapunov}^\uparrow$ *stability read consumes the* $\mathsf{DynSystem}^\uparrow$ *state-and-transition surface of chapter 195 before the quadratic row, positive-definite row, and decrease ledger can be read. The route remains finite through* $X, T, Q, P, D, H, C, G, N$ *and does not export a basin computation, infinite trajectory, differentiable flow, or solver semantics.*

Proof. Project the carrier of Definition 313.1. The state row X and transition row T are the entry surface of the packet, and both are named through the displayed $\mathsf{DynSystem}^\uparrow$ dependency before any quadratic read is available.

The decrease ledger of Definition 313.2 then reads Q, P, D only along the continuation routes C and package row G. The consumer boundary of Definition 313.4 confirms that a downstream stability consumer receives exactly these finite rows.

Thus the substantive sibling dependency is the transition row supplied by chapter 195. Since the route is exhausted by $X, T, Q, P, D, H, C, G, N$, no basin computation, infinite trajectory, differentiable flow, or solver semantics is exported. □

Theorem 313.6 (Lyapunov stability transport)**.** *Let* $L = (X, T, Q, P, D, H, C, G, N)$ *be an accepted* $\mathsf{Lyapunov}^\uparrow$ *carrier and let* $L' = (X', T', Q', P', D', H', C', G', N')$ *be obtained by displayed* hsame *transport of all carrier rows, preserving the* Cont *routes named by the provenance row. Then* L' *is again an accepted* $\mathsf{Lyapunov}^\uparrow$ *carrier, and its finite stability read is the transported positive-definite and one-step decrease ledger over* X'*,* T'*, and* Q'*.*

Lean4 ✓ `BEDC.Derived.LyapunovUp.LyapunovPacket stability transport`

Proof. Transport the state and transition rows first. The preserved continuation routes identify the successor and predecessor slots of T' with the transported state rows in X'.

Next transport the quadratic certificate and its positive-definite witness. Because the order boundary is the same RealAlgOrder row carried by the provenance, the transported sign witness remains attached to Q' on the transported state rows.

Finally transport the decrease ledger, package row, and local name row. Repack the nine transported coordinates as $\mathsf{LyapunovLedger}(X', T', Q', P', D', H', C', G', N')$. The stability read of the new packet is exactly the transported positive-definite and decrease rows, since the carrier has no additional stability coordinate. □

Lean4 ✓ `BEDC.Derived.LyapunovUp.LyapunovLedger stability transport`

Theorem 313.7 (Lyapunov public export)**.** *The decrease ledger,* $\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$ *obligation surface, consumer boundary, and stability transport export exactly the finite state, transition, positivity, decrease, transport, continuation, provenance, and naming rows of* $\mathsf{Lyapunov}^{\uparrow}$*.*

The public export contains no minimizer-existence principle, no global asymptotic completeness theorem, no analytic regularity assertion, and no standard-model dynamics claim.

Proof. Project the finite carrier of Definition 313.1. The only state and transition data are the rows X and T, and the only certificate data are the quadratic row, the positivity witness, and the decrease ledger.

Apply Definition 313.2 and Definition 313.3. These identify the decrease row and the local naming surface over the displayed finite packet.

Use Definition 313.4 and Definition 313.6 to expose the consumer-facing rows and their displayed transport behavior. Since all cited rows are projections from the same finite carrier, the excluded analytic and standard-model claims have no coordinate in the public export. □

Closure status for $\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$**.**

Constructive story: A Lyapunov stability row is seeded by a finite BHist packet of state rows, transition rows, a quadratic certificate row, a positive-definite witness row, a one-step decrease ledger, hsame transports, Cont routes, Pkg provenance naming $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, RealAlgOrder, and $\mathsf{Riccati}^{\uparrow}$, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$, publicClosed)

Scope closed: Definition 313.7 binds only the finite state, transition, quadratic, positivity, one-step decrease, transport, continuation, provenance, and name rows listed in Definition 313.1.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Lyapunov}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LyapunovUp.LyapunovLedger\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close differentiable Lyapunov theory, asymptotic convergence, basin computation, controller synthesis, external solver semantics, or a standard bridge.

Upgrade path: Bridged closure requires a standard bridge over the finite state, transition, positivity, decrease, transport, continuation, provenance, and naming rows.

Chapter 314

A Concrete Naming Certificate for $\mathsf{DyadicCompletion}^{\uparrow}$

The $\mathsf{DyadicCompletion}^{\uparrow}$ surface is the constructive completion handoff between finite dyadic observations and the regular real-name layer. Following the Bishop–Bridges constructive-analysis treatment of Cauchy completion for dyadics, the chapter records only scheduled finite observations, explicit regular-tail witnesses, and visible precision ledgers. It does not identify streams by quotient classes, select limits by choice, or replace the stream surface by Dedekind cuts.

314.1 Carrier Surface

Definition 314.1 (DyadicCompletion finite-window carrier)**.** *A* $\mathsf{DyadicCompletion}^{\uparrow}$ *carrier row is a finite* BHist *packet* (d, w, r, s, L, P) *with the following visible coordinates:*

1. *dyadic observation rows* d *accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *certified stream windows* w *requested through* $\mathsf{StreamName}^{\uparrow}$*;*
3. *a regular-tail classifier row* r *whose comparisons are read through* $\mathsf{RegSeqRat}^{\uparrow}$*;*
4. *a real-name boundary row* s *naming the data consumed by* $\mathsf{Real}^{\uparrow}$*;*
5. *a precision ledger* L *recording every finite schedule, tail bound, and window restriction used by the packet;*
6. *a* Pkg *provenance row* P *naming the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *certificates.*

The carrier stores finite windows and their ledgers. It contains no completed host real, no quotient class of Cauchy streams, no hidden precision schedule, and no Dedekind-cut classifier.

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.TasteGate.DyadicCompletionTasteGate single carrier alignment`

Theorem 314.2 (DyadicCompletion NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicCompletion}^{\uparrow}}$ *obligation surface consists exactly of the carrier of Definition 314.1, the componentwise classifier that compares dyadic observations and stream-window rows, the regular-tail stability row read through* $\mathsf{RegSeqRat}^{\uparrow}$*, the* $\mathsf{Real}^{\uparrow}$ *handoff row for accepted real-name consumers, and the precision ledger boundary that records every schedule consumed by the packet.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionPacket namecert obligation surface`

Proof. Project the carrier coordinates. The dyadic, stream, regular-tail, real-handoff, ledger, and provenance rows are all stored as finite BHist data.

The classifier is componentwise because a consumer can compare only the rows it can read: dyadic observations through $\mathsf{DyadicRatCore}^{\uparrow}$, scheduled windows through $\mathsf{StreamName}^{\uparrow}$, and tail comparisons through $\mathsf{RegSeqRat}^{\uparrow}$.

The $\mathsf{Real}^{\uparrow}$ handoff is therefore a boundary projection rather than a selected limit. Repacking these projections gives the listed obligation surface and no additional completion object. □

Theorem 314.3 (DyadicCompletion constant dyadic embedding)**.** *Every accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *row q determines a* $\mathsf{DyadicCompletion}^{\uparrow}$ *carrier whose scheduled stream windows all read q, whose regular-tail classifier rows are reflexive dyadic comparisons, whose* $\mathsf{Real}^{\uparrow}$ *boundary row is the constant real-name handoff, and whose precision ledger records only the requested windows and their constant readback routes.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionPacket constant embedding`

Proof. Use the constant $\mathsf{StreamName}^{\uparrow}$ schedule with value row q at each requested finite window. Each endpoint read is already accepted by $\mathsf{DyadicRatCore}^{\uparrow}$.

For the regular-tail coordinate, compare any two displayed windows by the reflexive dyadic classifier on the same row q. The bound rows are the trivial constant-tail witnesses consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

Finally attach the $\mathsf{Real}^{\uparrow}$ handoff row that reads the same constant windows. The ledger records the schedule requests and the constant routes, so no quotient, limit selector, or hidden precision choice is introduced. □

Theorem 314.4 (DyadicCompletion regular-tail stability)**.** *Let C be an accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *carrier. If a second carrier C' is obtained by* hsame *transport of the dyadic observation rows, certified stream windows, regular-tail comparisons, real-name boundary row, ledger, and provenance, then C' is accepted and has the same regular-tail classifier surface over* $\mathsf{RegSeqRat}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionWindowCarrier regular tail stability`
Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionWindowPacket regular tail stability`
Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionPacket regular tail stability`

Proof. Transport the dyadic observation rows and certified stream windows along the displayed hsame comparisons. The $\mathsf{StreamName}^{\uparrow}$ schedule is finite, so each requested window is repacked with the transported row.

The regular-tail classifier is a finite family of dyadic bound ledgers. Transport each bound row and each comparison route componentwise through $\mathsf{RegSeqRat}^{\uparrow}$.

The real-name boundary and precision ledger are projections of the same windows. Transporting those projections yields the accepted carrier C' and preserves the regular-tail surface. □

Theorem 314.5 (DyadicCompletion StreamName handoff)**.** *Every accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *carrier exposes a* $\mathsf{StreamName}^{\uparrow}$ *handoff: for each finite precision request, the handoff returns exactly the scheduled dyadic observation row, its window certificate, the regular-tail comparison rows visible at that precision, and the ledger entries proving that the same rows are available to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionWindowPacket streamname handoff`

Proof. Unpack the carrier. A precision request first selects a finite stream window, and the carrier stores the dyadic observation row and certificate attached to that window.

The regular-tail row records the comparisons visible at the same precision. The ledger points to the shared schedule entry, so the $\mathsf{RegSeqRat}^{\uparrow}$ consumer and the $\mathsf{Real}^{\uparrow}$ boundary read the same finite rows.

Thus the handoff is only a projection from stored data. It does not synthesize a global stream outside the certified windows. □

Theorem 314.6 (DyadicCompletion ledger boundary)**.** *The precision ledger of an accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *carrier is the whole boundary between dyadic finite observations and real-name consumption: every visible* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *row used by the carrier is named in the ledger, and no consumer can project an unstored modulus, quotient representative, Dedekind cut, or ambient completed real from the packet.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionPacket ledger boundary`

Proof. Inspect the ledger coordinate in Definition 314.1. It lists the finite schedules, tail bounds, window restrictions, and continuation routes consumed by each displayed row.

Any downstream read by $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ must pass through one of those listed routes. Since the carrier has no coordinate for an unstored modulus, quotient representative, Dedekind cut, or completed host real, such data cannot be projected from the packet. □

Theorem 314.7 (DyadicCompletion scoped dependency packet)**.** *The carrier of Definition 314.1, the componentwise classifier of Definition 314.2, the constant dyadic embedding of Definition 314.3, the regular-tail stability of Definition 314.4, the* $\mathsf{StreamName}^{\uparrow}$ *handoff of Definition 314.5, the* $\mathsf{Real}^{\uparrow}$ *handoff boundary named in the carrier, and the ledger boundary of Definition 314.6 depend only on finite* BHist *rows, displayed* hsame *comparisons, visible* Cont *routes, the* Pkg *provenance row, and the named* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source certificates.*

Thus every accepted $\mathsf{DyadicCompletion}^{\uparrow}$ *packet is scoped by the finite BEDC rows and source certificates listed above.*

Proof. Project the finite packet coordinates of Definition 314.1. They are dyadic observation rows, certified stream windows, regular-tail rows, a real-name handoff row, ledger entries, and one Pkg provenance row naming the source certificates.

The componentwise classifier is the classifier component of Definition 314.2. The constant embedding row is Definition 314.3, and the regular-tail row is transported by Definition 314.4; each cited block reads only finite BHist rows through hsame, Cont, and the displayed package provenance.

The consumer handoff is Definition 314.5. Its stream-name and real-name reads are the same scheduled rows already present in the carrier, with the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ source certificates named by the package row.

Finally apply Definition 314.6. The ledger names every $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ row available to a consumer, and the carrier has no coordinate for a quotient stream, Dedekind cut, host real, or chosen limit. □

Theorem 314.8 (DyadicCompletion public RegSeqRat Real export)**.** *The carrier of Definition 314.1, the* $\mathsf{NameCert}_{\mathsf{DyadicCompletion}^{\uparrow}}$ *obligation surface of Definition 314.2, the constant dyadic embedding of Definition 314.3, regular-tail stability of Definition 314.4,* $\mathsf{StreamName}^{\uparrow}$ *handoff of Definition 314.5, ledger boundary of Definition 314.6, and scoped dependency packet of Definition 314.7 assemble the public* $\mathsf{DyadicCompletion}^{\uparrow}$ *export consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

The export exposes only finite dyadic observations, certified stream windows, regular-tail rows, real-boundary rows, precision ledger rows, BHist*,* hsame*,* Cont*,* Pkg*, and source certificates. It exports no quotient stream, Dedekind cut, host real, choice-selected limit, or completeness principle.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionWindowPacket public regseqrat real export`

Proof. Project the finite packet from Definition 314.1. Its coordinates are scheduled dyadic observations, certified stream windows, regular-tail rows, real-name handoff rows, ledger entries, and package provenance.

Compose the public export from the cited blocks. The obligation surface supplies the carrier and classifier rows, the constant embedding supplies dyadic constants, regular-tail stability transports tail rows, and the $\mathsf{StreamName}^{\uparrow}$ handoff fixes the downstream stream-window reads.

The ledger boundary names the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer-facing rows, while the scoped dependency package binds all reads to finite BHist rows, hsame, Cont, Pkg, NameCert, and the listed source certificates.

The excluded principles are therefore outside the export. A quotient stream, Dedekind cut, host real, choice-selected limit, or completeness principle would require a coordinate not present in the finite packet. □

Theorem 314.9 (DyadicCompletion finite-window standard bridge)**.** *The carrier of Definition 314.1, the* $\mathsf{NameCert}_{\mathsf{DyadicCompletion}^{\uparrow}}$ *obligation surface of Definition 314.2, the* $\mathsf{StreamName}^{\uparrow}$ *handoff of Definition 314.5, regular-tail stability of Definition 314.4, the ledger boundary of Definition 314.6, the scoped dependency packet of Definition 314.7, and the public* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *export of Definition 314.8 determine the standard constructive finite-window reading of* $\mathsf{DyadicCompletion}^{\uparrow}$*.*

Every consumer read factors through finite dyadic observation rows, stream-window rows, regular-tail rows, real-boundary rows, precision ledger rows, BHist*,* hsame*,* Cont*,* Pkg*, and the listed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source certificates. The bridge exports no quotient Cauchy stream, Dedekind cut, host real limit, choice-selected limit, or completeness principle.*

Lean4 ✓ `BEDC.Derived.DyadicCompletionUp.DyadicCompletionWindowPacket finite window standard bridge`

Proof. Project the carrier from Definition 314.1. The projected coordinates are the dyadic observation rows, stream-window rows, regular-tail rows, real-name boundary row, precision ledger, and package provenance. These are the only rows available to the $\mathsf{NameCert}_{\mathsf{DyadicCompletion}^{\uparrow}}$ obligation surface.

Use Definition 314.5 for window reads. The handoff sends each downstream stream access to a scheduled stream-window row, and Definition 314.4 transports the regular-tail rows along the displayed classifier rather than through an unstored tail representative.

Apply Definition 314.6. The ledger lists every precision, real-boundary, continuation, history, package, and source-certificate row readable by a consumer, so no unstored modulus, quotient representative, Dedekind cut, host real limit, choice-selected limit, or completeness witness is available.

Finally use Definition 314.8 to route the same finite rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. Since the scoped dependency packet of Definition 314.7 binds those reads to BHist, hsame, Cont, Pkg, and the listed source certificates, the resulting consumer schema is exactly the standard constructive finite-window reading. □

Theorem 314.10 (DyadicCompletion mature RegSeqRat Real consumer coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{DyadicCompletion}^{\uparrow}$ *packet receives precisely the* $\mathsf{NameCert}_{\mathsf{DyadicCompletion}^{\uparrow}}$ *obligation surface, constant dyadic embedding, regular-tail stability,* $\mathsf{StreamName}^{\uparrow}$ *handoff, ledger boundary, scoped dependency packet, public* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *export, and finite-window standard bridge of Definition 314.9.*

The coverage contains no quotient completion, Dedekind cut, trichotomy, host equality, chosen limit, or extra completion principle.

Proof. Project the packet through Definition 314.2 and Definition 314.3. This exposes the carrier, classifier, and constant dyadic rows available to any consumer.

Next apply Definition 314.4 and Definition 314.5. These rows keep the regular tail and stream-name reads inside the displayed finite schedules and real-boundary handoff rows.

Use Definition 314.6 together with Definition 314.7. The ledger and scoped package list every BHist, hsame, Cont, Pkg, source certificate, precision, and real-boundary row readable by the consumer.

Finally apply Definition 314.8 and Definition 314.9. The public export and standard bridge repack exactly the same finite rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. Since none of the cited rows contains quotient completion, Dedekind cuts, trichotomy, host equality, or a chosen limit, those data are not part of the mature consumer coverage. □

Theorem 314.11 (DyadicCompletion finite-window ledger source exhaustion)**.** *Every finite-window read of an accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *packet by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through exactly the ledger-named source rows of Definition 314.1: dyadic observation rows,* $\mathsf{StreamName}^{\uparrow}$ *window rows, regular-tail comparison rows, real-boundary rows,* BHist *history,* hsame *transport,* Cont *continuation, and* Pkg *provenance.*

Conversely, every ledger-named source row in that carrier is readable by the corresponding $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer after the displayed transport through the public finite-window handoff.*

The exhaustion statement contains no quotient completion, Dedekind cut, trichotomy, host equality, chosen limit, or hidden precision schedule.

Proof. First use Definition 314.5. A downstream window read is sent to one of the displayed $\mathsf{StreamName}^{\uparrow}$ window rows, and the handoff provides no alternate stream carrier.

Next combine Definition 314.6 with Definition 314.8. The ledger boundary lists the dyadic observation, regular-tail, real-boundary, history, transport, continuation, package, and provenance rows, while the public export reuses those same rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

For the reverse direction, apply Definition 314.9. Each ledger-named row is repacked as a standard finite-window read along the displayed transport, so the corresponding $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer can read it without adding data.

Finally invoke Definition 314.10. The mature coverage theorem identifies the complete consumer boundary, so the two directions exhaust the same finite ledger rows and exclude quotient completion, Dedekind cuts, trichotomy, host equality, chosen limits, and hidden precision schedules. □

Theorem 314.12 (DyadicCompletion RegSeqRat window exhaustion)**.** *For any accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *finite-window packet, every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exposed by the public finite-window export factors through one of the ledger-named rows already listed by the packet: dyadic observation rows, scheduled* $\mathsf{StreamName}^{\uparrow}$ *window rows, regular-tail classifier rows, real-boundary rows,* hsame *transports,* Cont *routes, and* Pkg *provenance rows.*

Conversely, every ledger-named regular-tail or window row supplies the same $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read after the displayed finite-window standard bridge. This exhaustion exports no quotient Cauchy stream, Dedekind cut, host real limit, hidden precision schedule, or choice-selected limit.*

Proof. First apply Definition 314.5. It sends the scheduled finite read to the displayed $\mathsf{StreamName}^{\uparrow}$ window row, so the consumer has no alternate stream schedule outside the packet.

Next use Definition 314.4. The regular-tail classifier is transported along the packet's hsame and Cont rows, and the transport preserves exactly the ledger-named tail comparison row read by the $\mathsf{RegSeqRat}^{\uparrow}$ consumer.

Combine Definition 314.6 with Definition 314.8. These theorems identify the visible dyadic observation, real-boundary, history, transport, continuation, and package provenance rows exported to public $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

For the converse, apply Definition 314.9 and Definition 314.11. A ledger-named regular-tail or scheduled-window row is repacked through the same finite-window standard bridge, and the source exhaustion theorem says that no further source row is available to the consumer.

Thus the two readings have the same public $\mathsf{RegSeqRat}^{\uparrow}$ row boundary. The excluded quotient streams, cuts, host limits, hidden schedules, and selected limits are absent because none of the cited ledger, handoff, stability, export, or bridge rows carries them. □

Theorem 314.13 (DyadicCompletion terminal finite-window seal determinacy)**.** *Consider two* $\mathsf{DyadicCompletion}^{\uparrow}$ *finite-window packets whose dyadic observation row,* $\mathsf{StreamName}^{\uparrow}$ *window row, regular-tail row,* $\mathsf{Real}^{\uparrow}$ *boundary row, ledger row, provenance row, and downstream handoff route agree by the displayed* hsame*,* Cont*, and* Pkg *witnesses. Then every terminal* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through the same public finite-window row. No second terminal seal, hidden precision schedule, quotient completion, host equality, or choice-selected limit is exported by the DyadicCompletion boundary.*

Proof. First apply Definition 314.5. The common $\mathsf{StreamName}^{\uparrow}$ window row fixes the visible stream handoff for both packets, and the displayed hsame witness gives the same finite observation boundary.

Next combine Definition 314.8 with Definition 314.6. The regular-tail row, $\mathsf{Real}^{\uparrow}$ boundary row, ledger row, provenance row, and Pkg source row are therefore read from a single public packet surface, while the listed Cont routes give the same downstream handoff for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Now use Definition 314.9. The shared ledger and transport rows repack the two finite-window packets as the same standard public row, so transitivity of the displayed hsame comparisons identifies the row consumed by either terminal seal.

Finally invoke Definition 314.10. Mature coverage says that the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers listed above exhaust the terminal boundary; hence no further seal, precision schedule, quotient completion, host equality, or choice-selected limit can be exported outside the identified finite-window row. □

Theorem 314.14 (DyadicCompletion RealUp route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exposed by an accepted* $\mathsf{DyadicCompletion}^{\uparrow}$ *finite-window packet is exhausted by the same finite route through the dyadic observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-tail rows of chapter 301, the dyadic precision rows of chapter 291, the* $\mathsf{Real}^{\uparrow}$ *boundary of chapter 38, the precision ledger, the displayed* hsame *and* Cont *transports, the* Pkg *row, and the local* NameCert *provenance row.*

Consequently no DyadicCompletion$^{\uparrow}$ *terminal* Real$^{\uparrow}$ *seal can be read through a second route outside the listed stream-window, regular-tail, dyadic, real-boundary, ledger, transport, continuation, package, and provenance rows.*

Proof. Project the accepted packet through Definition 314.1. The carrier exposes the finite tuple of dyadic observations, stream-window names, regular-tail certificates, real-boundary handoff, precision ledger, history comparison, continuation, package source, and local provenance rows on which a DyadicCompletion$^{\uparrow}$ consumer may depend.

Next apply Definition 314.11. The ledger admits only the scheduled dyadic and StreamName$^{\uparrow}$ windows named by the packet, so the Real$^{\uparrow}$ seal cannot import an additional finite-window source. Apply Definition 314.12 to the regular-tail side; the RegSeqRat$^{\uparrow}$ read is forced through the same window, ledger, hsame, Cont, and Pkg route.

The sibling chapters provide the route endpoints. The stream-window rows come from chapter 126, the regular-tail classifier rows from chapter 301, the dyadic precision rows from chapter 291, and the terminal real-name boundary from chapter 38. These are precisely the rows already recorded by the local NameCert provenance surface.

Finally use Definition 314.13. Once the dyadic observation, stream-window, regular-tail, real-boundary, ledger, transport, continuation, package, and provenance rows agree, the terminal Real$^{\uparrow}$ seal has no alternate public finite-window row. Hence the displayed route exhausts every DyadicCompletion$^{\uparrow}$ real-seal read. □

Theorem 314.15 (DyadicCompletion regular-tail budget refinement)**.** *Let an accepted* DyadicCompletion$^{\uparrow}$ *finite-window packet carry a certified tail-radius seal whose regular-tail radius is refined along the* StreamName$^{\uparrow}$ *handoff row. Then the refined seal induces the same* RegSeqRat$^{\uparrow}$ *classifier window as the original packet and preserves the terminal* Real$^{\uparrow}$ *seal route exhausted in Definition 314.14.*

The induced budget witness is the one consumed by chapter 1053: it records the same dyadic observation rows, scheduled StreamName$^{\uparrow}$ *windows, regular-tail comparison rows, precision ledger entries, displayed* hsame *and* Cont *transports,* Pkg *provenance, and local* NameCert *source certificate, with only the tail-radius ledger made sharper.*

Proof. First apply Definition 314.4. Refining the tail-radius seal along the displayed hsame and Cont rows transports the regular-tail comparison data without changing the RegSeqRat$^{\uparrow}$ classifier window read from the packet.

Next use Definition 314.5. The sharpened radius is attached to the same scheduled StreamName$^{\uparrow}$ handoff row, so the stream-window source and the regular-tail classifier still share the ledger row visible to the downstream consumer.

Finally invoke Definition 314.14. Since the dyadic observation, stream-window, regular-tail, real-boundary, ledger, transport, continuation, package, and provenance rows are the same rows after refinement, the terminal Real$^{\uparrow}$ seal is forced through the same finite route. The only exported extra datum is the sharper budget witness used by chapter 1053. □

Closure status for NameCert$_{\text{DyadicCompletion}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{DyadicCompletion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 314.10 identifies the finite dyadic observation, stream-window, regular-tail, real-boundary, ledger, transport, continuation, package, public export, and standard bridge rows as the mature constructive RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ consumer boundary over the listed BEDC source certificates.

Constructive story: A DyadicCompletion$^{\uparrow}$ packet is generated by scheduled dyadic observations, certified stream windows, regular-tail comparison rows, a real-name handoff boundary, and a precision ledger that records every consumed schedule and tail bound.

Formal verification: FormalStatus(NameCert$_{\text{DyadicCompletion}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicCompletionUp.DyadicCompletionPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close the Lean namespace, a complete real construction, quotient equality of Cauchy streams, Dedekind real equivalence, host real limits, choice-selected limits, or completeness.

Upgrade path: The paper-side mature consumer coverage is complete; remaining work is Lean-side formalization and bridge checking for the same DyadicCompletion$^{\uparrow}$ packet.

Chapter 315

A Concrete Naming Certificate for $\mathsf{FastCauchySeq}^{\uparrow}$

The $\mathsf{FastCauchySeq}^{\uparrow}$ interface names explicit-rate Cauchy sequence packets over dyadic-rational observations. Bishop and Bridges treat a displayed Cauchy rate as constructive data; the BEDC packet records the finite ledger version of that data, connecting $\mathsf{CauchyModulus}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ without quotient completion or a selected limit.

Theorem 315.1 (FastCauchySeq sibling source lattice)**.** *Every accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet reads its scheduled observation stream through chapter 126, its threshold table through chapter 292, and its endpoint/radius ledger through chapter 291.*

The same packet sends its forgetful regularity handoff through chapter 301 and its consumer seal through chapter 38. This lattice introduces no quotient real, no selected limit, and no ambient completeness row.

Proof. Read the carrier tuple of Definition 315.2. Its stream coordinate is the scheduled $\mathsf{StreamName}^{\uparrow}$ observation surface, its modulus coordinate is the $\mathsf{CauchyModulus}^{\uparrow}$ threshold table, and its dyadic rows are the endpoint and radius ledger supplied by $\mathsf{DyadicRatCore}^{\uparrow}$.

For the outgoing boundaries, apply Definition 315.5 to remove only the local certificate row while keeping the finite regularity data. Then apply Definition 315.6 to restrict the visible surface to the rows a $\mathsf{Real}^{\uparrow}$ consumer may read.

Both passages are projections from the same explicit-rate carrier. Since the carrier has no completed-real coordinate, quotient-real coordinate, selected-limit coordinate, or ambient completeness row, none is introduced by recording the sibling source lattice. □

315.1 Explicit-rate carrier and classifier

Definition 315.2 (FastCauchySeq carrier)**.** *A* $\mathsf{FastCauchySeq}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (s, \mu, E, R, L, T, P, N)$$

with the following visible rows:

1. *s is a scheduled* $\mathsf{StreamName}^{\uparrow}$ *dyadic-rational observation stream;*
2. *μ is a* $\mathsf{CauchyModulus}^{\uparrow}$ *precision-threshold table, displaying for each requested precision row a finite index bound after which the stream is required to be close;*

3. *E lists the dyadic endpoint and dyadic radius rows read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *at the visible precisions;*
4. *R is the late-pair comparison ledger recording that any two displayed observations beyond the threshold named by μ fall inside the displayed dyadic radius row for that precision;*
5. *L records the componentwise* hsame *transports for the stream, threshold, endpoint, radius, and comparison rows;*
6. *T records the displayed* Cont *windows through which the finite reads of s, μ, E, and R are consumed;*
7. *P is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$ *row for this packet.*

The carrier is the finite explicit-rate sequence surface only. It contains no completed real, no quotient equality of Cauchy names, and no selected limiting value.

Definition 315.3 (FastCauchySeq classifier)**.** *For accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packets F and F', the classifier* $\mathsf{FastCauchySeqClassifier}(F, F')$ *holds when the scheduled stream rows, Cauchy-modulus threshold rows, dyadic endpoint and radius rows, late-pair comparison ledgers,* hsame *transports,* Cont *windows,* Pkg *provenance, and local* NameCert *rows transport componentwise through the displayed classifiers of their source chapters.*

The classifier identifies two packets only at the level of the visible explicit-rate ledger. It does not assert equality of quotient real numbers, does not choose a limit, and does not add a completeness principle to $\mathsf{Real}^{\uparrow}$*.*

Theorem 315.4 (FastCauchySeq NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$ *obligation surface has five finite rows:*

1. *carrier: the* BHist *packet of Definition 315.2;*
2. *classifier: the componentwise classifier of Definition 315.3;*
3. *stability: the* hsame *and* Cont *rows transporting the scheduled stream, threshold table, dyadic endpoint and radius rows, and late-pair comparison ledger;*
4. *handoff: the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity window obtained by forgetting the local row N and retaining the stream, modulus, dyadic, and comparison rows;*
5. *seal boundary: the* $\mathsf{Real}^{\uparrow}$ *consumer boundary that can read only the displayed stream, modulus, dyadic, and provenance rows.*

These rows are exactly the explicit-rate naming certificate for $\mathsf{FastCauchySeq}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FastCauchySeqUp.FastCauchySeqPacket semantic name certificate`

Proof. Project the carrier coordinates from Definition 315.2. The first two rows expose the scheduled dyadic observation stream and the precision-threshold table, while the dyadic and late-pair rows expose the finite comparison data at the requested precisions.

The stability row is obtained by transporting each projected coordinate through the displayed hsame and Cont routes recorded by L and T. Because each route is componentwise, the transported packet has the same finite obligation shape.

The final two rows are projections of the same packet through the named source boundaries. Forgetting N gives the $\mathsf{RegSeqRat}^{\uparrow}$ regularity window, and restricting to the $\mathsf{Real}^{\uparrow}$ seal boundary removes every coordinate not listed in the carrier, classifier, stability, and provenance rows. □

Theorem 315.5 (FastCauchySeq forgetful handoff to RegSeqRat)**.** *Every accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet forgets to a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity window. The forgotten packet keeps the scheduled* $\mathsf{StreamName}^{\uparrow}$ *dyadic-rational observation stream, the visible* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, the dyadic endpoint and radius rows from* $\mathsf{DyadicRatCore}^{\uparrow}$*, the late-pair comparison ledger, and the provenance over* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$*.*

The handoff drops only the local $\mathsf{FastCauchySeq}^{\uparrow}$ *certificate row. It does not drop the finite regularity evidence required by* $\mathsf{RegSeqRat}^{\uparrow}$*, and it does not produce a quotient real or a selected limit.*

Proof. Start from the carrier tuple of Definition 315.2. Remove the local row N and keep the stream, modulus, dyadic endpoint, dyadic radius, comparison, transport, window, and provenance coordinates.

The $\mathsf{CauchyModulus}^{\uparrow}$ rows close the finite threshold requests: for each displayed precision, the row supplies the index bound used by the late-pair comparison ledger. The ledger then shows that the visible stream values beyond that bound lie inside the corresponding dyadic radius window.

These are precisely the finite regularity rows consumed by $\mathsf{RegSeqRat}^{\uparrow}$. Repacking them yields the regular sequence window, while the absence of any completed-real coordinate prevents the handoff from selecting a limit or asserting quotient equality. □

Lean4 ✓ `BEDC.Derived.FastCauchySeqUp.FastCauchySeqPacket regseqrat forgetful handoff`

Theorem 315.6 (FastCauchySeq Real seal consumer boundary)**.** *A* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet may read only the scheduled observation stream, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, the dyadic endpoint and radius rows, the late-pair comparison ledger, and the provenance rows named by* Pkg*.*

The consumer receives no quotient real, no equality of Cauchy names, no completed metric object, and no selected limit. Its surface is the finite explicit-rate name carried by the stream, modulus, dyadic, comparison, hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Project the accepted packet through Definition 315.2 and Definition 315.3. The only rows available to a downstream consumer are the stream, modulus, dyadic, comparison, transport, window, provenance, and local certificate rows.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ forgetful handoff of Definition 315.5. It presents the same finite regularity window to the real-name boundary and leaves behind the local certificate row.

Finally restrict along the $\mathsf{Real}^{\uparrow}$ classifier transport boundary. That boundary consumes the displayed regular sequence data and provenance only. Since the carrier has no quotient-real coordinate, no completed metric coordinate, and no selected limit coordinate, none can be read by a $\mathsf{Real}^{\uparrow}$ consumer. □

Lean4 ✓ `BEDC.Derived.FastCauchySeqUp.FastCauchySeqPacket real seal consumer boundary`

Theorem 315.7 (FastCauchySeq tail-modulus Real boundary)**.** *For an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$ *consumer read factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff together with the visible* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, dyadic endpoint and radius rows, late-pair comparison ledger, componentwise* hsame *transports, displayed* Cont *windows, and* Pkg *provenance named by the carrier.*

The factorization keeps the local NameCert *row as the certificate for the finite explicit-rate surface and exposes no quotient real equality, no selected limit, and no completed metric object. Thus the* $\mathsf{Real}^{\uparrow}$ *boundary is the finite tail-modulus read of the scheduled stream, not a completion or limit-selection principle.*

Proof. Unpack Definition 315.2. The packet contains the scheduled stream, the explicit $\mathsf{CauchyModulus}^{\uparrow}$ threshold table, dyadic endpoint and radius rows, the late-pair comparison ledger, hsame transport rows, Cont windows, Pkg provenance, and the local NameCert row.

Apply Definition 315.5. This keeps the stream, modulus, dyadic, comparison, transport, window, and provenance rows as the finite regular-rational window and forgets only the local $\mathsf{FastCauchySeq}^{\uparrow}$ certificate row from the outgoing $\mathsf{RegSeqRat}^{\uparrow}$ packet.

Now restrict that window by Definition 315.6. The $\mathsf{Real}^{\uparrow}$ consumer may inspect exactly the scheduled observations, threshold rows, dyadic endpoint-radius data, late-pair ledger, transport, window, provenance, and naming rows already visible in the carrier.

Because the carrier contains no quotient-real coordinate, no selected limit, and no completed metric object, the resulting Real-facing read cannot acquire any such row after the handoff. □

Theorem 315.8 (FastCauchySeq public explicit-rate export)**.** *The carrier and classifier of Definition 315.2 and Definition 315.3, the* $\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$ *obligation surface of Definition 315.4, the* $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff of Definition 315.5, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 315.6 jointly export the reusable public* $\mathsf{FastCauchySeq}^{\uparrow}$ *explicit-rate interface.*

The exported interface consists exactly of the scheduled finite stream, $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, dyadic endpoint and radius rows, late-pair ledger, componentwise* hsame *transports, visible* Cont *windows,* Pkg *provenance, and* NameCert *row. It exposes those rows as the common surface for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers without adding quotient real equality, a completed metric object, or a selected limit.*

Lean4 ✓ `BEDC.Derived.FastCauchySeqUp.FastCauchySeqPacket public explicit rate export`

Proof. Project the packet through Definition 315.2. This exposes the scheduled stream, explicit threshold table, dyadic endpoint and radius rows, late-pair ledger, transport table, continuation windows, package provenance, and local name-certificate row.

Use the classifier of Definition 315.3 to make the same projected rows invariant under the displayed componentwise transports. The obligation surface of Definition 315.4 then repacks those projections as the finite $\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$ surface.

The $\mathsf{RegSeqRat}^{\uparrow}$ side is exactly the forgetful handoff of Definition 315.5. It retains the stream, threshold, dyadic, comparison, transport, window, and provenance rows and forgets only the local certificate row as a local certificate.

The $\mathsf{Real}^{\uparrow}$ side is exactly the consumer boundary of Definition 315.6. Since that boundary can read only the same finite explicit-rate rows, the assembled public interface exports no coordinate for quotient equality, completed metrics, or limit selection. □

Theorem 315.9 (FastCauchySeq standard explicit-rate bridge)**.** *The public explicit-rate export of Definition 315.8 is precisely the standard constructive explicit-rate Cauchy-sequence schema over scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, the late-pair comparison ledger, visible* hsame *and* Cont *transport,* Pkg *provenance, and the local* NameCert *row.*

In one direction, every public $\mathsf{FastCauchySeq}^{\uparrow}$ *row supplies one standard explicit-rate datum: a scheduled observation stream, an explicit threshold assignment for each tolerance row, dyadic*

endpoint and radius evidence for the observed window, and a late-pair comparison certificate whose transport and provenance are visible. In the reverse direction, every standard datum accepted at this paper site is one of those public $\mathsf{FastCauchySeq}^{\uparrow}$ *rows, read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff and the* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Thus the bridge adds no quotient real, completed metric object, selected limit, or hidden threshold selector to the public export.

Proof. Project the carrier and classifier of Definition 315.2 and Definition 315.3. The projection exposes exactly the scheduled stream, explicit threshold table, dyadic endpoint and radius rows, late-pair comparison ledger, componentwise transport rows, continuation windows, package provenance, and local name-certificate row named by the public export.

For the forward direction, apply the $\mathsf{RegSeqRat}^{\uparrow}$ forgetful handoff of Definition 315.5. It reads the same finite stream, threshold, dyadic, comparison, transport, window, and provenance rows as a regular-rational explicit-rate datum, while the Definition 315.6 records the matching $\mathsf{Real}^{\uparrow}$ consumer face.

For the reverse direction, inspect what is accepted at this site as a standard explicit-rate datum. The only available coordinates are the rows exported by Definition 315.8; the classifier identifies representatives only through visible hsame and Cont transport, and Pkg plus NameCert provide provenance rather than extra analytic structure.

Since neither the carrier nor the two consumer boundaries contain a coordinate for quotient-real equality, a completed metric object, a selected limit, or a hidden threshold selector, the two readings are the same finite BEDC explicit-rate schema. □

Theorem 315.10 (FastCauchySeq modulus precision-window factorization)**.** *Let F be an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet, and let D be a displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *row selecting a finite precision window for F. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read and every* $\mathsf{Real}^{\uparrow}$ *consumer read over D factors through the same visible finite rows of F: the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row selected by D, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows read at that precision, the late-pair ledger for the selected window, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

Conversely, each displayed threshold, endpoint, radius, late-pair, transport, continuation, provenance, and name row indexed by D appears in both the $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff and the* $\mathsf{Real}^{\uparrow}$ *seal boundary. Thus the finite-window read over D is a factorization of explicit Cauchy-rate data through the public* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet, not a read through quotient real equality, a completed metric object, a selected limit, or a hidden threshold selector.*

Proof. Project the accepted packet through Definition 315.2. The displayed row D selects only a finite precision window, so the carrier projection exposes exactly the corresponding threshold row, dyadic endpoint and radius rows, late-pair ledger, transport rows, continuation routes, provenance row, and local naming row.

For the $\mathsf{RegSeqRat}^{\uparrow}$ read, apply Definition 315.5. The handoff forgets only the local $\mathsf{FastCauchySeq}^{\uparrow}$ certificate row as a local certificate and retains the selected stream, threshold, dyadic, comparison, transport, window, and provenance coordinates indexed by D.

For the $\mathsf{Real}^{\uparrow}$ read, use Definition 315.6 together with Definition 315.7. These boundaries allow a real-name consumer to inspect the same displayed explicit-rate rows and no coordinate outside the finite tail-modulus surface.

Finally compare with Definition 315.9. The bridge identifies the public explicit-rate schema with the visible $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, late-pair, hsame, Cont, Pkg, and NameCert rows, so every row selected by D occurs in both consumer faces and every such consumer face factors

through those rows. Since the carrier and the two boundaries contain no quotient-real equality, completed metric object, selected limit, or hidden threshold selector, none is introduced by the factorization. □

Theorem 315.11 (FastCauchySeq synchronized precision window)**.** *Let F and F' be accepted* FastCauchySeq$^{\uparrow}$ *packets, and let D and D' be displayed* DyadicPrecision$^{\uparrow}$ *rows selecting finite precision windows for them. Suppose the classifier relates F and F', and that D and D' select* hsame *precision windows whose transported rows agree componentwise: the* CauchyModulus$^{\uparrow}$ *threshold row, the* DyadicRatCore$^{\uparrow}$ *endpoint and radius rows, the late-pair ledger, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows.*

Then every RegSeqRat$^{\uparrow}$ *handoff read and every* Real$^{\uparrow}$ *consumer-boundary read for F over D and for F' over D' factors through one synchronized finite precision packet carrying exactly those transported threshold, endpoint, radius, late-pair,* Cont*,* hsame*,* Pkg*, and* NameCert *rows. The synchronized packet asserts no quotient real equality, selects no limit, constructs no completed metric object, and contains no hidden threshold selector.*

Proof. Unpack the carrier and classifier of Definition 315.2 and Definition 315.3. The two accepted packets expose only scheduled stream rows, threshold rows, endpoint-radius rows, late-pair ledgers, transport rows, continuation windows, package provenance, and local naming rows. The classifier compares those coordinates componentwise.

Apply Definition 315.10 to F and D. It factors every RegSeqRat$^{\uparrow}$ handoff and every Real$^{\uparrow}$ boundary read through the finite precision window selected by D.

Apply the same theorem to F' and D'. The hypotheses identify the selected threshold, endpoint, radius, late-pair, Cont, hsame, Pkg, and NameCert rows with the transported rows from the first factorization, so the two finite windows are the same synchronized packet.

Finally use Definition 315.5 and Definition 315.6. These are the only handoff and seal-boundary rows cited by the factorization, so both consumers read the synchronized finite packet and no quotient equality, selected limit, completed metric object, or hidden threshold selector is introduced. □

Theorem 315.12 (FastCauchySeq tail-window overlap exhaustion)**.** *Let F be an accepted* FastCauchySeq$^{\uparrow}$ *packet, and let D_0 and D_1 be displayed* DyadicPrecision$^{\uparrow}$ *rows selecting two finite precision windows inside F. Suppose the two windows have a shared finite tail subwindow whose transported rows agree: the common* CauchyModulus$^{\uparrow}$ *threshold row, the* DyadicRatCore$^{\uparrow}$ *endpoint and radius rows, the late-pair ledger, the displayed* hsame *and* Cont *routes, the* Pkg *provenance row, and the local* NameCert *row all occur as visible rows of the same packet.*

Then every RegSeqRat$^{\uparrow}$ *handoff read and every* Real$^{\uparrow}$ *consumer-boundary read on the overlap factors through exactly that common finite tail packet. The factorization uses only the displayed threshold row, endpoint and radius rows, late-pair ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already visible in F; it asserts no quotient real equality, selects no limit, builds no completed metric object, and invokes no hidden threshold selector.*

Proof. Apply Definition 315.10 to F with the window D_0. This presents each RegSeqRat$^{\uparrow}$ handoff and each Real$^{\uparrow}$ boundary read over D_0 through the finite packet determined by its visible threshold, dyadic, late-pair, transport, provenance, and naming rows.

Apply the same factorization theorem to F with the window D_1. On the shared tail subwindow, the two factorized packets expose the same threshold row, the same endpoint and radius rows, the same late-pair ledger, the same hsame and Cont routes, and the same Pkg and NameCert rows.

Use Definition 315.11 only for these common transported rows. Since both windows lie inside the same accepted packet F, the synchronization does not add a second source packet; it identifies the overlap as the unique visible tail packet already carried by F.

The consumer restrictions are exactly those of Definition 315.5 and Definition 315.6. They read the common finite rows and no quotient real equality, selected limit, completed metric object, or hidden threshold selector. □

Theorem 315.13 (FastCauchySeq tail-stationary precision-window determinacy)**.** *Let F and F' be accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packets, and let D and D' be displayed precision rows for finite consumer windows. Suppose the selected* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows agree by* hsame*, the selected* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows agree by* hsame*, the late-pair ledgers are joined by the same displayed* Cont *routes, and the* Pkg *and* NameCert *rows carry the same visible provenance.*

Then every $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and every* $\mathsf{Real}^{\uparrow}$ *consumer-boundary read over those windows factors through one stationary finite packet carrying the common threshold, endpoint, radius, late-pair,* Cont*,* hsame*,* Pkg*, and* NameCert *rows. This stationary packet introduces no quotient real equality, selected limit, completed metric object, or hidden threshold selector.*

Proof. Project the two accepted packets through the explicit-rate carrier. The only rows visible at the selected windows are the modulus threshold, dyadic endpoint and radius, late-pair ledger, continuation route, provenance, and local naming rows.

Apply Definition 315.10 to both windows. Each factorization presents the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ consumer-boundary read through its selected finite packet.

Use Definition 315.11 to identify the transported threshold, endpoint, radius, late-pair, Cont, hsame, Pkg, and NameCert rows. The hypotheses make these rows stationary across the two tail displays, so both finite packets are the same visible packet.

Finally apply Definition 315.5 and Definition 315.6. These consumer faces forget only local certificate packaging and read only the displayed boundary rows, hence the shared stationary packet adds no quotient real equality, selected limit, completed metric object, or hidden threshold selector. □

Theorem 315.14 (FastCauchySeq prefix-window eventual lock)**.** *Let F and F' be accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packets, and let D and D' be displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *rows. Suppose the classifier relates F and F', and suppose the selected precision rows transport the same* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius rows, the same late-pair ledger, the same displayed* Cont *route, the same* Pkg *provenance row, and the same local* NameCert *row.*

Then every $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read and every* $\mathsf{Real}^{\uparrow}$ *seal-boundary read for F over D and for F' over D' factors through one shared finite prefix-window packet carrying exactly those transported rows. The shared packet introduces no quotient real equality, no selected limit, no completed metric object, and no hidden threshold selector.*

Proof. Unpack Definition 315.2 and Definition 315.3. The accepted packets expose only the scheduled stream row, modulus threshold row, dyadic endpoint-radius rows, late-pair ledger, transport route, continuation window, package provenance, and local naming row, and the classifier compares these displayed rows.

Apply Definition 315.10 to each side. It factors the $\mathsf{RegSeqRat}^{\uparrow}$ handoff reads and the $\mathsf{Real}^{\uparrow}$ seal-boundary reads through the precision windows selected by D and D'.

Use Definition 315.11 to identify the transported finite rows. The common modulus threshold, endpoint-radius data, late-pair ledger, Cont route, Pkg provenance, and local NameCert row therefore assemble one shared prefix-window packet.

Finally cite Definition 315.5 and Definition 315.6. These consumer restrictions are exactly the handoff and seal-boundary interfaces used above, so no quotient real equality, selected limit, completed metric object, or hidden threshold selector is added. □

Theorem 315.15 (FastCauchySeq tail-cofinal RegSeqRat Real boundary)**.** *Let F be an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet, and let $D \leq D'$ be displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *windows whose selected tail rows are cofinal in the same* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold ledger. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read and every* $\mathsf{Real}^{\uparrow}$ *seal-boundary read over D' factors through the same visible threshold, dyadic endpoint and radius, late-pair,* hsame*,* Cont*,* Pkg*, and* NameCert *rows as the corresponding read over D.*

The common factor is the displayed cofinal tail packet of F. It asserts no quotient real equality, selects no limit, builds no completed metric object, and invokes no hidden threshold selector.

Proof. Project F through Definition 315.2. The accepted carrier exposes the scheduled stream row, the $\mathsf{CauchyModulus}^{\uparrow}$ threshold ledger, dyadic endpoint-radius rows, late-pair rows, hsame and Cont transport, Pkg provenance, and local NameCert rows.

Apply Definition 315.10 to the window D and again to the window D'. Each read is thereby presented through its selected finite precision packet rather than through any completed real object.

Use Definition 315.12 to pass to the shared cofinal tail and then Definition 315.14 to identify the common finite packet. The cofinality hypothesis supplies the same threshold ledger and the same visible endpoint-radius, late-pair, hsame, Cont, Pkg, and NameCert rows for both displays.

Finally apply Definition 315.5 and Definition 315.6. These two consumer boundaries read only the displayed cofinal tail packet, so they add no quotient real equality, selected limit, completed metric object, or hidden threshold selector. □

Theorem 315.16 (FastCauchySeq prefix-modulus Real seal factorization)**.** *Let F and F' be accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packets with a shared prefix-window eventual lock. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and every* $\mathsf{Real}^{\uparrow}$ *seal-boundary read over that locked prefix factors through one prefix-modulus packet.*

The prefix-modulus packet contains exactly the transported $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius rows, the late-pair ledger, the visible* hsame *and* Cont *routes, the* Pkg *provenance, and the local* NameCert *row. It contains no quotient real equality, selected limit, completed metric object, hidden threshold selector, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 315.10 to both accepted packets. Each side presents its $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal-boundary read through a finite packet carrying the selected modulus threshold, dyadic endpoint-radius rows, late-pair ledger, transport, provenance, and local naming rows.

Use Definition 315.11 to identify the transported rows. The synchronized window equates the visible threshold, endpoint-radius, late-pair, hsame, Cont, Pkg, and NameCert coordinates without adding a second threshold selector.

Now apply Definition 315.14. The lock collapses the synchronized selected prefixes to one shared prefix-modulus packet, so both finite displays feed the same consumer-readable data.

Finish with Definition 315.5 and Definition 315.6. These two consumer boundaries read only the displayed prefix-modulus packet, hence no quotient real equality, selected limit, completed metric object, hidden threshold selector, or $\mathsf{Real}^{\uparrow}$ completeness row is introduced. □

Theorem 315.17 (FastCauchySeq tail-lock Real boundary)**.** *Let F be an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packet whose accepted windows are cofinal in the displayed tail rows, and suppose the corresponding*

prefix windows eventually lock. Then every $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and every* $\mathsf{Real}^{\uparrow}$ *seal-boundary read factors through one common finite* BHist *packet.*

That packet carries exactly the $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius rows, the late-pair ledger, the visible* hsame *and* Cont *routes, the* Pkg *provenance, and the local* NameCert *row. The factorization introduces no quotient real equality, selected limit, completed metric object, hidden threshold selector, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted packet through Definition 315.2. The visible rows are the scheduled stream, the modulus threshold ledger, the dyadic endpoint-radius data, late-pair comparisons, transport routes, provenance, and local name certificate.

Apply Definition 315.12. Cofinality of the accepted tail windows forces every later $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal-boundary read to pass through the same finite tail display, rather than through an ambient completed real object.

Next use Definition 315.14. The eventual lock identifies the finite prefix packet that all sufficiently late displayed windows share, including the threshold, endpoint-radius, late-pair, hsame, Cont, Pkg, and NameCert coordinates.

Finally apply Definition 315.16. Its prefix-modulus packet is therefore the common BHist packet through which both the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal-boundary consumer read. Since the cited factorization reads only that finite packet, it adds no quotient real equality, selected limit, completed metric object, hidden threshold selector, or $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 315.18 (FastCauchySeq explicit-rate RegSeqRat Real handoff)**.** *Let F be an accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *prefix-modulus packet whose public explicit-rate surface is displayed over a* $\mathsf{StreamName}^{\uparrow}$ *schedule, a* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, late-pair comparison ledgers,* hsame *and* Cont *transport,* Pkg *provenance, and local* NameCert *rows. Then the same finite packet determines exactly both downstream reads: the* $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff and the* $\mathsf{Real}^{\uparrow}$ *seal consumer boundary.*

The handoff exports precisely those visible rows and no further coordinate. In particular it exports no quotient real equality, hidden threshold selector, completed metric object, selected limit, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the public explicit-rate surface of Definition 315.8. It presents the accepted packet through the displayed schedule rows, modulus thresholds, dyadic endpoint-radius data, late-pair comparisons, transport routes, provenance, and naming rows.

Next factor the consumer-facing windows by Definition 315.10 and Definition 315.11. These two results identify the common finite packet that both downstream surfaces read, so no second selector or hidden completion coordinate is introduced.

Finally read the two consumers through Definition 315.5 and Definition 315.6. The first theorem identifies the regular-rational forgetful source rows, and the second theorem identifies the real-seal source rows. Since both are applied to the same prefix-modulus packet, the finite BEDC handoff is exactly the shared source surface named in the statement. □

Theorem 315.19 (FastCauchySeq cofinal-tail Real-seal coherence)**.** *Let F_0 and F_1 be accepted* $\mathsf{FastCauchySeq}^{\uparrow}$ *packets whose explicit-rate streams admit a common cofinal tail. Suppose their synchronized precision windows agree on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows, late-pair comparison rows, transport rows,* Pkg *provenance, and* NameCert *rows. Then the* $\mathsf{Real}^{\uparrow}$ *seal source surfaces obtained through the* $\mathsf{RegSeqRat}^{\uparrow}$ *forgetful handoff agree on that cofinal tail.*

The agreement is only the displayed finite consumer boundary: the shared tail contains the selected $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, dyadic endpoint and radius rows, comparison rows,* hsame

and Cont *routes,* Pkg *provenance, and local* NameCert *provenance. It contains no quotient real equality, selected limit, completed metric object, hidden threshold selector, or* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project both packets by Definition 315.10. This presents each consumer read through a finite modulus precision window rather than through any completed real object.

Synchronize the two windows with Definition 315.11. The hypothesis identifies the displayed endpoint, radius, comparison, transport, package, and naming rows in the synchronized surface.

Use Definition 315.14 to pass from the synchronized prefixes to the stable cofinal tail rows. Then apply Definition 315.15 to read that tail as the same $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ consumer boundary.

Finally invoke Definition 315.16. The $\mathsf{Real}^{\uparrow}$ seal factors through the shared prefix-modulus packet on the cofinal tail, so both accepted packets expose the same finite BEDC source surface and no quotient equality, selected limit, completion object, hidden selector, or completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$.

Constructive story: A fast-Cauchy-sequence packet is generated from a scheduled dyadic-rational observation stream, visible Cauchy-modulus threshold rows, dyadic endpoint and radius rows, late-pair comparison ledgers, componentwise hsame and Cont transports, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 315.8, Definition 315.9, Definition 315.10, and Definition 315.11 bind the public export, standard explicit-rate bridge, modulus precision-window factorization, and synchronized finite-window consumer surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastCauchySeq}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FastCauchySeqUp.FastCauchySeqPacket\_public\_explicit\_rate\_export`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close completeness, quotient equality of Cauchy names, selected limits, or standard bridge data.

Upgrade path: The next step is Lean formalization and checking of the explicit-rate bridge.

Chapter 316

A Concrete Naming Certificate for $\mathsf{SignedDigitStream}^{\uparrow}$

The $\mathsf{SignedDigitStream}^{\uparrow}$ interface names finite-window signed-digit stream presentations over dyadic endpoint rows. Following Bishop–Bridges constructive analysis, the packet records only observed signed digits, normalization carries, endpoint rows, and stream provenance; it does not form a quotient of infinite streams or appeal to host real equality.

The signed-digit finite-window packet consumes the schedule surface of chapter 126, the endpoint normalization rows of chapter 291, and the downstream real-name handoff through chapter 301 and chapter 38.

316.1 Finite signed-digit windows

Definition 316.1 (SignedDigitStream carrier)**.** *A* $\mathsf{SignedDigitStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, N, C, P, E, L)$$

with the following visible rows:

1. *D is a finite list of signed dyadic digit rows, each digit drawn from the displayed signed finite alphabet and indexed by a unary position row;*
2. *N is the unary observation schedule supplied by* $\mathsf{StreamName}^{\uparrow}$*;*
3. *C is the carry and normalization ledger for every adjacent digit pair visible in the window;*
4. *P is the* $\mathsf{StreamName}^{\uparrow}$ *provenance row naming the scheduled stream requests and the* Pkg *routes used to read them;*
5. *E lists the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows obtained by normalizing each displayed finite prefix;*
6. *L records that all hidden data lies outside the requested finite window.*

The carrier accepts only packets whose visible prefixes normalize to the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows through the carry ledger C. It contains no infinite stream object outside the named schedule, no quotient of presentations, and no completed* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Definition 316.2 (SignedDigitStream classifier)**.** *For accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *carriers S and T, the classifier* $\mathsf{SignedDigitStreamClassifier}(S, T)$ *holds when every shared finite observation window normalizes to classifier-equal dyadic endpoints and the matching carry ledgers transport componentwise. Concretely, for each unary position read by both* $\mathsf{StreamName}^{\uparrow}$ *schedules, the normalized*

DyadicRatCore$^{\uparrow}$ *endpoint rows agree by the* DyadicRatCore$^{\uparrow}$ *classifier, the digit rows are related by the displayed* hsame *routes before normalization, and the carry ledger repacks to the same endpoint read.*

The classifier compares finite normalized observations, not infinite tails. A classified pair therefore exposes agreement of all windows actually requested by the schedules and supplies no host real equality, choice of a global stream representative, or quotient equality of digit streams.

Theorem 316.3 (SignedDigitStream sibling route through scheduled dyadic endpoints)**.** *Every accepted* SignedDigitStream$^{\uparrow}$ *packet factors through the concrete-instance sibling route*

$$\textit{chapter } 126 \to \textit{chapter } 291 \to \textit{chapter } 301 \to \textit{chapter } 363 \to \textit{chapter } 38$$

before any Real$^{\uparrow}$ *seal consumer can read it. The route reads the* StreamName$^{\uparrow}$ *schedule and provenance rows, the normalized* DyadicRatCore$^{\uparrow}$ *endpoint rows, the* RegSeqRat$^{\uparrow}$ *handoff surface, the signed-digit real finite-window boundary, and then the* Real$^{\uparrow}$ *seal rows.*

The route exports no quotient stream equality, selected limit, host real equality, or ambient completeness datum.

Proof. Project the carrier rows (D, N, C, P, E, L) of Definition 316.1. The rows N and P are exactly the finite-window schedule and provenance surface supplied by StreamName$^{\uparrow}$, so every downstream read first passes through chapter 126.

The row E is the displayed list of normalized endpoint rows obtained from the finite prefixes and the carry ledger C. These are DyadicRatCore$^{\uparrow}$ rows, so the next visible dependency is chapter 291; the digit and carry data only justify those endpoint rows and do not create an independent completed endpoint.

Use Definition 316.9 to expose the regular rational consumer surface. It reads only scheduled normalized dyadic endpoints and their ledgers, giving the RegSeqRat$^{\uparrow}$ stage. Then use Definition 316.10 to identify the finite-window seal boundary that passes through the signed-digit real surface before reaching the Real$^{\uparrow}$ seal rows.

Every object read along the route is one of the six projected carrier rows or one of the downstream rows named by those two handoff theorems. Since none of those rows contains a quotient stream equality, selected limit, host real equality, or ambient completeness datum, the sibling route exhausts the visible dependency surface before any Real$^{\uparrow}$ seal consumer can read the packet. □

Theorem 316.4 (SignedDigitStream finite-window transport)**.** *Let S be an accepted* SignedDigitStream$^{\uparrow}$ *carrier. Suppose T is obtained by transporting the signed digit rows, unary schedule rows, carry ledger,* StreamName$^{\uparrow}$ *provenance, normalized* DyadicRatCore$^{\uparrow}$ *endpoint rows, and hidden-tail ledger of S through displayed* hsame, Cont, Pkg, *and* NameCert *routes. If every transported prefix normalizes to the transported endpoint row with the transported carry ledger, then T is an accepted* SignedDigitStream$^{\uparrow}$ *carrier and* SignedDigitStreamClassifier(S, T) *holds on each shared finite window.*

The transport preserves only the finite signed-digit observation surface and introduces no quotient stream, ambient real equality, or completed real endpoint.

```
Lean4 ✓  BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket window transport
  ↪ variant  BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamWindowPacket window transport
```

Proof. Project the carrier of Definition 316.1. The digit rows, unary schedule, carry ledger, provenance, endpoint rows, and hidden-tail ledger are the only coordinates available for transport.

Move each coordinate along the displayed hsame and Cont routes. The hypothesis says that prefix normalization is preserved after the move, so every visible prefix of T still has the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row.

Apply the classifier of Definition 316.2 on a shared finite window. The endpoint classifiers come from the transported $\mathsf{DyadicRatCore}^{\uparrow}$ rows, and the carry ledgers match by the componentwise ledger transport. Since no extra coordinate is created, the transported packet exports exactly the same finite observation surface. □

Theorem 316.5 (SignedDigitStream prefix truncation stability)**.** *Let $S = (D, N, C, P, E, L)$ be an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *carrier, and let $N_{\leq k}$ be a displayed finite prefix of its unary* $\mathsf{StreamName}^{\uparrow}$ *schedule. The restricted tuple*

$$S_{\leq k} = (D_{\leq k}, N_{\leq k}, C_{\leq k}, P_{\leq k}, E_{\leq k}, L_{\leq k})$$

repacks as an accepted $\mathsf{SignedDigitStream}^{\uparrow}$ *carrier whenever each restricted digit, schedule, carry, endpoint, provenance, and ledger row is the displayed prefix row inherited from S.*

The classifier on the restricted packet is exactly the restriction of SignedDigitStreamClassifier *to windows contained in $N_{\leq k}$. Its* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal boundary are the corresponding finite prefix reads named by Definition 316.9 and Definition 316.10. The restriction introduces no quotient tail, selected limit, host real equality, completeness datum, or bridge row.*

Proof. Project the six carrier rows of Definition 316.1. The displayed prefix $N_{\leq k}$ selects exactly the digit rows $D_{\leq k}$, carry ledger $C_{\leq k}$, provenance row $P_{\leq k}$, endpoint rows $E_{\leq k}$, and hidden-tail ledger $L_{\leq k}$ already present in S.

Apply Definition 316.4 with identity transport on the retained rows. Prefix normalization is inherited from the accepted packet S, so the restricted tuple satisfies the same carrier acceptance condition on every retained unary window.

Use Definition 316.2 for the classifier row. Since the restricted schedules contain only windows already shared with S, the classifier relation is the original shared-window classifier read on the smaller schedule.

Finally read the downstream surfaces through Definition 316.9 and Definition 316.10. They expose only the scheduled finite prefix endpoints and their ledgers, and no extra tail, limit, host equality, completeness, or bridge coordinate is created by the restriction. □

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket prefix truncation stability`

Theorem 316.6 (SignedDigitStream adjacent-window endpoint coherence)**.** *Let $S = (D, N, C, P, E, L)$ be an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *carrier, and let $N_{\leq k}$ and $N_{\leq k+1}$ be adjacent displayed unary prefix windows of its* $\mathsf{StreamName}^{\uparrow}$ *schedule. The normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row and carry ledger read on $N_{\leq k}$ are exactly the restriction of the endpoint row and carry ledger read on $N_{\leq k+1}$, with the restriction taken through the displayed* hsame, Cont, Pkg, *and* NameCert *rows.*

Consequently every $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and every* $\mathsf{Real}^{\uparrow}$ *finite-window boundary consumer reads the same restricted endpoint on the overlap $N_{\leq k}$. The coherence statement is internal to the finite* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet and adds no quotient tail, selected limit, host real equality, completeness datum, or standard bridge row.*

Proof. Project the six carrier rows of Definition 316.1. The two adjacent prefix windows select the displayed rows $D_{\leq k}, C_{\leq k}, P_{\leq k}, E_{\leq k}, L_{\leq k}$ and $D_{\leq k+1}, C_{\leq k+1}, P_{\leq k+1}, E_{\leq k+1}, L_{\leq k+1}$ already present in the accepted packet S.

Apply Definition 316.5 to both prefixes. This repacks $S_{\leq k}$ and $S_{\leq k+1}$ as accepted packets whose visible rows are inherited from S through the displayed hsame, Cont, Pkg, and NameCert routes.

The smaller prefix is a common unary window for these two accepted packets. Apply Definition 316.8 on that shared window. It identifies the normalized endpoint row and carry ledger read from $S_{\leq k}$ with the restriction of the corresponding rows read from $S_{\leq k+1}$.

Finally route the consumer reads through Definition 316.9 and Definition 316.10. Both consumers read only the finite scheduled endpoint and ledger rows, so their overlap read is the same restricted endpoint and no tail, selected limit, host real equality, completeness datum, or standard bridge coordinate is available to export. □

Theorem 316.7 (SignedDigitStream prefix handoff idempotence)**.** *Let $S = (D, N, C, P, E, L)$ be an accepted* SignedDigitStream$^{\uparrow}$ *carrier, and let $S_{\leq k}$ be the restriction of S to the displayed unary schedule prefix $N_{\leq k}$. Applying the* RegSeqRat$^{\uparrow}$ *handoff or the* Real$^{\uparrow}$ *finite-window seal boundary to $S_{\leq k}$ gives exactly the same visible rows as first applying the handoff or boundary to S and then restricting the resulting consumer surface to $N_{\leq k}$.*

The common surface consists only of the retained digit rows, schedule rows, carry and normalization ledgers, provenance rows, endpoint rows, hidden-tail ledger rows, visible hsame *and* Cont *routes,* Pkg *rows, and* NameCert *rows. It contains no quotient tail, selected limit, host real equality, completeness datum, or standard bridge.*

Proof. Project the carrier rows of S and of the restricted packet $S_{\leq k}$. By Definition 316.5, the retained digit, schedule, carry, provenance, endpoint, and ledger rows are the corresponding prefix rows inherited from S.

Apply Definition 316.9 to both packets. A RegSeqRat consumer reads only scheduled finite endpoint observations and their radius ledgers, so restricting the already exported handoff to $N_{\leq k}$ selects the same rows as exporting from $S_{\leq k}$.

Apply Definition 316.10 in the same way. The seal boundary reads the finite window rows already listed above, so prefixing before or after the boundary yields the same visible surface and creates none of the excluded tail, limit, host-equality, completeness, or bridge coordinates. □

Theorem 316.8 (SignedDigitStream common-window determinacy)**.** *Let S and T be accepted* SignedDigitStream$^{\uparrow}$ *carriers such that* SignedDigitStreamClassifier(S, T) *holds. For every unary window requested by both* StreamName$^{\uparrow}$ *schedules, the normalized* DyadicRatCore$^{\uparrow}$ *endpoint row read from S and the normalized endpoint row read from T are classifier-equal, and the carry ledgers repack to the same endpoint read.*

The conclusion is only a finite-window determinacy statement: it compares scheduled signed-digit prefixes, normalized endpoint rows, and carry ledgers, and it introduces no quotient stream, Real$^{\uparrow}$ *equality, or ambient host endpoint.*

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamWindowPacket common window determinacy`

Proof. Unfold the classifier of Definition 316.2. A common scheduled unary window is one of the shared finite observation rows named by the two StreamName$^{\uparrow}$ schedules, so the classifier exposes the two digit prefixes, their carry ledgers, and their normalized endpoint rows at that same window.

Project the endpoint component of the classifier row. The row states that the two normalized DyadicRatCore$^{\uparrow}$ endpoints are compared by the DyadicRatCore$^{\uparrow}$ classifier, hence the endpoint reads are equal in the only public sense available to the packet.

It remains to identify the carry data used by the read. The componentwise carry-ledger transport in the classifier repacks the two prefix normalizations into the same endpoint row. Since

the carrier has no quotient-stream or $\mathsf{Real}^{\uparrow}$ coordinate, this exhausts the common-window conclusion. □

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamClassifier common window determinacy`

Theorem 316.9 (SignedDigitStream RegSeqRat handoff)**.** *An accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *carrier hands a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer only the scheduled normalized dyadic endpoint rows and their radius ledgers. For every unary precision window requested by the* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer, the handoff reads a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the signed digit prefix selected by that schedule, the carry ledger that normalizes the prefix, and the resulting* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row. The consumer may use these rows as the endpoint observations of a regular rational stream.*

The handoff does not assert regularity by itself, does not seal a $\mathsf{Real}^{\uparrow}$ *name, and does not identify signed-digit presentations by host real equality.*

Proof. Read the accepted carrier through Definition 316.1. A precision request first selects a finite schedule row and then exposes exactly the digit prefix, carry ledger, provenance, and normalized endpoint row attached to that schedule.

The target $\mathsf{RegSeqRat}^{\uparrow}$ consumer needs endpoint observations and radius ledgers. The endpoint row is already a $\mathsf{DyadicRatCore}^{\uparrow}$ row, and the radius budget is read from the same unary precision window that selected the prefix.

Regularity is a separate $\mathsf{RegSeqRat}^{\uparrow}$ obligation: it compares pairs of endpoint rows by dyadic bounds. The signed-digit handoff supplies the finite endpoint source for those comparisons and no additional real-seal, quotient, or host-equality data. □

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket regseqrat handoff`

Theorem 316.10 (SignedDigitStream RealUp finite-window seal boundary)**.** *For any accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet, a* $\mathsf{Real}^{\uparrow}$ *seal may inspect only the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, signed digit prefix, carry and normalization ledger, normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, visible* hsame *and* Cont *routes,* Pkg *provenance, and hidden-tail ledger already exposed by Definition 316.1 and Definition 316.9.*

It may not read a quotient stream, host real equality, hidden tail value, selected limit, completeness datum, or standard bridge row.

Proof. Project the accepted carrier tuple of Definition 316.1. Its coordinates are the finite digit rows, the $\mathsf{StreamName}^{\uparrow}$ schedule, the carry and normalization ledger, provenance, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, and the hidden-tail ledger.

Use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 316.9 to identify the regular-rational observations seen by a downstream seal. Every such observation is obtained from a scheduled finite window and its normalized endpoint row, together with the visible ledger data attached to that window.

The excluded data would require an additional coordinate: a quotient stream, host real equality witness, hidden tail value, selected limit, completeness datum, or standard bridge row. No such coordinate appears in the carrier or in the handoff row, so the $\mathsf{Real}^{\uparrow}$ seal boundary is exactly the listed finite-window surface. □

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket real seal semantic name certificate`

Theorem 316.11 (SignedDigitStream NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet* (D, N, C, P, E, L) *of Definition 316.1;*

2. *classifier: the shared-window normalized endpoint comparison of Definition 316.2;*
3. *transport: the componentwise window transport of Definition 316.4;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: the scheduled endpoint row exported by Definition 316.9;*
5. $\mathsf{Real}^{\uparrow}$ *seal boundary: the finite-window exclusion of quotient streams, host real equality, hidden tail values, selected limits, completeness data, and standard bridge rows from Definition 316.10.*

These rows form the seed certificate surface for finite-window signed-digit stream names over dyadic rows.

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket namecert obligation surface`

Proof. The carrier row supplies the signed digits, schedule, carry ledger, provenance, normalized endpoints, and hidden-tail boundary. This fixes the source data of the name.

The classifier and transport rows identify the only stable motion of that data: shared windows are compared after normalization to $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, and componentwise transport repacks the same finite rows.

The handoff row records exactly what a $\mathsf{RegSeqRat}^{\uparrow}$ consumer may read, namely scheduled normalized dyadic endpoints and their visible ledgers. The $\mathsf{Real}^{\uparrow}$ seal boundary excludes quotient streams, host real equality, hidden tail values, selected limits, completeness data, and standard bridge rows, so the five listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$ surface. □

Theorem 316.12 (SignedDigitStream scoped finite-window package)**.** *Every accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet determines a scoped BEDC package whose carrier row is the finite* BHist *packet* (D, N, C, P, E, L)*, whose classifier row compares shared finite windows through* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint equality, whose transport row is componentwise* hsame *and* Cont *transport, whose consumer row is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and whose ledger row hides only unobserved windows.*

The package reads the signed-digit stream surface only through the finite rows named by BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*, together with the L10 source certificates that supply those rows.*

Proof. Use the carrier of Definition 316.1 as the package spine. Its tuple (D, N, C, P, E, L) supplies the finite digit rows, schedule rows, carry ledger, provenance, endpoint rows, and hidden-tail ledger.

The classifier is the shared-window endpoint comparison of Definition 316.2. Transport is the componentwise hsame and Cont movement supplied by Definition 316.4.

The consumer face is exactly the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 316.9. Finally, Definition 316.11 identifies the five rows as the local $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$ surface, so the scoped package has no additional quotient tail, host-real, or completed-real coordinate. □

Theorem 316.13 (SignedDigitStream public interface export)**.** *The finite carrier of Definition 316.1, classifier of Definition 316.2, checked window transport of Definition 316.4,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 316.9,* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary of Definition 316.10,* $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$ *obligation surface of Definition 316.11, and scoped window package of Definition 316.12 assemble the reusable public* $\mathsf{SignedDigitStream}^{\uparrow}$ *interface.*

The interface exposes finite digit rows, schedule rows, carry ledger rows, normalized dyadic endpoint rows, hidden-window ledger rows, hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. It exports no quotient tail, host real equality, completed real endpoint, or standard bridge.*

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket public interface export`

Proof. Use the carrier and classifier definitions as the visible spine. The carrier provides the finite digit rows, schedule rows, carry ledger, provenance, endpoint rows, and hidden-tail boundary; the classifier reads a common scheduled window and compares only the normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints.

The checked window transport theorem gives stable movement of this finite surface across hsame and Cont routes. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff then identifies the only downstream reads: scheduled normalized endpoints and the visible ledgers attached to the requested precision.

Finally combine the local $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$ surface with the scoped package. These results bind the carrier, classifier, transport, consumer row, $\mathsf{Real}^{\uparrow}$ seal boundary, and hidden-window ledger to finite BHist, Pkg, and NameCert rows, excluding quotient tails, host real equality, hidden tail values, selected limits, completeness data, completed real endpoints, and standard bridges from the exported interface. □

Theorem 316.14 (SignedDigitStream public Real boundary export)**.** *The public interface export, finite-window transport, common-window determinacy,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *finite-window seal boundary,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows assemble a* $\mathsf{Real}^{\uparrow}$*-consumable finite signed-digit stream boundary.*

The exported surface has no quotient tail, host real equality, selected limit, completeness datum, or standard bridge.

Proof. Start from the carrier of Definition 316.1 and the classifier of Definition 316.2. They provide the finite signed-digit rows, schedule rows, carry ledger, provenance, normalized dyadic endpoint rows, and hidden-window ledger as the only public coordinates.

The transport and determinacy steps are Definition 316.4 and Definition 316.8. These keep the common window and normalized endpoints stable through displayed hsame and Cont routes.

The consumer faces are Definition 316.9 and Definition 316.10. They identify the regular-rational handoff and the $\mathsf{Real}^{\uparrow}$ seal boundary as reads of the same finite rows.

Finally apply Definition 316.13. The interface already packages the carrier, classifier, checked transport, consumer handoff, seal boundary, BHist, Pkg, and NameCert rows, so reading it as a public $\mathsf{Real}^{\uparrow}$ boundary adds no quotient tail, host real equality, selected limit, completeness datum, or standard bridge. □

Theorem 316.15 (SignedDigitStream public transport bridge)**.** *The transported signed-digit windows of Definition 316.4, the common-window determinacy of Definition 316.8, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 316.9, and the public* $\mathsf{Real}^{\uparrow}$ *boundary of Definition 316.14 assemble a finite transport bridge for* $\mathsf{SignedDigitStream}^{\uparrow}$*.*

Across that bridge, $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers read the same normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, scheduled digit rows, carry ledgers, visible* hsame *and* Cont *transports,* Pkg *provenance, and* NameCert *rows. The bridge exposes no quotient stream, selected limit, host real equality, completeness datum, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket public transport bridge`

Proof. Begin with Definition 316.4. It transports the finite signed-digit window componentwise, keeping the scheduled digit rows, carry ledger, normalized endpoint rows, and visible provenance inside the same finite carrier surface.

Apply Definition 316.8. The common scheduled window therefore has a single normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row for both downstream readings, with the carry ledger attached to that same row.

Now use Definition 316.9 and Definition 316.14. The first exposes the transported endpoint row to $\mathsf{RegSeqRat}^{\uparrow}$; the second exposes the same finite endpoint, transport, package, and name-certificate rows to $\mathsf{Real}^{\uparrow}$.

Thus the bridge is exactly the shared finite transport route between those two consumer faces. A quotient stream, selected limit, host real equality, completeness datum, or standard bridge row would require a coordinate outside the carrier, transport, determinacy, handoff, and public boundary rows. □

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamWindowPacket public transport bridge`

Theorem 316.16 (SignedDigitStream Real–RegSeqRat window correspondence)**.** *On every scheduled unary window of an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet, the normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows and carry ledgers exported to* $\mathsf{RegSeqRat}^{\uparrow}$ *are the same finite rows read by the* $\mathsf{Real}^{\uparrow}$ *seal boundary, up to the displayed* hsame *and* Cont *transports and the packet's* Pkg *and* NameCert *provenance.*

This correspondence contains no quotient tail, host real equality, selected limit, completeness datum, or standard bridge row.

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket real regseqrat window correspondence`

Proof. Project the accepted packet through the carrier of Definition 316.1. A scheduled unary window exposes the finite signed digit prefix, carry and normalization ledger, provenance, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, and hidden-window ledger attached to that schedule.

Use common-window determinacy from Definition 316.8. It identifies the endpoint read at the shared schedule as the same normalized dyadic row in the only public sense available to the packet, and it keeps the carry ledger tied to that endpoint row.

Now apply Definition 316.9 and Definition 316.10. The first exports the scheduled normalized endpoint and its visible ledger to $\mathsf{RegSeqRat}^{\uparrow}$; the second permits the $\mathsf{Real}^{\uparrow}$ seal to inspect exactly the same scheduled endpoint, carry, transport, package, and name-certificate rows.

Thus both consumers read the same normalized endpoint and carry data through the displayed hsame and Cont routes. Any quotient tail, host real equality row, selected limit, completeness datum, or standard bridge row would require a coordinate absent from Definition 316.1 and from both consumer boundaries. □

Theorem 316.17 (SignedDigitStream prefix consumer invariance)**.** *Let* $S = (D, N, C, P, E, L)$ *be an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet, and let* $N_{\leq k}$ *be a displayed prefix schedule in its finite unary window. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer request that factors through* $N_{\leq k}$ *reads from the prefix packet* $S_{\leq k}$ *exactly the restriction of the consumer-visible rows read from* S*: the scheduled signed-digit prefix, carry and normalization ledger, normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, visible* hsame *and* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

The prefix read exposes no quotient tail, host real equality, selected limit, completeness datum, or standard bridge row.

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket prefix consumer invariance`
↪ variant `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamWindowPacket prefix consumer invariance`

Proof. Apply Definition 316.5 to the accepted packet and the displayed schedule $N_{\leq k}$. This repacks the prefix carrier as $S_{\leq k}$ while retaining only the scheduled signed-digit prefix, carry ledger, normalized endpoint row, transport rows, package provenance, and local name-certificate row.

Use Definition 316.8 on the retained window. The theorem identifies the normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and the attached carry ledger as the same visible rows before and after prefix restriction, with the displayed hsame and Cont routes providing the only allowed comparison.

For a $\mathsf{RegSeqRat}^{\uparrow}$ request, project the retained rows through Definition 316.9. For a $\mathsf{Real}^{\uparrow}$ request, project them through Definition 316.10. In both cases the consumer factors through the same prefix carrier rows and cannot read outside the scheduled packet surface.

Finally apply Definition 316.16. It identifies the two consumer readings on the common finite window, so the rows visible from $S_{\leq k}$ are exactly the restricted rows visible from S and none of the excluded quotient, host-real, limit, completeness, or standard bridge data can enter the prefix read. □

Theorem 316.18 (SignedDigitStream prefix-window transport lock)**.** *Let $S = (D, N, C, P, E, L)$ be an accepted $\mathsf{SignedDigitStream}^{\uparrow}$ packet, let $N_{\leq k}$ be a displayed prefix schedule inside its finite unary window, and let T be the transported packet obtained from S by Definition 316.4. Restricting T to the transported prefix gives the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ finite-window consumer rows as first restricting S by Definition 316.5 and then transporting those prefix rows.*

The locked rows are exactly the transported BHist packet rows, scheduled signed-digit prefix, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, carry and normalization ledger, displayed hsame and Cont routes, Pkg provenance, $\mathsf{NameCert}$ rows, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ finite-window seal.

Proof. Begin with Definition 316.4. It transports the accepted packet componentwise, so the BHist rows, scheduled $\mathsf{StreamName}^{\uparrow}$ window, carry ledger, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows remain on the same finite packet surface.

Apply Definition 316.5 before and after this transport. Prefix restriction only narrows the displayed schedule to $N_{\leq k}$ and keeps the same endpoint, carry, provenance, and certificate rows available to consumers.

Next use Definition 316.8. The prefix window and the transported prefix window determine one normalized endpoint and one carry ledger through the displayed hsame and Cont routes, so there is no second consumer-facing row to choose.

Finally invoke Definition 316.17. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ finite-window seal read exactly the prefix-restricted packet rows on both paths. Thus transport after prefixing and prefixing after transport expose the same finite consumer surface. □

Theorem 316.19 (SignedDigitStream standard finite-window bridge)**.** *The public interface export of Definition 316.13, public $\mathsf{Real}^{\uparrow}$ boundary of Definition 316.14, and $\mathsf{Real}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ window correspondence of Definition 316.16 identify the standard finite-window bridge rows for $\mathsf{SignedDigitStream}^{\uparrow}$ as exactly the normalized dyadic endpoint rows, carry ledger, scheduled unary window, displayed hsame rows, displayed Cont routes, Pkg provenance, and $\mathsf{NameCert}$ provenance already carried by the packet.*

The bridge exposes no quotient tail, host real equality, selected limit, completeness datum, or bridgeChecked claim.

Proof. Project the public packet through Definition 316.13. The public surface is the scheduled signed-digit window together with the normalized dyadic endpoint, carry ledger, transport rows, package provenance, and name-certificate rows.

Apply Definition 316.14. The $\mathsf{Real}^{\uparrow}$ boundary reads the same finite rows and does not add a quotient tail, host real equality row, selected limit, or completeness witness.

Finally apply Definition 316.16. The regular-rational and real consumers therefore meet on the same normalized endpoint and carry rows through the displayed hsame and Cont routes, so the standard finite-window bridge has no further source coordinate and carries no bridgeChecked assertion. □

Theorem 316.20 (SignedDigitStream mature finite-window consumer coverage)**.** *An accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *finite-window packet exposes to every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exactly the following rows: the prefix-truncation surface, common-window determinacy surface,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, public* $\mathsf{Real}^{\uparrow}$ *boundary, transport bridge,* $\mathsf{Real}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *window correspondence, prefix consumer invariance, and standard finite-window bridge rows of Definition 316.19.*

The coverage contains no hidden tail, quotient equality, selected limit, completeness datum, trichotomy datum, or extra consumer coordinate beyond the displayed endpoint, carry, transport, package, and name-certificate rows.

Proof. Start with the requested finite window. Apply Definition 316.5 to replace the consumer read by the displayed prefix-truncated packet, so the read contains only the finite signed-digit prefix, scheduled unary window, endpoint row, and carry ledger.

Next apply Definition 316.8. The shared schedule fixes the normalized endpoint and carry rows seen by all consumers through the same displayed hsame and Cont transports.

Use Definition 316.9 for the $\mathsf{RegSeqRat}^{\uparrow}$ consumer side and Definition 316.16 for the common real and regular-rational readback. These two rows identify the same finite endpoint, carry, package, and name-certificate coordinates on both consumer boundaries.

Finally apply Definition 316.19. The standard finite-window bridge repacks exactly the rows already exposed by the prefix, determinacy, handoff, and correspondence theorems. Since none of those sources contains a hidden tail, quotient equality, selected limit, completeness witness, or trichotomy row, no such datum is available to a downstream consumer. □

Theorem 316.21 (SignedDigitStream canonical finite-tail normal form)**.** *Every accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet has a canonical finite-tail normal form determined by its displayed prefix window, signed digit rows, carry ledger,* hsame *rows,* Cont *rows,* Pkg *provenance, and local* NameCert *rows.*

Any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of the packet factors through that normal form. The factorization introduces no hidden tail coordinate, quotient equality, selected limit, completeness datum, trichotomy datum, or host real equality.*

Lean4 ✓ `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket consumer read chain unary closed`

Proof. First apply Definition 316.5. It replaces the packet by the displayed finite prefix without changing the endpoint and carry rows visible to consumers.

Use Definition 316.8 to identify the normal endpoint and carry ledger determined by the shared prefix window. This gives the canonical finite-tail representative over the displayed signed digit rows.

Next apply Definition 316.9 and Definition 316.14. The regular-rational and real consumer boundaries both read the same finite endpoint, carry, transport, provenance, and naming rows.

Finally invoke Definition 316.20. Every downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read is one of the finite-window consumer rows already carried by the normal form, so no excluded tail, quotient, limit, completeness, trichotomy, or host-equality coordinate is introduced. □

Theorem 316.22 (SignedDigitStream scheduled-window exhaustion)**.** *Let* $S = (D, N, C, P, E, L)$ *be an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer request admitted by*

the mature finite-window consumer boundary factors through one displayed unary schedule window of N. The endpoint row, carry ledger, hsame *rows,* Cont *rows,* Pkg *provenance, and local* NameCert *row read by the consumer are exactly the rows carried by that scheduled window.*

Consequently the consumer surface exposes no unscheduled endpoint, hidden tail coordinate, quotient equality, selected limit, completeness datum, trichotomy datum, host real equality, or standard bridge row outside the displayed unary schedule.

Proof. Start with an admitted finite-window request. Apply Definition 316.20 to reduce the read to the prefix, determinacy, handoff, public boundary, correspondence, invariance, and standard finite-window rows already exposed by the accepted packet.

Next apply Definition 316.5. The requested read is identified with a displayed prefix of the unary schedule N, so the consumer can only address rows attached to that scheduled window.

Use Definition 316.8 together with Definition 316.16. These fix the normalized endpoint and carry ledger for both $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers through the same displayed hsame and Cont routes.

Finally invoke Definition 316.19 and Definition 316.21. The remaining package provenance and local name-certificate rows are exactly those carried by the scheduled window, and the cited normal form contains no coordinate for an unscheduled endpoint, hidden tail, quotient equality, selected limit, completeness datum, trichotomy datum, host real equality, or extra standard bridge row. □

Theorem 316.23 (SignedDigitStream adjacent carry confluence)**.** *Let S and S' be accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *packets with the same displayed unary schedule and the same signed digit prefix. Suppose their displayed adjacent carry ledgers may differ, but both normalize to endpoint rows accepted by the packet classifier. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer window and every* $\mathsf{Real}^{\uparrow}$ *consumer window reads classifier-equal normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows through the same* BHist *ledger,* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *provenance.*

The confluence statement is finite-window only. It introduces no quotient stream, hidden tail, selected limit, completeness principle, trichotomy datum, or host real equality.

Proof. Begin with the shared displayed schedule and signed digit prefix. Apply Definition 316.8 to both packets. The common window fixes the normalized endpoint row up to the packet classifier and keeps the comparison inside the displayed carry ledger, hsame, and Cont routes.

Next use Definition 316.21. Each adjacent carry ledger normalizes to the canonical finite-tail normal form over the same prefix window, so the two endpoint rows are classifier-equal finite $\mathsf{DyadicRatCore}^{\uparrow}$ rows rather than distinct tail presentations.

For a $\mathsf{RegSeqRat}^{\uparrow}$ request, project the canonical endpoint rows through Definition 316.9. For a $\mathsf{Real}^{\uparrow}$ request, project them through Definition 316.16. Both consumer readings therefore pass through the same BHist, Pkg, and NameCert provenance already present in the packets.

The cited rows contain no coordinate for quotient streams, hidden tails, selected limits, completeness, trichotomy, or host real equality. Hence adjacent carry choices can only change the displayed normalization ledger, and any accepted normalization reaches the same finite consumer row. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}$**.**

Constructive story: A signed-digit stream packet is generated from finite signed dyadic digit rows, a unary $\mathsf{StreamName}^{\uparrow}$ observation schedule, carry and normalization ledgers, $\mathsf{StreamName}^{\uparrow}$ provenance, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, and a ledger hiding only the unobserved tail beyond the requested window.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 316.20 gives the mature finite-window consumer boundary for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ by packaging the prefix, determinacy, handoff, public boundary, correspondence, and standard bridge rows through the displayed endpoint, carry, transport, package, and name-certificate data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDigitStream}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SignedDigitStreamUp.SignedDigitStreamPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close full stream regularity, $\mathsf{Real}^{\uparrow}$ sealing, host real equality, quotient presentations, completeness, or bridgeChecked status.

Upgrade path: Only axiom-purity and bridge checking remain on the verification axis for this mature finite-window consumer boundary.

Chapter 317

A Concrete Naming Certificate for $\mathsf{RealBall}^{\uparrow}$

The $\mathsf{RealBall}^{\uparrow}$ interface names the finite dyadic-radius neighborhood surface around a $\mathsf{Real}^{\uparrow}$ center. In Bishop-style constructive analysis, a real neighborhood is read through finite rational approximations and explicit positive radii; the BEDC packet below keeps that reading at the visible BHist level, with no host real topology or open-cover compactness datum.

317.1 Finite dyadic-window carrier

Definition 317.1 (RealBall finite-window carrier)**.** *A* $\mathsf{RealBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (c, r, W, P, L)$$

with the following visible rows:

1. *c is a* $\mathsf{Real}^{\uparrow}$ *center name read through a scheduled* RegSeqRat *stream and its* $\mathsf{StreamName}^{\uparrow}$ *finite-window requests;*
2. *r is a positive dyadic-radius row from* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. *W is a finite dyadic-window witness listing the displayed center approximants, the displayed radius row, and the* $\mathsf{DyadicInterval}^{\uparrow}$ *interval rows that cover the visible portion of the ball;*
4. *P is the* Pkg *provenance tying the carrier to* $\mathsf{Real}^{\uparrow}$*,* RegSeqRat*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{DyadicInterval}^{\uparrow}$ *source certificates;*
5. *L is the ledger that hides only the unobserved tail beyond the displayed window and records that every public read is made through W and P.*

The carrier predicate is $\mathsf{RealBallWindow}(c, r, W, P, L)$*. Two accepted carriers classify together only when their center names, positive dyadic radii, visible interval rows, provenance rows, and tail ledgers transport by displayed* hsame *and by the same* Cont *routes named in P.*

Lean4 (definition) `BEDC.Derived.RealBallUp.RealBallWindowPacket`

Theorem 317.2 (RealBall radius transport)**.** *Let $B = (c, r, W, P, L)$ be an accepted* $\mathsf{RealBall}^{\uparrow}$ *carrier, and let $B' = (c', r', W', P', L')$ be obtained by displayed* hsame *transport of the* $\mathsf{Real}^{\uparrow}$ *center row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the visible* $\mathsf{DyadicInterval}^{\uparrow}$ *window rows, and the*

provenance and ledger rows. If the same Cont *routes named by P and P' preserve the* RegSeqRat *and* $\mathsf{StreamName}^{\uparrow}$ *reads of the center, then B' is again an accepted* $\mathsf{RealBall}^{\uparrow}$ *carrier.*

The transported classifier reads the same finite dyadic-radius neighborhood surface: it changes only the displayed representatives of c, r, W, P, and L, and it introduces no Dedekind cut, trichotomy row, completeness datum, host real ball, or open-cover topology.

Lean4 ✓ `BEDC.Derived.RealBallUp.RealBallWindowPacket radius transport`

Proof. Project the five coordinates of Definition 317.1. The center coordinate is read through $\mathsf{Real}^{\uparrow}$, RegSeqRat, and $\mathsf{StreamName}^{\uparrow}$, so the stated hsame and Cont routes transport each displayed center approximant in the finite window.

Transport the positive dyadic-radius row through $\mathsf{DyadicRatCore}^{\uparrow}$ and transport the displayed interval rows through $\mathsf{DyadicInterval}^{\uparrow}$. The visible window witness W' is obtained by repacking exactly these transported rows.

Finally transport the provenance row and the tail ledger. The ledger still hides only the unobserved tail beyond the displayed window, so the accepted packet has the same classifier surface and no additional topological or completeness coordinate. □

Theorem 317.3 (RealBall finite-window NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet* (c, r, W, P, L) *of Definition 317.1;*
2. *classifier and stability: the transported-center and transported-radius statement of Definition 317.2;*
3. *finite window: the witness W listing the displayed* RegSeqRat *center approximants, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, and the* $\mathsf{DyadicInterval}^{\uparrow}$ *interval rows visible at the requested precision;*
4. *provenance: the* Pkg *row naming the consumed* $\mathsf{Real}^{\uparrow}$, RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{DyadicInterval}^{\uparrow}$ *certificate boundaries;*
5. *ledger boundary: the row L recording that only the unobserved tail beyond the displayed window is hidden.*

These five rows are the full public seed surface for the finite dyadic-radius neighborhood name.

Lean4 ✓ `BEDC.Derived.RealBallUp.RealBallWindowPacket finite window namecert`

Proof. The carrier row supplies the center, radius, finite window, provenance, and tail-ledger coordinates. This fixes the source domain of the name.

The classifier and stability row is Definition 317.2. It says that the only permitted motion is componentwise hsame transport along the displayed Cont routes named by the provenance row.

The remaining rows are direct projections from the carrier. Since the construction records no host real topology, open-cover family, completeness principle, or hidden equality quotient, the five listed rows exhaust the $\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$ obligation surface. □

Theorem 317.4 (RealBall metric-neighborhood handoff)**.** *A downstream* $\mathsf{Met}^{\uparrow}$ *consumer may read an accepted* $\mathsf{RealBall}^{\uparrow}$ *carrier only as a finite real-neighborhood surface: a* $\mathsf{Real}^{\uparrow}$ *center name, a positive dyadic radius, a visible dyadic interval window, and the provenance and tail ledger of Definition 317.1. The handoff supports metric-local arguments that consume displayed dyadic tolerances around a real name.*

The handoff is not an open-cover topology and does not assert a basis theorem, Heine–Borel compactness, completeness of $\mathsf{Real}^{\uparrow}$*, or choice of a global covering family.*

Lean4 ✓ `BEDC.Derived.RealBallUp.RealBallWindowCarrier metric neighborhood handoff`

Proof. Read the carrier through Definition 317.1. The only metric-facing coordinates are the center name, the positive dyadic radius, and the displayed dyadic interval window around the center approximants.

The transport theorem Definition 317.2 supplies the stability needed by a metric consumer: replacing the center name or radius by displayed hsame representatives repacks the same finite neighborhood surface.

No coordinate of the carrier contains an open cover, a covering selection principle, a completed metric object, or a host topology. A $\mathsf{Met}^{\uparrow}$ consumer therefore receives exactly the finite dyadic-radius read described in the statement. □

Theorem 317.5 (RealBall standard obligation boundary)**.** *The* $\mathsf{RealBall}^{\uparrow}$ *chapter claims only finite dyadic-window balls around* $\mathsf{Real}^{\uparrow}$ *names. Its dependency list is exactly the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* RegSeqRat*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicInterval}^{\uparrow}$ *certificate surface named by the provenance row.*

This places the chapter inside the same concrete-instance lattice as chapter 38, chapter 126, and chapter 301: the $\mathsf{RealBall}^{\uparrow}$ *carrier consumes* $\mathsf{Real}^{\uparrow}$ *names through* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* RegSeqRat *readback before exposing dyadic-window balls.*

It does not claim real-number completeness, order trichotomy, locatedness outside the displayed window, Heine–Borel compactness, equivalence with host real open balls, a host topological bridge, or any open-cover compactness principle.

Lean4 ✓ `BEDC.Derived.RealBallUp.RealBallWindowPacket standard obligation boundary`

Proof. The positive-radius datum is a $\mathsf{DyadicRatCore}^{\uparrow}$ row, the center datum is a $\mathsf{Real}^{\uparrow}$ name read through RegSeqRat and $\mathsf{StreamName}^{\uparrow}$, and the visible ball surface is a finite collection of $\mathsf{DyadicInterval}^{\uparrow}$ rows. These are exactly the dependencies listed in the carrier provenance.

Apply Definition 317.3 to exhaust the public obligation surface. Every accepted read is one of the five finite rows listed there.

The excluded principles would require coordinates not present in the packet: global real order data, completed metric limits, open-cover families, compactness witnesses, or a host topological bridge. Since the carrier has no such coordinate, the boundary is precisely the finite dyadic-window claim. □

Theorem 317.6 (RealBall scoped dependency packet)**.** *Every accepted* $\mathsf{RealBall}^{\uparrow}$ *finite-window packet is scoped by the center* $\mathsf{Real}^{\uparrow}$ *stream read, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, visible* $\mathsf{DyadicInterval}^{\uparrow}$ *window rows,* Pkg *provenance, and the local ledger boundary. All transport used by the packet factors through* BHist *carrier rows, displayed* hsame *identifications,* Cont *routes, and* NameCert *rows.*

Consequently the accepted real-ball surface is a finite dyadic-neighborhood packet over the L10 primitives named in the carrier. It does not add a host topological ball, Heine–Borel compactness data, a global open-cover family, or completeness of the real-name carrier.

Proof. Project the carrier of Definition 317.1. The center coordinate is read as a $\mathsf{Real}^{\uparrow}$ name through the displayed RegSeqRat and $\mathsf{StreamName}^{\uparrow}$ rows, while the radius coordinate is the positive $\mathsf{DyadicRatCore}^{\uparrow}$ row stored in the same packet.

Next compose Definition 317.2 with the visible $\mathsf{DyadicInterval}^{\uparrow}$ window rows. This identifies the permitted hsame and Cont movement of the finite dyadic neighborhood without adding any coordinate outside the carrier.

The consumer face is the metric-neighborhood handoff of Definition 317.4. The local $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ surface of Definition 317.3 supplies the provenance and ledger rows used by that handoff.

Finally apply Definition 317.5. It rules out host topology, compactness, and completeness coordinates, so the scoped dependency packet is exactly the finite center, radius, window, provenance, and ledger surface stated above. □

Lean4 ✓ `BEDC.Derived.RealBallUp.RealBallWindowPacket scoped dependency packet`

Theorem 317.7 (RealBall public NameCert export)**.** *The finite-window carrier of Definition 317.1, the radius transport of Definition 317.2, the finite-window* $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ *surface of Definition 317.3, the metric-neighborhood handoff of Definition 317.4, the standard obligation boundary of Definition 317.5, and the scoped dependency packet of Definition 317.6 assemble a public* $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ *export for downstream* $\mathsf{Met}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *consumers.*

The exported boundary is exactly the finite dyadic-window neighborhood surface: a $\mathsf{Real}^\uparrow$ *center read through* $\mathsf{RegSeqRat}$ *and* $\mathsf{StreamName}^\uparrow$*, a positive* $\mathsf{DyadicRatCore}^\uparrow$ *radius, visible* $\mathsf{DyadicInterval}^\uparrow$ *rows,* Pkg *provenance, and the tail ledger. It exports no host topology, open-cover compactness, trichotomy, completeness datum, or host real ball.*

Proof. Start with the carrier of Definition 317.1. Its rows give the center name, positive dyadic radius, finite visible window, provenance packet, and ledger boundary that a public consumer can inspect.

Apply Definition 317.2 to fix the stability row, and apply Definition 317.3 to package the carrier, classifier, finite window, provenance, and ledger as the local $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ surface.

Next use the metric-neighborhood handoff of Definition 317.4. This is the consumer-facing read for $\mathsf{Met}^\uparrow$ and $\mathsf{Real}^\uparrow$: it exposes the finite dyadic-radius neighborhood without changing the underlying BEDC packet.

Finally combine the standard obligation boundary of Definition 317.5 with the scoped dependency packet of Definition 317.6. These two rows state exactly which dependencies are public and which coordinates are absent, so repacking the listed rows gives the claimed $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ export and nothing beyond it. □

Theorem 317.8 (RealBall standard finite-window bridge)**.** *The public export of Definition 317.7 determines the standard finite-window bridge schema for* $\mathsf{RealBall}^\uparrow$*. The source row is the* $\mathsf{Real}^\uparrow$ *center read through* $\mathsf{RegSeqRat}^\uparrow$ *and* $\mathsf{StreamName}^\uparrow$*, the radius row is the positive* $\mathsf{DyadicRatCore}^\uparrow$ *datum, and the window row is the visible* $\mathsf{DyadicInterval}^\uparrow$ *packet. The only bridge routes are the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, and ledger rows.*

This bridge exports no host topology, open-cover compactness, trichotomy, completeness datum, or host real ball.

Proof. Begin with the public export of Definition 317.7. It exposes exactly the center, positive radius, finite window, provenance, and ledger rows of the $\mathsf{NameCert}_{\mathsf{RealBall}^\uparrow}$ packet.

Apply Definition 317.6 to identify the permitted routes. The center row factors through $\mathsf{Real}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{StreamName}^\uparrow$; the radius row factors through $\mathsf{DyadicRatCore}^\uparrow$; and the visible window factors through $\mathsf{DyadicInterval}^\uparrow$.

Finally use Definition 317.5. It excludes host topology, compactness, trichotomy, completeness, and host-ball coordinates. Projecting the displayed rows and repacking them gives exactly the finite bridge schema stated above. □

Theorem 317.9 (RealBall finite-window consumer coverage)**.** *Let* $B = (c, r, W, P, L)$ *be an accepted* $\mathsf{RealBall}^\uparrow$ *packet. Every public read of* B *by a* $\mathsf{Met}^\uparrow$ *consumer, by a* $\mathsf{RegSeqRat}^\uparrow$ *read of*

the center, or by a $\mathsf{Real}^{\uparrow}$ *neighborhood consumer factors through exactly the center row* c*, the positive dyadic radius row* r*, the visible* $\mathsf{DyadicInterval}^{\uparrow}$ *window rows* W*, the* Pkg *provenance row* P*, the ledger* L*, and the* BHist*,* hsame*,* Cont*, and* NameCert *rows isolated by Definition 317.6, Definition 317.7, and Definition 317.8.*

The consumer surface contains no coordinate for host topology, open-cover compactness, Heine–Borel data, trichotomy, completeness, or host real balls.

Proof. Project the accepted packet through the carrier of Definition 317.1. The projection exposes only the center name, positive dyadic radius, finite visible dyadic window, provenance packet, and tail ledger, so these rows form the whole candidate packet seen by a consumer.

Use the radius transport of Definition 317.2 and the finite-window $\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$ surface of Definition 317.3. They keep the positive radius row and the visible window stable under the displayed hsame and Cont classifiers, so replacing the packet by a classified equal one does not reveal a new read coordinate.

Next apply the metric-neighborhood handoff and the public export of Definition 317.4 and Definition 317.7. These are exactly the public routes used by $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ neighborhood consumers: each route reads one of the displayed rows and then repacks it as the same finite-window NameCert boundary.

Finally use the standard bridge of Definition 317.8. The bridge has no extra source, pattern, or ledger coordinate beyond the center, radius, visible-window, provenance, BHist, hsame, Cont, and NameCert rows already listed. Since the bridge also excludes host topology, open-cover compactness, Heine–Borel data, trichotomy, completeness, and host real balls, every public consumer read is exhausted by the same visible packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$**.**

Constructive story: A real-ball packet is generated by a $\mathsf{Real}^{\uparrow}$ center name read through RegSeqRat and $\mathsf{StreamName}^{\uparrow}$, a positive $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, visible $\mathsf{DyadicInterval}^{\uparrow}$ window rows, Pkg provenance, and a ledger hiding only the unobserved tail.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$, maturePackageClosed)

Scope closed: The finite-window carrier of Definition 317.1, radius transport of Definition 317.2, finite-window NameCert surface of Definition 317.3, metric handoff of Definition 317.4, boundary row of Definition 317.5, scoped dependency packet of Definition 317.6, public export of Definition 317.7, standard bridge of Definition 317.8, and consumer coverage of Definition 317.9 are bound to BHist, hsame, Cont, Pkg, NameCert, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{DyadicInterval}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealBall}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RealBallUp.RealBallWindowPacket\_finite\_window\_namecert`

Bridge status: none

Origin: human

Not claimed: This chapter does not close completeness, trichotomy, Heine–Borel compactness, host real topology, or host real balls.

Upgrade path: Further Lean work concerns the public export, standard bridge, and consumer-coverage targets beyond the cited finite-window NameCert surface.

Chapter 318

A Concrete Naming Certificate for $\mathsf{EulerLagrange}^{\uparrow}$

The $\mathsf{EulerLagrange}^{\uparrow}$ interface names the finite variational-equation surface consumed by $\mathsf{Pde}^{\uparrow}$ and $\mathsf{NoetherSymmetry}^{\uparrow}$. In the classical calculus of variations, as presented for example by Gelfand–Fomin and by Giaquinta–Hildebrandt, the Euler–Lagrange equation is read from an action functional, admissible variations, a first-variation computation, and boundary conditions. The BEDC packet below records only the finite rows needed to cite that surface.

318.1 Finite variational packet

Definition 318.1 (EulerLagrangeUp BHist carrier)**.** *An* $\mathsf{EulerLagrange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, V, F, B, Q, R, C, H, P, N)$$

with the following visible rows:

1. *A is an action-functional row naming the displayed Lagrangian or energy expression through* $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source certificates;*
2. *V is an admissible-variation row naming the finite variation family and the source-domain restrictions it must respect;*
3. *F is a first-variation ledger that records the displayed derivative read, the finite cancellation rows, and the residual equation row;*
4. *B is a boundary-condition row saying which boundary terms are fixed, vanish, or remain visible to a downstream consumer;*
5. *Q is the* $\mathsf{Pde}^{\uparrow}$ *source row that receives the residual equation as an equation surface;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar row used for action values, first-variation values, and displayed coefficient comparisons;*
7. *C records the* Cont *provenance routes by which the action, variation, boundary, PDE, and scalar rows are read;*
8. *H records the* hsame *transport rows for the displayed action, variation, first-variation ledger, boundary row, PDE row, and scalar row;*

9. *P is the* Pkg *provenance row naming* Derivative$^\uparrow$, Integral$^\uparrow$, Pde$^\uparrow$, Real$^\uparrow$, *and* NoetherSymmetry$^\uparrow$ *as the finite consumers and suppliers of the packet;*
10. *N is the* NameCert *provenance row that registers the packet under* EulerLagrange$^\uparrow$.

The carrier predicate is EulerLagrangePacket$(A, V, F, B, Q, R, C, H, P, N)$. *Two accepted packets classify together only when the ten displayed rows transport componentwise by the* hsame *witnesses and the same* Cont *routes named in C and P.*

Theorem 318.2 (EulerLagrange local action row)**.** *Every accepted* EulerLagrange$^\uparrow$ *carrier exposes a local action-functional row A together with its* Derivative$^\uparrow$, Integral$^\uparrow$, *and* Real$^\uparrow$ *provenance. A consumer may read from A only the displayed integrand or energy expression, the displayed derivative route used for the first variation, the displayed integral route used for the action value, and the scalar rows named by the* Real$^\uparrow$ *boundary.*

The row does not assert existence of minimizers, smooth compactness, a host manifold of fields, or any unrecorded analytic regularity.

Proof. Project the A, R, C, and P coordinates from Definition 318.1. These coordinates explicitly name the action expression, the derivative and integral provenance, the real scalar reads, and the finite continuation routes.

The classifier clause permits only componentwise hsame transport of those rows. Hence every local action read available to a downstream consumer is one of the displayed rows of the packet, and no host variational object can be inferred from the carrier. □

Theorem 318.3 (EulerLagrange variation ledger and boundary row)**.** *For an accepted carrier $E = (A, V, F, B, Q, R, C, H, P, N)$, the admissible-variation row V, the first-variation ledger F, and the boundary-condition row B jointly determine the public variational ledger. The ledger records which displayed variations are admitted, which first-variation terms are retained or cancelled, and which boundary terms are fixed or exposed.*

No consumer may read an integration-by-parts theorem, a density argument for arbitrary test variations, or a hidden boundary regularity fact unless that datum appears as one of the finite rows in V, F, or B.

Proof. Read the three coordinates V, F, and B from the carrier. The admissible variations are exactly the entries of V, the computed first-variation terms are exactly the ledger entries of F, and the boundary policy is exactly the row B.

The transport rows in H keep these three coordinates synchronized under hsame. The Cont routes in C say where the rows may be read, but they do not add a new analytic rule.

Therefore the public variational ledger is exhausted by the three displayed coordinates. Any stronger calculus principle would require an additional row, which is not present in the carrier. □

Theorem 318.4 (EulerLagrange PDE source handoff)**.** *The* Pde$^\uparrow$ *row Q of an accepted* EulerLagrange$^\uparrow$ *carrier is the only PDE-facing handoff of the residual first-variation equation. It names the displayed equation row produced from F after applying the boundary policy B, together with the* Cont, hsame, *and* Pkg *provenance that connect it to* Pde$^\uparrow$.

The handoff does not claim well-posedness, ellipticity, hyperbolicity, weak-solution existence, or equivalence with a standard analytic PDE model.

Proof. Start with the first-variation ledger F and boundary row B. Their displayed residual equation is the equation surface recorded in the Pde$^\uparrow$ coordinate Q.

The provenance rows C, H, and P then identify how this equation surface is transported and packaged for a $\mathsf{Pde}^{\uparrow}$ consumer. Since Q is a finite source row, it carries only the residual equation and its named dependencies.

No coordinate of the packet stores a solution theory, type classification of the PDE, regularity estimate, or bridge to an ambient analytic model. The stated handoff is therefore the whole PDE-facing surface. □

Theorem 318.5 (EulerLagrange Noether boundary consumer row)**.** *A* $\mathsf{NoetherSymmetry}^{\uparrow}$ *consumer of an accepted* $\mathsf{EulerLagrange}^{\uparrow}$ *packet may read only the action row, admissible-variation row, first-variation ledger, boundary-condition row, PDE residual row,* $\mathsf{Real}^{\uparrow}$ *scalar row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

This Noether-facing boundary contains no conserved-current theorem, symmetry-completeness theorem, analytic PDE regularity, or standard Noether bridge.

Proof. Project the accepted carrier of Definition 318.1. Its coordinates expose the action row, admissible variations, first-variation ledger, boundary policy, PDE residual row, scalar row, continuation routes, transport witnesses, package provenance, and local name certificate.

Use Definition 318.2 for the action-facing projection, and use Definition 318.3 for the admissible-variation, first-variation, and boundary-condition rows. These are the variational coordinates available before any Noether-facing read.

Apply Definition 318.4. The residual equation read by the $\mathsf{Pde}^{\uparrow}$ coordinate is the same finite row exposed to a $\mathsf{NoetherSymmetry}^{\uparrow}$ consumer, together with the displayed hsame, Cont, Pkg, and NameCert provenance.

Since the carrier has no coordinate for conserved currents, symmetry completeness, analytic regularity, or a standard Noether bridge, the Noether-facing consumer boundary is exactly the finite row list in the statement. □

Theorem 318.6 (EulerLagrange NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ *seed surface consists exactly of the carrier of Definition 318.1, the local action row of Definition 318.2, the variation ledger and boundary row of Definition 318.3, and the PDE source handoff of Definition 318.4. Its dependency list is the finite* Pkg *row naming* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{Pde}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{NoetherSymmetry}^{\uparrow}$*, together with the displayed* Cont*,* hsame*, and* NameCert *provenance rows.*

The surface does not claim the full calculus of variations, a variational existence theorem, natural boundary-condition completeness, Noether correspondence, analytic PDE regularity, or a standard bridge.

Proof. Assemble the obligation surface from the carrier. The definition supplies the action, variation, first-variation ledger, boundary, PDE, scalar, transport, package, and name-certificate rows.

The first three theorems identify the only public projections needed by consumers: local action reads, variational ledger reads, and the PDE handoff. These projections cover every coordinate used by $\mathsf{EulerLagrange}^{\uparrow}$.

The exclusions follow by inspection of the finite packet. None of its rows contains an existence theorem, a host variational model, a Noether bijection, analytic regularity, or a standard bridge, so those claims remain outside the seed surface. □

Lean4 ✓ `BEDC.Derived.EulerLagrangeUp.EulerLagrangePacket semantic name certificate`

Theorem 318.7 (EulerLagrange LagrangianMechanics route)**.** *An accepted* $\mathsf{EulerLagrange}^{\uparrow}$ *packet may be consumed as the variational-equation row of chapter 337 only through the displayed action*

row, admissible-variation row, first-variation ledger, boundary row, $\mathsf{Pde}^{\uparrow}$ *residual row,* $\mathsf{Real}^{\uparrow}$ *scalar row,* hsame *and* Cont *provenance,* Pkg *provenance, and local* NameCert *rows of Definition 318.1. The sibling* $\mathsf{LagrangianMechanics}^{\uparrow}$ *consumer surface reads that packet as the finite Euler–Lagrange residual interface and not as a host variational calculus.*

Proof. Project the accepted carrier of Definition 318.1. The local action theorem Definition 318.2 identifies the action-facing row and its derivative, integral, and scalar provenance.

Next use Definition 318.3. It binds the admissible-variation row, first-variation ledger, and boundary row as the finite variational ledger that feeds the residual equation.

Apply Definition 318.4 and Definition 318.6. These theorems place the residual equation in the $\mathsf{Pde}^{\uparrow}$ row and repack the action, variation, boundary, scalar, hsame, Cont, Pkg, and NameCert rows as the local obligation surface.

Finally identify the sibling consumer by Definition 337.3. The $\mathsf{LagrangianMechanics}^{\uparrow}$ interface consumes only the residual-equation row and the finite action ledger that feeds it, so the Euler-Lagrange packet reaches that chapter through the displayed rows and through no host variational calculus, boundary-completeness principle, or standard analytic model. □

Theorem 318.8 (EulerLagrange scoped dependency certificate)**.** *Every exported* $\mathsf{EulerLagrange}^{\uparrow}$ *row factors through the action row, admissible-variation row, first-variation ledger, boundary row,* $\mathsf{Pde}^{\uparrow}$ *residual-equation row,* $\mathsf{Real}^{\uparrow}$ *scalar row,* Cont*,* hsame*,* Pkg*, and* NameCert *provenance listed in Definition 318.1, Definition 318.2, Definition 318.3, Definition 318.4, and Definition 318.6.*

The scoped certificate exposes no minimizer existence, full calculus of variations, Noether correspondence, analytic PDE regularity, or standard bridge data.

Proof. Project the carrier of Definition 318.1. Its coordinates are the action row, admissible-variation row, first-variation ledger, boundary row, residual-equation handoff, scalar row, continuation routes, transport witnesses, package provenance, and local name certificate.

The action theorem supplies the only action-facing projection. The variation-ledger theorem supplies the only admissible-variation, first-variation, and boundary projections. The PDE handoff theorem supplies the only residual-equation projection to $\mathsf{Pde}^{\uparrow}$.

The NameCert obligation theorem repacks these projections with the listed Cont, hsame, Pkg, and NameCert provenance. None of the projected rows contains a minimizer, a complete calculus-of-variations principle, a Noether correspondence, analytic PDE regularity, or bridge data, so the scoped dependency boundary is exactly the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.EulerLagrangeUp.EulerLagrangePacket scoped dependency certificate`

Theorem 318.9 (EulerLagrange public export)**.** *Definition 318.6 and Definition 318.8 export the public* $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ *boundary over the action row, admissible-variation row, first-variation ledger, boundary row, PDE residual handoff,* $\mathsf{Real}^{\uparrow}$ *scalar row,* Cont *routes,* hsame *transport,* Pkg *provenance, and* NameCert *row.*

The exported boundary contains no variational existence theorem, natural-boundary completeness, Noether correspondence, analytic PDE regularity, or standard bridge.

Proof. Project the finite carrier of Definition 318.1. The action row, admissible-variation row, first-variation ledger, boundary row, PDE residual handoff, scalar row, continuation routes, transport witnesses, package provenance, and local name row are the carrier coordinates available to public consumers.

The obligation theorem identifies the $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ surface over those coordinates, and the scoped dependency certificate binds the same rows to the displayed source certificates. Repacking these two results gives the public boundary without adding any coordinate outside the carrier.

The exclusions are preserved because the carrier has no row for variational existence, natural-boundary completeness, Noether correspondence, analytic PDE regularity, or a standard bridge. The public export is therefore exactly the finite BEDC packet surface already projected by the two cited theorems. □

Theorem 318.10 (EulerLagrange standard bridge boundary)**.** *Any standard reading of the accepted* $\mathsf{EulerLagrange}^{\uparrow}$ *packet must read and write only the displayed action row, admissible-variation row, first-variation ledger, boundary row, PDE residual row,* $\mathsf{Real}^{\uparrow}$ *scalar row,* Cont *routes,* hsame *transport,* Pkg *provenance, and* $\mathsf{NameCert}$ *provenance exposed by Definition 318.1 and Definition 318.9. The bridge boundary exports no minimizer existence, full calculus of variations, Noether correspondence, analytic PDE regularity, or unrecorded standard-model data.*

Proof. Project the carrier of Definition 318.1. Its available coordinates are exactly the action row, admissible-variation row, first-variation ledger, boundary row, PDE residual row, scalar row, continuation routes, transport witnesses, package provenance, and local certificate provenance.

Apply Definition 318.9. It identifies those carrier coordinates as the public $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ boundary and supplies no additional standard-model slot. A standard bridge whose input is this accepted packet can therefore read the listed rows and can write back only the same rows after the displayed Cont and hsame transport.

The excluded analytic and variational claims are absent from both the carrier and the public export. Since no row records minimizer existence, a full calculus of variations, a Noether correspondence, PDE regularity, or external standard-model data, the standard bridge boundary is exactly the finite public packet already displayed. □

Theorem 318.11 (EulerLagrange finite PDE consumer example)**.** *The public export and standard bridge boundary of an accepted* $\mathsf{EulerLagrange}^{\uparrow}$ *packet determine a finite PDE-residual consumer surface over the action row, admissible-variation row, first-variation ledger, boundary-condition row,* $\mathsf{Pde}^{\uparrow}$ *residual row,* $\mathsf{Real}^{\uparrow}$ *scalar row, visible* hsame*, visible* Cont*,* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance.*

The consumer surface does not assert minimizer existence, the full calculus of variations, natural-boundary completeness, Noether correspondence, analytic PDE regularity, or external standard-model data. It is only the downstream finite-row read of the residual coordinate already present in the accepted packet.

Proof. Project the carrier of Definition 318.1. The carrier lists exactly the action row, admissible-variation row, first-variation ledger, boundary-condition row, PDE residual row, real scalar row, transport, continuation, package, and certificate provenance available to downstream consumers.

Apply Definition 318.9. It exposes those rows as the public $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ boundary and adds no analytic or variational coordinate beyond the displayed carrier surface.

Apply Definition 318.10. The standard reading can read and write only the same displayed rows, so the $\mathsf{Pde}^{\uparrow}$ residual coordinate is a legitimate downstream consumer example on the finite packet.

The conclusion follows by absence of extra carrier rows. Since the packet has no row for minimizer existence, full variational calculus, natural-boundary completeness, Noether correspondence, analytic PDE regularity, or external standard-model data, the finite PDE-residual consumer cannot export those claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$.

Constructive story: An Euler–Lagrange packet is generated by a displayed action row, an admissible-variation row, a first-variation ledger, a boundary-condition row, a $\mathsf{Pde}^{\uparrow}$ residual-equation row, a $\mathsf{Real}^{\uparrow}$ scalar row, Cont provenance, hsame transport, Pkg rows naming $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{NoetherSymmetry}^{\uparrow}$, and a NameCert provenance row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 318.9, Definition 318.10, and Definition 318.11 export the public $\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}$ boundary and finite PDE-residual consumer surface over the action row, admissible-variation row, first-variation ledger, boundary row, PDE handoff, scalar row, Cont, hsame, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerLagrange}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.EulerLagrangeUp.EulerLagrangePacket\_scoped\_dependency\_certificate`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close the full calculus of variations, variational existence, natural boundary-condition completeness, Noether correspondence, analytic PDE regularity, or standard-model completeness beyond the displayed bridge boundary.

Upgrade path: No further paper-axis obligation remains for the finite packet; verification-axis progress is tracked by the cited Lean target and bridge status.

Chapter 319

A Concrete Naming Certificate for $\mathsf{GradedAlgebra}^{\uparrow}$

The $\mathsf{GradedAlgebra}^{\uparrow}$ interface names the finite certificate surface of a graded algebra as used in the algebraic settings of Eisenbud 1995 and Weibel 1994. The BEDC packet records a base $\mathsf{Ring}^{\uparrow}$ carrier, a $\mathsf{Monoid}^{\uparrow}$ degree carrier, displayed homogeneous component predicates over BHist rows, multiplication rows whose Cont and Pkg ledger records degree addition, and a classifier comparing same-degree component witnesses by hsame. This fills the algebraic surface consumed by $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, and the $\mathsf{KoszulDuality}^{\uparrow}$ interface of chapter 267.

319.1 Finite graded-algebra carrier

Definition 319.1 (GradedAlgebra BHist carrier)**.** *A* $\mathsf{GradedAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, D, C, M, A, P)$$

with the following visible rows:

1. *R is a base* $\mathsf{Ring}^{\uparrow}$ *carrier row naming addition, multiplication, zero, one, and the ring ledger visible to the algebra packet;*
2. *D is a* $\mathsf{Monoid}^{\uparrow}$ *degree carrier row naming the displayed degree operation and unit degree;*
3. *C is a finite family of homogeneous component predicates on displayed* BHist *element rows, each predicate indexed by a visible degree row from D;*
4. *M is a multiplication ledger row saying that multiplying a witness in degree d with a witness in degree e produces a displayed witness in the component indexed by the monoid product $d \cdot e$;*
5. *A is the action row that reads the same base carrier through* $\mathsf{Module}^{\uparrow}$ *whenever a component is consumed as a module-like slice over R;*
6. *P is the* Pkg *provenance tying the packet to the cited* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{TensorProduct}^{\uparrow}$*, and* $\mathsf{KoszulDuality}^{\uparrow}$ *certificate boundaries.*

The carrier predicate is $\mathsf{GradedAlgebraPacket}(R, D, C, M, A, P)$*. The packet contains no quotient of the degree monoid, no host equality on elements, no external algebra object, and no hidden choice of homogeneous decomposition outside the displayed rows.*

Definition 319.2 (GradedAlgebra degree classifier)**.** *For accepted* $\mathsf{GradedAlgebra}^{\uparrow}$ *carriers* $G = (R, D, C, M, A, P)$ *and* $G' = (R', D', C', M', A', P')$*, the degree classifier relates a component witness* x *in degree* d *of* G *to a component witness* x' *in degree* d' *of* G' *exactly when the following finite data are displayed:*

1. *an* hsame *row comparing the degree witnesses* d *and* d' *through the* $\mathsf{Monoid}^{\uparrow}$ *carrier rows* D *and* D'*;*
2. *an* hsame *row comparing the element witnesses* x *and* x' *through the base* BHist *rows carried by* R *and* R'*;*
3. Cont *routes in* P *and* P' *showing that the two component predicates are read through the same indexed component slot;*
4. *a* Pkg *row recording that no comparison uses host equality, quotient representatives, or an unrecorded decomposition choice.*

This relation is written $\mathsf{GradedSameDegree}(G, x, d, G', x', d')$ *and is the only classifier exported by the seed packet.*

Theorem 319.3 (GradedAlgebra component stability)**.** *Let* $G = (R, D, C, M, A, P)$ *be an accepted* $\mathsf{GradedAlgebra}^{\uparrow}$ *carrier. If the base ring row, degree monoid row, homogeneous component predicates, multiplication ledger, module-action row, and provenance row are transported by displayed* hsame *along the same* Cont *routes named by* P*, then the transported packet* $G' = (R', D', C', M', A', P')$ *is again an accepted* $\mathsf{GradedAlgebra}^{\uparrow}$ *carrier.*

Moreover, every component witness classified by GradedSameDegree *in* G *transports to a same-degree component witness in* G' *using only the displayed* BHist*,* hsame*,* Cont*, and* Pkg *rows.*

Proof. Project the six coordinates of Definition 319.1. The base ring and degree monoid rows transport first, so the target packet has the same visible ring operations and the same displayed degree operation after hsame replacement.

Transport each homogeneous component predicate through the stated Cont route. Since each component slot is indexed by a visible degree row, the transported degree row determines the transported component slot without using a quotient of the degree monoid.

The multiplication ledger and module-action row then repack over the transported carrier. The ledger still sends degree d and degree e to the displayed product degree in the transported $\mathsf{Monoid}^{\uparrow}$ row, while the action row still reads each component through the transported $\mathsf{Module}^{\uparrow}$ surface.

Finally apply Definition 319.2. Its four rows are exactly the transported degree comparison, element comparison, component-slot route, and provenance row, so the same-degree witness remains visible in G'. □

Lean4 ✓ `BEDC.Derived.GradedAlgebraUp.GradedAlgebraPacket component stability`

Theorem 319.4 (GradedAlgebra product ledger)**.** *For accepted* $\mathsf{GradedAlgebra}^{\uparrow}$ *carrier* $G = (R, D, C, M, A, P)$*, the product ledger exposes exactly the following data for homogeneous witnesses* x *in degree* d *and* y *in degree* e*: a base* $\mathsf{Ring}^{\uparrow}$ *multiplication row, the* $\mathsf{Monoid}^{\uparrow}$ *degree product row* $d \cdot e$*, a* $\mathsf{TensorProduct}^{\uparrow}$*-readable row for the pair of homogeneous inputs, a component witness for the product in degree* $d \cdot e$*, and* Pkg *provenance tying these rows to* R*,* D*,* C*,* M*, and* P*.*

The ledger exports no host equality between products, no quotient degree class, no unindexed multiplication outside the homogeneous component surface, and no algebra object beyond the displayed BHist *rows.*

Proof. Read x and y through the component predicates of Definition 319.1. Their degree rows are visible rows of the same $\mathsf{Monoid}^{\uparrow}$ carrier, so the target degree for a product is the displayed monoid product row $d \cdot e$.

Project the multiplication ledger M. This row consumes the base $\mathsf{Ring}^{\uparrow}$ multiplication surface and the two component witnesses, then returns a displayed component witness indexed by the visible product degree.

The $\mathsf{TensorProduct}^{\uparrow}$ dependency is used only to expose the pair of homogeneous inputs as a finite tensor-readable row. The product row still returns through M and the component predicate indexed by $d \cdot e$.

The provenance row P records each route used by the product read. Since the carrier has no coordinate for host equality, quotient degrees, ungraded multiplication, or an external algebra object, the listed rows exhaust the product ledger. □

Lean4 ✓ `BEDC.Derived.GradedAlgebraUp.GradedAlgebraPacket product ledger`

Theorem 319.5 (GradedAlgebra homogeneous product scope exactness)**.** *For an accepted* $\mathsf{GradedAlgebra}^{\uparrow}$ *carrier* $G = (R, D, C, M, A, P)$*, any consumer-visible homogeneous product row for witnesses in degrees* d *and* e *is exactly the* BHist *row supplied by* M *over the* $\mathsf{Ring}^{\uparrow}$ *multiplication row, the* $\mathsf{Monoid}^{\uparrow}$ *degree product row* $d \cdot e$*, the* $\mathsf{TensorProduct}^{\uparrow}$ *input-pair row, the* $\mathsf{Module}^{\uparrow}$ *action row when the product is consumed as a slice, the* Cont *routes, and the* Pkg *provenance of* G*. The packet exports no host product equality, quotient degree, or unindexed multiplication.*

Proof. Project the carrier G and its product ledger M. The base multiplication is read through R, while the target degree is the displayed product $d \cdot e$ in the degree monoid row D.

Read the pair of homogeneous inputs through the $\mathsf{TensorProduct}^{\uparrow}$ row and the component witnesses through C. When a consumer views the product as a slice, the same displayed product row is read through the module-action coordinate A rather than through an additional multiplication operation.

The remaining data are the Cont routes and the Pkg provenance stored in P. Since Definition 319.1 contains no coordinate for host product equality, quotient degrees, or unindexed multiplication, the listed rows exhaust every consumer-visible homogeneous product read. □

Theorem 319.6 (GradedAlgebra NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet* (R, D, C, M, A, P) *of Definition 319.1;*
2. *degree classifier: the same-degree comparison relation of Definition 319.2;*
3. *component stability: the transported-component statement of Definition 319.3;*
4. *product ledger: the homogeneous product row of Definition 319.4 and the scope exactness row of Definition 319.5;*
5. *provenance: the* Pkg *row tying the packet to* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{TensorProduct}^{\uparrow}$*, and* $\mathsf{KoszulDuality}^{\uparrow}$ *consumers.*

These rows are the full public seed surface for the graded-algebra name, and every downstream read must pass through BHist *rows,* hsame *classifiers,* Cont *routes, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.GradedAlgebraUp.GradedAlgebraPacket namecert obligation surface`

Proof. The carrier row fixes the displayed ring, degree, component, multiplication, action, and provenance coordinates. This is the source domain of the name.

The degree classifier supplies the only comparison of homogeneous witnesses. It compares same-degree component witnesses by displayed hsame rows and by the component routes named in the package, so it does not invoke host equality or quotient representatives.

The component-stability theorem supplies transport of the carrier and classifier, while the product-ledger and product-scope theorems supply the only multiplication read available to $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, and $\mathsf{KoszulDuality}^{\uparrow}$ consumers.

No remaining coordinate of the packet contains a host algebra, a quotient degree monoid, a decomposition choice, or an unrecorded product operation. The five listed rows therefore exhaust the $\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$ obligation surface. □

Theorem 319.7 (GradedAlgebra obligation scope certificate)**.** *The carrier, degree classifier, component stability, product ledger, homogeneous product scope exactness, and provenance rows bind the* $\mathsf{GradedAlgebra}^{\uparrow}$ *obligation surface to* $\mathsf{Ring}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{KoszulDuality}^{\uparrow}$, BHist, hsame, Cont, *and* Pkg *without host algebra equality, quotient degree, or unrecorded decomposition.*

Proof. Project the carrier of Definition 319.1. Its visible rows are the base ring, degree monoid, homogeneous component predicates, multiplication ledger, module-action row, and package provenance.

The classifier and stability data are Definition 319.2 and Definition 319.3. They keep same-degree component witnesses inside displayed BHist, hsame, Cont, and Pkg rows.

The product data are Definition 319.4 and Definition 319.5. These bind homogeneous multiplication to the displayed $\mathsf{Ring}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, and $\mathsf{KoszulDuality}^{\uparrow}$ source surfaces.

Finally apply Definition 319.6. Since the packet contains no host algebra equality, quotient degree, or unrecorded decomposition coordinate, the named rows close the obligation scope. □

Theorem 319.8 (GradedAlgebra public dependency export)**.** *The public* $\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$ *export consists exactly of the base* $\mathsf{Ring}^{\uparrow}$ *row, the degree* $\mathsf{Monoid}^{\uparrow}$ *row, the homogeneous component predicates, the multiplication ledger, the module-action row, the same-degree classifier, the* Cont *routes, and the* Pkg *provenance already listed in the obligation surface. A public consumer may project these rows and their named dependencies through* BHist, hsame, Cont, *and* Pkg, *and no other graded-algebra coordinate is exported.*

In particular, the export contains no quotient degree monoid, host algebra equality, canonical decomposition of elements into homogeneous components, or unrecorded product operation.

Proof. Project the carrier of Definition 319.1. Its public coordinates are the base ring row, degree monoid row, homogeneous component predicates, multiplication ledger, module-action row, and package provenance.

Use Definition 319.2 for the same-degree classifier. Its comparisons are displayed hsame reads through the recorded component routes, so the classifier does not add quotient degree data or host equality.

Apply Definition 319.4 and Definition 319.5 to identify the only exported multiplication reads. Then use Definition 319.7 to repack the carrier, classifier, ledger, continuation, and provenance rows as the public dependency surface.

No coordinate in the assembled surface records quotient degree monoids, canonical decompositions, host algebra equality, or products outside the ledger. The listed rows therefore exhaust the public $\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$ export. □

Closure status for $\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$.

Constructive story: A graded-algebra packet is generated by a base $\mathsf{Ring}^{\uparrow}$ row, a $\mathsf{Monoid}^{\uparrow}$ degree row, homogeneous component predicates over displayed BHist rows, a multiplication ledger whose Cont and Pkg records follow degree multiplication, and same-degree component classifiers built from hsame.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$, publicClosed)

Scope closed: The finite carrier, degree classifier, component stability, product ledger, and NameCert obligation surface bind the displayed ring, degree, component, multiplication, action, and consumer surfaces for $\mathsf{GradedAlgebra}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GradedAlgebra}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GradedAlgebraUp.GradedAlgebraPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient degrees, canonical homogeneous decomposition, host algebra equality, Koszul dual equivalence, or a standard bridge.

Upgrade path: Bridge closure requires a bridge theorem for the displayed ring, degree, component, multiplication, action, classifier, continuation, and package rows.

Chapter 320

A Concrete Naming Certificate for $\mathsf{NestedInterval}^{\uparrow}$

The $\mathsf{NestedInterval}^{\uparrow}$ interface names finite dyadic nested-interval packets. Following the constructive reading of nested intervals in Bishop and Bridges, Constructive Analysis, the BEDC carrier below records only displayed endpoint rows, inclusion ledgers, finite observation schedules, regular-rational reads, real-seal boundary rows, and certificate provenance. It does not assert a completed infinite chain or any classical nested-interval completeness principle.

The finite nested-interval packet consumes the dyadic endpoint certificate of chapter 291, the finite-window schedule discipline of chapter 126, the regular-rational readback surface of chapter 301, and the seal-facing boundary of chapter 38. These four source routes are the visible lattice behind the endpoint rows, observation schedule, readback rows, and real-facing boundary recorded below.

320.1 Finite dyadic nested packets

Definition 320.1 (NestedInterval finite carrier)**.** *A* $\mathsf{NestedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (I, O, W, S, R, E, T, P, C)$$

with the following visible rows:

1. *I is a finite list of dyadic interval rows indexed by a unary prefix;*
2. *O records for each visible row its lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints, the endpoint-order ledger, and the displayed dyadic-width ledger;*
3. *W records nested inclusion rows saying that each displayed successor interval is contained in the preceding displayed interval;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule provenance naming the finite prefix requests that selected the rows of I;*
5. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window reads available from the displayed endpoint and width ledgers;*
6. *E records the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows that a later real-name consumer may inspect without reading an infinite tail;*

7. *T is the componentwise* hsame *and* Cont *transport table for the interval, endpoint, width, schedule, regular-rational, and seal-facing rows;*
8. *P is the* Pkg *provenance tying the packet to* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and* DyadicInterval$^{\uparrow}$ *source certificates;*
9. *C is the* NameCert *provenance row for the finite nested-interval name.*

The carrier predicate is NestedIntervalPacket(N). *Two accepted carriers classify together only when the displayed endpoint rows, endpoint-order ledgers, width ledgers, nested-inclusion rows, schedule rows, regular-rational reads, seal-facing rows,* Pkg *provenance, and* NameCert *provenance transport by the same* hsame *and* Cont *routes named in T and P.*

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalTasteGate single carrier alignment`

Theorem 320.2 (NestedInterval endpoint transport)**.** *Let $N = (I, O, W, S, R, E, T, P, C)$ be an accepted* NestedInterval$^{\uparrow}$ *carrier, and let N' be obtained by componentwise displayed* hsame *transport of every lower endpoint, upper endpoint, endpoint-order ledger, width ledger, nested-inclusion row, schedule row,* RegSeqRat$^{\uparrow}$ *read,* Real$^{\uparrow}$ *seal-facing row,* Pkg *provenance row, and* NameCert *provenance row. If the same* Cont *routes named by T and P preserve the source certificate boundaries, then N' is again an accepted* NestedInterval$^{\uparrow}$ *carrier.*

The transported classifier reads exactly the transported finite dyadic interval chain and introduces no point-selection operator, no completed intersection object, and no host real completeness datum.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFinitePacket endpoint transport`

Proof. Project the carrier of Definition 320.1. The lower and upper endpoints are DyadicRatCore$^{\uparrow}$ rows, and the endpoint-order and width ledgers are ordinary displayed rows attached to the same interval slots.

Transport the interval rows and their endpoint ledgers componentwise through the stated hsame routes. The Cont routes named in T and P identify the same source certificate boundaries after transport, so every transported row still has the same dyadic endpoint role.

Finally transport the schedule, regular-rational, seal-facing, package, and name-certificate rows. Repacking these transported coordinates gives the accepted carrier N', and the classifier has no coordinate other than those finite rows. □

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier endpoint transport`

Theorem 320.3 (NestedInterval window refinement)**.** *For an accepted* NestedInterval$^{\uparrow}$ *carrier, restricting the unary prefix to a shorter displayed window or extending it by a row whose endpoint-order, width, and nested-inclusion ledgers are displayed produces another accepted finite nested-interval packet over the corresponding visible window. The classifier on the shared rows is unchanged, and the new row is read only through its* DyadicRatCore$^{\uparrow}$ *endpoints, inclusion ledger,* StreamName$^{\uparrow}$ *schedule provenance, and package routes.*

Thus finite-window refinement is stable without asserting convergence of the infinite chain.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalPacket window refinement`

Proof. For a restriction, discard the interval rows beyond the shorter unary prefix together with their endpoint, width, inclusion, regular-rational, and seal-facing reads. The remaining rows are still exactly the coordinates required by Definition 320.1.

For an extension, append the new displayed interval row and its lower endpoint, upper endpoint, order ledger, width ledger, and inclusion witness from the previous visible terminal row. Append the corresponding schedule, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, Pkg, and $\mathsf{NameCert}$ provenance rows.

The shared-window classifier is componentwise, so it is unchanged on rows already present before the restriction or extension. No step introduces an infinite intersection, compactness argument, or choice of a point. □

Theorem 320.4 (NestedInterval RegSeqRat handoff)**.** *An accepted* $\mathsf{NestedInterval}^{\uparrow}$ *carrier hands a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer only the displayed finite-window data: the selected unary prefix, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the width ledger for each visible interval, and the nested-inclusion ledger connecting adjacent visible rows. A downstream regular-rational stream may read those rows as finite approximants and bounds.*

The handoff does not certify regularity beyond the displayed rows, does not seal a completed $\mathsf{Real}^{\uparrow}$ *point, and does not identify an intersection element.*

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalRegSeqRatWindow handoff`

Proof. Read the packet through Definition 320.1. The only $\mathsf{RegSeqRat}^{\uparrow}$-facing coordinates are the scheduled finite-window reads R and the endpoint and width rows from O.

The schedule row S tells the consumer which unary prefix was requested. For each visible index, the lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints and the width ledger give the finite approximant and its explicit bound, while W records the adjacent inclusion relation.

Since the handoff reads only these displayed rows, it cannot supply a tail-regularity proof, a real seal, or a selected point in the intersection. Those are separate obligations for later consumers. □

Theorem 320.5 (NestedInterval NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: a constant dyadic interval packet whose every visible row repeats the same ordered* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower and upper endpoints;*
2. *endpoint classifier transport: the componentwise endpoint, order-ledger, width-ledger, schedule, handoff, seal-facing, package, and certificate transport of Definition 320.2;*
3. *nested-window refinement stability: the restriction and extension behavior of Definition 320.3;*
4. *ledger exactness: the assertion that the displayed endpoint order, displayed dyadic widths, and displayed adjacent inclusions exhaust the finite interval-chain ledger;*
5. *Real and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: the bounded finite rows exported by Definition 320.4 and by the seal-facing rows E of Definition 320.1.*

These rows are the full seed certificate surface for finite dyadic nested-interval names.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier semantic name certificate`

Proof. Carrier habitation is witnessed by the constant packet: choose one ordered dyadic interval row and repeat it along the displayed unary prefix, with identity inclusion and the same endpoint and width ledger at each row.

Endpoint classifier transport is Definition 320.2. Nested-window refinement stability is Definition 320.3. Together they describe the only permitted motion of accepted finite packets.

Ledger exactness follows by projection from the carrier coordinates O and W: every endpoint order, dyadic width, and adjacent inclusion used by the name is one of those displayed rows.

Finally, the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff rows are precisely R and E, bounded by the finite window named by S. Since no coordinate names an infinite chain, a completed intersection, or a selected point, the five listed rows exhaust the $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ obligation surface. □

Theorem 320.6 (NestedInterval Real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$*-facing boundary of an accepted* $\mathsf{NestedInterval}^{\uparrow}$ *carrier is limited to the displayed rows E and the finite endpoint, width, inclusion, schedule, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows that justify them. A downstream* $\mathsf{Real}^{\uparrow}$ *consumer may inspect these rows as a bounded finite window around prospective regular-rational data.*

The chapter does not claim Cantor nested-interval convergence, real completeness, compactness, Heine–Borel, extraction of a point by choice, or any equality with a host real intersection.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier real seal boundary`

Proof. Project the seal-facing coordinate E from Definition 320.1. Its dependencies are the finite endpoint, width, inclusion, schedule, and regular-rational rows named in the same carrier.

Use Definition 320.4 to identify the regular-rational part of the boundary. A $\mathsf{Real}^{\uparrow}$ consumer receives only the finite approximants and displayed bounds already present in the packet.

The excluded principles would require additional coordinates: an infinite chain object, a completed intersection, compactness data, or a choice operator selecting a point. The carrier contains none of these, so the seal boundary is exactly the displayed finite surface. □

Theorem 320.7 (NestedInterval obligation scope packet)**.** *The finite carrier of Definition 320.1, endpoint transport of Definition 320.2, window refinement of Definition 320.3,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 320.4,* $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ *obligation surface of Definition 320.5, and* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 320.6 depend only on* BHist *endpoint rows, finite dyadic interval packets, displayed* hsame *comparisons, visible* Cont *routes,* Pkg *provenance,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source certificates.*

Thus every accepted $\mathsf{NestedInterval}^{\uparrow}$ *packet is scoped by finite interval rows and the named BEDC source certificates.*

Proof. Start from the finite carrier of Definition 320.1. Its rows provide the dyadic interval endpoints, endpoint-order ledgers, width ledgers, adjacent inclusion rows, schedule provenance, handoff rows, and seal-facing rows.

Apply Definition 320.2 to account for endpoint transport, then apply Definition 320.3 to account for finite-window restriction and extension. These two results give the permitted hsame and Cont movement of the nested interval rows.

The consumer boundary is supplied by Definition 320.4 and by the finite $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ surface of Definition 320.5. Together they bind the scheduled $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ rows, Pkg provenance, and $\mathsf{NameCert}$ rows to the carrier.

Finally use Definition 320.6. The seal boundary exports only displayed finite rows to $\mathsf{Real}^{\uparrow}$ consumers and excludes open-cover compactness, choice of a nested intersection point, and host real completeness, so the scoped package is exactly the finite dependency surface stated above. □

Theorem 320.8 (NestedInterval public finite-window boundary)**.** *The finite carrier of Definition 320.1, endpoint transport of Definition 320.2, window refinement of Definition 320.3,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 320.4,* $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ *obligation surface of Definition 320.5,* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 320.6, and scoped dependency package of Definition 320.7 assemble a single public finite-window boundary for* $\mathsf{NestedInterval}^{\uparrow}$*.*

Every consumer-visible row of that boundary is one of the displayed dyadic endpoint rows, endpoint-order rows, width rows, inclusion rows, schedule rows, regular-rational handoff rows, seal-facing rows, hsame *transport rows, visible* Cont *routes,* Pkg *provenance rows, or* NameCert *rows. The boundary exports no infinite intersection point, compactness theorem, host real equality, or choice principle.*

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier public window boundary`

Proof. Project the accepted packet through the carrier of Definition 320.1. The carrier supplies the dyadic endpoint packets, endpoint-order ledgers, width ledgers, adjacent inclusion rows, schedules, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, $\mathsf{Real}^{\uparrow}$ seal-facing rows, and visible provenance rows.

Use endpoint transport of Definition 320.2 and window refinement of Definition 320.3 to account for movement of the carrier rows across hsame and Cont routes. These two blocks leave the boundary inside finite endpoint, width, inclusion, and schedule data.

Next use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the local $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$ obligation surface to bind the regular-rational windows and package provenance to the same finite rows. Apply the $\mathsf{Real}^{\uparrow}$ seal boundary to restrict seal-facing reads to displayed endpoint, inclusion, schedule, and handoff data.

Finally, the scoped dependency package fixes the permitted primitives for downstream consumers. An infinite intersection point, compactness theorem, host real equality, or choice principle would require an additional coordinate outside those seven displayed blocks, so none is exported by the public boundary. □

Theorem 320.9 (NestedInterval consumer bridge boundary)**.** *The public finite-window boundary of Definition 320.8 bridges to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers by routing only the displayed dyadic endpoint rows, endpoint-order rows, width rows, inclusion rows, schedule rows, regular-rational handoff rows, seal-facing rows,* hsame *and* Cont *transports,* Pkg *provenance, and* NameCert *rows.*

This bridge boundary adds no infinite intersection point, compactness theorem, host real equality, or choice principle.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier consumer bridge boundary`

Proof. Apply Definition 320.8. The public boundary already restricts every consumer-visible coordinate to finite dyadic endpoint, order, width, inclusion, schedule, handoff, seal-facing, transport, provenance, and name-certificate rows.

Next use Definition 320.4 for the regular-rational consumer and Definition 320.6 for the real-name consumer. Their shared inputs are exactly the finite rows displayed by the public boundary.

The bridge therefore has no readable coordinate outside the intersection of those three surfaces. An infinite intersection point, compactness theorem, host real equality row, or choice operator would require a separate coordinate, and none is present in the routed boundary. □

Theorem 320.10 (NestedInterval consumer bridge finite-route exactness)**.** *Every public bridge read from the boundary of Definition 320.9 factors through exactly one displayed finite route: a dyadic endpoint row, endpoint-order row, width row, adjacent inclusion row,* $\mathsf{StreamName}^{\uparrow}$ *schedule row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row,* $\mathsf{Real}^{\uparrow}$ *seal-facing row, visible* hsame *or* Cont *transport,* Pkg *provenance row, or* NameCert *row.*

No other public bridge coordinate is available. In particular, the route has no infinite intersection read, compactness read, host real equality read, or choice read.

Lean4 ✓ `BEDC.Derived.NestedIntervalUp.NestedIntervalPacket consumer bridge finite route exactness`

Proof. Start with the public finite-window boundary of Definition 320.8. It lists the only consumer-visible coordinates of the packet: displayed endpoint, order, width, inclusion, schedule, handoff, seal-facing, transport, package, and name-certificate rows.

Use Definition 320.4 for the $\mathsf{RegSeqRat}^{\uparrow}$ consumer route. This pins every regular-rational read to the scheduled finite endpoint and width rows already displayed by the public boundary.

Use Definition 320.6 for the $\mathsf{Real}^{\uparrow}$ consumer route. This pins every real-facing seal read to the same finite endpoint, inclusion, schedule, handoff, transport, provenance, and certificate rows.

Finally apply Definition 320.9. Since the bridge boundary is the union of those displayed finite routes and the transport/provenance rows named above, any public bridge read is one of those rows. An infinite intersection, compactness principle, host real equality, or choice operator would require an extra coordinate not present in the boundary. □

Theorem 320.11 (NestedInterval shared-window RegSeqRat endpoint cofinality)**.** *Let P and Q be accepted* $\mathsf{NestedInterval}^{\uparrow}$ *packets with finite windows that overlap at a scheduled unary observation n. Suppose both packets display the dyadic endpoint rows, width and adjacent-inclusion ledgers,* $\mathsf{StreamName}^{\uparrow}$ *schedule rows,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}$ *provenance required by Definition 320.10. Then the overlapping observation exposes the same* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint row in both packets up to* hsame *transport:*

$$\mathsf{Schedule}(n, P) \wedge \mathsf{Schedule}(n, Q) \Longrightarrow \mathsf{hsame}(\mathsf{RegSeqRatEndpoint}(P, n), \mathsf{RegSeqRatEndpoint}(Q, n)).$$

The comparison reads only the displayed dyadic endpoints, width ledgers, inclusion ledgers, schedule rows, continuation routes, package provenance, and name-certificate rows. It does not read a completed intersection point, compactness datum, host real equality row, or choice operator.

Proof. First restrict both accepted packets to the shared scheduled observation using Definition 320.3. The restriction keeps exactly the displayed dyadic endpoints, width rows, adjacent-inclusion rows, $\mathsf{StreamName}^{\uparrow}$ schedule entries, Cont routes, Pkg rows, and $\mathsf{NameCert}$ provenance attached to the overlap.

Next apply Definition 320.4 to each restricted packet. This turns the shared finite endpoint and width data into the $\mathsf{RegSeqRat}^{\uparrow}$ endpoint rows read by the two consumers.

Compare the two reads through Definition 320.10. Since the consumer bridge has only one available finite route at the overlapping scheduled observation, both regular-rational endpoint rows are transported through the same displayed route and are therefore identified by hsame.

Finally use Definition 320.12 to exclude any alternative coordinate. A completed intersection, compactness principle, host real equality, or choice-selected point would require data outside the accepted finite-row bridge, so it cannot affect the shared-window endpoint comparison. □

Theorem 320.12 (NestedInterval standard finite-row bridge acceptance)**.** *Any standard nested-interval bridge accepted for this chapter factors through exactly the public finite-window boundary of Definition 320.8, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer rows of Definition 320.9, and the real seal boundary of Definition 320.6. The accepted reading consists only of displayed dyadic endpoints, endpoint-order rows, width rows, adjacent inclusion rows, schedule rows, regular-rational hand-off rows, real seal-facing rows,* hsame *and* Cont *transport rows,* Pkg *provenance, and* $\mathsf{NameCert}$ *provenance.*

The bridge exposes no infinite intersection point, compactness theorem, host real equality, Heine–Borel principle, or choice operator.

Proof. Project the accepted packet through the carrier of Definition 320.1. The projection exposes precisely the dyadic endpoint packets, endpoint-order ledgers, width ledgers, adjacent inclusion

rows, schedule provenance, regular-rational handoff rows, seal-facing rows, and visible provenance coordinates.

Use Definition 320.8 to identify these coordinates as the exported public finite-window rows. This fixes the standard finite reading at the level of displayed endpoints, width and inclusion ledgers, schedules, transports, package provenance, and name-certificate rows.

Use Definition 320.9 for the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ routing. The consumer bridge reads the same finite rows through the regular-rational handoff and the seal-facing boundary, so it supplies no additional coordinate beyond the public boundary.

Finally use Definition 320.6 to identify the real-facing rows as finite seal reads rather than host real equalities. Thus the accepted standard bridge is exactly the finite public and consumer route.

An infinite intersection point, compactness theorem, host real equality row, Heine–Borel principle, or choice operator would require a carrier coordinate outside Definition 320.1, and no such coordinate is present. □

Theorem 320.13 (NestedInterval Real seal nonselection)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{NestedInterval}^{\uparrow}$ *packet factors through finite dyadic interval rows, unary prefix data,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *reads, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary. More precisely, the read is contained in the public finite-window boundary of Definition 320.8 and the seal-facing route of Definition 320.6.*

Consequently the packet cannot export a chosen intersection point, a host real intersection, a quotient representative, compactness data, Heine–Borel data, or a choice operator.

Proof. Start with the public finite-window boundary of Definition 320.8. It lists the only consumer-visible rows: dyadic endpoint rows, endpoint-order rows, width rows, adjacent inclusion rows, schedule rows, regular-rational handoff rows, seal-facing rows, transport rows, provenance rows, and name-certificate rows.

Apply Definition 320.6. The $\mathsf{Real}^{\uparrow}$ seal can be reached only from the displayed finite endpoint, width, inclusion, schedule, and $\mathsf{RegSeqRat}^{\uparrow}$ rows already present in that public boundary.

The carrier therefore exposes finite windows and seal rows, not a point selected from an infinite intersection. A host real intersection, quotient representative, compactness datum, Heine–Borel datum, or choice operator would require an additional coordinate outside the displayed finite-row boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}$**.**

Constructive story: A nested-interval packet is generated by finite BHist dyadic interval rows indexed by a unary prefix, lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, endpoint-order and width ledgers, adjacent nested-inclusion rows, $\mathsf{StreamName}^{\uparrow}$ schedule provenance, $\mathsf{RegSeqRat}^{\uparrow}$ finite-window reads, $\mathsf{Real}^{\uparrow}$ seal-facing rows, and Pkg and NameCert provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 320.12 is the mature finite-row surface binding the standard nested-interval reading to the public finite-window boundary, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer rows, seal boundary, BHist, hsame, Cont, Pkg, and NameCert provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedInterval}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.NestedIntervalUp.NestedIntervalFiniteCarrier\_public\_window\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Cantor nested-interval convergence, completeness, compactness, Heine–Borel, extraction of a point by choice, or host real intersections.

Upgrade path: The remaining work is formal verification of the finite standard bridge and later checked bridge status.

Chapter 321

A Concrete Naming Certificate for $\mathsf{FenchelDuality}^{\uparrow}$

The $\mathsf{FenchelDuality}^{\uparrow}$ interface names the finite convex-analysis certificate surface for a displayed primal functional, a displayed dual functional, and their pairing ledger. In Rockafellar's convex-analysis presentation and in Boyd–Vandenberghe 2004, Fenchel duality is read through convex functions, epigraphs, conjugate pairings, and extended-real comparisons; the BEDC packet below keeps only the finite rows needed to name that surface. It depends on $\mathsf{NameCert}_{\mathsf{ConvexSet}^{\uparrow}}$ (chapter 208), $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ (chapter 47), $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ (chapter 48), $\mathsf{NameCert}_{\mathsf{LPDuality}^{\uparrow}}$ (chapter 235), $\mathsf{NameCert}_{\mathsf{TopVecSpace}^{\uparrow}}$ (chapter 205), $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ (chapter 38), and $\mathsf{NameCert}_{\mathsf{SemidefiniteCone}^{\uparrow}}$ (chapter 310).

321.1 Finite primal-dual packet

Definition 321.1 (FenchelDualityUp carrier)**.** *A* $\mathsf{FenchelDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, D, E, Q, C, R, L)$$

with the following visible rows:

1. *P is a convex primal functional row over a displayed* $\mathsf{TopVecSpace}^{\uparrow}$ *or* $\mathsf{VecSpace}^{\uparrow}$ *domain;*
2. *D is a convex dual functional row over the displayed dual-facing space;*
3. *E is a pairing row that records the bilinear evaluation used to compare primal and dual witnesses;*
4. *Q records order and extended-value comparison rows through* $\mathsf{Real}^{\uparrow}$*;*
5. *C records convex epigraph rows through* $\mathsf{ConvexSet}^{\uparrow}$ *for the primal and dual functionals;*
6. *R records optional cone-facing compatibility rows through* $\mathsf{SemidefiniteCone}^{\uparrow}$ *when the functional packet is consumed by semidefinite or conic optimization;*
7. *L is the* Pkg *provenance and readback ledger tying P, D, E, Q, C, and R to the displayed source certificates and to the finite* hsame *and* Cont *routes by which those rows may be transported.*

The carrier predicate is $\mathsf{FenchelPacket}(P, D, E, Q, C, R, L)$*. Two accepted packets classify together only when the displayed primal row, dual row, pairing row, comparison row, epigraph rows, optional cone-facing rows, and provenance ledger transport by* hsame *along the same* Cont *routes named in L.*

Definition 321.2 (Fenchel duality carrier)**.** *A paper-facing* $\mathsf{FenchelDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, f, d, \ell, g, w, z, t, \pi, n)$$

with the following displayed rows:

1. X *is the* $\mathsf{VecSpace}^{\uparrow}$ *source row carrying the primal vector-space boundary.*
2. f *is the primal convex function row over the displayed* $\mathsf{ConvexSet}^{\uparrow}$ *domain.*
3. d *is the dual functional row, exposed as a* $\mathsf{LinMap}^{\uparrow}$*-facing read rather than as an ambient dual space.*
4. ℓ *is the linear pairing row connecting primal witnesses to dual reads.*
5. g *is the conjugate ledger row, recording the finite Fenchel conjugacy data available to consumers.*
6. w *is the weak-duality inequality row, read through* $\mathsf{Real}^{\uparrow}$ *comparisons.*
7. z *is the zero-gap witness row when the packet displays equality of primal and dual ledger values.*
8. t *records classifier transport for* $X, f, d, \ell, g, w, z, \pi, n$.
9. π *is the* Pkg *provenance over* $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. n *is the local* $\mathsf{NameCert}_{\mathsf{FenchelDuality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, f, d, \ell, g, w, z, t, \pi, n$. *It has no coordinate for a selected optimizer, Hahn–Banach extension, separating hyperplane, topological dual completion, or analytic strong-duality theorem.*

Theorem 321.3 (FenchelDualityUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FenchelDuality}^{\uparrow}}$ *seed surface consists of the following five finite obligations:*

1. *carrier row: the* BHist *packet of Definition 321.1, including the displayed primal functional, dual functional, pairing, comparison, epigraph, cone-facing, and provenance coordinates;*
2. *classifier row: componentwise* hsame *on exactly those displayed coordinates, with no comparison of hidden topological vector spaces, host quotient spaces, or unrecorded conjugate functions;*
3. *stability row: whenever each displayed coordinate is transported along the* hsame *witnesses and* Cont *routes named by the ledger, the transported packet is again an accepted* $\mathsf{FenchelDuality}^{\uparrow}$ *carrier;*
4. *exact ledger readback: every primal-dual pairing, extended-value comparison, and epigraph membership claim read by a consumer must be one of the rows recorded in the packet or a finite readback through its* Pkg *provenance;*
5. *consumer boundary: the surface does not claim separation theorems, Hahn–Banach, strong duality existence, host quotient spaces, analytic closedness of epigraphs, or an external conjugacy theorem.*

These obligations define only the BEDC-kernel naming surface for Fenchel-style primal-dual data; they do not assert the analytic hypotheses under which a classical duality theorem has equality of optimal values.

Lean4 ✓ `BEDC.Derived.FenchelDualityUp.FenchelDualityCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.FenchelDualityUp.FenchelDualityPrimalDualPacket namecert obligation surface`

Proof. Read the carrier from Definition 321.1. Its coordinates list exactly the rows a Fenchel-style consumer can inspect: primal and dual functional rows, a pairing row, real-valued or extended-valued comparison rows, convex epigraph rows, optional cone-facing compatibility rows, and package provenance.

The classifier and stability obligations are componentwise. Transporting an accepted packet along the displayed hsame witnesses replaces representatives of the same visible rows, and the Cont routes named in the ledger keep the reads inside the cited $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, and $\mathsf{SemidefiniteCone}^{\uparrow}$ certificate boundaries.

The readback clause is therefore exact: a consumer obtains a pairing value, an order comparison, or an epigraph membership only by projecting the corresponding finite ledger row. Since no coordinate stores a separating hyperplane principle, a Hahn–Banach extension, a strong-duality existence witness, a host quotient, or an analytic closure theorem, the excluded classical principles remain outside the seed surface. □

Theorem 321.4 (Fenchel duality NameCert obligations)**.** *For every accepted paper-facing carrier*

$$K = (X, f, d, \ell, g, w, z, t, \pi, n)$$

of Definition 321.2, the local $\mathsf{NameCert}_{\mathsf{FenchelDuality}^{\uparrow}}$ *surface consists exactly of primal convex-function exposure through* f*, dual functional exposure through* d*, linear pairing through* ℓ*, conjugate-ledger readback through* g*, weak-duality comparison through* w*, zero-gap witness exposure through* z*, classifier transport through* t*, provenance through* π*, and local naming through* n*.*

Proof. Project the accepted carrier of Definition 321.2. The rows X, f, d, ℓ expose the primal vector-space boundary, convex functional, dual functional, and pairing data through the displayed $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ certificate routes.

Next read the convex-duality ledger rows. The conjugate row g, weak-duality row w, and optional zero-gap row z are finite packet coordinates, so a consumer receives only recorded $\mathsf{Real}^{\uparrow}$ comparisons and ledger witnesses.

Finally the transport, provenance, and naming rows t, π, n preserve that surface under classifier comparison. Since the carrier has no coordinate for selected optimizers, Hahn–Banach, separating hyperplanes, topological dual completion, or an analytic strong-duality theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 321.5 (FenchelDualityUp carrier classifier scope)**.** *Every accepted* $\mathsf{FenchelDuality}^{\uparrow}$ *packet exposes the visible* BHist *carrier rows for the primal functional, dual functional, pairing,* $\mathsf{Real}^{\uparrow}$ *comparison,* $\mathsf{ConvexSet}^{\uparrow}$ *epigraph data, optional* $\mathsf{SemidefiniteCone}^{\uparrow}$ *compatibility, and* Pkg *provenance named in Definition 321.1. Its classifier is exactly the componentwise* hsame *classifier on those displayed rows along the finite* Cont *routes recorded by the provenance ledger.*

Proof. Project the accepted packet along the coordinates of Definition 321.1. The definition lists P, D, E, Q, C, R, and L as the full visible carrier, so each consumer-facing carrier read is one of those rows.

Apply Definition 321.3 to identify the same carrier coordinates as the local naming-certificate surface. Since the obligation surface records no further coordinate, the exposed carrier and classifier scope is exactly the displayed primal, dual, pairing, comparison, epigraph, compatibility, provenance, hsame, and Cont packet. □

Theorem 321.6 (FenchelDualityUp hsame stability scope)**.** *If the displayed primal row, dual row, pairing row,* $\mathsf{Real}^{\uparrow}$ *comparison row,* $\mathsf{ConvexSet}^{\uparrow}$ *epigraph rows, optional* $\mathsf{SemidefiniteCone}^{\uparrow}$ *compatibility row, and* Pkg *provenance of an accepted* $\mathsf{FenchelDuality}^{\uparrow}$ *packet are transported componentwise by the* hsame *witnesses and* Cont *routes named in the ledger, then the transported packet remains an accepted carrier and has the same classifier surface.*

Proof. Use the classifier clause in Definition 321.1. It classifies packets only by componentwise hsame on P, D, E, Q, C, R, and L along the Cont routes already named by the provenance ledger.

Then apply the stability row of Definition 321.3. Transport of each displayed coordinate stays inside the finite $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, and $\mathsf{SemidefiniteCone}^{\uparrow}$ certificate boundaries, so the transported packet is accepted by the same carrier predicate and no additional classifier field is introduced. □

Theorem 321.7 (FenchelDualityUp primal dual ledger readback)**.** *Every certified* $\mathsf{FenchelDuality}^{\uparrow}$ *answer packet reads back the primal row, dual row, pairing row, comparison ledger, epigraph row, and provenance ledger named by its source packet. The readback introduces no Hahn–Banach principle, separation theorem, strong-duality existence witness, host quotient space, or analytic closedness assertion.*

Proof. Project the ledger coordinate L of the carrier. By Definition 321.1, every consumer readback is tied to the displayed primal, dual, pairing, comparison, epigraph, optional cone-facing, and provenance rows by finite hsame and Cont routes.

Apply the exact ledger readback and consumer-boundary clauses of Definition 321.3. Those clauses allow only the recorded pairing, extended-value comparison, and epigraph membership rows to be read, and they explicitly leave Hahn–Banach, separation, strong duality, host quotient spaces, and analytic closedness outside the certificate surface. □

Theorem 321.8 (FenchelDualityUp mature consumer route)**.** *Every paper-side consumer of the* $\mathsf{FenchelDuality}^{\uparrow}$ *bridge reads the finite primal-dual packet only through the already displayed* $\mathsf{ConvexSet}^{\uparrow}$ *epigraph rows,* $\mathsf{LPDuality}^{\uparrow}$ *linear-duality rows,* $\mathsf{TopVecSpace}^{\uparrow}$ *vector-space boundary,* $\mathsf{Real}^{\uparrow}$ *comparison ledger, and optional* $\mathsf{SemidefiniteCone}^{\uparrow}$ *compatibility rows. The reusable surface is therefore the mature convex optimization bridge recorded by* P, D, E, Q, C, R, L*, not a separation theorem, Hahn–Banach principle, strong-duality existence witness, host quotient space, or analytic closedness assertion.*

Proof. Start from the carrier classifier scope of Definition 321.5. It exposes exactly the primal row, dual row, pairing row, real comparison row, convex epigraph rows, optional cone-facing row, and provenance ledger.

Use hsame stability from Definition 321.6. Transport of the packet stays inside the displayed $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{SemidefiniteCone}^{\uparrow}$ certificate boundaries, so each named consumer reads the same finite bridge rather than a host analytic object.

Finally apply the primal-dual ledger readback of Definition 321.7. Since the ledger introduces no separation theorem, Hahn–Banach principle, strong-duality existence witness, host quotient space, or analytic closedness assertion, the paper-side consumer route is exhausted by the displayed mature bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{FenchelDuality}^{\uparrow}}$**.**

Constructive story: A Fenchel-duality packet is generated by displayed primal and dual convex functional rows, a pairing row over the visible vector-space boundary, $\mathsf{Real}^{\uparrow}$ extended-value comparison rows, $\mathsf{ConvexSet}^{\uparrow}$ epigraph rows, optional $\mathsf{SemidefiniteCone}^{\uparrow}$ compatibility rows, and Pkg provenance with finite hsame and Cont transport.

Theory closure: TheoryGate(NameCert$_{\mathsf{FenchelDuality}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped FenchelDuality$^{\uparrow}$ paper surface covers the carrier classifier scope, hsame stability scope, primal-dual ledger readback, and the displayed BHist, Cont, hsame, Pkg, and NameCert rows of Definition 321.1.

Formal verification: FormalStatus(NameCert$_{\mathsf{FenchelDuality}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FenchelDualityUp.FenchelDualityPrimalDualPacket\_namecert\_obligation\_surface`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close separation theorems, Hahn–Banach, strong duality existence, host quotient spaces, analytic closedness, or an external conjugacy theorem.

Upgrade path: No further paper-side closure grade remains for this chapter; bridge checking is recorded only on the independent verification axis.

Chapter 322

$\mathsf{DyadicPrecision}^{\uparrow}$ naming certificate construction

The $\mathsf{DyadicPrecision}^{\uparrow}$ surface names the finite precision schedule that sits between $\mathsf{DyadicRatCore}^{\uparrow}$ radius rows, $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ regularity checks, and the $\mathsf{Real}^{\uparrow}$ seal boundary. The carrier below is a BEDC packet of visible rows only: a finite unary precision request, the dyadic radius it selects, the finite stream window it opens, and the provenance ledger through which later real-name constructions read the same finite data.

The schedule is the finite reader shared by chapter 291, chapter 126, chapter 301, and chapter 38. It consumes dyadic radius rows and stream observation windows before $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ read the same finite request through their regularity and seal boundaries.

322.1 Finite precision schedule

Definition 322.1 (Dyadic precision schedule carrier)**.** *A* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedule carrier is a finite* BHist *packet*

$$S = (n, \rho, W, H, C, P, N, L)$$

with the following visible rows:

1. *n is a unary precision index row, read as a finite* $\mathsf{UnaryHistory}$ *packet and inhabited by the empty precision row or by iterating the unary successor constructor;*
2. *ρ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row selected by n, normally read as the displayed dyadic tolerance 2^{-n} with its denominator ledger;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row naming exactly the finite stream positions requested at precision n;*
4. *H records the componentwise* hsame *witnesses for the unary index, dyadic radius, and finite observation window;*
5. *C records the displayed* Cont *routes that transport the radius and window rows without changing the selected precision request;*
6. *P is the* Pkg *provenance tying the schedule to the visible* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source boundaries;*
7. *N is the local* $\mathsf{NameCert}$ *row for the schedule packet;*

8. *L is the ledger stating that every consumer-visible precision, radius, and window read is one of the rows above.*

The carrier predicate is $\mathsf{DyadicPrecisionSchedule}(n, \rho, W, H, C, P, N, L)$. *The packet has no quotient of precision requests, no completed real object, no hidden Cauchy modulus, and no external choice of an infinite stream tail.*

Lean4 (definition) `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionUp`
Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.taste gate`

Definition 322.2 (Dyadic precision schedule classifier)**.** *For accepted carriers*

$$S = (n, \rho, W, H, C, P, N, L) \quad \text{and} \quad S' = (n', \rho', W', H', C', P', N', L'),$$

the predicate $\mathsf{DyadicPrecisionClassifier}(S, S')$ *is componentwise finite sameness. It consists of displayed* hsame *rows identifying the unary precision indices n and n', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows ρ and ρ', the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W and W', and the package ledgers P and P'.*

The classifier transports only those rows and the attached Cont *routes. It does not compare hidden tails of streams, identify regular sequences by a quotient, or assert equality of* $\mathsf{Real}^{\uparrow}$ *names.*

Theorem 322.3 (Dyadic precision schedule NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicPrecision}^{\uparrow}}$ *seed surface consists of five finite obligations:*

1. *carrier inhabitation is generated from the empty precision row and the unary successor row, together with the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row and the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected by that row;*
2. *classifier reflexivity, symmetry, and transitivity are the componentwise laws for the displayed* hsame *rows on the unary index, dyadic radius, finite window, and package ledger;*
3. *carrier transport along the classifier repacks the same schedule with transported* hsame *and* Cont *rows;*
4. *ledger exactness says that every radius read and every finite observation-window read is one of the displayed ρ and W rows recorded in the carrier ledger;*
5. *source-domain visibility says that every consumed row remains inside the named* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source boundaries recorded by* Pkg *and* NameCert.

These obligations define only the finite precision-schedule certificate surface.

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionUp semantic name certificate`

Proof. For carrier inhabitation, begin with the empty unary row and attach the base dyadic radius and the empty observation window. A successor precision request appends one unary row, refines the visible dyadic radius ledger through $\mathsf{DyadicRatCore}^{\uparrow}$, and opens the finite window requested by $\mathsf{StreamName}^{\uparrow}$.

The classifier laws are obtained by applying reflexivity, symmetry, and transitivity componentwise to the displayed hsame rows. Each law acts on the unary index, radius row, window row, and package ledger, so the result remains a finite schedule classifier.

Transport follows by repacking the carrier after those componentwise moves. Ledger exactness is projection from L: a consumer can read a precision, radius, or window only by selecting one stored row. The provenance row P and the local naming row N keep those projections inside the stated source domains and add no hidden real-number principle. □

Theorem 322.4 (Dyadic precision schedule RegSeqRat handoff)**.** *An accepted* DyadicPrecision$^{\uparrow}$ *schedule supplies exactly the finite precision and observation-window rows consumed by a* RegSeqRat$^{\uparrow}$ *regularity check. The handoff consists of the unary index row n, the* DyadicRatCore$^{\uparrow}$ *radius row ρ, the finite* StreamName$^{\uparrow}$ *window W, the componentwise* hsame *comparisons, the displayed* Cont *routes, and the* Pkg *and* NameCert *provenance attached to those rows.*

The handoff does not supply a quotient sequence, a completed metric point, an infinite stream tail, or any additional regularity principle beyond the displayed finite rows.

Proof. Project an accepted schedule through Definition 322.1. The rows relevant to RegSeqRat$^{\uparrow}$ are exactly the precision index, dyadic radius, and finite stream window, together with their transport and provenance rows.

A RegSeqRat$^{\uparrow}$ regularity check consumes those rows by reading endpoints at the selected finite window and comparing them against the displayed dyadic tolerance. The schedule contributes no other coordinate, so the regularity check cannot obtain a quotient sequence, a completion point, or an unrecorded tail from the handoff. □

Theorem 322.5 (Dyadic precision schedule RealUp boundary)**.** *The same finite* DyadicPrecision$^{\uparrow}$ *schedule can be read at the* Real$^{\uparrow}$ *seal boundary as a precision request, a dyadic radius row, a finite observation-window row, and the visible* hsame*,* Cont*,* Pkg*, and* NameCert *provenance of those rows.*

Reading the schedule at the seal boundary does not add Dedekind cuts, quotient reals, trichotomy, completeness, host real equality, or any standard bridge datum.

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule realup boundary`

Proof. Use Definition 322.4 to identify the finite rows passed to the regular rational source. The Real$^{\uparrow}$ seal reads that source through the same scheduled precision, radius, and window rows.

Since the schedule carrier has no coordinate for a cut, quotient, order trichotomy, completeness witness, host real equality, or bridge object, none can appear when the same rows are read at the seal boundary. The boundary is therefore the finite precision schedule itself, viewed through the displayed provenance. □

Theorem 322.6 (DyadicPrecision obligation closure)**.** *The carrier of Definition 322.1, classifier of Definition 322.2, local* NameCert *obligations of Definition 322.3,* RegSeqRat$^{\uparrow}$ *handoff of Definition 322.4,* Real$^{\uparrow}$ *boundary of Definition 322.5, and ledger-exactness rows exhaust the obligation surface for finite dyadic precision schedules.*

Because this chapter is marked with AI origin, promotion beyond seed remains conditional on the future TasteGate-backed Lean target. The closure packet adds no hidden tail, quotient real, trichotomy, completeness row, or bridge row.

Proof. Project the carrier of Definition 322.1. Its visible rows are the unary precision index, dyadic radius row, finite observation window, transport rows, provenance rows, local naming row, and exact schedule ledger.

Use Definition 322.2 and Definition 322.3 for stability, carrier transport, ledger exactness, and source-domain visibility. These rows account for every local schedule obligation.

The outgoing consumers are Definition 322.4 and Definition 322.5. They read only the same finite precision, radius, window, transport, provenance, and naming rows, so no hidden tail, quotient real, trichotomy, completeness datum, or bridge row is present. □

Theorem 322.7 (Dyadic precision empty schedule exactness)**.** *Let*

$$S = (n, \rho, W, H, C, P, N, L)$$

be a DyadicPrecision$^{\uparrow}$ *schedule carrier in the sense of Definition 322.1. Suppose S is obtained from the empty unary precision branch of the carrier-inhabitation obligation in Definition 322.3. Then n is the empty unary precision row, ρ is the base* DyadicRatCore$^{\uparrow}$ *dyadic radius row selected by that empty row, and W is the empty* StreamName$^{\uparrow}$ *observation-window row.*

Consequently, every consumer-visible precision, radius, or observation-window read from S is exhausted by those three displayed rows, together with the provenance ledger L and the visible hsame*,* Cont*,* Pkg*, and* NameCert *rows already carried by S.*

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionEmptySchedule exactness`
Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule empty branch readback`

Proof. Project the schedule carrier through Definition 322.1. The visible coordinates of S include exactly one unary precision index row n, one selected DyadicRatCore$^{\uparrow}$ radius row ρ, one finite StreamName$^{\uparrow}$ observation-window row W, the componentwise hsame witnesses H, the displayed Cont routes C, the provenance package P, the local naming row N, and the ledger L.

Now restrict to the empty unary precision branch of the carrier-inhabitation obligation in Definition 322.3. In that branch there is no successor precision request. The carrier therefore has no later unary row from which to select a refined tolerance and no requested stream position from which to open a nonempty observation window. The selected radius coordinate is exactly the base dyadic radius row supplied by DyadicRatCore$^{\uparrow}$ for the empty precision row, and the selected window coordinate is exactly the empty StreamName$^{\uparrow}$ observation window.

It remains only to check consumer coverage. By the ledger clause of Definition 322.1, every consumer-visible precision, radius, and window read is one of the displayed schedule rows recorded by L. In the empty branch those displayed rows are precisely the empty unary row n, the base dyadic radius row ρ, and the empty observation window W, with the already-carried H, C, P, and N rows providing transport and provenance. Since the carrier has no quotient of precision requests, no completed real object, no hidden Cauchy modulus, and no external stream tail, there is no further coordinate from which an additional consumer read could be obtained. □

Theorem 322.8 (Dyadic precision successor schedule exactness)**.** *Let*

$$S = (n, \rho, W, H, C, P, N, L)$$

be a DyadicPrecision$^{\uparrow}$ *schedule carrier in the sense of Definition 322.1. Suppose S is obtained from the successor unary precision branch of the carrier-inhabitation obligation in Definition 322.3. Then the successor row is the displayed unary extension of its predecessor precision row, ρ is the transported and refined* DyadicRatCore$^{\uparrow}$ *radius row selected by that successor, and W is the finite* StreamName$^{\uparrow}$ *observation window opened by that successor.*

Consequently, every consumer-visible precision, radius, or observation-window read from the successor branch factors through n, ρ, W, H, C, P, N, and L. The successor branch supplies no hidden Cauchy modulus, completed real object, quotient precision request, external stream tail, trichotomy, completeness, or bridge datum.

Proof. Project the schedule carrier through Definition 322.1. The projected rows are the unary index n, the selected radius row ρ, the finite observation window W, the hsame witnesses H, the displayed Cont routes C, the provenance row P, the local naming row N, and the exact ledger L.

Now restrict to the successor unary precision branch of Definition 322.3. That branch appends one unary successor row to the predecessor precision row. The same branch selects the refined $\mathsf{DyadicRatCore}^{\uparrow}$ radius row and opens the finite $\mathsf{StreamName}^{\uparrow}$ observation window requested at that successor precision.

The displayed hsame witnesses and Cont routes transport the predecessor schedule rows to the successor rows without changing the finite source surface. Ledger exactness says that every consumer-visible precision, radius, and window read is one of the displayed rows recorded by L, while source-domain visibility keeps those rows inside the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ boundaries named by P and N.

The carrier has no coordinate for hidden stream tails, quotient precision requests, completed reals, trichotomy, completeness, or bridge data. Thus every consumer read from the successor branch factors through the displayed tuple $(n, \rho, W, H, C, P, N, L)$. □

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule successor schedule exactness`

Theorem 322.9 (Dyadic precision monotone refinement handoff)**.** *If a* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedule carrier refines a unary precision row through the displayed* hsame *and* Cont *routes, then the transported radius row and observation-window row form the consumer-visible refinement surface for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*. Every public read factors through the carrier rows* n*,* ρ*,* W*,* H*,* C*,* P*,* N*, and* L*.*

The handoff exposes no hidden Cauchy modulus, completed real object, or external stream tail.

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionScheduleSurface monotone refinement handoff`

Proof. Unpack the schedule carrier of Definition 322.1. Its visible rows already name the unary precision request, selected dyadic radius, finite observation window, componentwise hsame witnesses, displayed Cont routes, package provenance, local NameCert row, and exact ledger.

Use the classifier of Definition 322.2 to read a refinement as componentwise movement of those finite rows. The transported radius and transported observation window remain the displayed ρ and W coordinates, carried by H and C rather than by an external sequence model.

The consumer boundaries are Definition 322.4 and Definition 322.5. They show that $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consume the same precision, radius, window, transport, provenance, and naming rows, so every public read factors through the carrier ledger L.

The empty case of Definition 322.7 is the base instance of this surface. In the successor case the same carrier rows are refined through the displayed routes, and no hidden Cauchy modulus, completed real object, or external stream tail is introduced. □

Theorem 322.10 (DyadicPrecision common-window readback determinacy)**.** *Let*

$$S = (n, \rho, W, H, C, P, N, L)$$

and

$$S' = (n', \rho', W', H', C', P', N', L')$$

be accepted $\mathsf{DyadicPrecision}^{\uparrow}$ *schedules. If* $\mathsf{DyadicPrecisionClassifier}(S, S')$ *holds, then every shared unary precision-window read consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or by the* $\mathsf{Real}^{\uparrow}$ *seal factors through the same transported rows* $n, \rho, W, H, C, P, N, L$ *and* $n', \rho', W', H', C', P', N', L'$*.*

In particular, the consumer-visible precision, selected radius, observation-window, hsame *transport,* Cont *route,* Pkg *provenance, and* NameCert *provenance are determined by the displayed classifier rows. The readback exposes no hidden stream tail, completed real object, quotient precision request, external Cauchy modulus, trichotomy, completeness witness, or bridge datum.*

Proof. Project both schedules through Definition 322.1. The only consumer-facing rows are the unary precision row, selected dyadic radius row, finite observation window, componentwise hsame witnesses, displayed Cont routes, package provenance, local naming row, and exact ledger.

Use the classifier of Definition 322.2. It transports those coordinates componentwise, so a shared unary precision-window request has the same displayed source rows on both sides after applying the recorded hsame witnesses and Cont routes.

Now apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff in Definition 322.4 and the $\mathsf{Real}^{\uparrow}$ boundary in Definition 322.5. Both consumer surfaces read the transported precision, radius, window, transport, provenance, and naming rows already listed by the two schedule carriers.

The empty branch is covered by Definition 322.7, and the successor branch by Definition 322.8. Finally use Definition 322.9 to identify a refined shared window with the same displayed rows. Since none of these rows contains an external stream tail, completed real object, quotient request, or hidden modulus, the shared-window readback is determined exactly by the two displayed classifier tuples. □

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule common window readback`

Theorem 322.11 (DyadicPrecision schedule obligation export)**.** *The carrier of Definition 322.1, classifier of Definition 322.2, local* NameCert *obligations of Definition 322.3,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 322.4,* $\mathsf{Real}^{\uparrow}$ *boundary of Definition 322.5, and empty-schedule exactness of Definition 322.7 together with successor-schedule exactness of Definition 322.8 form the full local obligation surface for the finite dyadic precision schedule.*

Every obligation is read inside a finite BHist *packet and is transported only through the displayed* hsame *and* Cont *rows, with* Pkg *provenance and local* NameCert *naming over the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer surfaces.*

Proof. Project the schedule through the carrier and classifier. The visible rows are the unary precision request, dyadic radius, finite observation window, componentwise hsame witnesses, displayed Cont routes, package provenance, local naming row, and exact ledger.

Apply the local NameCert obligation theorem to account for carrier inhabitation, classifier laws, transport, ledger exactness, and source-domain visibility. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ boundary then read the same precision, radius, window, transport, provenance, and naming rows as consumer surfaces.

The empty-schedule and successor-schedule exactness theorems supply the two constructor-boundary cases for the finite schedule. Together these rows exhaust the local obligation export, because the carrier has no hidden stream tail, quotient real, trichotomy, completeness datum, or bridge coordinate. □

Theorem 322.12 (DyadicPrecision TasteGate carrier witness)**.** *For any future* ChapterTasteGate *instance over* $\mathsf{DyadicPrecision}^{\uparrow}$, *the source scope is exactly the carrier of Definition 322.1, classifier of Definition 322.2, obligation rows of Definition 322.3, handoff of Definition 322.4,* $\mathsf{Real}^{\uparrow}$ *boundary of Definition 322.5, and constructor-boundary exactness from Definition 322.7 and Definition 322.8.*

Equivalently, if

$$S = (n, \rho, W, H, C, P, N, L)$$

is an accepted schedule carrier, then every candidate TasteGate event-flow read is sourced by the finite precision row n, *dyadic radius row* ρ, *observation-window row* W, *displayed* hsame *rows* H, *displayed* Cont *routes* C, Pkg *provenance* P, *local* NameCert *row* N, *and exact ledger* L. *It*

cannot read a hidden stream tail, quotient real equality, trichotomy, completeness, selected limit, or standard bridge row.

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionScheduleTasteGate visible rows`

Proof. Project the accepted schedule through the finite BHist packet of Definition 322.1. The projection lists the precision, radius, observation-window, transport, provenance, name, and ledger rows as the whole carrier tuple, so these rows give the source tokens of the event-flow reading.

Use the classifier of Definition 322.2 to transport the event-flow componentwise. The classifier moves only the displayed unary precision rows, dyadic radius rows, finite observation-window rows, package ledgers, and their attached hsame and Cont witnesses, so no additional event source is created by transport.

Now apply Definition 322.11. Its row exhaustion statement identifies the carrier, classifier, local NameCert obligations, consumer handoffs, empty branch, successor branch, and exact ledger as the full local schedule surface. Thus every candidate ChapterTasteGate read factors through the displayed hsame, Cont, Pkg, and NameCert rows.

The ground-compiler reading of the same packet is therefore the no-hidden-input condition together with conservativity over those visible rows. Since no coordinate of the carrier contains an infinite tail, quotient real equality, trichotomy, completeness witness, selected limit, or standard bridge datum, the TasteGate obligations cannot obtain one from the schedule. □

Theorem 322.13 (DyadicPrecision TasteGate compiler-obligation handoff)**.** *Let*

$$S = (n, \rho, W, H, C, P, N, L)$$

be an accepted DyadicPrecision$^{\uparrow}$ *schedule packet. Any future* ChapterTasteGate *proof for this AI-origin chapter must discharge the no-hidden-input, event-flow conservativity, flow-level round-trip, channel bijection, and source-channel separation obligations by applying the* GroundCompiler *theorem rows* no_hidden_input*,* event_flow_conservativity*,* flow_level_round_trip*,* channel_bijection*, and* source_channel_separation *to the visible carrier event-flow tokens listed in Definition 322.12.*

Consequently the handoff exposes only the finite precision row n, dyadic radius row ρ, finite window row W, displayed hsame *rows H, displayed* Cont *routes C,* Pkg *provenance P,* NameCert *row N, and ledger L. It exposes no hidden stream tail, quotient real, selected limit, trichotomy, completeness witness, or bridge datum.*

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionUp tastegate compiler obligation handoff`

Proof. Project the accepted packet through Definition 322.1. The projection gives exactly the eight visible schedule coordinates, and Definition 322.12 reads those coordinates as the carrier event-flow tokens for a future ChapterTasteGate proof.

The compiler obligations are then not handwritten carrier facts. The future Lean route must apply no_hidden_input for hidden-source exclusion, event_flow_conservativity for event-flow preservation, flow_level_round_trip for encoding round trip, channel_bijection for channel identity, and source_channel_separation for separating source rows from channel rows.

Finally use Definition 322.11. Its absence clauses and the carrier clauses above exhaust the schedule surface by the finite precision row, dyadic radius row, finite window row, hsame rows, Cont routes, Pkg provenance, NameCert row, and ledger. Since no coordinate contains a hidden stream tail, quotient real, selected limit, trichotomy, completeness witness, or bridge datum, the handoff cannot expose one. □

Theorem 322.14 (DyadicPrecision radius-window coverage)**.** *Every accepted* DyadicPrecision$^{\uparrow}$ *schedule packet exposes exactly the unary precision row, the selected* DyadicRatCore$^{\uparrow}$ *radius row, the finite* StreamName$^{\uparrow}$ *observation-window row, the displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, local* NameCert *row, and exact ledger rows of Definition 322.1.*

The coverage is constructor-exact: the empty schedule is the empty carrier surface of Definition 322.7, every successor schedule is the successor carrier surface of Definition 322.8, and monotone refinement transports only those displayed radius and window rows through Definition 322.9.

Proof. Project the accepted packet through Definition 322.1. The projection names the unary precision coordinate, selected dyadic radius coordinate, finite observation window, componentwise hsame witnesses, displayed Cont routes, package provenance, local name, and exact ledger as the whole schedule carrier.

The empty constructor is handled by Definition 322.7. The successor constructor is handled by Definition 322.8. These two cases cover the finite schedule constructors.

Finally apply Definition 322.9. It keeps refinement inside the selected radius and finite window rows, so no hidden stream tail, quotient real equality, selected limit, trichotomy, completeness witness, or bridge datum is added by refinement. □

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule radius window coverage`

Theorem 322.15 (DyadicPrecision classifier consumer transport)**.** *For accepted schedule packets S and S', the classifier relation* DyadicPrecisionClassifier(S, S') *transports every* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer read through the displayed carrier rows only. In particular, the consumer-visible precision, radius, window,* hsame*,* Cont*,* Pkg*,* NameCert*, and ledger reads are exactly the rows listed in Definition 322.2; no consumer transport may read an additional stream tail, quotient real, selected limit, trichotomy, completeness witness, or bridge coordinate.*

Proof. Start from the classifier of Definition 322.2. Its clauses identify the transported precision, radius, window, transport, provenance, name, and ledger coordinates componentwise between S and S'.

Use Definition 322.10 to read both consumer packets through the same finite observation-window surface. The readback does not quantify over an unbounded tail, because the only window coordinate in the carrier is the displayed finite StreamName$^{\uparrow}$ row.

Apply Definition 322.11. The exported obligation surface is the same carrier, classifier, local NameCert row, handoff, boundary, and ledger surface. Thus every RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ consumer read factors through the displayed rows and through no other coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicPrecisionUp.DyadicPrecisionSchedule classifier consumer transport`

Theorem 322.16 (DyadicPrecision RegSeqRat RealUp consumer exhaustion)**.** *The* RegSeqRat$^{\uparrow}$ *handoff of Definition 322.4 together with the* Real$^{\uparrow}$ *boundary of Definition 322.5 exhausts the downstream consumer surface of an accepted* DyadicPrecision$^{\uparrow}$ *schedule packet.*

Consequently the packet exposes only the finite precision row, selected dyadic radius row, finite observation-window row, displayed hsame *rows, displayed* Cont *routes,* Pkg *provenance, local* NameCert *row, and exact ledger rows. It exposes no hidden stream tails, quotient reals, trichotomy, completeness, selected limits, or bridge data.*

Proof. First apply Definition 322.4. The RegSeqRat$^{\uparrow}$ consumer receives only the finite precision schedule, dyadic radius, finite window, displayed transport rows, provenance, name, and ledger exported by the schedule carrier.

Next apply Definition 322.5. The $\mathsf{Real}^{\uparrow}$ boundary reads the same finite rows as a boundary object and does not add quotient real equality, trichotomy, completeness, or a selected limiting real.

Finally use Definition 322.13. The TasteGate handoff sends the compiler obligations to the same visible carrier event-flow tokens, so the combined consumer surface is exhausted by the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows already named above. □

Theorem 322.17 (DyadicPrecision public consumer boundary)**.** *The public* $\mathsf{DyadicPrecision}^{\uparrow}$ *consumer boundary is the finite packet consisting of the precision index, dyadic radius row, finite observation-window row, classifier transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *boundary, exact ledger, local* NameCert *row, and the TasteGate-visible carrier-event rows of Definition 322.12.*

Every public consumer read factors through that packet by the radius-window coverage, classifier consumer transport, $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer exhaustion, and TasteGate compiler-obligation handoff. The boundary adds no closure promotion, hidden stream tail, quotient real, selected limit, trichotomy, completeness witness, or bridge datum.*

Proof. First apply Definition 322.14. It lists the finite precision index, dyadic radius, observation window, hsame, Cont, Pkg, NameCert, and exact ledger rows carried by an accepted schedule packet.

Use Definition 322.15 to keep every consumer comparison inside those displayed carrier rows. The classifier therefore supplies no hidden stream tail, quotient real equality, selected limit, trichotomy, completeness witness, or bridge coordinate.

Next invoke Definition 322.16. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ boundary read the same finite precision, radius, window, transport, provenance, name, and ledger packet.

Finally apply Definition 322.13 together with the visible carrier-event rows named in Definition 322.12. The public boundary is therefore precisely the consumer packet named above, with no paper-axis closure promotion asserted here. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicPrecision}^{\uparrow}}$.

Constructive story: A dyadic precision schedule is generated by finite unary precision rows, visible $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledgers, finite $\mathsf{StreamName}^{\uparrow}$ observation windows, componentwise hsame transport, displayed Cont routes, and Pkg and NameCert provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicPrecision}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridge scope packages the finite precision schedule, dyadic radius row, observation window, classifier transport, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ boundary, public export, local NameCert row, exact ledger, and TasteGate visible rows through Definition 322.11 and Definition 322.13.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicPrecision}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DyadicPrecisionUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close $\mathsf{RegSeqRat}^{\uparrow}$ regularity, $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, trichotomy, hidden stream tails, or a standard bridge.

Upgrade path: No further paper-axis upgrade remains; future acceptance work is bridgeChecked formal validation over the same public finite precision schedule, radius, window, transport, provenance, NameCert, ledger, and TasteGate rows.

Chapter 323

A Concrete Naming Certificate for $\mathsf{FastCauchy}^{\uparrow}$

The $\mathsf{FastCauchy}^{\uparrow}$ interface records a finite, modulus-normalized Cauchy-name surface feeding the regular-sequence and real-name layers. In Bishop-style constructive analysis, a Cauchy name is useful only together with visible rate data; the BEDC packet keeps that rate as finite certificate data over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The construction consumes the finite observation discipline of chapter 126, the endpoint ledger of chapter 291, the regular-sequence handoff of chapter 301, and the seal boundary of chapter 38. These sibling rows are the dependency route used by the carrier and handoff theorems below: the fast Cauchy packet is a finite certified readout between stream observations, dyadic endpoints, regular tails, and the real-name seal, not an isolated Cauchy predicate.

323.1 Finite modulus-normalized Cauchy names

Definition 323.1 (FastCauchy carrier)**.** *A* $\mathsf{FastCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (s, \mu, E, R, L, T, P, N)$$

with the following visible rows:

1. *s is a scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation table whose displayed entries are dyadic-rational observations;*
2. *μ is a* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold table normalized so that each requested precision row names the finite index after which all scheduled comparisons are read;*
3. *E is the dyadic endpoint ledger read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *for every scheduled observation and precision row used by μ;*
4. *R is the late-pair radius ledger, recording for each precision row that any two scheduled observations beyond the threshold named by μ fit inside the displayed dyadic radius;*
5. *L records the componentwise* hsame *transport rows for the stream, threshold, dyadic endpoint, and late-pair ledgers;*
6. *T records the* Cont *windows through which those finite reads are consumed by downstream certificates;*

7. *P is the* Pkg *provenance for the named source chapters* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ *row for the displayed packet.*

The carrier is a finite explicit-rate Cauchy name. It contains no quotient stream, no host real equality, no selected limit, and no standard bridge datum.

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyTasteGate single carrier alignment`

Definition 323.2 (FastCauchy classifier)**.** *For accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packets C and C', the classifier* $\mathsf{FastCauchyClassifier}(C, C')$ *holds when the scheduled observation windows, modulus threshold rows, dyadic endpoint ledgers, late-pair radius ledgers,* hsame *transports,* Cont *windows,* Pkg *provenance, and local* NameCert *rows compare componentwise through their displayed source classifiers.*

The classifier is exact only for the finite modulus-normalized ledger. It does not assert equality of completed real numbers, equality of quotient Cauchy streams, or existence of a chosen limit.

Theorem 323.3 (FastCauchy modulus transport)**.** *Let C be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet. If the scheduled observation rows of C transport along the displayed* hsame *rows and the threshold table μ transports along the displayed* Cont *windows, then the transported packet preserves the same modulus-normalized late-pair radius obligations.*

Concretely, every precision row in μ keeps its finite threshold, every scheduled pair beyond that threshold keeps its dyadic endpoint rows from $\mathsf{DyadicRatCore}^{\uparrow}$*, and the late-pair radius ledger remains readable by the same* $\mathsf{CauchyModulus}^{\uparrow}$ *classifier.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFiniteCarrier modulus transport`

Proof. Project the carrier tuple of Definition 323.1. The scheduled stream, threshold table, dyadic endpoint ledger, and late-pair ledger are separate finite rows, so each row can be transported through its displayed hsame coordinate.

The Cont row supplies the finite window in which downstream consumers may read the transported data. Because the threshold row is preserved, every pair of scheduled observations whose indices are beyond that threshold remains in the same late-pair comparison window.

Repack the transported stream, threshold, endpoint, radius, transport, window, provenance, and local certificate rows. The result has the same carrier shape and satisfies the classifier of Definition 323.2, so the modulus-normalized obligations are preserved. □

Theorem 323.4 (FastCauchy handoff to RegSeqRat)**.** *Every accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet presents a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence window by forgetting only the local* $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ *row. The handoff keeps the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation rows, the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold table, the dyadic endpoint rows from* $\mathsf{DyadicRatCore}^{\uparrow}$*, the late-pair radius ledger, the* hsame *and* Cont *transport rows, and the* Pkg *provenance.*

The handoff therefore supplies regularity data to $\mathsf{RegSeqRat}^{\uparrow}$ *without producing a completed* $\mathsf{Real}^{\uparrow}$ *object, quotienting Cauchy names, or selecting a limiting value.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket regseqrat handoff`

Proof. Start with the packet of Definition 323.1. Remove the local row N and retain the stream, modulus, endpoint, radius, transport, window, and provenance coordinates.

For each precision row, the $\mathsf{CauchyModulus}^{\uparrow}$ threshold names the finite index after which comparisons must be checked. The late-pair radius ledger supplies exactly those checks over the scheduled dyadic endpoint rows.

These retained rows are the regular-sequence surface consumed by $\mathsf{RegSeqRat}^{\uparrow}$. Since the construction never adds a quotient, host real equality, or selected limit coordinate, the resulting handoff is only the finite regularity window. □

Theorem 323.5 (FastCauchy Real boundary finite window)**.** *Every accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet exposes to a* $\mathsf{Real}^{\uparrow}$ *consumer only the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold table, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the late-pair radius ledger, the* hsame *and* Cont *transport rows, and the* Pkg *provenance retained by Definition 323.4.*

It does not export a selected limit, quotient stream, host real equality, or completeness datum.

Proof. Project the packet of Definition 323.1. Its visible coordinates are the stream observations, modulus table, endpoint rows, late-pair radius ledger, transport windows, provenance row, and local certificate row.

Invoke Definition 323.4. After forgetting the local $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ row, the retained coordinates are exactly the finite regularity surface that a $\mathsf{Real}^{\uparrow}$ consumer may inspect through the regular-sequence boundary.

There is no further $\mathsf{Real}^{\uparrow}$ coordinate in the carrier. In particular, the packet contains no selected limit, quotient stream, host real equality witness, or completeness datum beyond the listed finite rows. □

Theorem 323.6 (FastCauchy NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ *obligation surface has the following finite rows:*

1. *carrier: the* BHist *packet of Definition 323.1;*
2. *classifier: the componentwise comparison of Definition 323.2;*
3. *stability: the* hsame *and* Cont *transport rows preserving scheduled observations, modulus thresholds, dyadic endpoints, and late-pair radius estimates;*
4. *regular-sequence handoff: the* $\mathsf{RegSeqRat}^{\uparrow}$ *window of Definition 323.4;*
5. *real-name boundary: the* $\mathsf{Real}^{\uparrow}$ *consumer may read the finite regularity window and provenance only.*

These rows are exactly the seed naming-certificate surface for $\mathsf{FastCauchy}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyPacket semantic name certificate`

Proof. The carrier and classifier rows are the displayed objects of Definition 323.1 and Definition 323.2. They enumerate all finite coordinates visible to the local certificate.

The stability row is Definition 323.3. It shows that transporting the stream and threshold coordinates preserves the late-pair radius obligations read by the classifier.

The regular-sequence handoff is Definition 323.4. The real-name boundary is the same finite handoff viewed as input data for $\mathsf{Real}^{\uparrow}$, with no additional completion or equality principle. □

Theorem 323.7 (FastCauchy obligation closure packet)**.** *The* $\mathsf{FastCauchy}^{\uparrow}$ *carrier, componentwise classifier, modulus transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *finite-window boundary, provenance row, and ledger exclusions form the complete obligation surface for* $\mathsf{FastCauchy}^{\uparrow}$*.*

Every downstream read factors through the scheduled $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows named in the packet. The packet exports no quotient stream, selected limit, host real equality, completeness datum, or standard bridge.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket obligation closure certificate`

Proof. The carrier row is Definition 323.1. It lists the scheduled stream, modulus threshold, dyadic endpoint, late-pair radius, transport, window, provenance, and local naming rows, and it excludes quotient streams, selected limits, host real equality, and bridge data.

The classifier row is Definition 323.2. It compares exactly the visible rows componentwise through the source classifiers, so every accepted comparison remains inside the finite modulus-normalized ledger.

The transport obligation is Definition 323.3. It preserves the scheduled observations, threshold rows, endpoint ledgers, and late-pair radius obligations through the displayed hsame and Cont coordinates.

The outgoing consumer boundaries are Definition 323.4 and Definition 323.5. They route downstream reads through the named $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ source surfaces while retaining only the finite regularity window and provenance rows.

Finally, Definition 323.6 gathers these rows as the local $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ surface. Since the carrier has no further completion, quotient, host-equality, selected-limit, or standard-bridge coordinate, the listed rows exhaust the obligation packet. □

Theorem 323.8 (FastCauchy scoped dependency certificate)**.** *Every exported* $\mathsf{FastCauchy}^{\uparrow}$ *row factors through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold table,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *finite-window boundary,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The certificate exports no quotient stream, selected limit, host real equality, completeness datum, or standard bridge.*

Proof. Start with the finite carrier and classifier of Definition 323.1 and Definition 323.2. They make the scheduled observations, threshold table, dyadic endpoint ledger, late-pair radius ledger, transport rows, provenance, and local naming row the only visible coordinates.

The scoped transport source is Definition 323.3. It uses the displayed hsame rows and Cont windows to preserve the finite modulus-normalized tail obligations, so the proof reads stability from BEDC transport primitives rather than from a host sequence model.

The downstream source rows are Definition 323.4 and Definition 323.5. They expose the same finite regularity window to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers while retaining the source-certificate provenance named by the packet.

Finally apply Definition 323.6 and Definition 323.7. The carrier, classifier, transport, handoff, boundary, provenance, and local naming rows listed there are exactly the exported rows; no quotient stream, host real equality, selected limit, completeness datum, or standard bridge coordinate is available. □

Theorem 323.9 (FastCauchy public interface export)**.** *The carrier, classifier, checked modulus transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *finite-window boundary, scoped dependency certificate,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows assemble a reusable public* $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ *interface for finite modulus-normalized Cauchy-name packets.*

Downstream consumers may cite this export instead of reopening the carrier. The exported interface contains no quotient stream, host real equality, selected limit, completeness datum, or standard bridge.

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFiniteCarrier public interface export`

Proof. Begin with the carrier of Definition 323.1 and the classifier of Definition 323.2. These give the scheduled stream rows, modulus threshold rows, dyadic endpoint rows, late-pair radius ledger,

transport rows, provenance package, local naming row, and finite ledger that every consumer can inspect.

The checked motion is the modulus transport of Definition 323.3. It keeps those rows inside the displayed hsame and Cont routes, so the public interface is stable without opening a host Cauchy sequence.

The consumer handoffs are Definition 323.4 and Definition 323.5. Together with Definition 323.6 and Definition 323.8, they repackage the same finite rows as a reusable $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ surface.

No cited component supplies a quotient stream, host real equality, selected limit, completeness datum, or standard bridge. The assembled interface therefore exports only the finite modulus-normalized packet rows named in the statement. □

Theorem 323.10 (FastCauchy public RegSeqRat Real seal)**.** *The public* $\mathsf{FastCauchy}^{\uparrow}$ *finite-modulus boundary of Definition 323.9 determines exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows of Definition 323.4 and the finite* $\mathsf{Real}^{\uparrow}$ *boundary rows of Definition 323.5. These are the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius ledger,* hsame *and* Cont *transport rows,* Pkg *provenance, and local* NameCert *rows.*

The sealed public boundary exposes no selected limit, completeness coordinate, quotient stream, or host real equality.

Proof. Start from the $\mathsf{RegSeqRat}^{\uparrow}$ handoff. It forgets only the local $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ row when forming the regular-sequence window and retains the scheduled observations, modulus thresholds, dyadic endpoints, late-pair radius ledger, transport rows, and provenance.

Read the same retained rows through the $\mathsf{Real}^{\uparrow}$ finite-window boundary. The real-name consumer receives the regular-sequence surface as finite input data, with no coordinate for a selected limit or host real equality.

The scoped dependency certificate and public interface export gather exactly these rows as the public $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$ surface. Therefore every public read is one of the scheduled stream observation, modulus threshold, dyadic endpoint, late-pair radius ledger, transport, provenance, or name rows, and no completeness coordinate or quotient stream is available. □

Theorem 323.11 (FastCauchy precision window restriction)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet and let D be a displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *finite precision row selecting a subwindow of the schedule of F. The packet restricted along D exposes exactly the selected* $\mathsf{StreamName}^{\uparrow}$ *observation rows, the selected* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, the selected* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the late-pair radius ledger over the selected window, the componentwise* hsame *and* Cont *rows, the restricted* Pkg *provenance, and the local* NameCert *rows attached to those selected coordinates.*

Its $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is the restriction of Definition 323.4 to the selected window, and its* $\mathsf{Real}^{\uparrow}$ *boundary is the restriction of Definition 323.5 to the same displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *row. The restriction introduces no selected limit, quotient stream, host real equality, completeness datum, or standard bridge.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket precision window restriction`

Proof. Project the carrier tuple of F from Definition 323.1. The displayed $\mathsf{DyadicPrecision}^{\uparrow}$ row selects a finite subwindow of the scheduled $\mathsf{StreamName}^{\uparrow}$ observations, so the restricted packet keeps precisely the matching threshold row, dyadic endpoint rows, late-pair radius entries, transport rows, provenance rows, and local naming rows.

Use Definition 323.3 on the selected rows. The transport theorem keeps the modulus threshold, dyadic endpoints, hsame comparisons, and Cont routes stable inside the same finite window, so the restriction has the same public coordinates as the parent packet, only with the schedule narrowed by D.

Apply Definition 323.10 to the restricted surface. Its $\mathsf{RegSeqRat}^{\uparrow}$ handoff is therefore the selected part of Definition 323.4, and its $\mathsf{Real}^{\uparrow}$ boundary is the selected part of Definition 323.5.

The seal theorem also rules out any extra coordinate for a selected limit, quotient stream, host real equality, completeness datum, or standard bridge. Thus the precision subwindow is only a finite public restriction of the displayed $\mathsf{FastCauchy}^{\uparrow}$ rows. □

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFiniteCarrier precision window restriction`

Theorem 323.12 (FastCauchy DyadicPrecision window cofinality)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet and let D be an accepted* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedule selecting a finite precision window for F. Every consumer-visible row of the packet restricted by D is exactly one of the selected* $\mathsf{StreamName}^{\uparrow}$ *observation rows, the selected* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, the selected* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the selected late-pair radius ledger entries, the displayed* hsame *and* Cont *transport rows, the restricted* Pkg *provenance rows, and the local* NameCert *rows attached to the selected coordinates.*

Conversely, every selected $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, every selected* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, and every late-pair radius ledger entry whose indices lie in the finite* $\mathsf{DyadicPrecision}^{\uparrow}$ *window appears in the restricted* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and in the finite* $\mathsf{Real}^{\uparrow}$ *boundary. The coverage is a coverage of public finite rows only: it does not assert a selected limit, quotient stream, host real equality, completeness datum, or standard bridge.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket dyadicprecision window cofinality`

Proof. Project F through Definition 323.1. This exposes the scheduled observation rows, modulus thresholds, dyadic endpoints, late-pair ledger entries, transport rows, provenance rows, and local name rows carried by the accepted packet.

Project D through Definition 322.1. The schedule chooses a finite window of those carried coordinates, so Definition 323.11 identifies the restricted packet with exactly the rows whose indices are selected by D.

Apply Definition 323.10. It identifies the public $\mathsf{RegSeqRat}^{\uparrow}$ handoff and finite $\mathsf{Real}^{\uparrow}$ boundary with the same selected threshold, endpoint, late-pair, transport, provenance, and naming rows.

The forward coverage follows because no other consumer-visible row survives the restriction. The converse follows because each selected threshold, endpoint, and late-pair row named by D is one of the rows sealed into both public consumer surfaces. □

Theorem 323.13 (FastCauchy shared-window RegSeqRat handoff)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet, and let D and D' be accepted* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedules for F whose selected finite windows are related by the* $\mathsf{DyadicPrecision}^{\uparrow}$ *classifier. The restricted* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows obtained from $F|D$ and $F|D'$ are the same displayed rows after the componentwise* hsame *and* Cont *transports carried by F, D, and D'.*

The common handoff consists exactly of the scheduled $\mathsf{StreamName}^{\uparrow}$ *observation rows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the late-pair radius rows, the componentwise* hsame *rows, the* Cont *rows, the restricted* Pkg *provenance rows, and the local* NameCert *rows attached to the common transported window. It contains no selected limit, quotient stream, host real equality, completeness datum, hidden tail, or standard bridge datum.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket shared window regseqrat handoff`

Proof. Project F through Definition 323.1. The projection exposes only the scheduled observation rows, modulus thresholds, dyadic endpoints, late-pair radius ledger, transport rows, package provenance, and local name-certificate rows available to a finite consumer.

Project D and D' through Definition 322.1. Use the classifier of Definition 322.2 together with Definition 322.10 to read the common finite window transported by the two accepted schedules.

Apply Definition 323.11 to both restrictions. Since both selected windows read back to the same transported window, the two restricted packets display the same scheduled observations, thresholds, endpoints, late-pair rows, hsame rows, Cont rows, Pkg rows, and NameCert rows.

Finally use Definition 323.12 and Definition 323.10. Cofinality says every selected row of the common window appears in the restricted $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the seal identifies the RegSeqRat and finite Real consumer rows with that same public finite surface. The excluded limit, quotient, equality, completeness, hidden-tail, and bridge data are not coordinates of this surface. □

Theorem 323.14 (FastCauchy modulus synchronization common RegSeqRat window)**.** *Let F and G be accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packets. Suppose that both packets display the same accepted* $\mathsf{DyadicPrecision}^{\uparrow}$ *row D, classifier-compatible* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows over the precision indices selected by D, and a common scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation window. Then the two packets determine one finite* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer package.*

The visible rows of this package are exactly the shared precision row D, the shared $\mathsf{StreamName}^{\uparrow}$ *schedule, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the synchronized* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows, the late-pair radius ledgers, the displayed* hsame *rows, the displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows carried by the two accepted packets. The package has no coordinate for a selected limit, quotient stream, host real equality, completeness datum, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket modulus synchronization common regseqrat window`

Proof. Project F and G through Definition 323.1. The projections expose only the displayed precision, stream-window, modulus, dyadic endpoint, late-pair, transport, provenance, and local naming rows named in the statement.

Apply Definition 323.12 to each packet with the shared row D. Every requested precision index is therefore forced through the same displayed $\mathsf{DyadicPrecision}^{\uparrow}$ window, so no consumer row can escape the shared schedule.

Use Definition 323.13 on the two restricted handoffs. The classifier-compatible threshold rows and the common $\mathsf{StreamName}^{\uparrow}$ window identify the same finite regular-sequence handoff rows, including the synchronized threshold rows and late-pair radius ledgers.

Finally apply Definition 323.10. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the finite $\mathsf{Real}^{\uparrow}$ boundary read the same finite surface, and the seal leaves no coordinate for selected limits, quotient streams, host real equality, completeness data, or standard bridge rows. □

Theorem 323.15 (FastCauchy standard RegSeqRat Real bridge)**.** *A standard bridge for the finite modulus-normalized* $\mathsf{FastCauchy}^{\uparrow}$ *packet factors through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius ledger,* hsame *and* Cont *transport rows,* Pkg *provenance, and local* NameCert *rows already isolated by Definition 323.10. Its* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer reads are exactly the public rows supplied by Definition 323.9, Definition 323.4, and Definition 323.5.*

The bridge exports no selected limit, completeness coordinate, quotient stream, or host real equality.

Proof. Begin with the public interface export. It lists the scheduled stream observations, modulus thresholds, dyadic endpoints, late-pair radius ledger, transport rows, package provenance, and local name rows available to any consumer.

Route the regular-rational side through Definition 323.4 and the real-name side through Definition 323.5. Both consumers read the same finite modulus-normalized rows and add no separate limit or equality coordinate.

Finally apply Definition 323.10. The seal identifies the two consumer surfaces as the public finite rows, so the standard bridge is exactly this RegSeqRat/Real factorization and nothing outside it. □

Theorem 323.16 (FastCauchy mature finite-window consumer catalogue)**.** *For an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet and any displayed finite precision window, the mature finite-window consumers read exactly the rows already exposed by the carrier and public interface: scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints, late-pair ledgers,* hsame *and* Cont *transport rows,* Pkg *provenance, and local* NameCert *rows. The* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *sides of the standard bridge read the same catalogue after the precision-window restriction.*

No mature finite-window consumer receives a coordinate for a selected limit, quotient stream, host real equality, completeness datum, hidden tail, or bridge object beyond these finite public rows.

Proof. Start with Definition 323.9. It lists the consumer-visible carrier rows: scheduled observations, threshold rows, endpoint rows, late-pair ledger entries, transport rows, package provenance, and local name-certificate rows.

Restrict that surface by the finite precision window. By Definition 323.11, the restriction keeps exactly the selected rows of the same kinds and introduces no new analytic coordinate.

Next apply Definition 323.10. The seal identifies the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the finite $\mathsf{Real}^{\uparrow}$ boundary with the restricted public rows.

Finally use Definition 323.15. The bridge routes both standard consumers through that sealed finite surface, so every mature finite-window read is one of the enumerated rows. The omitted limit, quotient, equality, completeness, tail, and independent bridge data are not coordinates of the carrier, the restricted public surface, or either standard consumer boundary. □

Theorem 323.17 (FastCauchy explicit-rate seal-window exhaustion)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet carrying its displayed explicit-rate* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, and let D be a displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *subwindow of that schedule. Every public* $\mathsf{Real}^{\uparrow}$ *seal read over the restricted packet factors through the displayed explicit-rate packet and the subwindow D: it is exactly one of the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius rows,* hsame *rows,* Cont *rows,* Pkg *provenance rows, or* NameCert *rows already isolated by Definition 323.11, Definition 323.10, and Definition 323.15.*

Conversely, each row of those displayed kinds whose index lies in the $\mathsf{DyadicPrecision}^{\uparrow}$ *subwindow is a public* $\mathsf{Real}^{\uparrow}$ *seal read of the restricted packet through the same explicit-rate factorization. The factorized seal has no coordinate for a selected limit, quotient stream, host real equality, completeness datum, hidden tail, or standard bridge datum beyond the finite rows named above.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket explicit rate seal window exhaustion`

Proof. Project the carrier of F from Definition 323.1. The projected packet exposes the scheduled stream observations, explicit modulus thresholds, dyadic endpoints, late-pair radius ledger,

transport rows, package rows, and name-certificate rows; these are the only displayed coordinates available before any consumer reads the packet.

Restrict this carrier along the displayed $\mathsf{DyadicPrecision}^{\uparrow}$ row D. By Definition 323.11, the restriction keeps exactly the rows whose indices lie in the finite precision subwindow, including the explicit-rate thresholds and the dyadic endpoint rows used by the late-pair radius ledger.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the finite $\mathsf{Real}^{\uparrow}$ boundary. The handoff identifies the regular-rational consumer rows with the same selected schedule, and Definition 323.10 identifies the $\mathsf{Real}^{\uparrow}$ seal reads with that finite public surface.

Finally use Definition 323.15. The bridge does not create an additional public coordinate; it only routes the displayed RegSeqRat and Real consumers through the same explicit-rate window. Therefore every public seal read is exhausted by the listed rows, and the omitted limit, quotient, equality, completeness, tail, and bridge data remain unavailable. □

Theorem 323.18 (FastCauchy uniform StreamName window extraction)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet and let D be a displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *row of its explicit-rate schedule. There is a finite schedule window, uniform for the displayed precision row D, whose visible surface consists exactly of the selected* $\mathsf{StreamName}^{\uparrow}$ *observation rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius ledger,* hsame *rows,* Cont *rows,* Pkg *provenance, and local* NameCert *rows.*

The induced $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer reads are precisely the restrictions of the public seal to that finite schedule window. The extraction has no coordinate for a selected limit, quotient stream, host real equality, completeness datum, hidden tail, or standard bridge object outside the displayed public rows.*

Lean4 ✓ `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket uniform streamname window extraction`

Proof. Project the carrier of F through Definition 323.1. This exposes only scheduled observations, explicit modulus thresholds, dyadic endpoints, late-pair radius entries, componentwise transport, package provenance, and local naming rows.

Apply Definition 323.11 to restrict the carrier along D. The restricted carrier keeps exactly the rows whose indices belong to the finite precision window.

Use Definition 323.12 to choose the same schedule window for every consumer read at the displayed precision. This makes the selected $\mathsf{StreamName}^{\uparrow}$ rows and the explicit modulus threshold uniform over the whole extracted surface.

Next invoke Definition 323.13. The $\mathsf{RegSeqRat}^{\uparrow}$ read is the handoff of the same selected schedule, endpoint, radius-ledger, transport, provenance, and naming rows.

Finally apply Definition 323.17. The public $\mathsf{Real}^{\uparrow}$ seal is exhausted by that finite explicit-rate window, so the consumer reads are exactly restrictions of the public seal. The excluded limit, quotient, equality, completeness, tail, and bridge objects require coordinates absent from the extracted carrier. □

Theorem 323.19 (FastCauchy common precision window connector)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet, and let D and D' be two displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *rows of its explicit-rate schedule. Suppose the selected threshold, endpoint, late-pair, and window rows for D and D' admit a common transported* $\mathsf{StreamName}^{\uparrow}$ *finite window. Then the two selected* $\mathsf{RegSeqRat}^{\uparrow}$ *handoffs and the* $\mathsf{Real}^{\uparrow}$ *seal reads factor through one finite consumer window.*

The common window contains exactly the scheduled observations, the $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the late-pair radius ledger, the displayed* hsame *and* Cont *rows, the* Pkg *provenance, and the local* NameCert *rows transported into that window. It contains no*

selected limit, quotient stream, host real equality, completeness datum, hidden tail, or independent bridge coordinate.

Proof. Apply Definition 323.18 to the two displayed precision rows. This places both precision selections inside finite $\mathsf{StreamName}^{\uparrow}$ windows whose visible coordinates are scheduled observations, thresholds, endpoints, radius rows, transport rows, provenance rows, and local naming rows.

Use Definition 323.12 on the common transported window. The cofinality row identifies the selected threshold, endpoint, late-pair, and window package for the two displayed $\mathsf{DyadicPrecision}^{\uparrow}$ reads, so the consumer reads use one finite schedule rather than two unrelated schedules.

Next apply Definition 323.13. The two $\mathsf{RegSeqRat}^{\uparrow}$ handoffs are restrictions of the accepted packet to the same scheduled observations, explicit modulus threshold, dyadic endpoints, late-pair radius ledger, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally use Definition 323.17. The $\mathsf{Real}^{\uparrow}$ seal reads are exhausted by the same finite explicit-rate surface, so the handoffs and seal reads all factor through one shared consumer window. □

Theorem 323.20 (FastCauchy StreamName–RegSeqRat–Real window intersection)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet, and let D and D' be two displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *rows of its explicit-rate schedule. Suppose their selected windows admit a common transported* $\mathsf{StreamName}^{\uparrow}$ *finite schedule window W. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows selected over W and the* $\mathsf{Real}^{\uparrow}$ *seal rows selected over W are exactly the same scheduled observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius rows,* hsame *rows,* Cont *rows,* Pkg *provenance rows, and* $\mathsf{NameCert}$ *rows.*

The intersection statement has no coordinate for a selected limit, quotient stream, host real equality, completeness datum, hidden tail, or $\mathsf{bridgeChecked}$ *status. It identifies only the finite public rows visible through the common schedule window W.*

Proof. Apply Definition 323.18 to the two displayed precision rows. The common transported window W carries the visible $\mathsf{StreamName}^{\uparrow}$ observations, explicit thresholds, dyadic endpoints, late-pair radius rows, transport rows, provenance rows, and local naming rows.

Use Definition 323.13 for the $\mathsf{RegSeqRat}^{\uparrow}$ side. Its handoff reads are obtained by restricting the accepted packet to the same scheduled observations and the same finite explicit-rate rows.

Use Definition 323.17 for the $\mathsf{Real}^{\uparrow}$ side. The seal reads are exhausted by the finite explicit-rate surface, so no additional real-completeness or quotient coordinate is introduced.

It remains to compare the two finite readouts. Apply Definition 323.12 to identify the two displayed precision selections inside W, and then apply Definition 323.10 to view the handoff and seal rows as the same public observations. The excluded limit, quotient, equality, completeness, tail, and checked bridge data are outside this finite-window intersection. □

Theorem 323.21 (FastCauchy finite precision window union lock)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet, and let D and D' be accepted* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedules whose selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows admit a displayed finite union window U. The* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer package of F restricted to U is exactly the union of the consumer rows obtained from the D and D' restrictions, with the overlap identified by Definition 323.13.*

The union package contains only the selected $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius ledger entries, displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows already visible on the two restricted windows. It introduces no hidden tail, selected limit, quotient stream, host real equality, completeness datum, or* $\mathsf{bridgeChecked}$ *row.*

Proof. Apply Definition 323.12 to the two precision schedules. Each restriction exposes exactly its selected observations, thresholds, endpoints, radius rows, transport rows, package provenance, and local name-certificate rows.

Use Definition 323.18 to place both selected surfaces inside the displayed union window U. The extraction supplies one finite $\mathsf{StreamName}^{\uparrow}$ schedule for the combined consumer read and keeps the two precision selections as subwindows of that schedule.

On the overlap, apply Definition 323.13. The common $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows are therefore the same transported rows, including the synchronized endpoint, radius, hsame, Cont, Pkg, and NameCert coordinates.

Finally use Definition 323.14. It identifies the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer package over the common modulus-normalized window, so the package over U is precisely the finite union of the two restricted packages with their overlap already identified. □

Theorem 323.22 (FastCauchy Bishop regular window induction)**.** *Let F be an accepted* $\mathsf{FastCauchy}^{\uparrow}$ *packet, and let $D_0, \ldots, D_{k-1}$ be a finite list of accepted* $\mathsf{DyadicPrecision}^{\uparrow}$ *schedules for F. Suppose their selected* $\mathsf{StreamName}^{\uparrow}$ *windows admit displayed finite successive union windows, so that each step unions the next selected precision window with the accumulated displayed window. Then the consumer package of F over the final displayed window is exactly the iterated union of the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows obtained from the individual precision restrictions.*

The package contains precisely the selected $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, late-pair radius ledger entries, displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows contributed by the finite list. It introduces no hidden tail, selected limit, quotient stream, host real equality, ambient completeness datum, or bridge row.*

Proof. Proceed by induction on the finite list of displayed precision schedules. For the empty list, the selected window is empty, so the consumer package has no selected $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row and is the empty finite union.

For a singleton list, apply Definition 323.12 to identify the selected schedule window. Then use Definition 323.13 to read the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the corresponding $\mathsf{Real}^{\uparrow}$ rows from that same finite precision restriction.

For the induction step, assume the accumulated displayed window already realizes the iterated union for the first schedules. Union the accumulated window with the next selected precision window and apply Definition 323.21. This identifies the new consumer package as the finite union of the accumulated package and the next restriction, with their overlap already matched by the shared-window handoff.

Finally apply Definition 323.14 to normalize the displayed union window to the common modulus surface. The normalized package is therefore exactly the iterated union of the individual $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ restrictions, and the excluded tail, limit, quotient, equality, completeness, and bridge coordinates remain absent at every finite step. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}$.

Constructive story: A fast Cauchy name is a finite BHist packet consisting of scheduled $\mathsf{StreamName}^{\uparrow}$ observation rows, explicit $\mathsf{CauchyModulus}^{\uparrow}$ threshold rows, dyadic endpoint rows from $\mathsf{DyadicRatCore}^{\uparrow}$, late-pair radius ledgers, componentwise hsame and Cont transport rows, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 323.16 and the explicit-rate seal-window exhaustion bind the finite modulus-normalized consumer catalogue over scheduled $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{CauchyModulus}^{\uparrow}$ thresholds, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, late-pair radius ledger, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchy}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FastCauchyUp.FastCauchyFinitePacket\_obligation\_closure\_certificate`
Bridge status: none
Origin: human
Not claimed: This chapter does not close RealUp completeness, quotient streams, host real equality, or selected limits.
Upgrade path: No further paper theory-axis step is recorded for this chapter; Lean formalization and checked bridge status remain separate formal-axis work.

Chapter 324

A Concrete Naming Certificate for $\mathsf{LyapunovStability}^{\uparrow}$

The $\mathsf{LyapunovStability}^{\uparrow}$ interface records the finite certificate surface used by standard Lyapunov stability arguments for displayed dynamical systems. Its finite descent ledger consumes the Lyapunov energy-certificate surface of chapter 313 and the displayed transition carrier of chapter 195 before it packages level-set containment for downstream control-stability consumers. The classical reference point is Khalil, *Nonlinear Systems*, together with the LaSalle-style Lyapunov stability tradition; the BEDC chapter names only the finite descent packet and its consumer boundary.

324.1 Carrier definition

Definition 324.1 (LyapunovStability finite BHist carrier)**.** *A* $\mathsf{LyapunovStability}^{\uparrow}$ *carrier is a finite* BHist *packet over displayed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. It consists of a state-carrier row* X*, a displayed state-transition row* Φ *from* $\mathsf{DynSystem}^{\uparrow}$*, a base state row* x_0*, a scalar-valued or matrix-valued energy row* V*, nonnegative comparison rows in* $\mathsf{Real}^{\uparrow}$*, a finite level row* α*, a finite descent ledger along the displayed transition row,* hsame *transports between corresponding state and energy rows,* Cont *routes for every transition and energy readback,* Pkg *provenance naming the dependency certificates, and a* $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$ *row for the packet.*

The classifier compares two accepted packets only by these displayed rows: state rows through $\mathsf{VecSpace}^{\uparrow}$*, transition rows through* $\mathsf{DynSystem}^{\uparrow}$*, energy and nonnegative comparison rows through* $\mathsf{Mat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*, and the finite descent ledger through the named* hsame*,* Cont*, and* Pkg *coordinates. It exports no analytic flow, invariant measure, asymptotic convergence theorem, or ambient normed-space completion.*

324.2 Finite stability surface

Theorem 324.2 (LyapunovStability energy descent obligation)**.** *For an accepted* $\mathsf{LyapunovStability}^{\uparrow}$ *carrier* P *as in Definition 324.1, every descent assertion visible to a consumer is one row of the finite ledger. Each such row reads a displayed transition face* $x \mapsto \Phi(x)$*, the two energy rows* $V(x)$ *and* $V(\Phi(x))$*, a* $\mathsf{Real}^{\uparrow}$ *comparison row saying that the successor energy is no larger than the source energy, and the* hsame*,* Cont*, and* Pkg *provenance that transports those rows inside* P*.*

Proof. Project the accepted carrier to its finite descent ledger. By the carrier definition, each ledger entry is attached to a displayed transition face and to two energy readbacks.

The nonnegative comparison row is read through $\mathsf{Real}^{\uparrow}$, while a matrix-valued energy row first passes through the displayed $\mathsf{Mat}^{\uparrow}$ or scalar readback route named in the packet. The same entry also carries the Cont route explaining how the successor state and the successor energy are obtained.

No other descent source appears in the carrier. Therefore every visible descent assertion is one of the finite rows listed in the theorem. □

Theorem 324.3 (LyapunovStability level-set containment)**.** *Let P be an accepted* $\mathsf{LyapunovStability}^{\uparrow}$ *carrier with finite level row α. If a displayed state row x satisfies the finite comparison row $V(x) \leq \alpha$ and the transition face from x is recorded in the descent ledger of P, then the successor state row $\Phi(x)$ has a displayed comparison row $V(\Phi(x)) \leq \alpha$ assembled from the same* $\mathsf{Real}^{\uparrow}$ *nonnegative comparison rows, energy ledger,* hsame *transports, and* Cont *provenance.*

Lean4 ✓ `BEDC.Derived.LyapunovStabilityUp.LyapunovStabilityCarrier level set containment`
Lean4 ✓ `BEDC.Derived.LyapunovStabilityUp.LyapunovStabilityPacket level set containment`

Proof. Read the finite descent row for the transition face from x to $\Phi(x)$. It supplies the comparison from the successor energy to the source energy.

Compose that comparison with the displayed level comparison for x. The composition is a finite $\mathsf{Real}^{\uparrow}$ comparison route, and the carrier records the required hsame and Cont provenance for both energy readbacks.

Repack the composed comparison as the level row for the successor state. The construction uses only rows already present in the accepted packet, so it does not introduce an analytic invariant set or an external flow object. □

Theorem 324.4 (LyapunovStability finite descent ledger exactness)**.** *For an accepted* $\mathsf{LyapunovStability}^{\uparrow}$ *carrier P as in Definition 324.1, every consumer-visible descent row, level row, or successor-energy comparison row factors through exactly one displayed finite descent-ledger entry. The factorization carries the recorded* $\mathsf{DynSystem}^{\uparrow}$ *transition row, the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Mat}^{\uparrow}$ *energy readback, the required* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the* $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$ *row of P. No analytic flow, invariant set, asymptotic theorem, or ambient completion supplies a further descent row for the packet.*

Proof. Project P to the coordinates listed in Definition 324.1. A consumer-visible descent or successor-energy comparison must first read a displayed transition face, then read the source and successor energy rows, and finally read a finite comparison row.

The descent entry is the same ledger row used by Definition 324.2. Composing that entry with the finite level comparison follows the construction of Definition 324.3, so the resulting level row has the same transition, energy, transport, continuation, package, and naming provenance.

There is no remaining carrier coordinate from which another descent row could be projected. The listed finite ledger entry therefore exhausts each accepted consumer-visible descent, level, or successor-energy comparison row. □

Theorem 324.5 (LyapunovStability NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$ *obligation surface consists of five rows: the finite carrier of Definition 324.1; the classifier that transports state, transition, energy, level, comparison, descent,* Cont*, and* Pkg *rows componentwise; the energy descent obligation of Definition 324.2; the level-set containment row of Definition 324.3; and the exact consumer boundary of Definition 324.4 over the displayed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with the carrier and classifier. They enumerate the finite packet coordinates and the componentwise transport relation.

Add the descent row, the level-set containment row, and the finite ledger exactness row. These are projections from the same finite ledger and use only the displayed transition, energy, comparison, continuation, and provenance rows.

The consumer boundary is then a closure of the displayed surface under projection. Since the packet has no row for an analytic flow, invariant measure, asymptotic convergence theorem, or ambient completion, those objects cannot be exported by the $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$ surface. □

Theorem 324.6 (LyapunovStability consumer boundary)**.** *Every downstream consumer of* $\mathsf{LyapunovStability}^{\uparrow}$*, including* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{Riccati}^{\uparrow}$*, and control-stability packets, may read only the displayed state-transition row, state-carrier row, scalar or matrix energy row,* $\mathsf{Real}^{\uparrow}$ *nonnegative comparison rows, finite descent ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *row of Definition 324.1. No consumer can require completeness, an asymptotic convergence theorem, invariant-measure data, or an external analytic flow bridge from this chapter.*

Proof. Take a consumer-visible readback from an accepted packet. The classifier lets the consumer compare only the displayed carrier coordinates and the finite ledger rows.

Apply the energy descent and level-set containment theorems to identify the stability-facing rows. These rows are exactly the transition, state, energy, comparison, continuation, transport, and package coordinates named in the carrier.

Because none of the excluded objects is a coordinate of the packet, no downstream consumer can project such an object from the certificate. The boundary is therefore the finite BEDC Lyapunov-stability packet itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$.

Constructive story: A Lyapunov-stability packet is seeded by a finite BHist row package containing a displayed $\mathsf{DynSystem}^{\uparrow}$ transition, a $\mathsf{VecSpace}^{\uparrow}$ state carrier, a scalar or $\mathsf{Mat}^{\uparrow}$ energy row, $\mathsf{Real}^{\uparrow}$ nonnegative comparisons, a finite descent ledger, hsame transports, Cont routes, Pkg provenance, and the $\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$ row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$, publicClosed)

Scope closed: Definition 324.5 and Definition 324.6 bind the finite descent ledger, level-set containment, consumer boundary, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LyapunovStability}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LyapunovStabilityUp.LyapunovStabilityPacket\_level\_set\_containment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close completeness, asymptotic convergence, invariant-measure claims, or an external analytic flow bridge.

Upgrade path: The remaining paper-side step is a bridge boundary for the same level-set containment surface.

Chapter 325

A Concrete Naming Certificate for $\mathsf{BousfieldLocalization}^{\uparrow}$

The $\mathsf{BousfieldLocalization}^{\uparrow}$ interface names the finite certificate surface for a displayed left Bousfield-localization packet over a model-category row. In Hirschhorn 2003 and in Goerss–Jardine 1999, local objects and inverted maps are controlled by a model-category background, a selected class of morphisms, and fibrant replacement tests; the BEDC packet below records only the displayed rows that a downstream consumer may inspect.

The dependency boundary is $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, and $\mathsf{Functor}^{\uparrow}$: chapter 158 supplies weak-equivalence, cofibration, fibration, and fibrant-object faces; chapter 61 types the object and arrow rows; and chapter 62 supplies classifier transport and readback routes without exporting a homotopy-category quotient.

325.1 Finite localizing packet

Definition 325.1 (BousfieldLocalization carrier)**.** *A* $\mathsf{BousfieldLocalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, L, T, P, N)$$

whose visible rows have the following shape:

1. *M is a displayed* $\mathsf{ModelCat}^{\uparrow}$ *row, typed through the displayed* $\mathsf{Cat}^{\uparrow}$ *object and morphism rows;*
2. *S is a finite selected morphism-class row whose arrows are the maps to be read as inverted by the localizing certificate;*
3. *L is a finite list of fibrant-local-object rows, each recording a displayed fibrant object together with the mapping-test rows by which S reads locally;*
4. *T is the classifier-transport ledger, consisting of the componentwise* hsame *witnesses and* Cont *routes that carry the model-category, selected-map, and local-object rows;*
5. *P is the* Pkg *provenance row naming the* $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, *and* $\mathsf{Functor}^{\uparrow}$ *dependency certificates used by the packet;*
6. *N is the* $\mathsf{NameCert}$ *ledger for the carrier, classifier, stability, exactness, and consumer-readback obligations.*

The carrier predicate is $\mathsf{BousfieldLocalizationPacket}(M, S, L, T, P, N)$*. Two accepted packets classify together only when the model-category row, selected morphism-class row, fibrant-local-object rows,*

classifier-transport ledger, package provenance, and NameCert *ledger transport by the displayed* hsame *witnesses along the same* Cont *routes named in T.*

Theorem 325.2 (BousfieldLocalization NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BousfieldLocalization}^{\uparrow}}$ *seed surface consists of the following five finite obligations:*

1. *carrier row: the* BHist *packet of Definition 325.1, including the model-category row, selected morphism-class row, fibrant-local-object rows, classifier-transport ledger,* Pkg *provenance, and* NameCert *ledger;*
2. *classifier row: componentwise* hsame *on exactly those displayed coordinates, with no comparison of hidden model structures, external homotopy categories, or unrecorded mapping-space objects;*
3. *stability row: whenever the displayed coordinates are transported along the* hsame *witnesses and* Cont *routes named by T, the transported packet is again an accepted* $\mathsf{BousfieldLocalization}^{\uparrow}$ *carrier;*
4. *exactness row: every local-object test, inverted-map read, and fibrant-row readback obtained by a consumer must be one of the rows recorded in M, S, L, T, P, and N or a finite readback through their named* Pkg *provenance;*
5. *ledger row: the* NameCert *ledger records the carrier, classifier, stability, exactness, and dependency-boundary checks without storing a global localization functor, a quotient homotopy category, or a choice of representatives.*

These obligations define only the BEDC-kernel naming surface for a displayed Bousfield-localization packet; they do not assert the existence theorem for a localization model structure.

Lean4 ✓ `BEDC.Derived.BousfieldLocalizationUp.BousfieldLocalizationFinitePacket semantic name certificate`
Lean4 ✓ `BEDC.Derived.BousfieldLocalizationUp.BousfieldLocalizationFiniteLocalizingPacket namecert obligations`

Proof. Read the carrier from Definition 325.1. Its six coordinates list exactly the finite rows a Bousfield-localization consumer can inspect: a model-category row, a selected class of maps, fibrant-local-object rows, classifier transport, package provenance, and the local certificate ledger.

The classifier and stability obligations are componentwise. Transporting an accepted packet along the displayed hsame witnesses replaces representatives only inside the visible rows, and the Cont routes in T keep the reads inside the named $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, and $\mathsf{Functor}^{\uparrow}$ boundaries.

Exactness is then a projection statement. A consumer obtains a local-object test, an inverted-map read, or a fibrant-row readback only from the recorded rows or from the finite provenance named by P and N.

No coordinate stores a localization functor, a host quotient category, a derived mapping object, or a representative choice. Those classical outputs therefore remain outside the seed surface. □

Theorem 325.3 (BousfieldLocalization local-object handoff)**.** *Every downstream homotopy consumer of an accepted* $\mathsf{BousfieldLocalization}^{\uparrow}$ *packet may read only the certified fibrant-local-object rows in L, the selected maps in S as inverted-map rows through the displayed model-category ledger M, and the finite* hsame, Cont, Pkg, *and* NameCert *provenance recorded in T, P, and N.*

The handoff exports no unlicensed quotient localization, no homotopy-category object outside the displayed rows, no global functorial localization, and no choice of representatives for local equivalence classes.

Lean4 ✓ BEDC.Derived.BousfieldLocalizationUp.BousfieldLocalizationPacket local object handoff

Proof. Project the accepted packet $B = (M, S, L, T, P, N)$. The local-object data are exactly the finite rows in L, and the inverted-map data are exactly the selected arrows in S interpreted through the displayed model-category ledger M.

Transport and readback use only the visible continuation ledger and provenance rows. The hsame witnesses and Cont routes in T carry the selected-map and local-object rows componentwise, while P and N name the dependency certificates and the finite readback surface.

It remains to check that nothing else is exported. Since the tuple contains no quotient-localization coordinate, no homotopy-category object coordinate, no global localization functor, and no representative-choice row, the consumer boundary is exhausted by the certified local-object and inverted-map rows. □

Theorem 325.4 (BousfieldLocalization empty selected-map boundary)**.** *Let $B = (M, S, L, T, P, N)$ be an accepted* BousfieldLocalization$^{\uparrow}$ *carrier in the sense of Definition 325.1. If the selected morphism-class row S is empty, then no downstream consumer has an inverted-map row to read from B. The remaining certified handoff surface is confined to the displayed model-category row M, the fibrant-local-object rows L, the classifier-transport ledger T, the* Pkg *provenance row P, and the* NameCert *ledger N.*

Proof. By Definition 325.1, the only coordinate of B whose entries are maps read as inverted by the localizing certificate is the selected morphism-class row S. The local-object rows in L record fibrant objects and their mapping-test rows, while M, T, P, and N record the model-category ledger, transport routes, package provenance, and certificate ledger. None of those five coordinates is a second selected morphism-class row.

Now assume that S is empty. If a downstream consumer had an inverted-map row to read, then Definition 325.3 would force that read to be one of the selected maps in S, interpreted through the displayed model-category ledger M. Since S has no displayed arrow entry, there is no such selected map. Hence every inverted-map consumer read is absent.

It remains only to identify what the consumer may still inspect. The exactness row in Definition 325.2 allows consumer reads only from the recorded rows of the packet or from their finite named provenance. After the empty row S contributes no inverted-map read, the visible nonempty handoff coordinates are exactly the possible reads from M, L, T, P, and N. This also preserves the exclusions in Definition 325.3: no quotient localization, homotopy-category object outside the displayed rows, global functorial localization, or representative-choice row is exported. □

Lean4 ✓ BEDC.Derived.BousfieldLocalizationUp.BousfieldLocalizationEmptySelectedMap boundary

Theorem 325.5 (BousfieldLocalization two-stage local-object readback)**.** *Let $B = (M, S, L, T, P, N)$ be an accepted* BousfieldLocalization$^{\uparrow}$ *carrier. Every downstream local-object test read obtained after choosing a selected map row in S and a fibrant-local-object row in L factors first through the displayed model-category row M and then through the transport and provenance rows T, P, and N.*

The readback exports no homotopy-category quotient, localization functor, derived mapping object, local-equivalence representative choice, or row outside the finite packet B.

Proof. Project B through Definition 325.1. The selected map rows live only in S, the fibrant-local-object rows live only in L, and both are typed against the displayed model-category coordinate M.

Apply the exactness row in Definition 325.2. A consumer read after selecting one row from S and one row from L must be a read from M, S, L, T, P, N or from their finite named provenance.

Use Definition 325.3 to order that readback. The selected map is interpreted through M, the fibrant-local-object test is read through L, and the allowed transport and naming data are exactly the hsame, Cont, Pkg, and NameCert rows recorded in T, P, and N.

In the degenerate selected-map case, Definition 325.4 removes the first-stage selected-map read. The remaining local-object reads still pass through M, L, T, P, and N, so no quotient, global localization functor, derived mapping object, or representative-choice row is exported. □

Theorem 325.6 (BousfieldLocalization scoped kernel boundary)**.** *For any accepted* BousfieldLocalization$^{\uparrow}$ *packet in the sense of Definition 325.1, every selected morphism-class row, fibrant-local-object row, classifier transport row, exact readback row, ledger-boundary row, and local-object handoff row factors through the displayed* ModelCat$^{\uparrow}$*,* Cat$^{\uparrow}$*,* Functor$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows listed in Definition 325.2.*

The scoped boundary exports no localized model structure, no global functorial localization, no homotopy-category quotient, no derived mapping object, and no representative choice.

Proof. Project the accepted packet $B = (M, S, L, T, P, N)$ through Definition 325.1. The selected morphism-class rows are exactly the entries of S, the fibrant-local-object rows are exactly the entries of L, and their model-category typing passes through the displayed row M.

Apply the five clauses of Definition 325.2. The carrier clause identifies the BHist packet; the classifier clause restricts comparison to componentwise hsame; the stability clause keeps transported packets inside the same displayed coordinates; the exactness clause confines consumer reads to M, S, L, T, P, N and their finite provenance; and the ledger clause records the certificate boundary without adding an external localization object.

Use Definition 325.3 and Definition 325.5 for the consumer-facing rows. They route selected maps, fibrant-local-object tests, and two-stage readbacks through M, S, L, T, P, and N, so each displayed handoff row factors through the same ModelCat$^{\uparrow}$, Cat$^{\uparrow}$, Functor$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert boundary.

The excluded objects would require coordinates not present in the carrier or ledger. There is no localized model-structure coordinate, no global localization-functor coordinate, no homotopy-category quotient coordinate, no derived mapping-object coordinate, and no representative-choice coordinate, so the scoped boundary exports none of them. □

Theorem 325.7 (BousfieldLocalization public export)**.** *The public* BousfieldLocalization$^{\uparrow}$ *export is exactly the finite packet rows* (M, S, L, T, P, N) *of Definition 325.1, routed through the scoped kernel boundary of Definition 325.6. A public consumer may read selected-map rows, fibrant-local-object rows, classifier transport by* hsame *and* Cont*,* Pkg *provenance, and the* NameCert *ledger, and no other coordinate is part of the exported object.*

The export does not include localized model structures, global localization functors, homotopy-category quotients, derived mapping objects, or representative choice.

Proof. Start with the carrier of Definition 325.1. The six coordinates name the only displayed rows available to the packet: model-category data, selected maps, fibrant-local-object reads, classifier transport, provenance, and the certificate ledger.

Apply Definition 325.2 to identify the carrier, classifier, stability, exactness, and ledger obligations on those same coordinates. Then use Definition 325.3 and Definition 325.5 to route the selected-map and fibrant-local-object reads through the displayed M, S, L, T, P, N rows.

Finally invoke Definition 325.6. It packages the finite rows as the scoped consumer boundary, so the public export is the same boundary viewed as the citable $\mathsf{BousfieldLocalization}^{\uparrow}$ object. Since the carrier has no coordinate for a localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice, none of those data can be exported. □

Theorem 325.8 (BousfieldLocalization bridge boundary)**.** *For an accepted finite public packet*

$$B = (M, S, L, T, P, N)$$

of Definition 325.1, the public export of Definition 325.7 determines the bridge boundary for $\mathsf{BousfieldLocalization}^{\uparrow}$ *by routing selected-map rows, fibrant-local-object rows, classifier transport, provenance, and* $\mathsf{NameCert}$ *rows through the displayed* $\mathsf{ModelCat}^{\uparrow}$*,* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *surfaces named by the packet.*

The bridge boundary exports no localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice.

Proof. Project the carrier $B = (M, S, L, T, P, N)$. The six coordinates give the model category row, selected-map row, fibrant-local-object row, classifier-transport ledger, package provenance, and certificate ledger used by the public packet.

Invoke Definition 325.2 to identify the carrier, classifier, stability, exactness, and ledger clauses. The selected-map and fibrant-local-object rows are therefore read only as displayed BHist data typed through the $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, and $\mathsf{Functor}^{\uparrow}$ dependency surfaces.

Use Definition 325.6 to pass from the local rows to the scoped kernel boundary, and then use Definition 325.3 together with Definition 325.5 for the consumer readback. These rows route classifier transport through hsame and Cont, and route provenance through Pkg and $\mathsf{NameCert}$.

Finally apply Definition 325.7. The public object is the same finite packet viewed as a consumer-facing boundary. Since the carrier has no coordinate for a localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice, no such datum can appear in the bridge boundary. □

Theorem 325.9 (BousfieldLocalization displayed packet bridge exactness)**.** *For an accepted* $\mathsf{BousfieldLocalization}^{\uparrow}$ *packet*

$$B = (M, S, L, T, P, N),$$

the displayed bridge boundary is exact exactly when every consumer read factors through the selected-map row S*, fibrant-local-object row* L*, model-category row* M*, classifier-transport row* T*, package row* P*, and* $\mathsf{NameCert}$ *row* N*. Equivalently, the bridge exports precisely those six packet rows as the finite localizing boundary and exports no localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice.*

Proof. For exactness to consumer factorization, apply Definition 325.2. It identifies the accepted packet carrier, componentwise classifier, stability rows, exact readback rows, and ledger coordinates over M, S, L, T, P, and N.

Use Definition 325.3 and Definition 325.5 to route the local-object reads through the selected-map row, fibrant-local-object row, and classifier-transport ledger. The readback route therefore factors through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surfaces named by the packet.

For consumer factorization to exactness, apply Definition 325.6 and Definition 325.7. A consumer read that factors through exactly the six displayed rows is the same scoped finite packet viewed as the public $\mathsf{BousfieldLocalization}^{\uparrow}$ boundary.

Finally invoke Definition 325.8. Its bridge boundary is precisely the finite packet over M, S, L, T, P, and N. Since none of those rows contains a localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice, the displayed bridge packet is exact with exactly the advertised exclusions. □

Theorem 325.10 (BousfieldLocalization mature finite localizing-packet consumer catalogue)**.** *For every accepted* $\mathsf{BousfieldLocalization}^{\uparrow}$ *packet*

$$B = (M, S, L, T, P, N),$$

every mature consumer read is one of the following five displayed rows:

1. *selected-map rows from S typed through the model-category row M;*
2. *fibrant-local-object test rows from L;*
3. *two-stage local-object readbacks through the classifier-transport ledger T;*
4. *dependency provenance through the package row P;*
5. *certificate and finite-ledger rows through N.*

The catalogue exports no localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice.

Proof. Project the accepted carrier through Definition 325.1. The coordinates M, S, L, T, P, and N are the only rows a consumer can inspect.

Use Definition 325.3 for the selected maps and fibrant-local-object tests, and use Definition 325.5 for the two-stage readbacks through T. These theorems place all localizing reads in the displayed M, S, L, and T rows.

Next apply Definition 325.6 and Definition 325.7. They identify the remaining dependency and certificate reads with the displayed Pkg row P and $\mathsf{NameCert}$ ledger N.

Finally invoke Definition 325.8. The bridge boundary is the same finite packet viewed as a consumer surface, so every mature read is exhausted by the five listed rows. Since the carrier has no coordinate for a localized model structure, global localization functor, homotopy-category quotient, derived mapping object, or representative choice, none of those objects is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BousfieldLocalization}^{\uparrow}}$**.**

Constructive story: A Bousfield-localization packet is generated by a displayed $\mathsf{ModelCat}^{\uparrow}$ row, a selected morphism-class row, fibrant-local-object rows, classifier transport by hsame and Cont, Pkg provenance, and a NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BousfieldLocalization}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridge boundary binds the finite carrier, componentwise classifier, stability, exact readback, ledger boundary, local-object handoff rows, scoped kernel boundary, public export, finite packet bridge, and the mature consumer boundary of Definition 325.10 over $\mathsf{ModelCat}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BousfieldLocalization}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.BousfieldLocalizationUp.BousfieldLocalizationEmptySelectedMap\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close the existence of a localized model structure, global functorial localization, homotopy-category quotients, derived mapping objects, or representative choices.

Upgrade path: No further paper theory-axis step is claimed; Lean formalization of the displayed bridge and mature catalogue remains separate.

Chapter 326

A Concrete Naming Certificate for $\mathsf{OptimalTransport}^{\uparrow}$

The $\mathsf{OptimalTransport}^{\uparrow}$ interface names the finite-support transport surface used by the classical optimal-transport presentation of Villani 2009. The BEDC chapter below restricts that surface to displayed finite source and target distribution rows, a finite cost ledger, a finite coupling table, and a linear-programming dual certificate; it does not import measure completion, arbitrary-measure minimizers, quotient spaces, or infinite Kantorovich duality.

The finite coupling carrier consumes the mass rows of chapter 186, the metric-backed cost rows of chapter 57, the feasible-region rows of chapter 208 and chapter 209, and the finite dual rows of chapter 235.

326.1 Finite-support transport packets

Definition 326.1 (OptimalTransport finite coupling carrier)**.** *An* $\mathsf{OptimalTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, \mu, \nu, C, \Gamma, M, S, L)$$

with the following visible rows:

1. *X and Y are finite support ledgers whose entries are carried by displayed source certificates;*
2. *μ and ν are* $\mathsf{Distribution}^{\uparrow}$ *rows on those supports, with finite mass entries and displayed summation ledgers;*
3. *C is either a displayed cost table row or a cost row read through* $\mathsf{Met}^{\uparrow}$ *on the paired support entries;*
4. *Γ is a finite nonnegative coupling table indexed by the displayed support pairs;*
5. *M is the pair of marginal-sum ledgers asserting that row sums of Γ read as μ and column sums read as ν;*
6. *S is the cost-summation ledger for the displayed finite sum of $C(x, y)\Gamma(x, y)$ over the support pairs;*
7. *L is the* Pkg *provenance tying the support, distribution, cost, coupling, marginal, convex, polytope, and dual rows to the cited source certificates.*

The classifier compares two accepted packets componentwise by hsame *on support entries, distribution masses, cost entries, coupling entries, marginal ledgers, cost ledgers,* ConvexSet$^{\uparrow}$ *and* Polytope$^{\uparrow}$ *feasible-region rows, and* LPDuality$^{\uparrow}$ *dual rows. No coordinate stores a host probability space, a completed measure, a quotient of couplings, or a selected optimizer over arbitrary measures.*

Lean4 (definition) `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingCarrier`

Theorem 326.2 (OptimalTransport coupling marginal ledger)**.** *For an accepted carrier* $T = (X, Y, \mu, \nu, C, \Gamma, M, S, L)$*, every marginal claim read by an* OptimalTransport$^{\uparrow}$ *consumer is a projection of the finite ledger* M*. The source marginal at a support row* x *is the displayed finite sum of* $\Gamma(x, y)$ *over entries* y *of* Y*, and the target marginal at a support row* y *is the displayed finite sum of* $\Gamma(x, y)$ *over entries* x *of* X*. These two families of displayed sums are the only marginal data exported by the carrier.*

The ledger exports no sigma-additivity statement, no completion of μ *or* ν*, and no selection principle for couplings outside the displayed finite support.*

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket marginal ledger`

Proof. Project the carrier of Definition 326.1. The only coordinates that mention marginals are the finite distribution rows μ, ν, the coupling table Γ, and the marginal ledger M.

For a source support entry x, read the row of Γ indexed by x and the finite summation row in M attached to that row. This gives the displayed equality between the row sum and the corresponding μ entry. The target case is identical after projecting the column summation row attached to y.

Since the proof uses only finite table projections and the displayed summation rows, there is no operation that can create an ambient measure, complete a sigma algebra, or search over couplings that are not already present in the packet. □

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingCarrier marginal ledger`

Theorem 326.3 (OptimalTransport cost summation ledger)**.** *For an accepted* OptimalTransport$^{\uparrow}$ *carrier, the transport cost exposed to a consumer is exactly the finite ledger* S *summing the displayed pair costs against the displayed coupling entries. If the cost row is read through* Met$^{\uparrow}$*, then each summand uses only the metric-distance row named for that support pair; if the cost row is displayed directly, then each summand uses only that table entry.*

The cost surface contains no integral over a completed product measure, no measurable-selection argument, and no comparison against an unlisted transport plan.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingCarrier cost summation ledger`

Proof. Read the cost coordinate C, the coupling table Γ, and the cost ledger S. The carrier has no other coordinate from which a scalar cost can be computed.

If C is metric-backed, each summand is obtained by the displayed Met$^{\uparrow}$ row for its support pair. If C is table-backed, each summand is the visible table entry. In both cases the finite summation ledger S records the multiplication by the corresponding coupling entry and the finite accumulation over support pairs.

Thus the exported cost is a repacking of listed rows. Product measures, measurable functions, and unlisted transport plans are not available to the projection. □

Theorem 326.4 (OptimalTransport LP-duality feasible surface)**.** *The feasible surface of an accepted* $\mathsf{OptimalTransport}^{\uparrow}$ *packet is the finite polytope determined by the nonnegativity rows for* Γ*, the two marginal ledgers of Definition 326.2, and the displayed cost objective of Definition 326.3. A dual-feasibility certificate is an* $\mathsf{LPDuality}^{\uparrow}$ *row whose inequalities are indexed only by the displayed support pairs and whose objective rows read only the displayed* μ *and* ν *masses.*

The theorem names the finite linear-programming surface only; it does not assert infinite-dimensional Kantorovich duality, analytic compactness, or existence of minimizers outside the displayed polytope.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingCarrier lpduality feasible surface`

Proof. Build the primal feasible region from the coordinates of the carrier. Nonnegativity of the table entries gives the half-space rows, and the marginal ledgers give the equality rows. These rows form the finite $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$ surface for the coupling table.

The objective is the finite cost ledger from Definition 326.3. A dual certificate is then read as an $\mathsf{LPDuality}^{\uparrow}$ packet over the same finite index set: one dual potential row for each displayed source or target support entry and one inequality row for each displayed support pair.

Because every primal and dual constraint is indexed by a listed support entry, the surface remains a finite linear program. Nothing in the construction invokes topological compactness, arbitrary-measure duality, or a minimizer-selection principle. □

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket lpduality feasible surface`

Theorem 326.5 (OptimalTransport finite-support dual-certificate exhaustion)**.** *Every* $\mathsf{OptimalTransport}^{\uparrow}$ *dual certificate read by a consumer is indexed only by the displayed source support rows, target support rows, visible mass rows, and cost inequalities of the finite LP surface of Definition 326.4. Conversely, each listed dual row is read through the finite coupling carrier of Definition 326.1.*

The certificate surface is exhausted by those finite support rows. It contains no sigma-additivity principle, arbitrary-measure minimizer, or infinite-dimensional Kantorovich duality datum.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket dual certificate exhaustion`

Proof. Project the finite coupling carrier. Its visible coordinates are the source and target support rows, the two finite distribution rows, the cost row, the coupling table, the marginal ledgers, the objective ledger, the feasible-region rows, the dual-certificate rows, and the Pkg provenance.

Use Definition 326.2 to identify the only consumer-readable mass constraints. Use Definition 326.3 to identify the only cost summation rows over the displayed coupling table.

Apply Definition 326.4. That theorem places each dual potential and each dual inequality over the same finite source and target supports and over the same displayed cost inequalities. Hence every dual certificate read by a consumer factors through the finite carrier rows, and every listed dual row is read back through that carrier.

No coordinate of the carrier stores a sigma-additive measure space, an arbitrary-measure optimizer, or an infinite-dimensional Kantorovich dual principle. The displayed finite LP rows therefore exhaust the dual-certificate surface. □

Theorem 326.6 (OptimalTransport NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{OptimalTransport}^{\uparrow}}$ *seed surface consists of the following finite rows:*

1. *carrier: the* BHist *packet of Definition 326.1;*
2. *classifier: componentwise* hsame *on support, distribution, cost, coupling, marginal, cost-summation, feasible-region, and dual rows;*
3. *marginal ledger: the two finite summation families of Definition 326.2;*
4. *cost ledger: the finite objective row of Definition 326.3;*
5. *feasible and dual surface: the* ConvexSet$^{\uparrow}$, Polytope$^{\uparrow}$, *and* LPDuality$^{\uparrow}$ *rows of Definition 326.4.*

These rows are sufficient to name the finite-support optimal-transport certificate surface and deliberately exclude measure completion, arbitrary-measure optimizer selection, quotient spaces, and Kantorovich duality beyond the finite linear-programming boundary.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket semantic name certificate`
Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket namecert obligation surface`
Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportPacket semantic name certificate`

Proof. The carrier row fixes the finite support entries, source and target distributions, cost rows, coupling table, marginal ledgers, objective ledger, and provenance. These are precisely the coordinates available to an OptimalTransport$^{\uparrow}$ consumer.

The classifier is componentwise because each coordinate already lives in a displayed BEDC row. Transport along hsame repacks support entries, masses, costs, coupling entries, marginal sums, and dual inequalities without changing the finite index set named by the packet.

The two ledger theorems account for the only marginal and objective reads. The feasible-surface theorem accounts for the only convex, polytope, and dual rows. Since no remaining coordinate contains a completed measure, a quotient coupling, an infinite support, or a global optimizer, the five rows exhaust the local seed certificate. □

Theorem 326.7 (OptimalTransport public finite LP certificate)**.** *An accepted finite-support* OptimalTransport$^{\uparrow}$ *packet exports a public* NameCert$_{\text{OptimalTransport}^{\uparrow}}$ *finite linear-programming certificate consisting exactly of the displayed source and target support rows, the* Distribution$^{\uparrow}$ *mass rows, the cost table or* Met$^{\uparrow}$ *cost rows, the finite coupling table, the two marginal ledgers, the finite objective ledger, the* ConvexSet$^{\uparrow}$ *and* Polytope$^{\uparrow}$ *feasible-region rows, the* LPDuality$^{\uparrow}$ *dual rows, and componentwise* hsame, Cont, *and* Pkg *provenance.*

The certificate contains no measure completion, arbitrary-measure minimizer selection, quotient space, or infinite Kantorovich-duality datum.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingCarrier public finite lp semantic certificate`
Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket public finite lp certificate`

Proof. Read the accepted finite-support carrier of Definition 326.1. Its coordinates supply the source and target support rows, distribution masses, cost entries, coupling table, and package provenance.

Apply Definition 326.2 and Definition 326.3. These two ledgers account for the only marginal reads and the only finite objective read over the displayed coupling table.

Next apply Definition 326.4. It packages the same finite support-pair coordinates as a $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$ feasible region together with the $\mathsf{LPDuality}^{\uparrow}$ dual rows.

Finally apply Definition 326.6. The obligation theorem repacks the carrier, ledgers, feasible surface, classifier transport, continuation rows, and package provenance as the public $\mathsf{NameCert}_{\mathsf{OptimalTransport}^{\uparrow}}$ certificate. Since no projected coordinate names a completed measure, arbitrary-measure optimizer, quotient space, or infinite duality principle, those objects are absent from the exported finite linear-programming certificate. □

Theorem 326.8 (OptimalTransport obligation closure)**.** *The rows of Definition 326.1, Definition 326.2, Definition 326.3, Definition 326.4, and Definition 326.6 exhaust the* $\mathsf{NameCert}_{\mathsf{OptimalTransport}^{\uparrow}}$ *obligation set over finite support, distribution, cost, coupling, marginal, feasible-polytope, dual,* hsame*,* Cont*, and* Pkg *rows.*

No measure completion, infinite Kantorovich duality, quotient coupling space, or optimizer selection is part of the obligation surface.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket obligation closure`

Proof. Project the finite carrier first. Its coordinates list the source and target supports, distribution masses, cost row, coupling table, marginal ledger, cost-summation ledger, feasible-region data, dual certificate, and package provenance.

The marginal and cost reads are exactly the ledgers of Definition 326.2 and Definition 326.3. The feasible and dual reads are exactly the finite linear-programming rows of Definition 326.4.

The NameCert theorem repacks those projected rows as the local certificate surface. Since the packet exposes no completed measure, infinite duality principle, quotient coupling space, or optimizer-selection coordinate, there is no remaining row from which a further obligation could be read. □

Theorem 326.9 (OptimalTransport public export surface)**.** *The public* $\mathsf{OptimalTransport}^{\uparrow}$ *export consists exactly of the finite support rows, finite distribution rows, displayed cost row, coupling table, marginal ledgers, cost ledger, finite feasible polytope surface,* $\mathsf{LPDuality}^{\uparrow}$ *dual row,* hsame *classifier transport,* Cont *routes, and* Pkg *provenance named by Definition 326.1, Definition 326.2, Definition 326.3, Definition 326.4, Definition 326.6, Definition 326.7, and Definition 326.8.*

The public export contains no measure completion, arbitrary-measure minimizer selection, quotient coupling space, or infinite-dimensional Kantorovich duality.

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket public export surface`

Proof. Read the finite carrier of Definition 326.1. Its visible coordinates are the source and target supports, the two finite distribution rows, the cost row, the coupling table, the marginal ledgers, the cost ledger, and the package provenance.

Apply Definition 326.2 and Definition 326.3 to identify the only allowed marginal and objective reads. Then use Definition 326.4 to expose the finite polytope surface and the $\mathsf{LPDuality}^{\uparrow}$ dual row over the same displayed support pairs.

Finally apply Definition 326.6 and Definition 326.8. These results repackage the finite carrier, ledgers, classifier transport, Cont routes, and Pkg provenance as the whole certificate surface. Because the packet has no completed measure coordinate, arbitrary-measure optimizer coordinate, quotient coupling coordinate, or infinite-dimensional duality coordinate, those objects are not part of the public export. □

Theorem 326.10 (OptimalTransport finite-support bridge package)**.** *The public finite-support packet of Definition 326.9, its marginal ledger, cost ledger, finite feasible polytope, and* $\mathsf{LPDuality}^{\uparrow}$ *dual row assemble a bridge boundary from the displayed finite-support* $\mathsf{Distribution}^{\uparrow}$ *rows, the cost row or* $\mathsf{Met}^{\uparrow}$ *row, the* $\mathsf{ConvexSet}^{\uparrow}$ *feasible surface, the* $\mathsf{Polytope}^{\uparrow}$ *feasible-polytope rows, and the* $\mathsf{LPDuality}^{\uparrow}$ *dual-feasibility row to the public* $\mathsf{OptimalTransport}^{\uparrow}$ *export.*

This bridge package exports exactly those finite rows together with hsame *classifier transport,* Cont *routes, and* Pkg *provenance. It excludes measure completion, arbitrary-measure minimizer selection, quotient coupling spaces, infinite Kantorovich duality, optimizer-existence theorems, and arbitrary representative choice.*

Lean4 ✓ `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket finite support bridge package`

Proof. Start with the carrier of Definition 326.1. Its coordinates are the finite source and target support rows, the two finite distribution rows, the cost row, the coupling table, the marginal ledgers, the objective ledger, the finite feasible-region data, the dual certificate row, and the provenance row.

The marginal ledger of Definition 326.2 and the cost ledger of Definition 326.3 identify the only marginal and objective reads. The feasible surface of Definition 326.4 identifies the $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, and $\mathsf{LPDuality}^{\uparrow}$ rows over the same finite support.

Apply Definition 326.9. That theorem repackages the displayed carrier, ledgers, LP-duality feasible surface, classifier transport, continuation routes, and package provenance as the public finite-support transport export.

Repack exactly those displayed finite rows as the bridge package. Since the carrier has no coordinate for completed measures, arbitrary-measure minimizer selection, quotient coupling spaces, infinite Kantorovich duality, optimizer existence, or representative selection, those objects cannot cross the bridge. □

Theorem 326.11 (OptimalTransport mature consumer boundary)**.** *For the finite-support transport packet of Definition 326.1, any public consumer whose rows are the displayed marginals, finite cost table, feasible coupling table, LP-duality certificate, classifier transport,* Cont *route,* Pkg *provenance, and finite-support bridge reads exactly the finite-support* $\mathsf{OptimalTransport}^{\uparrow}$ *package assembled in this chapter.*

Such a consumer cannot read arbitrary-measure minimizers, quotient couplings, infinite Kantorovich duality, analytic compactness, or a standard bridge.

Proof. First project the finite-support carrier. It displays the source and target supports, finite $\mathsf{Distribution}^{\uparrow}$ rows, the cost row, the coupling table, the marginal ledgers, the finite objective ledger, the feasible-region rows, the dual certificate row, classifier transport, continuation, and provenance.

Apply Definition 326.2 and Definition 326.3. These two ledgers account for the displayed marginals and finite cost table, with no coordinate for an arbitrary-measure optimizer.

Next use Definition 326.4, Definition 326.6, and Definition 326.7. They bind the feasible coupling table, LP-duality certificate, classifier transport, Cont route, and Pkg provenance to the same finite packet.

Finally compose Definition 326.9 with Definition 326.10. The resulting consumer boundary is exactly the public finite-support $\mathsf{OptimalTransport}^{\uparrow}$ package assembled by the chapter. The chapter's excluded objects are absent from every cited row, so quotient couplings, infinite duality, analytic compactness, and a standard bridge remain outside the read. □

Theorem 326.12 (OptimalTransport mature finite-support coverage)**.** *Every consumer read of an accepted finite-support* $\mathsf{OptimalTransport}^{\uparrow}$ *bridge packet factors through the displayed source and target supports, the* $\mathsf{Distribution}^{\uparrow}$ *mass rows, the cost table or* $\mathsf{Met}^{\uparrow}$ *cost rows, the finite coupling table, the marginal ledgers, the finite objective ledger, the* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{Polytope}^{\uparrow}$ *feasible-region rows, the* $\mathsf{LPDuality}^{\uparrow}$ *dual rows, and the displayed* hsame*,* Cont*, and* Pkg *provenance.*

The coverage surface excludes measure completion, arbitrary-measure minimizer selection, quotient coupling spaces, infinite-dimensional Kantorovich duality, analytic compactness, and representative selection.

Proof. Start with the finite coupling carrier of Definition 326.1. It contains exactly the displayed supports, distribution mass rows, cost row, coupling table, finite feasible-region data, dual row, and provenance row.

Apply Definition 326.2 and Definition 326.3. These identify the only marginal and objective reads as finite ledger rows over the displayed support pairs.

Next use Definition 326.4. The feasible and dual readers are precisely the displayed $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, and $\mathsf{LPDuality}^{\uparrow}$ rows over the same finite table.

Finally apply Definition 326.9 and Definition 326.10. The public export and bridge boundary repackage the carrier, marginal, objective, feasible, dual, classifier-transport, continuation, and provenance rows as the consumer surface.

No remaining coordinate is present for completed measures, arbitrary-measure minimizer selection, quotient coupling spaces, infinite-dimensional Kantorovich duality, analytic compactness, or representative selection. Therefore every accepted finite-support consumer read factors through the displayed mature coverage inventory. □

Theorem 326.13 (OptimalTransport mature finite-support consumer catalogue)**.** *For every accepted finite-support* $\mathsf{OptimalTransport}^{\uparrow}$ *packet, every mature finite-support consumer read is exhausted by the displayed finite source and target support rows,* $\mathsf{Distribution}^{\uparrow}$ *mass rows, displayed cost or* $\mathsf{Met}^{\uparrow}$ *cost rows, coupling table rows, marginal ledgers, finite objective ledger,* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{Polytope}^{\uparrow}$ *feasible rows,* $\mathsf{LPDuality}^{\uparrow}$ *dual rows, and* hsame*,* Cont*, and* Pkg *provenance.*

The catalogue exports no measure completion, arbitrary-measure minimizer selection, quotient coupling space, infinite Kantorovich duality, optimizer-existence theorem, or representative choice.

Proof. Project the finite coupling carrier of Definition 326.1. Its coordinates list the displayed supports, mass rows, cost rows, coupling table, marginal ledgers, finite objective ledger, feasible rows, dual rows, and provenance.

Apply Definition 326.2 and Definition 326.3. These account for all marginal and objective reads as finite ledger projections over the displayed support pairs.

Use Definition 326.4 for the $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, and $\mathsf{LPDuality}^{\uparrow}$ consumer rows. The feasible and dual reads are indexed only by the same finite support and coupling table.

Finally apply Definition 326.9 together with Definition 326.10. They repackage the carrier, ledger, feasible, dual, classifier-transport, continuation, and provenance rows as the public bridge surface. Since no coordinate stores completed measures, arbitrary-measure minimizers, quotient coupling spaces, infinite Kantorovich duality, optimizer-existence data, or representative choices, the catalogue exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{OptimalTransport}^{\uparrow}}$.

Constructive story: An optimal-transport packet is generated from displayed finite source and target support rows, $\mathsf{Distribution}^{\uparrow}$ mass rows, a cost table or $\mathsf{Met}^{\uparrow}$ cost row, a finite coupling table, marginal summation ledgers, a finite objective ledger, $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{Polytope}^{\uparrow}$ feasible-region rows, an $\mathsf{LPDuality}^{\uparrow}$ dual-feasibility row, and Pkg provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OptimalTransport}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridge surface binds the finite-support carrier of Definition 326.1, the marginal ledger of Definition 326.2, the cost ledger of Definition 326.3, the finite dual-feasible surface of Definition 326.4, the NameCert obligation surface of Definition 326.6, the public finite linear-programming certificate of Definition 326.7, the obligation closure of Definition 326.8, the public export surface of Definition 326.9, and the finite-support bridge package of Definition 326.10 through BHist, hsame, Cont, Pkg, Distribution$^{\uparrow}$, Met$^{\uparrow}$, ConvexSet$^{\uparrow}$, Polytope$^{\uparrow}$, and LPDuality$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{OptimalTransport}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.OptimalTransportUp.OptimalTransportFiniteCouplingPacket\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close measure completion, arbitrary-measure minimizer selection, quotient spaces, infinite-dimensional Kantorovich duality, analytic compactness, or a standard bridge.

Upgrade path: The finite-support mature catalogue is present; the remaining progress is Lean formalization and any later standard bridge for the displayed marginal, cost, feasible-polytope, dual-feasibility, classifier-transport, continuation, and provenance readers.

Chapter 327

A Concrete Naming Certificate for $\mathsf{Semiring}^{\uparrow}$

The $\mathsf{Semiring}^{\uparrow}$ interface names the finite algebraic surface that combines an additive commutative-monoid presentation with a multiplicative presentation on the same displayed carrier. In the classical account of Golan, *Semirings and their Applications* (1999), the object supports addition, multiplication, distributivity, and zero annihilation. The BEDC packet below records only the finite carrier rows, classifier rows, law ledgers, continuation routes, package provenance, and local $\mathsf{NameCert}$ row needed to name that surface without importing a host algebra or quotient equality.

The multiplicative side of the $\mathsf{Semiring}^{\uparrow}$ packet consumes the unit and associativity surface already isolated by chapter 40. This chapter adds the additive surface, distributive ledger, and zero-annihilation rows consumed by $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$ routes.

Definition 327.1 (Semiring finite-history source)**.** *A* $\mathsf{Semiring}^{\uparrow}$ *source is a finite* BHist *packet*

$$S = (A, M, Z, D, E, H, C, P, N)$$

with the following visible rows:

1. *A is an additive commutative-monoid surface over* $\mathsf{Add}^{\uparrow}$*, with the displayed carrier, zero row, addition row, associativity row, unit rows, and commutativity row named on the same finite carrier;*
2. *M is a multiplicative monoid surface over* $\mathsf{Mul}^{\uparrow}$ *when a unit row is present, and otherwise a multiplicative semigroup surface over* $\mathsf{Semigroup}^{\uparrow}$*;*
3. *Z is the shared-carrier and shared-classifier ledger saying that the additive and multiplicative rows classify exactly the same displayed carrier entries;*
4. *D records the left and right distributivity rows for multiplication over addition;*
5. *E records the left and right zero-annihilation rows;*
6. *H records the componentwise* hsame *transports for the carrier, additive surface, multiplicative surface, shared classifier, distributivity ledger, annihilation ledger, continuation routes, provenance, and local name;*
7. *C records the* Cont *routes that read addition, multiplication, distributivity, and annihilation comparisons from the same displayed carrier;*

8. *P is the* Pkg *provenance naming* $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, *and* $\mathsf{Magma}^{\uparrow}$ *as the imported certificate boundaries;*

9. *N is the local* NameCert *row for the resulting finite semiring surface.*

The carrier predicate is $\mathsf{SemiringLedger}(A, M, Z, D, E, H, C, P, N)$. *Two accepted packets classify together only when all nine rows transport by the displayed* hsame *data and follow the same* Cont *routes named in P.*

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringTasteGate single carrier alignment`

Theorem 327.2 (Semiring additive and multiplicative obligation surface)**.** *Let $S = (A, M, Z, D, E, H, C, P, N)$ be an accepted* $\mathsf{Semiring}^{\uparrow}$ *source. The algebraic operation surface exported by S is exactly the additive commutative-monoid row A, the multiplicative monoid or semigroup row M, and the shared-carrier ledger Z. In particular, every downstream operation read is routed through the same displayed carrier and classifier, and no host ring, subtraction row, order row, completion datum, or quotient carrier is exported by the packet.*

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringLedger additive multiplicative obligation surface`

Proof. Project the carrier of Definition 327.1. The additive row A supplies the zero, addition, associativity, unit, and commutativity data, while the multiplicative row M supplies the multiplication and associativity data, with a unit row only when the monoid presentation is present.

The shared-carrier ledger Z is the only bridge between those two operation surfaces. It identifies the displayed carrier and classifier rows used by A and M, so every operation read factors through the same finite BHist source.

The remaining rows are law ledgers, transport routes, provenance, and local naming data. None of them contains subtraction, additive inverses, an order primitive, a completion witness, or a quotient carrier, so the stated surface is exhausted by A, M, and Z. □

Theorem 327.3 (Semiring distributive ledger boundary)**.** *For an accepted* $\mathsf{Semiring}^{\uparrow}$ *source $S = (A, M, Z, D, E, H, C, P, N)$, the distributivity information is exactly the finite ledger D. Its left and right rows compare the two displayed histories for $x \cdot (y + z)$ and $x \cdot y + x \cdot z$, and for $(x + y) \cdot z$ and $x \cdot z + y \cdot z$, using only the addition and multiplication routes named by C on the carrier shared by Z.*

Proof. Read D as a ledger over the carrier fixed by Z. Each row names a finite comparison between histories already produced by the displayed addition and multiplication operations.

For the left distributive row, follow the continuation routes in C from x, y, and z to the two histories $x \cdot (y + z)$ and $x \cdot y + x \cdot z$. The ledger records their displayed comparison by hsame.

For the right distributive row, use the same carrier and classifier routes to compare $(x + y) \cdot z$ with $x \cdot z + y \cdot z$. No additional operation table is introduced, so D is exactly the distributive boundary. □

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringDistributiveLedger boundary`

Theorem 327.4 (Semiring zero-annihilation scope)**.** *For an accepted* $\mathsf{Semiring}^{\uparrow}$ *source $S = (A, M, Z, D, E, H, C, P,$ the zero-annihilation information is exactly the ledger E. It contains the left and right rows identifying $0 \cdot x$ and $x \cdot 0$ with the additive zero row from A, routed through the multiplicative surface M and the shared carrier ledger Z.*

Proof. The additive zero used in the statement is the zero row of A. By Z, this row belongs to the same displayed carrier on which the multiplication row of M is typed.

The first row of E reads the multiplication route for $0 \cdot x$ and compares the result with the additive zero. The second row reads the route for $x \cdot 0$ and performs the same comparison.

No cancellation law, additive inverse, or order-theoretic positivity row is available in the packet. Therefore the annihilation scope is exactly the two finite rows named in E. □

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringZeroAnnihilationLedger scope`

Theorem 327.5 (Semiring NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Semiring}^\uparrow}$ *obligation surface consists of five finite rows:*

1. *carrier and classifier: the* BHist *packet* $S = (A, M, Z, D, E, H, C, P, N)$ *of Definition 327.1;*
2. *operation surface: the additive, multiplicative, and shared-carrier rows of Definition 327.2;*
3. *law ledgers: the distributivity rows of Definition 327.3 and the zero-annihilation rows of Definition 327.4;*
4. *stability and routing: the componentwise* hsame *transports in* H *and the* Cont *routes in* C*;*
5. *provenance: the* Pkg *row naming* $\mathsf{Add}^\uparrow$, $\mathsf{Mul}^\uparrow$, $\mathsf{Monoid}^\uparrow$, $\mathsf{Semigroup}^\uparrow$, *and* $\mathsf{Magma}^\uparrow$.

These rows form the public seed surface consumed by $\mathsf{Ring}^\uparrow$, $\mathsf{Module}^\uparrow$, $\mathsf{Poly}^\uparrow$, *and* $\mathsf{Mat}^\uparrow$ *chapters that need a named semiring obligation source.*

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringFiniteSource semantic name certificate`

Proof. The carrier and classifier row fixes every coordinate that can be named by the certificate. The operation surface theorem identifies the only addition, multiplication, and shared-carrier reads exported by the source.

The distributivity and zero-annihilation boundary theorems identify the only law ledgers available from the packet. The stability and routing rows say how those ledgers transport under displayed hsame and which finite continuations are used to read them.

Finally, the package row lists the imported BEDC certificate boundaries. Since the packet has no coordinate for subtraction, additive inverses, host algebra, quotient equality, or analytic completion, the five rows listed above exhaust the local $\mathsf{NameCert}_{\mathsf{Semiring}^\uparrow}$ obligation surface and give downstream chapters only the named finite semiring source. □

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringLedger namecert obligation surface`

Theorem 327.6 (Semiring concrete history instance)**.** *For any accepted packet* $S = (A, M, Z, D, E, H, C, P, N)$ *of Definition 327.1, the additive row, multiplicative row, shared-carrier ledger, distributivity ledger, zero-annihilation ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Semiring}^\uparrow}$ *row constitute a concrete* BHist *semiring instance consumed by* $\mathsf{Ring}^\uparrow$, $\mathsf{Module}^\uparrow$, $\mathsf{Poly}^\uparrow$, *and* $\mathsf{Mat}^\uparrow$.

This instance exports no subtraction row, additive-inverse row, order primitive, quotient carrier, or host-algebra operation. Every downstream read passes through the finite displayed rows of the packet.

Proof. Project the coordinates $A, M, Z, D, E, H, C, P, N$ from the accepted Definition 327.1 packet. The additive, multiplicative, and shared-carrier rows are exactly the operation surface of Definition 327.2.

The distributivity rows are supplied by Definition 327.3, and the two zero-annihilation rows are supplied by Definition 327.4. These two ledgers give the only displayed law rows available to the semiring packet.

The hsame transports in H, the Cont routes in C, the Pkg provenance in P, and the local name row N are the remaining coordinates of Definition 327.5. Repacking these displayed coordinates yields the concrete BHist semiring instance read by $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$.

Since no coordinate of S names subtraction, additive inverse, order, quotient-carrier equality, or host algebra, no downstream consumer can read such data from the instance. □

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringFiniteSource concrete history instance`

Theorem 327.7 (Semiring public finite-source export)**.** *The finite carrier, classifier, additive surface, multiplicative surface, distributive ledger, zero-annihilation ledger, transport routes, provenance, and local* $\mathsf{NameCert}_{\mathsf{Semiring}^{\uparrow}}$ *row of Definition 327.5 export exactly the public* $\mathsf{Semiring}^{\uparrow}$ *source surface available to* $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *and* $\mathsf{Mat}^{\uparrow}$ *consumers.*

This public surface contains no additive inverses, quotient carriers, ordered semiring data, standard bridge, host algebra, or analytic completion datum. Every downstream read factors through the displayed BHist *rows of Definition 327.1.*

Proof. Project the source packet of Definition 327.1. Its nine rows give the finite carrier and classifier, the additive and multiplicative operation surfaces, the distributive and zero-annihilation ledgers, the componentwise hsame transports, the Cont routes, the Pkg provenance, and the local name row.

The operation surface is identified by Definition 327.2. The law boundary is identified by Definition 327.3 and Definition 327.4, so the available algebraic rows are exhausted by addition, multiplication, distributivity, zero annihilation, and the shared-carrier classifier.

Now apply Definition 327.5 and Definition 327.6. These statements repack exactly those displayed rows as the finite source consumed by $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$.

No cited row contains additive inverses, quotient carriers, order data, a standard bridge, host algebra, or analytic completion. Hence the exported public surface is precisely the named finite semiring source and nothing beyond it. □

Theorem 327.8 (Semiring bridge finite-source downstream consumption)**.** *Let* $S = (A, M, Z, D, E, H, C, P, N)$ *be an accepted packet of Definition 327.1. The public finite-source export of Definition 327.7, together with the concrete history instance of Definition 327.6, assembles exactly the bridge boundary consumed by* $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *and* $\mathsf{Mat}^{\uparrow}$.

That boundary consists only of the displayed additive row, multiplicative row, shared carrier and classifier ledger, distributivity ledger, zero-annihilation ledger, hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no additive inverses, quotient carrier, ordered semiring, standard model, or host algebra.*

Proof. Project the rows of the accepted packet S from Definition 327.1. The additive and multiplicative faces are the two operation rows; the shared carrier and classifier are carried by the same packet spine; the distributivity and zero-annihilation ledgers are the displayed law rows; and H, C, P, and N provide the transport, route, provenance, and naming coordinates.

Apply Definition 327.2, Definition 327.3, and Definition 327.4. These identify the operation and law rows available to any downstream algebra consumer.

Next apply Definition 327.5, Definition 327.6, and Definition 327.7. The first theorem exposes the local certificate obligations, the second repacks the displayed rows as the concrete BHist instance, and the third records the public finite-source export read by $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$.

By row exhaustion, the bridge boundary has no additional coordinate from which an additive inverse, quotient carrier, order row, standard model, or host algebra could be read. Hence the downstream consumption boundary is exactly the finite semiring source displayed above. □

Lean4 ✓ `BEDC.Derived.SemiringUp.SemiringUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Semiring}^{\uparrow}}$.

Constructive story: A semiring packet is generated by an additive commutative-monoid row, a multiplicative monoid or semigroup row, a shared carrier and classifier ledger, distributivity rows, zero-annihilation rows, hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Semiring}^{\uparrow}}$, publicClosed)

Scope closed: Definition 327.1, Definition 327.5, Definition 327.6, and Definition 327.7 bind the finite carrier, classifier, additive surface, multiplicative surface, distributive rows, zero-annihilation rows, transports, routes, provenance, and local name as the public $\mathsf{Semiring}^{\uparrow}$ source surface for $\mathsf{Ring}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, and $\mathsf{Mat}^{\uparrow}$ consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Semiring}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SemiringUp.SemiringFiniteSource\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close rings, modules, polynomial evaluation, matrix algebra, additive inverses, quotient carriers, ordered semirings, or a standard bridge.

Upgrade path: Bridged closure requires a standard-bridge boundary tying the public finite semiring source to downstream algebra consumers.

Chapter 328

A Concrete Naming Certificate for NestedDyadicInterval$^{\uparrow}$

The NestedDyadicInterval$^{\uparrow}$ interface names finite decreasing dyadic interval chains positioned between the public DyadicInterval$^{\uparrow}$ endpoint packet and the regular rational and real-name layers. Moore–Kearfott–Cloud interval analysis treats nested interval enclosures as standard constructive data; here the carrier is only a finite-window BEDC packet and carries no host completeness principle.

328.1 Finite decreasing dyadic chains

Definition 328.1 (NestedDyadicInterval finite-chain carrier)**.** *A* NestedDyadicInterval$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I_0, \ldots, I_n, S, R, P, L)$$

with the following visible rows:

1. *each I_i is a* DyadicInterval$^{\uparrow}$ *endpoint packet with lower and upper* DyadicRatCore$^{\uparrow}$ *endpoint rows and its displayed interval-order ledger;*
2. *S is the unary* StreamName$^{\uparrow}$ *schedule naming the finite prefix that selects the displayed interval chain;*
3. *R is the nested-refinement ledger: for every displayed successor pair, R stores the* Cont *route and* hsame *rows witnessing that I_{i+1} refines inside I_i endpointwise;*
4. *P is the* Pkg *provenance naming the consumed* DyadicRatCore$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *certificate surfaces;*
5. *L is the ledger that hides only the unobserved tail beyond the displayed finite chain and records that every public read factors through the shown intervals, S, R, and P.*

The carrier predicate is NestedDyadicIntervalPacket(K)*. Its dependency list is exactly* DyadicRatCore$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Theorem 328.2 (NestedDyadicInterval dyadic interval source route)**.** *Every accepted* NestedDyadicInterval$^{\uparrow}$ *carrier first consumes the displayed endpoint and order rows of chapter 306 and the dyadic endpoint ledger of chapter 291 before its nested-refinement ledger,* StreamName$^{\uparrow}$ *schedule,* RegSeqRat$^{\uparrow}$ *handoff, or* Real$^{\uparrow}$ *seal can be read.*

Thus downstream regular-rational and real-name consumers see nested-chain data only through those sibling source rows together with hsame*,* Cont*,* Pkg*, and* NameCert*. The route exports no completed intersection point, quotient interval, host real interval, hidden infinite tail, or completeness witness.*

Proof. Project the accepted carrier as $K = (I_0, \ldots, I_n, S, R, P, L)$ through Definition 328.1. The first carrier row says that each displayed I_i is already a $\mathsf{DyadicInterval}^{\uparrow}$ endpoint packet, with lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows and the displayed interval-order ledger attached to that packet.

Read the provenance row P next. It is the source route that names the consumed $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{DyadicInterval}^{\uparrow}$ certificate surfaces before the nested-refinement ledger, schedule, regular-rational handoff, or real seal is exposed. Therefore a later read of R, S, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, or the $\mathsf{Real}^{\uparrow}$ boundary can only replay the sibling dyadic rows already named by P.

The remaining public coordinates add only transport, containment, package, and naming evidence: hsame compares endpoint rows, Cont records adjacent containment, Pkg records provenance, and NameCert records the local certificate surface. None of these coordinates contains a completed intersection point, quotient interval, host real interval, hidden infinite tail, or completeness witness, so the downstream consumer receives exactly the displayed finite route claimed in the statement. □

Definition 328.3 (NestedDyadicInterval finite-window classifier)**.** *For accepted carriers* $K = (I_0, \ldots, I_n, S, R, P, L)$ *and* $K' = (I'_0, \ldots, I'_m, S', R', P', L')$*, the predicate* $\mathsf{NestedDyadicIntervalClassifier}(K, K')$ *says that every shared scheduled window transports endpointwise and that every common window has the same refined dyadic enclosure.*

Concretely, for each unary index requested by both S *and* S'*, the classifier supplies displayed* hsame *rows between the lower endpoints, displayed* hsame *rows between the upper endpoints, the matching* Cont *routes for the interval provenance, and equality of the nested-refinement ledger read on that common prefix. It does not quotient infinite chains and does not identify any completed intersection point.*

Theorem 328.4 (NestedDyadicInterval classifier laws)**.** *For accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *finite-chain carriers, the componentwise classifier on interval rows, dyadic endpoint rows, containment ledgers, scheduled window rows,* hsame*,* Cont*,* Pkg *provenance, and* NameCert *provenance is reflexive, symmetric, and transitive.*

Moreover, if a packet is related to an accepted carrier by this classifier and the displayed interval, endpoint, containment, window, route, package, and certificate rows transport through the classifier data, then the related packet is again an accepted $\mathsf{NestedDyadicInterval}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalClassifier laws`

Proof. Project an accepted carrier $K = (I_0, \ldots, I_n, S, R, P, L)$ through Definition 328.1. Reflexivity is obtained rowwise from the displayed hsame reflexivity on each dyadic endpoint row, the identity Cont route on each containment ledger entry, and the unchanged schedule, provenance, and tail ledger.

For symmetry, reverse each displayed endpoint hsame row and read the common scheduled window in the opposite direction. The containment rows are not replaced by a completed interval object; they are the same finite ledger entries read through the reversed classifier comparison on the shared window.

For transitivity, compose the endpoint hsame rows and compose the displayed Cont routes in each adjacent refinement slot. The scheduled-window rows align by the common finite prefix, so the containment ledger and provenance rows compose component by component.

If a classifier-related packet is transported from an accepted carrier, the preceding rowwise transports rebuild the finite list of interval packets, the nested-refinement ledger, the schedule, the Pkg row, the NameCert provenance, and the hidden-tail ledger. Repacking these rows gives an accepted carrier. No completed real point, quotient interval object, open-cover compactness witness, or choice of an infinite tail is introduced. □

Theorem 328.5 (NestedDyadicInterval window transport)**.** *Let $K = (I_0, \ldots, I_n, S, R, P, L)$ be an accepted* NestedDyadicInterval$^{\uparrow}$ *carrier. If each displayed interval I_i is transported endpointwise through* hsame *rows and through the same* Cont *routes named by R and P, and if the transported schedule names the same unary finite prefix, then the transported packet is again an accepted* NestedDyadicInterval$^{\uparrow}$ *carrier.*

The transported classifier reads the same finite decreasing dyadic chain on every shared scheduled window.

Proof. Project the carrier of Definition 328.1. Each visible interval row is already a DyadicInterval$^{\uparrow}$ packet, so endpointwise transport supplies transported lower endpoint rows, transported upper endpoint rows, and the interval-order ledger attached to the same displayed slot.

Now transport the refinement ledger. For each adjacent pair, the Cont route and hsame rows in R say that the successor interval refines inside its predecessor. Applying the same routes after endpoint transport gives the corresponding adjacent refinement row for the transported chain.

Finally transport S, P, and L. The schedule still exposes the same unary finite prefix, the provenance still names the same source certificate boundaries, and the ledger still hides only the tail beyond the displayed chain. Repacking these finite rows yields the accepted transported carrier and the same classifier surface on common windows. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket window transport`

Theorem 328.6 (NestedDyadicInterval RegSeqRat handoff)**.** *An accepted* NestedDyadicInterval$^{\uparrow}$ *carrier hands a* RegSeqRat$^{\uparrow}$ *consumer only finite-window data: the unary* StreamName$^{\uparrow}$ *schedule, the displayed* DyadicInterval$^{\uparrow}$ *endpoint packets, the nested-refinement ledger connecting adjacent visible intervals, and the* Pkg *provenance that ties these rows to* DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$.

A regular rational consumer may read the midpoint, endpoint, or width information available in the displayed dyadic enclosures, but the handoff does not certify a completed limit, a Dedekind cut, a quotient interval, or a host real interval.

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket regseqrat handoff`

Proof. Read the packet through Definition 328.1. The only RegSeqRat$^{\uparrow}$-facing coordinates are the scheduled finite prefix S, the endpoint packets I_i, the refinement ledger R, and the source provenance P.

For each displayed index, the DyadicInterval$^{\uparrow}$ row exposes lower and upper DyadicRatCore$^{\uparrow}$ endpoints and its interval-order ledger. The adjacent rows of R show that later visible intervals refine earlier visible intervals, so a regular rational consumer receives explicit finite enclosures and their displayed nesting evidence.

The tail ledger L states that nothing beyond the finite chain is public. Therefore the handoff cannot supply regularity beyond the displayed window, cannot select an intersection point, and cannot seal a host real interval. □

Theorem 328.7 (NestedDyadicInterval NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the finite* BHist *packet* $K = (I_0, \ldots, I_n, S, R, P, L)$ *of Definition 328.1;*
2. *classifier: the finite-window endpoint and refinement classifier of Definition 328.3;*
3. *window stability: the endpointwise transport statement of Definition 328.5;*
4. *handoff: the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *finite enclosure surface of Definition 328.6;*
5. *ledger exactness: the assertion that the hidden part of the name is only the unobserved tail beyond the displayed chain, while all public interval, schedule, refinement, provenance, and certificate reads are listed in the carrier.*

These rows are the full seed certificate surface for finite decreasing dyadic interval-chain names.

Proof. The carrier row fixes the visible source of every read: each interval is a $\mathsf{DyadicInterval}^{\uparrow}$ packet, the schedule is a $\mathsf{StreamName}^{\uparrow}$ prefix, and the provenance names exactly the dyadic rational, interval, stream, regular-rational, and real-name certificate surfaces.

The classifier row is the common-window comparison from Definition 328.3. It compares only displayed endpoints and displayed refinement ledgers, so it supplies the required stability relation without creating a quotient of chains.

Window stability is Definition 328.5. The handoff row is Definition 328.6. Ledger exactness follows from the coordinate L: every public read factors through $I_0, \ldots, I_n$, S, R, and P, while only the tail outside the finite chain is hidden. Thus the five rows exhaust the local naming obligation. □

Theorem 328.8 (NestedDyadicInterval singleton chain vacuity)**.** *Let* K *be an accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *carrier as in Definition 328.1. If the displayed interval chain of* K *has exactly one row, so that*

$$K = (I_0, S, R, P, L)$$

with no displayed interval I_i *for* $i > 0$*, then the nested-refinement ledger* R *has no successor-pair read. Consequently the public finite-window handoff of* K *is exactly the single* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint packet* I_0*, the unary* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the provenance row* P*, and the tail ledger* L*; there is no public adjacent-refinement datum to expose.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket singleton chain vacuity`

Proof. By Definition 328.1, the only entries recorded in the nested-refinement ledger R are the rows attached to displayed successor pairs I_{i+1} inside I_i. A carrier whose displayed chain consists only of I_0 has no displayed source index i for which both I_i and I_{i+1} occur in the chain. Hence the indexing set of successor-pair reads is empty, and every requested read of R over a displayed adjacent pair is vacuous.

The remaining public coordinates are read directly from the same carrier definition. The visible interval coordinate is the single endpoint packet I_0; the schedule coordinate is S; the provenance coordinate is P; and the ledger coordinate is L, which hides only the unobserved tail beyond the displayed finite chain and records that public reads factor through the shown data. Since R contributes no displayed successor-pair row in the singleton case, the handoff surface described in Definition 328.6 reduces exactly to I_0, S, P, and L together with the assertion that the adjacent-refinement component is empty. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket empty refinement source`

Theorem 328.9 (NestedDyadicInterval prefix truncation stability)**.** *Let $K = (I_0, \ldots, I_n, S, R, P, L)$ be an accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *carrier as in Definition 328.1. For any displayed prefix $I_0, \ldots, I_k$ with $k \leq n$, the restricted packet*

$$K_{\leq k} = (I_0, \ldots, I_k, S_{\leq k}, R_{\leq k}, P, L_{\leq k})$$

is again an accepted $\mathsf{NestedDyadicInterval}^{\uparrow}$ *carrier, where $S_{\leq k}$ requests only the prefix rows, $R_{\leq k}$ contains exactly the successor-pair rows with source index below k, and $L_{\leq k}$ hides the unrequested suffix together with the original tail. The classifier on $K_{\leq k}$ is the restriction of* $\mathsf{NestedDyadicIntervalClassifier}$ *to the displayed prefix window, and its* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is the corresponding restriction of Definition 328.6.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket prefix truncation stability`

Proof. Project the accepted packet through Definition 328.1. The interval rows $I_0, \ldots, I_k$ are already displayed $\mathsf{DyadicInterval}^{\uparrow}$ endpoint packets, so no endpoint object is reconstructed.

Restrict the schedule to the unary prefix positions at most k. Restrict the refinement ledger by deleting only those successor rows whose source interval is outside the requested prefix; every remaining row still carries the same Cont, hsame, Pkg, and $\mathsf{NameCert}$ provenance as in K. The endpoint rows on the smaller window are the transported endpoint rows supplied by Definition 328.5.

Finally repack the shared provenance row P with the ledger $L_{\leq k}$ that records both the original hidden tail and the now-unrequested suffix $I_{k+1}, \ldots, I_n$. Since every public read of the truncated packet is a public read already admitted by K, the carrier predicate holds. The classifier and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff are exactly the old classifier and handoff rows read on the smaller scheduled window; no completed point, quotient interval, hidden tail, or host real interval is introduced. □

Theorem 328.10 (NestedDyadicInterval adjacent refinement coverage)**.** *Let $K = (I_0, \ldots, I_n, S, R, P, L)$ be an accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *carrier. Every public read of the refinement ledger R is either empty when the displayed chain is the singleton chain, or is attached to a displayed adjacent pair I_{i+1} inside I_i for some $i < n$. In the nonempty case, the read exposes exactly the pair's* Cont *route, endpointwise* hsame *rows,* Pkg *source provenance, and* $\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$ *provenance recorded in the carrier.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket adjacent refinement coverage`

Proof. If the displayed chain has one interval row, Definition 328.8 says that there is no successor-pair index. Hence R contributes no public adjacent-refinement datum in that case.

Otherwise read R through Definition 328.1. The carrier definition assigns refinement-ledger rows only to displayed successor pairs, and each such row is indexed by a visible pair I_{i+1} inside I_i.

The same projection lists the row contents: the continuation route for the containment read, the endpointwise hsame witnesses, the package provenance naming the consumed dyadic-rational, interval, stream, regular-rational, and real-name certificate surfaces, and the local naming certificate provenance. There is no separate ledger coordinate from which a hidden refinement row could be read. □

Theorem 328.11 (NestedDyadicInterval RegSeqRat window exhaustion)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *packet can read only prefix-truncated endpoint packets, adjacent-refinement rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, and source provenance from* $\mathsf{DyadicRatCore}^{\uparrow}$,

DyadicInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *It cannot read a completed intersection point, a hidden infinite tail, a quotient interval, or a host-completeness witness from that packet.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket regseqrat window exhaustion`

Proof. Begin with the finite-window handoff of Definition 328.6. It identifies the consumer-facing coordinates as the schedule, the displayed endpoint packets, the adjacent refinement ledger, and the source provenance.

Apply Definition 328.9 to any scheduled finite window requested by the consumer. The requested endpoint data is therefore a prefix-truncated packet whose classifier is the restriction of the accepted packet classifier.

Apply Definition 328.10 to the ledger reads inside that prefix. Every refinement read is either empty in the singleton case or attached to a displayed adjacent pair with its recorded Cont, hsame, Pkg, and NameCert provenance.

The remaining coordinates are the finite StreamName$^{\uparrow}$ schedule and the same source provenance rows. Since the carrier ledger hides only unrequested suffix and tail data, no completed intersection point, hidden tail, quotient interval, or host-completeness witness can be projected by the regular rational consumer. □

Theorem 328.12 (NestedDyadicInterval finite prefix-induction derivation)**.** *For every accepted* NestedDyadicInterval$^{\uparrow}$ *packet and every displayed unary prefix length, induction on that prefix reconstructs exactly the endpoint rows, adjacent-refinement rows, containment ledger,* StreamName$^{\uparrow}$ *schedule,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *row visible in the prefix.*

The derivation preserves the chapter's refusal of completed intersection points, quotient intervals, hidden infinite tails, selected limits, and host completeness. No induction step adds a row outside the displayed finite BHist *window.*

Proof. For the base prefix, project the first interval row from Definition 328.1. The singleton-chain vacuity theorem Definition 328.8 gives an empty adjacent-refinement surface, while the displayed schedule, provenance, transport, replay, and local naming rows are already part of the same packet.

For the induction step, append one displayed refinement row. Apply Definition 328.9 to keep the extended prefix inside the accepted classifier, and then use Definition 328.10 to identify the new adjacent pair with its containment ledger, Cont route, endpointwise hsame rows, Pkg provenance, and local NameCert provenance.

The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumer boundary is then exhausted by Definition 328.11. It reads only prefix endpoint packets, adjacent-refinement rows, finite schedules, and source provenance, so the induction produces exactly the listed finite rows.

Finally apply ledger nonescape: the carrier ledger hides unrequested suffix and tail data rather than completing it. Hence the prefix induction cannot project a completed intersection point, quotient interval, hidden infinite tail, selected limit, or host-completeness witness. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket finite prefix induction derivation`

Theorem 328.13 (NestedDyadicInterval shared-prefix handoff)**.** *Let* $K = (I_0, \ldots, I_n, S, R, P, L)$ *and* $K' = (I'_0, \ldots, I'_{n'}, S', R', P', L')$ *be accepted* NestedDyadicInterval$^{\uparrow}$ *packets related by the finite-window classifier. If their schedules share a displayed unary prefix, then any* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer read on that prefix factors through the corresponding prefix-truncated endpoint packets,*

adjacent-refinement rows, hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *provenance on both sides.*

The handoff is determined by the displayed finite prefix alone. It does not export a completed intersection point, hidden infinite tail, quotient interval, host real interval, completeness witness, open-cover compactness datum, or standard bridge row.

Proof. Start with the classifier on K and K'. By Definition 328.5, a shared unary prefix transports only the displayed endpoint packets, schedules, refinement rows, and provenance coordinates.

Apply Definition 328.9 on both sides. The requested consumer window is therefore represented by prefix-truncated accepted packets, and the classifier restricts to the same displayed prefix rather than to an unobserved tail.

For each adjacent-refinement read inside the prefix, use Definition 328.10. The read is empty in the singleton case, or else attached to one displayed adjacent pair with its recorded Cont, hsame, Pkg, and NameCert rows.

Finally apply Definition 328.11. It exhausts the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer boundary by prefix endpoint packets, adjacent-refinement rows, finite schedules, and source provenance. Since the carrier ledger hides only data outside the displayed prefix, the shared-prefix read factors through the corresponding finite rows on both accepted packets. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket shared prefix handoff`

Theorem 328.14 (NestedDyadicInterval public NameCert export)**.** *The carrier of Definition 328.1, classifier of Definition 328.3, classifier laws, window transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, singleton-chain vacuity, prefix truncation stability, adjacent-refinement coverage, and* $\mathsf{RegSeqRat}^{\uparrow}$ *window exhaustion assemble the reusable public* $\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$ *export for finite decreasing dyadic chains.*

Every downstream read of this interface factors through displayed BHist *interval-chain rows, the unary* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint packets,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer boundary,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It exports no completed intersection point, hidden infinite tail, quotient interval, host real interval, completeness witness, open-cover compactness, or standard bridge.*

Proof. Project an accepted packet through the finite carrier of Definition 328.1. This exposes exactly the displayed dyadic interval endpoint packets, the unary schedule, the nested-refinement ledger, source provenance, and the ledger that hides only data outside the displayed finite window.

Use the classifier and classifier laws of Definition 328.3 to compare two accepted packets only through componentwise displayed endpoint packets, schedules, refinement rows, and provenance. Then apply Definition 328.5 to carry those rows across the requested finite window.

For any requested consumer window, apply Definition 328.9. The requested read is a prefix-truncated accepted packet, and its classifier is the corresponding restriction of Definition 328.3.

For ledger reads inside that prefix, apply Definition 328.8 and Definition 328.10. In the singleton case the adjacent-refinement surface is empty; otherwise each visible refinement read is one displayed adjacent pair together with its Cont, hsame, Pkg, and NameCert rows.

Finally apply Definition 328.11. It identifies the full consumer surface as prefix endpoint packets, adjacent-refinement rows, finite schedules, and source provenance, so every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ downstream read factors through the displayed finite BEDC rows and no excluded infinite or host-completeness datum is introduced. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket public finite window export`

Theorem 328.15 (NestedDyadicInterval standard finite-window bridge)**.** *A certified finite decreasing dyadic interval chain gives a standard bridge row for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers by exposing exactly its adjacent-refinement coverage, shared-prefix handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *window exhaustion, and public* $\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$ *export.*

The bridge claims no completeness principle, host real interval, Dedekind cut, open-cover compactness, quotient interval, hidden infinite tail, or selected limit.

Proof. Project the certified chain through Definition 328.10. Each visible ledger read is either the singleton empty surface or one adjacent refinement pair with its displayed Cont, hsame, Pkg, and NameCert rows.

Apply Definition 328.13. The requested consumer window is determined by the displayed finite prefix, so the handoff does not inspect an unobserved tail or any host interval coordinate.

Next use Definition 328.11. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer boundary is exhausted by prefix endpoint packets, adjacent-refinement rows, finite schedules, and source provenance.

Finally apply Definition 328.14. The public certificate repacks exactly those finite-chain rows as the standard finite-window bridge. Since the cited surfaces have no coordinates for completeness, host real intervals, Dedekind cuts, open-cover compactness, quotient intervals, hidden infinite tails, or selected limits, the bridge cannot claim them. □

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket standard bridge window`

Theorem 328.16 (NestedDyadicInterval successive refinement chain)**.** *Let* $I_0, \ldots, I_n$ *be a finite displayed chain of* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *entries whose adjacent pairs satisfy the displayed refinement rows. Then the adjacent rows compose into one accepted nested-dyadic-interval window over the same finite prefix,* BHist *ledger,* hsame *rows,* Cont *rows,* Pkg *provenance, and* NameCert *rows.*

The $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of the composed chain is exactly the shared-prefix handoff of that finite window. The composition does not select a limit, form a quotient interval object, or invoke a host completeness principle.*

Lean4 ✓ `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket successive refinement chain`

Proof. Begin with Definition 328.9. It restricts the displayed chain to the finite prefix inspected by the consumer and hides the unobserved tail.

Apply Definition 328.10 to each adjacent pair. Each step contributes only the displayed endpoint packets, refinement ledger, transport rows, package provenance, and local naming rows.

Use Definition 328.11 to exhaust the consumer boundary by the finite prefix and adjacent-refinement rows. This turns the list of adjacent steps into one accepted nested-dyadic-interval window.

Finally apply Definition 328.13 and Definition 328.14. The handoff is the shared-prefix handoff of the composed chain, and the public certificate reads only the same finite rows. Since the composed window has no coordinate for a selected limit, quotient interval, or host completeness principle, none is exported. □

Theorem 328.17 (NestedDyadicInterval refinement composition exhaustion)**.** *Let* $I_0, \ldots, I_n$ *be a finite displayed list of adjacent refinement rows composing a* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *window. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read and every* $\mathsf{Real}^{\uparrow}$ *consumer read of the composed window is exhausted by the displayed endpoint packets, the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, the adjacent* Cont *and* hsame *rows,* Pkg *provenance, and* NameCert *rows.*

Thus the composed refinement chain has no remaining consumer coordinate beyond the finite BEDC rows already displayed. It does not select a limit, form a quotient interval, invoke host completeness, read a hidden infinite tail, or use open-cover compactness.

Proof. Start with Definition 328.10. Each adjacent row contributes exactly its displayed endpoint packets, refinement ledger, Cont routes, hsame rows, Pkg provenance, and local NameCert rows.

Compose the adjacent rows with Definition 328.16. The finite list therefore becomes one accepted nested-dyadic-interval window over the same finite prefix and the same provenance rows.

Next apply Definition 328.11. This exhausts the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer boundary by the finite endpoint packets, finite schedule, adjacent refinement rows, and source provenance visible in the composed window.

Finally use Definition 328.13. The handoff identifies the consumer read with the shared finite prefix, so no selected limit, quotient interval, host-completeness witness, hidden infinite tail, or open-cover compactness datum is available to the composed chain. □

Theorem 328.18 (NestedDyadicInterval mature finite-window coverage)**.** *Every accepted* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *finite-chain consumer read is exhausted by the carrier, classifier, window transport, adjacent-refinement coverage, shared-prefix handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *window exhaustion, public* NameCert *export, and standard finite-window bridge rows already present in this chapter.*

The coverage contains no completeness principle, Dedekind cut, quotient interval, open-cover compactness datum, host real interval, hidden infinite tail, or selected limit.

Proof. Project the read through Definition 328.1 and Definition 328.3. The consumer sees only the finite endpoint packets, displayed schedule, adjacent-refinement ledger, hsame rows, Cont rows, Pkg provenance, and local NameCert rows admitted by the carrier and classifier.

Apply Definition 328.5. The requested read is transported inside the same finite packet surface, so no host interval, quotient interval, or hidden tail coordinate is added by changing representatives.

Next use Definition 328.10 together with the shared-prefix handoff already used by the chapter. The adjacent rows and the prefix rows cover every finite-chain consumer request, including the singleton case and each displayed refinement step.

Finally apply Definition 328.14 and Definition 328.15. The public certificate and the standard finite-window bridge repack exactly those finite rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, leaving no coordinate for completeness, Dedekind cuts, quotient intervals, open-cover compactness, host real intervals, hidden infinite tails, or selected limits. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$**.**

Constructive story: A nested dyadic interval packet is generated by a finite BHist chain of $\mathsf{DyadicInterval}^{\uparrow}$ endpoint packets, a unary $\mathsf{StreamName}^{\uparrow}$ schedule, a nested-refinement ledger using Cont and hsame, Pkg provenance for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and a ledger hiding only the unobserved tail.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 328.16, Definition 328.14, and Definition 328.15 bind the successive-refinement chain, finite-window export, standard finite-window bridge, BHist ledger, hsame, Cont, Pkg, and NameCert rows for $\mathsf{NestedDyadicInterval}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NestedDyadicInterval}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.NestedDyadicIntervalUp.NestedDyadicIntervalPacket\_public\_finite\_window\_export`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim completeness, Dedekind cuts, quotient intervals, open-cover compactness, host real intervals, hidden infinite tails, selected limits, or bridgeChecked status.

Upgrade path: Only Lean-side formal targets and bridge checking remain for this finite-window bridge boundary.

Chapter 329

A Concrete Naming Certificate for $\mathsf{StateSpaceModel}^{\uparrow}$

The $\mathsf{StateSpaceModel}^{\uparrow}$ interface records the finite state-transition packet shared by the control-facing certificates. Its BEDC surface consumes $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ for transition rows, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ for displayed matrices, $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ for state, input, and output rows, and $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ for the linear maps. The textbook reference point is Kailath, *Linear Systems* (1980).

The dependency anchors are $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, and $\mathsf{Riccati}^{\uparrow}$. The chapter names only the finite carrier consumed by those certificates, not a host trajectory space or an analytic realization theorem.

The transition row consumes the concrete dynamic-system certificate of chapter 195. The matrix rows consume chapter 49, and the state, input, and output carrier rows consume chapter 47. These are horizontal certificate dependencies of the displayed finite packet, not hidden analytic assumptions.

329.1 Finite state-transition carrier

Definition 329.1 (StateSpaceModel finite BHist carrier)**.** *A* $\mathsf{StateSpaceModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, U, Y, A, B, C, P, L)$$

whose displayed rows are:

1. *a state row X, an input row U, and an output row Y read through* $\mathsf{VecSpace}^{\uparrow}$*;*
2. *a transition matrix row A, an input-map matrix row B, and an observation-map matrix row C read through* $\mathsf{Mat}^{\uparrow}$*;*
3. *linear-map rows for $A : X \to X$, $B : U \to X$, and $C : X \to Y$ read through* $\mathsf{LinMap}^{\uparrow}$*;*
4. *a displayed finite transition ledger $x, u \mapsto Ax + Bu$ and observation ledger $x \mapsto Cx$ read through* $\mathsf{DynSystem}^{\uparrow}$*;*
5. Cont *routes for the transition, input, observation, and readback rows;*
6. hsame *transports for the state, input, output, matrix, linear-map, transition, observation,* Cont*, and provenance rows;*
7. Pkg *provenance row P naming the consumed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$ *certificates;*

8. a hidden-data ledger L recording that only the displayed rows are exported.

No host trajectory space, analytic semigroup, probability measure, infinite-horizon limit, external realization object, or unlogged state stream is a coordinate of the carrier.

Definition 329.2 (StateSpaceModel component classifier)**.** *The* $\mathsf{StateSpaceModel}^{\uparrow}$ *classifier compares two accepted packets componentwise. It compares the displayed state, input, and output traces through* $\mathsf{VecSpace}^{\uparrow}$*; the transition, input-map, and observation-map rows through* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{LinMap}^{\uparrow}$*; the finite transition and observation ledgers through* $\mathsf{DynSystem}^{\uparrow}$*; and the provenance through the named* Cont*,* hsame*, and* Pkg *rows.*

Two packets are classified together exactly when all displayed trace rows and all displayed provenance rows are hsame*-matched in this componentwise sense. The classifier has no coordinate for comparing hidden trajectories, host probability spaces, analytic flow objects, or limiting behavior.*

329.2 Control handoff surface

Theorem 329.3 (StateSpaceModel NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{StateSpaceModel}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the finite packet of Definition 329.1;*
2. *classifier: the componentwise comparison of Definition 329.2;*
3. *transport:* hsame *transport of state, input, output, matrix, linear-map, transition, observation,* Cont*, and* Pkg *rows preserves accepted packets;*
4. *finite trace ledger: every visible transition or observation read is one of the displayed $Ax + Bu$ or Cx ledger rows;*
5. *hidden-data ledger: the packet records that host trajectories, analytic realizations, stochastic data, and infinite-horizon objects are outside the exported carrier.*

Lean4 ✓ `BEDC.Derived.StateSpaceModelUp.StateSpaceModelPacket semantic name certificate`

Proof. Start with the carrier rows. They enumerate exactly the finite state, input, output, matrix, linear-map, transition, observation, continuation, transport, provenance, and hidden-ledger coordinates.

The classifier row is the componentwise comparison of those coordinates. Transport replaces each displayed coordinate by its hsame match and then repacks the same finite tuple, so accepted packets remain accepted.

The trace ledger rows are projections from the displayed transition and observation entries. Since the hidden-data ledger is also a coordinate of the packet, every consumer can see the boundary saying that the excluded host and limiting objects are not part of the exported surface. □

Lean4 ✓ `BEDC.Derived.StateSpaceModelUp.StateSpaceModelPacket namecert obligation surface`

Theorem 329.4 (StateSpaceModel control handoff)**.** *The control-facing certificates* $\mathsf{KalmanFilter}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, and* $\mathsf{Riccati}^{\uparrow}$ *may consume a* $\mathsf{StateSpaceModel}^{\uparrow}$ *packet only through the displayed finite state, input, output, transition, input-map, observation-map, trace-ledger,* hsame*,* Cont*, and* Pkg *rows of Definition 329.1. A downstream control certificate cannot demand an unlogged trajectory space, analytic flow, stochastic model, or limiting algebraic object from this handoff.*

Lean4 ✓ `BEDC.Derived.StateSpaceModelUp.StateSpaceModelPacket control handoff transport`

Proof. Read a downstream control request as a projection from the accepted carrier. The Kalman and Riccati consumers can project matrix, vector, covariance-facing, transition, and observation rows; the LQR consumer can project state, input, cost-adjacent transition, and horizon-facing rows; and the observability and controllability consumers can project the displayed C and B faces together with the transition row A.

All of these projections are among the finite rows named in Definition 329.1. The classifier of Definition 329.2 transports the same rows componentwise, so the handoff remains stable under representative change.

The excluded objects are not packet coordinates. Therefore no listed control-facing certificate can recover them from the $\mathsf{StateSpaceModel}^{\uparrow}$ handoff alone. □

Theorem 329.5 (StateSpaceModel ledger boundary)**.** *The* $\mathsf{StateSpaceModel}^{\uparrow}$ *ledger excludes host trajectory spaces, analytic semigroup objects, probability measures, and infinite-horizon limits. Every consumer-visible read factors through the displayed finite* BHist *rows for* X, U, Y, A, B, C, *finite transition and observation traces,* hsame *transport,* Cont *routing, and* Pkg *provenance.*

Proof. Project the accepted packet to its visible rows. The carrier lists state, input, output, matrix, linear-map, transition, observation, transport, continuation, provenance, and hidden-ledger entries, and no additional coordinate is available.

The hidden-data ledger states the exclusion boundary. A host trajectory space, analytic semigroup, probability measure, or infinite-horizon limit would require an additional coordinate or a separate certificate, so it cannot be read from this packet.

Thus every consumer-visible read is a finite BHist projection from the displayed state-space rows. □

Theorem 329.6 (StateSpaceModel obligation closure)**.** *Definition 329.1, Definition 329.2, Definition 329.3, Definition 329.4, and Definition 329.5 exhaust the finite* $\mathsf{NameCert}_{\mathsf{StateSpaceModel}^{\uparrow}}$ *obligation surface over* $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, Cont, hsame, *and* Pkg *rows.*

Every downstream control certificate reads only these finite rows and their displayed handoff boundaries.

Proof. Project the carrier coordinates. The visible tuple contains the state, input, output, matrix, linear-map, transition, observation, continuation, transport, package, and hidden-data ledger rows, and the classifier compares exactly those coordinates componentwise.

The NameCert obligation theorem records the same rows as the local certificate surface. The control handoff theorem then restricts Kalman, LQR, controllability, observability, and Riccati consumers to the displayed finite coordinates and trace ledgers.

The ledger-boundary theorem accounts for the excluded host trajectory, analytic-flow, stochastic, and infinite-horizon objects. Since no downstream control certificate can read outside the listed rows, the finite obligation surface is exhausted. □

Theorem 329.7 (StateSpaceModel public finite-transition export)**.** *Definition 329.1, Definition 329.2, Definition 329.3, Definition 329.4, Definition 329.5, and Definition 329.6 jointly export the public finite* $\mathsf{StateSpaceModel}^{\uparrow}$ *interface over* $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, Cont, hsame, *and* Pkg *rows.*

Kalman, LQR, controllability, observability, and Riccati consumers read only the displayed finite state, input, output, transition, input-map, observation-map, trace-ledger, transport, continuation, provenance, and local name rows.

Proof. Project the carrier rows. They display the state, input, output, matrix, linear-map, transition, observation, trace, continuation, transport, package, hidden-data ledger, and local naming coordinates used by the public $\mathsf{StateSpaceModel}^{\uparrow}$ packet.

Invoke the classifier and the NameCert obligation surface. Together they make the componentwise comparison and the local certificate boundary depend only on the displayed $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, Cont, hsame, and Pkg rows.

Use the control handoff, ledger boundary, and obligation closure to restrict downstream Kalman, LQR, controllability, observability, and Riccati reads to those finite rows. The same boundary excludes host trajectories, analytic semigroups, stochastic data, and infinite-horizon limits from the exported interface. □

Theorem 329.8 (StateSpaceModel public interface boundary)**.** *The public* $\mathsf{StateSpaceModel}^{\uparrow}$ *interface exports exactly the finite state, input, output, transition, input-map, observation-map, trace-ledger, classifier-transport,* hsame*,* Cont*,* Pkg*, and* NameCert *rows isolated by Definition 329.1, Definition 329.2, Definition 329.4, Definition 329.5, and Definition 329.6. No further public coordinate is part of the certificate boundary.*

Proof. Project the carrier and classifier definitions. They expose the finite state, input, output, transition, input-map, observation-map, trace-ledger, classifier-transport, continuation, sameness, package, and naming rows carried by a $\mathsf{StateSpaceModel}^{\uparrow}$ packet.

Compose those projections with Definition 329.4. The handoff routes every control-facing read through the same displayed rows, so no consumer-facing state-space coordinate can be read outside the finite certificate surface.

Apply Definition 329.5 and Definition 329.6 to close the projection under the visible ledger and obligation rows. These two results repack the projected rows as the public interface boundary and show that the exported surface is exhausted by the listed data.

The boundary therefore excludes host trajectory spaces, analytic semigroups, stochastic models, infinite-horizon limits, controllability tests, observability tests, filtering optimality, regulator optimality, and standard bridge data. □

Theorem 329.9 (StateSpaceModel standard finite-transition bridge)**.** *Any standard finite state-transition presentation accepted by this chapter is read into the same finite BEDC packet rows for state, input, output, transition, input map, observation map, trace ledger,* hsame*,* Cont*,* Pkg*, and* NameCert*. Conversely, every accepted* $\mathsf{StateSpaceModel}^{\uparrow}$ *packet exposes exactly that standard finite-transition surface and no further state-space coordinate.*

Proof. Start with an accepted standard finite-transition presentation. Apply Definition 329.7 to read the finite state, input, output, transition, input-map, observation-map, trace-ledger, transport, continuation, provenance, and local name rows.

Next apply Definition 329.8. This identifies the public interface with the same packet rows and prevents a consumer from reading outside the displayed $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, Cont, hsame, and Pkg surface.

For the converse direction, project an accepted packet through Definition 329.5. The ledger boundary says that every state-transition, input-map, observation-map, and trace read is one of the recorded finite rows, so the packet exposes exactly the standard finite transition surface.

The bridge does not assert controllability, observability, filtering optimality, regulator optimality, stochastic models, analytic trajectories, or infinite-horizon semantics; those objects are not coordinates of the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StateSpaceModel}^{\uparrow}}$.

Constructive story: A state-space-model packet is seeded by finite BHist rows for state, input, output, transition, input-map, observation-map, transition and observation traces, hsame transports, Cont routes, Pkg provenance, and a hidden-data ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StateSpaceModel}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public surface of Definition 329.7, Definition 329.8, and Definition 329.9 binds the finite state, input, output, matrix, linear-map, transition, observation, trace-ledger, transport, continuation, provenance, name, standard finite-transition bridge, and downstream control handoff rows to $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, Cont, hsame, and Pkg.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StateSpaceModel}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.StateSpaceModelUp.StateSpaceModelPacket\_semantic\_name\_certificate`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host trajectory spaces, analytic semigroups, probability measures, infinite-horizon limits, controllability tests, observability tests, filtering optimality, regulator optimality, or bridge-checked schema verification.

Upgrade path: The paper-axis treatment is mature; bridgeChecked status requires formal bridge-schema acceptance for the displayed state-space-model surface.

Chapter 330

A Concrete Naming Certificate for $\mathsf{FiniteVector}^{\uparrow}$

The $\mathsf{FiniteVector}^{\uparrow}$ interface names fixed-length finite component data over the already displayed $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, and optional $\mathsf{VecSpace}^{\uparrow}$ certificate surfaces. Pierce, Types and Programming Languages (2002), treats length-indexed finite sequences as typed finite data; the BEDC packet below keeps the same finite-index reading at the BHist level. It records the length row, component spine, index-component pairing rows, componentwise classifier ledger, read continuations, and package provenance, without forming an infinite sequence, quotient vector, basis choice, dimension theorem, or linear-independence principle.

The finite-vector source surface consumes chapter 28, chapter 36, chapter 34, and, when component provenance is present, chapter 47. These sibling rows supply the length, spine, index-component pairing, and optional component-provenance surfaces, so the $\mathsf{FiniteVector}^{\uparrow}$ carrier is not a kernel-only island.

330.1 Finite-vector carrier

Definition 330.1 (FiniteVector carrier)**.** *A* $\mathsf{FiniteVector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (n, L, I, C, R, P, H)$$

with the following visible rows:

1. *n is a* $\mathsf{Nat}^{\uparrow}$ *length row;*
2. *L is a* $\mathsf{List}^{\uparrow}$ *component spine whose displayed length is n;*
3. *I is a finite* $\mathsf{Prod}^{\uparrow}$ *spine of index-component pairing rows, one row for each visible index below n;*
4. *C is the optional component provenance row, read through* $\mathsf{VecSpace}^{\uparrow}$ *exactly when the entries are vector-space rows and omitted otherwise;*
5. *R is the componentwise* hsame *ledger comparing displayed components at matching indices;*
6. *P is the* Pkg *provenance tying the packet to the consumed* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*, and optional* $\mathsf{VecSpace}^{\uparrow}$ *certificate boundaries;*
7. *H is the ledger that hides only unused positions outside the displayed length and records every public component read through the displayed* Cont *routes.*

The carrier predicate is FiniteVectorPacket(n, L, I, C, R, P, H)*. It contains no infinite stream, no quotient of component rows, no ambient basis selection, and no dimension or linear-independence datum beyond the displayed finite length.*

Definition 330.2 (FiniteVector classifier)**.** *For accepted* FiniteVector$^{\uparrow}$ *carriers* $V = (n, L, I, C, R, P, H)$ *and* $V' = (n', L', I', C', R', P', H')$*, the classifier* FiniteVectorSame(V, V') *holds exactly when the following finite rows are displayed:*

1. *an* hsame *row comparing the* Nat$^{\uparrow}$ *length rows* n *and* n'*;*
2. *a* List$^{\uparrow}$ *classifier row comparing the component spines* L *and* L' *after length transport;*
3. *for each displayed index below the transported length, a* Prod$^{\uparrow}$ *classifier row comparing the corresponding index-component pairing rows in* I *and* I'*;*
4. *when vector-space component provenance is present, a* VecSpace$^{\uparrow}$ *classifier row comparing the corresponding component provenance rows* C *and* C'*;*
5. *the same* Cont *routes and* Pkg *provenance showing that every public read uses the displayed length, spine, pair rows, and componentwise* hsame *ledger.*

No other classifier is exported by the seed packet.

Theorem 330.3 (FiniteVector length-index stability)**.** *Let* $V = (n, L, I, C, R, P, H)$ *be an accepted* FiniteVector$^{\uparrow}$ *carrier. If the* Nat$^{\uparrow}$ *length row,* List$^{\uparrow}$ *component spine,* Prod$^{\uparrow}$ *index-component pair rows, optional* VecSpace$^{\uparrow}$ *component provenance, componentwise* hsame *ledger,* Cont *routes,* Pkg *provenance, and outside-length ledger are transported by displayed* hsame *along the same public routes named by* P*, then the transported packet* $V' = (n', L', I', C', R', P', H')$ *is again an accepted* FiniteVector$^{\uparrow}$ *carrier.*

Moreover, every public component read at a displayed index below n *transports to the matching read below* n'*, and the transported packet hides only positions outside the transported displayed length.*

Lean4 ✓ `BEDC.Derived.FiniteVectorUp.FiniteVectorPacket length index transport`

Proof. Project the seven coordinates of Definition 330.1. The Nat$^{\uparrow}$ row transports first, so the displayed length of the target packet is fixed before any component read is considered.

Transport the List$^{\uparrow}$ spine along that length row and then transport each Prod$^{\uparrow}$ index-component pair. The pair rows keep the index and the component in the same finite slot, so a read below the displayed length remains below the transported displayed length.

If the component provenance row is present, transport it through the stated VecSpace$^{\uparrow}$ classifier; otherwise the packet stays in the component-agnostic branch. The componentwise hsame ledger then repacks over the transported spine.

Finally transport the Cont routes, Pkg provenance, and outside-length ledger. Since H hides only positions outside the displayed length, its transported copy hides only positions outside the transported length, and the result is an accepted FiniteVector$^{\uparrow}$ carrier. □

Theorem 330.4 (FiniteVector componentwise ledger exactness)**.** *For accepted* FiniteVector$^{\uparrow}$ *carriers* V *and* V'*, a proof of* FiniteVectorSame(V, V') *is equivalent, as public BEDC data, to the finite componentwise ledger consisting of the transported length row, the transported list spine, the transported index-component pairing rows, the optional transported* VecSpace$^{\uparrow}$ *component provenance, and the displayed* hsame *comparison at every visible index.*

Consequently, the classifier does not compare hidden positions, does not form a quotient of component spines, and does not add a host vector equality outside the displayed ledger.

Lean4 ✓ `BEDC.Derived.FiniteVectorUp.FiniteVectorComponentLedger transport`

Proof. From a classifier proof, read off the five rows listed in Definition 330.2. They are exactly the length comparison, spine comparison, pair-row comparisons, optional vector-space provenance comparison, and componentwise hsame rows.

Conversely, suppose the finite ledger rows are displayed. Repack them in the order required by Definition 330.2: first the length row, then the list spine, then the pair rows, then the optional component provenance row, and finally the shared route and provenance rows.

The outside-length ledger is not part of the component comparison except for recording which positions are not publicly read. Therefore the equivalence exposes precisely the visible component rows and does not introduce a quotient, an infinite tail, or host vector equality. □

Lean4 ✓ `BEDC.Derived.FiniteVectorUp.FiniteVectorSame componentwise ledger exactness`

Theorem 330.5 (FiniteVector NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteVector}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the* BHist *packet* (n, L, I, C, R, P, H) *of Definition 330.1;*
2. *classifier and stability: the classifier of Definition 330.2 together with length-index transport from Definition 330.3;*
3. *length-index exactness: the visible* $\mathsf{Nat}^{\uparrow}$ *length row and the assertion that the* $\mathsf{List}^{\uparrow}$ *component spine has exactly that displayed length;*
4. *componentwise ledger: the* $\mathsf{Prod}^{\uparrow}$ *index-component pair rows and the* hsame *comparisons of Definition 330.4;*
5. *downstream consumer boundary: the* Pkg *and* Cont *rows exposing finite component reads to later vector, matrix, and finite-dimensional certificate consumers while hiding only positions outside the displayed length.*

These rows are the full public seed surface for the finite-vector name.

Lean4 ✓ `BEDC.Derived.FiniteVectorUp.FiniteVectorPacket semantic name certificate`
Lean4 ✓ `BEDC.Derived.FiniteVectorUp.FiniteVectorPacket namecert obligation surface`

Proof. The carrier row fixes the displayed length, component spine, pair rows, optional component provenance, componentwise ledger, provenance, and outside-length ledger. This gives the source domain of the finite-vector name.

The classifier and stability row is supplied by Definition 330.2 and Definition 330.3. Together they say that every accepted replacement is componentwise hsame transport along the displayed Cont routes.

The length-index and componentwise rows are projections of the same carrier. The length row bounds the public read surface, and the pair rows identify which component belongs to each visible index.

Finally the downstream boundary is exactly the Pkg, Cont, and outside-length ledger. Since no coordinate contains an infinite sequence, quotient vector, basis choice, dimension theorem, or linear-independence witness, the five listed rows exhaust the $\mathsf{NameCert}_{\mathsf{FiniteVector}^{\uparrow}}$ obligation surface. □

Theorem 330.6 (FiniteVector scoped kernel boundary)**.** *The* $\mathsf{FiniteVector}^{\uparrow}$ *carrier, classifier, length-index stability, componentwise ledger exactness, and* $\mathsf{NameCert}_{\mathsf{FiniteVector}^{\uparrow}}$ *obligation surface*

bind the public finite-vector surface exactly to $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, *optional* $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

The boundary exports no infinite stream, quotient vector, basis choice, dimension datum, or linear-independence datum. Later matrix, linear-map, and finite-dimensional consumers can read only the displayed finite length, spine, index-component rows, componentwise comparisons, routes, provenance, and local name certificate.

Proof. Start with the carrier of Definition 330.1. It displays the $\mathsf{Nat}^{\uparrow}$ length row, $\mathsf{List}^{\uparrow}$ component spine, $\mathsf{Prod}^{\uparrow}$ index-component rows, optional $\mathsf{VecSpace}^{\uparrow}$ provenance, componentwise ledger, route provenance, and outside-length ledger as one finite BHist packet.

The classifier of Definition 330.2 and the transport statement of Definition 330.3 identify how accepted replacements move along the displayed hsame and Cont routes. No comparison outside those rows is introduced.

The exactness theorem Definition 330.4 shows that visible component equality is precisely the finite length, spine, pair-row, optional component-provenance, and componentwise hsame ledger. The $\mathsf{NameCert}_{\mathsf{FiniteVector}^{\uparrow}}$ surface of Definition 330.5 then repacks those rows for downstream consumers.

The listed exclusions follow by projection: the carrier has no coordinate for an infinite stream, quotient vector, basis choice, dimension witness, or linear-independence witness. Therefore the public boundary is exactly the displayed kernel surface named above. □

Theorem 330.7 (FiniteVector standard schema bridge boundary)**.** *The public finite-vector surface of Definition 330.6 bridges to the standard bounded finite-sequence reading exactly through the* $\mathsf{Nat}^{\uparrow}$ *length row, the* $\mathsf{List}^{\uparrow}$ *component spine, the* $\mathsf{Prod}^{\uparrow}$ *index-component rows, optional* $\mathsf{VecSpace}^{\uparrow}$ *component provenance, the componentwise* hsame *ledger,* Cont *read routes,* Pkg *provenance, and the outside-length hiding ledger.*

The bridge exports no infinite stream, quotient vector, basis choice, dimension theorem, or linear-independence theorem.

Proof. Project the carrier of Definition 330.1. It exposes the length row, component spine, index-component rows, optional component provenance, componentwise ledger, read routes, package provenance, and outside-length ledger as the finite packet.

Use length-index stability from Definition 330.3. The displayed $\mathsf{Nat}^{\uparrow}$ length bounds every public component read, so accepted replacements preserve which $\mathsf{Prod}^{\uparrow}$ index-component rows are visible and which positions are hidden outside the length.

Apply Definition 330.4. Componentwise equality is exactly the visible hsame ledger on the displayed component spine, transported along the same Cont read routes and Pkg provenance.

Finally use Definition 330.6. The scoped kernel boundary exhausts all public finite-vector reads by the rows just projected. Since the packet has no coordinate for infinite streams, quotient vectors, basis choice, dimension, or linear independence, none of those structures is exported by the standard bridge. □

Theorem 330.8 (FiniteVector zero length has empty component-read surface)**.** *Let* $V = (n, L, I, C, R, P, H)$ *be an accepted* $\mathsf{FiniteVector}^{\uparrow}$ *carrier. If the displayed* $\mathsf{Nat}^{\uparrow}$ *length row* n *is the zero row* e *of Definition 28.1, then the public component-read surface of* V *is empty:*

1. *the* $\mathsf{List}^{\uparrow}$ *component spine* L *is displayed as the empty spine;*
2. *the* $\mathsf{Prod}^{\uparrow}$ *index-component spine* I *has no public index-component pair row;*
3. *the componentwise ledger* R *has no public component-comparison row;*

4. the ledger H records only outside-length hiding data and no public component read.

Moreover, if $V' = (n', L', I', C', R', P', H')$ is another accepted $\mathsf{FiniteVector}^{\uparrow}$ *carrier whose displayed* $\mathsf{Nat}^{\uparrow}$ *length row is also* e*, then a classifier row*

$$\mathsf{FiniteVectorSame}(V, V')$$

contains no indexed $\mathsf{Prod}^{\uparrow}$ *comparison and no componentwise read comparison. Its public comparison data are exactly the zero length comparison, the empty-spine* $\mathsf{List}^{\uparrow}$ *comparison, the displayed provenance rows, the shared* Cont *routes, and the outside-length ledgers.*

Proof. Project the accepted carrier through Definition 330.1. The first coordinate says that n is the displayed $\mathsf{Nat}^{\uparrow}$ length row, and the hypothesis identifies that row with the zero unary history e. The same carrier clause says that L is a $\mathsf{List}^{\uparrow}$ component spine whose displayed length is n. Therefore L has displayed length zero. By the zero endpoint exactness of Definition 36.79, a zero displayed list length is exactly the empty-spine readback. Thus the component spine contributes no component slot.

The third coordinate of Definition 330.1 defines I as a finite $\mathsf{Prod}^{\uparrow}$ spine with one index-component pairing row for each visible index below n. There is no visible index below the zero $\mathsf{Nat}^{\uparrow}$ row: a visible index below e would be a strict predecessor of the zero unary history, while Definition 28.1 generates zero as the empty base and generates larger unary rows only by one-extension. Hence the indexing set for I is empty, so the finite $\mathsf{Prod}^{\uparrow}$ spine has no public pair row.

The componentwise ledger R compares displayed components at matching indices. Since the preceding paragraph leaves no displayed index and the first paragraph leaves no component slot, there is no indexed component comparison for R to expose. The last carrier coordinate says that H hides only unused positions outside the displayed length and records every public component read through displayed Cont routes. A public component read requires a visible index below n together with its index-component row in I; both are absent at length zero. Therefore H records only the outside-length hiding data.

For the classifier statement, unfold Definition 330.2 for the two accepted zero-length carriers. The length row remains present and compares n with n'. The $\mathsf{List}^{\uparrow}$ row compares the two empty component spines obtained above. The indexed $\mathsf{Prod}^{\uparrow}$ clause ranges over displayed indices below the transported length, but that transported length is again zero, so the clause has no row to display. The optional component provenance coordinate can only contribute its displayed provenance declaration; it does not create an indexed component read because no component slot exists. Finally the Cont, Pkg, and outside-length ledger rows are exactly the remaining public rows named by the classifier. The exactness statement of Definition 330.4 confirms that no hidden position, quotient of spines, or host vector equality is added beyond this displayed ledger. □

Theorem 330.9 (FiniteVector successor length exposes terminal component read)**.** *Let $V = (n, L, I, C, R, P, H)$ be an accepted* $\mathsf{FiniteVector}^{\uparrow}$ *carrier. If the displayed* $\mathsf{Nat}^{\uparrow}$ *length row n is a successor row over a predecessor length m, then the public component-read surface of V decomposes into the predecessor finite-vector read surface of length m together with exactly one terminal* $\mathsf{Prod}^{\uparrow}$ *index-component row at the successor boundary.*

Equivalently, the displayed $\mathsf{List}^{\uparrow}$ *spine splits into the predecessor component spine and one terminal component slot, the* $\mathsf{Prod}^{\uparrow}$ *spine I splits into the predecessor index-component rows and the terminal index-component pairing row, the componentwise ledger R splits into the predecessor* hsame *comparison rows and the terminal component comparison, and H hides only positions outside the displayed successor length.*

Proof. Project the seven carrier coordinates from Definition 330.1. The first coordinate identifies n as the displayed $\mathsf{Nat}^{\uparrow}$ length row, and the hypothesis writes that row as the one-step successor of m. The zero endpoint of Definition 330.8 gives the base boundary for the induction over displayed unary length rows.

For the successor step, split the $\mathsf{List}^{\uparrow}$ spine along the successor form of the unary length row. The displayed prefix has length m, and the one remaining displayed slot is the terminal component slot at the successor boundary. No additional slot is available because the carrier requires the displayed length of L to be exactly n.

Now split the finite $\mathsf{Prod}^{\uparrow}$ spine I. By the carrier clause, I has one index-component pairing row for each visible index below n. The visible indices below the successor row are precisely the predecessor visible indices below m together with the terminal boundary index. Hence I is the predecessor pair-row spine plus exactly one terminal $\mathsf{Prod}^{\uparrow}$ index-component row.

Apply Definition 330.4 to identify the classifier rows exposed by this split. The predecessor part contributes the finite hsame rows over the predecessor spine, and the terminal pair row contributes exactly the terminal component comparison. The exactness theorem also rules out hidden-position comparisons, quotient spine equality, and host vector equality.

Finally inspect the last carrier coordinate. The ledger H records every public component read through the displayed Cont routes and hides only unused positions outside the displayed length. Since the displayed length is the successor row over m, the predecessor reads and the terminal read are public, while every position beyond that successor length remains hidden. □

Theorem 330.10 (FiniteVector public component recursion boundary)**.** *Every public component read of an accepted* $\mathsf{FiniteVector}^{\uparrow}$ *packet is generated by induction on the displayed* $\mathsf{Nat}^{\uparrow}$ *length row. The zero row is exactly the empty component-read surface of Definition 330.8; the successor row is exactly the predecessor read surface together with one terminal* $\mathsf{Prod}^{\uparrow}$ *index-component row, as in Definition 330.9.*

Consequently the public recursion boundary exports the displayed $\mathsf{Nat}^{\uparrow}$ *length, the finite* $\mathsf{List}^{\uparrow}$ *component spine, the finite* $\mathsf{Prod}^{\uparrow}$ *index-component rows, componentwise* hsame *ledger rows,* Cont *routes, and* Pkg *provenance. It exports no infinite stream, quotient vector, basis choice, dimension theorem, or linear-independence datum.*

Proof. Induct on the displayed unary $\mathsf{Nat}^{\uparrow}$ length row carried by Definition 330.1. In the zero case, apply Definition 330.8. The component spine, index-component spine, and componentwise ledger then expose no public component row.

In the successor case, apply Definition 330.9. The displayed packet splits into the predecessor read surface and exactly one terminal $\mathsf{Prod}^{\uparrow}$ index-component row at the successor boundary.

Finally use Definition 330.4 to identify the classifier ledger exposed by the induction. The ledger records precisely the displayed componentwise hsame rows and the associated $\mathsf{Cont}/\mathsf{Pkg}$ provenance, so it cannot supply an infinite stream, quotient vector, basis, dimension theorem, or linear-independence witness. □

Theorem 330.11 (FiniteVector standard bridge boundary)**.** *The public component-recursion packet of Definition 330.10 interprets as the standard finite-vector schema exactly through the displayed* $\mathsf{Nat}^{\uparrow}$ *length-index row, terminal component read, componentwise ledger,* $\mathsf{List}^{\uparrow}/\mathsf{Nat}^{\uparrow}/\mathsf{VecSpace}^{\uparrow}$ *provenance, visible* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows.*

The bridge carries no basis choice, quotient vector, infinite sequence, or dimension theorem. In particular, the standard reading is only the finite component packet generated by the displayed length recursion and its terminal read boundary.

Proof. First fix the length-index surface. Apply Definition 330.3 and Definition 330.5. These identify the displayed $\mathsf{Nat}^{\uparrow}$ length row, $\mathsf{List}^{\uparrow}$ component spine, optional $\mathsf{VecSpace}^{\uparrow}$ provenance, visible routes, package data, and local certificate rows used by the finite-vector packet.

Next read the component ledger. Use Definition 330.4 to identify visible component equality with the componentwise hsame rows, and use Definition 330.8 for the empty read at length zero.

For the successor boundary, apply Definition 330.9. It supplies the unique terminal $\mathsf{Prod}^{\uparrow}$ index-component row beyond the predecessor surface, so every displayed finite-vector component read is generated by the length recursion.

Finally apply Definition 330.10. The public recursion boundary repacks the length row, terminal reads, componentwise ledger, provenance, visible transports, and certificate rows as the standard finite-vector bridge. The carrier has no coordinate for basis choice, quotient vectors, infinite sequences, or a dimension theorem, so none is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVector}\uparrow}$.

Constructive story: A finite-vector packet is generated by a $\mathsf{Nat}^{\uparrow}$ length row, a $\mathsf{List}^{\uparrow}$ component spine, $\mathsf{Prod}^{\uparrow}$ index-component pair rows, optional $\mathsf{VecSpace}^{\uparrow}$ component provenance, componentwise hsame rows, Cont routes for public reads, Pkg provenance, and a ledger hiding only positions outside the displayed length.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteVector}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 330.6, Definition 330.7, Definition 330.8, Definition 330.9, Definition 330.10, and Definition 330.11 bind the finite-vector surface to $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, optional $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with zero-length and successor-length component-read surfaces serving as mature finite consumer examples.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteVector}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FiniteVectorUp.FiniteVectorPacket\_semantic\_name\_certificate`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close infinite sequences, quotient vectors, basis choice, a dimension theorem, or linear independence.

Upgrade path: No further paper-side theory grade remains for this finite length-indexed component package; bridge verification remains on the independent verification axis.

Chapter 331

A Concrete Naming Certificate for $\mathsf{Divisibility}^{\uparrow}$

The $\mathsf{Divisibility}^{\uparrow}$ interface names the finite arithmetic surface in which one natural-number row divides another by an explicitly displayed multiplication witness. In the sense of Hardy and Wright, *An Introduction to the Theory of Numbers*, divisibility is the relation that records a multiplier q with $bq = a$. The BEDC surface below keeps the relation constructive: the multiplier, product ledger, bounded search row, visible divisibility ledger, and provenance all remain finite BHist rows. It does not introduce quotient classes, ideal membership, or classical decidability.

The multiplier witness is read through the arithmetic rows supplied by chapter 28 and the $\mathsf{Mul}^{\uparrow}$ product surface, while bounded-search reads use the $\mathsf{Order}^{\uparrow}$ row. Downstream consumers such as chapter 64 and chapter 303 consume the divisibility ledger through the same finite NatUp, product, order, prime, and modular dependency surface.

Definition 331.1 (Divisibility finite-history carrier)**.** *A* $\mathsf{Divisibility}^{\uparrow}$ *source is a finite* BHist *packet*

$$D = (a, b, q, w, o, l, p)$$

with the following visible rows:

1. *a and b are accepted* $\mathsf{Nat}^{\uparrow}$ *rows naming the dividend and divisor;*
2. *q is an accepted* $\mathsf{Nat}^{\uparrow}$ *multiplier row;*
3. *w is the displayed* $\mathsf{Mul}^{\uparrow}$ *witness row whose multiplication ledger records $b \cdot q = a$;*
4. *o is an* $\mathsf{Order}^{\uparrow}$ *boundedness row recording the unary search envelope for q whenever the consumer asks for a bounded witness;*
5. *l is the visible divisibility ledger tying the a, b, q, w, and o rows to the same displayed classifier;*
6. *p is the* Pkg *provenance row naming the* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, *and* $\mathsf{ModularArithmetic}^{\uparrow}$ *certificate boundaries used by the packet.*

The carrier predicate is $\mathsf{DivLedger}(a, b, q, w, o, l, p)$. *Two accepted sources classify together only when all displayed rows transport componentwise by* hsame *and the multiplication and boundedness reads follow the* Cont *routes recorded in l and p.*

Lean4 (definition) `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier`

Theorem 331.2 (Divisibility multiplication-witness closure)**.** *Let $D = (a, b, q, w, o, l, p)$ be an accepted* $\mathsf{Divisibility}^{\uparrow}$ *source. The divisibility assertion exported by D is exactly the displayed* $\mathsf{Mul}^{\uparrow}$ *witness row w for $b \cdot q = a$, together with the* $\mathsf{Nat}^{\uparrow}$ *rows a, b, and q named in Definition 331.1. Every transport of D by the componentwise* hsame *rows in l gives another accepted source with the transported multiplication witness and no additional host divisibility predicate.*

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier mul witness transport closure`
Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityBHistCarrier mul witness closure`

Proof. Project the carrier of Definition 331.1. The rows a, b, and q supply the displayed natural-number data, and w is the only coordinate that reads a multiplication computation.

Transport each of these rows by the hsame data recorded in l. Since the ledger also carries the continuation route used to read multiplication, the transported packet has the same visible product comparison, now over the transported rows.

The remaining coordinates record boundedness and provenance. They contain no coordinate for an external divisibility relation, quotient class, or decidable predicate, so the multiplication witness row exhausts the exported divisibility assertion. □

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityLedger multiplication witness closure`

Theorem 331.3 (Divisibility order-bounded ledger)**.** *For an accepted source $D = (a, b, q, w, o, l, p)$, every bounded search read of the divisibility witness is exactly the* $\mathsf{Order}^{\uparrow}$ *row o together with the unary envelope recorded by l. The row bounds the displayed multiplier q inside the finite search region selected by the dividend row a when such a region is requested, and otherwise exports no order fact beyond the visible boundedness ledger.*

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityBHistCarrier order bounded ledger`

Proof. Read o through the carrier of Definition 331.1. It is typed over the same a, b, and q rows used by the multiplication witness, so its boundedness comparison cannot name a different multiplier.

When a bounded consumer requests a search envelope, the ledger l supplies the finite unary route that compares q with the selected bound. The route is a displayed order row, not a global well-ordering principle.

No coordinate in the source stores least witnesses, unbounded search, or classical existence. Thus the bounded read is exhausted by o and the finite envelope recorded by l. □

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier order bounded ledger`

Theorem 331.4 (Divisibility NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Divisibility}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the source of Definition 331.1 is inhabited by taking the divisor row b to be the certified natural row for 1, the multiplier row q to be a, and w to be the displayed* $\mathsf{Mul}^{\uparrow}$ *unit multiplication ledger;*
2. *classifier transport: the classifier compares accepted sources by componentwise* hsame *transport of a, b, q, w, o, l, and p;*
3. *multiplication-witness closure: Definition 331.2 is the only exported divisibility assertion;*
4. *order-bounded exactness: Definition 331.3 is the only exported bounded search surface;*

5. *downstream readback:* $\mathsf{Prime}^{\uparrow}$ *and* $\mathsf{ModularArithmetic}^{\uparrow}$ *consumers read only the divisor, multiplier, product, boundedness, ledger, and provenance rows of the same packet.*

These five rows form the full public seed obligation for naming finite divisibility data over certified natural-number histories.

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier namecert obligation surface`

Proof. The habitation row uses the displayed natural-number unit. With $b = 1$ and $q = a$, the $\mathsf{Mul}^{\uparrow}$ unit ledger supplies the witness row w, while the boundedness and divisibility ledgers are the finite rows that record this same calculation.

Classifier transport is componentwise. Transporting a, b, q, w, o, l, and p by their displayed hsame rows gives a packet with the same carrier shape and the same multiplication and boundedness continuation routes.

The multiplication and order theorems identify the only arithmetic reads exported from the carrier. Since the provenance row names exactly the natural, additive, multiplicative, order, prime, and modular certificates, downstream consumers can read no further object from the local $\mathsf{NameCert}_{\mathsf{Divisibility}^{\uparrow}}$ surface. □

Theorem 331.5 (Divisibility to Prime sibling route)**.** *Every* $\mathsf{Prime}^{\uparrow}$ *consumer of an accepted* $\mathsf{Divisibility}^{\uparrow}$ *row reads the displayed divisor, multiplier, multiplication witness, boundedness, ledger, and provenance rows of the carrier in Definition 331.1. The downstream prime certificate of chapter 64 consumes this route as a finite sibling dependency: divisor tests pass through b, q, w, o, l, and p, together with the dividend row a that indexes the same packet.*

Thus $\mathsf{Prime}^{\uparrow}$ *obtains no external divisibility predicate, quotient-like shortcut, gcd routine, least-witness selector, host congruence, or ambient number-theoretic decision principle from* $\mathsf{Divisibility}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier public prime modular consumer boundary`

Proof. Project the accepted row through Definition 331.1. The projection exposes the finite carrier $D = (a, b, q, w, o, l, p)$, with b the displayed divisor, q the displayed multiplier, w the multiplication witness, o the boundedness row, l the ledger, and p the provenance row.

Apply Definition 331.2. The only multiplicative claim available to the downstream prime packet is the displayed witness for $b \cdot q = a$; it is not an external divisibility predicate and it does not name a quotient carrier.

Apply Definition 331.3. The bounded part of the read is exhausted by o and l, so the route supplies a finite envelope rather than a least-witness search or a host decision procedure.

Finally read chapter 64 through the same provenance row p. The prime consumer receives the divisor-facing finite packet and no quotient-like shortcut, because every row used by the consumer is already one of the displayed $\mathsf{Divisibility}^{\uparrow}$ rows. □

Theorem 331.6 (Divisibility prime and modular boundary)**.** *Every* $\mathsf{Prime}^{\uparrow}$ *or* $\mathsf{ModularArithmetic}^{\uparrow}$ *consumer of an accepted* $\mathsf{Divisibility}^{\uparrow}$ *source must factor through the same finite rows $D = (a, b, q, w, o, l, p)$. A prime-oriented consumer may read b, q, and the product ledger w as the visible witness surface for divisor tests. A modular-arithmetic consumer may read the same witness as a zero-remainder certificate only through the displayed* $\mathsf{Mul}^{\uparrow}$ *product,* $\mathsf{Order}^{\uparrow}$ *boundedness row, and provenance carried by p.*

No primality decision, gcd computation, quotient residue class, finite-field structure, or host congruence is exported by the $\mathsf{Divisibility}^{\uparrow}$ *packet.*

Proof. Project the accepted source through Definition 331.1. The prime-facing read obtains only the divisor row, multiplier row, and multiplication witness already present in D.

For a modular consumer, the same witness is read through the product row and the boundedness ledger. The provenance row p names the $\mathsf{ModularArithmetic}^{\uparrow}$ boundary, but it does not add a quotient carrier or external congruence relation.

Thus both downstream reads factor through the finite source itself. Any primality decision, gcd computation, quotient residue class, finite-field law, or host congruence would require a coordinate absent from D. □

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier prime modular boundary`

Theorem 331.7 (DivisibilityUp public certificate)**.** *For every accepted finite* $\mathsf{Divisibility}^{\uparrow}$ *source* $D = (a, b, q, w, o, l, p)$*, Definition 331.1, Definition 331.2, Definition 331.3, Definition 331.4, and Definition 331.6 assemble the public* $\mathsf{NameCert}_{\mathsf{Divisibility}^{\uparrow}}$ *consumer certificate over* BHist*, multiplication witnesses, order-bounded ledgers,* hsame*,* Cont*, and* Pkg*.*

It exports no gcd algorithm, Euclidean division, primality decision, quotient residue class, finite-field law, or host congruence.

Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier public certificate export`
Lean4 ✓ `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier public consumer certificate`

Proof. Project D through Definition 331.1. The projection exposes only the dividend row, divisor row, multiplier row, multiplication witness row, boundedness row, divisibility ledger, and provenance row.

Apply Definition 331.2 to identify the only multiplicative read, and apply Definition 331.3 to identify the only bounded-search read. These two steps leave no coordinate for an external divisibility predicate or quotient object.

Apply Definition 331.4 to package the carrier, classifier transport, multiplication, boundedness, and downstream readback rows as the displayed $\mathsf{NameCert}_{\mathsf{Divisibility}^{\uparrow}}$ surface.

Finally, use Definition 331.6 to expose the same finite packet as the public $\mathsf{NameCert}_{\mathsf{Divisibility}^{\uparrow}}$ consumer certificate for $\mathsf{Prime}^{\uparrow}$ and $\mathsf{ModularArithmetic}^{\uparrow}$. Any gcd algorithm, Euclidean division, primality decision, quotient residue class, finite-field law, or host congruence would require a row absent from the certificate. □

Theorem 331.8 (Divisibility bounded-witness public readback)**.** *Every public consumer of an accepted* $\mathsf{Divisibility}^{\uparrow}$ *packet that asks for a bounded witness reads exactly the multiplier row* q*, the multiplication witness* w *for* $b \cdot q = a$*, the order envelope* o*, the ledger* l*, and the provenance row* p *from Definition 331.1. PrimeUp and ModularArithmeticUp consumers factor through those rows, together with the displayed dividend and divisor entries that index the same carrier packet.*

The public readback exports no gcd algorithm, least witness, quotient residue class, finite-field law, or host decidability predicate. Its complete bounded witness surface is the finite row family already displayed by the $\mathsf{Divisibility}^{\uparrow}$ *carrier, multiplication witness, order-bounded ledger, and prime/modular consumer boundary.*

Proof. Project the accepted packet through Definition 331.1. The only rows available to a bounded-witness consumer are the multiplier q, the multiplication witness w, the order envelope o, the ledger l, and the provenance row p, indexed by the displayed dividend and divisor entries.

Apply Definition 331.2. The product check is read solely as the displayed witness for $b \cdot q = a$, so the consumer receives no separate quotient object or external divisibility predicate.

Next apply Definition 331.3. The bounded part of the readback is exactly the order envelope and ledger row already present in the carrier, not a least-witness search or a host decidability routine.

Finally use Definition 331.6. PrimeUp and ModularArithmeticUp can consume the bounded witness only through the same finite rows, so no gcd algorithm, quotient residue class, finite-field law, or host decidability predicate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{Divisibility}\uparrow}$.

Constructive story: Divisibility is generated by finite natural-number rows for a, b, and q, a displayed multiplication witness for $b \cdot q = a$, an optional bounded search row, a visible divisibility ledger, and provenance over the arithmetic dependency certificates.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Divisibility}\uparrow}$, maturePackageClosed)

Scope closed: Definition 331.1, Definition 331.2, Definition 331.3, Definition 331.4, Definition 331.5, and Definition 331.6 bind the carrier, multiplication-witness row, order-bounded ledger, NameCert obligation surface, Prime sibling route, and prime/modular consumer boundary for finite divisibility data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Divisibility}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.DivisibilityUp.DivisibilityFiniteHistoryCarrier\_order\_bounded\_ledger`

Bridge status: none

Origin: human

Not claimed: This chapter does not close gcd algorithms, Euclidean division, primality decisions, unique factorization, quotient residue classes, finite fields, or a standard bridge.

Upgrade path: The paper-side theory axis is complete for the finite divisibility consumer catalogue over the multiplication-witness, bounded-ledger, prime/modular, hsame, Cont, and Pkg rows.

Chapter 332

A Concrete Naming Certificate for $\mathsf{ModulusOfConvergence}^{\uparrow}$

The $\mathsf{ModulusOfConvergence}^{\uparrow}$ interface names the explicit convergence-rate packet used by the Bishop–Bridges constructive-analysis reading of regular rational sequences and real completion. It is a support object for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, adjacent to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, and $\mathsf{NestedDyadicInterval}^{\uparrow}$. It records finite precision requests and late observation witnesses; it does not form a quotient completion, choose a limit, or appeal to host real equality. Its concrete dependency route consumes chapter 291 for finite precision rows, chapter 126 for stream observation windows, chapter 301 for regular-readback witnesses, and chapter 38 for the real-seal handoff.

Definition 332.1 (ModulusOfConvergence BHist carrier)**.** *A* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *carrier packet is a finite* BHist *row package*

$$M = (p, u, \mu, s, r, \ell, P)$$

with the following visible coordinates:

1. *p is the precision-request history row, read as a unary or dyadic precision demand through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *u is the unary index selector row that chooses the threshold index bound returned for the request p;*
3. *μ is the modulus-function row, read as the displayed assignment from precision rows to threshold rows;*
4. *s is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule whose entries are tested after the selected bound;*
5. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *late-observation witness row showing that every displayed observation past the chosen bound satisfies the requested precision;*
6. *ℓ is the finite ledger recording the precision request, selected bound, late-observation witness, and the dependency anchors* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*;*
7. *P is the* Pkg *provenance for the same dependency anchors and for the local* $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$ *row.*

The carrier accepts exactly those finite packets whose listed rows are present in one history and are connected by displayed Cont *windows. It contains no infinite choice of thresholds, quotient stream, completed real, or hidden limit selector.*

Theorem 332.2 (ModulusOfConvergence NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$ *obligation surface consists of the carrier packet of Definition 332.1, the classifier identifying two packets when corresponding unary precision rows choose* hsame *index bounds and the same late-observation ledger, the* hsame *transport rows for precision and threshold histories, the* Cont *extension rows for finite observation windows, and the exactness ledger naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceBHistCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket namecert obligation surface`

Proof. Project an accepted packet to the seven carrier coordinates. The precision row, unary selector, modulus row, observation schedule, late-observation witness, ledger, and provenance row are all visible in the same finite BHist package.

The classifier compares the precision and threshold coordinates through hsame and then compares the late-observation ledger through the displayed source certificates. The same ledger rows name the dyadic precision source, the stream observation source, the regular-sequence witness, and the real-name consumer boundary.

The stability and exactness fields are therefore just the finite transport and window-extension rows already carried by the packet. No additional completion, choice, or quotient coordinate is available for the local naming certificate. □

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket semantic name certificate`

Theorem 332.3 (ModulusOfConvergence composition stability)**.** *If two accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packets have composable* Cont *windows, then their composed packet preserves the precision request, the selected threshold bound, the modulus-function row, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *late-observation witness, and the dependency ledger through the same* hsame *classifier. The composition exports only the displayed finite convergence-rate ledger.*

Proof. Restrict each packet to the finite rows named by Definition 332.1. The only data available for composition are the precision request, unary selector, modulus row, observation schedule, late witness, ledger, and package provenance.

Compose the Cont windows along their shared displayed histories. The composite window keeps the same selected threshold bound for the transported precision request and carries forward the same late-observation witness rows.

Finally transport the resulting rows through the componentwise hsame classifier of Definition 332.2. The composed packet has the same carrier shape and no constructor for a chosen limit or quotient completion, so composition is stable inside the finite ledger. □

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket composition stability`
Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceCarrier composition stability`

Theorem 332.4 (ModulusOfConvergence tail restriction stability)**.** *For every accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packet and every tail restriction whose first index is* hsame*-above the selected bound, the restricted packet is again accepted. Its classifier keeps the same precision request, modulus row, late-observation witness, and dependency ledger, while the* Cont *window is restricted to the displayed tail.*

Lean4 ✓ BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceRatePacket tail restriction stability
Lean4 ✓ BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceCarrier tail restriction stability
Lean4 ✓ BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket tail restriction stability

Proof. Start with the selected threshold bound in the unary index selector. A tail whose first index is hsame-above that bound remains inside the late window named by the precision request.

Restrict the observation schedule and late-observation witness rows to that tail. The dyadic endpoint rows and regular-sequence witness rows are inherited from the original packet because the carrier ledger already records them for all displayed observations beyond the bound.

Repack the restricted schedule with the same precision request, modulus row, ledger, and provenance. The Cont route is the restricted route, and the componentwise hsame comparisons are unchanged on the retained rows, so the tail packet is accepted by the same classifier. □

Theorem 332.5 (ModulusOfConvergence paired late-window common refinement)**.** *Let M_0 and M_1 be accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packets with transported precision requests, selected threshold rows, and displayed late-tail restrictions. If both packets are consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *reads over the same displayed dependency boundary, then there is one common displayed late window through which both restricted schedules and both late-observation witnesses factor by* hsame *and* Cont*.*

The common window exposes only the two precision rows, the transported threshold rows, the modulus rows, the observation schedules, the late-observation witnesses, the dependency ledgers, the Pkg *provenance rows, and the local* $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$ *rows. It does not expose quotient completion, a selected real limit, a global threshold choice, or host real equality.*

Proof. Apply Definition 332.4 to M_0 and M_1. Each transported precision request keeps its selected threshold row and restricts its observation schedule and late-observation witness to a displayed late tail.

Next apply Definition 332.3 to the two restricted tails along their displayed Cont windows. The composite window is the common late window, and both restricted schedules factor through it by the componentwise hsame classifier.

Use Definition 332.6 to identify the rows visible to downstream consumers. The only exposed coordinates are the two precision rows, transported thresholds, modulus rows, schedules, witnesses, dependency ledgers, package provenance rows, and local naming-certificate rows.

Finally apply Definition 332.8. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads therefore share that finite late-window ledger, and no quotient completion, selected real limit, global threshold choice, or host real equality is introduced. □

Theorem 332.6 (ModulusOfConvergence ledger exactness)**.** *An accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packet exposes exactly the precision request, selected bound, late-observation witness, and dependency anchors* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *recorded in its ledger. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet factors through those rows and the displayed* hsame*,* Cont*, and* Pkg *provenance.*

Proof. Read the ledger coordinate ℓ of Definition 332.1. By definition it records the precision request, the bound selected by the unary selector, the late-observation witness, and the four dependency anchors.

Apply the obligation surface of Definition 332.2. It permits downstream reads only through the componentwise classifier, finite window extension, and package provenance already present in the carrier.

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can inspect exactly the displayed convergence-rate ledger and no extra completion object, quotient equality, global modulus choice, or selected real limit. □

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket ledger exactness`

Theorem 332.7 (ModulusOfConvergence scoped dependency packet)**.** *For an accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packet, the carrier, classifier, composition, tail-restriction, and ledger exactness rows form one scoped BEDC dependency packet over* BHist*. Precision and threshold transport are read only through* hsame*, finite observation windows are read only through* Cont*, provenance is read only through* Pkg*, and the public certificate row is read only through* $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$*.*

The packet preserves the displayed $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *anchors. It excludes quotient completion, selected real limits, and global threshold choice.*

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceCarrier scoped dependency packet`

Proof. Project the seven carrier coordinates of Definition 332.1. They are the precision request, unary threshold selector, modulus row, observation schedule, late-observation witness, dependency ledger, and package/name provenance surface.

Apply Definition 332.2 to bind the accepted packet to the displayed classifier, transport rows, carrier obligations, and NameCert registration.

Apply Definition 332.3 and Definition 332.4. They keep the same BHist, hsame, Cont, and Pkg rows stable under composition and tail restriction.

Finally apply Definition 332.6. It says that every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer read factors through the listed BEDC rows and keeps the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ anchors visible. Hence no completion object, quotient equality, selected limit, or global threshold choice is exposed by the scoped packet. □

Theorem 332.8 (ModulusOfConvergence RegSeqRat and RealUp public export)**.** *An accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packet exports a public consumer package for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*. The package consists exactly of the precision request, unary threshold selector, modulus row,* $\mathsf{StreamName}^{\uparrow}$ *observation schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *late-observation witness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger,* hsame *transports,* Cont *windows, and* Pkg *provenance displayed in this chapter.*

Every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the accepted packet factors through this public package. Such a consumer cannot read a completion coordinate, quotient equality, selected limit, or global choice of thresholds.*

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceRatePacket semantic name certificate`

Proof. The seven carrier rows are those of Definition 332.1: the precision request, unary threshold selector, modulus row, observation schedule, late-observation witness, dependency ledger, and provenance/certificate surface.

The carrier, classifier, and stability fields are fixed by Definition 332.2. Late-window stability is supplied by Definition 332.4, so changing the visible observation bound transports only the displayed BHist, hsame, Cont, and Pkg rows.

Consumer factorization is exactly Definition 332.6. The exact BEDC scope is then the scope of Definition 332.7: the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ anchors are retained, and no completion coordinate, quotient equality, selected limit, or global threshold choice is available. □

Theorem 332.9 (ModulusOfConvergence finite-rate bridge packet)**.** *Any standard bridge for the displayed convergence-rate package may read exactly the precision request, unary threshold selector, modulus row,* $\mathsf{StreamName}^{\uparrow}$ *observation schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *late-observation witness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger,* hsame *transports,* Cont *windows, and* Pkg *provenance isolated by Definition 332.8. It may not read quotient completion, selected real limits, global threshold choice, or host real equality.*

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceFiniteRateBridge rows`

Proof. Project the carrier through Definition 332.1. The visible rows are the finite precision request, unary threshold selector, modulus row, scheduled observations, late-observation witness, endpoint ledger, and package/name provenance.

Apply Definition 332.2 to bind these rows to the accepted $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$ surface. Tail restriction through Definition 332.4 keeps the same BHist, hsame, Cont, and Pkg coordinates when a consumer narrows the visible observation window.

Ledger exactness is supplied by Definition 332.6. The dependency boundary is the packet of Definition 332.7, and the public $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ read is exactly Definition 332.8.

The excluded data are not coordinates of the carrier. A bridge that reads a quotient completion, a selected real limit, a global threshold choice, or host real equality would therefore leave the displayed finite-rate package rather than using the boundary proved here. □

Theorem 332.10 (ModulusOfConvergence standard bridge)**.** *Let* $M = (p, u, \mu, s, r, \ell, P)$ *be an accepted* $\mathsf{ModulusOfConvergence}^{\uparrow}$ BHist *packet. Then* M *determines the standard explicit-modulus-of-convergence schema by reading exactly the precision request* p*, selected threshold* $u(p)$*, modulus row* μ*,* $\mathsf{StreamName}^{\uparrow}$ *observations* s*,* $\mathsf{RegSeqRat}^{\uparrow}$ *late-window witness* r*,* hsame *transports,* Cont *windows,* Pkg *provenance, dependency ledger* ℓ*, and* NameCert *row* P *displayed in this chapter.*

Conversely, any displayed standard explicit-modulus package whose rows are exactly those BEDC rows repacks as the same $\mathsf{ModulusOfConvergence}^{\uparrow}$ *public packet. The bridge exports no quotient completion, no selected real limit, and no global choice of thresholds outside the displayed finite threshold row.*

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceStandardBridge boundary`

Proof. Project the carrier of M using Definition 332.1. The precision request, selected unary threshold, modulus row, observation schedule, late-window witness, ledger, and package/name provenance are already the coordinates of the accepted packet.

The naming and obligation surface is the surface of Definition 332.2. It fixes the classifier, the displayed BHist rows, the hsame transports, the Cont windows, the Pkg provenance, and the NameCert row without adding any host-level completion coordinate.

Composition and tail restriction preserve the same visible package by Definition 332.3 and Definition 332.4. Ledger exactness then identifies every consumer read with one of the listed rows, and Definition 332.8 gives the public $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ export surface.

For the converse direction, start from a displayed standard package with exactly these BEDC rows. Repack the precision request, threshold row, modulus row, observation schedule, late-window witness, ledger, and provenance into the carrier tuple M. The same NameCert obligation surface and stability rows give acceptance, so the repacked object is the original public $\mathsf{ModulusOfConvergence}^{\uparrow}$ packet. □

Theorem 332.11 (ModulusOfConvergence threshold-ledger completeness)**.** *Every convergence-rate consumer of the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *packet is completely determined by the precision*

request, selected unary threshold, modulus row, StreamName$^{\uparrow}$ *schedule,* RegSeqRat$^{\uparrow}$ *late-window witness,* hsame *transports,* Cont *windows,* Pkg *provenance, dependency ledger, and* NameCert *row.*

The export contains no quotient completion, selected real limit, global threshold choice outside the finite ledger, or host real equality.

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceRatePacket threshold ledger completeness`

Proof. Start with the carrier of Definition 332.1. It presents the precision request, selected unary threshold, modulus row, stream schedule, late-window witness, dependency ledger, package provenance, and local naming evidence as the only rows in the accepted packet.

Apply Definition 332.2 and Definition 332.3. These identify the classifier, stability, hsame, Cont, Pkg, and NameCert coordinates and show that composition keeps consumer reads on the displayed finite threshold-ledger surface.

Tail restriction is handled by Definition 332.4. Ledger exactness then follows from Definition 332.6, so a consumer can inspect only the finite request, threshold, schedule, late-window, transport, continuation, provenance, and ledger rows.

The public export and standard bridge are exactly Definition 332.8, Definition 332.9, and Definition 332.10. Together they show that the RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ read and the standard explicit-modulus schema use the same finite rows and do not export quotient completion, selected real limits, global threshold choice outside the finite ledger, or host real equality. □

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceCarrier threshold ledger completeness`

Theorem 332.12 (ModulusOfConvergence mature tail consumer example)**.** *A* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer that narrows to a later displayed tail reads the same finite convergence-rate package: the precision request, unary threshold selector, modulus row,* StreamName$^{\uparrow}$ *observations,* RegSeqRat$^{\uparrow}$ *late-window witness,* DyadicRatCore$^{\uparrow}$ *endpoint ledger,* hsame*,* Cont*,* Pkg*, dependency ledger, and local* NameCert *row.*

The tail read contains no quotient completion, selected real limit, global threshold choice outside the finite ledger, or host real equality.

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceRatePacket tail consumer boundary`

Proof. Start with Definition 332.4. It keeps a later displayed tail on the same precision request, unary threshold selector, modulus row, observation schedule, and late-window witness.

Apply Definition 332.6. The endpoint and dependency ledgers exhaust the finite coordinates that the tail consumer can inspect.

Next use Definition 332.8. This gives the RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumer surface over the same finite convergence-rate rows, together with hsame, Cont, Pkg, and local NameCert evidence.

Finally apply Definition 332.10. The standard explicit-modulus bridge reads exactly the displayed BEDC rows, so narrowing to a later tail cannot introduce quotient completion, a selected real limit, a global threshold choice outside the ledger, or host real equality. □

Theorem 332.13 (ModulusOfConvergence synchronized double-tail restriction)**.** *Let an accepted* ModulusOfConvergence$^{\uparrow}$ BHist *packet display a precision request, unary threshold selector, modulus row,* StreamName$^{\uparrow}$ *observation schedule,* RegSeqRat$^{\uparrow}$ *late-window witness,* DyadicRatCore$^{\uparrow}$ *endpoint ledger, dependency ledger,* Pkg *provenance, and local* NameCert *row. Suppose it also displays two unary tail restrictions, each connected to the next packet by concrete* Cont *rows. The twice-restricted*

packet preserves the same precision request, selector, modulus row, late-window witness, dependency ledger, package provenance, and NameCert row as the original packet.

Moreover, every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the second tail factors through the original accepted packet, using only the two displayed* Cont *windows and the finite rows already present in that packet.*

Lean4 ✓ `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergenceCarrier synchronized double tail restriction`

Proof. Apply Definition 332.4 to the first displayed tail. This keeps the precision request, unary threshold selector, modulus row, late-window witness, dependency ledger, package provenance, and local NameCert row on the restricted packet.

Apply Definition 332.4 again to the restricted packet and its second displayed tail. The same rows are preserved through the second restriction, so the twice-restricted packet still reads the original finite convergence-rate data.

Use Definition 332.3 to compose the two displayed Cont windows. The composite window identifies the second tail as a single displayed restriction of the original packet, without introducing a new threshold choice or hidden representative.

Finally combine Definition 332.6, Definition 332.8, and Definition 332.10. Ledger exactness identifies the finite rows available to a consumer, the public export supplies the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ read surface, and the bridge shows that the second tail consumer factors through the original packet rather than through any quotient completion, selected real limit, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfConvergence}^{\uparrow}$ packet is generated by a visible precision request, a unary threshold selector, a displayed modulus row, scheduled $\mathsf{StreamName}^{\uparrow}$ observations, a $\mathsf{RegSeqRat}^{\uparrow}$ late-observation witness, and a finite dependency ledger naming $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 332.10, Definition 332.11, and Definition 332.12 give the standard bridge, complete threshold-ledger consumer surface, and mature tail consumer theorem for the explicit finite convergence-rate package over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfConvergence}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ModulusOfConvergenceUp.ModulusOfConvergencePacket\_semantic\_name\_certificate`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a quotient completion, a selected real limit, a global threshold choice outside the finite ledger, or a Lean-checked bridge.

Upgrade path: No further paper-side theory-axis upgrade remains for this chapter; formal verification remains the next axis.

Chapter 333

A Concrete Naming Certificate for $\mathsf{CauchyFilter}^{\uparrow}$

The $\mathsf{CauchyFilter}^{\uparrow}$ interface names the finite filterbase surface used by the real-completion boundary. Following Bishop-style Cauchy completion and the Kelley–Bourbaki language of filters and directed bases, the BEDC object keeps only explicit finite rows: directed finite-tail dyadic neighbourhood evidence over regular Cauchy observations, refinement ledgers, modulus rows, and handoff rows for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBall}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Definition 333.1 (CauchyFilter finite-filterbase carrier)**.** *A* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carrier is a finite* BHist *packet*

$$F = (B, R, M, L, P)$$

with the following visible rows:

1. *B is a finite list of directed finite-tail neighbourhood packets, each packet carrying a* $\mathsf{StreamName}^{\uparrow}$ *scheduled tail window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy observation row, and a nonempty finite family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic neighbourhood witnesses;*
2. *R is a refinement ledger whose row for two displayed base packets names a third displayed tail packet refining both requested packets through a smaller dyadic tolerance window;*
3. *M is a* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning to each requested precision a finite tail packet after which all displayed dyadic neighbourhood witnesses around the regular Cauchy observations are within the requested radius;*
4. *L is the ledger exposing the base-packet rows, the refinement rows, the modulus rows, the componentwise* hsame *comparisons, the dyadic tolerance inclusions, and the displayed* Cont *routes used by refinements and precision reads;*
5. *P is the* Pkg *provenance row naming the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBall}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *certificate boundaries consumed by the packet.*

The carrier accepts only these finite rows. Classification is componentwise transport of B, R, M, L, and P through the displayed hsame *and* Cont *data.*

Theorem 333.2 (CauchyFilter directed refinement obligation)**.** *For every accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier $F = (B, R, M, L, P)$ and every two tail packets displayed in B, the refinement row in R is a displayed* Cont *and* hsame *obligation: it names a finite tail packet in the same carrier whose dyadic neighbourhood rows refine both inputs through the continuation routes recorded by L.*

Proof. Project the accepted carrier through Definition 333.1. The row R is stored with the tail packets it refines, so the two input packets and the refined packet are all visible finite BHist rows over the same regular Cauchy observation surface.

Read the continuation routes from L. Each route records how the refined packet restricts to the requested input packet, and the accompanying hsame rows compare the displayed dyadic neighbourhood witnesses componentwise.

Thus directedness is not a global existence assertion. It is exactly the finite refinement row already stored in the carrier, together with its displayed Cont and hsame witnesses. □

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterCarrier directed refinement endpoint surface`

Theorem 333.3 (CauchyFilter Cauchy modulus obligation)**.** *Let $F = (B, R, M, L, P)$ be an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier. The modulus row M makes F stable under smaller dyadic tail windows: for each displayed precision request, M selects a finite tail packet, and every later refinement row in R carries regular Cauchy observations whose* $\mathsf{DyadicRatCore}^{\uparrow}$ *neighbourhood ledger is included in the requested radius.*

Proof. Fix a precision request and project the selected row of M. It names a tail packet in B and records the requested dyadic radius through the $\mathsf{CauchyModulus}^{\uparrow}$ boundary.

Now take any later refinement row displayed by R. By Definition 333.2, the later packet refines the selected packet through explicit Cont routes and componentwise hsame comparisons.

The dyadic distance bound is read from the same finite ledger L. Since the bound is attached to the selected precision and transported along the displayed refinement rows, the carrier supplies a constructive Cauchy witness using only its finite tail, modulus, regular observation, and dyadic neighbourhood rows. □

Theorem 333.4 (CauchyFilter metastability consumer route)**.** *Every accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier consumed by a* $\mathsf{CauchyMetastability}^{\uparrow}$ *reader must expose the finite filterbase row, directed refinement row, Cauchy modulus row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before any metastable finite-tail verdict is read. The route exports no quotient filter, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier through Definition 333.1. This exposes the finite filterbase row together with the stream schedules, regular rational observations, dyadic neighbourhood witnesses, refinement ledger, modulus row, evidence ledger, and provenance row.

Next apply Definition 333.2 and Definition 333.3. The first theorem supplies the directed refinement row and its displayed transport, while the second supplies the precision-to-tail modulus and tolerance accounting needed by a finite-tail request.

Finally route the consumer through the finite-tail stability packet of chapter 2093. A $\mathsf{CauchyMetastability}^{\uparrow}$ reader can inspect only the displayed request, window, tolerance, readback, transport, replay, provenance, and local naming rows, so the CauchyFilter input must present exactly the listed finite rows before the metastable verdict is available. □

Theorem 333.5 (CauchyFilter NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyFilter}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the finite-filterbase carrier of Definition 333.1;*
2. *classifier transport: accepted carriers compare by componentwise* hsame *transport of the tail packets, regular observation rows, dyadic neighbourhood witnesses, refinement ledgers, modulus rows, evidence ledgers, and provenance rows, with reads routed through displayed* Cont *data;*

3. *directedness: the displayed refinement row of Definition 333.2;*
4. *Cauchy modulus: the precision-to-base selection and dyadic-radius readback of Definition 333.3;*
5. *downstream handoff:* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers read only the tail packets, stream schedules, regular Cauchy observations, dyadic neighbourhood witnesses, real-ball bounds, refinement rows, modulus rows, ledger rows, and provenance rows of the same finite packet.*

These five rows form the seed obligation for naming completion-facing Cauchy filterbase data over displayed histories.

Proof. The carrier row is Definition 333.1. It supplies the finite tail packets, regular observation rows, dyadic neighbourhood witnesses, refinement ledger, Cauchy-modulus row, evidence ledger, and provenance row.

Classifier transport is componentwise. Transporting the five carrier coordinates through their displayed hsame rows and following the recorded Cont routes repacks the same finite-filterbase shape.

Directedness and the Cauchy property are the two distinguished projections of the carrier. The first projection is Definition 333.2; the second is Definition 333.3.

For downstream readback, a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer can cite only the rows named by P and exposed by L. Those rows are exactly the base packets, stream schedules, regular Cauchy observations, dyadic neighbourhood witnesses, real-ball bounds, refinement rows, modulus rows, ledger rows, and provenance rows of the carrier, so the surface is exhausted by the five listed obligations. □

Theorem 333.6 (CauchyFilter carrier transport obligation)**.** *Let* $F = (B, R, M, L, P)$ *and* $F' = (B', R', M', L', P')$ *be displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carriers. Suppose the evidence ledger exhibits componentwise* hsame *rows for* B *with* B'*,* R *with* R'*,* M *with* M'*,* L *with* L'*, and* P *with* P'*, and suppose every read used by a refinement or precision request is routed through the displayed* Cont *rows. Then classifier transport preserves the accepted carrier tuple:* F *is accepted exactly when the transported tuple* F' *is accepted, and all accepted reads remain inside the displayed finite rows* (B', R', M', L', P')*.*

Proof. Unpack the finite carrier of Definition 333.1. Acceptance is expressed only through the five displayed coordinates B, R, M, L, and P.

Transport is componentwise. The tail packets, regular observation rows, dyadic neighbourhood witnesses, refinement ledger, modulus row, evidence ledger, and provenance row are compared by the named hsame witnesses, while each refinement or precision read follows a displayed Cont route recorded in the ledger.

Apply Definition 333.5 to identify the classifier row as the transport row of the local $\mathsf{NameCert}$ surface. The transported carrier therefore has exactly the same finite-filterbase shape, with rows read from (B', R', M', L', P').

No host filter, selected limit, quotient real, or ambient completeness object is introduced by this transport. The construction moves only the displayed finite rows and their recorded hsame and Cont witnesses. □

Theorem 333.7 (CauchyFilter refinement directedness package)**.** *For an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier* $F = (B, R, M, L, P)$*, the directedness obligation is the finite package consisting of the two input tail packets in* B*, the selected third tail packet named by the row of* R*, the two displayed* Cont *routes from that selected packet to the inputs, the componentwise* hsame *witnesses for the*

transported dyadic neighbourhood rows, and the modulus row of M that reads the selected packet at every requested precision.

Proof. Choose two displayed tail packets in B. By Definition 333.2, the corresponding row of R names a third displayed tail packet together with the continuation routes and componentwise comparison rows needed to refine both inputs.

The selected packet is still a tail packet of the same finite carrier, so the modulus row can be read without leaving the displayed surface. Apply Definition 333.3 to attach the requested precision row and its dyadic-radius ledger to the same refinement package.

The resulting object is a directedness package, not an ambient filter principle. Every component is one of the rows already listed in Definition 333.1 and exposed by Definition 333.5. □

Theorem 333.8 (CauchyFilter Real and RegSeqRat handoff non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier* $F = (B, R, M, L, P)$ *reads only the displayed tail packets, the* $\mathsf{StreamName}^{\uparrow}$ *schedules, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular observation rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *neighbourhood rows, the* $\mathsf{RealBall}^{\uparrow}$ *bounds, the refinement rows, the Cauchy-modulus rows, the ledger rows, and the* Pkg *provenance rows named by* L *and* P*. The handoff does not read a host filter, a selected limit, a quotient real, or an ambient completeness principle.*

Proof. Project the downstream handoff row from Definition 333.5. It lists the rows available to $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers: tail packets, stream schedules, regular Cauchy observations, dyadic neighbourhood rows, real-ball bounds, refinement rows, modulus rows, ledger rows, and provenance rows.

The carrier definition fixes where these rows live. By Definition 333.1, the base packets are in B, refinement data is in R, precision data is in M, read evidence is in L, and downstream provenance is in P.

Directed refinement and precision readback do not add further carrier data. They are the displayed rows supplied by Definition 333.2 and Definition 333.3.

Thus the Real and regular rational sequence handoff is closed under the named finite rows. Since no step constructs or selects an external filter limit, the handoff cannot escape through host filters, selected limits, quotient reals, or ambient completeness. □

Theorem 333.9 (CauchyFilter L10 dependency handoff lattice)**.** *Every accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff is a horizontal L10 dependency lattice for the real-completion surface. Its tail packets are read through chapter 126, its dyadic neighbourhood witnesses through chapter 291, its regular readbacks through chapter 301, and its final seal through chapter 38. The lattice is exhausted by the finite carrier rows, directed refinement rows, Cauchy-modulus rows, ledger evidence, and provenance rows of the accepted carrier; it contains no host filter, selected limit, quotient real, or ambient completeness datum.*

Proof. Start with an accepted carrier $F = (B, R, M, L, P)$. The carrier definition Definition 333.1 places every tail packet in the displayed finite row B, and each such packet carries its BHist evidence, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ regular observation row, dyadic neighbourhood rows, real-ball bounds, ledger reads, and downstream provenance. Hence the horizontal tail layer of the handoff is read through the StreamName and DyadicRatCore concrete-instance chapters cited in the statement.

The middle layer is forced by the two obligations of the accepted carrier. Apply Definition 333.2 to read a displayed refinement row of R whenever two tail packets are compared, and apply Definition 333.3 to read the displayed precision row of M together with its retained dyadic-radius

ledger. These rows are still entries of B, R, M, and L, so they add refinement and Cauchy control without introducing a new completion object.

Finally apply Definition 333.8. It identifies the regular-sequence and real handoff boundary as exactly the displayed tail, stream, regular observation, dyadic, real-ball, refinement, modulus, ledger, and Pkg provenance rows. Therefore the RegSeqRat readback layer and the Real sealing layer are the cited sibling concrete-instance dependencies, and the whole L10 handoff is closed before any host filter, selected limit, quotient real, or ambient completeness principle can be read. □

Theorem 333.10 (CauchyFilter Real-RegSeq completion scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *completion-facing consumer of an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet factors through the displayed finite* FilterBase *row, directed refinement obligation, Cauchy modulus row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RealBall}^{\uparrow}$ *bounds,* Pkg *provenance, componentwise* hsame *transport, displayed* Cont *replay, and local* NameCert *rows of that packet. The scope contains no quotient filters, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the finite-filterbase carrier of Definition 333.1. It places the base packets, refinement witnesses, precision rows, ledger evidence, and provenance in the displayed coordinates B, R, M, L, P.

Apply the directed refinement and Cauchy-modulus obligations of Definition 333.2 and Definition 333.3. These obligations read the finite filter-base row, a refinement row for each comparison, and the modulus row with its dyadic-radius ledger.

Next use the Real and RegSeqRat handoff non-escape theorem Definition 333.8. It identifies the downstream completion-facing reads as the displayed tail packets, $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{RegSeqRat}^{\uparrow}$ regular rows, $\mathsf{RealBall}^{\uparrow}$ bounds, ledger rows, and Pkg provenance already carried by B, R, M, L, P.

Finally use the L10 dependency handoff lattice of Definition 333.9. The lattice routes the same rows through StreamName, DyadicRatCore, RegSeqRat, and Real consumers before any completion-facing seal is read. Since no ingredient adds quotient filters, selected limits, countable choice, host equality, or ambient completeness, the displayed rows exhaust the completion scope. □

Theorem 333.11 (CauchyFilter uniform-space completion handoff)**.** *Let* $F = (B, R, M, L, P)$ *be an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carrier whose provenance row exposes the same finite entourage and filterbase surface consumed by* $\mathsf{UniformSpace}^{\uparrow}$*. Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read admitted from* F *factors first through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *neighbourhood witnesses,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{RealBall}^{\uparrow}$ *bounds, and the finite* $\mathsf{UniformSpace}^{\uparrow}$ *entourage rows before any completion-facing seal is available. Consequently the handoff described in chapter 339 and consumed by chapter 95 and chapter 57 receives only the finite ledger displayed by the Cauchy filterbase carrier, not a selected limit, quotient stream, host filter, or ambient completeness object.*

Proof. Project the carrier $F = (B, R, M, L, P)$. The base row B supplies the displayed tail packets, R supplies directed refinement, M supplies Cauchy precision, L supplies ledger reads, and P supplies provenance. By Definition 333.9, these rows already identify the StreamName windows, DyadicRatCore neighbourhood witnesses, RegSeqRat readback rows, RealBall bounds, ledger evidence, and real-facing provenance.

Next route the finite filterbase as uniform entourage data. Apply Definition 339.6: the UniformSpaceUp interface exposes only finite entourage rows, diagonal rows, refinement rows, symmetry rows, composition rows, endpoint hsame transport, and Pkg provenance. Since the provenance

row of F exposes precisely this surface, the CauchyFilterUp read may be consumed by the UniformSpaceUp handoff without adding a further carrier coordinate.

The completion-facing consumer therefore receives the displayed finite ledger in the stated order: stream windows, dyadic observations, regular readback, real-ball bounds, and uniform entourage rows. None of the preceding projections constructs a chosen limit, quotient stream equality, host filter, or ambient completeness witness, so no such object can be read before the completion-facing seal. □

Definition 333.12 (CauchyFilter finite subfilterbase restriction datum)**.** *Let $F = (B, R, M, L, P)$ be an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carrier in the sense of Definition 333.1. A finite subfilterbase restriction datum for F is a displayed finite sublist $B' \subseteq B$ together with two preservation rows:*

1. *refinement preservation: whenever b_0 and b_1 are displayed in B' and the row of R for (b_0, b_1) names a refined tail packet b_2, the packet b_2 is displayed in B' and the corresponding* Cont *and* hsame *witnesses in L are retained in the restricted ledger;*
2. *modulus preservation: for every precision request admitted by the carrier, the tail packet selected by the row M is displayed in B', and the associated dyadic-radius readback row of L is retained in the restricted ledger.*

The restricted packet is

$$F|_{B'} = (B', R|_{B'}, M|_{B'}, L|_{B'}, P),$$

where $R|_{B'}$ keeps exactly the refinement rows whose two inputs lie in B', $M|_{B'}$ keeps the same precision-to-base selections, $L|_{B'}$ keeps only the ledger rows used by those tail, regular observation, dyadic neighbourhood, refinement, modulus, Cont*, and* hsame *reads, and P is the unchanged downstream provenance row.*

Theorem 333.13 (CauchyFilter finite subfilterbase restriction)**.** *Let $F = (B, R, M, L, P)$ be an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carrier. If $B' \subseteq B$ is a finite subfilterbase restriction datum for F in the sense of Definition 333.12, then the restricted packet*

$$F|_{B'} = (B', R|_{B'}, M|_{B'}, L|_{B'}, P)$$

is an accepted $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase carrier. Its downstream handoff boundary is the same finite* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealBall}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *provenance boundary carried by F.*

Proof. The carrier rows are finite because B' is a displayed finite sublist of the finite tail row B, while $R|_{B'}$, $M|_{B'}$, and $L|_{B'}$ are obtained by retaining displayed rows from the finite ledgers R, M, and L of Definition 333.1. Each tail packet in B' is already a tail packet of B, hence it carries the same BHist row, $\mathsf{StreamName}^{\uparrow}$ scheduled window, $\mathsf{RegSeqRat}^{\uparrow}$ regular observation row, and nonempty finite family of $\mathsf{DyadicRatCore}^{\uparrow}$ neighbourhood rows that it carried in F.

Directedness is inherited row by row. Take two tail packets displayed in B'. In the accepted carrier F, Definition 333.2 reads the corresponding row of R as a displayed refinement packet together with its Cont routes and componentwise hsame witnesses. Refinement preservation in the restriction datum says that this refined packet is again displayed in B' and that the required continuation and comparison witnesses remain in $L|_{B'}$. Therefore $R|_{B'}$ supplies exactly the directed refinement obligation required of the restricted carrier.

The Cauchy modulus row is also inherited without changing the selected bases. For a displayed precision request, Definition 333.3 reads the row of M in F as a selected tail packet together with

the dyadic radius ledger controlling every later displayed refinement. Modulus preservation says that the selected packet lies in B' and that the associated radius readback row remains in $L|_{B'}$. If a later refinement is read inside $R|_{B'}$, it is one of the refinement rows of R, so the same dyadic bound from L applies to it, and the retained row in $L|_{B'}$ records that bound. Thus $M|_{B'}$ is a constructive Cauchy modulus for the restricted packet.

It remains to identify the handoff boundary. The restriction does not replace P; it only removes base, refinement, modulus, and ledger rows not used by B'. Hence a downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer can still read exactly the provenance categories listed in Definition 333.5: tail packets, stream schedules, regular Cauchy observations, dyadic neighbourhood rows, real-ball bounds, refinement rows, modulus rows, ledger rows, and the same Pkg provenance boundary. Since every retained row is a displayed row of the accepted carrier and the two preservation conditions keep all refinement and modulus rows needed for the CauchyFilter obligations, $F|_{B'}$ repacks as an accepted $\mathsf{CauchyFilter}^{\uparrow}$ carrier with the same finite downstream handoff boundary. □

Chapter 334

A Concrete Naming Certificate for $\mathsf{BaireSpace}^{\uparrow}$

The $\mathsf{BaireSpace}^{\uparrow}$ interface names Brouwer-style Baire space as a certificate for natural-number-indexed streams whose observations are displayed finite unary histories. The BEDC surface does not treat infinite sequences as primitive histories. It records finite prefix windows, a prefix-cylinder classifier, stream-name scheduling rows, and package provenance that can be read by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The schedule row s is consumed from chapter 126; $\mathsf{Nat}^{\uparrow}$ supplies the unary index rows, and the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ boundaries are the downstream consumer surfaces for the displayed finite prefix windows.

Definition 334.1 (BaireSpace stream carrier)**.** *A* $\mathsf{BaireSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (s, w, c, l, p)$$

with the following visible rows:

1. *s is a* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose nth displayed request returns a finite* $\mathsf{UnaryHistory}$ *answer row;*
2. *w is a finite prefix-window ledger listing the requested natural indices and the unary answer rows displayed for those indices;*
3. *c is the prefix-cylinder classifier row: two packets classify together over a displayed window exactly when their listed unary answer rows are paired componentwise by* hsame *through the same scheduled indices;*
4. *l is the local ledger exposing the* BHist *packet, the* $\mathsf{UnaryHistory}$ *answer rows, the finite index window, the componentwise* hsame *comparisons, and the* Cont *routes used by prefix restriction;*
5. *p is the* Pkg *provenance row naming the* $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *certificate boundaries available to consumers.*

The carrier predicate is $\mathsf{BairePrefix}(s, w, c, l, p)$*. It accepts only finite prefix displays of a natural-number-indexed stream; every public read is routed through the selected window and the displayed classifier row.*

Lean4 ✓ `BEDC.Derived.BaireSpaceUp.TasteGate.BaireSpaceTasteGate single carrier alignment`

Theorem 334.2 (BaireSpace prefix-cylinder stability)**.** *Let $B = (s, w, c, l, p)$ be an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *carrier. If B' is obtained from B by transporting the stream schedule, every displayed*

unary answer in a finite prefix window, the classifier row, the local ledger, and the provenance row by the hsame *and* Cont *data recorded in l, then B' is again an accepted carrier. The prefix-cylinder classifier for the transported window reads exactly the transported unary rows and no tail beyond that window.*

Lean4 ✓ `BEDC.Derived.BaireSpaceUp.BaireSpacePrefixPacket prefix cylinder stability`

Proof. Project the carrier of Definition 334.1. The only data visible to the classifier are the stream schedule, the finite index window, and the unary answer rows listed by that window.

Transport each listed row through the displayed hsame comparisons in the ledger. The Cont routes in the same ledger restrict the stream readback to the transported prefix, so the repacked tuple has the same carrier shape.

The classifier row is a cylinder classifier. It compares exactly the displayed prefix entries, and no coordinate in the carrier stores an infinite tail as a history object. Thus the transported packet is accepted with the transported prefix-cylinder read. □

Theorem 334.3 (BaireSpace prefix-restriction composition)**.** *Let $B = (s, w, c, l, p)$ be an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *carrier. Suppose the ledger l displays a prefix restriction from B to B_1 and then a second prefix restriction from B_1 to B_2, each recorded by the corresponding* hsame *comparisons and* Cont *routes on the stream schedule, finite index window, unary answer rows, classifier row, and provenance row. Then the composite restriction is again an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet.*

Its visible unary rows are exactly the twice-restricted subwindow of w, its classifier is the componentwise hsame *restriction of the original prefix-cylinder classifier c to that subwindow, and its provenance is still the same* Pkg *row p. The composite packet introduces no hidden infinite tail and no host sequence equality.*

Proof. Project the accepted packet B through Definition 334.1. The first displayed restriction is a transport of the schedule, window, unary rows, classifier, ledger, and provenance along rows already recorded in l.

Use Definition 334.2 on this first restriction. The result is an accepted prefix packet B_1 whose classifier reads only the first restricted finite subwindow.

Apply the same stability theorem to the second displayed restriction out of B_1. The Cont routes compose inside the displayed ledger, so the resulting packet reads precisely the subwindow obtained by restricting the original prefix window twice.

It remains to identify the classifier and provenance. The classifier at each listed coordinate is obtained by the componentwise hsame restriction of the original cylinder row c, and the Pkg coordinate is transported as the same provenance row p.

Finally, the ledger coverage statement of Definition 334.7 rules out every consumer read not listed in the finite window. Hence the composite construction cannot expose an infinite tail or an equality of host sequences. □

Lean4 ✓ `BEDC.Derived.BaireSpaceUp.BaireSpacePrefixPacket prefix restriction composition`

Theorem 334.4 (BaireSpace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier: the stream carrier of Definition 334.1;*
2. *classifier: the prefix-cylinder row comparing displayed unary answers componentwise over a finite index window;*
3. *stability: prefix-window transport as in Definition 334.2;*

4. *exactness: a public observation is readable from the certificate iff it is one of the finite unary rows listed by the selected prefix window;*
5. *provenance: downstream reads factor through the displayed* $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *package rows carried by* p.

These rows are the complete seed obligation for naming countable stream space data over BEDC histories.

Proof. The carrier and classifier rows are the first two projections of Definition 334.1. They expose the scheduled stream read, the finite index window, and the unary answer rows over that window.

Stability is supplied by Definition 334.2. Transporting the schedule, window, answer rows, ledger, and provenance repacks the same finite prefix surface.

For exactness, inspect the coordinates of the packet. A public observation can only be read from the prefix ledger w through the classifier row c and the routes in l. The provenance row p names the available consumers but does not add a second observation channel. □

Theorem 334.5 (BaireSpace real-name handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packet reads the stream only through finite prefix windows. A regular rational sequence consumer may read a requested unary index and its scheduled answer row, and a real-name consumer may combine that finite read with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *provenance named by* p. *Neither consumer receives a primitive infinite history, a completed real, or an unlogged tail equality.*

Lean4 ✓ `BEDC.Derived.BaireSpaceUp.BaireSpacePrefixPacket real name handoff transport`

Proof. Project the accepted packet through Definition 334.1. The stream schedule is present only through finite request rows, and the answers are the unary histories displayed in w.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer therefore obtains a finite index request and the corresponding unary answer row. Transport of that read is controlled by the prefix-cylinder classifier and the ledger routes.

A $\mathsf{Real}^{\uparrow}$ consumer can additionally use the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance named by p when interpreting the finite answer as a dyadic or regular sequence component. The packet still exposes only the selected prefix rows, so no completed infinite object is exported. □

Theorem 334.6 (BaireSpace Real stream prefix determinacy)**.** *Let* $B = (s, w, c, l, p)$ *and* $B' = (s', w', c', l', p')$ *be accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packets. Suppose their scheduled finite prefix indices agree and the unary answer rows at each displayed index are paired componentwise by* hsame. *Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the same finite stream-prefix boundary from* B *and* B'.

The determinacy is only for logged finite prefix rows. It does not identify primitive infinite histories, host sequences, completed reals, or unlogged tails.

Proof. Apply Definition 334.2 to transport the componentwise unary answer rows through the common scheduled prefix indices. The transported classifier still reads only the displayed finite window.

Use Definition 334.3 to compare any consumer-requested subwindow of the two packets inside the same restricted prefix. The composed restriction preserves the schedule, answer rows, classifier, ledger, and provenance rows needed by the consumer.

For $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, invoke Definition 334.5. Those consumers receive only the finite prefix requests and unary answer rows, with the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance available to the real-name interpretation.

Finally apply Definition 334.7. Since the ledger covers exactly the displayed stream packet, unary answer rows, prefix window, classifier, hsame comparisons, Cont routes, and provenance, there is no row from which an infinite history, host sequence equality, completed real, or unlogged tail could distinguish the two reads. □

Theorem 334.7 (BaireSpace ledger coverage)**.** *For an accepted carrier $B = (s, w, c, l, p)$, the ledger l covers every public row of the* $\mathsf{BaireSpace}^{\uparrow}$ *packet: the* BHist *stream packet, the* UnaryHistory *answer rows, the finite prefix window, the prefix-cylinder classifier, the componentwise* hsame *comparisons, the prefix-restriction* Cont *routes, and the package provenance. The ledger exports no apartness principle, choice functional, host sequence equality, or real-completeness assertion.*

Proof. Read the packet coordinates one by one. The stream schedule s, prefix window w, classifier c, and provenance p are all named fields of the carrier, and the ledger l records the comparison and restriction routes among them.

Any public row must be read from one of those fields. Prefix restriction uses the displayed Cont route, while equality of observations uses the displayed componentwise hsame rows in the cylinder classifier.

The omitted objects would require extra coordinates: an apartness principle, a choice functional, a host equality on full sequences, or a completeness theorem for real names. No such coordinate occurs in the carrier, so the ledger coverage is exactly the listed finite surface. □

Theorem 334.8 (BaireSpace cylinder-tail coverage)**.** *Every finite prefix-cylinder consumer read from an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packet factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule, the prefix-restriction ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and the* $\mathsf{Real}^{\uparrow}$ *handoff boundary. The read exposes no completed infinite sequence, no classical topology, and no choice-selected branch.*

Proof. Read the stream carrier of Definition 334.1. Its public coordinates are the schedule, finite prefix window, prefix-cylinder classifier, ledger, and provenance rows, so a cylinder-tail consumer begins with a finite window named by that carrier.

Use Definition 334.2 to keep the cylinder answer stable under displayed prefix reads, and Definition 334.3 to compose the restriction ledger. These steps route the consumer through the same hsame transport and Cont paths already recorded in the packet.

The real-facing boundary is exactly the finite handoff of Definition 334.5, and Definition 334.7 exhausts the remaining public rows. No carrier coordinate stores a completed sequence, a topological completion, or a choice-selected branch, so such data cannot be exported by a prefix-cylinder read. □

Theorem 334.9 (BaireSpace choice-spread continuity route)**.** *Every finite-prefix consumer of an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packet that feeds a* $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, *or* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *read must factor through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{UnaryHistory},\ \textit{prefix-cylinder classifier},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}.$$

The route reaches the $\mathsf{ChoiceSequence}^{\uparrow}$ *answer rows,* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix admission, and* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *finite-modulus boundary only after the stream schedule, unary prefix window, prefix-cylinder classifier, and* Pkg *provenance have been displayed. It exports no completed Baire function, no host sequence equality, no choice-selected branch, no external topology, and no real-completeness datum before any* $\mathsf{Real}^{\uparrow}$*-facing handoff.*

Proof. Start with Definition 334.8. It forces every tail-sensitive read of a $\mathsf{BaireSpace}^{\uparrow}$ prefix packet through the displayed stream schedule, finite prefix ledger, hsame transport, Cont routes, package provenance, and real-facing boundary.

The finite classifier is pinned by Definition 334.10. Thus a consumer cannot inspect a choice-sequence answer, a spread-law extension row, or a continuity modulus before it has displayed the unary prefix window and the prefix-cylinder classifier.

For the choice-sequence side, use chapter 1482; for spread admission, use chapter 1532; for the continuity consumer, use chapter 3465. Each sibling chapter consumes only finite prefix rows, so each sits downstream of the same $\mathsf{StreamName}^{\uparrow}$ schedule and Pkg provenance rather than upstream of them.

Finally apply Definition 334.5 and Definition 334.7. These exhaust the rational and real-facing boundary and the local ledger rows, leaving no coordinate for a completed host stream, a selected branch, host equality, topology, or real completeness. □

Theorem 334.10 (BaireSpace obligation window classifier)**.** *Every accepted* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet supplies the obligation surface consisting of the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, displayed* $\mathsf{UnaryHistory}$ *answers, prefix-cylinder classifier, componentwise* hsame*, prefix-restriction* Cont*,* Pkg *provenance, and consumer handoff rows for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

The classifier is exhausted by finite windows. It excludes host sequence equality, choice-selected branches, and classical topology.

Proof. Project the accepted prefix packet through Definition 334.1. The carrier provides the finite schedule, index window, displayed unary answers, classifier row, ledger, and package provenance.

Use Definition 334.2 and Definition 334.3 to keep every classifier read inside finite prefix windows under the displayed hsame and Cont rows.

Apply Definition 334.4 for the local naming surface, Definition 334.5 for the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer rows, and Definition 334.7 for exhaustion of the public ledger.

Finally, Definition 334.8 rules out any tail read not already listed in the finite cylinder packet. Thus the obligation surface is exactly the finite window classifier, with no host sequence equality, choice-selected branch, or classical topology. □

Theorem 334.11 (BaireSpace finite cylinder stream boundary)**.** *Every accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packet consumed by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *is exhausted by the finite schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, displayed* $\mathsf{UnaryHistory}$ *answers, prefix-cylinder classifier, componentwise* hsame *rows,* Cont *restriction routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ *ledger. These rows are the exact finite-cylinder boundary for stream-name, regular-sequence, and real-facing consumers.*

The boundary excludes host sequence equality, infinite tails, classical topology, choice-selected branches, and completed reals.

Proof. Project the accepted packet through Definition 334.1. The carrier exposes the finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{Nat}^{\uparrow}$ index rows, displayed unary answers, prefix-cylinder classifier, transport routes, package provenance, and local certificate ledger.

Apply Definition 334.5 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer rows reached from that schedule. The handoff uses the same finite prefix-cylinder coordinates and introduces no completed real or host stream equality.

Use Definition 334.7 for the remaining package and certificate rows, and use Definition 334.8 to rule out any read from an undisplayed infinite tail. The obligation window classifier of Definition 334.10 then pins the classifier surface to the same finite window.

Thus every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer read is one of the projected finite-cylinder rows. Since those rows contain no host sequence equality, infinite tail object, classical topological principle, choice-selected branch, or completed real, none can appear at the boundary. □

Theorem 334.12 (BaireSpace scoped finite-consumer boundary)**.** *Every* $\mathsf{BaireSpace}^{\uparrow}$ *consumer read factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{Nat}^{\uparrow}$ *index rows, displayed* UnaryHistory *answers, the prefix-cylinder classifier, componentwise* hsame*, prefix-restriction* Cont*,* Pkg *provenance, and the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows.*

This finite boundary exhausts the scoped public surface. It exposes no host sequence equality, no choice-selected branch, no completed infinite sequence, and no classical topology.

Proof. Project the consumer read through the stream carrier of Definition 334.1. The only displayed coordinates are the schedule, finite index window, unary answer rows, classifier, transport ledger, package provenance, and downstream handoff rows.

Apply Definition 334.2 to keep the prefix-cylinder answer stable under displayed finite reads. Then use Definition 334.3 to compose the prefix-restriction Cont rows along the same componentwise hsame route.

The real-facing and rational-facing consumer rows are supplied by Definition 334.5. The remaining public coordinates are exhausted by Definition 334.7 and by the obligation window classifier of Definition 334.10.

Therefore any scoped consumer read is already one of the finite prefix-cylinder packet reads listed above. Since none of the cited rows constructs a host sequence equality, a choice branch, a completed infinite sequence, or a classical topological object, none is available at the scoped boundary. □

Theorem 334.13 (BaireSpace public prefix-cylinder certificate)**.** *The stream carrier of Definition 334.1, the obligation window classifier of Definition 334.10, and the finite-consumer boundary of Definition 334.12 export the public* $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ *surface for the same prefix-cylinder packet.*

That surface consists exactly of the finite $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{Nat}^{\uparrow}$ *index rows, displayed* UnaryHistory *answers, the prefix-cylinder classifier, componentwise* hsame*, prefix-restriction* Cont*,* Pkg *provenance, and the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows. It excludes host sequence equality, choice-selected branches, completed infinite sequences, classical topology, apartness on Baire space, and real completeness.*

Proof. Start with an accepted prefix-cylinder packet and project it through Definition 334.1. This gives the finite schedule, index rows, displayed unary answers, classifier coordinate, transport routes, and package provenance that are available to the public name certificate.

The classifier row is the one identified by Definition 334.10. Its finite-window description is stable under the componentwise hsame and prefix-restriction Cont rows already used by the scoped boundary.

The downstream rows are exactly the rows supplied by Definition 334.5. The remaining ledger coordinates are exhausted by Definition 334.7, while Definition 334.8 prevents a hidden tail read from entering the packet.

Finally apply Definition 334.12. It states that every consumer read has already factored through this finite prefix-cylinder list, so repackaging the same rows as the public $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ surface adds no host equality, branch selector, completed sequence, topological principle, Baire-space apartness, or real completeness field. □

Theorem 334.14 (BaireSpace prefix-stream bridge)**.** *The public prefix-cylinder certificate of Definition 334.13 has a conservative stream-name reading: every accepted prefix-cylinder read factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answer rows, componentwise* hsame*, prefix-restriction* Cont*,* Pkg *provenance, and the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows. Reading the stream-name packet back recovers exactly those rows.*

The bridge adds no host sequence equality, choice-selected branch, completed infinite sequence, classical topology, apartness on Baire space, or real completeness.

Lean4 ✓ `BEDC.Derived.BaireSpaceUp.BaireSpaceUp StdBridge`

Proof. Project the public certificate through Definition 334.13. Its visible rows are already the finite schedule, finite index window, unary answers, classifier, componentwise transport, prefix-restriction route, package provenance, and downstream handoff rows.

Use Definition 334.12 to identify every standard finite-prefix read with one of these displayed packet rows. This prevents a read of an infinite host sequence or a choice-selected branch from being inserted into the prefix-cylinder interface.

Apply Definition 334.8. The finite cylinder tail is already accounted for by the displayed prefix window, so no completed infinite sequence or hidden tail coordinate enters the readback.

For downstream consumers, apply Definition 334.5. The rational and real-facing handoffs remain finite prefix reads and do not require real completeness, classical topology, or a Baire-space apartness principle.

Finally use Definition 334.7 to exhaust the remaining ledger rows. Reading the standard finite prefix-cylinder packet back into $\mathsf{BaireSpace}^{\uparrow}$ therefore recovers exactly the displayed carrier and handoff surface, with no additional host equality, branch selector, completed sequence, topological object, apartness row, or real-completeness row. □

Theorem 334.15 (BaireSpace finite prefix-cylinder coverage)**.** *Every finite stream-name consumer read of a certified* $\mathsf{BaireSpace}^{\uparrow}$ *packet factors through a displayed prefix-cylinder row.*

Compatible nested prefix cylinders share a common finite prefix transport through the same BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ *rows.*

Proof. Start with Definition 334.2. It keeps each accepted finite read inside the displayed prefix-cylinder classifier, finite $\mathsf{Nat}^{\uparrow}$ index rows, unary answer rows, and stream schedule.

For nested cylinders, apply Definition 334.3. The composed restriction route supplies the common finite prefix transport through the same hsame, Cont, package, and local certificate rows.

Use Definition 334.5 for rational and real-facing consumers. Those reads remain finite stream-name handoffs rather than completed sequences or real-completeness assumptions.

Finally combine Definition 334.12 with Definition 334.14. The scoped boundary exhausts the finite consumer rows, and the bridge identifies the same rows with the stream-name presentation, yielding the stated finite prefix-cylinder coverage. □

Theorem 334.16 (BaireSpace StreamName prefix sieve)**.** *Every finite prefix-cylinder read consumed by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows,* $\mathsf{UnaryHistory}$ *answer rows, prefix-cylinder classifier, componentwise* hsame *rows,* Cont *restriction routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$ *ledger.*

The sieve exports no choice-selected branch, host sequence equality, completed sequence, or real-completeness row.

Proof. Begin with Definition 334.2. It keeps each finite observation inside the displayed prefix-cylinder classifier and the finite $\mathsf{Nat}^{\uparrow}$ index and unary answer rows.

Use Definition 334.3 to compose the restriction routes. The resulting read still travels through the same $\mathsf{StreamName}^{\uparrow}$ schedule, componentwise hsame rows, Cont routes, package provenance, and local naming ledger.

For rational and real consumers, apply Definition 334.5. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows are therefore finite prefix reads of the displayed stream schedule rather than completed host sequences.

Finally invoke Definition 334.7. It exhausts the remaining certificate rows, so no choice-selected branch, host sequence equality, completed sequence, or real-completeness row is exported. □

Theorem 334.17 (BaireSpace finite-prefix consumer completeness)**.** *Every accepted finite consumer of the public prefix-cylinder bridge factors through one displayed finite prefix window. The window consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answer rows, the prefix-cylinder classifier, componentwise* hsame *rows, prefix-restriction* Cont *routes,* Pkg *provenance, and the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows.*

This factorization exports no host sequence equality, choice branch, completed infinite sequence, apartness row, external topology, or real completeness row.

Proof. Start with the public prefix-cylinder certificate of Definition 334.13. Its exposed rows are already the finite schedule, prefix index window, displayed unary answers, classifier coordinate, componentwise transport, prefix-restriction route, package provenance, and downstream handoff surface.

Read the same packet through Definition 334.14. This identifies every bridge consumer with the stream-name presentation over the same finite prefix window, so the consumer has no separate access to a host sequence or a choice-selected branch.

The remaining coverage is ledger-local. Apply Definition 334.7 to exhaust the certificate rows and Definition 334.8 to keep tail information inside the displayed finite cylinder. No completed infinite sequence or external topological row is produced.

Finally apply Definition 334.5. The $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ consumers therefore receive only the listed handoff rows, not real completeness, apartness on Baire space, or any additional host equality principle. Hence every accepted finite consumer factors through the displayed finite prefix window. □

Theorem 334.18 (BaireSpace prefix intersection normal form)**.** *Two compatible finite prefix-cylinder consumers of an accepted* $\mathsf{BaireSpace}^{\uparrow}$ *packet, whose requested* $\mathsf{Nat}^{\uparrow}$ *prefix windows overlap through displayed* hsame *and* Cont *routes, factor through one displayed finite prefix window. The normal-form window carries the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows,* $\mathsf{UnaryHistory}$ *answer rows, the prefix-cylinder classifier, componentwise* hsame *rows,* Cont *restrictions,* Pkg *provenance, and the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows.*

The intersection read exports no host sequence equality, choice branch, completed infinite sequence, apartness row, external topology, or real-completeness row.

Proof. Use Definition 334.2 on each consumer. Both reads remain inside finite prefix-cylinder rows whose schedule, requested $\mathsf{Nat}^{\uparrow}$ indices, unary answers, and prefix classifier are displayed in the accepted packet.

Apply Definition 334.3 to the overlap routes. Compatibility through hsame and Cont lets the two requested windows compose to one finite window carrying the common componentwise transport, restriction provenance, and Pkg row.

Read the resulting window through Definition 334.14 and Definition 334.16. These identify the intersection as a $\mathsf{StreamName}^{\uparrow}$ schedule read and exhaust the finite prefix coordinates before any completed host sequence or choice-selected branch can be formed.

Finally invoke Definition 334.17. The $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ consumers receive only the listed handoff rows, so the compatible intersection has the stated finite normal form and exports none of the excluded topology, apartness, or real-completeness data. □

Theorem 334.19 (BaireSpace prefix-cylinder separation)**.** *Let two accepted* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packets have a displayed common finite prefix window. If, at the first listed mismatch in that window, one displayed* UnaryHistory *answer row is not related by the packet classifier to the corresponding row of the other packet, then the two packets do not classify together through that prefix-cylinder surface.*

Later tail rows, host sequence equality, completed infinite sequences, external topology, apartness data, and choice-selected branches are irrelevant to this separation claim.

Proof. Project both packets to their finite prefix-cylinder carriers. The projected rows are the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{Nat}^{\uparrow}$ prefix indices, displayed UnaryHistory answers, the prefix classifier, componentwise hsame transport, prefix-restriction Cont routes, package provenance, and the downstream handoff rows.

Use Definition 334.2 on both packets. Every classifier read that can affect the prefix-cylinder comparison is restricted to the displayed unary answer rows inside the common finite window.

Apply Definition 334.3 and Definition 334.18. These put the two packets on one displayed prefix window, with the same visible transport and restriction provenance, before any later tail coordinate is consulted.

At the first listed mismatch, the classifier row required for joint classification is absent by hypothesis. The packet carrier has no hidden host sequence equality, classical topology datum, or tail row that can replace this displayed classifier coordinate. Hence the two packets cannot classify together through the prefix-cylinder surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BaireSpace}^{\uparrow}$ packet is generated by a $\mathsf{StreamName}^{\uparrow}$ schedule, finite $\mathsf{Nat}^{\uparrow}$ prefix-window requests, displayed UnaryHistory answer rows, a prefix-cylinder classifier, componentwise hsame comparisons, prefix-restriction Cont routes, and Pkg provenance for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ consumers.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 334.13, Definition 334.14, and Definition 334.16 close the public prefix-cylinder certificate, stream-name readback, and finite prefix-cylinder consumer completeness surface through the $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{Nat}^{\uparrow}$ index rows, displayed unary answers, prefix-cylinder classifier, componentwise hsame, prefix-restriction Cont, Pkg provenance, and $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ handoff rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BaireSpace}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.BaireSpaceUp.BaireSpacePrefixPacket\_real\_name\_handoff\_transport`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close choice sequences as completed host functions, apartness on Baire space, continuity theorems, real completeness, or an external topology of Baire space.

Upgrade path: No further upgrade – chapter at top of formal axis.

The $\mathsf{DyadicApproximation}^{\uparrow}$ hub reads its terminal L10 route through chapter 291, chapter 126, chapter 301, and chapter 360; the child files below keep that route finite and export no quotient stream, selected host limit, host real equality, solver oracle, ambient completeness principle, or hidden completion object.

Chapter 335

A Concrete Naming Certificate for $\mathsf{DyadicApproximation}^{\uparrow}$

The $\mathsf{DyadicApproximation}^{\uparrow}$ interface names the finite dyadic observation surface that lies between $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the sealing boundary of $\mathsf{Real}^{\uparrow}$. Bishop–Bridges constructive analysis treats rational approximants with explicit precision as the computational face of real-number construction; the BEDC packet below keeps that face as finite BHist rows rather than as quotient reals, hidden limits, or choice-selected representatives.

chapter 291 supplies the endpoint row, chapter 126 supplies the finite observation window, chapter 301 supplies the regular readback surface, and chapter 38 supplies the seal boundary. These are sibling certificate edges consumed by the carrier rows below, not ambient completion assumptions or host equality assumptions.

335.1 Finite dyadic observation packets

Definition 335.1 (DyadicApproximation carrier)**.** *A* $\mathsf{DyadicApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (n, d, w, l, p)$$

with the following visible rows:

1. *n is a unary precision row naming the requested dyadic accuracy;*
2. *d is a* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row whose denominator ledger is readable at the precision named by n;*
3. *w is a certified finite observation window from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, displaying the scheduled stream read that produced the endpoint row;*
4. *l is a ledger tying the finite window to a* $\mathsf{Real}^{\uparrow}$ *seal candidate by recording the regular-sequence source rows, dyadic-radius budget, and continuation route consumed at the real-name boundary;*
5. *p is the* Pkg *provenance naming the consumed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *certificate surfaces.*

The classifier compares two accepted packets at the same precision when the precision rows are related by displayed hsame*, the dyadic endpoint rows are related by the transported* $\mathsf{DyadicRatCore}^{\uparrow}$

classifier, and the finite-window readbacks through $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *are equivalent after transporting their ledgers componentwise. The carrier contains no quotient of regular sequences, no ambient real endpoint, no hidden completion object, and no selected limit.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier semantic name certificate`

335.2 Certificate obligations

Theorem 335.2 (DyadicApproximation finite-window obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the constant dyadic zero row from* $\mathsf{DyadicRatCore}^{\uparrow}$*, read through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule and regular sequence, inhabits the carrier at every unary precision;*
2. *classifier laws: reflexivity, symmetry, and transitivity are obtained by componentwise* hsame *transport of the precision row, dyadic endpoint row, certified finite window, real-seal ledger, and provenance row;*
3. *carrier transport: any displayed* hsame *and* Cont *route preserving* n*,* d*,* w*,* l*, and* p *repacks an accepted carrier as an accepted carrier;*
4. *finite-window exactness: every consumer-visible approximation read factors through exactly the displayed precision row, dyadic endpoint row, finite observation window, seal ledger, and provenance row;*
5. *real-seal handoff: the ledger* l *is the only boundary datum exported to* $\mathsf{Real}^{\uparrow}$*, and it presents the finite regular-sequence source rows needed for a real-name seal candidate.*

These rows define the naming-certificate surface for finite dyadic approximants with explicit precision, following the Bishop–Bridges constructive-analysis reading of rational approximations.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier precision window determinacy`

Proof. For carrier habitation, choose the displayed zero packet of $\mathsf{DyadicRatCore}^{\uparrow}$ at the requested precision and read it through the constant stream schedule. The corresponding $\mathsf{RegSeqRat}^{\uparrow}$ window has the zero endpoint at every displayed index, and the real-seal ledger records the same constant regular source.

For the classifier laws, inspect only the coordinates of Definition 335.1. Reflexivity uses identity hsame rows and identity Cont routes on n, d, w, l, and p. Symmetry reverses those componentwise comparisons. Transitivity composes the same finite rows and then repacks the common dyadic endpoint comparison through $\mathsf{DyadicRatCore}^{\uparrow}$.

Carrier transport follows the same inventory. A transport route preserving the five visible coordinates moves the accepted precision, endpoint, window, ledger, and provenance rows to the target packet, so acceptance is recovered by repacking those rows.

For finite-window exactness, a consumer can project only the rows named in the carrier tuple. The endpoint read is the $\mathsf{DyadicRatCore}^{\uparrow}$ row d, the scheduled observation is w, and the downstream boundary is l with provenance p.

The real-seal handoff is therefore forced: $\mathsf{Real}^{\uparrow}$ receives the regular-sequence source rows exposed by l and no quotient real, hidden completion point, or chosen limit. This is exactly the finite rational-approximant discipline used by the constructive analysis precedent. □

Theorem 335.3 (DyadicApproximation Real seal handoff)**.** *Every accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet* $A = (n, d, w, l, p)$ *exports to a* $\mathsf{Real}^{\uparrow}$ *consumer only the finite regular-sequence source surface recorded by* w *and* l*: the unary precision row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readback, the dyadic-radius budget, the displayed* hsame *and* Cont *routes, and the* Pkg *provenance.*

If two accepted packets have the same precision and equivalent finite-window readback after componentwise hsame *transport, then their* $\mathsf{Real}^{\uparrow}$ *seal candidates receive the same finite source surface. The handoff introduces no quotient real, host real equality, metric-completion point, or choice-selected representative.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier real seal handoff`

Proof. Project the carrier of Definition 335.1. The window row w supplies the scheduled finite observations, and the ledger row l records how those observations are read as a regular-sequence source for the $\mathsf{Real}^{\uparrow}$ boundary.

The exported precision and endpoint rows are already visible as n and d. The dyadic-radius budget and regular-window readback are the displayed coordinates of l, and the routing and provenance rows are supplied by the same ledger together with p.

Now compare two accepted packets with transported equivalent finite-window readback. Componentwise hsame identifies the precision rows, transports the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint comparison, and moves the scheduled $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows to the same readback surface.

Since the $\mathsf{Real}^{\uparrow}$ consumer can inspect only the finite source rows just listed, the two seal candidates receive the same boundary datum. No coordinate remains from which a quotient real, host equality witness, completion point, or selected representative could be produced. □

Theorem 335.4 (DyadicApproximation common precision refinement)**.** *Let* $A = (n, d, w, l, p)$ *and* $A' = (n', d', w', l', p')$ *be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets. Suppose there is a displayed common unary precision row* n_* *with* $\mathsf{hsame}(n, n_*)$ *and* $\mathsf{hsame}(n', n_*)$*, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows over* n_**, and a common finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback window carried by displayed* Cont *routes. Then every* $\mathsf{Real}^{\uparrow}$ *seal consumer comparing the two packets receives one finite source surface: the precision row* n_**, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint comparison, the common observation window, the regular-sequence ledger, the displayed* hsame *and* Cont *routes, and the* Pkg *provenance for both accepted packets.*

The comparison remains a finite dyadic approximation comparison. It does not assert quotient-real equality, host real equality, equality of completion points, or equality of choice-selected representatives.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier common precision refinement`

Proof. Project both accepted packets through Definition 335.1. This exposes exactly the precision rows, dyadic endpoint rows, finite windows, seal ledgers, and provenance rows that a downstream consumer may inspect.

Transport the precision coordinates to the displayed row n_*. Along the same displayed hsame routes, move the endpoint rows into the common $\mathsf{DyadicRatCore}^{\uparrow}$ comparison surface over n_*.

The shared finite window identifies the $\mathsf{StreamName}^{\uparrow}$ schedule and the $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger for both packets. The displayed Cont routes carry the seal ledger rows to that same readback surface, so the regular-sequence source rows and dyadic-radius budget are read through one finite boundary.

Apply Definition 335.3 to A and to A'. Each handoff exports only the projected finite source rows, and after the transports above those rows are the same precision, endpoint-comparison, window, ledger, route, and provenance surface.

Thus the $\mathsf{Real}^{\uparrow}$ boundary sees one finite source surface. Since all exported coordinates come from the carrier projection and the displayed transports, no quotient real, host real equality, completion point, or choice-selected representative is introduced. □

Theorem 335.5 (DyadicApproximation classifier transport)**.** *Let $A = (n, d, w, l, p)$ and $A' = (n', d', w', l', p')$ be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets. Suppose the precision rows n, n', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows d, d', the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readbacks w, w', the* $\mathsf{Real}^{\uparrow}$ *seal ledgers l, l', the displayed* Cont *routes, and the provenance rows p, p' are related by componentwise displayed* hsame *transport along the same carrier coordinates of Definition 335.1.*

Then the transported packet still satisfies the $\mathsf{DyadicApproximation}^{\uparrow}$ *classifier. The resulting comparison is precisely a finite-window dyadic approximation comparison: it compares the transported precision row, transported dyadic endpoint row, transported stream and regular-sequence readback, transported seal ledger, displayed continuation routes, and transported* Pkg *provenance. It does not become quotient-real equality, host-real equality, metric-completion equality, or equality of selected limits.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier classifier transport`

Proof. Project the five coordinates of Definition 335.1. The precision coordinate gives the row along which the comparison is read, the endpoint coordinate gives the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison, the window coordinate gives the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback, the ledger coordinate gives the $\mathsf{Real}^{\uparrow}$ seal boundary, and the provenance coordinate records the consumed certificate surfaces.

Transport these coordinates by the displayed hsame rows. The precision row moves to n', the dyadic endpoint comparison moves to d', and the finite-window readback moves through the same $\mathsf{StreamName}^{\uparrow}$ schedule and $\mathsf{RegSeqRat}^{\uparrow}$ readback surface. The displayed Cont routes are transported with those rows, so no consumer obtains an additional route outside the carrier.

Reuse the endpoint classifier supplied by $\mathsf{DyadicRatCore}^{\uparrow}$ and the finite-window readback described in Definition 335.3. The seal ledger l' records only the transported regular-sequence source rows, dyadic-radius budget, and continuation boundary available to $\mathsf{Real}^{\uparrow}$, while p' records the transported Pkg provenance.

Repack these transported rows into the classifier of the target packet. Carrier exhaustiveness leaves exactly the five projected coordinates, so there is no coordinate from which a completion point, quotient real, selected limit, host-real equality, or quotient-real equality could be introduced. □

Theorem 335.6 (DyadicApproximation precision weakening)**.** *Let $A = (n, d, w, l, p)$ be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet. If a displayed unary row m is no finer than n through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, then A has a coarser accepted read at m. The* $\mathsf{Real}^{\uparrow}$ *seal handoff of this coarser read exposes only the transported precision row m, the transported dyadic endpoint, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the displayed* hsame *and* Cont *routes, and the* Pkg *provenance.*

The weakening rule does not introduce a quotient real, host real equality, completion point, or selected representative.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier precision weakening`

Proof. Start from the accepted packet A and apply Definition 335.3. This projects the only source rows visible to a $\mathsf{Real}^{\uparrow}$ consumer: precision, dyadic endpoint, stream window, regular-sequence readback, seal ledger, routes, and provenance.

Use the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger witnessing that m is no finer than n to transport the precision coordinate to the coarser row. Then apply Definition 335.4 with the common precision row m, reading the original packet and its coarser radius-ledger projection through the same finite window.

Apply Definition 335.5 to repack the transported endpoint, window, regular-sequence readback, seal ledger, displayed routes, and provenance as an accepted $\mathsf{DyadicApproximation}^{\uparrow}$ classifier read at m.

The final handoff is again the one described by Definition 335.3. Every exported coordinate is a transported finite row, so no quotient real, host real equality, completion point, or selected representative is available. □

Theorem 335.7 (DyadicApproximation kernel scope package)**.** *The carrier of Definition 335.1, the finite-window obligation rows, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, common precision refinement, classifier transport, and precision weakening jointly form a scoped BEDC package over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* NameCert*.*

Every accepted consumer read factors through that package. The package exports no quotient real, host real equality, completion point, or selected representative.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier kernel scope package`

Proof. Begin with Definition 335.2. It supplies the five local obligation rows for carrier habitation, classifier laws, carrier transport, finite-window exactness, and real-seal handoff over the displayed packet coordinates.

Apply Definition 335.3 for the $\mathsf{Real}^{\uparrow}$ boundary. The handoff sends downstream consumers only the finite regular-sequence source surface recorded by the window and ledger rows.

Next use Definition 335.4 and Definition 335.5. These keep precision comparison, endpoint comparison, window readback, seal ledger, displayed hsame rows, and displayed Cont routes inside the same finite package.

Finally apply Definition 335.6. Coarser precision reads are transported through the same carrier, ledger, and provenance rows. Thus every accepted consumer read factors through the stated scoped BEDC package, and no excluded completed or quotient object is exported. □

Theorem 335.8 (DyadicApproximation finite-window scope)**.** *Every accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite-window packet factors through the displayed* BHist *carrier rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and the local* NameCert *rows recorded in Definition 335.1. No consumer of the accepted window receives an additional coordinate beyond those displayed rows.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier window scope`

Proof. Project an accepted packet along the five carrier coordinates of Definition 335.1. The precision, endpoint, window, seal-ledger, and provenance coordinates give the displayed finite rows through which the packet is read.

Use Definition 335.2 for the finite obligation package and Definition 335.5 for stability under the visible hsame and Cont rows. These theorems keep the accepted classifier inside the displayed BHist, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Pkg, and NameCert surface.

The carrier has no sixth coordinate. Thus any accepted dyadic approximation window factors through exactly the displayed rows and exports no hidden quotient-real or completion coordinate. □

Theorem 335.9 (DyadicApproximation Real seal scope)**.** *Any* Real$^{\uparrow}$ *seal read from an accepted* DyadicApproximation$^{\uparrow}$ *packet consumes only the finite dyadic window, the* RegSeqRat$^{\uparrow}$ *handoff, the transported precision and endpoint rows, the visible* hsame *and* Cont *routes, and the* Pkg *and* NameCert *provenance already displayed by the packet.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier real seal scope`

Proof. Start with an accepted dyadic approximation packet and project the finite window by Definition 335.8. The source rows available to a seal read are the precision, dyadic endpoint, stream schedule, regular-sequence readback, visible transports, continuations, and provenance.

Apply Definition 335.3. Its handoff sends the Real$^{\uparrow}$ seal to the same finite dyadic window and RegSeqRat$^{\uparrow}$ readback surface, with the dyadic-radius budget and displayed routes carried by the packet ledger.

Consequently the seal read consumes only the finite handoff rows. It does not select a limit, identify a quotient real, or add a host equality coordinate. □

Theorem 335.10 (DyadicApproximation Real seal radius route)**.** *Let* $A = (n, d, w, l, p)$ *be an accepted* DyadicApproximation$^{\uparrow}$ *packet. For any coarser precision row* m *certified by the displayed* DyadicRatCore$^{\uparrow}$ *radius ledger, the resulting* Real$^{\uparrow}$ *seal read factors through the same finite route package: the transported precision row, transported dyadic endpoint, the* StreamName$^{\uparrow}$ *window, the* RegSeqRat$^{\uparrow}$ *readback, visible* hsame *and* Cont *routes, and* Pkg *provenance.*

This radius route does not introduce quotient reals, host real equality, completion points, or selected representatives.

Proof. Start with Definition 335.3. It projects the Real$^{\uparrow}$ seal read to the precision row, dyadic endpoint, finite StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, radius ledger, displayed routes, and provenance.

Apply Definition 335.6 to the coarser precision row m. The displayed DyadicRatCore$^{\uparrow}$ radius ledger transports the precision coordinate and keeps the endpoint and regular-sequence readback inside the same finite packet.

Use Definition 335.4 to compare the original packet and its coarser read through one common precision surface, then apply Definition 335.5 to repack the transported endpoint, window, readback, routes, and provenance as an accepted classifier read.

Carrier exhaustiveness for Definition 335.1 leaves only the five displayed coordinates. Hence the common route has no coordinate for a quotient real, host equality, completion point, or selected representative. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier real seal radius route certificate`

Theorem 335.11 (DyadicApproximation ledger exclusion scope)**.** *The scoped* DyadicApproximation$^{\uparrow}$ *ledger exports no quotient stream, selected limit, host real equality, trichotomy, or completeness row. Its exported data are exactly the finite-window rows of Definition 335.8 and the* Real$^{\uparrow}$ *handoff rows of Definition 335.9.*

Proof. Read the ledger through the finite-window factorization. The carrier exposes only the displayed BHist, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Read any seal coordinate through the Real-seal scope theorem. That theorem routes the seal through the finite dyadic window and regular-sequence handoff already present in the packet, so the seal contributes no completed-real coordinate.

The excluded objects are not fields of the carrier and are not produced by the handoff. Therefore the scoped ledger has no export row for quotient streams, selected limits, host real equality, trichotomy, or completeness. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier ledger exclusion scope`

Theorem 335.12 (DyadicApproximation public finite-window export)**.** *The public* NameCert$_{\text{DyadicApproximation}^{\uparrow}}$ *export is the finite packet interface* (n, d, w, l, p) *over* BHist*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*, visible* hsame*, visible* Cont*,* Pkg*, and* NameCert*.*

Every Real$^{\uparrow}$*-facing consumer read from this public export factors through that finite-window interface. The export supplies no quotient real, host equality, completion point, trichotomy row, or selected representative.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier public finite window export`

Proof. Project an accepted public packet through the package of Definition 335.7. The five coordinates are the precision row n, dyadic endpoint d, finite observation window w, seal ledger l, and provenance row p, all carried over the displayed BHist, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, hsame, Cont, Pkg, and NameCert surface.

Apply Definition 335.8 to identify the finite-window readback. This shows that any consumer access to the packet is already a read of the displayed endpoint, schedule, regular-sequence window, transport, continuation, package, and certificate rows.

Route the Real$^{\uparrow}$ seal through Definition 335.9. The seal is therefore a handoff through the same finite dyadic window and RegSeqRat$^{\uparrow}$ readback surface, rather than an additional completed-real coordinate.

Finally use Definition 335.11. Its ledger exclusions give the negative claims: the public export has no row for quotient reals, host equality, completion points, trichotomy, or selected representatives. □

Theorem 335.13 (DyadicApproximation standard finite-approximation bridge)**.** *The public finite-window export of Definition 335.12, together with Definition 335.3, Definition 335.4, and Definition 335.5, determines a conservative standard finite dyadic-approximation reading and readback.*

The readback recovers exactly the displayed precision, dyadic endpoint, StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal ledger, displayed* hsame*, displayed* Cont*, package rows, and local* NameCert *evidence. It adds no quotient real, host real equality, completion point, trichotomy row, selected representative, or classical approximation completeness.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier standard finite approximation bridge`

Proof. Project the accepted packet through Definition 335.12. The public surface is the five-coordinate packet (n, d, w, l, p): precision, dyadic endpoint, finite observation window, seal ledger,

and provenance. These rows already include the displayed StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal boundary, hsame, Cont, package, and certificate data.

Apply Definition 335.4 to place the standard finite-precision read and the BEDC readback on the same displayed precision row. Then use Definition 335.5 to carry the endpoint classifier through the visible hsame and Cont data without leaving the finite-window packet.

Finally route the seal through Definition 335.3 and return to the public export. The readback is therefore exactly the displayed precision, dyadic endpoint, stream window, regular-sequence readback, seal ledger, transport, continuation, package, and certificate surface, with no coordinate for quotient reals, host equality, completion points, trichotomy, selected representatives, or classical approximation completeness. □

Theorem 335.14 (DyadicApproximation rational interval window handoff)**.** *Let an accepted* DyadicApproximation$^{\uparrow}$ *packet expose its finite approximation coordinates, and let an accepted* RationalInterval$^{\uparrow}$ *packet expose a finite containment ledger recording the displayed dyadic endpoint at the same unary precision. Then every* Real$^{\uparrow}$*-facing consumer read factors through the common finite surface: the precision row, dyadic endpoint, rational endpoint, rational order and containment rows,* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback, visible* hsame*, visible* Cont*,* Pkg*, and local* NameCert *provenance rows.*

The handoff supplies no quotient real, selected limit, host real equality, located subset, or interval-arithmetic closure. It only identifies the shared finite endpoint window already carried by the two accepted packets and their displayed containment ledger.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier rational interval window handoff`

Proof. Project the dyadic packet through Definition 335.12. This gives the precision row, dyadic endpoint, stream window, regular-sequence readback, Real$^{\uparrow}$ seal ledger, hsame, Cont, package, and local certificate rows available to any public consumer.

Route the seal through Definition 335.3. The seal contributes no completed-real coordinate; it is read through the same finite dyadic endpoint window and the same RegSeqRat$^{\uparrow}$ readback boundary.

Project the rational packet through Definition 354.15. Its public surface exposes the rational endpoint, rational order row, containment row, finite window, transport, continuation, package, and certificate provenance.

Use the containment-ledger hypothesis to identify the rational endpoint window with the displayed dyadic endpoint at the common unary precision. After this identification, every listed consumer coordinate is one of the two carrier coordinates already projected above.

The conclusion follows by carrier-coordinate exhaustion. Since neither carrier has a row for quotient reals, selected limits, host real equality, located subsets, or interval arithmetic, the handoff cannot export those objects. □

Theorem 335.15 (DyadicApproximation consumer completeness)**.** *Every finite consumer of a* DyadicApproximation$^{\uparrow}$ *packet reads exactly the dyadic precision index, rational interval window, approximation row, classifier transport, precision weakening row,* Real$^{\uparrow}$ *seal handoff, displayed* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *evidence. Conversely, these rows repack as the complete finite dyadic-approximation consumer surface.*

The consumer surface exports no quotient real, host real equality, completion point, trichotomy row, selected representative, located subset, or classical approximation-completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier consumer completeness`

Proof. Project an accepted packet through the finite-window obligations of Definition 335.8. This exposes the precision index, dyadic endpoint, finite observation window, regular-sequence readback, visible transport, continuation, package, and local certificate rows.

Refine the consumer read to a common precision by Definition 335.4. Then use Definition 335.5 for the classifier route and Definition 335.6 for the visible weakening row.

Route the seal through Definition 335.9 and Definition 335.3. The seal contributes only the displayed $\mathsf{Real}^{\uparrow}$ handoff row already bounded by the same finite dyadic window.

Next collect the public packet surface with Definition 335.12 and the standard bridge of Definition 335.13. The rational endpoint part is the window handoff of Definition 335.14, so every finite consumer coordinate is one of the listed rows.

For the converse direction, start with precisely the listed rows. The public finite-window export supplies the NameCert package, the standard bridge packs the dyadic readback, and the rational interval handoff packs the shared endpoint window, giving a valid finite approximation consumer read.

None of the cited packet projections contains a quotient-real coordinate, host equality, completion point, trichotomy row, selected representative, located subset, or classical completeness principle. These objects therefore cannot be exported by the complete consumer surface. □

Theorem 335.16 (DyadicApproximation directed precision refinement associativity)**.** *Let three accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets carry compatible displayed common unary precision refinements. Suppose the left-associated read first refines the first two packets to their displayed common precision and then refines that visible packet with the third, while the right-associated read first refines the last two packets and then refines the first packet with that visible packet. Then the two reads expose the same consumer-visible finite packet: the transported precision row, dyadic endpoint,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal ledger, visible* hsame*, visible* Cont*,* Pkg *provenance, and local* NameCert *rows.*

No quotient real, host real equality, completion point, selected representative, trichotomy row, or choice principle is used to identify the two associated reads.

Proof. Apply Definition 335.4 to the left pair, obtaining the displayed common precision packet for the first two accepted reads. Apply the same theorem to the right pair, obtaining the displayed common precision packet for the last two accepted reads.

Apply Definition 335.4 again to the left-associated packet and the third packet. Apply it once more to the first packet and the right-associated packet. Compatibility of the two displayed common unary precision refinements makes these two terminal precision rows the same displayed row.

Use Definition 335.5 on the two terminal refinements. The transported classifier rows identify the dyadic endpoint, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal ledger, visible hsame, visible Cont, package provenance, and local certificate rows carried by the two associated packets.

Finally apply Definition 335.15. Every finite consumer coordinate is exhausted by the listed rows, and the converse packing direction shows that these rows determine the same finite dyadic-approximation consumer surface. Since the completeness theorem exports no quotient-real or choice-selected coordinate, the associativity law is entirely a finite displayed-packet statement. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier directed precision refinement associativity`

Theorem 335.17 (DyadicApproximation terminal refinement window)**.** *Let a finite family of accepted* DyadicApproximation$^{\uparrow}$ *packets be equipped with displayed common unary precision refinements, and suppose those displayed refinements are mutually compatible whenever two refinement brackets overlap. Then the iterated refinement surface has one terminal finite consumer window. That window consists exactly of the terminal precision row, the dyadic endpoint, the* StreamName$^{\uparrow}$ *window, the* RegSeqRat$^{\uparrow}$ *readback, the* Real$^{\uparrow}$ *seal ledger, the* RationalInterval$^{\uparrow}$ *containment row when that row is present, the visible* hsame *and* Cont *transport routes, the* Pkg *provenance row, and the local* NameCert *rows.*

The terminal window is independent of binary bracketing of the finite refinement family. It exports no quotient real, host equality, completion point, trichotomy row, selected representative, located subset, or choice principle.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal refinement window`

Proof. Proceed by induction on the finite refinement family. In the singleton case, the accepted packet projection supplies exactly the displayed precision row, dyadic endpoint, StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal ledger, visible hsame and Cont routes, Pkg provenance, and local NameCert rows. If the rational containment row is present, it is the row supplied by Definition 335.14.

For the binary step, assume terminal windows have already been constructed for the two finite subfamilies. Apply Definition 335.4 to their displayed terminal packets. Compatibility of the displayed common precision rows gives a single terminal precision row for the combined family.

Transport the classifier data along the common precision row by Definition 335.5. The transported rows carry the dyadic endpoint, stream window, regular rational readback, real seal ledger, visible hsame route, visible Cont route, package provenance, and local certificate rows into the combined terminal packet.

Use Definition 335.16 to compare two binary bracketings of the same finite family. The theorem identifies the consumer-visible packet rows produced by the two bracketings, so the terminal window is independent of the chosen binary refinement tree.

Finally apply Definition 335.15. It shows that the listed coordinates exhaust the finite consumer surface, while Definition 335.14 accounts for the optional rational-interval containment row. Since the involved packet coordinates contain no quotient-real, host-equality, completion-point, trichotomy, selected-representative, located-subset, or choice-principle field, none of those objects can be exported by the terminal window. □

Theorem 335.18 (DyadicApproximation mesh cell consumer boundary)**.** *Let $A = (n, d, w, l, p)$ be an accepted* DyadicApproximation$^{\uparrow}$ *packet. Suppose a* DyadicMesh$^{\uparrow}$ *consumer or a* ValidatedNumerics$^{\uparrow}$ *consumer reads A only through a displayed mesh-cell request whose cell interval is a* RationalInterval$^{\uparrow}$ *row when such a row is present. Then the consumer boundary is exactly the finite packet consisting of the unary precision row n, the* DyadicRatCore$^{\uparrow}$ *endpoint d, the* StreamName$^{\uparrow}$ *schedule carried by w, the* RegSeqRat$^{\uparrow}$ *finite readback carried by w and l, the* RationalInterval$^{\uparrow}$ *cell row when present, the* Real$^{\uparrow}$ *seal ledger, the visible* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows.*

This boundary exports no quotient real equality, host real equality, completed cell, infinite mesh, solver-correctness claim, or choice-selected representative. The mesh-cell read is a finite BEDC packet read and not a completion, infinitary mesh, or external numerical-solver principle.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier mesh cell consumer boundary`

Proof. Project the dyadic packet through Definition 335.12. This gives the unary precision row, dyadic endpoint, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal ledger, visible hsame transports, visible Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ rows.

If the mesh-cell request carries a rational-interval cell row, apply Definition 335.14. The handoff identifies the displayed dyadic endpoint with the rational endpoint window at the same unary precision and contributes only the finite $\mathsf{RationalInterval}^{\uparrow}$ order and containment rows.

For a $\mathsf{DyadicMesh}^{\uparrow}$ reader, apply Definition 358.6. The mesh contributes the level row, cell-index row, cell coverage row, containment ledger, displayed refinement route, endpoint readback, transport rows, continuation routes, provenance, and local name row, all still inside the finite mesh-cell packet.

For a $\mathsf{ValidatedNumerics}^{\uparrow}$ reader, use Definition 357.15. Its finite enclosure surface reads the same interval row, endpoint provenance, finite request, regular-rational observation window, real-name readback token, containment ledger, transports, routes, provenance, and local name row.

Finally compare the two consumer reads with Definition 335.29. The normal form exhausts the common mesh-cell consumer surface by the listed rows, so no remaining coordinate can carry quotient real equality, host real equality, a completed cell, an infinite mesh, solver correctness, or a choice-selected representative. □

Theorem 335.19 (DyadicApproximation nested window exhaustion)**.** *Every finite chain of compatible* $\mathsf{DyadicApproximation}^{\uparrow}$ *consumer refinements has one terminal displayed window. Every consumer read in the chain factors through exactly the terminal precision row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the displayed* hsame *transports, the visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *ledger.*

When a $\mathsf{RationalInterval}^{\uparrow}$ *containment row is present in the chain, it is read through that same terminal window. The exhaustion exports no quotient real, host equality, completion point, trichotomy row, selected representative, located subset, or choice principle.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier nested window exhaustion`

Proof. Induct over the finite refinement chain. The empty and singleton chains are handled by projecting the accepted packet carrier: the only consumer-visible rows are the displayed precision row, endpoint row, observation window, transport routes, package provenance, and local naming ledger.

For the binary extension step, apply Definition 335.17 to the terminal window already obtained for the initial segment and to the next compatible refinement packet. This produces one displayed terminal precision row for the extended chain.

Use Definition 335.16 to remove dependence on the chosen bracketing of the finite chain. Any two ways of building the terminal window identify the same consumer-visible precision, endpoint, observation, transport, package, and ledger rows.

Finally apply Definition 335.15. It exhausts all finite consumer reads through the terminal row, while Definition 335.14 accounts for the containment row when present. Since the carrier contains none of the excluded completion, quotient, trichotomy, selection, located-subset, or choice coordinates, the nested chain exports none of them. □

Theorem 335.20 (DyadicApproximation terminal window idempotence)**.** *Let a compatible finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *refinement chain have terminal displayed window* T*. If* T *is adjoined as one further compatible refinement, then the extended chain has the same terminal precision row, dyadic endpoint row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal ledger, optional*

$\mathsf{RationalInterval}^{\uparrow}$ *containment row, visible* hsame *transports, visible* Cont *transports,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *ledger as the original chain.*

Thus repeating a terminal displayed window contributes no consumer-visible row. The extended read exports no quotient-real equality, host equality, completed real, choice principle, or selected representative coordinate beyond the rows already present in T.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal window idempotence`

Proof. Start with the original finite chain. Apply Definition 335.17 to obtain its terminal displayed window T and the listed precision, endpoint, observation, seal, transport, package, containment, and local certificate rows.

Now append T as a compatible refinement packet. Applying Definition 335.17 again produces a terminal row for the extended chain, but the new binary refinement step uses the same displayed terminal precision row as both the accumulated window and the adjoined packet.

Use Definition 335.16 to compare the bracketing that first forms T with the bracketing that first forms T and then appends T. Associativity identifies the consumer-visible precision, endpoint, stream, regular-sequence, seal, containment, transport, package, and local ledger rows in the two bracketings.

Finally apply Definition 335.15. Since all finite consumer reads factor through those identified rows, the appended terminal window has no additional quotient-real, host-equality, completion, choice, or selected-representative coordinate to export. □

Theorem 335.21 (DyadicApproximation terminal window subchain absorption)**.** *Let a compatible finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *refinement chain have terminal displayed window T. Any finite compatible subchain whose packets factor through T may be inserted, deleted, or repeated without changing the terminal precision row, dyadic endpoint row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal ledger, optional* $\mathsf{RationalInterval}^{\uparrow}$ *containment row, visible* hsame *routes, visible* Cont *routes,* Pkg *provenance, or local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *ledger.*

Equivalently, the displayed terminal row is an absorbing finite normal form for every already-terminal compatible subchain. The normal form exports no quotient real equality, host equality, completed real, trichotomy, choice principle, or selected representative coordinate beyond the rows already visible through T.

Proof. Begin with the given finite chain and apply Definition 335.19. This places every compatible finite refinement packet in the same directed displayed-window spine and supplies the common precision, endpoint, stream, regular-sequence, seal, containment, transport, package, and local certificate rows.

For a subchain whose packets factor through T, apply Definition 335.20 to each displayed terminal packet in the subchain. Repetition of such a packet contributes no new consumer-visible row, and deletion removes only rows already identified with the terminal row.

It remains to compare the positions of the subchain inside the whole finite refinement. Use Definition 335.16 to rebracket the refinement so the subchain is read either before, after, or at the displayed terminal row. The associated bracketings have the same precision, endpoint, stream, regular-sequence, seal, containment, visible transport, package, and local ledger data.

Finally apply Definition 335.15. Every finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and optional $\mathsf{RationalInterval}^{\uparrow}$ consumer read factors through the terminal displayed row, so insertion, deletion, and repetition of an already-terminal compatible subchain leave exactly the same public finite-window surface. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal refinement subchain absorption`

Theorem 335.22 (DyadicApproximation window RegSeqRat source exhaustion)**.** *Every accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet together with a finite compatible refinement chain determines exactly one displayed* $\mathsf{RegSeqRat}^{\uparrow}$*-facing source surface. The surface consists of the unary precision row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the* $\mathsf{Real}^{\uparrow}$ *seal ledger, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *row.*

If the same packet is read through the $\mathsf{RationalInterval}^{\uparrow}$ *handoff or the* $\mathsf{Real}^{\uparrow}$ *seal boundary, those consumers read that same source surface and no other row. The exhaustion exports no quotient real equality, selected limit, trichotomy row, located subset, or choice principle.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier window regseqrat source exhaustion`

Proof. Start with the finite refinement chain and apply Definition 335.19. This gives one terminal displayed window containing the precision, endpoint, stream, regular-sequence, transport, continuation, package, and local certificate rows seen by every consumer read in the chain.

Next apply Definition 335.9. The $\mathsf{Real}^{\uparrow}$ seal ledger is a finite boundary row attached to the same terminal window, so it does not contribute a completed real or a selected limit outside the displayed packet surface.

Use Definition 335.3 to read the terminal window as a real-seal source for $\mathsf{RegSeqRat}^{\uparrow}$. The handoff identifies the regular rational observation row, seal ledger, visible hsame route, visible Cont route, and package provenance as the only real-facing source data.

Finally apply Definition 335.14. It shows that any rational-interval consumer sees the same endpoint and observation rows, so the terminal window is the unique displayed source surface for all the named consumers. Since all cited surfaces are finite packet rows, quotient real equality, selected limits, trichotomy, located subsets, and choice principles are not exported. □

Theorem 335.23 (DyadicApproximation overlap-directed seal exhaustion)**.** *Let two accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets carry overlapping finite* $\mathsf{StreamName}^{\uparrow}$ *windows and overlapping finite* $\mathsf{RegSeqRat}^{\uparrow}$ *observation windows. Suppose their overlap is equipped with a displayed directed common unary precision refinement, so that both dyadic endpoints and both seal ledgers are transported to the same visible precision row. Then every* $\mathsf{Real}^{\uparrow}$ *seal read from either packet factors through one finite boundary surface: the common precision row, the transported dyadic endpoints, the* $\mathsf{RationalInterval}^{\uparrow}$ *refinement window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source readback, the visible* hsame *transports, the visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows.*

This finite boundary exports no quotient-real equality, host real equality, completion point, trichotomy row, selected representative, located subset, or choice principle. The overlap theorem is only a coverage statement for the displayed finite rows already present in the two accepted packets and their directed refinement data.

Proof. First put the two accepted packets on the same displayed precision row. Apply Definition 335.16 to make the directed refinement independent of the binary route used to reach the common row.

Next apply Definition 335.17 to the overlapping two-packet family. This produces the terminal finite window containing the common precision row, transported endpoints, stream window, regular

rational observation window, seal ledger, visible transports, continuations, package provenance, and local certificate rows.

Use Definition 335.14 to read the transported dyadic endpoints through the shared $\mathsf{RationalInterval}^{\uparrow}$ refinement window. The handoff keeps the endpoint and containment data inside the same finite overlap surface.

Then apply Definition 335.22. It identifies the $\mathsf{RegSeqRat}^{\uparrow}$ source readback and shows that both $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers read the same displayed source surface.

Finally route the seal boundary through Definition 335.9. Since the seal scope has no coordinate for quotient-real equality, host equality, completion points, trichotomy, selected representatives, located subsets, or choice data, the overlap-directed read cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier overlap directed seal exhaustion`

Theorem 335.24 (DyadicApproximation overlap seal idempotence)**.** *Let two accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets satisfy the hypotheses of Definition 335.23, and let their resulting finite* $\mathsf{Real}^{\uparrow}$ *seal boundary be read again through any compatible displayed overlap refinement. Then the second read factors through the same common precision row, the same transported dyadic endpoint rows, the same* $\mathsf{RationalInterval}^{\uparrow}$ *refinement window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source readback, the same visible* hsame *transports, the same visible* Cont *routes, the same* Pkg *provenance, and the same local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows as the first seal read.*

Thus repeated $\mathsf{Real}^{\uparrow}$ *seal reads through a fixed finite overlap boundary are idempotent as displayed packet reads. The idempotence assertion exports no quotient-real equality, host real equality, completion point, trichotomy row, selected representative, located subset, or choice principle.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier overlap seal idempotence`

Proof. First apply Definition 335.23 to the two accepted packets and their displayed directed common precision refinement. This identifies the first $\mathsf{Real}^{\uparrow}$ seal read with the finite overlap boundary containing the common precision row, transported endpoint rows, rational interval window, regular rational source readback, visible transports, continuations, package provenance, and local certificate rows.

Next apply Definition 335.20 to the compatible displayed overlap refinement used by the second read. Re-reading a terminal finite window through a compatible refinement returns the same terminal window, so the second seal read cannot introduce a new precision row, endpoint row, consumer route, package row, or local certificate row.

Finally apply Definition 335.22. It identifies the unique $\mathsf{RegSeqRat}^{\uparrow}$-facing source surface seen by the $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, and therefore transports the second seal read back to the same source surface already named by the first overlap-directed read. Since each cited result is a statement about finite displayed packet rows, the repeated read has no coordinate for quotient-real equality, host real equality, completion points, trichotomy, selected representatives, located subsets, or choice data. □

Theorem 335.25 (DyadicApproximation overlap refinement chain normal form)**.** *Let two accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets satisfy the overlap hypotheses of Definition 335.23. For every finite chain of compatible displayed overlap refinements, every repeated* $\mathsf{Real}^{\uparrow}$ *seal read factors through one terminal normal-form packet. This packet consists exactly of the common precision row, the transported dyadic endpoint rows, the* $\mathsf{RationalInterval}^{\uparrow}$ *refinement window, the*

$\mathsf{RegSeqRat}^{\uparrow}$ *source readback, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows.*

The normal form is a finite displayed packet normal form for overlap reads. It does not contain quotient-real equality, host equality, completion points, trichotomy rows, selected representatives, located subsets, or choice data.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier overlap refinement chain normal form`

Proof. First apply Definition 335.23. The first common overlap boundary supplies the displayed precision row, transported endpoints, rational-interval window, regular rational readback, visible transport routes, package provenance, and local certificate rows shared by the two accepted packets.

Next run the finite compatible refinement chain through Definition 335.20. Once the common boundary has been read as a terminal displayed window, compatible re-reading stabilizes at the same terminal packet and cannot add a second endpoint, precision, route, provenance, or certificate row.

Then apply Definition 335.22. This identifies the unique $\mathsf{RegSeqRat}^{\uparrow}$-facing source surface seen by both the $\mathsf{RationalInterval}^{\uparrow}$ window and the repeated $\mathsf{Real}^{\uparrow}$ seal reads, so the terminal packet is the single source normal form for the whole chain.

Finally apply Definition 335.9. The $\mathsf{Real}^{\uparrow}$ boundary has only the displayed seal rows named above, so the normal-form packet excludes quotient-real equality, host equality, completion points, trichotomy rows, selected representatives, located subsets, and choice data. □

Theorem 335.26 (DyadicApproximation overlap coarsening seal stability)**.** *Let two accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets satisfy Definition 335.23. Suppose both packets are weakened to a common coarser unary precision row through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. Then every compatible* $\mathsf{Real}^{\uparrow}$ *seal reread factors through the same finite overlap surface, now read at the coarser precision row with transported endpoints, the* $\mathsf{RationalInterval}^{\uparrow}$ *refinement window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, visible* hsame *and* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows.*

The coarsened reread exports no quotient-real equality, host real equality, completion point, trichotomy row, selected representative, located subset, or choice principle.

Proof. Apply Definition 335.6 to each accepted packet, using the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger to transport the endpoint and seal rows to the common coarser precision row.

Next use Definition 335.16 to identify the two weakening routes with one directed coarsening surface. The common row therefore carries the same transported endpoint, interval, stream, regular-sequence, transport, continuation, package, and local certificate data.

Apply Definition 335.23 on this coarser row. It supplies the finite overlap boundary seen by the $\mathsf{Real}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ consumers.

Finally apply Definition 335.24. A compatible reread of the already displayed seal boundary remains on that same finite boundary, so no completed real, host equality, representative choice, located subset, trichotomy, or choice data can be exported. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier overlap coarsening seal stability`

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$**.**

Constructive story: A finite dyadic approximation packet is generated from a unary precision row, a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, a $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ finite observation window, a $\mathsf{Real}^{\uparrow}$ seal ledger, visible hsame and Cont transport, and Pkg provenance.

Theory closure: TheoryGate(NameCert$_{\text{DyadicApproximation}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 335.8, Definition 335.9, Definition 335.11, Definition 335.12, Definition 335.13, Definition 335.14, Definition 335.15, Definition 335.16, and Definition 335.17 form the public finite-window export, standard finite dyadic-approximation readback, rational interval handoff, finite consumer surface, directed refinement associativity, and terminal common-refinement window for the accepted dyadic window, Real$^\uparrow$ seal handoff, and ledger exclusions over BHist, DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, RationalInterval$^\uparrow$, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{DyadicApproximation}^\uparrow}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient reals, host real equality, completion points, or choice-selected representatives.

Upgrade path: No further paper theory-axis grade remains; formal verification remains a separate axis for the same dyadic approximation packet.

Theorem 335.27 (DyadicApproximation terminal window prefix projection)**.** *Let a compatible finite* DyadicApproximation$^\uparrow$ *refinement chain have terminal displayed window T. For every initial compatible subchain, the projected displayed window consists exactly of the prefix rows of T: the precision row, the dyadic endpoint row, the* StreamName$^\uparrow$ *row, the* RegSeqRat$^\uparrow$ *observation row, the* Real$^\uparrow$ *seal row, the optional* RationalInterval$^\uparrow$ *containment row when present, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance, and the local* NameCert$_{\text{DyadicApproximation}^\uparrow}$ *row.*

After this prefix projection, rereading the full compatible chain returns the same terminal window T. The projection exports no quotient real, host equality, completion point, trichotomy, selected representative, located subset, or choice principle.

Proof. Apply Definition 335.19 to the full finite chain. This gives the terminal window T and identifies every consumer-visible row with the terminal precision, endpoint, stream, regular-sequence, seal, containment, transport, package, and local certificate rows.

Apply the same exhaustion theorem to the chosen initial compatible subchain. The prefix read has its own displayed terminal window, but its rows are drawn from the same directed finite refinement spine as the full chain.

Use Definition 335.17 to compare the terminal row of the initial subchain with the terminal row of the full chain. The comparison says that each visible prefix row is the corresponding initial row of T, including the optional rational-interval containment row when that row is part of the displayed chain.

Now remove bracketing dependence by Definition 335.16. Whether the full terminal window is read before or after projecting the initial subchain, the same precision, endpoint, observation, seal, transport, provenance, and local certificate rows are obtained.

Finally apply Definition 335.15. Every finite consumer read after the prefix projection factors through the same terminal window T, and Definition 335.14 accounts for the optional containment row. Since the packet has no quotient-real, host equality, completion, trichotomy, representative-selection, located-subset, or choice coordinate, none is exported by the projection. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal window prefix projection`

Theorem 335.28 (DyadicApproximation terminal-window two-stage factorization)**.** *Let $A = (n, d, w, l, p)$ be an accepted terminal* DyadicApproximation$^\uparrow$ *packet with terminal displayed window T. Suppose its terminal readback is projected by Definition 335.27, and suppose the same packet is consumed through the validated enclosure route of Definition 335.32.*

Then the two reads factor through one finite public surface. That surface is exactly the terminal precision row, the transported dyadic endpoint row, the StreamName$^{\uparrow}$ *window, the* RegSeqRat$^{\uparrow}$ *readback window, the* Real$^{\uparrow}$ *seal ledger, the validated containment ledger, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance rows, and the local* NameCert *rows. Reading first by terminal prefix projection and then through the validated enclosure route gives the same displayed surface as reading first through the enclosure route and then commuting back to the terminal dyadic window.*

The factorization introduces no quotient real, host-real equality, solver-correctness assertion, or selected completion point.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal window two stage factorization`

Proof. First apply Definition 335.27. The terminal readback of A is reduced to the prefix rows of T: precision, endpoint, stream, regular-rational observation, real-seal ledger, visible transport, continuation, provenance, and local certificate data.

Next exhaust the finite packet read. By Definition 335.30, every consumer-visible read of the accepted terminal packet is already among those finite rows together with the displayed validated enclosure handoff. Apply Definition 335.31 to remove dependence on which compatible finite refinement exposes the terminal surface.

Now read the enclosure side. Apply Definition 335.32 to identify the DyadicMesh$^{\uparrow}$ cell and ValidatedNumerics$^{\uparrow}$ enclosure read with the same terminal precision, endpoint, regular-rational, real-seal, containment, transport, continuation, provenance, and local certificate rows.

Finally apply Definition 335.34. This commutes the terminal dyadic route with the mesh-and-enclosure route, so both orders factor through the single finite surface listed in the statement. Every cited boundary is a finite BEDC readback boundary, hence none supplies a quotient real, host-real equality, solver-correctness row, or chosen completion point. □

Theorem 335.29 (DyadicApproximation DyadicMesh ValidatedNumerics consumer normal form)**.** *Let $A = (n, d, w, l, p)$ be an accepted* DyadicApproximation$^{\uparrow}$ *packet. Let $M = (\ell, J, I, E, H, C, P, N)$ be an accepted* DyadicMesh$^{\uparrow}$ *cell whose* RationalInterval$^{\uparrow}$ *row I contains the transported dyadic endpoint d at the displayed precision n. Let V be a* ValidatedNumerics$^{\uparrow}$ *enclosure reading the same mesh cell through the displayed enclosure row I and the same containment ledger.*

Then the three accepted packets determine one finite RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer surface. The surface consists exactly of the precision row n, the* DyadicRatCore$^{\uparrow}$ *endpoint d, the* RationalInterval$^{\uparrow}$ *containment row I, the* DyadicMesh$^{\uparrow}$ *level row ℓ, the cell-index row J, the* RegSeqRat$^{\uparrow}$ *readback carried by w and l, the* Real$^{\uparrow}$ *seal ledger of A, the displayed* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance rows, and the local* NameCert *rows of the three packets.*

The normal form exports no quotient real, solver-correctness claim, floating-point semantics, host equality, or classical completeness principle.

Proof. Project the accepted dyadic packet first. By Definition 335.12 and Definition 335.17, its public finite window is the displayed precision row, dyadic endpoint, stream window, regular-rational readback, real-seal ledger, visible transports, continuations, package provenance, and local certificate data.

Project the mesh packet next. By Definition 358.4 and Definition 358.6, the mesh contributes only the level row, cell-index row, rational interval coverage row, dyadic containment ledger, refinement route, endpoint readback, package provenance, and local certificate row.

Read the validated enclosure through Definition 357.15. Its consumer boundary is the finite enclosure row, endpoint provenance, precision request, regular-rational readback, real-name readback token, containment routes, transports, package provenance, and local certificate row.

The shared-containment hypothesis identifies the dyadic endpoint read from A, the rational interval row I from M, and the enclosure row read by V at the displayed precision. After that identification, all remaining rows are the listed hsame, Cont, Pkg, and local NameCert data.

Repack those projected rows as the consumer surface. Carrier exhaustion in the three cited packets leaves no coordinate for quotient reals, solver correctness, floating-point hardware semantics, host equality, or classical completeness. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier dyadicmesh validatednumerics consumer normal form`

Theorem 335.30 (DyadicApproximation finite packet readback exhaustion)**.** *Let $A = (n, d, w, l, p)$ be an accepted* DyadicApproximation$^{\uparrow}$ *finite packet, and let C be a finite-packet consumer whose reads of A are routed through the public finite-window export, the terminal overlap chain, the overlap refinement normal form, and the* DyadicMesh$^{\uparrow}$/ValidatedNumerics$^{\uparrow}$ *consumer normal form.*

Then the readback surface seen by C is exactly the displayed precision row n, the DyadicRatCore$^{\uparrow}$ *endpoint window, the finite* RegSeqRat$^{\uparrow}$ *readback window, the* Real$^{\uparrow}$ *seal token, the refinement ledger, the overlap route, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance rows, the local* NameCert$_{\text{DyadicApproximation}^{\uparrow}}$ *rows, and the validated enclosure handoff carried by the displayed* DyadicMesh$^{\uparrow}$/ValidatedNumerics$^{\uparrow}$ *consumer boundary.*

The readback surface has no coordinate for a completed real, infinite mesh, solver state, host interval object, host equality, selected representative, trichotomy row, or choice data.

Proof. First project the accepted packet through Definition 335.12. The public rows are the displayed precision, dyadic endpoint, stream window, regular-rational readback, real-seal ledger, visible transport and continuation routes, package provenance, and local certificate data.

Next stabilize the overlap access. Apply Definition 335.20 to the terminal read and then apply Definition 335.25 to the compatible overlap refinement chain. These two results identify the terminal overlap route with the same finite packet rows and leave no second endpoint, precision row, provenance row, or seal coordinate.

Now pass to the downstream consumer. By Definition 335.29, the DyadicMesh$^{\uparrow}$ cell and the ValidatedNumerics$^{\uparrow}$ enclosure read only the displayed containment row, mesh level and cell-index rows, endpoint window, enclosure handoff, containment routes, package provenance, and local certificate rows attached to the same dyadic packet.

Coordinate exhaustion finishes the claim. Every row available to C appears in one of the three projected surfaces just listed, and the listed surfaces have no carrier field or certificate row for a completed real, infinite mesh, solver state, host interval object, host equality, selected representative, trichotomy row, or choice data. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier finite packet readback exhaustion`

Theorem 335.31 (DyadicApproximation refined finite-packet readback determinacy)**.** *Let A and A' be accepted* DyadicApproximation$^{\uparrow}$ *finite packets. Suppose their transported precision rows,* DyadicRatCore$^{\uparrow}$ *endpoint windows, terminal overlap routes, overlap refinement normal forms,* RegSeqRat$^{\uparrow}$ *readback windows,* Real$^{\uparrow}$ *seal tokens, and* DyadicMesh$^{\uparrow}$/ValidatedNumerics$^{\uparrow}$ *consumer*

normal forms are the same displayed BEDC rows. Suppose also that their visible hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows agree.*

Then every displayed precision refinement reads the same public finite surface from A and A'. The common read is exactly the displayed precision row, endpoint window, terminal overlap route, overlap normal form, regular-rational readback window, real seal token, mesh-and-validated consumer handoff, visible transport rows, continuation routes, package provenance, and local certificate rows named above.

The determinacy statement does not identify completed real values, does not export host interval objects, does not decide a trichotomy row, and does not read solver state or choice data.

Proof. First exhaust the finite readback of each packet. Apply Definition 335.30 to both A and A'. The only rows available to either consumer are the displayed precision, endpoint, readback, seal, refinement, overlap, transport, continuation, package, local certificate, and $\mathsf{DyadicMesh}^{\uparrow}/\mathsf{ValidatedNumerics}^{\uparrow}$ handoff rows.

Next normalize the overlap side. Use Definition 335.25 for the displayed overlap refinement chain of each packet. The hypotheses identify the terminal route and the overlap normal form, so the two packet reads have the same terminal finite route after any displayed precision refinement.

Finally normalize the downstream consumer. By Definition 335.29, the mesh cell and validated enclosure can read only the common containment, mesh-level, cell-index, endpoint-window, regular-rational, real-seal, transport, continuation, package, and local certificate rows. Since those rows are assumed equal as displayed BEDC data, the public finite read is the same.

The cited exhaustion and normal-form results also list the absent coordinates. Thus the conclusion cannot be strengthened into completed-real equality, host interval equality, trichotomy, solver-state agreement, or a choice principle. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier refined finite packet readback determinacy`

Theorem 335.32 (DyadicApproximation terminal validated enclosure readback)**.** *Let a compatible finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *refinement chain have terminal displayed window T. Suppose the prefix projection of the chain has stabilized and the overlap refinement chain reads in the displayed overlap normal form. If a* $\mathsf{DyadicMesh}^{\uparrow}$ *finite cell and a* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure consume that same terminal window, then the terminal readback surface is exactly the same finite enclosure surface consumed by those two downstream packets.*

Componentwise, the common surface consists of the terminal precision row, the dyadic endpoint and radius route, the $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the* $\mathsf{DyadicMesh}^{\uparrow}$ *rational-interval cell row, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment ledger, the* $\mathsf{Real}^{\uparrow}$ *readback token, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance rows, and the local* NameCert *rows. The readback does not assert solver correctness, floating-point semantics, classical real completeness, or host real equality.*

Proof. First stabilize the terminal dyadic window. Apply Definition 335.27 to identify every prefix read with the corresponding row of T: precision, endpoint, regular-rational observation, real-seal token, visible transport, continuation, provenance, and local certificate rows.

Next stabilize overlap access. Apply Definition 335.25. The overlap read has the same terminal precision row, transported endpoint route, rational-interval window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal boundary, visible hsame transports, visible Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ rows.

Now pass to the finite cell and enclosure consumers. By Definition 335.29, the $\mathsf{DyadicMesh}^{\uparrow}$ cell and the $\mathsf{ValidatedNumerics}^{\uparrow}$ enclosure read the dyadic terminal window only through the displayed precision row, endpoint row, rational-interval cell row, mesh level and cell-index rows, validated containment ledger, regular-rational readback, real-seal token, transport routes, provenance, and local certificate rows.

The mesh handoff and the enclosure boundary identify the downstream rows. Apply Definition 358.6 for the rational interval cell, containment ledger, endpoint readback, and mesh provenance. Apply Definition 357.15 for the validated enclosure row, precision request, containment ledger, real-name readback token, transport rows, continuation routes, package provenance, and local certificate row.

Repack the identified rows as one finite readback surface. The preceding steps account for every component listed in the theorem statement, and the cited consumer boundaries contain no coordinate for solver correctness, floating-point semantics, classical real completeness, or host real equality. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal validated enclosure readback`

Theorem 335.33 (DyadicApproximation terminal mesh enclosure compatibility)**.** *Let an accepted finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet have consumer read with terminal displayed window* T*. For any downstream read through the* $\mathsf{DyadicMesh}^{\uparrow}$ *standard finite mesh bridge and the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure packet, the mesh cell and the validated enclosure consume the same terminal window* T*.*

Equivalently, the rational interval cell, dyadic endpoint-radius-order ledger, $\mathsf{Real}^{\uparrow}$ *boundary token, validated enclosure row, visible* hsame *rows, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows agree as one terminal finite surface. This common surface exports no completed real, infinite mesh, solver correctness row, floating-point semantics row, or compactness row.*

Proof. First exhaust the accepted finite packet. Apply Definition 335.30 to read the terminal displayed packet as its precision row, dyadic endpoint and radius rows, order ledger, regular-rational observation window, $\mathsf{Real}^{\uparrow}$ boundary token, visible transports, continuation routes, package provenance, and local certificate rows.

Next identify the terminal enclosure route. Apply Definition 335.32. The validated enclosure row can see the finite dyadic approximation only through the terminal displayed window T and the rows listed there.

The mesh and enclosure consumers are then put into the same normal form. Apply Definition 335.29 to identify the $\mathsf{DyadicMesh}^{\uparrow}$ cell row and the $\mathsf{ValidatedNumerics}^{\uparrow}$ containment row with the terminal precision, endpoint-radius-order, real boundary, transport, continuation, provenance, and local certificate rows.

Finally use the standard finite mesh bridge, Definition 358.8. It states that the displayed mesh read is exactly the rational interval cell read by the downstream consumer. Combining the four cited readbacks leaves a single terminal finite surface and no coordinate in which a completed real, infinite mesh, solver correctness assertion, floating-point semantics assertion, or compactness claim could be exported. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal mesh enclosure compatibility`

Theorem 335.34 (DyadicApproximation validated enclosure route commutation)**.** *Let* $A = (n, d, w, l, p)$ *be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet. Let* $M = (\ell, J, I, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$

cell whose RationalInterval$^{\uparrow}$ *row I contains the transported dyadic endpoint d at the displayed precision n. Let V be an accepted* ValidatedNumerics$^{\uparrow}$ *enclosure over that same cell and containment ledger.*

The route that first reads A through Definition 335.12 and then exposes the Real$^{\uparrow}$ *seal boundary reads exactly the same finite* RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *consumer surface as the route that first reads M through Definition 358.6 and then exposes V through Definition 357.15. The common surface consists precisely of the precision row, transported dyadic endpoint, rational-interval cell, mesh level and cell index, regular rational observation window, validated containment ledger, real readback token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

This correspondence does not assert solver correctness, floating-point semantics, quotient-real equality, host real equality, or classical completeness.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier validated enclosure route commutation`

Proof. First normalize the three packets together. Apply Definition 335.29 to the accepted dyadic approximation packet, the mesh cell containing its transported endpoint, and the validated enclosure over the same cell. This produces one finite consumer surface over the listed precision, endpoint, mesh, regular-rational, real-seal, transport, continuation, provenance, and local certificate rows.

Next identify the dyadic route. Apply Definition 335.27 to replace every finite prefix read of A by the terminal displayed window, and apply Definition 335.12 to read the public packet through that finite window. The Real$^{\uparrow}$ seal therefore remains attached to the same RegSeqRat$^{\uparrow}$ observation window and dyadic endpoint row.

Now identify the mesh and enclosure route. Apply Definition 358.4 to read the mesh cell as the rational interval containing the transported endpoint, and apply Definition 358.6 to pass that cell to the validated enclosure surface.

Finally apply Definition 357.15 and Definition 357.16. These theorems say that the validated route exposes exactly the same finite rational-interval, regular-rational, real-readback, containment, transport, continuation, provenance, and local certificate rows. Repacking those rows gives the claimed route commutation, while the excluded semantic obligations remain outside every cited boundary. □

Theorem 335.35 (DyadicApproximation mesh validated Real boundary)**.** *Let $A = (n, d, w, l, p)$ be an accepted* DyadicApproximation$^{\uparrow}$ *packet. Let $M = (\ell, J, I, E, H, C, P, N)$ be an accepted* DyadicMesh$^{\uparrow}$ *cell whose* RationalInterval$^{\uparrow}$ *cell I contains the transported dyadic endpoint d at the displayed precision n. Let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p', n')$ be an accepted* ValidatedNumerics$^{\uparrow}$ *enclosure over the same rational-interval cell and the same containment ledger.*

Suppose the displayed precision row, dyadic endpoint row, rational-interval cell, containment ledger, regular-rational window, mesh level and index, real-readback token, visible hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *rows are shared by the three accepted packets. Then the DyadicApproximation terminal finite-window route, the DyadicMesh cell-containment route, and the ValidatedNumerics real-readback route expose exactly one finite* RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *consumer surface.*

The shared surface is precisely the precision row, endpoint row, rational-interval cell, mesh level and index, regular-rational window, validated containment ledger, real readback token, visible hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no solver correctness, floating-point semantics, host real equality, quotient reals, or classical completeness.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier mesh validated real boundary`

Proof. First expose the DyadicApproximation terminal route. Apply Definition 335.30 to account for every public read of A, and apply Definition 335.32 to tie the terminal finite window to the validated enclosure readback rows.

Next identify the common three-packet normal form. Apply Definition 335.29 to the accepted dyadic packet, the mesh cell containing its transported endpoint, and the validated enclosure over the same cell and containment ledger. This produces one finite surface over the listed precision, endpoint, mesh, regular-rational, real-readback, transport, continuation, provenance, and local naming rows.

Now compare the routes. Apply Definition 335.34 to identify the route that reads the terminal DyadicApproximation finite window and then the $\mathsf{Real}^{\uparrow}$ seal with the route that passes through the DyadicMesh cell and the ValidatedNumerics enclosure.

Finally attach the neighboring consumer boundaries. Apply Definition 358.2 for the mesh level, cell-index, rational-interval, containment-ledger, transport, continuation, provenance, and local name rows. Apply Definition 358.6 and Definition 357.3 to read the validated enclosure as the same finite real-readback surface.

The carrier-exhaustion, normal-form, route-commutation, mesh-containment, and validated-readback boundaries leave no coordinate for solver correctness, floating-point semantics, host real equality, quotient reals, or classical completeness. The three routes therefore expose exactly the single finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer surface stated above. □

Definition 335.36 (DyadicApproximation mesh-enclosure exactness packet)**.** *Let $A = (n, d, w, l, p)$ be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *row, let $M = (\ell, J, I, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *cell row, and let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p', n')$ be an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row. A DyadicApproximation mesh-enclosure exactness packet is the finite BEDC packet consisting of these three rows together with the following shared coordinates:*

1. *a shared displayed precision row identifying n, ℓ, and ε as the same finite request through visible* hsame *rows;*
2. *the rational interval cell I and its dyadic endpoint-radius-order ledger E, with d read as the transported endpoint contained in I;*
3. *the* $\mathsf{Real}^{\uparrow}$ *finite-window seal read from w, l, ρ, and λ, with no completed real coordinate added to the packet;*
4. *the visible* hsame *transports that identify the dyadic endpoint, regular-rational observation window, mesh cell, and enclosure readback rows;*
5. *the visible* Cont *routes by which the dyadic finite window, mesh cell containment, rational-interval coverage, validated readback, and precision refinement are inspected by consumers;*
6. *the* Pkg *provenance rows p, P, and p' naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, *and* $\mathsf{ValidatedNumerics}^{\uparrow}$ *dependency surface;*
7. *the local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *provenance rows.*

The packet is finite: its carrier is only the accepted rows and the displayed transport, continuation, provenance, and local naming evidence listed above.

Lean4 (definition) `BEDC.Derived.DyadicApproximationUp.DyadicApproximationMeshEnclosureExactnessPacket`

Theorem 335.37 (DyadicApproximation finite mesh-enclosure exactness)**.** *Let $A = (n, d, w, l, p)$ be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *row, let $M = (\ell, J, I, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *cell row whose* $\mathsf{RationalInterval}^{\uparrow}$ *cell I contains the transported endpoint d at the displayed precision, and let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p', n')$ be an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row over the same cell and containment ledger. Assume the visible* hsame, Cont, Pkg, *and local* NameCert *rows identify the precision request, rational interval cell,* $\mathsf{Real}^{\uparrow}$ *finite window seal, and enclosure readback rows as in Definition 335.36.*

Then every accepted mesh-cell or validated-enclosure consumer read through that packet factors through one finite $\mathsf{Real}^{\uparrow}$ *surface. The factorization is exactly the DyadicApproximation* $\mathsf{Real}^{\uparrow}$ *seal handoff and finite-window scope, the DyadicMesh cell-containment and rational-interval coverage rows, the DyadicMesh handoff to ValidatedNumerics, and the ValidatedNumerics real readback and precision-refinement containment rows.*

Consequently the mesh-cell consumer and the validated-enclosure consumer read the same finite surface: the shared precision row, transported dyadic endpoint, rational interval cell, endpoint-radius-order ledger, regular-rational observation window, $\mathsf{Real}^{\uparrow}$ *readback token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *provenance. The statement does not assert completed real equality, infinite mesh refinement, solver correctness, floating-point semantics, or choice-selected limits.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier finite mesh enclosure exactness`
Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationMeshEnclosureExactnessPacket consumes real surface`

Proof. First project the accepted dyadic row. Apply Definition 335.3 to expose the only $\mathsf{Real}^{\uparrow}$ handoff surface carried by A, and apply Definition 335.8 to restrict all dyadic reads to the displayed finite window. This produces the precision row, transported endpoint, regular-rational observation window, seal ledger, visible transports, continuation routes, package provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ row.

Next project the mesh rows. Apply Definition 358.2 for the level, cell index, rational interval cell, dyadic containment ledger, transport, continuation, provenance, and local naming rows. Apply Definition 358.4 to read I as the complete finite coverage object, and apply Definition 358.6 to identify the mesh handoff rows consumed by the enclosure.

Now project the validated enclosure rows. Apply Definition 357.3 to tie the $\mathsf{Real}^{\uparrow}$ readback token to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ window and precision request. Apply Definition 357.5 to keep every refined finite read inside the same rational interval cell and containment ledger.

Finally repack the projected rows into the exactness packet of Definition 335.36. The identifications in the hypotheses make the dyadic seal handoff, mesh cell coverage, and validated enclosure readback one shared finite BEDC surface. Every cited boundary is a finite packet boundary, so no completed real equality, infinite mesh refinement, solver correctness, floating-point semantics, or choice-selected limit is exported. □

Theorem 335.38 (DyadicApproximation terminal finite-packet Real readback)**.** *Let a compatible finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *refinement chain have accepted terminal packet T. Suppose a consumer reads T through one of the following interfaces: terminal prefix projection, mesh-cell enclosure, validated readback, or the* $\mathsf{Real}^{\uparrow}$ *seal route. If the displayed transport, continuation, package, and*

local naming rows identify these interfaces over the same terminal window, then all four reads factor through one finite terminal surface.

That surface is exactly the terminal precision row, the transported dyadic endpoint, the $\mathsf{RationalInterval}^{\uparrow}$ *cell, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal ledger, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment row, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance, and the local* NameCert *provenance. It exports no completed real equality, solver correctness, floating-point semantics, or choice-selected limit.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal finite packet real readback`

Proof. First put terminal prefix access and terminal readback in the same normal form. Apply Definition 335.28. It identifies each prefix-projected read with the terminal precision, endpoint, stream-name window, regular-rational readback, $\mathsf{Real}^{\uparrow}$ seal row, visible transport and continuation rows, package provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ row.

Next attach the finite mesh and validated enclosure surface. Apply Definition 335.37. This reads the transported endpoint through the rational interval cell, the endpoint containment ledger, the validated readback rows, and the shared $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ finite surface.

Now compare the terminal mesh route with the terminal validated route. Apply Definition 335.33 to keep the mesh-cell consumer on the same terminal precision, endpoint, rational-interval, containment, transport, continuation, provenance, and local certificate rows. Apply Definition 335.32 to read the validated enclosure through that same terminal finite surface.

The preceding identifications list every coordinate available to the four consumer interfaces. Repacking those coordinates gives one terminal finite-packet surface, and none of the cited boundaries contains a coordinate for completed real equality, solver correctness, floating-point semantics, or a choice-selected limit. □

Theorem 335.39 (DyadicApproximation common-refinement enclosure handoff)**.** *Let* $A = (n, d, w, l, p)$ *and* $A' = (n', d', w', l', p')$ *be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets. Suppose they admit a displayed common precision refinement* R *whose endpoint rows, terminal finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *observation windows,* $\mathsf{Real}^{\uparrow}$ *seal tokens, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows are the same finite BEDC data.*

Let $M = (\ell, J, I, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *cell row and let* $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, q, m)$ *be an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row. Suppose the mesh-cell row and the validated-enclosure row consume the same refined window* R*: the displayed precision row is shared, the transported dyadic endpoint is contained in the rational interval cell* I*, the endpoint-radius-order ledger* E *is the validated containment ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window and* $\mathsf{Real}^{\uparrow}$ *readback token are the rows read by* V*.*

Then the mesh-cell handoff and the validated-enclosure handoff factor through one finite $\mathsf{Real}^{\uparrow}$ *surface. That surface is exactly the shared precision row, transported dyadic endpoint, rational interval cell, endpoint-radius-order ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window,* $\mathsf{Real}^{\uparrow}$ *readback token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *rows. The factorization exports no completed real equality, host interval equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limit.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier common refinement enclosure handoff`

Proof. First identify the common refined dyadic read. Apply Definition 335.31 to A, A', and the displayed refinement R. The two accepted finite packets therefore expose the same precision, endpoint, terminal-window, regular-rational, real-seal, transport, continuation, provenance, and local certificate rows.

Next attach the mesh and enclosure rows to that common read. Apply Definition 335.37 to the accepted dyadic row selected by R, the mesh cell M, and the validated enclosure V. This places the dyadic seal handoff, mesh cell containment, rational-interval coverage, validated containment ledger, and $\mathsf{Real}^{\uparrow}$ readback token on one finite surface.

Now read the terminal enclosure boundary. Apply Definition 335.32. The terminal window consumed by M and V has the same precision row, transported endpoint, regular-rational observation window, containment ledger, visible transport rows, continuation rows, package provenance rows, and local certificate rows as the common refinement.

Finally compare the downstream handoff surfaces. Apply Definition 358.6 for the mesh-to-enclosure handoff rows, and apply Definition 357.15 for the validated enclosure boundary. These two boundaries name the same rational interval cell, endpoint-radius-order ledger, observation window, real-readback token, transport rows, continuation rows, package provenance, and local certificate rows already obtained from the dyadic common refinement.

All cited boundaries are finite BEDC readback statements. Repacking their shared rows gives the single $\mathsf{Real}^{\uparrow}$ surface listed in the theorem statement, and no cited component supplies completed real equality, host interval equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limit. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier common refinement enclosure certificate`

Theorem 335.40 (DyadicApproximation refinement-seal commutation)**.** *Let $A = (n, d, w, l, p)$ and $A' = (n', d', w', l', p')$ be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets admitting a displayed common precision refinement R. Let $M = (\ell, J, I, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *cell row, and let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, q, m)$ be an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row consuming the same refined endpoint selected by R.*

Then the route that first takes the common refinement and then reads the $\mathsf{Real}^{\uparrow}$ *seal, and the route that first reads the mesh enclosure and then performs validated readback, expose one finite* $\mathsf{Real}^{\uparrow}$ *surface. That surface is exactly the displayed precision row, transported dyadic endpoint, rational interval cell, endpoint-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window,* $\mathsf{Real}^{\uparrow}$ *seal token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *rows.*

The commutation is only this finite consumer-surface equality. It does not assert completed real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limits.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier refinement seal commutation`

Proof. Begin with the shared refinement endpoint. Apply Definition 335.39 to the accepted packets A and A', the displayed refinement R, the accepted mesh cell M, and the accepted validated enclosure V. This places the precision row, transported endpoint, rational interval cell, endpoint-radius ledger, regular-rational window, real seal token, transport rows, continuation rows, package rows, and local certificate rows on one finite handoff surface.

Next compare the two consumer routes through that surface. Apply Definition 335.34. It identifies the terminal dyadic-window route with the mesh-enclosure and validated-readback route without adding any coordinate beyond the displayed finite rows.

Now seal the terminal endpoint against the mesh enclosure. Use Definition 335.33 to transport the terminal $\mathsf{RegSeqRat}^{\uparrow}$ window and the $\mathsf{Real}^{\uparrow}$ seal token through the same rational interval cell and endpoint-radius ledger.

Finally apply Definition 335.37. This restricts every mesh-cell and validated-enclosure read to the finite packet boundary already produced by the common-refinement handoff. Repacking the shared rows gives the stated single $\mathsf{Real}^{\uparrow}$ surface, and the cited finite boundaries provide no completed real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limit. □

Theorem 335.41 (DyadicApproximation shared Real-boundary determinacy)**.** *Let A and A' be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets. Suppose the two packets display the same precision row, transported dyadic endpoint, rational-interval cell,* $\mathsf{DyadicMesh}^{\uparrow}$ *level and index rows,* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback window,* $\mathsf{Real}^{\uparrow}$ *readback token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *rows.*

Then every mesh-cell consumer, validated-enclosure consumer, and terminal dyadic finite-window consumer that reads either packet through the displayed BEDC rows reads the same finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *boundary surface. The surface is exactly the shared precision row, transported endpoint, rational-interval cell, mesh level, mesh index, containment ledger, regular-rational window, real-readback token, visible transport rows, continuation routes, package provenance, and local name-certificate rows.*

The determinacy is finite. It does not assert completed real equality, quotient-real equality, solver correctness, floating-point semantics, or classical completeness.

Proof. First compare the two dyadic packets before any downstream consumer is read. Apply Definition 335.31 to A and A'. The hypotheses identify precisely the displayed precision, endpoint, regular-rational, real-readback, mesh-and-validated handoff, transport, continuation, provenance, and local certificate rows available to both packets.

Next put each packet together with its mesh cell and validated enclosure in normal form. Apply Definition 335.29 to the accepted packet, the displayed mesh cell, and the validated enclosure over the same rational-interval cell and containment ledger. The normal form has no coordinate beyond the rows listed in the statement.

Now identify the two routes that consumers may use. Apply Definition 335.34 to compare the terminal dyadic finite-window route with the route through the mesh cell and validated enclosure. Apply Definition 335.35 to expose the shared Real-boundary surface carried by that comparison.

Finally restrict mesh and enclosure reads to the finite packet boundary. Apply Definition 335.37. Its factorization says that every accepted mesh-cell or validated-enclosure read passes through the same finite $\mathsf{Real}^{\uparrow}$ surface already identified above.

Combining the refined packet determinacy, the three-packet normal form, route commutation, shared Real-boundary comparison, and finite mesh-enclosure exactness leaves a single displayed BEDC surface for every listed consumer. Since each cited boundary is a finite packet boundary, the conclusion exports no completed real equality, quotient-real equality, solver correctness, floating-point semantics, or classical completeness. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier shared real boundary determinacy`

Theorem 335.42 (DyadicApproximation finite-packet Real-seal stability)**.** *Let A and A' be accepted finite* $\mathsf{DyadicApproximation}^{\uparrow}$ *packets sharing a displayed common refinement R. Let M be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *cell, and let V be an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure over the same rational interval selected by the refined dyadic endpoint. Suppose the visible* hsame *transports,* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows identify the finite-packet route, the mesh-cell route, the validated-enclosure route, and the* $\mathsf{Real}^{\uparrow}$ *seal route over that common refinement.*

Then the terminal finite-packet readback and the $\mathsf{Real}^{\uparrow}$ *seal expose the same finite surface after refinement, whether the consumer reaches it through the common-refinement route or through the validated enclosure route. This surface is exactly the precision row, transported dyadic endpoint,* $\mathsf{RationalInterval}^{\uparrow}$ *cell,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal token,* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment row, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *rows.*

The stability assertion is only this finite seal surface. It does not assert completed real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limits.

Proof. First put the terminal readback in its finite normal form. Apply Definition 335.38. This identifies the terminal precision, transported endpoint, stream-name window, regular-rational readback, $\mathsf{Real}^{\uparrow}$ seal row, transport rows, continuation routes, package provenance, and local name-certificate rows visible to the finite packet.

Next compare the common-refinement route with the validated-enclosure route. Apply Definition 335.40 to the accepted packets, the displayed refinement, the accepted mesh cell, and the accepted validated enclosure. This places the rational interval cell, containment row, refined endpoint, $\mathsf{RegSeqRat}^{\uparrow}$ observation window, and $\mathsf{Real}^{\uparrow}$ seal token on one finite consumer surface.

Finally use Definition 335.41. The shared boundary theorem says that every mesh-cell, validated-enclosure, and terminal finite-window read with the displayed rows reads the same $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary. Repacking those rows gives exactly the surface stated above, and the cited finite boundaries provide no coordinate for completed real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limits. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier finite packet real seal stability`

Theorem 335.43 (DyadicApproximation terminal seal common-refinement pullback)**.** *Let $A = (n, d, w, l, p)$ and $A' = (n', d', w', l', p')$ be accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets. Let terminal finite-window* $\mathsf{Real}^{\uparrow}$ *seal consumers read the visible terminal boundary of A and A' through displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *windows,* $\mathsf{Real}^{\uparrow}$ *tokens, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *rows.*

Then equality of the visible terminal seal boundary is equivalent to the existence of a common-refinement enclosure handoff through one finite BHist *surface. The witnessing surface consists exactly of a common precision row, transported dyadic endpoint, terminal finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *token,* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh row,* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *rows.*

Under this equivalence the forward implication starts from the terminal $\mathsf{Real}^{\uparrow}$ *seal boundary and produces the common finite surface; the reverse implication starts from the common-refinement enclosure handoff and reads back the same terminal seal boundary. The criterion exports no completed real equality, quotient-real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limit.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal seal common refinement pullback`

Proof. For the forward direction, first put the two terminal seal consumers into the finite-packet readback form. Apply Definition 335.38 to the terminal packet read by each consumer. The visible terminal boundary therefore has only the displayed precision, endpoint, terminal-window, regular-rational, real-token, transport, continuation, package, and local certificate rows.

Now use the assumed equality of the visible terminal $\mathsf{Real}^{\uparrow}$ seal boundary. Apply Definition 335.41 to the two accepted packets with those identified terminal rows. The determinacy statement says that every terminal finite-window consumer, mesh-cell consumer, and validated-enclosure consumer must read the same finite BEDC boundary.

Choose the resulting mesh and enclosure rows as the common-refinement witness. Their shared precision row, transported endpoint, terminal window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ token, visible transports, continuation routes, package rows, and local name-certificate rows form the required finite BHist surface.

For the reverse direction, begin with the displayed common-refinement enclosure handoff. Apply Definition 335.39 to A, A', the common refinement, the mesh row, and the validated enclosure row. This places the dyadic endpoint, mesh cell, containment ledger, regular-rational window, real token, transport rows, continuation rows, package rows, and local certificate rows on one finite surface.

Next restrict the handoff to the finite packet boundary. Apply Definition 335.37. Then apply Definition 335.34 to identify the terminal finite-window route with the mesh-enclosure and validated-readback route.

The two routes now expose the same finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ rows, with the same hsame, Cont, Pkg, and local $\mathsf{NameCert}$ coordinates. Reading those rows through the terminal seal consumers gives equality of the visible terminal seal boundary. Since every cited component is a finite BEDC boundary statement, the pullback supplies no completed real equality, quotient-real equality, solver correctness, floating-point semantics, infinite mesh selection, or choice-selected limit. □

Theorem 335.44 (DyadicApproximation diagonal mesh-selector boundary)**.** *Let A be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet, let M be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *finite cell, and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet. Suppose these packets display the same unary precision row, selected diagonal window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, rational-interval cell,* $\mathsf{Real}^{\uparrow}$ *readback token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *rows.*

Then the diagonal-completion consumer of D and the mesh or validated dyadic consumer of A and M read the same finite $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *boundary surface. The shared surface is exactly the displayed precision row, selected diagonal window, regular-rational readback, rational-interval cell, real readback token, visible transport rows, continuation routes, package provenance, and local name-certificate rows.*

This boundary theorem does not identify quotient streams, host real numbers, selected limits, infinite meshes, solver correctness, or classical completeness.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier diagonal mesh selector boundary`

Proof. Start with the dyadic side. Apply Definition 335.41 to the accepted dyadic packet and its displayed mesh and validated-readback rows. This reduces every dyadic finite-window, mesh-cell, and validated-enclosure read to the listed $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary surface.

Next read the mesh packet through Definition 358.8. The bridge shows that the standard finite mesh consumer has no coordinate beyond the unary level, finite cell index, rational interval, endpoint and real readback, transport, continuation, package, and local certificate rows already listed in the statement.

Now turn to the diagonal-completion packet. Apply Definition 359.5 to factor the consumer read through the displayed modulus row, diagonal selector, scheduled window, and regular-rational readback. Then apply Definition 359.6 to identify the visible $\mathsf{Real}^{\uparrow}$ seal ledger with those extracted finite windows and their readback ledger.

The hypotheses identify the precision, selected window, rational interval, regular-rational readback, real token, transports, continuations, package provenance, and local certificates on both routes. Therefore both consumers project the same finite boundary surface. Since every cited step is a finite BEDC packet boundary, none of the excluded analytic or quotient objects is exported. □

Theorem 335.45 (DyadicApproximation terminal selector determinacy)**.** *Let A and A' be accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets. Suppose their terminal rows are identified by displayed* hsame *comparisons, they expose the same* $\mathsf{RationalInterval}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the same* $\mathsf{Real}^{\uparrow}$ *seal-facing route through visible* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *provenance.*

Then the terminal selector reads of A and A' classify together. Equivalently, after the terminal finite-window surface is fixed, the selector cannot depend on an unlogged mesh endpoint or on any endpoint coordinate outside the accepted finite packets.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier finite packet terminal selector determinacy`

Proof. First normalize both terminal reads. Apply Definition 335.38 to each accepted packet. The terminal selector is thereby expressed through the terminal precision row, transported endpoint, $\mathsf{RationalInterval}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal ledger, visible transports, continuation routes, package provenance, and local certificate rows.

Next place the two packets on a common enclosure surface. Apply Definition 335.39. The shared rational interval, regular-rational readback, and real seal route are therefore read from one finite $\mathsf{Real}^{\uparrow}$ boundary surface.

Now commute the refinement with the seal route. Apply Definition 335.40. This shows that reading the terminal selector before the seal and reading through the common refinement before the seal expose the same finite coordinates.

Finally use Definition 335.41. Since the hypotheses identify the terminal rows, interval window, regular readback, real seal route, transports, continuations, and provenance, every permitted consumer route reads the same boundary. No cited surface contains an unlogged mesh endpoint, so the terminal selector has no remaining coordinate on which a different classification could depend. □

Theorem 335.46 (DyadicApproximation terminal diagonal completion handoff)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet. Suppose the terminal route of T and the diagonal row of D display the same terminal precision, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, transported dyadic endpoint,* $\mathsf{Real}^{\uparrow}$ *seal ledger, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *provenance.*

Then the $\mathsf{Real}^{\uparrow}$ *consumer route through T and the* $\mathsf{Real}^{\uparrow}$ *consumer route through D expose the same finite surface: terminal precision, transported dyadic endpoint, scheduled stream win-*

dow, regular-rational readback, $\mathsf{Real}^{\uparrow}$ *seal ledger, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *provenance.*

The handoff is only this finite BEDC surface. It does not assert completed real equality, quotient streams, selected host limits, solver correctness, or choice-selected completion points.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal diagonal completion handoff`

Proof. First normalize the terminal packet route. Apply Definition 335.38 to T. The terminal consumer is thereby read through the displayed terminal precision, transported endpoint, scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal row, transport rows, continuation routes, package provenance, and local dyadic certificate.

Next normalize the diagonal-completion route. Apply Definition 359.5 to D to expose the diagonal selector, scheduled window, and regular-rational readback selected by the requested row. Apply Definition 359.6 to factor the $\mathsf{Real}^{\uparrow}$ seal ledger through those same extracted finite windows and readback rows.

The hypotheses identify the scheduled window and readback rows on the two routes, together with the terminal precision, transported endpoint, seal ledger, transports, continuations, package provenance, and local certificates. Repacking those identified coordinates gives one shared finite surface for both $\mathsf{Real}^{\uparrow}$ consumers.

The cited dyadic and diagonal boundaries expose only finite packet data. Consequently the repacked surface has no coordinate for completed real equality, quotient streams, selected host limits, solver correctness, or choice-selected completion points. □

↪ variant `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal diagonal completion handoff`

Theorem 335.47 (DyadicApproximation terminal diagonal readback composition)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet. Suppose T has the finite-packet* $\mathsf{Real}^{\uparrow}$ *readback surface of Definition 335.38, and suppose T and D are matched by the terminal diagonal handoff of Definition 335.46.*

Then every $\mathsf{Real}^{\uparrow}$ *consumer route through the matched diagonal-completion packet factors through the same terminal precision, transported dyadic endpoint,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, visible* hsame *rows, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *rows already exposed by the terminal packet. The composition exports no completed real equality, quotient stream, selected host limit, solver correctness, or choice-selected completion point.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal diagonal readback composition`

Proof. First normalize the terminal dyadic packet. The finite-packet readback theorem factors every terminal $\mathsf{Real}^{\uparrow}$ read through the displayed terminal precision, endpoint, stream-name window, regular-rational readback, real seal, transport rows, continuation routes, package provenance, and local dyadic certificate.

Next apply the terminal diagonal handoff. Its hypotheses identify the terminal route of T with the diagonal row of D across the same precision, endpoint, scheduled window, readback, seal ledger, transports, continuations, package provenance, and local certificate rows.

Composing these two surfaces gives one finite route for the diagonal-completion consumer. The readback data first appears in the terminal dyadic packet and is then repacked as the diagonal-completion handoff, so the $\mathsf{Real}^{\uparrow}$ consumer sees the same finite rows in the same terminal precision boundary.

The two cited theorems carry the same non-escape boundary. Therefore the composite surface has no coordinate for completed real equality, quotient streams, selected host limits, solver correctness, or choice-selected completion points. □

Theorem 335.48 (DyadicApproximation terminal interval-mesh readback totality)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet. Every terminal interval-mesh read of T factors through the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *window, the* $\mathsf{DyadicMesh}^{\uparrow}$ *cell selector, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row, the terminal finite-packet readback, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Equivalently, the read is total over the displayed finite packet: its public surface is exactly the terminal precision row, transported dyadic endpoint, rational interval cell, dyadic mesh selector, validated enclosure ledger, scheduled $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *provenance. It exports no host interval, quotient real, or unrecorded completeness datum.*

Proof. First normalize the terminal finite-packet read. Apply Definition 335.38. The terminal packet is read through the displayed precision row, transported endpoint, scheduled stream window, regular-rational readback, $\mathsf{Real}^{\uparrow}$ seal ledger, transport rows, continuation routes, package provenance, and local certificate row.

Next attach the mesh and enclosure coordinates. Apply Definition 335.33 to put the terminal mesh route and the validated enclosure route on the same terminal window. Apply Definition 335.37 to identify the rational interval cell, dyadic mesh selector, endpoint containment ledger, and validated enclosure row with that same finite surface.

Finally pass through the diagonal-completion handoff. By Definition 335.46, the terminal packet and the diagonal route expose one shared $\mathsf{Real}^{\uparrow}$ consumer surface over the displayed terminal rows.

The listed rows exhaust the packet coordinates available to an interval-mesh consumer. Since the cited boundaries contain no host interval, quotient real, or ambient completeness field, the total readback exports none of those data. □

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal interval mesh readback totality`

Theorem 335.49 (DyadicApproximation finite-packet StreamName Real-seal factorization)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet. Suppose the displayed terminal route for T contains its terminal precision, dyadic endpoint, scheduled* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal ledger, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *row, and the diagonal-completion selector rows consumed by D.*

Then the route that first applies the StreamName $\mathsf{Real}^{\uparrow}$ *seal factorization diamond and the route that first applies the* $\mathsf{DyadicApproximation}^{\uparrow}$ *terminal diagonal-completion handoff expose the same finite* BHist *surface. That surface is exactly the StreamName finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, dyadic endpoint rows, terminal precision row,* $\mathsf{Real}^{\uparrow}$ *seal token, diagonal selector row, transport rows, continuation rows, package provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ *provenance.*

The surface assertion is only this finite packet factorization. It does not assert completed real equality, quotient streams, selected host limits, solver correctness, metric completeness, or any standard bridge.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier finite packet streamname real seal factorization`
Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier streamname real seal factorization certificate`

Proof. First normalize the dyadic terminal route. Apply Definition 335.38 to the accepted terminal packet T. This reads the terminal consumer through the displayed terminal precision, dyadic endpoint, scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal row, transport rows, continuation rows, package provenance, and local dyadic certificate row.

Next normalize the refinement and seal route. Apply Definition 335.42 to place the terminal readback and the $\mathsf{Real}^{\uparrow}$ seal on one finite refinement surface. Apply Definition 335.43 to pull that seal surface back to the visible terminal boundary.

Now identify the StreamName source tuple. Apply Definition 38.349 to the scheduled $\mathsf{StreamName}^{\uparrow}$ finite window and its displayed prefix refinement. The diamond leaves exactly the StreamName finite-window row, accepted $\mathsf{RegSeqRat}^{\uparrow}$ source row, dyadic endpoint rows, continuation ledger, endpoint transports, and provenance rows visible to the $\mathsf{Real}^{\uparrow}$ seal.

Finally identify the diagonal-completion consumer rows. Apply Definition 335.46 to the same terminal packet and diagonal-completion packet. The handoff contributes the diagonal selector row and confirms that the terminal precision, endpoint, scheduled window, regular-rational readback, seal ledger, transports, continuations, package rows, and local certificate row are the rows consumed by the diagonal route.

The two normal forms therefore list the same finite coordinates. Repacking those coordinates gives one BHist surface containing exactly the StreamName finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic endpoint rows, terminal precision row, $\mathsf{Real}^{\uparrow}$ seal token, diagonal selector row, transport rows, continuation rows, package provenance, and local $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$ provenance. None of the cited finite surface theorems supplies a coordinate for completed real equality, quotient streams, selected host limits, solver correctness, metric completeness, or a standard bridge. □

Definition 335.50 (DyadicApproximation terminal dyadic-net consumer surface)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet matched to T by the displayed terminal diagonal handoff. The terminal dyadic-net consumer surface of (T, D) is the finite* BHist *surface consisting exactly of the terminal precision row, the transported dyadic endpoint row, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell, the* $\mathsf{DyadicMesh}^{\uparrow}$ *selector row, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal row, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *selector row, the visible* hsame *rows, the visible* Cont *routes, the* Pkg *provenance rows, and the visible* $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$, $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$, *and* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *rows.*

Lean4 (definition) `BEDC.Derived.DyadicApproximationUp.DyadicApproximationTerminalDyadicNetConsumerSurface`

Theorem 335.51 (DyadicApproximation terminal dyadic-net consumption)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet and let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet matched to T by the displayed terminal diagonal handoff. Suppose a terminal dyadic-net*

consumer reads T through any displayed route built from the terminal finite-packet Real$^{\uparrow}$ *readback, the interval-mesh readback, the terminal diagonal readback, and the* StreamName$^{\uparrow}$ *real-seal factorization rows.*

Then the consumer factors through the terminal dyadic-net consumer surface of Definition 335.50. Its observable coordinates are exactly the terminal precision, dyadic endpoint, rational interval cell, DyadicMesh$^{\uparrow}$ *selector,* ValidatedNumerics$^{\uparrow}$ *enclosure,* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal,* RealDiagonalCompletion$^{\uparrow}$ *selector,* hsame*,* Cont*,* Pkg*, and* NameCert *rows listed there. The factorization exports no quotient streams, selected host limits, solver correctness, metric completeness, or host real equality.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal dyadic net consumption`

Proof. First normalize the terminal real readback. Apply Definition 335.38 to the accepted terminal packet T. This exposes the terminal precision, transported endpoint, scheduled stream window, regular-rational readback, Real$^{\uparrow}$ seal ledger, transport rows, continuation routes, package provenance, and local dyadic certificate row.

Next attach the interval net. Apply Definition 335.48. The terminal read is total over the displayed rational interval cell, the DyadicMesh$^{\uparrow}$ selector, the ValidatedNumerics$^{\uparrow}$ enclosure row, and the same terminal finite-packet real boundary.

Now compose with the diagonal consumer. Apply Definition 335.47 to the matched pair (T, D). The diagonal route contributes the RealDiagonalCompletion$^{\uparrow}$ selector and keeps the same precision, endpoint, stream window, regular-rational readback, real seal, transports, continuations, package rows, and local certificate rows.

Finally identify the stream seal source. Apply Definition 335.49. It identifies the scheduled StreamName$^{\uparrow}$ finite-window row, RegSeqRat$^{\uparrow}$ readback, dyadic endpoint rows, terminal precision row, Real$^{\uparrow}$ seal token, diagonal selector row, transport rows, continuation rows, package provenance, and local dyadic certificate provenance as one finite packet surface.

The four normal forms list the same terminal coordinates and no additional coordinate. Repacking those coordinates gives exactly the surface of Definition 335.50. Since the cited rows contain no quotient-stream representative, selected host limit, solver-correctness proof, metric-completeness witness, or host real-equality field, the resulting consumer surface exports none of those data. □

Theorem 335.52 (DyadicApproximation terminal dyadic-net bridge)**.** *Let T be an accepted terminal* DyadicApproximation$^{\uparrow}$ *finite packet, and let D be an accepted* RealDiagonalCompletion$^{\uparrow}$ *packet matched to T by the displayed terminal diagonal handoff. Suppose a bridge-facing consumer reads the matched pair as a finite* Real$^{\uparrow}$ *and completion packet through the terminal dyadic-net consumer route.*

Then the exported bridge-facing surface is exactly the terminal dyadic-net consumer surface of Definition 335.50. It is the same finite packet through terminal precision, transported dyadic endpoint, RationalInterval$^{\uparrow}$ *cell,* DyadicMesh$^{\uparrow}$ *selector,* ValidatedNumerics$^{\uparrow}$ *enclosure,* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal,* RealDiagonalCompletion$^{\uparrow}$ *selector, visible* hsame *rows, visible* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

The bridge exports no quotient streams, selected host limits, solver correctness, metric completeness, or host real equality.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal dyadic net bridge`

Proof. First read the matched pair through the terminal dyadic-net consumer surface. Apply Definition 335.51. This already exposes the terminal precision, dyadic endpoint, rational interval cell, mesh selector, validated enclosure, stream window, regular-rational readback, real seal, diagonal selector, transports, continuation routes, package rows, and local certificates.

Next identify the completion side. Apply Definition 335.46. The matched $\mathsf{RealDiagonalCompletion}^{\uparrow}$ packet reads the same terminal precision, endpoint, scheduled window, regular-rational readback, real seal, transports, continuations, package provenance, and local certificate provenance as the terminal dyadic packet.

Now compose the readback route. Apply Definition 335.47. The diagonal consumer cannot replace the terminal rows by completed real equality or by a selected completion point; it repacks the same finite terminal rows for the completion-facing read.

Finally attach the stream seal source by Definition 335.49. The scheduled $\mathsf{StreamName}^{\uparrow}$ row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic endpoint, terminal precision, $\mathsf{Real}^{\uparrow}$ seal, diagonal selector, transports, continuations, package provenance, and local dyadic certificate provenance are therefore one finite surface.

The four cited surfaces name the same displayed coordinates. Repacking them as the bridge-facing packet gives exactly Definition 335.50. Since none of those surfaces contains a coordinate for quotient streams, selected host limits, solver correctness, metric completeness, or host real equality, the bridge cannot export such data. □

Theorem 335.53 (DyadicApproximation terminal tail-budget exhaustion)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet, let D be an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet matched to T by the terminal diagonal handoff, and let B be a* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *precision request consumed by the same terminal packet. Suppose the displayed transports and continuation routes identify the terminal precision row used by T, the diagonal-completion readback of D, and the finite budget request of B.*

Then the consumed budget factors through exactly the terminal precision row, the transported dyadic endpoint, the $\mathsf{RationalInterval}^{\uparrow}$ *cell, the* $\mathsf{DyadicMesh}^{\uparrow}$ *selector, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal, visible* hsame *rows, visible* Cont *routes,* Pkg *provenance, and local* NameCert *rows already exposed by the terminal packet and its matched diagonal-completion surface.*

The terminal packet exports no completed real equality, quotient stream, selected host limit, solver correctness, metric completeness, or host real equality to the finite budget consumer.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal tail budget exhaustion certificate`

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal tail budget exhaustion`

Proof. First normalize the terminal finite-packet real readback. Apply Definition 335.38. The terminal packet is read through the displayed terminal precision, endpoint, rational interval cell, validated enclosure, stream window, regular-rational readback, real seal, transports, continuation routes, package provenance, and local certificate.

Next attach the terminal dyadic-net route. Apply Definition 335.51. This identifies the dyadic endpoint with the $\mathsf{RationalInterval}^{\uparrow}$ cell and $\mathsf{DyadicMesh}^{\uparrow}$ selector used by the terminal consumer, keeping the validated enclosure and real seal on the same finite surface.

Now consume the precision request. Apply Definition 385.5 to force the $\mathsf{FiniteErrorBudget}^{\uparrow}$ request through the displayed terminal tail ledger. Apply Definition 379.16 to compose that ledger with the $\mathsf{DyadicTailBound}^{\uparrow}$ row recorded by the dyadic packet.

Finally use the diagonal side. The terminal diagonal readback composition of Definition 335.47 keeps the matched RealDiagonalCompletion$^{\uparrow}$ packet on the same terminal precision, endpoint, stream window, readback, real seal, transports, continuations, package provenance, and local certificate rows.

The cited surfaces contain the same terminal rows and no additional consumer channel. Repacking them gives exactly the factorization stated above, and the non-escape clauses from the terminal readback and diagonal-completion surfaces exclude completed real equality, quotient streams, selected host limits, solver correctness, metric completeness, and host real equality. □

Theorem 335.54 (DyadicApproximation terminal tail-budget Real-completion route)**.** *Let T be an accepted terminal* DyadicApproximation$^{\uparrow}$ *finite packet. Suppose T is matched with a* RealDiagonalCompletion$^{\uparrow}$ *packet D, a* FiniteErrorBudget$^{\uparrow}$ *request B, a* DyadicTailBound$^{\uparrow}$ *ledger row, a* StreamName$^{\uparrow}$ *window, a* RegSeqRat$^{\uparrow}$ *readback, and a* Real$^{\uparrow}$ *seal row by the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows of the terminal packet.*

Then the matched packet exposes exactly the terminal budget-to-seal route: the finite budget request is read through the terminal precision row, the transported dyadic endpoint, the RationalInterval$^{\uparrow}$ *cell, the* DyadicMesh$^{\uparrow}$ *selector, the* ValidatedNumerics$^{\uparrow}$ *enclosure, the stream window, the regular-rational readback, the dyadic tail-bound ledger, and the* Real$^{\uparrow}$ *seal before it is presented to the Real-completion consumer surface. No completed real equality, quotient stream, host limit, solver oracle, or metric-completion object is exported by this route.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal tail budget real completion route`

Proof. First normalize the terminal finite packet. The finite-packet Real readback theorem Definition 335.38 places the terminal precision, endpoint, stream window, regular-rational readback, validated enclosure, Real$^{\uparrow}$ seal, transports, continuation routes, package provenance, and local certificate on one finite surface.

Next attach the diagonal-completion side. The terminal diagonal readback composition theorem Definition 335.47 identifies the matched RealDiagonalCompletion$^{\uparrow}$ packet with the same terminal precision, endpoint, stream window, readback, seal, transport, continuation, package, and certificate rows.

Now consume the mesh and budget rows. The terminal dyadic-net consumption theorem Definition 335.51 adds the RationalInterval$^{\uparrow}$ cell and DyadicMesh$^{\uparrow}$ selector, while Definition 335.53 forces the FiniteErrorBudget$^{\uparrow}$ request through the displayed DyadicTailBound$^{\uparrow}$ ledger and the same terminal seal route.

These cited surfaces share the same terminal rows. Repacking them gives precisely the budget-to-seal route stated above, and their non-escape clauses leave no coordinate for completed real equality, quotient streams, host limits, solver oracles, or metric-completion objects. □

Theorem 335.55 (DyadicApproximation DyadicRatCore dependency route)**.** *Every accepted* DyadicApproximation *finite packet consumes the* DyadicRatCore$^{\uparrow}$ *signed-exponent row, scale-window row, and finite-tail row of chapter 291 before it may hand a mesh, validated enclosure, or terminal seal to* Real$^{\uparrow}$*. The route passes through the local finite-packet readback and exactness surfaces of Definition 335.30, Definition 335.31, Definition 335.38, and Definition 335.37; it exports no host rational equality, quotient stream, selected limit, or ambient completion object.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier dyadicratcore dependency route`

Proof. Start with the finite-packet readback exhaustion theorem. It places every accepted DyadicApproximation consumer read on the displayed precision, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint window, stream window, regular-rational readback, $\mathsf{Real}^{\uparrow}$ seal, refinement, overlap, transport, continuation, package, and local certificate rows.

Use the refined finite-packet determinacy and terminal finite-packet Real readback theorems next. These keep the transported precision, endpoint window, validated enclosure, stream window, regular-rational readback, and $\mathsf{Real}^{\uparrow}$ seal on one finite packet surface.

Finally apply the finite mesh-enclosure exactness surface. The mesh or enclosure handed to $\mathsf{Real}^{\uparrow}$ is exact only after the DyadicRatCore signed exponent, scale window, and finite tail have been consumed through the cited sibling chapter. Therefore the dependency route is a substantive DyadicRatCore route rather than a decorative cross-reference. □

Theorem 335.56 (DyadicApproximation terminal budget-seal uniqueness)**.** *Let T_0 and T_1 be accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packets. Suppose both packets are matched to the same terminal precision row, transported dyadic endpoint,* $\mathsf{RationalInterval}^{\uparrow}$ *cell,* $\mathsf{DyadicMesh}^{\uparrow}$ *selector,* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, visible* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Then T_0 and T_1 present the same Real-completion budget-to-seal route to every consumer. In particular, a consumer that enters through the finite budget row, the dyadic endpoint, the mesh selector, the validated enclosure, the stream window, the regular-rational readback, the tail-bound ledger, or the terminal $\mathsf{Real}^{\uparrow}$ *seal sees one terminal route surface.*

This uniqueness statement does not identify completed reals. It refuses completed real equality, quotient streams, selected host limits, solver oracles, metric-completion objects, and host real equality as distinguishing data.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal budget seal uniqueness`

Proof. First normalize each terminal packet through the tail-budget surface. Apply Definition 335.53 to T_0 and to T_1. Since the hypotheses match the same terminal precision row, endpoint, interval cell, mesh selector, validated enclosure, stream window, readback, tail ledger, seal, transports, continuations, package provenance, and local certificate rows, both packets factor through the same finite budget ledger.

Next pass from the finite budget ledger to the Real-completion consumer surface. Apply Definition 335.54. This keeps the budget request, dyadic tail-bound ledger, stream readback, and $\mathsf{Real}^{\uparrow}$ seal on the displayed terminal route before any Real-completion consumer can inspect the packet.

Now compare the finite readbacks. The determinacy theorem Definition 335.31 says that two accepted finite packets with the same transported rows expose the same readback surface. Thus the visible terminal rows carried by T_0 and T_1 cannot split into two consumer-facing budget-to-seal routes.

Finally record the dyadic-core dependency. Apply Definition 335.55 to both packets. The endpoint, mesh, enclosure, and terminal seal are available only through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ signed-exponent, scale-window, and finite-tail rows, so no hidden host rational comparison or ambient completion channel remains.

The four cited surfaces have the same visible rows under the hypotheses. Repacking them gives one terminal Real-completion route, and the non-escape clauses in those surfaces leave no coordinate in which completed real equality, quotient streams, selected host limits, solver oracles, metric-completion objects, or host real equality could distinguish T_0 from T_1 for any consumer. □

DyadicApproximation terminal finite packets are consumed through the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger of chapter 291, the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, and the $\mathsf{RealCauchyCompletion}^{\uparrow}$ seal rows of chapter 360. This records the horizontal route from terminal dyadic observations to the Real-completion consumer as a finite sibling dependency rather than an ambient completion assumption.

Theorem 335.57 (DyadicApproximation L10 Real-completion sibling route)**.** *Let T be an accepted terminal* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet, and let a* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer read the terminal real-facing row of T. Suppose the terminal packet is matched with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows of chapter 291, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal rows of chapter 360 by the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the terminal packet.*

Then every accepted terminal read consumed by that $\mathsf{RealCauchyCompletion}^{\uparrow}$ *surface factors through the sibling chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow}$$

before the completion consumer may inspect the terminal seal. The exported surface is exactly the finite packet route where DyadicRatCore supplies the finite observation and ledger rows, StreamName schedules finite windows over those rows, RegSeqRat supplies regular readback, and RealCauchyCompletion consumes the resulting seal rows together with hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance. It exports no quotient stream, selected host limit, host real equality, solver oracle, ambient completeness principle, or completion object outside the displayed sibling rows.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier l10 real completion sibling route`

Proof. First normalize the terminal Real-completion route. The terminal tail-budget theorem Definition 335.54 places the accepted terminal packet on the displayed budget-to-seal route through terminal precision, transported endpoint, interval cell, mesh selector, validated enclosure, finite tail ledger, and $\mathsf{Real}^{\uparrow}$ seal.

Next attach the diagonal handoff. The theorem Definition 335.46 identifies the terminal route with the diagonal completion packet on the same $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, seal ledger, transport rows, continuation rows, package provenance, and local certificate rows.

Now read the completion consumer through the sibling completion surface. Apply Definition 360.3, Definition 360.7, and Definition 360.8. These RealCauchyCompletion rows force the completion read through the modulus-selected diagonal packet, dyadic observation ledger, finite stream window, regular-rational readback, and real seal.

Finally apply Definition 360.18. Repacking the dyadic terminal route with the RealCauchyCompletion export yields exactly the sibling chain through chapter 291, chapter 126, chapter 301, and chapter 360. The non-escape clauses in the cited terminal, diagonal, and completion theorems exclude quotient streams, selected host limits, host real equality, solver oracles, ambient completeness, and any completion object outside the displayed finite packet rows. □

Theorem 335.58 (DyadicApproximation top-hub L10 sibling route)**.** *Every top-level* $\mathsf{DyadicApproximation}^{\uparrow}$ *read exported through the hub of chapter 335 factors through the same four sibling faces*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow}$$

named by chapter 291, chapter 126, chapter 301, and chapter 360. The hub-visible export is exhausted by the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of those sibling faces together with the local terminal* DyadicApproximation$^{\uparrow}$ *rows. It exports no quotient stream, selected host limit, host real equality, solver oracle, ambient completeness principle, or hidden completion object.*

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier top hub l10 sibling route`

Proof. Begin with the sibling-route theorem Definition 335.57. It already places every terminal RealCauchyCompletion$^{\uparrow}$ read of an accepted DyadicApproximation$^{\uparrow}$ packet on the four-face chain through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and RealCauchyCompletion$^{\uparrow}$.

The terminal tail-budget Real-completion route of Definition 335.54 supplies the terminal budget-to-seal route consumed by the hub. The terminal diagonal completion handoff of Definition 335.46 identifies that terminal route with the same diagonal packet, scheduled window, regular-rational readback, seal ledger, visible hsame transports, visible Cont routes, Pkg provenance, and local NameCert rows.

Finally use the RealCauchyCompletion readback and export rows: Definition 360.3, Definition 360.7, Definition 360.8, and Definition 360.18. These rows force the top-level hub read through the same finite completion route and leave no coordinate for a quotient stream, selected host limit, host real equality, solver oracle, ambient completeness principle, or hidden completion object. □

Theorem 335.59 (DyadicApproximation terminal Real-completion exit criterion)**.** *Let T be an accepted terminal* DyadicApproximation$^{\uparrow}$ *finite packet. A terminal read of T can be admitted for consumption by the* RealCauchyCompletion$^{\uparrow}$ *surface only if the read displays the* DyadicRatCore$^{\uparrow}$ *observation ledger of chapter 291, a finite* StreamName$^{\uparrow}$ *window from chapter 126, the* RegSeqRat$^{\uparrow}$ *regular readback rows of chapter 301, the* RealCauchyCompletion$^{\uparrow}$ *seal rows of chapter 360, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *support rows carried by the terminal packet.*

Every admitted terminal RealCauchyCompletion$^{\uparrow}$ *read must expose those five families of displayed rows and must pass through the four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow}.$$

The support criterion has no substitute coordinate: quotient stream equality, selected host limits, host real equality, solver oracles, ambient completeness, and hidden completion objects cannot replace any displayed face or support row.

Lean4 ✓ `BEDC.Derived.DyadicApproximationUp.DyadicApproximationCarrier terminal real completion exit criterion`

Proof. Apply Definition 335.57 to the admitted terminal consumer read. Its conclusion already factors the read through the DyadicRatCore$^{\uparrow}$ ledger, finite StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, and RealCauchyCompletion$^{\uparrow}$ seal rows, with the visible hsame transports, Cont routes, Pkg provenance, and local NameCert provenance on the same terminal packet. The terminal packet itself supplies the shared BHist support.

The top-hub route of Definition 335.58 and the RealCauchyCompletion readback and export rows of Definition 360.3, Definition 360.7, Definition 360.8, and Definition 360.18 identify this finite support surface. They do not by themselves create an admission converse: the cited rows leave the completion consumer with only the displayed finite faces and support rows, so none of quotient stream equality, selected host limits, host real equality, solver oracles, ambient completeness, or hidden completion objects can witness terminal admission. □

Chapter 336

A Concrete Naming Certificate for $\mathsf{ComputableReal}^{\uparrow}$

The $\mathsf{ComputableReal}^{\uparrow}$ interface names effective real presentations by finite BEDC packets rather than by completed real objects. Following the Bishop–Bridges constructive-analysis treatment of explicit moduli and the Weihrauch computable-analysis treatment of names as effective approximation procedures, the packet below records a dyadic approximation program only through certified finite observation windows, a unary Cauchy modulus, dyadic endpoint rows, a regular-sequence handoff, and a $\mathsf{Real}^{\uparrow}$ seal boundary.

The effective approximation packet consumes chapter 126 for finite observation windows, chapter 292 for unary thresholds, chapter 291 for dyadic approximation rows, chapter 301 for the regularity handoff, and chapter 38 for the seal boundary. These are imported L10 sibling rows for coordinates s, μ, d, r, ℓ, not local free parameters and not host real data.

Definition 336.1 (ComputableReal source packet)**.** *A* $\mathsf{ComputableReal}^{\uparrow}$ *source packet is a finite* BHist *record*

$$X = (s, \mu, d, r, \ell, h, c, p, n)$$

with the following visible rows:

1. *s is a* $\mathsf{StreamName}^{\uparrow}$ *observation schedule naming the certified finite windows through which the approximation program is read;*
2. *μ is a unary* $\mathsf{CauchyModulus}^{\uparrow}$ *row naming, for each requested precision row, the finite threshold after which observations are compared;*
3. *d is the family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets read on the scheduled windows selected by s and μ;*
4. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff ledger built from those dyadic packets and modulus rows;*
5. *ℓ is the* $\mathsf{Real}^{\uparrow}$ *seal-source row that consumes the regular sequence surface without selecting a completed real value;*
6. *h records the componentwise* hsame *rows for stream windows, modulus rows, dyadic packets, regularity rows, and the seal source;*
7. *c records the* Cont *routes through which certified finite windows are read by downstream consumers;*
8. *p is the* Pkg *provenance row naming the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependency certificates;*

9. n is the local $\mathsf{NameCert}_{\mathsf{ComputableReal}^\uparrow}$ *registration row.*

The carrier predicate is $\mathsf{ComputableRealSource}(s, \mu, d, r, \ell, h, c, p, n)$*. Its classifier compares two accepted packets by running the existing regular-sequence classifier on every certified finite window and transporting all displayed rows componentwise by* hsame *and* Cont*. It does not quotient completed reals or appeal to host real equality.*

Theorem 336.2 (ComputableReal NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ComputableReal}^\uparrow}$ *obligation surface consists of five rows:*

1. *carrier habitation: the carrier is inhabited by the constant dyadic stream whose every certified window reads the same* $\mathsf{DyadicRatCore}^\uparrow$ *zero packet, with the unary modulus selecting the first threshold row;*
2. *classifier transport: accepted packets compare only by pointwise* hsame *transport of the displayed* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{CauchyModulus}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{Real}^\uparrow$*,* Cont*,* Pkg*, and* NameCert *rows;*
3. *modulus-window stability: restricting, enlarging, or refining a certified finite window preserves the late-observation comparisons named by the same unary modulus row;*
4. $\mathsf{Real}^\uparrow$ *seal handoff: every real-facing read factors through the* $\mathsf{RegSeqRat}^\uparrow$ *handoff surface and then through the displayed* $\mathsf{Real}^\uparrow$ *seal source row;*
5. *ledger coverage: the packet exposes no row beyond the stream schedule, modulus table, dyadic approximation rows, regularity handoff, seal source, transport, continuation, provenance, and local naming registration.*

These rows are the complete seed obligation for constructive effective real names over certified dyadic approximations.

```
Lean4 ✓  BEDC.Derived.ComputableRealUp.ComputableRealSource namecert obligation surface
Lean4 ✓  BEDC.Derived.ComputableRealUp.ComputableRealPacket namecert obligation surface
```

Proof. For habitation, take the constant scheduled stream whose dyadic observation row is the certified zero packet. The unary modulus can select the first threshold for every precision request, and the regularity handoff reads identical dyadic endpoints on every finite window.

For classifier transport, project the carrier of Definition 336.1. Each visible coordinate is a finite BEDC row, so the classifier is obtained by applying the $\mathsf{RegSeqRat}^\uparrow$ finite-window classifier after transporting the corresponding stream, modulus, dyadic, regularity, seal, continuation, provenance, and local naming rows.

For window stability, compare two certified finite windows by passing to their displayed common refinement. The unary modulus still names the same threshold rows, and the dyadic observations beyond those thresholds are the rows already stored in the packet, so restriction and refinement do not create a new approximation object.

Finally, the $\mathsf{Real}^\uparrow$ read is exactly the seal-source row ℓ fed by the regular-sequence ledger r. Since the carrier has no coordinate for a completed real, a quotient stream, a selected limit, or host real equality, the listed ledger rows exhaust the local naming surface. □

Theorem 336.3 (ComputableReal RealUp handoff)**.** *Every accepted* $\mathsf{ComputableReal}^\uparrow$ *packet presents a* $\mathsf{Real}^\uparrow$ *seal source only through its* $\mathsf{RegSeqRat}^\uparrow$ *regularity handoff, dyadic approximation rows, certified* $\mathsf{StreamName}^\uparrow$ *windows, unary* $\mathsf{CauchyModulus}^\uparrow$ *threshold rows, and displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The handoff exports no completed real carrier, no selected limit, no quotient of Cauchy names, and no appeal to classical real completeness.

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSource realup handoff`
Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealPacket realup handoff`

Proof. Project the source packet of Definition 336.1. The rows s, μ, and d give the scheduled dyadic approximation data, while r is the regular rational handoff built from those finite observations.

Read the seal-source row ℓ. It is indexed by the same regularity handoff and therefore consumes only the finite scheduled windows, dyadic endpoint rows, modulus thresholds, transport rows, continuation routes, and provenance already visible in the carrier.

No additional coordinate remains from which a completed real object or selected limiting value could be read. The $\mathsf{Real}^{\uparrow}$ handoff is therefore a finite source-surface handoff, not a completeness principle. □

Theorem 336.4 (ComputableReal ledger coverage)**.** *Every consumer of an accepted* $\mathsf{ComputableReal}^{\uparrow}$ *packet factors through the same finite ledger:*

$$(s, \mu, d, r, \ell, h, c, p, n).$$

The stream schedule, modulus table, dyadic approximation rows, regularity handoff, seal source, transport rows, continuation routes, package provenance, and local naming registration cover all readable coordinates of the packet.

Consequently the packet exports no oracle beyond its certified observation windows, no real-number quotient, no metric completion object, and no hidden standard bridge.

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket ledger coverage`

Proof. Start from the carrier definition. Its tuple lists exactly nine coordinates, and each coordinate is a finite row over the existing BEDC kernel and certificate interfaces.

Any stream-facing consumer reads s and the certified windows named by c. Any rate-facing consumer reads μ. Any approximation-facing consumer reads d. Any regular-sequence or real-facing consumer reads r and ℓ, with h, p, and n carrying the transport and provenance required to keep those reads inside the same packet.

Thus every consumer projection is one of the displayed coordinates or a projection through the listed transport and continuation rows. Since no other coordinate exists, external oracles, quotient reals, completion objects, and standard bridge data are outside the ledger. □

Theorem 336.5 (ComputableReal standard bridge boundary)**.** *The standard mathematical reading of an accepted* $\mathsf{ComputableReal}^{\uparrow}$ *packet is the constructive effective real name consisting of an explicit dyadic approximation procedure and a unary convergence modulus. The BEDC bridge boundary is exactly the finite source surface of Definition 336.1 together with the obligation rows of Definition 336.2, Definition 336.3, and Definition 336.4.*

This bridge boundary is compatible with Bishop–Bridges constructive reals and Weihrauch-style computable names, but it does not assert classical real completeness, a quotient of names, extensional equality of completed reals, or a total comparison operation.

Proof. Read the accepted packet as the finite program interface encoded by s, μ, and d: certified windows select dyadic observations, and the unary modulus selects the finite thresholds at which precision requests are checked.

Apply Definition 336.2 to identify the carrier, pointwise classifier, modulus-window stability, seal handoff, and ledger coverage rows. Apply Definition 336.3 to identify the only real-facing read, and apply Definition 336.4 to rule out unlisted consumer coordinates.

The resulting bridge boundary is therefore the standard effective real-name surface interpreted through finite BEDC rows. The external precedents explain the intended mathematical reading, while the BEDC packet itself retains only the displayed approximation, modulus, regularity, seal, transport, and provenance rows. □

Theorem 336.6 (ComputableReal RegSeqRat seal factorization)**.** *Every accepted packet of Definition 336.1 factors its real-facing read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, unary* $\mathsf{CauchyModulus}^{\uparrow}$ *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff,* $\mathsf{Real}^{\uparrow}$ *seal-source row,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The factorization exports no completed real, quotient of names, selected limit, host equality, or classical completeness row. Its full readable surface is the finite BHist *ledger*

$$(s, \mu, d, r, \ell, h, c, p, n).$$

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket regseqrat seal factorization`

Proof. Project the nine coordinates of the source packet. The rows s, μ, and d supply the certified stream windows, unary thresholds, and dyadic observations. The row r is the regular-sequence handoff, while ℓ is the only $\mathsf{Real}^{\uparrow}$ seal-source row.

Apply Definition 336.3 to the real-facing read. It identifies the $\mathsf{Real}^{\uparrow}$ surface as a read through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the same finite dyadic windows, with h, c, p, and n carrying transport, continuation, package, and naming provenance.

Apply Definition 336.4 to exhaust the consumer coordinates. Then apply Definition 336.5 to separate this finite effective-real reading from completed reals, quotient names, selected limits, host equality, and classical completeness. □

Theorem 336.7 (ComputableReal RationalStream–RegSeqRat schedule factorization)**.** *Every accepted packet of Definition 336.1 whose schedule row is read as a finite* $\mathsf{RationalStream}^{\uparrow}$ *observation packet factors its* $\mathsf{Real}^{\uparrow}$*-facing read through the* $\mathsf{RationalStream}^{\uparrow}$ *pointwise* $\mathsf{Rat}^{\uparrow}$ *obligation rows, the* $\mathsf{RationalStream}^{\uparrow}$ *to* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{ComputableReal}^{\uparrow}$ $\mathsf{RegSeqRat}^{\uparrow}$ *seal factorization, and the* $\mathsf{ComputableReal}^{\uparrow}$ *ledger coverage theorem.*

The exported consumer surface consists exactly of the $\mathsf{StreamName}^{\uparrow}$ *schedule row, unary-history index rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, the* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

The factorization exports no quotient Cauchy class, selected limit, host real equality, or classical completeness principle.

Proof. Project the source packet of Definition 336.1. Its schedule row supplies the finite $\mathsf{StreamName}^{\uparrow}$ windows, and the assumed $\mathsf{RationalStream}^{\uparrow}$ reading interprets those windows as finite rational observations with unary-history indices and pointwise $\mathsf{Rat}^{\uparrow}$ obligation rows.

Use the $\mathsf{RationalStream}^{\uparrow}$ carrier and handoff rows to pass from the scheduled rational observations to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ handoff. The $\mathsf{CauchyModulus}^{\uparrow}$ thresholds select the late comparison rows, while the $\mathsf{DyadicRatCore}^{\uparrow}$ observations remain the finite approximation rows visible in the computable-real packet.

Apply Definition 336.6 to factor the real-facing read through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal row. Apply Definition 336.3 to identify the resulting seal read as the only $\mathsf{Real}^{\uparrow}$ consumer surface.

Finally apply Definition 336.4. The repacked rows are precisely the schedule, unary-history index, pointwise rational, threshold, dyadic observation, regular-sequence handoff, seal, transport, continuation, package, and naming rows listed in the statement, with no coordinate left for quotient Cauchy classes, selected limits, host real equality, or classical completeness. □

Theorem 336.8 (ComputableReal Real seal refinement coverage)**.** *Restricting or refining a certified finite observation window of an accepted* $\mathsf{ComputableReal}^{\uparrow}$ *packet preserves the same* $\mathsf{Real}^{\uparrow}$ *seal-source surface. The unary modulus row selects the same late-observation comparison boundary, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is read through the same finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*

Consequently every consumer read after the window change remains inside the finite BEDC rows

$$(s, \mu, d, r, \ell, h, c, p, n)$$

of Definition 336.1.

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket real seal refinement coverage`

Proof. Compare the original certified window and the restricted or refined window by passing to their displayed common refinement. The source packet coordinates s, μ, d, r, ℓ, h, c, p, and n remain the rows being read; no new carrier or limiting object is introduced by the window change.

The row μ names the same late-observation threshold, and the stream row s names the finite window through which the relevant dyadic observations in d are selected. The regular-sequence handoff r therefore receives the same kind of finite comparison data before and after refinement.

Apply Definition 336.6 to keep the real-facing read factored through r and ℓ. Apply Definition 336.3 and Definition 336.4 to keep the seal read and all consumer projections inside the finite BEDC ledger. □

Theorem 336.9 (ComputableReal classifier transport)**.** *For two accepted source packets of Definition 336.1, componentwise* hsame *transport of the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, Cont, *and* Pkg *rows preserves the* $\mathsf{ComputableReal}^{\uparrow}$ *classifier. The comparison remains a finite-window regular-sequence comparison and never becomes equality of completed real values.*

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket classifier transport`

Proof. Project the tuple $X = (s, \mu, d, r, \ell, h, c, p, n)$ from Definition 336.1. Each coordinate is a displayed BEDC row, and the classifier described there reads only the scheduled finite windows and the regular-sequence handoff built from those windows.

Transport s, μ, d, r, ℓ, c, and p componentwise by the given hsame rows. The $\mathsf{StreamName}^{\uparrow}$ row keeps the same certified window shape, the $\mathsf{CauchyModulus}^{\uparrow}$ row keeps the same unary thresholds, and the $\mathsf{DyadicRatCore}^{\uparrow}$ rows keep the same dyadic endpoints after transport.

The $\mathsf{RegSeqRat}^{\uparrow}$ classifier is then applied on the transported finite windows. The $\mathsf{Real}^{\uparrow}$ seal row is carried only as the displayed handoff target, so completed-real equality, quotient names, and selected limits remain outside the ledger. □

Theorem 336.10 (ComputableReal modulus-window stability)**.** *Let X be an accepted* $\mathsf{ComputableReal}^{\uparrow}$ *source packet. Restrict, enlarge, or refine a certified finite observation window along the* $\mathsf{StreamName}^{\uparrow}$

schedule while retaining the same unary $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row. Then the late-window comparison rows read from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet family and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff remain accepted by the same source packet.*

Proof. Read the observation schedule s and modulus row μ from Definition 336.1. A certified window is visible only through the $\mathsf{StreamName}^{\uparrow}$ row and the Cont routes that read dyadic observations from that finite window.

For a restriction, project the retained dyadic rows and the same modulus threshold. For an enlargement or refinement, pass through the displayed common finite window named by the stream schedule. In each case, the Cont route still reads the dyadic endpoints from d, and the unary threshold remains the row named by μ.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff consumes exactly those transported finite-window comparisons. Since the packet has no coordinate for a completed real, selected limit, or host metric completion, the modulus-window stability claim remains inside the displayed ledger. □

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket modulus window stability`

Theorem 336.11 (ComputableReal bridge boundary)**.** *The standard effective-real boundary of an accepted* $\mathsf{ComputableReal}^{\uparrow}$ *packet is the finite BEDC surface consisting of the source packet of Definition 336.1, the classifier transport row of Definition 336.9, the modulus-window row of Definition 336.10, the* $\mathsf{Real}^{\uparrow}$ *handoff, and the ledger-coverage row. The boundary consumes the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *dependency certificates together with* BHist*,* hsame*,* Cont*, and* Pkg *rows.*

It does not assert extensional equality of completed real values, a selected limit, a quotient of names, or a classical completeness principle.

Proof. Start with the source packet and project its visible dependency rows. The stream schedule, modulus row, dyadic packet family, regular-sequence handoff, and real seal form the whole carrier surface.

Use Definition 336.9 for componentwise transport of the classifier and Definition 336.10 for certified window changes. Then use Definition 336.3 and Definition 336.4 to identify the only real-facing read and the exact consumer ledger.

These projections give the standard effective-real reading through finite BEDC rows. No displayed coordinate names completed-real equality, selected limits, quotient reals, or host completeness, so those objects stay outside the bridge boundary. □

Theorem 336.12 (ComputableReal public export)**.** *The source packet,* $\mathsf{NameCert}_{\mathsf{ComputableReal}^{\uparrow}}$ *obligation surface, RealUp handoff, ledger coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal factorization, Real seal refinement coverage, classifier transport, modulus-window stability, and bridge boundary form the public computable-real export over* BHist *schedules. Every downstream consumer may read only the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows.*

The export exposes no quotient real equality, no selected limit, no classical completeness principle, and no standard bridge datum outside the finite ledger.

Proof. Begin with the source packet of Definition 336.1. Its displayed coordinates are the stream schedule, unary modulus row, dyadic packet family, regular-sequence seal, real-facing handoff, and package provenance rows.

Use Definition 336.2 to name the obligation surface and Definition 336.9 to transport the classifier across the visible coordinates. Use Definition 336.10 to keep finite windows stable under restriction, enlargement, and refinement.

Apply Definition 336.3, Definition 336.4, Definition 336.6, and Definition 336.8 to identify the real-facing exit, the consumer ledger, the regular-sequence seal factor, and the Real seal refinement coverage.

Finally, Definition 336.11 records the boundary objects not projected by the effective-real surface. Thus the theorem exports a consumer packet for later Real and modulus chapters, not a Real completeness theorem and not a standard bridge. □

Theorem 336.13 (ComputableReal effective-window determinacy)**.** *Let* $(s, \mu, d, r, \ell, h, c, p, n)$ *and* $(s', \mu', d', r', \ell', h', c', p', n')$ *be accepted* $\mathsf{ComputableReal}^{\uparrow}$ *source packets. Suppose their consumer reads display the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows, unary* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row,* $\mathsf{Real}^{\uparrow}$ *seal-source row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *provenance. Then the two reads expose one consumer surface for every effective-window coordinate.*

This determinacy is only a statement about the displayed finite BEDC surface. It does not assert completed real equality, identify quotient names, select a limit, or invoke classical completeness.

Lean4 ✓ `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket effective window determinacy`

Proof. Project both accepted packets through Definition 336.1. The common display fixes the stream windows, modulus thresholds, dyadic endpoints, regular-sequence handoff, real seal-source row, and all transport and provenance coordinates.

Apply Definition 336.6 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ handoff coordinate in both reads. Apply Definition 336.8 to place the $\mathsf{Real}^{\uparrow}$ seal-source row over the same refinement surface.

The bridge boundary of Definition 336.11 removes any additional completed-real coordinate from the consumer read. It leaves only the finite rows listed in the ledger.

Finally use Definition 336.4. Since the ledger exhausts the consumer coordinates and every ledger row agrees by hypothesis, the two reads present the same effective-window consumer surface, with no extra channel for quotient names, selected limits, or classical completeness. □

Theorem 336.14 (ComputableReal effective RegSeqRat modulus package)**.** *Every accepted* $\mathsf{ComputableReal}^{\uparrow}$ *packet whose displayed finite windows agree under* hsame *and whose unary* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row is retained determines a single finite BEDC consumer package. The package consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *observation window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal-source row, the displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance.*

The same package is stable under restriction, enlargement, and common refinement of certified windows. It asserts no completed-real equality, quotient name, selected limit, trichotomy, or classical completeness principle.

Proof. Project the accepted source tuple through Definition 336.1. This yields the displayed stream schedule, unary modulus threshold, dyadic endpoint family, regular-sequence handoff, Real seal source, transport routes, package provenance, and local name provenance.

Use Definition 336.10 for restriction, enlargement, and common refinement of certified windows. The retained unary threshold row stays on the same displayed carrier, and the hsame agreement of finite windows is transported componentwise by Definition 336.9.

The $\mathsf{RegSeqRat}^{\uparrow}$ coordinate is then fixed by Definition 336.6. The $\mathsf{Real}^{\uparrow}$ seal-source row is fixed by Definition 336.8 together with Definition 336.3.

Finally apply Definition 336.4 and Definition 336.13. The ledger lists exactly the consumer-visible coordinates, and determinacy identifies the common finite-window read as one package. Since no listed coordinate is a completed real, quotient name, selected limit, trichotomy witness, or classical completeness datum, the asserted package contains none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableReal}^{\uparrow}}$.

Constructive story: A computable real source is generated by certified finite stream-observation windows, a unary Cauchy modulus, dyadic approximation packets, a regular-sequence handoff, a RealUp seal source, and visible transport and provenance rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ComputableReal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 336.5, Definition 336.11, Definition 336.12, Definition 336.13, and Definition 336.14 record the finite effective-real bridge surface, public export, and downstream effective-name consumer evidence over the source packet, NameCert obligation surface, classifier transport, modulus window, $\mathsf{Real}^{\uparrow}$ handoff, ledger coverage, BHist, hsame, Cont, Pkg, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ComputableReal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ComputableRealUp.ComputableRealSourcePacket\_effective\_window\_determinacy`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close classical real completeness, quotient equality of names, total comparison of completed reals, selected limits, or a machine-checked standard bridge.

Upgrade path: No further paper-axis theory upgrade remains; formal theorem checking and bridge checking remain on the verification axis.

Chapter 337

A Concrete Naming Certificate for $\mathsf{LagrangianMechanics}^{\uparrow}$

The $\mathsf{LagrangianMechanics}^{\uparrow}$ interface names the finite mechanics surface whose action ledger can be consumed by Euler–Lagrange and Noether-facing certificates. Following Arnold's *Mathematical Methods of Classical Mechanics* and Abraham–Marsden's *Foundations of Mechanics*, the standard mathematical story starts from configurations, velocities, an action functional, admissible variations, endpoint policy, Euler–Lagrange equations, symplectic flow, and conserved-current reads. The BEDC packet below records only the finite rows needed to cite that surface.

The symplectic-flow row S is read through the sibling surface of chapter 138, and the Noether-current row J is read through the sibling surface of chapter 269. Those rows are dependency surfaces of the finite packet, not hidden global solution or conservation-law principles.

Definition 337.1 (LagrangianMechanicsUp BHist carrier)**.** *A* $\mathsf{LagrangianMechanics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, V, A, R, E, G, S, J, H, C, P, N)$$

with the following visible rows:

1. *Q is the configuration row naming the finite position or field entries available to the local mechanics packet;*
2. *V is the velocity row naming the displayed tangent or rate entries paired with Q;*
3. *A is the action-functional row recording the finite Lagrangian expression, integration route, scalar read, and dependency certificates used to evaluate action values;*
4. *R is the variation row naming the admissible finite variation family and the routes through which the first action variation is read;*
5. *E is the endpoint row recording fixed, vanishing, or exposed endpoint terms;*
6. *G is the Euler–Lagrange row that receives the residual equation surface obtained from A, R, and E;*
7. *S is the symplectic-flow row naming the finite phase-flow read available after the action ledger has selected the displayed dynamics;*
8. *J is the Noether-current row naming the finite conserved-current read available to a Noether-facing consumer;*

9. *H records the* hsame *provenance rows for configuration, velocity, action, variation, endpoint, Euler–Lagrange, symplectic-flow, and Noether-current data;*
10. *C records the* Cont *provenance routes by which the displayed rows are read;*
11. *P is the* Pkg *provenance row naming the action, variation, Euler–Lagrange, symplectic, and Noether dependency certificates used by the packet;*
12. *N is the* NameCert *provenance row registering the packet under* $\mathsf{LagrangianMechanics}^{\uparrow}$.

The carrier predicate is $\mathsf{LagrangianMechanicsPacket}(Q, V, A, R, E, G, S, J, H, C, P, N)$. *Two accepted packets classify together only when the twelve displayed rows transport componentwise by the* hsame *witnesses and the same* Cont *routes named in C and P.*

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.TasteGate.LagrangianMechanicsTasteGate single carrier alignment`

Theorem 337.2 (LagrangianMechanicsUp action ledger obligation)**.** *Every accepted* $\mathsf{LagrangianMechanics}^{\uparrow}$ *carrier exposes a finite action ledger consisting exactly of the configuration row Q, velocity row V, action-functional row A, variation row R, endpoint row E, and the displayed* hsame, Cont, Pkg, *and* NameCert *provenance rows that connect them. A consumer may read the Lagrangian expression, scalar action value, admissible variation family, and endpoint policy only through those rows.*

The ledger asserts no minimizer existence, no global solution theory, no ambient smooth manifold of curves, and no unrecorded regularity condition.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket action ledger obligation`

Proof. Project the carrier of Definition 337.1. The coordinates Q, V, A, R, and E are the only rows that store the displayed configuration, velocity, action, variation, and endpoint data.

The transport and continuation coordinates H and C specify how these rows move under hsame and how they are read by consumers. The package and name certificate rows P and N record provenance, but they do not add analytic objects beyond the visible packet.

Thus the action ledger is exhausted by the listed finite rows. Any minimizer, global dynamics theorem, smooth host manifold, or hidden regularity principle would require an additional coordinate absent from the carrier. □

Theorem 337.3 (LagrangianMechanicsUp Euler-Lagrange interface)**.** *For an accepted carrier $L = (Q, V, A, R, E, G, S, J, H, C, P, N)$, the Euler–Lagrange-facing interface is the single residual-equation row G together with the action, variation, endpoint, transport, continuation, and provenance rows that feed it. An* $\mathsf{EulerLagrange}^{\uparrow}$ *consumer may read only the equation surface displayed by G and the finite action ledger named in Definition 337.2.*

The interface does not claim variational completeness, integration by parts in a host calculus, boundary-condition completeness, or equivalence with a standard analytic equation model.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket eulerlagrange interface`

Proof. Start from the action ledger supplied by Definition 337.2. The variation row R and endpoint row E determine which first-variation terms remain visible to the packet.

The coordinate G records the residual equation surface obtained from those rows. Since G is a finite row of the same carrier, its reads are constrained by the displayed hsame and Cont provenance in H and C.

No coordinate stores an ambient variational calculus, a boundary completeness principle, or a standard analytic equation model. Therefore the Euler–Lagrange-facing interface is exactly the finite row G together with the action ledger that feeds it. □

Theorem 337.4 (LagrangianMechanicsUp Noether consumer boundary)**.** *A Noether-facing consumer of an accepted* $\mathsf{LagrangianMechanics}^{\uparrow}$ *carrier may read only the action ledger rows* Q, V, A, R, E*, the Euler–Lagrange residual row* G*, the symplectic-flow row* S*, the Noether-current row* J*, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The current row* J *is a finite consumer-facing ledger entry, not a theorem asserting a complete Noether correspondence.*

The boundary exports no symmetry classification, no conservation-law completeness theorem, no Hamiltonian equivalence, and no standard bridge to a smooth phase-space model.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket noether consumer boundary`

Proof. Read the accepted packet through Definition 337.1. Its Noether-facing coordinates are the action ledger rows, the residual equation row, the symplectic-flow row, the current row, and the four provenance rows.

Apply Definition 337.2 to identify the only action data available to the consumer. Apply Definition 337.3 to identify the only Euler–Lagrange residual read.

The remaining rows S and J are finite ledgers for phase-flow and current reads. Since the carrier has no coordinate for a symmetry classifier, Hamiltonian equivalence, conservation-law completeness theorem, or standard bridge, the Noether-facing boundary is exactly the stated row list. □

Theorem 337.5 (LagrangianMechanicsUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LagrangianMechanics}^{\uparrow}}$ *seed surface consists of the carrier of Definition 337.1, the action ledger of Definition 337.2, the Euler–Lagrange interface of Definition 337.3, and the Noether consumer boundary of Definition 337.4. Its dependency surface is the finite* Pkg *row naming the action, variation, Euler–Lagrange, symplectic, and Noether certificate boundaries, together with the displayed* hsame*,* Cont*, and* NameCert *provenance rows.*

The seed surface does not close minimizer existence, global dynamics, Hamiltonian equivalence, complete Noether correspondence, analytic regularity, or a standard bridge.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket namecert obligation surface`

Proof. Assemble the seed surface from the carrier. The definition supplies the configuration, velocity, action, variation, endpoint, Euler–Lagrange, symplectic-flow, Noether-current, transport, continuation, package, and name-certificate rows.

The action-ledger theorem identifies the only action-facing projection. The Euler–Lagrange theorem identifies the only residual-equation projection. The Noether boundary theorem identifies the only phase-flow and conserved-current consumer rows.

These projections cover every coordinate of the finite packet. The exclusions follow because none of the rows stores a minimizer, global solution theorem, Hamiltonian equivalence, complete Noether correspondence, analytic regularity principle, or standard bridge. □

Theorem 337.6 (LagrangianMechanics finite obligation carrier)**.** *The finite* $\mathsf{LagrangianMechanics}^{\uparrow}$ *obligation carrier consists of* BHist *rows for configuration history, velocity history, action-density data, a variation token, and visible* Pkg *and* NameCert *provenance. Each row is read as part of the carrier of Definition 337.1, and no row introduces a smooth manifold, a completed path space, or an analytic extremal.*

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket finite obligation carrier`

Proof. Project the carrier tuple of Definition 337.1. The configuration and velocity histories are the rows Q and V, while the action-density and variation data are read from the action and variation coordinates A and R.

The provenance rows are the package and local naming coordinates P and N. Repacking these projected rows gives the finite obligation carrier. Since the carrier list contains no host manifold, path-space completion, or analytic extremal coordinate, those objects are outside the obligation carrier. □

Theorem 337.7 (LagrangianMechanics action ledger policy)**.** *The* $\mathsf{LagrangianMechanics}^{\uparrow}$ *action ledger exposes only finite action-comparison rows, boundary variation rows, and* Cont *transport through the displayed action, variation, endpoint, and continuation coordinates of Definition 337.1. It does not solve an Euler–Lagrange equation, select an analytic minimizer, or supply a smooth manifold model.*

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket action ledger policy`

Proof. Start from the action ledger obligation already identified in Definition 337.2. Its readable rows are the finite configuration, velocity, action, variation, endpoint, transport, continuation, and provenance coordinates.

The comparison data are therefore the finite action rows and the boundary variation rows transported along the displayed Cont routes. Repacking those rows yields the ledger policy, and the excluded analytic objects are absent from the carrier coordinates. □

Theorem 337.8 (LagrangianMechanics Euler boundary scope)**.** *The Euler boundary for* $\mathsf{LagrangianMechanics}^{\uparrow}$ *is a scoped obligation over the displayed finite variation rows. Its dependencies are exactly the* BHist *carrier rows, the* hsame *transports, the* Cont *routes, and the visible* Pkg *and* NameCert *provenance needed to read the Euler–Lagrange residual row of Definition 337.3.*

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket euler boundary scope`

Proof. Use the Euler–Lagrange interface theorem to identify the residual row G and the finite action, variation, endpoint, transport, continuation, and provenance rows that feed it. These are all rows of the same BHist carrier.

Transporting the displayed variation rows uses only the hsame witnesses in H and the continuation routes in C. Repacking them with P and N gives the scoped Euler boundary, with no further analytic equation surface added. □

Theorem 337.9 (LagrangianMechanics public NameCert boundary)**.** *For an accepted* $\mathsf{LagrangianMechanics}^{\uparrow}$ *carrier, the finite configuration, velocity, action, variation, endpoint, Euler–Lagrange residual, symplectic-flow read, Noether-current read,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows assemble the public* $\mathsf{NameCert}_{\mathsf{LagrangianMechanics}^{\uparrow}}$ *boundary. A downstream mechanics or Noether-facing consumer may read the public boundary only through this finite packet.*

The boundary exports no minimizer existence theorem, no global dynamics model, no Hamiltonian equivalence, no complete Noether correspondence, no analytic regularity theorem, and no standard bridge.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket public namecert boundary`

Proof. Start with the action ledger of Definition 337.2. It supplies the finite configuration, velocity, action, variation, endpoint, transport, package, and name-certificate rows.

The Euler–Lagrange interface of Definition 337.3 adds the residual row that is read from the same action, variation, endpoint, hsame, and Cont surface. The Noether consumer boundary of Definition 337.4 adds only the symplectic-flow and Noether-current rows.

Use Definition 337.5 and Definition 337.6 to repack these rows as one local certificate carrier. The action ledger policy of Definition 337.7 and the Euler boundary scope of Definition 337.8 show that the public read exposes exactly the same finite packet and no additional analytic object. □

Theorem 337.10 (LagrangianMechanics finite standard bridge surface)**.** *The public finite action, Euler–Lagrange residual, symplectic-flow read, Noether-current read,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows determine the paper-side standard bridge surface for the finite* LagrangianMechanics$^{\uparrow}$ *schema. Standard action, residual, flow, and current reads may be interpreted only through these displayed rows.*

The bridge surface exports no minimizer, global dynamics theorem, Hamiltonian equivalence, analytic regularity theorem, or complete Noether correspondence.

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket finite standard bridge surface`

Proof. Begin with the public boundary of Definition 337.9. It already exposes the finite action, residual, flow, current, carrier, transport, package, and name-certificate rows as the only public mechanics packet.

The standard action read is the action ledger of Definition 337.2. The standard residual read is the Euler–Lagrange interface of Definition 337.3. The standard flow and current reads are the finite rows of Definition 337.4.

Repack these reads with the obligation surface of Definition 337.5, the finite carrier of Definition 337.6, and the boundary scope of Definition 337.8. The resulting bridge surface uses only the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Since none of the cited rows contains a minimizer, global dynamics theorem, Hamiltonian equivalence, analytic regularity theorem, or complete Noether correspondence, the finite standard bridge surface has exactly the stated non-claim boundary. □

Theorem 337.11 (LagrangianMechanics symplectic and Noether sibling route)**.** *For an accepted* LagrangianMechanics$^{\uparrow}$ *carrier*

$$L = (Q, V, A, R, E, G, S, J, H, C, P, N),$$

any consumer of the symplectic-flow row S or the Noether-current row J factors through the displayed Symplectic$^{\uparrow}$ *and* NoetherSymmetry$^{\uparrow}$ *sibling surfaces named in P, together with the* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows of the same packet. The route exports no hidden global solution theorem, Hamiltonian equivalence, complete conservation law, or ambient analytic phase-space model.*

Lean4 ✓ `BEDC.Derived.LagrangianMechanicsUp.LagrangianMechanicsPacket symplectic noether sibling route`

Proof. Start with the carrier definition. The rows S and J are explicit coordinates of the finite packet, and the provenance row P names the symplectic and Noether dependency certificates consumed by those coordinates.

Use the Noether consumer boundary and public NameCert boundary. They identify the only consumer-facing mechanics rows as the finite action ledger, residual row, symplectic-flow read, Noether-current read, and the structural provenance rows H, C, P, N.

Finally apply the finite standard bridge surface. It allows standard action, residual, flow, and current reads only through the displayed packet. Since no coordinate stores a global solution, Hamiltonian equivalence, complete conservation law, or ambient phase-space model, every use of S or J remains inside the stated sibling route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LagrangianMechanics}^{\uparrow}}$.

Constructive story: Lagrangian mechanics is represented by finite rows for configuration, velocity, action functional, variation, endpoint policy, Euler–Lagrange residual, symplectic-flow read, Noether-current read, and provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LagrangianMechanics}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 337.1, Definition 337.2, Definition 337.3, Definition 337.4, Definition 337.5, Definition 337.8, Definition 337.9, and Definition 337.10 bind the public-facing finite action, Euler–Lagrange, symplectic-flow, Noether-current, public NameCert packet boundary, and paper-side finite standard bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LagrangianMechanics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close minimizer existence, global dynamics, Hamiltonian equivalence, complete Noether correspondence, analytic regularity, or bridge checking.

Upgrade path: No further paper-axis theory grade remains for this finite bridge surface; Lean verification remains on the formal axis.

Chapter 338

A Finite Observation Filter Surface for $\mathsf{CauchyFilter}^{\uparrow}$

The $\mathsf{CauchyFilter}^{\uparrow}$ surface names the finite observation filter used by the Real and Completion phase when a consumer needs explicit Cauchy data without host filters or choice-selected subsequences. It sits between $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as a visible packet of late-window evidence.

Its dependency lattice is the Real-and-Completion row already exposed by the sibling certificates: chapter 126 supplies the finite observation windows, chapter 292 supplies threshold rows, chapter 291 supplies endpoint rows, chapter 301 supplies the regular-rational compatibility handoff, and chapter 38 supplies the sealed consumer endpoint. Thus $\mathsf{CauchyFilter}^{\uparrow}$ is not a parallel completion carrier; it is the finite late-window filter surface routed through those certificates.

The directed finite-window ledger is consumed by the CauchyCriterion surface of chapter 392 before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ reader can treat the packet as Cauchy evidence. $\mathsf{CauchyFilter}^{\uparrow}$ supplies the visible late-window evidence, while $\mathsf{CauchyCriterion}^{\uparrow}$ supplies the criterion boundary that blocks host filters, ultrafilters, and choice-selected subsequences from entering the consumer route.

Definition 338.1 (CauchyFilter carrier)**.** *A* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (s, d, m, e, r, t, c, P, N)$$

with the following displayed rows:

1. s *is the* $\mathsf{StreamName}^{\uparrow}$ *observation family whose entries are visible in the packet;*
2. d *is a directed finite-window index ledger ordering the finite tails through which observations are read;*
3. m *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row assigning a displayed precision request to a finite tail bound;*
4. e *is the dyadic endpoint row package supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
5. r *is the compatibility row against* $\mathsf{RegSeqRat}^{\uparrow}$*, recording the same late-window comparisons as regular rational sequence evidence;*
6. t *contains the* hsame *and* Cont *transport rows used to compare and extend finite windows;*
7. c *is the consumer-facing ledger naming* $\mathsf{Real}^{\uparrow}$ *as the sealed reader of the finite filter packet;*

8. *P is the* Pkg *provenance for the dependency anchors* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilter}^{\uparrow}}$ *row.*

The carrier accepts exactly those packets whose listed rows occur in one finite history and whose transport routes are displayed in that history. It does not contain a host filter, ultrafilter, chosen subsequence, metric completion point, or quotient stream.

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterTasteGate single carrier alignment`

Theorem 338.2 (CauchyFilter finite-window coverage)**.** *Every accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *carrier covers its public observations by a directed finite window. More precisely, each visible* $\mathsf{StreamName}^{\uparrow}$ *read in the packet factors through the index ledger d, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row m, the dyadic endpoint rows e, the* hsame *and* Cont *transport rows t, and the* Pkg *provenance P.*

Proof. Start from the carrier decomposition in Definition 338.1. The observation family, directed window ledger, threshold row, endpoint rows, transport rows, and provenance row are all coordinates of the same finite BHist packet.

For a displayed observation read, choose the finite tail window named by the ledger. The threshold row supplies the bound for the requested precision, and the dyadic endpoint rows give the finite values inspected inside that tail.

The remaining comparison is routed through the displayed hsame and Cont transport rows. Since the packet contains no constructor for an external filter object, the public read is exhausted by the finite window and its package provenance. □

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket finite window coverage`
Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterCarrier finite window coverage`

Theorem 338.3 (CauchyFilter RegSeqRat handoff)**.** *An accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet hands off to* $\mathsf{RegSeqRat}^{\uparrow}$ *exactly through its compatibility row r. The handoff preserves the* $\mathsf{StreamName}^{\uparrow}$ *observation family, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the directed finite-window ledger, and the displayed* hsame, Cont, *and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterCarrier regseqrat handoff`
Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket regseqrat handoff`

Proof. Project the packet to the compatibility row r and the rows named in Definition 338.1. The row r is already the displayed regular-sequence comparison against the same late-window observations.

Transport along the listed hsame rows and restrict along the displayed Cont windows. These operations do not change the stream observations, threshold row, dyadic endpoints, or directed window ledger; they only re-read the same finite rows through the $\mathsf{RegSeqRat}^{\uparrow}$ certificate.

Thus the handoff has no additional carrier beyond the visible compatibility row and the package provenance. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives finite late-window evidence, not a quotient stream or a completed real. □

Theorem 338.4 (CauchyFilter Real seal consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet factors through the displayed finite filter surface: the* $\mathsf{StreamName}^{\uparrow}$ *observations, directed finite-window ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row,* hsame *and* Cont *transport, and* Pkg *provenance. The boundary exports no metric-completion point, host real equality, or choice-selected tail.*

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterFiniteWindowSurface real seal consumer boundary`

Proof. Read the consumer ledger coordinate c of the carrier. It names $\mathsf{Real}^{\uparrow}$ as a sealed reader of the same finite filter rows rather than as a source of extra completion data.

Apply the finite-window coverage theorem to the underlying observations, then apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff theorem to the compatibility row. Together they identify every available consumer read with a displayed tail window, endpoint package, threshold row, transport route, and provenance row.

There is no remaining coordinate from which a metric-completion point, host real equality, or selected infinite subsequence could be read. The real-facing boundary is therefore the finite filter surface itself. □

Theorem 338.5 (CauchyFilter public obligation catalogue)**.** *The public* $\mathsf{NameCert}_{\mathsf{CauchyFilter}^{\uparrow}}$ *obligation catalogue has exactly five rows:*

1. *the carrier row of Definition 338.1;*
2. *the classifier row identifying two packets when they admit a common finite filter refinement with the same threshold, endpoint, and compatibility rows;*
3. *stability under* hsame *transport of the observation, threshold, endpoint, compatibility, and provenance rows;*
4. *exactness of visible filter-window reads through the directed finite ledger and displayed* Cont *routes;*
5. *a ledger-exclusion row forbidding host filters, ultrafilters, choice-selected subsequences, metric-completion points, and quotient streams.*

These rows are the entire public surface used by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers.*

Proof. The carrier row is the finite packet specified in Definition 338.1. Its coordinates already list the observation family, finite-window ledger, threshold, endpoints, regular-sequence compatibility, transport, provenance, and local naming row.

The classifier compares two packets by refining their directed ledgers to a common finite window and then comparing the threshold, endpoint, compatibility, transport, and provenance rows through hsame. This gives the stability row at the same time, because transported packets retain the rows used by the classifier.

Exactness follows by re-reading each public observation through the same finite ledger and Cont route. The ledger-exclusion row is forced by the carrier shape: none of the listed external filter, completion, choice, or quotient objects is a coordinate of the packet. □

Theorem 338.6 (CauchyFilter common-refinement classifier)**.** *Two accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packets classify together exactly when their directed finite-window ledgers admit a displayed common finite refinement carrying the same* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row,* hsame *and* Cont *transport rows, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket common refinement classifier`

Proof. Project both accepted packets through the carrier decomposition of Definition 338.1. The visible coordinates are finite BHist rows, so the two directed ledgers can be compared by passing to the displayed common window named by the classifier row.

On that common window, compare the threshold, endpoint, regular-sequence compatibility, transport, and provenance coordinates through the recorded hsame rows. The displayed Cont routes re-read the same observations without adding a host filter or a selected subsequence.

Repack the matched rows as one $\mathsf{NameCert}_{\mathsf{CauchyFilter}^{\uparrow}}$ classifier judgement. Conversely, reading such a classifier judgement exposes precisely the common finite refinement and the listed coordinate equalities, so it returns the same condition. □

Theorem 338.7 (CauchyFilter refinement transport)**.** *Refining the directed finite window of an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet preserves acceptance and preserves every visible observation through the displayed* hsame *and* Cont *transport rows.*

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket refinement transport`

Proof. Begin with the finite-window coverage theorem. The public observations already factor through the directed ledger, threshold row, endpoint rows, transport rows, and provenance row of the accepted packet.

Apply Definition 338.6 to the original window and the refined window. The common-refinement classifier identifies the same $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{CauchyModulus}^{\uparrow}$ thresholds, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, $\mathsf{RegSeqRat}^{\uparrow}$ compatibility row, and package provenance on the refined ledger.

The refined packet is therefore accepted by the same finite carrier rows. No host filter, ultrafilter, or choice-selected subsequence is introduced, because the transport uses only the displayed hsame comparisons and Cont routes. □

Lean4 ✓ `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket window refinement transport`

Theorem 338.8 (CauchyFilter Real-seal refinement exhaustion)**.** *After any common finite refinement, every* $\mathsf{Real}^{\uparrow}$ *consumer read from an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet factors through the same finite filter rows and exports no metric-completion point, host real equality, quotient stream, or choice-selected tail.*

Proof. Use Definition 338.6 to replace the consumer-facing packet by its common refined finite window. The classifier keeps the visible observations, thresholds, endpoints, compatibility, transport, and provenance rows fixed on that window.

Apply Definition 338.3 to the compatibility row. The handoff shows that the regular-rational read is the same finite late-window evidence, not an external quotient stream.

Finally apply Definition 338.4. The $\mathsf{Real}^{\uparrow}$ ledger reads only the finite filter surface and therefore has no coordinate from which a metric-completion point, host real equality, or choice-selected tail could be exported. □

Theorem 338.9 (CauchyFilter scoped dependency packet)**.** *The carrier of Definition 338.1, finite-window coverage of Definition 338.2,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 338.3,* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 338.4, common-refinement classifier of Definition 338.6, refinement transport of Definition 338.7, and Real-seal refinement exhaustion of Definition 338.8 form one scoped BEDC packet over* BHist*.*

Observation reads in this packet pass through $\mathsf{StreamName}^{\uparrow}$*, precision rows through* $\mathsf{CauchyModulus}^{\uparrow}$*, endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$*, compatibility through* $\mathsf{RegSeqRat}^{\uparrow}$*, and every public transport through the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project an accepted packet to the carrier rows of Definition 338.1. Apply Definition 338.2 to identify the finite BHist window and its $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{CauchyModulus}^{\uparrow}$ precision rows, and $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows.

The compatibility and consumer boundaries are the next projections. Use Definition 338.3 for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and Definition 338.4 for the $\mathsf{Real}^{\uparrow}$ seal. Both cite only displayed finite rows and package provenance.

It remains to bind refinement. Apply Definition 338.6 and Definition 338.7 to move between finite windows through the recorded hsame and Cont data, then use Definition 338.8 to close the consumer readback under common refinement.

Repacking these projections gives the single scoped packet. Since every cited projection is a coordinate of the displayed BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, or NameCert surface, there is no channel for a host filter, ultrafilter, quotient stream, or choice-selected subsequence. □

Theorem 338.10 (CauchyFilter finite-filter standard bridge)**.** *An accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet determines the standard finite Cauchy-filter schema exactly through the displayed* $\mathsf{StreamName}^{\uparrow}$ *observation rows, directed finite-window ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row, displayed* hsame *routes,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Conversely, any standard finite-filter package whose rows are exactly those displayed BEDC rows repacks as the same public $\mathsf{CauchyFilter}^{\uparrow}$ *packet. The bridge has no coordinate for host filters, ultrafilters, quotient streams, metric-completion points, or choice-selected subsequences.*

Proof. Project the accepted packet through the carrier of Definition 338.1. Apply Definition 338.2 to expose the finite observation windows, directed ledger, $\mathsf{StreamName}^{\uparrow}$ rows, $\mathsf{CauchyModulus}^{\uparrow}$ threshold row, and $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints as the visible finite-filter data.

Next use Definition 338.3 to read the $\mathsf{RegSeqRat}^{\uparrow}$ compatibility row and Definition 338.4 to keep the $\mathsf{Real}^{\uparrow}$ consumer boundary on the same finite surface. These projections add no carrier beyond the displayed finite rows.

Apply Definition 338.9 to gather the displayed hsame routes, Cont routes, Pkg provenance, and local NameCert row into the scoped dependency packet. This is exactly the standard finite Cauchy-filter schema because the cited packet has no additional coordinate.

For the converse direction, start with a standard finite-filter package whose coordinates are exactly the displayed rows. Repack those rows in the carrier, handoff, refinement, seal, transport, provenance, and local naming positions just projected. The result is the same public $\mathsf{CauchyFilter}^{\uparrow}$ packet because no further coordinate remains to choose.

The excluded objects are not hidden cases of the construction. Host filters, ultrafilters, quotient streams, metric-completion points, and choice-selected subsequences are absent from every cited projection, so the bridge cannot read or export them. □

Theorem 338.11 (CauchyFilter RegSeqRat Real seal exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and every* $\mathsf{Real}^{\uparrow}$ *seal consumer read from an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet factors through one finite surface: stream observations, the directed finite-window ledger, the modulus threshold, dyadic endpoints, the compatibility row,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

No consumer read from this export contains a host filter, ultrafilter, quotient stream, metric-completion point, or choice-selected subsequence.

Proof. Begin with Definition 338.2. It gives the stream observations, directed finite-window ledger, modulus threshold, and dyadic endpoints as displayed finite rows of the accepted packet.

Next apply Definition 338.3. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives exactly the compatibility row over those finite observations and endpoints.

Apply Definition 338.4. The $\mathsf{Real}^{\uparrow}$ seal consumes the same finite filter surface, with no additional host filter or completion coordinate.

It remains to bind refinement. Use Definition 338.6 and Definition 338.7 for the common finite-window classifier and its hsame and Cont transport, then apply Definition 338.9 to repack the package and local naming rows.

The cited projections exhaust the consumer coordinates. Therefore each $\mathsf{RegSeqRat}^{\uparrow}$ handoff and each $\mathsf{Real}^{\uparrow}$ seal read factors through the single finite surface stated above, and the excluded host objects have no displayed row from which they could be read. □

Theorem 338.12 (CauchyFilter finite-filter mature coverage)**.** *Every public or standard finite-filter consumer read of an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet is exhausted by the displayed* $\mathsf{StreamName}^{\uparrow}$ *observations, directed finite-window ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility,* $\mathsf{Real}^{\uparrow}$ *seal, common-refinement classifier,* hsame *and* Cont *transport,* Pkg *provenance, and local* NameCert *row.*

The covered surface preserves the exclusions of host filters, ultrafilters, quotient streams, metric-completion points, and choice-selected subsequences.

Proof. First apply Definition 338.2. It supplies the displayed stream observations, directed finite-window ledger, threshold row, and dyadic endpoint rows as the finite carrier available to consumers.

Next use Definition 338.3 and Definition 338.4. These identify the $\mathsf{RegSeqRat}^{\uparrow}$ compatibility read and the $\mathsf{Real}^{\uparrow}$ seal as reads of the same finite carrier, with no completion or host-filter coordinate added.

Refinement is controlled by Definition 338.6 and Definition 338.7. The first theorem classifies common finite-window refinements, and the second transports that classifier through the displayed hsame and Cont routes.

Finally apply Definition 338.10 and Definition 338.11. The standard finite-filter packet and the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer boundary read exactly the listed rows, while host filters, ultrafilters, quotient streams, metric-completion points, and choice-selected subsequences remain outside the packet. □

Theorem 338.13 (CauchyFilter mature Real consumer example)**.** *A* $\mathsf{Real}^{\uparrow}$ *seal consumer supplied with an accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *packet and a* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row obtains a finite late-window package consisting only of* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints, the compatibility row,* hsame, Cont, Pkg, *and local* NameCert *evidence.*

The package contains no host filter, ultrafilter, quotient stream, metric-completion point, or choice-selected subsequence.

Proof. First apply Definition 338.2. It supplies the visible stream observations, finite-window ledger, modulus threshold, and dyadic endpoint rows of the accepted packet.

Next use Definition 338.3. The $\mathsf{RegSeqRat}^{\uparrow}$ compatibility row is a read of the same finite late-window surface, so it adds no host filter or quotient-stream coordinate.

Apply Definition 338.4. This identifies the $\mathsf{Real}^{\uparrow}$ seal consumer as a reader of the finite filter package rather than of a selected completion object.

Finally use Definition 338.11. It exhausts the combined $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback surface by the listed rows, leaving the excluded host objects outside the package. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyFilter}^{\uparrow}$ packet is generated from visible stream observations, finite late-window ledgers, $\mathsf{CauchyModulus}^{\uparrow}$ thresholds, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, $\mathsf{RegSeqRat}^{\uparrow}$ compatibility rows, $\mathsf{Real}^{\uparrow}$ seal-consumer rows,

common-refinement classifiers, refinement-transport evidence, componentwise hsame comparisons, displayed Cont routes, package provenance, and local naming evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyFilter}\uparrow}$, maturePackageClosed)

Scope closed: The chapter closes the CauchyFilter$^{\uparrow}$ carrier, finite-window coverage, RegSeqRat$^{\uparrow}$ handoff, Real$^{\uparrow}$ seal boundary, common-refinement classifier, refinement transport, and mature finite-filter coverage as finite NameCert data.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyFilter}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CauchyFilterUp.CauchyFilterPacket\_finite\_window\_coverage`

Bridge status: none

Origin: human

Not claimed: The chapter does not close host filters, ultrafilters, quotient streams, metric-completion points, or choice-selected tails.

Upgrade path: A complete StandardBridge instance for the accepted finite-filter packet would raise the bridge axis.

Chapter 339

A Concrete Naming Certificate for $\mathsf{UniformSpace}^{\uparrow}$

The $\mathsf{UniformSpace}^{\uparrow}$ surface names the finite entourage interface between $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Filter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$. Bourbaki–Kelley uniform-space practice treats entourages as the common language behind topology, filters, Cauchy data, and completion; the BEDC certificate keeps only finite BHist rows, visible classifier transport, and explicit refinement ledgers.

Definition 339.1 (UniformSpaceUp BHist entourage carrier)**.** *A* $\mathsf{UniformSpace}^{\uparrow}$ *carrier is a finite packet*

$$U = (X, E, D, R, S, C, P, N)$$

where X is the carried point-history source, E is a finite list of entourage rows over $X \times X$, D records the diagonal row for each listed entourage, R records finite refinement witnesses, S records symmetry readback, C records composition readback, P is the Pkg *provenance ledger, and N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$ *row. The point classifier is inherited from the X row and is stable under* hsame *transport on both coordinates of every entourage row.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket semantic name certificate`

Theorem 339.2 (UniformSpaceUp finite filterbase diagonal obligation)**.** *Every accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier exposes a finite filterbase of entourage rows. For each displayed entourage $V \in E$, the diagonal row records that every accepted point history x has (x, x) in V, and the finite filterbase ledger records a displayed refinement row for every pair of entourages named by downstream* $\mathsf{Filter}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket filterbase diagonal obligation`

Proof. Unpack the carrier of Definition 339.1. The diagonal component D is indexed by the same finite entourage list E, so each consumer-visible entourage carries its own diagonal witness.

For a pair of displayed entourages, the refinement component R supplies a third displayed entourage and the two inclusion reads needed by a filterbase consumer. Repacking those rows gives the finite filterbase obligation without forming an external filter object. □

Theorem 339.3 (UniformSpaceUp diagonal refinement obligation)**.** *If a* $\mathsf{UniformSpace}^{\uparrow}$ *consumer requests a diagonal neighbourhood from a* $\mathsf{Met}^{\uparrow}$ *ball row or a* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row, the carrier factors the request through one of its finite entourage rows and a displayed refinement ledger. The request sees only the point source, the chosen entourage, its diagonal row,* hsame *transport on endpoints, and the* Pkg *provenance row.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpaceEntouragePacket diagonal refinement obligation`

Proof. Read the request as a finite row over the same point-history source X. The $\mathsf{Met}^{\uparrow}$ and $\mathsf{Topology}^{\uparrow}$ anchors supply only neighbourhood data, not a host uniformity.

The carrier supplies a named entourage row whose diagonal witness covers the requested point and whose refinement ledger records the inclusion into the requested neighbourhood. Endpoint changes are transported by the inherited hsame classifier, and the visible provenance remains the single package row P. □

Theorem 339.4 (UniformSpaceUp symmetry and composition obligation)**.** *For every displayed entourage V in an accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier, the symmetry row S gives a displayed entourage V^{op} with the endpoints exchanged. For every displayed entourage V, the composition row C gives a displayed refinement W such that two successive W-steps read as one V-step. Both rows are finite* BHist *ledgers stable under the inherited point classifier.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket symmetry composition obligation`

Proof. The symmetry component is a row-by-row endpoint exchange in the finite entourage list. Applying hsame on the first and second coordinates carries the exchanged row to classified representatives.

The composition component names a finite witness W and a ledger read from $W \circ W$ into V. Because both W-steps and the target V-step are rows of the same carrier, the composition readback is stable under the same endpoint transport and package provenance. □

Theorem 339.5 (UniformSpaceUp concrete entourage instance)**.** *Every accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier $U = (X, E, D, R, S, C, P, N)$ exports a concrete entourage instance whose public reads are finite* BHist *predicates only. The instance consists of the displayed entourage rows E, the reflexivity witnesses in D, the symmetry witnesses in S, the composition witnesses in C, the componentwise* hsame *endpoint transport in the inherited point classifier, the displayed* Cont *replay route for consumer requests, and the* Pkg *provenance row P.*

A $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Filter}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*, or* $\mathsf{Completion}^{\uparrow}$ *consumer may project those rows, but it may not replace them by parameter-equivalence transport, host uniformity, quotient equality of Cauchy streams, open-cover compactness, or a selected completion point.*

Proof. Project the carrier of Definition 339.1. The finite entourage list E is the only predicate family available to the consumer, and the diagonal component D gives the displayed reflexivity row for each entourage.

Read the symmetry and composition rows next. The symmetry component S exchanges endpoints inside the displayed finite ledger, while the composition component C records the finite refinement by which two successive displayed steps are read as one requested entourage step.

Endpoint changes are handled by the inherited point classifier through componentwise hsame transport, and consumer observations replay through the displayed Cont routes. The provenance row P records that all rows belong to the same $\mathsf{UniformSpace}^{\uparrow}$ packet.

Since these projections exhaust the carrier coordinates used by an entourage consumer, no parameter-equivalence transport, host uniformity, quotient stream equality, open-cover compactness datum, or selected completion point is exported by the concrete instance. □

Theorem 339.6 (UniformSpaceUp Cauchy and completion handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of a* $\mathsf{UniformSpace}^{\uparrow}$ *packet factors through the finite entourage filterbase, diagonal rows, refinement rows, symmetry rows, composition rows,* hsame *endpoint transport, and*

Pkg *provenance of Definition 339.1. The handoff introduces no choice-selected completion point, quotient stream equality, open-cover compactness, or host-space uniformity.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpaceClassifierPacket cauchy completion handoff`

Proof. Project the accepted packet to its finite filterbase rows. A Cauchy consumer reads finite entourage obligations exactly as in the $\mathsf{CauchyFilter}^{\uparrow}$ surface: each demanded precision row is matched by a displayed entourage and a refinement witness.

A completion consumer may retain the same finite Cauchy ledger and provenance rows, but there is no coordinate in the carrier for a selected limit point or a quotient class of streams. Therefore the handoff is only the visible entourage data listed in the theorem. □

Theorem 339.7 (UniformSpaceUp classifier transport obligation)**.** *Let* $U = (X, E, D, R, S, C, P, N)$ *be an accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier as in Definition 339.1. Endpoint* hsame *transport of the point source* X *and of every displayed entourage row repacks another accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier with the same finite entourage list, diagonal ledger, refinement ledger, symmetry ledger, composition ledger,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket classifier transport obligation`

Proof. Unpack the carrier definition. The point classifier is the one inherited from the X row, and every entourage row is explicitly declared stable under hsame on both coordinates.

Transport the endpoint histories first, then apply the same transport to the diagonal, refinement, symmetry, and composition rows. The carrier has no coordinate outside these finite ledgers, so the transported rows repack with the unchanged package provenance and naming certificate. □

Theorem 339.8 (UniformSpaceUp entourage ledger exactness)**.** *Every consumer-visible entourage read from an accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier is exhausted by the finite list* E*, the diagonal row* D*, the refinement row* R*, the symmetry row* S*, the composition row* C*, and the* Pkg *provenance row. No consumer can read an additional host uniformity, open-cover compactness principle, quotient stream equality, or choice-selected completion datum from the carrier.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket entourage ledger exactness`

Proof. Project the accepted packet to the coordinates of Definition 339.1. The only entourage-bearing coordinate is the finite list E, and the only certified reads from this list are the rows named by D, R, S, and C.

Apply the diagonal obligation of Definition 339.2, the neighbourhood refinement obligation of Definition 339.3, and the symmetry and composition obligation of Definition 339.4. These rows account for every displayed consumer read while the package row P accounts for provenance. □

Theorem 339.9 (UniformSpaceUp common refinement Cauchy handoff)**.** *Any* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *request through two compatible entourage demands of an accepted* $\mathsf{UniformSpace}^{\uparrow}$ *carrier factors through a displayed common refinement row. The exported handoff contains only the point source, the two demanded entourage rows, the common refinement witness, endpoint* hsame *transport, the relevant* Cont *observation windows, and* Pkg *provenance; it exports no choice-selected limit, quotient stream, open-cover compactness, or host uniformity.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket common refinement cauchy handoff`

Proof. Start with the two compatible demands as finite reads from the entourage list E. The filterbase diagonal obligation provides the common refinement row from R and the inclusions back into the two demanded rows.

Transport endpoint representatives by Definition 339.7. The refinement, diagonal, symmetry, and composition rows remain rows of the same finite carrier after this transport.

Finally apply the handoff theorem Definition 339.6. A Cauchy or completion consumer receives the finite refinement ledger and provenance, but the carrier contains no coordinate for a selected limit, quotient stream, open-cover compactness datum, or ambient host uniformity. □

Theorem 339.10 (UniformSpaceUp common refinement functoriality)**.** *Let U_0, U_1, and U_2 be accepted* $\mathsf{UniformSpace}^{\uparrow}$ *packets. Suppose the displayed finite entourage ledger contains a compatible common-refinement row from U_0 to U_1 and a compatible common-refinement row from U_1 to U_2. Then every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *request routed through U_2 also routes through U_0 by the composed finite entourage ledger.*

The composed handoff contains only the point-history source, the two displayed common-refinement rows, diagonal witnesses, endpoint hsame *transport, the relevant* Cont *observation windows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$ *rows. It introduces no quotient equality of Cauchy streams, no selected completion point, no choice-based completion data, and no ambient host uniformity.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket common refinement functoriality`

Proof. First compose the two displayed common-refinement rows. The filterbase diagonal, neighbourhood refinement, symmetry, and composition obligations keep the composite as a finite entourage ledger row of the same BEDC form.

Next apply Definition 339.7 to each handoff. Endpoint representatives move along the displayed hsame rows, and the Cont observation windows remain rows of the accepted packets.

Now invoke Definition 339.9 for the route from U_2 to U_1, and invoke it again for the route from U_1 to U_0. The first application supplies the finite ledger consumed by the second, so the $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ request is read at U_0 through the composed common refinement.

Finally repack the resulting ledger with the same BHist, hsame, Cont, Pkg, and NameCert provenance. Since the construction only concatenates displayed finite rows, it never forms a quotient of Cauchy streams and never selects a completion point. □

Theorem 339.11 (UniformSpaceUp scoped entourage package)**.** *The carrier of Definition 339.1, together with Definition 339.2, Definition 339.3, Definition 339.4, Definition 339.6, Definition 339.7, Definition 339.8, and Definition 339.9, and Definition 339.10, forms one scoped BEDC packet over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Every $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumer of this packet receives only finite entourage rows, displayed endpoint transports, continuation routes, package provenance, and the local naming row.*

Proof. Unpack the carrier of Definition 339.1. Its displayed coordinates are the point-history source, the finite entourage list, the diagonal row, the refinement row, the symmetry row, the composition row, transport and continuation data, package provenance, and local naming evidence.

The diagonal and filterbase projections are exactly Definition 339.2 and Definition 339.3. The symmetry and composition projections are supplied by Definition 339.4.

Classifier transport is supplied by Definition 339.7, and carrier exhaustion is supplied by Definition 339.8. These projections keep all consumer reads on the same BHist, hsame, Cont, Pkg, and NameCert surface.

Finally apply Definition 339.6 and Definition 339.9. The CauchyFilter$^{\uparrow}$ and Completion$^{\uparrow}$ consumers receive the finite entourage rows and their common refinement data, with no selected completion point, quotient stream, open-cover compactness datum, or ambient host uniformity. □

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket scoped entourage package`

Theorem 339.12 (UniformSpaceUp entourage public export)**.** *The finite filterbase diagonal row, diagonal refinement row, symmetry and composition rows, classifier transport row, entourage ledger, and common-refinement Cauchy handoff rows of the scoped* UniformSpace$^{\uparrow}$ *packet assemble into the public downstream packet for* CauchyFilter$^{\uparrow}$ *and* Completion$^{\uparrow}$ *consumers.*

The public packet exports only finite entourage data over BHist*, displayed* hsame *endpoint transport,* Cont *observation routes,* Pkg *provenance, and local* NameCert *evidence.*

Proof. Start with the scoped package of Definition 339.11. It binds the carrier of Definition 339.1 to the finite entourage rows, transport rows, continuation routes, package provenance, and local naming evidence.

The finite filterbase and diagonal parts are supplied by Definition 339.2 and Definition 339.3. Symmetry and composition are supplied by Definition 339.4.

Classifier transport and ledger exhaustion are supplied by Definition 339.7 and Definition 339.8. These rows ensure that public consumers read the same finite entourage surface after endpoint transport.

Finally use Definition 339.6 and Definition 339.9. They identify the downstream CauchyFilter$^{\uparrow}$ and Completion$^{\uparrow}$ requests with the common finite-refinement packet, leaving selected limits, quotient streams, open-cover compactness data, and host uniformities outside the export. □

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket entourage public export`

Theorem 339.13 (UniformSpace finite-entourage standard bridge)**.** *The finite filterbase diagonal row, diagonal refinement row, symmetry and composition rows, classifier transport row, entourage ledger, common-refinement Cauchy handoff rows, displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows assemble the standard finite-entourage bridge for* UniformSpace$^{\uparrow}$*.*

The bridge exposes exactly the public finite-entourage packet of Definition 339.12. It does not add choice-based completion, quotient equality of Cauchy streams, open-cover compactness, host uniformity, or selected limits.

Proof. Begin with the scoped finite-entourage package of Definition 339.11. Its carrier is the displayed BHist of Definition 339.1, equipped with the finite diagonal, refinement, symmetry, composition, classifier, ledger, handoff, continuation, package, and naming rows.

Apply Definition 339.12. The public export repackages the same rows for CauchyFilter$^{\uparrow}$ and Completion$^{\uparrow}$ consumers, so the standard bridge has a fixed source packet and a fixed public target packet over the same hsame, Cont, Pkg, and NameCert coordinates.

The Cauchy side is supplied by Definition 339.6 and Definition 339.9. These theorems carry common finite-refinement data across the downstream Cauchy requests without choosing a completion point or identifying streams by quotient equality.

It remains only to record the boundary of the bridge. Since the cited package, public export, and Cauchy handoff theorems contain no rows for choice-based completion, open-cover compactness, host uniformity, or selected limits, the assembled bridge is exactly the finite-entourage standard bridge stated here. □

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket standard bridge`

Theorem 339.14 (UniformSpace finite Cauchy and completion consumer examples)**.** *The finite-entourage bridge for* $\mathsf{UniformSpace}^{\uparrow}$ *yields two concrete downstream consumer examples.*

First, a $\mathsf{CauchyFilter}^{\uparrow}$ *request reads the common refinement row of the finite entourage packet as its finite Cauchy witness ledger. Second, a* $\mathsf{Completion}^{\uparrow}$ *request reads the same finite Cauchy ledger as completion input without selecting a limit and without quotienting streams.*

Both consumers use only the finite entourage list, diagonal witnesses, common refinement rows, symmetry rows, composition rows, endpoint hsame *transport,* Cont *observation windows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$ *row exported by the bridge.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket finite cauchy completion consumer examples`

Proof. Use Definition 339.12. It fixes the public finite-entourage packet containing the finite entourage list, diagonal witnesses, symmetry and composition rows, classifier transport, endpoint hsame rows, Cont windows, package provenance, and local naming row.

Apply Definition 339.13. The standard bridge exposes that packet as the public $\mathsf{UniformSpace}^{\uparrow}$ source for downstream consumers and adds no selected completion point, quotient stream equality, open-cover compactness, host uniformity, or selected limit.

Next use Definition 339.6. It routes the finite $\mathsf{CauchyFilter}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ requests through the same displayed finite-entourage packet.

Finally apply Definition 339.9. The common refinement row supplies the finite Cauchy witness ledger for the $\mathsf{CauchyFilter}^{\uparrow}$ read, and the same ledger is retained by the $\mathsf{Completion}^{\uparrow}$ read without selecting a limit or quotienting streams. □

The metric comparison below consumes the distance-witness carrier of chapter 57: the entourage rows are read as finite filters only after the $\mathsf{Met}^{\uparrow}$ distance witness has supplied the visible-context boundary used by the comparison.

Theorem 339.15 (UniformSpace entourage-filter-metric comparison)**.** *An accepted* $\mathsf{UniformSpace}^{\uparrow}$ *packet compares two point histories only through the displayed entourage rows, filter-neighbourhood rows,* $\mathsf{Met}^{\uparrow}$ *distance witness rows when those witnesses are present, componentwise* hsame *transports on the two endpoints,* Cont *routes, and* Pkg *provenance.*

The comparison exports no host uniformity, open-cover compactness principle, quotient topology equality, choice-selected completion datum, or ambient metric space equality. A downstream $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Filter}^{\uparrow}$*, or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer therefore receives exactly the finite comparison surface carried by the* $\mathsf{UniformSpace}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.UniformSpaceUp.UniformSpacePacket entourage filter metric comparison`

Proof. Project the accepted packet by Definition 339.11. This supplies the point source, finite entourage list, diagonal, refinement, symmetry, composition, transport, continuation, package, and local naming rows over the same BHist surface.

Next replay the filter-neighbourhood witnesses. The finite filterbase diagonal obligation and the diagonal refinement obligation identify each requested neighbourhood with a displayed entourage row and a displayed refinement route, while componentwise hsame carries the two endpoint histories through the same classifier surface.

When a $\mathsf{Met}^{\uparrow}$ distance witness row is present, it is read as one of the neighbourhood anchors already routed through the displayed entourage ledger. The classifier transport obligation preserves that row under endpoint transport, and the common-refinement handoff keeps the filter and Cauchy reads on the same finite continuation routes.

Finally apply Definition 339.8. Since the exactness boundary has no coordinate for host uniformity, open-cover compactness, quotient topology equality, or ambient metric equality, the comparison is exhausted by the displayed entourage, filter, metric-witness, hsame, Cont, and Pkg rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$.

Constructive story: A uniform-space packet is generated by a carried point-history source, a finite list of entourage rows, diagonal witnesses, finite filterbase refinements, symmetry rows, composition rows, endpoint hsame transport, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 339.12, Definition 339.13, and Definition 339.14 export the finite filterbase diagonal, diagonal refinement, symmetry and composition, classifier transport, entourage ledger, common-refinement Cauchy handoff rows, common-refinement functoriality, standard finite-entourage bridge, and finite Cauchy and completion consumer examples as the public $\mathsf{UniformSpace}^{\uparrow}$ downstream packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformSpace}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.UniformSpaceUp.UniformSpacePacket\_standard\_bridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: The chapter does not close choice-based completion, quotient equality of Cauchy streams, open-cover compactness, host uniformity, or selected limits.

Upgrade path: No further formal-axis upgrade remains for this checked finite-entourage standard bridge.

Chapter 340

A Concrete Naming Certificate for $\mathsf{LocatedCauchy}^{\uparrow}$

The $\mathsf{LocatedCauchy}^{\uparrow}$ interface records the finite located-Cauchy packet that sits between explicit Cauchy-rate names and the $\mathsf{Real}^{\uparrow}$ seal. Following Bishop and Bridges, Constructive Analysis, the packet treats located rational ball data as constructive evidence attached to a Cauchy name. It does not form Dedekind cuts, quotient streams, selected real limits, trichotomy witnesses, choice data, or open-cover compactness objects.

Its finite packet consumes the schedule row from chapter 126, endpoint and tolerance rows from chapter 291, precision thresholds from chapter 292, the regular-name handoff from chapter 301, and the seal boundary from chapter 38.

Definition 340.1 (LocatedCauchy finite carrier)**.** *A* $\mathsf{LocatedCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, D, M, W, H, C, P, N)$$

with the following visible coordinates:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the requested precision windows;*
2. *D is the displayed family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read on those windows;*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *precision row that assigns each requested precision to a finite observation threshold;*
4. *W is the family of located dyadic-ball witnesses for each displayed precision request, with center and radius read from the dyadic endpoint rows;*
5. *H records componentwise* hsame *transport for schedules, endpoint rows, modulus rows, and located-ball witnesses;*
6. *C records the* Cont *readback routes through which a consumer reaches finite precision windows, dyadic endpoint rows, modulus rows, and located-ball witnesses;*
7. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as the dependency anchors;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchy}^{\uparrow}}$ *provenance row.*

The carrier predicate is $\mathsf{LocatedCauchyPacket}(S, D, M, W, H, C, P, N)$*. A constant dyadic stream inhabits the carrier by taking S to repeat one scheduled dyadic endpoint, M to return the same*

threshold for every requested precision, and W to display the corresponding degenerate located dyadic ball at each requested window.

Definition 340.2 (LocatedCauchy classifier)**.** *Two accepted* $\mathsf{LocatedCauchy}^{\uparrow}$ *packets* $L = (S, D, M, W, H, C, P, N)$ *and* $L' = (S', D', M', W', H', C', P', N')$ *are classified alike when the displayed schedules agree componentwise, the dyadic endpoint rows agree componentwise, the modulus rows agree on every requested precision window, the located-ball witnesses agree through their displayed endpoint and radius rows, the* hsame *rows transport those components, the* Cont *routes expose the same finite readback surface, and the* Pkg *provenance rows name the same dependency anchors.*

The classifier has no coordinate for host real equality, completed quotient streams, Dedekind-cut membership, trichotomy, choice-selected representatives, or open-cover compactness. It compares only the finite located windows and the provenance rows displayed in Definition 340.1.

Theorem 340.3 (LocatedCauchy window stability)**.** *Let L be an accepted* $\mathsf{LocatedCauchy}^{\uparrow}$ *packet. If a requested precision window is refined through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row, then the refined readback still factors through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule, the same family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, and the transported located dyadic-ball witness family. Thus precision refinement changes only which finite window is read; it does not expose a completed real value or an unbounded stream object.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier window stability`

Proof. Project L through Definition 340.1. The modulus coordinate M is the only row that answers a precision request, and it answers by naming a finite observation threshold inside the schedule S.

At the refined threshold, the endpoint data are read from D and the located ball data are read from W. The transport coordinate H supplies the componentwise hsame rows needed to move these visible rows along an accepted classifier comparison.

The continuation coordinate C exhausts the readback routes. Since no carrier coordinate stores a quotient stream, host real, or selected limit, refinement cannot reveal data outside the displayed finite located window. □

Theorem 340.4 (LocatedCauchy NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedCauchy}^{\uparrow}}$ *seed surface consists of the following five rows:*

1. *carrier inhabitation by the constant dyadic stream described in Definition 340.1;*
2. *classifier reflexivity, symmetry, and transitivity obtained from componentwise* hsame *over schedules, endpoints, modulus rows, located balls, continuation routes, and package provenance;*
3. *carrier transport along the classifier rows of Definition 340.2;*
4. *precision-window stability under modulus refinement as in Definition 340.3;*
5. $\mathsf{Real}^{\uparrow}$ *seal boundary, where consumers see only finite located windows and the dependency provenance needed to pass toward* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier semantic name certificate`

Proof. The carrier row is the finite packet displayed in Definition 340.1. The constant dyadic stream gives an inhabited packet because each requested precision reads the same dyadic endpoint and the same degenerate located ball.

The classifier laws are inherited component by component from the displayed hsame rows. Reflexivity uses the identity rows, symmetry reverses each componentwise comparison, and transitivity composes the visible comparisons on schedules, endpoints, modulus data, located-ball witnesses, continuation routes, and provenance.

Transport repacks the carrier coordinates after those classifier comparisons. Window stability is exactly Definition 340.3, and the final row is the seal boundary isolated in Definition 340.5. □

Theorem 340.5 (LocatedCauchy real seal boundary)**.** *Every* Real$^{\uparrow}$*-facing read exported by an accepted* LocatedCauchy$^{\uparrow}$ *packet factors through a finite requested precision window, its* CauchyModulus$^{\uparrow}$ *threshold, the scheduled* DyadicRatCore$^{\uparrow}$ *endpoint rows, and the displayed located dyadic-ball witness. The packet supplies a constructive real-name neighbor for* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumers, but it claims no Dedekind cut, quotient stream, trichotomy principle, choice-selected representative, or open-cover compactness principle.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier real seal boundary`

Proof. Take any consumer route from the packet toward the real seal. By the carrier definition, every such route is one of the displayed Cont rows in C, and each route names the precision window it reads.

The precision window is resolved by the modulus row M, then by the scheduled endpoint rows in D and the located-ball witnesses in W. These are precisely the finite rows exposed to RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers.

The exclusions follow from absence of coordinates. The packet stores no cut predicate, no quotient-stream equality, no trichotomy witness, no selected representative, and no open-cover compactness datum, so none of those objects can be exported through the real-facing readback. □

Theorem 340.6 (LocatedCauchy scoped dependency package)**.** *Any accepted* LocatedCauchy$^{\uparrow}$ *packet whose displayed rows satisfy Definition 340.1, Definition 340.2, Definition 340.3, Definition 340.4, and Definition 340.5 determines a scoped dependency package over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* CauchyModulus$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

Every Real$^{\uparrow}$*-facing consumer read from that package is exhausted by those displayed rows. The package cannot export a Dedekind cut, quotient stream, host real equality, trichotomy, choice data, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier scoped dependency package`

Proof. Project the accepted packet through Definition 340.1. This exposes the schedule, dyadic-endpoint rows, modulus row, located-ball witnesses, transport rows, continuation routes, package provenance, and local name certificate as one finite BHist surface.

Use Definition 340.2 to transport the exposed rows componentwise. The classifier compares only the displayed schedules, endpoint rows, modulus data, located-ball witnesses, continuation routes, and package provenance, so no external real-equality or quotient coordinate is introduced.

For precision refinement, apply Definition 340.3. It keeps every refined read on the same scheduled finite window surface and preserves the visible DyadicRatCore$^{\uparrow}$, CauchyModulus$^{\uparrow}$, and located-ball rows.

The five local certificate rows are exactly the surface named by Definition 340.4. Finally, Definition 340.5 exhausts every Real$^{\uparrow}$-facing consumer read by the displayed finite rows, which proves the stated exclusions. □

Theorem 340.7 (LocatedCauchy public Real consumer export)**.** *The public* LocatedCauchy$^{\uparrow}$ *export to* Real$^{\uparrow}$ *consumers is exactly the finite packet surface consisting of the* StreamName$^{\uparrow}$ *schedule,* DyadicRatCore$^{\uparrow}$ *endpoint rows,* CauchyModulus$^{\uparrow}$ *precision row, located dyadic-ball witnesses, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Every Real$^{\uparrow}$*-facing consumer read is exhausted by that finite packet surface. The export does not supply Dedekind cuts, quotient streams, host real equality, trichotomy, choice data, selected representatives, or open-cover compactness.*

Proof. Expose the public surface using Definition 340.6. It projects an accepted packet to the displayed schedule, dyadic endpoint rows, modulus data, located-ball witnesses, transport rows, continuation routes, package provenance, and local name certificate over the named BEDC components.

For refined consumer reads, apply Definition 340.3. Refinement changes only which finite threshold and endpoint rows are read from the same packet; it does not add a quotient stream or completed real object.

Finally use Definition 340.5. That boundary exhausts the Real$^{\uparrow}$-facing routes by the finite located windows and records the exclusions for Dedekind cuts, quotient streams, host real equality, trichotomy, choice data, selected representatives, and open-cover compactness. □

Theorem 340.8 (LocatedCauchy finite-window bridge boundary)**.** *Every* Real$^{\uparrow}$*-facing read of an accepted* LocatedCauchy$^{\uparrow}$ *packet factors through exactly the finite* StreamName$^{\uparrow}$ *schedule,* CauchyModulus$^{\uparrow}$ *threshold,* DyadicRatCore$^{\uparrow}$ *endpoint rows, located dyadic-ball witnesses, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of the packet.*

Conversely, every displayed finite window accepted by the carrier is exported by Definition 340.7. The boundary exports no Dedekind cut, quotient stream, host real equality, trichotomy, choice datum, selected representative, or open-cover compactness witness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier finite window bridge boundary`

Proof. Project the accepted packet through Definition 340.1. This gives the finite schedule, modulus threshold, endpoint rows, located-ball witnesses, transport routes, package provenance, and local name-certificate rows.

Use Definition 340.2 to identify consumer reads only componentwise through the displayed rows. Then apply Definition 340.3; a refined read stays on a finite scheduled window and changes neither the carrier nor the visible dependency surface.

Next apply Definition 340.5 and Definition 340.6. These results bind the finite located-Cauchy window to the scoped dependency package over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

For the reverse direction, an accepted displayed finite window is already one of the public packet windows. Therefore Definition 340.7 exports exactly that window to Real$^{\uparrow}$ consumers and supplies no additional row. □

Theorem 340.9 (LocatedCauchy standard finite-packet bridge)**.** *The public* Real$^{\uparrow}$ *consumer export of Definition 340.7 and the finite-window boundary of Definition 340.8 give a conservative standard finite located-Cauchy packet reading over exactly the displayed* StreamName$^{\uparrow}$ *schedule,* DyadicRatCore$^{\uparrow}$ *endpoint rows,* CauchyModulus$^{\uparrow}$ *precision row, located dyadic-ball witnesses, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

This standard packet reading exports no Dedekind cut, quotient stream, host real equality, trichotomy, choice datum, selected representative, or open-cover compactness witness. Every standard

consumer field is read from the displayed finite packet rows, and every displayed finite packet row remains a $\mathsf{LocatedCauchy}^{\uparrow}$ *carrier row.*

Proof. First assemble the scoped dependency package from Definition 340.6. It fixes the BEDC components $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the visible carrier, classifier, transport, continuation, package, and name-certificate rows.

Next read the public $\mathsf{Real}^{\uparrow}$ surface through Definition 340.7. This exports exactly the scheduled finite packet rows and records that no completed real object, quotient stream, or choice-based representative is supplied to the consumer.

Finally apply Definition 340.8. Its forward direction factors every $\mathsf{Real}^{\uparrow}$-facing read through the displayed finite window, and its reverse direction says that every accepted displayed window is already exported by the public surface.

Composing these three readings gives the standard finite located-Cauchy packet bridge. Since each step is conservative over the same finite rows, the bridge cannot introduce Dedekind cuts, host real equality, trichotomy, selected representatives, choice data, quotient coordinates, or open-cover compactness. □

Theorem 340.10 (LocatedCauchy finite bridge consumer completeness)**.** *Every public* $\mathsf{Real}^{\uparrow}$*-facing consumer of the standard finite-packet bridge of Definition 340.9 factors through exactly the rows named by Definition 340.7, Definition 340.8, and Definition 340.9: the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *precision row, located dyadic-ball witnesses, componentwise* hsame*, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows over the scoped dependency package of Definition 340.6.*

Conversely, every accepted finite packet row in that displayed carrier is repacked as the same standard finite bridge surface. The package exports no Dedekind cut, quotient stream, host real equality, trichotomy, selected representative, choice datum, or open-cover compactness witness.

Proof. Start with the scoped dependency package of Definition 340.6. It fixes the only BEDC components available to the construction: $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the carrier, classifier, transport, continuation, package, and name-certificate rows visible in the finite located-Cauchy packet.

For the forward direction, apply Definition 340.7 to restrict each $\mathsf{Real}^{\uparrow}$-facing read to the public packet surface. Then apply Definition 340.8 and Definition 340.9; together they say that the standard bridge reads exactly the displayed finite schedule, endpoint, modulus, located-ball, transport, continuation, package, and certificate rows.

For the reverse direction, take an accepted row of the displayed finite packet. The finite-window boundary already identifies it as a public packet row, and the public export sends precisely those rows to the $\mathsf{Real}^{\uparrow}$ consumer surface. The standard finite-packet bridge does not add a second consumer surface; it only repackages the same accepted rows under the standard finite bridge surface.

All exclusions are inherited from the same three results. Since none of their carriers contains a Dedekind cut, quotient stream, host real equality, trichotomy witness, selected representative, choice datum, or open-cover compactness witness, the consumer-completeness package has no coordinate from which such data could be read. □

Theorem 340.11 (LocatedCauchy bridge readback consumer factorization)**.** *Every public consumer read of a* $\mathsf{LocatedCauchy}^{\uparrow}$ *finite-window bridge factors through the displayed* BHist *schedule,*

dyadic endpoint rows, CauchyModulus$^{\uparrow}$ *row, located dyadic-ball witnesses, componentwise transport rows, displayed route rows, package provenance,* Cont, Pkg, hsame, *and local* NameCert *rows supplied by Definition 340.6, Definition 340.8, and Definition 340.10.*

The factorization has no hidden readback coordinate for host real equality, quotient streams, selected representatives, choice data, trichotomy, or open-cover compactness. Any consumer output admitted by the finite bridge is therefore a read of the same finite BEDC rows already present in the scoped dependency package.

Proof. Apply Definition 340.8. It exposes the bridge row as the displayed finite-window surface and identifies every Real$^{\uparrow}$-facing read with the scheduled finite packet rows.

Next apply Definition 340.6. This recovers the permitted consumer readback coordinates: the BHist schedule, endpoint rows, modulus row, located dyadic-ball witnesses, transport rows, route rows, provenance, Cont, Pkg, hsame, and local NameCert data.

Finally apply Definition 340.10. It exhausts the public consumer surface over those rows and excludes host real equality, quotient streams, selected representatives, choice data, trichotomy, and open-cover compactness as possible readback coordinates. □

Closure status for NameCert$_{\text{LocatedCauchy}^{\uparrow}}$.

Constructive story: A LocatedCauchy$^{\uparrow}$ packet is generated from a finite StreamName schedule, DyadicRatCore endpoint rows, a CauchyModulus precision row, located dyadic-ball witnesses, componentwise transport rows, continuation readback routes, package provenance for DyadicRatCoreUp, StreamNameUp, CauchyModulusUp, RegSeqRatUp, and RealUp, and a local NameCert provenance row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 340.6, Definition 340.7, Definition 340.8, Definition 340.9, Definition 340.10, and Definition 340.11 bind the scoped package, public Real$^{\uparrow}$ consumer export, finite-window boundary, standard finite-packet bridge, complete public consumer surface, and bridge readback factorization over BHist, hsame, Cont, Pkg, NameCert, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchy}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LocatedCauchyUp.LocatedCauchyCarrier\_finite\_window\_bridge\_boundary`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Dedekind cuts, quotient streams, host real equality, trichotomy, choice, or open-cover compactness.

Upgrade path: No further theory-axis grade is available for this package; remaining work belongs to the verification axis.

Theorem 340.12 (LocatedCauchy constant window degeneracy)**.** *Let $L = (S, D, M, W, H, C, P, N)$ be the constant dyadic-stream* LocatedCauchy$^{\uparrow}$ *carrier described in Definition 340.1. Thus there is one displayed dyadic endpoint row d such that S repeats the scheduled endpoint d, the modulus row M returns the same observation threshold for every requested precision, and W displays the degenerate located dyadic ball attached to d at each requested window.*

Then every consumer-visible precision request for L reads the same degenerate dyadic endpoint. More precisely, for any requested precision window q, the Cont *readback route selected by C factors through the* StreamName$^{\uparrow}$ *row that schedules d, the* CauchyModulus$^{\uparrow}$ *row that returns the constant threshold, the* DyadicRatCore$^{\uparrow}$ *endpoint row d, and the located-ball witness in W whose left endpoint, center endpoint, and right endpoint are all the displayed row d. Hence the classifier of Definition 340.2 sees no window-dependent endpoint variation in the constant packet: changing q changes only the name of the finite request, not the dyadic endpoint or the degenerate located ball read at that request.*

Proof. Project the carrier through Definition 340.1. The constant-stream clause fixes the visible coordinates: the schedule S repeats one scheduled dyadic endpoint row d, the modulus row M

returns the same threshold for each requested precision, and the witness family W assigns to each requested window the degenerate located dyadic ball corresponding to that same endpoint.

Let q be a requested precision window. The only route from a consumer request to packet data is one of the continuation rows recorded in C. By the carrier definition, such a route first reads the requested window through S and M, then reads the endpoint data from D, and then reads the located-ball witness from W. Since S is constant and M returns the fixed threshold, this route reaches the same scheduled endpoint row d for the request q as for any other request.

At that row, D is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint family of the packet, and the constant-stream clause says that the located witness attached by W is degenerate at d. Thus the left endpoint, center endpoint, and right endpoint exposed by the located dyadic ball are all the same displayed dyadic row d.

Finally, Definition 340.2 compares only the displayed schedules, endpoint rows, modulus rows, located-ball witnesses, transport rows, continuation routes, and provenance. None of those coordinates contains a second endpoint for the constant packet, so a change of requested precision window cannot introduce endpoint variation. It only selects another finite request whose readback again reaches the same degenerate dyadic ball at d. □

Theorem 340.13 (LocatedCauchy tail-window seal factorization)**.** *Let $L = (S, D, M, W, H, C, P, N)$ be an accepted $\mathsf{LocatedCauchy}^{\uparrow}$ packet over the carrier of Definition 340.1. For every tail precision window obtained by refining the displayed $\mathsf{CauchyModulus}^{\uparrow}$ row M, the consumer read selected by the continuation routes in C factors through the same seal boundary as the $\mathsf{Real}^{\uparrow}$ surface of Definition 340.5: the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S, the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint family D, the located dyadic-ball witness family W, the componentwise transport rows H, the displayed Cont routes C, the Pkg provenance P, and the local $\mathsf{NameCert}$ rows N.*

The factorization is stable under later finite reads. Refining M may select a later threshold and a later finite endpoint window, but it does not cross the seal boundary or expose a second $\mathsf{Real}^{\uparrow}$ carrier. In the constant-packet boundary case of Definition 340.12, every tail window reads the same degenerate endpoint row and the same degenerate located dyadic ball.

Thus no tail-window consumer read exports a Dedekind cut, quotient stream, host real equality, trichotomy witness, selected representative, choice datum, or open-cover compactness witness.

Proof. First apply Definition 340.3. A refinement of the displayed modulus row changes only the finite threshold and endpoint window selected inside the accepted packet. The scheduled stream row, endpoint family, located-ball witness family, transport rows, continuation routes, package provenance, and local certificate rows remain the displayed rows of L.

Next use Definition 340.5. It identifies the $\mathsf{Real}^{\uparrow}$-facing seal boundary with those same displayed rows and records that the public real surface has no coordinate for completed cuts, quotient streams, host equality, trichotomy, selected representatives, choice data, or open-cover compactness.

Then apply Definition 340.8. Its finite-window bridge says that every accepted finite read through a refined modulus window is already a read of the scheduled stream, dyadic endpoints, located witnesses, transport, continuation, package, and certificate rows fixed by the seal boundary.

For the constant-packet boundary case, invoke Definition 340.12. The refined tail request again reaches the same endpoint row d and the same degenerate located ball at d, so the preceding factorization specializes without adding a window-dependent endpoint.

Composing these results gives the claimed tail-window factorization. Since each step remains inside the displayed BEDC rows of the accepted packet, the consumer receives no data outside the RealUp seal boundary. □

Chapter 341

A Concrete Naming Certificate for $\mathsf{PolePlacement}^{\uparrow}$

The $\mathsf{PolePlacement}^{\uparrow}$ interface records the finite closed-loop ledger used by control certificates that choose a displayed feedback row and target a displayed characteristic-polynomial row. In the standard finite-dimensional control setting of Chen, *Linear System Theory and Design* (1999), pole placement relates the state matrix, input matrix, feedback gain, and requested closed-loop characteristic polynomial. The BEDC surface below names only the finite rows of that relation. It exports no analytic spectrum, no host trajectory space, and no stabilizing feedback existence theorem.

The state and input rows are consumed from chapter 329, controllability evidence from chapter 297, and matrix, vector-space, and linear-map rows from chapter 49, chapter 47, and chapter 48.

Definition 341.1 (PolePlacement finite BHist carrier)**.** *A* $\mathsf{PolePlacement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Pi = (X, U, A, B, K, C, T, H, R, P, L)$$

whose visible rows are:

1. *a state row X and an input row U read through* $\mathsf{VecSpace}^{\uparrow}$ *and the state/input face of* $\mathsf{StateSpaceModel}^{\uparrow}$*;*
2. *a transition matrix row A and input matrix row B read through* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{LinMap}^{\uparrow}$*;*
3. *a feedback gain row K read as a finite matrix and linear-map row from X to U;*
4. *a closed-loop row C whose displayed matrix ledger records $C = A + BK$ over the same state carrier;*
5. *a target characteristic row T and characteristic-comparison ledger H recording the requested finite polynomial row and its displayed comparison with the characteristic row of C;*
6. hsame *comparisons for the state, input, A, B, K, C, target, characteristic, continuation, and provenance rows;*
7. Cont *routes R for reading multiplication BK, addition $A + BK$, closed-loop readback, and characteristic-row comparison;*
8. *a* Pkg *provenance row P naming* $\mathsf{StateSpaceModel}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$*;*
9. *a ledger boundary row L saying that no host trajectory space, analytic spectrum, eigenvalue multiplicity theorem, or stabilizing feedback theorem is part of the carrier.*

Two accepted packets classify together only when every displayed row in this inventory is transported componentwise by the recorded hsame *comparisons and every matrix, linear-map, closed-loop, and target-polynomial read follows one of the recorded* Cont *routes.*

Lean4 (definition) `BEDC.Derived.PolePlacementUp.PolePlacementSourcePacket`

Theorem 341.2 (PolePlacement closed-loop ledger)**.** *Every accepted* $\mathsf{PolePlacement}^{\uparrow}$ *packet exposes exactly the finite state row X, input row U, transition row A, input row B, gain row K, closed-loop row C, target characteristic row T, characteristic-comparison ledger H,* hsame *rows,* Cont *routes, and* Pkg *provenance rows listed in Definition 341.1. The closed-loop readback is the displayed ledger $C = A + BK$ and no additional host closed-loop system is exported.*

Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementSourcePacket closed loop ledger`
Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementCarrier closed loop ledger`

Proof. Project the accepted carrier of Definition 341.1. The projection exposes the finite state, input, matrix, gain, closed-loop, target, comparison, transport, continuation, provenance, and boundary rows.

The closed-loop row is read through the displayed matrix multiplication and addition routes. The multiplication route supplies the BK ledger, the addition route supplies the $A + BK$ ledger, and the closed-loop row C is the recorded endpoint of that finite calculation.

Transporting the packet replaces each coordinate by its displayed hsame match and follows the same Cont routes. Since the carrier has no coordinate for a host trajectory space or analytic closed-loop object, every consumer-visible closed-loop read is exhausted by the finite ledger. □

Theorem 341.3 (PolePlacement NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PolePlacement}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: an accepted packet is inhabited by choosing displayed* $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{LinMap}^{\uparrow}$ *source rows and recording the gain, closed-loop, target, comparison, transport, continuation, and provenance rows of Definition 341.1;*
2. *classifier transport: accepted packets compare by componentwise* hsame *transport of X, U, A, B, K, C, T, H, R, P, and L;*
3. *closed-loop ledger exactness: the only closed-loop row exported by the packet is the displayed $C = A + BK$ ledger of Definition 341.2;*
4. *target-polynomial ledger: every target read is the finite row T together with the comparison ledger H against the characteristic row of C;*
5. *downstream consumer boundary:* $\mathsf{Riccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, *and* $\mathsf{StateSpaceModel}^{\uparrow}$ *consumers may read only the finite state, input, matrix, gain, closed-loop, target, comparison, transport, continuation, and provenance rows of the same packet.*

Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementSourcePacket namecert obligation surface`

Proof. The habitation row is a finite packet assembly. Each required coordinate is a displayed BHist row, and the provenance row names the consumed state-space, controllability, matrix, vector-space, and linear-map certificates.

Classifier transport is componentwise. Transporting X, U, A, B, K, C, T, H, R, P, and L along the displayed hsame rows preserves the same carrier inventory and the same matrix and characteristic-read routes.

The closed-loop obligation is exactly Definition 341.2. The target-polynomial obligation is the corresponding projection of T and H, so it records a finite comparison row rather than an analytic spectral theorem.

Finally, the downstream boundary is the package projection. Since the package row names only the displayed control and algebra certificates, the listed consumers cannot recover host trajectories, unlogged spectra, or feedback existence principles from the $\mathsf{PolePlacement}^{\uparrow}$ surface. □

Theorem 341.4 (PolePlacement consumer boundary)**.** *Every downstream read from an accepted* $\mathsf{PolePlacement}^{\uparrow}$ *packet factors through the finite rows of Definition 341.1. Riccati, LQR, Kalman-Filter, and StateSpaceModel consumers may use the displayed* $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* Pkg *rows, but they cannot demand an analytic spectrum, a host trajectory space, a stabilizing controller theorem, a controllability-rank proof, or an external feedback solver from this certificate.*

Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementSourcePacket consumer boundary`

Proof. Take a consumer-visible readback from an accepted packet. By Definition 341.1, the readback is a projection from the finite carrier rows or from the provenance row naming the consumed certificates.

Apply Definition 341.2 to the closed-loop part. This identifies the only available feedback read as the displayed $A + BK$ row and its target characteristic comparison.

The listed consumers therefore receive only finite BEDC rows and their transport and continuation data. Any analytic spectrum, host trajectory, stabilizing-controller theorem, rank proof, or external solver would require a coordinate absent from the packet. □

Theorem 341.5 (PolePlacement standard finite bridge)**.** *The finite* $\mathsf{PolePlacement}^{\uparrow}$ *packet of Definition 341.1, together with the closed-loop ledger, NameCert obligation surface, and consumer boundary, interprets as the standard finite pole-placement schema: a state row, input row, transition matrix row A, input matrix row B, gain row K, closed-loop row $A + BK$, and requested target characteristic row are all displayed as finite* BHist *rows with explicit transport, continuation, and provenance. This bridge exports no analytic spectral assumption, pole-existence theorem, host trajectory semantics, or feedback optimality claim.*

Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementSourcePacket standard finite bridge`

Proof. Begin with the carrier. It gives the state and input rows, the matrix and linear-map rows for A, B, and K, the displayed closed-loop endpoint, and the target characteristic ledger.

Use Definition 341.2 to identify the closed-loop schema field with the finite $A+BK$ calculation. Use Definition 341.3 to package the carrier, classifier transport, ledger exactness, target-polynomial ledger, and downstream boundary as the local name-certificate surface.

Apply Definition 341.4 to restrict every public read to the same finite rows. The resulting interpretation is exactly the standard finite pole-placement data shape, while the excluded analytic and optimality claims remain outside the certificate. □

Theorem 341.6 (PolePlacement scoped dependency packet)**.** *The closed-loop ledger of Definition 341.2, target-polynomial ledger and certificate surface of Definition 341.3, consumer boundary of Definition 341.4, and standard finite bridge of Definition 341.5 form one scoped dependency packet. Its only source rows are the displayed* BHist *state, input, matrix, gain, closed-loop, target, characteristic-comparison,* hsame, Cont, Pkg, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, *and* $\mathsf{ControlControllability}^{\uparrow}$ *rows.*

Proof. Project the packet through Definition 341.1. The available coordinates are exactly the finite state and input rows, matrix rows, gain row, closed-loop row, target row, characteristic-comparison row, transports, continuations, and package provenance named in the statement.

Apply Definition 341.2 to identify the closed-loop projection, and apply Definition 341.3 to bind the target-polynomial ledger and certificate obligations to the same carrier rows.

For every consumer projection, use Definition 341.4; for the standard finite data shape, use Definition 341.5. Both results factor through the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, BHist, hsame, Cont, and Pkg rows.

Repacking these projections gives a single scoped dependency packet. No analytic spectrum, host trajectory space, unlogged rank proof, or external feedback solver is introduced, because no such coordinate occurs in the projected rows. □

Theorem 341.7 (PolePlacement public consumer export)**.** *The scoped dependency packet of Definition 341.6 exports a public finite-control interface to* $\mathsf{Riccati}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{KalmanFilter}^{\uparrow}$*, and* $\mathsf{StateSpaceModel}^{\uparrow}$ *consumers. The exported interface consists exactly of the state row, input row, transition matrix row, input matrix row, gain row, closed-loop row* $A + BK$*, target characteristic row, characteristic-comparison ledger, displayed* hsame *rows,* Cont *routes, and* Pkg *provenance over* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{StateSpaceModel}^{\uparrow}$*, and* $\mathsf{ControlControllability}^{\uparrow}$*.*

It exports no analytic spectrum, host trajectory space, stabilizing feedback theorem, controllability-rank completeness, feedback optimality, or external solver.

Lean4 ✓ `BEDC.Derived.PolePlacementUp.PolePlacementCarrier public consumer export`

Proof. Start with the closed-loop ledger of Definition 341.2. It identifies the displayed transition, input, gain, and $A + BK$ rows as finite ledger coordinates.

Apply Definition 341.3 to attach the target characteristic row, characteristic-comparison ledger, displayed hsame rows, Cont routes, and package provenance to the same accepted carrier.

Use Definition 341.4 for the downstream read boundary, and use Definition 341.5 for the standard finite-control packet shape. These two results identify exactly the $\mathsf{Riccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, and $\mathsf{StateSpaceModel}^{\uparrow}$ consumer surface named in the statement.

Finally apply Definition 341.6. Since that packet has no coordinate for analytic spectra, host trajectory spaces, stabilizing feedback existence, rank-completeness, optimality, or solver data, none of those objects can be exported by the public interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolePlacement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PolePlacement}^{\uparrow}$ packet is seeded by finite BHist rows for state, input, transition matrix, input matrix, gain, closed-loop matrix $A + BK$, target characteristic row, characteristic-comparison ledger, hsame transports, Cont routes, and Pkg provenance over the state-space, controllability, matrix, vector-space, and linear-map certificates.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PolePlacement}^{\uparrow}}$, publicClosed)

Scope closed: Definition 341.6 binds the finite pole-placement carrier, closed-loop ledger, target-polynomial ledger, NameCert obligation rows, consumer boundary, standard finite data shape, and displayed BHist, hsame, Cont, Pkg, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, and $\mathsf{ControlControllability}^{\uparrow}$ dependencies, while Definition 341.7 exposes the downstream finite-control public export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PolePlacement}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic spectral theory, pole-existence theorems, stabilizing feedback existence, controllability-rank completeness, host trajectory semantics, feedback optimality, or a machine-checked standard bridge.

Upgrade path: Bridged closure requires a standard finite-control bridge over the same displayed state, input, matrix, feedback, target-polynomial, transport, continuation, package, and consumer rows.

Chapter 342

A Concrete Naming Certificate for $\mathsf{ContinuedFraction}^{\uparrow}$

The $\mathsf{ContinuedFraction}^{\uparrow}$ interface names finite continued-fraction observation packets as constructive real-name neighbors. Following Khinchin's continued-fraction treatment, the BEDC surface records partial quotients, convergent rows, certified dyadic error radii, stream observations, regular rational handoff rows, and a real-name boundary without completing an infinite expansion or quotienting tails.

The partial-quotient spine consumes the $\mathsf{Nat}^{\uparrow}$ carrier of chapter 28. The convergent numerator and denominator rows consume the rational arithmetic surface of chapter 37, and the certified error-radius row consumes chapter 291. The scheduled handoff and seal route then consume chapter 126, chapter 301, and chapter 38, locating the finite continued-fraction carrier in the Real-completion dependency lattice.

Definition 342.1 (ContinuedFraction carrier packet)**.** *A* $\mathsf{ContinuedFraction}^{\uparrow}$ *carrier packet is a finite* BHist *record*

$$C = (a, p, q, e, s, r, b, \ell, P)$$

with the following visible rows:

1. *a is the finite partial-quotient spine over* $\mathsf{Nat}^{\uparrow}$*;*
2. *p and q are the convergent numerator and denominator rows read over* $\mathsf{Rat}^{\uparrow}$*, including the displayed recurrence ledger for each finite index;*
3. *e is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-radius row attached to the displayed convergent window;*
4. *s is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule through which finite convergent windows are exposed;*
5. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row built from the scheduled rational convergents and their dyadic radius rows;*
6. *b is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row that consumes the regular sequence surface without selecting a completed real value;*
7. *ℓ is the finite ledger naming the recurrence rows, endpoint rows, transport windows, and dependency anchors;*
8. *P is the* Pkg *provenance row for the carrier and the local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *registration.*

The carrier accepts exactly those packets whose rows occur in one finite history and whose recurrence, endpoint, schedule, handoff, boundary, and provenance entries are connected by displayed hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket`

Definition 342.2 (ContinuedFraction classifier)**.** *Two accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packets classify together when every shared finite convergent window has classifier-equal numerator, denominator, and dyadic radius endpoint rows through the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *classifiers, and when the transported* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *recurrence-ledger, and* Pkg *rows agree through the same displayed* hsame *and* Cont *routes.*

Theorem 342.3 (ContinuedFraction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier inhabitation by the empty partial-quotient spine or by a singleton convergent with visible* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* Pkg *rows;*
2. *classifier transport along the componentwise* hsame *and* Cont *routes named in Definition 342.2;*
3. *recurrence-ledger stability for the finite convergent numerator and denominator rows;*
4. *finite-window* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff for scheduled convergent windows and dyadic radius rows;*
5. $\mathsf{Real}^{\uparrow}$ *boundary exactness through the displayed seal row and no other real-facing coordinate.*

Proof. For habitation, use either the empty spine with an empty scheduled window or a singleton spine whose first convergent is carried by the displayed rational endpoint rows. The dyadic radius, stream schedule, handoff row, boundary row, ledger, and package provenance are then the corresponding finite rows in the same BHist packet.

For transport, project the carrier coordinates of Definition 342.1. The classifier of Definition 342.2 compares only shared finite windows, so every transported row is one of the named numerator, denominator, radius, schedule, handoff, boundary, ledger, or provenance coordinates.

For stability and exactness, read the recurrence ledger before the handoff rows. The ledger records the finite recurrence steps for convergents, the handoff consumes only scheduled finite windows, and the real-facing boundary is the single seal row b. □

Theorem 342.4 (ContinuedFraction RegSeqRat handoff)**.** *Every accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet exports a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff only through scheduled finite convergent windows, their* $\mathsf{Rat}^{\uparrow}$ *numerator and denominator rows, their* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, the displayed recurrence ledger, and the accompanying* hsame, Cont, *and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket regseqrat handoff`

Proof. Project an accepted packet to the rows a, p, q, e, s, r, ℓ, and P. The schedule s selects the finite convergent windows, while p and q provide the rational endpoint rows and e provides the dyadic radius rows for those windows.

The handoff row r is indexed by exactly that scheduled data. Transport of a handoff comparison therefore factors through the componentwise hsame rows, the displayed Cont routes, and the package provenance already present in the carrier packet.

No coordinate of the carrier stores an infinite tail, a quotient class of tails, or a completed real. The handoff is consequently a finite regular-sequence surface over the visible convergent ledger. □

Theorem 342.5 (ContinuedFraction L10 sibling dependency route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *boundary read of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet first factors through the existing L10 sibling route chapter 291, chapter 126, chapter 301, and chapter 38. Only after those* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows have been read through the displayed carrier route may the finite continued-fraction seal be consumed.*

Proof. Project the carrier of Definition 342.1. The rows e, s, r, b are exactly the dyadic-radius row, stream observation schedule, regular-rational handoff, and real seal boundary; there is no second coordinate for a real-facing read outside these rows.

Apply Definition 342.4. It forces the $\mathsf{RegSeqRat}^{\uparrow}$ boundary through the scheduled finite convergent windows, their $\mathsf{Rat}^{\uparrow}$ numerator and denominator rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, and the displayed hsame, Cont, and Pkg provenance.

The $\mathsf{Real}^{\uparrow}$ read then consumes the same handoff row before it reaches the seal coordinate b. Thus the boundary route is the concrete-instance sibling lattice route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, followed by the finite continued-fraction seal and no kernel-only island remains. □

Theorem 342.6 (ContinuedFraction RealUp boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row and the seal boundary row carried by Definition 342.1. It exposes the same finite convergent, radius, schedule, recurrence, transport, and provenance ledger, and it does not assert total continued-fraction expansion for every real, uniqueness of infinite expansions, measure-theoretic typicality, or quotient equality of infinite tails.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket real boundary`
↪ variant `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket real boundary scope`

Proof. Read the boundary coordinate b of the carrier. Its source is the regular sequence handoff row r, so any real-facing consumer first passes through the scheduled convergent windows and their rational and dyadic endpoint rows.

The classifier has no separate real-equality clause. Boundary comparisons are therefore inherited from the finite-window $\mathsf{RegSeqRat}^{\uparrow}$ handoff and from the transport, continuation, ledger, and provenance rows already visible in the packet.

Since the carrier stores only a finite partial-quotient spine and finite window data, it cannot provide a total expansion theorem, an infinite-tail uniqueness theorem, a typicality theorem, or a quotient equality principle. The $\mathsf{Real}^{\uparrow}$ boundary is exactly the displayed seal surface. □

Theorem 342.7 (ContinuedFraction digit-tail stability)**.** *Restrict an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet to a displayed finite digit tail. The restricted packet preserves the same* $\mathsf{Rat}^{\uparrow}$ *convergent rows,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *boundary ledger visible in the original packet, with no host infinite expansion and no selected real endpoint.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket digit tail stability`

Proof. Project the carrier of Definition 342.1 to its finite partial-quotient spine, convergent rows, schedule, handoff, boundary row, recurrence ledger, and provenance. The classifier of Definition 342.2 compares only the displayed digit and window rows.

Apply Definition 342.3 to keep the carrier and classifier obligations inside the finite ledger. The restriction to a digit tail is therefore a projection of the same recorded rows, not a passage to an infinite expansion object.

The projected schedule feeds the same finite convergent windows into Definition 342.4. The real-facing read is then the boundary of Definition 342.6, so the restricted tail inherits the displayed $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ ledger without selecting an endpoint. □

Theorem 342.8 (ContinuedFraction recurrence ledger obligation)**.** *For every accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *carrier packet, the partial-quotient spine, the numerator and denominator recurrence rows, and the dyadic radius rows are finite* BHist *rows. Their comparisons are stable exactly along the displayed* hsame *and* Cont *routes named by the carrier ledger, and no recurrence datum is read outside those finite rows.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket recurrence ledger transport`

Proof. Project the packet of Definition 342.1 to the rows a, p, q, e, and ℓ. These are coordinates of one finite BHist record, so the partial quotient, recurrence, and radius data are already finite rows.

The accepted-carrier clause requires the coordinates to be connected by displayed hsame and Cont routes. Hence transport of a recurrence row or a radius row is componentwise transport along the ledger routes, with no extra tail coordinate available. □

Theorem 342.9 (ContinuedFraction finite handoff obligation)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet is exhausted by scheduled finite convergent windows, their* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row. Every handoff read factors through those rows together with the displayed recurrence and dyadic-radius ledger.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket finite handoff obligation`

Proof. Project the carrier to the schedule s, handoff row r, ledger ℓ, package row P, and local name-certificate registration. By Definition 342.1, the schedule selects the visible finite convergent windows and the handoff consumes precisely that scheduled surface.

The package row supplies provenance for the same carrier packet, and the local $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ row registers the displayed classifier discipline. Thus a handoff read can be reconstructed from the scheduled finite windows, the recurrence and radius rows, P, and the local certificate row. □

Theorem 342.10 (ContinuedFraction real-boundary obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *boundary read of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet factors through the same finite* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff used by the carrier. The boundary exposes no completed real, no quotient tail, and no host equality beyond the finite handoff, boundary row,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *dependency rows.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket real boundary transport`

Proof. Project the packet to the handoff row r, boundary row b, ledger ℓ, and package row P. The boundary coordinate in Definition 342.1 is explicitly the $\mathsf{Real}^{\uparrow}$ seal row fed by the regular-sequence handoff.

Since the carrier has no coordinate for a completed real, an infinite quotient tail, or host equality, no boundary read can expose such an object. Its data are the finite handoff rows, the seal boundary, the package provenance, and the local name-certificate dependency rows already present in the packet. □

Theorem 342.11 (ContinuedFraction convergent-window standard bridge)**.** *A finite* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet determines a conservative standard finite convergent window. Its BEDC readback recovers exactly the* $\mathsf{Nat}^{\uparrow}$ *partial quotient spine,* $\mathsf{Rat}^{\uparrow}$ *convergent rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, finite ledger,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row.*

This bridge excludes total expansion, uniqueness of infinite expansions, measure-typicality, quotient tails, and completed real equality.

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket convergent window standard bridge`

Proof. Project the finite continued-fraction carrier and classifier. The carrier coordinates supply the partial quotient spine, rational convergent rows, dyadic radius row, observation schedule, handoff row, seal boundary, finite ledger, package provenance, and local name-certificate evidence.

Restrict the digit spine to the displayed finite window. The recurrence stability and finite handoff obligation identify the same $\mathsf{Rat}^{\uparrow}$ convergent rows and $\mathsf{DyadicRatCore}^{\uparrow}$ radius row at every displayed position, so the standard window has no additional convergent coordinate.

Compose Definition 342.9 with Definition 342.10. The handoff sends the finite convergent window through $\mathsf{RegSeqRat}^{\uparrow}$, and the real-boundary obligation seals the resulting $\mathsf{Real}^{\uparrow}$ row without adding quotient-tail or completed-real equality data.

Reading the conservative standard window back componentwise therefore recovers the original BEDC finite ledger, package provenance, and local name-certificate row. Since every step stays inside the finite carrier and classifier rows, the bridge contains no total expansion theorem, no uniqueness statement for infinite expansions, no measure-typicality claim, no quotient tail, and no completed real equality. □

Theorem 342.12 (ContinuedFraction scoped boundary)**.** *The carrier of Definition 342.1, classifier of Definition 342.2, NameCert obligations of Definition 342.3,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 342.4, recurrence ledger of Definition 342.8, finite handoff of Definition 342.9, and* $\mathsf{Real}^{\uparrow}$ *boundary of Definition 342.10 assemble one scoped* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet.*

The packet is over finite partial quotient windows, rational convergent rows, dyadic radius rows, scheduled $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Use Definition 342.1 and Definition 342.2 to identify the visible carrier and classifier rows. The obligation surface is recorded by Definition 342.3.

The recurrence and finite handoff rows are supplied by Definition 342.8 and Definition 342.9. The external handoff and boundary are supplied by Definition 342.4, Definition 342.6, and Definition 342.10.

Repack these finite rows as the consumer boundary for the accepted $\mathsf{ContinuedFraction}^{\uparrow}$ packet. The displayed history, equality, continuation, package, and local certificate dependencies account for every row used by the packet, so no convergence theorem for arbitrary continued fractions is claimed. □

Theorem 342.13 (ContinuedFraction finite-packet standard bridge)**.** *The finite partial-quotient spine, rational convergent rows, dyadic radius rows, scheduled observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *boundary, displayed* hsame *rows,* Cont *read routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet assemble the standard finite continued-fraction packet boundary for* $\mathsf{ContinuedFraction}^{\uparrow}$*. The assembled boundary is exactly*

the finite packet surface; it excludes total expansion of every real, uniqueness of infinite tails, quotient equality, and measure-theoretic claims.

Proof. First build the finite handoff surface. Apply Definition 342.4 to the scheduled convergent windows, rational numerator and denominator rows, dyadic radius rows, recurrence ledger, hsame transport, Cont routes, and Pkg provenance.

Next attach the real-facing seal. Apply Definition 342.6 and Definition 342.10; both read the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff and boundary coordinate, so the real row contributes no separate completed real, quotient-tail equality, or host equality principle.

Finally repack the displayed rows through Definition 342.12. The scoped packet already contains the finite spine, convergent rows, radius rows, schedule, handoff, boundary, transport, continuation, package, and local name-certificate rows, so it is the standard finite continued-fraction packet boundary and nothing stronger. □

Theorem 342.14 (ContinuedFraction prefix convergent window)**.** *Every finite prefix of a certified* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet determines a convergent numerator and denominator window whose* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff reads only the displayed digit, recurrence, tail,* hsame, Cont*, package, and local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *rows.*

The window contributes no completed real, quotient tail equality, total expansion theorem, or host equality principle.

Proof. Begin with the scheduled convergent read of Definition 342.4. It identifies the finite partial-quotient prefix, numerator and denominator rows, dyadic radius row, observation schedule, hsame transport, Cont routes, and package provenance available to the regular-sequence consumer.

Next use Definition 342.7. The digit and tail rows remain within the same finite prefix surface, so extending or restricting a prefix does not add an external host equality or quotient-tail coordinate.

Apply Definition 342.8 to bind the convergent numerator and denominator rows to the displayed recurrence ledger. Then repack the same rows through Definition 342.13. The resulting consumer window is exactly the finite continued-fraction packet surface named in the statement. □

Theorem 342.15 (ContinuedFraction rational-interval window handoff)**.** *Every accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet whose convergent window is read as a* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint window exposes exactly the finite partial-quotient spine, the* $\mathsf{Rat}^{\uparrow}$ *numerator and denominator rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row.*

The handoff exports no completed real, quotient tail, total expansion theorem, or interval arithmetic law.

Proof. First read the scheduled convergent rows through Definition 342.4. This identifies the finite partial-quotient spine, rational numerator and denominator rows, dyadic radius row, stream schedule, hsame transports, Cont routes, and package provenance as the regular rational source seen by the consumer.

Next apply Definition 342.9. The finite handoff exhaustion accounts for the displayed recurrence and endpoint window rows, so the $\mathsf{RationalInterval}^{\uparrow}$ consumer can only read the finite endpoint shape already present in the packet.

Finally use Definition 342.6. The real-facing seal is the boundary row of the same finite handoff, while the rational-interval endpoint and containment rows are only its finite consumer shape. Thus no completed real, quotient tail, total expansion theorem, or interval arithmetic law is exported. □

Lean4 ✓ `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket rational interval window handoff certificate`

Theorem 342.16 (ContinuedFraction nested cylinder boundary)**.** *Let an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet display two finite prefix windows whose smaller prefix is obtained by restricting the larger finite partial-quotient spine. Then the smaller* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint window are exactly the restriction of the larger packet's finite partial-quotient spine, recurrence ledger, dyadic radius row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *rows.*

The boundary exports no completed real, quotient tail, total expansion theorem, or interval arithmetic law.

Proof. First compare the two displayed prefix windows. Apply Definition 342.7 to see that restricting the larger finite prefix preserves exactly the digit, tail, scheduled observation, hsame, Cont, package, and local name-certificate rows already visible in the smaller prefix.

Next read both prefixes through Definition 342.14. The convergent numerator and denominator rows, recurrence ledger, dyadic radius row, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the smaller window are therefore the restricted rows of the larger packet, with no extra host equality or quotient-tail coordinate.

Finally apply Definition 342.15 and Definition 342.10. These identify the rational-interval endpoint window and real-facing boundary as consumer views of the same finite handoff, so the nested cylinder boundary is precisely the finite restriction stated above and exports nothing beyond that surface. □

Theorem 342.17 (ContinuedFraction common-refinement cylinder exhaustion)**.** *Let P and Q be accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packets whose finite prefix cylinders overlap, and let a finite consumer compare the two packets only through that overlap. Then the comparison refines to a displayed common prefix cylinder, and the common consumer surface is exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{Real}^{\uparrow}$ *boundary row, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row.*

The common prefix cylinder exports no completed real equality, quotient tail, total expansion, or interval arithmetic law.

Proof. First apply Definition 342.16 to the overlapping finite prefix cylinders. It restricts both accepted packets to the same displayed prefix rows: finite partial quotients, recurrence ledger, scheduled observations, dyadic radius row, transports, routes, provenance, and local name-certificate data.

Next apply Definition 342.15. The restricted prefix is read as the rational-interval endpoint window together with the regular rational handoff, so the consumer sees the interval endpoint surface only through the displayed finite rows.

Now use Definition 342.9. The finite handoff repacks the convergent and recurrence rows into the packet obligation surface without adding a completed real, quotient tail, or total expansion coordinate.

Finally apply Definition 342.10. The real-facing boundary row is the boundary of the same finite cylinder handoff, and the cited handoff theorems provide exactly the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows named in the statement. □

Theorem 342.18 (ContinuedFraction convergent prefix window normal form)**.** *Every finite* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read of an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet restricted to a certified prefix factors through one displayed prefix normal form. The normal form consists exactly of the finite partial-quotient spine, numerator and denominator convergent rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *boundary row, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$ *row.*

The factorization is unchanged by restricting to a smaller certified prefix or by passing to a common overlapping prefix cylinder. It exports no completed real equality, quotient tail, total expansion theorem, interval arithmetic law, or host equality principle.

Proof. First apply Definition 342.14. It reads the certified prefix through the finite partial-quotient spine, numerator and denominator convergent rows, dyadic radius row, schedule, regular rational handoff, visible transports, continuation routes, package provenance, and local name-certificate row.

Next apply Definition 342.15. The rational-interval view is only the endpoint boundary of the same finite convergent window, so it does not add an interval arithmetic law or a host equality principle.

Use Definition 342.16 for prefix restriction. The smaller certified prefix inherits exactly the restricted spine, recurrence, radius, schedule, handoff, boundary, transport, continuation, package, and local certificate rows from the larger packet.

Finally apply Definition 342.17. Any consumer comparison over overlapping certified prefixes refines to the same common prefix cylinder, and that cylinder contains precisely the rows listed in the normal form. Completed real equality, quotient tail, total expansion, interval arithmetic, and host equality are absent from those finite rows. □

Theorem 342.19 (ContinuedFraction dyadic approximation extraction)**.** *Let P be an accepted* $\mathsf{ContinuedFraction}^{\uparrow}$ *packet whose displayed rows include a finite convergent window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, a* $\mathsf{StreamName}^{\uparrow}$ *schedule, a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and a* $\mathsf{Real}^{\uparrow}$ *boundary row. Then the same visible rows determine an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet.*

The precision row of the $\mathsf{DyadicApproximation}^{\uparrow}$ *packet is the displayed window precision of P, its endpoint row is the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *convergent/radius endpoint, its window is the scheduled regular-rational readback, and its ledger is the existing real-boundary handoff. The extraction introduces no quotient real, infinite-tail equality, selected endpoint, or completeness principle.*

Proof. Start with Definition 342.4. It reads the accepted continued-fraction prefix as the displayed regular-rational source window, with the schedule, recurrence ledger, transport rows, continuation routes, package provenance, and local name-certificate row still visible.

Next apply Definition 342.15. The same prefix window supplies the finite rational endpoint row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius row transports that endpoint into the dyadic endpoint coordinate required by the approximation packet.

Use Definition 342.17 to identify the finite consumer surface when the packet is restricted or compared inside a common prefix cylinder. The precision, endpoint, scheduled readback, ledger, transport, route, provenance, and local certificate rows remain exactly the displayed rows named in the statement.

Finally apply Definition 335.2 and Definition 335.3. These repack the transported precision, endpoint, window, and ledger rows as an accepted $\mathsf{DyadicApproximation}^{\uparrow}$ packet and seal its $\mathsf{Real}^{\uparrow}$ boundary through the existing handoff. Since every coordinate came from a displayed finite row, no quotient real, infinite-tail equality, selected endpoint, or completeness principle is added. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuedFraction}^{\uparrow}$ packet is generated by a finite partial-quotient spine, rational convergent rows, dyadic radius rows, scheduled observation windows, a regular-sequence handoff, a real-name seal boundary, and a finite provenance ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 342.4, Definition 342.6, Definition 342.12, and Definition 342.13 form the bridged public export surface by binding the finite carrier, classifier, recurrence-ledger obligation, regular-sequence handoff obligation, real-boundary obligation, standard finite-packet boundary, and displayed hsame, Cont, Pkg, and NameCert dependencies for continued-fraction packets over existing BEDC kernels.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuedFraction}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ContinuedFractionUp.ContinuedFractionPacket\_convergent\_window\_standard\_bridge`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close total expansion for every real, uniqueness of infinite expansions, measure-theoretic typicality, quotient equality of infinite tails, or machine-checked standard bridges.

Upgrade path: No further paper-side theory grade remains for the finite continued-fraction packet; formal verification and bridge checking remain separate.

Chapter 343

A Concrete Naming Certificate for $\mathsf{BishopReal}^{\uparrow}$

The $\mathsf{BishopReal}^{\uparrow}$ interface records the Bishop–Bridges 1985 constructive-analysis reading of real names inside the BEDC certificate layer. Its carrier is not a Dedekind cut, a quotient real, or a completed host number. It is a finite certificate surface over regular rational names, locatedness evidence, apartness evidence, explicit modulus rows, and the visible $\mathsf{Real}^{\uparrow}$ seal.

This chapter consumes finite observation schedules from chapter 126, dyadic endpoint and tolerance rows from chapter 291, the regular-Cauchy naming route from chapter 371, and the visible seal boundary of chapter 38. These are substantive carrier and ledger dependencies rather than adjacent topics: the packet coordinates $S, Q, D, M, L, A, E, H, C, P, N$ route every public constructive-real read through those surfaces.

Definition 343.1 (BishopReal regular located carrier)**.** *A* $\mathsf{BishopReal}^{\uparrow}$ *packet is a finite* BHist *row*

$$B = (S, Q, D, M, L, A, E, H, C, P, N)$$

with the following visible coordinates:

1. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite windows through which the rational approximation stream is inspected;*
2. *Q is a certified* $\mathsf{RegSeqRat}^{\uparrow}$ *name read on those scheduled windows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint packet used for finite tolerance and interval observations;*
4. *M is the explicit* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row controlling the regular sequence windows;*
5. *L is the displayed* $\mathsf{LocatedReal}^{\uparrow}$ *evidence row;*
6. *A is the displayed* $\mathsf{ApartnessReal}^{\uparrow}$ *evidence row;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal consumed by downstream real-facing interfaces;*
8. *H records componentwise* hsame *transport for $S, Q, D, M, L, A,$ and E;*
9. *C records the visible* Cont *routes through which a consumer reaches the finite stream windows, dyadic endpoint packets, modulus rows, locatedness rows, apartness rows, and seal row;*
10. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as the dependency anchors;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopReal}^\uparrow}$ *ledger row.*

The carrier predicate is $\mathsf{BishopRealPacket}(S, Q, D, M, L, A, E, H, C, P, N)$*. Its classifier compares two accepted packets only by passing through common finite stream windows, the same dyadic endpoint packet family, the displayed modulus rows, the locatedness and apartness evidence rows, and the visible* $\mathsf{Real}^\uparrow$ *seal. No coordinate stores a Dedekind cut, quotient real, trichotomy principle, selected limit, or classical completeness witness.*

Lean4 (definition) `BEDC.Derived.BishopRealUp.BishopRealPacket`

Theorem 343.2 (BishopReal located-apartness transport)**.** *For any two accepted* $\mathsf{BishopReal}^\uparrow$ *packets* $B = (S, Q, D, M, L, A, E, H, C, P, N)$ *and* $B' = (S', Q', D', M', L', A', E', H', C', P', N')$*, a componentwise* hsame *comparison of their stream schedules, regular rational names, dyadic endpoint packets, modulus rows, locatedness rows, apartness rows, and* $\mathsf{Real}^\uparrow$ *seals transports the local classifier from* B *to* B'*. Downstream reads of locatedness or apartness therefore factor through the same finite stream-window and dyadic-endpoint surface, never through trichotomy or host real equality.*

Proof. Project the two carrier rows through Definition 343.1. The schedule coordinates give the only stream windows a consumer may inspect, and the dyadic endpoint coordinates give the only finite tolerance packets available at those windows.

Transport the regular rational, modulus, locatedness, apartness, and seal rows componentwise along the displayed hsame witnesses. Each read is reached through one of the visible Cont routes in the carrier, so the transported packet has the same finite observation shape as the source packet.

The forbidden host data are absent from the carrier. Since no coordinate stores trichotomy, quotient equality, a cut, or a selected limit, the classifier cannot export any such principle after transport. □

Lean4 ✓ `BEDC.Derived.BishopRealUp.BishopRealPacket located apartness transport`

Theorem 343.3 (BishopReal RegSeqRat readback)**.** *Every* $\mathsf{Real}^\uparrow$ *seal read exported by an accepted* $\mathsf{BishopReal}^\uparrow$ *packet factors through the certified* $\mathsf{RegSeqRat}^\uparrow$ *name* Q*, the scheduled finite windows* S*, the* $\mathsf{DyadicRatCore}^\uparrow$ *endpoint packets* D*, and the explicit* $\mathsf{ModulusOfConvergence}^\uparrow$ *row* M*. The readback exposes only the regular rational name, its dyadic finite observations, and the displayed real seal.*

Lean4 ✓ `BEDC.Derived.BishopRealUp.BishopRealPacket regseqrat readback`

Proof. Let a downstream consumer read the seal coordinate E of a packet B. By the carrier definition, E is stored together with the schedule S, regular name Q, dyadic packet family D, and modulus row M.

The continuation coordinate C gives the finite routes by which E is reached. Each such route also names the stream window, dyadic endpoint packet, and modulus row used for the observation. Thus the seal read has the advertised regular-sequence handoff as part of its visible packet.

Nothing else remains to read. The packet has no coordinate for a quotient class, cut membership predicate, choice-selected endpoint, or classical completeness argument, so the readback is exhausted by S, Q, D, M, and E. □

Theorem 343.4 (BishopReal ledger obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{BishopReal}^\uparrow}$ *ledger surface consists exactly of the carrier, classifier, stability, readback, and dependency rows displayed by Definition 343.1, Definition 343.2, and Definition 343.3. Its dependency anchors are* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{LocatedReal}^\uparrow$*,* $\mathsf{ApartnessReal}^\uparrow$*,* $\mathsf{ModulusOfConvergence}^\uparrow$*, and* $\mathsf{Real}^\uparrow$*.*

Proof. The carrier row is the packet of Definition 343.1. It lists the stream schedule, regular rational name, dyadic endpoint packet, modulus, locatedness, apartness, seal, transport, continuation, provenance, and local ledger coordinates.

The classifier and stability rows are supplied by Definition 343.2. They compare packets only through common finite stream windows and componentwise transport of the visible rows.

The exactness and readback row is Definition 343.3. The provenance coordinate P and ledger coordinate N list precisely the seven dependency anchors named in the statement, so the ledger surface has no further exported row. □

Theorem 343.5 (BishopReal carrier construction obligation)**.** *The carrier obligation for* $\mathsf{BishopReal}^{\uparrow}$ *is to construct the finite packet of Definition 343.1 from a certified* $\mathsf{RegSeqRat}^{\uparrow}$ *name, scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint packets, a displayed* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row, displayed* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$ *evidence rows, the visible* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, finite* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ *ledger.*

Proof. Read the coordinates of Definition 343.1. They list exactly the regular rational name, finite observation windows, dyadic endpoint packets, convergence modulus, locatedness and apartness rows, real seal, transport, continuation, provenance, and local ledger.

The construction obligation is therefore finite: provide those displayed rows and repack them as the carrier predicate. No Dedekind cut, quotient real, trichotomy principle, selected limit, or classical completeness coordinate is available to the constructor. □

Theorem 343.6 (BishopReal classifier stability obligation)**.** *The classifier stability obligation for* $\mathsf{BishopReal}^{\uparrow}$ *is componentwise transport under* hsame *for the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows along the finite* Cont *routes displayed in the carrier. Accepted packets may classify together only through those finite routes and their shared BEDC dependency surface.*

Proof. Apply Definition 343.2. It transports the stream schedule, regular rational name, dyadic endpoint packet, modulus row, locatedness row, apartness row, and $\mathsf{Real}^{\uparrow}$ seal componentwise along the displayed hsame witnesses.

The same theorem records that every transported read is reached through one of the finite Cont routes in the carrier. Thus the classifier is stable exactly on the displayed packet surface and does not acquire host real equality, trichotomy, or a quotient comparison. □

Theorem 343.7 (BishopReal readback ledger exhaustion obligation)**.** *The readback-ledger exhaustion obligation for* $\mathsf{BishopReal}^{\uparrow}$ *is that every exported* $\mathsf{Real}^{\uparrow}$ *seal read factors through the certified* $\mathsf{RegSeqRat}^{\uparrow}$ *name, the scheduled finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint packets, the displayed* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row, and the dependency anchors* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The exhausted ledger excludes Dedekind cuts, quotient reals, trichotomy, selected limits, and classical completeness.*

Proof. First apply Definition 343.3. It says each $\mathsf{Real}^{\uparrow}$ seal read passes through the regular rational name, scheduled windows, dyadic endpoint packets, and explicit convergence modulus.

Then apply Definition 343.4. Its provenance and local ledger rows list exactly the seven dependency anchors in the statement, so the readback has no further exported row.

The non-constructive real principles are absent twice: they are not carrier coordinates, and they are not ledger anchors. The readback surface is therefore exhausted by the finite BEDC rows already displayed. □

Theorem 343.8 (BishopReal public NameCert export)**.** *The carrier construction, classifier stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, located-apartness transport, readback-ledger exhaustion, explicit modulus rows, and* $\mathsf{Real}^{\uparrow}$ *seal assemble the public* $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ *export over regular rational names. Every public read factors through exactly the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows.*

The public export contains no Dedekind cut, quotient real, trichotomy, selected limit, or classical completeness row.

Proof. First read the carrier from Definition 343.1. It lists the regular rational name, finite stream windows, dyadic endpoint packets, convergence modulus, locatedness and apartness rows, visible real seal, transport, continuation, provenance, and local naming ledger.

Next apply Definition 343.2 and Definition 343.3. These identify the componentwise transport rows and the $\mathsf{Real}^{\uparrow}$ seal readback as finite reads of the same constructive-real packet.

Then use Definition 343.4, Definition 343.5, Definition 343.6, and Definition 343.7. Together they exhaust the carrier, classifier, ledger, construction, and readback rows named in the statement.

Project the visible BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows from that surface. Repacking those rows gives the public $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ export.

Dedekind cuts, quotient reals, trichotomy, selected limits, and classical completeness remain outside the export because none is present in the carrier or ledger surface. □

Theorem 343.9 (BishopReal standard finite-packet bridge)**.** *Definition 343.8, Definition 343.3, and Definition 343.7 determine the standard Bishop-style finite packet bridge for* $\mathsf{BishopReal}^{\uparrow}$ *over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{ApartnessReal}^{\uparrow}$*,* $\mathsf{ModulusOfConvergence}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The bridge packet consists only of the regular rational name, locatedness row, apartness row, modulus row,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *ledger rows displayed by the BEDC carrier.*

Consequently the standard constructive-real packet exported by $\mathsf{BishopReal}^{\uparrow}$ *has no Dedekind cut, quotient real, trichotomy, selected limit, or classical completeness coordinate.*

Lean4 ✓ `BEDC.Derived.BishopRealUp.BishopRealPacket standard finite packet bridge`

Proof. First project the public export from Definition 343.8. It supplies the finite BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows of the accepted constructive-real packet.

Next identify the regular rational name and the visible $\mathsf{Real}^{\uparrow}$ seal through Definition 343.3. The readback passes only through the certified $\mathsf{RegSeqRat}^{\uparrow}$ name, scheduled finite windows, dyadic endpoint packets, and explicit modulus row.

Finally apply Definition 343.7. It says every standard field of the readback is one of the displayed finite BEDC rows over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The excluded real principles are not present in the public export, not present in the readback, and not present in the exhausted ledger. The resulting packet is therefore exactly the standard finite-packet bridge claimed in the statement. □

Theorem 343.10 (BishopReal regular-Cauchy located Real seal)**.** *Let* $B = (S, Q, D, M, L, A, E, H, C, P, N)$ *be an accepted* $\mathsf{BishopReal}^{\uparrow}$ *packet. Suppose its* $\mathsf{RegSeqRat}^{\uparrow}$ *name, locatedness row, apartness row, modulus row, and* $\mathsf{Real}^{\uparrow}$ *seal are transported along the displayed* hsame *witnesses and read only through the displayed* Cont *routes of Definition 343.1. Then* B *exports one finite constructive-real seal surface.*

The surface consists exactly of the regular rational name Q, the scheduled finite windows S, the dyadic endpoint packets D, the locatedness row L, the apartness row A, the modulus row M, the $\mathsf{Real}^{\uparrow}$ *seal E, the visible* hsame *transports, the visible* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ *ledger. Every public read of the accepted packet factors through those rows and through no Dedekind cut, quotient real, trichotomy principle, selected limit, or classical completeness witness.*

Lean4 ✓ `BEDC.Derived.BishopRealUp.BishopRealPacket regular cauchy located real seal`

Proof. First read the regular rational and seal coordinates. Apply Definition 343.3 to identify every $\mathsf{Real}^{\uparrow}$ seal read with the certified $\mathsf{RegSeqRat}^{\uparrow}$ name, scheduled finite windows, dyadic endpoint packets, explicit modulus row, and displayed seal.

Next transport the constructive-real evidence rows. Apply Definition 343.2 to carry the regular rational name, locatedness row, apartness row, modulus row, and seal row along the displayed hsame witnesses and through the same finite Cont routes.

Now exhaust the public ledger. Apply Definition 343.7 to show that every exported read is accounted for by the listed dependency anchors and finite BEDC rows.

Finally apply Definition 343.9. It repackages those rows as the standard Bishop-style finite packet over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since the bridge has no coordinate for cuts, quotient reals, trichotomy, selected limits, or classical completeness, the same exclusions hold for the seal surface. □

Theorem 343.11 (BishopReal finite constructive-real consumer completeness)**.** *Every finite consumer read of an accepted* $\mathsf{BishopReal}^{\uparrow}$ *packet factors through exactly the regular rational name, scheduled finite windows, dyadic endpoint packets, locatedness row, apartness row, modulus row, visible* $\mathsf{Real}^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ *rows of Definition 343.10. Conversely those rows repack as the finite constructive-real consumer surface of the accepted packet.*

Thus the mature consumer surface for $\mathsf{BishopReal}^{\uparrow}$ *is precisely the displayed regular-Cauchy located real packet. It contains no Dedekind cut, quotient real, trichotomy principle, selected limit, or classical completeness witness.*

Proof. Project the accepted carrier through Definition 343.1. The carrier lists the regular rational name, scheduled finite windows, dyadic endpoint packets, locatedness row, apartness row, modulus row, visible seal, transport routes, package row, and local certificate ledger as the available public coordinates.

Next transport the real evidence through Definition 343.2 and read back the regular name through Definition 343.3. These two results identify every constructive-real read with the displayed regular rational name, its finite windows, its dyadic endpoint packets, and its explicit modulus and locatedness-apartness rows.

Now exhaust the ledger. Apply Definition 343.7 and Definition 343.8. The first theorem says that no finite read leaves the displayed ledger, while the second exports that ledger as the public NameCert surface of the accepted packet.

Finally apply Definition 343.9 and Definition 343.10. They repack the same rows as the standard finite constructive-real packet and the regular-Cauchy located $\mathsf{Real}^{\uparrow}$ seal. Since the carrier and both repackings have no coordinate for cuts, quotient reals, trichotomy, selected limits, or classical completeness, those structures cannot be consumed from the finite packet. □

Theorem 343.12 (BishopReal located-apartness seal factorization)**.** *Every locatedness or apartness read exported by an accepted* $\mathsf{BishopReal}^{\uparrow}$ *packet factors through the same finite scheduled*

$\mathsf{RegSeqRat}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint packets,* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row, and* $\mathsf{Real}^{\uparrow}$ *seal row. Equivalently, the public constructive-real evidence read is one packet surface containing the regular rational name, scheduled finite windows, dyadic endpoints, explicit convergence modulus, locatedness row, apartness row, visible* hsame *transports, displayed* Cont *routes,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$ *evidence, and the visible* $\mathsf{Real}^{\uparrow}$ *seal.*

The factorization exports no trichotomy principle, quotient real, Dedekind cut, host real equality, selected representative, or classical completeness witness.

Proof. First transport the constructive-real evidence. Apply Definition 343.2; locatedness and apartness travel along the displayed hsame witnesses and remain on the same finite Cont routes as the regular rational name, modulus row, and seal row.

Next read back the regular name. By Definition 343.3, every seal-facing read of the regular rational coordinate is represented by scheduled finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint packets, the explicit modulus row, and the visible $\mathsf{Real}^{\uparrow}$ seal.

Now exhaust the certificate ledger. Apply Definition 343.7; no locatedness or apartness read can leave the displayed dependency anchors, transport rows, continuation routes, package provenance, and local name evidence.

Finally apply Definition 343.11. It repackages exactly the transported locatedness, apartness, modulus, regular name, endpoint, and seal rows as the finite constructive-real consumer surface. Since the surface has no coordinate for trichotomy, quotient reals, cuts, host real equality, selected representatives, or classical completeness, the same exclusions hold for every exported locatedness or apartness read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopReal}^{\uparrow}$ packet is generated from a certified $\mathsf{RegSeqRat}^{\uparrow}$ name, finite stream windows, dyadic endpoint packets, an explicit convergence modulus, locatedness evidence, apartness evidence, componentwise transport rows, continuation routes, package provenance, and the visible $\mathsf{Real}^{\uparrow}$ seal.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 343.1, Definition 343.2, Definition 343.3, Definition 343.4, Definition 343.5, Definition 343.6, Definition 343.7, Definition 343.8, Definition 343.9, Definition 343.10, Definition 343.11, and Definition 343.12 name the finite constructive real-neighbor certificate surface, standard bridge, regular-Cauchy located $\mathsf{Real}^{\uparrow}$ seal, finite constructive-real consumer surface, and located-apartness seal factorization over regular rational names, locatedness, apartness, explicit modulus rows, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopReal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BishopRealUp.BishopRealPacket\_standard\_finite\_packet\_bridge`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Dedekind-cut reals, quotient reals, trichotomy, selected limits, classical completeness, or formal verification of the finite consumer theorem.

Upgrade path: No further paper-axis theory grade remains; Lean verification remains on the formal axis.

Chapter 344

A Concrete Naming Certificate for $\mathsf{StoppingTime}^{\uparrow}$

Classically, a stopping time is a random time whose event $\{\tau \leq n\}$ is visible from the information available by time n, as in Williams, *Probability with Martingales* (1991), or Durrett, *Probability: Theory and Examples* (2019). The $\mathsf{StoppingTime}^{\uparrow}$ surface below keeps only the finite-prefix version of that object: a BEDC ledger over probability and process rows, with no host filtration, no optional sampling theorem, and no choice of an infinite stopping rule.

The finite prefix ledger is a carrier and classifier input from chapter 184 for probability rows, chapter 93 for visible information prefixes, and chapter 190 when the process row is read through a martingale dependency. These dependencies are the rows consumed by the packet surface below, not ambient background structure.

Definition 344.1 (StoppingTime source packet)**.** *A* $\mathsf{StoppingTime}^{\uparrow}$ *source packet is a finite* BHist *record*

$$S = (\Omega, X, F, n, w, h, c, p)$$

with the following visible rows:

1. Ω *is the consumed* $\mathsf{ProbSpace}^{\uparrow}$ *probability-source row;*
2. X *is a finite family of* $\mathsf{RandomVar}^{\uparrow}$ *process-observation rows, read either directly or through a consumed* $\mathsf{Martingale}^{\uparrow}$ *or* $\mathsf{MarkovChain}^{\uparrow}$ *dependency row;*
3. F *is a finite filtration-prefix ledger whose* k*th row lists the observable event names available up to the displayed unary time* $k \leq n$*;*
4. n *is the unary time row naming the finite prefix horizon;*
5. w *is the stopping-rule witness row: for each displayed* $k \leq n$*, the event row saying that the rule has stopped by* k *is read from the* F_k *ledger and the process rows* $X_0, \ldots, X_k$*;*
6. h *records componentwise* hsame *transport for the probability source, process rows, unary time row, filtration-prefix rows, and stopping witness;*
7. c *records the* Cont *read routes from prefix observations to the event row* $\{w \leq k\}$ *and from dependency certificates to the visible process rows;*
8. p *is the* Pkg *provenance row naming the consumed* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*,* $\mathsf{Martingale}^{\uparrow}$*, or* $\mathsf{MarkovChain}^{\uparrow}$ *dependency certificates.*

The carrier predicate StoppingTimeSource$(\Omega, X, F, n, w, h, c, p)$ *accepts exactly such finite packets. Its classifier transports the displayed rows componentwise and compares the stopping witness only through the same finite filtration prefix; it does not compare host stochastic processes or infinite sigma-algebra towers.*

Lean4 (definition) `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket`

Theorem 344.2 (StoppingTime NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$ *obligation surface consists of the source packet of Definition 344.1, carrier habitation by the rule that stops at the displayed horizon n, classifier transport by componentwise* hsame*, filtration-prefix stability under certified prefix restriction, ledger coverage for all stopping-event reads, and the standard finite stopping-time boundary. These rows consume only* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*,* $\mathsf{Martingale}^{\uparrow}$ *or* $\mathsf{MarkovChain}^{\uparrow}$ *dependency packets,* BHist *histories,* hsame *rows,* Cont *routes, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket namecert obligation surface`

Proof. For habitation, take the displayed rule that stops at the horizon row n. Every event $\{w \le k\}$ is then either absent before the horizon or the terminal event at the horizon, both read through the finite probability ledger of Ω and the displayed prefix rows.

Classifier transport is obtained by projecting the tuple $S = (\Omega, X, F, n, w, h, c, p)$. Each coordinate has a corresponding hsame comparison, and the Cont routes transport the read of each event $\{w \le k\}$ along the matched process and filtration-prefix rows.

The remaining obligations are exactness statements about the same finite ledger. Prefix restriction keeps only rows whose time coordinate lies in the certified prefix, and ledger coverage says that no stopping-event read has a source outside Ω, X, F, n, w, h, c, and p. □

Theorem 344.3 (StoppingTime filtration-prefix stability)**.** *Let $S = (\Omega, X, F, n, w, h, c, p)$ be an accepted* $\mathsf{StoppingTime}^{\uparrow}$ *source packet. Restrict the unary time horizon to a certified prefix $m \le n$ and retain only the rows $X_0, \ldots, X_m$, $F_0, \ldots, F_m$, the witness events $\{w \le k\}$ for $k \le m$, and the corresponding* hsame*,* Cont*, and* Pkg *rows. The resulting finite packet is again an accepted* $\mathsf{StoppingTime}^{\uparrow}$ *source packet over the prefix horizon m.*

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket prefix transport`
Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket filtration prefix stability`

Proof. Unfold Definition 344.1. Acceptance is stated row by row over the displayed unary prefix, so each retained time $k \le m$ was already one of the accepted time rows of the original packet.

Project the retained process rows, filtration-prefix rows, and stopping witness events. The read route for $\{w \le k\}$ still factors through F_k and $X_0, \ldots, X_k$, because those rows all lie inside the retained prefix.

The inherited hsame comparisons and Cont routes have the same endpoints after prefix restriction, and the Pkg provenance row still names the same dependency certificates. Repacking these rows gives the accepted prefix packet. □

Theorem 344.4 (StoppingTime ledger coverage)**.** *Every consumer-visible read from an accepted* $\mathsf{StoppingTime}^{\uparrow}$ *packet factors through one of the displayed rows of Definition 344.1: the* $\mathsf{ProbSpace}^{\uparrow}$ *source, the* $\mathsf{RandomVar}^{\uparrow}$ *process rows, the optional* $\mathsf{Martingale}^{\uparrow}$ *or* $\mathsf{MarkovChain}^{\uparrow}$ *dependency row, the unary time row, the finite filtration-prefix ledger, the stopping witness events, the* hsame *comparisons, the* Cont *read routes, and the* Pkg *provenance. No host filtration, infinite stopping time, optional sampling statement, or selected limit is exported.*

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket ledger coverage`

Proof. Inspect the carrier tuple. It has exactly the eight coordinates $(\Omega, X, F, n, w, h, c, p)$, and each coordinate is a finite BEDC row or a finite family indexed by the displayed unary prefix.

A probability read factors through Ω. A process read factors through X and its dependency certificate. A stopping-event read factors through the finite filtration-prefix row F_k, the process prefix ending at k, and the continuation route recorded in c.

There is no carrier coordinate for a host filtration tower, an infinite random time, a theorem about stopped processes, or a limiting construction. Thus the listed finite rows cover all readable packet content. □

Theorem 344.5 (StoppingTime standard boundary)**.** *The standard mathematical boundary represented by* $\mathsf{StoppingTime}^{\uparrow}$ *is the finite-prefix stopping rule: for every displayed time $k \leq n$, the event that the rule has stopped by k is read from the finite information ledger F_k and the process observations through time k. The BEDC packet agrees with the classical stopping-time intuition from finite probability-process prefixes, but it does not assert a host filtration, completed sigma-algebra sequence, optional stopping theorem, stopped martingale construction, or standard stochastic-process bridge.*

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket standard boundary`

Proof. Read the stopping witness row w through the source packet. For each displayed $k \leq n$, the witness event $\{w \leq k\}$ is attached by a Cont route to the finite ledger F_k and the process prefix $X_0, \ldots, X_k$.

The $\mathsf{ProbSpace}^{\uparrow}$ row supplies the probability-source boundary, while the $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, or $\mathsf{MarkovChain}^{\uparrow}$ dependency row supplies the process rows consumed by the packet. The provenance row p records that these are the only dependency certificates read by the stopping rule.

Therefore the standard interpretation is exactly the finite-prefix adaptedness condition for stopping rules. Any theorem requiring an infinite filtration, stopped process, optional sampling, or a host stochastic-process object lies outside this seed certificate. □

Theorem 344.6 (StoppingTime scoped dependency envelope)**.** *The source packet of Definition 344.1, the* $\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$ *obligation surface, filtration-prefix stability, ledger coverage, and standard boundary bind every accepted stopping-rule row to finite* BHist *prefixes, displayed* hsame *transport,* Cont *read routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$ *row. No host filtration, infinite stopping time, optional stopping theorem, or selected stochastic-process limit is introduced.*

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket scoped dependency envelope`

Proof. Project the source tuple $S = (\Omega, X, F, n, w, h, c, p)$ from Definition 344.1. Its probability source, process rows, unary horizon, filtration-prefix ledger, stopping witness, transport, read routes, and provenance are all finite rows of one displayed packet.

Apply Definition 344.2 and Definition 344.3 to keep habitation, classifier transport, and prefix restriction inside the same finite BHist ledger. The accepted stopping-event row $\{w \leq k\}$ is therefore read only through the displayed prefix F_k, the process prefix up to k, and the corresponding hsame and Cont rows.

Apply Definition 344.4 and Definition 344.5 to close the consumer boundary. Those rows already name the Pkg provenance and local $\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$ registration, and they explicitly leave host filtrations, infinite stopping times, optional stopping, and stochastic-process limits outside the packet. □

Theorem 344.7 (StoppingTime public consumer export)**.** *The scoped packet of Definition 344.6 exports a public finite-prefix stopping-rule certificate to* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, *and* $\mathsf{MarkovChain}^{\uparrow}$ *consumers. The exported certificate consists exactly of the probability source row, process rows, unary horizon, finite filtration-prefix ledger, stopping-event witness rows,* hsame *transport,* Cont *read routes, and* Pkg *provenance.*

It exports no host filtration, infinite stopping time, optional stopping theorem, stopped martingale construction, or selected stochastic-process limit.

Lean4 ✓ `BEDC.Derived.StoppingTimeUp.StoppingTimeSourcePacket public consumer export`

Proof. Begin with Definition 344.2. It binds the probability source, process rows, unary horizon, stopping-event witness rows, transport, read routes, and package provenance to the local accepted packet.

Apply Definition 344.3 to keep the filtration read finite at every displayed horizon. Then use Definition 344.4 to identify the stopping-event witness rows as ledger entries over the same prefix.

Next apply Definition 344.5; it is the public probability-process boundary through which $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, and $\mathsf{MarkovChain}^{\uparrow}$ consumers read the packet.

Finally compose these rows with Definition 344.6. The envelope contains no host filtration, infinite stopping time, optional stopping theorem, stopped martingale construction, or selected limit, so the public certificate exports none of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$.

Constructive story: A stopping-time packet is generated from a displayed $\mathsf{ProbSpace}^{\uparrow}$ source, finite $\mathsf{RandomVar}^{\uparrow}$ process rows optionally supplied through $\mathsf{Martingale}^{\uparrow}$ or $\mathsf{MarkovChain}^{\uparrow}$, a unary time horizon, a finite filtration-prefix ledger, stopping-event witness rows, componentwise hsame transport, finite Cont read routes, and Pkg provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 344.1, Definition 344.2, Definition 344.3, Definition 344.4, Definition 344.5, Definition 344.6, and Definition 344.7 bind the obligation surface, filtration-prefix stability, ledger coverage, finite standard-boundary bridge, exported finite-prefix stopping-rule certificate, public finite-prefix stopping-rule examples, cross-consumer completeness for $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, and $\mathsf{MarkovChain}^{\uparrow}$ finite-prefix consumers, BHist prefixes, hsame transport, Cont routes, Pkg provenance, and local NameCert row as the full mature public finite-prefix stopping-rule scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StoppingTime}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host filtrations, infinite stopping times, optional stopping, stopped martingales, selected limits, or a standard stochastic-process bridge.

Upgrade path: The paper-axis treatment is mature; further progress is formal verification and bridge checking.

Chapter 345

A Concrete Naming Certificate for $\mathsf{FreeMonoid}^{\uparrow}$

The $\mathsf{FreeMonoid}^{\uparrow}$ interface names finite BEDC word histories over a displayed payload row. In the standard algebraic account, as in Mac Lane's treatment of free algebraic constructions and the usual formal-language presentation, a free monoid is generated by finite words with empty word and concatenation. The BEDC certificate below records that surface without importing a host free-monoid primitive.

Definition 345.1 (FreeMonoid word carrier)**.** *Fix a payload history source row* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and a payload classifier* $\sim_A$*. A* $\mathsf{FreeMonoid}^{\uparrow}$ *word is a finite* BHist *spine*

$$W = (s, \ell, p, h, c, q)$$

whose positions are licensed by $\mathsf{List}^{\uparrow}$*, whose length row* ℓ *is licensed by* $\mathsf{Nat}^{\uparrow}$*, whose payload entries* p *are carried by* A*, whose endpoint history* h *is represented by the displayed list spine, whose route row* c *records the* Cont *reads for empty word, singleton insertion, and concatenation, and whose provenance row* q *records the* Pkg *dependencies on* A*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*, and* $\mathsf{Monoid}^{\uparrow}$*.*

Two accepted words classify together exactly when their length rows agree and their corresponding payload positions compare by hsame *over the payload rows, transported along the displayed* Cont *routes. The classifier does not identify words by an external quotient or by a host list type.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidTasteGate single carrier alignment`

Theorem 345.2 (FreeMonoid empty word unit)**.** *The empty* BHist *spine is an accepted* $\mathsf{FreeMonoid}^{\uparrow}$ *word, its length row is the* $\mathsf{Nat}^{\uparrow}$ *zero row, and concatenating it on either side of an accepted word returns a word classified with the original endpoint. The unit rows are read only through the empty-word constructor, the* $\mathsf{List}^{\uparrow}$ *empty-spine license, the* $\mathsf{Nat}^{\uparrow}$ *zero-length row, the* $\mathsf{Monoid}^{\uparrow}$ *append unit surface, and the same* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier empty word unit`

Proof. Use the empty constructor of the finite list spine for the position row and the zero row of $\mathsf{Nat}^{\uparrow}$ for the length row. This gives a carried word with no payload entry and no hidden endpoint.

For the two unit laws, read concatenation through the $\mathsf{Monoid}^{\uparrow}$ append surface. The left and right empty-history continuation laws identify the resulting endpoint with the original word endpoint under hsame, while the length ledger reads zero plus the displayed length and the displayed length plus zero as the same $\mathsf{Nat}^{\uparrow}$ row. □

Theorem 345.3 (FreeMonoid concatenation associativity)**.** *For accepted* $\mathsf{FreeMonoid}^{\uparrow}$ *words* u*,* v*, and* w*, both concatenation parenthesizations*

$$(u \cdot v) \cdot w \quad \textit{and} \quad u \cdot (v \cdot w)$$

are accepted and classify together. The witness is the finite BHist *append-associativity route together with the* $\mathsf{List}^{\uparrow}$ *append-spine associativity row, the* $\mathsf{Nat}^{\uparrow}$ *length-addition associativity row, and componentwise* hsame *transport of the payload positions.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier concat associativity`

Proof. First concatenate the displayed list spines. The $\mathsf{List}^{\uparrow}$ append readback keeps the positions finite and records the same ordered payload rows in both parenthesizations.

Next apply the $\mathsf{Monoid}^{\uparrow}$ append associativity surface to the endpoint histories. The resulting hsame route transports the endpoint row without changing any payload entry.

Finally compare the length rows. Both paths read the same finite addition ledger for the three displayed lengths, so the $\mathsf{Nat}^{\uparrow}$ row agrees with the positionwise payload transport. These rows repack as the classifier witness for the two accepted concatenation endpoints. □

Theorem 345.4 (FreeMonoid NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$ *obligation surface consists of exactly five finite rows:*

1. *carrier habitation by the empty word of Definition 345.2;*
2. *classifier reflexivity, symmetry, and transitivity from the* $\mathsf{List}^{\uparrow}$ *position classifier, the* $\mathsf{Nat}^{\uparrow}$ *length row, and the payload* hsame *rows;*
3. *carrier transport under the classifier by componentwise payload transport and length agreement;*
4. *concatenation closure by the* $\mathsf{Monoid}^{\uparrow}$ *append surface and the* $\mathsf{List}^{\uparrow}$ *represented-spine append route;*
5. *ledger exactness for empty-word, singleton, and concatenation reads.*

No host free-monoid carrier, external quotient, infinite word, or language recognizer is part of the obligation.

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier namecert obligation surface`

Proof. The habitation row is the empty finite spine with the zero length row. This is the carried endpoint fixed by Definition 345.2.

The classifier laws are inherited componentwise. $\mathsf{List}^{\uparrow}$ supplies the finite position comparison, $\mathsf{Nat}^{\uparrow}$ supplies length agreement, and the payload classifier supplies the hsame rows on each displayed position.

For transport and closure, move each payload position along the classifier route and then use the displayed append readback. Concatenation remains a finite spine, and the endpoint route is the $\mathsf{Monoid}^{\uparrow}$ append continuation.

The ledger has only three read forms: empty word, singleton insertion, and concatenation. Each read is stored with its source provenance and length row, so every accepted consumer read projects back to one of those displayed rows. □

Theorem 345.5 (FreeMonoid ledger normal form)**.** *Every accepted* $\mathsf{FreeMonoid}^{\uparrow}$ *consumer read has a unique displayed ledger shape: empty word, singleton insertion of one payload row, or concatenation of two previously accepted word rows. The normal form records the source provenance,*

the $\mathsf{List}^{\uparrow}$ *position spine, the* $\mathsf{Nat}^{\uparrow}$ *length row, the* $\mathsf{Monoid}^{\uparrow}$ *append route when present, and the componentwise payload* hsame *rows. It exports no separate evaluator for words.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier ledger normal form`

Proof. Inspect the carrier row of Definition 345.1. The only constructors recorded in the route row are empty word, singleton insertion, and concatenation.

In the empty case, the position spine is empty and the length row is the $\mathsf{Nat}^{\uparrow}$ zero row. In the singleton case, the spine has one displayed payload entry and one source provenance read. In the concatenation case, the two input word rows and the append route determine the displayed output spine and the length-addition row.

These cases exhaust the route row, so a consumer cannot read a hidden word evaluation, host list operation, or language-level recognizer from the packet. □

Theorem 345.6 (FreeMonoid public seal)**.** *The public* $\mathsf{FreeMonoid}^{\uparrow}$ *seal is the finite packet consisting of the word carrier of Definition 345.1, the empty-unit and associativity rows of Definition 345.2 and Definition 345.3, the* $\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$ *obligation surface of Definition 345.4, and the ledger normal form of Definition 345.5. This packet is the consumer-facing certificate for finite word histories over the displayed payload source.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier public seal`

Proof. Package the carrier, classifier, unit, concatenation, and ledger rows named in the preceding blocks. Each row is a finite BHist, hsame, Cont, or Pkg dependency over the payload source, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$.

The public seal exposes exactly those rows. Since the carrier has no coordinate for a host free-monoid object, external quotient, infinite stream, automaton, or language semantics, downstream chapters can consume only the displayed finite word-history certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$.

Constructive story: A free-monoid word is generated as a finite BHist spine whose positions are licensed by $\mathsf{List}^{\uparrow}$, whose length row is licensed by $\mathsf{Nat}^{\uparrow}$, whose concatenation route is the $\mathsf{Monoid}^{\uparrow}$ append surface, and whose payload positions compare componentwise by hsame under the displayed Pkg provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 345.6 binds the finite word-history packet with empty word, singleton insertion, concatenation, length addition, componentwise payload transport, source provenance, empty concatenation inversion from Definition 345.7, singleton concatenation inversion from Definition 345.8, and the concrete $\mathsf{List}^{\uparrow}$ continuation monoid instance of Definition 345.9.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host free-monoid quotients, infinite words, formal-language recognizers, automata semantics, or a standard bridge.

Upgrade path: Remaining work is formal verification of the displayed finite word-history consumer package and standard-bridge checking.

Theorem 345.7 (FreeMonoid empty concatenation inversion)**.** *Fix one payload source row* $A : \mathsf{BHist} \to \mathsf{Prop}$ *and one payload classifier* $\sim_A$*. Let* u *and* v *be accepted* $\mathsf{FreeMonoid}^{\uparrow}$ *words for this same payload,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*, and* $\mathsf{Monoid}^{\uparrow}$ *setup, and let* $u \cdot v$ *be their displayed concatenation word. If* $u \cdot v$ *is classified with the empty* $\mathsf{FreeMonoid}^{\uparrow}$ *word, then* u *is classified with the empty word and* v *is classified with the empty word, all under the same payload and provenance rows.*

Proof. Unpack the carrier of Definition 345.1. The accepted word u has a finite position spine s_u, a $\mathsf{Nat}^{\uparrow}$ length row ℓ_u, payload entries over A, and provenance through the fixed A, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$ packet. The word v has the analogous rows s_v and ℓ_v. The concatenation ledger records the output position spine as the displayed $\mathsf{List}^{\uparrow}$ append of s_u and s_v, and its length row as the displayed $\mathsf{Nat}^{\uparrow}$ addition row $\ell_u + \ell_v$.

The hypothesis says that this concatenation output is classified with the empty word. By the classifier clause in Definition 345.1, its length row is therefore classified with the $\mathsf{Nat}^{\uparrow}$ zero row, and its position spine is classified with the empty $\mathsf{List}^{\uparrow}$ spine. The ledger normal form of Definition 345.5 leaves no hidden word evaluator: in the concatenation case, these two rows are exactly the append spine and the length-addition row just displayed.

Apply empty-result inversion for the finite append surface recorded in Definition 12.2, equivalently the contrapositive form of Definition 4.11, to the displayed append spine $s_u \cdot s_v$ classified with the empty spine. Both factors are classified with the empty spine. The same inversion applied to the $\mathsf{Nat}^{\uparrow}$ length-addition row identifies both ℓ_u and ℓ_v with the zero length row.

There are now no remaining payload positions in either factor: an empty $\mathsf{List}^{\uparrow}$ spine has no displayed position at which a payload comparison can be stored, and the zero $\mathsf{Nat}^{\uparrow}$ row agrees with that empty presentation. The Pkg provenance has not changed, since the concatenation row in Definition 345.1 uses the same payload source, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$ packet for the two inputs and the result. Thus the classifier clauses for u against the empty word and for v against the empty word are exactly the empty spine comparison, the zero length comparison, the vacuous componentwise payload comparison, and the same continuation provenance. Therefore both factors are classified with the empty $\mathsf{FreeMonoid}^{\uparrow}$ word under the original setup. □

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier empty concat inversion`

Theorem 345.8 (FreeMonoid singleton concatenation inversion)**.** *Fix one payload source row $A : \mathsf{BHist} \to \mathsf{Prop}$ and one payload classifier $\sim_A$. Let u and v be accepted $\mathsf{FreeMonoid}^{\uparrow}$ words for this same payload, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$ setup, and let $u \cdot v$ be their displayed concatenation word. If $u \cdot v$ is classified with the singleton $\mathsf{FreeMonoid}^{\uparrow}$ word carrying payload row a, then exactly one factor is classified with the empty word and the other factor is classified with the singleton word carrying a, all under the same $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, hsame, Cont, and Pkg provenance rows.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier singleton concat inversion`

Proof. Unpack the carrier of Definition 345.1. The two accepted factors have finite position spines s_u and s_v, length rows ℓ_u and ℓ_v, componentwise payload rows over A, and the same displayed provenance packet. Their concatenation ledger records the output spine as the $\mathsf{List}^{\uparrow}$ append of s_u and s_v and the output length as the $\mathsf{Nat}^{\uparrow}$ addition row $\ell_u + \ell_v$.

The singleton-output hypothesis identifies that same ledger with the displayed one-position spine and the $\mathsf{Nat}^{\uparrow}$ successor-of-zero length row. By the ledger normal form of Definition 345.5, the concatenation case has no further evaluator: the only possible source of that singleton row is the append spine and its paired length row.

Apply finite append singleton inversion to the displayed $\mathsf{List}^{\uparrow}$ spine $s_u \cdot s_v$. There are two cases. In the left-empty case, s_u is the empty spine and s_v is the singleton spine carrying the unique displayed payload position. In the right-empty case, s_u is that singleton spine and s_v is the empty spine. The corresponding $\mathsf{Nat}^{\uparrow}$ length row is inverted in the same case split: the empty factor has zero length, and the singleton factor has the one-position length row.

It remains to transport the payload row. In each case the componentwise classifier in Definition 345.1 has exactly one displayed position on the singleton side and no displayed position on

the empty side. Thus the unique payload comparison for a belongs to the singleton factor, while the empty factor has the vacuous payload comparison.

The Pkg and Cont rows are unchanged, because the concatenation clause of Definition 345.1 uses the same payload source, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Monoid}^{\uparrow}$ packet for both inputs and the result. Repacking the classifier clauses gives the two stated alternatives, and the finite spine inversion makes them disjoint: the unique singleton position cannot lie in both factors. □

Theorem 345.9 (FreeMonoid ListUp Cont concrete instance)**.** *The* $\mathsf{FreeMonoid}^{\uparrow}$ *word carrier of Definition 345.1, with the empty word as unit and certified concatenation read through the* $\mathsf{List}^{\uparrow}$ *spine and displayed* Cont *routes, forms a concrete* BHist*-grounded* $\mathsf{Monoid}^{\uparrow}$ *instance. The left and right unit laws are the empty-word unit surface of Definition 345.2; associativity is the certified concatenation associativity of Definition 345.3; and carrier and classifier stability are supplied by the existing* $\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$ *obligation surface over* hsame*,* Cont*,* Pkg*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*, and* $\mathsf{Monoid}^{\uparrow}$ *rows.*

This theorem is the sibling lattice link to chapter 40: the finite word carrier supplies a concrete $\mathsf{Monoid}^{\uparrow}$ *instance by using the* $\mathsf{List}^{\uparrow}$ *spine, the empty word, certified concatenation, and displayed* Cont *routes rather than an abstract monoid carrier detached from BEDC histories.*

Lean4 ✓ `BEDC.Derived.FreeMonoidUp.FreeMonoidWordCarrier list cont concrete instance`

Proof. Read a word carrier as its finite $\mathsf{List}^{\uparrow}$ spine, length row, componentwise payload rows, continuation routes, and package provenance. The empty word is the displayed empty spine with zero length, so Definition 345.2 gives both unit laws through the same BHist carrier rows.

Concatenation is the certified append ledger on the $\mathsf{List}^{\uparrow}$ spine together with the corresponding length and payload ledgers. Applying Definition 345.3 identifies the two triple-concatenation readbacks by the displayed hsame comparisons and the same Cont routes.

Finally use the $\mathsf{NameCert}_{\mathsf{FreeMonoid}^{\uparrow}}$ obligation surface. It is the carrier and classifier stability statement for these concrete rows, so the resulting $\mathsf{Monoid}^{\uparrow}$ instance is grounded in the listed BHist data rather than in abstract carrier or classifier equivalence parameters. □

Chapter 346

A Concrete Naming Certificate for $\mathsf{BitVector}^{\uparrow}$

The $\mathsf{BitVector}^{\uparrow}$ interface names fixed-length bit strings as finite BEDC data rather than as host-level binary strings. Knuth, *The Art of Computer Programming*, Vol. 4A, treats fixed-length bit strings as finite combinatorial objects; the BEDC packet below keeps that reading at the BHist level by using $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ rows with explicit length and provenance ledgers.

The fixed-length source consumes the sibling chapters chapter 32, chapter 36, and chapter 28. $\mathsf{Bool}^{\uparrow}$ supplies the visible bit rows, $\mathsf{List}^{\uparrow}$ supplies the spine and component classifier route, and $\mathsf{Nat}^{\uparrow}$ supplies the unary width row that fixes the public length ledger.

Definition 346.1 (BitVector BHist source)**.** *A* $\mathsf{BitVector}^{\uparrow}$ *source packet is a finite* BHist *record*

$$B = (n, s, l, p)$$

with the following visible coordinates:

1. *n is a unary* $\mathsf{Nat}^{\uparrow}$ *length row;*
2. *s is a* $\mathsf{List}^{\uparrow}$ *spine whose entries are* $\mathsf{Bool}^{\uparrow}$ *histories and whose displayed list length is transported from n;*
3. *l is the length ledger recording that every public bit read of s lies at an index below n and that no row beyond the displayed length is read;*
4. *p is the* Pkg *provenance naming the consumed* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, hsame, *and* Cont *certificate rows.*

The carrier predicate is $\mathsf{BitVectorSource}(n, s, l, p)$*. The classifier on accepted packets compares the unary length rows by* hsame*, compares the list spines by the* $\mathsf{List}^{\uparrow}$ *classifier, compares each visible bit by the* $\mathsf{Bool}^{\uparrow}$ *classifier, and requires the same* Cont *routes and* Pkg *provenance. No host bit-string equality, machine word, numeric binary encoding, or hidden padding convention is exported.*

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorTasteGate single carrier alignment`

Theorem 346.2 (BitVector carrier stability)**.** *Let $B = (n, s, l, p)$ be an accepted* $\mathsf{BitVector}^{\uparrow}$ *source packet. If the unary length row, the list-of-bool spine, the length ledger, and the provenance row are transported by displayed* hsame *along the same* Cont *routes named by p, then the transported packet $B' = (n', s', l', p')$ is again an accepted* $\mathsf{BitVector}^{\uparrow}$ *source packet.*

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorBHistSource carrier stability`
Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSourcePacket carrier stability`

Proof. Project the four coordinates of Definition 346.1. The $\mathsf{Nat}^{\uparrow}$ length row transports first, so the displayed target length is fixed before any bit read is considered.

Transport the $\mathsf{List}^{\uparrow}$ spine along that length row. Each visible component of the spine is a $\mathsf{Bool}^{\uparrow}$ history, so the component reads transport by the componentwise $\mathsf{Bool}^{\uparrow}$ classifier and the displayed hsame rows.

Finally transport the length ledger and Pkg provenance through the same Cont routes. The ledger still permits exactly the positions below the transported length and hides every position beyond it, so the transported packet is accepted. □

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorPacket carrier transport`
Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorCarrier hsame stability`
Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSource carrier stability`

Theorem 346.3 (BitVector classifier laws)**.** *The* $\mathsf{BitVector}^{\uparrow}$ *classifier is reflexive, symmetric, and transitive on accepted packets by componentwise use of the* $\mathsf{Nat}^{\uparrow}$ *length classifier, the* $\mathsf{List}^{\uparrow}$ *spine classifier, and the* $\mathsf{Bool}^{\uparrow}$ *entry classifier.*

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSourceClassifier laws`

Proof. For reflexivity, take the reflexive hsame row on the displayed unary length and the reflexive $\mathsf{List}^{\uparrow}$ comparison of the list spine. Each entry read is a $\mathsf{Bool}^{\uparrow}$ row and hence compares with itself by the boolean classifier.

For symmetry, reverse the displayed length comparison, then reverse the $\mathsf{List}^{\uparrow}$ spine comparison and every visible $\mathsf{Bool}^{\uparrow}$ entry comparison. The same ledger and provenance rows are read in the opposite order.

For transitivity, compose the two displayed length comparisons, compose the two list-spine comparisons, and compose the entrywise boolean comparisons at each transported index. The length ledger keeps the index set fixed under the transport, so no hidden position enters the composite classifier. □

Theorem 346.4 (BitVector ledger coverage)**.** *Every public read of an accepted* $\mathsf{BitVector}^{\uparrow}$ *packet factors through the finite ledger* (n, s, l, p)*. The packet exposes exactly the unary length row, the list-of-bool spine, the length exactness ledger, the componentwise* $\mathsf{Bool}^{\uparrow}$ *classifier rows, the* Cont *routes, and the* Pkg *provenance.*

Consequently the packet exports no host binary-string primitive, no numeric interpretation of the bits, no padding policy, and no machine-word overflow rule.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorFiniteLedger ledger coverage`

Proof. Start from the carrier definition. Its tuple lists exactly four coordinates, and each coordinate is a finite row over the existing BEDC kernel and finite data certificates.

A length-facing consumer reads n. A bit-facing consumer reads s only through the positions admitted by l. A transport-facing consumer reads the componentwise $\mathsf{Bool}^{\uparrow}$ and $\mathsf{List}^{\uparrow}$ classifier rows together with the displayed Cont routes and provenance named by p.

No other coordinate is present in the packet. Therefore every public consumer projection is one of the displayed rows or a projection through those route and provenance rows. □

Theorem 346.5 (BitVector NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{BitVector}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the empty vector is carried by the zero* $\mathsf{Nat}^{\uparrow}$ *length row, the empty* $\mathsf{List}^{\uparrow}$ *boolean spine, the empty length ledger, and the corresponding* Pkg *provenance;*
2. *classifier laws: reflexivity, symmetry, and transitivity are inherited componentwise from the length, list, and boolean classifiers of Definition 346.3;*
3. *length-ledger exactness: the public bit positions are exactly the positions below the displayed unary length row;*
4. *transport under* hsame*: the carrier stability of Definition 346.2 repacks every transported length, spine, ledger, and provenance row as another accepted packet;*
5. *downstream finite-data consumption: list, product, matrix-index, and error-code chapters may read only the displayed finite bit spine, length row, componentwise classifier rows, continuation routes, and provenance ledger.*

These rows are the complete seed obligation for the fixed-length bit-vector name.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSourcePacket namecert obligation surface`

Proof. For carrier habitation, choose the zero unary length row and the empty $\mathsf{List}^{\uparrow}$ spine of $\mathsf{Bool}^{\uparrow}$ entries. The length ledger has no public bit position to expose, and the provenance row records only the consumed finite data certificates.

For the classifier laws, apply Definition 346.3. The classifier is built from the displayed length comparison, list-spine comparison, and entrywise boolean comparisons, so the three equivalence laws are inherited componentwise.

For length-ledger exactness, unfold the carrier. A public bit read must be a read of the list spine at an index admitted by l, and l admits exactly the indices below n.

For transport, apply Definition 346.2. The same hsame and Cont routes transport the length row, the boolean spine, the ledger, and the provenance without adding a hidden binary-string comparison.

For downstream use, every listed consumer receives only the rows displayed in Definition 346.4. Thus a list, product, matrix-index, or error-code construction can consume fixed-length bit data without treating bit strings as host primitives. □

Theorem 346.6 (BitVector finite-data anchor)**.** *The* $\mathsf{BitVector}^{\uparrow}$ *source packet is a finite-data anchor between* $\mathsf{List}^{\uparrow}$ *and later finite algebraic encodings. It can be consumed as a length-indexed boolean list with explicit* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* hsame*,* Cont*, and* Pkg *rows, and it supplies no additional arithmetic, hashing, coding, or matrix operation.*

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSourcePacket finite data anchor`

Proof. Read the packet through Definition 346.1. The only payload row is a $\mathsf{List}^{\uparrow}$ spine of $\mathsf{Bool}^{\uparrow}$ entries, and the only shape row is the unary $\mathsf{Nat}^{\uparrow}$ length with its ledger.

Apply Definition 346.4. It identifies every public consumer route with the finite length, boolean spine, classifier rows, continuations, and provenance.

The exclusions follow by projection: the packet has no coordinate for a numeric interpretation, hash evaluation, error-correction metric, matrix indexing operation, or host machine word. Those later chapters may consume the finite bit-vector source, but they must add their own operation rows. □

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorFiniteLedger finite data anchor`

Theorem 346.7 (BitVector public finite-algebra export)**.** *Every downstream finite algebraic encoding consuming* $\mathsf{BitVector}^{\uparrow}$ *receives only the unary* $\mathsf{Nat}^{\uparrow}$ *length row, the* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Bool}^{\uparrow}$ *entries, the length ledger, componentwise classifier rows,* hsame *transports,* Cont *routes, and* Pkg *provenance from Definition 346.1. It receives no host bit-string equality, numeric binary primitive, padding policy, hash evaluator, or machine-word overflow rule.*

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorFiniteLedger public finite algebra export`

Proof. The source of every public read is Definition 346.1. Carrier stability is supplied by Definition 346.2, and the componentwise classifier laws are supplied by Definition 346.3.

Apply Definition 346.4 to identify the visible length ledger and its admitted list positions. Apply Definition 346.5 to package the carrier, classifier, stability, ledger, transport, and provenance rows as the public certificate surface.

Finally use Definition 346.6. It states that finite algebraic consumers may read the fixed-length boolean list surface but must add their own arithmetic, coding, hash, matrix, or machine-word operation rows.

Thus the public export is exactly the displayed finite-data packet. Since the packet has no coordinate for host bit-string equality, binary arithmetic, padding, hashing, or overflow, none of those structures can be exported. □

Theorem 346.8 (BitVector scoped dependency binding)**.** *The* $\mathsf{BitVector}^{\uparrow}$ *source packet, carrier stability, classifier laws, ledger coverage,* NameCert *obligation surface, and finite-data anchor are scoped exactly over* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

The scoped packet exposes no host bit-string equality, numeric binary interpretation, padding convention, machine-word overflow, hashing semantics, or error-correction metric.

Proof. The source rows are the rows of Definition 346.1. Carrier stability, classifier laws, and ledger coverage are fixed by Definition 346.2, Definition 346.3, and Definition 346.4.

The accepted certificate surface is Definition 346.5, and the downstream finite-data role is Definition 346.6. Together these results list all consumer projections: length, boolean spine, classifier rows, transport rows, finite ledger rows, package provenance, and certificate registration.

Every projection is one of the displayed BEDC rows over $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. There is no coordinate for host bit-string equality, binary numeric interpretation, padding, overflow, hashing, or an error-correction metric, so none can be exported by the scoped package. □

Theorem 346.9 (BitVector Boolean-list width exhaustion)**.** *An accepted* $\mathsf{BitVector}^{\uparrow}$ *packet is completely exhausted by its unary* $\mathsf{Nat}^{\uparrow}$ *width row, its* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Bool}^{\uparrow}$ *histories, component classifier rows, length ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence. Conversely, those displayed finite-data rows supply the abstract fixed-width* $\mathsf{BitVector}^{\uparrow}$ *interface.*

The interface has no extra coordinate for host bit-string equality, numeric binary encoding, padding convention, hashing semantics, matrix indexing, overflow, or error-correction metrics.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorFiniteLedger semantic name certificate`

Proof. Start from the source carrier of Definition 346.1. It exposes only a unary width row and a list spine of boolean component histories.

Apply Definition 346.3 to read the component classifier rows and Definition 346.4 to identify the length ledger. These two results account for the fixed-width and component readback coordinates.

Use Definition 346.5 for the displayed hsame, Cont, Pkg, and local NameCert rows, and apply Definition 346.6 to expose the finite-data consumer surface.

Finally apply Definition 346.8. The scoped packet contains exactly the listed $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows, so the abstract fixed-width interface is supplied by those rows and no external bit-string or arithmetic coordinate remains. □

Theorem 346.10 (BitVector standard bridge boundary)**.** *The public* $\mathsf{BitVector}^{\uparrow}$ *packet interprets as a standard finite boolean vector exactly through the unary* $\mathsf{Nat}^{\uparrow}$ *length row, the* $\mathsf{List}^{\uparrow}$ *spine, the* $\mathsf{Bool}^{\uparrow}$ *component rows, the component classifier, the length ledger,* hsame *transport, and* Pkg *provenance already listed in the public export and scoped dependency theorems.*

The bridge boundary has no coordinate for numeric binary encodings, hashing, matrix indexing, overflow, or error-correction metrics.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorUp StdBridge`

Proof. Read the public packet through Definition 346.7. It exposes only the unary length, boolean list spine, component classifier rows, length ledger, hsame transports, Cont routes, and package provenance.

Apply Definition 346.8. This fixes the whole dependency surface over $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, so the standard finite boolean-vector reading has no additional field to inspect.

The exclusions follow from the same projection. Numeric binary encodings, hashing, matrix indexing, overflow, and error-correction metrics are absent from both cited surfaces, so none belongs to the bridge boundary. □

Theorem 346.11 (BitVector fixed-width consumer completeness)**.** *Every finite-data consumer of* $\mathsf{BitVector}^{\uparrow}$ *is completely determined by the unary* $\mathsf{Nat}^{\uparrow}$ *width row,* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Bool}^{\uparrow}$ *histories, component classifier rows, length ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

The fixed-width surface excludes host bit-string equality, numeric binary primitives, padding, hashing, matrix indexing, overflow, and error-correction metrics.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorSource fixed width consumer completeness`

Proof. First apply Definition 346.7. It exposes the unary width row, boolean list spine, component classifier rows, length ledger, transport, continuation, package, and local name evidence as the public finite-data readback surface.

Next use Definition 346.8. This binds the same rows over $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, so the consumer has no separate host bit-string or arithmetic coordinate to inspect.

Apply Definition 346.9. It proves that the abstract fixed-width interface is exhausted by exactly those rows and that padding, hashing, matrix indexing, overflow, and error-correction metrics are not exported fields.

Finally Definition 346.10 identifies the same surface with the standard finite boolean-vector boundary. Therefore every finite-data consumer read is determined by the displayed fixed-width BEDC rows, with numeric binary primitives and host equality still excluded. □

Theorem 346.12 (BitVector fixed-length consumer determinacy)**.** *Any downstream consumer that factors through the* $\mathsf{BitVector}^{\uparrow}$ *public export, in particular a* $\mathsf{Trie}^{\uparrow}$ *key read, is determined by the unary* $\mathsf{Nat}^{\uparrow}$ *length row, the* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Bool}^{\uparrow}$ *entries, the length ledger, componentwise classifier rows,* hsame *transports,* Cont *routes, and* Pkg *provenance.*

No padding, overflow, hash, or host bit-string equality can alter the read.

Lean4 ✓ `BEDC.Derived.BitVectorUp.BitVectorBoolSpineLedger fixed length consumer determinacy`

Proof. Apply Definition 346.4. It fixes the displayed positions of the finite boolean list by the unary length row and the length ledger.

Next use Definition 346.7. The public export exposes exactly the $\mathsf{Nat}^{\uparrow}$ length, $\mathsf{List}^{\uparrow}$ spine, $\mathsf{Bool}^{\uparrow}$ component rows, classifier rows, transport, continuation, and provenance data available to downstream finite-data consumers.

Finally apply Definition 346.10. The standard finite boolean-vector boundary reads the same public surface, so a $\mathsf{Trie}^{\uparrow}$ key read or any other consumer through the export has no coordinate from which padding, overflow, hashing, or host bit-string equality could change the result. □

Theorem 346.13 (BitVector mature Trie key consumer example)**.** *A* $\mathsf{Trie}^{\uparrow}$ *key read that factors through a* $\mathsf{BitVector}^{\uparrow}$ *public export is determined by the unary* $\mathsf{Nat}^{\uparrow}$ *width row, the* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Bool}^{\uparrow}$ *histories, component classifier rows, the length ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

The consumer receives no host bit-string equality, numeric binary encoding, padding rule, overflow convention, hashing coordinate, or matrix-indexing operation.

Proof. Apply Definition 346.4. It exposes only the finite positions named by the unary width row and the length ledger.

Use Definition 346.7 to read the exported surface. The available data are the $\mathsf{List}^{\uparrow}$ spine of $\mathsf{Bool}^{\uparrow}$ histories, component classifier rows, transport, continuation, provenance, and local certificate evidence.

Next apply Definition 346.9 and Definition 346.10. These identify the same fixed-length finite boolean-vector surface without adding binary numerals, padding, overflow, hashing, or matrix indexing.

Thus a $\mathsf{Trie}^{\uparrow}$ key read is one downstream consumer of that finite surface. Once the read factors through the $\mathsf{BitVector}^{\uparrow}$ export, every observable key coordinate is already fixed by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{BitVector}^{\uparrow}}$.

Constructive story: A bit-vector packet is generated by a unary $\mathsf{Nat}^{\uparrow}$ length row, a $\mathsf{List}^{\uparrow}$ spine of $\mathsf{Bool}^{\uparrow}$ histories, a length ledger exposing exactly the displayed positions, and Pkg provenance for the finite-data dependencies.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BitVector}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 346.7, Definition 346.8, Definition 346.10, Definition 346.11, Definition 346.12, and Definition 346.13 identify the public fixed-length $\mathsf{Bool}^{\uparrow}/\mathsf{List}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ packet with the standard finite boolean-vector boundary and give the mature consumer-completeness surface, including the Trie key consumer example, component classifier rows, length ledger, hsame transport, Cont routes, Pkg provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BitVector}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.BitVectorUp.BitVectorPacket\_carrier\_transport`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close numeric binary encodings, hashing semantics, error-correction metrics, matrix indexing operations, machine-word overflow, or Lean bridge checking.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 347

A Concrete Naming Certificate for $\mathsf{RealLimit}^{\uparrow}$

The $\mathsf{RealLimit}^{\uparrow}$ interface names the finite saturated frontier where a scheduled regular rational stream can be read as a real-facing seal source. Following Bishop–Bridges constructive analysis, convergence is carried by explicit Cauchy and modulus data, not by a choice-selected limiting point. The packet below therefore records only certified stream windows, dyadic tolerance rows, regular-sequence reads, and the visible $\mathsf{Real}^{\uparrow}$ endpoint row available to downstream consumers.

The sibling dependency route is finite throughout: chapter 126 supplies the schedule surface, chapter 301 supplies the regular rational source, chapter 291 supplies the dyadic tolerance surface, and chapter 38 supplies the real-seal endpoint consumed by $\mathsf{RealLimit}^{\uparrow}$. These four surfaces are the rows later read as S, Q, D, E in the carrier packet; the limit interface only binds their displayed route through hsame, Cont, Pkg, and NameCert.

Definition 347.1 (RealLimit BHist carrier)**.** *A* $\mathsf{RealLimit}^{\uparrow}$ *packet is a finite* BHist *source row*

$$R = (S, Q, E, D, H, C, P, N)$$

with the following visible coordinates:

1. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite windows through which the approximation stream is inspected;*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream read on those scheduled windows;*
3. *E is the proposed* $\mathsf{Real}^{\uparrow}$ *seal endpoint consumed by real-facing interfaces;*
4. *D is the displayed family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance windows used for finite error reads;*
5. *H records the componentwise* hsame *rows for stream schedule, regular-sequence source, seal endpoint, and dyadic tolerance rows;*
6. *C records the visible* Cont *routes through which a consumer reaches the selected windows and their common refinements;*
7. *P is the* Pkg *provenance row naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and L0–L5 kernel dependencies;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger row.*

The carrier predicate is $\mathsf{RealLimitPacket}(S, Q, E, D, H, C, P, N)$*. Its classifier compares two accepted packets by passing to common finite refinements of the stream schedules and reading the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row there. All agreement is then transported componentwise through the displayed* hsame *rows and the same* Cont *routes; no completed real equality or quotient of streams is used.*

Theorem 347.2 (RealLimit NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier: the finite packet of Definition 347.1 with its schedule, source stream, seal endpoint, dyadic tolerance windows, transport, continuation, provenance, and local ledger;*
2. *classifier: common finite schedule refinements followed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance read;*
3. *stability: componentwise* hsame *transport and visible* Cont *refinement preserve the selected stream window and tolerance comparison;*
4. *exactness/readback: every* $\mathsf{Real}^{\uparrow}$ *read factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance window used by the schedule;*
5. *ledger: the* Pkg *and* NameCert *rows expose only the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and L0–L5 kernel dependencies.*

This is an obligation surface for finite constructive limit names; it is not a claim of real completeness.

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket namecert obligation surface`

Proof. Project the carrier of Definition 347.1. The displayed coordinates already list the schedule, source stream, proposed seal endpoint, dyadic tolerance rows, transport witnesses, continuation routes, provenance, and local ledger.

For the classifier row, compare two packets only after selecting a common finite refinement of their visible schedules. The $\mathsf{RegSeqRat}^{\uparrow}$ source stream is read at that refined window, and the dyadic tolerance row is the one named by the same packet coordinate.

Stability follows by transporting each visible coordinate through its stored hsame row and reading refinements only through the displayed Cont routes. The readback row is exact because the sole $\mathsf{Real}^{\uparrow}$ coordinate is the seal endpoint E, and E is reached from the regular-sequence and dyadic tolerance rows already present in the packet.

The ledger row is exhausted by P and N. Since the carrier has no coordinate for a completed real, no quotient stream, and no choice-selected limit, those objects cannot be exported by the local certificate. □

Theorem 347.3 (RealLimit RegSeqRat stream-name handoff)**.** *Every accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet hands a* $\mathsf{StreamName}^{\uparrow}$ *schedule to a* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream before any* $\mathsf{Real}^{\uparrow}$ *seal read is available. More precisely, a real-facing consumer may read the seal endpoint only through the scheduled regular-sequence tolerance window, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *error row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, no equality of quotient streams, and no choice-selected limiting object.*

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket scheduled handoff transport`

Proof. Read the accepted packet $R = (S, Q, E, D, H, C, P, N)$ through Definition 347.1. The coordinate S selects the finite observation windows, while Q supplies the regular rational stream read on those windows.

The dyadic tolerance coordinate D records the finite error window used for the same read. A consumer reaches it only through the continuation routes in C, and endpoint replacement is constrained by the componentwise transport rows in H.

The proposed real-facing endpoint is the single coordinate E. Because E is packaged together with S, Q, D, H, C, P, and N, any read of E has the scheduled regular-sequence handoff as its visible source. There is no remaining coordinate from which a completeness principle, stream quotient, or selected limiting value could be obtained. □

Theorem 347.4 (RealLimit common-refinement classifier)**.** *For accepted* $\mathsf{RealLimit}^{\uparrow}$ *packets*

$$R = (S, Q, E, D, H, C, P, N) \quad \textit{and} \quad R' = (S', Q', E', D', H', C', P', N'),$$

the local classifier can place R and R' in the same real-limit name only through a displayed common finite $\mathsf{StreamName}^{\uparrow}$ *schedule refinement. On that common refinement it reads the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *error rows, and the componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the two packets. The classifier has no row for completed real equality, quotient-stream equality, or a selected limit.*

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket common refinement certificate`

Proof. Project both packets through Definition 347.1. The schedule coordinates S, S' are the only sources of observation windows, and the source-stream coordinates Q, Q' are the only regular rational rows available at those windows.

Refine the displayed schedules to a common finite window. On that window the classifier reads the $\mathsf{RegSeqRat}^{\uparrow}$ tolerance row named by the source stream coordinates and the $\mathsf{DyadicRatCore}^{\uparrow}$ error rows named by D, D'.

The remaining comparison data is componentwise. Transport is supplied by H, H', continuation routes by C, C', package provenance by P, P', and local ledger evidence by N, N'. Repacking these displayed rows gives the classifier, and the carrier contains no coordinate from which completed real equality, quotient-stream equality, or a selected limit could be read. □

Theorem 347.5 (RealLimit transport stability boundary)**.** *For an accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet,* hsame *transport and* Cont *refinement preserve the selected* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream, the* $\mathsf{Real}^{\uparrow}$ *seal endpoint, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance windows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger. Every transported row remains a displayed row of the finite packet surface.*

Proof. Read the packet $R = (S, Q, E, D, H, C, P, N)$ from Definition 347.1. The coordinates H and C are exactly the stored componentwise transport and refinement routes for the schedule, source stream, seal endpoint, dyadic tolerances, package provenance, and local ledger.

Apply Definition 347.4 to identify the common finite schedule surface on which a transported comparison is allowed to take place. The classifier exposes only the displayed schedule, source, tolerance, transport, continuation, provenance, and certificate rows.

Transporting along H and refining along C therefore keeps every consumer row inside the same finite packet surface. Since no other carrier coordinate is available, the seal endpoint, dyadic tolerance windows, provenance, and ledger remain displayed rows rather than hidden real-completion data. □

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket transport stability boundary`

Theorem 347.6 (RealLimit readback ledger exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet factors through the scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *source, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the displayed* hsame *and* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger. The readback surface excludes* $\mathsf{Real}^{\uparrow}$ *completeness, quotient-stream equality, and selected limiting objects.*

Proof. Start with Definition 347.3. It routes a real-facing read through the finite $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ source stream, the dyadic tolerance row, transport, continuation, package provenance, and local certificate.

Use Definition 347.4 for paired readbacks. It shows that compatible packet reads share only a displayed common finite schedule refinement and the same scheduled tolerance surface.

Use Definition 347.5 for replacement of rows along hsame and Cont. The transported read remains on the visible packet coordinates, so the ledger is exhausted by S, Q, D, H, C, P, N together with the seal endpoint E.

No coordinate of the carrier stores real completeness, quotient-stream equality, or a selected limiting object. Those objects therefore cannot appear in the $\mathsf{Real}^{\uparrow}$-facing readback ledger. □

Theorem 347.7 (RealLimit scoped window boundary)**.** *The carrier, scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, common-refinement classifier, transport-stability boundary, and readback ledger exhaustion assemble a scoped finite* $\mathsf{RealLimit}^{\uparrow}$ *packet over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The packet is finite and real-facing. It excludes $\mathsf{Real}^{\uparrow}$ *completeness, quotient-stream equality, selected limits, and choice data.*

Proof. Project the accepted packet as $R = (S, Q, E, D, H, C, P, N)$ from Definition 347.1. The coordinates are the schedule, regular rational source, real seal endpoint, dyadic tolerance rows, componentwise transport, continuation routes, package provenance, and local name-certificate ledger.

Apply Definition 347.3 before any seal read. This routes the real-facing coordinate E through the finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the displayed H, C, P, N rows.

For paired packets, use Definition 347.4. Common finite refinements are the only classifier surface available to compare scheduled source reads. For replacement of rows, use Definition 347.5; the transported packet remains on the same displayed finite coordinates.

Finally close by Definition 347.6. The readback ledger is exhausted by S, Q, E, D, H, C, P, N, so the scoped boundary adds no completeness principle, quotient-stream equality, selected limit, or choice object beyond the finite packet. □

Theorem 347.8 (RealLimit classifier obligation)**.** *For accepted* $\mathsf{RealLimit}^{\uparrow}$ *packets* $R = (S, Q, E, D, H, C, P, N)$ *and* $R' = (S', Q', E', D', H', C', P', N')$*, the local classifier may identify their real-limit names only through common finite* $\mathsf{StreamName}^{\uparrow}$ *schedule refinements, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance read on those refinements, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows, and the visible* hsame *and* Cont *routes carried by the two packets. It has no classifier row for completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient-stream equality, or a selected limit.*

Proof. Project the two carriers using Definition 347.1. The schedule coordinates S and S' are the only rows that choose finite observation windows, and the regular-sequence coordinates Q and Q' are the only rows read at those windows.

Apply Definition 347.3. The handoff forces each real-facing read to pass through the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ source stream and its $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row.

Thus a classifier comparison can only pass to common finite schedule refinements, compare the same tolerance read there, and transport the dyadic windows through H, H' and C, C'. Since the carrier has no additional row for completed real equality, quotient-stream equality, or a selected limit, no such row can enter the classifier. □

Theorem 347.9 (RealLimit stability obligation)**.** *Componentwise* hsame *transport and visible* Cont *refinement preserve the selected* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream, the* $\mathsf{Real}^{\uparrow}$ *seal endpoint, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *row of an accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet.*

Proof. Project $R = (S, Q, E, D, H, C, P, N)$ from Definition 347.1. The coordinates H and C are exactly the stored transport and refinement witnesses for the visible rows of the packet.

Use Definition 347.3 to identify the consumer path through S, Q, D, H, C, P, and N before the seal endpoint E is read. Any refinement of the consumer path is therefore a refinement of these displayed rows.

The componentwise hsame rows preserve each named coordinate, while the Cont routes restrict refinements to the visible observation windows. Hence the selected schedule, source stream, seal endpoint, dyadic tolerance rows, package row, and local certificate row remain the same visible packet data. □

Theorem 347.10 (RealLimit readback obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read from an accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance window selected by its* $\mathsf{StreamName}^{\uparrow}$ *schedule. The readback cannot inspect a real-completeness principle, quotient-stream equality, or a selected limiting object.*

Proof. Project the packet $R = (S, Q, E, D, H, C, P, N)$. The endpoint coordinate E is the only real-facing seal row, and the schedule, source stream, and tolerance coordinates S, Q, and D are the finite data attached to that row.

Apply Definition 347.3. It routes every read of E through the scheduled regular rational stream, the displayed dyadic tolerance window, and the visible transport, continuation, package, and certificate rows.

No coordinate of Definition 347.1 stores completeness, quotient-stream equality, or a selected limit. Therefore the readback surface has no source from which any of those objects can be read. □

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket readback obligation`

Theorem 347.11 (RealLimit transported readback ledger stability)**.** *Let* $R = (S, Q, E, D, H, C, P, N)$ *and* $R' = (S', Q', E', D', H', C', P', N')$ *be accepted* $\mathsf{RealLimit}^{\uparrow}$ *packets. Suppose the displayed packet rows are related componentwise by* hsame *on the* $\mathsf{StreamName}^{\uparrow}$ *schedules* S, S'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* Q, Q'*, the* $\mathsf{Real}^{\uparrow}$ *endpoint rows* E, E'*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance windows* D, D'*, the* Pkg *provenance rows* P, P'*, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *rows* N, N'*, and suppose the visible schedule refinement routes are the same* Cont *routes carried by* C *and* C'*.*

Then every $\mathsf{Real}^{\uparrow}$*-facing readback ledger row of* R *transports to a* $\mathsf{Real}^{\uparrow}$*-facing readback ledger row of* R' *through those same* Cont *refinement routes. The transported ledger remains inside the finite seal surface given by the schedules, source rows, endpoint rows, tolerance windows, componentwise* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It does not expose* $\mathsf{Real}^{\uparrow}$ *completeness, quotient-stream equality, or a selected limiting object.*

Proof. Start with Definition 347.3. Each read of the real-facing endpoint E is routed through the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance window, transport rows, continuation routes, package row, and local naming ledger before any consumer can observe the endpoint.

Next use Definition 347.5. The assumed componentwise hsame relations and shared Cont refinements replace the visible rows of R by the corresponding visible rows of R' without leaving the displayed finite carrier surface.

Finally apply Definition 347.10. The readback row that survives this transport still factors through the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ source and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance window. Since none of the transported coordinates stores completeness, quotient-stream equality, or a selected limit, the readback ledger cannot acquire any such row. □

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket transported readback ledger stability`

Theorem 347.12 (RealLimit ledger obligation)**.** *The* $\mathsf{RealLimit}^{\uparrow}$ *ledger exposes only* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *L0–L5 kernel rows,* Pkg, *and* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$. *It exposes no completed real, quotient stream, host equality, analytic convergence theorem, or selected limit.*

Proof. Project the provenance and local ledger coordinates P and N from Definition 347.1. By definition, P names the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and L0–L5 kernel dependencies, while N is the local $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ row.

Apply Definition 347.3. The handoff shows that the real-facing endpoint is reached only through the same finite schedule, regular-sequence, dyadic tolerance, transport, continuation, package, and certificate rows.

The ledger is therefore exhausted by the displayed package and certificate coordinates. Since no carrier coordinate names a completed real, quotient stream, host equality, analytic convergence theorem, or selected limit, the ledger cannot expose any of them. □

Theorem 347.13 (RealLimit scoped kernel interface)**.** *Every accepted* $\mathsf{RealLimit}^{\uparrow}$ *packet whose readback, classifier, stability, and ledger rows satisfy Definition 347.8, Definition 347.9, Definition 347.10, and Definition 347.12 has its real-facing seal read wholly scoped by the displayed* BHist *packet, componentwise* hsame *transport,* Cont *schedule refinements,* Pkg *provenance,* $\mathsf{NameCert}$ *ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *endpoint row.*

Within that scope the $\mathsf{Real}^{\uparrow}$ *endpoint is only a sealed consumer row reached through the handoff of Definition 347.3. The binding does not export a completed real, quotient stream, host equality, analytic convergence theorem, or selected limiting object.*

Proof. Project an accepted packet as $R = (S, Q, E, D, H, C, P, N)$ from Definition 347.1. The schedule, regular rational source, endpoint seal, dyadic tolerances, transport, continuation, package, and naming coordinates are the only rows available to the four obligations.

Apply Definition 347.3. It sends the real-facing endpoint E through the same S, Q, D, H, C, P, and N rows, so the downstream $\mathsf{Real}^{\uparrow}$ handoff has no wider surface than the displayed BEDC packet.

The classifier obligation reads only those coordinates; the stability obligation transports them by hsame and Cont; the readback obligation factors every endpoint read through the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows; and the ledger obligation is exhausted by P and N.

Since no coordinate of the carrier stores a completed real, quotient stream, host equality, analytic convergence theorem, or selected limiting object, none of the four obligations can acquire such an object through the scoped kernel interface. □

Theorem 347.14 (RealLimit public real-window export)**.** *The carrier, classifier, transport, readback, ledger, and scoped kernel interface theorems assemble one public* $\mathsf{RealLimit}^{\uparrow}$ *consumer export over the same* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance window and* $\mathsf{Real}^{\uparrow}$ *endpoint row kept inside that finite surface.*

This export exposes only the scheduled regular rational stream, dyadic tolerance windows, real-facing seal row, transport and continuation routes, package provenance, and local naming ledger. It exports no completed real, quotient stream equality, analytic convergence theorem, or selected limiting object.

Proof. Project the carrier rows from Definition 347.1. The schedule, regular rational source stream, real-facing seal, dyadic tolerance window, transport rows, continuation rows, package provenance, and local naming ledger are the available finite-window coordinates.

Apply Definition 347.3 to route each real-facing read through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows.

Use Definition 347.6 to account for every consumer readback row. The ledger names the same BHist packet, hsame transports, Cont refinements, Pkg provenance, and $\mathsf{NameCert}$ row used by the scoped kernel.

Finally invoke Definition 347.13. It packages the classifier, transport, readback, and ledger obligations into the same finite BEDC surface, giving one public consumer export without adding any completed real, quotient stream equality, analytic convergence theorem, or selected limit. □

Theorem 347.15 (RealLimit standard finite seal bridge)**.** *The public real-window export, scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, scoped kernel interface, and readback ledger exhaustion determine a conservative standard finite-seal reading and readback for the displayed* $\mathsf{RealLimit}^{\uparrow}$ *packet.*

The readback recovers exactly the $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream,* $\mathsf{Real}^{\uparrow}$ *endpoint row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, componentwise* hsame*, visible* Cont*,* Pkg *provenance, and local* $\mathsf{NameCert}$ *ledger. It exports no* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limit object, or host completion principle.*

Proof. Apply Definition 347.3. This fixes the scheduled stream-name and regular rational source coordinates used by every finite seal read.

Next use Definition 347.6. It accounts for the $\mathsf{Real}^{\uparrow}$ endpoint row, dyadic tolerance rows, componentwise hsame transport, visible Cont routes, package provenance, and local naming ledger.

Apply Definition 347.13 to keep the classifier, transport, readback, and ledger obligations inside the displayed finite BEDC carrier.

Finally use Definition 347.14. The carrier coordinates have now been exhausted: the standard readback is precisely the listed finite-seal packet, and no row remains from which to export $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, a selected limit object, or a host completion principle. □

Theorem 347.16 (RealLimit standard bridge boundary)**.** *The displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, and downstream* $\mathsf{Real}^{\uparrow}$ *finite-window rows determine the standard bridge boundary for a candidate* $\mathsf{RealLimit}^{\uparrow}$ *name. Every standard finite observation of such a candidate factors through Definition 347.14.*

The boundary introduces no quotient stream equality, selected limit object, or $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RealLimitUp.RealLimitPacket standard bridge boundary`

Proof. Start with the obligation surface of Definition 347.2. Its rows are the finite BHist carrier, componentwise hsame routes, Cont schedule refinements, package provenance, and local $\mathsf{NameCert}$ ledger.

Use Definition 347.3 to route the candidate stream observation through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows. Then apply Definition 347.4 and Definition 347.5 to keep classifier and transport reads inside the same finite-window surface.

Next apply Definition 347.6. It accounts for the readback and ledger rows used by every finite consumer of the candidate name.

Finally use Definition 347.13 and Definition 347.14. These package the exhausted rows as the public real-window export, so every standard finite observation factors through that export. None of the cited rows contains a quotient stream equality, selected limit object, or $\mathsf{Real}^{\uparrow}$ completeness witness, hence the standard boundary cannot introduce one. □

Theorem 347.17 (RealLimit finite real-seal consumer completeness)**.** *Every finite consumer read of a displayed* $\mathsf{RealLimit}^{\uparrow}$ *packet factors through exactly the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source stream row, the* $\mathsf{Real}^{\uparrow}$ *endpoint row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the componentwise* hsame *rows, the visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger. Conversely, those rows repack as the finite real-seal consumer surface.*

This consumer surface adds no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit object, or host completion principle beyond the standard bridge boundary.*

Proof. Use Definition 347.3 to identify the scheduled stream name and regular rational source stream that any finite consumer can inspect.

Apply Definition 347.6. This accounts for the $\mathsf{Real}^{\uparrow}$ endpoint row, dyadic tolerance rows, componentwise hsame transports, visible Cont routes, package provenance, and local naming ledger.

Next use Definition 347.14. It presents the same exhausted rows as the public real-window packet available to downstream finite consumers.

Finally apply Definition 347.16. The bridge boundary contributes no hidden completion, quotient, or selected-limit row, so the displayed rows are also sufficient in the reverse direction to repack the finite real-seal consumer surface. □

Theorem 347.18 (RealLimit finite-seal readback transport determinacy)**.** *Let p and p' be accepted* $\mathsf{RealLimit}^{\uparrow}$ *packets. Suppose the displayed rows connect their schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *stream,* $\mathsf{Real}^{\uparrow}$ *endpoint, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance coordinates by componentwise* hsame *transport. Suppose also that the readback rows are* $\mathsf{Cont}(\mathsf{endpoint}, \mathsf{tolerance}, \mathsf{readback})$ *and* $\mathsf{Cont}(\mathsf{endpoint}', \mathsf{tolerance}', \mathsf{readback}')$.

Then every finite real-seal consumer read through p transports to the corresponding read through p' on the same BEDC consumer surface: the scheduled $\mathsf{StreamName}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *stream row,* $\mathsf{Real}^{\uparrow}$ *endpoint row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, finite handoff row, visible* Cont *route, local readback row,* Pkg *provenance row, and* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger row. The determinacy claim does not assert* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected limit object, or host completion.*

Proof. Start with Definition 347.3. It places the consumer read on the displayed schedule and $\mathsf{RegSeqRat}^{\uparrow}$ source stream before any endpoint interpretation is used.

Apply Definition 347.5. The assumed componentwise hsame rows carry the schedule, stream, endpoint, and tolerance coordinates from p to p' while keeping the handoff and visible Cont route inside the same finite packet surface.

Use Definition 347.10. The two displayed readback rows are therefore read as the same local BEDC obligation row over the transported endpoint and tolerance coordinates, with the Pkg provenance and $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ ledger still visible.

Finally apply Definition 347.17. Since that surface is exactly the schedule, stream, endpoint, tolerance, handoff, route, ledger, readback, package, and name-certificate rows, the transported readback has no additional coordinate on which a second consumer interpretation could depend. The excluded completion, quotient, selected-limit, and host-completion principles are not present in any cited row. □

Theorem 347.19 (RealLimit stream-tail window normal form)**.** *For accepted* $\mathsf{RealLimit}^{\uparrow}$ *packets whose scheduled* $\mathsf{StreamName}^{\uparrow}$ *rows refine to the same displayed finite tail window, every* $\mathsf{Real}^{\uparrow}$*-facing readback factors through exactly the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the displayed* hsame *transports, the visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$ *ledger. The tail-window normal form is determined entirely by those finite rows.*

The normal form exports no completed real equality, quotient stream, selected limit, host completion witness, or choice data.

Proof. Project each accepted packet to the finite carrier of Definition 347.1. The projected rows are the scheduled stream-name coordinate, the regular rational source coordinate, the dyadic tolerance coordinates, componentwise hsame transport, visible Cont routes, package provenance, and the local naming ledger.

Apply Definition 347.4 to the two scheduled $\mathsf{StreamName}^{\uparrow}$ rows. The common finite tail window fixes the classifier surface on which the readback rows are compared, so later tail data cannot change the displayed source or tolerance coordinates.

Use Definition 347.6 and Definition 347.14. These identify every $\mathsf{Real}^{\uparrow}$-facing readback row with the same regular rational source row, dyadic tolerance rows, componentwise transports, continuation routes, package provenance, and naming ledger on the common window.

Finally invoke Definition 347.16. The boundary contains no coordinate for completed real equality, quotient stream formation, selected limits, host completion, or choice data. Carrier exhaustion therefore leaves exactly the stated finite tail-window normal form. □

Theorem 347.20 (RealLimit shifted-schedule readback boundary)**.** *Let P be a displayed* $\mathsf{RealLimit}^{\uparrow}$ *packet whose* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{StreamName}^{\uparrow}$ *schedule is transported by a certified unary successor row or by a certified finite tail selector. Then every finite* $\mathsf{Real}^{\uparrow}$ *seal readback of the transported packet exposes the same displayed endpoint, readback ledger,* hsame *rows,* Cont *routes, and* Pkg *provenance rows as the original packet after schedule transport.*

The boundary exports only that finite schedule-transport surface. It does not export a completeness theorem, selected limit, quotient representative, or host-index equality.

Proof. Apply Definition 347.3. The certified successor or finite tail selector changes only the displayed schedule route and keeps the $\mathsf{RegSeqRat}^{\uparrow}$ source stream and $\mathsf{StreamName}^{\uparrow}$ row visible.

Next use Definition 347.11. The transported readback ledger remains attached to the same endpoint row through the displayed hsame and Cont transport rows, with the Pkg provenance still part of the finite packet.

Finally apply Definition 347.14 and Definition 347.17. These identify the finite $\mathsf{Real}^{\uparrow}$ seal consumer surface with exactly the transported schedule, source stream, endpoint, readback ledger, hsame, Cont, and package rows. Since none of those rows contains completeness, a selected limit, a quotient representative, or host-index equality, the shifted-schedule readback cannot export such data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimit}^{\uparrow}}$.

Constructive story: A finite limit-name packet records a stream schedule, regular rational source stream, proposed real seal endpoint, dyadic tolerance windows, transport rows, continuation routes, package provenance, and local naming ledger.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealLimit}\uparrow}$, maturePackageClosed)

Scope closed: Definition 347.13, Definition 347.6, Definition 347.3, Definition 347.14, Definition 347.15, Definition 347.16, Definition 347.17, and Definition 347.18 bind the classifier, stability, readback, ledger, public real-window export, standard finite-seal bridge, standard bridge boundary, finite real-seal consumer surface, and transported readback determinacy to the finite carrier of Definition 347.1, componentwise hsame transport, Cont schedule refinements, Pkg provenance, the NameCert ledger, and the downstream Real$^{\uparrow}$ endpoint row.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealLimit}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.RealLimitUp.RealLimitPacket\_readback\_obligation`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, quotient stream equality, a selected limit object, or host completion.

Upgrade path: No further theory-axis upgrade is available for this chapter; formal upgrades remain on the verification axis.

Chapter 348

A Concrete Naming Certificate for $\mathsf{ModelPredictiveControl}^{\uparrow}$

The $\mathsf{ModelPredictiveControl}^{\uparrow}$ interface records the finite receding-horizon surface used by control packets that repeatedly solve a displayed finite prediction problem and export only the first control row. The reference point is Rawlings, Mayne, and Diehl, *Model Predictive Control: Theory, Computation, and Design* (2017), together with Boyd and Vandenberghe, *Convex Optimization* (2004) for finite convex constraint rows. The BEDC certificate below names the finite rows of that surface over $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, and $\mathsf{ControlControllability}^{\uparrow}$. It does not assert infinite-horizon optimality, analytic convergence, stochastic control, recursive feasibility beyond the displayed terminal rows, or host optimizer semantics. The receding-horizon row consumes the LQR finite-control surface of chapter 296 and the Riccati terminal ledger of chapter 307 before the ConvexSet and Polytope constraint rows are read.

Definition 348.1 (Model predictive control BHist carrier)**.** *A* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, U, N, A, Q, W, J, T, O, F, H, R, P, C)$$

whose visible rows are:

1. *a current-state row X read through* $\mathsf{StateSpaceModel}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$*;*
2. *an input row U read through* $\mathsf{VecSpace}^{\uparrow}$ *and the displayed control dependency certificates;*
3. *a finite prediction-horizon row N read as a unary length bound;*
4. *a dynamics-transition row A read through* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{StateSpaceModel}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, and* $\mathsf{VecSpace}^{\uparrow}$*;*
5. *a quadratic-cost row Q read through* $\mathsf{LQR}^{\uparrow}$ *and* $\mathsf{Riccati}^{\uparrow}$*;*
6. *a constraint-window row W naming the displayed admissible state and control rows through* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{Polytope}^{\uparrow}$ *for each horizon index;*
7. *a stage-cost ledger J read through* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and the finite-cost face of* $\mathsf{LQR}^{\uparrow}$*;*
8. *a terminal-cost or terminal-region row T read through* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{Riccati}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*, and* $\mathsf{Polytope}^{\uparrow}$ *when those rows are present;*
9. *an optimizer-candidate row O containing only a displayed finite control sequence candidate and its finite rollout ledger;*

10. *a first-control extraction row F selecting the initial control entry from O;*
11. *componentwise* hsame *transports H for the state, horizon, transition, window, cost, terminal, candidate, first-control, and provenance rows;*
12. Cont *rollout routes R for reading the finite prediction recursion, stage-cost accumulation, terminal read, and first-control extraction;*
13. *a* Pkg *provenance row P naming the consumed state-space, dynamical, matrix, vector-space, real, LQR, Riccati, convex-set, polytope, observability, and controllability certificates;*
14. *a local* NameCert *provenance row C for the finite receding-horizon surface.*

Two accepted packets classify together only when these displayed rows transport componentwise by H and every rollout, cost, terminal, and first-control read follows one of the recorded Cont *routes in R.*

Lean4 (definition) `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlBHistCarrier`

Theorem 348.2 (Model predictive control finite-horizon obligation)**.** *Every accepted* ModelPredictiveControl$^{\uparrow}$ *packet exposes its prediction ledger as exactly the finite row family consisting of the current-state row, the input row, the unary horizon row, the displayed dynamics-transition row, the quadratic-cost rows, the rollout* Cont *routes, and the provenance rows of Definition 348.1. The prediction surface exports no host trajectory space, no infinite rollout, and no unlogged model semantics.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket finite horizon obligation`

Proof. Project the accepted carrier of Definition 348.1. The only coordinates that can generate prediction rows are X, U, N, A, Q, R, P, and the transport data H.

Read the rollout one horizon index at a time. Each step follows a recorded Cont route from the displayed state and transition rows to the next displayed state row, so the ledger is finite and indexed by N.

Transporting the packet replaces each displayed coordinate by its hsame match and reuses the same rollout routes. Since the carrier has no coordinate for a host trajectory space or an infinite stream, the prediction ledger is exhausted by these finite rows. □

Theorem 348.3 (Model predictive control constraint ledger)**.** *For an accepted* ModelPredictiveControl$^{\uparrow}$ *packet, every admissibility read is the constraint-window row W together with the horizon index, state and control rows, rollout routes, convex and polytopal provenance, and terminal row recorded in Definition 348.1. The packet exports no ambient feasible set, no compactness assertion, and no recursive-feasibility theorem beyond the displayed terminal-cost or terminal-region row.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlConstraintLedger semantic name certificate`

Proof. Project the constraint coordinate W from the carrier. Its entries are indexed by the same finite horizon row N used by the prediction ledger, and they refer only to the displayed state and control rows through the consumed ConvexSet$^{\uparrow}$ and Polytope$^{\uparrow}$ certificates.

Use the rollout routes to align each constraint entry with the corresponding predicted state and candidate control row. The admissibility read is therefore a row projection from W, not an external feasible-set predicate.

The terminal coordinate T may record a terminal cost or terminal region row, but it is another displayed finite row. No coordinate supplies compactness, global feasibility, or recursive feasibility beyond this displayed terminal surface. □

Theorem 348.4 (Model predictive control receding-horizon boundary)**.** *The control exported by an accepted* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *packet is exactly the first-control extraction row F applied to the displayed optimizer-candidate row O. The stage-cost ledger J, terminal row T, constraint window W, rollout routes R,* hsame *transports H, and provenance rows P and C certify the finite candidate surface, but they do not assert that O is a globally optimal host solution.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket receding horizon boundary`

Proof. Project the candidate row O. It contains a displayed finite control sequence candidate together with the rollout and cost rows that make the candidate visible inside the packet.

Apply the extraction row F. This row selects the initial control entry from the displayed finite sequence and exports that row as the receding-horizon control read.

The cost, constraint, terminal, transport, continuation, and provenance rows certify how the candidate is displayed. None of these rows contains a host minimizer, a global optimality proof, or an external solver state, so the exported control row is precisely the first displayed entry of O. □

Theorem 348.5 (Model predictive control consumer boundary)**.** *Every downstream reader of an accepted* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *packet sees only the finite state, input, horizon, dynamics, quadratic-cost, constraint, rollout, terminal, first-control, transport, continuation, and provenance rows of Definition 348.1. A consumer cannot read an infinite-horizon optimum, analytic convergence proof, stochastic controller, or host optimizer semantics from the packet.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket consumer boundary`

Proof. Project an arbitrary downstream read of the accepted packet. The carrier provides only the displayed coordinates X, U, N, A, Q, W, J, T, O, F, H, R, P, and C.

The prediction, constraint, and receding-horizon theorems identify the only control-facing rows: finite rollout, finite admissibility, and first-control extraction. Each one is reached by a recorded Cont route and transported componentwise by hsame.

Since the packet has no coordinate for infinite horizons, analytic convergence data, stochastic laws, or host optimizer semantics, no consumer read can export those objects. □

Theorem 348.6 (Model predictive control NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ModelPredictiveControl}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the finite packet of Definition 348.1;*
2. *classifier transport: the componentwise* hsame *rows for the carrier coordinates and the* Cont *routes that preserve the displayed reads;*
3. *finite horizon rollout: the finite prediction rows of Definition 348.2;*
4. *constraint ledger exactness: the admissibility rows of Definition 348.3;*
5. *consumer boundary: the* Pkg *and* NameCert *rows of Definition 348.5 naming* $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, *and* $\mathsf{ControlControllability}^{\uparrow}$ *as the only imported certificate surfaces.*

These rows form the finite seed obligation surface for naming receding-horizon model predictive control data in BEDC.

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket namecert obligation surface`

Proof. The carrier row fixes every coordinate of the packet. Its displayed state, horizon, transition, window, cost, terminal, candidate, extraction, transport, continuation, package, and local name rows are all finite BHist data.

The finite-horizon, constraint-ledger, receding-horizon, and consumer-boundary theorems identify the only rollout, admissibility, exported-control, and downstream-consumer reads. Each read is a projection from the same finite packet and follows one of the recorded Cont routes.

The provenance row then lists exactly the consumed control and algebra certificates. Since no extra coordinate carries host optimization, analytic minimizers, compactness, infinite horizons, or unlogged feasibility data, the five rows exhaust the local $\mathsf{NameCert}_{\mathsf{ModelPredictiveControl}^{\uparrow}}$ surface. □

Theorem 348.7 (Model predictive control scoped kernel support)**.** *The finite MPC carrier, finite-horizon obligation, constraint ledger, receding-horizon boundary, consumer boundary, and* NameCert *obligation surface are scoped exactly by the displayed* BHist *packet rows,* hsame *transport rows,* Cont *rollout routes,* Pkg *provenance,* NameCert *provenance, and the cited* $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, *and* $\mathsf{ControlControllability}^{\uparrow}$ *sources. The scoped support exports no host optimizer semantics, infinite-horizon optimality, stochastic control, or unlogged feasibility data.*

Proof. Project the carrier coordinates $M = (X, U, N, A, Q, W, J, T, O, F, H, R, P, C)$ from Definition 348.1. These coordinates are the only state, input, horizon, transition, cost, constraint, terminal, candidate, first-control, transport, continuation, package, and naming rows in the packet.

Apply Definition 348.2, Definition 348.3, Definition 348.4, and Definition 348.5. They identify the prediction rows, admissibility rows, first-control read, and downstream consumer reads as projections from the same finite carrier, transported by hsame and routed by the displayed Cont rollout data.

Finally repack the five rows of Definition 348.6. The Pkg row names exactly the listed control, algebra, convexity, and real-number sources, while the local NameCert row records the receding-horizon surface. Coordinate exhaustion then excludes host optimizer semantics, infinite-horizon optimality, stochastic control, and unlogged feasibility data. □

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket scoped kernel support`

Theorem 348.8 (Model predictive control finite-horizon prefix restriction)**.** *Let* $M = (X, U, N, A, Q, W, J, T, O, F,$ *be an accepted* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *packet, and let* $m \leq N$ *be a certified unary prefix horizon. Form the prefix packet* $M_{\leq m}$ *by replacing the horizon row* N *by* m*, retaining the unindexed state, input, transition, quadratic cost, package, and local name-certificate rows, and retaining exactly the prediction, constraint-window, stage-cost, terminal, optimizer-candidate, first-control,* hsame*, and* Cont *rows whose horizon addresses lie in the prefix horizon* m*. Then* $M_{\leq m}$ *is an accepted* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *packet over horizon* m*.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket finite horizon prefix restriction`

Proof. Unfold the carrier of Definition 348.1. Every coordinate required by a $\mathsf{ModelPredictiveControl}^{\uparrow}$ packet is one of

$$X, U, N, A, Q, W, J, T, O, F, H, R, P, C.$$

After prefix restriction, the rows X, U, A, Q, P, and C are the same displayed rows, the horizon coordinate is the certified prefix m, and every indexed row in W, J, T, O, H, and R is a retained row whose address already occurred under the original horizon N.

The prediction ledger is stable under this operation. By Definition 348.2, an accepted MPC packet exposes its prediction surface exactly through the current-state, input, unary horizon, dynamics-transition, quadratic-cost, rollout Cont, transport, and provenance rows. Restricting from N to m keeps precisely those rows at the prefix addresses and changes no rollout endpoint outside the retained prefix. Thus the prediction ledger of $M_{\leq m}$ is the finite prediction ledger over m.

The admissibility, cost, and terminal reads are also prefix-local. By Definition 348.3, every admissibility read comes from the constraint-window row W together with the same horizon index, state and control rows, rollout routes, convex and polytopal provenance, and terminal row. Since the prefix packet keeps exactly the rows indexed by the certified prefix horizon, each retained constraint, stage-cost, and terminal read has the same source row and the same recorded route as it had in M. No ambient feasible set or external compactness assertion is used to rebuild acceptance.

For the receding-horizon export, use Definition 348.4. The exported control is exactly the first-control extraction row F applied to the displayed optimizer-candidate row O. The prefix packet retains that first-control row and the initial candidate entry, so the receding-horizon boundary remains the same boundary read inside the shorter packet.

It remains to check transport and provenance. The componentwise hsame rows and the Cont routes retained in $M_{\leq m}$ have the same endpoints after restriction, because both endpoints are prefix rows. The Pkg row still names the same consumed $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, and $\mathsf{ControlControllability}^{\uparrow}$ certificates, and the local $\mathsf{NameCert}$ row still names the finite receding-horizon surface. Repacking these retained carrier, prediction, constraint, first-control, transport, continuation, package, and name-certificate rows gives the accepted $\mathsf{ModelPredictiveControl}^{\uparrow}$ packet over m. □

Theorem 348.9 (Model predictive control finite rollout bridge boundary)**.** *For every finite public* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *packet displayed by the carrier, rollout, constraint, prefix, package, and* $\mathsf{NameCert}$ *rows of this chapter, there is a bridge boundary from that packet to the abstract receding-horizon control interface. The finite-horizon prefix restriction supplies the source row, the constraint ledger supplies hidden constraint provenance, and the consumer boundary transports the exported packet through* Cont *and* hsame *without exposing host optimization primitives.*

Lean4 ✓ `BEDC.Derived.ModelPredictiveControlUp.ModelPredictiveControlPacket finite rollout bridge boundary`

Proof. Begin with the finite source packet. Apply Definition 348.8 to replace an accepted horizon by any certified prefix while retaining exactly the displayed current-state, input, rollout, first-control, package, and local $\mathsf{NameCert}$ rows at prefix addresses.

Next record the hidden feasibility side. By Definition 348.3, every constraint-window, admissibility, stage-cost, and terminal read used by the prefix packet is accounted for by the displayed ledger rows and the named convex, polytopal, and terminal provenance. Thus the bridge source carries constraint provenance as package data rather than as an ambient host feasible set.

Now export the control boundary. Apply Definition 348.4 to identify the consumer-visible control as the first-control extraction from the displayed optimizer-candidate row. Then apply Definition 348.5 to move this packet through the Cont routes and componentwise hsame rows.

The resulting boundary exposes only the finite MPC public packet: carrier rows, rollout rows, constraint provenance, prefix horizon, package provenance, and the local NameCert row. Since neither cited boundary has a constructor field for host optimizer semantics, infinite-horizon optimality, stochastic control, or unlogged feasibility data, the transported packet is a bridge boundary to the abstract receding-horizon interface and not an exposure of host optimization primitives. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModelPredictiveControl}^{\uparrow}}$.

Constructive story: A $\mathsf{ModelPredictiveControl}^{\uparrow}$ packet is seeded by finite BHist rows for current state, finite prediction horizon, dynamics transition, constraint window, stage-cost ledger, terminal-cost or terminal-region row, optimizer-candidate row, first-control extraction, hsame transports, Cont rollout routes, Pkg provenance, and local NameCert provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModelPredictiveControl}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 348.7, Definition 348.8, and Definition 348.9 bind the finite MPC carrier, prediction ledger, constraint window, first-control boundary, consumer boundary, finite-horizon prefix export, and bridge boundary to displayed BHist rows, hsame transport, Cont rollout routes, Pkg provenance, local NameCert provenance, and the cited control-source interfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModelPredictiveControl}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close infinite-horizon optimality, analytic convergence, stochastic control, host optimizer semantics, recursive feasibility beyond displayed terminal rows, or a standard bridge.

Upgrade path: The remaining work is formal-axis verification of the displayed finite MPC packet.

Chapter 349

A Concrete Naming Certificate for $\mathsf{Trie}^{\uparrow}$

The $\mathsf{Trie}^{\uparrow}$ interface names finite prefix-indexed data without importing host strings, host maps, or a host trie primitive. The reference point is Okasaki, *Purely Functional Data Structures* (1998), where tries are finite prefix trees for keyed lookup; the BEDC surface below reads that structure as finite $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{BitVector}^{\uparrow}$ rows.

The concrete dependency route consumes chapter 32 for branch tags, chapter 36 and chapter 346 for finite key paths, chapter 33 for terminal payload rows, and chapter 28 for depth bounds. When grammar derivation ledgers are indexed by terminal-word prefixes, the prefix subledgers of a $\mathsf{Trie}^{\uparrow}$ packet can be consumed by chapter 356 as finite derivation-prefix lookup rows rather than as a host map or an ambient language index.

Definition 349.1 (Trie BHist source)**.** *A* $\mathsf{Trie}^{\uparrow}$ *source packet is a finite* BHist *record*

$$T = (k, v, l, b, p)$$

with the following visible coordinates:

1. *k is a finite key-path row, read either as a* $\mathsf{BitVector}^{\uparrow}$ *path or as a* $\mathsf{List}^{\uparrow}$ *spine of branch histories;*
2. *v is an* $\mathsf{Option}^{\uparrow}$ *terminal payload row, recording whether the current prefix carries a value and, when present, exposing only the displayed value row;*
3. *l is a* $\mathsf{Nat}^{\uparrow}$ *length and depth ledger recording the finite prefix length, the depth of each inspected node, and the fact that every public read is below that depth;*
4. *b is the branch ledger whose branch tags are* $\mathsf{Bool}^{\uparrow}$ *rows and whose left and right subprefix reads are routed only through displayed finite list or bit-vector positions;*
5. *p is the* Pkg *provenance naming the consumed* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{BitVector}^{\uparrow}$*,* hsame*, and* Cont *certificate rows.*

The carrier predicate is $\mathsf{TrieSource}(k, v, l, b, p)$*. Its classifier compares accepted packets by componentwise key-path comparison, optional payload comparison, branch-tag comparison, length-ledger comparison, and the same displayed* hsame*,* Cont*, and* Pkg *routes. The packet contains no host string equality, host finite map, function extensionality, hidden ordering on keys, or unbounded tree object.*

Theorem 349.2 (Trie carrier stability)**.** *Let $T = (k, v, l, b, p)$ be an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet. If the key path, optional payload, depth ledger, branch ledger, and provenance row are transported by*

displayed hsame *along the* Cont *routes named by* p*, then the transported packet* $T' = (k', v', l', b', p')$ *is again an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket carrier stability`
↪ variant `BEDC.Derived.TrieUp.TriePacket carrier stability`

Proof. Project the five coordinates of Definition 349.1. The $\mathsf{Nat}^{\uparrow}$ ledger transports first, so every later prefix and branch read is bounded by the transported finite depth row.

Transport the key row through its displayed $\mathsf{BitVector}^{\uparrow}$ or $\mathsf{List}^{\uparrow}$ presentation. Each branch tag is a $\mathsf{Bool}^{\uparrow}$ row, so the branch ledger transports componentwise along the same key-path positions.

Finally transport the optional payload and provenance rows. The $\mathsf{Option}^{\uparrow}$ row preserves the absent case and transports the present payload only when the present tag is displayed, while the Pkg row keeps the same finite certificate dependencies. Thus the transported record is another accepted trie source packet. □

Theorem 349.3 (Trie classifier laws)**.** *The* $\mathsf{Trie}^{\uparrow}$ *classifier is reflexive, symmetric, and transitive on accepted packets by componentwise use of the* $\mathsf{BitVector}^{\uparrow}$ *or* $\mathsf{List}^{\uparrow}$ *key-path classifier, the* $\mathsf{Option}^{\uparrow}$ *payload classifier, the* $\mathsf{Bool}^{\uparrow}$ *branch classifier, and the* $\mathsf{Nat}^{\uparrow}$ *ledger classifier.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket classifier laws`

Proof. For reflexivity, take the reflexive classifier row for the key path, the optional payload, each boolean branch tag, and the finite depth ledger. The provenance and continuation rows are read unchanged.

For symmetry, reverse each displayed component comparison. The absent and present cases of the optional payload reverse by the $\mathsf{Option}^{\uparrow}$ classifier, and every branch tag reverses by the $\mathsf{Bool}^{\uparrow}$ classifier.

For transitivity, compose the two key-path comparisons, the two optional payload comparisons, the two branch-ledger comparisons, and the two depth ledger comparisons. The depth ledger fixes the finite set of visible prefix positions, so no hidden branch can enter the composite classifier. □

Theorem 349.4 (Trie ledger coverage)**.** *Every public read of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet factors through the finite ledger* (k, v, l, b, p)*. Consumers see exactly finite prefix rows, boolean branch rows, optional terminal payload rows, depth and length rows,* hsame *transports,* Cont *routes, and* Pkg *provenance.*

Consequently the packet exports no host string type, host map lookup, ambient function space, infinite tree, or implicit key ordering.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket ledger coverage`

Proof. Start from the carrier definition. Its tuple lists exactly the key, terminal payload, depth ledger, branch ledger, and provenance coordinates.

A prefix-facing consumer reads k through the displayed $\mathsf{BitVector}^{\uparrow}$ or $\mathsf{List}^{\uparrow}$ path. A branch-facing consumer reads b only through the boolean tags admitted by the finite depth ledger l. A value-facing consumer reads v only through the $\mathsf{Option}^{\uparrow}$ absent-or-present row.

The remaining public reads are the displayed hsame comparisons, Cont routes, and Pkg provenance named in p. Since no further coordinate is present, every consumer projection is one of these finite rows. □

Theorem 349.5 (Trie prefix-branch exactness)**.** *Every public branch read at a displayed finite prefix of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet is determined by the key path row, the* $\mathsf{Nat}^{\uparrow}$ *depth ledger, the* $\mathsf{Bool}^{\uparrow}$ *branch tag at that prefix, the optional payload row, visible* hsame *transport,* Cont *routes, and* Pkg *provenance from Definition 349.1.*

No host string equality, host map lookup, hidden branch coordinate, or infinite tree can change the read.

Proof. Project the tuple (k, v, l, b, p) from Definition 349.1. The key path row k, optional payload row v, depth ledger l, branch ledger b, and provenance row p are the only coordinates admitted by the accepted packet.

Use Definition 349.4 to bound public branch reads by the finite prefix rows displayed below the depth ledger. A consumer at such a prefix can inspect only the recorded branch tag, the corresponding key-path coordinate, the optional terminal payload row, and the visible transport and provenance rows.

Finally apply Definition 349.3 componentwise. Transport of the key, branch, payload, and depth rows preserves exactly the displayed finite coordinates, so the NameCert surface of Definition 349.10 receives a branch read whose value is fixed by the recorded rows and by no host-side tree structure. □

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket prefix branch exactness`

Theorem 349.6 (Trie prefix branch consumer exhaustion)**.** *Any accepted* $\mathsf{Trie}^{\uparrow}$ *prefix-branch consumer read factors through the displayed finite* BHist *prefix row, the* $\mathsf{Bool}^{\uparrow}$ *branch tag, the* $\mathsf{List}^{\uparrow}$ *path spine, the* $\mathsf{Nat}^{\uparrow}$ *depth counter, the* $\mathsf{Option}^{\uparrow}$ *terminal payload row, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ *ledger.*

The consumer surface is exactly this finite BEDC row surface. It exports no host map, external tree equality, unbounded lookup function, hidden node object, or host key ordering.

Proof. Project the source tuple (k, v, l, b, p) of Definition 349.1. The key coordinate supplies the displayed finite prefix row as a $\mathsf{List}^{\uparrow}$ spine, the branch ledger supplies the $\mathsf{Bool}^{\uparrow}$ branch tag, the depth coordinate supplies the $\mathsf{Nat}^{\uparrow}$ counter, and the terminal coordinate supplies the $\mathsf{Option}^{\uparrow}$ payload row.

Use Definition 349.5 to bind a public prefix-branch read to those coordinates. Its route records the same displayed hsame transports, Cont routes, and Pkg provenance named by the source packet.

Repack the resulting finite read through Definition 349.10. The local $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ ledger accepts exactly the carrier rows, classifier rows, prefix-branch determinacy rows, transport rows, and ledger-exactness rows already displayed in the packet.

Apply Definition 349.4 for the exclusion claim. Since every consumer projection is a row of (k, v, l, b, p) or the named local certificate ledger, no host map, external tree equality, unbounded lookup function, hidden node object, or host key ordering is exported. □

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket prefix branch consumer exhaustion`

Theorem 349.7 (Trie key-path exhaustion)**.** *Every public key-path read of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet is exhausted by the displayed* $\mathsf{BitVector}^{\uparrow}$ *or* $\mathsf{List}^{\uparrow}$ *key row, the* $\mathsf{Nat}^{\uparrow}$ *depth ledger, the* hsame *transports and* Cont *routes named by the packet, and the* Pkg *provenance row. No host string, host ordering, ambient finite map, or hidden key predicate contributes an additional public key-path coordinate.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket key path route exhaustion`

Proof. Project the key coordinate k and depth ledger l from Definition 349.1. The definition gives k only as a displayed $\mathsf{BitVector}^{\uparrow}$ path or as a $\mathsf{List}^{\uparrow}$ spine of branch histories, and it gives l as the bound for every public prefix read.

Use Definition 349.2 to see that transport preserves this same finite key presentation along the displayed hsame and Cont routes. Then apply Definition 349.3 componentwise to the accepted key rows.

Finally apply Definition 349.4. Since every public read is a projection from (k, v, l, b, p), the only remaining key-path data is the provenance named in p, and no host key structure is exported. □

Theorem 349.8 (Trie prefix-branch determinacy)**.** *At each displayed finite prefix below the* $\mathsf{Nat}^{\uparrow}$ *depth row of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet, the* $\mathsf{Bool}^{\uparrow}$ *branch tag selects exactly the recorded left or right subprefix. No unrecorded branch at that prefix is public.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket prefix branch determinacy`

Proof. Project the branch ledger b and the depth ledger l from Definition 349.1. The ledger l bounds the finite prefix positions that a consumer may inspect, and b records branch tags only at those displayed positions.

For any such prefix, read the branch tag through its $\mathsf{Bool}^{\uparrow}$ row. The two displayed boolean cases select the recorded left or right subprefix row, and Definition 349.3 transports equality of branch reads componentwise through the same finite coordinates.

The stability theorem Definition 349.2 preserves the branch ledger under hsame and Cont. Applying Definition 349.4 then excludes any branch coordinate not recorded in (k, v, l, b, p), so no extra branch is public. □

Theorem 349.9 (Trie lookup payload projection)**.** *Every lookup payload read of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet factors through the* $\mathsf{Option}^{\uparrow}$ *absent-or-present terminal row at the displayed key path. The read never factors through a host finite map, host lookup function, or ambient function space.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket lookup payload projection`

Proof. Project the terminal payload coordinate v from Definition 349.1. The coordinate is an $\mathsf{Option}^{\uparrow}$ row at the displayed key path, so a consumer can observe only the absent tag or the present tag together with the displayed payload row.

The key path used by the lookup is fixed by Definition 349.7. The payload row transports along the same displayed hsame and Cont routes by Definition 349.2, and the accepted-row comparison is componentwise by Definition 349.3.

Apply Definition 349.4 to close the projection. Since the finite tuple exports no host map or function coordinate, lookup payload reads can only pass through the displayed $\mathsf{Option}^{\uparrow}$ terminal row. □

Theorem 349.10 (Trie NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the empty trie is carried by the empty key path, an absent* $\mathsf{Option}^{\uparrow}$ *payload, the zero* $\mathsf{Nat}^{\uparrow}$ *depth ledger, the empty branch ledger, and the corresponding* Pkg *provenance;*
2. *classifier laws: reflexivity, symmetry, and transitivity are inherited componentwise from Definition 349.3;*

3. *prefix-branch determinacy: at each displayed finite prefix, a* $\mathsf{Bool}^{\uparrow}$ *branch tag selects exactly the recorded left or right subprefix row, and no unrecorded branch is public, as witnessed by Definition 349.7 and Definition 349.5;*

4. *transport under* hsame *and* Cont*: the stability theorem Definition 349.2 repacks transported key, payload, branch, ledger, and provenance rows as another accepted packet, with lookup payload projection fixed by Definition 349.9;*

5. *ledger exactness: downstream consumers may read only finite prefix rows, branch rows, optional terminal payload rows, length and depth rows, and the displayed provenance rows of Definition 349.4, with the key-path, branch, and payload projections separated by Definition 349.7, Definition 349.8, and Definition 349.9.*

These rows are the complete seed obligation surface for prefix-tree NameCert data over finite BEDC rows.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket namecert obligation surface`

Proof. For carrier habitation, choose the empty $\mathsf{BitVector}^{\uparrow}$ or empty $\mathsf{List}^{\uparrow}$ path, the absent $\mathsf{Option}^{\uparrow}$ payload, the zero $\mathsf{Nat}^{\uparrow}$ depth row, and the empty branch ledger. The provenance row records only the finite data certificates consumed by the packet.

For the classifier laws, apply Definition 349.3. Because the classifier is componentwise, the three equivalence laws are inherited from the key, option, boolean branch, and natural ledger classifiers.

For prefix-branch determinacy, apply Definition 349.7 and Definition 349.8. The key-path theorem fixes the finite prefix positions, and the branch theorem fixes the recorded left or right subprefix at each displayed position.

For transport, apply Definition 349.2. The same hsame and Cont routes move the visible rows without adding host string equality, host maps, or unbounded trees. Lookup payload reads are then projected through Definition 349.9.

For ledger exactness, apply Definition 349.4. Every downstream read is a projection from the finite tuple (k, v, l, b, p), so the NameCert obligation surface is exhausted by the listed rows. □

Definition 349.11 (Trie terminal value packet)**.** *For a finite Boolean-key path, the visible terminal-value row is the* BHist *packet*

$$\mathsf{TrieTerminal}(q, t, r, c, e, g, n)$$

whose coordinates are as follows:

1. *q is the finite key spine, presented as a* $\mathsf{List}^{\uparrow}$ *row of* $\mathsf{Bool}^{\uparrow}$ *branch histories;*

2. *t is the terminal* $\mathsf{Option}^{\uparrow}$ *value row, with the absent case carrying no payload and the present case exposing only the displayed value packet;*

3. *r is the displayed* hsame *transport row for the key spine and the terminal option row;*

4. *c is the* Cont *route taking each accepted prefix row to its extended prefix row and taking the terminal option row to the lookup-facing read;*

5. *e is the* BHist *branch ledger recording the visible left-or-right decision at each position of q;*

6. *g is the* Pkg *provenance naming the consumed* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, hsame, *and* Cont *rows;*

7. *n is the local* NameCert *row certifying that these displayed rows are the public terminal packet for the Boolean-key lookup.*

The packet contains no host map entry, host string key, ambient lookup function, or hidden node object.

Theorem 349.12 (Trie prefix-extension classifier stability)**.** *Let p and p' be displayed Boolean-key prefixes classified by the* $\mathsf{Trie}^{\uparrow}$ *classifier, and let b be a visible* $\mathsf{Bool}^{\uparrow}$ *branch row. If p is extended by b along the* $\mathsf{List}^{\uparrow}$ *key spine and the extension is transported through the displayed* hsame *row and* Cont *route of the packet, then the transported extension of p' is classified under the same branch label b.*

Lean4 ✓ `BEDC.Derived.TrieUp.TriePrefixExtensionClassifier stability`

Proof. Induct on the displayed $\mathsf{List}^{\uparrow}$ key spine. In the empty-spine case, the extension exposes only the first $\mathsf{Bool}^{\uparrow}$ branch row, and classifier reflexivity for that branch row supplies the transported label.

For the cons-spine step, apply the induction hypothesis to the tail prefix. The head branch row is transported by the same hsame route, and Definition 349.3 composes the head comparison with the classified tail comparison.

The Cont route then repacks the extended prefix as a public trie row. Definition 349.2 keeps the transported record accepted, so the resulting extension remains classified under the same visible branch label and no hidden host key coordinate is introduced. □

Theorem 349.13 (Trie Boolean-key path ledger coverage)**.** *Every admitted finite Boolean-key lookup path in a* $\mathsf{Trie}^{\uparrow}$ *packet carries a ledger entry for each displayed branch decision and one terminal* $\mathsf{Option}^{\uparrow}$ *value packet at the final prefix. These entries are precisely the visible* $\mathsf{Bool}^{\uparrow}$ *branch rows, the* $\mathsf{List}^{\uparrow}$ *key-spine rows, the transported* hsame *and* Cont *rows, the* Pkg *provenance, and the terminal packet of Definition 349.11.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieTerminalPacket boolean key path ledger coverage`

Proof. Induct on the displayed $\mathsf{List}^{\uparrow}$ key spine. For the empty spine, the lookup path is already at the terminal prefix, so the ledger contains the single $\mathsf{Option}^{\uparrow}$ terminal packet and the provenance rows named by Definition 349.11.

For a cons spine, the head is a visible $\mathsf{Bool}^{\uparrow}$ branch decision. Record that branch row in the ledger and apply the induction hypothesis to the remaining spine, transporting the tail ledger by the displayed hsame row.

Finally compose the Cont routes along the finite spine. The composed route preserves the branch entries and delivers the terminal $\mathsf{Option}^{\uparrow}$ packet at the final prefix, while Definition 349.4 excludes any additional host lookup, hidden key ordering, or infinite-tree coordinate. □

Theorem 349.14 (Trie branch read exactness)**.** *Let $T = (k, v, l, b, p)$ be an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet. Every public branch read of T factors through a displayed finite prefix position bounded by the* $\mathsf{Nat}^{\uparrow}$ *depth ledger, a* $\mathsf{Bool}^{\uparrow}$ *branch tag, the corresponding* $\mathsf{List}^{\uparrow}$ *or* $\mathsf{BitVector}^{\uparrow}$ *key-path row, the displayed* hsame *and* Cont *route, and the* Pkg *provenance named by p.*

Conversely, every displayed branch row of that form is an accepted public branch read of T. No host string, host finite map, ambient lookup function, or hidden key-order coordinate contributes an additional public branch read.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieBranchRead exactness`
Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket branch read exactness`

Proof. Project the five coordinates of Definition 349.1. The branch ledger b records only boolean branch tags at finite prefix positions, and the depth ledger l bounds the positions visible to public reads.

Apply Definition 349.4 to obtain the available coordinates. The key row is read through the displayed $\mathsf{List}^{\uparrow}$ or $\mathsf{BitVector}^{\uparrow}$ path, and the branch selector is read through its $\mathsf{Bool}^{\uparrow}$ row.

Use Definition 349.3 to transport the key-path, branch, and ledger coordinates componentwise along the named hsame and Cont routes. The provenance coordinate p supplies exactly the consumed certificate rows.

For the converse direction, start from a displayed prefix position, branch tag, key-path row, transport route, and provenance row named in the packet. These are precisely coordinates of (k, v, l, b, p), so they assemble to a public branch read accepted by the carrier.

The exclusions follow from absence of coordinates in Definition 349.1: host strings, host maps, ambient lookup functions, and key-order data cannot be projected from the packet. □

Theorem 349.15 (Trie key-prefix transport)**.** *For accepted* $\mathsf{Trie}^{\uparrow}$ *packets transported along the displayed* hsame *and* Cont *routes, key-path rows preserve finite prefix positions,* $\mathsf{Nat}^{\uparrow}$ *depth bounds, and* $\mathsf{Bool}^{\uparrow}$ *branch selectors. Thus a transported branch read remains on the same displayed finite prefix surface, with only the corresponding* $\mathsf{List}^{\uparrow}$ *or* $\mathsf{BitVector}^{\uparrow}$ *key row, branch tag, ledger row, and* Pkg *provenance transported.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket key prefix transport`

Proof. Begin with Definition 349.2. Transporting the accepted packet repacks the key, payload, branch, ledger, and provenance coordinates as another accepted $\mathsf{Trie}^{\uparrow}$ packet.

The component laws in Definition 349.3 transport the key row through its displayed $\mathsf{List}^{\uparrow}$ or $\mathsf{BitVector}^{\uparrow}$ presentation and transport each branch selector through its $\mathsf{Bool}^{\uparrow}$ classifier row. The $\mathsf{Nat}^{\uparrow}$ ledger travels componentwise with those same finite positions.

Finally, return to the source packet of Definition 349.1. The only key-path coordinates present there are the displayed key row, finite depth bound, branch selector, named hsame and Cont route, and Pkg provenance. Hence transport preserves exactly those prefix coordinates and no additional key structure. □

Theorem 349.16 (Trie lookup ledger exhaustion)**.** *Every lookup-facing consumer of an accepted* $\mathsf{Trie}^{\uparrow}$ *packet is exhausted by the displayed key path,* $\mathsf{Option}^{\uparrow}$ *terminal payload,* $\mathsf{Bool}^{\uparrow}$ *branch rows,* $\mathsf{Nat}^{\uparrow}$ *depth ledger,* hsame *rows,* Cont *routes, and* Pkg *provenance already present in the packet. The packet exports no host finite-map lookup, string equality, function extensionality, infinite tree, or key ordering.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieBHistSource lookup ledger exhaustion`
Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket lookup ledger exhaustion`

Proof. Use Definition 349.14 for branch-facing reads. Each such read is a displayed finite prefix position, boolean selector, key-path row, transport route, and provenance row from the packet.

Use Definition 349.15 for transported key-facing reads. It preserves the finite prefix positions, ledger bounds, and branch selectors under the displayed hsame and Cont routes.

The remaining lookup payload coordinate is the $\mathsf{Option}^{\uparrow}$ row already recorded in Definition 349.1. Apply Definition 349.4 and Definition 349.10 to collect the full packet surface.

Since the collected surface is exactly (k, v, l, b, p), every lookup-facing consumer projection is one of the displayed rows. Host finite maps, string equality, function extensionality, infinite trees, and key orderings are not coordinates of that packet, so they are not exported. □

Theorem 349.17 (Trie lookup prefix ledger factorization)**.** *Let $T = (k, v, l, b, p)$ be an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet and let q be a finite key prefix displayed by the* $\mathsf{List}^{\uparrow}$ *or* $\mathsf{BitVector}^{\uparrow}$ *key-path row. A lookup read at q factors through exactly one of two visible sources: the optional terminal payload row at q, or a branch ledger row extending q below the unary depth ledger. An absent terminal payload row alone is not evidence for absence below q unless the branch ledger below q is also exhausted.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket lookup prefix ledger factorization`

Proof. Project the source packet of Definition 349.1. The prefix q is read from the displayed key-path coordinate, while the optional terminal payload and branch rows are separate coordinates of the same packet.

Use Definition 349.14 to identify every branch read below q with a displayed finite prefix position, boolean selector, key-path row, transport route, and provenance row. Then use Definition 349.16 to see that no other lookup-facing source is available.

Split the optional payload row at q. In the present case, the lookup read uses the terminal $\mathsf{Option}^{\uparrow}$ payload projection. In the absent case, the absence of that terminal coordinate only removes the terminal source at q.

It remains to account for reads below the prefix. Those reads must pass through branch ledger rows extending q; only when that branch ledger is also exhausted does the packet provide finite evidence for absence below the prefix. No host map lookup, string equality, or ambient trie semantics is imported. □

Theorem 349.18 (Trie finite branch consumer boundary)**.** *Let $T = (k, v, l, b, p)$ be a concrete* $\mathsf{Trie}^{\uparrow}$ BHist *source packet from Definition 349.1, with displayed key path k, optional terminal payload v, unary depth ledger l, boolean branch rows b, and* Pkg *provenance p over* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{BitVector}^{\uparrow}$ *certificates. Every finite branch consumer read of T factors through exactly one of two visible projections: either the terminal payload projection of Definition 349.9, or the branch route determined by Definition 349.8.*

These two projections exhaust the public consumer surface. No host string, host finite map, infinite tree, ambient lookup function, or key-order semantics is introduced by a finite branch read.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket finite branch consumer boundary`

Proof. Unfold the source packet of Definition 349.1. Its public coordinates are the key path, optional payload, depth ledger, branch ledger, and provenance rows named in the theorem.

Split on the displayed optional terminal branch. In the absent case, there is no terminal payload coordinate to read, so the consumer must remain on the finite branch route. In the present case, the only terminal read is the $\mathsf{Option}^{\uparrow}$ payload row, and Definition 349.9 supplies that projection.

For the branch route, use Definition 349.7 to cover the finite key positions. At each covered prefix, Definition 349.8 fixes the unique displayed $\mathsf{Bool}^{\uparrow}$ branch row selected by the packet.

Finally apply Definition 349.4 to account for provenance. The ledger contains only the displayed $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{BitVector}^{\uparrow}$, hsame, Cont, and Pkg rows, so no host string, host map, infinite-tree coordinate, ambient lookup function, or key-order semantics can be projected by the consumer. □

Lean4 ✓ `BEDC.Derived.TrieUp.TrieFiniteBranchConsumer boundary`

Theorem 349.19 (Trie public NameCert export)**.** *The source packet of Definition 349.1, the obligation surface of Definition 349.10, the branch read exactness of Definition 349.14, the lookup ledger exhaustion of Definition 349.16, the prefix ledger factorization of Definition 349.17, and the finite branch consumer boundary of Definition 349.18 assemble the public* $\mathsf{NameCert}_{\mathsf{Trie}^\uparrow}$ *surface for the finite key path, optional payload, unary depth ledger, branch rows, displayed* hsame *and* Cont *routes,* Pkg *provenance, and* $\mathsf{Bool}^\uparrow$*,* $\mathsf{List}^\uparrow$*,* $\mathsf{Option}^\uparrow$*,* $\mathsf{Nat}^\uparrow$*, and* $\mathsf{BitVector}^\uparrow$ *certificates.*

This public export contains no host strings, host finite maps, infinite trees, key ordering, or ambient lookup function.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket public namecert export`

Proof. Project the scoped packet rows from Definition 349.1. They are exactly the finite key path, optional terminal payload, unary depth ledger, branch ledger, transport routes, continuation routes, and provenance rows.

Apply Definition 349.10 to package these rows as the local $\mathsf{NameCert}_{\mathsf{Trie}^\uparrow}$ obligation surface. The branch rows are identified by Definition 349.14, and the lookup-facing payload and ledger reads are exhausted by Definition 349.16.

The prefix boundary is Definition 349.17: a missing terminal row at a prefix is separated from exhaustion of branch rows below that prefix, so the public export records finite failed-lookup evidence only when both visible sources have been accounted for.

Finally use Definition 349.18. It states that every finite branch consumer read factors through the displayed terminal payload or branch route, so the assembled packet is a public consumer surface over precisely the named rows.

The excluded objects are not coordinates of the source packet and are not introduced by any of the cited projections. Hence host strings, host finite maps, infinite trees, key ordering, and ambient lookup functions are absent from the public $\mathsf{NameCert}_{\mathsf{Trie}^\uparrow}$ export. □

Theorem 349.20 (Trie public export route decomposition)**.** *The public* $\mathsf{NameCert}_{\mathsf{Trie}^\uparrow}$ *export decomposes every finite consumer route into exactly one of two displayed routes: the optional terminal payload projection or the Boolean branch route below the unary depth ledger. In both routes the key path, payload row when present, branch row, depth row, displayed* hsame *transport,* Cont *routes, and* Pkg *provenance are preserved as rows of the same finite packet.*

No host string, host finite map, infinite tree, ambient lookup function, or key-order semantics is introduced by this decomposition.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket public export route decomposition`

Proof. Project the public packet through Definition 349.19. This exposes the finite key path, optional payload, unary depth ledger, branch rows, displayed transport, continuation routes, and package provenance.

For the terminal route, apply Definition 349.9. If the terminal payload is present, the consumer read is exactly the displayed $\mathsf{Option}^\uparrow$ payload row at the selected key path. If it is absent, there is no terminal payload coordinate to read.

For the branch route, apply Definition 349.14. It identifies every branch-facing read below the unary depth ledger with the corresponding displayed $\mathsf{Bool}^\uparrow$ branch row.

Finally use Definition 349.16. The ledger has no third consumer route beyond the optional terminal payload projection and the finite branch route, and all named key, payload, branch, depth, transport, continuation, and provenance rows remain inside the same public $\mathsf{NameCert}_{\mathsf{Trie}^\uparrow}$ export. □

Theorem 349.21 (Trie standard finite packet bridge)**.** *The finite* $\mathsf{Trie}^{\uparrow}$ *packet of Definition 349.1 and the public export of Definition 349.19 bridge to the standard finite prefix-tree reading. The visible key path, optional payload, unary depth ledger,* $\mathsf{Bool}^{\uparrow}$ *branch rows,* BHist *packet rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ *rows are exactly the public bridge surface.*

The bridge exports no host string, host finite map, hidden infinite tree, key ordering, or ambient lookup function.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket standard finite packet bridge`

Proof. Project the finite packet as (k, v, l, b, p) from Definition 349.1. The coordinates are the displayed key path, optional terminal payload, unary depth ledger, branch rows, and provenance rows, with the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surfaces already named by the source and public export.

Use Definition 349.7 to identify the standard prefix-tree path with the displayed finite key row. Use Definition 349.8 to identify each finite branch decision with its recorded $\mathsf{Bool}^{\uparrow}$ row below the depth ledger.

Apply Definition 349.9 for the optional terminal payload. Then apply Definition 349.19 to package the path, payload, branch, transport, continuation, provenance, and local naming rows as the public $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ surface.

Finally, exhaust the remaining reads through the unary ledger. Every standard finite prefix-tree read is below the displayed depth row and therefore lands in one of the projected rows. Since the packet has no coordinate for host strings, host maps, hidden infinite trees, key orderings, or ambient lookup functions, none is exported by the bridge. □

Theorem 349.22 (Trie terminal-prefix readback coverage)**.** *Every terminal read of a finite* $\mathsf{Trie}^{\uparrow}$ *packet factors through the displayed finite key prefix, optional terminal payload row,* $\mathsf{Bool}^{\uparrow}$ *branch row, unary depth ledger,* hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ *row.*

Conversely, any finite read assembled from exactly those rows repacks as a terminal-prefix read of the same finite packet. The readback surface contains no host string, host map, infinite tree, key-order semantics, ambient lookup function, or balancing invariant.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieTerminalPacket prefix readback coverage`
Lean4 ✓ `BEDC.Derived.TrieUp.TerminalPrefixReadback.TrieSourcePacket terminal prefix readback coverage`

Proof. For a prefix-facing read, apply Definition 349.6. It keeps the consumer inside the displayed key prefix, unary depth ledger, branch rows, transport routes, continuation routes, provenance, and local naming evidence.

For a terminal key-path read, apply Definition 349.13. The terminal payload is read at the displayed prefix endpoint, and every branch decision is one of the recorded $\mathsf{Bool}^{\uparrow}$ rows below the unary depth ledger.

Use Definition 349.14 to identify the branch route used by the terminal read. Then apply Definition 349.21 to repack the same rows as a finite prefix-tree reading.

The converse direction reverses this packaging: a read built from the key prefix, optional payload, branch, depth, transport, continuation, provenance, and local certificate rows is exactly a terminal-prefix read of the displayed finite packet. Since no cited row names host strings, host maps, infinite trees, key-order semantics, lookup functions, or balancing invariants, none is available to the readback. □

Lean4 ✓ `BEDC.Derived.TrieUp.TrieTerminalPacket terminal prefix readback coverage`

Theorem 349.23 (Trie finite consumer completeness)**.** *Every finite consumer of the* $\mathsf{Trie}^{\uparrow}$ *bridge factors through the key-path row, the optional terminal payload row, the unary length and depth ledger, the* $\mathsf{Bool}^{\uparrow}$ *branch rows, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and the* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{BitVector}^{\uparrow}$ *certificates already named by Definition 349.19 and Definition 349.21.*

Conversely, any finite read assembled from exactly those rows repacks as a valid finite trie consumer read of the bridged packet. The completeness surface exports no host strings, host finite maps, infinite trees, key-order semantics, ambient lookup function, or balancing invariant.

Lean4 ✓ `BEDC.Derived.TrieUp.TrieSourcePacket finite consumer completeness`

Proof. Project the source packet of Definition 349.1. Its carrier coordinates are the key path, optional terminal payload, unary ledger, branch rows, and provenance row, together with the displayed hsame, Cont, Pkg, and local certificate data.

For the forward direction, read branch-facing coordinates through Definition 349.14. Transported key prefixes are carried by Definition 349.15, while lookup-facing rows are exhausted by Definition 349.16. The boundary theorem Definition 349.18 then leaves only the terminal payload projection or the displayed branch route.

Apply Definition 349.19 to collect these coordinates as the public $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$ surface. Then apply Definition 349.21 to identify the same surface with the standard finite prefix-tree reading.

For the converse direction, start with a finite read using only the listed rows. The public export supplies the local certificate package, and the standard bridge repacks the key path, payload, ledger, branch, transport, continuation, provenance, and certificate rows into the accepted finite trie consumer surface.

No cited theorem introduces a coordinate outside the source packet. Hence host strings, host finite maps, infinite trees, key-order semantics, ambient lookup functions, and balancing invariants remain outside the exported consumer interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$.

Constructive story: A trie packet is generated by a finite key-path row, an optional terminal payload row, a unary length and depth ledger, boolean branch rows, displayed transport routes, and package provenance over Bool, List, Option, Nat, and BitVector certificates.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 349.1, Definition 349.10, Definition 349.14, Definition 349.16, Definition 349.17, Definition 349.18, Definition 349.19, Definition 349.21, and Definition 349.23 assemble the public finite trie NameCert export, bridge it to the standard finite prefix-tree reading, and exhaust finite consumer reads over the key path, optional payload, unary depth ledger, branch rows, displayed transport and continuation routes, package provenance, and Bool, List, Option, Nat, and BitVector certificates.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Trie}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.TrieUp.TriePrefixSubledger\_provenance\_exhaustion`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host strings, host finite maps, function extensionality, infinite trees, key-order semantics, balancing invariants, or host lookup behavior.

Upgrade path: No further paper-side theory grade remains for the finite trie packet surface; formal bridge verification remains on the separate verification axis.

Definition 349.24 (Displayed Trie prefix subledger)**.** *Let* $T = (k, v, l, b, p)$ *be an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet. A displayed prefix subledger of* T *is a finite packet*

$$U = (k_U, v_U, l_U, b_U, p_U; \iota)$$

with a displayed inclusion row ι into the visible prefix positions of T such that:

1. *k_U is a finite* $\mathsf{BitVector}^{\uparrow}$ *path or* $\mathsf{List}^{\uparrow}$ *spine whose entries are exactly the key-path entries of k at the positions selected by ι;*
2. *the selected positions are prefix-closed: whenever a displayed position is selected, every displayed predecessor prefix of that position in the key-path ledger is also selected;*
3. *v_U is the* $\mathsf{Option}^{\uparrow}$ *payload row of T restricted to selected terminal positions, so the absent tag stays absent and a present tag exposes only the same displayed payload row already exposed by v;*
4. *l_U is the* $\mathsf{Nat}^{\uparrow}$ *depth ledger obtained by restricting l to the selected positions, with each selected read still below the corresponding depth bound;*
5. *b_U is the branch ledger obtained by retaining exactly the* $\mathsf{Bool}^{\uparrow}$ *branch tags and left or right subprefix reads of b whose source and target prefixes are selected by ι;*
6. *p_U is the projected* Pkg *provenance row naming only the consumed* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{BitVector}^{\uparrow}$, hsame, *and* Cont *rows that occur in the retained coordinates.*

The associated restricted packet is $T|_U = (k_U, v_U, l_U, b_U, p_U)$.

Theorem 349.25 (Trie prefix subledger restriction carrier)**.** *Let $T = (k, v, l, b, p)$ be an accepted* $\mathsf{Trie}^{\uparrow}$ *source packet, and let U be a displayed prefix subledger of T in the sense of Definition 349.24. Then the restricted packet*

$$T|_U = (k_U, v_U, l_U, b_U, p_U)$$

is again an accepted $\mathsf{Trie}^{\uparrow}$ *source packet under the same* $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{BitVector}^{\uparrow}$ *dependency certificates.*

Lean4 ✓ `BEDC.Derived.TrieUp.TrieDisplayedPrefixSubledger restriction carrier`

Proof. Unpack the accepted source packet by Definition 349.1. The original carrier consists only of the key-path row, optional terminal payload row, finite depth ledger, boolean branch ledger, and package provenance row. The displayed subledger definition selects rows from exactly those five coordinates and records the inclusion row ι.

First consider the key coordinate. Since k_U is obtained by selecting displayed positions of the finite $\mathsf{BitVector}^{\uparrow}$ path or $\mathsf{List}^{\uparrow}$ branch-history spine of k, it is again a finite key-path row of the same kind. Prefix-closedness of the selected positions ensures that every selected non-root prefix has its predecessor prefix in the same restricted packet, so no branch read in k_U points outside the restricted carrier.

Next consider the $\mathsf{Nat}^{\uparrow}$ ledger. The row l_U is the restriction of l to selected positions. Every public read admitted by l_U was already a public read admitted by l, and the depth inequality for that read is the same displayed $\mathsf{Nat}^{\uparrow}$ row. Thus the restricted depth ledger is finite and still bounds all selected key and branch reads.

For the branch coordinate, each retained entry of b_U is a $\mathsf{Bool}^{\uparrow}$ tag already present in b. The definition of displayed prefix subledger keeps a branch entry only when its source prefix and the selected left or right subprefix row are both selected by ι. Therefore the branch ledger of the restricted packet has no public unrecorded branch, and every visible branch selection remains routed through displayed finite list or bit-vector positions.

For the optional payload coordinate, the absent case is inherited unchanged from v. In the present case, v_U exposes only a payload row already exposed by the original $\mathsf{Option}^{\uparrow}$ row v at a

selected terminal prefix. Thus no payload comparison, hidden value equality, or terminal read is introduced by the restriction.

Finally, p_U is the projection of the original provenance row p to the dependency rows actually used by k_U, v_U, l_U, and b_U. It names the same kinds of certificates as p: $\mathsf{Bool}^\uparrow$, $\mathsf{List}^\uparrow$, $\mathsf{Option}^\uparrow$, $\mathsf{Nat}^\uparrow$, $\mathsf{BitVector}^\uparrow$, hsame, and Cont. Since Definition 349.4 says every public read of an accepted trie packet factors through the finite ledger coordinates, deleting unselected rows cannot create a host string equality, a host finite map lookup, an ambient function space, an infinite tree, or an implicit ordering on keys.

The five projected coordinates therefore satisfy the same carrier clauses as Definition 349.1. Hence $T|_U = (k_U, v_U, l_U, b_U, p_U)$ is an accepted $\mathsf{Trie}^\uparrow$ source packet under the same finite dependency certificates. □

Theorem 349.26 (Trie prefix subledger provenance exhaustion)**.** *Let $T = (k, v, l, b, p)$ be an accepted $\mathsf{Trie}^\uparrow$ source packet, and let U be a displayed prefix subledger of T. Every public read of the restricted packet*

$$T|_U = (k_U, v_U, l_U, b_U, p_U)$$

factors through the restricted key, value, depth, branch, and provenance rows k_U, v_U, l_U, b_U, p_U. In particular the restricted read can use only the projected $\mathsf{Bool}^\uparrow$, $\mathsf{List}^\uparrow$, $\mathsf{Option}^\uparrow$, $\mathsf{Nat}^\uparrow$, $\mathsf{BitVector}^\uparrow$, hsame, Cont, and Pkg coordinates named by p_U; it cannot introduce a host finite map, hidden string equality, unlisted ordering on keys, ambient function space, or infinite-tree datum.

Lean4 ✓ `BEDC.Derived.TrieUp.TriePrefixSubledger provenance exhaustion`

Proof. Start with the displayed prefix subledger of Definition 349.24. Its inclusion row ι selects prefix-closed positions from the original finite key-path ledger, and its restricted rows k_U, v_U, l_U, b_U, p_U are projections of the original five carrier coordinates.

Apply Definition 349.25. The restricted tuple is again an accepted $\mathsf{Trie}^\uparrow$ source packet under the same finite dependency certificates, so its public reads are ordinary Trie reads rather than host-side lookups.

Finally apply Definition 349.4 to $T|_U$. Ledger coverage exhausts public access by the finite rows of the accepted packet. Since those rows are precisely the restricted rows supplied by the displayed subledger, the only available provenance is the projected row p_U, and none of the excluded host or infinite coordinates can enter the restricted read. □

Lean4 ✓ `BEDC.Derived.TrieUp.TrieDisplayedPrefixSubledger public read exhaustion`
↪ variant `BEDC.Derived.TrieUp.TrieDisplayedPrefixSubledger provenance exhaustion`

Theorem 349.27 (Trie List prefix sibling route)**.** *Let $T = (k, v, l, b, p)$ be an accepted $\mathsf{Trie}^\uparrow$ source packet, and let U be a displayed prefix subledger of T. Every read of $T|_U$ as Trie data factors first through the $\mathsf{List}^\uparrow$ spine row of chapter 36, the $\mathsf{Option}^\uparrow$ payload row of chapter 33, the $\mathsf{Bool}^\uparrow$ branch tags, the $\mathsf{Nat}^\uparrow$ depth ledger, componentwise hsame transport, displayed Cont replay, and the projected Pkg provenance row before the local $\mathsf{Trie}^\uparrow$ carrier may expose the restricted prefix.*

Proof. Project the displayed prefix subledger of Definition 349.24. The selected key coordinate k_U is a finite $\mathsf{BitVector}^\uparrow$ path or $\mathsf{List}^\uparrow$ spine, the payload coordinate v_U is the restricted $\mathsf{Option}^\uparrow$ row, the branch coordinate b_U retains only displayed $\mathsf{Bool}^\uparrow$ tags, and the depth coordinate l_U is the projected $\mathsf{Nat}^\uparrow$ ledger.

The restriction theorem Definition 349.25 repacks exactly those projected coordinates as an accepted $\mathsf{Trie}^\uparrow$ source packet. Its proof already requires the branch source and target prefixes to

remain selected by the inclusion row, so the displayed list spine and optional payload row remain upstream of every Trie read.

Finally apply Definition 349.26. Public reads of the restricted packet are exhausted by k_U, v_U, l_U, b_U, p_U, and p_U names only the projected finite certificates, hsame transport, and Cont replay. Thus no restricted Trie read can bypass the List and Option sibling rows or treat a host finite map, string equality, ordering, ambient function space, or infinite tree as Trie data. □

Chapter 350

A Concrete Naming Certificate for $\mathsf{RationalStream}^{\uparrow}$

The $\mathsf{RationalStream}^{\uparrow}$ interface names finite packets for pointwise rational stream observations on the path from $\mathsf{Rat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$ and the $\mathsf{Real}^{\uparrow}$ seal. Following Bishop-style constructive analysis of rational Cauchy streams, the packet keeps the observation schedule, rational point rows, classifier rows, and provenance visible without forming quotient Cauchy classes or selecting a completed real limit.

Concretely, the packet is the visible finite-window route from chapter 37 through chapter 126 into chapter 301 and finally the seal surface of chapter 38.

Definition 350.1 (RationalStream carrier)**.** *A* $\mathsf{RationalStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (u, s, q, \rho, h, c, p, n)$$

with the following visible rows:

1. *u is the* $\mathsf{UnaryHistory}$ *index row naming the certified finite observation positions available to the packet;*
2. *s is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting which rational observation is read at each displayed unary index;*
3. *q is the family of pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows read along the schedule s;*
4. *ρ is the family of pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows comparing scheduled rational observations at the same displayed indices;*
5. *h records the componentwise* hsame *transport rows for indices, schedules, rational carriers, and rational classifiers;*
6. *c records the* Cont *routes through which finite index windows and their scheduled rational reads are exposed to consumers;*
7. *p is the visible* Pkg *provenance row naming the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *dependency certificates used by the packet;*
8. *n is the local* $\mathsf{NameCert}_{\mathsf{RationalStream}^{\uparrow}}$ *registration row.*

The carrier predicate is $\mathsf{RationalStreamPacket}(u, s, q, \rho, h, c, p, n)$*. Its classifier compares two accepted packets only by transporting the displayed schedule and applying the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows on the certified unary indices.*

Theorem 350.2 (RationalStream pointwise RatUp obligations)**.** *Every accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet exposes its rational observations only through the pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier family* q *and classifier family* ρ *indexed by the displayed* UnaryHistory *row* u*. Reflexive, symmetric, transitive, and transported comparisons of rational observations are therefore obligations over the stored* $\mathsf{Rat}^{\uparrow}$ *rows, not over a quotient or host rational stream object.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket pointwise rat obligations`

Proof. Project the carrier of Definition 350.1. The only coordinates that contain rational point data are q and ρ, and both are indexed through the visible unary row u and schedule row s.

The transport coordinate h moves the displayed point rows by hsame, while c records the continuation routes used to read finite windows. Repacking these projections gives the four pointwise comparison obligations entirely on the stored $\mathsf{Rat}^{\uparrow}$ rows. □

Theorem 350.3 (RationalStream StreamName schedule boundary)**.** *The* $\mathsf{StreamName}^{\uparrow}$ *schedule boundary of an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet is exactly the row* s *together with the unary index row* u*, the continuation routes* c*, and the transported schedule witnesses in* h*. A consumer may ask which finite rational observation is read at a displayed index, but the packet supplies no unbounded stream object outside those scheduled finite reads.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket streamname schedule boundary`

Proof. Read the schedule coordinate s from the carrier. Its domain of visible reads is the unary index row u, and each consumer route is one of the continuation rows stored in c.

If the same schedule is transported, the corresponding witness is one of the hsame rows in h. No other coordinate stores a stream selector, so the schedule boundary is exhausted by these rows. □

Theorem 350.4 (RationalStream RegSeqRat handoff)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet may read only the finite stream-name schedule* s*, unary index row* u*, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier rows* q *and* ρ*, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The handoff records a regularity request surface for rational finite windows; it does not assert Cauchy completeness, selected limits, or equality of completed real values.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket regseqrat handoff transport`

Proof. Start with the accepted packet and project the coordinates read by a regular sequence consumer. The schedule s chooses the finite observations, u names their indices, and the rows q and ρ provide the rational carrier and classifier data at those indices.

The provenance coordinates h, c, p, and n record transport, continuation, dependency, and registration information. Repacking exactly these rows gives the $\mathsf{RegSeqRat}^{\uparrow}$ handoff surface, with no coordinate left for completion or a selected limit. □

Theorem 350.5 (RationalStream NameCert seed obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RationalStream}^{\uparrow}}$ *seed surface consists of five rows:*

1. *carrier: the finite packet of Definition 350.1;*
2. *classifier: pointwise* $\mathsf{Rat}^{\uparrow}$ *comparison over the scheduled unary indices;*
3. *stability:* hsame *transport of the unary index row, schedule row, point carriers, and point classifiers;*

4. *ledger: the finite windows, continuation routes, package provenance, and local naming registration visible in c, p, and n;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: the regular rational-stream request surface of Definition 350.4.*

These five rows form the seed obligation sketch for rational stream packets on the current $\mathsf{Real}^{\uparrow}$ *and completion path.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket namecert seed obligations`

Proof. Carrier and classifier are obtained by projecting the packet coordinates named in Definition 350.1. The pointwise obligation theorem identifies the rational comparison rows, and the schedule-boundary theorem identifies the finite observation schedule.

Stability is the projection of the displayed hsame coordinate h. The ledger is the projection of c, p, and n, and the regular-sequence row is the handoff supplied by Definition 350.4. These projections account for every carrier coordinate, so the five seed obligations are complete for the displayed packet. □

Theorem 350.6 (RationalStream common-window classifier stability)**.** *If two accepted* $\mathsf{RationalStream}^{\uparrow}$ *packets classify together, then their comparison factors through a displayed common finite unary window. On that window the two* $\mathsf{UnaryHistory}$ *rows are related by the stored* hsame *witnesses, the* $\mathsf{StreamName}^{\uparrow}$ *schedules select the same displayed finite observation positions, and every rational comparison is one of the pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows ρ transported along the packet data. The stability claim exposes only finite windows and stored rational rows; it does not identify quotient Cauchy classes, choose a limit, or assert equality inside completed* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the two carrier tuples from Definition 350.1. The only classifier coordinates are the pointwise rational rows ρ, and the only index transport coordinates are the displayed hsame rows in h.

Use Definition 350.2 to reduce every rational comparison to the stored $\mathsf{Rat}^{\uparrow}$ carrier and classifier rows. Use Definition 350.3 to identify the finite schedule rows and the common unary window on which the comparison is read.

The remaining coordinates c, p, and n record consumer routes, dependency provenance, and local registration. None supplies a quotient Cauchy object, a selected limiting real, or completed $\mathsf{Real}^{\uparrow}$ equality, so the common-window classifier stability is exhausted by the displayed finite packet rows. □

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket common window classifier stability`

Theorem 350.7 (RationalStream common-denominator window exhaustion)**.** *Any finite consumer read comparing scheduled rational observations from two accepted* $\mathsf{RationalStream}^{\uparrow}$ *packets can be refined to one displayed unary window and one displayed common-denominator* $\mathsf{Rat}^{\uparrow}$ *comparison ledger over the pointwise carrier and classifier rows. The refinement uses only the stored* $\mathsf{UnaryHistory}$ *indices, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules, the pointwise rational carrier rows q, the pointwise classifier rows ρ, and the* hsame *and* Cont *transport routes already present in the packets. It introduces no quotient Cauchy classes, selected limits, or completed* $\mathsf{Real}^{\uparrow}$ *equality.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket common denominator window exhaustion`

Proof. Start with the consumer read and project the two accepted carrier packets of Definition 350.1. The StreamName schedule boundary Definition 350.3 identifies the finite unary indices on which the compared observations are read.

Apply the common-window classifier stability theorem Definition 350.6. It refines the two schedule reads to a single displayed finite unary window and transports the pointwise rows along the stored hsame witnesses.

On that window, use the pointwise RatUp obligations of Definition 350.2. Each scheduled comparison is a rational comparison between displayed $\mathsf{Rat}^{\uparrow}$ rows, so the finite list of denominators read by the consumer can be represented by one common-denominator comparison ledger inside the same pointwise carrier and classifier surface.

The ledger is routed through the displayed Cont rows and repacked with the same package and local naming data. Since every row used in the construction is already a finite packet coordinate, the refinement has no coordinate from which a quotient Cauchy class, selected limit, or completed $\mathsf{Real}^{\uparrow}$ equality could be obtained. □

Theorem 350.8 (RationalStream schedule transport exactness)**.** *Transporting an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet by the stored* hsame *rows transports exactly its* UnaryHistory *index,* $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalStream}^{\uparrow}}$ *row. The transported packet is again accepted with the same finite schedule surface, and the transport introduces no unbounded stream, quotient Cauchy class, selected limit, or completed* $\mathsf{Real}^{\uparrow}$ *equality.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamSchedule transport exactness`
↪ variant `BEDC.Derived.RationalStreamUp.RationalStreamPacket schedule transport exactness`

Proof. Start from the carrier tuple $(u, s, q, \rho, h, c, p, n)$ in Definition 350.1. The transport coordinate h is componentwise, so it names exactly the allowed movements of the unary index row, schedule row, rational carriers, and rational classifiers.

The schedule boundary theorem Definition 350.3 identifies the finite stream-name surface after transport. The pointwise obligation theorem Definition 350.2 keeps the rational rows inside the transported $\mathsf{Rat}^{\uparrow}$ data, and Definition 350.4 records that c, p, and n are the only consumer, provenance, and registration rows exposed to the regular-sequence reader.

Repacking these transported coordinates gives another accepted packet with the same finite support. Since every coordinate has been accounted for, no coordinate remains from which an unbounded stream, quotient class, chosen limit, or completed real equality could be introduced. □

Theorem 350.9 (RationalStream RegSeqRat ledger completeness)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read from an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet is exhausted by the scheduled unary indices, the pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier and classifier rows,* hsame *and* Cont *transport data,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RationalStream}^{\uparrow}}$ *ledger. The ledger therefore records a complete finite rational-stream handoff surface and excludes quotient Cauchy classes, selected limiting reals, and completed* $\mathsf{Real}^{\uparrow}$ *equality.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket regseqrat finite window surface`

Proof. Project the accepted packet carrier. Its scheduled consumer-visible rows are u and s for finite indices and stream-name choices, together with q and ρ for the pointwise rational data.

Apply Definition 350.4. It states that a regular-sequence consumer may read only those scheduled rational rows and the displayed hsame, Cont, Pkg, and NameCert provenance.

The pointwise obligation theorem Definition 350.2 and the schedule boundary theorem Definition 350.3 identify the rational and schedule parts of that handoff. The remaining coordinates are exactly h, c, p, and n, so the consumer ledger is complete for the packet and leaves no row for quotient completion or a selected real limit. □

Theorem 350.10 (RationalStream finite-window carrier transport)**.** *Let*

$$R = (u, s, q, \rho, h, c, p, n)$$

be an accepted $\mathsf{RationalStream}^{\uparrow}$ *packet. Transport the displayed* $\mathsf{UnaryHistory}$ *index row,* $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows,* Cont *windows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalStream}^{\uparrow}}$ *row by the componentwise* hsame *rows stored in* h*. The transported rows again form an accepted* $\mathsf{RationalStreamPacket}$*, and the accepted carrier exposes no coordinate beyond that finite transported window.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket finite window carrier transport`

Proof. Project the carrier in Definition 350.1. The coordinate h lists exactly the hsame rows available for the unary index, schedule, point carriers, and point classifiers, while c, p, and n are the continuation, package, and naming rows transported with them.

Apply Definition 350.3 to u, s, c, and h to keep the finite stream-name schedule fixed at the transported window. Apply Definition 350.2 to q and ρ so every rational comparison remains a transported $\mathsf{Rat}^{\uparrow}$ row.

Repack the transported p and n rows with these schedule and pointwise rational data. This reconstructs the accepted packet, and the projection accounts for every component of R. □

Theorem 350.11 (RationalStream pointwise classifier laws)**.** *The* $\mathsf{RationalStream}^{\uparrow}$ *classifier is reflexive, symmetric, and transitive on accepted packets whenever those three laws are read pointwise from the stored* $\mathsf{Rat}^{\uparrow}$ *classifier rows at every displayed unary index. These laws are stable under the* hsame *transports in the packet, with the* $\mathsf{StreamName}^{\uparrow}$ *schedule fixed by the displayed schedule boundary.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket pointwise classifier laws`

Proof. Use Definition 350.2 on each displayed unary index. Reflexivity, symmetry, and transitivity of packet classification are then the corresponding pointwise $\mathsf{Rat}^{\uparrow}$ classifier obligations for the rows q and ρ.

The classifier comparison is read only along the schedule boundary from Definition 350.3. Transport through hsame changes the displayed rows but not the finite schedule surface, so the three laws remain pointwise and no quotient stream equality is introduced. □

Theorem 350.12 (RationalStream RegSeqRat obligation package)**.** *The carrier of Definition 350.1, the pointwise classifier laws of Definition 350.11, the finite-window transport of Definition 350.10, the schedule boundary of Definition 350.3, the finite ledger of Definition 350.9, and the handoff of Definition 350.4 form the complete* $\mathsf{RationalStream}^{\uparrow}$ *obligation package consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*. The package supplies no quotient stream, selected limit, or completed real equality.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket regseqrat obligation package`

Proof. Assemble the carrier and seed rows from Definition 350.5. The schedule boundary and pointwise rational obligations identify the finite stream-name and $\mathsf{Rat}^{\uparrow}$ surfaces, while the classifier-law theorem supplies the three classifier laws over those surfaces.

Use the finite-window transport theorem for stability under the stored hsame rows. Use Definition 350.9 for the finite consumer ledger and Definition 350.4 for the regular-sequence handoff consumed downstream.

These inputs exhaust the rows of the displayed packet. Since none of the rows contains a quotient stream object, a selected limit, or a completed $\mathsf{Real}^{\uparrow}$ equality proof, the package records exactly the obligation surface needed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 350.13 (RationalStream scoped boundary)**.** *The carrier of Definition 350.1, pointwise* $\mathsf{Rat}^{\uparrow}$ *obligations of Definition 350.2,* $\mathsf{StreamName}^{\uparrow}$ *schedule boundary of Definition 350.3,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 350.4, classifier stability of Definition 350.11, exact schedule transport of Definition 350.10, and ledger completeness of Definition 350.9 assemble a scoped finite rational-stream packet.*

This packet is over BHist*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*. It exports no quotient Cauchy class, selected limit,* $\mathsf{Real}^{\uparrow}$ *completeness row, trichotomy, or bridge.*

Proof. Project the carrier and pointwise rational rows from Definition 350.1 and Definition 350.2. These projections give the finite unary index rows, displayed rational data, and classifier coordinates.

Next use the StreamName schedule boundary and exact carrier transport. The theorems Definition 350.3 and Definition 350.10 keep the finite schedule and transported rows inside the displayed UnaryHistory, hsame, and Cont surface.

Finally apply Definition 350.4, Definition 350.11, and Definition 350.9. They identify the regular-sequence handoff, classifier stability, and complete consumer ledger over the displayed package and naming rows.

Repacking these rows gives the scoped finite boundary. Since the cited packet coordinates contain no quotient Cauchy class, selected limit, $\mathsf{Real}^{\uparrow}$ completeness row, trichotomy, or bridge, none is exported by the handoff surface. □

Theorem 350.14 (RationalStream public export)**.** *The carrier of Definition 350.1, pointwise* $\mathsf{Rat}^{\uparrow}$ *obligations,* $\mathsf{StreamName}^{\uparrow}$ *schedule boundary,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, common-window classifier stability, schedule transport exactness, ledger completeness, and scoped boundary form the public* $\mathsf{RationalStream}^{\uparrow}$ *export consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

This public export is a finite BEDC packet over BHist*,* UnaryHistory*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It exports no quotient Cauchy class, selected limit, completed* $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, or standard bridge.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket public export`

Proof. Begin with the carrier and pointwise rational rows. Apply Definition 350.1 and Definition 350.2 to project the visible BHist, UnaryHistory, and $\mathsf{Rat}^{\uparrow}$ coordinates.

Next build the finite schedule surface. Use Definition 350.3 for the $\mathsf{StreamName}^{\uparrow}$ boundary and Definition 350.10 for exact transport of the displayed hsame and Cont rows.

Now attach the downstream handoff rows. Apply Definition 350.11, Definition 350.9, and Definition 350.4 to obtain classifier stability, ledger completeness, and the $\mathsf{RegSeqRat}^{\uparrow}$ consumer surface.

Finally repack the result through Definition 350.13. The package and naming rows are exactly the displayed Pkg and NameCert provenance. Since the scoped boundary has no coordinate for a quotient Cauchy class, selected limit, completed $\mathsf{Real}^{\uparrow}$ equality, trichotomy, or standard bridge, the public export exposes none of those rows. □

Theorem 350.15 (RationalStream RegSeqRat RealUp consumer coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet factors through exactly the finite unary*

index row, the StreamName$^{\uparrow}$ *schedule, the pointwise* Rat$^{\uparrow}$ *carrier and classifier rows, the displayed* hsame *rows, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *row already present in Definition 350.1.*

The coverage surface exposes no quotient Cauchy class, selected limit, host stream equality, completed Real$^{\uparrow}$ *equality, or hidden completion coordinate.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket regseqrat realup consumer coverage`

Proof. Start from the public packet of Definition 350.14. Its exported rows are the finite BEDC packet over BHist, UnaryHistory, hsame, Cont, Pkg, and NameCert with the pointwise rational data and downstream-facing schedule already displayed.

Next apply Definition 350.10. This keeps every transported consumer read on the finite unary index row and the displayed hsame and Cont routes, so no host stream equality is introduced when the consumer selects a window.

Use Definition 350.11 to identify each endpoint read with the visible Rat$^{\uparrow}$ carrier and classifier rows. Then use Definition 350.9 to exhaust the regular-sequence ledger entries available to downstream consumers.

Finally apply Definition 350.4. The handoff repackages exactly the displayed finite index, schedule, rational endpoint, transport, route, package, and naming rows. Since none of the cited surfaces has a coordinate for quotient Cauchy classes, selected limits, host stream equality, completed Real$^{\uparrow}$ equality, or hidden completion coordinates, no RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ consumer can factor through such data. □

Theorem 350.16 (RationalStream standard bridge)**.** *For a carried* RationalStream$^{\uparrow}$ *packet over unary* StreamName$^{\uparrow}$ *observations and* Rat$^{\uparrow}$ *endpoint classifiers, the public export of Definition 350.14, finite-window carrier transport of Definition 350.10, pointwise classifier laws of Definition 350.11, and the* RegSeqRat$^{\uparrow}$ *obligation package of Definition 350.12 determine a conservative standard reading as a finite-window rational sequence prefix.*

Reading that finite standard prefix back into the BEDC packet recovers the same BHist *window, denominator rows, endpoint classifier rows,* StreamName$^{\uparrow}$ *schedule rows, and* RegSeqRat$^{\uparrow}$ *ledger rows. The bridge adds no quotient sequence, selected real limit, host equality of streams, or hidden completion coordinate.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamUp StdBridge`

Proof. Project the public packet using Definition 350.14. This exposes only finite BHist windows, unary schedule rows, rational endpoint data, transport rows, package provenance, and naming evidence.

Use Definition 350.10 to align the chosen standard prefix with the carried finite window. The same displayed hsame and Cont rows transport the window and schedule, so the readback has the original StreamName$^{\uparrow}$ observation rows.

Apply Definition 350.11 at each exposed endpoint. The numerator, denominator, and endpoint classifier rows are therefore the same rows after the standard-prefix readback.

Finally use Definition 350.12 for the downstream handoff. It supplies the RegSeqRat$^{\uparrow}$ ledger rows consumed by the finite prefix and records that all hidden data remains in the displayed ledger, which gives conservativity and excludes quotient sequences, selected limits, host stream equality, and hidden completion coordinates. □

Theorem 350.17 (RationalStream standard bridge common-window readback)**.** *For an accepted* RationalStream$^{\uparrow}$ *packet over unary* StreamName$^{\uparrow}$ *observations and* Rat$^{\uparrow}$ *endpoint classifiers, the*

standard bridge readback of Definition 350.16 factors through one displayed finite unary common window. The rows of that window are exactly the $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger rows, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

The readback exports no quotient Cauchy class, selected limit, completed $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, or host stream equality. Hence the same finite common window is the readback surface consumed by both* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket standard bridge common window readback`

Proof. First put the endpoint classifiers on a single finite comparison surface. Apply Definition 350.6 to align the carried pointwise classifier rows, and then apply Definition 350.7 to read the denominator comparisons inside that same displayed unary window.

Next transport the carrier rows without changing the readback surface. Use Definition 350.10 to keep the schedule, pointwise rational carriers, pointwise classifiers, hsame rows, Cont routes, package provenance, and local naming rows on the chosen window.

Then exhaust the regular-sequence consumer ledger with Definition 350.9. This identifies the $\mathsf{RegSeqRat}^{\uparrow}$ rows consumed downstream with the same schedule, classifier, carrier, transport, route, package, and certificate rows already displayed by the finite window.

Finally apply Definition 350.16. The standard bridge reads back only the finite rational sequence prefix, so the readback has no coordinate for quotient Cauchy classes, selected limits, completed $\mathsf{Real}^{\uparrow}$ equality, trichotomy, or host stream equality.

The Phase Real and Completion fit is the following: traditional target, finite rational stream readback; BEDC source, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; carrier, the displayed $\mathsf{RationalStream}^{\uparrow}$ packet; classifier, the common finite-window classifier; stability, the hsame and Cont transport; ledger, the finite window rows; core theorem, Definition 350.17; exactness, readback equality of rows; standard bridge, finite rational sequence prefix; status, proof target. □

Theorem 350.18 (RationalStream RealUp finite-window consumer example)**.** *For a carried* $\mathsf{RationalStream}^{\uparrow}$ *packet over unary* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers, the public export of Definition 350.14, finite-window carrier transport of Definition 350.10, pointwise classifier laws of Definition 350.11, the* $\mathsf{RegSeqRat}^{\uparrow}$ *obligation package of Definition 350.12, and the standard bridge of Definition 350.16 determine a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer packet.*

The visible rows of this consumer are exactly BHist *windows,* UnaryHistory *schedules,* hsame *transport,* Cont *routes,* Pkg *provenance,* NameCert *evidence,* $\mathsf{Rat}^{\uparrow}$ *endpoint classifiers,* $\mathsf{StreamName}^{\uparrow}$ *schedules, and* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger rows. It exports no quotient Cauchy class, selected limit,* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, or machine-checked bridge.*

Proof. Project the carried packet through Definition 350.14. This exposes the finite BHist window, unary observation schedule, pointwise rational endpoint rows, package provenance, and local naming evidence.

Replay the finite window with Definition 350.10. The displayed hsame rows and Cont routes identify the transported finite-window carrier without adding any host stream equality.

Use Definition 350.11 endpointwise. Each visible $\mathsf{Rat}^{\uparrow}$ classifier row is preserved by the finite transport, so the consumer reads the same rational endpoint data as the source packet.

Consume the regular-sequence rows through Definition 350.12. The resulting ledger is the complete $\mathsf{RegSeqRat}^{\uparrow}$ surface available to the downstream finite-window consumer.

Finally close by Definition 350.16. The bridge reads only the finite rational prefix and returns exactly the displayed history, schedule, transport, route, package, naming, classifier, and ledger

rows. Since none of the cited packets has a coordinate for quotient Cauchy classes, selected limits, completed $\mathsf{Real}^{\uparrow}$ equality, trichotomy, or machine-checked bridge evidence, the consumer exports none of those data. □

Theorem 350.19 (RationalStream RealUp seal consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of the* $\mathsf{RationalStream}^{\uparrow}$ *standard bridge of Definition 350.16 may read only the finite unary schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, common-window classifier, finite-window transport, package provenance, and local* $\mathsf{NameCert}$ *row already named by Definition 350.16. It cannot read quotient Cauchy classes, a selected limit, completed* $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, or host-level standard real data.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamSealConsumer boundary`

Proof. Project the carrier from Definition 350.1. This exposes only finite unary index rows, pointwise rational endpoint data, denominator rows, and the displayed package and naming coordinates carried by the rational stream packet.

Apply Definition 350.14 and Definition 350.16. The public export gives the $\mathsf{RegSeqRat}^{\uparrow}$- and $\mathsf{Real}^{\uparrow}$-facing packet, while the standard bridge reads it as a finite-window rational sequence prefix with the same visible schedule, classifier, transport, package, and naming rows.

Use Definition 350.9 to exhaust the regular-sequence handoff rows. The handoff ledger accounts for the downstream consumer reads by finite prefix, rational endpoint, schedule, and provenance coordinates.

Finally apply Definition 350.13. Since that boundary has no coordinate for quotient Cauchy classes, selected limits, completed $\mathsf{Real}^{\uparrow}$ equality, trichotomy, or host-level standard real data, a $\mathsf{Real}^{\uparrow}$-facing consumer of the standard bridge cannot project any such datum. □

Theorem 350.20 (RationalStream RealUp seal common-window normal form)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of a* $\mathsf{RationalStream}^{\uparrow}$ *standard bridge factors through one displayed common finite unary window. The rows of that window are exactly the* $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger rows, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *evidence.*

The normal form exports no quotient Cauchy class, selected limit, completed $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, or host standard stream equality. Any* $\mathsf{Real}^{\uparrow}$ *seal consumer therefore reads the same finite common window consumed by the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *packet surfaces.*

Lean4 ✓ `BEDC.Derived.RationalStreamUp.RationalStreamPacket real seal common window normal form`

Proof. Project the carried rational stream packet through Definition 350.1. The carrier projection exposes only the finite unary index, the stream schedule, pointwise rational rows, visible transport and route data, package provenance, and local naming evidence.

Use Definition 350.6 to align the pointwise rational classifiers with a single common window. Then apply Definition 350.7 to put the denominator rows read by the endpoint classifiers on that same displayed finite window.

Next apply Definition 350.9. This exhausts the $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows available to any downstream consumer and keeps them attached to the same schedule, classifier, transport, route, package, and certificate rows.

Finally combine Definition 350.16 with Definition 350.19. The standard bridge gives the finite rational-prefix reading, and the seal boundary rules out every read outside the displayed rows. Repacking those rows gives the claimed common finite-window normal form and excludes quotient

Cauchy classes, selected limits, completed $\mathsf{Real}^{\uparrow}$ equality, trichotomy, and host standard stream equality. □

Theorem 350.21 (RationalStream finite-window seal determinacy)**.** *For accepted* $\mathsf{RationalStream}^{\uparrow}$ *packets, agreement of the displayed* UnaryHistory *index row,* $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *carrier rows, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, common-window ledger, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row determines exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the* $\mathsf{Real}^{\uparrow}$ *seal consumer reads.*

Conversely, every $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read and every* $\mathsf{Real}^{\uparrow}$ *seal consumer read of an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet is one of those finite rows. No quotient Cauchy class, selected limit, completed* $\mathsf{Real}^{\uparrow}$ *equality, trichotomy row, or host standard stream equality is determined or exported by the finite-window seal.*

Proof. Project the accepted packet through Definition 350.1. This exposes the finite unary index row, schedule row, pointwise rational carrier and classifier rows, denominator data, displayed transport and route rows, package provenance, and local naming evidence.

Use the schedule boundary and the pointwise $\mathsf{Rat}^{\uparrow}$ obligations to identify which displayed rows may be read at each finite index. Then apply Definition 350.6 to place the classifier rows on the same common window.

Apply Definition 350.7. The denominator rows needed by the pointwise rational classifiers are exhausted by that common finite window, so no further denominator coordinate remains for a downstream consumer.

Next use Definition 350.9. It turns the displayed schedule, pointwise rational rows, classifier rows, transports, routes, package provenance, and naming row into exactly the $\mathsf{RegSeqRat}^{\uparrow}$ handoff surface.

For the $\mathsf{Real}^{\uparrow}$ seal direction, invoke Definition 350.19 and Definition 350.20. The seal consumer boundary admits only finite packet rows, and the normal form places all such reads on the same common window.

The converse follows by reading the same projections in the opposite order: any $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal read must pass through the carrier projection, the common-window classifier stability, the denominator exhaustion step, and the seal normal form. Hence each consumer coordinate is one of the listed finite rows. □

Theorem 350.22 (RationalStream synchronized seal readback exhaustion)**.** *Any synchronized finite-window read of an accepted* $\mathsf{RationalStream}^{\uparrow}$ *packet factors through exactly its displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule, pointwise* $\mathsf{Rat}^{\uparrow}$ *rows, common-window classifier rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal consumer boundary, displayed* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows.*

The read exports no quotient stream equality, selected limit, completed $\mathsf{Real}^{\uparrow}$ *equality, trichotomy row, or host stream equality beyond that finite seal boundary.*

Proof. Use Definition 350.6 and Definition 350.7 to place the classifier and denominator reads on one displayed finite window.

Then apply Definition 350.9, Definition 350.19, and Definition 350.21. These theorems identify the handoff, seal consumer read, and exclusion boundary with the same finite packet rows. □

Theorem 350.23 (RationalStream tail-window determinacy)**.** *Two* $\mathsf{RationalStream}^{\uparrow}$ *packets with the same finite unary schedule, common-denominator window, pointwise* $\mathsf{Rat}^{\uparrow}$ *classifier rows, and*

hsame-*transported tail selector determine the same finite* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *seal consumer package on that window.*

The conclusion is only the finite tail-window package: it exports no quotient Cauchy class, selected limit, completed Real$^{\uparrow}$ *equality, trichotomy row, or host standard stream equality.*

Proof. Apply Definition 350.6 to place the classifier readings on the shared finite window. Then use Definition 350.7 for the common denominator rows.

Transport the tail selector with Definition 350.10. The transported carrier exposes the same schedule, classifier, hsame, Cont, Pkg, and local NameCert rows to every finite consumer.

Finally apply Definition 350.20 and Definition 350.21. They repack the shared tail-window read as one RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ package. □

Closure status for NameCert$_{\text{RationalStream}^{\uparrow}}$.

Constructive story: A rational stream packet is represented by finite unary index rows, a StreamName schedule, pointwise RatUp carrier and classifier rows, and visible transport, continuation, package, and NameCert provenance on the path to RegSeqRat and the RealUp seal.

Theory closure: TheoryGate(NameCert$_{\text{RationalStream}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 350.13, Definition 350.16, and Definition 350.19 bind the finite carrier, pointwise classifier laws, finite-window transport, schedule boundary, finite ledger, regular-sequence handoff, standard bridge, RealUp seal consumer boundary, and displayed BHist, UnaryHistory, hsame, Cont, Pkg, and NameCert dependencies consumed by RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RationalStream}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RationalStreamUp.RationalStreamSchedule\_transport\_exactness`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient Cauchy classes, selected limits, RealUp completeness, trichotomy, or a machine-checked standard bridge.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 351

A Concrete Naming Certificate for $\mathsf{RegularLanguage}^{\uparrow}$

The $\mathsf{RegularLanguage}^{\uparrow}$ interface names finite automaton recognition as a displayed BEDC packet rather than as a host quotient of languages. Following the finite-automaton presentations of Hopcroft–Ullman (1979) and Sipser (2012), the packet records a finite alphabet, finite state row, transition ledger, input word, deterministic run, and accepting endpoint entirely through finite BHist, hsame, Cont, and Pkg rows.

The packet consumes the $\mathsf{List}^{\uparrow}$ word-spine carrier through chapter 36, the $\mathsf{FinSet}^{\uparrow}$ alphabet and state rows through chapter 112, and the $\mathsf{Computable}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ recognizer boundary through chapter 199 and chapter 280. These adjacent certificates supply the displayed finite rows consumed by the automaton packet; they do not import host language-set semantics or an external recognizer quotient.

This places $\mathsf{RegularLanguage}^{\uparrow}$ as the finite-automaton sibling between chapter 356 and chapter 280. Context-free grammar consumers read the regular-language comparison through terminal-yield rows, and $\mathsf{TuringMachine}^{\uparrow}$ consumers read the same accepted-word ledger as a finite recognizer boundary.

Definition 351.1 (RegularLanguage finite-automaton carrier)**.** *A* $\mathsf{RegularLanguage}^{\uparrow}$ *packet is a finite record*

$$R = (\Sigma, Q, q_0, F, \delta, w, r, a, h, c, p)$$

with the following displayed rows:

1. Σ *is a finite alphabet row licensed by* $\mathsf{FinSet}^{\uparrow}$*;*
2. Q *is a finite state row licensed by* $\mathsf{FinSet}^{\uparrow}$*;*
3. q_0 *is the start-state row carried by* Q*;*
4. F *is the accepting-state carrier inside* Q*;*
5. δ *is the transition table row whose entries read a state and an alphabet symbol and return the next displayed state;*
6. w *is the input word as a* $\mathsf{List}^{\uparrow}$ *history row over* Σ*;*
7. r *is the deterministic run ledger over* BHist*, beginning at* q_0 *and consuming the word spine one symbol at a time;*
8. a *is the accepted endpoint row, consisting of the final run state and its membership in* F*;*

9. *h stores the componentwise* hsame *transport rows for alphabet, state, transition, word, run, and accepting endpoint data;*
10. *c stores the* Cont *routes for consuming the displayed word spine and reading each transition step;*
11. *p stores the* Pkg *provenance naming* List$^{\uparrow}$, FinSet$^{\uparrow}$, Computable$^{\uparrow}$, *and* TuringMachine$^{\uparrow}$.

The carrier predicate is RegularLanguageAutomatonPacket(R). *It contains only the displayed finite automaton data and the accepted input row; it exports no host language set, infinite automaton, quotient recognizer, or selected external semantics.*

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageTasteGate single carrier alignment`

Definition 351.2 (RegularLanguage accepted-word classifier)**.** *Two accepted* RegularLanguage$^{\uparrow}$ *packets classify together exactly when their alphabet rows, state rows, transition tables, input words, deterministic run ledgers, and accepting endpoints compare by the displayed* hsame *rows in the packet. The comparison is componentwise:*

$$\Sigma \sim_{\mathsf{hsame}} \Sigma', \quad Q \sim_{\mathsf{hsame}} Q', \quad q_0 \sim_{\mathsf{hsame}} q_0', \quad F \sim_{\mathsf{hsame}} F',$$
$$\delta \sim_{\mathsf{hsame}} \delta', \quad w \sim_{\mathsf{hsame}} w', \quad r \sim_{\mathsf{hsame}} r', \quad a \sim_{\mathsf{hsame}} a'.$$

Each displayed comparison is transported along the stored Cont *routes and is checked against the same* Pkg *provenance. No classifier clause compares unaccepted words or identifies languages extensionally outside the packet.*

Lean4 (definition) `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonClassifier`

Theorem 351.3 (RegularLanguage deterministic run ledger)**.** *For every accepted* RegularLanguage$^{\uparrow}$ *packet, the run ledger is determined by the start-state row, the transition table row, and the displayed* List$^{\uparrow}$ *input spine. At the empty word spine, the run endpoint is the start-state row. At a cons spine, the next run row is obtained by reading the head symbol, applying the unique transition-table entry to the current state, and continuing through the tail spine.*

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket run ledger deterministic`
Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket deterministic run ledger`

Proof. Induct on the displayed List$^{\uparrow}$ word spine. In the empty case, the carrier row requires the initial run row to be q_0, and no transition route is available to change it.

In the cons case, project the current state row and the head alphabet symbol from the carrier. The transition table is finite and deterministic, so the stored table entry supplies one next-state row. The continuation route then hands the tail spine to the induction hypothesis.

The ledger records no additional branch. Therefore every displayed consumer read of the run endpoint is reconstructed by these start, transition, and spine rows. □

Theorem 351.4 (RegularLanguage run-prefix restriction)**.** *For an accepted* RegularLanguage$^{\uparrow}$ *packet and any displayed finite prefix of its* List$^{\uparrow}$ *input word spine, the restricted run ledger is determined by the same start-state row, the transition table rows along the prefix, the prefix word rows, componentwise* hsame *rows,* Cont *routes, and* Pkg *provenance. The endpoint of the restricted run transports to the corresponding endpoint of the full accepted run along the displayed classifier rows.*

This restriction introduces no host language quotient, external regular expression equivalence, minimization datum, infinite automaton, or hidden automaton state.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket run prefix restriction`

Proof. Induct on the displayed prefix. In the empty-prefix case, the restricted run ledger is the start-state row, which is the empty-spine case of Definition 351.3.

For a cons prefix, read the head symbol and the current state from the packet. The deterministic transition table supplies the unique next displayed state, and the Cont route passes the remaining prefix to the induction hypothesis. Thus the restricted ledger is determined only by the prefix word rows and the transition rows exposed along that prefix.

Apply Definition 351.3 to the full word spine and to the displayed prefix restriction. The two ledgers agree on every state reached inside the prefix because they use the same start row and the same transition-table entries.

Finally use Definition 351.5 for the transported endpoint rows. The endpoint comparison is componentwise through hsame, Cont, and Pkg, and no external language quotient, minimization, infinite automaton, or hidden state is available in the packet. □

Theorem 351.5 (RegularLanguage classified word transport)**.** *If two accepted* $\mathsf{RegularLanguage}^{\uparrow}$ *packets classify together, then their accepted-word endpoints transport together: the final state of the first run is carried to the final state of the second run, and acceptance in the first accepting-state carrier is carried to acceptance in the second accepting-state carrier through the displayed* hsame *rows.*

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket classified word transport`

Proof. Use the classifier definition to obtain componentwise comparisons of the alphabet row, state row, transition table, input word, run ledger, and accepting endpoint. The word comparison fixes the same displayed spine shape under $\mathsf{List}^{\uparrow}$ transport.

Apply Definition 351.3 to both packets. At each spine step, the transported transition-table entry sends transported states and transported symbols to transported next states. Thus the final run states compare by the displayed state hsame row.

Finally project the accepting endpoint row. Since the accepting-state carrier is transported componentwise with the state row, the accepted endpoint of the first packet maps to the accepted endpoint of the second packet. □

Theorem 351.6 (RegularLanguage NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation for displayed finite automata with a finite alphabet, finite state carrier, start state, accepting-state carrier, transition table, input word, and accepted deterministic run;*
2. *classifier equivalence from componentwise* hsame *comparison of the alphabet, state, transition, word, run, and accepting endpoint rows;*
3. *deterministic run reconstruction along the displayed* $\mathsf{List}^{\uparrow}$ *input spine by Definition 351.3;*
4. *accepted-word endpoint exactness by Definition 351.5;*
5. *consumer boundary: every public read factors through the displayed* BHist, hsame, Cont, *and* Pkg *rows, with no hidden automaton state, host language quotient, or infinite recognition semantics.*

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket namecert obligation surface`

Proof. Carrier habitation is the finite automaton packet of Definition 351.1. Its rows are finite alphabet and state carriers, a selected displayed start state, a displayed accepting-state carrier, a finite transition table, and a finite input spine.

The classifier row is Definition 351.2. Reflexive, symmetric, and transitive comparison are inherited componentwise from the displayed hsame rows and from the finite table and list-spine transports.

The ledger and exactness rows are supplied by Definition 351.3 and Definition 351.5. They reconstruct the run from the start, transition table, and word spine, then identify the accepted endpoint under classifier transport.

The consumer boundary is obtained by projecting the packet coordinates. Since the carrier has no coordinate for a host language set, quotient recognizer, or infinite automaton, downstream chapters can read only the displayed finite certificate rows. □

Theorem 351.7 (RegularLanguage scoped kernel dependency envelope)**.** *For* $\mathsf{RegularLanguage}^{\uparrow}$*, the finite-automaton carrier, accepted-word classifier, deterministic run ledger, classified-word transport, and local* NameCert *obligation surface are all read from finite* BHist *word rows,* hsame *transport of accepted endpoints,* Cont*-composed run routes,* Pkg *provenance, and the existing* $\mathsf{List}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*, and* $\mathsf{TuringMachine}^{\uparrow}$ *certificate rows.*

This scoped envelope exports no host regular expression, host-language set extensionality, minimization uniqueness claim, infinite automaton, or external recognizer quotient.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket scoped kernel dependency envelope`

Proof. Read the carrier from Definition 351.1. Its alphabet, state, transition, word, run, accepting endpoint, transport, continuation, and provenance coordinates are finite displayed rows, with the named upstream certificates appearing only through Pkg provenance.

Use Definition 351.2 to bind classifier reads to componentwise hsame transport. Then apply Definition 351.3 and Definition 351.5; together they attach the accepted endpoint to the displayed word spine and to the transported final state.

Finally repack the five-field local certificate surface with Definition 351.6. Every field is one of the displayed finite packet rows or one of the named upstream certificate rows, so the excluded host-language and infinite-automaton objects have no coordinate in the scoped envelope. □

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket scoped dependency reassociation witness`

Theorem 351.8 (RegularLanguage public accepted-word export)**.** *The finite-automaton carrier, accepted-word classifier, deterministic run ledger, classified-word transport, and* $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$ *obligation surface assemble a public* $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$ *export whose readable rows are exactly the finite alphabet row, finite state row, start row, accepting-state row, transition table,* $\mathsf{List}^{\uparrow}$ *word spine, deterministic run ledger, accepted endpoint, componentwise* hsame *rows,* Cont *routes, and* Pkg *provenance.*

The export contains no host language quotient, external regular-expression equivalence, minimization result, infinite automaton, selected recognizer semantics, or hidden automaton state.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket public accepted word export`

Proof. Read the finite alphabet, finite state row, start state, accepting-state carrier, transition table, and input word spine from Definition 351.1.

Use Definition 351.2 to bind public classifier reads to the componentwise hsame rows. Then apply Definition 351.3 to obtain the displayed deterministic run ledger over the $\mathsf{List}^{\uparrow}$ spine.

Next apply Definition 351.5. It identifies the accepted endpoint as the transported endpoint of the same finite run, with the stated Cont routes and package provenance.

Finally use Definition 351.6 to repack these rows as the public accepted-word $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$ export. Since none of the carrier or obligation rows is a host language quotient, regular-expression equivalence, minimization result, infinite automaton, selected recognizer semantics, or hidden automaton state, the public packet exports no such object. □

Theorem 351.9 (RegularLanguage transition-ledger standard boundary)**.** *The standard finite-automaton reading of a public* $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$ *packet may consume exactly the finite alphabet row, finite state row, start row, accepting-state row, transition table,* $\mathsf{List}^{\uparrow}$ *word spine, deterministic run ledger, accepted endpoint, displayed* hsame *transport rows,* Cont *routes, and* Pkg *provenance.*

It cannot read a host language quotient, regular-expression equivalence, minimization result, infinite automaton, or hidden recognizer state.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket transition ledger standard boundary`

Proof. Read the carrier rows from Definition 351.1. These are precisely the finite alphabet, finite state, start, accepting-state, transition-table, word-spine, deterministic-run, and accepted-endpoint rows available to the standard finite-automaton reading.

Use Definition 351.2 to bind classifier reads to the displayed hsame comparisons. Then apply Definition 351.3; the run is reconstructed from the displayed start row, transition table, and $\mathsf{List}^{\uparrow}$ word spine.

Next apply Definition 351.5. It carries the accepted endpoint through the displayed transport rows and does not add a host-language coordinate.

Finally apply Definition 351.8. The public export repacks the carrier, classifier, run-ledger, endpoint, continuation, and package rows as the visible certificate surface, while the excluded quotient, equivalence, minimization, infinite-automaton, and hidden recognizer objects are not rows of that surface. □

Theorem 351.10 (RegularLanguage standard finite-automaton bridge boundary)**.** *The standard finite-automaton reading of the public* $\mathsf{RegularLanguage}^{\uparrow}$ *packet consumes exactly the finite state row, start row, accepting row, transition ledger, input-word* $\mathsf{List}^{\uparrow}$ *rows, deterministic run ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and* NameCert *surface already exported by Definition 351.8.*

The bridge boundary supplies no minimization theorem, Kleene algebra surface, pumping-lemma witness, language-complement closure, or machine-checked bridge evidence.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket standard automaton bridge boundary`

Proof. Read the automaton rows from Definition 351.1. The finite state row, start row, accepting row, transition table, and input-word $\mathsf{List}^{\uparrow}$ spine are all displayed fields of the carrier.

Use Definition 351.2 to bind accepted-word reads to the componentwise hsame transport, then apply Definition 351.3. This supplies the finite deterministic run ledger over exactly the displayed transition and word rows.

Finally apply Definition 351.8. It repackages those rows with the Cont routes, Pkg provenance, and NameCert surface, while the excluded minimization, algebraic, pumping, complement, and machine-checking objects have no coordinate in the public packet. □

Theorem 351.11 (RegularLanguage finite bridge consumer completeness)**.** *Every finite consumer of the standard finite-automaton bridge for* RegularLanguage$^{\uparrow}$ *factors through the displayed alphabet row, finite state row, start row, accepting-state row, transition table,* List$^{\uparrow}$ *word spine, deterministic run ledger, endpoint acceptance row,* hsame *transport rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Conversely, those rows repack into the same bridge boundary: no host language quotient, regular-expression equivalence, minimization result, infinite automaton, hidden recognizer state, or bridge-external semantic coordinate is needed to reconstruct the finite consumer surface.

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket finite bridge consumer completeness`

Proof. First read the finite carrier from Definition 351.1. This provides the alphabet row, finite state row, start row, accepting-state row, transition table, and List$^{\uparrow}$ word spine consumed by the bridge.

Use Definition 351.2 to identify the accepted-word classifier with the displayed hsame rows. Then apply Definition 351.3; the deterministic run ledger is reconstructed from the start row, transition table, and word spine.

Next apply Definition 351.5 and Definition 351.8. These results carry the endpoint acceptance through the displayed transport, Cont routes, Pkg provenance, and local NameCert surface.

For the converse direction, apply Definition 351.10. It repacks exactly the same displayed rows as the standard finite-automaton bridge boundary. Since the listed exclusions are not rows of that boundary, neither factorization nor repacking can export them. □

Theorem 351.12 (RegularLanguage displayed automaton consumer examples)**.** *An accepted* RegularLanguage$^{\uparrow}$ *packet supplies three finite downstream consumers for the displayed standard finite-automaton bridge: a* List$^{\uparrow}$ *word-spine reader, a* Computable$^{\uparrow}$ *step-bounded transition reader, and a* TuringMachine$^{\uparrow}$ *finite run-trace reader. Each consumer may read only the finite alphabet row, finite state row, transition table, word spine, deterministic run ledger, endpoint acceptance row, displayed* BHist, hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

These consumers do not read regular-expression quotient equivalence, minimization, infinite automata, hidden recognizer semantics, or host language equality.

Proof. Expose the accepted carrier rows with Definition 351.1. The alphabet, state, transition table, word spine, deterministic ledger, endpoint row, and displayed BHist provenance are the finite data available to all three consumers.

Use Definition 351.2 to bind the endpoint classifier to the displayed hsame and Cont rows. Then apply Definition 351.3; this gives the List$^{\uparrow}$ word-spine reader and the step-bounded transition reader without leaving the finite table.

Next apply Definition 351.5. The same run ledger transports the endpoint acceptance row, so the finite run-trace reader can consume the ledger through the displayed Pkg and local NameCert rows.

Finally apply Definition 351.8 and Definition 351.10. They repack the word-spine, transition, and run-trace readers as conservative finite downstream reads of the standard bridge. The excluded quotient, minimization, infinite-automaton, hidden-semantics, and host-language objects are not rows of that bridge boundary. □

Theorem 351.13 (RegularLanguage prefix bridge consumer boundary)**.** *Every finite prefix consumer of a public* $\mathsf{RegularLanguage}^{\uparrow}$ *packet factors through the displayed prefix word spine, the restricted deterministic run ledger, endpoint transport,* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *rows exported by the public finite-automaton bridge.*

Such a consumer cannot read minimization data, host language equality, regular-expression quotient equivalence, infinite automata, or hidden recognizer state.

Proof. First restrict the displayed input to the finite prefix under inspection. By Definition 351.4, the consumed prefix is read through the same $\mathsf{List}^{\uparrow}$ word spine and the corresponding restricted deterministic run ledger.

Next use Definition 351.9 to identify which rows a standard finite-automaton reader may consume: the transition ledger, endpoint row, displayed hsame transports, Cont routes, Pkg provenance, and local NameCert rows.

Apply Definition 351.10 to repack those rows as the public finite-automaton bridge surface. Then Definition 351.12 shows that the word-spine, step-bounded transition, and finite run-trace consumers are all conservative reads of that same displayed surface.

The excluded objects are not coordinates of the restricted run ledger or of the public bridge surface. Hence a finite prefix consumer has no route to minimization, host language equality, regular-expression quotient equivalence, infinite automata, or hidden recognizer state. □

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket prefix bridge consumer boundary`

Theorem 351.14 (RegularLanguage empty-word acceptance exactness)**.** *Let* $R = (\Sigma, Q, q_0, F, \delta, w, r, a, h, c, p)$ *be a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet satisfying* $\mathsf{RegularLanguageAutomatonPacket}(R)$*. If the displayed* $\mathsf{List}^{\uparrow}$ *input spine* w *is empty, then the accepted endpoint row* a *is exactly the start-state row together with accepting-carrier membership:*

$$a = (q_0, \eta)$$

for the displayed witness $\eta : q_0 \in F$*. In particular, the state coordinate of the accepted endpoint is* q_0*, and the membership coordinate is membership of that same start-state row in the accepting-state carrier* F*.*

Lean4 ✓ `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket empty word acceptance exactness`

Proof. Unfold the carrier row of Definition 351.1. The accepted endpoint coordinate a has only two displayed fields: the final run state and a witness that this final state lies in F.

Since w is the empty $\mathsf{List}^{\uparrow}$ spine, Definition 351.3 applies to the accepted packet and identifies the final run state with the initial state q_0. The empty spine has no head symbol and no tail step, so no transition-table route in c can replace this endpoint by a later state.

Substitute this empty-spine endpoint identification into the two-field endpoint row. Its state field becomes q_0. Its remaining field is still the accepting-carrier witness supplied by the endpoint row; after the substitution, that witness has type $q_0 \in F$. Therefore the endpoint row has the displayed form $a = (q_0, \eta)$ with $\eta : q_0 \in F$, and there is no further endpoint coordinate in the packet that could carry additional acceptance data. □

Theorem 351.15 (RegularLanguage DFA prefix consumer determinacy)**.** *Let* $R = (\Sigma, Q, q_0, F, \delta, w, r, a, h, c, p)$ *be a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet satisfying* $\mathsf{RegularLanguageAutomatonPacket}(R)$*, and let* u *be a displayed finite prefix of the* $\mathsf{List}^{\uparrow}$ *word spine* w*. Every accepted prefix consumer endpoint over* u *is determined by the restricted transition ledger, the initial state row* q_0*, the prefix length row, visible* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

In particular, two accepted prefix consumer endpoints with the same displayed finite-prefix rows have the same endpoint state and the same accepting-carrier membership row.

Proof. Apply Definition 351.3 to the displayed prefix. The deterministic ledger fixes the unique run segment starting at the initial state row q_0 and consuming exactly the prefix length row of u.

Next use Definition 351.13. It restricts any prefix consumer to the displayed word spine, the transition ledger segment, endpoint transport, visible hsame transports, Cont routes, Pkg provenance, and local NameCert rows. Hence no accepted prefix endpoint can depend on data outside the restricted ledger.

For the empty prefix boundary, apply Definition 351.14. The endpoint is the start-state row together with the displayed accepting-carrier membership row, so this base case is also determined by the same finite-prefix data.

The positive-length case follows by repeatedly reading the next transition row from the restricted deterministic ledger. Since each step has one displayed transition route from the current state and symbol, the endpoint state after the prefix is fixed, and the accepted endpoint keeps only its membership row in F.

Thus all accepted prefix consumer endpoints with the same finite-prefix rows coincide on the state and membership coordinates. Minimization, host language equality, regular-expression quotient equivalence, infinite automata, and hidden recognizer state are excluded because they are not coordinates of the restricted prefix consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$**.**

Constructive story: A regular-language packet is represented by finite alphabet and state rows, a displayed transition table, a ListUp word spine, a deterministic BHist run ledger, endpoint acceptance, componentwise hsame transport, Cont routes for consuming the spine, and Pkg provenance naming ListUp, FinSetUp, ComputableUp, and TuringMachineUp.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 351.1, Definition 351.2, Definition 351.3, Definition 351.5, Definition 351.6, Definition 351.7, Definition 351.8, Definition 351.10, Definition 351.11, Definition 351.12, Definition 351.13, Definition 351.14, and Definition 351.15 bind the finite automaton carrier, accepted-word classifier, run ledger, endpoint transport, NameCert obligation rows, finite upstream certificate envelope, public accepted-word export, standard finite-automaton bridge boundary, finite bridge consumer completeness, displayed automaton consumer examples, finite prefix bridge consumer boundary, empty-word acceptance exactness, and DFA prefix consumer determinacy.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularLanguage}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularLanguageUp.RegularLanguageAutomatonPacket\_public\_accepted\_word\_export`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host language quotients, external regular-expression equivalence, minimization, infinite automata, or machine-checked bridge evidence.

Upgrade path: No further paper theory-axis upgrade is claimed while formal and bridge verification remain separate.

Chapter 352

A Concrete Naming Certificate for $\mathsf{FiniteMap}^{\uparrow}$

The $\mathsf{FiniteMap}^{\uparrow}$ interface records finite key-value lookup data as displayed BEDC history rows. Its carrier is an association-list presentation over the existing $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, and $\mathsf{Option}^{\uparrow}$ finite-data chapters from chapter 36, chapter 34, and chapter 33, with finite support read through chapter 112 when a consumer needs a displayed support boundary.

352.1 Finite-map source

Definition 352.1 (FiniteMapUp NameCert source)**.** *A* $\mathsf{FiniteMap}^{\uparrow}$ *source is a finite* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *rows whose first component is a displayed key history and whose second component is a displayed value history. The empty spine is the empty-map carrier. A lookup query is evaluated by scanning the displayed spine from left to right and returning an* $\mathsf{Option}^{\uparrow}$ *classifier witness:* none *when no displayed key row matches the query classifier, and* some *of the first displayed value row whose key matches the query classifier otherwise.*

The source also carries a duplicate-key ledger. Each ledger row records the visible key order, the first-hit lookup position for that key when it exists, the later key rows suppressed by the first hit, and the BHist *source provenance of every displayed key-value row. Thus a finite map is not a host function, a primitive finite set of graph pairs, or a quotient of presentations; it is exactly a finite displayed row spine with an* $\mathsf{Option}^{\uparrow}$ *lookup classifier and a suppression ledger.*

Theorem 352.2 (FiniteMapUp carrier and classifier obligations)**.** *The* $\mathsf{FiniteMap}^{\uparrow}$ *carrier described in Definition 352.1 satisfies the local carrier and classifier obligations for a naming certificate. The carrier is inhabited by the empty* $\mathsf{List}^{\uparrow}$ *spine. The lookup classifier is reflexive, symmetric, and transitive because each query is compared through the key classifier and the result is an* $\mathsf{Option}^{\uparrow}$ *witness over the first matching* $\mathsf{Prod}^{\uparrow}$ *row. Carrier rows transport componentwise along* hsame *for keys and values, along the displayed* Cont *lookup route, and along the recorded* Pkg *provenance. No quotient carrier or host-level map equality is part of the obligation surface.*

Proof. The empty spine supplies the carrier inhabitant. For a nonempty spine, every row is already a $\mathsf{Prod}^{\uparrow}$ key-value row inside a $\mathsf{List}^{\uparrow}$ carrier, so the source predicates are inherited componentwise from those chapters.

Reflexivity of lookup compares a query against the same displayed scan. Symmetry and transitivity are inherited from the key classifier and from the $\mathsf{Option}^{\uparrow}$ classifier on the returned witness, while the first-hit rule fixes the selected value row.

Transport follows the same finite spine. Move each key and value through its hsame witness, carry the lookup route through Cont, and keep the package row naming the source certificate. Since equality is read only from the displayed scan and its option witness, there is no quotient carrier to construct. □

Theorem 352.3 (FiniteMapUp lookup Option exactness)**.** *For every accepted* $\mathsf{FiniteMap}^{\uparrow}$ *source and every displayed query key, lookup returns exactly one* $\mathsf{Option}^{\uparrow}$ *witness. It returns* none *precisely when no displayed key row in the finite spine matches the query classifier. It returns* $\mathsf{some}(v)$ *precisely when the first displayed key row matching the query classifier has value row* v*, and every later matching key row is recorded only in the duplicate-key ledger.*

Lean4 ✓ `BEDC.Derived.FiniteMapUp.FiniteMapCarrier lookup option exactness`

Proof. Proceed by induction on the displayed $\mathsf{List}^{\uparrow}$ spine. For the empty spine, there is no displayed key row, so the lookup classifier is the single none witness and the ledger is empty.

For a cons spine, inspect the head $\mathsf{Prod}^{\uparrow}$ row. If its key matches the query classifier, the first-hit rule returns the head value as the unique some witness, and the tail contributes only suppressed duplicate rows for the same key.

If the head key does not match, apply the induction hypothesis to the tail. A tail none remains none for the whole spine, while a tail some remains the first matching row because the head was rejected. These cases exhaust the displayed finite spine. □

Theorem 352.4 (FiniteMapUp ledger transport)**.** *Let two accepted* $\mathsf{FiniteMap}^{\uparrow}$ *sources have key and value rows related componentwise by* hsame*, with lookup continuations related by* Cont *and package rows related by* Pkg*. Then the visible key order, first-hit lookup result, suppressed duplicate-key rows, and source-provenance rows transport together. In particular, lookup and the duplicate-key ledger are preserved by the displayed transport data.*

Lean4 ✓ `BEDC.Derived.FiniteMapUp.FiniteMapCarrier ledger transport`

Proof. Transport the spine row by row. A transported head key has the same classifier outcome against the transported query key, and the transported value row is the value read by the corresponding $\mathsf{Option}^{\uparrow}$ witness.

The first-hit index is therefore stable under the displayed transport. Rows before it remain nonmatching rows, the selected row remains selected, and rows after it that match the same key remain ledger entries rather than lookup outputs.

The Cont routes identify the lookup readback, and the Pkg rows identify the certificate sources used by the scan. Thus the lookup result and the duplicate-key ledger move as one finite displayed object. □

Theorem 352.5 (FiniteMapUp NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FiniteMap}^{\uparrow}}$ *obligation surface consists of five rows: the finite* $\mathsf{List}^{\uparrow}$ *carrier of* $\mathsf{Prod}^{\uparrow}$ *key-value rows from Definition 352.1; the lookup classifier into* $\mathsf{Option}^{\uparrow}$ *with reflexivity, symmetry, and transitivity from Definition 352.2; lookup exactness from Definition 352.3; carrier transport along* hsame*,* Cont*, and* Pkg *from Definition 352.4; and duplicate-key ledger soundness for visible key order, first-hit selection, suppressed duplicates, and source provenance.*

Proof. Collect the carrier row, classifier row, exactness row, transport row, and ledger row. Each is expressed using only the displayed finite spine, product-coded key-value histories, option-valued lookup witness, and finite support boundary supplied by the existing finite-data chapters.

The first four rows are exactly the preceding definition and theorems. The ledger soundness row is the remaining projection of the same source: it records which later rows are suppressed and why the first matching row is the only lookup output.

These five rows are closed under consumer readback because every consumer-visible fact is obtained by scanning the finite displayed spine. No primitive finite-map object, ambient host function, or quotient equality is exported. □

Definition 352.6 (FiniteMapUp displayed support boundary)**.** *Let M be an accepted* $\mathsf{FiniteMap}^{\uparrow}$ *source in the sense of Definition 352.1, presented by a finite* $\mathsf{List}^{\uparrow}$ *spine xs of* $\mathsf{Prod}^{\uparrow}$ *key-value rows. Define* $\mathsf{KeyRows}(xs) : \mathsf{HistSpine}$ *by structural readback from the displayed spine: the empty spine maps to* nil*, and a cons row whose* $\mathsf{Prod}^{\uparrow}$ *first component is the displayed key history k maps to the cons spine with head k and tail* $\mathsf{KeyRows}(xs')$*.*

The displayed support boundary of M is the $\mathsf{FinSet}^{\uparrow}$ *enumeration row*

$$\mathsf{FiniteMapSupport}_{A,\sim_A}(M)(z) \iff \mathsf{FinSetEnumCarrier}_{A,\sim_A}(\mathsf{KeyRows}(xs))(z),$$

where A and $\sim_A$ are the key-side carrier and classifier consumed by the finite-support read. Its ledger contains, for every position of the original $\mathsf{List}^{\uparrow}$ *spine, the corresponding projected key position in* $\mathsf{KeyRows}(xs)$*, the* BHist *provenance of the original key-value row, and the first-hit or suppressed-duplicate status recorded by the finite-map duplicate-key ledger.*

Theorem 352.7 (FiniteMapUp displayed support readback)**.** *Let M be an accepted* $\mathsf{FiniteMap}^{\uparrow}$ *source with displayed* $\mathsf{List}^{\uparrow}$ *spine xs of* $\mathsf{Prod}^{\uparrow}$ *key-value rows. Reading M through the* $\mathsf{FinSet}^{\uparrow}$ *support boundary of Definition 352.6 exposes exactly the displayed key rows of xs:*

$$\mathsf{FiniteMapSupport}_{A,\sim_A}(M)(z) \iff \exists k : \mathsf{BHist}.\ \mathsf{SpineMember}(k, \mathsf{KeyRows}(xs)) \wedge z \sim_A k.$$

Moreover, if two positions of xs display keys in the same $\sim_A$-class, both projected key positions remain in the support ledger, while the $\mathsf{Option}^{\uparrow}$ *lookup read of Definition 352.3 returns only the value at the first matching position and records the later matching positions as suppressed duplicate-key ledger entries.*

Proof. Unfold the support boundary definition. Its carrier is the finite-set enumeration carrier $\mathsf{FinSetEnumCarrier}_{A,\sim_A}(\mathsf{KeyRows}(xs))$. By Definition 112.7, membership in that carrier is equivalent to finite-spine occurrence in $\mathsf{KeyRows}(xs)$ modulo the key classifier. This gives the displayed equivalence.

It remains to identify the ledger and the duplicate behavior. The map $\mathsf{KeyRows}$ is defined by structural readback from the $\mathsf{List}^{\uparrow}$ spine. In the empty case, both the finite-map source and the support spine are empty, so there is no support position and no lookup output. In a cons case, the head $\mathsf{Prod}^{\uparrow}$ row exposes one displayed key history k and one displayed value history v. The support projection adds exactly the key row k at the corresponding head position and then applies the same projection to the tail. Induction on the finite spine therefore gives a position-preserving ledger from rows of xs to rows of $\mathsf{KeyRows}(xs)$, with no added key row and no omitted displayed key row.

Now suppose a later row of xs has a key in the same $\sim_A$-class as an earlier displayed key. The projected key row is still present as a position of $\mathsf{KeyRows}(xs)$, hence it remains visible in the support ledger. At the carrier level, Definition 112.12 says that inserting a source-carried key already represented in the finite spine does not change the induced $\mathsf{FinSet}^{\uparrow}$ classifier. Thus duplicate key positions are ledger entries for the same finite-support membership class rather than additional support classes.

Finally apply Definition 352.3. Lookup scans the original displayed $\mathsf{List}^{\uparrow}$ spine from left to right, so the first key row matching the query supplies the unique some value witness, and every later matching key row is recorded only in the duplicate-key ledger. The support boundary and the lookup boundary therefore read the same displayed key positions, but they use them differently: $\mathsf{FinSet}^{\uparrow}$ records finite support by projected key occurrence modulo the key classifier, while $\mathsf{Option}^{\uparrow}$ lookup emits only the first-hit value and keeps duplicate keys as ledger provenance. □

Theorem 352.8 (FiniteMapUp public support lookup boundary)**.** *The public* $\mathsf{FiniteMap}^{\uparrow}$ *export is exactly the finite* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *key-value rows together with the* $\mathsf{Option}^{\uparrow}$ *first-hit lookup witness, the displayed* $\mathsf{FinSet}^{\uparrow}$ *support boundary, the duplicate-key ledger,* hsame *transport,* Cont *lookup routes, and* Pkg *provenance. Every public read of lookup or support factors through these rows, and no host finite map or quotient graph equality is exported.*

Lean4 ✓ `BEDC.Derived.FiniteMapUp.FiniteMapCarrier public support lookup boundary`

Proof. Start with Definition 352.3. It identifies every lookup read with the scan of the displayed $\mathsf{List}^{\uparrow}$ spine and the resulting $\mathsf{Option}^{\uparrow}$ first-hit witness, while later matching keys remain only in the duplicate ledger.

Apply Definition 352.4 to move the lookup and duplicate rows through the displayed hsame transports, Cont lookup routes, and Pkg provenance. These are the only transport rows available to a public finite-map read.

Finally use Definition 352.7. The support side reads the same displayed key positions through the $\mathsf{FinSet}^{\uparrow}$ support boundary. Combining the lookup exactness, ledger transport, and support readback rows exhausts the public export, with no host finite map and no quotient graph equality left to read. □

Theorem 352.9 (FiniteMapUp support lookup consumer determinacy)**.** *Let M and M' be accepted* $\mathsf{FiniteMap}^{\uparrow}$ *source packets whose displayed* $\mathsf{List}^{\uparrow}$ *spines of* $\mathsf{Prod}^{\uparrow}$ *key-value rows agree up to the displayed key classifier and value classifier, whose* $\mathsf{Option}^{\uparrow}$ *first-hit lookup witnesses agree for every displayed query, whose* $\mathsf{FinSet}^{\uparrow}$ *support boundaries expose the same projected key positions, and whose duplicate-key ledger rows,* hsame *transports,* Cont *lookup routes, and* Pkg *provenance rows agree. Then every public lookup/support consumer reads the same finite BEDC surface from M and M'. In particular, duplicate key rows remain visible as support-ledger positions and transport provenance, but only the first displayed matching row contributes the* $\mathsf{Option}^{\uparrow}$ *value returned to a lookup consumer.*

Lean4 ✓ `BEDC.Derived.FiniteMapUp.FiniteMapCarrier support lookup consumer determinacy`

Proof. First apply Definition 352.3 to both packets. The assumed agreement of displayed $\mathsf{List}^{\uparrow}$ spines and first-hit witnesses forces every public lookup query to reach the same $\mathsf{Option}^{\uparrow}$ read: absent queries read none on both sides, and present queries read the value at the first displayed matching row on both sides.

Next apply Definition 352.7. The public support consumer sees only the projected key positions through the $\mathsf{FinSet}^{\uparrow}$ boundary. Since those projected positions agree, the two support surfaces have the same membership readback modulo the displayed key classifier, including later positions whose keys match an earlier row.

The duplicate ledger is then transported by Definition 352.4. Agreement of the hsame rows, Cont lookup routes, and Pkg provenance means that every duplicate position and every package-origin row has the same public transport account in the two packets.

Finally use Definition 352.8. That boundary exhausts public lookup/support reads by the displayed spine, first-hit lookup witness, support readback, duplicate ledger, transport rows, lookup routes, and package provenance. All of these rows agree by the preceding steps, so no public consumer can distinguish the two packets. Later duplicate keys may still be inspected as support-ledger positions, but the lookup value is already fixed by the first matching displayed row. □

Theorem 352.10 (FiniteMap standard schema bridge boundary)**.** *The public surface of Definition 352.8 and Definition 352.9 supplies the standard finite-map schema bridge exactly through the finite* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *key-value rows, the* $\mathsf{Option}^{\uparrow}$ *first-hit lookup witness, the* $\mathsf{FinSet}^{\uparrow}$ *displayed support boundary, the duplicate-key ledger,* hsame *transports,* Cont *lookup routes, and* Pkg *provenance. Every standard finite-map consumer read factors through those rows.*

The bridge exports no host finite map, quotient graph equality, balanced tree, hashing table, or asymptotic cost claim.

Proof. Project lookup exactness through Definition 352.3. The finite $\mathsf{List}^{\uparrow}$ spine and $\mathsf{Prod}^{\uparrow}$ key-value rows determine the first-hit $\mathsf{Option}^{\uparrow}$ witness, so a consumer lookup can read only the displayed first matching row or the displayed absence witness.

Project support readback through Definition 352.7. The support boundary is the displayed $\mathsf{FinSet}^{\uparrow}$ read of the same key projection, and duplicate key positions remain ledger rows rather than additional support classes.

Transport the duplicate-key ledger with Definition 352.4. The hsame rows, Cont lookup routes, and Pkg provenance carry exactly the visible lookup and support rows across accepted replacements.

Finally apply Definition 352.8. Together with the consumer determinacy theorem Definition 352.9, it exhausts all public finite-map reads by the displayed support and lookup rows. Since the packet contains no coordinate for host maps, quotient graph equality, balanced trees, hashing, or cost, none of those structures is exported by the bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMap}^{\uparrow}}$.

Constructive story: A finite-map packet is represented by a finite ListUp spine of ProdUp key-value BHist rows, an OptionUp first-hit lookup witness, a visible duplicate-key ledger, componentwise hsame transport, Cont lookup routes, Pkg provenance, and finite support readback through FinSetUp when needed.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteMap}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 352.6, Definition 352.7, Definition 352.8, Definition 352.9, and Definition 352.10 bind the bridge surface from the displayed association-list support and first-hit lookup boundary to public finite-map reads over $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteMap}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FiniteMapUp.FiniteMapCarrier\_public\_support\_lookup\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close extensional finite-map quotient equality, host functions, balanced trees, hashing, asymptotic cost, or an ambient finite-set graph primitive.

Upgrade path: Formal verification and bridge checking remain the next verification-axis work for the same finite support and lookup surface.

Chapter 353

A Concrete Naming Certificate for $\mathsf{RegularityModulus}^{\uparrow}$

The $\mathsf{RegularityModulus}^{\uparrow}$ interface names the finite rate witness used by regular dyadic and rational Cauchy data before the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ carriers consume it. In the Bishop-style constructive reading of regular Cauchy moduli, a precision request is not answered by an invisible choice of a tail. It is answered by a displayed rate row together with the finite observations and transport rows that certify the requested dyadic window.

The dyadic-window row is consumed through chapter 291, the finite schedule through chapter 126, the regular-sequence handoff through chapter 301, and the terminal seal through chapter 38. These sibling surfaces are exactly the provenance route recorded by the packet row p.

Definition 353.1 (RegularityModulus dyadic-window carrier)**.** *A* $\mathsf{RegularityModulus}^{\uparrow}$ *carrier packet is a finite* BHist *packet*

$$R = (k, m, w, h, c, p, n)$$

with the following visible rows:

1. *k is a unary precision row naming the requested dyadic scale;*
2. *m is a unary modulus row naming the threshold or window size supplied for that precision;*
3. *w is a dyadic window witness over* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, including the displayed endpoint, radius, and comparison rows read by the regularity test;*
4. *h records the componentwise* hsame *transport rows for the precision, modulus, dyadic-window, and consumer-facing coordinates;*
5. *c records the displayed* Cont *routes by which a precision request, a modulus row, and a dyadic observation window are read in the same finite packet;*
6. *p records the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyModulus}^{\uparrow}$*;*
7. *n is the local* NameCert *row exposing the accepted carrier and its classifier surface.*

The carrier accepts only packets in which these seven rows are displayed in one finite history and the consumer reads follow the recorded Cont *routes. It contains no quotient of streams, no selected real endpoint, and no unrecorded tail choice.*

Lean4 (definition) `BEDC.Derived.RegularityModulusUp.RegularityModulusDyadicWindowCarrier`

Theorem 353.2 (RegularityModulus NameCert seed obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularityModulus}^{\uparrow}}$ *seed obligation surface consists of five finite rows:*

1. *carrier habitation: the zero precision request is carried by the zero dyadic window, with unary rows* k *and* m *and a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero endpoint ledger;*
2. *classifier laws: two packets classify together by componentwise* hsame *on* k, m, w, h, c, p, *and* n*;*
3. *monotone precision transport: increasing the requested precision transports the modulus row and refines the dyadic window through the displayed* Cont *routes;*
4. *ledger exactness: every visible dyadic window read by a consumer is exactly the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger named by* w *and routed by* c*;*
5. *downstream consumption:* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *read the regularity rate through the same finite packet, without quotienting streams or choosing a tail outside the carrier.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusDyadicWindowCarrier seed obligation surface`

Proof. The carrier row is Definition 353.1. For the zero precision request, take the displayed unary zero rows and the zero dyadic endpoint ledger supplied by $\mathsf{DyadicRatCore}^{\uparrow}$; these rows inhabit the packet without using a nonconstructive selector.

Classifier laws are componentwise. Transport each row of the seven-coordinate packet by the corresponding hsame witness and then replay the same Cont routes. The accepted carrier shape is unchanged because every row read by the classifier is already visible.

For monotone precision, a refined request replaces k by a transported unary row and uses the modulus row m to expose the refined window. Ledger exactness then says that the only dyadic observations available to consumers are the endpoint, radius, and comparison rows in w.

Finally, the provenance row p names exactly the downstream consumers: $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyModulus}^{\uparrow}$ over the $\mathsf{StreamName}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ observation surface. Those consumers can therefore read only the finite window and rate rows carried by R. □

Theorem 353.3 (RegularityModulus window monotonicity)**.** *Let* $R = (k, m, w, h, c, p, n)$ *be an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *carrier packet. If a precision row* k' *refines* k *through the displayed unary comparison and the carrier records the corresponding transport in* h *and* c*, then the dyadic window read at* k' *is a refinement of* w *using the same modulus provenance* m *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint family.*

The refined read remains a finite BHist *packet with transported* hsame, Cont, Pkg, *and* NameCert *rows, and it introduces no additional stream quotient or hidden modulus selector.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusDyadicWindowCarrier window monotonicity`
Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket window monotonicity`

Proof. Project the accepted packet through Definition 353.1. The precision row, the modulus row, the dyadic window, and the transport ledger are all coordinates of the same finite history.

Follow the displayed comparison from k to k'. The row h transports the precision and observation coordinates, while c gives the continuation route from the refined precision request to the refined dyadic window.

Repack the transported endpoint, radius, and comparison rows with the same m, p, and n rows. The construction has no coordinate from which a quotient stream or hidden selector could be read. □

Theorem 353.4 (RegularityModulus RegSeqRat consumption boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet can read only the unary precision row* k*, the modulus row* m*, the dyadic window* w*, the transported* hsame *rows* h*, the displayed* Cont *routes* c*, the provenance row* p*, and the local name row* n*. The regular rational sequence carrier receives these rows as a finite regularity-rate witness over* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket regseqrat consumption boundary`

Proof. Read the consumer boundary from the provenance coordinate p of Definition 353.1. Its $\mathsf{RegSeqRat}^{\uparrow}$ entry names the stream schedule and dyadic observation surface that the regular rational sequence carrier is allowed to consume.

The dyadic comparison itself is stored in w, and its transport through refined precision requests is stored in h and c. Hence a $\mathsf{RegSeqRat}^{\uparrow}$ read can recover the rate row and the observation ledger only by projecting those displayed coordinates.

Since the carrier has no quotient stream coordinate and no host real endpoint, the handoff is exactly the finite rate witness over the $\mathsf{StreamName}^{\uparrow}$/$\mathsf{DyadicRatCore}^{\uparrow}$ rows already present in the packet. □

Theorem 353.5 (RegularityModulus RealUp seal handoff)**.** *When a* $\mathsf{Real}^{\uparrow}$ *seal is built from a* $\mathsf{RegSeqRat}^{\uparrow}$ *source carrying an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet, the real-name boundary consumes only the finite regularity-rate rows of that packet. The seal handoff preserves the displayed dyadic window, the modulus row, the* hsame *transport, the* Cont *routes, and the* Pkg *provenance, and it does not add a quotient representative, a selected limit, or a choice-based Cauchy tail.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket realup seal handoff`

Proof. Start with the $\mathsf{RegSeqRat}^{\uparrow}$ consumption boundary of Definition 353.4. It exposes the regular rational source as a finite schedule of dyadic observations together with the rate row and its transport ledger.

The $\mathsf{Real}^{\uparrow}$ seal reads this source through the same finite schedule. Its only regularity information is the dyadic window and modulus data already present in the $\mathsf{RegularityModulus}^{\uparrow}$ packet.

Transport the seal boundary along the displayed hsame rows and follow the recorded Cont routes. Repacking these rows preserves the finite handoff surface and leaves no coordinate for a quotient representative, selected limit, or choice-based tail. □

Theorem 353.6 (RegularityModulus dyadic-window exactness)**.** *For an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet* $R = (k, m, w, h, c, p, n)$*, every consumer-visible dyadic window read is exactly the endpoint, radius, comparison, precision, modulus, transport, continuation, package, and name row displayed by* w, h, c, p, n*. Conversely, every displayed row of this form is accepted as a* $\mathsf{RegularityModulus}^{\uparrow}$ *consumer read.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket dyadic window exactness`

Proof. Project R through Definition 353.1. The carrier stores the unary precision and modulus rows together with the dyadic window data, so the endpoint, radius, comparison, precision, and modulus coordinates are precisely the displayed rows.

For refined precision requests, apply Definition 353.3. This transports the same window data through the recorded hsame and Cont rows without creating a new endpoint or comparison coordinate.

The downstream direction is fixed by the existing handoffs. Use Definition 353.4 for the $\mathsf{RegSeqRat}^{\uparrow}$ read and Definition 353.5 for the $\mathsf{Real}^{\uparrow}$ seal. Both consume only the package and name rows already shown by p and n.

Thus every accepted read is one of the displayed dyadic-window rows, and every displayed row is accepted because it is a coordinate of the carrier together with its transport, continuation, package, and local naming evidence. □

Theorem 353.7 (RegularityModulus common-window exactness)**.** *Let*

$$R = (k, m, w, h, c, p, n) \quad \textit{and} \quad R' = (k', m', w', h', c', p', n')$$

be accepted $\mathsf{RegularityModulus}^{\uparrow}$ *packets. Suppose the displayed unary precision rows of R and R' both refine to a common row k_*, and suppose the dyadic-window, transport, continuation, package, and local naming rows classify componentwise through the displayed* hsame *and* Cont *routes. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of either transported packet factors through one common finite* BHist *window carrying the shared precision row, modulus row, dyadic endpoint rows, transported window,* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *surface.*

The common readback surface exports no quotient stream, selected limit, hidden modulus selector, host real equality, or choice-based tail.

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket common window exactness`

Proof. Project both accepted packets through Definition 353.1. The seven carrier coordinates are the only available readback rows for precision, modulus, dyadic-window data, transport, continuation, provenance, and local naming.

Use Definition 353.3 twice, once for R and once for R', to transport both precision requests to k_*. The componentwise hsame rows identify the transported precision, modulus, and dyadic endpoint coordinates, while the displayed Cont routes identify the finite readback path used by consumers.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, apply Definition 353.4; for a $\mathsf{Real}^{\uparrow}$ seal, apply Definition 353.5. In both cases the consumer can read only the transported finite regularity-rate rows already present in the common carrier surface.

Carrier-coordinate exhaustion gives the conclusion. Since neither packet has a coordinate for quotient streams, selected limits, hidden modulus selection, host real equality, or choice-based tails, none of those objects can be exported by the common window. □

Theorem 353.8 (RegularityModulus shared-rate consumer exhaustion)**.** *For compatible accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packets*

$$R = (k, m, w, h, c, p, n) \quad \textit{and} \quad R' = (k', m', w', h', c', p', n'),$$

any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read factors through the common finite* BHist *window carrying the shared precision row, modulus row, dyadic endpoint rows, transported window,* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *surface. The consumer read has no quotient stream, selected limit, hidden modulus selector, host real equality, or choice-based tail.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket shared rate consumer exhaustion`

Proof. Project both packets through Definition 353.1. The carrier exposes only precision, modulus, dyadic-window, transport, continuation, package, and local naming rows.

Apply Definition 353.7. It supplies the common finite readback window carrying the shared precision row, modulus row, transported dyadic endpoint rows, hsame witnesses, Cont routes, provenance, and local certificate surface.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, use Definition 353.4. For a $\mathsf{Real}^{\uparrow}$ consumer, use Definition 353.5. Both handoffs consume only the finite rows already present in the common window.

Since neither packet carries a quotient stream, selected limit, hidden modulus selector, host real equality, or choice-based tail, no consumer read can export any of those objects. □

Theorem 353.9 (RegularityModulus obligation completeness)**.** *The dyadic-window carrier, monotonicity row,* $\mathsf{RegSeqRat}^{\uparrow}$ *consumption boundary,* $\mathsf{Real}^{\uparrow}$ *seal handoff, dyadic-window exactness, common-window exactness,* hsame *and* Cont *transport,* Pkg *provenance, and local* NameCert *row form the full public obligation surface for* $\mathsf{RegularityModulus}^{\uparrow}$*.*

This surface contains no host modulus function, selected real limit, quotient completion, or global choice of windows.

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket obligation completeness`

Proof. Begin with Definition 353.2. It binds the local NameCert row to the dyadic-window carrier of Definition 353.1, including unary precision, modulus, dyadic endpoint, transport, continuation, package, and name coordinates.

Apply Definition 353.3. This supplies the monotonicity row and keeps refined precision reads on the same displayed hsame and Cont transport surface.

The downstream reads are fixed by Definition 353.4 and Definition 353.5. They show that $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive only the carrier, package, and name rows already present in the obligation surface.

Finally use Definition 353.6 and Definition 353.7. These exactness results exhaust single-window and common-window consumer reads, so no host modulus function, selected real limit, quotient completion, or global window choice is part of the public surface. □

Theorem 353.10 (RegularityModulus carrier and classifier obligations)**.** *An accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet whose seven displayed* BHist *rows are those of Definition 353.1 supplies carrier habitation and componentwise classifier laws for unary precision, unary modulus, dyadic window endpoints, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

No additional carrier coordinate is required for a host modulus function, selected limit, quotient stream, or global choice of observation windows.

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket carrier classifier obligations`

Proof. Read the packet through Definition 353.1. The seven coordinates are inhabited exactly when the precision row, modulus row, dyadic window row, transport row, continuation row, package row, and local name row are present.

The classifier laws are componentwise because each consumer-facing predicate is read from one of these displayed rows. The seed obligation surface of Definition 353.2 identifies the NameCert coordinate with the same finite carrier, so the classifier surface is not allowed to introduce a separate host-side witness.

Window compatibility is controlled by Definition 353.3. It keeps refined precision and modulus observations on the same carrier coordinates, and the dyadic exactness of Definition 353.6 exhausts the visible endpoint rows. □

Theorem 353.11 (RegularityModulus transport and ledger obligations)**.** *For two accepted* RegularityModulus$^{\uparrow}$ *packets connected by the displayed* hsame *and* Cont *rows, transport preserves the precision request, unary modulus row, dyadic observation window, and finite ledger readback recorded by the carrier. The transported packet has exactly the same consumer-visible regularity-rate rows after passage to a common dyadic window.*

Proof. Start from the carrier rows of Definition 353.1. The hsame row transports the precision, modulus, and dyadic endpoint classifiers, while the Cont row records the finite observation route used by downstream readers.

Apply Definition 353.3 to move both precision requests to the common requested rate. The transported rows remain rows of the same finite packet, because refinement only changes the displayed window index and does not add a new carrier coordinate.

Finally use Definition 353.6 for each single packet and Definition 353.7 for the paired packet. These exactness statements say that the ledger visible after transport is precisely the precision, modulus, dyadic-window, transport, continuation, package, and naming surface. □

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket transport ledger obligations`

Theorem 353.12 (RegularityModulus RegSeqRat and Real consumers)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* RegularityModulus$^{\uparrow}$ *packet reads only the finite regularity-rate packet: unary precision, unary modulus, dyadic-window endpoints, componentwise* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

The consumer boundary contains no selected real limit, quotient completion, analytic modulus, host equality, or choice-based tail.

Proof. For the rational-sequence side, apply Definition 353.4. It presents the RegSeqRat$^{\uparrow}$ read as a finite regularity-rate packet rather than as a completed stream object.

For the real-seal side, apply Definition 353.5. The Real$^{\uparrow}$ consumer receives the same precision, modulus, dyadic-window, transport, continuation, package, and name rows.

The exactness theorems Definition 353.6 and Definition 353.7 exhaust the single and paired readback surfaces. Hence neither consumer can read a selected limit, quotient completion, analytic modulus, host equality, or choice-based tail from the packet. □

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket regseqrat real consumer obligations`

Theorem 353.13 (RegularityModulus standard finite-rate bridge)**.** *The public packet determined by Definition 353.9, Definition 353.10, Definition 353.11, and Definition 353.12 has a conservative standard finite regularity-rate reading. Its readable rows are exactly unary precision, unary modulus, dyadic-window endpoint data, displayed* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *evidence.*

Reading the standard finite regularity-rate packet back into RegularityModulus$^{\uparrow}$ *recovers the same carrier rows and the same* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer surfaces. The bridge adds no selected real limit, quotient completion, analytic modulus, host equality, or choice-based tail.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket standard rate bridge`

Proof. Project the accepted packet through the seven carrier coordinates of Definition 353.1. The obligation and classifier theorems identify these coordinates as unary precision, unary modulus, dyadic-window endpoint data, displayed transport, continuation routes, package provenance, and local naming evidence.

Use Definition 353.11 to keep a standard finite-rate read on the same transported carrier rows. The exactness statements cited there ensure that passage to a common dyadic window changes only the displayed finite window and does not create a further analytic row.

For downstream consumers, apply Definition 353.12. Both the $\mathsf{RegSeqRat}^{\uparrow}$ read and the $\mathsf{Real}^{\uparrow}$ seal read factor through the same unary precision, unary modulus, dyadic-window, hsame, Cont, Pkg, and NameCert packet.

Reading back from the standard side therefore returns exactly those carrier rows. Since none of the cited packet coordinates contains a selected real limit, quotient completion, analytic modulus, host equality, or choice-based tail, the standard bridge is conservative over the displayed finite regularity-rate packet. □

Theorem 353.14 (RegularityModulus mature finite-rate consumer completeness)**.** *Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyModulus}^{\uparrow}$ *consumer of the standard finite regularity-rate bridge of Definition 353.13 factors through the same seven visible rows: unary precision, unary modulus, dyadic-window endpoints, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *evidence.*

This consumer-completeness statement supplies no selected real limit, quotient completion, host equality, analytic modulus, or choice-selected tail. It is complete only for the finite regularity-rate packet exposed by $\mathsf{RegularityModulus}^{\uparrow}$ *and the named downstream consumers.*

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket mature rate consumer completeness`

Proof. Start from the consumer surface in Definition 353.8. It exhausts the packet read by the shared finite-rate consumers and leaves only the unary precision, unary modulus, dyadic-window, transport, continuation, package, and name-certificate rows visible.

Use Definition 353.11 to keep that surface stable under the componentwise hsame transport ledger. The transported consumer therefore still reads the same finite rows, with no added analytic or host-equality coordinate.

Apply Definition 353.12 for the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, and then apply Definition 353.13. The bridge identifies the standard readback with exactly the same seven finite rows.

The $\mathsf{CauchyModulus}^{\uparrow}$ consumer is part of the same package provenance named by the standard bridge, so its finite rate read is exhausted by the displayed precision and modulus rows together with the same transport, continuation, package, and local naming evidence. Since none of the cited rows contains a selected limit, quotient completion, host equality, analytic modulus, or choice-selected tail, the mature finite-rate consumer boundary is complete. □

Theorem 353.15 (RegularityModulus scheduled-tail closure)**.** *Let an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet carry a scheduled finite tail transported along its displayed* hsame *and* Cont *ledger. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read using that transported scheduled tail still factors through the same seven visible rows: unary precision, unary modulus, dyadic-window endpoints, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *evidence.*

The closure is only a finite scheduled-tail closure for the accepted regularity-rate packet. It adds no selected limit, quotient completion, host equality, analytic modulus, or choice-selected tail.

Proof. First place the scheduled tail on a common finite window. Apply Definition 353.7; the transported tail read is represented by the shared precision row, modulus row, dyadic-window endpoint rows, hsame rows, Cont routes, Pkg provenance, and NameCert surface.

Next use Definition 353.11. The transport ledger keeps the scheduled tail inside the same carrier coordinates after passage to the common dyadic window, so transport does not create an additional analytic row or a host-equality row.

Finally apply Definition 353.14. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers named there exhaust their reads through the seven finite rows already produced by the common-window and transport steps. Since none of those rows contains a selected limit, quotient completion, analytic modulus, host equality, or choice-selected tail, the transported scheduled-tail read is closed at the finite regularity-rate boundary. □

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket scheduled tail closure`

Theorem 353.16 (RegularityModulus DyadicMesh window factorization)**.** *Let an accepted* $\mathsf{RegularityModulus}^{\uparrow}$ *packet expose a finite regularity-rate window, and let a compatible accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet expose a finite mesh cell whose endpoint-radius-order ledger is read through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows. Then every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{ValidatedNumerics}^{\uparrow}$ *consumer read that uses both packets factors through one finite* BHist *packet.*

That packet contains exactly the unary precision row, unary modulus row, mesh level row, cell-index row, rational-interval coverage row, dyadic endpoint, radius, and order ledger, componentwise hsame *rows, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows of the two accepted packets. The factorization exports no quotient stream, selected limit, host real equality, solver-correctness witness, floating-point semantics, or infinite mesh refinement.*

Proof. Start with the scheduled finite regularity window. By Definition 353.15, the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ side of the read is already exhausted by the precision, modulus, dyadic-window, transport, continuation, package, and local name rows.

Next use Definition 353.14. It identifies the same finite-rate rows as the complete rate-window surface available to the named completion-facing consumers.

Place the mesh read on the compatible finite cell. Apply Definition 358.3 to keep the chosen cell stable under displayed mesh refinement, and apply Definition 358.8 to read the mesh level, cell index, rational-interval coverage, dyadic endpoint-radius-order ledger, transport, continuation, package, and local name rows as one standard finite mesh packet.

Finally align the dyadic endpoint rows through Definition 291.7 and Definition 291.8. These theorems put denominator ledgers and common-exponent endpoint comparisons into the same finite $\mathsf{DyadicRatCore}^{\uparrow}$ window. Repack the rate rows and mesh rows into the shared BHist packet. Since every cited component is finite packet data, the excluded quotient, limit, host-real, solver, floating-point, and infinite-refinement structures have no row from which a consumer can read. □

Lean4 ✓ `BEDC.Derived.RegularityModulusUp.RegularityModulusPacket dyadicmesh window factorization`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularityModulus}^{\uparrow}$ packet is seeded by finite BHist rows for unary precision, unary modulus, a dyadic observation window over $\mathsf{DyadicRatCore}^{\uparrow}$, componentwise hsame transport, displayed Cont routes, Pkg provenance over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyModulus}^{\uparrow}$, and local NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularityModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 353.9, Definition 353.10, Definition 353.11, Definition 353.12, Definition 353.13, Definition 353.14, and Definition 353.16 bind the carrier, classifier, transport, ledger, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, standard finite regularity-rate readback, mature consumer-completeness rows, and the finite $\mathsf{DyadicMesh}^{\uparrow}$ handoff to BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularityModulus}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a checked Lean carrier, real completeness, uniqueness of real representatives, analytic moduli, or choice-selected tails.

Upgrade path: No further paper-side theory-axis upgrade remains for this finite regularity-rate packet.

Chapter 354

A Concrete Naming Certificate for $\mathsf{RationalInterval}^{\uparrow}$

The $\mathsf{RationalInterval}^{\uparrow}$ interface names finite closed rational intervals on the path from $\mathsf{Rat}^{\uparrow}$ and dyadic rational cores to $\mathsf{RegSeqRat}^{\uparrow}$ and the $\mathsf{Real}^{\uparrow}$ finite-window seal. In the constructive analysis tradition of Bishop and Bridges, an interval is not a hidden located subset of the real line. It is a displayed pair of rational endpoints, an order witness, and the finite ledgers by which consumers read containment and refinement.

The dependency route first consumes the $\mathsf{Rat}^{\uparrow}$ endpoint surface of chapter 37 and the dyadic-core radius discipline of chapter 291; only after those finite endpoint and tolerance rows are displayed does the interval hand finite windows to chapter 301.

Definition 354.1 (RationalInterval BHist carrier)**.** *A* $\mathsf{RationalInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (l, r, o, \kappa, h, c, p, n)$$

with the following visible rows:

1. *l is the left endpoint row, a* $\mathsf{Rat}^{\uparrow}$ *carrier displayed in the packet;*
2. *r is the right endpoint row, a* $\mathsf{Rat}^{\uparrow}$ *carrier displayed in the packet;*
3. *o is the order witness row certifying the displayed endpoint comparison $l \leq r$ through the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger;*
4. *κ is the containment ledger recording exactly which rational sample rows are certified to lie between l and r;*
5. *h records the componentwise* hsame *transport rows for endpoints, the order witness, containment rows, and consumer-facing coordinates;*
6. *c records the* Cont *routes by which a finite consumer reads the endpoint pair, order witness, containment ledger, and refinement windows;*
7. *p is the visible* Pkg *provenance row naming the* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependency certificates used by the packet;*
8. *n is the local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *registration row.*

The carrier predicate is $\mathsf{RationalIntervalPacket}(l, r, o, \kappa, h, c, p, n)$*. Its classifier compares two accepted packets only by transporting the displayed endpoint rows and replaying the* $\mathsf{Rat}^{\uparrow}$ *order and containment ledgers over the same finite* BHist *surface.*

Theorem 354.2 (RationalInterval NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *seed obligation surface consists of six finite rows: the carrier of Definition 354.1, endpoint equality through the displayed* $\mathsf{Rat}^{\uparrow}$ *classifiers, the endpoint order witness, containment ledger exactness,* hsame *and* Cont *stability for finite refinements, and the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff boundary for finite rational windows.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket namecert obligation surface`

Proof. Project the eight carrier coordinates. The endpoint rows l and r provide the rational carrier data, while the order witness o and containment ledger κ provide the comparison rows exposed to interval consumers.

The stability rows are exactly h and c. The dependency and local naming surface is carried by p and n, so the six listed obligations account for every row of the packet. □

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket consumer readback semantic name certificate`

Theorem 354.3 (RationalInterval endpoint order witness)**.** *Every accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet displays a* $\mathsf{Rat}^{\uparrow}$ *endpoint pair* (l, r) *together with an order witness row* o *certifying* $l \leq r$ *in the same finite history. Transporting the interval by its stored* hsame *rows transports both endpoints and the order witness componentwise, and it introduces no apartness decision, total order on completed reals, or selected real endpoint outside the displayed* $\mathsf{Rat}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket endpoint order transport`

Proof. Read the endpoint and order coordinates from Definition 354.1. The order witness row o is stored next to l and r, so a consumer cannot obtain the interval without also reading the finite comparison ledger that makes the endpoint order visible.

Under transport, h supplies the componentwise hsame rows for the two endpoints and for o. Repacking those transported rows preserves the same ordered endpoint surface and leaves no coordinate for completed real order or a selected external endpoint. □

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket endpoint order classifier`
Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalEndpointRows order witness`

Theorem 354.4 (RationalInterval containment ledger exactness)**.** *For an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet* $I = (l, r, o, \kappa, h, c, p, n)$*, every rational containment read is exactly one of the finite rows in* κ *routed by* c *and transported by* h*. Conversely, any displayed row of* κ *with its endpoint, order, transport, continuation, package, and local naming evidence is accepted as a* $\mathsf{RationalInterval}^{\uparrow}$ *containment read.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket containment ledger exactness`

Proof. Project I through Definition 354.1. The only containment coordinate is κ, and its routes to consumers are the displayed Cont rows in c.

The forward direction follows by reading the consumer route and then the corresponding containment row. The transported endpoint and order data needed to interpret that row are supplied by h and o.

For the reverse direction, start with a displayed row of κ together with the matching endpoint, order, transport, continuation, package, and name rows. These are exactly the coordinates required by the carrier, so the row is accepted as a containment read. □

Theorem 354.5 (RationalInterval RegSeqRat handoff)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet may read only the displayed* $\mathsf{Rat}^{\uparrow}$ *endpoints, the endpoint order witness, the finite containment ledger, and the associated* hsame, Cont, Pkg, *and* NameCert *rows. The handoff supplies a closed rational window for regular rational sequence arguments; it does not assert Cauchy completeness, select a limit, or identify completed* $\mathsf{Real}^{\uparrow}$ *values.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket regseqrat handoff`

Proof. Start with the carrier packet and project the rows visible to a regular rational sequence consumer. The endpoint rows and order witness delimit the finite window, while κ records the rational samples certified to lie inside it.

The remaining consumer-facing information is precisely the transport, continuation, provenance, and naming data in h, c, p, and n. Repacking those rows gives the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and leaves no row for a quotient Cauchy class or a completed real equality. □

Theorem 354.6 (RationalInterval RealUp seed boundary)**.** *When a finite-window* $\mathsf{Real}^{\uparrow}$ *construction consumes an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet, the seed boundary is the same displayed rational interval packet: two* $\mathsf{Rat}^{\uparrow}$ *endpoints, the order witness, containment ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *row. The boundary is a rational closed window and not a completed real object, a located subset, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket realup seed boundary`

Proof. Use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 354.5. It exposes the rational window as finite endpoint and containment rows, with all reads routed through the same transport and continuation data.

The $\mathsf{Real}^{\uparrow}$ seed consumer can therefore inherit only this finite packet. Since the carrier has no coordinate for a completed real, a located subset, or a bridge witness, the seed boundary remains the displayed rational interval surface. □

Theorem 354.7 (RationalInterval endpoint containment transport)**.** *For accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packets connected by displayed* hsame *rows, endpoint order and containment transport together: the lower endpoint, upper endpoint, order witness, finite containment ledger,* Cont *routes,* Pkg *provenance, and local* NameCert *row remain the complete interval readback surface after transport.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket endpoint containment transport`

Proof. Project the packet through Definition 354.1. The carrier has endpoint coordinates, one endpoint-order coordinate, one containment ledger, and the visible transport, continuation, package, and naming rows.

Apply Definition 354.3 to transport the order witness along the displayed endpoint identifications. The transported endpoints are still the two $\mathsf{Rat}^{\uparrow}$ rows of the same finite carrier.

Use Definition 354.4 for the containment coordinate. Since each containment read is interpreted with the transported endpoint and order rows, the transported ledger is exactly the same finite interval surface and contains no completed-real or located-subset coordinate. □

Theorem 354.8 (RationalInterval common refinement window)**.** *Two accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packets with compatible endpoint and containment rows refine to one finite rational observation window whose visible data are the transported endpoints, the common endpoint-order witness, the common containment ledger, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket common refinement window`

Proof. Begin with the endpoint rows of the two carrier packets from Definition 354.1. Compatibility supplies the displayed hsame routes needed to compare lower endpoints with lower endpoints and upper endpoints with upper endpoints.

The common order row is obtained by applying Definition 354.3 after those endpoint transports. The containment rows are then read against that common ordered window.

Finally apply Definition 354.4. It exhausts the finite containment ledger by rational samples, endpoint-order evidence, transport, continuation, package, and name rows, so the common refinement exports one finite rational observation window and no completed $\mathsf{Real}^{\uparrow}$ object. □

Theorem 354.9 (RationalInterval directed refinement basis)**.** *Any finite family of compatible accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packets admits one displayed finite rational-window basis packet. Its rows are exactly the transported* $\mathsf{Rat}^{\uparrow}$ *lower and upper endpoints, endpoint-order witnesses, containment ledgers, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *evidence needed by the family.*

This basis is directed for $\mathsf{RegSeqRat}^{\uparrow}$ *and finite-window* $\mathsf{Real}^{\uparrow}$ *consumers: every member packet maps to it by finite refinement rows, and every consumer read of the basis is one of the displayed rational-window rows. It exports no completed* $\mathsf{Real}^{\uparrow}$ *equality, located subset, apartness witness, interval arithmetic law, or standard bridge.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket directed refinement basis`

Proof. Start with two compatible accepted packets. Apply Definition 354.8 to obtain their common finite rational observation window. The resulting packet already contains only transported endpoint rows, the order witness, containment ledger, displayed transport, continuation, package, and local naming evidence.

Extend from two packets to a finite family by iterating the binary construction along the displayed finite list of packets. At each step, use Definition 354.11 to compose the previous basis row with the next common-refinement row, so the accumulated packet remains a single finite rational window.

The endpoint rows of the accumulated basis are controlled by Definition 354.12. Lower endpoints transport only along lower $\mathsf{Rat}^{\uparrow}$ coordinates, upper endpoints transport only along upper $\mathsf{Rat}^{\uparrow}$ coordinates, and the order witness is replayed over those transported rows.

Finally apply Definition 354.13. It accounts for all $\mathsf{RegSeqRat}^{\uparrow}$ and finite-window $\mathsf{Real}^{\uparrow}$ reads of the basis, so no completed real equality, located subset, apartness witness, interval arithmetic law, or standard bridge can be projected from the directed finite-window packet. □

Theorem 354.10 (RationalInterval RealUp consumer exactness)**.** *Every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet reads only the two endpoint rows, the endpoint-order witness, the containment ledger, displayed* Cont *routes,* Pkg *provenance, local* NameCert *evidence, and the* hsame *rows needed to transport those data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket realup consumer exactness`

Proof. Use Definition 354.6. It identifies the $\mathsf{Real}^{\uparrow}$ seed boundary with the displayed rational interval packet rather than with a completed real, located subset, or bridge witness.

Endpoint order is supplied by Definition 354.3, and containment is supplied by Definition 354.4. Both rows are coordinates of Definition 354.1, with consumer access routed through the displayed Cont, Pkg, NameCert, and hsame evidence.

Thus the real consumer cannot observe anything outside the finite interval packet. In particular, quotient Cauchy classes, located subsets, interval arithmetic laws, and standard bridges are not part of this consumer surface. □

Theorem 354.11 (RationalInterval refinement composition)**.** *Suppose two displayed finite* $\mathsf{RationalInterval}^{\uparrow}$ *refinement rows share the middle rational endpoint window. If each row is accepted through its displayed* $\mathsf{Rat}^{\uparrow}$ *endpoints, endpoint-order witness, containment ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *evidence, then the two refinements compose to a single accepted finite refinement row from the first window to the last. The composed row replays the shared endpoint classifier rows, concatenates the two finite* Cont *routes, and repackages exactly the same containment-ledger coordinates.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalRefinement composition`

Proof. Project the two refinement packets through Definition 354.1. Their visible coordinates are only endpoint rows, one order witness for each endpoint window, containment ledger rows, transport rows, continuation routes, package provenance, and local naming evidence.

The stored endpoint classifier rows identify the middle lower endpoint with the next lower endpoint, and the middle upper endpoint with the next upper endpoint. With those rows fixed, the two order witnesses read the same middle window and can be replayed as one finite refinement step across the outer endpoints.

Concatenate the two displayed Cont routes and carry along the two containment ledgers. The containment coordinates are already interpreted against the transported endpoint and order rows, so repacking them yields the outer refinement packet and introduces no completed $\mathsf{Real}^{\uparrow}$ equality, located subset, or bridge coordinate. □

Theorem 354.12 (RationalInterval endpoint transport classifier)**.** *Endpoint transport for accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packets preserves the interval classifier componentwise: lower endpoints transport only along lower* $\mathsf{Rat}^{\uparrow}$ *rows, upper endpoints transport only along upper* $\mathsf{Rat}^{\uparrow}$ *rows, and the endpoint-order coordinate is read against those transported components. Consequently endpoint transport cannot introduce a completed* $\mathsf{Real}^{\uparrow}$ *equality row or an apartness row.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket endpoint transport classifier`

Proof. Read the endpoint coordinates from Definition 354.1. The carrier separates the lower endpoint, upper endpoint, and endpoint-order witness, and each classifier row is attached to its corresponding endpoint coordinate.

Use the stored hsame rows to transport lower endpoints along lower coordinates and upper endpoints along upper coordinates. The order witness is then replayed over those two transported coordinates, so the classifier has the same lower, upper, and order components after transport.

No coordinate of the packet contains a completed $\mathsf{Real}^{\uparrow}$ equality or an apartness witness. Since transport only reuses the displayed endpoint classifier rows and the order witness coordinate, neither kind of row can be created by endpoint transport. □

Theorem 354.13 (RationalInterval finite-window consumer exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and every finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet factors through the displayed endpoint, order, containment, transport, continuation, package, and local* NameCert *rows. These rows exhaust the consumer surface of the finite rational window.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket window consumer exhaustion`

Proof. Project the accepted carrier tuple. Its observable coordinates are precisely the two $\mathsf{Rat}^{\uparrow}$ endpoints, the endpoint-order witness, the finite containment ledger, the displayed hsame rows, Cont routes, Pkg provenance, and the local naming evidence.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, apply Definition 354.5. The handoff reads the rational window only through those endpoint, order, containment, transport, continuation, package, and name rows.

For a finite-window $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 354.6 and then Definition 354.10. The seed boundary and consumer exactness identify the same finite packet as the complete readback surface, so the consumer has no additional row for quotient reals, located subsets, apartness, or standard bridges. □

Theorem 354.14 (RationalInterval standard boundary bridge)**.** *For an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet, the endpoint order witness, finite containment ledger, endpoint transport classifier, refinement composition, and finite-window consumer exhaustion determine a conservative standard rational-interval boundary. Reading that boundary back recovers exactly the displayed* BHist *packet: the lower endpoint, upper endpoint, order row, containment ledger, endpoint transport rows, continuation routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows.*

The boundary is conservative. It excludes completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, interval arithmetic laws, and host standard interval data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket standard boundary bridge`

Proof. Project the accepted packet through Definition 354.1. The visible coordinates are the lower $\mathsf{Rat}^{\uparrow}$ endpoint, upper $\mathsf{Rat}^{\uparrow}$ endpoint, order row, containment ledger, endpoint transport rows, continuation routes, package provenance, and local name-certificate evidence.

Use Definition 354.3 to place the two endpoint rows in the conservative boundary order. Apply Definition 354.4 to identify the finite containment rows as the entire membership ledger for the displayed window.

Next compose Definition 354.11 with Definition 354.12. Refinement keeps the lower and upper endpoints on the same finite rational window, and endpoint transport keeps each classifier row attached to its displayed endpoint coordinate.

Finally apply Definition 354.13. Every $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ consumer has already factored through the projected rows, so reading the conservative boundary back componentwise recovers the original BHist packet and adds no completed real, quotient class, located subset, interval law, or host standard interval field. □

Theorem 354.15 (RationalInterval public rational-window handoff)**.** *The endpoint-order witness, containment ledger, endpoint transport classifier, common-refinement window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seed boundary, and finite-window consumer exhaustion assemble a public* $\mathsf{RationalInterval}^{\uparrow}$ *handoff packet over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The handoff exposes only the finite rational-window rows named by those objects. It excludes completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, and standard bridges.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket public rational window handoff`

Proof. Project the carrier tuple from Definition 354.1. The visible data are the two endpoint rows, their order witness, containment ledger, endpoint transport coordinates, continuation routes, package provenance, and local naming evidence.

Use Definition 354.3 and Definition 354.4. These identify the ordered endpoint pair and the exact finite containment rows that a public rational-window handoff may expose.

Compose Definition 354.8 with Definition 354.12. The common window supplies the shared finite rational interval surface, and endpoint transport keeps the lower and upper classifier rows attached to their own $\mathsf{Rat}^{\uparrow}$ coordinates.

Finally repack the resulting finite window through Definition 354.5, Definition 354.6, and Definition 354.13. These consumer theorems account for all $\mathsf{RegSeqRat}^{\uparrow}$ and finite-window $\mathsf{Real}^{\uparrow}$ reads, and none of the repacked coordinates contains completed real equality, quotient Cauchy data, located subset structure, apartness, interval arithmetic, or a standard bridge. □

Theorem 354.16 (RationalInterval endpoint-width ledger)**.** *Every certified* $\mathsf{RationalInterval}^{\uparrow}$ BHist *packet exposes exactly the lower endpoint row, the upper endpoint row, a dyadic width witness, the finite containment row, and the* Pkg *provenance row inherited from the displayed* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces.*

No quotient real, host interval, ambient located subset, interval arithmetic operation, or completed equality relation is projected by this ledger.

Proof. Project the certified packet through the same carrier rows used by Definition 354.15. The endpoint coordinates are the two displayed $\mathsf{Rat}^{\uparrow}$ rows, while the width and containment coordinates are the dyadic witness and finite containment row.

Apply Definition 354.3 to bind the endpoint pair to the visible order surface. Then use Definition 354.4 to identify the containment component as the whole finite ledger for that endpoint window.

Finally read the package coordinate componentwise. The public handoff of Definition 354.15 lists the only consumer-facing rows, so a quotient real, host interval, located subset, interval operation, or completed equality relation has no carrier coordinate from which it could be projected. □

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket endpoint width ledger`

Theorem 354.17 (RationalInterval dyadic transport stability)**.** *Transporting both endpoint histories and the dyadic width witness along the displayed* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *classifiers preserves the* $\mathsf{RationalInterval}^{\uparrow}$ *classifier under componentwise* hsame *transport and* Cont *continuation routes.*

The transported packet remains a finite rational-window interval packet over the same endpoint, width, containment, and package rows.

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket dyadic transport stability`

Proof. Start with the displayed lower and upper endpoint histories. By Definition 354.3, their order relation is recorded as a visible endpoint row and is therefore transported with the endpoints.

Use Definition 354.8 to place the source and target reads on a common finite rational window. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 354.5 transports the finite regular-sequence observations through the same hsame and Cont rows.

The dyadic width witness is part of the displayed finite window, so it follows the transported endpoints along the same classifier route. Apply Definition 354.4 to re-identify the containment row after transport, yielding the original $\mathsf{RationalInterval}^{\uparrow}$ classifier shape. □

Theorem 354.18 (RationalInterval RegSeqRat window coverage)**.** *Every finite* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window admitted by a* $\mathsf{RationalInterval}^{\uparrow}$ *source packet has a containment or refinement witness in the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger.*

Equivalently, a finite regular rational sequence consumer cannot read outside the endpoint, dyadic-width, containment, refinement, and provenance rows already displayed by the interval packet.

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket regseqrat window coverage`

Proof. Take a finite $\mathsf{RegSeqRat}^{\uparrow}$ observation window admitted by the source packet. By Definition 354.5, that window reaches the interval only through the displayed rational endpoint and containment rows.

Apply Definition 354.8 to compare the consumer window with the source interval window. The common refinement supplies the finite row on which both observations are read.

Use Definition 354.4 to decide the corresponding containment entry, and use Definition 354.3 to keep the refined row between the displayed endpoints. The public handoff of Definition 354.15 then exhausts all remaining consumer reads through the same finite ledger. □

Theorem 354.19 (RationalInterval regular-window dyadic endpoints)**.** *Every finite* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window admitted by a certified* $\mathsf{RationalInterval}^{\uparrow}$ *packet is bounded by the displayed lower endpoint, upper endpoint, and dyadic-width rows.*

Any refinement or containment witness consumed by $\mathsf{Real}^{\uparrow}$ *factors through those same endpoint, width, containment,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, with no quotient real or completed equality coordinate.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket regular window dyadic endpoints`

Proof. First expose the visible endpoint and width surface by applying Definition 354.16. This gives the lower endpoint row, upper endpoint row, dyadic-width witness, containment row, and package provenance read by the finite interval packet.

Next transport the endpoint and width data along classifier reads. Apply Definition 354.17; the transported packet remains over the same endpoint, width, containment, BHist, hsame, Cont, package, and local certificate rows.

Finally use Definition 354.18. It exhausts every finite $\mathsf{RegSeqRat}^{\uparrow}$ consumer read by a containment or refinement entry in the displayed ledger, so the $\mathsf{Real}^{\uparrow}$ consumer boundary factors through the same finite rows and adds no quotient real or completed equality coordinate. □

Theorem 354.20 (RationalInterval finite consumer completeness)**.** *For every accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet, every finite consumer read of endpoint order, endpoint containment, common refinement, dyadic transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *window coverage, or* $\mathsf{Real}^{\uparrow}$ *finite-window handoff factors through the displayed rational-interval packet rows and the standard boundary bridge.*

The packet rows available to such a consumer are exactly the displayed $\mathsf{Rat}^{\uparrow}$ *endpoint rows, the dyadic-width row, the finite containment and common refinement rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *window row, the* $\mathsf{Real}^{\uparrow}$ *finite-window handoff row, the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and the standard boundary bridge. The completeness claim does not assert completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, or interval arithmetic.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket finite consumer completeness`

Proof. Start with a finite consumer read accepted by the $\mathsf{RationalInterval}^{\uparrow}$ packet. If the read asks for endpoint order, containment, common refinement, or a finite-window handoff, apply Definition 354.13 to place it on one of the displayed rational-interval rows.

For a boundary-facing read, apply Definition 354.14. This routes the same finite packet through the standard boundary bridge without adding a completed real equality, a quotient Cauchy class, or a located-subset carrier.

For a transported endpoint or width read, apply Definition 354.17. For a regular rational sequence observation, apply Definition 354.18. These two facts keep dyadic transport and $\mathsf{RegSeqRat}^{\uparrow}$ windows inside the existing endpoint, width, containment, refinement, provenance, and handoff rows.

The listed cases exhaust the finite consumer interfaces named in the statement. Interval arithmetic and completed real structure require carriers or relations absent from the packet rows, so they are not produced by this factorization. □

Theorem 354.21 (RationalInterval nested consumer window factorization)**.** *Let I be an accepted* $\mathsf{RationalInterval}^{\uparrow}$ BHist *packet. Suppose two finite consumers read only endpoint containment, common refinement,* $\mathsf{RegSeqRat}^{\uparrow}$ *window coverage, or* $\mathsf{Real}^{\uparrow}$ *finite-window handoff rows, and suppose the second read is displayed under the first by a finite* Cont *route. Then the nested pair factors through one displayed finite rational-window packet over the same* $\mathsf{Rat}^{\uparrow}$ *endpoints, endpoint-order row, containment ledger,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *row.*

The factor packet is purely finite. It contains no completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy class, located subset datum, host interval object, apartness row, or interval arithmetic operation.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket nested consumer window factorization`

Proof. Apply Definition 354.20 to each consumer read. Both reads land on displayed finite interval rows: endpoints, order, containment, refinement, regular-window, finite-window handoff, transport, continuation, provenance, and local naming evidence.

Next place the two readback windows on a common finite rational packet. Use Definition 354.8 for the binary common window, and use Definition 354.9 to choose it as a basis for the finite nested family of reads.

The displayed nesting route is then composed with the refinement routes by Definition 354.11. The composite route still reads the same endpoint, order, containment, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally apply Definition 354.14. The standard reading is conservative over the finite rational-window packet, so completed real equality, quotient Cauchy classes, located subsets, and host interval objects remain outside the factorization. □

Theorem 354.22 (RationalInterval window overlap consumer amalgamation)**.** *Let two finite* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers, or two finite-window* $\mathsf{Real}^{\uparrow}$ *consumers, read an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet. Suppose their reads are compatible on displayed endpoint rows, endpoint order, containment rows, common-refinement rows, or handoff rows, and suppose the two windows overlap through displayed* hsame *and* Cont *routes.*

Then the two reads amalgamate into one finite rational-window packet over the same $\mathsf{Rat}^{\uparrow}$ *endpoints. The amalgamated packet carries the endpoint-order row, the containment ledger, the common finite refinement row, the* BHist *histories, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows consumed by the two original reads.*

The amalgamation introduces no completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy class, located subset, apartness predicate, interval arithmetic operation, or host interval datum.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket window overlap consumer amalgamation`

Proof. First apply Definition 354.20 to the two consumer reads. Each read is already exhausted by finite interval rows: endpoint data, order, containment, refinement, finite-window handoff, BHist histories, hsame, Cont, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ row.

Next use Definition 354.8 on the compatible overlapping windows. This gives a binary finite rational window which carries both displayed reads over the same endpoint boundary.

Choose the displayed finite basis for that overlap by Definition 354.9. The chosen packet keeps the endpoint-order row, containment ledger, common-refinement evidence, handoff rows, hsame transport, and Cont routes in the rational-window surface.

Finally apply Definition 354.21 to factor any nested access through the chosen window. The row exhaustion from the first step shows that no completed real equality, quotient Cauchy class, located subset, apartness row, interval arithmetic, or host interval datum is available to the amalgamation. □

Theorem 354.23 (RationalInterval overlap RegSeqRat handoff exactness)**.** *Let an accepted* $\mathsf{RationalInterval}^{\uparrow}$ BHist *packet be read by two overlapping finite* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers, or by two overlapping finite-window* $\mathsf{Real}^{\uparrow}$ *consumers. Suppose the two reads agree on displayed endpoint rows, endpoint order, containment rows, common-refinement rows, and finite handoff rows.*

Then both reads factor through one rational-window packet. Its exact exported rows are the two $\mathsf{Rat}^{\uparrow}$ *endpoints, the endpoint-order witness, containment ledger, width ledger, directed refinement, displayed* hsame *rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows. The factorization exports no completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy class, located subset, apartness predicate, interval arithmetic operation, or host interval datum.*

Proof. Apply Definition 354.22 to the two overlapping reads. This gives the common finite rational window carrying the shared endpoints, order, containment, refinement, BHist histories, displayed hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ rows.

Use Definition 354.18 for finite $\mathsf{RegSeqRat}^{\uparrow}$ reads and Definition 354.10 for finite-window $\mathsf{Real}^{\uparrow}$ reads. These two consumer exactness results identify the handoff rows as the displayed rational endpoint rows, containment ledger, width ledger, and directed-refinement data of the same window.

Finally cite Definition 354.14. The bridge is conservative over the finite BEDC packet, so the handoff remains internal to the rational-window rows and cannot export completed real equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, or host interval data. □

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket overlap regseqrat handoff exactness`

Theorem 354.24 (RationalInterval common refinement associativity)**.** *Let I, J, and K be three compatible accepted* $\mathsf{RationalInterval}^{\uparrow}$ *finite packets. Suppose their displayed pairwise common-refinement rows share the appropriate middle endpoint windows, so that one may first refine I with J and then with K, or first refine J with K and then with I.*

The two iterated refinements determine the same consumer-visible finite rational-window packet. They expose the same transported $\mathsf{Rat}^{\uparrow}$ *endpoint rows, endpoint-order witnesses, containment and refinement rows, dyadic-width row,* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window row,* $\mathsf{Real}^{\uparrow}$ *finite-window handoff row, visible* BHist, hsame, Cont, Pkg, *and local* NameCert *rows. The claim does not assert completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, or host standard interval data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket common refinement associativity`

Proof. Form the two binary common-refinement windows by applying Definition 354.8 to the compatible pairs. The shared middle endpoint hypotheses put both binary windows over the same displayed endpoint-order and containment rows.

Compose the refinement rows in the left-associated direction with Definition 354.11. Apply the same theorem in the right-associated direction. Both composites are therefore finite BHist packets whose refinement entries are built from the same pairwise rows.

Identify the transported endpoint coordinates componentwise by using Definition 354.12. Lower endpoints, upper endpoints, endpoint-order witnesses, containment rows, and dyadic-width rows agree at the consumer surface because each coordinate is read through the same transported $\mathsf{Rat}^{\uparrow}$ classifier data.

Finally apply Definition 354.20. It says every finite consumer read of the two composites factors through exactly the displayed rational-window rows, the $\mathsf{RegSeqRat}^{\uparrow}$ observation row, the $\mathsf{Real}^{\uparrow}$ finite-window handoff row, the visible BHist, hsame, Cont, Pkg, and local NameCert rows. Hence the two iterated refinements expose one finite consumer packet and add none of the excluded completed-real, quotient, located-subset, apartness, arithmetic, or host standard interval structure. □

Theorem 354.25 (RationalInterval midpoint bisection window)**.** *Let I be an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *packet with displayed lower endpoint, upper endpoint, dyadic-width witness, containment ledger, visible* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows. Suppose a midpoint row m is displayed on the same* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint surface, and suppose two containment subledgers display the left window from the lower endpoint to m and the right window from m to the upper endpoint.*

Then the two half-windows are accepted $\mathsf{RationalInterval}^{\uparrow}$ *finite packets. They share the midpoint boundary row, their endpoint and dyadic-width rows are read from the same displayed surface, and every* $\mathsf{RegSeqRat}^{\uparrow}$ *or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer read of either half-window is one of the original endpoint, width, containment, refinement,* BHist*,* hsame*,* Cont*,* Pkg*, or local* NameCert *rows. Conversely, every original finite consumer read whose endpoint and containment rows lie in one half factors through the corresponding half-window packet.*

The bisection window exports no completed $\mathsf{Real}^{\uparrow}$ *equality, located subset datum, open cover, quotient Cauchy class, apartness row, interval arithmetic operation, or standard bridge datum.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket midpoint bisection window`

Proof. Use Definition 354.16 to read the lower endpoint, upper endpoint, midpoint row, and dyadic-width witnesses from the same $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ surface. Apply Definition 354.3 to obtain the displayed endpoint order for the left and right half-windows.

Next construct the two finite packets. Apply Definition 354.8 to the parent window and the midpoint row, and compose the resulting refinement routes with Definition 354.11. This gives accepted left and right $\mathsf{RationalInterval}^{\uparrow}$ packets whose common endpoint is exactly the displayed midpoint boundary.

The order in which the parent containment ledger, midpoint row, and two subledgers are grouped does not affect the consumer-visible packet. This is Definition 354.24, applied to the two ways of parenthesizing the same finite refinement data.

Finally apply Definition 354.20. Every $\mathsf{RegSeqRat}^{\uparrow}$ or finite-window $\mathsf{Real}^{\uparrow}$ read of a half-window is exhausted by the listed endpoint, width, containment, refinement, BHist, hsame, Cont, Pkg, and local NameCert rows, and the same exhaustion gives the converse factorization for parent consumer reads restricted to either half. □

Theorem 354.26 (RationalInterval refinement choice neutrality)**.** *Fix a* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint packet over* BHist*. For any two common-refinement windows whose endpoint, containment, dyadic-transport,* Cont*,* Pkg*, and local* NameCert *rows satisfy the directed-refinement and associativity hypotheses, the resulting finite consumer packages for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *are* hsame*-equivalent.*

Thus a finite $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may use any displayed common-refinement witness for the fixed endpoint packet without selecting completed real equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, or host standard interval data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalRefinement choice neutrality`
↪ variant `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket refinement choice neutrality`

Proof. Use Definition 354.8 to put both refinement choices over explicit finite endpoint and containment windows, and use Definition 354.9 to compare their directed basis rows inside one displayed finite BHist packet.

Compose the comparison routes with Definition 354.11. Then Definition 354.12 identifies the endpoint, containment, dyadic-width, Cont, Pkg, and local NameCert rows seen after the displayed hsame transport.

Finally apply Definition 354.24. It identifies the two readings at the finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer surface exposed by the fixed endpoint packet. □

Theorem 354.27 (RationalInterval bisection refinement nesting)**.** *Let I be an accepted* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint packet with displayed lower endpoint, midpoint row, upper endpoint, containment ledger, dyadic-width rows, visible* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows. Suppose the midpoint bisection of Definition 354.25 displays the left and right half-windows, and suppose either half-window is refined once more by a finite rational-window packet over the same endpoint surface.*

Then the two-step choice factors through one finite rational-window packet over the original endpoint, midpoint, containment, dyadic-width, BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or finite-window* $\mathsf{Real}^{\uparrow}$ *consumer of the nested half-window reads this packet, and changing the displayed common-refinement witness changes only the* hsame *representative of the same finite consumer package.*

The nesting claim does not assert completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, or host interval data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket bisection refinement nesting`

Proof. First apply Definition 354.25. It constructs the accepted left and right half-window packets from the parent endpoint, midpoint, containment, dyadic-width, BHist, hsame, Cont, Pkg, and local NameCert rows.

Next compose the chosen half-window refinement with the parent bisection route. The finite parenthesization of that two-step route is exactly the content of Definition 354.24; it repacks the displayed endpoint and containment rows as one rational-window packet over the original endpoint surface.

Finally use Definition 354.26. Any alternate displayed common-refinement witness for the same nested half-window has the same finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer package up to hsame. Since all rows used by the construction are the listed finite BEDC rows, no completed-real equality, quotient class, located-subset datum, apartness row, interval operation, or host interval structure is exported. □

Theorem 354.28 (RationalInterval refinement normal form)**.** *Any finite tree of accepted* $\mathsf{RationalInterval}^{\uparrow}$ *containment, bisection, and common-refinement reads consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or by a finite-window* $\mathsf{Real}^{\uparrow}$ *consumer factors through one left-associated directed rational-window basis packet. The packet carries the same displayed* $\mathsf{Rat}^{\uparrow}$ *lower and upper endpoint rows, dyadic-width rows, containment ledgers,* BHist *entries,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows as the original tree.*

Every $\mathsf{RegSeqRat}^{\uparrow}$ *or finite-window* $\mathsf{Real}^{\uparrow}$ *read of the tree is therefore a read of this left-associated basis packet, and changing the displayed common refinement or bisection witnesses changes only the representative of the same finite consumer package. The normal form exports no completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy class, located subset, apartness row, interval arithmetic, or host standard interval.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket refinement normal form`

Proof. Induct on the finite refinement tree. A leaf is already an accepted $\mathsf{RationalInterval}^{\uparrow}$ rational-window packet, so its endpoint, width, containment, BHist, hsame, Cont, Pkg, and NameCert rows give the required basis packet.

For a binary common-refinement node, apply Definition 354.8 to place the two child packets over a common finite rational window. Then use Definition 354.9 to choose the left-associated directed basis for the finite family of endpoint and containment rows.

For a nested refinement node, compose the child basis route with the displayed outer route by Definition 354.11. The rebracketing needed to keep the basis left-associated is exactly Definition 354.24.

For a bisection node, apply Definition 354.25 to expose the two half-window packets and then use the induction hypothesis on the selected half. The rows remain over the same endpoint, midpoint, dyadic-width, containment, BHist, hsame, Cont, Pkg, and local NameCert surface.

Finally apply Definition 354.26. It identifies all displayed witness choices at the finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer surface. Since every step only repacks the listed finite rational-window rows, the excluded completed-real, quotient, located-subset, apartness, arithmetic, and host-standard structures are not exported. □

Theorem 354.29 (RationalInterval normal form terminal seal compatibility)**.** *Let T be any finite* $\mathsf{RationalInterval}^{\uparrow}$ *containment, bisection, and common-refinement tree whose terminal rows are consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or by a finite-window* $\mathsf{Real}^{\uparrow}$ *read. Replacing T by the left-associated directed normal form of Definition 354.28 preserves the terminal seal surface exactly: the visible* $\mathsf{Rat}^{\uparrow}$ *lower and upper endpoints, dyadic-width rows, containment and refinement ledgers,* BHist *entries,* hsame *rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows are the same finite rows for every terminal consumer.*

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and any finite-window* $\mathsf{Real}^{\uparrow}$ *consumer sees the same terminal packet before and after normal-form replacement. The compatibility statement exports no completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy class, located subset, apartness row, interval arithmetic operation, or host interval datum.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalRefinement terminal seal compatibility`

Proof. Apply Definition 354.28 to factor the finite tree through the left-associated directed rational-window basis. The factorization is defined only by repacking the displayed endpoint, dyadic-width, containment, refinement, history, transport, continuation, package, and local certificate rows.

For a $\mathsf{RegSeqRat}^{\uparrow}$ terminal read, use Definition 354.20. That theorem says that every finite consumer read factors through the displayed packet rows, so the normal form has the same seal surface as the original tree at the $\mathsf{RegSeqRat}^{\uparrow}$ boundary.

For a finite-window $\mathsf{Real}^{\uparrow}$ terminal read, use Definition 354.21. It restricts the read to the nested finite rational-window rows and hence to the same endpoint, width, ledger, BHist, hsame, Cont, Pkg, and NameCert coordinates preserved by the normal form.

The construction never introduces coordinates for completed real equality, quotient Cauchy classes, located subsets, apartness, interval arithmetic, or host interval data. Those objects therefore remain outside the terminal seal surface. □

Theorem 354.30 (RationalInterval dyadic nested amalgamation normal form)**.** *Let $\mathcal{R}$ be a finite family of nested or overlapping* $\mathsf{RationalInterval}^{\uparrow}$ *consumer reads whose endpoint rows are compatible* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows over the same rational boundary. Then $\mathcal{R}$ has a single displayed rational-window packet carrying endpoint order, containment, common-refinement evidence, finite-window handoff,* BHist *histories, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ *rows.*

Every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *finite-window consumer of those reads factors through that packet. The factorization uses only the displayed dyadic endpoint rows and rational-window ledgers; it does not introduce completed* $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, located subsets, apartness predicates, interval arithmetic operations, or host interval data.*

Lean4 ✓ `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket dyadic nested amalgamation normal form`

Proof. First apply Definition 354.20 to each consumer read. Each member of $\mathcal{R}$ is exhausted by endpoint order, containment, finite refinement, finite-window handoff, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$ rows.

Use Definition 354.22 on overlapping reads and Definition 354.21 on nested reads. These two reductions place the family inside finite rational-window packets with compatible dyadic endpoints.

Apply Definition 354.24 to associate the resulting refinement witnesses, and use Definition 354.27 to keep all dyadic endpoint rows inside the same nested boundary. The finite family condition lets these binary amalgamations be iterated until one displayed rational-window packet remains.

Finally invoke Definition 354.28. The normal form is the announced packet, and the finite-window handoff rows provide the required factorization for every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$.

Constructive story: A rational closed interval is represented by two displayed RatUp endpoint rows, an endpoint-order witness, a finite containment ledger, and visible hsame, Cont, package, and NameCert rows consumed by RegSeqRat and RealUp finite-window arguments.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 354.7, Definition 354.8, Definition 354.24, Definition 354.10, Definition 354.14, and Definition 354.20 bind the endpoint, order, containment, transport, package, local NameCert rows, common-refinement associativity, standard boundary bridge, and finite consumer completeness to BHist, hsame, Cont, Pkg, NameCert, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ for the finite interval packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalInterval}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RationalIntervalUp.RationalIntervalPacket\_standard\_boundary\_bridge`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close completed RealUp equality, quotient Cauchy classes, located subsets, or interval arithmetic.

Upgrade path: No further theory-axis upgrade is available for this chapter; formal upgrades remain on the verification axis.

Chapter 355

A Concrete Naming Certificate for $\mathsf{RationalBall}^{\uparrow}$

The $\mathsf{RationalBall}^{\uparrow}$ interface names finite rational-radius balls on the path from rational intervals and dyadic radius rows to metric and real-name consumer windows. In the constructive analysis tradition of Bishop and Bridges, a metric ball used here is a located finite witness: a center row, a visible nonnegative radius row, and the finite ledgers by which rational samples are certified inside the radius. It is not an open-cover topology object and it carries no hidden completed real.

$\mathsf{RationalBall}^{\uparrow}$ consumes the finite endpoint-and-containment discipline of chapter 354. The ball ledger refines interval containment by adding the visible center row, nonnegative radius row, and rational-radius comparison route.

Definition 355.1 (RationalBall BHist carrier)**.** *A* $\mathsf{RationalBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (c, r, o, h, k, p, n)$$

with the following visible rows:

1. *c is the center row, a* $\mathsf{Rat}^{\uparrow}$ *carrier displayed in the packet;*
2. *r is the dyadic or rational radius row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *as the radius budget for the ball;*
3. *o is the nonnegative radius and order witness row, certifying that the displayed radius is usable as a finite metric bound;*
4. *h records the componentwise* hsame *transport rows for the center, radius, order witness, containment ledger, consumer-facing metric rows, and real-window handoff coordinates;*
5. *k is the containment ledger recording exactly which finite rational sample rows are certified to lie within the displayed radius of c;*
6. *p is the visible* Pkg *provenance row naming the consumed* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *certificate surfaces;*
7. *n is the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *registration row.*

The carrier predicate is $\mathsf{RationalBallPacket}(c, r, o, h, k, p, n)$*. Its classifier compares two accepted packets only by transporting the displayed center and radius rows, then replaying the finite containment ledger through the same displayed* BHist *surface.*

Lean4 ✓ `BEDC.Derived.RationalBallUp.TasteGate.RationalBallTasteGate single carrier alignment`

Theorem 355.2 (RationalBall NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *seed obligation surface consists of six finite rows: the carrier of Definition 355.1, center equality through the displayed* $\mathsf{Rat}^{\uparrow}$ *classifier, radius compatibility through* $\mathsf{DyadicRatCore}^{\uparrow}$*, the nonnegative radius witness, containment ledger exactness, and* hsame *stability for the* Cont *routes that expose the* $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumer windows.*

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket namecert obligations`

Proof. Project the seven carrier coordinates. The center row c, radius row r, and order witness o provide the finite rational-radius data.

The containment surface is exactly the ledger k, interpreted with the transport rows in h. The package row p names the downstream interval, regular-sequence, metric, and real-name surfaces, while n supplies the local registration row.

These rows account for the stated seed obligations. No coordinate remains for an open cover, a completed metric space point, or an ambient real ball. □

Theorem 355.3 (RationalBall containment exactness)**.** *For an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet* $B = (c, r, o, h, k, p, n)$*, every rational containment read is exactly one row of* k *transported by* h *and interpreted against the displayed center* c*, radius* r*, and nonnegative radius witness* o*. Conversely, any row of* k *with those carrier, transport, package, and local naming coordinates is accepted as a* $\mathsf{RationalBall}^{\uparrow}$ *containment read.*

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket containment exactness`

Proof. Project B through Definition 355.1. The only containment coordinate is the ledger k, and each row of k is read with the displayed center, radius, and order witness.

For the forward direction, a consumer route can only access a row named by the carrier. Transporting that row uses the componentwise hsame data in h, so the read remains inside the same finite rational-radius packet.

For the reverse direction, start with a displayed row of k together with c, r, o, h, p, and n. These are precisely the carrier coordinates needed to interpret the row as a rational containment read. □

Theorem 355.4 (RationalBall refinement transport)**.** *Suppose two accepted* $\mathsf{RationalBall}^{\uparrow}$ *packets are connected by displayed* hsame *rows for their centers, radius rows, nonnegative radius witnesses, and finite containment ledgers. Then any rational sample accepted by the first packet transports to a rational sample accepted by the second packet through the same* Cont *route surface. The transport introduces no completed metric point, no total real equality, and no open-cover refinement principle.*

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket radius refinement transport`

Proof. Use Definition 355.3 to identify the sample read as one displayed ledger row of the first packet. The hypotheses provide the matching hsame rows for the center, radius, order witness, and ledger.

Move the ledger row along those displayed transports. The package row keeps the same $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces visible to the consumer.

Apply the reverse direction of Definition 355.3. The transported row has exactly the coordinates required by the second packet, so it is accepted there and no external metric completion datum is introduced. □

Theorem 355.5 (RationalBall full packet refinement transport)**.** *Let an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet be given by center, radius, containment ledger, provenance, local name row, generated order row, transport row, endpoint row, a probe bundle, and a package row. If componentwise* hsame *transport is supplied for the center, radius, containment ledger, provenance, and local name row, and if the target side regenerates the* Cont *rows from center to radius to order, from order to containment to transport, and from transport to provenance to endpoint together with the matching* Pkg *row, then the transported data form an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet. The same construction also produces the derived order sameness, transport sameness, and endpoint sameness rows.*

The construction is internal to the displayed rational-ball packet: it introduces no completed metric point, compactness principle, open-cover refinement, or quotient-real equality.

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket refinement transport`

Proof. Project the accepted packet into its unary rows, three Cont routes, and package endpoint row. The displayed hsame hypotheses move the center, radius, containment ledger, provenance, and local name row componentwise.

Rebuild the generated order, transport, and endpoint rows using the target Cont routes. The BEDC kernel rule that Cont respects displayed sameness then gives the order sameness, transport sameness, and endpoint sameness rows.

Finally repack the transported unary rows, regenerated Cont routes, and target Pkg row into the accepted packet. Every object used in the construction is already one of the displayed packet coordinates or a regenerated route between them, so no completed metric point, compactness, open-cover, or quotient-real equality datum is introduced. □

Theorem 355.6 (RationalBall RealUp window handoff)**.** *A* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet may read only the finite rational-radius window recorded by the carrier: the* $\mathsf{Rat}^{\uparrow}$ *center, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the nonnegative radius witness, finite rational containment ledger, displayed* hsame *transports,* Cont *routes,* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row.*

The handoff supplies a finite rational-radius seed for real-name consumer windows. It does not assert completeness of $\mathsf{Real}^{\uparrow}$, *compactness of balls, or an open-cover topology theorem.*

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket realup window handoff`

Proof. Project the carrier of Definition 355.1. The center, radius, order witness, containment ledger, transport rows, provenance, and local naming row are the only coordinates exposed by the packet.

Use Definition 355.3 to exhaust the rational sample reads available to a downstream consumer. Use Definition 355.4 to see that every such read is stable under displayed finite refinements.

Thus a $\mathsf{Real}^{\uparrow}$ consumer receives exactly the finite rational-radius window and its provenance through the named BEDC surfaces. Since the packet has no coordinate for a completed real, an open cover, or compactness data, none can be exported at the handoff. □

Theorem 355.7 (RationalBall interval center containment)**.** *Let* $B = (c, r, o, h, k, p, n)$ *be an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet, and let* $I = (l, u, o_I, \kappa, h_I, c_I, p_I, n_I)$ *be a displayed* $\mathsf{RationalInterval}^{\uparrow}$ *window whose center row is the same* $\mathsf{Rat}^{\uparrow}$ *row* c *and whose endpoint rows are the rational-radius endpoints determined by* c *and* r*. For every rational sample row* q *carried by the same finite* BHist *surface,* q *is accepted by the* $\mathsf{RationalBall}^{\uparrow}$ *containment ledger* k *exactly when its transported row is accepted by the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint-containment ledger* κ *under the same displayed* hsame *transports,* Cont *routes, and* Pkg *provenance.*

The correspondence supplies only finite rational-radius containment data for the displayed center window. It does not assert metric completeness, compactness of balls, or any open-cover topology principle.

Proof. For the ball-to-interval direction, apply Definition 355.3. A containment read of q is one row of k interpreted against the center c, radius r, order witness o, transport rows h, continuation routes, package provenance, and local name row.

Transport that displayed row through the endpoint window. The shared $\mathsf{Rat}^{\uparrow}$ center and radius endpoints identify the transported sample with the endpoint-containment row of I. Apply Definition 355.4, Definition 354.3, and Definition 354.4 to read the row as one accepted $\mathsf{RationalInterval}^{\uparrow}$ containment fact.

For the interval-to-ball direction, start with the endpoint-containment row in κ. The same endpoint theorem and containment-ledger exactness identify it as a finite row over the displayed center, radius endpoints, order witness, hsame, Cont, and Pkg data. Repack that row through Definition 355.3; the carrier coordinates are exactly c, r, o, h, k, p, and n.

Finally use the RationalInterval consumer theorems named by the package provenance to keep the transported read inside the finite endpoint-window surface. No proof step introduces a completed metric point, a compact ball, or an open-cover datum. □

Theorem 355.8 (RationalBall carrier-radius obligation)**.** *Every admitted* $\mathsf{RationalBall}^{\uparrow}$ *packet consists of a* $\mathsf{Rat}^{\uparrow}$ *center history, a positive rational radius row, a containment classifier row over* BHist *observations, and* Pkg *provenance for the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surfaces.*

These are the only carrier-radius rows visible to downstream consumers.

Proof. Project the admitted packet through Definition 355.1. The carrier lists the center, radius, order witness, containment ledger, transport rows, package provenance, and local naming row.

Apply Definition 355.2. It isolates the center and radius rows as the finite data required by the local $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ surface.

Use Definition 355.3 to identify the containment classifier as the whole ledger over BHist observations. The remaining visible dependency rows are precisely the package coordinates named in the carrier. □

Theorem 355.9 (RationalBall center transport obligation)**.** *Transporting the center history and radius row along the displayed* $\mathsf{Rat}^{\uparrow}$ *classifier preserves the* $\mathsf{RationalBall}^{\uparrow}$ *containment classifier under componentwise* hsame *and* Cont *routes.*

The transported packet exposes the same finite rational-radius consumer surface and no completed metric point.

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket center transport obligation`

Proof. Begin with the center and radius rows supplied by Definition 355.1. The displayed $\mathsf{Rat}^{\uparrow}$ classifier provides the transport route for the center, and the radius row travels with the same finite rational-radius packet.

Apply Definition 355.4. It moves a containment sample through the matching hsame rows and the displayed Cont route without introducing an external metric completion datum.

Finally use Definition 355.3 in the target packet. The transported center and radius rows determine exactly the same containment classifier shape, so the $\mathsf{RationalBall}^{\uparrow}$ consumer surface is preserved. □

Theorem 355.10 (RationalBall ledger exactness obligation)**.** *The* $\mathsf{RationalBall}^{\uparrow}$ *ledger exposes exactly the center, radius, containment, refinement,* $\mathsf{Real}^{\uparrow}$ *window handoff, and dependency provenance rows of the finite rational-radius packet.*

It exposes no compactness theorem, open-cover topology, quotient real equality, or completed metric-space point.

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket ledger exactness certificate`

Proof. Project the ledger coordinates from Definition 355.1. The center and radius rows are the carrier entries, and the dependency provenance is the displayed Pkg row.

Apply Definition 355.2 to bind those rows to the local naming surface. Then use Definition 355.3 and Definition 355.4 to exhaust the containment and refinement reads.

Finally apply Definition 355.6. The only real-facing consumer data are the finite rational-radius window and its provenance, so compactness, open-cover topology, quotient real equality, and completed metric points have no ledger coordinate. □

Theorem 355.11 (RationalBall RegSeqRat window obligations)**.** *A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet reads exactly the* $\mathsf{Rat}^{\uparrow}$ *center,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, nonnegative-radius witness, finite containment ledger,* hsame *transports,* Cont *routes,* Pkg *provenance over* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row.*

The obligation surface exports no real completeness theorem, compactness of metric balls, open-cover topology, quotient real equality, or completed metric point.

Proof. Project the carrier rows from Definition 355.1. The visible data are the rational center, dyadic radius, nonnegative-radius witness, finite containment ledger, transport rows, continuation routes, package provenance, and local naming row.

Apply Definition 355.2 to bind these rows to the accepted $\mathsf{RationalBall}^{\uparrow}$ packet. Then use Definition 355.3 to exhaust the finite containment ledger.

Next apply Definition 355.4. It preserves the same rational-radius window under the displayed hsame and Cont routes, with package provenance over the rational-interval, regular-sequence, metric, and real-facing consumers.

Finally use Definition 355.6. The real-facing boundary is only the finite window read by the $\mathsf{RegSeqRat}^{\uparrow}$ consumer, so real completeness, compactness, open-cover topology, quotient real equality, and completed metric points have no exported row. □

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket regseqrat window obligations certificate`

Theorem 355.12 (RationalBall public finite consumer export)**.** *Every public* $\mathsf{RationalBall}^{\uparrow}$ *finite consumer read factors through exactly the center* $\mathsf{Rat}^{\uparrow}$ *row, the dyadic or rational radius row, the nonnegative-radius witness, the containment ledger, the* hsame *transports, the* Cont *routes, the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row. Conversely, those rows repack as the public finite rational-radius consumer surface.*

Proof. Project the carrier through Definition 355.1. The center row, radius row, nonnegative-radius witness, containment ledger, transport rows, continuation routes, package provenance, and local naming row are exactly the finite coordinates of an accepted packet.

Apply Definition 355.3 to exhaust the containment reads. Then use Definition 355.4 to carry the same rows through the displayed hsame and Cont routes.

Next apply Definition 355.6 and Definition 355.11. These two window statements identify the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads as the same finite rational-radius packet surface.

Thus the forward direction factors every public consumer read through the listed rows, and the reverse direction repacks those rows as the public consumer surface. Completeness, compactness, open-cover topology, quotient real equality, and completed metric points are excluded because the carrier has no coordinate for them. □

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket public finite consumer export`

Theorem 355.13 (RationalBall DyadicApproximation window correspondence)**.** *Let R be an accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet and let D be an accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *packet. Suppose the two packets share the unary precision row, the dyadic endpoint or radius row, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RationalInterval}^{\uparrow}$ *containment ledger, visible* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *rows naming those finite coordinates.*

Then R and D determine the same finite $\mathsf{Real}^{\uparrow}$*-facing consumer surface. Conversely, those shared rows repack as the* $\mathsf{RationalBall}^{\uparrow}$ *finite consumer read.*

Proof. Start with Definition 355.12. It exposes the $\mathsf{RationalBall}^{\uparrow}$ public read as the center row, dyadic or rational radius row, containment ledger, displayed transports, continuation routes, package provenance, and local naming row.

Next apply Definition 355.6. The $\mathsf{RationalBall}^{\uparrow}$ real-facing boundary is only the finite window consumed by the regular rational readback, so the displayed unary precision, stream name, and rational-interval containment rows are sufficient to identify the $\mathsf{Real}^{\uparrow}$ surface.

For the dyadic side, use Definition 335.14 and Definition 335.12. These identify the accepted $\mathsf{DyadicApproximation}^{\uparrow}$ packet with the same finite dyadic endpoint or radius row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{RationalInterval}^{\uparrow}$ containment ledger.

The shared hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows therefore give the same BHist consumer surface on both sides. Reading the surface from R gives the forward correspondence; reading the same rows in the reverse direction repacks the $\mathsf{RationalBall}^{\uparrow}$ finite consumer read.

No step uses compactness, open-cover topology, quotient real equality, a selected limit, or a completed metric point, since none of these is a row of the shared finite window. □

Theorem 355.14 (RationalBall standard finite-window bridge)**.** *The public finite consumer export,* $\mathsf{RegSeqRat}^{\uparrow}$ *window obligations,* $\mathsf{Real}^{\uparrow}$ *window handoff, refinement transport, and containment exactness assemble the standard finite rational-radius bridge for* $\mathsf{RationalBall}^{\uparrow}$*. The bridge exposes exactly the center row, radius row, nonnegative-radius witness, containment ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row.*

The bridge adds no real completeness, compactness of metric balls, open-cover topology, quotient real equality, or completed metric point.

Proof. Start with Definition 355.12. It fixes the public finite consumer rows: center, radius, nonnegative-radius witness, containment ledger, transports, continuation routes, package provenance, and local naming row.

Apply Definition 355.11 and Definition 355.6. These identify the regular rational source window and real-facing endpoint window as reads of the same finite rational-radius packet.

Next use Definition 355.4. It preserves the packet under the displayed hsame and Cont routes while retaining the same package provenance.

Finally apply Definition 355.3. The containment ledger is exhausted by the listed finite rows, so the assembled bridge has no coordinate for real completeness, compactness, open-cover topology, quotient real equality, or a completed metric point. □

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallUp StdBridge`

Theorem 355.15 (RationalBall mature finite consumer completeness)**.** *Every finite consumer read of the standard finite-window bridge for* $\mathsf{RationalBall}^{\uparrow}$ *factors through exactly the* $\mathsf{Rat}^{\uparrow}$ *center row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *radius row, the nonnegative-radius witness, the containment ledger, the* hsame *transports, the* Cont *routes, the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row. Conversely, those rows repack as the mature finite rational-radius consumer surface.*

This completeness statement exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, compactness of metric balls, open-cover topology, quotient real equality, or completed metric point.*

Proof. First apply Definition 355.12. It identifies every public finite consumer read with the center, radius, nonnegative-radius, containment, transport, continuation, package, and local naming rows of the accepted packet.

Next apply Definition 355.11 and Definition 355.6. These two results identify the regular rational source window and the real-facing endpoint window as reads of the same finite rational-radius packet.

Then use Definition 355.14. The bridge packages those public, regular-rational, and real-facing rows without adding a new consumer coordinate.

For the converse direction, repack exactly the listed rows as the mature finite rational-radius consumer surface. The excluded real-completeness, compactness, open-cover, quotient-equality, and completed-point principles remain absent because none is a coordinate of the bridge or of the public finite consumer export. □

Theorem 355.16 (RationalBall dyadic-radius readback)**.** *Every accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet whose radius row is displayed by a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance window gives a* $\mathsf{Real}^{\uparrow}$*-facing readback determined exactly by the* $\mathsf{Rat}^{\uparrow}$ *center row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the* $\mathsf{RationalInterval}^{\uparrow}$ *containment ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, displayed* hsame *transports, visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *ledger.*

The readback exports no metric completion point, open-ball compactness claim, open-cover topology, quotient real equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted packet to the finite carrier of Definition 355.1. In the dyadic-radius case, the consumer-visible coordinates are the rational center row, dyadic radius row, containment ledger, transport rows, continuation routes, package provenance, and local naming ledger.

Apply Definition 355.3. This identifies the center-radius containment row inside the displayed $\mathsf{RationalInterval}^{\uparrow}$ ledger and leaves no separate metric completion coordinate.

Use Definition 355.11 and Definition 355.6. These route the regular rational observation window and the $\mathsf{Real}^{\uparrow}$-facing read through the same finite dyadic-radius packet.

Finally invoke Definition 355.15. The mature consumer surface is exhausted by the center, dyadic radius, containment, observation, transport, continuation, package, and naming rows, so the stated readback contains none of the excluded completion, compactness, topology, quotient, or real-completeness data. □

Theorem 355.17 (RationalBall metric positive containment transport)**.** *An accepted* $\mathsf{RationalBall}^{\uparrow}$ *packet with nonnegative radius and finite containment ledger transports the same positive containment row through the* $\mathsf{Met}^{\uparrow}$ *distance consumer. The transported row exposes only the* $\mathsf{Rat}^{\uparrow}$ *center, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *radius, the finite containment ledger, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and the local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *row.*

The transport adds no real completeness, compactness of metric balls, open-cover topology, quotient real equality, or completed metric point.

Lean4 ✓ `BEDC.Derived.RationalBallUp.RationalBallPacket metric positive containment transport`

Proof. First use Definition 355.12. It exposes the finite center, radius, nonnegative-radius witness, containment ledger, transport rows, continuation routes, package provenance, and local naming row of the accepted packet.

Next apply Definition 355.11 and Definition 355.6. These obligations identify the regular rational source window and the real-facing endpoint window as reads of the same finite containment ledger.

Then invoke Definition 355.14. The bridge carries the public finite rows into the standard rational-radius window without adding a completeness, compactness, topology, quotient, or completed point coordinate.

Finally read the metric side through Definition 355.15. The $\mathsf{Met}^{\uparrow}$ distance consumer receives the same positive containment row, with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance, so the transport is exactly finite-window containment repacking. □

Theorem 355.18 (RationalBall nested containment consumer composition)**.** *Let $B_0 = (c_0, r_0, H_0, R_0, P_0, C_0)$ and $B_1 = (c_1, r_1, H_1, R_1, P_1, C_1)$ be accepted $\mathsf{RationalBall}^{\uparrow}$ packets, and let q be a query history. Suppose the displayed $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ refinement row says that the finite rational ball of B_1 is contained in the finite rational ball of B_0. If q is accepted by the containment row of B_1, then q is accepted by the containment row of B_0.*

Moreover every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{Met}^{\uparrow}$ consumer receives the composed finite containment ledger formed from the two packet ledgers, the displayed hsame transports, the Cont routes, the Pkg provenance over $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the two local $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ rows. The composition introduces no host metric, choice-selected point, real-completeness row, compactness datum, topology row, or quotient real equality.

Proof. First apply Definition 355.4. It transports the inner packet B_1 through the displayed outer containment row of B_0 and keeps the containment witness as a finite $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row.

Next use Definition 355.7. The query history q is already accepted by the containment row of B_1, and the transported refinement row places that accepted read inside the containment row of B_0.

For regular and real consumers, pass the composed ledger through Definition 355.11 and Definition 355.14. These theorems keep the same finite observation windows and standard rational-radius bridge rows, so the consumer sees the composed containment ledger rather than a rebuilt host argument.

Finally apply Definition 355.17. The $\mathsf{Met}^{\uparrow}$ read is the same positive containment row transported through the finite packet data. Since every step used only displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, no external metric structure or choice data enters the composed handoff. □

Theorem 355.19 (RationalBall DyadicApproximation nested window readback)**.** *Let $B_0, \ldots, B_n$ be nested accepted $\mathsf{RationalBall}^{\uparrow}$ packets with displayed inner-to-outer $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ containment rows. Then the composed containment ledger determines the same terminal $\mathsf{DyadicApproximation}^{\uparrow}$ finite window as the dyadic-approximation correspondence and the terminal refinement theorem.*

The terminal window is read from the innermost accepted packet, transported through the composed containment ledger, and presented with the same dyadic endpoint, precision row, scheduled

finite readback, $\mathsf{Real}^{\uparrow}$ *handoff, visible* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *rows. The readback introduces no host metric, choice-selected point, compactness datum, topology row, completed real, or quotient real equality.*

Proof. First apply Definition 355.13 to the terminal accepted rational-ball packet. This supplies the finite $\mathsf{DyadicApproximation}^{\uparrow}$ window with its precision, endpoint, scheduled readback, handoff ledger, transport, route, provenance, and local certificate rows.

Next use Definition 355.18. The inner-to-outer containment rows compose into a single finite ledger, and every regular, real, metric, or dyadic consumer sees that composed ledger rather than a rebuilt host argument.

Apply Definition 335.17 to identify the terminal dyadic window selected by the refinement chain. The terminal precision row and dyadic endpoint row are the same rows produced by the rational-ball correspondence, transported along the containment ledger.

Finally use Definition 335.19. It states that nested dyadic windows exhaust to the terminal finite window through the displayed refinement rows. Thus the rational-ball containment ledger and the dyadic terminal refinement expose the same finite readback surface, with no host metric, choice-selected point, compactness, topology, completed-real, or quotient-equality row. □

Theorem 355.20 (RationalBall nested containment refinement monotonicity)**.** *For any finite chain* $B_0, \ldots, B_n$ *of accepted* $\mathsf{RationalBall}^{\uparrow}$ *packets whose adjacent* $\mathsf{RationalInterval}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *containment rows witness that each displayed ball is contained in the preceding displayed ball, every query history accepted by the containment row of* B_n *is accepted by the containment row of every earlier* B_i*.*

Every $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Met}^{\uparrow}$ *consumer receives the iterated finite containment ledger assembled from the adjacent packet ledgers, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$ *rows. The finite chain exports no host metric, choice-selected point, compactness datum, topology row, completed real, or quotient real equality.*

Proof. Induct on the finite chain. For the singleton chain, the terminal packet is already the initial packet, and the containment read is the packet's displayed finite containment row.

For the extension step, assume the statement for a chain ending in B_n and append an accepted packet B_{n+1} whose adjacent containment row places it inside B_n. Apply Definition 355.18 to compose the accepted read of B_{n+1} with the containment row of B_n.

The induction hypothesis then transports the resulting accepted read through every earlier containment row. The composed ledger is the finite concatenation of the adjacent packet ledgers together with the displayed hsame, Cont, Pkg, and local NameCert rows used at each step.

For consumer handoff, use Definition 355.11, Definition 355.14, and Definition 355.17. These theorems read the same iterated containment ledger for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$, without adding any host metric, selected point, compactness, topology, completed-real, or quotient-equality row. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$**.**

Constructive story: A rational ball packet is generated by a rational center row, a dyadic or rational radius row, a nonnegative radius witness, a finite rational containment ledger, displayed transport rows, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 355.1, Definition 355.8, Definition 355.9, Definition 355.10, Definition 355.6, Definition 355.11, Definition 355.12, Definition 355.13, Definition 355.14, Definition 355.15, Definition 355.17, and Definition 355.18 bind the carrier definition, the carrier-radius, center-transport, ledger-exactness, $\mathsf{Real}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ window, public finite consumer rows, dyadic-approximation window correspondence, standard finite rational-radius bridge, mature finite consumer completeness, metric positive containment transport, and nested containment composition to

BHist, hsame, Cont, Pkg, NameCert, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ for the finite rational-radius packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBall}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RationalBallUp.RationalBallPacket\_refinement\_transport`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close real completeness, compactness of metric balls, open-cover topology, or quotient real equality.

Upgrade path: No further formal-axis upgrade remains because the finite rational-radius packet is bridge-checked.

Chapter 356

A Concrete Naming Certificate for $\mathsf{ContextFreeGrammar}^{\uparrow}$

The $\mathsf{ContextFreeGrammar}^{\uparrow}$ interface names a context-free grammar through a finite grammar packet rather than through a host language set. In the Hopcroft–Ullman presentation, a context-free grammar is given by a finite terminal alphabet, finite nonterminal carrier, start symbol, and finite production relation; here each of those rows is displayed as finite BHist data with $\mathsf{List}^{\uparrow}$ word rows and explicit provenance toward $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ consumers.

The yield and bridge boundary is consumed by chapter 351 for regular-language comparison and by chapter 280 for machine-recognizer boundary checks. Thus the grammar packet is read as a finite sibling surface in the language-recognition lattice, not only as a kernel-local carrier.

Definition 356.1 (ContextFreeGrammar BHist carrier)**.** *A* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet is a finite record*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

with the following displayed rows:

1. Σ *is the terminal alphabet row, exposed as finite* BHist *data;*
2. N *is the nonterminal carrier row, disjoint from the terminal rows by the displayed classifier surface;*
3. S *is the start nonterminal row inside* N*;*
4. P *is the finite production ledger, whose rows have one nonterminal left side and a* $\mathsf{List}^{\uparrow}$ *right side over terminal and nonterminal symbols;*
5. w *is the terminal yield word, displayed as a* $\mathsf{List}^{\uparrow}$ *row over* Σ*;*
6. d *is the finite derivation ledger, beginning at* S *and replacing one displayed nonterminal occurrence by a row of* P *at each step;*
7. y *is the readback row identifying the terminal fringe of the final sentential form with the word row* w*;*
8. h *stores componentwise* hsame *transports for alphabets, nonterminals, production rows, sentential forms, derivation rows, and yield rows;*
9. c *stores the* Cont *routes for reading a production row and continuing along the finite derivation ledger;*

10. *p stores* Pkg *provenance naming* List$^{\uparrow}$, RegularLanguage$^{\uparrow}$, *and* TuringMachine$^{\uparrow}$ *consumer surfaces;*

11. *n is the local* NameCert$_{\text{ContextFreeGrammar}^{\uparrow}}$ *registration row.*

The carrier predicate is ContextFreeGrammarPacket(G). *It exports only the displayed finite grammar, production, derivation, yield, transport, continuation, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarTasteGate single carrier alignment`

Theorem 356.2 (ContextFreeGrammar production stability)**.** *If two accepted* ContextFreeGrammar$^{\uparrow}$ *packets classify together by the displayed* hsame *rows, then every production row of the first packet transports to exactly one production row of the second packet with transported left nonterminal and transported* List$^{\uparrow}$ *right side. The transport preserves terminal and nonterminal tags and introduces no host grammar relation.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket production stability`

Proof. Project the packets through Definition 356.1. A production row is a visible entry of the finite ledger P with one nonterminal left side and one displayed right-side word.

Use the componentwise hsame rows in h to transport the left nonterminal, the terminal and nonterminal tags, and the List$^{\uparrow}$ right side. The Cont route in c records that the transported row is read as the same production position in the target packet.

The reverse read is the same finite ledger projection in the second packet. Since the carrier has no coordinate for a host production relation, only the displayed production rows can be transported. □

Theorem 356.3 (ContextFreeGrammar derivation concatenation)**.** *For an accepted* ContextFreeGrammar$^{\uparrow}$ *packet, two derivation ledger segments whose endpoint and start sentential forms agree by a displayed* hsame *row concatenate to a finite derivation ledger over the same production rows. The concatenated ledger preserves the* List$^{\uparrow}$ *sentential-form rows,* BHist *production witnesses,* Cont *routes, and* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket derivation concatenation`

Proof. Read the two finite derivation segments from the d coordinate of Definition 356.1. The connecting hypothesis identifies the terminal segment endpoint and the next segment start through a displayed sentential-form transport row.

Append the second ledger after the first. Each step still reads a production row of P, so Definition 356.2 transports the production witnesses along the same packet classifier when needed.

The composed Cont route first consumes the first segment and then the second segment. All rows remain inside the finite carrier, so no infinite derivation tree or host closure operation is introduced. □

Theorem 356.4 (ContextFreeGrammar yield readback)**.** *Every accepted terminal yield read of a* ContextFreeGrammar$^{\uparrow}$ *packet factors through the displayed start row S, the finite production ledger P, the finite derivation ledger d, and the* List$^{\uparrow}$ *terminal word row w. Conversely, a finite derivation ledger ending in a terminal sentential form with readback row y repacks as an accepted yield read of the same packet.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket yield readback`

Proof. For the forward direction, project the carrier rows. The yield coordinate is the readback row y, and it is attached to the final sentential form of the finite derivation ledger d.

Use Definition 356.2 at each derivation step to keep the production witness inside the displayed ledger. The terminal fringe is then read as the $\mathsf{List}^{\uparrow}$ word row w through the stored continuation route.

For the reverse direction, start with a derivation ledger whose final form has the displayed terminal readback row. These are exactly the carrier coordinates S, P, d, y, and w, so they repack as the accepted yield read. □

Theorem 356.5 (ContextFreeGrammar pushdown handoff)**.** *Every accepted terminal-yield read of a* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet can be handed to a* $\mathsf{PushdownAutomaton}^{\uparrow}$ *consumer by reading the start row S, the finite production ledger P, the finite derivation ledger d, the* $\mathsf{List}^{\uparrow}$ *sentential-form rows, and the terminal-yield readback y as stack-transition and run-ledger rows. The handoff exports no infinite parse tree, host language set, grammar quotient, or unbounded machine trace.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPushdownHandoff`

Proof. Project the grammar carrier of Definition 356.1. The start row S initializes the pushdown control and the stack row, while the production ledger P names exactly the finite rewrites that may be replayed by the stack-transition consumer.

Read the finite derivation ledger d step by step. At each derivation step, apply Definition 356.2 to keep the production witness inside the displayed packet. The unreduced sentential-form suffix is pushed into the stack row, and the rewritten prefix is recorded as the next run-ledger entry.

Finally route the terminal readback through Definition 356.4. The terminal word row w and readback row y become the accepted endpoint of the $\mathsf{PushdownAutomaton}^{\uparrow}$ run ledger, with the same hsame, Cont, and Pkg provenance.

Every object used in the handoff is a finite carrier coordinate. Since the grammar packet has no coordinate for an infinite parse tree, host language set, grammar quotient, or unbounded machine trace, the pushdown consumer cannot extract one through this route. □

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket pushdown handoff`

Theorem 356.6 (ContextFreeGrammar terminal-yield finite window)**.** *For an accepted* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

and a displayed finite derivation window selected by a $\mathsf{ProbeBundle}$*, suppose the selected production rows are stable under the packet's componentwise* hsame *rows, the selected derivation segments concatenate through their displayed sentential-form endpoints, and the final sentential form has the readback row y. Then every selected terminal row is read through the same* BHist *carrier,* hsame *transport,* Cont *route,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$ *ledger of G. No selected nonterminal production row is exposed as a terminal yield row.*

Proof. Begin with the selected production rows in the finite window. Apply Definition 356.2 to each selected production step, so the left nonterminal, the right-side $\mathsf{List}^{\uparrow}$ row, and the terminal and nonterminal tags remain inside the displayed production ledger P with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Next assemble the selected route. Apply Definition 356.3 to the displayed derivation segments whose endpoints agree by the packet's sentential-form hsame rows. The concatenated route is still a finite ledger over P, so it does not introduce a host grammar step or an unlisted production row.

Finally apply Definition 356.4 to the terminal sentential form and the readback row y. The readback projects only terminal rows through the displayed bundle and identifies them with the $\mathsf{List}^{\uparrow}$ word row w. Since nonterminal tags are preserved by production stability and the readback row is terminal-only, no nonterminal production row escapes as a terminal yield. □

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket terminal yield window`

Theorem 356.7 (ContextFreeGrammar NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation for the terminal alphabet, nonterminal carrier, start symbol, finite production ledger, terminal word, derivation ledger, readback row, transport rows, continuation routes, provenance, and local registration;*
2. *production stability under componentwise* hsame *transport by Definition 356.2;*
3. *derivation concatenation for finite ledger segments by Definition 356.3;*
4. *yield readback from the final terminal sentential form to the displayed* $\mathsf{List}^{\uparrow}$ *word row by Definition 356.4;*
5. *consumer boundary:* $\mathsf{RegularLanguage}^{\uparrow}$ *and* $\mathsf{TuringMachine}^{\uparrow}$ *readers receive only the displayed grammar packet, finite production and derivation ledgers, yield row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket namecert obligation surface`

Proof. Carrier habitation is exactly the packet of Definition 356.1. Its coordinates are finite grammar rows, a finite production ledger, a finite derivation ledger, and a terminal yield row.

The production and derivation rows are supplied by Definition 356.2 and Definition 356.3. Together they keep every derivation step inside the displayed finite ledger and preserve the transport and continuation routes.

The readback row is Definition 356.4. It identifies the terminal fringe with the displayed $\mathsf{List}^{\uparrow}$ word and repacks the same rows as the accepted yield read.

Finally project the provenance row p. It names the downstream $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ consumer surfaces without adding a host language quotient, parse-tree infinitary semantics, pumping theorem, or recognizer completeness claim. □

Theorem 356.8 (ContextFreeGrammar RegularLanguage–TuringMachine bridge boundary)**.** *The public grammar and yield packet of Definition 356.1 exports to* $\mathsf{RegularLanguage}^{\uparrow}$ *and* $\mathsf{TuringMachine}^{\uparrow}$ *consumers only the terminal alphabet, nonterminal carrier, start row, finite production ledger, finite derivation ledger,* $\mathsf{List}^{\uparrow}$ *terminal yield row,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

The exported boundary does not include ambiguity, parsing completeness, pumping arguments, Chomsky normal form, or recognizer completeness.

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket regularlanguage turingmachine bridge boundary`

Proof. Project the finite grammar packet of Definition 356.1. The terminal alphabet, nonterminal carrier, start row, production ledger, derivation ledger, terminal yield row, transport rows, continuation routes, provenance, and naming row are exactly the displayed carrier coordinates.

Use Definition 356.2 to keep every production witness inside the finite ledger under componentwise transport. Use Definition 356.3 to compose consumer derivation windows without introducing an infinitary derivation object.

Apply Definition 356.4 and Definition 356.6. These identify the consumer-visible yield as the same $\mathsf{List}^{\uparrow}$ terminal word row read through the finite derivation ledger and the stored terminal readback surface.

Finally apply Definition 356.7 and repack the rows named by the consumer boundary. Since the packet has no coordinate for ambiguity, parsing completeness, pumping arguments, Chomsky normal form, or a complete recognizer, none of those structures is exported to $\mathsf{RegularLanguage}^{\uparrow}$ or $\mathsf{TuringMachine}^{\uparrow}$ readers. □

Theorem 356.9 (ContextFreeGrammar regular-language inclusion certificate)**.** *Every accepted* $\mathsf{RegularLanguage}^{\uparrow}$ *finite-automaton packet yields a* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet whose production ledger simulates the deterministic transition ledger and whose terminal yield is the accepted* $\mathsf{List}^{\uparrow}$ *word. The construction reads only the finite alphabet row, state row, transition table, initial row, accepting endpoint row, input-word spine, deterministic run ledger, componentwise transport rows, continuation rows, and package provenance of chapter 351; it does not pass through a host language-set semantics or an ambient subset relation.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarRegularLanguageInclusionCertificate`

Proof. Start from the grammar carrier of Definition 356.1. For each visible automaton transition row in the deterministic run ledger of Definition 351.3, create the corresponding finite production row carrying the current state, consumed terminal, and next state as displayed nonterminal and terminal coordinates.

Use Definition 356.2 along the run ledger. Componentwise transport of the automaton rows therefore transports the simulated production rows without replacing them by an external language predicate.

Finally apply Definition 356.4. The derivation consumes exactly the stored $\mathsf{List}^{\uparrow}$ word spine, so the terminal yield row is the accepted word row named by the regular-language packet. Repacking the production ledger, derivation ledger, yield row, transports, continuation, provenance, and local name gives the required $\mathsf{ContextFreeGrammar}^{\uparrow}$ packet. □

Theorem 356.10 (ContextFreeGrammar consumer boundary determinacy)**.** *Let G and H be accepted finite* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packets. Suppose their terminal alphabet rows, nonterminal carrier rows, start rows, production ledgers, derivation ledgers, yield rows,* hsame *rows,* Cont *rows,* Pkg *provenance rows, and local* NameCert *rows agree componentwise. Then G and H induce the same consumer-visible boundary for both* $\mathsf{RegularLanguage}^{\uparrow}$ *and* $\mathsf{TuringMachine}^{\uparrow}$ *readers.*

The common boundary is the finite grammar and yield surface exported by Definition 356.8. It contains no grammar-equivalence quotient, ambiguity row, parsing-completeness row, pumping argument, Chomsky-normal-form witness, or recognizer-completeness coordinate.

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket consumer boundary determinacy`

Proof. Apply Definition 356.2 to the equal production ledgers. The production witnesses of G and H stay inside the same finite terminal and nonterminal rows under the displayed componentwise hsame transports.

Next apply Definition 356.3. The equal derivation ledgers concatenate through the same sentential-form endpoints and therefore expose the same finite derivation route to downstream consumers.

Apply Definition 356.4. Since the terminal yield rows agree, both packets read back to the same $\mathsf{List}^{\uparrow}$ terminal word through the same terminal-only surface.

Finally use Definition 356.8. That boundary is built only from the terminal alphabet, nonterminal carrier, start row, production ledger, derivation ledger, yield row, hsame, Cont, Pkg, and local NameCert rows. Componentwise equality of exactly those rows therefore forces the same $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ consumer-visible boundary, while the excluded grammar and recognizer structures have no exported coordinate. □

Theorem 356.11 (ContextFreeGrammar parse-tree derivation boundary)**.** *Every parse-tree-shaped consumer read of an accepted* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet factors through the finite production rows, derivation concatenation rows, terminal yield window,* $\mathsf{List}^{\uparrow}$ *word row,* $\mathsf{TuringMachine}^{\uparrow}$ *bridge boundary,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows of Definition 356.1. The read exports no infinitary parse-tree semantics, host language quotient, pumping theorem, or grammar-equivalence oracle.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket parse tree derivation boundary`

Proof. Project the accepted packet through Definition 356.1. A parse-tree-shaped consumer read can name productions, sentential-form steps, and terminal leaves only through the displayed finite production ledger, derivation ledger, readback row, transport rows, continuation routes, provenance, and local naming row.

Apply Definition 356.2 to keep each production read inside the finite production ledger. Apply Definition 356.3 to assemble the tree-shaped read as a finite derivation route whose subderivations meet at displayed sentential-form endpoints.

Next apply Definition 356.4 and Definition 356.6. These identify the consumer-visible leaves with the terminal yield window and the $\mathsf{List}^{\uparrow}$ word row, while preserving the same BHist, hsame, Cont, Pkg, and NameCert provenance.

Finally apply Definition 356.8 and Definition 356.10. The bridge exposes only the displayed grammar and yield surface to $\mathsf{TuringMachine}^{\uparrow}$ and $\mathsf{RegularLanguage}^{\uparrow}$ readers, and determinacy says that agreeing finite rows determine the same consumer-visible boundary.

Since no carrier coordinate stores an infinitary parse-tree semantics, host language quotient, pumping theorem, or grammar-equivalence oracle, none of those structures is exported by the parse-tree-shaped read. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$.

Constructive story: A context-free grammar packet is generated by finite terminal and nonterminal rows, a start row, finite production rows, a ListUp terminal word row, a finite derivation ledger, a terminal yield readback row, displayed transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 356.1, Definition 356.2, Definition 356.3, Definition 356.4, Definition 356.7, Definition 356.8, Definition 356.10, Definition 356.11, and Definition 356.12 bind the carrier, production and derivation rows, yield readback, grammar obligation surface, paper bridge boundary, consumer boundary determinacy, parse-tree-shaped consumer boundary, and finite derivation-prefix subledger treatment for the displayed grammar packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close grammar equivalence, ambiguity, parsing completeness, pumping arguments, Chomsky normal form, context-free language closure theorems, or machine bridge verification.

Upgrade path: Machine bridge verification requires a Lean bridge theorem for the displayed grammar packet and its regular-language and Turing-machine consumer surfaces.

Theorem 356.12 (ContextFreeGrammar derivation prefix carrier)**.** *Let*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

be an accepted $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet. Suppose the finite derivation ledger* d *is displayed as a split into a prefix ledger* d_{pre} *and a suffix ledger* d_{suf}*, with* d_{pre} *beginning at the start row* S *and ending at the split sentential-form row* u*. Then* d_{pre} *repacks as an accepted derivation-segment carrier*

$$(S, P, d_{\mathrm{pre}}, u, h_{\mathrm{pre}}, c_{\mathrm{pre}}, p, n)$$

over the same start row, production ledger, package provenance, and local $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$ *row. The restricted rows* h_{pre} *and* c_{pre} *are exactly the componentwise* hsame *transports and* Cont *routes of* h *and* c *whose derivation entries lie in the displayed prefix. The repacked segment contains no suffix step, no terminal-yield readback requirement, and no production row outside* P*.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket derivation prefix carrier`

Proof. Project the accepted packet through Definition 356.1. The coordinate d is a finite derivation ledger beginning at the displayed start row S, and each step of d replaces one displayed nonterminal occurrence by a production row already listed in the finite production ledger P.

The displayed split selects a finite initial subledger d_{pre}. Since it is an initial segment of d, its first sentential-form row is still S, and its last sentential-form row is the split row u. Every step retained in d_{pre} is therefore a step of the original ledger, so its production witness is already a row of P. Applying Definition 356.2 to these retained steps keeps the left nonterminal row, right-side $\mathsf{List}^{\uparrow}$ row, terminal and nonterminal tags, and production witness inside the same displayed production ledger.

Restrict the transport coordinate h to the alphabet, nonterminal, production, sentential-form, and derivation rows that occur in d_{pre}. This gives h_{pre}. Restrict the continuation coordinate c to the route entries that read those same prefix steps. This gives c_{pre}. Both restrictions are finite because the selected prefix is a finite subledger of the finite ledger d.

Now repack the rows S, P, d_{pre}, u, h_{pre}, c_{pre}, p, and n. They provide exactly the data needed for a finite derivation segment: a start row, the production ledger used at each step, the retained derivation rows, the endpoint row, the corresponding transports, the corresponding continuation routes, package provenance, and the local naming row. The yield row w and readback row y are not used, because the claim stops at the split sentential form and does not assert that u is terminal.

Finally, Definition 356.3 checks the boundary: concatenating the repacked prefix with the displayed suffix along the split sentential-form row recovers a finite derivation ledger over the same production rows. Thus the prefix repacking is a genuine subledger carrier of the accepted packet, not a separate grammar, parse tree, host closure, or recognizer-completeness assertion. □

Lemma 356.13 (ContextFreeGrammar BHist forgetful projection boundary)**.** *Let*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

be an accepted $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet, and put*

$$\partial_{\mathsf{BHist}}(G) = ((\Sigma, N, S, w), (P, d, y), h, c, p, n).$$

Every BHist*-level read of* G *visible to* $\mathsf{RegularLanguage}^{\uparrow}$ *or* $\mathsf{TuringMachine}^{\uparrow}$ *consumers factors through* $\partial_{\mathsf{BHist}}(G)$*. The projection retains the target carrier rows* Σ, N, S, w*, the production and derivation ledger rows* P, d, y*, the classifier transports* h*, the continuation routes* c*, the provenance*

row p, and the local $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$ *row n. It forgets only display labels outside those coordinates. In particular, the projection carries no ambiguity row, grammar-equivalence quotient, parse-tree semantics, pumping datum, Chomsky-normal-form witness, or recognizer-completeness coordinate.*

Proof. Project the accepted packet through Definition 356.1. The carrier definition lists exactly the finite grammar rows Σ, N, S, the production ledger P, the terminal word row w, the derivation ledger d, the readback row y, the transport row h, the continuation row c, the provenance row p, and the local naming row n. The displayed projection $\partial_{\mathsf{BHist}}(G)$ keeps all rows that a BHist consumer can inspect after dropping the richer presentation labels.

For production reads, apply Definition 356.2. A production witness is already a row of P, and its transported left nonterminal, right-side $\mathsf{List}^{\uparrow}$ word, terminal tags, and nonterminal tags are read only through the componentwise transports in h and the route entries in c. Thus forgetting the surrounding presentation name does not remove any production coordinate available to the consumer.

For derivation and yield reads, apply Definition 356.3, Definition 356.4, and Definition 356.6. These results force a consumer derivation to pass through the finite ledger d, the same production ledger P, the terminal readback row y, and the $\mathsf{List}^{\uparrow}$ terminal word row w, while preserving the displayed BHist, hsame, Cont, Pkg, and NameCert rows. All of these rows are coordinates of $\partial_{\mathsf{BHist}}(G)$.

The adjacent consumer chapters chapter 351 and chapter 280 receive the grammar packet only through the consumer boundary described in Definition 356.7 and Definition 356.8. That boundary exports precisely the displayed grammar and yield rows, transport rows, continuation routes, provenance, and local naming row retained by $\partial_{\mathsf{BHist}}(G)$.

Finally, Definition 356.10 says that componentwise agreement on those retained rows forces the same $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ consumer-visible boundary. Hence any row not present in $\partial_{\mathsf{BHist}}(G)$ cannot change a visible consumer read. Since the carrier has no coordinate for ambiguity, grammar equivalence, infinitary parse-tree semantics, pumping arguments, Chomsky normal form, or recognizer completeness, the forgetful projection cannot export any of those structures. □

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket bhist forgetful projection boundary`

Lemma 356.14 (ContextFreeGrammar BHist classifier alignment boundary)**.** *Let*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

and

$$G' = (\Sigma', N', S', P', w', d', y', h', c', p', n')$$

be accepted $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packets. Suppose the local packet classifier compares them through the displayed componentwise* hsame *rows on terminal alphabets, nonterminal carriers, start rows, production ledgers, terminal word rows, derivation ledgers, readback rows, continuation routes, package provenance, and local naming rows. Then every* BHist *classifier read of G handed to a* $\mathsf{RegularLanguage}^{\uparrow}$ *or* $\mathsf{TuringMachine}^{\uparrow}$ *consumer transports exactly the following displayed rows:*

1. *the grammar carrier rows Σ, N, S and their counterparts Σ', N', S';*
2. *each production row of P to the corresponding production row of P', with transported left nonterminal, terminal and nonterminal tags, and* $\mathsf{List}^{\uparrow}$ *right side;*

3. *each retained derivation row of d to the corresponding derivation row of d', with the same sentential-form endpoint comparison;*
4. *the terminal yield row w, the readback row y, the transport rows h, continuation routes c, package provenance p, and local* $\mathsf{NameCert}_{\mathsf{ContextFreeGrammar}^{\uparrow}}$ *row n, together with their counterparts.*

The same read forgets every row outside this list. In particular, it does not supply a $\mathsf{RegularLanguage}^{\uparrow}$ *state carrier, accepting-state carrier, transition table, deterministic run ledger, accepted endpoint, or a* $\mathsf{TuringMachine}^{\uparrow}$ *configuration trace, deterministic transition table, halted-repetition row, tape readback row, or bounded simulation row unless that row is separately displayed by the receiving consumer's own carrier. Thus the receiving classifier gate must re-read the grammar rows through Definition 356.1; it cannot treat a copied source excerpt as an additional transported coordinate.*

Lean4 ✓ `BEDC.Derived.ContextFreeGrammarUp.ContextFreeGrammarPacket bhist classifier alignment boundary`

Proof. Project both accepted packets through Definition 356.1. The carrier definition lists only the finite terminal alphabet, nonterminal carrier, start row, production ledger, terminal word, derivation ledger, readback row, transport row, continuation row, provenance row, and local naming row. Hence a local classifier comparison between G and G' has no coordinate other than those listed rows.

For production rows, apply Definition 356.2. A production witness is a visible entry of P, and transport carries it to one visible entry of P' with the transported left nonterminal and transported $\mathsf{List}^{\uparrow}$ right side. The same theorem preserves terminal and nonterminal tags and rules out any host production relation. Therefore the production part of the classifier alignment is exactly the rowwise movement from P to P'.

For derivation rows, apply Definition 356.3 to the displayed sentential-form endpoints and production witnesses. Every transported derivation step remains a finite ledger entry over the same transported production surface. Then apply Definition 356.4 and Definition 356.6. These force the consumer-visible terminal fringe to be the displayed $\mathsf{List}^{\uparrow}$ word row w and its transported counterpart w', read through y, y' and the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Now compare the receiving consumers. The $\mathsf{RegularLanguage}^{\uparrow}$ accepted-word classifier of Definition 351.2 is a classifier on a displayed finite-automaton packet: alphabet, state, transition, input-word, run, and accepting-endpoint rows. The grammar packet contributes only the terminal-yield word and the provenance route named by Definition 356.8; it does not contain the finite-automaton state, transition, run, or endpoint coordinates. Thus those rows cannot be transported from G by the grammar classifier.

Similarly, Definition 280.6 reads the $\mathsf{TuringMachine}^{\uparrow}$ source classifier from finite state and tape alphabets, unary time and head-position rows, a $\mathsf{Seq}^{\uparrow}$ configuration trace, deterministic transition data, and bounded $\mathsf{Computable}^{\uparrow}$ trace rows. The context-free grammar carrier contains none of those machine-source coordinates. Its bridge contribution is only the displayed grammar and yield surface identified by Definition 356.8.

Finally apply Definition 356.10. Componentwise agreement on the retained grammar, production, derivation, yield, transport, continuation, provenance, and local naming rows determines the same $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ consumer-visible boundary. Since all other candidate rows are absent from Definition 356.1, they are forgotten by the BHist classifier projection and cannot be recovered by quoting the source presentation rather than re-reading the displayed carrier rows. □

Theorem 356.15 (ContextFreeGrammar displayed implication completion boundary)**.** *Let*

$$G = (\Sigma, N, S, P, w, d, y, h, c, p, n)$$

be an accepted $\mathsf{ContextFreeGrammar}^{\uparrow}$ *packet. Every displayed forward classifier or readback implication named in the local grammar chapter is completed in exactly one of the following two ways.*

1. *Production transport is two-sided on the visible production rows. A production row transported from one accepted packet to a classified mate is read back as the same finite ledger position with transported left nonterminal, transported* $\mathsf{List}^{\uparrow}$ *right side, and the same terminal and nonterminal tag rows.*
2. *Derivation concatenation is two-sided only at a displayed split row. Two finite derivation segments concatenate through an* hsame *endpoint row, and a ledger with a displayed prefix–suffix split repacks its prefix as the carrier of Definition 356.12; no arbitrary factorization is inferred without such a split row.*
3. *Yield readback is two-sided on the visible rows S, P, d, y, and w. An accepted yield read factors through those rows, and a finite derivation ledger ending in a terminal sentential form with readback row y repacks as the same accepted yield read.*
4. *Terminal-window, consumer-boundary, parse-tree-shaped, and forgetful* BHist *projection reads are one-way interpretation boundaries. They factor a consumer read through the displayed finite carrier, transport, continuation, provenance, and naming rows, but they do not reconstruct grammar equivalence, ambiguity, parsing completeness, pumping data, Chomsky normal form, infinitary parse-tree semantics, or recognizer completeness.*

Thus the chapter has no unrecorded reverse implication: each named forward surface either has its reverse on the same visible rows or is explicitly bounded as a one-way interpretation of the displayed BHist *packet.*

Proof. Project G through Definition 356.1. The only available coordinates are the terminal alphabet row Σ, nonterminal carrier row N, start row S, finite production ledger P, terminal word row w, derivation ledger d, readback row y, transport row h, continuation row c, provenance row p, and local naming row n.

For production rows, use Definition 356.2. Its forward direction transports a displayed production row along the componentwise hsame rows in h. Its readback direction is the same finite ledger projection in the target packet. Since the carrier has no coordinate for a host production relation, the reverse can only return to the visible production row with the transported left nonterminal, transported $\mathsf{List}^{\uparrow}$ right side, and preserved terminal and nonterminal tags.

For derivation rows, use Definition 356.3 together with Definition 356.12. Concatenation gives the forward implication from two displayed segments and an hsame endpoint row to a finite derivation ledger over P. The reverse implication is available exactly when the ledger already displays a prefix–suffix split: the prefix repackages over the same start row, production ledger, restricted transports, restricted continuation routes, provenance, and local NameCert row. Without a displayed split row, there is no carrier coordinate naming an arbitrary factorization, so the reverse stops at this boundary.

For yield rows, apply Definition 356.4. The forward implication factors an accepted terminal-yield read through the start row, finite production ledger, finite derivation ledger, terminal readback row, and $\mathsf{List}^{\uparrow}$ word row. The converse in the same theorem starts from exactly those displayed

rows and repacks them as the accepted yield read of the same packet. This is a genuine two-sided completion because both directions use the same visible rows S, P, d, y, and w.

For selected terminal windows, use Definition 356.6. It sends a displayed finite derivation window to terminal rows read through the same BHist, hsame, Cont, Pkg, and NameCert provenance. The theorem also records the obstruction to reversal: a selected terminal row exposes no selected nonterminal production row as a terminal yield row. Thus terminal-window reading is a one-way interpretation unless the carrier already supplies the production and derivation window.

For downstream consumers, use Definition 356.7, Definition 356.8, Definition 356.10, Definition 356.11, and Definition 356.13. These results all project the same displayed grammar and yield surface to $\mathsf{RegularLanguage}^{\uparrow}$ and $\mathsf{TuringMachine}^{\uparrow}$ readers. Determinacy says that componentwise agreement on the exported rows forces the same consumer-visible boundary, but none of those rows is a host language quotient, grammar comparison oracle, parse-tree semantics, pumping datum, Chomsky-normal-form witness, ambiguity classifier, or complete recognizer.

The displayed implications have therefore been exhausted by the carrier rows. Production transport and yield readback have reverses on the same visible rows; derivation concatenation has a reverse only at displayed split rows; and the terminal-window, consumer-boundary, parse-tree-shaped, and forgetful BHist reads are recorded as one-way interpretation boundaries. No further reverse implication can be formed inside the chapter without adding a coordinate absent from Definition 356.1. □

Chapter 357

A Concrete Naming Certificate for $\mathsf{ValidatedNumerics}^{\uparrow}$

The $\mathsf{ValidatedNumerics}^{\uparrow}$ interface names certified finite real enclosures in the interval-analysis tradition of Moore, but only through BEDC-visible finite data. A packet carries displayed rational or dyadic endpoints, a requested precision row, a regular rational observation window, a real-name readback token, and a containment ledger whose reads are routed by Cont. It is therefore a consumer of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{ModulusOfConvergence}^{\uparrow}$, not a theorem about host floating-point arithmetic or external numerical solvers.

The precision-to-tail portion of the validated enclosure is consumed through chapter 379. The $\mathsf{DyadicTailBound}^{\uparrow}$ packet supplies the dyadic tolerance, $\mathsf{CauchyModulus}^{\uparrow}$ schedule, finite tail ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal row that $\mathsf{ValidatedNumerics}^{\uparrow}$ reuses as its containment ledger.

Definition 357.1 (ValidatedNumerics BHist source)**.** *A* $\mathsf{ValidatedNumerics}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$$

with the following visible rows:

1. *I is a* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure row whose endpoints are read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *ε is the requested precision row;*
3. *ω is a* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row supplying the finite precision-to-window request used by the packet;*
4. *ρ is a* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window containing the displayed rational approximants inspected at that request;*
5. *λ is a* $\mathsf{Real}^{\uparrow}$ *readback token for the real name being observed through the finite window;*
6. *h records the componentwise* hsame *transports for the enclosure, precision, modulus, observation, readback, and containment rows;*
7. *c records the* Cont *routes by which each displayed approximation is read inside the enclosure row I;*
8. *p is the package provenance naming* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{ModulusOfConvergence}^{\uparrow}$*;*
9. *n is the local* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *registration row.*

The carrier predicate is ValidatedNumericsPacket$(I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$*. Its classifier compares two accepted packets only by transporting these displayed rows and replaying the finite containment routes.*

Lean4 (definition) `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsUp`
Lean4 (definition) `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket`

Theorem 357.2 (ValidatedNumerics interval enclosure obligation)**.** *Every accepted* ValidatedNumerics$^{\uparrow}$ *packet exposes one finite* RationalInterval$^{\uparrow}$ *enclosure row I and a containment ledger saying that each displayed approximation read from the* RegSeqRat$^{\uparrow}$ *window ρ remains inside I along a named* Cont *route. The obligation surface is the enclosure row, endpoint provenance through* DyadicRatCore$^{\uparrow}$ *or* Rat$^{\uparrow}$*, the precision request, the modulus row, the observation window, and the containment ledger routed by c and transported by h.*

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket interval enclosure obligation`

Proof. Project the carrier of Definition 357.1. The only interval coordinate is I, and the only routes from displayed approximations to interval containment are the Cont rows stored in c.

The endpoint source is read from the package provenance p, while ε, ω, and ρ give the precision request, modulus row, and regular rational observation window. Transporting the packet uses h on exactly those rows, so the accepted enclosure obligation is exhausted by the finite packet coordinates. □

Theorem 357.3 (ValidatedNumerics real readback soundness)**.** *For an accepted* ValidatedNumerics$^{\uparrow}$ *packet $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$, the* Real$^{\uparrow}$ *readback token λ is sound only through the displayed* RegSeqRat$^{\uparrow}$ *window ρ and the precision row supplied by* ModulusOfConvergence$^{\uparrow}$*. A consumer may read the real name as a finite validated enclosure at the requested precision, but the packet exports no general convergence proof, no classical real completeness, and no equality of completed real representatives.*

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsReadbackPacket real readback soundness`

Proof. Read λ together with ρ and ω from the same carrier. The token is not interpreted in isolation: the only available precision request is ε, and the only finite observations attached to that request are the displayed rows of ρ.

The containment routes from Definition 357.2 then place each displayed approximation inside I. Repacking λ with these window and containment rows gives the finite readback soundness statement, while no carrier coordinate remains for completeness or completed-real equality. □

Theorem 357.4 (ValidatedNumerics finite-window Real seal containment)**.** *Let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$ be an accepted* ValidatedNumerics$^{\uparrow}$ *packet. Any finite-window* Real$^{\uparrow}$ *seal read whose source is routed through the displayed* RegSeqRat$^{\uparrow}$ *window ρ, the* ModulusOfConvergence$^{\uparrow}$ *row ω, and the packet* Cont *containment ledger lands in the same* RationalInterval$^{\uparrow}$ *enclosure row I.*

The conclusion is a finite enclosure statement over the packet surface. It does not assert solver correctness, floating-point semantics, classical real completeness, or equality of completed representatives.

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket finite window seal containment`

Proof. Project the accepted packet. By Definition 357.2, the only enclosure exported by the carrier is I, and the only routes from displayed $\mathsf{RegSeqRat}^{\uparrow}$ observations into an interval are the Cont ledger rows stored in c.

Next read the $\mathsf{Real}^{\uparrow}$ token through Definition 357.3. The readback is available only with the finite window ρ and the modulus row ω, so the seal has no independent completed-real interface through which it could choose a second enclosure.

The $\mathsf{StreamName}^{\uparrow}$ finite-window handoff and the $\mathsf{RegSeqRat}$ finite-window $\mathsf{Real}^{\uparrow}$ seal interface therefore expose the same consumer rows: the window ρ, the modulus row ω, the transported hsame rows, and the Cont ledger. Repacking those rows gives containment in I, and the packet contains no additional consumer row for solver semantics, hardware floating-point interpretation, completeness, or equality of completions. □

Theorem 357.5 (ValidatedNumerics precision refinement containment)**.** *Let $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$ be an accepted $\mathsf{ValidatedNumerics}^{\uparrow}$ packet. Suppose V' refines the requested precision row to a smaller finite request while preserving the same $\mathsf{RationalInterval}^{\uparrow}$ enclosure I, the $\mathsf{ModulusOfConvergence}^{\uparrow}$ row, the $\mathsf{RegSeqRat}^{\uparrow}$ observation window, the $\mathsf{Real}^{\uparrow}$ readback token, the displayed hsame transports, and the Cont containment routes. Then every approximation read from the refined request is still contained in the same finite enclosure row I.*

The refinement is entirely internal to the displayed packet surface. It introduces no solver-correctness theorem, no floating-point hardware semantics, and no $\mathsf{Real}^{\uparrow}$ completeness datum beyond the readback token already carried by V.

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsReadbackPacket precision refinement containment`

Proof. Project the packet source using Definition 357.1. The refined request changes only the finite precision row; the enclosure I, modulus row, observation window, readback token, transport rows, containment routes, package provenance, and local name row remain the displayed coordinates of the packet.

Apply Definition 357.2. Since the same Cont containment routes are preserved, each refined approximation read is routed back into the same finite enclosure row I.

Next apply Definition 357.3. The $\mathsf{Real}^{\uparrow}$ token stays tied to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ window and the finite modulus request, so refinement does not create a completed real object or an external convergence principle.

Finally use Definition 357.7. Solver correctness, floating-point semantics, and classical completeness are not packet rows, hence they cannot enter the containment proof for the refined read. □

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket precision refinement containment`

Theorem 357.6 (ValidatedNumerics five-row NameCert sketch)**.** *The seed $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ surface is assembled from five visible dependency rows:*

1. *a $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint or precision budget row;*
2. *a $\mathsf{RationalInterval}^{\uparrow}$ enclosure row;*
3. *a $\mathsf{ModulusOfConvergence}^{\uparrow}$ precision-to-window row;*
4. *a $\mathsf{RegSeqRat}^{\uparrow}$ observation-window row;*
5. *a $\mathsf{Real}^{\uparrow}$ readback row.*

Together with the containment ledger, hsame *transports,* Cont *routes,* Pkg *provenance, and local name row, these rows form the finite validated numerics naming certificate.*

Proof. Start from the packet coordinates in Definition 357.1. The five listed rows are exactly the dependency-bearing coordinates: endpoints or precision budget from I and p, the interval row I, the modulus row ω, the observation row ρ, and the readback token λ.

The remaining coordinates are structural. The transport row h preserves the displayed rows under classifier movement, c routes containment reads, p records the package provenance, and n registers the local name. Repacking these coordinates gives the seed naming-certificate surface without adding a host numerical-analysis theorem. □

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket five row namecert sketch`

Theorem 357.7 (ValidatedNumerics dependency boundary)**.** *The dependency boundary of* ValidatedNumerics$^{\uparrow}$ *is precisely* DyadicRatCore$^{\uparrow}$, RationalInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and* ModulusOfConvergence$^{\uparrow}$, *together with the BEDC kernel rows* BHist, hsame, Cont, Pkg, *and* NameCert. *The packet does not depend on floating-point hardware semantics, a solver-correctness theorem, a classical completeness principle for real numbers, or a global convergence claim beyond the displayed finite modulus request.*

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsKernel dependency boundary`

Proof. Inspect the package row p of Definition 357.1. It names exactly the five concrete dependency surfaces listed in the statement, and every consumer read passes through the kernel rows h, c, p, and n.

The interval enclosure obligation and real readback soundness theorems use no other coordinate. Since floating-point execution, solver correctness, classical completeness, and global convergence are not rows of the finite packet, they are outside the dependency boundary. □

Theorem 357.8 (ValidatedNumerics carrier and classifier obligations)**.** *The nine displayed rows of Definition 357.1 supply the local carrier and classifier obligations for* ValidatedNumerics$^{\uparrow}$. *The carrier is inhabited by any finite packet whose interval, precision, modulus, observation, readback, transport, containment, package, and local name rows are displayed. Its classifier compares accepted packets componentwise through the interval row, precision row, modulus request, regular rational window, real readback token, containment routes,* hsame *transport,* Pkg *provenance, and local* NameCert *row, and carrier transport is exactly transport of these nine rows.*

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket carrier classifier obligations`

Proof. Read the packet definition row by row. Inhabitation is the assertion that the displayed finite tuple $V = (I, \varepsilon, \omega, \rho, \lambda, h, c, p, n)$ is present as a BHist packet, so no additional analytic carrier is required.

For classifier laws, compare two accepted packets only through the nine coordinates listed in the definition. The interval, precision, modulus, observation, and readback entries are component rows, while h, c, p, and n record the transport, continuation, package, and local naming data used by the comparison.

Carrier transport is therefore componentwise. Moving an accepted packet along the displayed hsame rows and replaying the same Cont containment routes keeps the tuple inside the carrier, with the package and local name rows recording the same source surface. □

Theorem 357.9 (ValidatedNumerics containment ledger transport)**.** *Interval containment ledgers in an accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *packet are preserved by the displayed* hsame *and* Cont *rows. If the enclosure row, precision request, modulus row, regular rational observation window, and* $\mathsf{Real}^{\uparrow}$ *readback token are transported componentwise, then every containment read from the transported observation window lands in the transported* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure through the transported* Cont *ledger.*

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket containment ledger transport`

Proof. Start with Definition 357.2. It identifies the containment ledger as the finite list of Cont routes from displayed approximations in ρ into the interval row I.

Transport the packet componentwise using the h rows of Definition 357.1. The source approximation row, the target interval row, the precision request, the modulus row, and the real readback token move together, so the containment source and target remain matched.

Finally replay the transported c rows. Each ledger entry is a displayed continuation route, hence it remains a continuation route after transport and preserves the same finite containment claim inside the transported enclosure. □

Theorem 357.10 (ValidatedNumerics refined readback containment determinacy)**.** *Let V and V' be accepted* $\mathsf{ValidatedNumerics}^{\uparrow}$ *packets with the same transported interval row, precision request, modulus request,* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window,* $\mathsf{Real}^{\uparrow}$ *readback token, containment ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. After any displayed precision refinement of the request, the two packets yield the same public finite-enclosure read.*

The determinacy is only a finite readback boundary. It does not assert solver correctness, hardware semantics, completed-real equality, or classical completeness.

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket refined readback containment determinacy`

Proof. First apply Definition 357.3. The common $\mathsf{Real}^{\uparrow}$ token is read through the same $\mathsf{RegSeqRat}^{\uparrow}$ observation window and finite modulus request, so both packets expose the same real-name readback row.

Next use Definition 357.5. A displayed precision refinement narrows the requested finite enclosure by the same transported interval row and the same modulus request on both sides.

Finally apply Definition 357.9. The containment ledger, hsame transports, Cont routes, Pkg provenance, and local NameCert row are shared, so the refined containment read is transported identically for V and V'. Therefore the public finite-enclosure read after refinement is the same, while solver correctness, hardware semantics, completed-real equality, and classical completeness remain outside the packet. □

Theorem 357.11 (ValidatedNumerics NameCert obligation surface)**.** *The* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *obligation surface is assembled from the carrier and classifier obligations of Definition 357.8, the interval enclosure obligation of Definition 357.2, the real readback soundness of Definition 357.3, precision refinement containment from Definition 357.5, containment ledger transport from Definition 357.9, package provenance, and the local* NameCert *row.*

This obligation surface is finite over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *and it does not assert solver correctness, floating-point hardware semantics, host real equality, or classical real completeness.*

Proof. First assemble the carrier and classifier rows from Definition 357.8. These rows bind the accepted packet to the nine displayed coordinates of the source definition.

Next attach the enclosure and readback rows. Apply Definition 357.2 for the interval ledger and Definition 357.3 for the $\mathsf{Real}^{\uparrow}$ token read through the $\mathsf{RegSeqRat}^{\uparrow}$ window and finite modulus request.

The refinement and transport rows are supplied by Definition 357.5 and Definition 357.9. They keep the finite enclosure claim stable under smaller precision requests and displayed hsame/Cont movement.

Finally add the package provenance and local naming row from Definition 357.1. Together with Definition 357.7, this exhausts the local obligation surface and leaves solver correctness, floating-point semantics, host real equality, and classical completeness outside the packet. □

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket namecert obligation surface`

Theorem 357.12 (ValidatedNumerics obligation closure scope)**.** *The finite enclosure packet obligations for* $\mathsf{ValidatedNumerics}^{\uparrow}$ *are exhausted by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *endpoint row, the* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure row, the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *precision-to-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the* $\mathsf{Real}^{\uparrow}$ *readback row, the containment ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *rows.*

No obligation row in this scope asserts host numerical-solver correctness, floating-point hardware semantics, completed-real equality, or a global convergence theorem beyond the displayed finite modulus request.

Proof. Project the packet source of Definition 357.1. Its displayed coordinates are the rational or dyadic endpoint data, the enclosure interval, the precision request, the modulus-to-window row, the finite observation window, the readback token, containment routes, transport row, package provenance, and local name row.

Apply Definition 357.2. This identifies the containment part of the obligation surface as the finite enclosure row together with the displayed continuation ledger.

Next apply Definition 357.3. The readback obligation is tied to the $\mathsf{RegSeqRat}^{\uparrow}$ observation window and the finite modulus request, so it contributes no completed-real equality or external convergence principle.

Finally use Definition 357.6 and Definition 357.7. The five dependency rows and the BEDC kernel rows listed there exhaust the local naming surface, while solver correctness, floating-point hardware semantics, classical completeness, and global convergence remain outside the packet boundary. □

Theorem 357.13 (ValidatedNumerics exported enclosure bridge)**.** *The public* $\mathsf{ValidatedNumerics}^{\uparrow}$ *export bridges a finite validated enclosure packet to exactly the displayed boundary consisting of the* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure, the dyadic or rational endpoint row, the finite modulus request, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the* $\mathsf{Real}^{\uparrow}$ *readback token, the containment ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

The bridge does not export solver correctness, floating-point hardware semantics, host real equality, or classical completeness.

Proof. Start from the finite enclosure obligation. Apply Definition 357.2; this fixes the $\mathsf{RationalInterval}^{\uparrow}$ enclosure, endpoint row, finite modulus request, and continuation ledger as the carrier side of the public packet.

Next attach the real readback boundary. Apply Definition 357.3; the $\mathsf{Real}^{\uparrow}$ token is read only through the $\mathsf{RegSeqRat}^{\uparrow}$ observation window and finite modulus request, so no host real equality is introduced.

The exported packet must remain stable under sharper finite requests. Apply Definition 357.5 and Definition 357.9; refinement narrows the interval while the hidden containment ledger is carried by the displayed hsame transports and Cont routes.

Finally apply Definition 357.11. The local NameCert row and Pkg provenance repack the preceding rows as the public bridge boundary. Since each step projects from the displayed finite packet, solver correctness, floating-point hardware semantics, host real equality, and classical completeness remain outside the bridge. □

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsFiniteEnclosure exported bridge`

Theorem 357.14 (ValidatedNumerics public finite-enclosure consumer certificate)**.** *A scoped* $\mathsf{ValidatedNumerics}^{\uparrow}$ *packet exports a public finite-enclosure consumer certificate. The interval enclosure obligation gives the carrier row, real readback soundness gives the* $\mathsf{Real}^{\uparrow}$ *interpretation, precision refinement containment supplies monotone narrowing, and containment ledger transport records the hidden refinement ledger. Downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers may use the enclosure packet without assuming any host real-number primitive.*

Proof. First build the finite enclosure source. Apply Definition 357.2; the displayed endpoint and interval rows provide the carrier-side enclosure data, while the continuation routes record the finite containment obligation.

Next interpret the enclosure as a real readback packet. By Definition 357.3, the readback is routed through the $\mathsf{RegSeqRat}^{\uparrow}$ observation window, the finite modulus request, and the named $\mathsf{Real}^{\uparrow}$ row, so the consumer sees a BEDC real-name interpretation rather than a host completed-real primitive.

The public certificate must also be stable under tighter requests. Apply Definition 357.5; smaller precision demands retain the enclosing row by monotone narrowing and preserve the same displayed transport surface.

Finally apply Definition 357.9. The hidden refinement ledger is carried as Cont/hsame provenance, not as a solver oracle. Repacking these four rows gives a public finite-enclosure consumer certificate usable by downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers, with solver correctness, floating-point semantics, and classical real completeness remaining outside the exported packet. □

Theorem 357.15 (ValidatedNumerics validated enclosure bridge boundary)**.** *The public finite-enclosure consumer certificate of Definition 357.14 and the* NameCert *obligation surface of Definition 357.11 assemble a paper bridge boundary whose readable rows are exactly the* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *endpoint provenance, the finite* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *request, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window, the* $\mathsf{Real}^{\uparrow}$ *readback token, the containment ledger, the* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

Solver correctness, floating-point hardware semantics, host real equality, and classical real completeness are not bridge rows.

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsUp StdBridge`

Proof. Begin with the interval enclosure of Definition 357.2. It supplies the $\mathsf{RationalInterval}^{\uparrow}$ enclosure together with dyadic or rational endpoint provenance and the finite precision request visible to the consumer.

Next apply Definition 357.3. The readback token is interpreted through the $\mathsf{RegSeqRat}^{\uparrow}$ observation window and the finite $\mathsf{ModulusOfConvergence}^{\uparrow}$ request, so the row remains a displayed BEDC real-name read rather than host real equality.

Use Definition 357.5 and Definition 357.9 for the boundary rows that survive sharper requests. They identify the containment ledger and transport it through the displayed hsame and Cont surface.

Finally apply Definition 357.11 and Definition 357.14. The NameCert surface supplies the local name and Pkg provenance, and the consumer certificate repacks exactly those rows as the conservative bridge boundary. Since every bridge row is projected from this finite packet, solver correctness, floating-point hardware semantics, host real equality, and classical real completeness remain outside the bridge boundary. □

Theorem 357.16 (ValidatedNumerics finite consumer example completeness)**.** *Every finite consumer example using the displayed validated-enclosure bridge of Definition 357.15 factors through the* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure row, dyadic-or-rational endpoint provenance, finite* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *request,* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window,* $\mathsf{Real}^{\uparrow}$ *readback token, containment ledger, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$ *row.*

Conversely, those displayed rows repack as a valid finite consumer example for the validated-enclosure bridge. The repacking preserves the exclusions on solver correctness, floating-point semantics, host real equality, and classical real completeness.

Proof. Begin with a finite consumer example and apply Definition 357.14. The certificate projects the example to the rational-interval enclosure, endpoint provenance, finite precision request, regular rational observation window, real readback token, containment ledger, transport rows, package provenance, and local naming row.

Next apply Definition 357.15. It identifies the same rows as the bridge boundary visible to downstream consumers, so no solver, hardware, host-real, or classical-completeness coordinate is introduced by the bridge read.

Use Definition 357.5 to keep the enclosure stable under finite precision requests. Then apply Definition 357.9 to carry the containment ledger through the displayed hsame and Cont surface.

For the converse direction, start from the listed displayed rows. Repack the enclosure row, endpoint provenance, finite modulus request, observation window, readback token, containment ledger, transport rows, package provenance, and local name row by the public finite-enclosure consumer certificate. The validated-enclosure bridge boundary then reads exactly that packet as a valid finite consumer example, with the same excluded semantic obligations remaining outside the exported surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ValidatedNumerics}^{\uparrow}$ packet is generated by a rational or dyadic interval enclosure, a requested precision row, a finite modulus-of-convergence request, a regular rational observation window, a real-name readback token, containment routes, transport rows, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 357.15 and Definition 357.14 bind the finite certified-enclosure bridge surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsReadbackPacket\_real\_readback\_soundness`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close general convergence, floating-point hardware semantics, classical real completeness, or solver correctness.

Upgrade path: No further upgrade remains; the chapter is at the top of the formal axis.

Theorem 357.17 (ValidatedNumerics refinement composition containment)**.** *Let*

$$V_0 = (I, \varepsilon_0, \omega, \rho, \lambda, h, c, p_0, n_0),$$
$$V_1 = (I, \varepsilon_1, \omega, \rho, \lambda, h, c, p_1, n_1),$$
$$V_2 = (I, \varepsilon_2, \omega, \rho, \lambda, h, c, p_2, n_2)$$

be accepted $\mathsf{ValidatedNumerics}^{\uparrow}$ *packets. Suppose* V_1 *refines the requested precision row of* V_0 *to a smaller finite request, and* V_2 *refines the requested precision row of* V_1 *to a smaller finite request. Assume that both refinement steps preserve the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *enclosure* I*, the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row* ω*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window* ρ*, the* $\mathsf{Real}^{\uparrow}$ *readback token* λ*, the* hsame *transport row* h*, and the* Cont *containment ledger* c*. Then the composite refinement from* V_0 *to* V_2 *routes every displayed approximation read at the terminal request into the original finite enclosure row* I *through the original containment ledger* c*.*

The conclusion is a containment statement over the displayed finite packet surface. It does not add solver correctness, floating-point hardware semantics, completed-real equality, or a global convergence principle.

Lean4 ✓ `BEDC.Derived.ValidatedNumericsUp.ValidatedNumericsPacket refinement composition containment`

Proof. Project the three accepted packets through Definition 357.1. The first refinement changes only the precision coordinate from ε_0 to ε_1 and keeps I, ω, ρ, λ, h, and c fixed as displayed rows. The second refinement changes only the precision coordinate from ε_1 to ε_2 and keeps the same six displayed rows fixed again.

By the classifier description in Definition 357.1, accepted packets are compared componentwise by the displayed rows and by replay of the finite containment routes. The two displayed comparisons therefore compose on the enclosure, modulus, observation window, readback token, transport row, and containment ledger. The transitivity part of Definition 3.8 supplies the corresponding hsame comparison for the transported rows, so V_2 still displays the same containment source window ρ, the same target enclosure I, and the same ledger c as V_0.

Apply Definition 357.5 to the second refinement V_1 to V_2. Since the enclosure row of V_1 is the same displayed row I inherited from V_0, every approximation read from the terminal refined request is contained in I.

It remains only to identify this containment as a read through the original ledger rather than through an unrelated copy. Apply Definition 357.9. The source approximation rows, the target interval row, the modulus row, the readback token, and the containment routes are transported by the same h and replay the same c at both refinement steps. Hence each terminal approximation read from V_2 lands in the original finite enclosure row I through the original Cont routes recorded in c.

Finally, Definition 357.3 keeps the readback token tied to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ window and finite modulus request, and Definition 357.7 excludes any extra analytic or host-numeric coordinate from the argument. □

Chapter 358

A Concrete Naming Certificate for $\mathsf{DyadicMesh}^{\uparrow}$

The $\mathsf{DyadicMesh}^{\uparrow}$ interface names finite dyadic grid packets used by real-completion and validated-numerics consumers. In the Moore–Kearfott–Cloud interval-analysis reading, a mesh is a finite grid of rational interval cells with explicit enclosure ledgers; in BEDC it is only a displayed BHist package over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicPrecision}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{ValidatedNumerics}^{\uparrow}$. The mesh cell row consumes the $\mathsf{RationalInterval}^{\uparrow}$ endpoint carrier of chapter 354 and exposes its validated enclosure ledger to the $\mathsf{ValidatedNumerics}^{\uparrow}$ consumer surface of chapter 357.

Before those interval and validated-numerics consumers read an enclosure, the finite mesh packet consumes the dyadic endpoint and radius rows licensed by chapter 291; $\mathsf{DyadicMesh}^{\uparrow}$ refines and packages that endpoint data into finite cell ledgers rather than treating dyadic rational endpoints as a merely adjacent surface.

Definition 358.1 (DyadicMesh BHist source)**.** *A* $\mathsf{DyadicMesh}^{\uparrow}$ *source is a finite* BHist *packet*

$$M = (\ell, J, I, E, H, C, P, N)$$

with the following visible rows:

1. *ℓ is a unary level row naming the displayed grid precision;*
2. *J is the finite dyadic cell-index row at level ℓ;*
3. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint row for the cell selected by J;*
4. *E is the containment ledger against* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording the dyadic lower endpoint, upper endpoint, midpoint, radius, and endpoint-order reads for the cell;*
5. *H stores componentwise* hsame *transport for the level, cell index, rational interval endpoints, dyadic endpoint reads, and containment ledger;*
6. *C stores the* Cont *routes by which refinement, endpoint readback, and downstream enclosure consumers inspect the same finite cell;*
7. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicPrecision}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{ValidatedNumerics}^{\uparrow}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$ *registration row.*

The classifier identifies accepted mesh packets by transporting them to a common displayed refinement level, comparing the refined cell-index rows there, and then comparing the rational interval endpoints by the inherited $\mathsf{RationalInterval}^{\uparrow}$ hsame *rows. The carrier has no coordinate for a completed real cell, infinite grid, host interval tree, or solver object.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.DyadicMeshUp.DyadicMeshUp`

Theorem 358.2 (DyadicMesh cell containment obligation)**.** *Every accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet* $M = (\ell, J, I, E, H, C, P, N)$ *exposes one finite dyadic cell. Its public containment surface is exactly the unary level row* ℓ*, the dyadic cell-index row* J*, the rational interval row* I*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius reads stored in* E*, the displayed transport rows* H*, the continuation routes* C*, the package provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshCarrier cell containment obligation`
Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket cell containment obligation`

Proof. Project the carrier of Definition 358.1. The selected cell is not reconstructed from an ambient grid object; it is the displayed pair of a level row and a finite cell-index row.

Read the rational interval coordinate I and the dyadic containment ledger E. These two rows provide the endpoint, midpoint, radius, and endpoint-order data visible to a consumer.

The remaining coordinates route and certify those reads. The transport row H, continuation row C, package row P, and naming row N exhaust the public containment surface, leaving no coordinate for a completed cell or external solver state. □

Theorem 358.3 (DyadicMesh refinement stability)**.** *Let* $M = (\ell, J, I, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet. If* M' *is obtained by appending displayed refinement evidence to* C *so that the level and cell-index rows refine to a finer dyadic row while the covered rational interval cell is transported by the same endpoint* hsame *rows, then* M' *is again an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet and covers the same real cell boundary visible to* $\mathsf{Real}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket refinement stability`

Proof. Start with the finite packet of Definition 358.1. Appending refinement evidence changes only the displayed continuation route by which the finer level and cell-index rows are read.

Transport the rational interval endpoint rows through H. Because the same endpoint hsame rows are preserved, the $\mathsf{RationalInterval}^{\uparrow}$ boundary seen after refinement is the transported form of I rather than a new completed real cell.

Repack the refined level, refined cell index, transported interval, dyadic containment ledger, updated continuation route, package provenance, and local name row. These are exactly the carrier coordinates, so the refined packet is accepted and the visible $\mathsf{Real}^{\uparrow}$ boundary is unchanged. □

Theorem 358.4 (DyadicMesh RationalInterval coverage)**.** *For every accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet* $M = (\ell, J, I, E, H, C, P, N)$*, the rational interval row* I *is the complete coverage object exported by the mesh cell. A* $\mathsf{RationalInterval}^{\uparrow}$ *consumer may read the endpoint order, containment ledger, and refinement routes through* I*,* E*,* H*, and* C*, but it cannot read an infinite mesh, a host subset of* $\mathsf{Real}^{\uparrow}$*, or a global real-completion proof from the mesh packet.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket rational interval coverage`

Proof. Project the interval coordinate I from the carrier. It is the only coverage row of the packet, and its endpoint and order data are interpreted by the existing $\mathsf{RationalInterval}^{\uparrow}$ surface.

Use the dyadic ledger E only to identify the dyadic endpoint and radius rows that support the interval read. The routes H and C transport and expose those finite rows to the consumer.

Since the carrier contains no additional coverage coordinate, the exported coverage is precisely I with its finite dyadic ledger. Infinite meshes, host subsets, and real-completion proofs are outside the packet surface. □

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket rationalinterval coverage`

Theorem 358.5 (DyadicMesh NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$ *obligation surface has five rows:*

1. *carrier: the finite* BHist *packet of Definition 358.1;*
2. *classifier: common-refinement comparison of level and cell-index rows followed by endpoint* hsame *comparison through* $\mathsf{RationalInterval}^{\uparrow}$*;*
3. *stability:* Cont *appends refinement evidence while preserving the transported rational interval cell;*
4. *exactness: every endpoint, midpoint, radius, order, and containment read is one of the dyadic ledger rows in E routed by C and transported by H;*
5. *ledger: the* Pkg *and local* NameCert *rows record exactly the dependency surface over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicPrecision}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{ValidatedNumerics}^{\uparrow}$.

These rows are the full seed naming surface for finite dyadic meshes.

Proof. The carrier row is the source packet itself. It lists the only visible coordinates from which a mesh cell can be read.

The classifier row first transports mesh packets to a common displayed refinement level. Once the cell-index rows are compared there, the rational interval endpoints are compared through the inherited endpoint hsame rows.

Stability is the refinement route of Definition 358.3. Exactness is the assertion that the dyadic endpoint, midpoint, radius, order, and containment reads are exactly the ledger entries in E routed by C and transported by H.

The ledger row is the package and local-name boundary. It records the named dependency certificates and therefore completes the seed NameCert surface without adding an analytic or solver coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket namecert obligation surface`

Theorem 358.6 (DyadicMesh ValidatedNumerics handoff)**.** *A* $\mathsf{ValidatedNumerics}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet may read exactly the finite enclosure package consisting of the level row, dyadic cell index,* $\mathsf{RationalInterval}^{\uparrow}$ *cell coverage row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *containment ledger, displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *refinement route,* $\mathsf{DyadicApproximation}^{\uparrow}$ *endpoint readback,* $\mathsf{Real}^{\uparrow}$ *boundary token,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The handoff supplies a finite enclosure mesh cell and does not assert solver correctness, floating-point hardware semantics, classical real completeness, or a global mesh refinement theorem.*

Proof. Read the mesh packet through Definition 358.2. The level, cell index, interval row, dyadic containment ledger, transport, continuation, provenance, and local name rows are the only public coordinates.

Apply Definition 358.4. The coverage object exported to the validated-numerics side is the finite $\mathsf{RationalInterval}^{\uparrow}$ row, supported by the dyadic endpoint and radius ledger.

Use Definition 358.3 for precision movement. Refinement is a displayed continuation route and therefore preserves the same covered cell boundary visible to the real-name consumer.

Thus the validated-numerics handoff is a finite enclosure mesh cell with named dependency rows. No row supplies solver correctness, hardware semantics, classical completeness, or a global refinement theorem. □

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket validatednumerics handoff`

Theorem 358.7 (DyadicMesh public export boundary)**.** *The public* $\mathsf{DyadicMesh}^{\uparrow}$ *export is exactly the finite packet rows of Definition 358.1 together with the containment obligation of Definition 358.2, refinement stability of Definition 358.3,* $\mathsf{RationalInterval}^{\uparrow}$ *coverage of Definition 358.4, the* $\mathsf{NameCert}$ *surface of Definition 358.5, and the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *handoff of Definition 358.6.*

This public boundary exports finite mesh-cell rows only. It does not export an infinite mesh, a host subset of $\mathsf{Real}^{\uparrow}$*, solver correctness, floating-point semantics, or classical real completeness.*

Proof. Project the source packet of Definition 358.1. Its visible rows are the level, finite cell index, rational interval cell, dyadic containment ledger, transport rows, continuation routes, package provenance, and local name row.

Apply Definition 358.2 and Definition 358.3. These identify the exported cell surface and show that displayed refinement changes only the finite reading route while preserving the transported cell boundary.

Next use Definition 358.4 and Definition 358.5. The coverage row is the finite $\mathsf{RationalInterval}^{\uparrow}$ cell, and the naming surface records the carrier, classifier, stability, exactness, and ledger rows without adding an analytic mesh object.

Finally apply Definition 358.6. The same finite rows are the only public enclosure rows available to downstream $\mathsf{ValidatedNumerics}^{\uparrow}$ consumers. Since every row is projected from the finite packet, infinite meshes, host subsets, solver correctness, floating-point semantics, and classical real completeness remain outside the export. □

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket public export boundary`

Theorem 358.8 (DyadicMesh standard finite mesh bridge boundary)**.** *For an accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packet* $M = (\ell, J, I, E, H, C, P, N)$ *satisfying Definition 358.1, the standard finite mesh reading available to downstream consumers is exactly the tuple consisting of the unary level row* ℓ*, the finite dyadic cell-index row* J*, the* $\mathsf{RationalInterval}^{\uparrow}$ *coverage row* I*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius-order ledger* E*, the displayed* $\mathsf{DyadicPrecision}^{\uparrow}$ *refinement route, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *endpoint readback, the* $\mathsf{Real}^{\uparrow}$ *boundary token, the componentwise* hsame *transports in* H*, the* Cont *routes in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

This bridge exports one finite enclosure mesh cell to $\mathsf{ValidatedNumerics}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers. It does not export host subsets, floating-point hardware semantics, solver correctness, completed-real compactness, or infinite mesh refinement.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket standard finite mesh bridge boundary`

Proof. Project the carrier rows from Definition 358.1. The packet supplies the unary level, finite cell index, rational interval row, dyadic containment ledger, transport rows, continuation routes, package provenance, and local name row as finite BHist data.

Use Definition 358.2 for the selected cell. The containment obligation identifies the public enclosure cell and rules out any ambient grid object or solver state.

Apply Definition 358.4 to read the interval coverage object. The exported coverage is the finite $\mathsf{RationalInterval}^{\uparrow}$ row supported by the dyadic endpoint, radius, and order ledger.

Next use Definition 358.3. Displayed refinement changes only the continuation route and preserves the transported cell boundary seen by the $\mathsf{Real}^{\uparrow}$ consumer.

Finally combine Definition 358.6 with Definition 358.7. Their common public surface is precisely the finite enclosure mesh cell named above, so host subsets, floating-point hardware semantics, solver correctness, completed-real compactness, and infinite refinement remain outside the bridge. □

Theorem 358.9 (DyadicMesh common refinement endpoint determinacy)**.** *Let $M = (\ell, J, I, E, H, C, P, N)$ and $M' = (\ell', J', I', E', H', C', P', N')$ be accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packets satisfying Definition 358.1. Suppose both packets are displayed at a common refinement level and, after transport to that level, agree on the refined finite cell-index row and on the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint* hsame *rows. Then every* $\mathsf{DyadicApproximation}^{\uparrow}$ *endpoint read and every* $\mathsf{Real}^{\uparrow}$ *boundary token exported through the two packets factors through the same finite* $\mathsf{RationalInterval}^{\uparrow}$ *row, endpoint-radius-order ledger,* Cont *route,* Pkg *provenance row, and local* NameCert *row.*

Equivalently, the consumer-visible rows for these three handoffs are the same finite BHist *rows routed through the common endpoint* hsame, Cont, Pkg, *and* NameCert *coordinates. The theorem does not assert a host subset of* $\mathsf{Real}^{\uparrow}$*, a completed real cell, solver state, floating-point semantics, or an infinite mesh.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket common refinement endpoint determinacy`

Proof. Start from Definition 358.1. Each accepted packet is a finite tuple of level, cell-index, rational-interval endpoint, dyadic ledger, transport, continuation, package, and name rows; no additional mesh object is available to downstream consumers.

Apply Definition 358.3 to both packets at the displayed common level. The hypothesis that the transported cell-index rows agree fixes the refined cell boundary through the shared hsame and Cont coordinates.

Next use Definition 358.4. Agreement of the transported endpoint hsame rows fixes the unique exported $\mathsf{RationalInterval}^{\uparrow}$ coverage row and the endpoint-radius-order ledger, and therefore fixes each endpoint read seen by the $\mathsf{DyadicApproximation}^{\uparrow}$ consumer.

Now apply Definition 358.6. The enclosure handoff reads exactly the finite cell, rational-interval endpoint, dyadic ledger, continuation, package, and local naming rows already fixed by the common refinement packet.

Finally use Definition 358.8. Its bridge surface names the $\mathsf{Real}^{\uparrow}$ boundary token and the endpoint readback as projections of the same finite BHist packet, so the two accepted packets give the same consumer-visible endpoint and boundary rows. The conclusion is only this finite-row determinacy claim and introduces no host subset, completed real cell, solver state, floating-point semantics, or infinite mesh. □

Theorem 358.10 (DyadicMesh validated terminal readback determinacy)**.** *Let M and M' be accepted* $\mathsf{DyadicMesh}^{\uparrow}$ *packets displayed at a common refinement as in Definition 358.9. Suppose a terminal accepted* $\mathsf{DyadicApproximation}^{\uparrow}$ *finite packet reads the same* $\mathsf{RationalInterval}^{\uparrow}$ *cell, endpoint-radius-order ledger,* RegSeqRat *observation window, and* $\mathsf{Real}^{\uparrow}$ *boundary token, and suppose a* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure consumes that same cell through the visible* hsame *transports,*

Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{DyadicMesh}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^\uparrow}$ *rows.*

Then every mesh-cell consumer, terminal dyadic readback, and validated enclosure read factors through one finite $\mathsf{Real}^\uparrow$ *boundary surface. The surface is the common refined cell with its rational interval endpoint, endpoint-radius-order ledger, terminal dyadic packet readback, validated enclosure row, observation window, transport rows, continuation routes, package rows, and local certificate rows. It does not assert completed real equality, a host subset of* $\mathsf{Real}^\uparrow$*, solver correctness, floating-point hardware semantics, or an infinite refinement object.*

Proof. First apply Definition 358.9. The common refinement fixes the finite cell row, rational interval endpoint, endpoint-radius-order ledger, $\mathsf{Real}^\uparrow$ boundary token, transport rows, continuation routes, package rows, and local mesh certificate row seen by the two displayed mesh packets.

Next read the terminal dyadic packet through Definition 335.38. The terminal packet contributes only its finite RegSeqRat observation window, rational interval cell, endpoint ledger, and boundary token, so it has the same finite target surface fixed by the mesh refinement.

Now use Definition 335.39 and Definition 335.41. These identify the terminal readback and the enclosure handoff as projections of the same finite rational-interval boundary, with the shared $\mathsf{Real}^\uparrow$ token and without a completed real equality claim.

Finally invoke the $\mathsf{ValidatedNumerics}^\uparrow$ bridge boundary used by the dyadic handoff. Its enclosure read consumes exactly the fixed cell, endpoint ledger, observation window, transport, continuation, provenance, and local certificate rows. Repacking those rows gives the single finite $\mathsf{Real}^\uparrow$ boundary surface claimed in the statement. □

Theorem 358.11 (DyadicMesh refinement enclosure export)**.** *Let* $M = (\ell, J, I, E, H, C, P, N)$ *and* $M' = (\ell', J', I', E', H', C', P', N')$ *be accepted* $\mathsf{DyadicMesh}^\uparrow$ *packets satisfying Definition 358.1. Suppose both are displayed at a common refinement level, agree there on the refined finite cell-index row and the transported* $\mathsf{RationalInterval}^\uparrow$ *endpoint rows, and are consumed by a* $\mathsf{DyadicApproximation}^\uparrow$ *endpoint read and a* $\mathsf{ValidatedNumerics}^\uparrow$ *enclosure over that same refined cell.*

Then the two mesh packets export exactly one finite enclosure surface to those consumers. The surface consists of the refined level row, refined cell-index row, $\mathsf{RationalInterval}^\uparrow$ *endpoint row, dyadic endpoint-radius-order ledger,* $\mathsf{DyadicApproximation}^\uparrow$ *endpoint readback,* $\mathsf{Real}^\uparrow$ *boundary token, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{DyadicMesh}^\uparrow}$, $\mathsf{NameCert}_{\mathsf{DyadicApproximation}^\uparrow}$, *and* $\mathsf{NameCert}_{\mathsf{ValidatedNumerics}^\uparrow}$ *rows.*

This export is a finite consumer boundary. It does not supply a host subset of $\mathsf{Real}^\uparrow$*, a completed real cell, solver correctness, floating-point semantics, or an infinite mesh.*

Lean4 ✓ `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket refinement enclosure export`

Proof. First apply Definition 358.9. The displayed common refinement fixes the refined level, refined cell-index row, transported rational interval endpoint, endpoint-radius-order ledger, endpoint readback, boundary token, transport rows, continuation rows, package rows, and local mesh certificate row seen by both packets.

Next expose the public mesh boundary. Use Definition 358.7 to restrict the exported rows to the finite packet surface of Definition 358.1; no ambient mesh or host real subset is introduced.

Now read the validated enclosure consumer. Apply Definition 358.6. Its handoff rows are the same finite cell, rational-interval endpoint, dyadic ledger, displayed refinement route, endpoint readback, real boundary token, transport, continuation, package, and local naming rows already fixed by the common refinement.

Finally apply Definition 335.37. It says that the dyadic endpoint read and validated enclosure read factor through the same finite mesh-enclosure packet. Repacking these fixed rows gives the single consumer surface listed in the statement, with no solver semantics, floating-point semantics, completed-real cell, host subset, or infinite mesh. □

Theorem 358.12 (DyadicMesh mature finite mesh treatment)**.** *The mature finite-mesh treatment for* $\mathsf{DyadicMesh}^{\uparrow}$ *consists exactly of the finite level row, finite cell-index row, rational-interval coverage row, dyadic endpoint, radius, and order ledger, displayed transport rows, continuation routes, package provenance, and local* NameCert *rows of the accepted packet.*

This treatment combines cell containment, refinement stability, $\mathsf{RationalInterval}^{\uparrow}$ *coverage, the* NameCert *surface, the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *handoff, the public export boundary, and the standard finite mesh bridge boundary. It exports no infinite mesh, host subset of* $\mathsf{Real}^{\uparrow}$*, floating-point hardware semantics, solver correctness, completed-real compactness, or global refinement theorem.*

Proof. Project the finite packet of Definition 358.1. Its rows are the level, cell index, rational interval cell, dyadic containment ledger, transport data, continuation routes, provenance, and local name.

Use Definition 358.2, Definition 358.3, and Definition 358.4 to identify the finite mesh cell, its displayed refinement route, and its rational-interval coverage object.

Next apply Definition 358.5, Definition 358.6, and Definition 358.7. These rows repack the same finite cell for naming and downstream validated-numerics consumers.

Finally use Definition 358.8. It is the standard finite-mesh boundary for the packet, and none of the projected rows contains an infinite mesh, a host subset, solver semantics, hardware semantics, completed-real compactness, or a global refinement theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicMesh}^{\uparrow}$ packet is generated by a unary level row, a finite dyadic cell-index row, a $\mathsf{RationalInterval}^{\uparrow}$ cell coverage row, a $\mathsf{DyadicRatCore}^{\uparrow}$ containment ledger, componentwise hsame transports, displayed Cont refinement routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 358.1, Definition 358.2, Definition 358.3, Definition 358.4, Definition 358.5, Definition 358.6, Definition 358.7, Definition 358.8, Definition 358.9, Definition 358.10, Definition 358.11, and Definition 358.12 specify the public finite mesh carrier, containment row, refinement route, rational-interval coverage row, NameCert obligation surface, $\mathsf{ValidatedNumerics}^{\uparrow}$ handoff rows, standard finite mesh bridge boundary, common-refinement endpoint determinacy, validated terminal readback determinacy, refinement enclosure export, and mature finite-mesh treatment.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicMesh}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicMeshUp.DyadicMeshPacket\_standard\_finite\_mesh\_bridge\_boundary`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close solver correctness, floating-point hardware semantics, classical real completeness, infinite meshes, or a global refinement theorem.

Upgrade path: No further paper-side theory grade remains; formal verification remains on the separate formal axis.

Chapter 359

A Concrete Naming Certificate for $\mathsf{RealDiagonalCompletion}^{\uparrow}$

The $\mathsf{RealDiagonalCompletion}^{\uparrow}$ interface names the concrete diagonal completion packet that sits between the regular Cauchy family used in the real-completeness capstone and the finite $\mathsf{Real}^{\uparrow}$ seal surface. Following the Bishop–Bridges constructive-analysis treatment of Cauchy completion, the packet keeps the regular rational input, its explicit modulus, the chosen diagonal windows, and the real-seal ledger as visible certificate data. It does not form a quotient of regular streams, select a limit by choice, or import host real completeness.

The diagonal completion packet consumes the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular rational sequence carrier of chapter 301, and the real seal boundary of chapter 38. This chapter is the finite ledger that routes those sibling surfaces through the displayed diagonal selector without quotient streams, host limits, or choice-selected representatives.

The packet consumes the diagonal-window source of chapter 372 and instantiates the finite diagonal route from chapter 9238 before exposing the $\mathsf{Real}^{\uparrow}$ seal. Thus its carrier records the substantive path from regular Cauchy diagonal data to completion-facing seal data rather than standing as a kernel-only packet.

Definition 359.1 (RealDiagonalCompletion source packet)**.** *A* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *source packet is a finite* BHist *row*

$$D = (X, M, \Delta, S, R, E, P, N)$$

with the following visible coordinates:

1. *X is a* $\mathsf{RegSeqRat}^{\uparrow}$*-indexed regular Cauchy input family, with every inspected member read as a regular rational sequence;*
2. *M is an explicit unary modulus row selecting a late enough member of X for each requested precision;*
3. *Δ is the diagonal selector row that sends a precision request to the selected member of X and to the regular-sequence window read inside that member;*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of all windows requested by the diagonal selector;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger for the selected diagonal windows;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *classifier and seal ledger attached to the diagonal representative;*

7. *P is the* Pkg *provenance naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* NameCert *source certificates;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *provenance row.*

The classifier compares two accepted source packets only by the displayed modulus rows, selected diagonal windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal ledgers, transporting each component through the visible* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionTasteGate single carrier alignment`

Theorem 359.2 (RealDiagonalCompletion carrier habitation)**.** *Every regular Cauchy input family over* $\mathsf{RegSeqRat}^{\uparrow}$ *with an explicit unary modulus row and displayed finite readbacks determines an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *carrier by placing the input family, modulus, diagonal selector,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger,* $\mathsf{Real}^{\uparrow}$ *seal ledger, package provenance, and local* NameCert *row into the source packet of Definition 359.1.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket carrier habitation`

Proof. Start with the finite rows supplied by the regular Cauchy input. The family row gives the $\mathsf{RegSeqRat}^{\uparrow}$-indexed source, and the modulus row gives a displayed unary bound for each requested precision.

Use the diagonal selector to choose the visible late member and the visible regular-sequence window inside that member. The selected windows form the finite $\mathsf{StreamName}^{\uparrow}$ schedule, and the corresponding rational observations form the $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger.

Attach the $\mathsf{Real}^{\uparrow}$ seal ledger to the same selected rows. The provenance row names only the displayed source certificates, and the local NameCert row records the packet assembly, so the resulting carrier contains no quotient, selected host limit, or hidden completion datum. □

Theorem 359.3 (RealDiagonalCompletion classifier stability)**.** *Let D and D' be accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packets. If their modulus rows, diagonal selector rows, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal ledgers,* Pkg *provenance, and local* NameCert *rows are transported componentwise through displayed* hsame *comparisons and* Cont *routes, then the transported packet is accepted and the local classifier relates it to the original packet.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket classifier stability`

Proof. Project the packet coordinates of Definition 359.1. Each coordinate is finite certificate data, so a displayed hsame comparison transports the row without changing the carrier kind.

The continuation routes identify which finite windows are read after transport. The diagonal selector is therefore transported together with the $\mathsf{StreamName}^{\uparrow}$ schedule, and the selected readbacks remain the same $\mathsf{RegSeqRat}^{\uparrow}$ observations seen through the transported routes.

Finally repack the $\mathsf{Real}^{\uparrow}$ seal ledger and the package provenance. Since the classifier compares exactly these rows, the transported packet is accepted and classified with the original one. □

Theorem 359.4 (RealDiagonalCompletion ledger exactness)**.** *For every accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet, each* $\mathsf{Real}^{\uparrow}$ *consumer read of the diagonal representative factors through one displayed diagonal selector row, one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the associated* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational observation ledger, and the visible* Pkg *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket ledger exactness`

Proof. Read the accepted packet through its source definition. The $\mathsf{Real}^{\uparrow}$ seal coordinate is not independent data; it is attached to the selected diagonal windows and their regular rational readbacks.

A consumer precision request first meets the unary modulus row, then the diagonal selector row, and then the scheduled finite stream window. The readback ledger records the $\mathsf{RegSeqRat}^{\uparrow}$ observation and its $\mathsf{DyadicRatCore}^{\uparrow}$ rational evidence at that same window.

Thus the visible ledger exhausts the read. There is no remaining coordinate from which a quotient representative, host real endpoint, hidden tail choice, or ambient completeness proof could be exported. □

Theorem 359.5 (RealDiagonalCompletion diagonal window extraction)**.** *For every accepted* RealDiagonalCompleti *packet D and every unary precision request n, the consumer read of the diagonal representative factors through a displayed modulus row $M(n)$, a displayed diagonal selector row $\Delta(n)$, one scheduled* $\mathsf{StreamName}^{\uparrow}$ *window $S(n)$, and one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row $R(n)$. These rows are transported only by the visible* hsame *comparisons and* Cont *routes carried by the packet provenance.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket window extraction`

Proof. Project D by Definition 359.1. The modulus coordinate supplies the finite member of the regular Cauchy input family selected for the unary request n.

Use the diagonal selector coordinate to identify the displayed window inside that member. The scheduled $\mathsf{StreamName}^{\uparrow}$ row is exactly the finite window named by this selector, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row is the corresponding regular rational observation.

Repack these rows with the visible Pkg and NameCert provenance. Since the packet classifier transports exactly these coordinates through hsame and Cont, the extracted window is a consumer-facing factor of the diagonal read rather than an ambient limit selection. □

Theorem 359.6 (RealDiagonalCompletion real seal factorization)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal ledger E of an accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet is determined by the family of extracted diagonal windows of Definition 359.5 together with the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger. No host limit, quotient representative, or ambient completeness proof is exported by this factorization.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket seal factorization`

Proof. Combine Definition 359.4 with the window extraction surface of Definition 359.5. Every consumer read of the diagonal representative has already passed through the modulus row, selector row, scheduled stream window, and regular rational readback.

The source packet attaches the $\mathsf{Real}^{\uparrow}$ seal coordinate to exactly those finite readbacks. Therefore two accepted packets with the same extracted windows and the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger expose the same $\mathsf{Real}^{\uparrow}$ seal ledger to consumers.

The conclusion is only a finite ledger factorization. It does not form a quotient of regular streams, choose a host real limit, or appeal to an ambient completeness theorem. □

Theorem 359.7 (RealDiagonalCompletion common-refinement seal invariance)**.** *Let D_0 and D_1 be accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packets. Suppose the diagonal selectors of D_i read selected* $\mathsf{StreamName}^{\uparrow}$ *windows $S_i(n)$ with displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support, and suppose that for the inspected boundary requests these windows both transport to one common-refinement* $\mathsf{StreamName}^{\uparrow}$ *packet $S_*(n)$. Suppose further that the transported* $\mathsf{RegSeqRat}^{\uparrow}$ *readback*

rows and the transported $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledgers of* D_0 *and* D_1 *agree on* $S_*(n)$ *for each such boundary request.*

Then the attached $\mathsf{Real}^{\uparrow}$ *seal ledgers of* D_0 *and* D_1 *are classifier-equal at the shared boundary. The equality is only the finite seal-ledger equality carried by the displayed* hsame *and* Cont *transports through the two* Pkg *and* NameCert *rows. It does not assert quotient stream equality, host real equality, a selected host limit, countable choice, ambient completeness, or a host-level equality of completed real numbers.*

Proof. Apply Definition 359.5 to both accepted packets. For each inspected boundary request this exposes the modulus row, diagonal selector, selected $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ readback used by the $\mathsf{Real}^{\uparrow}$-facing consumer read.

Next apply Definition 359.6 to both packets. The attached $\mathsf{Real}^{\uparrow}$ seal ledger is read only through those extracted windows and the displayed regular-rational readbacks, so the comparison of seal ledgers reduces to the comparison of the transported finite-window data.

Use Definition 38.661 on the shared common-refinement window $S_*(n)$. The hypotheses give agreement of the transported $\mathsf{RegSeqRat}^{\uparrow}$ readbacks and $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledgers on that window, so the terminal $\mathsf{Real}^{\uparrow}$ seal route is unique at the shared boundary.

Finally transport this terminal route back through the displayed Pkg and NameCert rows of D_0 and D_1. Componentwise hsame supplies the classifier equality of the seal ledgers, while the Cont routes record that the consumer reads pass through the same selected boundary data. None of the cited rows contains quotient stream equality, host real equality, selected limits, countable choice, or ambient completeness, so the conclusion is exactly the finite classifier equality stated above. □

Theorem 359.8 (RealDiagonalCompletion constant rational handoff)**.** *If the input regular Cauchy family is the stationary family generated by a* $\mathsf{Rat}^{\uparrow}$ *row* q*, then the accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet classifies its diagonal representative with the constant* $\mathsf{Real}^{\uparrow}$ *embedding of* q*. The diagonal selector always reads the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member, and every selected* $\mathsf{StreamName}^{\uparrow}$ *window returns the same rational row through the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket constant rational handoff`

Proof. The stationary family has a constant regular rational source row. Its modulus can be the displayed unary zero-threshold row, because every selected member of the family reads the same rational value.

The diagonal selector therefore chooses a window of the same stationary $\mathsf{RegSeqRat}^{\uparrow}$ member at every requested precision. The $\mathsf{StreamName}^{\uparrow}$ schedule may vary the requested window, but the readback ledger always returns the row q with its $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ evidence.

The $\mathsf{Real}^{\uparrow}$ seal ledger is the constant rational handoff for those same rows. Componentwise hsame transport and the displayed Cont routes preserve the stationary readback, so the classifier identifies the diagonal representative with the constant embedding of q. □

Theorem 359.9 (RealDiagonalCompletion NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier habitation: Definition 359.2 assembles the finite source packet from the regular Cauchy input, unary modulus, diagonal selector, finite stream schedule, readback ledger, real seal, provenance, and local ledger;*
2. *classifier stability: Definition 359.3 transports exactly the modulus, selector, selected windows, readbacks, and seal ledgers through* hsame *and* Cont*;*

3. *diagonal-window exactness: Definition 359.4 forces each consumer read through a displayed selector window and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
4. *constant rational handoff: Definition 359.8 identifies stationary rational inputs with their constant* $\mathsf{Real}^{\uparrow}$ *embeddings;*
5. *ledger non-escape: the packet exposes only* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, Pkg, *and* NameCert *rows, and contains no quotient stream, host real equality, choice-selected limit, or ambient completion object.*

Lean4 ✓ `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket namecert obligation surface`

Proof. The first four rows are exactly the preceding carrier, stability, exactness, and stationary-handoff theorems. They list the finite source coordinates and describe how a consumer reaches every displayed read.

For the ledger boundary, inspect the source packet. Its coordinates name only the dependency certificates and the local provenance rows, so a consumer cannot obtain quotient, choice, or host-completion data from the carrier.

These five rows are precisely the naming-certificate obligations for the diagonal completion object. They turn the diagonal representative into a concrete BHist packet surface while this chapter leaves machine-checked real completeness unclaimed. □

Theorem 359.10 (RealDiagonalCompletion scoped consumer seal package)**.** *Every accepted* RealDiagonalCompleti *packet satisfying the source row of Definition 359.1, the carrier construction, classifier stability, ledger exactness, diagonal window extraction, seal factorization, constant rational handoff, and* NameCert *obligation surface determines a scoped dependency package over* BHist, hsame, Cont, Pkg, NameCert, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Rat}^{\uparrow}$.

Every $\mathsf{Real}^{\uparrow}$*-facing consumer read of that package factors through the displayed diagonal selector, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal ledger, and provenance rows. No quotient stream, choice-selected limit, host real equality, or ambient completeness object is part of the scoped package.*

Proof. Project the source packet of Definition 359.1. Its visible coordinates are the finite BHist carrier row, the displayed transport routes through hsame and Cont, the package and local NameCert provenance, and the source certificates for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$.

The carrier row is supplied by Definition 359.2. Classifier transport is supplied by Definition 359.3, and each consumer read is forced through the displayed diagonal-window ledger by Definition 359.4 and Definition 359.5.

The seal route is the factorization of Definition 359.6. For stationary rational inputs, Definition 359.8 keeps the handoff inside the same $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The complete local obligation surface is the five-row NameCert packet of Definition 359.9. These rows exhaust the available fields of the accepted packet.

Since none of the cited rows contains a quotient stream, choice-selected limit, host real equality, or ambient completeness object, the scoped consumer seal package has exactly the dependency surface stated above. □

Theorem 359.11 (RealDiagonalCompletion public RealUp handoff)**.** *Every accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet is repacked as a consumer-facing* $\mathsf{Real}^{\uparrow}$ *seal read through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$,

StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows only. The repacking is exactly the composition of diagonal-window extraction, real-seal factorization, constant rational handoff, and scoped primitive dependency over the accepted packet surface.*

The handoff exports no Real$^{\uparrow}$ *completeness theorem, quotient equality of regular streams, selected limit, uniqueness of limits, or host real construction.*

Proof. Begin with the extracted consumer window of Definition 359.5. It routes each requested read through the displayed modulus, diagonal selector, finite StreamName$^{\uparrow}$ window, and RegSeqRat$^{\uparrow}$ readback row.

Next apply the seal factorization of Definition 359.6. The Real$^{\uparrow}$ seal is read from those extracted windows and their displayed rational observations, with hsame and Cont accounting for the permitted transports.

For stationary rational inputs, Definition 359.8 identifies the same surface with the constant Rat$^{\uparrow}$ handoff inside the RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ dependency rows.

Finally use the scoped dependency package of Definition 359.10. That package exhausts the accepted packet fields by BHist carrier data, package provenance, local NameCert rows, and the named dependency certificates. Since none of these rows contains completeness, quotient equality, a selected limit, uniqueness of limits, or a host real carrier, the public read has exactly the stated boundary. □

Theorem 359.12 (RealDiagonalCompletion public finite completeness interface)**.** *Every accepted* RealDiagonalCompletion$^{\uparrow}$ *packet satisfying the source packet, carrier construction, classifier stability, ledger exactness, diagonal-window extraction, seal factorization, constant rational handoff,* NameCert *obligation surface, and scoped consumer seal package exports a public* Real$^{\uparrow}$*-facing finite completeness interface.*

For every requested consumer read, the interface factors the read through an explicit unary modulus, the displayed diagonal selector, a finite StreamName$^{\uparrow}$ *window, a* RegSeqRat$^{\uparrow}$ *readback, a* DyadicRatCore$^{\uparrow}$ *observation ledger, a* Real$^{\uparrow}$ *seal row,* Pkg *provenance, and a local* NameCert *row. It exports no quotient stream, selected host limit, uniqueness theorem, or ambient real completeness theorem.*

Proof. Start from the source packet of Definition 359.1. The carrier theorem Definition 359.2 supplies the finite rows, and Definition 359.3 transports those rows only along the displayed hsame and Cont routes.

The read itself is the diagonal-window readback of Definition 359.4 and Definition 359.5. These two results exhibit the unary modulus, diagonal selector, finite stream window, regular-sequence readback, and dyadic observation ledger used by the consumer.

The Real$^{\uparrow}$ output row is the seal factorization of Definition 359.6. For constant rational inputs, Definition 359.8 keeps the read inside the same Rat$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ dependency rows.

Finally, Definition 359.9, Definition 359.10, and Definition 359.11 repack these finite rows into the public consumer interface. Since the cited packet rows contain no quotient stream, selected host limit, uniqueness theorem, or ambient real-completeness object, the public interface has exactly the stated boundary. □

Theorem 359.13 (RealDiagonalCompletion standard finite bridge)**.** *Any accepted* RealDiagonalCompletion$^{\uparrow}$ *packet satisfying the source packet of Definition 359.1 and the public finite completeness interface of Definition 359.12 repacks as the standard BEDC bridge surface over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Every bridge read factors through the displayed modulus, diagonal selector, finite stream window, regular-sequence readback, real seal, and provenance rows. The bridge exports no quotient stream, host real equality, selected limit, uniqueness theorem, or ambient completeness object.

Proof. The carrier habitation theorem Definition 359.2 supplies the accepted finite packet, and Definition 359.4 identifies the exact ledger through which a consumer read must pass. Together these fix the bridge carrier as a BHist surface rather than an external completion object.

For each requested read, use Definition 359.5 to expose the modulus, selector, finite $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ readback. Then apply Definition 359.6 to route the same read into the displayed $\mathsf{Real}^{\uparrow}$ seal, with the constant rational case constrained by Definition 359.8.

The scoped consumer seal package of Definition 359.10 records the permitted hsame, Cont, Pkg, and NameCert routes. The public $\mathsf{Real}^{\uparrow}$ handoff of Definition 359.11 and the finite completeness interface of Definition 359.12 then repack those routes as the public bridge surface.

It remains to check that the bridge has no extra coordinate. The source packet lists exactly X, M, Δ, S, R, E, P, and N; the preceding theorems account for each of these rows. Coordinate exhaustion therefore rules out any quotient stream, host real equality, selected limit, uniqueness theorem, or ambient completeness object. □

Theorem 359.14 (RealDiagonalCompletion bridge readback roundtrip)**.** *Reading the standard finite packet of Definition 359.13 back into BEDC recovers the same accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *rows: the unary modulus row, diagonal selector row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, package provenance, and local* NameCert *row.*

The readback is conservative over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *It exports no quotient stream, selected host limit, uniqueness theorem, host real equality, or ambient real completeness.*

Proof. Begin with the bridge packet of Definition 359.13. Its coordinates are the displayed packet rows of Definition 359.1; there is no additional host coordinate to interpret.

Apply Definition 359.5 to read back the modulus, diagonal selector, finite $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ row. The same projection also recovers the dyadic observation ledger because Definition 359.4 identifies the accepted read ledger with those displayed coordinates.

Next use Definition 359.6 and Definition 359.11. These repack the recovered window and ledger as the same $\mathsf{Real}^{\uparrow}$ seal row, with all transports accounted for by the displayed hsame and Cont routes.

Finally apply Definition 359.12. The bridge readback forgets only the wrapper naming the standard finite packet and keeps the same Pkg provenance and local NameCert row. Projection followed by repacking therefore returns the accepted BEDC packet with no quotient stream, selected host limit, uniqueness theorem, host real equality, or ambient completeness coordinate. □

Theorem 359.15 (RealDiagonalCompletion mature consumer package)**.** *The carrier, classifier stability, ledger exactness, diagonal-window extraction, seal factorization, public finite completeness interface, standard finite bridge, and bridge readback for accepted* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packets assemble a mature consumer package for downstream* $\mathsf{Real}^{\uparrow}$ *completion consumers.*

Every read in the package is routed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The package exports no quotient stream, host limit, uniqueness theorem, or ambient completeness object.*

Proof. Start with the accepted source packet of Definition 359.1. The carrier theorem Definition 359.2 supplies the finite packet rows, while Definition 359.3 records that consumer transport uses only displayed hsame and Cont routes.

Next recover the read coordinates. The ledger exactness theorem Definition 359.4 fixes the observable row, and Definition 359.5 exposes the modulus, diagonal selector, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger used by each read.

The $\mathsf{Real}^{\uparrow}$ output is then the seal of Definition 359.6. The constant rational boundary of Definition 359.8 keeps the $\mathsf{Rat}^{\uparrow}$ case inside the same visible rational, regular-sequence, dyadic, and real rows.

Finally apply the public package results. The NameCert obligation surface of Definition 359.9, the scoped consumer seal package of Definition 359.10, the public finite completeness interface of Definition 359.12, the standard finite bridge of Definition 359.13, and the bridge readback of Definition 359.14 together account for every exported coordinate. Since those cited rows contain only the named BEDC dependency surfaces, the mature consumer package has exactly the stated boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDiagonalCompletion}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$-indexed regular Cauchy input, an explicit unary modulus, a diagonal selector, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal ledger, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 359.10, Definition 359.12, Definition 359.13, Definition 359.14, and Definition 359.15 bind the finite diagonal completion carrier, classifier transport, diagonal-window readback exactness, constant rational handoff, public $\mathsf{Real}^{\uparrow}$ finite completeness interface, standard finite bridge, BEDC readback surface, and mature consumer package over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDiagonalCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealDiagonalCompletionUp.RealDiagonalCompletionSourcePacket\_ledger\_exactness`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of regular streams, uniqueness of limits, host real construction, or machine bridge checking.

Upgrade path: No further paper-side theory-grade target is open; verification-axis audit cleanliness and machine bridge checking remain separate obligations.

The exact-boundary consumer of this completion surface is chapter 1120, which reads the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as a finite boundary packet.

Chapter 360

A Concrete Naming Certificate for $\mathsf{RealCauchyCompletion}^{\uparrow}$

The $\mathsf{RealCauchyCompletion}^{\uparrow}$ interface names the Bishop–Bridges constructive Cauchy completion surface for real names. It sits above $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the diagonal packet $\mathsf{RealDiagonalCompletion}^{\uparrow}$, and it exports only the $\mathsf{Real}^{\uparrow}$ seal rows that can be read through those certificates. The chapter records a concrete completion object for the real-and-completion phase without forming quotient classes of regular streams, selecting limits by choice, or appealing to host real equality.

The regular-family input side is consumed through the already licensed $\mathsf{RegularCauchyName}^{\uparrow}$ boundary of chapter 371. That boundary packages the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal route that $\mathsf{RealCauchyCompletion}^{\uparrow}$ uses as its visible regular-family input surface.

Definition 360.1 (RealCauchyCompletion carrier)**.** *A* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (X, \mu, D, E, P, N)$$

with the following visible coordinates:

1. *X is a* $\mathsf{RegSeqRat}^{\uparrow}$*-indexed regular Cauchy family, with every inspected member presented through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational observations;*
2. *μ is the displayed unary modulus row that assigns a late family member to each requested precision;*
3. *D is the associated* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet, including its diagonal selector, scheduled windows, and regular-sequence readback ledger;*
4. *E is the consumer-facing* $\mathsf{Real}^{\uparrow}$ *seal row attached to the diagonal representative;*
5. *P is the* Pkg *provenance row naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source certificates;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *row.*

The local classifier compares two carriers only by displayed hsame *transport of the modulus rows, diagonal packets, readback ledgers,* $\mathsf{Real}^{\uparrow}$ *seal rows, package provenance, and local certificate rows.*

Lean4 (definition) `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier`

Theorem 360.2 (RealCauchyCompletion NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *surface consists of five obligations: carrier habitation by the packet of Definition 360.1; classifier stability under displayed* hsame *transport and* Cont *routes; diagonal handoff through* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*; non-escape of quotient, choice, host real equality, and ambient completeness data; and the standard Bishop reading as a constructive Cauchy completion over visible regular rational names.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier namecert obligation surface`

Proof. Carrier habitation is the assembly of the displayed family, modulus, diagonal packet, real seal, package provenance, and local certificate row. The classifier is componentwise because each coordinate is a visible finite row, and the only permitted transports are the stored hsame comparisons followed by the recorded Cont routes.

The diagonal handoff is delegated to $\mathsf{RealDiagonalCompletion}^{\uparrow}$: every consumer read of the candidate limit first passes through the modulus, diagonal selector, scheduled $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger. The carrier therefore exposes only the rows named by $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, Pkg, and NameCert.

The Bishop reading is exactly this finite witness surface. It supplies a constructive regular-Cauchy completion interface while withholding quotient stream equality, choice-selected limits, host real equality, and any ambient completeness theorem not carried by the named certificate rows. □

Theorem 360.3 (RealCauchyCompletion diagonal handoff)**.** *For every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier* C *and every unary precision request* n*, the* $\mathsf{Real}^{\uparrow}$ *consumer read of the completed name factors through the modulus row* $\mu(n)$*, the associated* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet* D*, one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier diagonal handoff`

Proof. Project the carrier by Definition 360.1. The modulus coordinate selects the late member of the regular Cauchy family that is relevant for the request n.

Use the diagonal packet coordinate D to read the chosen member through its diagonal selector. The selected finite stream window and its regular-sequence readback are exactly the rows exposed by the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ ledger.

Attach the $\mathsf{Real}^{\uparrow}$ seal row E to that readback and retain the package and local certificate provenance. Since the consumer reaches E only along this displayed route, the handoff factors through the stated finite rows. □

Theorem 360.4 (RealCauchyCompletion non-escape boundary)**.** *An accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier exports no quotient of regular streams, no host real carrier, no host real equality, no choice-selected limit, and no uniqueness principle beyond the displayed* $\mathsf{Real}^{\uparrow}$ *classifier rows. Every exported read is mediated by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier non escape boundary`

Proof. Inspect the carrier coordinates. Each field is either a finite history row, a displayed transport route, a package provenance row, a local certificate row, or one of the named dependency certificates.

The diagonal coordinate already records its own non-escape boundary: it exposes selected windows and readbacks, not quotient representatives or host limits. Passing through $\mathsf{Real}^{\uparrow}$ adds only the seal row and its classifier-visible comparisons.

Thus a downstream consumer can cite precisely the listed rows. There is no coordinate from which quotient stream equality, host real equality, a selected limit, or an ambient completeness object can be recovered. □

Theorem 360.5 (RealCauchyCompletion standard Bishop reading)**.** *Read in the Bishop–Bridges constructive-analysis sense,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *is the completion interface for regular Cauchy families over rational names: the input is a* $\mathsf{RegSeqRat}^{\uparrow}$*-indexed regular Cauchy family with an explicit modulus, and the output is the* $\mathsf{Real}^{\uparrow}$ *name obtained by diagonal extraction and finite ledger transport through* $\mathsf{RealDiagonalCompletion}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier standard bishop reading`

Proof. The input side is fully displayed: the regular Cauchy family is a $\mathsf{RegSeqRat}^{\uparrow}$ row family, and the modulus is a unary row stored in the carrier. No convergence fact is inferred from an ambient metric space or from an external completeness principle.

The output side is the diagonal representative already carried by $\mathsf{RealDiagonalCompletion}^{\uparrow}$. Its windows are finite $\mathsf{StreamName}^{\uparrow}$ reads, its rational observations are $\mathsf{DyadicRatCore}^{\uparrow}$ rows, and its real-facing surface is the $\mathsf{Real}^{\uparrow}$ seal.

This is the Bishop-style reading of completion as an explicit transformation from regular Cauchy data to a real name with a visible modulus and ledger. The certificate records the construction interface and its boundary, leaving machine checking of the full real-completion theorem to the corresponding Lean namespace. □

Theorem 360.6 (RealCauchyCompletion carrier transport stability)**.** *Let* $C = (X, \mu, D, E, P, N)$ *be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier. Any displayed* hsame *transport that preserves the* X*,* μ*,* D*,* E*,* P*, and* N *coordinates through the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* Pkg*,* Cont*, and* NameCert *rows preserves both the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier predicate and the local classifier. In particular, transported packets are compared only by the same component rows that classified the original packet.*

Proof. Project the carrier by Definition 360.1. The regular family coordinate X and the modulus coordinate μ are transported through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ rows, so the input side remains a visible regular Cauchy family with the same precision schedule.

The diagonal coordinate D is transported through $\mathsf{RealDiagonalCompletion}^{\uparrow}$, and the seal coordinate E is transported through $\mathsf{Real}^{\uparrow}$. The recorded Cont routes carry the finite readback ledgers along those transports, so the consumer-facing route continues to expose the same scheduled stream windows and rational observations.

Finally, P and N transport through the package and local certificate rows. Since the classifier of Definition 360.1 compares exactly these displayed coordinates, componentwise transport preserves accepted carriers and preserves classifier equivalence. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier transport stability`

Theorem 360.7 (RealCauchyCompletion modulus diagonal exactness)**.** *For an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier* $C = (X, \mu, D, E, P, N)$ *and a unary precision request* n*, the row* $\mu(n)$ *determines the displayed diagonal packet used for that request together with its finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows. The route contains no quotient of regular streams, no host real equality, and no choice-selected limit.*

Proof. The modulus row is a coordinate of the carrier, not an external completeness principle. Evaluating it at n selects the late regular-sequence member whose readback must be inspected for that precision.

The diagonal coordinate D then gives the corresponding displayed packet: a scheduled stream window, the regular-sequence ledger for that window, and the dyadic rational observation rows used by the readback. These rows are finite history data already named by the carrier provenance.

Because the selected route is exhausted by μ, D, and the finite readback ledgers, there is no remaining operation that could form quotient stream equality, appeal to host real equality, or select a limit by choice. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier modulus diagonal exactness`

Theorem 360.8 (RealCauchyCompletion real seal readback scope)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier is consumed only through the diagonal readback route named by* μ *and* D*. Every consumer read of the seal is mediated by the displayed finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *packet rows.*

Proof. The seal coordinate E is attached to the diagonal representative carried by D. A consumer therefore reaches E only after the modulus row has selected the relevant regular-sequence member and the diagonal ledger has exposed the finite window and rational observation rows.

The Cont routes record how those rows are read without leaving the displayed packet. The package provenance P and local certificate row N bind the dependency certificates that make the route public to downstream consumers.

Thus the exported seal has a restricted readback scope: it is a real-facing boundary for the named diagonal packet, not an independent host real, quotient class, or ambient completion object. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier real seal readback scope`

Theorem 360.9 (RealCauchyCompletion stream-regseq-tail seal factorization)**.** *Let* $C = (X, \mu, D, E, P, N)$ *be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-and-completion consumer read of the exported seal factors through a finite* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the diagonal packet* D*, and then the* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

The exported route is exactly the displayed completion chain carried by C*. It exports no quotient of regular streams, no host real equality, no selected limit, and no ambient completeness principle.*

Proof. Project the carrier by Definition 360.1. The coordinates X and μ supply the visible regular-family input and the finite precision request that selects the relevant stream window.

Apply the diagonal handoff of Definition 360.3. The selected request reaches the diagonal packet D only through the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger already recorded in the carrier route.

Finally use Definition 360.4 and Definition 360.8. These rows identify E as the only exported $\mathsf{Real}^{\uparrow}$ seal reached by the route, while excluding quotient stream equality, host real equality, choice of a selected limit, and ambient completeness from the consumer surface. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier stream regseq tail seal factorization`

Theorem 360.10 (RealCauchyCompletion tail selector fusion)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose completion read is served by a* $\mathsf{RegularCauchyName}^{\uparrow}$ *source and by a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-budget meet over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window. Every* $\mathsf{Real}^{\uparrow}$ *seal read of C factors through one finite route consisting of the* $\mathsf{StreamName}^{\uparrow}$ *tail-compatible window request, the* $\mathsf{RegularCauchyName}^{\uparrow}$ *dyadic-radius ledger, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-budget meet, the* $\mathsf{Real}^{\uparrow}$ *name-seal source factorization, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

The fused route is exactly the displayed finite consumer packet. It has no quotient stream equality, no host real equality, no selected limit, no metric-completeness datum, and no ambient completeness coordinate.

Proof. Start with Definition 360.9. It places every completion-facing seal read on the route through the selected finite stream window, regular-sequence readback row, dyadic observation ledger, diagonal packet, and exported $\mathsf{Real}^{\uparrow}$ seal.

Next use the StreamName tail-compatible request and coverage rows, Definition 126.542 and Definition 126.543. They identify the finite window as the certified observation index, tail-selector route, finite-window budget surface, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and shared support rows.

Attach the CauchyLimitSeal component by Definition 386.135 and Definition 386.137. The selector row, observation-budget row, dyadic ledger, regular handoff, real seal, transports, continuations, provenance, and local name are therefore read inside the same tail-budget meet.

Finally apply Definition 38.686. Repacking the shared StreamName, RegularCauchyName, CauchyLimitSeal, and Real name-seal rows gives one finite consumer route over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, with the listed quotient, host-equality, selected-limit, metric-completeness, and ambient-completeness coordinates absent. □

Theorem 360.11 (RealCauchyCompletion RegularCauchyName terminal handoff)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose regular-family input is read through an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *source packet $G = (S, Q, \rho, L, E_G, P_G, N_G)$. Every completion-facing* $\mathsf{Real}^{\uparrow}$ *seal read of C factors through the RegularCauchyName finite window ledger L, the dyadic-radius row ρ, the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet D, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal row E.*

The handoff exports no quotient stream, no selected host limit, no host real equality, no ambient completeness row, and no metric-completeness row. It is a finite certificate route through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data of the two accepted packets.*

Proof. Start from the accepted $\mathsf{RegularCauchyName}^{\uparrow}$ packet G. By Definition 371.9, every precision read of G is exhausted by the displayed finite route through the stream schedule, dyadic observation row, dyadic-radius row, finite ledger, seal row, package provenance, and local certificate.

Use Definition 371.10 to identify the real-facing endpoint of that route. The $\mathsf{Real}^{\uparrow}$ seal exposed by G is reached only after the same dyadic-radius row, $\mathsf{StreamName}^{\uparrow}$ handoff, and $\mathsf{RegSeqRat}^{\uparrow}$ readback have been displayed as certificate rows.

Now pass to the accepted completion carrier C. By Definition 360.9, the completion-facing seal read factors through the selected finite stream window, the regular-sequence readback row, the dyadic observation ledger, the diagonal packet D, and then the exported $\mathsf{Real}^{\uparrow}$ seal row E.

Combining the two routes gives the terminal handoff from the RegularCauchyName finite ledger into the RealDiagonalCompletion packet and then into the RealCauchyCompletion seal row. Finally, Definition 360.4 excludes quotient streams, selected host limits, host real equality, ambient completeness, and metric-completeness data from the exported surface. □

Theorem 360.12 (RealCauchyCompletion ProbeBundle window coverage)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier, and let Q be a finite* ProbeBundle *of unary precision requests. For every request n bundled in Q, the consumer read factors through the displayed row $\mu(n)$, the associated* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet, one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger.*

Moreover, the whole bundled family repacks as a single finite consumer surface over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The assembled surface carries the local* SemanticNameCert *whose carrier is the bundled row satisfying* hsame *with the surface,* UnaryHistory*, and* $\mathsf{Pkg}^{\mathrm{sig}}$*, whose pattern forgets only the* $\mathsf{Pkg}^{\mathrm{sig}}$ *row, whose ledger records* $\mathsf{Pkg}^{\mathrm{sig}}$ *together with* hsame*, and whose classifier is* hsame*. This surface contains no quotient stream, no selected host limit, no host real equality, and no ambient completeness object.*

Proof. Project the carrier by Definition 360.1. For each bundled precision request n, apply Definition 360.3 to obtain the route through $\mu(n)$, the diagonal packet D, the finite stream window, the regular-sequence readback row, and the dyadic observation ledger.

The selected row is the same row identified by Definition 360.7. Thus the bundled requests do not create separate hidden completion data; they reuse the displayed diagonal rows attached to the unary modulus.

Finally, apply Definition 360.8 to the family of reads. Since Q is finite, the pointwise routes assemble into one finite BHist consumer surface with the recorded hsame transports, Cont routes, Pkg provenance, and local NameCert row, and the same non-escape boundary is preserved for the bundle.

The checked Lean target supplies exactly this certificate row: its source predicate requires hsame with the bundle surface, UnaryHistory, and $\mathsf{Pkg}^{\mathrm{sig}}$; its pattern drops only $\mathsf{Pkg}^{\mathrm{sig}}$; its ledger keeps $\mathsf{Pkg}^{\mathrm{sig}}$ and hsame; and its classifier is the same hsame relation. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier probe bundle window coverage`

Theorem 360.13 (RealCauchyCompletion selector-refusal boundary)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier, and let Q be a finite* ProbeBundle *of precision requests. The bundled read determined by Q may use only the displayed modulus rows, the diagonal packet, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, the* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data carried by C.*

The same bundled read cannot be reinterpreted as a selected host limit, a quotient stream equality, an ambient completeness object, or a host real equality. It remains a finite certificate-row read of the completion interface.

Proof. Apply Definition 360.12 to the finite request bundle Q. This routes every bundled precision request through the displayed modulus row, the finite stream window, the regular-sequence readback row, the dyadic observation ledger, and the assembled BHist consumer surface.

The selected diagonal row is the one fixed by Definition 360.7. Hence the bundle reuses the carrier's visible diagonal packet and does not acquire any separate completion coordinate beyond the rows already named by C.

For the consumer endpoint, use Definition 360.8. The only real-facing object reached by the bundle is the $\mathsf{Real}^{\uparrow}$ seal along the displayed Cont, Pkg, and NameCert route.

Finally apply Definition 360.4. It excludes selected host limits, quotient stream equality, host real equality, and ambient completeness from the exported surface, so the finite bundle remains a certificate-row read rather than a limit selector. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionUp StdBridge`

Theorem 360.14 (RealCauchyCompletion finite bundle induction)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier, and let Q be a finite* ProbeBundle *of unary precision requests. The completion-facing bundled read determined by Q is generated by induction on the constructors of Q.*

For the empty bundle, the generated surface is the empty finite consumer surface over the displayed Pkg *row P and* NameCert *row N. For an extension of a bundle by one unary precision request n, the generated surface appends the route through $\mu(n)$, the selected* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet D, one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then fuses that route with the induction surface through the same* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Consequently every finite bundled completion read of C is a finite certificate-row route. It exports no quotient stream equality, no selected host limit, no host real equality, no countable-choice coordinate, no metric-completeness datum, and no ambient completeness object.

Proof. Induct on the finite ProbeBundle constructors. In the empty case there is no unary precision request to route, so the consumer surface is exactly the empty finite surface over the carrier's displayed Pkg and NameCert rows.

For the extension step, suppose the induction surface has already been generated for the previous bundle. Apply Definition 360.12 to the added unary request n inside the extended bundle. This supplies the route through $\mu(n)$, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the dyadic observation ledger, and the assembled consumer surface.

Use Definition 360.7 to identify the selected row with the carrier's displayed diagonal packet D. The new one-request route therefore has the same BHist, hsame, Cont, Pkg, and NameCert interface as the induction surface, so the two finite surfaces fuse without adding any host completion coordinate.

Finally apply Definition 360.13 and Definition 360.4. These keep quotient stream equality, selected host limits, host real equality, countable choice, metric-completeness data, and ambient completeness outside the generated finite route. □

Theorem 360.15 (RealCauchyCompletion obligation closure package)**.** *The carrier of Definition 360.1, the* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *obligation surface, the diagonal handoff, the non-escape boundary, and the Bishop reading jointly supply the full paper-side obligation surface for* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

The package preserves the exclusions on quotient streams, host real equality, selected limits, and ambient completeness: every exported completion read is mediated by the displayed regular family row, unary modulus row, diagonal packet, finite stream window, regular-rational readback, dyadic observation ledger, $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. The carrier row is Definition 360.1. It lists the regular Cauchy family, modulus, diagonal packet, real seal, package provenance, and local certificate as finite packet coordinates.

The obligation surface is Definition 360.2. It binds carrier habitation, classifier stability, diagonal handoff, non-escape, and Bishop-style reading as the five rows consumed by the local certificate.

The diagonal and real-facing readback rows are Definition 360.3, Definition 360.7, and Definition 360.8, with finite bundle coverage supplied by Definition 360.12. They force every consumer

read through the modulus-selected diagonal packet, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic observation ledger, and $\mathsf{Real}^{\uparrow}$ seal.

The boundary and interpretation rows are Definition 360.4 and Definition 360.5. They record that the packet is a constructive regular-Cauchy completion interface while withholding quotient stream equality, host real equality, choice-selected limits, and ambient completeness.

Assembling these five displayed rows gives the stated paper-side obligation closure package. No extra coordinate is introduced, so the exclusions are preserved componentwise. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier obligation closure package`

Theorem 360.16 (RealCauchyCompletion regular-family source adequacy)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier. The visible regular-family source surface is adequate for the exported* $\mathsf{Real}^{\uparrow}$ *completion read precisely when every unary precision request factors through the same displayed rows already named by the carrier: the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row X, the modulus row μ, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet D, a finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the local* $\mathsf{Real}^{\uparrow}$ *seal row E, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Equivalently, the regular-family input route and the public $\mathsf{Real}^{\uparrow}$ *seal readback determine each other over exactly those rows. Neither direction includes a quotient stream, selected host limit, host real equality, ambient completeness, arithmetic package, order package, or metric-completeness row.*

Proof. For the forward direction, project C through Definition 360.1. The projection exposes X, μ, D, E, P, and N as the carrier coordinates available to a completion-facing consumer.

Apply Definition 360.3, Definition 360.7, and Definition 360.8. These rows force each unary precision request through the modulus-selected diagonal packet, one finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the local $\mathsf{Real}^{\uparrow}$ seal row. For a finite request bundle, Definition 360.12 repacks the same pointwise route as one BHist consumer surface with the displayed hsame, Cont, Pkg, and NameCert rows.

For the reverse direction, start from a public completion read whose every unary request factors through precisely the listed rows. Repack the family row, modulus, diagonal packet, finite windows, dyadic ledger, seal row, provenance, and local naming row as the carrier coordinates of Definition 360.1.

Use Definition 360.2 to discharge the local carrier, classifier-stability, handoff, non-escape, and Bishop-reading obligations, then use Definition 360.15 to bind those obligations as the public $\mathsf{RealCauchyCompletion}^{\uparrow}$ package. The non-escape boundary of Definition 360.4 and the Bishop reading of Definition 360.5 exclude quotient streams, selected host limits, host real equality, ambient completeness, and the unrelated arithmetic, order, and metric-completeness rows from both readings. □

Theorem 360.17 (RealCauchyCompletion regular-name route agreement)**.** *Let $C = (X, \mu, D, E, P, N)$ be an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier. The* $\mathsf{RegularCauchyName}^{\uparrow}$ *regular-family input route and the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *finite completion-transformer route determine the same public* $\mathsf{Real}^{\uparrow}$ *seal surface exactly when both routes factor through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows attached to C.*

The agreement is a finite seal-surface agreement, not an equality of quotient streams, a selected limit, a host real equality, a trichotomy result, or an ambient completeness principle.

Proof. First read the regular-family side through Definition 360.16. It forces every unary precision request through the regular family row, modulus row, diagonal packet, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the local $\mathsf{Real}^{\uparrow}$ seal row.

Next read the public seal side through Definition 360.18. This repacks the same rows as a public $\mathsf{Real}^{\uparrow}$-facing consumer surface over BHist, componentwise hsame, displayed Cont, Pkg provenance, and the local NameCert row.

Finally compare with the completion-transformer route of chapter 378. The unit boundary, diagonal-bind stability, schedule composition boundary, and public real-seal factorization there put every $\mathsf{CauchyCompletionMonad}^{\uparrow}$ read through the same finite window, dyadic-observation, regular-readback, diagonal, seal, transport, route, provenance, and naming rows. Since both routes reach the public seal only through this displayed surface, they agree exactly on that surface and on no host-level completion object. □

Theorem 360.18 (RealCauchyCompletion public real-name export)**.** *The carrier of Definition 360.1, the diagonal handoff, the non-escape boundary, carrier transport stability, modulus-diagonal exactness, real-seal readback scope,* ProbeBundle *window coverage, and obligation-closure package assemble into a public* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer export over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The exported surface is the $\mathsf{Real}^{\uparrow}$*-facing name route through the displayed regular family row, unary modulus row,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, and the local certificate. It is not a quotient equality theorem, a host completeness theorem, or a proof of host equality of real numbers.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier public real name export`

Proof. Begin with the finite carrier of Definition 360.1. It supplies the regular family row, modulus, diagonal packet, real seal, package provenance, and local naming row that form the consumer-visible packet.

The diagonal route is supplied by Definition 360.3 and sharpened by Definition 360.7. These results force each precision read through the displayed modulus row, the associated diagonal packet, one finite stream window, one regular rational readback, and the dyadic observation ledger.

Transport and finite-window reuse are supplied by Definition 360.6 and Definition 360.12. They show that the same rows assemble over BHist, hsame, Cont, Pkg, and NameCert for finite consumer bundles.

Finally apply Definition 360.8, Definition 360.4, and Definition 360.15. These identify the public object as a $\mathsf{Real}^{\uparrow}$-facing name surface and preserve the exclusions on quotient stream equality, selected host limits, host real equality, and ambient completeness. □

Theorem 360.19 (RealCauchyCompletion regular-family input interface)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier binds its* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *input rows to the completion carrier of Definition 360.1. The binding sends the regular family row X and unary modulus row μ to the diagonal packet D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* Pkg *provenance, and local* NameCert *row, all inside the displayed* BHist*,* hsame*, and* Cont *surface.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier regular family input interface`

Proof. Start with the completion carrier. Its coordinates explicitly contain the regular-family row, the unary modulus, the diagonal packet, the real seal, package provenance, and local naming certificate.

Apply the diagonal handoff of Definition 360.3. For each precision request, the modulus selects the family row that is read through one finite stream window, one regular-rational readback, and the dyadic observation ledger.

The exact selected row is the one identified by Definition 360.7. The $\mathsf{Real}^{\uparrow}$ seal then remains in the scope described by Definition 360.8, so the input rows are bound to the completion carrier without creating quotient stream equality, host real equality, or a selected host limit. □

Theorem 360.20 (RealCauchyCompletion downstream consumer handoff)**.** *For any accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier and any finite consumer request bundle, the downstream handoff is the route through the unary modulus, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *diagonal packet, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal,* Pkg*, and* NameCert*. The route is stable under the displayed* hsame *and* Cont *rows and exports no data outside that surface.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier downstream consumer handoff`

Proof. Apply Definition 360.12 to the finite bundle. Each request is reduced to the modulus-selected diagonal packet, its finite stream window, its regular-rational readback, and the dyadic observation rows.

Use Definition 360.6 to transport the same rows through the displayed hsame comparisons and Cont routes. Package provenance and the local naming row stay attached to the consumer-facing surface throughout the transport.

Finally, Definition 360.8 identifies the exported endpoint as the $\mathsf{Real}^{\uparrow}$ seal reached through that route. The non-escape boundary of Definition 360.4 excludes quotient streams, selected host limits, host real equality, and ambient completeness from the downstream handoff. □

Theorem 360.21 (RealCauchyCompletion CauchyLimitSeal route)**.** *An accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier and an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet determine the same public finite seal route exactly when both reads factor through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. This is the sibling seal route recorded by chapter 386; it is not a quotient-stream equality, a selected host-limit principle, or an ambient completeness theorem.*

Proof. Project the $\mathsf{RealCauchyCompletion}^{\uparrow}$ carrier by Definition 360.1. Its visible coordinates supply the regular-family row, unary modulus, diagonal packet, real seal, package provenance, and local certificate row.

Apply the diagonal handoff of Definition 360.3 and the public real-name export of Definition 360.18. These rows force every completion read through the displayed finite stream window, regular-rational readback, dyadic ledger, diagonal packet, real seal, BHist surface, componentwise hsame transport, displayed Cont route, Pkg provenance, and local NameCert row.

Compare this route with the $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer handoff of chapter 386. The sibling packet reaches its public seal through the same displayed windows, readbacks, dyadic observations, diagonal completion packet, seal row, transports, route records, provenance, and naming rows.

Since both public reads reach the seal only along this common finite surface, the two routes agree exactly on that surface. The non-escape boundary from Definition 360.4 keeps quotient streams, selected host limits, host real equality, and ambient completeness outside the conclusion. □

Theorem 360.22 (RealCauchyCompletion finite request fusion)**.** *Let C be an accepted* RealCauchyCompletion$^{\uparrow}$ *carrier, and let Q and Q' be finite* ProbeBundle *request surfaces for C. If the unary precision rows selected by Q and Q' are* hsame*-compatible, then their downstream handoff routes fuse into one finite* BHist *consumer packet over the same unary modulus,* RealDiagonalCompletion$^{\uparrow}$ *diagonal packet,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *observations,* Real$^{\uparrow}$ *seal,* Pkg *provenance, and* NameCert *row. The fused packet exports no quotient stream equality, selected host limit, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier finite request fusion`

Proof. Apply the ProbeBundle window coverage theorem separately to Q and Q'. Each finite request surface is reduced to modulus-selected diagonal reads, scheduled stream windows, regular-rational readbacks, and dyadic observation rows from the accepted carrier.

Use the assumed hsame-compatibility of the unary precision rows to identify the modulus channel shared by the two surfaces. Modulus-diagonal exactness then places both request families on the same displayed diagonal packet and its associated finite windows.

Next apply carrier transport stability and the downstream consumer handoff. The two finite handoff packets are combined by finite packet union, while the stored hsame and Cont rows transport the readbacks, dyadic observations, real seal, provenance, and local naming row along the same carrier surface.

Finally inherit the non-escape boundary componentwise from the two handoffs. Since neither input packet contains a quotient stream, selected host limit, host real equality, or ambient completeness field, the fused finite consumer packet cannot export any such row. □

Theorem 360.23 (RealCauchyCompletion scoped consumer route)**.** *The regular-family input interface of Definition 360.19, the downstream handoff of Definition 360.20, and the obligation package of Definition 360.15 assemble the public consumer theorem for* RealCauchyCompletion$^{\uparrow}$*. Its scope is exactly the completion-facing route through* RegularCauchyFamily$^{\uparrow}$*, modulus selection, diagonal packet, finite windows, regular-rational readback, dyadic ledger,* Real$^{\uparrow}$ *seal,* ProbeBundle *coverage,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier scoped consumer route`

Proof. The input side is supplied by Definition 360.19. It binds the RegularCauchyFamily$^{\uparrow}$ rows to the carrier coordinates and to the modulus-selected diagonal readback route.

The consumer side is supplied by Definition 360.20. It packages finite request bundles through the same diagonal packet, stream windows, regular-rational readbacks, dyadic ledger, real seal, package provenance, and local certificate rows.

The package side is Definition 360.15. Together with carrier stability, modulus-diagonal exactness, real-seal readback scope, and the non-escape boundary, it gives a scoped public theorem about the concrete BEDC completion interface and nothing stronger than that interface. □

Theorem 360.24 (RealCauchyCompletion standard bridge boundary)**.** *The standard BEDC completion-interface reading of any accepted* RealCauchyCompletion$^{\uparrow}$ *carrier factors exactly through the displayed* RegularCauchyFamily$^{\uparrow}$ *row, the unary modulus row, the* RealDiagonalCompletion$^{\uparrow}$ *packet, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *observations, the* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. This boundary exports no quotient stream, host equality of real numbers, selected limit, or ambient completeness theorem.*

Proof. Use Definition 360.19 to place the accepted carrier on the regular-family row, unary modulus row, diagonal packet, finite stream windows, regular-rational readbacks, dyadic observation ledger, real seal, package provenance, and local naming row.

The downstream side is then Definition 360.20. It shows that every finite consumer request reads the same visible route through BHist, hsame, Cont, Pkg, and NameCert.

Finally combine Definition 360.8 with Definition 360.4. These rows identify the exported endpoint as the $\mathsf{Real}^{\uparrow}$ seal and exclude quotient streams, host real equality, selected limits, and ambient completeness from the standard completion-interface reading. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier standard bridge boundary`

Theorem 360.25 (RealCauchyCompletion bridge handoff)**.** *The public real-name export of Definition 360.18, the regular-family input interface of Definition 360.19, the downstream consumer handoff of Definition 360.20, and the standard bridge boundary of Definition 360.24 jointly supply the paper bridge needed for the public-to-bridged* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *closure step.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier bridge handoff`

Proof. The public export supplies the $\mathsf{Real}^{\uparrow}$-facing name surface. The input interface pins that surface to the regular-family row, unary modulus, diagonal packet, stream windows, regular-rational readbacks, dyadic observations, package provenance, and local certificate.

The downstream handoff shows that all finite consumer bundles pass through the same rows, with hsame transport and Cont routes preserving the visible package. The standard boundary then records that this is exactly the standard BEDC completion-interface reading.

Composing these four displayed rows gives a paper bridge from the public certificate to the standard completion schema. The conclusion remains paper-side only: it does not assert machine bridge checking, quotient equality, host equality, selected limits, or ambient completeness. □

Theorem 360.26 (Real Cauchy completion mature limit-seal consumption)**.** *The regular-family row of Definition 360.19, the diagonal packet of Definition 360.3, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal of Definition 360.8,* Pkg *provenance, and the* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *row jointly supply the finite consumer boundary used by downstream completion examples. The boundary is a limit-seal consumption surface and does not claim quotient equality of regular streams, host equality of real numbers, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier mature limit seal consumption`

Proof. Start with an accepted carrier. The regular-family interface fixes the source rows and unary modulus, while the diagonal handoff identifies the scheduled packet through which each precision request is read.

The finite windows and dyadic ledger then give the concrete observation route. Each request exposes one $\mathsf{StreamName}^{\uparrow}$ window, its $\mathsf{DyadicRatCore}^{\uparrow}$ observations, and the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ readback before any real-facing row is consumed.

The real-seal readback scope places the $\mathsf{Real}^{\uparrow}$ seal after that route. Package provenance and the local NameCert row attach the dependency certificates that downstream completion examples are allowed to cite.

Thus the mature consumer boundary is finite and seal-facing: it packages the regular-family row, diagonal packet, windows, dyadic ledger, regular readback, seal, provenance, and naming row.

Since no additional coordinate is introduced, quotient equality, host real equality, selected limits, and ambient completeness remain outside the claim. □

Theorem 360.27 (RealCauchyCompletion family tail synchronization)**.** *Let C and C' be accepted* RealCauchyCompletion$^{\uparrow}$ *carriers whose regular-family rows are classified through the* RegSeqRat$^{\uparrow}$ *common-refinement classifier, and let the requested precisions be carried by one finite* ProbeBundle*. Then the two* Real$^{\uparrow}$ *seal reads synchronize through a single displayed tail window before either seal is read.*

The window consists of the shared unary modulus bound selected from the two carriers, one StreamName$^{\uparrow}$ *window for the finite request bundle, one* RegSeqRat$^{\uparrow}$ *readback row supplied by the common refinement, and one* DyadicRatCore$^{\uparrow}$ *observation ledger. The synchronized route stays inside the rows of Definition 360.1,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it exports no quotient equality of regular streams, host equality of real numbers, selected limit, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Proof. Project the two accepted carriers through Definition 360.1. This exposes the two regular-family rows, their unary modulus rows, diagonal packets, finite stream-window ledgers, regular-rational readbacks, dyadic observation ledgers, real-seal rows, package provenance, and local naming rows.

Use the regular-family input interface of Definition 360.19 and the downstream consumer handoff of Definition 360.20. These two rows identify the only route by which a finite precision request may travel from the regular-family input to a seal-facing consumer read.

The RegSeqRat$^{\uparrow}$ common-refinement classifier and the StreamName$^{\uparrow}$ pointwise handoff supply the shared finite window. The unary modulus bound is read once for the bundled requests, and the same window feeds both regular-rational readback rows.

Finally read the corresponding DyadicRatCore$^{\uparrow}$ observation ledger before either Real$^{\uparrow}$ seal is consumed. The real-seal readback scope and non-escape boundary then show that both seal reads are synchronized by that displayed finite tail route, with no exported quotient stream, selected host limit, host real equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier family tail synchronization`

Theorem 360.28 (RealCauchyCompletion budgeted common refinement)**.** *Let C and C' be accepted* RealCauchyCompletion$^{\uparrow}$ *carriers whose regular-family rows classify through one* RegSeqRat$^{\uparrow}$ *common-refinement row, and let B be an accepted* RealObservationBudget$^{\uparrow}$ *packet carrying the same finite* ProbeBundle *request surface. Then the synchronized* Real$^{\uparrow}$ *seal route factors through one budgeted common-refinement packet.*

This packet consists exactly of the RealObservationBudget$^{\uparrow}$ *precision rows, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *readbacks and displayed radii,* RegSeqRat$^{\uparrow}$ *handoff rows,* Real$^{\uparrow}$ *seal boundary, componentwise* hsame *transports, displayed* Cont *route replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient equality of regular streams, host real equality, selected limit, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Apply Definition 360.27 to the two accepted carriers and the shared finite request surface. This gives the common tail window through which both seal reads pass before either Real$^{\uparrow}$ seal is consumed.

Now read the accepted RealObservationBudget$^{\uparrow}$ packet. Window coverage in Definition 611.3 identifies the same finite StreamName$^{\uparrow}$ windows, while Definition 611.4 places every RegSeqRat$^{\uparrow}$ handoff after the precision row, windows, dyadic readbacks, and displayed radii.

The real seal boundary of Definition 611.5 puts the seal rows after those budget coordinates, and Definition 611.6 excludes quotient equality, host real equality, selected limits, ambient completeness, and hidden unbounded schedules from the packet.

Finally use Definition 360.20 to replay the synchronized route for downstream consumers, and use Definition 360.4 to keep the resulting route inside the displayed BHist, hsame, Cont, Pkg, and NameCert surface. Thus the completion synchronization and the observation budget are the same finite common-refinement packet at the consumer boundary. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier budgeted common refinement`

Theorem 360.29 (RealCauchyCompletion null-tail budget correspondence)**.** *Let C and C' be accepted* RealCauchyCompletion$^{\uparrow}$ *carriers whose regular-family rows are classified through one* RegSeqRat$^{\uparrow}$ *common-refinement row, and let one finite* ProbeBundle *carry the requested precision observations. On the displayed BEDC rows, the synchronized* Real$^{\uparrow}$ *seal route of Definition 360.27 is equivalent to giving one* NullSequence$^{\uparrow}$ *zero-tail packet whose tolerance-window package is carried by one* DyadicTailBound$^{\uparrow}$ *finite budget ledger.*

The correspondence lives only over BHist, hsame, Cont, Pkg, NameCert, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, NullSequence$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, *and* Real$^{\uparrow}$. *It exports no quotient streams, selected limits, host real equality, trichotomy, or ambient completeness.*

Proof. Begin with the synchronized tail route. By Definition 360.27, both accepted completion carriers read the same finite tail window before either Real$^{\uparrow}$ seal is consumed; this window is built from the common RegSeqRat$^{\uparrow}$ refinement, the finite StreamName$^{\uparrow}$ request window, and the DyadicRatCore$^{\uparrow}$ observation ledger.

Turn that shared window into zero-tail evidence by applying Definition 380.10. The common-window requirement is supplied by Definition 380.13, so the two carrier reads are compared by one NullSequence$^{\uparrow}$ packet rather than by a host equality of limits.

Now account for the precision budget. Apply Definition 379.16 to package the tolerance windows into one finite DyadicTailBound$^{\uparrow}$ ledger, and apply Definition 379.8 to route the Real$^{\uparrow}$ seal through that ledger and the regular-rational readback.

Conversely, a NullSequence$^{\uparrow}$ zero-tail packet equipped with the stated DyadicTailBound$^{\uparrow}$ budget ledger determines the same displayed tail window for the finite request bundle. Re-reading it through the common RegSeqRat$^{\uparrow}$ refinement gives exactly the synchronized seal route, and all named transports remain inside the listed BEDC rows. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier null tail budget correspondence`

Theorem 360.30 (RealCauchyCompletion synchronized null-tail consumer)**.** *Let C and C' be accepted* RealCauchyCompletion$^{\uparrow}$ *carriers whose regular-family rows classify through one* RegSeqRat$^{\uparrow}$ *common-refinement row, and let one finite* ProbeBundle *carry the precision requests. Then the synchronized seal route and the null-tail dyadic-budget packet assemble into one consumer-visible* Real$^{\uparrow}$ *equality surface.*

This surface is exactly the finite BEDC route over BHist, hsame, Cont, Pkg, NameCert, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, NullSequence$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, *and* Real$^{\uparrow}$*: two accepted carrier reads are equal for the consumer only by passing through the common tail window, the shared zero-tail certificate, the finite dyadic budget ledger, and the resulting* Real$^{\uparrow}$ *seal. It exports no quotient stream equality, selected limit, host real equality, trichotomy, or ambient completeness theorem.*

Proof. First build the common consumer route. Apply Definition 360.27 to the two accepted carriers and the shared ProbeBundle. This gives one finite tail window before either Real$^{\uparrow}$ seal is

consumed, with the window read through the common $\mathsf{RegSeqRat}^{\uparrow}$ refinement, the $\mathsf{StreamName}^{\uparrow}$ request window, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

Next convert the synchronized window into the null-tail budget packet. Apply Definition 360.29. Its forward direction packages the same synchronized route as one $\mathsf{NullSequence}^{\uparrow}$ zero-tail packet equipped with one $\mathsf{DyadicTailBound}^{\uparrow}$ finite budget ledger; its reverse direction rereads such a packet as the same synchronized seal route.

The certificate ingredients are the displayed null-sequence and dyadic-tail rows. Use Definition 380.10 and Definition 380.13 for the shared zero-tail window, then use Definition 379.16 and Definition 379.8 for the finite tolerance ledger and its $\mathsf{Real}^{\uparrow}$ seal factorization.

Combining these rows gives a reusable consumer equality surface rather than a stronger equality principle. Every transport still runs through BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so quotient streams, selected limits, host real equality, trichotomy, and ambient completeness remain outside the claim. □

Lean4 ✓ `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionCarrier synchronized null tail consumer`

Theorem 360.31 (RealCauchyCompletion tail-budget coherence consumer)**.** *Every accepted* RealCauchyCompletio *carrier $C = (X, \mu, D, E, P, N)$ whose diagonal packet consumes a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row reads its completion-facing* $\mathsf{Real}^{\uparrow}$ *seal only after the regular-family row, unary modulus row, diagonal packet, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the tail-budget row of chapter 906 have been displayed. The consumer route excludes quotient stream equality, selected limits, host real equality, and ambient completeness.*

Proof. Project the accepted carrier through Definition 360.1. This exposes the regular-family row X, unary modulus μ, diagonal packet D, completion-facing seal E, package provenance P, and local naming row N.

Apply the diagonal handoff of Definition 360.3. The precision request must pass through μ, the diagonal packet D, one finite $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger before the seal row can be read.

Now read the consumed $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ row from chapter 906. Its observation gate supplies the tail-threshold, finite observation service, window service, dyadic ledger, regular readback, and seal handoff that the diagonal packet is using as its budget-row source.

Finally use the real-seal readback scope of Definition 360.8 and the downstream consumer handoff of Definition 360.20. They keep the completion-facing seal inside the displayed regular-family, modulus, diagonal, finite-window, readback, dyadic-ledger, tail-budget, hsame, Cont, Pkg, and NameCert route, so no quotient stream equality, selected limit, host real equality, or ambient completeness is exported. □

Theorem 360.32 (RealCauchyCompletion tail-budget seal route)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose diagonal packet consumes a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row and whose completion-facing seal is compared through a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window reads its* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route consisting of that* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the tail-budget threshold, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-budget meet of chapter 386, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

The route exports no quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness principle.

Proof. Project the accepted carrier through Definition 360.1. This displays the completion carrier as its regular-family row, unary modulus, diagonal packet, completion-facing seal, package provenance, and local naming row.

Apply the diagonal handoff of Definition 360.3. The same finite $\mathsf{StreamName}^{\uparrow}$ window must carry the precision request through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the completion-facing seal can be consumed.

Now apply Definition 360.31. It forces the diagonal packet to expose the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ threshold and its finite observation service inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

Attach the sibling seal packet from chapter 386. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ route compares the completion-facing seal over the same finite window and meets the tail-budget row before the $\mathsf{Real}^{\uparrow}$ seal is read.

Finally use Definition 360.8. Since both the completion carrier and the seal packet have already been routed through the finite window, readback row, dyadic ledger, tail-budget threshold, $\mathsf{CauchyLimitSeal}^{\uparrow}$ meet, structural transports, package provenance, and local naming row, the exported surface contains no quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness. □

Theorem 360.33 (RealCauchyCompletion tail-fusion budget consumer)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose diagonal packet consumes both a* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *row and a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row reaches its* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route consisting of the regular-family row, unary modulus row, diagonal packet, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *shared-threshold row of chapter 658, the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget row of chapter 906, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The resulting consumer route exports no quotient stream equality, selected limit, host real equality, trichotomy, or ambient completeness principle.

Proof. Start with the displayed completion carrier. The family-tail synchronization row identifies the shared finite tail window through which the regular-family input, unary modulus, diagonal packet, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger must pass before any $\mathsf{Real}^{\uparrow}$ seal is read.

Next add the budgeted common-refinement row. It places the same synchronized window under the finite budget surface used by the consumer request bundle, so the precision rows, displayed radii, regular readback, dyadic observations, hsame transports, Cont routes, Pkg provenance, and local naming rows are read as one packet.

Now read the synchronized null-tail consumer row together with the tail-budget coherence consumer row. The null-tail row supplies the consumer-visible equality surface over the shared tail window, while the tail-budget row forces the diagonal packet to expose the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ budget ledger before the completion-facing seal is consumed.

Finally insert the $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ shared-threshold row into that route. The tail-fusion packet contributes only the finite cofinal threshold and seal-facing handoff already named by its public consumer surface; it does not introduce quotient stream equality, a selected host limit, host real equality, trichotomy, or ambient completeness. Hence the $\mathsf{RealCauchyCompletion}^{\uparrow}$ seal is reached exactly through the finite fusion-and-budget route stated above. □

Theorem 360.34 (RealCauchyCompletion tail-cofinality schedule factorization)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose diagonal packet consumes a* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *row, a*

NullSequence$^{\uparrow}$ *zero-tail packet, and the tail-budget and tail-fusion rows of Definition 360.31 and Definition 360.33 reaches its* Real$^{\uparrow}$ *seal only through the displayed finite schedule factorization.*

That factorization consists exactly of the regular-family row, unary modulus, diagonal packet, the TailCofinalitySchedule$^{\uparrow}$ *finite-window route, the* NullSequence$^{\uparrow}$ *zero-tail package,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It exports no quotient streams, selected limits, host real equality, trichotomy, or ambient completeness.*

Proof. Start with the accepted completion carrier and apply Definition 360.33. This forces every seal-facing consumer to expose the regular-family row, unary modulus, diagonal packet, finite StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ ledger, RegSeqRat$^{\uparrow}$ readback, fusion row, budget row, structural transports, provenance, and local naming row before the Real$^{\uparrow}$ seal is read.

Next read Definition 360.30. It identifies the same finite tail window with the NullSequence$^{\uparrow}$ zero-tail package and the dyadic budget ledger, so the consumer-visible equality surface is still a displayed BEDC packet rather than a quotient stream or selected host limit.

Now insert the TailCofinalitySchedule$^{\uparrow}$ packet. The schedule real-seal readback theorem Definition 681.6 orders the schedule rows as the unary request, finite window, dyadic ledger, regular-rational readback, and real seal. The scoped-binding theorem Definition 681.8 keeps that route inside BHist, hsame, Cont, Pkg, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and the local NameCert row.

Combining these three routes gives a single factorization from the completion diagonal packet to the Real$^{\uparrow}$ seal. Since each cited route excludes quotient streams, selected limits, host real equality, trichotomy, ambient completeness, hidden tail oracles, and unbounded schedules, the assembled carrier reaches the seal only through the finite schedule and null-tail rows named in the statement. □

Theorem 360.35 (RealCauchyCompletion tail schedule budget exhaustion)**.** *Every accepted* RealCauchyCompletion *carrier whose diagonal packet consumes* TailCofinalitySchedule$^{\uparrow}$*,* RegularCauchyTailBudget$^{\uparrow}$*,* RegularCauchyTailFusion NullSequence$^{\uparrow}$*, and an accepted* RealObservationBudget$^{\uparrow}$ *finite request bundle has a single consumer-visible route through exactly the regular-family row, unary modulus, diagonal packet, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger, tail-budget row, tail-fusion shared-threshold row, null-tail zero package, observation-budget precision rows,* Real$^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and local* NameCert*.*

The route exports no quotient stream equality, selected limit, host real equality, trichotomy, countable choice, or ambient completeness.

Proof. Begin with the accepted completion carrier and apply Definition 360.34. The diagonal packet is already forced through the regular-family row, unary modulus, TailCofinalitySchedule$^{\uparrow}$ finite-window route, NullSequence$^{\uparrow}$ zero-tail packet, DyadicRatCore$^{\uparrow}$ ledger, StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal, hsame, Cont, Pkg, and local NameCert.

Next apply Definition 360.33. It refines the same seal-facing route by making the RegularCauchyTailFusion$^{\uparrow}$ shared-threshold row and the RegularCauchyTailBudget$^{\uparrow}$ budget row explicit before the completion seal is read.

Apply the budgeted common-refinement theorem Definition 360.28. It supplies the finite consumer window for the accepted RealObservationBudget$^{\uparrow}$ request bundle. Read it together with the tail-budget coherence theorem Definition 360.31; this keeps each precision row inside the displayed StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, hsame, Cont, Pkg, and local naming route.

Now use the synchronized null-tail consumer theorem Definition 360.30. It identifies the same finite window with the NullSequence$^{\uparrow}$ zero-tail package, so the route has one shared carrier packet rather than separate schedule, budget, fusion, null-tail, and observation-budget exits.

Finally apply the RealObservationBudget$^{\uparrow}$ non-escape theorem Definition 611.6. Intersecting the exclusions from the schedule factorization, fusion-budget consumer, budgeted common refinement, synchronized null-tail consumer, and observation-budget row removes quotient stream equality, selected limits, host real equality, trichotomy, countable choice, and ambient completeness from the exported surface. What remains is exactly the finite consumer-visible route named in the statement. □

Theorem 360.36 (RealCauchyCompletion bounded-monotone witness consumer)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose regular-family input is supplied by an accepted* BoundedMonotoneCauchyWitne *packet reads its completion-facing* Real$^{\uparrow}$ *seal through exactly the bounded source row, scheduled* StreamName$^{\uparrow}$ *windows, Cauchy witness modulus, dyadic tolerance ledger, located trap,* RegSeqRat$^{\uparrow}$ *readback, diagonal completion packet, observation-budget request bundle, local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The route is completion-facing only in this displayed sense. It exports no selected limit, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient completeness principle.

Proof. Start with the accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ packet. The completion boundary theorem Definition 535.30 keeps every completion-facing read inside the bounded source, regular-rational handoff, stream schedule, Cauchy witness, dyadic ledger, located trap, Real$^{\uparrow}$ seal, transport, route, provenance, and local naming rows.

Next use the tail-window service theorem Definition 535.25. It forces each tolerance request to pass through the scheduled StreamName$^{\uparrow}$ windows, the Cauchy witness modulus, and the dyadic tolerance ledger before the located trap or any seal-facing row can be read.

Now apply the Real seal handoff theorem Definition 535.12. This places the Real$^{\uparrow}$ seal after the bounded source row, RegSeqRat$^{\uparrow}$ readback, scheduled windows, Cauchy witness, dyadic ledger, and located trap, so the completion carrier receives a displayed regular-family input rather than an external limiting object.

Finally compose this input route with the RealCauchyCompletion$^{\uparrow}$ consumer rows. The diagonal handoff, downstream consumer handoff, and tail schedule budget exhaustion theorems Definition 360.3, Definition 360.20, and Definition 360.35 add the diagonal completion packet, observation-budget request bundle, BHist carrier surface, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert rows. Since none of the cited packets contains a coordinate for selected limits, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient completeness, the assembled completion-facing witness route exposes exactly the rows named in the statement. □

Theorem 360.37 (RealCauchyCompletion uniform criterion seal factorization)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose diagonal packet consumes an accepted* UniformCauchyCriterion$^{\uparrow}$ *row and a* CauchyLimitSeal$^{\uparrow}$ *row reaches its* Real$^{\uparrow}$ *seal only through the same finite seal route. That route consists of the regular-family row, unary modulus, uniform shared-tail threshold, Cauchy-limit seal selector, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

The factorization exports no quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness principle.

Proof. Start with the accepted completion carrier and apply Definition 360.3. The diagonal packet already routes the regular-family input, unary modulus, finite windows, regular-rational readback,

dyadic ledger, transport rows, continuation routes, package provenance, and local naming rows before the completion-facing seal is read.

Use Definition 360.18 and Definition 360.35. They keep the public $\mathsf{Real}^{\uparrow}$ seal on the displayed completion route and make the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows explicit.

Attach the uniform criterion side through Definition 435.4 and Definition 435.68. The accepted criterion contributes the shared family threshold and its refined finite-window route, not a quotient family or selected limiting object.

Now attach the Cauchy-limit seal side through Definition 386.30 and Definition 386.5. These rows pull the uniform criterion back to the same scheduled windows and require the seal selector, dyadic observation ledger, regular readback, transports, routes, provenance, and local naming rows before any $\mathsf{Real}^{\uparrow}$ consumer read.

The combined route is therefore one finite BEDC packet: regular family, unary modulus, uniform shared-tail threshold, Cauchy-limit selector, windows, readback, dyadic ledger, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The non-escape clauses in the completion, uniform-criterion, and Cauchy-limit-seal rows exclude quotient streams, selected limits, host real equality, trichotomy, countable choice, and ambient completeness from the exported seal. □

Theorem 360.38 (RealCauchyCompletion witness-ledger classifier lock)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier whose completion-facing consumer also reads an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet* $L = (Q, B, S, K, H, C, P, N)$ *reaches the public* $\mathsf{Real}^{\uparrow}$ *seal through the completion-seal classifier only by the shared finite route*

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

The shared route is read together with the $\mathsf{RealCauchyCompletion}^{\uparrow}$ *regular-family row, unary modulus, diagonal packet,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

The consumer-side lock is relative to the witness-ledger surface of chapter 902 and the completion-seal classifier surface of chapter 900. It exports no quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.

Proof. Start with the accepted $\mathsf{RealCauchyCompletion}^{\uparrow}$ carrier. The public real-name export theorem Definition 360.18 keeps the public $\mathsf{Real}^{\uparrow}$ read on the displayed completion route, and Definition 360.37 factors that route through the regular-family row, unary modulus, diagonal packet, finite windows, regular-rational readback, dyadic ledger, transports, continuations, provenance, and local naming rows.

Now read the accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ packet. The Real-completion handoff Definition 902.9 places every Real-facing ledger consumer on the displayed completion route. The synchronizer-classifier lock Definition 902.14 orders that route as $Q \longmapsto B \longmapsto S \longmapsto K$, and the ledger-exhaustion theorem Definition 902.12 removes any public coordinate outside Q, B, S, K, H, C, P, N and the structural rows.

Finally identify the two consumer paths at the shared classifier row K. The RealCauchyCompletion path supplies the completion-facing public $\mathsf{Real}^{\uparrow}$ seal, while the witness ledger supplies the same finite budget, observation, synchronizer, and classifier route. Since both cited chapters expose only the displayed BEDC rows, the combined consumer cannot export quotient stream equality, host real equality, a selected external limit, a countable-choice row, or an ambient completeness theorem. □

Theorem 360.39 (RealCauchyCompletion witness-ledger uniform seal route)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose completion-facing route is also read through an accepted* CauchyWitnessLedger$^{\uparrow}$ *packet and an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet factors through one finite seal route. The route consists exactly of the regular-family row, unary modulus, diagonal packet, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* CauchyWitnessLedger$^{\uparrow}$ *budget, synchronizer, and classifier rows,* UniformCauchyCriterion$^{\uparrow}$ *shared-threshold rows, the public* Real$^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The combined route exports no quotient stream, selected host limit, host real equality, countable-choice row, second family-limit route, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the completion-facing seal. The uniform criterion seal factorization theorem Definition 360.37 places the accepted RealCauchyCompletion$^{\uparrow}$ carrier on the finite regular-family, unary-modulus, diagonal-packet, window, readback, dyadic-ledger, transport, continuation, package, naming, and public Real$^{\uparrow}$ seal route.

Next read the witness-ledger side. The witness-ledger classifier lock Definition 360.38 identifies the completion-seal classifier with the shared finite route through the ledger budget, observation, synchronizer, and classifier rows, and Definition 902.55 exhausts downstream completion consumers at those displayed ledger rows.

Finally attach the uniform criterion consumer surface. The downstream Real consumer export Definition 435.21 and the Real-completion budget lock Definition 435.131 keep every RealCauchyCompletion$^{\uparrow}$ read behind the same shared-threshold and finite-window rows. Intersecting the three non-escape clauses leaves no coordinate for quotient streams, selected host limits, host real equality, countable choice, a second family-limit route, or ambient Real$^{\uparrow}$ completeness, so the sole exposed route is the finite BEDC packet named in the statement. □

Theorem 360.40 (RealCauchyCompletion shared-tail modulus consumer route)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose diagonal packet consumes an accepted* CauchySharedTailModulus$^{\uparrow}$ *row reaches the public* Real$^{\uparrow}$ *seal through a single finite route. The route consists of the regular-family row, unary modulus, shared-tail threshold, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The route is relative to the shared-tail modulus surface of chapter 983. It exports no quotient streams, selected limits, host real equality, countable-choice row, or ambient completeness principle.

Proof. Project the accepted completion carrier. The public real-name export theorem, Definition 360.18, keeps the Real$^{\uparrow}$ seal on the displayed completion path, while the diagonal handoff and downstream consumer handoff theorems bind that path to the regular-family row, unary modulus, finite windows, regular-rational readback, dyadic ledger, transports, continuations, provenance, and local naming rows.

Now attach the accepted CauchySharedTailModulus$^{\uparrow}$ packet. Its consumer contribution is the shared-tail threshold for the same tail comparison, not a separate limit selector or quotient-stream equality.

Combining the two projections gives one displayed route from the completion diagonal packet to the public seal. The exclusions in the completion and shared-tail surfaces remove quotient streams, selected limits, host real equality, countable choice, and ambient completeness from the exported carrier. □

Theorem 360.41 (RealCauchyCompletion dyadic tail-radius ledger route)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose diagonal packet consumes an accepted* DyadicTailRadiusLedger$^{\uparrow}$ *row reaches the public*

Real$^{\uparrow}$ *seal only after the dyadic tail-radius ledger has been displayed. The finite route is the regular-family row, unary modulus, shared tail request, dyadic tail-radius ledger, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The route is relative to the dyadic ledger surface of chapter 757. It exports no quotient-stream representative, selected limit, host real equality, countable-choice row, or ambient completeness theorem.

Proof. Start from the displayed completion route supplied by Definition 360.35. This already places the public seal after the regular-family row, unary modulus, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ ledger, diagonal packet, transports, continuations, provenance, and local naming rows.

Insert the DyadicTailRadiusLedger$^{\uparrow}$ packet at the tail-radius step. Its ledger row records the dyadic tolerance and radius accounting used by the same finite windows, so the completion consumer reads a displayed dyadic budget rather than an external completeness witness.

The resulting path is still the completion path. Since neither the completion route nor the dyadic ledger surface contains a coordinate for quotient representatives, selected limits, host real equality, countable choice, or ambient completeness, the public seal can only be reached through the rows named in the statement. □

Theorem 360.42 (RealCauchyCompletion CauchyCriterion sibling route)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose diagonal packet consumes accepted* CauchySharedTailModulus$^{\uparrow}$*,* DyadicTailRadiusLedger$^{\uparrow}$*, and* CauchyCriterion$^{\uparrow}$ *rows reaches the public* Real$^{\uparrow}$ *seal only through the combined finite rows of the regular-family row, unary modulus, shared-tail threshold, dyadic tail-radius ledger, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The combined route is relative to chapter 983, chapter 757, and chapter 392. It exports no quotient streams, selected limits, host real equality, countable-choice row, or ambient completeness principle.

Proof. First apply the shared-tail modulus route above. It identifies the completion seal path with the regular-family row, unary modulus, shared-tail threshold, scheduled windows, readback, dyadic ledger, structural transport, provenance, and local naming rows.

Next apply the dyadic tail-radius ledger route above. It shows that the same scheduled windows must expose their dyadic radius accounting before the public Real$^{\uparrow}$ seal is consumed.

Finally attach the sibling CauchyCriterion$^{\uparrow}$ packet. Its role is to certify the Cauchy-condition route for the displayed shared threshold and dyadic ledger, not to import a selected limit or ambient completeness theorem.

The three sibling routes therefore assemble one completion-seal consumer packet. Because each component route is confined to BHist, hsame, Cont, Pkg, and NameCert rows and excludes quotient streams, selected limits, host real equality, countable choice, and ambient completeness, the carrier reaches the public seal only through the finite rows listed in the statement. □

Theorem 360.43 (RealCauchyCompletion diagonal seal uniformity)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose diagonal packet consumes the same displayed* CauchyCriterion$^{\uparrow}$ *row and the same singleton or fixed finite-family* UniformCauchyCriterion$^{\uparrow}$ *rows factors through one finite completion route. The route consists of the regular-family row, shared unary modulus, dyadic tolerance, tail ledger, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The diagonal seal therefore exports no quotient streams, selected limits, host real equality, countable choice, ambient completeness principle, or second family-limit route. A downstream $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer reads only the displayed finite seal path.*

Proof. First apply Definition 360.42. It places the completion seal behind the regular-family row, unary modulus, finite stream windows, regular-rational readback, dyadic ledger, $\mathsf{Real}^{\uparrow}$ seal, and structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use Definition 392.29. The displayed $\mathsf{CauchyCriterion}^{\uparrow}$ packet and the singleton or fixed finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet share the same modulus, dyadic tolerance, tail ledger, stream-name source, regular-rational tail, and real-seal rows, so the family route restricts to the same finite CauchyCriterion path.

Finally apply Definition 435.21. Its downstream consumer clause keeps the $\mathsf{Real}^{\uparrow}$ seal on the uniform finite-family route. Intersecting the non-escape clauses from the three cited routes removes quotient streams, selected limits, host real equality, countable choice, ambient completeness, and any second family-limit route from the diagonal seal. □

Theorem 360.44 (RealCauchyCompletion StreamName RegSeqRat Real seal correspondence)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *packet whose regular family, unary modulus, diagonal packet, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows are displayed determines exactly the same public finite-window seal boundary as the composed handoff*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Conversely, any consumer that uses this composed handoff must factor through the displayed $\mathsf{RealCauchyCompletion}^{\uparrow}$ *regular-family, diagonal, stream-window, readback, dyadic-ledger, and real-seal rows before it reads a limit-seal boundary.*

The correspondence is relative to the finite-name, regular-sequence, dyadic, and real-name surfaces of chapter 126, chapter 301, chapter 291, and chapter 38. It exports no quotient stream, selected limit, host real equality, countable-choice row, or ambient completeness principle.

Proof. First use Definition 360.9. It places the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback on the same tail-seal route as the accepted completion packet, so the stream-to-regular handoff is not a separate public boundary.

Next apply Definition 360.18. This exports the public $\mathsf{Real}^{\uparrow}$ seal only through the displayed completion packet and the finite rows carried by the sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ chapters.

Use Definition 360.19 to identify the input side of the route with the regular-family row and unary modulus. Then apply Definition 360.43, which keeps the diagonal packet and the real-seal consumer on one finite completion path.

For the converse, take any consumer of the composed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ handoff. The cited factorization and export theorems force its stream windows, regular-rational readback, dyadic ledger, and real seal to pass through the displayed completion rows before a limit-seal boundary is available. The sibling chapters supply exactly those finite public surfaces, so no consumer can bypass the $\mathsf{RealCauchyCompletion}^{\uparrow}$ rows while still reading the advertised boundary. □

Theorem 360.45 (RealCauchyCompletion four-face exit uniqueness)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public completion read that exports a* $\mathsf{Real}^{\uparrow}$ *seal through the L10 phase route has a unique finite support spine, up to displayed* hsame *transport and common-refinement comparison of finite windows.*

The spine consists of

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the local BHist*,* Cont*,* Pkg*, and* NameCert *rows carried by the accepted completion packet.*

Any purported public route that omits one of the four faces, duplicates a second seal boundary, or replaces a face by quotient stream equality, selected limits, host real equality, countable choice, or ambient completeness is not classifier-equivalent to the accepted public $\mathsf{RealCauchyCompletion}^{\uparrow}$ *route.*

Proof. Begin with the StreamName-RegSeqRat-Real seal correspondence, Definition 360.44. It identifies the public seal boundary with the composed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ handoff and keeps the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger on the same displayed completion path.

Next apply current four-face phase support, Definition 360.52, and terminal support minimality, Definition 360.51. These results force every accepted L10 phase read to display the four faces and the local BHist, hsame, Cont, Pkg, and NameCert rows; removing or replacing any one of them collapses the route rather than producing another public completion witness.

Now compare two accepted presentations. Pull both to a certified finite common refinement of their stream windows and use Definition 360.56. The pulled-back routes determine the same BEDC completion witness, so the only variation left is the displayed hsame transport and the finite-window cover used to present the same support spine.

It remains to exclude alternate exits. Terminal support defect localization, Definition 360.54, places any missing face, missing support row, duplicated seal boundary, or substituted quotient, selected-limit, host-equality, choice, or completeness coordinate at a named defect. Such a presentation is therefore not classifier-equivalent to the accepted public route. □

Theorem 360.46 (RealCauchyCompletion tail-budget uniform seal boundary)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *packet whose finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail selector, dyadic budget row, and* $\mathsf{Real}^{\uparrow}$ *endpoint seal satisfy the StreamName-RegSeqRat tail-budget uniformity hypotheses has a uniform completion-facing seal boundary. Every later finite window accepted by the same pointwise* $\mathsf{StreamName}^{\uparrow}$ *classifier factors through the same displayed completion route.*

Consequently the completion seal exports no new limit selector, quotient stream, host equality, countable-choice row, ambient completeness datum, or second completion-facing boundary beyond the displayed regular-family row, unary modulus, diagonal packet, finite stream windows, regular-rational readback, dyadic budget ledger, $\mathsf{Real}^{\uparrow}$ *endpoint seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with Definition 360.44. It identifies the public completion seal boundary with the composed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ handoff, relative to the displayed regular-family, diagonal, stream-window, readback, dyadic-ledger, and real-seal rows.

Next apply Definition 126.105. For every later finite window accepted by the same pointwise StreamName classifier, the RegSeqRat tail selector and the RealUp budget consumer read the same tail-budget certificate. Thus enlarging the finite window changes only the visible window ledger.

Transport this uniform handoff back through Definition 360.9 and Definition 360.18. The first theorem keeps the StreamName windows and RegSeqRat readback on the completion tail-seal route, and the second theorem exposes the public RealUp seal only through the displayed completion rows.

The non-escape clauses of the cited routes now intersect at the same boundary: the later windows share one completion-facing route and cannot introduce a fresh limit selector, quotient stream, host equality, countable-choice row, ambient completeness datum, or second seal boundary. □

Theorem 360.47 (RealCauchyCompletion witness-ledger observation-budget route)**.** *Every accepted* RealCauchyCompletion$^{\uparrow}$ *carrier whose completion-facing consumer is read through both an accepted* CauchyWitnessLedger$^{\uparrow}$ *packet and an accepted* RealObservationBudget$^{\uparrow}$ *packet reaches the public* Real$^{\uparrow}$ *seal only through one shared finite route. The route consists exactly of the regular-family row, unary modulus, diagonal packet, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger, witness-ledger budget, synchronizer, and classifier rows, observation-budget precision, window, and disclosure rows,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The route exports no quotient equality of regular streams, host real equality, selected limit, hidden unbounded observation schedule, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. First apply the witness-ledger classifier lock Definition 360.38. It places the completion-facing public seal behind the regular-family row, unary modulus, diagonal packet, finite windows, regular-rational readback, dyadic ledger, and witness-ledger budget, synchronizer, and classifier rows. The uniform seal route Definition 360.39 keeps this consumer path on the same finite seal boundary.

Next apply the budgeted common-refinement theorem Definition 360.28. The accepted RealObservationBudget$^{\uparrow}$ packet supplies the precision rows, finite StreamName$^{\uparrow}$ windows, dyadic readbacks, regular handoffs, seal boundary, hsame transports, Cont replay, Pkg provenance, and local NameCert rows for the same request surface.

Finally intersect the observation-budget window coverage, real-seal boundary, and non-escape rows from Definition 611.3, Definition 611.5, and Definition 611.6. The two readings therefore share one finite completion-facing route, and all omitted objects stay outside the public Real$^{\uparrow}$ seal. □

Theorem 360.48 (RealCauchyCompletion finite-window completion coverage)**.** *For every accepted* RealCauchyCompletion$^{\uparrow}$ *packet and every finite* ProbeBundle *of unary precision requests, the completion-facing* Real$^{\uparrow}$ *seal read is exhausted by the displayed finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback rows,* DyadicRatCore$^{\uparrow}$ *observation ledger, diagonal packet,* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The exhausted route is the same finite-name path from StreamName$^{\uparrow}$ *through* RegSeqRat$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *into* Real$^{\uparrow}$*. It exports no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness coordinate, or completion principle beyond those displayed certificate rows.*

Proof. First apply Definition 360.12. It turns the finite ProbeBundle into one consumer surface whose pointwise requests pass through the displayed modulus rows, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback rows, DyadicRatCore$^{\uparrow}$ ledgers, and the structural BHist, hsame, Cont, Pkg, and NameCert data.

Next use the StreamName handoff and the sibling finite-window coverage rows for regular sequences, dyadic observations, and real-name seals. These identify each window read as the same finite route from StreamName$^{\uparrow}$ to RegSeqRat$^{\uparrow}$, then through DyadicRatCore$^{\uparrow}$ observation data into the public Real$^{\uparrow}$ seal.

Finally apply Definition 360.44 together with the non-escape boundary of Definition 360.4. The composed route is therefore the only completion-facing seal read available to the accepted packet,

and quotient streams, selected limits, host real equality, countable choice, ambient completeness, and any hidden completion coordinate remain outside the exported surface. □

Theorem 360.49 (RealCauchyCompletion monadic four-face exit correspondence)**.** *For every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *packet whose visible rows are the regular-family row, unary modulus, diagonal packet, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, the public L10 four-face exit coincides with the finite completion-transformer route read through* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*.*

Both readings expose exactly the same four roadmap objects $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Neither reading adds quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, or bridge-axis evidence.*

Proof. Start with Definition 360.44. It identifies the public completion seal with the finite handoff through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while keeping the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger on the same displayed completion route.

Apply Definition 360.48. Every finite precision request is exhausted by the displayed windows, readbacks, dyadic ledger, diagonal packet, real seal, and structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the L10 exit has no additional coordinate.

Now read the same boundary through chapter 378. Its unit, bind, public seal, finite-window exhaustion, and standard completion-boundary rows use the same four roadmap objects as the completion transformer. Intersecting the non-escape clauses on both sides gives one shared four-face exit and excludes quotient streams, selected limits, host real equality, countable choice, ambient completeness, and bridge-axis evidence. □

Theorem 360.50 (RealCauchyCompletion monadic unit-bind locality)**.** *For every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *packet whose public read is compared with an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit-and-bind route, the unit packet, bind packet, and downstream completion consumer factor through the same displayed L10 four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The shared route carries the same BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows as the direct* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *read.*

The comparison is local to that route. It introduces no second completion source, quotient stream, selected limit, host equality, countable-choice row, ambient completeness principle, or bridge-axis evidence.

Proof. Begin with the monadic four-face exit correspondence of Definition 360.49. It identifies the direct $\mathsf{RealCauchyCompletion}^{\uparrow}$ public read with the $\mathsf{CauchyCompletionMonad}^{\uparrow}$ completion-transformer read on the same four roadmap objects.

Next apply the Kleisli source triangle Definition 378.50. The unit source, bind source, and downstream consumer source are forced through the displayed finite route, so the monadic interface does not add another completion origin.

Now use the finite-route correspondence of Definition 378.49. The unit and bind packets inherit the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows already visible on the direct completion read, and the downstream consumer reads those rows rather than a separate quotient or selected-limit coordinate.

Finally intersect this monadic route with the terminal support minimality and non-escape rows local to this chapter. The only available unit-bind support is the displayed L10 four-face route,

while second completion sources, quotient streams, selected limits, host equality, countable choice, ambient completeness, and bridge-axis evidence remain outside the accepted public surface. □

Theorem 360.51 (RealCauchyCompletion terminal support minimality)**.** *Every accepted public L10 completion read through* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *retains the four displayed faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

It also retains the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by the completion packet. Deleting or replacing any one of these four faces, or deleting any one of the displayed structural rows, collapses the public completion route rather than producing another BEDC completion witness.*

The minimality claim is internal to the public L10 completion surface. It does not identify quotient streams, choose limiting representatives, import host real equality, assume countable choice, assert ambient completeness, or add bridge-axis evidence.

Proof. Start with the composed public seal handoff of Definition 360.44. That result identifies the completion boundary with the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ route, while the dyadic ledger remains the finite observation support for the same read. Thus the four faces appear as part of one route, not as replaceable decorations around the carrier.

Next apply Definition 360.48. Every finite precision request is exhausted by the displayed windows, readbacks, dyadic ledger, diagonal packet, real seal, and the structural BHist, hsame, Cont, Pkg, and NameCert rows. Removing one of these rows removes a named step used by the public request surface.

Now compare the same terminal read with the monadic four-face exit correspondence of Definition 360.49. The $\mathsf{CauchyCompletionMonad}^{\uparrow}$ reading and the direct $\mathsf{RealCauchyCompletion}^{\uparrow}$ reading expose the same four roadmap objects, so replacing a face does not yield another advertised completion exit; it changes the route outside the correspondence.

Finally use the non-escape boundary of Definition 360.4. Quotient streams, selected limits, host real equality, countable choice, ambient completeness, and bridge-axis evidence are excluded from the carrier surface. With no excluded object available as substitute support, any deletion or replacement collapses the public L10 completion route rather than yielding a second BEDC completion witness. □

Theorem 360.52 (RealCauchyCompletion current four-face phase support)**.** *A Phase Real and Completion exit read through* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *is valid only when the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

are all present as the displayed completion route. The same read must also display the BHist *histories, componentwise* hsame *transports,* Cont *replay rows,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *support rows carried by the accepted completion packet.*

Thus $\mathsf{RealCauchyCompletion}^{\uparrow}$ *supplies the roadmap-facing Phase Real and Completion exit exactly as a four-face BEDC support condition. It supplies no quotient stream equality, host real equality, selected limit, countable-choice row, ambient completeness principle, or bridge-axis evidence in place of any one of the four faces or support rows.*

Proof. First apply Definition 360.51. It says that every accepted public L10 completion read through $\mathsf{RealCauchyCompletion}^{\uparrow}$ retains the four displayed faces and the visible BHist, hsame, Cont,

Pkg, and NameCert support rows, and that deleting or replacing any one of them collapses the completion route.

Next use Definition 360.49. The direct RealCauchyCompletion$^{\uparrow}$ reading and the CauchyCompletionMonad$^{\uparrow}$ completion-transformer reading expose the same four roadmap objects, so the local completion route is already the Phase Real and Completion route rather than a separate completion-only support condition.

Finally compose with the Real-side exit criterion Definition 38.732. That criterion accepts a Phase Real and Completion endpoint read exactly when DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and the local Real$^{\uparrow}$ seal appear in order with their local records and the shared BHist, hsame, Cont, Pkg, and NameCert rows.

The excluded coordinates are removed by the non-escape clauses in the three cited results. Quotient streams, host real equality, selected limits, countable choice, ambient completeness, and bridge-axis evidence are not support rows of the four-face packet, so none can serve as substitute evidence for the phase exit read. □

Theorem 360.53 (RealCauchyCompletion four-face status noncollapse)**.** *A Phase Real and Completion exit read through* RealCauchyCompletion$^{\uparrow}$ *whose route displays*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows reads* closurestatus*,* formalstatus*,* leantarget*, and* bridgestatus *only from the four local face records.*

The same route has no aggregate formalstatus row, replacement leantarget, route-level bridge credential, standard bridge row, selected limit, quotient stream equality, host real equality, countable-choice row, metric-completeness datum, ambient completeness principle, or Real$^{\uparrow}$ *completeness coordinate.*

Proof. First apply Definition 360.52. It identifies the Phase Real and Completion read through RealCauchyCompletion$^{\uparrow}$ with the displayed four faces and with the shared BHist, hsame, Cont, Pkg, and NameCert support rows.

Next use Definition 38.732 and Definition 38.792. These results read the roadmap exit object-wise from the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and local Real$^{\uparrow}$ face records. Therefore the closure, verification, Lean-target, and bridge-status rows remain face-local reports.

It remains to rule out aggregate status substitutes. Apply Definition 38.833 and Definition 38.834. They block a route-level formalstatus row, replacement leantarget, aggregate bridge credential, or standard bridge row from replacing the four local face reports.

Finally apply the non-escape boundary of Definition 360.4. Quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, and bridge-axis evidence are outside the accepted completion carrier. Together with the face-local status reading, this also excludes metric-completeness data and a separate Real$^{\uparrow}$ completeness coordinate from the current four-face route. □

Theorem 360.54 (RealCauchyCompletion terminal support defect localization)**.** *Let a Phase Real and Completion exit read through* RealCauchyCompletion$^{\uparrow}$ *be presented as a public BEDC completion route. If that presentation omits or replaces any one of the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

or omits or replaces any visible BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *support row carried by the accepted completion packet, then the failure is witnessed at that named face or support row. The presentation is therefore not an accepted BEDC completion route through* RealCauchyCompletion$^{\uparrow}$*.*

No quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness principle, or bridge-axis evidence can repair the failed presentation or move the failure away from the omitted or replaced face or support row.

Proof. Apply the terminal support minimality theorem, Definition 360.51. It states that every accepted public L10 completion read retains the four displayed faces and the visible BHist, hsame, Cont, Pkg, and NameCert support rows, and that deleting or replacing any one of them collapses the route.

Next apply the monadic four-face exit correspondence of Definition 360.49. The direct $\mathsf{RealCauchyCompletion}^{\uparrow}$ read and the $\mathsf{CauchyCompletionMonad}^{\uparrow}$ read expose the same four roadmap objects, so a defect in one of those faces cannot be transferred to a different completion-transformer coordinate.

Now use Definition 360.52 together with the Real-side criterion Definition 38.732. These results identify the Phase Real and Completion endpoint with the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route and with the same local BHist, hsame, Cont, Pkg, and NameCert rows. Hence the failed presentation has a named local witness: exactly the face or support row that is absent or replaced.

Finally apply the non-escape boundary of Definition 360.4. Quotient streams, selected limits, host equality, countable choice, ambient completeness, and bridge-axis evidence are outside the accepted carrier surface. Since no excluded coordinate is available as substitute support, the malformed presentation is rejected at the located face or row and is not a BEDC completion route through $\mathsf{RealCauchyCompletion}^{\uparrow}$. □

Theorem 360.55 (RealCauchyCompletion terminal support refinement invariance)**.** *Let $\mathcal{P}$ be a coarse public L10 completion route through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, and let $\mathcal{Q}$ be a finite certified refinement of $\mathcal{P}$. Suppose both presentations display the same four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and suppose every refined member, refined overlap, and comparison map preserves the same visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by the accepted completion packet. Then the route accepted from $\mathcal{P}$ and the route accepted from $\mathcal{Q}$ determine the same BEDC completion witness through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*.*

This comparison changes only the finite cover presentation of the terminal four-face route. It introduces no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, or bridge-axis evidence.*

Proof. First apply terminal support minimality, Definition 360.51, and current four-face phase support, Definition 360.52. Both the coarse route and the refined route must therefore be read from the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the same visible BHist, hsame, Cont, Pkg, and NameCert support rows.

Next localize possible differences by Definition 360.54. Since every refined member, overlap, and comparison map preserves the named faces and support rows, no defect can be moved to a new completion coordinate. The only remaining difference is the finite cover by which the same terminal route is presented.

Apply the DyadicRatCore finite-cover comparison, Definition 291.120. It identifies the dyadic ledger and the four-face exit packet obtained from the coarse cover with the one obtained from the refined cover, because both presentations preserve the displayed DyadicRatCore, StreamName, RegSeqRat, Real, and support rows.

Finally use RegSeqRat common-tail cocycle descent, in particular Definition 301.211. The common-tail members, overlaps, and cocycle rows glue to the same Real seal-visible boundary and the same Phase $\mathsf{Real}^{\uparrow}$ and Completion audit-threshold coordinate. Thus the refined presentation changes only the finite support cover, while the accepted public completion witness through $\mathsf{RealCauchyCompletion}^{\uparrow}$ remains the same. □

Theorem 360.56 (RealCauchyCompletion terminal support common-refinement comparison)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be accepted public L10 completion routes through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. Suppose there is a certified finite common refinement $\mathcal{R}$ of the two presentations such that, after pullback to $\mathcal{R}$, both routes display the same four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and preserve the same visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by the accepted completion packet. Then $\mathcal{P}$ and $\mathcal{Q}$ determine the same BEDC completion witness through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*.*

The comparison is entirely internal to the certified common finite support. It introduces no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or bridge-axis evidence.*

Proof. First pull the two accepted presentations back to the common refinement $\mathcal{R}$. By terminal support minimality, Definition 360.51, both pulled-back routes are still read from the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and from the same visible BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply terminal support defect localization, Definition 360.54. Since the hypotheses preserve every named face and every support row after pullback to $\mathcal{R}$, a disagreement between the two accepted presentations cannot be relocated to a hidden completion coordinate.

Use terminal support refinement invariance, Definition 360.55, on the refinement from $\mathcal{P}$ to $\mathcal{R}$ and again on the refinement from $\mathcal{Q}$ to $\mathcal{R}$. Thus each original route determines the same BEDC completion witness as its pulled-back route on the certified common support.

It remains to identify the pulled-back terminal support itself. Apply Definition 126.221 to form the shared StreamName finite window, then Definition 126.405 to read the common terminal Real seal over that window. Finally use Definition 301.211 to keep the RegSeqRat common-tail atlas and dyadic ledger invariant over the same pullback.

The non-escape clauses in the preceding results rule out all substitute coordinates. Quotient streams, selected limits, host equality, countable choice, ambient completeness, metric-completeness data, aggregate status rows, bridge rows, and bridge-axis evidence are not part of the visible terminal support, so they cannot distinguish the two accepted completion witnesses. □

Theorem 360.57 (RealCauchyCompletion terminal support refinement composition)**.** *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be public L10 completion routes through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. Suppose $\mathcal{Q}$ is a finite certified refinement of $\mathcal{P}$, and $\mathcal{R}$ is a finite certified refinement of $\mathcal{Q}$. Suppose also that every refined member, overlap, and comparison map in both refinements preserves the same four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the same visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by the accepted completion packet. Then $\mathcal{P}$ and $\mathcal{R}$ determine the same BEDC completion witness through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*.*

The successive comparison remains internal to the terminal four-face route. It introduces no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or bridge-axis evidence.*

Proof. First apply terminal support refinement invariance, Definition 360.55, to the certified refinement $\mathcal{P} \to \mathcal{Q}$. The coarse route $\mathcal{P}$ and the intermediate route $\mathcal{Q}$ therefore read the same accepted BEDC completion witness from the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and the same visible support rows.

Apply the same theorem again to the certified refinement $\mathcal{Q} \to \mathcal{R}$. Since the second refinement preserves the same faces, overlaps, comparison maps, and BHist, hsame, Cont, Pkg, and NameCert rows, the intermediate route $\mathcal{Q}$ and the final route $\mathcal{R}$ also determine one completion witness.

It remains to exclude a new coordinate appearing only at the intermediate stage. Use terminal support defect localization, Definition 360.54. Any proposed difference between the two comparisons would have to be witnessed by an omitted or replaced face or support row, but the hypotheses preserve every named face, overlap, comparison map, and support row across both refinements.

Finally apply the non-escape boundary of Definition 360.4. Quotient streams, selected limits, host equality, countable choice, ambient completeness, and bridge-axis evidence are outside the accepted carrier surface, and the status and bridge rows cannot aggregate outside the four local face records. Thus the two certified refinement steps compose without changing the BEDC completion witness determined through $\mathsf{RealCauchyCompletion}^{\uparrow}$. □

Theorem 360.58 (RealCauchyCompletion terminal support refinement cocycle)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be two finite certified terminal-support refinements of one public L10 completion route through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. Suppose both refinements preserve the same four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and the same visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows carried by the accepted completion packet. Suppose moreover that the two refinements carry the same* $\mathsf{RegSeqRat}^{\uparrow}$ *common-tail pairwise packets on every retained overlap, and that their triple-overlap rows satisfy the same common-tail cocycle over the terminal four-face support. Then $\mathcal{P}$ and $\mathcal{Q}$ determine the same BEDC completion witness through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*.*

The path through a finite refinement diagram is therefore independent of the certified terminal-support cover. The comparison introduces no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or bridge-axis evidence.*

Proof. First restrict both refinements to the displayed terminal support. By terminal support minimality, Definition 360.51, and terminal support defect localization, Definition 360.54, every accepted comparison must be witnessed on the named $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and on the same BHist, hsame, Cont, Pkg, and NameCert rows.

Next compare the dyadic and four-face ledgers. The finite-cover refinement invariance theorem Definition 291.120 identifies the DyadicRatCore exit ledger obtained from either certified support cover, since the endpoint, radius, stream-window, regular-sequence, Real, and support rows are the same.

Now descend the common-tail data. Apply Definition 301.191 to the shared pairwise packets and triple-overlap rows, and then use Definition 301.198. These two results identify the two refined RegSeqRat common-tail presentations as one terminal four-face cocycle boundary.

Finally pass from the common terminal boundary to the completion witness. The Real L10 terminal four-face correspondence, Definition 38.455, repacks the DyadicRatCore, StreamName, RegSeqRat, and Real rows as the accepted terminal four-face read, and Definition 38.458 forbids a second pullback over the same visible rows. Together with terminal support refinement composition, Definition 360.57, this shows that both finite paths determine the same BEDC completion witness through $\mathsf{RealCauchyCompletion}^{\uparrow}$.

The exclusion clauses follow from the cited support and terminality results. None of the comparison data carries a quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus row, aggregate bridge row, standard bridge row, or bridge-axis evidence, so no such coordinate can distinguish the two certified refinement paths. □

Theorem 360.59 (RealCauchyCompletion request-section terminal exhaustion)**.** *For every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier and every finite* ProbeBundle *of unary precision requests, the terminal completion-facing read is exhausted by one finite request-section route through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Any other request-section route that agrees on these four faces and support rows is classifier-equivalent to it after finite common refinement.*

The exhausted route exports no quotient stream, selected limit, host real equality, countable-choice row, ambient completeness principle, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, or bridge-axis evidence.*

Proof. Start with the finite probe bundle. The probe-bundle window coverage theorem, Definition 360.12, replaces the unary precision requests by a finite completion window without adding any completion coordinate beyond the accepted carrier surface.

Apply finite-window completion coverage, Definition 360.48. This identifies the window with the four-face request-section route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Now compare another route agreeing on those rows. Use terminal support common-refinement comparison, Definition 360.56, to pull the two request-section presentations to a certified finite common refinement. Then terminal support refinement cocycle, Definition 360.58, makes the resulting classifier independent of the finite support cover chosen along the way.

It remains to exclude hidden support. Terminal support defect localization, Definition 360.54, places any missing or substituted face or support row at the named local defect. The non-escape boundary, Definition 360.4, prevents quotient streams, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, aggregate status rows, aggregate bridge rows, and bridge-axis evidence from serving as replacement output. Thus the finite request-section route exhausts the accepted terminal completion read. □

Theorem 360.60 (RealCauchyCompletion terminal exit locality)**.** *Every accepted terminal exit read through* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *for the Phase Real and Completion boundary factors through exactly the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by the accepted completion packet. The locality assertion is facewise: the dyadic ledger, finite stream windows,*

regular-rational readback, and local real seal remain separate rows, and no route-level status or bridge coordinate replaces any one of them.

The terminal read therefore has no quotient-stream row, selected-limit row, host real equality row, countable-choice row, metric-completeness datum, aggregate formalstatus *row, aggregate bridge row, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the request-section terminal exhaustion theorem, Definition 360.59. It exhausts every finite unary request bundle by a route through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and through the listed support rows.

Apply terminal support minimality, Definition 360.51, and current four-face phase support, Definition 360.52. These results force the same four faces to remain present for any accepted terminal read and prevent a replacement face from presenting the same completion witness.

The face-local status reading is supplied by Definition 360.53 together with Definition 38.792. The closure, verification, Lean-target, and bridge-status information is read from the four component records rather than from an aggregate route row.

Finally use the sibling route inventory. The CauchyCriterion route, Definition 360.42, the Real-side terminal correspondence Definition 38.455, and the RegSeqRat consumer normal form Definition 301.175 all expose only the same finite rows. Their non-escape clauses exclude quotient streams, selected limits, host equality, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the terminal exit surface. □

Theorem 360.61 (RealCauchyCompletion terminal exit normal form)**.** *For an accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier, the terminal Phase Real and Completion exit certificate has the following normal form. It is a finite BEDC packet whose four visible faces are exactly*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and whose only shared support rows are BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. A consumer reads the dyadic observation ledger, then the finite stream-window schedule, then the regular-rational readback and common-tail rows, and finally the local* $\mathsf{Real}^{\uparrow}$ *seal through those support rows.*

Conversely, any accepted terminal packet exposing precisely those four faces with the same displayed support rows determines the same Phase Real and Completion exit certificate. The normal form excludes quotient streams, selected limits, host real equality, countable-choice rows, metric-completeness data, aggregate formalstatus *rows, aggregate bridge rows, standard bridge rows, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward direction, apply terminal exit locality, Definition 360.60. Every accepted terminal read has already been reduced to the four named faces and the shared support rows.

Next identify the order of the rows. The request-section terminal exhaustion theorem, Definition 360.59, supplies the finite request route, while Definition 126.372 and Definition 291.110 align the stream-window, dyadic-ledger, RegSeqRat, and Real rows with the Phase Real and Completion exit surface.

For the reverse direction, suppose a terminal packet exposes exactly the four faces and support rows listed in the statement. Apply the Real L10 terminal four-face correspondence, Definition 38.455, to repack the displayed rows as the accepted terminal $\mathsf{Real}^{\uparrow}$ read. Then use terminal pullback uniqueness, Definition 38.458, and the RegSeqRat four-face terminal pullback, Definition 301.58, to identify the same completion-facing boundary.

It remains to rule out additional coordinates in the normal form. Terminal support defect localization, Definition 360.54, makes any missing, duplicated, or substituted face a named local

defect. The four-face status noncollapse theorem, Definition 360.53, and the non-escape boundary, Definition 360.4, remove quotient streams, selected limits, host equality, countable choice, metric-completeness data, aggregate status rows, aggregate bridge rows, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the accepted packet. □

Theorem 360.62 (RealCauchyCompletion terminal exit support invariance)**.** *Let $\mathcal{P}$ and $\mathcal{Q}$ be accepted terminal exit presentations of the same* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier. Suppose both presentations expose the four local faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and preserve the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows after passage to a certified finite common refinement. Then $\mathcal{P}$ and $\mathcal{Q}$ determine the same terminal Phase Real and Completion exit certificate.*

The invariant certificate is exactly the normal form of Definition 360.61. It is unaffected by the choice of finite support cover and contains no quotient streams, selected limits, host real equality, countable-choice rows, metric-completeness data, aggregate formalstatus *rows, aggregate bridge rows, standard bridge rows, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First pass both presentations to the certified finite common refinement. Terminal support common-refinement comparison, Definition 360.56, identifies the pulled-back routes whenever the four faces and the shared support rows agree there.

Next remove dependence on the chosen finite support cover. Terminal support refinement invariance, Definition 360.55, and terminal support refinement cocycle, Definition 360.58, show that the path through the refinement diagram changes only the cover presentation, not the BEDC completion witness.

Now repack the common refined support by the terminal exit normal form, Definition 360.61. The repacked packet has exactly the DyadicRatCore, StreamName, RegSeqRat, Real, and support rows listed in the statement, so both original presentations read the same Phase Real and Completion certificate.

The exclusion clauses are inherited from the same normal form and from terminal support defect localization, Definition 360.54. Any alleged difference outside the displayed rows would have to be a quotient stream, selected limit, host equality, choice row, metric-completeness datum, aggregate status row, aggregate bridge row, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, and none of these is part of the accepted terminal support. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCauchyCompletion}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$-indexed regular Cauchy family, an explicit unary modulus, a $\mathsf{RealDiagonalCompletion}^{\uparrow}$ diagonal packet, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observations, a $\mathsf{Real}^{\uparrow}$ seal row, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is the public real-name export, standard bridge boundary, and limit-seal consumer surface through the displayed regular-family row, unary modulus, diagonal packet, finite windows, regular-rational readback, dyadic ledger, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RealCauchyCompletionUp.RealCauchyCompletionUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of regular streams, uniqueness of limits, or host real construction.

Upgrade path: No further formal-axis upgrade remains for the checked standard bridge over the same finite regular-family, diagonal, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, Pkg, and NameCert rows.

Theorem 360.63 (RealCauchyCompletion sibling lattice route)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public seal route factors through the already displayed sibling surfaces chapter 126, chapter 301, chapter 291, and chapter 38, with chapter 371 supplying the regular-family input route.*

Equivalently, a public $\mathsf{RealCauchyCompletion}^{\uparrow}$ *read is a finite five-sibling NameCert factorization through the regular-family input, stream window, regular-rational readback, dyadic observation ledger, and exported* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient stream, no selected host limit, no host real equality, no countable-choice row, and no ambient completeness coordinate.*

Proof. Start with an accepted $\mathsf{RealCauchyCompletion}^{\uparrow}$ carrier. The diagonal handoff of Definition 360.3 routes each precision request through the modulus row, the diagonal packet, one finite $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger.

Next apply the stream-regseq-tail seal factorization, Definition 360.9. It keeps the same stream window, regular-rational readback, dyadic ledger, diagonal packet, and $\mathsf{Real}^{\uparrow}$ seal on one completion-facing route, so the sibling surfaces are not optional decorations around the carrier.

The regular-family side enters through chapter 371. The terminal handoff of Definition 360.11 identifies that input as the finite RegularCauchyName ledger feeding the same stream schedule, dyadic-radius row, regular readback, diagonal packet, and exported seal.

Finally use the real seal readback scope of Definition 360.8 and the public real-name export of Definition 360.18. These rows expose the consumer-facing endpoint only as the displayed $\mathsf{Real}^{\uparrow}$ seal over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data, and Definition 360.4 excludes quotient streams, selected host limits, host real equality, countable choice, and ambient completeness from the public route. □

Theorem 360.64 (RealCauchyCompletion four-face support nondegeneracy)**.** *Every accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public completion read that reaches the exported* $\mathsf{Real}^{\uparrow}$ *seal exhibits all four Phase Real and Completion faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*, componentwise* hsame*,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompletion}^{\uparrow}}$ *rows.*

The route is nondegenerate: deleting any one of the four faces, or deleting any one of the displayed support rows, changes the accepted packet read. It does not preserve the same public completion read through $\mathsf{RealCauchyCompletion}^{\uparrow}$*.*

Proof. Start with the accepted public completion read. The diagonal handoff of Definition 360.3 forces each precision request through the diagonal packet, one finite $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the seal can be consumed.

Next apply the stream-regseq-tail seal factorization Definition 360.9. It keeps the stream window, regular readback, dyadic ledger, diagonal packet, and exported $\mathsf{Real}^{\uparrow}$ seal on one completion-facing route. The sibling lattice route Definition 360.63 then identifies these rows with the four named sibling faces of the public NameCert factorization.

Now use the current four-face phase support theorem Definition 360.52. It reads the same route as the Phase Real and Completion exit and requires the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows alongside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply terminal support minimality and defect localization, Definition 360.51 and Definition 360.54. These results say that omitting or replacing any named face or support row is

witnessed at that same local row and leaves the accepted route. Hence such a deletion changes the packet read rather than preserving the same public completion read. □

Theorem 360.65 (RealCauchyCompletion exact-boundary downstream route)**.** *Every* RealCompletionExactBound *consumer of the* RealCauchyCompletion$^{\uparrow}$ *completion route factors through the* RealCauchyCompletion$^{\uparrow}$ *four-face support surface. The consumed route is the finite packet over* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *together with the displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *support rows.*

The downstream consumer exports no quotient stream, no selected host limit, no host real equality, no countable-choice row, and no ambient completeness row. It is a consumer of the four-face completion route, not an isolated completion object.

Proof. Start with an accepted RealCompletionExactBoundaryUp consumer of the completion route. By Definition 1120.6, the exact-boundary packet exposes exactly the four completion-facing faces StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ after its governing seal, classifier, witness-ledger, and budget rows.

Now read the same four faces on the RealCauchyCompletion$^{\uparrow}$ side. The sibling lattice route Definition 360.63 places every public seal read on the finite route through the regular-family input, stream window, regular-rational readback, dyadic observation ledger, and exported Real$^{\uparrow}$ seal.

Apply Definition 360.64. It says that the public completion read is nondegenerate exactly when the four faces and the visible BHist, hsame, Cont, Pkg, and NameCert support rows are present. Therefore the exact-boundary consumer can read the completion route only through that support surface.

Finally use Definition 360.4. Since the completion carrier has no coordinate for quotient streams, selected host limits, host real equality, countable choice, or ambient completeness, the downstream exact-boundary read cannot export any of those rows. It remains a finite consumer of the displayed four-face route. □

Theorem 360.66 (RealCauchyCompletion exact-boundary bidirectional pullback)**.** *Let an accepted* RealCauchyCompletion$^{\uparrow}$ *carrier and an accepted* RealCompletionExactBoundaryUp *packet display the same finite* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger, terminal* Real$^{\uparrow}$ *seal, and the same* BHist, hsame, Cont, Pkg, *and local* NameCert *support rows. Then the* RealCauchyCompletion$^{\uparrow}$ *public completion read factors through the* RealCompletionExactBoundaryUp *terminal consumer, and conversely the exact-boundary terminal consumer restricts back to the* RealCauchyCompletion$^{\uparrow}$ *four-face support surface.*

The pullback exports no quotient stream, no selected host limit, no host real equality, no countable-choice row, no metric-completeness row, no bridge row, and no ambient completeness row. It is exactly the shared finite four-face support route.

Proof. Start with the shared-support hypotheses. The downstream route theorem Definition 360.65 already places every exact-boundary consumption of the RealCauchyCompletion$^{\uparrow}$ completion route on the four-face surface carried by StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, and the displayed structural rows.

For the forward direction, apply Definition 1120.13. Since the exact-boundary packet displays the same four faces and the same BHist, hsame, Cont, Pkg, and local NameCert rows, its terminal consumer is the finite route through those rows. The public RealCauchyCompletion$^{\uparrow}$ read therefore factors through that terminal consumer and not through any separate completion coordinate.

For the reverse direction, apply Definition 360.64. The terminal exact-boundary consumer can agree with the public completion read only while retaining the same four faces and the same support

rows. Thus the exact-boundary terminal consumer restricts back to the $\mathsf{RealCauchyCompletion}^{\uparrow}$ four-face support surface.

It remains to exclude outside witnesses. The exact-boundary side is bounded by Definition 1120.14, while the completion side is bounded by Definition 360.4. Together these remove quotient streams, selected host limits, host real equality, countable choice, metric completeness, bridge rows, and ambient completeness from both directions. Hence the two accepted readings meet precisely in the shared finite support pullback. □

Chapter 361

A Concrete Naming Certificate for $\mathsf{DyadicBisection}^{\uparrow}$

The $\mathsf{DyadicBisection}^{\uparrow}$ interface names the finite bisection ledger used by real-and-completion consumers when an interval is refined by midpoint comparison. In the Bishop–Bridges constructive-analysis reading, bisection is a located finite interval process with explicit precision control; in the Moore interval-analysis reading, it is a finite enclosure refinement ledger. BEDC records the certificate rows themselves, not an analytic host search, a completed real intersection, or a hidden decision procedure.

The bisection carrier consumes the endpoint and dyadic-radius rows supplied by chapter 354, chapter 306, and chapter 291. These rows provide the initial endpoint carrier, the midpoint and dyadic-radius tolerance ledger, and the finite comparison scale needed before the packet can expose nested-window data or hand off a $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read.

Definition 361.1 (DyadicBisection carrier)**.** *A* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, \pi, M, L, J, E, R, S, H, C, P, N)$$

with the following visible coordinates:

1. *I is the initial* $\mathsf{RationalInterval}^{\uparrow}$ *row, including the endpoint order and endpoint readback rows used by downstream interval consumers;*
2. *π is the unary precision row that names the requested finite stopping window;*
3. *M is the finite chain of* $\mathsf{DyadicInterval}^{\uparrow}$ *or* $\mathsf{DyadicMesh}^{\uparrow}$ *midpoint-comparison rows inspected during the run;*
4. *L is the left/right branch ledger, with one displayed branch row for each midpoint comparison in M;*
5. *J is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row obtained by threading the selected subintervals;*
6. *E is the endpoint readback ledger for every retained lower endpoint, upper endpoint, midpoint, and displayed dyadic tolerance;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row extracted from the retained endpoint windows;*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *handoff surface through which real consumers read the same finite windows;*

9. *H stores the* hsame *transports for the initial interval, precision row, midpoint rows, branch ledger, nested interval row, endpoint readbacks, and handoff rows;*
10. *C stores the* Cont *routes for refinement, endpoint inspection, regular-sequence readback, and real-name consumption;*
11. *P is the* Pkg *provenance row naming* $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *registration row.*

The carrier exports only these finite rows. It has no coordinate for a host real root, a completed interval intersection, trichotomy on reals, Dedekind-cut data, or an ambient completeness proof.

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionTasteGate single carrier alignment`

Definition 361.2 (DyadicBisection classifier)**.** *Two accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carriers classify together when, after the displayed* hsame *and* Cont *transports, their initial interval rows, precision schedules, midpoint chains, branch ledgers, nested dyadic interval chains, endpoint readbacks,* $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{Real}^{\uparrow}$ *handoff rows refine to the same finite observation window under the displayed dyadic tolerance.*

Concretely, if the common stopping request is n, then each retained endpoint window is compared only after both packets are transported to the same finite tolerance row

$$\varepsilon_n = 2^{-n}.$$

Agreement means row-by-row equality of the transported branch choices and endpoint readbacks at that tolerance, followed by the inherited $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *classifiers. No comparison is made by appealing to a host real value or to an infinite intersection.*

Theorem 361.3 (DyadicBisection NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *surface has five obligations:*

1. *carrier: formation of the finite* BHist *packet of Definition 361.1;*
2. *classifier: the finite-window classifier of Definition 361.2;*
3. *stability: componentwise preservation of the interval, precision, branch, nested-interval, endpoint, and handoff rows under displayed* hsame *transport and* Cont *refinement routes;*
4. *ledger exactness: every left/right branch choice is exactly the branch row attached to the corresponding midpoint comparison, and every retained endpoint read is one of the displayed dyadic ledger rows;*
5. *handoff: the retained endpoint windows are repacked into* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows without adding a host real intersection or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier namecert obligations`

Proof. Carrier formation is the assembly of the displayed finite packet. Each coordinate is either a history row, a transport row, a continuation route, package provenance, or a local naming row owned by the listed dependency certificates.

The classifier is finite because the stopping request fixes the displayed tolerance and the inspected midpoint chain. After transporting both carriers to that window, comparison is componentwise on the initial interval, precision schedule, branch ledger, nested interval row, endpoint readbacks, and handoff rows.

Stability follows the same component order. Transport first moves the initial interval and precision rows, then the midpoint rows and branch choices, then the nested interval and endpoint ledgers, and finally the regular-sequence, stream, and real-name handoff rows.

Ledger exactness is the branch-row invariant: the kth branch coordinate is the only source of the kth left/right choice, and the retained endpoint rows are precisely the rows routed by C and transported by H.

The handoff obligation repacks the finite retained windows into $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces. Since the carrier contains no host real, infinite intersection, or completeness coordinate, no such object can be exported. □

Theorem 361.4 (DyadicBisection nested interval bridge)**.** *For every accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier B, the displayed branch ledger determines a finite* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row whose kth interval is the subinterval selected by the kth midpoint comparison. The bridge exposes the initial* $\mathsf{RationalInterval}^{\uparrow}$ *row, the dyadic midpoint ledger, the left/right branch ledger, the endpoint readbacks, and the* $\mathsf{NestedInterval}^{\uparrow}$ *row, and it exports no completed intersection of the nested intervals.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier nested interval bridge`

Proof. Project the carrier of Definition 361.1. The midpoint chain M and branch ledger L have the same finite length, so the kth midpoint comparison has a unique displayed branch row.

Thread the interval row forward along these branch choices. At each step, the retained lower endpoint, upper endpoint, midpoint, and tolerance rows are already present in the endpoint ledger E, and the transport rows in H record that the selected subinterval is the continuation of the previous interval.

Pack the resulting finite chain into the $\mathsf{NestedInterval}^{\uparrow}$ row J. This row is the bridge available to downstream consumers: it records the nested finite refinement path but does not claim that the infinite intersection exists as a host real object. □

Theorem 361.5 (DyadicBisection real stream handoff)**.** *For every accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier and every unary precision request n, the consumer-visible* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *reads factor through the retained dyadic endpoint window at tolerance 2^{-n}, the branch ledger prefix needed to reach that window, the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}$ *row.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier real stream endpoint scope`

Proof. Start with the precision row π and the displayed request n. The carrier uses that row to select a finite branch-prefix length and the corresponding retained endpoint window in E.

Use Definition 361.4 to route the selected endpoint window through the nested interval row. The branch prefix identifies the retained subinterval, while the endpoint ledger supplies the rational lower and upper reads at the requested tolerance.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R consumes those retained rational windows. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows in S then expose the same finite observation to real consumers through the displayed transports, continuation routes, package provenance, and local certificate row.

Thus every real-facing read has the stated finite factorization. The handoff does not use root existence, real trichotomy, Dedekind cuts, host real equality, or a host intersection theorem. □

Theorem 361.6 (DyadicBisection nested window handoff)**.** *For every accepted* DyadicBisection$^{\uparrow}$ *carrier and every unary precision request n, the* RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, *and* Real$^{\uparrow}$ *reads are determined by the unique branch-prefix ledger segment that reaches the retained nested interval window at tolerance 2^{-n}, together with the endpoint readbacks and displayed* hsame *and* Cont *transports. This handoff introduces no host intersection, completed real root, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier nested window handoff`

Proof. Project the branch ledger of Definition 361.1. The precision request n selects the finite prefix length whose branch choices reach the retained endpoint window recorded in E.

Apply Definition 361.4. The selected branch prefix determines the corresponding NestedInterval$^{\uparrow}$ row, and that row exposes the retained lower endpoint, upper endpoint, midpoint, and tolerance readbacks from the finite ledger.

Now apply Definition 361.5. The RegSeqRat$^{\uparrow}$ row reads those retained rational windows, while the StreamName$^{\uparrow}$ and Real$^{\uparrow}$ rows expose the same finite observation through the displayed transports and continuation routes.

The carrier has no field for an intersection of all retained intervals and no field for a completeness theorem. Therefore the nested window is the exact surface consumed by the handoff. □

Theorem 361.7 (DyadicBisection branch endpoint determinacy)**.** *For every accepted* DyadicBisection$^{\uparrow}$ *carrier and every unary precision request n, the finite branch-prefix ledger segment that reaches the retained window at tolerance 2^{-n} uniquely determines the retained lower endpoint, retained upper endpoint, midpoint, tolerance row, nested interval row, and the downstream* RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, *and* Real$^{\uparrow}$ *handoff rows. No host root, infinite intersection, trichotomy, or completeness row participates in this determination.*

Proof. Project the accepted carrier of Definition 361.1. The unary request n selects a finite prefix of the branch ledger, and each row of that prefix chooses one retained dyadic subinterval from the previous endpoint window.

Apply the nested interval bridge of Definition 361.4. The selected branch prefix fixes the retained lower endpoint, upper endpoint, midpoint, and tolerance readback in the endpoint ledger, then packs the same finite chain into the NestedInterval$^{\uparrow}$ row.

Next use the real stream handoff of Definition 361.5 and the nested window handoff of Definition 361.6. The RegSeqRat$^{\uparrow}$ row consumes the retained endpoint windows, and the StreamName$^{\uparrow}$ and Real$^{\uparrow}$ rows expose that same finite observation through the displayed hsame, Cont, Pkg, and NameCert rows.

The construction has used only the finite carrier rows, the nested interval bridge, and the handoff routes. Since none of these rows contains a host root, an infinite intersection, a trichotomy decision, or a completeness principle, the branch-prefix ledger segment is the determinant of the retained endpoint window and its downstream reads. □

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier branch endpoint transport`

Theorem 361.8 (DyadicBisection branch nested-window scope)**.** *For every accepted* DyadicBisection$^{\uparrow}$ *carrier over* DyadicInterval$^{\uparrow}$, DyadicMesh$^{\uparrow}$, NestedInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, *and* Real$^{\uparrow}$, *the selected branch ledger determines the finite nested-window row and transports it through the displayed* hsame *and* Cont *rows to the same* Real$^{\uparrow}$ *seal endpoint. The scope is the carried packet*

$$B = (I, \pi, M, L, J, E, R, S, H, C, P, N),$$

where the finite branch prefix in L determines the retained endpoint rows in E, the nested interval row J, the regular readback R, and the real-stream handoff surface S.

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier branch nested window scope`

Proof. Start with the branch ledger and its endpoint determination. Apply Definition 361.7: for a unary precision request, the finite branch prefix fixes the retained endpoint window, tolerance row, nested interval row, and downstream handoff rows.

Next identify the nested-window row. Apply Definition 361.4 to thread the branch choices through the dyadic midpoint chain and pack the retained subintervals into J.

Now use Definition 361.6. The same branch prefix and endpoint readbacks determine the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads through the displayed transport and continuation rows.

Finally apply Definition 361.5 to bind the resulting regular and stream reads to the real-facing handoff surface S. The construction never leaves the displayed carrier rows, so it exports no host root, infinite intersection, trichotomy decision, or ambient completeness row. □

Theorem 361.9 (DyadicBisection selector interval stability)**.** *For an accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier*

$$B = (I, \pi, M, L, J, E, R, S, H, C, P, N),$$

transporting the initial interval I, precision row π, midpoint chain M, and branch ledger L through the displayed hsame *ledger H and* Cont *routes C preserves the selected retained interval row. It also preserves the retained endpoint readbacks in E, the* $\mathsf{NestedInterval}^{\uparrow}$ *row J, and the downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff rows R, S.*

The stability statement is entirely internal to the displayed bisection carrier, package provenance P, and local $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *row N; it does not rely on a hidden root, trichotomy, or a completed interval intersection.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier retained interval nonempty iff`

Proof. First apply Definition 361.4. The transported branch ledger threads the transported midpoint chain into the same finite $\mathsf{NestedInterval}^{\uparrow}$ row, so the selected retained interval row remains a displayed row of the carrier.

Next use Definition 361.7. Once the retained interval row is fixed at the requested precision, the lower endpoint, upper endpoint, midpoint, tolerance row, and downstream handoff rows are determined by the same finite branch-prefix ledger.

Apply Definition 361.6 to identify the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads with that retained nested window and its endpoint readbacks. The transports in H and routes in C move these rows componentwise.

Finally use Definition 361.8. The selected branch ledger, nested row, endpoint ledger, and handoff rows remain inside the carried packet B, so no host root, real trichotomy, or completed intersection object participates in the stability claim. □

Theorem 361.10 (DyadicBisection public branch totality)**.** *For every accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier*

$$B = (I, \pi, M, L, J, E, R, S, H, C, P, N),$$

and every unary precision request n, the public finite branch-consumer route is exhausted by the branch endpoint transport, the branch nested-window scope, the real stream endpoint scope, and the retained interval nonempty read. The route exposes exactly the initial interval row, precision row, finite midpoint chain, selected branch-prefix ledger, retained endpoint readbacks, $\mathsf{NestedInterval}^{\uparrow}$

row, $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{Real}^{\uparrow}$ *handoff surface,* hsame *transports,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *row.*

It exports no host root, infinite intersection, trichotomy decision, Dedekind cut, hidden midpoint oracle, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier public branch totality`

Proof. Begin with the selected branch-prefix ledger for the request n. Apply Definition 361.7 to determine the retained lower endpoint, upper endpoint, midpoint, tolerance row, nested interval row, and downstream handoff rows.

Next apply Definition 361.8. This places the branch prefix, endpoint rows, nested row, regular readback, and real-stream handoff inside the displayed carrier packet and transports them through H and C.

Use Definition 361.5 to expose the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer reads at the same finite retained endpoint window.

Finally apply the retained interval stability theorem of this chapter. The retained interval row and its endpoint readbacks remain componentwise stable under the displayed transports and continuation routes. Hence the public route is exactly the finite packet listed above, with no off-table analytic object. □

Theorem 361.11 (DyadicBisection retained interval consumer route)**.** *Two accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carriers with the same transported initial interval, precision row, midpoint chain, branch ledger, retained endpoint readbacks, nested interval row,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{Real}^{\uparrow}$ *handoff surface,* hsame *transports,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *row have the same public retained-interval consumer route. Conversely, equality of the public retained-interval route is witnessed by componentwise agreement of those displayed finite rows.*

The route is the public object surface for dyadic bisection consumers. It does not assert a completed intersection, a host real root, real trichotomy, or a Dedekind-cut carrier.

Proof. For the forward direction, assume componentwise agreement of the displayed finite rows. Apply Definition 361.10 to both carriers. Each public branch-consumer route is exhausted by the same finite branch prefix, endpoint ledger, nested interval row, regular readback, stream window, real handoff, transport, continuation, package, and certificate rows.

The retained interval stability theorem identifies the retained interval row and endpoint readbacks after transport. The branch nested-window scope and real stream handoff then route the same retained window to the same public consumer surface.

For the reverse direction, suppose the public retained-interval consumer routes are equal. Project the route through Definition 361.10. The projection exposes only the listed finite carrier rows, so route equality supplies componentwise agreement of the interval, precision, midpoint, branch, endpoint, nested, regular, stream, real, transport, continuation, package, and certificate rows.

No further coordinate is available in the carrier. Thus equality of the public route is exactly componentwise agreement of the displayed finite rows, and the off-table host and completeness objects cannot participate. □

Lean4 ✓ `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier retained interval consumer route`

Theorem 361.12 (DyadicBisection StreamName retained route)**.** *Any retained-interval consumer of an accepted* $\mathsf{DyadicBisection}^{\uparrow}$ *carrier that hands off to* $\mathsf{StreamName}^{\uparrow}$ *must read the rational interval row, dyadic interval row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, retained branch endpoint row, nested window row, displayed* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$ *row before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read. The retained StreamName route exposes the finite bisection packet, not a host root, completed intersection, real trichotomy, or Dedekind-cut carrier.*

Proof. First use the nested interval bridge. It turns the midpoint chain and retained branch choices into the finite nested interval row and retained endpoint readbacks.

Next apply the real stream handoff and nested-window handoff. They route the retained endpoint data through the $\mathsf{StreamName}^{\uparrow}$ window and then toward the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads without leaving the displayed transport and continuation rows.

Finally apply the retained interval consumer route. It identifies the public retained route with the rational interval, dyadic interval, radius, retained endpoint, nested window, transport, continuation, package, and certificate rows, so no completion-level or host-level object is exposed before the finite StreamName route has been read. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicBisection}^{\uparrow}$ packet is generated from an initial $\mathsf{RationalInterval}^{\uparrow}$ row, a unary precision row, finite $\mathsf{DyadicInterval}^{\uparrow}$ or $\mathsf{DyadicMesh}^{\uparrow}$ midpoint comparisons, an exact left/right branch ledger, a $\mathsf{NestedInterval}^{\uparrow}$ refinement row, endpoint readbacks, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff rows, displayed hsame and Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public packet binds the carrier, finite-window classifier, NameCert obligation surface, nested interval bridge, real-stream handoff, branch endpoint determinacy, branch nested-window scope, selector interval stability, public branch totality, and retained-interval consumer route for dyadic bisection over $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicBisection}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicBisectionUp.DyadicBisectionCarrier\_retained\_interval\_nonempty\_iff`

Bridge status: none

Origin: human

Not claimed: This chapter does not close completeness, root existence, trichotomy, Dedekind cuts, uniqueness of a limit point, or host real intersection.

Upgrade path: No further paper-theory upgrade remains; Lean bridge checking remains a separate verification-axis task.

Chapter 362

A Concrete Naming Certificate for $\mathsf{StationaryRationalDiagonal}^{\uparrow}$

The $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ interface names the finite packet obtained from a stationary rational regular Cauchy input and its diagonal representative. It sits at the Real and Completion boundary: the source is a $\mathsf{Rat}^{\uparrow}$ row read as a constant $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ object, while the consumer sees only the associated $\mathsf{Real}^{\uparrow}$ seal surface. The packet carries no quotient of regular streams, host real equality, ambient completeness principle, or hidden limit selection.

The packet consumes substantive source rows from the rational seed surface of chapter 37, the constant-window carrier of chapter 126, the regular rational readback of chapter 301, and the seal boundary of chapter 38; these are the displayed inputs to the diagonal packet, not decorative references.

The stationary diagonal is the degenerate tail-selection case recorded by chapter 458: every requested selector window is constant at the displayed $\mathsf{Rat}^{\uparrow}$ seed, and the $\mathsf{Real}^{\uparrow}$ seal handoff therefore consumes the same selected-tail discipline through a singleton constant window.

Definition 362.1 (Stationary rational diagonal carrier)**.** *A* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ *carrier is a finite* BHist *row*

$$S = (q, C, R, D, E, H, K, P, N)$$

with the following visible coordinates:

1. *q is the* $\mathsf{Rat}^{\uparrow}$ *source row;*
2. *C is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule generated by q;*
3. *R is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose inspected windows all return q;*
4. *D is the diagonal selector row that always selects the stationary member and the requested regular-sequence window;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal ledger for the constant rational embedding;*
6. *H records the displayed* hsame *transports for the rational source, stream schedule, regular readback, diagonal selector, and real seal;*
7. *K records the visible* Cont *routes by which consumers reach the selected constant windows;*
8. *P is the* Pkg *provenance naming the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *dependency rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryRationalDiagonal}^{\uparrow}}$ *provenance row.*

The classifier compares accepted carriers only by the displayed $\mathsf{Rat}^{\uparrow}$ *seed classifier, the constant* $\mathsf{StreamName}^{\uparrow}$ *readback, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *window row, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing route, transported through the listed* hsame *and* Cont *rows.*

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalTasteGate single carrier alignment`

Theorem 362.2 (Stationary rational diagonal NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryRationalDiag}}$ *surface consists exactly of five rows:*

1. *carrier: the finite packet of Definition 362.1;*
2. *classifier: comparison of two accepted packets through their displayed* $\mathsf{Rat}^{\uparrow}$ *seed classifier and the induced constant* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows;*
3. *stability: componentwise* hsame *transport and visible* Cont *routing preserve the stationary seed, every constant window, the diagonal selector, and the real seal ledger;*
4. *exactness/readback: every consumer read factors through the same stationary rational seed and the same constant regular-rational window before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger: the* Pkg *and* NameCert *coordinates expose only* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows.*

No row of the surface is a quotient, a choice principle, or a host real equality.

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier namecert obligation surface`

Proof. Project the carrier of Definition 362.1. The listed coordinates already name the rational seed, constant stream, stationary regular readback, diagonal selector, real seal, transports, continuation routes, package provenance, and local ledger.

For the classifier row, compare two packets only by the seed classifier and its constant readbacks. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ coordinates are stationary, so every selected window reads the same rational row before the $\mathsf{Real}^{\uparrow}$ seal is consulted.

Stability follows by transporting each visible coordinate through its stored hsame row and using the displayed Cont routes to reach the same selected window after transport. The exactness row is then forced, because the real seal has no source except the stationary rational readback.

The ledger is exhausted by P and N. Since the carrier has no coordinate for quotient streams, host real equality, or a hidden selected limit, none of those objects can be exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier semantic name certificate`

Theorem 362.3 (Stationary rational diagonal Real handoff)**.** *Every accepted* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ *packet determines a concrete* $\mathsf{Real}^{\uparrow}$ *consumer surface for the stationary rational diagonal representative. The consumer route reads the* $\mathsf{Rat}^{\uparrow}$ *seed through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the diagonal selector, and the displayed* $\mathsf{Real}^{\uparrow}$ *seal ledger, with all replacements accounted for by visible* hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Start with the accepted carrier and read its constant stream coordinate. Every requested observation reaches the same $\mathsf{Rat}^{\uparrow}$ seed through the displayed Cont route.

The stationary $\mathsf{RegSeqRat}^{\uparrow}$ coordinate repackages those constant observations as the regular-rational readback used by the diagonal selector. Since the selector always chooses the stationary member, the selected window and the regular readback stay on the listed finite rows.

Finally read the $\mathsf{Real}^{\uparrow}$ seal ledger. It is attached to the same stationary readback and is transported only through the displayed hsame rows. Repacking these coordinates gives the real-facing handoff, and the carrier contains no additional endpoint object. □

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier real handoff`

Theorem 362.4 (Stationary rational diagonal consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ *packet factors through the finite rows listed in Definition 362.1. In particular, the consumer may use the* $\mathsf{Rat}^{\uparrow}$ *seed, constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, diagonal selector,* $\mathsf{Real}^{\uparrow}$ *seal ledger, visible* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and nothing outside that finite packet.*

Proof. Use the obligation surface of Definition 362.2 to place the carrier, classifier, stability, readback, and ledger rows on one local certificate surface.

Next apply the handoff of Definition 362.3. It routes every real-facing read through the stationary seed, the constant stream schedule, the regular-rational readback, the diagonal selector, and the real seal ledger.

It remains to delimit completion-facing consumers. Such a consumer reaches the packet only through the same $\mathsf{Real}^{\uparrow}$ seal-facing route and the same displayed provenance rows. Therefore the consumer boundary is exactly the finite packet surface stated in the theorem, with no coordinate for quotient equality, choice, or host real equality. □

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier consumer boundary`

Theorem 362.5 (Stationary rational diagonal scoped boundary)**.** *The carrier definition, NameCert obligation surface, Real handoff, and consumer boundary for* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ *form a single scoped dependency packet over exactly the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows displayed in Definition 362.1. Every accepted consumer read is routed through that finite packet, and the packet exports no quotient stream, host real equality, or selected limit.*

Proof. The carrier of Definition 362.1 lists the only coordinates available to the local certificate: the rational seed, constant stream schedule, stationary regular-rational readback, diagonal selector, real seal ledger, visible transports, visible continuation routes, package provenance, and local NameCert row.

The obligation surface of Definition 362.2 makes those coordinates into the full certificate interface. Its carrier, classifier, stability, exactness, and ledger rows mention precisely the same dependency names, so no consumer can acquire an additional source row through the certificate laws.

The handoff theorem Definition 362.3 routes every real-facing read through the constant stream, the stationary regular readback, the diagonal selector, and the real seal. The consumer boundary theorem Definition 362.4 then applies the same routing to completion-facing consumers.

Thus every permitted read has already passed through the displayed finite packet. Since none of the cited rows contains a coordinate for quotient streams, host real equality, or a selected limit, none of those objects is exported by the scoped boundary. □

Theorem 362.6 (Stationary rational diagonal classifier determinacy)**.** *Let D and D' be accepted* StationaryRationalDiagonal$^{\uparrow}$ *packets. If their displayed* Rat$^{\uparrow}$ *seed rows classify together on the local carrier, then their induced constant* StreamName$^{\uparrow}$ *schedules, stationary* RegSeqRat$^{\uparrow}$ *windows, diagonal selector rows, and* Real$^{\uparrow}$ *seal routes classify together through the stored* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Conversely, any accepted classifier comparison on those downstream rows restricts to the displayed rational seed classifier, since the rational seed coordinate is the only carrier coordinate that supplies rational content to the stationary diagonal read.*

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier classifier determinacy`

Proof. Project the carriers of D and D' along Definition 362.1. The only rational source row in each packet is the displayed Rat$^{\uparrow}$ seed; the remaining rows are the constant stream schedule, the stationary regular-rational readback, the diagonal selector, the real seal ledger, and the visible transport and package rows.

Stationarity reduces every selected regular-rational window to that same rational row. Therefore a seed classifier comparison transports directly to the constant StreamName$^{\uparrow}$ schedule and to each stationary RegSeqRat$^{\uparrow}$ window read through the stored hsame rows.

Apply the real handoff of Definition 362.3 and the scoped boundary of Definition 362.5. The diagonal selector and Real$^{\uparrow}$ seal route do not add a second rational source; they only route the stationary read through the displayed Cont, Pkg, and NameCert rows.

For the converse, restrict any accepted downstream classifier comparison back along the handoff route. The boundary theorem leaves exactly one carrier coordinate that can supply rational content, namely the displayed seed row, so the downstream classifier comparison determines the same local Rat$^{\uparrow}$ seed classifier. □

Theorem 362.7 (Stationary rational diagonal constant-window totality)**.** *Every unary request made against an accepted* StationaryRationalDiagonal$^{\uparrow}$ *packet reads the same displayed* Rat$^{\uparrow}$ *row. The read first follows the constant* StreamName$^{\uparrow}$ *schedule generated by the seed, then the stationary* RegSeqRat$^{\uparrow}$ *window row, and then the diagonal selector that returns that stationary window for the requested precision.*

Thus every consumer-visible rational observation is already the displayed seed row, transported only through the listed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier constant window totality`

Proof. Project the accepted carrier along Definition 362.1. Its only rational coordinate is the seed q, and its stream coordinate C is explicitly the constant schedule generated by q.

Read a unary request through the continuation route K. The schedule therefore returns the same rational row for the requested place, and the stationary RegSeqRat$^{\uparrow}$ coordinate R records that every displayed regular window has that same readback.

Finally apply the diagonal selector coordinate D. Since D always selects the stationary member and the requested regular-sequence window, no second rational row can be introduced. The transport and provenance coordinates merely move the same displayed rows, so the readback is total on the stated finite surface. □

Theorem 362.8 (Stationary rational diagonal seal factorization)**.** *For every accepted* StationaryRationalDiagonal$^{\uparrow}$ *packet, the* Real$^{\uparrow}$ *seal ledger factors only through the displayed* Rat$^{\uparrow}$ *seed, constant* StreamName$^{\uparrow}$ *schedule, stationary* RegSeqRat$^{\uparrow}$ *readback, diagonal selector, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

In particular, the seal cannot factor through quotient regular streams, a host real equality, an ambient completeness principle, or a selected limit outside the carrier.

Proof. Start from the real handoff theorem Definition 362.3. It identifies the seal-facing route as the route from the rational seed through the constant stream, stationary regular readback, diagonal selector, and real seal ledger.

Use the constant-window totality theorem Definition 362.7 to collapse each unary read on that route to the same displayed Rat$^{\uparrow}$ seed row. The Real$^{\uparrow}$ seal is therefore reached after a finite readback path whose source is the carrier coordinate q.

The remaining carrier coordinates are H, K, P, and N. These record transport, continuation, package provenance, and local certificate provenance; they do not contain quotient regular streams, host real equality, ambient completeness, or selected limits. Hence the stated factorization exhausts the seal ledger. □

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier seal factorization`

Theorem 362.9 (Stationary rational diagonal obligation-closure package)**.** *The* StationaryRationalDiagonal$^{\uparrow}$ *carrier, NameCert surface, Real handoff, consumer boundary, scoped boundary, classifier determinacy, constant-window totality, and seal factorization assemble an obligation-closed paper package. Its closed scope is exactly the displayed* Rat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *dependency surface.*

Proof. The carrier definition supplies the finite packet and its dependency names. The NameCert obligation surface records the carrier, classifier, stability, readback, and ledger rows on that packet.

The Real handoff and consumer boundary route all real-facing and completion-facing reads through the same finite coordinates. The scoped boundary then states that no additional quotient, host equality, or selected limit is available to consumers.

Classifier determinacy identifies the accepted comparison with the displayed rational seed classifier and its transported downstream rows. Constant-window totality and seal factorization complete the package by showing that every unary read and every real-seal read is exhausted by the same stationary rational route. These rows are precisely the named obligation surface, so the paper-side closure grade is obligation closure. □

Lean4 ✓ `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier obligation closure package`

Closure status for NameCert$_{\text{StationaryRationalDiagonal}^{\uparrow}}$.

Constructive story: A StationaryRationalDiagonal$^{\uparrow}$ packet is generated from one Rat$^{\uparrow}$ source row, its constant StreamName$^{\uparrow}$ schedule, a stationary RegSeqRat$^{\uparrow}$ readback, a diagonal selector, a Real$^{\uparrow}$ seal ledger, visible hsame and Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{StationaryRationalDiagonal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 362.4 binds the finite stationary rational diagonal carrier, NameCert surface, Real handoff, and consumer boundary over Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{StationaryRationalDiagonal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.StationaryRationalDiagonalUp.StationaryRationalDiagonalCarrier\_seal\_factorization`

Bridge status: none

Origin: human

Not claimed: This chapter does not close general real completeness, arbitrary regular Cauchy sequences, quotient equality of regular streams, uniqueness of limits, host real construction, or bridge checking.

Upgrade path: Public closure requires a public-object boundary for stationary rational diagonal consumers beyond the scoped seal factorization.

Chapter 363

A Concrete Naming Certificate for $\mathsf{SignedDigitReal}^{\uparrow}$

The $\mathsf{SignedDigitReal}^{\uparrow}$ interface names the real-facing certificate obtained when finite signed-digit stream windows are normalized into dyadic enclosures and routed to the $\mathsf{Real}^{\uparrow}$ seal. It sits above $\mathsf{SignedDigitStream}^{\uparrow}$ (chapter 316), DyadicRatCore and $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$ (chapter 335), $\mathsf{LocatedCauchy}^{\uparrow}$ (chapter 340), and $\mathsf{Real}^{\uparrow}$ (chapter 38). The packet is a finite-window name certificate, not a quotient of streams, a selected representative, a trichotomy principle, or a choice construction.

Definition 363.1 (SignedDigitReal carrier)**.** *A* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, E, L, H, C, P, N)$$

with the following visible coordinates:

1. *S is an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *row, carrying signed digits, unary observation schedules, normalization ledgers, and normalized dyadic endpoint rows;*
2. *W is the family of finite prefix windows read from the* $\mathsf{StreamName}^{\uparrow}$ *schedule of S;*
3. *E assigns to every displayed window in W a* DyadicRatCore *enclosure row, read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers exposed by the signed-digit normalization;*
4. *L is the* $\mathsf{LocatedCauchy}^{\uparrow}$ *located-window row stating that the displayed enclosures form a located Cauchy surface at the requested precisions;*
5. *H records componentwise* hsame *transport for signed digits, windows, enclosures, located rows, and real-seal boundary rows;*
6. *C records the* Cont *routes by which a consumer reads each finite prefix, its normalization ledger, its dyadic enclosure, its located row, and its real-seal boundary;*
7. *P is the* Pkg *provenance naming* $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{LocatedCauchy}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as dependency anchors;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *provenance row.*

The carrier predicate accepts exactly those packets whose every finite signed-digit prefix decodes to the displayed DyadicRatCore *enclosure and whose normalization windows are compatible with the* $\mathsf{Real}^{\uparrow}$ *seal boundary through the located-Cauchy row L.*

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealTasteGate single carrier alignment`

Definition 363.2 (SignedDigitReal classifier)**.** *For accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carriers* R *and* R'*, the classifier* $\mathsf{SignedDigitRealClassifier}(R, R')$ *holds when the underlying* $\mathsf{SignedDigitStream}^{\uparrow}$ *rows agree on every shared finite window, the decoded* $\mathsf{DyadicRatCore}$ *enclosures agree through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *classifier, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *enclosure routes transport componentwise, the located-Cauchy rows agree at each requested precision, and the induced* $\mathsf{Real}^{\uparrow}$ *seal boundaries are compared by the classifier of Definition 38.9.*

The classifier is finite-window data. It has no coordinate for quotient stream equality, selected representatives, trichotomy, choice, or host real equality.

Theorem 363.3 (SignedDigitReal finite normalization window)**.** *Let* $R = (S, W, E, L, H, C, P, N)$ *be an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier, and let* w *be a displayed finite prefix window in* W*. Finite normalization of the signed-digit prefix of* S *at* w *preserves the* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier surface and produces the corresponding* $\mathsf{DyadicRatCore}$ *enclosure row in* E*.*

The produced row is read entirely through the displayed BHist *prefix, the* hsame *transports in* H*, the* Cont *route in* C*, the* Pkg *provenance in* P*, and the local* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *row* N*. The normalization step creates no quotient tail, selected stream representative, trichotomy witness, choice datum, or host real endpoint.*

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket normalization window transport`

Proof. Project the carrier of Definition 363.1. The window w selects a finite signed-digit prefix from S, its signed-digit normalization ledger, the target dyadic enclosure row in E, and the corresponding located-Cauchy and real-seal boundary rows.

Apply the signed-digit finite-window transport and common-window determinacy from Definition 316.4 and Definition 316.8. These rows say that the normalized prefix is stable under the displayed hsame and Cont movement and that its endpoint read is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint for that window.

Read the endpoint as the enclosure row named by E. The $\mathsf{DyadicApproximation}^{\uparrow}$ route records the enclosure shape, while the $\mathsf{LocatedCauchy}^{\uparrow}$ row records that the same finite observation is acceptable at the requested precision before the $\mathsf{Real}^{\uparrow}$ seal reads it.

Repack the coordinates into the original carrier tuple. Every row used in the repacking is already in BHist, H, C, P, or N, so the operation preserves the carrier surface and cannot expose any of the excluded non-finite data. □

Theorem 363.4 (SignedDigitReal dyadic seal compatibility)**.** *Let* R *and* R' *be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carriers. Suppose their underlying* $\mathsf{SignedDigitStream}^{\uparrow}$ *rows agree on every finite normalization window under the signed-digit classifier, and suppose the decoded* $\mathsf{DyadicRatCore}$ *enclosure rows are transported by the displayed* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *routes. Then the two carriers determine the same* $\mathsf{Real}^{\uparrow}$ *seal boundary in the sense of the* $\mathsf{Real}^{\uparrow}$ *classifier of Definition 38.9.*

The compatibility is obtained by finite-window readback only. It uses no quotient of streams, no selected representative, no trichotomy branch, and no choice principle.

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket dyadic seal compatibility`

Proof. Use Definition 363.2 to reduce the comparison to shared finite normalization windows. On each such window, the two signed-digit prefixes are related by the $\mathsf{SignedDigitStream}^{\uparrow}$ classifier.

Apply Definition 363.3 to both carriers on the same window. The two normalization reads produce dyadic enclosure rows through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ routes.

The hypothesis transports these enclosure rows through the $\mathsf{DyadicApproximation}^{\uparrow}$ and $\mathsf{LocatedCauchy}^{\uparrow}$ routes. Thus every requested precision reaches the same located dyadic enclosure surface before the real seal is read.

Finally invoke the $\mathsf{Real}^{\uparrow}$ classifier specification of Definition 38.9. Since the real boundary sees only the located Cauchy enclosure rows just identified, the two carriers have the same seal boundary and no additional stream quotient or choice-based representative is introduced. □

Definition 363.5 (SignedDigitReal obligation ledger package)**.** *For an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier*

$$R = (S, W, E, L, H, C, P, N),$$

the obligation ledger package is the finite packet that reads these coordinates with the following roles:

1. *S is the signed-digit stream row, including digit data, unary window schedules, normalization ledgers, and endpoint rows;*
2. *W is the displayed family of finite windows;*
3. *E is the decoded* $\mathsf{DyadicRatCore}$ *enclosure row for each window, exposed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{DyadicApproximation}^{\uparrow}$*;*
4. *L is the* $\mathsf{LocatedCauchy}^{\uparrow}$ *located-row route binding the decoded enclosures to requested precisions;*
5. *the* $\mathsf{Real}^{\uparrow}$ *seal boundary is the public consumption row reached from E and L;*
6. *H records componentwise* hsame *rows for streams, windows, enclosures, located rows, and seal rows;*
7. *C records the* Cont *routes through finite prefixes, decoded enclosures, located rows, and the real seal;*
8. *P and N record the dependency package and local* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *provenance.*

The package has no coordinate for quotient streams, selected representatives, trichotomy, choice, or host real equality.

Theorem 363.6 (SignedDigitReal window classifier stability)**.** *Let $R = (S, W, E, L, H, C, P, N)$ and $R' = (S', W', E', L', H', C', P', N')$ be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carriers. If the signed-digit stream rows are related by the classifier on a displayed finite window, then the decoded dyadic enclosures, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *routes, the* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows, and the induced* $\mathsf{Real}^{\uparrow}$ *seal boundary transport through the same obligation ledger package of Definition 363.5.*

In particular, classifier-related windows do not create a second ledger for the enclosure or located-Cauchy read; the finite window, decoded enclosure, located row, and seal boundary are the displayed rows of the package.

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket window classifier stability`

Proof. Apply the finite normalization window theorem Definition 363.3 to R on the chosen displayed window. This projects the signed-digit prefix, its normalization ledger, the decoded dyadic enclosure row, and the corresponding located and seal rows from the obligation ledger package.

Apply the same theorem to R' on the classifier-related window. The stream classifier and common-window transport identify the finite prefixes and their normalization ledgers through the recorded componentwise hsame rows.

Transport the decoded enclosures through the $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{DyadicApproximation}^{\uparrow}$ rows, and then transport the precision witnesses through the $\mathsf{LocatedCauchy}^{\uparrow}$ rows. The continuation routes in C and C' preserve the order from prefix read to enclosure read to located read.

Finally, the $\mathsf{Real}^{\uparrow}$ seal boundary is reached only from the transported enclosure and located rows. Thus the classifier stability is witnessed inside the same finite ledger package, with no extra representative or quotient stream. □

Theorem 363.7 (SignedDigitReal seal ledger exactness)**.** *For every accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal boundary consumes exactly the decoded* DyadicRatCore *enclosure rows and* $\mathsf{LocatedCauchy}^{\uparrow}$ *ledger exposed by Definition 363.5. The seal read is routed through the signed-digit finite window, the normalization ledger, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *enclosure route, the located-Cauchy row, and the recorded* hsame, Cont, Pkg, *and* NameCert *provenance.*

The seal boundary exports no quotient stream, no selected signed-digit representative, no trichotomy branch, no choice datum, and no host real equality.

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket seal ledger exactness`

Proof. Project the obligation ledger package. Each seal read begins with a finite window in W and the signed-digit prefix of S displayed on that window.

Use the finite normalization window theorem to decode that prefix into the enclosure row named by E. The $\mathsf{DyadicApproximation}^{\uparrow}$ route records the enclosure shape and the $\mathsf{LocatedCauchy}^{\uparrow}$ row L records the precision-facing located evidence.

The dyadic seal compatibility theorem Definition 363.4 then reads the real boundary from exactly those enclosure and located rows. Componentwise hsame and Cont transport can move the rows between equivalent carrier presentations, but it does not add any analytic coordinate.

Since P and N list only the named dependency package and local certificate row, every exported seal fact is already accounted for by the ledger. The excluded quotient, representative choice, trichotomy, choice principle, and host real equality have no coordinate in the package. □

Theorem 363.8 (SignedDigitReal NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *seed surface consists of five finite obligations:*

1. *carrier: the* BHist *packet of Definition 363.1;*
2. *classifier: the finite-window comparison of Definition 363.2;*
3. *stability: carrier preservation under finite normalization windows as in Definition 363.3;*
4. *exact finite-window readback: the decoded* DyadicRatCore *enclosure row and its* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *routes;*
5. *ledger policy: the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 363.4, with quotient streams, selected representatives, trichotomy, choice, and host real equality excluded from the exported surface.*

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket namecert obligation surface`

Proof. The carrier and classifier rows provide the finite signed-digit source, the decoded dyadic enclosure rows, and the located-Cauchy route toward the real seal.

Stability is exactly the finite normalization theorem: a visible window read preserves the carrier and produces the dyadic enclosure row through the recorded history, transport, continuation, package, and certificate data.

Exact finite-window readback is the same displayed enclosure surface with its $\mathsf{DyadicApproximation}^{\uparrow}$ shape and $\mathsf{LocatedCauchy}^{\uparrow}$ precision witness. The ledger policy is the seal compatibility theorem,

which identifies the $\mathsf{Real}^{\uparrow}$ boundary while excluding all non-finite and non-constructive coordinates from the certificate. □

Theorem 363.9 (SignedDigitReal obligation transport package)**.** *Signed-digit stream rows, finite windows, dyadic enclosures,* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows transport together through the displayed* hsame, Cont, Pkg, *and* NameCert *ledger of an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier.*

The transport package is componentwise: it moves the signed-digit source, finite-window family, decoded DyadicRatCore *enclosures,* $\mathsf{DyadicApproximation}^{\uparrow}$ *routes, located-Cauchy rows, and real-seal boundary without adding quotient stream equality, selected representatives, trichotomy, choice, or host real equality.*

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket obligation transport package`

Proof. Project the carrier and the obligation ledger package. The coordinates S, W, E, and L list the source stream, finite windows, decoded dyadic enclosures, and located-Cauchy rows, while the real-seal boundary is read from the same enclosure and located rows.

Move each coordinate through the stored hsame rows and use the stored Cont routes to preserve the order from signed-digit prefix to normalization ledger, enclosure, located row, and seal boundary. The package provenance and local certificate row record that these are the only dependency names used by the transport.

Since no carrier coordinate stores quotient stream equality, a selected stream, a trichotomy branch, a choice datum, or host real equality, componentwise transport cannot export any such datum. □

Theorem 363.10 (SignedDigitReal exact readback boundary)**.** *Exact readback of every finite signed-digit window in an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier is exhausted by the decoded* DyadicRatCore *enclosure, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *route, the* $\mathsf{LocatedCauchy}^{\uparrow}$ *row, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary displayed in the obligation ledger package.*

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket exact readback boundary`

Proof. Choose a displayed finite window. The finite normalization window theorem Definition 363.3 reads that window through the signed-digit prefix and produces the corresponding decoded DyadicRatCore enclosure row.

The obligation ledger package then records the $\mathsf{DyadicApproximation}^{\uparrow}$ route attached to that enclosure and the $\mathsf{LocatedCauchy}^{\uparrow}$ row that exposes it at the requested precision. The seal ledger exactness theorem Definition 363.7 reads the real boundary from exactly those rows.

Thus the readback route stops at the displayed enclosure, approximation, located-Cauchy, and seal rows. The excluded quotient, selected representative, trichotomy, choice, and host equality data have no place in the route. □

Theorem 363.11 (SignedDigitReal tail-window export)**.** *Any accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *finite tail window exports to the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed* $\mathsf{SignedDigitStream}^{\uparrow}$ *prefix, the decoded* DyadicRatCore *enclosure, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *route, the* $\mathsf{LocatedCauchy}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *row.*

The export contains no quotient stream, selected representative, trichotomy branch, choice datum, or host real equality.

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket tail window export`

Proof. First apply Definition 363.3. A finite signed-digit tail window is read as a displayed $\mathsf{SignedDigitStream}^{\uparrow}$ prefix and produces the decoded $\mathsf{DyadicRatCore}$ enclosure row named by the carrier.

Next apply Definition 363.6. It moves the prefix, enclosure, approximation route, located-Cauchy row, and induced seal boundary through the componentwise hsame rows while staying inside the same obligation ledger.

Now use Definition 363.7 and Definition 363.10. These two statements identify the $\mathsf{Real}^{\uparrow}$ seal read with exactly the decoded enclosure, $\mathsf{DyadicApproximation}^{\uparrow}$ route, $\mathsf{LocatedCauchy}^{\uparrow}$ row, displayed Cont routes, Pkg provenance, and local NameCert row.

Since each cited row is a coordinate of the finite carrier or its obligation ledger, no quotient stream, selected representative, trichotomy branch, choice datum, or host real equality can be exported to the seal boundary. □

Theorem 363.12 (SignedDigitReal obligation-closure package)**.** *The* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier, classifier, finite normalization theorem, dyadic seal compatibility, window classifier stability, seal ledger exactness, obligation transport package, and exact readback boundary assemble the paper-side obligation-closure package for the* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Proof. The carrier and classifier define the finite signed-digit surface and its accepted comparisons. The finite normalization theorem supplies the per-window decoded dyadic enclosure, and dyadic seal compatibility identifies the induced $\mathsf{Real}^{\uparrow}$ boundary.

The obligation ledger package and window classifier stability bind these rows into a single finite transport surface. Seal ledger exactness shows that the real boundary consumes only the displayed decoded enclosures and located-Cauchy rows.

The obligation transport package records componentwise movement of the same rows, and the exact readback boundary states that every finite signed-digit window is exhausted by the decoded enclosure, approximation route, located-Cauchy row, and seal boundary. Together these statements close the paper obligation layer while leaving the formal axis unchanged. □

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket obligation closure package`

Theorem 363.13 (SignedDigitReal common-window seal determinacy)**.** *Let* $R = (S, W, E, L, H, C, P, N)$ *and* $R' = (S', W', E', L', H', C', P', N')$ *be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carriers. Suppose the signed-digit classifier relates* R *and* R' *on a common displayed finite window, and suppose the decoded* $\mathsf{DyadicRatCore}$ *enclosure rows on that window transport through the displayed* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *routes. Then the induced* $\mathsf{Real}^{\uparrow}$ *seal read is the same finite seal package on that window: it is read from the transported signed-digit prefix, normalization ledger, dyadic enclosure, approximation route, located-Cauchy row, and the recorded* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance.*

The determinacy is a common-window statement. It asserts no quotient stream equality, selected representative, trichotomy branch, choice datum, or host real equality.

Proof. Start with the finite normalization window theorem Definition 363.3. On the common window it reads each signed-digit prefix through the displayed carrier and produces the corresponding decoded $\mathsf{DyadicRatCore}$ enclosure row.

Apply the dyadic seal compatibility theorem Definition 363.4. The transported $\mathsf{DyadicApproximation}^{\uparrow}$ and $\mathsf{LocatedCauchy}^{\uparrow}$ routes ensure that both enclosure reads reach the same located-Cauchy surface before any $\mathsf{Real}^{\uparrow}$ seal boundary is consumed.

Use window classifier stability from Definition 363.6. It keeps the signed-digit source, finite window, decoded enclosure, located row, and seal boundary inside the same obligation ledger package under classifier-related transport.

Finally invoke seal ledger exactness from Definition 363.7. Since the seal consumes exactly the displayed decoded enclosure and located-Cauchy ledger, the two carriers determine the same finite seal package on the common window. The excluded quotient, representative, trichotomy, choice, and host-equality data are absent from the carrier and cannot enter the determinacy claim. □

Lean4 ✓ `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket common window seal determinacy`

Theorem 363.14 (SignedDigitReal tail-window seal determinacy)**.** *Let $R = (S, W, E, L, H, C, P, N)$ and $R' = (S', W', E', L', H', C', P', N')$ be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carriers. Suppose the two carriers agree on a displayed finite tail window through the* $\mathsf{SignedDigitStream}^{\uparrow}$ *classifier, and suppose the decoded* DyadicRatCore *enclosure rows transport through the displayed* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows. Then the* $\mathsf{Real}^{\uparrow}$ *seal read on that tail window is determined by the same finite prefix, normalization ledger, decoded enclosure, located row, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row.*

The determinacy is local to the tail window. It exports no quotient stream equality, selected signed-digit representative, trichotomy branch, choice datum, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First use the finite normalization window theorem and dyadic seal compatibility. The shared tail window is decoded into the displayed DyadicRatCore enclosure, and the enclosure is the datum routed toward the $\mathsf{Real}^{\uparrow}$ seal boundary.

Next apply window classifier stability. It keeps the signed-digit prefix, decoded enclosure, located row, and seal boundary attached to the same carrier ledger under componentwise hsame transport and displayed Cont replay.

Seal ledger exactness and the exact readback boundary identify the seal read with exactly the decoded enclosure and located-Cauchy rows visible on the finite tail window. No wider stream object or host equality row is consulted.

Finally apply common-window seal determinacy to the shared tail window. The two carriers therefore determine the same finite $\mathsf{Real}^{\uparrow}$ seal package through the prefix, normalization ledger, enclosure, located row, transport, continuation, package, and naming rows. □

Theorem 363.15 (SignedDigitReal scoped kernel interface)**.** *Every accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier exposes exactly the kernel rows named by its finite packet: the* $\mathsf{SignedDigitStream}^{\uparrow}$ *source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows reached by finite normalization, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *enclosure route, the* $\mathsf{LocatedCauchy}^{\uparrow}$ *located-window row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{SignedDigitReal}^{\uparrow}}$ *provenance rows.*

No consumer of the carrier can read a quotient stream, selected signed-digit representative, trichotomy branch, choice datum, host equality row, or additional analytic dependency outside this list.

Proof. Project the carrier

$$R = (S, W, E, L, H, C, P, N).$$

The source row S, window family W, decoded enclosure row E, located row L, transport row H, continuation row C, dependency package P, and local naming row N are precisely the displayed coordinates of Definition 363.1.

Apply the obligation ledger package of Definition 363.5. It identifies the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{LocatedCauchy}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ routes as reads of those same coordinates rather than extra carrier fields.

Finally use seal ledger exactness from Definition 363.7. The real-facing read consumes only the decoded enclosure and located-Cauchy ledger, so the exposed scoped interface is exhausted by the listed kernel rows. □

Theorem 363.16 (SignedDigitReal dependency scope exhaustion)**.** *Every public finite-window read of an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier is exhausted by the obligation ledger package, the finite normalization window theorem, seal-ledger exactness, and common-window seal determinacy. The read passes through the signed-digit source, the decoded dyadic enclosure, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *route, the* $\mathsf{LocatedCauchy}^{\uparrow}$ *row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the recorded* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

There is no remaining dependency slot for stream quotients, selected representatives, trichotomy, choice, host real equality, or real completeness.

Proof. Start from a displayed finite window in the obligation ledger package. The finite normalization theorem Definition 363.3 reads the signed-digit prefix and produces the corresponding decoded DyadicRatCore enclosure row.

Next route the enclosure through the displayed $\mathsf{DyadicApproximation}^{\uparrow}$ and $\mathsf{LocatedCauchy}^{\uparrow}$ rows. Seal ledger exactness in Definition 363.7 states that the $\mathsf{Real}^{\uparrow}$ boundary consumes exactly those rows and no wider analytic object.

For comparisons between carriers, apply common-window seal determinacy from Definition 363.13. It shows that a shared finite window reaches the same finite seal package through the same listed rows, so public reads have no dependency outside the obligation ledger. □

Theorem 363.17 (SignedDigitReal RealUp seal scope boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal boundary of an accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *carrier is reached only after a finite signed-digit prefix has been decoded into a* DyadicRatCore *enclosure and routed through the displayed* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows. The seal cannot be read directly from the signed-digit stream, from an infinite quotient, from a selected representative, or from host real equality.*

Proof. Choose a seal read from an accepted carrier. By the exact readback boundary Definition 363.10, the read first selects a finite signed-digit window and decodes it into the displayed dyadic enclosure row.

The obligation ledger package then supplies the $\mathsf{DyadicApproximation}^{\uparrow}$ route and the $\mathsf{LocatedCauchy}^{\uparrow}$ located row attached to that enclosure. These rows are the precision-facing data available before the real seal is consumed.

Apply seal ledger exactness. Since the seal consumes exactly the decoded enclosure and located-Cauchy ledger, any route that bypasses those rows or uses quotient, choice, selected-representative, or host-equality data is outside the $\mathsf{SignedDigitReal}^{\uparrow}$ carrier scope. □

Theorem 363.18 (SignedDigitReal public seal export)**.** *The public* $\mathsf{SignedDigitReal}^{\uparrow}$ *surface is exactly the finite route from signed-digit stream windows through* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosures,* $\mathsf{DyadicApproximation}^{\uparrow}$ *routes,* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary. Every exported seal read factors through the carrier, normalization window, dyadic seal compatibility, window classifier stability, seal ledger exactness, exact readback boundary, common-window seal determinacy, scoped kernel interface, and dependency scope exhaustion rows.*

Proof. Begin with the exact readback boundary. It selects a finite signed-digit window and decodes that window by the normalization theorem into the displayed DyadicRatCore$^{\uparrow}$ enclosure.

Route the decoded enclosure through dyadic seal compatibility, the DyadicApproximation$^{\uparrow}$ row, and the LocatedCauchy$^{\uparrow}$ row. Window classifier stability keeps these rows attached to the same displayed carrier under hsame transport.

Seal ledger exactness then identifies the Real$^{\uparrow}$ boundary as the only seal-consuming row. Common-window seal determinacy, the scoped kernel interface, and dependency scope exhaustion show that every public seal read is exhausted by this finite route. □

Theorem 363.19 (SignedDigitReal public representative exclusion)**.** *The public* SignedDigitReal$^{\uparrow}$ *seal surface contains no quotient stream equality, selected signed-digit representative, trichotomy branch, choice datum, or host real equality. Its consumers can read only the displayed signed-digit window, decoded dyadic enclosure,* DyadicApproximation$^{\uparrow}$ *route,* LocatedCauchy$^{\uparrow}$ *row,* Real$^{\uparrow}$ *seal boundary, and the recorded* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Project the scoped kernel interface of Definition 363.15. It lists the public coordinates available to a consumer and excludes quotient, representative, branch, choice, and host-equality data from the carrier.

Apply Definition 363.18. Every seal read factors through the finite window, dyadic enclosure, approximation route, located-Cauchy row, and real seal boundary.

Finally use dependency scope exhaustion from Definition 363.16. Since the listed rows exhaust every public read, no quotient stream equality, selected representative, trichotomy branch, choice datum, or host real equality can be recovered from the public surface. □

Theorem 363.20 (SignedDigitReal prefix refinement seal commutes)**.** *Let two accepted* SignedDigitReal$^{\uparrow}$ *packets refine the same signed-digit prefix window, and let the refined window be a narrower finite window accepted by the same window classifier. Then the refined window's decoded* DyadicRatCore *enclosure,* DyadicApproximation$^{\uparrow}$ *route,* LocatedCauchy$^{\uparrow}$ *row, and* Real$^{\uparrow}$ *seal boundary agree with the coarser route after componentwise* hsame *transport and the displayed* Cont *transport through the carrier ledger.*

Equivalently, every seal read along the refined route is the same finite Real$^{\uparrow}$ *boundary read as the coarser route once both reads are repacked through the shared signed-digit source row, decoded enclosure row, approximation route, located-Cauchy row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Start from the common accepted prefix window. The finite normalization window theorem Definition 363.3 decodes both the coarser and refined windows into the displayed DyadicRatCore enclosure rows, and window classifier stability from Definition 363.6 keeps those rows attached under componentwise hsame transport.

Next replay the route through the approximation and located-Cauchy rows. The carrier obligations and seal ledger exactness of Definition 363.7 state that the DyadicApproximation$^{\uparrow}$ route, LocatedCauchy$^{\uparrow}$ row, and Real$^{\uparrow}$ boundary are read only through the displayed Cont paths.

The exact readback boundary Definition 363.10 then identifies the refined read as a finite readback inside the same carrier ledger. Public representative exclusion from Definition 363.19 rules out any quotient stream, selected representative, trichotomy branch, choice datum, or host equality that could separate the two routes.

Thus the refined and coarser seal reads commute through the same finite ledger rows. Their decoded enclosure, approximation route, located-Cauchy row, and real seal boundary agree after the recorded hsame and Cont transports. □

Theorem 363.21 (SignedDigitReal window refinement directedness)**.** *Let $R = (S, W, E, L, H, C, P, N)$ and $R' = (S', W', E', L', H', C', P', N')$ be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *packets whose finite windows refine a common signed-digit source row. Then there is a shared finite refinement window for the two reads such that its decoded* DyadicRatCore *enclosure, its* $\mathsf{DyadicApproximation}^{\uparrow}$ *route, its* $\mathsf{LocatedCauchy}^{\uparrow}$ *row, and its* $\mathsf{Real}^{\uparrow}$ *seal boundary all factor through one shared finite budget recorded by the two packet ledgers and their componentwise* hsame *and* Cont *transports.*

The shared budget is directed only at the finite-window level. It exports no quotient stream, selected representative, trichotomy branch, choice datum, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the common signed-digit source row and the two accepted finite windows. Apply the finite normalization-window theorem Definition 363.3 to decode each window into its displayed DyadicRatCore enclosure row.

Use window-classifier stability from Definition 363.6 to transport the two decoded reads along the common signed-digit source. This identifies the prefix data, enclosure rows, approximation routes, located-Cauchy rows, and seal-boundary rows as finite ledger coordinates rather than as representatives of an infinite stream.

Now apply seal-ledger exactness Definition 363.7 and common-window seal determinacy Definition 363.13. These results force both seal reads to consume exactly the displayed decoded enclosure and located-Cauchy rows on a common finite window.

Finally use prefix-refinement seal commutation from Definition 363.20. It repacks the two refined routes through a single narrower window and the same finite BHist, hsame, Cont, Pkg, and NameCert provenance. Since every cited theorem works inside finite carrier coordinates, quotient streams, selected representatives, trichotomy, choice, host real equality, and $\mathsf{Real}^{\uparrow}$ completeness have no role in the directed shared-window budget. □

Theorem 363.22 (SignedDigitReal public refinement consistency)**.** *Let $R = (S, W, E, L, H, C, P, N)$ and $R' = (S', W', E', L', H', C', P', N')$ be accepted* $\mathsf{SignedDigitReal}^{\uparrow}$ *packets whose finite windows refine the same signed-digit source row. Suppose the decoded* DyadicRatCore *enclosure rows transport through the displayed* $\mathsf{DyadicApproximation}^{\uparrow}$ *and* $\mathsf{LocatedCauchy}^{\uparrow}$ *rows, and suppose the two routes are matched componentwise by* hsame *and the displayed* Cont *transports through their packet ledgers. Then the two packets determine the same public* $\mathsf{Real}^{\uparrow}$ *seal read.*

The public read is the finite seal boundary carried by the shared signed-digit source, the refined finite windows, decoded dyadic enclosures, approximation rows, located-Cauchy rows, $\mathsf{Real}^{\uparrow}$ *seal boundary,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It does not use quotient stream equality, selected representatives, trichotomy, choice, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite windows. The finite normalization theorem Definition 363.3 decodes both refined windows into displayed DyadicRatCore enclosure rows, and window classifier stability from Definition 363.6 keeps those rows attached to the same signed-digit source under componentwise hsame transport.

Next transport the decoded enclosure through the public route. Seal ledger exactness Definition 363.7 and exact readback boundary Definition 363.10 identify each $\mathsf{Real}^{\uparrow}$ read with the displayed $\mathsf{DyadicApproximation}^{\uparrow}$ and $\mathsf{LocatedCauchy}^{\uparrow}$ rows rather than with an external real representative.

Now pass to a common refinement. Prefix-refinement seal commutation from Definition 363.20 shows that a narrower accepted window preserves the same seal boundary, and window refinement directedness Definition 363.21 supplies such a shared finite window for the two accepted packets.

The two public reads therefore meet at the same finite $\mathsf{Real}^{\uparrow}$ seal boundary after the recorded hsame and Cont transports. Public representative exclusion from Definition 363.19 rules out quotient stream equality, selected representatives, trichotomy, choice, host real equality, and $\mathsf{Real}^{\uparrow}$ completeness as alternate ways to separate the two reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitReal}\uparrow}$.

Constructive story: A $\mathsf{SignedDigitReal}^{\uparrow}$ packet is generated from an accepted $\mathsf{SignedDigitStream}^{\uparrow}$ row, finite normalization windows, decoded DyadicRatCore enclosure rows, $\mathsf{DyadicApproximation}^{\uparrow}$ and $\mathsf{LocatedCauchy}^{\uparrow}$ routes, a $\mathsf{Real}^{\uparrow}$ seal boundary, and visible BHist, hsame, Cont, Pkg, and NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SignedDigitReal}\uparrow}$, scopedClosed)

Scope closed: The scoped kernel interface is exhausted by the carrier, classifier, finite normalization stability, seal compatibility, window classifier stability, exact readback boundary, transport package, common-window determinacy, and seal-boundary route over $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{LocatedCauchy}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SignedDigitReal}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.SignedDigitRealUp.SignedDigitRealPacket\_obligation\_closure\_package`

Bridge status: none

Origin: human

Not claimed: This chapter does not close full signed-digit regularity, RealUp completeness, quotient equality of streams, selected representatives, trichotomy, choice, host real equality, or bridge checking.

Upgrade path: Public export requires a downstream-facing object surface that consumes the same finite signed-digit, dyadic, located-Cauchy, and real-seal rows without widening the dependency scope.

Chapter 364

A Concrete Naming Certificate for $\mathsf{StationaryRationalCauchy}^{\uparrow}$

The $\mathsf{StationaryRationalCauchy}^{\uparrow}$ interface records the concrete stationary rational regular Cauchy input that appears in Definition 9238.6. It is the finite $\mathsf{Real}^{\uparrow}$ handoff certificate obtained by keeping the hidden $\mathsf{Rat}^{\uparrow}$ seed, its constant $\mathsf{StreamName}^{\uparrow}$ windows, the associated $\mathsf{RegSeqRat}^{\uparrow}$ member, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the local Pkg and $\mathsf{NameCert}$ rows visible.

The chapter consumes the diagonal representative supplied by chapter 362 and narrows it to the stationary rational Cauchy packet through the displayed $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ ledger. The dependency is substantive: the RealUp handoff below is the constant rational diagonal read with every non-stationary coordinate removed.

Definition 364.1 (StationaryRationalCauchy carrier)**.** *A* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (q, A, R, D, E, P, N)$$

with the following visible coordinates:

1. *q is the hidden* $\mathsf{Rat}^{\uparrow}$ *row supplying the rational seed;*
2. *A is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule whose every requested finite window reads the seed q;*
3. *R is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member whose every regular-sequence index returns the stream row A with the zero modulus row;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the same rational observation at every selected window;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the constant rational embedding of q;*
6. *P is the* Pkg *provenance row naming the* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{NameCert}$ *dependencies;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryRationalCauchy}^{\uparrow}}$ *row.*

The local classifier compares two carriers by the displayed hsame *transport of the seed rows, constant stream schedules, stationary regular-sequence members, dyadic ledgers, real seals, package provenance, and local certificate rows, with* Cont *routes recording the finite-window readback used by each consumer.*

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyTasteGate single carrier alignment`

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyUpTasteGate single carrier alignment ChapterTasteGate`

Theorem 364.2 (StationaryRationalCauchy NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{StationaryRationalCauchy}^{\uparrow}}$ *surface consists of four obligations over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows: carrier habitation by Definition 364.1; constant-stream classifier reflexivity for the* $\mathsf{Rat}^{\uparrow}$ *seed inside every* $\mathsf{StreamName}^{\uparrow}$ *window; stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *stability with the zero modulus and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger; and exact export of the constant rational* $\mathsf{Real}^{\uparrow}$ *seal without quotient stream equality, selected host limits, or ambient real completeness.*

Proof. Carrier habitation is the finite packet assembly. The seed row, constant stream schedule, stationary regular-sequence member, dyadic observation ledger, real seal, package provenance, and local certificate row are all visible coordinates of the BHist carrier.

Constant-stream reflexivity supplies the classifier row: each $\mathsf{StreamName}^{\uparrow}$ window reads the same $\mathsf{Rat}^{\uparrow}$ seed, so its hsame comparison is the reflexive comparison transported along the displayed Cont route for that finite window.

The regular Cauchy obligation is stationary. The zero modulus row is admissible because every member of the $\mathsf{RegSeqRat}^{\uparrow}$ family returns the same stream row and the same $\mathsf{DyadicRatCore}^{\uparrow}$ rational observation ledger.

The export obligation is ledger exactness. The Pkg and NameCert rows name exactly the dependency certificates used in the readback, and no field of the carrier contains a quotient of streams, a selected host limit, or an ambient completeness object. □

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyCarrier namecert obligations`

Theorem 364.3 (StationaryRationalCauchy diagonal handoff)**.** *For every accepted* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carrier* S *and every unary precision request* n*, the diagonal read used by the* $\mathsf{Real}^{\uparrow}$ *handoff factors through the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member of* S*, one finite constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{Rat}^{\uparrow}$ *seed row, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the visible* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by* S*.*

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyBHistCarrier diagonal handoff`

Proof. Project the packet of Definition 364.1. The requested precision n reaches the stationary regular-sequence member R, and R returns the same stream row at every index.

Read the scheduled finite window of A. Because A is constant, the selected window returns the seed row q with the same $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger independently of n.

The displayed hsame comparison transports this read through the stored Cont route, and the Pkg and NameCert rows record the provenance of the read. Thus the diagonal read is exactly the finite route listed in the statement. □

Theorem 364.4 (StationaryRationalCauchy finite-prefix uniformity)**.** *For every accepted* StationaryRationalCauchy *carrier* $S = (q, A, R, D, E, P, N)$ *and every finite unary prefix of precision requests, each request in the prefix reads the same* $\mathsf{Rat}^{\uparrow}$ *seed* q *through the same constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* A*, the same stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member* R*, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*. The only variation permitted across the prefix is the displayed* hsame *transport and* Cont *route used to reach the repeated rows.*

Proof. Induct over the finite prefix. The empty prefix has no requested read, so the carrier coordinates already supply the uniform seed, schedule, regular-sequence member, and ledger.

For a prefix extended by one request n, apply Definition 364.3 to the final request. The handoff routes n through R, then through the constant schedule A, and then back to the seed row q and ledger D.

The induction hypothesis gives the same rows for the earlier requests. Since the carrier of Definition 364.1 is stationary in q, A, R, D, the final request introduces no new stream schedule, regular-sequence member, or dyadic ledger. Its displayed hsame and Cont rows are the only request-indexed data. □

Theorem 364.5 (StationaryRationalCauchy classifier correspondence)**.** *Let $S = (q, A, R, D, E, P, N)$ and $S' = (q', A', R', D', E', P', N')$ be accepted* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carriers. The local classifier relates S and S' exactly when the displayed* $\mathsf{Rat}^{\uparrow}$ *classifier relates q and q' and the associated* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, Pkg, *and* NameCert *rows are transported componentwise by the recorded* hsame *comparisons and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyCarrier classifier correspondence`

Proof. For the forward direction, inspect the local classifier. Its only seed-sensitive coordinate is the hidden rational row, and every stream, regular-sequence, dyadic, real-seal, package, and certificate coordinate is defined as the stationary readback of that seed. Projecting the classifier therefore gives the displayed $\mathsf{Rat}^{\uparrow}$ comparison and the componentwise transport rows.

For the reverse direction, start with the $\mathsf{Rat}^{\uparrow}$ comparison between q and q'. Constant-stream reflexivity transports it to every finite $\mathsf{StreamName}^{\uparrow}$ window of A and A'.

The same finite-window transport lifts to the stationary $\mathsf{RegSeqRat}^{\uparrow}$ members because all regular-sequence indices read those constant stream rows. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers and $\mathsf{Real}^{\uparrow}$ seals are then the diagonal readbacks of the same transported windows.

Repack the transported coordinates with P, P' and N, N'. Since the local classifier compares exactly these rows, the two carriers are classified together. □

Theorem 364.6 (StationaryRationalCauchy ledger exactness)**.** *Every consumer-facing* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carrier is exhausted by the finite ledger*

$$\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows. The carrier exports no additional equality principle, host-real endpoint, non-stationary Cauchy family, or completeness theorem.*

Proof. Begin with the diagonal handoff of Definition 364.3. It shows that each consumer read first reaches the stationary $\mathsf{RegSeqRat}^{\uparrow}$ member and then a finite constant $\mathsf{StreamName}^{\uparrow}$ window.

The window read returns the hidden $\mathsf{Rat}^{\uparrow}$ seed and the matching $\mathsf{DyadicRatCore}^{\uparrow}$ observation. The $\mathsf{Real}^{\uparrow}$ seal row is attached to that same constant rational readback.

The remaining carrier coordinates are provenance: BHist supplies the finite packet, hsame and Cont supply the visible transports, and Pkg and NameCert name the certificate rows. These fields exhaust the exported ledger, so no further equality principle, host-real endpoint, non-stationary family, or completeness theorem is available from the carrier. □

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyCarrier ledger exactness`

Theorem 364.7 (StationaryRationalCauchy RegSeqRat handoff scope)**.** *The constant rational embedding carried by* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *into* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is scoped entirely by the displayed* $\mathsf{Rat}^{\uparrow}$ *seed, constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not require quotient stream equality, selected host limits, or ambient completeness.*

Proof. Begin with the carrier of Definition 364.1. Its finite packet exposes the seed q, constant schedule A, stationary member R, dyadic ledger D, real seal E, package row P, and local certificate row N.

The diagonal handoff of Definition 364.3 sends each precision request through R and then through a finite window of A back to q and D. The finite-prefix uniformity of Definition 364.4 shows that repeating the read across any finite request list introduces no new stream schedule, regular-sequence member, or ledger.

Classifier correspondence in Definition 364.5 transports exactly the displayed seed, stream, regular-sequence, dyadic, real, package, and certificate rows. Ledger exactness from Definition 364.6 then exhausts the exported $\mathsf{Real}^{\uparrow}$ read, excluding quotient stream equality, selected host limits, and ambient completeness from the handoff scope. □

Lean4 ✓ `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyUp StdBridge`

Theorem 364.8 (StationaryRationalCauchy RealUp embedding exactness)**.** *For every accepted* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carrier* $S = (q, A, R, D, E, P, N)$*, the exported* $\mathsf{Real}^{\uparrow}$ *seal* E *is exactly the constant rational embedding route through the* $\mathsf{Rat}^{\uparrow}$ *seed* q*, the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* A*, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *member* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Equivalently, every consumer read of E *is exhausted by the finite route*

$$\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

inside the carrier S*. The exported seal does not use quotient stream equality, a selected host limit, or ambient completeness.*

Proof. First read the diagonal route. By Definition 364.3, every precision request reaches the stationary member R, then a finite constant window of A, and then the seed row q with the matching ledger D.

Next make the route uniform. The finite-prefix uniformity theorem shows that any finite family of requests reads the same q, A, R, and D rows. Thus the $\mathsf{Real}^{\uparrow}$ seal cannot depend on a non-stationary stream coordinate.

Transport and exactness are then componentwise. Classifier correspondence moves exactly the displayed seed, stream, regular-sequence, dyadic, seal, package, and certificate rows, while ledger exactness exhausts the $\mathsf{Real}^{\uparrow}$ read by the finite chain shown above.

Finally apply the RegSeqRat handoff scope theorem. It identifies the same stationary route as the entire $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff, so E is precisely the constant rational embedding of q and not a quotient stream, selected host limit, or ambient completeness construction. □

Theorem 364.9 (StationaryRationalCauchy obligation package)**.** *For every accepted* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *carrier* $S = (q, A, R, D, E, P, N)$*, the obligation package is the finite package assembled from the carrier, the* NameCert *obligations, the diagonal handoff, finite-prefix uniformity, classifier correspondence, ledger exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff scope, and* $\mathsf{Real}^{\uparrow}$ *embedding exactness rows. Its visible coordinates are the hidden* $\mathsf{Rat}^{\uparrow}$ *seed* q*, constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* A*, stationary*

RegSeqRat$^{\uparrow}$ *member* R, DyadicRatCore$^{\uparrow}$ *ledger* D, Real$^{\uparrow}$ *seal* E, *package row* P, *local* NameCert *row* N, *and the displayed* BHist, hsame, *and* Cont *witnesses.*

Proof. First project the carrier and the local certificate obligations from Definition 364.1 and Definition 364.2. These rows provide the seed, constant schedule, stationary member, dyadic ledger, real seal, package provenance, local name, and transport witnesses.

Next assemble the finite request behavior. The diagonal handoff sends each precision request through R, A, q, and D, while finite-prefix uniformity shows that finite request lists reuse the same stationary rows.

Finally add the classifier, ledger, handoff, and embedding rows. Definition 364.5, Definition 364.6, Definition 364.7, and Definition 364.8 identify the same finite stationary package as the full obligation surface. □

Theorem 364.10 (StationaryRationalCauchy stationary readback)**.** *Every stationary readback from an accepted* StationaryRationalCauchy$^{\uparrow}$ *carrier factors through the hidden* Rat$^{\uparrow}$ *seed* q, *constant* StreamName$^{\uparrow}$ *windows* A, *stationary* RegSeqRat$^{\uparrow}$ *member* R, DyadicRatCore$^{\uparrow}$ *ledger* D, *package row* P, *local* NameCert *row* N, *and the displayed* BHist, hsame, *and* Cont *rows. No readback introduces a non-stationary stream, quotient stream equality, selected host limit, or ambient completeness datum.*

Proof. Begin with Definition 364.3. A precision request reaches R, then the constant window A, then the seed q and dyadic ledger D.

Use Definition 364.4 for multiple requests. The same q, A, R, D rows serve every finite prefix, so no new stream schedule or regular-sequence member appears.

Finally apply Definition 364.5. The package, certificate, BHist, hsame, and Cont rows transport the same stationary data componentwise, preserving the exclusions in the statement. □

Theorem 364.11 (StationaryRationalCauchy RealUp seal non-escape)**.** *The* Real$^{\uparrow}$ *seal exported by an accepted* StationaryRationalCauchy$^{\uparrow}$ *carrier does not provide quotient Cauchy classes, host sequence equality, selected limits, or ambient completeness. It is exactly the finite stationary route through the* Rat$^{\uparrow}$ *seed, constant* StreamName$^{\uparrow}$ *windows, stationary* RegSeqRat$^{\uparrow}$ *member,* DyadicRatCore$^{\uparrow}$ *ledger,* Pkg *provenance,* NameCert *row, and displayed* BHist, hsame, *and* Cont *witnesses.*

Proof. First read the seal through Definition 364.8. The seal is the constant rational embedding route through q, A, R, D, E, P, N.

Next use Definition 364.6. Ledger exactness exhausts the Real$^{\uparrow}$ read by the dyadic observation and stationary rational rows, leaving no coordinate for quotient Cauchy classes or host sequence equality.

Finally apply Definition 364.9. Since the obligation surface contains only the finite stationary carrier rows and their displayed transports, selected limits and ambient completeness are not exported by the seal. □

Closure status for NameCert$_{\text{StationaryRationalCauchy}^{\uparrow}}$.

Constructive story: A StationaryRationalCauchy$^{\uparrow}$ packet is generated from a hidden Rat$^{\uparrow}$ seed, a constant StreamName$^{\uparrow}$ schedule, a stationary RegSeqRat$^{\uparrow}$ member, a DyadicRatCore$^{\uparrow}$ observation ledger, a Real$^{\uparrow}$ seal, and displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Theory closure: TheoryGate(NameCert$_{\text{StationaryRationalCauchy}^{\uparrow}}$, publicClosed)

Scope closed: Definition 364.1, Definition 364.2, and the diagonal handoff, finite-prefix uniformity, classifier correspondence, ledger exactness, RegSeqRat$^{\uparrow}$ handoff scope, and Real$^{\uparrow}$ embedding exactness rows export the public stationary rational Real$^{\uparrow}$ embedding surface.

Formal verification: FormalStatus(NameCert$_{\text{StationaryRationalCauchy}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.StationaryRationalCauchyUp.StationaryRationalCauchyCarrier\_namecert\_obligations`
Bridge status: none
Origin: human
Not claimed: This chapter does not close the exact kernel dependency scope, bridge surface, or completion uniqueness obligations.
Upgrade path: Bridge closure requires recording the standard bridge row as the bridge surface without claiming completion uniqueness.

Chapter 365

A Concrete Naming Certificate for $\mathsf{FiniteWindowEnvelope}^{\uparrow}$

The $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ interface is the concrete finite observation boundary for the Real and Completion phase. It packages a selected $\mathsf{StreamName}^{\uparrow}$ window, its $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ seal row that reads the same rational window, and the $\mathsf{Real}^{\uparrow}$ endpoint classifier row that exposes the window to real-number consumers. The envelope is deliberately finite: it certifies exactly the inspected anchor, window, and scale rows, and it exports no diagonal limit, quotient of streams, host real equality, or completeness theorem.

The envelope is the finite-window handoff between chapter 126, chapter 291, chapter 301, and chapter 38. It consumes a selected $\mathsf{StreamName}^{\uparrow}$ window, a dyadic observation ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ seal row, and a $\mathsf{Real}^{\uparrow}$ endpoint classifier as one finite package.

Definition 365.1 (FiniteWindowEnvelope carrier)**.** *A* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier is a finite* BHist *package*

$$W = (S, A, B, L, Q, G, E, P, N)$$

with the following visible coordinates:

1. *S is the* $\mathsf{Rat}^{\uparrow}$ *stream source named by the surrounding* $\mathsf{StreamName}^{\uparrow}$ *certificate;*
2. *A is the selected finite anchor row, recording the inspected source point and the continuation route that reaches it;*
3. *B is the finite window row, recording the requested length, scale, and local subwindow bounds;*
4. *L is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for every rational read made inside B;*
5. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *window classifier row for the same source, anchor, and window;*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row that reads the dyadic ledger as a regular rational sequence window;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row for the same finite window;*
8. *P is the* Pkg *provenance naming the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{NameCert}$ *source certificates;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowEnvelope}^{\uparrow}}$ *provenance row.*

The local classifier relates two carriers exactly when their selected finite windows agree by the $\mathsf{StreamName}^{\uparrow}$ *window classifier, their* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers are equal row by row, their* $\mathsf{RegSeqRat}^{\uparrow}$

seal rows classify the same regular rational window, and their $\mathsf{Real}^{\uparrow}$ *finite-window endpoint classifiers agree after the displayed* hsame *and* Cont *transports.*

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeTasteGate single carrier alignment`

Theorem 365.2 (FiniteWindowEnvelope NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteWindowEnvelope}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier formation from the finite* BHist *package of Definition 365.1;*
2. *classifier reflexivity, symmetry, and transitivity for the combined* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows;*
3. *carrier transport under displayed* hsame *comparisons and finite window refinement routes;*
4. *seal handoff from the selected* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger to the* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
5. *ledger exhaustion for the hidden anchor, selected window, scale, subwindow bound, dyadic readback, and endpoint classifier rows.*

Proof. Carrier formation is immediate from the displayed finite package: every coordinate is a visible row, and the package provenance names exactly the certificates that own those rows.

For the classifier laws, apply the component laws already required by the four dependency certificates. Reflexivity uses the same source, anchor, window, dyadic ledger, regular-sequence seal, and endpoint row. Symmetry reverses each displayed classifier row. Transitivity composes the four component comparisons while keeping the anchor and window coordinates synchronized by the common $\mathsf{StreamName}^{\uparrow}$ row.

Transport under hsame and Cont is componentwise. A transported anchor transports its finite window, the dyadic ledger follows the same requested readbacks, and the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows are repacked against those transported finite reads.

The seal handoff and ledger exhaustion are the remaining two rows. The seal row consumes the same ledger recorded by the carrier, and the exhaustion row says that every finite observation has already appeared as one of the displayed anchor, window, scale, dyadic, regular-sequence, or endpoint coordinates. □

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeCarrier namecert obligations`

Theorem 365.3 (FiniteWindowEnvelope RegSeqRat seal handoff)**.** *For every accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row is determined by the selected* $\mathsf{StreamName}^{\uparrow}$ *window together with the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger. If two carriers have the same selected window and equal dyadic ledgers after the displayed transports, then their* $\mathsf{RegSeqRat}^{\uparrow}$ *seal rows classify the same regular rational finite observation.*

Proof. Project the carrier of Definition 365.1. The row Q names the finite $\mathsf{StreamName}^{\uparrow}$ window, while L records every dyadic rational read performed inside that window.

The $\mathsf{RegSeqRat}^{\uparrow}$ seal row G is attached to those same readbacks. Therefore a second carrier with the same selected window and the same dyadic ledger presents the same regular rational finite observation to the seal interface.

Transport only changes the displayed route used to reach the rows. Since the classifier compares the transported window and ledger before reading the seal row, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is invariant under the permitted hsame and Cont transports. □

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeCarrier regseqrat seal handoff`
↪ variant `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier regseqrat seal handoff`

Theorem 365.4 (FiniteWindowEnvelope RealUp window exactness)**.** *For every accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier, a* $\mathsf{Real}^{\uparrow}$ *finite-window endpoint read factors through exactly one selected* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and one* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. No endpoint read bypasses the carrier by appealing to a host real value, a quotient representative, a diagonal limit, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier realup window exactness`

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ endpoint read of the carrier. The endpoint row E is one of the visible coordinates of the source package, so the read first exposes the anchor and finite window rows to which E is attached.

The window row is classified by Q, and every rational observation inside it is recorded in L. By Definition 365.3, the regular rational view of the same finite observation is exactly the seal row G.

Thus the endpoint read has no independent source of data. It reaches the $\mathsf{Real}^{\uparrow}$ classifier only through the selected stream window, the dyadic ledger, and the regular-sequence seal, leaving no host value or completion object available to consumers. □

Theorem 365.5 (FiniteWindowEnvelope common refinement exactness)**.** *Let* $W = (S, A, B, L, Q, G, E, P, N)$ *and* $W' = (S', A', B', L', Q', G', E', P', N')$ *be accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carriers. If the local classifier relates* W *and* W'*, then their comparison factors through one displayed selected* $\mathsf{StreamName}^{\uparrow}$ *finite window, one transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, one* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row, and one* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row. Conversely, any such four transported rows, with the package and local certificate provenance already carried by* P, N, P', N'*, repack into the local* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *classifier.*

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier common refinement exactness`

Proof. For the forward direction, project the carrier tuple of Definition 365.1. The classifier can inspect only the selected window row Q, the dyadic ledger L, the seal row G, the endpoint row E, and their transported counterparts.

Use Definition 365.3 to identify the transported $\mathsf{RegSeqRat}^{\uparrow}$ seal from the common selected window and the common dyadic ledger. Then use Definition 365.4 to identify the endpoint classifier as the read of that same finite window through the same seal row.

It remains only to rule out hidden comparison data. Apply Definition 365.6: every anchor, scale, subwindow, readback, seal, endpoint, package, and certificate row has already appeared in the displayed carrier coordinates.

For the reverse direction, start with the displayed $\mathsf{StreamName}^{\uparrow}$ window, transported $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ seal, and $\mathsf{Real}^{\uparrow}$ endpoint classifier rows. Repack them with the provenance rows P, N, P', N' into the component order of Definition 365.1. Since the local classifier is exactly the componentwise comparison of those rows, the two carriers classify together. □

Theorem 365.6 (FiniteWindowEnvelope ledger exhaustion)**.** *Every accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier exhausts the hidden finite observation data: the anchor, window length, scale, subwindow*

bounds, selected offsets, dyadic readbacks, RegSeqRat$^{\uparrow}$ *seal,* Real$^{\uparrow}$ *endpoint classifier,* Pkg *provenance, and local* NameCert *row all occur as displayed coordinates of the carrier or as displayed component rows of those coordinates.*

Proof. Inspect the finite package W of Definition 365.1. The anchor coordinate records the selected source point and its continuation route, while the window coordinate records the length, scale, local bounds, and offsets.

The dyadic ledger coordinate records the rational readbacks. The StreamName$^{\uparrow}$ classifier, RegSeqRat$^{\uparrow}$ seal, and Real$^{\uparrow}$ endpoint classifier then consume those same rows rather than introducing additional finite observations.

The provenance coordinates P and N account for the package and local naming rows. Since every finite observation field is present in these coordinates or in their displayed component ledgers, the carrier leaves no hidden anchor, window, scale, or endpoint row outside the certificate. □

Theorem 365.7 (FiniteWindowEnvelope StreamName RegSeqRat scope)**.** *Every accepted* FiniteWindowEnvelope$^{\uparrow}$ *read from a selected finite* StreamName$^{\uparrow}$ *window to a* RegSeqRat$^{\uparrow}$ *seal is exhausted by the anchor row, window row,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *seal,* Real$^{\uparrow}$ *endpoint classifier,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in the carrier. The handoff has no additional hidden window selector, rational readback table, endpoint classifier, or completion-side equality field.*

Proof. Project the carrier of Definition 365.1. The selected anchor and window rows name the finite StreamName$^{\uparrow}$ read, while the dyadic ledger records every rational observation made inside that read.

By Definition 365.3, the RegSeqRat$^{\uparrow}$ seal consumes exactly that selected window and ledger. By Definition 365.4, the Real$^{\uparrow}$ endpoint classifier is attached to the same finite window rather than to a separate endpoint object.

The common refinement theorem Definition 365.5 shows that transported comparisons factor through the same displayed window, ledger, seal, and endpoint rows. Finally, Definition 365.6 accounts for the anchor, window, scale, subwindow, readback, package, and certificate rows, so the StreamName$^{\uparrow}$ to RegSeqRat$^{\uparrow}$ handoff is scoped by the listed data. □

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier streamname regseqrat scope`

Theorem 365.8 (FiniteWindowEnvelope tail projection)**.** *For every accepted* FiniteWindowEnvelope$^{\uparrow}$ *carrier* $W = (S, A, B, L, Q, G, E, P, N)$*, any displayed tail read or subwindow read of the accepted envelope factors through the same anchor row* A*, selected* StreamName$^{\uparrow}$ *finite window* Q*,* DyadicRatCore$^{\uparrow}$ *ledger* L*,* RegSeqRat$^{\uparrow}$ *seal* G*,* Real$^{\uparrow}$ *endpoint classifier* E*, and visible* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates already listed in the carrier. The projection introduces no quotient stream, diagonal completion, host real equality, or completeness object outside those rows.*

Proof. Project the carrier of Definition 365.1. The anchor, finite window, dyadic ledger, regular-sequence seal, endpoint classifier, package row, and local naming row are displayed coordinates, and the permitted tail or subwindow read must start from that finite window surface.

Apply Definition 365.3 to identify the RegSeqRat$^{\uparrow}$ view of the read with the same selected StreamName$^{\uparrow}$ window and DyadicRatCore$^{\uparrow}$ ledger. Then use Definition 365.5 to keep transported comparisons on the same window, ledger, seal, and endpoint rows.

The exhaustion step is Definition 365.6, while Definition 365.7 supplies the non-escape boundary for the stream-to-regular-sequence handoff. Hence every displayed tail or subwindow read is a projection of the listed carrier coordinates. □

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier tail projection`

Theorem 365.9 (FiniteWindowEnvelope tail-projection audit sync)**.** *Every accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier assembles its selected finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier, displayed* hsame *and* Cont *transports,* Pkg *provenance, and local* NameCert *rows into one audit-facing consumer package. Any tail read or subwindow read of the carrier factors through that package, and no such consumer route exports a diagonal limit, quotient stream, host real equality, or completeness object.*

Proof. Start with an accepted carrier $W = (S, A, B, L, Q, G, E, P, N)$. The carrier definition exposes the selected finite window Q, the dyadic ledger L, the seal row G, the endpoint row E, the package provenance P, and the local naming row N as visible coordinates.

The consumer route is assembled by the component exactness results. The handoff theorem Definition 365.3 identifies the $\mathsf{RegSeqRat}^{\uparrow}$ seal from the selected $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger. The endpoint theorem Definition 365.4 ties the $\mathsf{Real}^{\uparrow}$ classifier to the same finite window and seal.

For comparisons between accepted carriers, Definition 365.5 keeps the transported window, ledger, seal, and endpoint rows on one displayed refinement package. The exhaustion theorem Definition 365.6 accounts for the anchor, window, scale, subwindow, readback, provenance, and naming rows.

Finally, Definition 365.7 gives the stream-to-regular-sequence boundary, and Definition 365.8 applies that boundary to tail and subwindow reads. Thus every audit-facing read is a consumer of the same displayed package, with no remaining route to a diagonal limit, quotient stream, host real equality, or completeness object. □

Theorem 365.10 (FiniteWindowEnvelope endpoint-source determinacy)**.** *Let* $W = (S, A, B, L, Q, G, E, P, N)$ *be an accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier. Every endpoint read exposed by* W *has exactly the displayed source chain*

$$(Q, L, G, E)$$

where Q *is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite window,* L *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* G *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row, and* E *is the* $\mathsf{Real}^{\uparrow}$ *endpoint classifier row. The read has no second endpoint source outside the listed carrier coordinates and their displayed* hsame *and* Cont *transports.*

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeBHistCarrier endpoint source determinacy`

Proof. Project the carrier of Definition 365.1. The coordinates Q, L, G, E are the only rows in the package that name a selected window, record its rational observations, seal those observations as a regular rational window, and expose the endpoint classifier.

Apply Definition 365.4 to force every endpoint read through the same selected window, dyadic ledger, and seal row. Apply Definition 365.5 to keep transported comparisons on those four rows.

Finally, Definition 365.6 accounts for the anchor, scale, subwindow, readback, package, and local naming data. Since no hidden row remains outside the displayed carrier coordinates, the endpoint source is determined by (Q, L, G, E) together with the displayed transports. □

Theorem 365.11 (FiniteWindowEnvelope subwindow refinement closure)**.** *Let* $W = (S, A, B, L, Q, G, E, P, N)$ *be an accepted* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *carrier. Any displayed tail refinement or subwindow refinement of* W *repacks as the same* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *surface over the transported anchor row, selected window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal,* $\mathsf{Real}^{\uparrow}$ *endpoint classifier,* hsame,

Cont, Pkg, *and* NameCert *rows. The repacked surface exports no data beyond those transported rows.*

Proof. Begin with a displayed tail or subwindow refinement. By Definition 365.8, the refinement factors through the anchor, window, ledger, seal, endpoint, transport, package, and naming rows already displayed in W.

The stream-to-regular-sequence boundary is fixed by Definition 365.7. The endpoint row is fixed by Definition 365.10, so the refinement cannot choose an independent endpoint source after transport.

Repack the transported rows in the carrier order of Definition 365.1. The classifier of the repacked envelope is the same componentwise comparison described by the carrier definition, so the transported tail or subwindow read remains closed inside the FiniteWindowEnvelope$^{\uparrow}$ surface. □

Theorem 365.12 (FiniteWindowEnvelope RegularCauchyName consumer export)**.** *A* RegularCauchyName$^{\uparrow}$ *consumer, or a Real and Completion consumer that reads a* FiniteWindowEnvelope$^{\uparrow}$ *carrier, can read only the consumer package assembled from endpoint-source determinacy, subwindow refinement closure, and the audit-facing tail projection package of Definition 365.9. Such a consumer obtains the selected* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *seal,* Real$^{\uparrow}$ *endpoint row, displayed* hsame *and* Cont *transports,* Pkg *provenance, and* NameCert *row, but obtains no diagonal limit, quotient stream, host real equality, selected limit, or completeness object.*

Proof. The consumer first receives the audit-facing package of Definition 365.9. That package already lists the selected finite window, dyadic ledger, regular rational seal, endpoint classifier, transport rows, package provenance, and local naming row.

Endpoint-source determinacy identifies the endpoint reads with the same four source rows (Q, L, G, E). Subwindow refinement closure then shows that every tail or subwindow route repacks inside the same transported envelope surface.

Therefore a RegularCauchyName$^{\uparrow}$ consumer, and likewise a Real and Completion consumer, has access exactly to the displayed finite-window package. The excluded objects are not coordinates of the package and are not produced by the cited projection, determinacy, or refinement-closure steps. □

Theorem 365.13 (FiniteWindowEnvelope StreamName tail lock)**.** *Let* $W = (S, A, B, L, Q, G, E, P, N)$ *be an accepted* FiniteWindowEnvelope$^{\uparrow}$ *carrier. Every tail read and every subwindow read of the* BHist *package remains locked to the selected* StreamName$^{\uparrow}$ *window* Q*, the* DyadicRatCore$^{\uparrow}$ *ledger* L*, the* RegSeqRat$^{\uparrow}$ *seal row* G*, the* Real$^{\uparrow}$ *endpoint row* E*, and the displayed* hsame, Cont, Pkg, *and* NameCert *coordinates* P, N*. The read cannot switch to another stream window, another dyadic ledger, another regular rational seal, or a host real endpoint.*

Proof. Project the carrier coordinates. The selected window Q, ledger L, seal G, endpoint row E, package row P, and naming row N are visible rows of the same finite BHist package.

Apply Definition 365.8 to any tail or subwindow read. It factors the read through the displayed anchor, window, ledger, seal, endpoint, transport, continuation, package, and naming coordinates.

Use Definition 365.11 to repack the transported rows after refinement. Since the repacked surface has the same listed coordinates as its source, the read remains locked to the selected StreamName$^{\uparrow}$ window and its associated finite ledger. □

Theorem 365.14 (FiniteWindowEnvelope common window lock)**.** *Let* $W = (S, A, B, L, Q, G, E, P, N)$ *and* $W' = (S', A', B', L', Q', G', E', P', N')$ *be accepted* FiniteWindowEnvelope$^{\uparrow}$ *carriers. Every local classifier comparison between them factors through one common selected finite* StreamName$^{\uparrow}$

window and one transported DyadicRatCore$^{\uparrow}$ *ledger, after which the* RegSeqRat$^{\uparrow}$ *seal rows,* Real$^{\uparrow}$ *endpoint rows, displayed* hsame *and* Cont *routes,* Pkg *provenance, and* NameCert *rows are read from that same finite package. No comparison field is supplied by a quotient stream, a diagonal object, or host real equality.*

Proof. Start from the local classifier comparison. By Definition 365.5, the comparison exposes a single selected window row, a single transported dyadic ledger, one regular-sequence seal row, and one endpoint classifier row.

The seal is determined by Definition 365.3. Thus the comparison cannot choose a seal independently of the selected StreamName$^{\uparrow}$ window and the transported DyadicRatCore$^{\uparrow}$ ledger.

The endpoint side is determined by Definition 365.10. The exhaustion row Definition 365.6 then accounts for the transport, continuation, package, and naming rows, leaving no additional comparison source outside the finite carrier. □

Theorem 365.15 (FiniteWindowEnvelope Real seal root unblock)**.** *The* FiniteWindowEnvelope$^{\uparrow}$ *carrier supplies the root package needed by downstream Real,* RegSeqRat$^{\uparrow}$*, and Completion consumers: an accepted* BHist *package binds the selected* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *seal,* Real$^{\uparrow}$ *endpoint row, displayed* hsame *and* Cont *routes,* Pkg *provenance, and local* NameCert *row into one finite consumer surface. This package exports the exact finite-window endpoint row of Definition 365.4 and the audit-facing tail projection of Definition 365.9, but exports no quotient stream, host real equality, diagonal limit, or completeness theorem.*

Proof. Build the consumer surface from the carrier coordinates. The selected window, dyadic ledger, regular rational seal, endpoint row, transport routes, continuation routes, package provenance, and local naming row are all displayed in the same finite BHist package.

Use Definition 365.4 to identify the Real$^{\uparrow}$ endpoint read with the selected StreamName$^{\uparrow}$ window, the DyadicRatCore$^{\uparrow}$ ledger, and the RegSeqRat$^{\uparrow}$ seal. This supplies the Real-facing row.

Use Definition 365.9 to expose the same rows to tail and subwindow consumers. The StreamName tail lock and common window lock above keep refinements and comparisons inside the same finite package, so downstream Real, RegSeqRat$^{\uparrow}$, and Completion consumers receive one root-unblock surface and no external equality or completeness object. □

Lean4 ✓ `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeUp StdBridge`

Closure status for NameCert$_{\text{FiniteWindowEnvelope}^{\uparrow}}$.

Constructive story: A FiniteWindowEnvelope$^{\uparrow}$ packet is generated from a finite StreamName$^{\uparrow}$ window, a DyadicRatCore$^{\uparrow}$ ledger, a RegSeqRat$^{\uparrow}$ seal row, a Real$^{\uparrow}$ endpoint classifier, displayed transport and continuation rows, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{FiniteWindowEnvelope}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature theory scope is the standard finite-window envelope interface together with the consumer examples, tail lock, common window lock, and Real seal root-unblock package over the same StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ ledger, RegSeqRat$^{\uparrow}$ seal, Real$^{\uparrow}$ endpoint classifier, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteWindowEnvelope}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FiniteWindowEnvelopeUp.FiniteWindowEnvelopeCarrier\_regseqrat\_seal\_handoff`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close diagonal limits, quotient streams, host real equality, or completeness.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 366

A Concrete Naming Certificate for $\mathsf{DyadicSubdivision}^{\uparrow}$

The $\mathsf{DyadicSubdivision}^{\uparrow}$ interface names a finite subdivision certificate for one rational interval into dyadic mesh cells with validated enclosure rows attached. In the Moore–Kearfott–Cloud interval-analysis reading, subdivision refines an enclosure into finitely many checked sub-enclosures; in BEDC the object is only a finite BHist ledger over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{ValidatedNumerics}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

The unary bisection route used below is read through the sibling chapter 361 before being repacked as an ordered subdivision ledger. $\mathsf{DyadicSubdivision}^{\uparrow}$ consumes $\mathsf{DyadicBisection}^{\uparrow}$ only as a finite route source; it does not import an infinite refinement tree.

Definition 366.1 (DyadicSubdivision source)**.** *A* $\mathsf{DyadicSubdivision}^{\uparrow}$ *source is a finite* BHist *packet*

$$S = (I, L, C, M, V, P, N)$$

with the following visible rows:

1. *I is the parent* $\mathsf{RationalInterval}^{\uparrow}$ *row whose endpoint and order data are the interval being subdivided;*
2. *L is the unary dyadic level row used to read every child cell at one displayed mesh precision;*
3. *C is a finite ordered cell-list row, with each entry carrying its displayed position inside the parent interval;*
4. *M is the finite family of* $\mathsf{DyadicMesh}^{\uparrow}$ *cell packets indexed by C;*
5. *V is the finite family of* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment rows attached to the same cells;*
6. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{ValidatedNumerics}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSubdivision}^{\uparrow}}$ *row.*

The classifier compares accepted subdivision packets by transporting the parent interval, level row, ordered cell list, mesh cells, containment rows, package provenance, and local name row componentwise. The carrier has no coordinate for a host interval tree, infinite refinement process, solver state, or completed real subset.

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionTasteGate single carrier alignment`

Theorem 366.2 (DyadicSubdivision NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicSubdivision}^{\uparrow}}$ *obligation surface has five rows:*

1. *carrier: the finite packet of Definition 366.1;*
2. *classifier: componentwise* hsame *transport of the parent interval, dyadic level, ordered cell list,* $\mathsf{DyadicMesh}^{\uparrow}$ *packets,* $\mathsf{ValidatedNumerics}^{\uparrow}$ *rows, provenance, and local name;*
3. *stability: appending displayed refinement routes to the mesh family preserves the transported parent interval and refines only the finite cell ledger;*
4. *exactness: every child boundary and containment read is one of the* $\mathsf{DyadicMesh}^{\uparrow}$ *or* $\mathsf{ValidatedNumerics}^{\uparrow}$ *rows indexed by the ordered list* C*;*
5. *ledger: the package and local certificate rows record exactly the dependency surface over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{ValidatedNumerics}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

These rows are the full seed naming surface for finite dyadic subdivision certificates.

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource namecert obligations`

Proof. Project the packet of Definition 366.1. The carrier row displays the parent interval, common level, finite ordered list, mesh-cell family, validated containment family, package provenance, and local certificate row.

The classifier is componentwise because the packet contains no hidden geometric object. Transporting a subdivision means transporting exactly those visible rows by hsame and replaying the stored Cont routes for the mesh and validated containment reads.

Stability and exactness are then read from the same finite ledger. Refinement appends routes to the mesh-family reads, while every boundary or enclosure claim remains indexed by C and therefore belongs to one of the displayed $\mathsf{DyadicMesh}^{\uparrow}$ or $\mathsf{ValidatedNumerics}^{\uparrow}$ rows. □

Theorem 366.3 (DyadicSubdivision refinement stability)**.** *Let* $S = (I, L, C, M, V, P, N)$ *be an accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet. If* S' *is obtained by replacing each mesh entry in* M *by a finite displayed* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement whose rational-interval boundary transports to the same parent row* I*, and by transporting the corresponding containment entries in* V *along the same routes, then* S' *is an accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet for the same parent interval.*

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionPacket refinement stability`

Proof. Start with the finite subdivision packet. The parent interval row I, package provenance P, and local name row N are retained, so the outer naming surface is unchanged.

For each entry of the ordered list C, use the $\mathsf{DyadicMesh}^{\uparrow}$ refinement route to replace the displayed cell by a finer finite family. The endpoint comparison is still through the transported $\mathsf{RationalInterval}^{\uparrow}$ boundary rows, so no host interval or completed real subset is introduced.

Transport the validated containment rows through the same cell routes. Repacking the parent row, refined level data, refined ordered cell ledger, refined mesh entries, transported containment entries, provenance, and local certificate gives exactly the carrier of Definition 366.1. □

Theorem 366.4 (DyadicSubdivision mesh coverage)**.** *For every accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet* $S = (I, L, C, M, V, P, N)$*, coverage of the parent interval is represented solely by the finite ordered cell list* C *together with the* $\mathsf{DyadicMesh}^{\uparrow}$ *packets in* M*. A consumer may read each child rational interval, dyadic endpoint ledger, and adjacency order through those rows, but it cannot read an infinite mesh, a host subset of* $\mathsf{Real}^{\uparrow}$*, or a global interval-tree object from* S*.*

Proof. Read the parent interval row I and the ordered list C. The list is the only row that arranges the children, and it is finite by construction of the packet.

Each child coverage object is then projected from the matching $\mathsf{DyadicMesh}^{\uparrow}$ entry in M. Its rational interval and dyadic endpoint ledger are already the finite coverage surface supplied by the mesh certificate.

Since the subdivision carrier contains no further coverage coordinate, the parent coverage surface is exhausted by C and M. Infinite meshes, host subsets, and interval-tree data are outside the packet. □

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource mesh coverage`

Theorem 366.5 (DyadicSubdivision validated enclosure handoff)**.** *A* $\mathsf{ValidatedNumerics}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet may read exactly the parent* $\mathsf{RationalInterval}^{\uparrow}$ *row, common dyadic level row, ordered finite cell ledger, child* $\mathsf{DyadicMesh}^{\uparrow}$ *packets, attached* $\mathsf{ValidatedNumerics}^{\uparrow}$ *containment rows,* hsame *transports,* Cont *routes, package provenance, and local* $\mathsf{NameCert}$ *row. The handoff supplies a finite enclosure subdivision and does not assert solver correctness, floating-point hardware semantics, classical real completeness, or an infinite refinement theorem.*

Proof. Project the carrier. The validated-numerics side first sees the parent interval, the level row, and the ordered cell list; these rows determine which finite child packets can be inspected.

For each child, read the matching $\mathsf{DyadicMesh}^{\uparrow}$ packet and the corresponding $\mathsf{ValidatedNumerics}^{\uparrow}$ containment row. The hsame transports and Cont routes ensure that the mesh boundary and containment ledger are read along the same finite cell route.

Thus the handoff is the finite enclosure subdivision formed by these rows. No coordinate supplies solver correctness, hardware semantics, classical completeness, or an infinite refinement principle. □

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource validated enclosure handoff`

Theorem 366.6 (DyadicSubdivision refinement composition)**.** *Let* $S = (I, L, C, M, V, P, N)$ *be an accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet. Suppose* S' *is obtained from* S *by displayed finite* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement routes over the same parent* $\mathsf{RationalInterval}^{\uparrow}$ *row* I*, and suppose* S'' *is obtained from* S' *by a second displayed finite family of* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement routes over the same parent row. Then* S'' *is an accepted* $\mathsf{DyadicSubdivision}^{\uparrow}$ *packet for* I*.*

The mesh coverage and validated enclosure handoff exported by S'' *are the composed finite ledger: first the refined child rows from* S *to* S'*, then the refined child rows from* S' *to* S''*, together with the transported* $\mathsf{ValidatedNumerics}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The composition asserts no interval-tree semantics, infinite refinement process, host real subset, or solver-correctness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionPacket refinement composition`

Proof. Apply refinement stability to the first displayed refinement family. The result is the accepted packet S' over the same parent interval row I, with only its finite cell ledger, mesh rows, and validated containment rows refined.

Apply refinement stability again to the second displayed refinement family. Since its routes are also over the same parent row, the second application yields an accepted packet S'' whose parent interval and provenance surface remain the displayed rows of the original packet.

It remains to identify the consumer surface. Mesh coverage reads only the finite ordered cell list and the $\mathsf{DyadicMesh}^{\uparrow}$ packets; validated enclosure handoff reads the matching $\mathsf{ValidatedNumerics}^{\uparrow}$

rows along the same routes. Chaining those two finite ledgers gives the composed mesh and enclosure surface named in the statement, with no additional infinite or host-level object. □

Theorem 366.7 (DyadicSubdivision common refinement exhaustion)**.** *Let $S_0 = (I, L_0, C_0, M_0, V_0, P_0, N_0)$ and $S_1 = (I, L_1, C_1, M_1, V_1, P_1, N_1)$ be two accepted* DyadicSubdivision$^{\uparrow}$ *packets over the same displayed* RationalInterval$^{\uparrow}$ *parent row I. If both packets carry finite* DyadicMesh$^{\uparrow}$ *refinement ledgers for the same parent interval, then there is a finite accepted child-ledger over* DyadicMesh$^{\uparrow}$ *rows that refines both C_0 and C_1 and whose validated enclosure rows are transported through the corresponding* ValidatedNumerics$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes.*

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionPacket common refinement exhaustion`

Proof. Read both packets through Definition 366.1. Each subdivision has a displayed parent interval row, a dyadic level row, a finite ordered cell list, a finite family of mesh cells, and a finite family of validated enclosure rows.

Form the common child-ledger by taking the displayed finite intersections of the two ordered dyadic cell ledgers inside the same parent row I. Because every cell boundary is read from DyadicMesh$^{\uparrow}$ rows over one rational interval parent, the resulting ledger is again finite and ordered.

Apply Definition 366.3 to transport the first packet to this common ledger, and apply it again to transport the second packet to the same ledger. The validated rows move along the same cell routes, so the common ledger exhausts both displayed refinement families without adding an infinite mesh or host interval tree. □

Theorem 366.8 (DyadicSubdivision cell ledger transport)**.** *For an accepted* DyadicSubdivision$^{\uparrow}$ *packet $S = (I, L, C, M, V, P, N)$, componentwise* hsame *and* Cont *transport of the parent interval I, the dyadic level row L, the ordered cell list C, the* DyadicMesh$^{\uparrow}$ *cell family M, the* ValidatedNumerics$^{\uparrow}$ *enclosure family V, the package row P, and the local* NameCert *row N preserves the subdivision classifier.*

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource cell ledger transport`

Proof. The classifier is read directly from Definition 366.1. It compares only the parent interval, common level, ordered cell list, mesh cells, validated enclosure rows, and provenance rows.

Transport the parent interval and level first. Then transport each ordered cell-list entry and its matching DyadicMesh$^{\uparrow}$ row along the displayed Cont route, preserving its position in the finite ledger.

Transport the ValidatedNumerics$^{\uparrow}$ rows along the same cell routes. Since each containment row remains attached to the transported mesh cell indexed by the transported ordered list, the packet still satisfies the carrier conditions.

Repack the transported rows with P and N. The resulting packet is classified with the original one, and no global interval tree, solver state, or completed real subset is introduced. □

Theorem 366.9 (DyadicSubdivision NameCert package)**.** *The* DyadicSubdivision$^{\uparrow}$ *obligation package is assembled from the source row of Definition 366.1, the NameCert obligations of Definition 366.2, refinement stability, mesh coverage, validated enclosure handoff, refinement composition, common refinement exhaustion, and cell-ledger transport. The package is finite over* DyadicRatCore$^{\uparrow}$*,* RationalInterval$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* ValidatedNumerics$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource namecert package`

Proof. The source row supplies the displayed carrier and dependency coordinates. The NameCert obligations record carrier, classifier, stability, exactness, and ledger rows for the finite subdivision packet.

Refinement stability is Definition 366.3; mesh coverage is Definition 366.4; validated enclosure handoff is Definition 366.5. Together these rows give finite coverage and consumer exactness for a single displayed subdivision ledger.

Refinement composition is Definition 366.6, common refinement exhaustion is Definition 366.7, and cell-ledger transport is Definition 366.8. Repacking these rows gives the stated obligation package while preserving the non-escape clauses for interval-tree semantics, solver correctness, hardware semantics, classical completeness, and infinite refinement. □

Theorem 366.10 (DyadicSubdivision routed bisection normal form)**.** *Let S be an accepted* DyadicSubdivision$^{\uparrow}$ *packet obtained by iterated unary bisection of a parent* RationalInterval$^{\uparrow}$ *row. Then S has a unique visible finite normal-form packet consisting exactly of the parent interval row, unary level row, ordered cell-list row,* DyadicMesh$^{\uparrow}$ *cell packets,* ValidatedNumerics$^{\uparrow}$ *containment rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

If two unary bisection routes over the same parent row agree on these displayed rows, then their resulting packets classify together as DyadicSubdivision$^{\uparrow}$ *packets. The normal-form packet exports no global interval tree, solver correctness, floating-point semantics, classical real completeness, or infinite refinement.*

Proof. Start with the initial parent interval row and read every unary bisection as a displayed finite DyadicMesh$^{\uparrow}$ refinement route. Apply Definition 366.3 after each bisection step, and use Definition 366.6 to collapse the resulting chain into one finite accepted subdivision packet over the original parent row.

The visible rows of the normal form are exhausted by the finite mesh and enclosure ledgers. Apply Definition 366.7 to compare two routes through their common finite child-ledger, then apply Definition 366.8 to transport the parent interval, level, ordered cells, mesh packets, containment rows, provenance, and local certificate componentwise.

Finally, Definition 366.9 repacks these transported rows as the public DyadicSubdivision$^{\uparrow}$ certificate. Because the package contains only the listed finite rows and routes, agreeing bisection ledgers classify together, and no global interval tree, solver-correctness theorem, hardware semantics, classical completeness principle, or infinite refinement object is exported. □

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource routed bisection normal form`

Theorem 366.11 (DyadicSubdivision scoped finite-route package)**.** *The* DyadicSubdivision$^{\uparrow}$ *source row, NameCert obligations, refinement stability, mesh coverage, validated enclosure handoff, refinement composition, common refinement exhaustion, cell-ledger transport, and routed bisection normal form bind exactly the finite dependency surface*

DyadicRatCore$^{\uparrow}$, RationalInterval$^{\uparrow}$, DyadicMesh$^{\uparrow}$, ValidatedNumerics$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Every consumer read of an accepted subdivision packet factors through those displayed rows and routes. The scoped package does not expose a host interval tree, a solver-correctness theorem, floating-point hardware semantics, classical real completeness, or an infinite refinement object.

Proof. Start with the source row of Definition 366.1. It lists the parent rational interval, unary level, ordered cell ledger, mesh-cell family, validated enclosure family, package provenance, and local name row as the only carrier coordinates.

The NameCert obligations of Definition 366.2 fix the carrier, classifier, stability, exactness, and ledger rows over the named dependencies. Refinement stability, mesh coverage, and validated enclosure handoff then show how a consumer reads cells and enclosures through the same finite ledger.

Refinement composition, common refinement exhaustion, and cell-ledger transport account for route chaining and comparison between displayed bisection ledgers. The routed bisection normal form of Definition 366.10 repacks those chains into one visible finite packet.

Thus every consumer access factors through the displayed rows in the statement. Since no cited row contains an interval tree, solver state, hardware semantics, completeness principle, or infinite refinement coordinate, the scoped package has exactly the stated finite dependency surface. □

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource scoped finite route package`

Theorem 366.12 (DyadicSubdivision public finite-enclosure export)**.** *The source row, scoped finite-route package, routed bisection normal form, mesh coverage, and validated enclosure handoff assemble one public* DyadicSubdivision$^{\uparrow}$ *finite-enclosure certificate consumed by* ValidatedNumerics$^{\uparrow}$ *and* Real$^{\uparrow}$ *readers. Every read factors through the parent* RationalInterval$^{\uparrow}$ *row, unary dyadic level, ordered cell list,* DyadicMesh$^{\uparrow}$ *packets,* ValidatedNumerics$^{\uparrow}$ *containment rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The public export contains no interval tree, solver state, hardware semantics, classical real completeness, or infinite refinement object.*

Proof. Begin with the source row of Definition 366.1. It supplies the parent rational interval, level, finite cell ledger, mesh rows, containment rows, package provenance, and local name row.

Use the scoped finite-route package of Definition 366.11 together with the routed bisection normal form of Definition 366.10. These identify every accepted bisection route with one visible finite packet over the same rows.

Apply mesh coverage from Definition 366.4 and the validated enclosure handoff of Definition 366.5. They are the consumer-facing rows read by ValidatedNumerics$^{\uparrow}$ and Real$^{\uparrow}$.

Repack those rows as the public finite-enclosure surface. Since each cited row is finite and none contains an interval tree, solver state, hardware semantics, classical completeness principle, or infinite refinement coordinate, the exported certificate has exactly the stated boundary. □

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource public finite enclosure export`

Theorem 366.13 (DyadicSubdivision standard bridge certificate)**.** *The public finite-enclosure certificate of Definition 366.12 supplies a standard finite subdivision reading for the same parent interval. Its bridge fields are the parent* RationalInterval$^{\uparrow}$ *row, unary dyadic level, ordered child-cell ledger,* DyadicMesh$^{\uparrow}$ *packets,* ValidatedNumerics$^{\uparrow}$ *containment rows,* Real$^{\uparrow}$ *consumer handoff,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Every standard read of the finite subdivision factors through those displayed fields. The bridge exports no interval tree, solver-correctness theorem, floating-point hardware semantics, classical real completeness principle, or infinite refinement object.

Lean4 ✓ `BEDC.Derived.DyadicSubdivisionUp.DyadicSubdivisionSource standard bridge certificate`

Proof. Begin with the public finite-enclosure export. It already names the parent rational interval, common dyadic level, ordered cell list, mesh packets, validated containment rows, and consumer provenance as one public DyadicSubdivision$^{\uparrow}$ certificate.

Read this packet as a standard finite subdivision by projecting the same rows. The parent interval is the source field; the ordered child-cell ledger and $\mathsf{DyadicMesh}^{\uparrow}$ packets give the finite subdivision field; the $\mathsf{ValidatedNumerics}^{\uparrow}$ rows give the enclosure field; and the $\mathsf{Real}^{\uparrow}$ handoff is reached only through the displayed consumer routes.

The transport and provenance fields are exactly BHist, hsame, Cont, Pkg, and NameCert. Since the bridge has no other field, a standard reader cannot obtain an interval tree, solver-correctness theorem, hardware semantics, classical completeness principle, or infinite refinement object from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSubdivision}^{\uparrow}$ packet is generated from a parent $\mathsf{RationalInterval}^{\uparrow}$ row, a unary dyadic level row, a finite ordered cell-list row, $\mathsf{DyadicMesh}^{\uparrow}$ cell packets, $\mathsf{ValidatedNumerics}^{\uparrow}$ containment rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSubdivision}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 366.12 and Definition 366.13 export the finite $\mathsf{DyadicSubdivision}^{\uparrow}$ enclosure certificate and its standard finite subdivision reading for $\mathsf{ValidatedNumerics}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers through the parent rational interval, unary level, ordered cell list, mesh packets, containment rows, BHist, hsame, Cont, Pkg, and NameCert provenance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close global interval-tree semantics, solver correctness, floating-point hardware behavior, classical real completeness, infinite refinement, or bridge checking.

Upgrade path: Lean formalization and bridge checking are the remaining verification-axis upgrades for the displayed finite subdivision packet.

Chapter 367

A Concrete Naming Certificate for $\mathsf{FastConvergentSeries}^{\uparrow}$

The $\mathsf{FastConvergentSeries}^{\uparrow}$ interface names the finite certificate surface for a constructive series whose partial sums come with an explicit fast tail schedule. Following Bishop and Bridges, *Constructive Analysis*, where convergence of series is carried by effective Cauchy information, the packet records only displayed summand observations, partial-sum readbacks, tail bounds, real-seal rows, transport routes, and provenance. It does not form a quotient of streams, select an infinite sum, or import host completeness for real numbers.

Its fast tail schedule and finite comparison ledger consume the displayed dyadic tail-bound surface of chapter 379 before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal. Thus the tail-budget carrier is read through a concrete dyadic bound rather than an ambient convergence predicate.

The fast-tail packet is a consumer of the regular rational readback and real-seal surfaces of chapter 301 and chapter 38. It does not create a real endpoint: partial-sum rows Q, the tail schedule μ, and the tail-bound ledger T are routed into the existing $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and then into the $\mathsf{Real}^{\uparrow}$ seal row E.

Definition 367.1 (FastConvergentSeries BHist carrier)**.** *A* $\mathsf{FastConvergentSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, S, Q, \mu, T, R, E, L, C, P, N)$$

with the following visible rows:

1. *A is the* $\mathsf{Series}^{\uparrow}$ *source row naming the summand stream and its finite partial-sum ledger;*
2. *S is the* $\mathsf{Seq}^{\uparrow}$ *observation surface through which displayed summand rows and prefix rows are read;*
3. *Q is the finite family of partial-sum observations displayed at requested unary indices;*
4. *μ is a* $\mathsf{CauchyModulus}^{\uparrow}$ *fast tail schedule assigning each precision row a finite threshold after which tails are bounded by the requested tolerance;*
5. *T is the tail-bound ledger comparing displayed partial sums past μ and recording the resulting rational tolerance rows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *partial-sum readback surface obtained from Q, μ, and T;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing ledger for the real name represented by the partial-sum readbacks;*

8. *L records the componentwise* hsame *transports for summand observations, partial sums, tail bounds, schedules, and seal-facing rows;*
9. *C records the displayed* Cont *read routes through which finite consumers reach the summand, partial-sum, tail-bound, readback, and seal rows;*
10. *P is the* Pkg *provenance over* $\mathsf{Series}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the rational source rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ *provenance row.*

The carrier is a finite fast-convergence series packet. It contains no quotient stream, no chosen infinite sum, and no ambient completeness proof.

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesTasteGate single carrier alignment`

Definition 367.2 (FastConvergentSeries classifier)**.** *For accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packets F and F', the classifier* $\mathsf{FastConvergentSeriesClassifier}(F, F')$ *compares only the displayed partial-sum rows, tail-bound ledgers, fast tail schedules,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal-facing rows, componentwise* hsame *transports,* Cont *read routes,* Pkg *provenance, and local* NameCert *rows.*

It does not compare hidden summability predicates, host real endpoints, infinite stream quotients, or choice-selected sums. Any accepted transport must therefore preserve the finite rows listed in Definition 367.1.

Lean4 (definition) `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesClassifierSpec`

Theorem 367.3 (FastConvergentSeries constant zero seed)**.** *The stationary zero summand stream determines an accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *carrier. Its* $\mathsf{Series}^{\uparrow}$ *and* $\mathsf{Seq}^{\uparrow}$ *rows display only zero summand observations, every displayed partial sum is the zero row, the fast tail schedule may use the zero threshold for every precision request, the tail-bound ledger records zero tails, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is the constant zero regular sequence, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing row is the constant zero real name.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier constant zero seed`

Proof. Start with the zero summand row in the source carrier and read it through the $\mathsf{Seq}^{\uparrow}$ observation surface. Every finite prefix has the same zero partial-sum row, so the $\mathsf{Series}^{\uparrow}$ partial-sum ledger is inhabited by the displayed zero rows.

Choose the fast tail schedule that assigns the zero threshold to every precision request. Since every later partial sum is again zero, the tail-bound ledger compares identical displayed rows and records the requested tolerance as satisfied.

Repack the same zero rows as the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal-facing ledger. The hsame transports and Cont routes are componentwise identities on the displayed finite rows, and the package provenance names only the source certificates and the local NameCert row. □

Theorem 367.4 (FastConvergentSeries tail-sum boundary)**.** *For every accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet and every displayed precision request, any* $\mathsf{Real}^{\uparrow}$ *consumer read of the represented sum factors through one fast tail schedule row, the displayed partial-sum rows past that threshold, the tail-bound ledger comparing those rows, the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *partial-sum readback, and the seal-facing* $\mathsf{Real}^{\uparrow}$ *row. The consumer receives no coordinate for a host infinite sum, quotient equality of streams, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier tail sum boundary`

Proof. Project the carrier coordinates of Definition 367.1. A precision request first reads the schedule row μ, which supplies the finite threshold used by the tail-bound ledger.

The consumer then reaches the displayed partial sums through the $\mathsf{Seq}^{\uparrow}$ and $\mathsf{Series}^{\uparrow}$ rows. The tail-bound ledger compares only those finite partial-sum rows past the threshold and records the rational tolerance evidence visible at that precision.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff reads the same partial-sum ledger as a regular rational sequence, and the $\mathsf{Real}^{\uparrow}$ seal-facing row is attached to that handoff. Since the carrier has no coordinate for a host sum, quotient stream, or completeness proof, none can be exported by the read. □

Theorem 367.5 (FastConvergentSeries regular rational tail-bound handoff)**.** *For an accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet, the displayed partial-sum rows, fast tail schedule, and finite tail-bound ledger determine the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row used by the real-seal ledger. Classifier transport of the schedule rows and tail-bound rows transports that handoff row componentwise.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesTailBoundPacket regseqrat handoff`

Proof. Project the carrier rows Q, μ, T, R, and E from Definition 367.1. The partial-sum rows Q supply the displayed rational approximants, the schedule μ supplies the finite threshold for each requested precision, and the tail-bound ledger T records the comparison rows past that threshold.

Use Definition 367.4 to factor every real-facing read through the same schedule and finite comparison ledger. Thus the regular rational readback row R is the repacking of Q, μ, and T that the seal-facing row E is allowed to consume.

Under classifier transport, the schedule rows and tail-bound rows move by the stored componentwise hsame comparisons. Repack the transported finite rows as the transported $\mathsf{RegSeqRat}^{\uparrow}$ handoff row. No host infinite sum, quotient stream, or ambient completeness object is introduced in this repacking. □

Theorem 367.6 (FastConvergentSeries NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier inhabitation: Definition 367.3 supplies a concrete accepted packet for the stationary zero series;*
2. *classifier transport: Definition 367.2 transports displayed partial sums, tail-bound rows, schedules, readbacks, seal-facing rows,* hsame *routes,* Cont *routes, package provenance, and local provenance componentwise;*
3. *tail-sum boundary: Definition 367.4 forces every real-facing read through the displayed fast tail schedule and finite partial-sum comparison ledger;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: the partial-sum rows, schedule rows, and tail-bound ledger repack as the visible regular rational readback row R of Definition 367.1;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape: the seal-facing ledger E may read only the* $\mathsf{Series}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows displayed by the packet.*

These rows are exactly the finite naming-certificate surface for fast convergent series.

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier namecert obligation surface`

Proof. The first row is the explicit zero-series inhabitant. It provides summand observations, partial sums, tail bounds, regular readbacks, and the real seal without importing any completion object.

The classifier row is componentwise by definition. Each permitted transport is one of the displayed hsame comparisons or Cont read routes, so transported packets preserve the same partial-sum, tail-bound, schedule, readback, and seal-facing coordinates.

The boundary row is Definition 367.4. It shows that every real-facing read must pass through the finite schedule and partial-sum ledger before reaching the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

The final two rows are projections of the same carrier. The regular readback forgets no tail evidence needed by $\mathsf{RegSeqRat}^{\uparrow}$, and the real seal has no field from which a quotient stream, selected sum, or ambient completeness principle could be read. □

Theorem 367.7 (FastConvergentSeries scoped kernel factorization)**.** *For every accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet, the consumer route to a real-seal read is exhausted by the displayed* $\mathsf{Series}^{\uparrow}$ *source row,* $\mathsf{Seq}^{\uparrow}$ *observation surface,* $\mathsf{CauchyModulus}^{\uparrow}$ *fast tail schedule, finite tail-bound ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *partial-sum handoff,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *read routes,* Pkg *provenance, and local* NameCert *row.*

No real-seal consumer can factor the read through an ambient summability predicate, a quotient of summand streams, a selected host sum, or a completeness theorem outside those rows.

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier scoped kernel factorization`

Proof. First read the tail boundary. By Definition 367.4, every precision request reaches the fast tail schedule, the partial-sum rows past the displayed threshold, and the finite comparison ledger before it reaches any real-facing row.

Next identify the regular rational handoff. By Definition 367.5, the same partial-sum rows, schedule rows, and tail-bound ledger determine the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by the seal-facing ledger.

Finally bind the local certificate surface. By Definition 367.6, the only remaining routes are the displayed hsame transports, Cont reads, package provenance, and local NameCert row attached to the same carrier.

Combining these three projections exhausts the kernel route for the real-seal read. Since none of the cited rows contains an ambient summability predicate, quotient stream, selected sum, or completeness theorem, no such object can be exported by the scoped packet. □

Theorem 367.8 (FastConvergentSeries public tail-budget export)**.** *The constant-zero seed, tail-sum boundary,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-bound handoff, local* $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ *obligation surface, and scoped kernel factorization export a public finite tail-budget handoff for* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *consumers.*

The exported handoff is the finite-history route through the displayed $\mathsf{Series}^{\uparrow}$ *source row,* $\mathsf{Seq}^{\uparrow}$ *observation surface, partial-sum rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *fast tail schedule, finite tail-bound ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *reads,* Pkg *provenance, and local* NameCert *row. It asserts no convergence theorem beyond the displayed tail budget and supplies no quotient stream, selected infinite sum, host summability predicate, or ambient completeness object.*

Proof. The inhabitance row is Definition 367.3. It gives a concrete finite carrier shape whose summand observations, partial sums, tail ledger, regular readback, and real seal are all displayed rows.

The consumer boundary is Definition 367.4. Every real-facing read first passes through the fast tail schedule and the finite partial-sum comparison ledger for the requested precision.

The regular-rational handoff is Definition 367.5. It repacks the same partial-sum rows, schedule rows, and tail-bound ledger as the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by the real seal.

The local certificate rows are collected by Definition 367.6, and the full route is exhausted by Definition 367.7. Combining these results gives the public tail-budget handoff while keeping the conclusion inside the finite BHist packet.

Since each cited row is finite and displayed, the export contains only the named tail-budget surface. It does not assert a stronger convergence theorem or introduce any host summability, quotient, selected sum, or completeness datum. □

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier public tail budget export`

Theorem 367.9 (FastConvergentSeries public export)**.** *The carrier of Definition 367.1, the classifier of Definition 367.2, the constant-zero seed, the tail-sum boundary, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the local* $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ *obligation surface, and the scoped kernel factorization jointly export the public* $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ *handoff over the finite route through* $\mathsf{Series}^{\uparrow}$*,* $\mathsf{Seq}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Every public consumer of the fast-series certificate reads only displayed summand rows, partial-sum rows, tail-budget rows, regular rational readbacks, seal-facing rows, transport routes, and provenance. The export contains no quotient summand stream, selected infinite sum, host summability predicate, ambient completeness theorem, or bridge row.

Proof. First build the public carrier surface. The carrier definition lists the finite packet rows, the classifier definition restricts comparison to displayed rows, and Definition 367.3 supplies the concrete zero-series inhabitant.

Next identify the consumer route. By Definition 367.4, every real-facing precision request factors through the fast tail schedule and finite tail-bound ledger. By Definition 367.5, the same partial-sum and tail-budget rows determine the $\mathsf{RegSeqRat}^{\uparrow}$ handoff consumed by the seal-facing row.

Finally bind the certificate rows. The local obligation surface in Definition 367.6 supplies the $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$ rows, and Definition 367.7 exhausts the kernel route. The public tail-budget export of Definition 367.8 is therefore the same finite route presented as a public consumer object, with no hidden summability, quotient, selected sum, completeness, or bridge coordinate. □

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier public export`

Theorem 367.10 (FastConvergentSeries bridge tail-budget schedule)**.** *For every accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet and every displayed precision request, the public tail-budget export supplies a bridge-facing schedule row, a finite partial-sum ledger, and a tail-bound ledger. The schedule row is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold read for the request, the partial-sum ledger is the displayed* $\mathsf{Series}^{\uparrow}$ *and* $\mathsf{Seq}^{\uparrow}$ *prefix surface past that threshold, and the tail-bound ledger is the finite comparison surface consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*

The exported schedule is finite and displayed at the requested precision. It contains no quotient of summand streams, no host infinite-sum object, and no ambient completeness premise.

Proof. Project the carrier of Definition 367.1. The precision request reads the row μ, and that row supplies the displayed threshold used by the tail-budget ledger.

Apply Definition 367.4 to force the consumer route through the partial-sum rows past the threshold. Apply Definition 367.5 to identify the same rows as the finite comparison ledger used by the $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Finally use Definition 367.9. The public export names exactly those schedule, partial-sum, and tail-bound rows, so the bridge-facing schedule is a displayed finite ledger rather than an external convergence object. □

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier bridge tail budget schedule`

Theorem 367.11 (FastConvergentSeries bridge real-seal consumer)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the displayed tail-budget schedule of Definition 367.10. The seal consumer receives the* $\mathsf{Series}^{\uparrow}$ *source row,* $\mathsf{Seq}^{\uparrow}$ *observation surface,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, finite tail-bound ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row before it reaches the seal.*

No seal read factors through quotient stream equality, a selected host sum, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier bridge real seal consumer`

Proof. Start from the schedule handoff in Definition 367.10. It supplies the displayed threshold, the partial-sum ledger, and the finite tail-bound ledger for the requested precision.

Use Definition 367.5 to repack those rows as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff. Then apply Definition 367.7 to place the $\mathsf{Real}^{\uparrow}$ seal after that handoff and after the displayed hsame, Cont, Pkg, and NameCert rows.

The carrier has no coordinate for a quotient stream, selected host sum, or completeness theorem. The seal read is therefore exhausted by the finite route named in the statement. □

Theorem 367.12 (FastConvergentSeries bridge completion handoff)**.** *The public export of Definition 367.9, together with the bridge tail-budget schedule of Definition 367.10 and the real-seal consumer factorization of Definition 367.11, assembles the bridge-facing completion handoff for* $\mathsf{FastConvergentSeries}^{\uparrow}$*.*

The handoff offered to completion consumers is the finite route through $\mathsf{Series}^{\uparrow}$*,* $\mathsf{Seq}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*, displayed partial sums, finite tail bounds,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It explicitly excludes quotient streams, host sums, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier bridge completion handoff`
Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesTailBoundPacket bridge completion handoff`

Proof. First take the public export from Definition 367.9. It fixes the consumer-visible surface over the series source, sequence observation row, fast tail schedule, regular rational readback, real seal, transports, read routes, provenance, and local certificate.

Next attach the finite schedule. The bridge tail-budget schedule theorem supplies the requested precision row, the partial-sum ledger, and the tail-bound ledger. The tail-sum boundary and regular rational handoff theorems identify those rows as the only data used by the precision read.

Finally attach the seal. The real-seal consumer theorem and the scoped kernel factorization place the $\mathsf{Real}^{\uparrow}$ seal downstream of the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the displayed certificate rows.

Repacking these rows gives the completion-facing handoff while leaving no coordinate for quotient streams, host sums, or ambient completeness. □

Theorem 367.13 (FastConvergentSeries bridge boundary package)**.** *The public export of Definition 367.9 factors into the standard bridge-facing boundary consisting of the* Series$^{\uparrow}$ *source row,* Seq$^{\uparrow}$ *observation surface,* CauchyModulus$^{\uparrow}$ *fast tail schedule, finite tail-bound ledger,* RegSeqRat$^{\uparrow}$ *partial-sum handoff,* Real$^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *read routes,* Pkg *provenance, and local* NameCert *row.*

This boundary is the finite tail-budget surface consumed by downstream Real$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *readers. It introduces no quotient stream, selected infinite sum, host summability predicate, or ambient completeness theorem.*

Proof. First take the public handoff from Definition 367.9. Its statement already names the finite route through the series source, sequence observation surface, fast modulus, regular rational handoff, real seal, transports, read routes, provenance, and local certificate.

Next isolate the tail-budget component. By Definition 367.8, the public object exported to consumers is exactly the finite-history route through the displayed partial-sum rows, tail schedule, tail-bound ledger, RegSeqRat$^{\uparrow}$ readback, and Real$^{\uparrow}$ seal row.

The scoped kernel factorization of Definition 367.7 accounts for the remaining hsame, Cont, Pkg, and NameCert rows, while Definition 367.4 supplies the checked tail-sum boundary used by the precision read. Repacking these rows gives the bridge-facing boundary named in the theorem.

Because every cited row is a displayed finite carrier coordinate or a displayed transport route, the package has no coordinate for quotient streams, selected infinite sums, host summability predicates, or ambient completeness theorems. □

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier bridge boundary package`

Theorem 367.14 (FastConvergentSeries finite tail-budget example)**.** *The stationary zero summand series carries a mature finite tail-budget example for* FastConvergentSeries$^{\uparrow}$*. Its* Series$^{\uparrow}$ *source row and* Seq$^{\uparrow}$ *observation surface display zero summands and zero partial sums, its* CauchyModulus$^{\uparrow}$ *schedule uses the zero threshold for every precision request, its finite tail-bound ledger records zero tails, and its* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *rows read the resulting constant zero name.*

The example is entirely a finite carrier packet over Series$^{\uparrow}$*,* Seq$^{\uparrow}$*,* CauchyModulus$^{\uparrow}$*, finite tail bounds,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It does not select a host sum or appeal to ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesTailBoundPacket constant zero seed`

Proof. Begin with Definition 367.3. It supplies the zero summand observations, zero partial sums, zero tail ledger, regular-rational readback, and real seal as displayed carrier rows.

Apply Definition 367.10. For each precision request, the displayed schedule may choose the zero threshold, and the later partial-sum comparisons are all zero comparisons inside the finite tail-bound ledger.

Finally use Definition 367.13. The example is read only through the bridge-facing finite rows, so the stationary zero series supplies a concrete tail-budget package without a selected host infinite sum. □

Theorem 367.15 (FastConvergentSeries Cauchy consumer coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exported by the* $\mathsf{FastConvergentSeries}^{\uparrow}$ *bridge boundary is covered by the rows* $\mathsf{Series}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, *finite tail bounds,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert.

The consumer can inspect summand observations, displayed partial sums, the finite threshold schedule, the tail-bound ledger, the regular-rational readback, the real seal, transport rows, continuation routes, and provenance. It cannot inspect quotient streams, selected host sums, hidden summability predicates, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier cauchy consumer coverage`

Proof. First take the bridge boundary package from Definition 367.13. It names exactly the finite route through the source series, sequence observations, fast schedule, tail-bound ledger, regular readback, real seal, transports, continuations, provenance, and local certificate.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, apply Definition 367.5. The consumer receives the partial-sum rows, schedule rows, and finite comparison ledger repacked as the regular rational handoff.

For a $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 367.11. The seal read occurs only after the same tail-budget schedule and $\mathsf{RegSeqRat}^{\uparrow}$ handoff have been read.

The scoped kernel factorization then accounts for hsame, Cont, Pkg, and NameCert. Since none of these rows carries quotient equality, selected sums, summability predicates, or completeness principles, the coverage boundary is exactly the listed finite surface. □

Theorem 367.16 (FastConvergentSeries mature boundary package)**.** *The finite tail-budget example of Definition 367.14, the Cauchy consumer coverage of Definition 367.15, and the bridge boundary package of Definition 367.13 assemble the mature finite-tail-budget boundary for* $\mathsf{FastConvergentSeries}^{\uparrow}$.

The package provides a concrete downstream example and covers the completion-facing $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer routes while excluding quotient streams, selected host sums, hidden summability predicates, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastConvergentSeriesUp.FastConvergentSeriesCarrier mature boundary package`

Proof. The example theorem supplies an inhabited finite tail-budget packet: the stationary zero series, zero partial sums, zero threshold schedule, zero tail ledger, regular readback, and real seal.

The consumer coverage theorem supplies the downstream read boundary. It shows that both $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers inspect only the source, observation, schedule, tail-bound, readback, seal, transport, continuation, package, and local naming rows.

The bridge boundary package identifies the same rows as the bridge-facing public surface. Combining these three facts gives the mature boundary stated here, with every completion-facing read kept inside the finite packet and with no coordinate for quotient streams, host sums, summability predicates, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{FastConvergentSeries}^{\uparrow}$ packet is generated from displayed $\mathsf{Series}^{\uparrow}$ summand and partial-sum rows, the underlying $\mathsf{Seq}^{\uparrow}$ observation surface, a $\mathsf{CauchyModulus}^{\uparrow}$ fast tail schedule, finite tail-bound rows, $\mathsf{RegSeqRat}^{\uparrow}$ partial-sum readbacks, $\mathsf{Real}^{\uparrow}$ seal-facing rows, componentwise hsame transports, displayed Cont read routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The chapter fixes the constant-zero seed, tail-sum boundary, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, NameCert obligation surface, scoped kernel factorization, public tail-budget export, bridge schedule, bridge real-seal consumer, bridge completion handoff, and bridge boundary package for the fast convergent series certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastConvergentSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a public $\mathsf{Real}^{\uparrow}$ completeness theorem, quotient equality of streams, uniqueness of sums, host summability, or checked bridge schema.

Upgrade path: No further paper-axis closure grade remains; formal verification and bridge checking remain on the independent verification axis.

Chapter 368

A Concrete Naming Certificate for $\mathsf{FastSeriesProduct}^{\uparrow}$

The $\mathsf{FastSeriesProduct}^{\uparrow}$ interface names the finite certificate surface for the Cauchy product of two already accepted $\mathsf{FastConvergentSeries}^{\uparrow}$ packets. Following Bishop and Bridges, *Constructive Analysis*, where products of absolutely or effectively convergent series are controlled by explicit tail data and finite convolution estimates, the packet records only finite summand windows, finite convolution rows, pasted tail schedules, rational regular-sequence readbacks, real-seal rows, transport routes, package provenance, and local naming provenance. It does not form quotient streams, choose a host infinite sum, or appeal to ambient completeness of real numbers.

Definition 368.1 (FastSeriesProduct carrier)**.** *A* $\mathsf{FastSeriesProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (F, G, K, W, M, T, R, E, L, C, P, N)$$

with the following visible rows:

1. *F and G are accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *source packets, carrying displayed summand rows, partial-sum rows, fast tail schedules,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal-facing rows,* hsame *transports,* Cont *read routes, package provenance, and local certificate rows;*
2. *K is the finite convolution row whose inspected output coefficient at a displayed index n is the finite sum of displayed source summands $a_i b_{n-i}$ over the inspected range $0 \leq i \leq n$;*
3. *W is the finite partial-sum window ledger for the convolution rows, recording only displayed source windows and displayed product-prefix sums;*
4. *M is the pasted tail schedule, obtained by reading the fast tail schedules of F and G together with the finite source-window bounds needed for the bilinear tail estimate;*
5. *T is the tail-bound convolution ledger comparing displayed product-prefix sums past the threshold named by M;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the displayed product partial sums and their pasted tail schedule;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the same regular rational handoff;*
8. *L records the componentwise* hsame *transports for the two source packets, convolution rows, pasted schedule rows, tail-bound rows, regular handoff rows, and seal-facing rows;*

9. *C records the displayed* Cont *read routes by which consumers reach the source series, convolution window, pasted schedule, tail ledger, regular handoff, and real seal;*
10. *P is the* Pkg *provenance over* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the finite rational arithmetic rows used by the convolution, and the source certificate rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$ *provenance row.*

The carrier is a finite product certificate. It contains no quotient of streams, no host-level series multiplication, and no completeness theorem for real numbers.

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductTasteGate single carrier alignment`

The packet is not a standalone series object: its source rows F and G are accepted $\mathsf{FastConvergentSeries}^{\uparrow}$ packets from chapter 367, and all convolution, pasted schedule, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal rows consume that sibling's tail-budget surface.

Definition 368.2 (FastSeriesProduct classifier)**.** *For accepted* $\mathsf{FastSeriesProduct}^{\uparrow}$ *packets H and H', the classifier* $\mathsf{FastSeriesProductClassifier}(H, H')$ *compares only the two accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *source packets, their displayed finite convolution rows, product partial-sum windows, pasted tail schedules, tail-bound convolution ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *provenance.*

In particular, classifier transport is a finite-row comparison. It does not compare host sums, quotient representatives of streams, unbounded convolution functions, or a classical completeness proof for the product series.

Theorem 368.3 (FastSeriesProduct tail-bound convolution)**.** *Let F and G be accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packets displayed inside a* $\mathsf{FastSeriesProduct}^{\uparrow}$ *carrier. For every displayed product precision request, the pasted schedule row M supplies a finite threshold after which any two displayed product partial sums are compared by a finite convolution estimate using only:*

1. *the source tail schedules of F and G;*
2. *the finite source-window bounds recorded beside those schedules;*
3. *the displayed convolution rows K;*
4. *the product-prefix ledger W.*

The resulting tail-bound row in T is therefore a finite BEDC witness for the Cauchy-product tail at that precision, with no appeal to an infinite summation object.

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductCarrier tail bound convolution`

Proof. Read the two source packets through Definition 368.1. Each source supplies a displayed fast tail schedule and the finite windows on which the requested bounds are inspected.

The convolution row K computes each inspected product coefficient from finitely many displayed source summands. The product-prefix ledger W therefore compares two late product prefixes by splitting the difference into the finite head-window contribution and the two source-tail contributions named by the pasted schedule.

The head-window contribution is a finite rational arithmetic row, while the two tail contributions are bounded by the source schedules of F and G together with the displayed source-window bounds. Repacking these rows gives the tail-bound convolution ledger T.

Since every row used in the estimate is a coordinate of the carrier, the argument never calls a host Cauchy-product theorem, a quotient of streams, or an ambient completeness principle. □

Theorem 368.4 (FastSeriesProduct RegSeqRat handoff)**.** *For every accepted* $\mathsf{FastSeriesProduct}^{\uparrow}$ *packet and every displayed precision request, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is obtained by repacking the finite convolution partial sums in* W *together with the pasted schedule* M *and the tail-bound convolution ledger* T*. A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer can read the product name only through those three displayed rows and the recorded* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductCarrier regseqrat handoff`

Proof. Project the carrier

$$H = (F, G, K, W, M, T, R, E, L, C, P, N).$$

The partial-sum data offered to $\mathsf{RegSeqRat}^{\uparrow}$ is exactly the product-prefix ledger W, whose entries are built from finite convolution rows in K.

The regularity witness is the pasted schedule M together with the tail-bound ledger T. By Definition 368.3, those rows already contain the finite estimate needed for the requested precision.

The transport and continuation coordinates L and C identify how a consumer reaches the source windows, convolution windows, schedule rows, and tail ledger. Hence the handoff row R exports a regular rational sequence surface, not a hidden product series or a quotient stream. □

Theorem 368.5 (FastSeriesProduct NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$ *obligation surface consists of six rows:*

1. *carrier admission: an accepted packet is the finite* BHist *carrier of Definition 368.1;*
2. *classifier transport: Definition 368.2 compares only source packets, finite convolution rows, product-prefix windows, pasted schedule rows, tail ledgers, regular handoffs, real seals, transports, continuations, and provenance;*
3. *finite convolution exactness: each displayed product coefficient is read as the finite row* $\sum_{0 \le i \le n} a_i b_{n-i}$ *over inspected source summands;*
4. *tail-bound convolution: Definition 368.3 supplies the finite pasted tail estimate for product prefixes;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff: Definition 368.4 repacks the product partial sums, pasted schedule, and tail-bound ledger as the visible regular rational readback row;*
6. $\mathsf{Real}^{\uparrow}$ *seal compatibility: the seal row* E *may read only the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the finite source, convolution, schedule, tail, transport, continuation, package, and local naming rows carried by the packet.*

These rows are the concrete naming-certificate surface for Cauchy products of fast convergent series over the finite $\mathsf{Real}^{\uparrow}$ *seal path.*

Proof. Carrier admission and classifier transport are projections of the two definitions. They list exactly the rows that an accepted product packet stores and the rows that an accepted transport may compare.

Finite convolution exactness is built into the row K: each inspected output coefficient is a finite sum over displayed source summands, so no unbounded convolution function is exported to a consumer.

The tail-bound row is Definition 368.3. It explains how the pasted source schedules and finite source-window bounds control late product-prefix comparisons.

The regular handoff row is Definition 368.4. The $\mathsf{Real}^{\uparrow}$ seal row then reads only that handoff together with the visible transport, continuation, package, and local provenance rows.

Thus the obligation surface closes around the finite certificate packet. It leaves no coordinate for host sums, stream quotients, selected limits, or ambient real completeness. □

Theorem 368.6 (FastSeriesProduct classifier transport exactness)**.** *Let*

$$H = (F, G, K, W, M, T, R, E, L, C, P, N)$$

be an accepted $\mathsf{FastSeriesProduct}^{\uparrow}$ *carrier. If a displayed packet* H' *is obtained by transporting the two source packets, finite convolution row, product-window ledger, pasted schedule, tail-bound ledger, regular handoff, and real-seal row along the componentwise* hsame *data in* L *and the displayed* Cont *routes in* C*, while preserving the* Pkg *row* P *and local* $\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$ *row* N*, then the classifier accepts* H' *exactly as the transported product carrier. It cannot compare host products, quotient streams, unbounded convolution functions, or ambient completeness evidence.*

Proof. Project the accepted carrier through Definition 368.1. The classifier of Definition 368.2 reads precisely the two source packets, K, W, M, T, R, E, and the transport, continuation, package, and local certificate rows.

Transport each displayed coordinate using the corresponding hsame witness in L, and read the transported row only through the continuation routes recorded in C. The package and local certificate rows keep the transported packet inside the same finite naming surface.

Because the classifier definition has no projection for a host product, stream quotient, unbounded convolution object, or completeness theorem, acceptance is preserved exactly by those componentwise transported carrier rows. □

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductCarrier classifier transport exactness`

Theorem 368.7 (FastSeriesProduct convolution row exactness)**.** *For every accepted* $\mathsf{FastSeriesProduct}^{\uparrow}$ *carrier*

$$H = (F, G, K, W, M, T, R, E, L, C, P, N),$$

the convolution row K *supplies each inspected product coefficient as the finite displayed row* $\sum_{0 \le i \le n} a_i b_{n-i}$ *over the source summands read from* F *and* G*. The product-window ledger* W *and pasted tail ledger* T *may consume* K *only through those finite coefficient rows, together with the schedule row* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the provenance rows* P *and* N*.*

Proof. Read F and G as accepted $\mathsf{FastConvergentSeries}^{\uparrow}$ source packets. Their summand observations are already displayed finite rows, so the inspected coefficient at a fixed n is obtained by the finite rational sum named in the carrier.

The product-window ledger W records prefix sums built from those inspected coefficients. The pasted schedule M and tail-bound ledger T then use Definition 368.3 to compare late product prefixes through finite source windows and source tail schedules.

Finally Definition 368.4 repacks W, M, and T as the regular rational row R. Thus the convolution row is exact at the finite coefficient surface and is never replaced by a host-level Cauchy product. □

Theorem 368.8 (FastSeriesProduct real-seal factorization)**.** *For an accepted* $\mathsf{FastSeriesProduct}^{\uparrow}$ *carrier*

$$H = (F, G, K, W, M, T, R, E, L, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, which in turn is determined by the product-window ledger* W*, the pasted schedule row* M*, and the tail-bound ledger* T *built from the finite convolution row* K*. The rows* L*,* C*,* P*, and* N *certify componentwise transport, continuation readback, package provenance, and local naming, but add no quotient stream, selected product sum, host multiplication theorem, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductCarrier real seal factorization`

Proof. Start with the seal row E and project the regular handoff R from the same carrier. The carrier definition makes R the only regular rational route that the seal may read.

Use Definition 368.4 to identify R with the repacking of W, M, and T. Then use Definition 368.7 and Definition 368.3 to bind those rows to the finite convolution coefficients, pasted source schedules, and tail estimates.

The remaining coordinates L, C, P, and N only certify transport, readback, provenance, and naming for the displayed packet. Since the carrier has no field for quotient streams, a chosen product sum, host multiplication, or completeness, the real seal factors through the finite product certificate alone. □

Theorem 368.9 (FastSeriesProduct finite convolution obligation package)**.** *An accepted concrete* $\mathsf{FastSeriesProduct}^{\uparrow}$ BHist *packet*

$$H = (F, G, K, W, M, T, R, E, L, C, P, N)$$

supplies the finite convolution obligation package consumed by later $\mathsf{Real}^{\uparrow}$ *series work. The package consists exactly of the following obligation rows:*

1. *the carrier and classifier surface of Definition 368.1 and Definition 368.2;*
2. *the finite convolution tail-bound row of Definition 368.3;*
3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row of Definition 368.4;*
4. *the classifier transport exactness and convolution row exactness rows of Definition 368.6 and Definition 368.7;*
5. *the* $\mathsf{Real}^{\uparrow}$ *seal factorization row of Definition 368.8.*

Consequently a consumer of the product certificate may project the source packets, finite convolution rows, pasted schedule, tail ledger, regular rational handoff, real-seal row, transports, continuations, package provenance, and local naming row from H*, but obtains no quotient stream, selected limit, host real equality, or completeness principle.*

Proof. Project the displayed packet H. The first two coordinates F and G are the accepted source certificates, while $K, W, M, T, R, E, L, C, P, N$ are exactly the finite product rows named in Definition 368.1. The classifier surface is the same finite list read through Definition 368.2.

The tail-bound obligation is the row T, whose construction is recorded by Definition 368.3. The regular rational handoff is the row R, repacked from W, M, and T by Definition 368.4.

The exactness obligations are obtained by projecting the transport rows L and read routes C together with the finite convolution row K. Apply Definition 368.6 for transported packet acceptance, and apply Definition 368.7 for the displayed coefficient rows.

Finally Definition 368.8 projects the seal row E through the same regular handoff and finite ledger. Since every cited row is already a coordinate of H or a theorem about those coordinates, the package is assembled constructively by projection from the finite packet. No step forms a quotient stream, selects a limit, compares host real values, or invokes completeness. □

Theorem 368.10 (FastSeriesProduct scoped kernel interface)**.** *The carrier of Definition 368.1, the classifier of Definition 368.2, and the obligation rows of Definition 368.3, Definition 368.4, Definition 368.6, Definition 368.7, Definition 368.8, and Definition 368.9 form the scoped kernel interface for* $\mathsf{FastSeriesProduct}^{\uparrow}$ *over* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

A downstream consumer may project the two source series, finite convolution rows, product-prefix windows, pasted tail schedules, dyadic tail-bound ledgers, regular rational handoff, real-seal row, componentwise transports, continuation routes, package provenance, and local naming row from an accepted packet. It obtains no quotient stream, selected product sum, host multiplication theorem, or completeness principle.

Lean4 ✓ `BEDC.Derived.FastSeriesProductUp.FastSeriesProductUp StdBridge`

Proof. Project the accepted finite packet

$$H = (F, G, K, W, M, T, R, E, L, C, P, N).$$

The carrier and classifier definitions give the complete list of rows that the packet stores and compares.

The tail-bound, regular handoff, classifier-transport, convolution-exactness, and real-seal rows are then the named projections of that same packet. The finite convolution obligation package gathers those projections without adding an external object.

Thus the interface is exactly the displayed BEDC surface over the named source, tail-bound, regular rational, real-seal, transport, continuation, package, and naming rows. Because no cited row contains a quotient stream, selected product sum, host multiplication theorem, or completeness field, no such datum is exported by the scoped interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{FastSeriesProduct}^{\uparrow}$ packet is generated from two accepted $\mathsf{FastConvergentSeries}^{\uparrow}$ packets, finite convolution rows, product partial-sum windows, pasted fast tail schedules, tail-bound convolution ledgers, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 368.9, Definition 368.8, and Definition 368.10 bind the mature finite convolution, tail-bound, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, BHist, hsame, Cont, Pkg, and NameCert product interface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastSeriesProduct}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FastSeriesProductUp.FastSeriesProductCarrier\_tail\_bound\_convolution`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient equality of streams, uniqueness of product sums, host real multiplication, absolute convergence outside the displayed source packets, or bridge checking.

Upgrade path: No further upgrade remains; the chapter is at the top of the formal axis.

Chapter 369

A Concrete Naming Certificate for $\mathsf{UniformModulus}^{\uparrow}$

Inspired by chapter 9237; here re-expressed in the BEDC mathlib-free substrate.

The $\mathsf{UniformModulus}^{\uparrow}$ interface names the finite rational-fold packet that a compact-continuous consumer may read when it asks for a uniform precision history from a positive tolerance history. Its dependencies are the metric rows of $\mathsf{Met}^{\uparrow}$, the compact finite-net rows of $\mathsf{CompactMetric}^{\uparrow}$, the pointwise-modulus rows of $\mathsf{ContinuousMap}^{\uparrow}$, the positive rational order rows of $\mathsf{Rat}^{\uparrow}$, and the closed finite spine $\mathsf{ProbeBundle}$. The packet exports only the certified finite ledger from tolerance to precision; it does not expose an external modulus function, an open-cover compactness principle, a choice operator, or host real analysis. The sibling rows consumed here are the finite-net surface of chapter 129, the pointwise-modulus surface of chapter 130, the finite lower-fold surface of chapter 619, and the rational-order surface of chapter 37. The uniform-modulus packet consumes the compact finite-net rows of chapter 129 and the compact-continuous consumer surface of chapter 388; its finite lower-bound fold is the ledger row that those sibling packets hand to a modulus consumer. Finite-window compact consumers such as chapter 2372 read the same $\mathsf{UniformModulus}^{\uparrow}$ tolerance-only output only after the compact-net center-radius rows of chapter 129 have fed the finite fold and pointwise-modulus rows; no Real-facing or sequential compactness consumer receives a modulus outside that displayed compact-net route.

Definition 369.1 (UniformModulus BHist carrier)**.** *A* $\mathsf{UniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$$

with the following visible rows:

1. *E is a positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history read by the target metric consumer;*
2. *D is a positive* $\mathsf{Rat}^{\uparrow}$ *precision history proposed as the uniform modulus output;*
3. *$\mathcal{P}$ is a nonempty* $\mathsf{ProbeBundle}$ *spine of compact centers;*
4. *R records the positive* $\mathsf{Rat}^{\uparrow}$ *radius rows attached to the bundled centers, including the half-radius rows consumed by the finite fold;*
5. *C records the compact finite-net coverage rows saying that every source-point history is covered by one bundled center at the displayed half-radius;*
6. *M records the continuous pointwise-modulus rows at the bundled centers for the tolerance history E;*

7. *F is the finite lower-bound fold certificate over $\mathcal{P}$ proving that D is below every bundled half-radius ledger through the* $\mathsf{Rat}^{\uparrow}$ *classifier;*
8. *L records the componentwise* hsame *rows for tolerance, precision, radius, coverage, pointwise-modulus, and fold ledgers;*
9. *Q records the displayed* Cont *routes through which compact, metric, continuous-map, rational, and uniform-modulus consumers read the packet;*
10. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* ProbeBundle *source rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ *row.*

The classifier accepts exactly the packets whose tolerance, precision, radius, finite-net coverage, pointwise-modulus, lower-bound fold, comparison, continuation, package, and naming rows are all present in these displayed positions. It exports no external selector, no infinite cover, no quotient of point histories, and no analytic compactness theorem beyond the recorded finite packet.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusTasteGate single carrier alignment`

Theorem 369.2 (UniformModulus finite ProbeBundle fold)**.** *Let U be an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier in the sense of Definition 369.1. For its nonempty* ProbeBundle *spine $\mathcal{P}$, the positive* $\mathsf{Rat}^{\uparrow}$ *radius rows R and the finite lower-bound fold certificate F determine a positive precision history D such that every bundled half-radius ledger compares above D through the displayed* $\mathsf{Rat}^{\uparrow}$ *classifier and the componentwise* hsame *rows of L.*

The fold is internal to the finite ProbeBundle *packet. It has no coordinate for an ambient minimum of a set of rationals, a selected external enumeration, or a choice-selected modulus value.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket finite probe bundle fold`

Proof. Read the nonempty bundle row $\mathcal{P}$ from the carrier. Each bundled probe has a visible radius row in R, and each such row is a positive $\mathsf{Rat}^{\uparrow}$ history with the half-radius comparison needed by the compact finite-net coverage row.

Perform primitive recursion over the closed ProbeBundle spine. At the first bundled probe, use its half-radius history as the accumulator. At each successor probe, use the $\mathsf{Rat}^{\uparrow}$ order readback to replace the accumulator by the displayed lower of the current accumulator and the next half-radius row.

The certificate F records the comparison at every step, so the final precision history D is below every bundled half-radius ledger. Because the comparisons are stored as $\mathsf{Rat}^{\uparrow}$ rows and transported by the componentwise hsame rows in L, the result is stable under the $\mathsf{UniformModulus}^{\uparrow}$ classifier. □

Theorem 369.3 (UniformModulus ProbeBundle induction derivation)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$, the precision row D is obtained by primitive induction over the displayed nonempty* ProbeBundle *spine $\mathcal{P}$. The base row uses the first positive* $\mathsf{Rat}^{\uparrow}$ *half-radius ledger, and each successor step replaces the accumulator by the displayed lower* $\mathsf{Rat}^{\uparrow}$ *comparison with the next half-radius row. All comparisons are transported by L and replayed through Q, P, N.*

This derivation is a finite BEDC fold over $\mathcal{P}$. It does not appeal to compactness, a host minimum, or an external modulus selector.

Proof. Induct over the displayed nonempty ProbeBundle spine. In the base case, the first bundled probe supplies a positive $\mathsf{Rat}^{\uparrow}$ radius row, and its half-radius ledger becomes the initial accumulator for the precision row.

For the successor step, assume the accumulator is already below every half-radius ledger seen so far. Read the next bundled radius row from R, use the displayed $\mathsf{Rat}^{\uparrow}$ order comparison, and replace the accumulator by the lower of the old accumulator and the new half-radius ledger.

The fold certificate F records this comparison after each successor step. Therefore the terminal accumulator is a positive precision history D below every bundled half-radius row, exactly as in Definition 369.2.

Finally transport and replay the construction through L, Q, P, N. Componentwise hsame preserves the radius and comparison rows, while the displayed continuation, provenance, and naming rows expose the same finite induction to every consumer without invoking compactness, a host minimum, or an external selector. □

Theorem 369.4 (UniformModulus finite-net consumer threshold package)**.** *Let $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P,$ be an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet. If its fold coordinate F is supplied by an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet over the same* ProbeBundle *spine $\mathcal{P}$ and the same positive* $\mathsf{Rat}^{\uparrow}$ *radius rows R, then the* $\mathsf{UniformModulus}^{\uparrow}$ *precision row D is exactly the consumer threshold exported by that finite fold packet, transported through L and routed by Q, P, N.*

The package is a finite-row handoff from $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *to* $\mathsf{UniformModulus}^{\uparrow}$*. It uses no open-cover compactness, no host minimum, and no external modulus selector.*

Proof. Project the accepted $\mathsf{UniformModulus}^{\uparrow}$ packet and the accepted $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ packet supplying F. The shared rows identify the same nonempty ProbeBundle spine and the same positive $\mathsf{Rat}^{\uparrow}$ half-radius ledgers.

Apply Definition 369.2 to the $\mathsf{UniformModulus}^{\uparrow}$ packet. It reads F as the finite lower-bound fold that produces D below every bundled half-radius ledger.

Apply Definition 619.8 and Definition 619.9 to the supplying $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ packet. The first theorem supplies the lower-bound certificate, and the second supplies admissibility of the accumulator as a positive $\mathsf{Rat}^{\uparrow}$ precision history.

Transport the resulting threshold through the componentwise classifier row L, then route it through Q, P, N to the uniform-modulus consumer. Since both packets expose only their finite bundle, radius, fold, transport, route, provenance, and naming rows, no host minimum or compactness principle can replace the exported threshold. □

Theorem 369.5 (UniformModulus classifier transport stability)**.** *Let $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$ be an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet. If the tolerance row, precision row, finite-net coverage row, pointwise-modulus rows, rational fold rows, and downstream metric-read rows are transported componentwise along the displayed* hsame *entries of L and then read only through the displayed* Cont, Pkg, *and* NameCert *routes, the transported data remain one accepted classifier-facing compact-continuous packet.*

The transported packet preserves the same tolerance-to-precision ledger, finite ProbeBundle *coverage, pointwise-modulus rows, rational lower-bound fold, and metric consumer boundary. It introduces no external selector, open-cover compactness row, sequential compactness row, or host analytic modulus.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket classifier transport stability`

Proof. Project the carrier rows of Definition 369.1. The only classifier-readable coordinates are the tolerance, precision, bundle, radius, coverage, pointwise-modulus, fold, transport, continuation, provenance, and local naming rows listed there.

Transport those coordinates through the componentwise hsame entries in L. The tolerance and precision histories remain positive $\mathsf{Rat}^{\uparrow}$ rows, the compact finite-net row still reads the same bundled centers through $\mathcal{P}$, and the pointwise-modulus rows remain attached to those transported centers.

Replay the finite ProbeBundle fold of Definition 369.2 on the transported radius rows. The fold certificate moves with the rational comparison rows, so the transported precision remains below every transported half-radius ledger.

Finally route all downstream metric reads through Q, P, and N. The accepted packet is therefore reconstructed from displayed carrier rows and displayed continuation routes only, leaving no coordinate for an external selector, open-cover compactness, sequential compactness, or host analytic modulus. □

Theorem 369.6 (UniformModulus NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ *obligation surface consists of five finite rows: the carrier and classifier row of Definition 369.1; the compact finite-net row over* $\mathsf{CompactMetric}^{\uparrow}$ *and* ProbeBundle*; the continuous pointwise-modulus row over* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$*; the positive rational finite-fold row over* $\mathsf{Rat}^{\uparrow}$*; and the downstream boundary forcing every consumer read through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket namecert obligation surface`

Proof. The carrier row fixes the packet inventory. It lists the tolerance, precision, nonempty bundle, radius, coverage, pointwise-modulus, fold, comparison, continuation, package, and naming rows that the classifier may inspect.

The compact row is the coverage coordinate C together with the bundled centers of $\mathcal{P}$. The continuous row is M, which supplies the pointwise-modulus ledgers at those bundled centers and routes them through the metric source and target certificates.

The rational fold row is Definition 369.2. Repacking these rows with L, Q, P, and N gives exactly the finite local $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ surface, and no hidden selector or analytic compactness object is available to the consumer. □

Theorem 369.7 (UniformModulus root compact-continuous threshold)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the compact finite-net coverage row C together with the pointwise $\mathsf{ContinuousMap}^{\uparrow}$ *rows M determine the threshold surface read by a compact-continuous consumer. The row D is admitted as that threshold only after the finite* ProbeBundle *fold F has compared it with the displayed positive* $\mathsf{Rat}^{\uparrow}$ *radius rows R.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root compact continuous threshold`

Proof. Use the carrier definition Definition 369.1. The only compact-continuous source rows are the bundle $\mathcal{P}$, radius ledger R, coverage row C, pointwise-modulus rows M, and finite fold F.

Apply Definition 369.2 to produce the precision row D below the displayed half-radius rows. The fold is finite and internal to the ProbeBundle spine.

The obligation surface of Definition 369.6 then repacks the compact, continuous, rational-fold, transport, route, package, and local naming rows as the threshold surface. No additional compactness or continuity object is needed by the consumer. □

Theorem 369.8 (UniformModulus root net minimum row)**.** *If the fold coordinate F of an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet is supplied by a* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *row over the same finite* ProbeBundle *spine and positive* $\mathsf{Rat}^{\uparrow}$ *radius rows, then F supplies the positive precision row D without invoking a host minimum, an ambient finite-set operator, or an external selector.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root net minimum row`

Proof. Start with Definition 369.4. It identifies a supplying $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ packet as the source of the consumer threshold when the bundle and radius rows agree.

Use Definition 369.2. The row D is obtained by finite recursion over the displayed ProbeBundle spine and by positive $\mathsf{Rat}^{\uparrow}$ comparisons stored in F.

Finally apply classifier transport stability from Definition 369.5. Transport preserves the same fold row and cannot introduce a host minimum or external selector. □

Theorem 369.9 (UniformModulus root consumer package)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of the compact-continuous threshold factors through the displayed* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, hsame, Cont, Pkg, *and* NameCert *rows. The consumer-visible package is exactly*

$$E, D, \mathcal{P}, R, C, M, F, L, Q, P, N,$$

with no open-cover compactness, host minimum, sequential compactness, or external modulus coordinate.

Proof. Read the threshold using Definition 369.7. The compact row C, continuous rows M, finite bundle $\mathcal{P}$, radius rows R, and fold row F determine the threshold surface.

Apply Definition 369.8 to identify the positive $\mathsf{Rat}^{\uparrow}$ precision source. The precision is the finite fold output, not a host minimum.

Close the consumer route with Definition 369.6. Its downstream boundary forces every read through L, Q, P, N, so all compact-continuous consumers factor through the displayed BEDC rows and nothing outside them. □

Theorem 369.10 (UniformModulus carrier obligation)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$, carrier admission is exactly the displayed finite* BHist *packet of Definition 369.1. The admitted rows are the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance E, the positive* $\mathsf{Rat}^{\uparrow}$ *precision D, the nonempty* ProbeBundle *finite fold spine $\mathcal{P}$, the* $\mathsf{Rat}^{\uparrow}$ *tolerance and precision rows in R, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row C, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows M, the finite rational fold F, componentwise* hsame *data L, displayed* Cont *routes Q,* Pkg *provenance P, and the local* NameCert *row N.*

No admitted carrier coordinate is an open-cover compactness principle, a host minimum operation, or an external modulus selector.

Proof. Project the carrier tuple from Definition 369.1. Each row named in the statement occurs as one finite coordinate of the same BHist packet.

The ProbeBundle row supplies the finite fold spine, while the $\mathsf{Rat}^{\uparrow}$ rows provide the tolerance, precision, radius, and comparison data consumed by the fold. The compact row is the finite-net coverage coordinate, and the continuous row is the displayed pointwise-modulus ledger at bundled centers.

The remaining rows L, Q, P, and N are transport, continuation, package, and naming boundaries. Since carrier admission is exhausted by these displayed coordinates, there is no place for open-cover compactness, a host minimum, or an external modulus selector. □

Theorem 369.11 (UniformModulus classifier stability obligation)**.** *Classifier stability for an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet preserves the existing* BHist *carrier packet, the finite* $\mathsf{ProbeBundle}$ *fold, the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance and precision rows, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows, the componentwise* hsame *ledger, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

Thus any classifier-related packet reads the same finite tolerance-to-precision surface through displayed rows only. It cannot gain an open-cover compactness row, a host minimum, or an external modulus selector during transport.

Proof. Begin with the componentwise hsame data L. It transports the tolerance, precision, radius, coverage, pointwise-modulus, and fold ledgers without changing which coordinates are classifier-visible.

Use the displayed Cont routes Q to read the transported packet. The compact finite-net row still points to the same finite $\mathsf{ProbeBundle}$ spine, and the continuous pointwise-modulus rows remain attached to the bundled centers.

The Pkg provenance P and local $\mathsf{NameCert}$ row N bound the transported read. Because these rows only re-expose the finite carrier inventory, transport cannot introduce open-cover compactness, a host minimum, or an external modulus selector. □

Theorem 369.12 (UniformModulus ledger exactness obligation)**.** *The ledger exactness obligation for* $\mathsf{UniformModulus}^{\uparrow}$ *says that the precision row D is consumed only as the result of the finite* $\mathsf{ProbeBundle}$ *fold F over displayed* $\mathsf{Rat}^{\uparrow}$ *radius rows, with compact coverage supplied by C, pointwise-modulus rows supplied by M, componentwise* hsame *transport supplied by L, continuation routes supplied by Q, package provenance supplied by P, and local naming supplied by N.*

Every tolerance-to-precision read must therefore account for the same finite fold ledger. No read may replace the fold with open-cover compactness, a host minimum, or an external modulus selector.

Proof. Read the precision row D through Definition 369.2. Its only source is the finite lower-bound fold over the bundled radius rows.

Next read compactness and continuity through C and M. The finite-net coordinate supplies a bundled center, and the pointwise-modulus coordinate supplies the local tolerance handoff at that center.

Finally repack the route through L, Q, P, and N. These rows record transport, continuation, package, and local naming evidence for the same finite ledger, so the precision read has no remaining coordinate from which to import open-cover compactness, a host minimum, or an external modulus selector. □

Theorem 369.13 (UniformModulus compact-continuous handoff)**.** *Suppose a* $\mathsf{CompactMetric}^{\uparrow}$ *packet exposes a nonempty* $\mathsf{ProbeBundle}$ *finite-net row for a positive tolerance history and a* $\mathsf{ContinuousMap}^{\uparrow}$ *packet over the same* $\mathsf{Met}^{\uparrow}$ *source exposes pointwise-modulus rows at every bundled center. If these rows are admitted into a* $\mathsf{UniformModulus}^{\uparrow}$ *carrier, then the precision history produced by the finite lower-bound fold is a conservative uniform-modulus handoff for compact-continuous consumers: every source-distance read below that precision factors through one compact coverage row, the corresponding pointwise-modulus row, the target metric rows, and the same finite* $\mathsf{Rat}^{\uparrow}$ *fold certificate.*

The handoff adds no open-cover compactness row, no sequential compactness row, no host function-space continuity theorem, and no external modulus selector.

Proof. Start with the compact finite-net coordinate. For the requested tolerance history, the $\mathsf{CompactMetric}^{\uparrow}$ row supplies the nonempty bundle $\mathcal{P}$, the center histories, the radius rows, and the coverage witness assigning each source-point history to a bundled center.

Use the $\mathsf{ContinuousMap}^{\uparrow}$ coordinate at each bundled center. Its pointwise-modulus row gives the local implication from a source metric distance bounded by the displayed local radius to a target metric distance bounded by the requested tolerance budget.

Apply Definition 369.2. The produced precision history is below every bundled half-radius ledger, so the metric triangle rows route any pair of sufficiently close source histories through the same bundled center and then through the corresponding pointwise-modulus row.

All data used in the handoff are the rows displayed in Definition 369.1. Therefore the consumer receives a finite BEDC packet over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{ProbeBundle}$, not an external analytic uniform-continuity principle. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket compact continuous handoff`

Theorem 369.14 (UniformModulus compact metric source lock)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$$

as in Definition 369.1, every center and radius coverage read factors through the displayed $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row C, the finite* $\mathsf{ProbeBundle}$ *spine $\mathcal{P}$, and the* $\mathsf{Met}^{\uparrow}$ *distance witnesses attached to those rows. The source side exposes no open-cover coordinate, no host finite-subcover coordinate, and no selected center outside the carrier rows C, $\mathcal{P}$, and R.*

Consequently a compact coverage consumer of U can read only the bundled center, its displayed positive radius ledger, and the metric witnesses transported through L, Q, P, N.

Proof. Project the carrier inventory of Definition 369.1. The compact source coordinates available to the packet are exactly the finite bundle $\mathcal{P}$, the radius rows R, and the compact finite-net row C.

Apply Definition 369.13. Its compact step routes every source-distance read through a bundled center and the corresponding radius row before any pointwise-modulus row is consumed.

The boundary rows L, Q, P, N only transport, expose, package, and name the same displayed carrier data. Since the packet lists no open-cover or host finite-subcover coordinate, those objects cannot be read by a compact coverage consumer. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket compact metric source lock`

Theorem 369.15 (UniformModulus continuous map modulus lock)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

a pointwise-modulus row in M becomes consumer-visible only after the compact source lock has selected a bundled center through C and $\mathcal{P}$, and after the finite fold row F has produced the precision row D below the corresponding displayed radius ledger in R. The continuous-map row is therefore a local row attached to a selected compact center, not a standalone global modulus function.

All consumer-visible reads of M must remain inside the BHist *carrier, the* $\mathsf{Met}^{\uparrow}$ *source and target witnesses, the finite* $\mathsf{Rat}^{\uparrow}$ *fold, and the L, Q, P, N certificate boundary.*

Proof. First use the compact metric source lock of Definition 369.14. It fixes the source route through a displayed center in $\mathcal{P}$ and its radius ledger in R.

At that fixed center, the relevant continuous data are the corresponding row of M. The row is pointwise because it is indexed by the bundled center and the tolerance row E inside the same carrier.

Apply Definition 369.2. The fold row F produces D by finite recursion over the displayed bundle and records the comparison needed to consume the pointwise row uniformly after the selected-center route is known.

Finally transport the read through L, Q, P, N. These rows preserve the carrier boundary and leave no coordinate for an isolated global modulus function or a bypass of the compact source selection. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket continuous map modulus lock`

Theorem 369.16 (UniformModulus real precision ledger lock)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the precision row D is a positive $\mathsf{Rat}^{\uparrow}$ *and real-precision ledger row internal to the finite fold over R. It is transported to consumers only by the displayed* hsame *row L,* Cont *routes Q,* Pkg *provenance P, and local* NameCert *row N.*

Thus D is not a host minimum, an ambient infimum, or a selected modulus function. Its only consumer-visible role is the folded positive precision ledger used after compact source selection and pointwise continuous-map routing.

Proof. Begin with Definition 369.14. It confines compact coverage to the finite source bundle, the displayed radius rows, and the metric witnesses already present in the carrier.

Next use Definition 369.15. A pointwise-modulus row contributes to a uniform read only after the selected center is fixed and the finite fold has produced D.

Now apply Definition 369.22. That obligation identifies D as the positive rational precision ledger produced by the finite fold over R and consumed through C, M, L, Q, P, N.

Combining these routes exhausts the consumer-visible precision source. No host minimum, ambient infimum, or external selected modulus function remains as a possible source for D. □

Theorem 369.17 (UniformModulus finite-net center coverage obligation)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$$

as in Definition 369.1, every compact coverage read selects its center only from the displayed finite ProbeBundle *row $\mathcal{P}$. The tolerance row E, radius rows R, coverage row C, transport row L, continuation row Q, package row P, and local naming row N are the only rows through which the selected center may be exposed.*

Thus compact coverage cannot import an undisplayed center selector, an open-cover compactness principle, or a host finite-subcover operation. The selected center is a carrier coordinate read through C and transported through L, Q, P, N.

Proof. Project the carrier row of Definition 369.1. The only center spine present in the packet is the nonempty finite ProbeBundle row $\mathcal{P}$, and the only coverage coordinate is C.

Use the compact-continuous handoff of Definition 369.13. Its first step reads the finite-net coordinate and assigns each source-point history to a bundled center, not to an external selector.

The remaining rows E, R, L, Q, P, N record the tolerance budget, radius evidence, transport, continuation, provenance, and local name for the same bundled center read. Since the packet has no other center-bearing coordinate, coverage selects only displayed centers. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket finite net center coverage obligation`

Theorem 369.18 (UniformModulus pointwise-to-uniform route obligation)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every pointwise-modulus row in M reaches a uniform consumer only after the compact coverage row C has selected a center from $\mathcal{P}$ and the finite fold row F has produced the precision row D below the corresponding radius ledger in R. The route from pointwise data to a uniform read is therefore

$$E, \mathcal{P}, R, C, M, F, L, Q, P, N \to D,$$

with L, Q, P, N preserving the same displayed packet boundary.

No pointwise row may be read as a global modulus by itself, and no consumer may bypass the selected center, the rational fold, or the NameCert route.

Proof. Begin with the coverage side. By Definition 369.17, a source-point read reaches a center only through the displayed finite $\mathsf{ProbeBundle}$ row $\mathcal{P}$ and coverage row C.

At that center, the pointwise-modulus coordinate is the corresponding row of M. It is attached to the tolerance row E and the radius ledger R for the same bundled center.

Apply Definition 369.2. The fold row F produces the precision row D below every bundled half-radius ledger, so the local pointwise implication can be consumed uniformly after the selected-center route has been fixed.

Finally route the read through L, Q, P, N. These rows keep the pointwise-to-uniform handoff inside the displayed packet and leave no coordinate for an isolated global modulus or a bypass of the finite center and fold rows. □

Theorem 369.19 (UniformModulus rational fold floor)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the finite $\mathsf{ProbeBundle}$ *fold in F produces a positive* $\mathsf{Rat}^{\uparrow}$ *precision row D below every displayed half-radius row of R attached to a bundled center of $\mathcal{P}$. The comparisons are part of the same finite ledger and are read only through the* $\mathsf{Rat}^{\uparrow}$ *classifier, the componentwise transport row L, the continuation routes Q, the package provenance P, and the local naming row N.*

Thus the row D is a reusable rational floor for compact-continuous consumers. It is not a host minimum, an ambient infimum, a selected finite-subcover radius, or an external modulus value.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket rat positive fold floor`

Proof. Start with the finite fold theorem Definition 369.2. It constructs the precision row by primitive recursion over the closed nonempty spine $\mathcal{P}$ and stores each comparison against a displayed half-radius row in F.

The output row remains positive because every input half-radius row is a positive $\mathsf{Rat}^{\uparrow}$ row and each fold step records a positive lower comparison inside the same rational ledger. No external order operation is used to create the accumulator.

Finally transport the comparisons through L, Q, P, N. Since those rows are the only consumer boundary for the fold, every later read of D sees the same finite rational floor below the displayed half-radius rows. □

Theorem 369.20 (UniformModulus compact-continuous consumer boundary)**.** *Every compact-continuous consumer read of an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier factors through the displayed rows*

$$C, M, F, L, Q, P, N.$$

The compact row C chooses a bundled center from $\mathcal{P}$, the pointwise row M supplies the local modulus ledger at that center, the finite fold row F supplies the precision floor D, and the rows L, Q, P, N provide the transport, route, package, and local naming boundary.

No compact-continuous consumer may read a precision history through an open-cover compactness principle, a sequential compactness row, a host finite-subcover selector, or an external modulus function.

Proof. First use Definition 369.18. A pointwise row reaches a uniform read only after compact coverage has selected a displayed bundled center and the fold has produced the precision row.

Next apply Definition 369.19. The precision row consumed at that route is below every displayed half-radius ledger and is therefore reusable across the finite compact coverage packet.

The remaining access is through L, Q, P, N. These rows carry the same carrier data to the consumer boundary, so any read of the compact-continuous handoff is already a read through C, M, F, L, Q, P, N. □

Theorem 369.21 (UniformModulus NameCert threshold package)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$, the finite rational floor of Definition 369.19, the compact-continuous consumer boundary of Definition 369.20, and the local obligation surface of Definition 369.6 assemble the local* $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ *threshold package.*

The package has exactly five rows: carrier admission for $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$; compact finite-net coverage through C; pointwise continuous-map ledgers through M; positive rational lower-fold evidence through F; and consumer-boundary routing through L, Q, P, N.

Every downstream compact metric, continuous-map, or uniform-continuity consumer must consume this five-row packet before reading D as a uniform precision threshold.

Proof. Project the carrier row of Definition 369.1. It supplies the tolerance, precision, bundle, radius, coverage, pointwise, fold, transport, route, package, and naming coordinates used by the threshold package.

Use Definition 369.19 for the rational floor row. This identifies D as the positive finite-fold output below every displayed half-radius row.

Use Definition 369.20 for the consumer side. It forces every compact-continuous read through C, M, F, L, Q, P, N before the precision row is exposed.

Finally apply Definition 369.6. Its five finite rows are exactly the carrier, compact coverage, pointwise modulus, rational fold, and downstream boundary rows named above, so the local threshold package is complete. □

Theorem 369.22 (UniformModulus positive precision ledger obligation)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the positive precision row D is consumed only as the output of the finite $\mathsf{Rat}^{\uparrow}$ *fold row F over the displayed radius rows R on the finite* $\mathsf{ProbeBundle}$ *spine $\mathcal{P}$. Its downstream use must also pass through the compact coverage row C, pointwise-modulus row M, componentwise* hsame *row L, displayed* Cont *routes Q, package provenance P, and local naming row N.*

Thus D is never a host minimum, an ambient infimum, or an externally selected modulus value. It is a positive rational ledger row internal to the same finite carrier.

Proof. Read D using Definition 369.2. The theorem identifies the precision history with the result of primitive finite recursion over $\mathcal{P}$, using the positive $\mathsf{Rat}^{\uparrow}$ comparisons stored in R and F.

Use the pointwise-to-uniform route of Definition 369.18. Any consumer of D also consumes the compact coverage row C and the pointwise-modulus row M attached to the selected bundled center.

The rows L, Q, P, N then provide the transport, continuation, provenance, and local naming boundary. Since every precision read has already passed through the finite $\mathsf{Rat}^{\uparrow}$ fold and these displayed routes, no host minimum, infimum, or external modulus value is exported. □

Theorem 369.23 (UniformModulus scoped dependency interface)**.** *Any compact-continuous consumer of an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet can read the tolerance-to-precision handoff only through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows,* $\mathsf{Met}^{\uparrow}$ *distance witnesses, positive* $\mathsf{Rat}^{\uparrow}$ *fold rows,* $\mathsf{ProbeBundle}$ *membership rows, componentwise* hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *coordinates.*

The interface contains no open-cover compactness row, no sequential compactness row, no host minimum operation, and no external modulus function.

Proof. Begin with the finite $\mathsf{ProbeBundle}$ fold of Definition 369.2. It makes the precision row a positive $\mathsf{Rat}^{\uparrow}$ fold over the displayed bundle radii, so the handoff cannot read a host minimum or an external modulus function.

Transport the same rows through classifier stability in Definition 369.5. The $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows, $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus rows, $\mathsf{Met}^{\uparrow}$ distance witnesses, and rational fold rows remain the only classifier-visible dependency coordinates.

Now apply the NameCert obligation surface of Definition 369.6. It repacks those rows with $\mathsf{ProbeBundle}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ provenance as the complete local certificate surface.

Finally use the compact-continuous handoff of Definition 369.13. Every downstream tolerance-to-precision read follows the displayed finite-net and pointwise-modulus route, leaving no coordinate for open-cover compactness, sequential compactness, host minima, or external modulus functions. □

Theorem 369.24 (UniformModulus finite-fold scope closure)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *carrier* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, the finite* $\mathsf{ProbeBundle}$ *fold, classifier transport stability, and local* $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ *obligation surface bind the precision row* D *entirely to displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

In particular, every classifier-visible use of D *factors through the positive* $\mathsf{Rat}^{\uparrow}$ *fold rows* F*, the bundled probe spine* $\mathcal{P}$*, the componentwise transport ledger* L*, the continuation routes* Q*, the package provenance* P*, and the local naming row* N*. There is no further coordinate for a host minimum, an ambient finite-set operator, or an external modulus value.*

Proof. Read the carrier from Definition 369.1. The precision history D appears only as the proposed output row of the finite packet, while F records the rational comparisons that place D below the bundled half-radius ledgers.

Apply the finite $\mathsf{ProbeBundle}$ fold of Definition 369.2. It constructs the precision ledger by primitive finite recursion over $\mathcal{P}$ and stores each comparison as a displayed $\mathsf{Rat}^{\uparrow}$ row.

Use classifier transport stability from Definition 369.5. Transport can move the displayed rows along L, but it cannot add a new precision source outside the carrier coordinates.

Finally apply the local obligation surface of Definition 369.6. The rows Q, P, and N repack the fold, transport, and naming data as the complete classifier-facing surface, so every read of D is already exhausted by the displayed packet. □

Theorem 369.25 (UniformModulus center-radius readback)**.** *Every compact-continuous read of an accepted* UniformModulus$^{\uparrow}$ *packet first reaches a bundled center through the* CompactMetric$^{\uparrow}$ *finite-net rows and then uses the corresponding* ContinuousMap$^{\uparrow}$ *pointwise-modulus row. The source and target metric reads are mediated only by the displayed radius, coverage,* hsame*,* Cont*,* Pkg*, and* NameCert *witnesses of the same packet.*

Thus the center-radius route is a readback of displayed carrier data: compact coverage supplies the center, the finite fold supplies a precision below the relevant radius ledger, and the continuous row at that center supplies the tolerance handoff.

Proof. Start from the carrier inventory in Definition 369.1. The coverage row C is the only compact coordinate available to a source-point history, and it points to a center in the finite bundle $\mathcal{P}$ with its displayed radius rows R.

Use the finite-fold exactness supplied by Definition 369.2. Since D is below every bundled half-radius ledger through the displayed Rat$^{\uparrow}$ comparisons, any metric read below D can be routed through the radius ledger attached to the center chosen by C.

Now invoke the compact-continuous handoff of Definition 369.13. At the bundled center, the corresponding row of M is the ContinuousMap$^{\uparrow}$ pointwise-modulus ledger for the tolerance history E.

The transport and boundary data of Definition 369.5 and Definition 369.6 keep this route inside L, Q, P, and N. No metric read is therefore mediated by an undisplayed center selector or a host analytic compactness principle. □

Theorem 369.26 (UniformModulus compact-continuous non-escape)**.** *Downstream consumers of an accepted* UniformModulus$^{\uparrow}$ *packet cannot read open-cover compactness, sequential compactness, host minima, or external modulus functions. The classifier-visible coordinates are exhausted by the carrier rows of Definition 369.1 together with the scoped dependency interface of Definition 369.23.*

Consequently every compact-continuous tolerance-to-precision query must remain inside the same finite Rat$^{\uparrow}$ *fold,* CompactMetric$^{\uparrow}$ *finite-net coverage row,* ContinuousMap$^{\uparrow}$ *pointwise-modulus row,* Met$^{\uparrow}$ *distance witnesses,* ProbeBundle *membership rows,* hsame *transport ledger,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket compact continuous nonescape`

Proof. Project the accepted carrier. The definition lists all rows inspectable by the classifier and explicitly leaves no coordinate for open-cover compactness, sequential compactness, a host minimum, or an external modulus function.

Apply Definition 369.24 to the precision side. The precision row is tied to the finite rational fold over the displayed bundle and cannot be replaced by a host minimum or function-space selector.

Apply Definition 369.25 to the compact-continuous side. Every read reaches a displayed center and then a displayed pointwise-modulus row before reaching the target metric tolerance.

Finally use Definition 369.23. Since that interface is the complete consumer boundary for the carrier, any attempted downstream read outside these rows has no classifier-visible coordinate to inspect. □

Theorem 369.27 (UniformModulus modulus selection obligation)**.** *For an accepted* UniformModulus$^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the selected modulus row is exactly the positive Rat$^{\uparrow}$ *precision history D produced by the finite* ProbeBundle *fold F over the displayed radius rows R. Its consumer-visible selection route must pass*

through the compact finite-net rows C, the BHist *carrier, componentwise* hsame *rows L, displayed* Cont *routes Q,* Pkg *provenance P, and local* NameCert *row N.*

Thus a compact-continuous consumer cannot replace D by a host minimum, a choice-selected radius, an open-cover compactness datum, or an external modulus function.

Proof. Start with Definition 369.2. It constructs D by primitive recursion over the closed finite ProbeBundle spine and stores every comparison against the bundled radius rows in the fold ledger F.

Use Definition 369.25 to attach the selected row to compact coverage. The compact finite-net row C supplies the displayed center and radius readback, so the selected precision remains tied to R and the same BHist carrier.

Finally apply Definition 369.26. The rows L, Q, P, N are the only available transport, continuation, package, and naming boundary, so no undisplayed selector or host analytic modulus can enter the selection route. □

Theorem 369.28 (UniformModulus radius monotonicity obligation)**.** *For an accepted* UniformModulus$^{\uparrow}$ *packet $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$, every consumer read that weakens a displayed positive rational radius row in R must preserve the same folded precision obligation: the finite* ProbeBundle *fold still reads a positive row below the transported radius ledger, and the read remains mediated by C, the* BHist *carrier,* hsame *transport L,* Cont *routes Q,* Pkg *provenance P, and local* NameCert *row N.*

The monotonicity obligation is therefore internal to the displayed Rat$^{\uparrow}$ *comparisons and finite-net rows. It does not call an ambient order-theoretic infimum, a host metric radius theorem, or a separate compactness principle.*

Proof. Apply Definition 369.25. Every radius read is attached to a bundled center from $\mathcal{P}$ and is consumed through the compact finite-net row C before the target tolerance route is exposed.

Use Definition 369.2. Since the fold ledger records that D lies below every bundled half-radius row, weakening a displayed positive rational radius preserves the comparison needed by the same fold.

Close the route with Definition 369.26. The monotone radius read remains inside the finite Rat$^{\uparrow}$ comparison rows, L, Q, P, N, and the local carrier, leaving no coordinate for host order or analytic compactness data. □

Theorem 369.29 (UniformModulus compact-net coverage obligation)**.** *For an accepted* UniformModulus$^{\uparrow}$ *packet $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$, every compact-net coverage read is witnessed by a center already present in the finite* ProbeBundle *row $\mathcal{P}$ and by positive rational radius rows already present in R. The selected center, radius readback, pointwise-modulus handoff, and precision row are all consumed through the* BHist *carrier, compact finite-net row C, finite fold F,* hsame *transport L,* Cont *routes Q,* Pkg *provenance P, and local* NameCert *row N.*

No consumer may replace this coverage row with an open-cover compactness assertion, a sequential compactness principle, a choice-selected center, or an external finite-subcover operation.

Proof. Begin with Definition 369.25. It forces each compact-continuous read to reach a displayed bundled center and its radius row before any pointwise-modulus row is used.

Apply Definition 369.2. The precision row D is below the radius ledgers attached to every bundled center, so the coverage read and the folded modulus row are linked inside the same finite packet.

Use Definition 369.26 to finish. The compact-net coverage route is exhausted by $\mathcal{P}, R, C, M, F, L, Q, P, N$ and therefore has no row from which a host compactness or external finite-subcover operation can be read. □

Theorem 369.30 (UniformModulus ProbeBundle radius-source exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every radius read used by the finite fold* F *factors through the displayed* ProbeBundle *spine* $\mathcal{P}$*, the positive* $\mathsf{Rat}^{\uparrow}$ *radius rows* R*, the componentwise* hsame *transport ledger* L*, the displayed* Cont *routes* Q*, the package provenance row* P*, and the local* NameCert *row* N*.*

No radius consumed by the fold may be supplied by an open cover, a host metric radius selector, an ambient compactness theorem, or a row outside the local packet.

Proof. Project the carrier of Definition 369.1. The only rows that can feed F are the displayed finite spine $\mathcal{P}$ and the radius ledger R, with L, Q, P, N carrying transport, routing, provenance, and naming.

Apply Definition 369.2. It identifies the precision row D as the output of the finite fold over those bundled radius rows, so every comparison used by the fold is already a $\mathsf{Rat}^{\uparrow}$ row in the displayed packet.

Use Definition 369.29 to attach the same radius reads to compact finite-net coverage. Then close with Definition 369.26, which excludes all host compactness and external selector coordinates from the consumer route. □

Theorem 369.31 (UniformModulus pointwise-modulus center exactness)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row in* M *consumed by the uniform packet is attached to a center selected from the compact finite-net row* C *and the finite* ProbeBundle *spine* $\mathcal{P}$*. It is not attached to a standalone global function or to an undisplayed center.*

The center attachment is read through the $\mathsf{Met}^{\uparrow}$ *distance witnesses, the radius ledger* R*, the finite fold* F*, componentwise transport* L*, displayed routes* Q*, provenance* P*, and local naming* N*.*

Proof. Begin with Definition 369.25. A compact-continuous read reaches a displayed bundled center before it can consume a radius row or a pointwise-modulus row.

Apply Definition 369.29. The compact finite-net row C supplies the center used by the packet, and that center is already a member of $\mathcal{P}$.

Use Definition 369.30 to keep the attached radius and fold rows inside the same finite spine. The non-escape theorem Definition 369.26 then rules out standalone global functions, undisplayed centers, and external pointwise selectors. □

Theorem 369.32 (UniformModulus precision-consumer nonescape)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every consumer of the precision row* D *reads it only through the finite fold* F*, compact coverage row* C*, pointwise-modulus rows* M*, radius ledger* R*,* ProbeBundle *spine* $\mathcal{P}$*, transport ledger* L*, displayed* Cont *routes* Q*, package provenance* P*, and local* NameCert *row* N*.*

Thus a precision consumer cannot obtain D *from open-cover compactness, host minima, external selectors, ambient finite-subcover operations, or choice.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket precision consumer nonescape`

Proof. Start with the precision side of the carrier. The row D appears only as the folded positive $\mathsf{Rat}^{\uparrow}$ precision row, and the fold coordinate F records the finite comparisons against R over $\mathcal{P}$.

Use Definition 369.31. It ties every pointwise-modulus consumption of the selected precision to a center supplied by the compact finite-net row rather than by a global function.

Finally apply Definition 369.26. Since the consumer boundary is exhausted by $C, M, F, R, \mathcal{P}, L, Q, P, N$, no host minimum, open-cover argument, external selector, ambient finite-subcover operation, or choice principle can supply D. □

Theorem 369.33 (UniformModulus terminal compact-continuous route)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every compact-continuous precision consumer factors through exactly the compact finite-net row* C*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row* M*, the* $\mathsf{Met}^{\uparrow}$ *distance witnesses, the positive* $\mathsf{Rat}^{\uparrow}$ *fold rows* R, F, D*, the* ProbeBundle *spine* $\mathcal{P}$*, and the local transport, route, provenance, and naming rows* L, Q, P, N*.*

No such consumer reads open-cover compactness, sequential compactness, a host minimum, an external modulus function, quotient metric equality, or a choice-selected center.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusUp StdBridge`

Proof. Use Definition 369.13 to expose the compact-continuous handoff and Definition 369.25 to route each read through a displayed center and radius row.

Apply Definition 369.31 and Definition 369.30. These bind the pointwise-modulus row and every radius source to $C, M, R, F, \mathcal{P}$ and to the local rows L, Q, P, N.

Finally apply Definition 369.32. Repacking those routes gives the terminal Phase Compact and Continuous consumer boundary, with no remaining classifier-visible coordinate outside the displayed packet. □

Theorem 369.34 (UniformModulus root carrier row)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the root carrier row is the finite ProbeBundle *fold packet over* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows, positive* $\mathsf{Real}^{\uparrow}$ *precision rows, the* BHist *carrier, componentwise* hsame *transport* L*, displayed* Cont *routes* Q*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

Every compact-continuous read of the root carrier must therefore pass through the displayed bundle $\mathcal{P}$*, radius ledger* R*, coverage row* C*, pointwise rows* M*, finite fold* F*, and boundary rows* L, Q, P, N*.*

Proof. Project the carrier of Definition 369.1. Its root-facing data are exactly the finite bundle, radius ledger, compact coverage, pointwise modulus rows, fold certificate, and transport boundary displayed in U.

Apply Definition 369.29. The compact finite-net portion is read from $\mathcal{P}, R, C$ before any pointwise modulus row can be consumed.

Use Definition 369.2. The precision row is produced by the finite fold F over those displayed radius rows, and the output remains attached to the same BHist packet.

Finally close with Definition 369.26. The rows L, Q, P, N are the only consumer boundary, so the root carrier cannot import an open-cover compactness or choice-selection coordinate. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root carrier row exactness`

Theorem 369.35 (UniformModulus root classifier row)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, classifier transport is exactly the shared center-radius readback and positive-precision readback already displayed by the packet. A transported classifier read must use the compact finite-net row* C *to reach a center in* $\mathcal{P}$*, the radius ledger* R *to read the center radius, and the fold row* F *to expose the positive precision row* D *under* L, Q, P, N*.*

Proof. Begin with Definition 369.25. It identifies the classifier-visible center and radius route as the displayed route through $\mathcal{P}, R, C$.

Apply Definition 369.2. The positive precision row D is the folded readback from the same radius ledger, so classifier transport cannot change the threshold source.

Use Definition 369.28 to keep transported radius reads attached to the same finite fold obligation. The boundary rows L, Q, P, N then carry the shared readback without adding a host classifier or an analytic modulus. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root classifier row`

Theorem 369.36 (UniformModulus root consumer threshold row)**.** *Every compact-continuous consumer of an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet receives only the finite-net threshold row D, the pointwise modulus route through M, the rational fold floor F, and the ledger rows $\mathcal{P}, R, C, L, Q, P, N$. The consumer receives no open-cover compactness object, no sequential compactness row, no choice-like center selection, no host minimum, and no external modulus selector.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root consumer threshold row`

Proof. Read the consumer boundary through Definition 369.45. The exported rows are the tolerance, threshold, bundle, radius, coverage, pointwise, fold, transport, continuation, package, and naming rows of the same packet.

Apply Definition 369.35. It fixes center-radius and positive-precision readback as the shared classifier route available to the consumer.

Finally use Definition 369.26. After the finite-net threshold, pointwise route, rational fold floor, and ledger rows have been read, no coordinate remains for open-cover compactness, choice-like selection, a host minimum, or an external modulus selector. □

Theorem 369.37 (UniformModulus finite-net realizer unblock)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every compact finite-net read realized by a downstream compact-continuous consumer is witnessed by a displayed center in the finite ProbeBundle *row $\mathcal{P}$, a positive rational radius row in R, and the compact coverage row C. The only admissible transport from this realized finite net to the consumer boundary is through the componentwise* hsame *ledger L, the displayed* Cont *routes Q, the* Pkg *provenance row P, and the local* NameCert *row N.*

Thus the realized finite net is a BHist*-anchored packet over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* ProbeBundle*, not an open-cover compactness object, a host finite-subcover operation, or a choice-selected center.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket finite net realizer unblock`

Proof. Project the carrier of Definition 369.1. Its finite-net coordinates are exactly the bundled centers $\mathcal{P}$, the radius ledgers R, and the compact coverage row C.

Apply Definition 369.29. It identifies every compact-net coverage read with one of those displayed centers and radius rows before any consumer can ask for pointwise data.

Use Definition 369.26. The consumer boundary has no coordinate for open-cover compactness, host finite subcovers, or undisplayed center selection.

Finally route the realized row through L, Q, P, N. These are the displayed transport, continuation, package, and naming rows, so the realizer remains inside the same accepted packet. □

Theorem 369.38 (UniformModulus covered-point radius route)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, any source-point history covered by the compact row* C *and any precision row* D *produced by the finite fold* F *route through the same selected bundled center, its displayed positive rational radius row in* R*, and the* $\mathsf{Rat}^{\uparrow}$ *fold ledger before any pointwise-modulus row in* M *is consumed.*

The covered-point route is therefore a finite packet route through $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* ProbeBundle*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not use an ambient closest-center principle, an open-cover compactness row, or a host minimum for the source metric.*

Proof. Begin with Definition 369.25. A covered source-point history first reaches a center in $\mathcal{P}$ through C and then reads the radius row attached to that center.

Apply Definition 369.2. The precision row D is produced by the fold ledger F and is compared against the bundled radius rows before the pointwise-modulus row can be consumed.

Use Definition 369.28. Weakening or transporting a displayed radius keeps the same folded precision obligation, so the route cannot leave the $\mathsf{Rat}^{\uparrow}$ comparison rows.

The rows L, Q, P, N close the public route. They carry the covered-point, radius, fold, and pointwise rows to the consumer boundary without adding a host selector or compactness principle. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket covered point radius route`

Theorem 369.39 (UniformModulus pointwise metric consumer unblock)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, the finite-net realizer of Definition 369.37 and the covered-point radius route of Definition 369.38 assemble the consumer-facing compact-continuous packet over* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* ProbeBundle*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Every pointwise metric consumer of M *must first read the compact coverage center, the displayed radius row, and the folded precision ledger. The packet exports no open-cover compactness, no host minimum, no external modulus selector, and no choice-selected center.*

Proof. First use Definition 369.37. It supplies the finite $\mathsf{CompactMetric}^{\uparrow}$ realizer from the displayed bundle, radius rows, and compact coverage row.

Next use Definition 369.38. It forces each covered point to pass through the selected center, its radius ledger, and the $\mathsf{Rat}^{\uparrow}$ fold before the matching row of M is consumed.

Apply Definition 369.23. The consumer-facing coordinates are precisely $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows.

Finish with Definition 369.26. Since all consumer reads have already been routed through the displayed finite packet, there is no remaining coordinate for open-cover compactness, a host minimum, an external modulus selector, or a choice-selected center. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket pointwise metric consumer unblock`

Theorem 369.40 (UniformModulus mature compact-continuous package)**.** *An accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$$

assembles a mature compact-continuous package whose exported consumer surface consists exactly of the positive $\mathsf{Rat}^{\uparrow}$ *tolerance row* E*, the positive* $\mathsf{Rat}^{\uparrow}$ *precision row* D*, the finite nonempty* ProbeBundle *center spine* $\mathcal{P}$*, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coverage row* C*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows* M *at bundled centers, the* $\mathsf{Met}^{\uparrow}$ *distance witnesses used by the coverage*

and pointwise rows, the finite $\mathsf{Rat}^{\uparrow}$ *lower-bound fold* F*, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows* L, Q, P, N*.*

The package exports no open-cover compactness, sequential compactness, host minimum, ambient finite-set minimum, external modulus function, or choice-selected center. Every compact-continuous tolerance-to-precision consumer must therefore read the public package through the displayed carrier rows and through no further analytic or host-level coordinate.

Proof. Start with the finite precision row. By Definition 369.2, D is obtained from the finite positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold over the bundled radius rows, so the precision part of the public package is a displayed rational ledger rather than a host minimum.

Next apply Definition 369.13. The compact-to-continuous route consumes the finite-net coverage row C and the pointwise-modulus rows M at bundled centers, with the relevant $\mathsf{Met}^{\uparrow}$ distance witnesses read inside the same carrier.

Use Definition 369.25 to identify the center route. A consumer first reaches a center through the finite ProbeBundle spine and then reads the corresponding radius and pointwise-modulus rows, so no choice-selected center is exported.

Now invoke Definition 369.23. This pins the consumer surface to the positive $\mathsf{Rat}^{\uparrow}$ rows, ProbeBundle spine, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows displayed in U.

Finally apply Definition 369.26. The exclusion boundary removes open-cover compactness, sequential compactness, host minima, ambient finite-set minima, external modulus functions, and undisplayed center selectors, leaving exactly the mature compact-continuous package surface stated above. □

Theorem 369.41 (UniformModulus ProbeBundle membership exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every center, radius, and compact-coverage row consumed by the public compact-continuous route is witnessed by the displayed finite ProbeBundle *membership row* $\mathcal{P}$ *and is routed through the* BHist *carrier of Definition 369.1, the componentwise transport ledger* L*, the displayed continuation routes* Q*, the package provenance row* P*, and the local* NameCert *row* N*.*

Thus the finite consumer surface cannot read an undisplayed center, an external radius selector, or a compactness witness outside the packet rows $\mathcal{P}, R, C, L, Q, P, N$*.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket probe bundle membership exhaustion`

Proof. Project the carrier in Definition 369.1. Its compact-consumer coordinates list the finite bundle $\mathcal{P}$, the radius rows R, the compact coverage row C, and the boundary rows L, Q, P, N.

Apply Definition 369.2. The finite fold reads the same displayed ProbeBundle spine and records the positive rational comparisons needed by the precision row D.

Use Definition 369.13 and Definition 369.25. The handoff first reaches a bundled center and its radius row, then moves to compact coverage and pointwise data through the displayed packet route.

Finally invoke Definition 369.26. Since the nonescape boundary removes host compactness and undisplayed selectors, every consumed center, radius, and coverage row is exhausted by $\mathcal{P}, R, C, L, Q, P, N$. □

Theorem 369.42 (UniformModulus metric-distance route soundness)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every* $\mathsf{Met}^{\uparrow}$ *distance read below the folded precision*

row D factors through the compact coverage row C, the selected pointwise-modulus row in M, the target metric witnesses carried by the packet, and the finite fold ledger F.

The route is therefore a BHist*-anchored finite-distance route over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* ProbeBundle*, with transport and naming mediated only by L, Q, P, N.*

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket metric distance route soundness`

Proof. Begin with the carrier definition Definition 369.1. It places the source coverage row, pointwise-modulus rows, metric witnesses, fold ledger, and boundary rows inside the same accepted BHist packet.

Apply Definition 369.2. The precision row D is obtained from the finite ledger F, so any distance read below D must pass through the displayed rational comparisons recorded by that ledger.

Next use Definition 369.13. The compact-continuous handoff consumes C before selecting the pointwise row of M and the corresponding target metric witnesses.

Use Definition 369.25 to identify the center and radius row attached to the metric read. Close with Definition 369.26, which excludes any ambient distance route not represented in C, M, F, L, Q, P, N. □

Theorem 369.43 (UniformModulus compact-metric threshold factorization)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every compact-continuous tolerance-to-precision consumer factors through the finite ProbeBundle *center row $\mathcal{P}$, the* $\mathsf{CompactMetric}^{\uparrow}$ *coverage row C, the* $\mathsf{Met}^{\uparrow}$ *distance witness row, the* $\mathsf{Rat}^{\uparrow}$ *radius and fold ledger rows R, F, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row M before reading the folded precision row D.*

The factorization excludes open-cover compactness, sequential compactness, host finite-subcover selection, host minimum selection, and external modulus functions.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket compact metric threshold factorization`

Proof. Start with the finite-net realizer of Definition 369.37. It places the center row inside the finite ProbeBundle spine and ties the requested tolerance to displayed rational radius data.

Next use the covered-point radius route of Definition 369.38. The compact coverage row C can reach only a displayed center of $\mathcal{P}$ and the corresponding radius row in R.

Apply metric-distance route soundness from Definition 369.42. The source and target distance reads stay inside the packet route through $\mathsf{Met}^{\uparrow}$ witnesses, the selected pointwise-modulus row of M, and the finite fold ledger F.

Finally use consumer read exhaustion from Definition 369.45. Since downstream consumers read only $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, there is no remaining coordinate for open-cover compactness, sequential compactness, host finite-subcover selection, host minimum selection, or an external modulus function. □

Theorem 369.44 (UniformModulus root finite-net budget factorization)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every root tolerance-to-precision consumer factors through exactly the finite ProbeBundle *center row $\mathcal{P}$, the* $\mathsf{CompactMetric}^{\uparrow}$ *coverage row C, the* $\mathsf{Met}^{\uparrow}$ *distance witness row, the* $\mathsf{Rat}^{\uparrow}$ *radius and fold ledger rows R, F, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row M, and the boundary rows L, Q, P, N.*

The factorization excludes open-cover compactness, host finite-subcover selection, host minimum selection, and external modulus functions. In particular, the consumer may read the folded precision row D only after those displayed rows have supplied the center, coverage, distance, radius, pointwise, fold, transport, continuation, package, and local naming data.

Proof. First build the finite-net spine. Apply Definition 369.37; it places every center read inside the displayed ProbeBundle row $\mathcal{P}$ with its compact coverage row C and radius ledger R.

Next route covered points through displayed metric data. Use Definition 369.38 to reach the bundled center and radius row before any pointwise-modulus row is consumed.

Now apply Definition 369.42. The distance witness, selected pointwise-modulus row in M, and finite fold ledger F remain inside the same BHist packet and are transported only by L, Q, P, N.

Finally use Definition 369.45. Since the root consumer reads only $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, no coordinate remains for open-cover compactness, host finite-subcover selection, host minimum selection, or an external modulus function. □

Theorem 369.45 (UniformModulus consumer read exhaustion)**.** *For an accepted* UniformModulus$^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

any compact-continuous downstream consumer can read only the tolerance row E, folded precision row D, finite ProbeBundle *row $\mathcal{P}$, radius rows R, compact coverage row C, pointwise-modulus rows M, fold ledger F, transport ledger L, continuation routes Q, package provenance row P, and local* NameCert *row N.*

In particular the consumer surface excludes open-cover compactness, host minima, choice-selected centers, and external modulus functions.

Proof. Read the accepted carrier through Definition 369.1. The displayed tuple U contains exactly the tolerance, precision, bundle, radius, coverage, pointwise, fold, transport, continuation, package, and naming rows listed in the statement.

Apply Definition 369.2. This accounts for all reads of the folded precision row D and ties them to the finite bundle and radius ledger.

Apply Definition 369.13 together with Definition 369.25. These results account for compact coverage, selected center and radius readback, and the pointwise-modulus rows consumed by a compact-continuous downstream reader.

Finally use Definition 369.26. The nonescape theorem removes every remaining host or analytic coordinate, so the downstream consumer reads no data beyond $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$. □

Theorem 369.46 (UniformModulus metric source-distance factorization)**.** *For an accepted* UniformModulus$^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every source-distance read below the folded precision row D factors through the Met$^{\uparrow}$ *source-distance rows, the compact finite-net row C, the selected center of the displayed* ProbeBundle *spine $\mathcal{P}$, and the corresponding pointwise-modulus row of M.*

The factorization is a BHist *route in the carrier of Definition 369.1: the rational fold ledger F supplies the bound D, C chooses only a bundled center, M supplies the local continuous-map row at that center, and L, Q, P, N transport and name the same displayed packet.*

Proof. Read the packet through Definition 369.1. The carrier places the source metric rows, compact finite-net coordinate, bundled centers, pointwise-modulus rows, fold ledger, and boundary rows in one finite BHist packet.

Apply Definition 369.2. The precision row D is produced by the fold ledger F over displayed radius rows, so a source-distance read below D must first pass through the finite rational comparisons recorded by the packet.

Use Definition 369.42 together with Definition 369.13. The distance route reaches the compact coverage row C, selects a bundled center from $\mathcal{P}$, and then consumes the matching pointwise-modulus row of M.

Finally invoke Definition 369.45. Since the downstream consumer reads only $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, there is no remaining coordinate from which to import an external source-distance factorization. □

Theorem 369.47 (UniformModulus finite fold positive output)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the finite fold output D is a positive $\mathsf{Rat}^{\uparrow}$ *precision row produced solely by the fold certificate F over the displayed radius rows R of the finite* ProbeBundle *spine $\mathcal{P}$.*

All transport of this output is mediated by L, Q, P, N. Thus the row D cannot be read as a host minimum, an ambient finite-set infimum, or a selected external modulus value.

Proof. Begin with the carrier definition Definition 369.1. It identifies D as the precision row, R as the positive radius ledger, F as the finite lower-bound fold certificate, and L, Q, P, N as the only transport, continuation, provenance, and naming rows.

Apply Definition 369.2. Its primitive finite recursion over $\mathcal{P}$ produces a positive $\mathsf{Rat}^{\uparrow}$ accumulator and records every comparison against the displayed radius rows inside F.

Use Definition 369.42 and Definition 369.13 to see how the produced row is consumed. Both routes read D only through the packet fold before reaching compact coverage and pointwise data.

Close with Definition 369.45. Since every consumer read is exhausted by $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, the positive output row has no source outside F over R and no transport outside L, Q, P, N. □

Theorem 369.48 (UniformModulus compact-continuous consumer exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every compact-continuous consumer of the packet reads only the tolerance row E, folded precision row D, finite ProbeBundle *row $\mathcal{P}$, radius rows R, compact finite-net row C, pointwise-modulus rows M, finite fold ledger F, transport ledger L, continuation routes Q, package provenance row P, and local* NameCert *row N.*

In particular such a consumer cannot import open-cover compactness, a host minimum operation, a host finite-subcover selector, or a selected external modulus. The complete consumer surface is the displayed compact-continuous BHist *packet.*

Proof. Project the carrier in Definition 369.1. Its displayed rows are exactly $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, with no coordinate for open covers, host minima, external finite-subcover operators, or external modulus functions.

Apply Definition 369.2. This accounts for the precision row D as a positive finite fold over displayed radius rows, rather than as a host minimum.

Next apply Definition 369.13 and Definition 369.42. These results account for compact coverage, selected-center routing, source-distance reads, and pointwise-modulus consumption inside the packet.

Finally use Definition 369.45. It exhausts the downstream read boundary by the same eleven displayed rows, so the compact-continuous consumer has no channel for any non-packet compactness or modulus source. □

Theorem 369.49 (UniformModulus finite-net minimum handoff)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the displayed compact finite-net row C and the pointwise-modulus rows M supply the complete finite source from which the fold ledger F produces a positive rational lower-bound row D. The handoff is a finite ProbeBundle *fold over the radius rows attached to the bundled centers in $\mathcal{P}$.*

Thus the lower-bound row consumed by compact-uniform-continuity readers is a packet row built from $C, M, F, R, \mathcal{P}$ and transported only through L, Q, P, N. It is not a host minimum, an open-cover compactness principle, a finite-subcover selector, or an externally selected modulus.

Proof. Start from the carrier definition Definition 369.1. It places $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$ in one displayed BHist packet, with C as the compact finite-net row and M as the pointwise-modulus surface at the bundled centers.

Apply Definition 369.2. The fold reads the finite ProbeBundle spine $\mathcal{P}$ and the displayed positive radius rows R, then records the rational comparison steps inside F before exposing the lower-bound row D.

Use Definition 369.13 and Definition 369.42. These theorems account for the route from compact coverage and source distance to pointwise modulus rows, so the finite-net handoff has no separate compactness or host-minimum coordinate.

Finally invoke Definition 369.45. Every consumer read is exhausted by the displayed packet rows, so the positive rational lower bound can only be the row produced by F from $C, M, R, \mathcal{P}$ and transported through L, Q, P, N. □

Theorem 369.50 (UniformModulus rational lower-bound ledger)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the rational lower-bound ledger is exact for the displayed radius rows R and the finite fold certificate F. Each comparison used to expose the positive precision row D is a row of the packet, and each such row is tied to a bundled center from $\mathcal{P}$ through the compact finite-net and pointwise-modulus rows C and M.

The ledger therefore cannot be read as a host minimum computation, an open-cover compactness argument, an ambient compactness theorem, or a hidden selector for a modulus. It records only the rational lower-bound rows displayed by the packet.

Proof. Project the carrier in Definition 369.1. The only rows capable of producing and naming the lower-bound precision are R and F, with C and M supplying the finite-net and pointwise inputs and L, Q, P, N supplying transport, continuation, provenance, and naming.

Apply Definition 369.2. The theorem identifies the production of D with a finite rational fold over the displayed radius rows, so the ledger is exact for those rows and no unlisted comparison can contribute.

The route soundness of Definition 369.42 and the compact-continuous handoff of Definition 369.13 keep every downstream use on the same finite packet surface.

Close with Definition 369.45. Since all consumer reads factor through $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, the lower-bound ledger has no channel for host minima, open covers, hidden compactness, or external modulus choice. □

Theorem 369.51 (UniformModulus compact-continuous consumer exactness)**.** *Every compact-continuous consumer of an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet factors through exactly the displayed rows*

$$E, D, \mathcal{P}, R, C, M, F, L, Q, P, N.$$

The tolerance request, folded lower bound, finite ProbeBundle *spine, radius rows, compact finite-net row, pointwise-modulus rows, rational fold ledger, transport ledger, continuation routes, package provenance, and local* NameCert *row are the complete consumer surface.*

Consequently a compact-uniform-continuity or finite-net-minimum consumer can use the finite packet handoff, but it cannot import open-cover compactness, host minimum selection, host finite-subcover selection, quotient representatives, or any modulus outside the named packet rows.

Proof. Read the packet through Definition 369.1. The carrier lists exactly the rows $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$ and gives no coordinate for open covers, host minima, external selectors, quotient representatives, or off-packet modulus data.

Use Definition 369.2 to account for the folded lower-bound row D. The fold is finite over $\mathcal{P}$ and R and is recorded by F, so the consumer sees a rational packet row rather than a host minimum.

Apply Definition 369.13 together with Definition 369.42. These results keep compact coverage, source-distance routing, and pointwise-modulus consumption inside C and M.

Finally apply Definition 369.45. It states that the consumer read boundary is exhausted by the same displayed rows, which proves the claimed exactness of the compact-continuous surface. □

Theorem 369.52 (UniformModulus finite-net center route exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every center route consumed by a compact-continuous reader factors through the displayed rows $E, D, \mathcal{P}, R, C, L, Q, P, N$*. The center is read from the finite* ProbeBundle *spine* $\mathcal{P}$ *through the compact finite-net row* C*, its radius and precision constraints are the rows* R *and* D*, and the route reaches the consumer only through* L, Q, P, N*.*

Proof. Project the carrier through Definition 369.1. The only center-bearing rows are $\mathcal{P}$ and C, and the only precision and radius rows available to that route are D and R.

Apply Definition 369.2, Definition 369.13, and Definition 369.14. These results force compact source selection through the displayed finite bundle and radius ledger.

Finally use Definition 369.15 and Definition 369.51. Once the center route is fixed, every consumer read remains inside L, Q, P, N, so no open-cover, host finite-subcover, or selected external center can replace it. □

Theorem 369.53 (UniformModulus pointwise-modulus row consumption)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, each pointwise-modulus row in* M *is consumed only after the finite-net row* C *has selected a center from* $\mathcal{P}$ *and the fold ledger* F *has produced the precision row* D *from the displayed radius rows* R*. The row then reaches the compact-continuous consumer through the same* L, Q, P, N *boundary.*

Proof. Read the finite packet from Definition 369.1. The pointwise rows M are indexed by bundled centers, while F is the finite rational fold that exposes the shared precision row.

Use Definition 369.13 and Definition 369.14 to select the center through C and $\mathcal{P}$. Then use Definition 369.2 to produce D before any pointwise row is exported.

Apply Definition 369.15 and Definition 369.51. They keep the consumed row in the displayed compact-continuous packet and rule out a standalone global modulus function. □

Theorem 369.54 (UniformModulus fold-threshold nonescape)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the precision row D is exported only from the fold ledger F over displayed $\mathsf{Rat}^{\uparrow}$ *radius rows R. Its consumer route may pass through compact coverage C, pointwise rows M, and the boundary L, Q, P, N, but it cannot leave the finite fold surface for a host minimum, open-cover compactness principle, or external modulus selector.*

Proof. Start with Definition 369.1. The packet gives one precision row, one finite fold ledger, and the radius rows over which the fold is recorded.

Apply Definition 369.2. It identifies D as the positive output of the finite fold over R, not as a host minimum or selected external value.

Use Definition 369.13, Definition 369.14, and Definition 369.15 to account for the compact and pointwise rows that consume the threshold.

Close with Definition 369.51. Every exported precision read is exhausted by the displayed packet rows, so the fold threshold has no escape route outside F, R, L, Q, P, N and the compact-continuous rows that consume them. □

Theorem 369.55 (UniformModulus compact-continuous finite-fold correspondence)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the finite-net fold precision row D is available to a compact-continuous consumer if and only if the consumer route displays the compact finite-net center row C, the pointwise modulus rows M, the positive rational radius ledger R, the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound certificate F, and the local transport and provenance rows L, Q, P, N.*

Thus the compact-continuous consumer receives the finite-fold data precisely through the displayed BHist *packet,* ProbeBundle *spine,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *certificate, with no host minimum, external modulus selector, or ambient compactness row.*

Proof. For the forward direction, start from a consumer that receives the precision row D. Apply Definition 369.49 to identify the handoff as the finite-net minimum fold over the displayed compact row and bundled centers.

Use Definition 369.50. The available positive lower bound is the rational ledger over R, so the precision row is carried by the finite fold certificate F rather than by a host minimum. Then apply Definition 369.54 to keep the threshold route inside F, R, L, Q, P, N and the compact-continuous rows that consume them.

For the reverse direction, suppose the route displays C, M, R, F, L, Q, P, N with the finite ProbeBundle spine. Repack these rows through Definition 369.51. That exactness theorem exports the accepted compact-continuous surface from the same displayed packet, so the fold precision row D is available to the consumer. □

Theorem 369.56 (UniformModulus root carrier public row)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the public carrier row exposed to compact-uniform-modulus consumers is exactly the BHist *carrier of Definition 369.1 together with the finite* ProbeBundle *spine* $\mathcal{P}$*, compact-net coverage row* C*, pointwise-modulus rows* M*, rational lower-bound ledger* F*, and boundary rows* L, Q, P, N*.*

No root consumer receives a separate open-cover carrier, host finite-subcover carrier, external modulus carrier, or choice-selected center carrier.

Proof. Project the displayed carrier of Definition 369.1. Its public rows are $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, with the finite centers carried by the ProbeBundle coordinate and all reads anchored in the same BHist packet.

Apply Definition 369.37 and Definition 369.41. These identify the compact-net coverage and every consumed center with displayed bundle membership and the row C.

Use Definition 369.50. The folded lower-bound row is the exact finite ledger F over R, so no host minimum or external modulus carrier can supply the public precision row.

Finally invoke Definition 369.54. The threshold package reaches the consumer only through L, Q, P, N, which leaves the root carrier surface as the displayed BHist and ProbeBundle rows. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root carrier public row`

Theorem 369.57 (UniformModulus root classifier transport row)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every classifier-preserving root transport of the compact-uniform-modulus consumer surface factors through the componentwise* hsame *ledger* L*. The transport preserves the same displayed finite* ProbeBundle *carrier, compact-net coverage row, pointwise-modulus consumption route, rational lower-bound ledger, and* $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$ *threshold package.*

Thus an L*-mediated transport cannot replace the classifier by an ambient compactness classifier, a host metric classifier, or a selected external modulus classifier.*

Proof. Start with Definition 369.23. It fixes the classifier-facing surface to $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows.

Use Definition 369.38. A transported read must keep the selected bundled center, its radius row, and the fold ledger together before any pointwise-modulus row is consumed.

Apply Definition 369.53. The pointwise row reaches the consumer only after compact-net coverage and the folded threshold have both been read inside the same finite packet.

The boundary rows Q, P, N finish the route. Since L is the only classifier-preserving hsame row and the threshold is named by the displayed NameCert package, no ambient classifier or external modulus classifier has a transport coordinate. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root classifier transport row`

Theorem 369.58 (UniformModulus root ledger exactness row)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the root compact-uniform-modulus ledger is exact: compact-net coverage is read from C *over the finite* ProbeBundle *spine* $\mathcal{P}$*, pointwise-modulus consumption is read from* M*, the precision threshold is read from the rational lower-bound ledger* F *over* R*, and the exported packet is named only by* P *and* N*.*

Consequently the ledger contains no hidden compactness theorem, host minimum, finite-subcover selector, off-packet modulus, or quotient-representative read.

Proof. Read compact coverage through Definition 369.29 and Definition 369.37. These theorems bind every covered point to a center already present in $\mathcal{P}$ and to the compact row C.

Read pointwise consumption through Definition 369.39 and Definition 369.53. The consumer reaches M only after the finite coverage and radius rows have been consumed.

Read the threshold through Definition 369.50. The row D is exposed by the finite rational ledger F over R, not by a host minimum or off-packet modulus.

Finally apply Definition 369.51. It exhausts the exported rows by $E, D, \mathcal{P}, R, C, M, F, L, Q, P, N$, so the ledger exactness claim has no remaining coordinate for hidden compactness, external selection, or quotient data. □

Theorem 369.59 (UniformModulus compact-cover root row totality)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every compact-cover read made by a compact-continuous downstream consumer is exhausted by the displayed $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row* C*, the finite* $\mathsf{ProbeBundle}$ *spine* $\mathcal{P}$*, the positive* $\mathsf{Rat}^{\uparrow}$ *radius rows* R*, and the boundary rows* L, Q, P, N*.*

Thus the root compact-cover obligation is a finite packet obligation over the carrier of Definition 369.1. It has no coordinate for open-cover compactness, host finite-subcover selection, choice-selected centers, or an external cover selector.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root compact cover row totality`

Proof. Read the accepted carrier through Definition 369.1. The compact-cover coordinates visible to the consumer are exactly the bundled centers $\mathcal{P}$, the radius ledger R, the compact finite-net row C, and the transport, route, provenance, and naming boundary L, Q, P, N.

Apply Definition 369.29. It binds every compact-net coverage read to a center already present in $\mathcal{P}$ and to the radius rows already recorded in R.

Apply Definition 369.2 to keep the precision threshold tied to the same finite bundle and radius ledger. Then use Definition 369.26 to exclude any compactness or center-selection channel outside $C, \mathcal{P}, R, L, Q, P, N$. □

Theorem 369.60 (UniformModulus pointwise-modulus root row totality)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet* $U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N)$*, every pointwise-modulus read in* M *is available only after the compact-cover row* C *has selected a center bundled in* $\mathcal{P}$ *and routed the relevant source and radius data through the displayed* $\mathsf{Met}^{\uparrow}$ *witnesses. The read then reaches the consumer only through* L, Q, P, N*.*

Consequently the pointwise-modulus root row is a dependent finite-row handoff from compact coverage to metric witnesses to M*. It is not a global modulus function, a host continuity theorem, a choice-selected center rule, or a pointwise selector outside the carrier rows.*

Proof. Begin with Definition 369.1. The pointwise rows M are carrier rows indexed by bundled centers, and the only consumer route to such a center is the compact finite-net row C over the finite spine $\mathcal{P}$.

Apply Definition 369.29. It forces the compact-cover read to choose a displayed bundled center and radius row before the pointwise row can be consumed.

Use Definition 369.2 to keep the folded threshold D below the radius rows attached to those centers. The metric witnesses read by the pointwise row therefore remain on the displayed $\mathsf{Met}^{\uparrow}$ route rather than on a host selector.

Finally apply Definition 369.26. The rows L, Q, P, N are the only boundary through which the pointwise data can leave the packet, so M has no external modulus or undisplayed center coordinate. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root pointwise modulus row totality`

Theorem 369.61 (UniformModulus fold-threshold root exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the precision threshold D is exhausted by the finite $\mathsf{Rat}^{\uparrow}$ *fold row F over the displayed radius rows R. The compact-cover row C and pointwise rows M may consume the threshold through the packet route, but no consumer can replace it by a host minimum, open-cover compactness proof, choice-selected center, or external modulus selector.*

Proof. Project the rows of Definition 369.1. The only precision-producing coordinates are the displayed radius rows R, the fold certificate F, and the named precision row D.

Apply Definition 369.2. The fold is primitive finite recursion over the closed $\mathsf{ProbeBundle}$ spine and records every $\mathsf{Rat}^{\uparrow}$ comparison used to expose D.

Use Definition 369.29 to attach each consumption of D to displayed compact coverage and bundled centers. Then apply Definition 369.26; the threshold can leave the packet only through C, M, L, Q, P, N, so there is no host minimum, open-cover compactness row, choice-selected center, or external modulus selector. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root fold threshold exhaustion`

Theorem 369.62 (UniformModulus root-unblock radius family exhaustion)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

the root-unblock radius family consumed by any compact-continuous downstream reader is exactly the displayed finite $\mathsf{Rat}^{\uparrow}$ *family R indexed by the* $\mathsf{ProbeBundle}$ *spine $\mathcal{P}$ and read through the* $\mathsf{CompactModulusCover}^{\uparrow}$ *coverage row, the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *fold row F, the* BHist *carrier of Definition 369.1, componentwise* hsame *transport L, displayed* Cont *routes Q,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N.*

Thus the root-unblock precision route is exhausted by the displayed finite radius family. It has no coordinate for host open-cover compactness, external finite-subcover selection, hidden radius choice, or an ambient modulus selector.

Proof. Project the carrier through Definition 369.1. The only radius-family rows visible to a consumer are the finite bundle $\mathcal{P}$, the positive rational radius ledger R, the compact coverage row, and the fold certificate F.

Apply Definition 369.59. It anchors every compact-cover read in the displayed centers, radius rows, and compact finite-net row before the consumer boundary can be reached.

Use Definition 369.61. The precision row D is exhausted by the fold over R, so the root-unblock radius family cannot be replaced by a host minimum or ambient selector.

Finally apply Definition 369.60. The remaining route is through L, Q, P, N, leaving no coordinate for host open-cover compactness, external finite-subcover selection, or hidden radius choice. □

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root unblock radius family exhaustion`

Theorem 369.63 (UniformModulus root-unblock consumer threshold route)**.** *For an accepted* $\mathsf{UniformModulus}^{\uparrow}$ *packet*

$$U = (E, D, \mathcal{P}, R, C, M, F, L, Q, P, N),$$

every consumer threshold route from the requested tolerance row E *to the selected precision row* D *factors through the finite* $\mathsf{Rat}^{\uparrow}$ *radius family* R*, the* ProbeBundle *spine* $\mathcal{P}$*, the* $\mathsf{CompactModulusCover}^{\uparrow}$ *coverage row, the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *fold certificate* F*, the pointwise rows* M*, the* BHist *carrier, componentwise* hsame *transport* L*, displayed* Cont *routes* Q*,* Pkg *provenance* P*, and local* NameCert *row* N*.*

Consequently the uniform precision route is a finite threshold route inside the packet. It cannot use host open-cover compactness, external modulus selection, an undisplayed compact cover, or a choice-selected center.

Lean4 ✓ `BEDC.Derived.UniformModulusUp.UniformModulusPacket root unblock consumer threshold route`

Proof. Start with the threshold source and target rows in the carrier: E is the requested tolerance row, and D is the folded precision row.

Apply Definition 369.62. It fixes the available radius family as the displayed R indexed by $\mathcal{P}$ and routed through the compact coverage and fold rows.

Use Definition 369.60. It forces each pointwise-modulus read to pass through the compact-cover row, displayed metric witnesses, and the same boundary rows.

Close with Definition 369.61. Since every threshold read has already entered the displayed packet route, there is no remaining coordinate for open-cover compactness, external modulus selection, an undisplayed cover, or a choice-selected center. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformModulus}^{\uparrow}$ packet starts from a positive tolerance history, a compact finite-net ProbeBundle, continuous pointwise-modulus rows at the bundled centers, and positive rational radius rows; it folds the finite radius ledger to a precision history and routes every compact-continuous read through displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 369.13, Definition 369.23, Definition 369.24, Definition 369.25, Definition 369.26, Definition 369.40, Definition 369.46, Definition 369.47, and Definition 369.48 bind the public compact-continuous handoff surface for the displayed $\mathsf{UniformModulus}^{\uparrow}$ packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.UniformModulusUp.UniformModulusUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close a public uniform-continuity theorem, checked Lean implementation, bridge status, open-cover compactness, sequential compactness, or host analytic modulus selection.

Upgrade path: No further verification-axis upgrade remains for the standard compact-continuous handoff bridge.

Chapter 370

A Concrete Naming Certificate for GoldenMeanShift$^{\uparrow}$

The GoldenMeanShift$^{\uparrow}$ object is a BEDC-native finite-window name for binary BHist rows with no adjacent visible one rows, not an import of symbolic-dynamics machinery.

370.1 Finite binary windows

This finite no-adjacent-one carrier is consumed by the subshift finite type certificate of chapter 434 as the singleton forbidden-word instance. The shared dependency is the finite alphabet row, finite word row, forbidden adjacency ledger, acceptance ledger, and componentwise hsame classifier that records exactly the zero-zero, zero-one, and one-zero neighboring decisions.

Definition 370.1 (GoldenMeanShift carrier)**.** *A* GoldenMeanShift$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, Z, O, P, L)$$

with the following visible rows:

1. *W is a finite list of binary bit rows, each row displayed as either a zero bit row or a one bit row inside* BHist*;*
2. *Z is the all-zero witness row for the same window shape;*
3. *O is the adjacency ledger whose every neighboring pair in W is one of zero-zero, zero-one, or one-zero;*
4. *P is the* Pkg *provenance row naming the prefix and fold readbacks used to inspect the window;*
5. *L is the* NameCert *ledger recording that every accepted classifier row is a componentwise* hsame *comparison between displayed bit rows.*

The carrier accepts exactly finite binary windows whose visible bit rows contain no adjacent one rows. It contains no infinite sequence, no shift-invariant quotient, no entropy row, and no host symbolic-dynamics theorem.

Lean4 ✓ `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftTasteGate single carrier alignment`

Theorem 370.2 (GoldenMeanShift local stability)**.** *Let $G = (W, Z, O, P, L)$ be an accepted* GoldenMeanShift$^{\uparrow}$ *carrier. If a transported packet $G' = (W', Z', O', P', L')$ is obtained by moving each displayed bit*

row of W, each all-zero witness row, each adjacency-ledger row, and each provenance row through the displayed hsame, Cont, Pkg, *and* NameCert *routes, then G' is accepted whenever every transported adjacent pair remains one of zero-zero, zero-one, or one-zero.*

The induced classifier between G and G' is the componentwise hsame *classifier on visible bit rows, with the transported adjacency ledger as its exactness record.*

Lean4 ✓ `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftCarrier local stability`

Proof. Project the carrier rows of Definition 370.1. The only visible data are the binary window, the all-zero witness, the adjacency ledger, the prefix and fold provenance, and the local name-certificate ledger.

Move the displayed rows through the supplied hsame, Cont, Pkg, and NameCert routes. The hypothesis keeps each transported adjacent pair inside the listed local patterns, so no adjacent one-one pair can appear in the transported window.

For the classifier, compare the original and transported bit rows componentwise by hsame. The all-zero witness is transported by the same row movement, and the adjacency ledger gives the exactness witness that the accepted carrier condition is preserved. □

Theorem 370.3 (GoldenMeanShift fold-prefix closure)**.** *Let $G = (W, Z, O, P, L)$ be an accepted* GoldenMeanShift$^{\uparrow}$ *carrier, and let $W_{\leq k}$ be any displayed prefix read from W through the recorded* Cont *fold route and* Pkg *provenance. The restricted packet*

$$G_{\leq k} = (W_{\leq k}, Z_{\leq k}, O_{\leq k}, P_{\leq k}, L_{\leq k})$$

is an accepted GoldenMeanShift$^{\uparrow}$ *carrier. Its classifier rows are exactly the componentwise* hsame *rows inherited from G, and its ledger is the prefix restriction of the original* NameCert *ledger.*

Equivalently, every finite fold readback of an accepted no-adjacent-one window lands again inside the same carrier surface.

Proof. Read the prefix through the displayed Cont fold route and Pkg provenance. The route selects an initial segment of the same finite bit rows and also restricts the all-zero witness, adjacency ledger, provenance row, and name-certificate ledger.

Every adjacent pair inside $W_{\leq k}$ was already an adjacent pair inside W. Since G is accepted, each such pair is zero-zero, zero-one, or one-zero, and the restricted adjacency ledger records the same local fact.

The inherited classifier is componentwise hsame on the retained bit rows. The ledger exactness row is simply the restricted NameCert row, so the fold-prefix packet has no new tail, quotient, inverse-limit, entropy, or host shift coordinate. □

Theorem 370.4 (GoldenMeanShift adjacency ledger exhaustion)**.** *Every accepted neighboring-pair decision in a* GoldenMeanShift$^{\uparrow}$ *carrier $G = (W, Z, O, P, L)$ is exhausted by the displayed zero-zero, zero-one, or one-zero adjacency row in O together with the componentwise* hsame, Cont, Pkg, *and* NameCert *routes. No one-one decision, Fibonacci count, entropy row, inverse limit, or host symbolic-dynamics object can justify acceptance.*

For every displayed prefix or finite fold readback of G, the restricted packet preserves exactly the same local decision surface: each retained neighboring pair is accepted only by the corresponding restricted zero-zero, zero-one, or one-zero ledger row.

Proof. Project the carrier of Definition 370.1. The window rows, all-zero witness, adjacency ledger, provenance row, and local NameCert ledger are the only coordinates available to the classifier.

Inspect the adjacency ledger O. Each accepted neighboring pair is recorded as zero-zero, zero-one, or one-zero; the carrier definition has no fourth row that could certify a one-one neighbor.

Apply the prefix closure of Definition 370.3. Any displayed prefix or fold readback only restricts W, Z, O, P, and L, so every retained neighboring-pair decision is inherited from the same local ledger.

The componentwise hsame, Cont, Pkg, and NameCert routes transport these visible rows without adding another acceptance source. Thus Fibonacci counts, entropy rows, inverse limits, and host symbolic-dynamics objects remain outside the classifier surface. □

Theorem 370.5 (GoldenMeanShift NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{GoldenMeanShift}^{\uparrow}}$ *obligation surface consists of five finite obligations:*

1. *carrier habitation: the all-zero window Z is accepted because every displayed adjacent pair is zero-zero;*
2. *classifier laws: reflexivity, symmetry, and transitivity are inherited componentwise from* hsame *on the visible bit rows;*
3. *carrier transport: Definition 370.2 transports accepted windows through displayed* hsame*,* Cont*,* Pkg*, and* NameCert *routes;*
4. *fold-prefix closure: Definition 370.3 keeps every recorded prefix and fold readback inside the same carrier;*
5. *ledger exactness: the* NameCert *ledger records exactly the accepted bit rows, their adjacency ledger, the prefix and fold provenance, and the componentwise classifier rows.*

These obligations define the seed certificate surface for finite no-adjacent-one BHist *windows.*

Lean4 ✓ `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftCarrier semantic name certificate`

Proof. For habitation, take the all-zero window with the displayed all-zero witness. Every neighboring pair is zero-zero, so the adjacency ledger is generated without any one-one row.

For the classifier laws, work componentwise over the visible bit rows. Reflexivity, symmetry, and transitivity are the corresponding hsame laws applied at each displayed position, and the carrier restriction prevents the classifier from ranging over any hidden non-binary row.

Carrier transport is exactly the stability statement of Definition 370.2. The transported adjacency ledger is the witness that the accepted no-adjacent-one condition remains local to the finite BHist packet.

Fold-prefix closure is exactly Definition 370.3. A prefix or fold readback can only delete visible rows and restrict ledgers, so it cannot create an adjacent one-one pair.

Finally, ledger exactness follows by unfolding the five carrier rows of Definition 370.1. The ledger contains the accepted bit rows, adjacency decisions, prefix and fold provenance, and componentwise hsame classifier rows, and it contains no Fibonacci counting, Zeckendorf arithmetic, entropy, inverse-limit, or host symbolic-dynamics datum. □

Theorem 370.6 (GoldenMeanShift scoped ledger binding)**.** *For an accepted* $\mathsf{GoldenMeanShift}^{\uparrow}$ *carrier*

$$G = (W, Z, O, P, L),$$

the scoped BEDC ledger is exactly the finite no-adjacent-one BHist *window rows W, the all-zero witness Z, the zero-zero/zero-one/one-zero adjacency ledger O, the displayed fold-prefix* Cont *and* Pkg *provenance P, and the local* NameCert *row L. Every accepted row in this scope is bound*

componentwise to the hsame *classifier, the local stability route, the fold-prefix closure route, and the adjacency ledger exhaustion row. No row outside W, Z, O, P, L can contribute acceptance evidence.*

Proof. Project the carrier of Definition 370.1. The finite bit window, all-zero witness, adjacency decisions, prefix and fold provenance, and local naming ledger are the only displayed coordinates.

Use Definition 370.2 for componentwise hsame transport. The transport route moves exactly the displayed bit rows, all-zero witness, adjacency ledger, provenance row, and local NameCert row.

Use Definition 370.3 for finite prefix readbacks. A prefix only restricts W, Z, O, P, L, so it remains inside the same scoped ledger.

Finally apply Definition 370.4. The accepted adjacency alternatives are exhausted by the zero-zero, zero-one, and one-zero rows in O, leaving no further acceptance coordinate. □

Theorem 370.7 (GoldenMeanShift prefix transport scope)**.** *Let $G = (W, Z, O, P, L)$ be an accepted* GoldenMeanShift$^{\uparrow}$ *carrier, and let $G_{\leq k}$ be any finite prefix or fold readback supplied by the displayed* Cont *route and* Pkg *provenance. If $G_{\leq k}$ is transported componentwise by the stored* hsame *rows and the transported adjacency ledger still lists only zero-zero, zero-one, and one-zero rows, then the transported prefix remains an accepted* GoldenMeanShift$^{\uparrow}$ *carrier over the restricted* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Conversely, every accepted transported prefix of G is obtained by restricting the same five rows W, Z, O, P, L and then applying those displayed transport routes.

Lean4 ✓ `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftCarrier prefix transport scope`

Proof. First form the prefix packet with Definition 370.3. The theorem restricts the visible bit rows, all-zero witness, adjacency ledger, provenance row, and local certificate row.

Next transport the restricted packet through Definition 370.2. The stability hypothesis is exactly the retained adjacency condition, so the transported prefix remains accepted.

For the converse direction, project an accepted transported prefix back through the carrier coordinates. The prefix has no coordinate other than the restricted W, Z, O, P, L rows, and Definition 370.4 identifies the retained adjacency decisions as restrictions of the original local ledger. □

Theorem 370.8 (GoldenMeanShift adjacency scope exactness)**.** *For every accepted* GoldenMeanShift$^{\uparrow}$ *carrier, the adjacency scope is exact: an adjacent visible bit-pair is accepted precisely when the carrier ledger records it as zero-zero, zero-one, or one-zero, and the associated* NameCert *row records the same decision through the componentwise* hsame*,* Cont*, and* Pkg *routes. The scoped ledger therefore exports finite no-adjacent-one acceptance over* BHist *only, and it exports no Fibonacci count, entropy invariant, inverse-limit object, or host shift-space equality.*

Proof. Project the adjacency row O from Definition 370.1. Its accepted constructors are the three displayed local alternatives.

Apply Definition 370.4. This identifies every accepted neighboring-pair decision with one of those alternatives and rules out a one-one acceptance route.

The fold-prefix case follows by Definition 370.3: restriction only deletes rows and never adds another decision source. The transport case follows by Definition 370.2: the NameCert row follows the same componentwise hsame, Cont, and Pkg routes as the displayed adjacency ledger. □

Theorem 370.9 (GoldenMeanShift carrier row scope)**.** *The carrier rows W, Z, O, P, L of Definition 370.1 are exactly the finite* BHist *bit-window rows, the all-zero witness row, the zero-zero/zero-one/one-zero adjacency ledger, the displayed* Cont *and* Pkg *provenance rows, and the local* NameCert *ledger. No entropy coordinate, inverse-limit coordinate, infinite shift space, or host symbolic-dynamics row belongs to the carrier.*

Proof. Project the five coordinates in Definition 370.1. The W coordinate is the finite visible bit-window row, Z is the all-zero witness for that window shape, and O is the finite adjacency ledger whose accepted neighboring rows are zero-zero, zero-one, or one-zero.

The remaining coordinates are provenance and naming data: P records the displayed prefix and fold readbacks through Cont and Pkg, while L records the local NameCert ledger for the componentwise classifier rows. Since the definition lists no further coordinate, entropy, inverse limits, infinite host shifts, and external symbolic-dynamics objects have no carrier row to project. □

Theorem 370.10 (GoldenMeanShift classifier row scope)**.** *The* GoldenMeanShift$^{\uparrow}$ *classifier is componentwise* hsame *on the visible bit rows of W. Its transport surface is precisely the local stability route of Definition 370.2: transported bit rows, all-zero witnesses, adjacency ledgers, provenance rows, and* NameCert *rows remain accepted exactly when the transported adjacency ledger still lists only zero-zero, zero-one, or one-zero rows.*

Proof. The classifier row named in Definition 370.1 and Definition 370.5 compares displayed bit rows componentwise by hsame. There is no hidden quotient or ambient shift coordinate on which another classifier could act.

Apply Definition 370.2. That theorem transports exactly the visible bit rows, the all-zero witness, the adjacency ledger, the Cont and Pkg provenance, and the local NameCert row. The transported packet is accepted precisely under the same local adjacency alternatives, so the classifier transport surface is the local stability row and nothing larger. □

Theorem 370.11 (GoldenMeanShift ledger row scope)**.** *The* GoldenMeanShift$^{\uparrow}$ *ledger exhausts adjacency decisions by the zero-zero, zero-one, and one-zero rows. Its finite prefix and fold restrictions are exactly the restrictions supplied by Definition 370.3 together with the local decision exhaustion of Definition 370.4.*

Proof. Inspect the adjacency row O in Definition 370.1. An accepted neighboring pair is recorded only as zero-zero, zero-one, or one-zero, and Definition 370.4 rules out any one-one acceptance route or external counting row.

Now restrict along a displayed prefix or fold readback. By Definition 370.3, the restricted packet keeps only the corresponding subrows of W, Z, O, P, L. Applying Definition 370.4 to that restricted ledger shows that every retained adjacency decision is still one of the same three local rows. □

Theorem 370.12 (GoldenMeanShift finite-window standard bridge)**.** *The finite* GoldenMeanShift$^{\uparrow}$ *packet of Definition 370.1 is the standard finite golden-mean shift word interface: a public word is accepted exactly by the finite* BHist *bit-window row, the all-zero inhabitant row, the zero-zero/zero-one/one-zero adjacency ledger, the prefix and fold provenance rows, and the componentwise* hsame *and* NameCert *classifier rows. Componentwise* hsame *supplies the word comparison, and adjacency-ledger exactness supplies the complete no-adjacent-one acceptance record.*

This bridge exports only the finite-window word interface. It does not close entropy, Fibonacci counting, inverse limits, host symbolic dynamics, or an infinite shift-space presentation.

Lean4 ✓ `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftUp StdBridge`

Proof. Project the carrier rows W, Z, O, P, L. They are exactly the finite bit window, all-zero witness, adjacency ledger, provenance row, and local naming ledger of the public word packet.

Use Definition 370.10 for componentwise word comparison. The classifier compares visible bit rows by hsame and transports the same finite packet rows.

Use Definition 370.8 for the acceptance ledger. A neighboring pair is accepted precisely when O records zero-zero, zero-one, or one-zero, so the no-adjacent-one condition is exhausted by the finite ledger.

The bridge therefore identifies the BEDC packet with the standard finite word surface. The excluded symbolic-dynamics data have no carrier coordinate, no classifier row, and no ledger row in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GoldenMeanShift}^{\uparrow}}$.

Constructive story: A golden-mean finite window is generated from binary BHist rows together with an adjacency ledger excluding one-one neighbors, an all-zero inhabitant, prefix and fold provenance, and componentwise hsame classifier rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GoldenMeanShift}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped surface is the finite no-adjacent-one BHist carrier row, componentwise hsame classifier row, exhausted adjacency ledger row, local stability row, fold-prefix closure row, and NameCert obligation surface over Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GoldenMeanShift}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.GoldenMeanShiftUp.GoldenMeanShiftCarrier\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close Fibonacci counting, Zeckendorf arithmetic, entropy, inverse limits, or host symbolic-dynamics theorems.

Upgrade path: No further upgrade remains because the chapter is at the top of the formal axis.

Chapter 371

A Concrete Naming Certificate for $\mathsf{RegularCauchyName}^{\uparrow}$

The $\mathsf{RegularCauchyName}^{\uparrow}$ interface packages the shared boundary between $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as a concrete naming certificate. Following Bishop and Bridges, *Constructive Analysis*, 1985, the packet treats a real name as visible regular Cauchy data with dyadic-radius control, finite stream windows, and a consumer-facing real seal. It never classifies by quotient Cauchy classes, host sequence equality, selected limits, or ambient real completeness.

Its dependency list is $\mathsf{DyadicRatCore}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$.

The finite-window schedule is consumed through chapter 126, while the dyadic-radius observation ledger is consumed through chapter 291. These sibling routes are the only entry points by which the carrier exposes scheduled readback and rational-radius evidence before the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers can read the packet.

Definition 371.1 (RegularCauchyName NameCert source)**.** *A* $\mathsf{RegularCauchyName}^{\uparrow}$ *source packet is a finite* BHist *row*

$$G = (S, Q, \rho, L, E, P, N)$$

with the following visible coordinates:

1. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule of certified unary indices and finite observation windows;*
2. *Q assigns to each inspected window a* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational observation, read as a* $\mathsf{RegSeqRat}^{\uparrow}$ *point row;*
3. *ρ records the dyadic-radius bounds visible at the certified unary indices;*
4. *L is the finite-window regularity ledger that pairs each scheduled window with its radius row and pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison data;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal ledger consumed by downstream real-name readers;*
6. *P is the* Pkg *provenance naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and the ambient* $\mathsf{NameCert}$ *source certificates;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance row.*

The classifier compares two source packets only by common $\mathsf{StreamName}^{\uparrow}$ *windows, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, the finite regularity ledger, and the displayed* $\mathsf{Real}^{\uparrow}$ *seal rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameTasteGate single carrier alignment`

Theorem 371.2 (RegularCauchyName dyadic radius obligations)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet carries, for each certified unary index, a visible dyadic-radius row whose transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations bound the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *point rows inside the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window. The classifier can therefore compare regularity only through displayed radius rows and pointwise rational observations.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier dyadic radius obligations`

Proof. Project the source packet of Definition 371.1. The schedule coordinate names the certified unary index and finite window, while the point row supplies the inspected dyadic rational observation.

The radius coordinate is stored beside that same window in the regularity ledger. Transport along hsame and Cont therefore moves the index, the window, the radius row, and the point observation together.

Since the classifier is defined only from these displayed rows, regularity comparison is a dyadic-radius obligation on the finite ledger. It is not a quotient comparison of Cauchy classes or an equality of host sequences. □

Theorem 371.3 (RegularCauchyName StreamName handoff)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet and every consumer precision request, the requested read factors through a common* $\mathsf{StreamName}^{\uparrow}$ *window, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row attached to that window, and the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier. No read can bypass the finite schedule to inspect a host sequence.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNamePacket streamname handoff`

Proof. Start with a consumer request and read the packet coordinates. The $\mathsf{StreamName}^{\uparrow}$ schedule is the only coordinate that selects windows, so the request first lands in a displayed finite window.

Inside that window, the observation row is a $\mathsf{DyadicRatCore}^{\uparrow}$ rational row presented through the $\mathsf{RegSeqRat}^{\uparrow}$ pointwise classifier. The permitted transports are exactly the hsame comparisons and Cont routes recorded by the packet provenance.

Thus the handoff from stream name to regular rational row is finite and visible. A downstream reader receives the scheduled window and its transported rational evidence, not a hidden sequence object. □

Theorem 371.4 (RegularCauchyName RealUp seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal ledger of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet is consumed only after the StreamName window, dyadic-radius row, and pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed. The seal exports a real-name interface but does not export quotient stream equality, a choice-selected limit, or a host real carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier realup seal boundary`

Proof. Read the seal coordinate through the source packet. It is stored after the schedule, rational observation rows, radius rows, and finite regularity ledger, so a consumer reaches it only along those displayed coordinates.

The preceding handoff theorem identifies the exact route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$. The seal ledger records the $\mathsf{Real}^{\uparrow}$ consumption of that route rather than an independent completion object.

Consequently the boundary of the seal is a finite real-name surface. It withholds quotient stream equality, selected limits, host real carriers, and ambient completeness claims. □

Theorem 371.5 (RegularCauchyName NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *sketch has five rows:*

1. *carrier: the source packet of Definition 371.1 stores the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, dyadic-radius rows, finite regularity ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* Pkg *provenance, and local certificate row;*
2. *classifier: accepted packets are compared only by common stream windows, transported radius rows, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *comparisons, seal rows, and provenance;*
3. *dyadic-radius stability: Definition 371.2 keeps every regularity claim attached to visible dyadic bounds at certified unary indices;*
4. *finite-window ledger: Definition 371.3 forces every read through a scheduled finite window and its transported rational observation;*
5. *RealUp seal consumption: Definition 371.4 exposes only the* $\mathsf{Real}^{\uparrow}$ *consumer surface and no quotient, host-sequence, selected-limit, or ambient-completeness datum.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier semantic name certificate`

Proof. The carrier and classifier rows are immediate projections of the source packet. They list exactly the finite certificate coordinates and the visible comparisons used by the local classifier.

The dyadic-radius row is supplied by Definition 371.2. It pins regularity to transported $\mathsf{DyadicRatCore}^{\uparrow}$ radius evidence at certified unary indices.

The finite-window and seal rows are supplied by Definition 371.3 and Definition 371.4. Together they show that every consumer read passes through a scheduled window before reaching the $\mathsf{Real}^{\uparrow}$ seal, with no escape to quotient, host-sequence, selected-limit, or ambient-completeness data. □

Theorem 371.6 (RegularCauchyName obligation closure package)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose source is the packet of Definition 371.1 admits the five obligation rows needed for obligation closure:*

1. *dyadic-radius stability over transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows;*
2. *finite-window handoff through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule;*
3. *pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison for the inspected rational observations;*
4. *the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the scheduled stream and regular rational rows have been displayed;*
5. *ledger exhaustion over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Thus the packet closes the local obligation surface without adding quotient streams, host sequences, selected limits, or ambient real completeness.

Proof. Project the carrier and classifier rows from the source packet. The carrier supplies the schedule, rational observations, radius rows, finite regularity ledger, seal ledger, package provenance, and local certificate row. The classifier restricts comparison to common windows, transported radius evidence, pointwise rational rows, seal rows, and provenance.

The first obligation row is Definition 371.2. It attaches every regularity claim to visible dyadic bounds at the certified unary indices.

The second and third rows are obtained from Definition 371.3. A consumer precision request first factors through the finite $\mathsf{StreamName}^{\uparrow}$ window, then reads the transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation through the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

The fourth row is Definition 371.4. The $\mathsf{Real}^{\uparrow}$ seal is reachable only after the displayed stream, dyadic-radius, and regular rational rows have been consumed.

Finally, Definition 371.5 assembles these rows with the carrier and classifier projections. Since every listed datum is drawn from the displayed packet and its hsame, Cont, Pkg, and NameCert provenance, the ledger is exhausted over the named BEDC objects and contains no quotient stream, host sequence, selected-limit, or ambient-completeness row. □

Theorem 371.7 (RegularCauchyName classifier transport stability)**.** *Let $G = (S, Q, \rho, L, E, P, N)$ be the source packet of Definition 371.1. Any displayed* hsame *comparison and* Cont *route that transports the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row Q, the dyadic-radius row ρ, the finite ledger L, the* $\mathsf{Real}^{\uparrow}$ *seal row E, the* Pkg *provenance P, and the local* NameCert *row N preserves the* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *classifier over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier classifier transport stability`

Proof. Project the classifier from the source packet. Its only comparison rows are common $\mathsf{StreamName}^{\uparrow}$ windows, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observations, dyadic-radius rows, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ reads, seal rows, and provenance.

Apply the dyadic-radius obligation of Definition 371.2 to keep each radius bound attached to the same certified unary index and window during transport. Apply the StreamName handoff of Definition 371.3 to keep the scheduled window and transported rational observation on the same route.

The $\mathsf{Real}^{\uparrow}$ seal boundary in Definition 371.4 is downstream of those rows, so transporting the displayed coordinates also transports the seal-facing classifier row. The package theorem Definition 371.6 then repacks the transported rows as the same local NameCert classifier, with no host sequence, quotient stream, selected limit, or ambient completeness coordinate. □

Theorem 371.8 (RegularCauchyName dyadic radius transport composition)**.** *Let $G = (S, Q, \rho, L, E, P, N)$ be the source packet of Definition 371.1. Any componentwise* hsame *comparison and displayed* Cont *transport that preserves the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row Q, the dyadic-radius row ρ, the finite ledger L, the* $\mathsf{Real}^{\uparrow}$ *seal row E, the* Pkg *provenance P, and the local* NameCert *row N preserves the single dyadic-radius row consumed both by the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and by the* $\mathsf{Real}^{\uparrow}$ *seal factorization. The transported consumer route factors through the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows, with no quotient stream, host sequence equality, selected limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier dyadic radius transport composition`

Proof. Begin with the radius coordinate in the source packet. By Definition 371.2, the row ρ is attached to the certified unary index and to the visible dyadic bound used by the regularity obligation.

Use Definition 371.3 to place the consumer request on a finite $\mathsf{StreamName}^{\uparrow}$ window. On that window the same handoff exposes the transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation Q before the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier reads it.

Now apply Definition 371.7. The displayed hsame and Cont route transports S, Q, ρ, L, E, P, and N together, so the classifier-facing radius row cannot split from the finite ledger route.

Finally use Definition 371.10. The $\mathsf{Real}^{\uparrow}$ seal consumes the route only after the same dyadic-radius row has been displayed beside the scheduled window and rational readback. Thus the classifier consumer and the seal consumer share one finite BEDC route over BHist, hsame, Cont, Pkg, and NameCert, and no quotient stream, host-sequence equality, selected limit, or ambient completeness coordinate is introduced. □

Theorem 371.9 (RegularCauchyName window ledger exhaustion)**.** *For a source packet $G = (S, Q, \rho, L, E, P, N)$, every consumer precision read is exhausted by the displayed finite route through S, Q, ρ, L, E, P, and N. The route uses only* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNamePacket handoff nonempty iff`

Proof. Begin with the finite-window handoff. By Definition 371.3, a precision request first lands in the displayed $\mathsf{StreamName}^{\uparrow}$ schedule and then reaches the transported dyadic rational observation read through the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

The radius row for that same window is supplied by Definition 371.2. Thus the scheduled window, dyadic observation, radius evidence, and regular rational readback are one finite ledger route, not separate hidden data.

The $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 371.4 consumes this route after it has been displayed. Together with the Pkg and NameCert provenance rows in Definition 371.6, these entries exhaust the consumer ledger and leave no quotient stream, host sequence, selected limit, or ambient completeness row to read. □

Theorem 371.10 (RegularCauchyName real seal factorization)**.** *For every accepted source packet $G = (S, Q, \rho, L, E, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal row E factors through the displayed dyadic-radius row, the* $\mathsf{StreamName}^{\uparrow}$ *handoff, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before it can be consumed as a* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *row. The factorization is componentwise over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier real seal factorization`

Proof. Read the seal coordinate from the source packet. It is recorded after the stream schedule, dyadic observations, radius bounds, and finite regularity ledger, so it is not an independent real carrier.

Use Definition 371.3 to expose the scheduled finite window and its transported rational observation. Use Definition 371.2 to attach the dyadic-radius row to the same window, and use Definition 371.4 to place the $\mathsf{Real}^{\uparrow}$ seal after that handoff.

Finally apply Definition 371.6. The package theorem names the same BHist, hsame, Cont, Pkg, and NameCert provenance, so the seal factorization remains a finite certificate route and exports no quotient equality, selected limit, host real carrier, or ambient completeness theorem. □

Theorem 371.11 (RegularCauchyName tail selector factorization)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet read selected by a* $\mathsf{StreamName}^{\uparrow}$ *tail-compatible finite window factors through the schedule S, dyadic observation row Q, radius row ρ, ledger L,* $\mathsf{Real}^{\uparrow}$ *seal E, package provenance P, and local* NameCert *row N. The factorization is a finite route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The selected read has no quotient stream coordinate, no host sequence equality, no selected limit, and no ambient completeness coordinate.

Proof. Begin with the StreamName handoff theorem Definition 371.3. The tail-compatible window first exposes the schedule S and the transported dyadic observation row Q before any pointwise $\mathsf{RegSeqRat}^{\uparrow}$ readback is consumed.

Attach the radius and ledger rows by Definition 371.2 and Definition 371.9. These rows place ρ and L on the same selected finite window with the same rational readback and support data.

Next use Definition 371.4. It puts the $\mathsf{Real}^{\uparrow}$ seal E downstream of the displayed schedule, dyadic observation, radius, and ledger rows rather than beside them as an independent endpoint.

Finally apply Definition 371.10. Repacking S, Q, ρ, L, E, P, N with the displayed BHist, hsame, Cont, Pkg, and NameCert rows gives the asserted tail-selector route and leaves quotient streams, host sequence equality, selected limits, and ambient completeness outside the packet. □

Theorem 371.12 (RegularCauchyName common-window classifier scope)**.** *Two accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packets can be compared only after choosing a displayed common finite* $\mathsf{StreamName}^{\uparrow}$ *window. On that same window, the comparison transports the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows and dyadic-radius rows by the recorded* hsame *and* Cont *routes, then reads the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier for the transported rational observations. No classifier step may compare host sequences, quotient streams, selected limits, or unexhibited real carriers.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNamePacket common window classifier scope`

Proof. Project the two source packets of Definition 371.1. Their classifier coordinates name only finite stream windows, transported rational observations, radius rows, seal rows, and provenance.

Use Definition 371.3 to put a consumer comparison into a common displayed $\mathsf{StreamName}^{\uparrow}$ window. The allowed transports are the packet's hsame and Cont rows, so the window, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation, and the dyadic-radius row move together.

On the transported window, the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier is the only rational comparison exported by the packet. The package closure of Definition 371.6 confirms that no hidden quotient, host sequence, selected limit, or completeness datum is available to the classifier. □

Theorem 371.13 (RegularCauchyName common-window seal determinacy)**.** *Once two accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packets are compared on a displayed common* $\mathsf{StreamName}^{\uparrow}$ *window, their* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise classifier rows, finite ledger rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows are determined by that same common-window route. The determinacy is certificate-local: it factors through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows and excludes host sequences, quotient streams, selected limits, and ambient completeness.*

Proof. First put the comparison on the shared finite window. By Definition 371.3, every precision read reaches the displayed $\mathsf{StreamName}^{\uparrow}$ route before it can expose dyadic observations or regular rational readback.

Next attach the seal boundary. The $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 371.4 is downstream of the same finite window, and the NameCert obligation surface of Definition 371.5 keeps the dyadic radius rows, pointwise classifier rows, ledger rows, package provenance, and local certificate in the same finite packet.

Finally apply Definition 371.12. It restricts comparison to the transported common-window rows, so two packets that are compared there cannot vary their radius, classifier, ledger, or seal by appealing to host sequences, quotient streams, selected limits, or ambient completeness. □

Theorem 371.14 (RegularCauchyName finite-window ledger exhaustion)**.** *Every consumer precision request for an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet is exhausted by the displayed schedule row, the dyadic-radius row, the finite regularity ledger, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *row. The request receives no host sequence, quotient stream, selected limit, hidden completeness datum, or additional real carrier beyond those displayed packet rows.*

Proof. Project the packet of Definition 371.1. A precision request first reads the schedule coordinate, which selects a certified unary index and a finite $\mathsf{StreamName}^{\uparrow}$ observation window.

The dyadic-radius coordinate and the regularity ledger sit beside that window. By Definition 371.3, the consumer read follows the scheduled window to the transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

The Pkg row records the dependency provenance, and the local $\mathsf{NameCert}$ row records the certificate provenance for the same finite packet. Applying Definition 371.6 shows that these displayed rows exhaust the ledger available to the consumer. □

Theorem 371.15 (RegularCauchyName RealUp seal consumer scope)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet is reachable only after four finite faces have been displayed: the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row and dyadic-radius row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise classifier with its finite ledger, and the* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance rows. The exported object is only the finite real-name seal surface; it does not export quotient stream equality, a selected limit, host real completeness, an independent real carrier, an ambient completeness datum, or a metric-completeness row.*

Proof. Project the seal coordinate from the packet of Definition 371.1. The seal is stored after the schedule, dyadic observation rows, radius rows, and finite regularity ledger, so it is not an independent entry point.

Use Definition 371.3 to identify the displayed route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier. Then use Definition 371.4 to read the seal only after that route has been consumed.

The package closure in Definition 371.6 keeps the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance attached to the same finite ledger. After those rows are in place, apply Definition 371.10: the seal exposes only the finite real-name consumer row and withholds quotient equality, selected limits, host completeness, any independent real carrier, ambient completeness data, and metric-completeness rows. □

Theorem 371.16 (RegularCauchyName scoped carrier transport)**.** *For accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *source packets* $G = (S, Q, \rho, L, E, P, N)$*, every displayed* hsame *comparison and* Cont *route that transports the coordinates* S*,* Q*,* ρ*,* L*,* E*,* P*, and* N *over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* BHist*,* Pkg*, and* $\mathsf{NameCert}$ *preserves the source-packet carrier and the local classifier. The transported packet is again classified only by common stream windows, transported dyadic observations and radius rows, pointwise regular rational comparisons, the finite ledger, the seal row, and provenance.*

Proof. Project the packet carrier from Definition 371.1. Each coordinate named in the statement is one of the displayed rows of the source packet, so a componentwise hsame comparison or Cont route has no hidden coordinate to move.

The carrier rows are the first two rows of Definition 371.6: the schedule, rational observations, radius rows, ledger, seal, package provenance, and local certificate row are transported as the displayed packet itself.

The classifier row is supplied by the same package theorem and by Definition 371.12. Comparison is restricted to common windows, transported $\mathsf{DyadicRatCore}^{\uparrow}$ rows, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ reads, seal rows, and provenance, so the transported packet remains inside the same scoped carrier. □

Theorem 371.17 (RegularCauchyName finite-window readback exactness)**.** *Every consumer precision read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet factors through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation attached to that window, the displayed dyadic-radius row, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, and the finite ledger row. The readback exposes no host sequence coordinate and no quotient stream coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier finite window readback exactness`

Proof. Begin with the consumer request. By Definition 371.3, the request first lands in a scheduled finite $\mathsf{StreamName}^{\uparrow}$ window and then reads the transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation through the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

The radius evidence for that same window is exactly the row described by Definition 371.2. It is stored beside the observation in the finite regularity ledger, so the readback route keeps the window, observation, radius, and classifier rows together.

Finally apply Definition 371.14. The ledger exhausts the consumer-facing rows available to the request, and therefore excludes host sequence coordinates, quotient stream coordinates, selected limits, and hidden completeness data. □

Theorem 371.18 (RegularCauchyName RealUp seal ledger exhaustion)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet is consumed only after the finite-window row and the dyadic-radius row have been displayed. Its ledger is exhausted by* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *primitives, with no extra real carrier, quotient stream, selected limit, or ambient completeness row.*

Proof. Use Definition 371.4 to locate the seal behind the displayed stream window, dyadic-radius row, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ readback. The seal is a downstream consumer row, not a separate entry point.

The preceding theorem gives the exact readback route: scheduled $\mathsf{StreamName}^{\uparrow}$ window, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation, radius row, pointwise classifier, and ledger. Thus the seal can be consumed only after those rows have appeared.

The ledger exhaustion clause then follows from Definition 371.6 and Definition 371.15. Those results list the named BEDC primitives that account for the packet surface and rule out any additional real carrier, quotient stream, selected limit, or completeness row. □

Theorem 371.19 (RegularCauchyName window-radius-seal scoped package)**.** *For every accepted source packet* $G = (S, Q, \rho, L, E, P, N)$ *of Definition 371.1, the scheduled finite* $\mathsf{StreamName}^{\uparrow}$ *window, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, dyadic-radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, finite ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows form the scoped package consumed by* $\mathsf{RegularCauchyName}^{\uparrow}$. *No host sequence, quotient stream, selected limit, hidden completeness datum, or independent real carrier is available to the consumer.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier window radius seal scoped package`

Proof. First transport the carrier. By Definition 371.16, the displayed hsame comparisons and Cont routes preserve the source rows S, Q, ρ, L, E, P, and N as a single accepted packet over the named BEDC primitives.

Next read the finite window exactly. Apply Definition 371.17 to factor every precision request through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation, dyadic-radius row, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier, and finite ledger.

Finally attach the seal. The exhaustion theorem Definition 371.18 places the $\mathsf{Real}^{\uparrow}$ seal after those displayed rows and accounts for the remaining BHist, hsame, Cont, Pkg, and NameCert primitives. Repacking the rows named in Definition 371.1 gives exactly the scoped package in the statement, with no coordinate for a host sequence, quotient stream, selected limit, hidden completeness datum, or independent real carrier. □

Theorem 371.20 (RegularCauchyName stationary rational seal compatibility)**.** *Let $G = (S, Q, \rho, L, E, P, N)$ be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *source packet. If its displayed* BHist *rows are restricted to a stationary rational window, then the* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 371.19 factor the* $\mathsf{Real}^{\uparrow}$ *seal through the constant-rational exactness row of Definition 364.8. The factorization uses only the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, finite ledger, and visible seal row, and it exports no quotient stream, selected host limit, or ambient completeness datum.*

Proof. Start with the window-radius-seal scoped package of Definition 371.19. It lists the only rows available to a consumer of G: the scheduled window, dyadic observation and radius, pointwise regular rational readback, ledger, seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert evidence.

On the stationary rational subcase, the scheduled regular readback is exactly the constant $\mathsf{RegSeqRat}^{\uparrow}$ member governed by Definition 364.8. The hsame and Cont rows therefore transport the same displayed rational observation into the stationary seal route.

The two packages have the same consumer boundary: finite windows, dyadic observations, regular readback, package provenance, and local naming. Since neither package contains a quotient stream, selected host limit, or ambient completeness coordinate, the stationary rational seal is compatible with the $\mathsf{RegularCauchyName}^{\uparrow}$ seal without adding any extra row. □

Theorem 371.21 (RegularCauchyName diagonal window consumer boundary)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet $G = (S, Q, \rho, L, E, P, N)$, any diagonal completion consumer that reads the displayed window-radius-seal surface must pass through the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows named in Definition 371.19 before it reaches the routing boundary of Definition 372.9. The boundary keeps the consumer on scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite ledger rows, and the* $\mathsf{Real}^{\uparrow}$ *seal, with no quotient stream, selected host limit, or ambient completeness row.*

Proof. Project the accepted packet through the RegularCauchyName package theorem. Every consumer-visible coordinate is already displayed as a finite packet row, so there is no separate diagonal input hidden behind the seal.

Now apply the completion-consumer routing of Definition 372.9. Its diagonal route consumes regular windows through the same kind of scheduled readback and finite routing rows that the RegularCauchyName package exposes.

The handoff is therefore by row identity: scheduled windows match scheduled windows, dyadic observations and radius evidence match the diagonal tolerance boundary, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal remain downstream of BHist, hsame, Cont, Pkg, and NameCert. The

excluded objects are absent from both inventories, so the diagonal consumer cannot escape through them. □

Theorem 371.22 (RegularCauchyName completion handoff non-escape)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *factors through the displayed window-radius-seal package, the stationary rational compatibility row of Definition 371.20 when the packet is constant, and the diagonal completion-consumer boundary of Definition 371.21. Thus the handoff is exhausted by* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, scheduled windows, dyadic observations, regular readback, finite ledger rows, and the seal row, and never by quotient streams, selected host limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier completion handoff non escape`

Proof. First place the packet in the window-radius-seal scoped package. This supplies the visible finite rows that any completion-facing consumer must read before the $\mathsf{Real}^{\uparrow}$ seal can be used.

For constant rational packets, the preceding compatibility theorem identifies the seal route with the stationary rational exactness package. For nonconstant completion reads, the diagonal window theorem routes the consumer through the diagonal completion boundary instead.

Both routes have the same displayed source rows and the same exclusions. Repacking the common BHist, hsame, Cont, Pkg, and NameCert evidence gives a single completion handoff boundary for $\mathsf{RegularCauchyName}^{\uparrow}$, and the carrier has no coordinate from which a quotient stream, selected host limit, or ambient completeness object could be produced. □

Theorem 371.23 (RegularCauchyName common-window transport triad)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, a completion-facing comparison that chooses a common finite* $\mathsf{StreamName}^{\uparrow}$ *window transports three rows as one triad: the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the dyadic-radius row* ρ*, and the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier read attached to that same window. The triad is transported only by the packet's* BHist *entries,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier common window transport triad`

Proof. Project the accepted packet through Definition 371.1. The stream schedule, dyadic observation, radius row, regularity ledger, seal row, package provenance, and local certificate row are displayed as coordinates of the same finite source packet.

Use Definition 371.3 to put the comparison into a common finite $\mathsf{StreamName}^{\uparrow}$ window. Then use Definition 371.2 and Definition 371.12 to keep the $\mathsf{DyadicRatCore}^{\uparrow}$ observation, radius row, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier on that same transported window.

The permitted transport rows are exactly the packet's BHist, hsame, Cont, Pkg, and NameCert evidence. Since no packet coordinate names a quotient stream, host sequence, selected limit, or ambient completeness principle, the triad has no route through those objects. □

Theorem 371.24 (RegularCauchyName seal readback exhaustion)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal readback is exhausted by the displayed schedule row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* Q*, the dyadic-radius row* ρ*, the finite regularity ledger* L*, the package row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *row* N*. No readback step can obtain a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier seal readback exhaustion`

Proof. Start from the seal coordinate of Definition 371.1. The seal appears after the schedule, observation, radius, and ledger rows, so it is a consumer row reached through displayed finite data.

The finite readback route is Definition 371.17: every precision request factors through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation, dyadic-radius row, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier, and finite ledger. The real seal boundary is then supplied by Definition 371.18.

Finally apply the package accounting of Definition 371.6. The only remaining rows are BHist, hsame, Cont, Pkg, and NameCert provenance, which records the readback without adding quotient streams, host sequences, selected limits, independent real carriers, or ambient completeness. □

Theorem 371.25 (RegularCauchyName downstream completion handoff package)**.** *The downstream package exported by an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet to* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consists of the common-window transport triad of Definition 371.23, the seal readback exhaustion row of Definition 371.24, the* $\mathsf{StreamName}^{\uparrow}$ *handoff of Definition 371.3, and the* $\mathsf{Real}^{\uparrow}$ *seal consumer scope of Definition 371.15. This package is finite over* BHist, hsame, Cont, Pkg, *and* NameCert *and exports no quotient streams, host sequences, selected limits, independent real carriers, or ambient completeness.*

Proof. First assemble the finite window surface. The stream handoff chooses the displayed $\mathsf{StreamName}^{\uparrow}$ window, and the common-window transport triad keeps the $\mathsf{DyadicRatCore}^{\uparrow}$ observation, dyadic-radius row, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier attached to that window.

Next attach the real-name boundary. The seal readback exhaustion theorem accounts for the schedule, observation, radius, finite ledger, package row, and local certificate row; the seal consumer scope theorem identifies the resulting $\mathsf{Real}^{\uparrow}$ row as downstream of those displayed data.

The three consumers named in the statement read exactly this surface: subsequence consumers read finite windows, diagonal consumers read the transported diagonal boundary, and completion consumers read the real seal after the finite ledger. Repacking the shared BHist, hsame, Cont, Pkg, and NameCert rows gives one handoff package and leaves no coordinate for quotient streams, host sequences, selected limits, independent real carriers, or ambient completeness. □

Theorem 371.26 (RegularCauchyName bridge source surface)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *of Definition 371.1, the rows displayed by* G *supply the bridge source surface from* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *into the completion-facing consumer surface. The source surface consists exactly of the scheduled stream window* S, *dyadic observation row* Q, *dyadic-radius row* ρ, *finite regularity ledger* L, $\mathsf{Real}^{\uparrow}$ *seal* E, *package row* P, *local* NameCert *row* N, *and the ambient* BHist, hsame, Cont, *and* Pkg *transport rows already named by the window-radius-seal package.*

Proof. Start from the window-radius-seal package of Definition 371.19. It displays the source packet as a finite row package over the scheduled stream window, dyadic observation, radius evidence, pointwise regular rational readback, ledger, real seal, package provenance, and local certificate.

The finite-window route is exact by Definition 371.17. Thus the rows read by a completion-facing consumer are not reconstructed from a host sequence; they are the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows of the packet.

Finally use Definition 371.18 and Definition 371.15. These place the $\mathsf{Real}^{\uparrow}$ seal after the finite rows and identify the consumer-facing boundary as the same BHist, hsame, Cont, Pkg, and NameCert surface named in the statement. □

Theorem 371.27 (RegularCauchyName bridge consumer factorization)**.** *Every downstream* RealCauchyCompletion *or* RegularCauchyFamily$^{\uparrow}$ *consumer read of an accepted* RegularCauchyName$^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *factors through the displayed stream window, dyadic observation, radius row, pointwise* RegSeqRat$^{\uparrow}$ *classifier, finite ledger,* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The factorization has no additional completion datum before those rows and no independent consumer coordinate beyond them.*

Proof. First factor the read through the scoped package of Definition 371.19. Any downstream consumer begins with the same accepted packet rows, so its visible route must pass through S, Q, ρ, L, E, P, and N.

Next apply Definition 371.17. The stream window, dyadic observation, radius row, pointwise classifier, and ledger are consumed together; this gives the common finite part of the factorization for both RealCauchyCompletion$^{\uparrow}$ and RegularCauchyFamily$^{\uparrow}$ reads.

The remaining real-name step is governed by Definition 371.18 and Definition 371.15. They attach the Real$^{\uparrow}$ seal only after the finite rows and account for the BHist, hsame, Cont, Pkg, and NameCert evidence, so no further completion coordinate is available. □

Theorem 371.28 (RegularCauchyName bridge non-escape boundary)**.** *The bridge-facing surface exported by an accepted* RegularCauchyName$^{\uparrow}$ *packet exports no host sequence, quotient stream, selected limit, hidden completeness datum, or independent real carrier. Any consumer-side route to* RealCauchyCompletion$^{\uparrow}$ *or* RegularCauchyFamily$^{\uparrow}$ *is bounded by the rows of Definition 371.26 and the factorization of Definition 371.27.*

Proof. The exclusion already appears in the scoped package: Definition 371.19 lists the available rows and states that host sequences, quotient streams, selected limits, hidden completeness data, and independent real carriers are not coordinates of the package.

Exact readback preserves that boundary. By Definition 371.17, a consumer precision request can only read the scheduled window, dyadic observation, radius row, pointwise classifier, and ledger before reaching the seal.

Finally, Definition 371.18 and Definition 371.15 exhaust the Real$^{\uparrow}$ side of the route through the displayed ledger and seal rows. Since the packet has no remaining exported coordinate, the bridge-facing boundary cannot escape to any of the excluded objects. □

Theorem 371.29 (RegularCauchyName schedule-radius lock)**.** *For every accepted* RegularCauchyName$^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, a common displayed* StreamName$^{\uparrow}$ *window locks together the schedule row* S*, the dyadic-radius row* ρ*, the transported* DyadicRatCore$^{\uparrow}$ *observation row* Q*, and the pointwise* RegSeqRat$^{\uparrow}$ *readback carried by* L*. The lock is witnessed only by the packet's* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows and has no coordinate for quotient streams, host sequences, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier schedule radius lock`

Proof. First place the packet into the common StreamName$^{\uparrow}$ window supplied by Definition 371.3. This fixes the schedule row S as the window that every downstream read must inspect.

Next apply the dyadic-radius obligations of Definition 371.2. They attach the radius row ρ to the same displayed window and keep the tolerance request inside the finite BEDC packet.

Finally use the transport triad of Definition 371.23. It carries the DyadicRatCore$^{\uparrow}$ observation row Q and the pointwise RegSeqRat$^{\uparrow}$ readback along the already fixed window. Repacking these projections gives the lock over BHist, hsame, Cont, Pkg, and NameCert, with no remaining route to any excluded completion object. □

Theorem 371.30 (RegularCauchyName constant seal embedding)**.** *If an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *is stationary rational, then its downstream* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed schedule-radius-seal route and the stationary rational exactness row. The factorization uses the constant* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, the trivial dyadic-radius ledger, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it does not select a limit or invoke ambient real completeness.*

Proof. Begin with the displayed schedule-radius-seal package of Definition 371.19. In the stationary rational case, the schedule row repeatedly reads the same rational observation, while the radius and regularity ledgers are the visible constant rows of the packet.

The stationary rational compatibility theorem Definition 371.20 identifies this constant readback with the existing rational exactness route. Thus the seal is reached through the displayed packet rows rather than through a selected completion limit.

Repack the constant observation, radius ledger, pointwise readback, seal row, package row, and local certificate row. The resulting embedding is a projection of the accepted packet, so it remains over BHist, hsame, Cont, Pkg, and NameCert and has no host sequence, quotient stream, selected limit, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier constant seal embedding`

Theorem 371.31 (RegularCauchyName finite consumer choice-free totality)**.** *Every downstream completion-facing consumer read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *is exhausted by the displayed rows* S*,* Q*,* ρ*,* L*,* E*,* P*, and* N*, together with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The consumer route is total over this finite packet and choice-free: it never needs a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First read the consumer through the bridge source surface of Definition 371.26. This exposes exactly the scheduled stream window, dyadic observation, radius row, finite regularity ledger, seal row, package provenance, local certificate, and ambient transport rows.

Next apply the bridge consumer factorization of Definition 371.27. Any $\mathsf{RealCauchyCompletion}^{\uparrow}$ or $\mathsf{RegularCauchyFamily}^{\uparrow}$ read factors through those displayed rows before it can consume the $\mathsf{Real}^{\uparrow}$ seal.

The non-escape boundary of Definition 371.28 then rules out every excluded completion object. Since each consumer read projects one of the displayed rows or one of their BHist, hsame, Cont, Pkg, and NameCert witnesses, the route is total over the finite packet and requires no choice principle. □

Theorem 371.32 (RegularCauchyName public RealUp seal export)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, the public* $\mathsf{Real}^{\uparrow}$ *seal exported by* G *is reached through the existing* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* Q*, the dyadic-radius row* ρ*, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier readback inside* L*, the finite ledger* L*, the package row* P*, and the local* NameCert *row* N*. The export surface contains no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First read the source surface through Definition 371.26. It lists exactly the scheduled stream window, dyadic observation, radius row, finite regularity ledger, $\mathsf{Real}^{\uparrow}$ seal, package provenance, local certificate, and the ambient BHist, hsame, Cont, and Pkg transports.

Next use Definition 371.17. That theorem identifies every finite precision read with the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, dyadic-radius, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ rows, so the seal cannot be read through a host sequence or quotient stream.

Finally apply Definition 371.18 and Definition 371.15. They place the $\mathsf{Real}^{\uparrow}$ seal after the finite ledger and local certificate rows, preserving the exclusions named in the statement. □

Theorem 371.33 (RegularCauchyName completion consumer coverage)**.** *Every* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer route out of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet factors through the public seal surface of Definition 371.32. No consumer in these four families receives an additional quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness datum.*

Proof. Begin with the downstream completion handoff package of Definition 371.25. It binds completion-facing reads to the schedule, dyadic observation, radius, classifier, ledger, seal, package, and certificate rows of the accepted packet.

For subsequence and diagonal consumers, use Definition 371.21 together with Definition 371.23. These theorems keep the consumer read inside the same finite window and transported pointwise classifier rows.

For family and completion consumers, apply Definition 371.27. The factorization reaches the $\mathsf{Real}^{\uparrow}$ seal only through the public surface of Definition 371.32, and Definition 371.28 excludes every omitted completion object. □

Theorem 371.34 (RegularCauchyName common-window refinement)**.** *Let* $G = (S, Q, \rho, L, E, P, N)$ *and* $G' = (S', Q', \rho', L', E', P', N')$ *be two completion-facing reads of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet through one displayed* $\mathsf{StreamName}^{\uparrow}$ *window. If the reads share that common window, then they refine to the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, dyadic-radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier,* BHist *ledger,* hsame *transport,* Cont *route,* Pkg *provenance, and* $\mathsf{NameCert}$ *route. The refinement contains no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier common window refinement`

Proof. First put both reads into the common transport surface. The common-window transport triad of Definition 371.23 identifies the shared $\mathsf{StreamName}^{\uparrow}$ window as the only window through which the $\mathsf{DyadicRatCore}^{\uparrow}$ observation row and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier may be read.

Next attach the seal-readback row. By Definition 371.24, any read that reaches the $\mathsf{Real}^{\uparrow}$ side has already passed through the displayed window, observation, radius, classifier, ledger, package, and local certificate rows.

Finally use the public export and consumer coverage rows. The public seal export of Definition 371.32 keeps the $\mathsf{Real}^{\uparrow}$ seal on the same displayed packet, and Definition 371.33 says that the completion-facing consumers receive no additional route outside that packet. Thus the two reads refine to the same listed rows. □

Theorem 371.35 (RegularCauchyName synchronized precision pullback)**.** *Let* $G = (S, Q, \rho, L, E, P, N)$ *be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let* p *and* p' *be two displayed consumer precision requests. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *schedule admits one common finite refinement window* W *for both requests, and suppose both requests route through the* $\mathsf{CauchyRate}^{\uparrow}$ *schedule and tail-bound surface before any regular rational readback is consumed. Then the normalized readback packets for* p *and* p' *factor through the same window* W*, the same transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the same dyadic-radius row, the same pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the same* $\mathsf{Real}^{\uparrow}$ *seal*

ledger. No quotient stream, host sequence equality, selected limit, or ambient completeness datum participates in this pullback.

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameCarrier synchronized precision pullback`

Proof. Apply the common-window refinement theorem Definition 371.34 to align the two precision reads inside the displayed $\mathsf{StreamName}^{\uparrow}$ window. This identifies the transported dyadic observation, radius, classifier, ledger, transport, continuation, package, and naming rows as one finite comparison surface.

Next use the FiniteWindowEnvelope common-window lock Definition 365.14. The lock keeps the envelope read on the selected window W and prevents the dyadic ledger, regular-sequence seal, and endpoint row from being read through a second window.

Finally route both reads through the CauchyRate surface. Scheduled-window totality Definition 415.7 supplies the displayed schedule row, while tail-bound routing Definition 415.16 forces the regular rational readback through the schedule, tolerance, and source rows before the $\mathsf{RegSeqRat}^{\uparrow}$ row is available. The normalized readback packet of Definition 371.42 then exhausts the remaining $\mathsf{Real}^{\uparrow}$ seal ledger, so the two reads share the stated pullback and have no route through the excluded completion objects. □

Theorem 371.36 (RegularCauchyName dyadic-radius readback)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, the dyadic-radius row* ρ *is the exact finite readback surface consumed by* $\mathsf{RegularCauchySubsequence}^{\uparrow}$*,* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$*, and* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *after they enter the common* $\mathsf{StreamName}^{\uparrow}$ *window. These consumers read the window* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation* Q*, the radius row* ρ*, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier in* L*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes; they do not receive a host sequence, quotient stream, selected limit, independent real carrier, or ambient completeness datum.*

Proof. Begin with the common-window transport triad. Definition 371.23 places subsequence, diagonal, and family reads on the same displayed $\mathsf{StreamName}^{\uparrow}$ window and carries the dyadic observation and pointwise classifier along that window.

The radius row is then forced by the seal-readback route. Definition 371.24 exhausts the precision read before the $\mathsf{Real}^{\uparrow}$ seal by the displayed window, observation, radius, classifier, ledger, package, and local certificate rows.

The consumer side is exactly the one recorded by Definition 371.33. Combining that coverage with the public seal export of Definition 371.32 leaves the dyadic-radius row as the finite readback surface shared by the named completion consumers. □

Theorem 371.37 (RegularCauchyName real seal exactness)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, the public* $\mathsf{Real}^{\uparrow}$ *seal* E *is reached only after the common-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, dyadic-radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row,* BHist *ledger,* hsame *transport,* Cont *route,* Pkg *provenance, and* NameCert *route have been displayed. The seal exports no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First route the packet through its common window. The transport triad of Definition 371.23 supplies the displayed window, dyadic observation, radius-facing readback, and pointwise classifier rows before any seal-facing consumer can proceed.

Next apply Definition 371.24. It says that the seal-facing read is exhausted by those finite rows together with the ledger, package, and local naming routes.

The public export theorem Definition 371.32 identifies the $\mathsf{Real}^{\uparrow}$ seal reached by this route, and the completion consumer coverage theorem Definition 371.33 rules out all extra consumer data. Hence the seal is exact with respect to the displayed rows and has no remaining completion-level coordinate. □

Theorem 371.38 (RegularCauchyName mature boundary package)**.** *The* $\mathsf{RegularCauchyName}^{\uparrow}$ *mature boundary package is the finite row package obtained by assembling Definition 371.26, Definition 371.27, Definition 371.28, Definition 371.32, and Definition 371.33. It is a completion-facing package over the L10* $\mathsf{Real}^{\uparrow}$ *and completion spine, but it exports no quotient streams, host sequences, selected limits, independent real carrier, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameUp StdBridge`

Proof. First assemble the bridge rows. The source surface theorem names the finite packet rows, and the consumer factorization theorem forces completion-facing reads through those same rows.

Next attach the boundary exclusions. The bridge non-escape theorem rules out quotient streams, host sequences, selected limits, hidden completeness data, and independent real carriers before any consumer reads the seal.

Finally add the public seal export and consumer coverage theorems. Together they make the same finite surface available to $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers, yielding the mature completion-facing package stated above. □

Theorem 371.39 (RegularCauchyName RationalStream source admission)**.** *Let* $G = (S, Q, \rho, L, E, P, N)$ *be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is supplied by a* $\mathsf{RationalStream}^{\uparrow}$ *packet. Then the admitted source reads only the scheduled* $\mathsf{UnaryHistory}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{Rat}^{\uparrow}$ *observation rows displayed by* Q*, the* hsame *and* Cont *transport routes, and the* Pkg *and* NameCert *provenance rows. The admission exports no quotient stream, selected limit, host completion object, independent real carrier, or ambient completeness datum.*

Proof. First read the source through the displayed schedule. The RationalStream ledger completeness theorem Definition 350.9 fixes the $\mathsf{UnaryHistory}^{\uparrow}$ schedule and the $\mathsf{Rat}^{\uparrow}$ rows before the packet enters the $\mathsf{RegularCauchyName}^{\uparrow}$ window, while the common-window transport triad of Definition 371.23 carries those rows through S and Q.

Next attach the local provenance. The public seal export of Definition 371.32 keeps the source on the same displayed hsame, Cont, Pkg, and NameCert routes, and the completion coverage theorem Definition 371.33 prevents any completion-facing consumer from reading outside those rows.

Thus the RationalStream admission is exactly the visible scheduled source described in the statement, with no quotient, limit, or completeness data added. □

Theorem 371.40 (RegularCauchyName FiniteWindowEnvelope handoff)**.** *Every selected precision request for an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet factors through the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *handoff surface: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal route, the displayed* $\mathsf{Real}^{\uparrow}$ *endpoint row, and the same* BHist, hsame, Cont, Pkg, *and* NameCert *provenance. No request obtains a host sequence, quotient regular stream, selected limit, independent real carrier, or ambient completeness row.*

Proof. Begin with the selected finite precision request. The finite-window readback exactness theorem Definition 371.17 routes the request through the finite $\mathsf{StreamName}^{\uparrow}$ window, the transported

$\mathsf{DyadicRatCore}^{\uparrow}$ observation, the radius row, the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier, and the finite ledger.

The FiniteWindowEnvelope handoff surface is the same finite window with its dyadic ledger and regular-sequence seal route. By Definition 365.3, Definition 371.24, and Definition 371.32, the $\mathsf{Real}^{\uparrow}$ endpoint is reached only after those displayed rows and provenance routes have been read.

Finally, Definition 371.33 rules out all completion-side escapes, so the selected request factors through the finite-window envelope exactly as claimed. □

Theorem 371.41 (RegularCauchyName real-completion consumer triangle)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is supplied by* $\mathsf{RationalStream}^{\uparrow}$*, the completion-facing consumer surface is the triangle formed by Definition 371.39, Definition 371.40, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *downstream handoff row. A completion consumer receives exactly the displayed* $\mathsf{RationalStream}^{\uparrow}$ *source, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *finite window, the* $\mathsf{RegularCauchyName}^{\uparrow}$ *seal, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row, together with the named* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It receives no quotient completion, host representative, selected limit, hidden real carrier, or ambient completeness field.*

Proof. First assemble the source side. The RationalStream source admission theorem gives the visible scheduled source and its unary-history, rational-row, transport, package, and certificate provenance.

Next assemble the finite-window side. The FiniteWindowEnvelope handoff theorem factors the selected precision request through the finite $\mathsf{StreamName}^{\uparrow}$ window, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ seal route, and $\mathsf{Real}^{\uparrow}$ endpoint.

Finally attach the completion consumer. The downstream handoff of Definition 360.20 supplies the completion-facing consumer row. The public seal export of Definition 371.32 identifies the $\mathsf{RegularCauchyName}^{\uparrow}$ seal delivered to that row, and Definition 371.33 states that $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers receive no row beyond that displayed packet. The three sides therefore form the claimed consumer triangle. □

Definition 371.42 (RegularCauchyName normalized window readback)**.** *For an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *and a consumer precision request, the normalized readback packet of* G *is the finite row package*

$$\mathsf{NormRead}(G) = (W, Q_W, \rho_W, C_W, L_W, E_W, H_W, T_W, P_W, N_W)$$

obtained by projecting the common finite $\mathsf{StreamName}^{\uparrow}$ *window* W *from* S*, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation* Q_W *from* Q*, the dyadic-radius row* ρ_W*, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier* C_W *read from* L*, the finite ledger row* L_W*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E_W*, and the displayed provenance rows* H_W*,* T_W*,* P_W*, and* N_W *for* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The packet is normalized when every consumer-facing precision read of* G *reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through these displayed rows.*

Theorem 371.43 (RegularCauchyName window normalization coverage)**.** *Every consumer precision read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *factors through the normalized readback packet of Definition 371.42. The factorization first selects the common finite* $\mathsf{StreamName}^{\uparrow}$ *window, then reads the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, dyadic-radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, finite ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It exports no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. Begin with the finite precision request. The exactness theorem Definition 371.17 forces the request through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, transported $\mathsf{DyadicRatCore}^{\uparrow}$ observation, dyadic-radius row, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier, and finite ledger.

Next attach the seal-facing rows. The seal exhaustion theorem Definition 371.24 and the public export theorem Definition 371.32 place the $\mathsf{Real}^{\uparrow}$ seal after those finite rows and before any completion-facing consumer can proceed.

Finally project the provenance from the window-radius-seal package of Definition 371.19. Its BHist, hsame, Cont, Pkg, and NameCert rows are exactly the remaining coordinates of the normalized readback packet, so the consumer read has no route through any excluded completion object. □

Theorem 371.44 (RegularCauchyName normalized readback seal stability)**.** *Let G and G' be accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packets whose normalized readback packets are related by the displayed* hsame *comparisons and* Cont *routes on the common finite* $\mathsf{StreamName}^{\uparrow}$ *window, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, dyadic-radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, finite ledger, and* BHist, Pkg, *and* NameCert *provenance. Then the transported normalized packets preserve the* $\mathsf{Real}^{\uparrow}$ *seal boundary: a seal read for G is carried to the seal read for G' only through the same normalized rows, and no transport step introduces a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First read both packets through the common-window transport triad of Definition 371.23. The triad transports the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation, dyadic-radius row, and pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier as one finite unit.

The carrier and seal rows are then governed by Definition 371.16 and Definition 371.18. These results keep the finite ledger and $\mathsf{Real}^{\uparrow}$ seal downstream of the same displayed hsame and Cont rows.

Repack the transported coordinates as normalized readback packets. Since the provenance rows are already the BHist, Pkg, and NameCert rows named by the window-radius-seal package, the transported seal boundary remains inside the normalized packet and has no additional completion-facing coordinate. □

Theorem 371.45 (RegularCauchyName root-unblock normalization package)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, the normalized readback packet of Definition 371.42 forms the concrete normalization package for downstream* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumers. The package is assembled from Definition 371.19, Definition 371.17, Definition 371.18, Definition 371.43, and Definition 371.44. It is finite over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *with no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness row.*

Proof. First build the normalized finite surface. The window-radius-seal scoped package supplies the accepted packet rows, and finite-window readback exactness identifies the common $\mathsf{StreamName}^{\uparrow}$ window together with its transported dyadic observation, radius row, pointwise classifier, and finite ledger.

Next attach the consumer and seal boundaries. Real-seal ledger exhaustion places the $\mathsf{Real}^{\uparrow}$ seal after the finite rows, while window normalization coverage says that every consumer precision read factors through the normalized packet before reaching that seal.

Finally use normalized readback seal stability to transport the package across the displayed hsame comparisons and Cont routes. The downstream consumers named in the statement receive

exactly this normalized packet, so the package opens their finite readback obligations without adding quotient streams, host sequences, selected limits, independent real carriers, or ambient completeness. □

Theorem 371.46 (RegularCauchyName route exhaustion)**.** *Every accepted* RegularCauchyName$^{\uparrow}$ *consumer route is exhausted by the normalized readback packet of Definition 371.42, the* RationalStream$^{\uparrow}$ *source admission of Definition 371.39, the* FiniteWindowEnvelope$^{\uparrow}$ *handoff of Definition 371.40, and the* RealCauchyCompletion$^{\uparrow}$ *consumer triangle of Definition 371.41. Thus every route reads only the displayed* StreamName$^{\uparrow}$ *finite window,* DyadicRatCore$^{\uparrow}$ *observation, dyadic-radius row,* RegSeqRat$^{\uparrow}$ *classifier, finite ledger,* Real$^{\uparrow}$ *seal, and* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. No accepted route contains a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameUp concrete to schema`

Proof. Begin with the visible source. The RationalStream source admission theorem restricts the source side to the scheduled UnaryHistory$^{\uparrow}$ row, the StreamName$^{\uparrow}$ schedule, the rational observation rows, and the displayed provenance routes.

Next pass to the finite readback surface. The FiniteWindowEnvelope handoff theorem and the normalized window readback definition identify the same finite StreamName$^{\uparrow}$ window, transported DyadicRatCore$^{\uparrow}$ observation, dyadic-radius row, pointwise RegSeqRat$^{\uparrow}$ classifier, finite ledger, Real$^{\uparrow}$ seal, and BHist-anchored provenance.

Finally attach the consumer boundary. Window normalization coverage, normalized readback seal stability, and the root-unblock normalization package force every downstream read through that packet before it reaches the RealCauchyCompletion$^{\uparrow}$ consumer triangle. The excluded quotient, host, limit, carrier, and completeness rows therefore have no accepted route. □

Theorem 371.47 (RegularCauchyName stream-window refinement cover)**.** *For every accepted* RegularCauchyName$^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$*, any two consumer reads that share a displayed* StreamName$^{\uparrow}$ *finite window refine to one common finite cover. The rows of that cover are exactly the scheduled stream window, transported* DyadicRatCore$^{\uparrow}$ *observation, dyadic-radius row, pointwise* RegSeqRat$^{\uparrow}$ *classifier, finite ledger,* Real$^{\uparrow}$ *seal, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The cover exports no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness row.*

Proof. Start from the window-radius-seal scoped package of Definition 371.19. It lists precisely the accepted packet rows available around the shared StreamName$^{\uparrow}$ window, including the dyadic observation, radius row, regular rational classifier, ledger, seal, and provenance.

The shared finite window is then read exactly. Apply Definition 371.17 to both consumer reads; because the displayed StreamName$^{\uparrow}$ window is the same, their dyadic observation, radius row, pointwise RegSeqRat$^{\uparrow}$ classifier, and finite ledger refine to a single finite window interface. The RegSeqRat handoff already cited in the chapter repacks this same finite interface as the regular rational consumer row.

Finally attach the seal and consumer boundary. The real-seal ledger exhaustion theorem Definition 371.18 places the Real$^{\uparrow}$ seal after the common finite rows, and Definition 371.46 rules out any route through quotient streams, host sequences, selected limits, independent real carriers, or ambient completeness. Thus the two reads share exactly the finite cover named in the statement. □

Theorem 371.48 (RegularCauchyName window source determinacy)**.** *Let two accepted* RegularCauchyName$^{\uparrow}$ *consumer precision reads display the same* StreamName$^{\uparrow}$ *finite window, the same* DyadicRatCore$^{\uparrow}$

observation, the same dyadic-radius row, the same $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, the same finite ledger, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same* BHist, hsame, Cont, Pkg, *and* NameCert *provenance. Then the two reads determine the same* $\mathsf{RationalStream}^{\uparrow}$ *source admission and the same* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *handoff, up to the displayed transport rows. In particular, no separate host sequence, quotient stream, selected limit, independent real carrier, or ambient completeness coordinate can distinguish the two source windows.*

Proof. Read both precision requests through the normalized readback packet. By window normalization coverage, each request reaches its seal only after the displayed $\mathsf{StreamName}^{\uparrow}$ window, dyadic observation, radius row, classifier, ledger, and provenance rows have been fixed.

The equality of the displayed rows then fixes the source side. RationalStream source admission reads no data beyond the scheduled source rows and the BHist, hsame, Cont, Pkg, and NameCert provenance, so the two source admissions coincide along the stated transport rows.

It remains to identify the finite-window handoff. The FiniteWindowEnvelope handoff theorem uses exactly the same window, dyadic ledger, regular-sequence seal route, and $\mathsf{Real}^{\uparrow}$ endpoint. Normalized readback seal stability carries the common seal boundary across the displayed transports, leaving no additional coordinate able to separate the two reads. □

Theorem 371.49 (RegularCauchyName StreamName Real seal factorization)**.** *Every downstream* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet factors first through the* $\mathsf{StreamName}^{\uparrow}$ *finite window and then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal before reaching the* $\mathsf{Real}^{\uparrow}$ *seal. The factorization is carried by the normalized readback packet and its displayed* BHist, hsame, Cont, Pkg, *and* NameCert *provenance, not by a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness row.*

Proof. Start with any named downstream consumer read. The root-unblock normalization package says that the consumer receives the normalized readback packet rather than an external completion object.

Inside that packet, the order of access is fixed. The normalized window readback definition places the $\mathsf{StreamName}^{\uparrow}$ finite window before the transported dyadic observation, radius row, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier, finite ledger, and $\mathsf{Real}^{\uparrow}$ seal; the FiniteWindowEnvelope handoff identifies this middle segment as the finite-window envelope.

The Real seal is therefore reached only after the regular-sequence seal route. The real-completion consumer triangle supplies the downstream consumer row, while normalized readback seal stability preserves the same BHist-anchored transport and package provenance. Hence each named downstream read factors through the StreamName window and RegSeqRat seal before it can read the Real seal. □

Definition 371.50 (RegularCauchyName root dependency manifest)**.** *For an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet* $G = (S, Q, \rho, L, E, P, N)$ *whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$, *the root dependency manifest of* G *is the finite BEDC packet*

$$\mathsf{RootDep}_{\mathsf{RegularCauchyName}^{\uparrow}}(G) = (U, R, S, F, L, W, B, D, A, E, C)$$

whose rows are, respectively, the $\mathsf{UnaryHistory}^{\uparrow}$ *schedule row* U, *the* $\mathsf{RationalStream}^{\uparrow}$ *source row* R, *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S, *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and radius row* F, *the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and seal row* L, *the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *handoff row* W, *the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *consumer row* B, *the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *consumer row* D, *the* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *consumer row* A, *the* $\mathsf{Real}^{\uparrow}$ *seal row* E, *and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row* C.

Its provenance is the displayed finite route

$$\mathsf{Prov}(\mathsf{RootDep}_{\mathsf{RegularCauchyName}^{\uparrow}}(G)) = (\mathsf{BHist}_G, \mathsf{hsame}_G, \mathsf{Cont}_G, \mathsf{Pkg}_G, \mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow},G})$$

where BHist_G *records the finite ledger,* hsame_G *records classifier transport on the listed rows,* Cont_G *records the continuation routes between consumer reads,* Pkg_G *records the package provenance, and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow},G}$ *records the local naming certificate. The manifest contains no row outside this displayed dependency list.*

Theorem 371.51 (RegularCauchyName root dependency manifest exhaustion)**.** *Every downstream read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet by* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *factors through the root dependency manifest of Definition 371.50. In particular, the read reaches its source, finite-window handoff, normalized readback,* $\mathsf{Real}^{\uparrow}$ *seal, and completion consumer row only through the displayed* $\mathsf{UnaryHistory}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of that manifest.*

Proof. Begin with the source side. The RationalStream source admission theorem Definition 371.39 forces the visible source read through the $\mathsf{UnaryHistory}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, rational-observation, transport, package, and certificate rows recorded in the manifest.

Next attach the finite readback side. The FiniteWindowEnvelope handoff theorem Definition 371.40 routes each selected precision request through the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ seal route, $\mathsf{Real}^{\uparrow}$ endpoint row, and the same provenance rows.

Finally add the completion side. The real-completion consumer triangle Definition 371.41 supplies the $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer row, and the root-unblock normalization package Definition 371.45 supplies the downstream $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, and $\mathsf{RegularCauchyFamily}^{\uparrow}$ consumer rows. These four results cover every named downstream read, so each read factors through the manifest. □

Theorem 371.52 (RegularCauchyName root dependency manifest non-escape)**.** *The root dependency manifest of Definition 371.50 exports no quotient stream, host sequence, selected limit, independent real carrier, ambient completeness row, or classical choice row. Any downstream consumer of the manifest reads only the displayed finite rows and provenance routes named in the manifest.*

Proof. The source admission theorem Definition 371.39 already excludes quotient streams, selected limits, host completion objects, independent real carriers, and ambient completeness data from the RationalStream source side.

The finite-window handoff theorem Definition 371.40 repeats the same exclusion for selected precision requests and keeps them on the displayed finite-window and seal rows.

The real-completion consumer triangle Definition 371.41 and the root-unblock normalization package Definition 371.45 extend that boundary to the completion and downstream rows. Since none of the cited packets contains a coordinate for a quotient stream, host sequence, selected limit, independent real carrier, ambient completeness, or classical choice, the assembled manifest has no such export. □

Theorem 371.53 (RegularCauchyName manifest exit descent exactness)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose root dependency manifest is displayed descends to the Phase Real and Completion terminal exit exactly through the* $\mathsf{StreamName}^{\uparrow}$ *finite-section descent gluing row, the four-object*

StreamName$^{\uparrow}$ *route, and the* Real$^{\uparrow}$ *terminal seal manifest exit consumer. The resulting read uses only the displayed* UnaryHistory$^{\uparrow}$, RationalStream$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RegularCauchySubsequence$^{\uparrow}$, RegularCauchyDiagonal$^{\uparrow}$, RegularCauchyFamily$^{\uparrow}$, Real$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 371.50. It uses no quotient stream, host sequence, selected limit, independent real carrier, ambient completeness row, metric completeness row, countable-choice row, or bridge row.*

Proof. Begin with the local manifest. The exhaustion theorem Definition 371.51 says that every downstream RegularCauchyName$^{\uparrow}$ read reaches its source, finite-window handoff, normalized readback, Real$^{\uparrow}$ seal, and completion consumer only through the displayed root dependency manifest.

Now route the StreamName face. Apply Definition 126.594 to identify the finite-section descent row, and then apply Definition 126.500 to read that row through the current four-object Phase Real and Completion route. The visible exit coordinates are therefore the StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ faces already listed in the manifest, with the same BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply the Real terminal consumer exactness theorem Definition 38.830. It identifies the terminal Real$^{\uparrow}$ seal consumer with the manifest exit surface. The non-escape theorem Definition 371.52 removes every non-displayed route, so the descent has no quotient stream, host sequence, selected limit, independent real carrier, ambient or metric completeness witness, countable-choice coordinate, or bridge row. □

Theorem 371.54 (RegularCauchyName Phase Real and Completion exit interface)**.** *Every accepted* RegularCauchyName$^{\uparrow}$ *packet whose root dependency manifest is displayed presents the source-side interface of the four-object Phase Real and Completion exit exactly through* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 371.50. The interface exports no quotient stream, host sequence equality, selected limit, independent real carrier, ambient completeness, metric completeness, countable choice, or bridge row.*

Proof. Start from Definition 371.53. It identifies every accepted manifest-displayed RegularCauchyName$^{\uparrow}$ packet with the terminal exit descent and restricts that descent to the finite rows already recorded in Definition 371.50.

Next read the descent through the four-object route. The StreamName route theorem Definition 126.500 gives the StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ boundary, while Definition 38.726 says that this L10 interface has no additional carrier or support coordinate beyond those four faces and the displayed BEDC support rows.

Finally apply Definition 38.830. It pins the terminal Real$^{\uparrow}$ consumer to the same manifest exit surface. The exactness and exhaustion results together leave only the listed BHist, hsame, Cont, Pkg, and NameCert rows as support, and exclude quotient streams, host sequence equality, selected limits, independent real carriers, ambient or metric completeness, countable choice, and bridge rows. □

Theorem 371.55 (RegularCauchyName window-successor closure)**.** *Let an accepted* RegularCauchyName$^{\uparrow}$ *packet display the source-side Phase Real and Completion interface of Definition 371.54. If a successor finite* StreamName$^{\uparrow}$ *window refines the displayed window through the same root dependency manifest, then the successor read preserves the dyadic-radius row, the* RegSeqRat$^{\uparrow}$ *readback route, and the terminal* Real$^{\uparrow}$ *seal route of the packet. The successor step exports no host sequence, quotient stream, selected limit, independent real carrier, ambient completeness, metric completeness, countable-choice row, or bridge row.*

Proof. Start from the finite-window envelope handoff and the displayed manifest. The handoff keeps the successor request on the same $\mathsf{StreamName}^{\uparrow}$ observation schedule and the same dyadic tolerance ledger.

Apply the common-window classifier scope for $\mathsf{StreamName}^{\uparrow}$ windows. The successor window extends the inspected rows componentwise, so the dyadic-radius row and the $\mathsf{RegSeqRat}^{\uparrow}$ readback are transported rather than replaced.

Finally use the Phase Real exit interface. It pins the successor read to the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces and to the same BHist, hsame, Cont, Pkg, and NameCert support. Since none of those rows contains a host sequence, quotient stream, selected limit, independent real carrier, ambient completeness, metric completeness, countable choice, or bridge coordinate, the successor closure has no such export. □

Theorem 371.56 (RegularCauchyName four-face support minimality)**.** *An accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet with displayed root dependency manifest presents the Phase Real and Completion source-side interface if and only if its public support is exactly the four-face packet* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 371.50. No fifth support coordinate can be adjoined unless it is one of the refused rows: quotient stream, host sequence equality, selected limit, independent real carrier, ambient completeness, metric completeness, countable choice, or bridge row.*

Proof. The forward direction starts from the displayed manifest. By Definition 371.50, the only support rows available at the root are the named source, finite-window, regularity, handoff, consumer, real-seal, completion, provenance, package, and certificate rows. The exhaustion theorem Definition 371.51 says that every downstream read of the accepted packet factors through that list.

The exclusion part is supplied by Definition 371.52. It removes quotient streams, host sequence equality, selected limits, independent real carriers, ambient completeness, metric completeness, countable choice, and bridge data from the exported support. Thus any proposed fifth coordinate is either already one of the displayed rows or belongs to the refused list and cannot serve as additional support.

It remains to identify the exact four-face boundary. The descent exactness theorem Definition 371.53 carries the manifest to the terminal exit, and Definition 371.54 identifies that exit with the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ public interface together with the displayed BEDC support rows. Conversely, those four faces and rows are precisely the support used by the exit-interface theorem, so the same manifest presents the source-side interface. □

Theorem 371.57 (RegularCauchyName terminal exit no extra support)**.** *Any accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet with displayed root dependency manifest that satisfies the four-face support minimality theorem presents the Phase Real and Completion terminal exit with no support beyond* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 371.50. The terminal exit exports no quotient stream equality, host sequence equality, selected limit, independent real carrier, ambient completeness, metric completeness, countable choice, or bridge row.*

Proof. First descend the displayed manifest to the terminal exit. The manifest exit descent exactness theorem Definition 371.53 identifies the accepted packet with the Phase Real and Completion exit read and keeps that read on the displayed root manifest rows.

Next identify the public boundary of that exit. The exit-interface theorem Definition 371.54 names exactly the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces together with the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply the four-face support minimality theorem Definition 371.56 and the root dependency manifest non-escape theorem Definition 371.52. Minimality forbids adding a fifth support coordinate to the four-face interface, while non-escape refuses the quotient stream equality, host sequence equality, selected limit, independent real carrier, ambient or metric completeness, countable-choice, and bridge rows. Hence the terminal exit has exactly the displayed support and no additional row. □

Theorem 371.58 (RegularCauchyName RationalStream completion seal triangle)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*, the* $\mathsf{RationalStream}^{\uparrow}$ *source admission, normalized* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion consumer route determine one finite* BHist *consumer surface. The surface is the root dependency manifest of Definition 371.50 read through the public* $\mathsf{Real}^{\uparrow}$ *seal factorization of* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*. It contains no quotient stream, selected limit, host real equality, independent real carrier, or ambient completeness coordinate.*

Proof. First assemble the source and window sides. The RationalStream source admission theorem Definition 371.39 fixes the scheduled $\mathsf{RationalStream}^{\uparrow}$ source, while the root dependency manifest exhaustion theorem Definition 371.51 places the normalized $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal in the same displayed manifest.

Next attach the completion consumer. The public $\mathsf{Real}^{\uparrow}$ seal factorization for $\mathsf{CauchyCompletionMonad}^{\uparrow}$, Definition 378.5, routes every completion-facing seal read through its finite schedule, window, dyadic observation, diagonal readback, seal, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Along the RegularCauchyName route, those rows are exactly the completion face already recorded by the manifest.

Finally apply the non-escape boundary of Definition 371.52. Since the manifest and the completion-monad seal factorization share the same finite provenance surface and neither exposes a quotient stream, selected limit, host real equality, independent real carrier, or ambient completeness coordinate, the admitted rational source and the completion consumer determine one finite BHist consumer surface. □

Theorem 371.59 (RegularCauchyName Real-completion selector seal compatibility)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. Any selector-facing* $\mathsf{RealCompleteness}^{\uparrow}$ *or* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *consumer that reads the displayed finite window of G, its dyadic radius row, its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and its* $\mathsf{Real}^{\uparrow}$ *seal factors through two displayed routes: the root dependency manifest of Definition 371.50, and the ordered selector route*

$$\mu, \Delta, D, W, Q, S$$

of Definition 450.43. Consequently the selector-presented seal and the $\mathsf{RegularCauchyName}^{\uparrow}$ *finite-window seal expose the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *surface. They expose no quotient stream, selected host limit, host equality, choice row, or ambient completeness theorem.*

Proof. First read the $\mathsf{RegularCauchyName}^{\uparrow}$ side. The root dependency manifest exhaustion theorem Definition 371.51 places every downstream source, finite-window, readback, $\mathsf{Real}^{\uparrow}$ seal, and completion-consumer read of G inside the finite manifest of Definition 371.50.

Next attach the completion-facing seal. The RationalStream completion seal triangle Definition 371.58 identifies the admitted rational source, normalized window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{CauchyCompletionMonad}^{\uparrow}$ consumer route with one finite BHist consumer surface.

Now compare the selector presentation. The selector-window-seal factorization theorem Definition 450.43 orders every selector-facing $\mathsf{RealCompleteness}^{\uparrow}$ read as μ, Δ, D, W, Q, S, with transport, route replay, package provenance, and local naming carried by the same finite seal surface.

Finally apply the non-escape theorem Definition 371.52. Since both routes read only the displayed finite rows and the provenance coordinates BHist, hsame, Cont, Pkg, and NameCert, neither presentation can export a quotient stream, selected host limit, host equality, choice row, or ambient completeness theorem. □

Theorem 371.60 (RegularCauchyName Real-completion handoff spine)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$ *factors all Real-and-Completion consumer reads through the displayed finite route: the* $\mathsf{RationalStream}^{\uparrow}$ *source admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion consumer, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows. This route is the public handoff surface for Real-and-Completion consumers. It exports no quotient stream equality, selected limit, host real equality, choice row, or ambient completeness principle.*

Proof. First locate every consumer read in the root manifest. The root dependency manifest exhaustion theorem Definition 371.51 places the source, finite window, dyadic radius row, pointwise readback, $\mathsf{Real}^{\uparrow}$ seal, completion consumer, and provenance coordinates inside the finite manifest of Definition 371.50.

Next compress the admitted RationalStream route to the completion-facing surface. The RationalStream completion seal triangle Definition 371.58 identifies the $\mathsf{RationalStream}^{\uparrow}$ source admission, normalized $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{CauchyCompletionMonad}^{\uparrow}$ completion consumer as one finite BHist consumer surface.

Then compare this surface with the selector-facing Real-and-Completion presentation. The selector seal compatibility theorem Definition 371.59 carries the same finite route across every selector-facing $\mathsf{RealCompleteness}^{\uparrow}$ or $\mathsf{CauchyCompletionMonad}^{\uparrow}$ read, including the hsame, Cont, Pkg, and NameCert rows.

Finally apply the non-escape boundary of Definition 371.52. Since the manifest, completion seal triangle, and selector-compatible route expose only the displayed finite rows, no Real-and-Completion consumer read can acquire quotient stream equality, a selected limit, host real equality, a choice row, or an ambient completeness principle from this handoff. □

Theorem 371.61 (RegularCauchyName Real-completion prefix determinacy)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. Suppose two Real-and-Completion consumer reads of G use the same* $\mathsf{RationalStream}^{\uparrow}$ *source admission, the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the same pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{Real}^{\uparrow}$ *seal, and the same* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *consumer coordinate. Then both reads factor through the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface of the root dependency manifest of Definition 371.50. Consequently the two reads cannot diverge into quotient stream equality, selected host limits, host real equality, choice rows, or ambient completeness data.*

Proof. First identify the shared finite surface. The RationalStream completion seal triangle Definition 371.58 turns the common source admission, finite window, readback, $\mathsf{Real}^{\uparrow}$ seal, and completion consumer coordinate into one finite BHist consumer surface.

Next compare the selector-facing presentations. The selector seal compatibility theorem Definition 371.59 routes any compatible Real-and-Completion consumer read through that same finite surface and through the displayed hsame, Cont, Pkg, and NameCert coordinates.

Now apply the handoff spine Definition 371.60. Since both reads have the same source, window, radius, readback, seal, and completion coordinate, the public handoff route leaves no second finite prefix through which either read can be separated.

Finally use the root manifest non-escape theorem Definition 371.52. The shared prefix exports only the displayed finite provenance rows, so neither read can branch into quotient stream equality, selected host limits, host real equality, choice rows, or ambient completeness data. □

Theorem 371.62 (RegularCauchyName facewise exit handoff completeness)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. Its Real-and-Completion handoff is complete precisely through the four displayed L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Omitting any one of these faces, or replacing one by quotient stream equality, a selected host limit, host real equality, countable choice, ambient completeness, metric completeness, or a bridge row, lies outside the accepted handoff.*

Proof. First build the displayed handoff surface. The handoff spine Definition 371.60 places every Real-and-Completion consumer read of G on the source admission, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, completion consumer, and provenance rows. The prefix determinacy theorem Definition 371.61 then shows that this visible prefix leaves no second finite surface through which two such reads can diverge.

Next identify the face list with the exit criterion. The tail-budget handoff theorem Definition 371.78 fixes the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ side of the finite tail budget, while the Real L10 facewise status theorem Definition 38.537 records that the displayed Real exit faces are not interchangeable with quotient, choice, bridge, or host-completion substitutes.

Finally apply the root manifest non-escape theorem Definition 371.52. Since the accepted packet exports only the four displayed L10 faces and the listed BHist, hsame, Cont, Pkg, and NameCert support rows, removing a face or replacing it by any refused quotient, limit-selection, equality, choice, completeness, or bridge coordinate leaves the RegularCauchyName handoff rather than completing it. □

Theorem 371.63 (RegularCauchyName Real-completion exit threshold equivalence)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. The packet presents the current Phase Real and Completion exit threshold if and only if its accepted handoff exposes exactly the four local faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, each face carries its own local objectwise exit-threshold record, and the only support rows beyond those faces are the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the RegularCauchyName handoff.*

Equivalently, the threshold is read object by object on the four displayed faces. It is not supplied by quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, an aggregate formalstatus *row, an aggregate bridge row, or a standard-bridge substitute.*

Proof. For the forward direction, apply Definition 371.62. Any accepted Real-and-Completion handoff for G is already complete exactly through the four displayed faces and the BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply Definition 371.78. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ sides of the handoff remain stable under the certified tail-budget route, so the objectwise threshold read cannot be moved to a quotient stream, selected limit, host sequence equality, choice row, ambient completeness witness, or metric-completeness datum.

Use Definition 38.494 to identify the current Phase Real and Completion threshold. That result reads the threshold from the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with each face carrying its own local theory-closure and audit-clean verification rows.

For the reverse direction, suppose the four displayed faces expose their local objectwise exit-threshold records and the packet carries only the stated BHist, hsame, Cont, Pkg, and NameCert support rows. Apply Definition 38.494 to read those four records as the current exit threshold, and then apply Definition 371.62 to identify the same four records as the complete RegularCauchyName handoff surface.

Finally apply Definition 38.537. Aggregate route-level formalstatus rows, aggregate bridge rows, quotient streams, selected host limits, ambient completeness rows, metric-completeness rows, countable-choice rows, and standard-bridge substitutes cannot replace any one of the four local threshold faces, so the equivalence is exactly the displayed objectwise threshold. □

Theorem 371.64 (RegularCauchyName exit-threshold nonescape boundary)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. If G satisfies the Real-and-Completion exit threshold through the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, then the same handoff does not supply quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric completeness, an aggregate* formalstatus *row, an aggregate* leantarget *row, aggregate bridge data, or a* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Equivalently, the RegularCauchyName handoff witnesses only the objectwise L10 threshold on those four faces together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Every stronger read listed above lies outside the accepted threshold surface.*

Proof. First identify the positive threshold surface. By Definition 371.63, satisfying the Real-and-Completion exit threshold is exactly the exposure of the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next compare that surface with the handoff completeness criterion. Apply Definition 371.62. The accepted handoff is complete precisely through the four displayed faces and refuses replacements by quotient stream equality, selected host limits, host real equality, countable choice, ambient completeness, metric completeness, or bridge rows.

Now fix the stream and sequence sides. The tail-budget handoff theorem Definition 371.78 keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ coordinates finite and local, so no quotient stream equality, selected limit, or host real equality is created by the same tail-budget handoff.

Finally read the Real L10 threshold objectwise. The objectwise equivalence Definition 38.494 keeps the exit criterion on the four local face records, and the facewise noncollapse theorem Definition 38.537 excludes aggregate status, aggregate target, aggregate bridge, and $\mathsf{Real}^{\uparrow}$ completeness readings from replacing those local records. Thus the RegularCauchyName handoff reaches the positive exit threshold without exporting any stronger Real-and-Completion data. □

Theorem 371.65 (RegularCauchyName exit-threshold facewise locality)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. Satisfying the Real-and-Completion exit threshold is equivalent to carrying four independent local exit records on* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows of the RegularCauchyName handoff.*

The equivalence is facewise. Replacing any one of the four records by aggregate formalstatus*, aggregate* leantarget*, bridge data, quotient stream equality, a selected limit, countable choice, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness does not satisfy the threshold.*

Proof. First identify the threshold itself. By Definition 371.63, the Real-and-Completion exit threshold for G is exactly the exposure of the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the listed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next isolate the handoff surface. The facewise handoff completeness theorem Definition 371.62 says that the accepted RegularCauchyName handoff is complete precisely on those four faces and refuses quotient, selected-limit, equality, choice, completeness, and bridge substitutes.

Now read the StreamName and RegSeqRat coordinates locally. Apply Definition 371.78. The finite tail budget keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ records objectwise, so their exit contribution cannot be moved into a quotient stream, selected host limit, host equality, countable choice, or metric-completeness datum.

Finally compare with the Real L10 objectwise criterion. The theorem Definition 38.494 reads the Phase Real and Completion threshold from the four local records, and Definition 38.537 rules out aggregate status, aggregate target, bridge data, quotient substitutes, selected limits, countable choice, metric completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness as replacements for any one face. Therefore the threshold is exactly the four independent local records plus the displayed support rows. □

Theorem 371.66 (RegularCauchyName finite-section terminal pullback coherence)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*. Let F be an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *packet reading the same displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support as the exit-threshold handoff of G. Then the RegularCauchyName exit-threshold handoff and the finite-section consumer route determine one finite terminal pullback packet.*

This packet is the common finite BHist *section whose StreamName-facing and Real-facing terminal reads are identified by the Real finite-request terminal route. It contains no quotient stream, selected host limit, host equality, countable choice, ambient completeness, metric completeness, aggregate status row, bridge row, or second* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. First read the RegularCauchyName side. By Definition 371.63, the accepted exit threshold of G is the objectwise four-face surface on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next apply Definition 371.64. The same handoff cannot be enlarged by quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric completeness, aggregate status rows, aggregate target rows, bridge data, or a separate $\mathsf{Real}^{\uparrow}$ completeness claim.

Now read the finite-section side. The route consumer exactness theorem Definition 1215.5 says that an accepted $\mathsf{FiniteRealSection}^{\uparrow}$ packet using the displayed finite window, tolerance row, readback row, seal row, and support rows has exactly that finite consumer route. The nonescape theorem Definition 1215.6 rules out any additional completion or host coordinate outside the finite section.

It remains to identify the terminal packet. Apply Definition 38.705 to identify the StreamName-facing finite request with the Real-facing terminal section on the same finite BHist packet. Then apply Definition 38.708 to read the shared $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces as one accepted four-face terminal request.

The two routes therefore name one finite terminal pullback packet. Since the RegularCauchyName nonescape boundary and the FiniteRealSection nonescape theorem both refuse all coordinates beyond the displayed finite rows, the packet has no quotient stream, selected host limit,

host equality, countable choice, ambient completeness, metric completeness, aggregate status row, bridge row, or second $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 371.67 (RegularCauchyName terminal pullback exactness)**.** *Let G be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet whose visible source is admitted from* $\mathsf{RationalStream}^{\uparrow}$*, and let F be an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *packet. Suppose G and F read the same displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support. Then every Real-and-Completion consumer read compatible with both projections factors through the finite terminal pullback packet of Definition 371.66.*

The two projections are jointly determining: once the RegularCauchyName handoff projection and the FiniteRealSection route projection agree on the displayed rows, there is a unique compatible finite consumer route. No quotient stream, selected host limit, host equality, countable choice, ambient completeness, metric completeness, aggregate status row, bridge row, or second $\mathsf{Real}^{\uparrow}$ *seal can be added to that route.*

Proof. First identify the RegularCauchyName exit surface. Apply Definition 371.63 to read the accepted handoff of G as the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows.

Next apply Definition 371.64. The same handoff cannot acquire quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric completeness, aggregate status rows, aggregate target rows, bridge data, or a separate $\mathsf{Real}^{\uparrow}$ completeness claim.

Now read the finite-section projection. The route consumer exactness theorem Definition 1215.5 pins every compatible $\mathsf{FiniteRealSection}^{\uparrow}$ consumer to the displayed finite window, tolerance row, readback row, seal row, and support rows. The nonescape theorem Definition 1215.6 excludes any completion or host coordinate outside that finite route.

It remains to compare the two projections. By Definition 371.66, the RegularCauchyName handoff and the finite-section route determine one finite terminal pullback packet over the shared rows. Apply Definition 38.705 to identify the StreamName-facing request with the Real-facing terminal section on that packet, and use Definition 38.708 to read the four displayed faces as one accepted terminal request.

Therefore any Real-and-Completion consumer compatible with both projections must factor through that terminal pullback packet. Since both nonescape results refuse every coordinate beyond the displayed finite rows, the two projections are jointly determining and no quotient stream, selected host limit, host equality, countable choice, ambient completeness, metric completeness, aggregate status row, bridge row, or second $\mathsf{Real}^{\uparrow}$ seal can be attached. □

Theorem 371.68 (RegularCauchyName scheduled observation source totality)**.** *For every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, each unary schedule row admitted by the packet supplies a certified* $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ *observation row together with the displayed package provenance route. The supplied row is read through the packet's* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *coordinates, and every scheduled source read reaches one of these certified rows before it can be consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*.*

Proof. Induct over the finite schedule rows displayed by the accepted packet. The empty schedule has no requested observation, so the totality claim is the empty projection of the same BHist packet.

For a retained row, the RationalStream source admission theorem supplies the $\mathsf{RationalStream}^{\uparrow}$ source row, while the normalized window readback definition records the corresponding $\mathsf{StreamName}^{\uparrow}$

finite window and dyadic observation route. The displayed hsame and Cont rows transport this observation without introducing a second source.

The package provenance route is then the stored Pkg coordinate, with the local $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ row certifying that the observation belongs to the accepted packet. Repeating this projection along the finite schedule exhausts every scheduled source read. □

Theorem 371.69 (RegularCauchyName radius ledger monotone readback)**.** *In an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, the dyadic-radius ledger is monotone under certified unary index extension: extending a schedule row along the displayed* hsame *and* Cont *route can only read a radius row already compatible with the earlier request and the later certified request. Every such readback remains inside the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *tolerance surface carried by the normalized finite-window packet.*

Proof. Begin with the two certified unary index rows. The normalized window readback definition reads both through the same $\mathsf{StreamName}^{\uparrow}$ window surface, dyadic observation row, radius row, finite ledger, and package provenance.

Use the radius obligations already carried by the packet. The earlier radius request is preserved by classifier transport, and the later request extends it through the displayed continuation route, so the ledger read is monotone inside the stored finite radius table.

Finally apply window normalization coverage. It forces every consumer precision read to pass through the normalized window before the $\mathsf{RegSeqRat}^{\uparrow}$ classifier or $\mathsf{Real}^{\uparrow}$ seal is reached, so the readback remains inside the $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ tolerance surface. □

Theorem 371.70 (RegularCauchyName RegSeqRat Real seal root unblock)**.** *The scheduled observation source rows of Definition 371.68 and the monotone radius ledger of Definition 371.69 assemble a finite root-unblock package for every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet. This package is consumable by the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal route and the* $\mathsf{Real}^{\uparrow}$ *handoff, and it uses the displayed* $\mathsf{StreamName}^{\uparrow}$ *handoff, semantic* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *certificate,* BHist*,* hsame*,* Cont*, and* Pkg *rows rather than quotient streams or host equality.*

Proof. First assemble the source half of the package. Scheduled source totality supplies each unary row with its certified $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ observation and provenance route.

Next assemble the radius half. Radius-ledger monotone readback keeps each certified index extension inside the dyadic tolerance surface and the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ classifier ledger.

Finally attach the seal handoffs. The StreamName handoff, normalized readback seal stability, and root-unblock normalization package carry the assembled rows to the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The semantic naming certificate is already part of the packet, so no quotient stream, host equality, selected limit, or ambient completeness coordinate is needed. □

Theorem 371.71 (RegularCauchyName tail-window consumer factorization)**.** *Every tail-window consumer of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet factors through the scheduled observation row, the monotone dyadic-radius ledger, the normalized finite-window readback, and the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal route. The factorization reads only the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *coordinates. It cannot inspect quotient streams, selected limits, host sequence equality, independent real equality, or ambient completeness.*

Proof. First bind the tail-window request to its scheduled source. The scheduled observation source totality theorem Definition 371.68 supplies the certified $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ observation row and package provenance before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can read the packet.

Next route the tolerance information. The radius-ledger monotone readback theorem Definition 371.69 keeps certified index extensions inside the displayed $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ tolerance surface, while window normalization coverage Definition 371.43 forces each precision read through the normalized finite-window packet.

Finally attach the terminal seal route. The RegSeqRat Real seal root unblock theorem Definition 371.70 carries the scheduled source and radius ledger to the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The closure boundary records no quotient stream, selected limit, host sequence equality, independent real equality, or ambient completeness row, so the tail-window consumer has no escape from the displayed factorization. □

Theorem 371.72 (RegularCauchyName tail-window budget coherence)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let U and V be two accepted tail-window consumer reads of R whose requested precisions share a certified budget row in the dyadic-radius ledger. Then U and V factor through the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the same dyadic-radius ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the same* $\mathsf{Real}^{\uparrow}$ *seal row. The common factorization is displayed through the packet's* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance, and it contains no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First read the source rows. The scheduled observation source totality theorem Definition 371.68 supplies the certified $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ observation rows for both U and V through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ provenance.

Next compare the certified budget row. The radius ledger monotone readback theorem Definition 371.69 keeps each accepted precision read inside the same dyadic tolerance surface once the two requests share a certified ledger budget. The synchronized precision pullback theorem Definition 371.35 refines the two reads to one finite $\mathsf{StreamName}^{\uparrow}$ window and one $\mathsf{DyadicRatCore}^{\uparrow}$ observation row.

Finally apply tail-window consumer factorization. By Definition 371.71, both terminal reads reach the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and the $\mathsf{Real}^{\uparrow}$ seal only through that shared finite window, dyadic observation row, ledger row, and provenance surface. Since the factorization exposes no coordinate for a quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness premise, the common budget row is coherent for both accepted consumers. □

Theorem 371.73 (RegularCauchyName shared budget triple pullback)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let U, V, and W be three accepted tail-window consumer reads of R whose requested precisions share one certified budget row in the dyadic-radius ledger. Then U, V, and W factor through one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, one dyadic-radius ledger row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and one* $\mathsf{Real}^{\uparrow}$ *seal row. The common pullback is displayed only through the packet's* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance, and it exports no quotient stream, host sequence, selected limit, independent real carrier, or ambient completeness coordinate.*

Proof. First read the three scheduled sources. The scheduled observation source totality theorem Definition 371.68 supplies the certified $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ observation source for each of U, V, and W through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ rows.

Next pull all three precision requests back to the shared budget row. The radius ledger monotone readback theorem Definition 371.69 keeps each accepted request inside the same dyadic tolerance surface, and the synchronized precision pullback theorem Definition 371.35 identifies the finite window route attached to that certified row.

Apply the pairwise budget coherence theorem Definition 371.72 first to U and V, and then to U and W. The first application gives a common finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, dyadic-radius ledger row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and $\mathsf{Real}^{\uparrow}$ seal row for U and V. The second application gives the same kind of route for U and W.

Finally compare the two pairwise routes at the shared budget row. Since both routes contain the U consumer read and the same certified ledger row, they identify the same normalized finite-window readback and therefore the same downstream $\mathsf{RegSeqRat}^{\uparrow}$ classifier and $\mathsf{Real}^{\uparrow}$ seal. Tail-window consumer factorization Definition 371.71 keeps all three terminal reads inside this displayed provenance surface. □

Theorem 371.74 (RegularCauchyName shared budget finite intersection)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let $\mathcal{F}$ be a finite nonempty family of accepted tail-window consumer reads of R whose requested precisions share one certified budget row in the dyadic-radius ledger. Then all reads in $\mathcal{F}$ factor through one finite* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, one dyadic-radius ledger row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and one* $\mathsf{Real}^{\uparrow}$ *seal row. The common intersection is displayed only through the packet's* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance, with no quotient stream, selected limit, host sequence equality, independent real carrier, or ambient completeness coordinate.*

Proof. Induct over the finite family $\mathcal{F}$ after choosing one accepted read as the anchor. The one-read case is the tail-window consumer factorization of Definition 371.71; the first nontrivial two-read case is exactly the shared budget coherence theorem Definition 371.72.

For the three-read stage, apply the triple pullback theorem Definition 371.73. It identifies one finite $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, one ledger row, one $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and one $\mathsf{Real}^{\uparrow}$ seal row for the anchor and the two additional reads at the displayed shared budget.

For the induction step, assume the retained finite subfamily already factors through one displayed route. Compare that common route with the next accepted read at the same certified budget row. The pair theorem Definition 371.72 applies to the anchor read inside the common route and to the next read, so the next read is pulled back to the same finite window, dyadic observation row, ledger row, classifier row, and seal row.

Finally use tail-window consumer factorization again to close the boundary condition. Since every retained route is read through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$, the finite intersection has no coordinate for quotient streams, selected limits, host sequence equality, independent real carriers, or ambient completeness. □

Theorem 371.75 (RegularCauchyName tail-window budget route exhaustion)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let $\mathcal{F}$ be a finite nonempty family of accepted tail-window consumer reads of R whose requested precisions share one certified dyadic-radius budget row. Then every terminal read in $\mathcal{F}$ is exhausted by the single finite route through the scheduled observation row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, radius ledger row,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row,* $\mathsf{Real}^{\uparrow}$ *seal row, and the packet's* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance. This route exports no quotient stream, selected limit, host sequence equality, independent real carrier, countable-choice selector, or ambient completeness coordinate.*

Proof. First bind every consumer read to the scheduled source surface. Scheduled observation source totality Definition 371.68 supplies the certified $\mathsf{RationalStream}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ observation

row for each accepted read, and tail-window consumer factorization Definition 371.71 forces the read to pass through that row before it can reach the $\mathsf{RegSeqRat}^{\uparrow}$ classifier or $\mathsf{Real}^{\uparrow}$ seal.

Next bind the shared budget. Radius ledger monotone readback Definition 371.69 keeps every certified precision request inside the same dyadic tolerance surface, while tail-window budget coherence Definition 371.72 and shared-budget triple pullback Definition 371.73 identify the pair and triple routes attached to the common ledger row.

Finally apply shared-budget finite intersection Definition 371.74. It collapses the whole finite nonempty family to one finite $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, one radius ledger row, one $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and one $\mathsf{Real}^{\uparrow}$ seal row. The carrier inventory of the accepted packet contains only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ provenance beyond that route, so there is no off-route terminal coordinate from which a quotient stream, selected limit, host sequence equality, independent real carrier, countable-choice selector, or ambient completeness witness could be exported. □

Theorem 371.76 (RegularCauchyName cofinal budget refinement)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let $\mathcal{F}$ be a finite nonempty family of accepted tail-window consumer reads of R whose requested precisions share one certified dyadic-radius budget row b. If a second certified budget row b' refines b cofinally in the packet's dyadic-radius ledger, then the terminal route supplied by Definition 371.75 refines through the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the same radius ledger row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the same* $\mathsf{Real}^{\uparrow}$ *seal row. The refinement is read only through the packet's* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *provenance, and it exposes no quotient stream, host sequence equality, selected limit, countable-choice selector, or ambient completeness coordinate.*

Proof. First apply tail-window budget route exhaustion Definition 371.75 to the accepted family at the certified budget row b. The result supplies a single terminal route through the scheduled observation row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, radius ledger row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and $\mathsf{Real}^{\uparrow}$ seal row.

Next use shared-budget finite intersection Definition 371.74 to keep the whole finite family on that displayed route. The cofinal refinement row b' is compared with b inside the same finite ledger surface, so no member of $\mathcal{F}$ can choose an independent downstream route.

Now apply radius-ledger monotone readback Definition 371.69. It transports every certified precision read from b to the refining row b' without leaving the $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ tolerance surface carried by the normalized finite-window packet.

Finally apply synchronized precision pullback Definition 371.35. The pullback identifies the refining precision reads with the same finite $\mathsf{StreamName}^{\uparrow}$ window and the same $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, hence with the same radius ledger row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and $\mathsf{Real}^{\uparrow}$ seal row already supplied by exhaustion. The accepted packet has no exported coordinate for a quotient stream, host sequence equality, selected limit, countable-choice selector, or ambient completeness witness, so the cofinal refinement remains entirely within the finite route. □

Theorem 371.77 (RegularCauchyName tail-budget refinement invariance)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let $\mathcal{F}$ be a finite nonempty family of accepted tail-window consumer reads of R whose requested precisions share one certified dyadic-radius budget row b. If b' is a certified cofinal refinement of b in the packet's dyadic-radius ledger, then the public seal route through scheduled observation, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the radius ledger row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal is invariant under replacing b by b'. The invariant route is read only through the displayed* BHist, hsame, Cont,

Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *rows, and it exports no quotient stream, selected limit, host sequence equality, countable-choice selector, choice row, or ambient completeness coordinate.*

Proof. Start at the original budget row b. Tail-window budget route exhaustion Definition 371.75 supplies one public terminal route for the whole finite family, passing through the scheduled observation row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, radius ledger row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and $\mathsf{Real}^{\uparrow}$ seal.

Transport this route to the refining row b'. Cofinal budget refinement Definition 371.76 keeps every accepted read in $\mathcal{F}$ on the same normalized finite-window route while changing only the certified ledger budget row. Thus the refining row cannot introduce a second downstream classifier or seal surface.

Identify the StreamName and Real faces of the transported route. StreamName finite-window reindex determinacy Definition 126.639 says that the finite observation section is unchanged under the certified reindexing data carried by the displayed packet. Real–RegSeqRat–StreamName tightness Definition 38.773 then identifies the terminal $\mathsf{Real}^{\uparrow}$ seal as the same public seal read from the common $\mathsf{StreamName}^{\uparrow}$ section, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ classifier boundary.

Finally read the carrier inventory of the accepted $\mathsf{RegularCauchyName}^{\uparrow}$ packet. The route contains only the scheduled observation support, finite StreamName window, dyadic observation row, radius ledger row, RegSeqRat classifier row, Real seal row, and the displayed BHist, hsame, Cont, Pkg, and naming provenance. There is therefore no exported coordinate for a quotient stream, selected limit, host sequence equality, countable-choice selector, choice row, or ambient completeness witness, so the terminal public seal route is invariant under the cofinal budget refinement. □

Theorem 371.78 (RegularCauchyName tail-budget StreamName–RegSeqRat handoff)**.** *Let R be an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet, and let $\mathcal{F}$ be a finite nonempty family of accepted tail-window consumer reads of R whose requested precisions share one certified dyadic-radius budget row b. Suppose the StreamName and RegSeqRat faces of the accepted route satisfy the tail-budget uniformity condition of Definition 126.105. Then the public* $\mathsf{Real}^{\uparrow}$ *seal route is exactly the finite route supplied by Definition 371.75, and replacing b by any certified cofinal refinement b' leaves that public seal route invariant as in Definition 371.77. The handoff exports no quotient stream, selected limit, host sequence equality, countable-choice selector, choice row, ambient completeness witness, or metric-completeness datum.*

Proof. First apply tail-window budget route exhaustion Definition 371.75 to the accepted family at b. This pins the scheduled observation row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, radius ledger row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and terminal $\mathsf{Real}^{\uparrow}$ seal row as one finite public route for every consumer read in $\mathcal{F}$.

Next apply StreamName–RegSeqRat tail-budget uniformity Definition 126.105 to the displayed StreamName and RegSeqRat faces. Every later finite window accepted by the same pointwise StreamName classifier is identified with the same RegSeqRat tail selector and the same Real budget consumer, so enlarging the visible StreamName window changes only the finite window ledger and not the public seal route.

Finally apply tail-budget refinement invariance Definition 371.77. A certified cofinal refinement b' of the dyadic-radius budget row transports through the already pinned finite route, hence it keeps the scheduled observation, finite StreamName window, dyadic observation row, radius ledger row, RegSeqRat classifier, and terminal Real seal unchanged.

The carrier inventory of the accepted $\mathsf{RegularCauchyName}^{\uparrow}$ packet contains only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ provenance beyond that route. Thus the

handoff has no exported coordinate for quotient streams, selected limits, host sequence equality, countable choice, ambient completeness, or metric-completeness data. □

Theorem 371.79 (RegularCauchyName terminal finite-cover witness)**.** *Every accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet with displayed root dependency manifest and four-face support minimality admits a terminal finite-cover witness whose support consists exactly of the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows named in Definition 371.50. Every terminal* $\mathsf{Real}^{\uparrow}$ *read or completion-facing read factors through that witness, and the witness exports no quotient stream, host sequence equality, selected limit, independent real carrier, ambient completeness, metric completeness, countable-choice, or bridge row.*

Proof. First locate the source and consumer reads. The root dependency manifest exhaustion theorem Definition 371.51 places every downstream source, finite-window, readback, $\mathsf{Real}^{\uparrow}$ seal, and completion consumer read inside the displayed manifest of Definition 371.50.

Next pass from the manifest to the terminal exit. The manifest exit descent exactness theorem Definition 371.53 identifies the terminal route, and the Phase Real and Completion exit interface theorem Definition 371.54 reads that route through the four faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply the four-face support minimality theorem Definition 371.56 and the terminal no-extra-support theorem Definition 371.57. Minimality pins the witness to exactly those four faces and support rows, while no-extra-support refuses quotient stream data, host sequence equality, selected limits, independent real carriers, ambient or metric completeness, countable-choice coordinates, and bridge rows. Thus every terminal $\mathsf{Real}^{\uparrow}$ or completion-facing read factors through the displayed finite-cover witness and no other support row is exported. □

Theorem 371.80 (RegularCauchyName terminal finite-cover refinement reflection)**.** *Let W and W' be accepted terminal finite-cover witnesses for the same* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet with displayed root dependency manifest, and suppose that W and W' admit an accepted common refinement R. If the pullbacks of W and W' to R agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, *componentwise* hsame, Cont, Pkg, *and* NameCert *rows named by Definition 371.50, then W and W' present the same terminal* $\mathsf{Real}^{\uparrow}$ *read and the same completion-facing read. Conversely, sameness of those terminal reads reflects, after pullback to R, agreement on exactly those finite rows.*

The reflection is only a statement about the finite terminal support of $\mathsf{RegularCauchyName}^{\uparrow}$. *It does not identify quotient streams, host sequences, selected limits, countable-choice selectors, ambient completeness, metric completeness, aggregate formalstatus rows, bridge rows, or any row outside the root dependency manifest.*

Proof. First reduce each presentation to the terminal finite-cover witness surface. By Definition 371.79, both W and W' factor every terminal $\mathsf{Real}^{\uparrow}$ read and completion-facing read through the finite support listed in Definition 371.50. The four-face support minimality theorem Definition 371.56 pins that support to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the displayed BHist, componentwise hsame, Cont, Pkg, and NameCert rows.

For the forward direction, pull W and W' back to the accepted common refinement R. The StreamName finite-cover support bicriterion Definition 126.669 reads agreement on the displayed finite rows as sameness of the descended $\mathsf{Real}^{\uparrow}$ seal-visible source. The terminal no-extra-support theorem Definition 371.57 then transfers that sameness to the terminal $\mathsf{Real}^{\uparrow}$ read and to the completion-facing read, because no fifth support coordinate can change the exit.

For the reverse direction, assume the two witnesses present the same terminal $\mathsf{Real}^{\uparrow}$ read and the same completion-facing read. Apply Definition 371.79 to express both reads through the two finite-cover witnesses, and use Definition 371.57 to remove any possible contribution from rows outside the four-face terminal surface.

It remains only to reflect agreement back to the refinement. The StreamName finite-cover support bicriterion Definition 126.669 reflects equality of the descended terminal support packet to agreement, after pullback to R, on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, componentwise hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since Definition 371.56 identifies exactly those rows as the RegularCauchyName support, this is precisely the reflected row agreement claimed.

Finally record the refusal boundary. The witness theorem and the terminal no-extra-support theorem refuse quotient stream equality, host sequence equality, selected limits, independent real carriers, ambient completeness, metric completeness, countable choice, and bridge rows; the StreamName bicriterion also refuses aggregate formalstatus rows. Therefore neither direction imports quotient stream equality, host sequence equality, selected limits, countable choice, ambient completeness, metric completeness, aggregate formalstatus, bridge data, or any row not named by the root dependency manifest. □

Definition 371.81 (RegularCauchyName carrier)**.** *A* $\mathsf{RegularCauchyName}^{\uparrow}$ *carrier is a finite source-row packet*

$$R = (W, A, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that exposes only certified observation windows.*
2. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the scheduled observations.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the radius requested at each displayed window.*
4. *L is the regularity ledger tying W, A, T to one finite Cauchy-name surface.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after the window, readback, and tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, readback, tolerance, regularity ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the scheduled window through the dyadic tolerance and regular rational readback into the seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *row.*

The carrier is the source-row face of the regular Cauchy name route used by chapter 126, chapter 301, chapter 291, chapter 38, and chapter 1680. It exports no quotient stream, host sequence equality, selected limit, independent completed real, or ambient completeness witness.

Theorem 371.82 (RegularCauchyName NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *carrier $R = (W, A, T, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, regularity-ledger exactness,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier rows of Definition 371.81. The schedule, readback, tolerance ledger, regularity ledger, seal, transport, replay, provenance, and naming rows are the only coordinates available to a consumer.

The window route is already the route of Definition 371.3; the dyadic-radius route is the route of Definition 371.2; and the seal route is the route of Definition 371.4. These rows force every consumer read through the displayed finite schedule before any regular rational readback or Real seal is consumed.

Transport and replay are exhausted by H and C, while P and N source and name the same packet. Since no coordinate stores a quotient stream, host sequence, selected limit, independent completed real, or ambient completeness witness, the listed rows are exactly the local naming obligations. □

Theorem 371.83 (RegularCauchyName StreamName-RegSeqRat handoff)**.** *Every consumer read of an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *carrier* $R = (W, A, T, L, E, H, C, P, N)$ *reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row only through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*. The handoff is the forward source-row dependency from finite windows to regular rational readback before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Begin with a consumer precision request. By Definition 371.3, the request first enters a displayed $\mathsf{StreamName}^{\uparrow}$ window rather than a host sequence.

The tolerance row is then read through $\mathsf{DyadicRatCore}^{\uparrow}$. The dyadic-radius obligation theorem keeps that tolerance on the same scheduled window, so the regular rational readback in A is attached to W and T before any seal-facing row is available.

Finally apply the Real seal boundary and the finite-window readback exactness results already recorded in this chapter. They place E downstream of W, A, T, L and keep the handoff inside the displayed BHist, hsame, Cont, Pkg, and NameCert rows, with no quotient stream or selected limit added. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyName}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ rational observations, dyadic-radius rows at certified unary indices, a finite $\mathsf{RegSeqRat}^{\uparrow}$ regularity ledger, a $\mathsf{Real}^{\uparrow}$ seal row, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature boundary package, normalized window readback, window normalization coverage, normalized readback seal stability, and root-unblock normalization package close the paper surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite-window rows, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyName}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RegularCauchyNameUp.RegularCauchyNameUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close real completeness, quotient equality of regular streams, uniqueness of limits, host real construction, or any stronger bridge schema.

Upgrade path: No further upgrade – chapter at top of the formal axis.

Chapter 372

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonal}^{\uparrow}$

The $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ packet names the stationary and finite-window diagonal data used by the real-completeness construction before it is repacked into the broader $\mathsf{RealDiagonalCompletion}^{\uparrow}$ surface. Its carrier is deliberately smaller than a completion theorem: it records the inspected regular Cauchy rows, the diagonal selector, the stationary rational rows, and the local naming-certificate provenance through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The diagonal packet consumes the $\mathsf{StreamName}^{\uparrow}$ window carrier of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, and exports to the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ consumer of chapter 359.

Definition 372.1 (Regular Cauchy diagonal source)**.** *A* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row is a finite* BHist *packet*

$$D = (Q, S, R, L, W, P, N)$$

whose displayed coordinates are as follows:

1. *Q is the visible* $\mathsf{Rat}^{\uparrow}$ *seed row used for stationary rational inputs;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the stream windows read by the diagonal selector;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular rational sequence selected at the requested diagonal precision;*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row receiving the selected diagonal representative;*
5. *W is the finite list of requested diagonal windows and their* Cont *routes through the inspected regular Cauchy family;*
6. *P is the* Pkg *row naming the* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and local certificate dependencies;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}$ *provenance row.*

The source row contains no quotient of regular streams, no selected host limit, and no ambient completeness field.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalTasteGate single carrier alignment`

Definition 372.2 (Regular Cauchy diagonal carrier)**.** *The carrier of* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *consists of the source rows of Definition 372.1 whose diagonal windows are all read from displayed*

$\mathsf{RegSeqRat}^{\uparrow}$ *rows and whose stationary rational rows factor through the same* $\mathsf{Rat}^{\uparrow}$ *seed, the same* $\mathsf{StreamName}^{\uparrow}$ *constant-window witness, and the same* $\mathsf{Real}^{\uparrow}$ *seal row. The classifier relates two carrier rows exactly when their visible rational seeds, stream windows, regular-sequence reads, real-seal rows, and package provenance are transported componentwise by the displayed* hsame *comparisons and* Cont *routes.*

Lean4 (definition) `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier`

Theorem 372.3 (Regular Cauchy diagonal window coverage)**.** *Every consumer read of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *carrier factors through one displayed diagonal window, one* $\mathsf{StreamName}^{\uparrow}$ *window row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the associated* $\mathsf{Rat}^{\uparrow}$ *seed evidence for stationary rational rows, and the visible* Pkg *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier window coverage`

Proof. Project the accepted carrier through Definition 372.2. The only consumer-facing coordinates are the diagonal-window list, the stream-window row, the regular rational readback, the real-seal row, and the local provenance rows.

For a requested precision, the diagonal selector chooses a displayed member of the regular Cauchy family and a displayed window inside that member. The Cont route in the source packet records that this window is one of the finite rows already named by the $\mathsf{StreamName}^{\uparrow}$ schedule.

The selected value is then read through the $\mathsf{RegSeqRat}^{\uparrow}$ row and, in the stationary case, through the same $\mathsf{Rat}^{\uparrow}$ seed row. Repacking these visible rows with P and N gives the claimed factorization, and there is no additional carrier coordinate from which a host limit or quotient equality could be read. □

Theorem 372.4 (Regular Cauchy diagonal stationary compatibility)**.** *If a* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *carrier row is generated by a stationary rational seed* $Q : \mathsf{Rat}^{\uparrow}$*, then every diagonal window reads the same rational seed through the constant* $\mathsf{StreamName}^{\uparrow}$ *row, and the resulting* $\mathsf{Real}^{\uparrow}$ *seal is classified with the constant rational embedding of* Q*.*

Proof. In the stationary packet, every inspected member of the regular Cauchy family is the same regular rational row. The diagonal selector may vary the requested window, but the stream row is constant and each selected observation is transported by the same displayed hsame and Cont witnesses.

Thus the $\mathsf{RegSeqRat}^{\uparrow}$ readback row carries the seed Q at every diagonal window. The $\mathsf{Real}^{\uparrow}$ seal coordinate is attached to exactly those readbacks, so the constant rational embedding receives the same rational seed and the same local provenance.

The classifier of Definition 372.2 compares only the displayed seed, window, readback, seal, and package rows. It therefore identifies the stationary diagonal carrier with the constant rational $\mathsf{Real}^{\uparrow}$ embedding of Q. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier stationary compatibility`

Theorem 372.5 (Regular Cauchy diagonal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}$ *obligation surface consists of carrier habitation, componentwise classifier transport through* hsame *and* Cont*, diagonal-window coverage, stationary rational compatibility, and ledger non-escape over* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier namecert obligations`

Proof. Carrier habitation is the packaging operation of Definition 372.2: the source row supplies the rational seed, stream windows, regular-sequence readbacks, real seal, finite diagonal-window routes, and provenance rows.

Classifier transport is componentwise. Each accepted replacement is an hsame comparison of one displayed row or a Cont route identifying where that row is read inside the finite diagonal packet, so the transported packet remains within the same carrier definition.

Window coverage is exactly Definition 372.3, and stationary rational compatibility is exactly Definition 372.4. The ledger non-escape claim follows by inspecting the listed source coordinates: none is a quotient carrier, host-limit selector, ambient completeness theorem, or hidden real endpoint. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier semantic name certificate`

Theorem 372.6 (Regular Cauchy diagonal zero seed)**.** *For the source row $D = (Q, S, R, L, W, P, N)$ of Definition 372.1, the stationary rational seed with Q the zero* $\mathsf{Rat}^{\uparrow}$ *row, S the constant* $\mathsf{StreamName}^{\uparrow}$ *window row, and R the constant* $\mathsf{RegSeqRat}^{\uparrow}$ *readback inhabits the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *carrier. Its* $\mathsf{Real}^{\uparrow}$ *seal row L, finite window ledger W, package row P, and local* NameCert *row N are all displayed finite rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier zero seed`

Proof. Start with the stationary rational seed. The zero $\mathsf{Rat}^{\uparrow}$ row supplies the same displayed rational observation at every requested diagonal precision, and the constant $\mathsf{StreamName}^{\uparrow}$ row supplies the same finite window witness for each read.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback is therefore the constant regular rational row. By Definition 372.4, the resulting $\mathsf{Real}^{\uparrow}$ seal is classified with the constant rational embedding of the same seed.

Finally package the constant window routes with P and N. The carrier definition in Definition 372.2 requires no quotient stream, selected host limit, or ambient completeness field, so these displayed rows inhabit the diagonal carrier. □

Theorem 372.7 (Regular Cauchy diagonal selector stability)**.** *Let $D = (Q, S, R, L, W, P, N)$ be an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row. Transporting Q, S, R, L, W, P, and N componentwise by the displayed* hsame *comparisons and* Cont *routes preserves the diagonal selector classifier through the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier selector stability`

Proof. Project the classifier from Definition 372.2. It compares only visible rational seeds, stream windows, regular-sequence readbacks, real-seal rows, finite diagonal-window routes, and provenance.

The selector uses the window list W to choose a displayed window inside the inspected regular Cauchy family. Componentwise hsame comparisons and Cont routes move exactly those displayed windows and readback rows, so the selected diagonal row remains inside the same finite packet.

Apply Definition 372.3 for the consumer-facing route and Definition 372.5 for the local certificate surface. Together they show that the transported packet preserves the selector classifier without adding quotient streams, host limits, or ambient completeness data. □

Theorem 372.8 (Regular Cauchy diagonal RealUp seal handoff)**.** *For every accepted source row $D = (Q, S, R, L, W, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal row L is reached only through the displayed* $\mathsf{Rat}^{\uparrow}$ *seed row,* $\mathsf{StreamName}^{\uparrow}$ *window row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row. This handoff is the*

$\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}$ *seal-facing route consumed by downstream* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *packets.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier real seal handoff`

Proof. Read a consumer request through Definition 372.3. The request first selects a displayed diagonal window, then reads the corresponding $\mathsf{StreamName}^{\uparrow}$ window row and $\mathsf{RegSeqRat}^{\uparrow}$ regular rational row.

In the stationary case, Definition 372.4 identifies the same route with the $\mathsf{Rat}^{\uparrow}$ seed and its constant rational $\mathsf{Real}^{\uparrow}$ embedding. In the general accepted packet, the same coordinates are carried by Q, S, R, and W.

The obligation surface of Definition 372.5 records the seal row L only after those displayed rows and the provenance rows P and N. Hence the seal handoff is finite and visible, and no host limit, quotient stream, or ambient completeness theorem is exported to downstream completion packets. □

Theorem 372.9 (Regular Cauchy diagonal completion consumer bridge)**.** *Every downstream* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet factors through the displayed tuple* $D = (Q, S, R, L, W, P, N)$*, the diagonal-window coverage route of Definition 372.3, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff of Definition 372.8. The consumer receives no quotient stream, selected host limit, or ambient completeness row from the diagonal packet.*

Proof. Project the accepted packet through Definition 372.1 and Definition 372.2. The only exported coordinates are the rational seed Q, stream row S, regular readback R, seal row L, finite window ledger W, package row P, and local certificate row N.

Apply Definition 372.3 to route each completion read through a displayed diagonal window and its $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. Then apply Definition 372.8 to reach L through exactly those displayed rows.

The carrier definition has no coordinate for a quotient of streams, a chosen external limit, or an ambient completeness witness. Thus any completion consumer using the accepted packet can only consume the displayed tuple and the two named routes. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier completion consumer bridge`

Theorem 372.10 (Regular Cauchy diagonal stationary seal exactness)**.** *In the stationary rational case, the completion-facing seal route of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet is exactly the constant route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *supplied by Definition 372.4 and Definition 372.8.*

Proof. Start with a stationary source row $D = (Q, S, R, L, W, P, N)$. By Definition 372.2, the stream row S is the constant-window row over the same rational seed Q, and R is the displayed regular rational readback selected from that constant stream.

Apply Definition 372.4. Every diagonal window reads the same $\mathsf{Rat}^{\uparrow}$ seed through the constant $\mathsf{StreamName}^{\uparrow}$ row and the associated $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Apply Definition 372.8. The seal row L is reached only after those displayed rows, so the route to $\mathsf{Real}^{\uparrow}$ is precisely the constant stationary route and not a route through any host completion witness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier stationary seal exactness`

Theorem 372.11 (Regular Cauchy diagonal bridge obligation package)**.** *The bridge-facing obligation package for* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *consists of five finite obligations: carrier habitation, selector stability, diagonal-window coverage, stationary seal exactness, and completion-consumer routing. Together these are the obligations needed to pass from the public diagonal packet to the bridge-facing completion surface.*

Proof. Carrier habitation is obtained by projecting Definition 372.1 into Definition 372.2. The source row supplies Q, S, R, L, W, P, and N, and the carrier condition records that the selected diagonal reads stay inside those displayed rows.

Selector stability is Definition 372.7, and diagonal-window coverage is Definition 372.3. These two obligations show that transported packets preserve the visible selector and that every consumer read has a named finite window.

Stationary seal exactness is Definition 372.10. Completion-consumer routing is Definition 372.9. Repacking these five projections gives the bridge-facing package, while the source and carrier definitions exclude quotient streams, selected host limits, and ambient completeness rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier bridge obligation package`

Theorem 372.12 (Regular Cauchy diagonal standard bridge source)**.** *For every accepted source row* $D = (Q, S, R, L, W, P, N)$*, the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *bridge source is the finite tuple consisting of the* $\mathsf{Rat}^{\uparrow}$ *seed* Q*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* L*, the diagonal-window ledger* W*, the package row* P*, and the local* $\mathsf{NameCert}$ *row* N*, together with the displayed* BHist *carrier,* hsame *classifier,* Cont *routes, selector-stability proof, window-coverage proof, stationary-compatibility proof, and seal handoff to* $\mathsf{Real}^{\uparrow}$*. This tuple is a standard BEDC bridge source from the diagonal selector package to the real-completion consumer surface.*

Proof. Project the accepted packet through Definition 372.1 and Definition 372.2. These definitions supply exactly the visible carrier coordinates Q, S, R, L, W, P, and N, and they identify the classifier as componentwise hsame transport with displayed Cont routes.

The consumer-facing coverage row is Definition 372.3. Selector stability is Definition 372.7, and stationary rational compatibility is Definition 372.4. These three rows keep every selected diagonal read inside the displayed $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ surface.

Finally apply the local NameCert obligation surface of Definition 372.5 and the $\mathsf{Real}^{\uparrow}$ seal handoff of Definition 372.8. Repacking these rows gives the standard bridge source; no additional host completion object is needed to form the source. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier standard bridge source`

Theorem 372.13 (Regular Cauchy diagonal bridge readback exactness)**.** *Reading the* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *bridge source of Definition 372.12 back through its public surface recovers exactly the displayed tuple* $D = (Q, S, R, L, W, P, N)$*. The readback preserves the selector classifier because the only inspected rows are the* $\mathsf{Rat}^{\uparrow}$ *seed,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback,* $\mathsf{Real}^{\uparrow}$ *seal, finite window ledger, package provenance, and local* $\mathsf{NameCert}$ *provenance transported componentwise by* hsame *and* Cont*.*

Proof. Read the bridge source by projecting its carrier coordinate. By Definition 372.1, the carrier contains the seven displayed coordinates Q, S, R, L, W, P, and N and no further completion coordinate.

The bridge readback route is the same route used for diagonal-window coverage. Apply Definition 372.3 to identify the selected window and the matching $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, then apply Definition 372.8 to identify the exported seal row as L.

It remains to check the selector classifier. This is exactly Definition 372.7: componentwise hsame and Cont transport preserves the selected diagonal row and hence preserves the public readback tuple. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier bridge readback exactness`

Theorem 372.14 (Regular Cauchy diagonal bridge non-escape)**.** *The* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *standard bridge source exports no quotient stream, no selected host limit, no ambient completeness theorem, and no host equality. Every bridge consumer read factors through the displayed tuple* $D = (Q, S, R, L, W, P, N)$*, the finite window ledger, the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, and the local* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier bridge non escape`

Proof. Inspect the source coordinates in Definition 372.1. The tuple lists rational seed data, stream-window data, regular rational readback data, one real-seal row, finite windows, package provenance, and local certificate provenance.

The carrier definition in Definition 372.2 adds only the displayed acceptance and classifier conditions. It does not add a quotient stream, a chosen host limit, an ambient completeness theorem, or host equality.

The bridge source of Definition 372.12 is assembled from this same carrier together with window coverage, selector stability, stationary compatibility, NameCert obligations, and the RealUp seal handoff. Therefore every bridge read remains inside those named rows, and the listed host-level objects cannot be exported by the bridge. □

Theorem 372.15 (Regular Cauchy diagonal selector-window functoriality)**.** *Let* $D = (Q, S, R, L, W, P, N)$ *be an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row. If the displayed* hsame *comparisons and* Cont *routes transport the visible carrier rows of* D *to another accepted carrier row, then the diagonal selector transports the same finite* W *window route through the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinates. In particular, selector transport preserves the chosen window route and does not introduce a second selector outside the displayed finite ledger.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier selector window functoriality`

Proof. Project the transport data through Definition 372.2. The classifier compares only the displayed rational seed, stream-window row, regular readback, real-seal row, finite window ledger, and provenance rows, all carried by hsame or Cont.

Apply Definition 372.7 to the transported carrier row. The selector still reads a member of the same finite ledger W, and Definition 372.3 identifies the matching $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The bridge source of Definition 372.12 uses exactly these rows as its selector-facing input. Therefore the transported selector route is the displayed finite route, not a host-level re-selection or a route through an unlisted completion object. □

Theorem 372.16 (Regular Cauchy diagonal stationary-family exhaustion)**.** *Every stationary rational diagonal packet accepted by* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *is exhausted by the displayed route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$*. Equivalently, the packet has no stationary-family read outside the rational seed* Q*, the constant stream row* S*, the regular rational readback* R*, and the seal row* L *of its source tuple* $D = (Q, S, R, L, W, P, N)$*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier stationary family exhaustion`

Proof. Start with a stationary accepted carrier. By Definition 372.2, the stationary rows factor through the same $\mathsf{Rat}^{\uparrow}$ seed, the same constant $\mathsf{StreamName}^{\uparrow}$ witness, and the same $\mathsf{Real}^{\uparrow}$ seal row.

Apply Definition 372.4. Each diagonal window reads the same rational seed through the constant stream row and the associated $\mathsf{RegSeqRat}^{\uparrow}$ readback. Then apply Definition 372.10 to identify the completion-facing seal route with the displayed chain $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$.

Finally use Definition 372.14. Since the bridge exports no quotient stream, host limit, ambient completeness theorem, or host equality, there is no remaining stationary-family coordinate outside Q, S, R, and L. □

Theorem 372.17 (Regular Cauchy diagonal bridge route determinacy)**.** *For the standard* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *bridge source, bridge readback has exactly the displayed route through* $D = (Q, S, R, L, W, P, N)$*. Any bridge consumer read that recovers the public surface factors through the same source tuple, the same finite window ledger, and the same local* hsame*,* Cont*,* Pkg*, and* NameCert *provenance; there is no second bridge route outside* D*.*

Proof. Apply Definition 372.12. The bridge source is built from the seven displayed coordinates, the carrier classifier, the finite window route, selector stability, stationary compatibility, NameCert obligations, and the $\mathsf{Real}^{\uparrow}$ seal handoff.

Read the source back through Definition 372.13. The public readback recovers the same tuple $D = (Q, S, R, L, W, P, N)$ and preserves the selector classifier by componentwise hsame and Cont transport.

It remains to exclude competing routes. This is exactly the non-escape content of Definition 372.14: every bridge consumer read remains inside the displayed tuple, finite ledger, BEDC rows, and local provenance. Hence bridge readback is determined by D. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier bridge route determinacy`

Theorem 372.18 (Regular Cauchy diagonal mature consumer exhaustion)**.** *Every* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer read of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet factors through the displayed tuple* $D = (Q, S, R, L, W, P, N)$*, the finite window ledger, the diagonal-window coverage route of Definition 372.3, the seal handoff of Definition 372.8, and the bridge route determinacy of Definition 372.17. The consumer receives no quotient stream, selected host limit, ambient completeness row, or host equality from the diagonal packet.*

Proof. Start with an accepted diagonal packet and project its carrier through Definition 372.2. The available rows are the rational seed, stream-window row, regular rational readback, real-seal row, finite window ledger, package provenance, and local certificate provenance.

Apply Definition 372.3 to identify the visible finite window ledger, and apply Definition 372.8 to identify the $\mathsf{Real}^{\uparrow}$ seal read by downstream completion packets.

The remaining possible ambiguity is the bridge route by which the consumer reads the packet. Definition 372.17 identifies that route with the same displayed tuple $D = (Q, S, R, L, W, P, N)$, and Definition 372.14 excludes quotient streams, selected host limits, ambient completeness rows, and host equality. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier mature consumer exhaustion`

Theorem 372.19 (Regular Cauchy diagonal real-completion handoff)**.** *The displayed route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *is the completion-facing handoff from an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet into the current* $\mathsf{Real}^{\uparrow}$ *and completion phase boundary. The handoff is determined by stationary compatibility, stationary-family exhaustion, and bridge non-escape, and it introduces no completion object beyond the displayed seal row* L*.*

Proof. Begin with the stationary route supplied by Definition 372.4. It reads the same rational seed through the constant $\mathsf{StreamName}^{\uparrow}$ row and the associated $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback.

Apply Definition 372.16. This shows that the stationary packet has no family read outside Q, S, R, and the seal row L of $D = (Q, S, R, L, W, P, N)$.

Finally apply Definition 372.14. Since the bridge exports no quotient stream, selected host limit, ambient completeness theorem, or host equality, the only completion-facing object exposed by the route is the displayed $\mathsf{Real}^{\uparrow}$ seal row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier real completion handoff`

Theorem 372.20 (Regular Cauchy diagonal mature closure certificate)**.** *The* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *paper surface has a mature consumer package consisting of carrier habitation, selector stability, window coverage, stationary compatibility, seal handoff, the standard bridge source, readback exactness, bridge non-escape, mature consumer exhaustion, and real-completion handoff. Consequently every accepted diagonal selector packet consumed at the current* $\mathsf{Real}^{\uparrow}$ *completion boundary is accounted for by the displayed tuple* $D = (Q, S, R, L, W, P, N)$ *and its finite ledger.*

Proof. Collect the carrier and local certificate clauses first. Carrier habitation is supplied by the local obligation package of Definition 372.5, selector stability by Definition 372.7, window coverage by Definition 372.3, stationary compatibility by Definition 372.4, and seal handoff by Definition 372.8.

Next collect the bridge-facing clauses. The standard source is Definition 372.12, readback exactness is Definition 372.13, and non-escape is Definition 372.14.

It remains to connect the bridge surface to the completion consumers. This is exactly the mature consumer exhaustion of Definition 372.18 together with the real-completion handoff of Definition 372.19. These clauses package the accepted packet as the displayed tuple and finite ledger consumed at the current $\mathsf{Real}^{\uparrow}$ completion boundary. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier mature closure certificate`

Theorem 372.21 (Regular Cauchy diagonal source stability obligation)**.** *Let* $D = (Q, S, R, L, W, P, N)$ *be an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row. If the displayed* hsame *comparisons and* Cont *routes transport* Q*,* S*,* R*,* L*,* W*,* P*, and* N *to another accepted source row, then carrier admission, the diagonal selector classifier, the stationary rational route, and the* $\mathsf{Real}^{\uparrow}$ *seal route are transported through the same finite tuple. No transported source read leaves the rows named in* D*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier source stability obligation`

Proof. Project D through Definition 372.1 and Definition 372.2. The classifier compares only the rational seed, stream-window row, regular readback, seal row, finite window ledger, package row, and local naming row.

Apply Definition 372.7. The displayed hsame comparisons and Cont routes preserve the selector classifier through the $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ coordinates.

For the completion-facing read, apply Definition 372.8. The seal row L is reached only after the transported source rows, so source stability is exactly the finite transport of D. □

Theorem 372.22 (Regular Cauchy diagonal window ledger exactness)**.** *For every accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row* $D = (Q, S, R, L, W, P, N)$*, every diagonal consumer window is exactly one of the finite entries of the displayed ledger* W*. The entry names its* $\mathsf{StreamName}^{\uparrow}$ *window, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the stationary* $\mathsf{Rat}^{\uparrow}$ *seed route when present, and the* Cont *route by which the* $\mathsf{Real}^{\uparrow}$ *seal can later be read.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier window ledger exactness`

Proof. Start with a consumer window request. By Definition 372.3, the request factors through a displayed diagonal window, a stream-window row, a regular rational readback row, and the local provenance rows.

The carrier definition records no unlisted window source. Therefore the displayed window reached by the consumer is an entry of W, with the associated Cont route and readback rows already attached to that entry.

In the stationary case, Definition 372.4 identifies the same entry with the constant $\mathsf{Rat}^{\uparrow}$ seed route. The seal-facing route is then supplied by Definition 372.8, so the ledger entry is exact for downstream consumers. □

Theorem 372.23 (Regular Cauchy diagonal NameCert consumer surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}}$ *consumer surface for* $D = (Q, S, R, L, W, P, N)$ *consists exactly of carrier admission, source stability, finite window-ledger exactness, stationary rational compatibility, selector stability, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff. Each clause is read through* BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert *row already displayed by the source tuple.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier namecert consumer surface`

Proof. First collect the local certificate rows. Carrier admission and local naming obligations are Definition 372.5; source stability is Definition 372.21.

Next collect the selector-facing rows. Window-ledger exactness is Definition 372.22, stationary rational compatibility is Definition 372.4, and selector stability is Definition 372.7.

Finally attach the seal-facing read with Definition 372.8. These rows are all projections of the finite source tuple and its displayed transport routes, so the consumer surface has no quotient stream, selected host limit, ambient completeness theorem, or host equality. □

Theorem 372.24 (Regular Cauchy diagonal real-completion handoff package)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *source row receives the completion handoff through the tuple* $D = (Q, S, R, L, W, P, N)$*, the exact finite window ledger* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* L*. The handoff packages only the*

visible diagonal route and does not supply quotient streams, selected host limits, host equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier real completion handoff package`

Proof. Begin with the completion-consumer bridge of Definition 372.9. It routes every $\mathsf{RealDiagonalCompletion}^{\uparrow}$ or $\mathsf{RealCauchyCompletion}^{\uparrow}$ request through the displayed source tuple, the window coverage row, and the seal handoff.

Replace the coverage row by the exact ledger statement Definition 372.22. This identifies each requested window with an entry of W and fixes the associated $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ reads.

Then apply Definition 372.19. The completion-facing object is the displayed $\mathsf{Real}^{\uparrow}$ seal row L, reached through the finite route $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ where the stationary route applies. □

Theorem 372.25 (Regular Cauchy diagonal downstream-unblock package)**.** *The* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *downstream package for the real-completion diagonal route is the composition of source stability, window-ledger exactness, the NameCert consumer surface, and the real-completion handoff package. Consequently every downstream* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *read is accounted for by the displayed tuple* $D = (Q, S, R, L, W, P, N)$*, its finite ledger* W*, and the local* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. First apply Definition 372.21. This fixes the transport spine of the source tuple under the displayed hsame comparisons and Cont routes.

Next apply Definition 372.22. Every consumer window is one of the finite entries of W, with its stream-window route, regular readback, and seal-facing continuation already named.

The public certificate surface is then Definition 372.23. The real-completion read is Definition 372.24. Composing these rows gives the downstream package and excludes all non-displayed completion objects. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier downstream unblock package`

Theorem 372.26 (Regular Cauchy diagonal root source determinacy)**.** *For an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *root packet, the concrete source tuple* $D = (Q, S, R, L, W, P, N)$ *is determined as a displayed* BHist *row by the stationary* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *route, the componentwise* hsame *comparisons, the displayed* Cont *routes, the package row* Pkg*, and the local* NameCert *row. No second source for the same root packet can change* Q*,* S*,* R*,* L*,* W*,* P*, or* N *without changing one of those displayed rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier root source determinacy`

Proof. Project the root packet through Definition 372.1 and Definition 372.2. The carrier exposes exactly the seven rows Q, S, R, L, W, P, and N inside the displayed BHist packet.

The source-stability row of Definition 372.21 identifies transport of the tuple by the recorded hsame and Cont data. The stationary compatibility row of Definition 372.4 fixes the route from $\mathsf{Rat}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to the same $\mathsf{Real}^{\uparrow}$ seal.

Thus any competing root readback must agree on the carrier rows and on the Pkg and NameCert provenance. There is no hidden completion coordinate through which a different source tuple could be substituted. □

Theorem 372.27 (Regular Cauchy diagonal root selector totality)**.** *For every accepted root tuple $D = (Q, S, R, L, W, P, N)$ of* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *and every classifier-facing downstream request, the selector has a displayed finite response in the* BHist *ledger W. The response follows the route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *and carries only the visible* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data attached to D.*

Proof. Begin with a classifier-facing request and read the accepted packet through the carrier definition. The available selector data is the finite ledger W together with the stream row S, regular readback row R, seal row L, package row P, and naming row N.

Apply Definition 372.3. It routes the request to a displayed diagonal window, the matching $\mathsf{StreamName}^{\uparrow}$ row, and the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The selector-stability theorem Definition 372.7 transports this response across the recorded hsame and Cont witnesses. The real-seal handoff of Definition 372.8 then identifies the response as the same $\mathsf{Real}^{\uparrow}$ seal row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier root selector totality`

Theorem 372.28 (Regular Cauchy diagonal root window-ledger exactness)**.** *For a root packet with source tuple $D = (Q, S, R, L, W, P, N)$, every window read used by the Real and Completion handoff is exactly one displayed entry of the finite* BHist *ledger W. That entry names the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal route,* hsame *and* Cont *transport data,* Pkg *provenance, and local* $\mathsf{NameCert}$ *provenance; no additional window source is available.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier root window ledger exactness`

Proof. Read the downstream window through Definition 372.22. The theorem identifies the requested window with one of the finite entries in W and records the stream, regular rational, seal-facing, and provenance rows used by that entry.

The root source determinacy theorem above fixes the ambient tuple $D = (Q, S, R, L, W, P, N)$ in which this ledger lives. Hence the selected window cannot be re-read through another BHist source or through another package row.

Finally, Definition 372.14 excludes quotient streams, selected host limits, host equality, and ambient completeness rows. The ledger entry is therefore exact for the Real and Completion handoff. □

Theorem 372.29 (Regular Cauchy diagonal root seal route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported from a root tuple $D = (Q, S, R, L, W, P, N)$ factors through the displayed* BHist *route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$*, the exact finite ledger W, the visible* hsame *and* Cont *routes, the package row* Pkg*, and the local* $\mathsf{NameCert}$ *row. The root packet exports no quotient stream, selected limit, host real equality, or ambient completeness row for the seal.*

Proof. Start from the real-seal handoff Definition 372.8. It sends the seal request through the stationary rational row, the stream window, the regular rational readback, and the displayed $\mathsf{Real}^{\uparrow}$ seal row.

Use root window-ledger exactness to locate the requested observation inside W. This names the precise BHist entry and attaches the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows already carried by D.

The bridge non-escape theorem Definition 372.14 and mature consumer exhaustion Definition 372.18 rule out every remaining completion-side escape coordinate. Thus the seal route is exhausted by the displayed root packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier root seal route exhaustion`

Theorem 372.30 (Regular Cauchy diagonal root downstream-unblock package)**.** *The root obligation surface for* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *is the composition of source determinacy, selector totality, window-ledger exactness, seal route exhaustion, bridge non-escape, and mature consumer exhaustion for the concrete tuple* $D = (Q, S, R, L, W, P, N)$*. Consequently the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *handoff reads only the displayed* BHist *route* $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$*, the visible* hsame *and* Cont *transports, the* Pkg *row, and the local* $\mathsf{NameCert}$ *row.*

Proof. First assemble the root source, selector, and window rows. The preceding three theorems fix the tuple D, give a displayed selector response for each downstream request, and identify every requested window with an entry of W.

Next attach the seal boundary. Root seal route exhaustion identifies the completion-facing object as the same $\mathsf{Real}^{\uparrow}$ seal reached along the route $\mathsf{Rat}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$.

Finally compose Definition 372.14, Definition 372.18, and Definition 372.25. These rows show that the Real and Completion consumers obtain exactly the displayed packet and no external completion object. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier root downstream unblock package`

Theorem 372.31 (Regular Cauchy diagonal selector-fiber exhaustion)**.** *Every classifier-facing read of an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *diagonal packet factors through the displayed selector row of its source tuple* $D = (Q, S, R, L, W, P, N)$*. The factorization reads only the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*, the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* L*, and the visible* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance attached to* D*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier selector fiber exhaustion`

Proof. Project the accepted packet to the carrier of Definition 372.1. The carrier exposes exactly the rational seed row, the stream-window row, the regular readback row, the seal row, the finite window ledger, the package row, and the local naming certificate.

Apply Definition 372.3 and Definition 372.22. These clauses identify every classifier-facing window read with one of the selected entries of W and attach the matching $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ observations.

Finally apply Definition 372.7 and Definition 372.23. The remaining reads are precisely the visible transport, package, and naming rows, while the seal-facing coordinate is the displayed $\mathsf{Real}^{\uparrow}$ row supplied by the same source tuple. □

Theorem 372.32 (Regular Cauchy diagonal completion-seal composition)**.** *The* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer handoff for an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet is the composition of stationary seal exactness, the bridge obligation package, and the real-completion handoff package. This composition adds no host sequence equality, quotient stream, selected host limit, or ambient completeness data beyond the finite diagonal selector route and the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier completion seal composition`

Proof. Start with the stationary seal clause of Definition 372.4. It fixes the constant rational source route and the associated $\mathsf{RegSeqRat}^{\uparrow}$ readback used by the diagonal selector.

Attach the bridge obligations with Definition 372.12, Definition 372.13, and Definition 372.14. These results expose the bridge source, preserve the visible readback, and rule out completion data not carried by the displayed packet.

Now apply Definition 372.31 and Definition 372.24. The selector exhaustion supplies the finite packet route, and the handoff package identifies the completion-facing object as the same $\mathsf{Real}^{\uparrow}$ seal row. □

Theorem 372.33 (Regular Cauchy diagonal downstream route factorization)**.** *Every downstream consumer route from an accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet through* $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *factors through the finite diagonal selector route of Definition 372.31 and the completion-seal route of Definition 372.32. No downstream route obtains an additional carrier, classifier, host equality, quotient, or completion witness outside those two displayed routes.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier downstream route factorization`

Proof. First use Definition 372.18. It says that mature consumers of the diagonal packet read the same displayed tuple, finite ledger, readback row, seal row, and local provenance.

Next use Definition 372.25. The downstream package is already the composition of source stability, window-ledger exactness, the public $\mathsf{NameCert}$ surface, and the real-completion handoff package.

The $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ portions are therefore the selector-fiber route of Definition 372.31. The $\mathsf{RealCauchyCompletion}^{\uparrow}$ portion is the completion-seal composition of Definition 372.32, so every listed downstream route factors through those two routes and no extra object is exposed. □

Theorem 372.34 (Regular Cauchy diagonal formal target boundary)**.** *The formal target for* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *is the finite tuple* $D = (Q, S, R, L, W, P, N)$ *together with selector-fiber exhaustion, completion-seal composition, downstream route factorization, and bridge non-escape. A Lean target for this chapter should prove only this finite* BHist *route over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and should not prove quotient streams, selected host limits, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier formal target boundary`

Proof. Start with selector-fiber exhaustion from Definition 372.31. It identifies every classifier-facing read with the displayed tuple $D = (Q, S, R, L, W, P, N)$ and the finite window, dyadic, readback, seal, transport, package, and naming rows attached to it.

Next attach completion-seal composition from Definition 372.32. The completion-facing consumer reaches the same $\mathsf{Real}^{\uparrow}$ seal by the displayed diagonal selector route rather than by a host sequence equality or selected limit.

Now use downstream route factorization from Definition 372.33. It says that $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers all factor through the selector-fiber and completion-seal routes.

Finally apply bridge non-escape from Definition 372.14. Since the bridge exports no quotient stream, selected host limit, host equality, or ambient completeness theorem, the finite tuple and its displayed BHist route are the precise formal target boundary. □

Theorem 372.35 (Regular Cauchy diagonal name-normalization handoff)**.** *For every accepted* $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *packet whose displayed selector route is read by a* $\mathsf{RegularCauchyName}^{\uparrow}$ *normalized readback packet, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer handoff factors through exactly three finite surfaces: the diagonal selector-fiber route, the* $\mathsf{RegularCauchyName}^{\uparrow}$ *normalized window readback, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *downstream consumer handoff. The correspondence excludes quotient streams, host sequence equality, selected host limits, independent real carriers, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalUp.RegularCauchyDiagonalCarrier name normalization handoff`

Proof. Use the formal target boundary of Definition 372.34. It restricts the diagonal packet to the finite tuple $D = (Q, S, R, L, W, P, N)$ and the displayed BHist route over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

The diagonal contribution is the selector route of Definition 372.33. It says that every downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{RealCauchyCompletion}^{\uparrow}$ read from the accepted diagonal packet factors through the selector-fiber route and completion-seal route already present in the same finite packet.

The name contribution is the normalized readback package of Definition 371.45. That package reads the $\mathsf{RegularCauchyName}^{\uparrow}$ window through its displayed normalized rational-name surface, so the diagonal selector cannot acquire a second host stream, a quotient stream, or an independently selected limit while passing through the name packet.

Finally attach the completion consumer by Definition 360.20. Its handoff consumes the regular-family input through the displayed completion packet and does not add an independent real carrier or ambient completeness field. Ledger exhaustion and bridge non-escape leave only the three named surfaces, so the composed route is exactly the diagonal selector-fiber surface, the normalized window readback surface, and the downstream completion handoff surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ packet is generated from a stationary $\mathsf{Rat}^{\uparrow}$ seed row, a $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback, a $\mathsf{Real}^{\uparrow}$ seal row, finite diagonal-window Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is the standard bridge source, readback exactness, non-escape boundary, stationary-family exhaustion, and route determinacy of Definition 372.12, Definition 372.13, Definition 372.14, Definition 372.16, and Definition 372.17.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected host limits, ambient completeness, or bridge checking.

Upgrade path: Lean formalization and bridge checking remain for the same diagonal selector and completion-seal interface.

Chapter 373

A Concrete Naming Certificate for $\mathsf{RegularCauchyFamily}^{\uparrow}$

The $\mathsf{RegularCauchyFamily}^{\uparrow}$ interface is the named input certificate consumed by $\mathsf{RealCauchyCompletion}^{\uparrow}$. Following Bishop and Bridges, *Constructive Analysis*, 1985, it presents a regular Cauchy family as visible regular-rational member rows with an explicit Cauchy modulus, finite stream windows, dyadic observation ledgers, and diagonal-completion routes. The packet exports no quotient stream, no choice-selected limit, no host real carrier, and no ambient completeness object.

Each family member is read first through the regular-name discipline of chapter 371. Only after that sibling row has fixed the member surface do the modulus schedule, stream window, dyadic ledger, and diagonal route become consumable by the family packet, so the completion-facing handoff never treats a member as an unstructured rational stream.

Definition 373.1 (RegularCauchyFamily carrier)**.** *A* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, X, \mu, W, D, R, H, C, P, N)$$

with the following visible rows:

1. *I is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *member-index ledger inspected by the family consumer;*
2. *X assigns to each inspected row of I a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational member;*
3. *μ is the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule that selects a late member row for each requested precision;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window catalogue used to read the selected members;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger containing the rational endpoints, radius rows, comparison rows, and precision rows used by the member reads;*
6. *R is the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *diagonal-route ledger that records how the selected member windows feed the completion-facing diagonal representative;*
7. *H and C are respectively the displayed* hsame *comparisons and* Cont *routes used to transport the member, modulus, stream, dyadic, and diagonal rows;*
8. *P is the* Pkg *provenance naming* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, BHist, hsame, *and* Cont*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFamily}^{\uparrow}}$ *provenance row.*

The carrier deliberately has no coordinate for a quotient of streams, a selected limiting real, host sequence equality, or a choice principle. Every completion-facing datum is one of the displayed rows of F.

Lean4 (definition) `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier`

Definition 373.2 (RegularCauchyFamily classifier)**.** *For accepted carriers*

$$F = (I, X, \mu, W, D, R, H, C, P, N) \quad \text{and} \quad F' = (I', X', \mu', W', D', R', H', C', P', N'),$$

the classifier $\mathsf{RegularCauchyFamilyClassifier}(F, F')$ *compares only the displayed member-index ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *member rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledgers,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *diagonal routes, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It does not compare host functions extensionally and does not identify completed values by quotient equality.*

Lean4 (definition) `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyClassifier`

Theorem 373.3 (RegularCauchyFamily NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFamily}^{\uparrow}}$ *obligation sketch consists of five finite rows:*

1. *carrier habitation by the finite packet of Definition 373.1;*
2. *classifier stability under the displayed* hsame *and* Cont *transports of Definition 373.2;*
3. *modulus-window coverage: each precision request first reads μ, then one selected* $\mathsf{StreamName}^{\uparrow}$ *window, and then the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *member row;*
4. *dyadic ledger exactness: the endpoint, radius, comparison, and precision rows consumed by the selected member read are exactly the rows recorded in D;*
5. *diagonal-route compatibility: the completion-facing diagonal read uses only R together with the selected member, stream window, dyadic ledger, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier namecert obligation surface`

Proof. Carrier habitation is obtained by projecting the finite history packet and reading its named rows. The member-index ledger I, member table X, modulus μ, stream-window catalogue W, dyadic ledger D, diagonal-route ledger R, transport rows, package provenance, and local naming row are all explicit coordinates.

Classifier stability is componentwise. Transporting an accepted packet by the displayed hsame rows and then following the recorded Cont routes moves the member rows, modulus schedule, selected windows, dyadic observations, and diagonal routes together.

For modulus-window coverage, a consumer precision request reaches μ before it can inspect a member. The selected member read then passes through the corresponding $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ row named by X.

The dyadic and diagonal rows are exhausted by D and R. Repacking those rows with P and N gives the local naming certificate, and there is no remaining coordinate from which a quotient stream, a choice-selected limit, or a host real object could be read. □

Theorem 373.4 (RegularCauchyFamily diagonal handoff)**.** *For every accepted* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *carrier $F = (I, X, \mu, W, D, R, H, C, P, N)$ and every unary precision request n, the input consumed*

by RealCauchyCompletion$^{\uparrow}$ *factors through the explicit modulus row* $\mu(n)$*, one selected* RegSeqRat$^{\uparrow}$ *member row, one* StreamName$^{\uparrow}$ *finite window, the corresponding* DyadicRatCore$^{\uparrow}$ *observation ledger, and the* RealDiagonalCompletion$^{\uparrow}$ *route* R*. The handoff supplies a completion-facing regular Cauchy family without forming a quotient stream or selecting a limit outside the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier diagonal handoff`

Proof. Project the carrier of Definition 373.1. The modulus coordinate is the only row that selects a late family member for the precision request n.

The selected member is read through the member ledger X and the finite stream-window catalogue W. Its rational observation is the displayed RegSeqRat$^{\uparrow}$ row whose endpoint and comparison data are recorded by the dyadic ledger D.

The diagonal route R then repacks that same selected member read for RealCauchyCompletion$^{\uparrow}$. Because the route is mediated by H, C, P, and N, the completion consumer receives finite BEDC rows rather than a host sequence, quotient class, or selected limiting real. □

Theorem 373.5 (RegularCauchyFamily non-escape boundary)**.** *An accepted* RegularCauchyFamily$^{\uparrow}$ *carrier exports only the finite surface generated by* RegSeqRat$^{\uparrow}$*,* CauchyModulus$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RealDiagonalCompletion$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream, no choice-selected limit, no host real carrier, no host equality test, and no ambient completeness theorem.*

Proof. Inspect the coordinates of Definition 373.1. Each coordinate is a finite history row, a displayed transport row, a continuation route, package provenance, local naming provenance, or one of the named dependency certificates.

The classifier of Definition 373.2 uses only those coordinates. It never asks whether two host functions are extensionally equal and never compares completed values by a quotient relation.

The diagonal handoff of Definition 373.4 preserves the same boundary. Downstream completion reads receive the selected modulus, member, window, dyadic observation, and diagonal-route rows, so no hidden stream quotient, selected limit, host real carrier, host equality test, or ambient completeness object can escape. □

Theorem 373.6 (RegularCauchyFamily carrier transport stability)**.** *Let* $F = (I, X, \mu, W, D, R, H, C, P, N)$ *be an accepted* RegularCauchyFamily$^{\uparrow}$ *carrier. Any displayed* hsame *transport that preserves the* RegSeqRat$^{\uparrow}$ *member ledger,* CauchyModulus$^{\uparrow}$ *schedule,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observation ledger,* RealDiagonalCompletion$^{\uparrow}$ *route,* Cont *rows,* Pkg *provenance, and local* NameCert *row preserves the carrier predicate and the classifier of Definition 373.2.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier carrier transport stability`

Proof. Project the carrier by Definition 373.1. The rows I and X remain visible RegSeqRat$^{\uparrow}$ member data under the displayed hsame transport, and the schedule μ remains the same CauchyModulus$^{\uparrow}$ row after following the recorded Cont route.

The window catalogue W and dyadic ledger D transport componentwise. Thus each selected member read still passes through the same finite StreamName$^{\uparrow}$ window and the same DyadicRatCore$^{\uparrow}$ endpoint, radius, comparison, and precision rows.

The diagonal route R is carried through RealDiagonalCompletion$^{\uparrow}$, while P and N retain the package and local certificate provenance. Since the classifier compares exactly these displayed coordinates and no host extensional equality, the transported packet remains an accepted carrier with the same local comparison surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier transport stability`

Theorem 373.7 (RegularCauchyFamily modulus window exactness)**.** *For every accepted* RegularCauchyFamily$^{\uparrow}$ *carrier* $F = (I, X, \mu, W, D, R, H, C, P, N)$ *and every unary precision request* n*, the row* $\mu(n)$ *determines one displayed* RegSeqRat$^{\uparrow}$ *member read, one finite* StreamName$^{\uparrow}$ *window, the corresponding* DyadicRatCore$^{\uparrow}$ *observation rows, and the* RealDiagonalCompletion$^{\uparrow}$ *diagonal handoff route. The determination is entirely internal to the* BHist *packet, its* hsame *transports, its* Cont *routes,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier modulus window exactness`

Proof. The modulus coordinate is part of the carrier, so the request n first reads the displayed row $\mu(n)$. This row selects the late member named in the RegSeqRat$^{\uparrow}$ member table X over the inspected ledger I.

The selected member is read through the finite window catalogue W. The dyadic ledger D supplies exactly the rational endpoint, radius, comparison, and precision rows consumed by that windowed read.

Finally, the route R records how the same selected member read is handed to RealDiagonalCompletion$^{\uparrow}$. The transports H, continuations C, package provenance P, and local certificate N bind the route inside the finite packet, so no quotient stream, selected limit, host real carrier, or ambient completeness object is used. □

Theorem 373.8 (RegularCauchyFamily member selection determinacy)**.** *Let* $F = (I, X, \mu, W, D, R, H, C, P, N)$ *be an accepted* RegularCauchyFamily$^{\uparrow}$ *carrier, and let* n *be a unary precision request. Suppose two consumer reads both pass through the displayed row* $\mu(n)$ *and both claim to reach selected* RegSeqRat$^{\uparrow}$ *member rows, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observation ledgers, and the* RealDiagonalCompletion$^{\uparrow}$ *route. Then the two reads agree on the classifier-facing selected member, window, ledger, and diagonal-route surface of* F*. The determinacy is internal to the displayed carrier rows and does not assert quotient stream equality, host function equality, a selected limit, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier member selection determinacy`

Proof. Project the accepted carrier. The coordinates I, X, μ, W, D, and R are the only rows from which a precision request can obtain a member index, a selected regular-rational member, a finite stream window, a dyadic observation ledger, and a completion-facing diagonal route.

Apply Definition 373.7 to the request n. It identifies the displayed route through $\mu(n)$ as the unique carrier route that reaches the selected RegSeqRat$^{\uparrow}$ member read, its StreamName$^{\uparrow}$ window, the corresponding DyadicRatCore$^{\uparrow}$ observation rows, and the RealDiagonalCompletion$^{\uparrow}$ handoff.

If one consumer read is transported before comparison, apply Definition 373.6. The transported read preserves the same member ledger, modulus schedule, window catalogue, dyadic ledger, diagonal route, continuation rows, package row, and local naming row, so the classifier still sees the same selected surface.

Finally use Definition 373.5. Since the carrier exports no external selector, quotient stream, selected limit, host equality test, or ambient completeness object, there is no second source from which a different member/window/ledger route could be read. □

Theorem 373.9 (RegularCauchyFamily scoped route package)**.** *For every accepted* RegularCauchyFamily$^{\uparrow}$ *carrier* $F = (I, X, \mu, W, D, R, H, C, P, N)$*, the carrier row, classifier row, local* NameCert *obligations, diagonal handoff, non-escape boundary, transport stability, modulus-window exactness,*

and member-selection determinacy are exhausted by the displayed RegSeqRat$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealDiagonalCompletion$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. This is the exact scoped dependency surface supplied to* RealCauchyCompletion$^{\uparrow}$*: it exposes selected member windows, dyadic observation ledgers, diagonal handoff routes, and naming provenance, but no quotient stream, selected limit, host real carrier, host equality test, or ambient completeness object.*

Proof. The carrier and classifier are the finite tuple and componentwise comparison of Definition 373.1 and Definition 373.2. Projecting their coordinates gives only the member ledger I, member table X, modulus schedule μ, window catalogue W, dyadic ledger D, diagonal route R, displayed transports H, continuation routes C, package provenance P, and local naming row N.

The local obligation rows are those of Definition 373.3. The diagonal consumer route is Definition 373.4, and the exported boundary is Definition 373.5; together they identify the finite rows that reach RealCauchyCompletion$^{\uparrow}$.

Transport and precision selection are controlled by Definition 373.6 and Definition 373.7. Member choice at a fixed request is then determined by Definition 373.8.

After these projections, every named route has been accounted for by one of the displayed BEDC rows. Since the carrier has no remaining coordinate and the classifier never compares host presentations, there is no source for a quotient stream, selected limit, host real carrier, host equality test, or ambient completeness object. □

Theorem 373.10 (RegularCauchyFamily obligation closure package)**.** *The carrier, classifier,* NameCert *obligation rows, diagonal handoff, non-escape boundary, carrier transport stability, modulus-window exactness, member-selection determinacy, and scoped route package jointly supply the paper-side obligation package for* RegularCauchyFamily$^{\uparrow}$*. The package is grounded only in* BHist, hsame, Cont, Pkg, NameCert, RegSeqRat$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RealDiagonalCompletion$^{\uparrow}$.

Proof. The carrier and classifier are Definition 373.1 and Definition 373.2. They expose the finite member ledger, member rows, modulus schedule, stream windows, dyadic observations, diagonal route, transport rows, continuation rows, package provenance, and local naming row.

The local NameCert obligations are supplied by Definition 373.3. They bind carrier habitation, classifier stability, modulus-window coverage, dyadic ledger exactness, and diagonal-route compatibility as the rows consumed by downstream completion.

The handoff and boundary are Definition 373.4 and Definition 373.5. They ensure that RealCauchyCompletion$^{\uparrow}$ receives a displayed regular Cauchy family route and no quotient stream, choice-selected limit, host real carrier, host equality test, or ambient completeness theorem.

The stability, exactness, and determinacy rows are Definition 373.6, Definition 373.7, and Definition 373.8. The route package of Definition 373.9 binds these rows to the exact dependency surface consumed by RealCauchyCompletion$^{\uparrow}$. These rows assemble the stated obligation package without adding any coordinate outside the listed BEDC objects. □

Theorem 373.11 (RegularCauchyFamily standard bridge boundary)**.** *The public* RegularCauchyFamily$^{\uparrow}$ *packet bridges to the* RealCauchyCompletion$^{\uparrow}$ *input schema through exactly the displayed* RegSeqRat$^{\uparrow}$ *member rows, the* CauchyModulus$^{\uparrow}$ *schedule, the* StreamName$^{\uparrow}$ *finite windows, the* DyadicRatCore$^{\uparrow}$ *observation ledgers, the* RealDiagonalCompletion$^{\uparrow}$ *diagonal route, and the visible* hsame, Cont, Pkg, *and* NameCert *rows. This standard bridge exports the completion-facing regular-family input and no quotient stream, selected limit, host real equality, host sequence equality, or ambient completeness principle.*

Proof. Start with the diagonal handoff of Definition 373.4. It identifies the completion-facing route as the selected modulus row, one RegSeqRat$^{\uparrow}$ member, one finite StreamName$^{\uparrow}$ window, the matching DyadicRatCore$^{\uparrow}$ ledger, and the RealDiagonalCompletion$^{\uparrow}$ route.

Next apply Definition 373.7 and Definition 373.8. These results make the selected member, window, ledger, and diagonal route determined by the displayed carrier rows for each precision request.

Finally use Definition 373.10. The carrier, classifier, transport, package, and local certificate rows are already the whole public surface. Thus the bridge has no coordinate from which to read a quotient stream, selected limit, host real equality, host sequence equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamily standard bridge boundary`

Theorem 373.12 (RegularCauchyFamily standard bridge carrier)**.** *The carrier rows of Definition 373.1 supply the bridge-facing source surface for* RealCauchyCompletion$^{\uparrow}$*. That source surface consists exactly of the* RegSeqRat$^{\uparrow}$ *member rows, the* CauchyModulus$^{\uparrow}$ *schedule, the* StreamName$^{\uparrow}$ *finite windows, the* DyadicRatCore$^{\uparrow}$ *observation ledger, the* RealDiagonalCompletion$^{\uparrow}$ *routes,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in the accepted* RegularCauchyFamily$^{\uparrow}$ *packet.*

Proof. Project the carrier and classifier of Definition 373.1 and Definition 373.2. They expose exactly the member ledger, member table, modulus schedule, stream windows, dyadic ledger, diagonal route, transport rows, continuation rows, package row, and local naming row.

The diagonal-facing part is identified by Definition 373.4. The selected member route is then pinned down by Definition 373.7 and Definition 373.8.

Finally apply Definition 373.9 and Definition 373.5. These results show that the listed rows are the whole bridge-facing source surface and that no quotient stream, selected limit, host equality, or ambient completeness object is available to the completion consumer. □

Theorem 373.13 (RegularCauchyFamily completion schema bridge)**.** *For an accepted* RegularCauchyFamily$^{\uparrow}$ *carrier, the diagonal handoff, modulus-window exactness, member-selection determinacy, and scoped route package assemble the* RealCauchyCompletion$^{\uparrow}$ *completion schema bridge. The bridge is built without quotient streams, selected limits, host sequence equality, host real equality, or an external completeness carrier.*

Proof. Begin with Definition 373.4. It gives the completion-facing route through the displayed modulus row, selected RegSeqRat$^{\uparrow}$ member, finite StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ ledger, and RealDiagonalCompletion$^{\uparrow}$ route.

Next use Definition 373.7 and Definition 373.8. The selected window and ledger are determined by the carrier rows for the precision request, so the completion schema receives a stable finite route rather than a choice-selected stream representative.

The route package of Definition 373.9 supplies the remaining BHist, hsame, Cont, Pkg, and NameCert rows. The non-escape boundary of Definition 373.5 then rules out every unlisted completion source. □

Theorem 373.14 (RegularCauchyFamily bridge consumer exactness)**.** *Every* RealCauchyCompletion$^{\uparrow}$ *consumer read of the displayed* RegularCauchyFamily$^{\uparrow}$ *bridge factors through exactly the displayed modulus row, selected member row, finite window row, dyadic ledger row, diagonal route row, transport rows, package row, and local* NameCert *row. No consumer read factors through a quotient stream, a selected limit, host sequence equality, host real equality, or ambient completeness.*

Proof. Read the consumer route through Definition 373.1. The only available rows for a precision request are I, X, μ, W, D, R, H, C, P, and N.

Apply Definition 373.4 to identify the completion-facing chain. Apply Definition 373.7 and Definition 373.8 to make the selected member, window, dyadic ledger, and diagonal route exact.

The scoped route package of Definition 373.9 repacks the same rows for public consumption. Finally, Definition 373.5 excludes every route outside the displayed modulus, member, window, ledger, diagonal, transport, package, and naming surface. □

Theorem 373.15 (RegularCauchyFamily public bridge boundary)**.** *The public* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *packet factors from the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *member rows,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *routes,* BHist *ledger rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row into the completion-facing interface consumed by* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. The factorization exports no quotient stream, selected limit, host real equality, host sequence equality, or ambient completeness principle.*

Proof. Project the accepted packet using Definition 373.1 and Definition 373.2. The available coordinates are precisely the member table, modulus schedule, window catalogue, dyadic ledger, diagonal route, history transports, continuations, package row, and local naming row.

Apply Definition 373.11 and Definition 373.12. They identify the completion-facing interface as the same finite surface over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

The non-escape boundary of Definition 373.5 removes every unlisted completion source, so the public bridge boundary contains no quotient stream, selected limit, host equality row, or ambient completeness datum. □

Theorem 373.16 (RegularCauchyFamily finite induction route)**.** *Let* $F = (I, X, \mu, W, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *carrier. Induction on the finite inspected member ledger* I *reconstructs exactly the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *member rows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule reads, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, and the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *handoff route consumed by* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. The base case exposes no member read. In the successor case, adding one inspected ledger entry adds one displayed member-window-dyadic-diagonal route and preserves the non-escape boundary.*

Proof. Project the finite packet by Definition 373.1. The induction predicate says that every inspected prefix of I exposes only the corresponding rows of X, the schedule reads of μ, the finite windows in W, the dyadic observations in D, and the diagonal route in R, together with the displayed transport, package, and local naming rows.

For the empty inspected ledger, there is no member index to read. Thus no $\mathsf{RegSeqRat}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation, or $\mathsf{RealDiagonalCompletion}^{\uparrow}$ handoff can be produced except by an existing displayed coordinate of the packet, and no completion-facing route is exported.

For the successor step, assume the inspected prefix already reconstructs exactly its visible consumer routes. The next ledger entry is read through the modulus-window exactness of Definition 373.7: the schedule row selects one $\mathsf{RegSeqRat}^{\uparrow}$ member, one finite $\mathsf{StreamName}^{\uparrow}$ window, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ observation rows, and the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ route.

Member-selection determinacy, Definition 373.8, identifies this newly exposed route with the unique classifier-facing route for that precision request. The scoped route package of Definition 373.9 then repacks the old prefix routes and the new route into the same finite dependency surface.

Finally apply the non-escape boundary of Definition 373.5. The successor route adds only the displayed member-window-dyadic-diagonal row, so the induction preserves the absence of quotient streams, selected limits, host equality tests, and ambient completeness objects. □

Theorem 373.17 (RegularCauchyFamily downstream consumer exhaustion)**.** *Every* RealCauchyCompletion$^{\uparrow}$ *consumer route from an accepted* RegularCauchyFamily$^{\uparrow}$ *packet is exhausted by the selected member, window, ledger, and diagonal rows already named in that packet, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. There is no downstream route through a hidden stream representative, selected limit, host equality test, or ambient completeness object.*

Proof. Begin with Definition 373.14. It states that every consumer read factors through the displayed modulus, member, window, dyadic ledger, diagonal route, transport, package, and naming rows.

Use Definition 373.7 and Definition 373.8 to pin the selected member and finite window for each precision request. The dyadic observation ledger and diagonal route are then the rows named by Definition 373.4.

Finally apply Definition 373.15. The public bridge boundary has no remaining coordinate outside those rows, so the downstream consumer surface is exhausted by the named packet. □

Theorem 373.18 (RegularCauchyFamily completion consumer factorization)**.** *The public bridge boundary and downstream consumer exhaustion assemble the bridge package consumed by* RealCauchyCompletion$^{\uparrow}$*: each completion consumer first reads the* RegSeqRat$^{\uparrow}$ *member row,* CauchyModulus$^{\uparrow}$ *schedule,* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *observation, and* RealDiagonalCompletion$^{\uparrow}$ *route, and then transports only along the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamily completion consumer factorization`

Proof. Use Definition 373.15 as the source boundary. It lists the exact public rows available to the completion-facing interface.

Use Definition 373.17 as the consumer boundary. It shows that each downstream route is already one of the member, schedule, window, ledger, diagonal, transport, package, or naming rows.

Composing the two boundaries yields the stated factorization. The composition adds no new carrier coordinate and therefore cannot introduce quotient streams, selected limits, host equality tests, or ambient completeness objects. □

Closure status for NameCert$_{\text{RegularCauchyFamily}^{\uparrow}}$.

Constructive story: A RegularCauchyFamily$^{\uparrow}$ packet is generated from finite RegSeqRat$^{\uparrow}$ member rows, an explicit CauchyModulus$^{\uparrow}$ schedule, StreamName$^{\uparrow}$ finite windows, DyadicRatCore$^{\uparrow}$ observation ledgers, RealDiagonalCompletion$^{\uparrow}$ diagonal routes, displayed hsame and Cont rows, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyFamily}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 373.15, Definition 373.17, and Definition 373.18 export the public consumer surface and paper bridge sent to RealCauchyCompletion$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyFamily}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyFamilyUp.RegularCauchyFamilyCarrier\_namecert\_obligation\_surface`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, selected host limits, host real equality, or bridge checking.

Upgrade path: Bridge checking requires a Lean bridge theorem over the same RegularCauchyFamily consumer boundary.

Chapter 374

A Concrete Naming Certificate for $\mathsf{DiagonalLimit}^{\uparrow}$

The $\mathsf{DiagonalLimit}^{\uparrow}$ packet names the concrete regular-Cauchy-family diagonal construction that sits above the finite diagonal selectors and below the $\mathsf{Real}^{\uparrow}$ consumer seal. Following the Bishop–Bridges constructive-analysis treatment of Cauchy completion, it keeps the inspected regular rational family, explicit modulus rows, stream windows, dyadic readback ledger, diagonal completion packet, and real seal as visible certificate data. It does not quotient regular streams, select a host limit, or import ambient real completeness.

The diagonal packet consumes the tail-comparison surface of chapter 387 before its $\mathsf{RealDiagonalCompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows are read. The dependency is the finite modulus/window/tail-comparison ledger, not quotient stream equality or selected limit data.

Definition 374.1 (DiagonalLimit regular-Cauchy-family carrier)**.** *A* $\mathsf{DiagonalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, M, S, D, C, E, P, N)$$

with the following visible coordinates:

1. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$*-indexed regular Cauchy family whose inspected members are displayed as regular rational rows;*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row selecting a late enough family member for each requested consumer precision;*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of windows read by the diagonal construction;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger attached to the selected rational readbacks;*
5. *C is the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet that repacks the selected diagonal rows as a completion-facing finite surface;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed to downstream consumers;*
7. *P is the* Pkg *provenance naming the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependencies;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$ *provenance row.*

Two accepted packets are classified together exactly when the displayed family rows, modulus rows, stream windows, dyadic ledgers, diagonal-completion packets, real seals, package rows, and local certificate rows are transported componentwise by the recorded hsame *comparisons and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitUp.DiagonalLimitTasteGate single carrier alignment`

Theorem 374.2 (DiagonalLimit NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^\uparrow}$ *obligation surface consists of carrier habitation by the finite packet of Definition 374.1, componentwise classifier transport through displayed* hsame *and* Cont *rows, diagonal regularity bounds over the explicit* $\mathsf{CauchyModulus}^\uparrow$ *row, handoff to* $\mathsf{RealDiagonalCompletion}^\uparrow$ *and* $\mathsf{Real}^\uparrow$*, and ledger non-escape over* $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, BHist, Pkg, *and* NameCert.

Proof. Carrier habitation is the packet assembly in Definition 374.1. The visible rows name the regular Cauchy family, the modulus, the finite windows, the dyadic observations, the diagonal completion packet, the real seal, and the local provenance.

Classifier transport is componentwise. A permitted replacement is either an hsame comparison of one displayed row or a Cont route identifying where that row is read inside the finite packet, so transport preserves the same carrier boundary.

The remaining obligations are the diagonal regularity bound, the completion handoff, and non-escape boundary recorded in the following theorems. Together they list exactly the rows consumed by the local naming certificate. □

Theorem 374.3 (DiagonalLimit diagonal regularity bound)**.** *For every accepted* $\mathsf{DiagonalLimit}^\uparrow$ *carrier and every displayed unary precision request* n*, the modulus row* M *selects a finite family member and a finite* $\mathsf{StreamName}^\uparrow$ *window such that the selected* $\mathsf{RegSeqRat}^\uparrow$ *readback is bounded by the dyadic observation ledger* D *at the requested precision. The bound is transported only through the packet's displayed* hsame *comparisons and* Cont *routes.*

Proof. Project the accepted packet through Definition 374.1. The precision request is answered by the explicit $\mathsf{CauchyModulus}^\uparrow$ coordinate, so the selected family member is a displayed row rather than a hidden limit choice.

The selected member is read through the finite $\mathsf{StreamName}^\uparrow$ schedule. Its rational observation is recorded in the $\mathsf{RegSeqRat}^\uparrow$ family row and the accompanying $\mathsf{DyadicRatCore}^\uparrow$ ledger.

Since the classifier transports only displayed rows through hsame and Cont, the same bound follows after any accepted transport. No quotient stream or host real endpoint is used to obtain the bound. □

Theorem 374.4 (DiagonalLimit finite-family induction route)**.** *For every accepted* $\mathsf{DiagonalLimit}^\uparrow$ *carrier*

$$L = (F, M, S, D, C, E, P, N)$$

and every finite displayed family of $\mathsf{RegSeqRat}^\uparrow$ *rows inspected by* F*, the diagonal route to the* $\mathsf{Real}^\uparrow$ *consumer seal is generated by induction over the finite family index. At each index the induction state consists exactly of the displayed family row, the dyadic tail budget supplied through* D*, the finite* $\mathsf{StreamName}^\uparrow$ *window supplied through* S*, the* $\mathsf{RegSeqRat}^\uparrow$ *readback, the* $\mathsf{RealDiagonalCompletion}^\uparrow$ *seal factor in* C*, the* $\mathsf{Real}^\uparrow$ *seal row* E*, the visible* hsame *transport, the visible* Cont *replay route, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^\uparrow}$ *row* N*.*

The construction exports the same diagonal route after any accepted packet transport. It does not identify quotient streams, select an ambient limit, prove uniqueness of limits, or construct a host $\mathsf{Real}^\uparrow$ *object outside the displayed completion seal.*

Proof. Project the accepted carrier through Definition 374.1. The finite family is a visible list of inspected $\mathsf{RegSeqRat}^\uparrow$ rows, so the induction base is the empty inspected prefix together with the already displayed package and local certificate rows.

For the induction step, apply Definition 374.3 to the next precision request. The modulus row M selects the next finite family member and the finite $\mathsf{StreamName}^{\uparrow}$ window; the selected rational readback is recorded in the $\mathsf{RegSeqRat}^{\uparrow}$ row and bounded by the dyadic ledger D.

The step then repacks the selected readback into the completion-facing surface. Definition 374.5 records that this repacking passes through the same terminal precision, completion packet, $\mathsf{Real}^{\uparrow}$ seal, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$ row.

Transport is handled componentwise. The induction hypothesis is moved only along the displayed hsame comparisons and replayed only along the displayed Cont routes, so each finite prefix remains inside the same carrier boundary.

Finally, Definition 374.8 identifies the seal factor used by every consumer read, and Definition 374.6 excludes quotient stream equality, selected ambient limits, uniqueness of limits, and host real construction. Thus the full diagonal route is obtained by finite induction over the displayed family rows. □

Theorem 374.5 (DiagonalLimit Real-Cauchy handoff boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{DiagonalLimit}^{\uparrow}$ *packet factors through the same terminal precision, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet,* $\mathsf{Real}^{\uparrow}$ *seal, visible* hsame *rows, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$ *row.*

Proof. Begin with the selected diagonal window supplied by Definition 374.3. It exposes the precision, modulus-selected member, stream window, regular rational readback, and dyadic observation ledger.

The $\mathsf{RealDiagonalCompletion}^{\uparrow}$ coordinate C repacks exactly those displayed rows as the completion-facing finite surface. The $\mathsf{Real}^{\uparrow}$ seal E is attached to that same completion packet and not to an independent host real object.

Repacking the visible rows with the package and local certificate provenance gives the stated consumer handoff. The route has no coordinate for quotient equality, selected limits, uniqueness of limits, or ambient completeness. □

Theorem 374.6 (DiagonalLimit non-escape boundary)**.** *An accepted* $\mathsf{DiagonalLimit}^{\uparrow}$ *packet exports only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. It exports no quotient of regular streams, no host real equality, no selected limit, no uniqueness theorem for limits, and no ambient real-completeness principle.*

Proof. Inspect the carrier coordinates of Definition 374.1. Each coordinate is a displayed BEDC row or a local provenance row named in the packet.

The regularity theorem accounts for every precision-sensitive rational read, and the handoff theorem accounts for every $\mathsf{Real}^{\uparrow}$ consumer read. Thus every public route through the packet has already passed through visible finite data.

None of the carrier coordinates is a quotient class, a host-level equality witness, a selected limit object, a uniqueness theorem, or an ambient completeness field. Those objects cannot be recovered from the packet boundary. □

Theorem 374.7 (DiagonalLimit obligation transport scope)**.** *For any two accepted* $\mathsf{DiagonalLimit}^{\uparrow}$ *packets, accepted packet transport preserves the carrier of Definition 374.1 and its classifier only through the displayed* hsame *comparisons and* Cont *routes on the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$ *rows.*

Proof. Project both accepted packets through Definition 374.1. The carrier coordinates are exactly the displayed regular family, modulus, stream schedule, dyadic ledger, completion packet, real seal, package provenance, and local certificate row.

The classifier clause in the carrier definition admits only componentwise hsame comparisons and Cont read routes for those coordinates. Thus transport can rename or route a displayed row, but it cannot introduce a quotient stream, selected host limit, or ambient real object.

The precision-sensitive part of the carrier remains controlled by Definition 374.3, and the downstream route remains the handoff of Definition 374.5. The exclusions in Definition 374.6 show that no further row participates in the accepted transport scope. □

Theorem 374.8 (DiagonalLimit completion seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{DiagonalLimit}^{\uparrow}$ *packet factors through its* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows selected by the packet classifier.*

Proof. Start with an accepted packet and project its carrier as in Definition 374.1. The $\mathsf{Real}^{\uparrow}$ seal coordinate is not free-standing; it is one of the visible packet rows paired with the completion packet coordinate.

Apply Definition 374.3 to obtain the selected regular rational readback, stream window, and dyadic ledger for the requested precision. Then apply Definition 374.5, which routes the consumer read through the same terminal precision, completion packet, seal row, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$ row.

The non-escape boundary of Definition 374.6 rules out any alternate host-real or quotient-stream factor. Hence every seal read factors through the displayed completion packet and readback rows. □

Lean4 ✓ `BEDC.Derived.DiagonalLimitUp.DiagonalLimitCarrier completion seal factorization`

Theorem 374.9 (DiagonalLimit dependency stability package)**.** *The dependency package over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, Pkg, *and* NameCert *is stable under the displayed* $\mathsf{DiagonalLimit}^{\uparrow}$ *classifier.*

Proof. The package dependencies are exactly the coordinates and provenance rows listed in Definition 374.1. Stability therefore reduces to checking that each dependency row remains inside the same accepted packet boundary after classifier transport.

Definition 374.7 identifies that transport boundary as the displayed hsame and Cont rows only. Definition 374.3 keeps the regular-family, modulus, stream, and dyadic rows stable for every precision request.

Definition 374.5 keeps the completion and $\mathsf{Real}^{\uparrow}$ consumer route attached to the same provenance rows. Finally, Definition 374.6 excludes every dependency outside the named package, so the dependency package is stable under the displayed classifier. □

Theorem 374.10 (DiagonalLimit public export)**.** *The carrier of Definition 374.1, the NameCert surface of Definition 374.2, the diagonal regularity bound of Definition 374.3, the finite-family induction route of Definition 374.4, the real-Cauchy handoff boundary of Definition 374.5, the non-escape boundary of Definition 374.6, the transport scope of Definition 374.7, the completion seal factorization of Definition 374.8, and the dependency stability package of Definition 374.9 assemble the public* $\mathsf{DiagonalLimit}^{\uparrow}$ *export.*

The export consists of the displayed regular-family rows, explicit modulus row, finite stream-window schedule, dyadic readback ledger, diagonal completion packet, $\mathsf{Real}^{\uparrow}$ *seal row, component-wise transport, continuation routes, package provenance, and local naming row. It does not assert* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite carrier

$$L = (F, M, S, D, C, E, P, N).$$

The carrier and NameCert surface expose the regular-family, modulus, stream-window, dyadic-ledger, completion-packet, real-seal, package, and local naming rows.

The diagonal regularity and finite-family induction theorems route each requested read through the displayed modulus, selected stream window, rational readback, dyadic ledger, completion packet, and seal row. The handoff, transport, and dependency theorems keep those rows stable under the accepted hsame and Cont routes.

The non-escape and seal-factorization theorems exclude quotient streams, selected host limits, uniqueness of limits, host real construction, and ambient completeness. Hence the public export is the displayed finite diagonal-limit surface alone, with no claim of $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalLimit}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$-indexed regular Cauchy family, an explicit $\mathsf{CauchyModulus}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledgers, a $\mathsf{RealDiagonalCompletion}^{\uparrow}$ packet, a $\mathsf{Real}^{\uparrow}$ seal row, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope registers the carrier, displayed classifier route, diagonal regularity bound, completion-to-$\mathsf{Real}^{\uparrow}$ handoff boundary, ledger non-escape surface, accepted transport scope, seal factorization, and dependency stability over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiagonalLimit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DiagonalLimitUp.DiagonalLimitCarrier\_completion\_seal\_factorization`

Bridge status: none

Origin: human

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of regular streams, uniqueness of limits, host real construction, or bridge checking.

Upgrade path: No further theory-axis upgrade is pending; bridge checking remains a separate formal-axis obligation over the same displayed regular-family, modulus, stream-window, dyadic-ledger, completion-packet, $\mathsf{Real}^{\uparrow}$, Pkg, and NameCert rows.

Chapter 375

A Concrete Naming Certificate for $\mathsf{DiagonalModulus}^{\uparrow}$

The $\mathsf{DiagonalModulus}^{\uparrow}$ packet names the selector modulus used when regular rational streams are read along a diagonal and then sealed as real-name data. Following Bishop–Bridges constructive analysis, the packet keeps the modulus explicit: a consumer precision is converted into a displayed threshold, a finite stream window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and a $\mathsf{Real}^{\uparrow}$ seal row. It does not classify by quotient streams, host real equality, selected limits, or ambient completeness.

The $\mathsf{DiagonalModulus}^{\uparrow}$ route consumes the finite-window carrier of chapter 126, the regular rational readback of chapter 301, the displayed real seal of chapter 38, and the diagonal completion route of chapter 360. It also depends on $\mathsf{DyadicRatCore}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularityModulus}^{\uparrow}$, and $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, while still excluding quotient streams, host real equality, selected limits, and ambient completeness.

Definition 375.1 (DiagonalModulus carrier)**.** *A* $\mathsf{DiagonalModulus}^{\uparrow}$ *carrier row is a finite* BHist *packet*

$$\Delta = (n, \mu, w, r, d, e, P, N)$$

with the following visible coordinates:

1. *n is the unary precision request issued by the downstream real-name reader;*
2. *μ is the selected modulus threshold witnessing how far the diagonal read must inspect regular rational data at precision n;*
3. *w is the* $\mathsf{StreamName}^{\uparrow}$ *window selected from μ and n;*
4. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained inside the window w;*
5. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the dyadic rational bounds used by that readback;*
6. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed to the consumer after the readback is displayed;*
7. *P is the* Pkg *provenance row naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularityModulus}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *dependencies;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalModulus}^{\uparrow}}$ *provenance row.*

The carrier contains no quotient of streams, no host-level equality of reals, no selected limit object, and no completeness theorem field.

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusTasteGate single carrier alignment`

Definition 375.2 (DiagonalModulus classifier)**.** *Two* $\mathsf{DiagonalModulus}^{\uparrow}$ *carrier rows are classified together exactly when their displayed precision requests, modulus thresholds, stream windows, regular-sequence readbacks, dyadic observation ledgers, real-seal rows, package rows, and local certificate rows are transported componentwise by the recorded* hsame *comparisons and* Cont *routes. The classifier is not allowed to compare hidden sequences, quotient representatives, host real endpoints, or unrecorded limit selections.*

Theorem 375.3 (DiagonalModulus window selector totality)**.** *For every accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *carrier row and every displayed unary precision request* n*, the selected modulus threshold* μ *determines a finite* $\mathsf{StreamName}^{\uparrow}$ *window* w *and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r *inside that window. The selector therefore returns visible finite data for every consumer precision request.*

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket window selector total`
↪ variant `BEDC.Derived.DiagonalModulusUp.DiagonalModulusWindowCarrier selector total`

Proof. Project the carrier row of Definition 375.1. The precision coordinate n and the threshold coordinate μ are stored before any readback, so the window selection is read from displayed data rather than from an external sequence object.

The $\mathsf{StreamName}^{\uparrow}$ coordinate w records the finite window selected by that threshold. The readback coordinate r is then attached to the same window, and the allowed transport routes are exactly the hsame and Cont entries recorded by the carrier provenance.

Thus a consumer precision request always lands in a visible stream window and a visible regular rational row. No hidden limit selector is consulted. □

Theorem 375.4 (DiagonalModulus RegSeqRat handoff)**.** *Every* $\mathsf{DiagonalModulus}^{\uparrow}$ *read that reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *factors through the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the displayed readback row* r*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d *attached to that readback. The handoff exports only finite rational evidence and the recorded classifier routes.*

Proof. Start from an accepted carrier and a selected window. By Definition 375.3, the window and the regular rational readback are already displayed as coordinates of the same finite packet.

The dyadic ledger d is stored beside r. It records the rational observations and radius data used by the readback, so transport of the readback also transports the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ evidence.

Consequently the handoff into $\mathsf{RegSeqRat}^{\uparrow}$ is a finite route through w, r, and d. It does not expose host sequence equality or quotient equality of regular streams. □

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket regseqrat handoff`

Theorem 375.5 (DiagonalModulus RealUp seal non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal row* e *of an accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *packet can be consumed only after the precision request, modulus threshold, stream window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been displayed. The seal therefore exports a real-name surface without exporting selected limits, quotient equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket real seal nonescape`

Proof. Read the coordinates of Definition 375.1 in order. The seal row appears after the precision, threshold, window, readback, and dyadic ledger rows, so any consumer route to e must pass through those displayed coordinates.

The handoff theorem identifies the preceding rational evidence. Once that evidence is displayed, the seal row records only the $\mathsf{Real}^{\uparrow}$ consumer boundary attached to it.

Therefore the seal cannot be used as an escape to an unrecorded completion object. It withholds selected limits, quotient stream equality, host real equality, and ambient completeness data. □

Theorem 375.6 (DiagonalModulus NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalModulus}^{\uparrow}}$ *obligation surface consists of carrier habitation, displayed classifier transport, total finite-window selection,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through dyadic observation rows,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, and package dependency stability over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularityModulus}^{\uparrow}$, $\mathsf{RegularCauchyDiagonal}^{\uparrow}$, BHist, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket namecert obligation surface`

Proof. Carrier habitation is the packaging operation of Definition 375.1. The classifier row is the componentwise route of Definition 375.2, using only displayed hsame, Cont, Pkg, and dependency rows.

Total finite-window selection is exactly Definition 375.3. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is supplied by Definition 375.4, which keeps each readback attached to its stream window and dyadic observation ledger.

The seal boundary is supplied by Definition 375.5. Together these rows list the local certificate obligations and exclude hidden streams, quotient representatives, selected limits, host real equality, and ambient completeness fields. □

Theorem 375.7 (RealCauchyCompletion modulus classifier transport)**.** *A* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer may use an accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *packet by transporting its classifier through the displayed precision request, modulus threshold, stream window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, dyadic ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows. The transport supplies the diagonal modulus required by the completion packet without adding quotient streams, host real equality, choice-selected limits, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusWindowCarrier completion classifier transport`

Proof. Start with a $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer request. The $\mathsf{DiagonalModulus}^{\uparrow}$ packet answers the request by Definition 375.3, producing a displayed threshold, stream window, and regular rational readback.

The rational evidence transported to the completion packet is exactly the handoff of Definition 375.4. The downstream real-name boundary is exactly the non-escaping seal of Definition 375.5.

Since the classifier of Definition 375.2 compares only displayed rows and recorded routes, $\mathsf{RealCauchyCompletion}^{\uparrow}$ receives a concrete modulus packet rather than an external completion witness. The consumer therefore obtains the needed diagonal selector while quotient streams, host real equality, choice-selected limits, and ambient completeness remain outside the packet. □

Theorem 375.8 (DiagonalModulus bridge source package)**.** *For every accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *carrier row* $\Delta = (n, \mu, w, r, d, e, P, N)$*, the tuple* $(n, \mu, w, r, d, e, P, N)$ *together with the displayed window selector,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal non-escape row, and* RealCauchyCompletion

classifier transport is the bridge source from the public DiagonalModulus$^{\uparrow}$ *packet to the completion-consumer schema. The source uses only the recorded* BHist, hsame, Cont, Pkg, *and* NameCert *rows and exports no quotient stream, selected limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket bridge source package`

Proof. Project the carrier row of Definition 375.1. Its eight displayed coordinates are the precision request, selected modulus threshold, stream window, regular-sequence readback, dyadic ledger, real seal, package provenance, and local certificate provenance.

Apply Definition 375.3 to obtain the finite window selected by (n, μ), and apply Definition 375.4 to attach the RegSeqRat$^{\uparrow}$ readback and DyadicRatCore$^{\uparrow}$ ledger to that same window.

The real-name side is supplied by Definition 375.5, which keeps the seal behind the displayed precision, threshold, window, readback, and dyadic rows. The completion-facing transport is exactly Definition 375.7.

Repacking these rows gives the bridge source required by the completion-consumer schema. Because every component is one of the displayed carrier or provenance rows, the package has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 375.9 (DiagonalModulus mature boundary)**.** *The window selector totality,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal non-escape, local* NameCert$_{\text{DiagonalModulus}^{\uparrow}}$ *obligation surface,* RealCauchyCompletion$^{\uparrow}$ *classifier transport, and bridge source package jointly exhaust the mature diagonal-modulus boundary. The boundary is the displayed finite route through precision request n, modulus threshold μ,* StreamName$^{\uparrow}$ *window w,* RegSeqRat$^{\uparrow}$ *readback r,* DyadicRatCore$^{\uparrow}$ *ledger d,* Real$^{\uparrow}$ *seal e,* Pkg *provenance, and* NameCert *provenance; it exports no hidden modulus function, quotient stream, host equality, or ambient completeness row.*

Proof. Begin with the finite selector. The window selector totality theorem Definition 375.3 produces w from the displayed precision request and modulus threshold, and Definition 375.4 attaches the RegSeqRat$^{\uparrow}$ readback and DyadicRatCore$^{\uparrow}$ ledger to that same window.

Next close the real-facing and naming surfaces. The seal non-escape theorem Definition 375.5 keeps e behind the displayed rational route, and Definition 375.6 records the corresponding local NameCert$_{\text{DiagonalModulus}^{\uparrow}}$ obligations.

Finally apply Definition 375.7 and Definition 375.8. They move the same finite rows into the completion-consumer and bridge-facing packages, so the mature boundary is exhausted without exporting a hidden modulus function, quotient stream, host equality, or ambient completeness row. □

Closure status for NameCert$_{\text{DiagonalModulus}^{\uparrow}}$**.**

Constructive story: A DiagonalModulus$^{\uparrow}$ packet is generated from a unary precision request, a selected modulus threshold, a finite StreamName$^{\uparrow}$ window, a RegSeqRat$^{\uparrow}$ readback row, a DyadicRatCore$^{\uparrow}$ observation ledger, a Real$^{\uparrow}$ seal row, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DiagonalModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation surface is the carrier row, classifier, total finite-window selector, RegSeqRat$^{\uparrow}$ handoff, DyadicRatCore$^{\uparrow}$ ledger, Real$^{\uparrow}$ seal boundary, dependency stability, and RealCauchyCompletion$^{\uparrow}$ consumer transport.

Formal verification: FormalStatus(NameCert$_{\text{DiagonalModulus}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DiagonalModulusUp.DiagonalModulusPacket\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, quotient equality, selected limits, host real equality, ambient completeness, or bridge checking.

Upgrade path: Formal verification remains the route from the mature paper treatment to checked theorem, axiom-clean, and bridge-checked status over the same displayed precision, modulus, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, Pkg, and NameCert rows.

Chapter 376

A Concrete Naming Certificate for CauchyRegularization$^{\uparrow}$

The CauchyRegularization$^{\uparrow}$ interface names the finite certificate surface that turns a rational Cauchy stream with displayed modulus windows into the regular rational sequence data consumed by RegSeqRat$^{\uparrow}$ and then by Real$^{\uparrow}$. Following Bishop and Bridges, *Constructive Analysis*, the regularizing operation is carried by explicit tail moduli and finite approximants. The packet therefore records only the rational stream windows, dyadic modulus observations, finite-window regularized approximants, seal provenance, and Pkg/NameCert ledgers visible to BEDC consumers.

This packet consumes the modulus threshold surface of chapter 292 and the finite observation-window surface of chapter 126 before repacking the regular-sequence handoff. Row M is read through CauchyModulus$^{\uparrow}$ and row S is read through StreamName$^{\uparrow}$, not through a host sequence or quotient stream.

Definition 376.1 (CauchyRegularization finite carrier)**.** *A* CauchyRegularization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *S is the* RationalStream$^{\uparrow}$ *source row naming the inspected rational Cauchy stream through finite* StreamName$^{\uparrow}$ *observation windows;*
2. *M is the* CauchyModulus$^{\uparrow}$ *row naming, for each requested precision, the displayed tail threshold used by the regularization;*
3. *D is the finite family of* DyadicRatCore$^{\uparrow}$ *modulus-window packets that records the rational tolerance, dyadic radius, and comparison rows read at the inspected thresholds;*
4. *W is the finite window ledger that pairs each inspected source window with the modulus row that justifies reading it as a late Cauchy window;*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row obtained by repacking those late finite-window approximants as a regular rational sequence surface;*
6. *E is the* Real$^{\uparrow}$ *seal-provenance row attached to the same regular sequence handoff;*
7. *H records the componentwise* hsame *transports for source windows, modulus rows, dyadic observations, regularized approximants, and seal rows;*

8. *C records the displayed* Cont *routes by which consumers reach the stream, modulus, dyadic-window, regular-sequence, and seal coordinates;*
9. *P is the* Pkg *provenance over* $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRegularization}^{\uparrow}}$ *ledger.*

The carrier contains no quotient of Cauchy streams, no chosen host real, no ambient completeness theorem, and no hidden tail selector outside the displayed modulus row.

Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.TasteGate.CauchyRegularizationTasteGate single carrier alignment`

Definition 376.2 (CauchyRegularization modulus classifier)**.** *For accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *carriers G and G', the classifier* $\mathsf{CauchyRegularizationClassifier}(G, G')$ *compares only the displayed* $\mathsf{RationalStream}^{\uparrow}$ *windows,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds,* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus-window ledgers, finite-window approximants,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal-provenance rows, componentwise* hsame *transports,* Cont *read routes,* Pkg *provenance, and local* NameCert *ledgers.*

In particular, two carriers classify together only when their regularized finite windows are read from the same transported source windows and the same transported modulus thresholds. The classifier does not inspect host limits, completion quotients, or any convergence predicate not represented by the listed rows.

Lean4 (definition) `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationModulusClassifier`

Theorem 376.3 (CauchyRegularization NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyRegularization}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission: an accepted packet is exactly the finite* BHist *package of Definition 376.1;*
2. *modulus-window classifier stability: Definition 376.2 is preserved by the displayed* hsame *transports and* Cont *routes on source windows, modulus thresholds, dyadic observations, approximants, and seal rows;*
3. *regular-sequence handoff: the finite approximants selected by the modulus windows repack as the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. $\mathsf{Real}^{\uparrow}$ *seal compatibility: the seal-provenance row E reads only the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the finite stream, modulus, dyadic, transport, and package rows carried by the packet;*
5. *ledger exactness: every consumer-visible source window, modulus threshold, dyadic radius, approximant, handoff, seal, transport, continuation, package, and local certificate row occurs in the displayed carrier.*

Proof. Carrier admission is just the finite packet of Definition 376.1. Each coordinate is a visible history row, and the provenance row names exactly the source certificates that own those rows.

Classifier stability is componentwise. Transport a source window, its modulus threshold, and the dyadic observation ledger along the displayed hsame rows, then replay the recorded Cont routes. The regularized approximant is still read from the transported late window justified by the transported modulus threshold.

The regular-sequence handoff is obtained by projecting the row R. It forgets no evidence needed by $\mathsf{RegSeqRat}^{\uparrow}$, because the source window, modulus threshold, dyadic radius, and comparison rows remain in S, M, D, W, H, C.

Finally the $\mathsf{Real}^{\uparrow}$ seal reads the same handoff through E, and the ledger rows P and N account for package and local naming provenance. Since every consumer-visible coordinate is listed in the carrier, there is no extra host limit, quotient, hidden selector, or ambient completeness object to export. □

Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationFiniteCarrier namecert obligation surface`
Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationModulusClassifier stability`

Theorem 376.4 (CauchyRegularization regular-sequence handoff)**.** *For every accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *carrier and every displayed precision request, the* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read factors through one* $\mathsf{RationalStream}^{\uparrow}$ *finite observation window, one* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row for that precision, the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus-window ledger, and the finite-window approximant recorded in* W*. No* $\mathsf{RegSeqRat}^{\uparrow}$ *read can bypass those rows by using an unrecorded tail choice or a quotient of rational streams.*

Proof. Project the carrier $G = (S, M, D, W, R, E, H, C, P, N)$. A precision request first reaches the source stream through the displayed $\mathsf{StreamName}^{\uparrow}$ window in S and then reads the threshold supplied by M.

The dyadic tolerance and comparison evidence are exactly the rows recorded in D. The window ledger W pairs those rows with the inspected late source window, so the regularized approximant has a visible origin in the same finite packet.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is attached to that approximant ledger. Transport changes only the displayed routes in H and C; it does not add a hidden tail selector or a quotient stream coordinate. Hence every regular-sequence read factors through the stated finite rows. □

Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationFiniteCarrier regular sequence handoff`

Theorem 376.5 (CauchyRegularization RealUp seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *packet is mediated by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, the displayed rational-stream window, the modulus threshold, the dyadic modulus-window ledger, the regularized finite approximant, the* hsame *and* Cont *transport rows, and the* Pkg/NameCert *provenance. The packet exports no classical completeness claim, no host real equality, no selected limit, and no quotient of Cauchy streams.*

Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationSealRoute semantic name certificate`

Proof. Begin with a real-facing read. The only $\mathsf{Real}^{\uparrow}$ coordinate in the carrier is the seal-provenance row E, and E is attached to the regular rational handoff R.

Use Definition 376.4 to expand the source of R. The read must pass through the finite rational observation window, the displayed modulus threshold, the dyadic tolerance ledger, and the approximant row justified by that window.

The remaining coordinates are transport and provenance rows. They preserve the displayed route and name the dependency certificates, but they contain no field for a host real value, a quotient representative, a selected limit, or an ambient completeness theorem. Thus the seal boundary is exactly the finite BEDC surface stated above. □

Theorem 376.6 (CauchyRegularization common-modulus idempotence)**.** *For accepted* CauchyRegularization$^{\uparrow}$ *carriers regularized through the same transported* RationalStream$^{\uparrow}$ *observation windows,* CauchyModulus$^{\uparrow}$ *threshold rows,* DyadicRatCore$^{\uparrow}$ *modulus-window ledgers,* hsame *and* Cont *routes,* Pkg *rows, and* NameCert *ledger, a second* CauchyRegularization$^{\uparrow}$ *read through those same displayed rows yields the same* RegSeqRat$^{\uparrow}$ *handoff classifier and the same* Real$^{\uparrow}$ *seal-provenance row. The repeated read introduces no hidden tail selector, quotient stream, host limit, or completeness claim.*

Lean4 ✓ `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationFiniteCarrier common modulus idempotence`

Proof. Project the finite carrier $G = (S, M, D, W, R, E, H, C, P, N)$. The common modulus classifier identifies the displayed source window in S, the threshold row in M, the dyadic ledger in D, and the finite approximant ledger in W for the first read and for the second read.

Apply Definition 376.4 to both reads. Since the two reads use the same transported observation windows, modulus thresholds, dyadic rows, transport routes, package rows, and naming ledger, the handoff route can only be the displayed RegSeqRat$^{\uparrow}$ classifier row R.

It remains to compare the real-facing rows. Apply Definition 376.5. The Real$^{\uparrow}$ seal-provenance row is mediated by that same handoff, the same transport routes, and the same Pkg/NameCert provenance, so the repeated read has the identical displayed BEDC seal route and no additional selector or host-limit coordinate. □

Theorem 376.7 (CauchyRegularization RealUp seal idempotent route)**.** *For accepted* CauchyRegularization$^{\uparrow}$ *packets whose first and second regularization reads share the transported* RationalStream$^{\uparrow}$ *windows,* CauchyModulus$^{\uparrow}$ *threshold rows,* DyadicRatCore$^{\uparrow}$ *ledger rows,* RegSeqRat$^{\uparrow}$ *handoff row,* hsame *and* Cont *routes,* Pkg *provenance, and* NameCert *ledger, the* Real$^{\uparrow}$ *consumer seal route after the second read is the same displayed route as after the first read. The repeated real-facing read introduces no quotient stream, selected limit, host real equality, hidden tail selector, or ambient completeness coordinate.*

Proof. Project the finite carrier $G = (S, M, D, W, R, E, H, C, P, N)$. The shared carrier rows fix the source windows, modulus thresholds, dyadic ledgers, regularized approximants, handoff row, transport routes, package provenance, and local naming ledger for both reads.

Apply Definition 376.6. The RegSeqRat$^{\uparrow}$ handoff classifier and the Real$^{\uparrow}$ seal-provenance row are locked to the displayed rows R and E for the first read and for the second read.

Apply Definition 376.5 to expand both real-facing reads. Each read passes through the same handoff, transported rational-stream window, continuation route, package row, and local naming row, and the boundary lists no further consumer-visible coordinate.

Therefore the second Real$^{\uparrow}$ seal route is the same displayed route as the first. Since the carrier and the boundary exclusions contain no quotient stream, selected limit, host real equality, hidden tail selector, or ambient completeness coordinate, repeated regularization through the same finite rows adds no Real-facing escape route. □

Theorem 376.8 (CauchyRegularization Real seal source-modulus factorization)**.** *For an accepted* CauchyRegularization$^{\uparrow}$ *packet* $G = (S, M, D, W, R, E, H, C, P, N)$*, every* Real$^{\uparrow}$ *seal read factors through exactly the displayed finite* StreamName$^{\uparrow}$ *source window carried by* S*, the* CauchyModulus$^{\uparrow}$ *threshold row* M*, the* DyadicRatCore$^{\uparrow}$ *modulus-window ledger* D*, the approximant ledger* W*, the* RegSeqRat$^{\uparrow}$ *handoff row* R*, the seal provenance* E*, and the structural rows* H, C, P, N*. Conversely, those rows repack as the same Real-facing seal route. No quotient stream, selected limit, host real equality, hidden tail selector, ambient completeness theorem, or unrecorded modulus choice contributes to the read.*

Proof. Project the finite carrier of Definition 376.1. Its displayed rows are precisely $S, M, D, W, R, E, H, C, P, N$, so any Real-facing route must first account for the source window, modulus threshold, dyadic ledger, approximant ledger, handoff, seal provenance, transports, continuation routes, package row, and local naming ledger named there.

For the forward direction, apply Definition 376.4. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R is reached only through the displayed $\mathsf{StreamName}^{\uparrow}$ source window in S, the threshold row M, the dyadic modulus-window ledger D, and the approximant ledger W.

Then apply Definition 376.5. The $\mathsf{Real}^{\uparrow}$ seal reads that handoff through the seal-provenance row E and the structural rows H, C, P, N, and the boundary exposes no further consumer-visible coordinate.

For the converse direction, start from the displayed rows $S, M, D, W, R, E, H, C, P, N$. The handoff theorem repacks the source-modulus data as the same $\mathsf{RegSeqRat}^{\uparrow}$ route, and the seal-boundary theorem attaches exactly the Real-facing provenance row E with the same structural routes and ledgers.

Finally use Definition 376.7. If another Real-facing route were available from the same source, modulus, dyadic, approximant, handoff, seal, transport, package, and naming rows, the second read would differ from the first. Idempotence rules this out, so the factorization is exactly the displayed route and excludes quotient streams, selected limits, host real equality, hidden tail selectors, ambient completeness, and unrecorded modulus choices. □

Theorem 376.9 (CauchyRegularization Real seal normalization retract)**.** *Let $G = (S, M, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *packet. Suppose a second regularization packet reads the same transported* $\mathsf{RationalStream}^{\uparrow}$ *window row S, the same* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row M, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the same approximant ledger W, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, the same seal row E, and the same structural rows H, C, P, N. Then the* $\mathsf{Real}^{\uparrow}$ *consumer route of the second read retracts to the original seal route E.*

The retract exports no quotient stream equality, selected limit, host real equality, hidden tail selector, unrecorded modulus choice, or ambient completeness theorem.

Proof. Project the finite carrier of Definition 376.1 for the original packet and for the second packet. The two projections list the same source window, modulus threshold, dyadic ledger, approximant ledger, handoff, seal, transport, continuation, package, and naming rows.

Apply Definition 376.6. The repeated read cannot introduce a new $\mathsf{CauchyModulus}^{\uparrow}$ threshold row or a new $\mathsf{RegSeqRat}^{\uparrow}$ handoff; the handoff and seal provenance remain the displayed rows R and E.

Apply Definition 376.7. With the same Real-facing rows on both reads, the second consumer route is identified with the first consumer route at the seal surface.

Finally apply Definition 376.8. The first route has only the displayed source-modulus-seal factorization through $S, M, D, W, R, E, H, C, P, N$. Hence the second route retracts to the same seal row E, and the listed quotient, limit, equality, selector, modulus-choice, and ambient-completeness data have no exported coordinate. □

Theorem 376.10 (CauchyRegularization public real-regularization export)**.** *The finite carrier, modulus classifier, regular-sequence handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and Real-facing idempotent route assemble a public* $\mathsf{CauchyRegularization}^{\uparrow}$ *export over* $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The export exposes only displayed rational-stream windows, modulus thresholds, dyadic comparison rows, regularized approximants, the regular-sequence handoff, the real-seal provenance, and the named transport and package rows.*

It exposes no quotient stream, host real equality, selected limit, hidden tail selector, or ambient completeness theorem.

Proof. Start with the carrier and classifier. The carrier of Definition 376.1 lists exactly the finite rows available to a consumer, and the classifier of Definition 376.2 transports only those rows.

Apply Definition 376.3 to bind carrier admission, classifier stability, regular-sequence handoff, real-seal compatibility, and ledger exactness into the local naming surface.

Next use Definition 376.4. It routes every $\mathsf{RegSeqRat}^{\uparrow}$ read through a displayed rational window, a displayed modulus threshold, the dyadic ledger, and the finite approximant row.

Use Definition 376.5. The $\mathsf{Real}^{\uparrow}$ read is mediated by the same handoff, transport, continuation, package, and local naming provenance.

Finally use Definition 376.7. Repeated regularization through the same finite rows preserves that Real-facing route. The finite-row assembly therefore gives a public real-regularization export without quotient streams, host equality, selected limits, hidden tail selectors, or ambient completeness. □

Theorem 376.11 (CauchyRegularization public normalization retract export)**.** *Let $G = (S, M, D, W, R, E, H, C, P$ be an accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *packet, and suppose a second accepted* $\mathsf{CauchyRegularization}^{\uparrow}$ *packet has the same displayed source row S, modulus row M, dyadic ledger D, approximant ledger W, regular-sequence handoff R, seal row E, and structural rows H, C, P, N. Then the public real-regularization export of the second read retracts to the original public export surface determined by $S, M, D, W, R, E, H, C, P, N$, not only to the terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

This public retract exports no quotient stream, selected limit, host real equality, hidden tail selector, unrecorded modulus choice, or ambient completeness theorem.

Proof. First apply Definition 376.10 to the original packet. Its public export surface contains exactly the displayed rational-stream windows, modulus thresholds, dyadic comparison rows, regularized approximants, handoff row, real-seal provenance, transport rows, package row, and naming ledger.

Apply the same public export theorem to the second packet. Since the second accepted packet shares $S, M, D, W, R, E, H, C, P, N$, its public export has the same displayed carrier, classifier, handoff, seal, transport, package, and naming coordinates.

Next apply Definition 376.9. The Real-facing consumer route of the second read retracts to the original seal route E under the shared displayed rows.

Finally apply Definition 376.8. The seal row E is not an isolated terminal coordinate: it is reached only through the source window S, modulus row M, dyadic ledger D, approximant ledger W, handoff row R, and the structural rows H, C, P, N. Thus the retract returns the second public read to the original public export surface itself. The excluded quotient stream, selected-limit, host-equality, tail-selector, modulus-choice, and ambient-completeness data have no public coordinate available for the retract. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRegularization}^{\uparrow}$ packet is generated from finite $\mathsf{RationalStream}^{\uparrow}$ windows, explicit $\mathsf{CauchyModulus}^{\uparrow}$ threshold rows, $\mathsf{DyadicRatCore}^{\uparrow}$ modulus-window observations, regularized finite-window approximants, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal-provenance row, transport routes, package provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRegularization}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The obligation scope is the finite carrier, modulus-window classifier, regular-sequence handoff, $\mathsf{Real}^{\uparrow}$ seal compatibility, and ledger exactness over $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRegularization}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyRegularizationUp.CauchyRegularizationSealRoute\_semantic\_name\_certificate`
Bridge status: none
Origin: human
Not claimed: This chapter does not close classical real completeness, quotient equality of Cauchy streams, uniqueness of limits, selected host real values, or bridge checking.
Upgrade path: Bridged closure requires a concrete bridge theorem from the public regularization export to the abstract completion-facing interface.

Chapter 377

A Concrete Naming Certificate for $\mathsf{UnitDisk}^{\uparrow}$

The $\mathsf{UnitDisk}^{\uparrow}$ interface names the closed real unit-disk carrier as a finite BEDC packet assembled from real coordinate names, the product pairing surface, metric radius reads, interval or real-order bounds, and the boundary consumer already available to $\mathsf{S}^{1\uparrow}$. In the standard topology text of Munkres, Topology, the closed unit disk and its boundary circle are read as the familiar Euclidean disk and unit circle; this chapter records only the finite BHist certificate surface used by BEDC consumers.

The disk carrier consumes the $\mathsf{Met}^{\uparrow}$ radius witness surface as the public route from the displayed product point to the displayed origin. When that radius is exactly the real unit row, the same packet hands its boundary row to chapter 58 as the boundary consumer route, not as a separate host circle or quotient boundary object.

Definition 377.1 (UnitDisk BHist carrier)**.** *A* $\mathsf{UnitDisk}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (x, y, p, o, d, b, \partial, H, C, P, N)$$

with the following visible rows:

1. *x and y are two* $\mathsf{Real}^{\uparrow}$ *coordinate rows, each read through its own finite real-name seal;*
2. *p is the* $\mathsf{Prod}^{\uparrow}$ *pair row joining the two coordinate rows as one displayed planar point;*
3. *o is the origin row, formed from the two zero* $\mathsf{Real}^{\uparrow}$ *rows and paired through the same* $\mathsf{Prod}^{\uparrow}$ *surface;*
4. *d is the* $\mathsf{Met}^{\uparrow}$ *radius-distance witness reading the distance from p to o through the displayed metric route;*
5. *b is a nonnegative bound row, supplied either by* $\mathsf{Interval}^{\uparrow}$ *or by* $\mathsf{RealAlgOrder}^{\uparrow}$*, witnessing that the displayed radius is at most the real unit row;*
6. *∂ is an optional boundary row recording the exact unit-radius packet consumed by* $\mathsf{S}^{1\uparrow}$ *when the metric read is classified as boundary;*
7. *H and C are the componentwise* hsame *witnesses and displayed* Cont *routes for coordinate, pair, origin, radius, bound, and optional boundary transport;*
8. *P is the* Pkg *provenance naming the consumed* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Interval}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*, and* $\mathsf{S}^{1\uparrow}$ *source surfaces;*

9. N is the local $\mathsf{NameCert}_{\mathsf{UnitDisk}^\uparrow}$ *provenance row.*

The carrier classifier compares two accepted packets only by transporting these displayed rows componentwise through H and C. It has no coordinate for host Euclidean topology, open-cover compactness, quotient boundary identification, or a host subset of $\mathbb{R}^2$.

Lean4 ✓ `BEDC.Derived.UnitDiskUp.UnitDiskTasteGate single carrier alignment`

Theorem 377.2 (UnitDisk carrier stability)**.** *Let $U = (x, y, p, o, d, b, \partial, H, C, P, N)$ be an accepted* $\mathsf{UnitDisk}^\uparrow$ *carrier, and let $U' = (x', y', p', o', d', b', \partial', H', C', P', N')$ be obtained by componentwise displayed* hsame *transport of the two* $\mathsf{Real}^\uparrow$ *coordinate rows, the* $\mathsf{Prod}^\uparrow$ *point row, the origin row, the* $\mathsf{Met}^\uparrow$ *radius witness, the* $\mathsf{Interval}^\uparrow$ *or* $\mathsf{RealAlgOrder}^\uparrow$ *bound row, the optional* $\mathsf{S}^{1\uparrow}$ *boundary row, and the provenance rows. If the displayed* Cont *routes preserve the coordinate, pairing, metric-read, bound, and boundary ledgers, then U' is again an accepted* $\mathsf{UnitDisk}^\uparrow$ *carrier and the local classifier relates U to U'.*

Lean4 ✓ `BEDC.Derived.UnitDiskUp.UnitDiskOriginRows radius ledger`

Proof. Project the carrier through Definition 377.1. The two real coordinates, product pair, origin pair, radius witness, bound row, optional boundary row, routing rows, and provenance rows are all explicit finite packet coordinates.

Transport each coordinate by the displayed hsame witnesses, then repack the transported coordinates through the same $\mathsf{Prod}^\uparrow$ pairing route. The origin row is transported in parallel, so the metric witness still reads a radius from the transported point to the transported origin.

Use the stated Cont routes to move the radius comparison into the transported $\mathsf{Interval}^\uparrow$ or $\mathsf{RealAlgOrder}^\uparrow$ bound row. If a boundary row is present, the same transport carries it to the optional $\mathsf{S}^{1\uparrow}$ consumer row; if it is absent, no boundary consumer is exported.

Repacking these rows gives the accepted packet U'. Since the classifier inspects exactly the displayed rows, the transported packet is classified with U and no host topological coordinate is created. □

Lean4 ✓ `BEDC.Derived.UnitDiskUp.UnitDiskBHistCarrier carrier stability`

Theorem 377.3 (UnitDisk boundary SOne consumer)**.** *For every accepted* $\mathsf{UnitDisk}^\uparrow$ *carrier whose optional boundary row ∂ is present, a downstream* $\mathsf{S}^{1\uparrow}$ *consumer may read only the two* $\mathsf{Real}^\uparrow$ *coordinate rows, their* $\mathsf{Prod}^\uparrow$ *pair row, the* $\mathsf{Met}^\uparrow$ *unit-radius witness from the origin, the exact boundary ledger carried by ∂, and the associated* hsame, Cont, Pkg, *and* NameCert *provenance.*

The consumer boundary does not export open-cover compactness, quotient boundary identifications, host Euclidean topology, or an ambient subset construction for the disk.

Proof. Read the packet through Definition 377.1. The boundary coordinate is optional and, when present, is the only row routed to $\mathsf{S}^{1\uparrow}$.

The $\mathsf{S}^{1\uparrow}$ consumer receives the coordinate pair through $\mathsf{Prod}^\uparrow$ and the radius equality through the same $\mathsf{Met}^\uparrow$ witness used by the disk packet. The bound row confirms that the radius sits at the displayed real unit; the boundary row records exact unit-radius consumption rather than a separate host circle.

Apply Definition 377.2 to see that these rows are stable under the displayed componentwise classifier. The exported boundary surface is therefore the finite row ∂ with its provenance, not a quotient of boundary points or a topological subspace. □

Lean4 ✓ `BEDC.Derived.UnitDiskUp.UnitDiskBHistCarrier boundary sone consumer`

Theorem 377.4 (UnitDisk NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnitDisk}^\uparrow}$ *obligation surface consists of five finite rows:*

1. *carrier inhabitation: the origin row together with the two zero* $\mathsf{Real}^\uparrow$ *coordinate rows, their* $\mathsf{Prod}^\uparrow$ *pair row, the zero* $\mathsf{Met}^\uparrow$ *radius read, and the unit bound row inhabit the carrier of Definition 377.1;*
2. *classifier laws: reflexivity, symmetry, and transitivity are inherited componentwise from the displayed* hsame *comparisons on coordinates, product pairs, origin rows, metric witnesses, bounds, optional boundary rows, routing rows, and provenance;*
3. *carrier transport: Definition 377.2 transports accepted packets along the displayed* hsame *and* Cont *rows without adding a host point set;*
4. *exact ledger reads: every public radius read factors through the* $\mathsf{Met}^\uparrow$ *row* d *and every boundary read factors through the optional row* ∂ *of the same carrier;*
5. *consumer boundary: Definition 377.3 permits* $\mathsf{S}^{1\uparrow}$ *consumption of exact boundary packets while excluding open-cover compactness, quotient boundary identifications, and host Euclidean topology.*

Lean4 ✓ `BEDC.Derived.UnitDiskUp.UnitDiskBHistCarrier namecert obligation surface`

Proof. The carrier row is Definition 377.1. Its origin coordinate gives an accepted packet by taking both real coordinates to be the displayed zero real row, pairing them through $\mathsf{Prod}^\uparrow$, reading zero radius through $\mathsf{Met}^\uparrow$, and comparing zero to the real unit through the displayed nonnegative bound surface.

The classifier row is componentwise. Reflexivity uses each displayed row as itself, symmetry reverses the componentwise hsame witnesses, and transitivity composes them across the same coordinate, pair, origin, radius, bound, boundary, routing, and provenance positions.

Carrier transport is exactly Definition 377.2. It keeps the metric and bound rows tied to the transported coordinate pair and origin row, so an accepted disk packet remains inside the same finite carrier surface.

For ledger exactness, inspect the packet coordinates. The only radius coordinate is d, and the only boundary coordinate is ∂ when that row is present. A consumer can therefore read radius and boundary data only through those displayed rows and their provenance.

Finally apply Definition 377.3. The optional boundary handoff is a finite $\mathsf{S}^{1\uparrow}$ consumer surface and contains no open-cover compactness datum, quotient boundary construction, or host Euclidean topology. □

Theorem 377.5 (UnitDisk origin radius ledger)**.** *The packet whose coordinates* x *and* y *are the zero* $\mathsf{Real}^\uparrow$ *rows, whose point row* p *and origin row* o *are both the displayed* $\mathsf{Prod}^\uparrow$ *pairing of those zero rows, whose radius witness* d *is the zero* $\mathsf{Met}^\uparrow$ *read from* p *to* o*, whose bound row* b *records zero below the real unit through* $\mathsf{Interval}^\uparrow$ *or* $\mathsf{RealAlgOrder}^\uparrow$*, and whose provenance rows are the corresponding* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows is an accepted* $\mathsf{UnitDisk}^\uparrow$ *carrier. Its radius ledger is exactly the displayed zero-radius* $\mathsf{Met}^\uparrow$ *row and its unit-bound row.*

Proof. Use Definition 377.1 with both coordinates set to the displayed zero $\mathsf{Real}^\uparrow$ row. The product row p and the origin row o are then the same finite $\mathsf{Prod}^\uparrow$ pairing surface.

The metric read from the point to the origin is the displayed zero-distance row. The bound row records that this zero radius is below the real unit through the available $\mathsf{Interval}^\uparrow$ or $\mathsf{RealAlgOrder}^\uparrow$ comparison.

Package these coordinates with the identity hsame comparisons, the displayed Cont routes, P, and N. The carrier contains no other radius coordinate, so the radius ledger of this origin packet is exactly d together with the unit-bound row b. □

Theorem 377.6 (UnitDisk classifier law package)**.** *The* $\mathsf{UnitDisk}^{\uparrow}$ *classifier has reflexivity, symmetry, transitivity, and carrier transport componentwise on the displayed* $\mathsf{Real}^{\uparrow}$ *coordinate rows,* $\mathsf{Prod}^{\uparrow}$ *pair rows, origin rows,* $\mathsf{Met}^{\uparrow}$ *radius witnesses,* $\mathsf{Interval}^{\uparrow}$ *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *bound rows, optional* $\mathsf{S}^{1\uparrow}$ *boundary rows,* BHist *routing rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and* NameCert *provenance.*

Proof. Read the classifier from Definition 377.1. It compares only visible coordinates of the packet and the displayed transport rows; it does not compare ambient Euclidean points or host subsets.

Reflexivity uses the identity displayed comparison at each coordinate. Symmetry reverses each componentwise hsame comparison and the associated Cont route on the same coordinate, pair, radius, bound, boundary, and provenance rows.

Transitivity composes two displayed transports coordinate by coordinate. Since each transport keeps the metric witness tied to the transported point and origin and keeps the bound and optional boundary ledgers attached to the same radius row, the composite packet is again accepted.

Carrier transport is exactly Definition 377.2. Combining these componentwise laws gives the stated classifier law package. □

Theorem 377.7 (UnitDisk radius boundary exactness)**.** *Every* $\mathsf{UnitDisk}^{\uparrow}$ *radius consumer read factors through the displayed* $\mathsf{Met}^{\uparrow}$ *radius witness* d *of the carrier packet, and every boundary consumer read factors through the optional* $\mathsf{S}^{1\uparrow}$ *boundary row* ∂ *when that row is present. No consumer read exports a host Euclidean subset, an open-cover compactness datum, a quotient boundary identification, or any boundary object not displayed in the carrier.*

Proof. Project an accepted packet through Definition 377.1. The only radius coordinate is the $\mathsf{Met}^{\uparrow}$ witness d, and the only exact boundary coordinate is the optional row ∂.

For a radius request, the consumer first reads the two $\mathsf{Real}^{\uparrow}$ coordinates through the $\mathsf{Prod}^{\uparrow}$ row and the origin row, then reads the radius through d and the unit comparison through b. There is no separate Euclidean norm coordinate in the packet.

For a boundary request, apply Definition 377.3. If ∂ is present, the $\mathsf{S}^{1\uparrow}$ consumer receives exactly the boundary row and its provenance; if it is absent, the disk packet exports no boundary consumer surface.

The non-escape clauses follow by inspection of the carrier coordinates and by Definition 377.2, which preserves the same displayed metric and boundary rows under transport. □

Theorem 377.8 (UnitDisk product metric boundary package)**.** *For every accepted* $\mathsf{UnitDisk}^{\uparrow}$ *packet, the two* $\mathsf{Real}^{\uparrow}$ *coordinate rows, the* $\mathsf{Prod}^{\uparrow}$ *pair row, the origin row, the* $\mathsf{Met}^{\uparrow}$ *radius witness, the* $\mathsf{Interval}^{\uparrow}$ *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *bound row, the optional* $\mathsf{S}^{1\uparrow}$ *boundary row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance form the exact product-metric boundary package for* $\mathsf{UnitDisk}^{\uparrow}$*.*

The package exposes the product pairing, the metric radius comparison with the real unit row, and the optional circle-boundary consumer row as a single finite certificate. It exports no host Euclidean subset, open-cover compactness, quotient boundary identification, or unrecorded topology.

Proof. Start with Definition 377.1. It lists the two real coordinates, their product row, the origin row, the metric radius witness, the unit-bound row, the optional boundary row, and the transport and provenance rows as coordinates of one finite packet.

Apply Definition 377.2 to keep the product point, origin, radius, bound, boundary, and provenance rows stable under the displayed hsame and Cont transports. The classifier therefore remains componentwise on the same product-metric rows.

Use Definition 377.3 for the optional $\mathsf{S}^{1\uparrow}$ handoff and Definition 377.4 for the local carrier, classifier, transport, exact-ledger, and consumer obligations.

Finally combine Definition 377.6 with Definition 377.7. The resulting package is precisely the finite product-metric disk certificate with its boundary consumer row when present, and no host subset, compactness, quotient boundary, or unrecorded topological coordinate is introduced. □

Theorem 377.9 (UnitDisk carrier habitation)**.** *The concrete packet*

$$U = (x, y, p, o, d, b, \partial, H, C, P, N)$$

inhabits $\mathsf{UnitDisk}^{\uparrow}$ *whenever x and y are the displayed zero* $\mathsf{Real}^{\uparrow}$ *coordinate rows, p and o are the corresponding* $\mathsf{Prod}^{\uparrow}$ *pair rows, d is the zero* $\mathsf{Met}^{\uparrow}$ *radius ledger, b is the displayed unit-bound row, ∂ is absent or the exact* $\mathsf{S}^{1\uparrow}$ *unit-radius row, and H, C, P, N are the matching* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows.*

Proof. Apply Definition 377.5. It supplies the zero coordinate rows, the product point, the origin row, the zero metric radius read, and the unit-bound ledger.

Then apply Definition 377.4. Its carrier-inhabitation row repacks those displayed coordinates with the BHist, hsame, Cont, Pkg, and NameCert provenance.

If the optional boundary row is present, use Definition 377.3; if it is absent, the carrier remains accepted without exporting a boundary consumer. Both cases stay inside the concrete packet. □

Theorem 377.10 (UnitDisk classifier laws)**.** *For accepted packets*

$$U = (x, y, p, o, d, b, \partial, H, C, P, N)$$

and U' with the same concrete row shape, the $\mathsf{UnitDisk}^{\uparrow}$ *classifier is reflexive, symmetric, transitive, and stable under componentwise transport of the* $\mathsf{Real}^{\uparrow}$ *coordinate rows,* $\mathsf{Prod}^{\uparrow}$ *pair row, origin row,* $\mathsf{Met}^{\uparrow}$ *radius ledger, unit-bound row, optional* $\mathsf{S}^{1\uparrow}$ *boundary row, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows.*

Proof. Use Definition 377.6. It states the reflexivity, symmetry, transitivity, and transport laws componentwise for the displayed coordinate, pair, origin, radius, bound, boundary, routing, and provenance rows.

Apply Definition 377.2 to keep the transported packet accepted. The same theorem keeps the metric radius row attached to the transported point and origin and keeps the optional boundary row tied to the same displayed radius ledger.

The classifier therefore compares only rows of U and U'. It does not create a host Euclidean point, quotient boundary identification, or external topological equality. □

Theorem 377.11 (UnitDisk radius ledger exactness)**.** *Every public radius or boundary read of an accepted*

$$U = (x, y, p, o, d, b, \partial, H, C, P, N)$$

factors through the displayed $\mathsf{Met}^{\uparrow}$ *row d, the unit-bound row b, the optional* $\mathsf{S}^{1\uparrow}$ *boundary row ∂ when present, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows H, C, P, N.*

Proof. Apply Definition 377.7. It identifies d as the only radius coordinate and ∂ as the only boundary coordinate when the boundary row is present.

Use Definition 377.8 to route the two $\mathsf{Real}^{\uparrow}$ coordinate rows through p and o, then through the $\mathsf{Met}^{\uparrow}$ radius witness and the displayed unit-bound row.

Finally apply Definition 377.3 for the boundary case. The provenance rows H, C, P, N record the allowed hsame, Cont, Pkg, and NameCert reads, so no host Euclidean subset, compactness datum, or quotient boundary object can enter the ledger. □

Theorem 377.12 (UnitDisk scoped product-metric source)**.** *Every accepted* $\mathsf{UnitDisk}^{\uparrow}$ *packet reads its two* $\mathsf{Real}^{\uparrow}$ *coordinate rows through the displayed* $\mathsf{Prod}^{\uparrow}$ *pair row and the displayed origin row of Definition 377.1. The product-metric source of the disk certificate is therefore the finite row package*

$$(x, y, p, o, H, C, P, N)$$

rather than a host point of $\mathbb{R}^2$ *or an ambient Euclidean subset.*

Proof. Project the packet through Definition 377.1. The coordinate rows x and y, the product pair row p, and the origin row o are explicit carrier coordinates.

Apply Definition 377.2 to see that transport preserves the same coordinate, pair, and origin positions. Apply Definition 377.6 to read the classifier componentwise on those displayed rows.

The source row is consequently the finite product-metric package displayed by the carrier. No host point set or ambient Euclidean subset is added to the certificate. □

Theorem 377.13 (UnitDisk scoped radius-boundary route)**.** *For every accepted* $\mathsf{UnitDisk}^{\uparrow}$ *packet, each radius read factors through the displayed* $\mathsf{Met}^{\uparrow}$ *radius row* d*, the* $\mathsf{Interval}^{\uparrow}$ *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *unit-bound row* b*, and the transport and provenance rows of Definition 377.1. When the optional boundary row* ∂ *is present, every boundary read additionally factors through the displayed* $\mathsf{S}^{1\uparrow}$ *consumer row.*

Proof. Start with the carrier projection. The radius row is d, the unit-bound row is b, and the optional boundary row is ∂.

Use Definition 377.7. It states that public radius consumers read through d and that boundary consumers read through ∂ when that row is present.

Apply Definition 377.3 for the boundary handoff and Definition 377.8 for the combined product, metric, bound, and optional boundary package. These routes keep every radius or boundary request inside the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{Interval}^{\uparrow}$ or $\mathsf{RealAlgOrder}^{\uparrow}$, and $\mathsf{S}^{1\uparrow}$ rows. □

Theorem 377.14 (UnitDisk scoped consumer non-escape)**.** *Downstream consumers of an accepted* $\mathsf{UnitDisk}^{\uparrow}$ *packet cannot read host Euclidean subsets, open-cover compactness, quotient boundary identification, or unrecorded topology. Their public reads are restricted to the displayed* $\mathsf{Real}^{\uparrow}$ *coordinate rows,* $\mathsf{Prod}^{\uparrow}$ *pair and origin rows,* $\mathsf{Met}^{\uparrow}$ *radius row,* $\mathsf{Interval}^{\uparrow}$ *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *bound row, optional* $\mathsf{S}^{1\uparrow}$ *boundary row, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows.*

Proof. Inspect Definition 377.1. The carrier lists exactly the coordinate, product, origin, radius, bound, optional boundary, transport, route, package, and local naming rows.

Apply Definition 377.3 to boundary consumers and Definition 377.7 to radius consumers. Both theorems identify the displayed rows through which public reads must factor.

Finally use Definition 377.2 and Definition 377.6. Transport and classification remain componentwise on the same finite packet rows, so no host subset, compactness proof, quotient boundary object, or unrecorded topology can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnitDisk}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UnitDisk}^{\uparrow}$ packet is generated by two $\mathsf{Real}^{\uparrow}$ coordinate rows, a $\mathsf{Prod}^{\uparrow}$ pair row, an origin row, a $\mathsf{Met}^{\uparrow}$ radius witness, an $\mathsf{Interval}^{\uparrow}$ or $\mathsf{RealAlgOrder}^{\uparrow}$ unit-bound row, optional $\mathsf{S}^{1\,\uparrow}$ boundary consumption, transport routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnitDisk}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 377.9, Definition 377.10, Definition 377.11, Definition 377.12, Definition 377.13, and Definition 377.14 export the finite closed unit-disk certificate as the public product-metric and boundary-consumer surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnitDisk}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.UnitDiskUp.UnitDiskBHistCarrier\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close compactness, homotopy, quotient boundary identification, host Euclidean topology, or bridge checking.

Upgrade path: Bridge closure requires a standard bridge treatment for the exported product-metric and boundary-consumer surface.

Chapter 378

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonad}^{\uparrow}$

The $\mathsf{CauchyCompletionMonad}^{\uparrow}$ interface names the finite completion transformer that reads regular rational Cauchy data and returns a $\mathsf{Real}^{\uparrow}$ seal route. It consumes the visible real-and-completion primitives $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ without forming quotient reals, selecting limits by choice, or appealing to host completeness. The finite completion transformer consumes the $\mathsf{Real}^{\uparrow}$ seal surface of chapter 38 through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows already displayed in the carrier.

The finite completion transformer reads the window discipline of chapter 126, the dyadic observation ledger of chapter 291, the regular-rational readback of chapter 301, and the seal boundary of chapter 38.

Definition 378.1 (CauchyCompletionMonad carrier)**.** *A* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, W, D, \mu, \Delta, E, L, C, P, N)$$

with the following visible rows:

1. *X is the source regular-Cauchy family over* $\mathsf{RegSeqRat}^{\uparrow}$*;*
2. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected from the members of X;*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations read through the windows in W;*
4. *μ is a unary modulus and schedule row naming which family member and which window answer each requested precision;*
5. *Δ is the diagonal transport ledger that repacks the scheduled observations as a regular rational readback;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row attached to that diagonal readback;*
7. *L records the componentwise* hsame *comparisons for source-family rows, windows, dyadic observations, schedule rows, diagonal rows, and seal rows;*
8. *C records the displayed* Cont *routes through which consumers reach the schedule, window, observation, diagonal, and real-seal coordinates;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the local dependency certificates;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates and their displayed transports.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadTasteGate single carrier alignment`

Theorem 378.2 (CauchyCompletionMonad NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier habitation by constant regular rational data, using constant* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, stationary* $\mathsf{StreamName}^{\uparrow}$ *windows, constant* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the zero schedule, the diagonal readback of that stationary packet, and the corresponding* $\mathsf{Real}^{\uparrow}$ *seal row;*
2. *classifier transport for schedule rows, window rows, dyadic observation rows, diagonal rows, and seal rows by componentwise* hsame *and the displayed* Cont *routes;*
3. *diagonal handoff exactness: each real-facing precision read factors through the schedule row, the selected finite window, the displayed dyadic observation, and the diagonal readback row before reaching the* $\mathsf{Real}^{\uparrow}$ *seal;*
4. *non-escape of quotient reals, host real equality, choice-selected limits, ambient completeness principles, or hidden stream representatives;*
5. *public consumer boundary for* $\mathsf{Real}^{\uparrow}$ *completion reads, exposing only* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and local* NameCert *coordinates.*

These obligations specify the finite completion-interface surface and do not assert that $\mathsf{Real}^{\uparrow}$ *is complete as an ambient real object.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket namecert obligations`

Proof. For habitation, take a constant regular rational family. Its displayed stream windows are stationary, every dyadic observation is the same rational row, the unary schedule may select the initial member for each precision request, and the diagonal ledger repacks the same rows as the real-facing readback.

Classifier transport is coordinatewise. Transport the schedule, selected windows, dyadic observations, diagonal ledger, and real-seal row along the stored hsame comparisons, then compose with the displayed Cont read routes for the same coordinates.

For diagonal handoff exactness, project the carrier coordinates X, W, D, μ, Δ, and E. A consumer precision request first reads μ, then the chosen window in W, then the dyadic observation in D, and only then the diagonal readback Δ and seal row E.

The non-escape and public-boundary rows follow from the same projection. Since the packet contains no coordinate for a quotient real, host equality proof, selected limit, or ambient completeness theorem, a public $\mathsf{Real}^{\uparrow}$ read cannot export such data. □

Theorem 378.3 (CauchyCompletionMonad unit boundary)**.** *For every displayed rational row accepted by* $\mathsf{RegSeqRat}^{\uparrow}$*, the constant regular-rational family determines a* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit packet. Its* $\mathsf{StreamName}^{\uparrow}$ *windows are stationary, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations are the constant rational observations, its schedule row selects the constant family member, its diagonal readback is the same regular-rational row, and its* $\mathsf{Real}^{\uparrow}$ *seal row is reached through that constant packet.*

Proof. Choose the constant regular-rational family described in the habitation row of Definition 378.2. The finite windows and dyadic observations are stationary because every requested precision reads the same rational source row.

The schedule row may select the displayed constant member at each request. The diagonal ledger then repacks the same finite observations as a regular-rational readback, without creating a new limit object.

Finally attach the $\mathsf{Real}^{\uparrow}$ seal row after the diagonal readback. The carrier coordinates of Definition 378.1 contain exactly these rows, so the unit packet is a visible finite BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket unit boundary`

Theorem 378.4 (CauchyCompletionMonad diagonal-bind stability)**.** *Let M and M' be accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packets. If their source-family rows, schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows, diagonal readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows are transported componentwise by displayed* hsame *routes and read through the corresponding* Cont *routes, then the diagonal-bind classifier is stable: every consumer precision read of M and M' follows matched schedule, window, dyadic-observation, diagonal-readback, and seal coordinates.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket diagonal bind stability`

Proof. Unfold the classifier of Definition 378.1. It compares packets through the finite coordinates X, W, D, μ, Δ, E, L, C, P, and N and their displayed transports.

Move the source-family row, schedule, selected windows, dyadic observations, diagonal ledger, and seal row along the stored hsame comparisons. The Cont coordinates keep the same read order from a precision request to the schedule, then to the window and observation rows.

Apply the diagonal handoff row of Definition 378.2. It forces each real-facing read through the transported schedule, finite window, dyadic observation, and diagonal readback before the seal, so the classifier remains stable under the componentwise transport. □

Theorem 378.5 (CauchyCompletionMonad public Real seal factorization)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet factors through the schedule row, the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, and the final seal row. The packet exports no quotient real, selected limit, host real equality, ambient completeness theorem, or hidden stream representative.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket public real seal factorization`

Proof. Start from a public precision request. The carrier makes the schedule row μ the first coordinate that can choose a family member and window for that request.

The chosen window is read from W, its dyadic observation from D, and the corresponding regular-rational readback from Δ. This is exactly the diagonal handoff route stated in Definition 378.2.

The $\mathsf{Real}^{\uparrow}$ seal row E is reached only after that route. Since the carrier has no coordinate for quotient reals, selected limits, host equality, completeness, or hidden representatives, the listed finite rows exhaust every public real-seal factorization. □

Theorem 378.6 (CauchyCompletionMonad schedule composition boundary)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P,$ and $M' = (X', W', D', \mu', \Delta', E', L', C', P', N')$ be accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packets whose source-family rows are transported by displayed* hsame *rows, whose schedules are compatible on the*

inspected precision requests, whose finite StreamName$^{\uparrow}$ *windows and* DyadicRatCore$^{\uparrow}$ *observations match through displayed ledger rows, and whose consumer routes include a displayed* Cont *composition from the schedule of M to the transported schedule of M'. Then the composed finite completion packet still has every public* Real$^{\uparrow}$ *read factor through the composed schedule, the matched window ledger, the matched dyadic observation ledger, the* RegSeqRat$^{\uparrow}$ *diagonal readback, and the seal row. The composition exports no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket schedule composition boundary`

Proof. For constant schedules, apply the unit boundary of Definition 378.3. The composed packet has stationary windows, stationary dyadic observations, the same diagonal readback, and the same seal-facing route.

For transported schedules, use the diagonal-bind stability of Definition 378.4. The displayed hsame rows move source families, schedules, windows, observations, diagonal readbacks, and seal rows componentwise, while the displayed Cont composition preserves the consumer order.

It remains to identify the public route. Apply Definition 378.5. Every public read must pass through schedule, window, dyadic observation, diagonal readback, and seal coordinates, so the composed packet exposes exactly the same finite completion surface. □

Theorem 378.7 (CauchyCompletionMonad left-unit factorization)**.** *Let r be a displayed rational row accepted by* RegSeqRat$^{\uparrow}$*, and let M be the* CauchyCompletionMonad$^{\uparrow}$ *packet obtained by first forming the unit packet for r and then composing it through an accepted finite completion packet by the schedule composition boundary. Every public* Real$^{\uparrow}$ *read of the composite factors through the original constant regular-rational row, stationary* StreamName$^{\uparrow}$ *windows, constant* DyadicRatCore$^{\uparrow}$ *observations, the same* RegSeqRat$^{\uparrow}$ *diagonal readback, and the same seal row. The left-unit route exports no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket left unit factorization`

Proof. Start with the unit packet supplied by Definition 378.3. Its source family is constant, its finite windows are stationary, its dyadic observations are constant rational observations, and its diagonal readback is the original regular-rational row.

Use Definition 378.6 to compose this unit packet with the next accepted packet. The unit schedule selects the same source row at every precision, so the composed schedule has no additional consumer-visible choice before the stationary window and dyadic-observation rows.

Transport the diagonal and seal-facing rows with Definition 378.4. The componentwise hsame rows and displayed Cont routes preserve the original regular-rational readback and its seal-facing coordinate.

Finally apply Definition 378.5. The only public route runs through the constant row, stationary windows, constant dyadic observations, diagonal readback, and seal row, and the carrier contains no coordinate for a quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row. □

Theorem 378.8 (CauchyCompletionMonad right-unit factorization)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$ be an accepted* CauchyCompletionMonad$^{\uparrow}$ *packet. If M is composed on the seal-facing side with the unit boundary for its displayed* RegSeqRat$^{\uparrow}$ *diagonal readback, then every public consumer read preserves the packet's schedule row, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations,*

RegSeqRat$^{\uparrow}$ *diagonal readback, and public seal factorization. The right-unit route exports no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row.*

Proof. Begin with the accepted packet M and its diagonal readback Δ. The public factorization of Definition 378.5 already routes every real-facing read through μ, W, D, Δ, and the seal row E.

Attach the unit packet of Definition 378.3 to the displayed diagonal readback. Since that unit has stationary windows and constant dyadic observations over the same readback, it contributes no new schedule choice beyond the existing seal-facing read route.

Apply Definition 378.4 to compare the original packet with the packet followed by this unit. The stored hsame and Cont rows identify the schedule, window, observation, diagonal, and seal coordinates componentwise.

The schedule composition boundary of Definition 378.6 then identifies the composed consumer route with the original route. Thus the right-unit composite preserves the same packet ledger, and no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row is introduced. □

Theorem 378.9 (CauchyCompletionMonad bind associativity ledger)**.** *Let M, M', and M'' be accepted* CauchyCompletionMonad$^{\uparrow}$ *packets whose two successive finite bind compositions are defined by displayed schedule, window, dyadic-observation, diagonal, and seal ledger rows. The two associations of the bind composition have the same consumer-visible route through the composed schedule, selected finite* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *observation,* RegSeqRat$^{\uparrow}$ *diagonal readback,* Real$^{\uparrow}$ *seal,* BHist *entries, displayed* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows. Neither association exports a quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row.*

Proof. First compare each single bind step with Definition 378.6. Both associations must expose the same kind of finite route: schedule, selected window, dyadic observation, diagonal readback, and seal row.

Next apply Definition 378.4 to the intermediate packets. The componentwise hsame rows identify source-family rows, schedules, selected windows, dyadic observations, diagonal readbacks, and seal coordinates, while the displayed Cont rows keep the consumer order fixed.

The unit boundary of Definition 378.3 supplies the neutral constant packets at the ends of the ledger. These packets use stationary windows and constant dyadic observations, so they do not alter the visible BHist, Pkg, or local NameCert rows of the two associations.

Finally use Definition 378.5 for the public readout. Since every route passes through the same finite schedule, window, observation, readback, seal, BHist, hsame, Cont, Pkg, and local NameCert coordinates, associating the two bind steps in either order exposes the same ledger and no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket bind associativity ledger`

Theorem 378.10 (CauchyCompletionMonad downstream RegSeqRat consumer route)**.** *Every downstream* RegSeqRat$^{\uparrow}$ *or* RealCauchyCompletion$^{\uparrow}$ *consumer read of an accepted* CauchyCompletionMonad$^{\uparrow}$ *packet factors through the source regular-Cauchy family row, the unary schedule, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations, the* RegSeqRat$^{\uparrow}$ *diagonal readback, the public* Real$^{\uparrow}$ *seal row,* BHist *ledger entries, displayed* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *coordinates. The route exports no quotient stream, host real equality, selected limit, hidden representative, or ambient completeness principle.*

Proof. Start with an accepted packet $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$. A downstream consumer precision request first reads the unary schedule μ, so its source member is selected inside the displayed regular-Cauchy family row X rather than by an external selector.

The selected request then passes through the finite $\mathsf{StreamName}^{\uparrow}$ window in W and the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ observation in D. Apply Definition 378.4 to transport these rows along the stored hsame comparisons and the displayed Cont read routes without changing the classifier-facing surface.

Next use Definition 378.6. It identifies composed consumer reads with the same schedule, window, dyadic-observation, and diagonal-readback route, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains a finite packet route.

Finally apply Definition 378.3 for constant source packets and Definition 378.5 for public real-seal reads. The carrier rows BHist, hsame, Cont, Pkg, and N exhaust the public coordinates, so quotient streams, host real equality, selected limits, hidden representatives, and ambient completeness cannot be exported by the downstream route. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket downstream regseqrat consumer route`

Theorem 378.11 (CauchyCompletionMonad downstream completion consumer exactness)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet. Every downstream completion read by a* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or sealed* $\mathsf{Real}^{\uparrow}$ *consumer is exact over the displayed route: the schedule row μ, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Δ, the visible* $\mathsf{Real}^{\uparrow}$ *seal row E, the ledger row L, the* BHist *entries carried there, the displayed* hsame *transports, the displayed* Cont *routes C, the* Pkg *provenance P, and the local* NameCert *row N. Such a consumer reads no quotient stream, selected host limit, ambient completeness principle, hidden representative, or unlisted equality coordinate.*

Proof. Start with the downstream route of Definition 378.10. It already forces every downstream $\mathsf{RegSeqRat}^{\uparrow}$ or completion-facing read through the source family row, schedule, window, dyadic observation, diagonal readback, seal row, and provenance package.

Next apply the public real seal factorization of Definition 378.5. It identifies the only $\mathsf{Real}^{\uparrow}$-facing exit as the displayed seal row E after the same μ, W, D, Δ route has been read.

Finally use the schedule composition boundary of Definition 378.6. The BHist ledger L, the hsame transports, the Cont routes C, the Pkg row P, and the local NameCert row N record the consumer route without adding a second completion source or an ambient real-completeness witness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket downstream completion consumer exactness`

Theorem 378.12 (CauchyCompletionMonad unit bind visible seal preservation)**.** *For every accepted packet $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$ of* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*, binding on either side by a displayed unit packet preserves the same visible* $\mathsf{Real}^{\uparrow}$ *seal route through $X, W, D, \mu, \Delta, E, L, C, P, N$. The preserved route is read by downstream* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers only through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback,* BHist *ledger entries, displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *coordinates already present in M.*

Proof. Project the unit boundary of Definition 378.3. A unit packet contributes stationary windows, constant dyadic observations, the same regular-rational readback shape, and a visible seal row, but it has no coordinate for a hidden representative or a selected completion limit.

Apply diagonal-bind stability from Definition 378.4. The componentwise hsame rows compare the original packet with the unit-bound packet, and the displayed Cont routes preserve the order in which the schedule, window, observation, diagonal, and seal coordinates are read.

Finish with Definition 378.10. Since every downstream consumer still enters through the same finite schedule-window-observation-diagonal-seal route, unit binding preserves the visible seal surface over $X, W, D, \mu, \Delta, E, L, C, P, N$. □

Theorem 378.13 (CauchyCompletionMonad associative bind consumer route)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P$ $M' = (X', W', D', \mu', \Delta', E', L', C', P', N')$, and $M'' = (X'', W'', D'', \mu'', \Delta'', E'', L'', C'', P'', N'')$ be accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packets whose two displayed bind associations are defined. A downstream completion consumer of either association reads the same composed schedule, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows,* BHist *ledgers, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows. The two associations therefore present one consumer route over the displayed packet rows and export no quotient real, selected host limit, hidden stream representative, host equality, or ambient completeness datum.*

Proof. First compare the two associations with the bind associativity ledger of Definition 378.9. It places both bracketings on the same composed schedule, finite window, dyadic observation, diagonal readback, and seal coordinates.

Next use the schedule composition boundary of Definition 378.6 together with diagonal-bind stability from Definition 378.4. These results identify the intermediate packets by their displayed hsame rows and keep the consumer order fixed through the displayed Cont routes.

Finally apply the downstream route of Definition 378.10. The BHist ledgers, Pkg provenance rows, and local NameCert rows are the only exported bookkeeping coordinates, so the two bind associations expose the same completion-consumer route. □

Theorem 378.14 (CauchyCompletionMonad left unit law)**.** *Let r be a displayed constant regular-rational row accepted by* $\mathsf{RegSeqRat}^{\uparrow}$*. If the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit packet of r is followed by a diagonal bind packet, then the resulting classifier route factors through the same stationary* $\mathsf{StreamName}^{\uparrow}$ *windows, constant* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *ledger entries, displayed* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row as the unit packet. No quotient real, selected limit, host real equality, hidden representative, or ambient completeness datum is exported.*

Proof. Begin with the unit boundary of Definition 378.3. It supplies the constant source row, stationary finite windows, constant dyadic observations, the same regular rational readback, and the seal-facing row.

Apply diagonal-bind stability from Definition 378.4. The componentwise hsame rows and displayed Cont routes transport the schedule, window, observation, readback, and seal coordinates without changing the classifier-facing packet.

Use the schedule composition boundary of Definition 378.6. Since the first schedule is the constant unit schedule, the composed route has the same inspected window and dyadic-observation rows as the original constant packet.

Finally apply Definition 378.5 and Definition 378.10. They identify every public and downstream read with the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, BHist, hsame, Cont, Pkg, and NameCert rows, so the left unit is a finite packet law. □

Theorem 378.15 (CauchyCompletionMonad right unit law)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$ be an accepted* CauchyCompletionMonad$^{\uparrow}$ *packet. Binding M against the constant unit readback of its displayed* RegSeqRat$^{\uparrow}$ *diagonal row preserves every public* Real$^{\uparrow}$ *seal route through X, W, D, μ, Δ, E, L, C, P, and N. The right-unit bind exports no quotient real, selected limit, host real equality, hidden stream representative, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket right unit visible seal preservation`

Proof. Start with the accepted packet M. The public seal factorization of Definition 378.5 forces each real-facing read through the schedule, selected window, dyadic observation, diagonal readback, and seal row.

Attach the constant unit readback supplied by Definition 378.3. Its stationary windows and constant dyadic observations are read from the already displayed diagonal row, so it contributes no independent route to a public real.

Apply Definition 378.4. The displayed hsame and Cont rows compare the original packet and the packet followed by the unit along the same finite coordinates.

The schedule composition boundary of Definition 378.6 repacks this comparison as the original consumer route. The downstream route of Definition 378.10 then rules out quotient reals, selected limits, host equality, hidden representatives, and ambient completeness data. □

Theorem 378.16 (CauchyCompletionMonad associative bind law)**.** *Let M, M', and M'' be accepted* CauchyCompletionMonad$^{\uparrow}$ *packets with two displayed schedule-composition routes over the same selected* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observation rows, diagonal* RegSeqRat$^{\uparrow}$ *readbacks,* Real$^{\uparrow}$ *seals,* BHist *entries, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. If one route first binds M with M' and then binds the result with M'', while the other first binds M' with M'' and then binds M with the result, then the two routes have the same classifier-facing finite packet and export no quotient real, selected limit, host real equality, hidden representative, or ambient completeness row.*

Proof. First apply the schedule composition boundary of Definition 378.6 to each displayed association. Both public routes must pass through the same schedule, finite window, dyadic observation, diagonal readback, and seal coordinates.

Next use diagonal-bind stability from Definition 378.4. The componentwise hsame comparisons and displayed Cont routes identify the intermediate packets without changing the consumer-facing order.

The unit boundary of Definition 378.3 fixes the neutral constant packets at the ends of the displayed composition ledger. These packets have stationary windows and constant dyadic observations, so they do not alter the shared BHist, Pkg, or NameCert rows.

Finally apply the public seal factorization of Definition 378.5 and the downstream consumer route of Definition 378.10. Since every consumer read is exhausted by the same finite coordinates, the two associations expose the same classifier-facing packet. □

Theorem 378.17 (CauchyCompletionMonad regular seal unit)**.** *A constant* RegSeqRat$^{\uparrow}$ *source row with stationary* StreamName$^{\uparrow}$ *windows, constant* DyadicRatCore$^{\uparrow}$ *observations, identity* Cont *routes, componentwise reflexive* hsame *rows, the constant diagonal* RegSeqRat$^{\uparrow}$ *readback, and the displayed* Real$^{\uparrow}$ *seal forms the unit packet of* CauchyCompletionMonad$^{\uparrow}$*. This packet is a carrier of Definition 378.1 whose public read route is the constant schedule, the stationary finite window, the constant dyadic observation, the same regular readback, and the seal row.*

Proof. Project the carrier coordinates of Definition 378.1. The constant source row supplies X, the stationary window ledger supplies W, the constant dyadic observations supply D, and the constant schedule supplies μ.

Use the unit boundary of Definition 378.3. It identifies the diagonal readback with the same regular rational row and routes the seal-facing coordinate through the constant packet rather than through a quotient or selected limit.

Finally apply the public real seal factorization of Definition 378.5. The identity Cont routes and reflexive hsame rows keep the read on the displayed constant schedule, window, observation, readback, and $\mathsf{Real}^{\uparrow}$ seal coordinates, so the packet is the advertised unit. □

Theorem 378.18 (CauchyCompletionMonad unit seal route)**.** *For a displayed constant row r : $\mathsf{RegSeqRat}^{\uparrow}$, the unit carrier of $\mathsf{CauchyCompletionMonad}^{\uparrow}$ is the finite* BHist *route*

$$r \longmapsto W_r \longmapsto D_r \longmapsto \Delta_r \longmapsto E_r$$

where W_r is the stationary $\mathsf{StreamName}^{\uparrow}$ window ledger, D_r is the constant $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, Δ_r is the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E_r is the displayed $\mathsf{Real}^{\uparrow}$ seal. The route is transported only by componentwise hsame *rows, read only through displayed* Cont *coordinates, and recorded in the* Pkg *provenance and local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ row.*

Proof. Start from the regular seal unit of Definition 378.17. It supplies the constant $\mathsf{RegSeqRat}^{\uparrow}$ source row, stationary $\mathsf{StreamName}^{\uparrow}$ windows, constant $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the diagonal readback, and the $\mathsf{Real}^{\uparrow}$ seal.

The carrier definition gives the finite BHist coordinates that hold this route. Reflexive hsame rows identify the constant windows, observations, readback, and seal with themselves, while the displayed Cont rows keep the consumer read on those coordinates.

The Pkg row records provenance over the same visible primitives, and the local NameCert row records that no quotient real, selected limit, host equality, hidden stream representative, or ambient completeness datum is part of the unit route. □

Theorem 378.19 (CauchyCompletionMonad bind diagonal route)**.** *Let M and M' be accepted $\mathsf{CauchyCompletionMonad}^{\uparrow}$ packets. A displayed bind composition*

$$M \text{ bind } M' \rightsquigarrow (\mu \star \mu',\, W \star W',\, D \star D',\, \Delta \star \Delta',\, E \star E')$$

preserves the schedule-window-observation-diagonal-seal route: every public read passes through a matched $\mathsf{StreamName}^{\uparrow}$ window, a matched $\mathsf{DyadicRatCore}^{\uparrow}$ observation, a diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback, and a $\mathsf{Real}^{\uparrow}$ seal inside the same BHist *packet surface. The preservation uses only displayed* hsame *transport, displayed* Cont *composition,* Pkg *provenance, and the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ rows.*

Proof. Apply diagonal-bind stability from Definition 378.4. The source-family rows, schedules, finite windows, dyadic observations, diagonal readbacks, and real-seal rows move along componentwise hsame comparisons.

Use the schedule composition boundary of Definition 378.6. It fixes the consumer order: schedule first, then the selected $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation, the $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback, and finally the $\mathsf{Real}^{\uparrow}$ seal.

The resulting route stays inside the same finite BHist surface. The displayed Cont composition records the read path, the Pkg rows record provenance, and the local NameCert rows rule out any escape to quotient reals, selected limits, host equality, hidden representatives, or ambient completeness. □

Theorem 378.20 (CauchyCompletionMonad NameCert obligation surface)**.** *The concrete* $\mathsf{NameCert}_{\mathsf{CauchyCompletion}}$ *surface consists of carrier habitation, classifier transport, the unit seal route, the bind diagonal route, real-seal non-escape, and the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer boundary. Each row is expressed in the finite* BHist *coordinates over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *transported by* hsame, *read through* Cont, *and recorded by* Pkg *provenance together with the local* NameCert *row.*

Proof. Carrier habitation is supplied by the constant packet of Definition 378.2. Classifier transport is the componentwise movement of schedule, window, observation, readback, and seal rows along the stored hsame comparisons and displayed Cont reads.

The unit row is exactly Definition 378.18. It exhibits a constant $\mathsf{RegSeqRat}^{\uparrow}$ source, stationary $\mathsf{StreamName}^{\uparrow}$ windows, constant $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the diagonal readback, and the $\mathsf{Real}^{\uparrow}$ seal inside one BHist packet.

The bind row is Definition 378.19. It composes schedules, windows, observations, diagonal readbacks, and seals without leaving the displayed Pkg provenance or local NameCert surface.

For the real-seal and downstream boundaries, combine Definition 378.5 with Definition 378.10. Public $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ reads are exhausted by the finite route, so quotient reals, selected limits, host equality, hidden stream representatives, and ambient completeness rows are not exported. □

Theorem 378.21 (CauchyCompletionMonad bind associativity boundary)**.** *Let M, M', and M'' be accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packets whose two successive schedule compositions are defined by displayed* BHist *ledger rows. The public* $\mathsf{Real}^{\uparrow}$ *read route obtained by first composing M with M' and then with M'' is the same route as the one obtained by first composing M' with M'' and then composing the result with M: in both cases every read passes through the matched schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *comparisons, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *coordinates.*

Proof. First apply the schedule composition boundary of Definition 378.6 to the left association. Its public read route is forced through the composed schedule, matched finite windows, matched dyadic observations, diagonal readback, and seal row.

Apply the same schedule composition boundary to the right association. The hypotheses give the same displayed schedule, window, observation, diagonal, seal, BHist, Cont, Pkg, and local NameCert rows, so the two public routes have the same visible coordinates.

It remains to compare the intermediate packets. Use diagonal-bind stability from Definition 378.4; the componentwise hsame rows transport the two associations across the matched ledgers without changing the consumer order.

The bind associativity ledger of Definition 378.9 then repacks these matched coordinates as the same public $\mathsf{Real}^{\uparrow}$ read route. No additional quotient stream, selected limit, host equality, hidden representative, or ambient completeness coordinate is created by either association. □

Theorem 378.22 (CauchyCompletionMonad finite-window exhaustion)**.** *Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet is exhausted by the finite window route through the source family, schedule row, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *ledger, displayed* hsame, *displayed* Cont, Pkg *provenance, and local* NameCert *coordinates. Such a read cannot expose quotient streams, selected limits, host equality, hidden representatives, or ambient completeness data.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket finite window exhaustion`

Proof. Begin with a downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{RealCauchyCompletion}^{\uparrow}$ request. The downstream consumer route of Definition 378.10 forces the request through the source family, schedule, finite window, dyadic observation, diagonal readback, seal row, and public ledger coordinates.

For a $\mathsf{Real}^{\uparrow}$ request, apply the public real seal factorization of Definition 378.5. It gives the same finite route ending at the seal row, with no independent real-completeness channel.

The non-escape clauses are exactly the absent fields in the NameCert obligations of Definition 378.2. Since the accepted packet contains no coordinate for quotient streams, selected limits, host equality, hidden representatives, or ambient completeness, the finite window route exhausts every listed downstream read. □

Definition 378.23 (CauchyCompletionMonad RealCauchyCompletion handoff packet)**.** *A* CauchyCompletionMona *to* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff packet is the finite route*

$$(X, \mu, W, D, \Delta, E, L, C, P, N) \rightsquigarrow \mathsf{RealCauchyCompletion}^{\uparrow}$$

where X is the source regular-Cauchy family row, μ selects the inspected family member and precision window, W records the finite $\mathsf{StreamName}^{\uparrow}$ *windows, D records the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, Δ is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, E is the public* $\mathsf{Real}^{\uparrow}$ *seal, L records the displayed* BHist *and* hsame *comparisons, C records the displayed* Cont *consumer routes, P is the* Pkg *provenance row, and N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ *row. A* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer may read the packet only through this ordered route.*

Theorem 378.24 (CauchyCompletionMonad RealCauchyCompletion input factorization)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *input read of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet factors through the handoff packet of Definition 378.23. The read starts at a source regular-Cauchy family row, passes through the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, is repacked by the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, reaches the public* $\mathsf{Real}^{\uparrow}$ *seal, and is transported only by displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *rows.*

Proof. Start with a $\mathsf{RealCauchyCompletion}^{\uparrow}$ input request. The unit boundary of Definition 378.3 supplies the boundary case: a constant regular-Cauchy source has stationary finite windows, constant dyadic observations, a diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the displayed $\mathsf{Real}^{\uparrow}$ seal.

For transported and composed packets, use diagonal-bind stability from Definition 378.4. It keeps the family row, schedule, selected window, dyadic observation, diagonal readback, and seal aligned by componentwise hsame rows and displayed Cont routes.

Apply the public real seal factorization of Definition 378.5. The final consumer-facing row is the $\mathsf{Real}^{\uparrow}$ seal reached after the diagonal readback, not an external real equality or selected limit.

Finally use finite-window exhaustion from Definition 378.22. It says every $\mathsf{RealCauchyCompletion}^{\uparrow}$ read is exhausted by the same finite route, so the request factors through the handoff packet and through no other coordinate. □

Theorem 378.25 (CauchyCompletionMonad RealCauchyCompletion consumer package)**.** *The downstream* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer package exported by an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet is exactly the handoff packet of Definition 378.23 together with the unit boundary, diagonal-bind stability, public* $\mathsf{Real}^{\uparrow}$ *seal factorization, and finite-window exhaustion rows. The package supplies source regular-Cauchy family reads, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist, hsame, Cont, Pkg, *and local* NameCert *coordinates, and it exports no quotient stream, host real equality, selected limit, hidden representative, or ambient completeness datum.*

Proof. Assemble the source side from the handoff packet definition. It gives the ordered consumer route through X, μ, W, D, Δ, E, L, C, P, and N.

Attach the unit and bind rows. The unit boundary supplies constant input packets, while diagonal-bind stability transports composed packets without changing the displayed read order.

Attach the public seal and exhaustion rows. The public real seal factorization identifies the seal reached after diagonal readback, and finite-window exhaustion identifies the same finite route as the whole $\mathsf{RealCauchyCompletion}^{\uparrow}$ input surface.

Since the assembled package contains only the displayed family, window, dyadic, readback, seal, history, transport, continuation, provenance, and naming rows, it has no coordinate for quotient streams, host real equality, selected limits, hidden representatives, or ambient completeness. □

Theorem 378.26 (CauchyCompletionMonad regular-family unit)**.** *Every constant* $\mathsf{RegSeqRat}^{\uparrow}$ *family row determines the displayed* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit packet through stationary* $\mathsf{StreamName}^{\uparrow}$ *windows, constant* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the same diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the public* $\mathsf{Real}^{\uparrow}$ *seal row, finite* BHist *coordinates, reflexive* hsame *rows, identity* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$ *row. The packet exports exactly the unit route already visible in those rows, and it exports no quotient stream, selected limit, host equality, hidden representative, or ambient completeness datum.*

Proof. Start with the constant regular-rational family from the habitation row of Definition 378.2. The unit boundary of Definition 378.3 supplies the stationary $\mathsf{StreamName}^{\uparrow}$ windows, constant $\mathsf{DyadicRatCore}^{\uparrow}$ observations, constant schedule, diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback, and public $\mathsf{Real}^{\uparrow}$ seal row.

The regular seal unit of Definition 378.17 places these rows inside the carrier of Definition 378.1. The unit seal route of Definition 378.18 then records the same route in BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ coordinates.

Finally use the public real seal factorization of Definition 378.5. Since the factorization has no coordinate for quotient streams, selected limits, host equality, hidden representatives, or ambient completeness, the constant family row gives precisely the displayed unit packet. □

Theorem 378.27 (CauchyCompletionMonad bind closure package)**.** *The finite bind surface of* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *consists of the left-unit packet, the right-unit packet, bind associativity, schedule composition, and diagonal-bind stability, all expressed through the same source-family, finite-window, dyadic-observation, diagonal-readback,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. This package closes the public bind route for downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers without adding quotient streams, selected limits, host equality, hidden representatives, or ambient completeness.*

Proof. For the unit rows, combine the left-unit factorization of Definition 378.7, the right-unit factorization of Definition 378.8, and the explicit unit laws of Definition 378.14 and Definition 378.15. These rows keep every public read inside the displayed schedule, window, dyadic-observation, readback, seal, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates.

For composition, use the schedule composition boundary of Definition 378.6. Diagonal-bind stability from Definition 378.4 identifies the transported source-family rows, selected windows, dyadic observations, readbacks, and seals componentwise.

For associativity, apply the bind associativity ledger of Definition 378.9 and the bind associativity boundary of Definition 378.21. The associative bind law of Definition 378.16 repacks the two displayed associations as one consumer-facing finite route.

The finite-window exhaustion theorem Definition 378.22 then gives the consumer boundary for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The resulting bind package has no exported coordinate for quotient streams, selected limits, host equality, hidden representatives, or ambient completeness. □

Theorem 378.28 (CauchyCompletionMonad standard completion boundary)**.** *The regular-family unit package of Definition 378.26, the bind closure package of Definition 378.27, the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer route, the public* $\mathsf{Real}^{\uparrow}$ *seal factorization, and finite-window exhaustion assemble the standard completion-boundary package for* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*. The boundary is a finite BEDC package over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, and it does not include quotient streams, selected limits, host equality, hidden representatives, or ambient completeness.*

Proof. First take the unit side from Definition 378.26. It gives the constant regular-family packet and fixes the stationary window, constant dyadic-observation, diagonal readback, seal, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows used by a completion boundary.

Next take the bind side from Definition 378.27. The package supplies left unit, right unit, associativity, schedule composition, and diagonal-bind stability over the same finite packet coordinates.

Now attach the downstream consumer surface. The route of Definition 378.10 serves $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers, while the public real seal factorization of Definition 378.5 serves $\mathsf{Real}^{\uparrow}$ consumers.

Finally apply finite-window exhaustion from Definition 378.22. It states that every listed consumer read is exhausted by the finite window route, so the assembled package is the promised standard completion boundary and not an assertion of quotient streams, selected limits, host equality, hidden representatives, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadUp StdBridge`

Theorem 378.29 (CauchyCompletionMonad schedule totality)**.** *For every accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet* $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$*, each consumer precision request is answered first by the displayed unary schedule row* μ*, then by one finite* $\mathsf{StreamName}^{\uparrow}$ *window in* W*, and then by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation attached to that window. The request cannot be answered by an external selector outside the packet coordinates.*

Proof. Start with the carrier description in Definition 378.1. The only coordinate that names which family member and which window answer a precision request is the unary schedule row μ.

Apply the schedule composition boundary of Definition 378.6. It keeps composed consumer reads on the displayed schedule, matched window ledger, dyadic-observation ledger, diagonal readback, and seal row.

Finally use finite-window exhaustion from Definition 378.22. Every downstream read is exhausted by the finite route through μ, W, and D, so no further selector is available. □

Theorem 378.30 (CauchyCompletionMonad diagonal readback coverage)**.** *For every accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet* $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$*, each scheduled window and dyadic-observation pair has a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback* Δ *before the public* $\mathsf{Real}^{\uparrow}$ *seal row is exposed. Thus a consumer cannot reach the seal row directly from a window or observation coordinate.*

Proof. Project the carrier coordinates from Definition 378.1. The rows W and D record finite windows and dyadic observations, while Δ is the diagonal transport ledger that repacks the scheduled observations as regular rational readback data.

Use the public seal factorization of Definition 378.5. A public $\mathsf{Real}^{\uparrow}$ read passes through schedule, selected window, dyadic observation, $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback, and only then the final seal row.

The downstream consumer route of Definition 378.10 identifies the same read order for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers. Hence every scheduled window and observation pair is covered by Δ before any public seal exposure. □

Theorem 378.31 (CauchyCompletionMonad Real seal naturality)**.** *Componentwise* hsame *transport and displayed* Cont *routes commute with the passage from source-family, schedule, finite-window, and dyadic-observation rows to the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and public* $\mathsf{Real}^{\uparrow}$ *seal row of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet.*

Proof. Begin with two accepted packets related by the componentwise transport rows of the carrier. The classifier of Definition 378.1 compares source families, schedules, windows, dyadic observations, diagonal readbacks, and seal rows only through the displayed hsame and Cont coordinates.

Apply diagonal-bind stability from Definition 378.4. It moves the scheduled windows, dyadic observations, diagonal ledger, and seal row along the componentwise transports while preserving the consumer read order.

Now use the public seal factorization of Definition 378.5. Because every public seal read factors through the transported diagonal readback, the transport route commutes with the passage from scheduled data to Δ and then to the $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 378.32 (CauchyCompletionMonad completion root unblock package)**.** *The finite completion-root obligation surface for* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *consists of schedule totality, diagonal readback coverage, public* $\mathsf{Real}^{\uparrow}$ *seal naturality, the regular-family unit package, bind closure, downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer routes, and the standard completion boundary. This is the package consumed by* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; it contains only displayed source-family rows, unary schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seals,* BHist, hsame, Cont, Pkg, *and local* NameCert *coordinates.*

Proof. Assemble the request side from schedule totality in Definition 378.29. Each consumer precision request is routed by the displayed unary schedule and one finite window before it reaches any dyadic observation.

Attach diagonal readback coverage from Definition 378.30 and seal naturality from Definition 378.31. These rows place the $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback between every scheduled observation and the public $\mathsf{Real}^{\uparrow}$ seal, and they keep that placement stable under hsame transport and displayed Cont routes.

Add the regular-family unit package of Definition 378.26 and the bind closure package of Definition 378.27. The downstream consumer route of Definition 378.10 and the standard completion boundary of Definition 378.28 then give the same finite surface to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The assembled package has no coordinate for quotient streams, selected limits, host equality, hidden representatives, or ambient completeness. Since every read is exhausted by the displayed schedule, finite window, dyadic observation, diagonal readback, and seal route, none of those excluded objects can be supplied by the completion-root package. □

Theorem 378.33 (CauchyCompletionMonad root schedule-window normal form)**.** *Every completion-root request received by an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet factors first through the displayed unary schedule, then through one finite* $\mathsf{StreamName}^{\uparrow}$ *window, and then through one* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation attached to that window. The normal form is read only through the source-family rows, schedule row, finite-window ledger, dyadic-observation ledger, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Start from the schedule totality theorem Definition 378.29. It places each consumer precision request at the unary schedule row before any window or observation coordinate is visible.

Next apply finite-window exhaustion from Definition 378.22. The scheduled request is exhausted by one finite $\mathsf{StreamName}^{\uparrow}$ window and its attached $\mathsf{DyadicRatCore}^{\uparrow}$ observation, so no second window family or ambient selector is available.

Finally read the displayed Cont routes from the carrier and from the completion root package of Definition 378.32. These routes keep the schedule-window-observation path inside BHist, hsame, Cont, Pkg, and NameCert coordinates. □

Theorem 378.34 (CauchyCompletionMonad root diagonal-seal normal form)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet factors through the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback after the scheduled dyadic observation and before the public seal row. Thus the seal is reached by the finite route through schedule, window, dyadic observation, diagonal readback, displayed* hsame *transport, displayed* Cont *route,* Pkg*, and local* NameCert *provenance.*

Proof. Use diagonal readback coverage from Definition 378.30. It places the $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback after the scheduled window and dyadic observation and before any public seal exposure.

Apply seal naturality from Definition 378.31. Componentwise hsame transport and displayed Cont routes commute with the move from scheduled dyadic observations to the diagonal readback.

Finish with public seal factorization in Definition 378.5. The public $\mathsf{Real}^{\uparrow}$ read therefore has no direct route from a source-family row, window row, or dyadic-observation row to the seal without passing through the displayed diagonal readback. □

Theorem 378.35 (CauchyCompletionMonad root consumer exhaustion)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$, RealCauchyCompletion *and* $\mathsf{Real}^{\uparrow}$ *consumers of the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-root package have no coordinate beyond source-family rows, unary schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Proof. First apply the schedule-window normal form of Definition 378.33. It exhausts the request side of each consumer through the schedule, one finite window, and one dyadic observation.

Next apply the diagonal-seal normal form of Definition 378.34. It exhausts the seal-facing side through the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback and the public $\mathsf{Real}^{\uparrow}$ seal row.

The completion root package of Definition 378.32 assembles exactly those request-side and seal-facing rows for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. Since the package lists no quotient stream, selected limit, host equality, hidden representative, or ambient completeness coordinate, no consumer can read such a coordinate from the root surface. □

Definition 378.36 (CauchyCompletionMonad root consumer face triad)**.** *For an accepted* CauchyCompletionMonad *packet* $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$*, the root consumer face triad is the ordered public package consisting of the* $\mathsf{RegSeqRat}^{\uparrow}$ *face, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *face, and the* $\mathsf{Real}^{\uparrow}$ *face read through the same schedule-window-observation-diagonal-seal packet. The* $\mathsf{RegSeqRat}^{\uparrow}$ *face reads the source-family row* X*, the schedule row* μ*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the*

$\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, and the diagonal readback Δ. The* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *face reads that same route as the completion-consumer source package. The* $\mathsf{Real}^{\uparrow}$ *face reads the public seal row E only after the same scheduled observation and diagonal readback. All three faces use the displayed* BHist, hsame, Cont, Pkg, *and local* NameCert *rows L, C, P, N.*

Theorem 378.37 (CauchyCompletionMonad root face coverage)**.** *Every accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet supplies the root consumer face triad of Definition 378.36. In particular, the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *public faces are obtained from the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows by the single route through source-family data, unary schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and public* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. First use the root schedule-window normal form of Definition 378.33. It places every consumer precision request on the displayed unary schedule, finite window, and dyadic observation route before any downstream face is read.

Next use the root diagonal-seal normal form of Definition 378.34. It inserts the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback between the scheduled observation and the public $\mathsf{Real}^{\uparrow}$ seal, so the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ faces are read from one packet route.

The root consumer exhaustion theorem Definition 378.35 identifies the same finite route as the full consumer surface. The downstream consumer route of Definition 378.10 then lets $\mathsf{RealCauchyCompletion}^{\uparrow}$ consume exactly that source package, and the public seal factorization of Definition 378.5 supplies the final $\mathsf{Real}^{\uparrow}$ face. □

Theorem 378.38 (CauchyCompletionMonad root face non-escape)**.** *The root consumer face triad of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet exports no quotient stream, selected limit, host equality, hidden representative, or ambient completeness coordinate. Each public face is exhausted by the same displayed source-family, schedule, finite-window, dyadic-observation, diagonal-readback, seal,* BHist, hsame, Cont, Pkg, *and local* NameCert *rows.*

Proof. Begin with Definition 378.37. It gives the three public faces by the single schedule-window-observation-diagonal-seal route.

Apply root consumer exhaustion from Definition 378.35. That theorem lists the only coordinates available to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers at the completion root.

Finally use public seal factorization from Definition 378.5. Since even the public $\mathsf{Real}^{\uparrow}$ seal factors through the displayed diagonal readback and recorded provenance rows, none of the three faces can expose a quotient stream, selected limit, host equality, hidden representative, or ambient completeness coordinate. □

Theorem 378.39 (CauchyCompletionMonad schedule source membership)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P, N$ be an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet. For every consumer precision request routed by μ, the scheduled answer names a member of the displayed source-family row X. The membership is part of the same finite packet route through* BHist, hsame, Cont, Pkg, *and the local* NameCert *row; it is not supplied by an external family selector, quotient stream representative, selected limit, host equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 378.1. Its source row X and schedule row μ are coordinates of the same accepted packet, and the schedule-totality clause reads μ as the displayed route answering each precision request.

Apply the root schedule-window normal form of Definition 378.33. Before any window or dyadic observation can be inspected, the request has already passed through the displayed schedule row attached to X.

Finally use the completion root package of Definition 378.32. That package exposes only the finite source-family, schedule, window, observation, readback, seal, provenance, and naming rows, so the scheduled member is locked to X and cannot be spliced in from an external source. □

Theorem 378.40 (CauchyCompletionMonad window observation source lock)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P$ be an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet, and let a consumer precision request be scheduled by μ to a displayed member of X. The selected* $\mathsf{StreamName}^{\uparrow}$ *window in W and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in D are attached to that same scheduled source-family member and to the same requested precision route. No accepted read may replace the selected window or dyadic observation by data from another member of X or from an unlisted source row.*

Proof. Start with schedule source membership from Definition 378.39. It identifies the scheduled answer as a member of the displayed source-family row X before any window or observation coordinate is read.

Next apply the root schedule-window normal form of Definition 378.33. The scheduled request factors through one finite $\mathsf{StreamName}^{\uparrow}$ window and then through the $\mathsf{DyadicRatCore}^{\uparrow}$ observation attached to that window.

Use diagonal readback coverage from Definition 378.30. The same scheduled window and observation are the rows repacked by the diagonal readback, so the precision route, selected member, window, and observation remain one finite packet route.

The public seal factorization of Definition 378.5 excludes any second observation channel. Therefore the window and dyadic observation cannot be supplied by a different source-family member or an ambient selector. □

Theorem 378.41 (CauchyCompletionMonad diagonal seal source lock)**.** *Let $M = (X, W, D, \mu, \Delta, E, L, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet. If a precision request selects a source-family member through μ, then the diagonal readback Δ and the public* $\mathsf{Real}^{\uparrow}$ *seal row E preserve that same source membership through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation. The diagonal and seal route exports the displayed scheduled source, not a second completion source, hidden representative, selected limit, quotient real, or ambient completeness witness.*

Proof. First apply Definition 378.40. The scheduled member of X, selected window, dyadic observation, and precision route are already tied together before the diagonal row is inspected.

Now use the root diagonal-seal normal form of Definition 378.34. It places Δ after the scheduled dyadic observation and before the public $\mathsf{Real}^{\uparrow}$ seal row E.

Finally apply the public seal factorization theorem Definition 378.5. Since every public seal read passes through the same schedule-window-observation-diagonal route, the seal preserves the displayed source membership carried by X and cannot expose a spliced completion source. □

Theorem 378.42 (CauchyCompletionMonad flattened completion source)**.** *Every accepted* $\mathsf{CauchyCompletionMona}$ *packet exposes a flattened completion source before any completion read is consumed: the source-family row, unary schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, and diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback are all read as displayed* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows before the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer reaches a seal-facing coordinate.*

Proof. Begin with the unit boundary of Definition 378.3. It identifies the boundary case as a constant source family with stationary windows, constant dyadic observations, the same diagonal readback, and the displayed $\mathsf{Real}^{\uparrow}$ seal.

For a general accepted packet, apply the completion root package of Definition 378.32. Its schedule-totality and finite-window clauses put the source family, unary schedule, selected window, and dyadic observation before any completion-facing read.

Finally use diagonal-bind stability from Definition 378.4. The componentwise hsame rows and displayed Cont routes preserve the order source family, schedule, window, observation, diagonal readback, and seal, so the completion source is flattened before the consumer reaches the completion boundary. □

Theorem 378.43 (CauchyCompletionMonad flattened seal route)**.** *The flattened completion source of Definition 378.42, schedule composition, diagonal-bind stability, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer package compose into one* $\mathsf{Real}^{\uparrow}$ *seal route. This route reads the accepted packet through source-family rows, schedules, finite windows, dyadic observations, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, displayed transports, package provenance, and local naming before it reaches the public* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. First apply the schedule composition boundary of Definition 378.6. It fixes the consumer order through the displayed schedule, selected finite window, dyadic observation, diagonal readback, and seal row.

Next use diagonal-bind stability from Definition 378.4. The bind step transports the source families, schedules, windows, observations, diagonal readbacks, and seal rows componentwise without changing the read order.

Now attach the downstream consumer package of Definition 378.25. That package is precisely the $\mathsf{RealCauchyCompletion}^{\uparrow}$ reader of the same finite route.

Finish with the public $\mathsf{Real}^{\uparrow}$ seal factorization of Definition 378.5. Since every public seal read passes through the diagonal readback after the scheduled dyadic observation, the composed route is one flattened $\mathsf{Real}^{\uparrow}$ seal route. □

Theorem 378.44 (CauchyCompletionMonad flattened non-escape package)**.** *The flattened completion route for* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *is the package consisting of the flattened completion source of Definition 378.42, the flattened seal route of Definition 378.43, and the completion root package of Definition 378.32. It exports no quotient stream, selected limit, host real equality, hidden representative, or ambient completeness coordinate.*

Proof. Assemble the source side from Definition 378.42. It gives the source-family row, unary schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observations, and diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback before completion consumption.

Assemble the seal side from Definition 378.43. Schedule composition, diagonal-bind stability, the $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer package, and public $\mathsf{Real}^{\uparrow}$ seal factorization make the completion-facing read a single finite seal route.

The completion root package of Definition 378.32 contains exactly those displayed rows and no coordinate for quotient streams, selected limits, host real equality, hidden representatives, or ambient completeness. Therefore the flattened package cannot supply any such object to a downstream completion reader. □

Theorem 378.45 (CauchyCompletionMonad finite bind-domain coverage)**.** *Every bind-domain read of an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet factors through the source-family rows, the unary schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$

readback, the public $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*, displayed* Cont *routes,* Pkg*, and local* NameCert *rows. No bind-domain consumer can read a quotient stream, selected limit, host real equality, hidden representative, or ambient completeness coordinate outside this finite row package.*

Proof. Begin with the schedule composition boundary of Definition 378.6. It puts every bind request into the displayed schedule before any downstream completion reader can inspect a window, dyadic observation, diagonal readback, or seal row.

Next use diagonal-bind stability from Definition 378.4. The bind operation transports the source-family rows, schedules, finite windows, dyadic observations, diagonal readbacks, and seal rows componentwise along hsame and the displayed Cont routes.

Now apply the flattened completion source of Definition 378.42. The source side is already read through BHist, hsame, Cont, Pkg, and local NameCert rows before the $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer reaches a completion-facing coordinate.

Finally attach the flattened non-escape package of Definition 378.44. Since that package exports exactly the finite source, schedule, window, observation, readback, seal, provenance, and naming rows, every bind-domain read has the stated finite coverage and no escaping coordinate is available. □

Theorem 378.46 (CauchyCompletionMonad finite bind-domain induction)**.** *For any finite unary list of accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *bind-domain packets generated from the unit packet and the binary bind operation, every iterated bind-domain read factors through the same finite source-family rows, unary schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, public* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*, displayed* Cont *routes,* Pkg*, and local* NameCert *rows. The induced route exports no quotient stream, selected limit, host real equality, hidden representative, or ambient completeness coordinate.*

Proof. The base case is the unit packet. Apply the unit boundary of Definition 378.3 to identify the constant source-family row, stationary schedule, stationary finite windows, constant dyadic observations, diagonal readback, and public $\mathsf{Real}^{\uparrow}$ seal. Then apply finite bind-domain coverage from Definition 378.45 to place this unit read inside the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

For the binary step, assume the finite route has already been exposed for the current unary list. Apply the schedule composition boundary of Definition 378.6 to put the next bind request into the displayed schedule before it can inspect a window, dyadic observation, diagonal readback, or seal coordinate.

Use diagonal-bind stability from Definition 378.4 to transport the source-family rows, schedules, windows, observations, readbacks, and seals componentwise along hsame and the displayed Cont routes. The induction hypothesis supplies the same finite rows for the incoming packet, so the composed packet remains inside the finite bind-domain surface.

Finally apply the flattened completion source of Definition 378.42 and the flattened non-escape package of Definition 378.44. They repack the composed route through the same source-family, schedule, window, observation, readback, seal, provenance, and naming rows, and they provide no coordinate for quotient streams, selected limits, host real equality, hidden representatives, or ambient completeness.

The resulting finite induction witness is the source used by the Real seal functoriality, completion-consumer naturality, and Kleisli source and seal theorems below, so finite completion consumers read only the displayed bind-domain rows. □

Theorem 378.47 (CauchyCompletionMonad Real seal functoriality)**.** *Componentwise* hsame *transport of the source-family, schedule, finite-window, dyadic-observation, diagonal-readback,* Cont*,* Pkg*, and local* NameCert *rows commutes with passage to the public* Real$^{\uparrow}$ *seal of an accepted* CauchyCompletionMonad$^{\uparrow}$ *packet. Thus two transported bind-domain presentations reach the same public seal route whenever their displayed rows agree componentwise.*

Proof. Apply Real seal naturality from Definition 378.31. It states that componentwise hsame transport preserves the move from scheduled dyadic observations to the diagonal readback and then to the displayed seal row.

Use the flattened seal route of Definition 378.43 to place this naturality statement inside the bind-domain route. The route reads only source-family rows, schedules, finite windows, dyadic observations, diagonal readbacks, displayed transports, package provenance, and local naming before it reaches the seal.

Finish with public seal factorization from Definition 378.5. Since every public Real$^{\uparrow}$ read factors through the displayed diagonal readback and seal row, the transported bind-domain rows and the public seal passage commute. □

Theorem 378.48 (CauchyCompletionMonad completion-consumer naturality)**.** *Every downstream* RegSeqRat$^{\uparrow}$*,* RealCauchyCompletion$^{\uparrow}$*, or* Real$^{\uparrow}$ *consumer of an accepted* CauchyCompletionMonad$^{\uparrow}$ *packet receives the same finite route after componentwise transport and bind composition: source-family rows, unary schedule, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations, diagonal* RegSeqRat$^{\uparrow}$ *readback, the public* Real$^{\uparrow}$ *seal,* BHist*,* hsame*, displayed* Cont *routes,* Pkg*, and local* NameCert *rows.*

Proof. First apply finite bind-domain coverage from Definition 378.45. It supplies one finite route for every bind-domain read and excludes quotient streams, selected limits, host real equality, hidden representatives, and ambient completeness coordinates.

Next apply Real seal functoriality from Definition 378.47. It shows that componentwise transport of the displayed bind-domain rows commutes with the public Real$^{\uparrow}$ seal passage.

The downstream completion consumer exactness is the consumer side of Definition 378.25 together with the root consumer exhaustion of Definition 378.35. These results place RegSeqRat$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, and Real$^{\uparrow}$ consumers on the same source, schedule, window, dyadic-observation, readback, and seal route.

Finally use the flattened non-escape package of Definition 378.44. Since the transported bind composition and each downstream consumer read the same displayed rows and no external coordinate, the finite route received by each consumer is natural in the accepted packet. □

Theorem 378.49 (CauchyCompletionMonad L10 exit finite-route correspondence)**.** *For every accepted* CauchyCompletionMonad$^{\uparrow}$ *packet and every downstream* RegSeqRat$^{\uparrow}$*,* RealCauchyCompletion$^{\uparrow}$*, or* Real$^{\uparrow}$ *completion consumer, the consumer reaches the packet exactly when its visible read factors through the ordered L10 four-face packet consisting of the* DyadicRatCore$^{\uparrow}$ *endpoint and radius ledger, the* StreamName$^{\uparrow}$ *finite request window, the* RegSeqRat$^{\uparrow}$ *classifier and readback rows, and the* Real$^{\uparrow}$ *seal row. This factorization is the same as the completion-consumer naturality route: it carries the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *coordinates, and it exports no quotient stream, selected limit, host real equality, hidden representative, or ambient completeness coordinate.*

Proof. The forward direction starts with completion-consumer naturality from Definition 378.48. That theorem places every RegSeqRat$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, and Real$^{\uparrow}$ consumer on the finite

source-family, schedule, finite-window, dyadic-observation, diagonal-readback, public-seal, BHist, hsame, Cont, Pkg, and local NameCert route of the accepted packet.

The four faces identify this route with the L10 exit packet. Apply Definition 291.28 for the dyadic endpoint and radius ledger, Definition 126.504 for the finite request window, and Definition 301.69 for the classifier and readback rows. Then use Definition 450.13 and Definition 450.16 to seal the same finite diagonal row as the public $\mathsf{Real}^{\uparrow}$ completion route.

For the reverse direction, assume the visible consumer read factors through that ordered four-face packet with the displayed BHist, hsame, Cont, Pkg, and local NameCert coordinates. The handoff exactness theorems identify the dyadic ledger, request window, classifier, readback, and seal rows with the rows used by the completion-consumer naturality route.

Repack those rows through Definition 378.48. Since the factorization contains exactly the finite coordinates admitted by the accepted $\mathsf{CauchyCompletionMonad}^{\uparrow}$ packet and no escaping completion coordinate, the downstream consumer reaches the packet by the same finite route. □

Theorem 378.50 (CauchyCompletionMonad Kleisli source triangle)**.** *For every accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet, the route that first reads a unit packet, then reads a bind packet, and then passes to a downstream* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *completion consumer has the same displayed source triangle as the flattened consumer route. The source-family row, unary schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, public* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*, displayed* Cont *route,* Pkg*, and local* NameCert *row are read in the same finite order, with no additional completion source inserted between bind and consumer.*

Proof. Start with the unit boundary and the left-unit law, Definition 378.3 and Definition 378.14. They identify the unit packet with the displayed constant source-family, schedule, dyadic-observation, diagonal-readback, and seal rows.

Next attach the bind side using finite bind-domain coverage from Definition 378.45. The bind packet can read only the source-family, schedule, finite-window, dyadic-observation, readback, seal, BHist, hsame, Cont, Pkg, and local NameCert rows.

Now compare the resulting route with the flattened completion source of Definition 378.42. It presents the same rows before any completion consumer is allowed to inspect a seal-facing coordinate.

Finally apply completion-consumer naturality from Definition 378.48. Since the downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers receive exactly the displayed finite route, the unit-bind-consumer path and the flattened consumer path have the same source triangle. □

Theorem 378.51 (CauchyCompletionMonad Kleisli seal coherence)**.** *In every accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet, the left-unit route, the right-unit route, and the associative-bind route reach the same public* $\mathsf{Real}^{\uparrow}$ *seal route after componentwise* hsame *transport of the source-family, schedule, finite-window, dyadic-observation, diagonal-readback,* Cont*,* Pkg*, and local* NameCert *rows. Thus the Kleisli consumer boundary has one seal-facing route for unit, bind, and reassociated bind reads.*

Proof. First read the unit laws. Apply Definition 378.14 for the source-side unit route and Definition 378.15 for the consumer-side unit route.

Next read reassociation through Definition 378.16. It keeps the bind route on the same source families, schedules, finite windows, dyadic observations, diagonal readbacks, and seal rows after componentwise hsame transport.

The common seal route is supplied by the flattened seal route of Definition 378.43. Real seal functoriality from Definition 378.47 then transports this route through the displayed hsame, Cont, Pkg, and local NameCert rows.

Finish with completion-consumer naturality from Definition 378.48. Each downstream completion consumer reads the same transported finite route, so left-unit, right-unit, and associative-bind presentations reach one public $\mathsf{Real}^{\uparrow}$ seal route. □

Theorem 378.52 (CauchyCompletionMonad Kleisli non-escape)**.** *The assembled Kleisli consumer route for an accepted* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *packet exports no quotient stream, selected limit, host real equality, hidden representative, or ambient completeness coordinate. Its only exported completion-facing data are the displayed source-family row, unary schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, public* $\mathsf{Real}^{\uparrow}$ *seal route,* BHist*,* hsame*, displayed* Cont *route,* Pkg*, and local* NameCert *row.*

Proof. Begin with the Kleisli source triangle from Definition 378.50. It ties the unit packet, bind packet, and downstream completion consumer to the same displayed finite source route.

Use Kleisli seal coherence from Definition 378.51 to identify the seal-facing side. The left-unit, right-unit, and associative-bind presentations all reach the same public $\mathsf{Real}^{\uparrow}$ seal route under componentwise transport.

Finally apply the flattened non-escape package of Definition 378.44. That package excludes quotient streams, selected limits, host real equality, hidden representatives, and ambient completeness coordinates from the flattened completion route, and the Kleisli consumer route has just been identified with that route on both source and seal sides. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonad}^{\uparrow}$ packet is generated from source regular-Cauchy family rows over $\mathsf{RegSeqRat}^{\uparrow}$, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observations, a unary schedule, diagonal transport rows, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public $\mathsf{Real}^{\uparrow}$ seal factorization, schedule composition boundary, and downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer route supply the bridge-facing completion monad interface over the finite schedule, window, dyadic-observation, readback, seal, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionMonad}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchyCompletionMonadUp.CauchyCompletionMonadPacket\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient real construction, host real equality, uniqueness of limits, or choice-selected limits.

Upgrade path: No further formal-axis upgrade remains for the bridge-checked completion-monad surface.

Chapter 379

A Concrete Naming Certificate for $\mathsf{DyadicTailBound}^{\uparrow}$

The $\mathsf{DyadicTailBound}^{\uparrow}$ interface records the finite tail-budget certificate used by Bishop–Bridges constructive analysis when a real-facing construction presents explicit rational tail control. It sits between $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$: the packet names the displayed precision row, the modulus row that selects the inspected tail, the dyadic tolerance used for the comparison, the finite ledger of tail estimates, and the seal-facing readback route. It does not form quotient streams, select limits, identify host real equality, or assume ambient completeness.

The tail-budget packet consumes the CauchyModulus schedule of chapter 292, the dyadic tolerance ledger of chapter 291, and the regular-rational handoff of chapter 301. These sibling dependencies explain how later regular-Cauchy and $\mathsf{Real}^{\uparrow}$ packets read the same finite threshold, tolerance, and ledger rows without adding a quotient stream or an ambient completeness input.

Definition 379.1 (DyadicTailBound BHist carrier)**.** *A* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

with the following visible rows:

1. *p is a unary precision row naming the displayed request;*
2. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row that assigns the finite tail threshold used for p;*
3. *ϵ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose dyadic value is the comparison budget attached to p;*
4. *L is the finite tail comparison ledger: every inspected tail entry is paired with the threshold supplied by μ, the tolerance supplied by ϵ, and the displayed rational comparison row;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route obtained by reading the same finite tail ledger as regular rational sequence evidence;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached through R and through no other real-facing coordinate;*
7. *H records the componentwise* hsame *transports for precision, schedule, tolerance, tail-ledger, readback, and seal rows;*
8. *C records the displayed* Cont *routes by which consumers reach the schedule, tolerance, ledger, regular-sequence readback, and real seal;*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the rational source rows used by the ledger;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *provenance row.*

The classifier compares two accepted carriers only by these displayed rows, their componentwise hsame *transports, their* Cont *read routes, their* Pkg *provenance, and their local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundTasteGate single carrier alignment`

Theorem 379.2 (DyadicTailBound NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier habitation by the zero-tail packet, where the precision row is arbitrary, the schedule threshold is the displayed zero threshold, the dyadic tolerance is the zero-compatible* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget, the tail ledger records only zero comparisons, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is the constant zero row, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing row is the corresponding zero seal;*

2. *classifier transport under componentwise* hsame *and displayed* Cont *routes for the precision,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, finite tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* Pkg*, and local* NameCert *rows;*

3. *dyadic tolerance exactness: every accepted tail comparison uses exactly the tolerance row ϵ displayed in the carrier and cannot substitute an unrecorded rational radius;*

4. *regular-sequence handoff: the finite ledger L repacks as the visible* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R without dropping the schedule or tolerance rows that justify the tail budget;*

5. $\mathsf{Real}^{\uparrow}$ *seal non-escape: the seal-facing row E reads only through R, L, μ, ϵ, H, C, P, and N, and exports no quotient stream, selected limit, host real equality, or ambient completeness object.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier namecert obligations`

Proof. Carrier habitation is witnessed by the zero-tail packet. The schedule row supplies the zero threshold, the dyadic tolerance row is the displayed zero-compatible budget, and every ledger entry is the visible zero comparison. Repacking those rows gives the constant zero $\mathsf{RegSeqRat}^{\uparrow}$ readback and the matching $\mathsf{Real}^{\uparrow}$ seal-facing row.

Classifier transport is componentwise. Transport the precision request, schedule threshold, dyadic tolerance, ledger entry, readback row, and seal row along the stored hsame comparisons, then replay the recorded Cont routes. Since the classifier reads only the listed packet coordinates, the transported carrier is accepted by the same finite shape.

Dyadic tolerance exactness follows by projecting the ledger coordinate L. Each accepted comparison names the tolerance coordinate ϵ of the same carrier, so a consumer cannot replace it with an unrecorded rational radius or an ambient metric estimate.

The regular-sequence handoff is the projection to R. It keeps the finite ledger together with the schedule and tolerance evidence needed by $\mathsf{RegSeqRat}^{\uparrow}$, so no tail choice or quotient of streams is introduced during readback.

Finally the $\mathsf{Real}^{\uparrow}$ seal-facing row is reached only after the regular-sequence handoff. The carrier has no coordinate for selected limits, host real equality, quotient streams, or completeness principles, and therefore none of those objects can be exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier semantic name certificate`

Theorem 379.3 (DyadicTailBound zero packet carrier)**.** *For every displayed precision row p, the zero-tail packet*

$$T_0 = (p, \mu_0, \epsilon_0, L_0, R_0, E_0, H_0, C_0, P_0, N_0)$$

inhabits the $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier of Definition 379.1. The row μ_0 is the displayed schedule threshold for the zero request, ϵ_0 is the zero-compatible* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, L_0 is the finite ledger of zero tail comparisons at that threshold and tolerance, R_0 is the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback, E_0 is the* $\mathsf{Real}^{\uparrow}$ *zero seal reached through R_0, and H_0, C_0, P_0, N_0 are the matching* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *rows.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier zero packet carrier`

Proof. Build the packet by the displayed carrier coordinates in Definition 379.1. The precision coordinate is p, and the schedule, tolerance, ledger, readback, and seal coordinates are the named zero rows.

The ledger L_0 contains only zero tail comparisons, each paired with the stored threshold μ_0 and tolerance ϵ_0. Thus the packet satisfies the same finite ledger condition used by the classifier.

Apply Definition 379.7 to read the ledger as the constant zero $\mathsf{RegSeqRat}^{\uparrow}$ row R_0, and apply Definition 379.8 to route the $\mathsf{Real}^{\uparrow}$ seal through that readback. The transport, continuation, package, and local certificate rows are all displayed coordinates of the same packet, so no extra real-facing object is introduced. □

Theorem 379.4 (DyadicTailBound precision window restriction)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier. Every consumer read of the precision row p factors through the stored schedule threshold μ, the displayed dyadic tolerance row ϵ, the finite ledger L, and the displayed* Cont *routes in C. In particular, no consumer precision read may bypass the carrier by using an unrecorded threshold, a separate tolerance, or a ledger outside L.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier precision window restriction`

Proof. Project the packet through Definition 379.1. The precision row is stored in the same finite carrier as the schedule threshold, tolerance, ledger, readback, seal, transport, continuation, package, and naming rows.

Use Definition 379.6 to identify accepted consumer transport with componentwise hsame movement and displayed Cont reads. The classifier has no coordinate for an external threshold or tolerance.

The readback and real-facing routes are fixed by Definition 379.7 and Definition 379.8. Hence a precision consumer reaches the schedule, tolerance, ledger, and seal only through the stored packet routes. □

Theorem 379.5 (DyadicTailBound ledger append stability)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier. If a displayed tail-comparison row q is appended to L with the same schedule threshold μ, the same dyadic tolerance row ϵ, and the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *provenance rows, then the enlarged packet is accepted by the same carrier/classifier surface and its* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff still reads through the enlarged displayed ledger.*

Proof. Start from the carrier coordinates in Definition 379.1. The appended row q is not a new kind of evidence: it is another displayed tail comparison paired with the same threshold μ and tolerance ϵ already used by L.

Apply Definition 379.6. Because the schedule, tolerance, transport, continuation, package, and local certificate rows are preserved, classifier acceptance remains the same finite carrier condition with the enlarged ledger coordinate.

Apply Definition 379.7 to repack the enlarged ledger as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and then Definition 379.8 for the seal route. The handoff still factors through the displayed ledger, so the append operation does not create an unrecorded tail estimate or a separate real-facing route. □

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier ledger append stability`

Theorem 379.6 (DyadicTailBound classifier transport exactness)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicTailBound}^{\uparrow}$ BHist *carrier. If another displayed packet* T' *is obtained from* T *by the componentwise* hsame *transports recorded in* H *and by the displayed* Cont *routes recorded in* C*, while preserving the* Pkg *row* P *and the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *row* N*, then the classifier accepts* T' *exactly when the transported precision row, schedule row* μ*, tolerance row* ϵ*, ledger row* L*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *are the corresponding transported coordinates. In particular, classifier transport preserves accepted carriers and cannot replace the displayed tolerance, ledger, readback, or seal by unrecorded rows.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier classifier transport exactness`

Proof. Project the carrier coordinates of Definition 379.1. The classifier reads only the precision row, the $\mathsf{CauchyModulus}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, the tail ledger, the regular-sequence readback, the real seal, and the carried transport and provenance rows.

Apply the hsame witnesses in H component by component, and use the Cont routes in C only to read the transported displayed rows. The package row P and local $\mathsf{NameCert}$ row N are preserved as provenance for the same finite packet shape.

Since the classifier has no coordinate outside those projections, the transported packet is accepted precisely by the transported carrier rows. This rules out substitution of an ambient rational radius, an unlisted ledger, or a separate real-facing endpoint. □

Theorem 379.7 (DyadicTailBound regular-sequence handoff exactness)**.** *For every accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is the handoff obtained from the same schedule row* μ*, dyadic tolerance row* ϵ*, and finite tail ledger* L*. Any consumer of* R *through the displayed* Cont *routes must therefore retain the schedule, tolerance, and ledger data certified by the* Pkg *row* P *and the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier regseqrat handoff exactness`

Proof. Read the schedule, tolerance, ledger, and readback coordinates from the carrier. The row L records the inspected tail comparisons, and each comparison is indexed by the threshold named by μ and the dyadic budget named by ϵ.

The readback row R is the regular rational view of exactly those finite comparisons. Repacking L as R forgets no schedule or tolerance evidence, because those rows are part of the same displayed BHist packet and are reached by the same Cont routes.

The provenance rows P and N certify the dependency package and the local naming surface. They add no hidden stream quotient or selected limit, so the handoff remains the finite $\mathsf{RegSeqRat}^{\uparrow}$ row determined by μ, ϵ, and L. □

Theorem 379.8 (DyadicTailBound real-seal factorization)**.** *For an accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *seal row E factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the finite tail ledger L, the schedule row μ, and the dyadic tolerance row ϵ. The rows H, C, P, and N certify transport, readback, package provenance, and local naming for that factorization, but they do not supply a quotient of streams, a selected limit, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier real seal factorization`

Proof. Project the seal coordinate E from the carrier. By the carrier definition, the only real-facing route first reaches the $\mathsf{RegSeqRat}^{\uparrow}$ row R obtained from the displayed tail data.

Use Definition 379.7 to identify R with the repacking of L, μ, and ϵ. The transport and continuation rows certify that consumers reach these coordinates through the displayed BHist packet.

No coordinate of the carrier names a quotient stream, a chosen limiting real, host equality of real numbers, or a completeness principle. Therefore the seal row factors only through the finite readback and tail-budget rows listed in the statement. □

Theorem 379.9 (DyadicTailBound precision refinement stability)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier. If a displayed refinement replaces the precision row p by p' through the componentwise* hsame *transport recorded in H and through the displayed* Cont *routes recorded in C, then the same refinement transports the schedule row μ, dyadic tolerance row ϵ, ledger row L,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{Real}^{\uparrow}$ *seal row E, package row P, and local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *row N as one classifier-facing packet.*

The transported packet is again read only through the displayed rows of Definition 379.1; in particular, the refined precision request cannot detach the schedule, tolerance, ledger, readback, seal, package, or naming row from the original finite tail-budget certificate.

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier precision refinement stability`

Proof. Project the carrier of Definition 379.1. The precision row is not an isolated parameter: it is stored in the same finite packet as the schedule, tolerance, ledger, readback, seal, transport, continuation, package, and local certificate rows.

Apply the componentwise hsame witnesses recorded in H to the precision row and then to each dependent row listed in the statement. The displayed Cont routes in C give the only consumer reads of the transported schedule, tolerance, ledger, regular-sequence readback, and real seal.

Now use Definition 379.6 to identify classifier acceptance with those transported coordinates. The regular-sequence and real-facing projections remain the routes described by Definition 379.7 and Definition 379.8.

Thus the refinement moves the whole finite certificate at once. There is no coordinate from which a consumer could keep the refined precision while replacing the tail schedule, tolerance, ledger, readback, seal, package, or local naming row by unrecorded data. □

Theorem 379.10 (DyadicTailBound ledger threshold exhaustion)**.** *For every accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N),$$

each comparison consumed from the ledger L factors through the displayed threshold row μ and the displayed dyadic tolerance row ϵ. The carrier has no additional tail-estimate row, no ambient metric estimate, and no separate rational radius from which a ledger consumer can justify the same comparison.

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier ledger comparison determinacy`

Proof. Read the ledger coordinate L from Definition 379.1. By definition, every inspected tail entry is paired with the threshold supplied by μ and the tolerance supplied by ϵ.

The handoff theorem Definition 379.7 keeps the ledger together with exactly those two rows when it repacks the data as the $\mathsf{RegSeqRat}^{\uparrow}$ readback R. The seal theorem Definition 379.8 then reaches the $\mathsf{Real}^{\uparrow}$ row only through that readback and the same finite ledger.

The remaining rows H, C, P, and N record transport, consumer reads, package provenance, and local naming. They certify access to the displayed rows, but none of them carries an extra estimate.

Therefore any ledger comparison is exhausted by μ, ϵ, and the visible ledger entry. A downstream consumer cannot cite an unrecorded tail estimate without leaving the carrier surface. □

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier ledger threshold exhaustion`

Theorem 379.11 (DyadicTailBound seal consumer determinacy)**.** *Suppose two consumers read an accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

through the same displayed rows R, L, μ, ϵ, H, C, P, and N. Then both consumers obtain the same classifier-facing $\mathsf{Real}^{\uparrow}$ *seal row E. This determinacy is only a statement about the visible seal row of the carrier; it does not export host equality of real numbers, quotient equality of streams, a selected limit, or a completeness datum.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier seal consumer determinacy`

Proof. Project the seal coordinate E from the carrier. The definition gives a single real-facing row attached to the regular-sequence readback route.

By Definition 379.8, every consumer route to E factors through R, L, μ, and ϵ, with H, C, P, and N supplying only transport, readback, provenance, and local naming. The two consumers in the statement therefore traverse the same displayed factorization.

Apply Definition 379.6 to compare the resulting reads. Since the classifier sees only the listed rows, the two reads have the same classifier-facing seal coordinate.

The conclusion does not add a host real carrier or an equality principle. It records determinacy of the finite $\mathsf{Real}^{\uparrow}$ seal row already present in the packet. □

Theorem 379.12 (DyadicTailBound zero-tail carrier habitation)**.** *For every displayed precision row p, the zero-tail packet*

$$T_0 = (p, \mu_0, \epsilon_0, L_0, R_0, E_0, H_0, C_0, P_0, N_0)$$

inhabits the $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier. Here μ_0 is the displayed zero threshold, ϵ_0 is the zero-compatible* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, L_0 is the finite ledger whose accepted comparisons are zero comparisons at that threshold and tolerance, R_0 is the constant zero* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E_0 is the corresponding* $\mathsf{Real}^{\uparrow}$ *zero seal, H_0 and C_0 are the componentwise* hsame *and* Cont *rows for those displayed coordinates, and P_0, N_0 are the matching* Pkg *and local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *provenance rows.*

Proof. Build the packet by the carrier coordinates of Definition 379.1. The precision coordinate is the given row p, while the schedule, tolerance, ledger, readback, and seal coordinates are the displayed zero rows.

The ledger coordinate contains only zero comparisons, so every comparison is already indexed by the zero threshold μ_0 and the zero-compatible dyadic tolerance ϵ_0. Repacking this ledger gives the constant zero $\mathsf{RegSeqRat}^{\uparrow}$ readback R_0.

The $\mathsf{Real}^{\uparrow}$ row E_0 is reached through that readback. The transport, continuation, package, and local naming rows are the trivial componentwise rows for the same displayed packet, so the classifier accepts T_0 without adding quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 379.13 (DyadicTailBound tolerance ledger exhaustion)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier. Every accepted comparison consumed from L is exhausted by the displayed tolerance row ϵ, the displayed threshold row μ, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the componentwise* hsame *rows in H, the* Cont *routes in C, the package row P, and the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$ *row N. No accepted comparison in L can be justified by an unrecorded tolerance, external metric estimate, host real equality, quotient stream, or selected limit.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier tolerance ledger exhaustion`

Proof. Project the ledger coordinate L from the carrier. By Definition 379.1, each inspected tail entry is paired with the same threshold coordinate μ and the same dyadic tolerance coordinate ϵ displayed in the packet.

Apply Definition 379.7 to keep L attached to the readback row R. The hsame rows in H and the Cont routes in C only transport and read these displayed coordinates.

The provenance rows P and N certify the dependency package and local naming surface for this finite packet. Since no other comparison coordinate exists, the ledger is exhausted by μ, ϵ, R, H, C, P, and N. □

Theorem 379.14 (DyadicTailBound CauchyRate consumer route)**.** *For every accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *carrier*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N)$$

every $\mathsf{CauchyRate}^{\uparrow}$ *consumer read of T factors through the precision row p, schedule μ, dyadic tolerance ϵ, ledger L,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E, and displayed rows H, C, P, and N.*

The route exports no hidden rate selector, quotient stream, selected limit, host real equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier cauchy rate consumer route`

Proof. Project the accepted carrier. Its consumer-visible coordinates are exactly the precision row, schedule threshold, tolerance row, finite ledger, regular-sequence readback, real seal, transport rows, continuation routes, package provenance, and local naming row.

Use Definition 379.7. Any $\mathsf{CauchyRate}^{\uparrow}$ read that consumes a regular rational rate must pass through R, and R is obtained from the same schedule row μ, dyadic tolerance ϵ, and finite ledger L.

Apply Definition 379.8. A real-facing consumer reaches E only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback and the same tail-budget coordinates.

Use Definition 379.10 and Definition 379.13. These rows keep every consumed comparison tied to the threshold μ, tolerance ϵ, ledger L, transports H, continuations C, package row P, and local naming row N.

Since the carrier contains no hidden rate selector, quotient stream, selected limit, host equality, or ambient completeness field, every $\mathsf{CauchyRate}^{\uparrow}$ consumer route is exhausted by p, μ, ϵ, L, R, E, H, C, P, and N. □

Theorem 379.15 (DyadicTailBound consumer obligation package)**.** *The carrier of Definition 379.1, classifier transport exactness, zero-tail carrier habitation, tolerance ledger exhaustion,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness, and* $\mathsf{Real}^{\uparrow}$ *seal factorization assemble the* $\mathsf{DyadicTailBound}^{\uparrow}$ *consumer obligation package over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The package exposes only the displayed precision row, schedule threshold, dyadic tolerance, finite ledger, regular-sequence readback, real seal, componentwise transport rows, continuation routes, package provenance, and local naming row. It does not export quotient streams, selected limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier consumer obligation package`

Proof. Start with the finite carrier. It supplies the displayed rows $p, \mu, \epsilon, L, R, E, H, C, P, N$ and therefore fixes the consumer-visible packet shape.

Classifier transport is Definition 379.6. The inhabited base case is Definition 379.12, and the ledger exactness row is Definition 379.13.

The readback and real-facing rows are supplied by Definition 379.7 and Definition 379.8. These theorems force every consumer route through the finite ledger, the displayed tolerance and threshold, and the regular rational readback.

Assembling the listed rows gives the obligation package. Since each component is a projection or repacking of the same BHist packet, no extra carrier for quotient streams, selected limits, host real equality, or ambient completeness is introduced. □

Theorem 379.16 (DyadicTailBound budget composition)**.** *Let*

$$T_1 = (p, \mu, \epsilon, L_1, R_1, E_1, H, C, P, N) \quad \textit{and} \quad T_2 = (p, \mu, \epsilon, L_2, R_2, E_2, H, C, P, N)$$

be accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *packets sharing the displayed precision row, schedule threshold, dyadic tolerance, componentwise* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *row. If the second displayed tail ledger refines or extends the first through those same rows, then the concatenated ledger* L_1L_2 *is an accepted* $\mathsf{DyadicTailBound}^{\uparrow}$ *packet whose* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal factor through the composite displayed budget. The conclusion exports only the composed finite ledger surface and no quotient stream, selected limit, host real equality, or ambient completeness datum.*

Proof. Begin with the two accepted carriers and project their common rows. The precision, threshold, tolerance, transport, continuation, package, and local naming coordinates agree by hypothesis, so the only displayed change is the refinement from L_1 to L_2 and the concatenated ledger L_1L_2.

Apply Definition 379.5 along the displayed refinement rows to accept each appended comparison with the same μ, ϵ, H, C, P, and N. Then use Definition 379.10 to keep every comparison in the composite ledger tied to the displayed threshold and tolerance.

Classifier transport is controlled by Definition 379.6. It prevents the composite packet from replacing the shared provenance by an unrecorded tolerance, external estimate, or separate ledger surface.

Finally apply Definition 379.7 and Definition 379.8. The regular-sequence readback factors through the concatenated ledger, and the real-facing seal factors through that readback and the same composite displayed budget. □

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier budget composition`

Theorem 379.17 (DyadicTailBound schedule monotone consumer)**.** *Let*

$$T = (p, \mu, \epsilon, L, R, E, H, C, P, N) \quad \textit{and} \quad T' = (p', \mu', \epsilon, L', R', E', H, C, P, N)$$

be accepted $\mathsf{DyadicTailBound}^{\uparrow}$ *packets sharing the displayed tolerance, componentwise* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *row. Suppose the displayed refinement from p to p' transports the schedule threshold from μ to μ' through the same rows H and C, and suppose the refined ledger L' is read as the carried tail ledger for μ' and ϵ.*

Then every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of T' factors through the transported schedule row μ', the displayed dyadic tolerance row ϵ, the finite ledger L', and the shared transport, continuation, package, and naming rows. Such a consumer cannot reuse the old ledger through an unrecorded schedule, external radius, selected limit, quotient stream, host real equality, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.DyadicTailBoundUp.DyadicTailBoundCarrier schedule monotone consumer`

Proof. Start with the accepted packet T and apply Definition 379.9. The refinement of p to p' is not a free precision change; it transports the schedule, tolerance, ledger, readback, seal, package, and naming coordinates through the same H and C rows.

Next use Definition 379.10. Every comparison consumed from the refined ledger is exhausted by the transported threshold μ' and the displayed tolerance ϵ, so the ledger cannot be replayed through a schedule row absent from the packet.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, apply Definition 379.7. The readback row R' is obtained only by repacking L', μ', and ϵ along the displayed transport and continuation routes.

For a $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 379.8. The seal row E' is reached only through that regular-sequence readback and the same finite tail-budget data.

Finally use Definition 379.16 to keep composed refinements on the displayed budget surface. Since the carrier has no coordinate for an external radius, selected limit, quotient stream, host real equality, or ambient completeness, every refined consumer route must carry the schedule row together with the ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTailBound}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTailBound}^{\uparrow}$ packet is generated from a unary precision row, a $\mathsf{CauchyModulus}^{\uparrow}$ threshold schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a finite tail comparison ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback route, a $\mathsf{Real}^{\uparrow}$ seal-facing row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicTailBound}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper scope is the finite tail-budget interface through budget composition, ledger threshold exhaustion, the RegSeqRat$^{\uparrow}$ handoff, Real$^{\uparrow}$ seal factorization, classifier transport, consumer obligation packaging, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicTailBound}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, ambient completeness, or bridge checking.

Upgrade path: Machine bridge verification requires a Lean bridge theorem for the same public tail-budget interface and downstream completion schema.

Chapter 380

A Concrete Naming Certificate for $\mathsf{NullSequence}^{\uparrow}$

The $\mathsf{NullSequence}^{\uparrow}$ interface names the constructive zero-convergence packet consumed by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion-facing certificates when a displayed rational difference stream is required to shrink to zero. Following Bishop and Bridges, *Constructive Analysis*, the certificate records explicit finite tolerance schedules and observation windows rather than quotienting streams or appealing to a host limit.

The packet depends on the regular-rational difference stream of chapter 301, the dyadic tolerance rows of chapter 291, and the finite observation windows of chapter 126. A $\mathsf{NullSequence}^{\uparrow}$ certificate is a consumer of those rows before the $\mathsf{Real}^{\uparrow}$ seal can read zero-convergence: it does not manufacture a stream, tolerance ledger, or window surface of its own.

Definition 380.1 (NullSequence NameCert carrier)**.** *A* $\mathsf{NullSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (D, T, W, M, E, H, C, P, N)$$

with the following visible coordinates:

1. *D is a* $\mathsf{RegSeqRat}^{\uparrow}$ *difference stream row whose inspected values are displayed through finite rational readbacks;*
2. *T is a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule naming the positive dyadic bounds requested by a consumer;*
3. *W is a* $\mathsf{StreamName}^{\uparrow}$ *observation window that records exactly which finite portion of D is inspected for each tolerance row;*
4. *M is a* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *bound row selecting the threshold after which the displayed difference observations are below the tolerance in T;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-consumption row whose equality classifier may consume the zero-convergence packet;*
6. *H records the componentwise* hsame *rows comparing difference-stream observations, tolerance rows, windows, modulus rows, and seal rows;*
7. *C records the* Cont *routes through which downstream readers reach the finite stream, tolerance, bound, and seal rows;*

8. *P is the* Pkg *provenance ledger naming exactly the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sources;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *provenance row.*

The classifier for two accepted packets is equality of displayed zero-convergence packets under hsame *transport of the stream window, tolerance schedule, bound row, seal row, continuation routes, package ledger, and local naming row.*

Lean4 ✓ `BEDC.Derived.NullSequenceUp.NullSequenceTasteGate single carrier alignment`

Theorem 380.2 (NullSequence NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *surface consists exactly of five obligations: carrier habitation by the displayed packet of Definition 380.1; classifier reflexivity, symmetry, and transitivity inherited componentwise from* hsame *over windows, tolerances, bounds, and seal rows; carrier transport along the classifier; exact ledger exposure of only the finite stream, modulus, tolerance, seal, continuation, package, and local certificate rows; and* $\mathsf{Real}^{\uparrow}$ *classifier consumption of the null difference packet.*

Proof. Carrier habitation is obtained by displaying the zero difference stream, a dyadic tolerance row, a finite observation window, a bound row that sends the tolerance request to that window, the real-seal consumption row, and the provenance rows.

The classifier laws are inherited componentwise. Reflexivity uses the identity hsame rows for the displayed stream, tolerance, window, bound, and seal coordinates. Symmetry reverses those rows, and transitivity composes them without introducing any host sequence equality.

Transport repacks the same finite coordinates after applying the displayed hsame and Cont routes. Since the provenance ledger lists only the rows in Definition 380.1, every public read is exhausted by those rows and by the $\mathsf{Real}^{\uparrow}$ consumption boundary. □

Lean4 ✓ `BEDC.Derived.NullSequenceUp.NullSequenceCarrier namecert obligation surface`

Theorem 380.3 (NullSequence Real classifier consumption)**.** *For two* $\mathsf{Real}^{\uparrow}$ *seal rows presented by regular rational names, a* $\mathsf{NullSequence}^{\uparrow}$ *packet for their displayed difference stream is precisely the finite witness that the* $\mathsf{Real}^{\uparrow}$ *equality classifier may consume: each requested dyadic tolerance is routed through a* $\mathsf{StreamName}^{\uparrow}$ *window and a* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *bound, and the resulting tail estimate is exposed to the seal row through the recorded* Cont *and* Pkg *provenance.*

Proof. Read the two real-name presentations through their $\mathsf{RegSeqRat}^{\uparrow}$ rows and form the displayed difference stream coordinate D. A consumer tolerance is then read through the dyadic schedule T.

The modulus coordinate M chooses the finite window W where the relevant tail observations of D are inspected. The packet records that those observations are bounded by the tolerance row, so the real-seal row receives an explicit zero-convergence witness rather than a quotient comparison.

Conversely, a $\mathsf{Real}^{\uparrow}$ equality read that is certified by this packet can project only the difference stream, tolerance schedule, window, bound, seal, continuation, and provenance rows. Those are exactly the coordinates of Definition 380.1, so the classifier consumption is neither stronger nor weaker than the displayed null-difference packet. □

Theorem 380.4 (NullSequence downstream Real-completion consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *equality or completion-facing seal that consumes a* $\mathsf{NullSequence}^{\uparrow}$ *null-difference packet must route through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference stream, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *observation window, and the* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *threshold before reading the* $\mathsf{Real}^{\uparrow}$ *seal. In particular, any downstream read shared with the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *budget route of chapter 699 or a completion-facing handoff such as chapter 1628 receives the null-difference evidence only through those displayed finite rows.*

Proof. Start with the Real classifier consumption theorem Definition 380.3. It identifies the packet consumed by a $\mathsf{Real}^{\uparrow}$ equality read as the displayed null-difference row together with its tolerance, window, threshold, seal, continuation, and provenance rows.

Apply the carrier projection of Definition 380.1. The difference stream is a $\mathsf{RegSeqRat}^{\uparrow}$ row, the tolerance request is a $\mathsf{DyadicRatCore}^{\uparrow}$ row, the observation service is a $\mathsf{StreamName}^{\uparrow}$ row, and the threshold is supplied by $\mathsf{CauchyModulus}^{\uparrow}$ or $\mathsf{RegularityModulus}^{\uparrow}$ before the seal is public.

The cited downstream siblings consume the same finite route rather than a hidden host limit. Since no coordinate of the packet exports a quotient stream, selected limit, trichotomy, or ambient completeness proof, every Real or completion-facing seal read remains downstream of the displayed null-sequence rows. □

Definition 380.5 (NullSequence tolerance-window exactness package)**.** *For an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet*

$$Z = (D, T, W, M, E, H, C, P, N),$$

the tolerance-window exactness package is the same finite tuple, read with the following obligation-level roles:

1. *D is the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row whose zero-convergence is being consumed;*
2. *T is the accepted dyadic tolerance request row;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window selected for that tolerance request;*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *threshold row binding the request in T to the window in W;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that reads the resulting null-difference evidence;*
6. *H records the componentwise* hsame *transports for difference rows, tolerance rows, windows, thresholds, and seal rows;*
7. *C records the* Cont *routes from the consumer request to T, from T to W and M, and from the certified difference row to E;*
8. *P and N record the dependency package and local* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *provenance.*

The package is exact when every accepted tolerance request reaches the displayed threshold row M and window W before the $\mathsf{Real}^{\uparrow}$ *seal reads the row E. No coordinate of the package is a host limit, an unrecorded tail estimate, or a quotient of difference streams.*

Theorem 380.6 (NullSequence window threshold exactness)**.** *Let $Z = (D, T, W, M, E, H, C, P, N)$ be an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet equipped with the tolerance-window exactness package of Definition 380.5. Every accepted tolerance request consumed by Z is routed through the displayed tolerance row T, the displayed observation window W, and the displayed bound row M before it reaches any* $\mathsf{Real}^{\uparrow}$ *seal read.*

Thus the zero-convergence evidence exported by Z is exhausted by the $\mathsf{RegSeqRat}^{\uparrow}$ *difference row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *threshold, and the recorded* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance. It contains no unrecorded tail estimate and no host limit.*

Lean4 ✓ `BEDC.Derived.NullSequenceUp.NullSequenceCarrier window threshold exactness`

Proof. Project the carrier coordinates of Z. The tolerance request is the coordinate T, so any consumer read first lands in the displayed dyadic tolerance schedule rather than in an ambient metric statement.

The window coordinate W and threshold coordinate M are then selected by the recorded Cont route. The bound row says that the inspected tail of the displayed difference stream D is below the tolerance named by T on precisely that finite window.

Transport between equivalent presentations is componentwise. The rows in H move the difference row, tolerance row, window, threshold, and seal row, while C preserves the order in which the consumer reaches them.

Finally, P and N certify the same dependency and local naming provenance after transport. Since the only outgoing seal read is the row E, every accepted tolerance request has passed through T, W, and M, and there is no remaining coordinate from which an unrecorded tail estimate or host limit could be exported. □

Theorem 380.7 (NullSequence Real seal factorization)**.** *For two* $\mathsf{Real}^{\uparrow}$ *seal rows presented by regular rational names, any* $\mathsf{Real}^{\uparrow}$ *classifier consumption certified by a* $\mathsf{NullSequence}^{\uparrow}$ *packet factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row D, dyadic tolerance schedule T, observation window W, threshold row M, seal row E, and provenance package (H, C, P, N) of Definition 380.5.*

Equivalently, the $\mathsf{Real}^{\uparrow}$ *equality read cannot bypass the displayed null-difference package by using an unstated stream representative, a host real limit, a hidden tail estimate, trichotomy, or choice.*

Lean4 ✓ `BEDC.Derived.NullSequenceUp.NullSequenceCarrier real seal factorization`

Proof. Start with the $\mathsf{Real}^{\uparrow}$ classifier consumption theorem Definition 380.3. It identifies a $\mathsf{NullSequence}^{\uparrow}$ packet as the finite witness that the real equality classifier may consume for the displayed regular-rational difference row.

Apply Definition 380.6 to that packet. The theorem forces each tolerance read to pass through the tolerance schedule T, the window W, and the threshold M before reaching the seal row E.

The remaining coordinates are provenance rather than extra analytic data. The hsame rows in H, the Cont routes in C, and the package and local naming rows P, N record how the same difference, tolerance, window, threshold, and seal rows are transported.

Therefore the real-seal consumption factors through exactly the displayed package. Any route using a different stream representative, host limit, hidden tail estimate, trichotomy, or choice would require a coordinate absent from the carrier of Definition 380.1. □

Theorem 380.8 (NullSequence non-escape boundary)**.** *An accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet exports no Dedekind cut, no quotient of streams, no host real equality, no trichotomy principle, and no real-completeness theorem. Every public zero-convergence read factors through* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows.*

Proof. Inspect the carrier coordinates. Each coordinate is a finite history row, a displayed transport route, a package provenance row, a local naming row, or one of the named dependency certificates.

The bound row names only a finite observation threshold for a requested tolerance. Passing to the $\mathsf{Real}^{\uparrow}$ seal consumes that finite zero-convergence row; it does not create a host real carrier, a completed quotient class, or a comparison by trichotomy.

Thus a downstream reader can cite exactly the listed finite rows. Since no excluded analytic or order-theoretic object is a coordinate of the carrier, none can be exported by the certificate. □

Lean4 ✓ `BEDC.Derived.NullSequenceUp.NullSequenceCarrier non escape boundary`

Theorem 380.9 (NullSequence standard Bishop reading)**.** *In the Bishop–Bridges constructive-analysis reading,* $\mathsf{NullSequence}^{\uparrow}$ *is the finite certificate that a regular rational difference stream converges to zero: tolerances are explicit dyadic rows, observation windows are explicit stream-name rows, and the bound is an explicit Cauchy or regularity modulus consumed by* $\mathsf{Real}^{\uparrow}$ *equality.*

Proof. The input side is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ difference stream together with the dyadic tolerance schedule. No convergence claim is inferred from a host metric space or from an ambient completeness theorem.

The witnessing side is the finite window and bound row. For each tolerance request, the packet records the window after which the displayed difference observations are below the requested dyadic bound.

The output side is the $\mathsf{Real}^{\uparrow}$ seal-consumption row. It reads the finite zero-convergence ledger as equality evidence for regular rational names while withholding Dedekind cuts, quotient streams, host real equality, trichotomy, and real completeness. □

Theorem 380.10 (NullSequence zero-tail criterion)**.** *Let* $Z = (D, T, W, M, E, H, C, P, N)$ *be an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet over a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference stream. A* $\mathsf{Real}^{\uparrow}$ *equality-classifier consumer read is admitted by* Z *exactly when the read factors through the tolerance schedule* T*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound row read by* M*, the* Cont *routes in* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *row recorded by the tolerance-window package of Definition 380.5.*

Thus the classifier-facing zero-tail read for regular rational names is the same finite tolerance-window packet, transported through H *and sealed by* E*, and not an additional real-limit datum.*

Proof. First project the accepted tolerance request. By Definition 380.6, the request must pass through the displayed dyadic schedule T, the observation window W, and the threshold row M before it reaches any $\mathsf{Real}^{\uparrow}$ seal row.

Next apply Definition 380.7. The real equality classifier can consume the zero-tail evidence only after that same stream, tolerance, window, threshold, seal, transport, continuation, package, and local certificate package has been read.

For the converse direction, suppose the consumer read already factors through these displayed rows. The tolerance-window package supplies the bound row and window for the requested tolerance, and the factorization theorem routes the resulting null-difference evidence into the seal row E.

Finally use Definition 380.8. Since no host limit, quotient stream, hidden tail estimate, trichotomy, or completeness object is exported by the packet, an admitted $\mathsf{Real}^{\uparrow}$ zero-tail read is exactly the finite route just described. □

Theorem 380.11 (NullSequence classifier-composition window exactness)**.** *Let* $Z_0 = (D_0, T_0, W_0, M_0, E_0, H_0, C_0, P$ $Z_1 = (D_1, T_1, W_1, M_1, E_1, H_1, C_1, P_1, N_1)$*, and* $Z_2 = (D_2, T_2, W_2, M_2, E_2, H_2, C_2, P_2, N_2)$ *be accepted* $\mathsf{NullSequence}^{\uparrow}$ *packets. If the classifier transports* Z_0 *to* Z_1 *and* Z_1 *to* Z_2*, then the composed transport preserves the same displayed difference row* D*, tolerance row* T*, observation window* W*, threshold row* M*, seal row* E*, and provenance package* (H, C, P, N)*.*

Thus classifier composition remains a finite tolerance-window exactness claim over the rows of Definition 380.5; it does not introduce a host limit, a hidden tail estimate, a quotient stream, trichotomy, or choice.

Proof. Project the carrier of Definition 380.1. The classifier compares only the displayed difference, tolerance, window, threshold, seal, transport, continuation, package, and local certificate rows.

Apply the tolerance-window exactness package of Definition 380.5 to the first transport and then to the second. Each step keeps the tolerance request on T, the observation read on W, and the threshold read on M before the seal row is reached.

Use Definition 380.6 to identify the composed route. Since both component transports preserve exactly the same displayed rows, their composition still reads the original finite difference row through the transported T, W, and M coordinates.

Finally Definition 380.8 rules out any additional analytic coordinate. The composed classifier route therefore preserves the same tolerance-window packet and the same provenance package (H, C, P, N). □

Theorem 380.12 (NullSequence tolerance-window transport)**.** *For an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet* $Z = (D, T, W, M, E, H, C, P, N)$*, every transported tolerance request still reaches the displayed tolerance row* T*, observation window* W*, and threshold row* M *before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is read. The transport is carried only by the componentwise* hsame *ledger* H*, the displayed* Cont *routes* C*, the package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *row* N*.*

Consequently the transported request remains the same finite null-difference package consumed by $\mathsf{Real}^{\uparrow}$ *equality, not an unstated stream representative or host-limit assertion.*

Proof. Begin with the window-threshold exactness theorem Definition 380.6. It forces each accepted tolerance request to pass through T, W, and M before any seal read.

Transport the rows using the ledger H and replay the routes in C. Because the tolerance-window package lists P and N only as provenance rows, the transport can move the finite packet but cannot add a new bound, stream representative, or real-limit datum.

The seal factorization theorem Definition 380.7 then routes the transported null-difference evidence into E. The non-escape boundary of Definition 380.8 confirms that no quotient, trichotomy, choice, or host equality object is exported by this route. □

Theorem 380.13 (NullSequence common-window refinement)**.** *Let* $Z = (D, T, W, M, E, H, C, P, N)$ *and* $Z' = (D', T', W', M', E', H', C', P', N')$ *be accepted* $\mathsf{NullSequence}^{\uparrow}$ *packets whose* $\mathsf{RegSeqRat}^{\uparrow}$ *difference streams are classifier-related through the displayed* BHist *rows. For any pair of* $\mathsf{Real}^{\uparrow}$ *equality consumer reads of* Z *and* Z'*, if the reads name tolerance schedules in* $\mathsf{DyadicRatCore}^{\uparrow}$*, observation windows in* $\mathsf{StreamName}^{\uparrow}$*, threshold rows in* M, M'*, seal rows in* E, E'*, an* hsame *ledger* H, H'*,* Cont *routes* C, C'*,* Pkg *provenance* P, P'*, and local* $\mathsf{NameCert}_{\mathsf{NullSequence}^{\uparrow}}$ *rows* N, N'*, then the pair refines to one displayed* $\mathsf{StreamName}^{\uparrow}$ *common window and one displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row before either* $\mathsf{Real}^{\uparrow}$ *seal is read.*

The refined surface is carried only by the shown BHist *packet rows, the* hsame *transports, the* Cont *routes, the* Pkg *rows, and the local* $\mathsf{NameCert}$ *rows. It exports no unrecorded stream representative, host limit, quotient stream, trichotomy, or real-completeness object.*

Proof. First apply Definition 380.6 to both accepted packets. Each $\mathsf{Real}^{\uparrow}$ equality read must pass through its displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ observation window, and threshold row before any seal row is reached.

Next use Definition 380.11. The classifier-related difference streams and their componentwise transports compose to a single finite window comparison rather than to a second analytic tail claim.

Apply Definition 380.12 to move the two tolerance-window packages through the hsame ledgers H, H' and the Cont routes C, C'. The transported consumer reads therefore share one displayed window row and one displayed tolerance row before the seal coordinates are consulted.

Finally use Definition 380.7. The $\mathsf{Real}^{\uparrow}$ seals read only the factorized null-difference package, and the non-escape boundary of Definition 380.8 excludes any hidden limit, quotient stream, trichotomy, or completeness object from the refinement. □

Theorem 380.14 (NullSequence tail-induction modulus)**.** *From an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet and two displayed tolerance windows, primitive induction over the finite window-refinement ledger produces a common tail modulus that preserves the* $\mathsf{Real}^{\uparrow}$ *classifier consumption and never introduces quotient streams, host real equality, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tolerance-window exactness package of Definition 380.5. The two displayed tolerance requests are already routed through finite T, W, M rows, with the seal row E and provenance rows H, C, P, N attached to the same packet.

Apply window threshold exactness from Definition 380.6 to each request. This gives two finite thresholds and two finite observation windows before either $\mathsf{Real}^{\uparrow}$ seal can be read.

Run primitive induction over the finite refinement ledger supplied by Definition 380.13. The base step keeps the first displayed window, and each refinement step enlarges the displayed window only by the next finite row needed to cover both tolerance requests.

Finally transport the resulting common window by Definition 380.12. The seal still consumes the same factorized null-difference package, so the induction produces a common tail modulus without adding quotient streams, host real equality, trichotomy, or any completeness field. □

Theorem 380.15 (NullSequence obligation closure package)**.** *The* $\mathsf{NullSequence}^{\uparrow}$ *chapter presents an obligation-complete paper package for zero-tail consumption by* $\mathsf{Real}^{\uparrow}$ *equality over regular rational names. The package consists of the carrier of Definition 380.1, the componentwise classifier route of Definition 380.2, the tolerance-window exactness package of Definition 380.5, window-threshold exactness from Definition 380.6, classifier-composition exactness from Definition 380.11, tolerance-window transport from Definition 380.12,* $\mathsf{Real}^{\uparrow}$ *seal factorization from Definition 380.7, the zero-tail criterion of Definition 380.10, and the non-escape boundary of Definition 380.8.*

This package closes the paper obligation surface for finite tolerance schedules, classifier stability, transported observation windows, threshold exactness, and $\mathsf{Real}^{\uparrow}$ *null-difference consumption. It does not claim Dedekind cuts, quotient streams, host real equality, trichotomy, choice, or real completeness.*

Proof. The carrier and NameCert obligations identify the finite null-difference packet and the componentwise classifier rows. The tolerance-window package names the roles of D, T, W, M, E, H, C, P, and N.

Window-threshold exactness and tolerance-window transport show that every tolerance request reaches T, W, and M before the seal. Classifier-composition exactness shows that composing accepted transports preserves the same finite rows rather than producing a second tolerance ledger.

The seal factorization theorem and the zero-tail criterion identify the $\mathsf{Real}^{\uparrow}$ equality consumer as precisely the reader of this displayed null-difference packet. The non-escape boundary supplies the excluded-object clause, so the assembled package is obligation-complete on the paper axis while leaving formal verification to a separate target. □

Definition 380.16 (NullSequence root-unblock consumer surface)**.** *For an accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet*

$$Z = (D, T, W, M, E, H, C, P, N),$$

the downstream consumer surface consists of tolerance schedule exposure, $\mathsf{StreamName}^{\uparrow}$ *window routing, bound-row thresholding,* $\mathsf{Real}^{\uparrow}$ *seal consumption, and non-escape. Every consumer request*

must reach the displayed tolerance row T, observation window W, threshold row M, seal row E, transport rows H, continuation routes C, package row P, and local naming row N before it can be used by a later $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *equality argument.*

Theorem 380.17 (NullSequence root tolerance schedule exposure)**.** *For every accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet $Z = (D, T, W, M, E, H, C, P, N)$, each downstream consumer tolerance request reaches the concrete rows T, W, M of Definition 380.16.*

Proof. Apply Definition 380.6. It identifies the route from a requested dyadic tolerance to the displayed observation window and threshold row.

Then apply Definition 380.12. The same rows remain visible after transport through H and readback through C, with P and N carrying package and local naming provenance.

Thus every downstream tolerance request is exposed as a finite T, W, M request before any seal consumption occurs. □

Theorem 380.18 (NullSequence root real-equality handoff)**.** *For every accepted* $\mathsf{NullSequence}^{\uparrow}$ *packet $Z = (D, T, W, M, E, H, C, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *equality handoff reads the displayed null-difference packet through the seal row E and the finite tolerance-window rows D, T, W, M, H, C, P, N.*

Proof. Use Definition 380.7. It factors every real equality read through the null-difference packet and its displayed tolerance-window rows.

Use Definition 380.10. It identifies the accepted zero-tail evidence as the finite tolerance schedule and threshold route rather than as a host limit or quotient stream.

Therefore the real-equality handoff is constructive and finite-row based over $D, T, W, M, E, H, C, P, N$. □

Theorem 380.19 (NullSequence root downstream unblock package)**.** *The consumer surface of Definition 380.16, the tolerance exposure of Definition 380.17, the real-equality handoff of Definition 380.18, the non-escape boundary of Definition 380.8, and the obligation package of Definition 380.15 assemble the downstream* $\mathsf{NullSequence}^{\uparrow}$ *root-unblock package.*

Proof. The consumer surface names the rows $D, T, W, M, E, H, C, P, N$ and the five obligations that later regular-rational and real-equality chapters consume.

The tolerance exposure theorem supplies the T, W, M route for each tolerance request. The real-equality handoff theorem supplies the seal-facing route through E and the same finite tolerance-window packet.

Finally, the non-escape boundary and obligation closure package exclude host limits, quotient streams, trichotomy, choice, and real completeness. The assembled package is therefore a concrete downstream citation surface for finite null-sequence consumption. □

Theorem 380.20 (NullSequence standard bridge package)**.** *The public* $\mathsf{NullSequence}^{\uparrow}$ *packet has a standard bridge whose source rows are exactly the finite* BHist *difference stream, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{CauchyModulus}^{\uparrow}$ *or* $\mathsf{RegularityModulus}^{\uparrow}$ *modulus row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, the* hsame *and* Cont *transport rows, the* Pkg *provenance, and the* $\mathsf{NameCert}$ *row displayed in Definition 380.16.*

The bridge source is therefore the same finite zero-tail packet used by Definition 380.19; it does not add a Dedekind cut, quotient stream, host real equality, trichotomy, or real-completeness field.

Proof. Start from the public consumer surface. The rows $D, T, W, M, E, H, C, P, N$ already display the regular-rational difference stream, dyadic tolerance request, finite observation window, modulus bound, seal row, transport, continuation, package, and local naming data.

Use Definition 380.17 to expose the tolerance and window route, and use Definition 380.18 to expose the real-seal route through the same finite packet.

The source side of the bridge is then the downstream package of Definition 380.19. Its non-escape boundary is Definition 380.8, so the bridge reads exactly the listed rows and has no field for the excluded host or quotient data. □

Closure status for $\mathsf{NameCert}_{\mathsf{NullSequence}^\uparrow}$.

Constructive story: A $\mathsf{NullSequence}^\uparrow$ packet is generated from a $\mathsf{RegSeqRat}^\uparrow$ difference stream, a $\mathsf{DyadicRatCore}^\uparrow$ tolerance schedule, a $\mathsf{StreamName}^\uparrow$ observation window, a $\mathsf{CauchyModulus}^\uparrow$ or $\mathsf{RegularityModulus}^\uparrow$ bound row, a $\mathsf{Real}^\uparrow$ seal-consumption row, componentwise hsame transport, Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NullSequence}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper surface binds the NameCert obligations, tolerance-window exactness, $\mathsf{Real}^\uparrow$ seal factorization, non-escape boundary, common-window refinement, downstream unblock package, and standard bridge package for the finite $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{CauchyModulus}^\uparrow$ or $\mathsf{RegularityModulus}^\uparrow$, $\mathsf{Real}^\uparrow$, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NullSequence}^\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.NullSequenceUp.NullSequenceCarrier\_namecert\_obligation\_surface`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Dedekind cuts, quotient streams, host real equality, trichotomy, or real completeness.

Upgrade path: No further paper-axis theory upgrade remains; bridge checking remains over the same standard bridge, regular-rational, dyadic, stream-window, modulus, and $\mathsf{Real}^\uparrow$ seal rows.

Chapter 381

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequence}^{\uparrow}$

The $\mathsf{RegularCauchySubsequence}^{\uparrow}$ interface records the constructive tail-subsequence and reindexing surface used by regular Cauchy names. Following Bishop and Bridges, *Constructive Analysis*, 1985, it treats a subsequence as certified monotone and cofinal unary reindexing data over finite regular Cauchy rows. It never selects an infinite subsequence by choice and never changes the real seal through quotient streams, host sequence equality, or ambient real completeness.

Its dependency list is $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$.

Definition 381.1 (RegularCauchySubsequence carrier)**.** *A* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *source packet is a finite* BHist *row*

$$G = (S, T, W, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *S is a* $\mathsf{RegularCauchyName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row;*
2. *T is a certified unary reindexing row whose displayed order ledger is monotone and cofinal on the inspected indices;*
3. *W records the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows before and after reindexing;*
4. *R records the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, point observations, and regularity rows;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary read by downstream real-name consumers;*
6. *H, C, P, and N are respectively the visible* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequence}^{\uparrow}}$ *provenance rows.*

The classifier compares two packets only by the source row, the displayed monotone-cofinal reindexing ledger, common finite windows, transported dyadic-radius and regularity rows, the real-seal boundary, and the four provenance coordinates.

Lean4 (definition) `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceUp`

Theorem 381.2 (RegularCauchySubsequence NameCert obligations)**.** *Every accepted* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *packet with carrier Definition 381.1 carries the NameCert obligation rows for constructive tail-subsequence use:*

1. *the source row is an accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *packet;*

2. *the reindexing row is displayed as a monotone and cofinal unary ledger;*
3. *every chosen finite* $\mathsf{StreamName}^{\uparrow}$ *window is obtained by transporting an inspected source window along that ledger;*
4. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison rows are transported through the same window route;*
5. *the* hsame, Cont, Pkg, *and* NameCert *provenance exhausts the local certificate ledger.*

No obligation row contains quotient streams, selected subsequences by choice, host sequence equality, or real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier namecert obligations`

Proof. Read the carrier coordinates. The source coordinate supplies the regular Cauchy or regular rational packet, and the T coordinate supplies a finite monotone-cofinal unary ledger rather than a host-level subsequence selector.

For a $\mathsf{RegSeqRat}^{\uparrow}$ source, use the cofinal-tail rows such as Definition 301.21 and Definition 301.28. These identify the displayed tail window, the transported dyadic endpoints, the radius rows, and the pointwise regular rational comparisons on a common cofinal refinement.

For a $\mathsf{RegularCauchyName}^{\uparrow}$ source, use Definition 371.3 and Definition 371.6. They force each consumer read through a finite $\mathsf{StreamName}^{\uparrow}$ window and keep the dyadic-radius, pointwise $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows visible.

The classifier of Definition 381.1 admits no additional comparison coordinate. Hence the five obligation rows are exhausted by the displayed source, reindexing, window, transported regularity, and provenance ledgers, with no quotient, choice-selected subsequence, host equality, or completeness datum. □

Theorem 381.3 (RegularCauchySubsequence Real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *packet is read only after the source packet, monotone-cofinal reindexing ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, and pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *comparisons have been displayed. Therefore tail-subsequence reindexing preserves the real-name seal surface as a finite BEDC ledger and does not export quotient-stream equality, a choice-selected subsequence, host sequence equality, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier real seal boundary`

Proof. Start from an accepted carrier. The E coordinate is placed after the source, reindexing, window, transported radius, regularity, and provenance rows, so the real-facing read is reached only along that displayed route.

Apply Definition 381.2. It supplies the monotone-cofinal reindexing row, the finite-window transport, and the transported dyadic-radius and pointwise regular rational comparisons needed by the seal.

The seal itself is then the same kind of boundary used by Definition 371.4 and the RegSeqRat cofinal-tail seal rows. Since the packet has no coordinate for a quotient stream, an externally selected infinite subsequence, host sequence equality, or real completeness, no such datum can be read from the $\mathsf{Real}^{\uparrow}$ boundary. □

Theorem 381.4 (RegularCauchySubsequence monotone-cofinal window transport)**.** *Let* $G = (S, T, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *packet, and let* u *be a unary*

index whose source window is displayed by S. If the reindexing row T records a monotone-cofinal route from u to the selected index $T(u)$, then the packet transports the selected finite StreamName$^{\uparrow}$ *window at $T(u)$ together with its* DyadicRatCore$^{\uparrow}$ *radius row and pointwise* RegSeqRat$^{\uparrow}$ *comparison rows through the same* BHist *route. No selected window may be read without that monotone-cofinal row and its transported radius and regularity evidence.*

Proof. Project the coordinates S, T, W, and R from the accepted carrier. The row T is the only coordinate that names a reindexing route, so a selected window must be reached through that displayed monotone-cofinal ledger.

Use the window clause of Definition 381.2. It identifies each selected StreamName$^{\uparrow}$ window as the transport of an inspected source window along the same unary route.

The radius and regularity rows are stored in R beside that transported window. Their DyadicRatCore$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ reads therefore move with the window row, and the packet has no coordinate from which an independent subsequence window could be selected. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier monotone cofinal window transport`

Theorem 381.5 (RegularCauchySubsequence tail-classifier stability)**.** *Let G and G' be accepted* RegularCauchySubsequence$^{\uparrow}$ *packets. Suppose their source rows, monotone-cofinal reindexing ledgers, selected finite windows, dyadic-radius rows, regularity rows,* Real$^{\uparrow}$ *seal rows, and* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows are transported componentwise by displayed* hsame *routes. Then the local* RegularCauchySubsequence$^{\uparrow}$ *classifier gives the same tail-subsequence comparison for G and G' on every shared finite window.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier tail classifier stability`

Proof. Unfold the classifier described in Definition 381.1. It compares only the source, reindexing ledger, common finite windows, transported radius and regularity rows, real-seal boundary, and visible provenance.

Move those coordinates componentwise along the displayed hsame routes. The Cont rows preserve the read paths for the selected windows, while the Pkg and local NameCert rows keep the same dependency certificates attached to the transported packet.

Apply Definition 381.4 on each shared window. The window, radius, and regularity data remain tied to the same monotone-cofinal route, so the classifier has no remaining coordinate on which the transported comparison could differ. □

Theorem 381.6 (RegularCauchySubsequence Real seal factorization)**.** *Every* Real$^{\uparrow}$ *seal read from an accepted* RegularCauchySubsequence$^{\uparrow}$ *packet factors through the source packet, the monotone-cofinal reindexing ledger, the selected finite* StreamName$^{\uparrow}$ *windows, the transported* DyadicRatCore$^{\uparrow}$ *radius rows, and the* RegSeqRat$^{\uparrow}$ *regularity route before it reaches the seal row. Thus the real-facing boundary is a finite tail-reindexing ledger rather than a quotient-stream, choice-selected subsequence, host sequence equality, or completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier real seal factorization`

Proof. Begin with the seal row E of the accepted packet. By Definition 381.3, a consumer reaches E only after the source, reindexing, window, radius, and regularity rows have been displayed.

Apply Definition 381.4. It fixes the route from the source window through the monotone-cofinal ledger to the selected finite window and carries the corresponding dyadic-radius and regularity rows with it.

The remaining read is the $\mathsf{Real}^{\uparrow}$ seal row itself. Since the carrier contains no quotient stream, selected infinite subsequence, host equality, or completeness coordinate, the displayed route exhausts the factorization of every public real-seal read. □

Theorem 381.7 (RegularCauchySubsequence RegSeqRat sibling consumption)**.** *Every accepted* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *packet consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier and classifier surface of chapter 301 and the regular Cauchy name surface of chapter 371 before it exports its subsequence* $\mathsf{Real}^{\uparrow}$ *seal. The subsequence packet may transport finite windows and tail classifiers, but it cannot present a direct* $\mathsf{Real}^{\uparrow}$ *seal that bypasses the sibling* $\mathsf{RegSeqRat}^{\uparrow}$ *packet or the displayed* $\mathsf{RegularCauchyName}^{\uparrow}$ *source row.*

Proof. Start with an accepted packet $G = (S, T, W, R, E, H, C, P, N)$. The source row S is either a $\mathsf{RegularCauchyName}^{\uparrow}$ row or a $\mathsf{RegSeqRat}^{\uparrow}$ row, and the regularity entries in R are the pointwise $\mathsf{RegSeqRat}^{\uparrow}$ comparison rows attached to the selected finite windows.

Apply the monotone-cofinal window transport theorem. Each selected subsequence window is read through the same reindexing ledger T, and the transported dyadic-radius and regularity rows move with that window rather than being selected independently.

Apply tail-classifier stability. Componentwise hsame transport preserves the source row, transported windows, radius rows, regularity rows, seal row, and provenance rows, so a classifier-facing consumer cannot replace the sibling $\mathsf{RegSeqRat}^{\uparrow}$ surface by an unrecorded real seal.

Finally apply the real seal factorization theorem. The public $\mathsf{Real}^{\uparrow}$ seal is reached only after the source surface, monotone-cofinal ledger, transported finite windows, dyadic-radius rows, and regularity route have been displayed. Thus the sibling $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{RegularCauchyName}^{\uparrow}$ surfaces are consumed before the seal is exported, and the carrier has no coordinate for a bypassing direct $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 381.8 (RegularCauchySubsequence scoped kernel interface)**.** *For every accepted* RegularCauchySubseque *packet* $G = (S, T, W, R, E, H, C, P, N)$*, the complete carrier surface exposed to downstream BEDC consumers is exactly the source row* S*, the monotone-cofinal reindexing ledger* T*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows in* R*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity route in* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the visible* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance rows* H*,* C*,* P*, and* N*. No consumer of the packet can obtain an additional L10 kernel dependency, a quotient-stream representative, a choice-selected subsequence, host sequence equality, or ambient real completeness from the carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier scoped kernel interface`

Proof. Project the accepted carrier of Definition 381.1. Its coordinates list the source, reindexing ledger, window ledger, transported dyadic and regularity rows, seal row, and provenance rows, and the classifier compares no further coordinate.

Apply the NameCert obligation rows of Definition 381.2. They keep the source packet, monotone-cofinal ledger, window transport, dyadic-radius rows, pointwise regular rational comparisons, and provenance ledger visible in the same finite packet.

Use Definition 381.4 and Definition 381.5. These keep each selected window and its transported radius and regularity evidence tied to the displayed reindexing row under hsame and Cont transport.

Finally apply the seal factorization of Definition 381.6. The public real-facing read reaches E only through those displayed rows, so the listed coordinates are the scoped kernel interface and no unlisted dependency can be exported. □

Theorem 381.9 (RegularCauchySubsequence public export)**.** *The carrier of Definition 381.1, the NameCert obligations of Definition 381.2, the real seal boundary of Definition 381.3, the monotone-cofinal window transport of Definition 381.4, the tail-classifier stability of Definition 381.5, the real seal factorization of Definition 381.6, and the scoped kernel interface of Definition 381.8 assemble the public* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *export consumed by* $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Rat}^{\uparrow}$.

The exported object is finite BHist *reindexing data: a source row, a displayed monotone-cofinal unary ledger, selected finite stream windows, transported dyadic-radius and regularity rows, a real-seal boundary, and visible* hsame, Cont, Pkg, *and* NameCert *provenance. It is not a choice-selected infinite subsequence and not host sequence equality.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier public export`

Proof. Project the accepted carrier

$$G = (S, T, W, R, E, H, C, P, N).$$

The carrier and obligation theorems identify the source, monotone-cofinal reindexing ledger, finite windows, transported radius and regularity rows, seal row, and provenance rows.

The boundary, window-transport, classifier-stability, and seal-factorization theorems show that every downstream read travels through those displayed coordinates. The scoped kernel interface then records that no additional dependency is available to a consumer.

Therefore the public export is precisely the finite BHist reindexing surface consumed by the named regular Cauchy, regular rational, stream, dyadic, real, and rational chapters. Since the carrier has no coordinate for an externally selected infinite subsequence or host sequence equality, those objects are not exported. □

Theorem 381.10 (RegularCauchySubsequence standard bridge source)**.** *The public export of Definition 381.9 supplies the BEDC source side of a standard regular-Cauchy subsequence interface. Its bridge source is exactly the finite packet*

$$G = (S, T, W, R, E, H, C, P, N)$$

whose rows are the $\mathsf{RegularCauchyName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source, the monotone-cofinal unary reindexing ledger, the transported* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows, the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the visible* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequence}^{\uparrow}}$ *provenance. The standard readback is therefore a regular-Cauchy subsequence presentation whose subsequence map is read only from T, whose observations are read only from W and R, and whose real-name boundary is read only from E after those rows have been displayed.*

Proof. Start with the public export. It is assembled from the carrier, obligation, window-transport, classifier-stability, seal-factorization, and scoped-interface rows, so every field of the standard subsequence presentation has a named BEDC source.

The forward bridge read sends the finite packet to the standard presentation by reading the reindexing map from the displayed monotone-cofinal ledger T, the selected observations from W,

and the radius and regularity evidence from R. The real-facing endpoint is then read from E, after the source and transported window route have already been exposed.

For the reverse read, admit only a standard presentation whose source, reindexing, window, radius, regularity, seal, and provenance fields are these same displayed rows. Repacking those rows gives the original carrier packet G, and the public export theorem identifies it as the BEDC source side of the bridge. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier standard bridge source`

Theorem 381.11 (RegularCauchySubsequence bridge non-escape)**.** *The bridge source of Definition 381.10 exports no quotient stream, choice-selected infinite subsequence, host sequence equality, host real equality, or ambient real completeness principle. Any standard regular-Cauchy subsequence presentation admitted by the bridge must factor through the same finite* BHist *packet, the same componentwise* hsame *transport, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary used by the public export.*

Proof. The carrier contains no coordinate for a quotient stream, host sequence equality, host real equality, or completion object. Its only subsequence datum is the displayed finite monotone-cofinal ledger T.

Apply Definition 381.4. It forces every selected finite window to be transported along T together with its dyadic-radius and regularity rows, so a standard presentation cannot read an independently chosen infinite subsequence.

Apply Definition 381.5 and Definition 381.6. Classifier movement is componentwise hsame on the displayed rows, and every real-facing read factors through the same finite route before reaching E. Thus the bridge has no remaining field through which the excluded host or quotient data could enter. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier bridge non escape`

Theorem 381.12 (RegularCauchySubsequence bridged export)**.** *The carrier of Definition 381.1, the public export of Definition 381.9, the monotone-cofinal window transport of Definition 381.4, the tail-classifier stability of Definition 381.5, the real seal factorization of Definition 381.6, the standard bridge source of Definition 381.10, and the bridge non-escape result of Definition 381.11 assemble a bridged* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *export to the standard regular-Cauchy subsequence interface. The bridge preserves exactly the displayed finite source, reindexing, window, radius, regularity, seal, classifier, continuation, package, and naming rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier bridged export`

Proof. First assemble the BEDC side. The carrier and public export identify the complete finite packet, while the window transport theorem ties every selected observation to the monotone-cofinal reindexing ledger.

Next preserve the classifier and seal fields. Tail-classifier stability gives componentwise hsame transport for the source, reindexing, window, radius, regularity, seal, and provenance rows. Seal factorization keeps the real-facing endpoint behind that same finite route.

Finally apply the standard bridge source and bridge non-escape theorems. The forward and reverse reads use only the listed rows, and the excluded quotient, choice, host-equality, and completeness data have no bridge field. Therefore the public finite tail-subsequence export is the bridged regular-Cauchy subsequence interface at the paper level. □

Theorem 381.13 (RegularCauchySubsequence mature consumer example)**.** *A* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegularCauchyName}^{\uparrow}$ *consumer using a monotone-cofinal subsequence factors through the displayed* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *finite-window interface: the source row, monotone-cofinal reindexing ledger, finite observation windows, dyadic-radius rows, regularity rows, tail-classifier transport, real-seal row, standard bridge source,* hsame *witnesses,* Cont *routes,* Pkg *provenance, and* NameCert *row. The consumer never selects a quotient representative, a host subsequence function, a host stream equality, or a host real equality.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier mature consumer route`

Proof. Begin with the consumer's requested finite window. The carrier and Definition 381.9 expose the source, reindexing ledger, selected windows, radius rows, regularity rows, seal row, transport rows, routes, provenance, and local certificate.

Apply Definition 381.4. It routes the consumer window through the displayed monotone-cofinal ledger and keeps the requested observations attached to the same finite subsequence packet.

Next apply Definition 381.5 and Definition 381.6. The classifier-facing rows move through hsame and Cont, and the real-facing read reaches $\mathsf{Real}^{\uparrow}$ only through the stored seal row.

Finally use Definition 381.12. The bridge source and non-escape rows admit exactly the finite-window presentation already displayed, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegularCauchyName}^{\uparrow}$ consumer factors through the packet without selecting quotient representatives or host-level equality data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequence}^{\uparrow}$ packet is generated from a regular Cauchy or regular rational source, a displayed monotone-cofinal unary reindexing ledger, selected finite stream windows, transported dyadic-radius and regularity rows, a real-seal boundary, and visible provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySubsequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 381.13 records the mature consumer surface over the bridged export, carrier, public finite tail-subsequence export, monotone-cofinal finite-window transport, tail-classifier stability, $\mathsf{RegSeqRat}^{\uparrow}$ sibling consumption, $\mathsf{Real}^{\uparrow}$ seal factorization, standard bridge source, and bridge non-escape rows over $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySubsequence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchySubsequenceUp.RegularCauchySubsequenceCarrier\_public\_export`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close real completeness, quotient stream equality, host sequence equality, host real equality, selected infinite subsequences, uniqueness of limits, or bridge checking.

Upgrade path: No further paper-side theory grade is required for this finite-window consumer surface.

Chapter 382

A Concrete Naming Certificate for $\mathsf{DyadicSpline}^{\uparrow}$

The $\mathsf{DyadicSpline}^{\uparrow}$ interface names finite dyadic piecewise-linear approximation data as a BEDC naming certificate. Following de Boor, *A Practical Guide to Splines*, and Schumaker, *Spline Functions*, the standard finite spline reading is a knot row, coefficient rows, and a segment ledger. BEDC records only those finite displayed rows and the real-name boundary through which downstream approximation consumers read them; it does not add host functions, quotient reals, choice-selected limits, or an ambient vector space of splines.

Its dependency anchors are $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicApproximation}^{\uparrow}$. The finite spline carrier consumes the dyadic mesh carrier of chapter 358 for knot and adjacency rows, and the dyadic rational core of chapter 291 for endpoint coefficients.

Definition 382.1 (DyadicSpline finite carrier)**.** *A* $\mathsf{DyadicSpline}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, A, L, I, Q, R, H, C, P, N)$$

with the following visible rows:

1. *K is a* $\mathsf{DyadicMesh}^{\uparrow}$ *knot row, listing a finite ordered dyadic mesh and the cell adjacency data inspected by the spline reader;*
2. *A is a finite family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *coefficient rows, with two endpoint coefficient reads for each retained segment;*
3. *L is the piecewise-linear segment ledger, pairing each adjacent knot cell with its two coefficient rows and its displayed affine interpolation route;*
4. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger through which every segment domain and endpoint order is read;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by evaluating a requested finite window against the segment ledger and coefficient rows;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row through which downstream real-name consumers read the same finite observation windows;*
7. *H stores componentwise* hsame *transports for knots, coefficients, segment ledgers, interval cells, rational readbacks, real seals, provenance, and the local naming row;*

8. *C stores the displayed* Cont *routes for mesh refinement, segment readback, interval-cell inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *consumption, and* $\mathsf{Real}^{\uparrow}$ *sealing;*
9. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, *and the consumed local source certificates;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSpline}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting them to a common displayed dyadic refinement, comparing the refined knot rows, comparing the transported coefficient rows through $\mathsf{DyadicRatCore}^{\uparrow}$, *comparing cell boundaries through* $\mathsf{RationalInterval}^{\uparrow}$, *and then reading the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary rows. The carrier has no coordinate for a host function, a completed real graph, a quotient real, an infinite knot sequence, or a choice-selected limit.*

Lean4 (definition) `BEDC.Derived.DyadicSplineUp.TasteGate.DyadicSplineUp`
Lean4 ✓ `BEDC.Derived.DyadicSplineUp.TasteGate.DyadicSplineTasteGate single carrier alignment`

Theorem 382.2 (DyadicSpline NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicSpline}^{\uparrow}}$ *sketch has five rows:*

1. *carrier inhabitation: finite knot and coefficient packets inhabit the carrier of Definition 382.1;*
2. *classifier transport: accepted carriers are compared only after displayed transport of knots, coefficients, segment ledgers, rational cells,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows;*
3. *mesh-refinement stability: displayed dyadic refinement replaces each segment by a finite family of subsegments while preserving transported endpoint coefficients and interval-cell boundaries;*
4. *boundary non-escape: every rational readback and real-name seal factors through the finite segment ledger,* $\mathsf{RationalInterval}^{\uparrow}$ *cell row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row;*
5. *standard reading: the public piecewise-linear reading is exactly the finite knot row, coefficient rows, segment ledger, cell ledger, readback row, seal row, transport rows, continuation routes, package provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.DyadicSplineUp.DyadicSplineFiniteCarrier namecert obligation surface`

Proof. Carrier inhabitation is obtained by choosing a finite $\mathsf{DyadicMesh}^{\uparrow}$ knot row and attaching two $\mathsf{DyadicRatCore}^{\uparrow}$ coefficient reads to each adjacent knot cell. The segment ledger records the displayed affine interpolation route, and the remaining carrier coordinates are the interval, readback, seal, transport, continuation, provenance, and local name rows.

The classifier row is componentwise after common displayed refinement. Knot rows are compared through $\mathsf{DyadicMesh}^{\uparrow}$, coefficient rows through $\mathsf{DyadicRatCore}^{\uparrow}$, cell boundaries through $\mathsf{RationalInterval}^{\uparrow}$, and consumer-facing observations through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

The stability, non-escape, and standard-reading rows are the three public obligations proved by the following theorems. Together they list every coordinate a downstream consumer can inspect, so the seed certificate surface is finite and contains no host function or completed real graph. □

Theorem 382.3 (DyadicSpline mesh refinement handoff)**.** *Let $S = (K, A, L, I, Q, R, H, C, P, N)$ be an accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *carrier. If K' is obtained from K by displayed* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement and each old segment in L is replaced by the finite list of subsegments induced by the refined knot row, with endpoint coefficient rows transported through the same* hsame *witnesses, then the refined packet*

$$S' = (K', A', L', I', Q', R', H', C', P, N)$$

is again an accepted DyadicSpline$^{\uparrow}$ *carrier. Its* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer rows factor through the same finite piecewise-linear reading, now displayed on the refined mesh.*

Lean4 ✓ `BEDC.Derived.DyadicSplineUp.DyadicSplinePacket mesh refinement handoff`

Proof. Project the carrier of Definition 382.1. The knot row, coefficient rows, and segment ledger are finite, so a displayed mesh refinement subdivides each old segment into a finite family of adjacent subsegments.

Transport the endpoint coefficient rows along H. The refined segment ledger L' uses the transported endpoint rows and the same affine readback route restricted to each refined cell, while I' records the corresponding RationalInterval$^{\uparrow}$ cell boundaries.

Route the rational readback row Q' and real seal row R' through the refined continuation routes in C'. Repacking these rows gives exactly the carrier coordinates, so the refined packet is accepted and exposes the same piecewise-linear data on the finer displayed mesh. □

Theorem 382.4 (DyadicSpline endpoint mesh transport stability)**.** *Let $S = (K, A, L, I, Q, R, H, C, P, N)$ be an accepted* DyadicSpline$^{\uparrow}$ *packet, and let K' be a displayed refinement of the mesh row K. When the endpoint coefficient rows in A, the rational cell rows in I, and the segment ledger rows in L are transported componentwise along the displayed* hsame *witnesses in H, the refined rows assemble with the transported* RegSeqRat$^{\uparrow}$ *readbacks and* Real$^{\uparrow}$ *seals into one classifier-facing packet*

$$S' = (K', A', L', I', Q', R', H', C', P, N).$$

The accepted comparison between S and S' is exhausted by the displayed knot, endpoint, cell, ledger, readback, and seal transports together with the Cont *routes for refined-cell inspection. No host function, graph, quotient real, infinite knot sequence, or selected limit is introduced by the handoff.*

Proof. Project the carrier of Definition 382.1. The knot row, endpoint coefficient family, segment ledger, rational cell ledger, readback row, seal row, transport row, continuation row, provenance row, and local name row are all explicit coordinates of S.

Use the displayed hsame entries in H to move the knot row to K' and to move each endpoint coefficient row to the refined endpoint ledger. The rational cell rows move by the same componentwise transport, and each refined segment in L' records the transported endpoints and the corresponding cell row.

Route every refined rational observation through the displayed Cont entries in C. The RegSeqRat$^{\uparrow}$ readback row Q' and Real$^{\uparrow}$ seal row R' are therefore read only after the refined segment and rational cell rows have been selected.

Repacking these transported coordinates gives the packet S' named in the statement. Since every inspected coordinate is a carrier projection or a displayed continuation route, there is no remaining place for a host function, graph, quotient real, infinite knot sequence, or selected limit to enter the classifier. □

Theorem 382.5 (DyadicSpline RegSeqRat Real boundary)**.** *For every accepted* DyadicSpline$^{\uparrow}$ *carrier $S = (K, A, L, I, Q, R, H, C, P, N)$ and every finite consumer window, the* RegSeqRat$^{\uparrow}$ *readback and* Real$^{\uparrow}$ *seal are reached only by first selecting a displayed segment from L, reading its* RationalInterval$^{\uparrow}$ *cell from I, reading its endpoint coefficients from A, and routing the resulting finite rational observation through Q and R.*

The boundary exports finite rational observations and real-name seal rows only. It does not export quotient stream equality, an ambient real-valued function, a host graph, uniform convergence of a sequence of splines, or a choice-selected limit.

Lean4 ✓ `BEDC.Derived.DyadicSplineUp.DyadicSplinePacket regseqrat real boundary`

Proof. Start with the requested finite window. The carrier selects a segment through the knot row K and the segment ledger L; no other coordinate can choose a domain cell.

Read the selected cell through I and the two endpoint coefficient rows through A. The displayed affine interpolation route in L turns these finite dyadic rows into a rational observation at the requested window.

The observation is then routed through the $\mathsf{RegSeqRat}^{\uparrow}$ row Q and sealed through the $\mathsf{Real}^{\uparrow}$ row R. Since every step is a projection from the finite carrier and its displayed continuation routes, quotient stream equality, host functions, graphs, uniform convergence, and choice-selected limits remain outside the exported boundary. □

Theorem 382.6 (DyadicSpline standard piecewise-linear reading)**.** *The standard finite piecewise-linear reading of an accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *carrier is exactly the tuple consisting of the* $\mathsf{DyadicMesh}^{\uparrow}$ *knot row, finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *coefficient rows, segment ledger,* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

This reading is stable under the mesh-refinement handoff of Definition 382.3 and has the non-escape boundary of Definition 382.5. It is the whole public surface exported to $\mathsf{DyadicApproximation}^{\uparrow}$ *consumers.*

Proof. Project the carrier of Definition 382.1. Its rows are precisely the knot row, coefficient rows, segment ledger, cell ledger, rational readback, real seal, transport, continuation, provenance, and local name coordinates.

Apply Definition 382.3. Refinement changes the displayed mesh and segment ledger by finite subdivision, while preserving the transported endpoint coefficients and consumer-facing readback route.

Apply Definition 382.5. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary rows are reached only through the finite segment and cell ledgers, so no extra analytic object can be exported.

The downstream $\mathsf{DyadicApproximation}^{\uparrow}$ consumer therefore receives exactly the finite piecewise-linear tuple named in the statement. There is no remaining carrier coordinate from which a host function, completed real graph, quotient real, or choice-selected limit could be read. □

Theorem 382.7 (DyadicApproximation DyadicSpline consumer route)**.** *Every* $\mathsf{DyadicApproximation}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *packet must read the packet through the* $\mathsf{DyadicApproximation}^{\uparrow}$ *chapter surface of chapter 335 together with the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 382.1.*

The consumer route is exactly the finite piecewise-linear tuple of Definition 382.6, transported through the mesh-refinement handoff of Definition 382.3 and sealed by the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *boundary of Definition 382.5. It exports no host spline function, quotient real graph, selected limit, ambient completion, infinite knot row, or independent analytic approximation object.*

Proof. Start with an accepted packet $S = (K, A, L, I, Q, R, H, C, P, N)$. The standard reading theorem identifies the public surface of S with the displayed mesh row, endpoint coefficient rows, segment ledger, rational cell ledger, readback row, seal row, transports, continuations, provenance, and local name row.

A $\mathsf{DyadicApproximation}^{\uparrow}$ consumer reaches that surface through chapter 335. Its dyadic observations therefore read the $\mathsf{DyadicRatCore}^{\uparrow}$ coefficients along the $\mathsf{DyadicMesh}^{\uparrow}$ cells, inspect the

$\mathsf{RationalInterval}^{\uparrow}$ cell ledger, and pass the resulting finite observation through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

If the consumer asks for a finer displayed window, apply the mesh-refinement handoff. Refinement replaces the mesh and segment ledger by finite subdivided rows while keeping the transported endpoint coefficients, rational cell boundaries, and consumer-facing readback route inside the same carrier format.

Finally apply the RegSeqRat/Real boundary theorem. Since the route to the seal factors through the finite segment and cell ledgers before reaching Q and R, there is no coordinate from which the consumer can read a host function, quotient real graph, selected limit, ambient completion, infinite knot row, or separate analytic approximation object. □

Theorem 382.8 (DyadicSpline common refinement segment exhaustion)**.** *Let $S = (K, A, L, I, Q, R, H, C, P, N)$ and $S' = (K', A', L', I', Q', R', H', C', P', N')$ be accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *packets compared by classifier transport. Then their comparison refines to one displayed dyadic mesh and one finite segment ledger. Every compared endpoint coefficient, rational cell,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal read factors through that common mesh and ledger.*

The exhaustion exports only the shared $\mathsf{DyadicMesh}^{\uparrow}$ *row, finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint coefficients,* $\mathsf{RationalInterval}^{\uparrow}$ *cell rows, segment ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *data. It does not read a host function, completed graph, quotient real, infinite knot sequence, or selected limit.*

Proof. Project both accepted packets through Definition 382.1. The classifier compares knot rows, endpoint coefficient rows, rational cell rows, segment ledgers, readback rows, seal rows, and provenance rows only after displayed transport.

Apply Definition 382.3 to refine both mesh rows to a common displayed dyadic mesh. Then apply Definition 382.4 to transport the endpoint coefficients, interval cells, and segment ledgers onto that common mesh.

Finally use Definition 382.6. Once the packets have been transported to the same mesh, the public read is exactly the finite mesh, endpoint, cell, ledger, readback, seal, transport, continuation, provenance, and local naming tuple, so no host-level reading remains. □

Theorem 382.9 (DyadicSpline zero segment carrier)**.** *The constant zero endpoint coefficient packet inhabits the* $\mathsf{DyadicSpline}^{\uparrow}$ *carrier. Its segment ledger has the displayed zero* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows on every retained cell, its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is the zero readback route, and its* $\mathsf{Real}^{\uparrow}$ *seal factors through that same zero readback.*

This carrier is a finite BEDC packet over $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *only. It does not add a host zero function, completed graph, quotient real, infinite mesh, or selected limit.*

Proof. Start with any displayed finite $\mathsf{DyadicMesh}^{\uparrow}$ row and choose the zero $\mathsf{DyadicRatCore}^{\uparrow}$ coefficient row at both endpoints of each retained segment. The definition of Definition 382.1 then supplies the finite segment ledger, the rational cell ledger, transport rows, continuation rows, provenance, and local name row.

Read a consumer window through the segment ledger. Because both endpoint coefficient rows on every segment are the displayed zero rows, the resulting finite rational observation is the zero $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Apply Definition 382.5 to route the $\mathsf{Real}^{\uparrow}$ seal through that readback, and apply Definition 382.6 to identify the public packet with the finite zero mesh data. No separate host zero function or completed graph is introduced. □

Theorem 382.10 (DyadicSpline Real seal mesh factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer read of an accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *packet factors through the displayed mesh, cell ledger, endpoint coefficients,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* hsame, Cont, Pkg, *and* NameCert *rows. The consumer reaches the seal only after selecting a finite mesh cell, reading its endpoint coefficients, and passing the resulting rational observation through the regular-rational readback.*

The factorization exports no host function semantics, completed real graph, quotient real, uniform convergence claim, infinite knot sequence, or choice-selected limit.

Proof. Project the accepted packet through Definition 382.1. The only seal-facing coordinates are the mesh row, endpoint coefficient rows, segment ledger, rational cell ledger, readback row, real seal row, transports, continuations, provenance, and local naming row.

Apply Definition 382.5. A real-name consumer first chooses a displayed segment, reads its $\mathsf{RationalInterval}^{\uparrow}$ cell and endpoint coefficients, and then routes the finite rational observation through the $\mathsf{RegSeqRat}^{\uparrow}$ row before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Apply Definition 382.8 when the consumer compares transported packets, and apply Definition 382.6 to identify the exported surface. Thus every seal read factors through the finite mesh package, with no ambient analytic object available to the consumer. □

Theorem 382.11 (DyadicSpline refinement classifier transport)**.** *Let* $S = (K, A, L, I, Q, R, H, C, P, N)$ *and* $S' = (K', A', L', I', Q', R', H', C', P', N')$ *be accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *packets whose classifier comparison is displayed on a common dyadic refinement. Then the classifier transports the knot rows, endpoint coefficient rows, segment ledgers, rational cell rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows through that same refinement.*

The transported comparison is exhausted by the $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *coordinates of the two finite carriers. It has no coordinate for a host spline function, completed graph, quotient real, infinite knot sequence, or selected limit.*

Proof. Project both packets through Definition 382.1. The only classifier-facing rows are the displayed knot, endpoint, segment, cell, readback, seal, transport, continuation, provenance, and local naming coordinates.

Apply Definition 382.8. The comparison first moves both carriers to one displayed dyadic mesh and one finite segment ledger, so every later read is made on that common finite row.

Apply Definition 382.4. Endpoint coefficients, cell rows, segment entries, rational readbacks, and real seals move componentwise along the stored hsame witnesses and displayed Cont routes.

The standard reading theorem then identifies the transported packet with the same finite piecewise-linear surface. Since each inspected item is a carrier projection or displayed continuation route, no host-level analytic object remains in the classifier. □

Theorem 382.12 (DyadicSpline Real seal stability)**.** *For every accepted* $\mathsf{DyadicSpline}^{\uparrow}$ *packet and every displayed dyadic refinement accepted by Definition 382.3, each* $\mathsf{Real}^{\uparrow}$ *seal read after refinement is still routed through a selected segment row, its rational cell row, its transported endpoint coefficient rows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Thus refinement preserves the real-seal non-escape boundary: the seal read remains inside the finite $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *packet.*

Proof. Start with the refined packet supplied by Definition 382.3. Its refined mesh, endpoint rows, segment ledger, cell ledger, readback row, seal row, transports, continuations, provenance, and local name are again the coordinates of Definition 382.1.

Use Definition 382.10. A seal consumer first selects a finite mesh cell, reads its transported endpoint coefficients, and then reaches the $\mathsf{RegSeqRat}^{\uparrow}$ row before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

The boundary theorem Definition 382.5 excludes every route outside those finite rows. Therefore mesh refinement changes the displayed carrier rows but does not open a path to host functions, quotient reals, completed graphs, infinite meshes, or selected limits. □

Theorem 382.13 (DyadicSpline obligation closure package)**.** *The finite carrier of Definition 382.1, the NameCert obligation surface of Definition 382.2, the mesh-refinement handoff, the endpoint transport stability theorem, the rational-real boundary theorem, the standard piecewise-linear reading, the refinement classifier transport theorem, and the real-seal stability theorem assemble the* $\mathsf{DyadicSpline}^{\uparrow}$ *obligation package.*

The package supplies carrier habitation, classifier transport, mesh-refinement stability, readback boundary, real-seal stability, and the standard finite piecewise-linear reading for $\mathsf{DyadicSpline}^{\uparrow}$ *consumers. It asserts no host-function semantics, quotient real, completed spline graph, infinite knot sequence, uniform convergence theorem, or choice-selected limit.*

Proof. Carrier habitation and the row inventory are supplied by Definition 382.1 and Definition 382.2. They expose the finite mesh, endpoint, segment, cell, readback, seal, transport, continuation, package, and local name rows.

Mesh-refinement stability is Definition 382.3, with endpoint and cell transport supplied by Definition 382.4. The classifier transport theorem places the compared packets on the same displayed refinement before any consumer read.

The readback and seal boundaries are supplied by Definition 382.5 and Definition 382.12. Finally, Definition 382.6 identifies the public reading with exactly those rows, so the assembled obligation package has no remaining analytic escape coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSpline}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicSpline}^{\uparrow}$ packet is generated by a finite $\mathsf{DyadicMesh}^{\uparrow}$ knot row, finite $\mathsf{DyadicRatCore}^{\uparrow}$ coefficient rows, a piecewise-linear segment ledger, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSpline}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 382.6, Definition 382.13, and the refinement, boundary, classifier-transport, and $\mathsf{DyadicApproximation}^{\uparrow}$ consumer routes export the mature finite piecewise-linear packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSpline}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DyadicSplineUp.DyadicSplineFiniteCarrier\_namecert\_obligation\_surface`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host-function semantics, quotient reals, infinite knot sequences, uniform convergence of spline families, choice-selected limits, or bridge checking.

Upgrade path: Bridge-checked verification remains the next step beyond the mature finite spline refinement, boundary, classifier-transport, and approximation-consumer package.

Chapter 383

A Concrete Naming Certificate for $\mathsf{OptionalStopping}^{\uparrow}$

The $\mathsf{OptionalStopping}^{\uparrow}$ interface names the bounded finite-window form of Doob optional stopping. It records a stopped-process packet over $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, and $\mathsf{StoppingTime}^{\uparrow}$ in which the stopping index is visibly bounded, the stopped value is read from a finite filtration window, and every integrability and provenance row is carried as BEDC data. The chapter does not invoke measure completion, countable choice, unbounded stopping, or an ambient almost-sure quotient.

The bounded stopped-process packet consumes the martingale filtration and conditional-expectation rows of chapter 190 together with the probability, random-variable, and finite stopping-decision surfaces already named in the carrier; $\mathsf{OptionalStopping}^{\uparrow}$ only packages the finite stopped-value ledger over those sibling rows.

Definition 383.1 (OptionalStopping finite carrier)**.** *An* $\mathsf{OptionalStopping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (\Omega, X, T, b, V, F, I, H, C, P, N)$$

with the following visible rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the finite probability context used by the packet;*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *stopped-process family, indexed only over the displayed finite time window;*
3. T *is the* $\mathsf{StoppingTime}^{\uparrow}$ *row whose decision at each inspected time is read from the corresponding filtration window;*
4. b *is the bounded stopping index row, giving the finite terminal window in which the stopping decision must occur;*
5. V *is the stopped value row* X_T *read by the displayed branch that first satisfies the stopping row before* b*;*
6. F *is the finite filtration-window ledger carrying the random-variable reads, stopping decisions, and martingale conditioning rows;*
7. I *is the integrability and provenance row required for the bounded stopped process and each inspected martingale value;*
8. H *stores the displayed* hsame *transports for probability rows, random-variable rows, stopping-time decisions, stopped values, filtration ledgers, and integrability rows;*

9. *C stores the* Cont *routes by which consumers read the bounded stopping index, the stopped value, and the martingale-window handoff;*
10. *P is the* Pkg *provenance row naming* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OptionalStopping}^{\uparrow}}$ *registration row.*

The local classifier compares two carriers only through the transported bounded index, stopped value row, filtration-window ledger, integrability provenance, package row, and local certificate row. It contains no unbounded stopping argument and no quotient-level event equality.

Theorem 383.2 (OptionalStopping NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{OptionalStopping}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier: formation of the finite* BHist *packet of Definition 383.1;*
2. *classifier: componentwise comparison of the bounded index, stopped value, filtration-window ledger, integrability row, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row;*
3. *stopping-time stability: each transported stopping decision is read from the filtration window available at that finite time and remains bounded by b;*
4. *martingale-window handoff: the bounded finite-window Doob optional-stopping read factors through the displayed* $\mathsf{Martingale}^{\uparrow}$ *rows, the stopped value row V, and the filtration ledger F;*
5. *ledger exactness: every probability, random-variable, stopping-time, martingale, integrability, transport, continuation, package, and certificate row used by the stopped process appears in the displayed carrier.*

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier namecert obligations`

Proof. Carrier formation is the packet assembly of Definition 383.1. The bounded index, stopped value, filtration ledger, integrability row, transports, continuations, package provenance, and local certificate are all visible coordinates.

Classifier stability is componentwise. Transport the probability and random-variable rows first, then transport each finite stopping decision along its filtration-window route, and finally transport the stopped value and integrability rows attached to that selected branch.

For stopping-time stability, inspect the finite ledger F. Each decision row of T is recorded at the corresponding finite time and is read only from the filtration data available there; the bound b records that the selected branch must occur inside the displayed window.

The martingale-window handoff is the bounded finite-window form of Doob optional stopping. The stopped value row V is read from the first displayed stopping branch, and the relevant $\mathsf{Martingale}^{\uparrow}$ conditioning rows are exactly the rows stored in F and I.

Ledger exactness follows by projecting O. No consumer read can reach a probability, random variable, stopping-time decision, martingale conditioning row, integrability witness, transport, continuation, package, or local certificate outside the displayed coordinates. □

Theorem 383.3 (OptionalStopping bounded stopped-value readback)**.** *Let $O = (\Omega, X, T, b, V, F, I, H, C, P, N)$ be an accepted* $\mathsf{OptionalStopping}^{\uparrow}$ *carrier. The stopped value row V is read from the first displayed stopping branch of T that occurs before the bound b, together with the corresponding finite process row of X and filtration-window row of F. No stopped value may be read from an unbounded stopping branch or from a branch absent from the displayed finite ledger.*

Proof. Project the stopping row T, the bound b, the stopped value V, and the filtration ledger F from the accepted carrier. The carrier description makes V the row attached to the first displayed branch satisfying the stopping decision before the bound.

Use the stopping-time stability clause of Definition 383.2. It says that each stopping decision is read from the finite filtration data available at that time and remains bounded by b.

The stopped value is therefore obtained from the matching process row in X and the matching filtration-window row in F. Since no unbounded branch is a carrier coordinate, no consumer can read V from outside the displayed finite ledger. □

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier bounded stopped value readback`

Theorem 383.4 (OptionalStopping finite expectation window)**.** *Every expectation-style read of an accepted* OptionalStopping$^{\uparrow}$ *packet factors through the finite probability row, martingale-window rows, filtration-window ledger, bounded stopped value, integrability row, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row already present in the carrier. The read does not invoke measure completion, almost-sure quotient equality, or an unbounded optional-stopping theorem.*

Proof. Begin with the carrier tuple of Definition 383.1. The only probability and expectation-facing coordinates are Ω, the stopped-process rows X, the martingale and filtration ledger F, and the integrability row I.

Apply Definition 383.3 to identify the stopped value route. The selected row is finite, bounded by b, and tied to the displayed stopping-time and filtration windows.

The remaining expectation read can use only the transports, continuation routes, package provenance, and local certificate stored in H, C, P, and N. Thus every expectation-style consumer factors through the displayed finite ledger and never reaches a measure completion, almost-sure quotient, or unbounded optional-stopping principle. □

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier finite expectation window`

Theorem 383.5 (OptionalStopping no-escape boundary)**.** *The public boundary of* OptionalStopping$^{\uparrow}$ *exports the bounded finite-window stopped-value packet and its displayed expectation ledger only. It exports no unbounded stopping time, no measure completion, no almost-sure quotient, and no uniform-integrability principle beyond the finite integrability row recorded in the carrier.*

Proof. Inspect the accepted packet. Its time information is bounded by the displayed row b, and its stopped value is the finite readback of Definition 383.3.

The expectation-facing data is exactly the finite window of Definition 383.4. The integrability row I is part of that finite ledger, not a principle about unbounded families.

No coordinate of O names a measure completion, almost-sure quotient, unbounded stopping argument, or uniform-integrability theorem. Therefore none of those objects can be exported by the public boundary. □

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier no escape boundary`

Theorem 383.6 (OptionalStopping scoped dependency route)**.** *Every accepted* OptionalStopping$^{\uparrow}$ *carrier*

$$O = (\Omega, X, T, b, V, F, I, H, C, P, N)$$

scopes the bounded stopped-value read, finite expectation window, stopping-time stability, martingale-window handoff, and ledger exactness through exactly the displayed ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$,

Martingale$^{\uparrow}$, StoppingTime$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. No dependency route reaches an unbounded stopping branch, measure completion, almost-sure quotient, or unstated uniform-integrability principle.*

Proof. Apply Definition 383.2. Its five obligations provide carrier formation, componentwise classifier transport, stopping-time stability, martingale-window handoff, and ledger exactness for the displayed finite packet.

Next apply Definition 383.3. The stopped value row V is read only from the first displayed stopping branch before b, together with the corresponding process and filtration-window rows.

Apply Definition 383.4. It places every expectation-style read downstream of Ω, X, F, I, V, H, C, P, and N.

Finally apply Definition 383.5. The excluded unbounded, completed, quotient, and unstated integrability objects are absent from the carrier, so the dependency route is exactly the displayed BEDC row surface. □

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier scoped dependency route`

Closure status for NameCert$_{\text{OptionalStopping}^{\uparrow}}$.

Constructive story: An OptionalStopping$^{\uparrow}$ packet is generated from a finite probability row, bounded stopped-process random variables, a stopping-time row, a visible stopping bound, the stopped value, filtration-window ledger rows, integrability provenance, displayed transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{OptionalStopping}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge-facing paper surface is the bounded finite stopping-branch route through post-stop tail erasure and finite-filtration nonescape over ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, Martingale$^{\uparrow}$, and StoppingTime$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{OptionalStopping}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close unbounded stopping, measure completion, almost-sure quotient equality, uniform integrability beyond the displayed finite row, or bridge checking.

Upgrade path: Bridge checking requires a Lean-side standard bridge target for the finite stopping-branch route through post-stop tail erasure and finite-filtration nonescape.

Definition 383.7 (OptionalStopping post-stop truncation)**.** *Let*

$$O = (\Omega, X, T, b, V, F, I, H, C, P, N)$$

be an accepted OptionalStopping$^{\uparrow}$ *carrier. A post-stop truncation of O at a displayed branch $\tau < b$ is the finite packet*

$$O_{\leq\tau} = (\Omega, X_{\leq\tau}, T_{\leq\tau}, \tau, V, F_{\leq\tau}, I_{\leq\tau}, H_{\leq\tau}, C_{\leq\tau}, P, N)$$

obtained by retaining exactly the probability row Ω, the displayed process rows through τ, the stopping-decision rows through τ, the stopped value row V, the filtration-window rows through τ, the integrability rows through τ, the displayed transports and continuation routes whose endpoints remain in those rows, the same package provenance row P, and the same local certificate row N. All filtration-window, integrability, transport, and continuation rows whose time coordinate lies strictly after τ are omitted. The truncation is stopping-branch exact when τ is the first displayed stopping branch of T before b from which the stopped value row V is read.

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingTasteGate single carrier alignment`

Lemma 383.8 (OptionalStopping post-stop tail erasure)**.** *Let*

$$O = (\Omega, X, T, b, V, F, I, H, C, P, N)$$

be an accepted $\mathsf{OptionalStopping}^{\uparrow}$ *carrier. Suppose that* $\tau < b$ *is the first displayed stopping branch of* T *before the visible bound* b*, and let* $O_{\leq\tau}$ *be the stopping-branch exact post-stop truncation of Definition 383.7. Then* $O_{\leq\tau}$ *is an accepted* $\mathsf{OptionalStopping}^{\uparrow}$ *carrier. Moreover, the stopped value row of* $O_{\leq\tau}$ *is classifier-equal to the stopped value row* V *of* O*.*

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier post stop tail erasure`

Proof. Project the accepted carrier O through Definition 383.1. Its consumer-visible rows are exactly Ω, X, T, b, V, F, I, H, C, P, and N, and the local classifier compares stopped-value data only through the transported bounded index, stopped value row, filtration-window ledger, integrability provenance, package row, and local certificate row.

By Definition 383.3, the row V is read from the first displayed stopping branch of T before b, together with the corresponding process row of X and filtration-window row of F. The hypothesis says that this branch is τ. Therefore every row needed to read V lies among

$$\Omega, \quad X_{\leq\tau}, \quad T_{\leq\tau}, \quad \tau, \quad V, \quad F_{\leq\tau}, \quad I_{\leq\tau}, \quad H_{\leq\tau}, \quad C_{\leq\tau}, \quad P, \quad N.$$

The deleted rows have time coordinate strictly after τ, so none of them is an input to the stopped-value readback route.

Acceptance of the truncated packet is now the carrier formation clause of Definition 383.2 applied to the retained finite tuple. Stopping-time stability is preserved because every retained decision row was already read from its corresponding filtration-window row in O, and all those filtration-window rows remain present. The terminal bound for the truncated packet is the displayed branch τ, so the selected stopping decision is still inside the visible finite window.

The martingale-window handoff and integrability requirements are also retained. The handoff for the stopped value uses only the process, filtration, and integrability rows through τ by Definition 383.3, while Definition 383.4 places every expectation-style read downstream of the finite probability row, the stopped value row, the filtration-window ledger, the integrability row, the displayed transports, the continuation routes, package provenance, and the local certificate. Those are exactly the rows kept by $O_{\leq\tau}$ for the stopped branch.

It remains to compare stopped values. Both O and $O_{\leq\tau}$ read their stopped value from the same first stopping branch τ, the same process row at that branch, and the same filtration-window row at that branch. The stopped value coordinate is the same displayed row V, and the transports and continuation route that identify it with the branch read are retained in $H_{\leq\tau}$ and $C_{\leq\tau}$. Hence the local $\mathsf{OptionalStopping}^{\uparrow}$ classifier identifies the stopped-value row of the truncated packet with the original stopped value row V. □

Theorem 383.9 (OptionalStopping finite-filtration nonescape)**.** *Let* $O_{\leq\tau} = (\Omega, X_{\leq\tau}, T_{\leq\tau}, \tau, V, F_{\leq\tau}, I_{\leq\tau}, H_{\leq\tau}, C_{\leq\tau},$ *be a stopping-branch exact post-stop truncation of an accepted* $\mathsf{OptionalStopping}^{\uparrow}$ *carrier. Every expectation-facing consumer of the truncated packet must read the rows*

$$\Omega, \quad X_{\leq\tau}, \quad T_{\leq\tau}, \quad \tau, \quad V, \quad F_{\leq\tau}, \quad I_{\leq\tau}, \quad H_{\leq\tau}, \quad C_{\leq\tau}, \quad P, \quad N$$

before any stopped-value or expectation row is exported. No expectation-facing read may inspect a filtration row after τ*, an unbounded stopping branch, a measure completion, an almost-sure quotient, or an unstated integrability family.*

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingCarrier finite filtration nonescape`

Proof. First read the stopped value route. By Definition 383.3, the row V is obtained from the first displayed stopping branch, the matching process row, and the matching finite filtration-window row.

Now restrict the packet to the post-stop truncation. The tail-erasure lemma, Definition 383.8, says that the retained tuple is accepted and keeps the same stopped-value row. Thus the stopped-value read remains inside $\Omega, X_{\leq\tau}, T_{\leq\tau}, \tau, V, F_{\leq\tau}, I_{\leq\tau}, H_{\leq\tau}, C_{\leq\tau}, P, N$.

Next apply the finite expectation window of Definition 383.4. Every expectation-facing read must include the finite probability row, stopped-process rows, stopped value, filtration ledger, integrability row, displayed transports, continuation routes, package provenance, and local certificate. In the truncated packet those are exactly the displayed rows through τ.

Finally use the boundary theorem Definition 383.5. Since post-stop rows after τ are absent and the public boundary exports no unbounded stopping, measure completion, almost-sure quotient, or unstated uniform-integrability principle, the retained finite filtration is the only expectation-facing read surface. □

Lean4 ✓ `BEDC.Derived.OptionalStoppingUp.OptionalStoppingFiniteFiltrationNonescape`

Chapter 384

A Concrete Naming Certificate for $\mathsf{FoldMomentKernel}^{\uparrow}$

The $\mathsf{FoldMomentKernel}^{\uparrow}$ interface is a BEDC-native finite certificate surface.

The $\mathsf{FoldMomentKernel}^{\uparrow}$ interface names a finite collision-moment packet for folded fibers. It records only visible BHist rows for a finite window, fold sources, fiber ledgers, moment indices, collision counts, transport routes, continuation routes, package provenance, and local $\mathsf{NameCert}$ evidence. It does not import analytic spectral data, host counting semantics, quotient fibers, or an external measure space.

The window row W is consumed as the finite envelope surface of chapter 365; $\mathsf{FoldMomentKernel}^{\uparrow}$ adds fold-source, fiber-ledger, moment-index, and collision-count rows on top of that envelope rather than introducing an independent window carrier.

Definition 384.1 (FoldMomentKernel carrier)**.** *A* $\mathsf{FoldMomentKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, F, L, I, Q, T, C, P, N)$$

with the following visible rows:

1. *W is the finite window row naming the inspected source and target positions;*
2. *F is the fold-source row recording the displayed finite sources whose images are compared in the window;*
3. *L is the visible fiber ledger listing the inspected source entries above each displayed target entry;*
4. *I is the moment-index row naming the finite family of moment requests read by consumers;*
5. *Q is the collision-count row family, with one displayed finite count ledger for each inspected target and moment index;*
6. *T records componentwise* hsame *transports for windows, fold sources, fiber ledgers, moment indices, and collision-count rows;*
7. *C records the displayed* Cont *routes by which consumers reach the window, fold-source, fiber-ledger, moment-index, and collision-count rows;*
8. *P is the* Pkg *provenance naming the finite BEDC packets that own the source window, target window, fold relation, ledger rows, and count rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$ *row.*

The carrier contains no host cardinality function, no analytic spectrum, no ambient probability measure, and no quotient of fibers.

Lean4 ✓ `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelTasteGate zero window display`
Lean4 (definition) `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelUp`
Lean4 ✓ `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.FoldMomentKernelUp.taste gate`

Definition 384.2 (FoldMomentKernel classifier)**.** *For accepted* $\mathsf{FoldMomentKernel}^{\uparrow}$ *carriers* K *and* K'*, the classifier* $\mathsf{FoldMomentKernelClassifier}(K, K')$ *compares only the displayed window rows, fold-source rows, fiber-ledger rows, moment-index rows, collision-count rows, componentwise* hsame *transports,* Cont *read routes,* Pkg *provenance, and local* NameCert *evidence of Definition 384.1.*

The classifier does not compare hidden source sets, host counts, analytic moments, spectral values, quotient fibers, or unrecorded fold laws.

Theorem 384.3 (FoldMomentKernel zero-window packet)**.** *The empty source window with the zero target window determines an accepted* $\mathsf{FoldMomentKernel}^{\uparrow}$ *carrier. Its fold-source row is empty, its visible fiber ledger has no source entries, every moment-index read is represented by the displayed zero row, every collision-count row is zero, and all transport, continuation, package, and local naming rows are the corresponding finite identity rows.*

Lean4 ✓ `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelUp zero window packet`
↪ variant `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelZeroWindowPacket`

Proof. Start with the empty displayed window. There are no source entries to fold and no target fiber entries to inspect, so the fiber ledger is the empty finite ledger.

For each moment index carried by the packet, use the zero collision-count row. The count row is visible data in Q, not a host cardinality computation.

The transport and continuation routes are componentwise identity routes on the empty and zero rows. Repacking them with the package provenance and local certificate row gives the stated carrier. □

Theorem 384.4 (FoldMomentKernel NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier inhabitation: Definition 384.3 supplies the empty-source and zero-window packet;*
2. *classifier transport: Definition 384.2 transports windows, fold-source rows, fiber ledgers, moment indices, collision-count rows,* hsame *routes,* Cont *routes, package provenance, and local provenance componentwise;*
3. *fiber-ledger exactness: every visible source entry counted by the packet appears in the displayed fiber ledger* L*, and every ledger entry is tied to the displayed fold-source row* F *and window row* W*;*
4. *moment readback boundary: every moment read factors through* I*,* Q*,* L*,* T*,* C*,* P*, and* N *before reaching any downstream consumer;*
5. *non-escape: the packet exports no host counting operation, analytic moment, spectral theorem, quotient fiber, ambient measure, or unrecorded fold-completeness claim.*

Lean4 ✓ `BEDC.Derived.FoldMomentKernelUp.FoldMomentKernelTasteGate visible rows`

Proof. The carrier row is the finite packet of Definition 384.1, and Definition 384.3 gives a concrete inhabitant. Thus the local certificate has an admitted finite source surface.

Classifier transport is componentwise by Definition 384.2. A displayed hsame transport can move only window, fold-source, fiber-ledger, moment-index, collision-count, route, provenance, and local certificate rows already present in the packet.

Fiber-ledger exactness is obtained by projecting L together with F and W. Every counted source entry is read from that ledger, and every ledger entry is attached to the displayed fold-source and window rows by the recorded continuation routes.

Moment readback is a boundary statement about the visible rows I and Q. The consumer can inspect the requested moment only through the moment-index row, the associated collision-count ledger, the fiber ledger, transports, continuations, package provenance, and local certificate row.

Since those rows exhaust the carrier coordinates, there is no remaining field from which host counting, analytic spectral information, quotient fibers, ambient measure, or a hidden completeness principle could be exported. □

Theorem 384.5 (FoldMomentKernel standard bridge surface)**.** *The public finite* $\mathsf{FoldMomentKernel}^{\uparrow}$ *packet bridges to the standard finite collision-moment reading through exactly the displayed window, fold-source, fiber-ledger, moment-index, collision-count,* hsame*,* Cont*,* Pkg*,* NameCert*, and TasteGate rows of Definition 384.1. The bridge exports no host counting operation, quotient fiber, ambient measure space, analytic spectrum, or unrecorded fold law.*

Proof. Start from the carrier and classifier rows. The standard finite collision-moment reading can inspect only W, F, L, I, and Q, and transport between accepted packets is governed by the displayed rows T and C.

Apply the NameCert obligation surface. Fiber-ledger exactness ties counted source entries to L, F, and W, while moment readback ties each requested moment to I, Q, L, T, C, P, and N.

Finally use the TasteGate rows already recorded for the packet. They identify the visible rows used by the checked finite reading, so the standard bridge surface is exhausted by the named carrier coordinates and cannot export host counting, quotient fibers, ambient measure, analytic spectra, or hidden fold laws. □

Closure status for $\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FoldMomentKernel}^{\uparrow}$ packet is generated from finite window rows, fold-source rows, visible fiber ledgers, moment-index rows, collision-count rows, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped treatment binds the finite window, fold-source, fiber-ledger, moment-index, collision-count, hsame, Cont, Pkg, NameCert, and TasteGate rows certified at Definition 384.4, with instance BEDC.Derived.FoldMomentKernelUp.taste_gate

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FoldMomentKernelUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close analytic spectra, host cardinality, quotient fibers, ambient measure theory, collision asymptotics, or bridge checking.

Upgrade path: Remaining work is formal-axis strengthening of the existing TasteGate surface, not paper-side closure.

Theorem 384.6 (FoldMomentKernel finite-window restriction carrier)**.** *Let*

$$K = (W, F, L, I, Q, T, C, P, N)$$

be an accepted $\mathsf{FoldMomentKernel}^{\uparrow}$ *carrier, and let* Ω *be a displayed finite subwindow of the window row* W*. Suppose the packet has coherent restriction rows*

$$W_\Omega, F_\Omega, L_\Omega, I_\Omega, Q_\Omega, T_\Omega, C_\Omega, P_\Omega, N_\Omega$$

with the following properties:

1. W_Ω *is the selected finite source-and-target subwindow of* W*;*
2. F_Ω *contains exactly the fold-source entries of* F *whose visible source or target addresses lie in* W_Ω*;*
3. L_Ω *contains exactly the fiber-ledger entries of* L *above the displayed target entries of* W_Ω *and using fold sources from* F_Ω*;*
4. I_Ω *is the displayed finite family of moment requests retained for the subwindow;*
5. Q_Ω *contains exactly the collision-count ledgers of* Q *whose target and moment-index addresses lie in* W_Ω *and* I_Ω*;*
6. T_Ω *and* C_Ω *are the componentwise restrictions of the* hsame *transport rows and* Cont *read routes whose endpoints are among* W_Ω*,* F_Ω*,* L_Ω*,* I_Ω*, and* Q_Ω*;*
7. P_Ω *names precisely the package provenance rows consumed by the retained window, fold-source, ledger, moment-index, and count rows;*
8. N_Ω *is the local* $\mathsf{NameCert}_{\mathsf{FoldMomentKernel}^{\uparrow}}$ *row restricted to those same retained coordinates.*

Then

$$K_\Omega = (W_\Omega, F_\Omega, L_\Omega, I_\Omega, Q_\Omega, T_\Omega, C_\Omega, P_\Omega, N_\Omega)$$

is an accepted $\mathsf{FoldMomentKernel}^{\uparrow}$ *carrier.*

Proof. Unfold the carrier of Definition 384.1. A $\mathsf{FoldMomentKernel}^{\uparrow}$ carrier has only the nine displayed coordinates

$$W, F, L, I, Q, T, C, P, N.$$

The hypotheses provide a retained row for each of these coordinates over the finite subwindow Ω. Thus K_Ω has the required window, fold-source, fiber-ledger, moment-index, collision-count, transport, continuation, provenance, and local naming entries.

The fiber and count rows remain well-formed because the restriction is coordinatewise coherent. Every source entry in L_Ω is one already listed by the retained fold-source row F_Ω, and every retained count row in Q_Ω is indexed by a retained target entry of W_Ω and a retained moment request of I_Ω. Conversely, the hypotheses discard only rows whose visible addresses lie outside the selected subwindow, so no retained ledger entry loses its displayed source, target, or moment-index address.

The transport and readback rows also remain internal to the restricted packet. By the classifier scope of Definition 384.2, a $\mathsf{FoldMomentKernelClassifier}$ comparison may inspect only displayed window rows, fold-source rows, fiber-ledger rows, moment-index rows, collision-count rows, componentwise hsame transports, Cont routes, package provenance, and local $\mathsf{NameCert}$ evidence. The rows T_Ω and C_Ω are exactly the retained transports and routes with endpoints in those restricted coordinates, so every classifier or consumer read in the subwindow still factors through visible rows of K_Ω.

Finally apply the obligation surface of Definition 384.4. Carrier inhabitation is supplied by the displayed restricted packet itself, classifier transport is the componentwise restricted transport just described, fiber-ledger exactness is the retained F_Ω–L_Ω–W_Ω incidence, moment readback factors

through I_Ω, Q_Ω, L_Ω, T_Ω, C_Ω, P_Ω, and N_Ω, and non-escape follows because Definition 384.1 gives no coordinate for host cardinality, analytic spectra, quotient fibers, ambient measure, or unrecorded fold laws. Repacking these retained rows gives the accepted carrier K_Ω. $\square$

Chapter 385

A Concrete Naming Certificate for $\mathsf{FiniteErrorBudget}^{\uparrow}$

The $\mathsf{FiniteErrorBudget}^{\uparrow}$ interface names the finite precision-budget packet used on the $\mathsf{Real}^{\uparrow}$ and completion path. Following Bishop and Bridges, *Constructive Analysis*, it treats accuracy as visible dyadic tolerance data, diagonal scheduling, tail ledgers, finite budget consumption, rational regular-sequence readback, and a real-name seal. The packet sits above $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DiagonalModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{FastSeriesProduct}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$. It never forms quotient streams, chooses a host limit, compares host real equality, or appeals to ambient completeness.

Definition 385.1 (FiniteErrorBudget carrier)**.** *A* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (n, \epsilon, b, d, t, r, e, P, N)$$

with the following visible rows:

1. *n is the unary precision-request row read by downstream real and completion consumers;*
2. *ϵ is the dyadic tolerance row, displayed as a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius for the request n;*
3. *b is the* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector row naming the finite diagonal index schedule used to answer the request;*
4. *d is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tail ledger recording the tail estimate available at the selected schedule;*
5. *t is the finite budget-consumer row, either a* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet or a* $\mathsf{FastSeriesProduct}^{\uparrow}$ *packet, whose summand and convolution windows are bounded by d;*
6. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational name reached by the selected budget;*
7. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row exported to* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *after the finite request is answered;*
8. *P is the* Pkg *provenance naming the dyadic tolerance, diagonal selector, tail ledger, budget consumer, regular-sequence readback, and real seal packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *row.*

The carrier contains no quotient of regular streams, no host real equality test, no selected infinite limit, and no ambient completeness principle.

Lean4 (definition) BEDC.Derived.FiniteErrorBudgetUp.TasteGate.FiniteErrorBudgetTasteGate single carrier alignment
Lean4 (definition) BEDC.Derived.FiniteErrorBudgetUp.TasteGate.FiniteErrorBudgetUp
Lean4 ✓ BEDC.Derived.FiniteErrorBudgetUp.TasteGate.FiniteErrorBudgetTasteGate single carrier alignment

When the finite budget-consumer coordinate t represents a product budget, the row is imported from the finite convolution surface of chapter 368. In that case $\mathsf{FiniteErrorBudget}^{\uparrow}$ consumes the sibling packet's finite convolution rows, pasted tail schedules, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal as one visible budget row. The local packet then repacks that row through its own dyadic tolerance, diagonal selector, tail ledger, package provenance, and $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ coordinates.

Definition 385.2 (FiniteErrorBudget classifier)**.** *For accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carriers* E *and* E'*, the classifier* $\mathsf{FiniteErrorBudgetClassifier}(E, E')$ *compares only the displayed precision-request rows, dyadic tolerance rows,* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector rows,* $\mathsf{DyadicTailBound}^{\uparrow}$ *tail ledgers, finite budget-consumer rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* Pkg *provenance, and local* $\mathsf{NameCert}$ *evidence of Definition 385.1.*

Classifier transport is componentwise by hsame *on the visible rows and by* Cont *along the recorded consumer routes. The classifier does not compare hidden streams, host real values, chosen limits, quotient classes, or ambient completion objects.*

Theorem 385.3 (FiniteErrorBudget zero seed)**.** *The unary zero precision request with dyadic tolerance* 1*, the initial* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector, the zero* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger, the empty* $\mathsf{FastConvergentSeries}^{\uparrow}$ *budget consumer, the constant-zero* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the zero* $\mathsf{Real}^{\uparrow}$ *seal determine an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier.*

Proof. Use the zero request as the displayed precision row and choose the dyadic unit row as the tolerance. The initial diagonal selector and the zero tail ledger give a finite schedule whose tail cost is already below the displayed budget.

The empty fast-convergent-series packet consumes no summands, so its finite window is compatible with the zero tail ledger. Its regular rational readback is the constant-zero $\mathsf{RegSeqRat}^{\uparrow}$ name, and the real seal is the corresponding zero $\mathsf{Real}^{\uparrow}$ row.

All transports and continuations are identity rows on the displayed packet coordinates. Repacking these rows with the package provenance and local certificate row gives the required carrier. □

Theorem 385.4 (FiniteErrorBudget diagonal handoff)**.** *Every accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier* $E = (n, \epsilon, b, d, t, r, e, P, N)$ *exports a finite handoff into* $\mathsf{RealCauchyCompletion}^{\uparrow}$*: the request* n *is answered through the dyadic tolerance* ϵ*, the selector* b*, the tail ledger* d*, the budget consumer* t*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, and the* $\mathsf{Real}^{\uparrow}$ *seal* e *before any completion consumer can inspect the result.*

The handoff is stable under the classifier of Definition 385.2 and exports no quotient stream, host real equality, selected limit, or ambient completeness claim.

Proof. Start with the visible request row n. The dyadic tolerance row ϵ fixes the precision budget that can be consumed by the downstream completion read.

The diagonal selector b chooses a finite schedule, and the tail ledger d records the bound available at that schedule. The budget consumer t can be read only through that displayed schedule and ledger, so every finite summand or convolution window remains inside the packet.

The readback row r presents the resulting regular rational name, and the seal row e is the only row handed to $\mathsf{RealCauchyCompletion}^{\uparrow}$. Componentwise hsame transport and the displayed Cont routes preserve each of these coordinates, so classified replacements give the same handoff surface.

Since the carrier coordinates are exhausted by these visible rows, there is no field from which a quotient stream, host real equality test, selected infinite limit, or ambient completeness principle could be exported. □

Theorem 385.5 (FiniteErrorBudget tail-ledger consumption exactness)**.** *For every accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier* $E = (n, \epsilon, b, d, t, r, e, P, N)$*, every finite budget-consumer read used by the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff factors through the displayed* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger* d*, the* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector* b*, the dyadic tolerance* ϵ*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r*.*

If two accepted carriers classify together by Definition 385.2, then the consumer-visible ledger surface is preserved exactly: classified replacement can transport the same displayed rows through hsame *and* Cont*, but it cannot replace the handoff by an unrecorded stream tail, selected limit, host real equality, or ambient completeness datum.*

Proof. Project the selector row b and the tail ledger d from the carrier. The selector names the finite schedule for the request n, while d records the tail estimate available at that schedule against the displayed dyadic tolerance ϵ.

Project the budget-consumer row t. By the carrier definition, its summand or convolution windows are read only inside the schedule and tail bound already displayed by b and d, and the resulting rational name is exposed only through the readback coordinate r.

Classifier transport is componentwise. Therefore a classified carrier can transport only the precision row, tolerance row, selector, tail ledger, consumer row, readback row, seal row, package provenance, and certificate row already present in the packet.

Finally apply Definition 385.4. The exported completion surface is the path through ϵ, b, d, t, r, and e, so no unrecorded stream tail, selected limit, host equality test, or ambient completion datum is available to the consumer. □

Theorem 385.6 (FiniteErrorBudget classifier stability)**.** *Let* $E = (n, \epsilon, b, d, t, r, e, P, N)$ *and* $E' = (n', \epsilon', b', d', t', r', e', P', N')$ *be accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carriers. If they classify together by Definition 385.2, then the consumer-visible rows* $n, \epsilon, b, d, t, r, e, P, N$ *transport componentwise to* $n', \epsilon', b', d', t', r', e', P', N'$ *through the displayed* hsame *rows and the displayed* Cont *routes of the classifier.*

The transport preserves the precision request, dyadic tolerance, diagonal selector, tail ledger, finite budget consumer, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, package provenance, and local naming row as the complete visible surface. It introduces no hidden stream coordinate, host equality coordinate, selected limit, or ambient completeness row.*

Proof. Project the two accepted carriers. Their coordinates are exactly the rows named in Definition 385.1, so there is no further coordinate available for transport.

Now apply the classifier of Definition 385.2. Its clauses are componentwise hsame comparisons together with displayed Cont routes for the finite precision-budget reads, so each visible row of E is matched with the corresponding visible row of E'.

The route list is exhausted by $n, \epsilon, b, d, t, r, e, P, N$. Since neither carrier definition nor classifier definition contains a stream representative, host equality field, selected limit, or completeness witness, such data cannot be exported by classifier transport. □

Theorem 385.7 (FiniteErrorBudget zero-budget stability)**.** *The zero-budget seed of Definition 385.3 remains an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet under every classifier-preserving precision-row transport. The transported seed still consists only of the zero precision request, dyadic unit tolerance, initial diagonal selector, zero tail ledger, empty budget consumer, constant-zero* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, zero* $\mathsf{Real}^{\uparrow}$ *seal, package provenance, and local* NameCert *row.*

Proof. Start from the seed packet constructed in Definition 385.3. Each of its coordinates is one of the visible carrier rows of Definition 385.1.

Apply Definition 385.6 to the transported precision row. The classifier can move the zero request only through the displayed hsame and Cont rows, and the remaining tolerance, selector, ledger, consumer, readback, seal, provenance, and naming coordinates move with that same finite surface.

The resulting packet is accepted because the carrier predicates are the same componentwise rows used by the seed construction. No unrecorded stream tail, host real equality test, selected limit, or completeness witness is introduced by the transport. □

Theorem 385.8 (FiniteErrorBudget public finite-packet export)**.** *For every accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier* $E = (n, \epsilon, b, d, t, r, e, P, N)$*, the diagonal handoff of Definition 385.4, the tail-ledger consumption exactness of Definition 385.5, and the local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *rows of Definition 385.9 assemble a public finite packet export for* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Every downstream read factors through the displayed rows $n, \epsilon, b, d, t, r, e, P, N$*, componentwise* hsame *transport, and displayed* Cont *routes. The export contains no quotient stream, hidden host equality coordinate, selected limit, or ambient completeness principle.*

Proof. First project the carrier rows. The request n, tolerance ϵ, selector b, ledger d, budget consumer t, readback r, seal e, package row P, and naming row N are the whole finite packet.

Next apply Definition 385.4. It identifies the completion-facing route through the tolerance, selector, ledger, consumer, readback, and seal rows.

Apply Definition 385.5 to the same packet. This makes every finite consumer read pass through the displayed ledger and prevents replacement by an unrecorded tail surface.

Finally use Definition 385.9. Its carrier, classifier, zero-stability, diagonal-handoff, and non-escape rows repack the same finite data as the public export consumed by $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Theorem 385.9 (FiniteErrorBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier habitation by the zero-budget seed of Definition 385.3;*
2. *classifier transport by componentwise* hsame *and the displayed* Cont *routes of Definition 385.2;*
3. *zero-budget stability, asserting that the zero request, dyadic unit tolerance, initial selector, zero tail ledger, empty budget consumer, constant-zero readback, and zero real seal remain inside the carrier under classified replacement;*
4. *diagonal handoff into* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *by Definition 385.4;*
5. *non-escape of quotient streams, host real equality, selected limits, and ambient completeness beyond the displayed precision-budget rows.*

Proof. Carrier habitation is exactly the packet constructed in Definition 385.3. It supplies every carrier coordinate using visible finite rows.

Classifier transport follows from Definition 385.2. A classified replacement can move only the precision request, dyadic tolerance, diagonal selector, tail ledger, budget consumer, readback, seal, provenance, and local certificate rows that the carrier already displays.

Zero-budget stability is obtained by projecting the same coordinates from the zero seed after transport. Identity hsame rows preserve the zero request, unit tolerance, initial selector, zero tail ledger, empty consumer, constant-zero readback, and zero real seal.

The diagonal handoff row is supplied by Definition 385.4. Its exported surface is the finite path through ϵ, b, d, t, r, and e, so the completion consumer sees a real-seal read and not an unrecorded stream.

The non-escape row follows because all accepted carrier coordinates have been listed. None of them is a quotient constructor, a host equality oracle, a choice of limit, or an ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.FiniteErrorBudgetUp.FiniteErrorBudgetCarrier namecert obligations`

Theorem 385.10 (FiniteErrorBudget classifier row stability)**.** *Let $E = (n, \epsilon, b, d, t, r, e, P, N)$ and $E' = (n', \epsilon', b', d', t', r', e', P', N')$ be accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carriers. If* $\mathsf{FiniteErrorBudgetClassifier}(E, E')$ *holds, then classifier transport preserves exactly the precision request, dyadic tolerance, diagonal selector, tail ledger, budget consumer,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, package row, and local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *row displayed by Definition 385.1. No consumer-visible row outside that list is transported.*

Proof. Project the two carrier tuples using Definition 385.1. The visible rows are precisely $n, \epsilon, b, d, t, r, e, P, N$ and their primed counterparts.

Apply the classifier definition of Definition 385.2. Its transport clauses are componentwise hsame rows and displayed Cont routes for the same finite rows, so every transported consumer read remains on the listed surface.

The diagonal and tail-consumption rows then rule out hidden enlargement. Definition 385.4 routes completion reads through the displayed selector and seal path, while Definition 385.5 keeps the consumer-facing budget row attached to the displayed ledger. □

Theorem 385.11 (FiniteErrorBudget diagonal selector exactness)**.** *For every accepted carrier $E = (n, \epsilon, b, d, t, r, e, P, N)$, every diagonal handoff from* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *into* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *factors through the displayed selector b, tail ledger d, tolerance ϵ, finite budget consumer t, and readback row r before reaching the* $\mathsf{Real}^{\uparrow}$ *seal e. The handoff cannot replace any of these rows by an unrecorded schedule, tail estimate, tolerance, consumer, readback, or seal.*

Proof. Start from the carrier coordinates of Definition 385.1. The request row n exposes only one dyadic tolerance row, one diagonal selector, one tail ledger, one consumer, one readback, and one seal.

Apply Definition 385.4. It identifies the completion-facing path as the chain through ϵ, b, d, t, r, and e.

Use Definition 385.5 for the consumer step. The consumer read is bound to the same displayed selector and ledger, so diagonal handoff exactness is a statement about the packet itself rather than about an ambient completion object. □

Theorem 385.12 (FiniteErrorBudget completion-consumer boundary)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet sees only the displayed finite packet $n, \epsilon, b, d, t, r, e, P, N$, together with the classifier transports and continuation routes of Definition 385.2. Such a consumer cannot read quotient streams, host real equality, selected limits, or ambient completeness.*

Proof. First identify the public packet by Definition 385.1. These coordinates are the complete carrier surface presented to downstream consumers.

Next apply Definition 385.4 and Definition 385.5. Together they force every completion-facing read through the selector, tolerance, ledger, consumer, readback, and seal rows already displayed in the carrier.

Finally use Definition 385.9. Its local certificate rows package the same finite surface and state the non-escape boundary, so the completion consumer has no route to quotient streams, host real equality, selected limits, or ambient completeness. □

Definition 385.13 (FiniteErrorBudget scoped dependency tuple)**.** *For an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *carrier, the scoped dependency tuple is the displayed finite packet*

$$(n, \epsilon, b, d, t, r, e, P, N).$$

It is read as a tuple over the following BEDC rows:

1. n *is the precision request carried in* BHist *and read through* $\mathsf{StreamName}^{\uparrow}$ *naming discipline;*
2. ϵ *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row;*
3. b *is the* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector row;*
4. d *is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger row;*
5. t *is the finite budget-consumer row;*
6. r *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row;*
7. e *is the* $\mathsf{Real}^{\uparrow}$ *seal row;*
8. P *is the* Pkg *provenance row; and*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *row.*

Classifier movement of this tuple is only by the displayed hsame *rows and the displayed* Cont *routes of Definition 385.2. The tuple has no coordinate for a quotient stream, host real equality, selected limit, or ambient completeness.*

Theorem 385.14 (FiniteErrorBudget dyadic tolerance scope)**.** *For every accepted carrier* $E = (n, \epsilon, b, d, t, r, e, P, N)$*, the dyadic tolerance* ϵ*, diagonal selector* b*, and tail ledger* d *are consumed through the scoped dependency tuple of Definition 385.13. Any completion handoff or budget-consumer read that uses the tolerance must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row,* $\mathsf{DiagonalModulus}^{\uparrow}$ *row,* $\mathsf{DyadicTailBound}^{\uparrow}$ *row,* BHist *packet,* hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *evidence already listed there.*

Proof. Start with the carrier decomposition of Definition 385.1. The only dyadic tolerance row is ϵ, the only diagonal selector row is b, and the only tail ledger row is d.

Apply Definition 385.4. It sends every completion-facing request through ϵ, b, and d before any readback or real seal is presented.

Next apply Definition 385.5. The finite budget consumer is tied to the same selector and ledger, so the tail estimate consumed by the request cannot be replaced by an unrecorded schedule or ambient bound.

Finally use Definition 385.10 and Definition 385.8. Classifier movement is by the displayed hsame and Cont rows, and the public export repacks only the same finite packet with its Pkg and $\mathsf{NameCert}$ rows. □

Theorem 385.15 (FiniteErrorBudget RegSeqRat readback scope)**.** *For every accepted carrier $E = (n, \epsilon, b, d, t, r, e, P, N)$, the finite budget consumer t and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r factor through the same scoped dependency tuple before the* $\mathsf{Real}^{\uparrow}$ *seal e is exported. Consequently the readback seen by* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *is determined by the displayed finite packet and not by a hidden stream representative, host real equality, selected limit, or ambient completeness principle.*

Proof. Project the scoped tuple using Definition 385.13. The budget consumer t, readback row r, and real seal e are coordinates of the same finite BHist packet as the request, tolerance, selector, and ledger.

Apply Definition 385.5. This places every consumer read inside the displayed selector and tail-ledger surface and exposes its rational output only through r.

Then apply Definition 385.11. The handoff path reaches e only after passing through ϵ, b, d, t, and r, so the $\mathsf{Real}^{\uparrow}$ seal is downstream of the finite packet rather than an independent completion witness.

The public export theorem, Definition 385.8, packages the same readback for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$. Therefore the completion consumer cannot obtain any extra readback surface outside the scoped tuple. □

Theorem 385.16 (FiniteErrorBudget NameCert dependency exhaustion)**.** *For every accepted carrier $E = (n, \epsilon, b, d, t, r, e, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$ *dependency surface is exhausted by the scoped dependency tuple of Definition 385.13. The carrier row, classifier row, diagonal handoff row, tail-ledger consumption row, public export row, and completion-consumer boundary row all read only $n, \epsilon, b, d, t, r, e, P, N$ together with their displayed* hsame *and* Cont *routes.*

Proof. Use Definition 385.9 to enumerate the local certificate rows. Each row names carrier habitation, classifier transport, zero-budget stability, diagonal handoff, and non-escape for the visible finite packet.

The carrier and classifier surfaces are exactly the rows of Definition 385.1 and Definition 385.2. These give $n, \epsilon, b, d, t, r, e, P, N$ as the complete coordinate list and allow movement only through the displayed hsame and Cont routes.

The diagonal, tail-ledger, and public-export clauses are supplied by Definition 385.4, Definition 385.5, and Definition 385.8. They reuse the same tuple for the tolerance, selector, ledger, consumer, readback, real seal, package row, and naming row.

Finally apply Definition 385.12. Since every completion consumer is already confined to the displayed finite packet, the local $\mathsf{NameCert}$ has no remaining dependency slot through which a quotient stream, host real equality, selected limit, or ambient completeness principle could enter. □

Theorem 385.17 (FiniteErrorBudget dyadic request scope)**.** *Every accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet routes its precision request n, dyadic tolerance ϵ, diagonal selector b, and tail ledger d through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows of Definition 385.13. No dyadic request can be consumed through a row outside the finite carrier and classifier surface of the packet.*

Proof. Use Definition 385.1 to decompose the accepted packet as $E = (n, \epsilon, b, d, t, r, e, P, N)$. The precision request, tolerance, selector, and ledger are named coordinates of this carrier, with P and N recording the package provenance and local naming evidence.

Apply Definition 385.2 and Definition 385.10. They restrict classifier movement to the displayed hsame transports and Cont routes, so the dyadic rows cannot be replaced by an ambient tolerance or an unlisted tail estimate.

Finally use Definition 385.11. The selector and tail ledger are exact for the same finite request and dyadic tolerance, which binds the request-facing part of the packet to the scoped dependency tuple. □

Theorem 385.18 (FiniteErrorBudget StreamName readback scope)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet can be consumed only after the finite window, selector, tail ledger, and dyadic observations have been routed through the same finite packet. In particular, the* $\mathsf{StreamName}^{\uparrow}$ *readback path is downstream of the packet coordinates* $n, \epsilon, b, d, t, r, e, P, N$ *and not of a hidden stream representative.*

Proof. Start from Definition 385.5. It ties the consumer row t to the displayed selector and tail ledger, so a readback request has to pass through the finite window and dyadic observations already recorded in the packet.

Next apply Definition 385.4. The handoff places the readback row r after the tolerance, selector, and ledger rows, which makes the $\mathsf{RegSeqRat}^{\uparrow}$ readback a consequence of the same finite packet.

The public export theorem, Definition 385.8, repacks exactly this packet for downstream use. Hence a $\mathsf{StreamName}^{\uparrow}$ consumer has no separate readback route outside the displayed packet surface. □

Theorem 385.19 (FiniteErrorBudget real completion seal scope)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal-facing read of an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet factors through the finite request, tolerance, selector, tail ledger, budget consumer, readback, seal, transport, continuation, package, and naming rows before any consumer can cite it. Thus completion use is scoped to the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Proof. Apply Definition 385.8. It exposes the finite packet to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ only through the coordinates already listed in the scoped dependency tuple.

Use Definition 385.12. This boundary forces every completion-facing consumer to pass through the request, tolerance, selector, ledger, budget consumer, readback, and seal rows before the consumer can cite the real-completion surface.

Finally apply Definition 385.9. The local NameCert rows record the package, transport, continuation, and naming evidence for the same finite packet, so the seal-facing read has no remaining route through quotient streams, host real equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FiniteErrorBudgetUp.FiniteErrorBudgetUp StdBridge`

Theorem 385.20 (FiniteErrorBudget finite-row induction)**.** *Let* $E = (n, \epsilon, b, d, t, r, e, P, N)$ *be an accepted* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of* E *is obtained by finite induction over the displayed route*

$$n, \epsilon, b, d, t, r, e, P, N.$$

The base row is the zero seed of Definition 385.3; the step rows pass through the diagonal selector, tail-ledger exactness, budget-consumer boundary, readback, seal, package, and naming rows; and the terminal handoff is the diagonal handoff of Definition 385.4. No induction step may introduce a quotient stream, selected limit, host real equality, countable choice, or ambient completeness coordinate outside the displayed carrier.

Proof. Project the carrier of Definition 385.1. The zero precision row is the seed supplied by Definition 385.3, so the base case already lies inside the finite packet.

For the induction step, read the rows in order. The tolerance and selector rows are controlled by Definition 385.11; the tail row is controlled by Definition 385.5; and the consumer boundary is the same finite boundary used in Definition 385.12.

The classifier supplies the preservation invariant. By Definition 385.2 and Definition 385.10, each componentwise hsame transport and displayed Cont replay preserves the same packet route, so the induction hypothesis is transported only along named rows.

Finally apply Definition 385.4. It hands the readback and completion-facing rows to the public consumer after the finite selector and tail ledger have been displayed. The non-escape clauses in Definition 385.19 exclude quotient streams, selected limits, host real equality, countable choice, and ambient completeness, closing the finite-row induction. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteErrorBudget}^{\uparrow}$ packet is generated from a unary precision request, a dyadic tolerance row, a $\mathsf{DiagonalModulus}^{\uparrow}$ selector, a $\mathsf{DyadicTailBound}^{\uparrow}$ ledger, a finite $\mathsf{FastConvergentSeries}^{\uparrow}$ or $\mathsf{FastSeriesProduct}^{\uparrow}$ budget consumer, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 385.8, Definition 385.17, Definition 385.18, Definition 385.19, and Definition 385.9 bind the public finite-packet export to its dyadic request, $\mathsf{StreamName}^{\uparrow}$ readback, real-completion seal, package, continuation, and naming routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteErrorBudget}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.FiniteErrorBudgetUp.FiniteErrorBudgetCarrier\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient streams, host real equality, selected limits, ambient real completeness, or bridge checking.

Upgrade path: No further upgrade remains on the formal bridge axis.

The $\mathsf{CauchyLimitSeal}^{\uparrow}$ hub supplies the terminal seal consumed by chapter 1205 through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows; this route adds no quotient-stream equality, host limit, selected representative, choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 386

A Concrete Naming Certificate for $\mathsf{CauchyLimitSeal}^{\uparrow}$

The $\mathsf{CauchyLimitSeal}^{\uparrow}$ interface records the visible seal boundary between regular rational Cauchy data and the consumer-facing $\mathsf{Real}^{\uparrow}$ row. In the Bishop–Bridges constructive-analysis reading of Cauchy completion, a limit is presented by finite regular-name observations and an explicit schedule, not by quotienting streams or selecting a host limit. The chapter isolates that boundary as a concrete packet so later $\mathsf{Real}^{\uparrow}$ consumers can read only the displayed finite windows that pass through the seal.

The same seal boundary is consumed as the downstream finite route of the diagonal-limit construction in chapter 450. That consumer reads the StreamName, RegSeqRat, DyadicRatCore, and RealUp rows already named by the carrier, so the seal appears as a horizontal lattice edge between the finite completion route and the public real-facing endpoint.

Definition 386.1 (CauchyLimitSeal carrier)**.** *A* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, S, D, G, E, T, P, N)$$

with the following visible coordinates:

1. *X is a* $\mathsf{RegSeqRat}^{\uparrow}$ *family row presenting the regular rational Cauchy source to be sealed;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that lists the finite windows a consumer may inspect;*
3. *D is the* $\mathsf{DyadicRatCore}$ *observation ledger attached to those scheduled windows;*
4. *G is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row that routes the regular family through its displayed completion packet;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed to downstream real consumers;*
6. *T is the finite collection of* hsame *and* Cont *transport rows connecting the source family, scheduled windows, diagonal row, observation ledger, and seal row;*
7. *P is the* Pkg *provenance row naming the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependency certificates;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *row.*

The local classifier compares two carriers only by componentwise transport of these visible rows. It does not compare quotient representatives, host limits, or ambient completeness data.

Theorem 386.2 (CauchyLimitSeal NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *surface has four finite obligations: carrier habitation by Definition 386.1; classifier stability under the displayed* hsame *and* Cont *transport rows; exact factorization of every* $\mathsf{Real}^{\uparrow}$ *consumer read through the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, DyadicRatCore, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *rows; and non-escape of quotient stream equality, selected host limits, uniqueness theorems, and ambient real completeness data.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier namecert obligations`

Proof. Carrier habitation is the finite packet assembly. The regular family, schedule, observation ledger, completion row, seal row, transport rows, package provenance, and local certificate are all coordinates of the displayed BHist packet.

Classifier stability is componentwise. Each row is transported only by the stored hsame comparison and the corresponding Cont route, so an accepted packet stays inside the same visible dependency surface.

The exactness and boundary obligations follow from the shape of the carrier. Every $\mathsf{Real}^{\uparrow}$ read must pass through the scheduled finite windows and their dyadic observation ledger before reaching the completion row and seal row; no field contains a quotient stream, a selected host limit, a uniqueness theorem, or an ambient real-completeness datum. □

Theorem 386.3 (CauchyLimitSeal window factorization)**.** *For every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier* L *and every finite* $\mathsf{Real}^{\uparrow}$ *consumer request, the read factors through a visible* $\mathsf{StreamName}^{\uparrow}$ *window from* S*, the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *source row in* X*, the matching* DyadicRatCore *observation ledger in* D*, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row* G*, and the displayed transport rows in* T *before it reaches the seal row* E*.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealPacket window factorization`

Proof. Project the carrier of Definition 386.1. A consumer request first selects one of the scheduled finite windows because the seal row has no independent read channel.

Read the selected window through the $\mathsf{RegSeqRat}^{\uparrow}$ source family. The rational observation is recorded in the DyadicRatCore ledger, and the same scheduled row is passed to the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal coordinate.

Finally apply the stored hsame and Cont rows in T to transport the read to the $\mathsf{Real}^{\uparrow}$ seal. The path is therefore the stated finite factorization, with every consumer-facing value mediated by visible packet data. □

Theorem 386.4 (CauchyLimitSeal non-escape boundary)**.** *An accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier exports no quotient stream, no selected host limit, no uniqueness theorem for limits, and no ambient real completeness datum. Its public surface is exhausted by the finite* BHist *packet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source family, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* DyadicRatCore *observation ledger, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* hsame, Cont, Pkg, *and* NameCert *provenance.*

Proof. Inspect the carrier coordinates. Each coordinate is either a named dependency certificate row, a finite transport row, a package provenance row, or the local certificate row.

The completion coordinate is consumed only through its displayed diagonal row, and the real-facing coordinate is the seal row attached to that diagonal route. Neither coordinate adds a host real carrier or a quotient of regular streams.

Thus a downstream consumer can cite exactly the listed rows. Since no other coordinate exists in the packet, uniqueness theorems, selected limits, and ambient real completeness data cannot be exported by the $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface. □

Theorem 386.5 (CauchyLimitSeal RealUp consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier may read only the sealed finite-window route of Definition 386.3. In particular, the consumer boundary is the displayed packet surface over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$, *and it does not expose an independent completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier realup consumer boundary`

Proof. Begin with the factorization theorem. It forces each consumer read through the scheduled window, regular rational source row, dyadic observation ledger, completion diagonal row, and transport route before the read reaches the $\mathsf{Real}^{\uparrow}$ seal.

Use the non-escape boundary to identify what remains visible after the seal is reached. The package provenance and local certificate name the dependency rows, but they do not add quotient, host-limit, uniqueness, or ambient-completeness fields.

Therefore the $\mathsf{Real}^{\uparrow}$ consumer boundary is precisely the finite route listed in the statement. The seal is a public endpoint for displayed regular-Cauchy data, not a source of an independent real-completeness principle. □

Definition 386.6 (CauchyLimitSeal public route packet)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier* $L = (X, S, D, G, E, T, P, N)$, *its public route packet is the componentwise projection*

$$\mathsf{PubRoute}(L) = (X, S, D, G, E, T, P, N)$$

read as the visible $\mathsf{RegSeqRat}^{\uparrow}$ *source row, scheduled* $\mathsf{StreamName}^{\uparrow}$ *window family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* hsame *and* Cont *transport ledger,* Pkg *provenance row, and local* $\mathsf{NameCert}$ *row. The packet is a named public BEDC object: it contains only these carrier coordinates and no quotient stream, selected host limit, uniqueness theorem, host real construction, or ambient completeness datum.*

Theorem 386.7 (CauchyLimitSeal public route transport)**.** *Local classifier transport of accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carriers transports the public route packet componentwise. The transported packet preserves the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and it introduces no quotient stream, selected host limit, uniqueness theorem, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier public route transport`

Proof. Project the carrier $L = (X, S, D, G, E, T, P, N)$ and read Definition 386.6. The public route packet is exactly the visible coordinate list of the carrier, so classifier transport can act only on those coordinates.

Apply the carrier obligation theorem Definition 386.2. Its classifier-stability clause moves accepted carriers along the stored hsame comparisons and Cont routes, while the Pkg and $\mathsf{NameCert}$ rows keep the dependency names fixed.

Use the non-escape boundary Definition 386.4. Because the carrier contains no quotient stream, selected host limit, uniqueness theorem, or ambient completeness field, componentwise transport cannot create any such public coordinate. □

Theorem 386.8 (CauchyLimitSeal public route boundary)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *or completion-consumer route of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier factors through the public route packet of Definition 386.6. Conversely, every row of that packet is one of the displayed rows named by the finite-window factorization of Definition 386.3 and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary of Definition 386.5.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier public route boundary`

Proof. For a $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 386.5. The read is forced through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row, $\mathsf{Real}^{\uparrow}$ seal row, and visible transport, package, and naming rows.

For a completion-facing consumer, use the finite-window factorization Definition 386.3. It gives the same route before the read reaches the seal, so the completion-facing path also lands inside the packet coordinates X, S, D, G, E, T, P, N.

The reverse containment is just projection from Definition 386.6. Each packet row is one of the displayed carrier rows named by the two boundary theorems, and Definition 386.4 rules out any additional public row outside that list. □

Theorem 386.9 (CauchyLimitSeal tail-agreement public route factorization)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *read or completion-facing read of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet that consumes a paired tail-agreement comparison factors first through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *finite route*

$$R, S, W, D, A, H, C, P, N$$

and then through the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *public route packet*

$$X, S, D, G, E, T, P, N.$$

The combined route uses only $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data. It exports no quotient stream, selected host limit, host real equality, ambient completeness, or choice-selected tail.*

Proof. For the paired agreement side, use the Real–Completion handoff of Definition 699.5. The paired comparison enters through the displayed finite route $R, S, W, D, A, H, C, P, N$ before any downstream real or completion consumer can read it.

Apply window-agreement determinacy from Definition 699.64 and completion-handoff totality from Definition 699.65. These two rows make the agreement route exact for the finite windows and completion-facing handoff, so no second tail, quotient stream, host real equality, or choice-selected comparison is available.

Now pass the resulting consumer read to the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet. By Definition 386.6 and the public route boundary of Definition 386.8, the read factors through the carrier route X, S, D, G, E, T, P, N.

Finally combine the non-escape boundaries of the two chapters. The tail-agreement route contributes only its finite comparison surface, and the CauchyLimitSeal route contributes only the displayed source, schedule, dyadic, diagonal, seal, transport, package, and naming rows. Hence the composite route cannot export quotient streams, selected host limits, host real equality, ambient completeness, or a choice-selected tail. □

Theorem 386.10 (CauchyLimitSeal RegularCauchyName consumer link)**.** *Let a* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet expose its regular schedule, dyadic observation rows, dyadic-radius ledger, finite regularity ledger, and* $\mathsf{Real}^{\uparrow}$ *seal through the public packet named by Definition 371.32 and the completion consumer coverage of Definition 371.33. Repacking those displayed rows as*

$$X, S, D, G, E, T, P, N$$

determines exactly the same finite consumer boundary as the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *public route packet of Definition 386.6. Thus every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of the repacked name reads only the shared finite rows* X, S, D, G, E, T, P, N*, and no quotient stream, host sequence equality, selected limit, host real construction, or ambient completeness datum is exported.*

Proof. First read the $\mathsf{RegularCauchyName}^{\uparrow}$ side. The public seal export of Definition 371.32 places the $\mathsf{Real}^{\uparrow}$ endpoint on the displayed regular name packet, and the completion consumer coverage theorem Definition 371.33 states that completion-facing consumers receive no row beyond that same finite packet.

Next repack the displayed schedule, dyadic observation, radius ledger, finite regularity ledger, seal, transport, provenance, and local name rows as X, S, D, G, E, T, P, N. By the public route boundary of Definition 386.8, any $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of those rows factors through the public route packet of Definition 386.6.

Finally apply the non-escape boundary of Definition 386.4. Neither side of the repacking has a coordinate for quotient streams, host sequence equality, selected host limits, host real construction, or ambient completeness. Hence the two consumer routes expose exactly the same finite public boundary. □

Theorem 386.11 (CauchyLimitSeal RegSeqRat window non-escape)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *public route whose source row is a* $\mathsf{RegSeqRat}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and* $\mathsf{Real}^{\uparrow}$ *seal row. The route exports no quotient stream equality, selected host limit, uniqueness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier regseqrat window nonescape`

Proof. Project the carrier of Definition 386.1. Its source row is X, its schedule is S, its observation ledger is D, its diagonal completion row is G, and its public seal row is E.

Apply Definition 386.3. A public read beginning at the $\mathsf{RegSeqRat}^{\uparrow}$ source must pass through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, and the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row before it reaches the $\mathsf{Real}^{\uparrow}$ seal.

Finally combine Definition 386.5 with Definition 386.4. These boundaries confine the public endpoint to the displayed seal route and rule out quotient stream equality, selected host limits, uniqueness theorems, and ambient real completeness as available export coordinates. □

Theorem 386.12 (CauchyLimitSeal obligation closure package)**.** *The carrier definition, the* $\mathsf{NameCert}_{\mathsf{CauchyLimitSea}}$ *obligations, the finite-window factorization, the non-escape boundary, and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary jointly supply the full paper-side obligation surface for* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The package preserves the exclusions on quotient streams, selected host limits, uniqueness of limits, host real construction, and ambient completeness.*

Proof. Start with the carrier of Definition 386.1. Its coordinates name exactly the regular rational source, schedule, dyadic ledger, completion row, real seal, transport rows, package provenance, and local certificate that form the public packet.

Apply the component obligations in order. The NameCert obligation theorem supplies habitation, classifier stability, exactness, and non-escape requirements; the window factorization theorem identifies the consumer route through the scheduled finite observations; the non-escape theorem records the absent quotient, host-limit, uniqueness, and completeness fields; and the $\mathsf{Real}^{\uparrow}$ consumer-boundary theorem fixes the public endpoint of the route.

These rows are finite and exhaust the displayed packet surface. Since the carrier contains no further coordinate from which an omitted quotient, selected limit, uniqueness principle, host real, or ambient-completeness datum could be read, the listed component rows provide the whole paper-side obligation package. □

Theorem 386.13 (CauchyLimitSeal public export certificate)**.** *The carrier,* NameCert *obligations, finite-window factorization, non-escape boundary,* $\mathsf{Real}^{\uparrow}$ *consumer boundary, and obligation closure package assemble a public certificate for the sealed finite-window route over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The exported certificate exposes exactly the displayed source row, scheduled window, dyadic observation ledger, diagonal completion row, seal row, transport rows, package provenance, and local certificate.*

Proof. Start with the carrier of Definition 386.1. It supplies the displayed regular source, schedule, dyadic ledger, diagonal completion row, real seal, transport rows, package provenance, and local certificate.

Apply the NameCert obligations of Definition 386.2. They provide carrier habitation, classifier stability, exactness of the consumer route, and the non-escape clauses needed for a public certificate.

Next use the window factorization theorem Definition 386.3 and the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. These identify the public read route through the scheduled finite window, dyadic observation ledger, completion diagonal, and seal row.

Finally combine the non-escape boundary of Definition 386.4 with the obligation closure package of Definition 386.12. The rows listed there exhaust the public surface, so the assembled certificate exports the sealed finite-window route and nothing outside the stated BEDC coordinates. □

Theorem 386.14 (CauchyLimitSeal downstream schedule exhaustion)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or* $\mathsf{RegularCauchyName}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet reads a scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic observation ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, completion diagonal row, and* $\mathsf{Real}^{\uparrow}$ *seal row. The consumer-facing packet also carries only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it has no coordinate for quotient streams, selected host limits, uniqueness of limits, host real equality, or ambient completeness.*

Proof. Begin with the window factorization theorem Definition 386.3. It sends each read through a scheduled $\mathsf{StreamName}^{\uparrow}$ window, the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source row, the dyadic observation ledger, and the completion diagonal row before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. It confines Real-facing consumers to the displayed seal route and the visible history, transport, continuation, provenance, and local certificate rows.

For completion-facing reads, use the public route boundary of Definition 386.8. For regular-name reads, use the RegularCauchyName consumer link of Definition 386.10. These two rows put the completion and regular-name consumers inside the same scheduled route.

Finally apply the public export certificate of Definition 386.13. The certificate packages exactly the source, schedule, dyadic observation, diagonal completion, seal, transport, provenance, and local naming rows. Since the carrier and certificate contain no quotient-stream, selected-host-limit, uniqueness, host-real-equality, or ambient-completeness coordinate, every downstream consumer is exhausted by the displayed schedule. □

Theorem 386.15 (CauchyLimitSeal downstream completion handoff)**.** *A downstream completion consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier can pull back only along the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source, scheduled* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and* $\mathsf{Real}^{\uparrow}$ *seal. The handoff carries the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows and does not export an independent completeness principle, quotient stream, selected host limit, uniqueness theorem, or host real construction.*

Proof. Apply the window factorization theorem Definition 386.3. It forces every downstream read through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source row, the corresponding dyadic observation ledger, and the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row before reaching the $\mathsf{Real}^{\uparrow}$ seal.

Use the non-escape boundary of Definition 386.4. The accepted carrier has no field for quotient streams, selected host limits, uniqueness theorems, host real construction, or ambient completeness.

The public export certificate of Definition 386.13 packages the remaining rows as the consumer-facing handoff. Thus a completion consumer can pull back only through the displayed finite-window route and its visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates. □

Theorem 386.16 (CauchyLimitSeal null-sequence factorization)**.** *Let two sealed* $\mathsf{Real}^{\uparrow}$ *reads be presented by accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carriers. If their displayed difference is accepted by a* $\mathsf{NullSequence}^{\uparrow}$ *packet, then every equality-consumer route factors through the CauchyLimitSeal scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the* $\mathsf{NullSequence}^{\uparrow}$ *tolerance-window-threshold packet. The factored route carries only the visible* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows supplied by those packets, and it exports no ambient real completeness, quotient stream equality, selected host limit, or hidden tail witness.*

Proof. Begin with the local window factorization theorem Definition 386.3. It sends each sealed read through the scheduled finite window, the dyadic observation ledger, and the completion diagonal row before any $\mathsf{Real}^{\uparrow}$ seal coordinate is consumed.

Apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. This confines the equality-facing read to the displayed seal row and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use the downstream completion handoff of Definition 386.15. A completion consumer may pull back only along the same scheduled window, observation ledger, diagonal row, and seal route, so the comparison cannot introduce a second completion object.

Finally attach the $\mathsf{NullSequence}^{\uparrow}$ real-seal factorization of Definition 380.7. Its tolerance row, observation window, and threshold row supply the zero-tail evidence read by the two seals, and the combined route remains the finite CauchyLimitSeal route plus that displayed null-sequence packet. □

Theorem 386.17 (CauchyLimitSeal regular-source stability)**.** *Let L be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any classifier transport of the accepted seal preserves the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and the* $\mathsf{Real}^{\uparrow}$ *seal row through the stored* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of L.*

Proof. Project the carrier of Definition 386.1. The source row, schedule, dyadic ledger, diagonal row, seal row, transport rows, package row, and local certificate are named coordinates of the same finite BHist packet.

Apply the classifier-stability clause of Definition 386.2. Transport can move an accepted packet only along the displayed hsame comparison and the corresponding Cont route, while the Pkg and NameCert rows keep the dependency names fixed.

Finally use the non-escape boundary of Definition 386.4. Since no quotient stream, selected host limit, uniqueness theorem, or ambient completeness coordinate is available, the transported seal still exposes exactly the displayed regular source, scheduled windows, observation ledger, diagonal row, and seal row. □

Theorem 386.18 (CauchyLimitSeal diagonal-row compatibility)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal read consumed by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal factors through the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger as the finite route of Definition 386.3. Consequently the diagonal row G in the carrier cannot be replaced, for that seal read, by an independent completion object outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier diagonal row compatibility`

Proof. Begin with the window factorization theorem Definition 386.3. It sends each consumer read through a scheduled $\mathsf{StreamName}^{\uparrow}$ window, the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source row, the matching DyadicRatCore ledger, and then the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal coordinate before the read reaches the $\mathsf{Real}^{\uparrow}$ seal.

Use the downstream completion handoff of Definition 386.15. A completion consumer may pull back only along that displayed route, so the diagonal read is tied to the same scheduled window and observation ledger that feed the seal.

The carrier has no second completion coordinate. By Definition 386.4, no host limit, quotient stream, or ambient completeness datum can supply an alternate completion object; therefore the diagonal row consumed by the seal is the carrier row G routed through the stated finite ledger. □

Theorem 386.19 (CauchyLimitSeal seal-ledger exhaustion)**.** *The carrier of Definition 386.1, the* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *obligations, regular-source stability, diagonal-row compatibility, the* $\mathsf{Real}^{\uparrow}$ *consumer boundary, and the non-escape theorem exhaust the finite seal ledger for downstream* $\mathsf{Real}^{\uparrow}$ *and completion consumers. No further* BHist *row is available beyond the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Proof. Start with the carrier projection. Its finite packet lists exactly the regular source family, schedule, observation ledger, diagonal completion row, real seal row, transport rows, package provenance, and local certificate.

Apply the NameCert obligations of Definition 386.2. They supply habitation, classifier stability, exact finite-route factorization, and the exclusion of quotient, host-limit, uniqueness, and ambient-completeness data.

Next combine Definition 386.17 and Definition 386.18. The first keeps classifier transport on the displayed source, schedule, ledger, diagonal, and seal rows; the second prevents a downstream completion read from replacing the diagonal row by an object outside the packet.

Finally use the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5 and the non-escape boundary of Definition 386.4. These identify the public endpoint and rule out every listed hidden field, so the finite rows named in the statement exhaust the seal ledger for downstream real and completion consumers. □

Theorem 386.20 (CauchyLimitSeal scheduled-window pullback)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any downstream predicate read on the sealed* $\mathsf{Real}^{\uparrow}$ *endpoint E pulls back uniquely to a scheduled* $\mathsf{StreamName}^{\uparrow}$ *finite window in S, its* $\mathsf{RegSeqRat}^{\uparrow}$ *row in X, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row in D, and the displayed* hsame *and* Cont *transport rows in T. This pullback is a projection of the finite* BHist *packet named by P and N, and it has no host-limit coordinate, quotient-stream coordinate, or independent completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier scheduled window pullback`

Proof. Project the carrier coordinates X, S, D, G, E, T, P, N. The endpoint E is not stored as a free real object; it is the seal row reached through the scheduled route carried by the same finite BHist packet.

Apply the window factorization of Definition 386.3. A read on E must pass through one scheduled $\mathsf{StreamName}^{\uparrow}$ window from S, then through its $\mathsf{RegSeqRat}^{\uparrow}$ source row in X, and then through the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row in D.

The remaining movement is exactly the transport ledger T. The stored hsame and Cont rows identify the admissible route from the scheduled observation to the completion row G and the seal row E, while P and N name the dependency package and local NameCert.

Uniqueness is componentwise in the displayed packet. Since no additional carrier coordinate contains a host limit, quotient stream, or ambient completeness datum, a downstream predicate on E has only the scheduled finite-window pullback just described. □

Theorem 386.21 (CauchyLimitSeal dyadic observation replay)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier $L = (X, S, D, G, E, T, P, N)$, replaying any consumer read through the seal consumes exactly the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D together with the* Pkg *provenance P and the local* NameCert *row N. The replay does not consume an external tolerance, a hidden tail witness, a quotient-stream comparison, or a selected host limit.*

Proof. Start with the finite route of Definition 386.3. The consumer read is routed from S and X into the observation ledger D before it reaches the completion diagonal and the seal.

The replay step is therefore a ledger projection. Each dyadic observation used by the consumer is one of the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ rows in D, and its admissibility is recorded by the package provenance P and the local certificate N.

Use the non-escape theorem Definition 386.4. The carrier contains no external tolerance field, no hidden tail witness, no quotient-stream comparison, and no selected host-limit row. Hence the replay consumes exactly D, P, and N along the visible BHist, hsame, and Cont route. □

Theorem 386.22 (CauchyLimitSeal window-ledger normal form)**.** *Every completion or* $\mathsf{Real}^{\uparrow}$ *consumer replay of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier has a finite window-ledger normal form. It is determined by the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows. No quotient stream, selected host limit, uniqueness theorem, or ambient completeness coordinate participates in this normal form.*

Proof. Begin with the scheduled-window pullback of Definition 386.20. It places any downstream consumer read on a displayed $\mathsf{StreamName}^{\uparrow}$ window, its $\mathsf{RegSeqRat}^{\uparrow}$ source row, and the matching $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger.

Replay the dyadic observation by Definition 386.21. The ledger row used by the consumer is therefore the displayed row D together with the package and local certificate rows, not an external tolerance or hidden tail witness.

Finish with the non-escape theorem and the $\mathsf{Real}^{\uparrow}$ consumer boundary. They rule out quotient streams, selected host limits, uniqueness premises, and ambient completeness data, so the displayed route is the normal form for completion and $\mathsf{Real}^{\uparrow}$ replay. □

Theorem 386.23 (CauchyLimitSeal seal-route determinacy)**.** *If two accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer replays use the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, then they determine the same seal route for every completion or* $\mathsf{Real}^{\uparrow}$ *consumer. The comparison does not appeal to a quotient stream, selected host limit, uniqueness theorem, or ambient completeness coordinate.*

Proof. Use Definition 386.22 to put both consumer replays into the same displayed window-ledger normal form. The hypothesis identifies each row of that normal form componentwise.

The diagonal read is fixed by Definition 386.18. Once the scheduled window, regular row, and dyadic ledger agree, the completion consumer cannot replace the diagonal row by a separate completion object.

The seal endpoint is then fixed by the downstream completion handoff and the $\mathsf{Real}^{\uparrow}$ consumer boundary. Since the carrier has no quotient stream, selected host limit, uniqueness theorem, or ambient completeness coordinate, there is no further field that could distinguish two seal routes. □

Theorem 386.24 (CauchyLimitSeal completion-consumer replay)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier replays through the finite route determined by the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The replay exposes no quotient stream, selected host limit, uniqueness theorem, or ambient completeness coordinate to the consumer.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier completion consumer replay`

Proof. Start with a completion-facing consumer read. The scheduled-window pullback sends it to the displayed finite window, regular rational row, dyadic observation row, and transport route.

Apply the window-ledger normal form. The replay is therefore read from the listed carrier rows and their package and local naming provenance, rather than from a host completion principle.

Finally apply seal-route determinacy. Any second replay with the same displayed rows must follow the same route, and the non-escape boundary prevents the consumer from importing quotient streams, selected host limits, uniqueness theorems, or ambient completeness coordinates. □

Theorem 386.25 (CauchyLimitSeal completion-consumer exactness)**.** *Every completion-facing consumer of accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carriers has the exact finite route obtained by composing the scheduled-window pullback of Definition 386.20, the dyadic replay of Definition 386.21, the downstream completion handoff of Definition 386.15, and the null-sequence factorization of Definition 386.16. The route is wholly contained in the carrier coordinates* X, S, D, G, E, T, P, N *and their visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. First pull the completion-facing read back from the sealed endpoint. By Definition 386.20, the read lands on a scheduled finite window, its regular rational source row, the dyadic observation row, and the displayed transport rows.

Next replay the dyadic observation. By Definition 386.21, the consumer uses exactly the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger with the package provenance and local naming row, rather than any external tolerance or hidden tail witness.

Now apply the downstream completion handoff of Definition 386.15. It keeps the completion consumer on the same source row, scheduled window, observation ledger, diagonal row, and seal route.

Finally use the null-sequence factorization of Definition 386.16 when the consumer compares two sealed reads. The comparison attaches only the displayed $\mathsf{NullSequence}^{\uparrow}$ tolerance-window-threshold packet to the already fixed route, so the completion consumer receives exactly the finite path through X, S, D, G, E, T, P, N and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. □

Theorem 386.26 (CauchyLimitSeal public route exhaustion)**.** *Every public* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route available to a downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer is exhausted by the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *packet,* hsame *and* Cont *transports,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. No public route can read a quotient stream, selected host limit, hidden tail witness, or ambient completeness coordinate outside those carrier rows.*

Proof. Project the accepted carrier $L = (X, S, D, G, E, T, P, N)$ from Definition 386.1. The only public rows named by the packet are the regular source, schedule, dyadic ledger, diagonal completion row, real seal, transport ledger, package row, and local certificate.

Use the scheduled-window pullback of Definition 386.20 and the dyadic replay of Definition 386.21. These identify any consumer read with a scheduled finite window, its regular rational source row, the corresponding dyadic observation, and the stored transport route.

Now apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5 and the non-escape theorem of Definition 386.4. They leave no coordinate for quotient streams, selected host limits, hidden tail witnesses, or ambient completeness data, so the displayed rows exhaust the public route. □

Theorem 386.27 (CauchyLimitSeal bridge boundary)**.** *The abstract* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *bridge surface is supplied exactly by the finite route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row, displayed transport rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *row. A bridge consumer receives no quotient stream, selected host limit, hidden tail witness, independent completion object, or ambient completeness coordinate beyond that route.*

Proof. First compose the public route boundary of Definition 386.8 with the public export certificate of Definition 386.13. These rows identify the exported surface with the carrier coordinates X, S, D, G, E, T, P, N.

Next attach the downstream completion handoff of Definition 386.15. It keeps completion-facing reads on the same regular source, schedule, dyadic ledger, diagonal completion row, and seal route.

Use seal-ledger exhaustion from Definition 386.19. The seal-facing ledger has no additional row from which a quotient stream, selected host limit, hidden tail witness, or independent completion object could be read.

Finally apply public route exhaustion from Definition 386.26. The bridge boundary is therefore exactly the finite route listed in the statement, with the displayed transport, package, and local naming rows carrying the whole consumer-facing surface. □

Theorem 386.28 (CauchyLimitSeal pullback determinacy)**.** *Let two completion pullbacks of accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carriers use the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger inside the displayed carrier packet. Then they determine the same seal route*

through the RegSeqRat$^{\uparrow}$ *source,* RealCauchyCompletion$^{\uparrow}$ *diagonal row,* Real$^{\uparrow}$ *seal,* hsame *and* Cont *transport rows,* Pkg *provenance, and* NameCert *row. The pullback cannot introduce a quotient stream, a selected host limit, or an independent completion object.*

Proof. Begin with the scheduled-window pullback. Once the StreamName$^{\uparrow}$ window and DyadicRatCore$^{\uparrow}$ ledger row are fixed, Definition 386.20 fixes the corresponding RegSeqRat$^{\uparrow}$ source row and the displayed transport rows in the carrier.

Use diagonal-row compatibility from Definition 386.18. The completion read tied to that scheduled window and ledger must pass through the carrier diagonal row G, so it cannot be replaced by a separate completion object.

Finally apply the downstream completion handoff of Definition 386.15 and the public route exhaustion of Definition 386.26. The same fixed window and ledger therefore force the same route through the seal and leave no quotient-stream or selected-host-limit coordinate to distinguish another pullback. □

Theorem 386.29 (CauchyLimitSeal standard boundary package)**.** *The window factorization,* Real$^{\uparrow}$ *consumer boundary, downstream completion handoff, null-sequence factorization, public route exhaustion, and pullback determinacy assemble the standard boundary package for* CauchyLimitSeal$^{\uparrow}$*. The package presents the public seal route over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RealCauchyCompletion$^{\uparrow}$ Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it exports no host-limit, quotient-completion, ambient-completeness, or hidden-tail premise.*

Proof. First combine the window factorization of Definition 386.3 with the Real$^{\uparrow}$ consumer boundary of Definition 386.5. These rows route every public read from the schedule and regular source through the dyadic ledger, completion diagonal, transport rows, and seal.

Next attach the downstream completion handoff of Definition 386.15. It identifies the completion-facing side of the same route and prevents an independent completeness principle or completion object from entering the packet.

For equality-facing reads, use the null-sequence factorization of Definition 386.16. It adds only the displayed NullSequence$^{\uparrow}$ tolerance-window-threshold packet to the already fixed seal route.

Finally apply public route exhaustion and pullback determinacy from Definition 386.26 and Definition 386.28. The assembled rows give the standard boundary package because they exhaust the public carrier surface and determine each completion pullback from the displayed schedule and dyadic ledger. □

Theorem 386.30 (CauchyLimitSeal uniform-criterion pullback)**.** *Let an accepted* CauchyLimitSeal$^{\uparrow}$ *carrier present a sealed* Real$^{\uparrow}$ *read whose source family is governed by an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet. Then the sealed read pulls back through the same scheduled* StreamName$^{\uparrow}$ *windows, the same* DyadicRatCore$^{\uparrow}$ *observation row, the same* RegSeqRat$^{\uparrow}$ *source row, and the same* RealCauchyCompletion$^{\uparrow}$ *diagonal row used by the seal carrier. The pullback consumes the uniform family evidence only through its finite family ledger, shared window family, uniform modulus, dyadic tolerance, tail ledger, seal handoff, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier uniform criterion pullback`

Proof. Project the accepted CauchyLimitSeal$^{\uparrow}$ carrier. By Definition 386.20, every sealed read on the Real$^{\uparrow}$ endpoint pulls back to a scheduled StreamName$^{\uparrow}$ finite window, its RegSeqRat$^{\uparrow}$ source row, the matching DyadicRatCore$^{\uparrow}$ observation row, and the displayed transport route.

Now project the accepted UniformCauchyCriterion$^{\uparrow}$ packet. Its Definition 435.9 route reads the finite family ledger, shared window family, uniform modulus, dyadic tolerance, and tail ledger before any real-facing seal handoff is consumed.

The two projections meet at the shared scheduled windows and dyadic observation row. The CauchyLimitSeal diagonal row remains the displayed RealCauchyCompletion$^{\uparrow}$ coordinate, while the uniform criterion contributes only the finite family-tail evidence attached to those same windows.

Finally use Definition 435.8. The uniform packet has no quotient stream family, selected family limit, host real equality, or ambient completeness coordinate, so the pullback stays inside the sealed finite route. □

Theorem 386.31 (CauchyLimitSeal uniform tail seal route)**.** *For any accepted* CauchyLimitSeal$^{\uparrow}$ *carrier whose source family is certified by an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet, the route from family-tail evidence to the sealed* Real$^{\uparrow}$ *row factors through the shared family index, the shared* StreamName$^{\uparrow}$ *window family, the uniform* CauchyModulus$^{\uparrow}$ *schedule, the* DyadicRatCore$^{\uparrow}$ *tolerance row, the finite tail ledger, the* RealCauchyCompletion$^{\uparrow}$ *diagonal row, and the* Real$^{\uparrow}$ *seal. The composed route is exactly the composition of the CauchyLimitSeal window factorization, the* Real$^{\uparrow}$ *consumer boundary, and the UniformCauchyCriterion tail-ledger exactness surface.*

Proof. First apply Definition 435.9. It forces every accepted family-tail comparison through the family index, shared window family, uniform modulus, dyadic tolerance, and finite tail ledger.

Next apply Definition 386.3. The seal carrier reads the same scheduled window through the regular source row, the dyadic observation ledger, the completion diagonal row, and the displayed transport ledger before it reaches the seal.

Now attach Definition 386.5. A Real$^{\uparrow}$ consumer may read the result only after the displayed finite-window route has passed through the diagonal row and the seal row.

The composed route therefore has no branch outside the shared family index, uniform modulus, tolerance row, tail ledger, diagonal row, and seal. Package and local naming provenance are carried by the two packets' Pkg and NameCert rows. □

Theorem 386.32 (CauchyLimitSeal family seal non-escape)**.** *The composed seal route of Definition 386.31 exports no quotient stream family, no selected family limit, no ambient completeness principle, no host real equality, and no hidden tail witness. Its public surface is exactly the displayed* UniformCauchyCriterion$^{\uparrow}$ *family-tail packet together with the displayed* CauchyLimitSeal$^{\uparrow}$ *schedule, observation, diagonal, transport, package, and local naming rows.*

Proof. Start with the uniform side. By Definition 435.8, an accepted UniformCauchyCriterion$^{\uparrow}$ carrier exports only the family index ledger, shared windows, uniform modulus, dyadic tolerance, finite tail ledger, seal handoff, and visible kernel provenance rows.

Now read the seal side. The CauchyLimitSeal non-escape boundary of Definition 386.4 removes quotient streams, selected host limits, hidden tail witnesses, uniqueness theorems, and ambient completeness data from the sealed route.

Finally combine the two boundaries through Definition 386.30 and Definition 386.31. The pullback and tail route share the same windows, tolerance row, tail ledger, diagonal row, and seal, so no excluded coordinate can be introduced between the two packets. □

Theorem 386.33 (CauchyLimitSeal uniform family source)**.** *The family-source packet for the public* CauchyLimitSeal$^{\uparrow}$ *route is the finite row family over* CauchyModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, NullSequence$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* StreamName$^{\uparrow}$ *consisting of:*

1. *a shared* StreamName$^{\uparrow}$ *index schedule for the family windows;*

2. *a* $\mathsf{RegSeqRat}^{\uparrow}$ *source row for each scheduled regular rational stream;*
3. *a* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning the displayed tail modulus for the scheduled family;*
4. *a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for each requested window precision;*
5. *a* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row witnessing that family differences vanish inside the scheduled tail;*
6. *a* $\mathsf{Real}^{\uparrow}$ *seal handoff row for the downstream real consumer.*

The packet is read only through the public route of Definition 386.6, the uniform tail route of Definition 386.31, and the family non-escape boundary of Definition 386.32.

Proof. Begin with the public route packet of Definition 386.6. It supplies the scheduled $\mathsf{StreamName}^{\uparrow}$ window family, the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the dyadic observation ledger, the completion diagonal row, and the $\mathsf{Real}^{\uparrow}$ seal row.

Attach the uniform tail route of Definition 386.31. That route adds the shared $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows and reads the family tail through the same scheduled windows.

Finally use the family non-escape theorem Definition 386.32. It prevents quotient stream families, selected family limits, hidden tail witnesses, and ambient completeness data from entering the source packet, so the listed six row types exhaust the family-source surface. □

Theorem 386.34 (CauchyLimitSeal family window pullback)**.** *Every family-seal consumer window of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet factors through the existing public route packet and then pulls back along the uniform-criterion route. Concretely, a consumer read must pass through the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail comparison, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and the* $\mathsf{Real}^{\uparrow}$ *seal before it reaches any downstream family consumer.*

Proof. First apply the public route boundary of Definition 386.8. It places every real or completion-facing consumer read inside the projected route packet X, S, D, G, E, T, P, N.

Next apply the uniform-criterion pullback of Definition 386.30. The family window is pulled back to the shared index, tolerance, and tail rows before any seal read is consumed.

Now combine the result with Definition 386.31. The route carries the uniform modulus and null-tail comparison through the same scheduled windows used by the public seal. The family non-escape boundary of Definition 386.32 then excludes any hidden completion or selected family limit outside that pullback. □

Theorem 386.35 (CauchyLimitSeal family ledger exhaustion)**.** *The family ledger for the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route contains exactly three classes of rows: source-index rows naming the* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{RegSeqRat}^{\uparrow}$ *family entries, tail-seal rows naming the* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{NullSequence}^{\uparrow}$ *tail evidence, and* $\mathsf{Real}^{\uparrow}$ *handoff rows naming the diagonal completion route and final seal. It contains no hidden completion datum, no quotient stream family, no selected family limit, and no ambient completeness row.*

Proof. Read the source side through Definition 386.33. The only source-index data are the shared schedule and the displayed regular rational family entries.

Read the tail side through Definition 386.31. The only tail-seal data are the modulus, dyadic tolerance, null-tail comparison, and their displayed transport rows.

Read the endpoint through Definition 386.5 and the public route boundary of Definition 386.8. The endpoint is the diagonal completion route followed by the $\mathsf{Real}^{\uparrow}$ seal. The family non-escape theorem of Definition 386.32 rules out every additional hidden completion, quotient family, selected limit, or ambient completeness row, so the three listed classes exhaust the ledger. □

Theorem 386.36 (CauchyLimitSeal consumer ledger totality)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer read of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet is exhausted by the carrier rows* X, S, D, G, E, T, P, N *together with the family tail rows displayed in Definition 386.35. Equivalently, after the public route, family window pullback, and family ledger exhaustion are applied, there is no remaining consumer-visible row outside the source, schedule, dyadic observation, diagonal completion, seal, transport, provenance, naming, modulus, null-tail, and handoff ledgers.*

Proof. Start with window factorization from Definition 386.3. It places every consumer read inside the scheduled finite windows of the displayed source, observation, completion, and seal rows.

Next apply the public route boundary of Definition 386.8. The completion-facing and $\mathsf{Real}^{\uparrow}$-facing reads are therefore already routed through X, S, D, G, E, T, P, N before any downstream consumer receives them.

For family reads, pull the window back through Definition 386.34. The remaining tail-side rows are exactly the family rows listed by Definition 386.35. That theorem excludes hidden completion data, quotient stream families, selected family limits, and ambient completeness rows, so the combined list is total for consumers. □

Theorem 386.37 (CauchyLimitSeal hsame route uniqueness)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, any displayed* hsame *transport that preserves the public carrier rows and the family tail rows induces the same consumer route through* X, S, D, G, E, T, P, N *and the family ledger of Definition 386.35. Thus a consumer cannot obtain two distinct seal routes by replaying the same window, tail, and handoff ledgers through different* hsame *witnesses.*

Proof. The carrier rows are componentwise. Transport of X, S, D, G, E, T, P, N preserves the same source, schedule, dyadic observation, diagonal completion, seal, transport, provenance, and local naming data.

The family tail is fixed by Definition 386.35. Transport can replay the source-index rows, tail-seal rows, and $\mathsf{Real}^{\uparrow}$ handoff rows, but it cannot create a fourth ledger class.

Combine these two facts with totality from Definition 386.36. Since every consumer read is already exhausted by those rows, two transports that preserve them determine the same public route boundary and the same family handoff. □

Theorem 386.38 (CauchyLimitSeal window replay determinacy)**.** *Let a downstream consumer replay a scheduled* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *window using the stored* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, family tail rows, diagonal completion row, and* $\mathsf{Real}^{\uparrow}$ *seal. The replay determines the same finite consumer ledger each time: it reuses* X, S, D, G, E, T, P, N *and the family tail rows, and it cannot alter the completion route, insert a hidden limit, or bypass the seal.*

Proof. Replay begins at the stored schedule S. The window factorization theorem sends the read through the same source row X, dyadic observation row D, diagonal completion row G, and seal row E.

The family pullback theorem then fixes the tail-side rows for that window. By family ledger exhaustion, those rows are exactly the source-index, tail-seal, and handoff rows already named in the packet.

Finally apply route uniqueness from Definition 386.37. Replaying the same displayed rows cannot change the route, and the consumer totality theorem leaves no hidden limit, alternate completion route, or seal bypass to choose. □

Theorem 386.39 (CauchyLimitSeal source factorization)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer source factors through exactly the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *and* Cont *transport rows,* Pkg *provenance, and* $\mathsf{NameCert}$ *row of its carrier. Family tail rows enter only through the family ledger exhaustion of Definition 386.35; no consumer source may add a hidden completion row, selected limit, quotient family, alternate seal, or unlisted dependency outside that route.*

Proof. Start with window factorization from Definition 386.3. It sends every accepted consumer read through the scheduled $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ source row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row, and the displayed transport rows before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Next apply the public route boundary of Definition 386.8. The carrier rows X, S, D, G, E, T, P, N are exactly the public source, schedule, observation, diagonal, seal, transport, provenance, and naming rows available to a completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer.

For family-indexed reads, use the family window pullback of Definition 386.34. The pullback may add only the uniform tail evidence attached to the same scheduled window and the same dyadic observation row.

Then invoke the family ledger exhaustion theorem Definition 386.35. It confines those tail-side rows to the source-index, tail-seal, and handoff ledgers already named by the packet, so no hidden completion or selected family limit remains.

Finally combine consumer ledger totality, hsame route uniqueness, and window replay determinacy from Definition 386.36, Definition 386.37, and Definition 386.38. The total ledger is exhausted, transport cannot split it into two consumer sources, and replay of the displayed window determines the same sealed route each time. □

Theorem 386.40 (CauchyLimitSeal family tail admission)**.** *A family-tail row is admitted by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet only when the same finite packet displays its* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *source family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row,* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail comparison,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. A tail row lacking any one of these displayed coordinates is not a public family-tail row of the seal.*

Proof. Begin with the family-source packet of Definition 386.33. It names the schedule, regular source family, tail modulus, dyadic tolerance rows, null-tail comparison, and real handoff that make a family-tail row visible.

Next apply the family window pullback of Definition 386.34. The pullback attaches those rows to the same public route packet, so the diagonal completion row, seal row, transports, continuation route, provenance, and naming row are read in the same finite packet.

Finally use family ledger exhaustion from Definition 386.35. Since the ledger has only source-index, tail-seal, and handoff classes, a row missing one of the displayed coordinates has no remaining public channel through which it could be admitted. □

Theorem 386.41 (CauchyLimitSeal seal-window ledger compatibility)**.** *The seal-window route and the family-tail ledger of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet consume the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row. Thus a* $\mathsf{Real}^{\uparrow}$ *consumer cannot combine the seal route from one scheduled window with the family-tail evidence of another.*

Proof. Read the public route boundary first. By Definition 386.8, the seal-window route is the projected carrier route through the displayed schedule, observation ledger, diagonal completion row, and seal row.

For the family side, apply Definition 386.34. It pulls the family-tail read back along that same public route packet, so the scheduled window and dyadic observation row are shared before any tail evidence is consumed.

Window replay determinacy from Definition 386.38 rules out a mixed route. Replaying a different schedule or observation row would change the finite ledger, while the accepted packet exposes only the stored window and observation route. □

Theorem 386.42 (CauchyLimitSeal real consumer family exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet is exhausted by the carrier route together with the family-tail ledger. The consumer may read the displayed source schedule, observation row, tail modulus, null-tail comparison, diagonal completion row, seal row,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates, and it cannot add quotient streams, hidden completion rows, selected limits, or ambient completeness.*

Proof. Start with the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. It confines every real-facing read to the finite carrier route before the seal is exposed.

Attach the family-tail side by Definition 386.40 and Definition 386.41. These two results ensure that the tail row is admitted only with the displayed source, observation, modulus, null-tail, completion, seal, transport, continuation, provenance, and naming rows, and that it uses the same scheduled window as the seal route.

Finally apply consumer ledger totality from Definition 386.36. The combined carrier route and family ledger already exhaust the consumer-visible rows, so quotient streams, hidden completion rows, selected limits, and ambient completeness have no coordinate to enter. □

Theorem 386.43 (CauchyLimitSeal bridge interface)**.** *The* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier rows, public route packet, source factorization, family ledger exhaustion, and consumer ledger totality assemble the concrete bridge from the public sealed* $\mathsf{Real}^{\uparrow}$ *boundary to the abstract* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *interface. The bridge is entirely over the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source family,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *history surface,* hsame *transport,* Cont *continuation row,* Pkg *provenance, and* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealUp StdBridge`

Proof. Project the accepted packet to the carrier rows X, S, D, G, E, T, P, N of the public route packet. The public route boundary supplies the sealed $\mathsf{Real}^{\uparrow}$ side, while source factorization identifies the same rows as the abstract $\mathsf{CauchyLimitSeal}^{\uparrow}$ source surface.

The family side is supplied by ledger exhaustion. It restricts all tail and family rows to the displayed source-index, tail-seal, and handoff ledgers attached to the same schedule and dyadic observations.

Finally apply consumer ledger totality. Every public or completion-facing read is already routed through the carrier rows and the family ledger, so the bridge interface is obtained by projection and composition of those named surfaces. □

Theorem 386.44 (CauchyLimitSeal bridge route exactness)**.** *Every bridge read from the public sealed* $\mathsf{Real}^{\uparrow}$ *boundary into the abstract* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *interface factors exactly through the route chain Definition 386.8, Definition 386.34, Definition 386.35, and Definition 386.39. Hence*

no bridge consumer can read a row outside the public route, the pulled-back family window, the exhausted family ledger, and the factored source surface.

Proof. Begin with the public route boundary. It places every $\mathsf{Real}^{\uparrow}$ or completion-facing read inside the carrier route packet before any bridge handoff is interpreted.

For family-indexed reads, pull the same window back through the family window theorem. The pullback keeps the read on the shared schedule, source row, dyadic observation ledger, tail evidence, diagonal completion row, and seal row.

Apply family ledger exhaustion to remove any remaining row outside the source-index, tail-seal, and handoff ledgers. Source factorization then repacks the same routed rows as the abstract $\mathsf{CauchyLimitSeal}^{\uparrow}$ source surface, giving exactness of the bridge route. □

Theorem 386.45 (CauchyLimitSeal bridge non-escape)**.** *The CauchyLimitSeal bridge adds no quotient stream, selected host limit, uniqueness theorem, ambient completeness datum, hidden family limit, or alternate seal route. Its entire consumer-visible content is the public route packet, the pulled-back family ledger, the* $\mathsf{Real}^{\uparrow}$ *handoff, and the existing* hsame*,* Cont*,* Pkg*, and* NameCert *carrier rows.*

Proof. The bridge interface theorem confines the handoff to the carrier rows and the displayed family ledger. Thus any bridge consumer starts with only the public route packet and the already factored source surface.

Route exactness then forces every read through the public route boundary, family window pullback, family ledger exhaustion, and source factorization. Each of those theorems excludes hidden completion data, quotient stream families, selected limits, and alternate routes at the point where such a row could enter.

Therefore composition of the four route constraints leaves no escape row for the bridge to add. The bridge is a public projection of the sealed packet, not an additional completion or selection principle. □

Theorem 386.46 (CauchyLimitSeal RealUp uniqueness handoff)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, let two* $\mathsf{Real}^{\uparrow}$*-facing consumer reads both factor through the displayed* $\mathsf{StreamName}^{\uparrow}$ *scheduled window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback,* $\mathsf{Real}^{\uparrow}$ *seal route, bridge route exactness row, and bridge non-escape row. Then the two reads determine the same* $\mathsf{Real}^{\uparrow}$ *consumer surface up to* hsame*. The conclusion is only a consumer-surface handoff: it does not identify quotient streams, choose a limit, assert host real equality, or import ambient completeness.*

Proof. Begin with seal-route determinacy from Definition 386.23. Once the scheduled window, dyadic observation, regular-Cauchy readback, and seal route are fixed, replaying the accepted packet cannot select a second seal-facing route.

Next apply bridge route exactness from Definition 386.44. It forces each bridge read to pass through the same public route, pulled-back family window, exhausted family ledger, and factored source surface, so the two reads have the same displayed coordinates before the $\mathsf{Real}^{\uparrow}$ handoff is consumed.

Finally use bridge non-escape from Definition 386.45. Since the bridge contributes no quotient stream, selected host limit, uniqueness principle, alternate seal, or completeness datum, repacking the shared window and seal coordinates gives one $\mathsf{Real}^{\uparrow}$ consumer surface, with equality understood only through the packet's displayed hsame transport. □

Theorem 386.47 (CauchyLimitSeal source schedule ledger coherence)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any scheduled window selected from S determines the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *source row in X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row in D, and the displayed* hsame *and* Cont *transport rows in T. Thus a downstream consumer cannot read a scheduled window independently from its source, observation, and transport ledger rows.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier source schedule ledger coherence`

Proof. Project the accepted carrier to its finite route coordinates. The window coordinate lies in S, and the route packet stores its associated source, dyadic observation, diagonal, seal, transport, provenance, and naming rows in the same packet X, S, D, G, E, T, P, N.

Apply the window factorization theorem Definition 386.3. It sends a selected scheduled window through the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ source row in X and the matching $\mathsf{DyadicRatCore}^{\uparrow}$ observation row in D before any diagonal or seal read is consumed.

Finally use regular-source stability from Definition 386.17. Transport of the accepted packet is allowed only along the displayed hsame comparison and Cont route, so the scheduled source and dyadic ledger rows remain coherent under downstream replay. □

Theorem 386.48 (CauchyLimitSeal sealed endpoint route exhaustion)**.** *Every downstream read on the sealed* $\mathsf{Real}^{\uparrow}$ *endpoint of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier is exhausted by the source schedule, dyadic ledger, diagonal completion row, seal row, package provenance, and local* NameCert *row displayed in $L = (X, S, D, G, E, T, P, N)$. No endpoint read can add a hidden source schedule, independent dyadic observation, alternate completion row, or external endpoint route.*

Proof. Begin with the public route boundary of Definition 386.8. It confines each $\mathsf{Real}^{\uparrow}$ or completion-facing read to the projected route packet X, S, D, G, E, T, P, N before the sealed endpoint is exposed.

Next apply seal-ledger exhaustion from Definition 386.19. The endpoint-facing ledger is exactly the displayed source, schedule, observation, diagonal completion, seal, transport, provenance, and naming surface, so no second endpoint route is available in the carrier.

Use diagonal-row compatibility from Definition 386.18. A completion read tied to the selected schedule and dyadic observation must consume the carrier diagonal row G, which prevents replacing the sealed endpoint by an independent completion object. □

Theorem 386.49 (CauchyLimitSeal lattice-meet terminal factorization)**.** *Let a terminal* $\mathsf{CauchyLimitSeal}^{\uparrow}$ $\mathsf{Real}^{\uparrow}$ *read or completion-facing read use a schedule assembled from two accepted finite* $\mathsf{StreamName}^{\uparrow}$ *windows. Then the terminal read first factors through the finite-window lattice meet packet of Definition 126.257. Only after that meet surface is displayed may the consumer read the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* hsame *and* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows of the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet.*

Thus the terminal factorization exposes only the shared finite-window meet surface and the carrier route X, S, D, G, E, T, P, N. It supplies no quotient stream, selected host limit, host real equality, uniqueness principle, or ambient completeness coordinate.

Proof. Start with the two accepted $\mathsf{StreamName}^{\uparrow}$ windows. The finite-window lattice meet exhaustion theorem Definition 126.257 supplies one shared meet packet carrying the common classifier, ProbeBundle rows, InBundle membership rows, unary-history rows, dyadic ledger, transports, routes, package provenance, and local names.

Next pass the resulting schedule to the seal carrier. The window factorization theorem Definition 386.3 sends the scheduled window through the matching $\mathsf{RegSeqRat}^{\uparrow}$ source row and $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger before any diagonal completion or $\mathsf{Real}^{\uparrow}$ seal read is consumed.

Now apply the public route boundary of Definition 386.8. The terminal $\mathsf{Real}^{\uparrow}$ or completion-facing consumer is confined to the displayed carrier route X, S, D, G, E, T, P, N, and Definition 386.36 exhausts the consumer-visible ledger along that route.

Finally use non-escape from Definition 386.4. Since neither the StreamName meet packet nor the CauchyLimitSeal carrier has a coordinate for quotient streams, selected host limits, host real equality, uniqueness principles, or ambient completeness, the terminal read can only be the displayed finite meet followed by the displayed seal route. □

Theorem 386.50 (CauchyLimitSeal NameCert downstream unblock package)**.** *The source-schedule ledger coherence row of Definition 386.47 and the sealed endpoint route exhaustion row of Definition 386.48, together with Definition 386.2, Definition 386.5, and Definition 386.4, supply the downstream obligation package for* $\mathsf{Real}^{\uparrow}$ *and completion consumers over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The package exposes only the finite sealed boundary and does not claim real completeness, quotient stream identity, selected host limits, or uniqueness of ambient real values.*

Proof. Project the finite packet $L = (X, S, D, G, E, T, P, N)$. The $\mathsf{NameCert}$ obligations of Definition 386.2 provide carrier habitation, classifier stability, finite-route exactness, and the local naming row needed by downstream consumers.

Route reads through the scheduled window and dyadic ledger using Definition 386.47. This supplies the matching $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, and displayed hsame and Cont transports before a consumer reaches the completion or real endpoint.

Then apply sealed endpoint route exhaustion from Definition 386.48 and the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. These rows identify the diagonal completion, seal, package provenance, and local naming coordinates that a $\mathsf{Real}^{\uparrow}$ or completion consumer may read.

Finally use non-escape from Definition 386.4. Since the carrier has no coordinate for quotient streams, selected host limits, uniqueness principles, or ambient completeness, the assembled package is a constructive downstream boundary over exactly the displayed BEDC objects. □

Theorem 386.51 (CauchyLimitSeal RealCompleteness diagonal consumer exactness)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet factors through the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row, and displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows named by the seal packet. The consumer exports no quotient stream, selected host limit, independent completeness proof, or alternate diagonal route.*

Proof. Start from the RealCompleteness consumer relation opened in chapter 450. A diagonal consumer may read the seal only as a completion-facing or $\mathsf{Real}^{\uparrow}$-facing downstream boundary, so its entrance is the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet.

Apply source-schedule ledger coherence from Definition 386.47. This identifies the scheduled $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ source row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger before any diagonal or seal endpoint is consumed.

Next use sealed endpoint route exhaustion from Definition 386.48. The completion row, seal row, displayed BHist data, hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row are exactly the endpoint-facing route exposed by the packet.

Finally apply the downstream unblock package of Definition 386.50. Since that package has no coordinate for quotient streams, selected host limits, independent completeness proofs, or alternate diagonal routes, the RealCompleteness diagonal consumer is forced to factor through the same finite seal route. □

Theorem 386.52 (CauchyLimitSeal tail-choice exhaustion)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, every admitted tail-choice read is exhausted by a finite* $\mathsf{StreamName}^{\uparrow}$ *window, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the null-tail comparison row, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, the* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. No admitted tail choice can depend on a quotient representative, a selected host limit, an unbounded stream schedule, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail choice exhaustion`

Proof. Begin with the RealUp uniqueness handoff. It already forces any two real-facing reads through the same scheduled window, dyadic observation, regular-Cauchy readback, seal route, bridge exactness row, and bridge non-escape row before the consumer surface is compared.

Attach family-tail admission and seal-window ledger compatibility. These rows make the tail choice visible only as a finite comparison over the stored source window, observation ledger, null-tail row, diagonal completion row, seal row, transport row, route row, provenance row, and local certificate row.

Finally use bridge non-escape and consumer ledger totality. Since those rows leave no coordinate for quotient representatives, selected host limits, unbounded schedules, or ambient completeness, the displayed finite window exhausts every admitted tail-choice read. □

Theorem 386.53 (CauchyLimitSeal Real handoff exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *handoff produced by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet is the concatenation of the tail-choice exhaustion row with the public route packet and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary. The handoff exports only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier real handoff exhaustion`

Proof. First apply tail-choice exhaustion to place the handoff on a finite scheduled window with its regular source row, dyadic observation ledger, null-tail comparison, completion diagonal, and seal row.

Next compose with the public route boundary. The public route packet records exactly the same carrier coordinates X, S, D, G, E, T, P, N, so the tail-choice surface and the real-facing surface use one route spine.

The $\mathsf{Real}^{\uparrow}$ consumer boundary then supplies the endpoint. It allows the consumer to read the sealed finite-window route and nothing outside the displayed transport, route, provenance, and naming rows. □

Theorem 386.54 (CauchyLimitSeal ledger non-escape totality)**.** *The tail-choice ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal rows of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet are total for downstream Real-and-Completion consumers. Any consumer read that is not routed through those displayed ledgers is outside the packet and cannot be converted into a hidden completion row, alternate seal, quotient stream, chosen limit, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier ledger nonescape totality`

Proof. Use the real handoff exhaustion theorem to identify the only exported route. It is the finite concatenation of the tail-choice row, the public route packet, and the real consumer boundary.

For completion-facing reads, apply the completion-budget factorization and the observation-budget consumer package. These results keep the read on the same scheduled regular source, dyadic ledger, diagonal completion row, seal row, transport, continuation, provenance, and naming rows.

Thus the displayed ledgers are total for the downstream consumer surface. A read outside them has no carrier coordinate and therefore cannot be repacked as hidden completion data, an alternate seal, quotient equality, a chosen limit, or host real equality. □

Theorem 386.55 (CauchyLimitSeal RegSeqRat StreamName dyadic handoff)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet routes a consumer from its displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source family through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger before any* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row or* $\mathsf{Real}^{\uparrow}$ *seal row is consumed. The handoff is carried by the same* BHist *surface,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row that appear in the accepted seal packet.*

Proof. Start with the window factorization of Definition 386.3. It places every consumer read on the displayed regular source family, scheduled window, dyadic observation ledger, completion diagonal, and seal row of the carrier.

Next apply the scheduled-window pullback of Definition 386.20. The pullback forces a read that reaches the seal to pass first through the same $\mathsf{StreamName}^{\uparrow}$ window, its $\mathsf{RegSeqRat}^{\uparrow}$ source row, and the matching dyadic ledger.

Finally replay the dyadic observation by Definition 386.21. The replay identifies the observation row as the stored $\mathsf{DyadicRatCore}^{\uparrow}$ ledger with the visible transport, route, provenance, and naming rows, so no completion diagonal or seal row is available before that handoff has occurred. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier dyadic handoff`

Theorem 386.56 (CauchyLimitSeal completion budget factorization)**.** *Any completion-facing or* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet factors through Definition 386.55, the public route boundary, the family ledger exhaustion, and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary. The resulting consumer surface exports only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *history surface,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row; it exports no quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier completion budget factorization`

Proof. First use the handoff theorem above to put the consumer on the scheduled regular-source and dyadic-observation route before any completion or seal row is read.

Next apply the public route boundary of Definition 386.8 and family ledger exhaustion from Definition 386.35. These two constraints keep the read inside the carrier rows and the exhausted source-index, tail-seal, and handoff ledger classes.

Finally combine the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5 with the observation-budget consumer exactness of Definition 611.9. The real-facing and budget-facing consumers therefore read the same finite route surface, with no remaining coordinate for quotient streams, selected limits, host real equality, hidden family limits, or ambient completeness. □

Theorem 386.57 (CauchyLimitSeal root unblock package)**.** *The* CauchyLimitSeal$^{\uparrow}$ *carrier, the* RegSeqRat$^{\uparrow}$–StreamName$^{\uparrow}$–DyadicRatCore$^{\uparrow}$ *handoff of Definition 386.55, the completion-budget factorization of Definition 386.56, the bridge interface, bridge route exactness, and bridge non-escape jointly supply the root-unblock package for downstream* Real$^{\uparrow}$ *and completion consumers over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package gives one cited route from accepted seal packets to the downstream consumer boundary, and it does not add quotient completion, selected limits, host real equality, hidden family limits, or ambient completeness.*

Proof. Begin with the accepted carrier and attach the handoff theorem. This supplies the concrete regular-source, scheduled-window, and dyadic-observation route before the completion diagonal and seal are consumed.

Then compose with the completion-budget factorization. It turns the routed handoff into the consumer-facing surface used by completion and RealObservationBudget$^{\uparrow}$ reads, while preserving only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Finally apply the bridge interface of Definition 386.43, bridge route exactness from Definition 386.44, and bridge non-escape from Definition 386.45. These rows repack the same routed packet as the public bridge package and exclude every hidden completion, quotient, selection, host-equality, or ambient-completeness escape route. □

Theorem 386.58 (CauchyLimitSeal observation-budget pullback)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* CauchyLimitSeal$^{\uparrow}$ *packet, and let* $B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$ *be an accepted* RealObservationBudget$^{\uparrow}$ *packet. If the seal row* S_B *is transported to the seal row* E *of* L*, and the regular handoff rows* R_B *cover the source row* X*, then every budget read of* B *pulls back to the public route packet of* L*. The pulled-back route reads only the finite precision request, windows, dyadic ledger, regular handoff, seal row,* BHist *surface,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates already displayed by* B *and* L*.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier observation budget pullback`

Proof. Start with the finite budget packet. Window coverage from Definition 611.3 places each budget read over the displayed precision request E_B, finite windows W_B, and dyadic ledger D_B.

Next apply the regular handoff and seal boundary. The handoff theorem Definition 611.4 routes the regular rows R_B through the same finite windows, while Definition 611.5 makes S_B the real-facing seal boundary of the budget packet.

Transport the seal row to E and the handoff rows to X. The public route boundary of Definition 386.8 then identifies the resulting read with the route packet X, S, D, G, E, T, P, N of the accepted CauchyLimitSeal$^{\uparrow}$ carrier. □

Theorem 386.59 (CauchyLimitSeal budgeted tail comparison admission)**.** *A family-tail comparison row is admitted through an accepted* CauchyLimitSeal$^{\uparrow}$ *packet only when the same finite* RealObservationBudget$^{\uparrow}$ *packet displays the precision request, finite windows, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local* NameCert *row consumed by the seal packet. A comparison row that lacks any one of those displayed budget coordinates is not an admitted CauchyLimitSeal family-tail comparison row.*

Proof. First use family-tail admission from Definition 386.40. It requires the source family, schedule, tail modulus, null-tail comparison, diagonal completion row, seal row, transports, continuation route, provenance, and local naming row to be displayed in the accepted seal packet.

Next use seal-window ledger compatibility from Definition 386.41. This ties the tail comparison to the same scheduled window and dyadic observation row used by the seal route, so the comparison cannot be read from a different finite budget.

Finally apply $\mathsf{RealObservationBudget}^{\uparrow}$ consumer exactness from Definition 611.9. Exactness says the regular and real-facing consumers read precisely the budget packet, so an admitted tail comparison must carry the same precision, window, ledger, handoff, seal, transport, route, provenance, and naming rows. □

Theorem 386.60 (CauchyLimitSeal RealObservationBudget consumer package)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer that enters through an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet and exits through an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet reads exactly the union of the budget rows and the seal rows. The union is taken over the shared* BHist *surface,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *rows, and it exposes no quotient stream, selected host limit, ambient completeness principle, hidden family limit, or unbounded schedule.*

Proof. Begin with the observation-budget pullback of Definition 386.58. It sends every finite budget read into the public route packet of the accepted seal carrier, preserving the displayed budget rows and the shared transport, route, provenance, and naming data.

Attach the tail comparison by Definition 386.59. A family-tail comparison is admitted only when it is read through the same finite budget surface consumed by the seal route.

Now compose with consumer ledger totality from Definition 386.36 and real consumer family exhaustion from Definition 386.42. The budget route and the seal route already exhaust the consumer-visible rows, leaving no coordinate for quotient streams, selected host limits, ambient completeness, hidden family limits, or unbounded schedules. □

Theorem 386.61 (CauchyLimitSeal RealObservationBudget handoff)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet whose route passes through an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet factors its* $\mathsf{Real}^{\uparrow}$ *endpoint through the shared finite rows*

$$E, W, D, R, S, H, C, P, N$$

of chapter 611, together with the seal rows

$$X, S, D, G, E, T, P, N$$

already displayed by $\mathsf{CauchyLimitSeal}^{\uparrow}$*. The handoff exposes no quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier real observation budget handoff`

Proof. Begin with the observation-budget pullback of Definition 386.58. It sends the accepted budget packet to the public route of the accepted seal packet while preserving the displayed budget rows, seal rows, transports, routes, provenance, and local naming rows.

Next apply budgeted tail comparison admission from Definition 386.59. The tail comparison is admitted only when it uses the same finite $\mathsf{RealObservationBudget}^{\uparrow}$ precision, window, dyadic, regular-handoff, seal, transport, continuation, provenance, and naming rows.

Attach the $\mathsf{RealObservationBudget}^{\uparrow}$ consumer package of Definition 386.60. This identifies the consumer-facing route with the union of the budget packet and the seal packet over the shared BHist, hsame, Cont, Pkg, and NameCert surface.

Finally use ledger non-escape totality from Definition 386.54. The route has no remaining coordinate for a quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness, so projecting the common finite rows gives the stated handoff. □

Theorem 386.62 (CauchyLimitSeal consumer observation-budget readback)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet that requests an observation-budget row must factor through the displayed* $\mathsf{RealObservationBudget}^{\uparrow}$ *handoff, the finite tail selector,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The consumer cannot read an additional hidden budget, quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness witness.*

Proof. Start with the observation-budget pullback of Definition 386.58. It routes every budget-mediated real or completion read into the accepted seal packet while preserving the displayed budget rows, seal rows, transports, continuations, provenance, and local naming rows.

Apply the RealObservationBudget handoff of Definition 386.61. This identifies the $\mathsf{Real}^{\uparrow}$ endpoint with the shared finite budget and seal surface rather than with a detached completion object.

For selector-mediated reads, use the finite-budget selector pullback of Definition 386.63. It keeps the selected tail and real-seal rows on the same finite selector, budget, transport, route, provenance, and naming ledger.

Finally apply the consumer uniform route lock of Definition 386.78. Since the visible route is exhausted by the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, no hidden observation budget remains. □

Theorem 386.63 (CauchyLimitSeal finite-budget selector pullback)**.** *Let F be an accepted* $\mathsf{FiniteObservationBudget}$ *route whose displayed route surface is read over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Suppose the* $\mathsf{Real}^{\uparrow}$ *seal row selected by F is consumed by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and that this budget packet is then consumed by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Then the composite consumer route pulls back to one finite route over exactly the displayed selector, budget, and seal rows. The route exposes no quotient stream, selected host limit, host real equality, hidden family limit, countable choice, or ambient completeness principle.*

Proof. Start with the public export of $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ from Definition 782.14. It makes the selector route a finite packet over the displayed stream name, dyadic rational core, regular rational sequence, real seal row, history, transport, continuation, package, and naming rows.

Next attach the observation-budget pullback of Definition 386.58. The accepted $\mathsf{RealObservationBudget}^{\uparrow}$ consumer can read the selected seal only through its finite precision request, windows, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local naming data.

Now use the RealObservationBudget handoff of Definition 386.61. This identifies the $\mathsf{Real}^{\uparrow}$ endpoint of the composite route with the shared finite budget and seal rows rather than with a detached real-completion object.

Finally apply ledger non-escape totality from Definition 386.54. Since the selector export, budget pullback, and seal handoff have already accounted for every visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinate, no quotient stream, selected host limit, host real equality, hidden family limit, countable-choice witness, or ambient completeness witness remains available to the consumer. □

Definition 386.64 (CauchyLimitSeal RealTailAgreement route)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet*

$$L = (X, S, D, G, E, T, P, N),$$

the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of L is the finite handoff that first projects the admitted tail reads of L through the displayed source family, schedule, dyadic observation ledger, diagonal row, seal*

row, transports, continuation route, provenance, and local naming row, and then reads the paired tail-agreement rows of chapter 699. Its consumer surface is the displayed route

$$X, S, D, G, E, T, P, N \quad \longrightarrow \quad R, S, W, D, A, H, C, P, N,$$

where the right-hand packet is the paired source, finite window, dyadic readback, agreement, transport, continuation, provenance, and naming surface of $\mathsf{RealTailAgreementSeal}^{\uparrow}$. *The route is not a quotient-stream equality, a host real equality, a selected host limit, a hidden tail selector, or an ambient completeness row.*

Theorem 386.65 (CauchyLimitSeal RealTailAgreement handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet that compares two sealed tails factors through the route of Definition 386.64. The comparison may read only the CauchyLimitSeal rows* X, S, D, G, E, T, P, N *followed by the paired* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *rows* $R, S, W, D, A, H, C, P, N$*; it cannot substitute a quotient stream, host real equality, selected host limit, hidden family limit, unbounded schedule, or ambient completeness principle for the displayed agreement row.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier real tail agreement handoff`

Proof. Begin with the $\mathsf{RealObservationBudget}^{\uparrow}$ handoff of Definition 386.61. It places the completion-facing read on the finite budget rows and the accepted seal rows, with the shared BHist, hsame, Cont, Pkg, and NameCert surface already fixed.

Next project the sealed tails by Definition 386.70. The tail reads are therefore the displayed regular-family, schedule, dyadic, diagonal, seal, transport, continuation, provenance, and naming rows of the same accepted packet, not external tails chosen from a quotient or host limit.

Finally attach the paired-window route of Definition 699.52 and the Real–Completion boundary of Definition 699.58. These rows force every paired tail comparison through $W \to D \to A$, so the consumer factors through the route named in Definition 386.64. □

Theorem 386.66 (CauchyLimitSeal regular-tail-fiber consumer boundary)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Any accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer comparing sealed tails through an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet factors through the displayed CauchyLimitSeal rows* X, S, D, G, E, T, P, N*, the paired tail-fiber rows of chapter 893, the tail-meet route of chapter 560, and the common* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface. The route exports no quotient-stream equality, selected host limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness principle.*

Proof. Start with the RealTailAgreement handoff of Definition 386.65. It places the paired sealed-tail comparison on the displayed CauchyLimitSeal rows and the finite paired-tail agreement rows before any consumer-facing comparison is available.

Next apply ledger non-escape totality from Definition 386.54. The accepted packet therefore has no coordinate for quotient representatives, selected host limits, host real equality, hidden family limits, unbounded schedules, or ambient completion data outside the shown BHist, hsame, Cont, Pkg, and NameCert boundary.

Finally use the RegularCauchyTailMeet seal consumer tail-fiber factorization of Definition 560.92. That factorization routes the accepted tail-fiber packet through the paired regular-tail fiber surface and then through the tail-meet route, so the CauchyLimitSeal consumer obtains exactly the finite handoff stated above. □

Theorem 386.67 (CauchyLimitSeal Real consumer budget exhaustion)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Every* Real$^{\uparrow}$*- or completion-facing consumer of L whose route passes through an accepted* RealObservationBudget$^{\uparrow}$ *packet, an accepted* FiniteObservationBudgetSelector$^{\uparrow}$ *route, or the* RealTailAgreementSeal$^{\uparrow}$ *route of Definition 386.64 is exhausted by the displayed* RegSeqRat$^{\uparrow}$ *family,* StreamName$^{\uparrow}$ *schedule,* DyadicRatCore$^{\uparrow}$ *ledger,* RealObservationBudget$^{\uparrow}$ *packet,* RealTailAgreementSeal$^{\uparrow}$ *comparison rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No such consumer can add a quotient stream, selected host limit, host real equality, hidden family limit, countable-choice witness, or ambient completeness principle to that packet-local route.*

Proof. Start with the observation-budget pullback of Definition 386.58 and the RealObservation-Budget handoff of Definition 386.61. They place every budget-mediated real or completion read on the finite budget rows together with the accepted seal rows, over the common BHist, hsame, Cont, Pkg, and NameCert surface.

For selector-mediated reads, apply the finite-budget selector pullback of Definition 386.63. The selected Real$^{\uparrow}$ seal row remains tied to the same StreamName$^{\uparrow}$ schedule, RegSeqRat$^{\uparrow}$ family, DyadicRatCore$^{\uparrow}$ ledger, budget packet, transport rows, routes, provenance, and local naming row.

For paired-tail reads, use the route definition of Definition 386.64 and the handoff of Definition 386.65. The comparison therefore runs through the displayed RealTailAgreementSeal$^{\uparrow}$ rows and their $W \to D \to A$ agreement boundary rather than through an external equality object.

Finally apply ledger non-escape totality from Definition 386.54. The three routes have already accounted for the family, schedule, dyadic ledger, budget packet, paired-tail comparison rows, histories, transports, continuations, package provenance, and naming data, so no quotient stream, selected host limit, host real equality, hidden family limit, countable-choice witness, or ambient completeness coordinate remains available. □

Theorem 386.68 (CauchyLimitSeal diagonal tail selector boundary)**.** *The diagonal tail selector read associated to an accepted* CauchyLimitSeal$^{\uparrow}$ *packet remains a finite scheduled route through chapter 697. It may supply the scheduled window and cofinal tail rows consumed by the seal route, but it does not become a selected host limit, quotient stream representative, host convergence proof, hidden tail selector, or ambient completeness principle.*

Proof. Start with window factorization from Definition 386.3. It places the seal consumer on the displayed source family, schedule, dyadic observation ledger, completion diagonal, seal row, and transport route before any selector read is available.

Use tail extraction from Definition 386.70. The tail exposed to the diagonal route is a projection of the accepted seal packet, so it remains tied to X, S, D, G, E, T, P, N and the visible BHist, hsame, Cont, Pkg, and NameCert rows.

Now apply the DiagonalTailSelector$^{\uparrow}$ limit handoff and non-escape boundary from Definition 697.6 and Definition 697.42. The selector contributes a finite scheduled route only, and the cited non-escape boundary rules out selected host limits, quotient representatives, host convergence, hidden tail selectors, and ambient completeness. □

Theorem 386.69 (CauchyLimitSeal tail-fiber selector factorization)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Any consumer route for L that uses both a common-tail comparison and a diagonal selector factors through the shared finite envelope consisting of the* RegularCauchyTailFiber$^{\uparrow}$ *carrier rows, the* DiagonalTailSelector$^{\uparrow}$ *scheduled and cofinal window rows, the* RegSeqRat$^{\uparrow}$ *source family, the* StreamName$^{\uparrow}$ *schedule, the* DyadicRatCore$^{\uparrow}$ *ledger, the* Real$^{\uparrow}$ *seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorized route exports*

no quotient stream, selected host limit, hidden selector, host real equality, or ambient completeness coordinate.

Proof. Start with the diagonal selector boundary of Definition 386.68. It says that any selector read entering the accepted seal packet contributes only scheduled window and cofinal tail rows, with no selected host limit, quotient representative, host convergence proof, hidden selector, or ambient completeness datum.

Now attach the common-tail fiber side. The completion handoff scope of Definition 893.21 places completion-facing reads on the $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ carrier rows, while the source-seal ledger totality of Definition 893.19 accounts for the source, seal, history, transport, continuation, package, and naming rows.

Finally apply the consumer route lock of Definition 386.78. The seal admits no consumer coordinate beyond the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface. Combining this exhaustion with the selector boundary and the tail-fiber ledger leaves exactly the stated finite envelope. □

Theorem 386.70 (CauchyLimitSeal tail extraction)**.** *Every admitted tail read of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet is projected from the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and visible* hsame *and* Cont *ledger. The projection is the same public route consumed by the seal packet, not a tail chosen from a quotient stream, host limit, hidden family limit, or ambient completeness principle.*

Proof. Begin with family-tail admission. By Definition 386.40, a tail read is admitted only with the displayed source family, schedule, tail modulus, null-tail comparison, completion diagonal, seal row, transports, route, provenance, and local naming row.

Next attach seal-window ledger compatibility from Definition 386.41. The admitted tail therefore uses the same scheduled window and dyadic observation ledger as the seal route.

Finally pull the observation budget back along Definition 386.58. The tail read is projected through the displayed regular family, schedule, dyadic ledger, diagonal row, and visible hsame and Cont route, with no remaining coordinate for an external tail source. □

Theorem 386.71 (CauchyLimitSeal uniqueness boundary)**.** *Every equality or uniqueness-facing consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet may use only the visible null-sequence and tail-comparison route. It cannot import a host uniqueness theorem, quotient-stream equality, selected limit, hidden family limit, or ambient completeness datum.*

Proof. Start with the non-escape boundary of Definition 386.4. The accepted packet contains no host uniqueness theorem, quotient representative, selected limit, or ambient completeness coordinate.

For the comparison itself, use null-sequence factorization from Definition 386.16. Equality-facing reads can add only the displayed $\mathsf{NullSequence}^{\uparrow}$ tolerance-window-threshold row to the already fixed seal route.

Then apply budgeted tail comparison admission from Definition 386.59. A uniqueness-facing read is therefore admitted only through the same finite budget, scheduled window, dyadic ledger, regular handoff, seal, transport, route, provenance, and naming rows. □

Theorem 386.72 (CauchyLimitSeal formal target surface)**.** *The Lean-facing surface for* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is the displayed carrier packet* $L = (X, S, D, G, E, T, P, N)$ *over* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$, *with dependency rows for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$. *Its target obligations are exactly carrier admission, finite-window factorization, consumer readback,*

and exclusion of quotient streams, selected limits, host real equality, hidden family limits, and ambient completeness.

Proof. Use the NameCert obligation surface of Definition 386.2. It names the carrier, classifier stability, exact consumer route, and non-escape clauses as the local obligations.

Then apply window factorization from Definition 386.3 and the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. These rows identify the finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the observation ledger, the diagonal row, and the seal.

Finally pull the family side through Definition 386.30 and package the budget-facing endpoint by Definition 386.60. The resulting surface is the stated collection of concrete BEDC rows and no hidden completion coordinate remains. □

Theorem 386.73 (CauchyLimitSeal finite-route uniqueness envelope)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, a* $\mathsf{Real}^{\uparrow}$*- or completion-facing consumer route is uniqueness-admissible exactly when it factors through the displayed finite envelope consisting of the* $\mathsf{RegSeqRat}^{\uparrow}$ *source family,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row,* $\mathsf{Real}^{\uparrow}$ *seal row, null-comparison and tail-comparison rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. A route that requires a quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness datum is outside the envelope and is not uniqueness-admissible for the packet.*

Proof. For the forward direction, start with tail extraction from Definition 386.70. It projects every admitted tail read onto the displayed regular source family, scheduled window, dyadic observation ledger, completion diagonal, and visible transport route.

Next apply the uniqueness boundary of Definition 386.71. It adds only the visible null-sequence and tail-comparison route to the same finite surface and excludes host uniqueness, quotient-stream equality, selected limits, hidden family limits, and ambient completeness.

Now use ledger non-escape totality from Definition 386.54 and the formal target surface of Definition 386.72. These two rows say that downstream Real-and-Completion consumers have no remaining coordinate beyond the displayed history, transport, continuation, provenance, and naming rows.

Finally attach the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$ handoff of Definition 386.55. The consumer therefore reaches any uniqueness-facing comparison only through the stated finite envelope.

Conversely, any route already factoring through that envelope uses exactly the rows admitted by the tail-extraction, uniqueness-boundary, non-escape, target-surface, and handoff theorems. Replaying those rows gives an admitted packet-local uniqueness route, with no quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness datum to discharge. □

Theorem 386.74 (CauchyLimitSeal paired tail-agreement uniqueness)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, any two* $\mathsf{Real}^{\uparrow}$*- or completion-facing paired-tail agreement reads that factor through the finite-route uniqueness envelope of Definition 386.73 and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of Definition 386.65 expose the same packet-local agreement boundary. Conversely, every packet-local paired-tail uniqueness read for the packet factors through that finite envelope together with the displayed* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff. Thus the only admissible boundary is the one read from the packet's visible source family, schedule, dyadic ledger, completion diagonal, seal row, null-comparison and tail-comparison rows, transports, continuation routes, provenance, and local naming row.*

Proof. First project both paired-tail reads onto the accepted packet surface. Tail extraction in Definition 386.70 places each read on the displayed regular source family, scheduled window, dyadic observation ledger, completion diagonal, seal row, and visible transport route.

Next apply the uniqueness boundary and the finite-route envelope from Definition 386.71 and Definition 386.73. These two rows allow only the packet-local null-comparison and tail-comparison route to contribute to a uniqueness-facing read, while excluding quotient streams, host real equality, selected host limits, hidden family limits, unbounded schedules, and ambient completeness.

Now attach the handoff through Definition 386.65. The paired comparison is therefore read through the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ completion consumer ledger of Definition 699.52 and the Real–Completion boundary of Definition 699.58, so both reads reach the same $W \to D \to A$ agreement row.

Conversely, a packet-local paired-tail uniqueness read has no coordinate outside the accepted packet boundary by the same uniqueness and non-escape clauses. Replaying tail extraction, the finite envelope, and the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ handoff supplies the required factorization through the displayed route, with no additional host or completion datum to discharge. □

Theorem 386.75 (CauchyLimitSeal Lean carrier admission)**.** *A formal* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier is admitted only when its displayed* BHist *packet presents the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *source,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail criterion,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row consumed by the chapter route. A tuple lacking one of these rows is outside the carrier surface.*

Proof. Start from carrier habitation in Definition 386.2. The carrier is the finite packet and its accepted rows are the only rows available for classifier transport.

Use Definition 386.3 to place every consumer-facing read on the schedule, source, observation, diagonal, seal, and transport rows. The family-tail side is then pulled back by Definition 386.30, which supplies the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ criterion without adding an external family limit.

The consumer boundary of Definition 386.5 closes the admission test: if one displayed row is absent, the tuple has no route to the sealed $\mathsf{Real}^{\uparrow}$ endpoint and cannot be admitted as a formal carrier. □

Theorem 386.76 (CauchyLimitSeal verification handoff)**.** *The verification handoff for* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is the finite chain from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows through the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *pullback and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary to the* $\mathsf{RealObservationBudget}^{\uparrow}$ *package. Along this chain all reads preserve the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, so downstream Real and Completion formalization has a single concrete theorem surface to consume.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier verification handoff`

Proof. First apply window factorization and the $\mathsf{Real}^{\uparrow}$ consumer boundary. They route the source schedule and regular source through the observation ledger, completion diagonal, seal, and displayed transport rows.

Next use the uniform-criterion pullback of Definition 386.30. It aligns the family-tail criterion with the same scheduled route and prevents the handoff from selecting an independent host limit.

Finally attach the $\mathsf{RealObservationBudget}^{\uparrow}$ consumer package of Definition 386.60. Together with the carrier obligations of Definition 386.2, it gives the formalization side one route-preserving packet over the displayed history, transport, continuation, provenance, and naming rows. □

Theorem 386.77 (CauchyLimitSeal finite-window budget equivalence)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, and let* $B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$ *be an*

accepted RealObservationBudget$^{\uparrow}$ *packet. Assume that the seal row* S_B *and the regular-handoff rows* R_B *transport to the seal and source rows of* L*. Then a* Real$^{\uparrow}$*- or completion-facing consumer route is admitted through the pair* (L, B) *exactly when it factors through the common finite envelope consisting of the* StreamName$^{\uparrow}$ *window* W_B*, the* RegSeqRat$^{\uparrow}$ *source* X*, the* DyadicRatCore$^{\uparrow}$ *observation ledgers* D *and* D_B*, the* UniformCauchyCriterion$^{\uparrow}$ *tail ledger, the* RealObservationBudget$^{\uparrow}$ *budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*, the* Real$^{\uparrow}$ *seal row* E*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the two packets.*

Proof. For the forward direction, apply the observation-budget pullback of Definition 386.58. It sends every accepted budget read into the public route packet of L while preserving the precision request, finite windows, dyadic ledger, regular handoff, seal row, transports, continuations, provenance, and local naming rows.

Next apply budgeted tail comparison admission from Definition 386.59. The tail-facing comparison is admitted only through the same budget surface, so the UniformCauchyCriterion$^{\uparrow}$ tail ledger, the scheduled StreamName$^{\uparrow}$ window, and the RegSeqRat$^{\uparrow}$ source all remain in the displayed finite envelope.

Use the RealObservationBudget$^{\uparrow}$ consumer package of Definition 386.60 to identify the real or completion consumer route with the union of the budget rows and the seal rows over the shared BHist, hsame, Cont, Pkg, and NameCert surface.

Finally attach the verification handoff of Definition 386.76. This supplies the reusable finite route from StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ through the uniform criterion and the Real$^{\uparrow}$ boundary, so no quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness coordinate remains.

For the reverse direction, suppose the route already factors through the displayed envelope. Each coordinate is one of the rows admitted by the pullback, budgeted tail comparison, consumer package, and verification handoff just cited. Replaying those rows admits the same real or completion consumer route through (L, B), because no additional row is required outside the common finite packet. □

Theorem 386.78 (CauchyLimitSeal consumer uniform route lock)**.** *Any* UniformCauchyCriterion$^{\uparrow}$ *or completion-facing consumer entering through an accepted* CauchyLimitSeal$^{\uparrow}$ *packet must factor through the same displayed* RegSeqRat$^{\uparrow}$ *source,* StreamName$^{\uparrow}$ *scheduled window,* DyadicRatCore$^{\uparrow}$ *ledger,* RealCauchyCompletion$^{\uparrow}$ *diagonal row,* Real$^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The finite-window budget envelope admits no quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier consumer uniform route lock`

Proof. Start from the CauchyLimitSeal handoff surface of Definition 386.2 and the public route packet of Definition 386.8. These rows display the regular source, scheduled window, dyadic observation ledger, completion diagonal, Real seal, transports, continuation routes, provenance, and local naming row as the only consumer-visible entries.

Apply completion-budget factorization through Definition 386.77. It aligns completion-facing consumers with the shared finite envelope rather than with an independent completion or host-limit coordinate.

Use Definition 386.54 and Definition 386.30. These rows force the UniformCauchyCriterion$^{\uparrow}$ read to return through the same schedule, source, dyadic ledger, and displayed transport surface.

Finally compose Definition 386.5 with Definition 386.76. The accepted packet therefore locks every uniform or completion-facing consumer to the stated finite route over BHist, hsame, Cont, Pkg, and NameCert. □

Theorem 386.79 (CauchyLimitSeal budget-route pairing determinacy)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* CauchyLimitSeal$^{\uparrow}$ *packet, and let $B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$ be an accepted* RealObservationBudget$^{\uparrow}$ *packet. If L and B share the same finite window, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local* NameCert *rows, then they determine the same* Real$^{\uparrow}$*- or completion-facing consumer route. No quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness coordinate can distinguish the pair.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier budget route pairing determinacy`

Proof. Begin with the observation-budget pullback of Definition 386.58. It sends the accepted budget packet into the public route packet of L over the shared finite window, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local naming rows.

Apply budgeted tail comparison admission from Definition 386.59. The admitted comparison must use the same budget coordinates, so a tail-facing read cannot separate L from B after those rows agree.

Next attach the RealObservationBudget$^{\uparrow}$ consumer package of Definition 386.60. It identifies the real or completion consumer surface with the union of the budget and seal rows over the shared BHist, hsame, Cont, Pkg, and NameCert boundary.

Finally apply the consumer uniform route lock of Definition 386.78. The visible route is already exhausted by the displayed family, schedule, dyadic ledger, budget, seal, transport, continuation, provenance, and naming rows, leaving no quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness coordinate by which the pair could diverge. □

Theorem 386.80 (CauchyLimitSeal selector-seal consumer route)**.** *When a* RealCompletionSelectorSeal$^{\uparrow}$ *packet reads row L as a* CauchyLimitSeal$^{\uparrow}$ *consumer row, the consumer route factors through the existing CauchyLimitSeal family schedule, dyadic ledger, regular handoff, seal row, transport row, provenance row, and local* NameCert *row before the terminal* Real$^{\uparrow}$ *row is read. Equivalently, row L can be consumed only as the displayed finite packet over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RealCauchyCompletion$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

No selector-seal packet can use row L to read a detached host limit, quotient stream equality, hidden family limit, unbounded schedule, independent completion row, or ambient completeness principle before reaching the terminal Real$^{\uparrow}$ *endpoint.*

Proof. Apply Definition 386.79. It identifies any budgeted consumer read with the same finite window, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local naming rows whenever those displayed coordinates agree.

Use Definition 386.53 to place the Real$^{\uparrow}$ endpoint after the tail-choice exhaustion row, public route packet, and Real consumer boundary. This supplies the terminal row only after the CauchyLimitSeal route spine has already been consumed.

The selector-seal packet reads L exactly at that consumer position. Since determinacy fixes the budget-route pair and real handoff exhaustion leaves no extra exported coordinate, L is confined to the listed family schedule, dyadic ledger, regular handoff, seal, transport, provenance, and local NameCert rows before E is read. □

Theorem 386.81 (CauchyLimitSeal selector-seal public handoff)**.** *When a* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet consumes row* L *through the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer surface, the public handoff factors first through the displayed CauchyLimitSeal selector-seal route of Definition 386.80 and then through the ordered terminal selector-seal route*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

The composed public handoff exports no quotient stream, host real equality, hidden selector, detached limit, second completion route, or ambient completeness principle.

Proof. First apply Definition 386.80. It places the L read on the existing $\mathsf{CauchyLimitSeal}^{\uparrow}$ finite packet before the terminal $\mathsf{Real}^{\uparrow}$ row is consumed.

Next use Definition 1081.5. The sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ positions are exactly the public rows W, R, L, E inside the selector-seal handoff.

Now attach Definition 1081.10 and Definition 1081.11. These results force every terminal consumer through the ordered route $B, W, R, L, E, H, C, P, N$ and identify that route as the whole terminal selector-seal surface.

Finally apply Definition 1081.17. The structural rows H, C, P, N package the route already displayed and cannot promote a quotient stream, host real equality, hidden selector, detached limit, second completion route, or ambient completeness principle into the public handoff. □

Theorem 386.82 (CauchyLimitSeal consumer readback)**.** *A downstream* $\mathsf{Real}^{\uparrow}$ *or completion consumer reads back exactly the sealed route assembled by the root-unblock package, together with the tail-extraction row of Definition 386.70 and the uniqueness boundary of Definition 386.71. The readback exports only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Apply the root-unblock package of Definition 386.57. It gives the consumer one route from the accepted seal packet to the downstream $\mathsf{Real}^{\uparrow}$ or completion boundary.

Attach tail extraction. The tail row is already projected from the same displayed regular family, schedule, observation ledger, completion diagonal, and transport ledger, so it does not add a second consumer route.

Finally attach the uniqueness boundary. Equality-facing reads remain on the null-sequence and tail-comparison route admitted by the packet, and the readback has no coordinate for quotient completion, host uniqueness, selected limits, hidden family limits, or ambient completeness. □

Theorem 386.83 (CauchyLimitSeal RegSeqRat StreamName route lock)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet consumed by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing route factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, the* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer boundary of chapter 360. The factorization uses only the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows already exposed by the accepted seal route.*

Consequently the accepted consumer route exports no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completion row, or standard bridge row outside that finite L10 route.

Proof. Start with the consumer readback theorem, Definition 386.82. It confines every downstream $\mathsf{Real}^{\uparrow}$ or completion consumer to the accepted seal packet, its root-unblock route, tail-extraction row, uniqueness boundary, and the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Next attach the finite route theorems already exposed on the Real side. The four-face canonical route of Definition 38.450 and the seal lattice of Definition 38.451 force any completion-facing $\mathsf{Real}^{\uparrow}$ consumer carrying a $\mathsf{CauchyLimitSeal}^{\uparrow}$ row through the ordered $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces before the seal row is read.

Finally use the terminal comparison exhaustion of Definition 38.446 together with the $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer boundary cited above. These rows keep the completion-facing endpoint on the same displayed histories, transports, continuations, provenance, and local names, so no quotient stream, host sequence equality, selected limit, ambient completeness, metric-completion coordinate, or standard bridge coordinate remains available to the consumer. □

Theorem 386.84 (Real-completion seal selector consumer factorization)**.** *Let an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer be routed through an accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet and an accepted* ObserverFilterBudgetSel *budget packet for the current Real and Completion handoff. Then its consumer read factors through the concrete displayed chain*

$$\mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization uses only the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the three accepted packets. It exports no quotient stream, host real equality, hidden selector, detached limit, or ambient completeness coordinate.*

Proof. Start with the route lock of Definition 386.83. It confines the $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer to the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, with only BHist, hsame, Cont, Pkg, and NameCert support.

Next attach the selector-seal packet. The L10 handoff scope of Definition 1081.5 identifies the selector-seal consumer as the same finite route through selector budget, selected window, regular readback, limit-seal handoff, and terminal $\mathsf{Real}^{\uparrow}$ endpoint.

Finally attach the observer-budget packet. The bridge-ready route of Definition 1042.9 supplies the selected observer row, finite budget, selected $\mathsf{StreamName}^{\uparrow}$ window, dyadic ledger, regular handoff, and Real seal in that order. Combining the three displayed routes gives the stated factorization, and the coordinate exclusions in the cited theorems leave no quotient stream, host real equality, hidden selector, detached limit, or ambient completeness row for the consumer. □

Theorem 386.85 (CauchyLimitSeal finite-request section compatibility)**.** *Let P be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet whose public route is consumed by a finite request section S for the current Real and Completion handoff. Then the accepted consumer boundary is the same finite* BHist *packet whether it is read through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window section, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback section, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger, the local* $\mathsf{Real}^{\uparrow}$ *seal row, or the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *public route packet:*

$$S_{\mathsf{StreamName}^{\uparrow}} = S_{\mathsf{RegSeqRat}^{\uparrow}} = S_{\mathsf{DyadicRatCore}^{\uparrow}} = S_{\mathsf{Real}^{\uparrow}} = S_{\mathsf{CauchyLimitSeal}^{\uparrow}}$$

as displayed finite BHist *sections over the visible* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The compatibility exports no quotient stream equality, host real equality, selected limit, countable-choice schedule, ambient completeness principle, metric-completion row, or standard bridge row.*

Proof. Start with the route lock of Definition 386.83. It confines the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer to the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, so every face of S is read from the same public route packet.

Next read the Real-facing side through Definition 38.698 and Definition 38.704. These theorems identify the $\mathsf{Real}^{\uparrow}$ seal request with one finite BHist packet carrying the StreamName window, dyadic endpoint-radius ledger, RegSeqRat classifier and readback rows, componentwise hsame transport, displayed Cont route, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ row.

Now read the StreamName-facing side through Definition 126.577 and Definition 126.586. The pullback section has the same finite window face and the same dyadic, RegSeqRat, and Real seal faces, so it cannot introduce a second selector or a second completion-facing section.

Finally apply the non-escape boundary of Definition 386.4. Since the accepted packet has no coordinate for quotient stream equality, host real equality, selected limits, countable-choice schedules, ambient completeness, metric-completion rows, or standard bridge rows, the only consumer boundary left is the displayed finite BHist packet shared by all five readings. □

Theorem 386.86 (CauchyLimitSeal downstream consumer coverage)**.** *Let an accepted local* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer be read by a downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer through the selected* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *and* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packets of the current handoff. Then the consumer read is exhausted by the displayed finite route rows*

$$\mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

together with the $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer boundary, the tail-extraction row, the uniqueness boundary, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows already named in the selected packet.*

Equivalently, every accepted use of the local consumer is one of the listed selected observer, finite budget, finite window, dyadic ledger, regular readback, limit-seal handoff, terminal Real endpoint, completion-boundary, tail, comparison, transport, replay, provenance, or naming rows. A purported quotient stream, host sequence equality, host real equality, hidden selector, detached limit, selected host limit, hidden family limit, ambient completeness principle, metric-completion row, standard bridge row, arithmetic package, or order package is outside the displayed BEDC packet and is not exported by the consumer readback.

Proof. Start with the consumer factorization of Definition 386.84. It places any accepted local $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer that enters through the selected selector-seal and observer-budget packets on the concrete route

$$\mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The same theorem records that the route uses only the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows of the accepted packets, and that quotient streams, host real equality, hidden selectors, detached limits, and ambient completeness have no exported coordinate there.

Now apply the route lock of Definition 386.83. It places the local $\mathsf{CauchyLimitSeal}^{\uparrow}$ portion of the same read on the displayed $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{Real}^{\uparrow}$ seal row, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer boundary, with the componentwise transport row supplied by Definition 386.7. Thus the selector route cannot open a second local consumer surface; it can only read the same finite window, ledger, readback, seal, and completion-boundary rows.

The remaining local readback rows are already fixed by Definition 386.82. Tail-facing use is projected by Definition 386.70, and equality-facing use is confined by Definition 386.71. The transport component for those tail and comparison rows is the packet-local public route transport of Definition 386.7. These two rows account for the admitted tail and comparison alternatives

without adding quotient completion, host uniqueness, selected limits, hidden family limits, or ambient completeness.

Combining the factorized selector route with the locked local route and the readback tail and comparison rows along the public-route transport of Definition 386.7 gives the announced coverage list. Any purported downstream coordinate not on that list would have to bypass either the displayed selector chain, the locked $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$–$\mathsf{RealCauchyCompletion}^{\uparrow}$ surface, or the packet-local tail and uniqueness rows. Each bypass is excluded by the cited theorems, so the unlisted alternatives remain outside the BEDC packet and cannot be exported by the consumer readback. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet is generated from a displayed $\mathsf{RegSeqRat}^{\uparrow}$ family, a $\mathsf{StreamName}^{\uparrow}$ schedule, a DyadicRatCore observation ledger, a $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row, a $\mathsf{Real}^{\uparrow}$ seal row, finite hsame and Cont transports, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature surface consists of the sealed $\mathsf{Real}^{\uparrow}$ public route, family-window pullback, family ledger exhaustion, source factorization, bridge interface, bridge route exactness, and bridge non-escape over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient Cauchy completion, uniqueness of limits, host real construction, or ambient real completeness.

Upgrade path: no further upgrade – chapter at top of formal axis

Theorem 386.87 (CauchyLimitSeal budget root source totality)**.** *Let*

$$L = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, and let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet whose seal and regular-handoff rows are routed through* L*. Every* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing read accepted through this route is sourced exactly by the displayed seal rows* X, S, D, G, E, T, P, N *together with the displayed budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*. No further budget source, hidden completion row, quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate is available to the read.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier budget root source totality`

Proof. Start with the observation-budget pullback of Definition 386.58. It sends each budget read of B into the public route packet of L while preserving the precision request, window, dyadic ledger, regular handoff, seal row, transport, route, provenance, and naming rows displayed by B.

Next attach the $\mathsf{RealObservationBudget}^{\uparrow}$ consumer package from Definition 386.60. The consumer endpoint reads the union of the budget rows and the seal rows over the shared BHist, hsame, Cont, Pkg, and NameCert surface.

Finally use tail extraction from Definition 386.70 and the consumer exactness of Definition 611.9. These rows force the tail and budget consumers to remain on the same displayed packet, so a read outside X, S, D, G, E, T, P, N and $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$ has no source coordinate. □

Theorem 386.88 (CauchyLimitSeal root dyadic window exactness)**.** *In the setting of Definition 386.87, the dyadic ledger read by any accepted budget route is exactly the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window ledger already exposed by the seal packet. The budget ledger* D_B *and the seal-side dyadic row* D *are not independent observations; they are the same finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger transported through the displayed* hsame *and* Cont *rows of the route.*

Proof. First pull the budget route back along Definition 386.58. The pullback places the precision request, windows, and dyadic ledger of B on the public route packet of L.

Then apply tail extraction from Definition 386.70. The tail read of the seal packet is projected through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ family, $\mathsf{StreamName}^{\uparrow}$ schedule, and $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, with no external tail source.

The $\mathsf{RealObservationBudget}^{\uparrow}$ consumer package of Definition 386.60 and the exactness theorem Definition 611.9 identify the regular and real-facing budget consumers with that same packet. Hence the budget route reads precisely the scheduled window ledger already exposed by L. □

Theorem 386.89 (CauchyLimitSeal budget window source refusal)**.** *In the setting of Definition 386.87, any route that presents a budget dyadic window not identified with the scheduled seal-side* $\mathsf{StreamName}^{\uparrow}$ *window of Definition 386.88 is not a public* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing read of the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet.*

Proof. Apply source totality from Definition 386.87. It confines every accepted budget-facing read to the displayed seal rows and the displayed budget rows, with no hidden completion coordinate or independent source row.

Next apply dyadic-window exactness from Definition 386.88. The budget dyadic ledger is the scheduled seal-side $\mathsf{StreamName}^{\uparrow}$ window ledger transported through the displayed hsame and Cont rows.

Therefore a route that presents a different budget window has no source coordinate among the displayed seal and budget rows. It is not a public $\mathsf{RealObservationBudget}^{\uparrow}$-facing read of the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet. □

Theorem 386.90 (CauchyLimitSeal budget selector coherence)**.** *In the setting of Definition 386.87, let an accepted selector route be admitted through the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet as an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *read over the same precision request and finite* $\mathsf{StreamName}^{\uparrow}$ *window. Then the selector read and the observation-budget read meet on one displayed route over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Their selector leg and budget leg agree on the dyadic-window ledger identified by Definition 386.88, and the route exports no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier budget selector coherence`

Proof. First take the root choice-free route from Definition 386.111. It confines selector access to the accepted seal packet and its displayed regular source, stream window, dyadic ledger, real seal, transport, continuation, provenance, and naming rows.

Next apply the tail-budget meet of Definition 386.135 together with budget root source totality from Definition 386.87. These rows project the selector read and the observation-budget read to the same finite packet over the shared precision request and $\mathsf{StreamName}^{\uparrow}$ window.

Then use the dyadic-window exactness theorem Definition 386.88. It identifies the budget ledger with the scheduled seal-side dyadic window, so the selector leg and budget leg read one $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than two independent ledgers.

Finally apply completion-consumer non-escape from Definition 386.94. Since the common route has already been projected to the displayed finite packet, no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate can be exported. □

Theorem 386.91 (CauchyLimitSeal budget finite-approximation route)**.** *In the setting of Definition 386.87, let an accepted* FiniteCauchyApproximation$^{\uparrow}$ *packet read the same precision request and the same finite* StreamName$^{\uparrow}$ *window as the accepted* RealObservationBudget$^{\uparrow}$ *route. Then the observation-budget route and the finite-approximation route meet on one displayed packet over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The shared packet exports the precision request, finite window, dyadic observation ledger, finite error consumer row, real-facing handoff, transport, continuation, provenance, and local name, but it exports no quotient stream identity, host real equality, selected limit, hidden family limit, or ambient completeness coordinate.*

Proof. First apply budget selector coherence from Definition 386.90. It places the selector leg and the observation-budget leg on the same displayed route over the accepted CauchyLimitSeal$^{\uparrow}$ packet, with the precision request and finite StreamName$^{\uparrow}$ window already fixed.

Next use dyadic-window exactness from Definition 386.88. The dyadic ledger read by the budget route is the scheduled finite-window ledger exposed by the seal packet, so the finite approximation cannot introduce an independent DyadicRatCore$^{\uparrow}$ observation.

Finally use the finite consumer surface of chapter 601. Its accepted FiniteCauchyApproximation$^{\uparrow}$ packet consumes the same source, precision request, finite window, dyadic ledger, finite error row, handoff, transport, continuation, provenance, and naming rows. The non-escape boundary from the selector-coherence route therefore remains in force, excluding quotient stream identity, host real equality, selected limit, hidden family limit, and ambient completeness. □

Theorem 386.92 (CauchyLimitSeal budget selector approximation determinacy)**.** *In the setting of Definition 386.87, suppose an accepted selector route, an accepted* RealObservationBudget$^{\uparrow}$ *route, and an accepted* FiniteCauchyApproximation$^{\uparrow}$ *route read the same precision request and the same finite* StreamName$^{\uparrow}$ *window through one accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Then the selector leg, the observation-budget leg, and the finite-approximation leg determine the same displayed dyadic-window ledger and the same finite packet over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The common packet contains only the displayed precision request, finite window, dyadic ledger, finite error row, real-facing handoff, transport, continuation, provenance, and local naming rows; it contains no quotient stream identity, host real equality, selected limit, hidden family limit, or ambient completeness coordinate.*

Proof. First apply budget selector coherence from Definition 386.90. The selector route and the RealObservationBudget$^{\uparrow}$ route are forced onto one accepted CauchyLimitSeal$^{\uparrow}$ packet over the shared precision request and finite StreamName$^{\uparrow}$ window.

Next use dyadic-window exactness from Definition 386.88. It identifies the budget ledger with the scheduled seal-side DyadicRatCore$^{\uparrow}$ window, so the selector leg and the observation-budget leg cannot display different dyadic ledgers.

Then apply the budget finite-approximation route of Definition 386.91. The accepted FiniteCauchyApproximation route consumes that same precision request, window, dyadic ledger, finite error row, real-facing handoff, transport, continuation, provenance, and NameCert row, placing its finite packet on the already displayed route.

Finally use completion-consumer non-escape from Definition 386.94. After the three reads have been identified with the same finite packet, there is no remaining coordinate from which quotient stream identity, host real equality, selected limit, hidden family limit, or ambient completeness could be exported. □

Theorem 386.93 (CauchyLimitSeal root budget finite-approximation cofinality)**.** *In the setting of Definition 386.87, suppose an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *route refines the shared precision request to a stronger displayed dyadic window through the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Then the accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *route and the selector route factor through that stronger finite-approximation packet. The factorization preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows already displayed by the seal route, and it exports no quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate.*

Proof. First use dyadic-window exactness from Definition 386.88. The budget ledger is the scheduled $\mathsf{StreamName}^{\uparrow}$ dyadic window ledger carried by the seal packet, so a stronger displayed window is a refinement of the same ledger rather than an independent observation.

Next apply budget-selector coherence from Definition 386.90. It puts the selector route and the $\mathsf{RealObservationBudget}^{\uparrow}$ route on one displayed route over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Then use the budget finite-approximation route of Definition 386.91. The stronger finite packet imports the refined precision request, dyadic window, finite error row, real-facing handoff, transport, continuation, provenance, and local name without changing the seal source rows.

Finally apply completion-consumer non-escape from Definition 386.94. Since the budget leg, selector leg, and refined finite-approximation leg already factor through one finite packet, there is no exported coordinate for quotient streams, selected limits, host real equality, hidden family limits, or ambient completeness. □

Theorem 386.94 (CauchyLimitSeal root completion consumer non-escape)**.** *The completion-facing consumer obtained by composing an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *route with an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet can export only the rows listed in Definition 386.87 and the dyadic-window identification of Definition 386.88. It cannot export quotient streams, selected limits, host real equality, hidden family limits, or ambient completeness.*

Proof. Use Definition 386.87 to identify the complete source surface of the composed consumer. The consumer has only the displayed seal rows and the displayed budget rows.

Use Definition 386.88 for the window ledger. The dyadic observation consumed by the route is the scheduled $\mathsf{StreamName}^{\uparrow}$ window ledger already carried by the seal packet, not a second ledger chosen by the completion consumer.

Finally apply the consumer package of Definition 386.60, tail extraction from Definition 386.70, and $\mathsf{RealObservationBudget}^{\uparrow}$ consumer exactness from Definition 611.9. The composed route remains inside the finite packet, so none of quotient streams, selected limits, host real equality, hidden family limits, or ambient completeness can be exported. □

Theorem 386.95 (CauchyLimitSeal root formal target package)**.** *The source totality row of Definition 386.87, the dyadic-window exactness row of Definition 386.88, the budget-selector coherence row of Definition 386.90, and the completion-consumer non-escape row of Definition 386.94 jointly form the formal target package for the composed* $\mathsf{RealObservationBudget}^{\uparrow}$*-to-*$\mathsf{CauchyLimitSeal}^{\uparrow}$ *route. Source totality fixes all source rows, dyadic-window exactness identifies the budget ledger with the*

scheduled $\mathsf{StreamName}^{\uparrow}$ *window, budget-selector coherence binds selector and budget reads to that one ledger, and completion-consumer non-escape excludes quotient streams, selected limits, host real equality, hidden family limits, and ambient completeness.*

Proof. First use source totality. It fixes the composed route to the displayed seal rows X, S, D, G, E, T, P, N and the displayed budget rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, so every formal target read has a finite source coordinate.

Next use dyadic-window exactness. It identifies the budget ledger D_B with the scheduled $\mathsf{StreamName}^{\uparrow}$ window ledger already carried by the seal packet, so the route has one dyadic observation surface rather than two independent ledgers.

Then apply budget-selector coherence. The selector leg and the observation-budget leg meet on the same displayed route over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Finally use completion-consumer non-escape. Once the source rows, the dyadic ledger, and the selector-budget handoff have been fixed, the completion-facing consumer has no remaining coordinate for quotient streams, selected limits, host real equality, hidden family limits, or ambient completeness. These four rows therefore give the complete formal target package for the composed route. □

Theorem 386.96 (CauchyLimitSeal budget finite-approximation window exhaustion)**.** *In the setting of Definition 386.87, suppose an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *route, and an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet read the same precision request and the same finite* $\mathsf{StreamName}^{\uparrow}$ *window. Then every completion-facing read of that route factors through the displayed seal rows* X, S, D, G, E, T, P, N*, the displayed budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*, the dyadic-window ledger of Definition 386.88, and the finite-approximation packet rows of Definition 386.91. No independent window, quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate is exported.*

Proof. Begin with source totality from Definition 386.87. It confines the route to the displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows and $\mathsf{RealObservationBudget}^{\uparrow}$ rows, so any completion-facing read already has a finite source surface.

Next use dyadic-window exactness from Definition 386.88. The budget ledger is the scheduled finite $\mathsf{StreamName}^{\uparrow}$ window ledger exposed by the seal packet, so the route cannot introduce an independent window.

Now apply the budget finite-approximation route of Definition 386.91. The accepted $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ packet consumes the same precision request, finite window, dyadic ledger, finite error row, real-facing handoff, transport, continuation, provenance, and local name, placing its rows on the already displayed route.

Use selector approximation determinacy from Definition 386.92. The selector leg, observation-budget leg, and finite-approximation leg determine the same displayed dyadic-window ledger and the same finite packet over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Finally apply root completion consumer non-escape from Definition 386.94. Since the completion-facing consumer has already been forced through the seal rows, budget rows, dyadic-window ledger, and finite-approximation rows, it has no coordinate for an independent window, quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness. □

Theorem 386.97 (CauchyLimitSeal budget source uniqueness)**.** *In the setting of Definition 386.87, suppose two accepted* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing reads are routed through the same accepted*

$\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. If the two reads agree on the displayed seal rows* X, S, D, G, E, T, P, N*, the displayed budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*, the precision request, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, and the dyadic-window ledger identified by Definition 386.88, then they are the same public source route over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No second source coordinate, independent window, quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate is available.*

Proof. First apply source totality from Definition 386.87. It confines both accepted budget-facing reads to the displayed seal rows and displayed budget rows of the same accepted packet.

Next use dyadic-window exactness from Definition 386.88. The budget ledger of each read is the scheduled finite $\mathsf{StreamName}^{\uparrow}$ ledger already exposed by the seal packet, so the two reads cannot choose different dyadic-window ledgers once those displayed rows agree.

Now use selector coherence and finite-approximation routing. The coherence row of Definition 386.90 fixes the selector and budget legs on one displayed route, and Definition 386.91 places the finite approximation on the same packet.

Apply finite-approximation window exhaustion from Definition 386.96. Every completion-facing read is exhausted by the displayed seal rows, budget rows, dyadic-window ledger, and finite-approximation packet rows.

Finally apply root completion consumer non-escape from Definition 386.94. Since no second source coordinate, independent window, quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate is exported, the two reads name the same public source route over the displayed BEDC rows. □

Theorem 386.98 (CauchyLimitSeal tail schedule budget cofinality)**.** *In the setting of Definition 386.87, suppose the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet also consumes an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet for the same unary precision request and the same finite* $\mathsf{StreamName}^{\uparrow}$ *window. Then the selector route, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *route, the* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *route, and the tail schedule determine one cofinal displayed packet over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The cofinal packet exports no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate.*

Proof. First bind the selector and budget legs. Apply Definition 386.90 to place the selector route and the $\mathsf{RealObservationBudget}^{\uparrow}$ route on one displayed packet over the shared precision request, finite window, dyadic ledger, real handoff, transports, continuations, provenance, and local name.

Next attach the finite-approximation leg. Use Definition 386.91 and Definition 386.93 to identify every stronger finite window with a refinement of the same seal-side dyadic ledger rather than a second observation source.

Now consume the schedule. The real-seal readback and scoped binding rows of $\mathsf{TailCofinalitySchedule}^{\uparrow}$, namely Definition 681.6 and Definition 681.8, route the same unary request through the finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ threshold ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal before any completion-facing read.

Finally apply Definition 386.96. The selector, budget, finite-approximation, and schedule faces have all been forced through the same displayed packet, so no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate remains available. □

Theorem 386.99 (CauchyLimitSeal budget selector totality)**.** *In the setting of Definition 386.87, suppose an accepted selector route, an accepted* RealObservationBudget$^{\uparrow}$ *route, an accepted* FiniteCauchyApproximatic *packet, and an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet read the same unary precision request and the same finite* StreamName$^{\uparrow}$ *window through one accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Then the four legs determine one displayed L10 packet over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Every completion-facing read is exhausted by that displayed packet, and no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate is exported.*

Proof. First bind the selector and observation-budget legs by Definition 386.90. They read one precision request, one finite StreamName$^{\uparrow}$ window, and one dyadic ledger through the accepted CauchyLimitSeal$^{\uparrow}$ packet.

Next attach the finite-approximation leg using Definition 386.91. The finite packet is placed on the same RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Use selector approximation determinacy from Definition 386.92. Once the selector, budget, and finite-approximation legs share the request and window, their dyadic ledger and finite packet cannot split into separate public routes.

Now consume the schedule by Definition 386.98. The accepted TailCofinalitySchedule$^{\uparrow}$ packet refines the same finite window and does not supply a second tail oracle or a second completion-facing source.

Finally apply root completion consumer non-escape from Definition 386.94. Since every leg has already been forced through the displayed L10 packet, a completion-facing consumer has no remaining coordinate for quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness. □

Theorem 386.100 (CauchyLimitSeal budget tail selector lock)**.** *In the setting of Definition 386.87, suppose an accepted selector route, an accepted* RealObservationBudget$^{\uparrow}$ *route, an accepted* FiniteCauchyApproximatic *packet, and an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet read the same unary precision request and the same finite* StreamName$^{\uparrow}$ *window through one accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Then these four displayed legs determine exactly one tail-selector row over* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The locked row is the same row seen by the selector, the observation budget, the finite approximation, and the tail schedule, and it has no hidden tail oracle, quotient stream identity, selected limit, host real equality, hidden family limit, or ambient completeness coordinate.*

Proof. Begin with budget selector totality from Definition 386.99. It places the selector, observation-budget, finite-approximation, and tail-schedule legs on one displayed L10 packet over the named BEDC rows.

Next use dyadic-window exactness from Definition 386.88 and selector coherence from Definition 386.90. These two rows identify the selector leg and the budget leg with the same finite StreamName$^{\uparrow}$ window and DyadicRatCore$^{\uparrow}$ ledger.

Attach the finite packet through Definition 386.91. Then consume the schedule using Definition 386.98. The finite approximation and tail schedule therefore read the same tail-selector row rather than separate endpoint rows.

Finally apply root completion consumer non-escape from Definition 386.94. Since no external coordinate remains available, the shared tail-selector row is locked by the displayed selector, budget, finite-approximation, and schedule data. □

Theorem 386.101 (CauchyLimitSeal budget source endpoint exhaustion)**.** *In the setting of Definition 386.87, every completion-facing endpoint read accepted through the budget route is exhausted by the displayed seal rows X, S, D, G, E, T, P, N, the displayed budget rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, the dyadic-window ledger identified in Definition 386.88, the finite-approximation packet of Definition 386.91, and the tail schedule of Definition 386.98. No further endpoint, completion source, quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate is part of the public route.*

Proof. Start with budget root source totality from Definition 386.87. It confines every accepted budget-facing read to the displayed seal rows and displayed budget rows.

Next apply dyadic-window exactness from Definition 386.88. The completion-facing endpoint reads the scheduled finite $\mathsf{StreamName}^{\uparrow}$ window ledger already carried by the seal route, not an independent dyadic window.

Now attach the finite side by Definition 386.91 and Definition 386.96. These rows place the finite-approximation packet on the same displayed endpoint package.

Use tail-schedule budget cofinality from Definition 386.98 and the tail-selector lock of Definition 386.100. The schedule contributes only the displayed cofinal tail row over the same packet.

Finally apply completion-consumer non-escape from Definition 386.94. With the seal rows, budget rows, dyadic ledger, finite packet, and tail schedule already accounting for the endpoint, no additional completion-facing coordinate remains. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier budget source endpoint exhaustion`

Theorem 386.102 (CauchyLimitSeal completion route uniqueness)**.** *In the setting of Definition 386.87, suppose two accepted completion-facing routes through an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet agree on the displayed seal rows, the displayed* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows, the dyadic-window ledger, the* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet, and the* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *row. Then the two routes name the same public completion route over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *There is no second public route distinguished only by a hidden endpoint, quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate.*

Proof. First use budget source uniqueness from Definition 386.97. Agreement on the displayed seal and budget rows, together with the shared precision request, finite window, and dyadic ledger, identifies the public source route.

Next use budget selector totality from Definition 386.99. It puts the selector, budget, finite-approximation, and tail-schedule faces of both accepted routes on the same displayed L10 packet.

Apply endpoint exhaustion from Definition 386.101. Every completion-facing read of either route is exhausted by the seal rows, budget rows, dyadic ledger, finite-approximation packet, and tail schedule.

Finally apply root completion consumer non-escape from Definition 386.94. Since no hidden completion coordinate can distinguish the two accepted reads, the agreed displayed endpoint package names one public completion route. □

Theorem 386.103 (CauchyLimitSeal budget RealTailAgreement terminality)**.** *In the setting of Definition 386.87, suppose an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reads the same regular source rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic ledger, and* $\mathsf{Real}^{\uparrow}$ *seal surface as the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget route. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through one displayed packet over* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$,

RealTailAgreementSeal$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The terminal packet exports no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate.*

Proof. Begin with budget selector totality from Definition 386.99. It places the observation budget, finite-approximation packet, and tail-schedule row on the same displayed route as the accepted CauchyLimitSeal$^{\uparrow}$ budget packet.

Next apply endpoint exhaustion from Definition 386.101 and completion route uniqueness from Definition 386.102. These rows force every completion-facing read through the same finite endpoint package and identify any two routes that agree on the displayed budget, window, dyadic, approximation, and schedule rows.

Attach the tail-agreement seal by Definition 699.94. The accepted RealTailAgreementSeal$^{\uparrow}$ packet reads the same regular source, finite window, dyadic ledger, and Real$^{\uparrow}$ seal surface, so it is a consumer of the route already fixed by the budget packet.

Finally use Definition 699.95. The Real and Completion consumers of the tail-agreement packet are confined to the same displayed budget route. Since all listed rows have already been accounted for, quotient stream identity, selected limits, host real equality, hidden family limits, hidden tail oracles, and ambient completeness remain outside the terminal packet. □

Theorem 386.104 (CauchyLimitSeal terminal route exhaustion)**.** *In the setting of Definition 386.87, suppose accepted* RealObservationBudget$^{\uparrow}$, FiniteCauchyApproximation$^{\uparrow}$, TailCofinalitySchedule$^{\uparrow}$, *and* RealTailAgreementSeal$^{\uparrow}$ *legs read the same regular source rows, the same finite* StreamName$^{\uparrow}$ *window, the same* DyadicRatCore$^{\uparrow}$ *ledger, and the same* Real$^{\uparrow}$ *seal through one accepted* CauchyLimitSeal$^{\uparrow}$ *packet. Then every* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer is exhausted by that one displayed public route over* RealObservationBudget$^{\uparrow}$, FiniteCauchyApproximation$^{\uparrow}$, TailCofinalitySchedule$^{\uparrow}$, RealTailAgreementSeal$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The route exports no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, independent completion endpoint, or ambient completeness coordinate.*

Proof. First apply budget selector totality from Definition 386.99. The observation budget, finite approximation, and tail schedule are placed on one displayed L10 packet over the shared regular source rows, finite StreamName$^{\uparrow}$ window, dyadic ledger, Real$^{\uparrow}$ handoff, history, transport, continuation, provenance, and naming rows.

Next apply endpoint exhaustion from Definition 386.101. This confines every completion-facing endpoint read to the displayed seal rows, budget rows, dyadic-window ledger, finite-approximation packet, and tail schedule.

Now use completion route uniqueness from Definition 386.102. Since the accepted legs agree on the displayed budget, window, dyadic, approximation, and schedule rows, there is only one public completion route over the named BEDC surface.

Attach the RealTailAgreement face by Definition 386.103. Its accepted packet reads the same regular source rows, finite window, dyadic ledger, and Real$^{\uparrow}$ seal, so its Real and Completion consumers are terminals of the same displayed route rather than separate endpoints.

Finally apply root completion consumer non-escape from Definition 386.94. With all four legs already bound to the accepted CauchyLimitSeal$^{\uparrow}$ packet, no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, independent completion endpoint, or ambient completeness coordinate remains available to any Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer. □

Theorem 386.105 (CauchyLimitSeal budget-source factorization)**.** *In the setting of Definition 386.87, every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget-source read factors first through finite approximation window exhaustion, then through budget selector totality, then through tail-schedule cofinality, then through* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *terminality, and only then reaches the terminal route exhaustion row. Hence every completion-facing consumer of the read remains on the displayed finite budget route over* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Proof. Project the accepted finite packet rows from the budget-source carrier. The displayed read supplies the seal rows, observation-budget rows, dyadic-window ledger, finite-approximation packet, tail schedule, RealTailAgreement packet, history, transport, continuation, provenance, and local naming row.

Apply finite approximation window exhaustion from Definition 386.96. This places the finite packet on the same displayed window and dyadic ledger before any completion-facing consumer can inspect the route.

Next use budget selector totality from Definition 386.99. The selector, observation budget, finite approximation, and schedule rows are therefore a single displayed L10 packet rather than independent sources.

Now apply tail-schedule budget cofinality from Definition 386.98. The cofinal tail row is read through the same finite budget route and cannot supply a second endpoint or hidden tail oracle.

Attach terminality by Definition 386.103, then apply terminal route exhaustion from Definition 386.104. Since each stage consumes the same projected packet rows, every completion-facing consumer has no branch outside the displayed finite budget route. □

Corollary 386.106 (CauchyLimitSeal budget window refusal is a strict local obstruction)**.** *In the setting of Definition 386.87, the obstruction detected by Definition 386.89 is strictly local to the attempted reuse of the accepted budget route. More precisely, if a candidate consumer route presents a budget dyadic window not identified, in the transport sense of Definition 386.164, with the scheduled seal-side* $\mathsf{StreamName}^{\uparrow}$ *window, then that candidate is blocked only as a public* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing read of the named accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet and of its displayed root-source surface*

$$X, S, D, G, E, T, P, N \quad \textit{and} \quad E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The refusal cannot be reinterpreted as a global impossibility theorem about Cauchy limits, as a host theorem about real convergence, or as hidden external validation of an ambient completion principle. Every consumer-facing use must factor through the displayed seal rows, budget rows, the scheduled $\mathsf{StreamName}^{\uparrow}$ *window, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger transported through the displayed* hsame *and* Cont *rows in the sense of Definition 386.164 and Definition 301.14; any unlisted coordinate is rejected only for this named route.*

Proof. Start with root source totality from Definition 386.87. It identifies the complete source surface of every accepted $\mathsf{RealObservationBudget}^{\uparrow}$-facing read through the packet: the only seal rows are X, S, D, G, E, T, P, N, and the only budget rows are $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$. In particular, a consumer-facing read has no hidden completion row, no quotient stream row, no selected limit row, no host real-equality row, no hidden family-limit row, and no ambient completeness row from which an additional source could be read.

Next apply dyadic-window exactness from Definition 386.88. The budget ledger D_B is not an independent observation: it is the scheduled seal-side $\mathsf{StreamName}^{\uparrow}$ window ledger already

exposed by D, transported through the displayed hsame and Cont rows in the sense of the common-window transport theorem Definition 386.164 and the RegSeqRat finite-window transport closure Definition 301.14. Thus the only public window coordinate available to the budget route is the listed finite window coordinate.

Now use the refusal theorem Definition 386.89. A candidate route whose dyadic window is not identified with that scheduled seal-side window by the transport route of Definition 386.164 and Definition 301.14 has no coordinate in the displayed seal and budget surface. Therefore it is refused as a public $\mathsf{RealObservationBudget}^{\uparrow}$-facing read of this accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet.

This proves exactly the local obstruction. The argument never introduces a second packet, an ambient host convergence predicate, an external completeness principle, or a global countermodel. It only says that consumer-facing use of the named accepted route must pass through the displayed root-source rows and the displayed dyadic-window identification, so unlisted coordinates cannot be reused as public coordinates of that route. □

Theorem 386.107 (CauchyLimitSeal choice-free selector admission)**.** *Let*

$$Q = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. An accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *row*

$$A = (E_A, M_A, W_A, D_A, R_A, S_A, H_A, C_A, P_A, N_A)$$

or an accepted $\mathsf{LimitSelector}^{\uparrow}$ *row*

$$K = (R_K, n_K, \mu_K, k_K, W_K, D_K, S_K, H_K, C_K, P_K, N_K)$$

is admitted through Q only when its precision request, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row coincide with the public route packet of Q.*

For a $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *row this means that $E_A, M_A, W_A, D_A, R_A, S_A, H_A, C_A, P_A, N_A$ read the same scheduled windows, dyadic ledger, regular handoff, seal row, transports, continuation routes, provenance, and naming row as X, S, D, G, E, T, P, N. For a* $\mathsf{LimitSelector}^{\uparrow}$ *row this means that $n_K, \mu_K, k_K, W_K, D_K, S_K, H_K, C_K, P_K, N_K$ select one of the same finite windows and then read the same dyadic ledger, regular handoff, seal row, transports, continuation routes, provenance, and naming row.*

Proof. Start with the CauchyLimitSeal window factorization of Definition 386.3 and the public route boundary of Definition 386.8. Together they identify every completion-facing or $\mathsf{Real}^{\uparrow}$-facing read of Q with the projected route packet X, S, D, G, E, T, P, N.

For a $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ row, apply the choice-free schedule obligations and non-escape rows of Definition 650.2 and Definition 650.4. They force the request, schedule, selected windows, dyadic ledger, regular handoff, seal row, transports, routes, provenance, and local name to be displayed in the finite packet before any completion consumer can read them.

For a $\mathsf{LimitSelector}^{\uparrow}$ row, apply the selector obligations and finite witness route of Definition 673.2 and Definition 673.3. The modulus-selected index only opens a displayed finite $\mathsf{StreamName}^{\uparrow}$ window, and the following dyadic readback and seal are the rows visible to the selector packet.

Finally use the observation-budget pullback, budgeted tail comparison admission, $\mathsf{RealObservationBudget}^{\uparrow}$ consumer package, and verification handoff of Definition 386.58, Definition 386.59, Definition 386.60, and Definition 386.76. These rows admit a selector surface only when it lands on the same public route packet of Q, so a selector with any different request, window, ledger, handoff, seal, transport, route, provenance, or local name is outside the seal admission surface. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier choice free selector admission`

Theorem 386.108 (CauchyLimitSeal choice-free selector stability)**.** *Suppose accepted* CauchyLimitSeal$^{\uparrow}$ *carriers Q and Q' are related by local classifier transport, and suppose an accepted* ChoiceFreeLimitSchedule$^{\uparrow}$ *row or* LimitSelector$^{\uparrow}$ *row is admitted through Q by Definition 386.107. Then the transported seal admits the transported selector route componentwise: the same choice-free schedule row or the same modulus-selected finite witness row is carried to Q' through the displayed* hsame *transports and* Cont *replays, while the selected finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal,* Pkg *provenance, and local* NameCert *row remain the corresponding rows of the transported public route packet.*

Proof. Project the public route packet of Q. By Definition 386.7, classifier transport acts componentwise on the displayed RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Use selector admission for Q. The admitted ChoiceFreeLimitSchedule$^{\uparrow}$ or LimitSelector$^{\uparrow}$ route has no row outside that public route packet, so each of its request, window, ledger, handoff, seal, transport, replay, provenance, and naming coordinates is already one of the transported coordinates.

For the schedule case, apply Definition 650.5; for the selector case, apply the componentwise classifier clause in Definition 673.2. These rows preserve the same finite schedule or selected finite witness under the displayed hsame comparisons.

The observation-budget pullback and verification handoff of Definition 386.58 and Definition 386.76 then replay the transported route through Q'. Hence classifier transport preserves the admitted selector route component by component and keeps the same finite windows as the selected windows of the transported seal packet. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier choice free selector stability`

Theorem 386.109 (CauchyLimitSeal choice-free selector non-escape)**.** *An accepted selector route admitted by Definition 386.107 exports no hidden limit choice, quotient stream equality, host real equality, ambient completeness datum, unbounded schedule, hidden family limit, or alternate seal. Its public export is exhausted by the displayed* BHist *packet over the* ChoiceFreeLimitSchedule$^{\uparrow}$ *or* LimitSelector$^{\uparrow}$ *row and the* CauchyLimitSeal$^{\uparrow}$ *public route packet: precision request, selected finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal row,* hsame *transports,* Cont *replays,* Pkg *provenance, and local* NameCert *row.*

Proof. Start with selector admission. The admitted route is already identified with the public route packet of the seal, so every exported coordinate is one of the displayed request, window, ledger, handoff, seal, transport, route, provenance, and local naming rows.

Use the selector-side boundaries. The choice-free schedule non-escape theorem Definition 650.4 excludes quotient streams, host real equality, countable choice, hidden selectors, and ambient completeness from the schedule route, while the LimitSelector finite witness route and Real seal non-escape theorems, Definition 673.3 and Definition 673.5, exclude host-selected limits, quotient-stream equality, countable-choice selectors, and ambient completeness from the selector route.

Now apply the seal-side boundaries: public route boundary, observation-budget pullback, budgeted tail comparison admission, RealObservationBudget consumer package, and verification handoff. These rows keep the route on the same finite budget and seal packet, with no field for a hidden family limit, alternate seal, unbounded schedule, host real equality, quotient stream equality, or ambient completeness.

The carrier inventory therefore exhausts the export. Since neither the selector packet nor the admitted CauchyLimitSeal packet contains any of the excluded rows, no downstream $\mathsf{Real}^{\uparrow}$ or completion consumer can recover one from the admitted selector surface. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier choice free selector nonescape`

Corollary 386.110 (CauchyLimitSeal choice-free selector admission strict obstruction)**.** *Let*

$$Q = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. A local refusal of a proposed* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *row*

$$A = (E_A, M_A, W_A, D_A, R_A, S_A, H_A, C_A, P_A, N_A)$$

or of a proposed $\mathsf{LimitSelector}^{\uparrow}$ *row*

$$K = (R_K, n_K, \mu_K, k_K, W_K, D_K, S_K, H_K, C_K, P_K, N_K)$$

through Q is strict in the following sense. It blocks only the attempted reuse of that row as an admitted selector surface for the public route packet of Q. Its scope is limited to the named coordinates: precision request, scheduled or selected finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. It does not supply a global impossibility theorem for choice-free schedules or finite selectors, does not assert host* $\mathsf{Real}^{\uparrow}$ *equality or ambient completeness, and does not validate an external route outside the displayed packet.*

Proof. By Definition 386.107, admission through Q is available only when the proposed schedule or selector row lands on the public route packet X, S, D, G, E, T, P, N. In the schedule case this means that $E_A, M_A, W_A, D_A, R_A, S_A, H_A, C_A, P_A, N_A$ are read as the displayed request, schedule, window, ledger, handoff, seal, transport, replay, provenance, and naming rows of Q. In the selector case this means that $n_K, \mu_K, k_K, W_K, D_K, S_K, H_K, C_K, P_K, N_K$ first select one displayed finite window and then read the same ledger, handoff, seal, transport, replay, provenance, and naming rows of Q.

Thus a refusal can only be a failure of that displayed factorization. If the proposed route changes the request, window, dyadic ledger, regular handoff, seal row, transport, replay, provenance, or local name, the row is outside the admission surface for this carrier. The refusal has no further coordinate to inspect, because the admission theorem has already identified every consumer-facing use with the finite packet just listed.

Now apply Definition 386.109. An admitted selector route exports no hidden limit choice, quotient stream equality, host real equality, ambient completeness datum, unbounded schedule, hidden family limit, or alternate seal. The same inventory controls the rejected route: the rejection is read against the finite packet whose admissible exports are exhausted by the theorem. Therefore the failure cannot be repacked as a host theorem, an ambient completeness principle, an external validation, or a global nonexistence statement about all choice-free schedules or selectors.

The obstruction is consequently local and strict. It blocks exactly the named converse or reuse attempt through Q; every consumer-facing use that remains admissible must factor through the displayed rows of Definition 386.107, and every unlisted coordinate is rejected by the non-escape boundary of Definition 386.109. □

Theorem 386.111 (CauchyLimitSeal root choice-free route)**.** *Every accepted* CauchyLimitSeal$^{\uparrow}$ *packet has a root choice-free route assembled from its carrier, its public route packet, and any admitted* ChoiceFreeLimitSchedule$^{\uparrow}$ *or* LimitSelector$^{\uparrow}$ *selector surface. The route reads only the displayed* RegSeqRat$^{\uparrow}$ *source row,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *ledger,* Real$^{\uparrow}$ *seal row,* BHist *surface,* hsame *transports,* Cont *replays,* Pkg *provenance, and local* NameCert *rows, and it exports no hidden selector, quotient stream equality, selected host limit, ambient completeness datum, or hidden choice principle.*

Proof. Start with the public route boundary of Definition 386.8. It identifies each accepted seal read with the displayed route packet over the regular source, stream windows, dyadic ledger, seal row, transports, continuations, provenance, and local name.

Attach selector admission by Definition 386.107. A ChoiceFreeLimitSchedule$^{\uparrow}$ row or LimitSelector$^{\uparrow}$ row is admitted only when its request, windows, dyadic ledger, regular handoff, seal, transport, route, provenance, and naming coordinates coincide with that public route packet.

Finally use selector non-escape from Definition 386.109. Since the admitted selector surface already contains no hidden limit choice, quotient stream equality, selected host limit, ambient completeness datum, or hidden choice principle, the root route remains the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier root choice free route`

Theorem 386.112 (CauchyLimitSeal root budget seal coverage)**.** *The root route of an accepted* CauchyLimitSeal$^{\uparrow}$ *packet covers every* RealObservationBudget$^{\uparrow}$ *seal read admitted through the packet. The budget read factors through the same* RegSeqRat$^{\uparrow}$ *handoff,* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *observation ledger,* Real$^{\uparrow}$ *seal,* BHist *surface,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows, with no quotient stream equality, selected host limit, ambient completeness, alternate seal, or hidden family limit.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier root budget seal coverage`

Proof. Begin with observation-budget pullback from Definition 386.58. It sends every accepted budget read into the public route packet of the accepted seal while preserving the precision request, finite windows, dyadic ledger, regular handoff, seal row, transports, continuations, provenance, and local naming rows.

Use the budget root source totality theorem Definition 386.87. It says the composed budget read is sourced exactly by the displayed seal rows and displayed budget rows, so the budget cannot introduce a second source surface.

Then apply dyadic-window exactness and completion-consumer non-escape from Definition 386.88 and Definition 386.94. These rows identify the budget ledger with the scheduled seal window and exclude quotient equality, selected host limits, ambient completeness, alternate seals, and hidden family limits. □

Theorem 386.113 (CauchyLimitSeal root formal handoff package)**.** *The root choice-free route of Definition 386.111, the root budget seal coverage of Definition 386.112, the root-unblock package of Definition 386.57, and the completion-budget factorization of Definition 386.56 form one formal handoff package for downstream* Real$^{\uparrow}$ *and completion consumers. The package is the finite* CauchyLimitSeal$^{\uparrow}$ *route over* RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *and it excludes quotient stream equality, selected host limits, ambient completeness, hidden choice, hidden family limits, and host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier root formal handoff package`

Proof. First assemble the choice-free route. By Definition 386.111, selector access is already confined to the accepted seal packet and its displayed route rows.

Next attach the budget coverage. By Definition 386.112, every observation-budget seal read factors through the same regular source, stream window, dyadic ledger, real seal, transport, continuation, provenance, and local naming surface.

Now combine the two sides with Definition 386.57 and Definition 386.56. These rows bind selector, budget, and completion reads to one finite route and keep the route on the same displayed consumer surface.

The excluded coordinates are absent from each ingredient and from the accepted carrier. Therefore the resulting handoff package gives downstream consumers the named finite rows without quotient stream equality, selected host limits, ambient completeness, hidden choice, hidden family limits, or host real equality. □

Theorem 386.114 (CauchyLimitSeal root formal route readback)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion consumer of the formal handoff package of Definition 386.113 reads exactly the displayed regular source, stream windows, dyadic ledger, seal row, transports, continuation routes, provenance, and local name. The readback has no alternate selector, quotient stream equality, selected host limit, ambient completeness, hidden choice, hidden family limit, or host real equality.*

Proof. Begin with the choice-free route of Definition 386.111. It confines every admitted selector surface to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal row, BHist surface, hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row.

Next apply Definition 386.112. Every observation-budget seal read uses the same regular handoff, finite windows, dyadic observation ledger, seal row, transport, continuation, provenance, and local naming coordinates.

The formal handoff package of Definition 386.113 combines those rows with the completion-budget factorization of Definition 386.56. Thus downstream completion consumers receive one finite route rather than an additional selector or limit source.

The excluded coordinates are absent at each cited boundary. Since none of the ingredients exports quotient stream equality, a selected host limit, ambient completeness, hidden choice, a hidden family limit, or host real equality, the readback of the handoff package exposes exactly the displayed rows and nothing further. □

Theorem 386.115 (CauchyLimitSeal root choice-free downstream coverage)**.** *Let*

$$L = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Every accepted local downstream consumer admitted through the root choice-free route of* L *is exhausted by the finite route rows already displayed in the root handoff package: either a* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *or* $\mathsf{LimitSelector}^{\uparrow}$ *selector surface admitted through the same packet, a* $\mathsf{RealObservationBudget}^{\uparrow}$ *seal read covered by the root budget route, or the common readback over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Any proposed downstream coordinate outside these displayed alternatives, ledgers, and coverage rows remains outside the BEDC packet: no hidden selector, quotient stream equality, selected host limit, ambient completeness datum, alternate seal, hidden family limit, host real equality, or unlisted completion source is exported.*

Proof. Start with the root choice-free route of Definition 386.111. It identifies the accepted root route of L with the carrier, its public route packet, and any admitted $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ or $\mathsf{LimitSelector}^{\uparrow}$ selector surface. The componentwise transport used in this identification is the public route transport of Definition 386.7. The only selector alternatives available at that boundary are the displayed precision request, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal row, BHist surface, hsame transports, Cont replays, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Next apply the root budget seal coverage theorem Definition 386.112. Every admitted $\mathsf{RealObservationBudget}^{\uparrow}$ seal read is pulled back to the same regular handoff, finite window, dyadic observation ledger, seal row, transports, continuations, provenance, and local naming coordinates. Thus a budget-facing downstream consumer does not add a second source surface; it is one of the coverage rows already attached to the root route.

The two cited boundaries cover the local consumer cases named in the handoff file. Selector-facing consumers are covered by the choice-free route, budget-facing consumers are covered by the budget seal route, and the common readback is just the shared route over the listed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Therefore an accepted local downstream use factors through the displayed finite alternatives, ledgers, or coverage rows.

The excluded coordinates are absent from both inputs. The choice-free route exports no hidden selector, quotient stream equality, selected host limit, ambient completeness datum, or hidden choice principle, while the budget coverage route exports no quotient stream equality, selected host limit, ambient completeness, alternate seal, or hidden family limit. Since every accepted downstream consumer has already been factored through one of those displayed rows, an unlisted selector, quotient equality, host limit, completeness datum, alternate seal, hidden family limit, host real equality, or external completion source has no BEDC packet coordinate from which it could be read. □

Theorem 386.116 (CauchyLimitSeal StreamName dependency)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer read of a sealed Cauchy limit factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface of chapter 126 before the seal row is consumed. In the carrier notation*

$$L = (X, S, D, G, E, T, P, N),$$

the read must pass through the scheduled window row S, its displayed $\mathsf{RegSeqRat}^{\uparrow}$ *source row in X, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger in D, and the transport route in T before any downstream consumer reads the* $\mathsf{Real}^{\uparrow}$ *seal row E. The route excludes quotient streams, host limits, selected external limits, and ambient completeness data.*

The downstream $\mathsf{RegularCauchyName}^{\uparrow}$ *consumer of chapter 371 reads the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *row, so* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegularCauchyName}^{\uparrow}$ *share the finite window before the* $\mathsf{Real}^{\uparrow}$ *seal row is read.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier streamname dependency`

Proof. Start with the carrier of Definition 386.1. The schedule row S is the only row that lists finite windows, and the seal row E has no independent consumer channel.

Apply window factorization from Definition 386.3. It routes every finite consumer request through the $\mathsf{StreamName}^{\uparrow}$ window, the regular source row, the dyadic observation ledger, and the displayed transport rows before reaching the seal.

Finally use non-escape from Definition 386.4. Since quotient streams, host limits, selected external limits, and ambient completeness do not occur as carrier coordinates, the finite $\mathsf{StreamName}^{\uparrow}$ window surface is the required dependency child for every sealed-limit read. □

Theorem 386.117 (CauchyLimitSeal StreamName and RegularCauchyName lattice)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier* $L = (X, S, D, G, E, T, P, N)$*, every sealed-limit consumer that is also read by* $\mathsf{RegularCauchyName}^{\uparrow}$ *factors through the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *row* S *before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* X*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, diagonal row* G*, and* $\mathsf{Real}^{\uparrow}$ *seal* E*. The packet exports no second window schedule, quotient stream equality, selected host limit, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier streamname regular name lattice`

Proof. Begin with the StreamName dependency theorem Definition 386.116. It already places every sealed-limit read on the scheduled $\mathsf{StreamName}^{\uparrow}$ window before the seal row is consumed.

Apply window factorization from Definition 386.3. The route from S proceeds through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row X, the dyadic ledger D, the diagonal row G, and only then the $\mathsf{Real}^{\uparrow}$ seal row E.

Next use the scheduled-window pullback of Definition 386.20. This identifies the scheduled window as the common dependency surface for the seal route and for the $\mathsf{RegularCauchyName}^{\uparrow}$ consumer of chapter 371.

Since the carrier has no second schedule coordinate and no quotient, selected-limit, or ambient-completeness row, the shared lattice closes at the same finite window and cannot be rerouted through an external equality or host limit. □

Theorem 386.118 (CauchyLimitSeal StreamName Real terminal handoff)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier* $L = (X, S, D, G, E, T, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing sealed-limit consumer factors through the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *row* S*, then the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* X*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the diagonal row* G*, and only then the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The route exposes no second schedule, quotient stream equality, selected host limit, host real equality, or ambient completeness coordinate.*

Proof. Begin with the StreamName dependency theorem Definition 386.116. It places each sealed-limit read on the finite $\mathsf{StreamName}^{\uparrow}$ window before any downstream consumer can read the seal row.

Apply the StreamName and RegularCauchyName lattice theorem Definition 386.117. The shared row is the scheduled S row, and its displayed route proceeds through X, D, and G before reaching E.

Finally combine window factorization Definition 386.3, scheduled-window pullback Definition 386.20, and non-escape Definition 386.4. These local facts rule out a second schedule, quotient equality, selected host limit, host real equality, and ambient completeness as carrier exits, so the only $\mathsf{Real}^{\uparrow}$-facing terminal route is the displayed handoff through S, X, D, G, and E. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier streamname real terminal handoff`

Theorem 386.119 (CauchyLimitSeal StreamName bridge determinacy)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier* $L = (X, S, D, G, E, T, P, N)$*, every consumer-facing route, transport, readback, or terminal handoff that reaches the sealed* $\mathsf{Real}^{\uparrow}$ *endpoint is determined by the displayed local rows*

$$S, \quad X, \quad D, \quad G, \quad T, \quad P, \quad N, \quad E.$$

More precisely, the consumer first factors through the scheduled $\mathsf{StreamName}^{\uparrow}$ *window row* S*, then through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* X*, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the completion diagonal row* G*, and the transport row* T *before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is read; when the same use asks for public readback or certificate provenance, the only available readback coordinates are the displayed* P *and* N *rows on that route. No unlisted bridge coordinate, second schedule, quotient stream equality, selected host limit, host real equality, hidden tail witness, or ambient completeness datum is available to such a consumer.*

Proof. Begin with the StreamName dependency theorem Definition 386.116. It places every sealed-limit consumer read on the finite $\mathsf{StreamName}^{\uparrow}$ window surface before the seal row is consumed, and in the carrier notation $L = (X, S, D, G, E, T, P, N)$ it identifies the mandatory initial route through S, the displayed source row X, the dyadic ledger D, and the transport route T.

Apply the terminal handoff theorem Definition 386.118. This refines the same dependency route for $\mathsf{Real}^{\uparrow}$-facing consumers: after the scheduled row S and the displayed source and ledger rows X and D, the read must pass through the diagonal row G before it reaches the seal row E. Thus any route or handoff reaching the terminal real endpoint has the same ordered local support and cannot choose a different endpoint path.

For transport and readback uses, use the public route packet and public route boundary, Definition 386.6 and Definition 386.8. They restrict the public consumer surface to the displayed carrier rows and their visible transport, package, and naming data. Hence a transport use is read from T on the route already forced through S, X, D, G, E, while any public readback or certificate provenance can only use the displayed P and N rows attached to that same packet.

Finally apply non-escape from Definition 386.4. Since the accepted carrier contains no coordinate for a second schedule, quotient stream equality, selected host limit, host real equality, hidden tail witness, or ambient completeness datum, none of those entries can be supplied after the consumer has reached the public boundary. The local rows listed above therefore exhaust the possible consumer-facing bridge uses, and every unlisted bridge coordinate remains unavailable. □

Corollary 386.120 (CauchyLimitSeal StreamName dependency determinacy)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier, and fix the local rows named by the* $\mathsf{StreamName}^{\uparrow}$ *dependency route: the scheduled window row* S*, the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* X*, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the completion diagonal row* G*, the transport ledger* T*, the provenance row* P*, the local* $\mathsf{NameCert}$ *row* N*, and the sealed* $\mathsf{Real}^{\uparrow}$ *endpoint* E*. Then every accepted consumer-facing use of the* $\mathsf{CauchyLimitSeal}^{\uparrow}$–$\mathsf{StreamName}^{\uparrow}$ *dependency, whether it asks for a route, a transport, a readback, a finite normal form, or an exactness decision, is determined by those rows.*

Equivalently, the only consumer-visible route is the displayed finite route

$$S \to X \to D \to G \to T \to E,$$

with public readback and certificate provenance read only from P *and* N*. No accepted use can choose an alternate hidden coordinate, a second scheduled window, a duplicate classifier source, a quotient stream equality, a selected host limit, a host real equality, a hidden tail witness, or an ambient completeness datum.*

Proof. Start with the $\mathsf{StreamName}^{\uparrow}$ dependency theorem Definition 386.116. It forces every sealed-limit consumer read to enter through the finite $\mathsf{StreamName}^{\uparrow}$ window surface before the seal is consumed. In the displayed carrier notation this fixes the initial dependency rows: the schedule row S, its $\mathsf{RegSeqRat}^{\uparrow}$ source row X, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, and the transport row T.

Now apply the bridge determinacy theorem Definition 386.119. That theorem identifies every consumer-facing route, transport, readback, and terminal handoff reaching the sealed $\mathsf{Real}^{\uparrow}$ endpoint with the local rows

$$S, \quad X, \quad D, \quad G, \quad T, \quad P, \quad N, \quad E.$$

Thus once the dependency rows cited in the statement are fixed, the bridge has no remaining degree of freedom from which a second route or a second public readback could be selected.

The finite normal form is the same route read as a packet normal form. The scheduled window in S names the consumer window; X supplies the regular source row; D supplies the dyadic observation ledger; G supplies the completion diagonal; T supplies the visible BHist, hsame, and Cont transport; E is the sealed endpoint; and P, N supply the Pkg provenance and local NameCert readback. Since all of these rows are already fixed, a normal-form request must return that packet and cannot duplicate the classifier source in a second coordinate.

For exactness decisions, the same argument applies at the public boundary. Exactness is consumer-facing only through the accepted route to the sealed endpoint, and Definition 386.119 confines that route to the displayed local rows. The non-escape clauses cited there rule out quotient stream equality, selected host limits, host real equality, hidden tail witnesses, and ambient completeness data. Therefore any accepted exactness decision is read from the fixed finite route and its P, N certificate rows, with no alternate hidden coordinate or duplicate classifier source available. □

Theorem 386.121 (CauchyLimitSeal StreamName lattice orphan link)**.** *Every accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer route that uses the seal row factors through the sibling* $\mathsf{StreamName}^{\uparrow}$ *finite-window dependency of chapter 126 and through the finite-window lattice dependency of Definition 386.160 before any* $\mathsf{Real}^{\uparrow}$ *seal read. The route remains the displayed finite* X, S, D, G, E, T, P, N *surface and does not introduce a quotient stream, selected host limit, host real equality, hidden tail witness, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier streamname lattice orphan link`

Proof. Begin with the local StreamName dependency Definition 386.116. It places every sealed-limit read on the finite $\mathsf{StreamName}^{\uparrow}$ window surface before the seal row is consumed.

Next use the scheduled-window pullback Definition 386.20. The same finite-window surface is the common dependency for the scheduled window, the regular source row, the dyadic ledger, and the downstream seal route.

Finally apply the finite-window lattice dependency Definition 386.160. It routes the consumer through the StreamName meet surface before the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are read, and its non-escape clause rules out every hidden completion or host-level substitute named in the statement. □

386.1 CauchyLimitSeal Synchronizer Dependency

Theorem 386.122 (CauchyLimitSeal StreamName synchronizer pullback)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal read consumed through chapter 886 first factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface of chapter 126, the displayed* $\mathsf{RegularCauchyName}^{\uparrow}$ *route of chapter 371, and the local* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows before the synchronizer budget, selector, shared-tail, compatibility, transport, route, provenance, and naming rows are read.*

The route has no quotient stream equality, selected host limit, ambient completeness row, or direct $\mathsf{Real}^{\uparrow}$ *seal edge bypassing the finite window.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier streamname synchronizer pullback`

Proof. Begin with the StreamName dependency theorem Definition 386.116. It places every sealed-limit read on the finite $\mathsf{StreamName}^{\uparrow}$ window surface before the $\mathsf{Real}^{\uparrow}$ seal row is consumed.

Next apply the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegularCauchyName}^{\uparrow}$ lattice theorem Definition 386.117. It identifies the displayed regular Cauchy name route as the same finite scheduled window surface used by the seal consumer.

Now read the synchronizer chapter chapter 886. Its consumer rows are the budget, selector, shared-tail, compatibility, transport, route, provenance, and naming rows, and each row is attached to the already displayed seal route rather than to an independent $\mathsf{Real}^{\uparrow}$ edge.

Coordinate exhaustion gives the claim. The only available route into the synchronizer is through the finite window, the displayed regular Cauchy name route, and the local $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows; quotient stream equality, selected host limits, ambient completeness, and a direct $\mathsf{Real}^{\uparrow}$ bypass do not occur among the displayed coordinates. □

Theorem 386.123 (CauchyLimitSeal synchronizer observation-budget route exactness)**.** *Every synchronizer read that enters through an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet and exits through an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the observation-budget rows, and the synchronizer rows before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route is carried by the shared* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it exposes no quotient stream, selected limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness coordinate.*

Proof. Start with the synchronizer pullback of Definition 386.122. It places every synchronizer-facing seal read on the finite $\mathsf{StreamName}^{\uparrow}$ window surface, the displayed $\mathsf{RegularCauchyName}^{\uparrow}$ route, and the local $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows before the synchronizer budget is read.

Next attach the observation-budget consumer exactness theorem Definition 611.9. The accepted $\mathsf{RealObservationBudget}^{\uparrow}$ packet identifies its $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers with the same finite precision, window, dyadic ledger, handoff, seal, transport, route, provenance, and naming rows.

Finally apply Real-facing synchronizer budget exhaustion from Definition 886.6. The shared BHist, hsame, Cont, Pkg, and NameCert rows exhaust the composed route, so there is no remaining coordinate from which a quotient stream, selected limit, host real equality, hidden family limit, unbounded schedule, or ambient completeness principle can be read. □

Theorem 386.124 (CauchyLimitSeal synchronizer tail-budget nonescape)**.** *Every synchronizer read that passes through chapter 886, chapter 611, and an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegularCauchyName}^{\uparrow}$ *route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the local seal rows before the shared-tail budget or* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route exposes no quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness coordinate.*

Proof. First use the pullback theorem Definition 386.122. It places every synchronizer-facing seal read on the finite $\mathsf{StreamName}^{\uparrow}$ window and the displayed $\mathsf{RegularCauchyName}^{\uparrow}$ route before any synchronizer budget row is read.

Next apply the observation-budget route exactness theorem Definition 386.123. This identifies the $\mathsf{RealObservationBudget}^{\uparrow}$ leg with the same finite $\mathsf{RegSeqRat}^{\uparrow}$ handoff, observation-budget rows, synchronizer rows, and local $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal rows.

Finally apply the synchronizer budget exhaustion theorem Definition 886.6. The shared-tail and $\mathsf{Real}^{\uparrow}$ seal reads have already spent the displayed BHist, hsame, Cont, Pkg, and NameCert

coordinates, so carrier-coordinate exhaustion leaves no quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness coordinate to consume. □

Theorem 386.125 (CauchyLimitSeal synchronizer finite-window nonescape)**.** *Every synchronizer-facing* CauchyLimitSeal$^{\uparrow}$ *read that passes through* StreamName$^{\uparrow}$, RegularCauchyName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, *and the local seal rows factors through the displayed finite* StreamName$^{\uparrow}$ *window,* ProbeBundle *and* InBundle *membership, the* RegSeqRat$^{\uparrow}$ *handoff, the observation-budget rows, the synchronizer rows, and the local* BHist, hsame, Cont, Pkg, *and* NameCert *coordinates before any* Real$^{\uparrow}$ *seal is consumed. No quotient stream, hidden family limit, selected host limit, host real equality, unbounded schedule, ambient completeness principle, or direct* Real$^{\uparrow}$ *bypass is readable from such a route.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier synchronizer finite window nonescape`

Proof. First apply the StreamName and regular-name pullback theorem Definition 386.122. It places the synchronizer-facing seal read on the finite StreamName$^{\uparrow}$ window surface and the displayed RegularCauchyName$^{\uparrow}$ route before any synchronizer budget coordinate is used.

Next apply the observation-budget route exactness theorem Definition 386.123. The RealObservationBudget$^{\uparrow}$ leg is therefore carried by the same finite window, the RegSeqRat$^{\uparrow}$ handoff, the observation-budget rows, the synchronizer rows, and the local CauchyLimitSeal$^{\uparrow}$ seal rows.

Finally use the tail-budget nonescape theorem Definition 386.124. It exhausts the shared-tail route through the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates before the Real$^{\uparrow}$ seal is read.

These coordinates are the entire displayed synchronizer route. Since quotient streams, hidden family limits, selected host limits, host real equality, unbounded schedules, ambient completeness principles, and direct Real$^{\uparrow}$ bypasses are absent from that coordinate list, none of them can be read by the synchronizer consumer. □

Theorem 386.126 (CauchyLimitSeal synchronizer route factorization)**.** *Any synchronizer-facing* CauchyLimitSeal$^{\uparrow}$ *read factors through the* StreamName$^{\uparrow}$ *finite-window surface, the* RegularCauchyName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *handoff surfaces, the* RealObservationBudget$^{\uparrow}$ *rows, the* CauchySealBudgetSynchronizer$^{\uparrow}$ *rows of chapter 886, and the local* CauchyLimitSeal$^{\uparrow}$ *rows before the* Real$^{\uparrow}$ *seal is consumed. The factorization has no quotient stream equality, selected host limit, hidden family limit, host real equality, unbounded schedule, ambient completeness row, or direct* Real$^{\uparrow}$ *bypass.*

Proof. Start with the pullback theorem Definition 386.122. It places the synchronizer-facing read on the finite StreamName window, the displayed RegularCauchyName route, and the local CauchyLimitSeal rows before the synchronizer budget rows of chapter 886 are read.

Next use observation-budget route exactness from Definition 386.123. This identifies the RealObservationBudget leg with the RegSeqRat handoff, the observation-budget rows, the synchronizer rows, and the same local seal rows.

The tail-budget nonescape theorem Definition 386.124 then rules out quotient streams, selected host limits, hidden family limits, unbounded schedules, and ambient completeness coordinates along the shared-tail route.

Finally apply finite-window nonescape Definition 386.125. It exhausts the remaining BHist, hsame, Cont, Pkg, and NameCert coordinates before any Real$^{\uparrow}$ seal read, so the displayed factorization is the only synchronizer-facing route. □

Theorem 386.127 (CauchyLimitSeal RealCompletionSealClassifier handoff)**.** *Every synchronizer-facing* CauchyLimitSeal$^{\uparrow}$ *read consumed by an accepted* RealCompletionSealClassifierUp *packet factors through the finite* StreamName$^{\uparrow}$ *window surface, the* RegularCauchyName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *handoff surfaces, the* RealObservationBudget$^{\uparrow}$ *rows, the* CauchySealBudgetSynchronizer$^{\uparrow}$ *rows, the local* CauchyLimitSeal$^{\uparrow}$ *rows, and then the* RealCompletionSealClassifierUp *public export route of Definition 900.13. The handoff exports no quotient stream, selected host limit, host real equality, countable-choice row, or ambient completeness coordinate.*

Proof. Apply synchronizer route factorization, Definition 386.126. It places the consumed CauchyLimitSeal$^{\uparrow}$ read on the finite StreamName$^{\uparrow}$ window, RegularCauchyName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ handoff rows, RealObservationBudget$^{\uparrow}$ rows, CauchySealBudgetSynchronizer$^{\uparrow}$ rows, and local seal rows before the consumer reaches any Real$^{\uparrow}$ seal.

Now apply the public export theorem for the classifier, Definition 900.13. The accepted RealCompletionSealClassifierUp packet can consume the synchronizer-facing seal only through its displayed public rows, so the preceding synchronizer route is handed to the classifier as part of the same finite completion-seal export surface.

Finally combine the non-escape boundary already built into Definition 386.126 with the classifier non-escape boundary cited by Definition 900.13. Neither route contains a quotient stream, selected host limit, host real equality, countable-choice row, or ambient completeness coordinate, and the classifier public export adds no second channel for such data. □

Theorem 386.128 (CauchyLimitSeal common-window budget synchronizer exhaustion)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* CauchyLimitSeal$^{\uparrow}$ *carrier. Any synchronizer-facing seal read of* L *that consumes both the* RealObservationBudget$^{\uparrow}$ *route and the* CauchySealBudgetSynchronizer$^{\uparrow}$ *route first pulls back to the same* StreamName$^{\uparrow}$ *finite-window lattice surface of Definition 386.160. It then proceeds through the displayed* RegSeqRat$^{\uparrow}$ *source row, dyadic ledger, completion diagonal, synchronizer rows, observation-budget rows, and* Real$^{\uparrow}$ *seal row. The common route exports no second finite window, quotient stream equality, selected host limit, hidden family limit, host real equality, unbounded schedule, or ambient completeness coordinate.*

Proof. Begin with the finite-window lattice dependency of Definition 386.160. It pulls the accepted seal route back to the StreamName$^{\uparrow}$ finite-window lattice surface before the displayed RegSeqRat$^{\uparrow}$ source row, dyadic ledger, completion diagonal, and Real$^{\uparrow}$ seal row are consumed.

Next apply synchronizer route factorization, Definition 386.126. This places the synchronizer-facing leg on the same displayed StreamName$^{\uparrow}$, RegularCauchyName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, CauchySealBudgetSynchronizer$^{\uparrow}$, and local CauchyLimitSeal$^{\uparrow}$ rows rather than on an independent seal route.

Now apply observation-budget route exactness, Definition 386.123. The RealObservationBudget$^{\uparrow}$ leg is therefore consumed along the same finite precision, window, dyadic ledger, handoff, seal, transport, route, provenance, and naming rows used by the synchronizer leg.

Finally use finite-window nonescape Definition 386.125 and tail-budget nonescape Definition 386.124. Together they exhaust the remaining BHist, hsame, Cont, Pkg, and NameCert coordinates before the Real$^{\uparrow}$ seal is read, leaving no second finite window, quotient stream equality, selected host limit, hidden family limit, host real equality, unbounded schedule, or ambient completeness coordinate. □

Theorem 386.129 (CauchyLimitSeal observer-budget handoff)**.** *Every synchronizer-facing* CauchyLimitSeal$^{\uparrow}$ *read consumed by a Real completion consumer first factors through the finite* StreamName$^{\uparrow}$ *window, the* RegularCauchyName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *handoff, the* RealObservationBudget$^{\uparrow}$ *rows, the*

$\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *route, and the observer-budget limiting surface before any* $\mathsf{Real}^{\uparrow}$ *seal is public. The public handoff excludes quotient streams, selected host limits, hidden family limits, host real equality, unbounded schedules, countable choice, and ambient completeness.*

Proof. Begin with common-window budget synchronizer exhaustion, Definition 386.128. It places the synchronizer-facing seal read on the same finite $\mathsf{StreamName}^{\uparrow}$ lattice surface, $\mathsf{RegSeqRat}^{\uparrow}$ source row, dyadic ledger, completion diagonal, synchronizer rows, observation-budget rows, and $\mathsf{Real}^{\uparrow}$ seal row.

Next apply the RealCompletionSealClassifier handoff theorem, Definition 386.127. The Real completion consumer can receive the synchronizer-facing $\mathsf{CauchyLimitSeal}^{\uparrow}$ read only through the displayed classifier export route, so the finite handoff remains before the public $\mathsf{Real}^{\uparrow}$ seal.

Now use observation-budget route exactness, Definition 386.123. It identifies the $\mathsf{RealObservationBudget}^{\uparrow}$ leg with the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the observation-budget rows, the synchronizer rows, and the local $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet.

Carrier-coordinate exhaustion finishes the handoff. The displayed rows over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ are the whole route available to the consumer before the $\mathsf{Real}^{\uparrow}$ seal is public. Since quotient streams, selected host limits, hidden family limits, host real equality, unbounded schedules, countable choice, and ambient completeness are not coordinates of that route, none can be exported by the observer-budget limiting surface. □

Theorem 386.130 (CauchyLimitSeal synchronizer observer-real seal factorization)**.** *Every synchronizer-facing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read consumed by a Real completion consumer factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface, the* $\mathsf{RegularCauchyName}^{\uparrow}$ *route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *rows, the local* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal rows, and the public* $\mathsf{Real}^{\uparrow}$ *seal route before any completion-facing output is public.*

The factorization is a finite BEDC route over the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *coordinates. It exports no quotient stream equality, selected host limit, host real equality, unbounded schedule, countable-choice row, or ambient completeness coordinate.*

Proof. Start with synchronizer route factorization, Definition 386.126. It places every synchronizer-facing $\mathsf{CauchyLimitSeal}^{\uparrow}$ read on the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegularCauchyName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, the observation-budget rows, the synchronizer rows, and the local seal rows before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Next apply common-window budget synchronizer exhaustion. Definition 386.128 identifies the synchronizer and observation-budget legs with the same finite window, source row, dyadic ledger, completion diagonal, and seal route, so the Real-facing consumer cannot split them into two independent routes.

Now use the RealCompletionSealClassifier handoff theorem. It sends the already factorized synchronizer-facing read through the classifier public export surface, which keeps the completion-facing output behind the same finite seal route.

Finally apply observer-budget handoff. The observer-budget theorem states that the public Real completion read appears only after the $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, local $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows have all been traversed. Since quotient stream equality, selected host limits, host real equality, unbounded schedules, countable choice, and ambient completeness are absent from that traversed route, none can be supplied to the completion-facing output. □

Theorem 386.131 (CauchyLimitSeal witness-ledger synchronizer handoff)**.** *Every synchronizer-facing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read consumed by a* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *route, and the local* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal rows before the witness ledger can expose any completion-facing output. Once the ledger receives the synchronized seal read, the completion-facing route is exactly the displayed witness-ledger path*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with structural rows H, C, P, N *from Definition 902.5.*

The handoff exports no quotient stream equality, selected host limit, host real equality, countable-choice row, or ambient completeness coordinate.

Proof. First apply synchronizer route factorization, Definition 386.126. It places the consumed $\mathsf{CauchyLimitSeal}^{\uparrow}$ read on the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegularCauchyName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, the observation-budget rows, the $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ route, and the local seal rows before any Real-facing seal output is public.

Next apply observer-real seal factorization, Definition 386.130. The Real completion consumer therefore receives the synchronized seal only after the finite route through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ has already been traversed.

Finally apply the witness-ledger budget factorization theorem, Definition 902.5. The $\mathsf{CauchyWitnessLedger}^{\uparrow}$ consumer can expose a completion-facing row only by reading Q, then B, then S, then K, with H, C, P, N replaying and naming the same finite packet.

The excluded coordinates are absent from both sides of the composition. The synchronizer side has no quotient stream equality, selected host limit, host real equality, countable-choice row, or ambient completeness coordinate, and the witness-ledger route adds only the displayed Q, B, S, K, H, C, P, N rows. Thus the composed handoff has no external route through which any excluded coordinate can become public. □

Theorem 386.132 (CauchyLimitSeal synchronizer witness-ledger finality)**.** *Every synchronizer-facing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read that has passed through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegularCauchyName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$*,* $\mathsf{RealCompletionSealClassifierUp}$*, and* $\mathsf{CauchyWitnessLedge}$ *is final at the displayed finite witness-ledger route. Any public completion-facing output factors through the same* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, and* Q, B, S, K, H, C, P, N *coordinates, and cannot read quotient stream equality, selected host limit, host real equality, countable choice, ambient completeness, or a second seal edge.*

Proof. First compose synchronizer route factorization with observer-real seal factorization, using Definition 386.126 and Definition 386.130. The synchronizer-facing read is already on the finite stream-name, regular-Cauchy-name, rational-sequence, observation-budget, synchronizer, and local seal rows before any completion-facing output is public.

Next apply the witness-ledger synchronizer handoff of Definition 386.131. It sends the synchronized seal read into the displayed witness-ledger path and prevents the completion consumer from bypassing the ledger.

Finally use the budget factorization of Definition 902.5. The public output can read only Q, B, S, K, H, C, P, N together with the underlying BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ structure already traversed by the seal route. Since quotient stream equality, selected host limits, host real equality, countable choice, ambient completeness, and a second seal edge are absent from those coordinates, the witness-ledger route is final for the synchronized read. □

Theorem 386.133 (CauchyLimitSeal synchronizer witness-ledger scope)**.** *Every completion-facing output produced after a synchronizer-facing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read and a* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *handoff is scoped exactly by the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegularCauchyName}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *route, local* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows,* $\mathsf{RealCompletionSealClassifierUp}$ *row, and witness-ledger rows* Q, B, S, K, H, C, P, N*. The output has no coordinate for quotient-stream equality, selected host limits, host real equality, countable choice, ambient completeness, or a second seal edge.*

Proof. First compose synchronizer route factorization with observer-real seal factorization, using Definition 386.126 and Definition 386.130. The synchronizer-facing read is thereby placed on the finite stream-name window, regular-Cauchy-name row, rational-sequence source row, observation-budget rows, synchronizer route, local seal rows, and Real classifier route before any completion-facing output is public.

Next attach the witness-ledger synchronizer handoff from Definition 386.131. The handoff sends the synchronized read through the displayed ledger path Q, B, S, K with structural rows H, C, P, N, so the completion-facing consumer reads the ledger and the classifier as one finite route.

Finally apply witness-ledger finality, Definition 386.132. Its final route is the same synchronizer, classifier, and witness-ledger surface just assembled, so no completion-facing output can be scoped by a larger packet.

Carrier-coordinate exhaustion gives the exclusions. The displayed route contains the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, local $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCompletionSealClassifierUp}$, and witness-ledger rows, and it contains no quotient stream, selected host limit, host real equality, countable-choice, ambient-completeness, or second-seal coordinate. □

Corollary 386.134 (CauchyLimitSeal synchronizer strict local obstruction)**.** *Any displayed failure, refusal, separation, or countermodel row attached to the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *StreamName synchronizer pullback is strict only for the named synchronizer converse or reuse route that tries to consume the same finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegularCauchyName}^{\uparrow}$ *route,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows, synchronizer rows, and local seal rows. Such a row cannot be read as a global impossibility theorem, a host theorem about limits, a hidden external validation, a quotient-stream equality failure, an ambient-completeness failure, or a second independent* $\mathsf{Real}^{\uparrow}$ *seal obstruction.*

Proof. Apply tail-budget nonescape, Definition 386.124. It says that every synchronizer read passing through the synchronizer chapter, the observation-budget chapter, and an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet must first display the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegularCauchyName}^{\uparrow}$ route, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the local seal rows before the shared-tail budget or $\mathsf{Real}^{\uparrow}$ seal is consumed. It also states that this route exposes no quotient stream, selected host limit, hidden family limit, unbounded schedule, or ambient completeness coordinate.

Now apply observation-budget route exactness, Definition 386.123. The accepted $\mathsf{RealObservationBudget}^{\uparrow}$ leg and the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ leg are carried by the same finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, observation-budget rows, synchronizer rows, and shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates. Thus any consumer-facing use of a displayed negative row must factor through exactly those rows.

The strictness follows by coordinate exhaustion. A failure, refusal, separation, or countermodel row on this pullback can block the attempted converse or reuse of the displayed synchronizer route,

because that is the route whose rows it names. But the two cited theorems leave no additional coordinate for a global impossibility theorem, host limit theorem, hidden external validation, quotient-stream equality, ambient-completeness principle, or independent $\mathsf{Real}^{\uparrow}$ seal edge. Reinterpreting the row in any of those ways would require a coordinate not present in the finite synchronizer packet, so the obstruction is local to the named pullback route. □

Theorem 386.135 (CauchyLimitSeal tail-budget meet)**.** *Let*

$$L = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, and let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet over the same displayed precision request and finite* $\mathsf{StreamName}^{\uparrow}$ *window. If the seal row* S_B *and the regular handoff rows* R_B *are routed through* L*, then there is a single finite meet row over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *that binds the selector read, the observation-budget read, and the completion-consumer read to one BEDC route. The meet row contains no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail budget meet`

Proof. Start with tail extraction from Definition 386.70. It projects every admitted tail read of L from the displayed regular family, stream schedule, dyadic observation ledger, diagonal row, seal row, and visible transport route.

Attach the $\mathsf{RealObservationBudget}^{\uparrow}$ side using Definition 386.60. Since the budget packet and the seal packet share the displayed precision request and window, the budget rows and seal rows meet over the same BHist, hsame, Cont, Pkg, and NameCert surface.

Finally use budget root source totality from Definition 386.87. It says every accepted budget-facing read is sourced by the listed seal and budget rows, so the selector, budget, and completion reads have one finite route and no extra completion, quotient, equality, selection, hidden-family, or ambient-completeness coordinate. □

Theorem 386.136 (CauchyLimitSeal selector-budget transport)**.** *In the setting of Definition 386.135, componentwise* hsame *transport and the displayed* Cont *routes preserve the meet row through selector admission and observation-budget transport. Thus any transported selector read, budget read, or completion-facing read remains on the same precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *surface,* Pkg *provenance, and local* NameCert *row. Transport does not introduce quotient stream identity, host real equality, a choice-selected limit, a hidden family limit, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier selector budget transport`

Proof. Project the meet row supplied by Definition 386.135. Its readable coordinates are already the selector, budget, and completion-consumer coordinates on the shared finite route.

For the selector side, use Definition 386.108. It transports the admitted selector route componentwise through the displayed hsame comparisons and Cont replays while preserving the selected finite windows and dyadic ledger.

For the budget side, use the observation-budget pullback and consumer package from Definition 386.58 and Definition 386.60. These rows preserve the precision request, window, ledger,

regular handoff, seal, route, provenance, and naming rows. Hence every transported read remains inside the same finite meet packet. □

Theorem 386.137 (CauchyLimitSeal tail-budget meet exactness)**.** *Let L be an accepted* CauchyLimitSeal$^{\uparrow}$ *packet and let B be an accepted* RealObservationBudget$^{\uparrow}$ *packet over the same displayed precision request and finite* StreamName$^{\uparrow}$ *window. The two packets classify through the same finite tail-budget meet row if and only if their selector row, observation-budget row,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows agree componentwise inside the meet surface. This exactness statement excludes quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail budget meet exactness`

Proof. For the forward direction, project the finite meet row supplied by Definition 386.135. The meet has no coordinates other than the selector, budget, dyadic ledger, regular handoff, real seal, transport, continuation, provenance, and naming rows, so classification through the same row identifies those coordinates componentwise.

Apply Definition 386.136. It keeps the selector side and observation-budget side on the same precision request, stream window, dyadic ledger, regular handoff, real seal, transport row, continuation route, package row, and local name under every displayed hsame and Cont replay.

For the reverse direction, start from componentwise agreement of the listed coordinates. Repack the selector read, observation-budget read, dyadic ledger, regular handoff, real seal, transports, routes, provenance, and naming row into the meet supplied by Definition 386.135.

The repacked meet is accepted because each coordinate is already an accepted coordinate of L and B over the same precision request and finite window. Since no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate is present, the shared componentwise row is exactly the common classifier surface. □

Theorem 386.138 (CauchyLimitSeal regular-tail budget exactness)**.** *Suppose an accepted* CauchyLimitSeal$^{\uparrow}$ *packet and an accepted* RealObservationBudget$^{\uparrow}$ *packet share the same finite* StreamName$^{\uparrow}$ *window and the same* RegSeqRat$^{\uparrow}$ *handoff. Inside the tail-budget meet, the regular-tail budget is exact if and only if the selector row, observation-budget row, dyadic ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row agree.*

The exact regular-tail budget has no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.

Proof. For the forward direction, project the shared meet by Definition 386.135. Since the two packets share the StreamName window and RegSeqRat handoff, the remaining readable budget coordinates are precisely the selector, observation-budget, dyadic-ledger, real-seal, transport, continuation, provenance, and naming rows.

Apply Definition 386.136. It keeps the selector read and the observation-budget read on the same regular-tail route under the displayed hsame transports and Cont replays.

For the reverse direction, assume the listed rows agree componentwise. Repack those rows into the meet supplied by Definition 386.135; the already shared StreamName window and RegSeqRat handoff give the regular-tail spine, and the agreed rows fill the budget surface.

Finally use Definition 386.137. It identifies componentwise agreement inside the meet with classification through the same finite tail-budget row and excludes the listed quotient, host-equality, selected-limit, hidden-family, and ambient-completeness coordinates. □

Theorem 386.139 (CauchyLimitSeal completion-budget root unblock)**.** *The tail-budget meet of Definition 386.135 and its selector-budget transport of Definition 386.136 package the root-unblock and completion-consumer rows into one downstream surface. A* $\mathsf{Real}^{\uparrow}$ *or completion consumer may cite this surface as the finite route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *supplied by the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{RealObservationBudget}^{\uparrow}$ *packets. The package explicitly excludes quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier completion budget root unblock`

Proof. First take the root-unblock package from Definition 386.57. It supplies the downstream route from the accepted seal packet to the Real and completion consumer boundary.

Next refine that route by the tail-budget meet. The meet binds the selector read and the observation-budget read to the same displayed precision and window route before the completion consumer reads the packet.

Finally apply Definition 386.94. The completion-facing endpoint can export only the rows listed by the root source and dyadic-window exactness theorems, so the combined package has no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate. □

Theorem 386.140 (CauchyLimitSeal tail-budget consumer normal form)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer of the tail-budget meet factors through exactly the shared precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row named by Definition 386.135 and Definition 386.136. This consumer route has no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail budget consumer normal form`

Proof. First cite the meet theorem Definition 386.135. It binds the selector read, observation-budget read, and completion-consumer read to one finite route over the shared precision request, finite stream window, dyadic ledger, regular handoff, real seal, transport, continuation, provenance, and naming rows.

Next apply the selector-budget transport theorem Definition 386.136. It keeps those rows stable under componentwise hsame transport and displayed Cont replay, so a transported selector, budget, or completion-facing read cannot leave the same finite packet.

Finally close with Definition 386.139. The root-unblock package is therefore consumed only through the displayed meet and transport route, excluding quotient stream identity, host real equality, choice-selected limit, hidden family limit, and ambient completeness. □

Theorem 386.141 (CauchyLimitSeal tail-meet uniform selector route)**.** *Let*

$$L = (X, S, D, G, E, T, P, N)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet, let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted RealObservationBudget$^{\uparrow}$ *packet over the same displayed precision request and finite* StreamName$^{\uparrow}$ *window, and let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet whose displayed finite window, dyadic tolerance,* RegSeqRat$^{\uparrow}$ *tail row, and* Real$^{\uparrow}$ *seal handoff are exactly the rows read by the tail-budget meet. Then every* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer using the selector-budget route factors through one finite envelope: the* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *source row,* DyadicRatCore$^{\uparrow}$ *ledger,* UniformCauchyCriterion$^{\uparrow}$ *tail ledger,* RealObservationBudget$^{\uparrow}$ *rows,* CauchyLimitSeal$^{\uparrow}$ *seal row, and the shared* BHist, hsame, Cont, Pkg, *and* NameCert *rows. This route has no quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail meet uniform selector route`

Proof. Begin with the tail-budget meet of Definition 386.135. It binds the selector read, the observation-budget read, and the completion-consumer read to the displayed precision request, finite stream window, dyadic ledger, regular handoff, real seal, transport, continuation, provenance, and naming rows.

Transport that meet by Definition 386.136. The selector-budget route preserves the same StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert surface under every displayed replay.

Apply the consumer normal form of Definition 386.140. A Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer therefore receives the meet only through the stated finite envelope and cannot switch to a quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, or ambient completeness witness.

Finally attach the UniformCauchyCriterion side by Definition 560.18 and Definition 435.81. These frontiers force the displayed uniform tail ledger to read the same finite family, dyadic tolerance, regular-rational tail, and real-seal handoff before any uniform-family consumer inspects it, so U enters the selector-budget route through the same finite envelope. □

Theorem 386.142 (CauchyLimitSeal tail-meet refusal fork)**.** *Every* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer of the tail-budget meet has exactly one of the following two routes. Either it factors through the finite envelope consisting of the* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *source row,* DyadicRatCore$^{\uparrow}$ *ledger,* UniformCauchyCriterion$^{\uparrow}$ *tail ledger,* RealObservationBudget$^{\uparrow}$ *rows,* CauchyLimitSeal$^{\uparrow}$ *seal row, and shared* BHist, hsame, Cont, Pkg, *and* NameCert *rows; or its request is routed to the visible refusal boundary for quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, ambient completeness, or metric-completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail meet refusal fork`

Proof. First use the tail-budget meet of Definition 386.135. It supplies the common route over the selector read, observation-budget read, completion-consumer read, and shared structural rows.

Transport that common route with Definition 386.136. The transported consumer still sees only the finite precision request, window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows.

Apply the consumer normal form of Definition 386.140 together with the root unblock of Definition 386.139. These results force every Real$^{\uparrow}$ or Completion$^{\uparrow}$ read to enter the downstream surface through the displayed finite meet before any completion-facing inspection occurs.

Finally attach the uniform selector route from Definition 386.141. Inspecting its envelope shows that quotient stream, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, ambient completeness, and metric-completeness are not coordinates of the finite route. A request for any such coordinate therefore cannot be served by the envelope and is sent to the visible refusal boundary. □

Theorem 386.143 (CauchyLimitSeal tail-meet source exhaustion)**.** *Every* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer admitted by the tail-meet route exhausts exactly the finite source rows supplied by the* CauchyLimitSeal$^{\uparrow}$ *packet, the* RealObservationBudget$^{\uparrow}$ *packet, and the* UniformCauchyCriterion$^{\uparrow}$ *packet. The exhausted rows are the shared* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *source row,* DyadicRatCore$^{\uparrow}$ *ledger,* BHist *histories,* hsame *transports,* Cont *continuations,* Pkg *provenance rows, and* NameCert *naming rows, together with the displayed seal, observation-budget, and uniform-tail coordinates named by those packets.*

No admitted consumer may read an additional Real-completion source outside these finite rows. A request for quotient stream identity, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, ambient completeness, or metric-completeness is routed to the refusal boundary rather than to a second source envelope.

Proof. Begin with the uniform selector route of Definition 386.141. It joins the CauchyLimitSeal$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, and UniformCauchyCriterion$^{\uparrow}$ packets over one finite envelope with the shared StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply the refusal fork Definition 386.142. A Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer is therefore either on that finite envelope or outside the admitted route, where the listed non-envelope requests are refused.

The exact source rows are fixed by Definition 386.135. The meet binds the selector read, observation-budget read, and completion-consumer read to the same precision request, finite stream window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows.

Finally use the consumer normal form Definition 386.140. Since every admitted consumer has that normal form and the fork refuses the alternatives, the route exhausts exactly the finite rows supplied by the three source packets and exposes no additional Real-completion source. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail meet source exhaustion`

Theorem 386.144 (CauchyLimitSeal tail-meet diagonal compatibility handoff)**.** *Suppose an accepted* CauchyLimitSeal$^{\uparrow}$ *tail-meet route has finite envelope E, and suppose the same envelope is consumed by an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet through its root formal handoff surface. Then the handoff boundary is exactly the shared* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *source row,* DyadicRatCore$^{\uparrow}$ *ledger,* Real$^{\uparrow}$ *seal row,* UniformCauchyCriterion$^{\uparrow}$ *tail ledger,* BHist, hsame, Cont, Pkg, *and* NameCert *rows, together with the diagonal compatibility selector read from the accepted compatibility packet.*

The handoff exposes no quotient stream, host real equality, selected external limit, hidden diagonal oracle, independent completion row, or ambient completeness coordinate.

Proof. Begin with the tail-budget meet of Definition 386.135. It identifies the common precision request, stream window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows that any admitted Real-completion consumer must read.

Transport the route through Definition 386.136. The selector read stays inside the same finite envelope, so a downstream diagonal consumer sees the transported tail-meet rows rather than a second completion source.

Apply source exhaustion from Definition 386.143. The admitted route has already exhausted the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows before the diagonal compatibility packet reads the envelope.

Finally read the root formal handoff surface of $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$. That surface contributes only the displayed diagonal compatibility selector and consumes the already exhausted envelope. Since quotient stream identity, host real equality, selected external limits, hidden diagonal oracles, independent completion rows, and ambient completeness coordinates are absent from both finite surfaces, none can cross the handoff boundary. □

Theorem 386.145 (CauchyLimitSeal tail-meet diagonal source exhaustion)**.** *Every* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route exhausts exactly the shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail ledger,* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows, diagonal selector row, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

No such consumer retains a coordinate for quotient stream identity, host real equality, a selected external limit, a hidden diagonal oracle, an independent completion row, ambient completeness, or metric completeness outside the accepted handoff surface.

Proof. First use source exhaustion for the tail meet, Definition 386.143. It fixes the accepted Real and completion route as the finite envelope over the shared stream window, regular-rational source row, dyadic ledger, Real seal, uniform tail ledger, observation-budget rows, and structural BEDC rows.

Next attach the uniform selector route of Definition 386.141. This places the diagonal-facing selector read on the same finite route before any compatibility consumer receives the envelope.

Apply the diagonal compatibility handoff of Definition 386.144 together with the root formal route of Definition 646.183. The diagonal packet contributes only its displayed selector row and consumes the already bounded source route.

The refusal clauses in the handoff theorem then identify the absent coordinates. Quotient stream identity, host real equality, selected external limits, hidden diagonal oracles, independent completion rows, ambient completeness, and metric completeness are not fields of either finite surface, so the diagonal consumer exhausts exactly the listed shared rows. □

Theorem 386.146 (CauchyLimitSeal tail-budget meet consumer determinacy)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer admitted by the tail-budget meet is determined by the shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealObservationBudget}^{\uparrow}$ *budget row,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal row,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail row, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. Two admitted consumers that agree on these rows have the same public Real-completion read, and no quotient stream identity, host real equality, selected external limit, hidden diagonal oracle, independent completion row, or ambient completeness coordinate can distinguish them.*

Proof. Begin with the finite meet of Definition 386.135. It binds the selector read, observation-budget read, and completion-consumer read to one displayed envelope over the shared stream window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows.

Apply source exhaustion from Definition 386.143. The admitted Real or completion consumer has no second source outside the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally use diagonal source exhaustion from Definition 386.145. It identifies the same envelope after the diagonal-facing handoff, so agreement on the finite rows determines the public consumer read and leaves no absent coordinate able to separate the two consumers. □

Theorem 386.147 (CauchyLimitSeal tail-budget meet ledger exhaustion)**.** *The ledger of an admitted* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer of the tail-budget meet is exhausted by the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows displayed by the meet and its source-exhaustion theorems. After those rows are read, there is no remaining ledger coordinate for quotient stream identity, host real equality, selected external limits, hidden diagonal oracles, independent completion rows, ambient completeness, or metric completeness.*

Proof. Use Definition 386.135 to obtain the finite route binding the selector, budget, and completion reads. The ledger coordinates of that route are exactly the displayed stream, regular-rational, dyadic, Real, transport, continuation, provenance, and naming rows.

Refine the inventory with Definition 386.143. It adds the observation-budget and uniform-tail rows to the same finite envelope and refuses any second Real-completion source.

Then apply Definition 386.145. The diagonal-facing consumer receives only the already exhausted rows plus the displayed selector row, so the listed non-envelope objects have no ledger position to occupy. □

Theorem 386.148 (CauchyLimitSeal tail-budget meet Real handoff)**.** *The Real handoff exported by the tail-budget meet is exactly the finite route through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *It cannot import quotient stream identity, host real equality, selected external limits, hidden diagonal oracles, independent completion rows, ambient completeness, or metric completeness into the* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer boundary.*

Proof. Start from the consumer determinacy theorem above. A Real or completion consumer is fixed by the shared finite envelope, so a handoff can expose only rows already present in that envelope.

Apply ledger exhaustion above. Once the stream, regular-rational, dyadic, budget, seal, uniform-tail, transport, continuation, provenance, and naming rows are consumed, there is no remaining field through which a quotient stream, host equality, selected limit, hidden diagonal oracle, independent completion row, ambient completeness, or metric completeness witness could enter.

The handoff is therefore the same finite Real-completion boundary described by the meet, source exhaustion, and diagonal source exhaustion rows, with no ambient completion principle imported from outside the accepted packets. □

Corollary 386.149 (CauchyLimitSeal tail-budget meet strict local obstruction)**.** *Every displayed failure, refusal, separation, or countermodel row attached to the tail-budget meet is a strict local obstruction. If a* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer of the meet asks for quotient stream identity, selected host limit, host real equality, hidden family limit, unbounded schedule, independent completion row, ambient completeness, or metric-completeness, then the row blocks exactly that requested converse or reuse route outside the displayed finite envelope. The same row does not assert a global impossibility theorem, does not supply a host-level theorem about all Cauchy limits, and does not act as hidden external validation for any completion principle.*

Proof. Apply the refusal fork of Definition 386.142. It gives exactly two consumer-facing alternatives for the tail-budget meet. The consumer either factors through the finite envelope consisting of the StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ source row, DyadicRatCore$^{\uparrow}$ ledger, UniformCauchyCriterion$^{\uparrow}$ tail ledger, RealObservationBudget$^{\uparrow}$ rows, CauchyLimitSeal$^{\uparrow}$ seal row, and the shared BHist, hsame,

Cont, Pkg, and NameCert rows, or its request is routed to the visible refusal boundary for one of the listed non-envelope coordinates.

Use selector-budget transport from Definition 386.136 to identify the stable part of the admitted route. Transport preserves the same precision request, finite StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ ledger, RegSeqRat$^{\uparrow}$ handoff, Real$^{\uparrow}$ seal, BHist surface, Pkg provenance, and local NameCert row. Thus a transported selector read, budget read, or completion-facing read remains inside the displayed finite packet and gains no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.

Now inspect an obstruction row. If the requested coordinate is one of the listed non-envelope coordinates, the fork sends that coordinate to the refusal boundary because it is not a field of the finite envelope. If the requested coordinate is one of the envelope coordinates, transport keeps the request inside the admitted selector-budget packet and the obstruction row is not invoked. Therefore the row blocks only the named converse or reuse route that tries to read beyond the displayed packet.

This locality also fixes the interpretation of the obstruction. The proof has used only the finite envelope and the transported selector-budget packet, so the obstruction has no quantifier over all host Cauchy limits, no host theorem surface, and no external validation coordinate. It records that the specified consumer request cannot be served by this tail-budget meet route; it does not claim that the requested object is globally impossible or that an ambient completion principle has been validated elsewhere. □

386.2 CauchyLimitSeal regular tail-selector sibling link

Theorem 386.150 (CauchyLimitSeal regular tail-selector sibling link)**.** *Every accepted* CauchyLimitSeal$^{\uparrow}$ *seal route whose selector and observation-budget reads use the transport surface of Definition 386.136 factors first through the accepted* RegularCauchyTailSelector$^{\uparrow}$ *packet of chapter 458. The route reaches the shared* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal, and* RealObservationBudget$^{\uparrow}$ *meet only after the regular tail-selector packet has supplied its selected-tail witness and its finite support rows. Consequently the seal route exports no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier regular tail selector sibling link`

Proof. Begin with the regular tail-selector packet. The consumer package of Definition 458.9 supplies one finite selected-tail witness together with the requested precision, StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ regularity row, DyadicRatCore$^{\uparrow}$ comparison row, componentwise hsame transport, displayed Cont route, Pkg provenance, and local NameCert row.

Now attach the seal-side meet. By Definition 386.135, the CauchyLimitSeal$^{\uparrow}$ route and the RealObservationBudget$^{\uparrow}$ route share one finite meet over the same regular handoff, window, dyadic ledger, Real seal, transport, continuation, provenance, and naming rows.

Finally use selector-budget transport from Definition 386.136. It preserves the selector read, budget read, and completion-facing read along the same displayed route, so those reads cannot pass around the accepted RegularCauchyTailSelector$^{\uparrow}$ packet before consuming the shared window, ledger, handoff, seal, and meet rows.

The non-escape clauses of the cited selector package and meet route exclude quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness.

The sibling link is therefore a finite factorization through the displayed regular tail-selector packet rather than an additional completion principle. □

Theorem 386.151 (CauchyLimitSeal regular tail-selector budget compatibility)**.** *Every selector-budget read compatible with Definition 386.136 factors through the same accepted* RegularCauchyTailSelector$^{\uparrow}$ *packet before it reaches the* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and* RealObservationBudget$^{\uparrow}$ *rows. Thus the selector route, observation-budget route, and completion-facing route share one finite tail-selector factorization.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier regular tail selector budget compatibility`

Proof. Begin with the sibling-link theorem Definition 386.150. It places every compatible seal route through the accepted regular tail-selector packet before the stream, dyadic, regular-sequence, Real, and observation-budget rows are consumed.

Next apply selector-budget transport from Definition 386.136. The transported selector read, budget read, and completion-facing read remain on the same precision request, finite window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows.

Finally use the tail-budget meet of Definition 386.135. The meet binds the seal side and observation-budget side to the same finite route, so every compatible selector-budget read inherits the regular tail-selector factorization. □

Theorem 386.152 (CauchyLimitSeal regular tail-selector non-escape)**.** *After a compatible selector route is read through the accepted* RegularCauchyTailSelector$^{\uparrow}$ *packet, the* CauchyLimitSeal$^{\uparrow}$ *seal cannot export quotient stream identity, host real equality, a choice-selected limit, a hidden family limit, or ambient completeness. The exported boundary is only the selected-tail witness, the* StreamName$^{\uparrow}$ *window, the* DyadicRatCore$^{\uparrow}$ *ledger, the* RegSeqRat$^{\uparrow}$ *handoff, the* Real$^{\uparrow}$ *seal, the* RealObservationBudget$^{\uparrow}$ *meet, and the displayed* hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier regular tail selector nonescape`

Proof. Start from the budget compatibility theorem above. It fixes the selector, budget, and completion-facing reads on the same regular tail-selector packet and the same tail-budget meet.

Use the selector package theorem Definition 458.9. Its consumer surface contains the finite selected-tail witness and support rows, with no quotient representative, equality row, infinite-tail row, or completeness coordinate.

Finally apply Definition 386.136 and Definition 386.135. Transport preserves the finite route, and the meet adds no hidden completion coordinate. The named non-escape objects therefore cannot be exported after the selector route has been read. □

Theorem 386.153 (CauchyLimitSeal tail-selector window support determinacy)**.** *Every compatible* CauchyLimitSeal$^{\uparrow}$ *route that factors through the accepted* RegularCauchyTailSelector$^{\uparrow}$ *packet reads the same selected-tail witness,* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal, and* RealObservationBudget$^{\uparrow}$ *meet across the selector route, observation-budget route, and completion-facing route. Thus selector-facing, budget-facing, and completion-facing consumers share one finite support packet rather than three independent reads.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier tail selector window support determinacy`

Proof. First apply the sibling link of Definition 386.150. It routes every compatible seal read through the accepted regular tail-selector packet before the selected-tail witness, StreamName window, DyadicRatCore ledger, RegSeqRat handoff, Real seal, and observation-budget meet are consumed.

Next use budget compatibility from Definition 386.151. The selector route, observation-budget route, and completion-facing route are all factored through the same accepted packet and the same tail-budget meet, so their support rows are not independently chosen.

Finally apply the non-escape theorem Definition 386.152. Since no quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate is available, the three reads cannot diverge outside the displayed finite support. Their selected-tail witness, window, ledger, handoff, seal, and meet are therefore determined by the same packet. □

Theorem 386.154 (CauchyLimitSeal regular tail-selector public export)**.** *Every compatible public* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *export to a Real-facing completion consumer factors through one accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet, then through the shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{RealObservationBudget}^{\uparrow}$ *meet. The exported boundary is exactly this finite selector-support route and does not include quotient stream identity, host real equality, choice-selected limits, hidden family limits, or ambient completeness.*

Proof. First apply the sibling link of Definition 386.150. It places the public seal route through the accepted regular tail-selector packet before any Real-facing completion consumer can read the StreamName, DyadicRatCore, RegSeqRat, Real, or observation-budget rows.

Next use budget compatibility from Definition 386.151. The selector-facing, budget-facing, and completion-facing reads are the same finite route over the accepted packet and the tail-budget meet, so the public export cannot choose a second support surface.

Now apply the non-escape boundary of Definition 386.152. It removes quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness from the export inventory.

Finally use window support determinacy from Definition 386.153. The selected-tail witness, shared window, dyadic ledger, regular-sequence handoff, Real seal, and observation-budget meet are determined by the single accepted packet, so every compatible public export factors through precisely that route. □

Theorem 386.155 (CauchyLimitSeal regular tail-selector classifier stability)**.** *Let two accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *routes be related by the local classifier of the chapter. If the first route factors through the accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet of Definition 386.150, then the transported route factors through the transported regular tail-selector packet. The transported factorization preserves the selected-tail witness, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *meet, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No quotient stream identity, host real equality, choice-selected limit, hidden family limit, or ambient completeness coordinate is exported by the classifier transport.*

Proof. First apply the sibling link of Definition 386.150 to the source route. It gives the accepted regular tail-selector packet before the selected-tail witness, StreamName window, dyadic ledger, regular-sequence handoff, Real seal, observation-budget meet, transport, continuation, provenance, and naming rows are read.

Next use budget compatibility from Definition 386.151. The selector-facing, budget-facing, and completion-facing reads are one finite route, so the local classifier has a single support packet to transport rather than separate selector and budget witnesses.

Now apply choice-free selector stability from Definition 386.108. The classifier transport preserves the admitted selector route, the finite support windows, the dyadic ledger, the displayed hsame comparisons, and the Cont replay rows. Thus the transported route lands in the transported regular tail-selector packet with the same selected-tail witness surface.

Use window support determinacy from Definition 386.153 to identify the transported packet's public rows. The selected-tail witness, StreamName window, DyadicRatCore ledger, RegSeqRat handoff, Real seal, and RealObservationBudget meet are the rows determined by that one packet, and the displayed Pkg and NameCert rows travel with the same classifier route.

Finally invoke the non-escape theorem Definition 386.152. Since neither the source factorization nor the classifier transport contains quotient stream identity, host real equality, choice-selected limits, hidden family limits, or ambient completeness, the transported public export remains exactly the regular tail-selector factorization described above. □

Theorem 386.156 (CauchyLimitSeal StreamName cofinal tail-budget fusion)**.** *Every accepted* CauchyLimitSeal$^{\uparrow}$ *route whose selector-budget transport factors through the regular tail-selector sibling and whose completion-facing read uses the tail-budget meet factors through one finite envelope. The envelope consists of the* RegularCauchyTailSelector$^{\uparrow}$ *selected-tail witness, the* StreamName$^{\uparrow}$ *window, the* DyadicRatCore$^{\uparrow}$ *ledger, the* RegSeqRat$^{\uparrow}$ *handoff, the* Real$^{\uparrow}$ *seal, the* RealObservationBudget$^{\uparrow}$ *meet, the* CofinalTailBudget$^{\uparrow}$ *shared-tail row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Consequently the fused route exports no quotient stream identity, host sequence equality, hidden tail selector, countable-choice schedule, selected host limit, independent completion row, or ambient Real$^{\uparrow}$ *completeness.*

Proof. First apply the sibling link of Definition 386.150. It routes the accepted seal read through the regular tail-selector packet before the selected-tail witness, StreamName window, dyadic ledger, regular-sequence handoff, Real seal, transport, continuation, provenance, and naming rows are consumed.

Next use the tail-budget meet of Definition 386.135. The meet binds the selector-facing read, observation-budget read, and completion-facing read to the same finite route, so the completion consumer cannot choose a second row outside the selector packet.

Transport that common route by Definition 386.136. Componentwise hsame transport and the displayed Cont replay preserve the precision request, the finite StreamName$^{\uparrow}$ window, the DyadicRatCore$^{\uparrow}$ ledger, the RegSeqRat$^{\uparrow}$ handoff, the Real$^{\uparrow}$ seal, the BHist surface, the Pkg provenance, and the local NameCert row.

Now attach the cofinal shared-tail row by Definition 698.9. Its primitive scope is the finite shared-tail budget packet, so it adds only the public CofinalTailBudget$^{\uparrow}$ row aligned with the same regular handoff and completion-facing consumer.

Finally invoke the non-escape boundary of Definition 386.152. Together with the cofinal primitive scope, it removes quotient stream identity, host sequence equality, hidden tail selectors, countable-choice schedules, selected host limits, independent completion rows, and ambient Real$^{\uparrow}$ completeness from the exported route. □

Theorem 386.157 (CauchyLimitSeal regular tail-selector ProbeBundle coverage)**.** *Let Q be an accepted* CauchyLimitSeal$^{\uparrow}$ *route compatible with the regular tail-selector sibling link of Definition 386.150. Every finite* ProbeBundle *of selector-facing, observation-budget-facing, and completion-facing requests for Q factors through one finite envelope. The envelope consists of the* RegularCauchyTailSelector$^{\uparrow}$

selected-tail witness, the $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *meet, the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *shared-tail row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Thus finite bundled consumers receive the same route-level support as the pointwise fused route. The bundle exposes no quotient stream identity, host sequence equality, hidden selector, countable-choice schedule, selected host limit, independent completion row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct over the finite ProbeBundle. The empty bundle uses the finite envelope already displayed by the accepted route, with no consumer request to place outside it.

For a successor request, apply the sibling link of Definition 386.150 and budget compatibility from Definition 386.151. The new selector-facing, observation-budget-facing, or completion-facing read is forced through the accepted regular tail-selector packet before it can read the StreamName window, dyadic ledger, regular-sequence handoff, Real seal, observation-budget meet, transport, continuation, package, or naming rows.

Next use the tail-budget meet of Definition 386.135 and selector-budget transport from Definition 386.136. These identify the support of the new request with the accumulated support of the earlier bundle entries, so the induction step does not introduce a second budget row or a second selector packet.

Now attach the cofinal shared-tail row by the fusion theorem Definition 386.156. It places the $\mathsf{CofinalTailBudget}^{\uparrow}$ coordinate on the same StreamName, DyadicRatCore, RegSeqRat, Real, observation-budget, BHist, hsame, Cont, Pkg, and NameCert rows already carried by the bundle envelope.

Finally apply the non-escape theorem Definition 386.152. Since neither the pointwise route nor the finite induction step can export quotient stream identity, host sequence equality, hidden selectors, countable-choice schedules, selected host limits, independent completion rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the finite bundle repacks as one ProbeBundle coverage row over exactly the displayed envelope. □

Theorem 386.158 (CauchyLimitSeal ProbeBundle envelope terminality)**.** *Let Q be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route compatible with the regular tail-selector sibling link of Definition 386.150. Among all public factorization surfaces for finite* ProbeBundle *families of selector-facing, observation-budget-facing, and completion-facing requests for Q, the envelope of Definition 386.157 is terminal. It consists of the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *selected-tail witness, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *meet, the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *shared-tail row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Every alternative public route with the same visible reads is classifier-equal to this envelope. No quotient stream identity, host sequence equality, hidden selector, countable-choice schedule, selected host limit, independent completion row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness can separate a second public factorization surface from it.*

Proof. Start from the sibling link of Definition 386.150. It places every compatible accepted route through the regular tail-selector packet before any selector-facing, observation-budget-facing, or completion-facing request can read the StreamName window, dyadic ledger, regular-sequence handoff, Real seal, continuation, package, or naming rows.

Next combine budget compatibility from Definition 386.151, non-escape from Definition 386.152, and window-support determinacy from Definition 386.153. These three results force the selector, budget, and completion reads onto one displayed support packet and rule out every hidden coordinate that could distinguish another public surface.

Now use the cofinal tail-budget fusion theorem Definition 386.156. It attaches the CofinalTailBudget$^{\uparrow}$ shared-tail row to the same BHist, hsame, Cont, Pkg, and NameCert support, so the fused route has no independent completion row outside the regular tail-selector envelope.

Finally apply the finite ProbeBundle induction from Definition 386.157. The empty bundle has only the displayed envelope. Each successor request is already forced through that envelope by the preceding compatibility and fusion steps, so adding a request cannot create a second public route. Any alternative route with the same visible reads is therefore classifier-equal to the displayed envelope, which proves terminality. □

Corollary 386.159 (CauchyLimitSeal regular tail-selector strict local obstruction)**.** *Let an accepted* CauchyLimitSeal$^{\uparrow}$ *route be read through the regular tail-selector sibling surface of Definition 386.150. Any displayed failure, refusal, separation, or countermodel row attached to that sibling surface obstructs only the named converse or reuse route that tries to read outside the local factorization. Its scope is the selected-tail witness, the* StreamName$^{\uparrow}$ *window, the* DyadicRatCore$^{\uparrow}$ *ledger, the* RegSeqRat$^{\uparrow}$ *handoff, the* Real$^{\uparrow}$ *seal, the* RealObservationBudget$^{\uparrow}$ *meet, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It cannot be reinterpreted as a global impossibility for* CauchyLimitSeal$^{\uparrow}$ *packets, a host theorem about real equality or completeness, or a hidden external validation principle.*

Proof. Start with the non-escape theorem Definition 386.152. After the compatible selector route has been read through the accepted RegularCauchyTailSelector$^{\uparrow}$ packet, the exported boundary is exactly the selected-tail witness, the stream window, the dyadic ledger, the regular-sequence handoff, the Real seal, the observation-budget meet, and the displayed transport, continuation, provenance, and naming rows. The same theorem removes quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness from the export inventory.

Now apply classifier stability from Definition 386.155. Classifier transport preserves the same finite packet and the same displayed rows. Hence any consumer-facing use of a failure, refusal, separation, or countermodel row must factor through the transported local packet before it can be read by a selector-facing, budget-facing, or completion-facing consumer.

The obstruction is therefore strict. If the blocked request is a named converse or reuse route, the block occurs exactly where that route asks for an unlisted coordinate outside the finite packet. Since the classifier transport has no coordinate beyond the listed rows, the obstruction cannot be promoted to a global impossibility for other CauchyLimitSeal$^{\uparrow}$ packets, cannot become a host theorem asserting real equality or completeness, and cannot certify an external validation source. It records only the failure of the named local route to pass the displayed sibling boundary. □

386.3 CauchyLimitSeal finite-window lattice dependency

Theorem 386.160 (CauchyLimitSeal finite-window lattice dependency)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* CauchyLimitSeal$^{\uparrow}$ *carrier. Every* Real$^{\uparrow}$ *or completion consumer route that reads the seal row* E *factors first through the StreamName finite-window lattice meet surface of Definition 126.257, and then through the displayed* RegSeqRat$^{\uparrow}$ *source row* X*, the* DyadicRatCore$^{\uparrow}$ *ledger* D*, the* RealCauchyCompletion$^{\uparrow}$ *diagonal row* G*, and the* Real$^{\uparrow}$ *seal row* E*. The route carries only* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, and it exports no quotient stream equality, host real equality, selected host limit, hidden tail witness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier finite window lattice dependency`

Proof. Start with the carrier definition of Definition 386.1. The rows X, S, D, G, E, T, P, N are the only displayed coordinates available to a sealed consumer route.

Apply the finite-window factorization of Definition 386.3. Any read that reaches the seal must pass through a scheduled $\mathsf{StreamName}^{\uparrow}$ window, its displayed $\mathsf{RegSeqRat}^{\uparrow}$ source row, the matching dyadic ledger, and the completion diagonal row before the seal row is consumed.

Next pull the route back by Definition 386.20. This identifies the same scheduled window as the common dependency surface shared by the regular source, dyadic ledger, completion diagonal, and seal.

Use the StreamName lattice-meet exhaustion theorem Definition 126.257. The scheduled window therefore lands on the shared finite-window meet surface, whose readable coordinates are only the selected rational classifier rows, probe-bundle rows, membership rows, unary-history rows, dyadic ledger rows, componentwise transports, continuation routes, package provenance, and local names.

Finally apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. The sealed route can carry only the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance over X, D, G, E, so quotient stream equality, host real equality, selected host limits, hidden tail witnesses, and ambient completeness remain outside the exported surface. □

Theorem 386.161 (CauchyLimitSeal source-schedule lattice coherence)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ carrier. Any seal-facing consumer of L that factors through the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet uses the same displayed schedule row S, the same $\mathsf{RegSeqRat}^{\uparrow}$ source row X, the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, the same $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row G, and the same $\mathsf{Real}^{\uparrow}$ seal row E. No second $\mathsf{StreamName}^{\uparrow}$ window, quotient stream equality, selected limit, hidden tail witness, host real equality, or ambient completeness datum may replace this route.*

Proof. Use the finite-window lattice dependency of Definition 386.160. It places every seal-facing consumer route over the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet before the route reaches the displayed source, ledger, diagonal, and seal rows.

Apply the scheduled-window pullback of Definition 386.20. The pullback identifies the schedule row S as the common finite-window source for the regular source row X, dyadic ledger D, completion diagonal G, and seal row E.

Now use the StreamName lattice-meet exhaustion theorem Definition 126.257. Once the consumer has entered the meet, the readable window data are exhausted by the displayed finite-window coordinates and cannot be replaced by a second StreamName window or quotient stream equality.

Finally apply the RegSeqRat seal-route exclusion package of Definition 301.40. The regular-source seal route exports no selected limit, hidden tail witness, host real equality, or ambient completeness datum, so the only coherent source-schedule route left for the consumer is the displayed S, X, D, G, E handoff. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier source schedule lattice coherence`

Theorem 386.162 (CauchyLimitSeal RegSeqRat Real finite-window pullback)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ carrier. Suppose the source row X is consumed as a $\mathsf{RegSeqRat}^{\uparrow}$ packet and the seal row E is read by a $\mathsf{Real}^{\uparrow}$ consumer. Then the consumer route pulls back first through the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet, then through the $\mathsf{RegSeqRat}^{\uparrow}$ source row X, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal row G, and the $\mathsf{Real}^{\uparrow}$ seal row E. The pulled-back route exposes only BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance, and it exposes no quotient stream, selected limit, host real equality, or ambient completeness datum.*

Proof. Begin with the finite-window lattice dependency of Definition 386.160. It places every seal-facing read over the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet before the route reaches the displayed source, ledger, diagonal, and seal rows.

Next use the scheduled-window pullback of Definition 386.20. The scheduled window is the common dependency surface for the $\mathsf{RegSeqRat}^{\uparrow}$ source row X, the dyadic ledger D, the completion diagonal G, and the $\mathsf{Real}^{\uparrow}$ seal row E.

Apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. This fixes the endpoint read at the displayed seal row and permits only the visible BHist, hsame, Cont, Pkg, and NameCert provenance to travel with the consumer route.

Finally apply the RegSeqRat seal package exclusion of Definition 301.40. Since the regular source packet itself exports no quotient stream, selected limit, host real equality, or ambient completeness datum, the pulled-back CauchyLimitSeal route cannot acquire any such coordinate when the $\mathsf{Real}^{\uparrow}$ endpoint is consumed. □

Theorem 386.163 (CauchyLimitSeal middle-window meet determinacy)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Suppose two* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer routes reach the same seal row E through finite-window reads. After both routes are pulled back to the* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet, they use the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row X, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the same* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row G, and the same seal row E. Hence no second* $\mathsf{StreamName}^{\uparrow}$ *window, quotient stream equality, selected host limit, hidden tail witness, or ambient completeness datum can determine the seal route.*

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier middle window meet determinacy`

Proof. Begin with the finite-window lattice dependency of Definition 386.160. It places every $\mathsf{Real}^{\uparrow}$ or completion-facing seal read over the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet before the route reaches the displayed regular source, dyadic ledger, completion diagonal, and seal rows.

Apply the RegSeqRat Real finite-window pullback of Definition 386.162. Each route that reaches E is therefore pulled back through the same listed rows X, D, G, E, with only BHist, hsame, Cont, Pkg, and NameCert provenance available to the consumer.

Now use StreamName lattice-meet exhaustion, Definition 126.257. The two finite windows have a shared displayed meet surface, so any readable route data at the handoff is exhausted by the same finite-window coordinates rather than by a second stream window or a quotient-level stream equality.

Finally apply the RegSeqRat seal-route exclusion package of Definition 301.40. It rules out selected host limits, hidden tail witnesses, and ambient completeness data from the regular-source seal route. The only remaining route through the shared meet is the displayed X, D, G, E route, which gives determinacy of the middle finite-window handoff. □

Theorem 386.164 (CauchyLimitSeal common-window transport)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Suppose two seal-facing consumer routes first meet the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice surface before either route is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint. Transporting either route to this common finite window preserves the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row G, and the* $\mathsf{Real}^{\uparrow}$ *seal row E. The transported route exports only* BHist, hsame, Cont, Pkg, *and* NameCert *provenance, and it exports no quotient stream equality, host real equality, selected host limit, hidden tail witness, or ambient completeness datum.*

Proof. First use the middle-window meet determinacy of Definition 386.163. Once both seal-facing routes are pulled back to the shared $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet, the readable handoff is already fixed by the displayed rows X, D, G, E rather than by a second window or a quotient-level stream coordinate.

Next apply the RegSeqRat finite-window transport obligation of Definition 301.6. It identifies the transport task at the regular-source boundary as the preservation of the visible finite-window data over the same $\mathsf{StreamName}^{\uparrow}$ surface.

Then apply the RegSeqRat finite-window transport closure of Definition 301.14. This closes the transport inside BHist, hsame, Cont, Pkg, and NameCert provenance, so no quotient stream equality, host real equality, selected host limit, hidden tail witness, or ambient completeness coordinate is introduced by the move to the common window.

Finally use the CauchyLimitSeal RegSeqRat Real finite-window pullback of Definition 386.162. The $\mathsf{Real}^{\uparrow}$ endpoint therefore consumes the same displayed boundary rows X, D, G, E after transport, and the common finite window is reused for either seal-facing route. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier common window transport`

Theorem 386.165 (CauchyLimitSeal four-face finite-window lattice exhaustion)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every accepted consumer route that reaches a* $\mathsf{Real}^{\uparrow}$ *or completion endpoint through a* $\mathsf{StreamName}^{\uparrow}$ *finite-window meet is exhausted by exactly four displayed faces: the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{Real}^{\uparrow}$ *seal row E. The route carries only* BHist, hsame, Cont, Pkg, *and* NameCert *support, and it exports no quotient stream, host real equality, selected limit, ambient-completeness datum, hidden tail witness, metric-completeness datum, or standard bridge row.*

Proof. Begin with the finite-window lattice dependency of Definition 386.160. It places every endpoint-facing seal route over the $\mathsf{StreamName}^{\uparrow}$ finite-window meet before any regular source, dyadic ledger, completion diagonal, or seal row is read.

Apply the RegSeqRat Real finite-window pullback of Definition 386.162. This pulls the endpoint route back through the displayed source row X, tolerance ledger D, completion diagonal row G, and seal row E, with only BHist, hsame, Cont, Pkg, and NameCert support visible to the consumer.

Next use common-window transport from Definition 386.164. Transport to the shared finite window preserves the same displayed rows and does not create a second route through quotient stream equality, host real equality, selected limits, hidden tail witnesses, or ambient completeness.

Finally apply the StreamName finite-window lattice meet exhaustion theorem, Definition 126.257. Once the route has entered the meet, the consumer-visible lattice is already exhausted by the four displayed faces S, X, D, E together with the listed BEDC support rows. The completion diagonal remains an internal transport face for the seal route, while no metric-completeness datum or standard bridge row is exported to the endpoint. □

Theorem 386.166 (CauchyLimitSeal shared window route)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of L factors through the same finite route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RealCauchyCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

displayed by the rows S, X, D, G, E and carried by BHist, hsame, Cont, Pkg, *and* NameCert *provenance. The shared route is the root L10 handoff of Definition 386.176, and it has no quotient*

stream equality, selected limit, host real equality, hidden tail witness, or ambient completeness coordinate.

Proof. First apply the finite-window lattice dependency of Definition 386.160. It forces every seal-facing read to enter through the $\mathsf{StreamName}^{\uparrow}$ finite-window meet before any regular source, dyadic ledger, completion diagonal, or seal row is consumed.

Next use source-schedule lattice coherence from Definition 386.161. The meet route is not merely some finite route: it is the displayed schedule row S followed by the displayed X, D, G, E rows.

Finally transport along the common window by Definition 386.164. Transport preserves the displayed rows and the visible BHist, hsame, Cont, Pkg, and NameCert provenance. Composing this with the root L10 sibling route of Definition 386.176 gives the stated shared handoff and excludes every listed hidden coordinate. □

Theorem 386.167 (CauchyLimitSeal route audit projection)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. The route audit visible to downstream* $\mathsf{Real}^{\uparrow}$ *and completion consumers is exactly the tuple consisting of the* $\mathsf{StreamName}^{\uparrow}$ *window S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row G, the* $\mathsf{Real}^{\uparrow}$ *seal row E, and the structural provenance carried by* BHist, hsame, Cont, Pkg, *and* NameCert.

No consumer-visible audit row contains quotient stream equality, selected host limit, host real equality, hidden tail witness, ambient completeness, or an off-route seal source.

Proof. Begin with the shared window route of Definition 386.166. It identifies the only seal-facing route available to a downstream consumer with the displayed S, X, D, G, E handoff and its structural provenance.

Apply the $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 386.5. The endpoint projection may read the seal only through the displayed completion and seal rows, together with the visible history, transport, continuation, package, and local naming data.

Finally use the RegSeqRat seal-route exclusion package of Definition 301.40. The regular source contributes no quotient stream equality, selected host limit, host real equality, hidden tail witness, or ambient completeness coordinate. The audit projection is therefore exactly the displayed tuple and no larger route. □

Theorem 386.168 (CauchyLimitSeal five-face L10 exit route)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every accepted consumer route for L that reaches a* $\mathsf{Real}^{\uparrow}$ *or completion endpoint is exhausted by the ordered five-face L10 handoff*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RealCauchyCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

displayed by the rows S, X, D, G, E and carried only by BHist, hsame, Cont, Pkg, *and* NameCert *provenance. The route exports no quotient stream equality, selected host limit, host real equality, hidden tail witness, ambient completeness coordinate, or standard bridge row.*

Proof. First apply the finite-window lattice dependency of Definition 386.160. It places every endpoint-facing seal read over the $\mathsf{StreamName}^{\uparrow}$ finite-window meet before the regular source, dyadic ledger, completion diagonal, and real seal rows are consumed.

Next use the shared window route of Definition 386.166. The consumer route is therefore the ordered S, X, D, G, E handoff through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Now identify the audit surface by Definition 386.167. The downstream audit tuple contains exactly the finite window, source, dyadic ledger, completion diagonal, seal, and structural provenance rows, so no off-route row remains for the endpoint.

Finally apply the RegSeqRat seal-route exclusion package, Definition 301.40. It rules out quotient stream equality, selected host limit, host real equality, hidden tail witness, and ambient completeness on the regular-source side; the route audit projection rules out the same off-route coordinates on the seal side. Since no standard bridge row is among the displayed carrier rows, the five-face handoff is the whole L10 exit route. □

Theorem 386.169 (CauchyLimitSeal L10 four-face terminality)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every Phase Real and Completion consumer route for L terminates exactly at the four audited open-phase faces* $\mathsf{StreamName}^{\uparrow}(S)$*,* $\mathsf{RegSeqRat}^{\uparrow}(X)$*,* $\mathsf{DyadicRatCore}^{\uparrow}(D)$*, and* $\mathsf{Real}^{\uparrow}(E)$*. The completion diagonal G supplies only the internal completion-diagonal handoff, while T, P, N supply only transport, provenance, and local-name support.*

No quotient stream, host real equality, selected limit, countable-choice datum, ambient-completeness row, metric-completeness row, or standard bridge row is a terminal face of the accepted L10 exit route.

Proof. First apply the four-face finite-window lattice exhaustion theorem, Definition 386.165. It identifies the consumer-visible open-phase faces as the finite window S, regular source X, dyadic ledger D, and Real seal E, with only BHist, hsame, Cont, Pkg, and NameCert support.

Next use the shared window route of Definition 386.166. The same route passes through the displayed completion diagonal G, but that diagonal is an internal handoff between $\mathsf{DyadicRatCore}^{\uparrow}(D)$ and $\mathsf{Real}^{\uparrow}(E)$ rather than a fifth audited open-phase terminal face.

Now read the route audit projection from Definition 386.167. The audit tuple contains exactly the finite window, source, ledger, diagonal, seal, and structural provenance rows, so T, P, N can only act as transport, provenance, and local-name support for the four audited faces.

Finally apply the five-face L10 exit route of Definition 386.168. Its ordered handoff has no quotient stream equality, selected host limit, host real equality, hidden tail witness, ambient completeness coordinate, or standard bridge row. Since the four-face exhaustion already removes metric-completeness data from the endpoint surface, and no listed carrier row supplies countable choice, the accepted Phase Real and Completion route terminates exactly at $\mathsf{StreamName}^{\uparrow}(S)$, $\mathsf{RegSeqRat}^{\uparrow}(X)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$. □

Theorem 386.170 (CauchyLimitSeal window budget exhaustion)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. No* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer may replace the shared finite window of Definition 386.166 by a second schedule, quotient stream equality, selected host limit, hidden tail witness, or ambient completeness datum. The complete consumer budget is exhausted by the displayed finite window, the rows X, D, G, E, and the provenance* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with middle-window meet determinacy from Definition 386.163. Once a consumer read reaches the finite-window meet, the displayed rows X, D, G, E determine the handoff and a second $\mathsf{StreamName}^{\uparrow}$ schedule cannot determine the same seal route.

Next apply common-window transport from Definition 386.164. Transporting to the shared finite window preserves the carrier rows and does not introduce quotient stream equality, host real equality, selected host limit, hidden tail witness, or ambient completeness.

Finally compose with the route audit projection of Definition 386.167. Since the downstream audit tuple is exactly the displayed window, source, ledger, diagonal, seal, and structural provenance, all consumer-visible coordinates have been accounted for and the window budget is exhausted. □

Theorem 386.171 (CauchyLimitSeal window budget audit lock)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer whose audit projection reaches the seal row E is locked to the displayed route through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window meet, the schedule row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row G, the seal row E, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. No second schedule, quotient-stream equality, selected host limit, hidden tail witness, ambient completeness datum, or off-route seal source can replace any row of that audit route.*

Proof. Start with the finite-window lattice dependency of Definition 386.160. It places every downstream read that reaches E over the StreamName finite-window meet before the source, ledger, diagonal, and seal rows are consumed.

Apply common-window transport from Definition 386.164. Transport to the shared finite window preserves the displayed rows X, D, G, E and carries only the visible BHist, hsame, Cont, Pkg, and NameCert provenance.

Next use the shared window route of Definition 386.166. This identifies the route as the displayed S, X, D, G, E handoff rather than a second schedule, quotient stream, selected limit, hidden tail witness, or ambient completeness datum.

Finally combine the route audit projection of Definition 386.167 with the window budget exhaustion of Definition 386.170. The audit tuple has exactly the listed rows and provenance, and the exhausted budget leaves no off-route seal source or replacement row for a downstream consumer. □

Theorem 386.172 (CauchyLimitSeal Cauchy window transducer pullback)**.** *When a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer route is served through a* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet, the transducer rows pull back first to the* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet and then through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal rows. No quotient stream, selected host limit, hidden tail witness, or ambient completeness coordinate is introduced.*

Proof. Begin with Definition 386.164. It says that a seal-facing CauchyLimitSeal route transported to a common finite window preserves the displayed RegSeqRat source, dyadic ledger, completion diagonal, and Real seal rows while remaining inside BHist, hsame, Cont, Pkg, and NameCert provenance.

Apply Definition 1020.4. A Cauchy window transducer serving such a route can expose its RealUp seal row only after the displayed RegSeqRat readback row is available.

Finally use Definition 1020.5. The transducer rows can replay the finite window, tolerance, step, readback, selector handoff, transport, continuation, provenance, and naming data, but no row supplies a quotient stream, selected host limit, hidden tail witness, or ambient completeness coordinate. □

Theorem 386.173 (CauchyLimitSeal transducer route-budget exhaustion)**.** *When a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer route is served by a* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet and then read by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *endpoint, the complete route budget is exhausted by the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$

diagonal row, the $\mathsf{Real}^{\uparrow}$ *seal row, and the transducer finite-window, tolerance, step, selector-handoff, transport, continuation, provenance, and naming rows already cited by Definition 386.172. No quotient stream, selected host limit, hidden transducer schedule, hidden tail witness, host real equality, or ambient completeness coordinate remains available to the endpoint.*

Proof. Start with the Cauchy window transducer pullback of Definition 386.172. It places the served seal route over the $\mathsf{StreamName}^{\uparrow}$ finite-window lattice meet before the route reaches the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal rows.

Next apply the window budget exhaustion of Definition 386.170. This accounts for the CauchyLimitSeal-side budget: the shared finite window, the regular source, the dyadic ledger, the completion diagonal, the seal row, and the structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Now use the transducer handoff theorem Definition 1020.4. It forces the transducer service through the displayed readback row before a $\mathsf{Real}^{\uparrow}$ or completion-facing endpoint can consume the seal route.

Finally apply Definition 1020.5. The transducer contributes only its finite-window, tolerance, step, selector-handoff, transport, continuation, provenance, and naming rows. Since both sides of the handoff have exhausted their displayed budgets, no quotient stream, selected host limit, hidden transducer schedule, hidden tail witness, host real equality, or ambient completeness coordinate can be exported. □

Theorem 386.174 (CauchyLimitSeal transducer meet determinacy)**.** *If two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer routes are served by* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packets and both pull back to the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet, then the displayed schedule, source, ledger, diagonal, and seal rows*

$$S, X, D, G, E$$

together with the transducer finite-window, tolerance, step, selector-handoff, transport, continuation, provenance, and naming rows determine one consumer-visible route. The route exports no second schedule, quotient stream equality, selected host limit, hidden transducer schedule, hidden tail witness, host real equality, or ambient completeness coordinate.

Proof. Start with common-window transport from Definition 386.164. Once both served routes pull back to the same finite-window meet, transport preserves the displayed source, ledger, diagonal, seal, and structural rows rather than creating a second schedule.

Apply the transducer pullback theorem, Definition 386.172. It places the CauchyWindowTransducer service before the RegSeqRat readback and Real seal rows, so the transducer rows are read as part of the same finite-window route.

Now use transducer route-budget exhaustion from Definition 386.173. The transducer contributes only the displayed finite-window, tolerance, step, selector-handoff, transport, continuation, provenance, and naming rows.

Finally apply the window budget audit lock, Definition 386.171. With the CauchyLimitSeal rows and the transducer rows already exhausted, no hidden schedule, tail witness, quotient stream equality, selected host limit, host real equality, or ambient completeness coordinate remains to distinguish two consumer-visible routes over the same meet. □

Theorem 386.175 (CauchyLimitSeal window budget audit-lock formal handoff)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Suppose a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer route is served through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet and through the* $\mathsf{CauchyWindowTransducer}^{\uparrow}$

finite-window rows. Then the formal handoff exported to later consumers is exactly the displayed route

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RealCauchyCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the transducer finite-window, tolerance, step, selector-handoff, transport, continuation, provenance, and local naming rows, and with the visible BHist, hsame, Cont, Pkg, *and* NameCert *provenance carried by* L.

The handoff exports no quotient stream, selected host limit, hidden tail witness, host real equality, hidden transducer schedule, or ambient completeness coordinate. It also exports no route other than the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and transducer finite-window route.*

Proof. First build the CauchyLimitSeal side of the handoff. Apply the finite-window lattice dependency, Definition 386.160, and the common-window transport theorem, Definition 386.164. These place the consumer route over the shared $\mathsf{StreamName}^{\uparrow}$ finite-window meet and preserve the displayed X, D, G, E rows with only BHist, hsame, Cont, Pkg, and NameCert provenance.

Next identify the audit projection. Use Definition 386.167 to make the visible audit tuple exactly the window, source, dyadic ledger, completion diagonal, seal, and structural provenance rows. Then apply Definition 386.170 and Definition 386.171; the CauchyLimitSeal budget is exhausted by that tuple, so no quotient stream, selected host limit, hidden tail witness, host real equality, or ambient completeness coordinate remains on the seal side.

Now pull in the transducer service. The Cauchy-window transducer pullback theorem, Definition 386.172, places the transducer rows on the same finite-window route before the RegSeqRat readback and Real seal rows.

Finally apply transducer route-budget exhaustion and transducer meet determinacy, Definition 386.173 and Definition 386.174. These two results exhaust the displayed transducer rows and identify the same meet route for any served consumer, leaving exactly the stated formal handoff and no hidden transducer schedule or off-route completeness coordinate. □

386.4 CauchyLimitSeal root L10 sibling route

Theorem 386.176 (CauchyLimitSeal root L10 sibling route)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every accepted route from* L *to a* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal consumes, in order, the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic ledger of chapter 291, and the seal boundary of chapter 38. The route exports no quotient stream, selected host limit, host real equality, hidden tail oracle, or ambient completeness row.*

Proof. Begin with the finite-window lattice dependency of Definition 386.160. It places every Real or completion consumer route over the StreamName finite-window surface before the displayed regular source row, dyadic ledger, completion diagonal, and Real seal row are consumed.

Apply the RegSeqRat Real finite-window pullback of Definition 386.162. The pulled-back route reads the same $\mathsf{RegSeqRat}^{\uparrow}$ source row X, the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, and the same $\mathsf{Real}^{\uparrow}$ seal row E while carrying only BHist, hsame, Cont, Pkg, and NameCert provenance.

Finally use the common-window transport of Definition 386.164. Transport to the shared finite window preserves the displayed X, D, G, E route and adds no quotient stream equality, host real equality, selected host limit, hidden tail witness, or ambient completeness coordinate. Hence the

root route is exactly the sibling route through StreamName, RegSeqRat, DyadicRatCore, and RealUp. □

Lean4 ✓ `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier root l10 sibling route certificate`
↪ variant `BEDC.Derived.CauchyLimitSealUp.CauchyLimitSealCarrier root l10 sibling route`

Theorem 386.177 (CauchyLimitSeal L10 sibling route totality)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer route for L passes through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row before any local* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *read is public. A route missing any one of these four sibling surfaces is outside the accepted L10 sibling route.*

Proof. Start with the finite-window lattice dependency of Definition 386.160. It places the consumer over the StreamName window before the displayed source and seal rows are read.

Apply the RegSeqRat pullback of Definition 386.162. The consumer must use the regular readback row X and the same dyadic ledger D when it reaches the Real-facing seal row E.

Finally use the root sibling route theorem Definition 386.176. It identifies the complete route through StreamName, RegSeqRat, DyadicRatCore, and RealUp, so omitting one sibling surface leaves no accepted L10 route to the local NameCert row. □

Theorem 386.178 (CauchyLimitSeal stream-RegSeq seal factorization)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier $L = (X, S, D, G, E, T, P, N)$, every L10 consumer route factors through the displayed X, D, G, E rows together with the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance carried by T, P, N. The factorized route is the same StreamName finite-window, RegSeqRat regular-source, DyadicRatCore ledger, and RealUp seal route used in Definition 386.176.*

Proof. Use route totality from Definition 386.177 to place the consumer on the StreamName, RegSeqRat, DyadicRatCore, and RealUp sibling surface.

The RegSeqRat pullback then identifies the regular source with X and the dyadic ledger with D. The finite-window lattice dependency identifies the completion diagonal and Real-facing seal with the displayed G, E rows.

The common-window transport of Definition 386.164 carries this factorization through the same BHist, hsame, Cont, Pkg, and NameCert provenance. Thus the route is a factorization through the displayed carrier rows rather than a fresh analytic route. □

Theorem 386.179 (CauchyLimitSeal no unbudgeted limit)**.** *An accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *L10 sibling route exports no quotient stream, selected limit, hidden tail oracle, host real equality, or ambient completeness coordinate. Every limit-facing read is budgeted by the finite-window lattice, the RegSeqRat pullback, and the common-window transport before it reaches the displayed RealUp seal row.*

Proof. Apply the factorization theorem Definition 386.178. It reduces every L10 consumer route to the displayed X, D, G, E, T, P, N rows and their BHist, hsame, Cont, Pkg, and NameCert provenance.

The finite-window lattice supplies the StreamName budget, the RegSeqRat pullback supplies the regular-source budget, and common-window transport keeps both budgets on the same finite route. These three dependencies account for every limit-facing read before the RealUp seal is public.

Since quotient streams, selected limits, hidden tail oracles, host real equality, and ambient completeness are not coordinates of the factorized route, none can be exported by an accepted L10 sibling route. □

Theorem 386.180 (CauchyLimitSeal root source-schedule lattice consumption)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every accepted root L10 route first consumes the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule row S, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row D, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row G, and the* $\mathsf{Real}^{\uparrow}$ *seal row E, with transport, package, and naming supplied only by T, P, N. The route has no second schedule, quotient stream, selected host limit, host equality, or ambient completeness coordinate.*

Proof. Begin with the root L10 sibling route of Definition 386.176. It places every accepted root route on the StreamName, RegSeqRat, DyadicRatCore, and RealUp sibling surface and excludes quotient streams, selected host limits, host equality, hidden tail oracles, and ambient completeness rows.

Apply source-schedule lattice coherence from Definition 386.161. This strengthens the first step of the route: the StreamName surface is the displayed schedule row S, and the route continues through the displayed X, D, G, E rows rather than through a second schedule or an independent stream.

Use L10 sibling route totality from Definition 386.177 to rule out routes that omit any of the sibling rows. Finally apply Definition 386.179; the factorized route has no coordinate from which a selected host limit, host equality, quotient stream, or ambient completeness principle could be exported. □

Theorem 386.181 (CauchyLimitSeal root envelope-refinement exhaustion)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every root L10 route that reaches a* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal through the source-schedule lattice factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal, then through the finite witness-envelope surface of* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *and the bridged route surface of* $\mathsf{CauchyModulusRefinement}^{\uparrow}$*. The route exports no quotient stream, host real equality, selected host limit, hidden selector, independent completion carrier, or ambient completeness coordinate.*

Proof. First apply the source-schedule lattice consumption theorem Definition 386.180. It fixes the root L10 route on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before any Real-facing or completion-facing consumer can read the seal.

Next use the stream-RegSeq seal factorization of Definition 386.178. The factorized route carries only the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance attached to T, P, N, so there is no second stream source or unrecorded selector row.

The Real tail side is supplied by Definition 898.7. It places the tail witness read on the finite envelope surface rather than on a host limit, host equality, or quotient stream. The refinement side is supplied by Definition 885.154, which routes the modulus refinement through the displayed bridged surface before it can serve completion-facing consumers.

Finally apply the no-unbudgeted-limit theorem Definition 386.179. Since quotient streams, host real equality, selected host limits, hidden selectors, independent completion carriers, and ambient completeness coordinates do not occur in the displayed packet rows, no accepted route can export them. □

Theorem 386.182 (CauchyLimitSeal root sibling route exactness)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. An accepted root* $\mathsf{Real}^{\uparrow}$ *or completion-facing route exists*

exactly when it passes through the displayed $\mathsf{StreamName}^{\uparrow}$ *finite-window surface,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal row, and* $\mathsf{Real}^{\uparrow}$ *seal row, followed only by the transport, provenance, and local naming rows* T, P, N*.*

No omitted sibling row, quotient stream, selected host limit, host real equality, hidden tail oracle, independent completion carrier, or ambient completeness object can serve as an accepted root route for L*.*

Proof. For existence, use the root L10 sibling route and route totality. Theorems 386.176 and 386.177 place every accepted root Real or completion-facing consumer on the displayed StreamName, RegSeqRat, DyadicRatCore, and RealUp sibling surface.

Next apply stream-RegSeq seal factorization. Theorem 386.178 reads the same X, D, G, E rows together with the BHist, hsame, Cont, Pkg, and NameCert provenance carried by T, P, N, so the route is a displayed BEDC packet rather than an external completion object.

For necessity, use the no-unbudgeted-limit theorem. Theorem 386.179 removes quotient streams, selected limits, hidden tail oracles, host real equality, and ambient completeness from the accepted route surface.

Finally combine source-schedule lattice consumption with envelope-refinement exhaustion. Theorems 386.180 and 386.181 force the schedule, regular readback, dyadic ledger, completion diagonal, Real seal, and the subsequent T, P, N rows to appear in the displayed order. Omitting any sibling row or replacing it by a host-side completion object therefore leaves no accepted root route. □

Theorem 386.183 (CauchyLimitSeal budget-tail root join)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Suppose a* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing read and a* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$*-facing tail read are accepted through* L *over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger. Then the two reads join on one displayed L10 packet through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The joined packet exports the shared finite window, regular source row, dyadic ledger, RealUp seal surface, transport, route, package, and local naming rows. It exports no quotient stream, selected limit, host real equality, hidden family limit, hidden tail oracle, independent completion endpoint, or ambient completeness coordinate.

Proof. First apply root source-schedule lattice consumption from Definition 386.180. It fixes every accepted root route on the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal row, and structural BHist, hsame, Cont, Pkg, and NameCert rows.

Next use root envelope-refinement exhaustion from Definition 386.181. The tail-facing read is forced through the finite envelope surface rather than through a hidden host limit, quotient stream, or independent completion endpoint.

Now apply budget root source totality from Definition 386.87. Since the budget-facing read uses the same finite $\mathsf{StreamName}^{\uparrow}$ window and the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, it has no second source surface outside the displayed root L10 packet.

The tail boundary is supplied by Definition 898.7. It places the $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ leg on the same finite witness-envelope route before any RealUp readback is public.

Finally apply the no-unbudgeted-limit theorem Definition 386.179. The joined route contains only the displayed packet rows, so it cannot export a quotient stream, selected limit, host real equality, hidden family limit, hidden tail oracle, independent completion endpoint, or ambient completeness coordinate. □

Theorem 386.184 (CauchyLimitSeal selector-seal compatibility)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier, and let $Q = (B, W, R, L', E', H, C, P', N')$ be an accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose the two packets share the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint row, so that W is the displayed window of S, R is the displayed regular readback through D, $L' = L$, and $E' = E$.*

Then the selector-seal public route of Q factors through the public route packet of L. The consumer reads one displayed L10 handoff

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the shared finite-window, dyadic-ledger, transport, continuation, provenance, and local naming rows. It exposes no second completion route, quotient stream, host real equality, hidden selector, or ambient completeness coordinate.

Proof. First fix the CauchyLimitSeal route. By Definition 386.182, the accepted root route for L is exactly the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ packet, followed only by the transport, provenance, and local naming rows.

Next join the visible budget and tail reads. The budget-tail root join theorem, Definition 386.183, says that reads using the same finite window and the same dyadic ledger meet on one displayed L10 packet. Hence the shared W, R, L', E' rows of Q do not open an independent completion surface.

Finally identify the selector-seal route. The public route export theorem, Definition 1081.29, gives the route $B \to W \to R \to L' \to E'$, while Definition 1081.5 places that route inside the same L10 Real and Completion handoff. Since $L' = L$ and $E' = E$, the selector-seal consumer factors through the CauchyLimitSeal packet fixed in the first step, and the excluded coordinates are absent from both displayed carrier surfaces. □

Lemma 386.185 (CauchyLimitSeal root L10 downstream coverage)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every accepted local consumer that reads the root L10* $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *route for a* $\mathsf{Real}^{\uparrow}$ *or completion-facing use is covered by the displayed downstream rows*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RealCauchyCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the transport, provenance, package, and naming rows carried by T, P, N. Thus the consumer can use only the finite-window surface S, the regular readback row X, the dyadic ledger D, the completion diagonal G, the seal row E, and the visible BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support. A candidate local read that asks instead for a quotient stream, selected host limit, host real equality, hidden tail oracle, second finite-window source, independent completion carrier, or ambient completeness coordinate is outside the accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet.*

Proof. Apply the L10 sibling route totality theorem, Definition 386.177. Before any local $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ read is public, the consumer route has already passed through the $\mathsf{StreamName}^{\uparrow}$ finite-window surface, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seal row. Therefore any read that omits one of those sibling surfaces has no accepted route to the local naming row.

Now apply the common-window transport theorem, Definition 386.164, and then the root L10 sibling route theorem, Definition 386.176. The transport theorem carries the total route to the shared finite-window surface while preserving the displayed X, D, G, E rows and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. The root theorem then identifies this transported total

route with the root route through StreamName, RegSeqRat, DyadicRatCore, and RealUp, and it excludes quotient streams, selected host limits, host real equality, hidden tail oracles, and ambient completeness rows from the exported surface.

It remains only to identify the downstream rows available after the route is fixed. The stream-RegSeq seal factorization theorem, Definition 386.178, says that every L10 consumer factors through the displayed X, D, G, E rows and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance carried by T, P, N. The source-schedule lattice consumption theorem, Definition 386.180, adds the displayed schedule row S and rules out a second schedule or independent stream source.

Finally use the no-unbudgeted-limit theorem, Definition 386.179. Since the factorized downstream route contains only the listed finite-window, regular-readback, ledger, diagonal, seal, transport, provenance, package, and naming rows, none of the excluded coordinates can be consumed as part of an accepted local $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet. □

386.5 CauchyLimitSeal DyadicRatCore terminal dependency

Theorem 386.186 (CauchyLimitSeal DyadicRatCore terminal dependency)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ carrier. Every accepted terminal route from L to a $\mathsf{Real}^{\uparrow}$ or completion-facing seal consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291 before the displayed $\mathsf{Real}^{\uparrow}$ seal row is public.*

Equivalently, the accepted terminal route reaches $\mathsf{Real}^{\uparrow}$ only through the same StreamName finite-window surface, RegSeqRat regular readback, DyadicRatCore ledger, and RealUp seal packet fixed by the root L10 sibling route. No route that bypasses the DyadicRatCore ledger can be an accepted terminal CauchyLimitSeal route.

Proof. Start with the root L10 sibling route of Definition 386.176. It places every accepted Real-facing or completion-facing route through the StreamName surface, the RegSeqRat readback, the DyadicRatCore ledger, and the RealUp seal row in that order.

Use L10 sibling route totality from Definition 386.177. A terminal consumer that omits the dyadic ledger also omits one of the four required sibling surfaces, so it is outside the accepted L10 route before it can reach the local $\mathsf{NameCert}_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ row.

Finally apply the stream-RegSeq seal factorization of Definition 386.178. The factorized route reads the displayed X, D, G, E rows, with D serving as the DyadicRatCore tolerance ledger, before the RealUp seal row E is public. Thus DyadicRatCore is a terminal dependency of every accepted CauchyLimitSeal seal route. □

Theorem 386.187 (CauchyLimitSeal RegSeq-Real sibling dependency)**.** *Every accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff consumes $\mathsf{StreamName}^{\uparrow}$ for finite observations, $\mathsf{RegSeqRat}^{\uparrow}$ for regular Cauchy rows, and $\mathsf{Real}^{\uparrow}$ for the public seal and readback row. The handoff therefore factors through chapter 126, chapter 301, and chapter 38; it cannot be read as a standalone completion principle.*

Proof. Begin with the root L10 sibling route, Definition 386.176. It already places every accepted Real-facing or completion-facing route through the finite-window surface, regular readback, dyadic ledger, and Real seal boundary.

The finite observations are supplied by the StreamName surface. The scheduled window row records the finite observations before any regular Cauchy source or seal row is public.

The regular Cauchy content is supplied by the RegSeqRat row. The accepted handoff reads the regular source and its readback before it reaches the seal endpoint.

The public endpoint is supplied by the RealUp seal row. Since the accepted route has no coordinate for quotient streams, selected host limits, hidden tail oracles, host equality, or ambient

completeness, the CauchyLimitSeal packet is a sibling-dependent handoff rather than an isolated completion principle. □

Theorem 386.188 (CauchyLimitSeal RegSeqRat–Real handoff exactness)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. An accepted consumer route for L whose public endpoint is a* $\mathsf{Real}^{\uparrow}$ *seal is exact if and only if it factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *regular row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the local* $\mathsf{Real}^{\uparrow}$ *seal row, with all transport carried only by the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the packet.*

Equivalently, the exact Real-facing handoff is the finite route

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

read inside the public route packet of L. No quotient stream, selected host limit, host real equality, hidden tail oracle, independent completion object, or ambient completeness coordinate belongs to that exact handoff.

Proof. For the forward direction, start from the sibling dependency theorem Definition 386.187. It already places each accepted $\mathsf{Real}^{\uparrow}$ handoff through the StreamName finite observations, RegSeqRat regular rows, and RealUp public seal surface.

Next refine the dependency by the root L10 sibling route, Definition 386.176. This identifies the missing observation ledger as the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row and keeps the route inside the same finite-window, regular-source, dyadic-ledger, and Real-seal packet.

Finally apply the public route packet and non-escape boundary from Definition 386.6 and Definition 386.4. These surfaces leave only the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ transport and provenance rows, so an exact public endpoint cannot pass through an omitted sibling row or an external completion coordinate.

For the reverse direction, assume the consumer factors through the displayed StreamName window, RegSeqRat row, DyadicRatCore ledger, RealUp seal, and structural BEDC rows. The public route boundary of Definition 386.8 repacks those rows as the accepted public route packet of L.

Use the RealUp consumer boundary, Definition 386.5, to read the terminal seal from that packet. Because every coordinate of the proposed route is already a carrier row or one of its visible transport, continuation, provenance, and naming rows, the repacked route is an admissible $\mathsf{Real}^{\uparrow}$ consumer handoff.

The non-escape boundary then rules out any second route through quotient streams, selected limits, host equality, hidden tail data, independent completion objects, or ambient completeness. Thus the displayed factorization is not merely sufficient for a Real-facing read; it is exactly the accepted CauchyLimitSeal handoff surface. □

Theorem 386.189 (CauchyLimitSeal RegSeq–Real bridge determinacy)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any consumer-facing use of the RegSeq–Real sibling bridge for L, whether presented as a route, a transport, a readback, or a handoff, is admitted only through the local rows displayed by Definition 386.188: the* $\mathsf{StreamName}^{\uparrow}$ *finite window S, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *regular row X, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D, the local* $\mathsf{Real}^{\uparrow}$ *seal row E, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows carried by T, P, N.*

Consequently the bridge is determinate at the consumer surface: every accepted use factors through the finite public route

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

inside the public route packet of L. A requested quotient stream, selected host limit, host real equality, hidden tail oracle, independent completion object, alternate readback, second finite window, or ambient completeness coordinate is not a bridge coordinate of this RegSeq–Real sibling dependency.

Proof. Start with the sibling dependency theorem Definition 386.187. It says that an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff is not a standalone completion principle: the handoff consumes the StreamName finite observations, the RegSeqRat regular Cauchy rows, and the RealUp public seal and readback row.

Refine that dependency by the handoff exactness theorem Definition 386.188. Exactness identifies the consumer-facing bridge as the displayed finite route through the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ regular row, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and the $\mathsf{Real}^{\uparrow}$ seal, with transport carried only by the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Thus any route, transport, readback, or handoff that is accepted by the consumer has already been forced through those local rows.

It remains to rule out hidden alternatives. The root L10 sibling route of Definition 386.176 places every Real-facing or completion-facing route through the same finite-window, regular-readback, dyadic-ledger, and Real-seal surface and exports no quotient stream, selected host limit, host real equality, hidden tail oracle, or ambient completeness row.

Finally use the public route packet and boundary, Definition 386.6 and Definition 386.8. They identify the public coordinates of L as the displayed carrier rows and their visible transport, package, and naming data. The non-escape boundary Definition 386.4 prevents any omitted coordinate from being supplied after the route reaches the public surface. Therefore the cited local rows exhaust the RegSeq–Real sibling bridge, and every unlisted coordinate is unavailable to a consumer. □

Corollary 386.190 (CauchyLimitSeal RegSeq–Real sibling dependency determinacy)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ carrier, and suppose a consumer-facing use of the RegSeq–Real sibling dependency is accepted after the displayed local rows*

$$S, X, D, E, T, P, N$$

are fixed. Then the use has only the finite consumer surface named in Definition 386.189: the route

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow},$$

the corresponding window-ledger normal form, and the exactness decision supplied by Definition 386.188 are all read from the same displayed rows. In particular, an accepted route, transport, readback, handoff, normal-form claim, or exactness claim cannot choose an alternate finite window, a second regular row, a duplicate classifier source, a hidden tail coordinate, a quotient stream, a selected host limit, host real equality, an independent completion object, or ambient completeness data.

Proof. Apply the RegSeq–Real bridge determinacy theorem, Definition 386.189. It states that every consumer-facing use of the sibling bridge, whether presented as a route, transport, readback, or handoff, is admitted only through the local rows displayed by Definition 386.188: the finite window S, the regular row X, the dyadic ledger D, the seal row E, and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows carried by T, P, N.

Now use the handoff exactness theorem, Definition 386.188. Its exactness criterion identifies the accepted Real-facing handoff precisely with the displayed finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with all transport carried only by the visible packet rows. Hence a consumer that asks for an exactness decision has no independent source from which to

answer that decision; it must answer by checking whether the proposed use factors through the same finite route.

The same reasoning fixes the normal-form reading. A normal-form claim for this sibling dependency is a consumer-facing presentation of the accepted bridge, so by bridge determinacy it is already confined to the displayed rows. Since exactness makes those rows the entire Real-facing handoff surface, the normal form cannot split off a second window, a second regular readback, or a duplicate classifier source.

The excluded coordinates are therefore unavailable for every accepted consumer presentation. A quotient stream, selected host limit, host real equality, hidden tail coordinate, independent completion object, ambient completeness datum, alternate finite window, or duplicate classifier source is not among the rows named by determinacy and is also not part of the exact handoff surface. Once the local rows are fixed, every accepted use must pick the same finite route, the same normal-form surface, and the same exactness decision named by the chapter. □

Theorem 386.191 (CauchyLimitSeal tail-budget handoff determinacy)**.** *Let $L = (X, S, D, G, E, T, P, N)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. A* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer that reads the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, and local seal rows G, E, T, P, N receives a single displayed handoff route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Any two accepted reads with those same displayed rows agree up to the componentwise* hsame *transport and displayed* Cont *replay of T.*

The handoff route is only the finite L10 completion seal route. It does not identify quotient streams, selected host limits, host real equality, hidden selectors, independent completion endpoints, or ambient completeness data.

Proof. Start with the StreamName dependency theorem Definition 386.116. It forces every sealed-limit read through the scheduled finite window before the local seal row is consumed.

Next apply the tail-budget meet of Definition 386.135. The selector-facing read, observation-budget read, and completion-facing read meet on the same precision request, stream window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and naming rows.

Use the regular-tail selector sibling link, Definition 386.150. It places compatible selector reads before the StreamName, DyadicRatCore, RegSeqRat, Real, and observation-budget rows and prevents the selector side from becoming a detached support surface.

Finally apply the root L10 sibling route of Definition 386.176. The route through StreamName, RegSeqRat, DyadicRatCore, and RealUp is therefore the single displayed handoff route, with all equality of accepted reads mediated only by hsame transport and Cont replay. □

Theorem 386.192 (CauchyLimitSeal StreamName non-escape)**.** *For an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier $L = (X, S, D, G, E, T, P, N)$, the StreamName-facing completion seal route cannot import quotient stream equality, selected limit choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a second tail selector outside the displayed S, X, D, G, E, T, P, N rows. Every public consumer receives only the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget handoff, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the diagonal and Real seal rows, componentwise transport, replay, provenance, and local naming.*

Proof. Begin with Definition 386.116. The StreamName row is a required dependency of the sealed-limit read, so a consumer cannot reach E before passing through the scheduled window row S.

Apply the tail-budget consumer normal form of Definition 386.140. It lists the exact budget, handoff, seal, transport, continuation, package, and naming rows available to a RealUp or completion consumer.

Use the regular-tail selector non-escape theorem Definition 386.152. It removes quotient stream identity, host real equality, choice-selected limits, hidden family limits, and ambient completeness from selector-compatible seal reads.

Finally use the root sibling route exactness of Definition 386.182. The accepted root route has no coordinate for a second tail selector or an independent completion endpoint, so the StreamName-facing consumer remains inside the displayed finite route. □

Corollary 386.193 (CauchyLimitSeal tail-budget handoff strict local obstruction)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Any displayed obstruction, refusal, separation, or countermodel reading attached to the tail-budget handoff in this file is strict and local: it blocks only the attempted converse or reuse route that tries to replace the displayed* S, X, D, G, E, T, P, N *packet by quotient stream equality, selected limit choice, host* $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, a hidden selector, a second tail selector, an independent completion endpoint, or hidden external validation.*

The obstruction cannot be reinterpreted as a global impossibility theorem about Cauchy completion, as a host theorem about $\mathsf{Real}^{\uparrow}$*, or as an external validation principle. Every consumer-facing use still factors through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-budget handoff, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the diagonal and* $\mathsf{Real}^{\uparrow}$ *seal rows, the componentwise* hsame *transports, the displayed* Cont *replay, the* Pkg *provenance, and the local* NameCert *row; every unlisted coordinate is rejected only as an attempted exit from that finite route.*

Proof. Start with the handoff determinacy theorem Definition 386.191. It says that a consumer reading the displayed $\mathsf{StreamName}^{\uparrow}$ window rows, $\mathsf{RegSeqRat}^{\uparrow}$ tail-budget rows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rows, and local seal rows G, E, T, P, N receives a single handoff route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. Agreement of accepted reads is mediated only by the componentwise hsame transport and the displayed Cont replay of T.

Apply StreamName non-escape from Definition 386.192. It identifies the consumer-facing export inventory as the scheduled $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ tail-budget handoff, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the diagonal and Real seal rows, componentwise transport, replay, provenance, and local naming. In particular, quotient stream equality, selected limit choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and a second tail selector are not coordinates outside the displayed S, X, D, G, E, T, P, N rows.

Now consider any proposed reading of a displayed obstruction as more than a local block. If it is used to distinguish two accepted handoff reads, determinacy sends both reads back to the same displayed route and permits comparison only through hsame transport and Cont replay. If it is used to import a quotient stream, selected limit, host equality, hidden selector, independent completion endpoint, or ambient completeness row, non-escape rejects that coordinate because it is absent from the displayed StreamName-facing route. Thus the only route actually blocked is the attempted converse or reuse route from an unlisted coordinate back into the accepted finite packet.

Nothing in this argument constructs a global countermodel, a host-level impossibility theorem, or an external validation channel. The proof uses only the two local route facts just cited: determinacy of the displayed handoff and non-escape of the StreamName-facing export surface. The obstruction is therefore strict at the local handoff boundary and has no force outside the named converse or reuse route. □

386.6 CauchyCriterion Lattice Dependency

Theorem 386.194 (CauchyLimitSeal CauchyCriterion lattice dependency)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer route that exposes a* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal first consumes the* $\mathsf{CauchyCriterion}^{\uparrow}$ *budget-selector boundary of chapter 392. The resulting route factors through the existing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route rows together with the finite selector boundary before any seal row is public, and it exports no quotient stream, selected host limit, host real equality, countable-choice selector, hidden tail oracle, or ambient completeness coordinate.*

Proof. Start with the $\mathsf{CauchyLimitSeal}^{\uparrow}$ root L10 sibling route. It places every Real or completion-facing consumer on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local seal is read.

Apply the $\mathsf{CauchyCriterion}^{\uparrow}$ finite selector boundary of chapter 392. That boundary supplies the budget-selector row used by the same finite window and tolerance ledger, so the seal route cannot bypass the criterion packet on its way to a Real or completion-facing consumer.

The two route surfaces share only displayed BHist, hsame, Cont, Pkg, and NameCert rows. Therefore the combined consumer route exposes the finite criterion selector and the local CauchyLimitSeal packet, not any excluded quotient, host-limit, host-equality, choice, hidden-oracle, or ambient-completeness coordinate. □

386.7 CauchyLimitSeal RealCompletionExactBoundary sibling route

Theorem 386.195 (CauchyLimitSeal exact-boundary sibling route)**.** *Let* $L = (X, S, D, G, E, T, P, N)$ *be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *carrier. Every* RealCompletionExactBoundaryUp *consumer that uses the CauchyLimitSeal route must first read the public route packet of* L *through the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows before it can use the exact-boundary packet of chapter 1120. The sibling route adds no quotient stream, selected host limit, countable choice, host real equality, or ambient completeness premise.*

Proof. Start with the public route boundary of Definition 386.8. It places every completion-facing read of L inside the public route packet X, S, D, G, E, T, P, N, with the scheduled StreamName window, regular RegSeqRat source, DyadicRatCore ledger, RealCauchyCompletion diagonal, RealUp seal, transport, provenance, and local naming rows all displayed.

Next apply the root L10 sibling route of Definition 386.176. The route is the finite sibling route through StreamName, RegSeqRat, DyadicRatCore, and RealUp, and the structural tail of the packet is carried only by hsame, Cont, Pkg, and NameCert rows. Use the root sibling route exactness of Definition 386.182 to make this route necessary for any accepted root completion-facing consumer.

The exact-boundary packet of chapter 1120 is therefore consumed only after that displayed CauchyLimitSeal route is in scope. Since the CauchyLimitSeal route already excludes quotient streams, selected host limits, host real equality, hidden tail oracles, and ambient completeness rows, and since the exact-boundary packet records only the finite Real-completion boundary surface, the sibling handoff introduces none of those absent coordinates and no countable-choice premise. □

Chapter 387

A Concrete Naming Certificate for $\mathsf{CauchyTailComparison}^{\uparrow}$

The $\mathsf{CauchyTailComparison}^{\uparrow}$ interface records the finite common-tail comparison certificate used when a regular rational Cauchy name and its real-completion readback must be compared through the same visible window data. Following Bishop and Bridges, *Constructive Analysis*, where Cauchy comparison is controlled by explicit tail moduli and finite rational estimates, the packet stores two regular-name surfaces, one common modulus, one finite stream window, dyadic endpoint comparisons, tail-comparison rows, the $\mathsf{Real}^{\uparrow}$ classifier readback, package provenance, and a local naming row. It does not form quotient classes of streams, choose a host limit, compare host real values, or appeal to ambient completeness.

The common-tail comparison consumes the finite-window observation surface of chapter 126, the regular-name rows of chapter 301, the dyadic endpoint ledger of chapter 291, and the completion-facing handoff used by chapter 406. These sibling carriers determine where X, Y, μ, W, T, and R may be read: the two names are compared only after a shared stream window, through displayed dyadic endpoints, and with the readback row already shaped for the completion interface.

The shared window and dyadic endpoint rows are specifically read through the finite-window envelope surface of chapter 365. Thus the common-window comparison packet consumes the envelope's selected window and endpoint ledger before exporting the $\mathsf{Real}^{\uparrow}$ completion readback, rather than acting as an isolated host-level comparison.

Definition 387.1 (CauchyTailComparison carrier)**.** *A* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, \mu, W, T, R, P, N)$$

with the following visible rows:

1. *X and Y are the two displayed regular Cauchy names, each presented through* $\mathsf{RegSeqRat}^{\uparrow}$ *rows and through the* $\mathsf{Real}^{\uparrow}$*-facing regular-name surface consumed by the completion interface;*
2. *μ is the explicit unary modulus row that assigns the common tail threshold used for every displayed precision request;*
3. *W is the common finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger inspected after the threshold supplied by μ;*
4. *T is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-comparison ledger for the inspected windows of X and Y, including the rational lower and upper comparison rows used by the tail estimate;*

5. *R is the tail-comparison readback row that records, for each inspected precision, that the two late regular-name windows are compared through the same W and the same dyadic endpoint ledger T;*

6. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

7. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *row.*

The carrier contains no quotient of regular streams, no host real carrier, no selected limit, and no equality test outside the rows displayed in the packet.

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonTasteGate single carrier alignment`

Definition 387.2 (CauchyTailComparison classifier)**.** *For accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carriers Q and Q', the classifier* $\mathsf{CauchyTailComparisonClassifier}(Q, Q')$ *compares only the visible rows X, Y, μ, W, T, R, P, and N by displayed* hsame *transport and displayed* Cont *read routes.*

In particular, two packets classify together only when their two regular-name surfaces are read through transported common-tail windows, transported dyadic endpoint-comparison rows, transported $\mathsf{Real}^{\uparrow}$ *classifier readback rows, the same package provenance, and the same local naming row. The classifier does not inspect quotient representatives, host limits, host real equality, or any completion principle not named by those rows.*

Theorem 387.3 (CauchyTailComparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission: an accepted packet is exactly the finite* BHist *carrier of Definition 387.1;*

2. *classifier stability: Definition 387.2 is preserved by displayed* hsame *transports and* Cont *routes on the two regular-name rows, the unary modulus row, the common window, the dyadic endpoint comparisons, the readback row, package provenance, and local naming row;*

3. *tail-window stability: every accepted precision request reads both regular Cauchy names through the same threshold supplied by μ and the same finite window ledger W;*

4. $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff: the readback row R may be consumed by the completion-facing* $\mathsf{Real}^{\uparrow}$ *classifier only through the displayed regular-name, common-window, dyadic endpoint, transport, continuation, package, and local naming rows;*

5. *non-escape: the packet exports no quotient of regular streams, no selected host limit, no host real equality, no choice principle, and no ambient completeness object.*

Proof. Carrier admission is projection of the displayed packet. Each coordinate is a finite history row or provenance row already named in Definition 387.1.

Classifier stability is componentwise. Transport the two regular-name surfaces, the modulus row, the common window, the endpoint ledger, and the readback row along the stored hsame comparisons, then replay the displayed Cont routes. Since Definition 387.2 reads no other data, accepted packets remain classified by the same finite rows.

The common-tail row is the pair (μ, W). A precision request first reaches μ, then reads both names inside the window ledger W at the threshold selected by that same row.

The completion-facing handoff is the row R. It packages the common-tail comparison in the format visible to the $\mathsf{Real}^{\uparrow}$ classifier used by $\mathsf{RealCauchyCompletion}^{\uparrow}$, while retaining the dyadic endpoint ledger and the provenance rows that justify the read.

Finally, the carrier has no coordinate for quotient stream equality, host real equality, selected limits, choice, or ambient completeness. Those objects therefore cannot be exported by the local certificate. □

Theorem 387.4 (CauchyTailComparison tail-window stability)**.** *For every accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier*

$$Q = (X, Y, \mu, W, T, R, P, N)$$

and every displayed precision request, any transported consumer read that preserves μ*,* W*,* T*, and* R *compares the two late regular-name windows through the same finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-comparison ledger. No transported read may replace the common tail threshold by an unrecorded selector or split the two names across different windows.*

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCommonWindowCarrier semantic name certificate`

Proof. Project the modulus coordinate μ. For the requested precision it supplies the only threshold row that the carrier exposes to a consumer.

Project the common window coordinate W. Both X and Y are read through that window after the threshold named by μ, so the two regular-name observations have a shared finite source.

Project the endpoint ledger T and readback row R. The comparison rows in R cite exactly the dyadic endpoint data in T for the same inspected window.

Transport can change the displayed representatives only along the stored hsame and Cont routes. Because the carrier has no second threshold coordinate and no second window coordinate, a transported read cannot introduce an unrecorded selector or split the comparison across different windows. □

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier tail stability`

Theorem 387.5 (CauchyTailComparison real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *classifier read exported by an accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name rows, the unary modulus* μ*, the common finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-comparison ledger* T*, the tail-comparison readback row* R*, the* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The handoff supplies comparison evidence usable by* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *without adding a quotient of streams or a host real equality test.*

Proof. A real-facing read first reaches the two regular-name rows X and Y. These rows are the only regular Cauchy surfaces stored in the carrier.

The read then passes through μ and W, so both names are inspected after the same displayed threshold and inside the same finite stream window. The endpoint comparisons are exactly the dyadic rows in T.

The readback row R repacks those common-tail observations as the comparison surface consumed by the $\mathsf{Real}^{\uparrow}$ classifier in the $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff. Package provenance P and local naming row N record the certificates that own the route.

Since this factorization exhausts the carrier coordinates, the handoff contains no quotient stream, host real value, host real equality test, selected limit, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier real completion handoff`

Theorem 387.6 (CauchyTailComparison common-window transitivity)**.** *Let* $Q_{XY} = (X, Y, \mu_{XY}, W_{XY}, T_{XY}, R_{XY}$ *and* $Q_{YZ} = (Y, Z, \mu_{YZ}, W_{YZ}, T_{YZ}, R_{YZ}, P_{YZ}, N_{YZ})$ *be accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carriers. Suppose the two packets compare* X *with* Y *and* Y *with* Z *through transported common modulus rows, shared finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers, and suppose a displayed* Cont *refinement composes the two common-window ledgers through the shared* Y *row. Then there is an accepted finite packet* Q_{XZ} *comparing* X *with* Z *through the composed common window, with endpoint ledger obtained by the displayed dyadic rows and with no quotient stream, selected host limit, host real equality, or ambient completeness coordinate.*

Proof. Project the two accepted carriers and their classifiers. Each packet exposes only two regular-name rows, one modulus row, one common window, one endpoint ledger, one readback row, package provenance, and a local naming row.

Apply tail-window stability from Definition 387.4 to Q_{XY} and to Q_{YZ}. Both comparisons are forced through their displayed common modulus rows and through the finite windows named in their carriers.

Compose the displayed window ledgers by the given Cont refinement along the shared Y row. The dyadic endpoint rows in T_{XY} and T_{YZ} then provide the endpoint ledger for the composed X to Z comparison, and the classifier of Definition 387.2 admits that transported packet because it reads only those visible coordinates.

Finally use the real-completion handoff of Definition 387.5. The resulting $\mathsf{Real}^{\uparrow}$ read remains a finite common-tail comparison surface, so composing the window ledgers does not introduce quotient equality, a selected limit, host equality, or completeness data. □

Theorem 387.7 (CauchyTailComparison non-escape boundary)**.** *An accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier exports only the finite comparison surface generated by* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. It exports no quotient of regular Cauchy streams, no choice-selected limit, no host real carrier, no host real equality, and no ambient completeness theorem.*

Proof. Inspect the carrier coordinates. The two regular-name rows, the modulus, the window ledger, the endpoint ledger, and the readback row are finite displayed rows. The remaining coordinates are package provenance and the local naming row.

The classifier of Definition 387.2 compares only those rows by hsame and Cont. It therefore cannot introduce an external equality relation, a quotient representative, or a host real comparison.

The $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff of Definition 387.5 keeps the same boundary. A downstream consumer receives finite common-tail comparison evidence, not an ambient completion object or a selected limit. □

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier non escape boundary`

Theorem 387.8 (CauchyTailComparison common tail-window refinement)**.** *Let*

$$Q = (X, Y, \mu, W, T, R, P, N)$$

be an accepted $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier. For any two consumer reads of the compared* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *surfaces that preserve the displayed tail-comparison classifier, there is a common refinement of their inspected tail windows through the same modulus row* μ*, the same finite* $\mathsf{StreamName}^{\uparrow}$ *ledger* W*, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger* T*.*

The refinement is internal to the carrier: both reads factor through the late windows already named by R*, and no consumer may replace them by an unrecorded selector, a quotient representative, or a host-level Cauchy comparison.*

Proof. Project the accepted packet. The only threshold coordinate is μ, and the only finite stream-window coordinate is W.

Apply the tail-window stability row of Definition 387.4 to each consumer read. Both reads must pass through the threshold supplied by μ and through the same window ledger W.

The endpoint comparisons are then read from T, and the comparison readback is R. Since the classifier has no second threshold, no second window ledger, and no host comparison coordinate, the two reads refine to the displayed common tail window. □

Theorem 387.9 (CauchyTailComparison finite-window induction)**.** *Let*

$$Q = (X, Y, \mu, W, T, R, P, N)$$

be an accepted $\mathsf{CauchyTailComparison}^{\uparrow}$ *carrier. Every finite list of displayed precision requests for the compared* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *surfaces is discharged by induction over the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W*. The empty case reads only* μ*,* W*,* T*,* R*,* P*, and* N *and exports no comparison datum. The step case consumes one displayed window of* W*, preserves the same common threshold* μ *and endpoint ledger* T*, and replays the* $\mathsf{Real}^{\uparrow}$ *seal through the existing real-completion handoff.*

Thus the common-tail comparison is generated by finite displayed windows. It does not appeal to quotient streams, host real equality, selected limits, choice, or ambient completeness.

Proof. Project the carrier of Definition 387.1. The only rows available to a precision request are X, Y, μ, W, T, R, P, and N, and the induction measure is the finite displayed window ledger W.

For the empty list of requests, inspect the rows μ, W, T, R, P, and N. No endpoint comparison or real-facing seal is exported, so the base case is only a finite carrier read.

For the step case, consume one displayed window from W and leave the remaining requests to the induction hypothesis. Apply Definition 387.4 to keep the request past the same threshold μ, and use Definition 387.13 to keep every endpoint read attached to the same ledger T.

Now apply Definition 387.5. The $\mathsf{Real}^{\uparrow}$ seal produced at the step is replayed through the displayed common-tail route, and the remaining finite requests inherit the same rows by the induction hypothesis.

Finally use Definition 387.7. The carrier has no coordinate for quotient streams, host real equality, selected limits, choice, or ambient completeness, so the induction never leaves the finite $\mathsf{StreamName}^{\uparrow}$ window surface. □

Theorem 387.10 (CauchyTailComparison RealUp seal stability)**.** *Let* $Q = (X, Y, \mu, W, T, R, P, N)$ *and* $Q' = (X', Y', \mu', W', T', R', P', N')$ *be accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carriers. If* CauchyTailComparisonClassif *holds, then every* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by* R *transports to the corresponding read exported by* R' *through the displayed* hsame *and* Cont *rows on* X*,* Y*,* μ*,* W*,* T*,* P*, and* N*.*

Thus the $\mathsf{Real}^{\uparrow}$ *comparison seal is stable under the displayed tail comparison. It cannot be strengthened into host real equality, quotient-stream equality, or a selected completed limit.*

Proof. Unfold the classifier of Definition 387.2. It transports the two regular-name rows, the common modulus, the stream window, the dyadic endpoint ledger, the readback row, package provenance, and local name row.

Use the real-completion handoff of Definition 387.5. A $\mathsf{Real}^{\uparrow}$ seal-facing read can only pass through X, Y, μ, W, T, R, P, and N.

Transport each of these rows along the displayed classifier data. The target read is exactly the readback row R' in the target packet, and the non-escape boundary of Definition 387.7 prevents any host equality or selected limit from entering the seal. □

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier real seal stability`

Theorem 387.11 (CauchyTailComparison NameCert obligation package)**.** *The* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *obligation package is the five-row package consisting of:*

1. *the carrier row of Definition 387.1;*
2. *the classifier row of Definition 387.2;*
3. *the tail-window stability, common-window transitivity, and common-window refinement rows of Definition 387.4, Definition 387.6, and Definition 387.8;*
4. *the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal-stability rows of Definition 387.5 and Definition 387.10;*
5. *the non-escape boundary of Definition 387.7.*

Every consumer of this package receives only the finite rows X*,* Y*,* μ*,* W*,* T*,* R*,* P*, and* N*, together with their displayed* hsame *and* Cont *routes.*

Proof. The carrier and classifier rows give the finite packet and its visible transport relation. These rows fix the only data a consumer may inspect.

The tail-window rows identify the common modulus and stream-window ledger used by both compared regular-name reads. The common refinement theorem repacks the same information for paired $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

The handoff and seal-stability rows route comparison evidence into the real-completion-facing interface. The non-escape boundary then states that the package exports no quotient stream, selected limit, host real equality, or ambient completeness object. □

Theorem 387.12 (CauchyTailComparison classifier transport)**.** *Let* $Q = (X, Y, \mu, W, T, R, P, N)$ *and* $Q' = (X', Y', \mu', W', T', R', P', N')$ *be accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *carriers. If* $\mathsf{CauchyTailComparisonClassif}$ *holds, then classifier transport preserves exactly the two regular-name rows* X, Y*, the unary modulus* μ*, the common window* W*, the dyadic endpoint ledger* T*, the readback row* R*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *row* N*.*

Proof. Project the carrier rows by Definition 387.1. The displayed tuple has exactly the rows X, Y, μ, W, T, R, P, N.

Apply the classifier definition of Definition 387.2. Its comparison is componentwise by displayed hsame transport and displayed Cont read routes for the same rows.

Use Definition 387.4 and Definition 387.5 to identify the consumer-facing uses of the transported rows. The classifier therefore moves the finite common-tail packet and nothing outside it. □

Theorem 387.13 (CauchyTailComparison dyadic endpoint exactness)**.** *Every endpoint comparison exported by an accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *packet is read from the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* T *after the common threshold* μ *and common window* W *have been selected. The comparison cannot replace* T *by an unrecorded dyadic ledger, quotient-stream equality, selected limit, or host real comparison.*

Proof. Start with the carrier tuple of Definition 387.1. The only endpoint-comparison coordinate is T, and it is reached after the modulus row μ and common window W.

Apply Definition 387.4. Any precision read of the two regular names passes through the same threshold and same finite window.

Then use Definition 387.8. The common refinement keeps the endpoint rows attached to the displayed dyadic ledger, so an endpoint comparison has no route to an unrecorded ledger or host-level comparison. □

Theorem 387.14 (CauchyTailComparison scoped export)**.** *The scoped consumer surface of* $\mathsf{CauchyTailComparison}^{\uparrow}$ *is the finite package consisting of tail-window stability, dyadic endpoint exactness,* $\mathsf{Real}^{\uparrow}$ *seal stability, and the non-escape boundary over the displayed rows* X, Y, μ, W, T, R, P, N*. Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the package receives that surface and no quotient stream, selected limit, host real equality, or ambient completeness object.*

Proof. First use Definition 387.4. It fixes the common threshold and window through which both regular-name rows are read.

Next apply Definition 387.13. The endpoint comparisons are exactly the dyadic rows in T after that common tail window.

Use Definition 387.10 for the real-facing boundary. Seal reads transport only along the displayed rows, and Definition 387.7 excludes quotient streams, selected limits, host equality, and ambient completeness from the export. □

Theorem 387.15 (CauchyTailComparison window-composition seal exhaustion)**.** *Let*

$$Q_{XY} = (X, Y, \mu_{XY}, W_{XY}, T_{XY}, R_{XY}, P_{XY}, N_{XY})$$

and

$$Q_{YZ} = (Y, Z, \mu_{YZ}, W_{YZ}, T_{YZ}, R_{YZ}, P_{YZ}, N_{YZ})$$

be accepted $\mathsf{CauchyTailComparison}^{\uparrow}$ *packets composed along the shared regular-name row* Y*. Suppose their common-window ledgers are related by a displayed* Cont *route that composes the inspected windows and endpoint ledgers into rows*

$$\mu, W, T, R, P, N$$

for the comparison from X *to* Z*.*

Then every resulting $\mathsf{Real}^{\uparrow}$ *seal read factors through the composed rows* X*,* Z*,* μ*,* W*,* T*,* R*,* P*, and* N*. No quotient stream, host real equality, selected limit, or ambient completeness coordinate appears in the composed seal surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier window composition seal exhaustion`

Proof. Project the two accepted carriers. The first packet exposes only X, Y, μ_{XY}, W_{XY}, T_{XY}, R_{XY}, P_{XY}, and N_{XY}; the second exposes only Y, Z, μ_{YZ}, W_{YZ}, T_{YZ}, R_{YZ}, P_{YZ}, and N_{YZ}.

Apply Definition 387.4 to both packets. Each accepted precision read passes through its displayed threshold and finite common window, so the shared row Y is read through visible late-window data on both sides.

Use Definition 387.8 with the displayed Cont route. The common-window refinement supplies the composed rows μ, W, T, and R for the X to Z comparison without introducing a second threshold or an unrecorded endpoint ledger.

Now apply Definition 387.5 and Definition 387.10. The $\mathsf{Real}^{\uparrow}$ seal read is routed through the composed regular-name endpoints, common window, dyadic endpoint ledger, readback row, package provenance, and local naming row.

Finally use Definition 387.7. The composed carrier inventory has no coordinate for quotient streams, host real equality, selected limits, or ambient completeness, so the seal read is exhausted by X, Z, μ, W, T, R, P, and N. □

Theorem 387.16 (CauchyTailComparison standard bridge source)**.** *For an accepted* $\mathsf{CauchyTailComparison}^{\uparrow}$ *packet, the bridge-facing source for the standard regular-Cauchy completion interface is exactly the pair of* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name rows, the unary modulus row, the common finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger, the tail-comparison readback row, the displayed* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *row. Every standard completion-interface read factors through Definition 387.15 and the public export of Definition 387.14, with no quotient stream, host real equality, selected limit, or ambient completeness coordinate.*

Proof. Project the accepted carrier first. Its visible rows are the two regular names, the unary modulus, the finite common window, the dyadic endpoint ledger, the readback row, package provenance, and the local naming row.

Apply Definition 387.15. It shows that composed completion reads are exhausted by the displayed common-window and endpoint rows, so the standard interface cannot substitute an unrecorded stream equality or second endpoint ledger.

Next use Definition 387.5. The handoff is the public regular-Cauchy completion route generated from the same carrier rows, and Definition 387.10 keeps the real-facing seal attached to that route.

Finally apply Definition 387.7. The carrier has no field for quotient streams, host real equality, selected limits, or ambient completeness, so every standard completion-interface read remains a read of the displayed bridge source. □

Lean4 ✓ `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier standard bridge source`

Theorem 387.17 (CauchyTailComparison completion-classifier equivalence)**.** *Let*

$$Q = (X, Y, \mu, W, T, R, P, N)$$

and

$$Q' = (X', Y', \mu', W', T', R', P', N')$$

be accepted $\mathsf{CauchyTailComparison}^{\uparrow}$ *packets. The completion-facing* $\mathsf{Real}^{\uparrow}$ *classifier reads exported by* Q *and* Q' *determine the same finite BEDC route exactly when the following displayed rows agree componentwise: the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name rows, the common modulus rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *common-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers, the tail-comparison readback rows, the* Pkg *provenance rows, and the local* $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ *rows.*

Equivalently, equality of the completion-facing classifier route is neither coarser nor finer than agreement of

$$(X, Y, \mu, W, T, R, P, N) \quad \textit{and} \quad (X', Y', \mu', W', T', R', P', N')$$

on those displayed coordinates. The equivalence introduces no quotient stream, selected host limit, host real equality, choice row, or ambient completeness datum.

Proof. For the forward direction, project the common $\mathsf{Real}^{\uparrow}$ classifier read through Definition 387.5. The handoff exposes the completion-facing read only through the two regular-name rows, the modulus row, the common finite window, the dyadic endpoint ledger, the readback row, package provenance, and the local naming row.

Apply Definition 387.16. The standard bridge source identifies those exposed rows as the entire finite source of a regular-Cauchy completion-interface read, so equality of the classifier route projects back to componentwise agreement of the displayed packet coordinates.

For the reverse direction, assume componentwise agreement of the displayed rows. Transport the classifier along the agreed Pkg provenance and local $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$ rows, then use Definition 387.13 to keep the endpoint ledger exact.

Use Definition 387.4 and Definition 387.6 to keep every late precision read inside the common finite window. Then apply Definition 387.15; the transported rows rebuild the same completion-facing $\mathsf{Real}^{\uparrow}$ classifier route and leave no remaining coordinate to vary.

Finally apply Definition 387.7. Since neither packet carries a quotient stream, selected host limit, host real equality, choice row, or ambient completeness datum, the equivalence is exhausted by the displayed finite BEDC rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailComparison}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ regular-name rows, an explicit unary modulus, a common finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint comparisons, tail-comparison readback rows, $\mathsf{Real}^{\uparrow}$ classifier-facing handoff data, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 387.15 and Definition 387.16 bind the mature completion-interface consumer surface to the public common-tail comparison export over the displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailComparison}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyTailComparisonUp.CauchyTailComparisonCarrier\_standard\_bridge\_source`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of regular streams, uniqueness of limits, host real construction, ambient completeness, or bridge checking.

Upgrade path: No further paper-axis theory upgrade remains; formalization and bridge checking remain over the same displayed rows.

This hub is the finite handoff from chapter 129 and chapter 130 into the $\mathsf{CompactUniformContinuity}^{\uparrow}$ packet: $\mathsf{CompactMetric}^{\uparrow}$ supplies finite ProbeBundle net rows, while $\mathsf{ContinuousMap}^{\uparrow}$ supplies pointwise modulus rows.

Chapter 388

A Concrete Naming Certificate for $\mathsf{CompactUniformContinuity}^{\uparrow}$

The $\mathsf{CompactUniformContinuity}^{\uparrow}$ interface names the concrete certificate layer that realizes Definition 9237.6 as finite BEDC data. It sits over the same source metric as the compact and continuous-map packets: compactness contributes finite ProbeBundle nets from $\mathsf{CompactMetric}^{\uparrow}$, continuity contributes pointwise modulus rows from $\mathsf{ContinuousMap}^{\uparrow}$, the source and target metrics contribute MetricDistanceWitness rows, and the final precision is selected by positive-radius folds through $\mathsf{Rat}^{\uparrow}$. The packet exports a NameCert surface for the compact-uniform-continuity handoff; it does not expose an open-cover principle, sequential compactness, a host modulus function, or a selected real-valued minimum.

This packet is a downstream consumer of chapter 129 for finite ProbeBundle nets and of chapter 130 for pointwise modulus rows. The carrier below records those sibling dependencies as visible compact-net and continuity-ledger coordinates rather than as ambient analytic principles.

Definition 388.1 (CompactUniformContinuity carrier)**.** *A* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose distance reads are witnessed by* MetricDistanceWitness*;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance reads are witnessed by* MetricDistanceWitness*;*
3. *F is the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from X to Y;*
4. *E is a positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history requested by the target metric consumer;*
5. *D is the positive* $\mathsf{Rat}^{\uparrow}$ *precision history proposed as the compact-uniform output;*
6. *$\mathcal{P}$ is the nonempty finite* ProbeBundle *net supplied by the* $\mathsf{CompactMetric}^{\uparrow}$ *packet over X;*
7. *C records the compact finite-net coverage rows, including center histories and half-radius coverage witnesses for every source-point history;*
8. *M records the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows at the bundled centers, each read through the graph row F and the source and target metric ledgers;*
9. *R records the positive-radius* $\mathsf{Rat}^{\uparrow}$ *rows and comparison rows used by the fold, including the half-radius histories consumed by the finite cover;*

10. *W is the finite lower-bound fold witness proving that D is classifier-stably below every bundled half-radius row;*
11. *T records the componentwise* hsame *rows for source metrics, target metrics, graph rows, tolerance rows, precision rows, finite-net rows, pointwise-modulus rows, radius rows, and fold rows;*
12. *Q records the displayed* Cont *routes by which compact, continuous-map, metric, rational-fold, and compact-uniform-continuity consumers read the packet;*
13. *P is the* Pkg *provenance row over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and the closed* ProbeBundle *spine;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *row.*

The carrier contains only the displayed finite-net, pointwise-modulus, metric-distance, positive-rational, transport, continuation, package, and naming rows. It contains no hidden point selector, no quotient of source histories, no ambient minimum operator, and no analytic compactness theorem beyond the finite packet.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket carrier bhist rows`

Definition 388.2 (CompactUniformContinuity classifier)**.** *For accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carriers H and H', the classifier* $\mathsf{CompactUniformContinuityClassifier}(H, H')$ *compares exactly the visible rows of Definition 388.1. Source and target metric rows compare through their displayed* hsame *and* Cont *routes; compact finite-net rows compare through the same* ProbeBundle *spine and coverage witnesses; continuous-map rows compare through the pointwise modulus ledger; positive tolerance, precision, radius, and fold rows compare through the* $\mathsf{Rat}^{\uparrow}$ *classifier; and the package and local naming rows compare through their displayed* Pkg *and* NameCert *provenance.*

The classifier does not compare external functions, open covers, sequences, host real endpoints, selected minima, or unrecorded continuity data.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket classifier symmetric`

Theorem 388.3 (CompactUniformContinuity NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *obligation surface consists of carrier habitation, displayed classifier transport, compact finite-net coverage over* $\mathsf{CompactMetric}^{\uparrow}$ *and* ProbeBundle*, continuous pointwise-modulus handoff over* $\mathsf{ContinuousMap}^{\uparrow}$*, metric-distance routing through* MetricDistanceWitness*, positive-radius lower-bound folding over* $\mathsf{Rat}^{\uparrow}$*, and downstream non-escape through the recorded* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket namecert obligations`

Proof. Carrier habitation is the finite packet of Definition 388.1. Every row named by the compact-uniform certificate has a visible coordinate before any consumer can read it.

Classifier transport is componentwise by Definition 388.2. The compact rows transport finite-net coverage, the continuous rows transport pointwise modulus data, the metric rows transport distance witnesses, and the rational rows transport positive-radius comparisons and the lower-bound fold.

The obligation surface is therefore exactly the displayed handoff from compact finite nets and pointwise continuity to a uniform precision history. Repacking those rows with T, Q, P, and N gives the local $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ surface, while external selectors, sequence principles, open-cover compactness, and hidden minima remain outside the carrier. □

Theorem 388.4 (CompactUniformContinuity root uniform-modulus row)**.** *Every accepted* CompactUniformContinu *packet routes the requested tolerance row through the displayed* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, MetricDistanceWitness, *positive* Rat$^{\uparrow}$ *radius rows,* hsame, Cont, Pkg, *and local* NameCert *coordinates before exporting a uniform precision. The exported precision is therefore the visible uniform-modulus row of the packet, not a host modulus function or a selected real minimum.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root uniform modulus row`

Proof. Project the accepted packet through the carrier of Definition 388.1. The requested tolerance row is the displayed E coordinate, and the proposed uniform precision is the displayed D coordinate.

Apply Definition 388.3. It identifies the local route through compact finite-net coverage, pointwise ContinuousMap$^{\uparrow}$ modulus rows, metric-distance witnesses, positive Rat$^{\uparrow}$ comparisons, and the recorded transport, continuation, package, and naming rows.

The classifier of Definition 388.2 preserves exactly those coordinates. Thus a downstream consumer sees a uniform precision only after the finite route has passed through the named BEDC packet surface. □

Theorem 388.5 (CompactUniformContinuity root compact-net row)**.** *For every accepted* CompactUniformContinui *packet, the finite* ProbeBundle *net and its center-radius ledger are the only compactness data consumed by the compact-uniform-continuity handoff. No open-cover principle, sequential compactness principle, hidden point selector, or ambient compactness oracle is available to the handoff.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root compact net row`

Proof. Project the packet by Definition 388.1. The compactness coordinates are the finite ProbeBundle spine $\mathcal{P}$ and the coverage ledger C, with the related radius rows read through R and W.

Apply Definition 388.13. Every handoff read first chooses a bundled center and then follows the displayed center-radius and metric-distance rows.

Finally use Definition 388.14 to identify the precision consumed by the handoff with the finite positive-rational fold. Since the carrier has no coordinate for any other compactness witness, the finite net and center-radius ledger exhaust the compact data surface. □

Theorem 388.6 (CompactUniformContinuity root NameCert unblock package)**.** *The* CompactUniformContinuity$^{\uparrow}$ *root package consists of the following five rows:*

1. *the carrier and classifier rows of Definition 388.1 and Definition 388.2;*
2. *the finite-net row of Definition 388.5;*
3. *the uniform-modulus row of Definition 388.4;*
4. *the metric-distance routing and rational-fold rows recorded by Definition 388.3 and Definition 388.14;*
5. *the non-escape ledger and downstream package surface that feed Definition 388.13.*

These five rows are sufficient for a downstream consumer to use the compact-uniform-continuity certificate as a finite $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *package.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root namecert unblock package`

Proof. The first row is obtained by projecting the displayed packet carrier and its componentwise classifier. This gives the full finite coordinate list and the allowed transport rules.

The second row is Definition 388.5. It isolates the finite ProbeBundle and center-radius ledger as the compactness surface.

The third row is Definition 388.4. It identifies the route from a requested tolerance history to the exported uniform precision.

The fourth row is the metric and rational bookkeeping recorded by Definition 388.3 and sharpened by Definition 388.14. Those statements keep metric-distance witnesses and positive-rational folds inside the carrier.

The final row is obtained by feeding the preceding rows into Definition 388.13. The handoff exports exactly the downstream package surface and leaves no coordinate for external compactness, selected minima, or host metric inspection. □

Theorem 388.7 (CompactUniformContinuity source metric totality)**.** *In every accepted* CompactUniformContinuity *root packet, each source-center distance read used by the finite-net handoff factors through the source* $\mathsf{Met}^{\uparrow}$ *coordinate* X*, a displayed* MetricDistanceWitness *row, and the transport, continuation, package, and naming provenance* T, Q, P, N *before any pointwise modulus row is consumed. Thus the handoff has no route from a source point to a* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row except through the recorded source metric surface of the carrier.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket source metric totality`

Proof. Project the root packet by Definition 388.1. The source metric row X, the compact finite-net ledger $\mathcal{P}, C$, the pointwise-modulus ledger M, and the provenance rows T, Q, P, N are all visible coordinates before any consumer can read a source-center distance.

Use the classifier of Definition 388.2. It permits source metric transport only through the displayed hsame rows, the Cont routes, the package row, and the local naming row.

Now apply Definition 388.5 and Definition 388.6. The finite-net row chooses bundled centers only inside the carrier, and the root package lists the metric-distance route as part of the same NameCert surface. Therefore each accepted source-center read reaches M only after passing through X, MetricDistanceWitness, and T, Q, P, N. □

Theorem 388.8 (CompactUniformContinuity target metric totality)**.** *In every accepted* CompactUniformContinuity *root packet, each target-output tolerance read factors through the target* $\mathsf{Met}^{\uparrow}$ *coordinate* Y*, a displayed* MetricDistanceWitness *row, the centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus ledger* M*, and the same* T, Q, P, N *provenance that already controls source-center metric reads. No target tolerance row is available directly from a host metric outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket target metric totality`

Proof. Begin with the target tolerance route isolated by Definition 388.4. It sends the requested tolerance through compact finite-net coverage, centerwise $\mathsf{ContinuousMap}^{\uparrow}$ rows, target metric evidence, positive $\mathsf{Rat}^{\uparrow}$ comparisons, and the recorded transport, continuation, package, and naming rows.

Apply Definition 388.7. The source side of each centerwise modulus read has already been forced through X, MetricDistanceWitness, and T, Q, P, N before M is consumed.

The target side of the same centerwise row is the displayed $\mathsf{Met}^{\uparrow}$ coordinate Y. The classifier transports it only through the target MetricDistanceWitness row and the same T, Q, P, N provenance, so a target-output tolerance read cannot bypass the finite packet surface. □

Theorem 388.9 (CompactUniformContinuity fold witness stability)**.** *In every accepted* CompactUniformContinuity *root packet, the fold witness W transports the positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold below each bundled half-radius row under the classifier, and W is the only route by which the precision row D is accepted as a common lower bound for the finite handoff. The classifier therefore preserves the fold surface without introducing a host minimum, a selected real radius, or an unrecorded precision source.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket fold witness stability`

Proof. Project the rational ledger R, the fold witness W, and the precision row D from the carrier. By Definition 388.14, D is accepted exactly as the finite lower-bound fold over the positive $\mathsf{Rat}^{\uparrow}$ rows, with a displayed comparison below each bundled half-radius row.

Use Definition 388.11. It identifies R and W as the only rational route from the root package to the precision row D.

Finally apply Definition 388.8. The target-output reads that consume D already pass through Y, the centerwise modulus row M, and T, Q, P, N. Transporting W under the classifier therefore preserves the same positive $\mathsf{Rat}^{\uparrow}$ fold for every downstream target read, with no additional coordinate from which another lower bound could be read. □

Theorem 388.10 (CompactUniformContinuity root metric route)**.** *In every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *root packet, each source-center distance read and each target-output distance read factors through displayed* MetricDistanceWitness *rows, the transported* hsame *rows, the* Cont *handoff rows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *row. No distance used by the root package is read directly from a host metric outside those coordinates.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root metric route`

Proof. Project the packet through the carrier of Definition 388.1. The source and target metric coordinates are visible before the compact rows, continuous rows, rational rows, package provenance, and local naming row are consumed.

Apply the classifier of Definition 388.2. It transports source and target metric observations only through the displayed hsame rows and the named MetricDistanceWitness evidence.

Use Definition 388.4 for the route from a requested tolerance to a uniform precision, and use Definition 388.5 for the bundled center chosen from the finite net. The two rows meet at the same Cont, Pkg, and local NameCert surface, so the source-center and target-output reads remain inside the root packet. □

Theorem 388.11 (CompactUniformContinuity root rational fold route)**.** *In every accepted* CompactUniformContin *root packet $H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N)$, the precision row D is read only through the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger R and the finite lower-bound fold witness W. The root package has no host minimum coordinate and no selected real-radius operation.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root rational fold route`

Proof. Read the finite compact-net row by Definition 388.5. Its centers are the entries of $\mathcal{P}$, and their usable radii are the displayed positive rows in R.

Read the uniform-modulus row by Definition 388.4. The exported precision is the carrier coordinate D, not a host function chosen after the packet is built.

The witness W records the finite lower-bound fold over the positive $\mathsf{Rat}^{\uparrow}$ rows in R. Since the carrier coordinates list R, W, and D but no host minimum or real-radius selector, every precision read in the root package follows that rational fold route. □

Theorem 388.12 (CompactUniformContinuity root non-escape package)**.** *The* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *root package is the five-row package consisting of the carrier/classifier row, the compact-net row, the uniform-modulus row, the metric-route row, and the rational-fold row. This package excludes open covers, sequential compactness, host minima, and external modulus functions from every downstream root read.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root nonescape package`

Proof. The carrier/classifier row is the first row of Definition 388.6. It fixes the visible coordinates and the allowed componentwise transport.

The compact-net row is Definition 388.5, and the uniform-modulus row is Definition 388.4. Together they identify the finite ProbeBundle coverage surface and the exported precision surface.

The metric-route row is Definition 388.10. It keeps source-center and target-output distance reads behind displayed metric witnesses, hsame, Cont, Pkg, and local naming coordinates.

The rational-fold row is Definition 388.11. It keeps the precision row behind positive $\mathsf{Rat}^{\uparrow}$ radius rows and the finite fold witness. These five rows are exactly the root package surface, so there is no coordinate from which a downstream consumer could read an open cover, a sequence compactness principle, a host minimum, or an external modulus function. □

Theorem 388.13 (CompactUniformContinuity finite-net handoff)**.** *Suppose an accepted* CompactUniformContinuit *carrier has tolerance history E, precision history D, compact finite-net spine $\mathcal{P}$, coverage rows C, pointwise-modulus rows M, positive-radius rows R, and lower-bound fold witness W. Then every source-distance read below D factors through one bundled center in $\mathcal{P}$, the corresponding* MetricDistanceWitness *rows in X, the corresponding pointwise-modulus row in M, and the target metric rows in Y. The result is the uniform-modulus ledger promised by Definition 9237.6, read as finite* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *data.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket finite net handoff`

Proof. Read the compact coordinate first. The $\mathsf{CompactMetric}^{\uparrow}$ row supplies the nonempty ProbeBundle, center histories, half-radius histories, and coverage rows saying that each source-point history is covered by some bundled center.

Read the rational fold next. The witness W records a primitive finite fold over the positive-radius rows R, so the selected precision history D compares below every bundled half-radius through the $\mathsf{Rat}^{\uparrow}$ classifier and the transported hsame rows.

Given two source histories whose displayed distance is below D, the coverage row chooses the bundled center for the first history. The fold comparison and the source metric triangle row route the second history into the same local ball, with all distances witnessed by MetricDistanceWitness.

Finally use the $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus row attached to that bundled center and the target metric triangle row. Both source histories are inside the same displayed pointwise ball,

so the two target histories are within the requested tolerance row E. All reads pass through the carrier coordinates, hence the handoff is a $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ row rather than an external Heine–Cantor theorem. □

Theorem 388.14 (CompactUniformContinuity radius-fold exactness)**.** *In every accepted* CompactUniformContinuit *carrier* $H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N)$*, the precision row* D *is accepted by the classifier exactly as the finite lower-bound fold over the positive-radius ledger* R*. Every compact finite-net center in* $\mathcal{P}$ *receives a displayed half-radius comparison below* D*, and classifier transport through* T*,* Q*,* P*, and* N *preserves that fold surface.*

Consequently the uniform precision consumed by the finite-net handoff is the displayed rational fold output, not a hidden point selector, selected real minimum, open-cover compactness principle, sequential compactness principle, or external modulus function.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket radius fold exactness`

Proof. Project the positive-radius ledger R and the fold witness W from the carrier. The witness records a finite $\mathsf{Rat}^{\uparrow}$ lower-bound fold whose output is the displayed precision row D.

For each center in the finite ProbeBundle spine $\mathcal{P}$, the same ledger supplies the positive radius used by the pointwise-modulus row, and W supplies the comparison placing D below the corresponding half-radius history.

The classifier is componentwise on T, routed by Q, and packaged by P and N. Thus transport can carry the $\mathsf{Rat}^{\uparrow}$ comparisons and the fold row across classified carriers, but it cannot introduce a new minimum operation or an unrecorded radius source.

Feeding these transported comparisons into Definition 388.13 gives the same uniform-modulus ledger for every accepted representative. The handoff reads finite rational fold data rather than external compactness or host real selection data. □

Theorem 388.15 (CompactUniformContinuity real-seal boundary)**.** *When the source and target metrics are consumed through* $\mathsf{Real}^{\uparrow}$ *or interval-shaped* $\mathsf{Real}^{\uparrow}$ *packets, an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carrier can expose only the finite-net coverage rows,* MetricDistanceWitness *rows, pointwise-modulus rows, positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, and the resulting uniform precision history. It cannot expose host real equality, a selected real minimum, sequential compactness, open-cover compactness, or an external modulus function at the real-seal boundary.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket real seal boundary`

Proof. Project the carrier through its metric coordinates X and Y. A real-valued consumer reaches those coordinates only through displayed metric rows and their MetricDistanceWitness evidence.

The compact information available at the boundary is the finite ProbeBundle ledger C. The continuous information available there is the pointwise-modulus ledger M. The positive lower bound available there is the $\mathsf{Rat}^{\uparrow}$ fold row W over the radius ledger R.

Since these rows exhaust the carrier coordinates exposed to real-name consumers, the seal can export only the resulting finite uniform-precision packet. Host real equality, selected minima, sequence arguments, open covers, and external modulus functions have no coordinate from which they could be read. □

Theorem 388.16 (CompactUniformContinuity ProbeBundle coverage)**.** *In every accepted* CompactUniformContinu *carrier* $H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N)$*, every source-point read used by the compact-uniform-continuity handoff factors through the finite* ProbeBundle *spine* $\mathcal{P}$*, one displayed center*

history in C, the associated half-radius coverage row, and the same transported hsame, Cont, Pkg, *and* NameCert *rows that classify H.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket probe bundle coverage`

Proof. Project the carrier by Definition 388.1. The finite net coordinate $\mathcal{P}$ and the coverage ledger C are visible before the continuous-map and rational-fold rows are consumed.

The classifier of Definition 388.2 transports these rows componentwise. Thus a classified replacement of the carrier preserves the same bundled center histories, the same half-radius coverage witnesses, and the same continuation routes by which a source-point read reaches them.

The local obligation statement in Definition 388.3 identifies this coverage as part of the $\mathsf{NameCert}_{\mathsf{CompactUniformCon}}$ surface. Feeding the selected bundled center into Definition 388.13 gives the uniform-precision route, and Definition 388.15 confirms that no open-cover or sequence principle is exported in place of the finite ProbeBundle ledger. □

Theorem 388.17 (CompactUniformContinuity metric-distance routing)**.** *For every accepted* CompactUniformCont *carrier, each source metric read used to enter a bundled center ball and each target metric read used to compare output histories is routed through a displayed* MetricDistanceWitness *row in the source or target metric coordinate. No source or target distance is read directly from a host metric function outside the carrier.*

Proof. The carrier of Definition 388.1 names the source metric row X and target metric row Y as the only metric coordinates. Both are read through displayed MetricDistanceWitness evidence and transported with the componentwise rows T and routes Q.

By Definition 388.2, classified carriers compare metric rows only through those displayed witnesses. Hence a source-distance comparison below D, a half-radius comparison around a bundled center, and a target-distance comparison below E all remain inside the same metric-distance ledgers.

The finite-net route of Definition 388.13 uses exactly these source and target reads, while Definition 388.3 records metric-distance routing as a local naming obligation. The boundary statement in Definition 388.15 then prevents a real-name consumer from replacing those witnesses by host metric access. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket metric distance routing`

Theorem 388.18 (CompactUniformContinuity non-escape ledger)**.** *An accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet exports only the finite* ProbeBundle *coverage rows,* MetricDistanceWitness *rows, pointwise-modulus rows, positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, displayed transports, continuation routes, package provenance, and local naming provenance. It exports no open-cover compactness principle, no sequential compactness principle, no host minimum, and no external modulus function.*

Proof. Inspect the carrier in Definition 388.1. Its coordinates list the compact finite-net data, continuous pointwise-modulus rows, source and target metric witnesses, rational-radius fold data, transports, routes, package row, and local certificate row.

The classifier in Definition 388.2 compares exactly those coordinates. It has no component for open covers, sequences, host minima, or unrecorded modulus functions, so those objects cannot be introduced by classifier transport.

The obligation surface of Definition 388.3 and the handoff of Definition 388.13 use the same finite rows to produce the uniform precision history. Finally, Definition 388.15 states the real-seal boundary for these rows, so the exported packet remains a finite BEDC ledger and not a classical Heine–Cantor artifact. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket non escape ledger`

Theorem 388.19 (CompactUniformContinuity modulus choice surface)**.** *For every accepted* CompactUniformConti *packet, the uniform precision row consumed by the compact-uniform-continuity handoff is determined only by the finite* ProbeBundle *centers, the* ContinuousMap$^{\uparrow}$ *pointwise-modulus rows attached to those centers, the positive* Rat$^{\uparrow}$ *radius ledger, and the finite lower-bound fold witness. It is not determined by an open cover, a sequential compactness argument, a selected real minimum, or a host modulus function.*

Proof. Project the packet through the carrier in Definition 388.1. The only coordinates from which a precision row can be read are the finite ProbeBundle spine, the centerwise modulus ledger, the positive-radius ledger, and the fold witness.

The classifier in Definition 388.2 preserves those rows componentwise. The finite-net handoff of Definition 388.13 and the radius-fold exactness statement of Definition 388.14 then identify the consumed precision row with the displayed lower-bound fold output.

Finally, Definition 388.16, Definition 388.17, and Definition 388.18 rule out any route through unrecorded compactness, host minima, or external modulus selection. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket modulus choice surface`

Theorem 388.20 (CompactUniformContinuity center-radius compatibility)**.** *In every accepted* CompactUniformContinuity$^{\uparrow}$ *carrier, each bundled center in the finite* ProbeBundle *spine has a compatible half-radius row in the positive* Rat$^{\uparrow}$ *ledger. The source metric reads that place histories inside that half-radius ball and the target metric reads that consume the corresponding pointwise modulus are mediated by displayed* MetricDistanceWitness *rows.*

Proof. Read the finite-net coordinate and coverage ledger from Definition 388.1. Each center history appears with the half-radius coverage row supplied by the compact packet and with the positive-radius data recorded in R.

The radius-fold exactness theorem Definition 388.14 supplies the comparison between the displayed precision row and every bundled half-radius. The ProbeBundle coverage theorem Definition 388.16 keeps the same center and half-radius rows under classifier transport.

Metric access is then fixed by Definition 388.17. Source reads into the center ball and target reads from the pointwise-modulus conclusion both pass through MetricDistanceWitness rows, and the non-escape ledger of Definition 388.18 leaves no direct host-metric read outside those witnesses. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket center radius compatibility`

Theorem 388.21 (CompactUniformContinuity uniform precision consumer route)**.** *For an accepted* CompactUniformContinuity$^{\uparrow}$ *packet, any consumer read below the displayed precision factors through compact finite coverage, one bundled center row, the center's* ContinuousMap$^{\uparrow}$ *modulus*

row, source and target MetricDistanceWitness *rows, the transported* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *row. The route contains no open-cover compactness, sequential compactness, selected real minimum, or host modulus.*

Proof. Begin with the carrier and classifier. By Definition 388.1 the consumer can read only the displayed compact, continuous-map, metric, rational, transport, route, package, and naming coordinates, and by Definition 388.2 those coordinates are the complete comparison surface.

The finite-net handoff in Definition 388.13 routes a read below the displayed precision through a covered center. The radius-fold exactness statement Definition 388.14 supplies the half-radius comparison needed to enter the center row, and Definition 388.20 identifies the compatible metric witnesses used there.

At the center, the pointwise-modulus row from $\mathsf{ContinuousMap}^{\uparrow}$ produces the target tolerance read. The ProbeBundle coverage theorem, metric-distance routing theorem, modulus choice surface, and non-escape ledger then show that the whole consumer route is carried by hsame, Cont, Pkg, and local NameCert provenance rather than by open covers, sequences, host minima, or an external modulus. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket uniform precision consumer route`

Theorem 388.22 (CompactUniformContinuity modulus certificate)**.** *For an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet, the uniform-modulus certificate is the finite packet obtained from the compact finite-net rows* $\mathcal{P}$ *and* C*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows* M*, the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger* R*, and the lower-bound fold witness* W*. The certificate records a precision history* D *whose consumer route is mediated by* MetricDistanceWitness *rows and displayed* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance, with no appeal to open covers, selected subcovers, or a host modulus function.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket modulus certificate`

Proof. Project the carrier of Definition 388.1. The compact coordinates provide the finite ProbeBundle spine and coverage rows, while the continuous-map coordinate provides pointwise-modulus rows attached to those bundled centers.

Use the radius-fold exactness theorem Definition 388.14. It identifies D as the positive rational precision produced by folding the displayed radius ledger, and it compares that precision with the center radii read by the pointwise-modulus rows.

The finite-net handoff of Definition 388.13 and the uniform precision consumer route of Definition 388.21 then turn the displayed centerwise data into the requested uniform-modulus certificate. The non-escape ledger excludes any unrecorded compactness or host-level modulus coordinate. □

Theorem 388.23 (CompactUniformContinuity finite-net factorization)**.** *Every consumer read of the finite-net certificate for* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *factors through* $\mathsf{CompactMetric}^{\uparrow}$ *finite cover data,* $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{Rat}^{\uparrow}$ *dyadic or rational precision requests, and* $\mathsf{Real}^{\uparrow}$ *seal reads. The factorization is witnessed inside the displayed carrier rows* $\mathcal{P}$*,* C*,* M*,* R*,* W*,* T*,* Q*,* P*, and* N*.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket finite net factorization`

Proof. Begin with the compact row. By Definition 388.16, every source history admitted to the packet reaches a bundled center through the finite ProbeBundle coverage data supplied by $\mathsf{CompactMetric}^{\uparrow}$.

The metric-distance routing theorem Definition 388.17 supplies the $\mathsf{Met}^{\uparrow}$ witnesses used to compare the source history with the center and to consume the target-side tolerance. The positive-radius ledger and fold exactness theorem route the requested precision through $\mathsf{Rat}^{\uparrow}$ rows before the pointwise-modulus row is read.

At the real-facing boundary, Definition 388.15 restricts every $\mathsf{Real}^{\uparrow}$ seal read to the same finite-net, metric, continuous-map, and rational-fold rows. Transport and continuation data in T and Q, together with package and naming provenance in P and N, carry the factorization without adding open-cover or choice-selected data. □

Theorem 388.24 (CompactUniformContinuity RealUp consumer boundary)**.** *A* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet can read the compact finite-net coverage rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows,* $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{Rat}^{\uparrow}$ *precision and radius rows, transported* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *data. It cannot read open-cover compactness, choice-selected subcovers, host real equality, selected real minima, or an external host modulus.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket realup consumer boundary`

Proof. Read the boundary statement Definition 388.15. It already restricts real-name consumers to finite-net coverage, metric witnesses, pointwise-modulus rows, positive rational folds, and the resulting uniform precision history.

The modulus certificate of Definition 388.22 identifies the uniform precision as a finite carrier row rather than a host-level modulus. The finite-net factorization of Definition 388.23 records how compact, metric, rational, and real-facing reads pass through the displayed packet coordinates.

Finally apply the non-escape ledger Definition 388.18. Since the carrier has no coordinate for open covers, selected subcovers, host real equality, selected minima, or external modulus functions, none of these objects can be exported to a $\mathsf{Real}^{\uparrow}$ consumer. □

Theorem 388.25 (CompactUniformContinuity classifier transport)**.** *Let*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N)$$

and

$$H' = (X', Y', F', E', D', \mathcal{P}', C', M', R', W', T', Q', P', N')$$

be accepted $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carriers. If the source metric rows, target metric rows, graph rows, tolerance rows, precision rows, finite-net spines, coverage rows, pointwise-modulus rows, radius rows, lower-bound fold rows, continuation routes, package rows, and local* NameCert *rows are transported componentwise by the displayed* hsame, Cont, Pkg, *and* NameCert *evidence in the two packets, then* $\mathsf{CompactUniformContinuityClassifier}(H, H')$ *is preserved. In particular, classifier preservation is a statement about the displayed* BHist *rows of the carrier and not about open covers, selected subcovers, host minima, sequence compactness, or host modulus functions.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket classifier transport`

Proof. Project both packets through Definition 388.1. The carrier lists exactly the metric, graph, tolerance, precision, finite-net, coverage, pointwise-modulus, radius, fold, route, package, and naming rows that may be transported.

Apply the componentwise classifier of Definition 388.2. It compares the rows just projected and has no comparison slot for an open cover, selected subcover, sequence principle, host minimum, or host modulus function.

The local obligation theorem Definition 388.3 identifies these transported rows as the $\mathsf{NameCert}_{\mathsf{CompactUniformConti}}$ surface. The finite-net handoff, radius-fold exactness, uniform-precision consumer route, and $\mathsf{Real}^{\uparrow}$ consumer boundary of Definition 388.13, Definition 388.14, Definition 388.21, and Definition 388.24 show that downstream consumers read only the transported finite rows. Hence the classifier is preserved by the displayed componentwise transport. □

Theorem 388.26 (CompactUniformContinuity precision-route totality)**.** *For an accepted* CompactUniformContinui *carrier*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N),$$

every accepted read of the precision history D factors through the finite route

$$\mathcal{P}, C, M, R, W, T, Q, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *metric consumer may read it. Thus a real-facing metric consumer reaches the uniform precision only through compact finite-net coverage, pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows, positive* $\mathsf{Rat}^{\uparrow}$ *radius and fold rows, transported* hsame *evidence, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *data.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket precision route totality`

Proof. Start from the carrier of Definition 388.1. The precision row is the displayed coordinate D, while $\mathcal{P}, C, M, R, W, T, Q, P$, and N are the finite rows that govern how D is produced and consumed.

Use Definition 388.14 to identify D with the lower-bound fold over the positive-radius ledger R and witness W. Use Definition 388.13 to route the precision through the finite ProbeBundle spine $\mathcal{P}$, coverage rows C, and pointwise-modulus rows M.

The classifier and local obligation surface of Definition 388.2 and Definition 388.3 keep this route inside the transported hsame, Cont, Pkg, and NameCert rows T, Q, P, N. Finally Definition 388.21 and Definition 388.24 force every real-facing metric read to pass through that route before it can consume D. □

Theorem 388.27 (CompactUniformContinuity NameCert ledger exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactUniformCon}}$ *ledger is exhausted by the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coverage rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows,* MetricDistanceWitness *rows,* $\mathsf{Rat}^{\uparrow}$ *fold rows,* $\mathsf{Real}^{\uparrow}$ *seal reads, the transported* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It contains no row for open covers, selected subcovers, host minima, sequence compactness, or host modulus functions.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket namecert ledger exhaustion`

Proof. Inspect the carrier and classifier in Definition 388.1 and Definition 388.2. The carrier supplies the finite ledgers, and the classifier confirms that only those ledgers participate in accepted comparison and transport.

The NameCert obligation theorem Definition 388.3 names the same ledgers as the local certificate surface. The finite-net handoff of Definition 388.13 accounts for the compact coverage rows, and the radius-fold exactness theorem Definition 388.14 accounts for the rational fold rows.

It remains to account for downstream reads. The uniform-precision consumer route of Definition 388.21 carries precision reads through the transported hsame, Cont, Pkg, and NameCert rows, while Definition 388.24 accounts for the $\mathsf{Real}^{\uparrow}$ seal boundary. Since every named read is already in these ledgers and the carrier has no additional coordinate, the local NameCert ledger is exhausted by the listed rows. □

Theorem 388.28 (CompactUniformContinuity standard bridge source packet)**.** *For an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carrier*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N),$$

the displayed rows assemble the standard bridge source from compact finite-net data and pointwise continuous-map modulus data to a uniform precision history. The source consists exactly of the source and target metric rows X, Y, the graph row F, the requested tolerance E, the precision row D, the finite ProbeBundle *spine $\mathcal{P}$, coverage rows C, pointwise-modulus rows M, positive-radius ledger R, lower-bound fold witness W, transported* hsame *rows T, continuation routes Q, package provenance P, and local naming row N.*

Every bridge source read of the compact-uniform-continuity packet therefore factors through the finite carrier rows already named by Definition 388.1; it does not require a separate open-cover compactness object, sequence principle, selected real minimum, or host modulus.

Proof. Project the accepted carrier H. The metric, graph, tolerance, precision, finite-net, coverage, modulus, radius, fold, transport, continuation, package, and naming coordinates are all displayed before any bridge consumer is allowed to read the packet.

The compact part of the source is repacked from $\mathcal{P}$ and C, using Definition 388.16 to identify the bundled centers and coverage witnesses. The continuous part is repacked from F and M, and Definition 388.20 aligns those modulus rows with the radius ledger read through R and W.

The metric reads are then routed through X and Y by Definition 388.17. The output precision is the same row D consumed by Definition 388.21, so the bridge source is assembled by projection and repacking of displayed BHist rows. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket bridge source packet`

Theorem 388.29 (CompactUniformContinuity bridge non-escape boundary)**.** *A standard bridge consumer of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet can read only the finite* ProbeBundle *coverage rows,* MetricDistanceWitness *rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus rows,* $\mathsf{Rat}^{\uparrow}$ *fold ledger, and the displayed* hsame, Cont, Pkg, *and* NameCert *provenance. It cannot read open-cover compactness, a sequence principle, a selected real minimum, or a host modulus.*

Proof. Start with the bridge source packet of Definition 388.28. Its coordinates are exactly the rows of the accepted carrier, so a bridge consumer first reaches the finite-net, metric, pointwise-modulus, rational-fold, transport, continuation, package, and naming ledgers.

The non-escape ledger of Definition 388.18 rules out any additional compactness or modulus coordinate in the packet. The real-seal boundary of Definition 388.15 gives the same exclusion for real-name consumers, and Definition 388.19 identifies the displayed finite fold as the only source of the uniform precision.

Thus every bridge read remains inside the finite carrier surface. Since the carrier has no coordinate for open covers, sequences, selected minima, or host modulus functions, none of those objects can be projected, transported, or repacked by the bridge consumer. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket bridge non escape boundary`

Theorem 388.30 (CompactUniformContinuity standard bridge obligation package)**.** *The bridge source packet of Definition 388.28, the non-escape boundary of Definition 388.29, the finite-net handoff of Definition 388.13, the radius-fold exactness theorem Definition 388.14, the* ProbeBundle *coverage theorem Definition 388.16, the metric-distance routing theorem Definition 388.17, and the uniform-precision consumer route of Definition 388.21 assemble the standard bridge obligation package for* $\mathsf{CompactUniformContinuity}^{\uparrow}$*.*

The assembled package exports a finite BEDC bridge-facing route from compact finite-net coverage and pointwise continuity rows to the uniform precision history, while preserving the stated boundary against open covers, sequences, selected real minima, and host modulus functions.

Proof. First build the source packet. Use Definition 388.28 to project the carrier coordinates and repack them as the bridge-facing source over the finite BHist rows.

Next assemble the compact and metric routes. The finite-net handoff supplies the center route, radius-fold exactness supplies the lower-bound comparison, ProbeBundle coverage supplies the finite coverage spine, and metric-distance routing supplies the source and target distance witnesses.

The uniform-precision consumer route then consumes these rows to produce the requested precision history. The bridge non-escape boundary keeps the resulting package inside the finite carrier, transport, continuation, package, and naming provenance.

All components are therefore obtained by constructive projection from the displayed carrier and by repacking through previously named paper rows. No step adds an external compactness principle, a selected minimum, a sequence argument, or a host modulus. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket standard bridge obligation package`

Theorem 388.31 (CompactUniformContinuity finite-net modulus extraction)**.** *For an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *carrier*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N),$$

the finite uniform precision row D *is extracted from the* $\mathsf{CompactMetric}^{\uparrow}$ ProbeBundle *coverage rows* $\mathcal{P}, C$*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *centerwise modulus rows* M*, and the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold rows* R, W*. The extraction is stable under the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *rows, and it exports no open-cover subcover, sequence argument, selected real minimum, or host modulus.*

Proof. Project the compact rows first. The finite-net handoff of Definition 388.13 and the ProbeBundle coverage theorem Definition 388.16 identify the bundled centers and the half-radius coverage witnesses read from $\mathcal{P}$ and C.

Next extract the radius information. The radius-fold exactness theorem Definition 388.14 identifies D as the finite $\mathsf{Rat}^{\uparrow}$ lower-bound fold over R, W, so each centerwise modulus row in M receives a compatible precision comparison.

Finally apply the metric-distance routing theorem Definition 388.17 and the uniform-precision consumer route Definition 388.21. These routes consume only displayed metric witnesses, transports, continuations, package provenance, and local naming provenance, so the extracted modulus remains a finite BEDC row. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket finite net modulus extraction`

Theorem 388.32 (CompactUniformContinuity bridge modulus assembly)**.** *The standard compact-metric uniform-continuity certificate for an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet is assembled over* MetricDistanceWitness*,* ProbeBundle*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Its bridge-facing source is the tuple of carrier rows* $X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N$*, and its modulus conclusion is the finite precision row* D *extracted by Definition 388.31.*

Proof. Build the source from Definition 388.28. The metric coordinates X, Y, the graph coordinate F, the finite net $\mathcal{P}, C$, the pointwise modulus rows M, and the rational fold rows R, W are all visible before the bridge-facing certificate is read.

Use Definition 388.31 to obtain the finite precision row. Then route source and target distance reads through Definition 388.17 and consume the result through Definition 388.21.

The assembly is completed by the standard bridge obligation package of Definition 388.30. The displayed $\mathsf{Real}^{\uparrow}$ boundary is inherited from the real-seal rows already present in the carrier surface, so the certificate is a repacking of finite BEDC rows rather than a theorem about ambient compact metric spaces. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket bridge modulus assembly`

Theorem 388.33 (CompactUniformContinuity real metric seal non-escape)**.** *A* $\mathsf{Real}^{\uparrow}$ *metric consumer of the bridge modulus assembly in Definition 388.32 cannot replace the finite route by open-cover compactness, sequential compactness, host minima, or an external modulus. Every real-facing metric read remains routed through* MetricDistanceWitness *rows, the finite* ProbeBundle *coverage ledger, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows, the* $\mathsf{Rat}^{\uparrow}$ *fold, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Start from the bridge non-escape boundary Definition 388.29. It restricts a bridge consumer to the finite-net, metric, continuous-map, rational-fold, transport, continuation, package, and naming rows of the accepted carrier.

For real-facing metric reads, apply Definition 388.24 and Definition 388.17. These theorems keep source and target distance observations behind displayed MetricDistanceWitness evidence and prevent direct host metric access.

The finite precision row is the one assembled in Definition 388.32. Since that assembly depends on the finite-net extraction of Definition 388.31, a real metric consumer has no coordinate from which to read open covers, sequences, minima, or an external modulus. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket real metric seal nonescape`

Theorem 388.34 (CompactUniformContinuity constant-map example)**.** *A constant-map* CompactUniformContinui *consumer is represented by a finite BEDC packet whose carrier rows have one displayed target center, one centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row, a trivial* MetricDistanceWitness *target-distance row, the inherited* ProbeBundle *coverage ledger on the source, the* $\mathsf{Rat}^{\uparrow}$ *fold row selecting the displayed precision history, and the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance. Its uniform modulus conclusion is read from that single centerwise row through Definition 388.32.*

Proof. Project the constant-map carrier. The target metric read is attached to one displayed target center, while the source side keeps the same finite ProbeBundle coverage rows and package provenance as the ambient compact carrier.

Apply Definition 388.31. The radius fold has only the displayed centerwise modulus row to consume, so the extracted precision row is a finite BEDC row rather than a selected host modulus.

Finally route the packet through Definition 388.32 and Definition 388.33. The constant-map consumer reads the standard bridge packet without invoking open covers, sequences, selected minima, or external compactness. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket constant map consumer route`

Theorem 388.35 (CompactUniformContinuity finite-net composition example)**.** *Given two finite BEDC consumer packets whose compact finite-net rows, pointwise* ContinuousMap$^{\uparrow}$ *modulus rows,* MetricDistanceWitness *rows,* Rat$^{\uparrow}$ *fold rows,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance compose along a displayed intermediate metric row, their composition repacks those rows through Definition 388.32. The resulting consumer is again a finite-net compact-uniform-continuity packet, and its bridge modulus is obtained from the composed finite rows rather than from an open-cover subcover, sequence argument, selected real minimum, or host modulus.*

Proof. First compose the visible finite-net data. The intermediate metric row is read only through displayed MetricDistanceWitness evidence, and the source coverage rows remain the finite ProbeBundle rows of the first packet.

Next repack the pointwise modulus rows. Use Definition 388.31 for each packet and transport the resulting rational precision rows along the displayed hsame and Cont routes.

Apply Definition 388.32 to the composed rows. The bridge packet consumes the composed finite-net, pointwise-modulus, metric-distance, rational-fold, package, and naming data as its entire source.

Finally use Definition 388.33. Since the composed carrier has no open-cover, sequence, selected-minimum, or external-modulus coordinate, the composed consumer remains a finite BEDC packet. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket finite net composition route`

Theorem 388.36 (CompactUniformContinuity real-interval consumer example)**.** *A real-interval compact-uniform-continuity consumer is represented by a finite BEDC packet routing* Real$^{\uparrow}$ *metric reads through* MetricDistanceWitness *rows,* ProbeBundle *coverage,* ContinuousMap$^{\uparrow}$ *pointwise rows,* Rat$^{\uparrow}$ *fold rows,* hsame*,* Cont*,* Pkg*, and* NameCert$_{\text{CompactUniformContinuity}^{\uparrow}}$*. Its interval endpoint reads consume the bridge modulus assembly and the real metric seal boundary, so the consumer obtains a public real-interval modulus without open-cover compactness, sequential compactness, host real equality, selected minima, or ambient completeness.*

Proof. Start with the interval endpoint reads as Real$^{\uparrow}$ metric reads. By Definition 388.17, those reads must pass through displayed MetricDistanceWitness rows rather than through a host metric.

Apply Definition 388.16 and Definition 388.31. The interval consumer receives finite coverage rows, pointwise ContinuousMap$^{\uparrow}$ modulus rows, and the Rat$^{\uparrow}$ fold row that supplies the uniform precision history.

Now use Definition 388.32. The real-interval consumer reads the assembled bridge packet through the displayed hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 388.33. The consumer has no route to open-cover compactness, sequential compactness, host equality, selected real minima, or ambient completeness, so the example is a finite BEDC consumer packet over the existing carrier rows. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket real interval consumer route`

Theorem 388.37 (CompactUniformContinuity kernel net observation surface)**.** *For every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet*

$$H = (X, Y, F, E, D, \mathcal{P}, C, M, R, W, T, Q, P, N),$$

the finite ProbeBundle *net rows* $\mathcal{P}, C$*, the source and target* MetricDistanceWitness *rows, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows* M*, the positive* $\mathsf{Rat}^{\uparrow}$ *radius rows* R*, the finite lower-bound fold* W*, the* hsame *transport rows* T*, the displayed* Cont *routes* Q*, the package row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuit}}$ *row* N *form the complete observation surface of the kernel packet. No accepted packet exports an additional open-cover, sequential, host-minimum, or external-modulus coordinate.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket kernel net observation surface`

Proof. Project the accepted carrier. The carrier definition exposes exactly the source metric row X, target metric row Y, graph row F, tolerance row E, precision row D, finite net data $\mathcal{P}, C$, pointwise modulus table M, positive rational radius data R, finite fold row W, transports T, continuations Q, package provenance P, and local naming row N.

The finite-net handoff, ProbeBundle coverage theorem, metric-distance routing theorem, and radius-fold exactness theorem identify how these coordinates are read by consumers. Each read is therefore a projection from the displayed BHist packet followed by repacking through the stored hsame, Cont, Pkg, and NameCert rows.

Since the carrier has no field for open-cover compactness, sequential compactness, selected host minima, or an external modulus function, those objects cannot appear in the observation surface. The stated rows are consequently exhaustive for kernel-level observation. □

Theorem 388.38 (CompactUniformContinuity modulus ledger exhaustion)**.** *For every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet, the selected uniform precision row* D *is exhausted by the finite* ProbeBundle *net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows, positive* $\mathsf{Rat}^{\uparrow}$ *radii, finite lower-bound fold,* MetricDistanceWitness *routing rows, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance of Definition 388.37. The ledger for* D *cannot be read through open covers, selected minima, or host modulus choice.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket modulus ledger exhaustion`

Proof. Begin with the observation surface of Definition 388.37. It lists every coordinate available to the packet before a consumer requests the uniform precision.

Use radius-fold exactness to identify D with the finite lower-bound fold over the displayed positive rational rows. Then use finite-net modulus extraction to attach that fold to the ProbeBundle coverage rows and to the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows.

Metric-distance routing supplies only the displayed source and target witness rows, and the remaining transport, continuation, package, and naming rows preserve the same finite ledger. Because no carrier coordinate stores an open cover, a selected minimum, or a host modulus chooser, the precision row is exhausted by the finite rows listed above. □

Theorem 388.39 (CompactUniformContinuity real metric consumer triad)**.** *Every* $\mathsf{Real}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet factors through the compact-net row, the metric-distance row, and the continuous-map modulus row before it reads the uniform precision row* D*. The consumer route is therefore the triad*

$$\mathsf{ProbeBundle} \longrightarrow \mathsf{MetricDistanceWitness} \longrightarrow \mathsf{ContinuousMap}^{\uparrow}$$

inside the displayed finite packet, followed by the $\mathsf{Rat}^{\uparrow}$ *fold and the stored* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket real metric consumer triad`

Proof. Start with a real or metric consumer read. By the real-facing boundary and metric-distance routing the read cannot access a host metric directly; it must pass through the displayed MetricDistanceWitness rows.

The source-side compact information is supplied first by the finite ProbeBundle coverage rows. The target-side behavior is then supplied by the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ pointwise modulus rows, with the metric witnesses connecting the requested source and target observations.

Finally apply modulus ledger exhaustion. The uniform precision row D is read only after the compact-net, metric-distance, and continuous-map modulus rows have been assembled with the positive $\mathsf{Rat}^{\uparrow}$ fold and the stored provenance rows, so the consumer factors through the stated triad. □

Theorem 388.40 (CompactUniformContinuity equicontinuous family consumer)**.** *An equicontinuous family packet whose displayed family-member rows expose the same finite* $\mathsf{CompactMetric}^{\uparrow}$ ProbeBundle *coverage,* $\mathsf{ContinuousMap}^{\uparrow}$ *centerwise modulus rows,* MetricDistanceWitness *source and target rows, positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance admits a single compact-uniform-continuity consumer route for every displayed family member. The shared route reads the finite bridge modulus assembly through the common coverage and fold rows, so each member receives the same finite BEDC uniform-precision consumer boundary.*

The conclusion is a consumer route over the displayed family rows only. It does not use open-cover compactness, sequential compactness, host minima, or a host modulus.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket equicontinuous family consumer`

Proof. First extract the shared precision row. Apply Definition 388.31 to the common ProbeBundle coverage, centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus, and positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold rows carried by the family packet.

Next assemble the bridge-facing consumer route. Use Definition 388.32; the source and target metric reads for a displayed family member are supplied by its MetricDistanceWitness rows, while the uniform precision row is the shared finite row extracted from the family packet.

Finally apply Definition 388.33. The $\mathsf{Real}^{\uparrow}$ seal boundary keeps each displayed member inside the same finite-net, metric-witness, rational-fold, transport, continuation, package, and naming provenance, so the family consumer has no coordinate for open-cover compactness, sequential compactness, host minima, or a host modulus. □

Theorem 388.41 (CompactUniformContinuity net-modulus consumer triad)**.** *Every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet exposes a downstream consumer triad consisting of the finite* ProbeBundle *compact-net coverage supplied by* $\mathsf{CompactMetric}^{\uparrow}$*, the centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows transported through displayed* MetricDistanceWitness *rows, and the positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold that yields the uniform precision row. Every* $\mathsf{Real}^{\uparrow}$ *or compact-metric consumer read of the uniform modulus factors through these three finite rows together with the recorded* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *routes.*

The certificate therefore contains no open-cover compactness, sequential compactness, host minimum, or external modulus function. Its consumer surface is exactly the finite compact-net, pointwise-modulus, metric-witness, rational-fold, transport, continuation, package, and naming ledger displayed by the packet.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket net modulus consumer triad`

Proof. First read the compact-net row. By Definition 388.16, each admitted source history reaches a bundled center through the finite ProbeBundle coverage supplied by the $\mathsf{CompactMetric}^{\uparrow}$ side of the packet.

Next read the pointwise modulus row through the metric witnesses. Use Definition 388.17 to keep source and target distance observations behind displayed MetricDistanceWitness rows, and use Definition 388.13 to connect those rows with the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus data.

Finally read the uniform precision row. The radius-fold exactness theorem Definition 388.14 identifies the precision with the positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold, and Definition 388.38 says that this fold exhausts the displayed modulus ledger.

Apply the consumer boundary supplied by Definition 388.39 and Definition 388.18. Every real or compact-metric read is consequently routed through the three finite rows and their stored hsame, Cont, Pkg, and NameCert provenance, with no route to open covers, sequences, host minima, or an external modulus. □

Theorem 388.42 (CompactUniformContinuity root consumer triad)**.** *The root consumer surface of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet is the triad consisting of finite* ProbeBundle *coverage,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows, and* MetricDistanceWitness *routing. Its exported precision is the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger folded by the finite lower-bound witness, transported with the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *rows.*

This triad is the full root consumer package for compact-uniform-continuity reads. It has no route through open covers, sequential compactness, host minima, or external modulus functions.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root consumer triad`

Proof. Project the accepted carrier and classifier of Definition 388.1 and Definition 388.2. The visible coordinates already contain the compact finite-net rows, pointwise modulus rows, metric witnesses, rational-radius ledger, fold witness, transport rows, continuation routes, package provenance, and local naming row.

Use finite-net handoff, ProbeBundle coverage, and metric-distance routing to isolate the compact and metric sides of the route. The pointwise $\mathsf{ContinuousMap}^{\uparrow}$ rows are then consumed at the bundled centers, and radius-fold exactness supplies the positive rational lower-bound fold that exports the uniform precision row.

Finally apply the non-escape ledger and real metric seal boundary. Since every listed read is a projection from the displayed packet followed by transport through hsame, Cont, Pkg, and NameCert, no outside compactness principle or host modulus can enter the root consumer package. □

Theorem 388.43 (CompactUniformContinuity root downstream unblock package)**.** *The* CompactUniformContinuit *downstream package for* UniformModulus$^{\uparrow}$ *and compact-metric consumers consists of the* ProbeBundle *compact-net row, the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row, the* MetricDistanceWitness *route, the positive* Rat$^{\uparrow}$ *fold, the* Real$^{\uparrow}$ *seal boundary, and the displayed* hsame, Cont, Pkg, *and* NameCert$_{\text{CompactUniformContinuity}^{\uparrow}}$ *provenance. It is a finite packet over the root consumer triad and exports no open-cover compactness, host minimum, sequence compactness, or external modulus selector.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root downstream unblock package`

Proof. Project the root consumer triad of Definition 388.42. Its three source rows are the finite compact net, the pointwise modulus ledger, and the metric-witness route, with the positive rational fold supplying the exported precision.

Use finite-net handoff and metric-distance routing to attach each centerwise modulus read to the displayed ProbeBundle and MetricDistanceWitness rows. Then use rational fold export to identify the precision consumed by UniformModulus$^{\uparrow}$ and compact-metric consumers with the finite positive Rat$^{\uparrow}$ ledger.

Finally apply the compact-continuous NameCert export and the real seal boundary. Repacking those rows with hsame, Cont, Pkg, and NameCert provenance gives the downstream package and leaves no coordinate for open covers, host minima, sequence compactness, or external modulus selection. □

Theorem 388.44 (CompactUniformContinuity ProbeBundle center selection)**.** *In every accepted* CompactUniformContinuity$^{\uparrow}$ *root packet, source-side compact access is mediated by finitely many* CompactMetric$^{\uparrow}$ ProbeBundle *centers. A consumer may read a source point only by selecting one displayed center row, transporting it through the packet's* hsame *rows, reading the associated* Cont *route, and carrying the result through the* Pkg *provenance and local* NameCert$_{\text{CompactUniformContinuity}^{\uparrow}}$ *row.*

The selected center is a row of the finite packet, not an open-cover point, host compactness witness, or minimum-selection operation.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root probebundle center selection`

Proof. Project the root consumer triad of Definition 388.42. Its compact side is the finite ProbeBundle coverage row, already attached to CompactMetric$^{\uparrow}$ provenance.

Apply finite-net handoff and metric-distance routing. These rows force each source read through one bundled center before any modulus or target-distance row can be consumed.

Finally transport the selected center through the displayed hsame and Cont rows and repack it with Pkg and NameCert provenance. Since the carrier contains no open-cover or host-minimum coordinate, the center selection remains a finite packet read. □

Theorem 388.45 (CompactUniformContinuity pointwise modulus row)**.** *For each finite* CompactMetric$^{\uparrow}$ ProbeBundle *center read from an accepted* CompactUniformContinuity$^{\uparrow}$ *packet, the only modulus data available to the root consumer is the matching* ContinuousMap$^{\uparrow}$ *pointwise modulus row. That*

row is consumed together with the source and target MetricDistanceWitness *rows, the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger, and the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance.*

No root consumer can replace the pointwise row by an external modulus function or a host continuity theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root pointwise modulus row`

Proof. Begin with a center supplied by Definition 388.44. The center is already a displayed finite packet row.

Use the finite-net handoff to reach the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus row. Metric-distance routing then supplies exactly the source-center and target-output MetricDistanceWitness rows consumed by that modulus read.

The positive $\mathsf{Rat}^{\uparrow}$ radius ledger and the stored transport, continuation, package, and naming rows carry this pointwise read into the root packet. The carrier has no coordinate for an external modulus, so the displayed pointwise row exhausts the route. □

Theorem 388.46 (CompactUniformContinuity rational-fold consumer)**.** *Every root consumer of uniform precision in an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet factors through the finite* $\mathsf{Rat}^{\uparrow}$ *positive-radius fold. The fold consumes the finitely many centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows selected through the* $\mathsf{CompactMetric}^{\uparrow}$ ProbeBundle *centers, compares them only through displayed* MetricDistanceWitness *rows, and exports the precision with the packet's* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance.*

The consumer surface contains no open-cover compactness, host minimum, or external modulus selector.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket root rational fold consumer`

Proof. First assemble the finite center and pointwise rows using Definition 388.44 and Definition 388.45. These rows provide the compact, continuous, and metric inputs to the precision read.

Next apply radius-fold exactness. The exported precision is the positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold over the displayed centerwise radius rows, so the consumer never leaves the finite rational ledger.

Finally use the root consumer triad. Transport and continuation rows carry the folded precision with package and naming provenance, while the non-escape boundary excludes open covers, host minima, and external modulus selection. □

Theorem 388.47 (CompactUniformContinuity ProbeBundle-to-metric route)**.** *Every source-side root read in an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet passes from the finite* ProbeBundle *spine to displayed source and target* MetricDistanceWitness *rows before it can consume a* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus row. The route is mediated by the carrier's* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}}$ *provenance, and it never reads a host metric directly.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket probe metric route`

Proof. Begin with the finite-net coverage theorem. It sends each source-point history to a bundled center through the displayed ProbeBundle spine and coverage ledger.

Metric-distance routing then supplies the only source-center and target-output distance reads available to the packet. These witnesses are the metric coordinates of the carrier, transported componentwise by the classifier.

With those metric witnesses in place, the finite-net handoff consumes the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ pointwise modulus row. The non-escape ledger prevents replacing any part of this route by host metric access, an open cover, or an unrecorded selector. □

Theorem 388.48 (CompactUniformContinuity rational fold export)**.** *The only exported uniform precision of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet is the carrier row D read through the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger R and the finite lower-bound fold witness W. Every downstream consumer of uniform precision factors through this rational fold together with the displayed* ProbeBundle *coverage,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise rows,* MetricDistanceWitness *routes,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket rational fold export`

Proof. Read the precision coordinate D from the accepted carrier. The carrier also displays the positive radius ledger R and the fold witness W, but it has no coordinate for a host minimum or selected real-radius operation.

Radius-fold exactness identifies D as the finite lower-bound fold over R. The finite-net modulus extraction theorem then attaches this fold to the ProbeBundle coverage rows and the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows.

Metric-distance routing and the non-escape ledger keep the resulting precision route inside displayed BEDC rows. Therefore the rational fold is the sole exported uniform precision, and no downstream consumer can bypass it through external compactness or modulus choice. □

Theorem 388.49 (CompactUniformContinuity four-source handoff correspondence)**.** *For every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *root packet, the finite* ProbeBundle *compact-net rows, the pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows, the source and target* MetricDistanceWitness *rows, the positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, and the* $\mathsf{Real}^{\uparrow}$ *seal row form one finite handoff correspondence. A consumer may enter the handoff through any displayed compact, continuous, metric, rational, or real row, but every such entry reads the same carrier packet with the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance.*

The correspondence has no coordinate for open-cover compactness, sequence compactness, host minimum selection, or an external modulus. It is the root handoff of the displayed finite rows.

Proof. Start with the compact-net side. Apply Definition 388.13 to bind the finite ProbeBundle net to the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows.

Next read the metric and rational rows. Metric-distance routing supplies the only source and target distance observations, while radius-fold exactness identifies the positive $\mathsf{Rat}^{\uparrow}$ fold as the precision row consumed by the handoff.

Finally attach the real-facing boundary. Use the real seal boundary together with the rational fold export theorem to repack the compact, continuous, metric, rational, and real rows as one carrier correspondence. Since each step is a projection from the accepted packet followed by displayed transport, no outside compactness or modulus coordinate is introduced. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket four source handoff correspondence`

Theorem 388.50 (CompactUniformContinuity rational fold metric compatibility)**.** *In an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet, the rational-fold precision row is compatible with every source-center and target-output metric read consumed by the handoff of Definition 388.49. Each* $\mathsf{MetricDistanceWitness}$ *read is routed through the displayed compact-net center, then compared only against the precision exported by the positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold and its recorded* hsame *and* Cont *transports.*

Thus no metric consumer can replace the rational fold by a host minimum, an unrecorded real radius, or an external modulus choice.

Proof. Project the displayed metric witnesses from the carrier. By metric-distance routing, every source-center or target-output distance read is one of these witnesses and not a host metric call.

Apply finite-net handoff to place those metric witnesses on the same route as the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows. Radius-fold exactness then identifies the precision read by that route with the positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold.

The rational fold export theorem says that all downstream precision consumers factor through this fold. Therefore every metric read consumed by the handoff is compatible with the same rational precision row, and no hidden radius selector is available. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket rational fold metric compatibility`

Theorem 388.51 (CompactUniformContinuity compact-continuous NameCert export)**.** *The compact-net rows, pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows,* $\mathsf{MetricDistanceWitness}$ *rows, positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, and* $\mathsf{Real}^{\uparrow}$ *seal row of an accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *packet export a finite* $\mathsf{NameCert}_{\mathsf{CompactUniformConti}}$ *package. The exported package is exactly the displayed carrier, classifier, transport, continuation, package, and naming ledger over those rows.*

It exports no open-cover compactness, sequence compactness, host minimum, host modulus, or additional real-coordinate selector.

Proof. First assemble the finite handoff correspondence from Definition 388.49. This supplies the compact, continuous, metric, rational, and real rows as one accepted packet with fixed provenance.

Next apply Definition 388.50. The metric observations and rational precision row agree along the displayed finite-net route, so the exported package has a single precision ledger.

Finally use the real seal boundary and rational fold export theorem. The real-facing row can only read the sealed finite packet, while the rational fold remains the only exported precision. Repacking these rows gives the finite $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ package and leaves no coordinate for open covers, sequence compactness, host minima, or external modulus data. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket compact continuous namecert export`

Theorem 388.52 (CompactUniformContinuity four-source bridge determinacy)**.** *For every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *root packet, the consumer-facing bridge determined by Definition 388.49 and Definition 388.51 has a unique displayed finite source. Any route, transport, readback, or handoff through the compact, continuous, metric, rational, or real side factors through the finite* $\mathsf{ProbeBundle}$ *compact-net rows, the centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows, the source and target* $\mathsf{MetricDistanceWitness}$ *rows, the positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the common* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance.*

Consequently an unlisted bridge coordinate, including open-cover compactness, sequence compactness, host minimum selection, a hidden real-radius selector, or an external modulus, is unavailable to every such consumer.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket four source bridge determinacy`

Proof. Apply Definition 388.49. It states that a consumer may enter through any displayed compact, continuous, metric, rational, or real row, but every such entry reads the same accepted carrier packet and the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ provenance. Thus the compact entry is the finite ProbeBundle net row, the continuous entry is the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus row, the metric entry is one of the displayed source or target MetricDistanceWitness rows, the rational entry is the positive $\mathsf{Rat}^{\uparrow}$ fold row, and the real entry is the $\mathsf{Real}^{\uparrow}$ seal row.

Apply Definition 388.51. The export packages exactly the displayed carrier, classifier, transport, continuation, package, and naming ledger over those rows. Therefore a route is read through the displayed Cont continuation rows, a transport is read through the displayed hsame and classifier rows, a readback is a projection to one of the listed compact, continuous, metric, rational, or real rows, and a handoff is the finite correspondence obtained by repacking those projections with the same Pkg and $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ provenance.

These four consumer actions have the same source because the correspondence fixes the carrier packet before the export repacks it, and the export exposes no second carrier, classifier, transport, continuation, package, or naming ledger. Hence two displayed consumer entries cannot determine different bridge data: after projection they land in the same finite row list and after export they carry the same provenance.

The nonlisted cases are excluded by the same two rows. The correspondence explicitly has no coordinate for open-cover compactness, sequence compactness, host minimum selection, or an external modulus, and the export adds no open-cover, sequence, host-minimum, host-modulus, or additional real-coordinate selector. Since an unavailable coordinate is neither a projection from the carrier nor a transported component of the exported $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ package, no consumer-facing route, transport, readback, or handoff can factor through it. □

Corollary 388.53 (CompactUniformContinuity four-source handoff consumer determinacy)**.** *For every accepted* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *root packet, fix the finite* ProbeBundle *compact-net rows, the centerwise* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus rows, the source and target* MetricDistanceWitness *rows, the positive* $\mathsf{Rat}^{\uparrow}$ *fold rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the common* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance named by Definition 388.49. Then every accepted consumer-facing use that reports a route selection, a normal-form readback, or an exactness decision for the four-source handoff reports the one determined by those fixed rows.*

The finite route is the displayed handoff through compact-net coverage, centerwise continuity, source and target metric witnesses, rational fold precision, and the real seal. The normal-form readback is the ordered projection to those same rows with the displayed transport, continuation, package, and local naming provenance. The exactness decision accepts precisely the consumer uses that factor through that fixed finite route. No open-cover compactness coordinate, sequence compactness coordinate, host minimum selector, hidden real-radius selector, external modulus, alternate hidden coordinate, or duplicate classifier source can supply a different accepted route, normal form, or exactness decision.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket four source handoff consumer determinacy`

Proof. Fix the accepted root packet and the listed local rows. By Definition 388.49, a consumer may enter through any displayed compact, continuous, metric, rational, or real row, but

every such entry reads the same carrier packet with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ provenance. Thus the five source families are not competing sources; they are five entry faces of one finite handoff correspondence.

Apply Definition 388.52. It identifies the consumer-facing bridge as having a unique displayed finite source and forces every route, transport, readback, or handoff through the compact, continuous, metric, rational, or real side to factor through exactly the fixed row list. Since the rows have been fixed, two accepted consumer uses cannot choose distinct finite sources: after factorization, both land in the same compact-net rows, the same centerwise modulus rows, the same source and target metric-witness rows, the same rational fold rows, the same real seal row, and the same structural provenance.

A reported route selection is therefore the displayed factorization route through those rows. A reported normal-form readback is the same ordered projection of the fixed carrier: first the finite ProbeBundle compact-net and coverage rows, then the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ modulus rows, then the source and target MetricDistanceWitness rows, then the positive $\mathsf{Rat}^{\uparrow}$ fold, and finally the $\mathsf{Real}^{\uparrow}$ seal, all transported and repacked by hsame, Cont, Pkg, and the local NameCert row. A reported exactness decision is consequently the decision that a consumer use is accepted exactly when it factors through this fixed route and its structural provenance.

It remains to exclude alternate sources for the same public decision. If a purported duplicate classifier source is one of the displayed classifier, transport, continuation, package, or naming rows, then it is already part of the fixed provenance and determines the same route. If it is not one of those rows, then Definition 388.52 and Definition 388.49 leave it no projection, transport, readback, or handoff coordinate. The excluded open-cover, sequence, host-minimum, hidden real-radius, and external-modulus cases are therefore rejected before they can become accepted consumer-facing data. □

Theorem 388.54 (CompactUniformContinuity finite-row audit boundary)**.** *Every accepted* CompactUniformContin *consumer route factors through exactly the five displayed analytic rows of the packet: the* $\mathsf{CompactMetric}^{\uparrow}$ ProbeBundle *finite net, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row, the* MetricDistanceWitness *routing row, the positive* $\mathsf{Rat}^{\uparrow}$ *finite lower-bound fold, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary. The factorization is carried by the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ *provenance as the carrier.*

The accepted consumer surface has no coordinate for open-cover compactness, sequential compactness, host minima, or an external modulus function.

Proof. First assemble the compact and continuous rows. Apply Definition 388.13 to route source reads through the displayed finite ProbeBundle centers and into the centerwise $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus rows.

Next attach the metric and rational rows. The metric-distance routing theorem supplies the source-center and target-output MetricDistanceWitness reads, while Definition 388.14 identifies the exported precision with the positive $\mathsf{Rat}^{\uparrow}$ finite lower-bound fold.

The real-facing side is fixed by Definition 388.15. It says that a $\mathsf{Real}^{\uparrow}$ consumer reads the same compact, continuous, metric, and rational rows rather than a host real minimum or external modulus.

Finally use Definition 388.48 and Definition 388.51. These theorems package the rational fold and the compact-continuous rows as the finite $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ export. Since each cited route is a projection from the displayed carrier followed by hsame, Cont, Pkg, and local naming transport, the five rows exhaust the accepted consumer surface and exclude open-cover compactness, sequential compactness, host minima, and external modulus functions. □

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket finite row audit boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}$ ·

Constructive story: A $\mathsf{CompactUniformContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a continuous-map graph row, a positive tolerance history, a compact finite ProbeBundle net, pointwise-modulus rows at the bundled centers, positive $\mathsf{Rat}^{\uparrow}$ radius rows, a finite lower-bound fold selecting a precision history, componentwise hsame rows, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature surface fixes the bridge source, finite-net modulus assembly, real-facing seal boundary, and the constant-map, finite-net composition, and real-interval consumer packets over the displayed rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuity}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CompactUniformContinuityUp.CompactUniformContinuityPacket\_rational\_fold\_export`

Bridge status: none

Origin: human

Not claimed: This chapter does not close bridge status, axiom-purity status, open-cover compactness, sequential compactness, host real equality, selected real minima, or external modulus selection.

Upgrade path: The remaining route is axiom-purity and bridge status over the same finite compact-metric consumer packets.

Chapter 389

A Concrete Naming Certificate for NestedIntervalIntersection$^{\uparrow}$

The NestedIntervalIntersection$^{\uparrow}$ interface names the finite readback surface through which nested interval packets hand a diagonal interval-intersection claim to the real and completion layers. Bishop–Bridges constructive analysis treats nested interval intersection through explicit moduli and finite approximants; BEDC records only the finite diagonal rows, transport ledgers, and seal-facing boundary. It does not claim open-cover compactness, choice of an intersection point, or equality with a host real intersection.

The consumer route enters the L10 real-name chain through the regular Cauchy source surface of chapter 371; this chapter only adds the nested-interval selector and width-window ledger on top of that sibling carrier.

Definition 389.1 (Nested interval intersection carrier)**.** *A* NestedIntervalIntersection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (N, W, D, E, H, C, P, Q)$$

with the following visible coordinates:

1. *N is the source* NestedInterval$^{\uparrow}$ *or* NestedDyadicInterval$^{\uparrow}$ *chain row, including the displayed* DyadicInterval$^{\uparrow}$ *endpoint packets and their* DyadicRatCore$^{\uparrow}$ *lower and upper endpoint reads;*
2. *W is the displayed width surface, either as explicit width rows for the visible chain or as* RegularityModulus$^{\uparrow}$ *rows naming the requested finite precision window;*
3. *D is the finite diagonal selector: for each displayed precision request, it chooses one visible interval row whose width ledger is bounded by the requested tolerance;*
4. *E records the selected endpoint enclosures and the finite* RegSeqRat$^{\uparrow}$ *handoff rows read from those selected intervals;*
5. *H is the* Real$^{\uparrow}$ *seal-facing surface obtained by routing the selected regular-rational rows through the displayed endpoint and width ledgers;*
6. *C stores the* hsame *and* Cont *transports for the source chain, width or modulus rows, diagonal selector, handoff rows, and seal-facing rows;*
7. *P is the* Pkg *provenance naming* NestedInterval$^{\uparrow}$, NestedDyadicInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RegularityModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *source certificates;*

8. Q is the local $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *row for the finite intersection readback name.*

The carrier predicate is $\mathsf{NestedIntervalIntersectionPacket}(X)$*. Its public rows are exactly the displayed source-chain, width, diagonal-selector, endpoint-handoff, seal-facing, transport, package, and local certificate rows.*

Lean4 ✓ `BEDC.Derived.NestedIntervalIntersectionUp.NestedIntervalIntersectionTasteGate single carrier alignment`

Definition 389.2 (Nested interval intersection classifier)**.** *Two accepted* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *carriers classify together when their source nested-interval rows transport to the same displayed finite window and their diagonal selectors choose endpoint enclosures with the same requested tolerance rows.*

Concretely, after the hsame *and* Cont *transports recorded in C and C', the classifier compares the selected lower endpoint rows, selected upper endpoint rows, displayed width or* $\mathsf{RegularityModulus}^{\uparrow}$ *rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal-facing rows,* Pkg *provenance, and local* NameCert *rows componentwise. It does not quotient infinite chains and does not identify a selected point in the intersection.*

Theorem 389.3 (Nested interval intersection NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *surface has five obligations:*

1. *carrier: formation of the finite* BHist *packet of Definition 389.1;*
2. *classifier: the finite-window classifier of Definition 389.2;*
3. *diagonal stability: the selected interval row remains accepted when the source* $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *chain, width or* $\mathsf{RegularityModulus}^{\uparrow}$ *row, and endpoint packets are transported through the displayed* hsame *and* Cont *routes;*
4. *handoff exactness: every* $\mathsf{RegSeqRat}^{\uparrow}$ *row read by the carrier is one of the endpoint enclosures selected by the finite diagonal selector;*
5. *seal boundary: the* $\mathsf{Real}^{\uparrow}$ *surface consumes only the selected finite regular-rational rows, package provenance, and local certificate row.*

These obligations are the complete seed certificate surface for finite nested-interval intersection readback.

Proof. Carrier formation is the assembly of the displayed packet. The source chain is already supplied by $\mathsf{NestedInterval}^{\uparrow}$ or $\mathsf{NestedDyadicInterval}^{\uparrow}$, the endpoint packets are $\mathsf{DyadicInterval}^{\uparrow}$ rows, and the selected endpoint reads are $\mathsf{DyadicRatCore}^{\uparrow}$ rows attached to the same finite history.

The classifier is finite because each comparison is made at the requested displayed precision window. After transporting both packets to that window, the selected lower endpoints, selected upper endpoints, width or modulus rows, handoff rows, seal-facing rows, provenance rows, and local certificate rows are compared component by component.

Diagonal stability follows by transporting the source chain and the width surface first, then reading the same requested tolerance row through the transported selector. Since the selector chooses only from displayed intervals, the transported selected row remains a visible row of the transported chain.

Handoff exactness is a projection from the coordinate E: every regular-rational read is attached to one selected endpoint enclosure. The seal boundary is a projection from H, which routes those same finite rows to $\mathsf{Real}^{\uparrow}$ without adding an infinite chain, compactness principle, choice operator, or host real equality. □

Lean4 ✓ `BEDC.Derived.NestedIntervalIntersectionUp.NestedIntervalIntersectionCarrier namecert obligations`

Theorem 389.4 (Nested interval intersection finite diagonal handoff)**.** *For every accepted* NestedIntervalIntersection *carrier and every displayed precision request, the consumer-visible* $\mathsf{RegSeqRat}^{\uparrow}$ *read factors through exactly one selected finite interval row of the source* $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *chain, its* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint packet, its lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, and the displayed width or* $\mathsf{RegularityModulus}^{\uparrow}$ *witness that justifies the request.*

The handoff exports no completed intersection point, no hidden tail of the chain, and no host completeness datum.

Proof. Project the selector coordinate D from Definition 389.1. A precision request is answered by a visible interval row together with its displayed width or regularity-modulus witness.

Read the selected interval through the source chain coordinate N. Its lower and upper endpoints are the $\mathsf{DyadicRatCore}^{\uparrow}$ rows of the selected $\mathsf{DyadicInterval}^{\uparrow}$ packet, and the width or modulus row is the only bound supplied to the consumer at that request.

Now project the handoff coordinate E. The $\mathsf{RegSeqRat}^{\uparrow}$ row is obtained by repacking the selected finite endpoint enclosure and the displayed bound. Since E has no coordinate for an infinite tail or a completed point, every consumer read factors through the selected finite interval row just described. □

Lean4 ✓ `BEDC.Derived.NestedIntervalIntersectionUp.NestedIntervalIntersectionCarrier finite diagonal handoff`

Theorem 389.5 (Nested interval intersection selector transport determinacy)**.** *If two accepted* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *carriers classify together, their transported diagonal selectors choose the same visible interval row at each displayed precision window. Hence the selected endpoint packets, the displayed width or* $\mathsf{RegularityModulus}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing row are determined by classifier-visible data.*

In particular, selector transport cannot exchange the finite diagonal row for an unrecorded chain tail, a chosen intersection point, host real equality, or an ambient completeness witness.

Lean4 ✓ `BEDC.Derived.NestedIntervalIntersectionUp.NestedIntervalIntersectionCarrier selector transport determinacy`

Proof. Use the classifier definition to transport both carriers to the same displayed precision window. At that window, the source chain rows, endpoint packets, width or modulus rows, handoff rows, seal-facing rows, package provenance, and local certificate rows are compared componentwise.

Project the selector coordinate D from each transported carrier. Since the classifier compares the selected endpoint enclosures at the same requested tolerance, both selectors point to the same visible interval row of the transported finite window.

Apply Definition 389.4 to that selected row. The consumer-visible $\mathsf{RegSeqRat}^{\uparrow}$ handoff is then forced to factor through the same endpoint packet and the same displayed width or regularity-modulus witness.

The seal-facing coordinate is read from this handoff and the displayed transports. No remaining carrier coordinate contains a hidden chain tail, chosen intersection element, host equality row, or completeness datum. □

Theorem 389.6 (Nested interval intersection real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$*-facing boundary of an accepted* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *carrier is exactly the finite diagonal handoff surface of*

Definition 389.4, together with the displayed hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *row.*

In particular, the boundary does not assert Cantor nested-interval convergence, open-cover compactness, extraction of a point by choice, uniqueness of an intersection element, or equality with a host real value.

Proof. Start with the seal-facing coordinate H in Definition 389.1. Its inputs are the selected regular-rational rows from E, the transports in C, the package provenance P, and the local certificate row Q.

Use Definition 389.4 to identify the regular-rational inputs. Each such input has already factored through one visible interval row, its endpoint packet, and its displayed width or modulus witness.

The remaining coordinates are bookkeeping rows: hsame transports, Cont routes, Pkg provenance, and the local naming certificate. None of them contains a compactness theorem, a point-selection operator, a uniqueness claim, or a host real equality row. Therefore the real-seal boundary is precisely the finite surface stated above. □

Theorem 389.7 (Nested interval intersection public export)**.** *The public* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *interface exports exactly the finite diagonal selector, the selected endpoint enclosure, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, the* $\mathsf{Real}^{\uparrow}$ *seal-facing boundary, the displayed* hsame *and* Cont *transports, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *row named in Definition 389.4 and Definition 389.6.*

Every public consumer read of the packet therefore factors through the same selected finite interval row and cannot read a hidden chain tail, a chosen intersection point, open-cover compactness, host real equality, or ambient completeness.

Proof. Start from the finite diagonal handoff. By Definition 389.4, a public precision request exposes one selected interval row, its endpoint packet, its dyadic endpoint reads, and the displayed width or regularity witness for that request.

Pass next through the real-seal boundary. By Definition 389.6, the only rows added at the public boundary are the displayed transports, replay routes, provenance, and local naming row.

No carrier coordinate remains outside these two surfaces. The exported interface is therefore exactly the finite diagonal route plus its seal-facing bookkeeping, with no point-selection, compactness, host equality, or completeness datum. □

Theorem 389.8 (Nested interval dyadic selector coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer read of an accepted* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *packet is covered by one selected finite interval row. The selected row has lower and upper endpoint reads in* $\mathsf{DyadicRatCore}^{\uparrow}$*, its displayed width or* $\mathsf{RegularityModulus}^{\uparrow}$ *row supplies the requested* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and its provenance is exactly the recorded* Pkg *and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *route transported through the visible* hsame *and* Cont *rows.*

Consequently a diagonal-limit or completion consumer reads a dyadic endpoint enclosure selected by the finite selector, not a hidden chain tail, a chosen intersection point, or an ambient compactness witness.

Proof. Begin with the consumer-visible regular-rational read. By Definition 389.4, the read factors through exactly one visible interval row of the source chain, its endpoint packet, its lower and upper dyadic endpoint rows, and its displayed width or modulus witness.

Transport the selector data next. The determinacy result Definition 389.5 says that classified packets choose the same visible row at the same precision window, so the lower endpoint, upper endpoint, width row, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff remain tied to that selected interval.

Finally apply Definition 389.6. The $\mathsf{Real}^{\uparrow}$ boundary adds only the displayed hsame rows, Cont routes, Pkg provenance, and local naming row to the finite diagonal handoff.

The carrier has no coordinate for a completed point, open-cover compactness, choice, or host real equality. Hence every $\mathsf{Real}^{\uparrow}$-facing read is covered by the selected dyadic endpoint row and its recorded provenance. □

Theorem 389.9 (Nested interval intersection width-window refinement)**.** *For any accepted* NestedIntervalIntersectio *carrier*

$$X = (N, W, D, E, H, C, P, Q),$$

every consumer precision request refines through one displayed width row or $\mathsf{RegularityModulus}^{\uparrow}$ *row W, one selected interval row from the diagonal selector D, its* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint packet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row E, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary H. This refinement is transported only by the displayed* hsame *and* Cont *rows recorded in C, together with the package provenance P and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *row Q.*

Thus the width-window row is the exact finite surface consumed by diagonal-limit and completion-facing handoffs; it is not a claim of a chosen intersection point, open-cover compactness, host real equality, or ambient completeness.

Proof. Begin with Definition 389.4. It identifies the consumer-visible $\mathsf{RegSeqRat}^{\uparrow}$ read with one selected interval row, its endpoint packet, and the displayed width or regularity-modulus witness for the requested precision.

Next apply Definition 389.5. The selected interval row remains determined after the hsame and Cont transports, so the endpoint packet and width-window witness cannot be exchanged for an unrecorded chain tail.

Use Definition 389.6 to route the handoff into the $\mathsf{Real}^{\uparrow}$ surface. The seal sees the same finite diagonal package plus the displayed transports, provenance, and local naming row.

Finally Definition 389.8 covers every $\mathsf{Real}^{\uparrow}$-facing consumer read by the selected dyadic endpoint row. Therefore the width-window refinement is the reusable finite surface for the diagonal handoff. □

Theorem 389.10 (Nested interval intersection width-tail Cauchy handoff)**.** *Let*

$$X = (N, W, D, E, H, C, P, Q)$$

be an accepted $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *carrier. If the displayed width rows or* $\mathsf{RegularityModulus}^{\uparrow}$ *rows in W are cofinal for the requested tolerance schedule, then the selected endpoint enclosures determined by the diagonal selector D form a* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible Cauchy handoff consumed by the same* $\mathsf{Real}^{\uparrow}$ *seal boundary H.*

This handoff is carried only by the source chain N, the width or regularity ledger W, the finite selector D, the endpoint handoff rows E, the seal boundary H, the displayed transports C, the package provenance P, and the local $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$ *row Q. It asserts no chosen intersection point, open-cover compactness, host real equality, or ambient completeness.*

Proof. Begin with Definition 389.4. It identifies each requested precision with one selected interval row, its endpoint packet, and the displayed width or regularity witness.

Use Definition 389.5 to keep the selected endpoint enclosures fixed under the displayed transports. The selector therefore supplies one transported endpoint enclosure for each requested tolerance window.

Apply Definition 389.8 and Definition 389.9. Cofinality of the width or regularity rows makes the selected endpoint enclosures compatible with the requested Cauchy tolerance schedule, so the handoff is a $\mathsf{RegSeqRat}^{\uparrow}$-compatible tail surface.

Finally route the handoff through Definition 389.6. The $\mathsf{Real}^{\uparrow}$ seal reads the same finite endpoint, width, transport, package, and local certificate rows, and the carrier has no coordinate for point extraction, compactness, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalIntersection}^{\uparrow}$ packet is generated from a finite $\mathsf{NestedInterval}^{\uparrow}$ or $\mathsf{NestedDyadicInterval}^{\uparrow}$ chain, displayed $\mathsf{DyadicInterval}^{\uparrow}$ endpoint packets, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint reads, width or $\mathsf{RegularityModulus}^{\uparrow}$ rows, a finite diagonal selector, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, a $\mathsf{Real}^{\uparrow}$ seal-facing surface, displayed hsame and Cont transports, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public scope registers the carrier, finite-window classifier, five-row naming-certificate obligation surface, finite diagonal handoff, real-seal boundary, and public export theorem for nested interval intersection readback over $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RegularityModulus}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalIntersection}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.NestedIntervalIntersectionUp.NestedIntervalIntersectionCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Cantor nested-interval convergence, open-cover compactness, point extraction, uniqueness of an intersection element, or host real equality.

Upgrade path: No further paper-theory grade remains; the next formal-axis improvement is a checked standard bridge when available.

Chapter 390

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$

The $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ interface names the finite packet that turns the diagonal selector of Definition 9238.3 into a displayed completion-facing certificate. It sits over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet records the precision request, the explicit modulus value selected from the regular Cauchy input, the finite stream window read at that value, the rational readback, the dyadic observation ledger, and the real-name seal row through which downstream consumers inspect the candidate limit.

Definition 390.1 (RegularCauchyLimitModulus carrier)**.** *A* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \mu, n, k, w, r, d, e, P, N)$$

with the following visible rows:

1. *X is a* $\mathsf{RegSeqRat}^{\uparrow}$*-indexed regular Cauchy family, read as the displayed input of Definition 9238.2;*
2. *μ is the explicit modulus row carried by that input;*
3. *n is the unary precision request;*
4. *$k = \mu(1/(n+1))$ is the selected diagonal threshold row;*
5. *w is the* $\mathsf{StreamName}^{\uparrow}$ *window that reads the k-th regular rational sequence at the diagonal index $k + n + 1$;*
6. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the rational observation selected by w;*
7. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the displayed endpoint, radius, comparison, and precision rows used by the readback;*
8. *e is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row that exposes the same observation as a row of the candidate diagonal real name;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for an external limiting real, quotient stream equality, a hidden tail choice, or a host-level selector. Every consumer-visible value is one of the displayed rows of M.

Lean4 (definition) `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusPacket`

Definition 390.2 (RegularCauchyLimitModulus classifier)**.** *For accepted carriers*

$$M = (X, \mu, n, k, w, r, d, e, P, N) \quad \textit{and} \quad M' = (X', \mu', n', k', w', r', d', e', P', N'),$$

the classifier RegularCauchyLimitModulusClassifier(M, M') *compares only the displayed modulus rows, stream windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *ledgers,* Real$^{\uparrow}$ *seal rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The classifier is therefore a finite packet comparison: it neither compares host functions X and X' extensionally nor identifies two real names by a quotient construction.*

Lean4 (definition) `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusClassifier`

Theorem 390.3 (RegularCauchyLimitModulus NameCert obligations)**.** *The local* NameCert$_{\text{RegularCauchyLimitModulus}^{\uparrow}}$ *obligation sketch consists of five finite rows:*

1. *carrier habitation: every displayed regular Cauchy input with modulus μ and unary precision request n yields the finite packet of Definition 390.1;*
2. *modulus selector totality: the row $k = \mu(1/(n+1))$ is read from the displayed modulus coordinate and has a unary realization available to the stream-window reader;*
3. *diagonal readback stability: transporting the input family, modulus row, and precision request through the classifier transports the selected window w and readback r to the corresponding selected window and readback of the transported packet;*
4. Real$^{\uparrow}$ *seal non-escape: the seal-facing row e is reached only from the same* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *ledger already displayed in the packet;*
5. *ledger exactness: every consumer-visible dyadic observation is exactly the endpoint, radius, comparison, precision, transport, continuation, package, and name row recorded by d, P, and N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusNamecertObligations`

Proof. Carrier habitation projects the regular Cauchy input and its modulus row, then evaluates the modulus at the displayed precision request $1/(n+1)$. This supplies the selected threshold k and the diagonal index $k+n+1$ used by the stream-window row.

The selector row is total because μ is part of the input certificate, not a hidden functional. Reading μ at the requested precision is a finite BEDC operation that produces the unary row consumed by StreamName$^{\uparrow}$.

For stability, transport X, μ, and n by the classifier rows of Definition 390.2. The selected k, the window w, and the readback r are then reconstructed from the transported rows by the same displayed Cont routes.

The Real$^{\uparrow}$ boundary consumes only the readback and dyadic ledger already present in r and d. Repacking them with e, P, and N gives the local naming row and leaves no coordinate for a quotient representative or external limit selector. □

Theorem 390.4 (RegularCauchyLimitModulus diagonal totality)**.** *Let $M = (X, \mu, n, k, w, r, d, e, P, N)$ be an accepted* RegularCauchyLimitModulus$^{\uparrow}$ *carrier. Then the diagonal observation required by Definition 9238.3 is total at the precision request n: the selected threshold row $k = \mu(1/(n+1))$, the stream window w, the* RegSeqRat$^{\uparrow}$ *readback r, and the dyadic ledger d together determine the displayed value*

$$L_n = (X_k)_{k+n+1}$$

as a finite BEDC observation routed through P and N.

The totality statement is internal to the packet. It does not assert completion of $\mathsf{Real}^{\uparrow}$ *and does not choose a limit outside the displayed carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusPacket diagonal totality`

Proof. Project the carrier of Definition 390.1. Its modulus selector row fixes k, and its stream-window row fixes the finite read of X_k at the diagonal index $k + n + 1$.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback row r records the rational observation selected by that window. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d records the endpoint and comparison data needed by a dyadic consumer of the same observation.

Since the continuation routes and package provenance are carrier coordinates, the observation is routed through P and N without any additional selection step. This is exactly the finite diagonal read demanded by Definition 9238.3. □

Theorem 390.5 (RegularCauchyLimitModulus tail-modulus extraction)**.** *Let $M = (X, \mu, n, k, w, r, d, e, P, N)$ be an accepted* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packet. The selected threshold row $k = \mu(1/(n + 1))$, the finite* $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d form the finite-window* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff consumed by downstream regular-Cauchy and* $\mathsf{Real}^{\uparrow}$ *readers.*

The handoff factors through chapter 292, chapter 126, chapter 301, chapter 291, and chapter 38. It exports no host selector, quotient stream equality, external limit, or ambient completeness principle.

Proof. Project the displayed carrier rows of M. The regular input family X, its modulus row μ, the precision request n, the selected threshold k, the window w, the readback r, the ledger d, the seal row e, the package row P, and the local name row N are all finite coordinates of the same BHist packet.

Use diagonal totality from Definition 390.4. It identifies k as the row $\mu(1/(n+1))$ and identifies w, r, and d as the finite observation route for the diagonal read of X_k at index $k + n + 1$.

The finite-window closure of chapter 292 consumes exactly this thresholded window and readback form. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ chapters provide the window, rational readback, and dyadic ledger factors, so the modulus handoff is already present inside the displayed packet.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 390.6. The seal row e keeps every real-facing read inside the same finite packet, leaving no coordinate for a host selector, quotient stream equality, external limit, or ambient completeness principle. □

Theorem 390.6 (RegularCauchyLimitModulus RealUp seal boundary)**.** *For every accepted* $\mathsf{RegularCauchyLimitMo}$ *carrier $M = (X, \mu, n, k, w, r, d, e, P, N)$, a* $\mathsf{Real}^{\uparrow}$ *consumer can inspect the diagonal-limit modulus packet only through the displayed seal-facing row e, whose source is the selected* $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d. The boundary exports a finite real-name observation row and no host real value, quotient representative, hidden tail selector, or equality of completed limits.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusPacket real seal boundary`

Proof. Begin with the diagonal totality row of Definition 390.4. It identifies the only observation available at precision n as the windowed readback of X_k at index $k+n+1$.

The seal-facing coordinate e records the $\mathsf{Real}^{\uparrow}$ route for that same observation. Its source rows are w, r, and d, and its transport is supplied by the componentwise hsame and Cont coordinates named in the classifier.

Therefore any $\mathsf{Real}^{\uparrow}$ consumer read factors through the finite packet itself. Since the carrier includes no coordinate for an external real, a quotient representative, a hidden tail selector, or equality of completed limits, none of those objects can escape through the seal boundary. □

Theorem 390.7 (RegularCauchyLimitModulus seal factorization)**.** *Let $M = (X, \mu, n, k, w, r, d, e, P, N)$ be an accepted* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packet. Before a* $\mathsf{Real}^{\uparrow}$ *consumer can read the seal row e, the same packet must expose the* $\mathsf{CauchyModulus}^{\uparrow}$ *finite-window route, the* $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d. In particular, the row e is visible only after the displayed threshold $k = \mu(1/(n+1))$ has been consumed as the modulus row of chapter 292 and routed through the diagonal totality statement of Definition 390.4 and the Real seal boundary of Definition 390.6.*

The seal factorization excludes selected limits, quotient stream equality, hidden host selectors, and ambient completeness. It is a finite handoff through the displayed threshold, window, readback, dyadic ledger, and seal rows.

Proof. Project the carrier M. The selected threshold k, the window w, the readback r, the dyadic ledger d, and the seal row e are all displayed coordinates of the same BHist packet.

Apply diagonal totality from Definition 390.4. It identifies $k = \mu(1/(n+1))$ as the threshold row that schedules the finite diagonal observation and identifies w, r, and d as the observation route.

Now read this route as the finite-window modulus handoff of chapter 292. The threshold ledger is consumed before the stream window, regular rational readback, and dyadic ledger can be passed to a real-facing consumer.

Finally apply the Real seal boundary of Definition 390.6. Since e has only w, r, and d as its source rows, the $\mathsf{Real}^{\uparrow}$ consumer receives no selected limit, quotient stream equality, hidden host selector, or ambient completeness principle. □

Theorem 390.8 (RegularCauchyLimitModulus classifier transport exactness)**.** *Let*

$$M = (X, \mu, n, k, w, r, d, e, P, N) \quad \text{and} \quad M' = (X', \mu', n', k', w', r', d', e', P', N')$$

be accepted $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packets. If* $\mathsf{RegularCauchyLimitModulusClassifier}(M, M')$ *holds, then the selected threshold row k, the stream window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d, the* $\mathsf{Real}^{\uparrow}$ *seal row e, the package provenance P, and the local* $\mathsf{NameCert}$ *row N transport exactly to k', w', r', d', e', P', and N'.*

The transported packet exports only those displayed rows. It exports no host selector, no quotient equality of streams, and no external completed limit.

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitModulusUp.RegularCauchyLimitModulusPacket classifier transport exactness`

Proof. Unfold the carrier and classifier definitions of Definition 390.1 and Definition 390.2. The classifier supplies componentwise hsame rows for the displayed modulus and observation coordinates, displayed Cont routes for the stream-window and readback passages, and the visible Pkg and $\mathsf{NameCert}$ rows for provenance and naming.

Apply diagonal totality from Definition 390.4 to the source packet. The selected threshold is read from the displayed modulus row, and the stream window and $\mathsf{RegSeqRat}^{\uparrow}$ readback are reconstructed from the transported threshold by the same finite continuation routes.

The dyadic ledger and $\mathsf{Real}^{\uparrow}$ seal row then transport along the displayed ledger and seal coordinates. The RealUp seal boundary of Definition 390.6 ensures that a consumer can only inspect the transported observation through e', with source rows w', r', and d'.

Repack the transported threshold, window, readback, dyadic ledger, seal row, package provenance, and local naming row as the target packet M'. Since the carrier has no coordinates for a host selector, quotient stream equality, or completed limit outside the displayed rows, the transport exports none of those objects. □

Theorem 390.9 (RegularCauchyLimitModulus scoped kernel binding)**.** *The carrier, classifier, diagonal totality row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and classifier transport exactness of* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *depend only on the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. In particular, every consumer-facing route through the chapter factors through those rows before it reaches the selected diagonal threshold, the finite stream window, the rational readback, the dyadic ledger, or the real-name seal.*

No host selector, quotient stream equality, external limit, or ambient completeness field enters this scope.

Proof. Start with the carrier of Definition 390.1. Its coordinates are exactly the displayed regular-sequence input, modulus row, precision request, selected threshold, stream window, readback, dyadic ledger, real seal, package row, and local naming row.

The classifier of Definition 390.2 compares those coordinates by the recorded hsame rows and Cont routes. It gives no extra coordinate for a host selector, quotient stream equality, external limit, or ambient completeness field.

Apply diagonal totality from Definition 390.4. The selected observation is reconstructed from the displayed modulus, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

The $\mathsf{Real}^{\uparrow}$ boundary of Definition 390.6 then routes the same observation through the seal row, and Definition 390.8 shows that classifier transport preserves exactly those displayed rows. Thus the kernel scope is exhausted by the listed rows. □

Theorem 390.10 (RegularCauchyLimitModulus upstream lattice factorization)**.** *Every accepted* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packet factors its diagonal-modulus handoff through the four sibling rows chapter 126, chapter 301, chapter 291, and chapter 38 before the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$ *row is consumed. The factorization route is exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{Real}^{\uparrow}$ *seal row over the carrier of Definition 390.1.*

No quotient stream, host selector, selected host limit, or ambient completeness principle enters the factorization.

Proof. Project the accepted carrier. Its coordinates display the regular input and modulus row, the selected threshold, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{Real}^{\uparrow}$ seal row, the package provenance, and the local naming row.

Use diagonal totality from Definition 390.4. The selected threshold, window, and readback route the diagonal observation through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ sibling rows, while the dyadic ledger records the same observation in the $\mathsf{DyadicRatCore}^{\uparrow}$ row.

Use the RealUp seal boundary of Definition 390.6. It forces every real-facing inspection of the diagonal observation through the displayed $\mathsf{Real}^{\uparrow}$ seal row and leaves no coordinate for a host real, a quotient representative, or a selected external limit.

Finally apply classifier transport exactness from Definition 390.8 together with the scoped kernel binding of Definition 390.9. These results show that transport preserves exactly the same window, readback, ledger, seal, package, and naming rows, so the four sibling rows exhaust the upstream lattice before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$ row is consumed. □

Theorem 390.11 (RegularCauchyLimitModulus public RealUp export)**.** *The carrier of Definition 390.1, the classifier of Definition 390.2, and the five* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$ *rows of Definition 390.3 jointly export a public* $\mathsf{Real}^{\uparrow}$*-facing finite completion-modulus handoff. Every consumer read of that handoff factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *input family* X*, the explicit modulus row* μ*, the unary precision request* n*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger* d*, the* $\mathsf{Real}^{\uparrow}$ *seal row* e*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

The exported handoff excludes quotient stream equality, hidden tail choice, host selector, selected host limit, and ambient $\mathsf{Real}^{\uparrow}$ *completeness. It is a public row presentation of the finite diagonal-modulus packet, not a completion theorem.*

Proof. Project the displayed carrier rows. The regular Cauchy input, its modulus row, the precision request, the selected threshold, the stream window, the rational readback, the dyadic ledger, the seal row, provenance, and local name row are all coordinates of the same finite packet.

Use the modulus selector row from Definition 390.3 at the chosen precision threshold. The diagonal totality theorem then identifies the selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback as the finite observation consumed by the handoff.

Next apply classifier transport exactness. It keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows on the displayed diagonal readback route, while the $\mathsf{Real}^{\uparrow}$ seal boundary forces every real-facing read through the seal row e and the dyadic ledger d.

Finally use the seal non-escape and ledger exactness rows of the local naming obligations. Repack the displayed source family, modulus row, request, selected window, readback, ledger, seal, package, and name rows as the public handoff. Since the carrier has no coordinates for quotient stream equality, hidden tail choice, a host selector, a selected host limit, or ambient completeness, none of those objects is exported. □

Theorem 390.12 (RegularCauchyLimitModulus RealUp completion bridge)**.** *The public diagonal-modulus handoff of Definition 390.11 supplies exactly the bridge packet consumed by the* $\mathsf{Real}^{\uparrow}$ *completion interface. The same* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows are preserved, and the bridge exports no quotient real equality, selected host limit, or ambient completeness principle.*

Proof. Start with diagonal totality from Definition 390.4. It selects the displayed threshold, window, readback, and dyadic ledger from the regular input family and its modulus row, all inside the finite carrier.

Apply classifier transport exactness from Definition 390.8. The hsame and Cont routes preserve the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows through the local classifier, while the Pkg and $\mathsf{NameCert}$ rows keep the dependency certificate fixed.

Use the scoped kernel binding of Definition 390.9 and the public export theorem Definition 390.11. Together they identify the public handoff as the completion-facing packet over exactly the displayed rows.

The non-escape clauses in the local naming obligations exclude quotient real equality, selected host limits, and ambient completeness. Hence the $\mathsf{Real}^{\uparrow}$ completion interface consumes the public

diagonal-modulus handoff as a bridge packet without receiving any additional real-completeness datum. □

Theorem 390.13 (RegularCauchyLimitModulus tail-selector stability)**.** *Transporting the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *input family, its modulus row, and a unary precision request transports the selected threshold, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row of an accepted* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packet. The transported packet keeps the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$ *provenance and introduces no host tail selector.*

Thus selector stability is a finite classifier fact over the displayed input, modulus, request, threshold, window, readback, ledger, and seal rows.

Proof. Begin with diagonal totality. The selected threshold is read from the displayed modulus row at the displayed unary precision request, and the finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback are reconstructed from that threshold.

Apply classifier transport exactness to the transported input family, modulus row, and request. It carries the threshold, window, readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger through the same hsame and Cont routes.

Finally use the $\mathsf{Real}^{\uparrow}$ seal boundary and scoped kernel binding. The seal row transports with the same finite packet, and the carrier contains no coordinate for quotient stream equality, a hidden tail choice, or an ambient selector. □

Theorem 390.14 (RegularCauchyLimitModulus RealUp seal handoff)**.** *The* $\mathsf{Real}^{\uparrow}$ *completion-facing handoff exported by an accepted* $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ *packet is exactly the transported finite packet of Definition 390.13: displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *input, modulus row, unary request, selected threshold,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* Pkg *provenance, and local* NameCert *row.*

It is not quotient stream equality, not a hidden tail choice, and not a selected host limit. The completion interface receives only this finite seal-facing packet.

Proof. Use tail-selector stability to obtain the transported threshold, stream window, rational readback, dyadic ledger, and real seal row from the displayed input, modulus row, and precision request.

Next apply the $\mathsf{Real}^{\uparrow}$ seal boundary. Every real-facing read must pass through the transported seal row together with the dyadic ledger and the displayed source rows.

Finally combine scoped kernel binding with the RealUp completion bridge. The bridge consumes exactly the public diagonal-modulus packet over the displayed rows, so the handoff has no additional quotient-stream equality, hidden tail choice, selected host limit, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyLimitModulus}^{\uparrow}$ packet is generated by a displayed regular Cauchy family, its explicit modulus row, a unary precision request, the selected diagonal threshold, a finite $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, a $\mathsf{Real}^{\uparrow}$ seal-facing row, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 390.1, Definition 390.2, Definition 390.4, Definition 390.6, Definition 390.8, and Definition 390.9 bind the finite diagonal-modulus packet to the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, a public standard bridge, or checked Lean implementation.

Upgrade path: The next paper grade is a bridge theorem from the public diagonal-modulus handoff to the Real$^{\uparrow}$ completion interface over the same displayed RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 391

A Concrete Naming Certificate for $\mathsf{UniformContinuity}^{\uparrow}$

The $\mathsf{UniformContinuity}^{\uparrow}$ surface materializes the compact-uniform-continuity capstone of chapter 9237 as an ordinary concrete instance. It consumes the finite-net certificate of $\mathsf{Compact}^{\uparrow}$, the located distance rows of $\mathsf{Met}^{\uparrow}$, the pointwise modulus ledger of $\mathsf{Continuous}^{\uparrow}$, the rational lower-bound ledger of $\mathsf{Rat}^{\uparrow}$, and the closed $\mathsf{ProbeBundle}$ spine. The exported object is a tolerance-only uniform-modulus certificate; it is not a host-level modulus selector.

The dependency rows are read from sibling concrete certificates: the finite-net row from chapter 60, the source and target distance rows from chapter 57, the pointwise graph and modulus rows from chapter 59, and the lower-bound ledger from chapter 37.

Definition 391.1 (UniformContinuity BHist carrier)**.** *A* $\mathsf{UniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, f, \varepsilon, P, C, R, T, L, M, H, N)$$

with the following visible rows:

1. *X is the source metric row consumed through* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$*;*
2. *Y is the target metric row consumed through* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$*;*
3. *f is the graph row consumed through* $\mathsf{NameCert}_{\mathsf{Continuous}^{\uparrow}}$*;*
4. *ε is a positive tolerance row consumed through* $\mathsf{NameCert}_{\mathsf{Rat}^{\uparrow}}$*;*
5. *P is the finite* $\mathsf{ProbeBundle}$ *row supplied by the compact certificate;*
6. *C lists the center histories indexed by the bundled probes;*
7. *R lists the local positive radius and pointwise precision histories;*
8. *T records the half-radius coverage rows from* $\mathsf{NameCert}_{\mathsf{Compact}^{\uparrow}}$*;*
9. *L records the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the bundled radius rows;*
10. *M records the metric triangle transports in the source and target metrics;*
11. *H records* hsame *transport for the displayed histories;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$ *row.*

The packet is accepted only when these rows are tied by Cont *routes and* Pkg *provenance to the listed dependency certificates. No coordinate stores a function from tolerances to radii outside* BHist*.*

Lean4 (definition) `BEDC.Derived.UniformContinuityUp.UniformContinuityCarrier`

Definition 391.2 (UniformContinuity classifier)**.** *Two* $\mathsf{UniformContinuity}^{\uparrow}$ *packets*

$$U = (X, Y, f, \varepsilon, P, C, R, T, L, M, H, N)$$

and

$$U' = (X', Y', f', \varepsilon', P', C', R', T', L', M', H', N')$$

are classified together when the source and target metric rows are transported by their $\mathsf{Met}^{\uparrow}$ *classifiers, the graph row is transported by the* $\mathsf{Continuous}^{\uparrow}$ *classifier, the tolerance and folded precision rows are transported by the* $\mathsf{Rat}^{\uparrow}$ *classifier, the* ProbeBundle *membership spine is the same displayed finite spine after* hsame *transport, and the coverage, pointwise-modulus, lower-bound, and triangle rows are the corresponding transported rows. The classifier compares exactly the displayed certificate packet and does not compare host functions, external compact covers, or selected moduli.*

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.TasteGate.UniformContinuityTasteGate single carrier alignment`

Theorem 391.3 (Finite-net modulus fold)**.** *Let* U *be an accepted* $\mathsf{UniformContinuity}^{\uparrow}$ *carrier. The finite* ProbeBundle *row* P*, the pointwise-modulus rows* R*, and the* $\mathsf{Rat}^{\uparrow}$ *lower-bound ledger* L *determine a folded positive precision history* δ_ε *such that, for every bundled probe* p*, the fold certifies* δ_ε *below the half-radius row attached to* p*. The same folded history is stable under the classifier of Definition 391.2. Therefore the uniform precision read by a consumer is a BEDC history extracted from the finite bundle, not an external selection.*

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.UniformContinuityPacket finite net modulus fold`

Proof. Project the finite ProbeBundle spine from the carrier. Its center rows, local radius rows, and pointwise precision rows are displayed coordinates of U, and each row is attached to the compact and continuous dependency certificates by Cont and Pkg provenance.

Run primitive recursion over the bundle. At the empty boundary the accepted carrier supplies the nonempty compact-net witness, so the fold starts from a displayed positive radius row. At each successor probe, the $\mathsf{Rat}^{\uparrow}$ comparison row selects the certified lower of the accumulator and the next half-radius row, and stores the selected row back in L.

Classifier stability follows coordinatewise. Transporting U to U' moves the rational rows through the $\mathsf{Rat}^{\uparrow}$ classifier and moves the bundle membership witnesses through the same displayed finite spine. The recursion therefore reads the transported lower-bound ledger, so the folded precision history is classified with its transported representative. □

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.UniformContinuityCarrier finite net modulus fold`

Theorem 391.4 (UniformContinuity NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$ *obligation surface consists exactly of the following five rows:*

1. *the finite-net carrier row exposing the compact* ProbeBundle*, its center histories, and half-radius coverage witnesses;*
2. *the pointwise-to-uniform modulus fold row of Definition 391.3;*
3. *the* $\mathsf{Rat}^{\uparrow}$ *lower-bound ledger certifying the folded positive precision below every bundled half-radius row;*

4. *the metric triangle transport row carrying the source estimate through a chosen center and then carrying the two target estimates to the final ε bound;*
5. *the non-escape row saying that every consumer-visible modulus read factors through the displayed compact, continuous, rational, metric,* ProbeBundle*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$ *rows.*

Consequently the public certificate exports a tolerance-only uniform-modulus ledger and exports no host-level modulus selector, no unlisted open cover, no sequential compactness argument, and no chosen subsequence.

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.UniformContinuityPacket namecert obligation surface`

Proof. The finite-net row is the compact certificate projected to the displayed ProbeBundle, center histories, and half-radius coverage witnesses. Those rows are already part of the carrier, so the local packet has no hidden cover coordinate.

The pointwise-to-uniform row is the finite fold of Definition 391.3. Its output is a positive $\mathsf{Rat}^{\uparrow}$ history below every bundled half-radius row, and the classifier stability of that theorem gives the corresponding transport field for the local NameCert packet.

The metric row applies the source triangle inequality around the selected center, then applies the target triangle inequality to the two pointwise estimates produced by the continuous certificate. The rational ledger supplies the displayed comparison rows needed to keep the source distances inside the local pointwise radii.

Finally inspect the carrier coordinates of Definition 391.1. They list all dependency certificates, all finite bundle rows, all rational comparison rows, all metric transport rows, and the local naming row. A consumer-visible read must factor through one of these coordinates, so there is no readable slot for an external modulus selector, sequential compactness witness, or host cover. □

Theorem 391.5 (Compact continuous standard reading)**.** *When a classical compact metric source and a continuous graph are presented through* $\mathsf{NameCert}_{\mathsf{Compact}^{\uparrow}}$*,* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$*, and* $\mathsf{NameCert}_{\mathsf{Continuous}^{\uparrow}}$*, the accepted* $\mathsf{UniformContinuity}^{\uparrow}$ *packet reads as the usual statement that a continuous function on a compact metric space is uniformly continuous. The BEDC reading is the finite certificate surface: for each positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε, the packet returns a folded positive* $\mathsf{Rat}^{\uparrow}$ *precision history δ_ε whose validity is witnessed by finite-net coverage, pointwise continuity at bundled centers, rational lower-bound comparison, and metric triangle transport.*

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.UniformContinuityPacket compact continuous standard reading`

Proof. Start with the compact certificate. It supplies the finite ProbeBundle net and the half-radius coverage witnesses for the chosen tolerance scale. The continuous certificate supplies pointwise modulus rows at the bundled centers.

Apply Definition 391.3 to extract the single folded precision history. The fold uses only the displayed rational ledger and is stable under the packet classifier, so it is the public precision read by $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$.

For any two accepted source histories whose source distance is below this precision, choose the covering bundled center for the first history. The source triangle row places the second history within the same local pointwise radius, and the target triangle row combines the two pointwise estimates into the displayed tolerance bound. This is precisely the classical Heine–Cantor reading, expressed through finite BEDC certificate rows. □

Theorem 391.6 (UniformContinuity modulus non-escape)**.** *Every consumer-visible modulus row of an accepted* $\mathsf{UniformContinuity}^{\uparrow}$ *packet factors through the compact* $\mathsf{ProbeBundle}$ *row, the* $\mathsf{Continuous}^{\uparrow}$ *pointwise rows, the* $\mathsf{Rat}^{\uparrow}$ *lower-bound ledger, the* $\mathsf{Met}^{\uparrow}$ *triangle rows, and the local* $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$ *row of Definition 391.1, with the folded precision supplied by Definition 391.3. The packet exports no host function from tolerances to radii, no open-cover selection, no sequential compactness witness, and no choice-selected center.*

Lean4 ✓ `BEDC.Derived.UniformContinuityUp.UniformContinuityPacket modulus non escape`

Proof. Project the carrier coordinates of Definition 391.1. The only modulus-facing data are the compact $\mathsf{ProbeBundle}$ row P, the center and pointwise rows C and R, the compact coverage row T, the rational lower-bound ledger L, the metric triangle row M, and the local naming row N.

Apply the finite-net fold of Definition 391.3. It constructs the folded precision from the displayed bundle, pointwise rows, and rational ledger, and its classifier stability keeps that folded row inside the same transported packet.

The obligation surface of Definition 391.4 identifies this folded row as the local consumer-visible modulus route. The metric rows then use exactly the displayed triangle transports to turn the folded source precision into the target tolerance estimate.

Inspecting the carrier shape completes the non-escape claim. No coordinate stores a host tolerance-to-radius function, an open-cover selector, a sequential compactness witness, or a choice-selected center, so none of those objects can be exported by a modulus read. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformContinuity}^{\uparrow}$ packet is generated from $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, hsame, Cont, Pkg, and local NameCert rows that display finite-net coverage, pointwise control, rational lower bounds, metric triangle routing, and modulus readback.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public surface is the compact-continuous modulus export binding the compact ProbeBundle, $\mathsf{Continuous}^{\uparrow}$ pointwise rows, $\mathsf{Rat}^{\uparrow}$ lower-bound ledger, $\mathsf{Met}^{\uparrow}$ triangle rows, hsame, Cont, Pkg, and local NameCert dependencies for the displayed finite-net modulus packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host modulus selectors, open-cover compactness, sequential compactness, choice-selected centers, or bridge checking.

Upgrade path: No further paper-side theory-closure upgrade remains; formal and bridge checking remain separate axes.

Chapter 392

A Concrete Naming Certificate for $\mathsf{CauchyCriterion}^{\uparrow}$

The $\mathsf{CauchyCriterion}^{\uparrow}$ interface records the explicit constructive Cauchy criterion used by Bishop and Bridges, *Constructive Analysis*, when a real-facing packet is accepted only after a displayed dyadic tail budget has been checked. It is a finite consumer object over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$: a precision request is read through an explicit modulus, a dyadic tolerance row, a finite tail-bound ledger, a regular rational handoff, and finally a real seal. The certificate does not form quotient streams, identify host real equality, select a limit, or appeal to ambient completeness.

The precision route consumes the sibling carrier rows of chapter 292, chapter 379, chapter 291, and chapter 301 before reaching the $\mathsf{Real}^{\uparrow}$ seal. Thus M, T, Q, and W are not local free parameters: they are imported sibling rows whose finite route is repacked by the $\mathsf{CauchyCriterion}^{\uparrow}$ carrier.

Definition 392.1 (CauchyCriterion carrier)**.** *A* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, M, T, Q, E, H, C, P, N)$$

with the following visible rows:

1. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source window being tested, including the finite rational observations that the consumer may inspect;*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule that assigns to each displayed precision request the tail threshold used by the test;*
3. *T is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger recording the finite comparisons between tail observations beyond the threshold selected by M;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose dyadic budget is used by every comparison in T;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational handoff has been accepted;*
6. *H records the componentwise* hsame *transports for the source window, modulus schedule, tail ledger, dyadic tolerance, regular-sequence handoff, and seal rows;*
7. *C records the displayed* Cont *routes by which a consumer reads the modulus, tolerance, finite tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal;*
8. *P is the* Pkg *provenance naming the* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source certificates;*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^\uparrow}$ *provenance row.*

The classifier compares two accepted carriers only by these displayed rows, their componentwise hsame *transports, their* Cont *read routes, their* Pkg *provenance, and their local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier visible packet terminal closure`

Theorem 392.2 (CauchyCriterion tail-bound RealUp seal factorization)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^\uparrow$ *carrier. Every* $\mathsf{Real}^\uparrow$ *seal read exported from K factors through the displayed Cauchy criterion route:*

$$\mathsf{precision} \longmapsto M(\mathsf{precision}) \longmapsto Q \longmapsto T \longmapsto W \longmapsto E.$$

Equivalently, the consumer must pass through the $\mathsf{CauchyModulus}^\uparrow$ *schedule, the* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance row, the finite* $\mathsf{DyadicTailBound}^\uparrow$ *ledger, the* $\mathsf{RegSeqRat}^\uparrow$ *source handoff, and the* $\mathsf{Real}^\uparrow$ *seal. No exported read may use a quotient stream, host real equality, a selected limit, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail bound real seal`

Proof. Project the carrier of Definition 392.1. The consumer precision request first reaches the modulus row M, which supplies the displayed tail threshold for the requested precision.

The tolerance coordinate Q and tail ledger T are stored beside that threshold. Reading T therefore uses exactly the dyadic budget named by Q and the finite tail entries selected by M, not an unrecorded rational radius or an external metric estimate.

The source window W is the regular rational handoff for the same inspected tail. Transport along the recorded hsame comparisons and replay the Cont routes in C to keep the modulus, tolerance, ledger, and handoff rows in one visible BHist packet.

Finally the seal coordinate E is reached only after this regular-sequence handoff. The package row P and local certificate row N record the dependency certificates used by the route, and the carrier has no coordinate for quotient streams, host real equality, selected limits, or ambient completeness. Hence every seal read factors through the displayed criterion route. □

Theorem 392.3 (CauchyCriterion NameCert obligation sketch)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^\uparrow}$ *sketch has five rows:*

1. *carrier habitation: the packet of Definition 392.1 stores a* $\mathsf{RegSeqRat}^\uparrow$ *source window, a* $\mathsf{CauchyModulus}^\uparrow$ *schedule, a* $\mathsf{DyadicTailBound}^\uparrow$ *ledger, a* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance row, a* $\mathsf{Real}^\uparrow$ *seal, transport routes, package provenance, and local certificate provenance;*
2. *classifier transport: accepted packets are compared only by componentwise* hsame *and displayed* Cont *routes over the source window, modulus, tolerance, tail ledger, handoff, seal,* Pkg*, and* NameCert *rows;*
3. *tail-budget exactness: every accepted tail comparison uses the threshold supplied by M and the tolerance supplied by Q in the same carrier;*
4. $\mathsf{RegSeqRat}^\uparrow$ *handoff: the finite tail ledger repacks as the visible regular rational source window W before any real-facing seal read is exposed;*
5. $\mathsf{Real}^\uparrow$ *seal non-escape: Definition 392.2 exhausts every exported seal read and withholds quotient streams, host real equality, selected limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier namecert obligations`

Proof. The carrier and classifier rows are direct projections of the displayed packet. Since the classifier reads only the named coordinates, componentwise hsame transport and the recorded Cont routes preserve the accepted carrier shape.

Tail-budget exactness is read from the paired coordinates M, Q, and T. The threshold and tolerance used by a comparison are the rows displayed in the same finite packet, so the criterion cannot substitute hidden bounds.

The regular-sequence handoff is the projection to W after the finite tail ledger has been checked. The handoff keeps the inspected observations, threshold, tolerance, and ledger evidence visible to $\mathsf{RegSeqRat}^{\uparrow}$ consumers.

The seal row is supplied by Definition 392.2. It is reached only through the displayed criterion route, so the certificate exports no quotient, host-real, selected-limit, or completeness datum. □

Theorem 392.4 (CauchyCriterion RealObservationBudget tail-budget lock)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *read use the same precision request, the same finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source window, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row as the readback of K. Then the observation-budget read factors through the finite CauchyCriterion route $M, Q, T, W, E, H, C, P, N$ before any Real-facing seal is public.*

Consequently every such budget read sees the displayed modulus threshold, dyadic tolerance, finite tail ledger, regular rational handoff, real seal, transport, continuation, package, and local naming rows. It receives no quotient stream, host real equality, selected limit, hidden family limit, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail budget real observation lock`

Proof. First read the CauchyCriterion seal route. By Definition 392.2, any Real-facing seal exposed by K is reached only after the precision request has passed through the modulus row M, dyadic tolerance row Q, finite tail ledger T, regular rational source row W, and seal row E.

Next read the local naming surface. Definition 392.3 records tail-budget exactness, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and seal non-escape as rows of the same local NameCert, so the shared precision request, source window, and tolerance row cannot be detached from H, C, P, N.

Finally compose with the observation-budget consumer exactness row of Definition 611.9. Since that consumer reads only its displayed precision, window, dyadic, regular handoff, seal, transport, route, provenance, and naming rows, the shared rows force the budget read to pass through the same $M, Q, T, W, E, H, C, P, N$ route before the seal is public. The non-escape clauses on both packets exclude quotient streams, host real equality, selected limits, hidden family limits, and ambient completeness. □

Theorem 392.5 (CauchyCriterion modulus threshold stability)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ and $K' = (W', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carriers. Suppose the classifier relates them by the componentwise* hsame *rows and displayed* Cont *routes for the source window, modulus schedule, tail ledger, dyadic tolerance, regular-sequence handoff, seal row, package provenance, and local* NameCert *provenance. Then every precision request read from either carrier has the same displayed threshold coordinate selected by the transported modulus row and the same dyadic tolerance coordinate selected by the transported Q row. In particular, componentwise transport of the $W, M, T, Q, E, H, C, P, N$ packet preserves the threshold chosen by M and the dyadic tolerance row read by Q.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier modulus threshold stability`

Proof. Project both carriers using Definition 392.1. The only modulus coordinate visible to the classifier is M on the first packet and M' on the transported packet, and the only tolerance coordinate visible to the classifier is Q and Q'.

The hsame rows in H and H' identify the corresponding coordinates, while the Cont routes in C and C' describe how a consumer reaches them. Since the classifier has no access to any further modulus or tolerance carrier, a transported precision request can only be evaluated through the displayed modulus row and the displayed dyadic tolerance row.

The obligation sketch of Definition 392.3 makes this the classifier-transport row for the local NameCert. Repacking those projections shows that the selected threshold and tolerance survive transport unchanged as named rows of the accepted BHist packet. □

Theorem 392.6 (CauchyCriterion dyadic tail ledger exactness)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* CauchyCriterion$^{\uparrow}$ *carrier. Every accepted tail comparison recorded in T is read only through the threshold displayed by M, the dyadic tolerance displayed by Q, the finite* DyadicTailBound$^{\uparrow}$ *ledger row T, and the* DyadicRatCore$^{\uparrow}$ *tolerance certificate named in P. No comparison in T may be justified by an undisplayed modulus threshold, an ambient rational radius, a host-real metric estimate, or a tail-bound certificate outside the same carrier.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier dyadic tail ledger exactness`

Proof. Read the tail comparison from the ledger coordinate T of Definition 392.1. The carrier stores T beside the modulus row M and the dyadic tolerance row Q, so the comparison is indexed by the displayed threshold and the displayed dyadic budget.

The package row P names the DyadicTailBound$^{\uparrow}$ and DyadicRatCore$^{\uparrow}$ source certificates used by this ledger. The Cont routes in C then replay the same read path from the consumer to the threshold, tolerance, and ledger rows.

By Definition 392.3, tail-budget exactness is one of the local NameCert rows. Thus an accepted comparison can be repacked only with the displayed M, Q, T, and P data, and there is no carrier coordinate from which an undisplayed radius, host-real estimate, or external ledger certificate could be exported. □

Theorem 392.7 (CauchyCriterion RegSeqRat handoff exhaustion)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* CauchyCriterion$^{\uparrow}$ *carrier. Any* RegSeqRat$^{\uparrow}$*-to-*Real$^{\uparrow}$ *consumer read exported from K factors through the finite route*

$$W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Consequently the public read exports only the displayed source window, modulus threshold, dyadic tolerance, tail ledger, and seal row. It exports no quotient stream, selected limit, host real equality, uniqueness-of-limit claim, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier regseqrat handoff exhaustion`

Proof. Begin with the RegSeqRat$^{\uparrow}$ source window W in the carrier of Definition 392.1. A real-facing consumer must then pass through the modulus row M, because the accepted tail entries are defined only beyond the threshold selected by that schedule.

Next the read passes through Q and T. The dyadic row supplies the tolerance, and Definition 392.6 says that the ledger accepts comparisons only through this displayed threshold and tolerance.

Finally the seal coordinate E is reached by the factorization of Definition 392.2. The obligation sketch records this as the regular rational handoff and $\mathsf{Real}^{\uparrow}$ seal non-escape rows, so repacking the route exhausts every exported consumer read without adding quotient streams, selected limits, host real equality, uniqueness of limits, or ambient completeness. □

Theorem 392.8 (CauchyCriterion budgeted shared-tail seal compatibility)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier whose finite regular rational source window is also presented through the* $\mathsf{StreamName}^{\uparrow}$ *shared-tail budget route of Definition 126.400. Then every* $\mathsf{Real}^{\uparrow}$*-facing seal read exported from K factors through one common finite route consisting of the Cauchy modulus schedule M, the dyadic tolerance row Q, the tail ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row W, the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *tail-budget coverage row of Definition 757.3, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *public shared-threshold certificate of Definition 560.11, and the seal row E.*

The common route exports no quotient stream, selected limit, host real equality, countable-choice tail selector, independent tail oracle, metric completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier budgeted shared tail seal compatibility`

Proof. First apply Definition 392.7. It places each consumer route from the regular rational source to the seal inside the displayed W, M, Q, T, E packet and excludes quotient streams, selected limits, host real equality, uniqueness of limits, and ambient completeness.

Next apply Definition 392.2. The seal row E is therefore reached only after the modulus schedule, dyadic tolerance row, finite tail ledger, and regular rational handoff have all been read from the same accepted $\mathsf{CauchyCriterion}^{\uparrow}$ carrier.

Now use Definition 126.400 on the finite StreamName presentation of W. This identifies the same public readback and seal service with the finite $\mathsf{StreamName}^{\uparrow}$ window rows, ProbeBundle, InBundle, UnaryHistory, the dyadic tail-radius ledger, the shared threshold, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ finite-window ledger boundary.

Finally cite Definition 757.3 and Definition 560.11. The first fixes the admitted tail-radius read inside the finite precision, tail-window, StreamName-window, and dyadic readback rows; the second fixes the public shared-threshold row used by both the regular rational readback and the seal-facing route. Repacking these identifications gives the single finite route named in the statement, and none of the cited carriers contains a quotient stream, selected limit, host real equality, countable-choice selector, independent tail oracle, metric completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 392.9 (CauchyCriterion tail-budget coherence consumption)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier whose shared-tail seal route is presented as in Definition 392.8. If a* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet consumes this carrier at the Real-completion handoff, then the consumer must read the handoff as one finite boundary through the displayed CauchyCriterion rows M, Q, T, W, E, the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *coverage row, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row, and the displayed* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *rows*

$$M_B, O_B, S_B, W_B, D_B, R_B, A_B, L_B, H_B, C_B, P_B, N_B.$$

The consumption route exports no quotient stream, selected limit, host real equality, countable-choice selector, detached tail oracle, metric-completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First apply Definition 392.8. It binds every seal-facing read from K to the finite route through the Cauchy modulus schedule, dyadic tolerance row, tail ledger, regular rational source row, StreamName windows, dyadic tail-radius coverage row, shared threshold row, and seal row E.

Next apply the primitive scope and sibling-scope lock of Definition 917.9 and Definition 917.10. These theorems say that a $\mathsf{TailBudgetCoherence}^{\uparrow}$ consumer has no carrier rows beyond the named TailMeet, observation-budget, selector-budget, agreement-seal, limit-seal, StreamName, regular readback, dyadic-tolerance, transport, continuation, provenance, and local NameCert coordinates.

Finally use the route-scope and seal boundary results Definition 917.11 and Definition 917.6. They force the Real-completion consumer to pass through the TailBudgetCoherence rows only after the finite threshold, budget, window, tolerance, and readback surface has been displayed. Combining this with the CauchyCriterion shared-tail seal route gives one finite boundary, and none of the cited packets supplies quotient streams, selected limits, host real equality, countable-choice selectors, detached tail oracles, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail budget coherence consumption`

Theorem 392.10 (CauchyCriterion tail-budget observer route)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier whose shared-tail route is presented by Definition 392.8. Any observer-budget or finite-window observation consumer of K must factor through the displayed Cauchy modulus schedule M, dyadic tolerance row Q, tail ledger T,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row W, finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, shared-threshold certificate, and Real seal E before observing the public Real-facing endpoint.*

The route exports no quotient stream, selected limit, host real equality, detached tail oracle, or ambient completeness principle.

Proof. Apply Definition 392.8. It places every Real-facing seal read on the shared finite route through M, Q, T, W, the StreamName windows, the dyadic tail-radius coverage row, the regular shared-threshold certificate, and the seal row E.

Next use Definition 392.9. A downstream observation consumer can attach to that route only after the TailBudgetCoherence rows have made the finite threshold, budget, window, tolerance, readback, transport, continuation, package, and local naming data visible.

Finally apply RealObservationBudget consumer exactness, Definition 611.9. It reads observation budgets and finite windows through the same displayed precision and source rows, so the public endpoint is observed only after the Cauchy modulus schedule, tolerance row, tail ledger, source row, shared-threshold certificate, and seal have been exposed. The excluded quotient streams, selected limits, host real equalities, detached tail oracles, and ambient completeness principles are absent from the cited routes. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail budget observer route`

Theorem 392.11 (CauchyCriterion uniform-family quotient-refusal factorization)**.** *Let $K = (W, M, T, Q, E, H, C,$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier whose shared-tail route is presented through the budgeted seal theorem of Definition 392.8. Any finite-family or uniform-family consumer that asks for quotient-name equality, a selected limit, host real equality, hidden tail data, or ambient completeness must first consume the displayed CauchyCriterion rows M, Q, T, W, E, the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *coverage row, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row, the displayed* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *rows, and then the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *refusal row of Definition 998.13.*

Consequently the accepted carrier exports no quotient stream, selected limit, host equality, tail oracle, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle to such a family consumer.*

Proof. Apply Definition 392.8. This places every seal-facing read from K on the finite shared-tail route through M, Q, T, W, E, the StreamName window rows, the dyadic tail-radius coverage row, and the regular shared-threshold row.

Next apply Definition 392.9. A Real-completion family consumer can therefore attach to the route only after the TailBudgetCoherence rows have been displayed as threshold, budget, window, tolerance, readback, transport, continuation, package, and local naming data.

Now route the family request through Definition 998.13. A regular-tail read that reaches Real-completion must pass through the displayed StreamName, RegSeqRat, CauchyLimitSeal, RealUp seal, and refusal-ledger rows before any quotient-style request is interpreted.

Finally use Definition 998.11. Requests for quotient streams, selected limits, host real equality, hidden tail data, metric-completeness rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness are off the displayed route and are sent to the refusal ledger rather than exported as witnesses. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier uniform family quotient refusal factorization`

Definition 392.12 (CauchyCriterion classifier)**.** *For accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carriers*

$$K = (W, M, T, Q, E, H, C, P, N) \qquad \text{and} \qquad K' = (W', M', T', Q', E', H', C', P', N'),$$

the $\mathsf{CauchyCriterion}^{\uparrow}$ *classifier relates* K *and* K' *exactly when the accepted packets agree componentwise on the displayed source window, modulus schedule, tail ledger, dyadic tolerance, seal row, transport ledger, continuity-route ledger, package provenance, and local certificate provenance.*

Thus the comparison is the finite conjunction of the corresponding hsame *rows and displayed* Cont *route comparisons over* $W, M, T, Q, E, H, C, P, N$*. It has no additional coordinate for quotient streams, hidden real equality, selected limits, uniqueness of limits, or ambient completeness.*

Lean4 (definition) `BEDC.Derived.CauchyCriterionUp.CauchyCriterionClassifier`
Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionClassifier endpoint transport closure`

Theorem 392.13 (CauchyCriterion precision route normal form)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. Every accepted precision read exported from* K *has the normal form*

$$\mathsf{precision} \longmapsto M \longmapsto Q \longmapsto T \longmapsto W \longmapsto E.$$

The only rows used by this factorization are the displayed BHist *packet, the componentwise* hsame *transports in* H*, the displayed* Cont *routes in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionRoute precision endpoint closure`

Proof. Start with the carrier projection of Definition 392.1. A precision request first reads the modulus schedule M, which selects the finite threshold at which the tail ledger is inspected.

The dyadic tolerance row Q supplies the comparison budget, and the ledger row T records the accepted finite tail comparisons for that threshold and tolerance. The exactness statement of Definition 392.6 prevents replacing these rows by an undisplayed radius or an outside tail certificate.

The regular rational source row W is then the only handoff available before the seal row E. The exhaustion theorem Definition 392.7 and the seal factorization of Definition 392.2 show that every real-facing read passes through these rows.

Transport along H and replay the routes in C to keep the read inside the same finite BHist packet. The package row P and the certificate row N record the source certificates; no remaining carrier coordinate can export quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 392.14 (CauchyCriterion obligation closure package)**.** *The* $\mathsf{CauchyCriterion}^{\uparrow}$ *chapter presents an obligation-complete paper package for the explicit constructive Cauchy criterion over the displayed finite carrier. The package consists of the carrier of Definition 392.1, the classifier of Definition 392.12, tail-budget exactness from Definition 392.6, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exhaustion from Definition 392.7, the* $\mathsf{Real}^{\uparrow}$ *seal factorization from Definition 392.2, and the non-escape boundary forbidding quotient streams, selected limits, host real equality, uniqueness-of-limit claims, and ambient completeness.*

Consequently a future formal target for $\mathsf{CauchyCriterion}^{\uparrow}$ *may use this chapter as the paper-side obligation surface for carrier transport, classifier stability, dyadic tail-budget exactness, regular rational handoff exactness, and* $\mathsf{Real}^{\uparrow}$ *seal factorization, without claiming Lean verification or real completeness here.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier obligation closure package`

Proof. The carrier and classifier are displayed by Definition 392.1 and Definition 392.12. They identify the finite packet and the componentwise comparison rows on which every accepted read depends.

Tail-budget exactness is supplied by Definition 392.6. It binds each comparison to the same displayed modulus threshold, dyadic tolerance, tail ledger, and package provenance.

The regular rational handoff is exhausted by Definition 392.7. The precision-route normal form of Definition 392.13 then names the only route by which a precision request can reach a real-facing seal.

Finally Definition 392.2 supplies the $\mathsf{Real}^{\uparrow}$ seal factorization and the non-escape boundary. Repacking these clauses gives the stated paper-side obligation package while leaving Lean verification, quotient construction, selected limits, host real equality, uniqueness of limits, and ambient completeness outside the claim. □

Theorem 392.15 (CauchyCriterion scoped tail-seal transport)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. Any transported precision read that preserves the displayed* BHist *packet, the componentwise* hsame *rows in* H*, the displayed* Cont *routes in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ *row* N *factors through the kernel-visible route*

$$M \longmapsto Q \longmapsto T \longmapsto W \longmapsto E.$$

This factorization is stable under the classifier of Definition 392.12: classifier-related carriers transport the same modulus row, dyadic tolerance row, finite tail ledger, regular rational handoff, and $\mathsf{Real}^{\uparrow}$ *seal row as a single scoped packet.*

The scoped route introduces no quotient stream, selected limit, host real equality, uniqueness-of-limit claim, or ambient completeness row.

Proof. First project the precision route normal form of Definition 392.13. It identifies the only displayed route from a precision request to the real-facing seal, namely the passage through M, Q, T, W, and E inside the same carrier.

The dyadic tail ledger is exact by Definition 392.6. Hence the transported read cannot replace M, Q, or T by an undisplayed threshold, an outside radius, or an external tail certificate.

Next use Definition 392.7. The finite ledger reaches the regular rational source window W before any $\mathsf{Real}^{\uparrow}$ seal row is exposed, so the handoff is part of the same scoped packet rather than a separate stream quotient.

Finally apply the seal factorization of Definition 392.2. The classifier of Definition 392.12 preserves exactly the displayed carrier rows and their hsame, Cont, Pkg, and NameCert witnesses, so the same $M \longmapsto Q \longmapsto T \longmapsto W \longmapsto E$ route is transported to any classifier-related carrier. Since none of the excluded analytic data is a coordinate of the carrier, none is introduced by the scoped transport. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier scoped tail seal transport`

Theorem 392.16 (CauchyCriterion classifier stability scope)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ and $K' = (W', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carriers. If the classifier of Definition 392.12 relates K and K', then the comparison is scoped by one finite dependency packet: the carrier rows W, M, T, Q, E, the transport rows in H and H', the displayed* Cont *routes in C and C', the* Pkg *rows P, P', and the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ *rows N, N'. Under this packet, classifier stability preserves the same* BHist *source,* hsame *transport data,* Cont *read routes,* Pkg *provenance,* NameCert *provenance, dyadic tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal rows, and it introduces no quotient stream, selected limit, host real equality, uniqueness-of-limit claim, or ambient completeness coordinate.*

Proof. Project the classifier definition. Its only comparison clauses are the componentwise rows for $W, M, T, Q, E, H, C, P, N$ and the corresponding primed rows, so the carrier and classifier are already bound to the same displayed BHist packet.

The modulus threshold and tolerance coordinates are stable by Definition 392.5. These clauses identify the route by which a consumer reads M and Q after transport.

The tail ledger and regular rational handoff remain inside the same packet by Definition 392.6 and Definition 392.7. The seal row is then reached by Definition 392.2.

Thus every classifier-related read is repacked through H, H', C, C', P, P', and N, N' over the displayed carrier rows. Since the classifier has no field for the excluded analytic data, stability cannot export such a field. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier classified tail budget transport`

Theorem 392.17 (CauchyCriterion tail-budget scope)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. Every scoped tail-budget read in K is bound to the same finite packet as the carrier and classifier: the displayed* BHist *row for K, the* hsame *transports in H, the* Cont *routes in C, the* Pkg *provenance row P, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ *row N, the dyadic threshold and tolerance rows M, Q, the tail ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row W, and the* $\mathsf{Real}^{\uparrow}$ *seal row E. Consequently tail-budget exactness is not a separate analytic assumption; it is the packet-local claim that each accepted comparison uses the threshold selected by M and the dyadic budget recorded by Q before reaching W and E.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionTasteGate single carrier alignment`

Proof. Begin with the ledger projection T. The exactness theorem Definition 392.6 says that each accepted comparison in T is read through the displayed threshold M, tolerance Q, ledger row T, and package provenance P.

The hsame and Cont rows keep that ledger read in the same carrier. The transport row H records componentwise sameness for the carrier rows, while C records the consumer route by which the threshold, tolerance, and ledger are reached.

The handoff to W is supplied by Definition 392.7, and the seal row E is supplied by Definition 392.2. These two results bind the tail budget to the regular rational source and the real-facing seal without adding a new carrier coordinate.

Repacking the displayed clauses with the local NameCert row N gives the stated scoped tail-budget packet. No outside modulus, rational radius, host metric estimate, selected limit, or ambient completeness datum can enter, because none is a row of K. □

Theorem 392.18 (CauchyCriterion shared tail-budget seal comparison)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ and $K' = (W', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carriers related by the classifier of Definition 392.12. Suppose their modulus rows and dyadic tolerance rows select the same displayed tail-budget packet. Then the two* $\mathsf{Real}^{\uparrow}$ *seal reads factor through the same finite route*

$$M \longmapsto Q \longmapsto T \longmapsto W \longmapsto E$$

and

$$M' \longmapsto Q' \longmapsto T' \longmapsto W' \longmapsto E',$$

with the two routes compared only by the shared tail-budget rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The comparison exports no selected limit, uniqueness-of-limit claim, host metric estimate, quotient stream equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier shared tail budget seal comparison`

Proof. First apply the classifier stability scope of Definition 392.16. The comparison of K and K' is already confined to the displayed carrier rows, the componentwise hsame rows, the Cont routes, provenance, and local certificates.

Use the tail-budget scope of Definition 392.17. The shared modulus and dyadic tolerance hypothesis identifies the tail-budget packet read by M, Q, T with the packet read by M', Q', T'.

Next apply the scoped tail-seal transport theorem Definition 392.15. It forces each transported precision read to pass through the finite route from modulus to dyadic tolerance, tail ledger, regular rational handoff, and seal row.

Finally use the precision route normal form of Definition 392.13. Repacking the two displayed routes through the shared tail budget gives the stated comparison. None of the cited carrier rows contains selected limits, uniqueness of limits, host metric estimates, quotient equality, host real equality, or ambient completeness, so those objects cannot be exported. □

Theorem 392.19 (CauchyCriterion RealUp seal scope)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. The* $\mathsf{Real}^{\uparrow}$ *seal row E is scoped by the same finite packet that supports the carrier, classifier, dyadic tail ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. In particular, every exported seal read factors through M, Q, T, and W while retaining the displayed* BHist *carrier,* hsame *transport rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ *row. The seal scope therefore records a packet-local endpoint for the constructive Cauchy criterion and does not assert real completeness, quotient equality of streams, a selected limit, host real equality, or uniqueness of limits.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real seal scope`

Proof. Use the precision route normal form of Definition 392.13. It gives the displayed path from a precision request through M, Q, T, W, and E inside the carrier.

The scoped tail-seal transport theorem Definition 392.15 keeps this path stable under the carrier transports, Cont read routes, package provenance, and local certificate row. Thus the seal is not read from a separate real object or an ambient limit construction.

The classifier stability scope of Definition 392.16 then shows that classifier-related carriers preserve the same seal-facing dependency packet. The tail-budget scope of Definition 392.17 supplies the exact threshold, tolerance, and ledger rows used before the handoff reaches W.

Finally apply Definition 392.2. The seal row is the endpoint of the displayed constructive criterion route, and the carrier has no coordinate for completeness, quotient equality, selected limits, host real equality, or uniqueness of limits. □

Theorem 392.20 (CauchyCriterion public certificate)**.** *The carrier, classifier, tail-bound* $\mathsf{Real}^{\uparrow}$ *seal factorization,* NameCert *obligation sketch, modulus threshold stability, dyadic tail ledger exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exhaustion, precision route normal form, obligation closure package, scoped tail-seal transport, classifier stability scope, tail-budget scope, and* $\mathsf{Real}^{\uparrow}$ *seal scope assemble one exported* $\mathsf{CauchyCriterion}^{\uparrow}$ *public certificate over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicTailBound}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

This certificate is exactly the finite constructive Cauchy criterion packet of Definition 392.1 and Definition 392.12. It preserves the exclusions on quotient streams, selected limits, host real equality, uniqueness of limits, ambient completeness, and bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier public certificate`

Proof. First take the carrier and classifier from Definition 392.1 and Definition 392.12. These two displays provide the finite BHist packet, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$ row.

Next attach the obligation rows. The tail-bound seal factorization is Definition 392.2, the local naming obligations are Definition 392.3, and the modulus threshold stability is Definition 392.5.

The dyadic and rational handoff data come from Definition 392.6, Definition 392.7, and Definition 392.13. Together they force every precision route through the displayed modulus, dyadic tolerance, finite tail ledger, regular rational source window, and seal row.

Finally repack the scoped surface using Definition 392.14, Definition 392.15, Definition 392.16, Definition 392.17, and Definition 392.19. Since each cited theorem stays inside the same visible packet and records the same non-escape boundary, the assembled public certificate exports no quotient stream, selected limit, host real equality, uniqueness-of-limit claim, ambient completeness principle, or checked bridge. □

Theorem 392.21 (CauchyCriterion Real-completion consumer bridge)**.** *Every exported* $\mathsf{CauchyCriterion}^{\uparrow}$ *public packet consumed by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing interface factors through the displayed modulus, dyadic tolerance, finite tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *window,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row of the same finite carrier. The bridge exports no quotient stream, selected limit, host real equality, uniqueness-of-limit claim, ambient completeness principle, or bridge-checked verification.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real completion consumer bridge`

Proof. Start from the public packet of Definition 392.20. It already assembles the carrier, classifier, tail-bound seal factorization, local naming obligations, modulus threshold stability, dyadic tail ledger exactness, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, precision route normal form, scoped transport, classifier stability, tail-budget scope, and $\mathsf{Real}^{\uparrow}$ seal scope.

The consumer-facing route is the same finite route. The modulus threshold is fixed by Definition 392.5, the dyadic tolerance and tail ledger are fixed by Definition 392.6, and the regular rational handoff is fixed by Definition 392.7.

The $\mathsf{Real}^{\uparrow}$ endpoint is then read through Definition 392.2, Definition 392.13, Definition 392.15, and Definition 392.19. These theorems keep the endpoint inside the carrier rows rather than turning it into a host completeness assertion.

Repack those rows as the downstream bridge. Since every cited clause remains inside the displayed packet and the public certificate explicitly preserves the same exclusions, a Real-facing or completion-facing consumer receives the constructive criterion route but no quotient stream, selected limit, host real equality, uniqueness-of-limit theorem, ambient completeness principle, or checked bridge status. □

Theorem 392.22 (CauchyCriterion modulus-tail consumer exactness)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. The modulus row M, dyadic tail ledger T,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff W, and* $\mathsf{Real}^{\uparrow}$ *seal row E determine the same consumer-visible tail bound for every* $\mathsf{Real}^{\uparrow}$ *or completion-facing readback route through H, C, P, N.*

Equivalently, any such consumer route reads the tail bound by passing through the displayed modulus threshold, the same dyadic tolerance row, the same finite tail ledger, the same regular rational handoff, and the same real seal. It cannot obtain a different bound from quotient streams, selected limits, host real equality, uniqueness of limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier modulus tail consumer exactness`

Proof. Start with Definition 392.5. It fixes the threshold row M as the only modulus-facing entry point for a precision request in the carrier.

Apply Definition 392.6. The dyadic tolerance row and finite tail ledger T read the bound attached to that threshold, so a consumer cannot switch to a different tail comparison after the modulus row has been fixed.

Use Definition 392.7. The handoff row W exhausts the regular rational source window that can feed the tail ledger, and the route still passes through the displayed transport, continuation, package, and local naming rows.

Finally apply Definition 392.21. The $\mathsf{Real}^{\uparrow}$ and completion-facing consumer routes read the same seal row E after the modulus, ledger, and handoff rows have been fixed, so the consumer-visible tail bound is exact for every displayed readback route. □

Theorem 392.23 (CauchyCriterion tail-budget seal determinacy)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ and $K' = (W', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packets. If they display the same modulus row, dyadic tolerance row, finite tail ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, then they expose the same* $\mathsf{Real}^{\uparrow}$ *seal route through H, C, P, N. The seal route cannot be changed by quotient stream equality, a selected limit, host real equality, uniqueness of limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail budget seal determinacy`

Proof. Start with the shared tail-budget seal comparison of Definition 392.18. The seal row is read from the displayed modulus threshold, dyadic tail ledger, regular rational handoff, and real endpoint, not from an ambient limit construction.

Use Definition 392.19. The $\mathsf{Real}^{\uparrow}$ seal remains inside the carrier rows together with the displayed transports, continuations, package provenance, and local naming row.

Finally apply Definition 392.22. Once the modulus row, dyadic tolerance row, tail ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff are the same, every completion-facing readback route sees the same consumer-visible tail bound and hence the same seal route through H, C, P, N. □

Theorem 392.24 (CauchyCriterion RegSeqRat Real route lock)**.** *Every accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *public* $\mathsf{Real}^{\uparrow}$*-seal consumer factors through the Cauchy tail-bound row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the dyadic tolerance ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the displayed packet. No such consumer may pass through host real equality, a selected external limit, a quotient stream, or an ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier regseqrat real route lock`

Proof. Start with Definition 392.2. It places the tail-bound read inside the displayed modulus, dyadic tolerance, tail ledger, regular rational source, and real seal rows of the accepted packet.

Next use Definition 392.7. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff exhausts the regular rational source rows available to the CauchyCriterion consumer, so the route cannot replace them by quotient stream data or a selected external limit.

The finite-window input is supplied by Definition 126.79, and the real endpoint is kept inside the packet by Definition 392.19. Thus the consumer route reaches $\mathsf{Real}^{\uparrow}$ through the same finite window, source row, dyadic ledger, and seal row.

Finally apply Definition 392.23. Any route with the same modulus row, dyadic tolerance row, finite tail ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff exposes the same $\mathsf{Real}^{\uparrow}$ seal route through H, C, P, N, so alternate routes using host equality, external limits, quotient streams, or ambient completeness coordinates are ruled out. □

Theorem 392.25 (CauchyCriterion completion-facing readback normal form)**.** *Every completion-facing consumer route for an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet normalizes to the displayed route*

$$M \to Q \to T \to W \to E$$

together with the transport, continuation, provenance, and local naming rows H, C, P, N*. No normal form may use a selected limit, quotient stream, host real equality, uniqueness-of-limit theorem, ambient completeness principle, or unrecorded completion object.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier completion facing readback normal form`

Proof. First use the public certificate Definition 392.20. It assembles the carrier, classifier, modulus row, dyadic tolerance row, finite tail ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, transports, continuations, provenance, and local certificate into one finite packet.

Next apply Definition 392.21. Every completion-facing consumer factors through the same displayed modulus, dyadic tolerance, finite tail ledger, regular rational window, and real seal route.

The exactness theorem Definition 392.22 fixes the order of the readback: the modulus row feeds the dyadic tolerance and tail ledger, the ledger feeds the regular rational handoff, and the handoff reaches the $\mathsf{Real}^{\uparrow}$ seal through H, C, P, N.

Thus the consumer route has the stated normal form and has no coordinate from which to read selected limits, quotient stream equality, host real equality, uniqueness of limits, ambient completeness, or an unrecorded completion object. □

Theorem 392.26 (CauchyCriterion RegSeqRat StreamName RealUp handoff)**.** *Let $K = (W, M, T, Q, E, H, C, P, N$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier. Suppose the W row is read from a finite public* $\mathsf{StreamName}^{\uparrow}$ *observation window and is accepted by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. Then every* $\mathsf{Real}^{\uparrow}$ *seal read of K factors through exactly the route*

$\mathsf{StreamName}^{\uparrow}$ *finite window* $\to$ $\mathsf{RegSeqRat}^{\uparrow}$ *source row* $\to$ $\mathsf{CauchyModulus}^{\uparrow}, \mathsf{DyadicTailBound}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}$ *budg*

No quotient stream, selected host limit, host real equality, or ambient completeness row can enter this route.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier regseqrat streamname realup handoff`

Proof. Start with the carrier definition. Its displayed rows expose the regular rational handoff W, modulus row M, tail ledger T, dyadic tolerance row Q, real seal row E, and the transport, continuation, provenance, and local naming rows H, C, P, N.

Apply Definition 392.2. The tail-bound budget is read through the modulus, dyadic tolerance, tail ledger, and real seal rows already present in K.

Use Definition 392.7. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff exhausts the finite source rows available from the displayed observation window, so the source cannot be replaced by a quotient stream or by a selected host limit.

Finally apply Definition 392.25. The completion-facing route normalizes to the same displayed rows. Together with the sibling inputs named in the chapter introduction for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, this binds every seal read to the stated finite handoff route and excludes host real equality and ambient completeness rows. □

Theorem 392.27 (CauchyCriterion diagonal-budget readback compatibility)**.** *Let $K = (W, M, T, Q, E, H, C, P, N$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier, and let $B = (\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *packet through which the W, Q, E rows of K are read by a completion-facing consumer. Then every such readback first normalizes to the CauchyCriterion route*

$$M \to Q \to T \to W \to E$$

with H, C, P, N, and then to the diagonal-budget route

$$\epsilon \to M_B \to W_B \to D_B \to K_B \to S_B$$

with H_B, C_B, P_B, N_B. No quotient stream, selected limit, host real equality, uniqueness-of-limit theorem, trichotomy row, or ambient completeness coordinate may enter either side of the readback.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier diagonal budget readback compatibility`

Proof. First apply Definition 392.25. It normalizes every completion-facing read of K to $M \to Q \to T \to W \to E$ with the displayed transport, continuation, provenance, and naming rows.

Next use Definition 392.26. The W row is read through the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, while Q supplies the dyadic budget row and E supplies the $\mathsf{Real}^{\uparrow}$ seal.

Now pass to the diagonal-budget selector. The route factorization theorem Definition 853.14 identifies every completion-facing read through B with the finite route from ϵ through M_B, W_B, D_B, K_B to S_B.

Finally apply Definition 853.19 and Definition 853.31. These keep the RegSeqRat-to-Real handoff and the shared-tail modulus seal on the displayed diagonal-budget route.

The non-escape scope theorem Definition 853.24 then rules out quotient streams, selected limits, host real equality, uniqueness-of-limit inputs, trichotomy rows, and ambient completeness coordinates. □

Theorem 392.28 (CauchyCriterion RealTailAgreementSeal route correspondence)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Every accepted completion-facing consumer of* K *whose* $\mathsf{Real}^{\uparrow}$ *seal is mediated by an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet factors through the same finite route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow},\ \mathsf{RegularCauchyTailBudget}^{\uparrow},\ \mathsf{RegularCauchyTailMeet}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{R}$$

Conversely, every terminal $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route supplies the CauchyCriterion tail-budget,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows needed by Definition 392.26. The correspondence exports no selected limit, quotient-stream equality, uncapped observation, host real equality, ambient completeness principle, or second agreement coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real tail agreement seal route correspondence`

Proof. Start from the CauchyCriterion handoff theorem Definition 392.26. It fixes the $\mathsf{StreamName}^{\uparrow}$ source window, the $\mathsf{RegSeqRat}^{\uparrow}$ source row, the dyadic budget rows, and the $\mathsf{Real}^{\uparrow}$ seal as the only displayed handoff route for the accepted packet.

The tail-budget seal determinacy theorem Definition 392.23 and the completion-facing readback normal form Definition 392.25 then force every completion-facing consumer to read the same budget, regular rational, and real-seal rows before any $\mathsf{Completion}^{\uparrow}$ surface is exposed.

For the mediated case, apply Definition 699.20 and Definition 699.21. These identify the terminal $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer route as the finite path through the observation cap, regular Cauchy tail budget, shared tail meet, finite stream window, dyadic readback, agreement row, and the displayed $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal rows.

Projecting this terminal route back to the CauchyCriterion rows gives the forward factorization. Reading it in the other direction supplies exactly the tail-budget, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows required by the handoff theorem, while the non-escape clauses on both packets exclude every host, quotient, uncapped, completeness, or second-agreement route. □

Theorem 392.29 (CauchyCriterion and finite-family UniformCauchyCriterion route agreement)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Let* U *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose finite-family ledger is either singleton or a fixed finite family with the same displayed shared window row* W*, shared modulus row* M*, dyadic tolerance row* Q*, tail ledger* T*, and real seal row* E*. Assume that both packets consume the same* $\mathsf{StreamName}^{\uparrow}$ *source route, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tail route, and the same* $\mathsf{Real}^{\uparrow}$ *seal route as displayed by chapter 126, chapter 301, and chapter 38. Then the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *family route restricts to the same finite* $\mathsf{CauchyCriterion}^{\uparrow}$ *route*

$$M \to Q \to T \to W \to E.$$

Equivalently, the singleton or fixed finite family packet exposes no additional route beyond the CauchyCriterion modulus, tolerance, tail, window, and seal rows. The correspondence uses Definition 392.28, Definition 435.9, and Definition 435.11, and exports no quotient stream, selected limit, host real equality, ambient completeness principle, or independently chosen family limit.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier uniformcriterion finite family reduction`

Proof. Project the CauchyCriterion packet first. By Definition 392.28, the accepted K packet exposes the finite seal route through its displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, and this route is exactly the CauchyCriterion handoff through M, Q, T, W, E.

Now project the uniform packet along its singleton or fixed finite family ledger. The hypotheses identify the shared window row with W, the shared modulus row with M, the dyadic tolerance row with Q, the tail ledger with T, and the seal row with E componentwise.

Apply Definition 435.9. It says that every accepted family tail comparison in U is read through the displayed uniform tail ledger after the shared window, shared modulus, and dyadic tolerance rows have already been fixed.

Apply Definition 435.11. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer therefore sees the same finite tail evidence as the ledger route, so the family-side readback cannot insert a second stream source, a second modulus schedule, or an unrecorded tolerance row.

The cited $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ construction chapters fix the source, tail, and seal carriers used by both packets. After the componentwise identifications above, the uniform route and the single CauchyCriterion route expose the same finite path $M \to Q \to T \to W \to E$. The non-export clauses on the two packets then exclude quotient streams, selected limits, host real equality, ambient completeness, and any additional family-limit object. □

Theorem 392.30 (CauchyCriterion CompleteMetric consumer boundary)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet* $K = (W, M, T, Q, E, H, C, P, N)$ *must pass through the displayed tail-budget route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal before any completion-facing use. The consumer cannot import quotient streams, selected limits, host real equality, ambient completeness, or uniqueness of limits.*

Proof. Apply Definition 392.29 and Definition 392.28. They force the finite tail route through M, Q, T, W, E and place the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before completion-facing readback.

The terminal completion routes below then expose only the same sealed endpoint to $\mathsf{CompleteMetric}^{\uparrow}$ consumers. Since the packet has no quotient stream, selected limit, host equality, ambient completeness, or uniqueness-of-limit coordinate, the displayed tail-budget and Real-seal route is the whole consumer boundary. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier complete metric consumer boundary`

Theorem 392.31 (CauchyCriterion four-face pullback correspondence)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Suppose its* $\mathsf{StreamName}^{\uparrow}$ *finite window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and budget row,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail and readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row agree componentwise with a terminal four-face finite section. Then the* $\mathsf{CauchyCriterion}^{\uparrow}$ *consumer route*

$$M \to Q \to T \to W \to E$$

corresponds exactly to that four-face terminal pullback. Conversely, any terminal four-face pullback with those four rows supplies the same displayed $\mathsf{CauchyCriterion}^{\uparrow}$ *route. The correspondence exports no quotient stream equality, selected limit, host sequence equality, ambient Real completeness, metric-completeness datum, countable-choice row, or bridge row.*

Proof. For the forward direction, apply Definition 392.28 and Definition 392.30. These identify every CauchyCriterion consumer with the displayed finite route through M, Q, T, W, E before any completion-facing readback is allowed.

Next apply Definition 126.592 and Definition 126.586. Since the StreamName window, dyadic tolerance and budget, RegSeqRat tail readback, and Real seal rows are the same displayed rows, terminality reads the CauchyCriterion route as the unique four-face pullback route.

For the reverse direction, start with such a terminal four-face pullback and project its four faces. The projected rows are precisely W, Q, T, E, while the accepted packet supplies the modulus row M and the existing H, C, P, N certificates. Repacking these displayed rows gives the same route $M \to Q \to T \to W \to E$.

The final exclusions are coordinate exclusions. None of the cited terminality or boundary results adds a quotient equality, a selected limit, host sequence equality, Real completeness, metric completeness, countable choice, or a bridge certificate, so the correspondence is only the finite BEDC pullback described above. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier streamname four face pullback correspondence`

Theorem 392.32 (CauchyCriterion completion uniform-family factorization)**.** *Let $K = (W, M, T, Q, E, H, C, P, N$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let U be an accepted singleton or fixed finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet with the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. If the two packets identify the shared window, shared modulus, dyadic tolerance, tail ledger, and real-seal rows as in Definition 392.29, then they expose the same completion-facing route before any* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer can read the seal.*

The factorized route is the finite path through M, Q, T, W, E together with the shared stream-name source, regular-rational readback, dyadic ledger, BHist, hsame, Cont, Pkg, *and* NameCert *rows. It exports no quotient stream, selected limit, host real equality, ambient completeness principle, unrecorded completion object, or independently chosen family limit.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier completion uniform family factorization`

Proof. Start from Definition 392.29. It identifies the singleton or fixed finite-family uniform route with the same CauchyCriterion path $M \to Q \to T \to W \to E$ once the stream-name, regular-rational, dyadic, and real-seal rows agree.

Next apply Definition 392.25. The completion-facing consumer can read only the displayed CauchyCriterion rows, so the seal is reached through the same window, modulus, tolerance, tail, and real row.

Finally use Definition 435.21. It transports the downstream $\mathsf{Real}^{\uparrow}$ consumer on the uniform packet through the same finite family ledger. Combining these consumer clauses gives one route for $\mathsf{Completion}^{\uparrow}$ and $\mathsf{RealCauchyCompletion}^{\uparrow}$, with the quotient, selected-limit, host-equality, completeness, unrecorded-completion, and second-family-limit exits absent. □

Theorem 392.33 (CauchyCriterion tail-budget Real completion boundary)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer sharing the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows must factor through the finite route $M, Q, T, W, E, H, C, P, N$ before it can read a completion seal. Conversely, a consumer that displays this route has exactly the CauchyCriterion completion-boundary data available, with no quotient stream, selected limit, host real equality, ambient completeness row, or unrecorded completion object.*

Proof. Apply Definition 392.2 and Definition 392.25. They place every Real-facing seal and completion-facing read behind the displayed modulus, tolerance, tail, window, and seal rows of K.

Next use Definition 392.32. It identifies the same finite route for singleton and fixed finite-family consumers sharing the displayed source rows. The RealCauchyCompletion$^{\uparrow}$ chapter is then read as the downstream consumer surface: it receives only the route already exposed by K, and a consumer that already has $M, Q, T, W, E, H, C, P, N$ has all completion-boundary rows that the CauchyCriterion packet exports. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tail budget real completion boundary`

Theorem 392.34 (CauchyCriterion terminal RegSeqRat Real completion route)**.** *Let $K = (W, M, T, Q, E, H, C, P$ be an accepted* CauchyCriterion$^{\uparrow}$ *packet. Any terminal Real-and-Completion consumer sharing the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyModulus$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows factors through the finite route*

$$M, Q, T, W, E, H, C, P, N.$$

The route reads the stream-name source, regular-rational sequence row, dyadic tolerance core, Cauchy modulus schedule, uniform Cauchy criterion packet, and Real seal through the same visible BHist, hsame, Cont, Pkg, *and* NameCert *support rows. It exports no quotient stream, selected limit, host real equality, uniqueness-of-limit theorem, ambient completeness principle, or unrecorded completion object.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier terminal regseqrat real completion route`

Proof. First apply Definition 392.32. It identifies the singleton and fixed finite-family UniformCauchyCriterion$^{\uparrow}$ completion-facing reads with the same finite CauchyCriterion route before any completion consumer can read the Real seal.

Next use Definition 392.33. It confines every Completion$^{\uparrow}$ or RealCauchyCompletion$^{\uparrow}$ consumer sharing the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows to $M, Q, T, W, E, H, C, P, N$.

Now attach the RegSeqRat and StreamName handoff by Definition 392.26. This places the stream source, regular-rational readback, dyadic budget, and Real-facing read on the same packet.

Finally use Definition 392.28. The Real tail-agreement seal is already routed through the same modulus, tolerance, tail, window, and support rows, so a terminal consumer that asks for the displayed six source rows receives exactly the finite route above. The carrier has no coordinate for quotient streams, selected limits, host real equality, uniqueness-of-limit data, ambient completeness, or any off-route completion object. □

Theorem 392.35 (CauchyCriterion mature downstream examples)**.** *The public certificate, real-completion consumer bridge, modulus-tail consumer exactness, tail-budget seal determinacy, and completion-facing readback normal form for an accepted* CauchyCriterion$^{\uparrow}$ *packet give representative downstream consumers for the same displayed rows*

$$W, M, T, Q, E, H, C, P, N.$$

In particular, Real$^{\uparrow}$ *seal reads, completion-facing readback routes, and* RegSeqRat$^{\uparrow}$ *handoff consumers receive only those rows. They receive no quotient stream, selected limit, host real equality, uniqueness-of-limit theorem, ambient completeness principle, or unrecorded completion object.*

Proof. First assemble the public packet by Definition 392.20. The carrier rows $W, M, T, Q, E, H, C, P, N$ are the sole exposed certificate surface.

Next attach the downstream routes. Apply Definition 392.21 for completion-facing consumers and Definition 392.22 for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff path through the modulus, tail ledger, and dyadic tolerance rows.

Finally use Definition 392.23 and Definition 392.25. These results fix the $\mathsf{Real}^{\uparrow}$ seal read and the completion-facing normal form on the same finite row set, so none of the excluded quotient, host equality, limit-selection, uniqueness, completeness, or unrecorded completion data can be consumed downstream. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier mature downstream examples`

Lemma 392.36 (CauchyCriterion completion uniform-family downstream coverage)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let U be an accepted singleton or fixed finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet with the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Assume the shared window, shared modulus, dyadic tolerance, tail ledger, and real-seal rows are identified as in Definition 392.32.*

Every accepted local downstream consumer of the resulting completion uniform-family fact is exhausted by the finite alternatives

$$M, Q, T, W, E, H, C, P, N,$$

together with the displayed stream-name source, regular-rational readback, dyadic ledger, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. Equivalently, any consumer-facing use factors through one of the displayed CauchyCriterion route rows, the singleton or fixed finite-family uniform ledger rows, or the terminal* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *completion route rows of Definition 392.34. A request for a quotient stream, selected limit, host real equality, uniqueness-of-limit theorem, ambient completeness principle, unrecorded completion object, independently chosen family limit, or unlisted coordinate is outside the BEDC packet.*

Proof. Start with Definition 392.32. It identifies the accepted CauchyCriterion packet and the accepted singleton or fixed finite-family uniform packet before any $\mathsf{Completion}^{\uparrow}$ or $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer can read the seal. The common route is the finite path through M, Q, T, W, E, together with the shared stream-name source, regular-rational readback, dyadic ledger, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Now apply Definition 392.34. Any terminal Real-and-Completion consumer sharing the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows must factor through

$$M, Q, T, W, E, H, C, P, N.$$

That theorem also records that the route reads the stream-name source, regular-rational sequence row, dyadic tolerance core, Cauchy modulus schedule, uniform Cauchy criterion packet, and Real seal through the same visible support rows.

These two cited routes cover the local consumer surface. A completion-facing consumer is first confined by the factorization theorem, and a terminal consumer that also asks for the regular-rational and real-completion handoff is confined by the terminal route theorem. Since both routes name the same finite rows and both explicitly withhold quotient streams, selected limits, host real equality, uniqueness data, ambient completeness, unrecorded completion objects, and independent family limits, any request for an unlisted coordinate is not an additional case. It is simply not a coordinate exported by the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier completion uniform family downstream coverage`

Theorem 392.37 (CauchyCriterion StreamName Real budget-selector boundary)**.** *Let $K = (W, M, T, Q, E, H, C,$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$*-completion consumer or finite-window budget selector sharing the displayed*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

rows must read through the CauchyCriterion route

$$M, \quad Q, \quad T, \quad W, \quad E, \quad H, \quad C, \quad P, \quad N$$

before exposing a selector output. Conversely, a selector that displays this route has exactly the finite CauchyCriterion budget surface available for the shared StreamName–RegSeqRat–DyadicRatCore–Real read.

The boundary exports no quotient stream, selected limit, host real equality, ambient completeness principle, unrecorded completion object, or selector choice independent of $M, Q, T, W, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier streamname real budget selector boundary`

Proof. Start with the CauchyCriterion handoff Definition 392.26. It places the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ source row, the $\mathsf{DyadicRatCore}^{\uparrow}$ budget rows, and the $\mathsf{Real}^{\uparrow}$ seal on the displayed CauchyCriterion carrier rows.

Next apply the completion boundary Definition 392.33. It says that any $\mathsf{Completion}^{\uparrow}$ or $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer sharing those four faces must factor through $M, Q, T, W, E, H, C, P, N$ before it can read a completion seal.

Now use the finite-window selector surfaces from the StreamName and Real siblings. The request and coverage theorems Definition 126.542 and Definition 126.543 identify a finite-window selector by its selected observation, bundle membership, tail-selector route, finite budget surface, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the support rows $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. The budget handoff Definition 126.390 and the boundary Definition 126.316 then force the $\mathsf{Real}^{\uparrow}$ read to use the same finite surface.

Finally read the completion-facing exit through Definition 126.500. Its four faces are exactly $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no fifth completion coordinate. Combining this exit route with the CauchyCriterion handoff and tail-budget boundary gives the stated factorization before any selector output is exposed, and the excluded objects have no row in the resulting packet. □

Theorem 392.38 (CauchyCriterion finite-request section return)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet whose displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows agree with a public finite request section. Then the selector-seal route of K returns exactly the* $\mathsf{StreamName}^{\uparrow}$ *finite-request section before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Conversely, the displayed finite request section determines the $\mathsf{CauchyCriterion}^{\uparrow}$ *selector budget surface over the same $M, Q, T, W, E, H, C, P, N$ rows. The return route exports no quotient stream, selected limit, host real equality, countable-choice selector, ambient completeness principle, or second seal source.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite request section return`

Proof. First apply Definition 392.37. It places the selector-facing read on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before any selector output or $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Next use Definition 392.39 and Definition 392.40. These results put the selector route and its seal-facing pullback on the same finite structural surface carried by hsame, Cont, Pkg, and NameCert.

For the return direction, read the StreamName and Real finite-request locks. The StreamName terminality rows identify the finite request section by the selected window, bundle membership, tail-selector route, pointwise $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and support rows; the $\mathsf{Real}^{\uparrow}$ finite-request fusion and seal lock then force the seal-facing consumer to reuse that same section.

Finally apply Definition 392.41. The synchronizer is total on the common finite packet, so the CauchyCriterion selector budget surface is determined by the displayed section and cannot escape to quotient streams, selected limits, host real equality, countable choice, ambient completeness, or a second seal source. □

Theorem 392.39 (CauchyCriterion uniform selector-seal compatibility)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose the two presentations display the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite rows at the completion-facing boundary. Then the budget-selector output of the* $\mathsf{CauchyCriterion}^{\uparrow}$ *route and the shared-threshold seal output of the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *route factor through the same finite packet before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

The common packet is the displayed StreamName–RegSeqRat–DyadicRatCore–Real surface with its BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It contains no quotient stream, selected limit, host real equality, detached family threshold, ambient completeness principle, or second completion seal.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier uniform selector seal compatibility`

Proof. Begin with the CauchyCriterion side. Apply Definition 392.26 and Definition 392.33. These place the selector-facing read on the displayed $M, Q, T, W, E, H, C, P, N$ packet before any Real or Completion seal can be consumed.

Next expose the StreamName exit route. The finite-window request and coverage rows, together with Definition 126.390, Definition 126.316, and Definition 126.500, identify the four public faces as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with no fifth completion coordinate.

Now read the uniform presentation. Apply Definition 435.10. The family threshold request is transported as one finite route through the displayed window, threshold, tolerance, and tail rows, with the support rows supplied by H_U, C_U, P_U, N_U.

Finally apply Definition 435.4, Definition 435.57, and Definition 435.8. The uniform seal is downstream of the same finite packet, and the non-escape boundary removes quotient streams, selected limits, host equality, detached thresholds, ambient completeness, and any second completion seal. Hence the selector route and the shared-threshold seal route agree on the finite packet before the $\mathsf{Real}^{\uparrow}$ seal is exposed. □

Theorem 392.40 (CauchyCriterion selector-seal route pullback)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let $D = (B_D, T_D, W_D, R_D, L_D, H_D, C_D, P_D, N_D)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root formal source. Suppose the two packets share the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows at the selector-seal boundary.*

Then the $\mathsf{CauchyCriterion}^{\uparrow}$ *selector-seal route is the pullback of the diagonal root formal route along the structural rows*

$$\mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The resulting route is exactly the common StreamName–RegSeqRat–DyadicRatCore–Real surface with its structural support rows. It exports no quotient stream, selected limit, host real equality, countable-choice selector, ambient completeness principle, or second seal source.

Proof. Begin with the CauchyCriterion boundary Definition 392.37. It forces every selector-facing read through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before any selector output is exposed.

Next use the uniform selector-seal compatibility of Definition 392.39. This places the selector route on the same finite structural surface carried by hsame, Cont, Pkg, and NameCert, with no extra completion-facing coordinate.

Now read the diagonal root formal source through Definition 646.182 and Definition 646.183. The root route displays the finite source rows and consumes its support rows in the same structural order.

Finally apply Definition 646.186. The diagonal seal-facing route has no source outside its displayed finite tuple, so restricting it along the shared StreamName–RegSeqRat–DyadicRatCore–Real rows gives precisely the CauchyCriterion selector-seal route. The excluded objects have no coordinate in either accepted packet. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier selector seal route pullback`

Theorem 392.41 (CauchyCriterion selector-seal synchronizer totality)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let* $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose the two packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and finite-window selector rows at the completion-facing boundary. Then there is a total synchronizer route between the CauchyCriterion budget-selector surface and the UniformCauchyCriterion shared-threshold surface before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The synchronizer route is the common StreamName–RegSeqRat–DyadicRatCore–Real finite packet with its selector rows, BHist *support,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *rows. It exposes no quotient stream, selected limit, host real equality, detached threshold family, ambient completeness principle, or second completion seal.*

Proof. Begin with Definition 392.37. It places every CauchyCriterion selector-facing read on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route before exposing selector output.

Apply Definition 392.39. The CauchyCriterion selector route and the UniformCauchyCriterion shared-threshold seal route factor through the same finite packet before a $\mathsf{Real}^{\uparrow}$ seal is exposed.

Use the StreamName finite-window request and coverage rows together with the shared Real-facing handoff. They make the selector rows total over the displayed finite windows rather than over an ambient completion object.

Finally read the UniformCauchyCriterion shared-threshold transport and non-escape boundary. The uniform route contributes no detached threshold family or second seal, so the common packet gives the total synchronizer route claimed above. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier selector seal synchronizer totality`

Theorem 392.42 (CauchyCriterion common selector refinement)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* CauchyCriterion$^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet, and let D be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose the three packets share the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *and finite selector rows at the completion-facing boundary. Then the Cauchy selector route, the uniform shared-threshold route, and the diagonal root-formal route determine one common finite selector-refinement packet over* BHist, hsame, Cont, Pkg, *and* NameCert.

The refinement packet exposes no quotient stream, selected host limit, host real equality, detached threshold family, ambient completeness principle, or second completion seal.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier common selector refinement`

Proof. First assemble the pairwise selector boundary. The uniform selector-seal compatibility theorem, Definition 392.39, identifies the CauchyCriterion selector route and the UniformCauchyCriterion shared-threshold route inside one finite packet over the shared StreamName–RegSeqRat–DyadicRatCore–Real rows.

Next pull the diagonal route back to the same boundary. The route-pullback theorem, Definition 392.40, restricts the diagonal root-formal seal route along the displayed finite rows and prevents it from exposing a detached completion object.

Finally apply synchronizer totality, Definition 392.41. Its synchronizer packet already carries the finite selector rows, BHist support, componentwise hsame transports, displayed Cont routes, Pkg provenance, and NameCert rows. The StreamName finite-window and diagonal root-formal references used above put all three routes on that same finite window, so repacking them gives the common selector-refinement packet and no excluded coordinate remains. □

Theorem 392.43 (CauchyCriterion TailCofinalitySchedule selector cofinality)**.** *Let $K = (W, M, T, Q, E, H, C, P, N$ be an accepted* CauchyCriterion$^{\uparrow}$ *packet, and let $S = (E_S, W_S, D_S, R_S, L_S, H_S, C_S, P_S, N_S)$ be an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet. Suppose the two packets share the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *finite-window selector rows, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. Then the accepted packets determine one cofinal finite selector route through the Cauchy rows $M, Q, T, W, E, H, C, P, N$ and the schedule rows $E_S, W_S, D_S, R_S, L_S, H_S, C_S, P_S, N_S$ before any* Real$^{\uparrow}$ *or completion seal is public.*

The route is the common StreamName–RegSeqRat–DyadicRatCore–Real finite packet with its selector request, finite StreamName window, dyadic threshold ledger, regular-rational readback, Real seal boundary, and structural support rows. It exports no quotient stream, selected host limit, host real equality, countable-choice schedule, ambient completeness principle, hidden tail oracle, or second selector-budget source.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier tailcofinalityschedule selector cofinality`

Proof. First read the Cauchy side through Definition 392.37. This places every selector-facing CauchyCriterion$^{\uparrow}$ read on the displayed StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ rows before a selector output or Real seal can be inspected.

Next apply Definition 392.42. The Cauchy selector route is already repacked with the uniform and diagonal routes over the same finite rows, with support carried by BHist, componentwise hsame, displayed Cont, Pkg, and NameCert.

Now pass through the sibling schedule. The shared-tail budget factorization theorem, Definition 681.10, places the schedule packet on the finite route through E_S, W_S, D_S, R_S, L_S and the structural rows H_S, C_S, P_S, N_S, with no second tail-budget source.

Finally use Definition 681.14. It puts the completion-facing diagonal read on the same schedule route before any public $\mathsf{Real}^{\uparrow}$ or completion seal is available. Since both sides consume the displayed StreamName–RegSeqRat–DyadicRatCore–Real rows with the same support inventory, the Cauchy selector packet and the tail cofinality schedule determine one cofinal finite route, and the excluded quotient, selected-limit, host-equality, choice, completeness, hidden-oracle, and second-budget coordinates have no carrier slot on that route. □

Theorem 392.44 (CauchyCriterion Real-window diagonal selector lock)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let D be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and finite selector rows at the completion-facing boundary. Then every* $\mathsf{Real}^{\uparrow}$ *window selector read factors through one common finite packet over* BHist, hsame, Cont, Pkg, *and* NameCert.

The common packet contains no quotient stream, selected host limit, host real equality, detached threshold family, ambient completeness principle, or second seal.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real window diagonal selector lock`

Proof. First read the CauchyCriterion selector boundary through Definition 392.37. This places every Real-facing budget-selector read on the displayed StreamName–RegSeqRat–DyadicRatCore–Real rows before any external limit or completion coordinate can be inspected.

Next apply uniform selector-seal compatibility and route pullback, Definition 392.39 and Definition 392.40. The uniform shared-threshold route and the diagonal root-formal route are both restricted to the same finite selector surface.

Now use synchronizer totality, Definition 392.41. It supplies the total route that aligns the CauchyCriterion selector surface with the UniformCauchyCriterion shared-threshold surface over the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally invoke the common selector refinement theorem, Definition 392.42. Carrier-coordinate exhaustion leaves exactly the shared finite packet and removes every coordinate listed outside it, so each Real-window selector read is locked to that packet. □

Theorem 392.45 (CauchyCriterion four-face selector readback determinacy)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let D be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and finite selector rows at the completion-facing boundary. Then any two* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer readbacks preserving the common selector-refinement packet of Definition 392.42 and the same* BHist, hsame, Cont, Pkg, *and* NameCert *support rows determine the same finite selector-readback packet before any seal is read.*

The determinacy claim does not identify quotient streams, selected host limits, host real values, countable-choice selectors, ambient completeness witnesses, or a second seal source.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier four face selector readback determinacy`

Proof. First build the four-face boundary. Apply Definition 392.37 to place every CauchyCriterion selector-facing read on the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, and use Definition 392.38 to return the finite selector section without leaving those rows.

Next synchronize the two external routes. The uniform compatibility theorem, Definition 392.39, puts the shared-threshold route on the same selector surface, while the pullback theorem, Definition 392.40, restricts the diagonal root-formal route to that surface before a completion seal is exposed.

Now apply synchronizer totality, Definition 392.41, and then the common refinement theorem, Definition 392.42. Together they package the common finite selector rows with exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support.

Finally use terminal non-escape, Definition 392.46. Since both consumer readbacks preserve the same refinement packet and the same support rows, neither can add an extra selector coordinate after the common packet is fixed. The excluded quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness witness, and second seal source therefore have no coordinate along which the two readbacks could diverge. □

Theorem 392.46 (CauchyCriterion selector-refinement terminal non-escape)**.** *Let $K = (W, M, T, Q, E, H, C, P, N$ be an accepted* CauchyCriterion$^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet, and let D be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose the three packets share the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, and finite selector rows at the completion-facing boundary. Once the common selector-refinement packet of Definition 392.42 is assembled, every terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer factors through that packet.*

Consequently no terminal consumer exports quotient streams, selected host limits, host real equality, detached threshold families, ambient completeness, or a second completion seal.

Proof. First use Definition 392.37. It places the CauchyCriterion terminal read behind the displayed StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ selector boundary.

Next combine Definition 392.39 with Definition 392.40. The uniform shared-threshold route and the diagonal root-formal route both restrict to the same selector-seal surface, with support carried by BHist, hsame, Cont, Pkg, and NameCert.

Now apply Definition 392.41. The synchronizer is total on the finite selector rows, so any Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer reaching the boundary has a route through the common packet.

Finally read the assembled packet through Definition 392.42. Since that theorem has already removed quotient streams, selected host limits, host real equality, detached threshold families, ambient completeness, and second completion seals from the shared packet, the same exclusions hold for terminal consumers. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier selector refinement terminal nonescape`

Theorem 392.47 (CauchyCriterion diagonal selector terminality)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* CauchyCriterion$^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet, and let D be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose the three packets share the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, and finite selector rows at the completion-facing boundary. Then the common selector-refinement packet of Definition 392.42 is terminal among all* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumers preserving those displayed rows.*

Equivalently, every such consumer factors through the same selector-refinement packet before any terminal seal is read, and no consumer preserving the displayed rows can add quotient streams, selected host limits, host real equality, countable-choice selectors, ambient completeness, or a second seal source.

Proof. First assemble the shared packet by Definition 392.42. That theorem composes the CauchyCriterion selector route, the UniformCauchyCriterion shared-threshold route, and the DiagonalLimitCompatibility route over the common $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and finite selector rows.

Next use the budget-selector boundary Definition 392.37, the route pullback Definition 392.40, and synchronizer totality Definition 392.41. These rows place every Real or completion-facing read that preserves the displayed rows inside the same finite selector surface.

Finally apply terminal non-escape, Definition 392.46. Since every terminal consumer already factors through the common packet, the packet is terminal among consumers preserving the shared rows. The excluded quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness, and second seal source coordinates are absent from the assembled packet and cannot be supplied by a terminal consumer. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier diagonal selector terminality`

Theorem 392.48 (CauchyCriterion finite selector exhaustion)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let $U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let D be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and finite selector rows at the completion-facing boundary. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer preserving those displayed rows factors through exactly the finite selector-refinement packet of Definition 392.42.*

Conversely, any route through that selector-refinement packet supplies all selector data available to such consumers: the common finite selector rows, the BHist *support, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the* NameCert *rows. No quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness witness, detached threshold family, or second seal source is available to a consumer preserving only the displayed rows.*

Proof. First apply the boundary and return rows. The budget-selector boundary Definition 392.37 places every selector-facing read on the displayed StreamName–RegSeqRat–DyadicRatCore–Real surface, and the finite-request section return Definition 392.38 identifies the finite selector section before any $\mathsf{Real}^{\uparrow}$ seal is exposed.

Next align the seal-facing routes. The uniform compatibility theorem Definition 392.39 and the pullback theorem Definition 392.40 put the CauchyCriterion, UniformCauchyCriterion, and DiagonalLimitCompatibility routes on the same finite structural rows carried by BHist, hsame, Cont, Pkg, and NameCert.

Now use synchronizer totality and common refinement. The synchronizer theorem Definition 392.41 makes the finite selector surface total for the shared packets, while Definition 392.42 repacks that surface as one selector-refinement packet over exactly the displayed support rows.

Finally apply diagonal selector terminality, Definition 392.47. Any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer preserving the displayed rows must factor through the same refinement packet. Inspecting the packet coordinates gives the converse: a route through the packet provides precisely the selector rows and support rows listed in the statement, and every omitted object has no coordinate from which such a consumer could read it. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite selector exhaustion`

Theorem 392.49 (CauchyCriterion finite-observation budget selector pullback)**.** *Let $K = (W, M, T, Q, E, H, C, P,$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ be an*

accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Suppose the two packets share the displayed* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows at the selector-to-seal boundary, and suppose their* hsame, Cont, Pkg, *and* NameCert *support rows are compatible componentwise. Then the CauchyCriterion selector route and the finite observation-budget selector route determine the same finite pullback surface from selector request to* Real$^{\uparrow}$ *seal.*

The pullback surface contains exactly the displayed selector request, finite StreamName window, dyadic readback, RegSeqRat handoff, Real seal, transport, continuation, package, and local naming rows. It refuses quotient streams, selected host limits, host real equality, countable choice, hidden selectors, ambient completeness, and any second completion seal.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite observation budget selector pullback`

Proof. First place the CauchyCriterion side on the selector boundary. Apply Definition 392.37 and Definition 392.40; these identify the displayed StreamName, DyadicRatCore, RegSeqRat, and Real rows used before the seal is read.

Next make the CauchyCriterion selector total on that surface. Use Definition 392.41 and Definition 392.48 to show that every consumer preserving the displayed rows factors through the same finite selector-refinement packet.

Now read the finite observation-budget selector packet. The obligations of Definition 782.2 expose the same kind of finite budget, window, dyadic, handoff, seal, transport, continuation, package, and local naming rows. Route determinacy, Definition 782.7, forces the selector request to reach the Real seal along that displayed chain.

Finally apply the diagonal consumer boundary, Definition 782.10. Because both accepted packets preserve the same support rows, the two chains meet in one finite pullback surface. The excluded quotient stream, selected host limit, host real equality, countable-choice, hidden-selector, ambient completeness, and second-seal objects have no coordinate in either packet, so they cannot appear in the pullback. □

Theorem 392.50 (CauchyCriterion finite-observation budget selector uniqueness)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* CauchyCriterion$^{\uparrow}$ *packet, and let* $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ *and* $F' = (B', S', V', D', R', E_{F'}, H_{F'}, C_{F'}, P_{F'}, N_{F'})$ *be accepted* FiniteObservationBudgetSelector$^{\uparrow}$ *packets. Suppose both* F *and* F' *share the same packet* K*, the same displayed* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows at the selector-to-seal boundary, and componentwise compatible* hsame, Cont, Pkg, *and* NameCert *support rows. Then the two selector-to-seal pullback surfaces are the same finite BEDC packet before any* Real$^{\uparrow}$ *seal is read.*

Equivalently, the shared surface is determined by the selector request, finite StreamName window, dyadic readback, RegSeqRat handoff, pre-seal Real row, transport, continuation, package, and local naming rows. It carries no quotient streams, selected host limits, host real equality, countable choice, hidden selectors, ambient completeness, or second completion seals.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite observation budget selector uniqueness`

Proof. First apply the pullback theorem Definition 392.49 to the pair (K, F) and again to the pair (K, F'). Both applications land on a finite selector-to-seal surface with exactly the displayed StreamName, DyadicRatCore, RegSeqRat, Real, transport, continuation, package, and naming rows.

Next use finite selector exhaustion, Definition 392.48. Because both observation-budget packets preserve the same CauchyCriterion packet and the same displayed rows, every completion-facing consumer factors through the same selector-refinement packet before any Real seal is exposed.

It remains to identify the observation-budget routes into that packet. Route determinacy, Definition 782.7, forces each accepted selector request to reach the Real row along the displayed finite chain, and the diagonal consumer boundary, Definition 782.10, keeps terminal consumers on that same chain.

Thus the two accepted packets have no remaining coordinate on which their pullback surfaces can differ. Since quotient streams, selected host limits, host real equality, countable choice, hidden selectors, ambient completeness, and second completion seals are excluded from the shared packet, they cannot serve as witnesses of a second selector-to-seal surface. □

Theorem 392.51 (CauchyCriterion exact-boundary selector handoff)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let $X = (L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X, P_X, N_X)$ be an accepted* RealCompletionExactBoundaryUp *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows at the selector-to-seal boundary. Then the CauchyCriterion selector-to-seal surface factors through the exact-boundary packet X before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is read.*

The handoff exposes only the selector request, finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal Real boundary, and the shared hsame, Cont, Pkg, *and* NameCert *support rows. It refuses quotient streams, selected host limits, host real equality, countable choice, ambient completeness, and any second seal source.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier exact boundary selector handoff`

Proof. First apply Definition 392.49. The accepted CauchyCriterion packet and the accepted finite observation-budget selector packet meet on a single selector-to-seal pullback surface with the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows.

Next use Definition 392.48. It shows that every completion-facing consumer preserving those rows must pass through the same finite selector-refinement packet before a Real or Completion seal is read.

Now apply Definition 1120.6. The accepted exact-boundary packet exports exactly the StreamName, RegSeqRat, DyadicRatCore, and terminal Real faces on the route $L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X$ so the selector-refinement surface has only that exact-boundary consumer route available.

Finally use Definition 1120.5. The terminal Real seal cannot be read outside the exact-boundary packet. Since the pullback and exhaustion steps already removed quotient streams, selected host limits, host real equality, countable choice, ambient completeness, hidden selectors, and second seal sources, the selector-to-seal surface factors through the exact-boundary packet before any terminal Real seal is consumed. □

Theorem 392.52 (CauchyCriterion StreamName–Real budget selector terminal boundary)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let $X = (L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X, P_X$ be an accepted* RealCompletionExactBoundaryUp *packet. Suppose the packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows. Then every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *selector read of the shared surface factors through the exact-boundary selector handoff of Definition 392.51.*

Conversely, that handoff supplies all selector data available to such a terminal read: the selector request, finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal Real boundary, and the displayed hsame, Cont, Pkg, *and* NameCert *support rows. It supplies no quotient stream, selected host limit, host real equality, countable choice, ambient completeness, hidden selector, or second seal source.*

Proof. First apply Definition 392.49. The CauchyCriterion packet and the finite observation-budget selector packet meet on the displayed finite selector-to-seal surface.

Next use Definition 392.50. The finite surface is determined by the shared StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows, so no second selector surface remains for a terminal read.

Now apply Definition 392.51. It routes the shared selector surface through the accepted exact-boundary packet before a terminal Real or Completion selector read can consume the seal rows.

The exact-boundary inventory is fixed by Definition 1120.6. Finally use Definition 1120.5 to block any terminal seal escape. Thus the handoff gives exactly the selector data available to the read, and all excluded host, quotient, choice, completeness, hidden-selector, and second-seal coordinates remain outside the packet. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier streamname real budget selector terminal boundary`

Theorem 392.53 (CauchyCriterion selector-budget three-way terminality)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* CauchyCriterion$^{\uparrow}$ *packet, let* $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ *be an accepted* FiniteObservationBudgetSelector$^{\uparrow}$ *packet, and let* $X = (L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X, P_X, N_X)$ *be an accepted* RealCompletionExactBoundaryUp *packet. Suppose the three packets share the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows. Then the finite-request section, the finite observation-budget selector, and the exact-boundary handoff determine one terminal selector-to-seal packet before any terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *read.*

That terminal packet contains exactly the displayed selector request, finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal Real boundary, and the shared hsame, Cont, Pkg, *and* NameCert *support. It contains no quotient stream, selected host limit, host real equality, countable choice, ambient completeness, hidden selector, or second seal source.*

Proof. First apply Definition 392.38. The accepted CauchyCriterion packet returns its finite request section on the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows.

Next apply Definition 392.42. This places the finite request section on the common selector-refinement packet shared by the CauchyCriterion route and the completion-facing rows.

Now use Definition 392.50. It shows that the accepted finite observation-budget selector has no second packet with the same displayed rows, so the selector component of the terminal surface is unique.

Finally apply Definition 392.52. The exact-boundary handoff is the only terminal selector-to-seal packet available before a Real$^{\uparrow}$ or Completion$^{\uparrow}$ read. Its inventory already excludes quotient streams, selected host limits, host real equality, countable choice, ambient completeness, hidden selectors, and second seal sources, so the three presentations determine the same terminal packet. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier selector budget three way terminality`

Theorem 392.54 (CauchyCriterion terminal selector-seal no-choice exhaustion)**.** *Let* $K = (W, M, T, Q, E, H, C, P,$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let* $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let* $X = (L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X, P_X, N_X)$ *be an accepted* RealCompletionExactBoundaryUp *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows. Then every terminal selector-to-seal read of the shared surface factors through, in order, the finite-request section, the finite observation-budget selector, the exact-boundary handoff, and the three-way terminal packet of Definition 392.53.*

The resulting consumer-visible boundary exports only the selector request, finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal Real boundary, and the shared hsame, Cont, Pkg, *and* NameCert *support rows. It exports no quotient stream, selected host limit, host real equality, countable-choice schedule, ambient completeness, hidden selector, or second seal source.*

Proof. First apply Definition 392.38. The accepted CauchyCriterion packet returns the finite request section on the shared StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows.

Next use Definition 392.50. This pins the finite observation-budget selector to the same displayed rows and removes any second selector packet before a terminal read can consume the seal surface.

Now apply Definition 392.51. The finite selector route is handed to the accepted exact-boundary packet, so every terminal selector-to-seal read must pass through the displayed RealCompletionExactBoundaryUp rows before a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal is read.

Finally combine Definition 392.52 with Definition 392.53. The first theorem gives the terminal StreamName–Real boundary, while the second identifies the finite-request, finite-budget, and exact-boundary presentations as one terminal packet.

The carrier inventory of that packet is already exhausted by the displayed selector request, finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal Real boundary, and shared support rows. Therefore a terminal read has no exported quotient stream, selected host limit, host real equality, countable-choice schedule, ambient completeness, hidden selector, or second seal source available outside the stated route. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier terminal selector seal nochoice exhaustion`

Theorem 392.55 (CauchyCriterion budget-selector section exactness)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, let* $F = (B, S, V, D, R, E_F, H_F, C_F, P_F, N_F)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let* $X = (L, K_X, J_X, S_X, W_X, R_X, D_X, E_X, H_X, C_X, P_X, N_X)$ *be an accepted* RealCompletionExactBoundaryUp *packet. Suppose the three packets share the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows over the same* BHist *support. Then every finite observation request routed through the displayed* $\mathsf{StreamName}^{\uparrow}$–RegSeqRat–$\mathsf{Real}^{\uparrow}$ *selector surface has a unique budget-selector section.*

Replaying that section through the selector-seal route returns the same finite StreamName window, RegSeqRat handoff, DyadicRatCore tolerance row, terminal $\mathsf{Real}^{\uparrow}$ *boundary, and seal row carried by the original request. No quotient stream, host real equality, selected-limit oracle, countable-choice row, ambient completeness witness, hidden selector, or second seal source is introduced.*

Proof. First apply Definition 392.38. It returns the finite request section on the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows over the given BHist support.

Next use Definition 392.49. The accepted finite observation-budget selector and the accepted CauchyCriterion packet meet on the same selector-to-seal pullback surface, so the returned section is already routed through the displayed StreamName$^{\uparrow}$–RegSeqRat–Real$^{\uparrow}$ surface.

Now apply Definition 392.50. This proves that no second accepted budget-selector section with the same BHist, hsame, Cont, Pkg, and NameCert rows can determine a different finite window or seal-facing row.

Finally combine Definition 392.41 with Definition 392.54. The synchronizer makes replay through the selector-seal route total on the finite surface, and no-choice exhaustion says the replay has no exported coordinate for a quotient stream, host real equality, selected-limit oracle, countable-choice row, ambient completeness witness, hidden selector, or second seal source. Therefore replay returns exactly the original finite window and seal row. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier budget selector section exactness`

Theorem 392.56 (CauchyCriterion budget-selector terminal route exhaustion)**.** *Let $K = (W, M, T, Q, E, H, C, P,$ be an accepted* CauchyCriterion$^{\uparrow}$ *budget-selector route over the displayed* BHist *support. Among routes that share the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, finite selector, uniform threshold, and diagonal root-formal surfaces, the accepted route is terminal. Every such route factors through the finite rows already named by the boundary theorem, finite-request section return, uniform compatibility, diagonal pullback, common selector refinement, finite selector exhaustion, terminality, and budget-selector section exactness in this chapter.*

Consequently no accepted route over the same surfaces can add a quotient stream, host real equality, countable-choice selector, ambient completeness witness, detached threshold, or second seal. Any consumer that asks for one of those rows has left the CauchyCriterion$^{\uparrow}$ *budget-selector route rather than extending it.*

Proof. First apply the boundary theorem and finite-request section return. These place the accepted route on the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, continuation, package, and local naming rows.

Next use uniform compatibility and the diagonal pullback. They keep the uniform threshold and diagonal root-formal surfaces on the same finite support, so a route sharing those surfaces cannot detach its threshold or introduce a second root source.

Now apply common selector refinement and finite selector exhaustion. The selector row is already fixed by the shared finite request and by the accepted finite observation-budget selector, leaving no second selector packet with the same displayed rows.

Finally combine terminality with Definition 392.55. The section exactness theorem returns the same finite window and seal row under replay, while terminality blocks quotient streams, host real equality, countable-choice selectors, ambient completeness, detached thresholds, and second seals. Thus every shared-surface route factors through the named finite rows and no additional route coordinate remains. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier budget selector terminal route exhaustion`

Theorem 392.57 (CauchyCriterion StreamName Real budget bridge readiness)**.** *Every bridge-ready* CauchyCriterion$^{\uparrow}$ *selector route factors through the displayed* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$

readback, $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, finite-observation budget selector,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Apply Definition 392.56. It makes the accepted selector route terminal over the named finite surfaces.

Then read the boundary, section-return, and exactness theorems already cited there. They keep bridge readiness on $W, M, T, Q, E, H, C, P, N$ and add no quotient stream, hidden selector, countable-choice row, or ambient Real-completeness premise. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier streamname real budget bridge readiness`

Theorem 392.58 (CauchyCriterion selector budget formal target boundary)**.** *A later formal target for the* $\mathsf{CauchyCriterion}^{\uparrow}$ *StreamName-to-Real budget selector may consume only the finite selector-budget boundary named in Definition 392.57. It may not import quotient stream equality, countable choice, ambient Real completeness, or a hidden selector.*

Proof. Use bridge readiness to enumerate the admissible inputs. The formal target receives the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal, finite selector, transport, replay, provenance, and naming rows.

Now apply terminal route exhaustion. Any additional quotient equality, choice selector, completeness witness, or hidden selector would be a new coordinate outside the terminal finite route, so it is not part of the formal boundary. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier selector budget formal target boundary`

Theorem 392.59 (CauchyCriterion finite-window Real seal handoff)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Every accepted finite-window budget-selector read of* K *that is handed to a* $\mathsf{Real}^{\uparrow}$*-facing consumer, a* $\mathsf{Completion}^{\uparrow}$*-facing consumer, a* $\mathsf{RealCauchyCompletion}^{\uparrow}$*-facing consumer, or a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow}.$$

The route reads the CauchyCriterion rows M, Q, T, W, E *before the consumer sees the selected finite window, the regular-rational readback, the dyadic tolerance ledger, the Real seal row, and the limit-seal boundary. Its support rows are precisely the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by* H, C, P, N*.*

No accepted handoff exports a quotient stream, a selected limit, host real equality, a hidden tail oracle, ambient completeness, or a completion endpoint outside the displayed finite route.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite window real seal handoff`

Proof. Project the accepted CauchyCriterion carrier. The finite selector and budget rows of the chapter expose the route M, Q, T, W, E together with the structural rows H, C, P, N; no consumer can read beyond those rows before the selector surface has been displayed.

Next use the StreamName and RegSeqRat handoff route. The CauchyCriterion-to-StreamName theorem Definition 392.26 places the finite window, regular-rational readback, dyadic ledger, and Real seal on the same packet. The selector boundary Definition 392.37 then forces every finite-window budget-selector read through that route before a Real-facing or completion-facing consumer receives it.

Now attach the completion and seal boundary. The completion-boundary theorem Definition 392.33 confines $\mathsf{Completion}^{\uparrow}$ and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers to the same $M, Q, T, W, E, H, C, P, N$ surface, while the StreamName CauchyLimitSeal pullback Definition 126.396 and the public CauchyLimitSeal route boundary Definition 386.8 put the limit-seal consumer after the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal row.

Finally apply the non-escape clauses of the cited boundaries. The carrier has no coordinate for quotient streams, selected host limits, host real equality, hidden tail oracles, ambient completeness, or an extra completion endpoint, so each accepted handoff factors through the displayed finite route and nowhere else. □

Theorem 392.60 (CauchyCriterion limit-seal route stability)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet, and let K' be any displayed replay of the same finite-window handoff that preserves the rows $M, Q, T, W, E, H, C, P, N$ through the visible* BHist *packet, componentwise* hsame *transports, and* Cont *replay. Then the replayed handoff preserves the same route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow}.$$

No displayed replay of this route can substitute a quotient stream, hidden tail oracle, host real equality, selected limit, ambient completeness principle, or completion endpoint outside the rows already carried by K.

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier limit seal route stability`

Proof. Start with the finite handoff theorem. By Definition 392.59, every accepted finite-window read of K reaches the limit-seal boundary only through $M, Q, T, W, E, H, C, P, N$ and the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow}$ route.

Next identify the route coordinates. The RegSeqRat and StreamName handoff Definition 392.26 binds the regular readback to the StreamName window, while Definition 392.37 and Definition 392.33 keep the budget selector and completion-facing reads on the same finite rows.

Finally attach the seal side. Definition 126.396 pulls the seal read back to the StreamName budget surface, and Definition 386.8 confines public CauchyLimitSeal consumption to that surface. Since K' preserves exactly the displayed BHist, hsame, and Cont rows, carrier-coordinate exhaustion leaves no coordinate from which a quotient stream, hidden tail oracle, host real equality, selected limit, ambient completeness principle, or off-route completion endpoint could be read. □

Theorem 392.61 (CauchyCriterion Real L10 terminal-window sufficiency)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet whose finite-window budget-selector read follows the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow}.$$

Then the same displayed rows $M, Q, T, W, E, H, C, P, N$, together with their visible BHist, hsame, Cont, Pkg, *and* NameCert *coordinates, suffice to supply the Real L10 finite-window witness surface consumed by the public boundary.*

Conversely, any Real L10 public-boundary read sourced through K recovers exactly the selector rows $M, Q, T, W, E, H, C, P, N$ and no additional coordinate. The sufficiency and recovery clauses exclude quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic packages, order packages, and bridge rows.

Proof. First use Definition 392.59. It places every accepted finite-window budget-selector read of K on the displayed route and exposes the rows $M, Q, T, W, E, H, C, P, N$ before any Real-facing, completion-facing, or limit-seal consumer can read the packet.

Next apply Definition 392.60. The route remains fixed under displayed replay of the same BHist, hsame, and Cont rows, so the terminal-window read cannot be replaced by a quotient stream, hidden tail oracle, host equality row, selected limit, or completeness principle.

Now attach the Real L10 boundary. The lattice budget cover Definition 126.368 identifies the StreamName, RegSeqRat, DyadicRatCore, and Real seal faces as one finite route, with only the displayed support coordinates. Feeding that route into Definition 38.464 supplies exactly the finite-window witness surface consumed by the Real L10 public boundary.

For the converse, start from a Real L10 public-boundary read sourced through K. By Definition 38.466, the boundary can be sourced only through the ordered finite-window route displayed by the Real L10 lattice. Comparing this route with the CauchyCriterion handoff and its replay stability recovers precisely $M, Q, T, W, E, H, C, P, N$.

The exclusion clause is inherited from the cited route and L10 boundary theorems. None of those packets carries quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, arithmetic packages, order packages, or bridge rows, so no such row can be introduced at the terminal public boundary. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real l10 terminal window sufficiency`

Theorem 392.62 (CauchyCriterion finite-window terminal coverage)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Any terminal CauchyCriterion consumer whose read reaches a* $\mathsf{Real}^{\uparrow}$ *seal must pass through the displayed finite-window route consisting of the Cauchy modulus row, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *ledger carried by M, W, Q, E, H, C, P, N. No terminal read can replace that route by a quotient stream, hidden tail oracle, host real equality, selected limit, countable-choice row, ambient completeness principle, metric-completeness row, arithmetic package, order package, bridge row, or completion endpoint outside the displayed packet.*

Proof. First apply Definition 392.59. It places every accepted finite-window handoff on the route through the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic tolerance ledger, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary, with H, C, P, N supplying the visible transport, replay, provenance, and local naming rows.

Next apply Definition 392.60. Any displayed replay preserving $M, Q, T, W, E, H, C, P, N$ stays on the same route, so a terminal consumer cannot substitute a hidden stream source, off-route seal, or host equality coordinate.

Finally use Definition 392.61. The Real L10 public-boundary read consumes exactly the displayed terminal-window surface and recovers the same selector rows. Combining the three cited routes therefore covers every terminal CauchyCriterion read that reaches a $\mathsf{Real}^{\uparrow}$ seal, and the absent rows listed in the statement have no carrier coordinate from which they could be read. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite window terminal coverage`

Theorem 392.63 (CauchyCriterion Real seal finite-route uniqueness)**.** *Let $K = (W, M, T, Q, E, H, C, P, N)$ and $K' = (W', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packets. Suppose their public-boundary reads expose the same rows $M, Q, T, W, E, H, C, P, N$ through the same visible* BHist *packet, componentwise* hsame *transports,* Cont *replay, and matching* Pkg *and* NameCert *coordinates. Then the two Real L10 public-boundary reads determine the same terminal finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow}.$$

No quotient stream, host equality, selected limit, countable-choice row, ambient completeness principle, or bridge row can distinguish the two reads at the terminal public boundary.

Proof. First expose the common route. By Definition 392.59, each accepted read reaches the Real and limit-seal boundary only through the displayed rows $M, Q, T, W, E, H, C, P, N$ and the finite route from $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{CauchyLimitSeal}^{\uparrow}$.

Next compare the two displayed reads under replay. The hypothesis identifies the visible BHist, componentwise hsame, Cont, Pkg, and NameCert coordinates, so Definition 392.60 keeps both reads on the same replayed route rather than on two independently selected completion surfaces.

Now attach the public Real face. The lattice budget cover Definition 126.368 makes the StreamName, RegSeqRat, DyadicRatCore, and Real faces a single finite support surface. The Real L10 witness determinacy theorem Definition 38.464 then identifies the public-boundary witnesses carried by the two reads on that shared surface.

Finally use source-route exactness. By Definition 38.466, a Real L10 public-boundary read has no source outside the ordered finite-window route already exposed by the CauchyCriterion handoff. Hence the two reads determine the same terminal finite route, and the absent rows listed in the statement have no coordinate from which a distinction could be read. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier real seal unique finite route`

Corollary 392.64 (CauchyCriterion finite-window Real seal handoff determinacy)**.** *Let* $K = (W, M, T, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Fix the displayed rows* $M, Q, T, W, E, H, C, P, N$ *through the visible* BHist *packet, componentwise* hsame *transports,* Cont *replay, and matching* Pkg *and* NameCert *coordinates. Then every accepted consumer-facing use of the finite-window Real seal handoff of* K *determines the same terminal route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow}.$$

In particular, a $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{Completion}^{\uparrow}$*-facing,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*-facing, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing consumer must use the same finite-route normal form and the same source-route exactness verdict named by the chapter. No accepted use can introduce an alternate hidden coordinate, a duplicate classifier source, a quotient stream, host real equality, a selected limit, countable choice, ambient completeness, a bridge row, or a completion endpoint outside the fixed rows.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUp.CauchyCriterionCarrier finite window real seal handoff determinacy`

Proof. Take two accepted consumer-facing uses of the handoff after the rows $M, Q, T, W, E, H, C, P, N$ have been fixed. The needed transport comparison is the finite-route uniqueness result, Definition 392.63: its hypotheses are precisely the common visible BHist packet, componentwise hsame transports, Cont replay, and matching Pkg and NameCert coordinates over the fixed rows. Regard the second use as a displayed replay of the first along that cited comparison surface. By Definition 392.60, the replay cannot move the handoff away from the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow}$ route, and it cannot substitute any quotient stream, hidden tail oracle, host real equality, selected limit, ambient completeness principle, or off-route completion endpoint.

Now compare the two public-boundary reads. The hypotheses are exactly the row identifications required by Definition 392.63. Hence the two reads determine one terminal finite route, rather than two independently chosen consumer surfaces. The same comparison fixes the chapter's

terminal finite-route normal form, because the normal form is read only after the StreamName, RegSeqRat, DyadicRatCore, Real, and CauchyLimitSeal faces have been placed on that route.

For the exactness decision, use the same exhaustion of sources. The unique finite route already excludes any source outside the ordered finite-window route, while route stability keeps every replay on the same rows. Therefore an accepted consumer use has no coordinate from which to read a second classifier, a hidden exactness source, or an alternate completion endpoint. The only available decision is the source-route exactness verdict attached to the fixed finite-window Real seal handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterion}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source window, a $\mathsf{CauchyModulus}^{\uparrow}$ schedule, a $\mathsf{DyadicTailBound}^{\uparrow}$ finite tail ledger, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 392.16, Definition 392.17, Definition 392.19, Definition 392.21, and Definition 392.35 bind the carrier, classifier, BHist, hsame, Cont, Pkg, NameCert, dyadic tail ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal rows, completion-facing bridge, and representative downstream consumers to one finite constructive Cauchy criterion packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, ambient completeness, uniqueness of limits, or bridge checking.

Upgrade path: No further paper-axis theory step remains for this packet; formal verification and bridge checking remain separate verification-axis work.

Chapter 393

A Concrete Naming Certificate for $\mathsf{StreamDiagonalSelector}^{\uparrow}$

The $\mathsf{StreamDiagonalSelector}^{\uparrow}$ interface packages the finite diagonal selector row used between $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RealDiagonalCompletion}^{\uparrow}$. Following Bishop-style constructive diagonalization, a consumer precision request is answered by an explicit finite stream window, a regular-rational readback, a dyadic observation ledger, and a diagonal-completion packet. The certificate does not classify quotient streams, select a completed real by choice, or compare host real equality.

The selector is the finite routing surface between chapter 371 and chapter 359. It consumes the RegularCauchyName stream-window and dyadic-readback rows, then hands the same selected window to the RealDiagonalCompletion packet before any $\mathsf{Real}^{\uparrow}$ seal can be read.

Definition 393.1 (StreamDiagonalSelector carrier)**.** *A* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, \mu, W, R, D_H, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite stream windows that may be inspected by the diagonal consumer;*
2. *μ is the displayed selector row assigning each unary precision request to the diagonal window index read from S;*
3. *W is the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window, including the member index and observation index used by the diagonal read;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the rational observation extracted from the selected window;*
5. *D_H is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the endpoint, radius, comparison, and precision rows attached to the readback;*
6. *H records the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet that consumes the same selected window, readback, and dyadic ledger;*
7. *C records the displayed* Cont *routes from the selector request to S, μ, W, R, D_H, and H;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{StreamDiagonalSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the listed finite rows, their componentwise hsame *transports, the displayed* Cont *routes, package provenance, and local naming row. It has no coordinate for a quotient stream, choice-selected limit, external real value, or host equality test.*

Lean4 ✓ `BEDC.Derived.StreamDiagonalSelectorUp.TasteGate.StreamDiagonalSelectorTasteGate single carrier alignment`

Theorem 393.2 (StreamDiagonalSelector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StreamDiagonalSelector}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation: the finite packet of Definition 393.1 stores the stream schedule, selector row, selected window, rational readback, dyadic observation ledger, diagonal-completion packet, transport routes, package provenance, and local naming row;*
2. *selector determinacy: for each displayed precision request, the selected window is the one read from μ and S, so two accepted packets with the same transported selector rows expose the same diagonal window coordinate;*
3. *finite-window transport: the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger transport together along the componentwise* hsame *and* Cont *rows recorded in C;*
4. *diagonal handoff exactness: the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet H consumes exactly the selected window W, readback R, dyadic ledger D_H, package provenance P, and local naming row N;*
5. *non-escape: every consumer-visible read is exhausted by the displayed selector packet and exports no quotient stream, no choice-selected limit, no host real equality, and no external completed-real carrier.*

Lean4 ✓ `BEDC.Derived.StreamDiagonalSelectorUp.StreamDiagonalSelectorTasteGate single carrier alignment`

Proof. Carrier habitation is obtained by projecting the finite history packet. The schedule S, selector row μ, selected window W, readback R, dyadic ledger D_H, diagonal packet H, continuation routes, package provenance, and local naming row are all named coordinates.

Selector determinacy follows because μ is part of the displayed packet. A precision request cannot select a different window without leaving the recorded route through S and μ, and the classifier exposes only the transported versions of those two rows.

For finite-window transport, move S, μ, and W by the componentwise hsame rows and replay the Cont routes in C. The readback R and ledger D_H are attached to that same selected window, so they move with it as a single finite observation.

The diagonal handoff is the projection to H. Since H is stored beside W, R, D_H, P, and N, the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ consumer receives precisely those rows and no hidden selector.

Finally inspect the carrier shape. There is no field for quotient stream equality, a choice-selected limit, host real equality, or an external completed real, so the local naming certificate cannot export any such datum. □

Theorem 393.3 (StreamDiagonalSelector RealUp handoff)**.** *For every accepted* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *carrier $D = (S, \mu, W, R, D_H, H, C, P, N)$, every downstream* $\mathsf{Real}^{\uparrow}$ *read factors through the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D_H, and the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet H. The handoff supplies a finite real-facing observation route and does not export a quotient stream, a choice-selected limit, a host real value, or host real equality.*

Lean4 ✓ `BEDC.Derived.StreamDiagonalSelectorUp.StreamDiagonalSelectorPacket real handoff`

Proof. Start from a downstream $\mathsf{Real}^{\uparrow}$ read and project the carrier. The only route from a precision request to a stream observation is the displayed path through S, μ, and the selected finite window W.

The selected window is then read through the regular rational row R. The dyadic ledger D_H records the endpoint, radius, precision, and comparison data consumed by that readback, so the observation is a finite BEDC row rather than an external stream value.

The $\mathsf{RealDiagonalCompletion}^{\uparrow}$ coordinate H repacks the same W, R, and D_H rows for the completion-facing boundary. Replaying the routes in C and the provenance in P keeps the downstream $\mathsf{Real}^{\uparrow}$ consumer inside the same local $\mathsf{NameCert}$ packet.

Since the carrier has no further coordinate beyond those rows, no quotient stream, choice-selected limit, host real value, or host real equality can be read from the handoff. □

Theorem 393.4 (StreamDiagonalSelector selector determinacy)**.** *If two accepted* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *reads pass through the same displayed selector row μ and the same* $\mathsf{StreamName}^{\uparrow}$ *schedule S, then they expose the same classifier-facing selected window coordinate. The determinacy is a statement about the visible selector packet and does not compare quotient streams, selected limits, host real values, or host equality.*

Lean4 ✓ `BEDC.Derived.StreamDiagonalSelectorUp.StreamDiagonalSelectorPacket selector determinacy`

Proof. Unpack the carrier of Definition 393.1. A downstream precision request reaches a selected window only by reading the schedule S and the selector row μ.

The NameCert obligation surface in Definition 393.2 records selector determinacy as one of the finite rows. Thus two reads through the same transported S and μ expose the same selected window coordinate to the classifier.

Finally apply Definition 393.3. The handoff has no alternate coordinate for a quotient stream, selected completed value, external real carrier, or host equality test, so no second selector source can alter that coordinate. □

Theorem 393.5 (StreamDiagonalSelector finite-window transport)**.** *The selected* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D_H, and the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *packet H transport together through the recorded* hsame *and* Cont *rows of an accepted* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *carrier.*

Proof. Unpack the accepted packet in Definition 393.1. The rows W, R, D_H, and H are stored as the selected finite observation route, and the row C stores the displayed continuation routes that connect them.

Use Definition 393.2. Its finite-window transport row states that the selected stream window, regular-rational readback, and dyadic ledger move together along the componentwise hsame and Cont rows.

The RealUp handoff of Definition 393.3 consumes the same W, R, D_H, and H rows. Therefore transport cannot detach the readback, ledger, or diagonal packet from the selected window. □

Theorem 393.6 (StreamDiagonalSelector handoff exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *handoff from an accepted* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *carrier consumes exactly the selected finite window W, readback row R, dyadic ledger D_H, diagonal-completion packet H, package row P, and local* $\mathsf{NameCert}$ *row N. It consumes no quotient stream, selected limit, host equality, or external completed-real carrier.*

Proof. Project the carrier of Definition 393.1. The real-facing rows after selector evaluation are precisely W, R, D_H, H, P, and N, with the recorded routes in C mediating access to them.

Apply Definition 393.2. Its handoff exactness and non-escape rows state that the diagonal packet consumes the selected finite window, readback, dyadic ledger, provenance, and local naming row without adding an external selector.

Finally use Definition 393.3. It identifies the downstream $\mathsf{Real}^{\uparrow}$ route with those same rows, so no quotient stream, selected completed value, host equality, or outside real carrier can be read by the handoff. □

Theorem 393.7 (StreamDiagonalSelector tail-budget readback)**.** *Let $D = (S, \mu, W, R, D_H, H, C, P, N)$ be an accepted* $\mathsf{StreamDiagonalSelector}^{\uparrow}$ *packet. If its selected finite window W is consumed by an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row, then the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *handoff H factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S, selector row μ, selected finite window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D_H,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the shared tail-threshold and observation-budget rows of the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet, and the local* $\mathsf{NameCert}$ *route N.*

Thus every such handoff remains a finite route through the selector packet and the tail-budget readback surface. It exports no quotient stream, selected host limit, ambient completeness principle, or host real equality.

Proof. Project the selector carrier. The only rows available to the diagonal consumer are the schedule S, selector row μ, selected window W, readback row R, dyadic ledger D_H, diagonal packet H, routes C, provenance P, and local name N.

Use selector determinacy, finite-window transport, and handoff exactness. These results keep the selected window attached to the same schedule and selector row, move W, R, D_H together through the displayed hsame and Cont data, and identify H as the packet consuming exactly those rows.

Now apply the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ observation gate and readback factorization to the budget row that consumes W. They route the window service through the shared tail threshold, observation budget, dyadic ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before any real-facing seal or completion-facing consumer is exposed.

Combining the two routes gives a single finite handoff from S, μ, W, D_H, R through the tail-budget rows to H and N. Since neither carrier stores quotient streams, selected host limits, ambient completeness, or host real equality, no such datum can be read from the composed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamDiagonalSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamDiagonalSelector}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ schedule, a displayed selector row, one selected finite stream window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, a $\mathsf{RealDiagonalCompletion}^{\uparrow}$ diagonal packet, displayed hsame and Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamDiagonalSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 393.1, Definition 393.2, and Definition 393.3 describe the seed finite diagonal selector certificate consumed by real-completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamDiagonalSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, a public standard bridge, or checked Lean implementation.

Upgrade path: Obligation closure requires checked carrier habitation, selector determinacy, finite-window transport, diagonal handoff exactness, and non-escape for $\mathsf{StreamDiagonalSelector}^{\uparrow}$.

Chapter 394

A Concrete Naming Certificate for $\mathsf{LocatedInterval}^{\uparrow}$

The $\mathsf{LocatedInterval}^{\uparrow}$ packet names a finite interval surface whose endpoints are carried as located real names while its inspection ledgers remain rational and dyadic. It consumes the dyadic refinement carrier of chapter 306 together with chapter 126, chapter 301, and chapter 38 for its endpoint-window and seal rows. In the Bishop–Bridges constructive-analysis setting, an interval is read through explicit endpoint names, rational cells, dyadic refinements, and finite real-name windows; it is not a host interval primitive and does not choose completed endpoint values.

Definition 394.1 (LocatedInterval carrier)**.** *A* $\mathsf{LocatedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet displayed in the same history;*
2. *U is the upper endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet displayed in the same history;*
3. *I is a* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger whose endpoint and containment rows give the rational windows currently enclosing the endpoint reads;*
4. *D is the optional finite family of* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement rows attached to selected rational cells;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the finite endpoint observations inspected from L and U;*
6. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used by those endpoint windows;*
7. *E records the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows for the lower and upper endpoint names;*
8. *H records the componentwise* hsame *transports for located endpoints, rational cells, dyadic refinements, stream windows, readbacks, and seal rows;*
9. *C records the displayed* Cont *read routes by which a finite consumer reaches exactly those endpoint, interval, refinement, window, readback, and seal rows;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the local rational source rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *provenance row.*

The carrier predicate is the assertion that these rows form one finite located interval packet. It has no coordinate for a host real interval, an endpoint choice functional, a quotient of real names, or an ambient order-completeness theorem.

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalTasteGate single carrier alignment`

Theorem 394.2 (LocatedInterval NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation, given by any displayed pair of* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows together with a rational cell ledger enclosing the inspected endpoint windows;*
2. *endpoint transport, which moves the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *rows componentwise through the stored* hsame *and* Cont *witnesses;*
3. *interval containment exactness, which says that every containment read is one of the rational cells in I or one of the dyadic refinements in D routed by the packet;*
4. *refinement handoff, which permits a* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row only as a finite refinement of a displayed* $\mathsf{RationalInterval}^{\uparrow}$ *cell;*
5. *non-escape of host real intervals, which forces every endpoint, containment, and seal read through L, U, I, D, S, R, E, H, C, P, and N.*

These rows are exactly the naming-certificate surface for located real endpoint data bound to rational and dyadic interval ledgers.

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket semantic name certificate`

Proof. Carrier habitation is obtained by packaging the two displayed endpoint names with one rational cell ledger and the finite observation windows used to read them. The packet stores those rows directly, so no external interval carrier is needed.

Endpoint transport is componentwise. The lower endpoint row and the upper endpoint row each move along their own hsame comparison and the displayed Cont route that reads the corresponding finite window.

Containment exactness is a projection from the carrier. A rational containment read must come from I, and a dyadic containment read must come from one of the refinement rows in D; both are routed by the same history and package provenance.

For the refinement handoff, read a dyadic row only together with the rational cell it refines. Repacking that pair preserves the interval surface because the consumer still sees the same endpoints, window routes, and readback rows.

The non-escape row follows by inspecting the coordinates of Definition 394.1. There is no field from which a host interval, chosen endpoint value, quotient real, or ambient completeness principle could be exported. □

Theorem 394.3 (LocatedInterval endpoint transport)**.** *Let two accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packets be related by the displayed componentwise transport rows. Then the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint, the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger, every present* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{Real}^{\uparrow}$ *seal rows transport together as one located interval packet.*

Consequently a consumer of the transported packet can read only the transported endpoint rows and the transported interval ledgers carried by the same BHist, hsame, Cont, Pkg, *and local* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket endpoint transport`

Proof. Project both packets through Definition 394.1. The hypotheses give separate transport rows for the lower endpoint, upper endpoint, rational cell ledger, dyadic refinements, stream windows, readbacks, and seal rows.

Transport the two endpoint names first. Their $\mathsf{LocatedReal}^{\uparrow}$ windows and $\mathsf{Real}^{\uparrow}$ seal-facing rows move by the stated componentwise hsame witnesses and keep the same finite read routes.

Next transport the interval ledger. The rational cells in I and the dyadic refinement rows in D are carried by their own displayed comparisons, so each transported containment row remains attached to the transported endpoint windows.

Finally repack the transported rows with the transported package and local naming provenance. Since the carrier admits only these projected rows, the transported packet has no additional consumer route beyond the displayed endpoint and interval data. □

Theorem 394.4 (LocatedInterval carrier habitation)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, the carrier-habitation obligation for $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *is witnessed exactly by the displayed* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L *and* U*, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I*, the optional* $\mathsf{DyadicInterval}^{\uparrow}$ *family* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local certificate row* N*.*

No further carrier coordinate is required or available.

Proof. Project the packet through Definition 394.1. Each row named in J is part of the same finite BHist display, and the package provenance records that those rows are read under $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

The habitation witness is therefore the finite tuple J itself, read with its displayed routes. Since the carrier definition contains no host interval coordinate and no chosen endpoint value, the certificate cannot draw on any ambient interval object. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket carrier habitation`

Theorem 394.5 (LocatedInterval containment exactness)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every containment read accepted by the local $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *is routed by* C *to either the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I *or a present* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row in* D*. Endpoint movement used by such a read is the movement supplied by Definition 394.3.*

Thus containment exactness is a finite projection from J*, not a claim about all host intervals between two completed real values.*

Proof. Inspect the carrier coordinates. The only interval-containment rows available to a consumer are the rational cells in I and the dyadic refinement rows in D, and every admissible read must pass through the displayed Cont route C and the provenance P.

If the containment read is compared after endpoint transport, apply Definition 394.3 first. The lower and upper $\mathsf{LocatedReal}^{\uparrow}$ rows, their finite windows, the readbacks, and the seal-facing rows move together, so the transported containment route still lands in the transported copy of I or D.

There is no separate containment source in Definition 394.1. Exactness therefore follows by ruling out any read not projected from the listed finite rows. □

Theorem 394.6 (LocatedInterval dyadic refinement handoff)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, a $\mathsf{DyadicInterval}^{\uparrow}$ *row may be handed to a finite consumer only together with the* $\mathsf{RationalInterval}^{\uparrow}$ *cell in* I *that its displayed route refines. Under the endpoint movement of Definition 394.3, the handoff remains attached to the transported rational cell ledger and the transported endpoint windows.*

The handoff obligation is therefore internal to BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *over the rows of* J*.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket dyadic refinement handoff`

Proof. Start with a present dyadic refinement row in D. By Definition 394.1, that row is stored only as part of the same finite packet as the rational ledger I, with the route C displaying how the refinement is reached from the inspected interval data.

Transported endpoint reads are handled before the handoff. Apply Definition 394.3 to move L, U, I, D, S, R, and E through the componentwise hsame rows in H while preserving the consumer routes.

The consumer can then receive the dyadic row only with its rational parent cell and the same package provenance. No rule in the carrier permits a dyadic refinement detached from I or from the displayed finite history. □

Theorem 394.7 (LocatedInterval real-seal non-escape)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every seal-facing read accepted by the local $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *factors through the* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L *and* U*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *rows* E*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local row* N*. Endpoint movement is exactly the movement supplied by Definition 394.3.*

Consequently the packet cannot export a host real interval, a selected endpoint value, quotient real equality, or ambient order completeness.

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket real seal non escape`

Proof. Read the seal-facing part of the carrier. The only real rows present are the lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint displays, their finite $\mathsf{StreamName}^{\uparrow}$ windows, the associated $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and the $\mathsf{Real}^{\uparrow}$ rows explicitly recorded in E.

When a seal-facing read is transported, apply Definition 394.3. The same componentwise hsame and Cont data move the endpoint windows, readbacks, rational cells, dyadic refinements, and seal rows as one packet.

It remains to exclude non-displayed real data. The carrier in Definition 394.1 has no coordinate for an ambient interval, no quotient of real names, and no completeness principle; every accepted read must therefore remain inside J and its local NameCert provenance. □

Theorem 394.8 (LocatedInterval real-seal ledger scope)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every real-seal read accepted by the local $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *exposes only the endpoint-window and interval-cell ledger already present in* J*: the* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L *and* U*, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I*, optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement rows* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window ledger* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local row* N*.*

Thus the real-seal ledger scope of a $\mathsf{LocatedInterval}^{\uparrow}$ *packet is the endpoint-window and interval-cell subledger of* J*; it does not contain a selected real endpoint value, a quotient of endpoint names, a host interval, or an external interval endpoint oracle.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket real seal ledger scope`

Proof. Project the packet through Definition 394.1. The only endpoint rows named there are L and U, the only interval-cell rows are I and D, and the only finite observation rows attached to the endpoints are the $\mathsf{StreamName}^{\uparrow}$ windows S, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks R, and the $\mathsf{Real}^{\uparrow}$ seal-facing rows E.

Endpoint movement is already constrained by Definition 394.3. Its componentwise transport keeps the lower endpoint, upper endpoint, windows, readbacks, and seal rows under the displayed hsame witnesses and Cont routes.

Containment and refinement movement is constrained by Definition 394.5 and Definition 394.6. These results keep rational cells and dyadic refinements attached to the displayed interval ledger.

It remains to rule out non-displayed seal data. Since the carrier has no coordinate beyond J and its Pkg and NameCert provenance, every accepted real-seal read is a projection from the listed endpoint-window and interval-cell rows. □

Theorem 394.9 (LocatedInterval containment transport scope)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, $\mathsf{RationalInterval}^{\uparrow}$ *containment exactness is stable under endpoint* hsame *movement, displayed* Cont *routes, and* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements. Rational containment reads and dyadic refinement handoffs remain attached to the same endpoint ledger and the same rational cell ledger of* J*.*

More precisely, every rational containment read is routed to I*, every dyadic refinement read is routed to a present row of* D *together with its rational parent cell in* I*, and endpoint transport carries these routes with* L*,* U*,* S*,* R*,* E*,* H*,* C*,* P*, and* N*.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket containment transport scope`

Proof. First read the rational containment side. By Definition 394.5, a containment read accepted by the local certificate must land in the rational cell ledger I or in a displayed dyadic refinement row of D.

Next read the refinement side. By Definition 394.6, a dyadic row is delivered only with the rational cell in I refined by its displayed route, so it cannot detach from the cell ledger.

Finally apply Definition 394.3. The endpoint hsame witnesses, displayed Cont routes, finite windows, readbacks, seal rows, rational cells, and dyadic refinements move as one finite packet, so containment and refinement remain attached to the same endpoint and cell ledgers after transport. □

Theorem 394.10 (LocatedInterval endpoint-window readback exhaustion)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every endpoint read accepted by the local $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *is exhausted by the two* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L *and* U*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, and the displayed transport, route, package, and provenance rows* H, C, P, N*.*

After endpoint transport, the same endpoint read lands in the transported copies of those rows and in no further endpoint source.

Proof. Begin with the carrier display in Definition 394.1. The only endpoint coordinates of J are the lower row L and the upper row U, and the only finite observations attached to them are the windows S, readbacks R, and seal-facing rows E.

Apply Definition 394.3 to any comparison of accepted packets. It transports L, U, S, R, E together with the displayed hsame rows H, continuation routes C, package row P, and local naming row N.

It remains to rule out an unlisted endpoint source. The carrier has no coordinate for a chosen endpoint value, quotient real equality, host interval, or completeness theorem, so every endpoint read is a projection from the displayed rows of J. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket endpoint window readback exhaustion`

Theorem 394.11 (LocatedInterval endpoint-cell seal factorization)**.** *For every displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

every endpoint consumer and every containment consumer factors through the same ordered route: the $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L, U*, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I*, any attached* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row in* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, and finally the* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows* H, C, P, N*. No host interval, selected endpoint value, quotient real equality row, or ambient completeness row can bypass this chain.*

Proof. Begin with endpoint reads. By Definition 394.10, any accepted endpoint consumer is exhausted by $L, U, S, R, E, H, C, P, N$, so the consumer has no endpoint source outside the displayed packet.

Now read the containment side. The rational cell component is forced through I by Definition 394.5, and a dyadic refinement component can be consumed only through a displayed row of D attached to its rational parent by Definition 394.6.

Endpoint and containment transport then share the same finite route. Apply Definition 394.3 and Definition 394.9 to keep the endpoint rows, cell ledger, refinements, windows, readbacks, seal rows, and BEDC rows moving componentwise through H, C, P, N.

Finally use Definition 394.7. The $\mathsf{Real}^{\uparrow}$ seal boundary admits only the displayed endpoint-window-readback route and contains no coordinate for a host interval, selected endpoint value, quotient real equality row, or ambient completeness principle. Hence every endpoint or containment consumer factors through the ordered rows of J. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket endpoint cell seal factorization`

Theorem 394.12 (LocatedInterval rational–dyadic cell chain)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every containment consumer reads either a $\mathsf{RationalInterval}^{\uparrow}$ *cell in* I *or a* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row in* D *attached to such a rational cell. Under endpoint transport, the rational cell and its displayed dyadic refinements remain attached to the transported endpoint windows and readbacks.*

Proof. Read the rational side first. By Definition 394.5, an accepted containment route lands in the rational ledger I unless it is a routed dyadic refinement.

For the dyadic side, apply Definition 394.6. A row of D can be consumed only together with the rational parent cell in I that its route refines.

Finally transport the endpoints by Definition 394.3. The endpoint rows, windows, readbacks, seal rows, rational cells, and dyadic refinements move through the same displayed H, C, P, N data, so the rational–dyadic chain cannot detach from the transported endpoint ledger. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket rational dyadic cell chain`

Theorem 394.13 (LocatedInterval scoped real-seal scope)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, every $\mathsf{Real}^{\uparrow}$ *seal-facing read accepted by the local* $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *factors through* $L, U, S, R, E, H, C, P, N$*. Such a read cannot introduce a host interval, endpoint choice, quotient real equality, or order-completeness principle.*

Proof. Use Definition 394.7 for the seal boundary. It already routes every seal-facing read through the endpoint rows, finite windows, readbacks, seal rows, transports, continuation routes, package provenance, and local naming row.

The endpoint part of that factorization is exhausted by Definition 394.10; the interval side is constrained by Definition 394.12. Together these two dependencies keep seal reads on the same finite endpoint-window and rational–dyadic cell ledger.

No remaining coordinate of Definition 394.1 can supply host interval data, an endpoint choice functional, quotient real equality, or order completeness. The scoped seal scope is therefore exactly the displayed subledger of J. □

Theorem 394.14 (LocatedInterval kernel scope package)**.** *For a displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, the $\mathsf{LocatedInterval}^{\uparrow}$ *scoped kernel package consists exactly of the following finite interface rows:*

1. *carrier habitation by the displayed tuple* J*;*
2. *endpoint transport through the componentwise* hsame *and* Cont *rows of* H *and* C*;*
3. *containment exactness for the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger* I*;*
4. *dyadic refinement handoff from rows of* D *to their rational parent cells in* I*;*
5. *real-seal non-escape through the endpoint, window, readback, and seal rows* L*,* U*,* S*,* R*, and* E*.*

All five rows are repacked over one scoped BEDC packet over BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket kernel scope package`

Proof. The carrier row is Definition 394.1 read as the finite tuple J. Carrier habitation is then the projection described by Definition 394.4.

The endpoint part of the interface is supplied by Definition 394.3 and sharpened to the displayed seal subledger by Definition 394.8. These two results give exactly the lower and upper endpoint routes visible in J.

The interval part is supplied by Definition 394.5, Definition 394.6, and Definition 394.9. Together they keep rational containment and dyadic refinement reads on the same finite endpoint and cell ledgers.

The seal boundary is Definition 394.7. Repacking these projected rows under the same BHist, Pkg, and NameCert data yields the scoped kernel interface, and no additional row can be introduced because the carrier contains no host interval or ambient completeness coordinate. □

Definition 394.15 (LocatedInterval public certificate interface)**.** *The public certificate interface for* $\mathsf{LocatedInterval}^{\uparrow}$ *is the finite tuple surface*

$$\mathcal{I}_{\mathsf{LocatedInterval}^{\uparrow}} = (L, U, I, D, S, R, E, H, C, P, N)$$

whose coordinates are the lower and upper endpoint rows L, U*, the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I*, the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, and the BEDC rows* H, C, P, N *for displayed* hsame *transports,* Cont *routes, package provenance, and local naming.*

Theorem 394.16 (LocatedInterval public export packet)**.** *For every displayed finite history row*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

accepted by the $\mathsf{LocatedInterval}^{\uparrow}$ *carrier, Definition 394.14 repacks exactly its carrier habitation, endpoint transport, rational containment exactness, dyadic refinement handoff, real-seal non-escape, scoped real-seal scope, and kernel-scope rows as one public* $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *export over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The export contains no coordinate outside the public certificate interface of Definition 394.15.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket public export packet`

Proof. First read the carrier surface from Definition 394.1 and Definition 394.4. This gives the finite tuple J with no hidden coordinate.

The endpoint component is supplied by Definition 394.3; the rational component is supplied by Definition 394.5; and the dyadic component is supplied by Definition 394.6. These are exactly the endpoint, cell, and refinement rows of $\mathcal{I}_{\mathsf{LocatedInterval}^{\uparrow}}$.

The Real-facing component is bounded by Definition 394.7 and scoped by Definition 394.13. Combining these rows through Definition 394.14 yields the named $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ export, and the displayed interface already lists every row used by the repacking. □

Theorem 394.17 (LocatedInterval public consumer route)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*, or* $\mathsf{DyadicInterval}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet factors through the public export packet of Definition 394.16. The consumer route is only the displayed carrier, endpoint transport, containment exactness, dyadic refinement, seal non-escape, and scoped seal ledger over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket public consumer route`

Proof. For endpoint-window and readback consumers, apply Definition 394.10. Their reads pass through the endpoint rows L, U, StreamName windows S, RegSeqRat readbacks R, and the same transport and continuation rows.

For rational and dyadic interval consumers, apply Definition 394.12. Its exactness and handoff dependencies are Definition 394.5 and Definition 394.6, so the consumer sees only the public cell ledger I and refinement ledger D.

For Real-facing consumers, apply Definition 394.13 and Definition 394.16. The seal rows E and BEDC rows H, C, P, N are already part of $\mathcal{I}_{\mathsf{LocatedInterval}^{\uparrow}}$, while the carrier and kernel package expose no coordinate beyond the displayed public route. □

Theorem 394.18 (LocatedInterval real comparison boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison that uses a public* $\mathsf{LocatedInterval}^{\uparrow}$ *packet is bounded by the endpoint windows, readbacks, rational cells, dyadic refinements, seal rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}$ *row named in Definition 394.16. It cannot read a host interval primitive, selected endpoint value, quotient real equality witness, or order-completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket real comparison boundary`

Proof. The comparison first enters through the public consumer route of Definition 394.17. Thus every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ read has already been restricted to $\mathcal{I}_{\mathsf{LocatedInterval}^{\uparrow}}$.

Apply Definition 394.7 for the seal boundary and Definition 394.13 for its scoped form. These results route the comparison through $L, U, S, R, E, H, C, P, N$ and keep the rational and dyadic side attached by Definition 394.12.

Finally use Definition 394.14. The package contains no host interval primitive, selected endpoint value, quotient real equality witness, or order-completeness principle, so the public comparison boundary has exactly the displayed rows and no additional real construction. □

Theorem 394.19 (LocatedInterval endpoint-cell route totality)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

each endpoint request and each containment request routes through the same displayed finite surface: the endpoint-window rows L, U, S, R, E*, the rational and dyadic cell rows* I, D*, and the BEDC rows* H, C, P, N*. No accepted endpoint or containment read uses a coordinate outside this surface.*

Proof. Endpoint-window and endpoint-readback requests are exhausted by Definition 394.10. They therefore read only the lower and upper endpoint rows, the attached $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{Real}^{\uparrow}$ seal rows, and the displayed transport and continuation rows.

Containment requests are exhausted by Definition 394.12. Rational containment reads enter through I, dyadic refinement reads enter through D, and both return to the same endpoint rows under the displayed containment transport.

Finally apply Definition 394.17. It repacks the endpoint and cell routes as one public consumer surface over J, so endpoint, rational containment, and dyadic containment requests are total exactly on the rows $L, U, I, D, S, R, E, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket endpoint cell route totality`

Theorem 394.20 (LocatedInterval stream readback cell compatibility)**.** *In every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks attached to the lower and upper endpoints remain compatible with transported rational and dyadic containment cells. Transport of an endpoint read along H and C preserves the containment-cell route through I and D, and transport of a containment read preserves the endpoint-window and readback rows through L, U, S, R, E.*

Proof. Begin with endpoint transport. By Definition 394.3, endpoint comparisons and readbacks move componentwise through the displayed hsame and Cont rows, so the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows remain attached to the same lower and upper endpoints.

Now use the containment transport scope of Definition 394.9. It states that rational containment rows and dyadic refinements are transported through the same endpoint package rather than through a separate interval carrier.

Combining these two transport statements with Definition 394.12 gives compatibility: endpoint-window readback cannot detach from I or D, and cell containment cannot detach from L, U, S, R, E. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket stream readback cell compatibility`

Theorem 394.21 (LocatedInterval downstream root-unblock package)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet $J = (L, U, I, D, S, R, E, H, C, P, N)$, the finite root package consumed by* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, *and* $\mathsf{DyadicInterval}^{\uparrow}$ *is exactly the public export packet of Definition 394.16, the real comparison boundary of Definition 394.18, the endpoint-cell route totality of Definition 394.19, and the stream-readback cell compatibility of Definition 394.20. This package excludes host intervals, selected endpoints, quotient real equality, and order completeness.*

Proof. The public export packet supplies the finite carrier-facing surface over $L, U, I, D, S, R, E, H, C, P, N$. It already names the endpoint, cell, seal, transport, continuation, provenance, and local naming rows available to downstream consumers.

The endpoint-cell route totality theorem gives a single route for endpoint requests and containment requests. The stream-readback cell compatibility theorem then shows that these routes remain attached under endpoint transport and containment transport.

Finally use the real comparison boundary. It is the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ edge of the same package, and it excludes host intervals, selected endpoints, quotient real equality, and order completeness. Hence the five downstream consumers receive exactly the displayed finite root package. □

Theorem 394.22 (LocatedInterval bridge-facing certificate package)**.** *For every displayed accepted packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

of $\mathsf{LocatedInterval}^{\uparrow}$, *the public export packet, public consumer route, and real comparison boundary assemble one bridge-facing certificate over exactly the rows* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$. *The certificate packages the endpoint rows, rational-cell ledger, dyadic-refinement ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows of J, and it exports no host interval primitive, selected endpoint value, quotient real equality, or order-completeness principle.*

Proof. Use Definition 394.16 for the certificate surface. It gives the public $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ export over the displayed endpoint, rational-cell, dyadic-refinement, seal, transport, continuation, package, and naming rows of J.

Use Definition 394.17 for downstream access. Every $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, and $\mathsf{DyadicInterval}^{\uparrow}$ consumer is forced through the same public rows, so the certificate has a single bridge-facing consumer surface rather than separate endpoint, interval, and real routes.

Use Definition 394.18 for the Real-facing boundary. It fixes the permitted comparison reads to the endpoint windows, readbacks, rational cells, dyadic refinements, seal rows, hsame transports, Cont routes, Pkg provenance, and local NameCert row.

Finally apply Definition 394.14. Its non-escape statement keeps the package inside the concrete rows of J, so the assembled certificate has exactly the displayed BEDC surface and cannot export a host interval primitive, selected endpoint value, quotient real equality, or order-completeness principle. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalUp StdBridge`

Theorem 394.23 (LocatedInterval endpoint order window)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet, each endpoint order read is a finite row over the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, the* $\mathsf{RationalInterval}^{\uparrow}$ *containment ledger, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *coordinates. The read cannot inspect a host interval order, an ambient endpoint selector, quotient real equality, or order completeness outside those rows.*

Proof. Start from the carrier of Definition 394.1. Its endpoint data are the displayed lower and upper rows together with the stream windows and regular rational readbacks attached to those rows.

Apply Definition 394.3 to move endpoint comparisons through the componentwise hsame rows and the displayed Cont routes. Apply Definition 394.10 for the readback boundary, so every order read is witnessed by the finite endpoint-window and regular-readback rows.

The $\mathsf{Real}^{\uparrow}$ boundary is supplied by Definition 394.7. Since the carrier has no coordinate for a host interval order, selected endpoint, quotient real equality, or completeness principle, the endpoint order window is exhausted by the named finite rows. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket endpoint order window`

Theorem 394.24 (LocatedInterval rational cell coverage route)**.** *Every rational or dyadic containment read of an accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet routes through the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger* I *or the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger* D*, and then through the endpoint transport row. Thus containment coverage is read only from* I*,* D*, the lower and upper endpoint rows, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Proof. Inspect a rational containment read. By Definition 394.5, the read is a projection of the $\mathsf{RationalInterval}^{\uparrow}$ cell ledger I attached to the displayed endpoint rows.

Inspect a dyadic containment read. By Definition 394.6, the read first enters the $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger D and then hands back to the same rational-cell and endpoint rows.

Finally apply Definition 394.9. The transport row preserves the endpoint, cell, and refinement coordinates componentwise, so rational and dyadic coverage cannot leave I, D, and the finite BEDC rows that carry them. □

Theorem 394.25 (LocatedInterval dyadic real-seal handoff)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

every dyadic refinement row in D *that contributes to a* $\mathsf{Real}^{\uparrow}$ *seal read must pass through its rational parent cell in* I*, the endpoint-window rows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* E*, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows* H, C, P, N*. The handoff has no coordinate for a host interval, selected endpoint value, quotient real, or ambient order-completeness principle.*

Proof. Project the carrier of Definition 394.1. Its refinement data appear as the displayed row D, and its seal-facing data appear only through the endpoint windows S, readbacks R, seal rows E, and the transport, continuation, provenance, and naming rows H, C, P, N.

Use containment exactness and dyadic refinement handoff. Theorems 394.5 and 394.6 force every dyadic read in D back through the rational parent cell ledger I before it can be consumed as interval evidence.

Now apply real-seal non-escape and rational-cell coverage. Theorems 394.7 and 394.24 keep the Real-facing read attached to the endpoint-window, regular-readback, seal, rational-cell, and dyadic-refinement rows of the same packet.

Finally use the carrier inventory. Since J contains no host interval coordinate, selected endpoint selector, quotient real carrier, or order-completeness field, the only possible handoff from dyadic refinement evidence to a $\mathsf{Real}^{\uparrow}$ seal read is the finite route through I, S, R, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket dyadic real seal handoff`

Theorem 394.26 (LocatedInterval downstream root package)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet, the downstream root package consists exactly of carrier habitation, endpoint order windows, rational-cell coverage routes, dyadic refinement handoff, real-seal non-escape, and containment transport scope over* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert*. Any downstream interval or* $\mathsf{Real}^{\uparrow}$ *consumer must factor through this finite package.*

Proof. Carrier habitation is Definition 394.4. It supplies the finite packet rows and rules out hidden host interval coordinates.

The endpoint part of the root package is Definition 394.23. The rational and dyadic part is Definition 394.24, with dyadic handoff supplied by Definition 394.6.

For Real-facing consumers, use Definition 394.7. For transport of containment reads, use Definition 394.9. Repacking these rows under the public export and consumer route of Definition 394.16 and Definition 394.17 yields the finite downstream package and no additional primitive. □

Definition 394.27 (LocatedInterval root obligation carrier)**.** *The root obligation carrier for* $\mathsf{LocatedInterval}^{\uparrow}$ *is the same finite packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

as in Definition 394.1, read with the following dependency surface:

1. L *and* U *are the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows;*
2. I *is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger and* D *is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to those cells;*
3. S *records the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows and* R *records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used by the windows;*

4. *E records the* $\mathsf{Real}^{\uparrow}$ *seal rows for the two endpoint names;*
5. *H records the componentwise* hsame *transports for endpoints, cells, refinements, windows, readbacks, and seals;*
6. *C records the displayed* Cont *routes used by endpoint, containment, refinement, readback, and seal consumers;*
7. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedInterval}^{\uparrow}}$ *row.*

This carrier obligation is the public root-facing presentation of J. It contains no host interval primitive, no selected endpoint value, no quotient real equality row, and no order-completeness principle.

Theorem 394.28 (LocatedInterval classifier stability)**.** *For accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carriers $J = (L, U, I, D, S, R, E, H, C, P, N)$ and $J' = (L', U', I', D', S', R', E', H', C', P', N')$, any displayed componentwise* hsame *transport comparing the endpoint rows, rational-cell ledgers, dyadic refinements, endpoint windows, readbacks, and seal rows preserves the local classifier. Thus endpoint and cell classification is read only through the rows listed in Definition 394.27.*

Proof. Start from the root obligation carrier of Definition 394.27. Its classifier-visible coordinates are the two endpoint rows, the rational and dyadic cell ledgers, the endpoint windows, the readbacks, and the seal rows, together with their displayed transport and route rows.

Apply Definition 394.3. It moves L, U, I, D, S, R, E through the componentwise hsame rows and the displayed Cont routes as one located interval packet.

The cell side is then constrained by Definition 394.9. Rational containment and dyadic refinement reads remain attached to the transported endpoint and cell ledgers, so the classifier has no independent endpoint or interval source to inspect. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket classifier stability`

Theorem 394.29 (LocatedInterval ledger exactness)**.** *Every rational or dyadic containment read of an accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carrier factors through the visible ledger rows I and D of Definition 394.27. A rational read is a read of a displayed* $\mathsf{RationalInterval}^{\uparrow}$ *cell in I; a dyadic read is a read of a displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row in D together with its rational parent cell. No containment read is supplied by a host interval order, a selected endpoint value, or a quotient-real comparison.*

Proof. For rational containment, apply Definition 394.5. It routes every accepted containment read to the rational cell ledger I unless the read is explicitly a dyadic refinement.

For dyadic containment, apply Definition 394.6. A row of D can be consumed only together with the rational cell in I that its displayed route refines.

Finally apply Definition 394.24 and Definition 394.12. These results keep the rational and dyadic ledgers attached to the endpoint windows and rule out any containment source outside the visible cell and refinement rows. □

Theorem 394.30 (LocatedInterval Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carrier receives only endpoint-window rows L, U, S, readback rows R, seal rows E, and the package rows H, C, P, N of Definition 394.27, with rational and dyadic containment information available only through I and D. The handoff exports*

no host interval primitive, no completed endpoint value, no quotient real equality, and no order-completeness witness.

Proof. Use Definition 394.7 for the Real-facing boundary. It forces every seal-facing read through the endpoint rows, endpoint windows, readbacks, seal rows, transport rows, continuation routes, package provenance, and local naming row.

Use Definition 394.18 for comparison reads. The permitted comparison surface is the same endpoint-window, readback, cell, refinement, seal, transport, route, package, and naming surface.

Finally apply Definition 394.26. It packages the carrier habitation, endpoint transport, rational-cell coverage, dyadic refinement handoff, real-seal non-escape, and containment transport scope for downstream Real and completion consumers. Since the package contains no host interval, endpoint selector, quotient real equality, or order-completeness coordinate, the handoff is exactly the finite BEDC surface displayed above. □

Theorem 394.31 (LocatedInterval endpoint-cell refinement completeness)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carrier*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

every endpoint-cell refinement read factors through $L, U, I, D, S, R, E, H, C, P, N$*. The read may inspect lower and upper located endpoints, rational cells, dyadic refinements, endpoint windows, readbacks, real seal rows, transport rows, continuation routes, package provenance, and local naming; it cannot inspect a host interval primitive, selected endpoint value, quotient real equality, or order-completeness principle.*

Proof. Start with endpoint transport. By Definition 394.3, endpoint comparisons and readbacks move componentwise through the displayed hsame and Cont rows.

Next use Definition 394.5. Rational cell reads are exactly reads of the displayed $\mathsf{RationalInterval}^{\uparrow}$ ledger I attached to the lower and upper endpoint rows.

For refinements, apply Definition 394.6. A dyadic refinement row in D is consumed together with its rational parent cell and the same endpoint transport.

Finally use Definition 394.30. The Real-facing part of the read stays inside S, R, E, H, C, P, N, and the carrier has no coordinate for host intervals, selected endpoint values, quotient real equality, or order completeness. □

Theorem 394.32 (LocatedInterval located-real window coverage)**.** *In every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carrier, the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint windows exhaust all endpoint observations. Any endpoint observation factors through the endpoint rows* L, U*, the window rows* S*, the readback rows* R*, the seal rows* E*, and the package rows* H, C, P, N*, with rational and dyadic cell information available only through* I *and* D*.*

Proof. Use Definition 394.10. It states that endpoint-window and endpoint-readback requests are exhausted by the lower and upper endpoint rows, their stream windows, their regular rational readbacks, and the displayed transport and continuation rows.

Apply Definition 394.3. Transport preserves those lower and upper endpoint observations componentwise, so an observation cannot detach from $L, U, S, R, E, H, C, P, N$.

It remains to account for interval cells. By Definition 394.12, rational cells and dyadic refinements attach to the endpoint windows through I and D rather than by adding a separate endpoint-observation source.

Thus the two located-real endpoint windows cover all endpoint observations, while containment data remains visible only through the finite cell ledgers. □

Theorem 394.33 (LocatedInterval rational-dyadic Real handoff exactness)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *root obligation carrier, rational cells, dyadic refinements, and* $\mathsf{Real}^{\uparrow}$ *seal rows form one exact finite route through*

$$L, U, I, D, S, R, E, H, C, P, N.$$

A downstream $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer receives this route as the public handoff and cannot separate rational containment, dyadic refinement, or real sealing into independent host objects.*

Proof. Begin with Definition 394.12. It ties rational containment cells in I and dyadic refinement rows in D to the same endpoint package.

Use Definition 394.32. The lower and upper endpoint observations are exhausted by the endpoint windows, readbacks, and seal-facing rows.

Now apply Definition 394.30. Every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives the endpoint-window rows, readback rows, seal rows, and package rows of the root obligation carrier, with rational and dyadic information available only through I and D.

Finally apply Definition 394.31. It rules out a second route through host intervals, selected endpoints, quotient real equality, or order completeness, so the rational-dyadic-real handoff is exactly the displayed finite route. □

Theorem 394.34 (LocatedInterval window-cell seal unification)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

every downstream $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer that reads an endpoint window, rational or dyadic containment cell, dyadic refinement, and* $\mathsf{Real}^{\uparrow}$ *seal factors through the single route*

$$L, U \to S \to R \to I, D \to E \to H, C, P, N.$$

No such consumer may replace this route by a host interval, selected endpoint, quotient-real equality, order-completeness principle, or ambient completion principle.

Proof. Start with endpoint windows. By Definition 394.32, the lower and upper endpoint observations are exhausted by $L, U, S, R, E, H, C, P, N$, while rational and dyadic cell information remains visible only through I and D.

Next use Definition 394.24. It identifies rational containment reads with the finite cell route attached to the accepted packet, so a consumer cannot detach containment from the endpoint-window rows.

For the seal-facing part, apply Definition 394.25. Dyadic refinement rows and $\mathsf{Real}^{\uparrow}$ seal rows are handed to downstream Real-facing readers as one packet-local surface, with no ambient completion coordinate.

Finally use Definition 394.33. It composes rational containment, dyadic refinement, and $\mathsf{Real}^{\uparrow}$ sealing into the displayed finite route. The sibling public surfaces in chapter 126, chapter 301, chapter 306, and chapter 38 supply the endpoint-window, readback, refinement, and Real-seal consumers, so none of them can introduce a second route through host intervals, selected endpoints, quotient-real equality, order completeness, or ambient completion. □

Theorem 394.35 (LocatedInterval StreamName Real cell synchronization)**.** *Let*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

and

$$J' = (L', U', I', D', S', R', E', H', C', P', N')$$

be accepted $\mathsf{LocatedInterval}^{\uparrow}$ *packets. If the two packets agree on the lower and upper located-real endpoint rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the rational and dyadic cell ledgers, the* $\mathsf{Real}^{\uparrow}$ *seal rows, and the displayed* hsame, Cont, Pkg, *and* NameCert *rows, then every* $\mathsf{Real}^{\uparrow}$ *or completion-facing endpoint-cell consumer reads the same finite route from* J *and* J'.

In particular, such a consumer cannot distinguish the packets by host interval order, selected endpoints, quotient real equality, or order completeness. Its read factors through the synchronized route

$$L, U \to S \to R \to I, D \to E \to H, C, P, N$$

and the corresponding primed route.

Proof. Start with classifier stability for the accepted $\mathsf{LocatedInterval}^{\uparrow}$ packets. Agreement on $L, U, S, R, I, D, E, H, C, P, N$ places both packets on the same displayed carrier rows before any endpoint-cell consumer reads them.

Use Definition 394.33. The rational containment cells, dyadic refinements, and $\mathsf{Real}^{\uparrow}$ seal rows form one exact finite route for each packet.

Apply Definition 394.34. It identifies the endpoint window, readback, cell, refinement, and seal rows as a single consumer route and excludes replacement by host intervals, selected endpoint values, quotient real equality, order completeness, or ambient completion.

Finally compare the two displayed routes componentwise through the shared hsame, Cont, Pkg, and NameCert rows. Since every consumer-facing coordinate agrees, the two packets are synchronized for every $\mathsf{Real}^{\uparrow}$ or completion-facing endpoint-cell read. □

Theorem 394.36 (LocatedInterval bridged public route)**.** *The public* $\mathsf{LocatedInterval}^{\uparrow}$ *packet over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *is the bridge-facing route consumed by downstream Real and completion interfaces. Every such consumer receives the lower endpoint, upper endpoint, rational cell, dyadic refinement, endpoint-window, readback, seal, transport, continuation, package, and naming rows as one finite public certificate.*

Proof. Use Definition 394.16 and Definition 394.17. They identify the public exported surface and the consumer route for the accepted packet.

Apply Definition 394.18. It keeps comparison reads inside the endpoint-window, readback, rational-cell, dyadic-refinement, seal, transport, route, package, and naming rows.

Next use Definition 394.33. The rational containment cells, dyadic refinements, and $\mathsf{Real}^{\uparrow}$ seal rows form one exact finite route for downstream Real and completion consumers.

The route is bridge-facing because it exposes the same public certificate rows to the outside interface while preserving the BEDC non-escape boundary. No host interval primitive, selected endpoint value, quotient real equality, or order-completeness row is exported. □

Theorem 394.37 (LocatedInterval real-stream bracket correspondence)**.** *For every accepted* $\mathsf{LocatedInterval}^{\uparrow}$ *packet*

$$J = (L, U, I, D, S, R, E, H, C, P, N),$$

the following two finite BEDC reads are equivalent as reads of the same packet-local certificate.

1. *The* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-cell read passes through the lower and upper endpoint rows, rational containment cells, dyadic refinement cells, endpoint-window rows, readback rows, real-seal rows, transports, continuations, package provenance, and local* NameCert *row of* J.

2. *The* $\mathsf{Real}^{\uparrow}$ *stream-classifier read receives its finite endpoint windows from* $\mathsf{StreamName}^{\uparrow}$*, its pointwise rational stream readbacks from* $\mathsf{RegSeqRat}^{\uparrow}$*, and its endpoint seal rows from the same* $\mathsf{Real}^{\uparrow}$ *rows carried by* J*.*

Thus rational and dyadic bracket containment for J *and the Real-facing stream-name substrate read determine one another only through the displayed finite rows*

$$L, U, I, D, S, R, E, H, C, P, N.$$

The correspondence exports no host interval primitive, selected endpoint value, quotient-real equality, ambient order-completeness principle, or unrecorded completion object.

Lean4 ✓ `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket real stream bracket correspondence`

Proof. First read the interval side through Definition 394.17. It makes every public consumer of J pass through the displayed endpoint, rational cell, dyadic refinement, readback, seal, transport, continuation, package, and naming rows.

Next apply Definition 394.33. This identifies rational containment, dyadic refinement, and $\mathsf{Real}^{\uparrow}$ sealing as one packet-local finite route rather than as three independent host structures.

For the stream-facing side, use the StreamName-RegSeqRat pointwise handoff together with the RegSeqRat real-seal handoff. The finite endpoint windows supplied by $\mathsf{StreamName}^{\uparrow}$ and the pointwise rational readbacks supplied by $\mathsf{RegSeqRat}^{\uparrow}$ are therefore the same windows and readbacks already present in S and R, and the Real rows seal them through E.

Finally apply the RealStreamClassifier selected endpoint readback at the $\mathsf{Real}^{\uparrow}$ endpoint surface. Since the classifier reads only the finite windows and their Real seal rows, the two displayed reads are inverse descriptions of the same certificate route. No step introduces host intervals, selected endpoints, quotient-real equality, order completeness, or ambient completion. □

Theorem 394.38 (LocatedInterval dyadic refinement induction)**.** *Let*

$$J = (L, U, I, D, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{LocatedInterval}^{\uparrow}$ *packet. For any finite displayed list of dyadic refinement rows in* D*, if the initial row is attached to its rational parent cell in* I *and to the endpoint-window rows* L, U, S, R, E*, and if each successor row is obtained by the displayed dyadic refinement handoff from the preceding row, then every row in the list remains attached to the same rational parent cell and endpoint-window package. The induction surface is only the finite rows* $D, I, L, U, S, R, E, H, C, P, N$*; it does not use a host interval subdivision tree, selected endpoints, quotient-real equality, order completeness, or ambient completion.*

Proof. The base case is the displayed attachment supplied by Definition 394.12. A dyadic row in D is read together with its rational parent cell in I, and the lower and upper endpoint windows are already the rows L, U, S, R, E of the same packet.

For the successor step, apply Definition 394.6. The handoff replaces the current dyadic cell by the next displayed refinement row while retaining the rational parent cell and the same endpoint-window rows.

Endpoint transport is controlled by Definition 394.3. Transport through H and replay through C preserve the lower endpoint, upper endpoint, stream window, regular readback, and Real-seal rows componentwise, so the refinement step cannot detach from L, U, S, R, E.

Finally use Definition 394.37. It identifies the interval-side dyadic bracket read and the Real-facing stream read as the same packet-local certificate route. Finite induction over the displayed rows of D therefore preserves attachment to I and to L, U, S, R, E at every stage. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedInterval}^\uparrow}$.

Constructive story: A $\mathsf{LocatedInterval}^\uparrow$ packet is generated from lower and upper $\mathsf{LocatedReal}^\uparrow$ endpoint rows, a $\mathsf{RationalInterval}^\uparrow$ cell ledger, optional $\mathsf{DyadicInterval}^\uparrow$ refinements, $\mathsf{StreamName}^\uparrow$ endpoint windows, $\mathsf{RegSeqRat}^\uparrow$ readbacks, $\mathsf{Real}^\uparrow$ seal rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedInterval}^\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 394.16, Definition 394.17, and Definition 394.18 close the public packet interface for finite located interval packets over $\mathsf{LocatedReal}^\uparrow$, $\mathsf{RationalInterval}^\uparrow$, $\mathsf{DyadicInterval}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedInterval}^\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.LocatedIntervalUp.LocatedIntervalPacket\_semantic\_name\_certificate`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close host interval primitives, selected endpoint values, quotient real equality, ambient order completeness, or bridge checking.

Upgrade path: No further upgrade remains on the formal bridge axis.

The $\mathsf{PicardContraction}^\uparrow$ packet consumes the metric sibling lattice through chapter 90, chapter 410, and chapter 128 before its $\mathsf{Real}^\uparrow$ seal route is read. Its finite modulus row is therefore a sibling dependency over Banach, Lipschitz-map, and complete-metric data rather than a kernel-only island.

Chapter 395

A Concrete Naming Certificate for $\mathsf{PicardContraction}^{\uparrow}$

The $\mathsf{PicardContraction}^{\uparrow}$ interface names the finite contraction-iteration packet used by $\mathsf{Ode}^{\uparrow}$ and $\mathsf{NewtonIteration}^{\uparrow}$ consumers when they read a displayed Picard step, a Lipschitz ledger, or an explicit Cauchy modulus. In standard functional analysis this surface is motivated by the Banach contraction principle as presented, for example, in Kreyszig, *Introductory Functional Analysis with Applications*. BEDC records only the finite rows consumed by downstream certificates: a Banach source row, a contraction-map row, a rational Lipschitz-ratio ledger, finite iterate rows, an explicit modulus window, transport routes, provenance, and a local $\mathsf{NameCert}$ row. It does not add a host fixed point, ambient completeness theorem, or selected limit.

chapter 90 is the substantive source for the Banach row read by this packet, and chapter 410 is the substantive source for the contraction-map and modulus ledger rows. The Picard carrier therefore exports a finite route between those sibling chapters and the $\mathsf{Ode}^{\uparrow}$ or $\mathsf{NewtonIteration}^{\uparrow}$ consumers that read only the displayed endpoint and seal rows. chapter 396 is the sibling carrier whose contraction-map, complete-metric, and fixed-point rows are structurally consumed by this Picard packet.

Definition 395.1 (PicardContraction BHist carrier)**.** *A* $\mathsf{PicardContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (B, T, L, I, M, E, H, C, G, N)$$

with the following visible rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row whose carried histories provide the state endpoints inspected by the finite iteration packet;*
2. *T is the contraction-map row, displaying the map-code history and the source-target* Cont *route through which a carried state history is read as its next Picard state;*
3. *L is the Lipschitz-ratio ledger, a positive* $\mathsf{Rat}^{\uparrow}$ *row below the unit bound together with the displayed metric comparison rows used by the contraction read;*
4. *I is the finite Picard-iterate ledger, listing the visible state rows $x_0, \ldots, x_n$ and the application rows reading x_{k+1} from T and x_k;*
5. *M is the Cauchy-modulus row, giving an explicit finite window after which the displayed iterate distances are bounded by the requested rational tolerance;*

6. *E is the endpoint and seal row through which downstream* $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumers read the last finite iterate and the modulus window;*
7. *H stores componentwise* hsame *transports for the Banach source, map-code row, Lipschitz ledger, finite iterate ledger, modulus row, endpoint row, provenance row, and local naming row;*
8. *C stores the displayed* Cont *routes for Banach-state reads, contraction-map application, finite iterate concatenation, modulus readback, and endpoint sealing;*
9. *G is the* Pkg *provenance row naming the consumed* $\mathsf{Banach}^{\uparrow}$, *RatUp, metric, iterate, modulus, and downstream certificate surfaces;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *row.*

The carrier is a finite packet. It has no coordinate for a completed orbit, a host fixed point, an ambient completeness theorem, an external contraction principle, or a choice-selected limit.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionCarrier` finite packet semantic certificate

Definition 395.2 (PicardContraction finite classifier)**.** *Two accepted* $\mathsf{PicardContraction}^{\uparrow}$ *carriers are classified as the same finite contraction packet exactly when the visible rows of Definition 395.1 compare componentwise: their* $\mathsf{Banach}^{\uparrow}$ *source rows are classified by the Banach carrier classifier, their contraction-map rows are transported through the displayed* Cont *route, their Lipschitz-ratio ledgers compare through the* $\mathsf{Rat}^{\uparrow}$ *order rows, their finite iterate ledgers are related by* hsame *at every retained endpoint, their Cauchy-modulus windows and endpoint seals are the same after the displayed transport rows, and their* Pkg *and local* NameCert *rows name the same visible packet.*

The classifier has no generator for comparing host functions, no generator for a fixed-point value, and no generator for an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.PicardContractionUp.PicardContractionClassifier`

Theorem 395.3 (PicardContraction finite iterate closure)**.** *Let P and Q be accepted* $\mathsf{PicardContraction}^{\uparrow}$ *carriers over the same* $\mathsf{Banach}^{\uparrow}$ *source row, contraction-map row, and Lipschitz-ratio ledger. If the endpoint row of P is related by* hsame *to the initial iterate row of Q, and the displayed iterate ledgers concatenate by a finite continuation row, then the concatenated packet is again an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *carrier. Its Lipschitz-ratio ledger is the transported ledger shared by P and Q, its finite Picard-iterate ledger is the concatenated* BHist *row, and its Cauchy-modulus and endpoint rows are read through the displayed* Cont *routes.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket` finite iterate closure

Proof. Project the carrier rows of P and Q. Both packets expose the same Banach source, contraction-map row, and Lipschitz-ratio ledger, while their iterate coordinates are finite BHist lists.

Use the endpoint comparison to splice the last visible state of P to the first visible state of Q. The componentwise hsame rows transport only the displayed endpoint histories and do not introduce an external orbit or limit.

Concatenate the two finite iterate ledgers by the given Cont row. The application rows of the first segment and the transported application rows of the second segment remain contraction-map reads under the same T row, and the Lipschitz-ratio ledger is reused through the transported L coordinate.

The modulus and endpoint coordinates of the concatenated packet are then read through the same displayed continuation routes. Repacking the Banach source, contraction row, ratio ledger,

concatenated iterate ledger, modulus row, endpoint seal, transports, provenance, and local name gives exactly the carrier of Definition 395.1. □

Theorem 395.4 (PicardContraction Banach ODE boundary)**.** *Every* $\mathsf{Ode}^{\uparrow}$ *or* $\mathsf{NewtonIteration}^{\uparrow}$ *consumer read of a* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the finite Picard rows, the* $\mathsf{Banach}^{\uparrow}$ *source row, the contraction-map ledger, the Lipschitz-ratio ledger, the explicit Cauchy-modulus row, and the displayed endpoint/seal routes of Definition 395.1. The consumer may not read a host fixed point, a completed Picard orbit, an ambient completeness theorem, a Banach fixed-point theorem, or a choice-selected limit from this packet.*

Proof. Start with an $\mathsf{Ode}^{\uparrow}$ or $\mathsf{NewtonIteration}^{\uparrow}$ read. The only Banach-facing coordinate is B, and the only iteration-facing coordinates are the contraction-map row T, the Lipschitz ledger L, the finite iterate ledger I, and the endpoint/modulus rows M and E.

The classifier of Definition 395.2 can transport only those rows. Banach-state movement is read through the Banach classifier, map application through the displayed Cont route, rational contraction bounds through L, and finite iterate comparison through the endpoint hsame rows.

No remaining carrier coordinate contains a fixed-point value or a completed limit. Therefore downstream ODE and Newton consumers receive exactly the finite contraction surface named in the statement, with full fixed-point convergence and ambient completeness outside the BEDC packet. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket banach ode boundary`

Theorem 395.5 (PicardContraction Cauchy modulus sibling route)**.** *Let* $P = (B, T, L, I, M, E, H, C, G, N)$ *be an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet. Before any* $\mathsf{Ode}^{\uparrow}$ *or* $\mathsf{NewtonIteration}^{\uparrow}$ *consumer can read the endpoint and seal row* E*, the packet reads the* $\mathsf{Banach}^{\uparrow}$ *source row* B *and the* $\mathsf{LipschitzMap}^{\uparrow}$ *contraction-map and modulus ledger rows* T, L, M *through the finite classifier and displayed continuation surface. Thus the consumer-facing endpoint is downstream of concrete sibling routes rather than a kernel-only island.*

Proof. Project the carrier in Definition 395.1. The Banach source row B, contraction-map row T, Lipschitz-ratio ledger L, finite iterate ledger I, Cauchy-modulus row M, and endpoint seal E are distinct visible rows of the accepted packet.

Use the classifier of Definition 395.2. It permits transport only componentwise through the displayed hsame and Cont rows, so the Banach and contraction-map data must be read before the endpoint seal can be compared or exported.

Finally apply Definition 395.4. Every $\mathsf{Ode}^{\uparrow}$ or $\mathsf{NewtonIteration}^{\uparrow}$ read factors through the Banach source, contraction-map ledger, Lipschitz-ratio ledger, explicit Cauchy modulus, and displayed endpoint route. This is exactly the finite sibling route from Banach and LipschitzMap material to the downstream endpoint consumer. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket cauchy modulus sibling route`

Theorem 395.6 (PicardContraction NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier inhabitation for the finite packet of Definition 395.1;*
2. *classifier transport for the componentwise comparison of Definition 395.2;*
3. *finite-iterate stability under the concatenation closure of Definition 395.3;*
4. *modulus and readback exactness for the Cauchy-modulus and endpoint/seal rows;*

5. *ledger non-escape, requiring every* $\mathsf{Ode}^{\uparrow}$ *and* $\mathsf{NewtonIteration}^{\uparrow}$ *consumer to factor through the displayed Banach, contraction, ratio, iterate, modulus, endpoint, transport, provenance, and local-name rows.*

Proof. The carrier row gives the finite inventory. It lists the Banach source, contraction map, Lipschitz ledger, iterate ledger, modulus row, endpoint seal, transport rows, continuation routes, package provenance, and local name.

The classifier row gives the only permitted transports, so every accepted replacement is componentwise and visible. The finite-iterate stability row is exactly Definition 395.3.

The modulus and readback row is the pair (M, E) together with its displayed continuation routes. Finally, Definition 395.4 supplies ledger non-escape for the ODE and Newton consumers. These five rows constitute the local naming-certificate surface and leave fixed-point convergence outside the seed packet. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket namecert obligation surface`

Theorem 395.7 (PicardContraction classifier stability)**.** *Finite* $\mathsf{PicardContraction}^{\uparrow}$ *packets over* BHist *preserve the classifier of Definition 395.2 under every* hsame*-compatible* Cont *row whose stage indices are read through the unary history ledger of Definition 395.1. In particular, transporting the* $\mathsf{Banach}^{\uparrow}$ *source, contraction-map row, Lipschitz ledger, finite Picard-iterate ledger, modulus row, endpoint seal,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *row componentwise produces another packet classified as the same finite contraction packet.*

Proof. Unpack the carrier rows. The only indexed iteration coordinate is the finite Picard ledger I, and its stage reads are carried by the unary history entries named in the packet.

Unpack the classifier. It compares accepted packets coordinate by coordinate through the displayed hsame transports and Cont routes, so an hsame-compatible continuation row can move a stage only by moving the visible unary history row that names it.

The Banach source, contraction row, rational ratio ledger, modulus row, endpoint seal, provenance, and local certificate are ordinary carrier coordinates. Transporting them componentwise therefore preserves exactly the classifier described in Definition 395.2, with no coordinate for a completed orbit or host fixed point. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket classifier stability`

Theorem 395.8 (PicardContraction contraction-step package)**.** *One finite contraction step in a* $\mathsf{PicardContraction}^{\uparrow}$ *carrier is packaged as a concrete* BHist *and* Cont *witness: it consists of the source stage row* x_k*, the displayed contraction-map row* T*, the application row reading* x_{k+1} *from* T *and* x_k*, the Lipschitz-ratio ledger row, and the endpoint/modulus readback route. This package is the finite step consumed by both Definition 395.3 and Definition 395.4.*

Proof. Project the finite Picard ledger from Definition 395.1. A step is not a host function application; it is the displayed pair of consecutive history rows together with the Cont route that reads the next row from the contraction-map coordinate.

Use Definition 395.3 to see that such steps concatenate inside the same BHist packet when the endpoint row of one finite segment is hsame to the initial row of the next segment.

Use Definition 395.4 for the consumer side. ODE and Newton reads receive this same finite step package through the Banach source, contraction map, Lipschitz ledger, modulus row, and endpoint seal, not through a Banach fixed-point theorem or a selected limit. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket contraction step package`

Theorem 395.9 (PicardContraction fixed-point ledger boundary)**.** *The finite contraction orbit ledger of a* $\mathsf{PicardContraction}^{\uparrow}$ *packet hides only the scheduled iterate index, the modulus row, and the endpoint witness selected inside the Picard packet. No Banach source outside the displayed packet, no external contraction principle, no ambient completeness theorem, no completed orbit, and no host fixed point enters the* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *surface.*

Proof. Start from the NameCert obligation surface. Its rows are carrier inhabitation, classifier transport, finite-iterate stability, modulus and endpoint readback exactness, and ledger non-escape.

The hidden data named in the statement are internal scheduling data for the finite packet: which retained iterate is read, which modulus row is used, and which endpoint witness is exported. Each is already a coordinate of Definition 395.1.

Apply Definition 395.7 to transport these rows without changing the accepted packet. Apply Definition 395.8 and Definition 395.4 to force every consumer read through the finite step package. Since the carrier has no coordinate for an external source or completed fixed point, the ledger cannot expose one through the local naming certificate. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket fixed point ledger boundary`

Theorem 395.10 (PicardContraction modulus-window transport)**.** *Let* $P = (B, T, L, I, M, E, H, C, G, N)$ *be an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *carrier. Any displayed* hsame *and* Cont *transport that preserves the rows* B*,* T*,* L*,* I*,* M*,* E*,* H*,* C*,* G*, and* N *of Definition 395.1 also preserves the modulus-window readback row* M *and the endpoint seal* E *as the exact rows consumed by the packet. The transported packet has the same finite classifier of Definition 395.2, the same finite-iterate closure surface of Definition 395.3, and the same ODE/Newton boundary of Definition 395.4.*

In particular, modulus-window readback is a BHist*-anchored transport claim about displayed finite rows. It does not introduce a host fixed point, a completed orbit, an ambient completeness theorem, or a choice-selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket modulus window transport`

Proof. Project the carrier named in Definition 395.1. The modulus coordinate M and endpoint seal E are explicit rows of the packet, so a transport that preserves all ten carrier rows already preserves the only locations from which a modulus window or endpoint read can be taken.

Use the classifier of Definition 395.2. It compares accepted packets componentwise through displayed hsame rows and Cont routes; hence M and E cannot be replaced by a host limit, a completed orbit, or an external convergence witness.

The finite-iterate closure theorem identifies how transported finite segments concatenate, and the Banach ODE boundary theorem identifies the consumer-facing route. Together, Definition 395.3 and Definition 395.4 force every modulus and endpoint read to remain on the displayed M and E rows. □

Theorem 395.11 (PicardContraction Banach source readback)**.** *Every Banach source read consumed by an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the rows*

$$B, T, I, H, C, G, N$$

of Definition 395.1. The read exposes the displayed $\mathsf{Banach}^{\uparrow}$ *source row, contraction-map row, finite iterate ledger, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and local naming row. It does not expose a host Banach fixed-point theorem, a completed Picard orbit, an ambient completeness proof, or a selected fixed point.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket banach source readback`

Proof. Project the accepted carrier $P = (B, T, L, I, M, E, H, C, G, N)$. The Banach source row B is a visible coordinate, and the only rows that can read it for a Picard step are the contraction row T, finite iterate ledger I, transport rows H, continuation routes C, provenance G, and local certificate N.

Apply Definition 395.8. A source read is consumed as part of a finite step package, so it passes through the displayed contraction application and the finite iterate ledger rather than through a completed orbit.

Finally use Definition 395.9. The carrier has no coordinate for a host fixed point or a Banach fixed-point theorem, and the local NameCert row can certify only the displayed finite source route. □

Theorem 395.12 (PicardContraction Lipschitz ledger window exactness)**.** *Every contraction-ratio read of an accepted* PicardContraction$^{\uparrow}$ *packet uses the displayed rational row L, the finite iterate ledger I, and the continuation routes C of Definition 395.1. The read may transport along the componentwise* hsame *rows and package provenance, but it has no route through an external Lipschitz oracle, an ambient norm estimate, or a host contraction principle outside the packet.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket lipschitz ledger window exactness`

Proof. Start from the ratio coordinate of the carrier. The row L is the only rational Lipschitz ledger stored in the packet, and the finite iterate ledger I is the only location where the ratio can be consumed by a displayed Picard step.

Use Definition 395.17. It keeps the contraction ratio as the visible rational ledger row and prevents a consumer from strengthening the read to an external Lipschitz estimate.

The continuation routes are fixed by Definition 395.8 and transported by Definition 395.7. Thus every ratio-window read remains on L, I, C inside the same finite carrier surface. □

Theorem 395.13 (PicardContraction modulus consumer exhaustion)**.** *Every* Ode$^{\uparrow}$, NewtonIteration$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *consumer of an accepted* PicardContraction$^{\uparrow}$ *packet factors through the finite carrier rows*

$$B, T, L, I, M, E, H, C, G, N.$$

The factorization uses the finite iterate closure, Banach source readback, Banach ODE boundary, contraction-step package, fixed-point ledger boundary, modulus-window transport, and Lipschitz ledger window exactness. No consumer receives a completed Picard orbit, host fixed point, external Lipschitz oracle, ambient completeness theorem, or selected limit.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket modulus consumer exhaustion`

Proof. First assemble the finite source and step surface. The finite iterate closure keeps iterate concatenation inside I, while Definition 395.11 and Definition 395.8 route source and step reads through B, T, I, H, C, G, N.

Next attach the ratio and modulus windows. Definition 395.12 fixes contraction-ratio reads on L, I, C, and Definition 395.10 fixes tolerance and endpoint reads on M and E under the displayed transport rows.

Finally inspect the consumer classes. The Banach ODE boundary supplies the Ode$^{\uparrow}$ and NewtonIteration$^{\uparrow}$ restriction, the fixed-point ledger boundary prevents replacement by a host fixed point, and the existing RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ handoff routes read only the same finite packet. The displayed rows therefore exhaust every listed consumer. □

Theorem 395.14 (PicardContraction downstream modulus exposure)**.** *Every downstream tolerance or endpoint request out of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet reaches the displayed* $\mathsf{Banach}^{\uparrow}$ *source row B, contraction row T, rational Lipschitz ledger L, finite iterate ledger I, modulus row M, endpoint seal E, componentwise* hsame *rows H,* Cont *routes C,* Pkg *provenance G, and local* NameCert *row N of Definition 395.1. The request has no route through a completed orbit, host fixed point, ambient completeness theorem, or choice-selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket downstream modulus exposure`

Proof. Project the carrier rows of Definition 395.1. The Banach source, contraction row, rational Lipschitz ledger, finite iterate ledger, modulus row, endpoint seal, transports, routes, provenance, and local certificate are the only visible coordinates.

Apply Definition 395.10 to the modulus and endpoint rows. It keeps every tolerance or endpoint read on M and E, transported only by the displayed H and C rows.

Use Definition 395.8 and Definition 395.9 for the finite-iterate side. These rows bind the request to the displayed Banach and contraction data without adding a completed Picard orbit or a host fixed point. □

Theorem 395.15 (PicardContraction Real-RegSeqRat consumer handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{PicardContraction}^{\uparrow}$ *packet reads only the finite Picard modulus-window packet*

$$B, T, L, I, M, E, H, C, G, N.$$

The handoff exposes the displayed Banach source, contraction row, rational Lipschitz ledger, finite iterate ledger, modulus row, endpoint seal, transport rows, continuation routes, provenance, and local naming row, but it does not expose a completed orbit, host fixed point, Banach fixed-point theorem, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRealRegSeqRatConsumerHandoff`

Proof. Start with downstream modulus exposure. It supplies the only modulus and endpoint route available to consumers and names the exact carrier rows through which such reads pass.

For $\mathsf{RegSeqRat}^{\uparrow}$ consumers, use the modulus row M and finite iterate ledger I as the regular-rational readback surface. The rational Lipschitz ledger L and the displayed continuation routes C record the finite transport needed by that read.

For $\mathsf{Real}^{\uparrow}$ consumers, use the endpoint seal E with the same M and I rows. The fixed-point ledger boundary prevents the seal from becoming a host fixed point, while the Banach ODE boundary prevents the route from becoming an ambient completeness theorem.

Repack the two consumer routes as one finite modulus-window handoff. Since every row is one of $B, T, L, I, M, E, H, C, G, N$, no completed orbit or host fixed point is exported. □

Theorem 395.16 (PicardContraction root-unblock package)**.** *The downstream modulus exposure of Definition 395.14, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer handoff of Definition 395.15, the finite-iterate closure of Definition 395.3, the Banach ODE boundary of Definition 395.4, the fixed-point ledger boundary of Definition 395.9, and the modulus-window transport of Definition 395.10 assemble the downstream root-unblock package for* $\mathsf{PicardContraction}^{\uparrow}$*.*

The package is exactly the finite surface over $\mathsf{Banach}^{\uparrow}$*, contraction, rational Lipschitz, finite iterate, modulus, endpoint seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not close a Banach fixed-point theorem, a completed orbit, a selected host limit, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootUnblockPackage`

Proof. First collect the finite-iterate side. The carrier and finite-iterate closure describe how displayed Picard ledger segments are read and concatenated inside the same BHist packet.

Next collect the consumer side. The Banach ODE boundary and fixed-point ledger boundary keep ODE, Newton, endpoint, and seal consumers downstream of the displayed Banach, contraction, Lipschitz, iterate, modulus, and endpoint rows.

Finally add the two root-facing rows. Downstream modulus exposure names the exact route for tolerance and endpoint requests, and the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ handoff packages those routes for regular-rational and real-name consumers. Together with modulus-window transport, these rows form the announced root-unblock package and leave no coordinate for the excluded host objects. □

Theorem 395.17 (PicardContraction ratio ledger nonexpansion)**.** *Let P and Q be accepted finite* $\mathsf{PicardContraction}^{\uparrow}$ *packets whose endpoint and initial rows splice as in Definition 395.3. If the two packets use the same positive rational Lipschitz ledger row L below the unit bound, then their concatenation reuses the transported ledger L and creates no stronger host contraction claim. The resulting* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *surface is still the componentwise* BHist *packet classified by Definition 395.2 and bounded at consumers by Definition 395.4.*

The conclusion excludes a new fixed point, completed Picard orbit, ambient completeness proof, Banach fixed-point theorem, and choice-selected limit.

Proof. Start with the carrier of Definition 395.1. The Lipschitz ratio is stored as the finite rational row L, and the iterate data are stored separately in the finite ledger I.

Apply Definition 395.3 to concatenate the two iterate ledgers. Its conclusion explicitly reuses the transported shared L coordinate, so the concatenated packet has no additional row from which a tighter or host-level contraction estimate could be read.

The classifier of Definition 395.2 then transports the concatenated packet componentwise. Finally, Definition 395.4 restricts ODE and Newton consumers to the displayed Banach, contraction, ratio, iterate, modulus, endpoint, provenance, and local-name rows. No omitted coordinate can assert a stronger contraction principle or a selected limiting point. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket ratio ledger nonexpansion`

Theorem 395.18 (PicardContraction finite consumer exhaustion)**.** *Every* $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer read of a* $\mathsf{PicardContraction}^{\uparrow}$ *packet is exhausted by the rows*

$$B, T, L, I, M, E, H, C, G, N$$

of Definition 395.1. The classifier of Definition 395.2, the finite concatenation theorem Definition 395.3, and the boundary theorem Definition 395.4 leave no consumer route through fixed points, completed orbits, ambient completeness, or choice-selected limits.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket finite consumer exhaustion`

Proof. For ODE and Newton reads, apply Definition 395.4. It already factors the consumer through the Banach source, contraction map, rational ledger, finite iterate rows, explicit modulus row, and endpoint/seal routes.

For $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads, use the same modulus and endpoint coordinates. The rows M and E are displayed inside the carrier, while H, C, G, and N provide only transport, continuation, provenance, and local naming evidence for those rows.

The finite-iterate closure theorem supplies the only operation that extends visible iterate data, and the classifier theorem supplies the only comparison operation. Since both operate on the carrier

rows listed in the statement, every listed consumer is exhausted by the same finite BHist packet and receives no host fixed point, completed orbit, completeness theorem, or selected limit. □

Theorem 395.19 (PicardContraction modulus row finite-tail lock)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the modulus row M is locked to the finite iterate ledger I, the rational Lipschitz row L, the displayed hsame *rows H, and the displayed* Cont *routes C. Any consumer that reads M must therefore read the same finite* BHist *packet and has no route to a completed Picard orbit, a host fixed point, ambient completeness, or a choice-selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket modulus row finite tail lock`

Proof. Project the carrier of Definition 395.1. The rows L, I, M, H, and C are displayed coordinates of the same packet, so the modulus row is not an external convergence witness.

Use the classifier stability and finite-iterate closure rows already recorded in this chapter. Classifier stability keeps M attached to the componentwise transport rows H and read routes C, while finite-iterate closure keeps every extension of visible Picard data inside I with the same rational ledger L.

Finally apply Definition 395.10. It identifies the modulus-window readback as the finite row M and transports it only through the displayed H and C coordinates, so downstream modulus consumers remain locked to the finite tail packet named in the statement. □

Theorem 395.20 (PicardContraction RegSeqRat RealUp seal package)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through*

$$B, T, L, I, M, E, H, C, G, N.$$

The consumer may use the downstream modulus exposure, the real–regular-rational handoff, and finite consumer exhaustion, but it cannot obtain a completed orbit, a host fixed point, a selected limit, or ambient completeness from the packet.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket regseqrat real seal package`

Proof. Begin with Definition 395.14. It gives seal-facing consumers the modulus and endpoint route only through M, E, H, and C, together with the Banach, contraction, rational, iterate, provenance, and naming rows.

Next use Definition 395.15. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal read the same finite modulus-window packet rather than a completed orbit or a host fixed-point theorem.

Finally apply Definition 395.18. Its consumer list includes both $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, so the seal package is exhausted by the carrier rows and cannot pass through any omitted completion, fixed-point, limit-selection, or ambient-completeness object. □

Theorem 395.21 (PicardContraction finite-tail diagonal seal handoff)**.** *Every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet whose contraction route is consumed by a finite-tail diagonal seal factors through the modulus-window rows of Definition 395.24, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal package of Definition 395.20, and the* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *finite-tail witness surface of chapter 559 before any completion-facing read. The route is the finite packet*

$$B, T, L, I, M, E, H, C, G, N$$

together with the downstream seal scope of Definition 395.25. It exports no selected limit, host fixed-point equality, quotient Cauchy name, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket finite tail diagonal seal handoff`

Proof. Start with the modulus-window row scope. It keeps the modulus row M and endpoint seal row E inside the displayed $\mathsf{PicardContraction}^{\uparrow}$ carrier under classifier transport and modulus-window transport.

Next apply the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal package. The regular-sequence and real-facing reads are already exhausted by the same finite carrier rows, so the finite-tail consumer cannot replace them by a completed orbit or a host fixed-point equality.

Finally use the downstream seal scope and the sibling route chapter 559. The finite-tail diagonal seal supplies the selected finite-tail witness between the regular-sequence read and the real-facing seal, and the Picard packet supplies only $B, T, L, I, M, E, H, C, G, N$. Since neither route contains a quotient Cauchy name, selected limit, ambient completeness row, or host fixed-point equality, no completion-facing read can obtain one from the handoff. □

Theorem 395.22 (PicardContraction root consumer triad exhaustion)**.** *The* $\mathsf{Ode}^{\uparrow}$ *and* $\mathsf{NewtonIteration}^{\uparrow}$ *family, the* $\mathsf{RegSeqRat}^{\uparrow}$ *family, and the* $\mathsf{Real}^{\uparrow}$ *family of consumers of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet are jointly exhausted by the same finite triad: the iterate ledger I, the modulus row M, and the endpoint seal row E, together with their surrounding* BHist, hsame, Cont, Pkg, *and* NameCert *rows in*

$$B, T, L, I, M, E, H, C, G, N.$$

No family receives a host fixed point, completed Picard orbit, selected limit, or ambient completeness claim.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket root consumer triad exhaustion`

Proof. First handle $\mathsf{Ode}^{\uparrow}$ and $\mathsf{NewtonIteration}^{\uparrow}$ consumers. Apply Definition 395.4 and Definition 395.9; these bind root-facing reads to the Banach source, contraction row, rational ledger, finite iterate ledger, modulus row, endpoint seal, and displayed transport and continuation routes.

Next handle $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The modulus row finite-tail lock of Definition 395.19 fixes M to I, L, H, and C, and the seal package of Definition 395.20 keeps regular-rational and real-facing reads inside the same carrier.

The three consumer families therefore meet at the finite triad I, M, E and its surrounding BEDC rows. Since the carrier has no further coordinate for a host fixed point, completed orbit, selected limit, or ambient completeness proof, the triad exhausts the root consumer surface. □

Theorem 395.23 (PicardContraction Banach source row scope)**.** *Every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet reads its Banach source, contraction row, rational Lipschitz ledger, finite iterate ledger, and local* NameCert *row only through the displayed* BHist *carrier rows*

$$B, T, L, I, H, C, G, N.$$

The read is mediated by the classifier of Definition 395.2, the finite-iterate closure of Definition 395.3, the Banach ODE boundary of Definition 395.4, and the fixed-point ledger boundary of Definition 395.9. It excludes host fixed points, completed Picard orbits, ambient completeness, and choice-selected limits.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket banach source row scope`

Proof. Project the displayed carrier of Definition 395.1. The rows B, T, L, I, H, C, G, N contain the source, contraction, rational ledger, finite iterate ledger, transports, continuations, package provenance, and local certificate data named in the statement.

Use the classifier of Definition 395.2 to keep comparisons componentwise on those rows. The finite-iterate closure theorem then accounts for every extension of visible iterate data without adding a completed orbit.

The modulus-window transport theorem identifies the M and E rows as the only modulus and endpoint channels adjacent to this Banach source read. The Banach ODE boundary, fixed-point ledger boundary, and finite consumer exhaustion theorem then restrict consumer reads to the same finite packet. Since none of the displayed rows is a host fixed point, an ambient completeness proof, or a selected limiting object, the Banach source row scope is exhausted by the carrier rows. □

Theorem 395.24 (PicardContraction modulus window row scope)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet, the modulus row M and endpoint seal row E are preserved under both the classifier route of Definition 395.2 and the transport route of Definition 395.10. The preserved rows remain part of the finite packet*

$$B, T, L, I, M, E, H, C, G, N$$

and do not become a completed orbit, an ambient completeness claim, a host fixed point, or a choice-selected limit.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket modulus window row scope`

Proof. Read M and E from the carrier of Definition 395.1. They are explicit finite rows of the packet, not derived host objects.

Apply the classifier route of Definition 395.2. Because classifier comparison is componentwise, the modulus and endpoint rows are transported only through the displayed hsame and Cont coordinates.

Apply Definition 395.10 to the same rows. The finite-iterate closure and fixed-point ledger boundary keep the endpoint seal downstream of visible iterate data, while the Banach ODE boundary prevents a consumer from replacing M or E by a host convergence theorem.

Finally, Definition 395.18 exhausts downstream reads by the displayed carrier rows. Thus preserving M and E under classifier and transport routes preserves exactly finite packet evidence. □

Theorem 395.25 (PicardContraction downstream seal scope)**.** *Every* $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer route out of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the rows named by Definition 395.18. Thus downstream seal reads are finite* BHist *reads of*

$$B, T, L, I, M, E, H, C, G, N$$

and never a route through host fixed points, completed orbits, ambient completeness, or choice-selected limits.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket downstream seal scope`

Proof. Begin with the carrier of Definition 395.1 and the finite consumer exhaustion theorem. Together they list exactly the rows available to the four downstream consumer families named in the statement.

For $\mathsf{Ode}^{\uparrow}$ and $\mathsf{NewtonIteration}^{\uparrow}$, the Banach ODE boundary supplies the consumer-facing restriction. For $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, the modulus-window transport and fixed-point ledger boundary keep the read on the displayed modulus and endpoint rows.

Finally, the classifier and finite-iterate closure theorems show that transport and finite concatenation do not add new carrier coordinates. The downstream seal scope is therefore the finite row scope of Definition 395.18, with no completed orbit, host fixed point, ambient completeness proof, or selected limit. □

Theorem 395.26 (PicardContraction metric complete Lipschitz consumer route)**.** *Let $P = (B, T, L, I, M, E, H, C,$ be an accepted $\mathsf{PicardContraction}^{\uparrow}$ packet. Every $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer of P reads the $\mathsf{Banach}^{\uparrow}$ source row through chapter 90, the $\mathsf{CompleteMetric}^{\uparrow}$ boundary through chapter 128, the contraction and Lipschitz rows through chapter 410, and the Cauchy-modulus row M before the endpoint seal E.*

Thus the consumer route is the finite dependency route

$$B, T, L, I, M, E, H, C, G, N$$

with the Banach, complete-metric, Lipschitz, and Cauchy-modulus rows visible before the endpoint seal. It does not export a host fixed point, completed orbit, ambient completeness theorem, selected limit, quotient stream equality, or external real-equality witness.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket metric complete lipschitz consumer route`

Proof. Project the packet P. The carrier rows are the Banach source B, contraction-map row T, rational Lipschitz ledger L, finite iterate ledger I, Cauchy-modulus row M, endpoint seal E, componentwise transport H, displayed continuation C, provenance G, and local naming row N.

Use the existing finite route theorems. The finite-iterate theorem Definition 395.3 keeps iteration inside the displayed Banach and contraction rows, Definition 395.4 routes ODE and Newton consumers through the Banach source and Lipschitz ledgers, and Definition 395.5 places the Cauchy-modulus row before the endpoint seal.

Now read the sibling boundaries. The Banach source is the packet row cited through chapter 90; the complete-metric boundary is the finite convergence boundary of chapter 128; and the contraction and modulus ledgers are the LipschitzMap rows of chapter 410.

It remains to inspect the excluded coordinates. None of $B, T, L, I, M, E, H, C, G, N$ is a host fixed point, completed orbit, ambient completeness theorem, selected limit, quotient stream equality, or external real-equality witness. Hence every listed downstream consumer receives only the displayed metric, complete, Lipschitz, and modulus route before the endpoint seal. □

Theorem 395.27 (PicardContraction Cauchy modulus handoff)**.** *For every accepted $\mathsf{PicardContraction}^{\uparrow}$ packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the row M is the only Cauchy-modulus evidence exported by the packet. Any Cauchy consumer may transport M through the displayed hsame rows H and read it through the displayed Cont routes C, but it cannot replace M by an external convergence proof, a completed Picard orbit, an ambient completeness theorem, or a choice-selected limiting point.

Proof. First isolate the modulus row. Apply Definition 395.10: the carrier exposes M as the exact modulus-window readback row, transported only through the displayed H and C coordinates.

Then apply Definition 395.18. Cauchy-facing reads are among the finite packet reads exhausted by $B, T, L, I, M, E, H, C, G, N$, so no consumer can obtain a second modulus source outside the packet.

The excluded objects are not carrier coordinates. Hence the exported Cauchy evidence is precisely the displayed row M with its finite transport and continuation witnesses. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket cauchy modulus handoff`

Theorem 395.28 (PicardContraction regular sequence consumer route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the finite rows*

$$B, T, L, I, M, E, H, C, G, N.$$

The read may use the Banach source row B, contraction-map row T, rational Lipschitz ledger L, finite iterate ledger I, modulus row M, endpoint and seal row E, transports H, continuations C, package provenance G, and local naming row N; it cannot read a completed orbit or a host-level regular sequence built outside the packet.

Proof. Start from Definition 395.27. It gives the regular-sequence consumer its only modulus evidence, namely the displayed row M.

Next use Definition 395.3 for the visible iterate data and Definition 395.17 for the rational Lipschitz ledger. These theorems keep the construction on B, T, L, I and do not create a stronger host contraction claim.

Finally apply Definition 395.18. Its consumer list includes $\mathsf{RegSeqRat}^{\uparrow}$, so the regular-sequence route is exhausted by $B, T, L, I, M, E, H, C, G, N$ and cannot pass through an omitted completed orbit. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket regseqrat consumer route`

Theorem 395.29 (PicardContraction real seal noncompletion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet receives only the displayed endpoint and seal row E, together with the finite route through*

$$B, T, L, I, M, H, C, G, N.$$

The packet does not export a fixed point, a Banach fixed-point theorem, ambient completeness, a completed Picard orbit, or a selected limit as part of the real-facing seal.

Proof. Use Definition 395.27 to bind the Cauchy evidence to M and Definition 395.28 to bind the regular-sequence route to the displayed finite rows.

Now read the real-facing boundary through Definition 395.4. The theorem permits consumers to cite the packet rows and the endpoint/seal coordinate, but it does not provide a host fixed point or a completed orbit.

Finally apply Definition 395.18. Since $\mathsf{Real}^{\uparrow}$ reads are exhausted by the carrier, the seal remains a finite endpoint row rather than a completion theorem or selected limiting object. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket real seal noncompletion`

Theorem 395.30 (PicardContraction public carrier surface)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the public carrier surface consists exactly of the $\mathsf{Banach}^{\uparrow}$ *source row B, contraction-map row T, rational Lipschitz ledger L, finite iterate ledger I, Cauchy-modulus row M, endpoint and* $\mathsf{Real}^{\uparrow}$ *seal*

row E, componentwise hsame *transports H, displayed* Cont *routes C,* Pkg *provenance G, and local* NameCert *row N. This surface is public over* Banach$^{\uparrow}$, Rat$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *and it contains no fixed point, ambient completeness proof, completed orbit, or choice-selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public carrier surface`

Proof. Start with the carrier of Definition 395.1. Its coordinates are precisely $B, T, L, I, M, E, H, C, G, N$, with no additional host fixed-point coordinate.

Use the classifier of Definition 395.2 and the classifier stability theorem. They compare and transport accepted packets componentwise, so the public surface remains the displayed finite carrier rather than a host theorem about contractions.

Finally apply Definition 395.18. It exhausts downstream ODE, Newton, regular-rational, and real-name reads by the same carrier rows, preserving the exclusions in the statement. □

Theorem 395.31 (PicardContraction public iterate ledger)**.** *The only public iteration evidence exported by an accepted* PicardContraction$^{\uparrow}$ *packet is the finite iterate ledger I together with the rational ratio ledger L and the displayed transport and continuation rows that read them. Downstream* NewtonIteration$^{\uparrow}$ *and* Ode$^{\uparrow}$ *consumers may use this finite evidence, but they receive no completed Picard orbit, host fixed point, ambient completeness theorem, or selected limiting point.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public iterate ledger`

Proof. First isolate the finite iteration rows through Definition 395.3. Concatenating visible iterate data stays inside the displayed ledger I and the surrounding carrier rows.

Next use Definition 395.17. The rational Lipschitz evidence remains the finite row L and cannot be replaced by a stronger host contraction estimate.

Finally apply Definition 395.4 and Definition 395.9. They are the consumer-facing restrictions for Ode$^{\uparrow}$ and NewtonIteration$^{\uparrow}$, and they rule out every omitted completion or fixed-point object. □

Theorem 395.32 (PicardContraction public seal export)**.** *For every accepted* PicardContraction$^{\uparrow}$ *packet, the public* Real$^{\uparrow}$ *seal export factors through the modulus row M, the endpoint and seal row E, and the finite route*

$$B, T, L, I, H, C, G, N.$$

The export is a finite BEDC carrier read over the public carrier surface of Definition 395.30; it does not claim a fixed point, ambient completeness, completed orbit, or choice-selected limit.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public seal export`

Proof. Begin with Definition 395.27. It identifies M as the only Cauchy-modulus evidence available to seal-facing consumers.

Then use Definition 395.28. The regular-sequence route reads the same finite carrier rows and supplies no completed orbit outside the packet.

Finally apply Definition 395.29 together with Definition 395.30. The seal is therefore the displayed endpoint row read through the finite route, with the fixed-point and completeness exclusions still in force. □

Theorem 395.33 (PicardContraction CompleteMetric handoff)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

every $\mathsf{CompleteMetric}^{\uparrow}$*-facing read of* P *factors through the finite rows*

$$B, T, L, I, M, E, H, C, G, N.$$

The row M *is the explicit Cauchy modulus exposed to the downstream metric consumer, and the row* E *is the endpoint and seal row available to real-facing consumers. The handoff exports no host fixed point, ambient completeness theorem, completed orbit, quotient stream, or selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket complete metric handoff`

Proof. First read the public carrier surface through Definition 395.30. It exposes exactly the rows $B, T, L, I, M, E, H, C, G, N$ and excludes the fixed-point and completion objects named in the statement.

Next isolate the metric evidence. By Definition 395.27, the only Cauchy-modulus evidence exported by the packet is the displayed row M.

The regular-sequence side is already finite. Apply Definition 395.28; any regular-rational read is exhausted by the same carrier rows and cannot pass through a completed orbit outside the packet.

Finally use Definition 395.32. The real-facing endpoint and seal data factor through M, E, and the displayed finite route, so a $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives the explicit modulus handoff without importing host completeness, quotient-stream data, or a selected limiting point. □

Theorem 395.34 (PicardContraction CauchyRate source extraction)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the finite iterate ledger I*, rational Lipschitz row* L*, Cauchy-modulus row* M*, endpoint seal* E*, displayed* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* G*, and local* NameCert *row* N *determine the* $\mathsf{CauchyRate}^{\uparrow}$*-facing source packet read from* P*. The extracted packet is a finite read of the same* BHist *carrier and contains no completed Picard orbit, ambient completeness proof, Banach fixed-point object, quotient stream equality, or selected limit.*

Proof. Project the public carrier surface of Definition 395.30. The rows named in the statement are visible coordinates of the finite packet, and the theorem excludes every host-level completion object from that surface.

Next isolate the effective-rate data. Apply Definition 395.31 to bind the visible iteration evidence to I and L, and use Definition 395.27 to bind the tolerance schedule to M. The endpoint and real-facing seal are then the displayed row E rather than a selected limit.

Finally use the transport and provenance rows H, C, G, N to package the read as a $\mathsf{CauchyRate}^{\uparrow}$ source. Since each row is already part of P, the extracted source is determined by the finite carrier and cannot introduce an additional completion or fixed-point premise. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket cauchyrate source extraction`

Theorem 395.35 (PicardContraction CauchyRate real seal compatibility)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal export from an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the* $\mathsf{CauchyRate}^{\uparrow}$*-facing source packet of Definition 395.34. In particular, a real-facing consumer must read the finite packet rows*

$$L, I, M, E, H, C, G, N$$

before it reads the seal, and it has no route that bypasses the finite rate packet by invoking ambient completeness, a completed orbit, a host fixed point, or a selected real representative.

Proof. Start with Definition 395.32. It factors the $\mathsf{Real}^{\uparrow}$ seal through the modulus row M, endpoint row E, and the finite carrier route surrounding B, T, L, I, H, C, G, N.

Now replace the rate-facing part of that route by the source extraction theorem. The rows L, I, M, E, H, C, G, N are exactly the finite rate, endpoint, transport, continuation, package, and naming rows used by the $\mathsf{CauchyRate}^{\uparrow}$ source packet.

It remains to rule out escape around the extracted source. Apply Definition 395.29; the real-facing boundary exports only the finite endpoint and seal row. Therefore seal compatibility factors through the extracted CauchyRate source and not through any omitted completion object. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket cauchyrate real seal compatibility`

Theorem 395.36 (PicardContraction finite-rate seal triangle)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the $\mathsf{CauchyRate}^{\uparrow}$ *source extraction,* $\mathsf{Real}^{\uparrow}$ *seal compatibility, and public seal export form a single finite triangle over the rows* L, I, M, E, H, C, G, N*. A downstream consumer may move from the Picard packet to the CauchyRate source, from that source to the Real seal, and back to the public seal export only through the displayed finite carrier rows; it cannot insert ambient completeness, a completed Picard orbit, a host fixed point, quotient stream equality, or a selected real representative at any side of the triangle.*

Proof. First read the source side through Definition 395.34. It identifies the finite rate-facing packet with the visible iterate, ratio, modulus, endpoint, transport, route, provenance, and local naming rows.

Next read the seal side through Definition 395.35. The real-facing path must pass through those same finite rate rows before the seal is read.

Finally close the public side with Definition 395.32. The public seal export factors through M, E, and the same finite carrier route, so the three sides share the displayed rows and exclude the omitted completion objects. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket finite rate seal triangle`

Theorem 395.37 (PicardContraction Newton ODE shared modulus route)**.** *Every* $\mathsf{NewtonIteration}^{\uparrow}$ *or* $\mathsf{Ode}^{\uparrow}$ *consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet uses the same finite modulus row* M*, rational ratio ledger* L*, and finite iterate ledger* I*. The two consumer families may read these rows through the displayed* hsame *transports* H*,* Cont *routes* C*,* Pkg *provenance* G*, and local naming row* N*, but neither consumer receives a second modulus source, a completed orbit, a host fixed point, an ambient completeness proof, or a selected limiting state.*

Proof. Start with the public iterate ledger. By Definition 395.31, the only iteration evidence exported to Newton and ODE consumers is the finite ledger I together with L and the displayed transport and route rows.

Attach the Cauchy evidence using Definition 395.27. This theorem identifies M as the only modulus evidence available from the packet.

Now apply Definition 395.43 and Definition 395.48. They exhaust both consumer families by the same finite packet rows, so Newton and ODE reads share L, I, M rather than branching to omitted completion or fixed-point data. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket newton ode shared modulus route`

Theorem 395.38 (PicardContraction RegSeqRat Real rate non-escape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet that reads rate information factors through the finite rows*

$$L, I, M, E, H, C, G, N.$$

The consumer may use the rational ratio ledger, finite iterate ledger, Cauchy modulus row, endpoint and seal row, transport rows, continuation routes, provenance, and local naming row, but it cannot bypass the finite rate packet through ambient completeness, a completed orbit, a host fixed point, quotient stream equality, selected limits, or extra real representatives.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket regseqrat real rate nonescape`

Proof. For regular-rational reads, use Definition 395.28. The route binds the consumer to the finite carrier rows and excludes a completed orbit built outside the packet.

For real-facing reads, use Definition 395.29. The seal boundary supplies only the endpoint and finite route, not a host fixed point or a selected limit.

Finally apply Definition 395.36. It forces both consumer families to pass through the extracted CauchyRate source and the public seal route, so no quotient stream equality, ambient completeness row, or extra real representative can be inserted around the finite rate packet. □

Theorem 395.39 (PicardContraction CauchyRate root-unblock package)**.** *The public* $\mathsf{PicardContraction}^{\uparrow}$ *surface assembles a* $\mathsf{CauchyRate}^{\uparrow}$*-facing root-unblock package for downstream* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*, and* $\mathsf{NewtonIteration}^{\uparrow}$ *consumers. The package consists of the carrier surface of Definition 395.30, the iteration ledger of Definition 395.31, the seal export of Definition 395.32, the source extraction of Definition 395.34, and the real-seal compatibility of Definition 395.35. Its conclusion is a finite rate/modulus bridge only; it is not a Banach fixed-point theorem, an ambient completeness theorem, a completed-orbit construction, or a selected-limit principle.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket cauchyrate root unblock package`

Proof. First assemble the public surface. The carrier theorem gives the visible rows $B, T, L, I, M, E, H, C, G, N$, and the public iterate ledger identifies the only iteration evidence available to root-facing consumers.

Next attach the seal route. The public seal export sends real-facing reads through M, E, and the finite carrier route, while the source extraction theorem repackages the rate-facing rows as the $\mathsf{CauchyRate}^{\uparrow}$ source packet.

Finally use real-seal compatibility to bind the downstream real path to that finite source. The cited theorems already exclude fixed points, completed orbits, ambient completeness, and selected limits, so the assembled package is exactly the finite rate/modulus bridge claimed in the statement. □

Theorem 395.40 (PicardContraction finite-modulus obligation triad)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the Banach source row B, the rational Lipschitz-ratio ledger L, and the Cauchy-modulus window M jointly form the finite $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *obligation surface for downstream modulus reads. The triad is read together with the contraction row T, finite iterate ledger I, endpoint seal E,*

componentwise hsame *transports H, displayed* Cont *routes C,* Pkg *provenance G, and local naming row N, and it contains no Banach fixed point, completed orbit, ambient completeness theorem, or selected limit.*

Proof. Unpack Definition 395.1. The rows B, L, and M are visible coordinates of the same finite packet, while T, I, E, H, C, G, and N record how those rows are consumed and transported.

Use Definition 395.11 for the source row and Definition 395.12 for the ratio ledger. These rows keep Banach and Lipschitz evidence inside the displayed carrier rather than in an external contraction principle.

Finally apply Definition 395.27 and Definition 395.19. They identify M as the only modulus evidence and lock it to I, L, H, and C. The three displayed rows therefore form the finite modulus obligation surface named in the statement. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket finite modulus obligation triad`

Theorem 395.41 (PicardContraction ratio-window carrier transport)**.** *Let*

$$P = (B, T, L, I, M, E, H, C, G, N)$$

and

$$P' = (B', T', L', I', M', E', H', C', G', N')$$

be accepted PicardContraction$^{\uparrow}$ *packets related by the finite classifier. Every* hsame*-compatible transport through the displayed* Cont *routes preserves the rational ratio ledger and the Cauchy-modulus window: L' and M' are read as the transported images of L and M inside the same carrier surface, not as external Lipschitz or convergence witnesses.*

Proof. Start with the classifier of Definition 395.2. It compares the accepted packets coordinate by coordinate through the displayed hsame and Cont rows, so the ratio and modulus coordinates are transported as packet rows.

Apply Definition 395.7 and Definition 395.10. The first theorem keeps the carrier comparison on the finite Banach, map, ratio, iterate, modulus, endpoint, provenance, and naming rows; the second identifies M and E as exact transported readback rows.

It remains to bind the ratio ledger. Use Definition 395.17 and Definition 395.12. They keep the rational ledger on L with its finite iterate and continuation context. Thus transport preserves both the ratio ledger and the modulus window without adding any host-level contraction or completion object. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket ratio window carrier transport`

Theorem 395.42 (PicardContraction Lean ratio-ledger nonexpansion sync)**.** *Let*

$$P = (B, T, L, I, M, E, H, C, G, N)$$

be an accepted PicardContraction$^{\uparrow}$ *packet whose finite iterate ledgers concatenate and whose Lipschitz row is the displayed positive rational ratio row below the unit bound. The concatenated packet reuses the transported ratio ledger L as its finite nonexpanding ledger row. It exports no stronger host contraction, no Banach fixed-point theorem, no completed orbit, no ambient completeness proof, and no selected limit.*

Proof. Apply Definition 395.17. The concatenated iterate ledger is read through the same positive rational row below the unit bound, so the concatenated packet cannot enlarge the ratio evidence beyond the displayed finite ledger.

Next use Definition 395.41. It keeps the ratio ledger and the Cauchy-modulus window inside the accepted carrier rows rather than turning them into an external host contraction or convergence witness.

Finally combine Definition 395.40 and Definition 395.46. They consume the same finite rows $B, T, L, I, M, E, H, C, G, N$: B for the source boundary, T for the contraction row, L for the rational ratio ledger, I for finite iterates, M for the modulus window, E for the endpoint seal, H and C for transport and displayed routes, G for Pkg provenance, and N for the local naming row. Since the public boundary has no coordinate for a stronger contraction principle, fixed point, completed orbit, ambient completeness proof, or selected limit, the concatenated packet exports only the transported finite ledger. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket lean ratio ledger nonexpansion sync`

Theorem 395.43 (PicardContraction Banach ODE consumer non-escape)**.** *Every* $\mathsf{Ode}^{\uparrow}$ *or* $\mathsf{NewtonIteration}^{\uparrow}$ *consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet can read only the finite Banach row B, contraction row T, rational ratio ledger L, finite iterate ledger I, Cauchy-modulus row M, endpoint seal E,* Pkg *provenance G, local* NameCert *row N, and the surrounding* BHist, hsame, *and* Cont *evidence. No ODE or Newton consumer receives a Banach fixed-point object, completed Picard orbit, ambient completeness proof, or selected limiting state from this packet.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket banach ode consumer nonescape`

Proof. Read the packet through Definition 395.1. The consumer-visible coordinates are exactly $B, T, L, I, M, E, H, C, G, N$.

For the ODE and Newton side, apply Definition 395.4. It already factors these consumer reads through the Banach source, contraction row, rational ledger, finite iterate rows, explicit modulus row, and endpoint/seal routes.

Now attach the modulus obligation and transport rows. The triad theorem Definition 395.40 binds B, L, and M as one finite obligation surface, while Definition 395.41 preserves that surface under displayed classifier transport. Finally Definition 395.18 exhausts every listed consumer by the same finite carrier rows, proving non-escape. □

Theorem 395.44 (PicardContraction public NameCert export)**.** *Every public* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *read used by* $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRate}^{\uparrow}$ *consumers factors through the displayed rows*

$$B, T, L, I, M, E, H, C, G, N$$

of the accepted $\mathsf{PicardContraction}^{\uparrow}$ *packet. The exported public surface assembles the carrier, classifier, finite-iterate closure, Banach and ODE boundary, modulus-window transport,* $\mathsf{CauchyRate}^{\uparrow}$ *extraction, real-seal compatibility,* Pkg *provenance, and local naming row into* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$. *No fixed point, completed orbit, ambient completeness object, quotient stream equality, or selected limit is exported from this public naming surface.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public namecert export`

Proof. Project the carrier of Definition 395.1. Its public coordinates are the ten displayed rows $B, T, L, I, M, E, H, C, G, N$.

Apply Definition 395.30, Definition 395.7, and Definition 395.3. These give the public row inventory, classifier transport, and finite iteration scope while keeping host fixed points, completed Picard orbits, ambient completeness proofs, quotient stream equalities, and selected limits outside the export surface.

For rate and modulus reads, use Definition 395.39. It routes the $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers through the same carrier, iterate-ledger, seal, source-extraction, and real-seal rows.

For root-facing ODE and Newton reads, apply Definition 395.43. The remaining consumer families are then exhausted by Definition 395.22. Hence every public $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ read has the stated source and no omitted completion, fixed-point, stream-equality, or limit-selection object can escape through the certificate. □

Theorem 395.45 (PicardContraction public Banach CauchyRate factorization)**.** *Every public* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{CauchyRate}^{\uparrow}$ *consumer of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the finite rows*

$$B, T, L, I, M, E, H, C, G, N.$$

The factorization reads the Banach source, contraction row, rational ratio ledger, finite iterate ledger, modulus window, endpoint seal, transports, routes, provenance, and local NameCert *row, and it exports no host fixed-point or completed-orbit object.*

Proof. Begin with the public export theorem Definition 395.44. It identifies the complete public certificate surface with the displayed packet rows.

For Banach-facing reads, apply Definition 395.30 and Definition 395.40. These bind B, T, L, and M to the same finite carrier and obligation triad.

For rate-facing reads, use Definition 395.39. The package factors through the same carrier, iterate, modulus, seal, transport, provenance, and naming rows, so every public Banach or CauchyRate consumer is exhausted by $B, T, L, I, M, E, H, C, G, N$. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public banach cauchyrate factorization`

Theorem 395.46 (PicardContraction public real consumer boundary)**.** *Public* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, *and* $\mathsf{NewtonIteration}^{\uparrow}$ *consumers of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet receive only the finite packet rows*

$$B, T, L, I, M, E, H, C, G, N.$$

They receive no fixed point, completed Picard orbit, ambient completeness proof, selected limit, quotient stream equality, or extra real representative from the public $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *export.*

Proof. First use Definition 395.44. It packages the public certificate as the displayed finite rows and states the missing completion and limit objects are not exported.

For real and regular-rational consumers, apply Definition 395.39. It routes the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads through the finite rate and seal rows.

For ODE and Newton consumers, apply Definition 395.43. The same finite rows exhaust the consumer surface, so no downstream real-facing or dynamical consumer receives data outside the packet. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public real consumer boundary`

Theorem 395.47 (PicardContraction public Banach source modulus seal)**.** *For every accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet*

$$P = (B, T, L, I, M, E, H, C, G, N),$$

the public packet exports exactly the $\mathsf{Banach}^{\uparrow}$ *source row* B*, contraction row* T*, Lipschitz ratio ledger* L*, finite iterate ledger* I*, Cauchy-modulus row* M*, endpoint seal* E*, and the displayed* BHist*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance rows* H, C, G, N*. This export is a finite source/modulus/seal surface and contains no host fixed point, completed orbit, ambient completeness proof, external contraction principle, or selected limit.*

Proof. Project the carrier of Definition 395.1. Its visible rows are precisely $B, T, L, I, M, E, H, C, G, N$, so the public export starts from the finite packet rather than from a host contraction theorem.

Use Definition 395.2, Definition 395.3, and Definition 395.41. These rows keep the source, contraction, ratio, and iterate data inside the finite classifier and its displayed transport surface.

For the modulus and seal side, apply Definition 395.34 and Definition 395.35. Together with Definition 395.30, they identify M and E as the public Cauchy-rate and real-seal rows. The exported packet is therefore exactly the stated finite source/modulus/seal surface. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public banach source modulus seal`

Theorem 395.48 (PicardContraction public ODE Newton export)**.** *Every* $\mathsf{Ode}^{\uparrow}$ *or* $\mathsf{NewtonIteration}^{\uparrow}$ *consumer route out of an accepted* $\mathsf{PicardContraction}^{\uparrow}$ *packet factors through the public packet of Definition 395.47. Such a consumer can read the displayed rows* $B, T, L, I, M, E, H, C, G, N$*, but it cannot read a completed Picard orbit, host fixed point, ambient completeness proof, selected limit, or external Banach fixed-point object.*

Proof. Start with Definition 395.4. It already routes ODE and Newton reads through the Banach source, contraction row, rational ledger, finite iterate rows, explicit modulus row, and endpoint/seal routes.

Apply Definition 395.43. This exhausts the same consumer families by the finite packet rows and excludes completed orbits, host fixed points, ambient completeness proofs, and selected limits.

Finally use Definition 395.47 to name the public packet through which these rows are exported. The ODE and Newton routes therefore factor through that public surface and no omitted completion object is readable. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public ode newton export`

Theorem 395.49 (PicardContraction public NameCert package)**.** *The public* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *export is the package assembled from the source/modulus/seal surface of Definition 395.47, the ODE and Newton route boundary of Definition 395.48, and the local obligation surface of Definition 395.6. It exports exactly the displayed finite rows* $B, T, L, I, M, E, H, C, G, N$ *for* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{CauchyRate}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*, and* $\mathsf{NewtonIteration}^{\uparrow}$ *readers.*

Proof. The local obligation surface supplies the carrier, classifier, finite-iterate, modulus-readback, and ledger non-escape rows of the naming certificate.

The public source/modulus/seal theorem identifies the rows exported to Banach, Cauchy-rate, regular-rational, and real-facing consumers. The public ODE/Newton theorem identifies the same finite packet as the only route for dynamical consumers.

Repack these rows under the local $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ entry. Since each component is already one of $B, T, L, I, M, E, H, C, G, N$, the assembled public package introduces no fixed point, completed orbit, ambient completeness proof, external contraction principle, or selected limit. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket public namecert package`

Theorem 395.50 (PicardContraction bridged source-window package)**.** *The checked public export, Banach and ODE consumer non-escape, ratio-ledger non-expansion, and root source-window checked handoff form the bridge packet from the concrete* PicardContraction$^{\uparrow}$ *carrier to the public interface. Concretely, every public read of an accepted packet factors through the displayed rows*

$$B, T, L, I, M, E, H, C, G, N$$

and the source-window handoff target `BEDC.Derived.PicardContractionUp.PicardContractionPacket_root_s` *records the checked root-side handoff for that same finite packet surface.*

Proof. Start with the public NameCert package. Definition 395.49 assembles the source/modulus/seal export, ODE and Newton route boundary, and local obligation surface as the public package over $B, T, L, I, M, E, H, C, G, N$.

Next use consumer non-escape and ratio control. Definition 395.43 keeps ODE and Newton reads inside the same finite carrier, while Definition 395.17 prevents the rational Lipschitz row from expanding into a stronger host contraction claim.

Finally attach the checked source-window handoff. The Lean target named in the statement is the source-window handoff for the root-side surface, so the paper bridge is the displayed public package together with the checked handoff, not an ambient Banach fixed-point theorem or a completed-orbit construction. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket bridged source window package`

Closure status for NameCert$_{\text{PicardContraction}^{\uparrow}}$.

Constructive story: A PicardContraction$^{\uparrow}$ packet is generated from a Banach$^{\uparrow}$ source row, a displayed contraction-map row, a positive rational Lipschitz-ratio ledger below the unit bound, a finite Picard-iterate ledger, an explicit Cauchy-modulus row, endpoint and seal rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PicardContraction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 395.1, Definition 395.2, Definition 395.3, Definition 395.4, Definition 395.6, Definition 395.7, Definition 395.8, Definition 395.9, Definition 395.10, Definition 395.17, Definition 395.18, Definition 395.44, Definition 395.34, Definition 395.35, Definition 395.36, Definition 395.37, Definition 395.38, Definition 395.39, Definition 395.45, Definition 395.46, Definition 395.47, Definition 395.48, Definition 395.49, and Definition 395.50 describe the mature consumer scope: the finite packet, classifier, finite-iterate closure, consumer boundary, NameCert surface, classifier stability, contraction-step package, fixed-point ledger boundary, modulus-window transport, ratio-ledger nonexpansion, finite consumer exhaustion, public certificate export, CauchyRate source extraction, CauchyRate Real seal compatibility, finite-rate seal triangle, Newton ODE shared modulus route, RegSeqRat Real rate non-escape, CauchyRate root-unblock package, Banach and CauchyRate factorization, real-facing consumer boundary, Banach source/modulus/seal export, ODE/Newton public export, public NameCert package, and source-window bridge packet.

Formal verification: FormalStatus(NameCert$_{\text{PicardContraction}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close full Banach fixed-point convergence, uniqueness of a host fixed point, ambient completeness, a choice-selected limit, or bridge checking.

Upgrade path: The theory axis is mature; the next obligation is formal-axis Lean verification for the displayed finite packet, CauchyRate, RegSeqRat, Real, ODE, and Newton consumer rows.

Definition 395.51 (PicardContraction root source-window packet)**.** *A* PicardContraction$^{\uparrow}$ *root source-window packet is a finite extension*

$$R = (B, T, L, I, M, E, H, C, G, N, U)$$

of the carrier rows of Definition 395.1. The rows $B, T, L, I, M, E, H, C, G, N$ *are the displayed* Banach$^{\uparrow}$ *source row, contraction-map row, rational Lipschitz ledger, finite Picard-iterate ledger,*

Cauchy-modulus window, endpoint and seal row, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ *row. The extra row U is a finite downstream consumer request row naming which of the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, *or* $\mathsf{CauchyRate}^{\uparrow}$ *routes is being read from the same packet.*

The row U is not an additional analytic object. It is a request index over the public carrier surface of Definition 395.30 and the public naming export of Definition 395.44. Thus the packet has no coordinate for a Banach fixed point, completed Picard orbit, ambient completeness theorem, quotient stream equality, or selected limit.

The sibling source surfaces are already the displayed NameCert boundaries: the Banach source surface in chapter 90, the Cauchy-rate consumer boundary in chapter 415, and the regular-rational naming boundary in chapter 371. Thus U only selects among downstream consumer families whose rows are present on those finite surfaces; it does not add analytic data beyond the packet rows $B, T, L, I, M, E, H, C, G, N$.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket request row certificate`

Theorem 395.52 (PicardContraction root source-window admission)**.** *Every packet $R = (B, T, L, I, M, E, H, C, G$ satisfying Definition 395.51 is admitted by the root* $\mathsf{PicardContraction}^{\uparrow}$ *consumer surface as a finite* BHist *source-window. Its analytic rows are exactly the carrier rows of Definition 395.1, its public read surface is the one described by Definition 395.30, and its consumer request row U may only choose among the displayed downstream routes named by Definition 395.44.*

Proof. Project the first ten coordinates of R. They are precisely the $B, T, L, I, M, E, H, C, G, N$ rows of Definition 395.1, so carrier admission is the ordinary finite BHist admission for the Picard packet.

Read the same rows through Definition 395.30. The public surface lists the Banach source, contraction row, Lipschitz ledger, finite iterate ledger, modulus row, endpoint seal, hsame transports, Cont routes, Pkg provenance, and local NameCert row.

Finally inspect U. By definition it only records which downstream consumer route is requested, and Definition 395.44 already restricts those routes to the displayed public rows. Hence admission adds no completed orbit, fixed point, ambient completeness witness, quotient stream equality, or selected limit. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket admission`

Theorem 395.53 (PicardContraction root modulus consumer lock)**.** *For every admitted root source-window packet $R = (B, T, L, I, M, E, H, C, G, N, U)$, any* $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read of the modulus or seal is locked to the displayed rows*

$$L, I, M, E, H, C, G, N.$$

The lock is the finite source extraction of Definition 395.34 together with the public carrier and public NameCert *rows of Definition 395.30 and Definition 395.44.*

Proof. Begin with the admitted packet of Definition 395.52. Its modulus row M, endpoint seal E, transports H, continuations C, package row G, and naming row N are visible carrier coordinates, not hidden completion data.

Apply Definition 395.34. It identifies the $\mathsf{CauchyRate}^{\uparrow}$-facing source as a finite read of L, I, M, E, H, C, G, N from the same BHist packet.

For $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, use Definition 395.44. The public $\mathsf{NameCert}_{\mathsf{PicardContraction}^{\uparrow}}$ export routes those reads through the same displayed finite rows. Therefore the consumer request row U can select only that finite modulus-and-seal route and cannot replace it by a Banach fixed-point theorem, completed orbit, ambient completeness proof, quotient stream equality, or selected limit. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket modulus consumer lock`

Theorem 395.54 (PicardContraction root downstream handoff exhaustion)**.** *Every downstream handoff out of an admitted root source-window packet* $R = (B, T, L, I, M, E, H, C, G, N, U)$ *is exhausted by the rows*

$$B, T, L, I, M, E, H, C, G, N, U.$$

For $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{NewtonIteration}^{\uparrow}$*, and* $\mathsf{CauchyRate}^{\uparrow}$ *consumers, the handoff is a projection of the displayed* BHist *carrier, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *row, and the request row* U*. No consumer route invokes a Banach fixed point, completed Picard orbit, ambient completeness theorem, external contraction principle, quotient stream equality, or selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket downstream handoff exhaustion`

Proof. First project the carrier coordinates. By Definition 395.51, the rows before U are exactly the rows of Definition 395.1; the extra row merely names the requested downstream route.

For rate, regular-rational, and real-facing consumers, apply Definition 395.53. It binds the modulus and seal handoff to L, I, M, E, H, C, G, N inside the same packet.

For ODE and Newton consumers, read the public surface through Definition 395.30 and then through Definition 395.44. These theorems route the Banach source, contraction row, ratio ledger, iterate ledger, modulus row, endpoint seal, transport rows, continuation rows, provenance, and local name through the finite public packet.

The request row U has no data beyond the selected consumer family. Since each family has already been routed through the displayed carrier and naming rows, the handoff is exhausted by $B, T, L, I, M, E, H, C, G, N, U$. □

Theorem 395.55 (PicardContraction source-window formal target)**.** *For a root source-window packet*

$$R = (B, T, L, I, M, E, H, C, G, N, U),$$

the displayed rows $B, T, L, I, M, E, H, C, G, N, U$ *constitute the exact paper-side formal target for the future Lean encoding of the* $\mathsf{PicardContraction}^{\uparrow}$ *source window. The target consists of carrier admission, modulus consumer lock, downstream handoff exhaustion, and the exclusion of fixed-point data, completion data, quotient stream equality, and selected-limit data.*

Thus the intended formal surface is the finite packet admitted by Definition 395.52, constrained by Definition 395.53, and exhausted by Definition 395.54.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket source window formal target`

Proof. The admission theorem first fixes the carrier. It identifies the first ten rows with the displayed $\mathsf{PicardContraction}^{\uparrow}$ carrier and reads U only as a downstream consumer request row.

The modulus-lock theorem then fixes the analytic handoff. Any $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read of the modulus or seal is forced through L, I, M, E, H, C, G, N inside the same finite packet.

The downstream-exhaustion theorem closes the consumer side. It covers the named $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, and $\mathsf{CauchyRate}^{\uparrow}$ routes and shows that each route is a projection of $B, T, L, I, M, E, H, C, G, N, U$.

Combining these three statements gives the formal target surface. Since each cited theorem explicitly excludes fixed points, completed Picard orbits, ambient completion, quotient stream equality, and selected limits, the future Lean target has no additional row to encode. □

Theorem 395.56 (PicardContraction root source-window obligation package)**.** *For a root source-window packet*

$$R = (B, T, L, I, M, E, H, C, G, N, U),$$

the consumer-facing root obligation package is exactly the conjunction of root source-window admission, root modulus consumer lock, downstream handoff exhaustion, and the source-window formal target. Equivalently, an accepted package is obtained by composing Definition 395.52, Definition 395.53, Definition 395.54, and Definition 395.55; it contains no Banach fixed-point existence theorem, completed Picard orbit, quotient stream equality, or selected limit.

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket obligation package`

Proof. Use root source-window admission first. It fixes the finite carrier rows $B, T, L, I, M, E, H, C, G, N$ and reads U only as a downstream request row.

Next use the root modulus consumer lock. It forces every $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read of the modulus or seal through L, I, M, E, H, C, G, N inside the same packet.

Then use downstream handoff exhaustion. It shows that the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, and $\mathsf{CauchyRate}^{\uparrow}$ consumer routes are projections of the displayed rows $B, T, L, I, M, E, H, C, G, N, U$.

Finally use the source-window formal target. That theorem identifies the exact paper-side formal surface as admission, modulus lock, downstream exhaustion, and the exclusions of fixed-point data, completion data, quotient stream equality, and selected-limit data.

Combining the four statements gives the advertised package. Since each component already leaves Banach fixed-point existence, completed Picard orbit, quotient stream equality, and selected limits outside the packet, the conjunction has no additional analytic coordinate. □

Theorem 395.57 (PicardContraction root source-window checked handoff)**.** *For a root source-window packet*

$$R = (B, T, L, I, M, E, H, C, G, N, U),$$

the same finite rows certified by the checked public export targets `PicardContractionPacket_public_namecert_ea` `PicardContractionPacket_banach_ode_consumer_nonescape`*, and* `PicardContractionPacket_ratio_ledger_` *consume the paper-side source window. The request row U may select only one of the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{NewtonIteration}^{\uparrow}$*, or* $\mathsf{CauchyRate}^{\uparrow}$ *consumer families, and each selected family reads the packet through $B, T, L, I, M, E, H, C, G, N$ rather than through a Banach fixed point, completed orbit, quotient stream equality, or selected limit.*

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionPacket root source window checked handoff`

Proof. First use the public naming export. It exposes the Banach source, contraction row, ratio ledger, finite iterate ledger, modulus row, endpoint row, transport row, route row, provenance row, and local naming row as the checked public NameCert surface.

Next use the Banach-to-ODE non-escape row. It routes ODE-facing consumer reads through the same finite carrier coordinates and blocks any replacement of the source window by a fixed-point theorem, completed orbit, or ambient completion principle.

Finally use the ratio-ledger non-expansion row. It keeps rate-facing and ratio-facing consumers inside the displayed Lipschitz and iterate ledger coordinates. Since U only names which of the five consumer families is requested, composing the three checked rows gives the advertised handoff without adding any new analytic coordinate. □

Theorem 395.58 (PicardContraction root source-window determinacy)**.** *For a root source-window packet*

$$R = (B, T, L, I, M, E, H, C, G, N, U),$$

the request row U determines only the downstream consumer family and never changes the source-window rows $B, T, L, I, M, E, H, C, G, N$. If two reads have the same $B, T, L, I, M, E, H, C, G, N$ rows and the same request family, then they have the same finite handoff surface for $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, *or* $\mathsf{CauchyRate}^{\uparrow}$.

Proof. Use root source-window admission from Definition 395.52. It fixes the carrier rows $B, T, L, I, M, E, H, C, G,$ and reads U only as a downstream request row.

For $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ requests, apply the modulus consumer lock of Definition 395.53. The rate, rational-readback, and real-facing handoffs are forced through the displayed Lipschitz row, iterate ledger, modulus row, endpoint seal, transport, replay, provenance, and local name.

For $\mathsf{Ode}^{\uparrow}$ and $\mathsf{NewtonIteration}^{\uparrow}$ requests, apply downstream handoff exhaustion from Definition 395.54. The two consumer families are projections of the same finite packet and do not introduce a second source window.

Finally cite the checked handoff of Definition 395.57. It excludes replacement of the finite handoff by fixed-point data, completed-orbit data, quotient stream equality, completion data, or selected-limit data.

Thus two reads that agree on the source-window rows and the request family have no remaining coordinate on which their finite handoff surfaces can diverge. □

Lean4 ✓ `BEDC.Derived.PicardContractionUp.PicardContractionRootSourceWindowPacket determinacy`

Chapter 396

A Concrete Naming Certificate for $\mathsf{BanachFixedPoint}^{\uparrow}$

The $\mathsf{BanachFixedPoint}^{\uparrow}$ interface names the finite BEDC contraction packet that carries a Banach fixed-point construction as displayed data. Its human reference is Banach's contraction principle, but the certificate records only the rows consumed by BEDC: a Banach carrier row, a complete-metric distance row, a contraction-map row, a rational Lipschitz ledger below the unit bound, a finite Picard ledger, an explicit Cauchy modulus, a regular-rational convergence handoff, a seal-facing real-name endpoint row, transport and continuation rows, package provenance, and a local $\mathsf{NameCert}$ row.

Its dependency list is $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The finite Picard ledger row I is consumed through the $\mathsf{PicardContraction}^{\uparrow}$ carrier of chapter 395, so the fixed-point packet exposes its actual contraction-iteration sibling before any completion handoff is read. The Picard iteration ledger is the concrete contraction-iteration sibling, and the fixed-point packet reaches completion only after consuming chapter 395 together with chapter 128 and the metric carrier.

Definition 396.1 (BanachFixedPoint carrier)**.** *A* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, d, T, \lambda, I, M, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *carrier row whose state histories provide the domain inspected by the fixed-point packet;*
2. *d is the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *distance row used to read finite iterate distances and the carried completeness surface;*
3. *T is the contraction-map row displaying the map-code history and the continuation route from a carried state row to the next Picard state;*
4. *λ is a rational Lipschitz ledger with a displayed positivity row and a displayed unit-bound row;*
5. *I is the finite Picard-iterate ledger, recording $x_0, \ldots, x_n$ together with the application rows for $x_{k+1} = T(x_k)$;*
6. *M is the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row built from the iterate ledger and the Lipschitz-ratio row;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *convergence handoff, exposing the regular rational name and the carried completion route used by consumers;*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing endpoint row attached to the handoff;*
9. *H stores componentwise* hsame *transports for the Banach, distance, contraction, ratio, iterate, modulus, handoff, endpoint, package, and naming rows;*
10. *C stores the displayed* Cont *routes for metric reads, contraction application, finite Picard concatenation, modulus readback, completion handoff, and endpoint sealing;*
11. *P is the* Pkg *provenance row naming the* $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependencies;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, a choice-selected limit, a global nonlinear-analysis theorem, or ambient completeness outside the carried $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Theorem 396.2 (BanachFixedPoint NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$ *obligation surface consists of carrier habitation, componentwise classifier transport, finite Picard-iterate stability, explicit Cauchy-modulus construction from the rational contraction ledger,* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *convergence handoff,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, endpoint determinacy, package dependency stability, and local naming for the finite carrier of Definition 396.1.*

Proof. Project the packet F. Carrier habitation is exactly the displayed tuple of Banach, metric, contraction, ratio, iterate, modulus, handoff, endpoint, transport, continuation, package, and naming rows.

Classifier transport is componentwise. The hsame rows move the displayed histories, while the Cont rows preserve the metric read, contraction application, Picard concatenation, modulus readback, handoff, and endpoint seal.

The Picard, modulus, handoff, uniqueness-boundary, and export obligations are the visible theorem rows of this chapter. Repacking those rows gives the local naming-certificate surface without adding a hidden fixed point or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket namecert obligations`

Theorem 396.3 (BanachFixedPoint iteration ledger)**.** *Every accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier exposes a finite* $\mathsf{PicardContraction}^{\uparrow}$ *ledger I over the carried* $\mathsf{Banach}^{\uparrow}$ *row B, distance row d, contraction row T, and rational Lipschitz ledger λ. For each displayed adjacent pair x_k, x_{k+1} in I, the application row reads x_{k+1} from T and x_k, and the distance comparison is read through d and λ.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket iteration ledger`

Proof. Read the I coordinate of Definition 396.1. It is a finite Picard ledger, so every adjacent pair in the ledger is a visible BHist row rather than a completed orbit.

The contraction-map coordinate T supplies the application route for each step. The distance row d supplies the metric read, and the rational ledger λ supplies the displayed Lipschitz comparison below the unit bound.

Thus the finite iteration surface is exactly the $\mathsf{PicardContraction}^{\uparrow}$ surface carried inside F. No infinite sequence object or host fixed-point value is read from the packet. □

Theorem 396.4 (BanachFixedPoint Cauchy handoff)**.** *For an accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier, the finite Picard ledger I and the rational Lipschitz ledger λ determine the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row M. The convergence handoff R routes this modulus through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row d, and the endpoint seal E exposes only the resulting* $\mathsf{Real}^{\uparrow}$*-facing name boundary.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket cauchy handoff`

Proof. Start with the finite iteration ledger supplied by Definition 396.3. The rational ledger λ is stored with a displayed unit-bound row, so repeated contraction estimates produce a finite threshold for each requested rational tolerance.

That threshold is the visible $\mathsf{CauchyModulus}^{\uparrow}$ coordinate M. It is constructed from displayed iterate distances and the rational ratio ledger, not from a host convergence theorem.

The handoff row R then records the regular rational name and the carried complete-metric route. The endpoint row E is seal-facing: it is consumable by $\mathsf{Real}^{\uparrow}$ readers only after the modulus and handoff rows have been displayed. □

Theorem 396.5 (BanachFixedPoint uniqueness boundary)**.** *If two accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carriers have the same displayed* $\mathsf{Banach}^{\uparrow}$ *row, distance row, contraction row, rational Lipschitz ledger, Cauchy-modulus row, and convergence handoff up to the recorded classifier transports, then their* $\mathsf{Real}^{\uparrow}$ *seal-facing endpoint rows are classifier-determinate. This uniqueness boundary is endpoint determinacy inside the BEDC classifier; it is not a claim of host real equality.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket uniqueness boundary`

Proof. Compare the two packets componentwise through their H and C rows. The shared Banach, distance, contraction, ratio, iterate, modulus, and handoff data force the same route to the seal-facing endpoint row.

The usual Banach uniqueness argument is used only as a finite classifier pattern: the contraction ratio below the unit bound collapses two displayed endpoint routes to the same certificate boundary. The conclusion is therefore a statement about the $\mathsf{Real}^{\uparrow}$ seal row after BEDC transport.

Because the carrier never stores a host real object, the theorem cannot export host equality. It records only determinacy of the endpoint row seen by the local classifier. □

Theorem 396.6 (BanachFixedPoint public export)**.** *Every public consumer of a* $\mathsf{BanachFixedPoint}^{\uparrow}$ *certificate must factor through the carrier rows*

$$B,\ d,\ T,\ \lambda,\ I,\ M,\ R,\ E,\ H,\ C,\ P,\ N.$$

The exported surface provides a finite Picard ledger, an explicit Cauchy modulus, a $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *convergence handoff, and a* $\mathsf{Real}^{\uparrow}$ *seal-facing endpoint row. It does not export host fixed-point equality, choice-selected limits, global nonlinear analysis, or completeness beyond the carried distance row.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket public export`

Proof. Take any public read of the certificate. The carrier definition lists all accessible rows, so the read must pass through the displayed Banach, metric, contraction, ratio, Picard, modulus, handoff, endpoint, transport, continuation, package, and naming coordinates.

Definition 396.3 identifies the finite Picard surface. Definition 396.4 identifies the modulus and convergence handoff. Definition 396.5 identifies the endpoint determinacy available to the classifier.

These rows repack as the public $\mathsf{BanachFixedPoint}^{\uparrow}$ surface. Since no other coordinate is present, the exported certificate cannot be used to read host fixed-point equality, an externally selected limit, global nonlinear-analysis structure, or ambient completeness beyond d. □

Theorem 396.7 (BanachFixedPoint scoped transport route)**.** *Every consumer route through the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier factors through the displayed* BHist *packet rows*

$$B,\ d,\ T,\ \lambda,\ I,\ M,\ R,\ E,\ H,\ C,\ P,\ N.$$

The row H supplies the componentwise hsame *routes for the Banach, distance, contraction, ratio, iterate, modulus, handoff, endpoint, provenance, and naming coordinates. The row C replays the corresponding* Cont *routes for metric reads, contraction application, finite Picard concatenation, modulus readback, completion handoff, and endpoint sealing. The row P records package provenance, and N supplies the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket scoped transport route`

Proof. Project the carrier of Definition 396.1. Its listed coordinates are precisely the rows displayed in the statement, so any consumer route starts by reading one of those coordinates.

The finite Picard surface is identified by Definition 396.3. The modulus and completion-facing handoff are identified by Definition 396.4. The endpoint boundary is identified by Definition 396.5.

Finally, Definition 396.6 says that public access has no route outside the displayed packet. Thus the only available scoped route is through the BHist rows, with H carrying hsame, C carrying Cont, P carrying Pkg provenance, and N carrying the local naming certificate. □

Theorem 396.8 (BanachFixedPoint completion handoff scope)**.** *The handoff from finite Picard data to the seal-facing endpoint of an accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier passes, in order, through the rational Lipschitz ledger λ, the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *distance row d, and only then the* $\mathsf{Real}^{\uparrow}$ *endpoint row E. This scoped handoff does not expose host fixed-point equality, a choice-selected limit, or ambient completeness beyond the carried row d.*

Proof. Begin with the finite Picard ledger from Definition 396.3. Its adjacent application rows are read through T, while all distance comparisons are read through d and the rational Lipschitz ledger λ.

Next apply Definition 396.4. The ratio ledger and finite Picard rows determine the explicit modulus M, and the handoff row R routes that modulus through the regular rational name and the carried complete-metric surface.

The endpoint row E is reached only after this handoff route is present. By Definition 396.6, there is no exported coordinate for host fixed-point equality, selected limits, or an ambient completeness theorem, so the completion handoff remains confined to the displayed BEDC rows. □

Theorem 396.9 (BanachFixedPoint endpoint fixedness)**.** *Let*

$$F = (B, d, T, \lambda, I, M, R, E, H, C, P, N)$$

be an accepted $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier. The endpoint row E is classifier-fixed under the carried contraction row T: reading T at the endpoint route supplied by the completion handoff R, and then sealing the result through the displayed rows R, E, H, C, P, N, returns the same* $\mathsf{Real}^{\uparrow}$ *seal-facing endpoint classifier as E. This is a fixed-point statement internal to the BEDC classifier; it does not assert host fixed-point equality or select a limit outside the displayed packet.*

Proof. Start from the finite contraction surface. By Definition 396.3, every application row in the Picard ledger is read through the same carried contraction row T, with distance comparisons read only through d and λ.

Use Definition 396.4 to pass from the finite ledger to the displayed modulus M and the handoff row R. The route to E is therefore already a route through the contraction ledger, the modulus, and the carried complete-metric surface rather than through an ambient convergence object.

Now compare the endpoint route with the route obtained by one more displayed read of T at that endpoint. The uniqueness boundary of Definition 396.5 identifies these two seal-facing routes inside the classifier, because they share the same B, d, T, λ, M, R data and the recorded componentwise transports.

Finally apply Definition 396.8. The sealing operation can pass only through R, E, H, C, P, N, so the comparison returns the same $\mathsf{Real}^{\uparrow}$ endpoint classifier and cannot be upgraded into host equality, a choice-selected limit, or an external completeness assertion. □

Theorem 396.10 (BanachFixedPoint contractive-orbit handoff)**.** *Every* $\mathsf{BanachFixedPoint}^{\uparrow}$ *endpoint read that consumes a* $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ *packet must factor through the contraction row* T*, the finite Picard ledger* I*, the residual Cauchy-modulus extraction, the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and the* $\mathsf{Real}^{\uparrow}$ *endpoint seal* E *before endpoint fixedness is read. The route is therefore a consumer of the finite orbit packet of chapter 3506, but it exports only the displayed* $\mathsf{BanachFixedPoint}^{\uparrow}$ *rows of Definition 396.1.*

Proof. Start with the carrier

$$F = (B, d, T, \lambda, I, M, R, E, H, C, P, N).$$

The carrier definition makes T, I, M, R, d, and E visible rows, while the $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ chapter supplies the finite orbit-to-modulus packet that a completion consumer may read before the endpoint boundary.

Apply Definition 396.4. The Picard ledger and rational contraction row determine the displayed modulus row M, so the orbit consumer enters the Banach fixed-point route through the same residual Cauchy-modulus extraction rather than through a selected orbit limit.

Next the handoff runs through R and the carried complete-metric row d. Only after that route is displayed can the $\mathsf{Real}^{\uparrow}$ endpoint seal E be read.

Finally apply Definition 396.9. Endpoint fixedness is therefore read after the contraction, Picard, modulus, completion, regular-rational, and real-seal rows have all been consumed, with no access to host fixed-point equality or a selected endpoint outside the packet. □

Theorem 396.11 (BanachFixedPoint Picard modulus consumption)**.** *The threshold row of a* $\mathsf{PicardFixedPointModulus}^{\uparrow}$ *packet consumed by an accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier is consumed exactly as the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *of*

$$F = (B, d, T, \lambda, I, M, R, E, H, C, P, N).$$

Thus the Banach endpoint route cannot replace that finite modulus row by selected limits, host fixed-point equality, quotient maps, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. It may only continue through the displayed handoff rows* R, d, E, H, C, P, N*.*

Proof. Read the $\mathsf{PicardFixedPointModulus}^{\uparrow}$ chapter as the source of the finite threshold packet. Its threshold row is a displayed Picard-tail modulus, not an endpoint selector and not a completed orbit equality.

Project the carrier of Definition 396.1. The only modulus coordinate available to the Banach fixed-point certificate is M, and every later endpoint read must still pass through R, the carried complete-metric row d, and the seal-facing endpoint row E.

Apply Definition 396.4. This identifies M as the Cauchy-modulus row determined by the finite Picard ledger and rational contraction row, matching the threshold consumption supplied by chapter 3442.

Finally apply Definition 396.9. Since endpoint fixedness is sealed only after M, R, d, E, H, C, P, N are displayed, no consumer can replace the threshold row by selected limits, host fixed-point equality, quotient maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 396.12 (BanachFixedPoint contraction Cauchy route)**.** *Every accepted* $\mathsf{BanachFixedPoint}^{\uparrow}$ *packet reads its fixed-point route in the following order. It first reads contraction data through the displayed* $\mathsf{ContractionMapping}^{\uparrow}$ *and* $\mathsf{PicardContraction}^{\uparrow}$ *rows carried by* T *and* I*. It then reads the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row* d*, and only then the* $\mathsf{Real}^{\uparrow}$ *endpoint seal* E*. The route exports no host fixed-point equality, selected limit, quotient-map equality, or ambient completeness outside the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. Start with the finite iteration surface. By Definition 396.3, every adjacent Picard step is read through the carried contraction row T, the finite ledger I, the metric row d, and the rational Lipschitz ledger λ. This is the displayed $\mathsf{ContractionMapping}^{\uparrow}$ and $\mathsf{PicardContraction}^{\uparrow}$ part of the packet, not a host orbit.

Next use Definition 396.4. The finite contraction ledger and the rational unit-bound row determine the explicit modulus M, and the handoff row R routes that modulus through the regular-rational name and the carried complete-metric surface.

Finally apply Definition 396.9. The endpoint fixedness comparison is sealed only after the route has reached R and E, so the result is an internal classifier statement about the $\mathsf{Real}^{\uparrow}$ endpoint boundary.

The refusal clauses are inherited from Definition 396.6 and Definition 396.8. Since the carrier contains no coordinate for host fixed-point equality, selected limits, quotient-map equality, or ambient completeness outside d, a consumer cannot extract any of those readings from the contraction-to-Cauchy route. □

Theorem 396.13 (BanachFixedPoint bridge packet scope)**.** *Any standard bridge for* $\mathsf{BanachFixedPoint}^{\uparrow}$ *must factor through the displayed packet rows*

$$B,\ d,\ T,\ \lambda,\ I,\ M,\ R,\ E,\ H,\ C,\ P,\ N,$$

the public export surface of Definition 396.6, and the completion handoff scope of Definition 396.8. Such a bridge may transport the finite Picard ledger, the explicit Cauchy modulus, the regular-rational completion handoff, and the seal-facing endpoint classifier, but it may not read host fixed-point equality, choice-selected limits, global nonlinear analysis, or ambient completeness outside the carried $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. Begin with the carrier projection from Definition 396.1. The displayed rows are the only accessible coordinates of an accepted packet.

Apply Definition 396.3 to identify the finite Picard route. Apply Definition 396.4 to route that ledger through the explicit modulus M, the regular-rational handoff R, and the carried complete-metric row d.

Next apply Definition 396.6 and Definition 396.7. Every public consumer and every transported route must pass through the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally apply Definition 396.8 and Definition 396.9. The endpoint comparison is therefore sealed inside R, E, H, C, P, N, so a standard bridge has no coordinate from which to extract host fixed-point equality, a choice-selected limit, or ambient completeness beyond the carried row d. □

Theorem 396.14 (BanachFixedPoint standard bridge)**.** *The displayed packet rows*

$$B,\ d,\ T,\ \lambda,\ I,\ M,\ R,\ E,\ H,\ C,\ P,\ N$$

provide the paper standard bridge from the $\mathsf{BanachFixedPoint}^{\uparrow}$ *certificate to the ordinary contraction-principle reading. The bridge preserves exactly the finite Picard ledger, explicit rational contraction modulus, completion handoff, seal-facing endpoint classifier, componentwise transport, replay, provenance, and local naming rows. It preserves no host fixed-point equality, choice-selected limit, quotient-map equality, global nonlinear-analysis structure, or ambient completeness beyond the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. Begin with the carrier projection from Definition 396.1. The rows $B, d, T, \lambda, I, M, R, E, H, C, P, N$ are the whole readable packet surface.

Apply Definition 396.13. Any standard bridge must factor through these rows, the public export surface, and the completion handoff scope.

The ordinary contraction-principle reading is therefore obtained only by reading the finite Picard ledger I, the rational contraction row λ, the explicit modulus M, the completion handoff R, and the endpoint classifier E. The structural rows H, C, P, N transport, replay, source, and name the same data.

The exclusions follow from the same packet scope. Since the bridge has no coordinate for host fixed-point equality, choice-selected limits, quotient-map equality, global nonlinear-analysis structure, or ambient completeness beyond d, none of those readings is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPoint}^{\uparrow}$ packet is generated from a $\mathsf{Banach}^{\uparrow}$ carrier row, a $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ distance row, a contraction-map row, a rational Lipschitz ledger below the unit bound, a finite $\mathsf{PicardContraction}^{\uparrow}$ ledger, an explicit $\mathsf{CauchyModulus}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ convergence handoff, a $\mathsf{Real}^{\uparrow}$ seal-facing endpoint row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 396.7 binds every consumer route to the displayed BHist, hsame, Cont, Pkg, and NameCert rows, while Definition 396.14 supplies the paper standard bridge to the ordinary contraction-principle reading.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BanachFixedPoint}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BanachFixedPointUp.BanachFixedPointPacket\_public\_export`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host fixed-point equality, choice-selected limits, global nonlinear analysis, ambient completeness beyond the carried $\mathsf{CompleteMetric}^{\uparrow}$ row, or bridge checking.

Upgrade path: No further theory-axis upgrade is pending for the displayed BanachFixedPoint packet.

Chapter 397

A Concrete Naming Certificate for CompletionUniversalProperty$^{\uparrow}$

The CompletionUniversalProperty$^{\uparrow}$ interface records the finite universal-property surface for the constructive real-completion phase. Following Bishop and Bridges, *Constructive Analysis*, where completion is handled through explicit Cauchy data and extension control rather than by selecting host limits, the packet names the data needed to extend uniformly continuous maps out of the regular rational completion envelope. It is a certificate over CauchyCompletionMonad$^{\uparrow}$, RealDiagonalCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ContinuousMap$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

The universal-property packet consumes the unit-and-bind schedule of chapter 378, the diagonal finite-window route of chapter 359, the regular rational source of chapter 301, the real seal of chapter 38, and the dense-map row of chapter 130. These sibling rows supply the source, diagonal, dense, seal, and map coordinates M, D, R, A, F of Definition 397.1; the local chapter packages extension and uniqueness ledgers over that displayed route.

The extension ledger is the downstream consumer of the Cauchy-name and completion-envelope surfaces: chapter 406 supplies the finite common-window presentation, and chapter 378 supplies the completion-envelope source. This is a substantive dependency row for the extension and uniqueness ledgers, not background prose.

Definition 397.1 (CompletionUniversalProperty carrier)**.** *A* CompletionUniversalProperty$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, D, R, A, F, E, L, P, N)$$

with the following visible rows:

1. *M is the* CauchyCompletionMonad$^{\uparrow}$ *completion-envelope row that supplies the completion-facing source packet;*
2. *D is the* RealDiagonalCompletion$^{\uparrow}$ *diagonal packet that reads regular rational Cauchy data through finite diagonal windows;*
3. *R is the* RegSeqRat$^{\uparrow}$ *regular rational input row whose stream windows provide the dense source observations;*
4. *A is the* Real$^{\uparrow}$ *real-name seal row through which the completed source is inspected;*
5. *F is the* ContinuousMap$^{\uparrow}$ *row on the dense regular rational surface, including its displayed modulus and graph-read ledger;*

6. *E is the extension-existence ledger, a finite* Cont *route from F, R, D, and A to a completion-facing* $\mathsf{ContinuousMap}^{\uparrow}$ *read;*
7. *L is the uniqueness ledger, a displayed* hsame *comparison between two extension rows whenever they agree on the same R, D, A, F, and E coordinates;*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{ContinuousMap}^{\uparrow}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionUniversalProperty}^{\uparrow}}$ *row.*

The carrier contains no quotient of regular streams, no selected host limit, no host-level complete metric space, and no equality principle outside the displayed hsame *and* Cont *ledgers.*

Lean4 ✓ `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyTasteGate single carrier alignment`

Theorem 397.2 (CompletionUniversalProperty extension obligations)**.** *Let $U = (M, D, R, A, F, E, L, P, N)$ be an accepted* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carrier. Every extension read exported by U factors through the completion-envelope row M, the diagonal-completion row D, the regular rational source row R, the* $\mathsf{Real}^{\uparrow}$ *seal A, the dense* $\mathsf{ContinuousMap}^{\uparrow}$ *row F, and the finite continuation ledger E.*

Consequently the extension obligation is finite: for each displayed precision request, the consumer reads a diagonal packet from D, a regular rational observation from R, a real-name seal from A, and a modulus-controlled graph row from F, then repacks them through E as a completion-facing continuous-map row. No quotient representative, host limit, or ambient completeness theorem is introduced by the extension read.

Lean4 ✓ `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyPacket extension obligations`

Proof. Project the accepted carrier. The completion source is the row M, and the only diagonal access to its real-name observations is the displayed row D.

The dense map data are exactly the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{ContinuousMap}^{\uparrow}$ row F. For a requested precision, F supplies the graph-read and modulus ledger on the dense surface, while D and A supply the completion-facing observation route.

The extension row E is the only coordinate that repacks those observations as a completion-facing continuous map. Since the carrier has no coordinate for a quotient of streams, a selected host limit, or a host complete space, the exported extension remains a finite BEDC packet. □

Theorem 397.3 (CompletionUniversalProperty uniqueness ledger)**.** *Let $U = (M, D, R, A, F, E, L, P, N)$ and $U' = (M, D, R, A, F, E', L', P, N')$ be accepted* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carriers with the same completion envelope, diagonal row, dense regular rational row, real seal, dense continuous-map row, and package provenance. If the extension ledgers E and E' agree on every displayed finite diagonal packet read from D and every displayed regular rational source read from R, then the two exported extension rows are compared by the displayed* hsame *ledger L and L'.*

The uniqueness claim is therefore a ledger statement over visible rows. It identifies extensions only by the carried hsame *comparison generated from their finite dense-source and diagonal reads, not by function extensionality over a host completion.*

Lean4 ✓ `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyPacket uniqueness ledger`

Proof. Read the common coordinates M, D, R, A, F, and P from the two packets. These rows fix the completion envelope, the dense-source observations, and the continuous-map data available to both extension ledgers.

The hypothesis says that E and E' agree on the displayed finite reads supplied by D and R. Transport those reads through the shared Cont routes in F and through the common real-name seal A.

The uniqueness coordinate is precisely the displayed hsame ledger. Applying L and L' to the transported extension rows yields the required comparison, with no appeal to host function equality or an external completion object. □

Theorem 397.4 (CompletionUniversalProperty NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompletionUniversalProperty}}$ *surface consists of five finite rows:*

1. *carrier: an accepted object is exactly the finite* BHist *packet of Definition 397.1;*
2. *classifier: two accepted packets classify together only by componentwise* hsame *transport of M, D, R, A, F, E, L, P, and N, and by the displayed* Cont *routes carried by the dense and extension map rows;*
3. *extension existence over finite diagonal packets: Definition 397.2 supplies the extension row from the displayed* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads;*
4. *uniqueness by displayed ledger: Definition 397.3 compares any two extension rows only through the carried* hsame *uniqueness ledger;*
5. *non-escape: every consumer read remains inside* $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyPacket namecert obligations`

Proof. Carrier admission and classification are projections from Definition 397.1. The classifier has no fields beyond the displayed packet coordinates, their hsame transports, and the Cont ledgers attached to the dense and extension map rows.

Extension existence is the obligation of Definition 397.2. It constructs the public extension read from finite diagonal-completion packets and regular rational observations.

Uniqueness is the ledger obligation of Definition 397.3. The comparison is carried by the displayed hsame row, so it does not require equality of host functions.

The non-escape row follows by inspecting the carrier. There is no coordinate for quotient streams, choice-selected limits, host completeness, or host real equality; every exported object is one of the listed BEDC rows or a finite route between them. □

Theorem 397.5 (CompletionUniversalProperty extension-uniqueness scope)**.** *Let* $U = (M, D, R, A, F, E, L, P, N)$ *be an accepted* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carrier over* $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The packet combines extension existence and uniqueness ledger stability into one* NameCert*-consumable* BHist *row: the extension ledger E reads the dense* $\mathsf{ContinuousMap}^{\uparrow}$ *row through the completion envelope, diagonal row, regular rational source, and real seal, while the uniqueness ledger L compares any two such extension rows by displayed* hsame *transport over the same provenance.*

Consequently the completion universal-property surface exports a single finite packet for extension existence, uniqueness, and non-escape. It does not export quotient streams, selected host limits, host complete spaces, or function extensionality outside the displayed hsame *and* Cont *rows.*

Proof. First construct the extension component. Apply Definition 397.2: every extension read factors through M, D, R, A, F, and the finite continuation ledger E.

Next bind uniqueness to the same displayed rows. Apply Definition 397.3: two extension rows agreeing on the finite diagonal and regular rational reads are compared by the carried hsame uniqueness ledger.

Finally repack the two components as a local naming-certificate surface. Apply Definition 397.4, which lists carrier admission, classifier transport, extension existence, uniqueness by displayed ledger, and non-escape over exactly the rows of U.

The resulting packet is therefore a single BHist scope for the completion universal property. Since every row is displayed in the carrier, the construction has no route to quotient streams, selected host limits, host complete spaces, or host-level function equality. □

Theorem 397.6 (CompletionUniversalProperty classifier transport)**.** *Let*

$$U = (M, D, R, A, F, E, L, P, N)$$

and

$$U' = (M', D', R', A', F', E', L', P', N')$$

be accepted $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carriers. Suppose the classifier records componentwise comparisons*

$$\mathsf{hsame}(M, M'), \quad \mathsf{hsame}(D, D'), \quad \mathsf{hsame}(R, R'), \quad \mathsf{hsame}(A, A'), \quad \mathsf{hsame}(F, F'),$$
$$\mathsf{hsame}(E, E'), \quad \mathsf{hsame}(L, L'), \quad \mathsf{hsame}(P, P'), \quad \mathsf{hsame}(N, N')$$

and the displayed Cont *routes on the completion envelope, diagonal packet, regular rational source, real seal, dense map, extension ledger, uniqueness ledger, package provenance, and local naming rows. Then accepted transport preserves the carrier of Definition 397.1 and preserves the* $\mathsf{NameCert}_{\mathsf{CompletionUniversalProperty}^{\uparrow}}$ *classifier. No transported row leaves the nine displayed coordinates.*

Proof. Project both accepted packets through Definition 397.1. The carrier boundary is exactly the ordered tuple of nine rows, and each row named in the transport hypothesis is one of those coordinates.

The componentwise hsame comparisons preserve the displayed row identities. The recorded Cont routes only say where the same rows are read inside the finite packet, so they do not add a new completion object, quotient stream, or host equality principle.

Therefore the classifier after transport is the same local classifier with primed coordinates. Since every transported component is still one of M', D', R', A', F', E', L', P', and N', the carrier and classifier are preserved on the accepted $\mathsf{CompletionUniversalProperty}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyPacket classifier transport`

Theorem 397.7 (CompletionUniversalProperty dense-source factorization)**.** *Let* $U = (M, D, R, A, F, E, L, P, N)$ *be an accepted* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carrier. Every exported extension read factors through the completion envelope* M*, the finite diagonal packet* D*, the dense regular-rational source* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* A*, the dense* $\mathsf{ContinuousMap}^{\uparrow}$ *row* F*, and the extension ledger* E*. In this factorization,* R *is the dense regular-rational source and* D *is the finite diagonal packet through which completion-facing reads are admitted.*

Proof. Apply Definition 397.2. It already identifies the six coordinates through which every extension read passes: M, D, R, A, F, and E.

It remains only to name the roles of the two source coordinates. By Definition 397.1, R is the regular rational input row whose finite stream windows provide dense source observations, while D is the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ row reading regular rational Cauchy data through finite diagonal windows.

Composing these coordinate projections with the extension obligation gives the stated factorization. No other carrier row can serve as a dense source or diagonal admission route, because the exported extension is repacked only by the finite continuation ledger E. □

Theorem 397.8 (CompletionUniversalProperty dense-source exhaustion)**.** *Let*

$$U = (M, D, R, A, F, E, L, P, N)$$

be an accepted $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *carrier, and let*

$$C = (S, W, R_0, T, E_0, Q, H, C_0, P_0, N_0)$$

be an accepted $\mathsf{CompletionDensity}^{\uparrow}$ *packet whose source rows display the same regular-rational source and terminal* $\mathsf{Real}^{\uparrow}$ *seal as the* R, A *coordinates of* U*. Every extension read exported by* U *that uses the dense source must first pass through the* $\mathsf{CompletionDensity}^{\uparrow}$ *route*

$$S, W, R_0, T, E_0, Q, H, C_0, P_0, N_0,$$

including the $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *row* Q*, before it reaches the diagonal packet* D*, the* $\mathsf{Real}^{\uparrow}$ *seal* A*, the dense map row* F*, and the extension ledger* E *of* U*.*

Thus the universal-property dense source is exhausted by the same finite density route that exposes source names, finite windows, regular rational readbacks, dyadic tolerances, completion embedding, regular Cauchy dense embedding, transport, replay, provenance, and local naming. No extension read may substitute a detached dense subset, quotient completion, selected host limit, countable-choice schedule, host real equality, or ambient completeness principle for that route.

Proof. First bind the dense source to the density packet. By Definition 1560.3, every regular Cauchy density read of the source image factors through $S, W, R_0, T, E_0, Q, H, C_0, P_0, N_0$; by Definition 1560.4, every Real-facing completion read using that dense image consumes the same route before the terminal real-name seal is reached.

Next identify the Real boundary inside the dense-image row. Apply Definition 1453.3 to the row Q: its exported $\mathsf{Real}^{\uparrow}$ read must pass through the displayed regular source, stream schedule, and dyadic tolerance ledger before any Real seal is available.

Finally compose this density route with the universal-property packet. The extension obligation of Definition 397.2 routes every exported extension through M, D, R, A, F, E, while Definition 397.7 identifies R as the dense regular-rational source and D as the diagonal admission packet.

Since the accepted density packet displays the same regular-rational source and terminal Real seal as R, A, the dense-source portion of any extension read must already have passed through $S, W, R_0, T, E_0, Q, H, C_0, P_0, N_0$. The carrier rows of U then carry that exhausted source into D, A, F, E. Neither packet contains a coordinate for a detached dense subset, quotient completion, selected host limit, countable choice, host real equality, or ambient completeness, so none can be exported by the composed route. □

Theorem 397.9 (CompletionUniversalProperty extension non-escape)**.** *For any accepted* CompletionUniversalProp *carrier, the extension-plus-uniqueness packet exposes no quotient stream, selected host limit, host complete metric space, or function extensionality outside the displayed* hsame *and* Cont *rows. Its visible data are exactly the completion envelope, diagonal packet, dense source, real seal, dense continuous-map row, extension ledger, uniqueness ledger, package provenance, and local naming row.*

Proof. The extension side is controlled by Definition 397.2. Each exported extension read is routed through M, D, R, A, F, and E, all of which are displayed coordinates of the accepted carrier.

The uniqueness side is controlled by Definition 397.3. Two extension rows are compared only by the carried hsame ledger over the same finite dense-source and diagonal reads.

Finally, Definition 397.4 lists the local certificate rows and its non-escape boundary. Since that list contains no quotient stream, host-selected limit, host complete metric space, or function-extensionality row, the extension-plus-uniqueness packet cannot export any of them outside the displayed hsame and Cont routes. □

Theorem 397.10 (CompletionUniversalProperty downstream consumer use)**.** *Any downstream completion consumer that uses the extension-plus-uniqueness interface of an accepted* CompletionUniversalProperty$^{\uparrow}$ *carrier factors through the same packet*

$$U = (M, D, R, A, F, E, L, P, N).$$

Its extension reads consume the completion envelope M, diagonal packet D, regular rational source R, real seal A, dense continuous-map row F, and extension ledger E; its uniqueness reads consume the displayed ledger L over the same coordinates; its classifier transport and dense-source reads are those of Definition 397.6 and Definition 397.7; and its boundary is the non-escape surface of Definition 397.9.

Proof. Start with the consumer's extension request. By Definition 397.2, the request factors through M, D, R, A, F, and E, so the consumer sees the same finite extension packet.

For uniqueness, apply Definition 397.3. Any comparison of two extension rows is read through the displayed hsame ledger L over the same completion envelope, diagonal packet, dense source, real seal, dense map, and provenance rows.

The classifier and dense-source uses are fixed by Definition 397.6 and Definition 397.7. They admit no alternate carrier because transport and dense reads remain inside the listed coordinates.

Finally, Definition 397.9 supplies the consumer boundary. The downstream consumer therefore uses the single finite packet U and no quotient stream, selected host limit, host complete metric space, or function-extensionality row. □

Closure status for NameCert$_{\text{CompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A CompletionUniversalProperty$^{\uparrow}$ packet is generated by a completion-envelope row, a diagonal-completion row, a regular rational source row, a real-name seal, a dense continuous-map row, a finite extension ledger, a uniqueness hsame ledger, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\text{CompletionUniversalProperty}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 397.1, extension existence in Definition 397.2, uniqueness ledger control in Definition 397.3, classifier transport in Definition 397.6, dense-source factorization in Definition 397.7, non-escape in Definition 397.9, and downstream consumer use in Definition 397.10 bind the finite completion route through RealCauchyCompletion$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CompletionUniversalProperty}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CompletionUniversalPropertyUp.CompletionUniversalPropertyPacket\_classifier\_transport`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient construction of real numbers, host completeness, function extensionality over completed spaces, or bridge checking.

Upgrade path: No theory-axis upgrade remains; bridge checking is a formal-axis matter.

Chapter 398

A Concrete Naming Certificate for $\mathsf{BoundedVariation}^{\uparrow}$

The $\mathsf{BoundedVariation}^{\uparrow}$ interface names the finite partition-variation packet used by real-analysis consumers of $\mathsf{Real}^{\uparrow}$ data. It follows the Jordan bounded-variation surface treated in Rudin, *Principles of Mathematical Analysis*, and Folland, *Real Analysis*, but expresses every read as a finite partition ledger over regular rational names, dyadic observations, endpoint transports, and package provenance. The packet does not introduce a completed function space, a supremum over all partitions, a variation measure, or a host real-valued function quotient.

Its dependency list is $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and interval-partition rows supplied by $\mathsf{RationalInterval}^{\uparrow}$ or $\mathsf{NestedInterval}^{\uparrow}$ when such endpoints are available.

The finite partition-variation ledger consumes the real-name seal of chapter 38, the regular rational stream surface of chapter 301, and the dyadic observation rows of chapter 291.

Definition 398.1 (BoundedVariation carrier)**.** *A* $\mathsf{BoundedVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, \Pi, X, D, V, R, H, C, P, N)$$

with the following visible coordinates:

1. *I is the rational or nested interval row on which the variation packet is read;*
2. *Π is a finite ordered interval-partition ledger with displayed endpoint rows and common-refinement rows;*
3. *X records the* $\mathsf{Real}^{\uparrow}$ *endpoint values or their* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name representatives at the partition points;*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows used to compare endpoint differences on each partition edge;*
5. *V is the finite dyadic variation-sum ledger attached to the displayed partition;*
6. *R records common-refinement comparison rows for transported partitions and endpoint reads;*
7. *H, C, P, and N are respectively the visible* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance rows.*

The classifier compares accepted packets only through common refinements of Π, transported endpoint values, dyadic variation-sum ledgers, endpoint-difference comparisons, and the displayed provenance rows.

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationTasteGate single carrier alignment`

Theorem 398.2 (BoundedVariation NameCert obligations)**.** *Every accepted* BoundedVariation$^{\uparrow}$ *packet with carrier Definition 398.1 carries the following naming-certificate obligations:*

1. *carrier habitation by a constant* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *endpoint row, with zero dyadic variation sum on every displayed finite partition;*
2. *classifier comparison by common refinement of the finite partition ledgers and componentwise endpoint transport;*
3. *dyadic variation-sum exactness for each displayed partition edge, read through* DyadicRatCore$^{\uparrow}$ *observations and* RegSeqRat$^{\uparrow}$ *endpoint comparisons;*
4. *carrier transport under displayed* hsame *and* Cont *routes for interval rows, partitions, endpoint names, dyadic observations, variation sums, and refinement rows;*
5. *boundary exclusion of completed function spaces, variation measures, host supremum principles, quotient functions, and hidden partitions.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier namecert obligations`

Proof. For habitation, take a constant real-name or regular-rational endpoint row. Every edge of a displayed finite partition has equal endpoint reads, so the dyadic edge differences and their finite sum are the zero variation ledger.

Classifier comparison is by common refinement. Refine two displayed partition ledgers to a shared finite ledger, transport the endpoint rows componentwise, and compare the dyadic edge observations along that shared list.

Variation-sum exactness is read edge by edge from the carrier. The DyadicRatCore$^{\uparrow}$ observations give the displayed endpoint-difference rows, and the finite sum row V records exactly those edge rows and no unlisted partition.

Transport uses the stored hsame and Cont coordinates. Moving the interval, partition, endpoint, dyadic, sum, and refinement rows along those routes repacks the same finite packet with the same local NameCert provenance.

The boundary exclusion follows because the carrier has no coordinate for a completed function space, variation measure, host supremum over all partitions, quotient of functions, or hidden partition family. □

Theorem 398.3 (BoundedVariation variation-ledger exactness)**.** *For every accepted* BoundedVariation$^{\uparrow}$ *packet and every displayed finite partition row* Π*, the public variation read factors through the interval row, the ordered endpoint ledger, the* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *endpoint names, the* DyadicRatCore$^{\uparrow}$ *edge-difference observations, and the finite dyadic variation-sum row* V*. A common-refinement read transports the same edge data and repacks the same finite sum ledger.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier variation ledger exactness`

Proof. Start from the displayed partition row Π. Its ordered endpoint ledger gives the only edges whose variation contribution may be read by the packet.

For each edge, read the two endpoint names through X and the dyadic comparison row through D. The regular-name and real-seal data are therefore consumed before the finite variation sum is formed.

The row V stores the finite sum of exactly those displayed edge observations. When a common refinement is supplied by R, the endpoint transports and dyadic edge rows are repacked along the refinement ledger, so the public read remains a finite partition-sum read. □

Theorem 398.4 (BoundedVariation common-refinement classifier exactness)**.** *Let B and B' be accepted* BoundedVariation$^{\uparrow}$ *packets with carrier coordinates as in Definition 398.1. The local* BoundedVariation$^{\uparrow}$ *classifier identifies B and B' if and only if there is a displayed common finite refinement whose interval row, transported endpoint* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *names,* DyadicRatCore$^{\uparrow}$ *edge observations, finite variation-sum ledger, and* Pkg *and* NameCert *provenance rows compare componentwise by the visible* hsame *and* Cont *routes stored in the two packets.*

Proof. Unfold the carrier of Definition 398.1. The classifier has no hidden coordinate beyond the interval row, finite partition ledger, endpoint names, dyadic edge observations, variation-sum ledger, common-refinement rows, and displayed hsame, Cont, Pkg, and NameCert provenance.

For the forward direction, project the classifier witness to its common-refinement rows. Apply Definition 398.3 to both packets on that refinement. Both public variation reads therefore factor through the same displayed endpoint ledger, dyadic edge observations, and finite variation-sum row, with the endpoint and provenance coordinates transported by the recorded hsame and Cont routes.

For the reverse direction, start from the displayed common finite refinement and its componentwise comparisons. Repack the transported interval, endpoint, edge-observation, variation-sum, Pkg, and NameCert rows into the local classifier data. Since the exactness theorem accounts for every public variation read through that finite ledger, no further comparison data are required. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier common refinement classifier exactness`

Theorem 398.5 (BoundedVariation carrier transport)**.** *Let B and B' be accepted* BoundedVariation$^{\uparrow}$ *packets. If their interval rows, finite partition ledgers, endpoint-name rows, dyadic observation rows, variation-sum ledgers, and common-refinement rows are transported componentwise by displayed* hsame *routes and read through the corresponding* Cont *routes, then the local classifier identifies the same bounded-variation packet on every common finite refinement.*

Proof. Unfold the classifier of Definition 398.1. It compares only the interval row, partition ledger, endpoint names, dyadic observations, finite variation sums, refinement rows, and provenance.

Move each of those coordinates along the displayed hsame routes. The Cont routes preserve the read order from an interval request to a partition, then to endpoint names, edge observations, and the finite sum row.

Apply Definition 398.3 on the common refinement. The transported endpoint and dyadic rows repack the same finite variation ledger, leaving no extra classifier coordinate on which the packets could differ. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier transport`

Theorem 398.6 (BoundedVariation no-escape boundary)**.** *The* BoundedVariation$^{\uparrow}$ *public boundary exports only finite interval-partition rows, transported endpoint names, dyadic edge-difference observations, common-refinement ledgers, finite variation sums, and visible* hsame, Cont, Pkg, *and* NameCert *provenance. It exports no completed function-space carrier, variation measure, supremum over all partitions, quotient of real-valued functions, or hidden partition family.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier public rows zero head absurd`

Proof. Inspect the carrier coordinates of Definition 398.1. The only analysis-facing data are the displayed interval row, the finite partition ledger, endpoint names, dyadic observations, the finite sum row, and common-refinement rows.

The exactness theorem Definition 398.3 shows that every public variation read factors through those rows. The transport theorem Definition 398.5 shows that moving the packet preserves the same finite ledger.

No coordinate names a completed function space, variation measure, host supremum over all partitions, quotient function, or hidden partition family. Hence none of those objects is exported by the public boundary. □

Theorem 398.7 (BoundedVariation partition scope binding)**.** *The carrier, NameCert obligation surface, and variation-ledger exactness rows of Definition 398.1, Definition 398.2, and Definition 398.3 bind the finite partition-variation packet to* DyadicRatCore$^{\uparrow}$, NestedInterval$^{\uparrow}$, RationalInterval$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The bound is a finite partition-sum scope: every variation read passes through the displayed interval row, partition ledger, endpoint rows, dyadic edge observations, finite variation sum, and local provenance before it can be consumed by a bounded-variation classifier.

Proof. Start with the carrier in Definition 398.1. It names the interval row, finite partition ledger, endpoint rows, dyadic observations, finite variation sum, common-refinement rows, and provenance rows.

Apply Definition 398.2. The local certificate records habitation, classifier comparison, variation-sum exactness, carrier transport, and boundary exclusion for those same coordinates.

Finally use Definition 398.3. Each public variation read factors through a displayed finite partition and its dyadic edge ledger, so the partition-sum scope is exactly the listed dependency stack. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier partition scope binding`

Theorem 398.8 (BoundedVariation refinement ledger scope)**.** *The common-refinement classifier exactness row of Definition 398.4, together with the variation-ledger exactness row of Definition 398.3, gives the scoped refinement ledger for* BoundedVariation$^{\uparrow}$. *Accepted packets compare only by a displayed common finite refinement whose interval, endpoint, dyadic observation, finite sum,* BHist, hsame, Cont, Pkg, *and* NameCert *rows are visible over* DyadicRatCore$^{\uparrow}$, NestedInterval$^{\uparrow}$, RationalInterval$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* StreamName$^{\uparrow}$.

No classifier comparison in this scope uses a hidden partition family, host supremum, variation measure, quotient function, or completed function-space carrier.

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger refinement ledger scope`

Proof. Project the classifier through Definition 398.4. The comparison data are a displayed common finite refinement with componentwise endpoint, dyadic, finite-sum, package, and naming rows.

Use Definition 398.3 on that refinement. Both public variation reads are finite sums over the displayed edge observations and transported endpoint rows.

Thus the refinement ledger is already the classifier scope. Since the carrier has no hidden partition family or host supremum coordinate, no additional comparison object can enter. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger refinement scope semantic name certificate`

Theorem 398.9 (BoundedVariation finite Jordan-split scope)**.** *For an accepted* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, a displayed finite partition may be split into positive-edge and negative-edge dyadic variation ledgers using only the listed endpoint rows. The split reads the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint names, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-difference observations, the finite partition ledger, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance.*

This is only a finite displayed partition split. It does not assert Jordan decomposition, a variation measure, a host supremum over all partitions, or a completed function-space object.

Proof. Project the finite partition ledger and endpoint rows from the carrier. Each edge already carries two endpoint reads and a displayed dyadic difference observation.

Split the finite list of edge observations by the sign recorded in the $\mathsf{DyadicRatCore}^{\uparrow}$ row. The positive-edge and negative-edge ledgers are therefore subledgers of the displayed partition, transported by the same hsame and Cont routes as the original variation sum.

Repack the two ledgers with the existing package and naming provenance. Since the construction uses only the displayed finite partition, it does not create a Jordan decomposition or a measure-valued object. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger jordan finite split scope`

Theorem 398.10 (BoundedVariation Jordan consumer exhaustion)**.** *Every Jordan-facing consumer of an accepted* $\mathsf{BoundedVariation}^{\uparrow}$ *packet must read finite partition rows, endpoint* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *names,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, finite variation sums, common-refinement rows, and visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance before any Jordan split is exported.*

The exported split is exhausted by the displayed finite partition ledgers. It contains no variation measure, host supremum over all partitions, completed function space, quotient real-valued function, hidden partition family, selected decomposition, or ambient Jordan theorem.

Proof. Start with the finite Jordan-split scope of Definition 398.9. Any positive-edge or negative-edge ledger is built from the displayed finite partition, its endpoint names, and its dyadic edge observations.

Use refinement additivity, Definition 398.11. When a Jordan-facing consumer compares two displayed packets, the common-refinement rows force every block sum to remain inside the same finite dyadic variation ledger.

Now apply the no-escape boundary, Definition 398.6. The public boundary exports only finite partition rows, transported endpoints, dyadic edge observations, common refinements, finite sums, and visible provenance.

Therefore the consumer has exhausted the available Jordan-facing data once it has read those rows. No carrier coordinate remains from which to export a variation measure, host supremum, completed function space, quotient function, hidden partition family, selected decomposition, or ambient Jordan theorem. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger jordan consumer exhaustion`

Theorem 398.11 (BoundedVariation refinement additivity scope)**.** *For two accepted* $\mathsf{BoundedVariation}^{\uparrow}$ *packets compared on a displayed common finite refinement, the variation read is additive over the refinement blocks in the finite dyadic ledger. Each block sum uses only* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, the common-refinement row, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance.*

The additivity claim is restricted to the displayed common refinement and exports no supremum principle, variation measure, hidden partition family, quotient function, or completed function space.

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger refinement additivity scope`

Proof. Start with the refinement ledger scope. It supplies one displayed common finite refinement whose endpoint and dyadic edge rows are visible to both packets.

Use variation-ledger exactness on each block of that common refinement. The finite dyadic sum for a block is obtained by summing exactly the edge observations in that block, with endpoint transports carried by the displayed hsame and Cont rows.

Concatenate the finite block sums inside the same common-refinement ledger. The package and naming rows remain unchanged, and the statement never ranges over all partitions or any completed function object. □

Theorem 398.12 (BoundedVariation regular Cauchy filter handoff)**.** *Let L be an accepted* $\mathsf{BoundedVariation}^{\uparrow}$ *partition ledger whose endpoint reads are transported along a directed* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *tail basis. Then every variation read exposed from L factors through the finite interval row, the ordered endpoint ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, the displayed common-refinement row, and the directed* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *meet before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The handoff exports only the displayed finite-variation ledger together with the regular Cauchy filter tail route. It exports no host supremum over partitions, variation measure, quotient function, selected limit, hidden cofinal tail, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger regular cauchy filter handoff`

Proof. Project the ledger through Definition 398.16. The interval row, ordered endpoint list, dyadic edge observations, finite sum row, and common-refinement row are the only coordinates available to a variation consumer.

Use Definition 398.3 and Definition 398.11. They force each displayed variation read to be assembled from the same finite edge observations and then repacked along the displayed common-refinement ledger.

Now apply the $\mathsf{RegularCauchyFilter}^{\uparrow}$ basis factorization of Definition 1445.5. The transported endpoint reads pass through the directed tail windows and their meet row before the regular-rational readback or Real seal can be consumed.

Finally use Definition 398.6. Since the variation side has only finite interval, endpoint, dyadic, sum, refinement, transport, replay, package, and naming rows, and the filter side contributes only the displayed directed meet, no absent analytic or quotient coordinate can enter the handoff. □

Theorem 398.13 (BoundedVariation partition-concatenation ledger)**.** *Let L_0 and L_1 be displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *partition ledgers in the sense of Definition 398.16. Suppose the final endpoint row of L_0 and the initial endpoint row of L_1 are related by a displayed transported endpoint row over the same* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *name, with matching* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance. Then the concatenated ordered partition row is again a displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *partition ledger, and its finite dyadic variation-sum row is the concatenation of the visible edge-observation sums of L_0 and L_1.*

This ledger composition is restricted to the displayed finite partitions and their transported shared endpoint. It does not assert Jordan decomposition, variation measures, host suprema, quotient function equality, completed function spaces, or any hidden partition family.

Proof. Project L_0 and L_1 through Definition 398.16. Each ledger exposes an interval row, an ordered finite partition, adjacent-edge rows, endpoint-name rows, dyadic edge observations, a finite variation sum, refinement rows, transport, continuation, package provenance, and local naming data.

Use the transported shared endpoint to identify the terminal endpoint row of L_0 with the initial endpoint row of L_1 inside the displayed hsame and Cont surface. This supplies the only splice row needed to concatenate the two ordered partition ledgers without adding an unrecorded endpoint.

Apply Definition 398.11 to read the two edge lists as finite dyadic blocks, and apply Definition 398.15 to keep the shared endpoint and the adjacent edge observations on the displayed dependency stack. The concatenated variation row is therefore the finite concatenation of the two visible edge-observation sums.

Finally use Definition 398.6. The construction has used only displayed partition, endpoint, dyadic, transport, continuation, package, and naming rows, so the resulting ledger cannot export a host supremum, variation measure, quotient function, completed function-space carrier, or hidden partition family. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger concatenation ledger`

Theorem 398.14 (BoundedVariation finite-variation-ledger consumer)**.** *Every accepted* BoundedVariation$^{\uparrow}$ *partition read factors through the* FiniteVariationLedger$^{\uparrow}$ *carrier rows before a bounded-variation consumer reads the variation value. The required route exposes the interval row, displayed finite partition, endpoint* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *names,* DyadicRatCore$^{\uparrow}$ *edge observations, finite variation sum, common-refinement row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Thus the bounded-variation packet consumes the reusable finite ledger of Definition 6254.1. It exports no host supremum over all partitions, variation measure, quotient function equality, completed function-space carrier, hidden partition family, or ambient Real$^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted packet through Definition 398.1 and the partition ledger through Definition 398.16. The visible rows are exactly the interval, displayed finite partition, endpoint names, dyadic edge observations, finite variation sum, common-refinement data, transport, replay, provenance, and local naming rows.

Apply Definition 398.3. It forces every public variation read to be assembled from the displayed endpoint ledger and the finite dyadic edge-observation sum.

Apply Definition 398.11 and Definition 398.13. Refinement and concatenation keep the same finite rows and only repack them along displayed common-refinement or splice coordinates.

Finally repack those rows as the FiniteVariationLedger$^{\uparrow}$ consumer route of Definition 6254.1. Since the source packet and the reusable ledger both lack host suprema, variation measures, quotient functions, completed function spaces, hidden partitions, and ambient completeness coordinates, no additional object can enter the bounded-variation read. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger finite variation ledger consumer`

Theorem 398.15 (BoundedVariation endpoint transport scope)**.** *The carrier transport and no-escape boundary rows of Definition 398.5 and Definition 398.6 give the endpoint transport scope*

for BoundedVariation$^{\uparrow}$. *Endpoint reads may move only along displayed* hsame *and* Cont *routes for the interval row, finite partition ledger,* Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *endpoint names,* StreamName$^{\uparrow}$ *read schedule,* DyadicRatCore$^{\uparrow}$ *observations, finite variation ledger,* BHist, Pkg, *and* NameCert *provenance.*

The endpoint transport scope exports no completed function space, variation measure, host supremum over all partitions, quotient real-valued function, or hidden partition row.

Proof. Apply Definition 398.5. It moves interval rows, finite partition ledgers, endpoint-name rows, dyadic observation rows, variation-sum ledgers, and common-refinement rows componentwise through displayed transport and continuation routes.

Then use Definition 398.6. The public boundary exports only the finite partition rows, transported endpoints, dyadic edge observations, common refinements, finite sums, and visible provenance.

The two rows together give the endpoint transport scope. Every endpoint read remains attached to the displayed dependency stack, and no absent host object is exported. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger endpoint transport scope`

Definition 398.16 (BoundedVariation partition ledger)**.** *A* BoundedVariation$^{\uparrow}$ *partition ledger is a finite* BHist *packet*

$$L = (I, \Pi, E, X, D, V, R, H, C, P, N)$$

whose rows refine the carrier of Definition 398.1 for a single displayed interval read:

1. *I is the displayed rational or nested interval row;*
2. *Π is a finite ordered partition of I with endpoint order and edge rows;*
3. *E is the finite edge ledger induced by adjacent endpoints of Π;*
4. *X records the endpoint* Real$^{\uparrow}$ *rows or their* RegSeqRat$^{\uparrow}$ *regular-name representatives;*
5. *D records the* DyadicRatCore$^{\uparrow}$ *edge observations used to compare adjacent endpoint values;*
6. *V is the finite dyadic variation sum over exactly the edge observations in D;*
7. *R records refinement rows comparing Π with displayed finer partitions and their transported edge ledgers;*
8. *H records the componentwise* hsame *transports for interval, partition, endpoint, edge-observation, finite-sum, refinement, package, and naming rows;*
9. *C records the displayed* Cont *routes from an interval request to endpoints, dyadic edge reads, finite variation sums, refinement comparison, and consumer exposure;*
10. *P and N are the* Pkg *provenance and local* NameCert$_{\text{BoundedVariation}^{\uparrow}}$ *rows.*

The ledger has no coordinate for a host supremum over all partitions, a completed function space, a variation measure, a quotient of real-valued functions, or a hidden partition family.

Theorem 398.17 (BoundedVariation partition refinement monotonicity)**.** *Let $L = (I, \Pi, E, X, D, V, R, H, C, P, N)$ be a* BoundedVariation$^{\uparrow}$ *partition ledger in the sense of Definition 398.16. If Π' is a displayed finite refinement of Π whose endpoint, edge-observation, and finite-sum rows are recorded in R, then the variation read on Π' is obtained by repacking the same finite edge-observation ledger through H and C. The refinement comparison therefore cannot introduce a host supremum over all partitions or any unrecorded variation object.*

Proof. Project the ledger L. The only partition rows available to a consumer are the displayed partition Π and the displayed refinement rows stored in R.

Read the refinement edge by edge. Each refined edge obtains its endpoint names from X and its dyadic observation from D after the transports recorded in H and the continuation routes recorded in C.

The finite sum row for the refinement is therefore a repacking of visible edge observations. Since the packet has no row ranging over all partitions and no measure-valued coordinate, the refinement read is monotone only inside the displayed finite ledger surface. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationPartitionLedger refinement monotonicity`

Theorem 398.18 (BoundedVariation Cauchy consumer boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{BoundedVariation}^{\uparrow}$ *packet may read bounded variation only through finite partition ledgers of Definition 398.16. Such a consumer receives endpoint regular-name or real-name rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, displayed refinement rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *data; it receives no host supremum, variation measure, completed function-space carrier, quotient function, or hidden partition family.*

Proof. Start from a consumer request. The carrier of Definition 398.1 and the ledger of Definition 398.16 expose only interval, partition, endpoint, dyadic observation, finite-sum, refinement, transport, route, provenance, and naming rows.

If the consumer is a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, it reads endpoint data through the regular-name rows in X before any dyadic edge observation is formed. If the consumer is a $\mathsf{Real}^{\uparrow}$ consumer, it reads the same endpoint positions through the displayed real-name rows and then through the dyadic observation rows.

Apply the refinement monotonicity row Definition 398.17. Refinement access remains finite and displayed, so the consumer boundary is exhausted by partition ledgers and dyadic observations rather than by any absent host supremum or completed function object. □

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier cauchy consumer boundary`

Theorem 398.19 (BoundedVariation real dependency scope)**.** *Every endpoint-variation read consumed from a* $\mathsf{BoundedVariation}^{\uparrow}$ *packet reads* $\mathsf{Real}^{\uparrow}$ *endpoint rows through chapter 38,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name rows through chapter 301,* $\mathsf{StreamName}^{\uparrow}$ *read windows through chapter 126, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations through chapter 291. The* $\mathsf{BoundedVariation}^{\uparrow}$ *packet only packages those sibling rows into a finite interval-partition ledger with visible transport, continuation, package, and naming provenance.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier real dependency scope`

Proof. Start from the carrier of Definition 398.1. Its endpoint coordinate X stores only $\mathsf{Real}^{\uparrow}$ endpoint values or $\mathsf{RegSeqRat}^{\uparrow}$ regular-name representatives, while D stores the $\mathsf{DyadicRatCore}^{\uparrow}$ observations attached to partition edges.

Use the partition ledger of Definition 398.16. A consumer reaches endpoints by first selecting a displayed finite partition and then reading the endpoint rows, stream windows, dyadic edge rows, and finite variation-sum row through the recorded routes.

Finally apply Definition 398.18. Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer is exhausted by those ledger rows, so the dependency scope is exactly the cited RealUp, RegSeqRatUp, StreamNameUp, and DyadicRatCoreUp sibling surface together with the local $\mathsf{BoundedVariation}^{\uparrow}$ provenance. □

Theorem 398.20 (BoundedVariation public NameCert export)**.** *Every public* $\mathsf{BoundedVariation}^{\uparrow}$ *consumer receives exactly the finite interval-partition rows of Definition 398.16, the endpoint rows of Definition 398.1, the dyadic variation rows, the common-refinement rows of Definition 398.4, the displayed* hsame *rows, the* Cont *replay rows, the* Pkg *provenance rows, and the local* NameCert *rows described by the carrier and obligation theorems.*

Lean4 ✓ `BEDC.Derived.BoundedVariationUp.BoundedVariationCarrier public namecert export`

Proof. Read the carrier through Definition 398.1. Its interval, endpoint, dyadic variation, refinement, transport, route, provenance, and naming coordinates are the only coordinates available to a $\mathsf{BoundedVariation}^{\uparrow}$ packet.

Next read the partition ledger through Definition 398.16. It adds only finite ordered partition rows, edge rows, endpoint rows, dyadic observations, variation sums, and common-refinement rows.

Apply Definition 398.2, Definition 398.7, and Definition 398.19. These results keep every public consumer on the same finite carrier, partition, endpoint, dyadic, refinement, hsame, Cont, Pkg, and NameCert surface, and no row outside that surface is present for export. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedVariation}^{\uparrow}$ packet is generated from an interval row, a finite ordered partition ledger, $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ endpoint names, $\mathsf{DyadicRatCore}^{\uparrow}$ edge observations, a finite dyadic variation-sum row, common-refinement rows, displayed transport, continuation, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 398.7, Definition 398.8, Definition 398.15, Definition 398.19, and Definition 398.20 bind finite partition variation, common-refinement comparison, endpoint transport, dyadic variation-sum ledger, sibling dependency scope, no-escape boundary, and the public finite-partition export as the paper-side bridge surface to the displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close completed function spaces, variation measures, host suprema over all partitions, quotient real-valued functions, hidden partition families, Jordan decomposition, Lean verification, or bridge checking.

Upgrade path: The paper-side mature package is recorded; the verification axis still requires Lean checking and bridge checking for the same public finite-variation consumer surface.

Chapter 399

A Concrete Naming Certificate for $\mathsf{WeierstrassMTest}^{\uparrow}$

The $\mathsf{WeierstrassMTest}^{\uparrow}$ interface names the finite majorant packet behind the Weierstrass M-test as used in Bishop–Bridges constructive analysis: uniform convergence is read from displayed majorant rows and finite tail ledgers. The chapter packages finite majorant rows for uniformly bounded series. It does not assert open-cover compactness, host completeness, quotient equality of function series, or an unrecorded infinite summation object.

The uniform-tail budget is consumed through chapter 435, the regular-sequence handoff through chapter 301, and the real-facing seal through chapter 38. Thus the carrier below records the L10 handoff lattice already used by its majorant, tail, regularity, and real-seal rows rather than treating the M-test packet as a kernel-only island.

Definition 399.1 (Weierstrass M-test carrier)**.** *A* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, M, B, T, R, E, H, C, P, N)$$

with the following visible rows:

1. *F is a displayed family of* $\mathsf{FastConvergentSeries}^{\uparrow}$ *summand windows, indexed only by the finite inspection rows named by the packet;*
2. *M is the majorant* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet whose tail rows supply the common comparison budget;*
3. *B is the finite domination ledger, recording each inspected summand-window bound against the corresponding majorant row;*
4. *T is the uniform tail ledger obtained by combining the domination ledger B with the fast tail schedule of M;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads the uniform tail ledger as regular rational convergence data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through R and through no separate real-facing coordinate;*
7. *H records the componentwise* hsame *transports for source windows, majorant rows, domination ledgers, tail ledgers, handoff rows, and seal rows;*
8. *C records the displayed* Cont *routes by which consumers read the source family, the majorant packet, the domination ledger, the uniform tail ledger, the regular handoff, and the real seal;*

9. *P is the* Pkg *provenance over* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the finite rational comparison rows used by the ledger;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and their recorded hsame, Cont, Pkg, *and local* NameCert *provenance.*

Lean4 (definition) `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier`

Theorem 399.2 (Weierstrass M-test NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}$ *surface consists of carrier admission, finite majorant domination, uniform tail exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal factorization for the carrier of Definition 399.1. A consumer may project the displayed source windows, the majorant* $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet, the domination ledger, the uniform tail ledger, the regular handoff, the real seal, and the transport, continuation, package, and naming rows, but obtains no open-cover compactness, host completeness, quotient of function series, or selected limiting sum.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier namecert obligations`

Proof. Carrier admission is the projection of the finite packet W. The displayed family F, majorant packet M, domination ledger B, tail ledger T, handoff row R, and seal row E are all stored coordinates.

The majorant obligation projects B. Each inspected source summand window is paired with the corresponding comparison row of M, so the certificate exposes a finite domination record instead of an ambient boundedness theorem.

Uniform tail exactness projects T and composes it with the $\mathsf{FastConvergentSeries}^{\uparrow}$ tail surface of M and the $\mathsf{DyadicTailBound}^{\uparrow}$ budget rows. Repacking T gives the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, and the $\mathsf{Real}^{\uparrow}$ seal E then reads only through that handoff.

The rows H, C, P, and N certify componentwise transport, consumer reads, dependency provenance, and local naming for the same packet. Since no coordinate names compact open covers, host completeness, quotient series, or a selected limiting sum, none can be exported by the certificate. □

Theorem 399.3 (Weierstrass M-test majorant tail exactness)**.** *For every accepted carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

the uniform tail ledger T is obtained by composing the finite domination ledger B with the tail schedule of the majorant $\mathsf{FastConvergentSeries}^{\uparrow}$ *packet M. Every displayed source-family tail comparison consumed from T therefore factors through the same majorant packet, the same finite domination rows, and the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *budget surface.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier majorant tail transport`

Proof. Project M and B from the carrier. The packet M supplies the fast-convergent majorant tail schedule, while B records the inspected domination comparisons between the summand windows in F and the majorant rows.

For a requested uniform precision, read the threshold supplied by M and apply the domination rows in B only over the finite inspected windows. The resulting comparison is recorded as the corresponding row of T.

The DyadicTailBound$^{\uparrow}$ budget rows certify that the tolerance used by the comparison is the displayed one. Thus a consumer of T cannot replace the majorant packet, domination ledger, or tail budget by unrecorded boundedness data. □

Theorem 399.4 (Weierstrass M-test RegSeqRat handoff)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier, the* RegSeqRat$^{\uparrow}$ *row R is obtained by repacking the displayed uniform tail ledger T together with the majorant packet M and the domination ledger B. A* RegSeqRat$^{\uparrow}$ *consumer can read the resulting uniformly convergent name only through those rows and through the stored* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier regseqrat handoff`

Proof. Read the carrier coordinates M, B, T, and R. The row T is already the finite tail ledger produced from the majorant and domination rows by Definition 399.3.

Repack that ledger as regular rational sequence evidence. The handoff retains the majorant schedule and domination rows because they are stored in the same BHist packet and reached by the displayed continuation routes.

The transport, package, and local naming rows certify the same finite dependency surface. Hence R is a visible RegSeqRat$^{\uparrow}$ handoff and not a quotient of function series, a chosen sum, or a completeness argument. □

Theorem 399.5 (Weierstrass M-test real-seal factorization)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

the Real$^{\uparrow}$ *seal row E factors through the* RegSeqRat$^{\uparrow}$ *handoff R, which is determined by the majorant packet M, the domination ledger B, and the uniform tail ledger T. The rows H, C, P, and N certify transport, continuation readback, package provenance, and local naming, but they do not supply open-cover compactness, ambient completeness, quotient equality of function series, or a selected limiting sum.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier real seal factorization`

Proof. Project the seal row E and the handoff row R from the carrier. By definition, E has no real-facing route except the regular rational readback R.

Use Definition 399.4 to identify R with the repacking of M, B, and T. Then use Definition 399.3 to bind T to the finite majorant comparison surface and the displayed tail budget.

The remaining rows are transport, continuation, package, and local naming witnesses for those displayed coordinates. Since the carrier has no coordinate for open covers, host completeness, quotient series, or selected sums, the real seal factors only through the finite majorant certificate. □

Theorem 399.6 (Weierstrass M-test uniform tail budget scope)**.** *Let $W = (F, M, B, T, R, E, H, C, P, N)$ be an accepted* WeierstrassMTest$^{\uparrow}$ *carrier whose displayed summand windows in F are dominated by the majorant packet M through the finite ledger B. Then the packet carries a single* BHist *uniform-tail budget T whose* RegSeqRat$^{\uparrow}$ *handoff row R and* Real$^{\uparrow}$ *seal row E are transported by the stored* hsame *rows and read by the stored* Cont *routes along the same package provenance P.*

Equivalently, a consumer of the WeierstrassMTest$^{\uparrow}$ *seal must pass through the majorant domination ledger, the uniform-tail budget, and the regular-sequence handoff before reaching E. No consumer obtains a separate host summation, quotient series, open-cover compactness, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier uniform tail budget scope`

Proof. First build the uniform-tail budget. Apply Definition 399.3: the ledger T is produced from the displayed domination rows B and the tail schedule of the majorant packet M.

Next route the budget to regular Cauchy data. Apply Definition 399.4 to repack T, together with M and B, as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R.

Finally route the regular handoff to the real-facing seal. Apply Definition 399.5: the seal row E factors through R and keeps the same H, C, P, and N provenance.

The composed route is a single displayed BHist budget scope. Since each step uses only the carried rows of W, no unrecorded summation object, quotient series, compactness principle, or completeness principle is introduced. □

Theorem 399.7 (Weierstrass M-test tail majorant ledger)**.** *For every accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

each tail comparison recorded in the uniform tail ledger T is pinned to the displayed majorant packet M and the finite domination ledger B. The carrier has no alternate route for deriving tail bounds from open-cover compactness, ambient completeness, a quotient series object, or a selected sum.

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier tail majorant ledger`

Proof. Project M, B, and T from the carrier. These are the only rows that name the majorant schedule, domination evidence, and uniform tail budget.

Apply Definition 399.3. It identifies every tail read in T with the majorant packet M and the domination ledger B.

The remaining rows H, C, P, N only transport, read, package, and locally name those displayed coordinates. Since the carrier has no coordinate for compactness, completeness, quotient series, or selected sums, no such object can supply a competing tail comparison. □

Theorem 399.8 (Weierstrass M-test uniform Cauchy handoff)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *row R of an accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier is exactly the uniform Cauchy handoff produced by the rows*

$$M, B, T, H, C, P, N.$$

It records the majorant packet, domination ledger, uniform tail budget, transport witnesses, continuation reads, package provenance, and local naming row, and it does not add a separate summation object or host completeness proof.

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier uniform cauchy handoff`

Proof. Begin with the ledger boundary. Apply Definition 399.7 to bind the tail budget T to M and B.

Next apply Definition 399.4. The handoff repacks the majorant-ledger and tail-budget data as the $\mathsf{RegSeqRat}^{\uparrow}$ row R.

Finally use the displayed H, C, P, N rows for transport, continuation readback, package provenance, and local naming. These rows certify the same uniform Cauchy surface and do not create a completed sum or a host completeness argument. □

Theorem 399.9 (Weierstrass M-test real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier factors through the uniform Cauchy handoff row* R *determined by*

$$M, B, T, H, C, P, N.$$

The seal row E *therefore exposes only the finite majorant-ledger surface and not a quotient series, selected sum, open-cover compactness proof, ambient completeness proof, or host-level uniform convergence theorem.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier real seal boundary`

Proof. First apply Definition 399.8. This identifies R as the regular rational handoff produced by the majorant, domination, and tail rows together with their transport and naming evidence.

Then apply Definition 399.5. The real-facing seal E factors through R, so every seal read must pass through the same finite handoff.

The excluded objects are absent from the carrier and from the handoff route. Thus the real seal remains a boundary over the displayed majorant-ledger surface. □

Definition 399.10 (Weierstrass M-test obligation surface)**.** *For an accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

the obligation surface consists of carrier admission, finite majorant domination, uniform tail exactness, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal factorization. It is read from the displayed summand windows* F*, majorant packet* M*, domination ledger* B*, uniform tail ledger* T*, regular handoff* R*, real seal* E*, and provenance rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestObligationSurface namecert rows`

Theorem 399.11 (Weierstrass M-test majorant domination package)**.** *For every accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, the carrier-admission and finite majorant-domination obligations of Definition 399.10 are witnessed by the local NameCert surface and the tail-majorant ledger.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier majorant domination package`

Proof. Apply Definition 399.2. It exposes the accepted carrier rows and the local naming obligations over $F, M, B, T, R, E, H, C, P, N$.

Apply Definition 399.7. It binds the displayed summand windows to the majorant packet through the finite domination ledger B.

Thus the majorant-domination package remains a finite BEDC packet over the listed rows. □

Theorem 399.12 (Weierstrass M-test majorant tail domination surface)**.** *For every accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, the local majorant-domination package and the tail-majorant ledger share the same displayed domination row* B*, tail row* T*, regular handoff* R*, and package provenance* P, N*.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier majorant tail domination surface`

Proof. Apply Definition 399.11 to expose the local naming surface and the read of the finite majorant-domination row.

Apply Definition 399.7 to expose the uniform tail read and its hsame alignment with the regular handoff row.

The combined surface is still the same finite carrier route through $F, M, B, T, R, E, H, C, P, N$ and adds no ambient convergence principle. □

Theorem 399.13 (Weierstrass M-test uniform-tail obligation package)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, the uniform tail and regular-rational handoff obligations of Definition 399.10 are witnessed by* M, B, T, R, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.TasteGate.WeierstrassMTestTasteGate single carrier alignment`

Proof. Use Definition 399.3. The majorant tail request is the finite source of the uniform tail budget.

Use Definition 399.6. It keeps the uniform budget inside the displayed tail ledger T and the provenance rows.

Finally use Definition 399.8. The regular-rational handoff row R is therefore read from the same finite majorant and tail rows. □

Theorem 399.14 (Weierstrass M-test real-seal obligation package)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, the* Real$^{\uparrow}$ *seal-factorization obligation of Definition 399.10 is the displayed route through* R, E, H, C, P, N *after the finite majorant and uniform-tail ledgers* M, B, T *have been read.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier real seal obligation package`

Proof. Apply Definition 399.4. It identifies R as the regular-rational row built from the domination and uniform-tail data.

Apply Definition 399.5 and Definition 399.9. These show that every real-facing seal read factors through R and the same finite majorant-ledger route.

The carrier has no coordinate for open-cover compactness, host completeness, quotient series, or a selected limiting sum, so the obligation remains inside the displayed packet. □

Theorem 399.15 (Weierstrass M-test obligation closure package)**.** *Every accepted* WeierstrassMTest$^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N)$$

admits the five obligation rows of Definition 399.10. The package is read over FastConvergentSeries$^{\uparrow}$*,* DyadicTailBound$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it exports carrier admission, finite majorant domination, uniform tail exactness,* RegSeqRat$^{\uparrow}$ *handoff, and* Real$^{\uparrow}$ *seal factorization without open-cover compactness, host completeness, quotient series, or a selected limiting sum.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier obligation closure package`

Proof. Project the accepted carrier W and read the displayed rows $F, M, B, T, R, E, H, C, P, N$. The definition of the obligation surface asks only for the five rows named in Definition 399.10.

Apply Definition 399.11. This supplies carrier admission and finite majorant domination from the local naming surface and the tail-majorant ledger.

Apply Definition 399.13. This supplies the uniform tail row and the regular-rational handoff through M, B, T, R, H, C, P, N.

Apply Definition 399.14. This supplies the real-seal factorization through R, E, H, C, P, N after the finite majorant and uniform-tail ledgers have been read.

The carrier non-escape inventory is exactly the listed packet surface. Since no coordinate names open-cover compactness, host completeness, quotient series, or a selected limiting sum, the combined package closes precisely the displayed obligation rows and no stronger host theorem. □

Theorem 399.16 (Weierstrass M-test scoped dependency package)**.** *For every accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

the finite series packet, dyadic tail bound, fast convergent series row, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *consumer routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}$ *row exhaust the scoped proof surface. Every domination, uniform tail, regular-sequence handoff, and real-seal read factors through those displayed rows.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier scoped dependency package`

Proof. Project the carrier and the obligation package from Definition 399.15. The rows $F, M, B, T, R, E, H, C, P, N$ supply the finite summand packet, majorant packet, dyadic tail ledger, uniform-tail route, regular rational handoff, real seal, transport rows, consumer routes, package provenance, and local naming row.

Route the estimates through the existing tail machinery. The dyadic tail bound is read through Definition 399.13, while the fast convergent series and majorant rows are read through Definition 399.11.

Route convergence reads through Definition 399.4, Definition 399.14, and Definition 399.9. These results keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ outputs attached to the same finite package, with no host infinite sum, ambient completeness principle, quotient series, or selected limiting sum in scope. □

Theorem 399.17 (Weierstrass M-test public convergence export)**.** *For every accepted* $\mathsf{WeierstrassMTest}^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

the finite series packet F*, majorant packet* M*, dyadic tail ledger* T*, uniform-tail route,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*,* hsame *rows* H*,* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *form the public M-test consumer surface. The surface exports convergence through the displayed majorant domination, uniform-tail obligation package, regular-rational handoff, and real-seal boundary, and it exports no open-cover compactness, host completeness, quotient equality of function series, or selected limiting sum.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier public convergence export`

Proof. First compose the scoped dependency package with the obligation closure package. This reads the accepted carrier as one finite packet and exposes exactly the source series, majorant, domination ledger, tail ledger, regular handoff, real seal, transport, continuation, package, and naming rows.

Next apply majorant domination and the uniform-tail obligation package. These rows turn the finite domination ledger and dyadic tail budget into the uniform-tail route consumed by the $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Finally apply the RegSeqRat handoff, the real-seal obligation package, and the real-seal boundary. They factor every real-facing convergence read through R and E, so the public consumer surface remains the displayed BHist packet and never uses compactness, host completeness, quotient series, or a selected sum. □

Theorem 399.18 (Weierstrass M-test bridge majorant seal)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, an accepted majorant-tail package consumed by a* Real$^{\uparrow}$ *convergence consumer must pass through the* RegSeqRat$^{\uparrow}$ *handoff* R*, the uniform-tail obligation package, the real-seal obligation package, and the* CauchyLimitSeal$^{\uparrow}$ *boundary before it exports the public convergence row of Definition 399.17. The bridge exposes only the displayed rows* $F, M, B, T, R, E, H, C, P, N$ *and no selected sum, quotient series, open-cover compactness, or host completeness.*

Lean4 ✓ `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier bridge majorant seal`

Proof. Begin with majorant-tail exactness. Definition 399.3 reads the accepted majorant packet and domination ledger as the source of the tail budget, and Definition 399.13 packages that budget as the uniform-tail row consumed by the carrier.

Next pass through the regular and real-facing rows. Definition 399.4 repacks the uniform-tail data as the RegSeqRat$^{\uparrow}$ handoff R, and Definition 399.14 factors the Real$^{\uparrow}$ seal read through R, E, H, C, P, N.

Finally compare the exported surface with the seal route. Definition 399.17 exposes the public convergence row from the same finite packet, while Definition 399.19 identifies the compatible CauchyLimitSeal$^{\uparrow}$ boundary. Hence the public export is reached only after the majorant-tail, uniform-tail, regular-handoff, real-seal, and CauchyLimitSeal rows have been read. □

Theorem 399.19 (Weierstrass M-test CauchyLimitSeal route)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier* $W = (F, M, B, T, R, E, H, C, P, N)$*, the uniform tail ledger* T*,* RegSeqRat$^{\uparrow}$ *handoff* R*, and* Real$^{\uparrow}$ *seal* E *determine a* CauchyLimitSeal$^{\uparrow}$*-compatible finite route through the displayed* FastConvergentSeries$^{\uparrow}$ *majorant,* DyadicTailBound$^{\uparrow}$ *budget,* RegSeqRat$^{\uparrow}$ *row,* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exports no quotient series, selected sum, host completeness, or open-cover compactness.*

The Cauchy-limit boundary is the finite seal route of chapter 386, consumed after the majorant-tail, uniform-tail, regular-handoff, and Real-seal rows have been displayed.

Proof. First read the uniform tail budget. By Definition 399.6, the tail ledger T is tied to the displayed majorant and dyadic budget rows, so every tail estimate remains inside the finite M-test packet.

Next pass to the regular rational handoff. The RegSeqRat handoff theorem Definition 399.4 routes the same tail data through R, and the real-seal factorization of Definition 399.5 then sends the real-facing read through E with the packet's BHist, hsame, Cont, Pkg, and NameCert rows.

Finally compare this route with the CauchyLimitSeal window and consumer boundaries, Definition 386.3 and Definition 386.5. Both boundaries require a displayed regular source route before the real seal is consumed, which is exactly the route supplied by T, R, and E. The non-escape clauses in the M-test public export keep quotient series, selected sums, host completeness, and compactness outside the packet. □

Theorem 399.20 (Weierstrass M-test finite-majorant exhaustion)**.** *For every accepted* WeierstrassMTest$^{\uparrow}$ *carrier*

$$W = (F, M, B, T, R, E, H, C, P, N),$$

every public M-test seal factors through the displayed majorant packet M, the finite domination ledger B, the uniform tail budget T, the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, and the* $\mathsf{Real}^{\uparrow}$ *seal E. Equivalently, the public seal is exhausted by the L10 route*

$$M, B \longrightarrow T \longrightarrow R \longrightarrow E$$

with H, C, P, N transporting, replaying, packaging, and naming the route.

No open-cover compactness, quotient series object, host summation operation, ambient completeness principle, selected limiting sum, or alternate real seal is available to the public consumer.

Proof. First bind the majorant surface. Apply Definition 399.3 and Definition 399.7: every tail comparison read from T is produced from the displayed majorant packet M and the finite domination ledger B.

Next pass through the regular handoff. Apply Definition 399.13 and Definition 399.4. These results identify R as the regular rational readback obtained from the same majorant packet, domination ledger, and uniform tail budget.

Finally seal the route at the real boundary. Apply Definition 399.5, Definition 399.9, and Definition 399.19. The seal row E is therefore reached only after the majorant, domination, tail, and regular handoff rows have been read.

The transport rows H, continuation rows C, package provenance P, and local naming row N replay the same displayed route. Since the carrier has no coordinate for open covers, quotient series, host summation, ambient completeness, selected sums, or alternate real seals, the public M-test seal is exhausted by the finite majorant packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}$.

Constructive story: A $\mathsf{WeierstrassMTest}^{\uparrow}$ packet is generated from displayed $\mathsf{FastConvergentSeries}^{\uparrow}$ summand windows, a majorant $\mathsf{FastConvergentSeries}^{\uparrow}$ packet, a finite domination ledger, a uniform tail ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature surface binds the majorant, uniform-tail, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, local NameCert rows, and public convergence export to one displayed BHist package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeierstrassMTest}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.WeierstrassMTestUp.WeierstrassMTestCarrier\_public\_convergence\_export`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host completeness, quotient equality of function series, selected limiting sums, or bridge checking.

Upgrade path: Bridge checking requires an explicit standard-bridge theorem for the same $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 400

A Concrete Naming Certificate for $\mathsf{FanTheorem}^{\uparrow}$

The $\mathsf{FanTheorem}^{\uparrow}$ interface names the finite-spread packet used when later compactness and uniform-continuity consumers read a constructive finite fan argument. In Bishop constructive analysis and in the Troelstra–van Dalen treatment of finite spreads and bars, the finite fan theorem is a boundedness principle for decidable bars on finitely branching binary spreads. BEDC records only the visible finite packet: binary branch rows, branch-prefix $\mathsf{List}^{\uparrow}$ and tree rows, a decidable finite bar classifier, unary depth bounds, componentwise hsame transports, Cont traversal routes, Pkg provenance, and a local $\mathsf{NameCert}$ row.

The packet is not Brouwer bar induction, not open-cover compactness, not a choice principle, and not a $\mathsf{Real}^{\uparrow}$ completeness theorem. It is a concrete finite naming surface that downstream certificates can consume without importing an ambient compact space.

The finite fan packet is a sibling input for chapter 388. $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumes $\mathsf{FanTheorem}^{\uparrow}$ only as a finite branch-depth and prefix-window certificate, never as open-cover compactness or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 400.1 (FanTheorem binary spread carrier)**.** *A* $\mathsf{FanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, D, W, H, C, P, N)$$

with the following visible rows:

1. *T is the finite binary spread row, displayed as a prefix-closed tree of* $\mathsf{Bool}^{\uparrow}$ *branch histories and branch-prefix* $\mathsf{List}^{\uparrow}$ *rows;*
2. *B is the decidable finite bar row, assigning to each displayed prefix a visible* $\mathsf{Bool}^{\uparrow}$ *classifier saying whether that prefix is barred;*
3. *D is the unary depth-bound ledger, a finite* $\mathsf{Nat}^{\uparrow}$ *row recording the candidate depth through which every branch-prefix read is inspected;*
4. *W is the finite witness window, listing the branch-prefix rows of length at most the displayed depth and the bar decisions actually read from B;*
5. *H stores componentwise* hsame *transports for the tree row, prefix rows, bar decisions, depth ledger, witness window, provenance row, and local naming row;*
6. *C stores the displayed* Cont *traversal routes for root reads, left and right branch extension, prefix restriction, depth comparison, and bar-decision readback;*

7. *P is the* Pkg *provenance naming the consumed* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite binary sequence, a host compact space, an open cover, a choice-selected branch, Brouwer bar induction, or real completeness.

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremTasteGate single carrier alignment`

Definition 400.2 (FanTheorem bar classifier)**.** *Two accepted* $\mathsf{FanTheorem}^{\uparrow}$ *carriers are classified as the same finite fan packet exactly when their displayed rows compare componentwise. Their binary spread rows agree by transported* $\mathsf{Tree}^{\uparrow}$ *and* $\mathsf{List}^{\uparrow}$ *prefix classifiers, their bar decisions agree by the displayed* $\mathsf{Bool}^{\uparrow}$ *rows, their depth ledgers agree by the unary* $\mathsf{Nat}^{\uparrow}$ *rows, their finite witness windows agree prefix by prefix, and their* Pkg *provenance and local* NameCert *rows name the same packet.*

The classifier has no generator for comparing infinite branches, no generator for an open-cover index, no generator for a choice function, and no generator for a $\mathsf{Real}^{\uparrow}$ *completion object.*

Lean4 (definition) `BEDC.Derived.FanTheoremUp.FanTheoremPacket bar classifier`
Lean4 (definition) `BEDC.Derived.FanTheoremUp.FanTheoremBarClassifier`

Theorem 400.3 (FanTheorem NameCert obligations)**.** *The finite packet of Definition 400.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ *obligation surface for a decidable finite bar on a binary spread. Carrier habitation is witnessed by the empty prefix tree with its displayed depth row, classifier stability is the componentwise classifier of Definition 400.2, and ledger exactness says that every accepted read factors through the displayed prefix tree, bar decision row, unary depth ledger, finite witness window,* hsame *transports,* Cont *traversal routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket namecert obligations`

Proof. For habitation, take the root prefix as the finite binary spread and use the displayed unary depth row for the witness window. The bar classifier row is decidable because it is a finite $\mathsf{Bool}^{\uparrow}$ ledger over the retained prefixes.

For stability, compare two accepted packets by the rows named in Definition 400.2. Tree transport moves prefix rows only through the displayed $\mathsf{Tree}^{\uparrow}$ and $\mathsf{List}^{\uparrow}$ classifiers, bar decisions move only through $\mathsf{Bool}^{\uparrow}$, and depth rows move only through $\mathsf{Nat}^{\uparrow}$.

For exactness, inspect any consumer read of the packet. The read must project one of the coordinates T, B, D, W, H, C, P, or N, and each coordinate is part of the finite carrier. Repacking these projections gives the stated local $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ surface without adding infinite branches, open covers, choice functions, bar induction, or real-completeness data. □

Theorem 400.4 (FanTheorem finite bar handoff)**.** *Every downstream compactness or uniform-continuity consumer that reads an accepted* $\mathsf{FanTheorem}^{\uparrow}$ *packet receives only the finite bar handoff: a prefix-closed binary spread row, a decidable finite bar classifier, a unary depth bound, the finite branch-prefix window through that depth, componentwise* hsame *transports, displayed* Cont *traversal routes,* Pkg *provenance, and the local* NameCert *row.*

The handoff supplies a bounded finite inspection surface for binary fan arguments. It does not supply Brouwer bar induction, open-cover compactness, a choice-selected branch, a host Cantor-space object, or a $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket finite bar handoff`

Proof. Project the accepted carrier of Definition 400.1. The only branch data available to the consumer are the finite tree row T and the finite prefix window W.

Read the bar information through B and the depth information through D. Decidability is the stored $\mathsf{Bool}^{\uparrow}$ row at each retained prefix, while boundedness is the stored unary $\mathsf{Nat}^{\uparrow}$ row; neither read mentions an infinite branch or an open cover.

The traversal routes are exactly the continuations in C, transported by the componentwise hsame rows in H. Repacking those rows with P and N gives the finite bar handoff, and coordinate exhaustion excludes Brouwer bar induction, choice, host Cantor space, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 400.5 (FanTheorem real completion boundary)**.** *When a* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, or* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer cites a* $\mathsf{FanTheorem}^{\uparrow}$ *packet, the citation can use only the finite fan rows of Definition 400.4. In particular, the citation may bound a finite branch-prefix inspection, but it cannot assert real completeness, quotient equality of real names, selection of a real limit, open-cover compactness, sequential compactness, or Brouwer bar induction.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket real completion boundary`

Proof. Start with the consumer citation and factor it through the local $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ surface of Definition 400.3. The cited packet exposes only the finite carrier rows and their displayed transport routes.

Apply the finite bar handoff of Definition 400.4. It identifies the available data as a binary spread row, decidable bar row, unary depth row, finite prefix window, transports, continuations, provenance, and local naming row.

None of these coordinates is a $\mathsf{Real}^{\uparrow}$ completion object, quotient classifier for real names, selected real limit, open-cover compactness witness, sequential compactness principle, or Brouwer bar-induction rule. The real-facing consumer therefore receives only a finite inspection bound and cannot read any stronger compactness or completeness principle from the packet. □

Theorem 400.6 (FanTheorem bar uniform modulus handoff)**.** *A* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *or finite compactness consumer citing an accepted* $\mathsf{FanTheorem}^{\uparrow}$ *packet can extract only a unary finite branch depth, the displayed finite witness window, and the pointwise modulus rows routed through the recorded* Cont *and* Pkg *provenance. The resulting uniform bound is a finite BEDC packet over the branch-prefix rows and their modulus ledgers, not an open-cover compactness theorem, a selected branch, choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket bar uniform modulus`

Proof. Read the accepted fan packet through Definition 400.4. The consumer receives the prefix-closed binary spread row, decidable bar row, unary depth ledger, finite witness window, componentwise transports, continuation routes, provenance, and local naming row.

The pointwise modulus rows consumed by a compact-uniform-continuity argument are therefore attached only to the retained branch-prefix window. The unary depth row bounds the branch inspection, and the displayed Cont routes record how each pointwise row is reached from the finite spread data.

Apply Definition 400.5 to the same citation. It prevents the consumer from replacing the finite inspection bound by $\mathsf{Real}^{\uparrow}$ completeness, an open-cover compactness principle, sequential compactness, Brouwer bar induction, or a choice-selected branch.

Repacking the depth, witness window, pointwise modulus rows, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row gives the uniform-modulus handoff. Since all coordinates are finite carrier rows, the handoff remains a BEDC packet. □

Theorem 400.7 (FanTheorem public uniform bar export)**.** *The carrier, classifier, NameCert obligations, finite-bar handoff, real-completion boundary, and uniform-modulus handoff assemble a single public* $\mathsf{FanTheorem}^{\uparrow}$ *export packet consumable by* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *and constructive compact-metric fan arguments. The public row exposes only the binary-spread carrier, bar classifier, finite subcover witness window, real-completion boundary, and uniform-modulus ledger; it does not add open-cover compactness or choice-selected bars.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket public uniform bar export`

Proof. First use Definition 400.1 and Definition 400.2. They identify the binary-spread carrier and the componentwise bar classifier that every exported packet must expose.

Next apply Definition 400.3 and Definition 400.4. These results bind the local naming obligations and supply the finite subcover witness window available to downstream compactness consumers.

Finally apply Definition 400.5 and Definition 400.6. The public packet therefore exports the finite bar rows and uniform-modulus ledger while retaining the stated boundary against open-cover compactness, choice-selected bars, Brouwer bar induction, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 400.8 (FanTheorem compact uniform continuity obligation surface)**.** *The public uniform bar export of Definition 400.7 supplies exactly the rows a* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer may read over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$*: finite binary-spread rows, decidable-bar rows, unary-depth rows, finite witness-window rows, the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite coverage surface, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus surface,* BHist *histories, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ *row.*

It supplies no open-cover compactness row, no choice-selected bar, no sequential compactness row, and no $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket compact uniform continuity obligation surface`

Proof. Start with the finite bar handoff of Definition 400.4. It contributes the finite binary-spread rows, decidable-bar classifier, unary-depth bound, finite witness window, transport rows, continuation routes, package provenance, and local naming row.

Use the bar uniform modulus handoff of Definition 400.6. That result is the route by which a compact-uniform-continuity consumer may read the finite witness window as a $\mathsf{UniformModulus}^{\uparrow}$-facing ledger over the relevant pointwise rows.

Now combine the public uniform bar export of Definition 400.7 with the compact-facing rows named there. The export is therefore readable by $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ consumers only through finite coverage, pointwise-modulus, and uniform-modulus coordinates.

Finally apply the real completion boundary of Definition 400.5. It removes any coordinate for open-cover compactness, choice-selected bars, sequential compactness, or $\mathsf{Real}^{\uparrow}$ completeness, so the listed rows are exactly the public obligation surface. □

Theorem 400.9 (FanTheorem standard finite-fan bridge)**.** *The public uniform bar export of Definition 400.7 interprets as the standard finite binary fan packet over finite spread rows, decidable bar rows, a unary depth bound, a finite witness window, the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite coverage surface, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus surface,* BHist *histories, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$ *row.*

This interpretation refuses Brouwer bar induction, open-cover compactness, choice-selected bars, sequential compactness, and $\mathsf{Real}^{\uparrow}$ *completeness. Those principles are not bridge data for the finite fan packet.*

Lean4 ✓ `BEDC.Derived.FanTheoremUp.FanTheoremPacket standard finite fan bridge`

Proof. First apply the finite bar handoff of Definition 400.4. It supplies the finite binary spread row, decidable finite bar classifier, unary depth ledger, finite witness window, componentwise transports, traversal routes, provenance, and local naming row.

Next apply the real completion boundary of Definition 400.5. This step removes any reading of the packet as $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, sequential compactness, choice-selected bar selection, or Brouwer bar induction.

Use the bar uniform modulus handoff of Definition 400.6. The finite witness window can be read as a $\mathsf{UniformModulus}^{\uparrow}$-facing ledger over pointwise modulus rows, with no additional compactness principle beyond the displayed finite rows.

Finally combine the public uniform bar export of Definition 400.7 with the compact uniform continuity obligation surface of Definition 400.8. The resulting bridge packet has exactly the finite spread, decidable bar, depth, witness-window, uniform-modulus, compact-coverage, pointwise-modulus, BHist, hsame, Cont, Pkg, and NameCert coordinates stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FanTheorem}^{\uparrow}$ packet is generated from a finite binary spread over BHist, a decidable finite bar predicate, a unary depth-bound ledger, branch-prefix $\mathsf{List}^{\uparrow}$ and tree rows, componentwise hsame transports, displayed Cont traversal routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 400.1, Definition 400.2, Definition 400.3, Definition 400.4, Definition 400.5, Definition 400.6, and Definition 400.7 bind the binary spread carrier, bar classifier, finite bar handoff, real completion boundary, bar uniform modulus handoff, and public uniform bar export over the displayed BHist, hsame, Cont, Pkg, and NameCert dependencies.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Brouwer bar induction, open-cover compactness, choice, host Cantor-space compactness, sequential compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The chapter is at the top of the paper closure axis; Lean verification and bridge checking remain separate verification-axis work.

Chapter 401

A Concrete Naming Certificate for $\mathsf{FibonacciCube}^{\uparrow}$

The $\mathsf{FibonacciCube}^{\uparrow}$ interface is a BEDC-native finite certificate surface. It packages the finite-path independent-set reading of Fibonacci cubes as visible BHist rows rather than importing an external graph library or theorem.

The carrier consumes the path-graph row from chapter 118, the word row from chapter 36, and the unary length row from chapter 28.

Definition 401.1 (FibonacciCube carrier)**.** *A* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (n, G, W, I, Z, D, P, N)$$

with the following displayed coordinates:

1. *n is a unary* $\mathsf{Nat}^{\uparrow}$ *length row;*
2. *G is the path-graph source row with vertices indexed by the visible initial segment of length n and edges only between successor indices;*
3. *W is a* $\mathsf{List}^{\uparrow}$ *word row over* $\mathsf{Bool}^{\uparrow}$ *whose support has no adjacent* 1 *entries;*
4. *I is the path-independent-set row listing exactly the positions where W carries* 1*;*
5. *Z is the Zeckendorf and* $\mathsf{GoldenMeanShift}^{\uparrow}$ *provenance row recording the same no-adjacent-*1 *classifier as a finite shift word;*
6. *D records dependency rows for* $\mathsf{Graph}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*, and* $\mathsf{Nat}^{\uparrow}$ *together with the visible* hsame *comparisons and* Cont *routes between path vertices, list positions, word support, and independent-set membership;*
7. *P is the* Pkg *provenance row naming the* $\mathsf{Graph}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{ZeckendorfNormal}^{\uparrow}$*, and* $\mathsf{GoldenMeanShift}^{\uparrow}$ *sources;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FibonacciCube}^{\uparrow}}$ *row.*

The local classifier compares carriers componentwise by the length row, path-graph row, no-adjacent word row, independent-set row, provenance row, dependency rows, package row, and local certificate row.

Lean4 ✓ `BEDC.Derived.FibonacciCubeUp.FibonacciCubeTasteGate single carrier alignment`

Theorem 401.2 (FibonacciCube carrier habitation)**.** *For each displayed unary length row n, the empty support word on the path graph of length n supplies an accepted* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier. Its path-independent-set row is empty, its no-adjacent-*1 *word row is the all-*0 *word, and its provenance is witnessed by the* $\mathsf{Graph}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, *and* $\mathsf{GoldenMeanShift}^{\uparrow}$ *dependency rows.*

Lean4 ✓ `BEDC.Derived.FibonacciCubeUp.FibonacciCubeCarrier habitation`

Proof. Build the path-graph source row from the unary length row and choose the all-0 list word. The support row is empty, so no edge of the path graph can have both endpoints in the support.

The empty support is also the independent-set row. The Zeckendorf and golden-mean-shift provenance rows read the same all-0 finite word, and the componentwise hsame and Cont routes are the identity routes on the displayed rows. □

Theorem 401.3 (FibonacciCube word and path-independent-set correspondence)**.** *For an accepted* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier, the no-adjacent-*1 *word row W and the path-independent-set row I determine each other on the visible path graph: a position belongs to I exactly when the corresponding entry of W is* 1*, and path independence is exactly the absence of adjacent* 1 *entries along successor edges.*

Lean4 ✓ `BEDC.Derived.FibonacciCubeUp.FibonacciCubePacket word path support correspondence`

Proof. Read I from W by taking the displayed support row. Since W has no adjacent 1 entries, no successor edge of the path graph has both endpoints in the support.

Conversely, read W from I by marking precisely the listed positions with 1 and all other visible positions with 0. Path independence of I forbids a successor pair from both being marked, so the reconstructed word satisfies the no-adjacent-1 classifier.

The two readings use only the stored vertex-position hsame comparisons and the finite Cont routes between list entries and graph vertices, so they remain inside the carrier. □

Theorem 401.4 (FibonacciCube finite path coverage)**.** *Every classifier read of an accepted* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier is covered by the finite path graph of the displayed unary length row. There is no vertex outside the initial segment, no edge outside successor adjacency, and no membership query outside the listed* $\mathsf{List}^{\uparrow}$ *word and path-independent-set rows.*

Proof. The carrier records the length row and path-graph source row as coordinates. Hence every vertex read is mediated by the visible initial segment, and every edge read is mediated by the successor-edge row.

The word row and independent-set row are indexed by the same displayed positions. The dependency coordinate records the hsame comparisons and Cont routes that identify list positions with path vertices, so a classifier query cannot name an external vertex, edge, or membership source. □

Theorem 401.5 (FibonacciCube classifier transport)**.** *Displayed* hsame *transport preserving the length row, path-graph row, word row, independent-set row, Zeckendorf and* $\mathsf{GoldenMeanShift}^{\uparrow}$ *provenance row, dependency rows, package row, and local certificate row preserves the* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier predicate and its local classifier.*

Proof. The classifier is componentwise by Definition 401.1. Transport of the length and path-graph rows preserves the visible initial segment and successor-edge relation.

Transport of the word, support, and independent-set rows preserves the no-adjacent-1 and path-independent-set readings because the relevant entries and successor-edge checks are exactly

the transported rows. The provenance, package, and local certificate rows then repack the same finite surface with no additional classifier data. □

Theorem 401.6 (FibonacciCube ledger non-escape)**.** *An accepted* $\mathsf{FibonacciCube}^{\uparrow}$ *carrier exports no ambient graph theory, no infinite binary shift, no external Fibonacci-counting theorem, and no external object. Its public ledger consists only of the finite path-graph row, no-adjacent word row, path-independent-set row, Zeckendorf and* $\mathsf{GoldenMeanShift}^{\uparrow}$ *provenance row, dependency rows for* $\mathsf{Graph}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*, and* $\mathsf{Nat}^{\uparrow}$*,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Proof. Inspect the carrier coordinates. Each exported field is a finite history row or a route between finite history rows.

The path-independent-set correspondence is recorded as a BEDC classifier relation between the word support and the path graph, not as an imported theorem about an external graph type. The counting interpretation is therefore only provenance for the concrete certificate surface.

Since no coordinate contains an ambient graph carrier, an infinite shift object, or an external artifact, a downstream consumer can cite only the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FibonacciCube}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FibonacciCube}^{\uparrow}$ packet is generated from a unary $\mathsf{Nat}^{\uparrow}$ length row, a finite $\mathsf{Graph}^{\uparrow}$ path row, a no-adjacent-1 $\mathsf{List}^{\uparrow}$ word row, a path-independent-set support row, Zeckendorf and $\mathsf{GoldenMeanShift}^{\uparrow}$ provenance, componentwise hsame transport, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FibonacciCube}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The obligation surface names carrier habitation, support correspondence, finite path coverage, classifier transport, and ledger non-escape for the concrete finite-path word/support packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FibonacciCube}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FibonacciCubeUp.FibonacciCubePacket\_word\_path\_support\_correspondence`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Fibonacci-cube enumeration, outside candidate records, imported theorem-source records, external formalization objects, graph-isomorphism completeness, or full Lean coverage of every obligation row.

Upgrade path: No further paper-side theory-axis upgrade remains for this finite path word/support packet; bridge verification remains a separate formal-axis task over the same $\mathsf{Graph}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{GoldenMeanShift}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 402

A Concrete Naming Certificate for $\mathsf{HausdorffCompletion}^{\uparrow}$

The $\mathsf{HausdorffCompletion}^{\uparrow}$ interface records the separated completion seal used when completion data must remain visible as named finite rows rather than becoming quotient-by-class data. Following Bourbaki, *General Topology*, and Bishop and Bridges, *Constructive Analysis*, the packet treats separatedness as displayed seal evidence over Cauchy and uniform data. It depends on $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

The separated-completion packet consumes the regular-sequence source of chapter 301, the uniform entourage route of chapter 339, and the diagonal handoff of chapter 378 before exposing its local Hausdorff seal.

Definition 402.1 (HausdorffCompletion carrier)**.** *A* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, U, A, M, L, C, P, N)$$

with the following visible rows:

1. *S is a* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row;*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route controlling the compared source windows;*
3. *A is the* $\mathsf{Real}^{\uparrow}$ *seal-facing separated classifier row;*
4. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *diagonal handoff row;*
5. *L records the displayed* hsame *transports for source, entourage, separated-seal, and completion-handoff rows;*
6. *C records the displayed* Cont *routes through the source, entourage, separated classifier, and handoff ledgers;*
7. *P is the* Pkg *provenance over the visible dependency certificates;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row.*

The carrier contains no quotient class of Cauchy filters, no host completion object, no selected representative, and no ambient completeness principle.

Theorem 402.2 (HausdorffCompletion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier habitation by a displayed* CauchyFilter$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *source, a* UniformSpace$^{\uparrow}$ *entourage route, a separated* Real$^{\uparrow}$ *classifier row, and a* CauchyCompletionMonad$^{\uparrow}$ *handoff;*
2. *separated classifier stability under the stored* hsame *transports and displayed* Cont *routes;*
3. *Cauchy-filter handoff from the source row through the entourage route into the completion handoff row;*
4. *completion-seal non-escape, stating that the* Real$^{\uparrow}$ *read is reached only through the separated classifier and the finite handoff ledger;*
5. *ledger exactness for the* Pkg *provenance and local* NameCert *row.*

These rows name separated completion data without turning the packet into a quotient of Cauchy classes or an ambient complete space.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionPacket namecert obligations`

Proof. Carrier habitation is projection of Definition 402.1. The source, entourage, separated classifier, handoff, transport, continuation, package, and naming rows are finite coordinates of the same BHist packet.

Classifier stability is componentwise. Move the source row, entourage route, separated seal row, and completion handoff along the stored hsame transports, then replay the displayed Cont routes for the same coordinates.

The Cauchy-filter handoff uses S, U, and M. A consumer first reads the Cauchy or regular rational source, then the entourage route controlling the displayed windows, and only then the completion handoff row.

The non-escape and exactness rows follow from the absence of any quotient-class, host-completion, selected-representative, or ambient-completeness coordinate. The package and local naming rows are therefore the complete provenance surface for the separated seal. □

Theorem 402.3 (HausdorffCompletion separated seal boundary)**.** *Every* Real$^{\uparrow}$ *seal-facing read exported by an accepted* HausdorffCompletion$^{\uparrow}$ *carrier factors through the* CauchyFilter$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *source row, the* UniformSpace$^{\uparrow}$ *entourage route, the separated classifier row, the* CauchyCompletionMonad$^{\uparrow}$ *diagonal handoff, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows. Thus separated completion is exported as finite seal data, not as quotient-by-class data, host completion equality, selected representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated seal boundary`

Proof. Start with an accepted carrier $H = (S, U, A, M, L, C, P, N)$. The only source coordinate is S, and the only uniform-control coordinate is the entourage route U.

Use the NameCert obligations of Definition 402.2. They force the separated classifier and completion handoff to remain attached to the same source, entourage, transport, continuation, package, and naming rows.

The Real$^{\uparrow}$ read reaches the separated classifier row A and the completion handoff row M only through that finite route. Since the carrier has no quotient, host completion, selected representative, or ambient completeness coordinate, the listed rows exhaust the public seal boundary. □

Theorem 402.4 (HausdorffCompletion uniform RegSeqRat seal coherence)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted HausdorffCompletion$^{\uparrow}$ *carrier whose source row S is a* RegSeqRat$^{\uparrow}$ *source. Then the uniform entourage route U, the separated* Real$^{\uparrow}$ *seal row A, the* CauchyCompletionMonad$^{\uparrow}$ *handoff row M, and the displayed support rows L, C, P, N cohere as one finite separated ledger over the source S.*

Every consumer of this ledger reaches the separated Real seal only through the displayed route

$$S, \quad U, \quad A, \quad M, \quad L, \quad C, \quad P, \quad N.$$

There is no direct RegSeqRat-to-Real completion edge through quotient classes, selected representatives, host completion equality, uniqueness-of-limit data, or ambient completeness.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier uniform regseqrat seal coherence`

Proof. Project the carrier through Definition 402.1. The source, uniform entourage, separated seal, handoff, transport, continuation, package, and local naming rows are the only displayed coordinates.

Use Definition 402.2. Its Cauchy-filter handoff and ledger rows attach the RegSeqRat$^{\uparrow}$ source to the entourage route, the completion handoff, and the separated seal through the same L, C, P, N support.

Apply Definition 402.3 and Definition 402.5. The seal-facing read and any classified replacement remain on the displayed route through S, U, A, M, L, C, P, N.

Finally use Definition 402.6. Since the ledger exports no quotient class, selected representative, host completion equality, uniqueness-of-limit datum, or ambient-completeness coordinate, the RegSeqRat source has no direct completion edge outside the finite separated ledger. □

Theorem 402.5 (HausdorffCompletion classifier transport)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted HausdorffCompletion$^{\uparrow}$ *carrier. Any classified replacement transports the source row S, the uniform entourage route U, the separated* Real$^{\uparrow}$ *classifier row A, the completion handoff M, the displayed* hsame *rows L, the* Cont *routes C, the* Pkg *provenance P, and the local* NameCert *row N componentwise. The transported packet is accepted only through those displayed rows, and the classifier cannot compare quotient classes, selected representatives, host completion equality, or ambient completeness evidence.*

Proof. Project the carrier of Definition 402.1. The source, entourage, separated seal, handoff, transport, continuation, package, and naming rows are the full list of displayed coordinates.

Use the stability row in Definition 402.2. It moves the separated classifier and completion handoff along the stored hsame transports and then replays the displayed Cont routes for the same coordinates.

The separated seal boundary of Definition 402.3 keeps the transported read attached to the source, entourage, handoff, package, and naming rows. Since the carrier has no quotient, representative, host completion, or ambient completeness coordinate, the classifier transport has no additional comparison source. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier classifier transport`

Theorem 402.6 (HausdorffCompletion ledger exactness)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier, the* Pkg *provenance row and the local* NameCert *row exhaust the public separated-completion surface. They record exactly the source row, uniform entourage route, separated* $\mathsf{Real}^{\uparrow}$ *classifier row, completion handoff, displayed* hsame *transports, and* Cont *routes. They cannot export quotient Cauchy classes, selected representatives, host completion equality, uniqueness of limits, or ambient completeness.*

Proof. Read the package and local certificate coordinates P and N from Definition 402.1. They name the visible dependency rows and the local separated-completion certificate surface.

Apply Definition 402.2. Its exactness row states that the package and naming data are the complete provenance surface for the separated seal.

Finally use Definition 402.3. Every exported $\mathsf{Real}^{\uparrow}$ seal-facing read factors through the same finite route, so no quotient class, selected representative, host completion equality, uniqueness-of-limit datum, or ambient completeness principle can be exported by the ledger. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier ledger exactness`

Theorem 402.7 (HausdorffCompletion uniform-entourage Cauchy handoff)**.** *Every separated-completion consumer read of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier routes through the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, then through the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route, and only then through the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row. The displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows record this order, so the consumer has no direct source-to-completion edge through quotient classes, selected representatives, host completion equality, uniqueness of limits, or ambient completeness.*

Proof. Project the carrier coordinates of Definition 402.1. The source row S, entourage route U, completion handoff row M, transport row L, continuation row C, package row P, and naming row N are displayed coordinates of the same finite packet.

Use the Cauchy-filter handoff clause of Definition 402.2. It places any source read through the entourage route before the completion handoff can be consumed.

Finally apply Definition 402.6. The package and local certificate expose exactly those rows and no quotient-completion, representative-selection, host-equality, limit-uniqueness, or ambient-completeness coordinate, so the listed handoff order exhausts the consumer route. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier uniform entourage cauchy handoff`

Theorem 402.8 (HausdorffCompletion real-seal ledger scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier is exhausted by the separated classifier row together with the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *provenance. The read cannot cite a quotient class of Cauchy filters, a selected representative, host completion equality, uniqueness of limits, or ambient completeness outside that finite ledger.*

Proof. Start with the separated classifier coordinate A in the carrier

$$H = (S, U, A, M, L, C, P, N).$$

The definition places the real-facing seal at A and supplies L, C, P, and N as the only transport, continuation, package, and naming coordinates around it.

Apply the completion-seal non-escape and ledger-exactness rows from Definition 402.2. They force the real-facing read to remain attached to the separated classifier and to the displayed provenance rows.

Use Definition 402.3 to exclude any unlisted route. Since the carrier has no quotient class, selected representative, host equality, limit-uniqueness, or ambient-completeness coordinate, the separated classifier ledger is the complete scope of every $\mathsf{Real}^{\uparrow}$ seal-facing read. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier real seal ledger scope`

Theorem 402.9 (HausdorffCompletion obligation unblock package)**.** *The carrier of Definition 402.1, classifier transport of Definition 402.5, uniform-entourage Cauchy handoff of Definition 402.7, real-seal ledger scope of Definition 402.8, and ledger exactness of Definition 402.6 assemble the five-row* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *obligation surface. The assembled package names exactly carrier habitation, classifier stability, Cauchy handoff, completion-seal non-escape, and ledger exactness over the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. First read carrier habitation from Definition 402.1. It supplies the source, entourage, separated classifier, completion handoff, transport, continuation, package, and local naming rows.

Classifier stability is supplied by Definition 402.5. The theorem keeps replacements componentwise over the displayed source, entourage, separated classifier, completion handoff, transport, continuation, package, and naming rows.

The handoff row is Definition 402.7. The seal row is Definition 402.8. These two results place consumer reads through the Cauchy source, uniform entourage, completion handoff, separated classifier, and displayed provenance routes.

Finally apply Definition 402.6. It identifies the package and local certificate as the complete provenance surface, so the five NameCert rows have no remaining slot for quotient completion, host equality, selected representatives, uniqueness of limits, or ambient completeness. □

Theorem 402.10 (HausdorffCompletion Cauchy-entourage handoff exactness)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier. Every accepted carrier handoff from a* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source into the completion row factors through the source row S, the entourage route U, the completion handoff row M, the displayed* hsame *transports L, the displayed* Cont *routes C, the package provenance P, and the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row N. The handoff has no direct edge from a Cauchy source to a quotient class, host completion object, selected representative, uniqueness-of-limit datum, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier cauchy entourage handoff exactness`

Proof. Project the carrier coordinates from Definition 402.1. The only source coordinate is S, the only uniform control coordinate is U, and the only completion-facing handoff coordinate is M.

Apply the Cauchy-filter handoff row of Definition 402.2. It sends any source read through the displayed entourage route before the completion handoff row is consumed.

Use Definition 402.3 to keep the handoff attached to the separated classifier route, and use Definition 402.6 to bind the route to L, C, P, N. Since the carrier contains no quotient, representative, host completion, limit-uniqueness, or ambient-completeness coordinate, no other handoff can be exported. □

Theorem 402.11 (HausdorffCompletion separated ledger transport scope)**.** *For an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier*

$$H = (S, U, A, M, L, C, P, N),$$

any classified replacement preserves the separated $\mathsf{Real}^{\uparrow}$ *classifier row and the completion handoff only by transporting the displayed source, entourage, classifier, handoff,* Pkg*, and* NameCert *rows along the stored* hsame *ledger* L *and replaying the displayed* Cont *routes* C*. The transport does not compare quotient Cauchy classes, selected representatives, host completion equality, uniqueness of limits, or ambient completeness evidence.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated ledger transport scope`

Proof. Read the displayed rows from Definition 402.1. The separated classifier and the completion handoff are coordinates of the same finite packet, so a classified replacement must move them with the surrounding source and entourage rows.

Apply Definition 402.5. It states that replacement is componentwise over S, U, A, M, L, C, P, N and that acceptance is recovered only through the displayed transport and continuation rows.

The separated seal boundary of Definition 402.3 and ledger exactness of Definition 402.6 rule out the unlisted comparison sources. Thus the separated classifier and completion handoff are preserved precisely on the displayed L and C transport scope. □

Theorem 402.12 (HausdorffCompletion separated source identity criterion)**.** *Let*

$$H = (S, U, A, M, L, C, P, N) \quad \textit{and} \quad H' = (S', U', A', M', L', C', P', N')$$

be accepted $\mathsf{HausdorffCompletion}^{\uparrow}$ *packets. The two packets expose the same separated completion identity only when their* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows agree, their* $\mathsf{UniformSpace}^{\uparrow}$ *entourage routes agree, their* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff rows agree, their* $\mathsf{Real}^{\uparrow}$ *separated classifier rows agree, and their provenance rows* L, C, P, N *agree componentwise with* L', C', P', N'*. No quotient Cauchy class, selected representative, host completion equality, uniqueness-of-limit datum, or ambient completeness witness can substitute for these displayed agreements.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated source identity criterion`

Proof. Begin with classifier transport, Definition 402.5. It identifies accepted replacement as componentwise transport over the displayed source, entourage, classifier, handoff, transport, continuation, package, and local naming rows.

Use the uniform-entourage Cauchy handoff and Cauchy-entourage handoff exactness, Definition 402.7 and Definition 402.10. These force the completion-facing identity read to pass through the source row, the entourage route, and the completion handoff row before the separated classifier is exported.

Finally apply the separated seal boundary and ledger exactness, Definition 402.3 and Definition 402.6. They bind the exposed identity to L, C, P, N and exclude hidden comparison routes, so sameness of the separated completion identity is exactly sameness of the displayed source, entourage, handoff, classifier, and provenance rows. □

Theorem 402.13 (HausdorffCompletion separated package exhaustion)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier, the* Pkg *row and the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *provenance exhaust the separated completion surface. They expose only the source row, entourage route, separated classifier row,*

completion handoff, displayed hsame *transports, and displayed* Cont *routes, and they export no quotient Cauchy class, selected representative, host equality, uniqueness-of-limit datum, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated package exhaustion`

Proof. Begin with the package and local naming coordinates P and N in Definition 402.1. They are provenance rows for the displayed source, entourage, classifier, handoff, transport, and continuation data.

Apply the exactness row of Definition 402.2. It identifies P and N as the complete provenance surface for the separated seal.

Finally use Definition 402.6, together with the classifier transport theorem Definition 402.5. The exported separated completion packet is exhausted by the displayed rows, so quotient classes, representatives, host equality, limit uniqueness, and ambient completeness never appear as package output. □

Theorem 402.14 (HausdorffCompletion separated reflection)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier. Every separated reflection consumer reads only the displayed Cauchy or regular-rational source S, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route U, the completion handoff M, the* $\mathsf{Real}^{\uparrow}$ *separated classifier row A, the* hsame *transports L, the* Cont *routes C, the* Pkg *provenance P, and the local* NameCert *row N. The reflection cannot import quotient completion classes, host completion equality, selected representatives, ambient completeness, or uniqueness outside the displayed classifier ledger.*

Proof. Project the finite carrier of Definition 402.1. It lists exactly the source, entourage, separated classifier, handoff, transport, continuation, package, and local naming rows.

Use classifier transport and the separated seal boundary. These force the reflected separated row to move along L and C while staying attached to the same source, entourage, handoff, and classifier coordinates.

Finally apply separated package exhaustion and ledger exactness. Since P and N exhaust the exported package, the reflected consumer has no further coordinate through which quotient completions, host equality, selected representatives, ambient completeness, or outside uniqueness could enter. □

Theorem 402.15 (HausdorffCompletion Cauchy source route scope)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier. Every separated-completion consumer route from a* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source is scoped by the displayed path from S through the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row U, then through the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff M, and then into the separated* $\mathsf{Real}^{\uparrow}$ *classifier row A, with L, C, P, N carrying transport, continuation, provenance, and local naming.*

Proof. Read the route from the carrier. The only source coordinate is S, the only uniform-control coordinate is U, and the only completion-facing handoff coordinate is M.

Apply the Cauchy-entourage handoff exactness theorem and the Cauchy-filter seal route. They place every source read on the route through S, U, and M before a separated classifier can be consumed.

Use separated ledger transport scope and separated package exhaustion to attach the route to L, C, P, N. The carrier and ledger exactness theorems leave no alternate source route through quotient Cauchy classes, host equality, selected representatives, ambient completeness, or an independent limit-uniqueness principle. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier cauchy source route scope`

Theorem 402.16 (HausdorffCompletion Real seal nonquotient boundary)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted HausdorffCompletion$^{\uparrow}$ *carrier. Every* Real$^{\uparrow}$ *seal-facing read is the separated classifier row A reached through the displayed Cauchy or regular-rational source S,* UniformSpace$^{\uparrow}$ *entourage route U,* CauchyCompletionMonad$^{\uparrow}$ *handoff M,* hsame *transports L,* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N. It is not a quotient-completion equality, host real equality, selected representative, or ambient completeness claim.*

Proof. Start from the separated seal boundary. It identifies the real-facing endpoint with the displayed classifier row A inside the finite carrier.

Apply classifier transport, Cauchy-entourage handoff exactness, and separated ledger transport scope. Together they bind the path into A to the displayed source, entourage, completion handoff, transport, and continuation rows.

Finally use separated package exhaustion and ledger exactness. The package and local certificate rows exhaust the exported Real seal surface, so no quotient-completion equality, host real equality, selected representative, or ambient-completeness coordinate can be read by separated reflection consumers. □

Theorem 402.17 (HausdorffCompletion separated reflection)**.** *Let*

$$H = (S, U, A, M, L, C, P, N)$$

be an accepted HausdorffCompletion$^{\uparrow}$ *carrier. If a separated classifier row is reflected through the displayed* hsame *transports L, the displayed* Cont *routes C, the package row P, and the local* NameCert *row N, then the reflected read preserves exactly the finite packet coordinates S, U, A, M, L, C, P, and N. The reflection has no quotient class, host completion equality, selected representative, uniqueness-of-limit datum, or ambient completeness coordinate from which to obtain additional separated-completion data.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated reflection`

Proof. Project the carrier of Definition 402.1. The packet exposes only the source row, entourage route, separated classifier row, completion handoff, displayed transport rows, continuation routes, package provenance, and local naming row.

Apply the stability and exactness rows of Definition 402.2. The separated classifier may move through hsame and Cont only along the stored rows, while P and N keep the provenance attached to those same coordinates.

Use the separated seal boundary of Definition 402.3 and the classifier transport theorem Definition 402.5. They identify the reflected classifier with the same finite route through S, U, A, M, L, C, P, and N.

Finally apply ledger exactness from Definition 402.6. Since the ledger exports no quotient, representative, host-completion, limit-uniqueness, or ambient-completeness coordinate, the reflected separated row preserves the packet and nothing beyond it. □

Theorem 402.18 (HausdorffCompletion Cauchy-filter seal route)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier, any source read from the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate reaches the* $\mathsf{Real}^{\uparrow}$ *separated seal only through the displayed* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route and the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff. No direct source-to-seal edge, quotient-completion edge, host-completion edge, selected-representative edge, or ambient-completeness edge is part of the packet.*

Proof. Begin with the carrier coordinates of Definition 402.1. The source coordinate is S, the uniform-control coordinate is U, and the completion handoff coordinate is M.

Apply the Cauchy-filter handoff and completion-seal non-escape rows from Definition 402.2. A source read must pass through the displayed entourage route before it reaches the handoff ledger, and the separated $\mathsf{Real}^{\uparrow}$ read is attached to that handoff.

The separated seal boundary in Definition 402.3 keeps the seal-facing read inside the same finite route. The classifier transport theorem Definition 402.5 shows that moving the classifier cannot introduce a different source-to-seal path.

Ledger exactness from Definition 402.6 removes the remaining escape cases. The package and local certificate record the displayed route through S, U, and M, and they export no quotient, host-completion, representative, or ambient-completeness edge. □

Theorem 402.19 (HausdorffCompletion uniform consumer package)**.** *Any downstream* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier receives exactly the carrier coordinates listed in Definition 402.1: the source row, uniform entourage route, separated classifier row, completion handoff, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The consumer receives no quotient class, host completion object, selected representative, uniqueness-of-limit witness, or ambient completeness principle.*

Proof. Read the public consumer interface from the carrier definition. Its coordinates are the eight rows S, U, A, M, L, C, P, and N, and the definition explicitly excludes quotient classes, host completion objects, selected representatives, and ambient completeness principles.

Use the NameCert obligations of Definition 402.2. They bind carrier habitation, classifier stability, source handoff, seal non-escape, and ledger exactness to those eight rows.

The separated boundary and classifier transport theorems, Definition 402.3 and Definition 402.5, show that $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers can read the packet only along the displayed entourage, handoff, classifier, transport, continuation, package, and naming routes.

Finally apply Definition 402.6. The package and local certificate exhaust the exported surface, so every downstream consumer receives exactly the carrier coordinates and no outside completion data. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier uniform consumer package`

Theorem 402.20 (HausdorffCompletion entourage-filter coherence)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier*

$$H = (S, U, A, M, L, C, P, N),$$

each accepted carrier handoff reads the source row S, then the displayed $\mathsf{UniformSpace}^{\uparrow}$ *entourage route U, then the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row M before any separated* $\mathsf{Real}^{\uparrow}$ *seal is consumed. No separated seal can be read directly from S, from a quotient Cauchy class, from a host completion equality, from a selected representative, or from an ambient completeness coordinate.*

Proof. Project the carrier of Definition 402.1. The only source-facing route begins at S, the only uniform-control route is U, and the only completion handoff route is M.

Apply the Cauchy-filter handoff and seal non-escape rows of Definition 402.2. These rows force a source read to pass through the displayed entourage route before the completion handoff can be presented to a separated consumer.

Use Definition 402.5 to check that changing the separated classifier moves only along the displayed L and C rows. Then apply Definition 402.6. The ledger has no quotient, representative, host-equality, uniqueness-of-limit, or ambient-completeness coordinate, so the handoff route is exactly S through U through M before the seal is consumed. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier entourage filter coherence`

Theorem 402.21 (HausdorffCompletion separated transport certificate)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier, any classified replacement of the separated row preserves the separated classifier coordinate A and the completion handoff coordinate M only through the displayed* hsame *transport rows L and the displayed* Cont *continuation rows C. The certificate does not preserve A or M by quotient equality, selected representatives, host completion equality, uniqueness of limits, or ambient completeness.*

Proof. Start from the finite carrier in Definition 402.1. The separated classifier row A and handoff row M occur in the packet together with the displayed transport row L and continuation row C.

The NameCert obligation surface in Definition 402.2 binds classifier stability and completion handoff to those displayed rows. It gives no separate preservation principle outside the packet.

Apply Definition 402.5. Classified replacement transports the relevant separated and handoff reads componentwise along L and routes their public consumption through C.

Finally use Definition 402.6. Since the ledger exports only the displayed carrier rows and package provenance, there is no hidden quotient, representative, host-completion, limit-uniqueness, or ambient-completeness source from which a stronger preservation claim could be read. □

Theorem 402.22 (HausdorffCompletion consumer handoff exhaustion)**.** *Every downstream* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier receives exactly the rows*

$$S, U, A, M, L, C, P, N$$

from the displayed packet. The handoff exports no quotient completion class, selected representative, host equality, uniqueness-of-limit witness, ambient-completeness principle, or unlisted completion datum.

Proof. Read the consumer packet from Definition 402.1. Its public rows are precisely the source row, entourage route, separated classifier row, completion handoff, displayed transport, displayed continuation, package provenance, and local naming row.

Apply Definition 402.2. The obligation rows attach carrier habitation, classifier stability, source handoff, separated non-escape, and ledger exactness to the same displayed packet.

Use Definition 402.5 for the separated replacement route and Definition 402.6 for the exported ledger. Together they show that the three downstream consumers can consume exactly S, U, A, M, L, C, P, and N, with no further completion-side coordinate. □

Theorem 402.23 (HausdorffCompletion Cauchy-filter NameCert obligations)**.** *Every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *Cauchy-filter or regular-rational source read must pass through the source row* S*, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route* U*, the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row* M*, and the displayed* Pkg *and* $\mathsf{NameCert}$ *provenance rows before it is consumed as a separated completion datum. The obligation surface provides no alternate read through quotient Cauchy classes, host completion equality, selected representatives, uniqueness of limits, or ambient completeness.*

Proof. Start from the carrier packet of Definition 402.1. Its source-facing coordinate is S, its uniform-control route is U, its completion handoff is M, and its public provenance is carried by P and N.

Apply the Cauchy-filter handoff and ledger rows of Definition 402.2. They attach every source read to the same finite route and bind the route to the displayed package and local certificate.

Use the separated seal boundary of Definition 402.3 to keep the separated endpoint inside the same packet. The consumer package of Definition 402.19 then shows that downstream reads receive only the listed rows, so the excluded host and quotient data have no obligation field. □

Theorem 402.24 (HausdorffCompletion separated limit uniqueness boundary)**.** *Whenever separated limit uniqueness is consumed from an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *packet, the consumed uniqueness is only the finite separated* $\mathsf{Real}^{\uparrow}$ *classifier row* A *transported by the displayed* hsame *rows* L *and* Cont *rows* C*. It is not host equality, quotient equality, a representative choice, an ambient completeness assertion, or an unrestricted uniqueness-of-limit principle.*

Proof. Read the separated component from Definition 402.1. The only separated classifier coordinate is A, and the only transport and continuation coordinates available to move it are L and C.

The NameCert obligations of Definition 402.2 bind classifier stability and ledger exactness to those displayed rows. The separated seal boundary of Definition 402.3 prevents a consumer from replacing that finite classifier route with host completion equality or an outside representative.

Finally apply the uniform consumer package of Definition 402.19. Since every downstream consumer receives the same finite packet S, U, A, M, L, C, P, N, any uniqueness read available to it is the transported separated classifier row, not an additional completion principle. □

Theorem 402.25 (HausdorffCompletion Real and RegSeqRat consumer package)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *packet receives exactly the rows*

$$S, U, A, M, L, C, P, N$$

from the displayed carrier. The consumer package exports no quotient completion class, host completion object, selected representative, host equality, uniqueness-of-limit witness, or ambient completeness row.

Proof. Project the finite packet of Definition 402.1. Its complete public surface is the source row, uniform route, separated classifier, completion handoff, displayed transport, displayed continuation, package provenance, and local certificate.

Use Definition 402.2 to attach carrier habitation, classifier stability, handoff, non-escape, and ledger exactness to those rows. The separated seal boundary of Definition 402.3 keeps the real-facing endpoint inside the same packet.

Apply the uniform consumer package of Definition 402.19. It exhausts the downstream $\mathsf{Real}^{\uparrow}$ and regular-rational consumption route by S, U, A, M, L, C, P, N, leaving no coordinate for any excluded host or quotient completion datum. □

Theorem 402.26 (HausdorffCompletion scoped kernel interface)**.** *The separated reflection, Cauchy-filter seal route, uniform consumer package, classifier transport, and ledger exactness rows of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier are scoped exactly by the displayed* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Within that scope the carrier exposes only the finite packet*

$$H = (S, U, A, M, L, C, P, N)$$

and its named route data; it does not invoke a host quotient completion, host completion equality, a choice-selected representative, uniqueness of limits, or an ambient completeness principle.

Proof. Start from the carrier definition and the five NameCert obligation rows. They place every accepted packet in a finite BHist carrier with the displayed source, uniform, separated-seal, handoff, transport, continuation, package, and naming coordinates.

Apply the Cauchy-entourage handoff exactness theorem, then the separated ledger transport and separated package exhaustion theorems. These results bind the handoff, transport, and provenance rows to the displayed $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ dependencies.

Next use separated reflection, the Cauchy-filter seal route, classifier transport, and the uniform consumer package theorem. They show that every separated read and every downstream consumer route remains inside the same finite packet coordinates.

Finally apply ledger exactness. Since the ledger exports no quotient class, host completion object, choice-selected representative, limit-uniqueness witness, or ambient-completeness row, the displayed kernel rows are the exact scoped interface. □

Theorem 402.27 (HausdorffCompletion public certificate)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier*

$$H = (S, U, A, M, L, C, P, N),$$

the carrier row, classifier transport row, Cauchy-filter seal route, separated package exhaustion row, and scoped kernel interface assemble the public $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *export over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The export exposes exactly the displayed source, entourage, separated classifier, completion handoff, transport, continuation, package, and local naming rows, and it exports no quotient completion object, host completion equality, selected representative, uniqueness-of-limit witness, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier public certificate`

Proof. The carrier definition supplies the finite packet $H = (S, U, A, M, L, C, P, N)$ and identifies the source, uniform entourage, separated classifier, completion handoff, displayed transport, continuation, package, and naming rows.

The NameCert obligations give the five public clauses: carrier habitation, classifier stability, Cauchy-filter handoff, completion-seal non-escape, and ledger exactness. The classifier transport theorem then moves the separated classifier and completion handoff only along the displayed hsame and Cont rows.

The Cauchy-filter seal route and separated package exhaustion theorems identify the consumer-facing route from S through U and M into the separated $\mathsf{Real}^{\uparrow}$ seal, with P and N as the exhaustive provenance rows. The uniform consumer package records the same route for downstream $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers.

Finally apply the scoped kernel interface. It binds all listed rows to the displayed $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert dependencies, so the public certificate has no remaining coordinate from which quotient completion, host equality, selected representatives, limit uniqueness, or ambient completeness could be exported. □

Theorem 402.28 (HausdorffCompletion separated reflector route)**.** *Every separated reflector read of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route, the* $\mathsf{Real}^{\uparrow}$ *separated seal row, the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row, the componentwise classifier transport, the displayed* hsame *and* Cont *continuation rows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row. The reflector has no independent route through quotient Cauchy classes, host completion equality, selected representatives, uniqueness of limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier separated reflector route`

Proof. Begin with the accepted carrier coordinates. The carrier theorem provides the source row, entourage route, separated seal, handoff, transport ledger, continuation ledger, package provenance, and local NameCert row as the complete displayed packet.

Use the NameCert obligations to keep the separated read attached to carrier habitation, classifier stability, Cauchy-filter handoff, seal non-escape, and ledger exactness. The classifier transport theorem supplies the componentwise movement of the reflector through the stored hsame transports and displayed Cont routes.

Next apply the Cauchy-filter seal route. It forces every source read to pass through the entourage route and completion handoff before it reaches the separated $\mathsf{Real}^{\uparrow}$ seal.

The separated package exhaustion and uniform consumer package theorems place the reflector output under the package and local NameCert rows. The scoped kernel interface then rules out any unlisted quotient, representative, host-completion, limit-uniqueness, or ambient-completeness route. □

Theorem 402.29 (HausdorffCompletion Cauchy-filter uniqueness boundary)**.** *Inside an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier, Cauchy-filter uniqueness is only classifier-separated uniqueness in the displayed ledger: two source reads are identified for public separated-completion use only when the* $\mathsf{Real}^{\uparrow}$ *separated classifier, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage route, the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff, the stored* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row certify the same finite route. This boundary does not assert quotient completion equality, selected representatives, host completion equality, ambient completeness, or uniqueness of limits outside the displayed classifier ledger.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier cauchy filter uniqueness boundary`

Proof. Read the uniqueness claim inside the carrier $H = (S, U, A, M, L, C, P, N)$. The only public source data are the $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ row S and the displayed entourage route U leading to the handoff row M.

The NameCert obligations and classifier transport theorem determine how the separated classifier row A can be transported and consumed. They permit comparison only through the stored hsame transports, displayed Cont routes, package row P, and local naming row N.

Use the Cauchy-filter seal route to bind source reads to the same route through S, U, and M. Use separated package exhaustion and the uniform consumer package to keep the public output inside P and N.

Finally apply the scoped kernel interface. Since the carrier contains no quotient completion object, selected representative, host completion equality, ambient completeness principle, or independent uniqueness-of-limit coordinate, the only uniqueness available to the public package is the classifier-separated uniqueness certified by the displayed ledger. □

Theorem 402.30 (HausdorffCompletion public route composition)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier*

$$H = (S, U, A, M, L, C, P, N),$$

the public separated read composes exactly through the source row S, the $\mathsf{UniformSpace}^{\uparrow}$ *entourage route U, the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff M, the separated* $\mathsf{Real}^{\uparrow}$ *classifier A, and the provenance packet L, C, P, N. No public read has an additional coordinate through quotient Cauchy classes, selected representatives, host completion equality, limit uniqueness outside A, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier public route composition`

Proof. Project the accepted carrier. The carrier theorem and NameCert obligations expose S, U, A, M, L, C, P, and N as the finite displayed rows, with carrier habitation, classifier stability, handoff, non-escape, and ledger exactness attached to those rows.

Next use classifier transport and the Cauchy-filter seal route. They force any separated read to move from S through U and M before it reaches the classifier row A, and they allow transport only along the displayed ledger rows L and C.

Finally apply the uniform consumer package and the scoped kernel interface. These rows place the public read under the package provenance P and local certificate N, and the Cauchy-filter uniqueness boundary keeps every identification inside the displayed separated classifier ledger. □

Theorem 402.31 (HausdorffCompletion downstream exactness package)**.** *The downstream* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier receive only the displayed packet*

$$(S, U, A, M, L, C, P, N)$$

through the public route composition of Definition 402.30. Their consumer surface contains the source row, entourage route, separated classifier, completion handoff, transport ledger, continuation ledger, package provenance, and local naming row, and it contains no quotient completion object, selected representative, host completion equality, ambient completeness datum, or independent limit-uniqueness witness.

Proof. Start with the uniform consumer package. It already names the $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumption route and records the consumer-facing coordinates S, U, A, M, L, C, P, and N.

Use the public route composition theorem to identify that consumer route with the exact separated read route through S, U, M, and A. The classifier transport row accounts for L and C, while separated package exhaustion accounts for P and N.

The scoped kernel interface and Cauchy-filter uniqueness boundary then remove every possible unlisted source of downstream data. Thus the consumers receive precisely the displayed packet and no host quotient, representative, equality, completeness, or external uniqueness component. □

Theorem 402.32 (HausdorffCompletion public separated certificate)**.** *The public route composition of Definition 402.30, the downstream exactness package of Definition 402.31, and the scoped kernel interface of Definition 402.26 assemble the public separated-completion certificate for* $\mathsf{HausdorffCompletion}^{\uparrow}$*. The certificate is exactly the finite* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *packet over* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, with no claim of quotient completion, selected representatives, uniqueness of limits outside the classifier ledger, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier public separated certificate`

Proof. First assemble the route row. The public route composition fixes the only separated read path through S, U, M, A, and the provenance rows L, C, P, N.

Next assemble the consumer row. The downstream exactness package shows that $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers receive that same displayed packet and no extra host-side completion data.

Finally attach the scoped kernel interface. It binds the route and consumer rows to the carrier, classifier transport, uniform consumer package, and Cauchy-filter uniqueness rows, so the resulting public separated-completion certificate is exactly the displayed finite $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ surface. □

Definition 402.33 (HausdorffCompletion standard separated reading)**.** *The standard separated-completion reading of an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *packet is the finite BEDC route*

$$(S, U, A, M, L, C, P, N)$$

read as follows: S *is the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* U *is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage-control row,* A *is the separated* $\mathsf{Real}^{\uparrow}$ *classifier row,* M *is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row,* L *is the displayed* BHist/hsame *transport ledger,* C *is the displayed* Cont *continuation ledger,* P *is the package-provenance row, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row. The reading contains exactly these rows and treats separated completion as classifier-separated export along this finite route.*

Theorem 402.34 (HausdorffCompletion BEDC-to-standard separated bridge)**.** *Every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *packet exports the standard separated-completion reading of Definition 402.33. The exported object is exactly the finite route*

$$(S, U, A, M, L, C, P, N)$$

with Cauchy or regular-rational source data, uniform entourage control, separated $\mathsf{Real}^{\uparrow}$ *classifier, completion-monad handoff, transport ledger, continuation ledger, package provenance, and local naming row. It exports no quotient Cauchy class, selected representative, host equality, ambient completeness principle, or independent host completion object.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionUp StdBridge`

Proof. Start with an accepted carrier $H = (S, U, A, M, L, C, P, N)$. The public separated certificate identifies these eight rows as the complete public $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ packet.

The public route composition orders the source row, entourage route, completion handoff, separated classifier, and provenance rows exactly as the standard separated reading requires. The downstream exactness package shows that every consumer receives the same rows and no additional completion data.

Finally use the Cauchy-filter uniqueness boundary and the scoped kernel interface. They rule out quotient classes, selected representatives, host equality, ambient completeness, and any host completion object outside the displayed classifier ledger. Thus the BEDC packet exports precisely the standard separated-completion reading. □

Theorem 402.35 (HausdorffCompletion standard-to-BEDC separated bridge)**.** *Any standard separated-completion reading with displayed rows*

$$(S, U, A, M, L, C, P, N)$$

where S is a $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, U is a* $\mathsf{UniformSpace}^{\uparrow}$ *entourage-control row, A is a separated* $\mathsf{Real}^{\uparrow}$ *classifier row, M is a* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row, L is a* $\mathsf{BHist/hsame}$ *transport ledger, C is a* Cont *continuation ledger, P is package provenance, and N is a local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row repacks as the public* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion certificate of Definition 402.32. The repacked certificate has no further quotient, representative, host-equality, or ambient-completeness coordinate.*

Proof. Read the standard separated-completion input as the eight displayed rows. The definition fixes the same row order and the same consumer interpretation as the public route composition.

Use the downstream exactness package to route the $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers through those rows. The transport and continuation ledgers provide the visible hsame and Cont data, while P and N provide package provenance and local naming.

Attach the scoped kernel interface and the public separated certificate. Since the standard reading has no additional quotient, representative, host-equality, or ambient-completeness field, the repacked object is exactly the existing public $\mathsf{HausdorffCompletion}^{\uparrow}$ certificate. □

Theorem 402.36 (HausdorffCompletion standard bridge source packet)**.** *For every accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier*

$$H = (S, U, A, M, L, C, P, N),$$

the bridge-facing source packet is exactly the displayed separated-completion row family: the $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row S, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row U, the separated* $\mathsf{Real}^{\uparrow}$ *classifier row A, the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *handoff row M, the transport and continuation rows L, C, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row N. This packet is the only bridge source offered by the public separated certificate.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier standard bridge source packet`

Proof. Start with the carrier inventory of Definition 402.1. It displays S, U, A, M, L, C, P, N as the complete finite tuple and assigns each coordinate to the corresponding source, entourage, separated classifier, handoff, transport, continuation, package, and local naming role.

Apply the public certificate of Definition 402.27. It shows that the public read of the carrier is already exhausted by the same row family, with no additional completion object or representative row.

Finally use the public separated certificate of Definition 402.32. The separated-completion reading consumed by a bridge is therefore the public packet itself, not a host-side completion carrier supplied outside H. □

Theorem 402.37 (HausdorffCompletion bridge non-escape boundary)**.** *Every standard separated-completion bridge read of the source packet in Definition 402.36 factors through the rows S, U, A, M, L, C, P, N. It cannot read quotient Cauchy classes, host completion equality, selected representatives, uniqueness outside the separated classifier ledger, or ambient completeness.*

Proof. The bridge source packet is the finite row family S, U, A, M, L, C, P, N. The carrier definition supplies no coordinate for quotient classes, selected representatives, host equality, uniqueness outside A, or an ambient completeness principle.

The public separated certificate identifies the separated read with the route through S, U, M, and A, transported only by L and C and repackaged only by P and N.

Use the downstream exactness package of Definition 402.31. It states that downstream $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers receive exactly the displayed packet. Thus the bridge has no further row from which hidden host completion data could enter. □

Theorem 402.38 (HausdorffCompletion standard bridge obligation package)**.** *The bridge obligation surface for $\mathsf{HausdorffCompletion}^{\uparrow}$ is the finite package consisting of Definition 402.36 and Definition 402.37, grounded in the public separated certificate of Definition 402.32. Its source field, classifier field, exactness field, ledger field, stability field, and non-escape field are all read from S, U, A, M, L, C, P, N.*

Proof. The source and classifier fields are S, U, A, M as exposed by the carrier and public separated certificate. The exactness and stability fields are the displayed transport and continuation rows L, C together with the separated classifier ledger.

The ledger field is the package provenance and local naming pair P, N. The downstream exactness package shows that those rows exhaust every consumer-facing use of the separated-completion packet.

The non-escape field is supplied by Definition 402.37. Combining these rows gives the bridge obligation surface without importing host completion structure into the BEDC carrier. □

Theorem 402.39 (HausdorffCompletion mature consumer examples)**.** *The $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMon}$ $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers of a $\mathsf{HausdorffCompletion}^{\uparrow}$ carrier all read the same separated-completion bridge packet*

$$S, U, A, M, L, C, P, N.$$

The example routes are the uniform entourage consumer through U, the regular-rational diagonal consumer through S and M, and the real-seal separated classifier consumer through A, all transported and packaged by L, C, P, N. These examples reuse the same finite rows and introduce no quotient completion, host equality, selected representative, uniqueness outside the separated classifier, or ambient completeness.

Proof. Begin with the public separated certificate and the standard bridge source packet. They identify S, U, A, M, L, C, P, N as the public separated-completion route and as the bridge-facing source packet.

Apply the bridge non-escape boundary and the standard bridge obligation package. These keep the uniform entourage read, the regular-rational diagonal read, and the real-seal separated classifier read inside the same finite row family.

Finally use the uniform consumer package already recorded by the chapter. Since $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers all project from the same packet, the examples add no quotient completion, host equality, selected representative, uniqueness outside A, or ambient completeness coordinate. □

Theorem 402.40 (HausdorffCompletion separated-limit induction)**.** *Let $H = (S, U, A, M, L, C, P, N)$ be an accepted* $\mathsf{HausdorffCompletion}^{\uparrow}$ *carrier whose Cauchy-filter route and separated comparison row are both exposed to a public consumer. For every finite refinement chain of displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *windows read from S, induction along the chain preserves the same regular-rational readback through S, the same* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row U, the same separated ledger A, the same* $\mathsf{Real}^{\uparrow}$ *seal-facing read, the same* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *transport through M, the same replay rows C, the same provenance row P, and the same local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$ *row N. Hence two terminal limit reads exported by that chain can differ only through a displayed failure of the separated comparison row; no quotient completion class, selected representative, host completion equality, or ambient completeness coordinate is available as an alternate source of difference.*

Proof. Use the Cauchy-filter uniqueness boundary of Definition 402.29. It confines every source comparison to the displayed route through S, U, M, the separated classifier A, the transport rows L, the replay rows C, the provenance row P, and the local naming row N.

For the induction base, a terminal read with no refinement is exactly the public separated packet of Definition 402.32. The uniform consumer package of Definition 402.19 gives the same row family to the $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers.

For the refinement step, replace one displayed Cauchy-filter or regular-rational window by its next displayed refinement. The Cauchy-filter boundary keeps the source read on the same entourage and completion-handoff route, while the separated limit uniqueness boundary of Definition 402.24 permits comparison only through the transported separated classifier row.

Thus every finite refinement preserves the same readback, entourage, separated ledger, seal-facing read, transport, replay, provenance, and local NameCert rows. If the two terminal reads disagree, the disagreement must be visible as failure of the separated comparison row; the public packet contains no other coordinate that could witness such a difference. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffCompletion}^{\uparrow}$ packet is generated from a Cauchy-filter or regular rational source, a uniform entourage route, a separated real-seal classifier, a completion-monad handoff, displayed transport and continuation rows, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 402.27, Definition 402.28, Definition 402.29, and Definition 402.26 bind the public separated-completion packet over the displayed $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows, with no host quotient completion or choice-selected representative.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HausdorffCompletion}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.HausdorffCompletionUp.HausdorffCompletionCarrier\_public\_separated\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not close quotient completion construction, host completion equality, selected representatives, ambient completeness, uniqueness of limits, or bridge checking.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 403

A Concrete Naming Certificate for CompletionReflection$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the completion reflection boundary is read here inside the BEDC mathlib-free substrate: completion data are admitted only as finite named rows, and reflection across the real seal is expressed by displayed transports rather than by host quotient objects. The CompletionReflection$^{\uparrow}$ packet ties together the construction surface of RealCauchyCompletion$^{\uparrow}$, the extension surface of CompletionUniversalProperty$^{\uparrow}$, and the separated boundary of HausdorffCompletion$^{\uparrow}$.

The reflection packet consumes the completion construction of chapter 360, the universal-property extension row from chapter 397, the separated boundary row from chapter 402, the regular-rational source row from chapter 301, and the real seal row from chapter 38. These sibling rows are the source of C, U, H, Q, A in the carrier tuple, and the CompletionReflection$^{\uparrow}$ packet only repackages their finite routes through hsame, Cont, Pkg, and NameCert.

Theorem 403.1 (CompletionReflection metric-existence route)**.** *Before a* CompletionReflection$^{\uparrow}$ *carrier exposes the* HausdorffCompletion$^{\uparrow}$ *separated boundary or the* RealCauchyCompletion$^{\uparrow}$ *construction handoff, it reads the metric-completion existence route of chapter 5721. The consumed route supplies only the displayed source metric, Cauchy schedule, dense embedding, finite windows, regular readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows; it does not add a selected quotient completion or host completeness theorem to the reflection packet.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket metric existence route`

Proof. Project the dependency paragraph and the carrier rows of Definition 403.3. The construction row C and separated boundary row H are the only completion-facing rows that a reflection consumer can inspect before transport and provenance are read.

The metric-completion existence chapter supplies the finite route that justifies reading those completion-facing rows as displayed BEDC data. Since that route is itself a finite NameCert packet, its contribution is provenance for C and H, not a new carrier coordinate.

Thus the handoff to HausdorffCompletion$^{\uparrow}$ or RealCauchyCompletion$^{\uparrow}$ remains inside the local tuple $C, U, H, D, Q, A, S, T, K, P, N$ and cannot expose a quotient completion, selected host limit, or ambient completeness theorem. □

Theorem 403.2 (CompletionReflection StreamName-RegSeqRat-Real factorization)**.** *Every accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *read factors through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the local reflection rows*

$$C, U, H, Q, A$$

before any completion-facing consumer can use the reflected object. The factorization is the finite route supplied by chapter 301, chapter 38, chapter 360, chapter 397, and chapter 402.

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket stream regseq real factor`

Proof. Project the dependency paragraph and the carrier rows. The completion construction row C, universal-property row U, Hausdorff boundary row H, regular-rational source row Q, and Real seal row A are exactly the local reflection rows available before transport and provenance are read.

Use the RegSeqRat and Real sibling rows to identify the finite source route. The StreamName finite-window surface feeds the regular rational readback, and the regular readback is the finite row through which the Real seal is consumed.

Now compose the completion-construction, universal-property, and Hausdorff-completion sibling rows. They allow the reflected object to be used only after the same finite C, U, H, Q, A route has been transported, replayed, cited, and named.

Thus a completion-facing consumer receives a reflected finite route, not a quotient stream, selected host limit, host real equality principle, or ambient completeness theorem. □

Definition 403.3 (CompletionReflection carrier)**.** *A* $\mathsf{CompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, U, H, D, Q, A, S, T, K, P, N)$$

with the following visible rows:

1. C *is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *completion-construction row;*
2. U *is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row attached to the same completion-facing source;*
3. H *is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated completion boundary row;*
4. D *is the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *diagonal real-seal row;*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row inspected by the diagonal and completion packets;*
6. A *is the* $\mathsf{Real}^{\uparrow}$ *seal row through which the completed object is consumed;*
7. S *records the componentwise* hsame *transports for construction, extension, separatedness, diagonal, regular-sequence, and real-seal rows;*
8. T *records the displayed* Cont *routes through the same rows;*
9. K *is the* Pkg *provenance over* $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
10. P *records the local reflection packet provenance over* BHist, hsame, Cont, *and* Pkg *rows;*
11. N *is the local* $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no selected host limit, no host equality of completed reals, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.TasteGate.CompletionReflectionTasteGate single carrier alignment`

Definition 403.4 (CompletionReflection classifier)**.** *For accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *carriers* R *and* R'*, the classifier* $\mathsf{CompletionReflectionClassifier}(R, R')$ *compares exactly the displayed rows of Definition 403.3. The construction row compares through* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, the extension row through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*, the separated boundary through* $\mathsf{HausdorffCompletion}^{\uparrow}$*, the diagonal row through* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*, the regular rational source through* $\mathsf{RegSeqRat}^{\uparrow}$*, and the seal row through* $\mathsf{Real}^{\uparrow}$*.*

The classifier has no component for a quotient of regular streams, a selected host limit, host real equality, or an external complete-space object. Transport is therefore confined to the displayed hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Lean4 (definition) `BEDC.Derived.CompletionReflectionUp.CompletionReflectionClassifier`

Theorem 403.5 (CompletionReflection NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission by the finite packet of Definition 403.3;*
2. *classifier transport by the visible comparison surface of Definition 403.4;*
3. *diagonal real-seal reflection from* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *into the* $\mathsf{Real}^{\uparrow}$ *seal row;*
4. *universal-property extension reflection from* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *back into the same completion and seal provenance;*
5. *non-escape from quotient streams, selected host limits, host equality, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket namecert obligations`

Proof. Carrier admission is projection of the finite packet. Each row required by the reflection boundary appears as a coordinate before any consumer can read it.

Classifier transport is componentwise. Use the displayed hsame rows to compare construction, extension, separatedness, diagonal, regular-sequence, and real-seal data, then replay the Cont routes attached to those same coordinates.

The diagonal reflection row reads D, Q, and A together. It states that the real-seal observation is reached through regular rational diagonal data and not through a host real representative.

The extension reflection row reads U against the same construction and seal provenance. Since the extension ledger is carried by U and routed through T, the reflected extension cannot introduce a separate completion object.

Finally the provenance rows K, P, and N account for package and local naming data. Because the carrier has no quotient-stream, selected-limit, host-equality, or ambient-completeness coordinate, no such object can be exported. □

Theorem 403.6 (CompletionReflection real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *packet factors through the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *construction row, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *diagonal row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated boundary row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The read exposes no quotient stream, selected host limit, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket real seal boundary`

Proof. Project an accepted carrier $R = (C, U, H, D, Q, A, S, T, K, P, N)$. The only real-facing coordinate is A, and it is paired in the packet with the completion construction row C, diagonal row D, regular source row Q, and separated boundary row H.

Use the classifier of Definition 403.4 to move between accepted representatives. It preserves exactly those rows by displayed transport and continuation, so any real-seal read remains attached to the same finite provenance.

Apply the non-escape row of Definition 403.5. There is no coordinate from which a quotient stream, host limit, host equality, or ambient completeness theorem could be read, so the real-seal boundary is the displayed finite route. □

Theorem 403.7 (CompletionReflection extension reflects)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N)$ be an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *carrier. Any completion-facing extension read exported by U reflects back to the same construction row C, diagonal row D, regular rational source Q, real-seal row A, separated boundary row H, and provenance rows S, T, K, P, N. No reflected extension can switch to a quotient stream, selected host limit, host equality principle, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket extension reflects`

Proof. Read the extension coordinate U inside the carrier. Its source and target provenance are not external: the same packet records the completion construction row C, diagonal row D, regular source row Q, and real-seal row A.

The separated boundary row H fixes the Hausdorff-facing route. Transport along S and T keeps the extension ledger attached to that route, while the package rows K and P record the dependency surface consumed by the local certificate.

The obligation surface in Definition 403.5 then repacks the extension read as a $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ row. Since every exported coordinate is visible in the carrier, the reflected extension has no path to unrecorded quotient, limit, equality, or completeness data. □

Theorem 403.8 (CompletionReflection carrier scope)**.** *The scoped carrier of* $\mathsf{CompletionReflection}^{\uparrow}$ *is exactly the finite packet*

$$(C, U, H, D, Q, A, S, T, K, P, N)$$

over $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *No scoped carrier coordinate names a quotient stream, selected host limit, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket carrier scope`

Proof. Project the accepted carrier through Definition 403.3. The definition lists the six construction-facing rows C, U, H, D, Q, A and then the five structural rows S, T, K, P, N.

The same definition also states the exclusions. Since every carrier read is a projection from this finite tuple, any object outside these coordinates has no scoped carrier slot from which it can be read. □

Theorem 403.9 (CompletionReflection classifier scope)**.** *The* $\mathsf{CompletionReflection}^{\uparrow}$ *classifier transports the scoped rows componentwise along the same $C, U, H, D, Q, A, S, T, K, P, N$ surface. It can compare construction, extension, separated-boundary, diagonal, regular-source, real-seal, transport, continuation, package, packet-provenance, and local naming rows, but it cannot add quotient streams, selected host limits, host real equality, or ambient completeness data.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket classifier scope`

Proof. Apply Definition 403.4. Its comparison clauses are exactly the rows already displayed in the carrier definition, with hsame and Cont supplying componentwise transport for the visible packet.

Because the classifier is defined only on accepted carriers and has no extra field, transport preserves the listed rows without creating a new coordinate. The excluded quotient, limit, equality, and completeness objects therefore remain outside the classifier scope. □

Theorem 403.10 (CompletionReflection consumer scope)**.** *Every consumer read exported by* $\mathsf{CompletionReflection}^{\uparrow}$ *factors through the displayed real-seal and extension rows: the real-facing read passes through* A *over* $C, D, Q, H, S, T, K, P, N$*, and the extension-facing read passes through* U *over the same construction, diagonal, regular-source, real-seal, separated-boundary, transport, continuation, package, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket consumer scope`

Proof. For real-facing consumers, apply Definition 403.6. It routes every $\mathsf{Real}^{\uparrow}$ read through the displayed construction, diagonal, regular-source, separated-boundary, transport, continuation, package, and naming coordinates.

For extension-facing consumers, apply Definition 403.7. The extension coordinate U reflects back to the same rows and cannot switch to an unrecorded completion object.

Combining the two routes exhausts the consumer surface: a consumer can read only the displayed real-seal row or the displayed extension row, each with the same finite provenance boundary. □

Theorem 403.11 (CompletionReflection public export)**.** *Any public consumer of* $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ *may read exactly the completion construction row* C*, universal-property extension row* U*, Hausdorff separated boundary row* H*, diagonal regular-sequence reflection row* D*, regular rational source row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal row* A*, displayed* hsame *and* Cont *transports* S, T*,* Pkg *provenance* K, P*, and the local* NameCert *row* N *from the carrier*

$$(C, U, H, D, Q, A, S, T, K, P, N).$$

The public export has no coordinate for quotient completion objects, selected host limits, host equality of completed reals, or ambient completeness.

Proof. First apply Definition 403.8. It identifies the public carrier surface with the finite tuple $C, U, H, D, Q, A, S, T, K, P, N$ and excludes quotient streams, selected host limits, host real equality, and ambient completeness from the carrier coordinates.

Next apply Definition 403.9. Public movement between accepted packets is componentwise along the same construction, extension, separated-boundary, diagonal, regular-source, real-seal, transport, continuation, package, packet-provenance, and local naming rows.

Finally apply Definition 403.10. Every public consumer read factors through the real-seal row or the extension row over the same finite provenance surface, so the exported $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ interface is exactly the displayed tuple and nothing outside it. □

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket public export`

Theorem 403.12 (CompletionReflection universal real-seal factorization)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N$ be an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *carrier. Every universal-property extension route exported by U factors through the displayed* $\mathsf{Real}^{\uparrow}$ *seal row A before any completion-facing consumer can read it. The factorization uses the completion construction row C, diagonal row D, regular rational source row Q, Hausdorff boundary row H, componentwise* hsame *rows S, displayed* Cont *routes T, package provenance K, P, and local* NameCert *row N.*

No universal-property consumer can bypass the real-seal row by reading a quotient completion object, selected host limit, host equality of completed reals, or ambient completeness theorem.

Proof. Begin with Definition 403.6. It identifies the $\mathsf{Real}^{\uparrow}$ seal row A as the real-facing boundary through which completion data become consumable.

Apply Definition 403.7. Any extension route supplied by U reflects back to the same construction row, diagonal row, regular source, real-seal row, separated boundary, and provenance surface.

Use Definition 403.10 to align the two reads. It states that real-facing and extension-facing consumers share the same finite rows, so the extension route cannot be consumed before the real-seal boundary is present.

Finally apply Definition 403.11. The public packet exposes exactly $C, U, H, D, Q, A, S, T, K, P, N$ and no quotient, selected limit, host equality, or ambient completeness coordinate, so the universal route factors through the displayed seal for every completion-facing consumer. □

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket universal real seal factorization`

Theorem 403.13 (CompletionReflection real-seal extension coherence)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N)$ be an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *packet. The public real-seal read through A and the universal-property extension read through U are coherent exactly when both reads factor through the same displayed rows*

$$C, D, Q, A, H, S, T, K, P, N.$$

Neither read may replace those rows with a quotient stream, selected host limit, host equality, or ambient completeness datum.

Proof. For the real-facing direction, apply Definition 403.6. Every $\mathsf{Real}^{\uparrow}$ read is routed through the displayed construction, diagonal, regular-source, separated-boundary, transport, continuation, package, and naming rows before it reaches the seal row A.

For the extension-facing direction, apply Definition 403.7. The route exported by U reflects back to $C, D, Q, A, H, S, T, K, P, N$, so the extension read cannot introduce a second completion surface.

Now align the two readings by Definition 403.10. Its consumer boundary says that the real-facing read and the extension-facing read use the same finite packet surface, with A and U as the two visible public entries.

Finally apply Definition 403.11. Since the public export exposes only $C, U, H, D, Q, A, S, T, K, P, N$ and excludes quotient streams, selected host limits, host equality, and ambient completeness, coherence is exactly shared factorization through the displayed rows. □

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket real seal extension coherence`

Theorem 403.14 (CompletionReflection regular-source tail reflection)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N)$ be an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *packet whose reflected completion read starts from a regular rational tail window. Then every such read factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RealCauchyCompletion}^{\uparrow} \longrightarrow \mathsf{CompletionUniversalProperty}^{\uparrow} \longrightarrow \mathsf{Hausdo}$$

before it reaches the local $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ *row N. The factorization uses the carrier rows C, D, Q, A, H, the classifier transport rows S, the displayed* Cont *routes T, package provenance K, P, and the local naming row N.*

No reflected regular-source tail read may replace this route by quotient streams, selected host limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket regular source tail reflection`

Proof. Start with the accepted carrier. By Definition 403.8, the only regular-source data visible to the packet are the diagonal reflection row D, the regular rational source row Q, the real-seal row A, the Hausdorff boundary row H, and the completion construction row C.

Apply Definition 403.5. The local NameCert obligations read Q through the same finite packet surface and route it through the completion construction, universal extension, Hausdorff separated boundary, diagonal reflection, real seal, transport, continuation, provenance, and naming rows.

Next use Definition 403.9. Componentwise hsame transport preserves the regular-source, real-seal, completion-construction, universal-property, and Hausdorff-boundary rows, so the reflected tail read cannot branch into an unlisted completion surface.

Finally apply Definition 403.11. The exported packet exposes only $C, U, H, D, Q, A, S, T, K, P, N$, which realizes the displayed $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{HausdorffCompletion}^{\uparrow}$ route before the local $\mathsf{NameCert}_{\mathsf{CompletionReflection}^{\uparrow}}$ row. Since quotient streams, selected host limits, host real equality, and ambient completeness are absent from that export, no regular-source tail reflection may read them. □

Theorem 403.15 (CompletionReflection standard bridge boundary)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N)$ be an accepted* $\mathsf{CompletionReflection}^{\uparrow}$ *packet. If the carrier row, classifier row, local* $\mathsf{NameCert}$ *obligations, real-seal boundary, extension-reflection row, consumer scope, provenance-route scope, and public export of R are displayed, then R induces a standard bridge boundary read only through the finite rows already named by this chapter:* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-seal row, classifier row,* hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}$ *row.*

The bridge boundary exports no quotient completion, no host equality of completed reals, no selected host limit, and no ambient completeness claim.

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket standard bridge boundary`

Proof. First read the accepted packet through its displayed local obligations. Apply Definition 403.5 to bind the completion construction, universal extension, separated boundary, diagonal reflection, regular rational source, real-seal, transport, continuation, package, provenance, and naming rows.

Next route the real-facing and extension-facing reads. Apply Definition 403.6 for the $\mathsf{Real}^{\uparrow}$ seal row and Definition 403.7 for the universal-extension row; both routes return to the same displayed construction, regular-source, real-seal, classifier, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ surface.

Now restrict the consumers and provenance routes. Apply Definition 403.10 and Definition 403.9 to ensure that every admitted read uses only the rows already visible in the chapter.

Finally apply Definition 403.11. The public export exposes exactly the finite completion-reflection tuple and excludes quotient completion objects, host equality, selected host limits, and ambient completeness. This is precisely the standard bridge boundary over the displayed rows, with no additional completion-side coordinate. □

Theorem 403.16 (CompletionReflection real-completion functor handoff)**.** *Let $R = (C, U, H, D, Q, A, S, T, K, P, N$ be an accepted* CompletionReflection$^{\uparrow}$ *packet whose regular-source row Q and real-seal row A are consumed by an accepted* RealCompletionFunctor$^{\uparrow}$ *packet. Then every completion-facing read handed from R to* CauchyCompletionFunctor$^{\uparrow}$ *factors through the displayed L10 route*

DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, hsame, Cont, Pkg, NameCert.

In particular, the handoff uses only the reflection rows $C, U, H, D, Q, A, S, T, K, P, N$, the four-face functor rows of RealCompletionFunctor$^{\uparrow}$*, and the unit-and-bind surface of* CauchyCompletionFunctor$^{\uparrow}$*. It exports no quotient completion object, selected limit, host real equality, ambient completeness theorem, or hidden completion coordinate.*

Lean4 ✓ `BEDC.Derived.CompletionReflectionUp.CompletionReflectionPacket real completion functor handoff`

Proof. Begin with the accepted reflection packet. By Definition 403.15, any standard bridge boundary read from R is already confined to the displayed regular-source, real-seal, classifier, hsame, Cont, Pkg, and local NameCert rows.

Read the consuming RealCompletionFunctor$^{\uparrow}$ packet next. Its terminal-seal non-escape theorem, Definition 2073.3, forces the same completion-facing request through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before the completion-functor and universal-property rows are inspected.

Now attach the CauchyCompletionFunctor$^{\uparrow}$ unit-and-bind surface. The unit laws of Definition 423.3 and the multiplication laws of Definition 423.4 read only the displayed source, regular-Cauchy, completion-seal, unit, bind, universal-property, transport, provenance, and local naming rows.

Composing these three routes gives the asserted handoff. Each step preserves the same displayed hsame, Cont, Pkg, and NameCert provenance, so there is no point at which a quotient completion object, selected limit, host real equality, ambient completeness theorem, or hidden completion coordinate can enter. □

Closure status for NameCert$_{\text{CompletionReflection}^{\uparrow}}$**.**

Constructive story: A CompletionReflection$^{\uparrow}$ packet is generated from completion construction, universal-property extension, Hausdorff separated boundary, diagonal regular-sequence reflection, real-seal, transport, continuation, package, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\text{CompletionReflection}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 403.8, Definition 403.9, Definition 403.10, Definition 403.11, and Definition 403.16 bind the finite carrier, componentwise classifier transport, public export, real-seal universal factorization route, and consumer read-back boundary over RealCauchyCompletion$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, HausdorffCompletion$^{\uparrow}$, RealDiagonalCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{CompletionReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completion construction, selected host limits, host equality of completed reals, ambient completeness, or bridge checking.

Upgrade path: No further paper-side theory closure remains; bridge verification remains a separate verification-axis obligation.

Chapter 404

A Concrete Naming Certificate for $\mathsf{LocatedCut}^{\uparrow}$

The $\mathsf{LocatedCut}^{\uparrow}$ interface records the constructive real described by located lower and upper rational information while keeping the completion-facing readback visible as finite rows. Following Bishop and Bridges, *Constructive Analysis*, locatedness is represented as explicit interval evidence at each unary precision, not as a quotient of cuts or a selected supremum. The packet depends on $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, dyadic interval ledgers, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The StreamName window row is read through chapter 126, the regular-rational handoff row follows the finite schedule boundary of chapter 301, and the seal row is the consumer-facing RealUp route of chapter 38.

Definition 404.1 (LocatedCut carrier)**.** *A* $\mathsf{LocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D^-, D^+, W, R, S, H, C, P, N)$$

with the following visible rows:

1. D^- *is the lower dyadic interval ledger, recording finite witnesses that a dyadic rational lies below the presented cut;*
2. D^+ *is the upper dyadic interval ledger, recording finite witnesses that a dyadic rational lies above the presented cut;*
3. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing, for each unary precision visible in the packet, the finite lower and upper ledger entries exposed to consumers;*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row extracted from the displayed located windows;*
5. S *is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached through the regular rational handoff;*
6. H *records the componentwise* hsame *transports for lower ledger, upper ledger, stream-window, regular-sequence, and real-seal rows;*
7. C *records the displayed* Cont *routes through the same rows;*
8. P *is the* Pkg *provenance over the displayed dyadic, stream, regular-sequence, and real-seal dependencies;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *row.*

The carrier contains no quotient cut, no host real equality, no selected supremum, and no use of Classical.choice.

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutTasteGate single carrier alignment`

Theorem 404.2 (LocatedCut NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$ *surface consists of six finite rows:*

1. *carrier habitation by lower and upper dyadic ledgers,* $\mathsf{StreamName}^\uparrow$ *windows, a* $\mathsf{RegSeqRat}^\uparrow$ *handoff, a* $\mathsf{Real}^\uparrow$ *seal row, displayed transports, continuations, package provenance, and local naming;*
2. *lower-ledger and upper-ledger stability under the stored* hsame *transports;*
3. *locatedness evidence at each unary precision, witnessed by a finite pair of dyadic bounds in the corresponding* $\mathsf{StreamName}^\uparrow$ *window row;*
4. *compatibility of those windows with the* $\mathsf{RegSeqRat}^\uparrow$ *handoff row;*
5. *factorization of the* $\mathsf{Real}^\uparrow$ *seal through the handoff row and displayed provenance;*
6. *exactness of the* Pkg *and local* NameCert *rows for the listed dependencies.*

These rows certify a located constructive real without quotient cuts, host real equality, selected suprema, or Classical.choice.

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier namecert obligations`

Proof. Carrier habitation is projection of Definition 404.1. The lower ledger, upper ledger, stream-window, regular-sequence, real-seal, transport, continuation, package, and local naming rows are finite coordinates of the same BHist packet.

Ledger stability is componentwise. Move the lower and upper dyadic rows along the stored hsame transports, then replay the displayed Cont routes for the same window entries. Since each unary precision appears only through a finite window row, locatedness is a displayed witness rather than an ambient order-completeness claim.

The compatibility and seal rows use the same finite path. The $\mathsf{RegSeqRat}^\uparrow$ handoff reads only the located dyadic windows, and the $\mathsf{Real}^\uparrow$ seal reads only that handoff together with the package and local naming rows.

The exclusions follow from the carrier shape. No coordinate stores a quotient cut, host real equality proof, selected supremum, or classical choice term, so the listed finite rows exhaust the certificate surface. □

Theorem 404.3 (LocatedCut regular Cauchy handoff)**.** *Every accepted* $\mathsf{LocatedCut}^\uparrow$ *packet exports a* $\mathsf{RegSeqRat}^\uparrow$ *handoff whose* n*th visible approximation is selected from the finite lower and upper dyadic ledgers displayed in the* $\mathsf{StreamName}^\uparrow$ *row for unary precision* n*. The handoff is transported only by the stored* hsame *rows and displayed* Cont *routes, and it introduces no quotient cut, host real equality, selected supremum, or* Classical.choice.

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier regular cauchy handoff`

Proof. Fix an accepted carrier $L = (D^-, D^+, W, R, S, H, C, P, N)$. For a visible unary precision n, the window row W supplies the finite lower and upper dyadic ledger entries associated with n.

The handoff row R is one of the carrier coordinates. Its readback is constrained by the NameCert obligations of Definition 404.2, so it may inspect only the displayed ledger entries in W and the stored transport and continuation rows.

Transport is therefore internal to the packet. Moving along H and replaying C changes the presented row names but not the finite dependency route from located windows to R.

Because neither the carrier nor the obligations contain a quotient-cut coordinate, a host real equality test, a supremum selector, or a classical choice row, the resulting regular Cauchy handoff is constructive and finite-row based. □

Theorem 404.4 (LocatedCut dyadic interval exhaustion)**.** *For every accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet and every unary precision row visible in its* $\mathsf{StreamName}^{\uparrow}$ *window ledger, the lower and upper dyadic interval rows supply a finite located interval whose width is bounded by that precision row and whose endpoints remain attached to the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. No appeal is made to a quotient of cuts, host real equality, selected supremum, or* Classical.choice*.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier dyadic interval exhaustion`

Proof. Read the visible unary precision row from W. By the locatedness clause in Definition 404.2, the same window carries a finite lower dyadic witness from D^- and a finite upper dyadic witness from D^+.

The interval bound is a displayed row of the packet, not a completeness theorem. Its width claim is checked against the unary precision row inside the finite window ledger, and the endpoint names remain tied to the same package and local naming coordinates.

Stability follows by replaying the stored hsame transports and Cont routes for the lower endpoint, upper endpoint, precision row, and window row. The transported interval is still read from the same finite packet.

Thus the visible precision rows are exhausted by displayed dyadic interval evidence. The construction never forms a cut quotient, never compares host real equalities, never selects a supremum, and never invokes classical choice. □

Theorem 404.5 (LocatedCut Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *carrier factors through the lower and upper dyadic ledgers, the* $\mathsf{StreamName}^{\uparrow}$ *located window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows. The exported seal cannot bypass those rows by using quotient cuts, host real equality, selected suprema, or* Classical.choice*.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier real seal nonescape`

Proof. Start with an accepted carrier $L = (D^-, D^+, W, R, S, H, C, P, N)$. The only real-facing coordinate is S, and the only handoff coordinate feeding it is R.

Apply Definition 404.3. The handoff R is reached through the located dyadic windows in W, with lower and upper evidence read from D^- and D^+.

Apply Definition 404.4. Each visible precision row is exhausted by finite interval evidence carrying the same transport, continuation, package, and local naming provenance.

The $\mathsf{Real}^{\uparrow}$ seal therefore has no escape route outside the displayed chain

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S.$$

Since the carrier has no quotient, host equality, supremum selector, or classical choice coordinate, the listed rows give the complete seal boundary. □

Theorem 404.6 (LocatedCut classifier transport)**.** *Let $L = (D^-, D^+, W, R, S, H, C, P, N)$ and $L' = (D'^-, D'^+, W', R', S', H', C', P', N')$ be accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packets. Classifier transport is componentwise over the lower ledger, upper ledger, stream-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, displayed* hsame *rows, displayed* Cont *routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *row. It transports no quotient cut, host real equality, selected supremum, or ambient order-completeness datum.*

Proof. Project the carrier tuple of Definition 404.1. Its visible coordinates are exactly the two dyadic ledgers, window row, handoff row, seal row, transport rows, continuation rows, provenance, and local naming row.

Apply the finite-row clauses of Definition 404.2. The lower and upper ledgers move along the stored hsame transports, while the displayed Cont routes replay the same window, handoff, and seal reads.

The non-escape part is immediate from the carrier shape. None of the transported coordinates stores a quotient cut, host real equality proof, supremum selector, or order-completeness object. □

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier classifier transport`

Theorem 404.7 (LocatedCut located-window exactness)**.** *For every accepted packet $L = (D^-, D^+, W, R, S, H, C, P$ and every visible unary precision row, the stream-window row W reads a finite lower dyadic entry from D^- and a finite upper dyadic entry from D^+. The corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is computed from that same displayed window and not from a quotient cut or selected supremum.*

Proof. Use the locatedness row of Definition 404.2. For each visible precision it requires a finite lower witness and a finite upper witness in the same $\mathsf{StreamName}^{\uparrow}$ window row.

Apply Definition 404.4. The dyadic endpoints and the precision row remain attached to the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Finally apply Definition 404.3. The handoff row R reads the located window supplied by W, so the visible precision is answered by finite dyadic data rather than by an ambient cut object. □

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier located window exactness`

Theorem 404.8 (LocatedCut seal-boundary exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet factors through the displayed chain*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S.$$

The seal boundary exposes no quotient cut, host real equality, selected supremum, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier seal boundary exactness`

Proof. Start with the packet $L = (D^-, D^+, W, R, S, H, C, P, N)$ of Definition 404.1. The only seal-facing row is S, and its only regular handoff input is R.

Use Definition 404.7 to identify the window read feeding R. Each visible precision reaches a finite lower and upper dyadic pair in W.

Apply Definition 404.5. It gives the same factorization from the dyadic ledgers through W and R to S, and its non-escape clause excludes quotient cuts, host equality, selected suprema, and ambient completeness. □

Theorem 404.9 (LocatedCut paired ledger coherence)**.** *For every accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet* $L = (D^-, D^+, W, R, S, H, C, P, N)$*, the lower dyadic ledger* D^- *and upper dyadic ledger* D^+ *are transported as one paired located-interval row. Componentwise* hsame *transport and the displayed* Cont *routes preserve the common precision window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* S*, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier ledger pair coherence`

Proof. Start with the carrier of Definition 404.1. The two dyadic ledgers occur as adjacent coordinates of the same finite BHist packet, and the window row W is the only row that exposes them to consumers.

Use Definition 404.2 for the ledger-stability clauses. They move D^- and D^+ along the stored hsame rows, while the displayed Cont routes replay the same finite window reads.

Apply Definition 404.4 and Definition 404.6. Each visible precision row is therefore carried as a lower-upper pair with its endpoint names, provenance, handoff route, and local certificate attached.

The transported object is a paired located interval row, not a quotient cut or host equality datum. The exclusions are exactly the non-escape exclusions recorded in the carrier and in Definition 404.8. □

Theorem 404.10 (LocatedCut window handoff totality)**.** *Every visible unary precision window in an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff route. The route reads its lower and upper dyadic entries from the same stream window, transports them through the paired ledger row, and reaches the regular rational handoff without quotient cuts, selected suprema, host real equality, or* Classical.choice*.*

Proof. Fix a visible unary precision row in W. By Definition 404.7, the row displays a finite lower dyadic entry from D^- and a finite upper dyadic entry from D^+.

Use Definition 404.9. The two endpoint entries travel together as the paired located interval row, so the precision window does not split into unrelated lower and upper consumers.

Now apply Definition 404.3. The handoff row R is computed from the displayed located window and the stored transport and continuation routes. Thus every visible window supplies the regular rational handoff route demanded by the $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ obligations.

Finally, Definition 404.5 excludes any alternate route through quotient cuts, selected suprema, host real equality, or classical choice. □

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier window handoff totality`

Theorem 404.11 (LocatedCut common-window refinement)**.** *For any two visible unary precision windows of an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet, there is a displayed stronger window row whose lower and upper dyadic ledger reads refine both windows at once. The common row transports the paired ledger entry through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *provenance, supplies one* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff route, and reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S.$$

No quotient cut, selected supremum, host real equality, ambient completeness principle, or Classical.choice *row is exposed by the refinement.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier common window refinement`

Proof. Start with the two visible unary precision windows. By Definition 404.7, each window reads finite lower and upper dyadic entries from D^- and D^+ inside the displayed stream-window ledger.

Refine the two reads by taking the stronger visible window row already present in the accepted packet. Apply Definition 404.9. The lower and upper endpoints remain a single paired ledger row, and the same hsame and Cont data transport that row for both original windows.

Now apply Definition 404.10. The common window feeds one $\mathsf{RegSeqRat}^{\uparrow}$ handoff route before any real-facing seal is read.

Finally use Definition 404.5. The route can end at the displayed $\mathsf{Real}^{\uparrow}$ seal row only through the finite provenance chain, so quotient cuts, selected suprema, host equality, ambient completeness, and classical choice remain outside the packet. □

Theorem 404.12 (LocatedCut Real seal factorization package)**.** *An accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet assembles a finite* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *obligation package consisting of the carrier, classifier transport, paired dyadic ledger coherence, located-window exactness, common-window refinement, window handoff totality, and real seal-boundary exactness. Consequently every* $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S$$

with its displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier real seal factorization package`

Proof. The carrier and finite obligation rows are supplied by Definition 404.1 and Definition 404.2. These rows name the two dyadic ledgers, stream-window row, regular rational handoff, real seal row, transport rows, continuation rows, provenance, and local certificate.

Apply Definition 404.6 and Definition 404.9. They bind classifier movement to componentwise finite-row transport and keep the lower and upper ledgers as one located interval row.

Next apply Definition 404.7, Definition 404.11, and Definition 404.10. Every visible unary precision window supplies the finite dyadic pair used by the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and any pair of visible windows has a common finite refinement before the real seal is reached.

Finally apply Definition 404.8. The seal row is reached only through the displayed chain from dyadic ledgers to W, then to R, then to S. This is exactly the finite obligation package, and it contains no quotient cut, host real equality, selected supremum, ambient completeness principle, or classical choice row. □

Theorem 404.13 (LocatedCut standard bridge boundary)**.** *Every public standard located-cut reading exported by an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *package factors through the finite route*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S$$

where D^- and D^+ are the lower and upper dyadic ledgers, W is the $\mathsf{StreamName}^{\uparrow}$ *window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and S is the* $\mathsf{Real}^{\uparrow}$ *seal. The bridge boundary retains the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *provenance and exports no quotient cut, selected supremum, host real equality, ambient completeness principle, or* Classical.choice *row.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier standard bridge boundary`

Proof. Begin with the carrier obligations of Definition 404.2. They name the two dyadic ledgers, the stream window row, the regular rational handoff, the real seal row, and the visible transport, continuation, package, and local naming rows.

Next use located-window exactness from Definition 404.7. Each standard located-cut read at a visible precision is represented by a finite pair of lower and upper dyadic entries inside the same window row W.

Apply window handoff totality from Definition 404.10. It routes those displayed windows into the $\mathsf{RegSeqRat}^{\uparrow}$ handoff without adding a quotient cut or a selected supremum.

Finally apply the real seal factorization package of Definition 404.12. The handoff reaches the $\mathsf{Real}^{\uparrow}$ seal only through the displayed provenance rows, so the standard boundary is exactly the stated route and has no host real equality, ambient completeness, or classical choice coordinate. □

Theorem 404.14 (LocatedCut scoped kernel route)**.** *Every consumer route of an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet is exhausted by the lower dyadic ledger* D^{-}*, the upper dyadic ledger* D^{+}*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* S*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *rows. No route can pass through a quotient cut, host real equality, selected supremum, ambient completeness row, or classical choice field.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier scoped kernel route`

Proof. Project the carrier of Definition 404.1. Its visible coordinates are exactly the two dyadic ledgers, the stream-window row, the regular rational handoff, the real seal, transports, continuations, package provenance, and local naming row.

Use Definition 404.7 and Definition 404.9. They show that every visible precision window reads a lower-upper dyadic pair from the same packet and transports it as one located interval row.

Then apply Definition 404.10 and Definition 404.12. The handoff and seal routes are already the displayed chain from (D^{-}, D^{+}) through W and R to S, with the same hsame, Cont, Pkg, and NameCert provenance. □

Theorem 404.15 (LocatedCut obligation scope package)**.** *The* $\mathsf{LocatedCut}^{\uparrow}$ *scoped* NameCert *surface consists exactly of the carrier of Definition 404.1, classifier transport, paired ledger coherence, located-window exactness, handoff totality, and seal-boundary exactness. This package binds the lower and upper dyadic ledgers,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows as the complete kernel scope for located-cut consumers.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier obligation scope package`

Proof. Start with carrier habitation and classifier movement. The carrier is Definition 404.1, and Definition 404.6 states that transport is componentwise over the finite rows named there.

Next package the dyadic and window obligations. The paired ledger coherence theorem Definition 404.9 keeps the lower and upper ledgers attached, while Definition 404.7 identifies the finite window entries consumed at each visible precision.

Finally package the handoff and seal obligations. The window handoff totality theorem Definition 404.10, the seal-boundary exactness theorem Definition 404.8, and the scoped kernel route Definition 404.14 exhaust the real-facing route without adding quotient cuts, host equality, selected suprema, ambient completeness, or classical choice. □

Theorem 404.16 (LocatedCut public NameCert export)**.** *The carrier, classifier transport, paired ledger coherence, located-window exactness, handoff totality, seal-boundary exactness, and scoped kernel route assemble the public* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *export over the lower and upper dyadic ledgers,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Every downstream located-real consumer receives this public export as a finite BEDC packet, rather than as a quotient cut, selected supremum, host real equality, ambient order-completeness principle, or classical choice object.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier public namecert export`

Proof. Begin with Definition 404.15. It packages the carrier, classifier transport, paired dyadic ledgers, located windows, regular rational handoff, real seal, transport, continuation, package, and local naming rows as the complete scoped kernel surface.

Apply the seal factorization package of Definition 404.12. It routes real-facing reads through the displayed chain from (D^-, D^+) to W, then to R, and then to S with the same BHist, hsame, Cont, Pkg, and NameCert provenance.

Finally use Definition 404.14. The public export is therefore a repacking of the scoped package and seal factorization rows, and it exposes no coordinate for quotient cuts, selected suprema, host real equality, ambient completeness, or classical choice. □

Theorem 404.17 (LocatedCut window monotonic refinement)**.** *Let an accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet refine its* $\mathsf{StreamName}^{\uparrow}$ *window to a narrower lower-upper dyadic interval inside the same located-cut ledger. The refined window preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row and the* $\mathsf{Real}^{\uparrow}$ *seal boundary through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance as the coarser window.*

In particular, narrowing the finite interval changes only the visible lower-upper dyadic window row. It does not introduce a quotient cut, selected supremum, host real equality, ambient completeness row, or classical choice datum.

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier window monotonic refinement`

Proof. Begin with located-window exactness from Definition 404.7. It represents every visible precision as a finite lower-upper dyadic pair inside the displayed $\mathsf{StreamName}^{\uparrow}$ window row.

Dyadic interval exhaustion supplies the narrower window as another finite lower-upper pair in the same ledger. Ledger-pair coherence from Definition 404.9 keeps the lower and upper rows attached to the same packet under componentwise hsame transport.

Apply common-window refinement from Definition 404.11. The refined and coarser windows route to the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff row through the displayed Cont paths, with unchanged Pkg provenance and local NameCert row.

Finally use seal-boundary exactness from Definition 404.8. Since the real seal consumes the handoff row reached by the displayed finite route, the narrower window preserves the same $\mathsf{Real}^{\uparrow}$ boundary and adds no quotient, selected supremum, host equality, ambient completeness, or classical choice coordinate. □

Theorem 404.18 (LocatedCut interval witness extraction)**.** *For every accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet and every visible unary precision row in its* $\mathsf{StreamName}^{\uparrow}$ *window ledger, the public read exposes a lower dyadic witness, an upper dyadic witness, a common* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff route, a* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *provenance. The extraction reads only these finite interval-witness rows and never replaces them by a quotient cut, selected supremum, host real equality, ambient completeness principle, or classical choice field.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier interval witness extraction`

Proof. Begin with located-window exactness from Definition 404.7. It identifies each visible unary precision row with a finite lower-upper dyadic interval in the $\mathsf{StreamName}^{\uparrow}$ window ledger.

Apply dyadic interval exhaustion and paired ledger coherence. These supply the lower dyadic witness and upper dyadic witness while keeping both endpoints attached to the same displayed packet and the same componentwise hsame transport.

Use common-window refinement from Definition 404.11. It places the two endpoint witnesses in a common $\mathsf{StreamName}^{\uparrow}$ window and routes that window to the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff through the visible Cont rows.

Finally apply seal-boundary exactness from Definition 404.8. The $\mathsf{Real}^{\uparrow}$ seal consumes the displayed handoff with unchanged BHist, hsame, Cont, Pkg, and NameCert provenance, so the extracted interval witness cannot escape to quotient, supremum, host equality, ambient completeness, or choice data. □

Theorem 404.19 (LocatedCut regular Cauchy bridge source)**.** *For every accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet, the finite source used by the* $\mathsf{RegSeqRat}^{\uparrow}$ *bridge is exactly the displayed route*

$$(D^-, D^+) \longrightarrow W \longrightarrow R.$$

Here D^- and D^+ are the lower and upper dyadic ledgers, W is the $\mathsf{StreamName}^{\uparrow}$ *window row that reads a common located interval, and R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. The bridge source retains the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *provenance and exports no quotient cut, selected supremum, host equality, trichotomy, ambient completeness row, or classical choice datum.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier regular cauchy bridge source`

Proof. Begin with interval witness extraction from Definition 404.18. Every visible precision exposes a lower dyadic witness, an upper dyadic witness, and a common $\mathsf{StreamName}^{\uparrow}$ window inside the accepted packet.

Apply common-window refinement from Definition 404.11. The refinement keeps the two dyadic ledger entries attached to one window row W under the same BHist, hsame, Cont, Pkg, and local naming provenance.

Use window handoff totality from Definition 404.10. The handoff to $\mathsf{RegSeqRat}^{\uparrow}$ is the row R, and it is reachable only after the displayed lower-upper dyadic pair and the common window have been read.

The exclusions follow from Definition 404.14. The scoped route has no coordinate for quotient cuts, selected suprema, host equality, trichotomy, ambient completeness, or classical choice, so the regular Cauchy bridge source cannot export any of them. □

Theorem 404.20 (LocatedCut Real seal bridge exactness)**.** *For every accepted* $\mathsf{LocatedCut}^{\uparrow}$ *packet, the exact bridge source for the* $\mathsf{Real}^{\uparrow}$ *seal is the finite route*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S.$$

The row S is reached only after the lower and upper dyadic ledgers have been read through the common $\mathsf{StreamName}^{\uparrow}$ *window W and passed to the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R. Thus the Real seal route is exact over the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^{\uparrow}}$ *provenance and excludes quotient cuts, selected suprema, host equality, trichotomy, ambient completeness, and choice.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier real seal bridge exactness`

Proof. Start with Definition 404.19. It supplies the concrete bridge source from (D^-, D^+) through the common window W to the regular Cauchy handoff R.

Next apply the real seal factorization package of Definition 404.12. The seal-facing read factors through the same finite dyadic ledger and window route before it reaches the $\mathsf{Real}^\uparrow$ seal row.

Finally use seal-boundary exactness from Definition 404.8. The seal row S is terminal for this displayed route, so no later consumer can replace the route by a quotient cut, selected supremum, host equality, trichotomy, ambient completeness principle, or choice field. □

Theorem 404.21 (LocatedCut bridge boundary package)**.** *For every accepted* $\mathsf{LocatedCut}^\uparrow$ *packet, the bridge boundary to the* $\mathsf{Real}^\uparrow$ *seal factors through exactly the displayed route*

$$(D^-, D^+) \longrightarrow W \longrightarrow R \longrightarrow S \longrightarrow H, C, P, N.$$

Here D^- and D^+ are the lower and upper dyadic ledgers, W is the common $\mathsf{StreamName}^\uparrow$ *window row, R is the* $\mathsf{RegSeqRat}^\uparrow$ *handoff row, S is the* $\mathsf{Real}^\uparrow$ *seal row, and H, C, P, N are the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$ *rows. The bridge boundary exports no quotient cut, selected supremum, host equality, trichotomy, ambient completeness, or* Classical.choice *datum.*

Lean4 ✓ `BEDC.Derived.LocatedCutUp.LocatedCutCarrier bridge boundary package`

Proof. Start with located-window exactness and paired ledger coherence, Definition 404.7 and Definition 404.9. They keep every visible precision attached to a finite lower-upper dyadic pair in the same displayed packet.

Use common-window refinement and window handoff totality. The dyadic pair is read through one $\mathsf{StreamName}^\uparrow$ window row W, and that window reaches the $\mathsf{RegSeqRat}^\uparrow$ handoff row R without adding a quotient cut or selected supremum.

Apply the regular Cauchy bridge source and Real seal bridge exactness, Definition 404.19 and Definition 404.20. These place the bridge source on the route from (D^-, D^+) through W and R to the $\mathsf{Real}^\uparrow$ seal row S.

Finally attach the structural rows H, C, P, N. Seal-boundary exactness and the public NameCert export keep those rows on the same finite route, so no quotient cut, selected supremum, host equality, trichotomy, ambient completeness, or classical choice field can be read from the bridge boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$.

Constructive story: A $\mathsf{LocatedCut}^\uparrow$ packet is generated from lower and upper dyadic interval ledgers, located $\mathsf{StreamName}^\uparrow$ precision windows, a $\mathsf{RegSeqRat}^\uparrow$ handoff, a $\mathsf{Real}^\uparrow$ seal-facing row, displayed transport and continuation rows, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 404.16 binds the scoped package and seal factorization rows as the public $\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$ export, and Definition 404.21 binds the bridge boundary through lower and upper dyadic ledgers, StreamName windows, RegSeqRat handoff, Real seal, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCut}^\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient cut construction, host real equality, selected suprema, ambient order completeness, uniqueness of represented reals, or bridge checking.

Upgrade path: Formal and bridge verification remain the route from the mature displayed dyadic ledger treatment to checked theorem, axiom-clean, and bridge-checked status over the same window, handoff, seal, package, and naming rows.

Chapter 405

A Concrete Naming Certificate for $\mathsf{CofinalSubsequence}^{\uparrow}$

The $\mathsf{CofinalSubsequence}^{\uparrow}$ interface packages the discovery-channel selector packet used when regular Cauchy data are consumed through a cofinal subsequence. Following Bishop and Bridges, *Constructive Analysis*, 1985, the selector is a displayed finite row: it records only inspected precision requests, their chosen indices, monotonicity checks, cofinal witnesses for the inspected thresholds, and the regular rational observations carried through those rows. It does not choose a host-level infinite subsequence, identify quotient streams, or appeal to ambient completeness.

Its dependency list is $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. The selector packet consumes the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38, so its visible rows are tied to the real L10 siblings rather than to a kernel-only island.

It also consumes the regular-Cauchy subsequence carrier boundary of chapter 381 and restricts it to the finite selector ledger $(S, \phi, W, D, R, E, H, C, P, N)$. Thus every cofinal read is a displayed NameCert row rather than a host subsequence function.

Definition 405.1 (CofinalSubsequence carrier)**.** *A* $\mathsf{CofinalSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, \phi, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *S is the accepted* $\mathsf{RegularCauchyName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose precision windows may be inspected;*
2. *ϕ is a unary selector row assigning each displayed precision request to a selected source index, together with monotonicity and cofinality witnesses for the inspected threshold rows;*
3. *W records the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows before and after applying ϕ;*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, comparison, and precision observations attached to the selected windows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads the selected rational observations as regular rational data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed after the selected windows, dyadic observations, and regular rational handoff have been displayed;*

7. *H records the displayed* hsame *transports for source, selector, window, observation, regularity, and seal rows;*
8. *C records the* Cont *routes through the same rows;*
9. *P is the* Pkg *provenance over the visible dependencies;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalSubsequence}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through these rows and their transported finite read paths. In particular, ϕ is a unary finite selector ledger, not a host subsequence function.

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceTasteGate single carrier alignment`

Theorem 405.2 (CofinalSubsequence NameCert obligations)**.** *Every accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *carrier of Definition 405.1 carries the following five NameCert obligation rows:*

1. *carrier habitation: the packet stores the source row, unary selector row, selected finite windows, dyadic observations, regular rational handoff, real seal, transport rows, continuation rows, package provenance, and local naming row;*
2. *selector discipline: the row ϕ is unary, monotone on displayed precision requests, and cofinal on every inspected threshold row;*
3. *finite-window exactness: each selected* $\mathsf{StreamName}^{\uparrow}$ *window is read from S only along ϕ, and its dyadic observation row D is transported with that window;*
4. *regular rational handoff: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R consumes exactly the selected window and dyadic observation rows before any* $\mathsf{Real}^{\uparrow}$ *seal read is exposed;*
5. *non-escape: the displayed* hsame, Cont, Pkg, *and* NameCert *rows exhaust the provenance of the packet, so no quotient stream, host sequence equality, choice-selected subsequence, or completeness datum is exported.*

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier namecert obligations`

Proof. Carrier habitation is obtained by projecting the coordinates of Q. The finite history row names the source, selector, selected windows, dyadic observations, regular rational handoff, seal, and provenance rows in one packet.

Selector discipline is read from the ϕ coordinate. Its domain is the displayed unary precision ledger, and its stored comparison rows certify monotonicity and cofinality only for inspected thresholds.

Finite-window exactness follows by replaying the continuation route from S through ϕ to W. The dyadic row D is attached to that selected window, so the endpoint, radius, comparison, and precision observations move with the same finite route.

The regular rational handoff is the projection to R. Since R is stored after W and D, every regular rational read is reached through the selected window and its dyadic observation ledger before the real seal row E is available.

Finally inspect the classifier. It contains no coordinate for quotient-stream equality, host sequence equality, an externally chosen infinite subsequence, or ambient completeness, and the rows H, C, P, and N account for all visible transport and provenance. □

Theorem 405.3 (CofinalSubsequence RegSeqRat consumer boundary)**.** *Let $Q = (S, \phi, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer*

read from Q factors through the displayed cofinal selector ϕ, the selected finite $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, the* $\mathsf{Real}^{\uparrow}$ *seal E, and the visible* hsame, Cont, Pkg, *and* NameCert *rows H, C, P, and N. Thus the consumer boundary is a finite cofinal selector certificate over regular Cauchy data, not a quotient stream, host sequence equality, choice-selected subsequence, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier regseqrat consumer boundary`

Proof. Start with a consumer read from the accepted carrier. The source row S can be inspected only through the unary selector coordinate ϕ, because no other coordinate names a selected index.

Apply the selector and finite-window clauses of Definition 405.2. They force the selected $\mathsf{StreamName}^{\uparrow}$ window W and its $\mathsf{DyadicRatCore}^{\uparrow}$ observation row D to travel along the same monotone cofinal route.

Next use the regular rational handoff clause. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches R only after the selected window and dyadic observation rows have been displayed, and a $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer reaches the real-facing seal only through the same R and E rows.

The remaining coordinates are provenance rows. The displayed hsame transports, Cont routes, Pkg certificate, and local NameCert row exhaust the consumer-visible ledger, so the carrier provides no further dependency outside the listed finite selector packet. □

Theorem 405.4 (CofinalSubsequence selector transport exactness)**.** *Let $Q = (S, \phi, W, D, R, E, H, C, P, N)$ and $Q' = (S', \phi', W', D', R', E', H', C', P', N')$ be accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *carriers. If the source row, unary selector row, selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows are transported componentwise by the displayed* hsame *coordinates, then Q and Q' determine the same accepted cofinal selector packet for every classifier-facing consumer read.*

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier selector transport exactness`

Proof. Project the accepted carriers to the rows listed in Definition 405.1. The classifier has no coordinate outside the source, selector, window, observation, handoff, seal, transport, continuation, package, and naming rows.

Use the selector discipline and finite-window clauses of Definition 405.2. They move the source row through the unary selector before any selected window is read, and they attach the dyadic observation row to that same selected window.

Transport the handoff and seal rows along the displayed hsame coordinates. The Cont rows keep the consumer route ordered from source to selector, from selector to window, from window to dyadic observation, and then to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal.

Finally apply the consumer boundary of Definition 405.3. Since every public read is exhausted by those transported rows, the accepted cofinal selector packet is unchanged at the classifier-facing surface. □

Theorem 405.5 (CofinalSubsequence window handoff exactness)**.** *For every accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *carrier $Q = (S, \phi, W, D, R, E, H, C, P, N)$ and every displayed precision request in its finite ledger, the selected* $\mathsf{StreamName}^{\uparrow}$ *window is reached through the unary cofinal selector ϕ, then its* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row is read, and only then the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R is available to a downstream consumer. No* $\mathsf{Real}^{\uparrow}$ *seal row is read before this selected-window and regular-rational handoff route has been displayed.*

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier window handoff exactness`

Proof. Start from a displayed precision request in the carrier. The selector discipline row of Definition 405.2 forces the request to pass through the unary selector ϕ and its inspected monotonicity and cofinality witnesses.

Apply the finite-window exactness row. It reads the selected $\mathsf{StreamName}^{\uparrow}$ window from S along ϕ and carries the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, comparison, and precision observations with that selected window.

Next use the regular rational handoff row of the same theorem. The $\mathsf{RegSeqRat}^{\uparrow}$ coordinate R consumes exactly the selected window and dyadic observation rows, so the handoff cannot be reached through an unselected stream window.

The consumer boundary of Definition 405.3 then places the $\mathsf{Real}^{\uparrow}$ seal after R and E. Thus every seal-facing read follows the selected-window, dyadic-observation, and regular-rational handoff route first. □

Theorem 405.6 (CofinalSubsequence public export package)**.** *The carrier of Definition 405.1, the NameCert obligations of Definition 405.2, the* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer boundary of Definition 405.3, the selector transport exactness of Definition 405.4, and the window handoff exactness of Definition 405.5 export one public finite selector surface through*

$$S, \phi, W, D, R, E, H, C, P, N.$$

Every downstream $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer receives the selected finite windows, dyadic observation rows, regular rational handoff, real seal, displayed transports, continuation routes, package provenance, and local naming row through that surface. It receives no quotient stream, host sequence equality, chosen infinite subsequence, or ambient completeness object.*

Proof. Start with the carrier inventory. The public coordinates are exactly $S, \phi, W, D, R, E, H, C, P, N$, and the NameCert obligations specify how each of these rows is admitted and routed.

The consumer boundary and selector transport exactness then identify the only route available to downstream readers. A consumer must pass from the source row through the unary selector, selected windows, dyadic observations, regular rational handoff, and real seal, with the visible transport and continuation rows preserving that order.

Finally apply the window handoff exactness theorem. It places every $\mathsf{RegSeqRat}^{\uparrow}$ handoff after the selected window and dyadic observation rows and places the real-facing seal after that handoff. Since the carrier has no coordinate for the excluded objects, the public export is precisely the finite selector surface listed above. □

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier public export package`

Theorem 405.7 (CofinalSubsequence finite-selector bridge schema)**.** *The public finite selector surface*

$$S, \phi, W, D, R, E, H, C, P, N$$

bridges the concrete $\mathsf{CofinalSubsequence}^{\uparrow}$ *carrier of Definition 405.1 to the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer schema. Every read is routed through the unary selector, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Proof. Begin with the local NameCert obligations. They admit the carrier rows and bind the unary selector, selected finite windows, dyadic observations, regular rational handoff, real seal, transports, routes, provenance, and local naming row as one finite packet.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ consumer boundary, selector transport exactness, and window handoff exactness. Together these force every downstream read to pass through the unary selector and selected windows before the dyadic observation and regular rational handoff are available.

Finally apply the public export package. It identifies the finite selector surface $S, \phi, W, D, R, E, H, C, P, N$ as the only public surface for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers. Since the carrier has no coordinate for a quotient stream, host sequence equality, selected infinite subsequence, or ambient completeness object, the bridge exports none of them. □

Lean4 ✓ `BEDC.Derived.CofinalSubsequenceUp.CofinalSubsequenceCarrier bridge selector schema`

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalSubsequence}^{\uparrow}$ packet is generated from regular Cauchy or regular rational source data, a displayed unary monotone cofinal selector row, selected finite stream windows, dyadic observation rows, a regular rational handoff, a real seal, and visible provenance.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalSubsequence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 405.1, Definition 405.2, Definition 405.3, Definition 405.4, Definition 405.5, Definition 405.6, and Definition 405.7 bind the bridged finite selector surface through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, host sequence equality, selected infinite subsequences, uniqueness of completion limits, ambient completeness, or bridge checking.

Upgrade path: No further paper-side theory upgrade remains; Lean verification and bridge checking stay on the formal axis.

Chapter 406

A Concrete Naming Certificate for $\mathsf{CauchySequenceSpace}^{\uparrow}$

The $\mathsf{CauchySequenceSpace}^{\uparrow}$ interface records the finite naming surface for the space of $\mathsf{CauchySequence}^{\uparrow}$ names itself. Following Bishop and Bridges, *Constructive Analysis*, and Bourbaki, *General Topology*, the packet keeps a displayed family of regular rational names together with the finite schedules and tolerance ledgers that make common Cauchy windows inspectable. It sits between $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$, and it exports only BHist, hsame, Cont, Pkg, and local NameCert rows.

This dependency is the concrete sibling route: the displayed regular-family row is consumed from chapter 301, the schedule row from chapter 126, and the completion handoff row from chapter 360.

The equality-facing finite-window classifier of chapter 1820 is consumed when a $\mathsf{CauchySequenceSpace}^{\uparrow}$ comparison is read as a common-window comparison before the $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff.

Definition 406.1 (CauchySequenceSpace carrier surface)**.** *A* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

with the following visible rows:

1. *F is a displayed finite family of* $\mathsf{RegSeqRat}^{\uparrow}$ *name rows;*
2. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that selects the inspected name and window rows;*
3. *W is a* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal-window ledger for the family F;*
4. *ϵ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the inspected windows;*
5. *Q is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff row reached from the same family, schedule, window, and tolerance data;*
6. *H records componentwise* hsame *transports for the family, schedule, cofinal-window ledger, tolerance ledger, and completion handoff;*
7. *C records the displayed* Cont *routes by which consumers inspect those rows;*
8. *P is the* Pkg *provenance over the displayed regular names, stream schedule, Cauchy window data, dyadic tolerances, and completion handoff;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row.*

The carrier contains no quotient of Cauchy names, no selected limiting name, no host completion equality, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.TasteGate.CauchySequenceSpaceTasteGate single carrier alignment`

Theorem 406.2 (CauchySequenceSpace NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier habitation by a displayed family of* $\mathsf{RegSeqRat}^{\uparrow}$ *names, a* $\mathsf{StreamName}^{\uparrow}$ *schedule, a* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal-window ledger, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff row, and the displayed* hsame, Cont, Pkg, *and local naming rows;*
2. *classifier by common finite Cauchy windows, comparing two accepted packets only through windows selected by their schedules and justified by the same visible cofinality ledger shape;*
3. *schedule and cofinality stability under the stored componentwise* hsame *transports and displayed* Cont *routes;*
4. *dyadic tolerance exactness, stating that every accepted window comparison uses the tolerance row visible in the packet;*
5. *completion handoff non-escape, stating that the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *row is reached only through the displayed family, schedule, cofinal-window, tolerance, transport, continuation, package, and naming rows.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier namecert obligation surface`

Proof. Carrier habitation is projection of Definition 406.1. The displayed family, schedule, cofinal-window ledger, tolerance ledger, completion handoff, transport, continuation, package, and naming rows are coordinates of the same finite BHist packet.

The classifier is by common finite windows. A comparison first reads the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ rows, then the cofinal-window evidence supplied by W, and finally the dyadic tolerance row ϵ attached to those windows.

Schedule stability is componentwise. Move the family, schedule, window ledger, tolerance ledger, and completion handoff along H, and replay only the consumer routes recorded in C.

Dyadic tolerance exactness and handoff non-escape follow from the absence of any quotient, selected limit, host equality, or ambient completeness coordinate. The completion row Q is therefore a finite handoff from the named Cauchy-window surface, not a separate completion object. □

Theorem 406.3 (CauchySequenceSpace completion handoff)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal-window ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the stored* hsame, Cont, Pkg, *and* NameCert *provenance rows. Thus the space of Cauchy names reaches completion only through the visible finite naming certificate, not through quotient-name equality, a selected representative, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier completion handoff`

Proof. Start with an accepted carrier $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$. The only family coordinate is F, the only schedule coordinate is σ, and the only Cauchy-window and tolerance coordinates are W and ϵ.

Use Definition 406.2. Its classifier, stability, and exactness rows force every inspected window to remain attached to the displayed schedule, cofinality ledger, dyadic tolerance ledger, and provenance rows.

The completion read reaches Q only after those rows have been read. Since the carrier has no quotient-name equality, selected representative, host real equality, or ambient completeness coordinate, the listed rows exhaust the handoff boundary. □

Theorem 406.4 (CauchySequenceSpace regular-window factorization)**.** *Let S be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N).$$

Every completion-facing consumer read of Q factors through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *family row F, the* $\mathsf{StreamName}^{\uparrow}$ *schedule σ, the cofinal Cauchy-window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ, the componentwise* hsame *transport row H, the displayed* Cont *read routes C, the package row P, and the local* NameCert *row N. No quotient-name equality, selected representative, host real equality, or ambient completeness row participates in this route.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier regular window factorization`

Proof. Project the accepted carrier by Definition 406.1. This identifies the only available coordinates as the displayed family F, schedule σ, window ledger W, tolerance row ϵ, completion handoff Q, transport row H, continuation row C, package row P, and local naming row N.

Use Definition 406.2 for the common-window classifier, schedule and cofinality stability, dyadic tolerance exactness, and completion handoff non-escape rows. These rows keep every inspected regular-name comparison on the same scheduled finite windows and on the same displayed dyadic tolerance data.

Apply Definition 406.3 to the terminal consumer read of Q. The route already reaches the completion handoff only through the named regular family, schedule, Cauchy-window ledger, dyadic tolerance row, transports, continuations, package provenance, and local certificate.

Finally, the carrier projection lists no quotient-name equality, selected representative, host real equality, or ambient completeness coordinate. Thus there is no alternate completion-facing route outside the displayed finite regular-window surface. □

Theorem 406.5 (CauchySequenceSpace common-window classifier transport)**.** *Let $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Transporting S along its displayed* hsame *row H and reading it only through the displayed* Cont *row C preserves the common-window classifier on the same* $\mathsf{RegSeqRat}^{\uparrow}$ *family F,* $\mathsf{StreamName}^{\uparrow}$ *schedule σ,* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal ledger W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ, completion handoff Q, package row P, and local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCommonWindowClassifierTransport`

Proof. Project the tuple by Definition 406.1. The family, schedule, cofinal ledger, tolerance ledger, completion handoff, transport row, continuation row, package row, and local naming row are the whole carrier inventory.

Use the classifier and stability clauses of Definition 406.2. They say that comparisons are made only on common finite Cauchy windows selected by σ and justified by the displayed cofinality row W.

The completion handoff theorem keeps the downstream read on the same packet rows. Hence transporting by H and reading by C cannot change the classifier into a quotient-name comparison, a selected representative comparison, a host real equality, or an ambient completeness assertion. □

Theorem 406.6 (CauchySequenceSpace cofinal schedule stability)**.** *For an accepted packet $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ of* $\mathsf{CauchySequenceSpace}^{\uparrow}$*, every cofinal Cauchy-window request admitted by W remains stable under the* $\mathsf{StreamName}^{\uparrow}$ *schedule σ, the componentwise* hsame *transports H, and the displayed* Cont *routes C. The accepted comparison stays on the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family F, the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ, the handoff row Q, the package row P, and the local* NameCert *row N.*

Proof. Begin with a cofinal request in W. By the carrier definition, W is stored together with the schedule σ and the family F, so the request has no independent schedule coordinate.

Apply the schedule and cofinality stability row of Definition 406.2. The row moves F, σ, W, ϵ, and Q only componentwise along H and replays only the consumer routes recorded in C.

Finally use Definition 406.4. It identifies the same family, schedule, cofinal ledger, tolerance ledger, transport, continuation, package, and naming rows as the entire completion-facing route, so stability is a property of the displayed packet and not of any hidden completion object. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier cofinal schedule stability`

Theorem 406.7 (CauchySequenceSpace dyadic tolerance ledger exactness)**.** *Let $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Every accepted comparison in the common-window classifier uses exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ stored beside the* $\mathsf{RegSeqRat}^{\uparrow}$ *family F,* $\mathsf{StreamName}^{\uparrow}$ *schedule σ,* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger W, completion handoff Q,* hsame *row H,* Cont *row C,* Pkg *row P, and* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier dyadic tolerance ledger exactness`

Proof. The tolerance coordinate of the carrier is the single row ϵ. There is no second dyadic ledger, host tolerance, quotient-name tolerance, or ambient completeness tolerance in Definition 406.1.

Use the dyadic tolerance exactness clause of Definition 406.2. It attaches every accepted window comparison to the tolerance row visible in the same packet.

The completion handoff theorem then carries that exact row to Q together with F, σ, W, H, C, P, and N. Thus the tolerance ledger consumed by completion is precisely ϵ, and no hidden tolerance source participates in the route. □

Theorem 406.8 (CauchySequenceSpace limit-seal factorization)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *limit-seal consumer read of the completion handoff row Q factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family F, the* $\mathsf{StreamName}^{\uparrow}$ *schedule σ, the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ, the completion handoff Q, the* hsame *transport row H, the displayed* Cont *routes C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row N.*

The read cannot use quotient-name equality, a selected representative, host real equality, or ambient completeness outside the displayed finite packet.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier limit seal factorization`

Proof. Project the carrier surface of Definition 406.1. The rows $F, \sigma, W, \epsilon, Q, H, C, P, N$ are the whole accepted packet.

Use the NameCert obligation surface of Definition 406.2. It supplies common-window classification, schedule stability, cofinality stability, dyadic tolerance exactness, and completion handoff non-escape for that same packet.

Apply Definition 406.4, Definition 406.5, Definition 406.6, and Definition 406.7. These results keep the read on the displayed regular family, schedule, cofinal window ledger, and tolerance ledger before it reaches Q.

Finally, Definition 406.3 identifies the terminal route to completion. Since the carrier lists no quotient-name equality, selected representative, host real equality, or ambient completeness row, the limit-seal consumer read has no hidden coordinate through which to bypass the displayed factorization. □

Theorem 406.9 (CauchySequenceSpace RegSeqRat StreamName Real seal factorization)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *limit-seal consumer route and every* $\mathsf{RealCauchyCompletion}^{\uparrow}$*-facing consumer route factors first through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family* F*, then through the* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, then through the cofinal-window ledger* W*, then through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* ϵ*, and only then reaches the completion handoff* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

The resulting route exports no quotient stream, no selected limit, no host real equality, and no ambient completeness coordinate outside $F, \sigma, W, \epsilon, Q, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier regseqrat streamname real seal factorization`

Proof. Start with the regular-family route. Apply Definition 406.4 to expose F as the finite rational family through which completion-facing reads must first pass.

Next attach the schedule and cofinal ledger. Use Definition 406.6 to keep σ and W as the only scheduled window surface available to the consumer.

Then read the tolerance and handoff rows. Apply Definition 406.7 for ϵ and Definition 406.14 for Q, so the completion-facing route cannot reach the handoff before the displayed dyadic tolerance row has been consumed.

Finally apply Definition 406.8. It identifies the $\mathsf{Real}^{\uparrow}$ seal boundary with the same displayed packet and excludes quotient streams, selected limits, host real equality, and ambient completeness coordinates from the consumer route. □

Theorem 406.10 (CauchySequenceSpace two-stage window composition)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

and

$$S' = (F', \sigma', W', \epsilon', Q', H', C', P', N')$$

be accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets. Suppose their regular-family rows, stream schedules, common-window ledgers, dyadic tolerance rows, completion-handoff rows, displayed* hsame *transports, and* Cont *routes are componentwise matched by the local classifier. Then every consumer*

read that first compares S and S' by common finite windows and then hands the result to completion factors through one shared finite window before it reaches Q or Q'.

The factorization uses only F, F', σ, σ', W, W', ϵ, ϵ', Q, Q', H, H', C, C', P, P', and N, N'. It has no route through quotient streams, selected limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier two stage window composition`

Proof. First build the common-window spine. Apply Definition 406.4 to each packet to identify the finite regular-family rows, schedules, cofinal ledgers, tolerance rows, handoff rows, transports, continuations, provenance, and local certificates as the whole read surface.

Next transport the classifier. Use Definition 406.5 to move the two accepted packets componentwise along the displayed hsame rows while replaying only the Cont routes already stored in the packets.

The schedule and tolerance rows now have no independent path. Apply Definition 406.6 for σ, σ', W, W' and Definition 406.7 for ϵ, ϵ'. These two rows bind every tolerated comparison to the same shared finite window before any completion-facing handoff is read.

Finally route the result to the terminal handoff through Definition 406.8. The carrier exposes no quotient stream, selected limit, host real equality, or ambient completeness coordinate, so the composed consumer cannot leave the displayed finite rows. □

Theorem 406.11 (CauchySequenceSpace shared-window completion consumer)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

and

$$S' = (F', \sigma', W', \epsilon', Q', H', C', P', N')$$

be accepted CauchySequenceSpace$^{\uparrow}$ *packets whose regular-family rows, schedules, cofinal-window rows, tolerance rows, and handoff rows classify together through the displayed local classifier. Any completion-facing consumer comparison factors through one shared finite Cauchy window before reading Q or Q'. The route uses only F, F', σ, σ', W, W', ϵ, ϵ', Q, Q', H, H', C, C', P, P', and N, N'.*

Proof. First build the shared window surface. Apply Definition 406.4 to both packets, so every completion-facing read is already restricted to the displayed regular-family, schedule, cofinal-window, tolerance, handoff, transport, continuation, package, and local naming rows.

Next transport the comparison through Definition 406.5. The classifier reads the two packets only by their common finite Cauchy windows and the componentwise hsame and Cont rows already stored in the carriers.

The schedule and tolerance rows then fix the consumer window. Use Definition 406.6 for σ, σ', W, W' and Definition 406.7 for ϵ, ϵ'.

Finally apply Definition 406.10. The comparison reaches Q or Q' only after the shared finite window has been read, and the displayed packet contains no quotient stream, selected limit, host real equality, or ambient completeness route. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier shared window completion consumer`

Theorem 406.12 (CauchySequenceSpace shared-window determinacy)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

and

$$S' = (F', \sigma', W', \epsilon', Q', H', C', P', N')$$

be accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets. Suppose they display the same regular-family row* F*, stream schedule* σ*, common-window ledger* W*, dyadic tolerance row* ϵ*, completion handoff* Q*, componentwise transport* H*, continuation route* C*, package provenance* P*, and local naming row* N*, and suppose their common-window classifiers agree on that displayed window. Then the two packets determine the same completion-facing consumer route before any* $\mathsf{Real}^{\uparrow}$ *seal is read.*

The determined route is the route through $F, \sigma, W, \epsilon, Q, H, C, P, N$*. No quotient stream, selected limit, host equality, or ambient completeness coordinate can distinguish the two completion-facing reads.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier shared window determinacy`

Proof. Project both carriers through Definition 406.1. Under the displayed row equalities, the only available completion-facing coordinates in either packet are the same $F, \sigma, W, \epsilon, Q, H, C, P, N$ rows.

Transport the comparison through Definition 406.5. Since the common-window classifiers agree, the accepted packets read the same finite window before any completion handoff is inspected.

Cofinal schedule stability and dyadic tolerance exactness now fix the routed window. Apply Definition 406.6 to the common σ, W rows and Definition 406.7 to the common ϵ row.

Finally use Definition 406.14 and Definition 406.11. The completion handoff remains the displayed Q row, and the shared-window consumer has no path through quotient streams, selected limits, host equality, or ambient completeness. The two completion-facing routes are therefore the same before any $\mathsf{Real}^{\uparrow}$ seal is read. □

Theorem 406.13 (Cauchy sequence space common-window induction)**.** *Let* $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Every accepted completion-facing comparison over* S *factors through induction on the displayed common-window ledger* W*, preserving the rows* $F, \sigma, \epsilon, Q, H, C, P, N$ *at each induction step. The induction starts from the empty inspected-window ledger, extends by one displayed common Cauchy window at a time, and replays the comparison only through the stored classifier, transport, and handoff rows.*

The resulting route excludes quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness. Those objects cannot be induction hypotheses because they are not rows of the carrier surface.

Proof. Project the carrier through Definition 406.1. The displayed rows available to the induction are exactly $F, \sigma, W, \epsilon, Q, H, C, P, N$.

Induct on the finite common-window ledger W. In the base case there is no inspected comparison outside the displayed family and schedule, so the route is the trivial replay through F, σ, H, C, P, N before any completion handoff is read.

For the step case, append one displayed common Cauchy window. Use Definition 406.5, Definition 406.6, and Definition 406.7 to keep the appended window on the same $F, \sigma, \epsilon, H, C, P, N$ rows.

Finally apply Definition 406.11 and Definition 406.14. They reassemble the completion-facing route through Q after the finite induction has consumed W, and the carrier supplies no coordinate for quotient streams, selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 406.14 (CauchySequenceSpace completion handoff stability)**.** *Let* $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. The completion handoff row* Q *is invariant under*

the same displayed hsame *transports H and* Cont *routes C that preserve the regular-family row F, schedule σ, common-window ledger W, and dyadic tolerance row ϵ. A transported handoff therefore names the same completion-facing finite packet and not a quotient-name comparison, selected limit, host real equality, or ambient completeness assertion.*

Proof. Read the handoff row from the carrier surface. The row Q is stored beside F, σ, W, ϵ, H, C, P, and N and has no separate host-completion coordinate.

Apply Definition 406.3. It states that downstream completion reads consume the same finite regular-family, schedule, window, tolerance, transport, continuation, provenance, and local certificate rows.

Use Definition 406.5 and Definition 406.6 to identify the transported rows componentwise. The dyadic tolerance row is fixed by Definition 406.7, so the handoff row is stable under exactly the displayed H and C routes. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier completion handoff stability`

Theorem 406.15 (CauchySequenceSpace regular family exhaustion)**.** *Every completion-facing consumer read of an accepted* CauchySequenceSpace$^{\uparrow}$ *packet*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

factors through the displayed regular family F, stream schedule σ, common-window ledger W, dyadic tolerance row ϵ, completion handoff Q, transport row H, continuation row C, package provenance P, and local NameCert *row N. No such read factors through quotient streams, selected limits, host real equality, or ambient completeness outside those rows.*

Proof. Begin with the finite packet. The carrier definition exposes exactly $F, \sigma, W, \epsilon, Q, H, C, P, N$ as the data available to a completion-facing consumer.

The window and schedule rows are exhausted by Definition 406.4, Definition 406.5, and Definition 406.6. These theorems keep the consumer on the displayed family, cofinal windows, schedule, transports, and continuations.

The tolerance and terminal rows are exhausted by Definition 406.7, Definition 406.14, and Definition 406.8. Together they force the read through ϵ and Q before any Real$^{\uparrow}$ seal is consumed.

Since no carrier row stores a quotient stream, selected limit, host real equality, or ambient completeness proof, the displayed finite rows are the whole consumer surface. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier regular family exhaustion`

Theorem 406.16 (CauchySequenceSpace diagonal-rate compatibility)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted CauchySequenceSpace$^{\uparrow}$ *packet. Any* RealDiagonalCompletion$^{\uparrow}$ *consumer read whose diagonal selector uses the same scheduled* StreamName$^{\uparrow}$ *windows, and any* CauchyRate$^{\uparrow}$ *consumer read whose effective route uses the same cofinal ledger, factor through the same finite rows $F, \sigma, W, \epsilon, Q, H, C, P, N$ before reaching the* Real$^{\uparrow}$ *seal.*

Thus diagonal finite completeness and effective-rate certification consume the displayed Cauchy-window packet itself. They do not introduce quotient streams, selected limits, host real equality, or ambient completeness data parallel to the accepted CauchySequenceSpace$^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier diagonal rate compatibility`

Proof. Project the accepted carrier through Definition 406.1. The visible coordinates are exactly the displayed family F, schedule σ, cofinal-window ledger W, tolerance row ϵ, completion handoff Q, transport row H, continuation row C, package row P, and local naming row N.

First route the diagonal selector. Apply Definition 406.4, Definition 406.5, and Definition 406.7 to keep every selected diagonal window on the same σ, W, ϵ rows. Then use Definition 359.12 to read the finite diagonal completion interface through those scheduled windows.

Next route the effective-rate consumer. The same common-window classifier and tolerance ledger exactness bind the cofinal ledger and dyadic tolerance row before any rate read can reach completion-facing data. The exported certificate of Definition 415.18 consumes only that effective route through the displayed regular-family, cofinal-ledger, tolerance, transport, continuation, package, and naming rows.

Finally apply Definition 406.8. It identifies the terminal $\mathsf{Real}^{\uparrow}$ seal route through the same packet coordinates. Since the carrier contains no quotient stream, selected limit, host real equality, or ambient completeness coordinate, neither the diagonal completion consumer nor the effective-rate consumer has a hidden route beyond $F, \sigma, W, \epsilon, Q, H, C, P, N$. □

Theorem 406.17 (CauchySequenceSpace obligation-closure package)**.** *The common-window classifier transport, cofinal schedule stability, dyadic tolerance exactness, completion handoff stability, and diagonal-rate compatibility rows form the obligation-closed* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet over the finite carrier surface of Definition 406.1. The closed obligation surface is exactly the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *cofinal-window ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff row,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceUp StdBridge`
↪ variant `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier obligation closure package`

Proof. The carrier surface and NameCert obligation surface are supplied by Definition 406.1 and Definition 406.2. They list the finite family, schedule, cofinal-window ledger, tolerance ledger, completion handoff, transport, continuation, package, and local certificate rows.

Common-window classifier transport is Definition 406.5. Cofinal schedule stability is Definition 406.6, and dyadic tolerance ledger exactness is Definition 406.7. These rows keep every accepted comparison on the same finite Cauchy-window packet.

Completion handoff stability is Definition 406.14. Diagonal-rate compatibility is Definition 406.16. Together they show that completion, diagonal, and effective-rate consumers all remain inside the same displayed packet, so the paper-side obligation surface is closed at the stated grade. □

Theorem 406.18 (CauchySequenceSpace kernel scope packet)**.** *The carrier surface, NameCert obligation surface, common-window classifier transport, cofinal schedule stability, dyadic tolerance exactness, completion handoff stability, regular family exhaustion, and diagonal-rate compatibility assemble into a single kernel-scoped* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Every completion-facing, diagonal-completion, effective-rate, or real-seal consumer must read through the displayed regular-family rows, stream schedule, common-window ledger, dyadic tolerance row, completion handoff, componentwise transports, continuation routes, package provenance, and local naming row. The packet excludes quotient completion, selected representatives, host real equality, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier kernel scope packet`

Proof. Begin with Definition 406.1 and Definition 406.2. They list the displayed family, schedule, cofinal-window ledger, tolerance ledger, completion handoff, transport, continuation, package, and local certificate rows, and they bind the classifier and stability obligations to those rows.

Next apply Definition 406.5, Definition 406.6, and Definition 406.7. These rows keep every accepted comparison on the same scheduled finite windows and the same displayed dyadic tolerance ledger.

Then use Definition 406.14 and Definition 406.15. The terminal completion-facing route and every regular-family consumer are exhausted by the same finite packet coordinates.

Finally apply Definition 406.16 and Definition 406.17. Diagonal finite-completion and effective-rate readers consume the same kernel-scoped surface, so quotient completion, selected representatives, host real equality, and ambient completeness have no row through which to enter. □

Theorem 406.19 (CauchySequenceSpace UniformCauchyCriterion consumption)**.** *Let $S = (F, \sigma, W, \epsilon, Q, H, C, P,$ be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Suppose its common-window ledger W is presented by an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose shared threshold and real-seal handoff read the same regular family, stream schedule, Cauchy-modulus row, dyadic tolerance row, and completion handoff. Then every completion-facing read of S factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *family F,* $\mathsf{StreamName}^{\uparrow}$ *schedule σ,* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ϵ, and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff Q.*

The consumer route has no row for quotient streams, selected limits, host equality, or ambient completeness. The $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *presentation supplies only the shared-threshold ledger and real-seal handoff already consumed by the displayed* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier uniformcriterion consumption`

Proof. Project the accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ carrier by Definition 406.1. The only completion-facing coordinates are the regular family F, schedule σ, common-window ledger W, dyadic tolerance row ϵ, handoff Q, transports, continuations, provenance, and local naming row.

Use Definition 406.5, Definition 406.6, and Definition 406.7. These rows keep the common-window classifier, schedule, cofinal Cauchy-modulus evidence, and dyadic tolerance on the displayed packet before completion-facing data can be read.

Now read the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ presentation through Definition 435.10 and Definition 435.57. The shared threshold and real-seal handoff are consumer routes over the same finite family, schedule, modulus, tolerance, and seal rows; they do not introduce an independent completion object.

Finally apply Definition 406.14 and Definition 406.18. The completion handoff stays on $F, \sigma, W, \epsilon, Q, H, C, P, N$, so the uniform-criterion ledger is consumed as a presentation of the same route rather than as quotient completion, a selected limit, host equality, or ambient completeness. □

Theorem 406.20 (CauchySequenceSpace modulus-family factorization)**.** *Let $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$ be an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Any completion-facing consumer of S that uses a* $\mathsf{CauchyModulus}^{\uparrow}$ *row or a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row must first factor through the displayed regular family F, the schedule σ, the modulus or cofinal-window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row ϵ, and the completion handoff Q before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal is read.*

The factorization route is

$$F, \sigma, W, \epsilon \longmapsto Q \longmapsto H, C, P, N.$$

It has no coordinate for quotient streams, selected limits, host equality, detached modulus selectors, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier modulus family factorization`

Proof. Project the accepted carrier through Definition 406.1. The displayed packet stores exactly the family F, schedule σ, common-window or modulus ledger W, dyadic tolerance row ϵ, completion handoff Q, transports, continuations, provenance, and local name.

Apply Definition 406.19 when the consumer presents W through a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row. The shared-threshold and real-seal reads still use the same regular family, schedule, modulus evidence, dyadic tolerance, and completion handoff already present in S.

For a direct $\mathsf{CauchyModulus}^{\uparrow}$ read, use Definition 406.6 and Definition 406.7. These theorems keep the modulus request on the displayed schedule, window ledger, and dyadic tolerance before Q can be consumed.

Finally use Definition 406.14 and Definition 406.18. The public completion handoff stays on $F, \sigma, W, \epsilon, Q, H, C, P, N$, so quotient streams, selected limits, host equality, detached modulus selectors, and ambient completeness have no route through the carrier. □

Theorem 406.21 (CauchySequenceSpace RealObservationBudget factorization)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Suppose a* $\mathsf{RealObservationBudget}^{\uparrow}$ *presentation records the same precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and radii,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows as the completion-facing request for S. Then that observation-budget presentation factors through the displayed CauchySequenceSpace rows before any completion-facing read of Q.*

The factorized route uses only the regular-family row F, schedule σ, common-window ledger W, dyadic tolerance row ϵ, completion handoff Q, componentwise transports H, continuation routes C, provenance row P, and local naming row N. It cannot use quotient streams, host equality, hidden limit selection, unbounded schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier real observation budget factorization`

Proof. Project the accepted carrier through Definition 406.1. The visible packet rows are exactly the regular rational family, stream schedule, common-window ledger, dyadic tolerance row, completion handoff, transports, continuations, provenance, and local name.

First bind the observation windows to the same CauchySequenceSpace route. Common-window classifier transport, cofinal schedule stability, and dyadic tolerance ledger exactness identify the precision-indexed windows, stream reads, dyadic readbacks, and displayed radii with W, σ, and ϵ.

Next route the source and seal consumers. Completion handoff stability keeps the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal boundary on the same Q row, and the kernel scope packet prevents any completion-facing read from leaving $F, \sigma, W, \epsilon, Q, H, C, P, N$.

Finally read the $\mathsf{RealObservationBudget}^{\uparrow}$ carrier through Definition 611.1, Definition 611.3, Definition 611.4, Definition 611.5, and Definition 611.6. Those rows present the same finite observation route and exclude quotient streams, host equality, hidden limit selection, unbounded schedules, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier public observation budget factorization`

Theorem 406.22 (CauchySequenceSpace scoped completion route inventory)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Every completion-facing consumer read, and every* $\mathsf{Real}^{\uparrow}$ *limit-seal read of the completion handoff, is scoped by the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family row* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, the* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyModulus}^{\uparrow}$ *window ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* ϵ*, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff* Q*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row* N*.*

Consequently the completion-facing route inventory of S *is a scoped BEDC route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It exports no quotient-name equality, selected representative, host real equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier scoped completion route inventory`

Proof. Start from the carrier surface in Definition 406.1 and the NameCert obligation surface in Definition 406.2. These two rows name the accepted tuple and bind its family, schedule, window ledger, tolerance row, completion handoff, transports, continuations, provenance, and local certificate to the same finite BHist packet.

Next isolate the completion-facing route. Apply Definition 406.3 and Definition 406.4. The terminal handoff to $\mathsf{RealCauchyCompletion}^{\uparrow}$ can only be read after the displayed regular family, scheduled finite windows, Cauchy ledger, dyadic tolerance, transport, continuation, package, and naming rows have been consumed.

Now bind the RealUp seal boundary. Use Definition 406.8 to route every limit-seal read through $F, \sigma, W, \epsilon, Q, H, C, P, N$, and use Definition 406.15 to show that no other regular-family consumer row remains outside the displayed packet.

The route inventory is therefore a scoped BEDC route over the primitive rows BHist, hsame, Cont, Pkg, and NameCert. Since the carrier surface supplies no quotient-name equality, selected representative, host real equality, or ambient completeness coordinate, none of those objects is exported by the completion-facing or limit-seal route. □

Theorem 406.23 (CauchySequenceSpace public completion export)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. The public completion export of* S *is the single BEDC surface consisting of the finite carrier rows* $F, \sigma, W, \epsilon, Q, H, C, P, N$*, the* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *obligation rows, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff row, common-window classifier transport, dyadic tolerance exactness, completion handoff stability, regular-family exhaustion, and the scoped route inventory of Definition 406.22.*

Every completion-facing consumer of S *must factor through that public export before reading a completion handoff or a* $\mathsf{Real}^{\uparrow}$ *seal. The export is public only as a BEDC route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*; it does not assert quotient completion, equality of Cauchy-name classes, uniqueness of limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier public completion export`

Proof. Start with Definition 406.1 and Definition 406.2. These identify the visible tuple $F, \sigma, W, \epsilon, Q, H, C, P, N$ and bind its local naming, transport, continuation, and package rows to the same finite packet.

Next add the completion-facing rows. Apply Definition 406.3, Definition 406.5, and Definition 406.7 to keep the handoff, classifier, windows, and dyadic tolerance on the displayed coordinates.

Then use Definition 406.14 and Definition 406.15. They show that completion reads and regular-family consumers remain inside the same finite route.

Finally package those rows by Definition 406.22. The cited inventory is already a BEDC route over BHist, hsame, Cont, Pkg, and NameCert, so the public export has exactly the stated surface and no stronger completion object. □

Theorem 406.24 (CauchySequenceSpace RealObservationBudget public consumer)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Suppose a* $\mathsf{RealObservationBudget}^{\uparrow}$ *presentation uses the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and radii,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal factors as the completion-facing request for* S*. Then that observation presentation factors through the public completion export of Definition 406.23.*

In particular, the observation consumer reads $F, \sigma, W, \epsilon, Q, H, C, P, N$ *and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *rows before it can present a real observation. It cannot bypass the public export by selecting a quotient name, a representative limit, a host real equality proof, an ambient completeness principle, or an unbounded schedule.*

Proof. First project the $\mathsf{RealObservationBudget}^{\uparrow}$ presentation through Definition 611.1. The relevant observation rows are the finite request, window, dyadic readback, handoff, seal, transport, and naming data.

Use Definition 611.3, Definition 611.4, and Definition 611.5 to align those rows with the same finite windows, regular-rational handoffs, dyadic readbacks, and $\mathsf{Real}^{\uparrow}$ seal factors named by S.

Now apply Definition 406.21. It already routes such an observation-budget presentation through $F, \sigma, W, \epsilon, Q, H, C, P, N$ before the completion handoff is read.

Finally compose with Definition 406.23 and Definition 611.6. The first theorem supplies the public export for the routed packet, and the second excludes quotient names, selected representatives, host equality, ambient completeness, and unbounded schedules from the observation route. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier public real observation consumer`

Theorem 406.25 (CauchySequenceSpace public nonescape)**.** *For every accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet* $S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$*, the public completion export of Definition 406.23 exposes no quotient-name equality, no selected representative, no host real equality, no ambient completeness principle, and no hidden limit selector. Its observable routes are exhausted by the displayed finite carrier, obligation, completion handoff, classifier transport, tolerance, stability, family-exhaustion, and scoped-inventory rows.*

Proof. The public export is defined by Definition 406.23 as a package of previously displayed BEDC rows. Each row is a carrier, naming, transport, continuation, package, tolerance, handoff, stability, exhaustion, or scoped-route row over the tuple $F, \sigma, W, \epsilon, Q, H, C, P, N$.

Apply Definition 406.18 and Definition 406.22. These two results state that completion-facing and limit-seal consumers are scoped by the displayed BEDC packet and that quotient completion, selected representatives, host real equality, and ambient completeness have no exported coordinate.

It remains to exclude a hidden limit selector. The only limit-seal route available to the export is the factorization of Definition 406.8, which routes through $F, \sigma, W, \epsilon, Q, H, C, P, N$. Since that route is already included in the scoped inventory and supplies no selector coordinate, the public export cannot expose one. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier public nonescape`

Theorem 406.26 (CauchySequenceSpace tail-window factorization)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Every tail-window completion-facing read of* S *factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, the cofinal Cauchy-window ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* ϵ*, the completion handoff* Q*, the transport row* H*, the continuation row* C*, the package row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row* N*. The route reaches completion and any* $\mathsf{Real}^{\uparrow}$ *seal only through those finite BEDC rows, not through quotient-name equality, selected representatives, host real equality, ambient completeness, or a hidden limit selector.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier tail window factorization`

Proof. Begin with the public completion surface. The public completion export theorem Definition 406.23 packages the visible tuple $F, \sigma, W, \epsilon, Q, H, C, P, N$ together with the completion handoff and scoped route inventory, while public nonescape Definition 406.25 excludes quotient, representative, host equality, completeness, and selector coordinates.

Next isolate the finite tail window. Regular-window factorization Definition 406.4 routes the completion-facing read through the displayed regular family, schedule, cofinal ledger, tolerance row, transport, continuation, package, and naming rows. Cofinal schedule stability Definition 406.6 keeps every cofinal request attached to the same schedule and ledger.

Finally read the tolerance and terminal handoff. Dyadic tolerance ledger exactness Definition 406.7 fixes the tolerance row before the completion handoff is consumed, and the public export supplies the handoff, transport, continuation, package, and NameCert rows. Thus every tail-window completion-facing read factors through the displayed finite route. □

Theorem 406.27 (CauchySequenceSpace quotient refusal route)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Any accepted consumer that asks for quotient-name equality, a selected representative, host real equality, or ambient completeness must route through chapter 998 after it has read the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, the cofinal-window ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* ϵ*, and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff* Q*.*

Thus the quotient-facing request is not a projection from $\mathsf{CauchySequenceSpace}^{\uparrow}$ *itself. It is a refusal route over the same finite Cauchy-window packet, with no additional quotient stream, selected limit, host equality, or completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier quotient refusal route`

Proof. First read the completion handoff of Definition 406.3. It forces every completion-facing consumer to pass through the displayed family, schedule, window, tolerance, transport, continuation, package, and local naming rows before the handoff row Q is consumed.

Next use public non-escape and the tail-window factorization. By Definition 406.25 and Definition 406.26, quotient-name equality, selected representatives, host real equality, ambient completeness, and hidden limit selectors are absent from the $\mathsf{CauchySequenceSpace}^{\uparrow}$ carrier and from its public completion export.

Finally apply the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ route of Definition 998.3. That sibling chapter is exactly the Real-completion refusal packet for requests that leave the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. Therefore a quotient-facing consumer may proceed only as a refusal-route consumer after the CauchySequenceSpace rows have been read. □

Theorem 406.28 (CauchySequenceSpace RealLimitUniquenessSeal public consumer)**.** *Let*

$$S_0 = (F_0, \sigma_0, W_0, \epsilon_0, Q_0, H_0, C_0, P_0, N_0)$$

and

$$S_1 = (F_1, \sigma_1, W_1, \epsilon_1, Q_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets. Suppose their public completion exports supply two* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff rows and that the comparison route displays the same* $\mathsf{RegSeqRat}^{\uparrow}$ *family row, the same finite* $\mathsf{StreamName}^{\uparrow}$ *window row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *classifier row for both exported seals. Then any* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *consumer comparing those two exported seals factors through exactly the two public completion exports, the shared comparison-window-tolerance route, and the two local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *rows.*

The consumer cannot read a selected limit, quotient stream equality, host real equality, detached selector choice, or ambient completeness theorem from the two $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets. Its only uniqueness-facing surface is the finite two-seal packet routed through* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceRealLimitUniquenessSealPublicConsumer`

Proof. First expose the two public completion endpoints. Apply Definition 406.23 separately to S_0 and S_1. The two applications provide the two $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff rows together with their displayed family, stream-window, dyadic tolerance, transport, continuation, package, and local NameCert rows.

Next restrict the comparison route to the shared finite data. Use Definition 406.25 to remove quotient-name equality, selected representatives, host equality, ambient completeness, and hidden limit selectors from both public exports. Then apply Definition 406.26 to keep each endpoint on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ family, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and terminal handoff route.

Finally read the uniqueness consumer through the sibling comparison packet. The obligation surface of Definition 1039.2 supplies exactly the two seal rows, shared regular-rational comparison row, finite window row, dyadic ledger, terminal $\mathsf{Real}^{\uparrow}$ classifier row, uniqueness handoff, transport, continuation, provenance, and local name. The nonescape theorem Definition 1039.3 prevents any replacement by a selected limit, quotient stream equality, host Real equality, detached selector, or ambient completeness theorem. Hence the consumer factors only through the two public exports, the shared comparison-window-tolerance route, and the local NameCert rows. □

Theorem 406.29 (CauchySequenceSpace finite-section pullback correspondence)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Suppose that the public completion export of* S *is read through an accepted finite request section of* $\mathsf{StreamName}^{\uparrow}$*, and that a* $\mathsf{Real}^{\uparrow}$ *seal consumer reads the same finite windows, dyadic tolerance rows, regular-rational handoff rows, terminal seal row, displayed* BHist *entry, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row. Then the exported completion handoff and the* $\mathsf{Real}^{\uparrow}$ *seal consumer are exactly the same four-face route named by the StreamName finite-section pullback exactness theorem:*

$$\mathsf{StreamName}^{\uparrow}, \qquad \mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

Equivalently, every such CauchySequenceSpace completion read pulls back to the StreamName finite section with the displayed window face, dyadic face, regular-rational face, and Real seal face, and then returns to the same public completion export. The correspondence exports no quotient stream, no selected representative, no host real equality, no countable-choice row, no ambient completeness principle, and no route-level aggregate closure or verification status.

Proof. First expose the public completion side. Apply Definition 406.23 to S to obtain the displayed tuple $F, \sigma, W, \epsilon, Q, H, C, P, N$, the $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff row, and the local $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ obligation rows.

Next restrict the read to finite StreamName data. Use Definition 406.26 to present the family, schedule, tolerance, and handoff rows as the same finite $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route that the accepted request section consumes.

Now invoke the StreamName pullback surface. Theorem 126.586 identifies the four visible faces of the request section, and Definition 126.589 keeps the Real seal, regular-rational handoff, dyadic tolerance row, and finite StreamName window locked to the same BHist, hsame, Cont, Pkg, and NameCert data.

Finally apply Definition 406.25 and Definition 38.494. The first result excludes quotient streams, selected representatives, host real equality, countable-choice routes, and ambient completeness from the CauchySequenceSpace export. The second keeps the closure and verification coordinates objectwise on the four faces rather than converting them into a route-level aggregate status. □

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier finite section pullback correspondence`

Theorem 406.30 (CauchySequenceSpace exact-boundary terminal pullback)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet, and let*

$$B = (L_B, K_B, J_B, S_B, W_B, R_B, D_B, E_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealCompletionExactBoundaryUp}$ *packet. Suppose the public completion export of* S *and the terminal consumer route of* B *are displayed over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the same terminal* $\mathsf{Real}^{\uparrow}$ *seal, the same* BHist *entry, the same componentwise* hsame *transports, the same* Cont *routes,*

the same Pkg *provenance, and the same local NameCert rows. Then the CauchySequenceSpace public completion read and the RealCompletionExactBoundary terminal consumer are one four-face terminal pullback route:*

$$\mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

This identification reads the finite Cauchy-name export through the exact-boundary terminal consumer and returns to the same public completion surface. It exports no quotient stream, no selected representative, no host real equality, no countable-choice row, no ambient completeness row, no metric-completeness row, and no bridge row.

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier real completion exact boundary terminal pullback`

Proof. First expose the CauchySequenceSpace public completion surface. Apply Definition 406.23 to S to obtain the displayed family, schedule, cofinal-window, dyadic tolerance, completion handoff, transport, continuation, provenance, and local naming rows. Then apply Definition 406.29 to identify that public read with the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route over the shared BHist, hsame, Cont, Pkg, and NameCert data.

Next read the RealCompletionExactBoundary side. Theorem 1120.6 says that the accepted boundary exports exactly the four terminal faces W_B, R_B, D_B, E_B after the governing rows L_B, K_B, J_B, S_B. Theorem 1120.13 places every completion-facing and terminal $\mathsf{Real}^{\uparrow}$ consumer on the same finite route through $L_B, K_B, J_B, S_B, W_B, R_B, D_B, E_B, H_B, C_B, P_B, N_B$.

Finally compare the two exclusions. The CauchySequenceSpace public nonescape theorem Definition 406.25 removes quotient streams, selected representatives, host real equality, countable-choice routes, and ambient completeness from the public export. The exact-boundary no-completeness theorem Definition 1120.14 removes quotient streams, host equality, selected limits, metric-completeness rows, bridge rows, and ambient-completeness coordinates from the terminal consumer. Since the hypotheses identify the displayed window, readback, tolerance, Real seal, history, transport, continuation, package, and local NameCert rows, no additional coordinate remains on either side; both consumers are the same four-face terminal pullback route. □

Theorem 406.31 (CauchySequenceSpace terminal pullback exactness)**.** *Let*

$$S = (F, \sigma, W, \epsilon, Q, H, C, P, N)$$

be an accepted $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet, and let an accepted L10 Phase Real and Completion read of S display a finite* $\mathsf{StreamName}^{\uparrow}$ *section, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, a* BHist *entry, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$ *row. This read is the L10 terminal exit route completely determined by the Phase Real and Completion terminal data if and only if the following two projections hold over those same rows:*

1. *its* $\mathsf{StreamName}^{\uparrow}$ *finite-section projection satisfies the pullback exactness criterion of Definition 126.586;*
2. *its* RealCompletionExactBoundaryUp *projection satisfies the terminal pullback of Definition 406.30.*

Thus the terminal exit route has no coordinate beyond the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. In particular it exports no quotient stream, no selected representative, no host real equality, no countable-choice row, no ambient completeness row, no metric-completeness row, no bridge row, and no route-level aggregate closure or verification status.*

Lean4 ✓ `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier terminal pullback exactness`

Proof. For the forward direction, assume the accepted read is already the L10 terminal exit route determined by the Phase Real and Completion terminal data. Apply Definition 406.29 to expose the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ four-face route carried by the public completion export. Then use Definition 126.586 to identify the finite-section projection over the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

The same terminal read also supplies the exact-boundary endpoint. Apply Definition 406.30 to read the RealCompletionExactBoundaryUp projection through the same finite window, readback row, tolerance row, terminal Real seal, history, transport, continuation, package, and local NameCert data.

For the reverse direction, suppose the two displayed projections satisfy those exactness criteria over the same rows. The finite-section correspondence first pins the CauchySequenceSpace public completion read to the StreamName four-face route, while the StreamName theorem makes that route terminal among finite request sections. The exact-boundary terminal pullback then identifies the RealCompletionExactBoundary consumer with the same four faces and support rows.

Finally apply Definition 38.494. Since the StreamName projection and the exact-boundary projection share the same objectwise $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows, the L10 exit is determined by those two projections. The nonescape clauses in the cited CauchySequenceSpace, StreamName, and exact-boundary theorems exclude quotient streams, selected representatives, host real equality, countable choice, ambient completeness, metric-completeness rows, bridge rows, and any route-level aggregate closure or verification status. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceSpace}^{\uparrow}$ packet is generated from a displayed family of regular rational names, a stream schedule, finite Cauchy-window evidence, dyadic tolerance data, a completion handoff, componentwise transport, displayed continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 406.23, Definition 406.25, and Definition 406.26 supply the mature paper surface: a public finite Cauchy-name route with the sibling $\mathsf{RegSeqRat}^{\uparrow}$ family, $\mathsf{StreamName}^{\uparrow}$ schedule, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ handoff displayed in the chapter.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceSpace}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchySequenceSpaceUp.CauchySequenceSpaceCarrier\_public\_completion\_export`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient completion construction, equality of Cauchy-name classes, uniqueness of limits, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: No further upgrade – chapter at top of the formal bridge axis.

Chapter 407

A Concrete Naming Certificate for $\mathsf{SqueezeReal}^{\uparrow}$

The $\mathsf{SqueezeReal}^{\uparrow}$ interface names the finite dyadic sandwich packet used when a constructive squeeze argument is consumed by $\mathsf{Real}^{\uparrow}$ and completion-facing certificates. The mathematical squeeze theorem is usually stated as a convergence principle for a middle sequence bounded between two convergent endpoints; BEDC records only the visible finite certificate: lower and upper $\mathsf{RegularCauchyName}^{\uparrow}$ rows, a middle $\mathsf{RegularCauchyName}^{\uparrow}$ row, a shared $\mathsf{DyadicTailBound}^{\uparrow}$ tolerance row, sandwich ledgers, a $\mathsf{Real}^{\uparrow}$ seal row, package provenance, and a local $\mathsf{NameCert}$ row.

The packet is not a quotient of streams, not an ambient equality of host reals, and not a choice-selected limit. It is a finite ledger surface over $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Definition 407.1 (SqueezeReal finite sandwich carrier)**.** *A* $\mathsf{SqueezeReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (l, x, u, \epsilon, L, U, S, E, P, N)$$

with the following visible rows:

1. *l is the lower* $\mathsf{RegularCauchyName}^{\uparrow}$ *row, read through its* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *observation surface;*
2. *x is the middle* $\mathsf{RegularCauchyName}^{\uparrow}$ *row whose real-facing seal is to be exposed only through the sandwich packet;*
3. *u is the upper* $\mathsf{RegularCauchyName}^{\uparrow}$ *row, read through the same kind of regular rational observation surface as l;*
4. *ϵ is the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row consumed by all three regular names;*
5. *L is the lower sandwich ledger certifying the displayed finite comparisons $l \leq x$ at every requested dyadic window;*
6. *U is the upper sandwich ledger certifying the displayed finite comparisons $x \leq u$ and the width bound $u - l \leq \epsilon$ at the same requested dyadic windows;*
7. *S is the shared* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the lower and upper ledgers have consumed the shared tolerance row;*
8. *E stores componentwise* hsame *transports for the three regular-name rows, the tolerance row, both sandwich ledgers, the seal row, provenance, and the local naming row;*

9. *P is the* Pkg *provenance naming the consumed* $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed host real, quotient stream equality, a selected limiting value, or an external completeness theorem.

Lean4 ✓ `BEDC.Derived.SqueezeRealUp.SqueezeRealTasteGate single carrier alignment`

Definition 407.2 (SqueezeReal packet classifier)**.** *Two accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *carriers are classified as the same finite sandwich packet exactly when their displayed rows compare componentwise. The lower, middle, and upper* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows agree through their scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *observations, the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *rows agree as the same tolerance request, the lower and upper sandwich ledgers transport through the displayed* hsame *rows, and the* $\mathsf{Real}^{\uparrow}$ *seal,* Pkg *provenance, and local* NameCert *rows name the same packet.*

The classifier has no generator for quotienting regular names, identifying host reals, selecting a limit, or replacing the two sandwich ledgers by a single ambient order proof.

Theorem 407.3 (SqueezeReal NameCert obligations)**.** *The carrier of Definition 407.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ *obligation surface. Its five obligation rows are:*

1. *carrier habitation by any displayed triple of regular dyadic rows whose lower and upper ledgers use the same tolerance request;*
2. *classifier transport by the componentwise classifier of Definition 407.2;*
3. *lower and upper sandwich ledger stability under the displayed* hsame *and* Cont *routes;*
4. *dyadic tail exhaustion, saying that every precision read factors through the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *row before any seal read;*
5. $\mathsf{Real}^{\uparrow}$ *seal boundary, saying that the public real-facing read exposes only the middle regular name after both sandwich ledgers have been consumed.*

Proof. For habitation, take the displayed regular-name rows l, x, and u together with a single tolerance row ϵ. The packet is accepted only when L and U name the finite comparisons over that same tolerance request, so the carrier is a concrete BHist tuple.

For classifier transport, use Definition 407.2. Each regular-name row is transported through its scheduled rational observations, the tolerance row is transported as a shared request, and both sandwich ledgers are repacked with the transported provenance.

For ledger stability and tail exhaustion, inspect any consumer read. A lower or upper comparison read must project L or U, and both ledgers cite ϵ before exposing a dyadic endpoint comparison. Thus the two bounding rows cannot be separated from the shared dyadic tail request.

For the seal boundary, project S only after the previous rows have been read. Since no coordinate of Q is a quotient stream, host real equality, selected limit, or completeness datum, the local $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ surface is exactly the displayed finite sandwich packet. □

Theorem 407.4 (SqueezeReal sandwich exactness)**.** *Every accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet consumes the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row and both bounding ledgers before exposing the* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, a read of S must factor through the rows ϵ, L, and U, where L certifies the lower comparisons $l \leq x$ and U certifies both the upper comparisons $x \leq u$ and the finite width bounds $u - l \leq \epsilon$ for the scheduled dyadic windows.*

Consequently a $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer receives a finite sandwich certificate for the middle regular name, not a host-real equality proof and not a selected limiting object.*

Lean4 ✓ `BEDC.Derived.SqueezeRealUp.SqueezeRealCarrier sandwich exactness`

Proof. Start with an accepted packet $Q = (l, x, u, \epsilon, L, U, S, E, P, N)$. By Definition 407.1, the only tolerance coordinate in the carrier is ϵ, and both ledgers are declared over the same requested dyadic windows.

Read the lower ledger L. It supplies exactly the finite comparisons from the lower regular row to the middle regular row at the scheduled windows; no seal row is projected by this read.

Read the upper ledger U. It supplies the finite comparisons from the middle regular row to the upper regular row and the displayed width bounds against the same tolerance row ϵ.

Only after these two reads may the consumer project S. Coordinate exhaustion shows that the exposed seal is the shared $\mathsf{Real}^{\uparrow}$ row named by the packet, and the consumer has not acquired quotient stream equality, host real equality, a selected limit, or a separate completeness principle. □

Theorem 407.5 (SqueezeReal RealUp boundary)**.** *When a* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or completion consumer cites an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet, the citation can use only the lower regular-name row, middle regular-name row, upper regular-name row, shared dyadic tolerance row, lower sandwich ledger, upper sandwich-and-width ledger, displayed transports, package provenance, and local naming row of Definition 407.1. The citation cannot infer a quotient of presentations, a host real equality, a selected limit, an ambient order-completeness theorem, or a standard bridge row.*

Lean4 ✓ `BEDC.Derived.SqueezeRealUp.SqueezeRealCarrier realup boundary`

Proof. Factor the consumer citation through the local $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ surface of Definition 407.3. The visible packet supplies three regular-name rows, one shared tolerance row, two sandwich ledgers, transports, continuations, provenance, and the local naming row.

Apply the exactness statement of Definition 407.4. Any real-facing read must consume the tolerance row and both bounding ledgers before the seal row is visible.

None of the displayed coordinates is a quotient stream, host real equality, selected limiting value, order-completeness theorem, or standard bridge. The real-facing boundary is therefore the finite dyadic sandwich surface and nothing stronger. □

Theorem 407.6 (SqueezeReal transport stability)**.** *Let* $Q = (l, x, u, \epsilon, L, U, S, E, P, N)$ *be an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet. Componentwise* hsame *transport of the lower, middle, and upper* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows, the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *row, both sandwich ledgers, the* $\mathsf{Real}^{\uparrow}$ *seal row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ *row preserves the carrier of Definition 407.1 and the classifier of Definition 407.2. The transported packet is still classified only by the displayed regular-name observations, the single tolerance request, the two transported ledgers, the seal row, provenance, and local naming data.*

Lean4 ✓ `BEDC.Derived.SqueezeRealUp.SqueezeRealCarrier transport stability`

Proof. Project the carrier coordinates from Definition 407.1. Each transported row named in the statement is already a visible coordinate of Q, and the transport ledger E records the componentwise hsame routes for exactly those coordinates.

The carrier is preserved by repacking the transported lower, middle, and upper regular-name rows with the transported tolerance row, the transported lower and upper ledgers, the transported seal, provenance, and local naming row. No additional coordinate is available in the displayed BHist packet.

The classifier is preserved by Definition 407.2. Classification compares the same rows componentwise after transport: scheduled regular-name observations, the shared tolerance request, both sandwich ledgers, the $\mathsf{Real}^{\uparrow}$ seal, the Pkg row, and the local $\mathsf{NameCert}$ row. □

Theorem 407.7 (SqueezeReal dyadic tail exhaustion)**.** *Every precision read of the real-facing seal in an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet factors first through the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row and then through both sandwich ledgers before any* $\mathsf{Real}^{\uparrow}$ *seal read. Thus a consumer can reach* S *only by the finite route* ϵ, L, U, S *inside the displayed* BHist *packet.*

Proof. Begin with a precision request addressed to the real-facing row. The carrier of Definition 407.1 stores only one dyadic tolerance coordinate, namely ϵ, so every such request has a unique tolerance row to consume.

Next read the sandwich ledgers. The lower ledger L records the finite lower comparisons against the middle regular name, and the upper ledger U records the finite upper comparisons and width bounds against the same tolerance request. These two ledger reads account for the sandwich data available to the request.

Only after those rows have been consumed can the seal coordinate S be projected. This is exactly the route isolated in Definition 407.4, so the precision read has no remaining access to quotient regular names, host equality, selected limits, or ambient completeness. □

Theorem 407.8 (SqueezeReal shared tail ledger determinacy)**.** *Let* $Q = (l, x, u, \epsilon, L, U, S, E, P, N)$ *be an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet. If two precision reads use the same lower, middle, and upper* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows* l*,* x*, and* u*, and the same shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row* ϵ*, then their lower ledger, upper ledger, width-bound surface, and* $\mathsf{Real}^{\uparrow}$ *seal-facing route agree at the classifier-facing packet level. The agreement is only the finite packet agreement of Definition 407.2; it does not identify host real values, quotient regular names, selected limits, or an ambient completeness proof.*

Lean4 ✓ `BEDC.Derived.SqueezeRealUp.SqueezeRealCarrier shared tail ledger determinacy`

Proof. Project the accepted carrier of Definition 407.1. The rows l, x, u, and ϵ are the only inputs from which a precision read can obtain the three regular-name surfaces and the shared dyadic tail request.

Apply the sandwich exactness and dyadic-tail exhaustion rows, Definition 407.4 and Definition 407.7. They force each read to consume the same tolerance row before reading L, U, the width-bound data inside U, and then the seal-facing route S.

If a read is carried through displayed transport before comparison, use Definition 407.6. The transported packet preserves the lower, middle, upper, tolerance, ledger, seal, package, and local naming rows that the classifier can inspect.

The classifier of Definition 407.2 now compares the same visible rows componentwise. By the real-facing boundary of Definition 407.5, there is no additional coordinate from which the two reads could obtain a different host equality, quotient representative, selected limit, or order-completeness route. □

Theorem 407.9 (SqueezeReal completion consumer package)**.** *The carrier and classifier of Definition 407.1 and Definition 407.2, together with the* $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ *obligations of Definition 407.3, sandwich exactness of Definition 407.4, real-facing boundary of Definition 407.5, transport stability of Definition 407.6, dyadic-tail exhaustion of Definition 407.7, and shared tail ledger determinacy of Definition 407.8, form the finite completion-facing packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,*

$\mathsf{Real}^{\uparrow}$*, and completion certificates. The packet excludes quotient regular names, host equality, selected limits, and ambient order completeness.*

Proof. Assemble the carrier and classifier rows first. They identify the finite BHist tuple and the componentwise equality route over the three regular-name rows, the shared tolerance request, both ledgers, the seal, provenance, and local naming data.

Add the naming-certificate obligations and sandwich exactness. These rows show that carrier habitation, classifier transport, ledger stability, tail exhaustion, and seal boundary are all witnessed inside the displayed packet, and that the seal can be read only through ϵ, L, and U.

Finally add transport stability, dyadic-tail exhaustion, and shared tail ledger determinacy. The resulting package is stable under the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows and supplies exactly the finite consumer surface needed by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion certificates. Since all rows have been exhausted, no quotient regular name, host equality, selected limit, or ambient order-completeness coordinate is present. □

Theorem 407.10 (SqueezeReal scoped dependency packet)**.** *For an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet* $Q = (l, x, u, \epsilon, L, U, S, E, P, N)$*, the carrier row, classifier row, sandwich exactness row, transport-stability row, and dyadic-tail exhaustion row factor through exactly the displayed dependencies*

$$\mathsf{DyadicTailBound}^{\uparrow},\quad \mathsf{RegularCauchyName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The packet has no further dependency coordinate for quotient regular-name equality, host real equality, selected limits, or ambient order completeness.

Proof. Read the carrier and classifier of Definition 407.1 and Definition 407.2. They expose only the three regular-name rows, the shared dyadic tail row, both sandwich ledgers, the real seal, transport data, continuation routes, package provenance, and local naming row.

The naming obligations of Definition 407.3 identify those same rows as the local $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$ surface. Sandwich exactness in Definition 407.4 and dyadic-tail exhaustion in Definition 407.7 consume the shared tolerance row before any ledger or seal read is exported.

Transport stability in Definition 407.6 preserves precisely the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates. The completion consumer package of Definition 407.9 then repacks the rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ without adding a coordinate outside the listed dependencies. □

Theorem 407.11 (SqueezeReal shared-tail real-seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet factors through the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *row together with both sandwich ledgers. In particular, the lower and upper ledger reads are forced to consume the same tail request before the seal row* S *can be read by a* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer.*

Proof. Start from the real-facing boundary of Definition 407.5. It states that a seal-facing consumer can see only the regular-name rows, the shared tolerance row, both ledger rows, the displayed transport and continuation data, provenance, and the local naming row.

Apply sandwich exactness from Definition 407.4. The lower and upper ledger reads are tied to the same middle regular-name row and to the same width-bound route, so neither side can supply an independent seal coordinate.

Use dyadic-tail exhaustion from Definition 407.7 and shared tail ledger determinacy from Definition 407.8. These rows force every precision read to pass through ϵ before reading L, U, and S, and transport stability preserves that factorization under the displayed classifier route. □

Theorem 407.12 (SqueezeReal completion handoff scope)**.** *The scoped completion-facing handoff for* $\mathsf{SqueezeReal}^{\uparrow}$ *is exactly the packet assembled from the scoped dependency packet of Definition 407.10, the shared-tail seal factorization of Definition 407.11, and the completion consumer package of Definition 407.9. It exports the finite dyadic sandwich handoff to* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and completion certificates, while excluding quotient regular names, host real equality, selected limits, and ambient order completeness.*

Proof. First assemble the dependency surface. The carrier, classifier, local naming obligations, transport stability, and dyadic-tail exhaustion rows give the finite BHist packet over $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Next attach the seal factorization. The shared tolerance row and both sandwich ledgers determine every $\mathsf{Real}^{\uparrow}$ seal read, so the completion-facing consumer cannot replace the finite ledger by a host equality or a selected limit.

Finally use Definition 407.9. It repacks the dependency and seal rows as the completion-facing handoff, and the boundary rows already rule out every coordinate not displayed in the accepted $\mathsf{SqueezeReal}^{\uparrow}$ carrier. □

Theorem 407.13 (SqueezeReal scoped tail ledger package)**.** *The carrier and classifier of Definition 407.1 and Definition 407.2, the sandwich exactness of Definition 407.4, the dyadic-tail exhaustion of Definition 407.7, the shared-tail determinacy of Definition 407.8, the completion consumer package of Definition 407.9, the scoped dependency packet of Definition 407.10, the shared-tail real-seal factorization of Definition 407.11, and the completion handoff of Definition 407.12 assemble one scoped finite ledger over*

$$\mathsf{DyadicTailBound}^{\uparrow},\quad \mathsf{RegularCauchyName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing read of the packet is exhausted by this ledger and cannot introduce quotient regular-name equality, host real equality, selected limits, or ambient order completeness.*

Proof. First collect the local packet rows. The carrier and classifier identify the finite sandwich tuple, and sandwich exactness forces every seal-facing read through the shared tolerance row and the two displayed ledgers.

Next bind the tail ledger. Dyadic-tail exhaustion and shared-tail determinacy show that all precision reads use the same $\mathsf{DyadicTailBound}^{\uparrow}$ request before consuming L, U, and S.

Finally assemble the completion-facing scope. The completion consumer package, scoped dependency packet, shared-tail real-seal factorization, and completion handoff repack the same rows over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since those rows are the entire displayed packet, no unlisted completion or host-real coordinate remains. □

Theorem 407.14 (SqueezeReal public interval seal export)**.** *An accepted* $\mathsf{SqueezeReal}^{\uparrow}$ *packet whose lower, middle, and upper rows satisfy the* $\mathsf{RegularCauchyName}^{\uparrow}$ *carrier clauses, whose tolerance is the shared* $\mathsf{DyadicTailBound}^{\uparrow}$ *row, and whose sandwich exactness,* $\mathsf{Real}^{\uparrow}$ *boundary, transport stability, and scoped tail ledger package are displayed, exports a public* $\mathsf{Real}^{\uparrow}$ *interval-seal certificate for the middle row. The certificate is the finite package over the lower and upper ledgers, the middle regular name, the shared tail request,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$*; it does not assert host real equality, select a limit, or appeal to ambient order completeness.*

Proof. Start from the three $\mathsf{RegularCauchyName}^{\uparrow}$ rows and the shared $\mathsf{DyadicTailBound}^{\uparrow}$ request. Sandwich exactness ties the middle row to the lower and upper ledgers, while dyadic-tail exhaustion makes the shared request the common precision source for both sides.

Apply the $\mathsf{Real}^{\uparrow}$ boundary and transport stability rows. They keep the interval read on the displayed sandwich ledgers, componentwise hsame transports, continuation routes, package provenance, and local naming row.

Before exposing the public interval seal, read the finite sandwich carrier through chapter 394: its endpoint-cell refinement and interval-containment route supply the located interval ledger consumed by the lower and upper rows.

The shared-tail real-seal factorization then places the middle seal after the shared tolerance and both ledger reads. Repack this route with the scoped tail ledger package to obtain the public interval-seal certificate for the middle row, with no host real equality or selected limit available as an additional coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SqueezeReal}^{\uparrow}$ packet is generated from lower, middle, and upper $\mathsf{RegularCauchyName}^{\uparrow}$ rows, a shared $\mathsf{DyadicTailBound}^{\uparrow}$ tolerance row, two finite sandwich ledgers, componentwise hsame transports, displayed Cont routes, Pkg provenance, a $\mathsf{Real}^{\uparrow}$ seal row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 407.1, Definition 407.2, Definition 407.3, Definition 407.4, Definition 407.5, Definition 407.6, Definition 407.7, Definition 407.8, Definition 407.9, Definition 407.10, Definition 407.11, Definition 407.12, Definition 407.13, and Definition 407.14 bind the finite shared-tail ledger and export the public interval-seal certificate route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SqueezeReal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SqueezeRealUp.SqueezeRealCarrier\_sandwich\_exactness`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim quotient equality of regular names, host real equality, a selected limit, ambient order completeness, or a standard bridge.

Upgrade path: Bridge closure requires a standard-facing route for the public interval-seal certificate.

Chapter 408

A Concrete Naming Certificate for $\mathsf{FoldDefectStokes}^{\uparrow}$

The $\mathsf{FoldDefectStokes}^{\uparrow}$ interface is a BEDC-native finite certificate surface.

The $\mathsf{FoldDefectStokes}^{\uparrow}$ interface names a finite fold-defect ledger whose boundary row accounts for the visible difference between a fold input and its projected fold output. It consumes the fold-source, visible fiber-ledger, and moment-index rows of the $\mathsf{FoldMomentKernel}^{\uparrow}$ carrier in chapter 384, and repackages their endpoint mismatch as its boundary-defect row B and interior ledger L. It is a BHist packet, not an import of external algebra, a chain complex, a topological Stokes theorem, or an ambient quotient calculus. The packet is meant to sit between finite fold consumers such as $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, and $\mathsf{FoldMomentKernel}^{\uparrow}$ while exporting only displayed rows, routes, provenance, and local $\mathsf{NameCert}$ evidence.

Definition 408.1 (FoldDefectStokes carrier)**.** *A* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, O, B, L, H, C, P, N)$$

with the following visible rows:

1. *I is the fold-input row naming the displayed finite source segment and its local fold order;*
2. *O is the projected fold-output row read from I through the displayed fold projection;*
3. *B is the boundary-defect row recording the visible mismatch left at the endpoints of the projection;*
4. *L is the interior ledger whose entries pair each internal fold step with the local defect contribution that cancels inside the segment;*
5. *H records componentwise* hsame *transports for the input, output, boundary-defect, and interior-ledger rows;*
6. *C records the* Cont *routes by which a consumer reads I, then O, then the boundary row B, and finally the interior ledger L;*
7. *P is the* Pkg *provenance naming the finite BEDC packets that own the fold source, projection rule, boundary read, and ledger read;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FoldDefectStokes}^{\uparrow}}$ *row.*

The carrier contains no infinite cochain object, no ambient differential operator, no quotient by invisible cancellations, no host summation law, and no external object.

Theorem 408.2 (FoldDefectStokes NameCert obligations)**.** *Every accepted* $\mathsf{FoldDefectStokes}^{\uparrow}$ *packet with carrier Definition 408.1 carries five local naming-certificate obligations:*

1. *carrier habitation by the empty fold input, empty projected output, empty boundary-defect row, and empty interior ledger;*
2. *classifier laws inherited componentwise from* hsame *on the displayed rows I, O, B, L, H, C, P, and N;*
3. *projection stability under the displayed* Cont *route from fold input I to projected output O;*
4. *boundary-ledger exactness: every internal defect contribution listed in L is paired with a displayed cancellation route, and the unpaired residue is exactly the boundary row B;*
5. *non-escape: every public read of the packet factors through I, O, B, L, H, C, P, and N before it can be consumed by another concrete instance.*

Lean4 ✓ `BEDC.Derived.FoldDefectStokesUp.FoldDefectStokesPacket namecert obligation surface`

Proof. For habitation, use the empty BHist row for I, O, B, and L. The identity hsame rows, trivial Cont routes, empty provenance, and local certificate row repack this as the empty fold-defect carrier.

The classifier laws are componentwise. Reflexivity, symmetry, and transitivity never inspect hidden fold semantics; they compare only the displayed input, output, boundary, ledger, transport, continuation, provenance, and certificate rows.

Projection stability is the stored route from I to O. Transporting the input row through hsame also transports the projected-output row through the corresponding component of H.

Boundary-ledger exactness is read from L and B. The ledger lists the internal defect contributions together with their displayed cancellation partners, while the boundary row records exactly the remaining endpoint contribution.

The non-escape obligation follows because those rows exhaust the carrier coordinates. A downstream consumer can reach no chain complex, host summation, quotient cancellation, or external calculus datum outside the displayed packet. □

Definition 408.3 (FoldDefectStokes empty carrier)**.** *The empty* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier is the packet of Definition 408.1 with empty fold-input row, empty projected-output row, empty boundary-defect row, empty interior ledger, identity componentwise* hsame *transports, trivial displayed* Cont *routes, empty* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FoldDefectStokes}^{\uparrow}}$ *row.*

Theorem 408.4 (FoldDefectStokes classifier laws)**.** *For every accepted* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier $S = (I, O, B, L, H, C, P, N)$, the local classifier transported by H is reflexive, symmetric, and transitive on the displayed input, output, boundary, ledger, route, provenance, and naming rows. These laws are componentwise* hsame *laws over the carrier of Definition 408.1 and add no ambient fold equality.*

Proof. Use the coordinates of Definition 408.1. Reflexivity is obtained by taking the identity hsame row on each of I, O, B, L, H, C, P, and N.

Symmetry reverses each displayed transport row. Since the classifier never reads an unrecorded fold operation, reversing the componentwise rows gives the same accepted surface.

Transitivity composes the displayed transports component by component. The boundary-ledger factorization of Definition 408.7 and the finite no-escape boundary of Definition 408.8 ensure that the composed classifier still mentions only the rows of the carrier. □

Theorem 408.5 (FoldDefectStokes projection stability)**.** *Let $S = (I, O, B, L, H, C, P, N)$ be an accepted* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier. Any accepted transport of the fold-input row I through the displayed classifier keeps the projected-output row O readable through the same I-to-O* Cont *route, with the transported boundary row B and ledger row L still factoring as in Definition 408.7.*

Lean4 ✓ `BEDC.Derived.FoldDefectStokesUp.FoldDefectStokesPacket projection stability`

Proof. Start from the I-to-O route stored in C. Transporting I by the displayed classifier also transports the corresponding output coordinate through the matching component of H.

The boundary and ledger coordinates are then read through the same continuation route. Apply Definition 408.7 to identify the transported public defect with the boundary row and the interior ledger.

Finally, Definition 408.8 rules out any alternate output route through an external fold semantics, quotient, or host summation law. □

Theorem 408.6 (FoldDefectStokes exactness obligation)**.** *For every accepted* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier $S = (I, O, B, L, H, C, P, N)$, the ledger L is exact at the public boundary row B: each internal defect contribution recorded in L is paired with a displayed cancellation route, and the remaining public defect is precisely the boundary read B certified by Definition 408.7.*

Proof. Inspect the ledger coordinate in Definition 408.1. Its entries are internal defect rows together with the displayed cancellation data that removes them from the endpoint read.

Apply Definition 408.7. The theorem says every public defect factors through B and L, so the only contributions not canceled inside L are exactly the entries represented by the boundary row.

The non-escape boundary of Definition 408.8 keeps the obligation local: there is no hidden differential, quotient cancellation, or host telescoping sum that could change the boundary after the ledger is read. □

Theorem 408.7 (FoldDefectStokes boundary-ledger factorization)**.** *Let $S = (I, O, B, L, H, C, P, N)$ be an accepted* $\mathsf{FoldDefectStokes}^{\uparrow}$ *carrier. Any public defect read from the projected fold output factors through the displayed route*

$$I \xrightarrow{C} O \xrightarrow{C} B \xleftarrow{C} L$$

together with the componentwise transport row H, package provenance P, and local certificate row N. In particular, an interior contribution affects the public boundary only by appearing in L with its displayed cancellation partner or by being recorded as part of the boundary row B.

Lean4 ✓ `BEDC.Derived.FoldDefectStokesUp.FoldDefectStokesPacket boundary ledger`

Proof. Start with a public request for a fold-defect read. The carrier first exposes the fold input row I and the projected fold-output row O through the stored continuation route.

The same route then reaches the boundary row B. No consumer can read a defect value before passing through the boundary row, because the carrier has no separate defect coordinate.

It remains to account for the interior ledger. Each internal contribution is listed in L together with the displayed cancellation route that removes it from the endpoint read. Contributions without such an internal cancellation are precisely the entries represented by B.

Transport row H, provenance P, and certificate row N make this factorization stable under accepted row movement. Thus the public defect read is a boundary-ledger read over the finite carrier and not an appeal to host algebra. □

Theorem 408.8 (FoldDefectStokes non-escape boundary)**.** *The* $\mathsf{FoldDefectStokes}^{\uparrow}$ *public boundary exports only the fold-input row, projected fold-output row, boundary-defect row, interior cancellation ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence. It exports no external import, no infinite complex, no ambient differential, no quotient of defects, no host telescoping sum, and no standard Stokes bridge.*

Lean4 ✓ `BEDC.Derived.FoldDefectStokesUp.FoldDefectStokesPacket non escape boundary`

Proof. Inspect the carrier coordinates of Definition 408.1. The only exported rows are I, O, B, L, H, C, P, and N.

By Definition 408.7, every public defect read factors through the boundary and ledger rows. The factorization has no coordinate for an external differential, quotient, infinite object, or host summation operation.

Therefore the public boundary can feed finite BEDC consumers that already understand fold inputs, boundary rows, and ledgers, but it cannot export any stronger calculus or bridge statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{FoldDefectStokes}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FoldDefectStokes}^{\uparrow}$ packet is generated from a finite fold-input row, projected fold-output row, boundary-defect row, interior cancellation ledger, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FoldDefectStokes}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope registers the finite carrier, five NameCert obligations, boundary-ledger factorization, and no-escape boundary for fold-defect packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FoldDefectStokes}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close external imports, infinite complexes, ambient differentials, quotient defect algebra, host telescoping sums, standard Stokes bridges, or bridge checking.

Upgrade path: Obligation closure requires Lean targets for carrier habitation, classifier laws, projection stability, boundary-ledger exactness, and the finite no-escape boundary under `BEDC.Derived.FoldDefectStokesUp`.

Chapter 409

A Concrete Naming Certificate for $\mathsf{CauchyConvolution}^{\uparrow}$

The $\mathsf{CauchyConvolution}^{\uparrow}$ packet records the finite Cauchy-product surface that sits between fast-series convolution ledgers and the regular rational real-name route. Following Bishop–Bridges constructive analysis, it keeps displayed source series, finite convolution windows, dyadic tail budgets, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, $\mathsf{Real}^{\uparrow}$ seal rows, transport data, continuation routes, package provenance, and local naming provenance as finite certificate data. It does not claim host real multiplication, quotient equality of streams, selected infinite sums, or ambient real completeness.

The packet consumes source-series and tail-budget carriers from chapter 367, chapter 368, and chapter 379 before handing the finite product-tail ledger to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

Definition 409.1 (CauchyConvolution carrier)**.** *A* $\mathsf{CauchyConvolution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, G, K, A, D, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *F and G are two accepted* $\mathsf{FastConvergentSeries}^{\uparrow}$ *source rows, including their displayed summand windows, fast tail schedules, regular rational readbacks,* $\mathsf{Real}^{\uparrow}$ *seal-facing rows,* hsame *transports,* Cont *routes, package provenance, and local certificate rows;*
2. *K is the finite convolution ledger whose inspected coefficient at index n is the displayed finite sum $\sum_{0\le i\le n} a_i b_{n-i}$ over the source summand rows;*
3. *A is the finite product-prefix ledger that reads only the displayed convolution coefficients in K;*
4. *D is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *budget ledger, including the source tail budgets, the finite source-window bounds, and the pasted product-tail budget consumed by the convolution row;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from A and D;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to that regular rational handoff;*
7. *H records the componentwise* hsame *transports for source rows, convolution ledgers, dyadic tail budgets, regular handoffs, and seal-facing rows;*
8. *C records the displayed* Cont *routes by which consumers read the source packets, finite convolution ledger, dyadic tail ledger, regular handoff, and real seal;*

9. *P is the* Pkg *provenance over* FastConvergentSeries$^{\uparrow}$, FastSeriesProduct$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and the finite rational arithmetic rows used by the convolution;*
10. *N is the local* NameCert$_{\text{CauchyConvolution}^{\uparrow}}$ *provenance row.*

The carrier is finite Cauchy-product data. It exports no host product of real numbers, no quotient stream, no selected infinite sum, and no real-completeness principle.

Theorem 409.2 (CauchyConvolution NameCert obligations)**.** *Every accepted* CauchyConvolution$^{\uparrow}$ *packet with carrier Definition 409.1 carries five local naming obligations:*

1. *carrier habitation by zero-product data, where both accepted source rows may be the displayed zero fast series, the convolution ledger is the zero finite ledger, the dyadic budget is the zero pasted budget, and the handoff and seal rows are the corresponding regular rational zero rows;*
2. *componentwise classifier laws over the two* FastConvergentSeries$^{\uparrow}$ *source rows, finite convolution ledger, dyadic tail ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row;*
3. *finite convolution ledger exactness, requiring each inspected coefficient row in K to be the finite displayed sum $\sum_{0\le i\le n} a_i b_{n-i}$;*
4. *dyadic tail-bound handoff, requiring the source fast-tail schedules and finite source-window bounds to feed the* DyadicTailBound$^{\uparrow}$ *ledger D and then the* RegSeqRat$^{\uparrow}$ *row R;*
5. Real$^{\uparrow}$ *seal non-escape, requiring every real-facing read to pass through the regular rational handoff and the displayed finite provenance rows.*

These obligations are the complete local certificate surface for the finite Cauchy-product route.

```
Lean4 ✓  BEDC.Derived.CauchyConvolutionUp.CauchyConvolutionPacket semantic name certificate
Lean4 ✓  BEDC.Derived.CauchyConvolutionUp.CauchyConvolutionCarrier namecert obligations
```

Proof. Carrier habitation is witnessed by the displayed zero source rows. Their finite convolution coefficients are zero, the product-prefix ledger is zero, and the dyadic tail budget is the zero budget. Repacking those rows gives the zero regular rational handoff and its real-seal-facing row.

The classifier obligations are componentwise projections of Definition 409.1. A permitted comparison reads only the two source packets, K, A, D, R, E, and the transport, continuation, package, and local naming rows.

The exactness obligation is local to the finite convolution ledger: at each inspected index n, the row records the finite sum over $0 \le i \le n$. The dyadic handoff obligation then reads the source tail schedules and finite source-window bounds into D, so R is reached through displayed tail-budget data rather than a host summation theorem.

Finally, the seal row E is attached only after the regular rational handoff has been displayed. Since the carrier has no coordinate for quotient streams, selected infinite sums, host multiplication, or completeness, every real-facing read remains inside the finite certificate packet. □

Theorem 409.3 (CauchyConvolution carrier admission)**.** *An accepted* CauchyConvolution$^{\uparrow}$ *carrier is admitted as the finite* BHist *packet*

$$Q = (F, G, K, A, D, R, E, H, C, P, N)$$

of Definition 409.1. Admission projects the two FastConvergentSeries$^{\uparrow}$ *source rows F, G, the finite convolution ledger K, the product-prefix ledger A, the* DyadicTailBound$^{\uparrow}$ *budget ledger D, the* RegSeqRat$^{\uparrow}$ *handoff row R, the* Real$^{\uparrow}$ *seal row E, and the transport, continuation, package, and local naming rows H, C, P, N.*

Proof. Project the accepted carrier. The coordinates $F, G, K, A, D, R, E, H, C, P, N$ are all rows of one finite packet, so the local $\mathsf{NameCert}_{\mathsf{CauchyConvolution}^{\uparrow}}$ carrier is inhabited by those projections.

The construction does not add a host product, a quotient stream, a selected infinite sum, or a completeness witness. Those objects are absent from the carrier, so admission remains a finite Cauchy-product packet over the displayed BEDC rows. □

Theorem 409.4 (CauchyConvolution tail-budget compatibility)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *carrier* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, the source fast-tail schedules and finite source-window bounds in* F *and* G*, together with the product-prefix rows in* A *over the inspected coefficients of* K*, determine exactly the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger* D *consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*.*

Lean4 ✓ `BEDC.Derived.CauchyConvolutionUp.CauchyConvolutionTailBudgetCompatibility`

Proof. Read the two source packets F and G. Their fast-tail schedules and finite source-window bounds are part of the displayed source rows, so they can be transported only through the componentwise rows in H and consumed only through the routes in C.

Next read the finite convolution ledger K and the product-prefix ledger A. The prefix rows inspect only finitely many Cauchy-product coefficients, and D records the pasted dyadic tail budget built from those finite inspections and the two source tail schedules.

The handoff row R consumes that same ledger D. Hence compatibility is a row-projection statement about $F, G, K, A, D, R, H, C, P, N$, with no appeal to host real multiplication or equality of quotient streams. □

Theorem 409.5 (CauchyConvolution source-tail seal chain)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, each real-facing seal read factors through the two* $\mathsf{FastConvergentSeries}^{\uparrow}$ *source tail schedules, the finite convolution ledger* K*, the product-prefix ledger* A*, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before reaching the* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

Lean4 ✓ `BEDC.Derived.CauchyConvolutionUp.CauchyConvolutionCarrierSurface source tail seal factorization`

Proof. Project the two source packets F and G. Their fast-tail schedules and finite source-window bounds are visible source rows of the carrier and are transported only by the componentwise rows in H.

Apply Definition 409.4. This binds the source tail schedules to the convolution ledger K, product-prefix ledger A, dyadic tail ledger D, and regular handoff row R along the displayed continuation routes.

Use Definition 409.7 to identify the product-tail comparison consumed by R. The comparison is built from inspected convolution coefficients, source-tail schedules, finite source-window bounds, and the dyadic tail budget.

Only after this chain is displayed can the seal row E be read. Since the carrier has no coordinate for host real multiplication, quotient streams, selected infinite sums, or completeness, the real-facing route cannot bypass the finite source-tail chain. □

Theorem 409.6 (CauchyConvolution Real seal factorization boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *carrier factors through the regular handoff row* R *and the finite Cauchy-product provenance consisting of* $F, G, K, A, D, H, C, P, N$*. The seal boundary does not expose host real multiplication, quotient stream equality, a selected infinite convolution sum, or ambient completeness.*

Proof. Begin with the seal row E. By the carrier shape, E is attached to the regular handoff R and has no independent route to an external real-valued multiplication operation.

Use Definition 409.4 to identify the evidence consumed by R. It is the finite product-prefix ledger A over K together with the dyadic budget D assembled from the displayed source rows F and G.

The remaining rows H, C, P, N preserve transport, consumer access, package provenance, and local naming. Therefore the real-facing seal factors through the finite Cauchy-product packet and cannot export any object outside that packet. □

Theorem 409.7 (CauchyConvolution finite tail ledger)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *carrier*

$$Q = (F, G, K, A, D, R, E, H, C, P, N),$$

the finite convolution ledger K, product-prefix ledger A, and $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger D determine the displayed product-tail comparison consumed by R. The comparison uses only source fast-tail schedules, finite source-window bounds, inspected convolution coefficients, componentwise* hsame *transports, and displayed* Cont *routes.*

Proof. Project the two $\mathsf{FastConvergentSeries}^{\uparrow}$ source rows from F and G. Their fast tail schedules and finite source-window bounds are already displayed rows of the carrier.

The convolution ledger K supplies only finite coefficients, and the product-prefix ledger A reads those coefficients over finite inspected windows. The dyadic ledger D pastes the source tail schedules with the finite source-window bounds to obtain the product-tail budget at the requested precision.

Transport along H and read along C preserve exactly these displayed rows. Thus the comparison consumed by R is a finite tail ledger over $\mathsf{DyadicTailBound}^{\uparrow}$, not an appeal to a host Cauchy product or an infinite convolution object. □

Theorem 409.8 (CauchyConvolution Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, whose regularity evidence is supplied by the finite product-prefix ledger A and the* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger D over the finite convolution row K. The seal route is therefore confined to* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{FastSeriesProduct}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Start from the seal coordinate E of the accepted carrier. By the carrier definition, E is attached to the same regular rational handoff row R and has no independent route to a host real object.

Apply Definition 409.7. It identifies the regularity evidence for R with the finite product-prefix ledger A and dyadic tail budget D, both built over the inspected convolution coefficients in K and the displayed source rows F and G.

The remaining coordinates H, C, P, and N record transport, continuation, package provenance, and local naming. They preserve the displayed route but do not add quotient streams, selected sums, host real multiplication, or completeness. Hence every exported seal read factors through the finite Cauchy-convolution packet alone. □

Definition 409.9 (CauchyConvolution obligation surface)**.** *For an accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet*

$$Q = (F, G, K, A, D, R, E, H, C, P, N),$$

the obligation surface consists of source-series admission, finite convolution coefficient ledger, dyadic tail-budget compatibility, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal non-escape. The surface is confined to* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{FastSeriesProduct}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Theorem 409.10 (CauchyConvolution obligation ledger exactness)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, the source-series, finite coefficient, and dyadic tail-budget obligations of Definition 409.9 are witnessed by the concrete rows* $F, G, K, A, D, H, C, P, N$*.*

Proof. Apply Definition 409.3. It admits the source series rows, finite convolution row, product-prefix ledger, dyadic tail-budget row, and provenance rows as one finite carrier.

Apply Definition 409.4. This binds the source fast-tail schedules and finite source-window bounds to the dyadic tail budget consumed by the regular handoff.

Finally apply Definition 409.7. The convolution coefficient ledger remains finite and tied to K, A, D before the handoff row R is read. □

Theorem 409.11 (CauchyConvolution obligation real handoff package)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal non-escape obligations of Definition 409.9 are exactly the displayed route through* R, E, H, C, P, N *together with the finite convolution ledger* F, G, K, A, D*.*

Proof. Use Definition 409.6. It factors the real-facing route through the regular rational handoff row and the finite convolution ledger.

Then use Definition 409.8. It identifies the exported seal as a read of R supported by A and D over K, with source rows F and G retained as finite provenance.

The transport, continuation, package, and naming rows preserve that route and do not introduce host real multiplication, quotient equality, selected infinite summation, or completeness. □

Theorem 409.12 (CauchyConvolution carrier classifier admission)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, the tuple* Q *supplies carrier admission for the source fast-series rows* F, G*, the finite convolution row* K*, the product-prefix ledger* A*, the dyadic tail budget* D*, the regular rational handoff* R*, the real seal row* E*, and the transport, continuation, package, and naming rows* H, C, P, N*. Its componentwise classifier transport is confined to* $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Apply Definition 409.3. It displays the accepted carrier coordinates and binds $F, G, K, A, D, R, E, H, C, P, N$ into one finite packet.

The classifier part is componentwise. Source classifiers are read from the two $\mathsf{FastConvergentSeries}^{\uparrow}$ rows, tail-budget classifiers from the $\mathsf{DyadicTailBound}^{\uparrow}$ row, handoff classifiers from R, and seal classifiers from E.

The history, sameness, continuation, package, and naming coordinates H, C, P, N transport these readings without adding any object outside the displayed rows. Hence the carrier and the classifier route are admitted by the same finite tuple. □

Theorem 409.13 (CauchyConvolution tail handoff obligation)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, the finite convolution ledger* K*, the product-prefix ledger* A*, and the dyadic budget* D *are exactly the tail handoff consumed by the regular rational row* R*.*

Proof. Use Definition 409.4 to bind the source fast-tail data to the dyadic budget row D.

Next apply Definition 409.7. It identifies the finite coefficient window K and the product-prefix ledger A as the concrete finite evidence that feeds the tail budget.

The handoff row R consumes this data as regular rational evidence. No additional infinite sum, quotient stream, host real product, or completeness principle is part of the handoff route. □

Theorem 409.14 (CauchyConvolution real seal non-escape obligation)**.** *For every accepted* $\mathsf{CauchyConvolution}^{\uparrow}$ *packet* $Q = (F, G, K, A, D, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read factors through* R, E, H, C, P, N *together with the finite data* F, G, K, A, D*. Such a read cannot expose host multiplication, quotient streams, selected sums, or completeness.*

Proof. Apply Definition 409.6. It routes the real-facing coordinate through the regular rational handoff and the finite convolution ledger.

Then apply Definition 409.8. The exported seal is a read of E over R, and R is supported by A and D over K with source provenance in F and G.

The remaining coordinates H, C, P, N only preserve history, continuation, package provenance, and local naming. Therefore the seal read has no route to host multiplication, quotient streams, selected sums, or completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyConvolution}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyConvolution}^{\uparrow}$ packet is generated from two accepted $\mathsf{FastConvergentSeries}^{\uparrow}$ source rows, a finite convolution ledger, finite product-prefix rows, dyadic tail-budget rows, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise transport, continuation readback, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyConvolution}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The scoped surface binds the carrier, convolution ledger, dyadic tail handoff, regular rational handoff, real seal non-escape, BHist, hsame, Cont, Pkg, and NameCert rows for the current Cauchy-product packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyConvolution}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyConvolutionUp.CauchyConvolutionCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host real multiplication, quotient equality of streams, selected infinite sums, uniqueness of products, real completeness, or bridge checking.

Upgrade path: Public closure requires an exported public Cauchy-product interface whose consumer rows are stable beyond the scoped Real and RegSeqRat dependency chain.

Chapter 410

A Concrete Naming Certificate for $\mathsf{LipschitzMap}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, a metric Lipschitz map is useful only when its bound is carried as explicit data. The $\mathsf{LipschitzMap}^{\uparrow}$ interface records that data as a finite BEDC packet: source and target $\mathsf{Met}^{\uparrow}$ rows, a $\mathsf{Real}^{\uparrow}$-valued rational Lipschitz-bound ledger, a $\mathsf{ContinuousMap}^{\uparrow}$ graph row, a $\mathsf{UniformModulus}^{\uparrow}$ row induced by the bound, componentwise hsame transports, visible Cont routes, Pkg provenance, and a local $\mathsf{NameCert}$ row. The packet is a connector for $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ consumers; it is not a Banach fixed-point theorem, a completeness claim, differentiability data, or host function extensionality.

The sibling dependency route is explicit. A $\mathsf{LipschitzMap}^{\uparrow}$ packet consumes the source and target metric rows from chapter 57, the graph row from chapter 130, and the induced modulus row from chapter 369. Its bound ledger is read through the $\mathsf{Real}^{\uparrow}$ surface of chapter 38. The fixed-point consumers in chapter 395 and chapter 396 read the same bound ledger as their contraction witness surface, so the Lipschitz packet is the bound-carrying connector between metric, continuity, modulus, real-bound, and fixed-point certificates.

Definition 410.1 (LipschitzMap carrier)**.** *A* $\mathsf{LipschitzMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, B, G, M, H, C, P, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row;*
3. *B is a rational Lipschitz-bound ledger, read as a* $\mathsf{Real}^{\uparrow}$*-valued nonnegative bound row for target distances;*
4. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the carried map;*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *modulus row induced by the displayed bound ledger;*
6. *H records componentwise* hsame *transports for source points, target points, distance rows, the bound ledger, graph rows, and modulus rows;*
7. *C records the* Cont *routes by which consumers read source distances, target distances, the Lipschitz-bound comparison, the graph row, and the uniform-modulus row;*

8. *P records* Pkg *provenance for* $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}$ *provenance row.*

The carrier accepts exactly those packets whose displayed graph reads and metric-distance rows are bounded by the ledger B and whose modulus row M is the finite uniform-modulus route induced by that same bound.

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier semantic carrier package`

Definition 410.2 (LipschitzMap classifier)**.** *For accepted* $\mathsf{LipschitzMap}^{\uparrow}$ *carriers*

$$L = (X, Y, B, G, M, H, C, P, N) \quad \textit{and} \quad L' = (X', Y', B', G', M', H', C', P', N'),$$

the classifier compares source and target metric-space rows, rational bound-ledger rows, graph rows, induced uniform-modulus rows, transport rows, continuation routes, package provenance, and local certificate rows componentwise through the displayed hsame, Cont, Pkg, *and* NameCert *surface.*

It does not compare host functions extensionally, form a quotient of maps, choose representatives, assert completeness of either metric space, or infer differentiability.

Lean4 (definition) `BEDC.Derived.LipschitzMapUp.LipschitzMapClassifier`

Theorem 410.3 (LipschitzMap NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}$ *seed surface consists of five finite obligations:*

1. *carrier: the packet of Definition 410.1;*

2. *classifier: the componentwise comparison of Definition 410.2;*

3. *stability:* hsame *and* Cont *transport preserve the source and target metric rows, bound ledger, graph row, and induced modulus row;*

4. *exactness: every target-distance read factors through the source-distance read, the displayed rational bound ledger, and the graph row;*

5. *ledger policy:* Pkg *and* NameCert *expose only the listed metric, real-bound, continuous-map, uniform-modulus, transport, and continuation rows.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier namecert obligation certificate`

Proof. Project the carrier. The source and target metric rows, rational bound ledger, graph row, induced modulus row, transport rows, continuation routes, package provenance, and local certificate row are exactly the visible coordinates.

The classifier is componentwise by definition, so stability is obtained by moving each displayed coordinate through the stored hsame rows and then following the stored Cont routes. This preserves the order from source distance to graph read to target distance and bound comparison.

Exactness follows because the target-distance row has no source except the graph row applied to source points and the bound ledger read against the source-distance row. The package and local certificate rows list no host function equality, completeness principle, differentiability datum, or chosen representative, so the ledger policy is exhausted by the displayed packet. □

Theorem 410.4 (LipschitzMap composition boundary)**.** *Let L and L' be accepted* $\mathsf{LipschitzMap}^{\uparrow}$ *carriers whose target* $\mathsf{Met}^{\uparrow}$ *row of L is classifier-compatible with the source* $\mathsf{Met}^{\uparrow}$ *row of L'. The composite consumer route is bounded by the displayed product of the two rational Lipschitz*

ledgers, reads the composite graph through the two ContinuousMap$^{\uparrow}$ *rows, and induces the composite* UniformModulus$^{\uparrow}$ *row through the two visible modulus routes.*

The boundary is a finite consumer boundary: it does not assert completeness, fixed points, differentiability, or host-level equality of composite functions.

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier composition boundary`

Proof. Read the first carrier through its source metric row, graph row, target metric row, and rational bound ledger. The target row is classifier-compatible with the source row of the second carrier, so the intermediate metric observation is a visible carried row rather than a host equality.

Read the second carrier through its graph row and bound ledger. The target distance of the second read is bounded by the second ledger applied to the intermediate distance, while the intermediate distance is bounded by the first ledger applied to the original source distance.

Repack the two bound reads as the displayed product ledger and route the graph reads through the two ContinuousMap$^{\uparrow}$ rows. The induced modulus route is the sequential use of the two UniformModulus$^{\uparrow}$ rows, and the exclusions follow because neither carrier contains completeness, fixed-point, differentiability, or host function-extensionality coordinates. □

Theorem 410.5 (LipschitzMap ledger nonescape)**.** *Every* UniformModulus$^{\uparrow}$, PicardContraction$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, *or* ContinuousMap$^{\uparrow}$ *consumer of an accepted* LipschitzMap$^{\uparrow}$ *carrier factors through the displayed source and target* Met$^{\uparrow}$ *rows, rational* Real$^{\uparrow}$ *bound ledger,* ContinuousMap$^{\uparrow}$ *graph row,* UniformModulus$^{\uparrow}$ *row, componentwise* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

No consumer can obtain a Banach fixed-point conclusion, completeness witness, differentiability row, quotient of maps, selected representative, or host function extensionality from the Lipschitz ledger alone.

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier ledger nonescape`

Proof. Use the obligation surface of Definition 410.3. It lists the carrier, classifier, stability, exactness, and ledger rows available to every local certificate consumer.

For a UniformModulus$^{\uparrow}$ consumer, read the modulus row M induced by the bound ledger B. For ContinuousMap$^{\uparrow}$ consumers, read the graph row G and the same transport routes. For PicardContraction$^{\uparrow}$ and BanachFixedPoint$^{\uparrow}$ consumers, the only available contraction-relevant datum is the rational bound ledger together with the metric rows and graph route.

Thus every consumer route stays inside the displayed finite packet. The excluded conclusions require coordinates not present in Definition 410.1, so they cannot escape from the local ledger. □

Theorem 410.6 (LipschitzMap complete-metric regular seal boundary)**.** *For an accepted* LipschitzMap$^{\uparrow}$ *packet*

$$L = (X, Y, B, G, M, H, C, P, N),$$

any CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer of the Lipschitz graph and bound must factor through the displayed source* Met$^{\uparrow}$ *row* X, *target* Met$^{\uparrow}$ *row* Y, *rational* Real$^{\uparrow}$ *bound ledger* B, ContinuousMap$^{\uparrow}$ *graph row* G, *induced* UniformModulus$^{\uparrow}$ *row* M, *componentwise* hsame *transports* H, *displayed* Cont *routes* C, Pkg *provenance* P, *and local* NameCert *row* N.

The handoff exports no Banach fixed point, completed orbit, host function equality, quotient map, selected limit, or ambient completeness row. The downstream seal surfaces are the complete-metric row of chapter 128, the regular-rational source row of chapter 301, and the real-name seal row of chapter 38.

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier completemetric regular seal boundary`

Proof. Start with Definition 410.3. It exposes exactly the carrier rows $X, Y, B, G, M, H, C, P, N$ together with the local classifier, stability, exactness, and ledger obligations.

Apply Definition 410.5. Every listed consumer of the Lipschitz graph and bound remains inside the displayed source metric, target metric, rational bound, graph, modulus, transport, continuation, package, and naming rows. Thus a completion-facing read cannot obtain an ambient completeness coordinate or a selected point from the Lipschitz packet.

Use Definition 410.7 for the modulus route. The induced row M is read from X, G, and B through the same transport and package surface, so the route toward regular-rational and real seal consumers is the carried modulus route rather than a host function equality or quotient map.

The sibling chapters chapter 128, chapter 301, and chapter 38 supply only the downstream seal surfaces. Since none of those surfaces is a coordinate of L, the LipschitzMap packet contributes only the displayed graph-and-bound ledger to their consumer routes. □

Theorem 410.7 (LipschitzMap uniform-modulus consumer handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzMap}^{\uparrow}$ *carrier*

$$L = (X, Y, B, G, M, H, C, P, N)$$

reads the modulus row only through the source $\mathsf{Met}^{\uparrow}$ *distance row* X*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* G*, the rational* $\mathsf{Real}^{\uparrow}$ *bound ledger* B*, displayed* hsame *and* Cont *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row* N*. The handoff exposes the same induced modulus row* M *and does not export host function extensionality, a quotient of maps, a selected representative, or a target-space principle outside the carried Lipschitz packet.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier uniform modulus consumer handoff`

Proof. Start with the obligation surface of Definition 410.3. It binds the carrier, classifier, stability, exactness, and ledger policy for the rows $X, Y, B, G, M, H, C, P, N$.

Apply ledger nonescape from Definition 410.5. A $\mathsf{UniformModulus}^{\uparrow}$ consumer is one of the listed consumers, so its read stays inside the displayed metric rows, graph row, bound ledger, induced modulus row, transport, continuation, provenance, and naming data.

Finally apply bound transport from Definition 410.9. Transport preserves the accepted Lipschitz packet and keeps M tied to the same bound ledger B. The consumer therefore receives the uniform-modulus data only through the carried source distance, graph, rational bound, and visible certificate rows. □

Theorem 410.8 (LipschitzMap public modulus export)**.** *For an accepted* $\mathsf{LipschitzMap}^{\uparrow}$ *packet*

$$L = (X, Y, B, G, M, H, C, P, N),$$

the public $\mathsf{UniformModulus}^{\uparrow}$ *row exported to any consumer is exactly* M*. It is induced from the rational* $\mathsf{Real}^{\uparrow}$ *bound ledger* B *through the graph row* G *and the source* $\mathsf{Met}^{\uparrow}$ *row* X*. No consumer may replace* M *by a host function modulus, a quotient of maps, or a target-space completeness principle.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier public modulus export`

Proof. First read the packet rows from Definition 410.3. The accepted carrier exposes $X, Y, B, G, M, H, C, P, N$ as the complete Lipschitz naming surface.

Next apply bound transport from Definition 410.9. The rational ledger B, graph row G, and induced modulus row M are transported as displayed rows of the same packet, so a consumer cannot detach the modulus from its source metric and graph evidence.

Finally use Definition 410.7. The handoff routes every $\mathsf{UniformModulus}^{\uparrow}$ read through X, G, B, M, H, C, P, N, hence the public row exported to the consumer is the carried row M and no host modulus, map quotient, or completeness principle is available. □

Theorem 410.9 (LipschitzMap bound transport)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^{\uparrow}$ *carrier. Any displayed* hsame *and* Cont *transport of the source* $\mathsf{Met}^{\uparrow}$ *row* X*, target* $\mathsf{Met}^{\uparrow}$ *row* Y*, graph row* G*, rational* $\mathsf{Real}^{\uparrow}$ *bound ledger* B*, and induced* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *preserves the accepted Lipschitz packet. The transported packet has the same finite consumer surface: source-distance reads still enter through* X*, target-distance reads still exit through* Y*, graph reads still pass through* G*, and every modulus read is routed through the row* M *induced by the ledger* B*.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier bound transport`

Proof. Project the carrier coordinates. The transport row H already lists the componentwise hsame moves for source points, target points, metric-distance rows, bound ledger entries, graph rows, and modulus rows.

Follow the stored Cont routes in C after each componentwise move. The source-distance route is transported first, the graph route is then read over the transported source and target rows, and the rational bound comparison is read from the transported ledger B.

The induced modulus row has no separate origin: it is the $\mathsf{UniformModulus}^{\uparrow}$ route generated from the same displayed bound ledger. Repack the transported coordinates with the original Pkg provenance P and local $\mathsf{NameCert}$ row N. No representative choice or host function equality is used. □

Theorem 410.10 (LipschitzMap target-distance exactness)**.** *For an accepted* $\mathsf{LipschitzMap}^{\uparrow}$ *carrier*

$$L = (X, Y, B, G, M, H, C, P, N),$$

every target-distance read factors through the displayed source-distance read in X*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* G*, and the rational* $\mathsf{Real}^{\uparrow}$ *bound ledger* B*. In particular, a consumer cannot read a target metric comparison from a hidden target-space principle, a quotient of maps, or an external function-extensionality assertion; the only exact route is the carried BEDC route*

source distance in X*,*
graph read through G*,*
bound comparison through B*,*
target distance in Y*.*

Proof. Read the target-distance coordinate from the carrier. By the classifier of Definition 410.2, target metric rows are compared only componentwise through the displayed hsame and Cont surface.

The graph coordinate G supplies the only route from source points to target points. The rational ledger B supplies the only nonnegative bound row for the target-distance comparison, and the source metric row X supplies the only source-distance input to that comparison.

Thus every target-distance read is obtained by composing those visible routes. The package row P and local certificate row N mention no completeness principle, quotient equality, representative selector, or host extensionality datum, so there is no second exactness route to project. □

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier target distance exactness`

Theorem 410.11 (LipschitzMap contraction-threshold handoff)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^{\uparrow}$ *carrier. If the rational bound ledger B is read through the displayed* $\mathsf{Real}^{\uparrow}$ *order as strictly below the unit threshold, then every* $\mathsf{PicardContraction}^{\uparrow}$ *or* $\mathsf{BanachFixedPoint}^{\uparrow}$ *consumer may use only the carried source* $\mathsf{Met}^{\uparrow}$ *row X, target* $\mathsf{Met}^{\uparrow}$ *row Y,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row G,* $\mathsf{UniformModulus}^{\uparrow}$ *row M, displayed* hsame *and* Cont *routes,* Pkg *provenance, and local* NameCert *row N.*

The handoff records contraction-threshold data for downstream packets, but it does not assert fixed points, completeness of X or Y, a quotient of maps, host function extensionality, or a selected representative.

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier contraction threshold handoff`

Proof. Start with target-distance exactness from Definition 410.10. It fixes the only target-distance route as the path from the source distance in X through the graph row G, bound ledger B, and target row Y.

Use bound transport from Definition 410.9. The strict subunit read of B is preserved by the displayed hsame and Cont movement, so the threshold datum remains attached to the same finite carrier.

Now apply the uniform-modulus consumer handoff of Definition 410.7. The modulus row M is read only through X, G, B and the local certificate rows, so a contraction consumer receives no independent analytic input.

Finally apply ledger nonescape from Definition 410.5. $\mathsf{PicardContraction}^{\uparrow}$ and $\mathsf{BanachFixedPoint}^{\uparrow}$ are listed consumers, and the packet contains no completeness, fixed-point, quotient, selector, or host-extensionality coordinate. The contraction-threshold handoff is therefore exactly the displayed finite route. □

Theorem 410.12 (LipschitzMap contraction-threshold factorization)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^{\uparrow}$ *carrier. If the bound ledger B is read as strictly below the unit threshold and a downstream fixed-point consumer supplies the required* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, *or* $\mathsf{BanachFixedPoint}^{\uparrow}$ *carrier row, then the consumer route factors through exactly*

$$X, Y, B, G, M, H, C, P, N$$

plus that displayed completeness or contraction row. The route cannot read fixed points, completeness, quotient-map equality, selected representatives, or host function extensionality from the $\mathsf{LipschitzMap}^{\uparrow}$ *packet itself.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier contraction threshold factorization`

Proof. Start with target-distance exactness from Definition 410.10. Every target-distance comparison used by the downstream consumer is routed through the source metric row X, graph row G, bound ledger B, and target metric row Y.

Use bound transport from Definition 410.9 together with the uniform-modulus handoff of Definition 410.7. These results keep the subunit threshold read attached to the displayed carrier and expose the modulus row M only through X, G, B, H, C, P, N.

Apply the contraction-threshold handoff of Definition 410.11. A $\mathsf{PicardContraction}^{\uparrow}$ or $\mathsf{BanachFixedPoint}^{\uparrow}$ consumer may therefore import only the displayed Lipschitz packet together with its own completeness or contraction row.

Finally apply ledger nonescape from Definition 410.5. The $\mathsf{LipschitzMap}^{\uparrow}$ packet has no coordinate for a fixed point, a completeness witness, quotient-map equality, a selected representative, or host function extensionality, so those data must come from the downstream carrier rather than from L. □

Theorem 410.13 (LipschitzMap identity carrier boundary)**.** *Let X be a transported $\mathsf{Met}^{\uparrow}$ row and let*

$$L = (X, X, B_{\mathsf{id}}, G_{\mathsf{id}}, M_{\mathsf{id}}, H, C, P, N)$$

be the packet whose source and target metric rows are the same transported row, whose graph row G_{id} is the identity graph over X, whose rational bound ledger B_{id} is the unit $\mathsf{Real}^{\uparrow}$ row, and whose modulus row M_{id} is the identity $\mathsf{UniformModulus}^{\uparrow}$ route. Then L forms an accepted $\mathsf{LipschitzMap}^{\uparrow}$ consumer boundary over the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it exposes no host function extensionality or selected representative.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier identity carrier boundary`

Proof. Project the carrier shape of Definition 410.1. With the same $\mathsf{Met}^{\uparrow}$ row in the source and target coordinates, the identity graph row supplies the carried $\mathsf{ContinuousMap}^{\uparrow}$ route from each source point history to the same target point history.

Use target-distance exactness from Definition 410.10. For the identity graph, every target-distance read is the same transported source-distance read, so the unit rational ledger B_{id} binds the target comparison to the original source metric row.

The identity $\mathsf{UniformModulus}^{\uparrow}$ route M_{id} is then the modulus induced by that unit bound. Bound transport from Definition 410.9 keeps B_{id}, G_{id}, and M_{id} attached to the same transported row through H and C.

Finally apply ledger nonescape from Definition 410.5. The packet contains only the displayed metric row, identity graph, unit bound ledger, identity modulus route, transports, continuations, provenance, and local certificate, so it cannot export host function extensionality or a selected representative. □

Theorem 410.14 (LipschitzMap obligation closure package)**.** *The $\mathsf{LipschitzMap}^{\uparrow}$ obligation surface is exhausted by the carrier of Definition 410.1, the classifier of Definition 410.2, bound transport Definition 410.9, target-distance exactness Definition 410.10, composition boundary Definition 410.4, and ledger nonescape Definition 410.5. These rows form a finite* BHist *package over* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$*.*

Proof. Collect the carrier and classifier rows first. They name exactly the finite coordinates and the componentwise comparison surface available to a local $\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}$ consumer.

Add the three transport and routing obligations. Bound transport preserves the accepted packet under displayed hsame and Cont movement. Target-distance exactness fixes the unique target-distance read route. Composition boundary fixes the product ledger, sequential modulus route, and local certificate route for composed consumers.

Finally apply ledger nonescape. It states that $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$ consumers factor through the displayed rows and do not acquire hidden completeness, differentiability, quotient, representative-selection, or host extensionality data. Repacking these named clauses gives precisely the obligation closure package. □

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier obligation closure package`

Theorem 410.15 (LipschitzMap bound dependency lock)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^{\uparrow}$ *carrier. Every* $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of* L *must read the rational* $\mathsf{Real}^{\uparrow}$ *bound ledger* B *together with the source and target* $\mathsf{Met}^{\uparrow}$ *rows* X, Y*, the graph row* G*, and the induced modulus row* M*. No such consumer may detach* B *into a fixed-point theorem, completeness theorem, differentiability claim, quotient-map claim, or host-extensionality claim.*

Proof. Start with target-distance exactness. It binds each target-distance read to the source and target metric rows X, Y, the graph row G, and the rational bound ledger B through the displayed carrier route.

Use bound transport to keep B attached to the same graph and modulus data under hsame and Cont movement. The ledger is transported only as part of the accepted packet, not as an independent real-number fact.

Now apply the consumer handoffs. Uniform-modulus consumers read the induced row M from B, X, Y, G, contraction-threshold consumers read the same bound through the displayed threshold route, and composition boundary preserves the bound only through the product-ledger route for compatible Lipschitz packets.

Finally apply ledger nonescape. Since the carrier contains no row for completeness, differentiability, quotient-map equality, selected representatives, or host function extensionality, every listed consumer remains locked to $X, Y, B, G, M, H, C, P, N$ and cannot detach B into one of those stronger claims. □

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier bound dependency lock`

Theorem 410.16 (LipschitzMap common-bound completion route)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^{\uparrow}$ *carrier. Every completion-facing consumer that reads the graph row* G*, the rational Lipschitz-bound ledger* B*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, and a* $\mathsf{Real}^{\uparrow}$ *seal must factor through one shared bound ledger together with the source and target* $\mathsf{Met}^{\uparrow}$ *rows* X, Y*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* G*, and the* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal boundary. This route does not supply ambient completeness, selected limits, quotient-map equality, host function extensionality, or differentiability.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier common bound completion route`

Proof. Start from the local $\mathsf{NameCert}_{\mathsf{LipschitzMap}^\uparrow}$ obligations. They make the accepted packet read exactly the displayed source metric row, target metric row, rational bound ledger, continuous graph row, induced modulus row, transports, continuations, provenance, and local naming certificate.

Apply ledger nonescape and the uniform-modulus handoff. The rational bound B can be used by a consumer only through the displayed Lipschitz packet, so the modulus row M remains tied to the same X, Y, B, G ledger rather than becoming an ambient regularity fact.

Next use the complete-metric regular seal boundary. The completion-facing $\mathsf{RegSeqRat}^\uparrow$ source row and $\mathsf{Real}^\uparrow$ seal are read only after the shared Lipschitz ledger has passed through the sibling $\mathsf{Met}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, and $\mathsf{UniformModulus}^\uparrow$ rows.

Finally apply Definition 410.15. It excludes detaching the common bound into completeness, selected limits, quotient-map equality, host function extensionality, or differentiability, and leaves the displayed $\mathsf{Met}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, $\mathsf{UniformModulus}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$ rows as the only available completion route. □

Theorem 410.17 (LipschitzMap separated-completion nonescape)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^\uparrow$ *carrier. Every completion-facing consumer routed through* $\mathsf{CompleteMetric}^\uparrow$, $\mathsf{SeparatedMetric}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, *and* $\mathsf{UniformModulus}^\uparrow$ *must read the rational Lipschitz-bound ledger B and the separated zero-distance route before any completion identity claim is exposed. The consumer receives only the common-bound completion route, target-distance exactness, contraction-threshold factorization, and the public zero-distance route of* $\mathsf{SeparatedMetric}^\uparrow$*; it receives no ambient completeness theorem, selected limit, quotient completion equality, host function extensionality, differentiability, or detached real equality.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier separated completion nonescape`

Proof. Start from Definition 410.16. It places the graph row G, rational bound ledger B, source and target metric rows X, Y, induced modulus row M, $\mathsf{RegSeqRat}^\uparrow$ source row, and $\mathsf{Real}^\uparrow$ seal on one shared completion-facing route.

Apply target-distance exactness and the contraction threshold factorization. The target-distance read is tied to X, Y, B, G, while contraction-facing consumers may use the bound only through the displayed threshold route and induced modulus row.

Finally use the public separated-metric zero-distance route of Definition 1411.42. A completion identity claim can be exposed only after the separated zero-distance classifier has been read, so the Lipschitz packet cannot bypass B or replace the separated route by quotient completion equality, selected limits, or host equality. □

Theorem 410.18 (LipschitzMap scoped modulus route)**.** *Let*

$$L = (X, Y, B, G, M, H, C, P, N)$$

be an accepted $\mathsf{LipschitzMap}^\uparrow$ *carrier. Before any separated-completion or uniform-continuity consumer reads L, the metric source and target rows X, Y, rational* $\mathsf{Real}^\uparrow$ *distance row B,* $\mathsf{ContinuousMap}^\uparrow$ *graph row G,* $\mathsf{UniformModulus}^\uparrow$ *row M, common bound, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LipschitzMap}^\uparrow}$ *row must pass through one finite surface. The consumer receives the scoped modulus route displayed by the packet, not a detached modulus, ambient completion theorem, quotient equality of maps, host function extensionality, differentiability claim, or selected representative.*

Lean4 ✓ `BEDC.Derived.LipschitzMapUp.LipschitzMapCarrier scoped modulus route`

Proof. Start with the local NameCert obligations. They expose exactly the source and target metric rows, the rational bound ledger, the continuous graph row, the induced modulus row, transport, replay, provenance, and local naming surface of L.

Use the uniform-modulus consumer handoff and public modulus export. A uniform-continuity consumer can read M only as the modulus induced by the same X, Y, B, G route, and the exported modulus remains tied to the displayed Lipschitz bound rather than becoming an ambient modulus fact.

Next apply bound transport, target-distance exactness, and the common-bound completion route. The bound row B is transported only with the graph and modulus data, each target-distance read is tied to X, Y, B, G, and every completion-facing read factors through the shared bound before reaching $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seals.

Finally use Definition 410.17. Separated-completion consumers must read the separated zero-distance route after the same scoped Lipschitz surface, so no consumer can bypass B, M, H, C, P, N or replace the packet by quotient equality, host extensionality, differentiability, selected representatives, or ambient completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzMap}^{\uparrow}$ packet is generated from source and target $\mathsf{Met}^{\uparrow}$ rows, a rational $\mathsf{Real}^{\uparrow}$ Lipschitz-bound ledger, a $\mathsf{ContinuousMap}^{\uparrow}$ graph row, a $\mathsf{UniformModulus}^{\uparrow}$ modulus row induced by the bound, componentwise hsame transports, visible Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 410.1, Definition 410.2, Definition 410.3, Definition 410.9, Definition 410.10, Definition 410.4, Definition 410.14, and Definition 410.18 bind the displayed bound, graph, modulus, transport, continuation, provenance, and local naming surface over $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Banach fixed point, completeness of the metric spaces, differentiability, quotient equality of maps, selected representatives, host function extensionality, or bridge checking.

Upgrade path: Formal verification and standard bridge checking remain separate from the mature paper treatment of the displayed bound, graph, modulus, transport, continuation, provenance, and local naming surface.

Chapter 411

Regular Cauchy Comparison NameCert Construction

The $\mathsf{RegularCauchyComparison}^{\uparrow}$ interface records the finite comparison layer used by the Real completeness proof target of section 9238.2. It compares two regular Cauchy names through shared finite windows, displayed dyadic observations, rational tolerance rows, and the downstream $\mathsf{Real}^{\uparrow}$ seal classifier. The packet does not select a host limit, quotient regular streams, or invoke ambient real completeness.

The common-window comparison consumes the finite window schedule of chapter 126 and the dyadic observation ledger of chapter 291 before the $\mathsf{Real}^{\uparrow}$ seal row is read.

Definition 411.1 (Regular Cauchy comparison carrier)**.** *A* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

with the following visible rows:

1. *X and Y are the two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy names under comparison;*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule used by both names for every inspected precision request;*
3. *D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation surface for the two windows named by W;*
4. *τ is the rational tolerance row against which the dyadic observations are compared;*
5. *L is the finite ledger recording which window reads, dyadic observations, and tolerance rows have been inspected;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-provenance row reached only after the common window and dyadic comparison rows have been displayed;*
7. *H stores componentwise* hsame *transports for the two names, the shared schedule, dyadic observations, tolerance row, finite ledger, and seal row;*
8. *C stores the displayed* Cont *routes by which consumer reads reach those same finite rows;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}$ *row.*

The carrier contains no quotient of regular names, no host real equality test, no selected limit, and no completeness witness outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonTasteGate single carrier alignment`

Definition 411.2 (Regular Cauchy comparison classifier)**.** *For accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packets* K *and* K'*, the classifier* $\mathsf{RegularCauchyComparisonClassifier}(K, K')$ *holds exactly when all displayed finite comparison windows refine to the same transported dyadic tolerance row and the* $\mathsf{Real}^{\uparrow}$ *seal rows classify identically after those refinements.*

Thus the classifier reads only the two $\mathsf{RegSeqRat}^{\uparrow}$ *names, the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule, displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, rational tolerance row, finite ledger,* $\mathsf{Real}^{\uparrow}$ *seal provenance,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not compare host sequences, quotient representatives, selected limits, or unrecorded real carriers.*

Theorem 411.3 (Regular Cauchy comparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission: accepted packets are exactly the finite* BHist *carriers of Definition 411.1;*
2. *classifier stability: Definition 411.2 is preserved by displayed* hsame *transports and* Cont *routes on the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, shared* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic observations, tolerance row, ledger, seal row, package provenance, and local naming row;*
3. *window exactness: every comparison request reads both names through the same displayed finite schedule* W *before inspecting the dyadic observation row* D*;*
4. *tolerance exactness: the dyadic observations are compared only against the rational tolerance row* τ *recorded in the packet ledger;*
5. *seal handoff: the* $\mathsf{Real}^{\uparrow}$ *classifier is consumed only after the common-window comparison and tolerance rows have been displayed.*

This surface is a comparison certificate for the Real completeness capstone and does not assert real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier semantic name certificate`

Theorem 411.4 (Regular Cauchy comparison window exactness)**.** *For every accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

and every displayed comparison request, both $\mathsf{RegSeqRat}^{\uparrow}$ *names are read through the same finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*, the selected observations are exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows in* D*, and the tolerance comparison is exactly the rational row* τ *recorded in* L*. No consumer read may split the two names across unrelated windows or replace* τ *by an unrecorded tolerance.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier window exactness`

Theorem 411.5 (Regular Cauchy comparison Real classifier handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *classifier read exported by an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the rational tolerance row, the finite ledger,* $\mathsf{Real}^{\uparrow}$ *seal provenance,* BHist*,* hsame*,* Cont*,* Pkg*, and*

NameCert *rows. The handoff supplies the finite comparison layer used by Definition 9238.5 without adding quotient stream equality, a selected host limit, host real equality, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier real classifier handoff`

Theorem 411.6 (Regular Cauchy comparison common-window stability)**.** *For every accepted* RegularCauchyComparison$^{\uparrow}$ *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

the shared window row W is stable under the displayed hsame *transports and* Cont *routes. A classified read of either X or Y must inspect the same transported* StreamName$^{\uparrow}$ *schedule before reading D and τ, and no read may split the two regular names across unrelated windows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier common window stability`

Proof. Project the carrier of Definition 411.1. It contains one shared StreamName$^{\uparrow}$ schedule W and no second window coordinate.

Apply Definition 411.2. The classifier moves the schedule only by the stored component of H and by the displayed consumer routes in C.

Now use Definition 411.4. Every comparison request reads both regular names through that common schedule before it reaches dyadic observations or tolerance rows. □

Theorem 411.7 (Regular Cauchy comparison tolerance-row exactness)**.** *For every accepted* RegularCauchyCompari *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

the rational tolerance row τ is the only tolerance inspected by comparison reads. Classified transport may move τ through the displayed hsame *and* Cont *routes, but it cannot substitute an unrecorded rational tolerance or a host-real equality threshold.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier tolerance row exactness`

Proof. Use Definition 411.1 to identify τ as the sole rational tolerance coordinate of the packet.

Apply Definition 411.4. Once the two regular names are read through W, the dyadic observations in D are compared against exactly the tolerance row recorded in L.

The classifier definition of Definition 411.2 permits only componentwise transport of that row. Therefore a consumer read can transport τ but cannot replace it by an absent tolerance. □

Theorem 411.8 (Regular Cauchy comparison Real seal boundary)**.** *Every* Real$^{\uparrow}$ *classifier read exported by an accepted* RegularCauchyComparison$^{\uparrow}$ *packet factors through the displayed rows $X, Y, W, D, \tau, L, E, H, C,$ The seal boundary is the finite comparison packet itself: it exposes no quotient stream equality, selected host limit, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier real seal boundary`

Proof. First apply Definition 411.6. The two regular names reach the comparison only through the shared transported window W.

Next apply Definition 411.7. The dyadic observations in D are compared only against the rational tolerance row τ recorded in the finite ledger L.

Finally use Definition 411.5. Its handoff route is exactly the path through $X, Y, W, D, \tau, L, E, H, C, P, N$, so the real-facing read has no access to quotient representatives, selected limits, host equality, or ambient completeness. □

Theorem 411.9 (Regular Cauchy comparison seal-compatible transitivity)**.** *Let accepted* RegularCauchyComparison *packets compare X to Y and Y to Z. If their transported* $\mathsf{StreamName}^{\uparrow}$ *schedules and rational tolerance rows refine to a common finite window ledger, then the composed comparison packet from X to Z has the same* $\mathsf{Real}^{\uparrow}$ *seal-compatible boundary through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The composed packet does not assert quotient stream equality, selected host limits, host real equality, or an ambient completeness principle.*

Proof. First apply Definition 411.6. The two comparison packets can be read on the common finite window ledger because their transported schedules refine to that ledger.

Next apply Definition 411.7. The composed read uses only the rational tolerance rows displayed by the two packets and their common ledger refinement; it cannot introduce a host-real equality threshold.

Finally use Definition 411.8 together with Definition 411.5. The comparison from X to Z reaches the $\mathsf{Real}^{\uparrow}$ seal through the same finite BHist, hsame, Cont, Pkg, and local NameCert rows, so the transitive handoff remains a seal-compatible finite comparison and not a statement about selected limits or quotient stream equality. □

Theorem 411.10 (Regular Cauchy comparison triangle-window composition)**.** *Let accepted* RegularCauchyComparison *packets compare X to Y and Y to Z. Suppose their* $\mathsf{StreamName}^{\uparrow}$ *schedules refine to one displayed finite window and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows refine to one visible comparison ledger. Then the two packets compose to an accepted X-to-Z comparison whose* $\mathsf{Real}^{\uparrow}$ *seal handoff factors through the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The composed comparison is only this finite triangle-window route. It does not introduce quotient stream equality, a selected host limit, host real equality, or an ambient completeness principle.

Proof. First use Definition 411.6. The two comparison packets are read on the single displayed window because both transported $\mathsf{StreamName}^{\uparrow}$ schedules refine to it.

Next use Definition 411.7. The $\mathsf{DyadicRatCore}^{\uparrow}$ observations and rational tolerances are consumed through the common ledger, so the composed route has no tolerance datum outside the finite packet.

The seal boundary is supplied by Definition 411.8. It keeps the $\mathsf{Real}^{\uparrow}$ handoff on the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 411.9. The X-to-Y and Y-to-Z finite comparisons therefore compose to the X-to-Z finite comparison, with no appeal to quotient streams, selected limits, host real equality, or completeness outside the carrier. □

Theorem 411.11 (Regular Cauchy comparison finite-window induction)**.** *Let $K = (X, Y, W, D, \tau, L, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet. Every accepted comparison route over a finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule is exhausted by induction over the displayed window ledger W.*

Along the induction, the route preserves the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, the tolerance row* τ*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N*.*

Consequently a comparison consumer that accepts all finite windows in W *receives exactly the displayed comparison surface*

$$X, Y, W, D, \tau, L, E, H, C, P, N$$

and cannot add a dyadic observation outside D*, a tolerance coordinate outside* τ*, a second real seal, quotient stream equality, a selected limit, host real equality, or an ambient completeness principle.*

Proof. Induct on the displayed finite window ledger W.

In the empty-window case, there is no window from which a fresh $\mathsf{DyadicRatCore}^{\uparrow}$ observation can be read. The comparison route is therefore restricted to the already displayed rows $D, \tau, L, E, H, C, P, N$, and the carrier supplies no coordinate for quotient streams or selected host limits.

For the successor window, apply common-window stability to keep the new $\mathsf{StreamName}^{\uparrow}$ cell on the same transported schedule. Then use tolerance-row exactness: the successor read may consume only the dyadic observation already recorded in D and the rational tolerance already recorded in τ for that cell.

The induction hypothesis preserves all earlier cells of W with the same structural rows H, C, P, N. Combining the earlier route with the successor read gives the full finite ledger, and the $\mathsf{Real}^{\uparrow}$ seal boundary closes the induction by forcing the final handoff through E rather than through host real equality or ambient completeness. □

Theorem 411.12 (Regular Cauchy comparison diagonal-budget handoff)**.** *Let* $K = (X, Y, W, D, \tau, L, E, H, C, P, N$ *be an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet. Suppose the shared window* W *and tolerance rows* D, τ, L *are consumed by an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *packet and an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. Then* K *exports one diagonal* $\mathsf{Real}^{\uparrow}$ *handoff through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

The exported handoff is only the finite route from comparison windows and dyadic tolerance ledgers into the diagonal budget and tail schedule. It does not expose quotient streams, selected host limits, host real equality, or an ambient completeness principle.

Proof. Start with common-window stability. The accepted comparison packet makes the two $\mathsf{RegSeqRat}^{\uparrow}$ names readable only through the transported $\mathsf{StreamName}^{\uparrow}$ window W, so the diagonal budget and tail schedule consume the same finite window rather than two unrelated stream views.

Next use tolerance-row exactness. The $\mathsf{DyadicRatCore}^{\uparrow}$ observations and the rational tolerance row τ are the only tolerance data recorded in D and L, hence the diagonal-budget consumer receives exactly the displayed tolerance ledger.

The $\mathsf{Real}^{\uparrow}$ seal boundary then fixes the endpoint of the handoff. The comparison packet reaches the real-facing row through E after the regular-rational, stream-window, dyadic, ledger, transport, route, provenance, and naming rows have been read.

Finally combine the finite route obligations already recorded by the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ and $\mathsf{TailCofinalitySchedule}^{\uparrow}$ consumers. Those packets consume the shared window and tolerance ledger before the tail schedule reaches its $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal. Since none of the three packets contains a coordinate for quotient streams, selected host limits, host real equality, or ambient completeness, the composite handoff remains the single displayed finite route. □

Theorem 411.13 (Regular Cauchy comparison classifier transport)**.** *Let*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet. If* K' *is obtained by transporting the displayed coordinates componentwise along the stored* hsame *routes in* H *and then reading them along the displayed* Cont *routes in* C*, then* K' *is accepted by the same classifier. The transport preserves the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the tolerance row* τ*, the finite ledger* L*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}$ *coordinate* N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier public rows zero head absurd`

Proof. Project the displayed carrier tuple of Definition 411.1. Each coordinate of K' is supplied by one stored component of H and one consumer route in C, so the transported packet has exactly the same visible rows as K at the BHist interface.

Apply the defining clause of Definition 411.2. The classifier reads only those projected rows, hence componentwise hsame transport and the recorded Cont reads preserve acceptance and preserve the two regular names, shared schedule, dyadic observations, tolerance row, ledger, seal row, package row, and local naming row. □

Theorem 411.14 (Regular Cauchy comparison tolerance row stability)**.** *For every accepted* $\mathsf{RegularCauchyCompari}$ *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

every displayed comparison request uses exactly the rational tolerance row τ *recorded in* L*. A classified replacement of* K *may transport* τ *along the stored component of* H *and read it along the displayed component of* C*, but it cannot replace* τ *by a tolerance row absent from the carrier ledger.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier tolerance row stability`

Proof. Use Definition 411.4 to identify the tolerance inspected by any displayed comparison request. The theorem pins that row to the rational tolerance coordinate τ recorded in L.

Now apply Definition 411.13. Classified replacement acts only through the stored componentwise transport and the displayed consumer read, so the tolerance coordinate can be carried across the accepted packet boundary but no unrecorded tolerance can enter the comparison. □

Theorem 411.15 (Regular Cauchy comparison common-window seal factorization)**.** *For every accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet, each* $\mathsf{Real}^{\uparrow}$ *seal read exported by the packet factors first through a shared* $\mathsf{StreamName}^{\uparrow}$ *window, then through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the rational tolerance row, the finite ledger, and the local certificate rows, before reaching the seal provenance row. Thus the seal handoff is mediated by the common comparison window and not by a direct host-real equality test or an unrecorded limit choice.*

Proof. Apply Definition 411.4 to force both regular names through the same finite window schedule before any dyadic observation is inspected.

Use Definition 411.14 for the next layer. The observed dyadic rows are compared only against the recorded rational tolerance row and the ledger that displays it.

Finally apply Definition 411.5. Its handoff route consumes the common-window comparison rows, the tolerance row, the finite ledger, and the local certificate coordinates before it reaches the $\mathsf{Real}^{\uparrow}$ seal row. □

Theorem 411.16 (Regular Cauchy comparison two-stage handoff)**.** *For every accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N),$$

every consumer-facing comparison first routes both $\mathsf{RegSeqRat}^{\uparrow}$ *names* X *and* Y *through the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule* W*. Only after that common-window read may the packet expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, the rational tolerance row* τ*, the finite ledger* L*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the displayed transports* H*, the displayed continuations* C*, the package provenance* P*, and the local naming row* N*.*

Proof. Begin with the common window. Apply Definition 411.6 to force the two regular names through the same displayed schedule W before any dyadic observation is read.

Next bind the numerical comparison layer. Apply Definition 411.14: the observation row D can be inspected only against the recorded rational tolerance τ and its finite ledger L.

Finally use Definition 411.5. The real-facing route consumes the common-window and tolerance rows before reaching E, H, C, P, N, so the handoff is a two-stage finite packet route rather than a direct real-completion read. □

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier two stage handoff`

Theorem 411.17 (Regular Cauchy comparison seal uniqueness boundary)**.** *Let* K *and* K' *be accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packets whose displayed rows agree on* $W, D, \tau, L, H, C, P, N$*. If two seal consumers read the packets through those same rows, then they obtain the same classifier-facing* $\mathsf{Real}^{\uparrow}$ *seal row. This uniqueness is a boundary statement about the displayed comparison certificate; it does not assert host real equality, select a limit, or identify quotient representatives of the two regular names.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier seal uniqueness boundary`

Proof. Project both packets to the classifier-facing rows. The shared $W, D, \tau, L, H, C, P, N$ coordinates give the same common-window schedule, dyadic observation layer, tolerance ledger, transport routes, continuation reads, package provenance, and local naming surface.

Apply Definition 411.13 to carry the agreed rows across the accepted packet boundary. Then apply Definition 411.15; a seal consumer can reach E only through that factored route.

Thus the two consumers see the same $\mathsf{Real}^{\uparrow}$ seal row at the classifier interface. The proof never introduces a host equality test or a selected completion limit, because neither is a coordinate of the displayed carrier. □

Theorem 411.18 (Regular Cauchy comparison three-window triangle)**.** *Let* K_{xy} *and* K_{yz} *be accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packets for regular Cauchy names* X, Y *and* Y, Z*. Suppose the* Y *rows in the two packets are classified through a shared transported* $\mathsf{StreamName}^{\uparrow}$ *window. Then their displayed* $W, D, \tau, L, E, H, C, P, N$ *rows refine to a third accepted comparison packet* K_{xz} *for* X *and* Z*.*

The $\mathsf{Real}^{\uparrow}$ *seal route of* K_{xz} *passes through the two shared finite windows and the dyadic tolerance ledger before it reaches the seal row. It does not pass through host real equality, a selected limit, or an unrecorded equality of regular-name representatives.*

Proof. Begin with the two finite windows. Apply Definition 411.6 to K_{xy} and K_{yz}, using the shared transported Y row to align the two displayed $\mathsf{StreamName}^{\uparrow}$ schedules.

Next compose the numerical rows. By Definition 411.14, each dyadic observation is read only with its recorded rational tolerance row and finite ledger. The two ledgers therefore provide the displayed tolerance surface needed for the X, Z comparison packet.

Now factor the real-facing seal. Apply Definition 411.15 to both source packets, then use Definition 411.17 at the shared Y window to identify the visible seal route carried across the middle packet boundary.

Finally repack the shared window rows, tolerance rows, ledgers, seal row, transports, continuations, provenance, and naming rows as K_{xz}. The constructed packet is accepted because every coordinate is one of the displayed $W, D, \tau, L, E, H, C, P, N$ rows exposed by the two source packets and their shared transported window. □

Theorem 411.19 (Regular Cauchy comparison intersection-window coverage)**.** *Let K be an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet comparing regular Cauchy names X and Y, and let a completion-facing consumer present two displayed schedules W_0 and W_1 through which it attempts to read the comparison. Then the consumer route must first expose a finite intersection schedule $W_\cap$ carrying the shared* $\mathsf{StreamName}^{\uparrow}$ *window for X and Y.*

On that same finite intersection schedule, the route must display the $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the rational tolerance row τ, the comparison ledger L, the* $\mathsf{Real}^{\uparrow}$ *seal row E, and the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows. No completion-facing read may bypass this intersection window by appealing directly to host real equality, quotient representatives, a selected limit, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier window intersection coverage`

Proof. First force the two displayed schedules to the common finite window. Apply Definition 411.6: the accepted comparison packet supplies a shared $\mathsf{StreamName}^{\uparrow}$ schedule before any dyadic observation can be read, so the two displayed schedules refine through an intersection row $W_\cap$.

Next attach the numerical rows to that intersection window. By Definition 411.14, the dyadic row D and rational tolerance row τ are read together with the finite ledger L; the consumer cannot detach the tolerance comparison from the displayed window.

Now use Definition 411.24. The classifier route reaches the shared window, dyadic observations, tolerance row, comparison ledger, seal row, and the visible BHist, hsame, Cont, Pkg, and NameCert rows as one finite packet.

Finally apply Definition 411.21. A completion-facing consumer can cite exactly that packet evidence, so the intersection-window coverage condition is necessary before the consumer reaches the $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 411.20 (Regular Cauchy comparison symmetric window swap)**.** *Let*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet. Swapping the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names X and Y preserves the shared* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the tolerance row τ, the ledger L, the seal row E, and the visible* hsame, Cont, Pkg, *and* NameCert *rows. The swapped packet therefore exposes the same* $\mathsf{Real}^{\uparrow}$ *classifier handoff through the same finite common-window comparison surface.*

Proof. First isolate the common window. By Definition 411.6, both regular names are read through the single displayed schedule W, so exchanging the order of X and Y does not create a second window row.

Next keep the numerical comparison rows fixed. Apply Definition 411.14: the dyadic observations in D are compared against exactly the recorded tolerance row τ and ledger L, and classified transport cannot substitute an unrecorded tolerance.

Now use Definition 411.13. The displayed hsame and Cont components move the two name coordinates componentwise while preserving the shared window, observation row, tolerance row, ledger, seal row, package provenance, and local certificate.

Finally apply Definition 411.5. The $\mathsf{Real}^{\uparrow}$ handoff reads only the same finite comparison surface, so the swapped presentation exposes the same classifier-facing handoff without importing host equality of regular streams. □

Theorem 411.21 (Regular Cauchy comparison completion consumer surface)**.** *Every* $\mathsf{Real}^{\uparrow}$*-completion consumer of an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet can cite exactly the finite comparison surface*

$$X, Y, W, D, \tau, L, E, H, C, P, N$$

as its packet evidence. The surface supplies the two regular names, the shared window schedule, the dyadic observations, the tolerance ledger, the seal row, transport and continuation evidence, package provenance, and local naming; it supplies no additional completion principle, host real equality, quotient representative, or chosen limit.

Proof. First package the route from the two regular names to the displayed comparison rows. Use Definition 411.16 to expose $X, Y, W, D, \tau, L, H, C, P, N$ in the required order.

Next package the seal boundary. Use Definition 411.17 to identify what a seal consumer can read when the comparison rows are fixed.

Finally apply Definition 411.5. Its consumer route is exactly the displayed packet surface, and the carrier has no coordinate for any further completion object. □

Theorem 411.22 (Regular Cauchy comparison NullSequence handoff)**.** *For every accepted* RegularCauchyComparis *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N),$$

if the displayed dyadic comparison ledger supplies a $\mathsf{NullSequence}^{\uparrow}$ *packet for the difference stream read from* X *and* Y*, then every* $\mathsf{Real}^{\uparrow}$ *equality seal read exported by* K *factors through the same shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the dyadic observation row* D*, the rational tolerance row* τ*, the finite comparison ledger* L*, and that* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail packet before reaching* E*. This handoff exports no quotient stream equality, host real equality, selected limit, trichotomy, or ambient completeness principle.*

Proof. Apply Definition 411.15. The $\mathsf{Real}^{\uparrow}$ seal route for K already passes through the shared $\mathsf{StreamName}^{\uparrow}$ window, the displayed dyadic observations, the tolerance row, and the finite ledger before it reaches the seal provenance row.

Use Definition 411.14 to identify the tolerance row used by the dyadic comparison ledger with the recorded row τ. The supplied $\mathsf{NullSequence}^{\uparrow}$ packet therefore reads the same difference stream and the same tolerance data that the comparison packet displays.

Finally apply Definition 380.10 and Definition 380.7. These results make the zero-tail packet the consumer boundary for the real equality seal, so the route reaches E only through the finite comparison surface and the null-sequence seal package, with no additional quotient, host equality, limit choice, trichotomy, or completeness object. □

Theorem 411.23 (Regular Cauchy comparison carrier admission)**.** *Every accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet*

$$K = (X, Y, W, D, \tau, L, E, H, C, P, N)$$

is a finite BHist *carrier over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}$ *row. Its admitted surface consists exactly of the two regular names, shared stream window, dyadic observation row, tolerance row, comparison ledger, real seal handoff, componentwise transports, consumer routes, provenance, and naming certificate.*

Proof. Project the accepted packet through the carrier definition and the local naming surface. The rows $X, Y, W, D, \tau, L, E, H, C, P, N$ display the two regular Cauchy names, the common finite schedule, dyadic observations, tolerance ledger, finite comparison ledger, real-seal handoff, transports, continuations, provenance, and local name.

Use the NullSequence handoff row to account for zero-tail comparison data when it is present. Because the handoff is read through the same $W, D, \tau, L, E, H, C, P, N$ coordinates, it does not add a second carrier or an external completion principle. □

Theorem 411.24 (Regular Cauchy comparison classifier window exactness)**.** *For an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet* K*, the common-window classifier and tolerance rows are exact under the displayed* hsame *and* Cont *routes. Every classifier read reaches the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, rational tolerance row* τ*, and comparison ledger* L *before it reaches the* $\mathsf{Real}^{\uparrow}$ *handoff* E*.*

Proof. Apply Definition 411.13. It transports classifier reads componentwise through the displayed hsame rows and reads them through the displayed Cont routes.

Next apply Definition 411.14 and Definition 411.15. These identify the tolerance and common-window rows used by the classifier with the rows τ and W already stored in the packet.

The resulting route reaches D and L in the same finite order before the seal handoff E. No classifier read can replace this route with host real equality, a quotient stream comparison, a selected limit, or ambient completeness. □

Theorem 411.25 (Regular Cauchy comparison seal handoff obligation)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal handoff of an accepted* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet factors only through the admitted carrier rows and the classifier-window exactness rows. It exports the displayed handoff* E *over* $X, Y, W, D, \tau, L, H, C, P, N$ *and exports no quotient representative, selected limit, host equality, trichotomy, or ambient completeness datum.*

Proof. Start with Definition 411.23. It fixes the finite packet rows that a seal consumer may inspect.

Use Definition 411.24 to force the classifier route through the shared window, dyadic observations, tolerance row, and comparison ledger. Then use Definition 411.5 to reach the real-facing handoff row E.

Finally apply Definition 411.21. The consumer surface is exactly the displayed packet evidence, so the seal handoff is an obligation row inside the carrier and not an appeal to quotient equality, host equality, a chosen limit, trichotomy, or completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier seal handoff obligation`

Theorem 411.26 (Regular Cauchy comparison dependency scope)**.** *The carrier admission, window exactness, Real classifier handoff, and common-window stability rows of Definition 411.23, Definition 411.4, Definition 411.5, and Definition 411.6 form the scoped comparison package over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Every comparison consumer is therefore forced through the two regular names, the shared stream window, the dyadic observation surface, the tolerance ledger, and the real seal handoff before it can read the packet as a $\mathsf{RegularCauchyComparison}^{\uparrow}$ *certificate.*

Proof. Begin with Definition 411.23. It admits exactly the finite packet rows over the listed BEDC dependencies and the local naming row.

Apply Definition 411.4 and Definition 411.6. These rows force both regular names through the same $\mathsf{StreamName}^{\uparrow}$ schedule before any dyadic observation or tolerance read is available.

Use Definition 411.5 for the final handoff. The real-facing classifier reads only the same comparison surface, so the dependency scope is the displayed package and no external completion object is added. □

Theorem 411.27 (Regular Cauchy comparison NullSequence seal scope)**.** *The tolerance exactness, seal factorization, and* $\mathsf{NullSequence}^{\uparrow}$ *handoff rows of Definition 411.7, Definition 411.15, and Definition 411.22 bind the comparison seal to the concrete zero-tail route over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

The scoped seal can be consumed only after the common-window dyadic comparison and the rational tolerance row have been displayed, and it exports no host equality, quotient representative, selected limit, trichotomy, or ambient completeness witness.

Proof. First apply Definition 411.7. The comparison reads the rational tolerance row already recorded in the finite ledger.

Next use Definition 411.15. It forces the $\mathsf{Real}^{\uparrow}$ seal route through the shared window, the dyadic observation row, the tolerance row, and the ledger.

Finally apply Definition 411.22. When a zero-tail packet is supplied, it is consumed through the same displayed comparison rows, so the seal remains inside the listed dependency scope. □

Theorem 411.28 (Regular Cauchy comparison StreamName RealUp scope package)**.** *The classifier transport, classifier-window exactness, seal handoff obligation, and completion consumer surface rows of Definition 411.13, Definition 411.24, Definition 411.25, and Definition 411.21 give the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *scoped package for* $\mathsf{RegularCauchyComparison}^{\uparrow}$*.*

This package binds the displayed $\mathsf{StreamName}^{\uparrow}$ *window route to the* $\mathsf{Real}^{\uparrow}$ *handoff over the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{NullSequence}^{\uparrow}$ *seal route when present, and the visible* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Use Definition 411.13 to move the comparison packet componentwise through the stored hsame and Cont rows. The transport preserves the two regular names, the shared window, dyadic observations, tolerance row, ledger, seal row, package row, and local naming row.

Apply Definition 411.24. The classifier reaches the shared window, dyadic observation row, rational tolerance row, and comparison ledger before it reaches the real handoff.

The public seal and consumer boundaries are then supplied by Definition 411.25 and Definition 411.21. Together they make the $\mathsf{StreamName}^{\uparrow}$ route and the $\mathsf{Real}^{\uparrow}$ handoff one scoped comparison package. □

Theorem 411.29 (Regular Cauchy comparison public export certificate)**.** *The carrier, classifier, window exactness, tolerance exactness,* $\mathsf{Real}^{\uparrow}$ *seal boundary,* $\mathsf{NullSequence}^{\uparrow}$ *handoff, and* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *scope package rows of Definition 411.23, Definition 411.13, Definition 411.4, Definition 411.7, Definition 411.15, Definition 411.22, and Definition 411.28 export one public* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *comparison certificate for downstream* $\mathsf{Real}^{\uparrow}$ *consumers.*

The exported certificate is a consumer-facing object over the displayed finite comparison packet. It introduces no quotient stream equality, selected host limit, host real equality, trichotomy claim, or ambient completeness theorem.

Proof. Begin with Definition 411.23 and Definition 411.13. These fix the finite packet and preserve its two regular names, shared stream window, dyadic observations, tolerance row, comparison ledger, seal row, package row, and local naming row under the displayed transport routes.

Next apply Definition 411.4 and Definition 411.7. They make the comparison readable only after the shared $\mathsf{StreamName}^{\uparrow}$ window and the rational tolerance row have both been displayed.

Use Definition 411.15 and Definition 411.22 for the real-facing seal. These rows keep the $\mathsf{Real}^{\uparrow}$ handoff tied to the same finite comparison evidence and to the $\mathsf{NullSequence}^{\uparrow}$ route when that route is present.

Finally apply Definition 411.28. It repacks the displayed window route and real handoff as one $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ comparison package, so the public export is exactly the stated consumer certificate and not an appeal to quotient equality, a selected host limit, host real equality, trichotomy, or ambient completeness. □

Theorem 411.30 (Regular Cauchy comparison standard bridge source)**.** *The standard bridge source for* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *is exactly the displayed comparison packet over two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the rational tolerance row, the finite comparison ledger, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the stored* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

No additional source object may enter the bridge read before these rows are displayed.

Proof. Start from the public export certificate of Definition 411.29. It already packages the two regular Cauchy names, shared stream window, dyadic observations, tolerance row, comparison ledger, real-facing seal, package row, and local naming row.

Apply the classifier and window exactness rows. They force the bridge source to read the shared $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ observations before reaching the $\mathsf{Real}^{\uparrow}$ seal handoff.

Finally use the transport and continuation rows. Since H, C, P, N are the only rows that move or replay the packet, a standard bridge source has exactly the displayed comparison data and no further source coordinate. □

Theorem 411.31 (Regular Cauchy comparison bridge non-escape)**.** *Every bridge-facing read from the public* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *certificate remains inside the source rows of Definition 411.30. The bridge exports no quotient regular stream, selected limit, host real equality, trichotomy, or ambient completeness witness.*

Proof. Project the source packet named in Definition 411.30. Its rows are the two regular names, shared stream window, dyadic observations, tolerance row, comparison ledger, real seal handoff, transport rows, continuation routes, package provenance, and local naming row.

Apply Definition 411.29. The public certificate is consumer-facing only over those displayed rows and explicitly excludes quotient stream equality, selected host limits, host real equality, trichotomy, and ambient completeness.

The bridge-facing read therefore has no remaining coordinate from which to obtain a quotient representative, a selected limit, an equality in a host real model, an order split, or a completeness principle. Those objects lie outside the displayed source. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyComparison}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, a shared $\mathsf{StreamName}^{\uparrow}$ schedule, displayed $\mathsf{DyadicRatCore}^{\uparrow}$ observations, a rational tolerance row, a finite ledger, $\mathsf{Real}^{\uparrow}$ seal provenance, displayed hsame and Cont routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature scope records carrier admission, classifier stability, common-window exactness, tolerance exactness, $\mathsf{Real}^{\uparrow}$ seal handoff, the public export certificate, the standard bridge source, and bridge non-escape over the displayed comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparison}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyComparisonUp.RegularCauchyComparisonCarrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close real completeness, quotient equality of regular streams, uniqueness of limits, host real construction, or bridge checking.

Upgrade path: Lean bridge checking remains the next verification step for the same $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ comparison package.

Chapter 412

A Concrete Naming Certificate for $\mathsf{ArzelaAscoli}^{\uparrow}$

The $\mathsf{ArzelaAscoli}^{\uparrow}$ interface names the constructive compact-metric reading of the Arzela–Ascoli theorem for families of continuous maps. A carrier records a family of $\mathsf{ContinuousMap}^{\uparrow}$ rows from a $\mathsf{CompactMetric}^{\uparrow}$ source into a $\mathsf{CompleteMetric}^{\uparrow}$ target, together with finite ProbeBundle nets, equicontinuity modulus rows, pointwise totally bounded image rows, displayed hsame and Cont transport, Pkg provenance, and a local NameCert row. It is a finite compact-continuity discovery target over $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, and $\mathsf{CompactUniformContinuity}^{\uparrow}$, not an open-cover compactness principle or a subsequence chosen by ambient choice.

The carrier consumes the finite-net source surface of chapter 129 through its $\mathsf{CompactMetric}^{\uparrow}$ row X, the target completion rows of chapter 128 through its $\mathsf{CompleteMetric}^{\uparrow}$ row Y, the family graph rows of chapter 130 through the finite ledger F, and the finite-modulus handoff of chapter 388 through the modulus row M. These sibling rows are consumed as carrier coordinates and ledger inputs, not as ambient compactness, function-space, or choice principles.

The family-level equicontinuity rows are read through chapter 413, while the finite modulus handoff is read through chapter 388; both are consumed only as finite net and modulus ledgers and do not add open-cover compactness or choice-selected subsequences.

Definition 412.1 (Arzela-Ascoli carrier)**.** *An* $\mathsf{ArzelaAscoli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, including its* $\mathsf{Met}^{\uparrow}$ *source, finite* ProbeBundle *net rows, and transported* $\mathsf{TotallyBounded}^{\uparrow}$ *coverage;*
2. *Y is the target* $\mathsf{CompleteMetric}^{\uparrow}$ *row whose distance and regular-Cauchy limit witnesses are read through displayed* $\mathsf{Met}^{\uparrow}$ *and completion rows;*
3. *F is a finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows from X to Y;*
4. *$\mathcal{P}$ records the finite source-net windows used to test the family;*
5. *E records equicontinuity modulus rows, each binding a requested target tolerance to one finite source-radius row uniformly over the family ledger;*
6. *I records pointwise* $\mathsf{TotallyBounded}^{\uparrow}$ *image rows for the finite family values at every bundled source-net point;*

7. *M records the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *or* $\mathsf{FanTheorem}^{\uparrow}$ *finite-modulus handoff rows that turn pointwise windows into a uniform finite source window;*
8. *B records the finite image-bundle selections and target metric distance witnesses used by the pointwise total-bounded rows;*
9. *H records componentwise* hsame *transports for source nets, target-completion rows, family rows, equicontinuity rows, pointwise image rows, finite-modulus rows, and bundle selections;*
10. *C records the visible* Cont *routes by which compact-metric, continuous-map, total-bounded, complete-metric, fan, and compact-uniform-continuity consumers read the packet;*
11. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, ProbeBundle, hsame, *and* Cont *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}$ *row.*

The carrier exports only finite net, modulus, image-bundle, completion-witness, transport, continuation, package, and naming rows. It contains no open-cover compactness row, no sequential compactness row, no unrecorded function-space topology, and no choice-selected subsequence.

Theorem 412.2 (Arzela-Ascoli NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}$ *surface consists exactly of carrier habitation by the packet of Definition 412.1; classifier transport of the source compact-metric rows, target complete-metric rows, family of continuous-map rows, equicontinuity rows, pointwise image-bundle rows, and finite-modulus rows; finite-net coverage through* ProbeBundle*; target-side completion readback through* $\mathsf{CompleteMetric}^{\uparrow}$*; and downstream non-escape through the recorded* hsame, Cont, Pkg, *and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ArzelaAscoliUp.ArzelaAscoliPacket namecert obligations`

Proof. Carrier habitation is the displayed tuple of Definition 412.1. Each source-net row, family row, target-completion row, equicontinuity row, pointwise image-bundle row, finite-modulus handoff, transport row, continuation route, package row, and local certificate row is visible before a consumer can read it.

Classifier transport is componentwise. The $\mathsf{CompactMetric}^{\uparrow}$ coordinate transports source finite nets, the $\mathsf{ContinuousMap}^{\uparrow}$ family coordinate transports graph and pointwise modulus rows, the $\mathsf{TotallyBounded}^{\uparrow}$ coordinate transports finite image bundles, and the $\mathsf{CompleteMetric}^{\uparrow}$ coordinate transports target regular-Cauchy limit witnesses.

The finite-modulus coordinate M is then transported through the same hsame and Cont routes. It may cite $\mathsf{FanTheorem}^{\uparrow}$ or $\mathsf{CompactUniformContinuity}^{\uparrow}$, but only through their finite branch-depth, witness-window, and compact-uniform-modulus ledgers.

These rows exhaust the certificate. Since the carrier has no coordinate for open covers, host function spaces, choice-selected subsequences, or ambient sequential compactness, the local naming certificate cannot export any of those objects. □

Theorem 412.3 (Arzela-Ascoli carrier habitation)**.** *There is a finite inhabited* $\mathsf{ArzelaAscoli}^{\uparrow}$ *carrier of the form*

$$A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N)$$

whose source row is a $\mathsf{CompactMetric}^{\uparrow}$ *row, whose target row is a* $\mathsf{CompleteMetric}^{\uparrow}$ *row, whose family rows are* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, whose pointwise image rows are* $\mathsf{TotallyBounded}^{\uparrow}$ *rows, whose finite handoff rows are read through* $\mathsf{FanTheorem}^{\uparrow}$ *or* $\mathsf{CompactUniformContinuity}^{\uparrow}$*, and whose visible bookkeeping is a* ProbeBundle, hsame, Cont, Pkg, *and* NameCert *packet as specified in Definition 412.1.*

Proof. Use the displayed packet shape of Definition 412.1. The source coordinate X, target coordinate Y, finite family ledger F, source-net packet $\mathcal{P}$, equicontinuity ledger E, image ledger I, handoff ledger M, image bundle B, transport row H, continuation row C, provenance row P, and naming row N form a finite BHist packet.

The equicontinuity net handoff of Definition 412.6 verifies that the source-net, family, pointwise-image, completion, and handoff rows are read as a single finite compact-continuity surface. No open-cover or subsequence coordinate is needed for habitation. □

Theorem 412.4 (Arzela-Ascoli completion-window readback)**.** *Let $A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N)$ be an accepted* ArzelaAscoli$^{\uparrow}$ *carrier. Every target-side completion read exported by A factors through the* CompleteMetric$^{\uparrow}$ *row Y, the finite* ProbeBundle *source windows $\mathcal{P}$, the* ContinuousMap$^{\uparrow}$ *family ledger F, the* TotallyBounded$^{\uparrow}$ *image rows I, the finite handoff row M from* FanTheorem$^{\uparrow}$ *or* CompactUniformContinuity$^{\uparrow}$*, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with a completion read in the target. The carrier first exposes the target coordinate Y, so every limit or regular-Cauchy read is a CompleteMetric$^{\uparrow}$ read rather than a host completion choice.

Next read the source windows $\mathcal{P}$ and family ledger F. By Definition 412.6, the selected source-net rows and equicontinuity rows route every family value through the same finite compact-continuity handoff.

The pointwise image rows I and image-bundle rows B then supply the finite totally bounded target packets. The finite handoff row M records whether the branch-depth or compact-uniform-continuity route is used, and H, C, P, N keep that read stable under classifier transport. □

Theorem 412.5 (Arzela-Ascoli non-escape obligation)**.** *The public boundary of an accepted* ArzelaAscoli$^{\uparrow}$ *packet exports only the source* CompactMetric$^{\uparrow}$ *row, target* CompleteMetric$^{\uparrow}$ *row, finite* ContinuousMap$^{\uparrow}$ *family ledger,* ProbeBundle *source-net rows, equicontinuity rows, pointwise* TotallyBounded$^{\uparrow}$ *image rows,* FanTheorem$^{\uparrow}$ *or* CompactUniformContinuity$^{\uparrow}$ *handoff rows, image-bundle rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence. It exports no open-cover compactness, no sequential compactness, no ambient function-space topology, and no choice-selected subsequence.*

Proof. Inspect the coordinates of Definition 412.1. They are exactly X, Y, F, $\mathcal{P}$, E, I, M, B, H, C, P, and N.

The handoff theorem Definition 412.6 routes compact-continuity consumers through finite source nets, equicontinuity ledgers, pointwise image bundles, target completion rows, and the provenance rows. This accounts for every downstream read named by the carrier.

Since no carrier coordinate names open covers, sequential compactness, a compact-open topology, or a subsequence selector, the local naming certificate cannot export those objects. □

Theorem 412.6 (Arzela-Ascoli equicontinuity net handoff)**.** *Every compact-continuity consumer reading an accepted* ArzelaAscoli$^{\uparrow}$ *packet obtains a finite handoff: one displayed* ProbeBundle *source net from* CompactMetric$^{\uparrow}$*, equicontinuity modulus rows over the* ContinuousMap$^{\uparrow}$ *family ledger, pointwise totally bounded image rows in the* CompleteMetric$^{\uparrow}$ *target, and the transported* Cont *and* Pkg *provenance tying those rows to the local* NameCert$_{\text{ArzelaAscoli}^{\uparrow}}$ *certificate.*

The handoff supplies finite net data for a constructive Arzela–Ascoli argument and does not supply open-cover compactness, sequential compactness, an ambient compact-open topology, or a choice-selected subsequence.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliUp.ArzelaAscoliPacket equicontinuity net handoff`

Proof. Project the source coordinate X and the source-net coordinate $\mathcal{P}$. The compact-metric input supplies finite ProbeBundle coverage, so a consumer source read is routed through displayed net centers and their coverage witnesses.

Read the family coordinate F together with the equicontinuity ledger E. For a requested target tolerance, the ledger selects source-radius rows uniformly over the displayed family entries and routes the corresponding continuous-map modulus reads through the same finite net.

Next project the pointwise image coordinate I and the image-bundle coordinate B. At each bundled source-net point, the target values of the displayed family are covered by finite totally bounded image rows and by target metric distance witnesses read inside the $\mathsf{CompleteMetric}^{\uparrow}$ target.

Finally attach the finite-modulus coordinate M and the provenance rows H, C, P, N. These rows preserve the same source net, equicontinuity windows, pointwise image bundles, and target-completion witnesses under classifier transport, and no omitted carrier coordinate can add open-cover compactness, sequential compactness, a host function-space topology, or a choice-selected subsequence. □

Definition 412.7 (Arzela-Ascoli modulus carrier)**.** *Let*

$$A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N)$$

be an accepted $\mathsf{ArzelaAscoli}^{\uparrow}$ *carrier. The* $\mathsf{ArzelaAscoli}^{\uparrow}$ *modulus carrier is the finite subcarrier*

$$A_M = (X, F, \mathcal{P}, E, I, M, H, C, P, N)$$

obtained by projecting the compact-metric source nets, continuous-map family rows, equicontinuity ledgers, pointwise totally bounded image rows, $\mathsf{FanTheorem}^{\uparrow}$ *or* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite handoff rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}$ *row from* A*.*

Theorem 412.8 (Arzela-Ascoli modulus NameCert obligations)**.** *For every accepted* $\mathsf{ArzelaAscoli}^{\uparrow}$ *packet, the modulus subcarrier of Definition 412.7 binds exactly the finite compact-continuity data needed by later modulus consumers: compact-metric source nets in* X*,* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows in* F*, finite source windows in* $\mathcal{P}$*, equicontinuity ledgers in* E*, pointwise totally bounded image rows in* I*, and* $\mathsf{FanTheorem}^{\uparrow}$ *or* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *handoff rows in* M*, transported through* H, C, P, N*.*

It supplies no open-cover compactness, ambient function-space topology, sequential compactness, or choice-selected subsequence.

Proof. Project the displayed carrier of Definition 412.1. The rows named in Definition 412.7 are exactly the source-net, family, equicontinuity, pointwise-image, finite-handoff, transport, replay, provenance, and naming coordinates of the accepted packet.

Apply Definition 412.6. It shows that every compact-continuity consumer reads finite source nets, equicontinuity windows, pointwise image bundles, target-completion witnesses, and finite handoff rows through one transported surface.

Finally use Definition 412.5. Since the accepted carrier exports no open-cover compactness row, host function-space topology, sequential compactness row, or subsequence selector, the projected modulus carrier cannot acquire any of those objects. □

Theorem 412.9 (Arzela-Ascoli scoped compact-continuity package)**.** *Let*

$$A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N)$$

be an accepted ArzelaAscoli$^{\uparrow}$ *carrier. Then A exports exactly the scoped compact-continuity packet consisting of the compact-metric source rows in X, the complete-metric target rows in Y, the continuous-map family ledger F, the* ProbeBundle *source-net rows $\mathcal{P}$, the equicontinuity rows E, the pointwise* TotallyBounded$^{\uparrow}$ *image rows I, the* FanTheorem$^{\uparrow}$ *or* CompactUniformContinuity$^{\uparrow}$ *handoff rows M, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance carried by B, H, C, P, N.*

Every compact-continuity read of this packet factors through those rows, and no read of A supplies open-cover compactness, sequential compactness, an ambient compact-open topology, an unrecorded function-space carrier, or a choice-selected subsequence.

Proof. First use Definition 412.2. It identifies the local certificate surface with carrier habitation, classifier transport, finite source-net coverage, target completion readback, and non-escape through the displayed transport, continuation, package, and naming rows.

Next apply Definition 412.4. This routes every target-side completion read through $Y, \mathcal{P}, F, I, M$ and the provenance rows, so the complete-metric target contribution is part of the same finite compact-continuity packet rather than an ambient completion choice.

Then combine Definition 412.5 with Definition 412.6. The non-escape boundary lists the exported coordinates, and the handoff theorem shows that compact-continuity consumers read finite source nets, equicontinuity windows, pointwise image bundles, target-completion witnesses, and finite handoff rows through one transported surface.

These component rows are exactly the coordinates displayed in A. Repacking them gives the scoped compact-continuity package, while the absence of any further carrier coordinate rules out open-cover compactness, sequential compactness, ambient function-space topology, unrecorded carriers, and choice-selected subsequences. □

Theorem 412.10 (Arzela-Ascoli public certificate)**.** *For every accepted* ArzelaAscoli$^{\uparrow}$ *carrier*

$$A = (X, Y, F, \mathcal{P}, E, I, M, B, H, C, P, N),$$

the scoped compact-continuity package of Definition 412.9 is the public $\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}$ *surface consumed by downstream compactness and completion chapters. It exports exactly the finite rows over* CompactMetric$^{\uparrow}$*,* CompleteMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*,* FanTheorem$^{\uparrow}$ *or* CompactUniformContinuity$^{\uparrow}$*,* ProbeBundle*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it exports no open-cover compactness, sequential compactness, choice-selected subsequence, ambient compact-open topology, or bridge checking.*

Proof. Start with the scoped compact-continuity package. It already repacks the compact-metric source rows, complete-metric target rows, continuous-map family ledger, source-net rows, equicontinuity rows, pointwise totally bounded image rows, finite modulus handoff rows, and the displayed structural provenance.

Read the package as a public surface. Downstream compactness and completion consumers receive the same rows through ProbeBundle source nets, CompleteMetric$^{\uparrow}$ target readback, ContinuousMap$^{\uparrow}$ family data, TotallyBounded$^{\uparrow}$ image rows, and the finite handoff supplied by FanTheorem$^{\uparrow}$ or CompactUniformContinuity$^{\uparrow}$.

The transport, replay, provenance, and local naming rows are exactly H, C, P, N. Because no further carrier coordinate appears in A, public consumption cannot add open-cover compactness, sequential compactness, a selected subsequence, an ambient compact-open topology, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoli}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source, a $\mathsf{CompleteMetric}^{\uparrow}$ target, a finite family ledger of $\mathsf{ContinuousMap}^{\uparrow}$ rows, finite ProbeBundle source nets, equicontinuity modulus rows, pointwise $\mathsf{TotallyBounded}^{\uparrow}$ image rows, finite $\mathsf{FanTheorem}^{\uparrow}$ or $\mathsf{CompactUniformContinuity}^{\uparrow}$ modulus handoff rows, displayed hsame and Cont transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 412.10 bridges the public compact-continuity packet at the paper-theory level over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, ProbeBundle, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoli}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ArzelaAscoliUp.ArzelaAscoliPacket\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, ambient compact-open topology, or a choice-selected subsequence.

Upgrade path: No further paper-axis closure remains; bridge checking is a verification-axis matter unless a checked standard bridge target is present.

Chapter 413

A Concrete Naming Certificate for $\mathsf{EquicontinuityFamily}^{\uparrow}$

The $\mathsf{EquicontinuityFamily}^{\uparrow}$ interface names the constructive finite-ledger surface for a family of continuous maps with a shared source metric, shared target metric, finite center probes, and rational tolerance histories. It is the BEDC reading of the equicontinuity object used in Kelley, *General Topology*, and Bishop–Bridges, *Constructive Analysis*, re-expressed as finite rows over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, and $\mathsf{CompactUniformContinuity}^{\uparrow}$. The packet does not claim open-cover compactness, host function extensionality, or a chosen function-space topology.

The finite-center modulus ledger consumes the compact-uniform continuity packet of chapter 388 before exposing the family-level equicontinuity row.

Definition 413.1 (EquicontinuityFamily carrier)**.** *An* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, F, \mathcal{P}, R, M, H, C, P, N)$$

with the following visible coordinates:

1. *X is the shared source* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *row whose finite source reads are routed through displayed metric and compact-metric data;*
2. *Y is the shared target* $\mathsf{Met}^{\uparrow}$ *row whose distance comparisons are routed through displayed metric-distance witnesses;*
3. *F is a finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows from X to Y;*
4. *$\mathcal{P}$ is a finite* $\mathsf{ProbeBundle}$ *of source centers used as the displayed testing window for the family;*
5. *R records the requested* $\mathsf{Rat}^{\uparrow}$ *tolerance histories and the source-radius histories offered in response;*
6. *M records pointwise modulus rows indexed by the centers in $\mathcal{P}$, uniformly over the finite family ledger F;*
7. *H records componentwise* hsame *transports for the source metric, target metric, graph rows, center bundle, rational tolerance rows, modulus rows, and naming rows;*
8. *C records the visible* Cont *routes by which metric, compact-metric, continuous-map, rational-tolerance, modulus, and equicontinuity-family consumers read the packet;*

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, hsame, *and* Cont *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$ *row.*

The carrier exports only the displayed source and target metric rows, finite family rows, rational tolerance rows, pointwise finite-center modulus rows, transport rows, continuation routes, package provenance, and local naming provenance. It contains no host function equality, no open-cover compactness, no selected modulus function, and no ambient function-space topology.

Lean4 ✓ `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyTasteGate single carrier alignment`

Theorem 413.2 (EquicontinuityFamily NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier habitation by the finite packet of Definition 413.1;*
2. *classifier transport comparing the shared source and target metric rows, the family graph rows, the pointwise modulus rows, the finite center bundle, the rational tolerance rows, the transport routes, and the provenance componentwise;*
3. *shared finite-center modulus coverage, stating that every displayed family read used by the packet factors through one center row of $\mathcal{P}$ and the corresponding modulus row in M;*
4. *rational tolerance stability, stating that tolerance and radius rows are transported through the* $\mathsf{Rat}^{\uparrow}$ *classifier and the stored* hsame *and* Cont *routes;*
5. *ledger non-escape, stating that every downstream read remains inside the displayed metric, continuous-map, modulus, package, and local naming rows.*

Lean4 ✓ `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyPacket namecert obligations`

Proof. Carrier habitation is projection of Definition 413.1. The shared source row, shared target row, family ledger, finite center bundle, rational tolerance ledger, pointwise modulus rows, transport rows, continuation routes, package provenance, and local naming row are all visible coordinates of the same finite packet.

Classifier transport is componentwise. The source and target metric rows transport their displayed distance witnesses, the $\mathsf{ContinuousMap}^{\uparrow}$ rows transport the graph and pointwise modulus reads, the center bundle transports the finite probe indices, and the rational rows transport requested tolerances together with their offered source radii.

For shared finite-center coverage, read a family member through F and a requested tolerance through R. The packet routes that read to a displayed center of $\mathcal{P}$ and then to the corresponding row of M, where the same source-radius history is used uniformly over the finite family ledger.

Rational tolerance stability follows from the same transported R rows and the displayed $\mathsf{Rat}^{\uparrow}$ comparisons. Replaying the Cont routes preserves which tolerance request, which radius response, and which center-indexed modulus row are being consumed.

These rows exhaust the certificate surface. Because the carrier has no coordinate for host function equality, open-cover compactness, selected modulus functions, or ambient function-space topology, no classifier or consumer route can export those objects. □

Theorem 413.3 (EquicontinuityFamily shared-center coverage)**.** *Every displayed family read from an accepted* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *packet factors through one finite* ProbeBundle *center in $\mathcal{P}$ and the corresponding shared modulus row in M. The read uses the shared source row X, shared target row*

Y, finite ContinuousMap$^{\uparrow}$ *family ledger F, rational tolerance row R, componentwise* hsame *rows H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N of Definition 413.1.*

The coverage statement exports no open-cover compactness, host function extensionality, selected global modulus function, or ambient function-space topology.

Lean4 ✓ `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyPacket shared center coverage`

Proof. Project the carrier of Definition 413.1. Its only center data are the finite probe bundle $\mathcal{P}$, and its only modulus data are the pointwise rows in M indexed by those centers.

Read a displayed family member through the finite ledger F and a tolerance through R. The NameCert obligation theorem routes that read to one center of $\mathcal{P}$ and the matching shared modulus row, uniformly over the finite family ledger.

The transport rows H and continuation routes C preserve the selected center and modulus row as finite packet data. Since the carrier has no coordinates for open covers, host function equality, selected modulus functions, or function-space topology, the shared-center read cannot escape to those objects. □

Theorem 413.4 (EquicontinuityFamily rational-radius transport)**.** *The* Rat$^{\uparrow}$ *tolerance and radius rows in an accepted* EquicontinuityFamily$^{\uparrow}$ *packet are transported only through the stored* hsame *rows and displayed* Cont *routes. Transport preserves the requested tolerance row, the offered rational source-radius row, the selected probe-center row, and the corresponding modulus row without selecting a modulus function outside the finite packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyPacket rational radius transport`

Proof. Begin with the tolerance ledger R in Definition 413.1. It records requested rational tolerances and offered source radii as displayed Rat$^{\uparrow}$ histories.

Apply the classifier transport row of Definition 413.2. The Rat$^{\uparrow}$ rows move componentwise through H, and the read routes are replayed through C.

Use shared-center coverage to keep the transported radius attached to the same finite probe-center and modulus-row pattern. The operation is therefore transport of stored rational rows, not selection of a hidden modulus function. □

Theorem 413.5 (EquicontinuityFamily ledger non-escape package)**.** *The carrier of Definition 413.1, classifier transport in Definition 413.2, shared-center coverage in Definition 413.3, and rational-radius transport in Definition 413.4 assemble the ledger non-escape package for* EquicontinuityFamily$^{\uparrow}$*. Every consumer read remains within the displayed* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* UniformContinuity$^{\uparrow}$*,* CompactUniformContinuity$^{\uparrow}$*,* Rat$^{\uparrow}$*,* ProbeBundle*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The package excludes open-cover compactness, host function extensionality, selected modulus functions, and ambient function-space topology.

Proof. First bind the packet rows. The carrier lists the shared metric rows, finite family ledger, probe bundle, rational histories, modulus rows, transports, routes, provenance, and local name row.

Next bind the coverage and transport rows. Shared-center coverage forces each family read through one displayed probe center and matching modulus row, while rational-radius transport keeps tolerance and radius data inside Rat$^{\uparrow}$, hsame, and Cont rows.

Finally apply the ledger non-escape clause of the NameCert obligation theorem. It states that downstream reads stay inside the displayed metric, continuous-map, modulus, package, and local naming rows. Repacking these rows gives the announced non-escape package, with no route to

open-cover compactness, host function extensionality, selected modulus functions, or function-space topology. □

Lean4 ✓ `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyPacket ledger non escape package`

Theorem 413.6 (EquicontinuityFamily public NameCert export)**.** *The finite-center modulus ledger of Definition 413.1, compact-uniform continuity dependency row, rational-radius transport of Definition 413.4, shared-center coverage of Definition 413.3, and ledger non-escape package of Definition 413.5 assemble the public* $\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$ *export. A public consumer receives exactly the displayed shared source and target metric rows, finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger, center bundle, rational tolerance and radius rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

The export does not add open-cover compactness, host function extensionality, a selected modulus function, or ambient function-space topology.

Proof. First bind the finite rows displayed by the carrier. The compact-uniform continuity dependency is read through the stored source metric or compact-metric row, target metric row, finite family ledger, center bundle, rational tolerance ledger, and center-indexed modulus rows.

Next use rational-radius transport and shared-center coverage. They force every tolerance request and radius response to remain attached to a displayed probe center and matching modulus row, uniformly over the finite family ledger.

Finally apply the ledger non-escape package. Since every consumer read is already routed through the visible transport, replay, provenance, and local naming rows, repacking those rows gives the public $\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$ export with no additional object outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EquicontinuityFamily}^{\uparrow}$ packet is generated from a shared source metric or compact-metric row, a shared target metric row, a finite family ledger of $\mathsf{ContinuousMap}^{\uparrow}$ graph rows, a finite ProbeBundle of source centers, rational tolerance histories, center-indexed modulus rows, displayed hsame and Cont transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public package exports the finite equicontinuity-family NameCert obligation surface consumed by compact-uniform and Arzela–Ascoli style routes through the carrier, finite-family ledger, shared source and target metric rows, finite center probes, rational tolerance histories, center-indexed modulus rows, displayed hsame and Cont transport, Pkg provenance, and local NameCert row.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EquicontinuityFamily}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.EquicontinuityFamilyUp.EquicontinuityFamilyPacket\_namecert\_obligations`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close open-cover compactness, host function extensionality, ambient function-space topology, selected modulus functions, or bridge checking.

Upgrade path: No theory-axis upgrade remains; a machine-checked standard bridge is the remaining formal-axis work for the same displayed equicontinuity-family packet.

Chapter 414

RegularCauchyInterleaving NameCert Construction

The $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ interface isolates the standard alternating construction for two regular Cauchy names. In the constructive-analysis reading of Bishop and Bridges, *Constructive Analysis*, the interleaving is not a quotient of two completed real values; it is a finite certificate that records which regular Cauchy observation is read at each unary selector step, how the two schedules share a modulus ledger, and where the resulting $\mathsf{Real}^{\uparrow}$ seal can be consumed.

The interleaving packet consumes the regular rational source of chapter 301, the finite schedule surface of chapter 126, and the terminal seal of chapter 38. The $\mathsf{RegSeqRat}^{\uparrow}$ sibling supplies the two regular names, $\mathsf{StreamName}^{\uparrow}$ supplies the finite schedules, and $\mathsf{Real}^{\uparrow}$ supplies the downstream seal that can be read only after the selected schedule and combined modulus ledger have been displayed.

Definition 414.1 (Regular Cauchy interleaving source)**.** *A* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *source packet is a finite* BHist *row*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy names whose point rows may be inspected;*
2. *S_A and S_B are the two* $\mathsf{StreamName}^{\uparrow}$ *schedules that expose the finite observation windows for A and B;*
3. *σ is the alternating unary selector row, with even selector rows reading from A and odd selector rows reading from B;*
4. *M is the combined modulus ledger, recording the common precision budget and the component budget used on each selected side;*
5. *E_A and E_B are the* $\mathsf{Real}^{\uparrow}$ *seal readbacks attached to the two source names;*
6. *E is the interleaved* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the selected source readback and the common modulus ledger;*
7. *H stores the componentwise* hsame *transports for the two regular names, both schedules, the selector, the combined modulus ledger, and all three seal rows;*

8. *C stores the displayed* Cont *routes for selecting a source side, reading its scheduled window, checking the combined modulus row, and exposing the interleaved seal;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}$ *row.*

The packet has no coordinate for a quotient real, a selected host limit, or an equality principle beyond the displayed hsame *and* Cont *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.TasteGate.RegularCauchyInterleavingTasteGate single carrier alignment`

Theorem 414.2 (Regular Cauchy interleaving classifier)**.** *Two accepted* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *source packets classify together exactly when their two* $\mathsf{RegSeqRat}^{\uparrow}$ *source names, two* $\mathsf{StreamName}^{\uparrow}$ *schedules, alternating selector rows, combined modulus ledgers, source* $\mathsf{Real}^{\uparrow}$ *readbacks, interleaved* $\mathsf{Real}^{\uparrow}$ *seal rows, package provenance, and local naming rows are transported componentwise by the displayed* hsame *comparisons and read through the displayed* Cont *routes. The classifier compares the selected finite window on the even side with the transported even-side window, and the selected finite window on the odd side with the transported odd-side window, always under the same combined modulus ledger.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket classifier transport`

Proof. Project the two accepted source packets. Their carrier rows expose only the two regular names, the two stream schedules, the selector, the modulus ledger, the seal readbacks, and the displayed transport and continuation rows.

For an even selector row, the classifier reads the scheduled window from the first $\mathsf{RegSeqRat}^{\uparrow}$ name through S_A, then checks the corresponding component of M. For an odd selector row, it performs the same finite read through S_B and the second component of M.

The rows H and C record exactly the transports and consumer routes used by these reads. Repacking the transported side-specific read with P and N gives the same classifier surface for the interleaved seal E, and no quotient real or host-limit comparison is available. □

Theorem 414.3 (Regular Cauchy interleaving NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}$ *surface for the source packet of Definition 414.1 consists of five finite rows:*

1. *carrier admission: the accepted object is the displayed* BHist *packet I;*
2. *classifier equivalence: classification is the finite even-and-odd window comparison of Definition 414.2;*
3. hsame *transport: the two source names, two schedules, selector, combined modulus ledger, and seal rows are preserved componentwise by the stored transport rows;*
4. *combined modulus exactness: each interleaved precision request is discharged by the displayed common budget and the component budget selected by σ;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape: the interleaved seal E can be consumed only after the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, its* $\mathsf{StreamName}^{\uparrow}$ *schedule row, and the combined modulus ledger have been displayed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket namecert obligations`

Proof. Carrier admission is the projection of the finite packet I. The source rows A, B, S_A, S_B, σ, M, E_A, E_B, and E are stored coordinates, while H, C, P, and N record transport, continuation, package, and local naming provenance.

Classifier equivalence is Definition 414.2. Its even and odd cases use the same packet rows and differ only in which source schedule the selector chooses.

The hsame row transports each displayed coordinate componentwise, and the Cont row records the consumer route through the selected schedule and modulus ledger. The combined modulus row is exact because a request is answered by the common budget together with the component budget named by σ.

Finally, the real-facing coordinate E is downstream of the selected regular readback and not an independent host real. Since the packet has no field for quotient equality, selected limits, or ambient completeness, the seal cannot escape the displayed interleaving certificate. □

Theorem 414.4 (Regular Cauchy interleaving carrier admission)**.** *Let*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

be a displayed packet satisfying the $\mathsf{RegSeqRat}^{\uparrow}$ *rows for A and B, the* $\mathsf{StreamName}^{\uparrow}$ *rows for S_A and S_B, the alternating selector row σ, the combined modulus ledger M, the three* $\mathsf{Real}^{\uparrow}$ *seal rows E_A, E_B, and E, the displayed* hsame *and* Cont *rows, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}$ *row. Then I is an accepted* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket carrier admission`

Proof. Project the displayed coordinates of I. The first two coordinates give the required regular Cauchy names, the next two give their finite stream schedules, and the selector and modulus coordinates give the alternating read policy and its common budget ledger.

The seal coordinates E_A, E_B, and E are already part of the same finite history row. The rows H and C supply the componentwise transport and consumer routes, while P and N supply the package provenance and local naming certificate.

Repack these projected rows in the order of Definition 414.1. The definition has no further carrier field, so the displayed packet is exactly an admitted carrier. □

Theorem 414.5 (Regular Cauchy interleaving selector parity exactness)**.** *For every accepted* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *source packet I, each even selector read factors through the first regular name A, the schedule S_A, and the first component of the combined modulus ledger M, while each odd selector read factors through the second regular name B, the schedule S_B, and the second component of M. Componentwise* hsame *transports and the displayed* Cont *routes preserve this even-side and odd-side partition.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket selector parity exactness`

Proof. Unpack I as in Definition 414.1. The selector coordinate σ is displayed as alternating, so an even selector row exposes the first source side and an odd selector row exposes the second source side.

In the even case, Definition 414.2 reads the finite window through A and S_A and checks the corresponding first component of M. In the odd case, the same classifier reads through B and S_B and checks the second component of M.

The stored row H transports the selector, both schedules, both regular names, and the combined modulus ledger componentwise. The stored row C only routes consumers through the selected schedule and the selected ledger component, so transport preserves the parity partition rather than mixing the two source sides. □

Theorem 414.6 (Regular Cauchy interleaving swap symmetry boundary)**.** *Let $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E$ be an accepted* RegularCauchyInterleaving$^{\uparrow}$ *source packet. Exchanging the two* RegSeqRat$^{\uparrow}$ *source names A, B, exchanging the two schedules S_A, S_B, and replacing the alternating selector σ by its opposite selector preserves the same public* Real$^{\uparrow}$ *seal boundary through the displayed* hsame *transports,* Cont *routes, combined modulus ledger,* Pkg *provenance, and local* NameCert *row. The boundary is only the finite selector-swapped interleaving surface; it does not assert quotient real equality, selected host limits, ambient completeness, or access to a third schedule.*

Proof. Apply Definition 414.5. The forward selector sends even reads through A, S_A and the first component of M, and odd reads through B, S_B and the second component.

After the source and schedule exchange, the opposite selector sends its odd reads through the transported A, S_A route and its even reads through the transported B, S_B route. Thus the two presentations inspect the same two finite windows with only the selector parity exchanged.

Use Definition 414.9 to keep each selected read attached to the same component of M. Then use Definition 414.10 to rule out any route through a schedule other than S_A or S_B.

Finally apply Definition 414.8. The real-facing seal can be consumed only after the selected source row, selected schedule, combined ledger, displayed transport, displayed continuation route, package provenance, and local name have been read. Since the carrier has no quotient-real, host-limit, ambient-completeness, or third-schedule coordinate, the swapped presentation preserves exactly the public seal boundary and nothing stronger. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket swap symmetry boundary`

Theorem 414.7 (Regular Cauchy interleaving selector transport determinacy)**.** *For classified accepted* RegularCauchyInterleaving$^{\uparrow}$ *packets, transported selector rows preserve the even and odd routes. Every even transported read factors through the transported A-side schedule and the first component of the combined modulus ledger, and every odd transported read factors through the transported B-side schedule and the second component. No transported read can switch sides or use a third schedule.*

Proof. Project the classifier statement of Definition 414.2. It compares the two source names, the two schedules, the selector row, and the combined modulus ledger componentwise through the displayed hsame and Cont data.

Apply Definition 414.5 before transport. Even reads are routed through A, S_A, and the first component of M, while odd reads are routed through B, S_B, and the second component.

Apply the same parity exactness after the hsame transport. The transported selector still exposes the transported even side through the transported A-schedule and the transported odd side through the transported B-schedule.

Use Definition 414.9 to keep the selected ledger component fixed. Then use Definition 414.10 to rule out unscheduled routes: the carrier has only the two transported schedules, so there is no coordinate from which a third schedule or side switch could be read. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket selector transport determinacy`

Theorem 414.8 (Regular Cauchy interleaving RealUp seal non-escape)**.** *Let I be an accepted* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *source packet with interleaved seal E. Any* $\mathsf{Real}^{\uparrow}$ *consumer of E must pass through the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the selected* $\mathsf{StreamName}^{\uparrow}$ *schedule, the combined modulus ledger, the displayed* hsame *transport and* Cont *route, and the package and naming rows P and N. No consumer of E can read quotient real equality, a host limit, or ambient completeness from the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket real seal non escape`

Proof. Carrier admission exposes only the fields listed in Definition 414.1. Thus the real-facing row E is not a standalone real object; it is a coordinate in the same finite packet as the two regular names, their schedules, the selector, and the ledger.

Definition 414.5 identifies the required readback path for each selector parity. The even route passes through A, S_A, and the first component of M, while the odd route passes through B, S_B, and the second component of M.

The NameCert obligations then attach the displayed hsame transport, Cont consumer route, package provenance, and local naming row to that selected path. Since the carrier definition contains no quotient-equality coordinate, no host-limit coordinate, and no ambient-completeness coordinate, a $\mathsf{Real}^{\uparrow}$ consumer has no route out of the displayed certificate. □

Theorem 414.9 (Regular Cauchy interleaving combined modulus exactness)**.** *Let*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *packet. Every precision request for the interleaved* $\mathsf{Real}^{\uparrow}$ *seal first reads the selector row σ, then reads the selected schedule S_A or S_B, and then checks the matching component of the combined modulus ledger M. Even requests use the A component of M, odd requests use the B component of M, and no request can consume E without one of these displayed ledger routes.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket combined modulus exactness`

Proof. Unpack the packet by Definition 414.1. The selector row, both schedules, the combined modulus ledger, and the interleaved seal are stored coordinates of the same finite BHist row.

For an even selector read, Definition 414.5 routes the request through A, S_A, and the first component of M. For an odd selector read, the same theorem routes the request through B, S_B, and the second component of M.

The Cont row records this order before the seal row E is exposed. The hsame, Pkg, and NameCert rows are attached to the same route, so the precision request has no separate path to a selected real limit, quotient equality, or ambient completeness. □

Theorem 414.10 (Regular Cauchy interleaving schedule window coverage)**.** *For an accepted packet $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$, the even and odd schedule windows exhaust the consumer reads of the interleaving. Every displayed consumer read is either an even read through S_A and the A side of M, or an odd read through S_B and the B side of M. There is no third schedule window and no unscheduled read of the interleaved seal.*

Proof. Inspect the source packet. Its only schedule coordinates are S_A and S_B, and its only side-selector coordinate is the alternating row σ.

The even case is the route through the first schedule. It reads the A source row, the S_A window, and the first component of the combined modulus ledger.

The odd case is the route through the second schedule. It reads the B source row, the S_B window, and the second component of the combined modulus ledger.

These two cases exhaust the selector row. Since the carrier has no coordinate for any further schedule, any consumer of E must pass through one of the two displayed windows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket schedule window coverage`

Theorem 414.11 (Regular Cauchy interleaving obligation closure package)**.** *The rows of an accepted packet* $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$ *assemble the obligation surface for* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The surface consists exactly of carrier admission, componentwise classifier transport, combined modulus exactness, schedule window coverage, and* $\mathsf{Real}^{\uparrow}$ *seal non-escape.*

Proof. Carrier admission is Definition 414.4. Componentwise classifier transport is Definition 414.2, with the parity partition supplied by Definition 414.5.

The combined modulus obligation is Definition 414.9. The coverage obligation is Definition 414.10. The seal boundary is Definition 414.8.

These rows are all read from the same finite packet through the displayed transport, continuation, package, and naming coordinates. Therefore the chapter supplies the obligation surface, while leaving quotient equality, selected host limits, ambient completeness, and bridge checking outside the claim. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket obligation closure package`

Theorem 414.12 (Regular Cauchy interleaving scoped kernel route)**.** *Every accepted packet from Definition 414.1 factors its carrier admission, classifier transport, selector parity, combined modulus, schedule-window coverage, and* $\mathsf{Real}^{\uparrow}$ *seal non-escape through exactly the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No part of the accepted carrier is routed through quotient equality of reals, selected host limits, or ambient completeness.*

Proof. Begin with the classifier of Definition 414.2. It reads only the two regular names, the two stream schedules, the alternating selector, the combined modulus ledger, the real seal rows, and the displayed transport and continuation coordinates.

The NameCert surface of Definition 414.3 supplies the carrier row, classifier row, transport row, exactness row, and seal-boundary row. The carrier admission theorem then repacks those coordinates as the finite BHist packet of the source definition.

Selector parity is exact by Definition 414.5. Combined modulus exactness is Definition 414.9, and schedule coverage is Definition 414.10. These three statements bind every consumer read to one of the two displayed schedules and to the matching component of the ledger.

Finally, Definition 414.8 keeps the $\mathsf{Real}^{\uparrow}$ seal downstream of the selected regular readback, selected schedule, combined modulus row, hsame transport, Cont route, package row, and naming row. Together with Definition 414.11, this exhausts the scoped kernel route and leaves no carrier coordinate for quotient equality, host limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket scoped kernel route`

Theorem 414.13 (Regular Cauchy interleaving public real seal export)**.** *The public* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *export of an accepted packet* $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$ *is exactly the finite* $\mathsf{Real}^{\uparrow}$ *seal boundary obtained from the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source names* A, B*, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules* S_A, S_B*, the alternating selector* σ*, the combined modulus ledger* M*, the source seals* E_A, E_B*, the interleaved seal row* E*, the displayed* hsame *transports* H*, the* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*. Every public* $\mathsf{Real}^{\uparrow}$ *consumer factors through this displayed packet and receives no quotient real equality, selected host limit, ambient completeness principle, or third schedule.*

Proof. Carrier admission is supplied by Definition 414.4. The admitted row is the finite BHist packet of Definition 414.1, so the public boundary has no carrier beyond the two source names, two schedules, selector, ledger, seals, transport rows, continuation rows, package row, and naming row.

Classifier transport is Definition 414.2. Selector parity exactness is Definition 414.5, combined modulus exactness is Definition 414.9, and schedule-window coverage is Definition 414.10. These rows force every consumer read to use either the A, S_A side with the first component of M or the B, S_B side with the second component of M.

The seal boundary is Definition 414.8. The obligation closure package of Definition 414.11 gathers the same rows, and Definition 414.12 binds them to the displayed kernel route.

Repack these rows as the public export packet. Since the repacked packet exposes only the displayed $\mathsf{Real}^{\uparrow}$ seal boundary, a public consumer obtains the named seal interface and none of quotient equality, a selected host limit, ambient completeness, or a third schedule. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket public real seal export`

Theorem 414.14 (Regular Cauchy interleaving bridge readback scope)**.** *Any bridge readback from the public* $\mathsf{Real}^{\uparrow}$ *seal export of an accepted* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *packet* $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H,$ *factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source names* A, B*, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules* S_A, S_B*, the alternating selector* σ*, the combined modulus ledger* M*, the source seal rows* E_A, E_B*, the interleaved seal row* E*, componentwise* hsame *transports* H*,* Cont *routes* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}$ *row* N*. The bridge readback has no route to a quotient real, selected host limit, ambient completeness principle, or third schedule beyond this displayed public seal surface.*

Proof. Carrier admission is Definition 414.4. It gives the finite packet rows that any bridge readback must start from: the two source names, their schedules, the selector, the modulus ledger, the three seal rows, transports, routes, provenance, and local naming row.

Selector parity exactness from Definition 414.5 determines which source side is read. Combined modulus exactness from Definition 414.9 supplies the matching ledger component, and schedule-window coverage from Definition 414.10 rules out a third schedule.

The seal boundary is controlled by Definition 414.8. The scoped kernel route of Definition 414.12 then binds the same rows to the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

Finally apply the public Real seal export, Definition 414.13. A bridge readback that begins from that public export can only read back the rows exported there, so it factors through $A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N$ and cannot access quotient equality, selected host limits, ambient completeness, or an unlisted schedule. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket bridge readback scope`

Theorem 414.15 (Regular Cauchy interleaving finite-tail consumer boundary)**.** *Every downstream consumer that reads the interleaved* Real$^{\uparrow}$ *seal of an accepted* RegularCauchyInterleaving$^{\uparrow}$ *packet through a finite-tail or diagonal-limit route must factor through one of the two displayed* StreamName$^{\uparrow}$ *schedules, the matching* RegSeqRat$^{\uparrow}$ *source row, the combined modulus ledger, and the local* NameCert *row before it can consume a* FiniteTailDiagonalSeal$^{\uparrow}$*-style witness. The route is the sibling surface of chapter 126, chapter 371, and chapter 559.*

Proof. Use selector parity exactness. An even read is routed through A, S_A and the first component of M, while an odd read is routed through B, S_B and the second component.

Apply combined modulus exactness and schedule-window coverage. Together they make the selected schedule, matching regular source row, and selected modulus component the only finite window route into the interleaved seal.

Use the Real seal non-escape theorem and the bridge readback scope. They keep every seal-facing consumer downstream of the displayed hsame transports, Cont routes, package row, and local NameCert row.

The finite-tail handoff therefore has no coordinate for quotient reals, host limits, ambient completeness, or a third schedule. Any FiniteTailDiagonalSeal$^{\uparrow}$-style witness consumed downstream must use the displayed finite-window route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket finite tail consumer boundary`

Theorem 414.16 (Regular Cauchy interleaving window-seal symmetry)**.** *For an accepted* RegularCauchyInterleaving *packet* $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$*, the two parity branches preserve the same public* Real$^{\uparrow}$ *seal surface. The even branch through* A, S_A *and the odd branch through* B, S_B *are distinguished only by selector parity and the selected component of the combined modulus ledger; after selector transport, schedule-window coverage, and real-seal non-escape, both branches expose the same interleaved seal row* E *and no additional public seal coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket window seal symmetry`

Proof. Use selector parity exactness to split any consumer read into the even branch and the odd branch. The even branch selects the A, S_A side and the first ledger component, while the odd branch selects the B, S_B side and the second ledger component.

Apply classifier transport and combined modulus exactness on each branch. These rows determine the selected source row and the matching modulus component before any public seal read occurs.

Schedule-window coverage rules out a third schedule and forces both parity branches through the two displayed StreamName$^{\uparrow}$ windows. The bridge readback scope keeps the same hsame, Cont, Pkg, and NameCert rows attached to the selected read.

Finally apply real-seal non-escape and the public Real seal export. Both branches terminate at the same interleaved seal row E, so the branch distinction is internal to the finite-window route and does not create a second public Real$^{\uparrow}$ seal surface. □

Theorem 414.17 (Regular Cauchy interleaving finite-tail budget route)**.** *Every finite-tail diagonal consumer of the interleaved* Real$^{\uparrow}$ *seal of an accepted packet* $I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$

factors through the two displayed StreamName$^{\uparrow}$ *schedules* S_A, S_B*, the matching* RegSeqRat$^{\uparrow}$ *source rows* A, B*, the combined modulus ledger* M*, the downstream* FiniteTailDiagonalSeal$^{\uparrow}$ *witness surface, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No finite-tail budget read bypasses this route through quotient equality, selected host limits, ambient completeness, or an unlisted schedule.*

Proof. Begin with the finite-tail consumer boundary. It already places any diagonal-limit or finite-tail consumer downstream of a selected schedule, a matching regular source row, the combined modulus ledger, and the local naming row.

Use selector parity exactness and schedule-window coverage to identify the only two possible schedule windows. The selected window is S_A for an even read and S_B for an odd read, with no further schedule coordinate in the packet.

Combined modulus exactness supplies the budget component attached to that selected window. Carrier admission supplies the finite BHist packet, and the scoped kernel route keeps the selected read attached to the displayed hsame, Cont, Pkg, and NameCert rows.

The sibling chapters chapter 126, chapter 371, and chapter 559 provide the schedule, regular-source, and finite-tail witness surfaces named by the boundary. Therefore every finite-tail budget read factors through the displayed route and has no bypass through the excluded public objects. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket finite tail budget route`

Theorem 414.18 (Regular Cauchy interleaving public bridge readiness)**.** *The obligation closure package, scoped kernel route, public real seal export, bridge readback scope, finite-tail consumer boundary, window-seal symmetry, and finite-tail budget route assemble the public bridge surface for* RegularCauchyInterleaving$^{\uparrow}$*. A bridge-facing consumer of an accepted packet can read the two* RegSeqRat$^{\uparrow}$ *source names, the two* StreamName$^{\uparrow}$ *schedules, selector parity, the combined modulus ledger, source and interleaved* Real$^{\uparrow}$ *seal rows,* BHist *support,* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *row, and the downstream finite-tail witness route; it cannot read quotient real equality, selected host limits, ambient completeness, or a third schedule.*

Proof. The obligation closure package gives the carrier, classifier, combined modulus, schedule coverage, and seal-boundary rows. The scoped kernel route binds these rows to the displayed RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert coordinates.

The public Real seal export exposes the seal row visible to outside consumers, and the bridge readback scope describes exactly which packet rows a bridge-facing readback can inspect. The finite-tail consumer boundary then places diagonal consumers downstream of the same selected schedule and budget route.

Window-seal symmetry shows that the even and odd branches preserve one public seal surface rather than two branch-specific exports. The finite-tail budget route supplies the budget and witness path needed by downstream finite-tail consumers.

Combining these rows yields the public bridge surface for RegularCauchyInterleaving$^{\uparrow}$. Since every component in the assembled surface is one of the displayed packet rows or one of the named sibling surfaces, the route has no coordinate for quotient equality of reals, selected host limits, ambient completeness, or any schedule beyond S_A and S_B. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket public bridge readiness`

Theorem 414.19 (Regular Cauchy interleaving RegSeqRat Real consumer factorization)**.** *Let*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, *or terminal* $\mathsf{Real}^{\uparrow}$ *consumer of the interleaved seal factors through the selected parity branch, then through the selected* $\mathsf{StreamName}^{\uparrow}$ *schedule, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the corresponding component of* M*, and finally through* E, H, C, P, N*. The factorization exports no quotient stream, selected host limit, host real equality, ambient completeness principle, or third schedule.*

Proof. Use selector parity exactness to split the consumer read. The even branch is routed through A and S_A, while the odd branch is routed through B and S_B.

Apply combined modulus exactness on the selected branch. The even route reads the first component of M, the odd route reads the second component of M, and the selected ledger component is fixed before any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, or terminal $\mathsf{Real}^{\uparrow}$ consumer can inspect the interleaved seal.

Schedule-window coverage rules out an unscheduled route. Thus the selected $\mathsf{StreamName}^{\uparrow}$ window and the matching $\mathsf{RegSeqRat}^{\uparrow}$ source row are the only finite-name entry points for the consumer.

Finally apply real-seal non-escape and the public Real seal export. These rows keep the consumer downstream of the interleaved seal E, the displayed hsame transports H, the Cont routes C, package provenance P, and the local naming row N, while leaving quotient streams, selected host limits, host real equality, ambient completeness, and any third schedule outside the exported route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket regseqrat real consumer factorization`

Theorem 414.20 (Regular Cauchy interleaving even–odd subsequence seal)**.** *Let*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *packet whose even reads use the source and schedule rows* A, S_A *and whose odd reads use* B, S_B*. An even-subsequence consumer recovers the* A*-side* $\mathsf{Real}^{\uparrow}$ *seal route, and an odd-subsequence consumer recovers the* B*-side* $\mathsf{Real}^{\uparrow}$ *seal route. Both recoveries pass through the same selector parity row, combined modulus row, schedule-window coverage row, and Real seal non-escape row, so neither consumer exports quotient stream equality, host real equality, an ambient completeness principle, or a third schedule.*

Proof. Selector parity exactness separates the two subsequence requests. Apply Definition 414.5: even reads are routed through A, S_A, while odd reads are routed through B, S_B.

Combined modulus exactness then fixes the matching ledger component for the selected side. Use Definition 414.9 so the even consumer reads the first component of M and the odd consumer reads the second component.

Schedule-window coverage rules out bypass routes. By Definition 414.10, every subsequence read must remain inside the displayed S_A or S_B window and cannot switch to a third schedule.

Finally apply Definition 414.8. The selected subsequence seal remains downstream of the selected source row, selected schedule, selected modulus component, displayed hsame transport, displayed Cont route, package provenance, and local naming row. Thus the two parity consumers recover their side-specific Real seal routes and nothing outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket even odd subsequence seal`

Theorem 414.21 (Regular Cauchy interleaving completion selector nonescape)**.** *Let*

$$I = (A, B, S_A, S_B, \sigma, M, E_A, E_B, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *packet. Every completion-facing consumer of* I *must read the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* A, B*, the parity selector* σ*, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules* S_A, S_B*, the merged window schedule governed by* M*, the dyadic tail modulus selected from* M*, and the visible* $\mathsf{Real}^{\uparrow}$ *seal route through* E_A, E_B, E, H, C, P, N *before it can consume any completion selector. The packet exports no quotient stream, hidden selected limit, host real equality, ambient completeness principle, or third schedule.*

Proof. Start with the classifier route. The selector parity exactness theorem separates every read into the even branch through A, S_A and the odd branch through B, S_B.

Apply combined modulus exactness and the finite-tail budget route. These rows place the merged window schedule and the selected dyadic tail budget inside M before any diagonal or completion-facing consumer can inspect the interleaved seal.

Use schedule-window coverage and the even–odd subsequence seal. The schedules S_A, S_B are the only windows available, and each subsequence recovery remains downstream of the selected source row, selected schedule, selected modulus component, and Real seal non-escape route.

Finally read H, C, P, N. These rows transport, replay, source, and name the same finite packet, but they add no selector coordinate. Therefore every completion selector remains inside the displayed interleaving route and cannot escape to quotient streams, hidden selected limits, host equality, ambient completeness, or a third schedule. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleavingUp.RegularCauchyInterleavingPacket completion selector nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ regular names, their $\mathsf{StreamName}^{\uparrow}$ schedules, an alternating unary selector, a combined modulus ledger, source $\mathsf{Real}^{\uparrow}$ readbacks, an interleaved $\mathsf{Real}^{\uparrow}$ seal row, displayed hsame and Cont rows, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 414.13 gives the public $\mathsf{Real}^{\uparrow}$ seal export, and Definition 414.18 records the checked public bridge readiness target over the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleaving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient equality of reals, selected host limits, ambient completeness, diagonal-completion uniqueness, or bridge checking.

Upgrade path: The remaining work is Lean-side formal bridge verification for the displayed public bridge surface.

Chapter 415

A Concrete Naming Certificate for $\mathsf{CauchyRate}^{\uparrow}$

The $\mathsf{CauchyRate}^{\uparrow}$ interface names the effective-rate certificate consumed between regular Cauchy data and real-completion readers. In the constructive-analysis reading of Bishop and Bridges, *Constructive Analysis*, a usable real name supplies an explicit precision request, a modulus schedule, a tolerance row, and a regular rational source before any completion-facing consumer may read a real seal. The packet below records exactly that finite route through $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$.

The same packet is the effective-rate dependency edge from chapter 126 and chapter 301 into the completion-facing chapter 38 seal boundary: a precision request selects a modulus schedule, the schedule opens a finite $\mathsf{StreamName}^{\uparrow}$ window, the window licenses the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the resulting regular row is admitted through the $\mathsf{Real}^{\uparrow}$ seal gate.

The rate-facing carrier is also the consumer surface of chapter 292. Its tolerance and readback route consumes the dyadic tail-bound ledger of chapter 379 and the regular-rational source of chapter 301 before the packet reaches the $\mathsf{RealCauchyCompletion}^{\uparrow}$ boundary.

Definition 415.1 (CauchyRate BHist source)**.** *A* $\mathsf{CauchyRate}^{\uparrow}$ *source packet is a finite* BHist *row*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N)$$

with the following visible coordinates:

1. *p is a unary precision row naming the requested effective rate;*
2. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row that assigns the finite inspection window for p;*
3. *τ is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row used to justify the scheduled tail comparison;*
4. *F is the* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *source row whose inspected family entries are read through the schedule μ;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row obtained from the same scheduled family window and tolerance row;*
6. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row reached only through Q, F, τ, and μ;*
7. *H records the componentwise* hsame *transports for precision, schedule, tolerance, family source, regular-sequence source, and completion-consumer rows;*

8. *C records the displayed* Cont *routes from the consumer request to the schedule, tolerance, source, regular readback, and completion boundary;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *row.*

The displayed transports and routes are the rows

$$\mathsf{hsame}(p,p'),\quad \mathsf{hsame}(\mu,\mu'),\quad \mathsf{hsame}(\tau,\tau'),\quad \mathsf{hsame}(F,F'),\quad \mathsf{hsame}(Q,Q'),\quad \mathsf{hsame}(E,E'),$$
$$\mathsf{Cont}(p,\mu),\quad \mathsf{Cont}(\mu,\tau),\quad \mathsf{Cont}((\mu,\tau),F),\quad \mathsf{Cont}((F,\tau),Q),\quad \mathsf{Cont}((Q,F,\tau,\mu),E).$$

The classifier compares accepted packets only by these displayed rows, their Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *row.*

Theorem 415.2 (CauchyRate NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *surface for the packet of Definition 415.1 consists of five finite obligations:*

1. *carrier exposure: the precision, schedule, tolerance, regular-family source, regular rational readback, completion consumer, transport, route, package, and local certificate rows are all visible coordinates of K;*
2. *classifier exactness: accepted packets are compared only by componentwise* hsame *transport and displayed* Cont *reads of those coordinates;*
3. *regular-family handoff: the row Q is read from the scheduled* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *source F using the tolerance row τ and no separate stream;*
4. *completion-consumer boundary: the row E is reachable only after the* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have been displayed;*
5. *ledger exhaustion: every rate-facing dependency is named by P and N, with no row for real completeness, quotient stream equality, selected limits, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier namecert obligations`

Proof. Project the packet K of Definition 415.1. Its carrier coordinates list precisely the rows used by the certificate, so carrier exposure is immediate from the finite BHist shape.

Classifier exactness follows by replaying the stored transports and continuation routes. The classifier has no coordinate outside p,μ,τ,F,Q,E,H,C,P,N, hence a transported accepted packet is compared only through those rows.

The regular-family handoff is the projection to Q. The route $\mathsf{Cont}((F,\tau),Q)$ keeps the scheduled family window and the dyadic tail bound together, so the readback cannot be replaced by an unrecorded rational stream.

The completion-consumer boundary is the projection to E. Since the only displayed route into E is $\mathsf{Cont}((Q,F,\tau,\mu),E)$, the consumer sees the modulus schedule, tolerance, family source, and regular rational readback before it sees a completion-facing row.

Finally, P and N account for the package provenance and the local naming surface. They add no coordinate for RealUp completeness, quotient stream equality, selected limits, or host real equality, so the ledger is exhausted by the displayed finite packet. □

Theorem 415.3 (CauchyRate regular-family handoff)**.** *For every accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet*

$$K=(p,\mu,\tau,F,Q,E,H,C,P,N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q *is obtained from the same* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row* μ*, the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row* τ*, and the same* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *source row* F*. Any consumer of* Q *through the displayed* Cont *routes must retain those three rows as the rate certificate for the inspected finite window.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier regular family handoff`

Proof. Read the schedule, tolerance, family, and regular-sequence coordinates from K. The row μ selects the finite inspection window, the row τ supplies the displayed tail budget, and the row F supplies the regular Cauchy family entries inspected on that window.

The continuation route $\mathsf{Cont}((F,\tau),Q)$ repacks exactly those rows as the $\mathsf{RegSeqRat}^{\uparrow}$ source row. Because τ and F are stored in the same packet as μ, the handoff cannot drop the rate schedule or substitute a separate tolerance ledger.

Any consumer of Q must use the displayed routes in C and the provenance rows P and N. These rows certify the same finite packet and do not introduce quotient stream equality, selected limits, host sequence equality, or a real-completeness principle. □

Theorem 415.4 (CauchyRate real-completion consumer boundary)**.** *For an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet*

$$K = (p,\mu,\tau,F,Q,E,H,C,P,N),$$

every read of the $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row* E *factors through the scheduled modulus row* μ*, the dyadic tail-bound row* τ*, the regular-family source row* F*, and the regular rational readback row* Q*. The boundary exports an effective completion-facing consumer route, not RealUp completeness, quotient stream equality, a selected limit, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier real completion consumer boundary`

Proof. Project the completion row from the source packet. The only route displayed for E is $\mathsf{Cont}((Q,F,\tau,\mu),E)$, so the consumer boundary is downstream of the schedule, tolerance, family source, and regular rational readback.

The handoff theorem Definition 415.3 identifies Q as the regular rational row obtained from F with the same μ and τ. Thus reading E necessarily preserves the effective-rate data that produced the readback.

The packet contains no coordinate for a quotient of streams, a selected limiting object, a host real equality test, or an ambient completeness witness. The completion-facing row is therefore a consumer boundary over the displayed effective certificate only. □

Theorem 415.5 (CauchyRate ledger exactness)**.** *Let* $K = (p,\mu,\tau,F,Q,E,H,C,P,N)$ *be an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *source packet. The ledger named by* P *and* N *is exact over the rows* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicTailBound}^{\uparrow}$*,* $\mathsf{RegularCauchyFamily}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*: every accepted rate read is accounted for by one of those displayed coordinates, and no hidden carrier may be used as an additional premise.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier ledger exactness`

Proof. Start from the finite carrier of Definition 415.1. The rows p,μ,τ,F,Q,E,H,C,P,N are the full list of data available to the local classifier.

The obligation theorem Definition 415.2 identifies the carrier, classifier, handoff, boundary, and ledger rows. The handoff theorem Definition 415.3 accounts for reads of Q, and Definition 415.4 accounts for reads of E.

It remains only to inspect provenance. The row P names the dependency package and N names the local certificate, both over the displayed finite packet. Since no other coordinate is present, no hidden carrier can serve as an additional premise for the effective-rate read. □

Theorem 415.6 (CauchyRate transport stability)**.** *Let $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet. Any componentwise* hsame *transport recorded in H, together with the displayed* Cont *routes recorded in C, preserves the* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *classifier on the transported precision, schedule, tolerance, regular-family source, regular rational readback, completion consumer, package, and local certificate rows.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier transport stability`

Proof. Project the classifier from Definition 415.1. It compares only the precision, schedule, tolerance, family, readback, consumer, transport, route, package, and local naming coordinates.

Transport each visible coordinate by the corresponding hsame row in H. Then replay the routes in C: first from the precision request to the schedule, then to the tolerance, then to the family source, then to the regular readback, and finally to the completion consumer boundary.

The ledger exactness theorem Definition 415.5 rules out any additional premise during this transport. Hence the transported packet is accepted by the same classifier surface and still withholds RealUp completeness, quotient stream equality, selected limits, and host real equality. □

Theorem 415.7 (CauchyRate scheduled window totality)**.** *For every accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N),$$

each precision request admitted by the packet is answered first by the displayed $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row μ. The consumer route must expose the finite window selected by μ before it may read the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row τ, the* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *source row F, or the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier scheduled window totality`

Proof. Project the carrier of Definition 415.1. The precision row p and the schedule row μ are stored as visible coordinates, and the route inventory contains $\mathsf{Cont}(p, \mu)$ as the first consumer step.

The downstream rows are reached only along the displayed continuation chain. The tolerance row is exposed by $\mathsf{Cont}(\mu, \tau)$, the regular-family source by $\mathsf{Cont}((\mu, \tau), F)$, and the regular rational readback by $\mathsf{Cont}((F, \tau), Q)$.

Apply Definition 415.12 to identify this chain as the only effective-rate consumer route, and apply Definition 415.6 to keep the same route under componentwise hsame transport. Hence every accepted precision request receives its scheduled finite window before any of the later rows can be read. □

Theorem 415.8 (CauchyRate finite-tail induction)**.** *Let $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet. Finite induction over the displayed unary precision row and the scheduled window preserves the route*

$$p \longmapsto \mu \longmapsto \tau \longmapsto F \longmapsto Q \longmapsto E$$

for every admitted tail request. At each finite stage the consumer reads the same $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row, the same* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *source, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the same completion-facing row stored in K; it never replaces them by quotient streams, selected limits, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The base request is the displayed precision row p in Definition 415.1. By Definition 415.7, the request first reaches the schedule row μ and exposes its finite window before any downstream row is visible.

For the induction step, enlarge only the displayed finite tail request admitted by the same schedule. The carrier still supplies the rows $p, \mu, \tau, F, Q, E, H, C, P, N$, and the continuation entries keep the new finite request on the route through μ and τ before the family row F is read.

Apply Definition 415.3. The readback row Q is obtained from the scheduled family and the dyadic tail-bound row already present in the packet, so the step cannot introduce a new rational stream or selected limiting object.

Finally use Definition 415.5. The ledger rows P and N exhaust the dependency surface for every finite tail stage, and the route into E remains the displayed completion-facing continuation. Thus finite induction preserves the route and refuses quotient streams, selected limits, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 415.9 (CauchyRate tolerance readback exactness)**.** *For every accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet* $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$*, the displayed tolerance row* τ *and the displayed regular-family row* F *determine the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *through the packet route* $\mathsf{Cont}((F, \tau), Q)$*. No hidden rational stream, quotient representative, host sequence equality, or selected limiting object participates in the readback.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier tolerance readback exactness`

Proof. The readback coordinate Q is part of the finite packet in Definition 415.1. Its incoming displayed route is exactly $\mathsf{Cont}((F, \tau), Q)$.

Apply Definition 415.3. It states that Q is obtained from the same $\mathsf{CauchyModulus}^{\uparrow}$ schedule, the same $\mathsf{DyadicTailBound}^{\uparrow}$ tolerance row, and the same $\mathsf{RegularCauchyFamily}^{\uparrow}$ source row carried by the packet.

Finally apply Definition 415.5. The package and local naming rows exhaust the available dependency surface, so there is no extra coordinate from which a hidden rational stream or host-level equality could determine Q. □

Theorem 415.10 (CauchyRate scoped consumer exhaustion)**.** *The scoped consumer surface of an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet is exhausted by its displayed carrier rows, regular-family handoff, completion-consumer boundary, ledger exactness, and transport stability over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Every consumer-facing read of* Q *or* E *is therefore a read of the same finite packet, not a use of an ambient completion principle or an unrecorded stream equality.*

Proof. Start with the carrier definition. The packet exposes only $p, \mu, \tau, F, Q, E, H, C, P, N$, so the consumer surface has no raw coordinate outside the displayed BHist row.

Use Definition 415.3 and Definition 415.4 for the two downstream reads. The first places Q behind the scheduled family and tolerance rows, while the second places E behind Q, F, τ, μ.

The ledger and transport steps close the surface. By Definition 415.5, P and N account for every accepted dependency over Pkg and NameCert; by Definition 415.6, the same classifier is preserved under the stored hsame and Cont rows. Thus the scoped consumer has exhausted exactly the displayed finite packet. □

Theorem 415.11 (CauchyRate BHist source scope)**.** *For every accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N),$$

the BHist *source scope is exactly the displayed coordinate list* $p, \mu, \tau, F, Q, E, H, C, P, N$*. No RealUp completeness principle, quotient stream equality, selected limit, host real equality, or unrecorded completion carrier is a dependency of the source packet.*

Proof. Read the carrier definition in Definition 415.1. It exposes precisely the ten rows $p, \mu, \tau, F, Q, E, H, C, P, N$, and no other coordinate is available to the packet classifier.

Apply Definition 415.2 to identify the visible obligations. Apply Definition 415.3 and Definition 415.4 to locate the only downstream reads of Q and E inside the same displayed source packet.

It remains to rule out hidden source dependencies. The ledger theorem Definition 415.5 exhausts the available package rows, and Definition 415.6 preserves the same classifier under componentwise transport. Hence the source scope is exactly the displayed BHist row list. □

Theorem 415.12 (CauchyRate effective route scope)**.** *Every effective-rate consumer route for an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet factors through the displayed continuation rows*

$$\mathsf{Cont}(p, \mu), \quad \mathsf{Cont}(\mu, \tau), \quad \mathsf{Cont}((\mu, \tau), F), \quad \mathsf{Cont}((F, \tau), Q), \quad \mathsf{Cont}((Q, F, \tau, \mu), E).$$

Thus a consumer may inspect the completion-facing row only after passing through the precision request, modulus schedule, dyadic tail bound, regular Cauchy family source, and regular rational readback recorded in the same packet.

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier effective route scope`

Proof. Start with the carrier in Definition 415.1. Its route row C displays the five continuation entries from p to μ, from μ to τ, from (μ, τ) to F, from (F, τ) to Q, and from (Q, F, τ, μ) to E.

Apply Definition 415.2 for classifier exactness and Definition 415.3 for the readback row Q. Then apply Definition 415.4 to place E downstream of Q, F, τ, μ.

The route cannot fork through an extra consumer row. By Definition 415.5, the ledger has no hidden carrier for an alternate read, and by Definition 415.6, componentwise transport preserves the same route inventory. Therefore every effective-rate consumer route factors through the displayed continuation chain. □

Theorem 415.13 (CauchyRate ledger scope)**.** *For an accepted* $\mathsf{CauchyRate}^{\uparrow}$ *packet* $K = (p, \mu, \tau, F, Q, E, H, C, P, N$ *the* Pkg *row* P *and local* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *row* N *exhaust the dependency ledger over* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicTailBound}^{\uparrow}$*,* $\mathsf{RegularCauchyFamily}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. No dependency outside that list may be used to justify an effective-rate read.*

Proof. Project the packet carrier of Definition 415.1. The package row P names the external dependency surface, while N names the local certificate surface for exactly the same finite packet.

Use Definition 415.2 to recover carrier exposure, classifier exactness, handoff, boundary, and ledger exhaustion. The handoff theorem Definition 415.3 accounts for $\mathsf{RegSeqRat}^{\uparrow}$, and Definition 415.4 accounts for $\mathsf{RealCauchyCompletion}^{\uparrow}$.

Finally, apply Definition 415.5 to close the package inventory and Definition 415.6 to keep it stable under hsame transport and displayed Cont routes. The only remaining rows are P and N, so the stated dependency list is exhaustive. □

Definition 415.14 (CauchyRate scoped obligation package)**.** *The scoped obligation package for* $\mathsf{CauchyRate}^{\uparrow}$ *is the accepted finite packet*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N)$$

of Definition 415.1, read as a BHist *carrier over exactly the displayed dependencies*

$$\mathsf{CauchyModulus}^{\uparrow},\quad \mathsf{DyadicTailBound}^{\uparrow},\quad \mathsf{RegularCauchyFamily}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{RealCauchyCompletion}^{\uparrow}.$$

Its scope consists of the precision row p, the modulus schedule μ, the tail-bound row τ, the regular Cauchy family row F, the regular rational readback Q, the completion consumer row E, the componentwise hsame *transports in H, the* Cont *routes in C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyRate}^{\uparrow}}$ *row N.*

The package admits a consumer only when the consumer route factors through these rows in the displayed order. It has no coordinate for RealUp completeness, host real equality, quotient stream equality, selected limits, or an unscoped completion oracle.

Theorem 415.15 (CauchyRate window monotonicity)**.** *Let $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ be a scoped obligation package for* $\mathsf{CauchyRate}^{\uparrow}$*. If a consumer refines a precision request from p to a stricter displayed request p', then the accepted schedule window for p' is obtained by transporting p' through the same* $\mathsf{CauchyModulus}^{\uparrow}$ *row μ and the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *row τ. The transported classifier may enlarge the inspected finite window, but it cannot replace μ, bypass τ, or read F, Q, E from a different package.*

Proof. Project the scoped carrier of Definition 415.14. The only row that turns a precision request into a finite inspection window is μ, and the only tail budget attached to that window is τ.

Transport the refined request by the precision component of H. Replaying the route $\mathsf{Cont}(p, \mu)$ for the transported request keeps the schedule inside the same $\mathsf{CauchyModulus}^{\uparrow}$ row, and replaying $\mathsf{Cont}(\mu, \tau)$ keeps the dyadic tail-bound ledger attached to the scheduled window.

The classifier described in Definition 415.1 compares packets only through displayed hsame and Cont rows. Thus a stricter request can ask the same schedule row for a larger visible window, but no classifier branch can substitute a different modulus, omit the tolerance row, or fetch the downstream rows from outside K. □

Theorem 415.16 (CauchyRate tail-bound routing)**.** *For every scoped* $\mathsf{CauchyRate}^{\uparrow}$ *package $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$, every read of the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q must first pass through the scheduled* $\mathsf{CauchyModulus}^{\uparrow}$ *row μ, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *row τ, and the* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *row F. In particular, no Q-consumer can inspect a regular rational readback before the tail-bound route through τ has been displayed.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier tail bound routing`

Proof. Read the route inventory from the scoped package. The route into Q is $\mathsf{Cont}((F, \tau), Q)$, and the row F is itself reached only after the schedule and tolerance rows have been exposed through the displayed route chain.

Apply the regular-family handoff theorem Definition 415.3. It identifies Q as the readback produced from the same scheduled family row and the same dyadic tail-bound row, so the rational readback is not an independent source row.

Finally inspect P and N. Their provenance names the displayed dependency package and the local certificate only; neither row adds an alternate path to Q. Hence a consumer of Q must see μ, τ, and F before it obtains the regular rational readback. □

Theorem 415.17 (CauchyRate real-seal consumption)**.** *Let $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ be a scoped* $\mathsf{CauchyRate}^{\uparrow}$ *package. Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer read of E factors through Q, F, τ, μ in that order before any completion-facing row is consumed. The factorization is a scoped*

completion-consumer route, not a proof of RealUp completeness, quotient equality of streams, a selected limit, or host real equality.

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier real seal consumption`

Proof. Project the completion-facing coordinate from K. The only displayed route into E is $\mathsf{Cont}((Q, F, \tau, \mu), E)$, so the consumer boundary begins with the regular rational readback and carries the family, tail-bound, and schedule rows with it.

Use Definition 415.16 for the upstream read of Q. That theorem forces the readback itself through μ, τ, and F before Q is available to any later consumer.

Combining the two routes gives the stated factorization through Q, F, τ, μ before E. The scoped obligation package has no coordinate for the excluded RealUp, quotient, selected-limit, or host-equality objects, so the completion-facing read consumes only the displayed effective-rate certificate. □

Theorem 415.18 (CauchyRate public certificate)**.** *Let $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ be a scoped* $\mathsf{CauchyRate}^{\uparrow}$ *package. The* BHist *source,* NameCert *obligations, regular-family handoff, real-completion consumer boundary, ledger exactness, transport stability, scheduled-window totality, tolerance readback exactness, scoped consumer exhaustion, source scope, effective-route scope, ledger scope, window monotonicity, tail-bound routing, and real-seal consumption assemble a single exported* $\mathsf{CauchyRate}^{\uparrow}$ *public certificate. Its public consumers are exactly the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readers that factor through the displayed effective-rate rows, and it exports no RealUp completeness, quotient stream equality, selected limit, or host equality.*

Lean4 ✓ `BEDC.Derived.CauchyRateUp.CauchyRateCarrier public certificate`

Proof. Project the scoped package $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$ from Definition 415.14. The BHist source of Definition 415.1 supplies the carrier rows, and Definition 415.2 exposes the local NameCert obligations attached to those rows.

The effective-rate spine is the one exposed by Definition 415.7, Definition 415.3, Definition 415.9, and Definition 415.15. These results identify the schedule, tolerance row, regular family, and readback without replacing the rows stored in K.

The route and ledger boundary is fixed by Definition 415.5, Definition 415.6, Definition 415.11, and Definition 415.12. Together with Definition 415.13, these results keep every transported consumer inside the same source, route, and ledger surface.

The downstream boundary is fixed by Definition 415.16, Definition 415.17, and Definition 415.10. A $\mathsf{RegSeqRat}^{\uparrow}$ reader consumes Q only after the scheduled tail-bound route, a $\mathsf{RealCauchyCompletion}^{\uparrow}$ reader consumes E only through Q, F, τ, μ, and a $\mathsf{Real}^{\uparrow}$ reader sees only that same effective-rate certificate.

It remains to inspect the excluded objects. The package has no coordinate for RealUp completeness, quotient stream equality, selected limits, or host equality, and the scoped consumer-exhaustion clause leaves no route that escapes $p, \mu, \tau, F, Q, E, H, C, P, N$. The assembled object is therefore a public $\mathsf{CauchyRate}^{\uparrow}$ certificate over the displayed effective-rate rows and nothing stronger. □

Theorem 415.19 (CauchyRate bridge source exactness)**.** *The standard effective Cauchy-rate bridge source is exactly the packet*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N)$$

of Definition 415.14. Its source rows are the precision request, schedule, tail-bound, regular family, regular-rational readback, completion consumer, displayed hsame *ledger, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit, or host equality.*

Proof. Project the scoped packet K from Definition 415.14. Its coordinates are precisely $p, \mu, \tau, F, Q, E, H, C, P, N$, so any bridge source read begins with the same precision, schedule, tail-bound, regular-family, readback, completion-consumer, ledger, route, package, and certificate rows.

Attach the effective-rate routes. Scheduled-window totality fixes the precision schedule, tail-bound routing forces the readback through μ, τ, F, Q, and real-seal consumption routes the completion-facing consumer through Q, F, τ, μ before E is read.

Finally use ledger exactness and the public certificate. The ledger theorem keeps the exposed rows equal to the displayed hsame, Cont, Pkg, and NameCert surface, while Definition 415.18 exports exactly that surface as the public $\mathsf{CauchyRate}^{\uparrow}$ object. Repacking these rows gives the standard effective Cauchy-rate bridge source and leaves no coordinate for the excluded RealUp completeness, quotient, selected-limit, or host-equality objects. □

Theorem 415.20 (CauchyRate standard bridge source packet)**.** *For the public* $\mathsf{CauchyRate}^{\uparrow}$ *certificate, the bridge source is exactly the finite packet*

$$K = (p, \mu, \tau, F, Q, E, H, C, P, N)$$

of Definition 415.1. Its rows are read over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *with no additional bridge-source coordinate.*

Proof. Project the public certificate through Definition 415.1. The displayed source has precisely the rows $p, \mu, \tau, F, Q, E, H, C, P, N$, and the local classifier compares only those rows together with their package and naming provenance.

Use Definition 415.18 to export the same packet as a public object. Its proof already assembles the source row, the NameCert obligations, the route scope, the ledger scope, the tail-bound route, and the completion-facing consumer boundary.

It remains to identify the bridge source. Since Definition 415.11, Definition 415.12, and Definition 415.13 exhaust the available rows, a standard bridge reading can cite exactly the displayed packet and no further coordinate. □

Theorem 415.21 (CauchyRate bridge non-escape boundary)**.** *Every bridge read from the public* $\mathsf{CauchyRate}^{\uparrow}$ *certificate is bounded by the displayed effective-rate packet* $K = (p, \mu, \tau, F, Q, E, H, C, P, N)$*. In particular, the bridge cannot read RealUp completeness, quotient stream equality, selected limits, host real equality, or any completion oracle not routed through* Q, F, τ, μ *and the stored* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row* E*.*

Proof. Start from the source packet fixed by Definition 415.20. The only completion-facing coordinate in that packet is E, and Definition 415.17 forces every read of E through Q, F, τ, μ before the completion-facing row is consumed.

The upstream readback is also bounded. By Definition 415.16, every read of Q passes first through the scheduled modulus row, the dyadic tail-bound row, and the regular-family row stored in the same packet.

Finally inspect the public ledger. The public certificate of Definition 415.18 and the scoped consumer exhaustion theorem rule out RealUp completeness, quotient stream equality, selected

limits, host real equality, and any unscoped completion oracle as exported rows. Therefore a bridge read cannot escape the finite effective-rate boundary. □

Theorem 415.22 (CauchyRate standard bridge obligation package)**.** *The CauchyRate standard bridge obligation package is assembled from the source packet of Definition 415.20, the non-escape boundary of Definition 415.21, the public certificate of Definition 415.18, scheduled-window totality, tail-bound routing, and real-seal consumption. The package presents the standard effective-rate reading as a consumer of* $p, \mu, \tau, F, Q, E, H, C, P, N$ *and nothing beyond that public surface.*

Proof. First build the bridge source. Apply Definition 415.20 to recover exactly the displayed BHist packet over the named dependency rows.

Next attach the effective-rate obligations. The schedule is supplied by Definition 415.7, the rational readback route is supplied by Definition 415.16, and the completion-facing consumption route is supplied by Definition 415.17.

Now repack the public surface. The public certificate in Definition 415.18 exports the same rows as the consumer-facing object, while Definition 415.21 excludes RealUp completeness, quotient stream equality, selected limits, host real equality, and unscoped completion reads.

These ingredients are exactly the bridge obligations: a fixed source packet, a scheduled finite window, a routed tail-bound readback, a completion consumer route, a public certificate, and a non-escape boundary. Hence the standard bridge obligation package is determined by $p, \mu, \tau, F, Q, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRate}^\uparrow}$.

Constructive story: A $\mathsf{CauchyRate}^\uparrow$ packet is generated from a unary precision row, a $\mathsf{CauchyModulus}^\uparrow$ schedule, a $\mathsf{DyadicTailBound}^\uparrow$ tolerance row, a $\mathsf{RegularCauchyFamily}^\uparrow$ source, a $\mathsf{RegSeqRat}^\uparrow$ readback, a $\mathsf{RealCauchyCompletion}^\uparrow$ consumer row, displayed hsame and Cont rows, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRate}^\uparrow}$, maturePackageClosed)

Scope closed: The public effective-rate route fixes exactly the $p, \mu, \tau, F, Q, E, H, C, P, N$ packet over $\mathsf{CauchyModulus}^\uparrow$, $\mathsf{DyadicTailBound}^\uparrow$, $\mathsf{RegularCauchyFamily}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{RealCauchyCompletion}^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRate}^\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CauchyRateUp.CauchyRateCarrier\_real\_completion\_consumer\_boundary`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, quotient equality of regular streams, selected limits, host real equality, or bridge checking.

Upgrade path: No further paper-side theory grade remains for the effective-rate bridge, standard source packet, and real-completion consumer boundary.

Chapter 416

A Concrete Naming Certificate for $\mathsf{RegularCauchyInterleave}^{\uparrow}$

The $\mathsf{RegularCauchyInterleave}^{\uparrow}$ interface names the finite merge schedule used when two regular Cauchy names are exposed through one visible stream of observations. The construction is not a quotient of completed real values and does not select a host limit; it records the rows that make every merged window, dyadic observation, modulus request, and real-name handoff inspectable.

The interleaved window table consumes the finite-window surface of chapter 126. Its dyadic observation ledger consumes the row-level rational core supplied by chapter 291, and its real-name handoff is read against the boundary discipline of chapter 38. Thus the packet is a horizontal dependency node in the completion lattice rather than a kernel-only island.

Definition 416.1 (Regular Cauchy interleave carrier)**.** *A* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, J, S, T, W, D, M, H, C, P, N)$$

with the following visible rows:

1. *I and J are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *member schedules whose regular Cauchy observations may be read;*
2. *S is the unary merge schedule naming, for each retained slot, whether the slot reads the I side or the J side;*
3. *T is the interleaved* $\mathsf{StreamName}^{\uparrow}$ *window table produced by applying S to the two source schedules;*
4. *W records the* $\mathsf{CauchyModulus}^{\uparrow}$ *precision requests attached to the selected windows;*
5. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows reached at the selected windows and precision requests;*
6. *M records the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff rows by which the merged finite observations are exposed to real-name consumers;*
7. *H displays the componentwise* hsame *transports for the two regular schedules, merge schedule, window table, modulus requests, dyadic observations, handoff rows, provenance, and local naming row;*
8. *C displays the* Cont *routes for reading a selected side, forming the interleaved window, checking the modulus request, observing the dyadic row, and handing the result to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*;*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of streams, host equality of completed reals, ambient completeness, or a choice-selected limit.

Definition 416.2 (Regular Cauchy interleave classifier)**.** *Two accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carriers are classified as the same finite merge packet exactly when all rows of Definition 416.1 transport componentwise through the displayed* hsame *rows and all consumer reads factor through the displayed* Cont *routes. The classifier compares the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules, the unary merge schedule, the interleaved* $\mathsf{StreamName}^{\uparrow}$ *table, the* $\mathsf{CauchyModulus}^{\uparrow}$ *request ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff rows, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}$ *row.*

The classifier has no generator for comparing completed streams or host real values. Equality is exhausted by the finite rows carried by the packet.

Theorem 416.3 (Regular Cauchy interleave NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}$ *obligation surface for the carrier of Definition 416.1 consists of carrier admission, classifier transport, finite merge-window exactness, modulus-observation compatibility, and real-handoff boundary. Every row consumed by the certificate is one of*

$$I, J, S, T, W, D, M, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleaveUp.RegularCauchyInterleavePacket semantic name certificate`

Proof. Carrier admission is the projection of the finite packet K. The rows I, J, S, T, W, D, and M are stored coordinates, while H, C, P, and N provide transport, continuation, package, and local naming evidence.

Classifier transport is exactly Definition 416.2. It moves source schedules, merge choices, window rows, precision requests, observations, and handoffs only through displayed hsame and Cont rows.

The merge-window and modulus-observation clauses are finite repackings. A selected slot of S names one side, T records the resulting window, W records the requested precision, and D records the dyadic observation at that request.

The real-handoff clause projects M. Since no carrier coordinate stores a quotient stream, host real equality, completeness proof, or selected limit, the certificate exports only the displayed finite handoff surface. □

Theorem 416.4 (Regular Cauchy interleave carrier admission)**.** *An accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carrier is admitted exactly as the displayed finite* BHist *packet*

$$K = (I, J, S, T, W, D, M, H, C, P, N)$$

of Definition 416.1. Carrier admission projects the two $\mathsf{RegSeqRat}^{\uparrow}$ *schedules I, J, the merge schedule S, the interleaved* $\mathsf{StreamName}^{\uparrow}$ *table T, the* $\mathsf{CauchyModulus}^{\uparrow}$ *request ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff rows M, and the rows H, C, P, N for transport, continuation, package provenance, and local naming.*

Proof. Project the accepted packet by Definition 416.1. Each listed row is a coordinate of the same finite BHist object, so the admission witness is obtained by reading those coordinates and repacking them as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}$ carrier.

There is no further admission case to discharge. The carrier has no coordinate for quotient equality of streams, a host equality proof between completed names, ambient completeness, or a selected limit, so none of those objects can enter the admitted packet. □

Theorem 416.5 (Regular Cauchy interleave classifier transport exactness)**.** *Let K and K' be accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carriers. If their rows $I, J, S, T, W, D, M, P, N$ are matched componentwise by the displayed* hsame *rows in H and read through the displayed* Cont *routes in C, then the* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *classifier transports exactly those rows and no other comparison data.*

Proof. Use Definition 416.2. The classifier reads the two source schedules, the unary merge schedule, the interleaved window table, the modulus requests, the dyadic observations, the handoff rows, package provenance, and the local naming row.

The displayed hsame rows give the componentwise replacements, and the displayed Cont routes give the permitted consumer reads. Since the classifier has no generator for completed stream equality or host real equality, the transported comparison is exactly the one carried by H and C over the stored packet rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleaveUp.RegularCauchyInterleaveCarrier classifier transport exactness`

Theorem 416.6 (Regular Cauchy interleave dyadic observation handoff)**.** *For every accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carrier $K = (I, J, S, T, W, D, M, H, C, P, N)$, each selected merge slot reaches its* $\mathsf{CauchyModulus}^{\uparrow}$ *request in W and its* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in D before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff row in M is consumed.*

Proof. Start with a selected slot of the unary merge schedule S. The slot chooses one of the displayed source schedules I or J and then reads the corresponding finite $\mathsf{StreamName}^{\uparrow}$ window row in T.

The same slot carries a precision request in W. Reading that request through the stored continuation route reaches the dyadic observation row in D, with the componentwise hsame rows preserving the selected side, window, request, and observation.

Only after these rows have been displayed does the handoff coordinate M become available to $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumers. Thus the handoff is a finite dyadic-observation route, not a shortcut through a quotient stream, completed real equality, completeness theorem, or selected limit. □

Theorem 416.7 (Regular Cauchy interleave window merge exactness)**.** *For every accepted* RegularCauchyInterleave *carrier $K = (I, J, S, T, W, D, M, H, C, P, N)$, each retained row of the interleaved* $\mathsf{StreamName}^{\uparrow}$ *window table T is obtained by reading exactly one source schedule, selected by the corresponding unary merge row of S, and by pairing that selected window with the corresponding* $\mathsf{CauchyModulus}^{\uparrow}$ *request in W and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in D.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleaveUp.RegularCauchyInterleaveWindowMergeExactness`

Proof. Project the selected slot of S. Its unary row has only the displayed left and right alternatives named by the carrier, so it chooses one of the two $\mathsf{RegSeqRat}^{\uparrow}$ member schedules I and J.

Read the chosen schedule through the stored Cont route. The resulting finite window is the corresponding row of T, and the componentwise hsame rows in H transport that row together with the chosen source schedule.

Next read the matching precision and observation rows. The ledger W stores the CauchyModulus$^{\uparrow}$ request for the same slot, and D stores the DyadicRatCore$^{\uparrow}$ observation reached at that request.

Repacking these projections gives the asserted table row. No additional stream quotient or host equality comparison can alter the selected window, because the carrier contains no such coordinate. □

Theorem 416.8 (Regular Cauchy interleave selected-slot exhaustion)**.** *For every accepted* RegularCauchyInterleave$^{\uparrow}$ *packet* $K = (I, J, S, T, W, D, M, H, C, P, N)$*, each retained slot of* S *selects exactly one of the two source schedules* I *and* J*. Along that same slot route the selected* StreamName$^{\uparrow}$ *row in* T*,* CauchyModulus$^{\uparrow}$ *request in* W*,* DyadicRatCore$^{\uparrow}$ *observation in* D*, and handoff row in* M *are read as one finite chain, while the unselected side remains present only as provenance in the packet.*

Proof. Project the retained slot from the unary merge schedule S. The carrier displays only the two alternatives I and J, so the slot inversion has one selected side and one unselected side.

Apply Definition 416.7. It identifies the window row in T, the precision request in W, and the dyadic observation row in D as the rows attached to the same selected slot.

Use Definition 416.6 to continue the same route into the handoff row M. The row is reached after the selected observation, so it cannot be read from the opposite source side.

The unselected source schedule is still part of K through provenance, transport, and continuation data, but it does not supply the retained slot. Since the carrier contains no quotient stream, host equality of completed reals, completeness object, or choice-selected limit, no external object can replace the displayed slot route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyInterleaveUp.RegularCauchyInterleaveCarrier selected slot exhaustion`

Theorem 416.9 (Regular Cauchy interleave real handoff boundary)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer read of a* RegularCauchyInterleave$^{\uparrow}$ *carrier factors through the finite merge schedule* S*, the interleaved window table* T*, the modulus request ledger* W*, the dyadic observation ledger* D*, and the handoff row* M*. The consumer cannot read a quotient of streams, a host equality proof between completed names, an ambient completeness theorem, or a choice-selected real value from the packet.*

Proof. Start with a consumer read. The only real-facing coordinate of the carrier is M, and M is stored together with the window, modulus, and dyadic observation rows that produce it.

Use Definition 416.7 to identify the finite window consumed by the read. Each window row of T is tied to one selected side of S, one precision request in W, and one observation row in D.

Use Definition 416.2 for transport. It allows the consumer route to move these rows componentwise through H and C, with the same package provenance P and local naming row N.

Thus the read factors through the displayed finite packet. Since the carrier has no field for quotient streams, host equality of completed reals, completeness, or selected limits, none of those objects can be exported at the RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ boundary. □

Lemma 416.10 (Regular Cauchy interleave readback stability)**.** *Let* $K = (I, J, S, T, W, D, M, H, C, P, N)$ *and* $K' = (I', J', S', T', W', D', M', H', C', P', N')$ *be accepted* RegularCauchyInterleave$^{\uparrow}$ *carriers whose two source schedules, merge schedules, interleaved windows, request ledgers, dyadic observations, and handoff rows are related componentwise by the displayed* hsame *rows. Then the*

$\mathsf{RegSeqRat}^{\uparrow}$ *readback route carried by* M *transports to the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route carried by* M' *through the same interleaved* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows. The transport introduces no selected subsequence, quotient representative, host equality of completed real names, or choice-selected limit.*

Proof. Start from the classifier transport exactness theorem. It says that the only comparison data available to accepted interleave packets are the componentwise hsame rows and the displayed Cont routes over $I, J, S, T, W, D, M, P, N$.

Apply window merge exactness to the transported slot. The selected side, interleaved window row, precision request, and dyadic observation remain the same finite route under the componentwise transport.

Use selected-slot exhaustion to rule out hidden extraction. Each retained slot has one displayed source side and one displayed handoff chain, so no subsequence selector or quotient representative can enter between the window row and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row.

The transported readback is therefore the same finite route read through M and M'. Since neither carrier stores host equality, ambient completeness, a selected limit, or choice data, the stability claim exports none of those coordinates. □

Lemma 416.11 (Regular Cauchy interleave Real seal)**.** *For every accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *carrier* $K = (I, J, S, T, W, D, M, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal available to a downstream consumer is downstream of the interleaved* $\mathsf{StreamName}^{\uparrow}$ *table* T*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *request ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback component of* M*. The seal exports only the displayed finite readback boundary and no ambient completeness theorem, quotient stream equality, selected subsequence, selected real value, or choice coordinate.*

Proof. Project the real-facing consumer route through the real handoff boundary. The boundary forces every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read to pass through S, T, W, D, M before the downstream consumer can inspect a seal.

Use dyadic observation handoff and window merge exactness. They identify the finite window, precision request, and dyadic observation rows that feed the readback component of M.

Apply readback stability for transported carriers. The same interleaved window and dyadic observation route remains stable under componentwise hsame, so the Real seal is attached to the finite readback boundary rather than to a quotient representative.

Thus the seal exports only the finite interleave packet rows together with H, C, P, N. No carrier coordinate supplies ambient completeness, quotient stream equality, selected subsequences, selected real values, or choice data. □

Definition 416.12 (Regular Cauchy interleave obligation surface)**.** *For a concrete packet*

$$K = (I, J, S, T, W, D, M, H, C, P, N)$$

of Definition 416.1, the $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *obligation surface consists exactly of carrier admission, classifier transport, finite merge-window exactness, modulus-observation compatibility, and the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff boundary. These obligations are read only from the source schedules* I, J*, unary merge schedule* S*, interleaved window table* T*, modulus request ledger* W*, dyadic observation ledger* D*, handoff rows* M*, transport rows* H*, continuation routes* C*, package row* P*, and local naming row* N*.*

Theorem 416.13 (Regular Cauchy interleave carrier-classifier obligation package)**.** *For every accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *packet* $K = (I, J, S, T, W, D, M, H, C, P, N)$*, the carrier-admission*

and classifier-transport obligations of Definition 416.12 are exactly the concrete carrier and classifier obligations of the chapter.

Proof. Read the packet through Definition 416.1. This exposes precisely the rows $I, J, S, T, W, D, M, H, C, P, N$ and no additional analytic coordinate.

The carrier side is the admission statement of Definition 416.4. The classifier side is the componentwise transport described by Definition 416.2 and proved in Definition 416.5.

Together these references give the carrier and classifier part of the obligation surface, with the same package row and local naming row retained for downstream reads. □

Theorem 416.14 (Regular Cauchy interleave window-modulus obligation package)**.** *For every accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *packet* $K = (I, J, S, T, W, D, M, H, C, P, N)$*, the finite merge-window and modulus-observation obligations of Definition 416.12 are witnessed by the displayed rows* $S, T, W, D, M, H, C, P, N$*.*

Proof. Apply Definition 416.7. It ties each selected merge slot in S to the window table T, the request ledger W, the dyadic observation row D, and the handoff row M before any consumer read is made.

Then apply Definition 416.6. It keeps the modulus request and dyadic observation in the same finite read path under the displayed transport and continuation rows.

Thus the window and modulus obligations are fulfilled inside the concrete packet, without adding quotient streams, host real equality, or selected limits. □

Theorem 416.15 (Regular Cauchy interleave real-handoff obligation package)**.** *For every accepted* $\mathsf{RegularCauchyInterleave}^{\uparrow}$ *packet* $K = (I, J, S, T, W, D, M, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff obligation of Definition 416.12 is exactly the boundary route through* $S, T, W, D, M, H, C, P, N$*.*

Proof. Use Definition 416.9. It states that every regular-rational or real-facing consumer read factors through the finite merge schedule, interleaved window table, modulus request ledger, dyadic observation ledger, and handoff row.

The remaining rows H, C, P, N preserve the same route as transport, continuation, package provenance, and local naming. Therefore the real-handoff obligation is a finite BEDC packet obligation, not an appeal to completion, quotient streams, or host equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyInterleave}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ member schedules, a unary merge schedule, an interleaved $\mathsf{StreamName}^{\uparrow}$ window table, $\mathsf{CauchyModulus}^{\uparrow}$ requests, $\mathsf{DyadicRatCore}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The obligation scope binds carrier admission, classifier transport, merge-window exactness, selected-slot exhaustion, real-handoff boundary, and the carrier-classifier, window-modulus, and real-handoff obligation packages to the displayed finite interleave packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterleave}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyInterleaveUp.RegularCauchyInterleavePacket\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, host equality of completed real names, ambient completeness, uniqueness of a merged limit, or bridge checking.

Upgrade path: Public closure requires a single consumer-facing interleaved-window certificate for downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers.

Chapter 417

RegularCauchySum NameCert Construction

The $\mathsf{RegularCauchySum}^{\uparrow}$ interface names the finite pointwise-addition surface for two regular rational Cauchy names. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ readback layer and the $\mathsf{Real}^{\uparrow}$ seal: a consumer can inspect the two scheduled input windows, the displayed dyadic endpoint addition rows, and the combined finite error ledger, but it cannot read a quotient of streams, a selected host limit, or an ambient completeness principle.

The packet consumes the finite-window discipline of chapter 126, the dyadic endpoint arithmetic of chapter 291, and the regular-rational readback boundary of chapter 301.

Definition 417.1 (RegularCauchySum carrier)**.** *A* $\mathsf{RegularCauchySum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observation windows are to be added pointwise;*
2. *W_A and W_B are aligned* $\mathsf{StreamName}^{\uparrow}$ *windows for a shared precision request and a shared unary observation index;*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from those windows;*
4. *D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint addition row obtained from the common-exponent addition window for $D_A + D_B$;*
5. *E is the finite* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger recording the two input tolerances, their sum, and the output tolerance assigned to D;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose scheduled point row is D with the tolerance budget recorded by E;*
7. *H stores componentwise* hsame *transports for both source names, both stream windows, the three dyadic endpoint rows, the error-budget ledger, and the readback row;*
8. *C stores the displayed* Cont *routes for reading the two input windows, forming the dyadic endpoint addition row, checking the combined budget, and exposing the readback;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySum}^{\uparrow}}$ *row.*

The packet is a finite addition certificate for regular Cauchy names. It stores no quotient stream, no selected limit, no host real equality, and no ambient completeness object.

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumTasteGate single carrier alignment`

Theorem 417.2 (RegularCauchySum NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchySum}^{\uparrow}}$ *surface for the carrier of Definition 417.1 consists of five finite rows:*

1. *carrier admission: the accepted object is the displayed* BHist *packet S with two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic endpoint rows, a budget ledger, transport rows, continuation rows, provenance, and local naming data;*
2. *classifier transport: classification transports the two source rows, the aligned windows, the endpoint addition row, the error ledger, and the readback row componentwise through the stored* hsame *and* Cont *rows;*
3. *stability: replacing either input by a classified regular name preserves the aligned window and recomputes the same kind of displayed dyadic addition row;*
4. *ledger exactness: the output tolerance row is the finite sum of the two input tolerance rows recorded in E;*
5. *downstream handoff: every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the sum first reads R together with A, B, W_A, W_B, D_A, D_B, D, E, H, C, P, and N.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier semantic name certificate`

Proof. Carrier admission is the direct projection of the tuple in Definition 417.1. Each source row, window row, endpoint row, budget row, transport route, continuation route, provenance row, and local naming row is an explicit coordinate of S.

Classifier transport is componentwise. The two $\mathsf{RegSeqRat}^{\uparrow}$ source rows transport their scheduled rational observations, the $\mathsf{StreamName}^{\uparrow}$ rows transport the aligned window reads, and the $\mathsf{DyadicRatCore}^{\uparrow}$ rows transport the endpoint addition calculation at the common exponent.

The budget row is finite. It records the two input tolerances before the sum row is exposed, so the output tolerance is read as the displayed addition of those two ledger entries. The downstream handoff is then the readback row R, and no omitted coordinate can add quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 417.3 (RegularCauchySum window budget)**.** *Given two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows A and B in a* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet and a shared precision request, the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows W_A and W_B display dyadic endpoint rows D_A and D_B. The packet displays a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D for $D_A + D_B$ and a finite ledger E whose output tolerance is the displayed sum of the two input tolerance rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier window budget`

Proof. Read the shared precision request through the two stream windows. Since W_A and W_B are aligned coordinates of the same packet, they expose the two endpoint rows at the same requested observation index.

Apply the displayed common-exponent addition window of $\mathsf{DyadicRatCore}^{\uparrow}$ to D_A and D_B. The result is the endpoint row D, still inside the carrier.

Finally read the finite budget ledger E. Its two input entries are the tolerances used by the two regular names, and its output entry is their displayed sum. This proves the window budget without invoking any completed real or quotient stream. □

Theorem 417.4 (RegularCauchySum symmetric input handoff)**.** *For every accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet* $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$*, any consumer read of the sum readback* R *factors through both aligned source windows* W_A, W_B*, both endpoint rows* D_A, D_B*, the common dyadic addition row* D*, and the finite error ledger* E*. Exchanging the two source rows preserves the same handoff surface up to the displayed componentwise classifier over the two sources, two windows, two endpoint rows, addition row, budget row, readback row, transports, continuations, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier symmetric input handoff`

Proof. Read the two inputs through Definition 417.3. The window budget exposes both aligned source windows, both endpoint rows, the common dyadic addition row, and the finite ledger before the readback row is consumed.

Repack those rows through Definition 417.6. The consumer of R therefore receives the paired source data together with D and E, rather than a one-sided source row or an ambient completed-real operation.

For the exchanged packet, use the componentwise classifier carried by H and read through C. It swaps the two source rows with their matching windows and endpoints while preserving the displayed addition row, budget ledger, readback row, package provenance, and local $\mathsf{NameCert}$ row.

The carrier has no coordinate for a host-level completed-real addition law, quotient stream, selected limit, or completeness object. Thus the symmetric handoff is exactly the finite paired-source surface described by the packet classifier. □

Theorem 417.5 (RegularCauchySum two-source Real seal symmetry)**.** *Let* $S = (A, B, W_A, W_B, D_A, D_B, D, E, R,$ *be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet. Exchanging the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows preserves the* $\mathsf{Real}^{\uparrow}$ *seal route: any real-facing consumer still reads both aligned* $\mathsf{StreamName}^{\uparrow}$ *windows* W_A, W_B*, both* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D_A, D_B*, the same finite error-budget row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the same local* $\mathsf{NameCert}_{\mathsf{RegularCauchySum}^{\uparrow}}$ *provenance through* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier two source real seal symmetry`

Proof. Start with Definition 417.4. It gives the componentwise exchange of the two source rows together with their matched windows and endpoint rows, while leaving the displayed addition row, finite budget row, readback row, transport rows, continuation rows, package provenance, and local naming row on the same carrier surface.

Apply Definition 417.6 to the exchanged surface. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer still receives the aligned endpoint and budgeted pointwise-sum data through R, so the swap has not introduced a one-sided source route or an ambient completed-real operation.

Finally use Definition 417.8. Since every $\mathsf{Real}^{\uparrow}$ consumer factors through the displayed readback R and its visible provenance, the real seal route is insensitive to the order of the two accepted source rows and remains the finite two-source route listed in the statement. □

Theorem 417.6 (RegularCauchySum RegSeqRat handoff)**.** *The budgeted pointwise-sum windows of Definition 417.3 repack as the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *of Definition 417.1. A consumer of*

R reads only the aligned input windows, the dyadic endpoint addition rows, the finite error-budget ledger, the displayed hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

Proof. Start with the finite window budget. For each requested precision, the packet exposes the two source endpoint rows, the dyadic sum endpoint, and the combined tolerance row.

Repack those displayed rows as a regular rational sequence readback. The point row of R is D, and the regularity evidence visible to the consumer is the tolerance ledger E routed through the same aligned windows.

The transport and continuation rows are inherited from the carrier. Since the repacking uses only coordinates already listed in Definition 417.1, it introduces no host sequence quotient and no selected limit. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier regseqrat handoff`

Theorem 417.7 (RegularCauchySum tail-selector handoff)**.** *Let $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet whose aligned* $\mathsf{StreamName}^{\uparrow}$ *windows are read at the same precision request as the one-source selected-tail witness surface of chapter 458. Then every* $\mathsf{RegularCauchySum}^{\uparrow}$ *readback that exposes the* $\mathsf{Real}^{\uparrow}$ *seal must first consume the selected-tail witnesses for A and B, the aligned windows W_A, W_B, the dyadic endpoint rows D_A, D_B, D, the finite error ledger E, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. The handoff exposes no quotient streams, host equality, selected limits, or ambient completeness.*

Proof. Project the sum packet. Its only source-facing rows are the two regular names, the aligned $\mathsf{StreamName}^{\uparrow}$ windows, the endpoint rows, the finite ledger, the readback row, and the structural route rows.

Read the sibling tail-selector surface next. The diagonal admissibility and precision-window exactness rows of chapter 458 supply one-source selected-tail witnesses attached to finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ support rows.

Use Definition 417.6 to repack the aligned endpoint and ledger rows as the sum readback R. The selected-tail witnesses remain attached through H, C, P, N, so the sum route consumes the sibling witnesses before the $\mathsf{Real}^{\uparrow}$ seal is exposed.

Finally apply the local non-escape and seal boundary rows. They rule out quotient streams, host equality, selected limits, and ambient completeness, leaving only the finite handoff surface listed in the statement. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier tail selector handoff`

Theorem 417.8 (RegularCauchySum RealUp seal boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet reads only the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R together with the visible provenance of the two source names, aligned windows, dyadic endpoint addition rows, finite error-budget ledger,* hsame *transports,* Cont *routes,* Pkg *row, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchySum}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier realup seal nonempty iff`

Proof. Factor the real-facing read through the regular readback. By Definition 417.6, R is already the repacking of the budgeted pointwise-sum windows.

The remaining consumer-visible data are exactly the provenance rows attached to that repacking: the two source names, aligned schedules, dyadic addition endpoints, budget ledger, transports, continuations, package row, and local naming row.

The carrier has no independent Real$^{\uparrow}$ equality field and no completion witness outside the readback. Therefore the real seal can consume only the displayed regular readback and its visible provenance. □

Theorem 417.9 (RegularCauchySum non-escape boundary)**.** *No accepted* RegularCauchySum$^{\uparrow}$ *packet exports quotient streams, selected limits, host real equality, or ambient completeness. Every consumer route is exhausted by the finite* BHist *rows named in Definition 417.1 and by the handoff rows of Definition 417.6 and Definition 417.8.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier non escape boundary`

Proof. Inspect the carrier inventory. The only source coordinates are the two regular names, two stream windows, dyadic endpoint rows, finite error budget, readback row, transports, continuations, package row, and local naming row.

The RegSeqRat$^{\uparrow}$ handoff theorem shows that regular consumers receive only the budgeted pointwise-sum windows. The Real$^{\uparrow}$ seal theorem places real consumers downstream of that same regular readback and its visible provenance.

These two routes exhaust the packet. Since no listed coordinate is a quotient stream, selected limit, host real equality, or ambient completeness theorem, no consumer can read such an object from the RegularCauchySum$^{\uparrow}$ certificate. □

Theorem 417.10 (RegularCauchySum budget associative visibility)**.** *For any finite list of accepted* RegularCauchySum$^{\uparrow}$ *pointwise-sum packets built by repeated pairing, composing their visible error-budget rows is associative at the consumer boundary: every rebracketing reads the same* DyadicRatCore$^{\uparrow}$ *endpoint-addition rows and the same* FiniteErrorBudget$^{\uparrow}$ *ledger entries, routed through the displayed source* RegSeqRat$^{\uparrow}$ *rows, aligned* StreamName$^{\uparrow}$ *windows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. No composed budget introduces quotient streams, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier budget associative visibility`

Proof. Read each summand packet through Definition 417.3. A single binary sum exposes two source endpoint rows, the dyadic sum endpoint, and the finite budget entry that combines the two requested tolerances.

For repeated sums, compose those binary ledger entries. Rebracketing changes only the order in which finite budget rows are paired; it does not change the displayed DyadicRatCore$^{\uparrow}$ endpoint-addition rows or the FiniteErrorBudget$^{\uparrow}$ entries consumed by the final window.

The handoff and non-escape results, Definition 417.6 and Definition 417.9, keep the composed route on the same source rows, aligned schedules, transports, continuations, package rows, and naming rows. Since the carrier has no hidden quotient or completion coordinate, the finite budget composition remains entirely visible. □

Theorem 417.11 (RegularCauchySum seal consumer exhaustion)**.** *Every* Real$^{\uparrow}$ *consumer of a* RegularCauchySum$^{\uparrow}$ *sum seal factors through the displayed* RegSeqRat$^{\uparrow}$ *readback, aligned* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *endpoint-addition row,* FiniteErrorBudget$^{\uparrow}$ *ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The seal has no consumer-facing coordinate for quotient streams, selected limits, host real equality, ambient completeness, or an independent completed-real addition law.*

Proof. Start with the real-facing seal boundary of Definition 417.8. It already places every $\mathsf{Real}^{\uparrow}$ read downstream of the regular readback R and the provenance attached to the two source names.

Use Definition 417.6 to expand R. The readback is the repacking of aligned windows, dyadic endpoint addition rows, and the finite error-budget ledger.

Finally apply Definition 417.9. The packet inventory is exhausted by those rows together with hsame, Cont, Pkg, and NameCert, so a real consumer cannot obtain any quotient stream, selected limit, host equality, ambient completeness principle, or separate completed-real addition object from the seal. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier seal consumer exhaustion`

Theorem 417.12 (RegularCauchySum consumer obligation package)**.** *For the* $\mathsf{RegularCauchySum}^{\uparrow}$ *carrier of Definition 417.1, the five local* $\mathsf{NameCert}_{\mathsf{RegularCauchySum}^{\uparrow}}$ *rows of Definition 417.2, the checked window budget of Definition 417.3, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 417.6, the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 417.8, and the consumer exhaustion row of Definition 417.11 assemble a finite consumer-facing obligation package over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *This package exposes exactly the scheduled source windows, dyadic endpoint-addition rows, combined finite error ledger, displayed transports, continuation routes, provenance row, regular readback row, and local naming row; it excludes quotient streams, selected limits, host real equality, ambient completeness, and an independent completed-real addition law.*

Proof. Project the carrier of Definition 417.1. Its coordinates are the two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, aligned $\mathsf{StreamName}^{\uparrow}$ windows, endpoint rows in $\mathsf{DyadicRatCore}^{\uparrow}$, the finite $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the hsame and Cont rows, the Pkg provenance, and the local NameCert row.

Read the five local naming obligations through Definition 417.2. They provide carrier admission, classifier transport, stability, ledger exactness, and downstream handoff, so every consumer-facing clause is already attached to a finite displayed row.

Use the checked window-budget theorem, Definition 417.3, for dyadic endpoint formation and budget exactness. Then route the displayed rows through Definition 417.6 and Definition 417.8, which place regular and real consumers downstream of the same readback and visible provenance.

Finally apply Definition 417.11. The seal has no further consumer coordinate, so the assembled package cannot export quotient streams, selected limits, host real equality, ambient completeness, or a separate completed-real addition law. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier consumer obligation package`

Theorem 417.13 (RegularCauchySum obligation surface)**.** *The* $\mathsf{RegularCauchySum}^{\uparrow}$ *carrier of Definition 417.1 exposes a finite obligation surface over the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, their aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, the two source* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, the displayed endpoint addition row, the finite error-budget ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal-facing route, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row. Every consumer obligation for the sum packet is a projection of those rows together with Definition 417.12.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier obligation surface`

Proof. Project the carrier tuple. Its source windows, endpoint rows, budget ledger, readback row, transports, continuations, package row, and local naming row are already listed as coordinates of S.

Use Definition 417.12 to assemble the consumer-facing obligations. The package reads the local naming rows, window-budget exactness, regular readback handoff, real-seal boundary, and seal consumer exhaustion from those displayed coordinates.

Thus the obligation surface is finite and BHist-anchored. It does not introduce a completed-real addition operation, quotient equality of names, selected limit, or ambient completeness principle. □

Theorem 417.14 (RegularCauchySum classifier stability)**.** *Let two accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packets have source rows, aligned windows, endpoint rows, finite error budgets, readbacks, real-seal routes, transports, continuations, package provenance, and local* NameCert *rows matched componentwise by their displayed* hsame *classifiers. Then every consumer route supplied by the obligation surface of Definition 417.13 transports to the corresponding route of the second packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier classifier stability`

Proof. Start with the componentwise classifier data. The two source rows transport together with their aligned windows, so the scheduled rational observations remain paired at the requested precision.

Transport the endpoint rows and budget ledger next. The dyadic endpoint addition row and the finite error-budget row stay attached to the transported source windows, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback still reads the displayed sum row through the same kind of ledger.

Finally route the transported readback through the real-seal boundary and the local package rows. By Definition 417.12, every consumer route is one of these displayed projections, so classifier stability follows without invoking completed-real addition or quotient equality. □

Theorem 417.15 (RegularCauchySum ledger exactness)**.** *For every accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet, the ledger consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read is exactly the finite error-budget row* E *attached to the two aligned source windows and the displayed dyadic endpoint addition row. The packet has no alternate ledger for host real addition, quotient-stream equality, selected limits, or ambient completeness.*

Proof. Inspect a read of the regular sum. The read first reaches the aligned windows W_A, W_B, the endpoint rows D_A, D_B, the displayed addition row D, and the finite budget row E.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff repacks exactly those rows as R. The $\mathsf{Real}^{\uparrow}$ seal boundary reads downstream of R, so a real consumer sees the same finite ledger rather than a separate completed-real operation.

The consumer obligation package exhausts the route through $A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N$. Since the carrier contains no additional budget or equality coordinate, ledger exactness is precisely the displayed finite row E. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier ledger exactness`

Theorem 417.16 (RegularCauchySum zero-source identity boundary)**.** *Let* $S = (A, B, W_A, W_B, D_A, D_B, D, E, R,$ *be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet, and suppose one source row is supplied by an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet* $Z = (q, s, r, m, \mu, e, h, c, p, n)$ *through the downstream handoff of Definition 624.7. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of* S *reads the other source window together with the zero packet route*

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

displayed by q, s, r, e, the dyadic zero endpoint carried by q and the zero regular row r, the unchanged finite error-budget ledger E, the same componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of the sum and zero packets.*

The boundary is an identity only at the finite consumer surface: replacing one source by the zero packet leaves the visible sum consumer route exhausted by the nonzero source window, the zero-source route, endpoint and budget rows, readback, seal row, transports, continuations, provenance, and naming data. It exports no host real addition law, quotient stream equality, selected limit, ambient completeness, or independent completed-real zero.

Proof. Begin with the sum packet. The window-budget theorem Definition 417.3 exposes the two aligned source windows, the source endpoint rows, the displayed endpoint-addition row, and the finite ledger E.

Now specialize one source to the zero packet. The downstream handoff Definition 624.7 supplies the finite zero-source boundary $q, s, r, m, \mu, h, c, p, n$, and the seal non-escape theorem Definition 624.6 routes every real-facing zero read through the same displayed zero seed, stream, regular row, meet row, and seal row.

Pass back to the sum consumer surface. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 417.6 repacks the nonzero source window, the zero source route, the endpoint rows, and E as the sum readback R. The $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 417.8 keeps every real consumer downstream of that readback.

Finally apply ledger exactness. By Definition 417.15, no alternate budget, equality, limit, completeness, or completed-real addition coordinate can be read. Projecting componentwise through $A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N$ leaves exactly the finite identity boundary named in the statement. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier zero source consumer route`

Theorem 417.17 (RegularCauchySum bridged public route)**.** *The public* $\mathsf{RegularCauchySum}^{\uparrow}$ *route is exhausted by the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-addition row, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal route, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row. A public consumer of the sum has no additional route through completed-real addition, quotient-stream equality, selected limits, host real equality, or ambient completeness.*

Proof. First read the public window and budget rows. The window-budget theorem Definition 417.3 fixes the aligned source windows, endpoint-addition row, and finite error ledger that any public sum read may inspect.

Next route the regular and real consumers. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 417.6 repacks the same endpoint and ledger rows as regular readback, while the $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 417.8 places the real-name route downstream of that readback.

Finally assemble the public package. The consumer obligation package of Definition 417.12 lists the transport, continuation, provenance, and local naming rows, and the seal consumer exhaustion theorem Definition 417.11 rules out any further consumer coordinate beyond the displayed public route. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier bridged public route`

Theorem 417.18 (RegularCauchySum standard bridge source packet)**.** *For every accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *carrier, the standard bridge source is exactly the finite packet of two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-addition row, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$

ledger, the RegSeqRat$^{\uparrow}$ *readback, the* Real$^{\uparrow}$ *seal-facing route, and the displayed* hsame, Cont, Pkg, *and* NameCert *rows already assembled by Definition 417.12. It exports no quotient stream, selected limit, host real equality, ambient completeness, or independent completed-real addition law.*

Proof. Start with the window budget of Definition 417.3. It supplies the two aligned source windows, the endpoint-addition row, and the finite error ledger.

Use the RegSeqRat$^{\uparrow}$ handoff and Real$^{\uparrow}$ seal boundary, Definition 417.6 and Definition 417.8. These route regular and real consumers through the same readback row and provenance surface.

Apply the consumer obligation package and ledger exactness. The package gathers the transport, continuation, package, and naming rows, while ledger exactness fixes the finite budget as the only error ledger. Therefore the bridge source is precisely the displayed public packet and not a quotient stream, host limit, host equality, ambient completeness field, or separate completed-real addition object. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier standard bridge source packet`

Theorem 417.19 (RegularCauchySum limit-classifier compatibility)**.** *For every accepted* RegularCauchySum$^{\uparrow}$ *packet* $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$*, any completion-facing* RegularCauchyLimitClassifier$^{\uparrow}$ *read that consumes the sum factors through the displayed* RegSeqRat$^{\uparrow}$ *readback row* R*. The factorization keeps the aligned* StreamName$^{\uparrow}$ *windows* W_A, W_B*, the* DyadicRatCore$^{\uparrow}$ *endpoint-addition row* D *with source endpoints* D_A, D_B*, the* FiniteErrorBudget$^{\uparrow}$ *ledger* E*, the* Real$^{\uparrow}$ *seal route, the componentwise* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *rows as the only public handoff surface.*

Equivalently, the limit-classifier side may consume the sum only as a sealed regular readback compatible with the diagonal-window public boundary of RegularCauchyLimitClassifier$^{\uparrow}$*. It does not receive quotient stream equality, a selected limit, host real equality, ambient completeness, or an independent completed-real addition operation.*

Proof. First expose the finite sum packet. The window-budget theorem Definition 417.3 supplies the aligned source windows, the two source endpoint rows, the displayed endpoint-addition row, and the finite error ledger.

Next pass from the endpoint packet to the regular readback. The RegSeqRat$^{\uparrow}$ handoff of Definition 417.6 repacks those displayed rows as R, while the Real$^{\uparrow}$ seal boundary of Definition 417.8 places every real-facing read downstream of the same regular row and its visible provenance.

Now compare this handoff with the classifier boundary. The diagonal-window public boundary for RegularCauchyLimitClassifier$^{\uparrow}$, Definition 443.4, admits completion-facing reads only through displayed stream windows, regular readbacks, dyadic ledgers, Real seal rows, BHist history, hsame, Cont, Pkg, and local NameCert data. Substituting the sum readback R into that boundary therefore preserves exactly the rows listed in the statement.

Finally apply the seal consumer exhaustion theorem Definition 417.11. The RegularCauchySum$^{\uparrow}$ carrier has no further consumer coordinate, so the classifier cannot obtain quotient stream equality, a selected limit, host real equality, ambient completeness, or a separate completed-real addition operation from the sum. □

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier limit classifier compatibility`

Theorem 417.20 (RegularCauchySum tail-selector compatibility)**.** *Let* $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C$ *be an accepted* RegularCauchySum$^{\uparrow}$ *packet, and let accepted* RegularCauchyTailSelector$^{\uparrow}$ *packets for the source rows* A *and* B *answer the same precision request. Every public sum read factors through the two selected-tail witness rows, the aligned* StreamName$^{\uparrow}$ *windows* W_A, W_B*, the*

DyadicRatCore$^{\uparrow}$ *endpoint addition row D with source endpoints D_A, D_B, the* FiniteErrorBudget$^{\uparrow}$ *ledger E, the* RegSeqRat$^{\uparrow}$ *readback row R, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows.*

The handoff exports no quotient stream, selected limit, host real equality, ambient completeness, or independent completed-real addition law. The tail selectors supply finite witness rows for the two source windows, and the sum packet supplies only the budgeted pointwise readback built from those rows.

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier tail selector compatibility`

Proof. First read the two tail-selector packets. The diagonal admissibility theorem Definition 458.3 supplies selected-tail witness rows for the same precision request, and Definition 458.5 identifies each witness with its finite StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ support rows.

Next compose those witnesses with the sum window. The window-budget theorem Definition 417.3 reads the aligned windows W_A, W_B, the endpoint rows D_A, D_B, the endpoint-addition row D, and the finite error ledger E at the shared request.

Now pass to the regular readback. The RegSeqRat$^{\uparrow}$ handoff of Definition 417.6 repacks the endpoint addition and budget rows as R, while the consumer obligation package records the hsame, Cont, Pkg, and local naming rows that keep the two selected-tail witnesses attached to the sum packet.

Finally combine seal exhaustion with limit-classifier compatibility. The seal consumer exhaustion theorem rules out any additional consumer coordinate, and Definition 417.19 keeps completion-facing reads downstream of the displayed regular readback. Thus the compatibility handoff is finite and cannot export quotient streams, selected limits, host equality, ambient completeness, or a separate completed-real addition law. □

Theorem 417.21 (RegularCauchySum shared-tail-meet handoff)**.** *Let $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C,$ be an accepted* RegularCauchySum$^{\uparrow}$ *packet. Suppose the source rows A and B have accepted* RegularCauchyTailSelector$^{\uparrow}$ *packets for the same precision request, and suppose those two selector packets are absorbed by one accepted* RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold packet for that request. Then every* Real$^{\uparrow}$*-facing read of the sum factors through the shared threshold row of the tail-meet packet, the two selected-tail witness rows, the aligned* StreamName$^{\uparrow}$ *windows W_A, W_B, the* DyadicRatCore$^{\uparrow}$ *endpoint addition row D with source endpoints D_A, D_B, the* FiniteErrorBudget$^{\uparrow}$ *ledger E, the* RegSeqRat$^{\uparrow}$ *readback row R, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows of the sum, selectors, and shared-tail meet packet.*

In particular, a diagonal Real$^{\uparrow}$ *consumer cannot read the two source selectors independently once the shared-tail meet certificate is supplied. Its sum read is mediated by the single shared threshold and then by the same endpoint-addition, budget, readback, transport, continuation, provenance, and naming surface already displayed by the* RegularCauchySum$^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier shared tail meet handoff`

Proof. First use Definition 417.20. It places every public sum read behind the two selected-tail witness rows, the aligned windows, endpoint rows, endpoint-addition row, finite error ledger, regular readback, and the displayed structural rows of the sum packet.

Next use Definition 560.39. The two selector witnesses for the common precision request are admitted into one paired schedule and one shared-threshold route rather than into two unrelated tail routes.

Then apply Definition 560.11. The accepted tail-meet packet exposes its threshold as a public finite certificate over the paired source rows, paired windows, source modulus rows, dyadic threshold ledger, comparison row, transports, continuations, provenance, and local naming data.

Finally compose this certificate with Definition 417.19. Completion-facing and diagonal $\mathsf{Real}^{\uparrow}$ reads of the sum must pass through the regular readback R, and the readback is already downstream of the shared threshold, selected-tail witnesses, aligned windows, endpoint addition, finite ledger, transports, routes, provenance, and naming rows. Hence the handoff exports no quotient stream, selected limit, host real equality, ambient completeness, or independent completed-real addition law. □

Theorem 417.22 (RegularCauchySum CauchyLimitSeal handoff)**.** *Let $S = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C$ be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet whose readback row R is consumed by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Then every* $\mathsf{Real}^{\uparrow}$*-facing read of the sum factors through the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows A, B, the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows W_A, W_B, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint addition row D with source endpoint rows D_A, D_B, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger E, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and then the public route packet of Definition 386.8.*

The handoff exposes only the displayed sum rows, the CauchyLimitSeal public route rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, selected host limit, ambient completeness coordinate, or independent completed-real addition law.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier cauchy limit seal handoff`

Proof. First expose the finite sum route. The window-budget theorem Definition 417.3 supplies the two source windows, endpoint rows, endpoint-addition row, and finite error ledger.

Next pass through the regular readback. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 417.6 repacks those rows as R, and the $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 417.8 places every real-facing read downstream of the same readback and visible provenance.

Now consume the downstream seal packet. The CauchyLimitSeal public route boundary Definition 386.8 accepts a displayed regular source route before exposing its seal-facing surface, so substituting R gives the stated route from the two source regular names through the sum packet and then through the public seal packet.

Finally apply the local non-escape clauses. The sum packet excludes quotient streams, selected host limits, ambient completeness, and independent completed-real addition, while the CauchyLimitSeal boundary keeps the downstream seal on its displayed public route. The composite handoff therefore exposes exactly the finite rows named in the statement. □

Theorem 417.23 (RegularCauchySum dyadic cofinal selector handoff)**.** *Let $S = (A, B, W_A, W_B, D_A, D_B, D, E, R,$ be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet. Suppose the source rows A and B are routed through accepted* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *packets at a common* $\mathsf{Rat}^{\uparrow}$ *precision, the selected windows are absorbed by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet, and the readback row R is consumed by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Then every* $\mathsf{Real}^{\uparrow}$*-facing read of the sum factors through the two dyadic cofinal selector packets of chapter 680, the shared threshold row of chapter 560, the endpoint-addition row D with source endpoint rows D_A, D_B, the finite budget row E, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the CauchyLimitSeal public route of Definition 386.8.*

Equivalently, the diagonal Real and Completion route for the sum is the finite composite of the two dyadic selector surfaces, the shared-tail meet certificate, the displayed $\mathsf{RegularCauchySum}^{\uparrow}$

endpoint-addition and budget rows, the regular readback, and the downstream seal packet. It exports no quotient stream, selected host limit, countable choice selector, host real equality, ambient completeness coordinate, or independent completed-real addition law.

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier dyadic cofinal selector handoff`

Proof. First route the two sources through the dyadic selector packets. The dyadic cofinal tail selector carrier and Real seal handoff in chapter 680 expose only the common $\mathsf{Rat}^{\uparrow}$ request, finite dyadic window family, selected tail row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal row, transports, continuations, provenance, and local naming.

Next connect those selector rows to the sum packet. Apply Definition 417.20. The selected-tail witnesses supplied by the two dyadic selector packets are the source-side tail rows consumed by the aligned windows W_A, W_B, the endpoint rows D_A, D_B, the endpoint-addition row D, the finite error ledger E, and the regular readback R.

Now absorb the two selected windows into one threshold. By Definition 417.21, together with the shared-threshold certificate from chapter 560, the two selector routes are no longer independent consumer coordinates. Every real-facing sum read passes through the shared threshold before it reaches the endpoint-addition, budget, readback, transport, continuation, provenance, and naming rows of the sum packet.

Finally compose with the downstream seal. The CauchyLimitSeal handoff of Definition 417.22 sends the readback R into the public route of Definition 386.8. The composite route is therefore exactly the dyadic-selector, shared-threshold, endpoint-addition, finite-budget, regular-readback, and seal-public-route surface named in the statement, with no additional quotient, choice, host-limit, host-equality, completeness, or completed-addition coordinate. □

Theorem 417.24 (RegularCauchySum Real seal route)**.** *Every* $\mathsf{RegularCauchySum}^{\uparrow}$ *consumer that reads the resulting* $\mathsf{Real}^{\uparrow}$ *seal factors first through the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* A, B*, the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows* W_A, W_B*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger with source rows* D_A, D_B *and addition row* D*, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row* E*, the shared-tail-meet certificate, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route packet before reading the seal. The seal read is therefore a finite composite of the displayed two-source regular names, window data, dyadic ledger, shared tail threshold, regular readback, and downstream seal rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier real seal route`

Proof. First expose the sum packet. The seal boundary Definition 417.8 and non-escape boundary Definition 417.9 keep every real-facing read downstream of the displayed carrier rows.

Next exhaust the local consumer surface. The seal consumer exhaustion theorem Definition 417.11 prevents a consumer from bypassing the $\mathsf{RegSeqRat}^{\uparrow}$ readback, while Definition 417.21 routes the two source windows through one shared threshold.

Now attach the downstream seal. The CauchyLimitSeal handoff Definition 417.22 sends the same readback into the public seal route, and the dyadic cofinal selector handoff Definition 417.23 supplies the finite dyadic selector surface feeding the shared threshold.

Composing these routes gives exactly the displayed two-source regular names, windows, dyadic ledger, shared-tail meet, readback, and CauchyLimitSeal rows before the Real seal is read. □

Theorem 417.25 (RegularCauchySum finite-budget non-escape)**.** *The Real-seal route of Definition 417.24 exports no quotient stream, no host real equality, no selected limit, no ambient completeness coordinate, and no independent tail oracle. Its public content is exactly the displayed two-source* $\mathsf{RegSeqRat}^{\uparrow}$ *surface, aligned* $\mathsf{StreamName}^{\uparrow}$ *window surface,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint*

ledger, finite budget row, shared-tail-meet handoff, CauchyLimitSeal$^{\uparrow}$ *handoff, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchySumUp.RegularCauchySumCarrier finite budget nonescape`

Proof. Begin with the non-escape boundary. The theorem Definition 417.9 excludes quotient stream equality, host real equality, selected limits, ambient completeness, and completed-real addition from the RegularCauchySum$^{\uparrow}$ carrier.

The selector side adds no independent oracle. The shared-tail-meet handoff Definition 417.21 absorbs the two selected windows into a single finite threshold, and the dyadic cofinal selector handoff Definition 417.23 keeps the selector reads inside displayed dyadic and window rows.

The seal side also remains finite. The CauchyLimitSeal handoff Definition 417.22 consumes the regular readback through a public route packet, and the seal consumer exhaustion theorem Definition 417.11 leaves no further consumer coordinate.

Therefore the composite route exports only the named finite rows and refuses every quotient, host-equality, selected-limit, completeness, or independent-tail claim. □

Closure status for NameCert$_{\text{RegularCauchySum}^{\uparrow}}$.

Constructive story: A RegularCauchySum$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, aligned StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ endpoint addition rows, a finite FiniteErrorBudget$^{\uparrow}$ ledger, displayed hsame transports, Cont routes, Pkg provenance, a RegSeqRat$^{\uparrow}$ readback row, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchySum}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 417.12, Definition 417.13, Definition 417.14, and Definition 417.15 close the public consumer-facing package over DyadicRatCore$^{\uparrow}$, FiniteErrorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchySum}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, selected limits, host real equality, ambient completeness, additive group laws for real names, or bridge checking.

Upgrade path: No further paper closure grade remains for this chapter; Lean formalization and bridge checking remain pending.

Chapter 418

A Concrete Naming Certificate for $\mathsf{BinaryExpansion}^{\uparrow}$

The $\mathsf{BinaryExpansion}^{\uparrow}$ interface names the finite dyadic-prefix certificate that reads a binary expansion as real-name data. Following Bishop and Bridges, *Constructive Analysis*, the packet treats binary observations as displayed dyadic rational prefixes with explicit precision windows and regular-sequence handoff rows. It sits over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and it exports only finite prefix, dyadic ledger, readback, transport, continuation, package, and local $\mathsf{NameCert}$ rows.

The packet consumes the sibling $\mathsf{StreamName}^{\uparrow}$ prefix-window surface of chapter 126, the $\mathsf{DyadicApproximation}^{\uparrow}$ partial-sum and tail-radius surface of chapter 335, and the $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ seal route of chapter 301 and chapter 38. Thus the binary packet is not an isolated carrier: its rows W, A, R, and E are transported consumer rows from those sibling certificates.

Definition 418.1 (BinaryExpansion carrier)**.** *A* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, A, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *D is a finite row of binary digits, each digit displayed as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *bit row with value* 0 *or* 1*;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window schedule recording which initial segment of D is inspected for each requested unary precision;*
3. *A assigns to every prefix window in W the corresponding* $\mathsf{DyadicApproximation}^{\uparrow}$ *row, read as a dyadic rational partial sum together with a dyadic tail-radius row;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row asserting that the displayed dyadic partial sums form a regular rational stream at the scheduled precisions;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-boundary row reached from the regular stream, with no host real endpoint outside the displayed packet;*
6. *H records componentwise* hsame *transport for digit rows, prefix windows, dyadic partial sums, regular-sequence rows, and real-seal rows;*
7. *C records the* Cont *routes by which consumers read a prefix, its dyadic approximation row, the regular-sequence handoff, and the real seal in that order;*

8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BinaryExpansion}^{\uparrow}}$ *row.*

The carrier predicate accepts exactly finite dyadic-prefix packets whose prefix sum, tail radius, regular-sequence handoff, and real-seal route are all displayed in these coordinates. It has no coordinate for an infinite bitstream as a host object, quotient equality of streams, a selected real representative, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionTasteGate single carrier alignment`

Theorem 418.2 (BinaryExpansion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BinaryExpansion}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier: the* BHist *packet of Definition 418.1;*
2. *classifier: componentwise transport of binary digit rows, prefix-window schedules, dyadic partial sums, tail-radius rows, regular-sequence handoff rows, and real-seal rows through the displayed* hsame *and* Cont *routes;*
3. *stability: any shared prefix window reads the same dyadic partial-sum row and the same tail-radius row after classifier transport;*
4. *exactness: the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers read only the dyadic approximation rows named by the prefix ledger;*
5. *ledger: the* Pkg *and local* NameCert *rows exhaust the dependency surface over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket namecert obligations`

Proof. Carrier habitation is the displayed tuple of Definition 418.1. Each digit row, prefix window, dyadic approximation row, regular handoff, real seal, transport route, continuation route, package row, and local naming row is visible before a consumer can inspect the packet.

The classifier row is componentwise. Digit values transport through $\mathsf{DyadicRatCore}^{\uparrow}$, prefix schedules through $\mathsf{StreamName}^{\uparrow}$, dyadic partial sums and tail radii through $\mathsf{DyadicApproximation}^{\uparrow}$, and the regular and real boundary rows through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

The stability and exactness rows are then read from the same finite prefix ledger. A shared window selects one displayed prefix, and the dyadic partial sum and tail-radius rows named for that window are the only rows available to the regular-sequence and real-seal consumers.

Finally, P and N list only the named dependency certificates and local provenance. Since the carrier has no coordinate for host infinite streams, quotient equality, selected representatives, or ambient completeness, the five rows exhaust the naming certificate surface. □

Theorem 418.3 (BinaryExpansion prefix-window stability)**.** *Let* $B = (D, W, A, R, E, H, C, P, N)$ *and* $B' = (D', W', A', R', E', H', C', P', N')$ *be accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carriers. Suppose a displayed prefix window of* W *and a displayed prefix window of* W' *are identified by the* $\mathsf{StreamName}^{\uparrow}$ *classifier and that the digit rows on those windows transport componentwise through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *classifier. Then the corresponding dyadic partial-sum rows, dyadic tail-radius rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal-boundary rows transport through the same* hsame, Cont, Pkg, *and local* NameCert *coordinates.*

Thus a prefix comparison determines the finite readback surface consumed by regular-sequence and real-name readers. It does not create a second approximation ledger, a quotient of bitstreams, or a choice-selected representative.

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket prefix window stability`

Proof. Project the two carrier packets. The matched prefix windows select finite digit rows from D and D' and select the dyadic approximation rows in A and A' attached to those windows.

Transport the digits through the $\mathsf{DyadicRatCore}^{\uparrow}$ classifier and the window schedules through $\mathsf{StreamName}^{\uparrow}$. The displayed continuation routes then read the same finite prefix length before any dyadic partial sum is inspected.

The dyadic approximation rows are computed inside the displayed ledger for that prefix. Their partial sums and tail radii therefore transport through the recorded hsame rows and are read through the recorded Cont routes.

Finally, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal boundary are downstream of those dyadic rows. Repacking the transported prefix, approximation, handoff, and seal rows gives the stated stability, with no extra ledger or representative outside the packet. □

Theorem 418.4 (BinaryExpansion RealUp readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer reading an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *packet may read exactly the finite route consisting of a* $\mathsf{StreamName}^{\uparrow}$ *prefix window, the binary digit rows on that window, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum and tail-radius ledger for the window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular handoff, and the displayed* $\mathsf{Real}^{\uparrow}$ *seal row.*

The readback boundary exports no host real value, no completed infinite binary stream, no quotient equality of expansions, no selected representative, and no ambient completeness principle.

Proof. Begin with the carrier of Definition 418.1. A real-name read must pass through the continuation route C, so it first selects a finite prefix window in W and the corresponding binary digit rows in D.

The selected window names a dyadic approximation row in A. That row supplies the partial sum and tail-radius ledger at the requested precision, both read through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{DyadicApproximation}^{\uparrow}$.

The regular-sequence handoff R consumes only those dyadic rows. The seal row E is then the displayed $\mathsf{Real}^{\uparrow}$ boundary reached from R, with H, C, P, and N recording transport, read route, dependency package, and local naming evidence.

Since every step is a projection from the packet, no host real endpoint, completed stream, quotient equality, selected representative, or ambient completeness theorem can be exported by the real readback. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket real readback boundary`

Theorem 418.5 (BinaryExpansion dyadic-ledger exhaustion)**.** *For an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier, every dyadic fact exposed to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer is one of the following rows: a displayed bit row in D, a prefix-window row in W, a partial-sum row in A, a tail-radius row in A, a regular handoff row in R, or the real-seal row in E, transported and routed by H, C, P, and N.*

Consequently the dyadic ledger is exhausted by the finite packet. There is no exported coordinate for an unbounded bitstream object, binary-expansion uniqueness, base-two endpoint convention, quotient stream equality, or classical real completeness.

Proof. List the dyadic-bearing coordinates of Definition 418.1. Digits live in D, window requests in W, and the prefix sums and tail radii in A; the downstream rows R and E only consume those displayed dyadic facts.

Transport can only move these rows componentwise through H, and continuation can only expose them along the routes recorded in C. The package and local certificate rows P and N record provenance, not additional dyadic data.

Thus every dyadic read available to regular-sequence or real-name consumers is already named by the finite ledger. The excluded objects have no coordinate in the packet and cannot be reconstructed from the exported rows. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket dyadic ledger exhaustion`

Theorem 418.6 (BinaryExpansion prefix tail-radius monotonicity)**.** *Let $B = (D, W, A, R, E, H, C, P, N)$ be an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier, and let $w \leq w'$ be displayed prefix-window rows in W with w' extending w by finitely many binary digit rows from D. The dyadic tail-radius row attached to w' is routed through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit ledger and the same* $\mathsf{DyadicApproximation}^{\uparrow}$ *approximation ledger as the row attached to w, with the extended prefix exposing only a smaller or equal displayed tail-radius bound.*

Equivalently, extending the finite prefix window preserves the tail-radius ledger consumed downstream by $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*: the old radius row transports to the extended-window comparison through H, the extended radius row is read through the corresponding continuation in C, and both rows remain in the package and local* NameCert *surface of Definition 418.1. No host infinite stream, quotient comparison, selected real representative, or ambient completeness principle is used to justify the monotonic read.*

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket prefix tail radius monotonicity`

Proof. Start with the carrier of Definition 418.1. The two windows w and w' are finite rows of the same $\mathsf{StreamName}^{\uparrow}$ schedule, and the displayed relation $w \leq w'$ says that the second row is obtained by appending finitely many digit rows already present in D.

Apply the prefix-window stability theorem Definition 418.3 to keep the common prefix, its digit rows, and the attached dyadic approximation rows on the same hsame and Cont routes. The dyadic-ledger exhaustion theorem Definition 418.5 then says that the only available tail-radius rows are the two rows already listed in A.

The tail radius of a displayed binary prefix is the dyadic remainder bound stored by the $\mathsf{DyadicApproximation}^{\uparrow}$ row for that prefix. Extending the prefix consumes additional $\mathsf{DyadicRatCore}^{\uparrow}$ bit rows, so the unrevealed suffix bound can only move to the displayed smaller-or-equal radius row named for w'.

Finally, Definition 418.4 identifies the downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads with this same finite ledger. Hence the monotonicity claim is a statement about displayed prefix rows and their recorded tail-radius entries, not about an external stream or a quotient real. □

Theorem 418.7 (BinaryExpansion regular handoff factorization)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read from an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier factors through three finite rows: the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window schedule W, the* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum row of A selected by that window, and the* $\mathsf{DyadicApproximation}^{\uparrow}$ *tail-radius row of A selected by that same window. The handoff row R contains no independent rational stream beyond these displayed prefix, sum, and radius rows.*

Thus any regular-sequence consumer receives a finite route

$$W \longrightarrow A_{\mathrm{sum}} \longrightarrow A_{\mathrm{tail}} \longrightarrow R$$

inside the BHist *packet of Definition 418.1, transported by H, read by C, and certified by P and N.*

Proof. Project the accepted carrier of Definition 418.1. A regular handoff can only inspect the row R, and the definition of the carrier records R as downstream of the prefix-window schedule and dyadic approximation ledger.

Use Definition 418.3 to identify the selected prefix window with its unique displayed partial-sum and tail-radius rows. The preceding monotonicity theorem keeps the tail-radius ledger stable when the consumer asks for a stronger precision window.

Apply Definition 418.5. It rules out any hidden rational stream, extra approximation table, or external regularity witness outside D, W, A, R, E, H, C, P, and N.

The readback boundary of Definition 418.4 gives the same factorization for every later real-facing consumer. Repacking the selected window, partial-sum row, tail-radius row, and handoff row along their displayed routes proves the claimed factorization. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket regular handoff factorization`

Theorem 418.8 (BinaryExpansion RealUp seal prefix coherence)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier is coherent with the same finite prefix ledger used by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. If the seal row E is cited at a requested precision, the citation factors through a displayed prefix window of W, its binary digit rows in D, its partial-sum and tail-radius rows in A, and the regular handoff row R before reaching E.*

Consequently the $\mathsf{Real}^{\uparrow}$ *seal cannot read a host infinite binary stream, quotient equality of binary expansions, endpoint convention data, a selected representative, or ambient real completeness. It reads only the finite prefix packet and its displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ consumer citation of the seal row E. By Definition 418.4, such a citation must first pass through a $\mathsf{StreamName}^{\uparrow}$ prefix window, the corresponding digit rows, the dyadic approximation ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The regular handoff factorization above identifies the regular row R with the selected prefix window, partial-sum row, and tail-radius row. The prefix tail-radius monotonicity theorem ensures that stronger precision requests remain on the same finite tail-radius ledger rather than creating a second approximation object.

Now apply Definition 418.3 to transport coherent prefix comparisons through H and C. The dyadic-ledger exhaustion theorem Definition 418.5 accounts for every dyadic fact that the seal can cite.

Since the carrier of Definition 418.1 has no coordinate for host streams, quotient equality, endpoint conventions, selected representatives, or ambient completeness, the seal read is coherent exactly with the displayed finite prefix ledger. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket real seal prefix coherence`

Theorem 418.9 (BinaryExpansion dyadic-window public readiness)**.** *Every accepted public read of an accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carrier factors through a finite dyadic window before it can be treated as public-ready. The route is exactly a* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit ledger in D, a* $\mathsf{StreamName}^{\uparrow}$ *prefix-window schedule in W, the corresponding* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum and tail-radius rows in A, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal row E, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows H, C, P, N.*

The public read exports no quotient bitstream, trichotomy decision, selected representative, endpoint convention, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the real readback boundary, Definition 418.4. It forces every real-facing read to pass through the finite prefix window, digit rows, dyadic approximation ledger, regular handoff, and seal row.

Use the regular handoff factorization, Definition 418.7. It identifies R with the same selected prefix window, partial-sum row, and tail-radius row, so the public route cannot insert a second rational stream.

Apply $\mathsf{Real}^{\uparrow}$ seal prefix coherence, Definition 418.8. The seal read remains coherent with the finite dyadic ledger and the displayed H, C, P, N support rows.

The carrier of Definition 418.1 contains no quotient bitstream, trichotomy oracle, selected representative, endpoint convention, or ambient completeness coordinate. Hence public readiness is exactly the displayed finite dyadic-window route. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket dyadic window public readiness`

Theorem 418.10 (BinaryExpansion prefix-refinement RealUp seal determinacy)**.** *Let $B = (D, W, A, R, E, H, C, P$ and $B' = (D', W', A', R', E', H', C', P', N')$ be accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *carriers. Suppose a displayed finite prefix-refinement chain identifies the inspected* $\mathsf{StreamName}^{\uparrow}$ *window of W with the inspected* $\mathsf{StreamName}^{\uparrow}$ *window of W', transports the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit rows through the displayed* hsame *routes, and transports the corresponding* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum and tail-radius rows through the displayed* hsame *and* Cont *routes. Then the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows and* $\mathsf{Real}^{\uparrow}$ *seal rows reached from those inspected windows determine the same classifier-facing* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

The determined boundary is exactly the boundary read through the common finite prefix ledger. It exports no host infinite stream, no quotient equality of binary expansions, no endpoint convention selection, and no ambient completeness principle.

Proof. Apply prefix-window stability, Definition 418.3, to the displayed refinement chain. The identified $\mathsf{StreamName}^{\uparrow}$ windows remain on one finite route, and the transported digit rows remain the only $\mathsf{DyadicRatCore}^{\uparrow}$ rows available to the comparison.

Apply dyadic-ledger exhaustion, Definition 418.5. The transported partial sums and tail radii exhaust the dyadic data exposed to downstream readers, so the refinement chain cannot introduce a separate approximation table or an external binary stream.

Use regular handoff factorization, Definition 418.7. Both $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows factor through the same inspected window, partial-sum row, and tail-radius row after transport through H, H' and readback through C, C'.

Finally apply $\mathsf{Real}^{\uparrow}$ seal prefix coherence, Definition 418.8. Since each seal citation reaches E or E' only through the already identified finite ledger and regular handoff, the classifier-facing seal boundary is the same.

The carrier inventory of Definition 418.1 lists no coordinate for host infinite streams, quotient equality of expansions, endpoint convention selection, or ambient completeness. Thus the determinacy claim is exactly the displayed packet claim over $D, W, A, R, E, H, C, P, N$ and $D', W', A', R', E', H', C', P', N'$. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket prefix refinement real seal determinacy`

Theorem 418.11 (BinaryExpansion endpoint ambiguity ledger boundary)**.** *Any downstream endpoint convention resolver for dual binary expansions must enter as a separate ledger or socket row outside the accepted* $\mathsf{BinaryExpansion}^{\uparrow}$ *packet. The packet of Definition 418.1 exports only finite digit rows, prefix windows, dyadic partial sums, tail-radius rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* hsame, Cont, Pkg, *and* NameCert *routes.*

Consequently no accepted BinaryExpansion$^{\uparrow}$ *carrier exports a host infinite stream, quotient equality of binary expansions, selected representative, or base-two endpoint convention. Such a convention can be consumed only after it is named by an additional finite BEDC row not present in* $D, W, A, R, E, H, C, P, N$.

Proof. Use prefix-window stability, Definition 418.3. It keeps every comparison of binary data on finite digit rows, finite prefix windows, and their displayed hsame and Cont routes.

Apply dyadic-ledger exhaustion, Definition 418.5. The only dyadic facts available to later readers are the digit rows in D, the prefix-window rows in W, the partial-sum and tail-radius rows in A, the regular handoff R, and the real-seal row E.

The regular handoff factorization, Definition 418.7, then shows that a RegSeqRat$^{\uparrow}$ consumer cannot receive an independent infinite bitstream or a second representative selector. The Real$^{\uparrow}$ seal prefix coherence theorem, Definition 418.8, gives the same boundary for real-facing consumers.

Therefore endpoint ambiguity is not resolved by the accepted packet. A base-two convention resolver must be represented by a separate ledger or socket row, because no coordinate of $D, W, A, R, E, H, C, P, N$ contains such a convention. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket endpoint ambiguity ledger boundary`

Theorem 418.12 (BinaryExpansion Cauchy modulus handoff)**.** *Let* $B = (D, W, A, R, E, H, C, P, N)$ *be an accepted* BinaryExpansion$^{\uparrow}$ *carrier. Every effective precision read needed by a* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or* CauchyModulus$^{\uparrow}$ *consumer factors through the same finite route: a prefix window in* W, *the corresponding dyadic partial-sum and tail-radius rows in* A, *the regular handoff row* R, *the real-seal row* E, *and the displayed* hsame, Cont, Pkg, *and* NameCert *rows* H, C, P, *and* N.

Thus the effective Cauchy-modulus handoff exported by B *is the displayed prefix-tail-radius ledger and its regular and real-facing routes. It contains no hidden modulus, host infinite bitstream, quotient stream equality, selected representative, or ambient completeness principle.*

Proof. Begin with a precision request made by a RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, or CauchyModulus$^{\uparrow}$ consumer. By prefix tail-radius monotonicity, Definition 418.6, strengthening the requested precision only moves to a displayed larger prefix window and its smaller-or-equal tail-radius row; it does not create a second modulus object.

Apply regular handoff factorization, Definition 418.7. The RegSeqRat$^{\uparrow}$ read is exactly the route from the selected window in W through the partial-sum and tail-radius rows of A into R.

Next use Real$^{\uparrow}$ seal prefix coherence, Definition 418.8. Any real-facing read of E reaches the seal only after passing through the same finite prefix ledger and the same regular handoff row.

Finally apply dyadic-ledger exhaustion, Definition 418.5. It accounts for all dyadic data available to the consumer inside $D, W, A, R, E, H, C, P, N$, so there is no additional coordinate from which a hidden modulus, infinite stream, quotient equality, representative choice, or completeness principle could be read. □

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket cauchy modulus handoff`

Theorem 418.13 (BinaryExpansion scoped dependency factorization)**.** *Let* $B = (D, W, A, R, E, H, C, P, N)$ *be an accepted* BinaryExpansion$^{\uparrow}$ *carrier. Every scoped obligation exported by* BinaryExpansion$^{\uparrow}$ *factors through exactly the displayed packet rows: the* DyadicRatCore$^{\uparrow}$ *binary digit rows in* D, *the* StreamName$^{\uparrow}$ *prefix-window rows in* W, *the* DyadicApproximation$^{\uparrow}$ *partial-sum and tail-radius rows in* A, *the* RegSeqRat$^{\uparrow}$ *handoff row* R, *the* Real$^{\uparrow}$ *seal row* E, *and the shared* hsame, Cont, Pkg, *and* NameCert *routes* H, C, P, *and* N.

Consequently the scoped surface exports no host infinite binary stream, quotient equality of streams, selected binary-expansion representative, endpoint convention resolver, or ambient completeness principle. Any downstream citation of a binary-expansion obligation is a citation of one of the displayed finite rows or of a route between such rows.

Lean4 ✓ `BEDC.Derived.BinaryExpansionUp.BinaryExpansionPacket scoped dependency factorization`

Proof. Start with the accepted carrier of Definition 418.1. Its coordinates already list the only rows made available to the chapter: digits, windows, dyadic approximation rows, the regular handoff, the real seal, and the transport, continuation, package, and local certificate routes.

The prefix-window stability theorem Definition 418.3 keeps every finite prefix read on the displayed D, W, H, and C routes. The dyadic-ledger exhaustion theorem Definition 418.5 then accounts for every dyadic fact by rows in D and A, including the partial sums and tail-radius entries consumed by later readers.

For regular-sequence consumers, apply Definition 418.7. It identifies the handoff row R with the selected prefix window, partial-sum row, and tail-radius row, so no independent rational stream is exported.

For real-name consumers, apply the readback boundary of Definition 418.4 and the seal coherence theorem Definition 418.8. These route every seal citation through the same prefix ledger before reaching E.

The remaining scoped obligations are provenance and certificate obligations. They are exactly the shared hsame, Cont, Pkg, and NameCert routes H, C, P, and N. Since the carrier has no coordinate for the excluded host-level objects, the factorization cannot read them. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinaryExpansion}^\uparrow}$.

Constructive story: A $\mathsf{BinaryExpansion}^\uparrow$ packet is generated from finite binary digit rows, $\mathsf{StreamName}^\uparrow$ prefix windows, $\mathsf{DyadicApproximation}^\uparrow$ partial-sum and tail-radius rows, a $\mathsf{RegSeqRat}^\uparrow$ regular handoff, a $\mathsf{Real}^\uparrow$ seal boundary, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BinaryExpansion}^\uparrow}, \mathsf{publicClosed})$

Scope closed: The public exported surface is the dyadic-window readiness route together with the $\mathsf{Real}^\uparrow$ seal prefix-coherence route for finite dyadic-prefix packets over $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{DyadicApproximation}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BinaryExpansion}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim binary-expansion uniqueness, base-two endpoint convention resolution, quotient equality of streams, selected representatives, host real equality, ambient real completeness, or bridge checking.

Upgrade path: Formal verification requires a checked Lean target for the public dyadic-window readiness route, $\mathsf{Real}^\uparrow$ seal prefix-coherence route, and their displayed carrier support over the same packet.

Chapter 419

A Concrete Naming Certificate for $\mathsf{BinaryEndpointNormalization}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BinaryEndpointNormalization}^{\uparrow}$ packet records the finite base-two endpoint convention that turns the dual terminating and repeating presentations of a dyadic boundary into one displayed normalization route. It is the separate ledger row required by chapter 418: the binary-prefix packet names dyadic readback through finite windows, while this packet names the endpoint-carry convention consumed after that readback. The carrier sits over the dyadic digit surface of chapter 291, the partial-sum and tail-radius surface of chapter 335, the finite-window schedule of chapter 126, the regular rational handoff of chapter 301, and the terminal seal of chapter 38.

Definition 419.1 (BinaryEndpointNormalization carrier)**.** *A* $\mathsf{BinaryEndpointNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, R, K, D, A, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the finite binary prefix row for the terminating-side presentation, including the last displayed* 1 *before a zero tail when such a boundary row is present.*
2. *R is the finite binary prefix row for the repeating-side presentation, including the carried predecessor prefix and the displayed one-tail row when such a boundary row is present.*
3. *K is the endpoint-carry ledger saying that L and R are the two displayed boundary prefixes for the same dyadic endpoint convention.*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit rows used by L, by R, and by the carried predecessor and tail rows named in K.*
5. *A records the* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum and tail-radius rows for the two prefixes and for the common boundary value.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that exposes the two prefixes at the same requested precisions.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the normalized partial-sum ledger.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, D, A, W, Q have been displayed.*

9. *H records componentwise* hsame *transport for the two prefix rows, the endpoint-carry ledger, the dyadic digit rows, the approximation rows, the window schedule, the regular handoff, and the real seal.*
10. *C records the displayed* Cont *replay from a consumer request through W, then D, then K, then A, then Q, and finally S.*
11. *P is the* Pkg *provenance over* $\mathsf{BinaryEndpointNormalization}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BinaryEndpointNormalization}^{\uparrow}}$ *row.*

The carrier stores no host infinite binary stream, quotient equality of streams, selected binary representative, ambient trichotomy, countable-choice selector, or host $\mathsf{Real}^{\uparrow}$ *equality coordinate.*

Lean4 (definition) `BEDC.Derived.BinaryEndpointNormalizationUp.BinaryEndpointNormalizationUp`

Lean4 ✓ `BEDC.Derived.BinaryEndpointNormalizationUp.BinaryEndpointNormalizationTasteGate single carrier alignment`

Theorem 419.2 (BinaryEndpointNormalization NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{BinaryEndpointNormalizatio}}$ *surface for Definition 419.1 consists exactly of carrier exposure for E, dual finite-prefix exposure through L and R, endpoint-carry exposure through K, dyadic digit exposure through D, partial-sum and tail-radius exposure through A, finite-window exposure through W, regular rational handoff through Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through S, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.BinaryEndpointNormalizationUp.BinaryEndpointNormalizationNameCertObligations`

Proof. Project the carrier tuple $E = (L, R, K, D, A, W, Q, S, H, C, P, N)$. The definition displays every row available to the local naming certificate before any consumer can ask for endpoint normalization.

The prefix and carry clauses are finite projections. The rows L and R expose the two boundary presentations, while K records that the terminating-side and repeating-side rows are the paired endpoint convention for the same dyadic boundary.

The dyadic and analytic clauses preserve the same route. The digit ledger D, the approximation ledger A, and the shared windows W feed the regular handoff Q before the real seal S can be read.

Finally, H, C, P, N transport, replay, source, and name the displayed rows. Since the excluded host objects are not coordinates of E, the listed rows exhaust the seed obligation surface. □

Theorem 419.3 (BinaryEndpointNormalization dual-prefix boundary)**.** *Every accepted* BinaryEndpointNormalizati *packet $E = (L, R, K, D, A, W, Q, S, H, C, P, N)$ exposes the terminating and repeating binary boundary presentations through the same finite endpoint-carry ledger. The terminating route reads the displayed prefix L, its* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit rows, and its* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum row. The repeating route reads the displayed prefix R, its carried predecessor digit rows, and its one-tail row through the same D, A, W surface.*

Thus the two boundary presentations meet only at the row K and at the downstream Q, S handoff. The packet exports no quotient of binary streams, no selected host representative, and no ambient real-equality proof.

Lean4 ✓ `BEDC.Derived.BinaryEndpointNormalizationUp.BinaryEndpointNormalization dual prefix boundary`

Proof. Start from the carrier of Definition 419.1. The only prefix rows available to the convention are L and R, and both are read through the same finite window schedule W.

Read the digit and approximation ledgers next. The row D supplies the digit evidence for both displayed prefixes, and A supplies the corresponding partial sums and tail radii. No consumer reaches Q before these rows have been displayed.

The endpoint-carry row K is the unique place where the terminating side and repeating side are paired. It records the carry relation between the last displayed terminating digit and the carried predecessor prefix with one-tail continuation.

The remaining rows H, C, P, N only transport, replay, source, and name this visible route. Therefore the boundary comparison is a finite packet comparison, not a quotient stream comparison or a host real equality claim. □

Theorem 419.4 (BinaryEndpointNormalization RealUp seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{BinaryEndpointNormalization}^{\uparrow}$ *packet factors through the displayed endpoint-carry ledger before reaching the terminal seal. More precisely, a seal consumer must read a shared* $\mathsf{StreamName}^{\uparrow}$ *window in* W*, the two finite prefix rows* L, R*, their* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit ledger* D*, their* $\mathsf{DyadicApproximation}^{\uparrow}$ *partial-sum and tail-radius ledger* A*, the endpoint-carry row* K*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* Q *before it can cite* S*.*

Consequently the terminal seal cannot escape to a host endpoint convention, an infinite binary stream, quotient equality of expansions, a selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BinaryEndpointNormalizationUp.BinaryEndpointNormalizationRealSealNonescape`

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ consumer citation of the seal row S. By the carrier definition, S is downstream of the regular handoff Q and is not an independent real-value coordinate.

The handoff row Q is reached only after the shared window schedule, the two finite prefix rows, and the dyadic approximation ledger have been read. The endpoint-carry row K is part of that route, so the consumer cannot bypass the displayed convention when the two boundary presentations are involved.

Transport and replay do not add further analytic data. The rows H and C preserve and expose the same finite route, while P and N record provenance and local naming.

Since no coordinate of E contains a host stream, quotient equality, selected representative, countable-choice selector, trichotomy branch, or ambient completeness theorem, the seal read is confined to $L, R, K, D, A, W, Q, S, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinaryEndpointNormalization}^{\uparrow}}$.

Constructive story: A $\mathsf{BinaryEndpointNormalization}^{\uparrow}$ packet is generated from two finite binary prefix rows, an endpoint-carry ledger, DyadicRatCore digit rows, DyadicApproximation partial-sum and tail-radius rows, StreamName windows, a RegSeqRat handoff, a terminal RealUp seal, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BinaryEndpointNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 419.1, Definition 419.2, Definition 419.3, and Definition 419.4 seed the finite base-two endpoint convention surface required downstream of BinaryExpansion readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BinaryEndpointNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close binary-expansion uniqueness, quotient equality of streams, selected representatives, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the carrier and prove the NameCert obligation surface, dual-prefix boundary theorem, and RealUp seal nonescape theorem for BEDC.Derived.BinaryEndpointNormalizationUp.

Chapter 420

A Concrete Naming Certificate for BinaryEndpointUniqueness$^{\uparrow}$

Following Bishop and Bridges on constructive real expansions, the BinaryEndpointUniqueness$^{\uparrow}$ packet records the finite determinacy surface that remains after raw binary-prefix readback has passed through the endpoint-normalization ledger. It consumes the binary readback rows of chapter 418 and the endpoint-carry convention of chapter 419, then exposes only the normalized prefix, dyadic ledger, regular handoff, terminal real seal, and local naming routes needed for classifier-facing uniqueness.

Definition 420.1 (BinaryEndpointUniqueness carrier)**.** *A* BinaryEndpointUniqueness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, E, D, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *B is the accepted* BinaryExpansion$^{\uparrow}$ *finite prefix-readback row whose digit windows supply the raw binary observation.*
2. *E is the accepted* BinaryEndpointNormalization$^{\uparrow}$ *endpoint-carry row naming the terminating-versus-repeating boundary convention consumed after B.*
3. *D is the shared* DyadicRatCore$^{\uparrow}$ *digit ledger used by the binary prefix, the endpoint-carry comparison, and the normalized boundary row.*
4. *W is the* StreamName$^{\uparrow}$ *finite-window schedule that exposes the inspected prefixes at the precisions requested by downstream consumers.*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row reached from the normalized dyadic partial sums and tail-radius rows.*
6. *S is the terminal* Real$^{\uparrow}$ *seal row reached only after the binary readback, endpoint normalization, dyadic ledger, window schedule, and regular handoff have been displayed.*
7. *H records componentwise* hsame *transport for B, E, D, W, R, S and for the normalized-prefix comparison row.*
8. *C records the displayed* Cont *replay from a consumer request through W, then B, then E, then D, then R, and finally S.*
9. *P is the* Pkg *provenance over* BinaryEndpointUniqueness$^{\uparrow}$, BinaryExpansion$^{\uparrow}$, BinaryEndpointNormalization$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BinaryEndpointUniqueness}^{\uparrow}}$ *row.*
The carrier predicate accepts exactly finite rows in which two binary presentations can be compared only after the endpoint-normalization row has identified their displayed boundary convention. It has no coordinate for quotient streams, host real equality, a choice-selected representative, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BinaryEndpointUniquenessUp.BinaryEndpointUniquenessTasteGate single carrier alignment`

Theorem 420.2 (BinaryEndpointUniqueness NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{BinaryEndpointUniqueness}^{\uparrow}}$ *surface for Definition 420.1 consists exactly of carrier exposure for U, raw binary readback exposure through B, endpoint-normalized convention exposure through E, shared digit exposure through D, finite-window exposure through W, normalized regular handoff through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through S, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier tuple $U = (B, E, D, W, R, S, H, C, P, N)$. The tuple lists every row available to the local naming certificate before a consumer can ask whether two binary endpoint presentations determine the same normalized seal.

The raw readback and endpoint-convention clauses are separate. The row B supplies only the finite prefix route described by $\mathsf{BinaryExpansion}^{\uparrow}$, while E supplies the endpoint-carry convention described by $\mathsf{BinaryEndpointNormalization}^{\uparrow}$.

The shared analytic clauses then force both routes through the same displayed finite objects. The digit ledger D, the window schedule W, and the regular handoff R are read before the seal S is visible.

Finally, H, C, P, N transport, replay, source, and name this route. Since no coordinate of U contains quotient streams, host real equality, a selected representative, trichotomy, or ambient completeness, the listed rows exhaust the seed obligation surface. □

Theorem 420.3 (BinaryEndpointUniqueness normalized prefix determinacy)**.** *Let $U = (B, E, D, W, R, S, H, C, P,$ be an accepted* $\mathsf{BinaryEndpointUniqueness}^{\uparrow}$ *carrier. If two finite binary prefix presentations are read from the* $\mathsf{BinaryExpansion}^{\uparrow}$ *row B and their endpoint boundary is routed through the* $\mathsf{BinaryEndpointNormalization}^{\uparrow}$ *row E, then the normalized prefix window selected in W determines a single classifier-facing dyadic ledger D and a single downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R.*

The determinacy is scoped to the displayed normalized route. It does not assert quotient equality of all binary streams, host equality of real numbers, trichotomy of arbitrary reals, choice of a preferred representative, or completeness of the ambient $\mathsf{Real}^{\uparrow}$ *surface.*

Proof. Begin with the two finite presentations read through B. By the binary readback boundary of chapter 418, each presentation exposes only finite digit rows, finite windows, dyadic partial-sum rows, a regular handoff boundary, and the displayed transport and continuation routes.

Now apply the endpoint-normalization row E. The endpoint-carry convention of chapter 419 is the only row in the carrier that may relate a terminating boundary presentation to a repeating-tail presentation.

Once the convention is read, the shared schedule W selects the normalized window and the shared ledger D supplies the corresponding digit facts. The continuation row C then routes the consumer to R through those displayed rows.

The transport row H keeps the comparison componentwise, and P, N record the package and local certificate provenance. Therefore the normalized prefix determines the classifier-facing

dyadic ledger and regular handoff inside U, with no appeal to a host stream quotient or external real equality. □

Theorem 420.4 (BinaryEndpointUniqueness RealUp seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{BinaryEndpointUniqueness}^{\uparrow}$ *carrier factors through the normalized endpoint route*

$$B \longrightarrow E \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

inside the finite packet of Definition 420.1. Consequently a consumer citing the seal S can observe only the raw binary readback row, the endpoint-normalization row, the shared finite window, the shared dyadic digit ledger, and the regular handoff named by the carrier.

The seal cannot escape to quotient equality of binary expansions, a host infinite stream, a selected binary representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer citation of the seal row S. By the carrier definition, S is not an independent real-value coordinate; it is downstream of the normalized regular handoff R.

The handoff row R is reached only after the consumer has read the finite binary readback row B, the endpoint-normalization row E, the selected window row W, and the shared digit ledger D. Thus the displayed continuation route prevents a direct jump from raw binary data to the terminal seal.

Transport and provenance do not add analytic data. The row H moves the displayed rows componentwise, C exposes them in order, and P, N record the package and local naming evidence.

Since the carrier has no coordinate for quotient streams, host infinite streams, selected representatives, countable-choice selectors, trichotomy branches, host real equality, or completeness principles, every seal citation is confined to the finite normalized route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinaryEndpointUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{BinaryEndpointUniqueness}^{\uparrow}$ packet is generated from a finite BinaryExpansion readback row, a BinaryEndpointNormalization convention row, DyadicRatCore digit rows, StreamName windows, a RegSeqRat handoff, a terminal RealUp seal, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BinaryEndpointUniqueness}^{\uparrow}}$, seed)

Scope closed: Definition 420.1, Definition 420.2, Definition 420.3, and Definition 420.4 seed the normalized base-two endpoint uniqueness surface between raw binary readback and RealUp classifier determinacy.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BinaryEndpointUniqueness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality of streams, host real equality, selected representatives, choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the carrier and prove the NameCert obligation surface, normalized prefix determinacy theorem, and RealUp seal nonescape theorem for BEDC.Derived.BinaryEndpointUniquenessUp.

Chapter 421

CauchyPairing NameCert Construction

The $\mathsf{CauchyPairing}^{\uparrow}$ interface records the Bishop–Bridges constructive-analysis pairing of two regular Cauchy names as one synchronized finite-window packet. It is a name-certificate source for downstream diagonal, completion, and real-seal chapters: consumers may inspect shared windows, dyadic ledgers, modulus schedules, and the paired $\mathsf{Real}^{\uparrow}$ seal, but they do not receive quotient equality of reals, selected host limits, or ambient completeness.

The two source rows are the paired-name analogue of the finite regular-Cauchy window interface in chapter 371: $\mathsf{CauchyPairing}^{\uparrow}$ consumes two such displayed regular names and synchronizes their window, dyadic-ledger, and seal rows instead of adding a new completed-real coordinate.

Definition 421.1 (Cauchy pairing NameCert source)**.** *A* $\mathsf{CauchyPairing}^{\uparrow}$ *source packet is a finite* BHist *row*

$$K = (A, B, W_A, W_B, L_A, L_B, \mu_A, \mu_B, \mu, E_A, E_B, E, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy names being paired;*
2. *W_A and W_B are finite* $\mathsf{StreamName}^{\uparrow}$ *window schedules for the two names, recorded on a common unary request ledger;*
3. *L_A and L_B are the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledgers, read through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
4. *μ_A and μ_B are the two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, and μ is the synchronized schedule that requests a common finite window large enough for both sides;*
5. *E_A and E_B are the source* $\mathsf{Real}^{\uparrow}$ *seal rows attached to the two regular names;*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the synchronized windows and the two dyadic ledgers;*
7. *H stores the componentwise* hsame *transports for the names, windows, dyadic ledgers, modulus rows, and seal rows;*
8. *C stores the displayed* Cont *routes by which a consumer request reaches the synchronized schedule, both selected windows, both dyadic ledgers, and the paired seal;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their stored transports.

Lean4 (definition) `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier`

Theorem 421.2 (Cauchy pairing NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}$ *surface for the source packet of Definition 421.1 consists of five rows:*

1. *carrier admission: the accepted object is the displayed finite packet K;*
2. *classifier stability: paired packets are compared componentwise by the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, the three* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, and the three* $\mathsf{Real}^{\uparrow}$ *seal rows;*
3. *synchronized-window exactness: every consumer precision request first reads μ, then reads the corresponding windows in W_A and W_B, and only then reads the dyadic ledgers L_A and L_B;*
4. *modulus compatibility: the synchronized schedule μ is accepted only as a finite ledger row that dominates the displayed side schedules μ_A and μ_B on the inspected requests;*
5. *seal non-escape: the paired* $\mathsf{Real}^{\uparrow}$ *seal E is consumed only after both source readbacks, their finite windows, and their dyadic ledgers have been displayed.*

This surface specifies synchronized finite-window pairing and does not assert quotient equality of completed reals or completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier namecert obligations`

Proof. Carrier admission is the projection of the packet K. All names, windows, ledgers, modulus schedules, seal rows, transports, continuation routes, package provenance, and local naming data are stored coordinates.

Classifier stability is componentwise. The displayed hsame rows transport the two regular names, the two window schedules, the two dyadic ledgers, the three modulus rows, and the three seal rows, while the displayed Cont rows preserve the consumer read order.

For synchronized-window exactness, a precision request must pass through μ before either side can be inspected. The two selected window rows are then read through W_A and W_B, and their rational observations are the dyadic ledger rows L_A and L_B.

The modulus compatibility and seal rows use the same finite route. Because the packet has no coordinate for quotient equality, selected host limits, host real equality, or ambient completeness, the paired $\mathsf{Real}^{\uparrow}$ seal cannot export data outside the displayed certificate surface. □

Theorem 421.3 (Cauchy pairing carrier admission)**.** *An accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *carrier is admitted as the finite* BHist *packet*

$$K = (A, B, W_A, W_B, L_A, L_B, \mu_A, \mu_B, \mu, E_A, E_B, E, H, C, P, N)$$

of Definition 421.1. Admission projects the two $\mathsf{RegSeqRat}^{\uparrow}$ *names, side windows, dyadic ledgers, side and synchronized modulus schedules, source and paired* $\mathsf{Real}^{\uparrow}$ *seals, and the transport, continuation, package, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier public rows zero head absurd`

Proof. Project the displayed packet. Every coordinate named in the statement is stored in the same finite BHist row, so the carrier witness is obtained by repacking those projections as the local $\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}$ source.

No separate host object is admitted. The packet contains no quotient equality of reals, no selected host limit, no host real equality proof, and no ambient completeness field. □

Theorem 421.4 (Cauchy pairing classifier transport exactness)**.** *Let K and K' be accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *packets. If the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, and the source and paired* $\mathsf{Real}^{\uparrow}$ *seal rows are matched by displayed* hsame *rows and read through the displayed* Cont *routes, then the classifier transports exactly those packet rows.*

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier classifier transport exactness`

Proof. Apply the classifier surface described by Definition 421.2. The classifier reads the two regular names, window schedules, dyadic ledgers, modulus rows, seal rows, package provenance, and local naming row.

The hsame rows provide the componentwise replacements, and the Cont rows preserve the order in which consumer requests reach the synchronized schedule, side windows, ledgers, and paired seal. Since no other comparison coordinate is present, classifier transport is exact for the displayed rows and cannot compare completed reals or selected limits. □

Theorem 421.5 (Cauchy pairing paired-seal non-escape)**.** *For every accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *carrier, the paired* $\mathsf{Real}^{\uparrow}$ *seal E is consumable only after both finite side windows W_A, W_B and both dyadic ledgers L_A, L_B have been displayed along the synchronized modulus route μ. The seal does not escape through quotient real equality, selected host limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier paired seal non escape`

Proof. Start with a consumer request for the paired seal. The route first reads the synchronized modulus schedule μ, which fixes a common finite request before either side window is inspected.

The request then reads the two side windows W_A and W_B and the corresponding dyadic ledgers L_A and L_B. The side schedules μ_A and μ_B remain attached to this synchronized route through the stored transport and continuation rows.

Only after both finite windows and both dyadic ledgers are displayed can the row E be consumed. Thus the paired seal is a finite certificate endpoint, not a route to quotient equality, a selected host limit, host real equality, or ambient completeness. □

Theorem 421.6 (Cauchy pairing synchronized-window stability)**.** *Let K and K' be accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *source packets. If their two* $\mathsf{RegSeqRat}^{\uparrow}$ *names, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules,* $\mathsf{Real}^{\uparrow}$ *seal rows,* Pkg *provenance, and local* NameCert *rows are transported componentwise by displayed* hsame *rows and read through the displayed* Cont *routes, then every finite consumer request follows matched synchronized windows on both sides and reaches matched paired* $\mathsf{Real}^{\uparrow}$ *seal rows.*

Proof. Project both accepted packets by Definition 421.1. The classifier can inspect only the two regular names, the selected finite windows, the dyadic ledgers, the modulus rows, the seal rows, and the recorded package and naming provenance.

Read a consumer request through the synchronized schedule. The request selects a common window budget before either source name is inspected, so the two side reads remain aligned under the same finite $\mathsf{StreamName}^{\uparrow}$ ledger.

Transport the selected side rows along the stored hsame comparisons and Cont routes. The two dyadic ledgers remain attached to their selected windows, and the paired seal remains downstream of those two readbacks.

Thus matched packets expose matched finite-window routes for every inspected request. The conclusion is a stability statement about the displayed BEDC certificate rows and does not introduce quotient equality, host limits, or an ambient completeness claim. □

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier synchronized window stability`

Theorem 421.7 (Cauchy pairing scoped dependency package)**.** *For the* $\mathsf{CauchyPairing}^{\uparrow}$ *source packet*

$$K = (A, B, W_A, W_B, L_A, L_B, \mu_A, \mu_B, \mu, E_A, E_B, E, H, C, P, N)$$

of Definition 421.1, the scoped dependency surface is exactly this packet over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Every consumer of the paired seal* E *must pass through the synchronized schedule* μ*, both finite windows* W_A, W_B*, both dyadic ledgers* L_A, L_B*, and the stored* hsame *and* Cont *transports before reading* E*. No quotient equality, selected host limit, host real equality, or ambient completeness enters the* $\mathsf{CauchyPairing}^{\uparrow}$ *scope.*

Proof. The carrier source is the finite packet of Definition 421.1. Its displayed coordinates list the two regular names, side windows, dyadic ledgers, side and synchronized modulus schedules, source and paired seal rows, transport rows, continuation routes, package provenance, and local naming row.

Definition 421.2 identifies the NameCert surface carried by those rows. Carrier admission gives the packet itself, classifier stability compares only displayed components, synchronized-window exactness forces every request through μ, and seal non-escape prevents a consumer from reaching E without the two side readbacks.

Definition 421.4 fixes the transport boundary: matched packets may move only along the stored hsame rows and displayed Cont routes. Definition 421.5 then places E strictly downstream of both finite windows and both dyadic ledgers.

Finally, Definition 421.6 shows that every inspected request follows the same synchronized finite-window route on matched packets. Since the packet has no coordinate for quotient equality, selected host limits, host real equality, or ambient completeness, the listed packet is the whole scoped dependency surface. □

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier scoped dependency package`

Theorem 421.8 (Cauchy pairing public export certificate)**.** *The carrier of Definition 421.1, the NameCert obligations of Definition 421.2, synchronized-window stability, paired-seal non-escape, classifier transport exactness, and the scoped dependency package assemble a public* $\mathsf{CauchyPairing}^{\uparrow}$ *certificate over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The certificate exposes exactly the two source names, side windows, dyadic ledgers, synchronized modulus, source and paired* $\mathsf{Real}^{\uparrow}$ *seals, transport routes, package provenance, and local naming row. It exports no quotient equality, selected host limit, host real equality, or ambient completeness.*

Proof. Start with the packet K of Definition 421.1. Its displayed coordinates are precisely the two regular names, side windows, dyadic ledgers, side and synchronized modulus schedules, source and paired seal rows, transport rows, continuation routes, provenance, and local naming row.

Definition 421.2 supplies the local NameCert obligations for that packet. Carrier admission exposes K, synchronized-window exactness routes each consumer request through μ, and seal non-escape keeps the paired $\mathsf{Real}^{\uparrow}$ row downstream of both side windows and both dyadic ledgers.

Definition 421.6 gives the public stability condition for matched finite requests. Definition 421.5 prevents the paired seal from being read without its two source readbacks, and Definition 421.4 fixes the classifier boundary to the displayed hsame and Cont rows.

Finally apply Definition 421.7. The scoped package lists exactly the BEDC rows named in the statement and contains no coordinate for quotient equality, selected host limits, host real equality, or ambient completeness, so the assembled public certificate exports only the displayed finite-row surface. □

Theorem 421.9 (Cauchy pairing meet source)**.** *For the* $\mathsf{CauchyPairing}^{\uparrow}$ *packet*

$$K = (A, B, W_A, W_B, L_A, L_B, \mu_A, \mu_B, \mu, E_A, E_B, E, H, C, P, N)$$

of Definition 421.1, the side moduli μ_A, μ_B *together with the synchronized modulus* μ *determine a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *source row. The source row is read through the same side windows, dyadic ledgers, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *data already present in* K*.*

Proof. Start with the carrier packet of Definition 421.1. Its displayed coordinates contain the two source names, the two side windows, the two dyadic ledgers, the side schedules μ_A, μ_B, and the synchronized schedule μ.

Definition 421.6 forces every inspected request through μ before either side window is read. Thus the synchronized schedule supplies the shared meet row, while μ_A and μ_B remain the two source modulus rows.

The classifier boundary and scoped package then provide the remaining rows. By Definition 421.4 and Definition 421.7, the meet source can use only the displayed hsame, Cont, Pkg, and NameCert rows already in K. □

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier meet source`

Theorem 421.10 (Cauchy pairing shared-bound consumer)**.** *Every shared-bound consumer of an accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *packet reads the common bound through the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row determined in Definition 421.9. In particular, the consumer first reads the synchronized modulus* μ*, then the side schedules* μ_A, μ_B*, side windows* W_A, W_B*, dyadic ledgers* L_A, L_B*, componentwise transports, continuation routes, provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.CauchyPairingUp.CauchyPairingCarrier shared bound consumer`

Proof. Let a shared-bound consumer be given. By Definition 421.2, synchronized-window exactness sends the request through μ before either source side can be inspected.

The meet source theorem identifies the shared-bound row as the finite $\mathsf{CauchyModulusMeet}^{\uparrow}$ source determined by μ_A, μ_B, μ and the side ledger data. Hence the consumer cannot bypass the meet row while still reading a common bound.

Finally apply Definition 421.8. The public certificate exposes only the side names, windows, dyadic ledgers, modulus rows, seal rows, transport routes, package provenance, and local naming row, so every shared-bound read factors through that displayed meet surface. □

Theorem 421.11 (Cauchy pairing Real handoff non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *handoff from an accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *packet may consume the paired seal* E *only after the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *source row, both source readbacks, both finite windows, both dyadic ledgers, componentwise transports, continuation routes, package provenance, and local naming row have been displayed. The handoff exports no quotient equality, selected host limit, host real equality, or ambient completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ handoff route. By Definition 421.5, the paired seal E is downstream of both finite windows and both dyadic ledgers along the synchronized modulus route.

Definition 421.9 places the side moduli and synchronized modulus into one shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ source row. Definition 421.10 then forces any shared-bound read used by the handoff through that meet row.

The remaining exclusions follow from Definition 421.8. The public export surface has no coordinate for quotient equality, selected host limits, host real equality, or ambient completeness, so the $\mathsf{Real}^{\uparrow}$ handoff cannot export them. □

Theorem 421.12 (Cauchy pairing bilinear error ledger)**.** *Let an accepted* $\mathsf{CauchyPairing}^{\uparrow}$ *packet carry two synchronized finite-window sources, shared dyadic bound rows, and the* $\mathsf{Real}^{\uparrow}$ *handoff surface of Definition 421.11. Then it exports a single bilinear error ledger row. The left and right error budgets are read through the same finite-window classifier determined by the synchronized modulus route, the shared-bound rows are read through the same* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *source, and the output route is stable under componentwise* hsame *transport on both source streams. The ledger exposes only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Start with synchronized-window stability from Definition 421.6. It forces matched finite requests on the two source streams to pass through the same synchronized modulus classifier before either side window is inspected.

Next apply the shared-bound consumer theorem of Definition 421.10. The left and right dyadic bound rows are read through the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ source, so the two side budgets are entries of one ledger rather than independent host-side error claims.

Finally use the $\mathsf{Real}^{\uparrow}$ handoff non-escape theorem of Definition 421.11. It keeps the output route downstream of the synchronized finite windows, dyadic ledgers, shared meet row, paired seal, transports, continuations, provenance, and local naming row. Componentwise hsame transport on both streams preserves those displayed rows, and no quotient equality, selected host limit, host real equality, or ambient completeness coordinate is available. □

Theorem 421.13 (Cauchy pairing synchronized-window bridge schema)**.** *The carrier of Definition 421.1, the local* $\mathsf{NameCert}$ *obligations of Definition 421.2, the public export certificate of Definition 421.8, the meet source of Definition 421.9, the shared-bound consumer of Definition 421.10, the bilinear error ledger of Definition 421.12, and the* $\mathsf{Real}^{\uparrow}$ *handoff non-escape theorem of Definition 421.11 assemble the bridge surface from the concrete synchronized packet to the abstract* $\mathsf{CauchyPairing}^{\uparrow}$ *schema. The bridge exposes only* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. Project the synchronized packet K of Definition 421.1. Its visible coordinates are the two regular names, finite windows, dyadic ledgers, side and synchronized modulus schedules, source and paired seal rows, transports, continuations, provenance, and local naming row.

Apply the public export certificate. It identifies the finite synchronized surface that a public consumer may read, while the meet source theorem turns the side moduli and synchronized modulus into the displayed $\mathsf{CauchyModulusMeet}^{\uparrow}$ source row.

Use the bilinear error ledger and then the $\mathsf{Real}^{\uparrow}$ handoff non-escape theorem. These force shared-bound, error-budget, and real-facing reads through the same synchronized finite-window route before any seal row is available.

Thus the abstract $\mathsf{CauchyPairing}^{\uparrow}$ schema receives exactly the rows listed in the statement. The packet contains no coordinate for quotient equality, selected host limits, host real equality, or ambient completeness, so none is exported by the bridge surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyPairing}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, synchronized finite $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers, side and paired $\mathsf{CauchyModulus}^{\uparrow}$ schedules, source and paired $\mathsf{Real}^{\uparrow}$ seal rows, displayed hsame and Cont routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 421.8 exports the public certificate over the displayed packet $K = (A, B, W_A, W_B, L_A, L_B, \mu_A, \mu_B, \mu, E_A, E_B,$ over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyPairing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of reals, host real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, diagonal uniqueness, or bridge checking.

Upgrade path: The next paper-side step is a bridge theorem for the same synchronized finite-window packet and public certificate.

Chapter 422

A Concrete Naming Certificate for $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$

The $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ packet names the finite-partition diagonal surface used by the stationary rational route into the real-completeness and completion ledger. It keeps the rational input rows, finite observation-window partition, diagonal representatives, constant stream classifiers, dyadic observation ledgers, real seal rows, and local naming provenance as displayed BEDC data. The packet contains no quotient stream, choice-selected limit, host real equality, or ambient completeness field.

The packet consumes the stationary rational source, finite stream window, dyadic observation ledger, and real seal through sibling surfaces chapter 37, chapter 126, chapter 291, and chapter 38.

Definition 422.1 (Stationary partition diagonal carrier)**.** *A* $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, \Pi, \Delta, K, D, E, L, C, P, N)$$

whose displayed coordinates are as follows:

1. Q *is the stationary* $\mathsf{Rat}^{\uparrow}$ *input row pair read by the rational source classifier;*
2. Π *is a finite partition of observation windows, including the segment bounds, selected anchors, and refinement routes inspected by the diagonal construction;*
3. Δ *is the finite list of diagonal representatives selected from the partition segments;*
4. K *is the constant* $\mathsf{StreamName}^{\uparrow}$ *classifier row that reads every segment through the same stationary source stream;*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for the segment representatives and their dyadic bounds;*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row receiving the finite diagonal representative;*
7. L *is the* BHist *ledger of displayed segment reads, refinement witnesses, and selected representatives;*
8. C *is the componentwise* hsame *and* Cont *route data used to transport the partition classifier;*
9. P *is the* Pkg *provenance naming the* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and completion-facing dependencies;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{StationaryPartitionDiagonal}^{\uparrow}}$ *provenance row.*

Two accepted carriers are classified together exactly when their rational input rows, partition windows, diagonal representatives, constant stream classifiers, dyadic ledgers, real seal rows, history ledgers, transport routes, package rows, and local certificate rows are transported componentwise by the displayed hsame *comparisons and* Cont *routes.*

Theorem 422.2 (Stationary partition diagonal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryPartitionDiagonal}^{\uparrow}}$ *obligation surface consists of carrier habitation, classifier stability across partition refinement, finite-window exactness, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and public export to the Phase Real and Completion diagonal-limit route.*

Lean4 ✓ `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier namecert obligations`

Proof. Carrier habitation is the finite packaging operation of Definition 422.1. The displayed rows name the stationary rational inputs, the finite partition, the diagonal representatives, the constant stream classifier, the dyadic ledger, the real seal, the history ledger, the transport routes, and the local provenance.

Classifier stability is componentwise. A refinement may subdivide partition segments, but it can only do so through the recorded hsame comparisons and Cont routes; the constant $\mathsf{StreamName}^{\uparrow}$ classifier therefore continues to read the same stationary rational source on each displayed segment.

Finite-window exactness, the seal boundary, and the public export are the three consumer-facing clauses proved below. Together they account for every read made by the local certificate and exclude quotient streams, selected limits, host real equality, hidden representatives, and ambient completeness data from the packet boundary. □

Theorem 422.3 (Stationary partition diagonal window exactness)**.** *For every accepted* $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ *carrier and every displayed segment of its finite partition* Π*, the diagonal representative in* Δ *is read through the same constant* $\mathsf{StreamName}^{\uparrow}$ *classifier row* K*, the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger row* D*, and the segment's displayed* BHist *route in* L*. Refining* Π *preserves the finite observation content through the recorded* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier window exactness`

Proof. Project the accepted carrier through Definition 422.1. A displayed segment of Π carries its bounds, selected anchor, and representative coordinate, and the representative is listed in Δ.

The classifier row K is constant, so every segment read uses the same stationary rational stream. The dyadic ledger D records the rational observation and bound attached to that segment, while L records the visible history route used by the read.

Under partition refinement, each smaller segment points back to a displayed parent segment through the route data C. Since the carrier classifier transports only by hsame and Cont, the refined ledger reads the same finite stationary observation content and introduces no hidden endpoint or host limit. □

Theorem 422.4 (Stationary partition diagonal seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ *carrier factors through its stationary* $\mathsf{Rat}^{\uparrow}$ *source row, finite partition window, diagonal representative, constant* $\mathsf{StreamName}^{\uparrow}$ *classifier,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* BHist *route, displayed* hsame *and* Cont *transport,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier seal boundary`

Proof. Begin with a public seal request and project the accepted carrier. The request can reach E only after choosing one finite partition window and the corresponding diagonal representative in Δ.

By Definition 422.3, that representative is read through K, D, and the displayed history route L. The route data C supplies only componentwise hsame comparisons and Cont witnesses, so transport cannot add a quotient stream or an unrecorded limit object.

Repacking the selected rows with P and N gives the stated factorization of the $\mathsf{Real}^{\uparrow}$ seal. The seal is therefore attached to the finite BEDC packet itself, not to an ambient real-completeness principle. □

Theorem 422.5 (Stationary partition diagonal public export)**.** *The public* $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ *export surface is the finite stationary diagonal packet of Definition 422.1 together with the exact window reads of Definition 422.3 and the seal factorization of Definition 422.4. This surface is the concrete Real and Completion NameCert route consumed by the Phase Real and Completion diagonal-limit construction.*

Lean4 ✓ `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier public export`

Proof. The public packet exposes exactly the carrier rows $Q, \Pi, \Delta, K, D, E, L, C, P, N$. These rows are all finite BEDC rows or local provenance rows.

The exactness theorem accounts for every finite-window read of the partition and every refinement of that finite partition. The seal-boundary theorem accounts for every downstream $\mathsf{Real}^{\uparrow}$ read and keeps it factored through the same visible rows.

Thus a Phase Real or Completion consumer receives a concrete NameCert surface: stationary rational inputs, finite partition windows, diagonal representatives, dyadic observations, real seal rows, history and transport ledgers, package provenance, and local certificate data. No exported coordinate is a quotient stream, selected host limit, hidden representative, or ambient completeness theorem. □

Theorem 422.6 (Stationary partition diagonal bridge readback exactness)**.** *Any standard bridge read of an accepted* $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ *packet recovers exactly the carrier tuple* $S = (Q, \Pi, \Delta, K, D, E, L, C, P, N)$*. The readback proceeds through the displayed partition windows* Π*, diagonal representatives* Δ*, constant* $\mathsf{StreamName}^{\uparrow}$ *classifier* K*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, local* BHist *provenance* L*, componentwise* hsame *and* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*, and it exposes no quotient stream, selected limit, host real equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier bridge readback exactness`

Proof. Project the bridge read through the carrier definition Definition 422.1. The only available carrier rows are $Q, \Pi, \Delta, K, D, E, L, C, P, N$, so the readback has no separate host-real or completion coordinate to return.

Window exactness is Definition 422.3. It identifies each displayed partition segment with its selected representative in Δ, the constant $\mathsf{StreamName}^{\uparrow}$ classifier row K, the dyadic ledger row D, and the local history row L.

The seal boundary Definition 422.4 fixes the $\mathsf{Real}^{\uparrow}$ handoff at E through the same displayed rows and through the recorded hsame and Cont transport data in C. The public export theorem Definition 422.5 then repackages exactly those rows with P and N for Real and Completion consumers.

Since the public export contains only this finite packet, the bridge readback is exact: it recovers S and cannot add a quotient stream, selected limit, host real equality, or ambient completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryPartitionDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{StationaryPartitionDiagonal}^{\uparrow}$ packet is generated from stationary $\mathsf{Rat}^{\uparrow}$ input rows, a finite partition of observation windows, diagonal representatives, a constant $\mathsf{StreamName}^{\uparrow}$ classifier, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledgers, a $\mathsf{Real}^{\uparrow}$ seal row, BHist ledgers, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StationaryPartitionDiagonal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public export is the finite stationary partition diagonal packet with exact window reads, refinement-stable classifier transport, $\mathsf{Real}^{\uparrow}$ seal factorization, and downstream Phase Real and Completion diagonal-limit consumption over the displayed rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StationaryPartitionDiagonal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.StationaryPartitionDiagonalUp.StationaryPartitionDiagonalCarrier\_bridge\_readback\_exa`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient streams, choice-selected limits, host real equality, ambient completeness, or bridge checking.

Upgrade path: Mature closure requires downstream Real and Completion consumer examples over the same stationary finite-partition diagonal packet.

Chapter 423

A Concrete Naming Certificate for CauchyCompletionFunctor$^{\uparrow}$

The CauchyCompletionFunctor$^{\uparrow}$ packet records the functorial completion surface obtained from a source metric presentation and a regular-Cauchy family. Following Bishop and Bridges, *Constructive Analysis*, the packet treats completion as a visible transformation of names: it consumes the RealCauchyCompletion$^{\uparrow}$ seal rows, the CauchyCompletionMonad$^{\uparrow}$ unit-and-bind schedule, and the CompletionUniversalProperty$^{\uparrow}$ extension-uniqueness rows, while its classifier remains confined to finite BHist, hsame, Cont, Pkg, and NameCert evidence.

The functor packet depends substantively on the seal route of chapter 360, the unit-and-bind rows of chapter 378, the extension-uniqueness rows of chapter 397, and the observation route of chapter 611. These are dependency rows for seal, unit/bind, extension-uniqueness, and finite observation service, not informal background.

Definition 423.1 (CauchyCompletionFunctor carrier)**.** *A* CauchyCompletionFunctor$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, R, S, U, B, E, P, T, N)$$

with the following visible rows:

1. *M is the source metric presentation row, including the displayed points, distance observations, and finite tolerance ledger read by the source classifier;*
2. *R is the regular-Cauchy family row, exposing the* RegSeqRat$^{\uparrow}$ *or* RegularCauchyFamily$^{\uparrow}$ *schedules that approximate each source point;*
3. *S is the* RealCauchyCompletion$^{\uparrow}$ *seal row reached from the same source metric and regular-Cauchy family data;*
4. *U records the* CauchyCompletionMonad$^{\uparrow}$ *unit schedule for source points and the bind schedule for completion-facing maps;*
5. *B is the* CompletionUniversalProperty$^{\uparrow}$ *extension-and-uniqueness row attached to the same unit and bind routes;*
6. *E is the classifier-transport ledger comparing two accepted packets only through displayed source rows, regular-Cauchy windows, completion seals, unit schedules, bind schedules, and extension-uniqueness rows;*

7. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyFamily}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
8. *T is the transport row recording the componentwise* hsame *and displayed* Cont *routes for M, R, S, U, B, and E;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctor}^{\uparrow}}$ *ledger.*

The carrier contains no quotient of Cauchy names, no selected limit object, no host-level completed metric space, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorTasteGate single carrier alignment`

Theorem 423.2 (CauchyCompletionFunctor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctor}^{\uparrow}}$ *surface for the carrier of Definition 423.1 consists of six finite rows: carrier admission, classifier transport through displayed source metric and regular-Cauchy rows,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal compatibility,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit schedule exactness,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *bind schedule exactness, and* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-uniqueness ledger exactness. Each row is read from $F = (M, R, S, U, B, E, P, T, N)$ and from the recorded* BHist, hsame, Cont, Pkg, *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorUp StdBridge`

Proof. Carrier admission is projection of the packet in Definition 423.1. The source metric row, regular-Cauchy family row, completion seal, monad schedule, universal-property row, classifier ledger, package provenance, transport row, and local naming ledger are displayed coordinates of one finite history.

Classifier transport uses only those displayed coordinates. A comparison first reads the source metric observations in M, then the regular-Cauchy windows in R, then the completion seal row S reached by the same finite schedule.

The unit, bind, and extension rows are the remaining coordinates U and B, transported by T and accounted for by P and N. Since the carrier has no quotient, selected limit, host completion, or ambient-completeness coordinate, these six rows exhaust the local certificate obligations. □

Theorem 423.3 (CauchyCompletionFunctor unit laws)**.** *For every accepted* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packet $F = (M, R, S, U, B, E, P, T, N)$, the unit part of U sends each displayed source metric point of M to the completion seal row S by the regular-Cauchy family route R. The left and right unit readings available to consumers are therefore the same finite route through M, R, S, U, P, T, and N, not an equality of host completion objects.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket unit laws`

Proof. Project the unit schedule from U. Its source is the displayed metric row M, and its target is the seal row S reached through the regular-Cauchy family row R.

The left-unit read begins with a source point and immediately follows the unit schedule into the displayed completion seal. The right-unit read begins with the same displayed completion-facing row and reads back through the stored source and regular-Cauchy schedule.

Both reads use the same package and transport evidence, namely P, T, and N. Thus the unit laws are certificate-level route equalities inside the packet, with no appeal to quotient completion equality or selected limits. □

Theorem 423.4 (CauchyCompletionFunctor multiplication laws)**.** *For every accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet, the bind part of the* CauchyCompletionMonad$^{\uparrow}$ *schedule composes two completion-facing reads only by reusing the displayed regular-Cauchy windows, the* RealCauchyCompletion$^{\uparrow}$ *seal rows, and the* CompletionUniversalProperty$^{\uparrow}$ *extension row. Associativity is therefore witnessed as equality of the finite bind ledger over R, S, U, B, E, P, T, and N.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket multiplication laws`

Proof. Read the bind schedule from U. A first completion-facing read consumes the regular-Cauchy family row R and reaches the seal row S; a second read is permitted only through the same kind of displayed seal and extension row.

The universal-property coordinate B records the extension route used by each bind. The classifier ledger E keeps both binds attached to the same source metric and regular-Cauchy windows, and the transport row T moves those rows componentwise.

Associativity follows by comparing the two parenthesized bind ledgers as finite row lists. They name the same R, S, U, B, E, P, T, and N coordinates and introduce no hidden completed carrier, quotient stream, selected limit, or ambient completeness principle. □

Theorem 423.5 (CauchyCompletionFunctor ledger exactness)**.** *Every consumer route exported by an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet is exhausted by the source metric row, regular-Cauchy family row,* RealCauchyCompletion$^{\uparrow}$ *seal row,* CauchyCompletionMonad$^{\uparrow}$ *unit-and-bind schedule,* CompletionUniversalProperty$^{\uparrow}$ *extension-uniqueness row, classifier transport ledger, package provenance, transport row, and local* NameCert *ledger. No consumer can obtain a quotient completion, a selected limit, a host completed metric space, or an ambient completeness theorem from the packet.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket ledger exactness`

Proof. Start with a consumer route and project the carrier coordinates of Definition 423.1. The only available rows are M, R, S, U, B, E, P, T, and N.

The NameCert obligation theorem accounts for carrier, classifier, completion seal, unit, bind, and universal-property rows. The unit laws identify the source-point route into the completion seal, while the multiplication laws identify the composed completion-facing route.

These rows exhaust the packet ledger. Since the carrier definition contains no additional coordinate, the consumer has no path to quotient completion equality, selected limits, host completed metric spaces, or ambient completeness. □

Theorem 423.6 (CauchyCompletionFunctor classifier transport)**.** *For two accepted* CauchyCompletionFunctor$^{\uparrow}$ *packets, classifier replacement is local to the displayed source metric row, the regular-Cauchy family row, the* RealCauchyCompletion$^{\uparrow}$ *seal, the* CauchyCompletionMonad$^{\uparrow}$ *unit schedule, the* CauchyCompletionMonad$^{\uparrow}$ *bind schedule, the* CompletionUniversalProperty$^{\uparrow}$ *row, the classifier ledger, the* hsame *rows, the displayed* Cont *rows, the* Pkg *provenance, and the local* NameCert *rows. A replacement that agrees on these coordinates cannot alter any exported consumer route, and a replacement that changes any exported route must be witnessed by one of these coordinates.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket classifier transport`

Proof. Project the two packets through Definition 423.1. The only displayed coordinates are $M, R, S, U, B, E, P, T, N$, and the local classifier reads them componentwise through the hsame and Cont routes recorded in T.

The source metric and regular-Cauchy rows determine the completion seal row. The unit and bind schedules are the two monad-facing parts of U, and the extension-uniqueness row is the public content of B.

Thus every accepted classifier replacement factors through the displayed ledger rows E, P, T, N. Since the carrier contains no hidden completed object or selected limit coordinate, there is no further place where a replacement can change a consumer route. □

Theorem 423.7 (CauchyCompletionFunctor universal-property extension exactness)**.** *Every extension-uniqueness consumer of an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet factors through the displayed* CompletionUniversalProperty$^{\uparrow}$ *row together with the* RealCauchyCompletion$^{\uparrow}$ *and* CauchyCompletionMonad$^{\uparrow}$ *rows. In particular, extension existence, extension uniqueness, unit compatibility, and bind compatibility are read from B only after the seal row S and the unit-and-bind row U have been projected from the same carrier.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket universal property extension exactness`

Proof. Begin with an extension-uniqueness consumer and project the carrier $F = (M, R, S, U, B, E, P, T, N)$. The only universal-property coordinate is B, so the consumer must first read the completion seal S and the monad schedule U that give B its displayed source and target routes.

Existence and uniqueness are then finite ledger reads from the CompletionUniversalProperty$^{\uparrow}$ row. Unit compatibility uses the unit part of U, while bind compatibility uses the bind part of U together with the same seal row S.

The classifier and package rows E, P, T, N record that these reads remain attached to the displayed source metric and regular-Cauchy family rows. No additional completion object, quotient equality, or ambient completeness theorem is available to the consumer. □

Theorem 423.8 (CauchyCompletionFunctor obligation closure package)**.** *The accepted* CauchyCompletionFunctor$^{\uparrow}$ *carrier assembles a five-row obligation surface: carrier admission for $F = (M, R, S, U, B, E, P, T, N)$; existing local* NameCert *obligations for carrier, classifier, seal, unit, bind, and universal-property rows; unit laws for the displayed source-to-seal route; multiplication laws for the finite bind ledger; and ledger exactness for the classifier transport and universal-property extension rows of Definition 423.6 and Definition 423.7. This surface is an obligation package over the displayed packet, not a host-level completion construction.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket obligation closure package`

Proof. Carrier admission is the tuple displayed in Definition 423.1. The existing NameCert obligations project the local carrier row, componentwise classifier, completion seal, unit schedule, bind schedule, and universal-property ledger from the same tuple.

The unit laws and multiplication laws provide the two monad rows. The unit row follows source points through M, R, S, U, and the multiplication row compares two bind readings through R, S, U, B, E.

It remains to account for escape from the obligation surface. The classifier transport theorem confines replacements to the displayed source, Cauchy-family, seal, unit, bind, universal-property, transport, package, and local certificate rows. The universal-property exactness theorem confines extension consumers to B together with S and U.

These five rows exhaust the packet coordinates. Therefore the obligation package is closed at the displayed carrier surface and introduces no quotient completion, selected limit, host completed metric space, or ambient completeness principle. □

Theorem 423.9 (CauchyCompletionFunctor source metric unit handoff)**.** *Every displayed source metric point admitted by an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet enters the completion seal only through* M, R, S, U, P, T, N*. The handoff reads the source metric presentation in* M*, the regular-Cauchy family route in* R*, the* RealCauchyCompletion$^{\uparrow}$ *seal in* S*, the unit part of the* CauchyCompletionMonad$^{\uparrow}$ *schedule in* U*, and the package, transport, and local naming rows* P, T, N*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket source metric unit handoff`

Proof. Project the carrier $F = (M, R, S, U, B, E, P, T, N)$. A source metric point is visible only as part of the source row M.

The unit schedule is the component of U that sends that displayed point through the regular-Cauchy route R into the seal row S. The package, transport, and local naming rows P, T, N record that this is the admitted public route.

No coordinate of the packet names a selected limit, quotient completion object, or host completed metric space. Thus the source point can enter the completion surface only through the displayed unit handoff. □

Theorem 423.10 (CauchyCompletionFunctor completion consumer handoff)**.** *Every completion-facing consumer of an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet reads* RealCauchyCompletion$^{\uparrow}$*,* CauchyCompletionMonad$^{\uparrow}$*, and* CompletionUniversalProperty$^{\uparrow}$ *through the same finite packet* $F = (M, R, S, U, B, E, P, T, N)$*. The consumer cannot read a completion seal without the unit-and-bind schedule, nor an extension-uniqueness row without the seal and monad rows that give it source and target routes.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket completion consumer handoff`

Proof. Begin with a completion-facing consumer and project the packet. The only completion seal is S, the only monad schedule is U, and the only universal-property row is B.

The consumer first reads S through the source and regular-Cauchy rows M, R. It then reads the unit and bind schedules from U, and only then reads extension and uniqueness data from B.

Classifier, package, transport, and local naming rows E, P, T, N keep these reads attached to the same packet. Since no other completion-facing coordinate exists, every such consumer is handed the same finite carrier surface. □

Theorem 423.11 (CauchyCompletionFunctor bind ledger consumer exhaustion)**.** *Every composed completion read exported by an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet is exhausted by the bind ledger* R, S, U, B, E, P, T, N*. The ledger records the regular-Cauchy windows, completion seals, unit-and-bind schedule, extension-uniqueness row, classifier transport, package provenance, componentwise transport, and local* NameCert *row; it exports no hidden quotient equality, selected limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket bind ledger consumer exhaustion`

Proof. Take a composed completion read. Its first and second completion-facing stages must use the bind part of U together with the regular-Cauchy windows in R and the seal rows in S.

The universal-property coordinate B supplies the extension route attached to the bind read. The classifier ledger E and the package, transport, and naming rows P, T, N record that the composed read remains inside the same finite carrier.

Those rows are exactly the bind-facing coordinates of F. Therefore the consumer is exhausted by R, S, U, B, E, P, T, N and has no access to quotient equality, selected limits, host completion objects, or ambient completeness. □

Theorem 423.12 (CauchyCompletionFunctor public completion export)**.** *Every accepted* CauchyCompletionFunctor *packet exports a single public completion surface through the displayed source metric row M, regular-Cauchy family row R,* RealCauchyCompletion$^{\uparrow}$ *seal S,* CauchyCompletionMonad$^{\uparrow}$ *unit-and-bind schedule U,* CompletionUniversalProperty$^{\uparrow}$ *row B, classifier ledger E, and transport and provenance rows P, T, N. Consumers obtain exactly this finite surface and obtain no quotient completion, selected limit, host completed metric space, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket public completion export`

Proof. Project the accepted carrier as $F = (M, R, S, U, B, E, P, T, N)$. The NameCert obligation rows admit the carrier, bind the classifier to the displayed source and regular-Cauchy rows, attach the completion seal, and expose the unit, bind, and universal-property rows from the same finite packet.

The unit laws send displayed source points through M, R, S, U, P, T, N. The multiplication laws compare composed completion reads through the bind ledger, and the universal-property extension exactness theorem confines extension and uniqueness consumers to B after S and U have been read.

It remains to account for public export. The completion consumer handoff forces every completion-facing consumer to use the same S, U, B rows over the same carrier, and the bind-ledger consumer exhaustion theorem confines composed reads to R, S, U, B, E, P, T, N.

Since the carrier has no additional coordinate, this finite row surface is the only public completion export. No consumer can extract quotient completion equality, a selected limit, a host completed metric space, or an ambient completeness theorem from the packet. □

Theorem 423.13 (CauchyCompletionFunctor observation-budget factorization)**.** *Let $F = (M, R, S, U, B, E, P, T,$ be an accepted* CauchyCompletionFunctor$^{\uparrow}$ *packet. Suppose its completion-facing seal rows are served by an accepted* RealObservationBudget$^{\uparrow}$ *packet $B_R = (E_R, W_R, D_R, R_R, S_R, H_R, C_R, P_R, N_R)$. Every public completion consumer of F factors first through the observation-budget route*

$$E_R \longmapsto W_R \longmapsto D_R \longmapsto R_R \longmapsto S_R$$

before reading the RealCauchyCompletion$^{\uparrow}$*, monad, or universal-property rows. The route exposes no quotient completion, selected limit, host completed metric space, ambient completeness theorem, host real equality, or hidden observation schedule.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket observation budget factorization`

Proof. Apply Definition 423.12. It confines public completion consumers of F to the displayed source row, regular-Cauchy row, completion seal, monad schedule, universal-property row, classifier ledger, transport rows, provenance rows, and local naming row.

Now use Definition 611.20. The accepted observation-budget packet supplies the finite route from the external observation row E_R through the window, dyadic, rational, and seal rows W_R, D_R, R_R, S_R before any completion-facing seal can be read.

Finally apply Definition 611.21. That package keeps the seal service attached to the same finite observation route and prevents a consumer from replacing it by a host completion object or unrecorded observation schedule.

Combining these readings, every public completion consumer reaches the rows S, U, B of F only after passing through E_R, W_R, D_R, R_R, S_R. Since neither packet has a coordinate for quotient completion equality, selected limits, host completed metric spaces, ambient completeness, host real equality, or hidden schedules, those objects do not enter the consumer route. □

Theorem 423.14 (CauchyCompletionFunctor observation-budget seal determinacy)**.** *Let $F = (M, R, S, U, B, E, P, T, N)$ be an accepted* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packet, and let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet $B_R = (E_R, W_R, D_R, R_R, S_R, H_R, C_R, P_R, N_R)$ serve the completion-facing seal rows of F. If two public completion consumers display the same observation route*

$$E_R \longmapsto W_R \longmapsto D_R \longmapsto R_R \longmapsto S_R$$

and read the same rows of F, then they determine the same completion seal S, the same unit-and-bind schedule U, and the same universal-property extension row B. The determination is only through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it exports no quotient completion, selected limit, host equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket observation budget seal determinacy`

Proof. First apply Definition 423.13. Both consumers must enter the completion-facing rows of F through the same displayed route E_R, W_R, D_R, R_R, S_R before S, U, B can be read.

Next apply Definition 423.12. Once the route has reached F, the only public completion surface is the displayed packet $M, R, S, U, B, E, P, T, N$, so the seal, unit-and-bind schedule, and extension row cannot be replaced by another completion-facing coordinate.

Use the finite-route and seal-ledger rows from the observation budget. Definition 611.20 keeps the route on E_R, W_R, D_R, R_R, S_R, and Definition 611.27 exhausts the seal ledger by $E_R, W_R, D_R, R_R, S_R, H_R, C_R, P$

Therefore two consumers with the same displayed route and the same F rows determine the same S, U, B rows. The non-escape clauses in the cited export and ledger theorems remove quotient completion, selected limits, host equality, and ambient completeness from the route, so no hidden coordinate can separate the two determinations. □

Theorem 423.15 (CauchyCompletionFunctor RegSeqRat-to-Real completion route)**.** *Let $F = (M, R, S, U, B, E, P, T, N)$ be an accepted* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packet. Suppose the regular row R is served by a* $\mathsf{RegSeqRat}^{\uparrow}$ *route and the completion seal S is served through* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* $\mathsf{RealObservationBudget}^{\uparrow}$*. Then every public completion consumer factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionMonad}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow} \longmapsto \mathsf{Real}$$

using the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of F. The route exports no quotient completion, selected limit, host completed metric space, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket regseq real endofunctor`

Proof. Apply Definition 423.12. It identifies the public completion surface of F with the displayed source row M, regular-Cauchy row R, completion seal S, monad schedule U, universal-property row B, classifier ledger E, and transport, provenance, and naming rows P, T, N.

Next apply Definition 423.10. The consumer can reach the completion-facing rows only through the same S, U, B coordinates over the same finite packet.

Use Definition 423.11 to keep composed completion reads inside R, S, U, B, E, P, T, N. The assumption that R is served by $\mathsf{RegSeqRat}^{\uparrow}$ and S by $\mathsf{RealCauchyCompletion}^{\uparrow}$ places the regular rational sequence route before the real completion seal, while U and B provide the monad and universal-property stages.

Finally apply Definition 423.13. The real observation-budget packet is read as the finite budget service for the completion seal, so the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows carry the route through $\mathsf{RealObservationBudget}^{\uparrow}$.

The carrier projection closes non-escape: F has no coordinate for quotient completion equality, selected limits, host completed metric spaces, or ambient completeness. Therefore no public completion consumer can export those objects outside the displayed finite route. □

Theorem 423.16 (CauchyCompletionFunctor Real-seal functoriality)**.** *Let*

$$F = (M, R, S, U, B, E, P, T, N)$$

and

$$F' = (M', R', S', U', B', E', P', T', N')$$

be accepted $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packets. Suppose they agree on the source metric row, regular-Cauchy row,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal, monad schedule, universal-property row, classifier ledger, transport row, provenance row, and local* $\mathsf{NameCert}$ *row. Then every* $\mathsf{RealObservationBudget}^{\uparrow}$*-served public completion consumer determines the same Real seal, the same unit-and-bind schedule, and the same universal-property row for F and F'.*

The functorial read is the displayed route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}$ $\mathsf{CompletionUniversalProperty}^{\uparrow}$, *and* $\mathsf{RealObservationBudget}^{\uparrow}$. *It has no distinguishing coordinate for quotient completion, selected limit, host completed metric space, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket real seal functoriality`

Proof. Project both functor packets through Definition 423.1. Under the displayed row agreement, the completion-facing coordinates in the two packets are the same source metric row, regular-Cauchy row, Real seal, monad schedule, universal-property row, classifier ledger, transport row, provenance row, and local naming row.

Use Definition 423.12 and Definition 423.13. The first theorem identifies the public completion surface, and the second forces every $\mathsf{RealObservationBudget}^{\uparrow}$-served read through the same finite budget route before S, U, B can be consumed.

Next apply Definition 423.10 and Definition 423.11. These keep the completion handoff, unit-and-bind schedule, universal-property row, and composed bind ledger inside the displayed R, S, U, B, E, P, T, N route.

Finally invoke Definition 423.15. It identifies the RegSeqRat-to-Real route through $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, and $\mathsf{RealObservationBudget}^{\uparrow}$. Since quotient completion, selected limits, host completed metric spaces, and ambient completeness have no carrier coordinate, no public consumer can distinguish the two Real-seal reads. □

Theorem 423.17 (CauchyCompletionFunctor finite approximation route)**.** *Let $F = (M, R_F, S_F, U_F, B_F, E_F, P_F,$ be an accepted* CauchyCompletionFunctor$^\uparrow$ *packet, and let $A = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* FiniteCauchyApproximation$^\uparrow$ *packet serving the regular row R_F. Suppose the functor route also names the* RegSeqRat$^\uparrow$, StreamName$^\uparrow$, RealCauchyCompletion$^\uparrow$, CauchyCompletionMonad$^\uparrow$, CompletionUniversal *and* RealObservationBudget$^\uparrow$ *rows used by the completion-facing consumer. Then every public completion consumer of F first reads the finite approximation rows*

$$R, n, k, W, D, E, S, H, C, P, N$$

and only then reads the RegSeqRat$^\uparrow$, StreamName$^\uparrow$, RealCauchyCompletion$^\uparrow$, CauchyCompletionMonad$^\uparrow$, CompletionUniversalProperty$^\uparrow$, *and* RealObservationBudget$^\uparrow$ *rows already named by the functor route. The composed route exports no quotient completion, selected limit, host completed metric space, ambient completeness theorem, or hidden finite-observation schedule.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket finite approximation route`

Proof. Start from Definition 423.15. It keeps every public completion consumer of F on the displayed route through the regular rational sequence row, the real completion seal, the monad schedule, the universal-property row, and the real observation-budget service.

Now insert the accepted finite approximation packet at the regular row. By Definition 601.4, any visible dyadic observation and any real-facing handoff from A first reads R, n, k, W, D, E before the handoff row can be used. By Definition 601.7, the same consumer then reaches the real-facing row S only after the transport, continuation, provenance, and local naming rows H, C, P, N have also been exposed.

Finally apply Definition 601.16. It packages the finite rows $R, n, k, W, D, E, S, H, C, P, N$ as the local NameCert$_{\text{FiniteCauchyApproximation}^\uparrow}$ service boundary, so the functor route cannot read the RegSeqRat$^\uparrow$, StreamName$^\uparrow$, RealCauchyCompletion$^\uparrow$, CauchyCompletionMonad$^\uparrow$, CompletionUniversalProperty$^\uparrow$, or RealObservationBudget$^\uparrow$ rows before the finite approximation service has been consumed.

The cited functor and finite-approximation boundaries both lack coordinates for quotient completion, selected limits, host completed metric spaces, ambient completeness, and hidden finite-observation schedules. The composed public route therefore exposes only the displayed finite approximation service followed by the completion route already named by F. □

Theorem 423.18 (CauchyCompletionFunctor tail-budget coherence factorization)**.** *Let $F = (M, R, S, U, B, E, P,$ be an accepted* CauchyCompletionFunctor$^\uparrow$ *packet, and let a* TailBudgetCoherence$^\uparrow$ *packet serve the finite Real-completion budget route. Any public completion consumer using both packets must first read the route through the TailMeet threshold, observation budget, selector budget,* StreamName$^\uparrow$ *window,* DyadicRatCore$^\uparrow$ *ledger,* RegSeqRat$^\uparrow$ *readback,* RealTailAgreementSeal$^\uparrow$, *and* CauchyLimitSeal$^\uparrow$ *before the functor rows S, U, B are exposed. The functor packet exports no quotient completion equality, selected limit, host completed metric space, ambient completeness theorem, or unbudgeted completion selector.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket tail budget coherence factorization`

Proof. Begin with Definition 423.12. It confines every public completion consumer to the displayed source, regular-Cauchy, completion seal, monad schedule, universal-property row, classifier ledger, transport, provenance, and naming rows of F.

Next use Definition 423.13. The completion-facing seal must already pass through the finite observation-budget route before the public completion rows can be read.

Combine this with Definition 423.11. Composed completion reads remain inside R, S, U, B, E, P, T, N, so the consumer cannot insert a second completion selector between the budget route and the functor packet.

Finally apply the public export boundary of $\mathsf{TailBudgetCoherence}^{\uparrow}$. Its finite Real-completion route supplies the TailMeet threshold, observation budget, selector budget, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ in front of the same S, U, B rows. Since neither packet carries quotient completion equality, selected limits, host completed metric spaces, ambient completeness, or an unbudgeted selector, the displayed route exhausts the public completion handoff. □

Theorem 423.19 (CauchyCompletionFunctor finite-observation selector factorization)**.** *Let $F = (M, R, S, U, B, E, P, T, N)$ be an accepted* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packet, and let $Q = (B_Q, S_Q, W_Q, D_Q, R_Q, E_Q,$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet serving the same completion-facing seal rows. Every public completion consumer of F factors first through the selector route*

$$B_Q \longmapsto S_Q \longmapsto W_Q \longmapsto D_Q \longmapsto R_Q \longmapsto E_Q$$

before reading the functor rows S, U, B from F. The composed route exports no quotient completion equality, selected limit, host completed metric space, ambient completeness theorem, host real equality, countable choice principle, or hidden selector.

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket finite observation selector factorization`

Proof. Begin with Definition 423.12. It confines every public completion consumer of F to the displayed source metric row, regular-Cauchy row, completion seal, monad schedule, universal-property row, classifier ledger, transport rows, provenance rows, and local naming row.

Next apply Definition 423.13. The completion-facing seal cannot be read directly from F; it first passes through the finite observation-budget service attached to the same seal-facing rows.

Use Definition 423.18 to keep the budgeted completion route coherent with the TailMeet threshold, StreamName window, dyadic ledger, RegSeqRat readback, tail-agreement seal, and Cauchy-limit seal. This prevents a consumer from inserting an unbudgeted completion selector between the observation route and the functor rows.

Finally apply Definition 782.10. The accepted selector packet Q supplies the diagonal finite-observation route $B_Q, S_Q, W_Q, D_Q, R_Q, E_Q$ before any completion-facing seal row is opened. Since the selector packet and the functor packet contain no coordinates for quotient completion equality, selected limits, host completed metric spaces, ambient completeness, host real equality, countable choice, or hidden selectors, every public completion consumer reaches S, U, B only through the displayed selector route. □

Theorem 423.20 (CauchyCompletionFunctor selector-budget determinacy)**.** *Let $F = (M, R, S, U, B, E, P, T, N)$ be an accepted* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *packet, and let $Q = (B_Q, S_Q, W_Q, D_Q, R_Q, E_Q, H_Q, C_Q, P_Q, N_Q)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet serving the same completion-facing seal rows. Suppose two public completion consumers have the same selector route*

$$B_Q \longmapsto S_Q \longmapsto W_Q \longmapsto D_Q \longmapsto R_Q \longmapsto E_Q$$

and read the same displayed source, regular-Cauchy, classifier, transport, provenance, and naming rows of F. Then the two consumers determine the same completion seal S, the same unit-and-bind schedule U, and the same universal-property row B.

The determined rows are only the displayed rows of F reached through the finite selector route. No consumer thereby obtains quotient completion equality, selected limits, host completed metric spaces, ambient completeness, host real equality, countable choice, or a hidden selector.

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket selector budget determinacy`

Proof. Apply Definition 423.19. Both consumers must pass through the same accepted selector route $B_Q, S_Q, W_Q, D_Q, R_Q, E_Q$ before reading the completion-facing rows of F.

Next use Definition 423.12. Once the selector route is fixed, the public completion surface of F exposes only $M, R, S, U, B, E, P, T, N$, so the completion seal, unit-and-bind schedule, and universal-property row have no second public source.

Apply Definition 423.11 to the composed reads. The monadic continuation ledger can consume only the displayed functor rows already reached through the selector route, and it cannot add a separate schedule or universal-property witness.

Finally apply Definition 423.14. The common finite observation route fixes the seal-facing read of S; with S fixed, the public export and bind-ledger exhaustion fix U and B as the corresponding displayed rows of F.

Carrier non-escape supplies the exclusions. Neither the selector packet nor the completion-functor packet contains coordinates for quotient completion equality, selected limits, host completed metric spaces, ambient completeness, host real equality, countable choice, or hidden selectors, so no such object can distinguish the two consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ packet is generated from a source metric row, regular-Cauchy family rows, a $\mathsf{RealCauchyCompletion}^{\uparrow}$ seal, a $\mathsf{CauchyCompletionMonad}^{\uparrow}$ unit-and-bind schedule, a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ extension-uniqueness row, classifier transport, package provenance, transport routes, and a local naming ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctor}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 423.12, Definition 423.13, Definition 423.3, Definition 423.4, Definition 423.6, and Definition 423.7 bind the public completion export, observation-budget factorization, unit laws, multiplication laws, classifier transport, and universal-property extension exactness as the mature treatment.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctor}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CauchyCompletionFunctorUp.CauchyCompletionFunctorPacket\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient completion construction, equality of Cauchy classes, selected limits, host completed metric spaces, ambient completeness, or bridge checking.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 424

A Concrete Naming Certificate for $\mathsf{ProductMetric}^{\uparrow}$

The $\mathsf{ProductMetric}^{\uparrow}$ interface records the ordinary product-metric construction as a finite BEDC certificate. The human reference is the Kelley-style product metric on a binary product, but the certificate exposes only carrier rows, classifier rows, distance ledgers, projection routes, package provenance, and a local naming row. It does not assert a quotient product, host pair equality, a selected real limit, or classical choice data.

This certificate consumes the sibling rows of chapter 57, chapter 38, and chapter 34. The $\mathsf{Met}^{\uparrow}$ component classifiers, $\mathsf{Real}^{\uparrow}$-valued distance witness rows, and $\mathsf{Prod}^{\uparrow}$ projection rows are read before the product-distance ledger is available to any $\mathsf{ProductMetric}^{\uparrow}$ consumer. The product-metric packet transports those three sibling surfaces through its finite BHist ledger rather than introducing a quotient product, selected host limit, or host pair-equality route.

Definition 424.1 (ProductMetric BHist carrier)**.** *A* $\mathsf{ProductMetric}^{\uparrow}$ *packet is a finite* BHist *row*

$$Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$$

with the following displayed coordinates:

1. *X and Y are two accepted* $\mathsf{Met}^{\uparrow}$ *carrier and classifier rows, including their visible source histories and local metric-space naming rows;*
2. *d_X and d_Y are the* $\mathsf{Real}^{\uparrow}$*-valued distance witness rows consumed from the two component* $\mathsf{Met}^{\uparrow}$ *packets;*
3. *Π is the displayed* $\mathsf{Prod}^{\uparrow}$ *pairing row whose first and second projections read the two component histories;*
4. *δ is the chosen product-distance ledger, either the displayed max row for the two component distances or a displayed dyadic-sum bound row for them;*
5. *H stores componentwise* hsame *transports for both component carriers, both component classifiers, both distance witnesses, the* $\mathsf{Prod}^{\uparrow}$ *row, and the product-distance ledger;*
6. *C stores the* Cont *routes for the two projections, the component distance reads, the product-distance ledger read, and the local product-metric consumer route;*
7. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ *row.*

The carrier is this finite packet. It contains no quotient product, no host pair equality, no selected real limit, and no classical choice field.

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricTasteGate single carrier alignment`

Theorem 424.2 (ProductMetric NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ProductMetric}^\uparrow}$ *surface for the carrier of Definition 424.1 consists of five finite obligations: carrier admission for Q, componentwise classifier transport along the two* $\mathsf{Met}^\uparrow$ *classifiers and the* $\mathsf{Prod}^\uparrow$ *pairing row, product-distance ledger exactness for δ, projection non-escape through the two displayed* Cont *routes, and package provenance through P and N. These obligations are read only from* $\mathsf{Met}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{Prod}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Carrier admission is the projection of the finite packet Q. The two component metric carriers, their classifiers, their real-valued distance witnesses, the product pairing row, the chosen distance ledger, the transport rows, the continuation rows, package provenance, and local naming row are all stored coordinates.

Classifier transport is componentwise. A classified change in the first component is read through the first $\mathsf{Met}^\uparrow$ classifier and the first projection of Π, while a classified change in the second component is read through the second classifier and the second projection of Π.

The product-distance ledger δ is then consumed after the two component distance witnesses have been displayed. Whether δ is represented as max or as a displayed dyadic-sum bound, it is a BEDC ledger row over the two visible $\mathsf{Real}^\uparrow$ distances, not a host-level metric proof.

The package and naming rows record that these obligations are the public surface. Since no coordinate of Q is a quotient product, host pair equality, selected real limit, or classical choice field, none of those data can be exported by the certificate. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier namecert obligations`

Theorem 424.3 (ProductMetric classifier stability)**.** *Let Q and Q' be accepted* $\mathsf{ProductMetric}^\uparrow$ *packets. If their two* $\mathsf{Met}^\uparrow$ *component carrier rows, classifiers,* $\mathsf{Real}^\uparrow$ *distance witness rows,* $\mathsf{Prod}^\uparrow$ *pairing rows, product-distance ledgers,* Pkg *provenance rows, and local* NameCert *rows are transported componentwise by the displayed* hsame *rows and read through the displayed* Cont *routes, then the* $\mathsf{ProductMetric}^\uparrow$ *classifier transports exactly those packet rows.*

Proof. Unpack Q and Q' by Definition 424.1. The only carrier data available to the classifier are the two metric components, the two distance witnesses, the product row, the distance ledger, and the recorded transport, continuation, provenance, and naming rows.

Transport the first component through the first $\mathsf{Met}^\uparrow$ classifier and the first projection of Π. Transport the second component through the second classifier and the second projection of Π.

The two distance witness rows transport with their source component rows, and the product-distance ledger transports after those two witnesses are matched. Thus the classifier is stable exactly on the displayed BEDC packet, with no appeal to host pair equality or quotient product equality. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier classifier stability`

Theorem 424.4 (ProductMetric component distance transport)**.** *Let* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ *be an accepted* $\mathsf{ProductMetric}^\uparrow$ *packet. Componentwise classifier transport of the two* $\mathsf{Met}^\uparrow$ *rows transports the two* $\mathsf{Real}^\uparrow$ *distance witness rows d_X and d_Y along the displayed* hsame *rows in H and the component distance reads in C. The product-distance ledger δ then reads exactly those transported component distances.*

Proof. Project the carrier of Definition 424.1. The first component route contains X and d_X, while the second component route contains Y and d_Y.

Apply Definition 424.3. Transport in the first component moves the first metric carrier, its classifier, and its distance witness together; transport in the second component moves the corresponding second rows.

The product-distance ledger is read only after both component distances have been displayed. Thus δ consumes the transported d_X and d_Y rows and not a host-level product metric or quotient equality. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier component distance transport`

Theorem 424.5 (ProductMetric projection route exactness)**.** *For every accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$*, each projection consumer route is exactly one of the two displayed* $\mathsf{Prod}^{\uparrow}$ *projection routes in* Π*. The first route reads* X *and* d_X*; the second route reads* Y *and* d_Y*; both routes carry only the corresponding* $\mathsf{Met}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *distance witness,* hsame *transport,* Cont *read, package row, and local naming row.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier projection route exactness`

Proof. Begin with a projection consumer. By Definition 424.1, the only product coordinate is Π, and Π exposes only the first and second projection reads.

The exact first route is the first half of Definition 424.10; it reaches X and d_X through the stored continuation row. The exact second route is the second half of the same theorem; it reaches Y and d_Y through its stored continuation row.

Component transport is already accounted for by Definition 424.4. Therefore no projection consumer can bypass Π to read host pair equality, a quotient product, a selected real limit, or classical choice data. □

Theorem 424.6 (ProductMetric MetricSpace obligation package)**.** *The downstream* $\mathsf{Met}^{\uparrow}$ *obligation package exposed by an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ *consists of component classifier transport, component distance transport, exact projection routes through* Π*, product-distance ledger exactness for* δ*, and the non-escape boundary for quotient products, host pair equality, selected real limits, and classical choice data.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier metricspace obligation package`

Proof. First collect the component obligations. Classifier transport is Definition 424.3, and component distance transport is Definition 424.4.

Next collect the product-facing routes. Projection route exactness is Definition 424.5, and the product-distance ledger is the δ row identified by Definition 424.2.

Finally apply Definition 424.11. The downstream metric-space consumer is anchored exactly at Π, the two component metric certificates, and δ, while the finite carrier excludes all non-displayed product or choice data. □

Theorem 424.7 (ProductMetric component diagonal boundary)**.** *For every accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet*

$$Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N),$$

any diagonal component comparison factors through the two $\mathsf{Met}^{\uparrow}$ *classifiers, the displayed* $\mathsf{Prod}^{\uparrow}$ *projections in* Π*, and the two* $\mathsf{Real}^{\uparrow}$ *distance witness rows* d_X *and* d_Y*. The comparison does not export host pair equality, quotient-product equality, a selected real limit, or classical choice data.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier component diagonal boundary`

Proof. Project the finite carrier of Definition 424.1. The only component data available to a diagonal comparison are the two metric-space rows, the two distance witnesses, and the product projection row Π.

Use Definition 424.3 to transport the two component classifiers together with their displayed rows. Then use Definition 424.5 to force every component read through the first or second projection route of Π.

Finally apply Definition 424.4. The comparison reaches d_X and d_Y only as $\mathsf{Real}^{\uparrow}$ witness rows attached to those component routes, so no host pair equality or quotient-product equality is introduced. □

Theorem 424.8 (ProductMetric distance-ledger triangle route)**.** *Every triangle-style consumer of the product-distance ledger δ first reads the two component distance witnesses d_X and d_Y, then reads the displayed max row or displayed dyadic-sum bound row chosen by δ. The route is anchored by the component classifiers, component distance transport, and the downstream* $\mathsf{Met}^{\uparrow}$ *obligation package, not by a host-level product metric proof.*

Proof. Start with the carrier of Definition 424.1. The ledger δ is stored only after the two component distance witness rows, so a triangle-style consumer has no earlier product-distance coordinate to read.

Apply Definition 424.3 and Definition 424.4. They transport the two component metric rows and expose the two real-valued distance witnesses before any product-distance ledger read.

Finally use Definition 424.6. Its anchor makes δ the product-facing metric-space ledger, represented by the displayed max row or displayed dyadic-sum bound row, and excludes quotient products, host pair equality, selected real limits, and classical choice data. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier distance ledger triangle route`

Theorem 424.9 (ProductMetric RealUp consumer package)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet sees exactly the product row Π, the component distance witnesses d_X and d_Y, the product-distance ledger δ, the transport rows H, the continuation routes C, the package provenance P, and the local naming row N. It does not see quotient products, host pair equality, selected real limits, or classical choice data.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier realup consumer package`

Proof. Read the consumer request through Definition 424.1. The real-valued rows exposed by the carrier are d_X, d_Y, and the ledger δ, with Π, H, C, P, and N recording how those rows are reached.

Use Definition 424.7 for the component side. It forces diagonal and projection-facing reads through the two component classifiers, Π, and the two displayed distance witnesses.

Use Definition 424.8 for the product-distance side. Since every ledger read first passes through d_X and d_Y and then through the displayed max or dyadic-sum row, a $\mathsf{Real}^{\uparrow}$ consumer receives only the finite packet rows named in the statement. □

Theorem 424.10 (ProductMetric projection non-escape)**.** *Every projection consumer of an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Prod}^{\uparrow}$ *row Π, the corresponding* $\mathsf{Met}^{\uparrow}$ *component carrier and classifier row, the matching* $\mathsf{Real}^{\uparrow}$ *distance witness, the stored* hsame *transport, the stored* Cont *projection route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}$ *row. No projection consumer can read a quotient product, host pair equality, selected real limit, or classical choice data from the packet.*

Proof. Start with a projection request. The packet has only one product coordinate, namely the displayed $\mathsf{Prod}^{\uparrow}$ row Π, so the request must read either its first projection route or its second projection route.

On the first route, the consumer reaches the first $\mathsf{Met}^{\uparrow}$ carrier and classifier and then the first real-valued distance witness. On the second route, it reaches the corresponding second component rows.

The transport and continuation coordinates keep each projection attached to its component route, and P and N record the provenance and local naming surface. Since the carrier does not include any further product object or host equality row, the two projection routes exhaust the packet boundary. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier projection non escape`

Theorem 424.11 (ProductMetric MetricSpace anchor)**.** *An accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet anchors a downstream* $\mathsf{Met}^{\uparrow}$ *consumer exactly at the displayed product carrier* Π*, the component* $\mathsf{Met}^{\uparrow}$ *classifiers, and the product-distance ledger* δ*. The consumer may use the two component* $\mathsf{Real}^{\uparrow}$ *distance witnesses and the ledger rule for max or displayed dyadic sum, but it receives no quotient product, no host pair equality, no selected real limit, and no classical choice data.*

Proof. Read the downstream metric-space request through the product carrier row. The row Π supplies only the two component projections, so every endpoint read is decomposed into first and second component reads.

For each component read, the corresponding $\mathsf{Met}^{\uparrow}$ certificate supplies the classifier and the $\mathsf{Real}^{\uparrow}$ distance witness. The product-distance ledger δ is then the sole distance row exported by the product packet.

Thus the downstream metric-space surface is anchored by Π, the two component metric certificates, and δ. All further data remain outside the carrier because they are not coordinates of the finite BHist packet. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier metricspace anchor`

Theorem 424.12 (ProductMetric scoped dependency package)**.** *The scoped dependency package of an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *carrier* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ *consists exactly of the two* $\mathsf{Met}^{\uparrow}$ *carrier and classifier rows, the two* $\mathsf{Real}^{\uparrow}$ *component distance witness rows, the* $\mathsf{Prod}^{\uparrow}$ *pairing and projection row* Π*, the product-distance ledger* δ*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ *provenance.*

Classifier stability, component distance transport, projection exactness, and the downstream $\mathsf{Met}^{\uparrow}$ *anchor all factor through this package and through no host product or quotient equality row.*

Proof. Start with the carrier definition. Its visible coordinates are precisely the two metric components, two real distance rows, product row, distance ledger, and transport, continuation, package, and naming data.

Apply classifier stability and component distance transport. These rows keep the two component classifiers and the two component distance witnesses aligned under the displayed hsame and Cont routes.

Apply projection route exactness and the metric-space anchor. They force every product-facing consumer through Π and then through δ. The listed rows therefore form the whole scoped dependency package. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier scoped dependency package`

Theorem 424.13 (ProductMetric distance-ledger scope)**.** *Every accepted* $\mathsf{ProductMetric}^{\uparrow}$ *distance-ledger read is scoped by the route*

$$(X, Y, \Pi) \longmapsto (d_X, d_Y) \longmapsto \delta.$$

The first stage reads the two $\mathsf{Met}^{\uparrow}$ *component classifiers through the* $\mathsf{Prod}^{\uparrow}$ *projections, the second stage transports the two* $\mathsf{Real}^{\uparrow}$ *component distance rows through* BHist, hsame, *and* Cont, *and the final stage consumes the displayed max or dyadic-sum product-distance ledger with* Pkg *and* $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ *provenance.*

This is the distance-ledger triangle route available to a downstream $\mathsf{Met}^{\uparrow}$ *consumer; it does not export host pair equality, a quotient product, or a selected real limit.*

Proof. Project the product row Π. Projection exactness says that every first-component or second-component read enters through one of its two displayed projection routes.

Use component distance transport to attach d_X and d_Y to those routes. The component classifiers supply the only admissible transport for the two real-valued distance witnesses.

Finally apply the distance-ledger triangle route and the scoped dependency package. The product-distance row δ is read after the component distances and is carried by the displayed package and naming provenance, so no external product metric proof can enter. □

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier distance ledger scope`

Theorem 424.14 (ProductMetric Real classifier bridge)**.** *Let* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ *be an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet. If the first and second component distance rows are transported by their component* $\mathsf{Met}^{\uparrow}$ *classifiers, then the product-distance ledger* δ *exports the same* $\mathsf{Real}^{\uparrow}$ *classifier row used by the downstream* $\mathsf{ProductMetric}^{\uparrow}$ $\mathsf{Met}^{\uparrow}$ *anchor. The bridge is the finite route from the component classifier reads through* d_X *and* d_Y *to the product* $\mathsf{Real}^{\uparrow}$ *row carried by* δ*.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier real classifier bridge`

Proof. Start with the component transport. Apply Definition 424.4 to move d_X and d_Y along the displayed component classifier rows and their hsame and Cont routes.

Next read the real-valued consumer surface. Apply Definition 424.9; it identifies the visible $\mathsf{Real}^{\uparrow}$ rows as exactly the two component distance witnesses, the product-distance ledger δ, and the transport, continuation, provenance, and naming rows.

Finally use the downstream anchor and scope theorem. By Definition 424.11, the metric consumer reads the product distance only through Π, the component classifiers, and δ; by Definition 424.13, δ is reached after the transported component distances. Therefore the product packet exports one $\mathsf{Real}^{\uparrow}$ classifier row for the product distance, with no host pair equality, quotient product, selected real limit, or external metric proof. □

Theorem 424.15 (ProductMetric Real classifier public export)**.** *Let* $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ *be an accepted* $\mathsf{ProductMetric}^{\uparrow}$ *packet. The scoped product packet exports a public* $\mathsf{Met}^{\uparrow}$*-facing* $\mathsf{Real}^{\uparrow}$ *classifier surface exactly through the two component* $\mathsf{Met}^{\uparrow}$ *distance rows* d_X *and* d_Y*, the* $\mathsf{Prod}^{\uparrow}$ *pairing row* Π*, the product-distance ledger* δ*, the displayed* hsame *and* Cont *transports, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ *row. It exports no host pair equality, quotient product, selected real limit, or external metric proof.*

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier real classifier public export`

Proof. First build the component distance surface. Apply Definition 424.4; the two component $\mathsf{Met}^{\uparrow}$ classifiers carry the $\mathsf{Real}^{\uparrow}$ distance rows d_X and d_Y through the displayed hsame and Cont routes.

Next collect the public consumer package. Apply Definition 424.9; a $\mathsf{Real}^{\uparrow}$ consumer sees only the product row Π, the component distance witnesses, the product-distance ledger δ, and the provenance and naming rows P and N.

Finally repack the route as a $\mathsf{Met}^{\uparrow}$-facing export. Apply Definition 424.11 to anchor the downstream metric consumer at Π, the component classifiers, and δ; apply Definition 424.13 to force the ledger read after the component distances; then apply Definition 424.14 to identify the exported $\mathsf{Real}^{\uparrow}$ classifier row. The excluded host and quotient data are absent because none is a coordinate of Q. □

Theorem 424.16 (ProductMetric standard MetricSpace bridge certificate)**.** *Let $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N$ be an accepted $\mathsf{ProductMetric}^{\uparrow}$ packet. Then Q supplies the standard $\mathsf{Met}^{\uparrow}$ product bridge certificate exactly through the two component $\mathsf{Met}^{\uparrow}$ classifier surfaces, the two $\mathsf{Real}^{\uparrow}$ component distance rows d_X and d_Y, the $\mathsf{Prod}^{\uparrow}$ pairing and projection row Π, the product-distance ledger δ, and the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ provenance.*

The bridge certificate is the public route that a downstream $\mathsf{Met}^{\uparrow}$ consumer reads when it treats the product carrier as a metric-space object. It has no additional coordinate for host pair equality, quotient products, selected real limits, compactness, or an external metric proof.

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier standard metricspace bridge certificate`

Proof. First collect the scoped packet. By Definition 424.12, every accepted product packet exposes exactly the two component metric rows, the two real distance rows, the product row, the distance ledger, and the displayed transport, continuation, package, and naming provenance.

Next identify the distance route. By Definition 424.13, every product-distance read passes from (X, Y, Π) through (d_X, d_Y) to δ. The existing $\mathsf{Met}^{\uparrow}$ semantic certificate surface supplies the $\mathsf{Met}^{\uparrow}$ distance-witness reading, and the existing $\mathsf{Prod}^{\uparrow}$ semantic certificate surface supplies the pairing and projection reading used by Π.

Finally repack the public consumer view. Apply Definition 424.15 to expose the $\mathsf{Real}^{\uparrow}$ classifier row carried by the product distance, and apply Definition 424.5 to keep each component read on the displayed projection route. These rows assemble the standard product $\mathsf{Met}^{\uparrow}$ bridge certificate, and no non-displayed host or quotient datum is available from Q. □

Theorem 424.17 (ProductMetric public MetricSpace export)**.** *Every accepted $\mathsf{ProductMetric}^{\uparrow}$ packet $Q = (X, Y, d_X, d_Y, \Pi, \delta, H, C, P, N)$ exports a public $\mathsf{Met}^{\uparrow}$ surface exactly through the two component $\mathsf{Met}^{\uparrow}$ classifier rows X, Y, the two $\mathsf{Real}^{\uparrow}$ distance rows d_X, d_Y, the $\mathsf{Prod}^{\uparrow}$ projection row Π, the product-distance ledger δ, componentwise hsame and displayed Cont routes H, C, Pkg provenance P, and the local $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ row N.*

The export excludes quotient products, host pair equality, selected real limits, compactness, and external metric proofs.

Lean4 ✓ `BEDC.Derived.ProductMetricUp.ProductMetricCarrier public metricspace export`

Proof. First apply the standard MetricSpace bridge certificate. It assembles the downstream product metric surface from the two component metric classifiers, component real-distance rows, the product projection row, the distance ledger, and the displayed transport, continuation, package, and naming rows.

Next apply the Real classifier public export. This identifies the $\mathsf{Real}^{\uparrow}$ surface consumed by the product distance as exactly the route through $d_X, d_Y, \Pi, \delta, H, C, P, N$.

Finally apply projection route exactness. Each component read enters through the corresponding displayed projection route in Π, so no consumer can bypass the packet to obtain host pair equality, a quotient product, a selected real limit, compactness, or an external metric proof. □

Theorem 424.18 (ProductMetric mature consumer exhaustion)**.** *Every mature* $\mathsf{ProductMetric}^{\uparrow}$ *consumer is exhausted by the public* $\mathsf{Met}^{\uparrow}$ *export of Definition 424.17, the standard bridge certificate of Definition 424.16, projection exactness through Definition 424.5, the* $\mathsf{Real}^{\uparrow}$ *classifier export of Definition 424.15, and the product-distance ledger routes of Definition 424.13. The exhausted surface contains exactly the two component metric rows, their real-valued distance witnesses, the* $\mathsf{Prod}^{\uparrow}$ *projection row, the product-distance ledger, the displayed* hsame *and* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$ *row.*

No mature consumer receives a quotient product, host pair equality, selected real limit, compactness witness, classical choice datum, or external metric proof.

Proof. Read the consumer through the public $\mathsf{Met}^{\uparrow}$ export. That theorem exposes the same finite packet surface already named by the carrier: component metrics, component distances, the product projection row, the product-distance ledger, transport rows, continuation routes, package provenance, and local naming data.

Next apply the standard bridge certificate and projection route exactness. These force every metric-space read to pass through the two component classifiers, the two displayed projections in Π, and the product-distance ledger δ.

Finally apply the $\mathsf{Real}^{\uparrow}$ classifier export and the distance-ledger scope theorem. They identify the real-valued part of the product metric as the route from d_X and d_Y to δ. Since none of the excluded host or quotient data is a coordinate of the packet, the listed routes exhaust the mature consumer boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProductMetric}^{\uparrow}$ packet is generated from two $\mathsf{Met}^{\uparrow}$ carrier and classifier rows, two $\mathsf{Real}^{\uparrow}$ distance witness rows, a $\mathsf{Prod}^{\uparrow}$ pairing row, a product-distance ledger, componentwise hsame transport, projection Cont routes, Pkg provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped rows are Definition 424.12 and Definition 424.13; the public export row is Definition 424.17, which binds every product-distance read to the displayed component metrics, pairing row, distance ledger, projection routes, package provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProductMetric}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ProductMetricUp.ProductMetricCarrier\_metricspace\_anchor`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient products, host pair equality, selected real limits, classical choice data, compactness of the product, or bridge checking.

Upgrade path: Bridge checking requires a Lean-checked standard bridge theorem for the displayed public $\mathsf{Met}^{\uparrow}$ export.

Chapter 425

A Concrete Naming Certificate for $\mathsf{CauchyErrorCertificate}^{\uparrow}$

The $\mathsf{CauchyErrorCertificate}^{\uparrow}$ interface records the explicit finite error ledger used by completion-facing real names. In the Bishop–Bridges constructive-analysis reading, convergence data are consumed through moduli and displayed bounds. This packet expresses that surface as BEDC rows over regular rational readback, dyadic tail control, finite error budgeting, provenance, and local naming.

The chapter consumes the finite observation-budget surface of chapter 611. That budget chapter fixes which $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows may be inspected, while the present certificate records the error ledger that bounds those inspected rows.

Definition 425.1 (CauchyErrorCertificate carrier)**.** *A* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, M, T, E, P, N)$$

with the following visible coordinates:

1. *R is the readback route through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the downstream* $\mathsf{Real}^{\uparrow}$ *seal used by the consumer request;*
2. *M is the unary precision modulus row selecting the finite index at which the requested dyadic tolerance is read;*
3. *T is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *or* $\mathsf{DiagonalModulus}^{\uparrow}$ *row that records the selected tail estimate and its diagonal admission packet;*
4. *E is the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger that accounts for the dyadic tolerance, tail contribution, rounding contribution, and transported consumer error;*
5. *P is the* Pkg *provenance row naming the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependency certificates;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}^{\uparrow}}$ *row.*

The local classifier compares two carriers only by componentwise hsame *transport of R, M, T, E, P, and N, together with the displayed* Cont *routes carried by the readback and budget rows.*

Theorem 425.2 (CauchyErrorCertificate NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}^{\uparrow}}$ *surface has five finite obligations: carrier habitation by Definition 425.1; classifier stability under the displayed* hsame *and* Cont *rows; exact routing of every requested dyadic error through M, T,*

and E before the read reaches R; provenance preservation for the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows named by P; and non-escape of quotient streams, selected host limits, ambient completeness, and host real equality.*

Proof. Carrier habitation is finite packet assembly. The readback route, modulus row, tail or diagonal row, budget ledger, package provenance, and local certificate are all displayed coordinates of C.

Classifier stability is componentwise. Each coordinate is transported only by the stored hsame comparison and the corresponding Cont route, so an accepted packet remains inside the same error-ledger surface.

The exactness, provenance, and boundary clauses follow from the coordinate list. A consumer error request must read the modulus, tail or diagonal packet, and finite budget ledger before reaching the readback route, while P records the dependency certificates and no omitted quotient, selected limit, host-completeness, or host-equality row is present. □

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket namecert obligations`

Theorem 425.3 (CauchyErrorCertificate modulus soundness)**.** *For every accepted* CauchyErrorCertificate$^{\uparrow}$ *carrier $C = (R, M, T, E, P, N)$ and every finite dyadic tolerance request, the selected unary precision row M determines a finite tail or diagonal read in T, and the resulting budget entry in E bounds the downstream* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *readback route R at that tolerance. The bound is a displayed dyadic ledger row, not a host real inequality.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket modulus soundness`

Proof. Read the requested tolerance through the unary modulus coordinate M. This chooses the finite index at which the packet is allowed to inspect the regular rational route.

Use T to admit the corresponding dyadic tail or diagonal row. The row records exactly the tail estimate needed by the selected precision and passes it to the finite budget ledger.

Finally read E before consuming R. Since E is a FiniteErrorBudget$^{\uparrow}$ ledger, the downstream bound is the displayed dyadic accounting row over the chosen tolerance and not an external inequality in a host real structure. □

Theorem 425.4 (CauchyErrorCertificate budget handoff closure)**.** *Let $C = (R, M, T, E, P, N)$ be an accepted* CauchyErrorCertificate$^{\uparrow}$ *packet. If the budget row E is transported to a classifier-equivalent budget row E', and if the displayed* Cont *routes $M \to T \to E'$ and $R \to E' \to P$ are supplied, then the transported tuple (R, M, T, E', P, N) is again an accepted* CauchyErrorCertificate$^{\uparrow}$ *packet over the same readback, modulus, tail, provenance, and local* NameCert *rows.*

Proof. Project the carrier rows from Definition 425.1. The accepted packet supplies unary admissibility for R, M, T, E, P, N, the original budget-facing routes, the BHist packet, the Pkg provenance, and the local naming row.

Transport only the budget admissibility along the displayed hsame comparison from E to E'. The readback, modulus, tail, provenance, and local NameCert rows remain the same coordinates of the packet.

Finally replace the two budget-facing continuation routes by the supplied routes $M \to T \to E'$ and $R \to E' \to P$. Repacking the unchanged BHist, Pkg, and NameCert provenance with the transported budget row yields the accepted tuple (R, M, T, E', P, N). □

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket budget handoff closure`

Theorem 425.5 (CauchyErrorCertificate non-escape boundary)**.** *An accepted* CauchyErrorCertificate$^{\uparrow}$ *carrier exports only its displayed readback route, unary precision modulus, dyadic tail or diagonal packet, finite error-budget ledger, package provenance, local naming row, and the attached* BHist*,* hsame*, and* Cont *transport. It exports no quotient stream equality, no selected host limit, no ambient completeness theorem, and no host real equality assertion.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket non escape boundary`

Proof. Inspect the carrier coordinates. Each visible field is one of R, M, T, E, P, or N, together with the ambient finite history and transport rows needed to compare them.

The modulus and budget rows control only finite dyadic requests. The tail or diagonal packet supplies the finite estimate consumed by that budget, and the readback route exposes only the named regular rational and real seal rows.

No coordinate contains a quotient of streams, a selected limit, an ambient complete space, or an equality proof for host real numbers. Therefore no such object can be exported by the certificate. □

Theorem 425.6 (CauchyErrorCertificate downstream handoff)**.** *Every downstream completion, diagonal, or validated-numerics consumer of an accepted* CauchyErrorCertificate$^{\uparrow}$ *carrier receives a finite packet over the same unary modulus, dyadic tail or diagonal packet, finite error-budget ledger,* DyadicRatCore$^{\uparrow}$ *observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* Real$^{\uparrow}$ *seal,* Pkg *provenance, and* NameCert *row. The handoff preserves the non-escape boundary of Definition 425.5.*

Proof. Begin with the modulus soundness theorem. It forces each requested tolerance through M, then through the displayed tail or diagonal packet T, and finally through the finite budget ledger E before the readback route R is consumed.

The provenance row P names the dependency certificates for the dyadic observations, stream windows, regular rational readbacks, and real seal. Transport along hsame and Cont therefore keeps the downstream packet on the same visible rows.

Apply the non-escape boundary. Since the carrier has no field for quotient streams, selected host limits, host completeness, or host real equality, the downstream handoff consists exactly of the listed finite BEDC rows. □

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket downstream handoff`

Theorem 425.7 (CauchyErrorCertificate budget-readback determinacy)**.** *Let* $C = (R, M, T, E, P, N)$ *and* $C' = (R', M', T', E', P', N')$ *be accepted* CauchyErrorCertificate$^{\uparrow}$ *carriers. Suppose the unary modulus rows* M, M'*, dyadic tail or diagonal rows* T, T'*, finite error-budget rows* E, E'*, and provenance rows* P, P' *are matched by the displayed* hsame *comparisons and* Cont *routes of the two packets. Then every completion, diagonal, or validated-numerics consumer reaches the same* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *readback boundary from* C *and* C'*. The packet has no alternate route through quotient streams, host limits, ambient completeness, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket budget readback determinacy`

Proof. Use Definition 425.3. A finite dyadic tolerance request must pass through the unary modulus row, then the tail or diagonal row, and then the finite error-budget ledger before it reaches the readback route.

The assumed hsame and Cont matches identify those three rows and their provenance rows between C and C'. Hence the downstream packet seen by a completion, diagonal, or validated-numerics consumer is the same packet described by Definition 425.6.

It remains to exclude hidden readback sources. By Definition 425.5, the carrier exports no quotient stream equality, selected host limit, ambient completeness theorem, or host real equality assertion. Thus the displayed modulus, tail, budget, and provenance ledger determines the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback boundary. □

Theorem 425.8 (CauchyErrorCertificate uniform budget handoff)**.** *Let U be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet and let $C = (R, M, T, E, P, N)$ be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *carrier consuming the tail ledger displayed by U. For every finite family tolerance request, the handoff factors through the shared window family of U, the shared modulus selected by its tail ledger, the dyadic tolerance row, the finite tail ledger, the finite Cauchy error budget E, and then the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback boundary R. The resulting packet is a finite BEDC error-budget handoff over the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, with no route through quotient streams, selected host limits, host real equality, or ambient completeness.*

Proof. Start on the uniform side. By Definition 435.9, each family tolerance request reaches a recorded window family, a shared modulus row, a dyadic tolerance row, and a finite tail-ledger entry before any consumer can read a tail estimate.

Next package that tail evidence for the regular-rational consumer. Apply Definition 435.16: the same finite ledger supplies the $\mathsf{RegSeqRat}^{\uparrow}$ route consumed by downstream real-name interfaces, rather than a selected limit or an ambient completeness principle.

Now pass the displayed tail row into C. The modulus soundness theorem Definition 425.3 routes the dyadic request through M, through the tail or diagonal coordinate T, and into the finite error-budget ledger E before the readback boundary R is available.

Finally apply Definition 425.7. Once the shared modulus, tail ledger, budget row, and provenance rows are fixed, every completion, diagonal, or validated-numerics consumer sees the same $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary. The non-escape boundaries of the two chapters exclude quotient streams, selected host limits, host real equality, and ambient completeness from the handoff. □

Theorem 425.9 (CauchyErrorCertificate modulus-budget composition)**.** *Let $C = (R, M, T, E, P, N)$ be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet. Every precision request routed by the unary modulus row M composes with the dyadic tail or diagonal packet T and the finite error-budget ledger E before reaching the readback row R. The composed route consumes only the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal, package provenance P, and local* NameCert *row N.*

Proof. Begin with the modulus row. By Definition 425.3, a requested tolerance enters the packet through M and is interpreted as a finite dyadic request.

Next read the tail or diagonal evidence. The same theorem routes the request through T, and the downstream handoff of Definition 425.6 keeps that evidence on the displayed dyadic observations, stream windows, regular rational rows, and real seal.

Finally consume the finite budget. The budget-readback determinacy theorem Definition 425.7 fixes the E and P rows before the consumer reaches R, while Definition 425.5 excludes quotient streams, host limits, ambient completeness, and host real equality. □

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket modulus budget composition`

Theorem 425.10 (CauchyErrorCertificate readback route uniqueness)**.** *Let $C = (R, M, T, E, P, N)$ and $C' = (R', M', T', E', P', N')$ be accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packets. If the modulus rows M, M', tail or diagonal rows T, T', finite error-budget rows E, E', and provenance rows P, P' are matched by the displayed* hsame *and* Cont *routes, then both packets determine the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback boundary. The boundary is therefore unique relative to the displayed M, T, E, P rows.*

Proof. First compose the request route in each packet by Definition 425.9. A readback can reach R or R' only through the corresponding M, T, E, P rows.

The assumed hsame and Cont matches identify those four rows componentwise. Applying Definition 425.7 then gives the same downstream completion, diagonal, or validated-numerics packet on both sides.

The non-escape boundary rules out any second readback source. Since neither carrier contains quotient streams, selected host limits, ambient completeness, or host real equality, the shared M, T, E, P ledger fixes the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary. □

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket readback route uniqueness`

Theorem 425.11 (CauchyErrorCertificate diagonal-tail compatibility)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet $C = (R, M, T, E, P, N)$, a diagonal row and a tail-bound row accepted by the same packet feed the same finite error-budget ledger E. Their compatibility is witnessed inside the displayed tail or diagonal packet T and does not require host completeness, selected limits, quotient streams, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket diagonal tail compatibility`

Proof. Inspect the carrier rows. The packet has one tail-or-diagonal coordinate T and one finite budget coordinate E, both named by the same provenance row P and local certificate N.

Apply Definition 425.3. Both diagonal and tail-bound requests are routed by the same modulus row M before they are checked against the packet T.

The budget row is then fixed by Definition 425.9. The downstream handoff and budget-readback determinacy theorems identify the same E before any consumer reaches R.

Finally apply the non-escape boundary. Because the carrier has no host-completeness or selected-limit coordinate, diagonal and tail evidence can be compatible only through the displayed finite budget ledger. □

Theorem 425.12 (CauchyErrorCertificate uniform diagonal compatibility)**.** *Let U be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, let D be an accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *row, and let $C = (R, M, T, E, P, N)$ be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet consuming both displayed inputs. If U and D feed the same accepted packet C, then every finite family tolerance request and every diagonal precision request pass through the same unary modulus row M, the same tail-or-diagonal packet T, and the same finite budget ledger E before reaching the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback boundary R. The combined route exports no quotient streams, selected host limits, ambient completeness principle, or host real equality.*

Proof. Start with the uniform-family route. By Definition 425.8, every finite family tolerance request reaches the Cauchy error certificate through the displayed uniform ledger and then enters the packet through the finite budget route visible in C.

Now read the diagonal request. The diagonal-tail compatibility theorem Definition 425.11 says that diagonal and tail evidence accepted by the same packet share the coordinate T and feed the same budget row E.

Compose the routes through the packet. By Definition 425.9, both requests are routed by the unary modulus row M, checked against the tail-or-diagonal packet T, and consumed by the ledger E before the readback boundary R is exposed.

Finally apply Definition 425.10 together with the non-escape boundary. Once M, T, E, and the provenance row are fixed, the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary is unique inside the displayed carrier, and the absent quotient, selected-limit, completeness, and host-equality objects cannot supply an alternate route. □

Theorem 425.13 (CauchyErrorCertificate budget-window exhaustion)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet* $C = (R, M, T, E, P, N)$*, every finite dyadic tolerance request is exhausted by the displayed budget-window route before the readback row* R *is available. The route first reads the unary precision modulus* M*, then the tail or diagonal packet* T*, then the finite error-budget ledger* E*, and only then exposes the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback boundary named by* R*.*

No consumer of the packet can bypass this M, T, E *route through quotient streams, selected host limits, ambient completeness, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket budget window exhaustion`

Proof. Begin with a finite dyadic tolerance request. The modulus soundness theorem Definition 425.3 forces the request through the unary modulus row M and selects the finite tail or diagonal read in T.

Next consume the finite budget ledger. By Definition 425.9, the selected T row feeds E before any downstream read reaches R.

Finally apply readback route uniqueness and the non-escape boundary. The theorems Definition 425.10 and Definition 425.5 exclude any second route through non-displayed host objects, so the M, T, E budget window is exhausted before R is readable. □

Theorem 425.14 (CauchyErrorCertificate shared modulus transport)**.** *Let* $C = (R, M, T, E, P, N)$ *and* $C' = (R', M', T', E', P', N')$ *be accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packets whose classifier comparison identifies the displayed* M, T, E, P *rows by* hsame *and* Cont *transport. Then transport preserves the shared unary modulus route: every finite tolerance request read in* C *and transported to* C' *is routed through the matched row* M' *before it reaches* T'*,* E'*, and* R'*.*

The transport creates no indexwise hidden modulus and exports no modulus function outside the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket shared modulus transport`

Proof. Use the classifier comparison componentwise. The hypothesis identifies the unary modulus rows, tail or diagonal rows, finite budget rows, and provenance rows by the stored hsame comparisons and continuation routes.

Read a finite tolerance request in C. Budget-window exhaustion routes it through M, T, E before exposing R; transporting along the classifier comparison carries these same displayed rows to M', T', E' in C'.

It remains to exclude a hidden modulus. The carrier of Definition 425.1 has only one modulus coordinate, and the non-escape boundary rules out additional host data. Hence transport preserves the shared unary modulus row instead of selecting an indexwise modulus outside the packet. □

Theorem 425.15 (CauchyErrorCertificate shared modulus exhaustion)**.** *Let $C = (R, M, T, E, P, N)$ be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet. Every finite tolerance read that enters the shared unary modulus row M is exhausted by the displayed sequence M, T, E, R: the read first selects the finite modulus boundary, then consumes the tail or diagonal row, then consumes the finite error-budget ledger, and only then reaches the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback boundary. No second modulus row, indexwise hidden selector, quotient stream equality, selected host limit, ambient completeness theorem, or host real equality is available to the read.*

Proof. Begin with the finite tolerance read. By Definition 425.3, the read enters the packet through the unary modulus row M and selects the tail or diagonal evidence in T.

Next exhaust the budget window. Theorem 425.13 routes the selected tail or diagonal row through the finite error-budget ledger E before the readback row R is exposed.

Finally exclude alternate modulus data. The shared modulus transport theorem Definition 425.14 keeps the route on the same displayed modulus row under classifier comparison, while the non-escape boundary removes quotient streams, selected host limits, ambient completeness, and host real equality from the carrier. □

Theorem 425.16 (CauchyErrorCertificate budget-seal comparison)**.** *Let $C = (R, M, T, E, P, N)$ and $C' = (R', M', T', E', P', N')$ be accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packets. If the displayed budget and seal-facing rows M, T, E, P and M', T', E', P' are matched by the packet classifiers, then every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer comparing C and C' sees the same finite budget route before reaching the readback boundary. The comparison is carried by the displayed modulus, tail-or-diagonal, budget, provenance,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not by host real equality or a selected limit.*

Proof. First route each consumer through the budget window. By Definition 425.15, every finite tolerance read in either packet passes through the displayed modulus, tail-or-diagonal, budget, and readback rows.

Now compare the packets. The hypotheses identify the M, T, E, P rows with M', T', E', P' by the packet classifiers, and Definition 425.14 carries the shared modulus route to the matched packet.

It remains to state what the seal-facing consumer receives. The readback route uniqueness theorem Definition 425.10 fixes the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary relative to the displayed budget rows, and the seal budget package Definition 425.18 repacks those rows for the finite seal surface. □

Theorem 425.17 (CauchyErrorCertificate Real consumer unblock)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet $C = (R, M, T, E, P, N)$, the shared modulus exhaustion route and budget-seal comparison surface expose a finite Real consumer unblock package. A downstream* $\mathsf{Real}^{\uparrow}$ *consumer receives the displayed unary modulus M, tail or diagonal packet T, finite error-budget ledger E, provenance row P, readback boundary R,* hsame *transports,* Cont *routes,* Pkg *data, and* NameCert *row, and the consumer has no route through quotient streams, selected host limits, ambient completeness, host real equality, or bridge checking.*

Proof. Start with the exhaustive read. Theorem 425.15 routes every finite tolerance request through M, T, E before R is available.

Next bind comparison for seal-facing consumers. Theorem 425.16 shows that the same displayed rows determine the comparison surface seen by $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Finally repack the consumer boundary. The scoped budget-window package Definition 425.22 supplies the displayed BHist, hsame, Cont, Pkg, and NameCert rows, while the non-escape boundary

excludes quotient streams, selected limits, ambient completeness, host real equality, and bridge checking. □

Theorem 425.18 (CauchyErrorCertificate seal budget package)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet* $C = (R, M, T, E, P, N)$*, modulus soundness, diagonal-tail compatibility, readback route uniqueness, and uniform diagonal compatibility assemble the finite* $\mathsf{Real}^{\uparrow}$ *seal budget package. Every completion-facing consumer receives the displayed unary modulus row* M*, tail or diagonal packet* T*, finite budget ledger* E*, provenance row* P*, and readback boundary* R *as one finite package over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The package closes the error-budget obligation surface for the displayed certificate, while leaving quotient streams, selected host limits, ambient completeness, host real equality, and bridge checking outside the chapter.

Lean4 ✓ `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket seal budget package`

Proof. First bind the soundness route. Theorem 425.3 supplies the finite tolerance-to-budget route through M, T, and E.

Next bind compatibility. Theorem 425.11 states that diagonal and tail evidence accepted by the same packet feed the same finite budget ledger, and Theorem 425.12 identifies the uniform-family and diagonal routes through the same M, T, E coordinates.

Finally bind the seal boundary. Theorem 425.10 fixes the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback boundary relative to the displayed rows, and the non-escape boundary excludes all host-side completion objects. Repacking these rows gives the announced finite $\mathsf{Real}^{\uparrow}$ seal budget package. □

Theorem 425.19 (CauchyErrorCertificate composed budget readback handoff)**.** *Let* $C = (R, M, T, E, P, N)$ *be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet. Suppose a binary-prefix* $\mathsf{BinaryExpansion}^{\uparrow}$ *regular handoff and a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *real-seal handoff are both consumed through the same displayed unary modulus* M*, the same tail-or-diagonal row* T*, the same finite error-budget ledger* E*, and the same package provenance row* P*. Then every* $\mathsf{Real}^{\uparrow}$ *completion-facing read factors through the common finite route*

$$\textit{binary prefix window} \longrightarrow \textit{dyadic approximation row} \longrightarrow \textit{uniform tail ledger} \longrightarrow \textit{Cauchy error budget}$$
$$\longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal.}$$

The composed handoff exports no quotient stream, selected host limit, host real equality, or ambient completeness theorem. The only displayed data available to the read are the binary prefix window, dyadic approximation row, uniform tail ledger, finite Cauchy budget, readback row R*, provenance row* P*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with the binary-prefix side. Theorem 418.7 factors the $\mathsf{BinaryExpansion}^{\uparrow}$ regular handoff through the inspected prefix window, dyadic approximation row, and regular-sequence handoff boundary. Thus the read does not receive an independent infinite bitstream or a second rational stream.

Next attach the uniform Cauchy side. Theorem 425.8 routes the uniform real-seal handoff through the displayed modulus M, the common tail-or-diagonal row T, and the finite ledger E. The shared M, T, E, P hypotheses identify the binary and uniform routes before either reaches the readback boundary.

Now fix the downstream boundary. Theorem 425.10 identifies the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback route relative to the displayed modulus, tail-or-diagonal, budget, and provenance rows.

Finally package the seal-facing read. Theorem 425.18 repacks the same displayed rows as the finite $\mathsf{Real}^{\uparrow}$ seal budget package, while the non-escape boundary leaves quotient streams, selected host limits, host real equality, and ambient completeness outside the composed handoff. □

Theorem 425.20 (CauchyErrorCertificate schedule-fusion shared-modulus budget)**.** *Let $C = (R, M, T, E, P, N)$ be an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet. Suppose C consumes a displayed* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *shared-bound certificate whose meet row is the same unary modulus row M, and suppose the same schedule rows are packaged by an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer read factors through the single finite route*

$$\textit{fused schedule row} \longrightarrow \textit{window ledger} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}\ \textit{readback} \longrightarrow \textit{shared meet modulus}$$
$$\longrightarrow \textit{dyadic tolerance ledger} \longrightarrow \textit{finite Cauchy error budget} \longrightarrow \mathsf{Real}^{\uparrow}\ \textit{seal.}$$

The route is carried only by the displayed schedule, window, readback, shared meet, dyadic tolerance, budget, provenance, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream, selected host limit, ambient completeness theorem, host real equality, or bridge-checking oracle content.*

Proof. First bind the Cauchy budget route. Theorem 425.19 factors the completion-facing read through the finite budget and readback handoff of the accepted $\mathsf{CauchyErrorCertificate}^{\uparrow}$ packet, so the read reaches R only after it has passed through M, T, E, and P.

Next bind the shared modulus surface. Theorem 462.28 supplies the displayed shared-bound certificate for the $\mathsf{CauchyModulusMeet}^{\uparrow}$ row, and Theorem 462.23 forces each shared-bound consumer through the source modulus rows and the meet row before any downstream read is exposed.

Now lock the schedule side. Theorem 449.4 routes the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers through the fused schedule, window ledger, and regular-rational readback rows. Theorem 449.13 keeps that schedule route independent of seal-facing data not displayed by the carrier.

Finally identify the common route. The hypotheses say that the meet row read by the shared-modulus certificate is the same row M consumed by C, and that the fused schedule rows are the rows seen by the downstream consumer. Combining the three displayed routes gives the announced sequence from schedule row through window ledger, readback, shared meet modulus, dyadic tolerance ledger, finite error budget, and Real seal.

The non-escape clauses of the cited certificates leave quotient streams, selected limits, ambient completeness, host real equality, and bridge-checking oracle content outside the packet. Thus no consumer can bypass the single finite route by appealing to a host completion object. □

Theorem 425.21 (CauchyErrorCertificate schedule-fusion consumer determinacy)**.** *Let $C = (R, M, T, E, P, N)$ and $C' = (R', M', T', E', P', N')$ be accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packets. Suppose both packets are consumed through the same displayed* $\mathsf{RealScheduleFusion}^{\uparrow}$ *schedule rows, the same* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *shared-bound row, the same* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail ledger, and matched M, T, E, P budget rows after componentwise* hsame *and the displayed* Cont *transports. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of C and C' reaches the same finite* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

The common boundary is determined only by the fused schedule row, shared modulus row, uniform tail ledger, matched finite error-budget rows, readback row, provenance row, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data. It uses no quotient stream equality, selected host limit, host real equality, or ambient completeness principle.*

Proof. First lock the schedule side. Theorem 425.20 forces both consumers through the same fused schedule route before either read reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal-facing boundary.

Next identify the budget route. The shared modulus transport theorem Definition 425.14 carries the matched M rows across the displayed hsame and Cont transports, while the shared $\mathsf{UniformCauchyCriterion}^{\uparrow}$ tail ledger fixes the displayed T row used by the two accepted packets.

Now fix readback. Theorem 425.10 identifies the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback boundary once M, T, E, P are matched. Thus the two consumers cannot separate after the fused schedule, shared modulus, tail ledger, finite budget, and provenance rows have been identified.

Finally repack the seal surface. Theorem 425.18 packages the identified rows as the finite $\mathsf{Real}^{\uparrow}$ seal boundary. The non-escape boundary of Definition 425.5 excludes quotient streams, selected host limits, host real equality, and ambient completeness from supplying a second consumer boundary. □

Theorem 425.22 (CauchyErrorCertificate scoped budget-window package)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet* $C = (R, M, T, E, P, N)$*, the displayed modulus row, tail-or-diagonal packet, finite error-budget ledger, provenance row, readback boundary,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows form the scoped dependency surface for every dyadic error request. The scoped package exports no quotient stream, selected host limit, host real equality, or ambient completeness row.*

Proof. First bind the request route. Theorem 425.3 routes each dyadic error request through the modulus row and into the displayed tail-or-diagonal packet.

Next bind the finite window. Theorem 425.13 exhausts the request through M, T, E before the readback row is visible, and Theorem 425.14 keeps that route stable under the displayed hsame and Cont transports.

Now fix the readback boundary. Theorem 425.10 identifies the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ boundary relative to the displayed modulus, tail-or-diagonal, budget, and provenance rows.

Finally apply Theorem 425.18. It repacks the displayed rows as the finite seal budget package, while its boundary clause leaves quotient streams, selected host limits, host real equality, and ambient completeness outside the scoped surface. □

Theorem 425.23 (CauchyErrorCertificate public export)**.** *For an accepted* $\mathsf{CauchyErrorCertificate}^{\uparrow}$ *packet* $C = (R, M, T, E, P, N)$*, the carrier, the* $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}^{\uparrow}}$ *obligations, modulus soundness, budget-window exhaustion, shared modulus transport, readback route uniqueness, Real consumer unblock, seal budget package, and scoped budget-window package assemble the public* $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}^{\uparrow}}$ *export for* $\mathsf{Real}^{\uparrow}$ *completion-facing readers.*

The exported consumer receives exactly R, M, T, E, P, N *together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. It receives no quotient stream, selected host limit, ambient completeness row, host real equality, or bridge checking.*

Proof. First bind the displayed packet. The carrier of Definition 425.1 supplies R, M, T, E, P, N, and Definition 425.2 fixes the local $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}^{\uparrow}}$ obligation surface over those rows.

Next bind the error-budget route. Theorem 425.3 sends each tolerance request through M and T into E, while Theorem 425.13 ensures that the budget route is exhausted before the readback boundary is available.

Now bind transport and readback. Theorem 425.14 preserves the shared modulus route under the displayed hsame and Cont rows, and Theorem 425.10 fixes the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback boundary relative to the displayed M, T, E, P rows.

Finally repack the public export. Theorem 425.17 states the consumer-facing package, Theorem 425.18 gathers it for completion-facing readers, and Theorem 425.22 supplies the scoped BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The result is a public export because it exposes only the finite certificate surface already displayed by the packet. The non-escape boundary of Definition 425.5 leaves quotient streams, selected host limits, ambient completeness, host real equality, and bridge checking outside the export, so the theorem does not assert a new host real fact. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}\uparrow}$.

Constructive story: A $\mathsf{CauchyErrorCertificate}^{\uparrow}$ packet is generated from a regular-rational and real readback route, a unary precision modulus, a dyadic tail or diagonal row, a finite error-budget ledger, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}\uparrow}, \mathsf{scopedClosed})$

Scope closed: The chapter displays the carrier, NameCert obligations, modulus soundness route, budget-window exhaustion, shared modulus transport, readback determinacy, and finite seal budget package for dyadic error certificates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyErrorCertificate}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyErrorCertificateUp.CauchyErrorCertificatePacket\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected host limits, ambient real completeness, host real equality, or bridge checking.

Upgrade path: Further formal closure requires audit-clean and axiom-clean verification of the same scoped budget-window package.

Chapter 426

A Concrete Naming Certificate for $\mathsf{CompletionFunctor}^{\uparrow}$

The $\mathsf{CompletionFunctor}^{\uparrow}$ surface names the finite functorial completion packet that transports regular rational Cauchy data and continuous maps through the existing completion envelope. Following Bishop and Bridges, *Constructive Analysis*, it treats completion by explicit Cauchy data, moduli, and extension ledgers rather than by quotient completions or equality of host functions. The packet connects the completion monad of chapter 378, the universal-property surface of chapter 397, the real Cauchy completion surface of chapter 360, and the reflection boundary of chapter 403.

Definition 426.1 (CompletionFunctor carrier)**.** *A* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, U, R, Q, A, G, E, T, H, C, P, N)$$

with the following displayed coordinates:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-envelope row that supplies unit and bind-facing completion reads;*
2. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row for continuous maps out of the dense regular rational surface;*
3. *R is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *real-completion row that seals regular rational Cauchy input as a* $\mathsf{Real}^{\uparrow}$ *read;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose finite stream windows supply the dense observations;*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *target seal row exposed by the completed source;*
6. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *dense map row, including its modulus, graph-read ledger, and finite input-output observation rows;*
7. *E is the transported extension ledger from G, Q, M, U, R, and A to a completion-facing* $\mathsf{ContinuousMap}^{\uparrow}$ *row;*
8. *T is the functoriality ledger comparing identity and composition reads by displayed* hsame *routes over the same source, map, extension, and seal rows;*
9. *H stores the componentwise* hsame *comparisons for the completion envelope, universal-property row, real-completion row, source row, target seal, dense map, extension ledger, functoriality ledger, package row, and local naming row;*

10. *C stores the* Cont *routes through which consumers read source windows, map graph entries, completion envelopes, extension rows, real seals, and functoriality ledgers;*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and local* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$ *row.*

Two accepted carriers classify together only when all displayed rows are transported componentwise through H and read through the routes in C. The carrier has no quotient of Cauchy streams, no selected host limit, no equality of host functions, and no ambient completeness field.

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorTasteGate single carrier alignment`

Theorem 426.2 (CompletionFunctor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$ *surface for Definition 426.1 consists of carrier admission, classifier stability, extension-ledger exactness, functorial identity and composition ledgers, unit-boundary compatibility, and non-escape of quotient completions, selected limits, host function equality, and ambient completeness. Every obligation is stated over the rows $M, U, R, Q, A, G, E, T, H, C, P, N$ and their displayed* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 426.1. The packet lists the completion envelope, universal-property row, real-completion row, dense regular rational source, real seal, dense continuous map, extension ledger, functoriality ledger, transport rows, continuation rows, package provenance, and local naming row.

Classifier stability is componentwise. The recorded hsame rows in H transport exactly the listed coordinates, while the Cont routes in C preserve the read order from source windows and graph entries to completion envelopes, extension rows, real seals, and functoriality ledgers.

Extension exactness is inherited as a visible row, not as an external principle. The coordinate E is the only route from the dense $\mathsf{ContinuousMap}^{\uparrow}$ row G and regular rational source Q into the completion-facing map row supplied by U.

The remaining public obligations are the three theorem surfaces below: functoriality, unit-boundary compatibility, and non-escape. Together they exhaust all consumer reads of the local $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$ packet. □

Theorem 426.3 (CompletionFunctor functoriality)**.** *Let $F = (M, U, R, Q, A, G, E, T, H, C, P, N)$ be an accepted* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier. Identity reads and composable pairs of dense* $\mathsf{ContinuousMap}^{\uparrow}$ *rows are transported to completion-facing* $\mathsf{ContinuousMap}^{\uparrow}$ *reads by the same finite extension ledger E and compared by the functoriality ledger T. The comparison uses only displayed* hsame *rows and* Cont *routes over M, U, R, Q, A, G, and E.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier functoriality`

Proof. For the identity route, read the dense identity map as a $\mathsf{ContinuousMap}^{\uparrow}$ graph row with its displayed modulus. The universal-property row U and extension ledger E repack that graph row through Q, M, R, and the seal A as a completion-facing continuous map.

For a composable pair, inspect the two dense graph ledgers in finite windows. The Cont routes in C preserve the order of source read, first map read, second map read, extension read, and real-seal read.

The ledger T compares the completion of the composite dense map with the composite of the two completion-facing extension rows. Since T is a displayed BHist coordinate whose comparisons are routed through H and C, the comparison is a local hsame ledger statement and not an equality of host functions. □

Theorem 426.4 (CompletionFunctor unit boundary)**.** *For every accepted* CompletionFunctor$^{\uparrow}$ *carrier, the unit row of* CauchyCompletionMonad$^{\uparrow}$ *and the real seal of* RealCauchyCompletion$^{\uparrow}$ *agree at the public boundary through the displayed rows M, R, Q, A, E, H, C, P, and N. A dense regular rational input is first read as Q, then admitted through the completion envelope M, and finally exposed through the* Real$^{\uparrow}$ *seal A without adding a quotient representative or a selected limit.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier unit boundary`

Proof. Start with the displayed regular rational source row Q. The completion envelope M supplies the unit-facing packet for this source, using finite stream windows and regular rational observations.

The real-completion row R and seal A are then read through the same source observations. The extension ledger E contributes only the continuation route needed by downstream continuous-map consumers; it does not change the source row or choose a host limit.

The comparison between the monad unit boundary and the real seal boundary is recorded by the componentwise transports in H and the read routes in C. Thus the public unit boundary is a finite packet route through the named rows, with no quotient completion or ambient completeness datum. □

Theorem 426.5 (CompletionFunctor extension ledger exactness)**.** *Let $F = (M, U, R, Q, A, G, E, T, H, C, P, N)$ be an accepted* CompletionFunctor$^{\uparrow}$ *carrier. Every completion-facing extension read produced from the dense* ContinuousMap$^{\uparrow}$ *row G and regular rational source row Q is exactly the displayed extension ledger E read through M, U, R, A, H, C, P, and N. The packet exports no second extension route, quotient completion, selected limit, or host function equality.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier extension ledger exactness`

Proof. Project the accepted carrier through Definition 426.1. The only row whose role is extension from dense graph data to a completion-facing continuous map is E, and its source rows are visibly G, Q, M, U, R, and A.

Apply the extension clause in Definition 426.2. It makes extension exactness a visible carrier row rather than an external principle, so every consumer read of the extended map reaches E through the displayed hsame transports in H and the Cont routes in C.

Finally use the non-escape inventory implicit in the carrier. Since no coordinate of F is a quotient completion, selected limit, host function equality, or ambient completeness field, no second extension object can be read from the packet. □

Theorem 426.6 (CompletionFunctor identity route stability)**.** *For an accepted* CompletionFunctor$^{\uparrow}$ *carrier $F = (M, U, R, Q, A, G, E, T, H, C, P, N)$ whose dense map row G is the identity* ContinuousMap$^{\uparrow}$ *row on the displayed dense source, the identity completion route is stable under the componentwise* hsame *rows in H and the* Cont *routes in C. Its public read is the identity clause of the functoriality ledger T, using the same extension ledger E and unit-boundary rows M, R, Q, and A.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier identity route stability`

Proof. Start with the identity part of Definition 426.3. The dense identity graph is read as a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row, then repacked by U and E through the source row Q and completion envelope M.

The unit boundary of Definition 426.4 identifies the source-side completed object with the $\mathsf{Real}^{\uparrow}$ seal A through M, R, and Q. Transporting the same rows by H and reading them through C keeps the identity route attached to the displayed source and seal rows.

The functoriality ledger T is therefore the only public identity comparison row. The route is stable as a finite BHist comparison and does not become an equality of host functions. □

Theorem 426.7 (CompletionFunctor composition route stability)**.** *For an accepted* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier* $F = (M, U, R, Q, A, G, E, T, H, C, P, N)$ *that displays composable dense* $\mathsf{ContinuousMap}^{\uparrow}$ *reads inside* G *and their transported extension rows inside* E*, the completion route for the composite is classified by the composition clause of* T*. Componentwise transport through* H *and consumer reads through* C *preserve the same route through* M*,* U*,* R*,* Q*,* A*,* G*,* E*,* P*, and* N*.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier composition route stability`

Proof. Read the composable dense graph rows through the second paragraph of Definition 426.3. The carrier records the source read, first graph read, second graph read, extension read, and real-seal read as finite routes.

Apply Definition 426.5. Both completion-facing maps and the completion-facing composite are read through E, so the composition comparison has a single extension ledger to inspect.

The ledger T then compares the extension of the dense composite with the composite of the two transported extension rows. Because T is transported by the same H and read by the same C, the composition route is stable without appealing to host function equality. □

Theorem 426.8 (CompletionFunctor completion consumer surface)**.** *Every public completion consumer of an accepted* $\mathsf{CompletionFunctor}^{\uparrow}$ *packet factors through the displayed carrier rows* $M, U, R, Q, A, G, E, T, H, C, P, N$*: completed-source reads use the unit boundary, extension reads use the exact ledger* E*, identity reads use the identity clause of* T*, composition reads use the composition clause of* T*, and all reads obey the non-escape boundary.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier completion consumer surface`

Proof. First split the consumer request by the row it asks to read. A completed-source request reaches M, R, Q, and A by Definition 426.4; an extension request reaches E by Definition 426.5.

For functorial requests, use the two route-stability rows. Identity reads are governed by Definition 426.6, while composition reads are governed by Definition 426.7.

Finally apply Definition 426.10. The listed rows exhaust the finite carrier, so the consumer surface contains no quotient completion, selected limit, hidden representative, host function equality, or ambient completeness field. □

Theorem 426.9 (CompletionFunctor scoped dependency spine)**.** *The carrier, exact extension ledger, unit boundary, identity route, composition route, and consumer surface of* $\mathsf{CompletionFunctor}^{\uparrow}$ *are scoped exactly over* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$ *row. No proof obligation of the packet reads outside these named dependencies.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier scoped dependency spine`

Proof. Project the finite carrier and local naming surface from Definition 426.2. This fixes the available rows as $M, U, R, Q, A, G, E, T, H, C, P, N$ and identifies their sources as the completion monad, universal-property, real-completion, regular-sequence, real-seal, continuous-map, transport, continuation, package, and local naming dependencies.

Apply Definition 426.5, Definition 426.4, Definition 426.6, and Definition 426.7. These four rows show that every extension, unit, identity, and composition read factors through the displayed ledger and the same hsame and Cont routes.

Finally use Definition 426.8. The consumer surface exhausts the packet, so quotient completion, selected limit, ambient completeness, hidden representatives, and host function equality are not in scope for $\mathsf{CompletionFunctor}^{\uparrow}$. □

Theorem 426.10 (CompletionFunctor non-escape boundary)**.** *No accepted* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier exports quotient completions, host function equality, selected limits, hidden representatives, or ambient completeness. Every public read of a completed object, a transported continuous map, an identity ledger, or a composition ledger factors through the finite* BHist *packet of Definition 426.1, the extension route of Definition 426.2, the functoriality ledger of Definition 426.3, and the unit boundary of Definition 426.4.*

Proof. Inspect the carrier inventory. Its analytic rows are the completion envelope, universal-property row, real-completion row, regular rational source, real seal, dense continuous map, extension ledger, and functoriality ledger, together with finite transport, continuation, package, and naming rows.

The NameCert obligation theorem makes E the only completion-facing extension route. The functoriality theorem makes T the only public route for identity and composition comparison. The unit-boundary theorem makes M, R, Q, and A the only public route from dense regular rational input to completed real seal.

These routes exhaust the packet. Since none of the listed coordinates is a quotient completion, host function equality proof, selected limit, hidden representative, or ambient completeness field, no public consumer can read such an object from the $\mathsf{CompletionFunctor}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier non escape`

Theorem 426.11 (CompletionFunctor public completion export)**.** *The* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier, functoriality ledger, unit boundary, extension-ledger exactness, identity route stability, composition route stability, completion consumer surface, scoped dependency spine, and non-escape boundary assemble into the public* $\mathsf{CompletionFunctor}^{\uparrow}$ *completion export consumed by* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, *and* $\mathsf{CompletionUniversalProperty}^{\uparrow}$. *This export exposes only the finite rows* $M, U, R, Q, A, G, E, T, H, C, P, N$ *of Definition 426.1 and their displayed* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier public completion export`

Proof. Start from the local carrier and obligation surface of Definition 426.2. It fixes the displayed rows, the componentwise classifier transports, the continuation routes, and the local $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$ packet read by downstream completion consumers.

The public completion map surface is then determined by the three central construction rows. The functoriality ledger is supplied by Definition 426.3, the unit-facing dense-to-completed boundary by Definition 426.4, and the single completion-facing extension read by Definition 426.5.

The route-level stability needed by consumers is already internal to the same packet. Identity reads use Definition 426.6, and composable dense-map reads use Definition 426.7, both over the same extension ledger and transport rows.

Finally combine the consumer inventory of Definition 426.8 with the dependency inventory of Definition 426.9 and the non-escape boundary of Definition 426.10. The resulting public export is exactly the completion-functor surface available to $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, and $\mathsf{CompletionUniversalProperty}^{\uparrow}$, with no quotient completion, selected limit, host function equality, hidden representative, or ambient completeness field added at the boundary. □

Theorem 426.12 (CompletionFunctor public extension interface)**.** *Let $F = (M, U, R, Q, A, G, E, T, H, C, P, N)$ be an accepted* $\mathsf{CompletionFunctor}^{\uparrow}$ *carrier. The packet exports a public completion-functor interface in which unit reads come from the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row M, extension reads come from the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row U and the exact ledger E,* $\mathsf{Real}^{\uparrow}$ *seals come from the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows R and Q together with A, and map transport comes from the* $\mathsf{ContinuousMap}^{\uparrow}$ *row G and functoriality ledger T. Every such read is routed through displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier public extension interface`

Proof. For the unit side, apply Definition 426.4. It identifies the public completed-source read as the route through M, R, Q, A, H, C, P, and N.

For extension reads, use Definition 426.5. The universal-property row U and exact ledger E give the only completion-facing extension path from the dense $\mathsf{ContinuousMap}^{\uparrow}$ row G.

For transported maps, combine Definition 426.3, Definition 426.6, and Definition 426.7. These theorems route identity and composition through T using the same displayed carrier rows.

Finally invoke Definition 426.8 and Definition 426.10. They exhaust the public consumer surface and rule out quotient completions, selected limits, hidden representatives, host function equality, and ambient completeness. □

Theorem 426.13 (CompletionFunctor Cauchy seal composition)**.** *Let*

$$F_1 = (M_1, U_1, R_1, Q_1, A_1, G_1, E_1, T_1, H_1, C_1, P_1, N_1)$$

and

$$F_2 = (M_2, U_2, R_2, Q_2, A_2, G_2, E_2, T_2, H_2, C_2, P_2, N_2)$$

be accepted $\mathsf{CompletionFunctor}^{\uparrow}$ *carriers whose target seal row of F_1 matches the source seal row of F_2 through displayed* hsame *and* Cont *routes. Then their completion-extension packets compose to a single* $\mathsf{CompletionFunctor}^{\uparrow}$ *packet whose composed route preserves carrier membership, classifier equality, and extension-ledger exactness. The public surface of the composite reads the same* $\mathsf{Real}^{\uparrow}$ *seal boundary, the same completion-facing* $\mathsf{ContinuousMap}^{\uparrow}$ *classifier row, and the exact extension ledger exported by the composed packet.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier cauchy seal composition`

Proof. First build the composed route. Apply Definition 426.7 to the composable dense map rows and their transported extension rows; the matching seal hypothesis identifies the target read of F_1 with the source read of F_2 inside the displayed hsame and Cont routes.

Next descend the ledger. Apply Definition 426.5 to each packet and then to the composed carrier route. The only completion-facing extension reads available are the displayed ledgers E_1, E_2, and the composed ledger carried by the resulting packet.

Finally export the consumer surface. Use Definition 426.12 to read the composed unit, seal, map-transport, and functoriality rows through the same package and naming provenance. This gives a single public $\mathsf{CompletionFunctor}^{\uparrow}$ packet without adding quotient completions, selected limits, host function equality, or ambient completeness. □

Theorem 426.14 (CompletionFunctor standard bridge surface)**.** *Every standard* $\mathsf{CompletionFunctor}^{\uparrow}$ *read over the carrier rows* $M, U, R, Q, A, G, E, T, H, C, P, N$ *factors through the public extension interface of Definition 426.12, the unit boundary of Definition 426.4, the extension-ledger exactness of Definition 426.5, the identity and composition route stability of Definition 426.6 and Definition 426.7, and the Cauchy seal composition of Definition 426.13. The surface exports no quotient completion, selected limit, host function equality, hidden representative, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier standard bridge surface`

Proof. Start with the public interface. Definition 426.12 exposes the unit, extension, seal, map-transport, functoriality, package, and naming rows of the accepted carrier and routes every public consumer through displayed hsame and Cont data.

Next identify the construction rows used by that interface. The unit side is Definition 426.4; the extension side is Definition 426.5; the functorial side is the pair of identity and composition route-stability theorems.

Finally apply Definition 426.13. It shows that compatible Cauchy seal reads compose inside the same finite packet, so the standard bridge surface is exactly the paper-side surface generated by these named rows. The non-escape clauses already present in the interface exclude quotient completions, selected limits, host function equality, hidden representatives, and ambient completeness. □

Theorem 426.15 (CompletionFunctor bridge-composition exactness)**.** *Any consumer of the* $\mathsf{CompletionFunctor}^{\uparrow}$ *standard bridge reads the same completion carrier through identity routes, composition routes, extension-ledger exactness, and Cauchy seal composition over the rows*

$$M, U, R, Q, A, G, E, T, H, C, P, N.$$

The bridge-composition read exposes no extra host completion, quotient object, selected limit, host function equality, hidden representative, or ambient completeness field beyond the standard bridge surface.

Proof. Start with the identity route. Definition 426.6 keeps the identity completion read attached to the displayed source, seal, extension ledger, transport rows, and continuation routes.

Next add the composition route and extension ledger. Definition 426.7 compares composable dense-map reads inside the same functoriality ledger, and Definition 426.5 says that every completion-facing extension read is exactly the displayed ledger E.

Now apply Cauchy seal composition. Definition 426.13 composes compatible seal reads inside a single $\mathsf{CompletionFunctor}^{\uparrow}$ packet and preserves carrier membership, classifier equality, and extension-ledger exactness.

Finally cite the standard bridge surface. Definition 426.14 is generated by these identity, composition, ledger, and seal rows, and its non-escape clauses exclude any extra host completion or quotient object. □

Lean4 ✓ `BEDC.Derived.CompletionFunctorUp.CompletionFunctorCarrier bridge composition exactness`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionFunctor}^{\uparrow}$ packet is generated from a $\mathsf{CauchyCompletionMonad}^{\uparrow}$ completion envelope, a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ extension row, a $\mathsf{RealCauchyCompletion}^{\uparrow}$ real-completion row, a $\mathsf{RegSeqRat}^{\uparrow}$ dense source, a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{ContinuousMap}^{\uparrow}$ dense map, a finite extension ledger, a functoriality ledger, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionFunctor}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 426.12 exports the public completion-functor interface over CauchyCompletionMonad$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ContinuousMap$^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompletionFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completions, host function equality, selected limits, ambient completeness, completed-category laws beyond the displayed finite ledger, or bridge checking.

Upgrade path: Bridge closure requires a standard bridge for the displayed completion-functor packet.

Chapter 427

A Concrete Naming Certificate for $\mathsf{RegularLimitUniqueness}^{\uparrow}$

The $\mathsf{RegularLimitUniqueness}^{\uparrow}$ interface records the separated uniqueness side of regular-Cauchy completion. In the Bishop–Bridges constructive-analysis presentation, a completed real is read through finite regular-name windows and explicit moduli; uniqueness is therefore a visible separated-seal certificate over two accepted completion candidates, not a quotient of streams or a selected host limit.

The packet compares two diagonal completion candidates only after their regular rational readbacks from chapter 301 pass through the finite observation windows of chapter 126, the diagonal route of chapter 359, the shared family surface of chapter 373, the separated boundary of chapter 402, and the $\mathsf{Real}^{\uparrow}$ seal rows of chapter 38.

Definition 427.1 (RegularLimitUniqueness carrier)**.** *A* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N)$$

with the following visible rows:

1. *F is the shared* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *input row;*
2. *D_0 and D_1 are two* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidate limit rows for the same F;*
3. *μ is the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row controlling the tail windows compared by both candidates;*
4. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed by the two diagonal candidates at the threshold selected by μ;*
5. *E_0 and E_1 are the two* $\mathsf{Real}^{\uparrow}$ *seal rows reached from those readbacks;*
6. *S is the separated comparison row, read through the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *boundary, stating that E_0 and E_1 classify at the same separated real seal;*
7. *H records the displayed* hsame *transports tying F, μ, the two diagonal rows, the two readbacks, the two seals, and the separated comparison together;*
8. *C records the displayed* Cont *routes through the shared family, threshold row, diagonal candidates, readback rows, real seals, and separated comparison;*

9. *P is the* Pkg *provenance over* RegularCauchyFamily$^\uparrow$, CauchyModulus$^\uparrow$, RealDiagonalCompletion$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, HausdorffCompletion$^\uparrow$, Real$^\uparrow$, BHist, hsame, *and* Cont*;*
10. *N is the local* NameCert$_{\text{RegularLimitUniqueness}^\uparrow}$ *row.*

The carrier has no coordinate for quotient streams, a chosen host limit, host real equality, or choice. Its public data are exactly the displayed finite rows above.

Theorem 427.2 (RegularLimitUniqueness NameCert obligations)**.** *The local* NameCert$_{\text{RegularLimitUniqueness}^\uparrow}$ *surface has five finite obligations:*

1. *habitation by a displayed diagonal candidate pair over the carrier of Definition 427.1;*
2. *classifier laws obtained by componentwise* hsame *transport of the shared family, threshold row, diagonal candidates, readbacks, real seals, separated comparison, continuation routes, package provenance, and local naming row;*
3. *carrier transport through the displayed routes from* RegularCauchyFamily$^\uparrow$ *to* RealDiagonalCompletion$^\uparrow$*, then through* RegSeqRat$^\uparrow$ *and* StreamName$^\uparrow$ *readbacks to the two* Real$^\uparrow$ *seals;*
4. *exactness through the separated* Real$^\uparrow$ *classifier row S, so the two accepted candidates are compared only at the displayed separated seal;*
5. *ledger non-escape excluding quotient streams, selected host limits, host real equality, and choice.*

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier namecert obligations`

Proof. Carrier habitation is projection of Definition 427.1. The shared family row, the two diagonal candidates, the common threshold, the two readbacks, the two seals, the separated comparison, transport rows, continuation routes, package provenance, and local certificate are finite coordinates of the same BHist packet.

Classifier stability is componentwise. Transport the shared RegularCauchyFamily$^\uparrow$ and CauchyModulus$^\uparrow$ rows first, then move the two RealDiagonalCompletion$^\uparrow$ rows and their RegSeqRat$^\uparrow$ readbacks along the displayed hsame rows while replaying the corresponding Cont routes.

Exactness is the separated-seal step. The only row that compares the two completion candidates after readback is S, and S is a displayed Real$^\uparrow$ classifier row read through the HausdorffCompletion$^\uparrow$ boundary.

The non-escape clauses follow from the carrier list. No coordinate contains a quotient of regular streams, a selected host limit, host real equality, or a choice principle, so the local certificate can export only the listed BEDC rows. □

Theorem 427.3 (RegularLimitUniqueness separated limit certificate)**.** *Let U be an accepted* RegularLimitUniqueness$^\uparrow$ *carrier. Suppose two accepted* RealDiagonalCompletion$^\uparrow$ *candidates in U read the same* RegularCauchyFamily$^\uparrow$ *input through the shared* CauchyModulus$^\uparrow$ *threshold and reach the same separated boundary row S. Then their* RegSeqRat$^\uparrow$ *readbacks classify through the same* Real$^\uparrow$ *seal: the two candidate limits are identified exactly by the displayed separated* Real$^\uparrow$ *classifier row, with visible* BHist, hsame, Cont, Pkg, *and* NameCert *provenance.*

The certificate exports no quotient stream comparison, selected host limit, host real equality, ambient completeness principle, or choice.

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier separated limit certificate`

Proof. Project U as in Definition 427.1. Both candidates point to the same $\mathsf{RegularCauchyFamily}^{\uparrow}$ row F and the same $\mathsf{CauchyModulus}^{\uparrow}$ threshold μ, so their readback routes begin at one displayed regular-Cauchy input surface.

Read each candidate through its diagonal row. The two $\mathsf{RealDiagonalCompletion}^{\uparrow}$ coordinates expose the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ readbacks and $\mathsf{StreamName}^{\uparrow}$ windows, and the stored hsame and Cont rows keep those reads attached to the shared family and threshold.

Now use the separated comparison row S. It is the only accepted comparison between E_0 and E_1, and it lives at the $\mathsf{Real}^{\uparrow}$ seal boundary through the $\mathsf{HausdorffCompletion}^{\uparrow}$ separated surface.

Repacking this route gives the stated certificate. Since the carrier contains no quotient, selected host limit, host real equality, ambient completeness, or choice coordinate, the equality claim is precisely the displayed separated classifier claim and nothing stronger. □

Theorem 427.4 (RegularLimitUniqueness classifier determinacy)**.** *Let U and U' be accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *carriers whose displayed* hsame *and* Cont *rows classify the shared* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *input, the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, the two* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidates, their* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, their* $\mathsf{Real}^{\uparrow}$ *seals, and their separated comparison rows. Then the two carriers determine the same separated limit route: every classified candidate pair reaches the corresponding* $\mathsf{Real}^{\uparrow}$ *seals through the displayed separated comparison row and through no additional host equality or chosen limit.*

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier classifier determinacy`

Proof. Project both carriers using Definition 427.1. The family row, threshold row, diagonal candidate rows, readback rows, seal rows, separated comparison rows, transport rows, continuation rows, package rows, and local certificates are all visible coordinates.

Apply the classifier obligations of Definition 427.2. Componentwise transport moves the shared family and threshold first, then moves each diagonal candidate, readback, seal, and separated comparison row along the displayed routes.

Now apply Definition 427.3 to either classified carrier. The separated comparison row is the only route that identifies the two candidate seals, so classification determines the same separated limit route without adding host equality, a quotient stream, or a selected limit. □

Theorem 427.5 (RegularLimitUniqueness completion consumer boundary)**.** *Every completion consumer that reads a* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet can inspect only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidate rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, shared family and threshold rows, separated comparison row, and the recorded* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance. The packet exports no quotient completion object, selected host limit, host real equality, ambient completeness principle, or choice witness.*

Proof. Read the carrier inventory. The completion-facing coordinates are the two $\mathsf{RealDiagonalCompletion}^{\uparrow}$ candidates and their $\mathsf{RegSeqRat}^{\uparrow}$ readbacks over the shared family and threshold, followed by the two $\mathsf{Real}^{\uparrow}$ seal rows and the separated comparison row.

The separated-limit certificate of Definition 427.3 consumes exactly those rows. It compares the candidate seals only after their readbacks have passed through the displayed diagonal and stream-name routes.

The non-exported objects are absent from the carrier. Since no coordinate contains a quotient completion, selected host limit, host real equality, ambient completeness theorem, or choice witness, no completion consumer can obtain one from the packet. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier completion consumer boundary`

Theorem 427.6 (RegularLimitUniqueness separated transport)**.** *Separatedness in a* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *carrier transports through the displayed* hsame *and* Cont *rows. If the separated comparison row* S *identifies the two* $\mathsf{Real}^{\uparrow}$ *seals reached from one accepted carrier, then any classified carrier carries the transported separated comparison for the transported seals, without introducing host-level equality or a quotient of regular streams.*

Proof. Begin with the separated comparison row S in an accepted carrier. By Definition 427.3, S sits after the shared family, threshold, diagonal candidates, regular readbacks, and two real seal rows.

Transport each upstream coordinate through the displayed hsame rows and replay the corresponding Cont routes. The classifier obligations of Definition 427.2 keep the separated comparison attached to the transported seals and to the same package and local certificate rows.

The transported statement is again a separated $\mathsf{Real}^{\uparrow}$ classifier claim. It does not become a host equality statement, and it does not require a quotient of the regular-stream readbacks. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier separated transport`

Theorem 427.7 (RegularLimitUniqueness obligation closure package)**.** *The* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet of Definition 427.1, together with the local* $\mathsf{NameCert}_{\mathsf{RegularLimitUniqueness}^{\uparrow}}$ *obligations of Definition 427.2, the separated-limit certificate of Definition 427.3, classifier determinacy of Definition 427.4, the completion consumer boundary of Definition 427.5, and separated transport of Definition 427.6, presents the five displayed obligation rows required by the regular-limit uniqueness surface:*

1. *carrier habitation for the finite row*

$$U = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N);$$

2. *classifier transport along the recorded* hsame *rows and replayed* Cont *routes for* $F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H,$
3. *readback-to-seal exactness from each* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidate through its displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback into the corresponding* $\mathsf{Real}^{\uparrow}$ *seal;*
4. *separated comparison exactness through the single displayed row* S *at the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *boundary;*
5. *ledger non-escape for quotient streams, selected host limits, host real equality, ambient completeness, and choice witnesses.*

These rows are sufficient as the paper-side obligation package for later scoped closure of the same regular-limit uniqueness claim.

Proof. The carrier habitation row is the finite packet displayed in Definition 427.1. The first NameCert obligation of Definition 427.2 supplies the accepted carrier instance over that packet.

Classifier transport is the second and third obligation row of Definition 427.2, refined by Definition 427.4 and Definition 427.6. These results keep the family, threshold, diagonal candidates, readbacks, real seals, separated comparison, package row, and local naming row attached to the same displayed BHist surface.

Readback-to-seal exactness follows from Definition 427.3. Each diagonal candidate is read through its $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows before it reaches a $\mathsf{Real}^{\uparrow}$ seal, so the certificate never compares unsealed host objects.

Separated comparison exactness is the role of S. The comparison occurs only at the separated $\mathsf{Real}^{\uparrow}$ classifier row, read through the $\mathsf{HausdorffCompletion}^{\uparrow}$ boundary.

Ledger non-escape is the completion-facing boundary of Definition 427.5. Every consumer route is a route through the listed BEDC rows, so the packet exports no quotient stream, selected host limit, host equality, ambient completeness theorem, or choice witness. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier obligation closure package`

Theorem 427.8 (RegularLimitUniqueness readback seal exactness)**.** *Let $U = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C$ be an accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *carrier. If D_0 and D_1 are two diagonal completion candidates over the same regular family row F and threshold row μ, then the packet can identify the candidates only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R_0, R_1 and the separated* $\mathsf{Real}^{\uparrow}$ *seal row S comparing E_0 and E_1.*

No identification route in the packet bypasses R_0, R_1, E_0, E_1, S, and no route factors through host real equality, a selected host limit, a quotient stream, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier readback seal exactness`

Proof. Project the accepted carrier. The two candidates share the same F and μ coordinates, and their readback rows are explicitly R_0 and R_1.

The diagonal rows expose only the displayed regular-rational readbacks before reaching real seals. The hsame and Cont coordinates keep those readbacks tied to the same family and threshold, but they do not add a separate equality object.

The separated comparison row is S. By Definition 427.3, candidate identification occurs after the readbacks reach E_0 and E_1, and the comparison is exactly the separated $\mathsf{Real}^{\uparrow}$ classifier row.

The exclusion statement follows from the ledger boundary in Definition 427.5. The omitted host and quotient objects are not coordinates of U. □

Theorem 427.9 (RegularLimitUniqueness dyadic-window separation pullback)**.** *Let $U = (F, D_0, D_1, \mu, R_0, R_1, E_0$ be an accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet. Suppose the two candidate routes share the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger read by those windows, and the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated comparison row whose endpoint is the* $\mathsf{Real}^{\uparrow}$ *seal comparison S.*

Then every completion consumer reaches the same $\mathsf{Real}^{\uparrow}$ *seal comparison through that finite pullback surface. The consumer route is forced through R_0, R_1, E_0, E_1, S and through the shared* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of U; it cannot be replaced by quotient streams, selected host limits, host real equality, ambient completeness, or choice.*

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier dyadic window separation pullback`

Proof. First apply Definition 427.8. The accepted packet identifies the two diagonal candidates only after the displayed readbacks R_0, R_1 reach the seal rows E_0, E_1 and the separated comparison row S.

Next apply Definition 427.12. The common family row F, threshold row μ, both diagonal candidates, and their transport, continuation, package, and naming rows remain inside the same $\mathsf{RegularLimitUniqueness}^{\uparrow}$ packet.

The finite pullback is supplied by Definition 126.356 and Definition 126.358. The shared $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal rows determine one finite surface for downstream seal comparison.

Finally use the completion consumer boundary from Definition 427.5. Since every consumer has already been routed through the displayed finite surface, no quotient stream, selected host limit, host equality, ambient completeness row, or choice witness can enter the uniqueness comparison. □

Theorem 427.10 (RegularLimitUniqueness seal-selector pullback)**.** *Let* $U = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C,$ *be an accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet. Suppose the two candidate routes share the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidate rows, the* $\mathsf{DiagonalLimit}^{\uparrow}$ *completion-seal factorization row, the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated row, and the* $\mathsf{Real}^{\uparrow}$ *seal comparison row* S*.*

Then every uniqueness consumer factors through the single finite selector-pullback surface

$$F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N.$$

The surface is obtained by first routing the comparison through the dyadic-window separation pullback, then through the checked $\mathsf{DiagonalLimit}^{\uparrow}$ *completion-seal factorization, and finally through the* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *completion-consumer boundary. It exports no quotient stream, selected host limit, host real equality, ambient completeness principle, or choice witness.*

Proof. Project the accepted packet U. Its carrier exposes exactly the shared family row, the two diagonal completion candidates, the threshold row, the readback rows, the two seal rows, the separated comparison row S, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Apply Definition 427.9. This routes each uniqueness consumer through the common $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and separated $\mathsf{Real}^{\uparrow}$ comparison row before any completion-facing comparison is exposed.

Next compose that finite window route with Definition 374.8. The checked $\mathsf{DiagonalLimit}^{\uparrow}$ factorization makes the $\mathsf{Real}^{\uparrow}$ seal read pass through the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ packet and the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows, so the selector route is the same finite pullback surface rather than an independent host-limit route.

Finally use Definition 427.5. Since the consumer boundary lists only the displayed carrier, transport, continuation, package, and naming rows, the composed selector-pullback route cannot import quotient streams, selected host limits, host real equality, ambient completeness, or choice. □

Theorem 427.11 (RegularLimitUniqueness choice-free ledger exhaustion)**.** *Every completion consumer route exported by an accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet is exhausted by*

$$F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N,$$

by the ambient BHist *packet structure, by* hsame *transport, by* Cont *continuation, by* Pkg *provenance, and by the local* $\mathsf{NameCert}_{\mathsf{RegularLimitUniqueness}^{\uparrow}}$ *row. The packet contains no quotient stream, selected host limit, host equality, ambient completeness principle, or choice witness.*

Proof. Inspect the carrier in Definition 427.1. Its named coordinates are exactly the displayed family, candidate, threshold, readback, seal, comparison, transport, continuation, package, and local certificate rows.

The consumer boundary theorem Definition 427.5 says that a completion consumer can inspect only those rows and their BHist, hsame, Cont, Pkg, and NameCert provenance. The separated transport theorem Definition 427.6 keeps the same inventory after classifier movement.

Therefore every exported route is already listed in the carrier ledger. Since quotient streams, selected host limits, host equality, ambient completeness, and choice are not listed coordinates, no consumer route can recover them from the packet. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier choice free ledger exhaustion`

Theorem 427.12 (RegularLimitUniqueness shared family scope)**.** *The carrier and classifier rows of Definition 427.1, Definition 427.2, and Definition 427.4 form the shared-family scope for* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *This scope binds the common* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *input, the two* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidate rows, their* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readbacks, the two* $\mathsf{Real}^{\uparrow}$ *seal rows, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *provenance into one finite uniqueness packet.*

The scope exports the family comparison needed by the L10 dependency stack without adding a quotient of streams, a selected host limit, host real equality, or choice.

Proof. Start with the carrier inventory of Definition 427.1. Its coordinates contain the shared family row, both diagonal candidates, both readbacks, both real seals, and the displayed transport, continuation, package, and naming rows.

Apply the local naming obligations of Definition 427.2. They identify habitation, classifier transport, carrier transport, separated exactness, and ledger non-escape as rows of that same finite packet.

Finally use Definition 427.4. Classification cannot move the two candidates outside the shared family surface, so the listed rows are the complete shared-family scope for the regular-limit uniqueness consumer. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier shared family scope`

Theorem 427.13 (RegularLimitUniqueness separated comparison scope)**.** *The separated comparison rows of Definition 427.3, Definition 427.6, and Definition 427.8 bind the uniqueness comparison to the concrete route*

$$\mathsf{RegularCauchyFamily}^{\uparrow} \longrightarrow \mathsf{RealDiagonalCompletion}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

with displayed BHist, hsame, Cont, Pkg, *and* NameCert *provenance. Thus the separated seal comparison is scoped exactly by the two readbacks, the two real seals, and the row* S*; it has no bypass through host equality or an unrecorded completion object.*

Proof. First read the two completion candidates through Definition 427.3. The certificate forces both candidates to share the same family and threshold before their readbacks reach the two real seals.

Next apply Definition 427.8. It makes R_0, R_1, E_0, E_1, S the only identification route available inside the packet.

The transport step is supplied by Definition 427.6. Transport preserves the separated comparison as a $\mathsf{Real}^{\uparrow}$ classifier row, so the displayed route is the entire separated comparison scope. □

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier separated comparison scope`

Theorem 427.14 (RegularLimitUniqueness ledger non-escape scope)**.** *The completion consumer boundary of Definition 427.5, the obligation package of Definition 427.7, and the ledger exhaustion row of Definition 427.11 give the ledger non-escape scope for* $\mathsf{RegularLimitUniqueness}^{\uparrow}$*. Every exported uniqueness route is a route through* $\mathsf{RegularCauchyFamily}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

No exported row leaves this scope for quotient stream equality, selected host limits, host real equality, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier ledger non escape scope`

Proof. Use Definition 427.5 to identify the rows visible to a completion consumer. The boundary lists only the family, candidate, readback, seal, separated comparison, transport, continuation, package, and naming data.

Apply Definition 427.7. It repacks those visible rows as carrier habitation, classifier transport, readback-to-seal exactness, separated comparison exactness, and ledger non-escape.

Finally apply Definition 427.11. Since every exported route is already exhausted by the displayed coordinates, no unlisted host or quotient object can enter the scoped uniqueness packet. □

Theorem 427.15 (RegularLimitUniqueness scoped handoff)**.** *The shared-family scope of Definition 427.12, the separated-comparison scope of Definition 427.13, and the ledger non-escape scope of Definition 427.14 jointly bind every* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *consumer route to* $\mathsf{RegularCauchyFamily}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Within this scoped handoff, no consumer may replace the displayed rows by a quotient stream, selected host limit, host equality, ambient completeness principle, or choice witness.

Lean4 ✓ `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier scoped handoff`

Proof. Begin with the shared-family scope. It fixes the common $\mathsf{RegularCauchyFamily}^{\uparrow}$ input, both $\mathsf{RealDiagonalCompletion}^{\uparrow}$ candidates, their $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ readbacks, the two $\mathsf{Real}^{\uparrow}$ seal rows, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Next use the separated-comparison scope. It binds the uniqueness comparison to the route from the shared family through diagonal completion and regular-rational readback into the two real seals, with the row S as the only separated comparison.

Finally apply the ledger non-escape scope together with the NameCert obligation package of Definition 427.2. The non-escape row excludes quotient streams, selected host limits, host equality, ambient completeness, and choice, so the three scoped rows hand off exactly the displayed consumer surface and nothing outside it. □

Theorem 427.16 (RegularLimitUniqueness public completion export)**.** *The carrier of Definition 427.1, the NameCert obligations of Definition 427.2, the separated-limit certificate of Definition 427.3, classifier determinacy from Definition 427.4, readback seal exactness from Definition 427.8, the shared-family scope of Definition 427.12, the separated-comparison scope of Definition 427.13, the scoped handoff of Definition 427.15, and the ledger non-escape row of Definition 427.14 assemble a single public completion-consumer export for two diagonal completion candidates over one regular-Cauchy family.*

The exported route is exactly

$$\mathsf{RegularCauchyFamily}^{\uparrow} \longrightarrow \mathsf{RealDiagonalCompletion}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the separated row S. It exports no quotient stream comparison, selected host limit, host equality, ambient completeness principle, or choice witness.

Proof. Start from the carrier and NameCert obligation package. They provide the shared family row, the two diagonal completion candidates, the shared threshold, the two readbacks, the two real seals, the separated row S, and the displayed BHist, hsame, Cont, Pkg, and NameCert provenance.

Next apply the separated-limit certificate, classifier determinacy, and readback seal exactness. These rows identify the only equality-facing comparison as the path from the two diagonal candidates through their $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ readbacks into the two $\mathsf{Real}^{\uparrow}$ seals, with S as the separated comparison row.

Finally use the shared-family scope, separated-comparison scope, scoped handoff, and ledger non-escape scope. Together they make the route public for completion consumers while excluding quotient streams, selected host limits, host equality, ambient completeness, and choice from the exported surface. □

Theorem 427.17 (RegularLimitUniqueness consumer determinacy)**.** *Let*

$$A = (F, D0, D1, \mu, R0, R1, E0, E1, S, H, C, P, N)$$

be an accepted $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet. Any two completion consumers that read the public completion export of Definition 427.16 determine the same separated* $\mathsf{Real}^{\uparrow}$ *seal comparison through the displayed row S. No such consumer can distinguish the two reads through quotient stream equality, selected host limits, host real equality, ambient completeness, or choice.*

Proof. Begin with classifier determinacy and readback seal exactness. These results identify the comparison route from the shared family F through the two diagonal completion rows $D0, D1$, the readback rows $R0, R1$, the seal rows $E0, E1$, and the separated row S.

Next apply the scoped handoff and the public completion export. They make that route the public completion-consumer surface for the displayed packet, so both consumers read the same separated comparison row rather than independent host data.

Finally use the ledger non-escape scope. It excludes quotient stream equality, selected host limits, host real equality, ambient completeness, and choice from the exported route, leaving S as the unique separated $\mathsf{Real}^{\uparrow}$ seal comparison available to either consumer. □

Theorem 427.18 (RegularLimitUniqueness bridge handoff)**.** *Every bridge consumer comparing two diagonal completion candidates over the same* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *input in a* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet must factor through the displayed rows*

$$F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N.$$

The handoff uses the public completion export of Definition 427.16 and the consumer determinacy of Definition 427.17: the comparison is read from the shared family F through the two diagonal candidates D_0, D_1, their regular-rational readbacks R_0, R_1, the two real seal rows E_0, E_1, and the separated comparison row S, with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *provenance.*

No bridge consumer may import quotient completion, selected host limits, host real equality, ambient completeness, or choice into this handoff. Those objects are outside the carrier ledger and outside the exported completion-consumer surface.

Proof. Compose the checked local rows. The NameCert obligations give carrier habitation, classifier transport, readback routing, separated exactness, and ledger non-escape for the finite packet of Definition 427.1.

Next use the separated-limit certificate, classifier determinacy, and readback-seal exactness. They force every comparison of the two diagonal candidates to pass through R_0, R_1, E_0, E_1, S after the shared family and threshold rows have been fixed.

The public completion export then exposes exactly this route to bridge consumers, and consumer determinacy makes two reads of that export agree at the same separated $\mathsf{Real}^{\uparrow}$ comparison row S.

Finally apply ledger non-escape. Since quotient completion, selected host limits, host real equality, ambient completeness, and choice are not carrier coordinates and are not exported rows, the bridge handoff is precisely the displayed BEDC route. □

Theorem 427.19 (RegularLimitUniqueness mature consumer route)**.** *Let*

$$A = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N)$$

be an accepted $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet whose rows display one shared* $\mathsf{RegularCauchyFamily}^{\uparrow}$ *route, two* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *candidates, readback-to-seal rows through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *a separated comparison row S, componentwise transports,* Cont *routes,* Pkg *provenance, and a local* $\mathsf{NameCert}$ *row. Then A gives a mature consumer route*

$$\mathsf{RegularCauchyFamily}^{\uparrow} \longrightarrow \mathsf{RealDiagonalCompletion}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

No consumer of this packet can replace that route by quotient completion, selected host limits, host real equality, or ambient completeness.

Proof. Start with the bridge handoff of Definition 427.18. It forces every comparison consumer for the two diagonal candidates to factor through the displayed rows $F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N$.

The public completion export of Definition 427.16 supplies the downstream route from the shared family through the two diagonal completion candidates and their readbacks into the two Real seals. Consumer determinacy from Definition 427.17 makes any two reads of that exported route agree at the same separated comparison row S.

Finally combine readback-seal exactness with the non-escape rows recorded in the bridge handoff. Since quotient completion, selected host limits, host real equality, and ambient completeness are outside the finite carrier and outside the exported completion-consumer surface, the displayed route is the mature consumer route for this packet. □

Theorem 427.20 (RegularLimitUniqueness L10 terminal handoff)**.** *Let*

$$A = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N)$$

be an accepted $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet whose public route is read by a current Phase Real and Completion terminal consumer. If that consumer uses the mature route of Definition 427.19, then its visible terminal boundary factors through exactly the finite four-face L10 packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *rows are used only as the separated-comparison consumer of this boundary: they read the shared family, the two diagonal completion candidates, the regular-rational*

readbacks, the two Real seal rows, and the separated comparison row S through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

The handoff exports no quotient completion, selected host limit, host real equality, countable-choice selector, ambient completeness principle, metric-completeness row, or standard bridge row.

Proof. First expose the local completion consumer. Apply Definition 427.18 and Definition 427.19. The accepted packet can be read by downstream consumers only through the displayed route from the shared $\mathsf{RegularCauchyFamily}^{\uparrow}$ family, through the two $\mathsf{RealDiagonalCompletion}^{\uparrow}$ candidates and their $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ readbacks, into the two $\mathsf{Real}^{\uparrow}$ seal rows and the separated comparison row S.

Next attach the current terminal boundary. By Definition 126.368, every StreamName, RegSeqRat, and Real budget cover used by the Phase Real and Completion surface is confined to the four sibling rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Now apply Definition 38.457. The current terminal $\mathsf{Real}^{\uparrow}$ read is accepted exactly when its visible rows factor through the finite four-face packet in the statement, and the converse read adds no further visible coordinate.

Finally use Definition 38.458. Since the mature $\mathsf{RegularLimitUniqueness}^{\uparrow}$ consumer reaches the terminal read through the same four faces and the same displayed coordinates, the consumer boundary is the unique terminal pullback over that packet. The exclusion clauses follow from the inventories in the cited handoff, budget-cover, exactness, and uniqueness theorems: quotient completion, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness rows, and standard bridge rows have no coordinate in the resulting handoff. □

Theorem 427.21 (RegularLimitUniqueness finite-window induction)**.** *Let*

$$A = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N)$$

be an accepted $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet. Suppose a uniqueness read is accepted over a displayed finite list of* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows attached to the shared family row F, the two diagonal candidates D_0, D_1, and the threshold row μ. Then induction on that finite window list reduces the read to the same separated* $\mathsf{Real}^{\uparrow}$ *seal row S.*

At each induction stage the active window is transported through the displayed hsame *rows, replayed through the* Cont *routes, repacked through* Pkg *provenance, and read through the local* NameCert *row. The resulting route stays inside the shared-family scope, the separated-comparison scope, the ledger non-escape scope, and the L10 terminal handoff; it does not introduce quotient streams, selected host limits, host real equality, ambient completeness, or choice.*

Proof. For the empty window list, the read has no active $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ row beyond the carrier data already displayed in A. Apply the shared-family scope and the separated-comparison scope to read the two diagonal candidates through R_0, R_1, E_0, E_1 and the separated row S.

For the cons step, assume the tail list has already been reduced to the same separated seal row S. The head window is a displayed $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ coordinate attached to F, D_0, D_1, μ, so componentwise hsame transport moves it along the accepted carrier without changing the comparison row.

Now replay the transported head through Cont and repack the route through Pkg and NameCert. The replay keeps the head window on the same readback-to-seal path as the tail, so the induction step again reaches R_0, R_1, E_0, E_1, S.

Finally apply the ledger non-escape scope and the L10 terminal handoff. Since every stage remains on the displayed finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, no host equality, quotient stream, selected limit, ambient completeness row, or choice witness can replace the separated seal row S. □

Theorem 427.22 (Separated Real-seal finality)**.** *Let*

$$A = (F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N)$$

be an accepted $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *packet whose comparison row S is read as a separated* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *comparison inside a* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer. Suppose the consumer route has passed through the readback seal rows R_0, R_1, E_0, E_1, the dyadic-window separation of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *boundary, the seal-selector pullback into* $\mathsf{Real}^{\uparrow}$*, the choice-free ledger exhaustion of the displayed route, and the L10 terminal handoff of Definition 427.20.*

Then every accepted limit read of A is final up to the displayed Real classifier: any later read that preserves the same BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows must factor through the same separated S row and hence through the same* $\mathsf{Real}^{\uparrow}$ *classifier surface. The finality route is finite:*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{CompleteMetric}^{\uparrow},$$

with $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *contributing only the shared family, readback-to-seal, separated comparison, transport, continuation, provenance, and local naming rows.*

Proof. Start from the terminal handoff in Definition 427.20. It has already confined the visible Real and completion boundary to the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, with no quotient completion, selected host limit, host real equality, countable-choice selector, ambient completeness principle, or standard bridge row.

Use the public completion export and consumer determinacy theorems to identify the only comparison row available to a downstream reader. The shared family, the two diagonal completion candidates, the regular-rational readbacks, and the two Real seal rows are all read through $F, D_0, D_1, \mu, R_0, R_1, E_0, E_1, S, H, C, P, N$, so a $\mathsf{CompleteMetric}^{\uparrow}$ consumer cannot acquire an additional limit coordinate outside S.

Now apply the separated-comparison part of the carrier. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal reads are already tied to the dyadic-window boundary and seal-selector pullback, while BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserve exactly the displayed rows. Thus any accepted later read that keeps the packet accepted must return to the same Real classifier surface.

The choice-free ledger exhaustion supplies the non-escape step. Since no selected subsequence, quotient class, host equality, Real completeness principle, or hidden metric-completeness witness is a coordinate of A, the displayed classifier is the final public limit read for this separated Real-seal route. □

Theorem 427.23 (Classifier route non-escape)**.** *No accepted* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *consumer may replace the classifier-finality route of Definition 427.22 by host equality, a selected subsequence choice, quotient classes, or a Real completeness principle. Any consumer that claims uniqueness for the two limit reads must exhibit the same finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*, the separated comparison row S, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

Proof. Read the carrier inventory from Definition 427.18 and Definition 427.19. These theorems place the shared family, diagonal completions, readbacks, seals, separated comparison, transport, continuation, provenance, and naming rows on the visible route.

Next use Definition 427.20. The Real-facing boundary has only the finite four-face route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; the $\mathsf{CompleteMetric}^{\uparrow}$ reader can use that route only after it passes through the same separated comparison and support rows.

Finally apply Definition 427.22. Finality is a statement about the displayed Real classifier surface, not about an ambient equality proof. Host equality, selected subsequence choice, quotient classes, and Real completeness have no carrier row, no transport row, no continuation row, no provenance row, and no local NameCert row in A, so replacing the route with one of them leaves the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularLimitUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularLimitUniqueness}^{\uparrow}$ packet is generated from one regular-Cauchy family, two diagonal completion candidates, a shared Cauchy threshold, two regular-rational readbacks, two real seal rows, one separated comparison row, displayed transport and continuation rows, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularLimitUniqueness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 427.18, together with the public completion export and consumer determinacy theorems, records the carrier, classifier transport, shared-family route, readback-to-seal exactness, separated comparison exactness, ledger non-escape rows, and bridge-consumer handoff for two regular-Cauchy completion candidates over the same displayed input and boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularLimitUniqueness}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularLimitUniquenessUp.RegularLimitUniquenessCarrier\_namecert\_obligations`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient completion construction, selected host limits, host real equality, ambient completeness, bridge checking, or a Lean implementation.

Upgrade path: The theory axis is mature for the displayed route from $\mathsf{RegularCauchyFamily}^{\uparrow}$ through $\mathsf{RealDiagonalCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; the remaining verification work is auditClean and bridgeChecked certification.

Chapter 428

DyadicStepFunction NameCert Construction

The $\mathsf{DyadicStepFunction}^{\uparrow}$ packet is the finite Real and Completion approximation object sitting between dyadic spline data and regular Cauchy consumers. It uses Bishop-style finite partitions: every cell, endpoint transport, and carried value is a displayed BEDC row, and no measure completion, quotient representative, selected limit, or ambient equality oracle is part of the carrier.

The partition and endpoint rows are consumed from the dyadic mesh and interval carriers of chapter 358 and chapter 306. The stepwise regular-Cauchy handoff is the finite-cell consumer of the spline rows in chapter 382: a spline packet supplies the knot and carried-value surface, while the step-function packet records the common-refinement cells, endpoint transports, and ledger entries needed before $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ reads can consume the approximation.

Definition 428.1 (Dyadic step function carrier)**.** *A* $\mathsf{DyadicStepFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\Pi, I, V, R, M, E, L, C, P, N)$$

whose displayed coordinates are as follows:

1. Π *is a finite dyadic partition supplied by the* $\mathsf{DyadicMesh}^{\uparrow}$ *route;*
2. I *lists the* $\mathsf{DyadicInterval}^{\uparrow}$ *cells of* Π*, including displayed endpoints and adjacency witnesses;*
3. V *assigns to every cell of* I *one carried value, read either as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row or as the corresponding* $\mathsf{Real}^{\uparrow}$ *row;*
4. R *records the cellwise* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *classifier reads used to compare carried values;*
5. M *is the refinement map from any common dyadic refinement back to the cells of* Π*;*
6. E *is the endpoint-transport ledger for interval bounds, shared endpoints, and subdivision witnesses;*
7. L *is the finite* BHist *ledger of cell coverage, value reads, refinement reads, and consumer handoff rows;*
8. C *is the componentwise* hsame *and* Cont *route data used by the classifier;*
9. P *is the* Pkg *provenance naming the* $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicSpline}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *dependencies;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *provenance row.*

Two accepted carriers are classified together exactly when their partitions refine to a displayed common dyadic partition and, on every refined cell, the transported carried values are classified together by the recorded $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *classifier row. The comparison is finite and cellwise: it is mediated only by* M*,* E*,* L*, and the displayed* hsame *and* Cont *routes in* C*.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionTasteGate single carrier alignment`

Theorem 428.2 (Dyadic step function NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *obligation surface consists of carrier habitation, refinement-stable classification by common dyadic refinement, exact finite cell coverage, stepwise handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers, and ledger exactness for partition refinement, endpoint transport, carried values, finite coverage, and absence of hidden quotient data.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier namecert obligations`

Proof. Carrier habitation is the finite packaging operation of Definition 428.1. The packet displays the dyadic partition, its interval cells, the carried values, the classifier reads, the refinement map, endpoint transport, the finite history ledger, route data, package provenance, and the local certificate row.

The classifier obligation is constructive because two packets are compared by first displaying a common dyadic refinement and then reading finitely many transported cell values. Each read is a $\mathsf{DyadicRatCore}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ classifier read already carried by the packet; the refinement map only selects the parent cell named in M.

The consumer obligations are the finite coverage and handoff clauses proved below. Together they show that every point read by a regular Cauchy or real consumer is accounted for by a visible cell, visible endpoints, a visible carried value, and a visible BEDC route. □

Definition 428.3 (Dyadic step function finite window carrier)**.** *A* $\mathsf{DyadicStepFunction}^{\uparrow}$ *finite-window carrier is the carrier of Definition 428.1 read as the public window*

$$W = (I, M, E, V, R, L, C, P, N).$$

Here I *exposes the finite partition cells,* M *exposes endpoint-parent transport under common refinement,* E *exposes endpoint ledgers,* V *exposes carried values,* R *exposes classifier rows,* L *exposes the finite consumer ledger,* C *exposes* hsame *and* Cont *routes,* P *exposes the* Pkg *provenance, and* N *exposes the local* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *row. The partition coordinate* Π *remains the source of the cells in* I*, but every downstream consumer reads the packet through the displayed window* W*.*

Lean4 (definition) `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionWindowCarrier`

Theorem 428.4 (Dyadic step function finite-window obligations)**.** *For every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *finite-window carrier* $W = (I, M, E, V, R, L, C, P, N)$*, the public obligation surface consists of finite partition-cell coverage, endpoint transport, carried-value exposure, classifier-row exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal reading, componentwise* hsame *and* Cont *transport,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *naming. Every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer read through* W *must first land on a displayed common-refinement cell and then consume only the rows* $I, M, E, V, R, L, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier window obligations`

Proof. Start with the carrier and NameCert obligations of Definition 428.1 and Definition 428.2. They expose the partition, cells, endpoint transport, carried values, classifier reads, finite ledger, transport rows, provenance, and local name.

Next route a consumer read through common refinement. Apply Definition 428.9: the read lands on one displayed cell in I, then uses M for the parent-cell route and E for endpoint transport.

Now bind the value side. Apply Definition 428.10 to identify the endpoint route with the carried-value route through V and R, and apply Definition 428.8 to record the read in L.

Finally use Definition 428.7 and Definition 428.11. The regular-sequence handoff and the real seal read consume the same finite rows, while C, P, N supply the hsame, Cont, Pkg, and NameCert surface for replay and naming. □

Theorem 428.5 (Dyadic step function finite-window non-escape)**.** *Every accepted* DyadicStepFunction$^{\uparrow}$ *finite-window carrier exports its step-function,* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and completion-facing behavior only through*

$$I, M, E, V, R, L, C, P, N.$$

It exports no hidden host step function, quotient representative, selected limit, measure completion, open-cover compactness, or ambient equality oracle; every downstream read is routed through common-refinement cell coverage and ledger exactness before any handoff or seal row is consumed.

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier nonescape`

Proof. Project the finite-window carrier through Definition 428.3. The rows $I, M, E, V, R, L, C, P, N$ are the only public coordinates of the window.

Use Definition 428.4 to force each consumer read onto a displayed common-refinement cell. The endpoint route is read from E, the parent-cell route from M, the carried value from V, the classifier row from R, and the ledger entry from L.

The handoff rows do not add a second surface. By Definition 428.7 and Definition 428.11, RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers receive the same finite-cell data after the ledger row has been exposed.

Finally apply Definition 428.8. Because every permitted read is already accounted for by the displayed rows and their hsame, Cont, Pkg, and NameCert data, none of the excluded host, quotient, completion, limit, compactness, or equality objects has an escape route. □

Theorem 428.6 (Dyadic step function refinement stability)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *carriers. If their partitions refine to a displayed common dyadic partition and every refined cell transports to parent cells whose carried values are classified together by the recorded* DyadicRatCore$^{\uparrow}$ *or* Real$^{\uparrow}$ *classifier, then the* DyadicStepFunction$^{\uparrow}$ *classifier identifies S and T. The identification is stable under further* hsame *and* Cont *transport of the displayed partition, endpoint, value, and ledger rows.*

Proof. Project the two carriers through Definition 428.1. The common dyadic refinement supplies a finite list of refined cells, and the refinement maps of the two packets point each refined cell back to one parent cell in the original partitions.

On each refined cell, the hypothesis gives a displayed value comparison through the recorded DyadicRatCore$^{\uparrow}$ or Real$^{\uparrow}$ classifier. Endpoint transport is read from the two E ledgers, while the corresponding history rows are read from the two L ledgers.

Further transport by hsame and Cont changes only the displayed route data. Since the classifier is defined by the finite refinement list and the componentwise transported value reads, the same finite comparisons witness the transported identification. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionPacket refinement stability`

Theorem 428.7 (Dyadic step function RegSeqRat handoff)**.** *Every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *carrier exports a stepwise constant* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the corresponding* $\mathsf{Real}^{\uparrow}$ *read by using the carried cell values on the finite dyadic partition. A* $\mathsf{DyadicSpline}^{\uparrow}$ *or regular Cauchy consumer can inspect the exported stream only through the displayed partition cells, endpoint transport, cell values, classifier reads, and* BHist *ledger rows of the carrier.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier regseqrat handoff`

Proof. For each cell of Π, choose the carried value already displayed in V. The handoff stream is constant on that cell and changes only at endpoints recorded in E.

Regularity is witnessed cellwise. A request at a finite precision reads the unique displayed cell covering the request, the transported carried value on that cell, and the corresponding $\mathsf{DyadicRatCore}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ classifier row in R.

A $\mathsf{DyadicSpline}^{\uparrow}$ or regular Cauchy consumer therefore receives only a finite stepwise surface. The consumer may refine the partition, but refinement is routed through M and cannot introduce a new cell value, a selected limit, or an unrecorded equality proof. □

Theorem 428.8 (Dyadic step function ledger exactness)**.** *For every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *carrier, the ledger L accounts for exactly the finite cell coverage of Π, the endpoint transport recorded in E, the refinement-parent relation recorded in M, and the carried-value reads recorded in V and R. No exported* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicSpline}^{\uparrow}$, *or completion-facing read factors through a hidden quotient, a measure completion, an unlisted representative, or an ambient choice principle.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier ledger exactness`

Proof. Project the carrier and inspect the finite cell list I. Exact coverage means every displayed consumer read names one cell of I, and every cell of I has its endpoints, adjacency data, and carried value recorded in the packet.

Endpoint and refinement exactness are the two route checks. Endpoint movement is read from E, and refinement movement is read from M; both ledgers point back to displayed cells and therefore cannot add an unlisted interval or value.

The exported consumer surfaces are then repacked from the same finite rows. Because the only available reads are V, R, M, E, L, C, P, and N, every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicSpline}^{\uparrow}$, or completion-facing request remains inside the visible BEDC packet. □

Theorem 428.9 (Dyadic step function common refinement cell coverage)**.** *Let S and T be accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *carriers, and let a* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{DyadicSpline}^{\uparrow}$ *consumer compare their exported step-function packets. Every such comparison refines to one displayed dyadic partition cell. On that cell, the consumer read is routed through the visible cell list I, refinement map M, endpoint ledger E, carried values V, classifier rows R, history ledger L, and the componentwise* hsame *and* Cont *route data of the carriers.*

Thus the common-refinement comparison is a finite cell-coverage statement over the displayed BEDC packet. It has no coordinate for an ambient host function, a quotient representative, a selected limit, or an unrecorded equality oracle.

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier common refinement cell coverage`

Proof. Project S and T through Definition 428.1. The displayed partition rows and cell lists give the only cells available to a consumer comparison.

Apply Definition 428.6. The comparison first moves both carriers to a common dyadic refinement, and each refined cell is sent back to its parent cell through the recorded refinement maps in M.

Apply Definition 428.8. Endpoint movement is read from E, carried values from V, classifier observations from R, and the finite history of the read from L. The remaining route data are exactly the displayed hsame and Cont entries in C, so every consumer read lands on one visible cell and no hidden representative can be inspected. □

Theorem 428.10 (Dyadic step function endpoint value transport determinacy)**.** *For accepted* DyadicStepFunction$^{\uparrow}$ *carriers compared after common dyadic refinement, the transported endpoint rows in E and the transported carried-value rows in V determine the same cellwise read. The determination is made through I, M, R, L,* hsame*, and* Cont *only, and does not depend on a hidden representative, a host step function, or an external choice of values.*

Proof. Project the accepted carriers through Definition 428.1. A refined cell has one displayed parent cell, named by M, and the endpoints of that parent cell are the rows transported by E.

Use Definition 428.6 to compare the transported parent cells. The carried value read is then the row selected from V and checked by the corresponding classifier row in R.

Finally apply Definition 428.8. The ledger L records both the endpoint transport and the carried-value read, so the two routes determine the same cellwise observation inside the displayed packet. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier endpoint value transport determinacy`

Theorem 428.11 (Dyadic step function RealUp seal window exactness)**.** *Every* Real$^{\uparrow}$ *seal read exported by an accepted* DyadicStepFunction$^{\uparrow}$ *carrier factors through finite common-refinement cell coverage, endpoint transport, the carried-value classifier, and the displayed ledger before the seal is consumed. Equivalently, a* Real$^{\uparrow}$ *consumer reaches the seal only after selecting a displayed cell in I, transporting its endpoints through E, transporting its parent by M, reading its carried value through V and R, and recording the read in L with the stored* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier real seal window exactness`

Proof. Start with the seal-facing handoff of Definition 428.7. The RegSeqRat$^{\uparrow}$ export and the corresponding Real$^{\uparrow}$ read are stepwise constant on the displayed finite partition.

Apply Definition 428.9. Any transported consumer comparison first reduces to a displayed common-refinement cell, so the seal read cannot bypass the finite cell list.

Apply Definition 428.10 to identify the endpoint route with the carried-value route on that cell. Then apply Definition 428.8 to place the resulting read in L before the Real$^{\uparrow}$ seal is consumed. This exhausts the available carrier coordinates and leaves no path to quotient data, host functions, or selected limits. □

Theorem 428.12 (Dyadic step function cellwise sum closure)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets. Suppose they admit a displayed common dyadic refinement whose refined cells are finite* BHist *cells, and suppose that on every refined cell the transported carried-value row of S, the transported carried-value row of T, and the displayed* DyadicRatCore$^{\uparrow}$ *or* Real$^{\uparrow}$ *sum row are recorded in*

the carrier ledger. Then the pointwise cellwise sum repacks as an accepted DyadicStepFunction$^{\uparrow}$ *packet.*

The repacked packet uses only the displayed refined cell list, endpoint transport, carried-value rows, classifier rows, finite ledger rows, and the componentwise hsame *and* Cont *route data. It does not introduce a host step function, a quotient representative, a selected limit, or an equality oracle outside the packet.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier cellwise sum closure`

Proof. Project S and T through Definition 428.1. The carrier exposes the finite cell list, endpoint rows, carried-value rows, classifier rows, ledger rows, and transport data from which the sum packet must be built.

Apply Definition 428.9. Every consumer-visible comparison of the two packets first lands on one displayed common-refinement cell, and the finite BHist list gives the induction spine for checking all cells.

On each refined cell, use Definition 428.10. The endpoint route and the carried-value route determine the same transported read, so the two summand values are the visible rows named by the refined packet data.

Finally apply Definition 428.8. The assumed DyadicRatCore$^{\uparrow}$ or Real$^{\uparrow}$ sum row is already recorded in the carrier ledger for that refined cell, and the stored hsame and Cont routes transport it componentwise. Repacking those finitely many rows gives the accepted DyadicStepFunction$^{\uparrow}$ sum packet with no access to host functions, quotient representatives, selected limits, or hidden equalities. □

Theorem 428.13 (Dyadic step function refinement intersection closure)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets with displayed common dyadic refinements. Suppose the two refined packets expose finite cell lists in I, refinement maps in M, endpoint transport in E, carried-value rows in V, classifier rows in R, ledger rows in L, continuation rows in C, provenance rows in P, and names in N. Then the finite common cell intersection of the displayed refinements repacks as an accepted* DyadicStepFunction$^{\uparrow}$ *packet.*

The repacked packet is built only from I, M, E, V, R, L, C, P, and N. It carries the inherited hsame *and* Cont *routes for each intersected cell and does not call a host step function, quotient representative, selected limit, or ambient equality oracle.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier refinement intersection closure`

Proof. Project both packets through Definition 428.1. The two displayed refinements provide finite cell lists, and their common intersections are therefore enumerated by finite BHist data.

Apply Definition 428.9. Every consumer-visible comparison first lands on one displayed common-refinement cell, so the intersection packet has a visible cell for each comparison that can be made from S and T.

Use Definition 428.6 to transport each intersected cell back to its parent cells in the two packets. Endpoint transport through E and parent selection through M keep the refined cell attached to the displayed carrier rows.

Finally apply Definition 428.10 and Definition 428.8. The carried values and classifier reads are already in V, R, and L, while C, P, and N replay the same local package. Repacking those finitely many rows gives an accepted intersection packet without adding any external analytic object. □

Theorem 428.14 (Dyadic step function common-refinement normal form)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets with displayed common dyadic refinements. Every* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or* DyadicSpline$^{\uparrow}$ *consumer comparison between S and T has a normal form consisting of one displayed refined cell, its parent-cell maps through M, endpoint transports through E, carried-value reads through V and R, a ledger entry in L, and the stored* hsame, Cont, Pkg, *and* NameCert *rows in C, P, and N.*

If two consumer routes name the same displayed refined cell, then their visible normal-form rows are the same M, E, V, R, L, C, P, N rows. The normal form does not inspect a host step function, a quotient representative, a selected limit, or an equality oracle outside the finite common-refinement packet.

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier common refinement normal form`

Proof. First isolate the finite cell. Apply Definition 428.9: every consumer comparison lands on one displayed common-refinement cell before any RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, or DyadicSpline$^{\uparrow}$ handoff can be read.

Next attach that cell to the packet rows. By Definition 428.6, the refined cell carries its parent-cell maps through M, and the endpoint transports remain the displayed rows in E.

Now bind the value and classifier reads. Use Definition 428.10 to identify the transported carried-value reads with the rows in V and R, then use Definition 428.8 to pin the consumer-facing comparison to the ledger entry in L.

Finally apply Definition 428.13. The intersection packet stores the same C, P, N local package rows for the refined cell, so two routes with the same displayed cell replay the same visible normal-form rows before reaching any downstream consumer. □

Theorem 428.15 (Dyadic step function cellwise product closure)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets. Suppose they admit a displayed common dyadic refinement whose refined cells are finite* BHist *cells, and suppose that on every refined cell the transported carried-value row of S, the transported carried-value row of T, and the displayed* DyadicRatCore$^{\uparrow}$ *or* Real$^{\uparrow}$ *product row are recorded in the carrier ledger. Then the pointwise cellwise product repacks as an accepted* DyadicStepFunction$^{\uparrow}$ *packet.*

The repacked packet uses only the displayed refined cell list, endpoint transport, carried-value rows, classifier rows, finite ledger rows, and the componentwise hsame *and* Cont *route data. It does not introduce a host step function, a quotient representative, a selected limit, or an equality oracle outside the packet.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier product closure`

Proof. Project S and T through Definition 428.1. The carrier exposes the same finite partition, endpoint, value, classifier, ledger, transport, and continuation rows used by the sum packet.

Apply Definition 428.13. The product is read on the displayed common cell intersection, so both factors have visible transported carried-value rows on the cell where multiplication is performed.

On each refined cell, use Definition 428.10. The endpoint route and carried-value route determine the same visible read for each factor before the product row is consumed.

Finally apply Definition 428.8 together with Definition 428.11. The assumed DyadicRatCore$^{\uparrow}$ or Real$^{\uparrow}$ product row is already a ledger entry for that cell, and the seal-facing read cannot escape the displayed finite window. Repacking these rows gives the accepted product packet without invoking host functions, quotient representatives, selected limits, or hidden equalities. □

Theorem 428.16 (Dyadic step function stepwise order window)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets. Any pointwise order comparison between S and T is witnessed after a displayed common dyadic refinement: on each refined cell, the transported carried-value rows and the displayed* DyadicRatCore$^{\uparrow}$ *or* Real$^{\uparrow}$ *order row determine the comparison visible to consumers. The order window is finite-cellwise and is recorded by the same I, M, E, V, R, L, C, P, and N rows as the packets themselves.*

Consequently the comparison cannot use a host function, a quotient representative, a selected limit, trichotomy on hidden real representatives, or an equality oracle outside the displayed carrier.

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier stepwise order window`

Proof. Begin with Definition 428.13. It gives a finite common-refinement packet whose cells cover all comparisons available from the two accepted packets.

For a fixed refined cell, apply Definition 428.6. The cell is transported back to displayed parent cells in S and T, and the endpoint rows remain attached to the refinement maps.

Use Definition 428.10 to identify the endpoint route with the carried-value route for both packets. The order row being compared is therefore the displayed DyadicRatCore$^{\uparrow}$ or Real$^{\uparrow}$ row named in the ledger, not a host-side comparison.

Finally apply Definition 428.8 and Definition 428.11. The ledger records each finite-cell order read before any seal-facing consumer can use it, so the comparison remains inside the displayed finite window. □

Theorem 428.17 (Dyadic step function scoped window package)**.** *Let S and T be accepted* DyadicStepFunction$^{\uparrow}$ *packets. Their scoped comparison packet is the finite common-refinement window whose cells cover all consumer-visible reads, whose endpoint rows are transported through E, and whose carried-value rows are read through V and R. The packet is entirely contained in the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the two carriers.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier scoped window package`

Proof. Start with Definition 428.9. It places every comparison on one displayed cell of a common finite dyadic refinement.

Apply Definition 428.6. The refinement maps send each refined cell back to parent cells, while endpoint data remain attached through the displayed endpoint-transport ledgers.

Use Definition 428.8 to close the packet. All cell coverage, endpoint transport, carried-value reads, and classifier observations are recorded in the finite ledger and routed only by hsame and Cont data. □

Theorem 428.18 (Dyadic step function Real-RegSeqRat scope lock)**.** *Every* RegSeqRat$^{\uparrow}$ *handoff and every* Real$^{\uparrow}$ *seal read exported by an accepted* DyadicStepFunction$^{\uparrow}$ *packet is locked to the same finite cell window. The seal receives only the displayed partition cells, endpoint transport, carried values, classifier rows, ledger entries, and the recorded* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier real regseqrat scope lock`

Proof. Apply Definition 428.7. The regular sequence surface is built from the carried cell values on the displayed finite partition.

Next apply Definition 428.11. The Real$^{\uparrow}$ seal is reached only after common-refinement coverage, endpoint transport, carried-value classification, and ledger recording have occurred.

The two handoffs therefore share the same finite scope. Since Definition 428.8 excludes hidden quotient data, host functions, selected limits, and ambient choices, no seal-facing row can escape the displayed window. □

Theorem 428.19 (Dyadic step function refinement ledger scope)**.** *The refinement ledger of an accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet is scoped by the finite rows* $\Pi, I, V, R, M, E, L, C, P, N$*. Common-refinement cell coverage, endpoint transport, value transport,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal exactness are all read from these rows, and no refinement consumer can introduce a host step function or unlisted representative.*

Proof. Project the carrier of Definition 428.1. The refinement map M, endpoint ledger E, finite history ledger L, and route row C are explicit coordinates of the packet.

Apply Definition 428.10. On each refined cell, endpoint transport and carried-value transport determine the same visible cellwise read.

Finally combine Definition 428.17 with Definition 428.18. The common refinement, regular handoff, and real seal are all ledger reads over $\Pi, I, V, R, M, E, L, C, P, N$, so the refinement scope is exact. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier refinement ledger scope`

Theorem 428.20 (Dyadic step function scoped obligation package)**.** *Every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet assembles a scoped finite obligation package consisting of carrier habitation, refinement stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, ledger exactness, common-refinement coverage, endpoint-value transport,* $\mathsf{Real}^{\uparrow}$ *seal-window exactness, cellwise sum closure, cellwise product closure, and the stepwise order window. The package is over the displayed* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicSpline}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *rows.*

The scoped package exports no measure completion, quotient representative, selected limit, host step function, or ambient equality oracle.

Proof. First project the carrier and classifier rows. The finite partition, dyadic interval cells, carried values, common-refinement maps, endpoint routes, classifier rows, ledger entries, continuation rows, package provenance, and local naming row are the only visible coordinates.

Apply the refinement stability, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, ledger exactness, common-refinement coverage, endpoint-value transport, and $\mathsf{Real}^{\uparrow}$ seal-window exactness theorems. These results bind every cellwise read to the displayed mesh, interval, rational, spline, regular-sequence, real, history, transport, continuation, package, and naming rows.

Finally use the cellwise sum, cellwise product, and stepwise order window theorems. They show that the visible finite-cell operations and comparisons repack inside the same scoped rows, so no measure completion, quotient representative, selected limit, host step function, or ambient equality oracle is exported. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier scoped obligation package`

Theorem 428.21 (DyadicStepFunction scope export)**.** *The scoped obligation package of Definition 428.20 exports exactly the downstream surface consisting of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *partition and value rows, the* $\mathsf{DyadicMesh}^{\uparrow}$ *and* $\mathsf{DyadicSpline}^{\uparrow}$ *refinement rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal-window exactness rows. Every downstream step-function consumer must read through those displayed rows together with the carrier's* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *coordinates.*

No exported scope includes open-cover compactness, a host function space, selected limits, quotient representatives, measure completion, or an ambient equality oracle.

Proof. First read the local scoped package. By Definition 428.20, every admitted packet is already restricted to the displayed mesh, interval, value, refinement, endpoint, ledger, continuation, provenance, and local naming rows.

Next isolate the rows visible to downstream consumers. The carrier and refinement ledger scope of Definition 428.1 and Definition 428.19 provide the $\mathsf{DyadicRatCore}^{\uparrow}$ partition and value rows together with the $\mathsf{DyadicMesh}^{\uparrow}$ and $\mathsf{DyadicSpline}^{\uparrow}$ refinement rows.

The analytic handoff is exactly the one already bounded locally. Apply Definition 428.18 for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and Definition 428.11 for the $\mathsf{Real}^{\uparrow}$ seal-window exactness.

Finally combine the non-escape clauses of Definition 428.8 and Definition 428.17. They exclude open-cover compactness, host function spaces, selected limits, quotient representatives, measure completion, and ambient equality oracles from the exported surface. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier scope export`

Theorem 428.22 (Dyadic step function refinement boundary coverage)**.** *Every accepted common-refinement consumer read of a* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet is covered by the displayed finite cells, endpoint ledgers, carried-value rows, classifier rows, and* BHist *ledger entries of the carrier. Equivalently, a consumer that reads through a common dyadic refinement must first land on one cell of the visible refinement window and then consume only the endpoint transport in* E*, the carried values in* V*, the classifier rows in* R*, the finite ledger* L*, and the stored* hsame *and* Cont *routes.*

The refinement boundary exports no host step function, measure completion, quotient representative, selected limit, or ambient equality oracle.

Proof. First apply Definition 428.9. It places every accepted consumer comparison on a displayed common-refinement cell and identifies the available read coordinates as $I, M, E, V, R, L, C, P, N$.

Next use Definition 428.10. On the selected cell, endpoint transport and carried-value transport determine the same visible read, so the consumer cannot split the endpoint route from the value route.

Finally apply Definition 428.18 and Definition 428.19. These keep $\mathsf{RegSeqRat}^{\uparrow}$ handoffs and $\mathsf{Real}^{\uparrow}$ seal reads inside the same finite ledger before any downstream consumer can inspect them. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier refinement boundary coverage`

Theorem 428.23 (Dyadic step function scope transport package)**.** *The carrier surface, refinement stability, endpoint-value determinacy,* $\mathsf{Real}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *scope lock, and refinement ledger scope assemble into one kernel-scoped* $\mathsf{DyadicStepFunction}^{\uparrow}$ *package over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package binds every accepted partition cell, common-refinement read, endpoint transport, carried-value row, classifier row, regular-sequence handoff, real seal read, and ledger entry to the displayed finite carrier rows.*

This scope transport package is the single packet consumed by downstream step-function, regular Cauchy, and real-facing readers; it has no coordinate for open-cover compactness, host function spaces, quotient representatives, selected limits, measure completion, or ambient equality.

Proof. Start with the carrier and classifier data of Definition 428.1. The packet exposes the finite partition, interval cells, carried values, refinement map, endpoint ledger, finite history ledger, continuation routes, provenance, and local naming row.

Use Definition 428.6 and Definition 428.10. They transport common-refinement cells and identify endpoint and carried-value reads by the stored hsame and Cont routes.

Next apply Definition 428.18 and Definition 428.19. These bind the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, and refinement ledger to the same finite rows.

Finally invoke Definition 428.20. It packages the carrier, stability, handoff, ledger, coverage, endpoint-value, and seal-window rows into the kernel-scoped obligation surface, so the stated scope transport package has no hidden analytic coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier scope transport package`

Theorem 428.24 (Dyadic step function real-seal public boundary)**.** *Every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet exports its* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal rows as one public boundary over the finite coordinates* $I, M, E, V, R, L, C, P, N$*. The boundary is assembled from the displayed partition cells, common-refinement maps, endpoint transport, carried values, classifier rows, finite ledger entries, continuation routes, package provenance, and local naming row.*

The public boundary exports no open-cover compactness, host step function, selected limit, quotient representative, measure completion, or ambient equality oracle.

Proof. First read the finite scope transport package of Definition 428.23. It already binds every partition cell, common-refinement read, endpoint route, value row, classifier row, ledger entry, continuation route, package row, and local naming row to the displayed carrier surface.

Next isolate the analytic handoff rows. Apply Definition 428.18 to bind the $\mathsf{RegSeqRat}^{\uparrow}$ handoff to the finite step-function rows, and apply Definition 428.11 to bind the $\mathsf{Real}^{\uparrow}$ seal-window read to the same visible cellwise data.

Finally repack those two rows through Definition 428.20. The resulting public boundary has exactly the coordinates $I, M, E, V, R, L, C, P, N$, so none of the excluded analytic or quotient objects has an exported coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier real seal public boundary`

Theorem 428.25 (Dyadic step function common-refinement public route)**.** *Every public consumer route for common-refinement data in an accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet factors through the displayed coordinates* $I, M, E, V, R, L, C, P, N$*. A route may read interval cells, mesh refinements, endpoint transport, carried values, classifier rows, finite ledger entries, continuation routes, package provenance, and the local* $\mathsf{NameCert}$ *row, but it cannot bypass those rows through a host step function, quotient representative, selected limit, measure completion, or ambient equality oracle.*

Proof. Start with the refinement boundary coverage of Definition 428.22. It places every accepted common-refinement consumer on a visible cell and identifies the available read coordinates as $I, M, E, V, R, L, C, P, N$.

The endpoint and carried-value routes then agree by Definition 428.10. This prevents a public consumer from splitting the endpoint route away from the value row after entering the common refinement.

The public analytic boundary is supplied by Definition 428.24. It attaches the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal rows to the same coordinates already exposed by the common-refinement coverage.

Therefore every public consumer route factors through $I, M, E, V, R, L, C, P, N$. The excluded host and quotient objects have no row in that factorization, so they cannot serve as bypass routes. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier common refinement public route`

Theorem 428.26 (Dyadic step function common-refinement Real correspondence)**.** *For accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packets compared through a displayed common dyadic refinement, the public consumer route through finite cells, endpoint transport, carried values, classifier rows, and finite ledger entries is the same route consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the* $\mathsf{Real}^{\uparrow}$ *seal-window read. Equivalently, after the common refinement has exposed* $I, M, E, V, R, L, C, P, N$*, a downstream consumer may pass from the cellwise step-function surface to the regular-Cauchy handoff and to the real seal boundary without introducing any further comparison coordinate.*

This correspondence exports no host step function, quotient representative, measure completion, selected limit, or ambient equality oracle.

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.CommonRefinementRealRegseq.DyadicStepFunctionCarrier common refinement real regseq correspondence`

Proof. First expose the common-refinement cells. Apply Definition 428.9 to place every accepted comparison on displayed dyadic cells and to keep the visible rows inside $I, M, E, V, R, L, C, P, N$.

Next transport the cell data. Apply Definition 428.10 so endpoint reads and carried-value reads follow the same stored hsame and Cont routes before any consumer leaves the finite refinement ledger.

The analytic handoff uses no separate route. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is bounded by Definition 428.18, and the public seal boundary is bounded by Definition 428.24. Both the regular-Cauchy row and the $\mathsf{Real}^{\uparrow}$ seal-window read therefore consume the same cell, endpoint, value, classifier, and ledger rows already exposed by the common refinement.

Thus the public common-refinement route and the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route are two readings of the same finite packet surface. The excluded host, quotient, completion, limit, and ambient equality objects have no coordinate in that packet, so they cannot distinguish the two routes. □

Theorem 428.27 (Dyadic step function public certificate)**.** *Every accepted* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet exports a public* $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ *certificate whose visible surface is exactly the finite dyadic partition, interval-cell row, mesh and common-refinement row, endpoint transport, carried values, classifier rows, finite ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$*. The certificate is the public export of the carrier rows* $I, M, E, V, R, L, C, P, N$*.*

This public certificate claims no host step functions, quotient representatives, measure completion, selected limits, or ambient equality.

Proof. First expose the finite common-refinement surface. Apply Definition 428.9 to place every accepted packet read on displayed dyadic cells, and use Definition 428.25 to keep the visible route inside $I, M, E, V, R, L, C, P, N$.

Next bind endpoint and value reads. Apply Definition 428.10; endpoint transport and carried values are therefore read through the same stored hsame and Cont routes before a public consumer can leave the finite ledger.

The real-facing boundary is already local to those rows. Use Definition 428.18 for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and Definition 428.24 for the $\mathsf{Real}^{\uparrow}$ seal boundary.

Finally apply Definition 428.26. It identifies the public common-refinement route with the regular-Cauchy and real-seal route, so repacking $I, M, E, V, R, L, C, P, N$ gives exactly the stated public $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$ certificate and leaves the refused host, quotient, completion, limit, and ambient-equality objects without coordinates. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionCarrier public certificate`

Theorem 428.28 (Dyadic step function standard bridge)**.** *The public* $\mathsf{DyadicStepFunction}^{\uparrow}$ *certificate of Definition 428.27 is transported through the displayed dyadic mesh, dyadic interval,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The bridge imports no host step function, ambient completeness object, selected limit, quotient representative, measure completion, or ambient equality oracle.*

Proof. Start from the public certificate surface. By Definition 428.27, every visible read of the packet is already exposed through the finite partition, interval-cell row, mesh and common-refinement row, endpoint transport, carried values, classifier rows, finite ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal row, transport, replay, provenance, and local naming.

The bridge therefore has no extra analytic coordinate to import. Use Definition 428.25 for the dyadic mesh and interval route, Definition 428.18 for the regular-Cauchy handoff, and Definition 428.24 for the real seal boundary.

Finally, Definition 428.26 identifies the common-refinement route with the regular-Cauchy and real-seal route. Thus the standard bridge is exactly the displayed packet transport through $I, M, E, V, R, L, C, P, N$, with the refused host and completion objects remaining outside the public certificate. □

Lean4 ✓ `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$.

Constructive story: A dyadic step-function packet is generated by a finite dyadic partition, displayed interval cells, carried values, common-refinement maps, endpoint transport, finite consumer ledgers, and visible transport and provenance rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 428.28 transports the public certificate surface over the displayed finite partition, dyadic interval, carried value, common-refinement, endpoint-transport, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows of the finite step-function packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicStepFunction}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.DyadicStepFunctionUp.DyadicStepFunctionUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close measure completion, quotient representatives, selected limits, host step functions, or ambient equality oracles.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 429

A Concrete Naming Certificate for $\mathsf{RegularCauchyDifference}^{\uparrow}$

The $\mathsf{RegularCauchyDifference}^{\uparrow}$ interface names the finite pointwise-subtraction surface for two regular rational Cauchy names. Following Bishop and Bridges, *Constructive Analysis*, it records the displayed difference stream that a $\mathsf{NullSequence}^{\uparrow}$ consumer may later inspect, without forming a quotient of streams, selecting a host limit, or appealing to ambient real completeness.

The two source rows are read through chapter 371, and the zero-shrinkage consumer row is read through chapter 380. The $\mathsf{RegularCauchyDifference}^{\uparrow}$ packet consumes those sibling surfaces to form a pointwise subtraction packet; it introduces no quotient stream equality, host real equality, selected limits, or ambient completeness.

Definition 429.1 (RegularCauchyDifference carrier)**.** *A* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, S, T, Z, H, C, P, N)$$

with the following displayed coordinates:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose scheduled rational observations are compared pointwise;*
2. *S is the pointwise dyadic subtraction schedule, aligning the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows of X and Y and forming the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row for $X - Y$ at each requested precision;*
3. *T is the transported* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the source windows, endpoint rows, requested dyadic bounds, and output tolerance assigned to the difference row;*
4. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *consumer row when the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference stream shrinks to zero;*
5. *H stores componentwise* hsame *transports for the two source rows, the aligned windows, the dyadic subtraction rows, the tolerance ledger, and the optional null-sequence consumer row;*
6. *C stores the* Cont *routes for reading the two source windows, forming the dyadic subtraction row, transporting the tolerance ledger, and exposing Z when zero-shrinkage is supplied;*
7. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDifference}^{\uparrow}}$ *row.*

The carrier is a finite difference certificate for regular Cauchy names. It stores no quotient stream, no selected limit, no host real equality, and no ambient completeness object.

Theorem 429.2 (RegularCauchyDifference NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDifference}^{\uparrow}}$ *surface for the carrier of Definition 429.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is the displayed packet D with two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, a pointwise subtraction schedule, a transported tolerance ledger, an optional* $\mathsf{NullSequence}^{\uparrow}$ *consumer row, transport rows, continuation rows, provenance, and local naming data;*
2. *classifier transport: classification transports the two source rows, aligned windows, dyadic subtraction rows, tolerance ledger, and null-sequence consumer componentwise through the stored* hsame *and* Cont *rows;*
3. *stability: replacing either source by a classified regular Cauchy name preserves the finite subtraction schedule and recomputes the same kind of displayed difference row;*
4. *ledger exactness: every output tolerance read by the difference row is the displayed tolerance in T, obtained from the two source windows and the dyadic subtraction schedule S;*
5. *downstream handoff: every* $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the difference first reads $X, Y, S, T, Z, H, C, P, N$.*

Proof. Carrier admission is projection from the tuple in Definition 429.1. The two source names, subtraction schedule, tolerance ledger, consumer row, transport routes, continuation routes, package row, and local naming row are all explicit coordinates of D.

Classifier transport is componentwise. The two $\mathsf{RegSeqRat}^{\uparrow}$ source rows transport their scheduled rational observations, the $\mathsf{StreamName}^{\uparrow}$ windows transport the finite reads, and the $\mathsf{DyadicRatCore}^{\uparrow}$ rows transport the endpoint subtraction calculation at the requested precision.

The tolerance ledger is finite. It records the source windows and the output tolerance before any consumer sees the difference row. The downstream handoff is therefore the displayed difference surface together with Z when zero-shrinkage is supplied, and no omitted coordinate can add quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 429.3 (RegularCauchyDifference regularity closure)**.** *Given accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows X and Y in a* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *packet, the schedule S and ledger T expose a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row. Each requested precision reads aligned* $\mathsf{StreamName}^{\uparrow}$ *windows for X and Y, forms the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint subtraction row, and records the output tolerance in the same finite ledger.*

Proof. Read the two regular source names through their aligned stream windows. The schedule S names the common finite observation at which the two endpoint rows are inspected.

Apply the displayed dyadic subtraction operation to the two endpoint rows. The resulting point row is still inside the packet, and the source regularity rows provide the tolerance entries that T records for the output request.

Repack these finite observations as the displayed $\mathsf{RegSeqRat}^{\uparrow}$ difference row. Since the construction uses only X, Y, S, T and the visible transport and continuation rows, it does not require a completed real, a selected limit, or a quotient stream. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceUp.RegularCauchyDifferenceCarrier regularity closure`

Theorem 429.4 (RegularCauchyDifference classifier transport)**.** *Let $D = (X, Y, S, T, Z, H, C, P, N)$ and $D' = (X', Y', S', T', Z', H', C', P', N')$ be accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *carriers classified by the displayed* hsame *and* Cont *routes. Transporting the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows transports the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subtraction schedule, the tolerance ledger, and the optional* $\mathsf{NullSequence}^{\uparrow}$ *consumer row componentwise through the same carrier surface.*

Proof. Project both carriers through Definition 429.1. The classifier has access only to the two source rows, the schedule, the tolerance ledger, the optional null-sequence row, and the displayed transport, continuation, package, and naming rows.

Apply the classifier clause of Definition 429.2. The source rows transport first; the aligned windows and dyadic endpoint subtraction rows are then read through the same schedule route.

The regularity closure of Definition 429.3 repacks the transported finite observations as the displayed difference row. Hence classifier transport stays inside the same finite carrier surface and does not use quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 429.5 (RegularCauchyDifference tolerance ledger exactness)**.** *For every accepted* RegularCauchyDifferenc *carrier $D = (X, Y, S, T, Z, H, C, P, N)$, every tolerance consumed by the displayed pointwise difference row is exactly the entry of the transported ledger T obtained from the two aligned source windows and the subtraction schedule S. No consumer may replace T by an external modulus, host real equality proof, selected limit, or ambient completeness row.*

Proof. Start with the output tolerance request in Definition 429.3. The request first reads aligned $\mathsf{StreamName}^{\uparrow}$ windows for X and Y, then forms the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint subtraction row named by S.

The ledger clause of Definition 429.2 records the output tolerance before any downstream consumer reads the difference. Thus the tolerance is the visible row in T and is transported by the displayed hsame and Cont routes.

Finally use the carrier inventory. Since no coordinate of D is an external modulus, host real equality proof, selected limit, or ambient completeness theorem, there is no other tolerance source available to a consumer. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceUp.RegularCauchyDifferenceCarrier tolerance ledger exactness`

Theorem 429.6 (RegularCauchyDifference RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *carrier $D = (X, Y, S, T, Z, H, C, P, N)$ receives the pointwise subtraction surface through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row, the tolerance ledger exactness row for T, and the* $\mathsf{NullSequence}^{\uparrow}$ *consumer row Z when zero-shrinkage is supplied. The handoff exposes no quotient stream, selected host limit, host real equality, or ambient completeness object.*

Proof. First obtain the displayed difference row from Definition 429.3. Its observations are the dyadic subtraction rows read from X, Y, and S.

Next apply Definition 429.5. This fixes the tolerance read as the ledger T, so the Real-facing route cannot substitute an external modulus or selected limit.

If the consumer asks for a zero-tail route, apply Definition 429.7. Otherwise the non-escape boundary of Definition 429.8 still forces the read through the displayed carrier rows, so every $\mathsf{Real}^{\uparrow}$ seal handoff is finite and visible. □

Theorem 429.7 (RegularCauchyDifference NullSequence consumption)**.** *If the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row of Definition 429.3 shrinks to zero, then the row Z is exactly the* $\mathsf{NullSequence}^{\uparrow}$ *consumer surface for that difference. Every tolerance request consumed by Z factors through the*

transported ledger T, the subtraction schedule S, the source rows X, Y, and the displayed hsame, Cont, Pkg, *and* NameCert *provenance of the carrier.*

Proof. Start with the displayed difference row supplied by Definition 429.3. It is already a regular rational stream read through the two source rows, the finite subtraction schedule, and the tolerance ledger.

When zero-shrinkage is supplied, project the coordinate Z. The NullSequence$^{\uparrow}$ consumer reads precisely the tolerance requests in T and the finite difference observations formed by S before exposing any real-seal equality evidence.

The provenance rows do not add analytic data. They route the same source rows, windows, endpoint subtraction rows, and tolerances through hsame, Cont, Pkg, and the local NameCert row. Thus the null-sequence consumption is exactly the finite difference packet and not an ambient equality of host reals. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceUp.RegularCauchyDifferenceCarrier nullsequence consumption`

Theorem 429.8 (RegularCauchyDifference non-escape boundary)**.** *No accepted* RegularCauchyDifference$^{\uparrow}$ *packet exports quotient streams, host real equality, selected limits, or ambient completeness. Every public read of a pointwise difference, and every zero-shrinkage read by* NullSequence$^{\uparrow}$*, is exhausted by the finite* BHist *rows named in Definition 429.1 and by the handoff rows of Definition 429.3 and Definition 429.7.*

Proof. Inspect the carrier inventory. The only analytic coordinates are two regular rational source rows, the finite subtraction schedule, the transported tolerance ledger, and the null-sequence consumer row when a zero-tail certificate is present.

The regularity closure theorem shows that regular consumers receive only the scheduled pointwise difference row and its tolerance ledger. The null-sequence consumption theorem places zero-tail consumers downstream of that same displayed difference row.

These routes exhaust the packet. Since no listed coordinate is a quotient stream, host real equality, selected limit, or ambient completeness theorem, no consumer can read such an object from the RegularCauchyDifference$^{\uparrow}$ certificate. □

Theorem 429.9 (RegularCauchyDifference difference carrier closure)**.** *Let $D = (X, Y, S, T, Z, H, C, P, N)$ be an accepted* RegularCauchyDifference$^{\uparrow}$ *carrier whose source rows X and Y are accepted* RegSeqRat$^{\uparrow}$ *rows. The displayed pointwise difference of X and Y is again read as a row inside the same* RegularCauchyDifference$^{\uparrow}$ *carrier: its windows are the aligned* StreamName$^{\uparrow}$ *windows named by S, its rational observations are the* DyadicRatCore$^{\uparrow}$ *subtraction rows named by S, and its tolerances are exactly the entries of T.*

Proof. Project the accepted carrier D. The two RegSeqRat$^{\uparrow}$ source rows, the alignment schedule, and the tolerance ledger are visible coordinates before any consumer reads the difference.

Apply Definition 429.3. It forms the displayed regular rational difference row by reading the aligned windows of X and Y, performing the dyadic subtraction recorded by S, and recording the output request in T.

The repacked row uses only the carrier coordinates already listed in Definition 429.1. Thus the difference remains inside the displayed RegularCauchyDifference$^{\uparrow}$ packet and does not pass through a quotient stream, selected limit, host real equality, or ambient completeness object. □

Theorem 429.10 (RegularCauchyDifference NullSequence classifier stability)**.** *For accepted* RegularCauchyDifferen *carriers classified by the displayed* hsame *and* Cont *routes,* NullSequence$^{\uparrow}$ *consumption is stable: if one displayed difference row is consumed by its zero-shrinkage row Z, then the classified carrier consumes the transported displayed difference through the corresponding* NullSequence$^{\uparrow}$ *row, with the same source rows, subtraction schedule, tolerance ledger, package provenance, and local naming route transported componentwise.*

Proof. Start with the zero-shrinkage read in Definition 429.7. The consumer row Z reads the difference only through the source rows X, Y, the subtraction schedule S, and the tolerance ledger T.

Transport the classified carrier componentwise along the displayed hsame and Cont rows. The source names, aligned windows, dyadic subtraction rows, tolerance ledger, package provenance, and local NameCert row are all moved by the same routes used in Definition 429.2.

The transported Z therefore consumes the transported displayed difference and no additional analytic witness. Stability is exactly the classifier action on the finite carrier rows, followed by the same NullSequence$^{\uparrow}$ consumer route. □

Theorem 429.11 (RegularCauchyDifference RealUp seal route)**.** *Every* Real$^{\uparrow}$ *seal read that uses a* RegularCauchyDifference$^{\uparrow}$ *packet factors through the displayed* DyadicRatCore$^{\uparrow}$ *subtraction rows, the* NullSequence$^{\uparrow}$ *consumer row when zero-shrinkage is supplied, the* RegSeqRat$^{\uparrow}$ *source and difference rows, the* StreamName$^{\uparrow}$ *windows, and the carrier's* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No* Real$^{\uparrow}$ *consumer can bypass these rows to obtain a quotient stream, selected limit, host real equality, or ambient completeness principle.*

Proof. First read the displayed difference row. By Definition 429.9, the route begins with the two RegSeqRat$^{\uparrow}$ source rows, their aligned StreamName$^{\uparrow}$ windows, the DyadicRatCore$^{\uparrow}$ subtraction rows, and the tolerance ledger.

If the seal route is a zero-tail route, apply Definition 429.7. The NullSequence$^{\uparrow}$ row is downstream of the displayed difference row and consumes only the tolerances and observations stored in the packet.

Finally apply the non-escape boundary of Definition 429.8. The remaining rows are the visible hsame transports, Cont routes, Pkg provenance, and local NameCert row, so every Real$^{\uparrow}$ read factors through the displayed carrier inventory. □

Theorem 429.12 (RegularCauchyDifference obligation package)**.** *For an accepted* RegularCauchyDifference$^{\uparrow}$ *carrier*

$$D = (X, Y, S, T, Z, H, C, P, N),$$

the local obligation package is exactly the finite package that combines carrier admission for D, classifier transport for the source rows, aligned windows, dyadic subtraction rows, tolerance ledger, and optional null-sequence row, regularity closure of the displayed pointwise difference, tolerance-ledger exactness for T, NullSequence$^{\uparrow}$ *consumption through Z, and the* Real$^{\uparrow}$ *seal route through the same displayed rows.*

Proof. Carrier admission is Definition 429.1 together with Definition 429.2. These identify the accepted object as the finite tuple D and list the source rows, schedule, ledger, consumer row, transport rows, continuation rows, package provenance, and local naming row.

Classifier transport and regularity closure are the next two obligations. Apply Definition 429.4 to move the source rows, aligned windows, subtraction rows, ledger, and optional consumer row

componentwise. Apply Definition 429.3 and Definition 429.9 to repack the transported observations as the displayed regular rational difference row.

The consumer obligations are exhausted by the ledger and seal theorems. Use Definition 429.5 for the tolerance row, Definition 429.7 for zero-shrinkage reads, and Definition 429.11 for Real-facing reads. The non-escape boundary of Definition 429.8 shows that no quotient stream, selected limit, host real equality, or ambient completeness row is available outside this package. □

Theorem 429.13 (RegularCauchyDifference schedule exactness)**.** *Every pointwise difference read exported by an accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *carrier* $D = (X, Y, S, T, Z, H, C, P, N)$ *is generated by the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows named by* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subtraction rows named by* S*, and the transported tolerance ledger* T*. In particular, a difference read has no alternative source outside the displayed source rows* X, Y*, schedule* S*, and ledger* T*.*

Proof. Begin with the displayed difference read. By Definition 429.9, its windows are the aligned $\mathsf{StreamName}^{\uparrow}$ windows named by S, its rational observations are the $\mathsf{DyadicRatCore}^{\uparrow}$ subtraction rows named by S, and its tolerances are the entries of T.

The exactness of the tolerance coordinate is Definition 429.5. It forces every output tolerance request through the transported ledger rather than an external modulus or host equality witness.

Finally apply Definition 429.8. Since the carrier exposes no quotient stream, selected limit, host real equality, or ambient completeness object, every exported pointwise difference read is precisely the scheduled finite read through X, Y, S, T. □

Theorem 429.14 (RegularCauchyDifference consumer exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *carrier factors through the displayed packet*

$$D = (X, Y, S, T, Z, H, C, P, N).$$

The regular consumer reads the displayed difference row through X, Y, S, T*; the null-sequence consumer reads the zero-shrinkage surface through* Z *and the same ledger; and the Real-facing consumer reads only the seal route supplied by those carrier rows.*

Proof. For the $\mathsf{RegSeqRat}^{\uparrow}$ consumer, apply Definition 429.3 and Definition 429.9. The consumer receives the displayed pointwise difference row through the two source rows, aligned windows, dyadic subtraction rows, and tolerance ledger.

For the $\mathsf{NullSequence}^{\uparrow}$ consumer, apply Definition 429.7. The zero-shrinkage row Z consumes the same displayed difference and the same tolerance ledger before exposing any downstream evidence.

For the $\mathsf{Real}^{\uparrow}$ consumer, apply Definition 429.6 and Definition 429.11. Both routes factor through the displayed regular row, optional null-sequence row, transport rows, continuation rows, package provenance, and local $\mathsf{NameCert}$ row. The non-escape boundary then rules out any consumer path outside D. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDifference}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyDifference}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a pointwise $\mathsf{DyadicRatCore}^{\uparrow}$ subtraction schedule over aligned $\mathsf{StreamName}^{\uparrow}$ windows, a transported tolerance ledger, a $\mathsf{NullSequence}^{\uparrow}$ consumer row when the difference shrinks, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifference}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped dependency surface is bound by Definition 429.12, Definition 429.13, and Definition 429.14: all BHist carrier rows, hsame transports, Cont routes, Pkg provenance, local NameCert data, $\mathsf{NullSequence}^{\uparrow}$ consumption, $\mathsf{RegSeqRat}^{\uparrow}$ difference reads, and $\mathsf{Real}^{\uparrow}$ seal routing factor through the same finite packet.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDifference}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.RegularCauchyDifferenceUp.RegularCauchyDifferenceCarrier\_tolerance\_ledger\_exactness`
Bridge status: paperBridge
Origin: human
Not claimed: This chapter does not close quotient streams, host real equality, selected limits, ambient completeness, completed-real subtraction laws, or bridge checking.
Upgrade path: No theory-axis upgrade remains; bridge-checked verification requires a checked standard bridge for the same subtraction schedule and tolerance ledger.

Chapter 430

A Concrete Naming Certificate for $\mathsf{ArchimedeanReal}^{\uparrow}$

The $\mathsf{ArchimedeanReal}^{\uparrow}$ interface records the finite rational-bound surface for a displayed $\mathsf{Real}^{\uparrow}$ name. Following Bishop and Bridges, *Constructive Analysis*, boundedness is read from explicit rational witnesses and dyadic inspection rows, not from a host supremum, quotient equality of streams, a choice-selected bound, or ambient completeness.

The packet is a $\mathsf{Real}^{\uparrow}$ consumer and therefore reads the displayed real-name seal through chapter 38 before exporting rational-bound rows. This sibling edge is substantive: the first coordinate of the Archimedean carrier is the $\mathsf{Real}^{\uparrow}$ name R, while Q, D, S, G, B are the bound, inspection, and ledger rows surrounding that real-name seal.

Definition 430.1 (ArchimedeanReal carrier)**.** *An* $\mathsf{ArchimedeanReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name whose rational boundedness is being certified;*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *row naming the exported rational bound;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row that presents the same bound at the dyadic inspection surface consumed by real-name windows;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite observations of R are read;*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attached to the real name, so that each inspected rational observation has a regular-Cauchy source;*
6. *B is the finite bound ledger comparing the inspected G observations against Q and D at the requested precision;*
7. *H stores componentwise* hsame *transports for the real name, rational bound row, dyadic bound row, stream window, regular-Cauchy handoff, and bound ledger;*
8. *C stores the* Cont *routes that expose the finite real-name windows, dyadic bound checks, regular-Cauchy reads, and rational-bound export;*
9. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanReal}^{\uparrow}}$ *row.*

The packet is an Archimedean bound certificate for a displayed real name. It stores no host supremum, no selected least bound, no quotient stream equality, and no ambient completeness object.

Lean4 (definition) `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealUp`

Theorem 430.2 (ArchimedeanReal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ArchimedeanReal}^{\uparrow}}$ *surface for the carrier of Definition 430.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet A with a* $\mathsf{Real}^{\uparrow}$ *name, a* $\mathsf{Rat}^{\uparrow}$ *bound row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound presentation, a* $\mathsf{StreamName}^{\uparrow}$ *inspection window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, a finite bound ledger, transport rows, continuation rows, package provenance, and local naming data;*
2. *classifier transport: classification transports the real name, rational bound, dyadic bound presentation, stream window, regular-Cauchy handoff, and bound ledger componentwise through the stored* hsame *and* Cont *routes;*
3. *stability: replacing R, Q, D, S, or G by classified rows preserves the same kind of finite rational-bound ledger and exposes the transported bound row;*
4. *ledger exactness: every accepted bound read is the displayed comparison in B between the inspected regular-Cauchy observations and the rational or dyadic bound rows Q, D;*
5. *rational-bound export: every* $\mathsf{Real}^{\uparrow}$ *order, locatedness, or completion-facing consumer first receives $Q, D, S, G, B, H, C, P, N$ before it can use the bound.*

Proof. Carrier admission is projection from the tuple in Definition 430.1. The real name, rational bound, dyadic presentation, stream window, regular-Cauchy handoff, bound ledger, transport routes, continuation routes, provenance, and local naming row are all explicit coordinates.

Classifier transport is componentwise. The $\mathsf{Real}^{\uparrow}$ name transports together with its $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, while the $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ bound rows transport the comparison side of the ledger.

The stability and exactness clauses then read only the finite ledger B. Since B compares the displayed regular-Cauchy observations with the displayed rational and dyadic bound rows, the exported bound cannot be replaced by a host supremum, a choice-selected bound, quotient stream equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier namecert obligations`

Theorem 430.3 (ArchimedeanReal rational-bound witness)**.** *Every accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet $A = (R, Q, D, S, G, B, H, C, P, N)$ exposes an explicit rational-bound witness for the* $\mathsf{Real}^{\uparrow}$ *name R: the witness is the pair of displayed rows $Q : \mathsf{Rat}^{\uparrow}$ and $D : \mathsf{DyadicRatCore}^{\uparrow}$ together with the finite ledger B comparing all inspected* $\mathsf{RegSeqRat}^{\uparrow}$ *observations of R against those rows. The witness is available without a host supremum, a choice-selected least bound, quotient stream equality, or ambient completeness.*

Proof. Start with the carrier inventory in Definition 430.1. The bound rows Q and D are coordinates of the accepted packet, and the real-name observations are read only through S and G.

Apply the ledger-exactness clause of Definition 430.2. Each finite inspection request is compared against the displayed rational row Q and the dyadic row D by the ledger B before any downstream consumer receives the bound.

Thus the exported witness is exactly the finite pair (Q, D) with its ledger B and provenance routes H, C, P, N. No coordinate of the packet is a supremum operator, a selected least bound, a quotient of streams, or a completeness principle, so none of those objects is available to a consumer of the rational-bound witness. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealPacket rational bound witness`

Theorem 430.4 (ArchimedeanReal rational-dyadic bound coherence)**.** *For every accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet* $A = (R, Q, D, S, G, B, H, C, P, N)$*, the* $\mathsf{Rat}^{\uparrow}$ *row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *form one coherent bound witness for the displayed* $\mathsf{Real}^{\uparrow}$ *name* R*. Every* $\mathsf{Real}^{\uparrow}$ *order, locatedness, completion, integer-part, or decimal-prefix consumer that reads either row must read both rows through the same finite ledger* B*,* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* G*,* hsame *transports* H*,* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ArchimedeanReal}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier rat dyadic bound coherence`

Proof. Project the carrier in Definition 430.1. The accepted packet exposes Q and D as named coordinates of a single finite $\mathsf{ArchimedeanReal}^{\uparrow}$ carrier rather than as two independent bound witnesses.

Apply Definition 430.2. Its ledger-exactness and rational-bound export clauses say that the comparison of inspected $\mathsf{RegSeqRat}^{\uparrow}$ observations against Q and D is mediated by the same ledger B, while the exported consumer surface keeps the same S, G, H, C, P, N rows.

Apply Definition 430.3. The public witness for the displayed real name is the pair Q, D together with the finite ledger B, so a consumer cannot take the $\mathsf{Rat}^{\uparrow}$ row without the dyadic presentation or take the dyadic row without the rational row.

Finally use Definition 430.7. Since the packet has no host supremum, selected least bound, quotient stream equality, host limit, or ambient completeness coordinate, there is no second bound surface through which a downstream order, locatedness, completion, integer-part, or decimal-prefix consumer could separate the two rows. □

Theorem 430.5 (ArchimedeanReal transported bound stability)**.** *Let* $A = (R, Q, D, S, G, B, H, C, P, N)$ *and* $A' = (R', Q', D', S', G', B', H', C', P', N')$ *be accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packets. If the real-name row, rational bound row, dyadic bound row, stream window, regular-Cauchy handoff, bound ledger, transport table, continuation table, provenance row, and local naming row are replaced only by rows classified through the displayed* hsame *and* Cont *routes, then the rational-bound witness of Definition 430.3 is transported unchanged as a witness surface: the exported* $\mathsf{Rat}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, and finite ledger comparison remain the same displayed bound rows for the classified* $\mathsf{Real}^{\uparrow}$ *name.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier transported bound stability`

Proof. The carrier of Definition 430.1 stores Q, D, and B as explicit coordinates, together with the transport table H and continuation table C. The NameCert obligations state that classification acts componentwise on exactly these rows.

Apply Definition 430.3. The witness is not an external bound; it is the visible pair Q, D with the ledger B comparing inspected $\mathsf{RegSeqRat}^{\uparrow}$ observations against those rows.

After replacing the packet by classified rows, the displayed hsame and Cont routes carry the same Q, D, B witness surface to A'. Therefore the rational and dyadic bound rows remain stable for the transported $\mathsf{Real}^{\uparrow}$ name. □

Theorem 430.6 (ArchimedeanReal consumer bound handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *order, locatedness, or completion-facing consumer of an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet receives the displayed rows* $Q, D, S, G, B, H, C, P, N$ *before it uses the bound on* R*. In particular, the consumer reads the rational bound through* $\mathsf{Rat}^{\uparrow}$*, the dyadic presentation through* $\mathsf{DyadicRatCore}^{\uparrow}$*, finite observations through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, and the comparison through the ledger* B *with its transport, continuation, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier consumer bound handoff`

Proof. Start from the carrier tuple. The real name R is not exported with a bare boundedness assertion; the packet also carries $Q, D, S, G, B, H, C, P, N$.

The rational-bound witness theorem identifies the witness used by a consumer as Q, D together with the ledger B. The NameCert obligations further require the consumer-facing export to preserve the stream window, regular-Cauchy handoff, transport rows, continuation routes, provenance, and local naming row.

Thus any order, locatedness, or completion-facing read of the bound must pass through those displayed rows before the bound is used. The handoff is a finite BEDC packet over the visible witness surface. □

Theorem 430.7 (ArchimedeanReal bound-ledger non-escape)**.** *An accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet exports no supremum, no selected least bound, no quotient stream equality, no host limit, and no ambient completeness coordinate. Its bound surface is only the displayed* $\mathsf{Rat}^{\uparrow}$ *row* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*, finite inspection rows* S, G*, ledger* B*, transport rows* H*, continuation rows* C*, provenance row* P*, and local* $\mathsf{NameCert}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier bound ledger nonescape`

Proof. Inspect Definition 430.1. The listed coordinates are exactly $R, Q, D, S, G, B, H, C, P, N$; none of them is a supremum operator, selected least-bound witness, quotient stream equality, host limit, or ambient completeness datum.

The NameCert obligations and rational-bound witness theorem both read boundedness from the same finite ledger surface. Since those theorems use only the carrier coordinates and their displayed transports, there is no additional export channel for a host-side bound object.

Therefore every consumer of the Archimedean bound remains inside the finite $Q, D, S, G, B, H, C, P, N$ ledger boundary. □

Theorem 430.8 (ArchimedeanReal scope closure)**.** *The carrier of Definition 430.1, the local* $\mathsf{NameCert}$ *obligations of Definition 430.2, the rational-bound witness of Definition 430.3, transported-bound stability from Definition 430.5, the consumer handoff of Definition 430.6, and the non-escape boundary of Definition 430.7 jointly bind every* $\mathsf{ArchimedeanReal}^{\uparrow}$ *read to the displayed* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

No accepted read exports a supremum, selected least bound, quotient stream equality, host limit, or ambient completeness coordinate.

Proof. Start with the carrier. It lists exactly the displayed real name, rational and dyadic bound rows, stream window, regular-Cauchy handoff, finite bound ledger, transport rows, continuation rows, package provenance, and local naming row.

Apply the NameCert obligations, rational-bound witness, transported-bound stability, and consumer handoff. These rows say that classification, bound export, transported witnesses, and downstream consumer reads all pass through the same $Q, D, S, G, B, H, C, P, N$ surface.

Finally apply the bound-ledger non-escape theorem. Since the only exported bound data are the displayed rational and dyadic rows with their finite ledger and routes, the scoped boundary contains no supremum, selected least bound, quotient stream equality, host limit, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier scope closure`

Theorem 430.9 (ArchimedeanReal public bound export)**.** *Every accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet* $A = (R, Q, D, S, G, B, H, C, P, N)$ *exports the public bound surface for the displayed* $\mathsf{Real}^{\uparrow}$ *name* R *exactly through the* $\mathsf{Rat}^{\uparrow}$ *bound row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *presentation row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *inspection windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* G*, the finite bound ledger* B*, componentwise transports* H*, continuation routes* C*, package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanReal}^{\uparrow}}$ *row* N*. A public order, locatedness, or completion-facing read of the Archimedean bound receives this displayed surface and no host supremum, selected least bound, quotient stream equality, host limit, or ambient completeness coordinate.*

Proof. Start with the NameCert obligations of Definition 430.2. They admit the carrier and bind the real name, rational bound, dyadic presentation, stream windows, regular-Cauchy handoff, bound ledger, transport rows, continuation rows, provenance, and local naming data as one finite packet.

The public bound itself is the rational-bound witness of Definition 430.3. Its visible witness rows are Q and D, and the ledger B compares the inspected regular-Cauchy observations against those rows.

Transported-bound stability from Definition 430.5 keeps that witness surface attached to classified packets. The consumer handoff theorem then forces every order, locatedness, or completion-facing consumer to pass through $Q, D, S, G, B, H, C, P, N$ before using the bound.

Finally apply Definition 430.7. Since the accepted packet has no carrier coordinate for a host supremum, selected least bound, quotient stream equality, host limit, or ambient completeness object, the public export is exactly the displayed BEDC bound surface. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier public bound export`

Theorem 430.10 (ArchimedeanReal budgeted observation handoff)**.** *Let a* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer accept an* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N).$$

Before the consumer may use the Archimedean rational-bound rows $Q, D, S, G, B, H, C, P, N$*, it must first read the finite* $\mathsf{RealObservationBudget}^{\uparrow}$ *budget rows for the relevant precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise transports, continuation routes, package provenance, and local naming row. The handoff combines the public bound export of Definition 430.9 with the budget-route readiness of Definition 611.18; it exports no host supremum, hidden limit selector, quotient stream equality, or ambient completeness.*

Proof. Project the accepted Archimedean packet A. The public bound export theorem identifies the only Archimedean bound surface available to a consumer: the rows $Q, D, S, G, B, H, C, P, N$ attached to the displayed $\mathsf{Real}^{\uparrow}$ name.

Now route the consumer through the finite observation budget. The readiness theorem for $\mathsf{RealObservationBudget}^{\uparrow}$ says that a completion-facing read must display the finite budget packet before it may reach the $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal route used downstream.

The two routes therefore compose in one order: budget rows first, Archimedean bound rows second. The non-escape clauses already present in the Archimedean carrier and in the observation-budget package exclude a host supremum, hidden limit selector, quotient stream equality, and ambient completeness from the combined handoff. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier budgeted observation handoff`

Theorem 430.11 (ArchimedeanReal budgeted Cauchy bound)**.** *Let a* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer display the finite budget rows for a precision request and accept an* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N).$$

Then every Cauchy-tail or completion-facing consumer receives the rational and dyadic bound rows Q, D *through the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, bound ledger, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *route* S, G, B, H, C, P, N*. The bound witness is available only through the composed budgeted route and exports no host supremum, selected least bound, quotient stream equality, hidden limit selector, or ambient completeness coordinate.*

Proof. Start with the public bound export of Definition 430.9. It identifies the Archimedean bound surface for the displayed $\mathsf{Real}^{\uparrow}$ name as the rational row Q, the dyadic row D, the stream windows S, the regular handoff G, the bound ledger B, and the structural rows H, C, P, N.

Now compose this surface with the budgeted observation handoff of Definition 430.10. The consumer first displays the finite $\mathsf{RealObservationBudget}^{\uparrow}$ rows for the precision request and only then reads the Archimedean bound rows.

The resulting Cauchy-tail or completion-facing read therefore receives a bound witness through the same $Q, D, S, G, B, H, C, P, N$ route already exposed by the accepted packet. Since neither theorem supplies a host supremum, selected least bound, quotient stream equality, hidden limit selector, or ambient completeness coordinate, the composed budgeted read cannot export one. □

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier budgeted cauchy bound`

Theorem 430.12 (ArchimedeanReal dyadic threshold totality)**.** *Let an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

be read by a finite $\mathsf{RealObservationBudget}^{\uparrow}$ *request carried by a unary* BHist *budget row. For every such finite request, there is a dyadic threshold row whose source is the displayed rational-bound witness* Q, D *of* A*, whose classifier is stable under the componentwise* hsame *transport rows, and whose ledger route is obtained by composing the Archimedean bound witness with the finite-window request route of* $\mathsf{RealObservationBudget}^{\uparrow}$*. The row is a finite threshold surface over* BHist*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *only; it exports no trichotomy row, host-real completeness object, hidden limit selector, quotient stream equality, or least-bound selector.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier dyadic threshold totality`

Proof. Start with the rational-bound witness of Definition 430.3. It displays the bound source through the rows Q, D and records the finite ledger comparison through B, so the requested threshold has a visible Archimedean source before any budget row is consumed.

Next compose with the budgeted observation handoff of Definition 430.10. The handoff requires the unary budget request to expose its finite $\mathsf{RealObservationBudget}^{\uparrow}$ rows before reading the Archimedean bound rows.

Use the $\mathsf{RealObservationBudget}^{\uparrow}$ window coverage and finite route coverage rows. By Definition 611.3, each dyadic readback used by the request is covered by a displayed finite $\mathsf{StreamName}^{\uparrow}$ window, and by Definition 611.20 the admitted route remains inside the precision, window, dyadic, regular handoff, seal, transport, continuation, provenance, and local naming rows.

The classifier stability is then the componentwise hsame stability already carried by the two packets. Since neither packet has coordinates for trichotomy, host-real completeness, hidden limit selection, quotient stream equality, or a selected least bound, the composed dyadic threshold row cannot export them. □

Theorem 430.13 (ArchimedeanReal Cauchy threshold forward route)**.** *Let a* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or completion-facing consumer use an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N).$$

Before any Cauchy-tail or $\mathsf{Real}^{\uparrow}$*-seal read is exported, the consumer must read the rational bound row* Q*, the dyadic bound row* D*, the finite stream windows* S*, the regular handoff* G*, and the threshold ledger obtained from* B, H, C, P, N *through the dyadic threshold route. The resulting forward route exports the Archimedean bound surface to Cauchy-facing consumers without importing a supremum, trichotomy, quotient stream equality, selected limit, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCauchyThresholdForwardRoute`

Proof. Begin with the public bound export of Definition 430.9. It identifies $Q, D, S, G, B, H, C, P, N$ as the visible bound surface attached to the displayed $\mathsf{Real}^{\uparrow}$ name.

Next apply the budgeted observation handoff of Definition 430.10. A precision-sensitive Cauchy read must pass through the finite observation-budget route before it consumes the Archimedean rational and dyadic rows.

Finally use the dyadic threshold totality theorem of Definition 430.12. The threshold row is sourced from Q, D, read through the same stream windows and regular handoff, and transported by H, C, P, N. Thus any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Cauchy-tail, or completion-facing export has already consumed the stated Archimedean threshold surface. □

Theorem 430.14 (ArchimedeanReal regular-Cauchy induction readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing Archimedean threshold read from an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

is produced by finite induction over the displayed $\mathsf{StreamName}^{\uparrow}$ *window rows* S *and the* $\mathsf{DyadicArchimedean}^{\uparrow}$ *scale rows supplied through* D, B*. The induction starts from the public bound surface* $Q, D, S, G, B, H, C, P, N$*, advances one displayed window request at a time, records the resulting dyadic threshold row, and then hands the completed regular-Cauchy readback to* $\mathsf{Real}^{\uparrow}$ *through the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. It exports no supremum, trichotomy, quotient stream equality, selected limit, countable-choice schedule, or ambient completeness coordinate.*

Proof. Induct on the displayed finite list of $\mathsf{StreamName}^{\uparrow}$ window requests supplied by the $\mathsf{RealObservationBudget}^{\uparrow}$ row. In the empty-window case, the public bound export of Definition 430.9 already identifies the available Archimedean surface as $Q, D, S, G, B, H, C, P, N$, and the budgeted observation handoff of Definition 430.10 says that the consumer has displayed the finite budget rows before any regular-Cauchy read is attempted. Thus the base read has no hidden observation coordinate to discharge.

For the induction step, append one displayed window request to the already-read finite prefix. The $\mathsf{RealObservationBudget}^{\uparrow}$ window-coverage row makes this added request a named $\mathsf{StreamName}^{\uparrow}$ row, and the finite-route coverage row keeps the request inside the displayed precision, dyadic observation, regular handoff, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming rows.

Apply dyadic threshold totality, Definition 430.12, to that single added request. The threshold row is sourced from the rational and dyadic bound rows Q, D, checked against the finite ledger B, transported componentwise by H, and replayed through C without adding a hidden selector, countable-choice schedule, or quotient-stream coordinate.

The step then hands the obtained threshold row to the $\mathsf{RegSeqRat}^{\uparrow}$ readback row G. Composing with the Cauchy threshold forward route of Definition 430.13 sends the finite prefix, together with the newly added request, to the downstream $\mathsf{Real}^{\uparrow}$ seal only through G, H, C, P, N.

Since the input window family is finite, repeated use of this step exhausts exactly the displayed requests in S. The induction therefore produces the stated regular-Cauchy readback from the budget rows and dyadic threshold rows alone. □

Theorem 430.15 (ArchimedeanReal located-threshold handoff)**.** *Every* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *consumer of an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

must read the rational bound row Q, dyadic bound row D, finite $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff G, bound ledger B, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before using a located threshold. The located threshold is therefore routed through the public Archimedean bound surface and the* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *L10 seal route; it exports no trichotomy, least-bound selector, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the public bound export of Definition 430.9. It identifies $Q, D, S, G, B, H, C, P, N$ as the only bound surface that a consumer may read from the accepted Archimedean packet.

Next apply the dyadic threshold totality theorem of Definition 430.12. The finite threshold row is sourced from Q, D and is transported through the same stream-window, regular handoff, ledger, transport, replay, provenance, and naming rows.

Finally route the consumer through the $\mathsf{ConstructiveRealLine}^{\uparrow}$ L10 seal surface of Definition 4041.3. A $\mathsf{ConstructiveRealLine}^{\uparrow}$ or $\mathsf{LocatedReal}^{\uparrow}$ read reaches the located threshold only after the dyadic, stream, regular-rational, real-seal, located boundary, transport, replay, provenance, and naming rows have been displayed. None of those rows contains trichotomy, a selected least bound, quotient stream equality, or ambient real completeness, so the composed handoff cannot export such data. □

Theorem 430.16 (ArchimedeanReal budgeted bound window exhaustion)**.** *Let an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

be consumed by a finite $\mathsf{RealObservationBudget}^{\uparrow}$ *request that uses the same displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows. Every budgeted Archimedean bound read is exhausted by the displayed observation-budget rows followed by the Archimedean rows $Q, D, S, G, B, H, C, P, N$. No second window family, unbudgeted dyadic threshold, hidden limit selector, quotient stream equality, host supremum, selected least bound, or ambient completeness coordinate may be substituted for that route.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier budgeted bound window exhaustion`

Proof. Begin with the rational-bound witness of Definition 430.3 and the public bound export of Definition 430.9. Together they identify $Q, D, S, G, B, H, C, P, N$ as the only Archimedean bound surface available from the accepted packet.

Now attach the finite observation budget. The budgeted observation handoff of Definition 430.10 orders the read: the consumer first exposes the relevant $\mathsf{RealObservationBudget}^{\uparrow}$ rows, then reads the displayed Archimedean bound rows.

The dyadic threshold totality theorem of Definition 430.12 supplies the finite threshold row for the unary request. The window coverage theorem Definition 611.3 keeps each dyadic readback inside the displayed $\mathsf{StreamName}^{\uparrow}$ windows, and the finite route coverage theorem Definition 611.20 keeps the route inside the budget, window, dyadic, regular handoff, seal, transport, continuation, provenance, and local naming rows.

Finally apply the bound-ledger non-escape theorem of Definition 430.7. Since neither the observation-budget route nor the Archimedean packet provides coordinates for an alternate window family, unbudgeted threshold, hidden selector, quotient stream equality, host supremum, selected least bound, or ambient completeness, every budgeted bound read is exhausted by the displayed rows. □

Theorem 430.17 (ArchimedeanReal public order consumer)**.** *Every public order, locatedness, or completion-facing consumer of an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

reads the packet only through the public bound surface $Q, D, S, G, B, H, C, P, N$*. Precision-sensitive reads must first pass through the relevant* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows before they consume the Archimedean bound rows, and every finite unary budget request obtains its dyadic threshold row through Definition 430.12. The consumer receives no supremum, selected least bound, quotient stream equality, host limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier public order consumer`

Proof. Start with the public bound export of Definition 430.9. It identifies $Q, D, S, G, B, H, C, P, N$ as the public surface seen by order, locatedness, and completion-facing consumers.

Next use the budgeted observation handoff of Definition 430.10. For a precision-sensitive read, the $\mathsf{RealObservationBudget}^{\uparrow}$ rows are consumed before the Archimedean rational-bound rows are used.

The dyadic threshold totality theorem of Definition 430.12 then supplies the finite threshold row for unary budget requests. Its source is the rational-bound witness through Q, D, and its route is the composed budget-window and Archimedean-bound route.

Finally apply the bound-ledger non-escape theorem of Definition 430.7. Since none of the forbidden host objects is a carrier coordinate, the public order consumer cannot project a supremum, selected least bound, quotient stream equality, host limit, or ambient completeness object. □

Theorem 430.18 (ArchimedeanReal integer bracket handoff)**.** *Let an accepted* $\mathsf{ArchimedeanReal}^{\uparrow}$ *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

be read by an integer-part or decimal-prefix consumer of the displayed real name. The consumer must first receive the public Archimedean bound surface $Q, D, S, G, B, H, C, P, N$*. If it then forms*

an ArchimedeanIntegerPart$^{\uparrow}$ *packet, the bracket row factors through the existing real or regular-Cauchy read, the Archimedean comparison row, the dyadic floor scale, the* Int$^{\uparrow}$ *candidate, and the successor bracket rows of Definition 1479.4. The combined handoff exports no host floor, trichotomy row, quotient stream equality, hidden integer search, selected representative, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier integer bracket`

Proof. Start with the rational-bound witness of Definition 430.3. It identifies the Archimedean source available from A as the displayed Q, D rows together with the finite ledger B and its routes H, C, P, N.

Apply the public order consumer theorem of Definition 430.17. An integer-part or decimal-prefix request is therefore not allowed to read the displayed real name through a bare host order; it first passes through the public surface $Q, D, S, G, B, H, C, P, N$.

Now suppose the consumer forms an ArchimedeanIntegerPart$^{\uparrow}$ packet. The bracket-route theorem Definition 1479.4 forces that packet to expose the real or regular-Cauchy read, then the Archimedean comparison row, then the dyadic floor scale, then the Int$^{\uparrow}$ candidate, and only then the successor bracket row. This is exactly the integer-bracket surface available downstream of the public Archimedean bound rows.

Finally use the no-hidden-search boundary of Definition 1479.5. Together with the non-escape boundary already attached to the Archimedean real packet, it excludes host floor selection, trichotomy over arbitrary real names, quotient stream equality, hidden integer search, selected representatives, and ambient completeness from the composed route. □

Closure status for NameCert$_{\text{ArchimedeanReal}^{\uparrow}}$.

Constructive story: An ArchimedeanReal$^{\uparrow}$ packet is generated from a displayed Real$^{\uparrow}$ name, an explicit Rat$^{\uparrow}$ bound row, a matching DyadicRatCore$^{\uparrow}$ presentation, finite StreamName$^{\uparrow}$ windows, a RegSeqRat$^{\uparrow}$ handoff, a bound ledger, componentwise hsame transports, Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{ArchimedeanReal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 430.8, Definition 430.9, Definition 430.17, and Definition 430.18 bind the carrier, NameCert obligations, rational-bound witness, transported-bound stability, public bound export, public order consumer, integer-bracket handoff, and bound-ledger non-escape rows to the displayed Real$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert public interface.

Formal verification: FormalStatus(NameCert$_{\text{ArchimedeanReal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ArchimedeanRealUp.ArchimedeanRealCarrier\_scope\_closure`

Bridge status: none

Origin: human

Not claimed: This chapter does not close least-upper-bound principles, quotient stream equality, selected host limits, ambient real completeness, or bridge checking.

Upgrade path: Bridge checking remains the next formal-axis target for the displayed public bound and order-consumer interface.

Chapter 431

A Concrete Naming Certificate for ObserverRegular$^{\uparrow}$

The ObserverRegular$^{\uparrow}$ chapter is a BEDC-native finite certificate surface. It packages finite prefix-observer data as a BEDC naming certificate for Real$^{\uparrow}$ and Completion$^{\uparrow}$ window schedules. Its public surface is a finite observer ledger over displayed prefixes, resolving states, stream windows, regular-rational readbacks, tolerance rows, transports, continuation routes, provenance, and a local naming row.

The dependency anchors are concrete sibling rows, not background names: chapter 126 supplies the finite windows, chapter 301 supplies the regular-rational readback, chapter 292 supplies the modulus schedule, chapter 291 supplies the dyadic tolerance ledger, chapter 38 supplies the Real$^{\uparrow}$ seal consumers, and chapter 95 supplies the Completion$^{\uparrow}$ readback boundary. An ObserverRegular$^{\uparrow}$ carrier consumes those sibling rows by resolving a finite prefix state before any Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer can read a window or readback.

Falsifiable prediction: Removing the resolving-state row prevents any Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer from distinguishing prefix observation from arbitrary StreamName$^{\uparrow}$ readback in the finite ledger.

Independence witness: Nearest siblings: streamname, regseqrat, cauchymodulus, real, completion. The ObserverRegular packet is not bijective with those siblings because it adds a resolving-state prefix row and schedule lock that none of them expose.

The proposed Lean follow-up contract for lean4/BEDC/Derived/ObserverRegularUp/TasteGate.lean must provide Nontrivial ObserverRegular$^{\uparrow}$ and FieldFaithful ObserverRegular$^{\uparrow}$ instances for the displayed observer packet.

Definition 431.1 (ObserverRegular prefix carrier)**.** *An* ObserverRegular$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, R, S, W, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet and prefix row, listing the retained symbols, the displayed prefix packets, and the finite prefix-extension ledger inspected by the observer;*
2. *R is the resolving-state row, listing the accepted observer states and the finite transition evidence by which a prefix reaches such a state;*
3. *S is the schedule row, naming the* Real$^{\uparrow}$ *and* Completion$^{\uparrow}$ *window schedules for which the observer packet is intended;*

4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed by each accepted resolving state;*
5. *L is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger, pairing each displayed window with the finite rational readback consumed by downstream schedules;*
6. *H stores componentwise* hsame *transports for alphabet rows, prefix rows, resolving-state rows, window rows, readback rows, tolerance rows, provenance, and the local naming row;*
7. *C stores the displayed* Cont *routes for prefix extension, resolving-state inspection, window display, regular-rational readback, tolerance checking, and the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer handoff;*
8. *P is the* Pkg *provenance row naming* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and the consumed local source certificates;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$ *row.*

The classifier compares accepted packets by first transporting them to a common displayed finite prefix row, then comparing the resolving-state rows, the $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and finally the consumer handoff rows. The carrier has no coordinate for quotient streams, host sequence equality, selected limits, ambient completeness, or choice-selected completion data.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularTasteGate single carrier alignment`

Theorem 431.2 (ObserverRegular resolving-window soundness)**.** *For every accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier $O = (A, R, S, W, L, H, C, P, N)$, each accepted observer state in R determines a displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window in W and a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in L. The readback is reached only through the finite prefix row A, the resolving-state row R, the displayed window row W, the stored* hsame *transports, and the displayed* Cont *routes.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier resolving window soundness`

Proof. Project the carrier of Definition 431.1. An accepted observer state is not a host-level predicate; it is one of the finite rows recorded in R together with the displayed prefix evidence from A.

Use the continuation routes in C to inspect that resolving row. The inspection selects a window entry in W, and the same displayed route selects the matching regular-rational readback entry in L.

The transport row H supplies the componentwise equality witnesses needed when the same state is read after prefix transport. Since the only exposed coordinates are A, R, W, L, H, and C, the soundness claim remains inside the finite prefix-observer packet. □

Theorem 431.3 (ObserverRegular prefix-resolution totality)**.** *For every accepted prefix row in an* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier*

$$O = (A, R, S, W, L, H, C, P, N),$$

exactly one of the finite observer routes is consumer-readable. Either the prefix row in A exposes no accepted resolving state, in which case no downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *window or readback row is readable from that prefix; or it exposes one of the displayed resolving-state rows in R, after which the only consumer route is through W, L, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier prefix resolution totality`

Proof. Project the prefix carrier of Definition 431.1. The prefix row A is the sole entry point for a consumer request, and the resolving-state row R is the sole coordinate that can authorize a downstream window or readback.

In the positive case, the accepted prefix exposes one of the displayed resolving-state rows in R. Apply resolving-window soundness to that row; it produces the finite $\mathsf{StreamName}^{\uparrow}$ window in W and the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback in L, and the remaining consumer route passes through H, C, P, N.

In the negative case, the accepted prefix exposes no resolving-state row. Coordinate exhaustion then leaves no row from which a downstream window, readback, $\mathsf{Real}^{\uparrow}$ seal request, or $\mathsf{Completion}^{\uparrow}$ readback can be obtained, because W and L are reached only through R and the displayed continuation route.

Thus every accepted prefix is total as a finite case analysis: either it has no consumer-readable downstream route, or it has exactly the resolving-state route through W, L, H, C, P, N. □

Theorem 431.4 (ObserverRegular NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$ *surface has five obligations:*

1. *carrier habitation: finite alphabet rows, prefix rows, resolving-state rows, window rows, readback ledgers, tolerance rows, transports, continuations, provenance, and the local naming row inhabit the carrier of Definition 431.1;*
2. *classifier transport: accepted carriers are compared only after displayed transport to a common finite prefix row and common resolving-state inspection surface;*
3. *resolving-window soundness: every accepted observer state determines the finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of Definition 431.2;*
4. *ledger exactness: the* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows are read only through the displayed prefix, resolving-state, window, and readback ledgers;*
5. $\mathsf{Real}^{\uparrow}$ *non-escape: every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer read factors through the finite observer ledger and never through quotient streams, host sequence equality, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier namecert obligations`

Proof. Carrier habitation is obtained by choosing finite displayed rows for the alphabet, prefixes, resolving states, windows, readbacks, tolerances, transports, continuations, provenance, and local name. These are exactly the coordinates listed in Definition 431.1.

Classifier transport first moves packets to a common prefix row and then inspects the resolving-state rows through the same displayed continuations. The resolving-window obligation is precisely Definition 431.2.

The ledger exactness and real-name boundary obligations follow from the absence of any carrier coordinate outside the finite observer packet. The tolerance rows and consumer rows can only be reached by projecting the listed coordinates and applying the stored continuation routes. □

Theorem 431.5 (ObserverRegular finite ledger non-escape)**.** *Every consumer-facing read of an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier is exhausted by the finite alphabet and prefix row, resolving-state row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

The ledger exports no quotient stream equality, host sequence equality, selected host limit, ambient completeness principle, or choice-selected completion witness.

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier finite ledger non escape`

Proof. Begin with a requested downstream read. The only entry point into the carrier is the displayed prefix row A, which selects the resolving-state row R and then the window and readback rows W and L.

The tolerance rows are reached by the same displayed readback route. The $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ entries therefore certify only the finite observation already selected by the observer ledger.

Finally route the read through H, C, P, and N. Those coordinates record transport, continuation, provenance, and naming data, not host-level equality, selected limits, or ambient completeness, so no non-displayed object can be exported. □

Theorem 431.6 (ObserverRegular StreamName handoff)**.** *The downstream handoff from an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consists exactly of the displayed window row, regular-rational readback ledger, tolerance rows, transports, continuation routes, package provenance, and local naming row selected by an accepted resolving state.*

For $\mathsf{Completion}^{\uparrow}$ *schedules, the same handoff remains finite: the completion consumer receives the* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback already exposed to the* $\mathsf{Real}^{\uparrow}$ *boundary, together with the displayed tolerance rows. No quotient stream, host sequence equality, selected limit, or ambient completeness coordinate is added.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier streamname handoff`

Proof. Apply Definition 431.2 to the accepted resolving state. It supplies the finite $\mathsf{StreamName}^{\uparrow}$ window and matching $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Apply Definition 431.5. The $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers can inspect only the selected window, readback, tolerance, transport, continuation, provenance, and naming rows.

Thus the handoff is exactly the finite packet described in the statement. Since the completion-facing schedule receives no coordinate beyond those rows, it cannot introduce quotient streams, host sequence equality, selected limits, or ambient completeness. □

Theorem 431.7 (ObserverRegular TasteGate obligation assembly)**.** *For an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier* $O = (A, R, S, W, L, H, C, P, N)$, *the* $\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$ *obligation surface is assembled by the displayed rows* A, R, W, L, H, C, P, *and* N *together with the TasteGate carrier alignment* `BEDC.Derived.ObserverRegularUp.ObserverRegularTasteGate_single_carrier_alignment` *attached to Definition 431.1. The assembly consists of carrier habitation, classifier transport across displayed prefix and resolving-state rows, resolving-window soundness, ledger exactness for tolerance reads, and non-escape of* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers from the finite observer packet.*

Proof. Carrier habitation is the finite packet displayed in Definition 431.1. The TasteGate carrier alignment attached to that definition identifies the singleton carrier used by the chapter with the displayed observer packet, so the obligation is about the rows already present in O.

Classifier transport uses H to move alphabet, prefix, resolving-state, window, readback, tolerance, provenance, and naming rows before comparison. The continuation row C then supplies the displayed inspection routes, so no classifier step ranges over a host stream or an ambient completion object.

Resolving-window soundness is Definition 431.2. Ledger exactness and consumer non-escape are the content of Definition 431.5 and Definition 431.6: every downstream read is routed through the finite observer ledger and its stored transports, continuations, provenance, and local name. □

Theorem 431.8 (ObserverRegular resolving-window ledger exactness)**.** *Every* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance read made by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier* $O = (A, R, S, W, L, H, C, P, N)$ *factors through* A*,* R*,* W*,* L*,* H*,* C*,* P*, and* N*. The schedule row* S *names the intended consumer schedules, but it contributes no separate tolerance witness outside the displayed prefix, resolving-window, readback, transport, continuation, provenance, and local naming rows.*

Proof. Start with a tolerance read requested by a downstream consumer. The read first selects a displayed prefix in A and an accepted resolving state in R.

Apply Definition 431.2. The selected resolving state determines the window row W and the regular-rational readback ledger L, and the read reaches those rows only through the stored transports H and continuations C.

The provenance row P records which $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ certificates are being consumed, while N records the local observer name. By Definition 431.5, there is no carrier coordinate for an external tolerance witness, quotient stream, selected limit, or ambient completeness principle. Thus each tolerance read factors through exactly the displayed finite ledger rows named in the statement. □

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier resolving window ledger exactness`

Theorem 431.9 (ObserverRegular schedule lock for real and completion consumers)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier* $O = (A, R, S, W, L, H, C, P, N)$ *uses the displayed schedule row* S *together with the same finite window row* W*, regular-rational readback ledger* L*, componentwise transport row* H*, continuation row* C*, package provenance* P*, and local naming row* N*. The row* S *may name the intended consumer schedule, but it contributes no independent tolerance witness, completion witness, selected limit, quotient stream equality, or host real equality.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier schedule lock real completion`

Proof. Start with a consumer route into $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$. By Definition 431.2, the accepted resolving state first selects the displayed finite $\mathsf{StreamName}^{\uparrow}$ window W and the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger L through the observer packet.

Apply Definition 431.5 and Definition 431.6. These force the consumer handoff through the same finite window, readback, transport, continuation, provenance, and naming rows; no external stream equality, host sequence equality, selected limit, or ambient completeness coordinate is available.

It remains to account for tolerance and completion-facing reads. The ledger exactness theorem Definition 431.8 says that such reads still factor through A, R, W, L, H, C, P, and N, while the existing TasteGate marker `BEDC.Derived.ObserverRegularUp.taste_gate` records the singleton carrier alignment for the displayed observer packet.

Thus the schedule row S is locked to the same finite rows consumed by $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$. It can name the intended schedule, but it cannot supply an independent tolerance or completion witness beyond the displayed observer ledger. □

Theorem 431.10 (ObserverRegular schedule coherence)**.** *For accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carriers*

$$O = (A, R, S, W, L, H, C, P, N) \quad \textit{and} \quad O' = (A', R', S', W', L', H', C', P', N')$$

whose schedule rows are classified together through the displayed hsame *and* Cont *routes, every* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumer read reaches coherent* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledgers through* W, L, H, C, P, N *and* W', L', H', C', P', N'*. The coherence is finite: it compares the displayed observer rows and exports no quotient stream, host sequence equality, selected limit, or ambient completeness witness.*

Proof. Start with two accepted observer carriers whose schedule rows are related by the displayed transports and continuation routes. The TasteGate alignment attached to Definition 431.1 fixes the singleton carrier surface for the observer packet, so the comparison is made through the visible rows rather than through a host stream.

Apply Definition 431.8 to both carriers. It forces each tolerance and readback request to factor through the accepted prefix row, resolving-state row, window row, readback ledger, transports, continuations, provenance, and local naming row.

Now use Definition 431.9. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer can read the intended schedule only together with the corresponding finite window and readback ledger. Since the two schedule rows are already classified through the displayed H, C and H', C' routes, the consumer reaches coherent windows W, W' and ledgers L, L' through the two finite observer packets.

The carrier inventory then gives non-escape. Neither packet contains a quotient stream, host sequence equality, selected limit, or ambient completeness coordinate, so schedule coherence remains a BEDC row comparison at the displayed observer boundary. □

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier schedule coherence`

Theorem 431.11 (ObserverRegular bridged consumer surface)**.** *Every bridge-ready* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverRegular}^{\uparrow}$ *carrier* $O = (A, R, S, W, L, H, C, P, N)$ *factors through the displayed rows* $A, R, S, W, L, H, C, P, N$ *together with the TasteGate single-carrier alignment cited at Definition 431.1. The consumer receives the finite prefix row, resolving-state row, schedule row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance reads, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row, and it receives no quotient stream, host sequence equality, selected limit, ambient completeness principle, or choice-selected completion witness.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier bridged consumer surface`

Proof. Start with the accepted carrier and the TasteGate single-carrier alignment. The alignment fixes the bridge-facing consumer at the displayed observer packet rather than at an ambient stream or completion object.

Resolving-window soundness supplies the first route: the consumer reaches a $\mathsf{StreamName}^{\uparrow}$ window and a $\mathsf{RegSeqRat}^{\uparrow}$ readback only through A, R, W, L, H, C. Finite ledger non-escape then prevents that route from leaving the listed BEDC rows.

Use the StreamName handoff and schedule lock for the consumer boundary. These force every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read to use the same schedule row S, window row W, readback ledger L, transports H, continuations C, provenance P, and local name N.

Finally apply schedule coherence. Since coherent consumer reads are compared through the displayed rows and the TasteGate carrier marker, no quotient stream, host sequence equality, selected limit, ambient completeness principle, or choice-selected completion witness can be supplied by the bridge-ready surface. □

Theorem 431.12 (ObserverRegular formal-target boundary)**.** *The Lean-facing formal target for* $\mathsf{ObserverRegular}^{\uparrow}$ *is exactly the finite packet*

$$O = (A, R, S, W, L, H, C, P, N),$$

the resolving-window soundness row of Definition 431.2, prefix-resolution totality from Definition 431.3, finite-ledger non-escape from Definition 431.5, the schedule lock for $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers from Definition 431.9, schedule coherence from Definition 431.10, and the bridged consumer surface of Definition 431.11. The target excludes quotient streams, host sequence equality, selected limits, ambient completeness, completion bridges, and any proof not routed through* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.ObserverRegularCarrier formal target boundary`

Proof. The target begins with the carrier of Definition 431.1. Its finite rows are the only data admitted to the proof surface: prefix and resolving-state data, schedule data, stream windows, regular-rational readbacks, transports, continuations, provenance, and the local name.

The soundness and totality clauses are the first route constraints. They force every accepted prefix either to stop before a consumer-readable row or to pass through the displayed resolving state before reaching a stream window and readback.

The non-escape, schedule-lock, and schedule-coherence clauses then define the consumer boundary. They keep every $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ read inside the displayed observer ledger and compare schedules only through the stored hsame and Cont rows.

The final exclusion is coordinate exhaustion. Since the target contains no quotient stream coordinate, host sequence equality coordinate, selected-limit coordinate, ambient-completeness coordinate, or completion-bridge coordinate, any proof using such data is outside the stated Lean-facing boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObserverRegular}^{\uparrow}$ packet is generated by a finite alphabet and prefix row, resolving-state row, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger, $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.ObserverRegularUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature scope records the finite observer ledger, schedule lock, StreamName handoff, non-escape boundary, bridge-facing consumer surface of Definition 431.11, and the sibling $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumer boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverRegular}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host sequence equality, selected limits, ambient completeness, completion bridges, or Lean verification.

Upgrade path: No further paper closure grade remains; Lean verification and bridge checking are recorded on their separate axes.

Chapter 432

A Concrete Naming Certificate for $\mathsf{FibonacciInverseLimit}^{\uparrow}$

The $\mathsf{FibonacciInverseLimit}^{\uparrow}$ interface is a BEDC-native finite certificate surface.

The $\mathsf{FibonacciInverseLimit}^{\uparrow}$ interface names the finite-prefix surface of a projective Fibonacci-modulus tower. It records compatible prefixes over already displayed BEDC packets and does not import external rings, quotient completion, choice, topology, Haar measure, compactness, or an external profinite-ring theorem.

The inverse-limit packet consumes finite $\mathsf{FibonacciCube}^{\uparrow}$ modulus-coordinate rows through chapter 401 and $\mathsf{StreamName}^{\uparrow}$ window scheduling through chapter 126. The $\mathsf{GoldenMeanShift}^{\uparrow}$ admissibility row remains a local finite-window dependency because no concrete $\mathsf{GoldenMeanShift}^{\uparrow}$ chapter label is currently part of the carrier surface.

Definition 432.1 (Fibonacci inverse-limit carrier)**.** *A* $\mathsf{FibonacciInverseLimit}^{\uparrow}$ *carrier is a finite compatible prefix packet*

$$Q = (G, F, W, R, H, C, P, N)$$

with the following visible rows:

1. *G is a finite* $\mathsf{GoldenMeanShift}^{\uparrow}$ *admissibility ledger for the inspected prefix windows;*
2. *F is a finite* $\mathsf{FibonacciCube}^{\uparrow}$ *modulus-coordinate ledger for the same windows;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule selecting exactly the displayed prefix packets;*
4. *R records the finite projection rows between adjacent Fibonacci-modulus windows and their compatibility witnesses;*
5. *H records componentwise* hsame *transports for the golden-mean rows, cube rows, stream-window rows, and projection rows inside* BHist*;*
6. *C records the displayed* Cont *routes by which consumers read a smaller prefix from a larger recorded prefix;*
7. *P is the* Pkg *provenance over* $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{FibonacciCube}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont*, and the local naming row;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FibonacciInverseLimit}^{\uparrow}}$ *row.*

The carrier contains only finite prefix packets and their displayed projection routes. It contains no completed topological carrier, no quotient of streams, no selected infinite point, no Haar measure row, and no external profinite-ring theorem.

Theorem 432.2 (Fibonacci inverse-limit NameCert obligations)**.** *Every accepted* $\mathsf{FibonacciInverseLimit}^{\uparrow}$ *packet with carrier Definition 432.1 carries exactly the following local naming-certificate obligations:*

1. *carrier habitation by the empty compatible prefix packet with empty* $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{FibonacciCube}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and projection rows;*
2. *prefix classifier transport along the componentwise* hsame *rows for* G, F, W, R, H, C, P, *and* N*;*
3. *projection compatibility for every displayed finite route from a larger Fibonacci-modulus prefix to a smaller prefix;*
4. *finite-prefix exhaustion: every public finite read of the inverse-limit surface factors through some row already selected by* W *and* R*;*
5. *ledger non-escape: downstream consumers can read only* G, F, W, R, H, C, P, *and* N.

Lean4 ✓ `BEDC.Derived.FibonacciInverseLimitUp.FibonacciInverseLimitPacket namecert obligation surface`

Proof. Carrier habitation uses the empty prefix window. The empty $\mathsf{GoldenMeanShift}^{\uparrow}$ and $\mathsf{FibonacciCube}^{\uparrow}$ rows, the empty $\mathsf{StreamName}^{\uparrow}$ schedule, and the empty projection ledger form a finite BHist packet with identity transport and local naming evidence.

Classifier transport is componentwise. Moving a prefix row by hsame also moves its golden-mean admissibility row, Fibonacci-cube modulus row, selected stream window, and projection witness through the matching entries of H.

Projection compatibility is the content of the route ledger R and the consumer routes C. A smaller prefix is obtained only by following a displayed route from a larger recorded prefix, so no hidden tower map is available.

Finite-prefix exhaustion follows from the selected window schedule W. A public read must first choose a recorded window and then read through the corresponding projection row in R.

The non-escape clause is a coordinate inspection of Definition 432.1. Since the carrier has no coordinate for topology, quotient completion, Haar measure, a selected infinite point, or an external profinite-ring theorem, none of those objects can be exported by its local certificate. □

Theorem 432.3 (Fibonacci inverse-limit prefix projection exactness)**.** *Let* $Q = (G, F, W, R, H, C, P, N)$ *be an accepted* $\mathsf{FibonacciInverseLimit}^{\uparrow}$ *carrier. For every displayed pair of Fibonacci-modulus prefix windows in* W*, each consumer read of the smaller window factors through the unique recorded projection row in* R*, then through the transported* $\mathsf{GoldenMeanShift}^{\uparrow}$ *and* $\mathsf{FibonacciCube}^{\uparrow}$ *rows named by* G *and* F*. Thus a finite prefix is exact relative to the recorded tower data and is not reconstructed from an unrecorded infinite point.*

Proof. Begin with a consumer read of a smaller prefix. The schedule W selects the requested finite window before any coordinate can be inspected.

The projection ledger R then supplies the displayed route from the larger recorded prefix to the smaller one. The route is stabilized by the matching hsame entries in H and by the Cont read in C.

Finally, the golden-mean and Fibonacci-cube rows G and F verify that the projected row remains inside the finite admissibility and modulus surfaces. No step asks for a completed inverse-limit point or a quotient of compatible streams. □

Theorem 432.4 (Fibonacci inverse-limit ledger non-escape)**.** *The public boundary of an accepted* $\mathsf{FibonacciInverseLimit}^{\uparrow}$ *packet exports only finite* $\mathsf{GoldenMeanShift}^{\uparrow}$ *admissibility rows, finite* $\mathsf{FibonacciCube}^{\uparrow}$ *modulus rows, selected* $\mathsf{StreamName}^{\uparrow}$ *windows, projection-compatibility rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence. It exports no compactness theorem, no topology of the inverse limit, no Haar measure, no quotient completion, no choice-selected infinite point, and no external profinite-ring theorem.*

Proof. Inspect the carrier coordinates of Definition 432.1. The rows G, F, W, R, H, C, P, and N exhaust the public packet.

By Definition 432.3, every finite consumer read reaches a selected window and a displayed projection row before it reaches a prefix value. This keeps all reads inside finite prefix data.

Therefore the packet can serve as a concrete BEDC naming certificate for finite inverse-limit projections, but it does not export compactness, inverse-limit topology, Haar measure, quotient completion, choice, or an external profinite-ring theorem. □

Theorem 432.5 (Fibonacci inverse-limit window-extension exhaustion)**.** *Let* $Q = (G, F, W, R, H, C, P, N)$ *be an accepted* $\mathsf{FibonacciInverseLimit}^{\uparrow}$ *carrier. Extending its finite prefix window adds only named larger-window rows to* W *and recorded projection rows to* R*. Every read of an older prefix either factors through the older selected window already present in* W*, or through a newly recorded larger window followed by the displayed projection row in* R *back to the older prefix. In both cases the provenance rows* G, F, W, H, C, P, N *remain the same public coordinates of the packet, and no completed inverse-limit point, quotient stream, selected infinite prefix, topology, Haar measure, or external profinite-ring theorem is imported.*

Proof. Inspect the carrier tuple $Q = (G, F, W, R, H, C, P, N)$. A window extension can add rows only to the finite window schedule W and the projection ledger R; the admissibility rows, modulus rows, transports, continuations, provenance, and local certificate remain the displayed coordinates of the same packet.

For an older-prefix read, apply Definition 432.3. If the read uses the old selected window, it is already exact over the old W and R rows. If it uses a larger recorded window, exactness forces the read through the recorded projection row in R before it reaches the older prefix.

The ledger non-escape theorem Definition 432.4 excludes any hidden source for the older prefix. Thus the extension exhausts public reads by named larger-window rows and recorded projections, with G, F, W, H, C, P, N preserved as the visible provenance surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FibonacciInverseLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{FibonacciInverseLimit}^{\uparrow}$ packet is generated from finite $\mathsf{GoldenMeanShift}^{\uparrow}$ admissibility rows, finite $\mathsf{FibonacciCube}^{\uparrow}$ modulus rows, selected $\mathsf{StreamName}^{\uparrow}$ windows, projection-compatibility rows, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FibonacciInverseLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope registers finite compatible prefix packets, prefix projection exactness, and ledger non-escape for Fibonacci-modulus projection surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FibonacciInverseLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim compactness, topology of the inverse limit, Haar measure, quotient completion, choice-selected infinite points, external imports, or external profinite-ring theorems.

Upgrade path: Obligation closure requires formal carrier encoding, prefix classifier transport, projection compatibility, finite-prefix exhaustion, and ledger non-escape under `BEDC.Derived.FibonacciInverseLimitUp`.

Chapter 433

A Concrete Naming Certificate for $\mathsf{DyadicRounding}^{\uparrow}$

The $\mathsf{DyadicRounding}^{\uparrow}$ interface names the finite dyadic-grid enclosure used on the constructive real path. Following Bishop and Bridges, *Constructive Analysis*, it records an adjacent dyadic cell around a regular rational or $\mathsf{Real}^{\uparrow}$ read at a displayed unary precision, without invoking a quotient real, selected limit, trichotomy, or host rounding function. The adjacent endpoint rows consume the dyadic arithmetic surface of chapter 291, and the handoff row is read by the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38.

Before exposing any $\mathsf{Real}^{\uparrow}$-facing enclosure, $\mathsf{DyadicRounding}^{\uparrow}$ consumes the endpoint carrier of chapter 291, the containment surface of chapter 354, and the tolerance ledger of chapter 385.

Definition 433.1 (DyadicRounding carrier)**.** *A* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (x, n, a, b, I, E, L, C, P, N)$$

with the following displayed coordinates:

1. *x is the source read, either a* $\mathsf{Real}^{\uparrow}$ *seal route or a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback already present on the constructive real path;*
2. *n is the* $\mathsf{UnaryHistory}$ *precision request that fixes the dyadic grid scale;*
3. *a and b are adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid endpoints at precision n;*
4. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *containment row witnessing that the inspected rational approximation of x lies inside the interval with endpoints a and b;*
5. *E is the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *tolerance row that bounds the readback error at the same precision;*
6. *L stores the componentwise* hsame *transports for the source read, unary precision, dyadic endpoint rows, interval containment row, and error-budget row;*
7. *C stores the* Cont *routes for reading the source, inspecting the dyadic grid, transporting containment, and exposing the tolerance ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UnaryHistory}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$ *row.*

The carrier is a finite rounding certificate. It stores no completed real representative, no limiting point, no order trichotomy branch, and no host rounding operation.

Theorem 433.2 (DyadicRounding NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$ *surface for the carrier of Definition 433.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet D with source read, unary precision, adjacent dyadic endpoints, interval containment, finite tolerance ledger, transport rows, continuation rows, package provenance, and local naming data;*
2. *classifier transport: classified source reads transport the same precision request, adjacent endpoint pair, containment row, error-budget row, and provenance rows componentwise through L and C;*
3. *grid adjacency: a and b are the displayed adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints at precision n;*
4. *ledger exactness: every tolerance consumed by the containment read is the displayed* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row E;*
5. *real handoff: every* $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer first reads $x, n, a, b, I, E, L, C, P, N$.*

Proof. Carrier admission is projection from Definition 433.1. The source row, precision request, endpoint pair, containment row, tolerance row, transport routes, continuation routes, package row, and local naming row are all visible coordinates of D.

Classifier transport is componentwise. A transported $\mathsf{Real}^{\uparrow}$ seal route or $\mathsf{RegSeqRat}^{\uparrow}$ readback carries the same unary precision, the same adjacent dyadic endpoint rows, and the same interval and budget ledgers through L and C.

The grid and ledger clauses are finite projections. Since the carrier records only the endpoint rows a, b, the containment row I, and the tolerance row E, downstream consumers have no coordinate from which to obtain a quotient real, selected limit, trichotomy split, or host rounding function. □

Theorem 433.3 (DyadicRounding grid sandwich)**.** *For every accepted* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier $D = (x, n, a, b, I, E, L, C, P, N)$, the source read x is exposed to dyadic-grid consumers only through the displayed sandwich by the adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints a and b at precision n, together with the* $\mathsf{RationalInterval}^{\uparrow}$ *containment row I and the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *tolerance row E.*

Lean4 ✓ `BEDC.Derived.DyadicRoundingUp.DyadicRoundingPacket grid sandwich`

Proof. Project the accepted carrier. The precision coordinate n fixes the grid scale, and the endpoint coordinates a and b are the only dyadic grid rows available at that scale.

Read containment through I. This row places the inspected rational approximation of x inside the displayed interval bounded by a and b.

Finally read the tolerance through E. The finite error budget is the only permitted widening of the containment row, so the consumer receives the sandwich data a, I, b, E rather than any ambient rounded value or host real comparison. □

Theorem 433.4 (DyadicRounding ledger exactness)**.** *Every tolerance, containment, and endpoint read exported by an accepted* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier is exhausted by the rows $n, a, b, I, E, L, C, P, N$ of Definition 433.1. No downstream read may replace these rows by an external modulus, completed-real equality, selected limit, trichotomy decision, or host rounding operation.*

Proof. Start with any exported dyadic rounding read. The NameCert obligation surface routes it through the precision request, adjacent endpoint rows, interval containment row, and finite error-budget row.

The transport and continuation rows L and C move only those displayed coordinates. They do not add analytic data beyond the source read and the finite ledgers already recorded in the packet.

Inspecting the carrier inventory completes the exactness claim. Since no coordinate stores an external modulus, completed-real equality, selected limit, trichotomy decision, or host rounding operation, no consumer can read one from the packet. □

Theorem 433.5 (DyadicRounding RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing handoff from an accepted* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier factors through the source row* x*, the unary precision* n*, the adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint pair* a, b*, the* $\mathsf{RationalInterval}^{\uparrow}$ *containment row* I*, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *tolerance row* E*, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The handoff exposes no quotient real, selected limit, trichotomy branch, or host rounding function.*

Proof. First read the source coordinate x. If it is already a $\mathsf{Real}^{\uparrow}$ seal route, the handoff remains on that route; if it is a $\mathsf{RegSeqRat}^{\uparrow}$ readback, the regular rational row is inspected only at the displayed precision n.

Next apply the grid sandwich theorem. The handoff passes through a, b, I, E, so every real-facing approximation is the finite dyadic enclosure with its recorded budget.

Finally apply ledger exactness. The remaining coordinates are provenance and transport rows, not analytic choice data, so the $\mathsf{Real}^{\uparrow}$ consumer receives exactly the finite rounding packet and nothing outside it. □

Theorem 433.6 (DyadicRounding transport stability)**.** *Let* $D = (x, n, a, b, I, E, L, C, P, N)$ *be an accepted* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier. If each displayed coordinate is transported componentwise along the* hsame *rows stored in* L *and then read along the* Cont *routes stored in* C*, the resulting packet is accepted by the same* $\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$ *surface and preserves the source read* x*, unary precision* n*, adjacent endpoints* a, b*, containment row* I*, finite budget row* E*, transport ledger* L*, continuation ledger* C*, package provenance* P*, and local naming row* N*.*

Proof. Start from the carrier inventory in Definition 433.1. The only movable data are the displayed source read, unary precision, endpoint pair, containment row, finite budget row, and provenance rows.

Apply Definition 433.3 after transport. The same transported precision row still selects the adjacent endpoint pair, and the same containment and budget rows provide the finite sandwich.

Apply Definition 433.4 and Definition 433.5. The transported packet exposes no extra rounding operation or real comparison, so the original NameCert obligations are preserved componentwise. □

Theorem 433.7 (DyadicRounding finite-grid coverage)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *read at unary precision* n *from a* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier is covered by the displayed adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints* a, b *together with the* $\mathsf{RationalInterval}^{\uparrow}$ *containment row* I *and the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row* E *of Definition 433.1. No accepted read at that precision can require a dyadic endpoint, interval witness, or tolerance ledger outside the displayed packet.*

Proof. First use Definition 433.3. It places the source read at precision n between the adjacent dyadic endpoints a and b through the containment row I.

Next use Definition 433.4. The only widening of that interval read is the finite budget row E, and every tolerance consumed by the read is exhausted by the displayed ledger.

Finally apply Definition 433.5. A $\mathsf{Real}^{\uparrow}$ read and a $\mathsf{RegSeqRat}^{\uparrow}$ read both pass through the same finite enclosure data before any downstream consumer sees them, so the displayed grid covers every accepted read at precision n. □

Theorem 433.8 (DyadicRounding carrier scope)**.** *The carrier of Definition 433.1 has no coordinate for order trichotomy, host rounding, quotient reals, or selected limits. Its complete analytic surface is the finite grid sandwich of Definition 433.3, the ledger exactness boundary of Definition 433.4, and the* $\mathsf{Real}^{\uparrow}$ *handoff of Definition 433.5.*

Proof. Inspect the packet $D = (x, n, a, b, I, E, L, C, P, N)$ from Definition 433.1. Each coordinate is a finite source, precision, dyadic endpoint, interval, budget, transport, continuation, package, or naming row.

The grid sandwich and ledger exactness theorems identify all analytic data available to consumers: adjacent endpoints, containment, and a finite error budget at the displayed precision.

The $\mathsf{Real}^{\uparrow}$ handoff theorem then accounts for the real-facing route. Since the carrier inventory contains no further coordinate, trichotomy, host rounding, quotient-real data, and selected limits are outside the scoped surface. □

Theorem 433.9 (DyadicRounding mature finite-grid envelope)**.** *For every accepted* $\mathsf{DyadicRounding}^{\uparrow}$ *carrier* $D = (x, n, a, b, I, E, L, C, P, N)$*, the mature finite-grid envelope is exactly the carrier row of Definition 433.1, the finite grid sandwich of Definition 433.3, the ledger exactness row of Definition 433.4, the* $\mathsf{Real}^{\uparrow}$ *handoff of Definition 433.5, and the carrier scope of Definition 433.8. Every rounded read is therefore confined to the displayed source, precision, adjacent dyadic endpoints, containment, finite error budget, transport, continuation, provenance, and local naming rows.*

The envelope exports no quotient real, selected limit, trichotomy branch, host rounding operation, or additional paper-side theory obligation.

Proof. Start with the carrier row. The packet displays only $x, n, a, b, I, E, L, C, P, N$, so every accepted read begins inside the named finite inventory.

Apply the grid sandwich and ledger exactness theorems. They bind the analytic content of the rounded read to the adjacent dyadic endpoints, interval containment, and finite error budget at the requested precision.

Use the $\mathsf{Real}^{\uparrow}$ handoff theorem to route any real-facing consumer through the same finite enclosure before it sees the source read. The carrier scope theorem then rules out every absent analytic coordinate, so no additional paper-side theory row remains outside the finite-grid envelope. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicRounding}^{\uparrow}$ packet is generated from a displayed source read, unary precision request, adjacent dyadic endpoint rows, interval containment, finite error budget, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 433.1, Definition 433.2, Definition 433.3, Definition 433.4, Definition 433.5, Definition 433.8, and Definition 433.9 bind the source read, unary precision, dyadic endpoints, interval containment, finite error budget, BHist, hsame, Cont, Pkg, and NameCert rows of the displayed rounding packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicRounding}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicRoundingUp.DyadicRoundingPacket\_grid\_sandwich`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient reals, selected limits, trichotomy of host order, host rounding operations, or a public standard bridge.

Upgrade path: No paper-side theory upgrade remains; only future machine bridge checking can raise the verification bridge grade.

Chapter 434

A Concrete Naming Certificate for $\mathsf{SubshiftOfFiniteType}^{\uparrow}$

Following Lind and Marcus symbolic dynamics, the $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ interface is a BEDC finite certificate surface. It packages finite BHist words, forbidden-window ledgers, shift-window reads, and regular-language handoff rows, not an ambient shift space or an external symbolic-dynamics object.

The singleton forbidden-adjacency instance is consumed by chapter 370, so the finite forbidden-ledger carrier here is not isolated from the symbolic-dynamics branch.

434.1 Finite adjacency windows

Definition 434.1 (Subshift of finite type carrier)**.** *A* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, F, L, R, H, C, P, N)$$

with the following displayed coordinates:

1. *A is the finite visible alphabet row, specialized here to the displayed zero and one bit rows when the GoldenMean sibling is read;*
2. *W is a finite word row over A, recorded as a list of visible* BHist *letter rows;*
3. *F is the finite forbidden-window ledger whose rows are displayed words of one fixed positive length and whose binary adjacency specialization is the forbidden-pair ledger;*
4. *L is the admissibility ledger recording, for every displayed window of W, that the window is absent from F;*
5. *R is the shift-window row recording the left and right one-step windows of W and the finite* $\mathsf{StreamName}^{\uparrow}$ *observation through which those windows are read;*
6. *H records componentwise* hsame *transport for alphabet rows, word rows, forbidden-window rows, admissibility rows, shift-window rows, replay rows, provenance rows, and local naming rows;*
7. *C records displayed* Cont *replay from the finite word to its subwindows and from the subwindows to the admissibility ledger;*
8. *P records the finite prefix, restriction,* $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, *and row-transport provenance through* Cont *and* Pkg*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{SubshiftOfFiniteType}^{\uparrow}}$ *row for the componentwise visible-word classifier.*

The classifier compares two carriers componentwise by hsame *on the alphabet row, each visible word row, the forbidden-window ledger, the admissibility ledger, the shift-window row, the* Cont *replay row, the provenance row, and the local certificate row. Acceptance is exactly finite ledger inspection: every fixed-length window displayed by W must be rejected by no row of F and certified in L as allowed.*

Lean4 ✓ `BEDC.Derived.SubshiftOfFiniteTypeUp.SubshiftOfFiniteTypeTasteGate single carrier alignment`

Theorem 434.2 (Subshift of finite type NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SubshiftOfFiniteType}^{\uparrow}}$ *obligation surface consists of five finite obligations:*

1. *carrier habitation: the empty word and every word shorter than the displayed forbidden-window length are accepted, because they display no forbidden window;*
2. *classifier laws: reflexivity, symmetry, and transitivity are inherited componentwise from* hsame *on visible alphabet rows, word rows, forbidden-window rows, admissibility rows, shift-window rows, replay rows, provenance rows, and certificate rows;*
3. *forbidden-window transport: displayed* hsame, Cont, Pkg, *and* NameCert *routes transport accepted carriers exactly when transported finite windows remain outside the transported forbidden ledger;*
4. *finite-prefix restriction: every displayed prefix, suffix, shift-window, or finite restriction of W retains the corresponding subledger of F, L, R, C, P, N and remains accepted;*
5. *non-escape: every accepted row is justified only by the finite visible alphabet, word, forbidden-window ledger, admissibility ledger, shift-window row, replay row, provenance row, and local certificate rows.*

Thus the certificate is a BEDC-native finite-window object over BHist *rows.*

Proof. For habitation, choose the empty word or any displayed word shorter than the forbidden-window length. There is no full forbidden window to inspect, so the admissibility ledger is empty and no forbidden window can be witnessed.

For the classifier laws, read each coordinate of Definition 434.1 componentwise. The visible rows are compared by hsame, and the packet has no further coordinate on which another classifier could act.

For forbidden-window transport, move the alphabet, word, forbidden-window ledger, admissibility ledger, shift-window row, replay row, provenance row, and certificate row through the displayed hsame, Cont, Pkg, and NameCert routes. The transported carrier is accepted exactly when every transported window remains absent from the transported forbidden ledger, because acceptance is defined by that finite inspection.

For finite-prefix restriction, select an initial segment, suffix segment, or one-step shift window of the word through the recorded Cont route and restrict the ledgers to windows that still appear in the segment. A retained window was already accepted in the original word, so the restricted admissibility ledger certifies it again.

Non-escape follows by projection of the carrier coordinates. The packet exposes no entropy row, inverse-limit object, quotient coordinate, infinite sequence carrier, or host symbolic-dynamics theorem, so a downstream NameCert consumer can cite only the displayed finite ledger rows and their routes. □

Lean4 ✓ `BEDC.Derived.SubshiftfinitetypeUp.SubshiftFiniteTypeCarrier semantic name certificate`

Theorem 434.3 (Golden mean instance)**.** *The* $\mathsf{GoldenMeanShift}^{\uparrow}$ *finite-window carrier is the one-forbidden-word instance of* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *obtained by taking the visible alphabet row* $A = \{0, 1\}$ *and the forbidden-adjacency ledger*

$$F = \{11\}.$$

For this instance, a finite word is accepted precisely when every neighboring pair is zero-zero, zero-one, or one-zero, and the inherited NameCert *obligations are the GoldenMeanShift carrier habitation, local stability, fold-prefix closure, adjacency-ledger exhaustion, and non-escape obligations.*

Proof. Specialize Definition 434.1 to the binary alphabet row and the singleton forbidden ledger whose only row is the adjacent pair 11. A neighboring pair is accepted exactly when the finite ledger inspection does not find that row.

The remaining adjacent binary pairs are zero-zero, zero-one, and one-zero. These are exactly the accepted adjacency rows used by the $\mathsf{GoldenMeanShift}^{\uparrow}$ carrier, and the componentwise hsame classifier is the same visible-word classifier.

The NameCert obligations restrict along the same finite ledgers. Habitation is witnessed by the empty word, one-letter words, and the all-zero word; local transport preserves acceptance by transporting the singleton forbidden row; prefix restriction deletes neighboring checks without adding any; and non-escape is inherited from the absence of any coordinate beyond the finite BHist word and ledger rows.

This one-forbidden-word instance is the sibling dependency consumed by chapter 370: $\mathsf{GoldenMeanShift}^{\uparrow}$ reads the same binary alphabet row and singleton forbidden adjacency ledger rather than introducing a separate symbolic-dynamics carrier. □

Theorem 434.4 (Subshift finite type adjacency carrier transport)**.** *For accepted* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *carriers, componentwise* hsame *transport of the visible alphabet row, finite word row, forbidden-adjacency ledger, acceptance ledger,* Cont *routes,* Pkg *provenance, and local* NameCert *row preserves the finite adjacency carrier whenever the forbidden-pair ledger is transported with the same rows.*

Proof. Project the carrier of Definition 434.1. The carrier predicate is stated only in terms of the displayed alphabet, word, forbidden-window ledger, admissibility ledger, shift-window row, replay row, provenance row, and local certificate rows.

Transport the forbidden-pair ledger and the admissibility ledger along the same componentwise hsame routes. Every neighboring pair in the transported word is the length-two image of a window already inspected by the original finite ledger.

The Cont routes and Pkg provenance then repack the transported rows into the same carrier shape. Since no further coordinate participates in the predicate, the transported packet remains an accepted finite adjacency carrier. □

Theorem 434.5 (Subshift finite type forbidden-word ledger exhaustion)**.** *Every accepted neighboring-pair decision in a* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *carrier is read from the displayed allowed and forbidden adjacency ledgers, as the length-two specialization of the forbidden-window surface. No decision is supplied by entropy, inverse limits, quotient shift spaces, external imports, or an ambient symbolic-dynamics theorem.*

Proof. Take a neighboring-pair decision in an accepted word row. By the carrier definition, acceptance is recorded by the finite admissibility ledger and by absence from the finite forbidden-window ledger specialized to length two.

The provenance row records only Cont and Pkg routes between the displayed rows. These routes name how the finite rows are read; they do not introduce any entropy value, inverse-limit object, quotient carrier, or host symbolic-dynamics theorem.

Thus the decision is exhausted by the local adjacency ledgers. A downstream consumer can recover the decision by projecting the word, forbidden-ledger, admissibility-ledger, shift-window, replay, provenance, and local NameCert coordinates. □

Theorem 434.6 (Subshift finite type golden-mean scope)**.** *The singleton forbidden one-one row recovers the existing* $\mathsf{GoldenMeanShift}^{\uparrow}$ *instance as a scoped* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *packet: its accepted neighboring pairs are exactly zero-zero, zero-one, and one-zero, and its rejected neighboring pair is exactly one-one.*

Proof. Specialize the visible alphabet row to zero and one, and specialize the forbidden-adjacency ledger to the single row one-one. The acceptance ledger inspects each neighboring pair of the displayed finite word against that singleton row.

If the pair is zero-zero, zero-one, or one-zero, it is absent from the forbidden ledger and is accepted by the displayed finite decision row. If the pair is one-one, it is present in the forbidden ledger and is rejected.

This is exactly the finite-window $\mathsf{GoldenMeanShift}^{\uparrow}$ classifier used in Definition 434.3. The Cont routes, Pkg provenance, and local NameCert row repack the same scoped packet without adding an infinite shift carrier. □

Theorem 434.7 (Subshift finite type window-language route)**.** *Every accepted* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *carrier factors its admissible-word decision through the finite route*

$$\mathsf{BHist} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{FiniteAutomaton}^{\uparrow} \longrightarrow \mathsf{RegularLanguage}^{\uparrow}$$

before any symbolic-dynamics consumer can read the word as a language element. The BHist *row supplies the displayed finite word, the* $\mathsf{StreamName}^{\uparrow}$ *row supplies the inspected finite windows, the* $\mathsf{FiniteAutomaton}^{\uparrow}$ *row replays the forbidden-window scan as a finite-state run, and the* $\mathsf{RegularLanguage}^{\uparrow}$ *row records the accepted-language endpoint. No infinite shift space, quotient presentation, entropy coordinate, invariant measure, or ambient compactness theorem is inserted into this route.*

Proof. Project the accepted carrier to $A, W, F, L, R, H, C, P, N$. The rows W, F, L give the finite word, the forbidden-window ledger, and the admissibility ledger; R names the shift-window observations of the same finite word.

Read R through the $\mathsf{StreamName}^{\uparrow}$ finite-window surface. This only exposes the inspected windows as finite BHist observations, so the consumer still has no access to an infinite sequence or quotient shift space.

Replay the same window scan as a $\mathsf{FiniteAutomaton}^{\uparrow}$ run whose states remember the finite suffix needed to test the forbidden ledger. The accepting endpoint is then a $\mathsf{RegularLanguage}^{\uparrow}$ row for the language of finite words avoiding F.

Finally, H, C, P, N transport and name that route componentwise. Since every coordinate used by the consumer is one of these displayed rows, the accepted word decision is exhausted by the finite window-language route. □

Theorem 434.8 (Subshift finite type scoped obligation package)**.** *The carrier of Definition 434.1, the five* $\mathsf{NameCert}_{\mathsf{SubshiftOfFiniteType}^{\uparrow}}$ *obligations of Definition 434.2, adjacency carrier transport, forbidden-word ledger exhaustion, the finite window-language route, and the* $\mathsf{GoldenMeanShift}^{\uparrow}$

singleton-forbidden scope form one scoped package over the displayed alphabet row, finite word row, forbidden ledger, admissibility ledger, shift-window row, BHist, StreamName$^{\uparrow}$, FiniteAutomaton$^{\uparrow}$, RegularLanguage$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *The package exports no entropy coordinate, inverse-limit object, quotient shift-space, external import, or ambient symbolic-dynamics theorem.*

Proof. Project the carrier coordinates $A, W, F, L, R, H, C, P, N$. They supply the visible alphabet row, finite word row, forbidden-window ledger, admissibility ledger, shift-window row, transport, replay, provenance, and local naming row that support the scoped package.

Use the five local obligations of Definition 434.2. Habitation, classifier laws, forbidden-window transport, finite-prefix restriction, and non-escape are all stated over the displayed finite rows and their hsame, Cont, Pkg, and NameCert routes.

Next apply adjacency carrier transport, forbidden-word ledger exhaustion, and the finite window-language route from Definition 434.4, Definition 434.5, and Definition 434.7. These rows keep accepted packets within the same finite adjacency surface, identify every neighboring-pair decision with the displayed ledgers, and route finite words through StreamName$^{\uparrow}$, FiniteAutomaton$^{\uparrow}$, and RegularLanguage$^{\uparrow}$ before any language consumer reads them.

Finally specialize by Definition 434.6. The singleton forbidden one-one row gives the Golden-Mean finite-window scope inside the same packet, while the non-escape clauses rule out entropy, inverse limits, quotient shift spaces, external imports, and ambient symbolic-dynamics data. □

Closure status for NameCert$_{\text{SubshiftOfFiniteType}^{\uparrow}}$.

Constructive story: A SubshiftOfFiniteType$^{\uparrow}$ packet is generated from a visible alphabet row, a finite BHist word row, a finite forbidden-window ledger, an admissibility ledger for inspected windows, a shift-window row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SubshiftOfFiniteType}^{\uparrow}}$, obligation)

Scope closed: The obligation surface names the finite forbidden-window carrier, its five local NameCert obligations, adjacency-carrier transport, forbidden-ledger exhaustion, the finite window-language route, and the GoldenMeanShift$^{\uparrow}$ singleton-forbidden-ledger scope.

Formal verification: FormalStatus(NameCert$_{\text{SubshiftOfFiniteType}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.SubshiftfinitetypeUp.SubshiftFiniteTypeCarrier\_semantic\_name\_certificate`

Bridge status: none

Origin: human

Not claimed: This chapter does not close entropy, inverse limits, quotient presentations, infinite shift spaces, invariant measures, external imports, or a public standard bridge.

Upgrade path: Scoped closure requires complete Lean coverage of adjacency-carrier transport, forbidden-ledger exhaustion, the finite window-language route, and the singleton-forbidden-row GoldenMeanShift$^{\uparrow}$ scope.

UniformCauchyCriterion$^{\uparrow}$ reads its shared finite-family threshold route through chapter 126, chapter 301, chapter 291, and chapter 38 before the child files expose the Cauchy-limit and Real-completion consumers. The packet exports no quotient stream, selected limit, host equality, countable choice, or ambient completeness row.

Chapter 435

A Concrete Naming Certificate for UniformCauchyCriterion$^{\uparrow}$

The UniformCauchyCriterion$^{\uparrow}$ interface names the finite family-level Cauchy test used in the Bishop–Bridges constructive reading of uniform convergence. It records one displayed dyadic tail criterion shared by a finite family of StreamName$^{\uparrow}$ windows. The certificate is a consumer surface for RegSeqRat$^{\uparrow}$, CauchyModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, and Real$^{\uparrow}$, and it withholds quotient streams, host real equality, selected limits, and ambient completeness.

The sibling UniformLimit$^{\uparrow}$ dossier consumes this criterion through the shared-threshold tail ledger recorded in chapter 463: its endpoint route reads the same windowwise family data, then passes through the UniformCauchyCriterion$^{\uparrow}$ row before the RegSeqRat$^{\uparrow}$ handoff. Locally, the dependency route is grounded in StreamName$^{\uparrow}$ finite windows and RegSeqRat$^{\uparrow}$ regular-sequence readback, so the criterion is a concrete-instance bridge between source windows and the uniform-limit consumer rather than a kernel-only tail predicate.

Definition 435.1 (UniformCauchyCriterion carrier)**.** *A* UniformCauchyCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite family index ledger naming the members whose windows are compared;*
2. *W is the shared finite* StreamName$^{\uparrow}$ *window family, indexed by I, through which every source observation is read;*
3. *M is one* CauchyModulus$^{\uparrow}$ *schedule used uniformly for all indices in I;*
4. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row whose dyadic budget is used by the family test;*
5. *T is the finite* DyadicTailBound$^{\uparrow}$ *tail-comparison ledger recording all accepted comparisons beyond the threshold selected by M and inside the windows named by W;*
6. *E is the* Real$^{\uparrow}$ *seal handoff row reached after the finite family tail criterion has been accepted;*
7. *H records the componentwise* hsame *rows for the family ledger, window family, modulus schedule, tolerance row, tail ledger, and seal handoff;*
8. *C records the displayed* Cont *routes by which a consumer reads the family index row, shared windows, uniform modulus, tolerance, finite tail ledger, and real seal;*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}$ *provenance row.*

The classifier compares two accepted carriers only through these displayed coordinates, their componentwise transports, their continuation routes, and their package and local naming provenance.

Lean4 (definition) `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionCarrier`

This carrier consumes the finite $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-tail readback surface of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. These sibling dependencies are exactly the horizontal lattice edges used by the shared-window, tail-ledger, and seal-handoff obligations below: the criterion is a finite family tail service between source windows and Real-facing consumers, not a kernel-only Cauchy predicate.

Theorem 435.2 (UniformCauchyCriterion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}$ *surface for the carrier of Definition 435.1 consists exactly of five rows:*

1. *carrier habitation by the displayed packet $U = (I, W, M, Q, T, E, H, C, P, N)$;*
2. *classifier transport by componentwise* hsame *and displayed* Cont *routes over the finite family ledger, shared window family, modulus schedule, tolerance row, tail ledger, seal handoff, package provenance, and local naming row;*
3. *uniform window stability, requiring every transported precision request to keep the same I-indexed* $\mathsf{StreamName}^{\uparrow}$ *window family and the same shared modulus row M before tail comparisons are read;*
4. $\mathsf{Real}^{\uparrow}$ *seal handoff, requiring the family tail ledger to pass through T and the real-facing row E after the shared threshold and dyadic tolerance have been displayed;*
5. *non-escape from quotient streams, host real equality, selected limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket namecert obligations`

Proof. Carrier habitation is projection from the finite packet in Definition 435.1. The family index ledger, window family, shared modulus, dyadic tolerance, finite tail ledger, real seal, transport rows, continuation routes, package row, and local naming row are all visible coordinates.

Classifier transport is componentwise. The hsame rows in H move the family ledger, windows, modulus, tolerance, tail ledger, and seal row, while the Cont rows in C replay the same read order for a consumer.

The uniform window row is the key obligation: every member named by I is inspected through the same W family and the same threshold schedule M. The seal handoff then reads Q and T before reaching E, and no remaining coordinate can export quotient streams, host real equality, selected limits, or ambient completeness. □

Theorem 435.3 (UniformCauchyCriterion window stability)**.** *Let $U = (I, W, M, Q, T, E, H, C, P, N)$ and $U' = (I', W', M', Q', T', E', H', C', P', N')$ be accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *carriers related by the displayed classifier. Every transported family precision request preserves the same finite index ledger, the same* $\mathsf{StreamName}^{\uparrow}$ *window family up to the componentwise* hsame *rows, and the same shared* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule up to the transported row. Thus every tail comparison in the transported carrier is still read through one uniform threshold and one displayed dyadic tolerance row for all family members.*

Proof. Project both carriers through Definition 435.1. The classifier has access only to $I, W, M, Q, T, E, H, C, P, N$ and to the corresponding rows of U'.

Transport the family index and window rows first. The rows in H compare the finite family ledger and the window family componentwise, and the routes in C describe how the consumer reaches those rows before any tail ledger is inspected.

Now transport the shared modulus row. Since M is a single coordinate of the packet rather than an index-dependent family of moduli, the transported request still reaches one threshold schedule before reading Q and T.

The tail ledger can therefore record only comparisons inside the transported window family and beyond the transported threshold. Repacking these projections gives the stated window stability and leaves no coordinate for a host-level pointwise criterion. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket window stability`

Theorem 435.4 (UniformCauchyCriterion RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The route first fixes the finite family ledger, then reads the shared $\mathsf{StreamName}^{\uparrow}$ *windows, then applies the single* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, then reads the dyadic tolerance row and the finite tail-comparison ledger before reaching the seal row. The seal exposes no completed family limit, no quotient of streams, no host real equality, and no ambient completeness principle.*

Proof. Start with a real-facing consumer request. By carrier projection, the request must first identify the finite family ledger I and the shared window family W that supplies the inspected observations.

Apply Definition 435.3. The request reaches the shared modulus row M before any comparison in T can be used, so the threshold is uniform over the displayed family.

The tolerance coordinate Q and the finite ledger T are then read in the same packet. These rows provide the accepted dyadic tail comparison, while H, C, P, N keep the transport, continuation, package, and local naming provenance visible.

Finally the seal coordinate E is reached only after that route has been completed. Since none of the carrier coordinates is a completed family limit, quotient stream, host real equality, or completeness theorem, the exported seal read cannot bypass the displayed finite criterion. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real seal handoff`

Theorem 435.5 (UniformCauchyCriterion shared-window obligation)**.** *Let I be a finite family ledger, W a* $\mathsf{StreamName}^{\uparrow}$ *window family over I, M a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, Q a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, T a* $\mathsf{DyadicTailBound}^{\uparrow}$ *tail ledger, E a* $\mathsf{Real}^{\uparrow}$ *seal row, and H, C, P, N the displayed* hsame, Cont, Pkg, *and local* NameCert *rows of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every family tail comparison admitted by this packet first fixes I, W, M, Q and then records its comparison in T; no comparison may be read through a quotient stream family, host real equality, selected limit, or ambient completeness object.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared window obligation`

Proof. Use Definition 435.3 to fix the finite family ledger and shared window family before a tail row is inspected.

Then apply Definition 435.9. It places every accepted comparison after the single threshold schedule and dyadic tolerance row.

Finally use Definition 435.8. The excluded objects are not rows of the packet, so the shared-window obligation is exhausted by $I, W, M, Q, T, H, C, P, N$. □

Theorem 435.6 (UniformCauchyCriterion seal-route obligation)**.** *For accepted rows $I, W, M, Q, T, E, H, C, P, N$ of a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$ *seal read follows the displayed finite route from I through W, M, Q, T to E. The route may transport and replay through H, C, P, N, but it has no coordinate for quotient streams, host real equality, selected limits, or ambient completeness.*

Proof. Begin with Definition 435.4. It identifies the route to E as the finite family ledger, shared windows, threshold schedule, dyadic tolerance, and tail ledger.

The structural rows H, C, P, N only transport, replay, package, and name that same route. By Definition 435.72, there is no second seal path inside the accepted packet. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket seal route obligation`

Theorem 435.7 (UniformCauchyCriterion non-escape obligation)**.** *The current-phase* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *obligation surface for rows $I, W, M, Q, T, E, H, C, P, N$ is exactly the shared-window tail route and the seal route displayed above. A consumer can inspect finite family comparisons and seal handoff through those rows only; it cannot obtain quotient streams, host real equality, selected limits, or ambient completeness from the criterion.*

Proof. Combine Definition 435.5 with Definition 435.6. The first theorem exhausts family tail comparisons, and the second exhausts real-seal reads.

Apply Definition 435.8 to the combined surface. Since all consumer reads have already been routed through $I, W, M, Q, T, E, H, C, P, N$, the excluded analytic objects remain outside the packet. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket nonescape obligation`

Theorem 435.8 (UniformCauchyCriterion non-escape boundary)**.** *An accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *carrier exports only the finite family index ledger, the shared* $\mathsf{StreamName}^{\uparrow}$ *windows, the uniform* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the finite* $\mathsf{DyadicTailBound}^{\uparrow}$ *comparison ledger, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows. It exports no quotient stream family, no host real equality, no selected family of limits, and no ambient completeness object.*

Proof. Inspect the carrier coordinates of Definition 435.1. Each row is a finite ledger, a displayed dependency certificate, a transport row, a continuation route, package provenance, or local naming provenance.

The window-stability theorem keeps every transported read inside the finite I, W, M, Q, T surface. The real-seal handoff theorem then forces every $\mathsf{Real}^{\uparrow}$ read to pass through that same surface before reaching E.

No excluded analytic object is a coordinate of the packet or a result of the displayed route. Therefore a downstream Weierstrass, completion, or real-seal chapter can consume the uniform Cauchy criterion only as the finite NameCert package described here. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket non escape boundary`

Theorem 435.9 (UniformCauchyCriterion tail-ledger exactness)**.** *Every accepted finite-family tail comparison in a* UniformCauchyCriterion$^{\uparrow}$ *carrier factors through the displayed packet*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

of Definition 435.1 and through one shared dyadic tolerance ledger. More precisely, the comparison first reads the finite family ledger I, the shared StreamName$^{\uparrow}$ *window family W, the single* CauchyModulus$^{\uparrow}$ *threshold schedule M, and the* DyadicRatCore$^{\uparrow}$ *tolerance row Q; it then records the accepted tail comparison in T before any* Real$^{\uparrow}$ *seal row E can be consumed. The transport rows H, read routes C, package row P, and local naming row N are the only provenance by which the comparison is replayed.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail ledger exactness`

Proof. Start with an accepted tail comparison. By Definition 435.2, the comparison is read from the carrier of Definition 435.1, so its available rows are exactly $I, W, M, Q, T, E, H, C, P, N$.

Apply Definition 435.3. The transported request keeps the same finite family ledger, the same shared window family, and one shared threshold schedule before the tail ledger is inspected.

The dyadic tolerance row Q and the comparison ledger T therefore form one displayed tolerance route for all accepted family comparisons. Finally Definition 435.4 places any real seal read after this route, and Definition 435.8 blocks any hidden completion or host equality escape. □

Theorem 435.10 (UniformCauchyCriterion shared-threshold transport)**.** *Let $U = (I, W, M, Q, T, E, H, C, P, N)$ and $U' = (I', W', M', Q', T', E', H', C', P', N')$ be accepted* UniformCauchyCriterion$^{\uparrow}$ *carriers related by the displayed classifier. Classifier transport preserves the single shared threshold schedule: every transported family-index read reaches M' as the transported image of the one schedule M, not as a family of independently selected indexwise schedules. Consequently all transported tail comparisons for indices in I' use the same threshold route before reading Q' and T'.*

Proof. Project the two carriers through Definition 435.1. The classifier can compare only the displayed coordinates and their recorded hsame, Cont, Pkg, and NameCert provenance.

Use the classifier-transport row of Definition 435.2. It transports the family ledger and window family componentwise, while the modulus coordinate is one row of the packet.

Now apply Definition 435.3. Since there is only one modulus coordinate before the tolerance and tail rows, the transported carrier can supply only the transported shared schedule M' for all family indices. The tail-ledger exactness route then forces Q' and T' to be read after that same schedule. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared threshold transport`

Theorem 435.11 (UniformCauchyCriterion RegSeqRat tail equivalence)**.** *For every accepted* UniformCauchyCriterion$^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

the finite tail-ledger route through the shared StreamName$^{\uparrow}$ *window family W, the single* CauchyModulus$^{\uparrow}$ *schedule M, the* DyadicRatCore$^{\uparrow}$ *tolerance row Q, and the* DyadicTailBound$^{\uparrow}$ *comparison ledger T is equivalent, as a BEDC consumer surface, to the* RegSeqRat$^{\uparrow}$ *finite-window regularity route whose* Real$^{\uparrow}$ *seal is E. Thus a consumer can reconstruct the same accepted finite evidence from either surface, and neither route introduces a quotient stream, selected limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.RegSeqRatTailEquivalence.UniformCauchyCriterionPacket regseqrat tail equivalence`

Proof. For the forward direction, begin with a consumer read of the tail ledger. By Definition 435.9, the read passes through I, W, M, Q, T before any seal row is consumed. By Definition 435.10, the threshold row remains one shared M-schedule across all transported family indices.

Now attach the regular-rational handoff. The carrier already stores the seal row E, and Definition 435.4 forces the real-facing read to pass through the same finite route before reaching E. The $\mathsf{RegSeqRat}^{\uparrow}$ regularity window therefore receives exactly the stream windows, shared modulus, dyadic tolerance, and accepted tail comparisons displayed in the packet.

For the reverse direction, start with the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window regularity route whose seal is E. Unpack its window read through the same carrier coordinates: the stream observations are the family windows W, the threshold schedule is the displayed modulus row M, and the dyadic tolerance budget is Q.

Repack those rows into the comparison ledger T. The transport and read routes H and C preserve the displayed coordinates, while P and N keep the package and local naming provenance attached. This reconstructs the finite $\mathsf{DyadicTailBound}^{\uparrow}$ ledger without adding any excluded completion, limit-selection, quotient, or host-equality object. □

Theorem 435.12 (UniformCauchyCriterion finite-tail-filter Real-window handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any finite-family tail route of* U *consumed by a* $\mathsf{RealWindowBudget}^{\uparrow}$ *or Real-completion consumer must factor through the displayed* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet of chapter 1119 before the Real-window selector reads the route.*

Concretely, the consumer first reads the shared $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the uniform modulus row* M*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*, and the tail ledger* T *as the UniformCauchyCriterion tail surface; the same read is then repacked as the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *cofinal-tail packet carrying the displayed stream-window, dyadic,* $\mathsf{RegSeqRat}^{\uparrow}$*, tail-budget,* $\mathsf{CauchyTailComparison}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; only after that packet is displayed may the* $\mathsf{RealWindowBudget}^{\uparrow}$ *selector consume the Real-facing route.*

This handoff does not assert quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, or any standard bridge row.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite tail filter real window handoff`

Proof. Start with tail-ledger exactness, Definition 435.9. It places every accepted finite-family tail comparison of U after the displayed index, window, modulus, dyadic tolerance, and tail-ledger rows I, W, M, Q, T.

Next attach the UniformCauchyCriterion Real seal route, Definition 435.4. The Real-facing read reaches E only after the same finite tail surface has been consumed, with H, C, P, N replaying transport, continuation, provenance, and local naming.

Now pass to the FiniteTailFilter packet. The finite-tail-filter Real-window consumer theorem, Definition 1119.6, reads the displayed cofinal-tail packet through its stream-window, dyadic, regular-sequence, tail-budget, common-tail, Real seal, transport, continuation, package, and local naming rows.

Finally apply the Real-window selector bridge, Definition 1119.8. It lets the $\mathsf{RealWindowBudget}^{\uparrow}$ selector read only after the $\mathsf{FiniteTailFilter}^{\uparrow}$ route is displayed, and it refuses a private selector,

detached dyadic ledger, selected limit, quotient stream, host real equality, countable-choice row, ambient completeness coordinate, or bridge row. Hence every Real-window or completion-facing consumer of U factors through the finite-tail-filter packet before the selector reads the route. □

Theorem 435.13 (UniformCauchyCriterion Real-completion window exhaustion)**.** *For every accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer route from finite family observations into a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then into the* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly the finite observation rows* I, W, M, Q, T, E*, with structural replay through* H, C, P, N*. Conversely, this displayed route supplies the full admissible finite evidence consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *handoff. No such route may pass through quotient streams, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion window exhaustion`

Proof. Start from the finite family observations. By Definition 435.3, any accepted transported read remains inside the shared $\mathsf{StreamName}^{\uparrow}$ window family W. By Definition 435.10, the same read uses the single modulus schedule M, not an indexwise choice of thresholds.

The tail side is then forced. Tail-ledger exactness sends each accepted comparison through the dyadic tolerance row Q and the finite comparison ledger T. The RegSeqRat tail equivalence theorem repackages exactly this finite route as the $\mathsf{RegSeqRat}^{\uparrow}$ regularity readback whose seal row is E.

Now attach the seal. The $\mathsf{Real}^{\uparrow}$ seal handoff consumes the displayed finite rows before any real-facing boundary is reached, while H, C, P, N replay the classifier transport, continuation, package provenance, and local naming structure already present in the accepted packet.

For the converse direction, begin with the displayed route $I, W, M, Q, T, E, H, C, P, N$. The same cited window, threshold, tail-ledger, RegSeqRat, and seal results reconstruct the admissible finite evidence in the order required by the $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ handoff. The non-escape boundary excludes quotient streams, selected limits, host real equality, and ambient completeness, so no additional completion object can be part of the handoff route. □

Theorem 435.14 (UniformCauchyCriterion obligation assembly)**.** *The carrier of Definition 435.1, Definition 435.2, Definition 435.3, Definition 435.9, Definition 435.10, Definition 435.4, and Definition 435.8 assemble the full* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *obligation package. The package consists of carrier habitation, classifier transport, shared-window stability, exact finite tail ledger readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and non-escape from quotient streams, host real equality, selected family limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket obligation assembly`

Proof. First bind carrier habitation. The finite packet $U = (I, W, M, Q, T, E, H, C, P, N)$ supplies the family ledger, shared windows, uniform modulus, dyadic tolerance, finite tail ledger, real seal, transport rows, read routes, package provenance, and local naming row.

Next bind transport and stability. The NameCert obligation theorem gives componentwise classifier transport, and the window-stability theorem keeps every transported family request inside the same window family and shared modulus route.

Now bind the tail and threshold rows. Tail-ledger exactness forces every accepted comparison through I, W, M, Q, T, while shared-threshold transport preserves the single M schedule across all transported family indices.

Finally attach the seal and boundary rows. The real-seal handoff routes each $\mathsf{Real}^{\uparrow}$ read through the same finite path, and the non-escape theorem states that no coordinate exports quotient streams, host real equality, selected limits, or ambient completeness. These rows are exactly the displayed obligation package. □

Theorem 435.15 (UniformCauchyCriterion subfamily restriction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose J is a displayed finite subfamily ledger whose every read factors through the index ledger I by visible* hsame *comparisons and* Cont *rows. Then the restricted packet $U|_J$ is again an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *carrier over the same shared schedule M, tolerance row Q, tail ledger route, real seal row E, package row P, and local* $\mathsf{NameCert}$ *row N.*

The restricted carrier reads only the displayed subfamily windows selected by J and inherits the hsame *and* Cont *rows already present in H and C. It does not introduce a quotient stream, selected limit, host equality row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket subfamily restriction`

Proof. First apply Definition 435.3. Each transported request from J factors through I, so the restricted window read stays inside the same finite window family W and uses the same shared schedule M.

Next apply Definition 435.9. The tail evidence read by the subfamily is the displayed subledger of T reached through I, W, M, Q, so no separate tolerance row or tail route is created.

The shared threshold is preserved by Definition 435.10. Since every J-read reaches the original packet through visible hsame and Cont rows, the restricted packet keeps the single schedule M rather than indexwise schedules.

Finally use Definition 435.4. The real-facing row remains E, and the package and local naming provenance are the inherited rows P and N. The construction has no carrier coordinate for quotient streams, selected limits, host equality, or ambient completeness, so none is added by the restriction. □

Theorem 435.16 (UniformCauchyCriterion RegSeqRat consumer package)**.** *For every accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

each $\mathsf{RegSeqRat}^{\uparrow}$ *consumer route factors through the finite family index ledger I, the shared* $\mathsf{StreamName}^{\uparrow}$ *window family W, the single* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule M, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row Q, the finite* $\mathsf{DyadicTailBound}^{\uparrow}$ *comparison ledger T, the* $\mathsf{Real}^{\uparrow}$ *seal E, the transport row H, the continuation route C, the package provenance P, and the local* $\mathsf{NameCert}$ *row N.*

Proof. First use Definition 435.14. It binds the carrier, local classifier transport, shared-window stability, finite tail ledger, real seal, and non-escape boundary as one obligation package.

Next apply Definition 435.11. The regular-rational read is the same finite evidence as the tail-ledger route, so the consumer must pass through I, W, M, Q, T before reading the real seal.

Finally use Definition 435.10 and Definition 435.4. The shared threshold and seal rows keep the consumer path inside H, C, P, N and expose no additional completion object. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseqrat consumer package`

Theorem 435.17 (UniformCauchyCriterion finite-family window exhaustion)**.** *Every transported request made to an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet uses the same finite index ledger, the same* $\mathsf{StreamName}^{\uparrow}$ *window family, the same shared modulus schedule, the same dyadic tolerance row, and the same finite tail ledger. Transport may rename the displayed rows through* hsame *and read them through* Cont*, but it cannot split the family into indexwise schedules or add a selected limit for any member of the family.*

Proof. Start from the NameCert obligations in Definition 435.2. They make classifier transport componentwise over the displayed packet coordinates.

Use Definition 435.3 to keep each transported request inside the same window family W. Then use Definition 435.10 to keep the threshold schedule as the one shared row M.

It remains to account for tolerance and tails. By Definition 435.9, every accepted comparison reads through Q and T after the shared window and modulus coordinates. The non-escape theorem then rules out indexwise schedules and selected limits because those objects have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family window exhaustion`

Theorem 435.18 (UniformCauchyCriterion Real seal completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet reaches the seal only after the finite family route through* $I, W, M, Q, T, E, H, C, P, N$*. The route supplies a finite uniform Cauchy certificate for the named family windows, but it does not supply quotient stream families, host real equality, selected family limits, or ambient completeness.*

Proof. First apply Definition 435.4. It sends every Real-facing read through the finite index ledger, shared windows, modulus, dyadic tolerance, and tail comparison ledger before the seal row E is consumed.

Next combine Definition 435.16 with Definition 435.17. These package the regular-rational and transported family reads over the same finite rows.

Finally use Definition 435.8 and Definition 435.14. The completion consumer receives the displayed finite certificate surface and no quotient stream family, host equality, selected limit, or completeness principle. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real seal completion boundary`

Definition 435.19 (UniformCauchyCriterion public export surface)**.** *The public export surface of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet is the finite row package*

$$\mathsf{Pub}(U) = (I, W, M, Q, T, E, H, C, P, N)$$

read from the carrier of Definition 435.1. It exposes the finite family index ledger I*, shared* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, single* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule* M*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*, finite* $\mathsf{DyadicTailBound}^{\uparrow}$ *tail ledger* T*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, transport rows* H*, continuation routes* C*, package provenance* P*, and local naming row* N*.*

The surface is public only as this finite row package. It contains no quotient stream family, no host real equality, no selected family limits, and no ambient completeness object.

Lean4 (definition) `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPublicExportSurface`

Theorem 435.20 (UniformCauchyCriterion public export package)**.** *Every accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet exports the public surface of Definition 435.19. The export is assembled from the carrier, NameCert obligations, window stability, tail-ledger exactness, shared-threshold transport, RegSeqRat consumer package, finite-family window exhaustion, and the* $\mathsf{Real}^{\uparrow}$ *seal completion boundary.*

Proof. First project the carrier of Definition 435.1. This supplies exactly the rows $I, W, M, Q, T, E, H, C, P, N$ named by the public export surface.

Next attach the local certificate obligations. The NameCert obligation theorem gives carrier habitation and classifier transport, while window stability and shared-threshold transport keep every transported family request inside the same shared window and modulus rows.

Now attach the tail and consumer rows. Tail-ledger exactness routes every accepted comparison through I, W, M, Q, T, the RegSeqRat consumer package reads the same finite evidence, and finite-family window exhaustion prevents indexwise schedules or selected limits from entering the route.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal completion boundary. The seal is reached only after the finite public rows have been displayed, and the non-escape clauses exclude quotient stream families, host real equality, selected family limits, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPublicExportSurface package closure`

Theorem 435.21 (UniformCauchyCriterion downstream Real consumer export)**.** *Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet can read only the public export surface* $\mathsf{Pub}(U) = (I, W, M, Q, T, E, H, C, P, N)$. *The consumer route factors through the shared threshold, finite tail ledger, RegSeqRat consumer package, finite-family window exhaustion, and the* $\mathsf{Real}^{\uparrow}$ *seal completion boundary before any real-facing row is available.*

Proof. Start with a downstream consumer read. By Definition 435.16, a regular-rational consumer must pass through the family ledger, shared windows, shared modulus, dyadic tolerance, tail ledger, real seal, transport rows, continuation routes, package row, and local naming row.

The same route is preserved for real-facing consumers. Shared-threshold transport keeps the modulus coordinate single, and finite-family window exhaustion keeps every request inside the displayed window family.

The completion-facing step is then the seal boundary of Definition 435.18. Since the boundary exposes only the finite public surface before E is read, the downstream consumer receives no quotient streams, host real equality, selected family limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket downstream real consumer export`

Theorem 435.22 (UniformCauchyCriterion finite-family Real seal route determinacy)**.** *Let* $U = (I, W, M, Q, T, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The Real-facing seal route of* U *is uniquely determined by the finite family ledger* I*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window family* W*, the single modulus schedule* M*, the dyadic tolerance row* Q*, and the tail ledger* T*.*

In particular, any downstream consumer that reaches the seal row E *through the same displayed rows* I, W, M, Q, T *reaches the same finite BEDC route through* E, H, C, P, N*. The route has no quotient stream family, host real equality, selected family limit, ambient completeness object, or choice schedule.*

Proof. Begin with the carrier of Definition 435.1. It displays exactly the finite packet rows $I, W, M, Q, T, E, H, C, P, N$, so the family ledger, shared windows, modulus schedule, dyadic tolerance, and tail ledger are public route coordinates rather than external analytic data.

Apply Definition 435.3. The shared StreamName windows remain the same family under the allowed transports. Then use Definition 435.4 to route every Real-facing read through the displayed rows before the seal row E is consumed.

Next apply Definition 435.9. Once I, W, M, Q, T are fixed, every accepted tail comparison reads through the same finite tolerance and tail ledger, so a consumer reaching E through those rows has not introduced a second route.

Finally use Definition 435.8. Since quotient stream families, host real equality, selected family limits, ambient completeness, and choice schedules are not packet coordinates, they cannot distinguish two consumers that display the same I, W, M, Q, T rows and the same seal handoff through E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family real seal route determinacy`

Theorem 435.23 (UniformCauchyCriterion endpoint budget reflector)**.** *Let $U = (I, W, M, Q, T, E, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any downstream endpoint-budget request on the Real-facing route reflects back to the same finite family ledger I, shared window family W, modulus schedule M, dyadic tolerance row Q, and tail ledger T before the seal row E is read. The reflected endpoint budget has no quotient stream, selected family limit, host real equality, ambient completeness object, or choice schedule.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket endpoint budget reflector`

↪ variant `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family endpoint budget reflector`

Proof. Begin with the finite-family Real seal route determinacy theorem, Definition 435.22. It identifies the route to the seal from the displayed rows I, W, M, Q, T and no external analytic coordinate.

Now use tail-ledger exactness, Definition 435.9. Any endpoint-budget request that reaches the Real-facing seal must first pass through the same tail comparison ledger and dyadic tolerance row.

Finally apply the public non-escape export, Definition 435.24. Since quotient streams, selected family limits, host real equality, ambient completeness, and choice schedules are absent from the public packet, the reflected endpoint budget can only be the finite I, W, M, Q, T readback before E. □

Theorem 435.24 (UniformCauchyCriterion public non-escape export)**.** *The public export surface of* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *is finite and non-escaping: it exports exactly $I, W, M, Q, T, E, H, C, P, N$ and excludes quotient stream families, host real equality, selected family limits, and ambient completeness. Any downstream consumer that claims one of the excluded objects is outside the public package.*

Proof. Inspect the definition of the public export surface. It contains only the ten displayed rows copied from the accepted carrier.

Use the NameCert obligations and the non-escape boundary to rule out extra coordinates. The certificate admits classifier transport, window stability, tail-ledger exactness, and real-seal handoff only through the rows already present in the carrier.

The downstream Real consumer export does not add a new object. It merely replays the RegSeqRat consumer package, finite-family window exhaustion, shared-threshold transport, and Real seal completion boundary. Therefore the public surface remains finite and cannot export quotient streams, host real equality, selected family limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket public non escape export`

Theorem 435.25 (UniformCauchyCriterion L10 exit public non-escape)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The L10 exit consumer read of* U *is public exactly through the finite export surface*

$$\mathsf{Pub}(U) = (I, W, M, Q, T, E, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *tail-equivalence route, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and, when the finite-observation comparison is displayed, the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *diagonal-budget pullback route.*

Equivalently, preserving these rows and routes is sufficient for the L10 exit consumer read; conversely every L10 exit consumer read factors back through these rows before any completion-facing endpoint is available. The read therefore exports no quotient stream family, selected family limit, countable choice schedule, host real equality, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket l10 exit public nonescape`

Proof. For sufficiency, first read the public surface through Definition 435.19 and Definition 435.20. These identify the accepted packet with the displayed finite rows $I, W, M, Q, T, E, H, C, P, N$ and place all carrier, NameCert, stability, tail-ledger, and seal data on that surface.

Next attach the exit routes. The tail read is the $\mathsf{RegSeqRat}^{\uparrow}$ route of Definition 435.11, while the completion-facing endpoint reaches the seal only along the displayed public rows. If a finite-observation selector is displayed, the pullback theorem Definition 435.26 places the same W, Q, T, E read inside that finite selector surface.

For necessity, start with any L10 exit consumer read. Apply Definition 435.24. Since the public export is exactly $\mathsf{Pub}(U)$ and has no excluded completion or host coordinate, the consumer must factor through $I, W, M, Q, T, E, H, C, P, N$ before any endpoint read can be interpreted.

Finally use Definition 435.76. The late-boundary route exhausts the no-choice completion side by the same public rows together with the displayed tail-equivalence and finite-budget routes. Thus preserving those rows is exactly the public L10 exit condition, and every such consumer read factors back through them. □

Theorem 435.26 (UniformCauchyCriterion finite-observation budget pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let*

$$B = (B_s, S, W_B, D, R, E_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet whose displayed window, dyadic ledger, regular readback, and seal rows agree componentwise with* W, Q, T, E *through the carried* hsame *and*

Cont *routes. Then every* UniformCauchyCriterion$^{\uparrow}$ *Real-facing or completion-facing consumer pulls back through the finite observation budget selector surface before E is read.*

The pullback route is the public non-escaping route of Definition 435.24, the finite-family Real seal route of Definition 435.22, and the shared selector route of Definition 782.20 inside chapter 782. Hence the consumer cannot read a quotient stream, selected host limit, choice schedule, host equality, or ambient completeness object from either packet.

Proof. Project the accepted packet U. The public non-escape export exposes only $I, W, M, Q, T, E, H, C, P, N$, and the finite-family Real seal route determinacy theorem locks the Real-facing route to I, W, M, Q, T before the seal row E is consumed.

Project the accepted selector packet B. By hypothesis, the selector rows W_B, D, R, E_B agree componentwise with the UniformCauchyCriterion rows W, Q, T, E through the carried hsame and Cont routes. Thus the displayed window, dyadic budget, regular readback, and seal rows are the same consumer surface up to the recorded BEDC transports.

Now attach the finite-observation selector pullback from Definition 782.20. It places the shared W, Q, T, E route inside the selector surface before any Real-facing endpoint can be read.

Finally apply non-escape on both packets. The UniformCauchyCriterion public export and the finite selector route exclude quotient streams, selected host limits, choice schedules, host equality, and ambient completeness. Therefore every Real-facing or completion-facing consumer of U must pass through the finite observation budget selector surface before E is read. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite observation budget pullback`

Theorem 435.27 (UniformCauchyCriterion finite-tail-filter alignment)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet. The shared-window route through $I, W, M, Q, T, E, H, C, P, N$ determines a* FiniteTailFilter$^{\uparrow}$ *cofinal-tail packet exactly when the displayed tail ledger factors through the same* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance row,* RegSeqRat$^{\uparrow}$ *readback,* DyadicTailBound$^{\uparrow}$ *budget,* CauchyTailComparison$^{\uparrow}$ *row, and* Real$^{\uparrow}$ *seal.*

Equivalently, a Real-facing or Completion-facing consumer of U is a finite-tail-filter consumer precisely when its route reads the common windows and dyadic tolerance from W and Q, reads the regular sequence tail from T, reads the finite tail budget and comparison row through the tail ledger certified by T, and reaches the seal row E only through the displayed structural rows H, C, P, N. Under this alignment the consumer receives no quotient stream family, host real equality, selected family limit, ambient completeness object, or choice schedule.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite tail filter alignment`

Proof. Start with tail-ledger exactness, Definition 435.9. It places every accepted tail comparison of U after the same shared StreamName$^{\uparrow}$ windows and DyadicRatCore$^{\uparrow}$ tolerance row, so the candidate finite-tail-filter route has no second window family or tolerance ledger.

Use the RegSeqRat tail equivalence of Definition 435.11. The regular-sequence readback carried by T is therefore the same readback used by the shared-window tail comparison. The finite-tail-filter tail-window exactness theorem, Definition 1119.4, then identifies the cofinal-tail packet rows with the displayed W, Q, T route.

Next apply the Real seal and completion boundary, Definition 435.18. The seal-facing part of the UniformCauchyCriterion route reaches E through the public rows H, C, P, N and does not expose a quotient stream, selected limit, host equality, ambient completeness object, or choice schedule.

Finally use the finite-tail-filter Real seal handoff, Definition 1119.5, and sibling tail handoff, Definition 1119.9. The first carries the same cofinal-tail packet to Real and Completion consumers; the second prevents a sibling consumer from substituting another tail selector, comparison row, or seal route. Hence the shared-window route of U determines exactly the finite-tail-filter cofinal-tail packet described above, and the reverse implication is obtained by reading the same packet rows back through W, Q, T, E, H, C, P, N. □

Theorem 435.28 (UniformCauchyCriterion finite-window Real-seal extraction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Its* $\mathsf{Real}^{\uparrow}$*-seal route is extracted from exactly the finite window family* W*, the dyadic tolerance row* Q*, the tail ledger* T*, and the structural rows* H, C, P, N*. The extraction exposes no quotient stream family, host real equality, selected family limit, choice schedule, or ambient completeness.*

Proof. Start with Definition 435.27. It identifies the Real-facing route with the shared finite-window route through W, Q, T, E, H, C, P, N and excludes any second window family or tail selector.

Apply Definition 435.18. The seal row E is read only after the public rows H, C, P, N have carried the finite tail and tolerance data to the Real-facing boundary.

Finally use Definition 435.24. Since the public route is already exhausted by $I, W, M, Q, T, E, H, C, P, N$, the extracted Real-seal route has no coordinate for quotient stream families, host real equality, selected family limits, choice schedules, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionFiniteWindowRealSealExtraction`

Theorem 435.29 (UniformCauchyCriterion finite-window selector uniqueness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

and

$$U' = (I', W', M', Q', T', E', H', C', P', N')$$

be accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packets. Suppose their finite family ledgers,* $\mathsf{StreamName}^{\uparrow}$ *window families, shared modulus rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows, and structural* hsame, Cont, Pkg, *and* NameCert *rows agree componentwise along the displayed BEDC transports.*

Then every $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of* U *reads the same finite-window selector surface as the corresponding consumer of* U' *before reaching the seal. The common surface is exactly the route through the finite family ledger,* $W, M, Q, T, E, H, C, P, N$*, equivalently through the matched route* $W', M', Q', T', E', H', C', P', N'$*. It has no coordinate for quotient stream families, selected limits, host real equality, choice schedules, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite window selector uniqueness`

Proof. Start with tail-ledger exactness, Definition 435.9. It places every accepted comparison of each packet after the displayed finite family ledger, the $\mathsf{StreamName}^{\uparrow}$ window row, the shared modulus row, and the dyadic tolerance row. The componentwise agreement hypotheses therefore identify the two tail-ledger routes before any consumer reaches a seal-facing row.

Next apply shared-threshold transport, Definition 435.10. The transported threshold request remains the same M-route, or the matched M'-route, under the recorded hsame and Cont rows. Thus a consumer cannot distinguish the packets by choosing another modulus schedule or by moving the tolerance read outside the displayed transport.

Now use RegSeqRat tail equivalence, Definition 435.11. The regular rational readback carried by T is the same tail evidence already fixed by the window and tolerance route, and the same statement holds for T'. Since the readbacks agree componentwise, the two packets present a single $\mathsf{RegSeqRat}^{\uparrow}$ selector surface to downstream consumers.

Finally apply finite-window Real-seal extraction, Definition 435.28. The $\mathsf{Real}^{\uparrow}$ seal row is reached only through the finite window family, dyadic tolerance row, tail readback, and structural rows. The matched H, C, P, N and H', C', P', N' rows carry the same hsame, Cont, Pkg, and $\mathsf{NameCert}$ evidence, so the Real-facing and completion-facing consumers read one finite-window selector surface before E or E' is consumed.

The excluded objects are not rows of that surface. Quotient stream families, selected limits, host real equality, choice schedules, and ambient completeness cannot supply an alternate selector, so the componentwise matched packets are indistinguishable to every Real-facing or completion-facing consumer up to the seal. □

Theorem 435.30 (UniformCauchyCriterion diagonal uniqueness exit)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose its terminal rows are consumed through an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *selector-budget route and through an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *comparison route, with both routes reading the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces.*

Then every Phase Real and Completion exit read of U factors through that same finite route. The terminal read uses the stream windows carried by W, the regular rational readback carried by T, the dyadic tolerance ledger carried by Q, and the Real seal carried by E, together with the structural rows H, C, P, N. No exit read has access to quotient streams, host real equality, selected limits, choice data, ambient completeness, or a second completion source.

Proof. Start with the public export of U. By Definition 435.24, every downstream consumer is already exhausted by the finite rows $I, W, M, Q, T, E, H, C, P, N$, so the exit read has no hidden analytic coordinate before the terminal rows are interpreted.

Next apply the terminal seal envelope, the completion–Real correspondence, and the finite-window extraction route. The envelope keeps the terminal seal downstream of the displayed rows, the correspondence identifies the Real-facing and Completion-facing endpoint reads on the same seal route, and Definition 435.28 extracts the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces from W, T, Q, E.

Now attach the diagonal selector-budget route. The diagonal compatibility surface supplies the same finite faces to the UniformCauchyCriterion terminal rows, while the uniqueness-seal comparison consumes those faces through the same selector budget. By Definition 646.207, the terminal consumer and the uniqueness-seal consumer collapse to one finite pullback route rather than two endpoint sources.

Finally combine this pullback with the non-escape clauses on both sides. Since the shared route has no coordinate for quotient streams, host equality, selected limits, choice data, ambient completeness, or an independent completion object, every Phase Real and Completion exit read of U factors through the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal uniqueness exit`

Theorem 435.31 (UniformCauchyCriterion diagonal exit classifier)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose two terminal exit reads of* U*, one Phase Real-facing and one Completion-facing, consume the same displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces, the same accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *selector-budget route, and the same accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *comparison rows.*

Then the two terminal Real-facing classifier rows are hsame-equivalent. In particular, no second completion source, quotient-stream equality, host real equality, selected limit, choice datum, ambient completeness coordinate, or independent endpoint witness can distinguish the two terminal classifier reads.

Proof. Apply Definition 435.30 to the two exit reads. Both reads factor through the common finite route carried by $I, W, M, Q, T, E, H, C, P, N$, so neither route has an extra source from which a different terminal classifier could be read.

The public rows are exhausted by Definition 435.24. The Real-seal part of the route is exhausted by Definition 435.28, which reads the displayed stream name, regular rational sequence, dyadic tolerance core, and Real seal from W, T, Q, E.

Now compare the selector and uniqueness consumers. The shared $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ route and the shared $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ comparison rows are collapsed by Definition 646.207, so the Phase Real-facing and Completion-facing classifier consumers lie over the same finite pullback route.

The classifier conclusion is therefore forced at the level of $BHist$ rows. Since *hsame* identifies terminal rows read from the same finite pullback route, the two terminal Real-facing classifier rows are *hsame*-equivalent, and the excluded coordinates have no remaining slot through which to separate them. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal exit classifier`

Theorem 435.32 (UniformCauchyCriterion diagonal exit nonescape)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal exit reads share the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$*, and* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *rows.*

Then every classifier distinction visible at the diagonal exit is witnessed inside the finite rows $BHist$*, hsame,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *carried by* H, C, P, N *and the terminal route* I, W, M, Q, T, E*. Quotient streams, host real equality, selected limits, choice data, ambient completeness, metric-completeness data, and independent completion sources are outside the accepted diagonal exit classifier.*

Proof. Apply Definition 435.30. The Phase Real-facing and Completion-facing terminal reads already factor through the same finite pullback route, whose displayed coordinates are the stream name, regular rational readback, dyadic tolerance ledger, Real seal, and structural rows H, C, P, N.

Next apply Definition 435.31. It identifies any two terminal classifier reads over that shared route at the level of *hsame*, so a visible distinction must be carried by the finite $BHist$ and classifier rows rather than by an external analytic separator.

The diagonal-limit pullback, Definition 646.207, supplies no separate selector route, and the real-limit uniqueness side is bounded by the same non-escape surface used above for the $\mathsf{Real}^{\uparrow}$ seal. Hence quotient streams, host real equality, selected limits, choice data, ambient completeness, metric-completeness data, and independent completion sources have no route slot in the accepted diagonal exit classifier. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal exit nonescape`

Theorem 435.33 (UniformCauchyCriterion diagonal budget extraction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal exit is read through the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *and* $\mathsf{RealLimitUniquenessSea}$ *rows. Then the accepted terminal exit extracts exactly the finite budget route*

$$W, Q, T, E, H, C, P, N$$

for its Phase Real and Completion-facing consumers. No quotient stream, selected limit, host equality, choice row, ambient completeness witness, or independent completion datum can replace that route.

Proof. First use Definition 435.30. It places every terminal exit read of U on the shared finite route carried by the stream window, dyadic tolerance ledger, regular rational readback, Real seal, and structural rows.

Next apply Definition 435.31 and Definition 435.32. The classifier is forced inside the same $BHist$, *hsame*, Cont, Pkg, and local $\mathsf{NameCert}$ rows, and the non-escape boundary removes the listed host or completion-side coordinates.

Now pull the selector budget back through Definition 646.206. Under the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads, the UniformCauchyCriterion terminal consumer receives precisely the displayed rows W, Q, T, E, H, C, P, N.

Finally use Definition 1120.6. The Phase Real and Completion exit sees only the four completion-facing faces and their structural budget rows, so the extracted route is finite and no external stream quotient, host-selected endpoint, choice principle, ambient completeness row, or detached completion datum remains available. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal budget extraction`

Theorem 435.34 (UniformCauchyCriterion exit route consumer coverage)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal exit is read through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *and* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$

rows. Every Phase Real and Completion consumer of that accepted terminal exit factors through W, Q, T, E, H, C, P, N together with those shared sibling rows. No quotient stream, host real equality, selected limit, choice row, ambient completeness witness, or independent completion datum can replace that finite route.

Proof. First apply Definition 435.30. It places every Phase Real and Completion exit read over the same displayed terminal route rather than over separate endpoint sources.

Next use Definition 435.31 and Definition 435.32. The classifier is read inside the finite $BHist$, $hsame$, Cont, Pkg, and local NameCert structure, and the nonescape theorem removes quotient streams, host equality, selected limits, choice data, ambient completeness, and independent completion sources from the accepted exit.

Now apply Definition 435.33. It identifies W, Q, T, E, H, C, P, N as the exact budget route delivered to the terminal consumer.

Finally pull this route along Definition 646.206 and Definition 1120.6. The selector-budget pullback supplies the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, while four-face exactness keeps the Phase Real and Completion consumers on the same boundary. Hence every terminal consumer is covered by the listed finite route and no excluded coordinate can serve as a replacement. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket exit route consumer coverage`

Theorem 435.35 (UniformCauchyCriterion diagonal structural-face coverage)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal exit is read through shared* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *and* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *rows. Then the four structural faces W, Q, T, E together with H, C, P, N are exhaustive for every diagonal-exit classifier and every Phase Real or Completion consumer route.*

In particular, no accepted diagonal-exit read can replace that route by quotient streams, host equality, selected limits, choice rows, ambient completeness, metric-completeness witnesses, or independent completion sources.

Proof. First apply Definition 435.30. It places the terminal exit over the shared finite route rather than over a second endpoint source.

Next apply Definition 435.31 and Definition 435.32. The classifier is read through the same finite $BHist$, $hsame$, Cont, Pkg, and local NameCert rows, and the nonescape boundary removes quotient streams, host equality, selected limits, choice data, ambient completeness, metric-completeness data, and independent completion sources.

Now use Definition 435.33 and Definition 435.34. These identify W, Q, T, E, H, C, P, N as the exact route delivered to every Phase Real and Completion consumer.

Finally pull the route through Definition 646.206 and Definition 1120.6. The selector-budget pullback supplies the shared stream, regular sequence, dyadic core, and Real faces, while four-face exactness keeps the Real-completion boundary on W, Q, T, E. Hence the listed structural faces and support rows cover every diagonal-exit classifier and consumer route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal structural face coverage`

Theorem 435.36 (UniformCauchyCriterion diagonal tail-budget exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal exit shares the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$, *and* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *rows. Then every tail-budget read at that terminal exit factors through the already extracted route*

$$W, Q, T, E, H, C, P, N.$$

No quotient stream, host equality, selected limit, choice row, ambient completeness, detached tail oracle, second completion source, or replacement tail schedule can serve in place of that route.

Proof. Start with diagonal budget extraction, Definition 435.33. It identifies W, Q, T, E, H, C, P, N as the finite route delivered to Phase Real and Completion-facing consumers of the accepted terminal exit.

Next apply exit route consumer coverage and structural-face coverage, Definition 435.34 and Definition 435.35. These keep every terminal consumer on the same stream-window, dyadic-tolerance, regular-readback, Real-seal, transport, route, provenance, and local naming rows.

Now attach the tail-cofinality sibling route. By Definition 435.133 and the TailCofinalitySchedule route theorem cited there, the cofinal tail schedule is consumed only as a sibling route on the same finite window, regular readback, and Real seal before the terminal exit is read.

The nonescape clause is inherited from the cited routes. None of the diagonal, consumer-coverage, structural-face, or tail-cofinality rows supplies quotient streams, host equality, selected limits, choice rows, ambient completeness, detached tail oracles, or independent completion sources, so every tail-budget read is exhausted by W, Q, T, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal tail budget exhaustion`

Theorem 435.37 (UniformCauchyCriterion diagonal terminal route exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal Phase Real and Completion read is shared with the displayed* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then every terminal read of* U *is exhausted by the finite route*

$$W, Q, T, E, H, C, P, N.$$

No accepted terminal consumer can replace that route by quotient streams, host equality, selected limits, choice data, ambient completeness, metric-completeness data, or a detached completion source.

Proof. First use Definition 435.35. It says that the four structural faces W, Q, T, E together with H, C, P, N cover every diagonal-exit classifier and every Phase Real or Completion consumer route.

Next use Definition 435.33. It identifies the budget delivered to the terminal consumer as exactly W, Q, T, E, H, C, P, N, so the terminal read has no unaccounted row outside the displayed packet.

Now combine the classifier and non-escape surfaces. By Definition 435.31 and Definition 435.32, any visible distinction at the terminal diagonal exit is already carried by the displayed $BHist$, $hsame$, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally apply Definition 435.34. Since every terminal consumer is already covered by the same finite route, the listed quotient, host-equality, choice, completeness, metric-completeness, and detached-completion coordinates have no accepted route slot from which they could replace W, Q, T, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal terminal route exhaustion`

Theorem 435.38 (UniformCauchyCriterion terminal Real-completion lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose terminal read shares the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$, *and* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *rows. Then every Real-and-Completion phase exit classifier is locked to the finite route*

$$W, Q, T, E, H, C, P, N.$$

The locked route is the one whose packet supplies $\mathsf{Cont}(I, T, R_t)$, $\mathsf{Cont}(T, E, R_E)$, $\mathsf{Cont}(R_t, R_E, R_B)$, *and* $\mathsf{Cont}(R_B, P, R_C)$, *whose package signs* R_t, R_E, R_B, *and* R_C, *and whose terminal completion read carries the local name row* N *while refusing both host-tail return continuations out of* R_B. *No quotient stream, host real equality, choice row, ambient completeness, detached completion endpoint, selected representative, or replacement tail schedule can refine that terminal classifier.*

Proof. Start with diagonal uniqueness exit and the diagonal exit classifier. They identify the accepted terminal read as a finite classifier route over the same stream name, regular rational readback, dyadic tolerance, Real seal, transport, continuation, provenance, and local naming rows.

Apply diagonal budget extraction along the concrete continuation chain. The accepted packet gives the index-to-tail read, the tail-to-seal read, the tail-read-to-seal-read budget read, and the budget-to-provenance completion read:

$$\mathsf{Cont}(I, T, R_t), \quad \mathsf{Cont}(T, E, R_E), \quad \mathsf{Cont}(R_t, R_E, R_B), \quad \mathsf{Cont}(R_B, P, R_C).$$

The same extraction supplies the package signatures of R_t, R_E, R_B, and R_C. Together with the packet signature for N, this makes the completion read a unary finite read over the displayed route and keeps the terminal local name tied to the accepted bundle.

Now use the host-tail refusal supplied by the same budget read. Both possible host-tail return continuations from R_B back to R_t are absurd, so the route cannot be refined by returning through an external tail oracle, selected representative, host equality row, or detached completion endpoint.

Finally combine exit-route consumer coverage, structural-face coverage, tail-budget exhaustion, and terminal-route exhaustion. The $\mathsf{TailCofinalitySchedule}^{\uparrow}$ row is consumed as a sibling schedule on the same finite route, and Definition 435.37 leaves no route slot outside W, Q, T, E, H, C, P, N. Therefore every terminal Real-and-Completion exit classifier is locked to that finite route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal real completion lock`

↪ variant `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion budget lock`

Theorem 435.39 (UniformCauchyCriterion Real-completion selector-route stability)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet whose terminal exit shares the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, DiagonalLimitCompatibility$^{\uparrow}$, RealLimitUniquenessSeal$^{\uparrow}$, *and* TailCofinalitySchedule$^{\uparrow}$ *rows. Suppose one Phase Real and Completion consumer route is replaced by another accepted route with the same extracted* W, Q, T, E, H, C, P, N *faces and the same displayed sibling rows.*

Then the replacement preserves the Phase Real and Completion selector boundary. Every admissible selector read remains exhausted by the same finite route W, Q, T, E, H, C, P, N*, together with the displayed stream-name, regular-sequence, dyadic-core, Real, diagonal-compatibility, uniqueness-seal, and tail-cofinality faces. No quotient stream, host equality, selected limit, countable-choice row, ambient-completeness witness, or independent completion datum can refine or distinguish the replacement route.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion selector route stability`

Proof. First apply diagonal budget extraction, Definition 435.33. It identifies the finite route W, Q, T, E, H, C, P, N delivered to Phase Real and Completion-facing consumers of the accepted terminal exit.

Next use exit route consumer coverage and structural-face coverage, Definition 435.34 and Definition 435.35. These theorems keep every consumer route on the same stream-window, dyadic-tolerance, regular-readback, Real-seal, transport, route, provenance, and local naming faces.

Now compare the original and replacement routes. By hypothesis they share the extracted W, Q, T, E, H, C, P, N faces and the same StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, DiagonalLimitCompatib RealLimitUniquenessSeal$^{\uparrow}$, and TailCofinalitySchedule$^{\uparrow}$ rows, so the replacement has no additional selector coordinate.

Finally apply diagonal terminal route exhaustion and the terminal Real-completion lock, Definition 435.37 and Definition 435.38. They exhaust all admissible Phase Real and Completion reads by the same finite route. Therefore quotient streams, host equality, selected limits, countable choice, ambient completeness, and independent completion data remain outside the preserved selector boundary. □

Theorem 435.40 (UniformCauchyCriterion diagonal uniqueness four-face terminal)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet whose terminal read shares the displayed* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, DiagonalLimitCompatibility$^{\uparrow}$, RealLimitUniquenessSeal$^{\uparrow}$, *and* TailCofinalitySchedule$^{\uparrow}$ *rows. Then every accepted UniformCauchyCriterion terminal read over those rows factors through the same four-face Phase Real and Completion packet with structural route*

$$W, Q, T, E, H, C, P, N.$$

The four faces are exactly the stream-window, dyadic-tolerance, regular-readback, and Real-seal faces, and the route is preserved through the same diagonal-compatibility, uniqueness-seal, and tail-cofinality rows. No quotient stream, host real equality, selected limit, countable-choice row, ambient-completeness witness, detached tail oracle, or independent completion datum can distinguish a second terminal read.

Proof. First apply diagonal budget extraction, Definition 435.33. It identifies W, Q, T, E, H, C, P, N as the finite budget route delivered to Phase Real and Completion-facing consumers of the accepted terminal exit.

Next use exit route consumer coverage, Definition 435.34. Every accepted terminal consumer is already routed through the same extracted budget together with the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, and $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ rows.

Now apply diagonal structural-face coverage, Definition 435.35. The route is forced to expose the four structural faces W, Q, T, E and the support rows H, C, P, N before the terminal classifier or any Phase Real and Completion consumer may be read.

Finally use selector-route stability, Definition 435.39. Since the original and replacement terminal reads share the extracted budget and the same diagonal-compatibility, uniqueness-seal, and tail-cofinality rows, the replacement preserves the Phase Real and Completion selector boundary. The listed quotient, host-equality, choice, completeness, tail-oracle, and detached-completion coordinates therefore have no accepted route slot. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal uniqueness four face terminal`

Theorem 435.41 (UniformCauchyCriterion terminal selector uniqueness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *terminal packet whose displayed finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row agree with the terminal rows of* U*. Suppose also that an accepted StreamName–RegSeqRat–RealUp tail-selector transport packet exposes the same finite windows, dyadic ledger, regular rational readback, Real seal, and visible transport rows.*

Then every $\mathsf{Real}^{\uparrow}$*-facing or completion-facing terminal consumer of these accepted packets reads one terminal selector surface before the seal. That surface is the common finite route through* W, D, R, E, H, C, P, N*, equivalently through the UniformCauchyCriterion rows* W, Q, T, E, H, C, P, N*. It has no coordinate for quotient streams, host real equality, hidden selectors, selected limits, choice schedules, metric-completeness rows, or ambient completeness.*

Proof. Start with the UniformCauchyCriterion side. By Definition 435.29, a $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer of U cannot read past the displayed finite-window selector surface before the seal. The surface is already fixed by the shared $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ tail readback, $\mathsf{Real}^{\uparrow}$ seal row, and the structural H, C, P, N rows.

Next attach the terminal observation-budget selector. The coverage theorem Definition 782.36 places every compatible terminal consumer of F on the finite route through B, S, W, D, R, E and the same H, C, P, N envelope. The seal inversion theorem Definition 782.50 identifies the seal-facing route with the terminal window route, so the consumer cannot introduce a second terminal selector between the window and the seal.

Now descend the StreamName–RegSeqRat–RealUp transport. By Definition 126.214, the public tail-selector interface is exactly the displayed finite ledger carrying the StreamName windows, regular rational readback, dyadic endpoint rows, RealUp readback, and visible hsame/Cont/Pkg/NameCert

support. Since those rows agree with the terminal rows of U and F, the transport contributes the same selector surface rather than a separate one.

Combining the three surfaces gives a single terminal selector before E. The excluded data are absent from every cited route: quotient streams, host real equality, hidden selectors, selected limits, choice schedules, metric-completeness rows, and ambient completeness are not rows of the accepted UniformCauchyCriterion packet, the terminal observation-budget selector, or the public tail-selector transport. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal selector uniqueness`

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet is generated from a finite family index ledger, a shared $\mathsf{StreamName}^{\uparrow}$ window family, one uniform $\mathsf{CauchyModulus}^{\uparrow}$ schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a finite $\mathsf{DyadicTailBound}^{\uparrow}$ comparison ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature surface records the finite carrier, local naming obligations, shared-window stability, $\mathsf{Real}^{\uparrow}$ seal handoff, non-escape boundary, tail-ledger exactness, shared-threshold transport, RegSeqRat tail equivalence, downstream Real consumer export, and public non-escape export for uniform dyadic Cauchy tests over finite families of stream-name windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream families, host real equality, selected family limits, ambient completeness, public uniform convergence, or bridge checking.

Upgrade path: No further paper closure grade remains for this chapter; Lean formalization of the packet and exported consumer theorems remains pending.

The late boundary and overlap routes consume the L10 sibling surfaces chapter 126, chapter 301, chapter 38, and the existing dyadic tail and tolerance rows already named in the parent packet. They read those dependencies through the same shared window, threshold, dyadic tolerance, tail-ledger, and seal handoff coordinates.

Theorem 435.42 (UniformCauchyCriterion empty-family tail ledger)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose that I is the empty finite family index ledger and that T is the empty finite tail-comparison ledger. Then the shared-threshold route is accepted as the vacuous finite-family route*

$$\mathsf{empty}_I \longmapsto W \longmapsto M \longmapsto Q \longmapsto \mathsf{empty}_T \longmapsto E.$$

It has no family member at which a tail comparison can be requested, no completed family limit to export through the $\mathsf{Real}^{\uparrow}$ *seal, and no indexwise schedule distinct from the single displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row M.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket empty family tail ledger`

Proof. Project the accepted packet through Definition 435.1. Its only family-facing rows are the finite index ledger I, the shared window family W, the single modulus schedule M, the tolerance row Q, and the tail-comparison ledger T.

Because I is empty, there is no displayed family index through which a consumer can select a component window or request an indexwise threshold. The window-stability row of Definition 435.3 therefore has no component case to check; the shared row M remains the only threshold schedule visible in the packet.

Because T is empty, Definition 435.9 leaves no accepted tail comparison after the route through I, W, M, Q. Thus the finite family criterion is satisfied only in the vacuous sense: every possible comparison for an index in I is already exhausted by the absence of such an index and by the absence of a tail-ledger entry.

Now apply Definition 435.10. Transport can rename the displayed rows through hsame and replay their reads through Cont, but it cannot replace the single schedule M by a family of independently selected schedules. With the family ledger empty, there is not even an index at which such a schedule could be named.

Finally use Definition 435.4 and Definition 435.8. Every $\mathsf{Real}^{\uparrow}$ seal read still factors through I, W, M, Q, T, E, and in the present boundary this is precisely the displayed route through the empty family ledger and empty tail ledger. Since the packet has no coordinate for a completed family limit, quotient stream family, host real equality, selected limit, or ambient completeness principle, the seal handoff exports none of them. □

Theorem 435.43 (UniformCauchyCriterion single-member restriction exactness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let i be a displayed member of the finite family ledger I. The singleton restriction $U|_{\{i\}}$ reads the same shared rows $W, M, Q, T, E, H, C, P, N$ through the component selected by i. Its finite-family ledger is the one-member restriction of I, and it adds no pointwise limit, no indexwise modulus, and no endpoint read outside the original tail ledger.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket single member restriction exactness`

Proof. Project U through Definition 435.1. The family-facing rows are I, W, M, Q, T, and the singleton request supplies only a displayed member i of I.

Apply Definition 435.3. Restricting to the component named by i keeps the read inside the same shared window family W and transports only the selected component through the existing hsame rows.

Next use Definition 435.9 and Definition 435.10. The tail comparison for the singleton still passes through the same M, Q, T route, so the restricted packet has no coordinate for an independently chosen pointwise threshold.

Finally apply Definition 435.16 and Definition 435.18. A downstream consumer sees the same regular-rational and seal handoff rows, specialized to the displayed singleton, and cannot extract a pointwise limit from the restriction. □

Theorem 435.44 (UniformCauchyCriterion paired-family union ledger)**.** *Let J and K be two displayed subfamily ledgers of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N).$$

Suppose both subfamily ledgers factor through the same I, W, M, Q, T rows. Then the paired-family ledger $J \cup K$ is accepted over the same shared threshold route. Its tail comparisons are exactly the

union of the two displayed subfamily tail reads inside T, transported by the same H, C rows and sealed through the same E, P, N provenance.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket paired family union ledger`

Proof. Read the two subfamily ledgers as displayed restrictions of I. Since both factor through the same packet U, neither subfamily supplies an additional window family or modulus row.

Use Definition 435.3 to keep both subfamily reads on W. Use Definition 435.10 to keep both threshold requests on the single schedule M.

Apply Definition 435.9. The only accepted tail-comparison entries are the displayed rows of T after the route through I, W, M, Q, so the paired ledger is their finite union inside that same ledger.

It remains to package the paired read for consumers. Use Definition 435.16 and Definition 435.18: the union ledger is replayed by the same H, C rows and inherits the same E, P, N seal and provenance surface. □

Theorem 435.45 (UniformCauchyCriterion shared-schedule Real seal inheritance)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a singleton restriction or paired-family ledger of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet inherits the same shared schedule and seal handoff route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The inherited route may restrict or pair displayed family entries, but it cannot introduce a new Cauchy schedule, selected family limit, quotient stream family, host real equality, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared schedule real seal inheritance`

Proof. Begin with the real-facing consumer read. By Definition 435.4, every such read reaches E only after passing through I, W, M, Q, T.

For a singleton restriction, apply Definition 435.43. The selected component keeps the same shared schedule and specializes only the finite family ledger.

For a paired-family read, apply Definition 435.44. The paired ledger is assembled inside the same tail ledger and is replayed by the same transport and continuation rows.

Finally use Definition 435.16, Definition 435.18, and Definition 435.8. These package the inherited seal route and rule out any new schedule, selected limit, quotient stream family, host equality, or completeness principle. □

Theorem 435.46 (UniformCauchyCriterion shared-window threshold meet)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The shared window family W and the single modulus row M determine one finite threshold meet before any Q, T, E read is available. Every tolerance row, tail-comparison row, and* $\mathsf{Real}^{\uparrow}$ *seal row is therefore downstream of the same W, M route and cannot select an indexwise modulus family.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionSharedWindowThresholdMeet`

Proof. First project the carrier of Definition 435.1. The packet has one shared window family row W and one modulus schedule row M.

Apply Definition 435.3. Transport of a family request stays inside the same window family, so no later read can replace W by a separately chosen component window.

Next apply Definition 435.10. The threshold route is transported as the single row M, and the tolerance, tail, and seal coordinates are read only after that route has been fixed.

It remains to rule out later selection. Use Definition 435.9 and Definition 435.4: Q, T, E are consumer rows downstream of I, W, M, not sources for a second modulus family. □

Theorem 435.47 (UniformCauchyCriterion family-tail intersection)**.** *For every accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet and every nonempty subfamily request, the tail read is an intersection of the existing finite family ledger with the same W, M, Q, T route. It is not an independently chosen family of moduli, and it cannot detach a subfamily from the displayed index ledger I or the shared threshold route.*

Proof. Begin with the finite index ledger I. A nonempty subfamily request is a read of displayed indices from I, so it has no source outside the accepted packet.

Use Definition 435.3 and Definition 435.10. These keep the subfamily request on the same window family W and the same modulus row M.

Now apply Definition 435.9. A tail comparison is accepted only as a read through Q and T after the displayed family and threshold rows are fixed.

Thus the subfamily tail is the finite intersection of the already displayed family ledger with the same W, M, Q, T route. The packet has no coordinate for an independently chosen family of moduli. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket family tail intersection`

Theorem 435.48 (UniformCauchyCriterion RegSeqRat uniform consumer exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet is exhausted by the rows*

$$I, W, M, Q, T, E, H, C, P, N.$$

Such a consumer reads the shared window family, single threshold row, dyadic tolerance row, finite tail ledger, seal handoff, transport rows, continuation routes, package provenance, and local naming row. It cannot export quotient streams, selected limits, host real equality, or ambient completeness.

Proof. Start from Definition 435.11. The $\mathsf{RegSeqRat}^{\uparrow}$ read is the same finite evidence as the tail-ledger route, so it must pass through I, W, M, Q, T.

Apply Definition 435.46. This fixes the shared window and threshold rows before the consumer can inspect the tolerance, tail, or seal coordinates.

Use Definition 435.47 for nonempty subfamily reads, and use Definition 435.42 for the empty-family boundary. Both cases keep the consumer inside the finite family ledger and the same W, M, Q, T route.

Finally apply Definition 435.8. The remaining rows E, H, C, P, N package the seal, transport, continuation, provenance, and local name, but they add no quotient stream, selected limit, host real equality, or completeness principle. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseqrat uniform consumer exhaustion`

Theorem 435.49 (UniformCauchyCriterion tail-overlap budget exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every overlap of two finite subfamily tail requests in* U *factors through the same finite rows* $I, W, M, Q, T, E, H, C, P, N$*. The overlap has no second tail budget, quotient stream family, selected family limit, host real equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail overlap budget exhaustion`

Proof. First combine the paired-family union ledger with the shared-window threshold meet. By Definition 435.44, two finite family requests are joined only inside the displayed index ledger I. By Definition 435.46, their common window and threshold route is the same W, M route, not a pair of separately chosen schedules.

Next pass from the joined family request to the tail read. Use Definition 435.47: the common tail is an intersection of the displayed family ledger with W, M, Q, T. Thus the dyadic tolerance row and tail ledger row are inherited from U rather than rebuilt for the overlap.

Finally send the overlap to the consumers. Apply Definition 435.48 to exhaust the RegSeqRat side by $I, W, M, Q, T, E, H, C, P, N$, and apply Definition 435.57 to keep the Real seal side downstream of the same package. The non-escape boundary then rules out a second budget, quotient stream family, selected limit, host real equality, or ambient completeness row. □

Theorem 435.50 (UniformCauchyCriterion RegSeqRat–Real seal overlap)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any overlap of two finite subfamily tail requests that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer and then a* $\mathsf{Real}^{\uparrow}$ *seal factors through the same route*

$$I, W, M, Q, T, E, H, C, P, N.$$

The overlap cannot introduce a second RegSeqRat classifier, quotient stream family, selected family limit, host real equality, or ambient completeness witness.

Proof. First place the two subfamily requests on the common window. Apply Definition 435.46 to identify the shared window and threshold rows W, M before either request reaches its tolerance or tail data.

Next pass to the finite family-tail overlap. Apply Definition 435.47. The common tail evidence is the displayed intersection inside the finite ledger I and the same dyadic tolerance and tail rows Q, T.

Now read the RegSeqRat-facing consumer. By Definition 435.48, the consumer is exhausted by $I, W, M, Q, T, E, H, C, P, N$, so it has no coordinate for a second RegSeqRat classifier or quotient stream family.

Finally attach the Real seal. Use Definition 435.45 and Definition 435.57 to keep the seal read downstream of the same E, H, C, P, N rows. The non-escape boundary then rules out selected family limits, host real equality, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseqrat real seal overlap`

Theorem 435.51 (UniformCauchyCriterion RegSeqRat–Real seal pullback exactness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every displayed subfamily pullback that reaches both a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer and a* $\mathsf{Real}^{\uparrow}$ *seal is exact on the same route*

$$I, W, M, Q, T, E, H, C, P, N.$$

It cannot introduce a second RegSeqRat classifier, quotient stream family, selected family limit, host real equality, or ambient completeness witness.

Proof. First put the pullback on the shared threshold route. Apply Definition 435.46: the shared window row W and the single modulus row M are fixed before any tolerance, tail, or seal row is read.

Next restrict the subfamily tail. By Definition 435.47, the pullback is an intersection of the existing finite family ledger with the same W, M, Q, T route. It therefore has no separate index carrier or independently selected modulus family.

Now exhaust the RegSeqRat-facing side. Use Definition 435.48; the consumer is exhausted by $I, W, M, Q, T, E, H, C, P, N$ and has no coordinate for a second RegSeqRat classifier or quotient stream family.

Finally attach the Real seal through Definition 435.50. The overlap already reaches the $\mathsf{RegSeqRat}^{\uparrow}$ consumer and the $\mathsf{Real}^{\uparrow}$ seal through the same route. The non-escape boundary then rules out selected family limits, host real equality, and ambient completeness, so the displayed pullback is exact on the original packet rows. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseqrat real seal pullback exactness`

Theorem 435.52 (UniformCauchyCriterion shared-tail subfamily pullback)**.** *For every accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet* $U = (I, W, M, Q, T, E, H, C, P, N)$*, any downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a displayed subfamily pullback reads the same shared tail route through the finite index ledger. Such a consumer cannot introduce an indexwise modulus family or detach the pullback from the rows* $I, W, M, Q, T, E, H, C, P, N$*.*

Proof. Project the carrier rows of U. The only finite family ledger is I, the only shared window and threshold route is W, M, and the only tail and seal rows available to a consumer are Q, T, E with their displayed transport, continuation, provenance, and naming rows.

For a one-component pullback, use singleton restriction exactness. The singleton request is a read of the existing family ledger and therefore follows the same W, M, Q, T route. For two displayed components, use Definition 435.44; finite pairs are joined inside the displayed ledger before any tail read is exposed.

It remains to handle downstream consumers. Apply Definition 435.49. The pullback is already exhausted by the common finite route, so a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives only the shared tail package and has no coordinate for an indexwise modulus family. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared tail subfamily pullback`

Theorem 435.53 (UniformCauchyCriterion finite-index ledger transport)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every transported family request for* U *factors through the displayed finite ledger* I*, componentwise* hsame *transport, and displayed* Cont *routes before any request can reach* W, M, Q, T, E*. Thus the transported request is a replay of the public finite package and cannot choose a new index carrier, a new family window, or an external modulus schedule.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite index ledger transport`

Proof. Project the carrier of Definition 435.1. The only family ledger coordinate supplied by the packet is I, and the only transport and continuation coordinates supplied for family reads are H and C.

Apply Definition 435.3. Any transported family request that reaches W is already a read of the displayed window family over I, so it has no independent carrier from which a second family can be selected.

Next apply Definition 435.10. The same transported request reaches M, Q, T only by following the shared threshold route, with componentwise hsame evidence and the displayed Cont routes supplied by H and C.

Finally use Definition 435.4. The seal row E is downstream of the same finite ledger and transport route, so the transported request reaches W, M, Q, T, E only as a replay of I, H, C inside the public package. □

Theorem 435.54 (UniformCauchyCriterion finite-ledger route coverage)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$*-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer route for* U *factors through the displayed finite index ledger* I*, shared window* W*, single modulus* M*, tolerance row* Q*, tail ledger* T*, seal handoff* E*, transport row* H*, continuation row* C*, provenance row* P*, and local name* N*.*

Consequently the route ledger is exhausted by $I, W, M, Q, T, E, H, C, P, N$*. It has no coordinate for a quotient stream family, selected family limit, host equality, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite ledger route coverage`

Proof. First put transported family requests on the public ledger. By Definition 435.53, a transported request can reach W, M, Q, T, E only after reading I, H, C.

Next fix the threshold and tail rows. Apply Definition 435.46 to identify the shared W, M route, and apply Definition 435.47 to keep every nonempty subfamily tail inside the same I, W, M, Q, T ledger.

Now read the two consumer sides. The $\mathsf{RegSeqRat}^{\uparrow}$ side is exhausted by Definition 435.48. The shared $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ pullback route is exact by Definition 435.51.

Finally apply Definition 435.8. Since the packet exports only the displayed seal, transport, continuation, provenance, and naming rows after the tail ledger is fixed, no quotient stream family, selected limit, host equality, or ambient completeness coordinate remains. □

Theorem 435.55 (UniformCauchyCriterion shared-schedule cofinal pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every cofinal subfamily pullback of* U *that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and a* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through the same finite schedule*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T$$

before either consumer reads E, H, C, P, N*. The pullback therefore shares the displayed finite index ledger, window family, modulus schedule, dyadic tolerance row, and tail ledger, and it exports no quotient stream family, selected family limit, host real equality, or ambient completeness datum.*

Proof. First place the cofinal pullback on the finite ledger. By Definition 435.53, every transported family request already passes through I, H, C before it can reach W, M, Q, T, E.

Next identify the schedule read. Apply Definition 435.46: the shared window row W and the single modulus row M are fixed before any tolerance, tail, or seal coordinate is available.

Now restrict to the cofinal subfamily tail. Use Definition 435.47; the pullback is a finite intersection inside the displayed ledger and keeps the same W, M, Q, T route. Thus cofinality contributes no new index carrier and no independently selected modulus family.

Finally send the pullback to the two consumers. The $\mathsf{RegSeqRat}^{\uparrow}$ side is exhausted by Definition 435.48, and the shared $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ seal route is Definition 435.50. The non-escape boundary then excludes quotient stream families, selected limits, host real equality, and ambient completeness, leaving the same finite schedule for both consumers. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared schedule cofinal pullback`

Theorem 435.56 (UniformCauchyCriterion shared-tail ledger coverage)**.** *For every accepted* UniformCauchyCriterio *packet, the empty-family tail read, singleton restriction, paired-family read, and nonempty subfamily tail read are all exhausted by one shared route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T.$$

No listed tail read supplies a second tail ledger, an indexwise modulus family, or a detached comparison route outside the accepted packet.

Proof. Consider the four displayed kinds of family-tail consumer. For the empty-family boundary, apply Definition 435.42; the accepted tail evidence is the displayed empty read of the same T row.

For singleton reads, apply Definition 435.43. The selected component is a restriction of I and keeps the same W, M, Q, T route.

For paired reads, apply Definition 435.44. The pair is assembled as a finite union inside the existing tail ledger, not as a new schedule or comparison object.

For nonempty subfamily reads, apply Definition 435.47. The subfamily tail is the finite intersection of the displayed ledger with the same shared route.

These cases cover the available family-tail reads of the packet. Since each case enters T only through I, W, M, Q, the packet has no remaining coordinate from which a second tail ledger or detached route could be exported. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared tail ledger coverage`

Theorem 435.57 (UniformCauchyCriterion Real seal consumer factorization)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$-, $\mathsf{Real}^{\uparrow}$-, *or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet factors through the public finite package*

$$I, W, M, Q, T, E, H, C, P, N.$$

The factorization exposes the shared window, single threshold row, dyadic tolerance row, finite tail ledger, Real seal handoff, transport rows, continuation routes, provenance row, and local name. It cannot export quotient streams, selected limits, host real equality, or ambient completeness.

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$-facing consumer. By Definition 435.48, the consumer is exhausted by $I, W, M, Q, T, E, H, C, P, N$ and cannot leave that finite package.

For the shared threshold dependency, apply Definition 435.46. This fixes W and M before any tolerance, tail, or seal read is available, so later consumers inherit the same route.

Now pass to the $\mathsf{Real}^{\uparrow}$ seal boundary. Use Definition 435.18: the completion-facing handoff is exposed only as the seal row E over the already displayed family and tail ledger.

Finally apply Definition 435.8. The rows H, C, P, N package transport, continuation, provenance, and the local name, while the packet still exports no quotient stream, selected limit, host real equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real seal consumer factorization`

Theorem 435.58 (UniformCauchyCriterion shared tail gluing determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose two finite subfamily tail reads of* U *have the same displayed family intersection inside* I *and agree on the shared threshold route through* W, M, Q, T*. Then the glued shared-tail read induces the same* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer row and the same* $\mathsf{Real}^{\uparrow}$ *seal route through* E, H, C, P, N*. No second tail budget, quotient stream family, selected limit, host real equality, or ambient completeness coordinate can split the glued read after those rows agree.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared tail gluing determinacy`

Proof. First build the common tail ledger. Apply Definition 435.44; the two finite subfamily reads are assembled as a displayed union inside I and do not create a new schedule outside the packet.

Next identify the shared tail. By Definition 435.47, agreement of the displayed family intersection inside I makes the tail read a finite intersection of the same ledger along the same W, M, Q, T route.

Now exhaust the possible overlap budget. Use Definition 435.49; since the threshold route already agrees through W, M, Q, T, the glued read has no coordinate from which a second tail budget can be exported.

Pass to the seal-facing rows by Definition 435.50. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal handoff therefore see the same shared tail rather than two independent quotient stream families.

Finally apply Definition 435.57. The downstream consumer rows are exhausted by $I, W, M, Q, T, E, H, C, P, N$, so the glued shared tail fixes both the $\mathsf{RegSeqRat}^{\uparrow}$ consumer row and the $\mathsf{Real}^{\uparrow}$ seal route through E, H, C, P, N and leaves no selected limit, host real equality, or ambient completeness coordinate. □

Theorem 435.59 (UniformCauchyCriterion budgeted Real seal factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing seal read of* U *that consumes a uniform family tail factors through the displayed finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

together with the sibling budget surfaces of $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the dyadic tail/tolerance ledger. The factorization exposes only the finite index ledger, shared stream-name window, single threshold row, dyadic tolerance row, shared tail ledger, and Real seal handoff. It exports no quotient stream family, selected family limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket budgeted real seal factorization`

Proof. First exhaust the family-tail cases before any seal read is considered. Apply Definition 435.42 for the empty-family boundary and Definition 435.43 for singleton reads. Apply Definition 435.44 for paired-family reads and Definition 435.47 for nonempty subfamilies.

These four cases land in the shared tail ledger by Definition 435.56. Hence every uniform family tail reaches T only through the displayed I, W, M, Q route and carries no separate stream family, pointwise modulus family, or detached comparison row.

Next route the budget consumers. By Definition 435.48, the $\mathsf{RegSeqRat}^{\uparrow}$ read is exhausted by $I, W, M, Q, T, E, H, C, P, N$. By Definition 435.57, the $\mathsf{Real}^{\uparrow}$ seal read is downstream of the same finite package rather than a separate completion object.

The cited route is exactly the sibling chain through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the dyadic tail/tolerance ledger: W supplies the stream-name family surface, M, Q, T supply the RegSeqRat and dyadic budget rows, and E is the Real seal handoff. Since the route has no coordinate outside those rows, quotient stream families, selected family limits, host real equality, and ambient completeness are refused at the seal boundary. □

Theorem 435.60 (UniformCauchyCriterion terminal late-route exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every terminal* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing late-boundary consumer of* U *is exhausted by the shared finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

The exhaustion is obtained by splitting only along the displayed late cases: empty family, singleton restriction, paired-family ledger, and nonempty subfamily tail. After those cases are read, the $\mathsf{RegSeqRat}^{\uparrow}$ *consumer is handled by Definition 435.48, and the* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal consumer is handled by Definition 435.57. The terminal route exports no indexwise modulus family, quotient stream family, selected family limit, host real equality, or ambient completeness.*

Proof. Begin with the four late family cases. The empty-family boundary is Definition 435.42; the singleton restriction is Definition 435.43. Both enter the same displayed I, W, M, Q, T route.

The remaining family cases are paired-family and nonempty-subfamily reads. Apply Definition 435.44 for the paired ledger and Definition 435.47 for the nonempty subfamily tail. These two cases still use the same finite family and tail rows, so no separate modulus family or quotient stream family is introduced.

Now pass from family-tail exhaustion to terminal consumers. Apply Definition 435.48 to the $\mathsf{RegSeqRat}^{\uparrow}$-facing route; it reads exactly $I, W, M, Q, T, E, H, C, P, N$.

Finally apply Definition 435.57. The $\mathsf{Real}^{\uparrow}$ and completion-facing seal reads factor through the same package, with E as the displayed seal handoff and H, C, P, N as transport, continuation, provenance, and local naming. Since every late case has already been placed on that route, the accepted packet has no coordinate for a selected family limit, host real equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal late route exhaustion`

Theorem 435.61 (UniformCauchyCriterion finite tail readback coverage)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every finite tail readback request for U is covered by exactly one of the displayed routes: the empty-family route, the singleton restriction route, the paired-family route, or the nonempty-subfamily route. After that route is selected, every* $\mathsf{RegSeqRat}^{\uparrow}$*-,* $\mathsf{Real}^{\uparrow}$*-, or completion-facing consumer factors through*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

together with the transport, continuation, provenance, and local naming rows H, C, P, N.

Consequently no finite tail readback of U can export a quotient stream family, a selected family limit, host real equality, or ambient completeness outside the displayed packet.

Proof. Split the finite family request before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal is inspected. The empty request is handled by Definition 435.42; a one-member request is handled by Definition 435.43. A two-sided displayed family read is handled by Definition 435.44.

It remains to cover finite nonempty subfamily reads that are not already one of those boundary cases. Apply Definition 435.47. This places the read inside the displayed finite family ledger and the same W, M, Q, T route, with no source for an independently selected modulus family or detached tail ledger.

The four cases are exhaustive for finite tail readbacks because every request is either empty, a singleton, a paired displayed family, or a nonempty subfamily read. Apply Definition 435.56 to put all of them on the shared tail ledger, then apply Definition 435.60 to pass from that ledger to terminal consumers.

The consumer handoff is therefore the displayed route through I, W, M, Q, T, E and then through H, C, P, N. Since the accepted packet has no coordinate for quotient stream families, selected limits, host equality, or ambient completeness, none of those principles can escape through a finite tail readback. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite tail readback route`

Theorem 435.62 (UniformCauchyCriterion finite-tail window factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every finite tail readback covered by the empty-family, singleton, paired-family, or nonempty-subfamily case of Definition 435.61 factors through one L10 handoff route:*

$$I, W, M, Q, T \longmapsto \mathsf{FiniteTailFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the structural rows E, H, C, P, N*. The* $\mathsf{FiniteTailFilter}^{\uparrow}$ *stage is the finite-tail window route supplied by the tail-window exactness and Real-seal handoff rows of chapter 1119; the* $\mathsf{StreamName}^{\uparrow}$ *stage reads the finite-window classifier already displayed by* W*; the* $\mathsf{RegSeqRat}^{\uparrow}$ *stage consumes the tail readback carried by* T*; and the* $\mathsf{Real}^{\uparrow}$ *stage seals the same route through* E*.*

Thus the finite readback coverage of U *exports no indexwise modulus family, quotient stream family, selected limit, host real equality, ambient completeness principle, or detached tail filter outside the displayed packet.*

Proof. Begin with Definition 435.61. It reduces every finite tail readback to one of the empty, singleton, paired-family, or nonempty-subfamily routes before any terminal consumer is inspected.

Next apply Definition 435.56. The four routes share the same finite family ledger and the same tail ledger T, so their readback evidence is already on the displayed I, W, M, Q, T surface.

Now pass through the finite-tail window route. By Definition 1119.4, the relevant stream window, dyadic tolerance row, regular readback, tail-budget row, common-tail comparison, and seal-facing row are coordinates of one $\mathsf{FiniteTailFilter}^{\uparrow}$ packet. By Definition 1119.5, that packet is the route seen by Real and Completion consumers.

Finally identify the two terminal consumers. The RegSeqRat-facing part is exhausted by Definition 435.48, and the Real-facing part is exhausted by Definition 435.57. These two rows read the same W, T, E, H, C, P, N surface, so the L10 handoff has no carrier coordinate for an indexwise modulus family, quotient stream family, selected limit, host real equality, ambient completeness principle, or detached tail filter. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite tail window factorization`

Theorem 435.63 (UniformCauchyCriterion late-overlap exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every empty-family, singleton, paired-family, nonempty subfamily, and transported-family request that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the same public route*

$$I, W, M, Q, T, E, H, C, P, N.$$

The route exposes the finite index ledger, shared window family, single threshold row, dyadic tolerance row, tail ledger, Real seal handoff, transport rows, continuation routes, provenance row, and local name. It cannot export indexwise moduli, selected family limits, quotient stream families, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late overlap exhaustion`

Proof. First exhaust the family-tail cases. Use Definition 435.42 for the empty boundary, Definition 435.43 for singleton reads, Definition 435.44 for paired-family reads, and Definition 435.47 for nonempty subfamilies.

These four rows place every family-tail request on the same I, W, M, Q, T route. They leave no coordinate for a separate indexwise modulus or a detached family-tail ledger.

Next handle transported requests by Definition 435.53. Transport can replay the displayed finite ledger through H and C, but it cannot choose a new index carrier, window family, or external threshold schedule.

Now apply Definition 435.56. The empty, singleton, paired, and nonempty family reads are all covered by the shared tail ledger, so the transported case also reaches consumers through the same public package after its H, C replay.

Finally use Definition 435.57. Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer is exhausted by $I, W, M, Q, T, E, H, C, P, N$, and the seal boundary exports no indexwise moduli, selected family limits, quotient stream families, host real equality, or ambient completeness. □

Theorem 435.64 (UniformCauchyCriterion tail-budget Real route)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every Real-facing or completion-facing read of U factors first through the public route*

$$I, W, M, Q, T, E, H, C, P, N$$

and then through the $\mathsf{TailBudgetCoherence}^{\uparrow}$ *public boundary for finite tail-budget handoffs. The consumer receives the finite index ledger, shared window family, single threshold row, dyadic tolerance row, tail ledger, Real seal row, transports, continuation routes, provenance, and local name; it receives no direct Real equality, quotient completion object, selected family limit, or ambient completeness principle.*

Proof. Begin with Definition 435.63. It sends every empty-family, singleton, paired-family, nonempty subfamily, and transported-family request to the same finite route $I, W, M, Q, T, E, H, C, P, N$ before the request can reach a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer.

Next apply Definition 435.56. The tail row used by those consumers is the shared T ledger reached through I, W, M, Q, so the route already has the finite tail-budget shape expected by the tail-budget handoff.

Finally use Definition 435.57 together with Definition 917.13. The Real seal row E is therefore read as a finite public tail-budget export, and every stronger Real or completion object would require a coordinate outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail budget real route`

Theorem 435.65 (UniformCauchyCriterion shared-window nonescape)**.** *For an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet $U = (I, W, M, Q, T, E, H, C, P, N)$, the shared-window and tail-budget route rules out any consumer escape to indexwise moduli, quotient stream families, selected family limits, host real equality, or ambient completeness. Every such attempted read is refused unless it is represented by the displayed finite rows $I, W, M, Q, T, E, H, C, P, N$ and then by the* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *public route.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared window nonescape certificate`

Proof. The shared window is fixed by Definition 435.46. All later tolerance and tail requests inherit the single W, M route.

Use Definition 435.56 to place the empty, singleton, paired, and nonempty family reads on the same I, W, M, Q, T ledger. No separate indexwise modulus family or quotient stream family is introduced in any case.

Now apply Definition 435.57. The Real seal and completion-facing reads are exhausted by E, H, C, P, N over the same finite route.

The remaining excluded reads are exactly the stronger analytic objects denied by Definition 917.12. Thus the packet has no escape path to selected family limits, host real equality, or ambient completeness. □

Theorem 435.66 (UniformCauchyCriterion completion consumer package)**.** *Every completion-facing consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet is packaged by the finite chain of sibling surfaces*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{TailBudgetCoherence}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The package reads the shared window, rational regular-sequence readback, dyadic tolerance row, limit-seal handoff, finite tail-budget coherence row, and Real-facing seal only through $I, W, M, Q, T, E, H, C, P, N$*. It exports no quotient completion carrier, countable-choice selector, selected family limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket completion consumer package`

Proof. Start with the late-overlap theorem. By Definition 435.63, every family-tail or transported-family request reaching a completion consumer has already passed through the finite package $I, W, M, Q, T, E, H, C, P, N$.

Read the sibling rows in order. The shared window supplies the $\mathsf{StreamName}^{\uparrow}$-style finite window, the rational readback supplies the $\mathsf{RegSeqRat}^{\uparrow}$ surface, the tolerance row supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ datum, and the seal row supplies the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff.

The public export boundary of Definition 917.13 then repackages these rows as the $\mathsf{TailBudgetCoherence}^{\uparrow}$ route consumed by $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 435.65. Since every consumer-facing row stays inside the same finite package, the excluded quotient, selector, limit, equality, and completeness objects have no exported coordinate. □

Theorem 435.67 (UniformCauchyCriterion index restriction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let* J *be a displayed finite subledger of* I*. The restricted family request over* J *is accepted through the same route*

$$J \subseteq I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

It preserves the same shared window family, threshold schedule, dyadic tolerance row, tail ledger, and real-seal handoff, and it exports no selected limit, host equality, quotient stream, or ambient completeness datum.

Lean4 ✓ BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket index restriction

Proof. Begin with Definition 435.15. A displayed subledger J is read as a restriction of the finite index ledger I, so it has no independent family carrier.

Use Definition 435.3 and Definition 435.10. The restricted request remains on the same shared window family W and the same threshold schedule M before it reaches any tolerance or tail comparison row.

Apply Definition 435.9. The tail evidence for J is exactly the displayed subledger of T reached through I, W, M, Q.

Finally use Definition 435.57. The seal read is downstream of the same finite package, so the restricted packet cannot export selected limits, host equality, quotient streams, or ambient completeness. □

↪ variant BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket subfamily restriction

Theorem 435.68 (UniformCauchyCriterion window refinement)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose* W' *is a transported* $\mathsf{StreamName}^{\uparrow}$ *window refinement of* W *along the displayed* hsame *row* H *and displayed* Cont *routes* C*. Then the refined request is accepted through the same shared-threshold and dyadic-tail route* I, W, M, Q, T, E*, with* W' *serving only as the transported presentation of the original window family.*

Lean4 ✓ BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket window refinement exactness

↪ variant BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket window refinement seal route

Proof. Project the accepted carrier and the transported refinement. The refinement data are read through H and C, so W' is not a second window family but a transported presentation of the displayed W row.

Apply Definition 435.3. This keeps the refined $\mathsf{StreamName}^{\uparrow}$ request inside the same finite window family over the index ledger I.

Next use Definition 435.10. The threshold schedule remains the single row M, and the dyadic tolerance row Q is reached only after that shared threshold route.

Apply Definition 435.9 and then Definition 435.57. The refined window request consumes the same tail ledger and seal handoff, so it cannot detach a new modulus family, selected limit, host equality, quotient stream, or ambient completeness claim. □

Theorem 435.69 (UniformCauchyCriterion finite family union)**.** *Let* U_J *and* U_K *be accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packets whose displayed subfamily ledgers* J *and* K *use the same shared window family* W*, threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, real-seal handoff* E*, and compatible finite index ledgers inside a common* BHist *packet. Then the finite union ledger* $J \cup K$ *is accepted as a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *family read through the same* $W, M, Q, T, E, H, C, P, N$ *route.*

The union packet is assembled only by finite ledger union and componentwise transport. It does not select pointwise limits, compare host reals, quotient stream names, or appeal to ambient completeness.

Lean4 ✓ BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket family union

Proof. Read the two packets through their common displayed schedule and compatible finite ledgers. Compatibility means both subfamily requests already sit inside the same BHist inventory and do not supply separate threshold or tolerance coordinates.

Apply Definition 435.44. The paired-family ledger assembles $J \cup K$ as a finite union inside the existing index and tail ledgers.

Use Definition 435.10 and Definition 435.9. These keep the union request on the same W, M, Q, T route and force every tail comparison to be a displayed row of the shared ledger.

Finally apply Definition 435.57. The real-seal consumer reads the same finite package after the union ledger is assembled, so selected limits, host equality, quotient streams, and ambient completeness have no coordinate through which to enter. □

Theorem 435.70 (UniformCauchyCriterion shared window pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet. Suppose* UniformCauchyCriterion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *seal consumers all read the displayed rows* $I, W, M, Q, T, E, H, C, P, N$ *of* U*. Then their consumer-visible packets form one finite pullback surface over the shared window row* W*, shared threshold row* M*, dyadic tolerance row* Q*, tail ledger* T*, and seal row* E*.*

Equivalently, any two such consumers that agree on the displayed BMark *and* BHist *routes* H, C *have the same finite-window projection and the same seal-facing tail packet. No consumer can add a second stream-name family, RegSeqRat ledger, dyadic tolerance route, Real seal handoff, selected limit, host equality, quotient stream, or ambient completeness datum.*

Proof. First fix the window surface. Apply Definition 435.3: every StreamName$^{\uparrow}$ read of the accepted packet is already a read of the displayed family W over the finite index ledger I.

Next fix the threshold and tail coordinates. Apply Definition 435.10 and Definition 435.9. The route through M, Q, T is therefore the single transported route determined by H, C.

Now pass to the consumer boundary. By Definition 435.48, every RegSeqRat$^{\uparrow}$-visible read is exhausted by the same finite package $I, W, M, Q, T, E, H, C, P, N$. By Definition 435.57, the Real$^{\uparrow}$ seal read factors through that package rather than through a separate completion object.

Finally use Definition 435.68 for transported presentations and Definition 435.69 for finite unions of compatible subledgers. These two closure operations preserve the same W, M, Q, T, E, H, C surface, so the consumer-visible packets are glued only along their common displayed rows. That is precisely the finite pullback surface claimed above. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket shared window regseqrat real pullback`

Theorem 435.71 (UniformCauchyCriterion overlap refinement determinacy)**.** *Let* U *be an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet with displayed rows* $I, W, M, Q, T, E, H, C, P, N$*. If two refined window reads* W_1 *and* W_2 *overlap on a displayed finite subledger of* I *and both refinements are transported through compatible* H, C *routes, then the overlap determines the same tail ledger row* T *and the same seal route* E *for both reads.*

Thus an overlapping refinement cannot expose two tail ledgers, two Real seal routes, or a detached RegSeqRat tolerance route over the common finite-window surface.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket overlap refinement determinacy`

Proof. Apply Definition 435.68 to both refined reads. Each refinement is only a transported presentation of the same displayed window family W, so their overlap still lies over the finite index ledger I.

Use Definition 435.10. Compatibility of the displayed H, C routes forces both refined reads to reach the same threshold row M and dyadic tolerance row Q before any tail comparison is read.

Now apply Definition 435.9. On the common subledger, the tail evidence is exactly the displayed restriction of T.

For the seal boundary, use Definition 435.57. Since the seal row is downstream of the same finite package, the overlap has no coordinate from which a second seal route or detached tolerance route can be produced. □

Theorem 435.72 (UniformCauchyCriterion seal route uniqueness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A* $\mathsf{Real}^{\uparrow}$ *seal consumer of* U *cannot obtain a second seal route from the same accepted packet: every seal-facing read factors through the displayed route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The packet therefore has one consumer-visible seal handoff and no alternate handoff through selected limits, host real equality, quotient stream names, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket seal route uniqueness`

Proof. Start with Definition 435.57. Any $\mathsf{Real}^{\uparrow}$ seal-facing consumer is exhausted by the displayed finite package $I, W, M, Q, T, E, H, C, P, N$.

Apply Definition 435.10 and Definition 435.9. These identify the only path from the window family to the tail ledger as W, M, Q, T.

Finally use Definition 435.48. The RegSeqRat-facing side also has no route outside the same finite package, so a second seal handoff would require a second visible row not present in U. Hence every seal read is the displayed route through E. □

Theorem 435.73 (UniformCauchyCriterion late-boundary formal target packet)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *late-boundary packet. Suppose the empty-family tail ledger, paired-family union ledger, family-tail intersection, overlap-refinement determinacy, and seal-route uniqueness rows of* U *are displayed. Then these rows assemble one finite consumer surface over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{TailBudgetCoherence}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Equivalently, every late-boundary consumer reads the same displayed package $I, W, M, Q, T, E, H, C, P, N$ *before it reaches a family-tail, overlap, or Real-seal conclusion. The packet exports no second family ledger, detached tail budget, alternate overlap route, selected limit, quotient stream family, host equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late boundary formal target packet`

Proof. First read the family endpoints. Apply Definition 435.42 for the empty-family boundary and Definition 435.44 for paired family reads. Both rows remain inside the displayed index, window, modulus, dyadic tail, and tail-budget ledger I, W, M, Q, T.

Next read the common tail surface. Apply Definition 435.47. Nonempty subfamilies meet the same displayed tail ledger, so the empty, paired, and intersected reads share one finite family-tail package.

Now attach the boundary route. Use Definition 435.71 to make overlapping refinements read the same T row and seal route, and use Definition 435.72 to identify the Real-facing handoff as the displayed route through E.

The remaining rows H, C, P, N are exactly the transport, continuation, package, and local naming coordinates of the same packet. Since the preceding rows already exhaust the family-tail, overlap, and seal boundaries, there is no carrier coordinate from which an additional consumer surface can be exported. □

Theorem 435.74 (UniformCauchyCriterion diagonal-budget pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *family packet, and let*

$$B = (\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *packet. Suppose the displayed* $\mathsf{StreamName}^{\uparrow}$ *window of* U *agrees with* W_B*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of* U *agrees with* D_B*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of* U *agrees with* K_B*, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of* U *agrees with* S_B*, all after the displayed* hsame *transports and* Cont *routes are replayed. Then every family-tail or completion-facing read of* U *factors through the shared finite route*

$$\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B.$$

The pullback adds no separate family limit, quotient stream family, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal budget pullback`

Proof. First identify the UniformCauchyCriterion side. The shared-window pullback theorem, Definition 435.70, makes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers of U read the same finite $W, M, Q, T, E, H, C, P, N$ surface.

Next close the seal route inside U. Apply seal-route uniqueness, Definition 435.72, and the late-boundary formal target packet, Definition 435.73. They force every family-tail, overlap, or Real-seal consumer to pass through the displayed family packet before it reaches a completion-facing conclusion.

Now compare with the diagonal budget. The componentwise agreement hypotheses identify the shared window, tolerance ledger, RegSeqRat readback row, and Real seal boundary with W_B, D_B, K_B, S_B. The Cauchy diagonal budget route theorem, Definition 853.21, and limit-seal eligibility, Definition 853.22, then place the same consumer behind $\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B$.

Since both packets expose only displayed transport, continuation, provenance, and local naming rows, the common pullback leaves no coordinate for a separate family limit, quotient stream family, host real equality, countable-choice schedule, or ambient completeness principle. □

Theorem 435.75 (UniformCauchyCriterion paired-overlap real seal determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose the paired-family union row, family-tail intersection row, overlap-refinement determinacy row, seal-route uniqueness row, and diagonal-budget pullback row are displayed for* U*. Then every paired-overlap* $\mathsf{Real}^{\uparrow}$*-facing consumer factors through one finite route over*

$$I, W, M, Q, T, E, H, C, P, N$$

and through the shared $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *route determined by the diagonal-budget pullback.*

Thus the consumer-visible handoff is the same finite family-tail, overlap, seal, and diagonal-budget surface. It exports no second family limit, quotient stream family, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket paired overlap real seal determinacy`

Proof. First assemble the finite paired-family surface. Apply Definition 435.44; the two family requests are read as one finite union ledger inside the displayed index, window, modulus, dyadic tolerance, and tail rows.

Next force the overlap to remain on that surface. Apply Definition 435.47 and Definition 435.71. The common tail evidence is the displayed restriction of T, and every overlapping refinement reads the same seal-facing route.

Now close the Real-facing boundary. Apply Definition 435.72. Any $\mathsf{Real}^{\uparrow}$ consumer of the paired-overlap packet must pass through the displayed handoff E after reading the same $I, W, M, Q, T, H, C, P, N$ rows.

Finally transport the result to the shared diagonal-budget route by Definition 435.74. The diagonal route identifies the compatible window, tolerance, readback, and seal coordinates, so the paired-overlap consumer has no coordinate from which to export a second family limit, quotient stream family, host equality, countable-choice schedule, or ambient completeness datum. □

Theorem 435.76 (UniformCauchyCriterion late-boundary no-choice exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every empty-family, singleton, paired-family, nonempty subfamily, overlap-refinement, or diagonal-budget* $\mathsf{Real}^{\uparrow}$*-seal consumer of* U *factors through the same finite rows*

$$I, W, M, Q, T, E, H, C, P, N$$

and, when a $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *packet is displayed, through the corresponding diagonal route*

$$\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B.$$

Thus the late boundary has one consumer-visible family-tail, overlap, seal, and diagonal-budget surface. It exports no countable-choice schedule, selected family limit, quotient stream family, host real equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late boundary nochoice exhaustion`

Proof. First exhaust the family-tail cases inside the displayed packet. Apply Definition 435.42 for the empty-family boundary and Definition 435.43 for singleton reads. These cases use only the displayed index ledger, window row, threshold row, dyadic tolerance row, and tail ledger.

For finite family combinations, apply Definition 435.44 and Definition 435.47. Paired-family and nonempty subfamily consumers are assembled by finite ledger union or finite tail intersection inside I, W, M, Q, T, so they cannot supply an indexwise schedule or a separate family-limit coordinate.

Next close the overlap and seal boundary. Use Definition 435.71 to force overlapping refinements to read the same tail ledger row and Real-facing route, and use Definition 435.72 to identify every seal-facing read with the displayed handoff through E.

When a diagonal budget is present, apply Definition 435.74. The compatible window, tolerance, RegSeqRat readback, and Real seal coordinates are then read through $\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B$, not through a second completion-facing object.

Finally apply Definition 435.75. The rows H, C, P, N provide only transport, continuation, provenance, and local naming for the same finite route. The carrier inventory therefore contains no coordinate from which a countable-choice schedule, selected family limit, quotient stream family, host equality, or ambient completeness datum can escape. □

Theorem 435.77 (UniformCauchyCriterion late-overlap route exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every empty-family, singleton, paired-family, nonempty-subfamily, overlap-refinement, diagonal-budget,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$*-seal consumer of* U *is routed through the same finite surface*

$$I, W, M, Q, T, E, H, C, P, N.$$

When a diagonal budget is displayed, the same consumer also factors through the shared pullback route

$$\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B.$$

Thus paired-overlap, diagonal-budget, CauchyLimitSeal, and RealUp consumers have one late-boundary route and no separate family limit, quotient stream family, host real equality, countable-choice schedule, or ambient completeness coordinate.

Proof. First exhaust the elementary family cases. Apply Definition 435.42 and Definition 435.43 for empty-family and singleton reads; both remain inside the displayed I, W, M, Q, T rows.

Next assemble finite family consumers. Use Definition 435.44 and Definition 435.47. Paired-family and nonempty-subfamily reads are finite union and intersection routes over the same index, window, modulus, tolerance, and tail surface.

Now close the overlap and seal route. Apply Definition 435.71 and Definition 435.72. Overlap refinements, CauchyLimitSeal consumers, and RealUp-seal consumers must pass through the displayed handoff row E with support from H, C, P, N.

When a diagonal budget is present, apply Definition 435.74. The compatible window, dyadic tolerance ledger, RegSeqRat readback, and Real seal are then read through $\epsilon, M_B, W_B, D_B, K_B, S_B, H_B, C_B, P_B, N_B$ rather than through an additional completion object.

Finally combine Definition 435.75 with Definition 435.76. The first theorem identifies paired-overlap Real-seal consumers with the diagonal-budget pullback, and the second theorem exhausts all late-boundary no-choice consumers. Hence every listed consumer uses the same finite route and exports no separate family limit, quotient stream family, host real equality, countable-choice schedule, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late overlap route exhaustion`

Theorem 435.78 (UniformCauchyCriterion late-overlap Real-seal pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every late-overlap* $\mathsf{Real}^{\uparrow}$*-seal consumer of* U *pulls back to the same finite route over*

$$I, W, M, Q, T, E, H, C, P, N.$$

The pullback first reads the shared schedule and seal inheritance, then the tail-overlap budget, then the $\mathsf{RegSeqRat}^{\uparrow}$ *consumer route, and finally the* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal overlap. It exports no second modulus schedule, quotient stream family, selected family limit, host real equality, countable-choice route, or ambient completeness coordinate.*

Proof. Start with a late-overlap $\mathsf{Real}^{\uparrow}$-seal consumer of U. By Definition 435.45, singleton and paired-family reads inherit the same shared schedule M and the same seal handoff through E, H, C, P, N.

Next apply Definition 435.49. Any overlap budget is exhausted by the displayed rows $I, W, M, Q, T, E, H, C, P, N$ so the consumer has no route to a second threshold, tolerance ledger, or tail selector.

Now read the regular-rational side through Definition 435.48. The $\mathsf{RegSeqRat}^{\uparrow}$ consumer is already factored through the same family ledger, shared windows, shared modulus, dyadic tolerance, tail ledger, seal row, transport, continuation, provenance, and local naming rows.

Finally use Definition 435.50. The Real seal and the RegSeqRat readback meet on the same finite package, and the non-escape clauses in the cited overlap route exclude quotient stream families, selected family limits, host equality, choice schedules, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late overlap real seal pullback`

Theorem 435.79 (UniformCauchyCriterion late-overlap RegSeqRat consumer factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer that reaches a late-overlap* $\mathsf{Real}^{\uparrow}$*-seal boundary of* U *factors through the same route*

$$I, W, M, Q, T, E, H, C, P, N$$

and through the Real-seal pullback of Definition 435.78. The factorization exposes no independent regular-sequence classifier, quotient stream family, second Real seal, selected family limit, host real equality, countable-choice route, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late overlap regseqrat consumer factorization`

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$ consumer at the late-overlap boundary. Apply Definition 435.48; the consumer can read only the displayed package $I, W, M, Q, T, E, H, C, P, N$.

Next force compatibility with the overlap route. By Definition 435.49, the overlap budget is the same finite tail package, and by Definition 435.45, the schedule and seal handoff remain the inherited M and E rows.

Finally apply Definition 435.78. The Real-facing side has already been pulled back to the same finite route, so the RegSeqRat consumer cannot supply an independent classifier, quotient stream family, second seal, selected limit, host equality, choice schedule, or completeness coordinate. □

Theorem 435.80 (UniformCauchyCriterion late-overlap exact-boundary handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every late-overlap* $\mathsf{Real}^{\uparrow}$*-seal consumer of* U *that factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer route determines a* RealCompletionExactBoundaryUp *handoff exactly through the displayed finite rows*

$$I, W, M, Q, T, E, H, C, P, N.$$

The handoff is the exact-boundary consumer route supplied by the finite rows and exposes no quotient stream family, selected family limit, host real equality, countable-choice route, metric-completeness coordinate, ambient-completeness coordinate, or bridge row.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket late overlap exact boundary handoff`

Proof. Start from a late-overlap $\mathsf{Real}^{\uparrow}$-seal consumer of U whose access to the boundary factors through the $\mathsf{RegSeqRat}^{\uparrow}$ route. By Definition 435.78, the Real-facing consumer already pulls back to the displayed finite rows $I, W, M, Q, T, E, H, C, P, N$.

Next apply Definition 435.79. The $\mathsf{RegSeqRat}^{\uparrow}$ side reaches that same Real-seal boundary only through the same rows, so the consumer has no independent regular-sequence classifier, second Real seal, selected limit, host equality, choice schedule, or completeness coordinate.

Read the resulting boundary as an exact-boundary consumer. By Definition 1120.13, terminal Real consumption for an accepted RealCompletionExactBoundaryUp packet is exactly a finite displayed route, with no shorter route and no route enlarged by quotient, host-equality, selected-limit, choice, completeness, or bridge coordinates.

Finally apply Definition 1120.14. Since the exact-boundary packet supplies no insertion point for quotient streams, selected family limits, host real equality, countable choice, metric completeness, ambient completeness, or bridge rows, the late-overlap consumer determines its RealCompletionExactBoundaryU handoff through the displayed rows and nowhere outside them. □

Theorem 435.81 (UniformCauchyCriterion finite-family seal obligation package)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The rows* $I, W, M, Q, T, E, H, C, P, N$ *form the finite-family seal obligation surface for every* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail read or a* $\mathsf{Real}^{\uparrow}$ *seal read.*

The surface consists exactly of the finite index ledger I*, shared window family* W*, shared threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff row* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* NameCert *row* N*. It contains no selected limit, quotient stream, host real equality, ambient completeness proof, or detached tail oracle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family seal obligation package`

Proof. First expose the finite window spine. Apply Definition 435.3; every window read remains a read of the displayed family W over the finite index ledger I.

Next bind the threshold and tail data. Apply Definition 435.10 and Definition 435.9. These force the route from the family window to pass through the displayed M, Q, T rows.

Now pass to the consumers. By Definition 435.48, the $\mathsf{RegSeqRat}^{\uparrow}$ tail reader is exhausted by the rows of U. By Definition 435.57, the $\mathsf{Real}^{\uparrow}$ seal reader factors through the same displayed package.

Finally apply Definition 435.71 and Definition 435.72. Overlaps cannot split the tail ledger, and the seal route has no second public handoff. Hence the displayed rows are exactly the finite-family seal obligation surface. □

Theorem 435.82 (UniformCauchyCriterion finite-family RegSeqRat–Real handoff)**.** *For an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

every $\mathsf{RegSeqRat}^{\uparrow}$*-visible tail read and every* $\mathsf{Real}^{\uparrow}$ *seal read factor through the same shared-window route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The tail side and the seal side may refine the displayed finite ledger, but they cannot fork into separate window families, threshold schedules, dyadic tolerance routes, or completion routes.

Proof. Start with the shared window. Apply Definition 435.3 and Definition 435.10. They identify W and M as the common route before any tail or seal data can be read.

Use Definition 435.9. The $\mathsf{RegSeqRat}^{\uparrow}$-visible tail read is exactly the displayed T row after the dyadic tolerance row Q.

For the seal boundary, apply Definition 435.57. The $\mathsf{Real}^{\uparrow}$ seal read is downstream of that same tail ledger rather than an independent completion route.

Refinements do not alter this handoff. Apply Definition 435.71; any overlap refinement still determines the same tail ledger and the same seal route. The stated shared route follows. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family regseq real handoff`

Theorem 435.83 (UniformCauchyCriterion finite-family nonescape ledger)**.** *Let* $U = (I, W, M, Q, T, E, H, C, P, N$ *be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface of Definition 435.81. A consumer of this surface cannot inspect a selected limit, quotient stream, host real equality, ambient completeness witness, or any unledgered completion datum.*

All admissible reads are exhausted by the displayed window, threshold, tolerance, tail, seal, transport, continuation, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family seal nonescape`

Proof. Apply Definition 435.48. The RegSeqRat-facing read has no coordinate outside $I, W, M, Q, T, E, H, C, P, N$.

Apply Definition 435.57. The Real-facing read also factors through the same displayed package, so it cannot introduce a selected limit or ambient completion witness.

Use Definition 435.71 to rule out escape through refined windows. The overlap still determines the same tail ledger and seal route.

Finally use Definition 435.72. A second route to host equality, quotient streams, or completion data would require another visible seal handoff, and that handoff is absent from the accepted packet. □

Theorem 435.84 (UniformCauchyCriterion finite-family window seal exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. Every finite-window consumer route from* U *to a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail or a* $\mathsf{Real}^{\uparrow}$ *seal is exhausted by the displayed rows* $I, W, M, Q, T, E, H, C, P, N$*.*

In particular, no admissible route may bypass the shared window family W *or the tail ledger* T *to reach a selected limit, quotient stream, host real equality, ambient completeness witness, or detached tail oracle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family window seal exhaustion`

Proof. First read the finite-family seal obligation package of Definition 435.81. It lists $I, W, M, Q, T, E, H, C, P, N$ as the complete surface available to seal-facing consumers.

Next use the RegSeqRat–Real handoff of Definition 435.82. It forces both the regular-rational tail side and the Real seal side through the route from I to W, then through M, Q, T, E.

The nonescape ledger of Definition 435.83 removes selected limits, quotient streams, host equality, ambient completeness, and detached tail oracles from the consumer surface.

Finally apply the root seal package of Definition 435.97. It preserves the same finite rows under root-facing overlap refinement, so a route omitting W or T has no remaining public path to a tail or seal consumer. □

Theorem 435.85 (UniformCauchyCriterion shared seal-route determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. Suppose two* $\mathsf{RegSeqRat}^{\uparrow}$*-visible or* $\mathsf{Real}^{\uparrow}$*-visible consumers preserve the displayed finite index ledger* I*, shared window family* W*, shared threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. Then the two consumers determine the same route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

No consumer may split that route into a second $\mathsf{StreamName}^{\uparrow}$ *family,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row,* $\mathsf{Real}^{\uparrow}$ *seal row, quotient stream, host real equality, selected limit, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family shared seal route determinacy`

Proof. Start from the finite-family seal obligation package. By Definition 435.81, every admissible tail or seal consumer sees exactly the rows $I, W, M, Q, T, E, H, C, P, N$ and no completion-facing coordinate outside them.

Next apply the RegSeqRat–Real handoff. Definition 435.82 makes both $\mathsf{RegSeqRat}^{\uparrow}$ tail reads and $\mathsf{Real}^{\uparrow}$ seal reads factor through the same ordered route from I to E.

The nonescape and exhaustion results remove the possible branches. Definition 435.83 excludes selected limits, quotient streams, host real equality, ambient completeness, and detached tail data, while Definition 435.84 says that finite-window consumers cannot bypass W or T.

Thus two consumers preserving the displayed ledger rows have no coordinate at which to diverge. The shared route is determined as $I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$, with all repacking confined to H, C, P, N. □

Theorem 435.86 (UniformCauchyCriterion finite-family seal overlap determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. If two finite-family overlap refinements preserve the displayed rows* $I, W, M, Q, T, E, H, C, P, N$*, then both refinements determine the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row and* $\mathsf{Real}^{\uparrow}$ *seal handoff. Neither refinement can introduce a second* $\mathsf{StreamName}^{\uparrow}$ *family, detached tail oracle, selected limit, quotient stream, host real equality, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family seal overlap determinacy`

Proof. Start with the finite-family seal obligation package. By Definition 435.81, the rows $I, W, M, Q, T, E, H, C, P,$ are the complete seal-facing inventory for the accepted packet.

Next use the RegSeqRat–Real handoff. Definition 435.82 routes every tail read and every seal read through the same path from the finite index ledger to the seal handoff.

The shared seal-route determinacy theorem now compares the two refinements. Since both refinements preserve the displayed rows, Definition 435.85 leaves them with the same route from I through E and with repacking only along H, C, P, N.

Finally apply root seal pullback non-escape. Definition 435.98 removes selected limits, quotient streams, host real equality, detached tail oracles, and ambient completeness witnesses from the completion-facing surface. Thus the two overlap refinements determine the same tail row and seal handoff, with no second public route. □

Theorem 435.87 (UniformCauchyCriterion finite-family endpoint budget reflector)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal surface. Every finite-family endpoint request for* U *reflects through the displayed route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Consequently the $\mathsf{RegSeqRat}^{\uparrow}$ *tail reads and* $\mathsf{Real}^{\uparrow}$ *seal reads are the same budgeted endpoint route, and no request can split into a second* $\mathsf{StreamName}^{\uparrow}$ *family, selected limit, quotient stream, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family endpoint budget reflector`

Proof. First assemble the endpoint surface from the finite-family seal obligation package. Theorem 435.81 lists the rows $I, W, M, Q, T, E, H, C, P, N$ as the complete inventory available to a tail or seal consumer.

Next apply the RegSeqRat–Real handoff. Theorem 435.82 identifies the $\mathsf{RegSeqRat}^{\uparrow}$ tail side and the $\mathsf{Real}^{\uparrow}$ seal side with the same route from the finite index ledger through the shared window, threshold, tolerance, tail, and seal rows.

Now use the nonescape ledger and window seal exhaustion. Theorems 435.83 and 435.84 remove selected limits, quotient streams, host real equality, ambient completeness witnesses, detached tail oracles, and any bypass of the shared window or tail ledger.

Finally compose shared seal-route determinacy with seal overlap determinacy. Theorem 435.85 fixes the public route once the displayed rows are preserved, and Definition 435.86 prevents overlap refinements from adding a second stream family or seal handoff. Hence every finite-family endpoint request reflects through the single budgeted route displayed above. □

Theorem 435.88 (UniformCauchyCriterion finite-family route lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. Any* $\mathsf{RegSeqRat}^{\uparrow}$*-visible or* $\mathsf{Real}^{\uparrow}$*-visible consumer preserving* $I, W, M, Q, T, E, H, C, P, N$ *must use the single handoff route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The consumer cannot introduce a second $\mathsf{StreamName}^{\uparrow}$ *family, detached tail oracle, selected limit, quotient stream, host real equality, or ambient completeness datum.*

Proof. Start with the finite-family seal package and RegSeqRat–Real handoff. Definition 435.81 exposes exactly the rows $I, W, M, Q, T, E, H, C, P, N$, and Definition 435.82 routes both tail and real-facing consumers from the finite index ledger to the seal handoff.

Next apply the nonescape ledger and window seal exhaustion. Definition 435.83 removes selected limits, quotient streams, host real equality, ambient completeness, and detached tail data, while Definition 435.84 prevents a consumer from bypassing the shared window and tail ledger.

Now use shared seal-route determinacy together with seal overlap determinacy. Definition 435.85 fixes the route from I through E, and Definition 435.86 shows that overlap refinements preserving the displayed rows cannot split the tail row or the seal handoff.

Therefore every $\mathsf{RegSeqRat}^{\uparrow}$-visible or $\mathsf{Real}^{\uparrow}$-visible consumer preserving the displayed packet rows uses the handoff $I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$. The excluded data are not available as public coordinates, so they cannot provide a second route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionFiniteFamilyRouteLock`

Theorem 435.89 (UniformCauchyCriterion finite-family source exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. Any* $\mathsf{RegSeqRat}^{\uparrow}$*-visible or* $\mathsf{Real}^{\uparrow}$*-visible consumer preserving the finite-family seal route has source access exhausted by* $I, W, M, Q, T, E, H, C, P, N$*. It cannot introduce a second source ledger, stream family, dyadic tolerance source, tail oracle, quotient stream, host real equality, selected limit, or ambient completeness witness before the* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *handoff.*

Proof. Begin with the finite-family seal obligation package. It displays the full accepted source inventory $I, W, M, Q, T, E, H, C, P, N$, with I as the finite index ledger, W as the shared window

family, M as the pointwise modulus row, Q as the dyadic tolerance source, T as the tail oracle row, and E as the seal handoff.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ handoff. The handoff reads the same displayed source rows on the regular-rational side and on the real-seal side, so a consumer preserving the route has no independent source coordinate between T and E.

Next use the nonescape ledger and window seal exhaustion. These results remove selected limits, quotient streams, host real equality, ambient completeness witnesses, detached tail oracles, and any bypass of the finite window and tolerance rows from the public packet surface.

Finally combine shared seal-route determinacy with the route lock above. Shared determinacy fixes the row order from I through E, and Definition 435.88 prevents a consumer preserving $I, W, M, Q, T, E, H, C, P, N$ from selecting a second route. Hence the listed rows exhaust all source access available before the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ handoff. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family source exhaustion`

Theorem 435.90 (UniformCauchyCriterion CauchyLimitFunctor terminal handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet carrying the finite-family seal obligation surface. Suppose an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet preserves the same source seal,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealCompletionSealClassifierUp}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *endpoint used by the finite-family* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *handoff for* U*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and Completion consumer of the pair factors through the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

followed by the displayed functor packet rows. The terminal handoff introduces no quotient stream, selected host limit, host real equality, countable-choice row, detached completion object, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket cauchy limit functor terminal handoff`

Proof. Start with the finite-family side. The seal obligation package and the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ handoff, Definition 435.81 and Definition 435.82, place all tail and seal reads of U on the displayed path from I through E.

Next apply nonescape and route locking inside $\mathsf{UniformCauchyCriterion}^{\uparrow}$. Definition 435.83 removes selected limits, quotient streams, host real equality, detached tail data, and ambient completeness from the finite-family surface, while Definition 435.88 prevents a consumer preserving $I, W, M, Q, T, E, H, C, P, N$ from using a second tail or seal route.

Now consume the accepted $\mathsf{CauchyLimitFunctor}^{\uparrow}$ packet. Its NameCert obligations, finite observation exposure, structural transport, and Real-completion consumer boundary are supplied by Definition 1116.3, Definition 1116.5, Definition 1116.6, and Definition 1116.7. These results make the functor packet read only the preserved source seal, stream windows, regular-rational readback, dyadic tolerance ledger, classifier row, terminal endpoint, and structural rows.

Finally compose the two boundaries. The finite-family route supplies the source tail-to-seal handoff, and the functor packet transports exactly that source through its displayed rows. Since both packets exclude quotient stream equality, selected host limits, host real equality, countable choice, detached completion objects, ambient completeness, and second routes, the combined consumer has the single terminal route stated above. □

Theorem 435.91 (UniformCauchyCriterion finite-family real seal tail coprovenance)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The* $\mathsf{RegSeqRat}^{\uparrow}$*-visible tail row and the* $\mathsf{Real}^{\uparrow}$*-visible seal handoff are coprovenant over the same route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Neither side can be repacked with a different $\mathsf{StreamName}^{\uparrow}$ *family, detached tail row, selected limit, quotient stream, host real equality, or ambient completeness route.*

Proof. First read the finite-family seal obligation package. By Definition 435.81, the packet exposes exactly the rows $I, W, M, Q, T, E, H, C, P, N$ to tail-facing and seal-facing consumers.

Next apply the finite-family RegSeqRat–Real handoff. Definition 435.82 puts the $\mathsf{RegSeqRat}^{\uparrow}$ tail read and the $\mathsf{Real}^{\uparrow}$ seal read on the same path from I to E.

The nonescape ledger removes alternate completion data. Definition 435.83 excludes selected limits, quotient streams, host real equality, ambient completeness witnesses, and detached tail oracles from the public surface.

Finally use the shared seal-route determinacy theorem. Definition 435.85 prevents a second $\mathsf{StreamName}^{\uparrow}$ family or completion route from carrying either side. Thus the tail row and seal handoff have the same route provenance. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family real seal tail coprovenance`

Theorem 435.92 (UniformCauchyCriterion finite-family tail-seal coprovenance lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any consumer preserving the finite index ledger* I*, shared window family* W*, threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* NameCert *row* N *must route both the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail read and the* $\mathsf{Real}^{\uparrow}$ *seal read through the displayed path*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

No second $\mathsf{StreamName}^{\uparrow}$ *family, detached tail row, selected limit, quotient stream, host real equality, or ambient completeness route is available to that consumer.*

Proof. First read the finite-family seal obligation package, Definition 435.81. It exposes the rows $I, W, M, Q, T, E, H, C, P, N$ as the full public inventory of the accepted packet, so any preserved consumer surface has no coordinate beyond those displayed rows.

Next apply the finite-family RegSeqRat–Real handoff, Definition 435.82. The handoff places the $\mathsf{RegSeqRat}^{\uparrow}$ tail read and the $\mathsf{Real}^{\uparrow}$ seal read on the same route from I through E, with H, C, P, N carrying the transports, continuation routes, provenance, and local NameCert row.

The exclusions come from the nonescape ledger and window seal exhaustion, Definition 435.83 and Definition 435.84. These results remove selected limits, quotient streams, host real equality, ambient completeness routes, detached tail rows, and bypasses of the shared window or tail ledger from the packet surface.

Finally compose shared seal-route determinacy, seal overlap determinacy, and the finite-family route lock: Definition 435.85, Definition 435.86, and Definition 435.88. Determinacy fixes the displayed path for any consumer preserving $I, W, M, Q, T, E, H, C, P, N$, overlap determinacy prevents a repacking of the tail row or seal handoff, and route lock rules out a second completion-facing path. Thus the tail read and Real seal read have the same forced endpoint route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family tail seal coprovenance lock`

Theorem 435.93 (UniformCauchyCriterion finite-family tail cofinal seal reflector)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any tail-cofinal request that preserves the displayed finite index ledger I, shared window family W, shared threshold schedule M, dyadic tolerance row Q, tail ledger T, seal handoff E, componentwise* hsame *transports H, displayed* Cont *routes C, package provenance P, and local* $\mathsf{NameCert}$ *row N reflects through the same route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The request cannot introduce a second $\mathsf{StreamName}^{\uparrow}$ *family, detached tail oracle, selected limit, quotient stream, host real equality, ambient completeness datum, or completion-seal budget path outside the displayed packet.*

Proof. First read the finite-family seal obligation package and the RegSeqRat–Real handoff. The package exposes exactly the rows $I, W, M, Q, T, E, H, C, P, N$, while the handoff places every $\mathsf{RegSeqRat}^{\uparrow}$ tail read and every $\mathsf{Real}^{\uparrow}$ seal read on the route from the finite index ledger to the seal handoff.

Next apply the nonescape ledger. It removes selected limits, quotient streams, host real equality, ambient completeness witnesses, detached tail oracles, and any completion datum not carried by the displayed packet.

Now use shared seal-route determinacy and seal overlap determinacy. A tail-cofinal request preserving the displayed rows has no coordinate at which to split the $\mathsf{StreamName}^{\uparrow}$ family, tail ledger, or seal handoff from the accepted route.

Finally compose the endpoint budget reflector with the route lock and the real seal tail coprovenance theorem. The endpoint reflector binds the budgeted tail and seal endpoints, route lock fixes the ordered path $I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$, and coprovenance keeps the tail row and Real seal handoff on that same path. Thus the request reflects through the single displayed completion-seal budget path and no second route is available. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family tail cofinal seal reflector`

Theorem 435.94 (UniformCauchyCriterion RegSeqRat–Real window budget compatibility)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any finite-window budget refinement preserving $I, W, M, Q, T, E, H, C, P, N$ preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail read and the* $\mathsf{Real}^{\uparrow}$ *seal read simultaneously along*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

A refinement that changes one of these reads without the other is not an accepted refinement of the packet U.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseq real window budget compatibility`

Proof. Start with prefix and window stability. Definition 435.3 keeps the finite read on the displayed window family W, and Definition 435.88 fixes the only handoff route from I through E for consumers preserving the packet rows.

Next compare refinements by the finite-family seal overlap determinacy theorem. Definition 435.86 says that refinements preserving $I, W, M, Q, T, E, H, C, P, N$ determine the same tail row and the same seal handoff.

It remains to exclude hidden completion pullback. Apply Definition 435.98. No selected limit, quotient stream, host real equality, detached tail oracle, or ambient completeness witness is available as a separate budget coordinate.

Therefore a finite-window budget refinement that preserves the accepted packet preserves both visible reads at once. Changing only the tail read or only the seal read would introduce a route outside $I, W, M, Q, T, E, H, C, P, N$, so it is not an accepted refinement of U. □

Theorem 435.95 (UniformCauchyCriterion finite-family refinement handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any root-threshold use of U and any finite-family overlap refinement preserving the displayed rows $I, W, M, Q, T, E, H, C, P, N$ determine the same route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The handoff contains no selected limit, quotient stream, host real equality, detached tail oracle, or ambient completeness datum.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family refinement handoff`

Proof. First apply the finite-family seal overlap determinacy theorem, Definition 435.86. A refinement preserving the displayed rows cannot split the tail row or the seal handoff.

Next use Definition 435.96. It fixes every root-threshold use at the threshold schedule, tolerance ledger, seal non-escape row, and shared-tail meet compatibility, all inside the same packet U.

Now compose with Definition 435.97. The root seal package is exactly the route from I through E together with H, C, P, N.

Finally apply Definition 435.98. It removes selected limits, quotient streams, host real equality, detached tail oracles, and ambient completeness data from the completion-facing boundary, so the refinement and the root-threshold use have the same handoff route. □

Theorem 435.96 (UniformCauchyCriterion root-threshold dependency inventory)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The finite-family root seal package may use U only after the root-threshold refinement surface has fixed the following four dependencies: threshold schedule determinacy at M, tolerance ledger exactness at Q, T, seal non-escape at E, and shared-tail meet compatibility with* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$.

Thus every root seal use is routed through the displayed rows $I, W, M, Q, T, E, H, C, P, N$, and it introduces no selected limit, quotient stream, host real equality, or ambient completeness datum.

Proof. First use Definition 435.117. It fixes the threshold schedule row M reached from the finite index ledger and shared window rows.

Next use Definition 435.118. The downstream dyadic tolerance and tail comparison data are exactly the displayed Q, T ledger.

Then apply Definition 435.119. The seal consumer reaches E only through the finite root route and cannot export completion-facing data outside the packet.

Finally use Definition 435.120. Its compatibility with chapter 560 keeps the shared-tail meet on the same displayed root-threshold surface. The four dependencies therefore bind the root seal package without adding a selected limit, quotient stream, host real equality, or ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root threshold dependency inventory`

Theorem 435.97 (UniformCauchyCriterion finite-family root seal package)**.** *For an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

the finite-family root seal package is exactly the route

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

together with the componentwise hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local NameCert row* N*. Under overlap refinement the package preserves the same finite index ledger, shared window family, threshold schedule, tolerance row, tail ledger, and seal handoff, and it exports no selected limit, quotient stream, host real equality, ambient completeness witness, or detached tail oracle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family root seal package`

Proof. First use the finite-family seal obligation package. It lists the rows $I, W, M, Q, T, E, H, C, P, N$ as the complete seal-facing inventory for the accepted packet.

Next apply the finite-family RegSeqRat–Real handoff and the nonescape ledger. They identify the route from the finite index ledger through the shared window, threshold, tolerance, tail, and seal rows, and they exclude completion data outside the displayed package.

Finally use Definition 435.96. It binds the exact root-threshold dependency package, so overlap refinement preserves the same route and the finite-family package is stable as the root seal package. □

↪ variant `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root budget package`

Theorem 435.98 (UniformCauchyCriterion finite-family root seal pullback non-escape)**.** *Any completion-facing consumer of the finite-family root seal package for an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

must factor through the displayed rows $I, W, M, Q, T, E, H, C, P, N$ *and through the root-threshold dependency inventory. Therefore overlap refinement cannot introduce a selected limit, quotient stream, host real equality, detached tail oracle, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family root seal pullback nonescape`

Proof. Start with the finite-family root seal package of Definition 435.97. It identifies the public root seal route as

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

together with H, C, P, N.

Next apply Definition 435.96. It fixes the dependency inventory at threshold schedule determinacy, tolerance ledger exactness, seal non-escape, and shared-tail meet compatibility. Hence a completion-facing consumer reaches the seal only through the same displayed rows and the same root-threshold dependencies.

The finite-family RegSeqRat–Real handoff of Definition 435.82 keeps the tail side and the seal side on the shared route from I through E. The nonescape ledger of Definition 435.83 removes selected limits, quotient streams, host real equality, ambient completeness, and detached tail oracles from the consumer surface.

It remains to check overlap refinement. The root seal package already states that overlap refinement preserves the same finite index ledger, shared window family, threshold schedule, tolerance row, tail ledger, and seal handoff. Since the nonescape ledger removes every completion-facing coordinate outside those rows, refinement has no place to introduce the excluded objects. □

Theorem 435.99 (UniformCauchyCriterion root tail seal route totality)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every root-facing route from the finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *surface to either a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail read or a* $\mathsf{Real}^{\uparrow}$ *seal read factors through*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

and through the structural rows H, C, P, N*. No such route may introduce a selected limit, quotient stream, host real equality, detached tail oracle, ambient completeness witness, or second stream family.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root tail seal route totality`

Proof. Start with the finite-family root seal package of Definition 435.97. It identifies the only public root-facing spine from the finite index ledger through the shared window, threshold schedule, tolerance ledger, tail row, and seal row, with H, C, P, N carried as the structural rows.

Next apply the finite-family RegSeqRat–Real handoff, the finite-family nonescape ledger, and shared seal route determinacy. The handoff ties the $\mathsf{RegSeqRat}^{\uparrow}$ tail read and the $\mathsf{Real}^{\uparrow}$ seal read to the same finite spine, while the nonescape and determinacy rows remove any alternate completion-facing route after the shared window and tail rows have been fixed.

Finally use the root-threshold dependency inventory and Definition 435.98. The inventory fixes the dependencies used by the root seal route, and pullback non-escape prevents overlap refinement from adding selected limits, quotient streams, host real equality, detached tail oracles, ambient completeness witnesses, or a second stream family. Thus every root-facing tail or seal read is total over the displayed route and structural rows. □

Theorem 435.100 (UniformCauchyCriterion terminal Real-route certificate)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any Phase Real and Completion consumer that preserves the finite index ledger, shared window, threshold schedule, dyadic tolerance, tail ledger, seal handoff, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row reaches the terminal Real-facing route only through the ordered finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Such a consumer cannot extract a selected limit, quotient stream, host real equality, ambient completeness witness, detached tail oracle, or second $\mathsf{StreamName}^{\uparrow}$ *family.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal real route certificate`

Proof. First compose the finite-family seal obligation package with the finite-family RegSeqRat–Real handoff. The package theorem fixes the accepted inventory $I, W, M, Q, T, E, H, C, P, N$, and the handoff routes the Real-facing use through the same finite index ledger, shared window, threshold schedule, tolerance row, tail ledger, and seal handoff.

Next apply the window seal exhaustion and shared seal route determinacy theorems. They leave no alternate seal entry after the shared window and tail rows have been chosen, so a Phase Real or Completion consumer preserving H, C, P, N must stay on the ordered route from I to E.

Finally apply the finite-family nonescape ledger and the root seal pullback non-escape theorem. The nonescape ledger removes selected limits, quotient streams, host real equality, ambient completeness witnesses, and detached tail oracles from the consumer surface; the pullback theorem also prevents overlap refinement from adding any completion-facing coordinate outside the displayed rows. A second $\mathsf{StreamName}^{\uparrow}$ family would be another public route to the same seal, contradicting shared seal route determinacy, so the terminal certificate has exactly the displayed Real-facing route. □

Theorem 435.101 (UniformCauchyCriterion terminal seal classifier route)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any terminal Real-facing consumer preserving the finite index ledger* I*, shared window* W*, threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* NameCert *row* N *reaches the same public seal classifier exactly through the ordered route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *tail classifier,* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, and* $\mathsf{Real}^{\uparrow}$ *finite-window read surface are corresponding projections of that route, not separate completion data.*

Proof. First compose the finite-family seal package with the root tail seal route totality theorem. The package fixes $I, W, M, Q, T, E, H, C, P, N$ as the accepted packet inventory, while totality says every root-facing tail or seal read factors through the displayed route and through the structural rows H, C, P, N.

Next apply the finite-family RegSeqRat–Real handoff, Definition 38.276, Definition 38.278, and Definition 38.342. These results identify the $\mathsf{RegSeqRat}^{\uparrow}$ tail side, the $\mathsf{StreamName}^{\uparrow}$ finite-window side, and the $\mathsf{Real}^{\uparrow}$ finite-window read surface as the same finite-window classifier route seen from different interfaces.

Finally use shared seal-route determinacy. Once I, W, M, Q, T, E and the structural rows H, C, P, N are preserved, there is no second public seal handoff through which a terminal Real-facing consumer could obtain an independent classifier. Hence the three displayed classifier views are projections of the ordered route rather than additional completion data. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal seal classifier route`

Theorem 435.102 (UniformCauchyCriterion terminal RegSeqRat–Real classifier correspondence)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. For any terminal consumer preserving the finite index ledger* I*, shared window* W*, threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N*, the following three classifier views determine one another: the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail classifier, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, and the* $\mathsf{Real}^{\uparrow}$ *finite-window seal classifier. The determination is bidirectional because each view is exactly the corresponding projection of the ordered route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Thus a read of any one of the three classifiers is a read of the same terminal seal classifier through its own interface projection, and none of the three views imports a selected limit, quotient stream, host real equality, ambient completeness witness, detached tail oracle, or second stream family.

Proof. First compose the finite-family seal package with the finite-family RegSeqRat–Real handoff and the root tail seal route totality theorem. These results put every accepted packet on the common route from I to E, carry H, C, P, N as the structural rows, and make every root-facing tail or seal read factor through that route.

Next apply Definition 38.276, Definition 38.278, and Definition 38.342. They identify the RegSeqRat tail side, the StreamName finite-window side, and the Real finite-window read surface as interface projections of the same finite-window classifier.

For the forward direction, a RegSeqRat tail-classifier read fixes the route through I, W, M, Q, T, E. The StreamName and Real classifier reads are then the finite-window and seal projections supplied by the scoped interface and finite-window Real-seal interface.

For the reverse direction, a StreamName or Real classifier read reaches the same route through the finite-window read surface and the RegSeqRat–Real handoff. Root tail seal totality sends the resulting terminal read back to the RegSeqRat tail classifier, with no extra completion coordinate.

Finally use shared seal-route determinacy together with the nonescape ledger. Determinacy rules out a second public seal handoff once $I, W, M, Q, T, E, H, C, P, N$ are preserved, and the ledger excludes selected limits, quotient streams, host real equality, ambient completeness, detached tail oracles, and second stream families. Hence the three classifiers correspond bidirectionally as projections of the displayed route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal regseq real classifier correspondence`

Theorem 435.103 (UniformCauchyCriterion terminal quotient-refusal boundary)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any terminal consumer preserving the finite index ledger* I*, shared window* W*, threshold schedule* M*, dyadic tolerance row* Q*, tail ledger* T*, seal handoff* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N *remains confined to the route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

when it reads one of the $\mathsf{RegSeqRat}^{\uparrow}$ *tail classifier, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, or the* $\mathsf{Real}^{\uparrow}$ *finite-window seal classifier. If the consumer asks instead for quotient-stream equality, selected limits, host real equality, ambient completeness, a detached tail oracle, or a second* $\mathsf{StreamName}^{\uparrow}$ *family, that request is not a coordinate of* U *and must factor through chapter 998 by Definition 998.3 and Definition 998.11.*

Proof. Start with Definition 435.102. It identifies the $\mathsf{RegSeqRat}^{\uparrow}$ tail classifier, $\mathsf{StreamName}^{\uparrow}$ finite-window classifier, and $\mathsf{Real}^{\uparrow}$ finite-window seal classifier as mutually determining projections of the same ordered route from I through E.

Next apply Definition 435.101. Once the terminal consumer preserves $I, W, M, Q, T, E, H, C, P, N$, the classifier route has no second public seal handoff and no independent completion coordinate.

The refusal boundary inside $\mathsf{UniformCauchyCriterion}^{\uparrow}$ is supplied by Definition 435.83 and Definition 435.88. These theorems keep the accepted packet on the finite-family route and exclude selected limits, quotient streams, host real equality, detached tail oracles, ambient completeness, and second stream families from the packet surface.

It remains to route a quotient-facing request. Since none of those requests is a coordinate of U, the consumer must leave the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet and enter the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ surface of chapter 998. There Definition 998.3 sends the Real-completion request to the displayed finite route or the refusal ledger, and Definition 998.11 records that off-route quotient-style requests export no completion coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal quotient refusal boundary`

Theorem 435.104 (UniformCauchyCriterion finite-observation terminal synchronization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let*

$$F = (B, S, W', D, R, E', H', C', P', N')$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Suppose the shared-window, tolerance-ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer, and* $\mathsf{Real}^{\uparrow}$ *seal rows are identified by displayed componentwise transports*

$$W\,\mathsf{hsame}\,W', \qquad Q\,\mathsf{hsame}\,D, \qquad T\,\mathsf{hsame}\,R, \qquad E\,\mathsf{hsame}\,E'.$$

Then the terminal $\mathsf{Real}^{\uparrow}$ *route of* U *is exactly the finite selector route*

$$B \to S \to W' \to D \to R \to E'.$$

Equivalently, after the displayed transports, the route

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

is the same terminal handoff as the selector route through B, S, W', D, R, E'*. No quotient stream, host equality, countable choice, hidden selector, selected limit, host metric, or ambient completeness coordinate can mediate the synchronization.*

Proof. First use the terminal certificate, seal classifier route, and RegSeqRat–Real classifier correspondence for $\mathsf{UniformCauchyCriterion}^{\uparrow}$. By Definition 435.100, Definition 435.101, and Definition 435.102, any terminal Real-facing read of U is confined to the ordered route from I through E and sees the RegSeqRat, finite-window, and Real classifier faces as projections of that route.

Next apply the terminal quotient-refusal boundary, Definition 435.103. It removes quotient streams, selected limits, host equality, ambient completeness, detached tail oracles, and second stream-family data from the synchronized consumer surface.

Now use the finite-observation pullback Definition 782.20. The displayed transports identify the shared window, dyadic tolerance ledger, regular readback, and Real seal rows of U with the rows W', D, R, E' of F, so the UniformCauchyCriterion terminal read is pulled back to the finite selector route through the same visible rows.

Finally apply terminal route no-branching for the selector, Definition 782.28. Once the rows $B, S, W', D, R, E', H', C', P', N'$ and the synchronized UniformCauchyCriterion rows are fixed, there is no second terminal route through hidden selector, quotient, choice, host-metric, selected-limit, or completeness data. Hence the terminal Real route of U is exactly the finite selector route $B \to S \to W' \to D \to R \to E'$. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite observation terminal synchronization`

Theorem 435.105 (Uniform Cauchy criterion Real-window terminal route)**.** *Every accepted* UniformCauchyCriterio *terminal* $\mathsf{Real}^{\uparrow}$ *route that consumes a displayed* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet and a displayed* RealCompletionExactBou *packet factors through the same finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *selector-budget, seal-classifier, witness-ledger, transport,* Cont, Pkg, *and* NameCert *rows. The consumer receives no quotient stream, selected limit, host real equality, ambient completeness, metric-completeness, or choice row.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket realwindowbudget terminal route`

Proof. Start with the local terminal route for $\mathsf{UniformCauchyCriterion}^{\uparrow}$. The terminal Real-route certificate of Definition 435.100 and the terminal seal classifier route of Definition 435.101 confine every accepted terminal read to the displayed finite route from I through E, with H, C, P, N as the structural rows.

Next consume the $\mathsf{RealWindowBudget}^{\uparrow}$ packet. Its finite-route visibility theorem, Definition 1164.5, exposes the finite window, dyadic readback, regular handoff, Real seal, selector-budget, disclosure, transport, replay, provenance, and local naming rows. The Real-completion handoff of Definition 1164.4 keeps Real and Completion consumers on that same finite route rather than allowing quotient streams, host equality, selected limits, or ambient completeness.

Now attach the RealCompletionExactBoundaryUp packet. Four-face exactness, Definition 1120.6, identifies the StreamName, RegSeqRat, DyadicRatCore, and terminal Real faces as the only completion faces after the seal, classifier, witness-ledger, and budget rows. The terminal-route pullback and non-escape results, Definition 1120.9 and Definition 1120.3, prevent the boundary from

adding quotient streams, selected limits, host real equality, choice, ambient completeness, metric completeness, or a second terminal route.

Combining the three packet surfaces gives one displayed terminal route. The UniformCauchyCriterion rows supply the accepted terminal classifier route, the RealWindowBudget rows expose the finite observation budget used by that route, and the RealCompletionExactBoundary rows certify the four completion faces without extra completion data. Hence the cross-chapter consumer factors through exactly the finite rows listed in the statement. □

Theorem 435.106 (UniformCauchyCriterion finite-family terminal budget exhaustion)**.** *For any accepted finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, the terminal* $\mathsf{Real}^{\uparrow}$ *route, terminal seal-classifier route, terminal* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *classifier correspondence, finite-observation terminal synchronization, terminal quotient-refusal boundary, and* $\mathsf{RealWindowBudget}^{\uparrow}$ *terminal route exhaust every public terminal budget read. No public terminal consumer can export an additional host quotient, unbudgeted tail choice, selected limit, ambient completion witness, host real equality, or second stream-family route outside the displayed finite packet rows.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family terminal budget exhaustion`

Proof. Start with the terminal Real-route certificate and the terminal seal classifier route. By Definition 435.100 and Definition 435.101, every terminal Real-facing read stays on the finite route from I through E with H, C, P, N as structural rows.

Next use the terminal $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ classifier correspondence and finite-observation terminal synchronization. The correspondence identifies the regular-tail, finite-window, and Real classifier faces as projections of the same route, while Definition 435.104 pulls the terminal read through the displayed selector-budget rows rather than through hidden selector or completion data.

Now apply the terminal quotient-refusal boundary and the $\mathsf{RealWindowBudget}^{\uparrow}$ terminal route. The boundary sends quotient-style requests away from the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet surface, and Definition 435.105 confines budgeted Real and Completion reads to the finite StreamName, dyadic, regular, Real, selector-budget, seal-classifier, witness-ledger, transport, replay, provenance, and local NameCert rows.

These rows are finite projections of the same accepted packet surface. Since the cited terminal routes already account for the Real route, classifier route, regular-tail correspondence, finite-observation synchronization, quotient refusal, and Real-window budget, there is no missing public budget row into which a host quotient, unbudgeted tail choice, external completion witness, or second stream family could be inserted. □

Theorem 435.107 (Uniform Cauchy criterion finite-family budget seal composition)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{RealWindowBudget}^{\uparrow}$ *consumer that preserves the endpoint route of* U*, the* $\mathsf{StreamName}^{\uparrow}$ *four-object exit route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *real-seal handoff, and the* $\mathsf{RealWindowBudget}^{\uparrow}$ *completion handoff factors through the single budgeted seal path*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

It cannot introduce a second $\mathsf{StreamName}^{\uparrow}$ *family, quotient stream, host real equality, selected limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family budget seal composition`

Proof. First apply the finite-family endpoint budget reflector and route lock, Definition 435.87 and Definition 435.88. They identify every endpoint-facing read of U with the displayed budgeted route and prevent a second public seal path through an independent stream family or detached completion coordinate.

Next compose the three external handoff surfaces. The $\mathsf{StreamName}^{\uparrow}$ side is the four-object exit route of Definition 126.500; the regular-rational side is the seal handoff of Definition 301.22; and the Real-window side is the completion handoff of Definition 1164.4. These surfaces read the finite StreamName, RegSeqRat, DyadicRatCore, and Real faces as projections of one displayed BEDC route rather than as independently chosen completion data.

The local nonescape boundary then removes the absent coordinates. By Definition 435.83, selected limits, quotient streams, host real equality, ambient completeness, and detached tail data are not rows of the accepted packet.

Finally use the terminal Real-window route of Definition 435.105. Since the endpoint route, StreamName exit, RegSeqRat seal handoff, and RealWindowBudget completion handoff have all been identified with the same rows $I, W, M, Q, T, E, H, C, P, N$, the consumer has only the single factorization path displayed in the statement. □

Theorem 435.108 (Uniform Cauchy criterion terminal seal envelope)**.** *Every terminal consumer of an accepted finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, and the four-face boundary of* $\mathsf{RealCompletionExactBoundaryUp}$*. The envelope has no coordinate for quotient streams, host equality, selected limits, choice, host metric data, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket terminal seal envelope`

Proof. First use the finite-family budget seal composition and the Real-window terminal route. They identify the terminal route of the criterion with the single displayed path through the finite window, tolerance ledger, regular readback, and Real seal rows.

Next attach the Real-window budget and exact-boundary surfaces. The Real-window handoff supplies the budget packet, while four-face exactness identifies the StreamName, RegSeqRat, DyadicRatCore, and Real faces as the only completion-facing boundary.

Finally use the terminal quotient-refusal and non-escape clauses already cited above. They remove quotient streams, host equality, selected limits, choice, host metric data, and ambient completeness from the consumer surface, so the listed rows form the terminal envelope. □

Theorem 435.109 (Uniform Cauchy criterion completion Real correspondence)**.** *For every accepted finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, every completion-facing read of its terminal* $\mathsf{Real}^{\uparrow}$ *seal is the same read as the four-face* $\mathsf{RealCompletionExactBoundaryUp}$ *route over the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, and structural* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No host equality, quotient stream, selected limit, choice, host metric datum, or ambient completeness proof may mediate this correspondence.*

Proof. Start from Definition 435.108. It gives the completion-facing consumer a single route through the StreamName, RegSeqRat, DyadicRatCore, Real, and budget rows before the exact boundary is read.

Apply the Real-completion boundary pullback and four-face exactness. These results say that the terminal Real face is the fourth face of the same boundary whose earlier faces are the StreamName window, regular readback, and dyadic ledger.

The non-escape clauses then rule out the absent mediators. Since the accepted packet carries only the displayed finite rows and structural support, the completion read and the Real seal read are the same BEDC route. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket completion real correspondence`

Theorem 435.110 (Uniform Cauchy criterion terminal four-face window exactness)**.** *For every accepted finite-family* UniformCauchyCriterion$^{\uparrow}$ *packet, a terminal finite-window read is exact if and only if it factors through the displayed* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* Real$^{\uparrow}$ *seal,* RealWindowBudget$^{\uparrow}$ *packet, and* RealCompletionExactBoundaryUp *four-face boundary. No quotient stream, host equality, selected limit, choice row, host metric datum, ambient completeness proof, or second completion face may mediate the read.*

Proof. The forward direction starts from the terminal seal envelope, Definition 435.108. It supplies the single terminal route through the StreamName window, regular readback, dyadic ledger, Real seal, RealWindowBudget packet, and exact-boundary face.

Use the completion Real correspondence, Definition 435.109, to identify the Real seal with the fourth face of the same exact boundary. The first three faces are the StreamName window, RegSeqRat readback, and DyadicRatCore ledger already present in the envelope.

For the reverse direction, apply finite-window coverage, Definition 435.111. Any read outside the displayed route would ask for one of the refused mediators or a second completion face, so every factorization through the listed four-face route is exactly the terminal finite-window read. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionTerminalFourFaceWindowExactness`

Theorem 435.111 (Uniform Cauchy criterion finite-window coverage)**.** *Every finite-window observation used by a terminal* UniformCauchyCriterion$^{\uparrow}$ *consumer is covered by the same* StreamName$^{\uparrow}$ *window and* RegSeqRat$^{\uparrow}$ *readback that feed the* DyadicRatCore$^{\uparrow}$ *ledger, the* Real$^{\uparrow}$ *seal, the* RealWindowBudget$^{\uparrow}$ *packet, and the* RealCompletionExactBoundaryUp *four-face boundary. A consumer that asks for a quotient stream, host equality, selected limit, choice principle, host metric datum, or ambient completeness row has left the accepted terminal envelope.*

Proof. Use the finite-observation terminal synchronization theorem to pull any finite observation back to the displayed selector route. The synchronization fixes the shared window, dyadic ledger, regular readback, and Real seal rows.

Then use the budget seal composition and terminal envelope theorem. They keep the pulled-back finite observation on the same window-to-seal route consumed by RealWindowBudgetUp and by the exact completion boundary.

The exclusion statement follows from the same quotient-refusal and non-escape boundaries. Those objects are not rows of the finite terminal envelope, so they cannot cover or refine an accepted finite-window observation. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite window coverage`

Theorem 435.112 (Uniform Cauchy criterion finite-family Rat fold handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A finite family of shared-threshold windows transported through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Rat}^{\uparrow}$ *order readback admits a single* BHist *lower-bound fold row consumed by the finite-family seal of* U. *The fold row is read through the same* $I, W, M, Q, T, E, H, C, P, N$ *route and cannot introduce quotient streams, selected host limits, host real equality, detached tail oracles, countable choice, or ambient completeness.*

Proof. First use the finite-family seal obligation package. It supplies the accepted rows $I, W, M, Q, T, E, H, C, P, N$ and identifies the shared-window seal route that every finite-family consumer must preserve.

Next read each threshold through the rational order surface. The $\mathsf{StreamName}^{\uparrow}$ window provides the finite family of windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback supplies regular tail data, the $\mathsf{Real}^{\uparrow}$ seal keeps the terminal face on the displayed route, and the $\mathsf{Rat}^{\uparrow}$ order readback exposes only rational lower-bound comparisons.

Now fold over the finite family. Using the same minimum-selector pattern employed by the compact-uniform-modulus chapters, each rational threshold comparison contributes one displayed lower-bound row, and finite iteration combines those rows into a single BHist lower-bound fold row.

Finally repack the fold row into the shared-window seal route. The route lock, nonescape ledger, and terminal seal envelope keep the fold on $I, W, M, Q, T, E, H, C, P, N$, so the finite-family seal consumes the lower-bound row without importing any absent completion or choice coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite family rat fold handoff`

Theorem 435.113 (UniformCauchyCriterion root tail coverage)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet inside a* BHist *inventory. Every accepted family tail comparison visible to a downstream consumer factors through the finite rows* $I, W, M, Q, T, H, C, P, N$ *and the shared dyadic tolerance row* Q *of* U.

In particular, the comparison is a displayed finite-tail read after the route

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T,$$

with H, C, P, N *supplying only transport, continuation, provenance, and local naming rows. No consumer-visible tail comparison can bypass the shared window family, introduce a second modulus schedule, or detach an unledgered dyadic tolerance.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root tail coverage`

Proof. First bind the threshold route. Apply Definition 435.10: every accepted family read reaches the same threshold schedule M before it can inspect the dyadic tolerance row Q.

Next expose the finite tail ledger. Apply Definition 435.9. The accepted comparison is therefore a displayed row of T over the finite index and window rows I, W.

The structural rows do not add another comparison coordinate. They replay the same BHist route through H, C, P, N, so the consumer sees only the finite tail row attached to the shared dyadic tolerance row. □

Theorem 435.114 (UniformCauchyCriterion root seal readback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet. Every* Real$^{\uparrow}$ *seal read of U factors through the displayed route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The readback surface exposes the finite window, threshold, tolerance, tail, and seal rows only. It cannot expose quotient streams, selected limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root seal readback`

Proof. Start from the real-facing consumer. By Definition 435.57, a Real$^{\uparrow}$ seal read is exhausted by the displayed finite package $I, W, M, Q, T, E, H, C, P, N$.

Apply Definition 435.10 and Definition 435.9. These force the route from the family window to the seal handoff to pass through M, Q, T.

Finally use Definition 435.72. Since the accepted packet has no second seal path, quotient streams, selected limits, host real equality, and ambient completeness have no row through which to appear. □

Theorem 435.115 (UniformCauchyCriterion root downstream consumer surface)**.** *For an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

the root downstream consumer surface for RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, CauchyModulus$^{\uparrow}$, *and* Real$^{\uparrow}$ *is exactly the finite* BHist*-anchored package*

$$I, W, M, Q, T, E, H, C, P, N.$$

The StreamName$^{\uparrow}$ *and* CauchyModulus$^{\uparrow}$ *consumers read the shared family and threshold rows W, M over I; the* DyadicRatCore$^{\uparrow}$ *and* DyadicTailBound$^{\uparrow}$ *consumers read Q, T; the* RegSeqRat$^{\uparrow}$ *consumer reads the resulting uniform tail package; and the* Real$^{\uparrow}$ *consumer reads the seal row E. No downstream chapter may require quotient stream identity, selected family limits, host real equality, ambient completeness, or any row outside this package.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root downstream unblock`

Proof. First assemble the tail-facing surface. Apply Definition 435.113. The window, threshold, dyadic tolerance, and tail comparison data are already contained in the displayed finite rows.

Next pass to regular-rational consumers. By Definition 435.48, the RegSeqRat$^{\uparrow}$ side is exhausted by the same package $I, W, M, Q, T, E, H, C, P, N$.

Now read the real-facing boundary. Apply Definition 435.114. The Real$^{\uparrow}$ seal consumer reaches E only after the route through I, W, M, Q, T.

The remaining listed consumers are projections of these rows: StreamName$^{\uparrow}$ reads W, CauchyModulus$^{\uparrow}$ reads M, DyadicRatCore$^{\uparrow}$ reads Q, and DyadicTailBound$^{\uparrow}$ reads T. Therefore the finite package is the complete downstream surface for the root dependency. □

Theorem 435.116 (UniformCauchyCriterion root interface Lean companion)**.** *The root downstream surface of* UniformCauchyCriterion$^{\uparrow}$ *is machine-backed by the three RootInterfaces rows: root tail coverage, downstream seal interface, and threshold dependency inventory. These rows verify the same finite* BHist/Cont/Pkg^{sig} *interface exposed by the paper root obligations, with no quotient streams, selected limits, host* Real$^{\uparrow}$ *equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root interfaces lean companion`

Proof. Apply Definition 435.113. The tail-facing part of the RootInterfaces package is the same finite window, threshold, dyadic tolerance, and tail-comparison route used by downstream consumers.

Apply Definition 435.114. The seal-facing row reaches $\mathsf{Real}^{\uparrow}$ only through I, W, M, Q, T, E, so it cannot smuggle in quotient streams, selected limits, host $\mathsf{Real}^{\uparrow}$ equality, or ambient completeness.

Apply Definition 435.115. The remaining inventory rows are exactly the public downstream package $I, W, M, Q, T, E, H, C, P, N$. Thus the Lean companion rows verify the same finite $\mathsf{BHist}/\mathsf{Cont}/\mathsf{Pkg}^{\mathrm{sig}}$ interface that the paper surface exports. □

Theorem 435.117 (UniformCauchyCriterion root threshold schedule determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet in a* BHist *inventory. Every downstream threshold read of* U *is forced through the single displayed schedule row* M *after the finite index row* I *and the shared window-family row* W *have been read. No downstream consumer may insert a second threshold schedule, read* M *before* I, W*, or detach a threshold from the route recorded by* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root threshold schedule determinacy`

Proof. First read the root tail surface. By Definition 435.113, every downstream family-tail comparison already factors through the finite route I, W, M, Q, T.

Now apply Definition 435.10. The shared-threshold theorem identifies the schedule read as the same displayed row M for every accepted family window over I, W.

The structural rows only transport and replay that route. Since H, C, P, N do not add another schedule coordinate, any consumer-visible threshold read is the single row M reached after I, W. □

Theorem 435.118 (UniformCauchyCriterion root tolerance ledger exactness)**.** *For an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

every dyadic tolerance or tail comparison read visible at the root is exactly a displayed read of the Q, T *ledger after the schedule row* M*. Thus the dyadic tolerance surface is the finite ledger*

$$M \longmapsto Q \longmapsto T$$

over the already displayed rows I, W*, and it contains no unrecorded tolerance, detached tail bound, quotient-stream comparison, or selected limit.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root tolerance ledger exactness`

Proof. Start from the threshold route supplied by Definition 435.117. The consumer reaches the same row M after I, W before any dyadic tolerance can be inspected.

Apply Definition 435.9. It says that the tolerance and tail evidence is precisely the ledger pair Q, T over that schedule.

The root coverage theorem, Definition 435.113, rules out a second tail source at the same surface. Therefore every root tolerance or tail read is the displayed Q, T ledger and nothing outside it. □

Theorem 435.119 (UniformCauchyCriterion root seal consumer non-escape)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal consumer reaches the seal row* E *only through the route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

The consumer surface cannot read quotient streams, selected limits, host equality, ambient completeness, or any row outside the displayed BHist *package.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root seal consumer nonescape`

Proof. Read the seal through Definition 435.114. The seal-facing surface already factors through the finite rows $I, W, M, Q, T, E, H, C, P, N$.

Next use Definition 435.117 and Definition 435.118. These two root obligations force the path to E through the unique schedule row M and the exact Q, T tolerance ledger.

Finally apply Definition 435.57 and Definition 435.72. The real-facing consumer is exhausted by the displayed route, so quotient streams, selected limits, host equality, and ambient completeness have no coordinate through which to enter. □

Theorem 435.120 (UniformCauchyCriterion root shared tail meet)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose shared window-family schedule and tolerance ledger are consumed by a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing seal read then factors through the same finite rows* $I, W, M, Q, T, E, H, C, P, N$ *and through the sibling shared-tail packet of chapter 560. The factorization does not claim RealUp completeness, quotient-stream equality, selected limits, host equality, or an ambient completeness principle.*

Proof. Start with root threshold schedule determinacy. Definition 435.117 forces every downstream threshold read through the displayed schedule row M after I, W.

Next use tolerance ledger exactness. By Definition 435.118, the dyadic tolerance and tail comparison data consumed after M are exactly the finite Q, T ledger.

Finally apply root seal consumer non-escape, Definition 435.119. The $\mathsf{Real}^{\uparrow}$-facing seal route reaches E only through I, W, M, Q, T and the structural rows H, C, P, N, while chapter 560 consumes the same shared-tail surface as a sibling packet rather than as a quotient or completeness theorem. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root shared tail meet`

Theorem 435.121 (Uniform Cauchy criterion tail-meet root export)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted root UniformCauchyCriterion$^{\uparrow}$ *packet whose family thresholds share a* RegularCauchyTailMeet$^{\uparrow}$ *row. Then the shared tail-meet row is exported through the same finite schedule row M, tolerance ledger Q, T, root seal consumer E, componentwise* hsame *row H, displayed* Cont *route row C, package provenance P, and local* NameCert *row N. The export is therefore a consumer-facing root packet over the sibling source chapter 560, not a quotient-stream comparison, selected limit, host equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail meet root export`

Proof. First read the root schedule. By Definition 435.117, every threshold request reaches the displayed schedule row M after the finite index row I and the shared window-family row W.

Next read the tolerance ledger. By Definition 435.118, the dyadic tolerance and tail comparison data downstream of M are exactly the finite ledger rows Q, T.

Now apply the shared-tail theorem, Definition 435.120. Its hypothesis identifies the same schedule and ledger surface as the one consumed by the RegularCauchyTailMeet$^{\uparrow}$ sibling, and chapter 560 supplies that sibling source.

Finally use root seal consumer non-escape, Definition 435.119. The seal-facing route reaches E only through I, W, M, Q, T and the structural rows H, C, P, N, so the exported tail-meet row remains inside the finite root schedule, tolerance ledger, displayed Cont, Pkg, and local NameCert surface. □

Theorem 435.122 (UniformCauchyCriterion root budget package)**.** *For an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

the schedule determinacy and tolerance ledger exactness assemble the finite budget package over $I, W, M, Q, T, H, C, P, N$. The package records the unique threshold schedule, the exact dyadic tolerance ledger, the displayed transports, continuation routes, provenance, and local naming row, and it admits no detached tail bound, quotient-stream comparison, selected limit, host equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root budget package`

Proof. Begin with Definition 435.117. It forces every threshold read through the row M after I, W.

Apply Definition 435.118. The dyadic tolerance and tail comparison data are exactly the Q, T ledger downstream of that schedule.

The structural rows H, C, P, N only transport, replay, name provenance, and record the local certificate. Thus the finite budget package is the displayed union of the schedule row, tolerance ledger, and structural rows. □

Theorem 435.123 (UniformCauchyCriterion root downstream seal interface)**.** *Any downstream* Real$^{\uparrow}$ *seal consumer of an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet reads the package only through*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Any sibling RegularCauchyTailMeet$^{\uparrow}$ *consumption remains a finite shared-tail packet over the same displayed rows rather than a quotient-stream equality, selected limit, host equality, or completeness principle.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root downstream seal interface`

Proof. First use Definition 435.122. It supplies the finite budget package over $I, W, M, Q, T, H, C, P, N$.

Now apply Definition 435.119. The seal row E is reached only after the route I, W, M, Q, T has been consumed.

Finally use Definition 435.120. The sibling tail-meet consumer reads the same shared finite packet, so downstream seal use and tail-meet use remain inside the displayed interface. □

Theorem 435.124 (UniformCauchyCriterion finite-window Real seal extraction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *seal consumer that also exposes a* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window surface factors through the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

and through the sibling shared-tail packet of chapter 560. The extraction surface may read no quotient stream, selected limit, host equality, ambient completeness, or open-cover compactness row.

Proof. First assemble the finite budget package using Definition 435.122. This fixes the schedule, tolerance ledger, transport, continuation, provenance, and local naming rows as the displayed packet over $I, W, M, Q, T, H, C, P, N$.

Next apply Definition 435.119. The $\mathsf{Real}^{\uparrow}$ seal consumer can reach E only through the route I, W, M, Q, T, E, so quotient streams, selected limits, host equality, and ambient completeness have no readable coordinate on the seal side.

Then use Definition 435.120 and Definition 435.123. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ finite-window surface is the same sibling shared-tail packet recorded by chapter 560, and downstream seal use remains inside that displayed interface.

Thus the result is a finite-window extraction surface for Real-facing consumers. It is not a $\mathsf{Real}^{\uparrow}$ completeness theorem and does not assert a selected limit, quotient-stream equality, host equality, open-cover compactness, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite window real seal extraction`

Theorem 435.125 (UniformCauchyCriterion root budget seal eligibility)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The packet is eligible for a* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal consumer only through the displayed route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

and through the sibling shared-tail packet cited by chapter 560. The eligibility surface includes no quotient stream, selected limit, host equality, open-cover compactness, or ambient completeness coordinate.

Proof. First assemble the finite budget package using Definition 435.122. It fixes the displayed schedule, tolerance ledger, transport, continuation, provenance, and local naming rows over $I, W, M, Q, T, H, C, P, N$

Next apply Definition 435.123. Any downstream seal consumer reaches the seal row only through I, W, M, Q, T, E, while the sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ use remains a finite shared-tail packet.

Finally use Definition 435.124. That theorem identifies the Real-facing extraction route with the same finite-window seal path and the shared-tail packet of chapter 560, and it refuses quotient streams, selected limits, host equality, open-cover compactness, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root budget seal eligibility`

Theorem 435.126 (UniformCauchyCriterion threshold seal budget factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer that exposes* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window data factors through the same displayed finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Here M is the unique threshold schedule certified by the root threshold row, and Q, T are the exact dyadic tolerance ledger rows certified by the root tolerance ledger. The consumer receives no quotient stream, selected limit, host equality, open-cover compactness, or ambient completeness row.

Proof. First use Definition 435.117 and Definition 435.118. These identify the threshold schedule M and the dyadic tolerance rows Q, T as determined finite data, not as choices made by the Real-facing consumer.

Next apply Definition 435.122. It repacks I, W, M, Q, T with the displayed H, C, P, N provenance rows, so the consumer-facing surface is carried through hsame, Cont, Pkg, and local NameCert provenance rather than through an ambient completion principle.

Then combine Definition 435.119, Definition 435.120, and Definition 435.124. The seal row E can only be reached after the finite route, while the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows are the shared tail-meet rows already exposed to Real-facing extraction.

Finally apply Definition 435.125. It is exactly the seal-consumer boundary for $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads. Since none of the listed excluded objects has a row in U or in the shared finite-window package, the consumer receives no quotient stream, selected limit, host equality, open-cover compactness, or ambient completeness data. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket threshold seal budget factorization`

Theorem 435.127 (UniformCauchyCriterion tail-fiber seal compatibility)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose threshold schedule is consumed by a* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet. Then the UniformCauchyCriterion root threshold route and the tail-fiber packet share the same finite route to the* $\mathsf{Real}^{\uparrow}$ *seal: the displayed schedule I, W, M, tolerance ledger Q, T, tail-fiber packet,* hsame *transport H,* Cont *replay C, provenance P, and local* NameCert *row N before the seal row E is read. The compatibility surface is not quotient-stream equality, selected limits, host equality, open-cover compactness, countable choice, or ambient completeness.*

Proof. First use root threshold schedule determinacy and root tolerance ledger exactness from Definition 435.117 and Definition 435.118. These fix I, W, M, Q, T as the finite threshold and tolerance route.

Next apply the root shared tail meet and root budget seal eligibility theorems, Definition 435.120 and Definition 435.125. They place the shared-tail consumer and the Real-facing seal row on the same displayed route.

Finally use the RegularCauchyTailFiber route coverage of Definition 893.10. It identifies the tail-fiber Real-and-Completion route before any Real seal is read, so the UniformCauchyCriterion packet contributes only the schedule, tolerance, transport, replay, provenance, and local naming rows listed above. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail fiber seal compatibility`

Theorem 435.128 (UniformCauchyCriterion Real-completion exit exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any current Phase Real and Completion consumer route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the finite root route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

through the sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, and through the displayed* H, C, P, N *rows. The exit surface exports no quotient stream, selected limit, host equality, countable choice, open-cover compactness, or ambient completeness coordinate.*

Proof. First use Definition 435.117 and Definition 435.118. These fix the threshold schedule and dyadic tolerance ledger as the initial finite root route I, W, M, Q, T.

Next apply Definition 435.124. It places every Real-facing finite-window read at the seal row E after the same route and ties the stream and regular sequence surfaces to the sibling packet of chapter 560.

Finally combine Definition 435.125 with Definition 435.129. The completion consumer receives only the finite route, the sibling shared-tail packet, and the H, C, P, N rows, so quotient streams, selected limits, host equality, countable choice, open-cover compactness, and ambient completeness remain outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root real completion exit exhaustion`

Theorem 435.129 (UniformCauchyCriterion completion-functor budget handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Before any* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *consumer reads* U*, the readable surface factors through the finite budget package over* $I, W, M, Q, T, H, C, P, N$*, then through the sibling shared-tail packet of chapter 560, and only then through the Real seal row* E*. The handoff route exposes no quotient stream, selected limit, host equality, countable choice, or ambient completeness coordinate.*

Proof. First assemble the root budget package. By Definition 435.122, the schedule, tolerance ledger, transport rows, continuation routes, provenance, and local naming row are exactly the finite packet over $I, W, M, Q, T, H, C, P, N$.

Next pass that package through the downstream seal interface. Definition 435.123 places every seal-facing read on the route I, W, M, Q, T, E and keeps the sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ use on the same displayed rows.

Finally apply Definition 435.125. It identifies the permitted seal consumer surface with the displayed finite route and the shared-tail sibling of chapter 560. A $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ consumer therefore receives only that package-and-seal route, with no coordinate for quotient streams, selected limits, host equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket completion functor budget handoff`

Theorem 435.130 (UniformCauchyCriterion regular tail-fusion factorization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose shared tail surface is consumed by a tail-fusion completion functor. The functor-facing read of U must factor through the budget package of Definition 435.122, through the regular shared-tail source of chapter 560, and through the seal eligibility route of Definition 435.125. It is therefore a finite tail-fusion factorization, not a quotient-stream construction, selected-limit operation, host-equality transfer, countable-choice selection, or ambient-completeness claim.*

Proof. Begin with the finite package supplied by Definition 435.122. This fixes the threshold schedule M, the dyadic tolerance ledger Q, T, and the structural rows H, C, P, N before any fusion consumer can inspect the packet.

Now use Definition 435.120 together with chapter 560. The shared tail consumed by the functor is the displayed sibling packet over the same schedule and tolerance ledger, so tail fusion does not create a second stream identity or a selected limit.

Finally apply Definition 435.123 and Definition 435.125. These theorems place the Real-facing read at E after the finite budget route and keep every completion-functor access inside the displayed package. Host equality, countable choice, and ambient completeness remain absent because no row of U carries them. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regular tail fusion factorization`

Theorem 435.131 (UniformCauchyCriterion Real completion budget lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer may read U only after the budget package, the regular shared-tail sibling of chapter 560, and the Real seal row E have been traversed in the displayed finite route. This locks the Real completion surface to $I, W, M, Q, T, E, H, C, P, N$ and does not assert Real completeness, quotient-stream equality, selected limits, host equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion budget lock`

Proof. First use Definition 435.129. It already factors any completion-functor read through the finite budget package, the sibling shared-tail packet, and the seal row E.

Next specialize the consumer to $\mathsf{RealCauchyCompletion}^{\uparrow}$. The specialization changes only the reader name; it adds no carrier row to U and no second path around the schedule, tolerance ledger, transport, continuation, provenance, or local naming rows.

Finally combine Definition 435.130 with Definition 435.125. The Real completion consumer is locked to the finite package and seal route, so any claim of Real completeness, quotient-stream equality, selected limits, host equality, countable choice, or ambient completeness lies outside the accepted packet. □

Theorem 435.132 (UniformCauchyCriterion selector-seal handoff lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer that exposes a selector-seal packet must first traverse the finite budget package of Definition 435.122, the shared-tail sibling of chapter 560, and the* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *route*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E$$

from chapter 1081. The handoff surface refuses quotient-stream equality, selected limits, host equality, countable choice, ambient completeness, and open-cover compactness.

Proof. Begin with the root budget package. By Definition 435.122, the accepted packet exposes only the finite rows $I, W, M, Q, T, H, C, P, N$ before any Real or Completion consumer can read a terminal seal.

Next apply Definition 435.125 and Definition 435.131. These results place the seal-facing read after the same finite budget package and after the shared-tail sibling of chapter 560.

Now use Definition 1081.11. A consumer that exposes a selector-seal packet is exhausted by the concrete route $B, W, R, L, E, H, C, P, N$, with $B \mapsto W \mapsto R \mapsto L \mapsto E$ as the terminal Real and Completion handoff.

The displayed rows are therefore exhausted by the UniformCauchyCriterion root budget, the regular shared-tail sibling, and the selector-seal packet. None of those rows carries quotient-stream equality, selected limits, host equality, countable choice, ambient completeness, or open-cover compactness, so no consumer can export such data from the handoff. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket selector seal handoff lock`

Theorem 435.133 (UniformCauchyCriterion tail-cofinality schedule handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any* $\mathsf{RealCauchyCompletion}^{\uparrow}$*-facing consumer that reads the UniformCauchyCriterion root budget through* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *must factor through the sibling* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *route of chapter 681 before the Real seal row E is consumed. The resulting route exports no quotient stream, selected limit, host equality, countable choice, ambient completeness, or second tail-budget source.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail cofinality schedule handoff`

Proof. Begin with the root budget package. By Definition 435.122, the accepted packet exposes only the finite rows $I, W, M, Q, T, H, C, P, N$ before the seal row can be used.

Next apply Definition 435.131. A $\mathsf{RealCauchyCompletion}^{\uparrow}$-facing reader reaches E only after that finite package and the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ sibling have already fixed the shared-tail surface.

Now pass through the sibling schedule. Definition 681.9 places every completion-facing seal read on the ordered $\mathsf{TailCofinalitySchedule}^{\uparrow}$ route, and Definition 681.10 identifies the shared-tail budget with the same finite window, dyadic ledger, regular readback, and Real seal rows.

The displayed coordinates are exhausted by the package rows of U, the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-tail read, the $\mathsf{TailCofinalitySchedule}^{\uparrow}$ schedule route, and the Real seal row. None of these coordinates carries quotient streams, selected limits, host equality, countable choice, ambient completeness, or another tail-budget source, so no completion-facing consumer can export such data from the handoff. □

Theorem 435.134 (UniformCauchyCriterion common-window exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every Real-facing finite-window consumer of U that also consumes the public budget route of Definition 697.40 factors through the same common window rows $I, W, M, Q, T, E, H, C, P, N$, through the sibling tail-meet packet of chapter 560, and through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows. Such a consumer cannot introduce a second window, a selected host limit, quotient-stream equality, a countable-choice schedule, a hidden tail oracle, host real equality, open-cover compactness, or an ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket common window exhaustion`

Proof. First assemble the root budget. By Definition 435.122, the accepted packet exposes its finite schedule, tolerance ledger, seal row, transport rows, continuation rows, provenance, and local naming row as the displayed surface $I, W, M, Q, T, E, H, C, P, N$.

Next pass to the Real-facing seal route. Definition 435.124 identifies the finite-window extraction with the same root rows and with the sibling tail-meet packet, while Definition 435.131 keeps the $\mathsf{RealCauchyCompletion}^{\uparrow}$ read after that budget package and the seal row E.

Finally compare with the diagonal selector budget. Definition 697.40 and Definition 697.28 route every diagonal Real-seal consumer through the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal before any selector rows are used. The two routes therefore meet on the displayed finite window and tail-meet handoff, leaving no coordinate for a second window, a selected host limit, quotient-stream equality, countable choice, a hidden tail oracle, host equality, open-cover compactness, or ambient completeness. □

Theorem 435.135 (UniformCauchyCriterion root threshold synchronizer readback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose threshold route is consumed by a* $\mathsf{CauchySealBudgetSynchronize}$ *packet. Every synchronizer readback of the root route reads the rows in the order*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The synchronizer rows sit upstream of the $\mathsf{Real}^{\uparrow}$ *seal row* E *as the finite schedule, tolerance, transport, continuation, provenance, and local naming surface of* U*. The readback supplies no quotient stream, selected limit, host equality, countable choice, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root threshold synchronizer readback`

Proof. Begin with Definition 435.117. It forces every threshold request through the single displayed schedule row M after the finite index row I and the shared window-family row W.

Next apply Definition 435.118. The consumer reaches the dyadic tolerance and tail comparison ledger only as the displayed Q, T rows downstream of M.

Now use Definition 435.122. The synchronizer-facing surface is the finite package over $I, W, M, Q, T, H, C, P, N$, with H, C, P, N recording only displayed transports, continuation routes, provenance, and the local naming row.

Finally apply Definition 435.125. It places the $\mathsf{Real}^{\uparrow}$ seal read at E after the threshold and tolerance route, and it keeps any seal-facing or sibling-tail consumption inside the same finite package. Thus the synchronizer readback orders $I, W, M, Q, T, E, H, C, P, N$ without introducing quotient streams, selected limits, host equality, countable choice, or ambient completeness. □

Theorem 435.136 (UniformCauchyCriterion window refinement seal route)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. If a transported window refinement is accepted over the same finite threshold and tolerance route, then the refinement preserves the packet rows and routes the refined window through the seal read before returning to the root readback. The visible route is still only*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N,$$

with the finite BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows carrying the transported refinement. No quotient stream, selected limit, host real equality, countable-choice schedule, or ambient completeness coordinate is created by the refined window.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket window refinement seal route`

Proof. First use the shared-threshold transport rows of the accepted packet. The transport keeps the refinement on the same I, W, M, Q, T surface, so the refined window does not become a second schedule or a second tolerance ledger.

Next apply Definition 435.122. The root package exposes the transported rows only as the finite surface $I, W, M, Q, T, H, C, P, N$, with H, C, P, N recording the displayed history transport, continuation route, provenance, and local name.

Then use Definition 435.125. The seal row E is read after the finite threshold and tolerance package, so the transported refinement reaches the seal through the same root budget.

Finally invoke Definition 435.135. The synchronizer readback returns from the seal-facing route to the ordered root route $I, W, M, Q, T, E, H, C, P, N$. Since every step has used only the displayed rows, the refined window has no coordinate from which to export quotient streams, selected limits, host equality, countable choice, or ambient completeness. □

Theorem 435.137 (UniformCauchyCriterion root synchronizer tail-meet lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet consumed both by a* CauchySealBudgetSynchronizer$^{\uparrow}$ *packet and by a* RegularCauchyTailMeet$^{\uparrow}$ *sibling. Every synchronizer readback of the root threshold route must pass through the same* $I, W, M, Q, T, E, H, C, P, N$ *rows and through the sibling shared-tail packet before any completion-facing consumer can read the seal.*

Consequently the synchronizer is not a second tail source. It locks root threshold readback to the shared-tail meet, the finite budget package, and the seal-eligible route already displayed by U*.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root synchronizer tail meet lock`

Proof. Start with threshold schedule determinacy and tolerance ledger exactness. By Definition 435.117, every threshold request reaches the displayed schedule row M through I, W; by Definition 435.118, the downstream tolerance rows are exactly Q, T.

Next apply the shared-tail meet. Definition 435.120 identifies the tail surface consumed by the RegularCauchyTailMeet$^{\uparrow}$ sibling with the same root schedule and tolerance ledger, so the sibling read cannot supply an alternate tail packet.

Now use the root budget package and budget seal eligibility. Definition 435.122 packages $I, W, M, Q, T, H, C, P, N$ as the finite synchronizer-facing surface, and Definition 435.125 places the seal row E after that package.

Finally invoke Definition 435.135. The synchronizer readback is therefore the same ordered route $I, W, M, Q, T, E, H, C, P, N$ with the shared-tail sibling already fixed before any completion-facing consumer reads it. Quotient streams, selected limits, host equality, countable choice, and ambient completeness remain outside the packet. □

Theorem 435.138 (UniformCauchyCriterion StreamName RegSeqRat Real window lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet consumed by the Phase Real and Completion route. Every such consumption factors through one finite window-lock surface: the* StreamName$^{\uparrow}$ *finite-window rows, the* RegSeqRat$^{\uparrow}$ *readback rows, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* Real$^{\uparrow}$ *seal row* E*, componentwise* hsame*, the displayed* Cont *route, the package row* Pkg*, and the local* NameCert *row.*

Thus the Real-facing consumer reads U *only through the same finite threshold, tolerance, seal, history, continuation, provenance, and local-name rows. It cannot add quotient-stream equality, a selected host limit, countable choice, a second tolerance ledger, or an ambient completeness principle.*

Proof. Start with the root budget package. By Definition 435.122, the accepted packet exposes the finite synchronizer-facing surface $I, W, M, Q, T, H, C, P, N$, where H carries the componentwise hsame transport, C carries the displayed Cont route, P carries the package row, and N carries the local NameCert row.

Next apply Definition 435.120. The StreamName$^{\uparrow}$ finite-window rows and RegSeqRat$^{\uparrow}$ readback rows meet the same shared-tail packet, so the completion route has no second stream source or second readback surface.

Now use Definition 435.124. The $\mathsf{Real}^{\uparrow}$ seal row E is extracted only after the displayed threshold and tolerance ledger; the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data therefore remains the same finite ledger already carried by Q, T.

Finally invoke Definition 435.137. The synchronizer locks the root threshold route to the shared-tail meet before any Phase Real or Completion consumer reads the seal. Since each step stays inside the displayed rows of U, quotient-stream equality, selected host limits, countable choice, and ambient completeness have no coordinate in the window-lock surface. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket streamname regseqrat real window lock`

Theorem 435.139 (UniformCauchyCriterion Real seal frontier lock)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every Phase Real and Completion consumer of* U *factors through the root budget package, the regular shared-tail sibling, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route, and the final* $\mathsf{Real}^{\uparrow}$ *seal row* E*. This is the only completion-facing surface exported by* U*: it exports no quotient-stream equality, selected host limit, host equality, countable choice, second tolerance ledger, open-cover compactness, or ambient completeness coordinate.*

Proof. First use Definition 435.122. It exposes the accepted packet only as the finite root package over $I, W, M, Q, T, H, C, P, N$, with the threshold schedule, dyadic tolerance ledger, transport rows, continuation route, provenance, and local naming row already fixed before any seal-facing read occurs.

Next apply Definition 435.131. A Phase Real or Completion consumer reaches the final seal row E only after the budget package and the regular shared-tail sibling have been traversed, so the consumer cannot bypass the root package by selecting a host limit or a quotient stream.

Now combine Definition 435.137 with Definition 435.138. These results identify the shared-tail meet, the $\mathsf{StreamName}^{\uparrow}$ finite-window rows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ route with the same finite window that ends at the $\mathsf{Real}^{\uparrow}$ seal row E.

The carrier coordinates of U are therefore exhausted by $I, W, M, Q, T, E, H, C, P, N$ and by the displayed sibling routes already attached to those rows. None of those coordinates carries quotient-stream equality, a selected host limit, host equality, countable choice, a second tolerance ledger, open-cover compactness, or ambient completeness, so no Phase Real or Completion consumer can export such data from the packet. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real seal frontier lock`

Theorem 435.140 (UniformCauchyCriterion Real completion finite budget envelope)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every Phase Real and Completion consumer that reads the UniformCauchyCriterion root route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and the final* $\mathsf{Real}^{\uparrow}$ *seal must factor through the concrete* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *selector-seal packet of chapter 1081 before any terminal completion-facing read.*

The finite envelope is the root budget package, the regular shared-tail sibling, the selected StreamName$^{\uparrow}$ *window, the* RegSeqRat$^{\uparrow}$ *readback, the* CauchyLimitSeal$^{\uparrow}$ *handoff, the* RealCompletionSelectorSeal$^{\uparrow}$ *route, and the final* Real$^{\uparrow}$ *seal row. It exports no quotient-stream equality, selected host limit, host equality, countable choice, second tolerance ledger, open-cover compactness, detached selector, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionRealCompletionFiniteBudgetEnvelope`

Proof. Begin with the finite root package. Definition 435.122 displays the rows $I, W, M, Q, T, H, C, P, N$, and Definition 435.131 places every Real completion read after that package, the regular shared-tail sibling, and the seal row E.

Next use Definition 435.138. It identifies the Phase Real and Completion read with the same StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance ledger, and final Real$^{\uparrow}$ seal route. Definition 435.139 then adds the CauchyLimitSeal$^{\uparrow}$ handoff and confirms that this is the whole completion-facing frontier exported by U.

Finally apply Definition 1081.11. The selector-seal packet is terminal for the concrete Real and Completion handoff through the selector budget, selected window, readback, limit seal, terminal Real endpoint, and structural rows. Therefore any terminal completion-facing read of the UniformCauchyCriterion root route factors through that packet, and the packet acts as the terminal concrete handoff rather than as an ambient completeness principle. □

Theorem 435.141 (UniformCauchyCriterion diagonal selector seal pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted UniformCauchyCriterion$^{\uparrow}$ *packet, and let D be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *root-selector read. Suppose the two reads display the same* StreamName$^{\uparrow}$ *window, the same* RegSeqRat$^{\uparrow}$ *readback, the same* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the same* CauchyLimitSeal$^{\uparrow}$ *handoff, and the same terminal* Real$^{\uparrow}$ *seal row. Then every Phase Real and Completion terminal read factors through the same* RealCompletionSelectorSeal$^{\uparrow}$ *selector-seal packet.*

Thus the UniformCauchyCriterion route and the diagonal selector route have one terminal pullback surface. No quotient stream, selected host limit, host equality, countable choice, second tolerance ledger, detached selector, or ambient completeness coordinate can distinguish the two routes.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket diagonal selector seal pullback`

Proof. First apply Definition 435.140. The UniformCauchyCriterion terminal read is forced through the finite root budget package, the selected StreamName$^{\uparrow}$ window, the RegSeqRat$^{\uparrow}$ readback, the DyadicRatCore$^{\uparrow}$ ledger, the CauchyLimitSeal$^{\uparrow}$ handoff, the terminal Real$^{\uparrow}$ row, and the concrete RealCompletionSelectorSeal$^{\uparrow}$ packet.

Next use Definition 646.215. Since U and D share the displayed window, readback, ledger, handoff, and terminal seal rows, the diagonal root-selector read pulls back to the same completion-facing selector surface.

Now apply Definition 646.218. The selector budget is exactly the displayed finite route, so the diagonal read cannot replace the shared rows by a detached selector, a second tolerance ledger, a quotient stream, or a selected host limit.

Finally invoke Definition 1081.11. The RealCompletionSelectorSeal$^{\uparrow}$ packet is terminal for the selector-to-seal handoff, so both routes factor through the same terminal selector-seal packet before

any Phase Real or Completion terminal read is available. Host equality, countable choice, and ambient completeness remain outside that packet. □

Theorem 435.142 (UniformCauchyCriterion selector seal exact pullback)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every Phase Real and Completion consumer that reads the* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *selector-seal row of chapter 1081 from* U *factors first through the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E$$

and then through that sibling selector-seal packet.

The pullback is exact on the displayed schedule, tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *rows, the* Cont *route, the* Pkg *row, and the local* NameCert *row. It therefore exports no quotient-stream equality, selected limit, host equality, countable choice, detached selector, or ambient Real completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket selector seal exact pullback`

Proof. Begin with the finite root package. Definition 435.122 exposes $I, W, M, Q, T, H, C, P, N$ as the only schedule, tolerance, history, continuation, provenance, and local-name rows available to a consumer of U.

Next apply Definition 435.132 and Definition 435.131. These locks place the selector-seal handoff and every Real completion read after the same root budget package and after the seal row E.

Now use Definition 435.141. The diagonal selector route and the UniformCauchyCriterion route share one terminal selector-seal pullback surface whenever the window, readback, tolerance ledger, handoff, and terminal seal rows are the displayed rows of U.

Finally invoke the sibling carrier of chapter 1081, using Definition 1081.11. The sibling consumes exactly the same finite selector-to-seal row, so the pullback preserves the schedule, tolerance ledger, Real seal, hsame, Cont, Pkg, and NameCert rows without introducing quotient-stream equality, selected limits, host equality, countable choice, detached selectors, or ambient Real completeness. □

Theorem 435.143 (UniformCauchyCriterion root threshold seal roundtrip)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *root threshold packet. Suppose a finite Phase Real and Completion read of* U *passes through the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *endpoint seal, the* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *finite budget envelope, the diagonal selector seal pullback, and the selector-seal exact pullback displayed in this chapter.*

Then the route that reads from the UniformCauchyCriterion root threshold into the $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *selector seal and back to the UniformCauchyCriterion root is a finite threshold-seal roundtrip. The outgoing selector route and the returning root route display the same* BHist *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *endpoint row under the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *classifier. No quotient-stream equality, selected host limit, host equality, countable choice, detached selector, second tolerance ledger, open-cover compactness, or ambient completeness coordinate is created by the roundtrip.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket root threshold seal roundtrip`

Proof. First bind the three visible endpoint names. Definition 435.138 identifies the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the terminal $\mathsf{Real}^{\uparrow}$ seal row exported by U as one displayed Phase Real and Completion window.

Next apply Definition 435.140. Any completion-facing read of the root route must pass through the finite root budget package, the regular shared-tail sibling, the selected window, the readback row, the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, the $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ selector-seal packet, and the terminal $\mathsf{Real}^{\uparrow}$ endpoint row. This fixes the finite budget row used by both directions of the route.

Now use Definition 435.141 and Definition 435.142. The diagonal selector route and the selector-seal route pull back to the same displayed selector surface, and exactness preserves the schedule, tolerance ledger, terminal seal, componentwise hsame rows, Cont route, Pkg row, and local NameCert row.

Finally return to the root threshold synchronizer. Definition 435.135 supplies the root readback for the displayed threshold window, and Definition 435.137 makes that readback deterministic on the shared finite tail. Since the outgoing selector read and the returning root read are forced through the same finite rows, they expose the same BHist window, the same $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the same terminal $\mathsf{Real}^{\uparrow}$ endpoint row under the chapter classifier. The excluded host-level and completeness coordinates remain outside the finite packet. □

Theorem 435.144 (UniformCauchyCriterion common-window diagonal-selector coherence)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose its root threshold seal factors through a* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *source schedule and through a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *shared-window route, with the displayed common window, diagonal selector row, and refined modulus row all read from the same finite route of* U*.*

Then the common window, the diagonal selector row, and the refined modulus row determine the same $\mathsf{Real}^{\uparrow}$ *seal budget. The resulting consumer surface exports no additional selector row, no second tolerance ledger, no host-level selected limit, no quotient-stream equality, no countable-choice row, and no ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.RootThresholdAndSealRefinement.UniformCauchyCriterionPack common window diagonal selector coherence`
Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket common window diagonal selector coherence`

Proof. First apply Definition 435.134. The accepted packet exposes the common window, readback, tolerance ledger, terminal seal, componentwise hsame route, Cont route, Pkg row, and local NameCert row as one finite surface.

Next use Definition 435.141. The diagonal selector route can reach the terminal seal only through the displayed selector pullback surface, so it cannot detach a selector from the common window.

Now apply the $\mathsf{CauchyModulusRefinement}^{\uparrow}$ shared-window route, using Definition 885.155 and Definition 885.87. The refined modulus row keeps the selected window, tolerance readback, and $\mathsf{Real}^{\uparrow}$ seal on the same public handoff route.

Finally invoke Definition 435.143. The root threshold route returns to the same UniformCauchyCriterion root rows after the selector-seal handoff. Hence the common window, diagonal

selector row, and refined modulus row determine one $\mathsf{Real}^{\uparrow}$ seal budget, while all detached selector, choice, quotient, selected-limit, and ambient-completeness coordinates remain outside the exported packet. □

Theorem 435.145 (UniformCauchyCriterion current Phase Real exit consumer exhaustion)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *root threshold packet. Every current Phase Real and Completion consumer route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *factors through the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The exported surface is the root budget package, the finite-window Real seal extraction, the selector-seal exact pullback, and the Real completion finite-budget envelope displayed in this chapter. It exports no quotient stream, selected host limit, countable choice, host equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket phase real exit consumer exhaustion`

Proof. First assemble the root budget package. By Definition 435.122, the accepted packet exposes only the finite schedule, tolerance ledger, transport, continuation, provenance, and local naming rows $I, W, M, Q, T, H, C, P, N$ before any Real-facing seal read occurs.

Next apply finite-window Real seal extraction and the StreamName–RegSeqRat window lock. Definition 435.124 and Definition 435.138 place the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal row on the same displayed route ending at E.

Now use the selector-seal exact pullback. Definition 435.142 keeps the selector-facing read on the same schedule, tolerance ledger, terminal seal, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally apply Definition 435.140 and the roundtrip theorem above. Every Phase Real and Completion consumer is exhausted by the finite route through $I, W, M, Q, T, E, H, C, P, N$ and the named sibling selector-seal packet, leaving no coordinate for quotient streams, selected host limits, countable choice, host equality, or ambient completeness. □

Theorem 435.146 (UniformCauchyCriterion StreamName RegSeqRat seal synchronization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *root threshold packet. The* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row used by any current Phase Real and Completion consumer are synchronized by the same finite route* $I, W, M, Q, T, E, H, C, P, N$*. No consumer can replace one of these rows by a second stream source, second regular readback, selected host limit, quotient-stream equality, host equality, countable choice, or ambient completeness coordinate.*

Proof. Start with Definition 435.138. It identifies the $\mathsf{StreamName}^{\uparrow}$ finite-window rows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seal row as one completion-facing window-lock surface.

Next use Definition 435.137. The root synchronizer ties that window-lock surface to the shared finite tail and prevents a second stream or readback source from entering the route.

Finally apply Definition 435.143. The outgoing selector route and returning root route display the same BHist window, the same $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the same terminal $\mathsf{Real}^{\uparrow}$ endpoint row under the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ classifier. The synchronization row used by Definition 435.145 is therefore exactly the displayed finite route $I, W, M, Q, T, E, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket regseqrat streamname seal synchronization`

Theorem 435.147 (UniformCauchyCriterion selector seal synchronization)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *root threshold packet. The* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *route, and terminal* $\mathsf{Real}^{\uparrow}$ *seal are synchronized by the same finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

No consumer may replace the selector-seal row by quotient-stream equality, selected host limit, countable choice, a second tolerance ledger, or ambient completeness.

Proof. Start with the stream and readback rows. Definition 435.146 places the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal on the same route $I, W, M, Q, T, E, H, C, P, N$.

Next attach the selector-seal row. By Definition 435.142, every selector-seal read from U first passes through the same schedule, tolerance ledger, terminal seal, transport, continuation, provenance, and local naming rows.

Now use the finite budget envelope. Definition 435.140 keeps the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ route, and terminal $\mathsf{Real}^{\uparrow}$ endpoint inside the displayed Phase Real and Completion route.

Finally apply Definition 1081.11. The sibling selector-seal packet is terminal for the concrete selector-to-seal handoff, so a consumer has no coordinate from which to substitute quotient-stream equality, selected host limits, countable choice, a second tolerance ledger, or ambient completeness for the selector-seal row. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket selector seal synchronizatio`

Theorem 435.148 (UniformCauchyCriterion Phase Real route iff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A current Phase Real and Completion consumer through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is admissible exactly when it preserves the ordered finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N,$$

including the selector-seal row whenever that row is present. Equivalently, the consumer may read only the root budget package, the finite-window Real seal extraction, the StreamName–RegSeqRat synchronization rows, and the selector-seal synchronization rows displayed for U.

This criterion exports no quotient streams, selected host limits, host equality, countable choice, second tolerance ledgers, open-cover compactness, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket phase real route iff`

Proof. For the forward direction, begin with the accepted packet. Definition 435.122 supplies the root budget package, so the consumer starts from the displayed finite schedule, tolerance ledger, transport, continuation, provenance, and local naming rows.

Next apply Definition 435.124. The Real-facing seal is extracted from the same finite window, with no extra stream carrier or host-limit coordinate.

Now use Definition 435.146. It synchronizes the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal on the ordered route $I, W, M, Q, T, E, H, C, P, N$.

Finally, when the selector-seal row is present, apply Definition 435.147. The selector-facing handoff is therefore part of the same finite route, not an additional consumer coordinate.

For the reverse direction, suppose the consumer preserves exactly the displayed ordered route and the selector-seal row when present. The preserved root rows give the consumer surface accepted by Definition 435.122; the preserved finite Real seal gives the surface accepted by Definition 435.124; and the preserved synchronization rows give the surfaces accepted by Definition 435.146 and Definition 435.147.

These are exactly the current Phase Real and Completion consumer locks already available in the chapter. Since every displayed row is finite and packet-local, the reverse construction has no place to introduce quotient streams, selected host limits, host equality, countable choice, second tolerance ledgers, open-cover compactness, or ambient completeness. □

Theorem 435.149 (UniformCauchyCriterion Phase Real endpoint determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Any current Phase Real and Completion consumer preserving the finite ordered route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *has a unique displayed endpoint row: the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *reached after the same schedule* M*, tolerance ledger* Q, T*, transport* H*, route* C*, provenance* P*, and local name* N*.*

Conversely, preserving that endpoint row together with the selector-seal row is exactly the endpoint component of the admissible Phase Real route of Definition 435.148. The endpoint determination exposes no quotient stream, selected host limit, host equality, countable choice, second tolerance ledger, open-cover compactness, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket phase real endpoint determinacy`

Proof. First read the accepted packet through Definition 435.122. The root budget package fixes the finite schedule, tolerance ledger, transport, continuation route, provenance, and local naming rows available to every consumer.

Next apply Definition 435.124. The terminal $\mathsf{Real}^{\uparrow}$ seal is extracted from the same finite window, so an endpoint consumer has no separate stream quotient, selected limit, or host-equality row from which to choose another endpoint.

Now synchronize the intermediate route. The StreamName–RegSeqRat synchronization of Definition 435.146 and the selector-seal synchronization of Definition 435.147 place the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows on the same displayed packet $I, W, M, Q, T, E, H, C, P, N$.

Finally use Definition 435.148. The forward direction gives uniqueness of the displayed terminal row for every admissible Phase Real and Completion consumer. The reverse direction reads the same endpoint and selector-seal rows already displayed in the packet; it does not invoke a new completion principle, countable choice, open-cover compactness, a second ledger, or ambient completeness. □

Theorem 435.150 (UniformCauchyCriterion RealCompletionExactBoundary handoff)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A current Phase Real and Completion consumer preserving the displayed route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *determines a* $\mathsf{RealCompletionExactBoundaryUp}$ *handoff exactly when the seal, classifier, witness-ledger, synchronizer, StreamName, RegSeqRat, DyadicRatCore, terminal Real, transport, continuation, provenance, and local naming rows are displayed as the finite exact-boundary packet.*

Equivalently, the accepted consumer can pass from the UniformCauchyCriterion route lock to the RealCompletionExactBoundary surface precisely through the finite rows already visible on the Phase Real route. It does not introduce quotient stream equality, a selected host limit, host real equality, countable choice, metric completeness, ambient completeness, or an additional terminal consumer coordinate.

Proof. First use Definition 435.148. It identifies the admissible Phase Real and Completion consumer with preservation of the displayed route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$.

Next apply Definition 435.149. The endpoint row is the terminal $\mathsf{Real}^{\uparrow}$ seal reached through the same schedule, tolerance ledger, transport, continuation route, provenance, and local name, so the consumer has a single terminal Real row to hand forward.

Now read the same endpoint as an exact-boundary consumer. Definition 1120.13 identifies terminal Real consumption with the finite exact-boundary rows, including seal, classifier, witness-ledger, synchronizer, StreamName, RegSeqRat, DyadicRatCore, terminal Real, transport, continuation, provenance, and local naming.

Finally apply Definition 1120.14. Since the exact-boundary packet has no coordinate for quotient streams, selected host limits, host real equality, countable choice, metric completeness, or ambient completeness, the handoff exists exactly at the displayed finite packet and nowhere outside it. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion exact boundary handoff`

Theorem 435.151 (UniformCauchyCriterion overlap roundtrip confluence)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose two refined finite-window reads of* U *overlap on a common tail and each read passes through the diagonal selector row, the selector-seal exact pullback, and the root threshold seal roundtrip displayed in this chapter.*

Then the two reads determine the same finite Phase Real and Completion route

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

In particular, the overlap cannot produce a second selector, a second tolerance ledger, a second selector-seal route, quotient-stream equality, a selected host limit, countable choice, host equality, or an ambient completeness coordinate.

Proof. Start with the overlapping refined windows. By Definition 435.71, the common tail forces both refinements to read the same tail ledger row and the same seal-facing route of U.

Next attach the diagonal selector surface. Definition 435.144 identifies the common window, diagonal selector row, and refined modulus row as one $\mathsf{Real}^{\uparrow}$ seal budget, so neither refined read can detach a selector or tolerance ledger from that budget.

Now pass through the selector-seal interface. Apply Definition 435.147 and Definition 435.143. The outgoing selector route and the returning root route display the same BHist window, the same $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the same terminal $\mathsf{Real}^{\uparrow}$ endpoint row.

Finally use Definition 435.148. Since both refined reads preserve exactly the displayed ordered route and the selector-seal row when present, they are admissible through the same finite Phase Real and Completion route. The excluded quotient, host-limit, choice, host-equality, and ambient-completeness coordinates never appear in the packet-local surface, and hence cannot distinguish the two overlap reads. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket overlap roundtrip confluence`

Theorem 435.152 (UniformCauchyCriterion root threshold exhaustion)**.** *Every accepted root threshold read for a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer is exhausted by the displayed threshold schedule, tolerance ledger, shared-tail meet, selector-seal synchronization, and overlap roundtrip rows already named in this chapter. In packet form, the read is the finite route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

It cannot import a selected limit, quotient stream equality, ambient completeness principle, countable-choice selector, or standard bridge row.

Proof. Start with the schedule and tolerance rows. By Definition 435.117, every threshold request reaches the displayed schedule row M through I, W; by Definition 435.118, the downstream tolerance rows are exactly Q, T.

Attach the shared-tail and seal rows. The shared-tail meet rows in this chapter feed the same finite route into Definition 435.147, which synchronizes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, selector-seal, and terminal $\mathsf{Real}^{\uparrow}$ faces over $I, W, M, Q, T, E, H, C, P, N$.

Finally apply Definition 435.151. Overlapping refined reads return to the same finite Phase Real and Completion route, so the threshold consumer has no remaining coordinate for selected limits, quotient streams, ambient completeness, countable choice, or a standard bridge row. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.RootThresholdAndSealRefinement.UniformCauchyCriterionPack root threshold exhaustion`

Theorem 435.153 (UniformCauchyCriterion common-tail uniform criterion bridge)**.** *Let P be an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *packet, let T_0 and T_1 be two certified cofinal tail shifts of the* $\mathsf{StreamName}^{\uparrow}$ *schedule carried by P, and let the two shifted reads land in a common-tail window packet of Definition 301.180 with common refinement W_*. Suppose the finite-request* $\mathsf{Real}^{\uparrow}$ *seal pullback of the common-tail packet is the surface of Definition 301.184. If the displayed rows of that pullback align with an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet*

$$U = (I, W, M, Q, T, E, H, C, P_U, N),$$

so that W_* is read as the shared family window W, the dyadic endpoint and distance rows are read through the tolerance row Q, the common-tail comparison rows are read through the tail

ledger T, and the seal-source row is read through E, then the same finite rows determine the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *route*

$$I \longmapsto W \longmapsto M \longmapsto Q \longmapsto T \longmapsto E.$$

Thus the RegSeqRat common-tail pullback recovers a uniform finite-family criterion through the shared window W, threshold schedule M, tolerance row Q, tail ledger T, and seal row E.

The bridge is only a finite BEDC packet comparison. It asserts no quotient stream, selected limit, host real equality, ambient completeness principle, or second $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. First read the RegSeqRat side. By Definition 301.184, the two cofinal tail reads and their common refinement expose exactly the finite source surface consisting of the $\mathsf{StreamName}^{\uparrow}$ rows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, and distance rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier and regularity rows, $\mathsf{Real}^{\uparrow}$ seal-source rows, componentwise hsame, displayed Cont, package provenance, and local $\mathsf{NameCert}$ provenance.

Next impose the row alignment with U. The common refinement W_* is read as the shared family window W, the dyadic rows are read as the tolerance row Q, the common-tail comparison rows are read as the finite ledger T, and the seal-source row is read as E. The finite family ledger I and the shared schedule row M are therefore the uniform criterion coordinates through which those rows are consumed.

Apply Definition 435.115. Once the aligned rows are viewed as a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ packet, every downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer is exhausted by the finite package $I, W, M, Q, T, E, H, C, P_U, N$.

Now use Definition 435.27. The Real-facing route and the shared finite-window route agree on W, Q, T, E, H, C, P_U, N, so the common-tail comparison rows cannot split into a separate tail selector outside the uniform packet.

Finally apply Definition 435.28. It extracts the Real-seal route from the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces of W, T, Q, E. The non-escape clauses in the RegSeqRat pullback and the UniformCauchyCriterion extraction remove quotient streams, selected limits, host real equality, ambient completeness, hidden completion carriers, and second seal routes, leaving exactly the uniform finite-family criterion route through W, M, Q, T, E. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket common tail uniform criterion bridge`

Theorem 435.154 (UniformCauchyCriterion tail-budget bridge)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The packet exports a Real-facing tail-budget bridge exactly when the displayed Real-facing route factors through the same named sibling rows:*

$$\mathsf{CauchyTailModulusSeal}^{\uparrow} \longrightarrow \mathsf{RealWindowBudget}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

Equivalently, every Real-completion or compact-uniform consumer of U receives the tail budget only by reading the shared window route, the seal route, the finite-tail-filter alignment, the Real-completion window exhaustion, and the public non-escape route already named in this chapter. The bridge exports no quotient stream, selected limit, countable-choice row, detached window, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket tail budget bridge`

Proof. For the forward direction, start with an exported Real-facing tail-budget read. The shared-window obligation and shared-threshold transport force the read through the common I, W, M, Q, T surface before the seal row can be inspected.

Next attach the seal side. The seal-route obligation and Real-seal handoff place every $\mathsf{Real}^{\uparrow}$ endpoint behind the same finite route, while the finite-tail-filter alignment identifies the visible tail ledger with the cofinal-tail sibling surface rather than with an external selector.

The named sibling rows are then read in order. The $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ row supplies the tail-modulus seal, the $\mathsf{RealWindowBudget}^{\uparrow}$ row keeps the budget attached to the displayed finite window, the $\mathsf{StreamName}^{\uparrow}$ row supplies the common window, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ row supplies the terminal seal.

For the reverse direction, assume the displayed consumer route already factors through those six sibling rows. The Real-completion window exhaustion theorem keeps the completion-facing request inside the same finite window and tolerance budget, and the public non-escape export prevents any off-packet analytic coordinate from being added.

Repack the route through H, C, P, N. The accepted packet therefore exports exactly the Real-facing tail-budget bridge described above, and any attempted route through quotient streams, selected limits, countable choice, detached windows, or ambient completeness is excluded by the same public boundary. □

Theorem 435.155 (UniformCauchyCriterion CauchyLimitSeal sibling route)**.** *The* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *finite-family tail route is the sibling consumer surface for chapter 386, chapter 301, and chapter 38. For an accepted packet*

$$U = (I, W, M, Q, T, E, H, C, P, N),$$

the shared window family W, the single threshold schedule M, the dyadic tolerance row Q, and the tail ledger T feed the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route before the* $\mathsf{Real}^{\uparrow}$ *seal handoff E is read. The route exports no quotient stream family, selected limit, host real equality, or ambient completeness object.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket cauchy limit seal sibling route`

Proof. Project U through Definition 435.1. The only source rows available to a tail consumer are the finite family ledger I, the shared $\mathsf{StreamName}^{\uparrow}$ windows W, the shared threshold schedule M, the dyadic row Q, the tail ledger T, and the structural rows H, C, P, N.

The tail comparison itself is pinned down by Definition 435.9. It forces every accepted comparison to pass through I, W, M, Q, T before any real-facing row can be consumed.

The regular-rational readback is the same finite surface by Definition 435.11. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ sibling sees the shared-window and tail-ledger route, not a quotient stream or selected family limit.

Finally apply Definition 435.4. The seal row E is reached only after the displayed family-tail route has been read, so the $\mathsf{CauchyLimitSeal}^{\uparrow}$ sibling route feeds the $\mathsf{Real}^{\uparrow}$ handoff without importing host real equality or ambient completeness. □

Definition 435.156 (UniformCauchyCriterion Real-completion finite-envelope carrier)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. The Real-completion finite-envelope carrier of U is the packet*

$$\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U) = (U, W_{\mathsf{StreamName}^{\uparrow}}, R_{\mathsf{RegSeqRat}^{\uparrow}}, Q_{\mathsf{DyadicRatCore}^{\uparrow}}, S_{\mathsf{CauchyLimitSeal}^{\uparrow}}, E_{\mathsf{RealCauchyCompletion}^{\uparrow}}, H, C, P, N).$$

Its rows are exactly the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *window family* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* R *reached by the uniform tail ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route* S*, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public* $\mathsf{Real}^{\uparrow}$ *export* E*, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}$ *rows* H, C, P, N*. A row is admitted to the carrier only when it is displayed by this packet or by one of the named sibling routes attached to the same* U*.*

Lean4 (definition) `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionRealCompletionFiniteEnvelope`
Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionRealCompletionFiniteEnvelopeCarrier rows`

Theorem 435.157 (UniformCauchyCriterion Real-completion finite-envelope NameCert obligations)**.** *For the carrier* $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$ *of Definition 435.156, the local* NameCert *obligations are the following finite rows:*

1. *carrier exposure: every admitted row is one of* $U, W_{\mathsf{StreamName}^{\uparrow}}, R_{\mathsf{RegSeqRat}^{\uparrow}}, Q_{\mathsf{DyadicRatCore}^{\uparrow}}, S_{\mathsf{CauchyLimitSeal}^{\uparrow}}, E_{\mathsf{RealC}}$
2. *window-family stability:* hsame *transport of a* $\mathsf{StreamName}^{\uparrow}$ *window request remains inside the same displayed* W *family and is replayed only through the visible* Cont *row;*
3. *regular-rational handoff: every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by a consumer is reached from the uniform tail ledger of* U *and not from a second stream source;*
4. *dyadic tolerance ledger exactness: every tolerance comparison consumed at the completion boundary is read from* $Q_{\mathsf{DyadicRatCore}^{\uparrow}}$ *before the tail-meet route is sealed;*
5. *public* $\mathsf{Real}^{\uparrow}$ *export: the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *route exports the named* $\mathsf{Real}^{\uparrow}$ *surface only after* $W_{\mathsf{StreamName}^{\uparrow}}$*,* $R_{\mathsf{RegSeqRat}^{\uparrow}}$*,* $Q_{\mathsf{DyadicRatCore}^{\uparrow}}$*, and* $S_{\mathsf{CauchyLimitSeal}^{\uparrow}}$ *have been displayed;*
6. *non-escape: no consumer row carries quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, or a bridge row outside* $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$*.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion finite envelope namecert obligations`

Proof. Project the carrier definition. It lists the only admitted rows and records the structural rows H, C, P, N as transport, continuation, provenance, and local naming data rather than analytic sources.

Window stability and the regular-rational handoff are inherited from the existing $\mathsf{UniformCauchyCriterion}^{\uparrow}$ route: the $\mathsf{StreamName}^{\uparrow}$ windows are carried by W, and the $\mathsf{RegSeqRat}^{\uparrow}$ rows are reached through the uniform tail ledger before any completion-facing read is available.

The dyadic and public-export obligations are downstream of the same route. The tolerance ledger is the displayed $Q_{\mathsf{DyadicRatCore}^{\uparrow}}$ row, the tail-meet route is the displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ row, and the public $\mathsf{Real}^{\uparrow}$ export is read only through $\mathsf{RealCauchyCompletion}^{\uparrow}$ after those rows have been fixed.

Thus every local naming obligation is discharged by one of the displayed rows. Since the carrier has no further coordinate, the excluded host-level, choice, completeness, quotient, and bridge data cannot be introduced by a consumer. □

Theorem 435.158 (UniformCauchyCriterion Real-completion finite-envelope consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet factors*

through the finite-envelope carrier $\mathcal{E}_{\mathsf{Real}^\uparrow}(U)$ of Definition 435.156. The factorization uses exactly the displayed rows

$$U, W_{\mathsf{StreamName}^\uparrow}, R_{\mathsf{RegSeqRat}^\uparrow}, Q_{\mathsf{DyadicRatCore}^\uparrow}, S_{\mathsf{CauchyLimitSeal}^\uparrow}, E_{\mathsf{RealCauchyCompletion}^\uparrow}, H, C, P, N$$

and no row outside that list can be read by such a consumer.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion finite envelope consumer exhaustion`

Proof. Begin with a $\mathsf{Real}^\uparrow$ or $\mathsf{Completion}^\uparrow$ consumer of U. The consumer must first read the accepted UniformCauchyCriterion packet, so its family request is already tied to the finite $I, W, M, Q, T, E, H, C, P, N$ surface.

Next pass through the real-seal and tail-meet routes. The $\mathsf{StreamName}^\uparrow$ window family, $\mathsf{RegSeqRat}^\uparrow$ source rows, and $\mathsf{DyadicRatCore}^\uparrow$ tolerance ledger are the rows that feed the $\mathsf{CauchyLimitSeal}^\uparrow$ route. No completion-facing read is available before those rows are displayed.

Finally use the public export row. The $\mathsf{RealCauchyCompletion}^\uparrow$ route repacks the same displayed rows as a public $\mathsf{Real}^\uparrow$ surface, while H, C, P, N replay transport, continuation, provenance, and local naming. This exhausts the consumer surface by the finite envelope. □

Theorem 435.159 (UniformCauchyCriterion Real-completion finite-envelope tail-window stability)**.** *In $\mathcal{E}_{\mathsf{Real}^\uparrow}(U)$, every $\mathsf{StreamName}^\uparrow$ window row used by a $\mathsf{Real}^\uparrow$ or $\mathsf{Completion}^\uparrow$ consumer is tied to the corresponding $\mathsf{RegSeqRat}^\uparrow$ source row under the displayed* hsame *and* Cont *rows. Transport may rename the window and replay the readback, but it cannot replace $W_{\mathsf{StreamName}^\uparrow}$ or $R_{\mathsf{RegSeqRat}^\uparrow}$ by a second window family or a second regular-rational source.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionRealCompletionFiniteEnvelope tail window stability`

Proof. Read the window request inside the carrier. The only window-family row is $W_{\mathsf{StreamName}^\uparrow}$, and the only regular-rational source rows are the displayed $R_{\mathsf{RegSeqRat}^\uparrow}$ rows reached from the uniform tail ledger.

Apply the structural rows. The hsame data compare windows componentwise, while the Cont row replays the same readback route after transport. Both rows act on the displayed carrier coordinates rather than creating a new source.

Therefore any transported completion-facing consumer still reads the same $W_{\mathsf{StreamName}^\uparrow}$ and $R_{\mathsf{RegSeqRat}^\uparrow}$ pair. A second window family or a second regular-rational source has no coordinate in the envelope. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion finite envelope tail window stability`

Theorem 435.160 (UniformCauchyCriterion Real-completion finite-envelope no-choice extraction)**.** *The finite-envelope carrier $\mathcal{E}_{\mathsf{Real}^\uparrow}(U)$ exports no quotient stream, no selected host limit, no host real equality, no countable-choice selector, no ambient completeness principle, and no metric-completeness data. Any attempted extraction of one of those objects from a $\mathsf{Real}^\uparrow$ or $\mathsf{Completion}^\uparrow$ consumer is therefore outside the displayed carrier and outside its local* NameCert *obligations.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion finite envelope no choice extraction`

Proof. Inspect the carrier rows. The analytic rows are the StreamName window family, RegSeqRat source rows, dyadic tolerance ledger, CauchyLimitSeal tail-meet route, and RealCauchyCompletion public export. None of these rows stores a quotient stream, a selected limit, host equality, a choice selector, or completeness data.

The structural rows do not add analytic content. BHist records the visible history surface, hsame transports displayed rows, Cont replays displayed continuations, Pkg records provenance, and NameCert names the same local packet.

By consumer exhaustion, every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer has already factored through this list. Hence the listed host-level and choice-based objects cannot be extracted from the envelope. □

Theorem 435.161 (UniformCauchyCriterion Real-completion finite envelope)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer entering through U factors through exactly one finite envelope before it can read a real-facing or completion-facing result. The envelope is the displayed route through the* $\mathsf{StreamName}^{\uparrow}$ *window family W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows reached by the uniform tail ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Q, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public* $\mathsf{Real}^{\uparrow}$ *export, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by the packet and its sibling routes.*

Thus the accepted consumer surface has no coordinate for quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, or bridge rows outside the displayed finite envelope.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion finite envelope`

Proof. Begin with the finite-envelope carrier of Definition 435.156. It displays the accepted packet, the $\mathsf{StreamName}^{\uparrow}$ window family, the $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the dyadic tolerance ledger, the CauchyLimitSeal tail-meet route, the RealCauchyCompletion public export, and the visible structural rows in one packet.

Use the NameCert obligation surface of Definition 435.157. It records carrier exposure, window-family stability, regular-rational handoff, dyadic tolerance exactness, public $\mathsf{Real}^{\uparrow}$ export, and non-escape as the finite obligations for the same displayed packet.

Next apply consumer exhaustion and tail-window stability. Definition 435.158 forces every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer through the displayed carrier rows, and Definition 435.159 keeps the $\mathsf{StreamName}^{\uparrow}$ windows tied to the $\mathsf{RegSeqRat}^{\uparrow}$ source rows under hsame and Cont.

Finally invoke the no-choice extraction boundary of Definition 435.160. Since the consumer surface has already been exhausted by the carrier and its obligations, the Real and completion-facing surface has no coordinate for quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, or bridge rows outside the displayed finite envelope. □

Theorem 435.162 (UniformCauchyCriterion Real-completion synchronizer terminal envelope)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that factors through $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$ and preserves the Phase Real route reaches its terminal* $\mathsf{Real}^{\uparrow}$ *row through the same finite synchronizer envelope. That envelope consists exactly of the* $\mathsf{StreamName}^{\uparrow}$ *window*

family, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route, the* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *route when present, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public export, and the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows attached to* U.

Consequently the terminal Real row is not obtained from quotient streams, selected host limits, host real equality, countable choice, a second stream source, a second readback source, a second tolerance source, metric completeness, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion synchronizer terminal envelope`

Proof. Start with the finite envelope. By Definition 435.161, the consumer has already passed through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. By Definition 435.158, there is no additional consumer coordinate outside that list.

Next use the no-choice boundary. Definition 435.160 rules out quotient streams, selected host limits, host real equality, countable choice, ambient completeness, and metric-completeness data. The tail-window stability theorem also keeps the StreamName window family and RegSeqRat readback rows tied to the same hsame and Cont transport, so no second stream or readback source can enter the envelope. The dyadic tolerance row is already the displayed $Q_{\mathsf{DyadicRatCore}^{\uparrow}}$ ledger, so no second tolerance source is available.

Now identify the terminal route. Apply Definition 435.147 when the selector-seal row is present; it places the selector-facing route on the same StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, and terminal $\mathsf{Real}^{\uparrow}$ surface.

Finally apply Definition 435.148 and Definition 435.149. The first theorem identifies Phase Real admissibility with preservation of the displayed route, and the second theorem makes the endpoint row determined by that route. Hence every admissible Real or Completion consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ row through precisely the finite synchronizer envelope displayed above, with no appeal to open-cover compactness or ambient completeness. □

Theorem 435.163 (UniformCauchyCriterion finite-envelope route determinacy)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Consider two* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumers of* U *that preserve the Phase Real route and factor through the finite envelope* $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$. *If the two consumers read the same displayed* $\mathsf{StreamName}^{\uparrow}$ *rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route, optional* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *row,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *export, and the same* BHist, hsame, Cont, Pkg, *and* NameCert *rows, then they reach the same terminal* $\mathsf{Real}^{\uparrow}$ *row.*

Thus Phase Real and Completion consumers of $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *have a deterministic terminal Real route once their finite-envelope rows agree. No quotient stream, selected host limit, host real equality, countable-choice selector, second tolerance ledger, metric-completeness object, open-cover compactness datum, or ambient completeness principle can separate the two terminal rows.*

Proof. First exhaust both consumers through the finite envelope. Apply Definition 435.158 to place each consumer inside the displayed carrier, and apply Definition 435.159 to keep the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows tied to the same hsame and Cont transport.

The remaining excluded coordinates cannot distinguish the consumers. By Definition 435.160, neither consumer can read quotient streams, selected host limits, host equality, choice selectors, ambient completeness, or metric-completeness data from the envelope.

Now synchronize the selector-facing part of the route. When the optional $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ row is present, Definition 435.147 places it on the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ surface already read by the two consumers. When the row is absent, the common finite envelope has no selector coordinate to separate.

Finally use the Phase Real route criterion and endpoint determinacy. Definition 435.148 identifies both consumers as preserving the same admissible finite route, and Definition 435.149 makes the terminal endpoint row determined by that route. Since the displayed rows agree, the two terminal $\mathsf{Real}^{\uparrow}$ rows agree as well. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket finite envelope route determinacy`

Theorem 435.164 (UniformCauchyCriterion Real-envelope source minimality)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. A* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing terminal envelope for* U *is admissible exactly when its displayed route contains the same* $\mathsf{StreamName}^{\uparrow}$ *window family, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route, the same* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *export row, and the packet's visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Equivalently, the Real-completion finite envelope $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$ *is source-minimal among admissible terminal envelopes: deleting any listed row destroys either the local* NameCert *obligation surface, the consumer factorization, the terminal synchronizer envelope, or route determinacy; adjoining quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, a second stream source, a second readback source, or a second tolerance source produces no new admissible terminal envelope for* U*.*

Proof. First read the displayed carrier and its local obligations. Definition 435.156 lists the only rows of $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$, and Definition 435.157 records why the window family, regular-rational rows, dyadic ledger, tail-meet route, public export, and structural packet rows are all required by the local NameCert surface.

For the forward direction, take an admissible terminal envelope. By Definition 435.158, every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer already factors through the displayed rows of $\mathcal{E}_{\mathsf{Real}^{\uparrow}}(U)$. By Definition 435.162, the terminal Real route uses the same synchronizer rows, and by Definition 435.163, agreement on those rows determines the terminal $\mathsf{Real}^{\uparrow}$ row. Hence an admissible terminal envelope cannot omit or replace one of the displayed sources.

For the reverse direction, suppose the route displays exactly those rows. The finite-envelope carrier supplies the local NameCert obligations, consumer exhaustion supplies the factorization through the displayed route, and the synchronizer theorem supplies the terminal Real-facing row. Route determinacy then identifies the resulting terminal row as the unique endpoint for that displayed route.

It remains to exclude extra sources. The no-choice extraction boundary in Definition 435.160 rules out quotient streams, selected host limits, host real equality, countable choice, ambient completeness, and metric-completeness data. Tail-window stability in Definition 435.159 prevents a second stream or readback source, and the dyadic ledger coordinate in the carrier prevents a second tolerance source. Thus the listed rows are both sufficient and source-minimal. □

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real envelope source minimality`

Theorem 435.165 (UniformCauchyCriterion Real-envelope witness extraction)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Let* $\mathcal{T}$ *be an admissible* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing terminal envelope for* U *whose displayed rows satisfy the source-minimality characterization of Definition 435.164. Then* $\mathcal{T}$ *determines a finite BEDC witness packet*

$$\mathcal{W}_{\mathsf{Real}^{\uparrow}}(U, \mathcal{T}) = (W_{\mathsf{StreamName}^{\uparrow}}, R_{\mathsf{RegSeqRat}^{\uparrow}}, Q_{\mathsf{DyadicRatCore}^{\uparrow}}, S_{\mathsf{CauchyLimitSeal}^{\uparrow}}, E_{\mathsf{RealCauchyCompletion}^{\uparrow}}, H, C, P, N).$$

Its coordinates are exactly the displayed $\mathsf{StreamName}^{\uparrow}$ *window family, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows reached by the uniform tail ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *tail-meet route, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *public export, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{UniformCauchyCriterion}^{\uparrow}}$ *rows. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer read of* $\mathcal{T}$ *factors through this witness packet before the terminal* $\mathsf{Real}^{\uparrow}$ *row is consumed.*

The witness packet has no coordinate for quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, a second stream source, a second readback source, or a second tolerance source.

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real envelope witness extraction`

Proof. Start with carrier exposure. The finite-envelope carrier of Definition 435.156 displays the accepted packet together with $W_{\mathsf{StreamName}^{\uparrow}}$, $R_{\mathsf{RegSeqRat}^{\uparrow}}$, $Q_{\mathsf{DyadicRatCore}^{\uparrow}}$, $S_{\mathsf{CauchyLimitSeal}^{\uparrow}}$, $E_{\mathsf{RealCauchyCompletion}^{\uparrow}}$, and the structural rows H, C, P, N. The NameCert obligations of Definition 435.157 identify these rows as the local carrier, stability, exactness, export, and non-escape data.

Now apply consumer exhaustion and terminal routing. By Definition 435.158, every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer read factors through the displayed carrier rows. By Definition 435.162, the terminal Real-facing read is reached only after the synchronizer rows have been displayed.

Route determinacy turns the displayed rows into a witness rather than a mere description. Definition 435.163 makes the terminal $\mathsf{Real}^{\uparrow}$ row determined by agreement on the listed rows, and Definition 435.164 says an admissible terminal envelope satisfying the source-minimality characterization cannot omit or replace any of them. Therefore $\mathcal{T}$ canonically yields the finite packet $\mathcal{W}_{\mathsf{Real}^{\uparrow}}(U, \mathcal{T})$ with exactly those coordinates.

Finally exclude external analytic sources. The no-choice extraction theorem Definition 435.160 rules out quotient streams, selected host limits, host real equality, countable choice, ambient completeness, and metric-completeness data. Tail-window stability rules out second stream and readback sources, while the dyadic ledger row in the carrier fixes the tolerance source. Hence every consumer read reaches the terminal $\mathsf{Real}^{\uparrow}$ row through the witness packet and through no additional source. □

Theorem 435.166 (UniformCauchyCriterion Real-completion terminal readiness)**.** *Let*

$$U = (I, W, M, Q, T, E, H, C, P, N)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let* $\mathcal{T}$ *be an admissible* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing terminal envelope for* U *satisfying the source-minimality characterization of Definition 435.164.*

Any terminal Real-and-Completion consumer compatible with the finite witness packet $\mathcal{W}_{\mathsf{Real}^\uparrow}(U, \mathcal{T})$*, the finite-selector UniformCauchy Real readiness route of Definition 782.56, and the Real L10 four-face exit sufficiency route of Definition 38.493 factors through the single displayed terminal readiness surface*

$$\mathsf{StreamName}^\uparrow,\ \mathsf{DyadicRatCore}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow,\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

This terminal readiness surface has no coordinate for quotient streams, host real equality, selected host limits, countable choice, metric-completeness rows, ambient completeness, aggregate formalstatus *rows, or aggregate bridge rows. In particular it is a readiness theorem for consuming the finite Real-and-Completion route, not a claim of* $\mathsf{Real}^\uparrow$ *completeness or a route-level bridge.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCriterionUp.UniformCauchyCriterionPacket real completion terminal readiness`

Proof. First extract the finite witness packet. By Definition 435.165, the admissible envelope $\mathcal{T}$ canonically exposes exactly the finite rows $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{CauchyLimitSeal}^\uparrow$, $\mathsf{RealCauchyCompletion}^\uparrow$, and the structural support H, C, P, N carried by U. Its no-escape clause removes quotient streams, selected host limits, host real equality, countable choice, ambient completeness, metric-completeness data, and duplicate stream or tolerance sources.

Next attach the selector-side terminal route. Apply Definition 782.56. It turns any compatible finite-selector consumer into a reader of the same public rows $\mathsf{StreamName}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert, without hidden selectors, detached limits, metric-completeness rows, ambient completeness, quotient streams, host real equality, or countable choice.

Now attach the objectwise Phase Real and Completion exit. By Definition 38.493, the Phase Real and Completion exit is supplied exactly by the four local records for $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$. The cited sufficiency theorem supplies no aggregate formalstatus row, no aggregate bridge row, no quotient stream, no selected limit, no countable-choice row, no metric-completeness datum, and no $\mathsf{Real}^\uparrow$ completeness claim.

The three routes therefore have the same terminal consumer surface. The finite witness packet gives the UniformCauchy envelope rows, the selector theorem gives the terminal public readiness route, and Real L10 exit sufficiency certifies that the Phase Real and Completion exit is objectwise over the same four faces. The no-escape clauses in the UniformCauchy envelope together with the Real L10 face-status noncollapse results prevent every listed external coordinate from serving as an alternate terminal read. □

The public $\mathsf{UniformCauchyCriterion}^\uparrow$ route is a Phase Real and Completion lattice node: its finite budget reads through chapter 126, chapter 301, chapter 291, chapter 38, chapter 386, and chapter 1081 before any Real-facing consumer can use the seal. The route exports only finite BHist, hsame, Cont, Pkg, and local NameCert rows, with no quotient stream, selected host limit, host equality, countable choice, or ambient Real-completeness coordinate.

Chapter 436

A Concrete Naming Certificate for $\mathsf{DedekindCut}^{\uparrow}$

The $\mathsf{DedekindCut}^{\uparrow}$ interface records the rounded located cut presentation of constructive real analysis in the style of Bishop–Bridges and Troelstra–van Dalen. It keeps lower and upper rational observations as explicit $\mathsf{Rat}^{\uparrow}$ rows, rather than quotienting cut predicates or comparing host reals.

The $\mathsf{DedekindCut}^{\uparrow}$ carrier consumes the rounded located-cut surface of chapter 404 and the rational observation rows of chapter 37. Its lower and upper predicate rows are therefore a finite BEDC ledger over rational observations rather than an isolated host cut predicate.

The lower and upper rational-observation rows consume the rational certificate of chapter 37, and the comparison handoff is exported only to the real seal boundary of chapter 38.

Definition 436.1 (DedekindCut BHist source)**.** *A* $\mathsf{DedekindCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L, U, I, R, O, E, Q, H, T, P, N)$$

with the following displayed coordinates:

1. *L is the lower* $\mathsf{Rat}^{\uparrow}$ *predicate row, listing rational observations certified to lie below the represented real;*
2. *U is the upper* $\mathsf{Rat}^{\uparrow}$ *predicate row, listing rational observations certified to lie above the represented real;*
3. *I stores lower and upper inhabitedness witnesses, with displayed rational rows on both sides;*
4. *R stores roundedness witnesses for the lower and upper rows, so each observed rational admits a strictly better observed rational on the same side;*
5. *O stores locatedness witnesses, assigning each rational interval $a < b$ a displayed lower-side or upper-side observation;*
6. *E stores disjointness evidence preventing the same rational observation from being accepted by both sides;*
7. *Q stores rational-embedding rows, including the lower and upper boundary readings for embedded* $\mathsf{Rat}^{\uparrow}$ *points;*
8. *H stores the componentwise* hsame *routes for the lower predicate, upper predicate, roundedness, locatedness, disjointness, and rational embedding rows;*

9. *T stores the* Cont *routes that expose finite lower and upper rational observations to downstream consumers;*
10. *P stores* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCut}^{\uparrow}}$ *evidence.*

The classifier compares agreement of the displayed lower and upper rational observations under $\mathsf{Rat}^{\uparrow}$ *comparison. It never compares quotient cuts, host real equality, or choice-selected limits.*

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutTasteGate single carrier alignment`

Theorem 436.2 (DedekindCut NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DedekindCut}^{\uparrow}}$ *obligation surface for the carrier of Definition 436.1 consists of five finite rows:*

1. *carrier admission: the accepted object is exactly the packet C with displayed lower and upper* $\mathsf{Rat}^{\uparrow}$ *predicate rows, inhabitedness, roundedness, locatedness, disjointness, rational embedding rows, transports, continuations, package provenance, and local naming evidence;*
2. *classifier transport: classification transports the lower and upper rational observations componentwise through the stored* hsame *and* Cont *routes and compares them only by* $\mathsf{Rat}^{\uparrow}$ *comparison;*
3. *stability: replacing any displayed rational observation by a classified* $\mathsf{Rat}^{\uparrow}$ *observation preserves the rounded located cut laws inside the same carrier surface;*
4. *exactness: every consumer-visible lower or upper decision is the row supplied by L, U, O, E, and Q, with no hidden host-real predicate;*
5. *ledger: every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer first reads $L, U, I, R, O, E, Q, H, T, P, N$.*

Proof. Carrier admission is projection from the tuple of Definition 436.1. The lower and upper rational predicate rows, their inhabitedness, roundedness, locatedness, disjointness, rational embedding, transport, continuation, package, and naming rows are all displayed coordinates.

Classifier transport is componentwise. A lower-side observation is moved through the lower $\mathsf{Rat}^{\uparrow}$ route, an upper-side observation is moved through the upper $\mathsf{Rat}^{\uparrow}$ route, and the comparison used by the classifier is the displayed rational comparison between those transported rows.

Exactness and the ledger clauses follow from the same inventory. Since the carrier contains no quotient cut, host real equality proof, selected limit, or ambient completeness object, every consumer-visible decision is exhausted by the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutCarrier namecert obligation surface`

Theorem 436.3 (DedekindCut rounded located laws)**.** *Every accepted* $\mathsf{DedekindCut}^{\uparrow}$ *carrier $C = (L, U, I, R, O, E, Q, H, T, P, N)$ presents a constructive rounded located cut over* $\mathsf{Rat}^{\uparrow}$*: both sides are inhabited, lower and upper observations are rounded on their own sides, each rational interval $a < b$ is decided by the locatedness row O, and disjointness row E prevents a rational observation from being accepted by both sides.*

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutCarrier located rounded cut laws`

Proof. Read the inhabitedness row I to obtain displayed rational observations on both sides. These witnesses are part of the finite carrier and do not require a completed real.

Read the roundedness row R. For a lower observation it supplies a stronger lower observation; for an upper observation it supplies a stronger upper observation. The witnesses are again $\mathsf{Rat}^{\uparrow}$ rows transported through the displayed carrier routes.

For a rational interval $a < b$, the locatedness row O gives the lower-side or upper-side observation named by the packet. The disjointness row E then prevents overlap of the two displayed predicates, so the cut laws remain internal to the finite $\mathsf{DedekindCut}^{\uparrow}$ surface. □

Theorem 436.4 (DedekindCut rational embedding boundary)**.** *For every rational row embedded through Q in an accepted* $\mathsf{DedekindCut}^{\uparrow}$ *carrier, the lower and upper boundary observations are exactly the displayed* $\mathsf{Rat}^{\uparrow}$ *comparison rows against that rational. A consumer cannot replace those rows by quotient equality of cuts, host real equality, or an external order on completed reals.*

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutCarrier rational embedding boundary`

Proof. Project the rational embedding row Q. Its lower boundary read is a displayed comparison placing an observed rational below the embedded row, and its upper boundary read is the corresponding displayed comparison above the embedded row.

Transport along H and T preserves only these rational observations. The classifier of Definition 436.2 compares lower and upper sides by $\mathsf{Rat}^{\uparrow}$ comparison, so the embedded rational has no separate equality or order witness outside the carrier. □

Theorem 436.5 (DedekindCut RealUp comparison handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing comparison consumer of an accepted* $\mathsf{DedekindCut}^{\uparrow}$ *carrier receives only the displayed lower and upper* $\mathsf{Rat}^{\uparrow}$ *observation surface, the rounded located cut laws, and the rational embedding boundary. The handoff supplies no quotient real, host equality, selected limit, or choice-selected comparison.*

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutCarrier realup comparison handoff`

Proof. First apply Definition 436.3. It supplies the inhabitedness, roundedness, locatedness, and disjointness rows needed for a constructive cut comparison.

Next apply Definition 436.4. It fixes the rational boundary reads as displayed $\mathsf{Rat}^{\uparrow}$ comparisons rather than external real comparisons.

The non-escape claim is then immediate from the carrier inventory in Definition 436.1. The only exported analytic data are lower and upper rational observations with their local evidence and provenance. □

Theorem 436.6 (DedekindCut quotient-free boundary)**.** *The displayed* $\mathsf{DedekindCut}^{\uparrow}$ *carrier, rounded located laws, rational embedding boundary, and* $\mathsf{Real}^{\uparrow}$ *comparison handoff license only a classifier-mediated comparison interface over* BHist*,* hsame*,* Pkg*, and* NameCert*. They do not introduce a quotient object, a completeness theorem, or an alternative construction of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the carrier of Definition 436.1. It lists lower and upper rational-observation rows, local law rows, rational embedding rows, componentwise transport, continuation, package, and naming data, but no quotient carrier or host real equality coordinate.

The admissible classifier surface is fixed by Definition 436.2. Its comparison reads are mediated by the displayed BHist rows and the local NameCert evidence, so the interface remains a certificate boundary rather than a new real object.

Apply Definition 436.3 and Definition 436.5. These give the rounded located laws and the comparison handoff while keeping every read on rational observations and their provenance. Hence the boundary permits classifier-mediated comparison only, with no quotient object, completeness theorem, or alternative $\mathsf{Real}^{\uparrow}$ construction. □

Lean4 ✓ `BEDC.Derived.DedekindCutUp.DedekindCutCarrier quotient free boundary`

Theorem 436.7 (DedekindCut bridged Real comparison interface)**.** *The rounded located cut carrier, rational embedding boundary, and* $\mathsf{Real}^{\uparrow}$ *comparison handoff together form the BEDC-side bridge interface from lower and upper* $\mathsf{Rat}^{\uparrow}$ *observation rows to* $\mathsf{Real}^{\uparrow}$ *comparison consumers. The interface exports no quotient cut, selected limit, ambient completeness, or host equality.*

Proof. Start with the NameCert obligation surface of Definition 436.2. It fixes the carrier reads as lower and upper $\mathsf{Rat}^{\uparrow}$ predicate rows, local law rows, rational embedding rows, transports, continuations, provenance, and local naming.

Apply the rounded located cut laws of Definition 436.3. These laws make the lower and upper observation rows constructive enough for comparison consumers without introducing a quotient cut.

Use the rational embedding boundary and the $\mathsf{Real}^{\uparrow}$ comparison handoff, Definition 436.4 and Definition 436.5. Together they expose the bridge-facing comparison surface from rational observation rows to $\mathsf{Real}^{\uparrow}$ consumers while refusing selected limits, ambient completeness, and host equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCut}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DedekindCut}^{\uparrow}$ packet is generated from lower and upper $\mathsf{Rat}^{\uparrow}$ predicate rows, inhabitedness, roundedness, locatedness, disjointness, rational embedding rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DedekindCut}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 436.1, Definition 436.2, Definition 436.3, Definition 436.4, Definition 436.5, Definition 436.6, and Definition 436.7 bind the bridge-facing lower and upper rational-observation NameCert surface over $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DedekindCut}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.DedekindCutUp.DedekindCutCarrier\_realup\_comparison\_handoff`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient cuts, host real equality, selected limits, ambient real completeness, or bridge checking.

Upgrade path: No further theory-axis upgrade remains; standard-model bridge checking is the remaining independent verification-axis work.

Chapter 437

A Concrete Naming Certificate for $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$

The $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ interface records the finite seal packet consumed by the transported stationary diagonal path. It packages the transported $\mathsf{Rat}^{\uparrow}$ seed classifier, the constant $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ window seal, the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ observation boundary, and the $\mathsf{Real}^{\uparrow}$ seal-facing row. It is a finite naming certificate, not a real-completeness theorem.

The transported stationary window-seal packet consumes the constant-window route of chapter 126, the regular rational handoff of chapter 301, the finite observation boundary of chapter 365, and the real-facing seal row of chapter 38 through the route $R, S, G, W, E, H, C, P, N$ before any consumer reads the transported seal.

Definition 437.1 (Transported stationary window-seal carrier)**.** *A* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, S, G, W, E, H, C, P, N)$$

with the following visible coordinates:

1. *R is the transported* $\mathsf{Rat}^{\uparrow}$ *classifier row for the two stationary rational seeds;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier induced by R;*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal row read from the same constant stream window;*
4. *W is the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet naming the selected anchor, window, dyadic ledger, regular-rational seal, and endpoint classifier;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row obtained from G and W;*
6. *H records the transported stationary diagonal handoff rows that identify the displayed diagonal representatives with the same constant rational route;*
7. *C stores the displayed* hsame *and* Cont *transports for the seed classifier, stream window, regular-rational seal, finite envelope, and real seal;*
8. *P is the* Pkg *provenance naming* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *source certificates;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TransportedStationaryWindowSeal}^{\uparrow}}$ *row.*

The local classifier relates two carriers exactly when their transported $\mathsf{Rat}^{\uparrow}$ *classifier rows agree and the induced* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, package, and local certificate rows compare componentwise after the displayed routes in* C.

Theorem 437.2 (Transported stationary window-seal NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{TransportedStationaryWind}}$ *surface has five obligations:*

1. *carrier: formation of the finite packet of Definition 437.1;*
2. *classifier: componentwise comparison of the transported* $\mathsf{Rat}^{\uparrow}$ *seed classifier, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet,* $\mathsf{Real}^{\uparrow}$ *row, transports, package provenance, and local certificate row;*
3. *transported window exactness: the* $\mathsf{Rat}^{\uparrow}$ *classifier, constant* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal read the same transported stationary row;*
4. *seal exhaustion: every* $\mathsf{Real}^{\uparrow}$ *consumer read factors through the displayed* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row;*
5. *non-escape: the carrier has no coordinate for quotient streams, host equality, choice, selected host limits, or real completeness.*

Lean4 ✓ `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealPacket namecert obligations`

Proof. Carrier formation is the finite assembly of the displayed tuple. The transported rational classifier supplies R, the stationary stream read supplies S, the regular-rational boundary supplies G, and the finite observation envelope supplies W. The real-facing row E, transported diagonal rows H, route table C, provenance row P, and local naming row N are all visible coordinates of the same BHist packet.

For the classifier obligation, compare the packet component by component after applying the displayed hsame and Cont routes. The transported $\mathsf{Rat}^{\uparrow}$ row determines the constant stream window, the stream window determines the regular-rational finite seal, and the envelope ties the same window and dyadic ledger to the real-facing endpoint row.

Transported window exactness is the stationary diagonal calculation read through a finite window. The transported $\mathsf{Rat}^{\uparrow}$ classifier induces the constant $\mathsf{StreamName}^{\uparrow}$ classifier, and the $\mathsf{RegSeqRat}^{\uparrow}$ row consumes exactly that window with the displayed ledger. Thus the three public readings are the same stationary row at the window boundary.

Seal exhaustion follows by projecting W and E. The $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ packet exposes the selected anchor, finite window, dyadic ledger, regular-rational seal, and endpoint classifier, and the $\mathsf{Real}^{\uparrow}$ row can read only those exposed fields plus the displayed transports and provenance.

The non-escape clause is constructor inspection. None of $R, S, G, W, E, H, C, P, N$ is a quotient stream, a host equality proof, a choice operator, a selected host limit, or a real-completeness witness. Therefore the certificate remains a finite transported window-seal packet. □

Theorem 437.3 (Transported stationary window-seal readback determinacy)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *carrier* $T = (R, S, G, W, E, H, C, P, N)$, *the transported* $\mathsf{Rat}^{\uparrow}$ *classifier row* R *determines the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal* G. *If two accepted carriers have the same transported* $\mathsf{Rat}^{\uparrow}$ *classifier after the displayed* hsame *and* Cont *routes, then their constant stream windows and regular-rational finite-window seals agree at the public readback boundary.*

Proof. Unpack the carrier of Definition 437.1. The row R is the transported rational classifier, S is the constant stream window induced by that row, and G is the regular-rational seal read from the same window.

The transported window exactness clause in Definition 437.2 states that R, S, and G read the same transported stationary row. The classifier clause transports these rows only componentwise through the displayed hsame and Cont routes.

Hence equality of the transported $\mathsf{Rat}^{\uparrow}$ classifier at the displayed route fixes the induced constant $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ seal at the public readback boundary. □

Lean4 ✓ `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealPacket readback determinacy`

Theorem 437.4 (Transported stationary window-seal Real factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exposed by an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet* W *and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal* G*. The real-facing row* E *may read the selected anchor, finite window, dyadic ledger, regular-rational seal, and endpoint classifier only through* W *together with the transported seal row* G *and the displayed transport and provenance rows.*

Lean4 ✓ `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealPacket real factorization`

Proof. Project the coordinates G, W, E from the carrier tuple. The finite-window envelope W names the anchor, window, dyadic ledger, regular-rational seal, and endpoint classifier, while E is the real-facing row obtained from that envelope and the seal row.

Apply the seal-exhaustion clause of Definition 437.2. It requires every $\mathsf{Real}^{\uparrow}$ consumer read to pass through W and the same $\mathsf{RegSeqRat}^{\uparrow}$ seal row.

The route table C and provenance row P preserve that factorization under the displayed transports. Therefore no $\mathsf{Real}^{\uparrow}$ read bypasses the finite envelope and regular-rational seal. □

Theorem 437.5 (Transported stationary window-seal uniqueness)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *and* $T' = (R', S', G', W', E', H', C', P', N')$ *be accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *carriers. Suppose they are displayed over the same transported* $\mathsf{Rat}^{\uparrow}$ *row, the same constant* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *finite seal, the same* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet, and the same* $\mathsf{Real}^{\uparrow}$ *seal-facing row after the* hsame *rows and* Cont *routes named by* H, H', C, C'*. Then* T *and* T' *expose the same public seal row.*

The uniqueness claim is only about the finite public seal row. It does not identify quotient streams, host equalities, selected host limits, or real-completeness data.

Proof. Apply the carrier and classifier clauses of Definition 437.2. They restrict comparison of T and T' to the displayed transported rational row, constant stream window, regular-rational seal, finite envelope, real-facing row, transport rows, continuation routes, provenance, and local naming rows.

Next use Definition 437.3. Agreement of the transported $\mathsf{Rat}^{\uparrow}$ row under the displayed hsame and Cont routes fixes the constant $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window seal at the public readback boundary.

Apply Definition 437.4. Any $\mathsf{Real}^{\uparrow}$ seal-facing read must factor through the same finite envelope W and the same regular-rational seal row G, so the public seal row exposed by T and T' is the same row.

Finally, Definition 437.6 blocks the only possible escapes from this finite comparison. Quotient streams, host equality, selected host limits, and real-completeness data are not carrier coordinates, so they cannot refine or split the public seal row. □

Theorem 437.6 (Transported stationary window-seal ledger non-escape)**.** *An accepted* TransportedStationaryWindo *carrier exports no quotient stream, host equality, choice, selected host limit, or real-completeness datum. Its public surface consists only of the transported* $\mathsf{Rat}^{\uparrow}$ *classifier, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet,* $\mathsf{Real}^{\uparrow}$ *seal-facing row, displayed transports, package provenance, local* NameCert *row, and the ambient finite* BHist *packet.*

Lean4 ✓ `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealLedgerNonescape`

Proof. Inspect the carrier tuple $T = (R, S, G, W, E, H, C, P, N)$. Each coordinate is one of the finite transported classifier, window, seal, envelope, real-facing, transport, continuation, provenance, or naming rows.

The non-escape clause of Definition 437.2 explicitly excludes quotient streams, host equality, choice, selected host limits, and real completeness from that tuple. The readback determinacy and real-factorization statements use only the same visible coordinates.

Consequently the transported stationary window-seal packet has no export path outside the displayed finite BEDC ledger. □

Theorem 437.7 (Transported stationary window-seal carrier obligation)**.** *For an accepted* TransportedStationaryWi *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

carrier admission is exactly the finite projection of Definition 437.1. The admitted rows are the transported $\mathsf{Rat}^{\uparrow}$ *classifier* R*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal* G*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet* W*, the* $\mathsf{Real}^{\uparrow}$ *seal-facing row* E*, the transported stationary handoff row* H*, the route table* C*, the provenance row* P*, and the local naming row* N*.*

Proof. Project the displayed carrier tuple. The definition lists the complete coordinate inventory of an admitted packet, and Definition 437.2 names carrier formation as the first local obligation.

The readback-determinacy theorem then identifies R, S, G as the same transported stationary window read at the public boundary. The Real factorization theorem identifies W, E as the only real-facing seal route.

The uniqueness and ledger non-escape theorems rule out any additional carrier coordinate. Thus carrier admission is precisely the finite row projection displayed above. □

Theorem 437.8 (Transported stationary window-seal seed window readback)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the transported $\mathsf{Rat}^{\uparrow}$ *seed row* R *determines the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal* G *through the same finite route* H, C, P, N*. The seed-window readback obligation is therefore the public projection of* R, S, G, H, C, P, N*, not a separate stream equality or host-limit claim.*

Proof. Project the carrier by Definition 437.7. The rows R, S, G are admitted as the transported seed classifier, constant stream window, and regular-rational finite-window seal of the same packet.

Apply Definition 437.3. Agreement at R fixes the public readback of S and G through the displayed hsame and Cont routes.

The support rows H, C, P, N keep that readback inside the finite NameCert surface. Ledger non-escape prevents any stream quotient, host equality, selected limit, or real-completeness coordinate from refining the seed-window read. □

Theorem 437.9 (Transported stationary window-seal Real seal route)**.** *For an accepted* $\mathsf{TransportedStationaryWind}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *seal read factors through* $R, S, G, W, E, H, C, P, N$ *and cannot bypass the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row* W*. The Real-facing route reads the transported seed through* R, S, G*, enters the finite envelope at* W*, exposes the seal row* E*, and replays only the displayed transport, route, provenance, and naming rows.*

Proof. Start with Definition 437.4. It forces every $\mathsf{Real}^{\uparrow}$ read exposed by the packet through the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ packet W and the same $\mathsf{RegSeqRat}^{\uparrow}$ seal row G.

Use Definition 437.8 to identify the seed-facing part of the route. The rows R, S, G provide the transported rational seed, the constant window, and the regular-rational seal that feed the finite envelope.

Finally apply carrier admission and ledger non-escape. The admitted route has only $R, S, G, W, E, H, C, P, N$, so no consumer can reach a $\mathsf{Real}^{\uparrow}$ seal by skipping W or by importing quotient streams, host equality, selected limits, choice, or real-completeness data. □

Theorem 437.10 (Transported stationary window-seal root RegSeqRat window ownership)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G *is owned by the same transported* $\mathsf{Rat}^{\uparrow}$ *row* R *and the same constant* $\mathsf{StreamName}^{\uparrow}$ *window row* S*. A root consumer may read* G *only as the finite-window seal attached to the route*

$$R \longmapsto S \longmapsto G$$

through the displayed rows H, C, P, N*; it cannot detach* G *as an independent regular-rational source.*

Proof. Project the packet by Definition 437.7. The admitted carrier contains R, S, and G as visible rows of one finite transported window-seal packet, together with the same transport, continuation, provenance, and local naming rows H, C, P, N.

Apply Definition 437.8. It identifies the transported rational seed row, the constant stream window, and the regular-rational seal as one public readback route inside the finite NameCert surface.

Finally use Definition 437.3. Since agreement at R fixes both S and G at the public boundary, the seal row G has no separate owner beyond the displayed R, S, H, C, P, N route. □

Theorem 437.11 (Transported stationary window-seal root Real seal non-escape)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

every root-facing $\mathsf{Real}^{\uparrow}$ *seal read must pass through the finite rows* $R, S, G, W, E, H, C, P, N$*. The read first uses the owned regular-rational window route* R, S, G*, then the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row* W*, then the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, with* H, C, P, N *carrying only displayed transport, continuation, package, and naming data. It cannot import quotient streams, host equality, selected limits, choice, or real-completeness data.*

Proof. Begin with Definition 437.4. Every $\mathsf{Real}^{\uparrow}$ read exposed by the accepted packet factors through the finite envelope W and the same regular-rational seal row G.

Use Definition 437.9 to name the complete public route. It places the transported seed, constant window, regular-rational seal, envelope, real-facing row, and structural rows in the finite order $R, S, G, W, E, H, C, P, N$.

The non-escape boundary is then Definition 437.6. Quotient streams, host equality, selected host limits, choice, and real-completeness data are not coordinates of the packet, so no root-facing Real seal read can enter through them. □

Theorem 437.12 (Transported stationary window-seal root downstream handoff package)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the downstream root handoff package consumed by completion-facing chapters consists exactly of the finite window ownership row and the Real seal non-escape row:*

1. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal owned by the transported* $\mathsf{Rat}^{\uparrow}$ *row R and the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S through H, C, P, N;*
2. *every root-facing* $\mathsf{Real}^{\uparrow}$ *seal read factors through $R, S, G, W, E, H, C, P, N$ and has no quotient-stream, host-equality, selected-limit, choice, or real-completeness input.*

Thus a downstream consumer receives a finite BEDC packet over $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, not an ambient completion object.*

Proof. The first row is Definition 437.10. It binds the regular-rational seal row to the transported rational seed and the constant stream window before any envelope or Real-facing consumer reads the packet.

The second row is Definition 437.11. It forces every Real seal read through the displayed finite packet and excludes all hidden quotient, equality, limit, choice, and completeness inputs.

Package these two rows with Definition 437.9. The result is the finite downstream handoff surface: consumers may reuse $R, S, G, W, E, H, C, P, N$, but they do not receive any object outside the displayed BEDC carrier. □

Theorem 437.13 (Transported stationary window-seal classifier obligation)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *and* $T' = (R', S', G', W', E', H', C', P', N')$ *be accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packets. Any classifier comparison between them is componentwise across the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows: R compares with R', S with S', G with G', W with W', and E with E' only through the* hsame *rows,* Cont *routes, provenance rows, and local naming rows carried by $H, H', C, C', P, P', N, N'$.*

Proof. Apply the classifier row of Definition 437.2. It permits only componentwise comparison of the transported seed classifier, the induced stream window, the regular-rational seal, the finite envelope, the real-facing row, and the displayed transport and provenance rows.

Use readback determinacy to keep the $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ comparisons on the same stationary route. Use Real factorization to keep the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ comparisons on the same envelope and seal row.

Uniqueness repacks these componentwise comparisons as one public seal row, while ledger non-escape prevents the classifier from reading quotient streams, host equality, selected limits, or real-completeness data. □

Theorem 437.14 (Transported stationary window-seal ledger obligation)**.** *Every consumer read of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet factors through the finite rows*

$$R, S, G, W, E, H, C, P, N.$$

Requests for quotient streams, host equality, choice, selected limits, or real completeness have no carrier field and therefore cannot be exported by the transported stationary window-seal ledger.

Proof. Start from Definition 437.6. Its non-escape clause states that the packet exports only the finite transported classifier, stream, regular-rational seal, finite envelope, real-facing, transport, route, provenance, and naming rows.

Readback determinacy and Real factorization describe all displayed consumer paths: the stationary window path passes through R, S, G, and the real-facing path passes through W, E together with the same transport and provenance rows.

It remains only to exclude hidden completion data. The carrier definition and uniqueness theorem leave no coordinate for quotient streams, host equality, selected limits, or real completeness, so the ledger obligation is exactly the finite non-escape boundary. □

Theorem 437.15 (Transported stationary window-seal root source lock)**.** *For every root consumer of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the first readable source is the transported $\mathsf{Rat}^{\uparrow}$ *seed row* R *locked through the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal* G*. The consumer may reach the finite envelope, Real-facing row, or completion-facing handoff only after the single route* R, S, G, H, C, P, N *has been fixed by the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Project the root consumer through the carrier of Definition 437.1. Its available source coordinates are exactly the transported rational row R, the constant stream window S, the regular-rational seal G, and the displayed transport, route, provenance, and naming rows H, C, P, N.

Apply Definition 437.3. It makes R, S, and G one stationary readback route, so a consumer cannot read the stream window or the regular-rational seal as a second root source.

The sibling surfaces named at the start of this chapter supply the same ordering: the constant-window route of chapter 126, the regular-rational handoff of chapter 301, and the transported $\mathsf{Rat}^{\uparrow}$ row are read before the finite envelope and Real-facing row. Ledger non-escape then blocks quotient streams, host equality, selected host limits, choice, and real-completeness data from supplying another source. □

Theorem 437.16 (Transported stationary window-seal root Real factorization)**.** *For every Real-facing root read of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the read factors through the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row* W *and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G*. Thus the root handoff from the transported rational source to* $\mathsf{Real}^{\uparrow}$ *is the finite factorization*

$$R, S, G, H, C, P, N \longrightarrow W \longrightarrow E.$$

It does not factor through an ambient quotient stream, host equality, selected host limit, choice principle, completion witness, or real-completeness datum.

Proof. Begin with Definition 437.15. The root source has already been fixed as the transported $\mathsf{Rat}^{\uparrow}$ row read through the constant $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ seal.

Apply Definition 437.4. Every $\mathsf{Real}^{\uparrow}$ consumer path must pass through the finite envelope W and the same regular-rational seal row G, with H, C, P, N carrying only the displayed transport, continuation, provenance, and local naming data.

The finite observation boundary of chapter 365 and the real-facing seal row of chapter 38 are therefore joined only by the displayed packet. Uniqueness identifies the resulting public seal row, and ledger non-escape removes every off-packet completion or real-completeness route. □

Theorem 437.17 (Transported stationary window-seal root ledger exhaustion)**.** *The root handoff surface of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet exposes exactly*

$$R, S, G, W, E, H, C, P, N.$$

It exposes the transported $\mathsf{Rat}^{\uparrow}$ *row, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet,* $\mathsf{Real}^{\uparrow}$ *seal-facing row, displayed transports, route table, provenance row, and local naming row. It exposes no quotient stream, host equality, choice operator, selected host limit, completion witness, or real-completeness datum.*

Proof. The carrier definition gives the root ledger its complete coordinate list. The NameCert obligations and classifier obligation allow only componentwise transport of those displayed rows.

Use the root source lock to account for the source side R, S, G, H, C, P, N. Use the root Real factorization theorem to account for the envelope and Real-facing side W, E.

Finally apply uniqueness and ledger non-escape. Uniqueness shows that the public root handoff is determined by the displayed finite rows, while ledger non-escape states that quotient streams, host equality, selected limits, choice, completion witnesses, and real-completeness data are not carrier coordinates. Hence the L10 handoff surface is exhausted by $R, S, G, W, E, H, C, P, N$. □

Theorem 437.18 (Transported stationary window-seal obligation package)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the consumer-facing obligation package is the finite conjunction of carrier admission, classifier comparison, transported $\mathsf{Rat}^{\uparrow}$*-to-*$\mathsf{StreamName}^{\uparrow}$ *readback determinacy,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *factorization of every* $\mathsf{Real}^{\uparrow}$ *read, public-row uniqueness, and ledger non-escape. The package ranges only over the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of* T*.*

Proof. First project the carrier and local naming surface. The carrier definition supplies the rows $R, S, G, W, E, H, C, P, N$, while Definition 437.2 supplies the local carrier, classifier, transported-window exactness, seal-exhaustion, and non-escape obligations.

Next assemble the readback and Real-facing parts. Apply Definition 437.3 to read the transported $\mathsf{Rat}^{\uparrow}$ row through the constant $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window seal. Apply Definition 437.4 to force every $\mathsf{Real}^{\uparrow}$ consumer path through the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ packet and the same $\mathsf{RegSeqRat}^{\uparrow}$ seal row.

Finally repack the consumer boundary. The uniqueness theorem identifies the public seal row determined by the displayed components, the carrier-obligation theorem and classifier-obligation theorem expose the finite admission and comparison clauses, and Definition 437.14 together with ledger non-escape blocks quotient streams, host equality, selected limits, choice, and real-completeness data. □

Theorem 437.19 (Transported stationary window-seal selected-tail stability)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ and $T' = (R', S', G', W', E', H', C', P', N')$ be accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packets. Suppose the rows R, S, G, W compare with R', S', G', W' under the displayed* hsame *rows and* Cont *routes carried by H, H', C, C'. Then the selected finite-window tail route used by G and W is stable under the same transport: the transported route selected for T and the route selected for T' expose the same public* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal and* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *boundary.*

Moreover, every $\mathsf{Real}^{\uparrow}$ *seal read through E or E' still factors through the transported rows R, S, G, W, H, C, P, N and $R', S', G', W', H', C', P', N'$. The comparison does not introduce quotient streams, selected host limits, choice, or real-completeness data.*

Proof. First use readback determinacy. Agreement of R with R' through the displayed hsame and Cont routes fixes the corresponding constant $\mathsf{StreamName}^{\uparrow}$ windows S, S' and the regular-rational finite-window seals G, G' at the public readback boundary.

Next apply the classifier obligation. Its componentwise comparison keeps R, S, G, W aligned with R', S', G', W' through the same transported rows rather than through an ambient stream equality. Hence the finite tail route selected by G and W is carried to the finite tail route selected by G' and W'.

Now apply Real factorization. Any $\mathsf{Real}^{\uparrow}$ read through E must pass through W and G, and any read through E' must pass through W' and G', together with the displayed transport, route, provenance, and local naming rows.

Finally use ledger non-escape. Since quotient streams, selected host limits, choice, and real-completeness data are not coordinates of either carrier, transport stability can only assert equality of the displayed finite-window route and its public Real-facing factorization. □

Theorem 437.20 (Transported stationary window-seal public export certificate)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

Definition 437.2, Definition 437.3, Definition 437.4, Definition 437.5, Definition 437.6, and Definition 437.18 assemble one public export certificate over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *Every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the finite public rows $R, S, G, W, E, H, C, P, N$, and the certificate exports no quotient stream, host equality, selected limit, choice principle, or real-completeness datum.*

Proof. Project the accepted packet as $T = (R, S, G, W, E, H, C, P, N)$. The NameCert obligation theorem supplies the carrier, classifier, transported-window exactness, seal-exhaustion, and non-escape clauses for exactly these rows.

The readback determinacy theorem fixes the $\mathsf{Rat}^{\uparrow}$ input, the $\mathsf{StreamName}^{\uparrow}$ stationary window, and the $\mathsf{RegSeqRat}^{\uparrow}$ seal as one finite route. The Real factorization theorem then sends every $\mathsf{Real}^{\uparrow}$-facing consumer through the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row and that same seal route.

Use uniqueness to identify the public row determined by the displayed carrier and classifier data. The obligation package collects these clauses into one consumer-facing certificate, and ledger non-escape removes every coordinate not present in $R, S, G, W, E, H, C, P, N$.

Thus the exported certificate is public over the listed BEDC kernel and setup objects, while quotient streams, host equality, selected limits, choice, and real-completeness data remain outside its ledger surface. □

Theorem 437.21 (Transported stationary window-seal obligation readiness)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the checked NameCert *obligations, readback determinacy, and* $\mathsf{Real}^{\uparrow}$ *factorization rows assemble the obligation-readiness package for the Real and Completion route over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *This package consists of carrier admission for T, componentwise public readback from R through S and G, factorization of every* $\mathsf{Real}^{\uparrow}$*-facing read through W and G, the public export certificate, and ledger non-escape. It does not claim RealUp completeness, quotient-stream equality, host equality, selected limits, or choice.*

Proof. The carrier and classifier surface is the checked NameCert obligation row of Definition 437.2. It supplies the finite packet coordinates, the componentwise comparison rule, transported-window exactness, seal exhaustion, and the non-escape boundary.

The readback and real-facing routes are the two other checked rows. Apply Definition 437.3 to fix the transported $\mathsf{Rat}^{\uparrow}$ row, constant $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ finite-window seal as one public readback route. Apply Definition 437.4 to force every $\mathsf{Real}^{\uparrow}$ read through the displayed $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ packet and the same regular-rational seal row.

It remains to package these rows as the obligation-readiness surface. The public export certificate of Definition 437.20 collects the carrier, readback, factorization, uniqueness, and package clauses over the listed BEDC objects, while Definition 437.6 blocks quotient streams, host equality, choice, selected limits, and real-completeness data. Therefore the accepted packet exposes exactly the finite readiness package stated above. □

Theorem 437.22 (Transported stationary window-seal window source lock)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the rows R, S, G form one stationary source route through the displayed transport row H, continuation row C, provenance row P, and local naming row N. The $\mathsf{Rat}^{\uparrow}$ *classifier row R, constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row G cannot be read as separate sources by a downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer.*

Proof. Project the accepted carrier T. The rows R, S, and G are listed in the same finite packet as the transport, continuation, provenance, and naming rows.

Apply Definition 437.3. It fixes the transported $\mathsf{Rat}^{\uparrow}$ row, the constant $\mathsf{StreamName}^{\uparrow}$ window, and the $\mathsf{RegSeqRat}^{\uparrow}$ seal as one public readback route.

Now use the package and ledger rows. The route can be transported through H, followed through C, named by P, and exposed by N, but there is no carrier coordinate that separates R, S, or G into an independent source for a downstream consumer. □

Theorem 437.23 (Transported stationary window-seal stationary diagonal handoff)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet $T = (R, S, G, W, E, H, C, P, N)$, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row W consumes the same stationary source route R, S, G, H, C, P, N before the* $\mathsf{Real}^{\uparrow}$ *seal row E is read. Thus the diagonal window handoff is ordered:*

$$(R, S, G, H, C, P, N) \longrightarrow W \longrightarrow E.$$

No consumer may bypass W and read E directly from an ambient stream, quotient equality, selected limit, or completeness datum.

Proof. Begin with the source route of Definition 437.22. It identifies the transported rational row, stationary window row, and regular-rational seal row as one displayed route through H, C, P, N.

Apply Definition 437.4. Every $\mathsf{Real}^{\uparrow}$-facing read must factor through the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row W and the same regular-rational seal route.

The seal row E is therefore downstream of W. Ledger non-escape rules out a direct route from an ambient stream, quotient equality, selected limit, or completeness datum into E, so the displayed handoff has exactly the order stated above. □

Theorem 437.24 (Transported stationary window-seal Real seal non-escape package)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the Real and RegSeqRat handoff package is the finite assembly of the carrier row, readback determinacy, Real-facing factorization, uniqueness, public export certificate, and ledger non-escape:*

1. *the carrier is exactly* $R, S, G, W, E, H, C, P, N$*;*
2. *readback determinacy locks* R, S, G *to one transported stationary source route;*
3. *Real-facing factorization sends every seal read through* W *and* G *before* E*;*
4. *uniqueness identifies the public row determined by the displayed carrier and classifier data;*
5. *the ledger exports no quotient stream, host equality, selected limit, choice principle, or real-completeness datum.*

Proof. The carrier clause is the packet of Definition 437.1, and the NameCert obligations are those of Definition 437.2.

Readback determinacy is supplied by Definition 437.3; the source-lock theorem above repackages that determinacy as a single route through H, C, P, N.

For the Real-facing side, use Definition 437.23. It combines the Real factorization theorem with the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row, forcing the seal row E to be downstream of W and G.

Finally apply Definition 437.5, Definition 437.20, and Definition 437.6. These give the public row, the export package, and the non-escape boundary, which together form the exact Real and RegSeqRat handoff package stated above. □

Theorem 437.25 (Transported stationary window-seal envelope window exactness)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row* W *may be read only after the stationary source route*

$$R \longmapsto S \longmapsto G$$

has been fixed through the displayed hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The row* W *is therefore the finite envelope over the transported* $\mathsf{Rat}^{\uparrow}$ *classifier row* R*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G*, and it must be read before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is available to any completion-facing or real-facing consumer.*

Proof. Project the accepted carrier. The only rows available before the envelope are the displayed stationary route R, S, G and its transport, continuation, package, and local naming provenance H, C, P, N.

Apply Definition 437.22. It binds R, S, and G as one public source route, so a consumer cannot select an independent stream window or regular-rational seal before reading W.

Next use Definition 437.23. It places the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row after the stationary route and before the $\mathsf{Real}^{\uparrow}$ seal row.

Finally, Definition 437.6 removes the possible bypasses. Quotient streams, host equality, selected limits, choice, and real-completeness data are not carrier coordinates, so W is exactly the envelope window at the stated boundary. □

Theorem 437.26 (Transported stationary window-seal constant-window triad)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the rows* R*,* S*, and* G *form one constant-window triad under the displayed rows* H, C, P, N*. The transported* $\mathsf{Rat}^{\uparrow}$ *classifier, the constant* $\mathsf{StreamName}^{\uparrow}$ *window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal are transported, replayed, packaged, and named together; no downstream* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Completion}^{\uparrow}$ *consumer may read one of the three as a separate source.*

Proof. Start from the carrier and classifier obligations in Definition 437.2. They admit exactly the rows $R, S, G, W, E, H, C, P, N$ and compare accepted packets componentwise through the displayed transport and continuation rows.

Apply Definition 437.3. It identifies the readback of the transported rational row with the constant stream window and the regular-rational seal, all through the same public route.

Use Definition 437.22 to repackage that route under H, C, P, N. The structural rows can transport, replay, package, and name the triad, but they do not split it into separate source certificates.

The envelope and Real-facing rows consume the triad only through Definition 437.25 and Definition 437.4. Thus any completion-facing or real-facing read sees R, S, G as one constant-window triad. □

Theorem 437.27 (Transported stationary window-seal completion consumer boundary)**.** *Every completion-facing or real-facing consumer of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet factors through the finite public packet*

$$R, S, G, W, E, H, C, P, N.$$

The consumer first reads the constant-window triad R, S, G*, then the envelope row* W*, then the seal row* E*, with* H, C, P, N *carrying only displayed transport, continuation, package, and local naming data. The packet provides no quotient stream, host equality, selected limit, choice principle, or real-completeness datum.*

Proof. First assemble the source side. By Definition 437.26, the transported rational row, constant window row, and regular-rational seal row are one finite source triad under H, C, P, N.

Next assemble the envelope side. Apply Definition 437.25; it forces each completion-facing or real-facing route to read W after R, S, G and before E.

Now apply the checked Real-facing package. The Real factorization theorem, the stationary diagonal handoff theorem, and Definition 437.24 bind the seal read to the same rows $R, S, G, W, E, H, C, P, N$.

Finally close the consumer boundary with Definition 437.6. Since quotient streams, host equality, selected limits, choice, and real-completeness data are absent from the packet, every consumer route factors through the displayed finite BEDC packet and nothing outside it. □

Theorem 437.28 (Transported stationary window-seal tail-cofinality handoff)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Every completion-facing or real-facing read first factors through the transported stationary triad*

$$R \longmapsto S \longmapsto G,$$

then through the scheduled tail-cofinality route

$$E_T \longmapsto W_T \longmapsto D_T \longmapsto R_T \longmapsto L_T$$

with structural rows H_T, C_T, P_T, N_T, and only then through the finite envelope W and seal row E of T. The combined handoff exports no quotient-stream equality, selected host limit, host real equality, choice principle, hidden tail oracle, ambient completeness, or second completion route.

Proof. Begin with the local stationary source. By Definition 437.26, the rows R, S, G are one transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad under the displayed H, C, P, N rows.

Next apply Definition 437.25. It places the finite envelope row W downstream from that triad and upstream from the local seal row E, so the transported packet has no route that reaches W or E before the triad is fixed.

Now read the schedule packet on the same window and seal row. The real-seal readback theorem Definition 681.6 routes every schedule-facing $\mathsf{Real}^{\uparrow}$ read through E_T, W_T, D_T, R_T, L_T, and Definition 681.8 keeps the same route inside BHist, hsame, Cont, Pkg, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the local $\mathsf{NameCert}$ row.

For completion-facing consumers, use Definition 681.9. It places the completion seal read after the same scheduled route, and Definition 681.10 keeps the shared finite window and tail budget as the unique displayed factorization for the participating Real and Completion rows.

Finally combine the schedule exclusions with Definition 437.6. Neither packet contains a coordinate for quotient-stream equality, selected host limits, host real equality, choice, hidden tail oracles, ambient completeness, or another completion route. Thus every completion-facing or real-facing consumer must pass through the transported triad, the schedule rows, and then the finite envelope and seal row in the order stated. □

Lean4 ✓ `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealTailCofinalityHar`

Theorem 437.29 (Transported stationary window-seal root envelope exhaustion)**.** *Every root consumer read of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N)$$

passes first through the transported seed route

$$R, S, G, H, C, P, N,$$

then through the finite envelope row W, and then through the $\mathsf{Real}^{\uparrow}$*-facing seal row E. Any* $\mathsf{Completion}^{\uparrow}$*-facing or tail-cofinality consumer that accepts the packet must reuse this same envelope route:*

$$(R, S, G, H, C, P, N) \longrightarrow W \longrightarrow E.$$

It cannot introduce quotient streams, host equality, selected limits, choice, hidden tail oracles, real-completeness data, or a second completion route.

Proof. Start with the transported seed. By Definition 437.26, the rows R, S, G form one constant-window triad carried only by the displayed transport, continuation, package, and local naming rows H, C, P, N.

Next read the finite envelope. Apply Definition 437.25; it places W downstream from the transported triad and upstream from the $\mathsf{Real}^{\uparrow}$ seal row E. The root read therefore reaches E only after the same finite envelope has been fixed.

For completion-facing consumers, use Definition 437.27. It forces each completion-facing or real-facing route to factor through $R, S, G, W, E, H, C, P, N$, with W between the source triad and the seal row. For tail-cofinality consumers, Definition 437.28 keeps the schedule rows downstream of the transported triad and upstream of the same envelope and seal rows.

Finally apply Definition 437.6. Since quotient streams, host equality, selected limits, choice, hidden tail oracles, real-completeness data, and second completion routes are not packet coordinates, every root consumer read is exhausted by the displayed route. □

Theorem 437.30 (Transported stationary window-seal root readback totality)**.** *Every root-facing read of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N)$$

passes through the transported $\mathsf{Rat}^{\uparrow}$ *row* R*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet* W*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E *in that order. Thus the root-facing readback route is total exactly along*

$$R \longrightarrow S \longrightarrow G \longrightarrow W \longrightarrow E,$$

with H, C, P, N *carrying only the displayed transport, continuation, provenance, and local naming rows.*

Proof. Unpack the carrier in Definition 437.1. Its visible coordinates already separate the transported rational seed row R, the constant stream window S, the regular-rational seal row G, the finite envelope W, and the real-facing seal row E.

The seed-window part is fixed by Definition 437.8. It makes every root-facing seed read pass from the transported $\mathsf{Rat}^{\uparrow}$ row to the induced $\mathsf{StreamName}^{\uparrow}$ window and then to the $\mathsf{RegSeqRat}^{\uparrow}$ seal row.

Next apply Definition 437.4 and Definition 437.25. These place W after the same seed-window-seal route and place the $\mathsf{Real}^{\uparrow}$ seal row E after W.

No other root-facing coordinate remains. The transport, continuation, provenance, and local naming rows H, C, P, N only record the displayed route, so the readback is total along the stated order. □

Theorem 437.31 (Transported stationary window-seal root package exactness)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the package row* P *is exactly the provenance boundary over*

$$\mathsf{Rat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{FiniteWindowEnvelope}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{NameCert}.$$

It records no provenance outside these rows, and every public export of the packet reads P *only through this displayed boundary.*

Proof. Begin with the obligation surface in Definition 437.2. Its carrier, classifier, transported-window, seal-exhaustion, and non-escape clauses list exactly the source rows used by the packet.

The public export certificate Definition 437.20 then collects these rows over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The only package coordinate available in the carrier is P.

Therefore P is the provenance boundary for the displayed BEDC objects. Since the same certificate excludes quotient streams, host equality, selected limits, choice, and real-completeness data, no external provenance can be imported through the package row. □

Theorem 437.32 (Transported stationary window-seal root consumer nonescape)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet*

$$T = (R, S, G, W, E, H, C, P, N)$$

factors through the displayed finite packet. Such a consumer cannot import quotient streams, host equality, choice, selected host limits, hidden tail oracles, real-completeness data, or a second completion route.

Proof. Start with Definition 437.6. It inspects the finite packet ledger and removes quotient streams, host equality, selected limits, choice, hidden tail oracles, real-completeness data, and alternate completion routes from the available coordinates.

For completion-facing reads, apply Definition 437.27. It forces each $\mathsf{Completion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to factor through $R, S, G, W, E, H, C, P, N$ rather than through an ambient completion object.

Finally use Definition 437.34. The shared $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ seal budget is exhausted by the displayed packet and its schedule-facing rows, so no downstream consumer can attach another source outside the finite route. □

Theorem 437.33 (Transported stationary window-seal tail schedule envelope order)**.** *Let*

$$T = (R, S, G, W, E, H, C, P, N)$$

be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let*

$$Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Every accepted completion-facing or real-facing packet read must encounter the transported rows* R, S, G *before the TailCofinalitySchedule rows* $E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T$*, and must encounter those schedule rows before the envelope and seal rows* W, E *of* T*. Hence the only exposed root-to-completion order is*

$$R, S, G, H, C, P, N \longrightarrow E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T \longrightarrow W \longrightarrow E.$$

Proof. First fix the transported source. By Definition 437.26, the rows R, S, G are the constant transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad under the displayed H, C, P, N rows. Thus any accepted read that uses the packet begins with the transported triad rather than with a schedule row.

Next attach the shared tail schedule. The handoff theorem Definition 437.28 places the TailCofinalitySchedule route after the transported triad for both completion-facing and real-facing consumers, while retaining the same finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ seal row.

Finally apply Definition 437.29. It places the finite envelope row W and the $\mathsf{Real}^{\uparrow}$-facing seal row E after every transported root read. Combining the three constraints gives exactly the displayed order and leaves no earlier access to W or E. □

Theorem 437.34 (Transported stationary window-seal shared window budget exhaustion)**.** *For any accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$ *and any accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *consumed on*

its same finite window and seal row, the shared $\mathsf{StreamName}^{\uparrow}$ *window budget and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal budget are exhausted by*

$$R, S, G, H, C, P, N, E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T, W, E.$$

No accepted consumer can append a hidden tail oracle, a second $\mathsf{StreamName}^{\uparrow}$ *family, an additional* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row, quotient-stream equality, selected host limits, host equality, choice, or ambient real-completeness data to that budget.*

Proof. The root packet owns the finite window and seal budget through Definition 437.25: the transported triad reaches W and E only through the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. Thus the root side has no spare coordinate where a separate window family or seal family could be stored.

The tail-cofinality handoff Definition 437.28 then consumes the schedule packet on that same window and seal row. Its schedule rows therefore refine the already displayed budget instead of creating a new one.

It remains to exclude external resources. Apply Definition 437.6; the packet ledger has no coordinate for hidden tail oracles, quotient-stream equality, selected limits, host equality, choice, ambient completeness, or another completion route. Hence the displayed rows exhaust the shared window and seal budget. □

Theorem 437.35 (Transported stationary window-seal completion route uniqueness)**.** *Let T and Q be as in Definition 437.34. Any completion-facing consumer and any* $\mathsf{Real}^{\uparrow}$*-facing consumer that preserve the displayed transported rows, schedule rows, finite envelope, and seal row determine the same route*

$$(R, S, G, H, C, P, N) \longrightarrow (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T) \longrightarrow W \longrightarrow E.$$

Consequently the transported stationary window-seal packet has a unique completion route at the exposed NameCert boundary.

Proof. Start at the consumer boundary. By Definition 437.27, every completion-facing or real-facing consumer of T factors through the displayed transported rows and cannot bypass $R, S, G, W, E, H, C, P, N$.

The root envelope exhaustion theorem Definition 437.29 fixes the root route from the transported rows to W and then to E. The tail schedule envelope order theorem Definition 437.33 inserts the schedule rows in the only position compatible with both the transported triad and the envelope boundary.

Finally Definition 437.34 rules out any second window family, second seal budget, hidden tail oracle, or ambient completion datum. Two consumers preserving the displayed rows therefore preserve the same finite route, so the completion route is unique at the exposed boundary. □

Theorem 437.36 (Transported stationary window-seal schedule-envelope determinacy)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet and let $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Any completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer preserving the transported source rows R, S, G and the tail schedule rows of Q must read the finite envelope row W before it reads the* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Equivalently, the consumer route has the deterministic exposed order

$$R, S, G, H, C, P, N \longrightarrow E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T \longrightarrow W \longrightarrow E.$$

Proof. The transported source is fixed by the constant-window triad and the tail-cofinality handoff. These place the $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows before the schedule packet and force the schedule packet to use the same displayed finite window and seal row.

The finite envelope position is then fixed by Definition 437.29. It sends every accepted root read through W before the $\mathsf{Real}^{\uparrow}$-facing seal row E.

Finally Definition 437.33 and Definition 437.34 exclude a second schedule, a second finite envelope, or an earlier seal read. The displayed order is therefore the only deterministic completion-facing order. □

Theorem 437.37 (Transported stationary window-seal Real RegSeqRat source totality)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$*-facing seal row* E *is sourced only through the transported* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *triad* R, S, G *together with the displayed transport, package, and NameCert rows* H, C, P, N*. No* $\mathsf{Real}^{\uparrow}$ *consumer of* E *can draw a source from quotient-stream equality, host equality, selected limits, choice, ambient real completeness, hidden tail oracles, or an independent completion object.*

Proof. Start with the envelope exactness theorem. The row E is exposed as the seal-facing $\mathsf{Real}^{\uparrow}$ row reached from the finite $\mathsf{StreamName}^{\uparrow}$ envelope W and the $\mathsf{RegSeqRat}^{\uparrow}$ seal row, and both are carried by the transported triad R, S, G.

The provenance and nonescape theorems then identify the available BEDC rows: $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. There is no additional carrier coordinate from which another real source could be read.

For completion-facing uses, Definition 437.28 and Definition 437.27 keep the same source triad upstream of the shared schedule and seal. Thus every $\mathsf{Real}^{\uparrow}$-facing read of E is total over exactly the displayed transported source route. □

Theorem 437.38 (Transported stationary window-seal NameCert route exhaustion)**.** *Let* T *and* Q *be as above. The exposed* $\mathsf{NameCert}$ *route surface of* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *is exhausted by the transported source triad, the tail schedule packet, the finite envelope, and the* $\mathsf{Real}^{\uparrow}$ *seal row:*

$$(R, S, G, H, C, P, N), (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T), W, E.$$

Every accepted completion-facing or $\mathsf{Real}^{\uparrow}$*-facing consumer preserving these rows determines the unique route of Definition 437.35, and every* $\mathsf{Real}^{\uparrow}$ *seal source is the triad-total source of Definition 437.37.*

Proof. First apply Definition 437.36. It fixes the order from the transported rational source through the tail schedule and finite envelope to the $\mathsf{Real}^{\uparrow}$ seal.

Next apply Definition 437.37. It identifies the source of the seal row with the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad, with no external real or completion source.

Finally use Definition 437.35. Since the shared window budget is already exhausted, the NameCert surface has no remaining coordinate for another source, schedule, envelope, or seal. The displayed rows therefore exhaust the deterministic route surface. □

Theorem 437.39 (Transported stationary window-seal source triad totality)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The source triad* R, S, G *is total over the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows through the displayed support rows* H, C, P, N*: every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing route that reads* T *first reads the transported rational classifier* R*, the constant stream-name window* S*, and the regular rational seal row* G *through the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. No*

route may read a completion object, quotient stream, selected host limit, host equality row, choice datum, or real-completeness witness before this triad is fixed.

Proof. Project the accepted packet through Definition 437.2. Its carrier and classifier obligations expose R, S, G together with H, C, P, N as the named source surface of the transported stationary window seal.

Use Definition 437.26 and Definition 437.45. They identify the $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows as one ordered source triad before any tail schedule, finite envelope, or $\mathsf{Real}^{\uparrow}$ seal is consumed.

The L10 handoff basis Definition 437.48 then places every completion-facing or $\mathsf{Real}^{\uparrow}$-facing read behind this same triad. The non-escape clauses of Definition 437.6 exclude quotient streams, selected host limits, host equality, choice, and real-completeness data from the source surface. □

Theorem 437.40 (Transported stationary window-seal envelope seal exactness obligation)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The finite envelope row W and the* $\mathsf{Real}^{\uparrow}$ *seal row E are exact only after the source triad R, S, G, H, C, P, N and the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row have been fixed. Thus every accepted envelope-seal consumer reads the route*

$$R, S, G, H, C, P, N \to W \to E$$

and no consumer may obtain W or E from an earlier real row, quotient-stream equality, host equality, selected host limit, choice datum, hidden tail oracle, or ambient completeness principle.

Proof. Begin with source-triad totality above. It fixes R, S, G, H, C, P, N as the only source available before the finite envelope and seal rows are read.

Apply Definition 437.25 and Definition 437.29. These theorems put the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row W downstream of the triad and upstream of the $\mathsf{Real}^{\uparrow}$ seal row E.

Finally use Definition 437.55 and Definition 437.59. They force every downstream envelope-seal read to preserve the same window and seal rows, while the listed excluded objects remain outside the accepted packet. □

Theorem 437.41 (Transported stationary window-seal completion handoff exhaustion)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff is exhausted by the transported source triad, finite envelope, seal row, structural support, and displayed tail-cofinality schedule rows:*

$$R, S, G, W, E, H, C, P, N, E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T.$$

There is no remaining handoff coordinate for quotient streams, host equality, choice, selected limits, hidden tail oracles, real-completeness data, or a second completion source.

Proof. First assemble the root handoff surface. The source-triad totality and envelope-seal exactness obligations above give R, S, G, H, C, P, N, then W, then E as the ordered local route.

Attach the schedule rows with Definition 437.28, Definition 437.33, and Definition 437.58. These bind the tail-cofinality packet to the same finite stream window and regular rational seal before the envelope-seal handoff is opened.

Now apply Definition 437.60 and Definition 437.62. They make downstream $\mathsf{Completion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers read exactly the displayed root rows together with the displayed schedule rows.

The completion route is unique by Definition 437.35. Since the accepted packets expose no quotient streams, host equality, choice, selected limits, hidden tail oracles, real-completeness data, or second completion source, the displayed rows exhaust every handoff. □

Theorem 437.42 (Transported stationary window-seal tail route totality)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet and let* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Every accepted completion-facing route that preserves the transported source rows of* T *and the schedule rows of* Q *is total over the displayed route*

$$(R, S, G, H, C, P, N) \longrightarrow (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T) \longrightarrow W \longrightarrow E.$$

It has no additional tail oracle, second $\mathsf{StreamName}^{\uparrow}$ *window, second* $\mathsf{RegSeqRat}^{\uparrow}$ *seal, quotient-stream equality, selected host limit, choice principle, ambient real-completeness witness, or independent* $\mathsf{Completion}^{\uparrow}$ *carrier.*

Proof. First fix the transported source route. The carrier obligation and Definition 437.26 identify R, S, G as one transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route under H, C, P, N.

Next attach the schedule. Apply Definition 437.28; it places the TailCofinalitySchedule packet on the same finite window and seal row and routes completion-facing consumers through $E_T, W_T, D_T, R_T, L_T, H_T, C$ after the transported triad.

Now read the envelope and seal. The completion-consumer boundary and Definition 437.29 force the finite envelope W and the $\mathsf{Real}^{\uparrow}$ seal row E to be read after the transported source and schedule rows.

Finally use Definition 437.34 and Definition 437.35. The shared budget has no remaining coordinate for another source, schedule, envelope, completion carrier, or real-completeness datum, so the displayed completion-facing route is total. □

Theorem 437.43 (Transported stationary window-seal Real observation budget coherence)**.** *For accepted packets* $T = (R, S, G, W, E, H, C, P, N)$ *and* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *as above, the* $\mathsf{Real}^{\uparrow}$ *observation budget exported by* T *is coherent with the tail schedule budget of* Q*. Every* $\mathsf{Real}^{\uparrow}$*-facing read of* E *uses the transported* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *source route* R, S, G, H, C, P, N*, the scheduled tail rows* $E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T$*, and the finite envelope row* W *before it reaches* E*.*

In particular, no accepted $\mathsf{Real}^{\uparrow}$ *observer can split the observation budget into an envelope-only real row, a schedule-only real row, a selected host limit, quotient-stream equality, host real equality, choice, ambient completeness, or a second completion source.*

Proof. Begin with the Real factorization. By Definition 437.4 and Definition 437.16, every $\mathsf{Real}^{\uparrow}$ read of E factors through the same $\mathsf{RegSeqRat}^{\uparrow}$ seal row G and the finite envelope W.

The tail schedule is attached to exactly that window and seal row. Use Definition 437.28 to place Q between the transported source triad and the envelope-seal pair, and use Definition 437.33 to keep the schedule rows upstream of W and E.

The budget comparison is then forced by Definition 437.34. It states that the shared $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ budget is exhausted by the transported rows, the schedule rows, and the envelope-seal rows.

Finally apply ledger non-escape and completion-route uniqueness. Since neither packet exposes quotient streams, selected limits, host real equality, choice, ambient completeness, or another completion source, every $\mathsf{Real}^{\uparrow}$ observation budget is the single coherent budget displayed above. □

Theorem 437.44 (Transported stationary window-seal stationary window ledger exhaustion)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ and $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ be accepted packets as above. The stationary window ledger available to* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *and its tail schedule consumer is exhausted by*

$$R, S, G, H, C, P, N, E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T, W, E.$$

Every accepted NameCert route preserving the same transported source, schedule, finite envelope, and $\mathsf{Real}^{\uparrow}$ *seal is the route of Definition 437.35; there is no unused stationary-window coordinate for another tail schedule, another finite envelope, another real seal, or an independent completion route.*

Proof. First account for the root ledger. The carrier obligation, root source lock, and root ledger exhaustion theorems identify the available transported window rows as $R, S, G, W, E, H, C, P, N$ and exclude all off-packet real and completion data.

Next account for the schedule-facing ledger. The tail-cofinality handoff consumes the accepted packet Q on the same finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ seal row, and the shared-window budget exhaustion theorem records the exact schedule rows available to the combined route.

Now combine the two inventories. The tail route totality theorem above places the transported source rows, schedule rows, finite envelope, and real seal into one completion-facing route, while Real observation budget coherence keeps the $\mathsf{Real}^{\uparrow}$ read on that same route.

Finally apply NameCert route exhaustion and completion-route uniqueness. Any accepted route preserving the displayed rows has no coordinate left for another schedule, envelope, seal, or completion object, so the stationary window ledger is exhausted by the stated finite packet. □

Theorem 437.45 (Transported stationary window-seal root completion source order)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Every completion-facing root-obligation route for T first fixes the transported* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *source triad R, S, G, H, C, P, N of chapter 126 and chapter 301, then fixes the tail schedule packet Q of chapter 681, and only then reads the finite envelope W and the* $\mathsf{Real}^{\uparrow}$ *seal row E of chapter 38. The order is a root-obligation package for completion consumers, not a proof of ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the completion-consumer boundary. By Definition 437.27, every completion-facing consumer is forced through the displayed transported rows of T before it can read a seal-facing row. The source rows are precisely the transported rational route, the finite stream-name route of chapter 126, and the regular rational seal route of chapter 301.

Next attach the schedule. The handoff theorem Definition 437.28 places the packet of chapter 681 after the transported source triad and before the envelope-seal read.

Finally use Definition 437.37. It identifies the $\mathsf{Real}^{\uparrow}$ row of chapter 38 as sourced by the same transported $\mathsf{RegSeqRat}^{\uparrow}$ seal, so the finite envelope and real seal are downstream rows of the root package rather than a real-completeness principle. □

Theorem 437.46 (Transported stationary window-seal finite-window schedule coherence)**.** *For accepted packets $T = (R, S, G, W, E, H, C, P, N)$ and $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ as above, the tail schedule rows and finite envelope rows consume the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *seal. Thus the completion-facing root-obligation route has one* $\mathsf{StreamName}^{\uparrow}$ *window from chapter 126, one* $\mathsf{RegSeqRat}^{\uparrow}$ *seal from chapter 301, one* $\mathsf{TailCofinalitySchedule}^{\uparrow}$

packet from chapter 681, and one $\mathsf{Real}^{\uparrow}$ *seal-facing row from chapter 38, with no split between schedule and envelope budgets.*

Proof. Start with the shared source. The completion-consumer boundary Definition 437.27 keeps every accepted completion-facing read on the transported triad R, S, G, H, C, P, N, so the finite window is the one supplied by chapter 126 and the seal row is the one supplied by chapter 301.

The schedule handoff then consumes exactly those rows. By Definition 437.28, the packet of chapter 681 is attached to the same finite window and the same regular rational seal before the envelope is read.

The real-facing source theorem Definition 437.37 sends the seal row of chapter 38 back to that same transported $\mathsf{RegSeqRat}^{\uparrow}$ source. Hence the schedule and envelope rows are coherent over one finite window and one seal, with no second budget from a completion object. □

Theorem 437.47 (Transported stationary window-seal Real completion refusal cover)**.** *For accepted packets* $T = (R, S, G, W, E, H, C, P, N)$ *and* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *as above, the completion-facing route determined by the transported source triad, the tail schedule, the finite envelope, and the* $\mathsf{Real}^{\uparrow}$ *seal refuses every off-packet route through quotient streams, selected host limits, choice, ambient completeness, or host equality. Its visible sources remain the* $\mathsf{StreamName}^{\uparrow}$ *route of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal of chapter 301, the schedule packet of chapter 681, and the* $\mathsf{Real}^{\uparrow}$ *row of chapter 38.*

Proof. First read the route boundary. The theorem Definition 437.27 excludes any completion-facing read that bypasses the displayed packet T, so quotient streams, host equality, selected limits, choice, and ambient completeness have no entry point at the root boundary.

Next impose the tail schedule. By Definition 437.28, the schedule packet of chapter 681 is inserted only after the finite $\mathsf{StreamName}^{\uparrow}$ row of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ seal of chapter 301 are fixed.

Finally apply Definition 437.37. The $\mathsf{Real}^{\uparrow}$ seal row of chapter 38 is sourced by the transported regular-rational seal, so the route covers all completion-facing refusal clauses without asserting any host real-completeness theorem. □

Theorem 437.48 (Transported stationary window-seal L10 handoff obligation basis)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Its L10 handoff basis for Real and Completion consumers is the finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{FiniteWindowEnvelope}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

read through the visible rows $R, S, G, W, E, H, C, P, N$ *and through the sibling surfaces of chapter 37, chapter 126, chapter 301, chapter 291, chapter 365, chapter 38, and chapter 681. The basis is an obligation route for later checked carriers; it is not a host completion theorem.*

Proof. Begin with the local obligation theorem Definition 437.2. It exposes the accepted packet only through the carrier rows $R, S, G, W, E, H, C, P, N$, with componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming.

Next use Definition 437.4. The $\mathsf{Real}^{\uparrow}$ read cannot appear before the regular-rational seal and finite envelope rows; it must pass through the same $\mathsf{RegSeqRat}^{\uparrow}$ source and the same $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row already displayed by the packet.

Finally attach the completion-facing side by Definition 437.27 and Definition 437.28. These place every completion consumer after the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source, after the dyadic and finite-window ledger rows, and before any off-packet real-completeness datum could be read. The result is exactly the stated finite handoff basis. □

Theorem 437.49 (Transported stationary window-seal dyadic window ledger totality)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the dyadic window ledger available to the Real and Completion route is total only through the displayed rows* $R, S, G, W, E, H, C, P, N$ *and the sibling certificates for* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*. In particular, any accepted dyadic tolerance, finite-window, or regular-readback consumer must read the finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal of chapter 301, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of chapter 291 before it can use* W *or* E*.*

Proof. First project the accepted packet. The obligation theorem Definition 437.2 lists exactly the root rows $R, S, G, W, E, H, C, P, N$ and gives no separate dyadic or window carrier outside them.

The finite window and regular seal are then fixed by Definition 437.4 and by the sibling StreamName and RegSeqRat chapters. The dyadic ledger is read only as the tolerance and observation surface compatible with those same rows, through chapter 291.

The route boundary Definition 437.27 and the schedule handoff Definition 437.28 rule out any second window budget, second regular seal, or detached dyadic ledger. Hence the dyadic window ledger is total exactly over the displayed packet and the named sibling certificates. □

Theorem 437.50 (Transported stationary window-seal Real RegSeq consumer exactness)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* T *factors through the finite route from transported* $\mathsf{Rat}^{\uparrow}$ *through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *displayed in Definition 437.48. No such consumer can import quotient streams, host equality, selected limits, choice, hidden tail oracles, or real-completeness data.*

Proof. Start with the Real-facing side. By Definition 437.4, every $\mathsf{Real}^{\uparrow}$ read of E factors through the same $\mathsf{RegSeqRat}^{\uparrow}$ seal row and the finite envelope W, so there is no direct host-real read.

Now use Definition 437.27. It forces every $\mathsf{Completion}^{\uparrow}$ consumer back to the displayed root packet rather than to quotient streams, host equality, selected limits, choice, or ambient real completeness.

Finally apply Definition 437.28. The only tail information available to the consumer is the scheduled sibling route of chapter 681; no hidden tail oracle is introduced. Together with the StreamName, RegSeqRat, DyadicRatCore, FiniteWindowEnvelope, and Real sibling chapters, this gives the exact finite consumer factorization claimed above. □

Theorem 437.51 (Transported stationary window-seal open-phase exit boundary)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every accepted* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* T *that preserves the displayed rows* $R, S, G, W, E, H, C, P, N$ *lands on the same four-face Phase Real and Completion route as Definition 38.530 and Definition 38.534. Its terminal boundary is the finite route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows. No quotient stream, selected host limit, host equality, choice row, ambient real-completeness datum, or standard bridge row can replace one of those four faces.*

Proof. First apply Definition 437.48. The accepted transported packet exposes only the finite handoff route through $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, read through the rows $R, S, G, W, E, H, C, P, N$ and their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Next use Definition 437.50. It forces every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer preserving those rows to enter the Real-facing route through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

and finite-window surfaces, without quotient streams, selected limits, choice, host equality, hidden tail data, or real-completeness input.

Now compare this transported route with the open-phase Real route. Theorem 38.530 places every public $\mathsf{Real}^{\uparrow}$ seal read in the open phase after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ sibling surfaces and before the local $\mathsf{Real}^{\uparrow}$ naming row.

Finally apply Definition 38.534. The terminal Phase Real and Completion packet is determined by the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Since the transported consumer preserves $R, S, G, W, E, H, C, P, N$ and supplies no extra accepted face, it reaches exactly that four-face exit boundary. □

Theorem 437.52 (Transported stationary window-seal root carrier admission)**.** *Let $T = (R, S, G, W, E, H, C, P, N$ be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet. The root Real and Completion handoff route admits exactly the displayed carrier rows $R, S, G, W, E, H, C, P, N$: the transported $\mathsf{Rat}^{\uparrow}$ classifier R, constant $\mathsf{StreamName}^{\uparrow}$ window S, $\mathsf{RegSeqRat}^{\uparrow}$ seal row G, finite envelope W, $\mathsf{Real}^{\uparrow}$ seal row E, transport row H, route row C, provenance row P, and local naming row N. This admission is the carrier and $\mathsf{NameCert}$ obligation surface of Definition 437.1 and Definition 437.2, read through the L10 handoff basis of Definition 437.48.*

Proof. Project the accepted carrier. The carrier definition supplies exactly the nine visible rows $R, S, G, W, E, H, C, P, N$, and the NameCert obligation theorem identifies the transport, continuity, package, and local naming obligations attached to them.

Next apply the L10 handoff basis. It orders the same rows through the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, so the Real and Completion handoff has no row outside the displayed packet.

Thus carrier admission for the root route is precisely the projection of T together with its displayed $\mathsf{NameCert}$ obligations. □

Theorem 437.53 (Transported stationary window-seal window envelope readback)**.** *For accepted packets $T = (R, S, G, W, E, H, C, P, N)$ and $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ sharing the displayed finite window and seal rows, the finite envelope W and the $\mathsf{Real}^{\uparrow}$ seal row E are readable only after the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and tail-schedule rows have been fixed. Equivalently, the readback order is the root completion source order of Definition 437.45 together with the finite-window schedule coherence of Definition 437.46.*

Proof. Begin with the source order. The root completion source theorem fixes the transported rational source, the finite stream-name window, and the regular rational seal before any envelope or Real-facing row is read.

Attach the dyadic and schedule rows using the L10 handoff basis and the tail handoff theorem. These place the dyadic ledger and tail-schedule packet between the source triad and the envelope-seal boundary.

Finally use finite-window schedule coherence. Since the schedule and envelope consume the same finite $\mathsf{StreamName}^{\uparrow}$ window and the same $\mathsf{RegSeqRat}^{\uparrow}$ seal, W and E cannot be read from a separate window budget or from an earlier Real-facing source. □

Theorem 437.54 (Transported stationary window-seal completion consumer exhaustion)**.** *Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer accepted by a $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet factors through the displayed route*

$$R, S, G \longrightarrow Q \longrightarrow W, E \longrightarrow H, C, P, N.$$

No such consumer can import quotient streams, host equality, selected limits, choice, ambient completeness, or an independent completion carrier.

Proof. First use the carrier admission theorem above. It gives the consumer only the displayed packet rows and their NameCert obligations.

Next combine the root completion source order with the window envelope readback theorem. The source triad, tail schedule, finite envelope, and Real seal form one ordered route; every Real-facing or Completion-facing read must pass through it.

The refusal clauses then follow from the completion-consumer boundary and the Real completion refusal cover. Quotient streams, host equality, selected limits, choice, ambient completeness, and an independent completion carrier are not coordinates of the packet and are not supplied by any sibling handoff row. □

Theorem 437.55 (Transported stationary window-seal root envelope readback totality)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite window and regular-rational seal. Then every* $\mathsf{Completion}^{\uparrow}$*-facing or* $\mathsf{Real}^{\uparrow}$*-facing readback of the root packet is total only along the ordered finite route*

$$R, S, G, H, C, P, N \longrightarrow E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T \longrightarrow W \longrightarrow E.$$

In particular, the readback cannot bypass the transported $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, cannot read the finite envelope before the tail schedule rows, and cannot read the* $\mathsf{Real}^{\uparrow}$ *seal row before the shared-window budget has been exhausted.*

Proof. Start with the root carrier admission and the L10 handoff basis. These expose exactly the transported source rows R, S, G, the transport-package-name rows H, C, P, N, and the sibling route through $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next attach the readback order. The root completion source order fixes the source triad before any envelope read, and the tail-cofinality handoff places the schedule packet between the transported source and the finite envelope.

Finally use envelope exhaustion and shared-window budget exhaustion. They force the route to pass through W and then E, while excluding a second window family, second seal budget, hidden tail oracle, or off-route readback source. □

Theorem 437.56 (Transported stationary window-seal completion seal lock)**.** *For accepted packets* T *and* Q *as in Definition 437.55, the completion-facing seal exposed by* T *is locked to the same finite route. Any consumer that reaches the* $\mathsf{Completion}^{\uparrow}$ *boundary through* T *and preserves the displayed rows determines the completion route of Definition 437.35; it has no second completion route, no quotient-stream carrier, no host equality carrier, no selected limit, no choice datum, and no ambient completeness datum.*

Proof. First read the seal through the deterministic schedule-envelope order. The consumer must pass from the transported source and naming rows through the tail schedule rows before it can reach W and E.

The Real RegSeqRat source totality theorem then identifies the $\mathsf{Real}^{\uparrow}$ seal source with the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad, rather than with a host-real or quotient-stream source.

The completion route uniqueness theorem supplies the lock. Since shared-window budget exhaustion has already consumed the only finite window and seal budget available to the packet, there is no coordinate left for another completion carrier or a second route to the same seal. □

Theorem 437.57 (Transported stationary window-seal downstream route readiness)**.** *The* TransportedStationaryW *root surface is ready for every downstream* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, *and* $\mathsf{Completion}^{\uparrow}$ *consumer that preserves the accepted packet rows. Such a consumer must use the single finite BEDC route*

$$R, S, G, H, C, P, N \longrightarrow E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T \longrightarrow W \longrightarrow E,$$

and the NameCert *route surface is exhausted by exactly these rows. No downstream consumer receives quotient streams, host equality, selected limits, choice, ambient completeness, hidden tail oracles, or an independent completion route from this root chapter.*

Proof. Combine root envelope readback totality with completion seal lock. The first theorem gives the only admissible order from the transported source and naming rows through the tail schedule, finite envelope, and Real seal; the second theorem locks any completion-facing read to that same route.

Now apply NameCert route exhaustion. The displayed rows already account for the carrier, classifier, transport, continuity, package, local naming, tail schedule, finite envelope, and seal obligations.

It remains only to identify the downstream surface. The named downstream chapters consume precisely those rows as finite BEDC inputs, so preserving the packet rows preserves the route. Since the route surface is exhausted, none of the refused host or quotient data can be supplied by the root packet. □

Theorem 437.58 (Transported stationary window-seal completion source triad closure)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet, and let* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. Before any finite-envelope read of* W *or* $\mathsf{Real}^{\uparrow}$ *seal read of* E*, the transported* $\mathsf{Rat}^{\uparrow}$ *row* R*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G *form the closed source triad for the route.*

Proof. Begin with Definition 437.55. It orders every completion-facing or Real-facing readback through the transported rows before the tail schedule, finite envelope, and seal rows can be reached.

Now apply Definition 437.56. The lock prevents a completion consumer from choosing another completion carrier, quotient stream, host equality row, selected limit, choice datum, or ambient completeness witness before reading the seal.

The accepted schedule packet Q is therefore downstream of the already fixed source rows. Since the schedule is consumed on the same finite window and regular-rational seal, the triad R, S, G is closed as the source surface before W or E is read. □

Theorem 437.59 (Transported stationary window-seal envelope Real row totality)**.** *Let* T *and* Q *be as in Definition 437.58. Every* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *read and every* $\mathsf{Real}^{\uparrow}$*-facing read factors first through the closed source triad* R, S, G *and then through the schedule rows* $E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T$ *before reaching* W *and* E*. No such read receives a quotient stream, host equality, selected limit, choice principle, ambient completeness principle, or independent completion carrier.*

Proof. Use Definition 437.55 to fix the route order. The theorem places the transported source triad before the schedule rows, the schedule rows before the finite envelope, and the finite envelope before the Real seal row.

Next use Definition 437.56. It identifies the completion-facing seal with that same finite route, so a completion consumer cannot supply a separate carrier or host-level equality source while preserving the accepted packet rows.

Finally invoke Definition 437.57. The downstream surface already lists the permitted StreamName, RegSeqRat, FiniteWindowEnvelope, Real, TailCofinalitySchedule, and Completion consumers. Thus every envelope or Real read is total through the displayed triad and schedule rows before it reaches W and E. □

Theorem 437.60 (Transported stationary window-seal downstream obligation basis)**.** *For accepted packets T and Q as above, the downstream obligation basis consumed by* $\mathsf{Completion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *routes is exactly the pair of rows supplied by Definition 437.58 and Definition 437.59: the closed transported source triad R, S, G, followed by the accepted schedule rows and then by W, E. This basis is a finite BEDC obligation surface, not a quotient stream, host real equality, selected limit, choice principle, ambient completeness principle, or independent completion object.*

Proof. First take the source part from Definition 437.58. It supplies the transported $\mathsf{Rat}^{\uparrow}$, constant $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows before any envelope or Real seal read.

Next take the envelope and seal part from Definition 437.59. It routes every $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ read through the accepted schedule rows before reaching W and E.

The downstream route theorem Definition 437.57 packages these rows as the consumer surface. Hence Completion and Real routes consume precisely this finite obligation basis, and no off-route completion or host-real datum is added. □

Theorem 437.61 (Transported stationary window-seal window envelope source lock)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row W is source-locked to the transported* $\mathsf{StreamName}^{\uparrow}$ *row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row G inside the finite* BHist *carrier of T: every envelope-facing read must pass through the displayed* hsame *rows,* Cont *transports,* Pkg *provenance row P, and local* $\mathsf{NameCert}$ *row N before it reaches W. No accepted read of W supplies an independent stream window, regular-rational seal, host equality, quotient stream, selected limit, choice datum, or ambient completeness witness.*

Proof. Begin with the local obligation theorem for T. It exposes the accepted packet only through $R, S, G, W, E, H, C, P, N$, with the finite BHist carrier, displayed hsame and Cont rows, package provenance, and local naming already fixed.

Next apply the window-source and envelope-window exactness theorems. They identify the constant stream window and the regular-rational seal as the only source rows from which the finite envelope can be read.

Finally use root consumer nonescape. Since the Pkg and $\mathsf{NameCert}$ rows name only the displayed BEDC sources, an envelope-facing consumer has no coordinate from which to read a second window, a second seal, or any host-level completion datum. □

Theorem 437.62 (Transported stationary window-seal Real completion route totality)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer preserving T has the total finite route*

$$\mathsf{BHist}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{FiniteWindowEnvelope}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert},$$

instantiated by the rows S, G, W, E, H, C, P, N of T. The route is total for these consumers because each read either factors through the finite envelope W before the $\mathsf{Real}^{\uparrow}$ *row E, or is refused as outside the accepted packet surface.*

Proof. First use the window envelope source lock. It fixes the only admissible source for W as the transported StreamName and RegSeqRat rows carried by T.

Next apply the root envelope readback totality and downstream route readiness theorems. Together they place every Real-facing and completion-facing consumer on the displayed finite route from the source rows through W and then to E.

The final step is the completion seal lock. It makes refusal of all off-route host data part of the same route, so totality is not supplied by quotient streams, host equality, selected limits, choice, or ambient completeness. It is supplied by the accepted rows of T. □

Theorem 437.63 (Transported stationary window-seal route provenance nonescape)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The route provenance carried by P and named by N is nonescaping: any* $\mathsf{Real}^{\uparrow}$ *or completion consumer that preserves the* BHist*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of T receives exactly the finite route through S, G, W, E, H, C, P, N. It receives no external real-completion source and no route provenance not named by the accepted packet.*

Proof. Use the public export certificate to collect the packet rows over the displayed BEDC sources. The package row P and naming row N are part of the same finite carrier, so provenance is read from inside T rather than from a host object.

Now apply the Real completion route totality theorem. It identifies the complete consumer route through S, G, W, E, H, C, P, N, with the envelope and Real rows reached only after the source rows and transports have been fixed.

Finally apply ledger nonescape and route exhaustion. The accepted packet has no additional coordinate for a second route provenance row, an external completion carrier, or an unlisted real source. Thus provenance is confined to the displayed rows. □

Theorem 437.64 (Transported stationary window-seal root L10 source cover)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every accepted root L10 read exposes its source cover through the displayed* $\mathsf{Rat}^{\uparrow}$ *row R, constant* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal G,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row W,* $\mathsf{Real}^{\uparrow}$ *row E, and the structural* BHist/hsame/Cont/Pkg/NameCert *rows H, C, P, N. Thus the completion-facing source surface of T is exactly*

$$R, S, G, W, E, H, C, P, N,$$

with no detached quotient stream, host equality row, selected limit, or ambient real-completeness datum.

Proof. Start with Definition 437.10. It assigns the displayed $\mathsf{RegSeqRat}^{\uparrow}$ seal G to the same constant $\mathsf{StreamName}^{\uparrow}$ window S and finite envelope W carried by the accepted packet.

Next apply Definition 437.4. The $\mathsf{Real}^{\uparrow}$ row E is read only through that finite RegSeqRat/envelope route, with the transported support rows H, C, P, N already present.

Finally use Definition 437.38. Since the exposed NameCert route is exhausted by the transported source triad, tail schedule, finite envelope, and Real seal, every accepted L10 source cover is the displayed row family and no extra completion or host-real coordinate remains. □

Theorem 437.65 (Transported stationary window-seal root L10 envelope stability)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *and* $T' = (R', S', G', W', E', H', C', P', N')$ *be accepted* TransportedStationaryWindowSeal *packets whose displayed rows are related componentwise by* hsame *and whose seal, envelope, and Real rows are connected by the displayed* Cont *routes. Then* T *and* T' *preserve the same finite envelope-to-*$\mathsf{Real}^{\uparrow}$ *seal route:*

$$\mathsf{StreamName}^{\uparrow}(S) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(G) \longrightarrow \mathsf{FiniteWindowEnvelope}^{\uparrow}(W) \longrightarrow \mathsf{Real}^{\uparrow}(E)$$

and likewise for the primed packet under the same componentwise transport. In particular, envelope stability is determined at the finite L10 source cover, not by a selected host limit or an external real-completeness witness.

Proof. First read the classifier obligation through Definition 437.13. Componentwise hsame keeps the accepted rows of T and T' on the same BEDC classifier surface.

Next use Definition 437.33. The schedule row places the finite envelope after the StreamName and RegSeqRat source rows and before the Real-facing seal row.

The shared budget is then fixed by Definition 437.34. No second window or tail coordinate can be inserted between the transported source and the finite envelope.

Finally apply Definition 437.36. The displayed Cont routes therefore preserve the same envelope-to-Real seal route for both packets, and stability is witnessed inside the finite source cover itself. □

Theorem 437.66 (Transported stationary window-seal root L10 downstream readiness)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every downstream completion-facing consumer of* T *receives exactly the finite root L10 source cover of Definition 437.64 together with the envelope-stability rows of Definition 437.65. The handoff supplies the rows*

$$R, S, G, W, E, H, C, P, N$$

and no quotient stream, host equality, selected limit, choice principle, or real-completeness datum.

Proof. Begin with Definition 437.12. It states the downstream package using the finite transported source, envelope, Real seal, and structural support rows of the accepted packet.

Then apply the source-cover theorem above. It identifies the exact L10 rows that a downstream consumer can read from T.

Use the envelope-stability theorem above to keep those rows stable under componentwise hsame and displayed Cont transport. The finite envelope-to-Real route is therefore part of the handed-off package rather than an external completion datum.

Finally invoke Definition 437.6. The ledger excludes off-packet quotient streams, host equalities, selected limits, choice principles, and ambient real-completeness witnesses, so the downstream readiness surface is exactly the finite source cover plus the stable envelope route. □

Theorem 437.67 (Transported stationary window-seal RegSeqRat Real lock)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* G *and the* $\mathsf{Real}^{\uparrow}$ *row* E *are locked to the same* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *packet* W*: every accepted Real-facing read of* E *factors through the finite* BHist *carrier of* T*, the componentwise* hsame *rows, the displayed* Cont *transports, the* Pkg *provenance row* P*, and the local* NameCert *row* N *that already name* G *and* W*. Thus the RegSeqRat and Real rows are not two host streams; they are two reads of one transported finite-window seal route.*

Proof. Start with Definition 437.37. It identifies every $\mathsf{Real}^{\uparrow}$-facing read with the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad R, S, G, together with the support rows H, C, P, N.

Next use Definition 437.33. The finite envelope row W is the only position at which the transported regular-rational seal can be read before the Real row E, so the RegSeqRat and Real rows share the same finite-window packet.

Finally read the sibling certificates chapter 126, chapter 301, chapter 365, and chapter 38 through the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows of T. The accepted packet has no coordinate for a second regular seal or a detached host real source, giving the stated lock. □

Theorem 437.68 (Transported stationary window-seal tail budget coverage)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet. Every tail-budget read, tail-schedule read, or completion-facing schedule read exposed by T factors through the same finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ seal row, and $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ packet:*

$$R, S, G, H, C, P, N \longrightarrow W \longrightarrow E.$$

No accepted tail read introduces a selected host limit, quotient-stream equality, hidden tail oracle, independent completion object, or second real-completeness source.

Proof. First apply Definition 437.33. It places every tail-schedule row after the transported source triad and before the finite envelope and Real seal reads.

Then apply Definition 437.38. The exposed $\mathsf{NameCert}$ route is exhausted by the transported source triad, the schedule packet, the finite envelope, and the $\mathsf{Real}^{\uparrow}$ seal row; no unused schedule coordinate remains.

The sibling chapters chapter 126, chapter 301, and chapter 365 supply the finite window, regular-rational seal, and finite envelope surfaces consumed by the route. Since the accepted packet exposes only those BEDC rows and their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, selected host limits and quotient-stream equalities cannot enter as tail budget data. □

Theorem 437.69 (Transported stationary window-seal L10 exit readiness)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet. The Phase Real and Completion handoff consumes $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ only through the displayed finite seal route of T:*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{FiniteWindowEnvelope}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff is exit-ready for the open Phase Real and Completion surface because the RegSeqRat/Real lock and tail-budget coverage determine the available rows. It makes no claim that $\mathsf{Real}^{\uparrow}$ is complete, no claim that quotient streams have been identified, and no claim of a selected host limit.

Proof. Begin with Definition 437.48. It lists the only L10 route available to Real and Completion consumers: the finite passage through the StreamName, RegSeqRat, DyadicRatCore, FiniteWindowEnvelope, and Real sibling surfaces.

Use Definition 437.67 to bind the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads to the same finite envelope packet W. Use Definition 437.68 to keep every tail-budget and schedule-facing read on that same finite route.

Finally compare with chapter 126, chapter 301, chapter 365, and chapter 38. These sibling surfaces provide the named rows consumed by the transported seal, while the non-escape clauses exclude quotient-stream equality, selected host limits, hidden tail data, and ambient real-completeness witnesses. The packet is therefore ready to hand off to Phase Real and Completion exactly at the finite seal boundary stated above. □

Theorem 437.70 (Transported stationary window-seal tail-budget carrier)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The finite tail/window budget carrier read by T is the single route*

$$\mathsf{StreamName}^{\uparrow}(S), \quad \mathsf{RegSeqRat}^{\uparrow}(G), \quad \mathsf{FiniteWindowEnvelope}^{\uparrow}(W), \quad \mathsf{Real}^{\uparrow}(E)$$

carried by the displayed BHist *row H, componentwise* hsame *transport,* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}$ *row N. Any tail-cofinality or completion-facing consumer of T reads that carrier through the same finite window budget and not through a selected host tail, quotient stream, detached Real row, or off-packet completeness datum.*

Proof. Begin with Definition 437.26. It fixes the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source triad under the displayed support rows of T.

Next apply Definition 437.28. The tail schedule is attached to that same finite StreamName and RegSeqRat surface, so every tail-facing read uses the already displayed window budget.

Finally use Definition 437.29. It places the finite envelope W and the Real-facing seal row E downstream of the same transported source. The carrier is therefore exactly the displayed finite tail/window route, with no selected host tail, quotient stream, detached Real row, or off-packet completeness datum. □

Theorem 437.71 (Transported stationary window-seal StreamName-RegSeqRat-Real route)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The public root route exposed to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers has the fixed order*

$$\mathsf{StreamName}^{\uparrow}(S) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(G) \longrightarrow \mathsf{FiniteWindowEnvelope}^{\uparrow}(W) \longrightarrow \mathsf{Real}^{\uparrow}(E),$$

with the $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ *rows H, C, P, N transported along the route. No accepted consumer can read E before the finite envelope or separate G from the stationary StreamName window.*

Proof. First apply Definition 437.26. The StreamName and RegSeqRat rows are already one transported finite source surface, so a consumer cannot choose an independent regular sequence row.

Then use Definition 437.29. It forces every root read to pass through the finite envelope before the $\mathsf{Real}^{\uparrow}$ seal row is available.

Finally invoke Definition 437.38. The displayed $\mathsf{NameCert}$ surface is exhausted by the transported source triad, tail schedule, finite envelope, and Real seal. Hence the only public route order is the StreamName-RegSeqRat-envelope-Real route stated above. □

Theorem 437.72 (Transported stationary window-seal root-readiness package)**.** *Let $T = (R, S, G, W, E, H, C, P, N$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The packet is root-ready for the current Real and Completion dependency surface exactly through the tail-budget carrier of Definition 437.70 and the StreamName-RegSeqRat-Real route of Definition 437.71. Thus every public root consumer sees only the finite rows S, G, W, E and the displayed* $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ *support rows H, C, P, N.*

Proof. The carrier theorem Definition 437.70 supplies the finite tail/window budget used by tail-cofinality and completion-facing consumers. It fixes the available carrier rows as S, G, W, E with the support rows H, C, P, N.

The route theorem Definition 437.71 then orders those rows for public StreamName, RegSeqRat, and Real reads. Together with Definition 437.38, there is no remaining root coordinate outside the displayed carrier and route.

Consequently the accepted packet supplies the root-readiness package needed by the current Real and Completion dependency surface. It does not assert full real completeness, quotient-stream identification, host real equality, a selected host limit, a metric-completeness principle, or a bridge row. □

Theorem 437.73 (Transported stationary window-seal tail cofinality totality)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every accepted read of the tail schedule attached to T passes first through the* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet on the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S and* $\mathsf{RegSeqRat}^{\uparrow}$ *row G, together with the support rows H, C, P, N, before it reaches any finite envelope row W or* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently the tail-facing obligation surface of T is the finite route

$$R, S, G, H, C, P, N \longrightarrow \mathsf{TailCofinalitySchedule}^{\uparrow} \longrightarrow W \longrightarrow E,$$

with no earlier envelope, detached tail oracle, quotient stream, host equality, selected limit, or ambient completeness coordinate.

Proof. Start with Definition 437.10. It assigns the $\mathsf{RegSeqRat}^{\uparrow}$ seal G to the constant $\mathsf{StreamName}^{\uparrow}$ window S already carried by the accepted packet.

Next use Definition 437.33. The tail schedule row is read before the finite envelope and before the Real-facing seal, so a tail-facing consumer has no route that reaches W or E first.

Finally apply Definition 437.37 and Definition 437.38. These identify the Real source and the public NameCert surface with the displayed rows only, hence the tail schedule read is total over the stated finite route. □

Theorem 437.74 (Transported stationary window-seal dyadic radius readback)**.** *Let $T = (R, S, G, W, E, H, C, P, N$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The dyadic-radius evidence consumed by T is exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback attached to the constant* $\mathsf{StreamName}^{\uparrow}$ *window S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal G. It is not a separate radius oracle and it does not add a coordinate beyond the displayed* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BHist/hsame/Cont/Pkg}$, *and* $\mathsf{NameCert}$ *rows.*

Proof. First apply Definition 437.10. The window and regular-sequence rows are owned by the same transported source triad, so the radius readback is tied to S and G rather than to an independent oracle.

Then use Definition 437.33. The dyadic evidence is consumed on the same route that places the tail schedule before the finite envelope and Real seal.

The non-escape side is supplied by Definition 437.11 and Definition 437.38. Since the Real seal and NameCert route expose only the displayed packet rows, a second dyadic-radius source has no public coordinate on the accepted surface. □

Theorem 437.75 (Transported stationary window-seal Real handoff non-escape)**.** *Let $T = (R, S, G, W, E, H, C, P,$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or Completion handoff through T factors through the displayed route*

$$R, S, G \longrightarrow \mathsf{TailCofinalitySchedule}^{\uparrow} \longrightarrow W \longrightarrow E \longrightarrow H, C, P, N.$$

The handoff exports no quotient stream, host equality, selected limit, choice coordinate, detached radius oracle, or real-completeness coordinate.

Proof. Begin with Definition 437.37. It states that the Real-facing row E is sourced only through the transported $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ triad together with the displayed support rows.

Next apply Definition 437.33. It inserts the tail cofinality schedule and finite envelope in the only order compatible with the transported triad and the Real-facing seal.

Finally use Definition 437.11 and Definition 437.38. These exclude quotient streams, host equality, selected limits, choice, detached radius data, and real-completeness witnesses from the exported route, so the Real or Completion handoff remains inside $R, S, G, W, E, H, C, P, N$. □

Definition 437.76 (Transported stationary window-seal source-triad completion packet)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the source-triad completion packet is the finite BEDC route consisting of:*

1. *the transported source triad* R, S, G*, where* R *is the transported* $\mathsf{Rat}^{\uparrow}$ *source row,* S *is the constant* $\mathsf{StreamName}^{\uparrow}$ *window row, and* G *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row owned by that window;*
2. *the finite envelope and seal rows* W, E*, where* W *is the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row read after the triad and* E *is the* $\mathsf{Real}^{\uparrow}$*-facing seal row read after* W*;*
3. *the support rows* H, C, P, N*, carrying the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data for the same route;*
4. *the completion-facing consumer route*

$$R, S, G, H, C, P, N \longrightarrow W \longrightarrow E \longrightarrow \mathsf{Completion}^{\uparrow}.$$

The packet has no coordinate for quotient streams, host equality, selected host limits, choice, ambient real completeness, or hidden tail oracles; every such consumer must read the same displayed source triad before it can use the envelope row or the Real-facing seal.

Theorem 437.77 (Transported stationary window-seal source-window admission)**.** *Let* $T = (R, S, G, W, E, H, C, P$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet equipped with its source-triad completion packet. The packet admits a source-window triad only in the displayed form* R, S, G*:* R *is the transported* $\mathsf{Rat}^{\uparrow}$ *classifier row,* S *is the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and* G *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row owned by that window. No accepted read of* T *can replace this triad by a detached classifier, a moving stream window, or an independent seal source.*

Proof. Read the accepted carrier through the NameCert obligation surface of Definition 437.2. Its coordinates are exactly $R, S, G, W, E, H, C, P, N$, and the source side is exposed before the envelope and Real-facing rows.

The seed-window readback fixes the finite-window coordinate. By Definition 437.10, the $\mathsf{RegSeqRat}^{\uparrow}$ seal G is owned by the same constant $\mathsf{StreamName}^{\uparrow}$ row S, while R supplies the transported $\mathsf{Rat}^{\uparrow}$ classifier for that row.

Ledger non-escape then excludes alternate admissions. The package and naming rows P, N, together with the displayed BHist, hsame, and Cont support in H, C, leave no coordinate from which a detached classifier, moving window, or independent $\mathsf{RegSeqRat}^{\uparrow}$ source could be read. □

Theorem 437.78 (Transported stationary window-seal stationary transport coherence)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet equipped with its source-triad completion packet. The stationary handoff row* H *and route table* C *preserve the same* $R \to S \to G$ *route componentwise under* hsame *and* Cont*. Hence transport through* H, C *cannot create a detached* $\mathsf{RegSeqRat}^{\uparrow}$ *source or a second stream-name route for the seal.*

Proof. Readback determinacy first fixes the order of the source side. The route table can be consumed only after the accepted packet has exposed R, S, G, and Definition 437.55 keeps every later envelope read downstream from that source route.

The root window-ownership clause controls the seal coordinate. By Definition 437.10, the row G is a $\mathsf{RegSeqRat}^{\uparrow}$ seal on the same constant $\mathsf{StreamName}^{\uparrow}$ row S, so componentwise hsame may identify transported presentations of that row but cannot allocate a new source for G.

Finally, the Real-seal route forces coherence through completion-facing use. Apply Definition 437.4 and Definition 437.11: the Cont data in C carries the displayed source route into W and E, and no detached $\mathsf{RegSeqRat}^{\uparrow}$ row is available outside that route. □

Theorem 437.79 (Transported stationary window-seal completion handoff nonescape)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet equipped with its source-triad completion packet. Every completion-facing consumer of T must pass through the displayed rows $R, S, G, W, E, H, C, P, N$. In particular, such a consumer cannot import quotient streams, host equality, choice, selected limits, or ambient real-completeness data in place of the transported $\mathsf{Rat}^{\uparrow}$ classifier, constant $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ seal, finite envelope, Real-facing seal, and support rows.*

Proof. The downstream handoff package is rooted in the displayed packet. Use Definition 437.48 and Definition 437.57: they expose the completion-facing route only through the source triad, finite envelope, Real-facing seal, and the rows H, C, P, N.

The Real seal supplies no external escape. By Definition 437.11 and Definition 437.37, the row E is read from the finite $\mathsf{RegSeqRat}^{\uparrow}$ route through W, not from host equality, selected limits, choice, or real-completeness assumptions.

NameCert route exhaustion handles the remaining consumers. Apply Definition 437.38: the local name and package rows N, P, together with BHist, hsame, and Cont in H, C, enumerate the accepted handoff coordinates, so any completion-facing consumer that preserves T has no off-route coordinate for quotient streams or independent completion data. □

Theorem 437.80 (Transported stationary window-seal envelope RealUp totality)**.** *Let $T = (R, S, G, W, E, H, C, P$ be an accepted $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet equipped with its source-triad completion packet. Every $\mathsf{Real}^{\uparrow}$-facing read of E and every completion-facing read that preserves E factors through the $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ source triad and the $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row before the Real seal. More precisely, the read route is determined by*

$$R, S, G \longrightarrow W \longrightarrow E,$$

together with the displayed support rows H, C, P, N.

Proof. First use the carrier and NameCert obligation surface of Definition 437.2. It exposes the accepted packet only through $R, S, G, W, E, H, C, P, N$, with componentwise transport, package provenance, and local naming.

Next apply Definition 437.10. This fixes the $\mathsf{RegSeqRat}^{\uparrow}$ row G as a seal on the same finite $\mathsf{StreamName}^{\uparrow}$ row S, so the route cannot reach a finite envelope before the source triad has been read.

Finally use Definition 437.4, Definition 437.29, and the L10 handoff basis of Definition 437.48. These place the finite envelope row W between the source triad and the Real-facing row E, and they keep completion consumers on that same finite route. □

Theorem 437.81 (Transported stationary window-seal completion route nonescape)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet equipped with its source-triad completion packet. No completion-facing consumer preserving* T *can import quotient streams, host equality, selected limits, choice, ambient real completeness, or a hidden tail oracle. Such a consumer has access only to the transported source triad, the finite envelope, the Real seal row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows.*

Proof. Begin with readback determinacy, using Definition 437.55. It orders every accepted completion-facing read through the transported rows before the finite envelope and the Real seal can be consumed.

The Real side is controlled by Definition 437.11 and Definition 437.37. These rule out host equality, selected limits, choice, and ambient real completeness as sources for the seal row, because E is sourced by the displayed $\mathsf{RegSeqRat}^{\uparrow}$ route and the finite envelope.

It remains to exclude off-route completion data. Apply Definition 437.27, Definition 437.28, and Definition 437.50. The sibling L10 handoff files force any tail or completion consumer back through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, leaving no coordinate for a hidden tail oracle or independent completion source. □

Theorem 437.82 (Transported stationary window-seal source triad completion)**.** *Let* $T = (R, S, G, W, E, H, C, P,$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet equipped with the source-triad completion packet of Definition 437.76. The accepted route completes the* $\mathsf{Rat}^{\uparrow}$–$\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *source triad before any finite-envelope,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Completion}^{\uparrow}$ *consumer can read the packet:*

$$R, S, G, H, C, P, N \longrightarrow W \longrightarrow E.$$

The completed triad is carried by the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows only; it has no quotient-stream coordinate, host equality row, selected limit, choice datum, or ambient-completeness witness.*

Proof. Start with the carrier and NameCert obligation surface of Definition 437.2. It exposes the accepted packet exactly as the rows $R, S, G, W, E, H, C, P, N$, with local naming and package provenance attached to the same route.

Apply the root source lock Definition 437.15. It fixes R, S, G as one transported source route, so the stream window and regular-rational seal cannot be read as detached sources.

Finally combine readback determinacy with Definition 437.80. The finite envelope and Real-facing rows are reachable only after the completed triad has been read, and the non-escape clauses remove quotient streams, host equality, selected limits, choice, and ambient completeness from the route. □

Theorem 437.83 (Transported stationary window-seal tail-filter source triad handoff)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet and let* $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed on the same finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row. The finite-tail-filter and cofinal-tail surface receives exactly the completed source triad* R, S, G *together with its support rows* H, C, P, N*, then hands it through the schedule rows of* Q *before the route reaches* W *and* E*.*

Equivalently, every accepted tail-filter or cofinal-tail read preserving the packet is routed through

$$R, S, G, H, C, P, N \longrightarrow E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T \longrightarrow W \longrightarrow E.$$

No such handoff exports a quotient stream, host equality, selected limit, hidden tail oracle, choice principle, ambient-completeness datum, or independent completion carrier.

Proof. First use Definition 437.82. It supplies the completed $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source triad before any envelope or Real-facing read.

Next attach the tail surface. Apply Definition 437.73 and Definition 437.28; these place the accepted schedule packet Q on the same finite window and regular-rational seal row, downstream of R, S, G, H, C, P, N and upstream of W, E.

The completion boundary is then determined by Definition 437.58 and Definition 437.60. They identify the consumer-facing obligation basis with the same completed triad and schedule rows, so no separate tail oracle or completion carrier remains available.

Finally apply Definition 437.81. It excludes quotient streams, host equality, selected limits, choice, ambient completeness, and hidden tail data from the exported handoff. □

Theorem 437.84 (Transported stationary window-seal L10 exit source triad readiness)**.** *For accepted packets $T = (R, S, G, W, E, H, C, P, N)$ and $Q = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$ as above, the completed source-triad route exposes exactly the L10 Phase Real and Completion faces*

$$\mathsf{Rat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{TailCofinalitySchedule}^{\uparrow}, \mathsf{FiniteWindowEnvelope}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Completion}^{\uparrow}$$

The exposure is through the finite route R, S, G, H, C, P, N, the schedule rows of Q, and the envelope-seal rows W, E. It is therefore exit-ready for later L10 Real and Completion formalization, while making no claim of quotient-stream equality, host equality, selected limits, choice, or ambient real completeness.

Proof. Begin with the source-triad completion theorem above. It closes the $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source side of the route inside the accepted packet rows.

Use Definition 437.83 to pass that completed source side through the finite-tail-filter and cofinal-tail surface. The handoff supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{TailCofinalitySchedule}^{\uparrow}$ faces only as downstream reads of the same finite window and regular-rational seal.

Now apply Definition 437.48, Definition 437.67, and Definition 437.69. These identify the Phase Real and Completion exit surface with the finite envelope row W, the Real-facing seal row E, and the same displayed support rows.

The non-escape boundary is inherited from Definition 437.81 and Definition 437.75. Since the accepted route has no coordinate for quotient streams, host equality, selected limits, choice, or ambient completeness, the L10 exit surface consists exactly of the displayed Phase Real and Completion faces. □

Theorem 437.85 (Transported stationary window-seal checked obligation basis)**.** *For an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet $T = (R, S, G, W, E, H, C, P, N)$, the obligation basis is the triple already isolated by the checked NameCert rows:*

1. *the* $\mathsf{NameCert}_{\mathsf{TransportedStationaryWindowSeal}^{\uparrow}}$ *obligation surface of Definition 437.2;*
2. *the transported readback determinacy of Definition 437.3;*
3. *the* $\mathsf{Real}^{\uparrow}$ *factorization of Definition 437.4.*

Equivalently, the basis is the finite route

$$R, S, G, W, E, H, C, P, N$$

together with the componentwise BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows that the three checked theorems expose.*

Proof. The NameCert obligation theorem supplies the carrier, classifier, transported-window exactness, seal-exhaustion, and non-escape rows for T.

Readback determinacy then fixes the public $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ seal after the transported $\mathsf{Rat}^{\uparrow}$ classifier row is fixed through the displayed routes.

Finally, Real factorization forces every $\mathsf{Real}^{\uparrow}$ read to pass through the same finite envelope W and seal row G. These three rows are exactly the accepted obligation basis, and no additional carrier coordinate is introduced. □

Theorem 437.86 (Transported stationary window-seal formal target boundary)**.** *The formal target boundary for* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *obligation closure is exactly the checked Lean NameCert obligation theorem cited by the chapter closure row. It covers the finite carrier, componentwise classifier, transported-window exactness, seal exhaustion, and non-escape clauses for* $R, S, G, W, E, H, C, P, N$*. It does not include quotient streams, host equality, choice, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The target boundary is read from the NameCert obligation theorem rather than from an ambient real or completion object. Its inputs are the visible packet rows and their displayed structural support.

The readback and Real-factorization theorems remain companion checked rows, but they do not enlarge the target boundary beyond the NameCert obligation theorem. They only show that the existing boundary determines the stationary window readback and the Real-facing factorization.

The excluded objects are not carrier coordinates of T and are not supplied by the theorem boundary. Hence they cannot be part of the formal target recorded for obligation closure. □

Theorem 437.87 (Transported stationary window-seal obligation-closure handoff)**.** *The obligation-closure handoff from* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *to downstream Real and Completion consumers is carried by the already displayed rows for* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Every accepted downstream read must enter through that finite row set before it can use the checked obligation basis of Definition 437.85.*

Proof. Begin with the source route. The transported $\mathsf{Rat}^{\uparrow}$ row, constant $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ seal row give the source triad for the packet.

Next attach the finite envelope and Real-facing rows. The finite envelope supplies the observation boundary, and the Real row can only read it through the checked factorization.

The remaining rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provide the structural support and provenance for the same route. Therefore the downstream handoff is exactly the displayed finite row set, with no extra quotient, host, choice, selected-limit, or completeness input. □

Theorem 437.88 (Transported stationary window-seal constant-window exactness)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row* S *is read only from the transported* $\mathsf{Rat}^{\uparrow}$ *row* R *and then from the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G*. No completion-facing consumer may read* S *from a quotient stream, host equality, selected limit, choice principle, ambient real completeness object, hidden tail oracle, or second completion carrier.*

Proof. The carrier row of Definition 437.7 exposes exactly the displayed packet coordinates $R, S, G, W, E, H, C, P, N$

Apply the seed-window row Definition 437.8. It places the constant $\mathsf{StreamName}^{\uparrow}$ read at S after the transported $\mathsf{Rat}^{\uparrow}$ source row R and before the $\mathsf{RegSeqRat}^{\uparrow}$ seal row G.

Readback determinacy, Definition 437.3, then fixes the same order for every accepted consumer. Since no further carrier coordinate is present, the displayed R, S, G route is the exact constant-window route. □

Theorem 437.89 (Transported stationary window-seal Real seal factorization)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read reaches* E *only through the finite route*

$$R, S, G, W, H, C, P, N.$$

The row W *is the finite-window envelope through which the transported* $\mathsf{Rat}^{\uparrow}$*, constant* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows become available to the* $\mathsf{Real}^{\uparrow}$*-facing seal row* E*.*

Proof. Begin with envelope exactness. The row Definition 437.25 identifies W as the finite $\mathsf{StreamName}^{\uparrow}$ envelope carried by the transported source rows.

The completion-consumer boundary of Definition 437.27 then permits a completion-facing read of E only after the same source rows have passed through W.

Finally, the Real factorization row Definition 437.4 supplies the $\mathsf{Real}^{\uparrow}$ seal-facing endpoint. Thus every Real seal read factors through R, S, G, W, H, C, P, N, with W as the finite-window envelope. □

Theorem 437.90 (Transported stationary window-seal completion source nonescape)**.** *For every accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet* $T = (R, S, G, W, E, H, C, P, N)$*, no completion-facing consumer may import quotient-stream equality, host real equality, selected limits, choice, hidden tail oracle, real-completeness data, or a second completion carrier outside the displayed packet. Every admissible source for such a consumer must be read from the packet rows* $R, S, G, W, E, H, C, P, N$ *and their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support.*

Proof. The ledger nonescape row, Definition 437.6, excludes quotient-stream equality, host equality, selected limits, choice, and ambient completeness data from the packet ledger.

The shared-window budget is already exhausted by Definition 437.34. Hence there is no remaining finite-window coordinate from which a hidden tail oracle or second completion carrier could be read.

Root consumer nonescape, Definition 437.32, applies the same exclusion to downstream completion-facing consumers. Therefore every admissible source is one of the displayed packet rows together with its structural BEDC support. □

Theorem 437.91 (Transported stationary window-seal constant StreamName classifier coverage)**.** *Let* $T = (R, S, G, W, E, H, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The transported* $\mathsf{Rat}^{\uparrow}$ *classifier row* R *covers the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite seal* G *through the same displayed route*

$$R, S, G, H, C, P, N.$$

Every read of the constant window or its finite seal is therefore a read of this transported classifier route, with the support rows H, C, P, N *carrying the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data.*

Proof. Begin with the carrier and NameCert obligation surface of Definition 437.2. It exposes the accepted packet only through $R, S, G, W, E, H, C, P, N$, so the source side has no coordinate outside the transported rational classifier, constant stream name, and RegSeqRat seal rows.

Next apply readback determinacy, Definition 437.3. The constant $\mathsf{StreamName}^{\uparrow}$ row S is read after R and before the finite seal row G, and the same order is preserved by the displayed support rows.

Finally use route exhaustion, Definition 437.38. Since the exposed route surface is exhausted by the transported source triad and its support rows, the classifier coverage for the constant window and finite seal is exactly R, S, G, H, C, P, N. □

Theorem 437.92 (Transported stationary window-seal finite envelope handoff exhaustion)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing handoff through the finite envelope W and seal row E is exhausted by the same finite route*

$$R, S, G \longrightarrow W \longrightarrow E$$

together with the support rows H, C, P, N. Such a handoff cannot import quotient streams, host equality, choice, hidden tail oracles, selected limits, or ambient real completeness.

Proof. First apply Definition 437.4. It places every $\mathsf{Real}^{\uparrow}$ seal read behind the transported $\mathsf{Rat}^{\uparrow}$, constant $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and finite envelope rows.

Then use Definition 437.36. The schedule fixes the envelope W between the source triad and the Real-facing row E, so a handoff preserving W, E must preserve the same finite route.

It remains to exclude off-route sources. Apply Definition 437.38: the support rows H, C, P, N enumerate the accepted route surface, leaving no coordinate for quotient streams, host equality, choice, hidden tail data, selected limits, or ambient real completeness. □

Theorem 437.93 (Transported stationary window-seal Real completion exit readiness)**.** *Let $T = (R, S, G, W, E, H, C, P, N)$ be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet. The displayed route*

$$R, S, G, W, E, H, C, P, N$$

is the exact $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ *exit-readiness packet consumed by downstream L10 chapters: R supplies the transported rational classifier, S supplies the constant stream-name window, G supplies the RegSeqRat finite seal, W supplies the finite envelope, E supplies the Real-facing row, and H, C, P, N supply the BEDC support, package, and naming rows.*

Proof. Start with the constant-window coverage theorem above. It identifies the source side of the packet as the finite transported route R, S, G, H, C, P, N.

Next use the finite-envelope handoff theorem above. It extends the same route through W and E while excluding quotient-stream, host-equality, choice, hidden-tail, selected-limit, and ambient-completeness sources.

Finally combine Definition 437.3, Definition 437.4, and Definition 437.36. These rows fix the read order from the transported classifier through the finite seal and envelope to the Real-facing endpoint. Thus downstream L10 consumers receive exactly $R, S, G, W, E, H, C, P, N$ as their $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ exit-readiness packet. □

Definition 437.94 (Transported stationary window-seal scoped route)**.** *For an accepted* TransportedStationaryWind *packet*

$$T = (R, S, G, W, E, H, C, P, N),$$

the scoped route is the finite dependency packet that reads R as the transported $\mathsf{Rat}^{\uparrow}$ *classifier, S as the induced* $\mathsf{StreamName}^{\uparrow}$ *finite window, G as the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal on that window, W as the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *observation packet with its* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, E as the* $\mathsf{Real}^{\uparrow}$*-facing completion boundary, H as the transported handoff row, C as the displayed* hsame *and* Cont *route table, P as the* Pkg *provenance row, and N as the local* NameCert *row.*

Its dependency basis is exactly the sibling surface

$\mathsf{Rat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{FiniteWindowEnvelope}^{\uparrow}, \mathsf{TailCofinalitySchedule}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Completion}^{\uparrow}$

The route supplies a finite $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *handoff boundary; it does not add quotient streams, host equality, selected limits, choice, or full real-completeness data.*

Theorem 437.95 (Transported stationary window-seal scoped dependency basis)**.** *Let* $T = (R, S, G, W, E, H, C, P,$ *be an accepted* $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ *packet equipped with the scoped route of Definition 437.94. Every public read of* T *factors through the displayed rows* $R, S, G, W, E, H, C, P, N$ *and through the named dependencies* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *No public read of the packet has a further dependency coordinate.*

Proof. Start with the carrier and naming surface. The carrier definition and Definition 437.2 expose only the rows $R, S, G, W, E, H, C, P, N$, with the componentwise classifier, displayed BHist support, hsame comparisons, Cont transports, Pkg provenance, and local $\mathsf{NameCert}$ row.

Next apply the source-triad completion packet and the L10 handoff basis. Theorems 437.82 and 437.48 place the public route through $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ in that finite order.

Finally use the Real and completion boundary theorems. Theorems 437.11, 437.27, and 437.38 rule out quotient streams, host equality, selected limits, choice, ambient real completeness, and hidden completion carriers. Thus every public read factors through exactly the scoped dependency basis displayed above. □

Theorem 437.96 (Transported stationary window-seal source-triad scope exhaustion)**.** *In the scoped route of Definition 437.94, the source triad* R, S, G *cannot be detached from the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{R}$ *route. The row* R *determines the constant stream window* S*, the row* S *owns the regular-rational seal* G*, and every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read of* W *or* E *must consume that same triad before it reaches the finite envelope or the seal row.*

Proof. First read the transported source side. The readback determinacy and root window-ownership theorems identify R as the transported $\mathsf{Rat}^{\uparrow}$ row, S as the induced constant $\mathsf{StreamName}^{\uparrow}$ window, and G as the $\mathsf{RegSeqRat}^{\uparrow}$ seal owned by that window.

Next attach the completion packet. The source-triad completion packet and the tail-filter handoff theorem keep R, S, G upstream of the tail schedule, dyadic ledger, finite envelope, and Real-facing seal.

It remains to exclude detached sources. Apply Definition 437.37, Definition 437.45, and Definition 437.38. These theorems identify every Real and Completion consumer with the same source triad and leave no public coordinate for a moving stream window, independent regular-rational seal, or off-route real source. □

Theorem 437.97 (Transported stationary window-seal real-completion boundary)**.** *The scoped route of Definition 437.94 supplies only the finite* $\mathsf{Real}^{\uparrow}$*-facing completion boundary determined by* $R, S, G, W, E, H, C, P, N$*. It does not supply quotient streams, host equality, selected limits, choice, full real completeness, or an independent* $\mathsf{Completion}^{\uparrow}$ *carrier beyond the finite handoff named by the packet.*

Proof. Begin with the Real-facing seal. By Definition 437.4 and Definition 437.11, the row E is read only through the finite envelope W, the regular-rational seal G, and the displayed transport and provenance rows.

Next use the completion handoff controls. Theorems 437.54, 437.75, and 437.50 force every completion-facing consumer back to the same finite source triad, tail schedule, envelope, and Real seal.

Finally apply route exhaustion. The NameCert route exhaustion theorem enumerates the visible packet rows and support rows, so there is no remaining coordinate for quotient streams, host equality, selected host limits, choice, full real completeness, or an independent completion carrier. The route therefore supplies exactly the finite Real-facing completion boundary and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{TransportedStationaryWindowSeal}\uparrow}$.

Constructive story: A $\mathsf{TransportedStationaryWindowSeal}^{\uparrow}$ packet is generated from a transported RatUp classifier, constant StreamNameUp finite window, RegSeqRatUp seal row, FiniteWindowEnvelopeUp observation packet, RealUp seal-facing row, displayed transports, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TransportedStationaryWindowSeal}\uparrow}, \mathsf{scopedClosed})$

Scope closed: The scoped route is closed over RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, FiniteWindowEnvelopeUp, TailCofinalityScheduleUp, RealUp, CompletionUp, BHist, hsame, Cont, Pkg, and NameCert, read through the finite packet rows $R, S, G, W, E, H, C, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TransportedStationaryWindowSeal}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TransportedStationaryWindowSealUp.TransportedStationaryWindowSealPacket\_namecert\_obli`

Bridge status: none

Origin: human

Not claimed: This chapter does not close full real completeness, quotient-stream equality, host real equality, selected host limits, or choice principles.

Upgrade path: Public closure requires a public export theorem for this scoped transported window-seal route.

Chapter 438

A Concrete Naming Certificate for $\mathsf{MonotoneCauchy}^{\uparrow}$

The $\mathsf{MonotoneCauchy}^{\uparrow}$ interface records the finite surface through which a located monotone Cauchy packet is read by interval and real-seal consumers. In the Bishop–Bridges constructive-analysis reading, monotone convergence is exposed through displayed Cauchy windows, located upper bounds, and interval enclosures rather than by trichotomy, quotient reals, open-cover compactness, or a choice of a limiting representative.

The located enclosure row I consumes the interval surface of chapter 394: its lower endpoint, upper endpoint, and location proof are the interval rows read by the monotone packet. The finite tail-window and shared-threshold discipline follows the same RegularCauchy tail route recorded in chapter 560, so the monotone schedule and modulus rows are consumed as displayed tail data rather than as a hidden limiting representative.

Definition 438.1 (Monotone Cauchy carrier)**.** *A* $\mathsf{MonotoneCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

with the following visible coordinates:

1. *F is a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rational-name row carrying the inspected monotone Cauchy observations;*
2. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that selects the finite windows read from F;*
3. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that routes each requested dyadic tolerance to a finite tail window;*
4. *λ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of tolerance rows, monotone comparison rows, and located upper-bound readbacks;*
5. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *enclosure row that records the lower endpoint, upper endpoint, and location proof consumed by interval handoff readers;*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached from the same regular sequence, schedule, modulus, dyadic ledger, and located interval enclosure;*
7. *H records componentwise* hsame *transports for the regular sequence, schedule, modulus, dyadic ledger, located interval, and real-seal rows;*
8. *C records the displayed* Cont *routes by which consumers inspect those rows;*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *source certificates;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ *row.*

The monotonicity ledger is the displayed dyadic comparison row

$$\lambda_{\mathrm{mono}}(i,j) : i \le j \Rightarrow F_i \le F_j,$$
$$\lambda_{\mathrm{cau}}(\varepsilon) : \sigma(\mu(\varepsilon)) \text{ is a tail window below } \varepsilon.$$

The located upper-bound and interval ledger is

$$\lambda_{\mathrm{loc}}(\varepsilon) : F_{\sigma(\mu(\varepsilon))} \le I^{+} \text{ and } I^{-} \le F_{\sigma(\mu(\varepsilon))},$$
$$I(\varepsilon) = [I^{-}(\varepsilon), I^{+}(\varepsilon)] \text{ is the located interval row read by } Q.$$

The carrier contains no quotient-name equality, selected representative, host real equality, ambient completeness theorem, trichotomy row, or choice operator.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier carrier admission window`
Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier interval real seal route certificate`

Theorem 438.2 (Monotone Cauchy NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier admission for the displayed packet of Definition 438.1;*
2. *classifier by common finite windows selected from the schedule σ and compared through the same regular-sequence, modulus, tolerance, interval, and seal rows;*
3. *monotone-row transport, stating that the displayed monotonicity ledger is preserved by the componentwise* hsame *rows in H and by the consumer routes in C;*
4. *located interval handoff, stating that every accepted interval read reaches the displayed upper-bound and enclosure ledger before any real-seal read;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, stating that the seal row Q consumes only $F, \sigma, \mu, \lambda, I, H, C, P, N$ and no hidden completion coordinate.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier namecert obligation surface`

Proof. Carrier admission is projection of the finite packet in Definition 438.1. Each coordinate is a displayed history row, transport row, continuation route, package row, or local naming row.

The classifier is by common finite windows. Two accepted packets are compared only after their schedules select the same requested tail window and their dyadic tolerance, monotone comparison, located interval, and real-seal rows have been transported componentwise.

Monotone-row transport uses the hsame rows in H to move the displayed comparisons and the Cont routes in C to replay the same reads. The located interval handoff is then the projection of the ledger λ_{loc} and the enclosure row I.

Finally, the real-seal boundary is the row Q. Since the carrier has no quotient-name equality, selected representative, host real equality, trichotomy row, ambient completeness row, or choice operator, no such coordinate can be read by a downstream consumer. □

Theorem 438.3 (Monotone Cauchy located interval handoff)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{LocatedInterval}^{\uparrow}$ *consumer read of* M *factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row* μ*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and upper-bound ledger* λ*, and the located interval row* I*, transported only by* H *and* C *with provenance* P, N*.*

The handoff asserts a finite located enclosure for the monotone Cauchy packet; it does not assert a selected limiting representative, host real equality, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier located interval handoff`

Proof. Project the carrier. The interval coordinate is I, and every route to it first exposes the scheduled regular-rational observations F, the selected window σ, the modulus row μ, and the dyadic ledger λ.

The located ledger records the lower and upper enclosure rows used by the interval consumer. Transporting an accepted packet moves those rows componentwise through H and replays only the displayed routes in C.

Thus the interval handoff is exhausted by $F, \sigma, \mu, \lambda, I, H, C, P, N$. None of the excluded analytic objects is present as a carrier coordinate, so no interval consumer can read it from this packet. □

Theorem 438.4 (Monotone Cauchy Real seal boundary)**.** *For an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$ *seal read factors through the same displayed regular sequence* F*, schedule* σ*, Cauchy modulus* μ*, dyadic monotonicity and located upper-bound ledger* λ*, located interval row* I*, transport row* H*, continuation row* C*, package row* P*, and local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ *row* N*.*

Consequently the real seal receives a finite monotone Cauchy and located-interval surface, not quotient-name equality, a selected representative, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier real seal boundary`

Proof. Use the NameCert obligation surface to force each consumer route onto a common finite window and to preserve the monotone row under transport.

Apply Definition 438.3. The located interval row is reached only after the regular sequence, schedule, modulus, dyadic ledger, transports, continuations, package, and local certificate have already been read.

The seal coordinate Q consumes that same finite surface. Since no excluded completion or host-real coordinate occurs in Definition 438.1, the real-seal boundary cannot escape the displayed packet. □

Theorem 438.5 (Monotone Cauchy ledger exhaustion)**.** *Every public read of an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *carrier is exhausted by the displayed monotonicity ledger, Cauchy tail ledger, located upper-bound ledger, located interval row, real-seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row of Definition 438.1.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier ledger exhaustion`

Proof. Inspect the carrier coordinates. The regular rational observations, stream schedule, Cauchy modulus, dyadic comparison rows, located interval row, and real-seal row are all explicit fields of the same finite BHist packet.

The transport and continuation fields account for all permitted movement of those rows. The package and local naming rows account for provenance, but they do not add analytic data.

Therefore every public monotone Cauchy, located interval, or real-seal read has already appeared in the displayed ledger. There is no remaining coordinate for quotient equality, a selected representative, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice. □

Theorem 438.6 (Monotone Cauchy common-window classifier)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted MonotoneCauchy$^{\uparrow}$ *carrier. Any classifier comparison of M with another accepted carrier is made after both packets expose a common finite window selected through the* StreamName$^{\uparrow}$ *schedule σ and the* CauchyModulus$^{\uparrow}$ *row μ. On that window the classifier compares only the* RegSeqRat$^{\uparrow}$ *observations, the* DyadicRatCore$^{\uparrow}$ *tolerance and monotonicity ledger, the* LocatedInterval$^{\uparrow}$ *enclosure, the* Real$^{\uparrow}$ *seal row, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project the classifier row from Definition 438.2. It requires common finite windows before accepted packets are compared.

The window is selected by σ and μ. Once the window is fixed, the regular rational observations in F and the dyadic ledger λ supply the tolerance, monotonicity, and located upper-bound rows available for comparison.

The interval row I and seal row Q are downstream of that same window. The transport and continuation rows H and C replay only the displayed reads, while P and N name the same local package, so no quotient or host-real comparison is introduced. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier common window classifier`

Theorem 438.7 (Monotone Cauchy modulus tail stability)**.** *For every accepted carrier $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$, transporting M by the componentwise* hsame *rows in H preserves the finite tail comparison selected by μ. The monotone row and Cauchy tail row in λ therefore remain attached to the same* RegSeqRat$^{\uparrow}$ *observations and the same* StreamName$^{\uparrow}$ *window route before any located interval or real-seal consumer reads them.*

Proof. Read the tail ledger in Definition 438.1. The row $\lambda_{\mathrm{cau}}(\varepsilon)$ ties the requested tolerance to the scheduled window $\sigma(\mu(\varepsilon))$, and the monotone row λ_{mono} compares observations on that displayed window.

Transport the packet through H. Componentwise transport moves F, σ, μ, and λ together, and the continuation routes in C replay the same window read before exposing interval or seal data.

Apply the ledger exhaustion theorem Definition 438.5. Since every public read is already exhausted by the displayed ledger, the transported tail comparison cannot detach from the modulus route or acquire an additional completion premise. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier modulus tail stability`

Theorem 438.8 (Monotone Cauchy Real seal factorization obligation)**.** *Let $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal read of Q factors through the common finite-window classifier, the modulus tail route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *located ledger, and the* $\mathsf{LocatedInterval}^{\uparrow}$ *row I before any completion-facing read is consumed. The factorization exports no quotient real, selected representative, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice operator.*

Proof. Use Definition 438.6 to place the comparison on a common finite window. The regular sequence, stream schedule, modulus row, and dyadic ledger are all visible before the seal row is compared.

Next apply Definition 438.7. The Cauchy tail and monotone comparison rows remain attached to the same modulus-selected window under transport, so the downstream located interval row consumes the same finite ledger.

Finally apply Definition 438.3. The real-seal boundary of Definition 438.4 then reads Q only after I and its upstream ledger have been displayed, and the carrier contains none of the excluded analytic objects. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier real seal factorization obligation`

Theorem 438.9 (Monotone Cauchy located trap certificate)**.** *Let $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *carrier. The monotone ledger row and Cauchy modulus row determine a finite* $\mathsf{LocatedInterval}^{\uparrow}$ *trap row for the tail window selected by σ and μ. The trap certificate uses only F, σ, μ, λ, I, H, C, P, and N, and it records the lower observation, upper guard, tolerance, and transport route before any* $\mathsf{Real}^{\uparrow}$ *seal read is available.*

No quotient real, selected representative, host real order, trichotomy, open-cover compactness, ambient completeness, or choice operator is part of this certificate.

Proof. Project the carrier through Definition 438.1. The rows F, σ, μ, and λ display the regular rational observations, the selected finite window, the modulus tail, and the monotone located upper-bound ledger.

Apply Definition 438.6. The trap is built on the same common finite window used by classifier comparisons, so every row needed for the located interval is already visible before interval handoff.

Next use Definition 438.7. Transport through H keeps the Cauchy tail and monotone comparison rows attached to the same scheduled observations, and C replays the window read without adding a completion premise.

Finally apply Definition 438.3 and Definition 438.5. The located row I receives exactly the displayed trap data, while P and N name the same local package. The carrier contains no coordinate from which the excluded analytic objects could enter. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier located trap certificate`

Theorem 438.10 (Monotone Cauchy tail-window Real seal certificate)**.** *For every accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$, each* $\mathsf{Real}^{\uparrow}$ *seal read of Q factors through the finite located trap certificate and the same modulus-selected tail window before reaching Q. The seal-facing path is therefore*

$$(F, \sigma, \mu, \lambda) \quad \textit{then} \quad I \quad \textit{then} \quad Q,$$

with componentwise transport supplied by H and displayed continuation supplied by C.

The path does not branch through quotient equality, a selected real representative, host real order, open-cover compactness, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier tail window real seal certificate`

Proof. Start with Definition 438.9. It gives a finite located trap whose rows are already selected by the stream schedule, the Cauchy modulus, and the monotone dyadic ledger.

Apply Definition 438.8. Every seal-facing read must pass through the common finite-window classifier, the modulus tail route, the dyadic located ledger, and the located interval row before it consumes Q.

Use Definition 438.4 for the final handoff. The boundary reads only the displayed located interval and the seal row, and Definition 438.5 rules out any additional hidden coordinate for the excluded analytic objects. □

Theorem 438.11 (Monotone Cauchy located interval cell coherence)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Every located interval cell read in I is coherent with the same finite* $\mathsf{StreamName}^{\uparrow}$ *window selected by σ and μ, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *observations in F, and the same dyadic ledger λ before the* $\mathsf{Real}^{\uparrow}$ *seal row Q is read. The lower and upper endpoint rows of the located interval, the rational or dyadic cell row being inspected, and the seal-facing route therefore form one finite window:*

$$(F, \sigma, \mu, \lambda) \quad \textit{then} \quad I \quad \textit{then} \quad Q.$$

The coherence statement uses only the displayed BHist *surface, the componentwise* hsame *transports in H, the* Cont *routes in C, the* Pkg *provenance row P, and the local* NameCert *row N. It does not invoke a quotient real, selected representative, host real order, trichotomy, ambient completeness principle, or choice row.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier located interval cell coherence`

Proof. Project the packet through Definition 438.1. The only tail-window rows visible before the interval and seal stages are F, σ, μ, λ together with their displayed transport, continuation, provenance, and naming rows.

Apply Definition 438.9. It binds the located interval row I to the regular rational observations in F, the finite window selected by σ and μ, and the monotone dyadic ledger λ before any $\mathsf{Real}^{\uparrow}$ seal row is available.

Use Definition 394.35 for the endpoint-cell step. The lower endpoint row, upper endpoint row, rational or dyadic cell ledger, endpoint windows, regular readbacks, and seal-facing rows of the located interval remain a single finite route, so a cell read in I cannot detach from the displayed endpoint rows.

Finally apply Definition 438.10. The seal row Q consumes the same located trap and the same modulus-selected finite window, so the interval cell, endpoint rows, regular observations, dyadic ledger, and Real seal are read from one carrier surface. Since the carrier has no coordinate for quotient reals, selected representatives, host real order, trichotomy, ambient completeness, or choice, none of those objects can enter the coherence route. □

Theorem 438.12 (Monotone Cauchy CauchyCriterion seal handoff)**.** *Let $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Suppose the* $\mathsf{Real}^{\uparrow}$ *seal row Q is consumed by an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet through the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

$\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then every such consumer route factors through the finite Monotone Cauchy tail-window certificate and then through the CauchyCriterion finite criterion route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{CauchyModulus}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff remains on the same finite BHist *surface, with the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It cannot introduce quotient equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, or any separate completion oracle.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier cauchycriterion seal handoff`

Proof. Apply Definition 438.10. It forces each $\mathsf{Real}^{\uparrow}$ seal read of Q through the located trap, the modulus-selected tail window, and the displayed StreamName, dyadic, regular rational, transport, continuation, provenance, and naming rows of M.

Next use Definition 392.7. A CauchyCriterion consumer can read a regular rational source and the $\mathsf{Real}^{\uparrow}$ seal only through its displayed source window, modulus threshold, dyadic tolerance, tail ledger, and seal row. The shared-row hypothesis identifies those rows with the route already exposed by the Monotone Cauchy certificate.

Finally apply Definition 392.8. The StreamName presentation, dyadic tail coverage, regular shared threshold, and seal row remain one finite shared-tail route, so repacking through BHist, hsame, Cont, Pkg, and NameCert preserves the same seal surface. None of the participating carriers contains a coordinate for quotient equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, or an independent completion oracle. □

Theorem 438.13 (Monotone Cauchy located Real-completion handoff)**.** *Let* $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Every Real-and-Completion consumer of* M *that reads the* $\mathsf{Real}^{\uparrow}$ *seal* Q *factors through the finite located trap certificate, the CauchyCriterion route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{CauchyModulus}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

and the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows before the seal is read.*

The consumer boundary therefore exposes only the finite rows $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *together with the accepted CauchyCriterion handoff surface. It does not export quotient equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, or a separate completion oracle.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier located real completion handoff`

Proof. First apply Definition 438.9. It places the located read behind the common finite window, the monotone dyadic ledger, and the displayed located interval row.

Next use Definition 438.10. The $\mathsf{Real}^{\uparrow}$ seal Q can be consumed only after the located trap, modulus-selected tail window, and transport rows have been displayed.

Now apply Definition 438.12. The Real-and-Completion consumer is forced through the CauchyCriterion route from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$.

Finally use the ledger-exhaustion rows, including Definition 438.5. They repack the same finite route through BHist, hsame, Cont, Pkg, and NameCert, leaving no carrier coordinate for quotient equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, or an independent completion oracle. □

Theorem 438.14 (Monotone Cauchy public seal window determinacy)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N) \quad \textit{and} \quad M' = (F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$$

be accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packets. Suppose their* F, σ, μ, λ *tail-window rows are displayed on the same finite window, and their* I, Q, H, C, P, N *rows are transported componentwise by the shown* hsame *data. Then any public* $\mathsf{Real}^{\uparrow}$ *seal read of* Q *and any public* $\mathsf{Real}^{\uparrow}$ *seal read of* Q' *factor through the same finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{CauchyModulus}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before reaching the seal row.

Consequently the two public seal reads determine the same finite tail-window seal surface under the displayed hsame *transport. They cannot diverge through quotient equality, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier public seal window determinacy`

Proof. First apply Definition 438.6. The two accepted packets are compared only after the schedule and modulus rows expose the same finite tail window for the displayed F, σ, μ, λ data.

Next use Definition 438.7. The componentwise hsame transport keeps the Cauchy and monotone rows attached to that shared window, so neither packet can move the seal read to a different tail route.

The interval stage is supplied by Definition 438.9. The located trap consumes the transported dyadic ledger and produces the same $\mathsf{LocatedInterval}^{\uparrow}$ surface before any $\mathsf{Real}^{\uparrow}$ seal row is read.

Finally apply Definition 438.10 and Definition 438.5. The former forces each public seal read through the displayed tail-window route, and the latter says that every public read has already appeared in the finite ledger. Since no excluded quotient, representative, host-real, compactness, completeness, or choice coordinate occurs in the carrier, the two transported seal reads have the same finite seal surface. □

Theorem 438.15 (Monotone Cauchy window tail-cofinality handoff)**.** *Let* $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *seal consumer of* M *that also uses a* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *route must pass through the displayed monotone ledger, the Cauchy modulus row, the located trap certificate, the finite tail-window schedule, the dyadic threshold ledger, and the located interval row before the seal row* Q *is readable. The handoff supplies no host real order, quotient equality, selected representative, ambient completeness, open-cover compactness, or choice operator.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier window tail cofinality handoff`

Proof. Begin with Definition 438.9. The rows F, σ, μ, λ expose the monotone ledger and Cauchy modulus data, and the located trap certificate places the finite interval row I on that same selected tail window.

Next use Definition 438.10. It forces every seal-facing read of Q through the located trap and the modulus-selected tail window before the $\mathsf{Real}^{\uparrow}$ seal row can be consumed.

Finally apply Definition 681.3 to the sibling schedule route. Its finite route carries the tail window through the dyadic threshold ledger before any downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read, and the two carriers contain none of the excluded analytic or choice coordinates. □

Theorem 438.16 (Monotone Cauchy obligation package)**.** *The rows F, σ, μ, λ, I, Q, H, C, P, and N of an accepted* MonotoneCauchy$^{\uparrow}$ *carrier assemble into the five-row* NameCert *obligation package for located monotone Cauchy data: carrier admission, classifier by common finite windows, monotone-row transport through the modulus tail, located interval handoff, and* Real$^{\uparrow}$ *seal non-escape. Each row is read from the displayed packet and its local* Pkg *and* NameCert *data.*

The package claims only this displayed finite-window obligation surface. It does not close quotient reals, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier obligation package`

Proof. Carrier admission is the displayed carrier of Definition 438.1. It exposes the regular sequence, stream schedule, modulus, dyadic ledger, located interval, real seal, transport, continuation, package, and naming rows.

For classification, apply Definition 438.6. Accepted packets are compared only after the common finite window selected by σ and μ is displayed.

For transport, apply Definition 438.7. The monotone and Cauchy rows in λ remain attached to the same scheduled observations under the componentwise hsame rows in H.

For the downstream rows, apply Definition 438.9 and Definition 438.10. The located interval row consumes the finite trap, and every real-seal read remains inside that same tail window.

Finally use Definition 438.5. Since every public read has already appeared in the displayed ledger, the five obligation rows are exhausted by the packet data and do not import any excluded completion or choice principle. □

Theorem 438.17 (Monotone Cauchy scoped tail-window package)**.** *For an accepted* MonotoneCauchy$^{\uparrow}$ *carrier, the common finite window, the* CauchyModulus$^{\uparrow}$ *row μ, the* DyadicRatCore$^{\uparrow}$ *monotone ledger λ, the* LocatedInterval$^{\uparrow}$ *trap I, the* Real$^{\uparrow}$ *seal Q, and the displayed* hsame, Cont, Pkg, *and* NameCert *rows form a single scoped packet. Tail-window readers and* Real$^{\uparrow}$ *readers consume the packet only through*

$$F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

No such reader can pass through quotient reals, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier scoped tail window package`

Proof. First expose the common finite window by applying Definition 438.6. The stream schedule σ and modulus row μ determine the finite observations of F that any accepted comparison may read.

Next apply Definition 438.7. The CauchyModulus$^{\uparrow}$ tail and the DyadicRatCore$^{\uparrow}$ monotone ledger remain attached to that same finite window under the componentwise hsame rows in H.

The located part is supplied by Definition 438.9. It turns the monotone ledger and upper-bound rows into the displayed located interval trap I without introducing any ambient completeness datum.

For Real$^{\uparrow}$ readers, use Definition 438.10. The seal row Q consumes the common window, modulus tail, dyadic ledger, located trap, transport rows, continuation routes, package provenance, and local naming row.

Finally apply Definition 438.5. Every permitted reader has already appeared in the displayed rows $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the scoped packet has no hidden route to the excluded analytic or choice principles. □

Theorem 438.18 (Monotone Cauchy located scope package)**.** *For an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *carrier, the located interval trap is a scoped packet over* $F, \sigma, \mu, \lambda, I, H, C, P, N$*. Every located read first passes through the common finite tail window, the monotone dyadic ledger, and the located trap certificate before it reaches the displayed interval row.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier located scope package`

Proof. Start with Definition 438.6. It selects the common finite window through the schedule σ and modulus μ before any accepted packets are compared.

Apply Definition 438.9. The located trap is read from the dyadic monotonicity and upper-bound ledger λ over the same finite tail window.

Finally use Definition 438.3. The interval row I is reached only through the displayed regular sequence, schedule, modulus, ledger, transport, continuation, package, and naming rows. □

Theorem 438.19 (Monotone Cauchy tail-window scope lock)**.** *The tail-window classifier of an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet is locked to the finite* BHist *rows selected by* σ *and* μ*. Tail stability, located interval transport, and consumer-visible classifier reads are routed only through* H*,* C*,* P*, and* N*, with no quotient real, selected representative, host order, or choice operator.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier tail window scope lock`

Proof. Use Definition 438.6 to expose the common finite tail window. The classifier compares only the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ observations and the dyadic tolerance, monotonicity, interval, and seal rows visible on that window.

Apply Definition 438.7. The monotone and Cauchy rows remain attached to the scheduled observations under the componentwise hsame transport in H.

The continuation routes in C replay the same finite reads, and Definition 438.5 excludes every hidden analytic coordinate. Thus the tail-window classifier is locked to the displayed packet. □

Theorem 438.20 (Monotone Cauchy Real seal scope package)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet factors through the located scope package and the tail-window classifier. The seal uses only* $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$*, so RealUp-facing consumers receive a displayed regular rational sequence, a finite tail window, a located interval trap, and the local* $\mathsf{NameCert}$ *provenance.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier real seal scope package`

Proof. Apply Definition 438.8. It factors each seal-facing path through the common finite-window classifier, the modulus tail route, the dyadic located ledger, and the located interval row.

Use Definition 438.18 to bind the located trap and interval handoff to the displayed rows $F, \sigma, \mu, \lambda, I, H, C, P, N$.

Finally apply Definition 438.19 and Definition 438.4. The seal row Q consumes the same finite surface and has no route to quotient reals, selected representatives, host real equality, ambient completeness, or choice. □

Theorem 438.21 (Monotone Cauchy public tail-window certificate)**.** *The scoped tail-window classifier, located scope package, and* $\mathsf{Real}^{\uparrow}$ *seal scope package assemble a single public* $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ *export over rows* $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$*. The public certificate exposes the finite tail-window schedule, monotone and Cauchy modulus rows, located interval trap, real-seal row, componentwise transports, continuation routes, package provenance, and local naming row, and exports no quotient real, selected representative, host real equality, ambient completeness, or choice principle.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier public tail window certificate`

Proof. Start with Definition 438.6. It identifies the finite tail window through which the scheduled observations F, σ, μ are read.

Use Definition 438.18 and Definition 438.19. The located rows λ and I remain attached to the same tail window, while the lock prevents consumer reads from moving outside H, C, P, N.

Finally apply Definition 438.20 and Definition 438.5. The real-seal row Q is repacked through the displayed finite surface, and the ledger excludes every route to quotient reals, selected representatives, host real equality, ambient completeness, or choice. □

Theorem 438.22 (Monotone Cauchy bridge-facing certificate)**.** *Let $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. If M is consumed through a* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet, or if a* $\mathsf{Real}^{\uparrow}$ *seal consumer reads M through Q, then both consumers read the same finite tail-window surface: the regular rational row F, scheduled windows σ, modulus row μ, dyadic ledger λ, located trap I, seal row Q, transport rows H, continuation routes C, package provenance P, and local naming row N. Neither consumer may replace that surface by quotient real equality, selected representatives, ambient completeness, or choice-selected tails.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier bridge facing certificate`

Proof. Apply Definition 438.21. It already assembles the public $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ export over exactly $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and excludes every hidden analytic coordinate.

For the $\mathsf{Real}^{\uparrow}$ seal consumer, use Definition 438.20. The seal row Q factors through the located scope package and the tail-window classifier, so its read is the same finite surface exported by the public certificate.

For the bounded monotone witness consumer, use chapter 535. That chapter reads the monotone source, witness rows, located trap, and seal handoff through the displayed route $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Restricting to the $\mathsf{MonotoneCauchy}^{\uparrow}$ component leaves exactly $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so both consumers share the same tail-window surface. □

Theorem 438.23 (Monotone Cauchy public route classifier handoff)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Every public completion-facing classifier read and every* $\mathsf{Real}^{\uparrow}$ *seal-facing classifier read of M factors through the same ordered surface: the located interval trap I, the monotone tail-window certificate over λ, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row F, the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail-stability row μ, and the* $\mathsf{Real}^{\uparrow}$ *seal row Q, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows in H, C, P, N. No public route may replace this surface by a quotient stream, host sequence equality, selected limit, ambient completeness principle, trichotomy row, or choice datum.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier public route classifier handoff`

Proof. Begin with the carrier of Definition 438.1. It displays $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as the only regular sequence, schedule, modulus, located ledger, interval, seal, transport, continuation, provenance, and naming rows available to a public classifier.

Apply Definition 438.3 and Definition 438.7. The located interval row I is reached only after the regular sequence, scheduled window, modulus row, monotone dyadic ledger, componentwise hsame transport, continuation routes, package row, and local naming row have been displayed; the Cauchy and monotone tail rows remain attached to that same window.

Next use Definition 438.8 and Definition 438.22. Seal-facing and completion-facing consumers therefore read the same ordered surface before reaching Q, and the bounded witness route restricts to the $\mathsf{MonotoneCauchy}^{\uparrow}$ rows without changing the tail-window classifier.

Finally apply Definition 438.5 and the scope package of Definition 438.17. Every public coordinate has already appeared in the displayed ledger $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so no unlisted quotient stream, host sequence equality, selected limit, ambient completeness principle, trichotomy row, or choice datum can serve as a replacement route. □

Theorem 438.24 (Monotone Cauchy mature route summary)**.** *For an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N),$$

the carrier, common-window classifier, modulus-tail stability, located interval handoff, $\mathsf{Real}^{\uparrow}$ *seal boundary, public tail-window certificate, and bridge-facing certificate assemble one finite located monotone Cauchy route over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The route exposes the displayed regular rational sequence, scheduled tail window, modulus row, located dyadic ledger, located interval row, seal row, transports, continuations, provenance, and naming row, and it excludes quotient reals, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, and choice.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier mature route summary`

Proof. Begin with the carrier of Definition 438.1. It displays $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as the only regular sequence, schedule, modulus, located ledger, interval, seal, transport, continuation, provenance, and naming rows.

Apply Definition 438.6 and Definition 438.7. The common-window classifier fixes the finite observations of F through σ and μ, while modulus-tail stability keeps the Cauchy and monotone rows attached to that window under H and C.

Next use Definition 438.3 and Definition 438.4. The located interval row I and the seal row Q are therefore read only after the finite tail-window and dyadic ledger have been displayed.

Finally apply Definition 438.21 and Definition 438.22. These certificates package the same finite surface for public and bridge-facing consumers and exclude quotient reals, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, and choice. □

Theorem 438.25 (Monotone Cauchy bounded-witness tail factorization)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. If an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet consumes the tail-window rows, located interval rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows of* M*, then every bounded-witness public tail read factors through the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *common-window classifier, the located-interval handoff, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the ledger-exhaustion rows before it reaches the bounded-witness surface.*

Thus the bounded witness receives the displayed rows $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *as a finite tail-window route, not as a quotient real, host equality, trichotomy row, ambient completeness coordinate, or selected limiting representative.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier bounded witness tail factorization`

Proof. First expose the tail window by applying Definition 438.6. The scheduled observations F, σ and the modulus row μ determine the finite window through which any accepted bounded-witness tail read can inspect M.

Next apply Definition 438.3. The located interval row I is reached only after the regular sequence, scheduled window, modulus, dyadic ledger, transports, continuations, package row, and local certificate have already been displayed.

For the seal side, use Definition 438.4 and the bounded-witness handoff surface cited in Definition 438.22. The seal row Q is therefore consumed through the same tail-window and located-interval route that the bounded-witness packet reads.

Finally apply Definition 438.5. Every public coordinate has already appeared in the displayed rows $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the bounded-witness consumer has no remaining route to a quotient real, host equality, trichotomy row, ambient completeness coordinate, or selected limiting representative. □

Theorem 438.26 (Monotone Cauchy diagonal terminal pullback handoff)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet whose public* $\mathsf{Real}^{\uparrow}$ *seal is consumed by an accepted* DiagonalLimitCompatibili *terminal packet. Then every completion-facing classifier read and every* $\mathsf{Real}^{\uparrow}$ *seal-facing classifier read of* M *factors through the same terminal finite pullback over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the monotone tail-window rows* F, σ, μ, λ, *the located-interval row* I, *and the displayed* BHist/hsame/Cont/Pkg/NameCert *support rows in* H, C, P, N.

The handoff exports no quotient stream, host real equality, selected limit, trichotomy row, ambient completeness principle, metric-completeness datum, bridge row, or choice datum.

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier diagonal terminal pullback handoff`

Proof. First apply Definition 438.23. The public completion-facing classifier and the $\mathsf{Real}^{\uparrow}$ seal-facing classifier of M already read the same located interval trap, monotone tail-window certificate, $\mathsf{RegSeqRat}^{\uparrow}$ row, $\mathsf{CauchyModulus}^{\uparrow}$ tail-stability row, $\mathsf{Real}^{\uparrow}$ seal row, and support rows.

Next use Definition 438.25. It keeps the bounded tail-window and located-interval reads downstream of the same $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ surface, so the monotone source cannot contribute a second consumer route through a quotient real, trichotomy row, or selected limiting representative.

Now compose that surface with the diagonal terminal packet by applying Definition 646.173. The consumer hypothesis identifies the public $\mathsf{Real}^{\uparrow}$ seal of M with the terminal $\mathsf{Real}^{\uparrow}$ read exported by the diagonal packet, so the two packets share one finite seal handoff over the displayed support rows.

Finally apply Definition 646.172 and Definition 126.504. The diagonal terminal route is complete exactly through its finite terminal route, and the L10 exit boundary identifies all completion-facing and Real-seal-facing reads with the terminal pullback over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the displayed support rows. The cited non-escape clauses exclude quotient streams, host real equality, selected limits, trichotomy, ambient completeness, metric-completeness data, bridge rows, and choice. □

Theorem 438.27 (Monotone Cauchy cofinal criterion seal readback)**.** *Let*

$$M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet. Suppose its* $\mathsf{Real}^{\uparrow}$ *seal row* Q *is consumed both through a* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *route and through an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *finite criterion route, and*

suppose the same seal-facing read is also visible to the diagonal terminal pullback route. Then every seal-facing read of M factors through the single finite route $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, together with the sibling finite-window, dyadic-threshold, shared-tail, and criterion rows.

The readback has no quotient stream, selected representative, host real equality, trichotomy row, ambient completeness principle, metric-completeness datum, bridge row, or choice datum. Its entire public surface is the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support over the finite Monotone Cauchy, tail-cofinality, CauchyCriterion, and diagonal terminal rows.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyUp.MonotoneCauchyCarrier cofinal criterion seal readback`

Proof. First apply Definition 438.15. The $\mathsf{TailCofinalitySchedule}^{\uparrow}$ consumer must read the seal through the monotone ledger, Cauchy modulus row, located trap, finite tail-window schedule, dyadic threshold ledger, and located interval row before Q becomes available.

Next use Definition 438.12. The accepted $\mathsf{CauchyCriterion}^{\uparrow}$ consumer reads the same $\mathsf{Real}^{\uparrow}$ seal only through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ chain, with the shared-tail and criterion rows still attached to the finite carrier surface.

Now apply Definition 438.26. The diagonal terminal packet identifies the completion-facing and Real-seal-facing classifier reads with the same finite terminal pullback, so the cofinal schedule and criterion consumers cannot split into a second completion or representative route.

Finally use Definition 438.5. Every public read has already appeared in the displayed rows $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and their sibling finite-window, dyadic-threshold, shared-tail, criterion, and terminal rows. Since those carriers expose no coordinate for the excluded analytic or choice data, the composite seal readback is exhausted by the displayed BEDC ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneCauchy}^{\uparrow}$ packet is generated from a displayed regular rational sequence, a stream schedule, a Cauchy modulus row, dyadic monotonicity and located upper-bound ledgers, a located interval row, a real-seal row, componentwise transport, displayed continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope registers carrier admission, common finite-window classification, monotone-row transport, located interval handoff, $\mathsf{Real}^{\uparrow}$ seal non-escape, ledger exhaustion, and the mature route summary of Definition 438.24 for located monotone Cauchy packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient reals, selected representatives, host real equality, trichotomy, open-cover compactness, ambient completeness, or choice.

Upgrade path: No further paper-side theory-axis work remains; the remaining work is formal and bridge verification of the same finite tail-window, located-interval, real-seal, transport, package, and naming rows.

Chapter 439

A Concrete Naming Certificate for $\mathsf{IsometricEmbedding}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the metric-space treatment in Kreyszig, *Introductory Functional Analysis*, the $\mathsf{IsometricEmbedding}^{\uparrow}$ interface records a distance-preserving injection only as a finite BEDC certificate. It is a completion-boundary object for $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, and $\mathsf{HausdorffCompletion}^{\uparrow}$: downstream consumers see displayed metric and distance rows, not a host function equality or a quotient map.

Definition 439.1 (IsometricEmbedding carrier)**.** *An* $\mathsf{IsometricEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
3. *F is the graph row displaying the embedding as a finite BEDC relation from source observations to target observations;*
4. *D_X is the source* MetricDistanceWitness *row for displayed source endpoints;*
5. *D_Y is the target* MetricDistanceWitness *row for the displayed target endpoints selected by F;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-reflection row comparing the real-distance read from D_X with the real-distance read from D_Y;*
7. *T stores the componentwise* hsame *transports for X, Y, F, D_X, D_Y, and R;*
8. *Q stores the* Cont *routes for source reads, graph reads, target reads, and real-distance reflection reads;*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, MetricDistanceWitness, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IsometricEmbedding}^{\uparrow}}$ *row.*

The carrier contains no quotient map, no host function object, no selected representative of a completion class, and no external completion theorem.

Lean4 (definition) `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier`

Definition 439.2 (IsometricEmbedding classifier)**.** *For accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *carriers* I *and* I'*, the classifier* $\mathsf{IsometricEmbeddingClassifier}(I, I')$ *compares the packet rows componentwise. The source and target rows compare through their* $\mathsf{Met}^{\uparrow}$ *classifiers, the graph rows compare through the displayed relation row, the two distance-witness rows compare through their* $\mathsf{Real}^{\uparrow}$ *distance observations, and the distance-reflection row compares only after the same source and target distance reads have been transported.*

The classifier has no component for host-level function equality, quotient-map equality, selected representatives, or ambient completion principles. Its entire comparison surface is the displayed hsame*,* Cont*,* Pkg*, and local* NameCert *data.*

Lean4 (definition) `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingClassifier`

Theorem 439.3 (IsometricEmbedding NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{IsometricEmbedding}^{\uparrow}}$ *surface for the carrier of Definition 439.1 consists of five finite rows:*

1. *carrier admission by the displayed packet* I*;*
2. *classifier transport by the componentwise comparison surface of Definition 439.2;*
3. *distance preservation from the source* MetricDistanceWitness *row* D_X *to the target* MetricDistanceWitness *row* D_Y *through the* $\mathsf{Real}^{\uparrow}$ *reflection row* R*;*
4. *reflection into separated completion-facing consumers through the displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
5. *non-escape from quotient maps, host functions, selected representatives, and external completion theorems.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier namecert obligations`

Proof. Carrier admission is projection of the finite packet. The source metric row, target metric row, graph row, two distance-witness rows, real-distance reflection row, transport rows, continuation rows, package provenance, and local naming row are all visible coordinates.

Classifier transport is componentwise. The source and target classifiers move the endpoint rows, the graph row moves the displayed embedding relation, and the distance-witness rows move only along the transported endpoint data.

Distance preservation is then read from R. Since R is paired with D_X and D_Y in the carrier, the target distance read cannot be separated from the source distance read that generated it.

The separated completion-facing row consumes the same displayed metric and real-distance data. The package and naming rows record that every downstream read remains inside this surface, so no quotient map, host function, selected representative, or external completion theorem is exported. □

Theorem 439.4 (IsometricEmbedding distance reflection)**.** *Let* $I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$ *be an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *carrier. Any source metric-distance read exposed by* D_X *reflects through the graph row* F *into the target metric-distance read exposed by* D_Y*, and the comparison of those distances is made only through the displayed* $\mathsf{Real}^{\uparrow}$ *row* R *together with the stored* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier distance reflection`

Proof. Project the accepted carrier. The source distance read is stored in D_X, and the only target endpoint route attached to it is the displayed graph row F.

Read the target distance through D_Y. The transport row T keeps the source endpoint, graph endpoint, and target endpoint data aligned, while Q records the continuation route by which the target distance is consumed.

Finally read the real-distance comparison through R. The carrier has no separate host function or quotient-map coordinate, so the reflected target distance is the displayed $\mathsf{Real}^{\uparrow}$ comparison attached to D_X and D_Y. □

Theorem 439.5 (IsometricEmbedding completion boundary)**.** *Every* $\mathsf{CompletionReflection}^{\uparrow}$ *or* $\mathsf{HausdorffCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *carrier may use the embedding only through the displayed source and target* $\mathsf{Met}^{\uparrow}$ *rows, the source and target* $\mathsf{MetricDistanceWitness}$ *rows, the* $\mathsf{Real}^{\uparrow}$ *distance-reflection row, and the stored* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance. No completion-facing consumer receives a quotient map, host function equality, selected representative, or external completion theorem from the packet.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier completion boundary`

Proof. Start with a completion-facing consumer. Its metric input must read either the source row X, the target row Y, or the graph row F, because those are the only carrier coordinates exposing the embedding.

The distance input is then forced through D_X, D_Y, and R. By Definition 439.4, this is exactly the displayed source-to-target real-distance reflection route.

Apply the non-escape row of Definition 439.3. The packet inventory is exhausted by metric, real-distance, transport, continuation, package, and naming rows, so completion and separated-completion consumers cannot obtain non-displayed quotient, function, representative, or completion-theorem data. □

Theorem 439.6 (IsometricEmbedding carrier admission)**.** *Let*

$$I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$$

be an accepted $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet. The carrier admitted by the local* $\mathsf{NameCert}_{\mathsf{IsometricEmbedding}^{\uparrow}}$ *surface is exactly the displayed* BHist *packet with source and target* $\mathsf{Met}^{\uparrow}$ *rows, graph row, source and target distance witnesses,* $\mathsf{Real}^{\uparrow}$ *distance-reflection row, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local naming row. No admitted carrier field is a host function, quotient map, completion representative, or external completion theorem.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.TasteGate.IsometricEmbeddingTasteGate single carrier alignment`

Proof. Project the carrier definition Definition 439.1. Its ten coordinates are the complete visible inventory of the accepted packet.

Read the carrier-admission row of Definition 439.3. That row admits the packet by those coordinates and by no separate analytic object, because the package and local certificate only certify the displayed BHist surface.

The excluded objects have no coordinate in I. Therefore the admitted carrier is precisely the metric, graph, distance, reflection, transport, route, provenance, and naming packet. □

Theorem 439.7 (IsometricEmbedding classifier transport)**.** *For accepted packets* $I = (X, Y, F, D_X, D_Y, R, T, Q, P,$ *and* $I' = (X', Y', F', D'_X, D'_Y, R', T', Q', P', N')$*, any classifier comparison between* I *and* I' *is componentwise: source and target rows are compared by their* $\mathsf{Met}^{\uparrow}$ *classifiers, graph rows by the*

displayed relation row, distance witnesses by their endpoint and $\mathsf{Real}^{\uparrow}$ *distance observations, and reflection rows only after the same transported source and target distance reads have been displayed.*

Proof. Start from the classifier definition Definition 439.2. It gives no global equality test for the packet and no host function equality; each visible row is compared at its own BEDC surface.

Transport the source and target rows through the corresponding components of T and T'. The graph rows then compare the displayed source observations with the displayed target observations, and the distance-witness rows compare only the endpoint data selected by those transported graph rows.

Finally read the reflection rows R and R'. Since the classifier waits for the source and target distance observations before comparing the $\mathsf{Real}^{\uparrow}$ reflection data, classifier preservation is componentwise and remains inside the displayed hsame, Cont, Pkg, and NameCert provenance. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier classifier transport`

Theorem 439.8 (IsometricEmbedding separated consumer route)**.** *Every separated completion-facing consumer of an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet* $I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$ *factors through the displayed metric rows* X, Y*, the graph row* F*, the distance witnesses* D_X, D_Y*, the* $\mathsf{Real}^{\uparrow}$ *reflection row* R*, and the provenance rows* T, Q, P, N*. The consumer receives a distance-preserving route, not a quotient map, host function equality, selected representative, or ambient completion principle.*

Proof. Project the consumer route from Definition 439.5. A completion-facing read can access the embedding only by reading the metric and graph rows of the accepted packet.

Apply Definition 439.4. The distance data then pass from D_X through the graph row to D_Y, and the comparison is consumed only through the $\mathsf{Real}^{\uparrow}$ reflection row R.

The package rows T, Q, P, N record transport, continuation, provenance, and local naming for this same finite packet. Since no excluded completion object occurs in the carrier, the separated consumer route is exactly the displayed metric-distance route. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier separated consumer route`

Theorem 439.9 (IsometricEmbedding distance preservation obligation)**.** *Let*

$$I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$$

be an accepted $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet. The distance-preservation obligation says that every displayed source-distance read in* D_X *is transported along the graph row* F *to the displayed target-distance read in* D_Y*, and the equality of the two distance observations is read only through the* $\mathsf{Real}^{\uparrow}$ *row* R *and the provenance rows* T, Q, P, N*.*

Proof. Project the carrier and classifier surfaces of Definition 439.1 and Definition 439.2. They bind the source metric row X, target metric row Y, graph row F, distance witnesses D_X, D_Y, and real-distance row R as one finite packet.

Apply the NameCert obligations of Definition 439.3. The distance-preservation row is the clause that pairs the source distance witness with the target distance witness through R.

Finally use distance reflection. Since no host function or quotient map is exported, the preservation claim is exactly the displayed source-to-target real-distance comparison. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier distance preservation obligation`

Theorem 439.10 (IsometricEmbedding completion reflection handoff)**.** *Every* $\mathsf{CompletionReflection}^{\uparrow}$ *consumer of an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet* $I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$ *receives its embedding input through the displayed metric rows* X, Y*, graph row* F*, distance witnesses* D_X, D_Y*, real-distance reflection row* R*, and provenance rows* T, Q, P, N*. The handoff supplies a separated completion-facing route, not a quotient map, host function equality, selected representative, or ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier completion reflection handoff`

Proof. Begin with the completion boundary theorem Definition 439.5. It restricts every completion-facing consumer to the metric, graph, distance, reflection, transport, route, package, and naming rows of the accepted packet.

Use classifier transport to keep these rows componentwise under the displayed hsame and Cont routes. The NameCert obligations then identify the same rows as the local certificate surface.

The separated consumer route theorem supplies the final handoff: downstream consumers obtain a distance-preserving route through the displayed packet and receive no completion object outside it. □

Theorem 439.11 (IsometricEmbedding Hausdorff separation boundary)**.** *Every* $\mathsf{HausdorffCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet can use separatedness only through the displayed rows*

$$X, Y, F, D_X, D_Y, R, T, Q, P, N.$$

The packet may certify the finite distance-reflection route needed by a separated consumer, but it does not identify completion classes, choose representatives, export quotient maps, or assert an external Hausdorff-completion theorem.

Proof. Read the packet through the carrier definition. The only rows visible to a Hausdorff separatedness consumer are the source and target metric rows, the graph row, the two distance witnesses, the real-distance reflection row, and the transport and provenance rows.

Apply the distance-preservation obligation and the separated consumer route. Together they give the finite route from source distance data to target distance data and then to the separated completion-facing consumer.

The completion boundary theorem supplies the non-escape boundary. Because the carrier has no quotient map, selected representative, or ambient completion theorem, separatedness remains a consumer boundary over the displayed distance route. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier hausdorff separation boundary`

Theorem 439.12 (IsometricEmbedding real-distance route determinacy)**.** *For every accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet* $I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$*, each source-target distance comparison routes through exactly the displayed metric rows* X, Y*, graph row* F*, distance witnesses* D_X, D_Y*, real-distance row* R*, transport rows* T*, continuation route* Q*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{IsometricEmbedding}^{\uparrow}}$ *row* N*. The route exports no host function equality, quotient completion class, or selected representative.*

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier real distance route determinacy`

Proof. Start with carrier admission. The admitted packet is precisely the displayed BHist surface with source and target metric rows, graph row, distance witnesses, real-distance reflection, transport, continuation, package provenance, and local naming.

Apply classifier transport and the distance preservation obligation. Classifier transport keeps packet comparison componentwise, while distance preservation identifies the only source-to-target distance comparison as the passage from D_X through F to D_Y and then to the $\mathsf{Real}^{\uparrow}$ row R.

Use the separated consumer route together with the Hausdorff separation boundary. A separated or Hausdorff completion-facing consumer receives only the finite distance route $X, Y, F, D_X, D_Y, R, T, Q, P, N$ and no quotient map, host function equality, selected representative, completion class, or ambient completion theorem.

Therefore the real-distance route is determined by the displayed BEDC rows. Any downstream completion or Hausdorff consumer must cite those rows rather than an external isometry object. □

Theorem 439.13 (IsometricEmbedding sibling completion lattice)**.** *The* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet is read in the concrete-instance lattice after the metric and real-distance witnesses of chapter 57 and chapter 38, and before completion-facing consumers in chapter 403 and chapter 402. Its carrier consumes* $\mathsf{Met}^{\uparrow}$ *endpoint rows and* $\mathsf{Real}^{\uparrow}$ *distance rows, then exposes only the displayed completion-facing route* $X, Y, F, D_X, D_Y, R, T, Q, P, N$*.*

Proof. Start with the carrier and distance-preservation obligation. The source and target rows X, Y are metric rows, while D_X, D_Y and R place their distance comparison in the $\mathsf{Real}^{\uparrow}$ surface.

Apply real-distance route determinacy. It fixes the only source-target comparison route as $X, Y, F, D_X, D_Y, R, T, Q, P, N$, so downstream readers do not receive a host isometry object or a quotient map.

The completion reflection handoff and Hausdorff separation boundary identify the consumers that read this packet. Therefore $\mathsf{IsometricEmbedding}^{\uparrow}$ sits between the sibling metric and real chapters and the completion-facing chapters as a finite dependency lattice over displayed distance data. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier sibling completion lattice`

Theorem 439.14 (IsometricEmbedding distance-route reflection lock)**.** *Every separated or completion-facing consumer comparing source and target distances in an accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *packet must route through the displayed metric rows* X, Y*, graph row* F*, distance witnesses* D_X, D_Y*,* $\mathsf{Real}^{\uparrow}$ *reflection row* R*, transport row* T*, continuation row* Q*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. The route cannot be replaced by host function equality, a quotient map, a selected representative, or an ambient completion theorem.*

Proof. Start with carrier admission and classifier transport. Definition 439.6 exposes exactly $X, Y, F, D_X, D_Y, R, T, Q, P, N$, and Definition 439.7 keeps comparisons componentwise on those rows.

Apply the distance-preservation obligation. Definition 439.9 identifies the distance comparison as the displayed passage from the source witness D_X through the graph row F to the target witness D_Y and then to the $\mathsf{Real}^{\uparrow}$ reflection row R.

Now use the separated and completion-facing boundaries. Definition 439.8, Definition 439.5, and Definition 439.11 confine separated and completion-facing consumers to the same finite packet.

Finally invoke real-distance route determinacy. Definition 439.12 rules out host function equality, quotient completion classes, selected representatives, quotient maps, and ambient completion theorems as substitutes for the displayed BEDC route. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier distance route reflection lock`

Theorem 439.15 (IsometricEmbedding obligation-closure surface)**.** *The obligation surface for* IsometricEmbedding$^{\uparrow}$ *is exhausted by the local carrier admission, classifier transport, distance-preservation obligation, separated consumer route, completion-reflection handoff, Hausdorff separation boundary, and real-distance route determinacy. Equivalently, every accepted packet can be consumed only through the displayed rows $X, Y, F, D_X, D_Y, R, T, Q, P, N$ and not through host function equality, quotient maps, selected representatives, ambient completion existence, or external Hausdorff separation principles.*

Proof. Start with carrier admission and classifier transport. Apply Definition 439.6 and Definition 439.7 to fix the displayed packet rows and keep comparisons componentwise.

Use the distance-preservation obligation next. By Definition 439.9, the source-to-target comparison is the route from D_X through F to D_Y and then to the Real$^{\uparrow}$ reflection row R.

Now bind the downstream consumers. The separated route of Definition 439.8, the completion-reflection handoff of Definition 439.10, and the Hausdorff boundary of Definition 439.11 leave no additional completion-facing row.

Finally apply Definition 439.12 and Definition 439.14. These results identify the same finite route as the only real-distance and completion-facing surface, so the listed local theorems exhaust the obligation-closure surface. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier obligation closure surface`

Theorem 439.16 (IsometricEmbedding scoped kernel route)**.** *Let $I = (X, Y, F, D_X, D_Y, R, T, Q, P, N)$ be an accepted* IsometricEmbedding$^{\uparrow}$ *packet. Every distance-preservation read from I is scoped by the* Met$^{\uparrow}$ *endpoint rows X, Y, the source and target distance rows D_X, D_Y, the* Real$^{\uparrow}$ *distance-reflection row R, componentwise* hsame *transport through T, displayed* Cont *replay through Q,* Pkg *provenance through P, and the local* NameCert *row N. The route is the displayed kernel surface fixed by the carrier, classifier, distance reflection, real-distance route determinacy, and completion-facing sibling lattice; it exports no host function equality, quotient map, selected representative, ambient completion theorem, or external Hausdorff separation principle.*

Proof. Begin with carrier admission and classifier transport. Definition 439.6 exposes exactly $X, Y, F, D_X, D_Y, R, T, Q, P, N$, and Definition 439.7 keeps those rows componentwise under the displayed transport and replay data.

Use distance reflection and distance preservation. Definition 439.4 and Definition 439.9 force every distance-preservation read from the source metric row through F, D_Y, and the Real$^{\uparrow}$ reflection row R.

Apply Definition 439.12 and the sibling lattice route of Definition 439.13. Together they identify the same finite route as the completion-facing kernel surface available to separated and completion consumers.

The local NameCert row and Pkg provenance then bind the scope. Since no carrier coordinate stores a host isometry, quotient map, selected representative, ambient completion theorem, or external Hausdorff principle, the listed rows are the full scoped kernel route. □

Lean4 ✓ `BEDC.Derived.IsometricEmbeddingUp.IsometricEmbeddingCarrier scoped kernel route`

Closure status for NameCert$_{\text{IsometricEmbedding}^{\uparrow}}$**.**

Constructive story: An IsometricEmbedding$^{\uparrow}$ packet is generated from source and target Met$^{\uparrow}$ rows, a displayed graph row, source and target MetricDistanceWitness rows, a Real$^{\uparrow}$ distance-reflection row, componentwise transport, continuation, package provenance, and local naming data.

Theory closure: TheoryGate(NameCert$_{\mathsf{IsometricEmbedding}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public export is the displayed isometric-embedding consumer route over Met$^{\uparrow}$, Real$^{\uparrow}$ distance rows, hsame, Cont, Pkg, and NameCert, with carrier, distance-preservation, classifier transport, and completion-facing bridge rows exposed for downstream use.

Formal verification: FormalStatus(NameCert$_{\mathsf{IsometricEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host-level isometry theorems, quotient maps, selected representatives, ambient completion existence, extension uniqueness, or bridge checking.

Upgrade path: The next closure grade requires mature examples and downstream comparison for completion-facing consumers; bridge checking remains a separate formal-axis obligation.

Chapter 440

A Concrete Naming Certificate for $\mathsf{Sobolev}^{\uparrow}$

The $\mathsf{Sobolev}^{\uparrow}$ interface records the finite naming surface for a Sobolev-style weak-derivative packet. Classical references such as Evans, *Partial Differential Equations*, and Adams–Fournier, *Sobolev Spaces*, describe the surrounding analytic theory. BEDC records only the finite certificate surface needed by later PDE, variational, compact-embedding, and functional-analysis consumers: a displayed domain row, metric or complete-metric base row, normed or Hilbert codomain row, integrability magnitude row, weak-gradient test row, package provenance, and local naming evidence.

The packet consumes the $\mathsf{Real}^{\uparrow}$ domain surface of chapter 38, the metric or complete-metric base through chapter 57 or chapter 128, and the weak-gradient magnitude route through the $\mathsf{Integral}^{\uparrow}$ and $\mathsf{Derivative}^{\uparrow}$ sibling certificates of chapter 123 and chapter 122. These dependencies are read as finite carrier rows and ledger routes, not as a host Sobolev space or completed function quotient.

The packet is not distribution theory, not Lebesgue completion, not quotient equality of functions, and not a completed function space. It is a finite BHist carrier whose accepted observations are compared by displayed rows and hsame routes.

Definition 440.1 (Sobolev BHist carrier)**.** *A* $\mathsf{Sobolev}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, M, V, L, G, H, C, P, N)$$

with the following displayed coordinates:

1. *D is the* $\mathsf{Real}^{\uparrow}$ *domain row naming the scalar observation surface on which the weak tests are read;*
2. *M is a* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row naming the ambient finite metric information used by the domain consumer;*
3. *V is a* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row naming the magnitude target for function and weak-gradient observations;*
4. *L is the* $\mathsf{Integral}^{\uparrow}$ *magnitude row recording the finite integrability ledger for the displayed function and weak-gradient reads;*
5. *G is the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient test row, displayed as the finite list of test-observation routes and their recorded weak-derivative responses;*

6. *H stores the componentwise* hsame *routes for the domain, base, codomain, integrability, weak-gradient, provenance, and local naming rows;*
7. *C stores the displayed* Cont *routes by which a consumer reads the domain row, metric base row, codomain norm row, integrability ledger, and weak-gradient test ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a host function space, a distributional dual, a quotient by almost-everywhere equality, a completed Lebesgue measure space, or a choice-selected representative.

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevTasteGate single carrier alignment`

Theorem 440.2 (Sobolev weak-gradient source totality)**.** *Every weak-gradient read from an accepted* $\mathsf{Sobolev}^{\uparrow}$ *carrier* $S = (D, M, V, L, G, H, C, P, N)$ *factors through the displayed* $\mathsf{Real}^{\uparrow}$ *domain row D, the* $\mathsf{Derivative}^{\uparrow}$ *test row G, componentwise* hsame *transport in H, and* Cont *replay in C. The read cannot start from a host weak-gradient predicate, distributional dual, quotient function, or completed measure carrier; its source is the finite* BHist *route recorded by the packet.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak gradient source totality`

Proof. Project the carrier of Definition 440.1. The only domain source is D, and the only weak-gradient source is the displayed test row G.

Next use H and C. Transport moves the domain and test rows componentwise, while replay exposes the same finite route to consumers without adding a background analytic object.

Finally repack the read through P and N. Since no carrier coordinate stores a host weak-gradient predicate, distributional dual, quotient function, or completed measure carrier, every weak-gradient source read is exhausted by D, G, H, C, P, N. □

Theorem 440.3 (Sobolev weak-derivative window obligation)**.** *Every Sobolev weak-derivative read from an accepted* $\mathsf{Sobolev}^{\uparrow}$ *carrier* $S = (D, M, V, L, G, H, C, P, N)$ *must expose finite test-function and derivative windows inside the displayed* $\mathsf{Derivative}^{\uparrow}$ *row G, the distributional pairing rows recorded by the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger L, the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route carried by the domain and magnitude rows, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *evidence N before any norm consumer may read it. The route exports no host distributional dual, quotient-function equality, selected representative, or ambient completed Sobolev space.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative window obligation`

Proof. Project the displayed carrier rows. The weak-derivative source is not an ambient predicate; it is the derivative-facing row G attached to the domain, base, codomain, and magnitude rows D, M, V, L.

Next read the test-function and derivative windows. The finite derivative tests are visible through G, while the distributional pairings and magnitude checks are visible through L over the same $\mathsf{Real}^{\uparrow}$ domain row and regular readback route.

Now expose the structural rows. Componentwise hsame transport H, displayed Cont replay C, provenance P, and local naming N keep the derivative window attached to the same carrier before a norm consumer may read it.

Since the carrier has no coordinate for a distributional dual, quotient function space, selected representative, or completed Sobolev carrier, the weak-derivative window is exhausted by the displayed finite route. □

Theorem 440.4 (Sobolev integral magnitude ledger stability)**.** *For every accepted* $\mathsf{Sobolev}^{\uparrow}$ *carrier* $S = (D, M, V, L, G, H, C, P, N)$*, the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L *transports componentwise through* H *and remains attached to the same displayed domain, codomain, and weak-gradient rows under* Cont *replay. The transported magnitude ledger exposes no Lebesgue-completion row, quotient-function row, selected representative, or host measure-space equality.*

Proof. Start with the magnitude coordinate L. It is a finite $\mathsf{Integral}^{\uparrow}$ row attached to the displayed domain D, codomain V, and weak-gradient tests G.

Transport is componentwise through H. Thus the magnitude row moves only with its visible domain, codomain, weak-gradient, provenance, and local naming coordinates.

Replay through C preserves that attachment for downstream consumers. The omitted completion, quotient, representative, and host-equality objects are not coordinates of the packet, so the magnitude ledger cannot expose them after transport. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier integral magnitude ledger stability`

Theorem 440.5 (Sobolev obligation package)**.** *The rows* $D, M, V, L, G, H, C, P, N$ *of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *carrier form the finite* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *obligation package consumed by PDE, variational, compact-embedding, and functional-analysis successors. The package consists of carrier admission, domain and metric-base exposure, codomain magnitude exposure, integral magnitude accounting, weak-gradient test routing, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming evidence. It exports no distribution theory, quotient function equality, Lebesgue completion, trace theorem, compact embedding theorem, or completed Sobolev space.*

Proof. Read the carrier inventory. The displayed rows $D, M, V, L, G, H, C, P, N$ are the complete finite packet and provide the only source for successor chapters.

The analytic-facing rows are D, M, V, L, G. They expose the Real domain, metric or complete-metric base, normed or Hilbert codomain, integral magnitude ledger, and derivative test ledger before any successor can consume Sobolev data.

The structural rows H, C, P, N then transport, replay, source, and name the same packet. Coordinate exhaustion excludes distribution theory, quotient function equality, Lebesgue completion, trace, compact embedding, and completed Sobolev-space objects from the obligation package. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier obligation package`

Definition 440.6 (Sobolev classifier)**.** *Two accepted* $\mathsf{Sobolev}^{\uparrow}$ *carriers are classified as the same weak-derivative certificate exactly when their visible domain, base, codomain, magnitude, weak-gradient, provenance, and naming rows compare componentwise through the stored* hsame *routes. The classifier compares the* $\mathsf{Real}^{\uparrow}$ *domain row, the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row, the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row, the* $\mathsf{Integral}^{\uparrow}$ *magnitude row, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient test row, the* Pkg *row, and the local* $\mathsf{NameCert}$ *row.*

The classifier does not compare host functions, ambient distributions, completed L^p *classes, quotient representatives, or equality in a background Sobolev space.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevFiniteWindowClassifierSurface`

Theorem 440.7 (Sobolev NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *obligation surface for the carrier of Definition 440.1 consists of five finite rows:*

1. *carrier habitation by the zero weak-gradient packet, with a displayed* $\mathsf{Real}^{\uparrow}$ *domain row, metric base row, codomain norm row, zero* $\mathsf{Integral}^{\uparrow}$ *magnitude row, zero* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient test row, package provenance, and local naming evidence;*
2. *classifier reflexivity, symmetry, and transitivity by the componentwise classifier of Definition 440.6;*
3. *carrier transport under the classifier, preserving the same visible domain, metric base, codomain, integrability, weak-gradient, provenance, and naming rows up to* hsame*;*
4. *ledger exactness for weak-derivative observations, requiring every accepted derivative read to factor through the displayed* G *row and the magnitude ledger* L*;*
5. *completion-boundary non-escape, preventing any consumer from reading distribution theory, Lebesgue completion, quotient equality, or a host Sobolev space from the finite carrier.*

Proof. For habitation, use the displayed zero weak-gradient test row. The domain, metric base, codomain, zero magnitude, provenance, and local naming coordinates are all finite carrier rows, so no ambient function space is needed.

For classifier stability, compare accepted carriers by the rows named in Definition 440.6. Reflexivity, symmetry, and transitivity are inherited componentwise from the displayed hsame routes and are repacked into the same finite carrier shape.

For transport and exactness, inspect any downstream read. It can project only D, M, V, L, G, H, C, P, or N, and the weak-derivative read is the displayed route through G with its magnitude entry in L. Coordinate exhaustion gives the non-escape boundary. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier namecert obligation surface`

Theorem 440.8 (Sobolev obligation-closure export)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N$ *exports the paper-side obligation-closure package formed by the displayed rows* $D, M, V, L, G, H, C, P, N$*, the classifier of Definition 440.6, the weak-gradient ledger, and the integral magnitude ledger. The package supplies carrier admission, classifier stability, weak-gradient exactness, integral magnitude accounting, completion-boundary nonescape, componentwise transport, replay, provenance, and local naming for Sobolev consumers.*

The package still refuses distribution theory, quotient function equality, Lebesgue completion, trace theorem, compact embedding, and completed Sobolev spaces.

Proof. Project the carrier of Definition 440.1 and the obligation package of Definition 440.5. These expose exactly the rows $D, M, V, L, G, H, C, P, N$ and assign their domain, base, codomain, magnitude, weak-gradient, transport, replay, provenance, and naming roles.

Apply the classifier of Definition 440.6 and the NameCert obligation surface of Definition 440.7. They supply carrier admission, classifier stability, carrier transport, weak-derivative ledger exactness, and completion-boundary nonescape over the same finite rows.

Use Definition 440.9. It orders every weak-gradient read through D, M, V, L, G, with the $\mathsf{Integral}^{\uparrow}$ magnitude ledger L read before the $\mathsf{Derivative}^{\uparrow}$ weak-gradient row G.

Finally apply Definition 440.10. Since H, C, P, N only transport, replay, source, and name the same finite packet, the refused analytic objects cannot be recovered from the export package. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier obligation closure export`

Theorem 440.9 (Sobolev weak-derivative ledger)**.** *Every accepted weak-derivative observation exported by a* $\mathsf{Sobolev}^{\uparrow}$ *carrier factors through the finite route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G.$$

The route first fixes the displayed $\mathsf{Real}^{\uparrow}$ *domain row and metric base, then reads the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row, then records the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger before exposing the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient test row. It exports no distributional dual object, no completed* L^p *class, no quotient representative, and no host equality of functions.*

Proof. Start with a consumer request for a weak-derivative observation. Carrier projection forces the request to identify the domain row D and the metric or complete-metric base row M before any derivative data can be read.

The codomain and magnitude rows are then read through V and L. These rows specify the finite normed target and the accepted integrability ledger for the observation, not an ambient measure completion.

Only after that route reaches G does the weak-gradient test row become visible. Since H, C, P, and N record transport, continuation, provenance, and local naming without adding analytic coordinates, the exported observation is exactly the finite ledger described above. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative ledger`

Theorem 440.10 (Sobolev completion boundary)**.** *A* $\mathsf{Real}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, *functional-analysis, PDE, variational, or compact-embedding consumer citing an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet receives only the displayed carrier rows of Definition 440.1, the classifier of Definition 440.6, the five obligation rows of Definition 440.7, and the weak-derivative ledger of Definition 440.9. The citation cannot assert full distribution theory, Lebesgue completion, quotient equality of functions, density of smooth functions, compact embedding, trace theory, or a completed Sobolev space.*

Proof. Factor the consumer citation through the local $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ surface. The available data are the finite rows $D, M, V, L, G, H, C, P, N$ and the classifier routes between accepted packets.

Apply the weak-derivative ledger theorem to any derivative-facing read. It fixes the route through the displayed domain, base, codomain, magnitude, and weak-gradient rows before the observation is exported.

The excluded analytic objects are not coordinates of the packet and are not produced by the classifier or continuation rows. Therefore downstream consumers may use the finite weak-derivative certificate surface, but they cannot read any stronger completion, quotient, density, compactness, or trace result from this chapter. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier completion boundary`

Theorem 440.11 (Sobolev weak-derivative root-unblock package)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$ *exports a finite root package combining carrier admission, classifier stability, weak-derivative ledger exactness, metric or complete-metric boundary data,* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape. Any downstream analysis consumer must read this package through the displayed weak-gradient magnitude route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G$$

before reaching completion-facing rows.

The package supplies no distributional dual object, Lebesgue completion, quotient equality of functions, density theorem, compact embedding theorem, trace theorem, or completed Sobolev space.

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative root unblock package`

Proof. Project the carrier and classifier surfaces. The carrier rows $D, M, V, L, G, H, C, P, N$ are the complete visible inventory, and the classifier compares them componentwise through the stored hsame routes.

Apply the NameCert obligation surface. Carrier habitation, classifier stability, carrier transport, weak-derivative ledger exactness, and completion-boundary non-escape are already recorded as finite rows over the same packet.

Use the weak-derivative ledger theorem to identify the route used by every derivative-facing read. The read passes from the $\mathsf{Real}^{\uparrow}$ domain row through the metric or complete-metric base, the normed or Hilbert codomain row, the $\mathsf{Integral}^{\uparrow}$ magnitude ledger, and only then the $\mathsf{Derivative}^{\uparrow}$ weak-gradient test row.

Finally apply the completion boundary theorem. It confines downstream PDE, variational, functional-analysis, compact-embedding, and completion-facing consumers to the displayed carrier, classifier, obligation, and ledger rows. Coordinate exhaustion then gives the Real seal nonescape and rules out all stronger analytic objects listed in the statement. □

Theorem 440.12 (Sobolev weak-derivative carrier transport)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ and $S' = (D', M', V', L', G', H', C', P', N')$ be accepted $\mathsf{Sobolev}^{\uparrow}$ packets whose domain, base, codomain, magnitude, weak-gradient, provenance, and local naming rows are connected by the componentwise* hsame *entries recorded in H and H'. Any weak-derivative consumer transported from S to S' must carry the same* BHist *route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G$$

to

$$D' \longmapsto M' \longmapsto V' \longmapsto L' \longmapsto G'$$

through the displayed Cont *replay rows. The transport preserves the carrier, classifier, weak-derivative ledger, completion-boundary rows,* Pkg *provenance, and local* NameCert *evidence, and it supplies no distributional dual, quotient function class, completed measure carrier, compact embedding object, or trace row.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative carrier transport`

Proof. Begin with the classifier of Definition 440.6. It compares accepted packets only through the visible rows D, M, V, L, G, P, N and the componentwise hsame transport stored in H.

Next replay the consumer route through C and C'. Since every weak-derivative read is forced by Definition 440.9 to pass through the displayed domain, base, codomain, magnitude, and weak-gradient rows, transport replaces each row by its classified mate and keeps the same finite route.

Finally use the completion boundary theorem. The excluded analytic objects are absent from both carrier inventories, so transport across the classifier cannot create them. □

Theorem 440.13 (Sobolev classifier locality)**.** *The $\mathsf{Sobolev}^{\uparrow}$ classifier is local to the rows displayed by the carrier of Definition 440.1. A downstream consumer may decide sameness of accepted weak-derivative packets only by reading the* BHist *rows D, M, V, L, G, the* hsame *transport row H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* NameCert *row N. It cannot appeal to host function equality, equality in a quotient of measurable functions, distributional equality outside the displayed weak-gradient tests, ambient Hilbert completion, compact embedding, trace data, or density of smooth approximants.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier classifier locality`

Proof. Read the classifier definition. Its comparison surface is componentwise and mentions only the displayed domain, base, codomain, magnitude, weak-gradient, provenance, and naming rows.

Apply carrier transport. Any consumer trying to move a classified packet must replay the same Cont route through the same finite weak-derivative ledger, so the classifier remains local to the carrier inventory.

The nonlocal analytic objects listed in the statement are not coordinates of S. Coordinate exhaustion therefore blocks them from acting as classifier inputs or classifier witnesses. □

Theorem 440.14 (Sobolev root obligation package)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N$ *exports a root obligation package consisting of carrier admission, classifier stability, weak-gradient ledger exactness, completion-boundary nonescape, and* $\mathsf{Real}^{\uparrow}$ *seal nonescape. The package is read only through the displayed rows*

$$D, M, V, L, G, H, C, P, N.$$

The carrier row admits the packet, the classifier row keeps sameness componentwise, the weak-gradient ledger forces derivative-facing reads through D, M, V, L, G*, the completion boundary blocks ambient Sobolev completion, and the* $\mathsf{Real}^{\uparrow}$ *domain seal in* D *cannot be read outside the same finite route.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root obligation package`

Proof. Project the carrier of Definition 440.1. It lists exactly the displayed rows $D, M, V, L, G, H, C, P, N$, so carrier admission has no additional analytic coordinate.

Apply the classifier of Definition 440.6 and the NameCert obligation surface of Definition 440.7. These give classifier stability, carrier transport, weak-derivative ledger exactness, and completion-boundary nonescape over the same rows.

Use the weak-derivative ledger and completion boundary, Definition 440.9 and Definition 440.10. The derivative-facing route must pass through D, M, V, L, G, while completion-facing and Real-facing consumers remain inside the carrier and structural rows.

Finally repack the route through H, C, P, N. Since these rows only transport, replay, source, and name the same packet, the Real seal and completion boundary cannot escape the finite root obligation package. □

Theorem 440.15 (Sobolev weak-gradient consumer boundary)**.** *Every PDE, variational, and functional-analysis consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *weak-gradient read must pass through*

$$D, M, V, L, G, H, C, P, N$$

before reading Sobolev data. In particular, such a consumer first reads the $\mathsf{Real}^{\uparrow}$ *domain row* D*, the metric or complete-metric row* M*, the norm or Hilbert row* V*, the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger* L*, and the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row* G*, then uses* H, C, P, N *only for transport, replay, provenance, and local naming. It cannot read a host weak-gradient predicate, quotient function equality, distributional dual, Lebesgue completion, density theorem, compact embedding theorem, or trace theorem from the packet.*

Proof. Start with a downstream consumer request. The carrier definition forces the request to enter through the displayed inventory $D, M, V, L, G, H, C, P, N$.

Apply the weak-derivative ledger. It orders every derivative-facing observation from the Real domain row through the metric base, codomain, integral magnitude ledger, and derivative test row before the observation is exported.

Now use classifier locality and carrier transport. They keep sameness and transport local to the same finite rows, so PDE, variational, and functional-analysis consumers cannot substitute an ambient analytic object for the displayed route.

The completion boundary removes the excluded objects. Distributional duals, quotient equality, Lebesgue completion, density, compactness, trace, and host weak-gradient predicates are not coordinates of the packet, and H, C, P, N add only structural replay and naming. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak gradient consumer boundary`

Theorem 440.16 (Sobolev integral and derivative source package)**.** *In every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet, the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L *and the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row* G *are source rows for the weak-gradient ledger. Neither row can be replaced by distribution theory, Lebesgue completion, or an ambient quotient-function carrier:* L *supplies the displayed magnitude ledger,* G *supplies the displayed weak-gradient tests, and the route from* D *through* M, V, L, G *is the only analytic source package exported to downstream Real, metric, derivative, integral, Hilbert, PDE, variational, or functional-analysis consumers.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier integral derivative source package`

Proof. Project L and G from Definition 440.1. The row L is explicitly the integral magnitude ledger, and G is explicitly the derivative-facing weak-gradient test row.

Apply Definition 440.9. It places L immediately before G in the finite route from the Real domain row and metric base through the codomain row, so the weak-gradient read has a displayed source package.

Use Definition 440.10 and Definition 440.13. They state that downstream consumers and classifier reads remain inside the same finite carrier rows and cannot appeal to distribution theory, Lebesgue completion, or quotient equality.

Finally transport through H and replay through C. Provenance P and local naming N source the same L, G package, so no replacement analytic carrier can discharge the integral or derivative source obligations. □

Theorem 440.17 (Sobolev root-unblock certificate)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet exports a root-unblock certificate consisting exactly of the carrier row, local classifier, weak-derivative ledger, carrier transport row, classifier-locality row, completion-boundary row,* Pkg *provenance, and local* NameCert *evidence. Any Hilbert, metric, integral, derivative, PDE, variational, or compactness-facing consumer must first pass through these finite* BHist, hsame, *and* Cont *rows before reading the weak-gradient data. The certificate supplies no completed Sobolev space, no distributional dual, no quotient function equality, no selected representative, no density theorem, no compact embedding theorem, and no trace theorem.*

Proof. Project the accepted packet. The carrier exposes exactly $D, M, V, L, G, H, C, P, N$, and the NameCert obligation surface records habitation, classifier stability, transport, ledger exactness, and completion-boundary nonescape.

Use the weak-derivative ledger and carrier transport rows. They force every derivative-facing read to traverse the displayed domain, base, codomain, magnitude, and weak-gradient route, then preserve that route under componentwise hsame comparison.

Use classifier locality for downstream consumers. Hilbert, metric, integral, derivative, PDE, variational, and compactness-facing reads can cite only the local classifier and the replayed route, not a host analytic object outside the packet.

The completion boundary then removes the remaining escape routes. Since the excluded spaces, quotients, representatives, density results, compact embeddings, and traces are not carrier coordinates, the root-unblock certificate is exhausted by the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root unblock certificate`

Theorem 440.18 (Sobolev integral and derivative handoff)**.** *Every* $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *PDE, or variational consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet must traverse the displayed route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G \longmapsto H, C, P, N.$$

Thus weak-derivative data are consumed only through the finite integral and derivative ledgers. The handoff supplies no distribution theory, Lebesgue completion, quotient equality of functions, compact embedding theorem, trace row, selected representative, or completed Sobolev space.

Proof. Project the carrier rows of Definition 440.1. The only analytic-facing coordinates are the domain row D, the metric or complete-metric base row M, the codomain row V, the integral magnitude ledger L, and the weak-gradient test row G.

Apply the weak-derivative ledger theorem. It places every derivative-facing observation on the route from D through M, V, L and then to G, so the integral and derivative consumers read the same finite ledger rather than an ambient function space.

Use the root-unblock certificate to carry the route to Hilbert, metric, complete-metric, PDE, and variational consumers. The transport, replay, provenance, and local naming rows H, C, P, N preserve the displayed route without adding analytic coordinates.

Finally apply the completion boundary. The excluded distribution, completion, quotient, compactness, trace, representative, and completed-space objects are absent from the carrier inventory, so the handoff is exhausted by the finite route displayed above. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier integral derivative handoff`

Theorem 440.19 (Sobolev BHist carrier scope)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *read is scoped by the finite* BHist *carrier rows*

$$D, M, V, L, G, H, C, P, N$$

of Definition 440.1. A consumer may reach $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, *and* $\mathsf{Derivative}^{\uparrow}$ *only through those displayed rows, their componentwise* hsame *transports, their* Cont *replay route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}$ *evidence.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier bhist carrier scope`

Proof. Project the accepted carrier. The carrier definition lists exactly the domain row D, metric or complete-metric base M, codomain row V, magnitude ledger L, weak-gradient row G, and structural rows H, C, P, N.

Apply the classifier locality and carrier transport rows. They compare and move accepted packets only through the same displayed coordinates, with hsame and Cont preserving the route rather than adding a host analytic object.

Finally inspect a downstream analysis read. Since every readable sibling is named in P and replayed through C, the $\mathsf{Real}^{\uparrow}$, metric, complete-metric, norm, Hilbert, integral, and derivative surfaces can appear only as the finite rows already present in the BHist packet. □

Theorem 440.20 (Sobolev Hilbert and norm readback scope)**.** *Every Hilbert-facing or norm-facing readback from an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$ *or* $\mathsf{Norm}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows carried by*

$$D, M, V, L, G, H, C, P, N.$$

The route reads metric data through M*, Hilbert or norm data through* V*, integral magnitude through* L*, weak-derivative tests through* G*, Real and regular-rational data through the displayed domain and readback rows, and then the structural rows* H, C, P, N*. It exports no host function space, quotient equality, ambient Hilbert completion, or external completeness theorem.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier hilbert norm readback scope`

Proof. Project the carrier rows in order. The metric-facing coordinate is M, the Hilbert or norm-facing coordinate is V, the integral coordinate is L, and the derivative coordinate is G.

Read the Real and regular-rational surface through the same displayed domain and magnitude route. The domain row D anchors the scalar readback, while L and G carry the finite magnitude and weak-derivative observations that norm consumers inspect.

Transport and replay are then componentwise. The rows H, C, P, N preserve the same metric, Hilbert or norm, integral, derivative, Real, and regular-rational route and attach provenance and local naming.

Coordinate exhaustion gives the scope boundary. Because the carrier has no row for a host function space, quotient equality, ambient Hilbert completion, or external completeness theorem, none of those objects can be exported by a Hilbert or norm readback. □

Theorem 440.21 (Sobolev derivative norm ledger)**.** *Every derivative-facing* $\mathsf{Sobolev}^{\uparrow}$ *consumer first reads the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row* G *and the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L *through the displayed codomain row* V*. If* V *is Hilbert-facing, the same route reads* $\mathsf{Hilbert}^{\uparrow}$ *only through the finite* $\mathsf{Norm}^{\uparrow}$ *magnitude interface carried by* V *before the* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *row is exposed. Thus derivative magnitude, norm control,* hsame *transport,* Cont *replay, and* Pkg *provenance are one finite ledger, not a quotient Sobolev norm or a completed Hilbert-space argument.*

Proof. Begin with the integral and derivative handoff. It forces the consumer route through D, M, V, L, G before the structural rows are read.

The codomain row V is therefore fixed before L and G are interpreted. The $\mathsf{Integral}^{\uparrow}$ magnitude ledger reads weak-gradient size through L, and the $\mathsf{Derivative}^{\uparrow}$ row reads the corresponding weak-gradient tests through G.

If the codomain is Hilbert-facing, the Hilbert row remains the displayed codomain surface inside V. The route supplies norm data only by the finite norm interface already named there, and the later rows H, C, P, N merely transport, replay, source, and name the same derivative-norm ledger. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier derivative norm ledger`

Theorem 440.22 (Sobolev finite-window root package)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet exports its downstream-ready finite root package only through the rows*

$$D, M, V, L, G, H, C, P, N.$$

Here $\mathsf{Real}^{\uparrow}$ *is read through* D*,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *through* M*,* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *through* V*,* $\mathsf{Integral}^{\uparrow}$ *through* L*, and* $\mathsf{Derivative}^{\uparrow}$ *through* G*, with* H, C, P, N *carrying transport, replay, provenance, and local naming. Thus PDE, variational, functional-analysis, Hilbert, metric, integral, and*

derivative consumers receive a finite BHist *obligation package, not distributional duals, Lebesgue completion, quotient equality, density, compact embedding, trace rows, or selected representatives.*

Proof. Project the accepted carrier through Definition 440.1. Its visible coordinates are exactly $D, M, V, L, G, H, C, P, N$, and the local NameCert obligation surface records carrier admission, classifier stability, transport, ledger exactness, and completion-boundary non-escape over those rows.

Use the weak-derivative ledger and classifier locality rows. Every derivative-facing observation must traverse the displayed route from D through M, V, L to G, and every sameness decision remains local to the same carrier inventory and its componentwise hsame evidence.

Apply the derivative norm ledger. It places norm and Hilbert-facing magnitude reads inside V before the integral and derivative rows are exposed, so the finite root package is stable under downstream norm, integral, derivative, and completion-facing consumption.

The excluded analytic objects are absent from the carrier, classifier, weak-derivative ledger, and derivative norm ledger. Coordinate exhaustion therefore blocks distributional duals, Lebesgue completion, quotient equality, density, compact embedding, trace rows, and selected representatives from entering the exported root package. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier finite window root package`

Theorem 440.23 (Sobolev finite energy carrier)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet. Its finite energy carrier is exactly the displayed* BHist *route through the* $\mathsf{Real}^{\uparrow}$ *domain row D, the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row M, the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row V, the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger L, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row G, the componentwise* hsame *transport row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *row N. No Real, metric, integral, derivative, or Hilbert-facing consumer can read an energy datum outside these coordinates.*

Proof. Project the accepted carrier of Definition 440.1. The tuple $D, M, V, L, G, H, C, P, N$ is the complete coordinate list, and each analytic face is named by one of those rows.

Apply the NameCert obligation surface and Definition 440.22. Carrier admission, classifier stability, weak-derivative ledger exactness, and completion-boundary non-escape all refer back to the same finite inventory.

Coordinate exhaustion then handles the consumer side. A Real, metric, integral, derivative, or Hilbert read must enter through the named row assigned to that face; the remaining rows only transport, replay, source, and name the same route. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier finite energy carrier`

Theorem 440.24 (Sobolev weak-derivative scope)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet. Every weak-derivative observation of S is scoped by the finite path*

$$D, M, V, L, G, H, C, P, N.$$

The derivative row G is readable only after the domain, base, codomain, and integral magnitude rows D, M, V, L have been displayed, and the resulting observation is preserved only by the componentwise hsame *row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative scope`

Proof. Start from the weak-derivative ledger in the carrier definition. The row G is not a free-standing predicate; it is the derivative-facing coordinate attached to the already displayed rows D, M, V, L.

Use the classifier and transport rows. Sameness of accepted Sobolev packets is componentwise through H, and replay of a weak-derivative read is through C, so the observation cannot leave the same carrier route.

Finally apply the finite-window root package. It packages Real, metric, norm or Hilbert, integral, derivative, transport, replay, provenance, and naming rows as one finite BHist surface, which proves the stated scope by coordinate exhaustion. □

Theorem 440.25 (Sobolev complete-metric handoff)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* Sobolev$^{\uparrow}$ *packet. Any* CompleteMetric$^{\uparrow}$, Met$^{\uparrow}$, Hilbert$^{\uparrow}$, *PDE, variational, integral, or compactness-facing consumer receives the Sobolev data only through the finite handoff*

$$D, M, V, L, G, H, C, P, N.$$

The handoff may read complete-metric boundary information through M and Hilbert or norm information through V, but it does not export a completed Sobolev carrier, selected metric limit, quotient function class, distributional dual, density theorem, compact embedding, or trace theorem.

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier complete metric handoff`

Proof. Begin with the finite energy carrier. It fixes $D, M, V, L, G, H, C, P, N$ as the only readable rows and assigns metric or complete-metric data to M and norm or Hilbert data to V.

Next apply the weak-derivative scope. The integral and derivative rows L and G remain downstream of the same domain, base, and codomain rows, and their observations are preserved only by H, C, P, N.

The downstream handoff is therefore just a repackaging of the finite root package. Since no coordinate names a completed Sobolev carrier, selected limit, quotient class, distributional dual, density theorem, compact embedding, or trace theorem, none of those objects can be supplied to the listed consumers. □

Definition 440.26 (Sobolev finite-window carrier)**.** *A finite-window* Sobolev$^{\uparrow}$ *carrier is a displayed* BHist *packet*

$$W = (D, M, V, L, G, T, H, C, P, N)$$

over the following rows:

1. *D is the* Real$^{\uparrow}$ *domain row for the finite scalar observation window.*
2. *M is the* Met$^{\uparrow}$ *or* CompleteMetric$^{\uparrow}$ *base row attached to the same window.*
3. *V is the* Norm$^{\uparrow}$ *or* Hilbert$^{\uparrow}$ *codomain row through which function and weak-gradient magnitudes are read.*
4. *L is the* Integral$^{\uparrow}$ *magnitude row for the displayed function and weak-gradient observations.*
5. *G is the* Derivative$^{\uparrow}$ *weak-derivative ledger row, containing only the finitely inspected test routes and their responses.*
6. *T is the transport ledger for weak-derivative rows, recording how each displayed test route is moved under the local classifier.*
7. *H records componentwise* hsame *comparison for D, M, V, L, G, T, P, N.*

8. *C records displayed* Cont *replay from a Sobolev read through the finite window into $D, M, V, L, G, T, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *row for this finite window.*

The finite-window classifier compares accepted packets only through these rows. It does not compare ambient Hilbert completions, quotient function classes, measure-theoretic representatives, host equality of functions, or selected extensions beyond the displayed weak-derivative tests.

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevFiniteWindowCarrier field faithful`

Theorem 440.27 (Sobolev completion-boundary nonescape)**.** *Every completion-facing read of a finite-window* $\mathsf{Sobolev}^{\uparrow}$ *carrier $W = (D, M, V, L, G, T, H, C, P, N)$ is confined to*

$$D, M, V, L, G, T, H, C, P, N.$$

In particular, a consumer may read $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, *weak-derivative transport, replay, provenance, and local naming only through the displayed finite-window rows. It cannot export ambient Hilbert completion, quotient functions, measure-theoretic choice, host equality outside the displayed tests, density of smooth functions, compact embedding, trace theory, or a selected representative.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevFiniteWindowCarrier completion boundary nonescape`

Proof. Project the finite-window carrier of Definition 440.26. Its analytic rows are exactly D, M, V, L, G, T, and its structural rows are exactly H, C, P, N.

Read a completion-facing request through the replay row C. The route first reaches the $\mathsf{Real}^{\uparrow}$ domain row and metric or complete-metric base row, then the norm or Hilbert codomain row, and only then the integral magnitude and derivative test rows.

The transport ledger T and componentwise hsame row H move only those displayed weak-derivative rows. They do not add a quotient class, a measure-completion selector, or equality of host functions.

Finally P and N record provenance and local naming for the same finite window. Since the excluded analytic objects are absent from the carrier inventory and from the replay route, no completion-facing consumer can export them from this packet. □

Theorem 440.28 (Sobolev Hilbert completion descent)**.** *Any* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *PDE, variational, or completion-facing consumer of a* $\mathsf{Sobolev}^{\uparrow}$ *packet descends to the carrier rows of Definition 440.19 before it may cite weak-derivative data. The descent route passes through the displayed* $\mathsf{Real}^{\uparrow}$ *domain row, the metric or complete-metric base row, the codomain norm or Hilbert row, the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. It supplies no completed Sobolev space, distributional dual, quotient function equality, density theorem, compact embedding, trace row, or selected representative.*

Proof. Start with a Hilbert-facing or completion-facing consumer. The BHist carrier scope theorem forces the read to land on the displayed finite inventory $D, M, V, L, G, H, C, P, N$.

Use the derivative norm ledger. It identifies the analytic handoff as a route from the codomain row V through the integral magnitude row L and derivative row G, with Hilbert information restricted to the finite norm-facing surface in V.

The completion boundary then applies to the same route. The structural rows record transport, replay, provenance, and local naming only, so no completed Sobolev space, distributional dual, quotient function equality, density, compactness, trace, or selected representative can be read by descent. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier hilbert completion descent`

Theorem 440.29 (Sobolev weak-gradient ledger exhaustion)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet. Every weak-gradient read exported by S is exhausted by the displayed rows D, M, V, L, G together with the structural rows H, C, P, N. In particular, the read reaches* $\mathsf{Real}^{\uparrow}$ *only through D,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *only through M,* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *only through V,* $\mathsf{Integral}^{\uparrow}$ *only through L,* $\mathsf{Derivative}^{\uparrow}$ *only through G, and* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ *only through the carrier and structural rows already displayed by S.*

Proof. Project the accepted carrier. The weak-gradient route has no analytic coordinate beyond D, M, V, L, G, and its structural support is exactly H, C, P, N.

Apply the Sobolev weak-derivative ledger theorem. It forces each derivative-facing observation to pass through the domain, metric or complete-metric base, codomain, magnitude ledger, and weak-gradient test row before the observation is exported.

The BHist carrier scope theorem then accounts for the structural rows. The hsame transports, Cont replay, package provenance, and local NameCert evidence preserve the displayed weak-gradient route without adding an ambient analytic object. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak gradient ledger exhaustion`

Theorem 440.30 (Sobolev integral-derivative transport stability)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ and $S' = (D', M', V', L', G', H', C', P', N')$ be accepted* $\mathsf{Sobolev}^{\uparrow}$ *packets compared by the displayed componentwise* hsame *transport rows. Any consumer that transports an integral or derivative read from S to S' must transport the same route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G$$

to

$$D' \longmapsto M' \longmapsto V' \longmapsto L' \longmapsto G'$$

through the displayed Cont *replay rows, with provenance and local naming preserved by P, N and P', N'. The transport cannot introduce a host distribution, quotient function equality, completed measure carrier, compact embedding row, or trace row.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier integral derivative transport stability`

Proof. Begin with classifier locality. The classifier compares accepted packets only through the displayed carrier rows and the hsame transport surface.

Use carrier transport to move the route. The source read through D, M, V, L, G is replaced componentwise by the target read through D', M', V', L', G', and the replay rows C, C' preserve the same consumer order.

Finally apply weak-gradient ledger exhaustion. Since the source and target packets expose only the listed Real, metric, complete-metric, norm, Hilbert, integral, derivative, BHist, transport, replay, provenance, and NameCert rows, no excluded analytic object can appear during transport. □

Theorem 440.31 (Sobolev completion-facing root scope)**.** *Every completion-facing, PDE-facing, variational, compact-embedding, Hilbert, metric, or derivative consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet must first read the finite root scope*

$$D, M, V, L, G, H, C, P, N.$$

Thus completion-facing access remains confined to $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *as displayed carrier rows. It does not provide a completed Sobolev space, distributional dual, quotient equality of functions, density theorem, compact embedding theorem, trace theorem, or selected representative.*

Proof. Start with Sobolev Hilbert completion descent. It sends every Hilbert-facing or completion-facing request to the carrier rows before weak-derivative data are cited.

Apply weak-gradient ledger exhaustion to identify the analytic portion of the root scope. The rows D, M, V, L, G are the only path to Real, metric, complete-metric, norm, Hilbert, integral, and derivative reads.

Apply integral-derivative transport stability for transported consumers. The rows H, C, P, N preserve, replay, source, and name the same finite root scope, so downstream PDE, variational, compactness-facing, and completion-facing consumers cannot read outside it. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier completion facing root scope`

Theorem 440.32 (Sobolev trace-window obligation)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet exposes any boundary-facing trace-window read only through finite* BHist *rows and the displayed* hsame *transport surface. A consumer may read a trace window after it has traversed the carrier rows* $D, M, V, L, G, H, C, P, N$*, the weak-derivative ledger, the derivative norm ledger, and the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *evidence. The obligation supplies a finite boundary-window route, not a global trace theorem, selected representative, quotient function equality, or ambient measure-completion choice.*

Proof. Unpack the carrier of Definition 440.1. The only readable coordinates are the displayed domain, base, codomain, integral magnitude, weak-gradient, transport, replay, provenance, and local naming rows.

Use the weak-derivative ledger and derivative norm ledger. They force boundary-facing analytic data to pass through the same D, M, V, L, G route before the structural rows H, C, P, N can replay the observation.

Finally apply the completion-facing root scope. Since the root scope excludes completed Sobolev spaces, quotient equality, selected representatives, and ambient measure choice, a trace-window consumer receives only the finite BHist rows and their hsame transports. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier trace window obligation`

Theorem 440.33 (Sobolev local Poincare ledger)**.** *Every local derivative or integral window read from an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet is routed through displayed finite Poincare-style comparison ledgers. The route compares the* $\mathsf{Derivative}^{\uparrow}$ *row* G *against the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L *over the fixed domain, metric, and codomain rows* D, M, V*, then transports the comparison through* H *and replays it through* C*. It does not invoke an ambient measure selection, global representative, or unrecorded normed-space completion.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier local poincare ledger`

Proof. Begin with the derivative norm ledger. It places derivative magnitude and norm control inside the route from V through L to G, so every local comparison has a displayed analytic source.

Read the same route through the finite-window root package. The package exposes D, M, V, L, G before the structural rows, so the comparison is a finite ledger over the displayed domain, base, codomain, integral, and derivative data.

Transport now uses only H and C. These rows preserve and replay the displayed local comparison; they contain no coordinate for choosing an ambient measure carrier, selecting a representative, or completing a normed space. □

Theorem 440.34 (Sobolev completion consumer export)**.** *Every downstream* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *PDE, variational, or compactness-facing consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet reads only the Sobolev carrier, the weak-derivative ledger, the derivative norm ledger, the integral and norm rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The export surface contains no additional distributional dual, quotient-function equality, density theorem, compact embedding theorem, trace theorem, or completed Sobolev carrier.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier completion consumer export`

Proof. Start with Sobolev Hilbert completion descent. It sends each completion-facing consumer to the carrier rows before any weak-derivative datum can be cited.

Apply weak-gradient ledger exhaustion and integral-derivative transport stability. These theorems identify the analytic payload as the finite route through D, M, V, L, G and preserve it across H, C, P, N.

Use the trace-window obligation and local Poincare ledger for boundary-facing and local comparison reads. Both obligations remain inside the same carrier, transport, replay, provenance, and local NameCert rows, so the completion consumer export has no additional analytic coordinate. □

Theorem 440.35 (Sobolev Real completion root window)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet carries a Real-completion root window in its finite* BHist *surface. The* $\mathsf{Real}^{\uparrow}$ *row D supplies the scalar observation and terminal seal rows, the* $\mathsf{Integral}^{\uparrow}$ *row L and* $\mathsf{Derivative}^{\uparrow}$ *row G supply the weak-gradient magnitude ledger, and the structural rows H, C, P, N transport, replay, source, and name the same route. Thus a Real-facing completion consumer receives scalar and limit-facing rows only through the displayed carrier, not through ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of functions, selected representatives, or choice.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier real completion root window`

Proof. Start with the Sobolev completion consumer export. It sends every completion-facing request to the carrier rows before any analytic datum is read.

Apply the weak-derivative ledger. The scalar route begins at the $\mathsf{Real}^{\uparrow}$ domain row D and reaches the weak-gradient magnitude rows only through L and G.

Finally use the completion boundary. Since H, C, P, N merely preserve, replay, source, and name the same finite route, the Real-completion root window exposes no ambient Real completeness, quotient equality, selected representative, or choice coordinate. □

Theorem 440.36 (Sobolev metric completion root window)**.** *Every metric-facing or complete-metric-facing consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet must read distance and Cauchy-boundary information through the displayed base row M together with the carrier route $D, M, V, L, G, H, C, P, N$. The* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *surface supplies only the finite distance and Cauchy rows needed by the Sobolev packet, and the route exports no completed Sobolev carrier, function quotient, selected Cauchy limit, or unrecorded metric completion.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier metric completion root window`

Proof. Begin with the finite root scope. It records that completion-facing access is confined to the displayed rows $D, M, V, L, G, H, C, P, N$.

Read the metric base row M. This row is the only coordinate for $\mathsf{Met}^{\uparrow}$ or $\mathsf{CompleteMetric}^{\uparrow}$ distance and Cauchy information, so a consumer cannot obtain a separate completed function-space object.

Use the completion consumer export to repack the route. The rows H, C, P, N transport, replay, source, and name the same metric window, while the excluded quotient, selected-limit, and unrecorded-completion objects remain absent from the carrier. □

Theorem 440.37 (Sobolev Hilbert-norm root window)**.** *Every Hilbert-facing or norm-facing consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet reads the weak-gradient ledger through the codomain row* V*, the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L*, and the* $\mathsf{Derivative}^{\uparrow}$ *test row* G*. If* V *is displayed as a* $\mathsf{Hilbert}^{\uparrow}$ *row, the Hilbert structure is still read only through the same finite norm and weak-gradient ledgers; if* V *is displayed as a* $\mathsf{Norm}^{\uparrow}$ *row, no hidden inner-product or completed Hilbert carrier is supplied. In both cases the route is confined to* $D, M, V, L, G, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier hilbert norm root window`

Proof. Start with Sobolev Hilbert completion descent. It requires Hilbert-facing consumers to read the carrier rows before citing weak-derivative data.

Apply weak-gradient ledger exhaustion. The codomain row V, integral magnitude row L, and derivative row G are the only ledger rows that expose norm, Hilbert, integral, and derivative information.

Finally use integral-derivative transport stability. Transport across H and replay through C preserve the same weak-gradient ledger rows, while P and N only source and name them. No hidden inner-product object, completed Hilbert carrier, or quotient function class is introduced. □

Theorem 440.38 (Sobolev carrier energy window)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$ *exposes one finite energy window. The window consists of the* $\mathsf{Real}^{\uparrow}$ *domain row* D*, the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row* M*, the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row* V*, the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L*, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row* G*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *evidence* N*. No consumer can split this window into a separate measure-completion object, quotient function class, distributional dual, selected representative, or completed Sobolev carrier.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier energy window`

Proof. Project the carrier of Definition 440.1. Its visible coordinates are exactly $D, M, V, L, G, H, C, P, N$, with the domain, base, codomain, magnitude, weak-gradient, transport, replay, provenance, and naming roles fixed by the carrier definition.

Apply the NameCert obligation surface. It requires carrier habitation, classifier stability, carrier transport, weak-derivative exactness, and completion-boundary non-escape over the same finite rows.

Finally use the completion consumer export and Hilbert-norm root window. They force downstream completion, PDE, variational, Hilbert-facing, and norm-facing consumers to read the displayed energy rows together, so no omitted analytic object can be recovered from the packet. □

Theorem 440.39 (Sobolev weak-derivative route)**.** *Every weak-derivative-facing consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet must read the* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L *before it reads the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient response row* G*. The route is the finite carrier path*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G$$

transported by H *and replayed by* C*; it does not read distribution theory, quotient equality of functions, ambient almost-everywhere classes, or a host weak-derivative predicate.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak derivative route`

Proof. Begin with the weak-derivative ledger theorem. It orders every derivative-facing read through the domain, base, codomain, magnitude, and weak-gradient rows before the observation is exported.

The magnitude row L is therefore not optional. It is the only displayed $\mathsf{Integral}^{\uparrow}$ ledger available for the weak-gradient response, and the derivative row G is read only after that magnitude ledger is fixed.

Transport and replay add no analytic coordinates. The rows H and C preserve the same route, while P and N source and name it, so distribution theory, quotient equality, almost-everywhere classes, and host predicates remain outside the packet. □

Theorem 440.40 (Sobolev completion-norm handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet receives only the displayed Hilbert or norm row* V *together with the finite weak-gradient ledger* $D, M, V, L, G, H, C, P, N$*. If the base row* M *is read as* $\mathsf{CompleteMetric}^{\uparrow}$ *data, that read supplies only the displayed metric or Cauchy-boundary information needed by the packet; it does not create a completed Sobolev space, a quotient normed function space, a selected Cauchy limit, or a completed Lebesgue carrier.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier completion norm handoff`

Proof. Start with the metric completion root window. It confines metric-facing and complete-metric-facing reads to the displayed base row M together with the carrier route $D, M, V, L, G, H, C, P, N$.

Apply the Hilbert-norm root window. Hilbert-facing and norm-facing consumers read V only through the same weak-gradient ledger, with L and G providing the integral magnitude and derivative response rows.

Use the completion consumer export to repack the handoff. The exported surface is the finite norm and weak-gradient ledger plus structural transport, replay, provenance, and naming rows; no completed Sobolev space, quotient function carrier, selected limit, or completed Lebesgue object is a coordinate of the packet. □

Theorem 440.41 (Sobolev root-unblock package)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$ *exports a root-unblock package consisting of the* $\mathsf{Real}^{\uparrow}$ *domain surface* D*, the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row* M*, the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row* V*, the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger* L*, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient ledger* G*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *evidence* N*. Downstream analysis chapters may cite this package only through the displayed carrier, weak-derivative, completion-boundary, Hilbert-norm, and root-window routes already exposed in the chapter.*

The package supplies no host Sobolev space, distributional dual, quotient function class, completed Lebesgue carrier, selected representative, compact embedding theorem, trace theorem, or density theorem.

Proof. Project the carrier of Definition 440.1. The visible inventory is exactly $D, M, V, L, G, H, C, P, N$, and each row has the analytic or structural role named in the statement.

Apply the local NameCert obligation surface, the weak-derivative ledger, and the root obligation package. These results give carrier admission, classifier stability, weak-gradient ledger exactness, completion-boundary nonescape, and $\mathsf{Real}^{\uparrow}$ seal nonescape over the same finite rows.

Use the metric completion root window, Hilbert-norm root window, and completion-norm handoff to expose the downstream route. Metric, complete-metric, Hilbert, norm, integral, derivative, PDE, variational, and compactness-facing consumers must traverse the displayed carrier route before reading Sobolev data.

The excluded analytic objects are absent from the carrier and from the cited routes. Transport, replay, provenance, and local naming preserve the package but do not add new analytic coordinates. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root unblock package`

Theorem 440.42 (Sobolev finite-energy NameCert obligation)**.** *For every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$*, the finite-energy obligation surface exposes exactly the derivative row* G*, the integral magnitude row* L*, the norm or Hilbert row* V*, the complete-metric or metric base row* M*, and the terminal* $\mathsf{Real}^{\uparrow}$ *domain seal* D*, together with componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *evidence* N*. A finite-energy consumer may cite this surface only as one* BHist *packet; it does not receive a host weak-gradient predicate, completed Sobolev carrier, Lebesgue-completed measure space, quotient function equality, selected representative, compact embedding, or trace theorem.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier finite energy namecert obligation`

Proof. Project the carrier of Definition 440.1. Its finite-energy coordinates are the $\mathsf{Real}^{\uparrow}$ domain row D, metric or complete-metric row M, norm or Hilbert row V, integral magnitude row L, derivative row G, and structural rows H, C, P, N.

Apply the finite-energy carrier and weak-derivative ledger. These force energy reads to pass through the integral magnitude row before the derivative row is interpreted, with the norm or Hilbert row and metric base row already displayed.

Use the completion boundary and root-unblock package to keep the read inside the same NameCert surface. Transport, replay, provenance, and local naming preserve the finite packet, while completed Sobolev spaces, quotient functions, selected representatives, compact embeddings, trace rows, and host measure completions remain absent coordinates. □

Theorem 440.43 (Sobolev root weak-gradient carrier)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$ *exposes weak-gradient data only through the finite* BHist *route*

$$D, M, V, L, G, H, C, P, N.$$

Here D *is the* $\mathsf{Real}^{\uparrow}$ *domain row,* M *is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *base row,* V *is the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row,* L *is the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger,* G *is the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient ledger,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is local* $\mathsf{NameCert}$ *evidence. No root weak-gradient read may pass through distributional duals, Lebesgue completion, quotient function equality, compact embedding, or trace theory.*

Proof. Project the carrier rows. The accepted packet exposes exactly $D, M, V, L, G, H, C, P, N$, so every weak-gradient read begins with the displayed $\mathsf{Real}^{\uparrow}$, metric or complete-metric, norm or Hilbert, integral, and derivative coordinates.

Route the weak-gradient request through L and G. The integral ledger supplies the magnitude surface before the derivative row is interpreted, and the derivative row records only the displayed weak-gradient response.

The structural rows keep the read inside the same packet. Componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence preserve the route without adding distributional duals, Lebesgue completion, quotient equality, compact embedding, or trace data. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root weak gradient carrier`

Theorem 440.44 (Sobolev root norm-completion ledger)**.** *For every accepted* Sobolev$^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$*, any completion-facing or norm-facing root consumer must read the displayed ledger*

$$D, M, V, L, G, H, C, P, N$$

as one BHist *packet over* Real$^{\uparrow}$*,* Met$^{\uparrow}$ *or* CompleteMetric$^{\uparrow}$*,* Norm$^{\uparrow}$ *or* Hilbert$^{\uparrow}$*,* Integral$^{\uparrow}$*,* Derivative$^{\uparrow}$*, componentwise* hsame*, displayed* Cont*,* Pkg*, and* NameCert*. The ledger does not create a completed Sobolev space, a Lebesgue-completed measure carrier, quotient equality of functions, a selected representative, a distributional dual, compact embedding, or trace theory.*

Proof. Begin with the root-unblock package. It already presents $D, M, V, L, G, H, C, P, N$ as the downstream analysis surface, with M carrying the metric or complete-metric boundary and V carrying the norm or Hilbert boundary.

Route norm and completion reads through the weak-gradient ledgers. The consumer must pass through the integral row L and derivative row G before interpreting Sobolev data as energy or weak-gradient information.

Finally replay and source the same ledger through H, C, P, N. These rows preserve the finite packet and refuse completed Sobolev spaces, Lebesgue completion, quotient equality, selected representatives, distributional duals, compact embeddings, and traces as absent coordinates. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root norm completion ledger`

Theorem 440.45 (Sobolev root trace nonescape)**.** *Let* $S = (D, M, V, L, G, H, C, P, N)$ *be an accepted* Sobolev$^{\uparrow}$ *packet. Every boundary-facing, PDE-facing, or variational read that cites* Real$^{\uparrow}$*,* Met$^{\uparrow}$ *or* CompleteMetric$^{\uparrow}$*,* Norm$^{\uparrow}$ *or* Hilbert$^{\uparrow}$*,* Integral$^{\uparrow}$*, and* Derivative$^{\uparrow}$ *data from* S *is confined to the displayed* BHist *rows*

$$D, M, V, L, G, H, C, P, N$$

with componentwise hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. No trace operator, boundary value theorem, distributional dual, Lebesgue completion, quotient function equality, compact embedding, or selected representative escapes this finite route.*

Proof. Project the same carrier rows used by the root-unblock package. Boundary-facing consumers receive only the domain row D, base row M, codomain row V, integral row L, derivative row G, and structural rows H, C, P, N.

Route any claimed trace read through L and G. The packet permits an integral and weak-gradient ledger read, but it contains no separate boundary-value or trace coordinate.

Use transport, replay, provenance, and naming to close the route. Since H, C, P, N preserve only the displayed packet, distributional duals, Lebesgue completion, quotient equality, compact embedding, selected representatives, and trace theory cannot be exported by a Sobolev$^{\uparrow}$ citation. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier root trace nonescape`

Theorem 440.46 (Sobolev finite-energy induction route)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* $\mathsf{Sobolev}^{\uparrow}$ *finite-energy packet. Every accepted finite-energy consumer derived from S is obtained by induction over the displayed rows $D, M, V, L, G, H, C, P, N$: the domain, base, and codomain rows D, M, V are read first, the integral and weak-gradient rows L, G are read next, and the structural rows H, C, P, N then provide componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *evidence. The induction has no cases for distributional duals, Lebesgue completion, quotient function equality, selected representatives, compact embedding, or trace operators.*

Proof. Project the carrier of Definition 440.1. The projected row list is finite, ordered as $D, M, V, L, G, H, C, P, N$, and contains no host analytic coordinate beyond those displayed rows.

Induct over that finite list. The initial segment reads the $\mathsf{Real}^{\uparrow}$ domain row, the metric or complete-metric base row, and the norm or Hilbert codomain row. The next segment reads the $\mathsf{Integral}^{\uparrow}$ magnitude row and the $\mathsf{Derivative}^{\uparrow}$ weak-gradient row.

Use Definition 440.42, Definition 440.44, and Definition 440.45 to close the consumer-facing cases. These theorems identify the finite-energy surface, keep norm and completion reads inside the same packet, and block trace-facing escape.

Finally repack the induction result through H, C, P, N. Componentwise transport, replay, provenance, and local naming turn the row induction back into a $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ obligation, while distributional duals, Lebesgue completion, quotient function equality, selected representatives, compact embedding, and trace operators remain absent rather than additional induction cases. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevFiniteEnergyInductionRoute`

Theorem 440.47 (Sobolev test-function source)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *test-function read must expose the finite source route*

$$D, M, V, L, G, H, C, P, N$$

before any weak-gradient test is public. The row D supplies the $\mathsf{Real}^{\uparrow}$ *domain surface, M supplies the metric or complete-metric base, V supplies the norm or Hilbert codomain, L supplies the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger, G supplies the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient test row, and H, C, P, N transport, replay, source, and name the same packet. No test-function consumer may cite a host dense subspace, a quotient function representative, a distributional dual, or a selected approximating sequence outside these displayed rows.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier test function source`

Proof. Project the carrier of Definition 440.1. Its visible coordinates are exactly $D, M, V, L, G, H, C, P, N$, so a test-function read has no separate coordinate beyond the same finite Sobolev packet.

Apply the weak-derivative route and the integral-derivative source package. These place L before G and make the weak-gradient test public only after the integral magnitude row and derivative response row have both been displayed.

Finally use the root-unblock package. It repacks the carrier, weak-derivative, completion-boundary, Hilbert-norm, and root-window routes into one downstream surface, while the excluded host dense subspace, quotient representative, distributional dual, and selected approximation data remain absent from the carrier. □

Theorem 440.48 (Sobolev weak-gradient transport)**.** *If two accepted* $\mathsf{Sobolev}^{\uparrow}$ *packets have* hsame*-related carrier rows, then transport preserves the displayed weak-gradient ledger through the* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *rows. Concretely, the transported read of* $D, M, V, L, G, H, C, P, N$ *carries the same integral magnitude row* L*, the same weak-gradient response row* G*, and the same structural replay through* H, C, P, N*; it does not create host equality of functions, an ambient almost-everywhere quotient, a completed Lebesgue carrier, or an external weak-gradient predicate.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak gradient transport`

Proof. Start with the carrier energy window. The rows $D, M, V, L, G, H, C, P, N$ form one energy packet, and L and G are the only integral and derivative ledgers available for weak-gradient data.

Use componentwise hsame transport through H. Since H relates the displayed coordinates rather than host functions, the transported packet preserves the $\mathsf{Integral}^{\uparrow}$ magnitude row and the $\mathsf{Derivative}^{\uparrow}$ weak-gradient response row as carrier rows.

Replay through C and keep the provenance and naming rows P, N attached to the same route. No step adds host equality of functions, almost-everywhere quotient data, completed Lebesgue measure, or an external weak-gradient predicate. □

Theorem 440.49 (Sobolev PDE consumer admission)**.** *Every PDE-facing or variational consumer may cite* $\mathsf{Sobolev}^{\uparrow}$ *only through the finite rows*

$$D, M, V, L, G, H, C, P, N.$$

The citation must display the test-function source route of Definition 440.47 and the weak-gradient transport route of Definition 440.48. It supplies no host Sobolev space, quotient function class, Lebesgue completion, compact embedding theorem, trace theory, density theorem, selected representative, or undisplayed weak-gradient oracle.

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier pde consumer admission`

Proof. Begin with the root-unblock package. It is already the downstream analysis surface for Sobolev data and contains exactly the carrier rows $D, M, V, L, G, H, C, P, N$.

Add the test-function source theorem. A PDE or variational read cannot request weak-gradient tests before the domain, metric, codomain, integral, derivative, transport, replay, provenance, and naming rows have all been displayed.

Add the weak-gradient transport theorem. It ensures that any transported consumer still reads the same L and G ledgers through H, C, P, N, rather than through host equality, quotient functions, completed measure theory, or an external weak-gradient predicate.

The remaining exclusions follow by coordinate exhaustion. Compact embeddings, trace theory, density theorems, selected representatives, and host Sobolev spaces are not rows of the accepted packet, so they cannot be exported by a $\mathsf{Sobolev}^{\uparrow}$ citation. □

Theorem 440.50 (Sobolev weak-gradient energy carrier)**.** *For every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet* $S = (D, M, V, L, G, H, C, P, N)$*, the weak-gradient energy carrier is the finite route*

$$D \longmapsto M \longmapsto V \longmapsto L \longmapsto G \longmapsto H, C, P, N.$$

The $\mathsf{Real}^{\uparrow}$ *domain row* D*, metric or complete-metric row* M*, norm or Hilbert row* V*,* $\mathsf{Integral}^{\uparrow}$ *magnitude row* L*, and* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row* G *must be read together before any downstream PDE, variational, Hilbert, norm, integral, derivative, or completion consumer receives energy data.*

No energy read can be discharged by a host weak-gradient predicate, quotient function class, distributional dual, completed measure carrier, selected representative, compact embedding, trace row, or density theorem.

Proof. Start with the carrier energy window. It identifies $D, M, V, L, G, H, C, P, N$ as one finite energy window and rules out splitting it into a separate measure-completion object or quotient function class.

Apply the weak-derivative route and integral-derivative source package. They place the $\mathsf{Integral}^{\uparrow}$ magnitude row L immediately before the $\mathsf{Derivative}^{\uparrow}$ weak-gradient row G and make both rows the displayed source package for derivative-facing consumers.

Use the derivative norm ledger and Hilbert-norm root window. These keep norm and Hilbert-facing reads inside the codomain row V before the integral and derivative rows are interpreted.

Finally replay the route through H, C, P, N. Those rows transport, replay, source, and name the same weak-gradient energy carrier, while the excluded host objects remain outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier weak gradient energy carrier`

Theorem 440.51 (Sobolev completion-metric handoff)**.** *Every completion-facing, metric-facing, Hilbert-facing, PDE-facing, or variational consumer of an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet receives Sobolev data through the displayed handoff*

$$D, M, V, L, G, H, C, P, N.$$

The base row M is the only $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary read, the codomain row V is the only* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *magnitude read, and L, G are the only integral and derivative ledgers exported to the consumer. The handoff does not create a completed Sobolev carrier, selected Cauchy limit, quotient function class, distributional dual, completed Lebesgue carrier, compact embedding object, trace row, or density theorem.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier completion metric handoff`

Proof. Begin with the complete-metric handoff and completion-norm handoff. They force completion-facing consumers to read the base row M and codomain row V only as coordinates of the same finite Sobolev packet.

Apply the weak-gradient energy carrier. Integral and derivative data are then read through L and G downstream of D, M, V, with H, C, P, N preserving the route.

Use the completion boundary and completion-boundary nonescape results. These results block completed Sobolev spaces, quotient function carriers, selected limits, completed measure carriers, distributional duals, compact embeddings, traces, and density statements from entering the handoff.

Thus the completion-metric consumer receives exactly the displayed BHist route and its structural transport, replay, provenance, and naming rows. □

Theorem 440.52 (Sobolev source-triple handoff)**.** *Every accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet $S = (D, M, V, L, G, H, C, P, N)$ routes its weak-gradient consumer through the displayed source triple*

$$D \longmapsto M \longmapsto V$$

before the $\mathsf{Integral}^{\uparrow}$ *magnitude row L and the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row G are available. Here D is the* $\mathsf{Real}^{\uparrow}$ *domain row, M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *base row, and V is the* $\mathsf{Hilbert}^{\uparrow}$ *or* $\mathsf{Norm}^{\uparrow}$ *codomain row. The handoff supplies no distribution theory, Lebesgue completion, density of smooth functions, compact embedding, trace theorem, or completed Sobolev space.*

Proof. Project the carrier of Definition 440.1. Its analytic source rows are exactly D, M, V, L, G, with H, C, P, N serving only transport, replay, provenance, and local naming.

Apply the weak-derivative ledger, completion-boundary nonescape, and Hilbert-norm root window. These results order the read from the Real domain row through the metric or complete-metric base and the norm or Hilbert codomain before the integral magnitude and derivative weak-gradient rows are interpreted.

Finally use the root-unblock package and the completion-metric handoff. They package the same $D, M, V, L, G, H, C, P, N$ route for downstream completion-facing and weak-gradient consumers, while the excluded distributional, measure-completion, density, compactness, trace, and completed-space objects remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier source triple handoff`

Theorem 440.53 (Sobolev trace-boundary obligation)**.** *Every trace-facing, PDE-facing, variational, or boundary-value consumer of an accepted* Sobolev$^{\uparrow}$ *packet must first read the displayed* Real$^{\uparrow}$ *domain row, metric or complete-metric base row, norm or Hilbert codomain row,* Integral$^{\uparrow}$ *magnitude ledger,* Derivative$^{\uparrow}$ *weak-gradient row,* BHist *carrier rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The packet does not supply a trace theorem, boundary restriction operator, quotient function equality, density theorem, compact embedding, selected representative, or completed Sobolev space.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier trace boundary obligation`

Proof. Project the carrier of Definition 440.1. A boundary-facing consumer sees only the rows $D, M, V, L, G, H, C, P, N$, with D carrying the Real domain, M carrying the metric or complete-metric base, V carrying the norm or Hilbert codomain, and L, G carrying the integral and weak-gradient ledgers.

Apply the weak-derivative ledger, completion-boundary, classifier-locality, and integral-derivative handoff theorems. These results order every derivative-facing read through D, M, V, L, G and keep sameness, transport, and replay local to the displayed carrier rows.

Finally repack the route through H, C, P, N. Those rows preserve, replay, source, and name the same Sobolev packet; they do not add a trace theorem, boundary restriction operator, quotient function equality, density theorem, compact embedding, selected representative, or completed Sobolev space. Coordinate exhaustion gives the stated boundary obligation. □

Theorem 440.54 (Sobolev finite-energy obligation scope)**.** *Every finite-energy consumer of an accepted* Sobolev$^{\uparrow}$ *packet must expose the displayed* Real$^{\uparrow}$ *domain row D, the* Met$^{\uparrow}$ *or* CompleteMetric$^{\uparrow}$ *source row M, the* Norm$^{\uparrow}$ *or* Hilbert$^{\uparrow}$ *codomain row V, the* Integral$^{\uparrow}$ *magnitude ledger L, the weak* Derivative$^{\uparrow}$ *test row G, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before it reads Sobolev energy data. The scope refuses distribution theory, Lebesgue completion, quotient equality of functions, density of smooth functions, compact embedding, and trace theory.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevFiniteEnergyObligationScope`

Proof. Project the carrier of Definition 440.1. The finite-energy read has exactly the analytic rows D, M, V, L, G and the structural rows H, C, P, N.

Use the obligation package and weak-derivative ledger. They place the Real domain, metric or complete-metric source, norm or Hilbert codomain, integral magnitude, and derivative test rows on the same finite route before any analysis-facing consumer can cite the packet.

Finally apply the completion-boundary theorem. The refused analytic objects are not coordinates of the carrier and are not introduced by transport, replay, provenance, or local naming, so the finite-energy scope is exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier finite energy obligation scope`

Theorem 440.55 (Sobolev trace-boundary obligation route)**.** *Any boundary-facing, PDE-facing, or variational read that attempts to use* $\mathsf{Sobolev}^{\uparrow}$ *trace language must first pass through the same finite route*

$$D, M, V, L, G, H, C, P, N.$$

Here D *binds the* $\mathsf{Real}^{\uparrow}$ *domain row,* M *binds the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *source row,* V *binds the* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain row,* L *binds the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger,* G *binds the weak* $\mathsf{Derivative}^{\uparrow}$ *test row, and* H, C, P, N *bind transport, replay, provenance, and local naming. The route supplies no trace theorem, boundary restriction operator, Lebesgue completion, quotient equality of functions, density of smooth functions, compact embedding, or distribution theory.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevTraceBoundaryObligationRoute`

Proof. Start with Definition 440.53. It already confines trace-facing reads to the carrier rows and refuses trace operators, boundary restrictions, quotient function equality, density, compact embedding, and completed Sobolev spaces.

Add the finite-energy scope theorem above. It ensures that the same route contains the domain, source, codomain, magnitude, weak-derivative, transport, replay, provenance, and local naming rows before a boundary-facing consumer can read energy data.

The two routes are the same displayed packet. Therefore a trace-facing citation cannot bypass the finite-energy scope or recover distribution theory, Lebesgue completion, quotient equality, density, compact embedding, or trace theory from a hidden coordinate. □

Theorem 440.56 (Sobolev compact-embedding refusal boundary)**.** *A compact-embedding, completion-facing, PDE-facing, or functional-analysis consumer citing an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet receives only the finite-energy route through* $D, M, V, L, G, H, C, P, N$*. The route may record the* $\mathsf{Real}^{\uparrow}$ *domain,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *source,* $\mathsf{Norm}^{\uparrow}$ *or* $\mathsf{Hilbert}^{\uparrow}$ *codomain,* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger, weak* $\mathsf{Derivative}^{\uparrow}$ *tests, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. It does not supply compact embedding, density of smooth functions, trace theory, Lebesgue completion, quotient equality of functions, distribution theory, or a completed Sobolev space.*

Proof. Read the finite-energy obligation scope. It exposes the complete consumer route for energy data and leaves no separate compactness, density, trace, quotient, distributional, or completion coordinate.

Use the trace-boundary route to cover boundary-facing requests. That theorem forces even trace language through the same finite packet before any read is public.

Finally use the completion boundary of Definition 440.10. Since compact embedding and density are not coordinates of $S = (D, M, V, L, G, H, C, P, N)$ and cannot be generated by H, C, P, N, the consumer receives only the displayed finite-energy route. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier compact embedding refusal boundary`

Theorem 440.57 (Sobolev energy-trace scope obligation)**.** *Let $S = (D, M, V, L, G, H, C, P, N)$ be an accepted* $\mathsf{Sobolev}^{\uparrow}$ *packet. If a consumer exposes finite energy rows, weak derivative rows, boundary trace rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *row, then the consumer may export only the finite energy-to-trace scope route through $D, M, V, L, G, H, C, P, N$. It may not export Hahn–Banach separation, unrestricted distribution theory, measure completion, quotient function equality, ambient Hilbert or Banach completeness, or a trace operator detached from the displayed packet.*

Proof. Begin with the finite-energy scope obligations of Definition 440.54. They force every energy-facing read to display the Real domain row, metric or complete-metric source row, norm or Hilbert codomain row, integral magnitude ledger, weak derivative test row, transport, replay, provenance, and local naming before Sobolev energy data is consumed.

Next apply the trace-boundary obligation route of Definition 440.55. Trace-facing language is therefore read only along the same finite packet, with the boundary read downstream of the energy and weak-derivative rows rather than as an external boundary operator.

Finally combine the two routes. The readbacks to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ are ledger and seal rows inside the displayed package, while H, C, P, N preserve, replay, source, and name that package. Since no listed coordinate is Hahn–Banach separation, unrestricted distribution theory, measure completion, quotient function equality, or ambient Hilbert or Banach completeness, the exported scope is exactly the finite energy-to-trace route. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier energy trace boundary certificate`

Theorem 440.58 (Sobolev bridged finite-derivative window)**.** *Every Sobolev packet bridge read factors through the finite derivative-window route*

$$D, M, V, L, G, T, H, C, P, N.$$

The rows D, M, V, L, G expose the $\mathsf{Real}^{\uparrow}$ *domain, metric or complete-metric base, norm or Hilbert codomain, finite-energy magnitude ledger, and weak* $\mathsf{Derivative}^{\uparrow}$ *window; T records the trace-boundary window; and H, C, P, N provide componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any analytic consumer reads a* $\mathsf{Sobolev}^{\uparrow}$ *object. The bridge exports this finite derivative-window interface only; it does not export distribution theory, Lebesgue completion, quotient equality of functions, density of smooth functions, compact embedding, trace theory, or completed Sobolev spaces.*

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier finite derivative window bridge`

Proof. Begin with the finite-window carrier and the root package. The accepted packet exposes the domain, base, codomain, magnitude, weak-derivative, transport, replay, provenance, and local naming rows as finite BHist data.

Apply the finite-energy scope and trace-boundary routes. They force an analytic consumer to read the finite-energy rows and the boundary-facing rows through the same displayed packet, with no detached trace operator or completed function carrier.

Use the energy-trace scope obligation to align the two sides of the bridge. The derivative-window and trace-boundary windows are therefore read as one route through $D, M, V, L, G, T, H, C, P, N$ before any Sobolev object is public.

Finally repack the route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence. Since none of those structural rows adds distribution

theory, Lebesgue completion, quotient equality, density, compact embedding, trace theory, or completed Sobolev spaces, the bridge is confined to the finite derivative-window interface. □

Theorem 440.59 (Sobolev finite-energy consumer)**.** *Every downstream finite-energy consumer of a* $\mathsf{Sobolev}^{\uparrow}$ *packet must read the displayed* $\mathsf{Real}^{\uparrow}$ *domain row, a metric or complete-metric base row, a normed or Hilbert codomain row, the* $\mathsf{Integral}^{\uparrow}$ *magnitude ledger, the* $\mathsf{Derivative}^{\uparrow}$ *weak-gradient row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$ *row before it can export any energy estimate, weak-gradient comparison, or completion-facing boundary statement.*

Proof. Project the displayed Sobolev packet. The finite-energy consumer first obtains the Real-domain row and the metric or complete-metric base row, so the carrier is tied to the same ambient scale used by the weak-gradient certificate.

The codomain and magnitude data are then fixed. The norm or Hilbert row supplies the target size discipline, while the $\mathsf{Integral}^{\uparrow}$ ledger records the finite-energy magnitude read and the $\mathsf{Derivative}^{\uparrow}$ row records the weak-gradient test.

Transport and replay preserve this inventory. Componentwise hsame movement carries the same rows to an equivalent packet, and the Cont routes replay only the displayed finite-energy and weak-gradient reads.

Finally the Pkg provenance and local NameCert row name the accepted packet for downstream consumers. Since no row supplies distribution theory, Lebesgue completion, quotient function equality, trace theory, compact embedding, or completed Sobolev spaces, every finite-energy export factors through the listed local rows. □

Lean4 ✓ `BEDC.Derived.SobolevUp.SobolevCarrier mature finite energy consumer`

Closure status for $\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$.

Constructive story: A $\mathsf{Sobolev}^{\uparrow}$ packet is generated from a displayed $\mathsf{Real}^{\uparrow}$ domain row, a metric or complete-metric base row, a normed or Hilbert codomain row, an $\mathsf{Integral}^{\uparrow}$ magnitude ledger, a $\mathsf{Derivative}^{\uparrow}$ weak-gradient test row, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$, maturePackageClosed)

Scope closed: The chapter enumerates carrier habitation, classifier stability, weak-derivative ledger, finite-energy route, trace boundary, finite derivative-window bridge, completion-boundary nonescape, transport, replay, provenance, and local NameCert obligations for a concrete weak-derivative certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Sobolev}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close distribution theory, Lebesgue completion, quotient equality of functions, density of smooth functions, compact embedding, trace theory, or completed Sobolev spaces.

Upgrade path: No paper-side closure grade remains; Lean verification, bridge checking, distribution theory, Lebesgue completion, quotient function equality, trace theorem, compact embedding, and completed Sobolev spaces remain separate.

Chapter 441

A Concrete Naming Certificate for $\mathsf{AscoliModulus}^{\uparrow}$

The $\mathsf{AscoliModulus}^{\uparrow}$ interface names the finite equicontinuity-modulus packet consumed by $\mathsf{ArzelaAscoli}^{\uparrow}$. It is the constructive Bishop–Bridges finite-ledger object that binds a family of $\mathsf{ContinuousMap}^{\uparrow}$ rows over a $\mathsf{CompactMetric}^{\uparrow}$ source to one shared source-radius row for a requested $\mathsf{Rat}^{\uparrow}$ target tolerance, with finite-net stability and displayed handoff rows for compact-continuity consumers. The packet is a BEDC carrier over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, hsame, Cont, Pkg, and local NameCert rows; it is not an open-cover compactness theorem, a subsequence selector, or a host function-space compactness principle.

The source metric and finite-net row are consumed from chapter 129, and the displayed map-family rows are consumed from chapter 130 before the $\mathsf{AscoliModulus}^{\uparrow}$ packet packages the shared radius.

Definition 441.1 (AscoliModulus carrier)**.** *An* $\mathsf{AscoliModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, including its displayed metric row and the finite* ProbeBundle *net rows used by downstream compact-continuity reads;*
2. *Y is the target metric row read by the family of maps, with target-distance comparisons routed through displayed metric witnesses;*
3. *F is a finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows from X to Y;*
4. *T is the requested positive* $\mathsf{Rat}^{\uparrow}$ *target-tolerance row;*
5. *R is the shared positive* $\mathsf{Rat}^{\uparrow}$ *source-radius row selected for the whole finite family ledger;*
6. *$\mathcal{P}$ is the finite source-net* ProbeBundle *used to test the shared radius against compact source reads;*
7. *S records finite-net stability data, saying that source histories lying in one bundled radius window remain inside the modulus window demanded by every family row in F;*
8. *E records the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row whose finite tolerance ledger supplies the uniform family-side reading of T and R;*

9. *W records the* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise rows used at bundled source centers and the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *consumer handoff row;*
10. *H records componentwise* hsame *transports for source metric, target metric, family rows, tolerance rows, radius rows, probe bundles, finite-net stability rows, equicontinuity rows, uniform-continuity rows, and handoff rows;*
11. *C records the visible* Cont *routes by which compact-metric, continuous-map, uniform-continuity, equicontinuity-family, Arzela–Ascoli, rational, and probe-bundle consumers read the packet;*
12. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, hsame, *and* Cont *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AscoliModulus}^{\uparrow}}$ *row.*

The carrier exports only finite family, tolerance, radius, probe-bundle, stability, equicontinuity, uniform-continuity, transport, continuation, package, and naming rows. It contains no open-cover compactness row, no subsequence-choice row, no ambient function-space topology, and no host compactness theorem.

Theorem 441.2 (AscoliModulus equicontinuity ledger)**.** *For an accepted* $\mathsf{AscoliModulus}^{\uparrow}$ *carrier of Definition 441.1, the rows F, T, R, E, H, C, P, N determine one finite equicontinuity ledger: the requested tolerance row T is read through* $\mathsf{Rat}^{\uparrow}$*, the shared source-radius row R is read through the same rational classifier, every displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *row in F consumes that same R, and the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *coordinate E records this family-wide reading without introducing a separate modulus function.*

The ledger is stable under the stored hsame *transports and visible* Cont *routes. Any consumer reading it receives a finite tolerance-to-radius row over the displayed family ledger, not an external function assigning radii to all host tolerances.*

Lean4 ✓ `BEDC.Derived.AscoliModulusUp.AscoliModulusPacket equicontinuity ledger`

Proof. Project the tolerance row T, source-radius row R, and finite family ledger F from the carrier. The rational rows are visible $\mathsf{Rat}^{\uparrow}$ histories, and every family entry is a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row over the same source and target coordinates.

Read the $\mathsf{EquicontinuityFamily}^{\uparrow}$ coordinate E. Its finite family-side ledger binds the tolerance history to one radius history uniformly over the listed family rows, so no family entry may request an unrecorded source radius.

Finally route the ledger through H, C, P, and N. Componentwise transport preserves the same tolerance row, radius row, family ledger, and local naming boundary, so a downstream consumer can only recover the finite ledger displayed in the carrier. □

Theorem 441.3 (AscoliModulus finite-net stability)**.** *Let A be an accepted* $\mathsf{AscoliModulus}^{\uparrow}$ *carrier with finite source-net coordinate $\mathcal{P}$ and stability coordinate S. For every bundled source center in $\mathcal{P}$ and every family row in F, the stability row S routes a source history inside the displayed radius window R through the corresponding* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{UniformContinuity}^{\uparrow}$ *rows, producing a target-distance read bounded by the requested tolerance row T.*

The stability claim is finite in both indices: it ranges only over the bundled probe centers and the displayed family ledger. It does not assert a global open cover, a host-space uniform structure, or compactness of any function space.

Lean4 ✓ `BEDC.Derived.AscoliModulusUp.AscoliModulusPacket finite net stability`

Proof. Start with the compact source coordinate X and finite bundle $\mathcal{P}$. The bundle exposes only finitely many center rows, and each center read is routed through the source metric data already stored in X.

For a fixed bundled center, inspect the stability coordinate S. It points to the radius row R, the requested tolerance row T, and the listed family rows in F, so each family read is checked against the same source-radius window.

Use the pointwise rows collected in W to pass from the source metric comparison to the target metric comparison. Since S is indexed only by the finite bundle and the finite family ledger, the resulting stability certificate cannot export an infinite cover or a host function-space compactness assertion. □

Theorem 441.4 (AscoliModulus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AscoliModulus}^\uparrow}$ *surface consists exactly of five finite rows:*

1. *carrier habitation by the finite packet of Definition 441.1;*
2. *rational tolerance and radius classifier transport for T and R through* $\mathsf{Rat}^\uparrow$*;*
3. *family-wide equicontinuity ledger transport for F and E through* $\mathsf{ContinuousMap}^\uparrow$ *and* $\mathsf{EquicontinuityFamily}^\uparrow$*;*
4. *finite-net stability over $\mathcal{P}$ and S through* $\mathsf{CompactMetric}^\uparrow$*,* $\mathsf{ProbeBundle}$*, and* $\mathsf{UniformContinuity}^\uparrow$*;*
5. *Arzela–Ascoli handoff through W, H, C, P, N, forcing every downstream read to remain inside displayed* hsame*,* Cont*,* Pkg*, and local naming rows.*

Proof. Carrier habitation is immediate from Definition 441.1. The source, target, family, tolerance, radius, probe-bundle, stability, equicontinuity, uniform-continuity, handoff, transport, continuation, package, and naming rows are all visible coordinates of one finite packet.

The rational transport row handles the tolerance and radius histories. The family transport row handles the continuous-map ledger and the equicontinuity-family coordinate, preserving the fact that the same radius row is shared by every displayed family entry.

The finite-net stability row is Definition 441.3. Repacking it with the handoff coordinate W and the provenance rows H, C, P, N gives the whole local certificate surface, with no additional carrier coordinate through which open-cover compactness, subsequence choice, or host function-space compactness could enter. □

Theorem 441.5 (AscoliModulus Arzela-Ascoli handoff)**.** *Every* $\mathsf{ArzelaAscoli}^\uparrow$ *consumer reading an accepted* $\mathsf{AscoliModulus}^\uparrow$ *carrier obtains a finite equicontinuity-modulus row: the finite family ledger F, requested tolerance row T, shared source-radius row R, source-net bundle $\mathcal{P}$, stability row S, and equicontinuity row E are repacked through W, H, C, P, N as the equicontinuity modulus data named by the* $\mathsf{ArzelaAscoli}^\uparrow$ *carrier.*

The handoff supplies only the finite modulus rows needed by a constructive Arzela–Ascoli packet. It does not supply open-cover compactness, subsequence choice, compactness of a host function space, or a topology on all maps from the source to the target.

```
Lean4 ✓  BEDC.Derived.AscoliModulusUp.AscoliModulusPacket arzela ascoli handoff
```

Proof. Project the handoff coordinate W. It records exactly the $\mathsf{UniformContinuity}^\uparrow$ pointwise reads and the $\mathsf{ArzelaAscoli}^\uparrow$ consumer boundary that may read the $\mathsf{AscoliModulus}^\uparrow$ packet.

Attach the finite equicontinuity ledger of Definition 441.2 and the finite-net stability row of Definition 441.3. Together these rows provide the family ledger, requested tolerance, shared radius, bundled source centers, and target-distance stability data expected by the Arzela–Ascoli equicontinuity coordinate.

Finally route the repacked rows through H, C, P, and N. Since every exported item is already a visible carrier coordinate, the handoff gives $\mathsf{ArzelaAscoli}^{\uparrow}$ only a finite BEDC-kernel modulus packet and no ambient compactness or choice principle. □

Theorem 441.6 (AscoliModulus rational radius transport)**.** *Let $A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N)$ be an accepted $\mathsf{AscoliModulus}^{\uparrow}$ carrier. The requested tolerance row T and shared source-radius row R are transported as $\mathsf{Rat}^{\uparrow}$ rows through the displayed* BHist *data,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *row. Every consumer reading T or R receives exactly those rational rows inside the finite packet, not an open-cover radius selector, a subsequence choice, or a host function-space compactness principle.*

Lean4 ✓ `BEDC.Derived.AscoliModulusUp.AscoliModulusPacket rational radius transport`

Proof. Project the rational coordinates T and R from the carrier. Both are visible finite history rows and both are named by the same package and local certificate coordinates.

Apply Definition 441.2. The ledger binds the requested tolerance to the shared source radius uniformly over the displayed family ledger F, so the radius row cannot be replaced by an external selector.

Transport is then componentwise through H and C. The transported rows still read through P and N, and none of those coordinates contains an open cover, a subsequence selector, or a compactness theorem for a host function space. □

Theorem 441.7 (AscoliModulus family transport package)**.** *Let $A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N)$ be an accepted $\mathsf{AscoliModulus}^{\uparrow}$ carrier. The finite family ledger F, the equicontinuity row E, and the finite-net stability row S form one transported package over the displayed* BHist *rows,* hsame *comparisons,* Cont *routes,* Pkg *provenance, and local* NameCert *row. Any comparison of family entries factors through the common rational tolerance-radius pair T, R and the same probe bundle $\mathcal{P}$; it does not invoke open-cover compactness, subsequence choice, or host function-space compactness.*

Proof. Read the family ledger F, equicontinuity row E, and stability row S from the carrier. These rows live over the same compact source coordinate X, target metric coordinate Y, tolerance row T, radius row R, and probe bundle $\mathcal{P}$.

Use Definition 441.3 to check each bundled source center and each family row against the displayed radius window. Use Definition 441.6 to keep the tolerance and radius rows fixed during transport.

The package rows H, C, P, N then repack the family comparison data. Since the comparison is indexed only by the finite family ledger and finite probe bundle, there is no route to open covers, subsequence selection, or compactness of all host functions. □

Theorem 441.8 (AscoliModulus Arzela root handoff package)**.** *Every $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer reading an accepted $\mathsf{AscoliModulus}^{\uparrow}$ carrier $A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N)$ receives one finite root handoff package: the* BHist *source and target rows, finite family ledger F, rational tolerance-radius rows T, R, probe bundle $\mathcal{P}$, stability row S, equicontinuity row E, handoff row W, displayed* hsame *and* Cont *routes H, C,* Pkg *provenance P, and local* NameCert *row N. The handoff is a finite modulus package only; it is not an open-cover compactness proof, a subsequence-choice operator, or a host function-space compactness theorem.*

Proof. Start with the consumer handoff row W. It is a visible coordinate of the carrier and can read only the finite rows listed in the same package.

Attach the rational transport row from Definition 441.6 and the family transport package from Definition 441.7. Together they supply the tolerance, radius, finite family, probe-bundle, stability, and equicontinuity data needed by the Arzela–Ascoli consumer boundary.

Finally apply Definition 441.5. The resulting root handoff is exactly the finite BEDC-kernel modulus surface carried by $X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N$, so it cannot export compactness, choice, or host function-space structure beyond those rows. □

Theorem 441.9 (AscoliModulus shared-radius refinement boundary)**.** *Let $A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P,$ be an accepted* $\mathsf{AscoliModulus}^{\uparrow}$ *carrier. If the displayed finite source-net $\mathcal{P}$ is refined by additional bundled centers and the refinement rows are transported through the same H, C, P, N boundary, then the refined packet preserves the single shared source-radius row R and the requested tolerance row T for the whole displayed family ledger F.*

The refinement may add only bundled source centers and their corresponding finite-net stability reads. It does not introduce a second source-radius row, alter the requested tolerance row, change the family ledger, invoke open-cover compactness, choose a subsequence, or use compactness of a host function space.

Proof. First keep the rational rows fixed. By Definition 441.6, the requested tolerance row T and the shared source-radius row R are transported as the displayed $\mathsf{Rat}^{\uparrow}$ rows through the same BHist data, hsame comparisons, Cont routes, Pkg provenance, and local naming row.

Next transport the family surface. Apply Definition 441.7 to the finite family ledger F, the equicontinuity row E, and the finite-net stability row S. The family comparison data still factors through the common tolerance-radius pair T, R and the displayed probe-bundle boundary.

Finally repack the refined source-net reads through Definition 441.8. The additional bundled centers become extra finite rows under the same handoff boundary H, C, P, N, so the $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer still receives one finite modulus package with the original T, the original R, and the same family ledger F.

Since each step is a finite transport or handoff through displayed rows, no open-cover compactness proof, subsequence-choice operation, or host function-space compactness theorem is introduced by the refinement. □

Theorem 441.10 (AscoliModulus scoped public surface)**.** *The carrier of Definition 441.1, the equicontinuity ledger of Definition 441.2, the finite-net stability theorem of Definition 441.3, the rational-radius transport theorem of Definition 441.6, the family transport package of Definition 441.7, the Arzela root handoff package of Definition 441.8, and the shared-radius refinement boundary of Definition 441.9 assemble one scoped public surface for* $\mathsf{AscoliModulus}^{\uparrow}$*.*

Every consumer read of this surface factors through displayed $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{EquicontinuityFamily}^{\uparrow}$*,* $\mathsf{ArzelaAscoli}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. The surface excludes open-cover compactness, subsequence choice, ambient function-space compactness, and any topology on all maps from the source to the target.*

Proof. First read the carrier coordinates in Definition 441.1. They provide the compact source row, target metric row, finite family ledger, requested tolerance, shared radius, source-net bundle, stability row, equicontinuity row, handoff row, transport rows, continuation routes, package provenance, and local naming row.

The finite family side is fixed by Definition 441.2 and Definition 441.3. These two rows bind the common rational tolerance-radius pair to the displayed family ledger and bundled source centers.

The transport and handoff side is fixed by Definition 441.6, Definition 441.7, and Definition 441.8. Those results keep the same $T, R, F, \mathcal{P}, S, E, W, H, C, P, N$ rows visible to every $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer.

Finally apply Definition 441.9. Refining the finite source-net can add only bundled-center rows under the same boundary, so all consumer reads factor through the displayed public rows and no compactness, choice, or host function-space principle is exported. □

Theorem 441.11 (AscoliModulus empty-family vacuity)**.** *Let $A = (X, Y, F, T, R, \mathcal{P}, S, E, W, H, C, P, N)$ be an accepted* $\mathsf{AscoliModulus}^{\uparrow}$ *carrier as in Definition 441.1. Suppose that F is the empty finite family ledger, so no displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *row is indexed by F, and that $\mathcal{P} = \mathsf{Bnil}$ is the empty source-net* $\mathsf{ProbeBundle}$*. Then the finite-net stability and family-wide equicontinuity obligations in A are vacuous: S has no bundled-center and family-row pair to test, and E has no family-row instance to which the shared tolerance-radius pair T, R can be applied.*

Consequently an $\mathsf{ArzelaAscoli}^{\uparrow}$ *consumer reading this boundary case can receive only the displayed tolerance row T, the shared radius row R, the handoff and transport rows W, H, C, the package provenance P, and the local naming row N. It receives no nonempty finite-net stability instance, no family-member equicontinuity instance, no open-cover compactness row, no subsequence-choice row, and no host function-space compactness theorem.*

Proof. Project the rows of the carrier in Definition 441.1. The stability row S is indexed by two finite sources: a bundled source center in $\mathcal{P}$ and a displayed family row in F. This is exactly the finite indexing described by Definition 441.3. If $\mathcal{P} = \mathsf{Bnil}$, there is no bundled source center. If F is empty, there is no displayed $\mathsf{ContinuousMap}^{\uparrow}$ family row. Hence there is no index pair on which a source-radius test or target-distance read can be demanded from S.

The equicontinuity row E is controlled by Definition 441.2: it binds the requested tolerance row T and the shared radius row R uniformly over the displayed family ledger F. Since F has no entries, this uniform family-side clause has no instances. The row still records that T and R are the displayed rational tolerance-radius pair, but it cannot create a hidden family member, a separate modulus function, or an unlisted continuity obligation.

Apply the handoff theorem Definition 441.5 together with the rational-radius transport of Definition 441.6. The only non-indexed payload that remains available to the $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer is the displayed tolerance row T, the displayed radius row R, the handoff and transport surface W, H, C, the package row P, and the local certificate row N. The family transport package of Definition 441.7 has no family comparison to inspect, and the finite-net stability theorem has no probe-center comparison to inspect. Therefore the empty-family and empty-bundle boundary exports only those displayed rows and no compactness, choice, or host function-space structure beyond the accepted $\mathsf{AscoliModulus}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliModulus}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AscoliModulus}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source, a finite family ledger of $\mathsf{ContinuousMap}^{\uparrow}$ rows, a requested $\mathsf{Rat}^{\uparrow}$ target tolerance, a shared $\mathsf{Rat}^{\uparrow}$ source radius, finite ProbeBundle source-net stability rows, an $\mathsf{EquicontinuityFamily}^{\uparrow}$ ledger, $\mathsf{UniformContinuity}^{\uparrow}$ pointwise rows, an $\mathsf{ArzelaAscoli}^{\uparrow}$ consumer handoff, displayed hsame and Cont transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AscoliModulus}^{\uparrow}}, \mathsf{obligation})$

Scope closed: The seed scope registers the carrier, finite equicontinuity ledger, finite-net stability, local naming obligations, and Arzela–Ascoli handoff for displayed compact-metric family-modulus packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim open-cover compactness, subsequence choice, host function-space compactness, or a topology on all maps from the source to the target.

Upgrade path: Scoped closure requires the same compact-metric, continuous-map, uniform-continuity, equicontinuity-family, ProbeBundle, hsame, Cont, Pkg, and local NameCert rows to be bound as a single consumer-stable public surface.

Chapter 442

A Concrete Naming Certificate for $\mathsf{TaylorModel}^{\uparrow}$

The $\mathsf{TaylorModel}^{\uparrow}$ packet names the finite Taylor-jet enclosure method used in Berz-style and Makino–Berz validated numerics, but it records only finite BEDC rows. A packet exposes a center observation, dyadic jet coefficients, an interval or dyadic-ball remainder, an interval-arithmetic ledger, a real-name evaluation route, a validated readback certificate, package provenance, and a local naming row. It does not import host Taylor series semantics, analytic convergence, selected real limits, or quotient real equality.

The carrier consumes the interval-operation ledger of chapter 302, the dyadic-ball remainder surface of chapter 311, the validated readback certificate of chapter 357, and the NewtonIteration provenance route of chapter 223. These sibling rows are the horizontal dependency lattice for the center, jet, remainder, operation, readback, and provenance coordinates below.

Definition 442.1 (TaylorModel BHist carrier)**.** *A* $\mathsf{TaylorModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (c, J, R, I, E, V, P, N)$$

with the following displayed coordinates:

1. *c is a center* $\mathsf{Real}^{\uparrow}$ *finite observation row;*
2. *J is a finite jet ledger of dyadic coefficient histories, with every coefficient read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
3. *R is an* $\mathsf{IntervalArithmetic}^{\uparrow}$ *interval row or* $\mathsf{DyadicBall}^{\uparrow}$ *remainder enclosure attached to the same precision window;*
4. *I is the* $\mathsf{IntervalArithmetic}^{\uparrow}$ *operation ledger recording each polynomial, interval, and remainder operation used by the enclosure;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window evaluation route through which consumers read the center, coefficient, and remainder rows;*
6. *V is the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *readback certificate asserting that the displayed route returns an enclosure row rather than a selected host value;*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{IntervalArithmetic}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{ValidatedNumerics}^{\uparrow}$, *and* $\mathsf{NewtonIteration}^{\uparrow}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorModel}^{\uparrow}}$ *row.*

Two accepted carriers are classified together only when the center row, finite jet ledger, remainder enclosure, interval-operation ledger, evaluation route, readback certificate, package row, and local naming row are transported componentwise by the displayed hsame *comparisons and* Cont *routes.*

Lean4 (definition) `BEDC.Derived.TaylorModelUp.TaylorModelCarrier`

Theorem 442.2 (TaylorModel NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{TaylorModel}^\uparrow}$ *surface for the carrier of Definition 442.1 consists of five rows:*

1. *carrier habitation by the finite packet* $T = (c, J, R, I, E, V, P, N)$*;*
2. *classifier stability under componentwise* hsame *comparisons and displayed* Cont *routes for* c, J, R, I, E, V, P, N*;*
3. *finite-jet transport through the* $\mathsf{DyadicRatCore}^\uparrow$ *coefficient histories and their* $\mathsf{StreamName}^\uparrow$ *windows;*
4. *interval-remainder soundness through the* $\mathsf{IntervalArithmetic}^\uparrow$ *ledger and the displayed* R *enclosure;*
5. *non-escape from host Taylor series, analytic convergence, selected real limits, and quotient real equality.*

Proof. Carrier habitation is projection from Definition 442.1. The center, finite jet, remainder, ledger, evaluation, readback, package, and local certificate rows are visible coordinates of one finite packet.

Classifier stability is componentwise. The accepted classifier may replace a displayed row only by a recorded hsame comparison or by a Cont route that reads the same row inside the same packet.

The remaining rows are the coefficient transport, interval-remainder enclosure, and non-escape boundary stated below. Together they describe exactly what the local naming certificate must expose to a consumer of the finite Taylor-jet enclosure. □

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelCarrier namecert obligations`

Theorem 442.3 (TaylorModel finite-jet transport)**.** *For every accepted* $\mathsf{TaylorModel}^\uparrow$ *carrier* $T = (c, J, R, I, E, V, P, N)$*, every consumer-visible coefficient read factors through the finite jet ledger* J*, its displayed* $\mathsf{DyadicRatCore}^\uparrow$ *coefficient histories, the finite* $\mathsf{StreamName}^\uparrow$ *windows used to inspect them, the evaluation route* E*, and the componentwise* hsame *and* Cont *rows recorded by the packet classifier.*

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelFiniteJet transport`

Proof. Project the packet to its jet coordinate J. Since J is finite, each coefficient requested by a consumer is one of the displayed dyadic coefficient histories rather than a term of an ambient infinite series.

The consumer reaches a coefficient through the $\mathsf{StreamName}^\uparrow$ window named in J and through the evaluation route E. Transport along accepted classifiers preserves those two displayed rows by the stored hsame comparisons and Cont routes.

Thus the coefficient read remains inside the finite jet ledger and cannot appeal to a host Taylor series or to an unbounded coefficient family. □

Theorem 442.4 (TaylorModel interval-remainder soundness)**.** *Every enclosure exported by an accepted* $\mathsf{TaylorModel}^\uparrow$ *carrier is bounded by the displayed remainder row* R *and the* $\mathsf{IntervalArithmetic}^\uparrow$ *operation ledger* I*. The bound is read through the same center row* c*, finite jet ledger* J*, evaluation route* E*, and validated readback certificate* V*, and it exports no external analytic convergence theorem.*

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelCarrier interval remainder soundness`

Proof. Start with an exported enclosure read. The read first identifies the center row c and the finite jet ledger J used by the polynomial part of the packet.

The remainder coordinate R is then read through the interval or dyadic-ball enclosure named by the packet. The operation ledger I records the finite interval-arithmetic route by which polynomial and remainder rows are combined.

The validated readback row V certifies that the consumer receives the displayed enclosure. Since every operation has already passed through I and R, soundness is a statement about those finite rows and not about analytic convergence in a host real model. □

Theorem 442.5 (TaylorModel validated consumer boundary)**.** *Every* ValidatedNumerics$^{\uparrow}$ *consumer of an accepted* TaylorModel$^{\uparrow}$ *packet receives only the displayed center row, finite jet ledger, remainder enclosure, interval-operation ledger,* Real$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *evaluation route, readback certificate, package provenance, and local naming row. The consumer receives no host Taylor series object, no selected real limit, and no quotient real equality witness.*

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelCarrier validated consumer boundary`

Proof. Read the carrier coordinates of Definition 442.1. The rows available to a validated-numerics consumer are exactly c, J, R, I, E, V, P, N.

Apply Definition 442.3 to every coefficient request and Definition 442.4 to every enclosure request. These two routes account for all numeric information exposed by the packet.

No remaining coordinate names a host Taylor series, an analytic radius of convergence, a selected real limit, or quotient equality of real names. The consumer boundary is therefore the finite readback surface alone. □

Theorem 442.6 (TaylorModel seed public package)**.** *The carrier of Definition 442.1, the NameCert surface of Definition 442.2, the finite-jet transport of Definition 442.3, the interval-remainder soundness of Definition 442.4, and the validated consumer boundary of Definition 442.5 assemble the seed* TaylorModel$^{\uparrow}$ *public package over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, IntervalArithmetic$^{\uparrow}$, DyadicBall$^{\uparrow}$, ValidatedNumerics$^{\uparrow}$, *and* NewtonIteration$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelCarrier seed public package`

Proof. Project the finite packet

$$T = (c, J, R, I, E, V, P, N).$$

The carrier and NameCert surface expose the rows required for a local naming certificate.

The finite-jet theorem accounts for coefficient transport, and the interval-remainder theorem accounts for enclosure soundness through the displayed operation ledger. The validated consumer theorem then fixes the downstream boundary of every readback.

The package provenance P names exactly the dependency rows listed in the theorem statement. Hence the seed package is a finite BEDC naming certificate for Taylor-model enclosures, with no analytic object outside the displayed packet. □

Definition 442.7 (TaylorModel displayed finite-jet subwindow)**.** *Let $T = (c, J, R, I, E, V, P, N)$ be an accepted* TaylorModel$^{\uparrow}$ *carrier in the sense of Definition 442.1. A displayed finite-jet subwindow $J' \preceq_J J$ is a finite prefix or finite contiguous subledger of J equipped with the following retained rows:*

1. *every coefficient history displayed in J' is one of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *coefficient histories displayed in J;*
2. *the* $\mathsf{StreamName}^{\uparrow}$ *window attached to a retained coefficient is the same displayed window as in J, or a displayed smaller finite window reached by a* Cont *route already present in the packet;*
3. *the restricted evaluation row $E|_{J'}$ contains exactly the coefficient-read routes of E whose coefficient selector lies in J';*
4. *the restricted readback row $V|_{J'}$ validates only the finite reads routed through $E|_{J'}$;*
5. *the restricted provenance and local name rows $P|_{J'}$ and $N|_{J'}$ name the same dependency certificates as P and N, but only for the retained coefficient selectors and their windows.*

The projected TaylorModel read surface over J' is the finite packet

$$T|_{J'} = (c, J', R|_{J'}, I|_{J'}, E|_{J'}, V|_{J'}, P|_{J'}, N|_{J'}),$$

where $R|_{J'}$ and $I|_{J'}$ are the remainder and interval-operation rows read only at the retained coefficient selectors. The notation records a displayed packet restriction; it is not a quotient of Taylor series and it does not introduce an ambient coefficient family.

Lean4 (definition) `BEDC.Derived.TaylorModelUp.TaylorModelDisplayedFiniteJetSubwindow`

Proposition 442.8 (TaylorModel finite-jet prefix restriction)**.** *Let $T = (c, J, R, I, E, V, P, N)$ be an accepted* $\mathsf{TaylorModel}^{\uparrow}$ *carrier, and let $J' \preceq_J J$ be a displayed finite-jet subwindow as in Definition 442.7. Suppose every consumer-visible coefficient request under consideration has its coefficient selector in J' and that its* $\mathsf{StreamName}^{\uparrow}$ *window is one of the retained windows of J'. Then the projected packet $T|_{J'}$ is an accepted finite* $\mathsf{TaylorModel}^{\uparrow}$ *read surface for those consumers. Moreover, every such consumer-visible coefficient read factors through J', $E|_{J'}$, and the retained* $\mathsf{StreamName}^{\uparrow}$ *window, and no coefficient row in $J \setminus J'$ is accessed.*

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelCarrier finite jet prefix restriction`

Proof. Project the accepted carrier through Definition 442.1. The jet coordinate J is a finite ledger of displayed $\mathsf{DyadicRatCore}^{\uparrow}$ coefficient histories, and the only public coefficient-read routes are the finite $\mathsf{StreamName}^{\uparrow}$ windows and the evaluation route E attached to those histories.

By Definition 442.7, the subwindow J' retains only coefficient histories already displayed in J. Hence every coefficient row in J' is still a valid dyadic coefficient history, and every retained window is either the same finite window as in J or a smaller finite window reached by a displayed Cont route. Since a finite prefix or finite contiguous subledger of a finite ledger is finite, J' is again a finite coefficient ledger.

Restrict the evaluation and readback rows to the retained selectors. The row $E|_{J'}$ contains exactly the routes by which the selected coefficients are read, and $V|_{J'}$ validates exactly those finite routes as readback rows. The restricted provenance and local naming rows keep the same dependency certificates named by the original packet, now indexed only by the retained selectors. Thus the projected packet has the same kind of visible coordinates as the TaylorModel carrier, but with its coefficient surface replaced by the displayed finite subwindow.

Apply Definition 442.3 to a consumer-visible coefficient request for T. The original factorization passes through J, the coefficient history selected there, the attached $\mathsf{StreamName}^{\uparrow}$ window, the evaluation route E, and the componentwise hsame and Cont rows recorded by the packet classifier. The hypothesis says that the selector for this request lies in J' and that the attached window is

retained in J'. Therefore the same factorization factors through J' and $E|_{J'}$ after replacing each unrestricted row by its retained projection.

No discarded coefficient is needed in this read. A row in $J \setminus J'$ has no selector in the restricted consumer surface, no route in $E|_{J'}$, and no validation entry in $V|_{J'}$. The interval and remainder rows used by the projected surface are likewise the restricted rows $I|_{J'}$ and $R|_{J'}$, so a retained coefficient read cannot obtain an outside coefficient indirectly through the operation ledger.

Finally apply Definition 442.5. A validated consumer receives only the displayed center row, finite jet ledger, remainder enclosure, interval-operation ledger, evaluation route, readback certificate, package provenance, and local naming row. For the projected surface these rows are exactly c, J', $R|_{J'}$, $I|_{J'}$, $E|_{J'}$, $V|_{J'}$, $P|_{J'}$, and $N|_{J'}$. Thus the projected packet is an accepted finite TaylorModel read surface for the stated consumers, and every visible coefficient read is exhausted by the retained finite jet subwindow. □

Theorem 442.9 (TaylorModel finite-jet remainder obligation route)**.** *For every accepted* $\mathsf{TaylorModel}^{\uparrow}$ *carrier* $T = (c, J, R, I, E, V, P, N)$ *and every displayed finite-jet subwindow* $J' \preceq_J J$*, the obligation route for a remainder read consists exactly of carrier admission for* $T|_{J'}$*, classifier stability under* hsame *and* Cont*, finite-jet transport through the retained* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *rows, interval-remainder soundness through* $R|_{J'}$ *and* $I|_{J'}$*, validated consumer exposure through* $E|_{J'}$ *and* $V|_{J'}$*, and ledger nonescape through* $P|_{J'}$ *and* $N|_{J'}$*. No obligation row is supplied by host Taylor series semantics, analytic convergence, a selected real limit, quotient real equality, or a remainder object outside the displayed* BHist *packet.*

Proof. Carrier admission is the projected packet of Definition 442.8. The rows of $T|_{J'}$ display the center, retained finite jet, restricted remainder, restricted interval-operation ledger, evaluation route, readback certificate, provenance, and local naming data.

Classifier stability follows componentwise from the accepted $\mathsf{TaylorModel}^{\uparrow}$ classifier. The retained hsame and Cont rows transport only the displayed coordinates, so no hidden coefficient family or analytic series can enter the restricted read surface.

Apply finite-jet transport and interval-remainder soundness from Definition 442.3 and Definition 442.4. They force coefficient reads through the retained dyadic and StreamName rows and force enclosure reads through the restricted remainder and interval-operation ledger.

Finally use the validated consumer boundary of Definition 442.5. The consumer receives the restricted evaluation and readback rows together with provenance and local naming, while every refused object lacks a carrier coordinate and a replay edge. □

Lean4 ✓ `BEDC.Derived.TaylorModelUp.TaylorModelRemainderObligationRoute`

↪ variant `BEDC.Derived.TaylorModelUp.TaylorModelCarrier remainder obligation route`

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorModel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaylorModel}^{\uparrow}$ packet is generated from a center $\mathsf{Real}^{\uparrow}$ observation row, a finite dyadic jet ledger, an interval or dyadic-ball remainder enclosure, an $\mathsf{IntervalArithmetic}^{\uparrow}$ operation ledger, a $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ evaluation route, a $\mathsf{ValidatedNumerics}^{\uparrow}$ readback certificate, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorModel}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The scoped finite-jet and remainder route records the finite carrier, NameCert obligation surface, finite-jet transport, interval-remainder soundness, validated consumer boundary, finite-jet prefix restriction, and finite-jet remainder obligation route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaylorModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Taylor series semantics, analytic convergence, selected real limits, quotient real equality, public Taylor-model algebra, or bridge checking.

Upgrade path: The next verification step is checked carrier habitation, classifier stability under hsame and Cont, finite-jet transport, interval-remainder soundness, validated consumer boundary, and finite-jet remainder nonescape for the same packet rows.

Chapter 443

Regular Cauchy Limit Classifier NameCert Construction

The $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ interface is the finite classifier handoff between the diagonal-limit construction of Definition 9238.3 and the limit-existence target of Definition 9238.5. It packages the selected $\mathsf{RegSeqRat}^{\uparrow}$ diagonal extraction, the $\mathsf{StreamName}^{\uparrow}$ observation windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ rational ledgers, and the $\mathsf{Real}^{\uparrow}$ seal row through which a completion consumer sees the candidate limit. The packet is a classifier surface for the limit handoff; it is not an ambient completeness principle and it contains no host selected real.

The classifier is consumed by the finite observation-budget route of chapter 611. $\mathsf{RealObservationBudget}^{\uparrow}$ supplies the finite-window, dyadic-ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal boundary through which completion consumers inspect the classifier.

Definition 443.1 (Regular Cauchy limit classifier carrier)**.** *A* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy input family and μ is its explicit modulus row;*
2. *Δ is the diagonal extraction row assigning to each unary precision request the selected threshold and diagonal read demanded by Definition 9238.3;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family that realizes the selected reads of the regular rational names;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations obtained through W;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording endpoints, radii, comparison rows, and precision rows for those observations;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row exposing the same diagonal observations as the candidate real-name surface;*
7. *H stores componentwise* hsame *transports for the input family, modulus row, diagonal extraction, windows, readbacks, ledger, and seal row;*
8. *C stores displayed* Cont *routes for every consumer read that reaches those rows;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a host real value, a non-displayed limit selector, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier`

Theorem 443.2 (Regular Cauchy limit classifier stability)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitClassifier}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exactness: every exported rational observation is the endpoint, radius, comparison, and precision row recorded by D;*
2. $\mathsf{StreamName}^{\uparrow}$ *window exactness: each diagonal read uses the finite window named by W and no unrecorded stream access;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *extraction stability: transporting X, μ, and Δ through the displayed* hsame *and* Cont *rows transports the selected regular-sequence readback R;*
4. $\mathsf{Real}^{\uparrow}$ *seal discipline: the seal row E is reached only after the displayed window, readback, and dyadic ledger rows have been produced;*
5. *package and name closure: P and N record exactly the provenance and local classifier row consumed by downstream Real and Completion certificates.*

Consequently classifier transport preserves the finite diagonal-limit handoff from source packets to target packets without adding quotient representatives, hidden tail choices, host real equality tests, or ambient completeness assumptions.

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier stability`

Proof. Project the carrier of Definition 443.1. The rows X, μ, and Δ determine which regular rational name is read at each precision request, while W records the finite stream window that performs the read.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback R and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D are coordinates of the same packet. Transport by the stored hsame rows in H and consumer routes in C therefore moves the selected readback together with its displayed endpoint, radius, comparison, and precision data.

The $\mathsf{Real}^{\uparrow}$ row E consumes only the already displayed W, R, and D rows. Repacking those rows with P and N gives the transported local classifier surface and leaves no coordinate through which a non-displayed limit selector or quotient equality can enter. □

Theorem 443.3 (Regular Cauchy limit Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *classifier read exported by an accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *carrier*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

factors through the displayed diagonal extraction row Δ, the $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, and the seal row E. This handoff is the finite NameCert surface by which the Real completeness target sees a candidate limit classifier; it exports no quotient stream equality, selected host limit, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier real seal handoff`

Proof. Begin with the stability surface of Definition 443.2. It identifies the only classifier-visible diagonal observations as the rows produced by Δ, W, R, and D.

The seal row E is downstream of those observations. A consumer read into $\mathsf{Real}^{\uparrow}$ must therefore pass through the finite stream window and dyadic ledger before reaching the real-name handoff.

Finally, the carrier definition lists all coordinates of Q. Since none of them is a quotient representative, a host selected limit, a host real equality test, or an ambient completeness field, the exported classifier cannot supply any of those objects. □

Theorem 443.4 (Regular Cauchy limit classifier diagonal-window public boundary)**.** *Every public completion-facing read exported by an accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *carrier*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

factors through the displayed diagonal extraction row Δ*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Its provenance is the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; it exports no quotient stream equality, host selected limit, host real equality, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier diagonal window public boundary`

Proof. Project the carrier in Definition 443.1. The public-facing rows are Δ, W, R, D, E together with the transport, continuation, package, and naming rows H, C, P, N.

Apply Definition 443.2. The stability obligations make D the exact dyadic ledger, make W the finite window family for diagonal reads, and transport R together with the same displayed diagonal extraction data.

Now apply Definition 443.3. It places every $\mathsf{Real}^{\uparrow}$ classifier read downstream of Δ, W, R, D, E, so a completion-facing consumer receives only the sealed diagonal-window route.

The remaining exclusions follow by carrier inspection. Since the packet has no coordinate for quotient stream equality, a host selected limit, host real equality, or ambient completeness, the public boundary cannot export any of those objects. □

Theorem 443.5 (Regular Cauchy limit classifier public export)**.** *The public* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitClassifier}^{\uparrow}}$ *export of an accepted carrier*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

is exactly the tuple of displayed diagonal extraction data Δ*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows* D*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*,* BHist *history surface,* hsame *transports* H*,* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N *isolated by Definition 443.2, Definition 443.3, and Definition 443.4. It exports no quotient stream equality, host selected limit, host real equality, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier public export`

Proof. First apply Definition 443.2. It fixes the displayed diagonal extraction, window, readback, dyadic ledger, transport, package, and local naming rows as the stable classifier surface.

Next apply Definition 443.3. This places the $\mathsf{Real}^{\uparrow}$ seal downstream of the same Δ, W, R, D rows, so the real-name handoff contributes only the recorded seal row E.

Finally use Definition 443.4. It identifies the completion-facing public boundary with the visible BHist, hsame, Cont, Pkg, and NameCert provenance already carried by Q.

The carrier inventory has no coordinate for quotient stream equality, a host selected limit, host real equality, or ambient completeness. The public export is therefore exactly the displayed tuple and none of those excluded objects. □

Theorem 443.6 (Regular Cauchy limit classifier bridge source packet)**.** *For every accepted* RegularCauchyLimitClass *carrier, the bridge-facing source packet is exactly*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

with diagonal extraction Δ*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows* H, C, P, N*. The source packet exports no quotient stream equality, selected host limit, host real equality, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier bridge source packet`

Proof. Begin with the stability theorem Definition 443.2. It fixes the diagonal extraction, stream-window, regular readback, dyadic ledger, transport, continuation, package, and naming rows as the stable classifier surface.

Next use the Real seal handoff and diagonal-window public boundary, Definition 443.3 and Definition 443.4. These rows place the real-facing read downstream of Δ, W, R, D, E and keep the completion-facing boundary inside the same displayed packet.

Finally project the public export of Definition 443.5. The bridge source is therefore the public tuple Q itself, with no additional coordinate for quotient equality, a selected host limit, host real equality, or ambient completeness. □

Theorem 443.7 (Regular Cauchy limit classifier diagonal seal uniqueness)**.** *Let* Q *and* Q' *be accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packets. If they share the same diagonal extraction row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, and* $\mathsf{Real}^{\uparrow}$ *seal row up to the displayed* hsame *transports, then every public completion-facing classifier read factors through the same sealed diagonal-window route.*

The conclusion is a uniqueness statement for the public readback route only. It does not assert quotient stream equality, select a host limit, identify host real values, or use ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier diagonal seal uniqueness`

Proof. First apply classifier stability from Definition 443.2. The shared diagonal extraction, stream-window, regular readback, dyadic ledger, and seal rows remain the stable coordinates visible to both packets.

Next use the Real seal handoff of Definition 443.3. It places each real-facing read downstream of the same diagonal extraction, finite windows, readback rows, dyadic ledgers, and seal row.

Now apply the diagonal-window public boundary and public export, Definition 443.4 and Definition 443.5. They force every completion-facing read through the displayed BHist, hsame, Cont, Pkg, and NameCert route attached to those shared rows.

Finally use the bridge source packet of Definition 443.6. Since the bridge-facing packet has no coordinate for quotient stream equality, a selected host limit, host real equality, or ambient completeness, the shared sealed route is the only public route available to both classifiers. □

Theorem 443.8 (Regular Cauchy limit classifier observation-budget factorization)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet, and let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. If the diagonal windows, dyadic ledger, regular readback, and* $\mathsf{Real}^{\uparrow}$ *seal row of* Q *are read through the displayed rows of* B*, then every* $\mathsf{Real}^{\uparrow}$ *seal handoff of* Q *factors through the observation-budget route*

$$E_B \longmapsto W_B \longmapsto D_B \longmapsto R_B \longmapsto S_B.$$

The factorization is a finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route; it adds no quotient stream equality, host real equality, selected limit, ambient completeness premise, hidden limit selector, or unbounded schedule.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier observation budget factorization`

Proof. Start with the classifier rows. By Definition 443.2 and Definition 443.3, the only $\mathsf{Real}^{\uparrow}$ handoff exported by Q is downstream of Δ, W, R, D, E.

Now read those rows through the observation-budget packet. Definition 611.17 makes every admitted seal read pass through E_B, W_B, D_B, R_B, S_B, and Definition 611.31 places both the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers on that same finite route.

Finally apply the public boundary and non-escape results. Definition 443.4 keeps the classifier read inside the displayed public packet, while Definition 611.6 rules out quotient streams, host equality, hidden selectors, ambient completeness, and unbounded schedules. □

Theorem 443.9 (Regular Cauchy limit classifier finite-route exactness)**.** *Under the hypotheses of Definition 443.8, the classifier rows* $\Delta, W, R, D, E, H, C, P, N$ *and the budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B$ *determine the same finite window, dyadic ledger, regular readback, and seal boundary up to the displayed* hsame *transports and* Cont *routes. Equivalently, any accepted completion-facing read that agrees with both packets on those displayed rows is the same finite route from the diagonal classifier to the observation-budget seal boundary.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierFiniteRouteExactness`
↪ variant `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier finite route exactness`

Proof. First identify the classifier side. Definition 443.7 says that shared diagonal windows, readbacks, dyadic ledgers, and seal rows force every public completion-facing read through the same sealed diagonal-window route.

Next identify the observation-budget side. Definition 611.19 confines classifier comparison to $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, and Definition 611.31 routes every admitted $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through those rows.

The factorization theorem now composes the two identifications. The displayed hsame rows carry row equality, the displayed Cont rows replay the consumer route, and no additional analytic coordinate is available to distinguish the classifier route from the observation-budget route. □

Theorem 443.10 (Regular Cauchy limit classifier completion-consumer boundary)**.** *Composing an accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet with an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet through the factorization of Definition 443.8 gives a completion consumer only the finite rows*

$$\Delta, W, R, D, E, H, C, P, N$$

and

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

It gives no quotient stream equality, host real equality, selected limit, ambient completeness theorem, hidden limit selector, or unbounded observation schedule.

Proof. Read the composed packet through the classifier first. Definition 443.2 and Definition 443.3 restrict the limit-classifier handoff to Δ, W, R, D, E with the displayed transport, route, provenance, and local naming rows.

Read the same packet through the budget route next. Definition 611.17 and Definition 611.31 place every admitted consumer on E_B, W_B, D_B, R_B, S_B with H_B, C_B, P_B, N_B.

Finally combine diagonal uniqueness with budget non-escape. Definition 443.7 prevents a second public classifier route with the same displayed seal rows, and Definition 611.6 excludes quotient stream equality, host real equality, hidden selection, ambient completeness, and unbounded schedules from the budget side. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier completion consumer boundary`

Theorem 443.11 (Regular Cauchy limit classifier synchronizer factorization)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet, and let*

$$Z = (E_Z, S_Z, B_Z, T_Z, D_Z, L_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet. Suppose that the diagonal window, regular readback, dyadic ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows of* Q *agree componentwise with the selector, budget, shared-threshold, and seal rows displayed by* Z*. Then every completion-facing read of* Q *factors through the synchronizer route*

$$D_Z \longmapsto B_Z \longmapsto T_Z \longmapsto L_Z \longmapsto S_Z.$$

The resulting route is exhausted by the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the two packets. It exports no quotient stream, selected host limit, hidden selector, host real equality, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier synchronizer factorization`

Proof. Start from the observation-budget factorization. Definition 443.8 places every $\mathsf{Real}^{\uparrow}$ seal handoff of Q on the finite observation route, and Definition 443.9 identifies that route with the classifier rows visible to completion consumers.

Next read the synchronizer packet. The componentwise agreement hypotheses identify the classifier diagonal window, readback, dyadic ledger, and seal rows with the synchronizer selector,

budget, shared-threshold, and seal rows. The synchronizer carrier and obligation rows then force the read through D_Z, B_Z, T_Z, L_Z, S_Z before any completion-facing consumer is served.

Finally apply the completion-consumer boundary of Definition 443.10 together with the synchronizer non-escape rows. Since neither packet contains a quotient-stream coordinate, a host limit selector, host real equality, or an ambient completeness premise, the composed handoff exposes only the finite seal-budget synchronization surface. □

Theorem 443.12 (Regular Cauchy limit classifier budget-synchronizer route agreement)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet. Let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and let*

$$Z = (E_Z, S_Z, B_Z, T_Z, D_Z, L_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet. Suppose that B and Z read the same displayed diagonal window, regular readback rows, dyadic ledger rows, and* $\mathsf{Real}^{\uparrow}$ *seal row of Q through their displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Then the observation-budget route and the synchronizer route determine one finite completion-facing classifier route: the same* BHist *history rows, the same* hsame *transports, the same* Cont *replay, the same* Pkg *provenance, and the same local* NameCert *handoff.*

The agreement is a finite public-route statement. It does not assert quotient stream equality, host real equality, a selected limit, a hidden selector, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier budget synchronizer route agreement`

Proof. First invoke the observation-budget factorization. By Definition 443.8, every $\mathsf{Real}^{\uparrow}$ seal handoff of Q that is read through B factors through E_B, W_B, D_B, R_B, S_B with only the displayed transport, continuation, provenance, and naming rows.

Next invoke the synchronizer factorization. By Definition 443.11, the same displayed diagonal window, regular readback, dyadic ledger, and seal rows read through Z factor through D_Z, B_Z, T_Z, L_Z, S_Z before any completion-facing consumer is served.

Use diagonal-seal uniqueness and finite-route exactness to compare the two routes. Definition 443.7 identifies every public read with the sealed diagonal-window route, and Definition 443.9 identifies that public read with the finite observation-budget route. Hence the synchronizer route and the budget route are two presentations of the same completion-facing BHist, hsame, Cont, Pkg, and NameCert path.

Finally inspect the carrier coordinates of Q, B, and Z. None contains quotient stream equality, host real equality, a selected limit, a hidden selector, or an ambient completeness witness, so no further route can distinguish the two finite presentations. □

Theorem 443.13 (Regular Cauchy limit classifier seal-budget-synchronizer exhaustion)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet. Let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and let*

$$Z = (E_Z, S_Z, B_Z, T_Z, D_Z, L_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet. Suppose that the three packets agree on the displayed diagonal window, regular readback, dyadic ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows through their named* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes. Then every Real-completion-facing consumer route exported by this composite packet is exhausted by the shared finite route already identified by the classifier, observation-budget, and synchronizer surfaces.*

In particular, quotient-stream reads, host real equality, selected limits, hidden selectors, and ambient completeness premises have no exit from the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Proof. First apply the observation-budget factorization. Definition 443.8 puts every $\mathsf{Real}^{\uparrow}$ seal handoff of Q on the finite budget route, while Definition 443.10 states that completion consumers of this route can see only the displayed finite rows.

Next apply the synchronizer factorization. Definition 443.11 puts the same diagonal window, regular readback, dyadic ledger, and seal rows on the synchronizer route, and Definition 443.12 identifies that route with the observation-budget presentation.

Finally read the boundary rows componentwise. Since the shared route contains only BHist, hsame, Cont, Pkg, and NameCert data, every Real-completion-facing consumer terminates at that finite route and cannot obtain quotient streams, host real equality, selected limits, hidden selectors, or ambient completeness from the packet. □

Theorem 443.14 (Regular Cauchy limit classifier real tail-agreement seal factorization)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet. Let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, let*

$$Z = (E_Z, S_Z, B_Z, T_Z, D_Z, L_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, and let*

$$A = (R_A, S_A, W_A, D_A, A_A, H_A, C_A, P_A, N_A)$$

be an accepted $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Suppose that these packets read the same displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks,* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, and* $\mathsf{Real}^{\uparrow}$ *seal row of* Q *through their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes. Then every completion-facing classifier read factors through the finite chain*

$$\mathsf{RegularCauchyLimitClassifier}^{\uparrow} \longmapsto \mathsf{RealObservationBudget}^{\uparrow} \longmapsto \mathsf{CauchySealBudgetSynchronizer}^{\uparrow} \longmapsto \mathsf{RealTailAgreem}$$

The factorization exports no quotient stream equality, host real equality, selected limit, hidden tail selector, ambient completeness premise, or choice-selected tail.

Proof. First put the classifier on the observation-budget route. The factorization of Definition 443.8 sends every $\mathsf{Real}^{\uparrow}$ seal handoff of Q through E_B, W_B, D_B, R_B, S_B, and Definition 611.9 records that $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers of B read exactly those finite rows.

Next compare the budget route with the synchronizer route. The route agreement theorem Definition 443.12 identifies the observation-budget and synchronizer presentations, while Definition 886.6 exhausts the Real-facing synchronizer read by the request, seal, budget, tail schedule, selector, compatibility, transport, route, provenance, and local naming rows of Z.

Finally pass to the paired tail-agreement seal. The shared window, dyadic readback, regular source, and seal hypotheses place the same read on the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ boundary of Definition 699.58. Its completion handoff theorem Definition 699.65 then makes the displayed chain the whole completion-facing route, and the non-escape clauses in the cited packets exclude quotient streams, host real equality, selected limits, hidden tail selectors, ambient completeness, and choice-selected tails. □

Theorem 443.15 (Regular Cauchy limit classifier budgeted seal pullback)**.** *Let*

$$Q = (X, \mu, \Delta, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet. Let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and let*

$$Z = (E_Z, S_Z, B_Z, T_Z, D_Z, L_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet. Suppose that Q, B, and Z share the displayed diagonal window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, observation budget, synchronizer, and* $\mathsf{Real}^{\uparrow}$ *seal rows through their displayed* hsame, Cont, Pkg, *and* NameCert *routes. Then every completion-facing classifier read factors through the same finite pullback route determined by those shared rows.*

The pullback exports no quotient stream equality, host real equality, selected limit, hidden selector, unbounded schedule, or ambient completeness.

Proof. First combine the observation-budget factorization with finite-route exactness. By Definition 443.8, every seal handoff read through B lies on the finite observation route, and Definition 443.9 identifies that route with the classifier rows of Q.

Next combine the synchronizer factorization with route agreement. By Definition 443.11, the same displayed diagonal window, readback, dyadic ledger, and seal rows read through Z lie on the synchronizer route. The agreement theorem Definition 443.12 identifies the budget route and synchronizer route as one completion-facing BHist, hsame, Cont, Pkg, and NameCert path.

Finally apply the completion-consumer boundary. Definition 443.10 excludes quotient stream equality, host real equality, selected limits, hidden selectors, ambient completeness, and unbounded schedules from the composed packet, so the shared finite pullback is the whole classifier read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ packet is generated by a displayed regular Cauchy input, its modulus row, the diagonal extraction row, finite stream windows, regular-sequence readbacks, dyadic observation ledgers, a RealUp seal row, transport routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitClassifier}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 443.1, Definition 443.2, Definition 443.3, Definition 443.4, Definition 443.5, Definition 443.6, and Definition 443.7 bind the public limit-classifier handoff, bridge-facing source packet, and uniqueness boundary to the displayed DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyLimitClassifier}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyLimitClassifierUp.RegularCauchyLimitClassifierCarrier\_diagonal\_seal\_uni`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, or a public standard bridge.

Upgrade path: Axiom-purity and standard-bridge checking remain the next verification obligations for the existing diagonal-window classifier and seal-uniqueness surface.

Chapter 444

A Concrete Naming Certificate for $\mathsf{CollisionKernel}^{\uparrow}$

The $\mathsf{CollisionKernel}^{\uparrow}$ interface is a BEDC-native finite certificate surface.

The $\mathsf{CollisionKernel}^{\uparrow}$ interface names a finite collision-kernel packet built from the same finite-window discipline as $\mathsf{FoldMomentKernel}^{\uparrow}$, $\mathsf{GoldenMeanShift}^{\uparrow}$, and the Zeckendorf finite-word family. It records a displayed window, fold-source rows, target-fiber ledgers, a finite collision-kernel matrix row, moment-index rows, Hankel-shadow rows, transport rows, continuation routes, package provenance, and local naming evidence. It exports no host spectrum, ambient measure, infinite Hankel operator, quotient fiber, or analytic moment problem.

The dependency surface is concrete: $\mathsf{CollisionKernel}^{\uparrow}$ consumes the folded-fiber moment packet of chapter 384, the finite binary-window discipline of chapter 370, and the normal-form finite-word rows of chapter 278. Thus W, F, L, M, I, S are collision-kernel refinements of those sibling carriers rather than isolated kernel-only rows.

Definition 444.1 (CollisionKernel BHist carrier)**.** *A* $\mathsf{CollisionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, F, L, M, I, S, H, C, P, N)$$

with the following visible rows:

1. *W is the finite window row naming the inspected source addresses and target addresses;*
2. *F is the fold-source row listing the finite source entries whose target fibers are inspected;*
3. *L is the target-fiber ledger, with each retained source entry attached to a displayed target address in W;*
4. *M is the collision-kernel matrix row whose entries are finite ledger reads comparing pairs of target fibers inside W;*
5. *I is the moment-index row naming the finite family of moment requests consumed by downstream readers;*
6. *S is the Hankel-shadow row obtained by reading the visible moment-index rows against the finite matrix row;*
7. *H stores the componentwise* hsame *transports for window, fold-source, target-fiber ledger, matrix, moment-index, and Hankel-shadow rows;*
8. *C stores the displayed* Cont *routes through which consumers read W, F, L, M, I, and S;*

9. *P is the* Pkg *provenance over* $\mathsf{FoldMomentKernel}^{\uparrow}$, $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$ *row.*

The displayed transport, continuation, package, and naming surface is

$$\mathsf{hsame}(W, F, L, M, I, S, H)$$

$$\mathsf{Cont}(W, F, L) \quad \mathsf{Cont}(L, M, S) \quad \mathsf{Cont}(I, M, S)$$

$$\mathsf{Pkg}(P; \mathsf{FoldMomentKernel}^{\uparrow}, \mathsf{GoldenMeanShift}^{\uparrow}, \mathsf{ZeckendorfNormal}^{\uparrow}, \mathsf{List}^{\uparrow}, \mathsf{Nat}^{\uparrow})$$

$$\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}(N; W, F, L, M, I, S, H, C, P).$$

The carrier has no coordinate for a host collision-counting operation, an ambient spectrum, an external measure, an infinite Hankel matrix, or quotient fibers.

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.TasteGate.CollisionKernelTasteGate single carrier alignment`

Definition 444.2 (CollisionKernel classifier)**.** *For accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carriers K and K', the classifier* $\mathsf{CollisionKernelClassifier}(K, K')$ *compares only the displayed finite window, fold-source, target-fiber ledger, collision-kernel matrix, moment-index, Hankel-shadow, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows of Definition 444.1.*

The classifier does not inspect hidden source sets, host matrix entries, analytic spectra, measure spaces, quotient fibers, or unrecorded moment laws. Two packets classify together only through displayed transport of the same finite collision-kernel surface.

Lean4 (definition) `BEDC.Derived.CollisionKernelUp.CollisionKernelClassifier`

Theorem 444.3 (CollisionKernel zero-window packet)**.** *The empty source window with the zero target window determines an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carrier. Its fold-source row is empty, its target-fiber ledger has no retained source entries, its collision-kernel matrix row is the displayed zero finite matrix, every moment-index read is represented by the zero row, the Hankel-shadow row is zero, and all transport, continuation, package, and local naming rows are the corresponding finite identity rows.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier zero window packet`

Proof. Start with the empty displayed window. There are no source entries to fold and no target fiber entries to compare, so the target-fiber ledger is empty.

Use the displayed zero finite matrix as the collision-kernel row. Since every inspected matrix entry is a read against the empty ledger, each moment-index request also reads the zero row.

The Hankel-shadow row is therefore the zero shadow over the empty finite matrix. Repack the identity hsame transports, identity Cont routes, package provenance, and local naming row to obtain the accepted carrier. □

Theorem 444.4 (CollisionKernel NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier inhabitation: Definition 444.3 supplies the empty-source and zero-window packet;*

2. *classifier transport: Definition 444.2 transports the finite window, fold-source row, target-fiber ledger, matrix row, moment-index row, Hankel-shadow row,* hsame *routes,* Cont *routes, package provenance, and local naming row componentwise;*

3. *fiber-ledger exactness: every matrix entry is read from the displayed target-fiber ledger L, and every ledger entry is tied to the displayed fold-source row F and finite window row W;*
4. *moment-readback boundary: every moment or Hankel-shadow read factors through I, M, S, L, H, C, P, and N before reaching any downstream consumer;*
5. *non-escape from host spectra or measure: the packet exports no host spectral object, measure space, infinite Hankel operator, quotient fiber, ambient moment problem, or unrecorded collision-counting operation.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier namecert obligation surface`

Proof. The carrier row is the finite packet of Definition 444.1, and Definition 444.3 gives a concrete inhabitant.

Classifier transport is componentwise by Definition 444.2. A displayed hsame route can move only the visible window, fold-source, target-fiber ledger, matrix, moment-index, Hankel-shadow, continuation, package, and naming rows already present in the packet.

Fiber-ledger exactness is obtained by projecting L together with F and W. Every matrix entry in M is a finite read from those displayed target fibers, and every retained source entry remains attached to the fold-source and window rows by C.

Moment readback projects I, M, and S. The Hankel-shadow consumer reaches a visible shadow only through the moment-index row, the collision-kernel matrix row, the fiber ledger, transports, continuations, package provenance, and the local certificate.

Since those rows exhaust the carrier coordinates, no host spectrum, measure object, infinite Hankel operator, quotient fiber, ambient moment problem, or hidden counting operation can be exported. □

Theorem 444.5 (CollisionKernel Hankel-shadow boundary)**.** *Every Hankel-shadow read of an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carrier factors through the finite moment-index row I, the displayed collision-kernel matrix row M, the target-fiber ledger L, the componentwise* hsame *transports H, the displayed* Cont *routes C, the package row P, and the local* $\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$ *row N. The read cannot be strengthened into an infinite Hankel operator, host spectral value, measure-theoretic moment functional, quotient-fiber equality, or external analytic boundary condition.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier hankel shadow boundary`

Proof. Project the accepted carrier. The only rows mentioning a Hankel shadow are the finite moment-index row I, the matrix row M, and the shadow row S.

Read M through the target-fiber ledger L. By Definition 444.4, every matrix entry is exhausted by displayed ledger entries over the finite window and fold-source rows.

Now transport the requested shadow read through H and route it through C. Package provenance P and the local naming row N certify that the read remains inside the same finite packet.

The carrier has no coordinate for an infinite Hankel operator, host spectral value, ambient measure, quotient fiber, or analytic moment functional, so the Hankel-shadow read stops at the displayed finite boundary. □

Theorem 444.6 (CollisionKernel classifier transport scope)**.** *For accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carriers, classified replacement transports only the finite window row W, fold-source row F, target-fiber ledger L, collision-kernel matrix row M, moment-index row I, Hankel-shadow row S, componentwise* hsame *row H, displayed* Cont *routes C, package row P, and local* NameCert *row N. It has no*

transport coordinate for hidden source sets, host matrix entries, analytic spectra, ambient measures, quotient fibers, or unrecorded moment laws.

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier classifier transport scope`

Proof. Project the carrier in Definition 444.1. The packet lists exactly the rows W, F, L, M, I, S, H, C, P, and N.

Apply the classifier definition in Definition 444.2. Its comparison relation inspects only those displayed rows and the componentwise transport already stored in the carrier.

Use the classifier-transport row of Definition 444.4. It moves the finite window, fold-source, target-fiber ledger, matrix, moment-index, Hankel-shadow, transport, continuation, package, and local naming rows componentwise.

Finally apply Definition 444.5. Since Hankel reads themselves stop at the same finite boundary, classified replacement cannot export a hidden source set, host entry, analytic spectrum, measure object, quotient fiber, or unrecorded moment law. □

Theorem 444.7 (CollisionKernel fiber-ledger exactness)**.** *Every matrix entry in the collision-kernel row M of an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carrier is read from the displayed target-fiber ledger L, and each such ledger entry is tied to the fold-source row F and the finite window row W. The matrix row has no independent host collision-counting operation, hidden target fiber, quotient fiber, ambient spectrum, or measure-theoretic source.*

Proof. Begin with Definition 444.1. The matrix row M is part of the finite packet and is presented together with the target-fiber ledger L, fold-source row F, and finite window row W.

The NameCert obligation surface in Definition 444.4 contains the fiber-ledger exactness row. It states that every matrix entry is a finite read from displayed target fibers, with every retained source entry attached to F and W.

Use Definition 444.2 to keep the comparison componentwise across the same visible rows. Then apply Definition 444.5; any downstream Hankel read of M passes back through L before reaching S, H, C, P, or N.

Thus M is exhausted by the displayed ledger tied to F and W, with no independent host count, hidden fiber, quotient fiber, spectrum, or measure object. □

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier fiber ledger exactness`

Theorem 444.8 (CollisionKernel moment readback exhaustion)**.** *Every moment read and every Hankel-shadow read exported by an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *carrier factors through the displayed rows I, M, S, L, H, C, P, and N. The readback exports no ambient spectrum, measure object, infinite Hankel operator, quotient fiber, hidden counting operation, or unrecorded analytic moment functional.*

Proof. Read the moment-facing coordinates from Definition 444.1. Moment requests are named by I, evaluated against the displayed matrix row M, recorded in the Hankel-shadow row S, and tied back to the target-fiber ledger L.

Apply Definition 444.4. Its moment-readback boundary routes every moment or Hankel-shadow read through the visible moment, matrix, shadow, ledger, transport, continuation, package, and naming rows before a downstream consumer sees it.

Use Definition 444.5 to identify the same finite boundary for Hankel-shadow consumers. The classifier in Definition 444.2 supplies only componentwise transport across those rows.

Consequently the readback surface is exhausted by I, M, S, L, H, C, P, and N, and it has no coordinate for spectrum, measure, infinite Hankel structure, quotient fibers, hidden counting, or analytic moment laws. □

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier moment readback exhaustion`

Theorem 444.9 (CollisionKernel zero-window determinacy)**.** *Let $K = (W, F, L, M, I, S, H, C, P, N)$ and $K' = (W, F, L, M, I', S', H', C', P', N')$ be accepted zero-window* $\mathsf{CollisionKernel}^{\uparrow}$ *packets with the same finite window W, fold-source row F, target-fiber ledger L, collision-kernel matrix row M, and moment index whenever a moment read is requested. Then the two packets are classified by Definition 444.2; their visible collision-kernel output is determined by those shared finite rows.*

In particular, zero-window habitation cannot create a second classifier through hidden source sets, host matrix entries, quotient fibers, ambient measures, spectra, or unrecorded moment laws.

Proof. Start from the zero-window packet of Definition 444.3. It supplies an accepted finite packet whose source and target-fiber data are the displayed zero-window rows.

Project the classifier surface in Definition 444.2. The classifier inspects the finite window, fold source, target-fiber ledger, matrix row, moment-index row, Hankel-shadow row, transport row, continuation row, package provenance, and local naming row.

Because K and K' share W, F, L, M and the requested moment index, the non-shared rows can only replay the same displayed continuation and naming routes. The finite classifier therefore identifies the two zero-window packets on the shared ledger.

Finally use Definition 444.8. Any moment-facing read returns to I, M, S, L, H, C, P, N, so it cannot separate the two packets by an external source. □

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier zero window determinacy`

Theorem 444.10 (CollisionKernel fiber classifier composition)**.** *For accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *packets, componentwise* hsame *transports of the target-fiber ledger L compose through the displayed* Cont *rows and preserve the classifier of Definition 444.2. The composed classifier route still reads only the finite window W, fold-source row F, target-fiber ledger L, matrix row M, moment-index row I, Hankel-shadow row S, transport row H, continuation row C, package row P, and local* $\mathsf{NameCert}$ *row N.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier fiber classifier composition`

Proof. First isolate the ledger. By Definition 444.7, every matrix entry in M is read from the displayed target-fiber ledger L and tied back to the finite window W and fold-source row F.

Next compose the transports. The classifier-transport scope in Definition 444.6 permits only componentwise hsame movement along the displayed rows and only the displayed Cont routes for replaying consumers.

The classifier definition then has no further input to inspect. Applying Definition 444.2 after the composed ledger transport yields the same finite classification, and Definition 444.5 keeps any shadow read on the same ledger boundary. □

Theorem 444.11 (CollisionKernel scoped dependency route)**.** *Every scoped* $\mathsf{CollisionKernel}^{\uparrow}$ *consumer must read the finite window and target-fiber ledger through* $\mathsf{FoldMomentKernel}^{\uparrow}$, $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, *and* $\mathsf{Nat}^{\uparrow}$ *provenance before any Hankel-shadow or moment-index row is exported. The route has no host spectrum, ambient measure, quotient fiber, or infinite Hankel operator.*

Proof. Project the carrier of Definition 444.1. The rows $W, F, L, M, I, S, H, C, P, N$ are the only displayed coordinates, and the provenance row P names $\mathsf{FoldMomentKernel}^{\uparrow}$, $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ as the finite source route for scoped reads.

Use fiber-ledger exactness from Definition 444.7. Every matrix entry reaches the target-fiber ledger L through the finite window W and fold-source row F, so a consumer cannot read a target fiber independently of the sibling provenance.

Apply moment readback exhaustion and classifier transport scope from Definition 444.8 and Definition 444.6. Moment-index and Hankel-shadow rows are exported only after I, M, S, L, H, C, P, N have preserved the same finite packet.

Thus the sibling dependency rows exhaust the scoped route before any moment or shadow read is visible. Since the carrier contains no spectrum, measure, quotient fiber, or infinite Hankel coordinate, no such object is exported by the route. □

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier scoped route`

Theorem 444.12 (CollisionKernel moment ledger noninterference)**.** *In an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *packet* $K = (W, F, L, M, I, S, H, C, P, N)$*, a moment read or Hankel-shadow read may alter only the displayed moment-index row* I *or Hankel-shadow row* S *of the consumer route. It cannot alter the finite source window* W*, fold-source row* F*, matrix row* M*, package provenance* P*, or local* $\mathsf{NameCert}$ *row* N*.*

Thus moment-facing readback is noninterfering with the finite source ledger and the local naming surface of the collision-kernel packet.

Proof. Read the moment-facing rows from Definition 444.8. Every moment and Hankel-shadow request factors through I, M, S, L, H, C, P, N before it is exported.

The source side is already fixed by Definition 444.7. Matrix entries in M come from the target-fiber ledger L tied to F and W, so a moment read has no coordinate through which to rewrite the finite source window or fold-source row.

The naming side is fixed by Definition 444.6. Transport preserves only the displayed rows and replayed continuations; it does not replace package provenance or the local $\mathsf{NameCert}$ row.

Therefore a moment or Hankel-shadow consumer can vary only the moment-facing index or shadow rows of the read, while W, F, M, P, N remain the same visible ledger data. □

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelMomentLedgerNoninterference`

Theorem 444.13 (CollisionKernel matrix readback totality)**.** *Every matrix read of an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *packet* $K = (W, F, L, M, I, S, H, C, P, N)$ *is exhausted by the displayed rows* W, F, L, M, H, C, P, N*. More precisely, a consumer can reach an entry of* M *only as a finite read against the target-fiber ledger* L*, tied to the fold-source row* F *and the finite window row* W*, transported by* H*, routed by* C*, and recorded by* P *and* N*.*

Consequently the matrix row exports no host matrix operation, hidden source set, quotient fiber, ambient spectrum, measure object, or analytic collision law.

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier matrix readback totality`

Proof. Start with the carrier of Definition 444.1. The matrix row M is displayed only together with the window W, fold-source row F, target-fiber ledger L, transport row H, continuation row C, package row P, and local naming row N.

Apply Definition 444.7. It identifies every matrix entry as a read from L, with each retained source entry tied to F and W.

Now use Definition 444.6. The only permitted replacement of a matrix read is componentwise transport through the displayed hsame and Cont rows, with provenance and local naming still recorded by P and N.

Since those rows exhaust the available readback coordinates, the matrix read cannot export a host operation, hidden source set, quotient fiber, spectrum, measure, or analytic collision law. □

Theorem 444.14 (CollisionKernel downstream non-escape package)**.** *Every downstream Hankel-shadow or moment consumer of an accepted* $\mathsf{CollisionKernel}^{\uparrow}$ *packet factors through the finite rows* I, M, S, L, H, C, P, N *and then back through the matrix-readback boundary of Definition 444.13. Such a consumer cannot export an ambient spectrum, measure space, quotient fiber, infinite Hankel operator, analytic moment law, or hidden collision-counting operation.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier downstream non escape package`

Proof. First isolate the downstream read. By Definition 444.8, every moment or Hankel-shadow request factors through I, M, S, L, H, C, P, N.

Next identify the matrix component of that read. Definition 444.13 sends every access to M back to the displayed window, fold-source row, target-fiber ledger, transports, continuations, package provenance, and local naming row.

Finally apply Definition 444.5. A Hankel-shadow consumer stops at the same finite packet and has no coordinate for an infinite Hankel operator or analytic moment functional.

Thus every downstream read remains a finite consumer of the displayed packet and cannot export spectra, measures, quotient fibers, hidden counting, or analytic moment laws. □

Theorem 444.15 (CollisionKernel obligation closure package)**.** *The paper-side* $\mathsf{CollisionKernel}^{\uparrow}$ *obligation package consists of the zero-window inhabitant, classifier transport scope, fiber-ledger exactness, matrix readback totality, moment and Hankel-shadow readback exhaustion, and downstream non-escape package. Together these clauses bind the accepted finite carrier* $K = (W, F, L, M, I, S, H, C, P, N)$ *as a NameCert surface over displayed windows, fold-source rows, target-fiber ledgers, matrix rows, moment-index rows, Hankel-shadow rows, transports, continuations, package provenance, and local naming evidence.*

Lean4 ✓ `BEDC.Derived.CollisionKernelUp.CollisionKernelCarrier obligation closure package`

Proof. The inhabitant is Definition 444.3. It supplies a concrete accepted packet with empty source data, zero target window, zero matrix row, zero moment rows, and identity transport and naming data.

Classifier transport is Definition 444.6. It restricts classified replacement to the displayed rows of the carrier and excludes hidden source sets, host entries, analytic spectra, ambient measures, quotient fibers, and unrecorded moment laws.

Fiber exactness and readback are supplied by Definition 444.7, Definition 444.13, and Definition 444.8. These theorems force matrix, moment, and Hankel-shadow consumers to return to the finite ledger rows $W, F, L, M, I, S, H, C, P, N$.

Downstream non-escape is Definition 444.14. Repacking these clauses gives the finite paper-side obligation package for the $\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CollisionKernel}^{\uparrow}$ packet is generated from a finite window, fold-source rows, target-fiber ledgers, a finite collision-kernel matrix row, moment-index rows, a Hankel-shadow row, componentwise transport, continuation routes, package provenance, and local naming evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CollisionKernel}^{\uparrow}}$, obligation)

Scope closed: The obligation scope binds the finite carrier, classifier transport, fiber ledger exactness, moment readback, matrix readback, downstream non-escape, and obligation package for finite collision-kernel packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CollisionKernel}\uparrow}, \text{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic spectra, ambient measure theory, infinite Hankel operators, quotient fibers, asymptotic collision laws, or bridge checking.

Upgrade path: Scoped closure requires explicit dependency binding across carrier admission, classifier transport, ledger exactness, readback exhaustion, and downstream non-escape for the same finite packet.

Chapter 445

A Concrete Naming Certificate for $\mathsf{DyadicMidpoint}^{\uparrow}$

The $\mathsf{DyadicMidpoint}^{\uparrow}$ interface names the finite midpoint ledger consumed by dyadic interval bisection. In the Bishop–Bridges style of constructive dyadic interval bisection, a midpoint is not a hidden real-number choice: it is a displayed common-denominator calculation over two dyadic endpoint rows, followed by a branch-comparison row and the finite readback route made available to interval consumers.

The midpoint packet consumes endpoint and tolerance rows from chapter 291 and routes branch decisions to chapter 354, chapter 306, and chapter 320. The common-denominator scaling and midpoint rows are $\mathsf{DyadicRatCore}^{\uparrow}$ rows, while the interval chapters consume the branch-comparison row through the same finite branch ledger before it reaches $\mathsf{DyadicSubdivision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or later L10 real-readback consumers.

Definition 445.1 (DyadicMidpoint carrier)**.** *A* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, S, \mu, B, W, H, C, P, N)$$

with the following visible rows:

1. *L and R are the left and right endpoint rows accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$ *or exposed through a* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint surface;*
2. *S is the common-denominator scaling row that transports L and R to one dyadic denominator before addition;*
3. *μ is the midpoint row obtained from the displayed numerator sum and the doubled common denominator;*
4. *B is the branch-comparison row that records which side of μ a later bisection or locatedness consumer selects;*
5. *W is the finite-window readback row exposing the scaled endpoints, midpoint, branch comparison, and requested dyadic tolerance;*
6. *H stores the componentwise* hsame *transports for the two endpoints, the scaling row, the midpoint row, the branch row, and the finite readback row;*
7. *C stores the displayed* Cont *routes for endpoint inspection, midpoint readback, branch handoff, and downstream interval consumption;*

8. *P is the* Pkg *provenance row naming* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicSubdivision}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* NameCert *sources that may consume the midpoint ledger;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpoint}^{\uparrow}}$ *registration row.*

The carrier has no coordinate for host real equality, quotient endpoints, a completed interval intersection, an ambient completeness proof, or an unrecorded midpoint decision.

Lean4 (definition) `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCarrier`

Theorem 445.2 (DyadicMidpoint endpoint-source route)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *packet*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

the endpoint rows are read first as L and R through chapter 291 or through the rational interval endpoint surface of chapter 354. Only after those two endpoint rows and the common-denominator row S are present does the packet expose the midpoint row μ and the branch row B to the bisection consumer of chapter 361 through the displayed finite route W, H, C, P, N.

Thus a $\mathsf{DyadicBisection}^{\uparrow}$ *consumer cannot read a midpoint branch without first inheriting the dyadic endpoint data named by the source endpoint rows.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointEndpointSourceRoute`

Proof. Project the accepted carrier as

$$M = (L, R, S, \mu, B, W, H, C, P, N).$$

The first two coordinates are the only endpoint source rows in the packet. By Definition 445.1, they are accepted by $\mathsf{DyadicRatCore}^{\uparrow}$ or exposed through a $\mathsf{RationalInterval}^{\uparrow}$ endpoint surface.

The common-denominator row S is the only row that transports those two endpoint rows to a shared dyadic denominator. The midpoint row μ is then read from that displayed scaled endpoint ledger, so the midpoint calculation has no independent endpoint source.

The branch row B is the only coordinate that records a side selection at the midpoint. Hence a $\mathsf{DyadicBisection}^{\uparrow}$ consumer receives the branch through B and the finite route W, H, C, P, N, with L, R, S, μ as the displayed prerequisites.

No remaining coordinate of the carrier can supply host real equality, quotient endpoints, completed intersections, or ambient completeness. The coordinate list is exhausted by $L, R, S, \mu, B, W, H, C, P, N$, so the endpoint-source route into bisection is exactly the stated finite route. □

Theorem 445.3 (DyadicMidpoint NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicMidpoint}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier: formation of the* BHist *packet of Definition 445.1;*

2. *endpoint scaling: both endpoint rows are transported to the displayed common denominator before any midpoint row is read;*

3. *midpoint exactness: the row μ is the finite dyadic row obtained from the displayed scaled numerator sum and doubled common denominator;*

4. *branch comparison: the branch row B records the side of the displayed midpoint consumed by dyadic bisection, subdivision, or located-interval readback;*

5. *finite-window readback: every consumer read factors through W, H, C, P, and N;*

6. *non-escape: no host real equality, quotient endpoint, completed intersection, or ambient completeness row is exported.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointBranchRow consumer handoff`

Proof. Carrier formation is the finite packet assembly of Definition 445.1. The two endpoint rows, the common-denominator row, the midpoint row, the branch row, the readback row, and the provenance rows are all visible coordinates of the same BHist packet.

Endpoint scaling and midpoint exactness are read from S and μ. The scaling row names the common dyadic denominator, and the midpoint row is built by the displayed numerator-sum and denominator-doubling calculation.

Branch comparison is the row B. A bisection or locatedness consumer may route left or right only through that displayed comparison row, so the branch decision is not imported from an external real-order oracle.

Finally, every public read must pass through W, then through the displayed hsame and Cont routes and the package and naming rows. Since the carrier lists no host equality, quotient endpoint, completed intersection, or completeness coordinate, none is available to export. □

Theorem 445.4 (DyadicMidpoint endpoint classifier stability)**.** *Let*

$$M = (L, R, S, \mu, B, W, H, C, P, N)$$

and

$$M' = (L', R', S', \mu', B', W', H', C', P', N')$$

be accepted DyadicMidpoint$^{\uparrow}$ *carriers. If the endpoint rows L, R and L', R' are transported by the displayed* hsame *and* Cont *rows to the same finite dyadic endpoint window, then their common-denominator rows, midpoint rows, branch-comparison rows, and finite-window readbacks classify together through the local* DyadicMidpoint$^{\uparrow}$ *classifier.*

Proof. First transport the two endpoint pairs along the displayed endpoint routes in H, C and H', C'. The hypothesis places both carriers over the same finite dyadic endpoint window.

The common-denominator rows S and S' are then read over that shared window. Because both rows scale the same two transported endpoint rows, the scaled endpoint numerators and common denominator are the same finite ledger data.

The midpoint rows μ and μ' are obtained from those scaled endpoint numerators by the same displayed denominator-doubling calculation. Thus the midpoint readbacks classify together.

It remains to compare consumer-visible branching and readback. The branch rows B, B' inspect the same midpoint row against the same transported endpoint window, and W, W' expose only those transported rows through the package and naming provenance. The local classifier therefore preserves the full midpoint packet surface. □

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCarrier endpoint classifier stability`

Theorem 445.5 (DyadicMidpoint branch comparison handoff)**.** *For every accepted* DyadicMidpoint$^{\uparrow}$ *carrier $M = (L, R, S, \mu, B, W, H, C, P, N)$, any* DyadicBisection$^{\uparrow}$, DyadicSubdivision$^{\uparrow}$, *or* LocatedInterval$^{\uparrow}$ *consumer that branches at the midpoint consumes exactly the displayed branch-comparison row B together with the endpoint rows L, R, the common-denominator row S, the midpoint row μ, the finite readback row W, and the provenance rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointBranchComparisonHandoff`

Proof. Project the carrier. The only row that records a left/right comparison at the midpoint is B, and the only midpoint value available to that row is μ.

The endpoint and scaling prerequisites for the comparison are exactly L, R and S. Reading B therefore forces the consumer to read the same common-denominator endpoint window and the same midpoint row that produced the comparison.

The consumer-facing handoff is W. It exposes the endpoint, scaling, midpoint, and branch rows through the displayed transports, continuation routes, package provenance, and local naming row.

Thus the bisection, subdivision, or located-interval consumer receives the stated finite handoff and no hidden branch oracle. □

Theorem 445.6 (DyadicMidpoint finite-window consumer route)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier and every finite precision request, the consumer-visible route to* $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the finite readback row* W*, the displayed* hsame *transports* H*, the displayed* Cont *routes* C*, the package row* P*, and the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpoint}^{\uparrow}}$ *row* N*.*

Proof. Start with the finite precision request. The carrier answers it by selecting the finite readback row W, which contains the scaled endpoints, midpoint row, branch comparison, and dyadic tolerance visible at that request.

Endpoint and midpoint transport then proceeds through H and C. These rows move the endpoint, scaling, midpoint, and branch data componentwise, so every downstream interval or real-name consumer reads the same finite ledger.

The package row P records which surrounding certificates may consume the route, and N records the local naming registration. Therefore the route into rational intervals, dyadic intervals, nested intervals, finite-window envelopes, dyadic bisection, dyadic rational rows, and real names is exhausted by the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCarrier finite window consumer route`
Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCarrier readback closure`

Theorem 445.7 (DyadicMidpoint nested interval non-escape)**.** *Suppose a* $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{DyadicBisection}^{\uparrow}$ *packet consumes a* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier during a finite refinement step. The retained subinterval endpoint rows, the selected midpoint row, and the branch comparison remain inside the displayed* BHist *packet of Definition 445.1; the handoff exports no host real equality, no quotient endpoints, no completed interval intersection, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointNestedIntervalNonEscape`

Proof. Apply Definition 445.5. A refinement step can choose a retained subinterval only through the displayed branch row B and the midpoint row μ.

Next apply Definition 445.6. The selected endpoint rows and midpoint row are routed to the nested or bisection consumer through W, H, C, P, and N.

All rows used in the handoff are coordinates of the finite carrier or provenance attached to that carrier. Since Definition 445.1 contains no host real equality, quotient endpoint, completed interval intersection, or ambient completeness field, the finite refinement cannot export any such object. □

Theorem 445.8 (DyadicMidpoint carrier obligation surface)**.** *For an accepted* DyadicMidpoint$^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N)$$

over BHist*, the six rows of Definition 445.3 assemble a single carrier obligation surface. The carrier row is the packet of Definition 445.1; endpoint scaling and midpoint exactness are the S and μ rows; branch comparison is the B row; finite-window readback is the route through W, H, C, P, N; and non-escape is the absence of host real equality, quotient endpoint, completed interval intersection, and ambient completeness coordinates. This surface is stated entirely over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* DyadicRatCore$^{\uparrow}$*,* RationalInterval$^{\uparrow}$*,* DyadicInterval$^{\uparrow}$*, and* StreamName$^{\uparrow}$ *rows.*

Proof. Project the carrier in the displayed order. The first coordinate group gives the endpoint rows L, R, the common-denominator row S, the midpoint row μ, the branch row B, and the finite-window row W.

The transport and route rows are already part of the same packet. The componentwise hsame row H and the Cont row C carry the endpoint, scaling, midpoint, branch, and readback data without adding a second carrier.

The package and naming rows then bind the consumer side. The Pkg row P names the DyadicRatCore$^{\uparrow}$, RationalInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, and StreamName$^{\uparrow}$ surfaces that may read the packet, while N supplies the local NameCert registration.

Finally, Definition 445.1 lists no host real equality, quotient endpoint, completed interval intersection, or ambient completeness coordinate. Hence the six obligations form one finite obligation surface over exactly the stated rows. □

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCarrier obligation surface`

Theorem 445.9 (DyadicMidpoint bisection ledger exactness)**.** *Every bisection or subdivision read that retains an endpoint and compares against the displayed midpoint of an accepted* DyadicMidpoint$^{\uparrow}$ *carrier is exhausted by the rows*

$$L, R, S, \mu, B, W, H, C, P, N.$$

The retained endpoint is one of the endpoint rows or the midpoint row exposed through W; the midpoint comparison is the branch row B; and every transport, continuation, package, and naming dependency is one of H, C, P, N.

Proof. Start with the retained endpoint read. The carrier exposes only L, R as original endpoints and μ as the computed midpoint, so a retained subinterval endpoint must be read from those rows through the finite-window route W.

The comparison used to choose the branch is the row B. Its prerequisites are the displayed endpoint rows and the common-denominator row S, because the midpoint row μ is computed from that scaled ledger.

It remains to account for transport and consumer routing. The hsame row H and Cont row C move the same finite rows componentwise, and P, N record the package and local naming provenance. No other row is present in the carrier, so the bisection or subdivision ledger is exact. □

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointBisectionLedgerExactness`

Theorem 445.10 (DyadicMidpoint branch-ledger totality)**.** *For every accepted* DyadicMidpoint$^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

every bisection branch read is accounted for by the displayed branch row B together with the endpoint rows L, R, common-denominator row S, midpoint row μ, finite-window row W, transport row H, continuation row C, package row P, and local naming row N. No bisection consumer can obtain a branch outside this ledger.

The branch surface exports no host real equality, quotient endpoints, completed interval intersection, or completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointBranchLedgerTotality`

Proof. Project the carrier rows of Definition 445.1. The only branch-bearing coordinate is B, and its displayed prerequisites are the endpoint rows L, R, the common-denominator row S, and the midpoint row μ.

Apply Definition 445.5. It identifies every bisection or subdivision branch with the displayed branch-comparison row and routes the branch back through the same endpoint and midpoint ledger.

Finally use Definition 445.9. The finite-window, transport, continuation, package, and naming rows W, H, C, P, N exhaust the consumer-visible route, so there is no remaining coordinate for host real equality, quotient endpoints, completed interval intersection, or a completeness principle. □

Theorem 445.11 (DyadicMidpoint common-denominator consumer exactness)**.** *For every accepted* DyadicMidpoint$^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

any consumer that reads the midpoint row μ must first consume the displayed common-denominator row S for the endpoint rows L, R. The finite-window route W, componentwise hsame *row H, displayed* Cont *row C, package row P, and local naming row N preserve that same denominator ledger when the midpoint is handed to* RationalInterval$^{\uparrow}$*,* DyadicInterval$^{\uparrow}$*,* NestedInterval$^{\uparrow}$*,* StreamName$^{\uparrow}$*, or* Real$^{\uparrow}$ *consumers.*

Thus midpoint exactness is the exactness of the displayed common-denominator calculation, not equality of host real endpoints, quotient endpoint representatives, completed interval intersections, or any completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCommonDenominatorConsumerExactness`

Proof. Start from the carrier definition. The midpoint row μ is not a primitive endpoint; it is obtained from the endpoint rows L, R after the common-denominator scaling row S has been read.

Use Definition 445.9. It states that every bisection or subdivision read is exhausted by the rows $L, R, S, \mu, B, W, H, C, P, N$, so a midpoint consumer cannot bypass S.

Apply Definition 445.6. The finite-window route and the transport, continuation, package, and naming rows carry the same scaled endpoint ledger into downstream interval and real-name consumers.

The handoff therefore preserves exactness of the common-denominator calculation only. Since no host real equality, quotient endpoint, completed intersection, or completeness row is present in the carrier, none is exported. □

Theorem 445.12 (DyadicMidpoint common-denominator branch exhaustion)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

every branch read consumed by $\mathsf{DyadicBisection}^{\uparrow}$*,* $\mathsf{DyadicSubdivision}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *first reads the common-denominator row* S *and then the midpoint row* μ *and branch row* B *through* W, H, C, P, N*. No host real equality, quotient endpoint, completed interval intersection, or completeness principle is available to that consumer.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointCommonDenominatorBranchExhaustion`

Proof. Project the accepted carrier. The endpoint rows L, R, common-denominator row S, midpoint row μ, branch row B, finite-window row W, transport row H, continuation row C, package row P, and local naming row N are the whole displayed packet.

Apply Definition 445.5 to identify the branch read with the displayed branch-comparison row. Use Definition 445.9 and Definition 445.10 to bind the retained endpoint, midpoint, and branch decision to $L, R, S, \mu, B, W, H, C, P, N$.

Finally apply Definition 445.11. It forces every midpoint consumer to read S before μ, so the branch route available to bisection, subdivision, located-interval, and real-name consumers is exhausted by the common-denominator branch ledger and exports none of the excluded completion or host-equality rows. □

Theorem 445.13 (DyadicMidpoint selected-window denominator seal)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *consumer that reads the selected midpoint window must consume the common-denominator row* S*, midpoint row* μ*, branch row* B*, finite-window row* W*, and provenance rows* H, C, P, N *before any seal-facing read. No quotient endpoint, completed interval intersection, host real equality, or ambient completeness row can mediate that handoff.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointSelectedWindowDenominatorSeal`

Proof. Begin with Definition 445.11. It forces every midpoint read through the displayed common-denominator row S before the midpoint row μ can be consumed.

Apply Definition 445.12. The selected window then reads the branch row B through W, H, C, P, N, so a bisection, subdivision, located-interval, real-name, or stream-name consumer cannot select a midpoint branch outside the finite packet.

Finally use Definition 445.15. The $\mathsf{Real}^{\uparrow}$ seal route receives only the finite dyadic-name packet for the current window, and the same finite-window boundary is the one a $\mathsf{StreamName}^{\uparrow}$ consumer can observe.

The carrier contains no quotient endpoint, completed interval intersection, host real equality, or ambient completeness row. Therefore the selected-window denominator route is the only seal-facing handoff for midpoint consumers. □

Theorem 445.14 (DyadicMidpoint radius-halving window)**.** *Let a* $\mathsf{DyadicInterval}^{\uparrow}$ *row display endpoints* L, R *through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and let a positive dyadic radius budget* ρ *be attached to that interval row. An accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N)$$

splits the displayed interval into two child interval rows with endpoints (L, μ) *and* (μ, R)*. Each child radius budget is bounded by the parent half-radius* $\rho/2$*, and the endpoint, midpoint, branch, and readback rows remain transported through the same* $\mathsf{StreamName}^{\uparrow}$*- and* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible ledger* W, H, C, P, N*.*

The split exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness, no quotient equality of endpoints, and no completed interval intersection.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointRadiusHalvingWindow`

Proof. Read the interval endpoint data through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The common-denominator row S transports L and R to one dyadic denominator, and the midpoint row μ is computed from that scaled ledger by the dyadic midpoint calculation.

The branch row B then displays the two possible retained child rows. The left child uses endpoints (L, μ) and the right child uses (μ, R); both inherit endpoint admission from L, R, S, μ and do not introduce a third endpoint source.

The radius estimate is the finite denominator calculation already visible in the midpoint ledger. Since μ is the displayed midpoint of the scaled endpoint row, each child interval has dyadic width bounded by one half of the parent width, so its radius budget is bounded by $\rho/2$.

Finally apply the finite-window route W, H, C, P, N. Those rows are the only consumer-facing readback, transport, continuation, package, and naming rows in the carrier, so the same $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ ledger carries the child endpoint reads. The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, quotient endpoint equality, or completed interval intersections, so none is exported. □

Theorem 445.15 (DyadicMidpoint RealUp handoff without completion)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier*

$$M = (L, R, S, \mu, B, W, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *read of the midpoint ledger factors through the finite-window consumer route* W, H, C, P, N *and through the exact bisection rows* L, R, S, μ, B*. The handoff gives* $\mathsf{Real}^{\uparrow}$ *a finite dyadic-name packet for the current window; it does not export a completed interval intersection, a quotient endpoint equality, host real equality, or a completeness principle.*

The same non-completion boundary is inherited by $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *consumers that route through the midpoint ledger.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointRealHandoffNoncompletion`

Proof. Read the consumer route through Definition 445.6. It sends $\mathsf{Real}^{\uparrow}$ and the interval consumers through the finite readback row W, the transport row H, the continuation row C, the package row P, and the local naming row N.

Next apply Definition 445.9. The endpoint, common-denominator, midpoint, and branch data read by that route are exactly L, R, S, μ, B.

Use Definition 445.5 for the branch side. Any retained-window decision is the displayed branch-comparison row, not a host order or equality fact about completed reals.

Combining these rows gives $\mathsf{Real}^{\uparrow}$ only the finite dyadic-name packet for the current window. The carrier has no completed interval intersection, quotient endpoint equality, host real equality, or completeness coordinate to export, and the same boundary holds for the listed interval and stream-name consumers. □

Theorem 445.16 (DyadicMidpoint downstream DyadicRatCore package)**.** *For every accepted* $\mathsf{DyadicMidpoint}^{\uparrow}$ *carrier, endpoint classifier stability, branch-comparison handoff, finite-window consumer routing, and nested-interval non-escape assemble the downstream* $\mathsf{DyadicRatCore}^{\uparrow}$ *package consumed by* $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readers. The assembled package consists only of the rows* $L, R, S, \mu, B, W, H, C, P, N$ *and exports no host real equality, quotient endpoint, completed interval intersection, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DyadicMidpointUp.DyadicMidpointDownstreamDyadicRatCorePackage`

Proof. First use endpoint classifier stability to keep accepted midpoint packets over the same transported endpoint window. This fixes the endpoint rows, the common-denominator row, and the midpoint row seen by a downstream $\mathsf{DyadicRatCore}^{\uparrow}$ reader.

Next use branch-comparison handoff. It identifies the branch decision with the displayed row B and ties that decision back to L, R, S, μ rather than to an external order oracle.

The finite-window consumer route supplies the package boundary. It factors rational interval, dyadic interval, nested interval, finite-window, dyadic rational, and real-name reads through W, H, C, P, N.

Finally, nested-interval non-escape excludes every coordinate not present in the carrier. Thus the downstream $\mathsf{DyadicRatCore}^{\uparrow}$ package consumed by interval and $\mathsf{Real}^{\uparrow}$ readers is exactly the displayed finite midpoint ledger. □

Theorem 445.17 (DyadicMidpoint scoped support package)**.** *The carrier of Definition 445.1, the six* $\mathsf{NameCert}_{\mathsf{DyadicMidpoint}^{\uparrow}}$ *obligations of Definition 445.3, endpoint classifier stability, branch-comparison handoff, finite-window consumer routing, nested-interval non-escape, the* $\mathsf{Real}^{\uparrow}$ *handoff of Definition 445.15, and the downstream* $\mathsf{DyadicRatCore}^{\uparrow}$ *package of Definition 445.16 assemble one scoped support package for* $\mathsf{DyadicMidpoint}^{\uparrow}$.

This package is stated only over BHist, hsame, Cont, Pkg, NameCert, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It exports the finite midpoint ledger and its consumer routes, but it exports no host real equality, quotient endpoint, completed interval intersection, or completeness row.*

Proof. Project the carrier rows of Definition 445.1. The endpoint rows, common-denominator row, midpoint row, branch row, readback row, transport row, continuation row, package row, and local naming row are the whole displayed packet.

The local obligation surface is Definition 445.3. Endpoint classifier stability keeps transported endpoint windows aligned, branch-comparison handoff identifies the consumer branch with the row B, and finite-window consumer routing factors each downstream read through W, H, C, P, N.

Use nested-interval non-escape to keep bisection and nested-interval refinement inside the displayed packet. Use the $\mathsf{Real}^{\uparrow}$ handoff to record that real-name consumers receive only a finite dyadic-name packet for the current window, and use the downstream $\mathsf{DyadicRatCore}^{\uparrow}$ package to bind the rational, dyadic interval, nested interval, and real-name consumers to the same rows.

The projected rows list no host real equality, quotient endpoint, completed interval intersection, or ambient completeness coordinate. Therefore the cited carrier, obligations, stability, route, non-escape, real-name handoff, and downstream package form the stated scoped support package and export no object outside that dependency surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMidpoint}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMidpoint}^{\uparrow}$ packet is generated from two dyadic endpoint rows, a common-denominator scaling row, a finite midpoint calculation, a branch-comparison row, a finite-window readback row, displayed hsame and Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicMidpoint}^{\uparrow}}$, scopedClosed)

Scope closed: The obligation scope packages the endpoint rows, common-denominator scaling row, midpoint row, branch-comparison row, finite-window readback route, local classifier stability, consumer handoff, nested-interval non-escape boundary, Real$^{\uparrow}$ handoff, and downstream DyadicRatCore$^{\uparrow}$ surface for DyadicMidpoint$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicMidpoint}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host real equality, completed interval intersections, quotient endpoints, ambient completeness, root existence, or real trichotomy.

Upgrade path: Obligation closure requires formal carrier encoding, endpoint classifier stability, branch-comparison handoff checking, finite-window consumer route checking, and nested-interval non-escape checking for DyadicMidpoint$^{\uparrow}$.

Chapter 446

A Concrete Naming Certificate for DyadicCover$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, finite covers over located real data are useful when the finite list, the dyadic mesh, and the coverage witnesses are carried as explicit data. The DyadicCover$^{\uparrow}$ interface packages this information as a finite BEDC certificate for Real and Completion consumers. It depends on DyadicRatCore$^{\uparrow}$, DyadicMesh$^{\uparrow}$, RationalInterval$^{\uparrow}$, RationalBall$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and FiniteWindowEnvelope$^{\uparrow}$, and it does not invoke open-cover compactness, excluded middle, or a choice of subcover from an arbitrary family.

The cover carrier consumes four horizontal concrete-instance surfaces. The dyadic readback surface of chapter 291 supplies the center row K; the mesh-refinement surface of chapter 358 supplies the refinement row Q; the cell boundary of chapter 354 supplies the interval cell row I tied to the radius row R; and the window-envelope consumer route of chapter 365 supplies the finite coverage row W.

Definition 446.1 (DyadicCover carrier)**.** *A* DyadicCover$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, R, I, Q, W, H, C, P, N)$$

with the following visible coordinates:

1. *K is the finite list of dyadic centers, each read through a* DyadicRatCore$^{\uparrow}$ *row;*
2. *R is the finite list of dyadic radii, each routed through a* RationalBall$^{\uparrow}$ *row with rational non-negative radius data;*
3. *I is the finite list of rational interval cells, each routed through a* RationalInterval$^{\uparrow}$ *row and attached to the corresponding center-radius entry;*
4. *Q records the* DyadicMesh$^{\uparrow}$ *mesh level and the finite refinement rows between adjacent mesh levels;*
5. *W is the displayed* FiniteWindowEnvelope$^{\uparrow}$ *window whose points are covered by the interval cells;*
6. *H records the* hsame *transports for centers, radii, interval cells, mesh rows, window bounds, stream-name reads, regular-rational sequence reads, and real rows;*
7. *C records the* Cont *routes by which consumers read center membership, radius bounds, interval containment, mesh refinement, coverage witnesses, and total-bounded handoff rows;*

8. *P records* Pkg *provenance for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}$ *provenance row.*

The carrier accepts exactly those packets whose finite cell list covers the displayed finite window and whose refinement rows keep every later mesh cell inside a named earlier interval or rational ball.

Lean4 (definition) `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket`

Theorem 446.2 (DyadicCover NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}$ *seed surface consists of five finite obligations:*

1. *carrier: the packet of Definition 446.1;*

2. *classifier: componentwise comparison of centers, radii, interval cells, mesh rows, finite-window witnesses, transport rows, continuation routes, package provenance, and local certificate rows;*

3. *refinement stability: displayed* hsame *and* Cont *transport preserves the dyadic center list, rational radii, interval cells, mesh refinement rows, and coverage witnesses;*

4. *coverage exactness: every covered real read factors through a* $\mathsf{StreamName}^{\uparrow}$ *row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation row, a* $\mathsf{Real}^{\uparrow}$ *row, and a finite interval or ball cell in the displayed window;*

5. *ledger policy:* Pkg *and* NameCert *expose only the listed dyadic, rational, real, finite-window, transport, and continuation rows.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket namecert obligations`

Proof. Project the carrier. The finite center list, radius list, interval-cell list, mesh rows, window witnesses, transport rows, continuation routes, package provenance, and local certificate row are exactly the visible coordinates.

The classifier is componentwise over these rows. Transport stability follows by moving each coordinate through the stored hsame rows and then reading it through the corresponding Cont route.

Coverage exactness follows from the explicit witness attached to each point in the displayed finite window: the witness names the stream read, regular rational approximation, real row, and interval or ball cell used for that point. The package and certificate rows contain no arbitrary open family or subcover selector, so the ledger policy is exhausted by the displayed packet. □

Theorem 446.3 (DyadicCover finite refinement)**.** *Let D be an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *carrier and let Q' be a displayed later* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement row of its mesh coordinate Q. The refined packet is obtained by replacing each old cell with the finite list of named dyadic subcells routed by Q', transporting the corresponding rational interval and rational ball rows through* hsame*, and retaining the same* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *coverage witnesses after refinement.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite refinement`

Proof. Read the mesh coordinate Q and the displayed refinement row Q'. Each old dyadic cell has a finite named list of subcells, so the refined center, radius, and interval lists are built by finite list replacement.

Transport each subcell's rational interval and rational ball rows through the stored hsame entries, then read the containment route through Cont. The finite-window coordinate W is unchanged, while each old coverage witness now points to the named refined subcell contained in the old cell.

Repack the refined rows with the same Pkg provenance and local $\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}$ row. No infinite cover or selected subcover is introduced, because every operation is over the displayed finite mesh lists. □

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite refinement transport`

Theorem 446.4 (DyadicCover total-bounded handoff)**.** *Every total-bounded or compact-metric consumer of an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *carrier factors through the displayed finite list of dyadic centers, rational radii, rational interval cells,* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement rows,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *coverage witnesses, and the* Cont *routes that read* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data into those cells. The handoff is a finite packet handoff, not an appeal to open-cover compactness.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket total bounded handoff export`
Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket total bounded handoff`

Proof. Project the carrier and choose a consumer route. The only data addressed to total-bounded consumers are the finite center list, radius list, interval-cell list, mesh refinement rows, and coverage witnesses over the displayed window.

Follow the real-data read route. A covered point is read through its $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows into a $\mathsf{Real}^{\uparrow}$ row, then into the named rational interval or rational ball cell supplied by the coverage witness.

Thus the consumer receives exactly a finite cover packet with explicit mesh and radius data. Since the carrier contains no arbitrary indexed open family, no compactness principle and no choice of subcover can be projected from it. □

Theorem 446.5 (DyadicCover total-bounded bridge)**.** *An accepted* $\mathsf{DyadicCover}^{\uparrow}$ *packet over finite dyadic centers, rational radii, rational interval cells, mesh refinement rows, and* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *coverage witnesses supplies exactly the finite center-radius witness surface required by a* $\mathsf{TotallyBounded}^{\uparrow}$ *consumer. The bridge exports the listed centers, radii, cells, window incidences, refinement rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row, and exports no arbitrary open family or subcover selector.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket total bounded bridge`

Proof. Begin with the total-bounded handoff. It identifies the consumer-facing data as the finite center list, radius list, interval-cell list, mesh refinement rows, finite-window coverage witnesses, and real-data read routes.

Apply window-cell incidence. Each point covered inside the displayed window is attached to one listed center, one rational radius, one rational interval cell, and the corresponding stream, regular-rational, and real read route.

Use refinement witness exhaustion to account for later mesh rows. Every refined witness is consumed by finite replacement along the displayed refinement rows, and ledger non-escape prevents the bridge from exporting an open-cover compactness principle or a selected subcover. □

Theorem 446.6 (DyadicCover finite-window total-bounded correspondence)**.** *For an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *packet, the finite-window incidence rows and the* $\mathsf{TotallyBounded}^{\uparrow}$ *center-radius rows*

determine the same consumer-visible cover package. A consumer may read the package through FiniteWindowEnvelope$^{\uparrow}$ *cells or through finite center-radius rows, but both routes use the same dyadic centers, rational radii, interval cells, mesh refinement rows, coverage witnesses, transports, continuation routes, provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite window total bounded correspondence`

Proof. Read the finite-window side first. Window-cell incidence assigns every covered point in the displayed window to one named center, radius, interval cell, and real-data route.

Read the total-bounded side next. The total-bounded bridge presents the same listed centers and radii as the finite witness surface required by the consumer, with the same interval cells and mesh rows.

Finally compare the two routes through refinement witness exhaustion and ledger non-escape. Refinement changes only the displayed finite cells along named mesh rows, and no route introduces an arbitrary open family, subcover selector, quotient real equality, or ambient compactness theorem. □

Theorem 446.7 (DyadicCover ledger nonescape)**.** *Every* Real$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, *total-bounded, or completion consumer of an accepted* DyadicCover$^{\uparrow}$ *carrier factors through the displayed finite dyadic centers, dyadic radii, rational interval cells, mesh refinement rows, coverage witnesses,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

No consumer can obtain open-cover compactness, a choice function for arbitrary covers, a completeness theorem, a quotient of names, or host-level real extensionality from the DyadicCover$^{\uparrow}$ *ledger alone.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket ledger nonescape`

Proof. Use the obligation surface of Definition 446.2. It lists the carrier, classifier, refinement stability, coverage exactness, and ledger rows available to every local certificate consumer.

For a Real$^{\uparrow}$ or FiniteWindowEnvelope$^{\uparrow}$ consumer, read the stream, regular-rational, real, and finite-window rows that are already displayed in the carrier. For a total-bounded or completion consumer, read the finite center and radius lists together with the mesh refinement and coverage routes.

All of these routes stay inside the finite BHist packet. The excluded conclusions require coordinates not present in Definition 446.1, so they cannot escape from the local ledger. □

Theorem 446.8 (DyadicCover window-cell incidence)**.** *Let* $D = (K, R, I, Q, W, H, C, P, N)$ *be an accepted* DyadicCover$^{\uparrow}$ *carrier, where* K *is the finite dyadic center list,* R *is the finite rational-radius list,* I *is the finite rational-interval cell list,* Q *records the named real-data rows, and* W *is the displayed finite-window coordinate. Every point covered inside* W *by the accepted packet factors through one listed dyadic center, one rational radius, one rational interval cell, one* StreamName$^{\uparrow}$ *row, one* RegSeqRat$^{\uparrow}$ *row, one* Real$^{\uparrow}$ *row, and the displayed* Cont *route from the corresponding stream and regular-rational name into that real and cell incidence record.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket window cell incidence`

Proof. Project the tuple $D = (K, R, I, Q, W, H, C, P, N)$. The coverage coordinate of the carrier is indexed by the displayed finite window W, so a covered point comes with a coverage witness rather than with an arbitrary open-family membership assertion.

Read that witness through the carrier definition. It names a dyadic center from K, a rational radius from R, and a rational interval cell from I, and it also names the stream, regular-rational, and real rows in Q used to place the point in the cell.

Finally follow the stored Cont route. The route is part of the accepted packet, so the incidence surface consumed downstream is exactly the finite window-cell record listed in the carrier. □

Theorem 446.9 (DyadicCover refinement witness exhaustion)**.** *Every later mesh-refinement witness for an accepted* DyadicCover$^{\uparrow}$ *packet is exhausted by finitely replacing old listed cells by named dyadic subcells whose center, radius, interval,* hsame, Cont, Pkg, *and* NameCert *rows are already recorded in the packet ledger. Such a witness cannot introduce an arbitrary open family, an external indexed cover, or a selector for subcovers.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket refinement witness exhaustion`

Proof. Start with Definition 446.3. It decomposes a mesh refinement into finite replacement of the old rational interval cells by named subcells, preserving the displayed finite-window coordinate and the local read routes.

Use Definition 446.7 on the refined packet. The ledger exposes only the finite center, radius, interval, mesh, coverage, hsame, Cont, Pkg, and NameCert rows recorded in the carrier.

Therefore the refinement witness is exhausted by those finite replacements. The excluded open-family and subcover-selector data have no coordinate in the accepted packet and cannot be projected from the ledger. □

Theorem 446.10 (DyadicCover total-bounded radius ledger)**.** *Every total-bounded consumer of an accepted* DyadicCover$^{\uparrow}$ *packet reads exactly the finite center list, rational-radius list, rational-interval list, mesh-refinement rows, and finite-window coverage witnesses displayed by the carrier, with the radius data routed through the corresponding* RationalBall$^{\uparrow}$ *rows before the consumer receives a bounded-radius cover packet.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket total bounded radius ledger`

Proof. Use Definition 446.4. It identifies the consumer-facing data as the finite center list, rational radii, interval cells, mesh-refinement rows, coverage witnesses, and read routes into the real and finite-window coordinates.

Then apply Definition 446.7 to the same packet. The radius coordinates available to the consumer are the displayed rational-radius rows and their RationalBall$^{\uparrow}$ routing; no external metric ball family or choice of cover can be read from the ledger.

Thus the total-bounded handoff is a bounded-radius ledger over the listed finite rows, not a compactness principle or a host-level open-cover argument. □

Theorem 446.11 (DyadicCover finite-window obligation triad)**.** *For every accepted* DyadicCover$^{\uparrow}$ *packet*

$$D = (K, R, I, Q, W, H, C, P, N),$$

the finite dyadic center-radius rows (K, R)*, the mesh-refinement row* Q*, and the* FiniteWindowEnvelope$^{\uparrow}$ *coverage row* W *jointly form the finite-window obligation surface. This surface includes the routed* RationalInterval$^{\uparrow}$ *and* RationalBall$^{\uparrow}$ *cell rows, the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *consumer reads, the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows, and no arbitrary open-cover family, subcover selector, metric-completeness claim, or quotient equality of names.*

Proof. Project the packet through Definition 446.1. The center list K, radius list R, interval-cell list I, mesh row Q, and finite-window row W are the only coordinates from which coverage data can be read.

Apply Definition 446.2 to bind the local certificate rows and Definition 446.8 to bind each finite-window point to one displayed center, radius, interval cell, stream-name row, regular-rational row, real row, and continuation route.

Finally use Definition 446.7. It exhausts consumer access by the displayed finite carrier rows, so the center-radius data, mesh row, and finite-window coverage row form exactly the obligation triad named in the statement. □

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite window obligation triad`

Theorem 446.12 (DyadicCover mesh-refinement carrier transport)**.** *Let $D = (K, R, I, Q, W, H, C, P, N)$ be an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *carrier, and let Q' be a displayed later* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement row of Q. Transporting D along Q' through the stored* hsame *and* Cont *rows preserves the finite center list, rational-radius list, interval-cell ledger, finite-window coverage witnesses, package provenance, and local* NameCert *row as one accepted carrier surface. The transported packet is a mesh refinement of the same finite-window cover, not a cover chosen from an arbitrary host family.*

Proof. Begin with Definition 446.3. It replaces each old cell by a finite named list of dyadic subcells routed by Q', while retaining the same finite-window coordinate W and local read routes.

Next apply Definition 446.9. The refinement witness is exhausted by the displayed center, radius, interval, hsame, Cont, Pkg, and NameCert rows, so transport has no route to an external indexed cover.

The incidence and radius rows remain attached to the transported cells by Definition 446.8 and Definition 446.10. Repacking these rows gives the transported carrier surface and preserves the stated non-escape boundary. □

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket mesh refinement carrier transport`

Theorem 446.13 (DyadicCover finite-window envelope factorization)**.** *Every* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *total-bounded, or completion-facing read of an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *packet factors through the finite-window route*

$$W \longmapsto Q \longmapsto I \longmapsto (K, R) \longmapsto H, C, P, N.$$

The route first fixes the displayed envelope window, then reads the dyadic mesh and refinement rows, then chooses one listed rational interval or rational ball cell with its dyadic center and radius, and finally transports the read through the recorded BEDC provenance. It does not assert open-cover compactness, ambient completeness, quotient equality of real names, or a host-level subcover choice.

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite window envelope factorization`

Proof. Start from a finite-window or real-facing consumer read. By Definition 446.8, any point inside W is read through one displayed stream-name row, regular-rational row, real row, and one listed interval or ball cell.

Apply Definition 446.12. Mesh transport keeps that cell inside the same carrier surface after finite refinement, and Definition 446.9 exhausts the witness by the listed finite rows.

Finally attach the total-bounded side through Definition 446.4 and Definition 446.10. These rows factor bounded-cover consumers through the same centers, radii, interval cells, mesh, window, transport, package, and naming rows. The finite-window envelope route therefore accounts for every listed consumer without any excluded host principle. □

Theorem 446.14 (DyadicCover finite-window mesh incidence package)**.** *Let*

$$D = (K, R, I, Q, W, H, C, P, N)$$

be an accepted DyadicCover$^{\uparrow}$ *packet. Every accepted coverage request for a point in the displayed finite window factors through exactly one listed dyadic center in K, one rational radius in R, one rational interval cell in I, one mesh row in Q, one* StreamName$^{\uparrow}$ *row, one* RegSeqRat$^{\uparrow}$ *row, one* Real$^{\uparrow}$ *row, and the displayed* Cont *route that relates those rows inside the local* BHist*,* Pkg*, and* NameCert *ledger.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite window mesh incidence package`

Proof. First unpack the packet with Definition 446.2. This fixes the carrier coordinates, the local classifier, and the named transport and read routes available to a coverage consumer.

Next use Definition 446.8. The covered point is read from the finite-window coordinate W, so its incidence witness is one listed center, radius, interval cell, stream-name row, regular-rational row, real row, and continuation route.

Finally apply Definition 446.7. Since the ledger has no coordinate for an external cover or selector, the displayed incidence witness is the whole consumer-facing package for the finite window and mesh surface. □

Theorem 446.15 (DyadicCover refined-window total-bounded package)**.** *Every mesh-refined total-bounded consumer of an accepted* DyadicCover$^{\uparrow}$ *packet receives only the refined finite dyadic center list, rational radius list, rational interval cell list, and the transported finite-window coverage witnesses already carried by the packet. The consumer receives these rows through the same* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, and it receives no open-cover compactness principle, arbitrary subcover choice, or external metric-ball family.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket refined window total bounded package`

Proof. Start with Definition 446.3. It replaces cells by a finite list of named dyadic subcells and transports the finite-window witnesses along the displayed mesh refinement row.

Then use Definition 446.9. This exhausts the refined witness by the packet's center, radius, cell, transport, continuation, package, and naming rows, so no new cover family appears during refinement.

Attach the total-bounded read by Definition 446.4 and Definition 446.10. The bounded-radius consumer therefore sees the transported finite window package and the refined finite lists, with the non-escape boundary supplied by Definition 446.7. □

Theorem 446.16 (DyadicCover Real envelope coverage exactness)**.** *Every* Real$^{\uparrow}$ *or* FiniteWindowEnvelope$^{\uparrow}$ *coverage consumer of an accepted* DyadicCover$^{\uparrow}$ *packet factors through the same finite envelope rows: the displayed finite window, mesh-refined rational interval cells, dyadic centers, rational radii, stream-name rows, regular-rational rows, real rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *certificate. Such a consumer cannot read open-cover compactness, arbitrary subcover choice, quotient real equality, or host real extensionality from the packet.*

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket real envelope coverage exactness`

Proof. Begin with the finite-window incidence package from Definition 446.14. It gives the unique listed center, radius, interval cell, mesh row, stream row, regular-rational row, real row, and continuation route for each accepted coverage request.

Now transport along refinements using Definition 446.15. The refined packet keeps the same finite envelope surface and presents only transported finite-window witnesses to downstream coverage consumers.

Finally combine Definition 446.4 with Definition 446.7. These theorems bind the Real-facing and finite-envelope reads to the displayed rows and exclude compactness, arbitrary subcovers, quotient equality of reals, and host real extensionality. □

Theorem 446.17 (DyadicCover finite-subcover exhaustion)**.** *Any downstream total-bounded or Real-envelope consumer of an accepted* DyadicCover$^{\uparrow}$ *packet can only read the displayed finite center list, dyadic radius rows, rational interval cells, mesh-refinement rows, finite-window envelope,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. In particular, the consumer receives a finite subcover already present in the packet and cannot replace it by an ambient open-cover family, an arbitrary subcover choice, quotient equality of real names, or host real extensionality.*

Proof. First apply finite refinement. The displayed center, radius, interval, and mesh rows are refined only by the finite dyadic subcells recorded in the packet, so every downstream cover read starts from the same finite list.

Next use refinement witness exhaustion. It identifies each refined witness with the packet's own center, radius, cell, transport, continuation, package, and naming rows, leaving no independent cover-family coordinate.

Now attach the refined-window total-bounded package. The total-bounded consumer therefore receives the transported finite window and the same bounded-radius ledger rather than a separate compactness principle.

Finally use Real envelope coverage exactness. It repacks the finite-window and total-bounded rows for Real$^{\uparrow}$ and FiniteWindowEnvelope$^{\uparrow}$ consumers, so the displayed finite subcover exhausts the exported coverage surface. □

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite subcover exhaustion`

Theorem 446.18 (DyadicCover scoped package)**.** *The local obligation surface of* DyadicCover$^{\uparrow}$ *assembles into a scoped package over* DyadicRatCore$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* RationalInterval$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* FiniteWindowEnvelope$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The scope is exactly the carrier, naming, finite-window triad, mesh transport, finite-window envelope factorization, refined total-bounded handoff, and Real envelope exactness supplied by Definition 446.2, Definition 446.11, Definition 446.12, Definition 446.13, Definition 446.15, and Definition 446.16.*

Proof. First project an accepted DyadicCover$^{\uparrow}$ packet to its displayed carrier coordinates. The NameCert obligation theorem fixes the finite center, radius, interval, mesh, finite-window, history, continuation, package, and name rows that may be read from the packet.

Next apply the finite-window obligation triad and mesh-refinement carrier transport theorems. They bind the finite window to the dyadic mesh and carry the same rows across displayed refinements, so the scoped package has one transport surface rather than separate host-side choices.

Now use finite-window envelope factorization and the refined-window total-bounded package. These results route every finite-window, total-bounded, and completion-facing read through the packet's own center, radius, interval, mesh, envelope, hsame, Cont, Pkg, and NameCert rows.

Finally the Real envelope coverage exactness theorem repacks the same finite packet for Real$^{\uparrow}$ and FiniteWindowEnvelope$^{\uparrow}$ consumers. The assembled package is therefore scoped to the listed

BEDC objects, with no open-cover compactness, arbitrary subcover choice, quotient equality of names, or host real extensionality added. □

Theorem 446.19 (DyadicCover public total-bounded export)**.** *Every public total-bounded or Real-envelope consumer of an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *packet receives exactly the finite center list, rational radius list, interval cells, mesh refinement rows, finite-window envelope coverage,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row assembled in Definition 446.18. The export factors through Definition 446.13, Definition 446.15, and Definition 446.16; it exports no open-cover compactness, arbitrary subcover choice, metric completeness, quotient equality of names, or host real extensionality.*

Proof. Start with Definition 446.18. It assembles the carrier, naming, mesh, finite-window envelope, refined total-bounded, and Real envelope rows into one scoped package.

Apply Definition 446.13. This identifies the coverage surface as the displayed finite envelope over the packet's own center list, radius list, interval cells, mesh rows, transports, routes, package provenance, and local name row.

Next use Definition 446.15. A total-bounded consumer therefore reads only the refined finite window and the same bounded-radius ledger, not an arbitrary subcover selection.

Finally apply Definition 446.16. It repacks those same finite coverage rows for Real-envelope consumers, so the public export carries no open-cover compactness, metric completeness, quotient equality of names, or host real extensionality. □

Theorem 446.20 (DyadicCover finite-cover consumer determinacy)**.** *For an accepted* $\mathsf{DyadicCover}^{\uparrow}$ *packet*

$$D = (K, R, I, Q, W, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *or completion-facing consumer that reads the cover factors through the same finite center list* K*, rational radius list* R*, interval-cell list* I*, mesh refinement rows* Q*, finite-window witnesses* W*, componentwise transports* H*, continuation routes* C*, package provenance* P*, and local name* N*. If two consumer routes expose the same displayed packet rows, then they expose the same consumer-visible cover package.*

The packet supplies no open-cover compactness, arbitrary subcover choice, metric completeness, quotient equality of real names, or host real extensionality as an alternate cover source.

Lean4 ✓ `BEDC.Derived.DyadicCoverUp.DyadicCoverPacket finite cover consumer determinacy`

Proof. Project the accepted packet and apply the NameCert obligations. The finite center list, rational radius list, interval-cell list, mesh rows, finite-window witnesses, transports, routes, provenance, and local name are the complete cover surface.

Use finite refinement, total-bounded handoff, and window-cell incidence. These rows bind the cover to the displayed mesh and cells, so a consumer cannot replace the packet by an arbitrary open cover or an external subcover choice.

Next apply the refined-window total-bounded package, Real envelope coverage exactness, and finite-subcover exhaustion. They route total-bounded, Real-envelope, and completion-facing reads through the same finite window and bounded-radius ledger.

Finally combine the scoped package with the public total-bounded export. The consumer-visible cover package is therefore determined by $K, R, I, Q, W, H, C, P, N$, and two routes exposing those rows expose the same public cover package. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicCover}^{\uparrow}$ packet is generated from finite dyadic centers, dyadic radii, rational interval cells, mesh refinement rows, coverage witnesses for a displayed finite window, hsame transports, Cont read routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature consumer package is the finite carrier, naming, mesh transport, finite-window envelope, refined total-bounded, Real envelope package, Definition 446.5, the checked total-bounded handoff export of Definition 446.19, and the finite-cover consumer determinacy theorem of Definition 446.20 over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, arbitrary subcover choice, metric completeness, compact-metric equivalence, quotient equality of real names, host real extensionality, or bridge checking.

Upgrade path: No further paper closure grade remains; Lean verification and bridge checking remain pending.

Chapter 447

A Concrete Naming Certificate for $\mathsf{CollisionKernelSpectrum}^{\uparrow}$

The $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ packet is a BEDC-native finite certificate surface. It records a finite golden-mean fold moment ledger with a collision-kernel matrix and a spectral-shadow row, linking the finite-window surfaces of $\mathsf{GoldenMeanShift}^{\uparrow}$, $\mathsf{FibonacciCube}^{\uparrow}$, and $\mathsf{FoldMomentKernel}^{\uparrow}$ without importing external linear algebra or an ambient spectral theorem.

The packet consumes the finite sampling grid of chapter 451 and the finite transfer-row handoff of chapter 456: finite sampling supplies the visible phase window and aliasing rows, while the transfer-operator packet supplies the route by which those finite rows become the collision-kernel spectral shadow.

The fold-source row F is read as the finite path surface supplied by chapter 401, so the collision-kernel matrix extends that sibling carrier rather than introducing an isolated spectral packet.

Definition 447.1 (Collision kernel spectrum carrier)**.** *A* $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (G, F, V, I, K, S, H, T, P, N)$$

with the following visible coordinates:

1. *G is the finite golden-mean window row, displaying binary words accepted by the* $\mathsf{GoldenMeanShift}^{\uparrow}$ *no-adjacent-one ledger;*
2. *F is the fold-source row, recording the corresponding* $\mathsf{FibonacciCube}^{\uparrow}$ *finite path row and the fold map into visible fibers;*
3. *V is the visible fiber-multiplicity row, listing the finite fibers and their displayed cardinal ledgers;*
4. *I is the finite moment-index row used by* $\mathsf{FoldMomentKernel}^{\uparrow}$ *to read the selected fold moments;*
5. *K is the collision-kernel matrix row whose entries count displayed pair collisions between fiber rows inside the finite window;*
6. *S is the spectral-shadow row, a finite ledger of row sums, diagonal weights, and moment shadows read from K;*
7. *H is the adjacency-moment handoff row from the golden-mean and Fibonacci-cube ledgers to the fold-moment rows;*
8. *T stores the componentwise* hsame *and* Cont *routes for G, F, V, I, K, S, H;*

9. *P is the* Pkg *provenance naming* GoldenMeanShift$^{\uparrow}$, FibonacciCube$^{\uparrow}$, FoldMomentKernel$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *source rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CollisionKernelSpectrum}^{\uparrow}}$ *provenance row.*

The classifier compares two carriers only by the displayed finite rows $G, F, V, I, K, S, H, T, P, N$. No coordinate contains an infinite spectrum, an external matrix space, a Perron eigenvector, or an imported external object.

Theorem 447.2 (Collision kernel spectrum NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CollisionKernelSpectrum}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the finite packet of Definition 447.1;*

2. *classifier laws by componentwise* hsame *on golden-mean windows, fold-source rows, fiber multiplicities, moment indices, kernel entries, spectral shadows, handoff rows, transports, package provenance, and the local certificate row;*

3. *finite-window kernel exactness: every entry of K is read from displayed collisions between rows of V over the finite G, F, I ledger;*

4. *spectral-shadow exactness: every row of S is a finite row-sum, diagonal, or selected moment shadow of K;*

5. *non-escape: accepted carriers export no infinite spectrum, no host linear algebra, no choice of eigenbasis, and no external import.*

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier namecert obligations`

Proof. Carrier admission is the displayed tuple $C = (G, F, V, I, K, S, H, T, P, N)$. Each coordinate is a finite BHist row or a finite route between such rows.

For the classifier laws, compare the coordinates componentwise after the stored hsame and Cont routes. The local certificate row has no access to any datum beyond $G, F, V, I, K, S, H, T, P, N$, so reflexive, symmetric, and transitive reads are inherited coordinate by coordinate.

Kernel exactness is finite ledger inspection. A matrix entry of K is accepted only when the relevant pair of visible fiber rows in V is displayed over the same golden-mean window, fold-source row, and moment-index row.

Spectral-shadow exactness then projects S. Its row sums, diagonal weights, and selected moment shadows are computed from the displayed entries of K and the same handoff row H.

Non-escape follows from the carrier definition. Since no coordinate names an infinite spectrum, host matrix space, chosen eigenbasis, or external artifact, a downstream NameCert consumer can cite only the finite rows and their recorded routes. □

Theorem 447.3 (Collision kernel spectrum finite-window spectral shadow)**.** *For every accepted* CollisionKernelSpectrum$^{\uparrow}$ *carrier, the spectral-shadow row S is determined by the finite golden-mean window G, the fold-source row F, the visible fiber-multiplicity row V, the moment-index row I, and the collision-kernel matrix row K. Two accepted carriers with matching G, F, V, I, K rows therefore have matching spectral-shadow rows after the displayed* hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier finite window spectral shadow`

Proof. Project the carrier and read S through the spectral-shadow exactness obligation. Each row of S is a row-sum, diagonal weight, or selected moment shadow of the displayed kernel entries in K.

The kernel entries are themselves read from finite collisions between fiber rows in V over the same golden-mean window G, fold-source row F, and moment-index row I. Thus matching G, F, V, I, K rows transport the same finite calculations.

The stored routes in T and the provenance in P, N carry these componentwise readings between accepted packets. No external spectral theorem is needed because the shadow is only the finite ledger projected from K. □

Theorem 447.4 (Collision kernel spectrum ledger non-escape)**.** *Every exported route from an accepted* CollisionKernelSpectrum$^{\uparrow}$ *packet factors through the finite golden-mean window row, fold-source row, fiber-multiplicity row, moment-index row, collision-kernel matrix row, spectral-shadow row, adjacency-moment handoff row, componentwise transport row, package provenance, and local* NameCert *row. The packet exports no infinite shift space, no ambient graph theory, no host spectral measure, no selected eigenvector, and no external theorem.*

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier ledger non escape`

Proof. Take an exported consumer route and project the carrier of Definition 447.1. The only coordinates available are $G, F, V, I, K, S, H, T, P, N$.

The NameCert obligations account for carrier formation, classifier transport, finite-window kernel exactness, spectral-shadow exactness, and non-escape. Each of these obligations refers to a displayed finite row of the same packet.

Therefore every export factors through the listed finite coordinates. Since the carrier definition has no further coordinate, it cannot supply an infinite shift object, ambient graph theorem, host spectral measure, selected eigenvector, or imported external result. □

Theorem 447.5 (Collision kernel spectrum adjacency-moment handoff)**.** *The handoff row H sends each accepted golden-mean adjacency ledger in G through the finite* FibonacciCube$^{\uparrow}$ *fold-source row F to the moment-index rows used by* FoldMomentKernel$^{\uparrow}$*. Consequently every collision-kernel entry in K is indexed by a displayed finite adjacency path, a visible fiber row, and a finite moment index from the same carrier.*

Proof. Project H from the carrier. Its source is the golden-mean no-adjacent-one ledger G, and its intermediate row is the finite Fibonacci-cube fold source F.

The visible fiber row V records which fold fibers are present over that source, while I records the finite moments read by FoldMomentKernel$^{\uparrow}$. The handoff row ties these three displays to the same finite window.

Thus an entry of K cannot be indexed independently of the adjacency and moment ledgers. It is accepted only as a collision count between displayed fibers over G, F, V, I, transported by T and recorded by P, N. □

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier adjacency moment handoff`

Theorem 447.6 (Collision kernel spectrum window restriction)**.** *Let $C = (G, F, V, I, K, S, H, T, P, N)$ be an accepted* CollisionKernelSpectrum$^{\uparrow}$ *carrier, and let G' be a displayed golden-mean finite window obtained by restricting G to a smaller no-adjacent-one row already visible through the G, F, V, I*

ledger. The restricted packet formed by projecting F, V, I, K, S to that smaller window is again accepted, and its kernel entries and spectral-shadow rows are exactly the corresponding displayed rows of the original packet.

Proof. Project the accepted carrier through Definition 447.1. The finite-window data are displayed in G, while F, V, I record the fold-source, visible fibers, and moment indices read over that window.

Restricting to G' deletes no external datum; it only selects already displayed rows from G, F, V, I. The collision entries in K over the smaller window are therefore the same finite pair-collision rows, read after the same componentwise transport through T.

The spectral-shadow row S is a ledger of row sums, diagonal weights, and moment shadows of K. Projecting S along the restricted kernel rows preserves exactly those displayed finite calculations, with provenance still recorded by P, N. □

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier window restriction`

Theorem 447.7 (Collision kernel spectrum fiber-row symmetry)**.** *In an accepted* $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ *carrier, the collision count attached to two displayed visible fiber rows is unchanged when the two fiber rows are exchanged, provided the same finite golden-mean window, fold-source row, and moment-index row are retained. The transported entry remains a row of the same collision-kernel matrix K and feeds the same spectral-shadow index in S.*

Proof. Read the relevant kernel entry from K. By the NameCert obligations, such an entry is accepted only as a finite collision count between two displayed fiber rows in V over the same G, F, I ledger.

Swapping the two visible fiber rows changes only the order in which the finite pair is inspected. The componentwise hsame and Cont routes in T carry the exchanged pair back to the same finite carrier surface.

The moment index is not changed by this exchange. Hence the resulting count is the same displayed kernel row, and the spectral-shadow row S reads it at the same finite moment index. □

Theorem 447.8 (Collision kernel spectrum moment-index projection)**.** *Every exported spectral-shadow row of an accepted* $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ *carrier factors through a finite moment-index row I. In particular, the export can mention only the displayed rows $G, F, V, I, K, S, H, T, P, N$ and cannot cite an infinite spectrum, a host matrix space, an eigenbasis, or ambient linear algebra.*

Proof. Start from an exported spectral-shadow route and project the carrier. The only spectral-shadow coordinate is S, and each row of S is displayed as a row sum, diagonal weight, or selected moment shadow of entries in K.

The entries of K are indexed over the same finite ledger G, F, V, I. The moment information available to the export is therefore precisely the finite row I, transported through T and recorded with the handoff and provenance rows H, P, N.

No carrier coordinate names an infinite spectrum, a host matrix space, an eigenbasis, or ambient linear algebra. Thus every exported spectral-shadow row factors through I and the listed finite packet rows. □

Lean4 ✓ `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier moment index projection`

Theorem 447.9 (Collision kernel spectrum kernel-entry source lock)**.** *Every collision-kernel entry displayed in the row K of an accepted* $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ *carrier is sourced only from the*

golden-mean window G, the FibonacciCube$^{\uparrow}$ *fold-source row F, the visible fiber row V, the moment-index row I, and the componentwise* hsame *and* Cont *routes recorded in T. No entry of K can be read from an infinite shift space, an ambient graph adjacency object, a host matrix entry, an outside theorem, or an external formalization object outside the displayed packet.*

Proof. Project the carrier $C = (G, F, V, I, K, S, H, T, P, N)$ and focus on one displayed entry of K. By Definition 447.5, the entry is indexed by a finite adjacency path, a visible fiber row, and a finite moment index from the same carrier.

The golden-mean window and the fold-source row supply the allowed source coordinates. The visible fiber row supplies the two finite rows being compared, and the moment-index row supplies the selected read position.

The only transport allowed between these coordinates is the componentwise hsame and Cont data stored in T. Since the carrier has no further coordinate for an infinite shift space, host matrix entry, ambient adjacency theorem, outside theorem object, or external formalization object, every accepted kernel entry is locked to the displayed finite source rows. □

Theorem 447.10 (Collision kernel spectrum shadow classifier stability)**.** *If two accepted* CollisionKernelSpectrum$^{\uparrow}$ *carriers are classified equal by the displayed componentwise classifier on $G, F, V, I, K, H, T, P, N$, then their spectral-shadow rows S match after the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *routes. The classifier therefore transports spectral-shadow rows without invoking a host spectral measure, selected eigenvector, or ambient spectral theorem.*

Proof. Begin with the componentwise classifier. It identifies the visible finite rows $G, F, V, I, K, H, T, P, N$ through the stored hsame and Cont routes and records the same package provenance and local naming rows.

Apply Definition 447.3. The row S is determined by the finite rows G, F, V, I, K, so matching those rows forces the same row sums, diagonal weights, and selected moment shadows.

The remaining coordinates H, T, P, N carry the handoff, transport, package, and local certificate data needed to read this equality as a BHist route. Thus spectral-shadow stability is a finite classifier fact and has no dependency on host spectral structure. □

Theorem 447.11 (Collision kernel spectrum downstream readback package)**.** *Every downstream consumer of an accepted* CollisionKernelSpectrum$^{\uparrow}$ *spectrum packet factors through the finite tuple*

$$(G, F, V, I, K, S, H, T, P, N).$$

Such a consumer can read the displayed golden-mean window, fold-source row, fiber row, moment-index row, kernel row, spectral-shadow row, adjacency-moment handoff, componentwise transport, package provenance, and local naming row, but cannot read an infinite shift space, host spectral measure, selected eigenvector, host linear-algebra object, outside theorem object, or external formalization object.

Proof. First apply Definition 447.4. Every exported route from the packet already factors through the listed finite coordinates.

Next apply Definition 447.9. Any consumer that reads K can only recover entries sourced from G, F, V, I and transported through T.

Finally apply Definition 447.5. The adjacency, fold, and moment reads are tied to the same carrier, while S, H, P, N record the finite shadow, handoff, package, and naming rows. Since none of these coordinates contains an infinite shift space, host spectral measure, selected eigenvector, host linear-algebra object, outside or imported theorem object, or external formalization object, the downstream readback package is exactly the displayed tuple. □

Closure status for $\mathsf{NameCert}_{\mathsf{CollisionKernelSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ packet is generated from finite golden-mean windows, Fibonacci-cube fold-source rows, visible fiber multiplicities, finite moment indices, collision-kernel matrix rows, spectral shadows, handoff rows, componentwise transports, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CollisionKernelSpectrum}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The scoped surface records the finite carrier, local NameCert obligations, finite-window spectral shadow, ledger non-escape, adjacency-moment handoff, smaller-window restriction, fiber-row symmetry, moment-index projection, source lock, classifier stability, and downstream readback rows for collision-kernel spectrum packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CollisionKernelSpectrum}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CollisionKernelSpectrumUp.CollisionKernelSpectrumCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close infinite spectra, host linear algebra, Perron eigenvectors, spectral measures, external imports, or a standard bridge.

Upgrade path: Public closure requires a single exported $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ public certificate over the scoped spectrum packet.

Chapter 448

A Concrete Naming Certificate for $\mathsf{CauchyOscillation}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the Cauchy oscillation criterion is read here as a finite tail-window object: a real-facing consumer may inspect named rational observations, an explicit modulus schedule, dyadic tolerances, and a displayed ledger of tail comparisons, but it never receives a quotient stream, a chosen limit, host real equality, or an ambient completeness principle. The $\mathsf{CauchyOscillation}^{\uparrow}$ packet is the finite oscillation surface consumed by $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, and later $\mathsf{Real}^{\uparrow}$ completion certificates.

The $\mathsf{CauchyOscillation}^{\uparrow}$ packet is the single-tail finite-window sibling of $\mathsf{UniformCauchyCriterion}^{\uparrow}$. Its window, modulus, dyadic tolerance, and oscillation ledger rows are consumed by the finite-family route of chapter 435 before any $\mathsf{Real}^{\uparrow}$ seal can be read.

The oscillation packet consumes the $\mathsf{RegSeqRat}^{\uparrow}$ core of chapter 301, the threshold schedule of chapter 292, and the dyadic-to-Real handoff of chapter 291 and chapter 38.

Its oscillation ledger is therefore consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301 and the $\mathsf{DyadicTailBound}^{\uparrow}$ dependency before any $\mathsf{Real}^{\uparrow}$ seal consumer inspects the finite tail comparison.

Definition 448.1 (CauchyOscillation carrier)**.** *A* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, M, Q, T, S, H, C, P, N)$$

with the following visible rows:

1. *W is a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row, displaying the named rational observations available to the consumer;*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule selecting the tail region inspected for each precision request;*
3. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose dyadic budget is used by the oscillation test;*
4. *T is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *oscillation ledger comparing all named tail observations beyond the threshold supplied by M under the tolerance supplied by Q;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after the finite oscillation ledger has been accepted;*
6. *H records the componentwise* hsame *transports for the tail window, modulus schedule, dyadic tolerance, oscillation ledger, seal handoff, package row, and local certificate row;*
7. *C records the displayed* Cont *read routes by which consumers inspect W, M, Q, T, S, H, P, N;*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows, their componentwise hsame *transports, their* Cont *read routes, their package provenance, and their local naming rows.*

Lean4 (definition) `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier`

Theorem 448.2 (CauchyOscillation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier habitation: the packet of Definition 448.1 displays a finite tail-window row, a modulus threshold schedule, a dyadic tolerance row, an oscillation ledger, a* $\mathsf{Real}^{\uparrow}$ *seal handoff, transport routes, package provenance, and local naming provenance;*

2. *common tail-window classifier: accepted packets are compared only by componentwise* hsame *and displayed* Cont *routes over $W, M, Q, T, S, H, C, P, N$;*

3. *oscillation-bound exactness: every accepted comparison in T uses the threshold row M, the tolerance row Q, and the finite tail observations in W from the same packet;*

4. *stability under* hsame *and* Cont*: transported carriers preserve the selected tail window, threshold, tolerance, oscillation ledger, and seal handoff as one finite dependency packet;*

5. *ledger non-escape: accepted carriers export no quotient stream, no selected limit, no host real equality, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier namecert obligations`
↪ variant `BEDC.Derived.CauchyOscillationUp.CauchyOscillationPacket semantic name certificate`

Proof. Carrier habitation is the projection of the finite tuple

$$O = (W, M, Q, T, S, H, C, P, N).$$

Each coordinate is a displayed BHist row or a displayed route between such rows, so the packet supplies the advertised tail window, threshold schedule, tolerance row, oscillation ledger, seal handoff, provenance, and local certificate data.

For the common classifier, compare two accepted packets componentwise through the stored hsame rows and replay only the Cont routes recorded in C. Because the classifier has no further coordinate, it can read only $W, M, Q, T, S, H, C, P, N$ and cannot substitute an external tail object.

Oscillation-bound exactness is read from the paired rows M, Q, T together with W. The threshold selected by M determines which tail observations in W are compared, the dyadic budget in Q determines the comparison radius, and the ledger T records exactly those finite comparisons.

Stability follows by transporting the same rows through H and then replaying the same read routes in C. The package row P records the $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source certificates, while N records the local certificate row.

Finally, the carrier has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness. Repacking the projections therefore gives the five NameCert obligations and the stated non-escape boundary. □

Theorem 448.3 (CauchyOscillation tail-window exposure)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier. Every oscillation comparison exposed by* T *must first read the finite tail-window row* W *before it reaches the modulus threshold row* M*, the dyadic tolerance row* Q*, or the downstream* $\mathsf{Real}^{\uparrow}$ *seal row* S*.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier tail window exposure`

Proof. Open the carrier of Definition 448.1. The comparison ledger T is defined as the displayed tail-bound row comparing named observations, and those named observations live in the finite window W.

Apply the NameCert obligations of Definition 448.2. Oscillation-bound exactness ties each accepted comparison in T to the same W, M, Q coordinates of the carrier, while stability under hsame and Cont preserves that dependency under transport and replay.

Finally use the seal-handoff factorization of Definition 448.9. The terminal seal S is available only after the finite prefix W, M, Q, T has been accepted, so no comparison can bypass W by reading the threshold, tolerance, or seal rows alone. □

Theorem 448.4 (CauchyOscillation dyadic tolerance stability)**.** *For an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

componentwise hsame *transport and displayed* Cont *replay preserve the dyadic tolerance row* Q *together with the tail window* W*, the modulus threshold* M*, and the oscillation ledger* T*. A transported consumer therefore reads the same finite dependency packet* W, M, Q, T *before any seal handoff.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationDyadicToleranceStability`

Proof. By Definition 448.1, the transport row H and replay row C are componentwise rows over $W, M, Q, T, S, H, C, P, N$. They do not contain a separate tolerance object or a separate tail ledger.

The NameCert obligations of Definition 448.2 make the dependence exact: stability under hsame and Cont preserves the selected tail window, threshold, tolerance, oscillation ledger, and seal handoff as one finite packet.

The seal-handoff factorization theorem Definition 448.9 then keeps S downstream of W, M, Q, T. Hence any transported read that preserves Q also preserves the associated W, M, and T rows used by that tolerance. □

Theorem 448.5 (CauchyOscillation modulus-threshold exactness)**.** *Every modulus-root consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier factors through*

$$W, M, Q, T$$

before it reads the $\mathsf{Real}^{\uparrow}$ *seal row* S*. In particular, the threshold row* M *is never read as a free modulus witness: it is paired with the displayed tail window* W*, dyadic tolerance* Q*, and oscillation ledger* T *from the same packet.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier modulus threshold exactness`

Proof. Start with the modulus-root package theorem Definition 448.20. It packages the modulus-facing rows as the finite tail-window surface W, M, Q, T and places the structural rows H, C, P, N around the same packet.

Apply the tail-window exposure theorem Definition 448.3. It forces each comparison governed by the threshold row M to expose the finite window W before threshold, tolerance, ledger, or seal data can be consumed.

Now use dyadic tolerance stability from Definition 448.4. Transport and replay preserve Q only together with W, M, T, so a modulus-root consumer cannot detach the dyadic tolerance from the threshold and ledger rows.

Finally invoke the seal-handoff factorization theorem Definition 448.9. Since S is downstream of the accepted prefix W, M, Q, T, every modulus-root consumer reads those rows before any Real seal is available. □

Theorem 448.6 (CauchyOscillation RegSeqRat handoff)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the row W is exposed as a $\mathsf{StreamName}^{\uparrow}$ *finite tail-window or as the finite tail-window read consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *before any oscillation comparison in T is read. No consumer may inspect an oscillation comparison until the route through W, the threshold row M, and the dyadic tolerance row Q has displayed the regular-rational tail surface used by the comparison.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier regseqrat handoff`

Proof. Project the accepted carrier through Definition 448.1. The only stream-facing input is W, and the only comparison rows downstream of it are M, Q, T before the seal handoff S.

Apply Definition 448.3. It makes every tail comparison expose W before threshold, tolerance, ledger, or seal data are read, so W cannot be bypassed by a later oscillation consumer.

Use Definition 448.5. The threshold row M is tied to the same W, Q, T packet, which is precisely the finite tail-window route consumed by the regular-rational readback surface.

Finally, Definition 448.2 keeps the carrier, replay, provenance, and naming rows attached to that same route. Hence every oscillation comparison reads the StreamName or RegSeqRat tail-window surface before it reaches T. □

Theorem 448.7 (CauchyOscillation dyadic tail exactness)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier $O = (W, M, Q, T, S, H, C, P, N)$, each row of the oscillation ledger T is exactly the* $\mathsf{DyadicTailBound}^{\uparrow}$ *comparison over the same tail-window row W, modulus-threshold row M, and dyadic tolerance row Q. Componentwise* hsame *transport H and displayed* Cont *replay C preserve this dependence and do not create a second tolerance or tail-window source.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier dyadic tail exactness`

Proof. Begin with the carrier projection of Definition 448.1. The row T is listed only after W, M, Q, so its comparison data are not independent of the displayed tail-window, threshold, and tolerance rows.

Invoke dyadic tolerance stability, Definition 448.4. It states that transport and replay preserve Q together with W, M, T, preventing a transported ledger row from detaching the tolerance from the tail window.

Apply Definition 448.5. Since every modulus-root consumer reads W, M, Q, T before the seal, the threshold and dyadic tolerance used in T are the same rows projected from the accepted carrier.

The NameCert obligations of Definition 448.2 supply the exhaustive ledger clause. Thus every entry of T is the displayed $\mathsf{DyadicTailBound}^{\uparrow}$ comparison over W, M, Q, stable under H and C. □

Theorem 448.8 (CauchyOscillation Real seal nonescape)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier* $O = (W, M, Q, T, S, H, C, P, N)$*, each* $\mathsf{Real}^{\uparrow}$ *seal read* S *factors through* W, M, Q, T *and exports no quotient stream equality, selected limit, countable choice, or ambient completeness principle. The public seal is therefore a routed oscillation-bound handoff, not a detached real-completion object.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier real seal nonescape`

Proof. Project the carrier of Definition 448.1. The row S appears only as the terminal seal after the tail-window row W, threshold row M, dyadic tolerance row Q, and oscillation ledger T.

Apply Definition 448.9. It states that every displayed route to S factors through the finite prefix W, M, Q, T of the same carrier.

Now combine Definition 448.6 and Definition 448.7. The first theorem exposes the StreamName or RegSeqRat tail-window surface before any comparison is consumed, and the second identifies the ledger as the exact DyadicTailBound comparison over that same window, threshold, and tolerance.

Finally use Definition 448.2. Its nonescape clause leaves no row for quotient stream equality, selected limits, countable choice, or ambient completeness. Thus the Real seal read is only the routed handoff through W, M, Q, T. □

Theorem 448.9 (CauchyOscillation seal-handoff factorization)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier, and suppose that* S *is the exported* $\mathsf{Real}^{\uparrow}$ *seal handoff of* O*. Every displayed consumer read in* C *whose terminal row is* S *factors through the same finite oscillation surface*

$$(W, M, Q, T)$$

of that packet. More explicitly, such a read has a finite prefix reading the tail-window row W*, the threshold schedule* M*, and the dyadic tolerance row* Q*; it then reads the oscillation ledger* T *whose comparisons are made exactly from those three rows; only after that ledger prefix is accepted may the terminal projection expose* S*. No consumer read of* S *may replace this prefix by a quotient stream, a selected limit, host real equality, an ambient completeness principle, or rows taken from a different* $\mathsf{CauchyOscillation}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier seal handoff factorization`

Proof. By Definition 448.1, the carrier has exactly the displayed rows

$$W, M, Q, T, S, H, C, P, N.$$

The row S is not an independent real object: it is the $\mathsf{Real}^{\uparrow}$ seal handoff reached only after the finite oscillation ledger has been accepted. Thus any consumer read whose terminal row is S must be one of the displayed Cont routes recorded in C, and its final step is a projection from the accepted packet rather than an external construction of a real.

It remains to identify the prefix that makes the seal available. The ledger row T is specified as the $\mathsf{DyadicTailBound}^{\uparrow}$ oscillation ledger comparing the named tail observations in W beyond the threshold supplied by M under the dyadic tolerance supplied by Q. Hence a read of T cannot be

separated from the same W, M, and Q coordinates of O: the observations, cutoff schedule, and tolerance budget are precisely the data from which the ledger comparisons are recorded.

The NameCert obligations in Definition 448.2 make this dependence stable and exhaustive. Oscillation-bound exactness says that every accepted comparison in T uses the threshold row M, the tolerance row Q, and the finite tail observations in W from the same packet. Stability under hsame and Cont says that transported reads preserve the selected tail window, threshold, tolerance, ledger, and seal handoff as one finite dependency packet. Ledger non-escape says that the accepted carrier exports no quotient stream, selected limit, host real equality, or ambient completeness principle.

Therefore a displayed route to S has no available input other than the accepted packet rows and cannot bypass the accepted ledger. Its terminal seal read factors through the finite prefix W, M, Q, T of the same carrier, with T itself determined by W, M, and Q. This is the asserted handoff factorization. □

Theorem 448.10 (CauchyOscillation root window lock)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier consumed by a completion-facing* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or tail-bound reader. Every such consumer must read the finite root package* W, M, Q, T *before it can read the seal row* S*. The read cannot be replaced by a quotient stream, a selected limit, host real equality, or an ambient completeness claim.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier root window lock`

Proof. Start with the carrier definition, Definition 448.1. The rows W, M, Q, T are the only finite source of the tail window, threshold schedule, dyadic tolerance, and oscillation ledger.

Apply tail-window exposure and modulus-threshold exactness, Definition 448.3 and Definition 448.5. They force each completion-facing read to expose W and then to read M and Q as the rows that determine the ledger T.

Finally use the seal-handoff factorization, Definition 448.9. Since S is available only after the accepted prefix W, M, Q, T, the completion-facing reader has no route around the root package and no coordinate for the refused external objects. □

Theorem 448.11 (CauchyOscillation uniform criterion consumer)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier consumed by a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *reader. The reader must expose the finite tail-window row* W*, modulus-threshold row* M*, dyadic tolerance row* Q*, oscillation ledger* T*, transport row* H*, replay row* C*, provenance row* P*, and local name row* N *before it can use the packet as a uniform Cauchy criterion input. It cannot bypass those rows by reading a detached modulus, an unstated dyadic tolerance, a quotient stream, or a selected limit.*

Proof. Start with tail-window exposure, Definition 448.3. It makes W visible before any oscillation comparison can be consumed.

Apply modulus-threshold exactness, Definition 448.5. The threshold row M is tied to the same W, Q, T packet rather than to a free modulus witness.

Use dyadic-tail exactness, Definition 448.7. It identifies the ledger T as the displayed dyadic comparison over W, M, Q.

The structural rows H, C, P, N then transport, replay, source, and name that finite route. Therefore a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer receives exactly the displayed oscillation packet and has no route around the finite tail window, threshold, tolerance, and ledger. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier uniform criterion consumer`

Theorem 448.12 (CauchyOscillation real completion consumer)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier consumed by a* $\mathsf{Real}^{\uparrow}$ *completion-facing reader. The reader may reach the seal row S only after the finite prefix W, M, Q, T has been displayed and replayed through H, C, P, N. The packet exports no quotient stream equality, selected real limit, host real equality, countable choice, or ambient completeness principle.*

Proof. Begin with the root window lock above. It forces every completion-facing read to expose W, M, Q, T before S is available.

Apply Real seal nonescape, Definition 448.8. The seal row is a routed oscillation-bound handoff and not a detached real-completion object.

Use dyadic-tail exactness, Definition 448.7, to keep the terminal read tied to the same dyadic tolerance and tail ledger. The comparison feeding the seal is therefore the displayed T over W, M, Q.

Finally, the structural rows H, C, P, N preserve this route without adding quotient, selected-limit, host-equality, countable-choice, or ambient-completeness data. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier real completion consumer`

Theorem 448.13 (CauchyOscillation root-unblock package)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the root-unblock surface consumed by $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is exactly the finite package $W, M, Q, T, S, H, C, P, N$. Uniform-criterion consumers must read W, M, Q, T, H, C, P, N before using the oscillation packet, and real-completion consumers must read the same finite prefix before the seal row S. No consumer may replace this package by a detached modulus, unstated dyadic tolerance, quotient stream, selected limit, host real equality, countable choice, or ambient completeness theorem.*

Proof. Use the uniform criterion consumer theorem above. It exposes the finite tail window, threshold, tolerance, ledger, and structural rows before any $\mathsf{UniformCauchyCriterion}^{\uparrow}$ read is public.

Use the real completion consumer theorem above. It places the $\mathsf{Real}^{\uparrow}$ seal downstream from the same finite prefix and refuses every detached completion route.

The tail-window exposure, modulus-threshold exactness, dyadic-tail exactness, and Real-seal nonescape theorems supply the four locking points of the package: W must be visible, M must remain attached to W, Q, T, T must be the displayed dyadic comparison over W, M, Q, and S must remain downstream from the accepted ledger.

Thus the only public root-unblock surface is $W, M, Q, T, S, H, C, P, N$ itself. The listed forbidden objects are absent from the carrier and cannot be reconstructed by transport, replay, provenance, or naming. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRootUnblockPackage`

Theorem 448.14 (CauchyOscillation completion root route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N)$$

factors through the finite route W, M, Q, T*, then the* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback carried by* W*, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*, and only then the* $\mathsf{Real}^{\uparrow}$ *seal row* S*. The route exports no quotient stream equality, selected limit, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with tail-window exposure and the root window lock, Definition 448.3 and Definition 448.10. They force every completion-facing read to expose W, M, Q, T before the seal row S can be consumed.

Use the RegSeqRat handoff and dyadic tolerance stability, Definition 448.6 and Definition 448.4. These identify W as the displayed StreamName or RegSeqRat tail-window readback and keep the dyadic tolerance row Q attached to the same window, threshold, and ledger under H and C.

Apply dyadic tail exactness, Definition 448.7. The oscillation ledger T is exactly the displayed tail comparison over W, M, Q, so a downstream read has no detached tail-bound or quotient-stream coordinate.

Finally use Real seal nonescape and seal-handoff factorization, Definition 448.8 and Definition 448.9. They place S downstream from the same finite prefix and exclude selected limits, host equality, countable choice, and ambient Real completeness from the accepted route. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRootUnblockRoute`

Theorem 448.15 (CauchyOscillation L10 window admission)**.** *Let*

$$O = (W, M, Q, T, S, H, C, P, N)$$

be an accepted $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier. Every L10 completion-facing read of* O *must first expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q *and the* $\mathsf{StreamName}^{\uparrow}$ *finite window row carried by* W *before any oscillation ledger entry in* T *is consumed. The read may then proceed through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the* $\mathsf{Real}^{\uparrow}$ *seal row* S*, but it cannot start from a detached completion object or an ambient Real equality.*

Proof. Begin with the carrier of Definition 448.1. The rows W and Q are the visible finite window and dyadic tolerance coordinates of the packet.

Apply tail-window exposure, Definition 448.3. It forces every oscillation comparison to read W before threshold, tolerance, ledger, or seal data are consumed.

Use dyadic tolerance stability, Definition 448.4. The tolerance row Q is transported with the same window and threshold rows, so an L10 consumer cannot detach it from W.

Finally combine the RegSeqRat handoff and completion root route, Definition 448.6 and Definition 448.14. They place the regular-rational handoff and the Real seal after the same displayed prefix, excluding detached completion and ambient equality reads. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationL10WindowAdmission`

Theorem 448.16 (CauchyOscillation RegSeqRat Real seal route)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier* $O = (W, M, Q, T, S, H, C, P, N)$*, any consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *seal through the oscillation packet must pass through the finite route*

$$W, Q, M, T, S.$$

The row W exposes the StreamName$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *finite window, Q supplies the* DyadicRatCore$^{\uparrow}$ *tolerance, M fixes the threshold schedule, T records the tail comparison, and S is the terminal* Real$^{\uparrow}$ *seal. No route may replace this sequence by a quotient stream, selected limit, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRegSeqRatRealSealRoute`

Proof. Project the packet rows from Definition 448.1. The only Real-facing terminal row is S, and it is listed after the window, threshold, tolerance, and ledger rows.

Apply the RegSeqRat handoff theorem, Definition 448.6. It identifies the finite tail window W as the StreamName or regular-rational surface consumed before an oscillation comparison is read.

Use dyadic tolerance stability and dyadic tail exactness, Definition 448.4 and Definition 448.7. These keep Q attached to the same W, M prefix and identify T as the displayed comparison over that prefix.

The Real seal nonescape theorem, Definition 448.8, then makes S a routed seal rather than an independent real object. The refused quotient, selected-limit, choice, and completeness objects are not rows of the carrier. □

Theorem 448.17 (CauchyOscillation completion consumer boundary)**.** *Every completion consumer of an accepted* CauchyOscillation$^{\uparrow}$ *packet receives exactly the displayed tail-window, dyadic-tolerance, RegSeqRat-handoff, Real-seal, transport, replay, provenance, and local naming surface of*

$$W, M, Q, T, S, H, C, P, N.$$

It may consume the packet as completion input only after those rows have been exposed, and it cannot ask the packet for a hidden completed metric space, a selected limit, host equality of real names, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCompletionConsumerBoundary`

Proof. Start from the completion root route, Definition 448.14. It already factors each completion-facing read through the finite route W, M, Q, T and then to the seal row S.

The L10 window admission theorem above makes the entry point explicit: the dyadic tolerance row and finite window row are read before a ledger entry is consumed.

The RegSeqRat Real seal route above identifies the regular-rational handoff and terminal Real seal as downstream from the same window, threshold, tolerance, and comparison rows.

It remains to account for structural stability. The NameCert obligations, Definition 448.2, keep H, C, P, N attached as transport, replay, provenance, and local naming rows for the same packet. Since the carrier contains no hidden completed space, selected limit, host equality, or ambient completeness row, no completion consumer can read one from this boundary. □

Theorem 448.18 (CauchyOscillation tail tolerance package)**.** *For every accepted* CauchyOscillation$^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the finite tail-tolerance package consists of the StreamName$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *tail-window row W, the cutoff schedule M, the* DyadicRatCore$^{\uparrow}$ *tolerance row Q, and the* DyadicTailBound$^{\uparrow}$ *ledger T. Componentwise* hsame *transport H and displayed* Cont *replay C preserve this package as one finite boundary before the* Real$^{\uparrow}$ *seal is read.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationTailTolerancePackage`

Proof. Project the carrier of Definition 448.1. The rows W, M, Q, T list exactly the StreamName or RegSeqRat window, cutoff schedule, dyadic tolerance, and oscillation ledger used by the carrier.

Use the RegSeqRat handoff and dyadic tolerance stability, Definition 448.6 and Definition 448.4. These keep the tail-window and dyadic tolerance rows attached to the same ledger under H and C.

Apply dyadic tail exactness, Definition 448.7. It identifies every row of T with the displayed DyadicTailBound comparison over W, M, Q.

The seal-handoff factorization then places S strictly downstream of the accepted package. Thus completion consumers receive a finite tail-tolerance surface before any Real seal projection. □

Theorem 448.19 (CauchyOscillation completion consumer refusal)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N)$$

is confined to the finite route $W, M, Q, T, S, H, C, P, N$*. It cannot replace that route by quotient streams, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and it cannot read a seal that is detached from the displayed tail-window and dyadic-tolerance package.*

Proof. Begin with the NameCert obligations, Definition 448.2. Their carrier admission, exactness, transport, continuation, provenance, and naming clauses exhaust the accepted rows of O.

Combine the root window lock and tail tolerance package, Definition 448.10 and Definition 448.18. They make every completion-facing read pass through W, M, Q, T and preserve that package under H and C.

Now apply Real seal nonescape and seal-handoff factorization, Definition 448.8 and Definition 448.9. These route the terminal $\mathsf{Real}^{\uparrow}$ seal through the same finite prefix and exclude detached real-completion data.

Since the carrier has no row for quotient streams, selected limits, countable choice, host real equality, or ambient completeness, a completion consumer cannot obtain those objects from the accepted package. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCompletionConsumerRefusal`

Theorem 448.20 (CauchyOscillation modulus-root package)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

each modulus-root read packages $W, M, Q, T, S, H, C, P, N$ *as one finite tail-window surface. The row* W *supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *observations,* M *supplies the threshold schedule,* Q *supplies the dyadic tolerance,* T *supplies the oscillation comparison ledger, and* S *is the downstream* $\mathsf{Real}^{\uparrow}$ *seal available only after those rows are accepted. The structural rows* H, C, P, N *transport, replay, source, and name the same package, with no quotient stream, selected limit, host real equality, or ambient completeness principle.*

Proof. Open the carrier of Definition 448.1. The modulus-facing rows are exactly W, M, Q, T: a finite tail window, a threshold schedule, a dyadic tolerance, and an oscillation ledger comparing the displayed tail observations.

Apply the NameCert obligations of Definition 448.2. Oscillation-bound exactness keeps each comparison in T tied to the same W, M, Q coordinates, while stability under hsame and Cont preserves the finite dependency packet.

Now use the seal-handoff factorization theorem Definition 448.9. It places the terminal row S after the accepted prefix W, M, Q, T, so a modulus-root consumer cannot read the Real seal before the finite oscillation surface is visible.

Finally H, C, P, N only transport, replay, source, and name that same route. The carrier contains no coordinate for the refused ambient objects, so the modulus-root read is precisely the displayed finite package. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationModulusRootPackage`

Theorem 448.21 (CauchyOscillation dyadic-tail Real-seal route)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

every Real-facing oscillation consumer factors through the ordered route

$$W,\ M,\ Q,\ T$$

before it reads the $\mathsf{Real}^{\uparrow}$ *seal row* S*, where* W *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *tail-window row,* M *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule,* Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* T *is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *oscillation ledger. The route exports no quotient stream, no selected limit, no host real equality, and no ambient completeness principle.*

Proof. Start with tail-window exposure, Definition 448.3. It forces every oscillation comparison exposed by T to read the finite window W before threshold, tolerance, ledger, or seal data can be consumed.

Use dyadic tolerance stability, Definition 448.4. It preserves Q only together with W, M, T under componentwise hsame transport and displayed Cont replay.

Apply modulus-threshold exactness, Definition 448.5. It prevents M from being read as a free modulus witness by tying the threshold to the same tail window, dyadic tolerance, and oscillation ledger.

Finally apply seal-handoff factorization, Definition 448.9. Since the terminal seal S is available only after the accepted prefix W, M, Q, T, every Real-facing oscillation consumer follows the stated dyadic-tail route. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationDyadicTailRealSealRoute`

Theorem 448.22 (CauchyOscillation oscillation ledger transport)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the oscillation ledger T *transports as a ledger over the same* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *tail-window row* W*,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* M*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*. Under componentwise* hsame *transport and displayed* Cont *replay, no transported ledger may replace* T *by a tail comparison sourced from a different packet, an unstated* $\mathsf{DyadicTailBound}^{\uparrow}$ *row, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Open the carrier of Definition 448.1. The ledger row T is the displayed $\mathsf{DyadicTailBound}^{\uparrow}$ oscillation ledger, and its comparisons are specified over the same W, M, Q rows of the packet.

Apply the NameCert obligations of Definition 448.2. Oscillation-bound exactness ties every accepted comparison in T to W, M, Q, while stability under hsame and Cont preserves the selected tail window, threshold, tolerance, ledger, and seal handoff as one finite dependency packet.

Use dyadic tolerance stability, Definition 448.4, and modulus-threshold exactness, Definition 448.5. Together they prevent the transported read from detaching Q or M from the same tail-window and ledger rows.

Finally apply the dyadic-tail Real-seal route, Definition 448.21. Since any Real-facing consumer reaches S only after W, M, Q, T, there is no coordinate from which to import a ledger from another packet, an unstated dyadic tail-bound row, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier ledger transport`

Theorem 448.23 (CauchyOscillation tail-modulus comparison obligation)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the comparison between the tail-window row and the modulus row is an obligation over the same displayed packet: W *is the* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface,* M *is the tail threshold schedule,* Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* T *is the comparison ledger, and* S *is the downstream* $\mathsf{Real}^{\uparrow}$ *seal. A* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer may read this comparison only after the route through* W, M, Q, T, H, C, P, N *has been displayed.*

Proof. Use tail-window exposure, Definition 448.3. It makes the finite window W visible before any threshold, tolerance, ledger, or seal row is consumed.

Apply modulus-threshold exactness and dyadic tolerance stability, Definition 448.5 and Definition 448.4. These keep M and Q attached to the same tail-window and ledger rows under H and C.

Finally use ledger transport, Definition 448.22. The transported comparison is still over the displayed $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ row, the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, and the same downstream $\mathsf{Real}^{\uparrow}$ seal route. Thus a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ read receives the finite comparison obligation rather than a detached modulus witness. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier tail modulus comparison obligation`

Theorem 448.24 (CauchyOscillation tail-window carrier obligation)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the finite tail-window packet W, M, Q, T *carries the visible* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget used by each oscillation comparison. A downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer can read a tail-window comparison only through the displayed rows* W *for observations,* M *for the tail threshold,* Q *for the dyadic bound, and* T *for the finite comparison ledger, together with* H, C, P, N*.*

Proof. Inspect the carrier rows of Definition 448.1. The only row containing a dyadic tolerance is Q, and the only row containing tail comparisons is T.

The tail-window row W supplies the finite StreamName or RegSeqRat observations, while M selects the tail region in which those observations are compared. Thus a comparison row in T cannot be read before the packet has displayed the observation window, threshold schedule, and dyadic budget.

Componentwise transport through H and replay through C preserve these rows together. The package and local certificate rows P, N only source and name the same finite route, so every consumer receives the displayed tail-window carrier rather than an external oscillation object. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier tail window carrier obligation`

Theorem 448.25 (CauchyOscillation dyadic bound classifier obligation)**.** *Let* $O = (W, M, Q, T, S, H, C, P, N)$ *and* $O' = (W', M', Q', T', S', H', C', P', N')$ *be accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carriers. If the classifier identifies the displayed rows componentwise, then the oscillation budget transported from* O *to* O' *is the same visible* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *rows. In particular, classifier equality preserves the tail-window observations, threshold schedule, dyadic budget, oscillation ledger, and seal handoff as one finite dependency packet.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier dyadic bound classifier obligation`

Proof. Open the classifier described in Definition 448.1. It compares accepted carriers only by componentwise hsame rows, displayed Cont routes, package provenance, and local naming rows over the listed coordinates.

The dyadic budget is therefore transported only as the Q coordinate attached to the same W, M, T rows. The classifier has no second tolerance coordinate, no alternate RegSeqRat readback row, and no detached StreamName window from which a different oscillation budget could be imported.

Apply the transport clause in the NameCert obligations. The componentwise hsame rows preserve W, M, Q, T, S, and replay through C and C' reads the same displayed route. Hence classifier equality preserves the visible dyadic-bound route through the regular-rational or stream-window surface. □

Theorem 448.26 (CauchyOscillation uniform Cauchy consumer obligation)**.** *Every* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N)$$

receives only the displayed finite budget route

$$W,\ M,\ Q,\ T,\ H,\ C,\ P,\ N$$

before the $\mathsf{Real}^{\uparrow}$ *seal row* S *is exposed. The consumer obtains no quotient stream, selected limit, host real equality, ambient completeness principle, or hidden uniform modulus beyond the finite tail-window and dyadic-budget rows.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier uniform cauchy consumer obligation`

Proof. Start with the finite package in Definition 448.24. It identifies W, M, Q, T as the visible tail-window, threshold, dyadic-bound, and comparison-ledger route.

Use the classifier obligation of Definition 448.25. Transport and replay preserve the same budget route under componentwise hsame, so a uniform Cauchy consumer cannot replace the displayed dyadic budget by a hidden modulus.

Finally combine the existing uniform criterion handoff and completion-frontier obligation, Definition 448.27 and Definition 448.28. They place the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ read inside W, M, Q, T, H, C, P, N before the terminal seal S, while the carrier nonescape clause excludes quotient streams, selected limits, host equality, ambient completeness, and hidden uniform data. □

Theorem 448.27 (CauchyOscillation uniform criterion handoff)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the handoff to $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *is the finite row sequence* W, M, Q, T, H, C, P, N *followed by the* $\mathsf{Real}^{\uparrow}$ *seal row* S*. The handoff displays* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *as named source or consumer rows, and it exports no quotient stream, selected limit, host real equality, countable choice, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationUniformCriterionHandoff`

Proof. Start with the root window lock and completion root route, Definition 448.10 and Definition 448.14. They force the completion-facing route through W, M, Q, T before S can be read.

Use the RegSeqRat handoff and dyadic tail exactness, Definition 448.6 and Definition 448.7. These identify W as the $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ readback surface and Q, T as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and comparison ledger.

The seal-handoff factorization, Definition 448.9, then places the $\mathsf{Real}^{\uparrow}$ seal after the accepted finite prefix. The same prefix is exactly the packet that a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer may inspect, so the handoff remains finite and excludes the refused ambient objects. □

Theorem 448.28 (CauchyOscillation completion-frontier obligation)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the completion frontier consists only of the finite $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *tail window* W*, the modulus row* M*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*, the oscillation ledger* T*, the* $\mathsf{Real}^{\uparrow}$ *seal row* S*, and the transport, replay, provenance, and local naming rows* H, C, P, N*. A downstream* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives this frontier as an obligation packet and receives no ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCompletionFrontierObligation`

Proof. Project the carrier through Definition 448.1. The only completion-facing rows are the displayed finite rows $W, M, Q, T, S, H, C, P, N$.

Apply the tail tolerance package and completion consumer refusal, Definition 448.18 and Definition 448.19. They keep the tail-window, threshold, tolerance, ledger, and seal rows together and rule out quotient streams, selected limits, countable choice, host real equality, and ambient completeness.

Use the uniform criterion handoff of Definition 448.27. Since the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer reads the same finite prefix before the $\mathsf{Real}^{\uparrow}$ seal, the completion frontier is an obligation surface, not a completed-real construction. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier completion frontier obligation`

Theorem 448.29 (CauchyOscillation tail-window cofinality)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the tail-window row W *is cofinal for every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of the oscillation packet. Any such read must pass through* W, M, Q, T *before the seal row* S *is available, and it cannot replace that route by quotient streams, selected limits, host real equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier tail window cofinality`

Proof. Project the carrier of Definition 448.1. The only rows before the terminal seal are the displayed tail window W, modulus threshold M, dyadic tolerance Q, oscillation ledger T, and structural rows H, C, P, N.

Apply tail-window exposure and the RegSeqRat handoff, Definition 448.3 and Definition 448.6. These force every oscillation comparison to open the finite $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ window before threshold or ledger data are consumed.

Use dyadic tail exactness and the dyadic-tail Real-seal route, Definition 448.7 and Definition 448.21. They identify the ledger as the exact finite comparison over W, M, Q and place the $\mathsf{Real}^{\uparrow}$ seal downstream from that comparison.

Finally the completion-frontier obligation of Definition 448.28 keeps every completion-facing consumer inside the same finite packet, so no refused ambient object supplies an alternate cofinal route. □

Theorem 448.30 (CauchyOscillation modulus-tail monotonicity)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the modulus threshold row M is monotone only along the displayed tail route W, Q, T. If a downstream read tightens the dyadic tolerance or moves farther into the tail, the accepted comparison remains a read of the same W, M, Q, T packet under H and C; no detached modulus schedule or external completeness principle can supply the monotonicity witness.

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationModulusTailMonotonicity`

Proof. Start with modulus-threshold exactness, Definition 448.5. It ties M to the same tail-window, tolerance, and oscillation-ledger rows rather than to a free threshold parameter.

Use dyadic tolerance stability, Definition 448.4. Tightening or transporting the tolerance row preserves Q together with W, M, T under componentwise hsame transport and displayed Cont replay.

Apply ledger transport and tail-window cofinality, Definition 448.22 and Definition 448.29. The comparison ledger remains over the same finite tail route, and every downstream Real or completion read still reaches the seal only through that route.

Thus the monotonicity available to a consumer is exactly the finite modulus-tail monotonicity displayed by W, M, Q, T, H, C, P, N. □

Theorem 448.31 (CauchyOscillation Real-seal uniqueness boundary)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

any two downstream $\mathsf{Real}^{\uparrow}$ *or completion consumers that read the seal row S through the packet meet at the same finite oscillation boundary $W, M, Q, T, S, H, C, P, N$. The packet supplies no second Real seal, no host equality proof between seals, no selected limit, and no ambient completion theorem.*

Proof. Project the carrier. The row S is the only Real-facing seal coordinate, and it appears after the finite tail window, modulus threshold, dyadic tolerance, and oscillation ledger.

Use Real seal nonescape and seal-handoff factorization, Definition 448.8 and Definition 448.9. They prevent a consumer from reading S except through the accepted prefix W, M, Q, T.

Apply tail-window cofinality and modulus-tail monotonicity, Definition 448.29 and Definition 448.30. These keep both consumers on the same cofinal tail route and the same threshold-tolerance ledger under the displayed transport and replay rows.

Since H, C, P, N only transport, replay, source, and name that packet, the uniqueness boundary is the shared finite route itself, not host real equality or an ambient completion object. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier real seal uniqueness boundary`

Definition 448.32 (CauchyOscillation uniform-limit source subpacket)**.** *For an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

the uniform-limit source subpacket is

$$O_{\mathrm{ulim}} = (W, M, Q, T, H, C, P, N).$$

It consists of the $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *finite tail-window row* W*, the threshold row* M*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* Q*, the oscillation ledger* T*, and the transport, replay, provenance, and local naming rows. The* $\mathsf{Real}^{\uparrow}$ *seal row* S *remains downstream of this subpacket and is not a selected limit.*

Lean4 (definition) `BEDC.Derived.CauchyOscillationUp.CauchyOscillationUniformLimitSource`

Theorem 448.33 (CauchyOscillation uniform-limit tail handoff)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *packet must first display the uniform-limit source subpacket of Definition 448.32. The handoff is the finite route* W, M, Q, T, H, C, P, N*; it cannot be replaced by quotient streams, a selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with tail-window exposure and dyadic tolerance stability, Definition 448.3 and Definition 448.4. They force every oscillation comparison to open the finite tail row W and keep the tolerance row Q attached to the same W, M, T packet.

Use the RegSeqRat handoff and the uniform criterion handoff, Definition 448.6 and Definition 448.27. These identify the displayed source as the $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ tail route consumed by uniform Cauchy readers.

Finally apply the completion-frontier obligation, Definition 448.28. It keeps the $\mathsf{Real}^{\uparrow}$ seal downstream from O_{ulim} and refuses quotient-stream, selected-limit, host-equality, countable-choice, and ambient-completeness endpoints. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationUniformLimitTailHandoff`

Theorem 448.34 (CauchyOscillation completion-source exhaustion)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier, the completion source available to* $\mathsf{UniformLimit}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumers is exhausted by the route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal displayed in* $W, M, Q, T, S, H, C, P, N$*. No further completion source, quotient stream, selected limit, host real equality, or ambient completeness theorem is exported by the packet.*

Proof. Project the source subpacket O_{ulim} from Definition 448.32. Its rows name the stream or regular-rational window, the threshold schedule, the dyadic tolerance, and the oscillation ledger before any seal read.

Apply the dyadic-tail Real-seal route and completion-frontier obligation, Definition 448.21 and Definition 448.28. These place $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ on the same finite handoff and block ambient completion data.

Use Real-seal uniqueness boundary, Definition 448.31. Since the packet supplies only the displayed seal route and no second seal or host equality proof, the listed source chain exhausts the completion-facing surface. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCompletionSourceExhaustion`

Theorem 448.35 (CauchyOscillation RegSeqRat uniform-threshold preservation)**.** *Under componentwise* hsame *transport and displayed* Cont *replay, the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *tail-window row W preserves the uniform threshold row M together with the dyadic tolerance row Q and oscillation ledger T. A transported* $\mathsf{UniformLimit}^{\uparrow}$ *or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer therefore reads the same finite threshold packet W, M, Q, T, H, C, P, N before any* $\mathsf{Real}^{\uparrow}$ *seal handoff.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRegSeqRatUniformThreshold`

Proof. Begin with modulus-threshold exactness and modulus-tail monotonicity, Definition 448.5 and Definition 448.30. They tie M to the displayed tail route W, Q, T and prevent a detached threshold schedule from supplying the comparison.

Use ledger transport and the RegSeqRat handoff, Definition 448.22 and Definition 448.6. The transported ledger remains over the same $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{StreamName}^{\uparrow}$ window and the same dyadic tolerance row.

Finally apply the uniform-limit tail handoff, Definition 448.33. It presents W, M, Q, T, H, C, P, N as the finite source packet required before a uniform-limit or uniform-Cauchy consumer may reach the Real seal. □

Theorem 448.36 (CauchyOscillation completion-frontier public readiness)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier*

$$O = (W, M, Q, T, S, H, C, P, N),$$

every completion-frontier consumer must first read the finite source subpacket W, M, Q, T, H, C, P, N and then the $\mathsf{Real}^{\uparrow}$ *seal row S through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route. The public surface exports no quotient stream, selected limit, host equality, countable choice, or ambient completeness theorem.*

Proof. Use the uniform-limit tail handoff, Definition 448.33. It forces a uniform-limit or uniform-Cauchy reader to open the source subpacket W, M, Q, T, H, C, P, N before any seal read.

Apply completion-source exhaustion, Definition 448.34. The only completion-facing chain available to the packet runs through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the downstream $\mathsf{Real}^{\uparrow}$ seal.

Finally use RegSeqRat uniform-threshold preservation, Definition 448.35. It keeps transported consumers on the same finite threshold and tolerance route, so the completion frontier remains public readiness data rather than a quotient stream, selected limit, host equality, choice principle, or ambient completeness claim. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCompletionFrontierPublicReadiness`

Theorem 448.37 (CauchyOscillation audit-clean obligation package)**.** *Every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier supplies an audit-clean L10 root-unblock package*

$$W, M, Q, T, S, H, C, P, N$$

whose rows expose the tail window, modulus threshold, dyadic tolerance, oscillation ledger, Real seal, componentwise transport, displayed replay, provenance, and local NameCert obligations. The package is consumed by $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readers only through those displayed rows and exports no quotient stream, selected limit, host equality, countable choice, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier audit clean obligation package`

Proof. Start with the NameCert obligations of Definition 448.2. They project the accepted carrier into the visible rows $W, M, Q, T, S, H, C, P, N$ and identify carrier admission, exactness, transport, replay, provenance, and local naming as the whole certificate surface.

Use tail-window exposure and dyadic tolerance stability, Definition 448.3 and Definition 448.4. These force the comparison ledger to read the same finite tail window and dyadic tolerance under transport and replay.

Apply the RegSeqRat handoff and Real seal nonescape, Definition 448.6 and Definition 448.8. The regular-rational route and the terminal seal remain downstream of the displayed finite prefix.

Finally use completion-frontier public readiness, Definition 448.36. It identifies the same rows as the public L10 completion frontier and excludes the refused host objects, so the displayed packet is the requested audit-clean obligation package. □

Theorem 448.38 (CauchyOscillation tail-window audit package)**.** *The tail-window audit package of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier is exactly the finite subpacket*

$$W, M, Q, T, H, C, P, N.$$

It records the $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-window row, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *oscillation comparison ledger, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local NameCert rows before any* $\mathsf{Real}^{\uparrow}$ *seal handoff is read.*

Proof. Project the uniform-limit source subpacket of Definition 448.32. By definition it contains W, M, Q, T, H, C, P, N and leaves the seal row S downstream.

Apply tail-window exposure, RegSeqRat handoff, and dyadic-tail exactness, Definition 448.3, Definition 448.6, and Definition 448.7. These identify W as the finite window route, M as the threshold row, Q as the tolerance row, and T as the exact dyadic tail comparison ledger over those rows.

Use ledger transport and RegSeqRat uniform-threshold preservation, Definition 448.22 and Definition 448.35. The structural rows H, C, P, N preserve and name that same finite packet for uniform-limit and uniform-Cauchy consumers.

The seal-handoff factorization theorem Definition 448.9 keeps S after this subpacket. Therefore the audit package is exactly W, M, Q, T, H, C, P, N before any Real seal is public. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationTailWindowAuditPackage`

Theorem 448.39 (CauchyOscillation Real completion audit handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier reaches the seal row* S *only after consuming the audit package* W, M, Q, T, H, C, P, N. *The handoff factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{DyadicTailBound}^{\uparrow}$ *oscillation ledger, transport, replay, provenance, and local NameCert rows, and it never exports quotient stream equality, selected limits, host real equality, countable choice, or ambient Real completeness.*

Proof. Begin with the tail-window audit package of Definition 448.38. It fixes the finite source packet W, M, Q, T, H, C, P, N before any Real-facing read is allowed.

Use uniform-limit tail handoff and completion-source exhaustion, Definition 448.33 and Definition 448.34. These place $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers on the same finite source route.

Apply Real seal nonescape and seal-handoff factorization, Definition 448.8 and Definition 448.9. They make S a routed seal after W, M, Q, T rather than an independent Real-completion object.

Finally invoke completion-frontier public readiness, Definition 448.36. The public completion surface is exactly the displayed audit route plus the downstream seal, so no excluded host or quotient object is exported. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRealCompletionAuditHandoff`

Theorem 448.40 (CauchyOscillation common tail-window classifier)**.** *Let $O = (W, M, Q, T, S, H, C, P, N)$ and $O' = (W', M', Q', T', S', H', C', P', N')$ be accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carriers. A classifier comparison between them is common-tail only when it identifies the finite tail window W with W', the modulus schedule M with M', the dyadic tolerance row Q with Q', the oscillation ledger T with T', and the* $\mathsf{Real}^{\uparrow}$ *seal handoff S with S' through the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No comparison may replace the common window by a quotient stream, selected limit, or ambient completeness witness.*

Proof. Project the two carriers. Their classifier-visible rows are exactly the tail window, modulus, tolerance, ledger, seal, transport, replay, provenance, and local naming coordinates.

The common-tail condition is therefore componentwise. The hsame rows compare W, M, Q, T, S with their primed partners, while the Cont rows replay only those displayed comparisons.

Finally P, N and P', N' source and name the same finite shape. Because neither carrier contains a quotient stream, selected limit, or ambient completeness row, the classifier cannot escape the common displayed tail window. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCommonTailWindowClassifier`

Theorem 448.41 (CauchyOscillation ledger nonescape)**.** *For every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier $O = (W, M, Q, T, S, H, C, P, N)$, an oscillation ledger read from T can cite only the same finite tail window W, modulus schedule M, dyadic tolerance row Q,* $\mathsf{Real}^{\uparrow}$ *seal handoff S, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ *row N. It cannot export a detached modulus, a second dyadic tolerance, quotient stream equality, selected limit, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read T from the carrier. By definition it is the displayed oscillation ledger over the tail observations in W beyond the threshold supplied by M under the dyadic budget Q.

Use tail-window audit package and Real completion audit handoff. These theorems keep T inside the same W, M, Q, H, C, P, N route before the seal row S is available.

The remaining exclusions follow from the absence of carrier coordinates. Since O has no detached modulus, second tolerance, quotient-stream, selected-limit, host-equality, or completeness row, no ledger read can export one. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationCarrier ledger nonescape`

Theorem 448.42 (CauchyOscillation mature status route)**.** *The mature paper surface of* $\mathsf{CauchyOscillation}^{\uparrow}$ *is the finite completion-audit route assembled from the audit-clean obligation package, the Real completion audit handoff, the common tail-window classifier, and ledger nonescape. Every accepted carrier*

$$O = (W, M, Q, T, S, H, C, P, N)$$

is read through the displayed tail window, modulus threshold, dyadic tolerance, oscillation ledger, Real seal, componentwise transport, replay, provenance, and local NameCert rows before a completion-facing consumer may inspect the seal.

Proof. Start with the audit-clean obligation package of Definition 448.37. It exposes $W, M, Q, T, S, H, C, P, N$ as the whole carrier package for L10 completion readers.

Next use the Real completion audit handoff, Definition 448.39. The seal row S is available only after the finite audit package W, M, Q, T, H, C, P, N has been consumed.

Apply the common tail-window classifier and ledger nonescape, Definition 448.40 and Definition 448.41. They keep classifier comparison and ledger reads on the same displayed tail window and exclude detached modulus, second tolerance, quotient stream, selected limit, host equality, and ambient completeness rows.

Finally read the scoped status route of Definition 448.45. The same rows supply the mature paper surface because the audit package, seal handoff, classifier, and ledger refusal are all named inside the local $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ route. □

Theorem 448.43 (CauchyOscillation mature status formal target)**.** *The formalization target for the mature* $\mathsf{CauchyOscillation}^{\uparrow}$ *status route is the single finite assembly that consumes the carrier obligations of* CauchyOscillationCarrier_namecert_obligations*, the semantic certificate surface of* CauchyOscillationPacket_semantic_name_certificate*, the seal handoff of Definition 448.39, ledger transport through Definition 448.41, and the completion-consumer refusal route. For an accepted carrier* $O = (W, M, Q, T, S, H, C, P, N)$*, the target asserts that every mature completion-facing read is exhausted by the displayed tail window, modulus threshold, dyadic tolerance, oscillation ledger, Real seal, componentwise transport, replay, provenance, and local NameCert rows.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationMatureStatusRouteFormalTarget`

Proof. Start with the carrier and semantic certificate obligations. They provide the finite rows $W, M, Q, T, S, H, C, P, N$ and identify the local $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ surface as the only named packet available to a mature reader.

Next apply the seal handoff. The Real completion audit handoff places the seal row S downstream of the audit package, so no completion-facing consumer may inspect a detached Real endpoint before the tail-window, modulus, tolerance, ledger, transport, replay, provenance, and name rows have been exposed.

Use ledger transport and completion-consumer refusal. Ledger nonescape keeps every oscillation read inside the same displayed tail window, while the completion-consumer refusal excludes quotient streams, selected limits, host equality, countable choice, and ambient completeness from the exported surface.

The theorem is therefore the paper-side target for the Lean declaration named CauchyOscillationMatureStatusRou it repacks the checked component rows into one mature route while leaving the existing verified component markers as the active Lean evidence. □

Theorem 448.44 (CauchyOscillation bridge route)**.** *Every bridge-facing read of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *carrier factors through the audit package of Definition 448.38, the Real completion handoff of Definition 448.39, and the common tail-window classifier of Definition 448.40 before any completion consumer inspects the packet.*

Proof. The cited audit package fixes the finite tail-window, modulus, tolerance, ledger, transport, replay, provenance, and naming rows. The Real completion handoff places the seal after that package, and the common tail-window classifier keeps bridge comparisons inside the same displayed window. Therefore a completion consumer reaches only the routed bridge-facing packet. □

Theorem 448.45 (CauchyOscillation NameCert obligation scope)**.** *The* $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$ *obligation scope for an accepted carrier* $O = (W, M, Q, T, S, H, C, P, N)$ *is exactly the finite package consisting of the tail window* W*, modulus schedule* M*, dyadic tolerance* Q*, oscillation ledger* T*,* $\mathsf{Real}^{\uparrow}$ *seal* S*,* BHist *carrier rows,* hsame *transports,* Cont *replay,* Pkg *provenance, and local naming. The scope includes common tail-window classification and ledger non-escape, and it excludes quotient streams, selected limits, host real equality, countable choice, and ambient completeness.*

Proof. Start with the NameCert obligations of Definition 448.2. They list carrier habitation, common classifier, oscillation-bound exactness, transport-replay stability, and ledger non-escape over $W, M, Q, T, S, H, C, P, N$.

Apply common tail-window classifier and ledger nonescape above. The first fixes classifier comparisons to the same finite tail-window package, and the second prevents ledger reads from exporting detached completion data.

Thus the obligation scope is the displayed finite package itself. The refused quotient, selected-limit, host-equality, choice, and completeness objects are not rows of the carrier and are not introduced by transport, replay, provenance, or naming. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationNameCertObligationScope`

Theorem 448.46 (CauchyOscillation shared dyadic window scope)**.** *Every accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *packet reads* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *through one shared finite tail window* W*. The threshold row* M*, dyadic tolerance row* Q*, oscillation ledger* T*, and structural rows* H, C, P, N *preserve that same window before the* $\mathsf{Real}^{\uparrow}$ *seal* S *is available.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationSharedDyadicWindowScope`

Proof. Project the carrier and apply tail-window exposure. The row W is the only StreamName or RegSeqRat window coordinate, while M and Q are the only threshold and dyadic tolerance rows attached to it.

Use modulus-threshold exactness and dyadic-tail exactness. These identify T as the displayed comparison ledger over the same W, M, Q packet, so no later read can swap in another tail window.

Finally apply ledger transport. The rows H, C, P, N transport, replay, source, and name the same finite packet, keeping every completion-facing consumer inside the shared dyadic window. □

Theorem 448.47 (CauchyOscillation RegSeqRat Real nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *packet receives only the routed* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* W, M, Q, T, H, C, P, N *before the seal row* S*. It receives no quotient stream, selected limit, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationRegSeqRatRealNonescape`

Proof. Begin with the RegSeqRat handoff and the shared dyadic window scope. Together they place the regular-rational readback inside the same finite tail-window, modulus, dyadic tolerance, and ledger packet.

Apply Real seal nonescape and seal-handoff factorization. The terminal seal S is reached only after the displayed prefix, rather than from an independent completion object.

The carrier has no row for quotient streams, selected limits, host equality, choice, or ambient completeness. Transport, replay, provenance, and local naming therefore preserve the routed RegSeqRat handoff without adding any refused Real-facing data. □

Theorem 448.48 (CauchyOscillation tail-modulus route)**.** *Every oscillation tail read using a dyadic tail bound must route through* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *before a uniform-limit consumer can inspect it. For an accepted carrier* $O = (W, M, Q, T, S, H, C, P, N)$*, the dyadic tail bound is read as the displayed tolerance and oscillation ledger over the shared finite window* W*, not as a host modulus, quotient stream, selected tail, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationTailModulusRoute`

Proof. Start with the NameCert obligation scope of Definition 448.45. It fixes $W, M, Q, T, S, H, C, P, N$ as the displayed carrier rows and keeps the oscillation ledger inside the local naming surface.

Next apply the shared dyadic window scope of Definition 448.46. The CauchyModulus and DyadicTailBound reads use the same finite StreamName and RegSeqRat window, so the CauchyCriterion-facing tail request cannot swap in an external modulus or a selected infinite tail.

Finally use Definition 448.47. The RegSeqRat handoff reaches the Real seal only after the displayed tail window, modulus, dyadic tolerance, oscillation ledger, transport, replay, provenance, and naming rows have been consumed.

Thus a uniform-limit consumer inspects only the routed finite tail-modulus evidence. The route carries no quotient stream, host real equality, countable-choice schedule, selected limit, or ambient completeness coordinate. □

Theorem 448.49 (CauchyOscillation public export readiness)**.** *Every public completion-frontier read of an accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *packet factors through the readiness route of Definition 448.36, the tail-window audit package of Definition 448.38, and the RegSeqRat-to-Real nonescape route of Definition 448.47. The exported surface is the finite packet* $W, M, Q, T, S, H, C, P, N$*, not a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient completeness theorem.*

Proof. First read completion-frontier public readiness. It makes the displayed source subpacket and the downstream $\mathsf{Real}^{\uparrow}$ seal the public-facing route.

Next read the tail-window audit package. That package identifies the finite prefix W, M, Q, T, H, C, P, N before any seal handoff is available.

Finally apply RegSeqRat Real nonescape. The regular-rational route reaches S only through the displayed prefix, so the public export contains no refused completion object. □

Lean4 ✓ `BEDC.Derived.CauchyOscillationUp.CauchyOscillationPublicExportReadiness`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyOscillation}^{\uparrow}$ packet is generated from a finite $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{StreamName}^{\uparrow}$ tail window, a $\mathsf{CauchyModulus}^{\uparrow}$ threshold schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a $\mathsf{DyadicTailBound}^{\uparrow}$ oscillation ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyOscillation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 448.39, Definition 448.40, Definition 448.41, and Definition 448.42 record the Real-completion audit handoff through the common tail-window classifier and ledger nonescape for constructive Cauchy oscillation packets.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyOscillation}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, ambient completeness, real completeness, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the displayed tail-window, modulus, tolerance, oscillation-ledger, and seal-handoff rows.

Chapter 449

A Concrete Naming Certificate for $\mathsf{RealScheduleFusion}^{\uparrow}$

The $\mathsf{RealScheduleFusion}^{\uparrow}$ chapter packages the finite schedule rows used by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ into one displayed NameCert surface. It keeps the window envelope, regular Cauchy readback, observer regularity, real seal, and completion handoff in a single finite packet, so downstream consumers read one row rather than reconstructing the same schedule from several neighboring certificates.

The dependency anchors are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, and $\mathsf{ObserverRegular}^{\uparrow}$. This is a composite packet over those sibling certificates: it bundles the StreamName schedule row, RegSeqRat readback row, Real seal row, Completion handoff row, finite-window envelope, regular-Cauchy name, and observer-regular provenance into one finite schedule-fusion NameCert surface.

The schedule row is read through the countable-window carrier of chapter 126; the regular-rational readback is consumed through the visible real-name packet of chapter 371; and the seal coordinate is handed to the RealUp surface of chapter 38. These sibling certificates are the only public routes by which the fusion packet exposes schedule, readback, and seal data to downstream completion consumers.

Definition 449.1 (Real schedule fusion carrier)**.** *A* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the intended finite stream windows and the consumer-facing schedule identity;*
2. *W is the finite window ledger, recording the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *windows, local window indices, and displayed window inclusions used by schedule consumers;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, pairing each displayed window with the regular-rational readback supplied by the* $\mathsf{RegularCauchyName}^{\uparrow}$ *side;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *seal row, recording the real-name seal and the completion-handoff seal reached from the same readback row;*
5. *H stores componentwise* hsame *transports for schedule rows, finite windows, regular-rational readbacks, seal rows, provenance, and the local naming row;*

6. *C stores the displayed* Cont *routes for schedule inspection, window selection, regular-rational readback, real sealing, completion handoff, and local NameCert comparison;*
7. *P is the* Pkg *provenance row naming* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{ObserverRegular}^{\uparrow}$, *and the consumed local source certificates;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealScheduleFusion}^{\uparrow}}$ *row.*

The classifier compares accepted packets by transporting them to a common schedule row, then comparing the finite window ledger, regular-rational readback row, real and completion seal row, provenance row, and local name. The carrier exports no quotient stream, selected limit, ambient completeness principle, host real equality, or completion object beyond the displayed finite rows.

Lean4 ✓ `BEDC.Derived.RealScheduleFusionUp.RealScheduleFusionTasteGate single carrier alignment`

Theorem 449.2 (Real schedule fusion NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealScheduleFusion}^{\uparrow}}$ *surface has five obligations:*

1. *carrier habitation: finite schedule rows, window ledgers, regular-rational readbacks, real and completion seals, transports, continuations, provenance, and local naming inhabit the carrier of Definition 449.1;*
2. *classifier transport: accepted packets are compared only after componentwise transport to a common displayed schedule and window surface;*
3. *schedule-readback lock: the* $\mathsf{StreamName}^{\uparrow}$ *schedule row and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row remain paired through every consumer route;*
4. *real-completion handoff:* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers receive the same finite readback and seal data;*
5. *ledger boundary: no consumer route escapes to quotient streams, selected limits, ambient completeness, host real equality, or a host completion witness.*

Proof. Carrier habitation is exactly the finite packet displayed in Definition 449.1. The rows name the schedule, window ledger, readback, real and completion seals, transports, continuations, provenance, and local certificate.

Classifier transport uses the row H to move every displayed coordinate to a common schedule surface. The continuation row C then supplies the visible routes by which a consumer reads the finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the seal row.

The dependency anchors $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, and $\mathsf{ObserverRegular}^{\uparrow}$ account for the source rows consumed by this packet. Since the carrier has no coordinate for quotient streams, selected limits, ambient completeness, host real equality, or a host completion witness, every obligation stays inside the finite schedule-fusion ledger. □

Theorem 449.3 (StreamName and RegSeqRat schedule lock)**.** *For an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier* $F = (S, W, R, E, H, C, P, N)$*, every* $\mathsf{StreamName}^{\uparrow}$ *schedule read of S determines the same finite window ledger W and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R used by downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers. The schedule row names the route; it does not create an independent tolerance, readback, quotient stream, or selected limit.*

Proof. Project the carrier of Definition 449.1. The only schedule-facing coordinate is S, and the only displayed windows are the finite entries of W.

The route from S to W and then to R is stored in C and transported componentwise by H. Thus a $\mathsf{StreamName}^{\uparrow}$ read cannot bypass the finite ledger or select a different $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The anchors $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ and $\mathsf{RegularCauchyName}^{\uparrow}$ identify the finite window and regular Cauchy readback sources, while $\mathsf{ObserverRegular}^{\uparrow}$ supplies the observer-style consumer discipline. None of these anchors adds a carrier coordinate for quotient streams, selected limits, or host equality. □

Theorem 449.4 (Real and completion handoff)**.** *For an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier* $F = (S, W, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal and every* $\mathsf{Completion}^{\uparrow}$ *handoff in* E *is read from the same* S*,* W*, and* R *rows through the stored* hsame *transports and* Cont *routes. The completion-facing consumer receives no data beyond the finite schedule, window, readback, seal, provenance, and local naming rows.*

Proof. Begin with a real or completion consumer route. The only seal coordinate is E, so the route must reach E through the displayed schedule, window, and readback rows described in Definition 449.1.

The $\mathsf{Real}^{\uparrow}$ seal is tied to the $\mathsf{RegSeqRat}^{\uparrow}$ readback row by the continuation routes in C. The $\mathsf{Completion}^{\uparrow}$ handoff uses the same route and the same transported rows, so it cannot introduce a separate completion witness.

The anchors $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ specify the consumer side, and $\mathsf{RegularCauchyName}^{\uparrow}$ supplies the readback source. Because the carrier omits selected limits, ambient completeness, quotient streams, and host real equality, the handoff remains finite. □

Theorem 449.5 (Real schedule fusion ledger non-escape)**.** *Every consumer-facing route out of an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier factors through the finite schedule row* S*, window ledger* W*, regular-rational readback row* R*, real and completion seal row* E*, componentwise transport row* H*, continuation row* C*, package provenance* P*, and local naming row* N*. No route exports a quotient stream, selected limit, ambient completeness proof, host real equality, or host completion witness.*

Proof. Take any consumer-facing route and project the finite carrier. The available coordinates are exactly S, W, R, E, H, C, P, and N.

The schedule, window, readback, and seal rows are the only rows that carry mathematical content to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{Completion}^{\uparrow}$. The remaining rows transport, route, record provenance, and name the packet.

Therefore the route is exhausted by the displayed finite ledger. Since the carrier has no coordinate for quotient streams, selected limits, ambient completeness, host real equality, or a host completion witness, none can be exported. □

Theorem 449.6 (Real schedule fusion standard boundary)**.** *The standard boundary of* $\mathsf{RealScheduleFusion}^{\uparrow}$ *is the finite packet* $F = (S, W, R, E, H, C, P, N)$ *and the dependency anchors* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{RegularCauchyName}^{\uparrow}$*, and* $\mathsf{ObserverRegular}^{\uparrow}$*. Any public comparison between accepted packets is a comparison of those displayed rows after* hsame *transport; it is not a comparison of quotient streams, selected host limits, ambient completeness objects, or host real equality.*

Proof. The public boundary is read from the carrier definition. A comparison between accepted packets first transports the schedule, window, readback, seal, provenance, and naming rows by H.

After transport, the comparison uses the routes in C to inspect the finite rows named by the dependency anchors. The anchors identify the source certificates, but the packet does not expose their host interpretations or any completion object outside the finite ledger.

Thus the standard boundary is precisely the displayed row comparison. Quotient streams, selected host limits, ambient completeness objects, and host real equality are absent from the carrier and cannot appear in the public comparison. □

Theorem 449.7 (Real schedule fusion TasteGate obligation surface)**.** *The* ChapterTasteGate *surface for* RealScheduleFusion$^{\uparrow}$ *is exactly the single finite carrier* $F = (S, W, R, E, H, C, P, N)$ *of Definition 449.1. Its visible rows are the* StreamName$^{\uparrow}$ *schedule row* S*, the finite window ledger* W*, the* RegSeqRat$^{\uparrow}$ *readback row* R*, the* Real$^{\uparrow}$ *and* Completion$^{\uparrow}$ *seal row* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and the local* NameCert$_{\text{RealScheduleFusion}^{\uparrow}}$ *row* N*.*

Consequently any promotion of this chapter beyond seed strength must certify this same carrier surface: no carrier coordinate may be duplicated as a proof step, and no hidden quotient stream, selected limit, ambient completeness proof, host real equality, or host completion witness may be used as a substitute for one of the displayed rows.

Lean4 ✓ `BEDC.Derived.RealScheduleFusionUp.taste gate`

Proof. The carrier is the tuple displayed in Definition 449.1. Its local naming obligations are the five rows of Definition 449.2, so the TasteGate surface has a single finite object to inspect.

The schedule lock is Definition 449.3. The real and completion handoff is Definition 449.4. Together they force every schedule, readback, real-seal, and completion-facing read to pass through the rows S, W, R, E, H, C, P, N.

The ledger boundary is Definition 449.5, and the public boundary is Definition 449.6. These two boundaries rule out carrier-echo proof text: a proof may cite the finite rows and their routes, but it cannot replace a missing row by repeating the carrier description. □

Theorem 449.8 (Real schedule fusion TasteGate obligation closure)**.** *The carrier of Definition 449.1, the NameCert obligations of Definition 449.2, the schedule lock of Definition 449.3, the real and completion handoff of Definition 449.4, the ledger non-escape theorem of Definition 449.5, the standard boundary of Definition 449.6, and the TasteGate surface of Definition 449.7 form the complete obligation closure for the single* RealScheduleFusion$^{\uparrow}$ *carrier checked by* RealScheduleFusionTasteGate_singl

Proof. Use the checked TasteGate marker attached to Definition 449.1. It fixes the chapter to the single carrier surface inspected by the TasteGate, so the closure cannot introduce a second carrier row or an alternate schedule object.

The finite ledger rows are supplied by the displayed theorems. The NameCert obligation rows give carrier habitation, classifier transport, schedule-readback lock, real-completion handoff, and ledger boundary. The schedule-lock, handoff, non-escape, and standard-boundary theorems then account for each consumer-facing route named by those obligations.

Finally apply the TasteGate surface theorem. It states that any stronger chapter reading must certify the same finite carrier surface rather than repeat the carrier as a proof substitute. The listed rows therefore exhaust the obligation surface for the single carrier. □

Theorem 449.9 (Real schedule fusion schedule-lock consumer scope)**.** *For an accepted* RealScheduleFusion$^{\uparrow}$ *carrier* $F = (S, W, R, E, H, C, P, N)$*, every* StreamName$^{\uparrow}$ *consumer route through the schedule row* S *reaches the finite window ledger* W *and the* RegSeqRat$^{\uparrow}$ *readback row* R *only by the displayed* Cont *routes in* C *and the componentwise* hsame *transports in* H*. The scope contains no second schedule carrier and no alternate readback row.*

Proof. Read the carrier of Definition 449.1. It has one schedule coordinate S, one finite window coordinate W, and one regular-rational readback coordinate R.

The continuation row C stores the visible route from schedule inspection to window selection and then to regular-rational readback. The transport row H is the only mechanism by which another displayed packet can be compared with those rows.

Thus a $\mathsf{StreamName}^{\uparrow}$ consumer may reach W and R only through the stored route and transport witnesses. Since the carrier has no second schedule coordinate and no second readback coordinate, no alternate schedule carrier can be read by this chapter. □

Theorem 449.10 (Real and completion consumer scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier reads the same seal row E only after S, W, and R have been fixed by the schedule-lock scope. The real-facing and completion-facing reads therefore share the finite schedule, window, and regular-rational readback rows before any seal or handoff is exposed.*

Proof. Begin with a real-facing or completion-facing route. The only seal coordinate in the carrier is E, so the route must arrive at that row from the displayed schedule, window, and readback coordinates.

Apply Definition 449.9. It fixes S, W, and R before any downstream consumer can inspect the seal row.

The $\mathsf{Real}^{\uparrow}$ consumer reads the real seal from E, and the $\mathsf{Completion}^{\uparrow}$ consumer reads the completion handoff from the same E row. Because both reads share the fixed upstream rows, neither consumer can introduce a separate completion witness, selected limit, quotient stream, or host equality. □

Theorem 449.11 (Real schedule fusion scoped dependency closure)**.** *The* $\mathsf{RealScheduleFusion}^{\uparrow}$ *scoped dependency surface is the schedule-lock consumer scope of Definition 449.9, the real and completion consumer scope of Definition 449.10, and the checked TasteGate surface of Definition 449.7. It closes the current Real and Completion dependency row for the single finite carrier without adding a quotient stream, selected limit, ambient completeness proof, host real equality, or second schedule carrier.*

Proof. First use the schedule-lock consumer scope. It forces every $\mathsf{StreamName}^{\uparrow}$ read through S to reach W and R by the displayed continuation routes and componentwise transports.

Next use the real and completion consumer scope. It says that $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ both read the seal row E only after the same S, W, and R rows are fixed.

Finally apply the checked TasteGate surface. The TasteGate pins all of these rows to the single finite carrier $F = (S, W, R, E, H, C, P, N)$, so the scoped dependency closure has no place for an alternate schedule object, hidden completion witness, or host-side equality. □

Theorem 449.12 (Real schedule fusion window-merge scope)**.** *For an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier $F = (S, W, R, E, H, C, P, N)$, any consumer merge of* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *rows must pass through the displayed schedule row S, finite window ledger W, regular-rational readback row R, componentwise transport row H, continuation row C, package provenance P, and local naming row N. The merge exports no second schedule carrier, hidden window envelope, quotient stream, selected limit, or host equality.*

Proof. Start from the carrier of Definition 449.1. It displays one schedule coordinate S, one finite window coordinate W, one readback coordinate R, one transport row H, one continuation row C, one provenance row P, and one local naming row N.

Apply Definition 449.7. The checked TasteGate surface fixes these rows as the single visible carrier inspected by the chapter.

Apply Definition 449.9. A $\mathsf{StreamName}^{\uparrow}$ route can reach W and R only through the displayed Cont routes and componentwise hsame transports.

Apply Definition 449.10 and Definition 449.11. They keep downstream real and completion consumers on the same upstream schedule and window rows, so a window merge cannot introduce a second schedule carrier or hidden host-side equality. □

Theorem 449.13 (Real schedule fusion seal-schedule noninterference)**.** *For an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier* $F = (S, W, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read and every* $\mathsf{Completion}^{\uparrow}$ *handoff read of* E *leaves the upstream schedule row* S*, finite window ledger* W*, regular-rational readback row* R*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *unchanged as the displayed readback surface. Seal inspection cannot alter the schedule, window, or readback rows it consumes.*

Proof. Read the finite packet in Definition 449.1. The seal coordinate is E, while the upstream readback surface is S, W, R, H, C, P, N.

Apply Definition 449.9. It fixes the schedule row, finite window ledger, and regular-rational readback before a downstream consumer reaches the seal coordinate.

Apply Definition 449.10. It states that both real-facing and completion-facing reads inspect the same E only after the same S, W, R rows have been fixed.

Apply Definition 449.7 and Definition 449.11. The checked TasteGate and scoped closure leave no coordinate by which a seal read could rewrite the upstream schedule, window, readback, transport, route, provenance, or naming rows. □

Theorem 449.14 (Real schedule fusion ObserverRegular handoff)**.** *Every* $\mathsf{ObserverRegular}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealScheduleFusion}^{\uparrow}$ *carrier reads only the displayed observer-regularity rows already packaged by* P *together with the schedule row* S*, finite window ledger* W*, regular-rational readback row* R*, transport row* H*, continuation row* C*, and local naming row* N*. The handoff exposes observer regularity as finite provenance and route data, not as an ambient regularity predicate outside the carrier.*

Proof. Project the carrier of Definition 449.1. The dependency anchors include $\mathsf{ObserverRegular}^{\uparrow}$, and their consumer-facing record is carried by the provenance row P together with the displayed routes in C.

Apply Definition 449.7. It identifies the single carrier surface as S, W, R, E, H, C, P, N, so observer regularity can be read only through rows already present in that surface.

Apply Definition 449.9. The observer handoff shares the same schedule, window, and regular-rational readback rows used by schedule consumers.

Apply Definition 449.10 and Definition 449.11. Downstream real and completion reads therefore do not enlarge the observer handoff into an ambient regularity predicate outside the finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealScheduleFusion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealScheduleFusion}^{\uparrow}$ packet is generated by a $\mathsf{StreamName}^{\uparrow}$ schedule row, a finite window ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealScheduleFusion}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 449.12, Definition 449.13, and Definition 449.14 bind the window-merge, seal-schedule, and ObserverRegular consumer scopes to the checked TasteGate surface of Definition 449.7 with BEDC.Derived.RealScheduleFusionUp.taste_gate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealScheduleFusion}^{\uparrow}}, \mathsf{axiomClean})$

Bridge status: none

Origin: ai

Lean target: `BEDC.Derived.RealScheduleFusionUp.RealScheduleFusionTasteGate\_single\_carrier\_alignment`

Not claimed: This chapter does not close quotient streams, selected limits, ambient completeness, host real equality, host completion witnesses, or bridge checking.

Upgrade path: Scoped dependency closure requires checked finite-ledger theorems tying the schedule lock, real and completion handoff, non-escape, standard boundary, and TasteGate surface to downstream Real and Completion consumers.

Chapter 450

A Concrete Naming Certificate for $\mathsf{RealCompleteness}^{\uparrow}$

The $\mathsf{RealCompleteness}^{\uparrow}$ interface records the diagonal-limit target of chapter 9238 as a finite naming-certificate packet. It packages the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ observations, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ regular sequence rows, and $\mathsf{Real}^{\uparrow}$ seal row through which a completion consumer sees the candidate limit. The packet is a seed carrier for the Real and Completion substrate; it contains no quotient real, selected host limit, choice principle, or host completeness theorem.

Definition 450.1 (Real completeness finite carrier)**.** *A* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family of rational regular sequences;*
2. *μ is the explicit Cauchy modulus row naming, for each requested rational tolerance, a finite stage after which the family is stable;*
3. *Δ is the diagonal selector row, choosing the finite stage and rational observation used at each precision request;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger containing the dyadic endpoints, radii, comparison rows, and tolerance rows read by the diagonal selector;*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family through which the selected rational names are inspected;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row tying the selected windows to the regular rational sequence data;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row that presents the same diagonal observations as the candidate limit name;*
8. *H stores displayed* hsame *transports for the input family, modulus, selector, dyadic ledger, stream windows, readbacks, and seal row;*
9. *C stores the* Cont *routes by which consumers read the modulus, selected diagonal observations, and sealed candidate;*
10. *P is the* Pkg *provenance row over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *registration row.*

The local classifier compares two carriers only through these displayed rows. It does not identify quotient representatives, does not choose an external real number, and does not appeal to ambient completeness.

Theorem 450.2 (Real completeness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier formation for the finite packet of Definition 450.1;*
2. *classifier stability for the transported family X, modulus μ, diagonal selector Δ, observation ledger D, windows W, readback row Q, and seal row S;*
3. *modulus exactness, requiring every convergence bound consumed by the diagonal selector to be the explicit row recorded by μ;*
4. *diagonal readback exactness, requiring every selected rational observation to pass through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger;*
5. *package closure, requiring the* hsame*,* Cont*,* Pkg*, and* NameCert *rows to contain exactly the provenance used by downstream* $\mathsf{Real}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier namecert obligations`

Proof. Carrier formation is the tuple assembly of Definition 450.1. Each dependency row named by the diagonal-limit target is a visible coordinate of R.

Classifier stability is componentwise. Transport the regular sequence family and modulus first, then transport the selected stream windows and dyadic observation ledger, and finally transport the $\mathsf{Real}^{\uparrow}$ seal row reached by those observations.

The modulus and diagonal readback clauses are projections of the same packet. The row μ supplies the finite stage bounds, while Δ, W, Q, and D record exactly which rational observations are read at each precision request.

Package closure follows by projecting H, C, P, and N. No consumer route reaches a quotient real, external limit selector, or host completeness theorem because no such coordinate occurs in the carrier. □

Theorem 450.3 (Real completeness diagonal witness schedule)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. For every precision request read by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer, the diagonal witness schedule is a finite* BHist *route inside R: the request is bounded by the displayed modulus row μ, selected by Δ, read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, transported by the* hsame *rows in H, routed by* Cont *rows in C, and packaged by the* Pkg *and* NameCert *rows P, N before it reaches the seal row S. The schedule contains no quotient real, external limit choice, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier diagonal witness schedule`

Proof. Project the accepted carrier through Definition 450.1. The displayed tuple already contains the family row, modulus, diagonal selector, dyadic ledger, stream windows, regular readback, seal row, transports, continuations, package provenance, and local naming row.

Apply Definition 450.2. The modulus exactness clause attaches each consumed bound to μ, while the diagonal readback exactness clause forces every selected rational observation through Δ, D, W, Q.

It remains to expose the public route. The package-closure clause projects H, C, P, N, so the witness schedule is the finite BHist route recorded in the carrier itself. Since no external limit selector or completeness theorem is a coordinate of R, none can contribute a witness to the schedule. □

Theorem 450.4 (Real completeness tail budget stability)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted RealCompleteness$^{\uparrow}$ *carriers whose family, modulus, diagonal selector, dyadic ledger, stream-window, readback, and seal rows are related by the displayed* hsame *transports in the shared row H, and whose consumer routes use the same* Cont, Pkg, *and* NameCert *rows C, P, N. Then every tail budget used by Δ in R is transported to the corresponding tail budget used by Δ' in R', and both budgets are read through the same finite* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *route. Thus tail-budget stability is a property of the displayed packet rows and not of host real equality or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier tail budget stability`

Proof. Read both accepted carriers by their displayed coordinates. The only rows available for a tail-budget read are the regular family, modulus, selector, dyadic ledger, stream windows, readback row, seal row, transports, continuations, package provenance, and local naming row.

Use the classifier-stability clause of Definition 450.2. It transports $X, \mu, \Delta, D, W, Q, S$ componentwise along the displayed hsame data in H, while the shared C, P, N rows keep the consumer route fixed.

The modulus exactness and diagonal readback exactness clauses then identify the transported budget with the same finite readback path. Every consumed bound is still a μ or μ' row, and every selected observation still passes through the corresponding D, W, Q or D', W', Q' rows before the seal is read. No host equality or completeness coordinate is present to alter the transported budget. □

Theorem 450.5 (Real completeness RegSeq-to-seal handoff)**.** *For every accepted* RealCompleteness$^{\uparrow}$ *carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

the handoff from the RegSeqRat$^{\uparrow}$ *data to the* Real$^{\uparrow}$ *seal is completely determined by the finite rows of R. The regular family X supplies the rational names, the explicit modulus μ bounds the selected tail, Δ chooses the displayed diagonal observations, D records the* DyadicRatCore$^{\uparrow}$ *ledger, W exposes the* StreamName$^{\uparrow}$ *windows, Q performs the* RegSeqRat$^{\uparrow}$ *readback, and H, C, P, N route the handoff to the seal row S through* hsame, Cont, Pkg, *and* NameCert.

Proof. Begin with the carrier definition. It makes $X, \mu, \Delta, D, W, Q, S, H, C, P, N$ the whole naming surface for the diagonal-limit candidate.

Apply Definition 450.2. Carrier formation supplies the finite rows, modulus exactness binds the selected tail to μ, and diagonal readback exactness binds the selected observations to D, W, Q.

The final handoff is package closure. Project H, C, P, N and read the seal row S as the unique $\mathsf{Real}^{\uparrow}$ endpoint named by the carrier. The construction is therefore a handoff from $\mathsf{RegSeqRat}^{\uparrow}$ readbacks to the displayed seal, with no quotient-style real equality or host completeness argument. □

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier regseq seal handoff`

Theorem 450.6 (Real completeness public witness extractor obligation)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. Any public witness extractor for the diagonal-limit surface of R has the following obligation surface: it may inspect only the* BHist *rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$; it must read the rational witness through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows; it must route transports through* hsame*, continuations through* Cont*, provenance through* Pkg*, and local registration through* $\mathsf{NameCert}$*; and it must return only the* $\mathsf{Real}^{\uparrow}$ *seal row S as the completion-facing endpoint.*

Proof. First expose the finite public boundary. The carrier definition contains no hidden extractor field, so any public extractor can only project rows from the displayed BHist packet.

Apply Definition 450.2. The diagonal readback exactness clause forces rational witnesses through D, W, Q, and the modulus exactness clause forces every tail bound through μ.

Finally use package closure to project H, C, P, N. These rows are exactly the hsame, Cont, Pkg, and $\mathsf{NameCert}$ route by which the selected witness reaches S. Since S is the only $\mathsf{Real}^{\uparrow}$ seal coordinate in the carrier, the extractor has no public endpoint other than that seal row. □

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier witness extractor public obligation`

Theorem 450.7 (Real completeness finite obligation surface)**.** *For an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

all formalization-facing obligations for the local diagonal-limit certificate are exhausted by the finite packet rows: carrier formation for R; hsame *classifier stability for $X, \mu, \Delta, D, W, Q, S$ through H; modulus exactness for μ; diagonal readback exactness through Δ, D, W, Q over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*; and package closure through the* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows C, P, N to the* $\mathsf{Real}^{\uparrow}$ *seal S. No obligation mentions quotient equality of reals, selected host limits, countable choice, or an ambient completeness theorem.*

Proof. Project Definition 450.2. Its five clauses are already finite: carrier formation, classifier stability, modulus exactness, diagonal readback exactness, and package closure.

Now identify the rows consumed by those clauses. Carrier formation gives R; classifier stability reads H; modulus exactness reads μ; diagonal readback exactness reads Δ, D, W, Q over the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces; and package closure reads C, P, N before reaching the $\mathsf{Real}^{\uparrow}$ seal S.

The exclusion statement follows from the same projection. Quotient real equality, selected host limits, countable choice, and ambient completeness are absent from the packet, so they cannot appear among the obligations exposed by the local $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ surface. □

Theorem 450.8 (Real completeness dependency lattice)**.** *For every accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

the accepted completion route factors through the upstream packet lattice in the following order: $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledgers in* D*,* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks in* Q*,* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *selected-tail witnesses, and the final* $\mathsf{Real}^{\uparrow}$ *seal row* S*. The rows* H, C, P, N *preserve the same route through* hsame*,* Cont*,* Pkg*, and* NameCert*. The route refuses quotient real equality, selected host limits, countable choice, and ambient completeness.*

Proof. Begin with Definition 450.1. The displayed carrier already separates the dyadic ledger D, stream windows W, regular readback row Q, seal row S, and the transport, continuation, provenance, and naming rows H, C, P, N.

Apply Definition 450.2 and Definition 450.7. Their diagonal readback clauses force selected observations through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows, while package closure routes the result through H, C, P, N before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Now insert the finite-tail handoff supplied by chapter 559. Its selected-tail witness package sits exactly between the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal, using the same precision, window, source, witness, transport, route, provenance, and local naming discipline.

The sibling chapters chapter 126, chapter 301, chapter 291, and chapter 38 name the upstream rows consumed by this lattice. They provide the window, regular-rational, dyadic, and real-seal anchors without adding a quotient real or a host completeness principle.

Finally use Definition 450.17. Since quotient real equality, selected host limits, countable choice, and ambient completeness are absent from the carrier, they cannot appear in the dependency lattice. □

Theorem 450.9 (Real completeness modulus domain coverage)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. Every modulus request consumed by the diagonal selector* Δ *is a displayed unary request for a rational tolerance in the modulus row* μ*. The selected request is read through the finite route* D, W, Q, H, C, P, N*: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger supplies the rational tolerance and observation rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row expose the selected rational names, and the transport, continuation, package, and local naming rows record the public route. No modulus request is supplied by a quotient stream, selected host limit, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier modulus domain coverage`

Proof. Read the carrier coordinates in Definition 450.1. The rows μ and Δ are displayed separately: μ names the tolerance-indexed stage requests, while Δ consumes only finite selected stages and rational observations.

Apply Definition 450.2. Its modulus exactness clause binds every convergence bound consumed by the selector to the explicit μ row, and its diagonal readback exactness clause routes the selected rational observation through D, W, Q.

The public route is completed by the rows H, C, P, N. By Definition 450.13, those rows are precisely the path by which the finite diagonal observation reaches the $\mathsf{Real}^{\uparrow}$ candidate. By Definition 450.17, quotient streams, selected host limits, countable choice, and ambient completeness are not coordinates of this route. □

Theorem 450.10 (Real completeness selector readback determinacy)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carriers whose* X, μ, Δ, D, W, Q *rows are related by the displayed transports named in the shared* H *row and whose consumer reads use the same* C, P, N *route. Then two selector reads using those transported rows reach the same finite* $\mathsf{Real}^{\uparrow}$ *seal ledger. In particular, selector readback determinacy is a statement about the displayed rows* $X, \mu, \Delta, D, W, Q, H, C, P, N$ *and not about a quotient stream, selected host limit, countable choice, or ambient completeness theorem.*

Proof. Start with the two carrier displays. The carrier definition exposes exactly the rows that the selector can inspect: family, modulus, selector, dyadic ledger, stream windows, regular readback, transport rows, continuations, package provenance, and local naming.

Use Definition 450.2 for classifier stability. The shared H row transports X, μ, Δ, D, W, Q, while the shared C, P, N route keeps the consumer-facing readback path fixed.

Now apply Definition 450.13 and Definition 450.16. A selector read reaches $\mathsf{Real}^{\uparrow}$ only as the finite seal ledger formed from Δ through D, W, Q and then routed through H, C, P, N. Since both reads use the same transported rows and the same route, they reach the same ledger.

Finally Definition 450.17 rules out any external separator. There is no quotient stream, selected host limit, countable-choice row, or host completeness coordinate that could split the two readbacks after the finite route has been fixed. □

Theorem 450.11 (Real completeness consumer route totality)**.** *For every accepted carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

each completion-facing consumer read for $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *factors through exactly the displayed rows* $X, \mu, \Delta, D, W, Q, S, H, C, P, N$*. The factorization reads the regular family, explicit modulus, diagonal selector, dyadic ledger, stream windows, regular readback,* $\mathsf{Real}^{\uparrow}$ *seal, transports, continuations, package provenance, and local naming row, and it exposes no quotient stream, selected host limit, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier consumer route totality`

Proof. Begin with Definition 450.1. It lists the complete carrier surface, and no completion-facing row occurs outside $X, \mu, \Delta, D, W, Q, S, H, C, P, N$.

Apply Definition 450.2. Carrier formation and package closure make the displayed rows the full public route, while modulus exactness and diagonal readback exactness force any selected observation to pass through μ, Δ, D, W, Q before it can reach the seal row.

For $\mathsf{Real}^{\uparrow}$ reads, use Definition 450.13 and Definition 450.16: the candidate is the seal row S read through the finite diagonal route. For $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{CauchySequenceSpace}^{\uparrow}$ reads, use Definition 450.18 and Definition 450.22: the downstream consumer first reads the same finite seal route and then its displayed finite handoff rows.

The route is therefore total over the displayed carrier rows and nowhere else. By Definition 450.17, quotient streams, selected host limits, host equality, countable choice, and ambient completeness are absent from the public boundary. □

Theorem 450.12 (Real completeness public seal export)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. The public export consumed by* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *is exactly the seal row* S *reached through the displayed rows*

$$\mu, \Delta, D, W, Q, H, C, P, N.$$

Any two consumer reads with the same displayed rows reach the same seal ledger. The export does not claim quotient stream equality, selected host limits, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier public seal export`

Proof. Begin with consumer route totality. By Definition 450.11, every $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or $\mathsf{CauchySequenceSpace}^{\uparrow}$ read factors through the displayed carrier rows and no external completion object.

Identify the endpoint. Apply Definition 450.13 and Definition 450.5; the diagonal observations selected by Δ pass through D, W, Q and reach the public seal row S through μ, H, C, P, N.

Now compare two consumer reads with the same displayed rows. The selector readback determinacy theorem Definition 450.10 fixes the finite readback route, so both reads arrive at the same seal ledger.

Finally apply the non-escape boundary. By Definition 450.17, quotient streams, selected host limits, countable choice, and ambient completeness are not coordinates of the packet, and therefore cannot be exported by the public seal row. □

Theorem 450.13 (Real completeness diagonal limit candidate)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. The candidate limit exported to* $\mathsf{Real}^{\uparrow}$ *is the seal row* S *obtained from the selected diagonal observations* Δ *through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*. Every convergence read used by this candidate factors through the explicit modulus row* μ *and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start from the NameCert obligations of Definition 450.2. They bind the modulus row, diagonal selector, finite windows, regular-sequence readbacks, dyadic ledger, and seal row to the same carrier.

The diagonal selector Δ chooses only recorded stages and observations. The selected reads therefore pass through W, Q, and D before the seal row S is formed.

The convergence-facing part is controlled by μ. Since all consumer routes are stored in H, C, P, and N, the exported candidate is exactly the finite diagonal-limit surface seen by $\mathsf{Real}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier diagonal limit candidate`

Theorem 450.14 (Real completeness carrier transport)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. Any componentwise displayed* hsame *transport of the concrete packet rows* $X, \mu, \Delta, D, W, Q, S, H, C, P, N$ *that preserves the* Cont *routes in* C*, the* Pkg *provenance in* P*, and the local* NameCert *row* N *yields another accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier with the same public diagonal-limit surface.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier carrier transport`

Proof. Use the carrier definition to read every coordinate of R. The transported packet has the same row shape: regular sequence family, modulus row, diagonal selector, dyadic ledger, stream windows, readback row, $\mathsf{Real}^{\uparrow}$ seal, transport rows, continuation routes, package provenance, and local naming row.

Apply Definition 450.2. Its classifier-stability clause transports $X, \mu, \Delta, D, W, Q, S$ componentwise by hsame, while its package-closure clause keeps H, C, P, N attached to the same public routes.

The conclusion is therefore a repacking of the displayed rows. No ambient completeness theorem is invoked, because acceptance is checked by the finite carrier and the local NameCert obligations. □

Theorem 450.15 (Real completeness modulus diagonal exactness)**.** *For every accepted packet*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

each diagonal observation exported by $\mathsf{RealCompleteness}^{\uparrow}$ *is selected by* Δ*, bounded by the explicit modulus row* μ*, read through the* BHist *ledgers* D, W, Q*, transported by* hsame *rows in* H*, routed through* Cont *rows in* C*, and packaged by* P *and* N*. No exported diagonal read may use a modulus or rational observation absent from the concrete packet.*

Proof. Begin with Definition 450.13. It identifies the exported $\mathsf{Real}^{\uparrow}$ candidate with the seal row S obtained from Δ through W, Q, and D.

Apply Definition 450.2. Its modulus exactness clause binds every convergence bound to μ, and its diagonal readback exactness clause binds every selected rational observation to the displayed stream, readback, and dyadic rows.

The remaining rows H, C, P, N record transport, continuation, package, and naming discipline. Hence exactness is a finite ledger property of R, not a claim about host completeness. □

Theorem 450.16 (RealUp seal ledger discipline)**.** *For every accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *seal row* S *is the only completion-facing export. Its reads must pass through the displayed* BHist *packet rows* X, μ, Δ, D, W, Q*, the* hsame *transport row* H*, the* Cont *route row* C*, the* Pkg *row* P*, and the* NameCert *row* N*.*

Proof. Project the carrier through Definition 450.1. The only row typed as the candidate limit name is S, and the preceding rows record the family, modulus, selector, dyadic observations, stream windows, and regular-sequence readbacks that form it.

Use Definition 450.2 to keep the seal attached to the same classifier-stable rows. The package-closure clause says the public routes are exactly the displayed hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 450.13. It identifies the seal with the finite diagonal observations, so the $\mathsf{Real}^{\uparrow}$ boundary contains no selected external limit, quotient real, host equality, or host completeness proof. □

Theorem 450.17 (Real completeness no-escape boundary)**.** *The public boundary of* $\mathsf{RealCompleteness}^{\uparrow}$ *exports only the finite diagonal-limit candidate over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It exports no quotient real, no host equality on*

real numbers, no selected external limit, no countable-choice principle, and no host completeness theorem.

Proof. Inspect the carrier coordinates of Definition 450.1. The only completion-facing rows are the modulus, diagonal selector, finite stream windows, regular-sequence readbacks, dyadic observation ledger, and $\mathsf{Real}^{\uparrow}$ seal row.

Apply Definition 450.13. Every candidate-limit read is downstream of those displayed rows and their transport, continuation, package, and local naming data.

The excluded objects are absent from the tuple. Therefore the boundary cannot export a quotient real, host real equality, external selected limit, choice principle, or host completeness theorem. □

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier nonescape boundary`

Theorem 450.18 (Real completeness scoped finite seal pullback)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit read factors through the explicit modulus row, diagonal selector,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal, and downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *finite seal route displayed by the accepted carrier. The pullback exports no quotient stream, selected host limit, ambient completeness theorem, or host equality on completed reals.*

Proof. Start with the NameCert obligations of Definition 450.2. They bind carrier formation, classifier stability, modulus exactness, diagonal readback exactness, and package closure to the finite rows of the accepted packet.

Apply Definition 450.13 and Definition 450.15. The candidate read is selected by Δ, bounded by μ, and read through D, W, Q before it reaches the $\mathsf{Real}^{\uparrow}$ seal row.

Next use Definition 450.16. It identifies the $\mathsf{Real}^{\uparrow}$ seal as the only completion-facing export and keeps every consumer route inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ finite seal route consumes exactly that seal-facing row: explicit modulus, selected diagonal observations, stream windows, regular rational readback, dyadic ledger, and the same package provenance. The route is therefore a pullback along the finite seal consumer rather than a choice of an external completed object.

Finally apply Definition 450.17. Since quotient streams, selected host limits, ambient completeness, and host equality are absent from the carrier, they cannot appear in the scoped pullback. □

Theorem 450.19 (Real completeness candidate-uniqueness boundary)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packets whose* X, μ, Δ, D, W, Q *rows are related by the displayed* hsame *transports named in the same* H *row, and whose seal rows* S *and* S' *are produced through the same diagonal readback routes* C*, package provenance* P*, and local naming row* N*. Then every* $\mathsf{Real}^{\uparrow}$ *consumer classifies* S *and* S' *through the same displayed* $\mathsf{Real}^{\uparrow}$ *seal ledger. In particular, the packet cannot export two distinct completion-facing seal reads for the same transported diagonal family and the same readback routes.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier candidate uniqueness boundary`

Proof. Apply Definition 450.2. Its classifier-stability clause transports the family, modulus, selector, dyadic ledger, windows, readback row, and seal row through the displayed rows of H.

Next use Definition 450.14 and Definition 450.15. The transported packets have the same public diagonal-limit surface, and every convergence bound and selected rational observation is read from μ, Δ, D, W, Q through the same routes.

Now apply Definition 450.16. A $\mathsf{Real}^{\uparrow}$ consumer may inspect a seal only through the displayed packet rows and the shared H, C, P, N ledger. Since both seal rows are produced through those same diagonal readback routes, the consumer reaches the same displayed $\mathsf{Real}^{\uparrow}$ seal ledger.

Finally invoke Definition 450.17. There is no quotient real, host equality, selected external limit, or host completeness row that could separate S from S' outside the displayed ledger. The only exported classification is therefore the common finite seal-ledger classification. □

Theorem 450.20 (CauchyLimitSeal transport stability for real completeness)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packets whose* X, μ, Δ, D, W, Q *rows are related by the displayed* hsame *transports named in the shared* H *row, and whose seal rows* S *and* S' *are produced through the same continuation routes* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *row* N*. Then every downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of either seal factors through the same finite route: the explicit modulus, diagonal selector,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal, displayed* hsame *rows,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. Hence the transported diagonal-limit seal is stable not only for* $\mathsf{Real}^{\uparrow}$ *consumers but also for the finite* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route that consumes the seal.*

Proof. First use Definition 450.14. The componentwise hsame rows in H transport the visible family, modulus, selector, dyadic ledger, stream windows, readback row, and seal row while preserving the shared C, P, N routing data, so both packets present the same public diagonal-limit surface.

Next apply Definition 450.15. Every selected read used by either seal is bounded by the corresponding explicit modulus row and passes through Δ, D, W, Q or through their transported counterparts. Since those rows are tied by the same H and consumed through the same C, P, N data, the diagonal selector and rational readback route are the same finite route seen through transport.

Use Definition 450.16 to identify the $\mathsf{Real}^{\uparrow}$ seal as the unique completion-facing export for each packet. Then use Definition 450.18: any $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of that export reads the explicit modulus, selected diagonal observations, dyadic ledger, stream windows, regular-sequence readback, seal row, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

It remains to rule out a second downstream route. By Definition 450.19, the two seal rows have the same displayed $\mathsf{Real}^{\uparrow}$ seal-ledger classification for the transported diagonal family and shared readback routes. Finally Definition 450.17 excludes quotient streams, host real equality, selected external limits, choice principles, and host completeness rows. Thus any $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of either seal factors through the same finite route. □

Theorem 450.21 (Real completeness seal classifier stability)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packets. Suppose the family row, modulus row, diagonal selector, dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of* R *and* R' *are related by the displayed* hsame *transports named in the shared* H *row, and suppose the seal rows* S *and* S' *are read through the same* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *route. Then the* $\mathsf{Real}^{\uparrow}$ *classifier class exported by* S *is the same finite seal-ledger class exported by* S'*. Moreover, every downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read of either seal sees that same class through the displayed route consisting of* μ, Δ, D, W, Q, S*, the transported counterparts,* H, C, P, N*, and no ambient completion object.*

Proof. Apply Definition 450.14. The displayed hsame rows give componentwise transport of the family, modulus, diagonal selector, dyadic ledger, stream windows, regular rational readback, and seal rows while keeping the shared continuation, package, and local naming data fixed.

Next use Definition 450.15. The finite observations that feed the two seal rows are exactly the bounded diagonal reads selected by Δ and Δ' and recovered through D, W, Q and their transported counterparts.

Apply Definition 450.16. The only $\mathsf{Real}^{\uparrow}$ classifier endpoint available to either packet is the seal ledger exposed by the accepted carrier, so the two seal rows export the same finite seal-ledger class under the shared H, C, P, N route.

Now invoke Definition 450.20. A downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reads no separate completion data; it consumes the explicit modulus, diagonal selector, dyadic ledger, stream windows, regular readback, seal row, displayed transports, continuations, package provenance, and local naming row already identified above.

Finally use Definition 450.17. Quotient streams, selected host limits, host equality, countable choice, and ambient completeness are not coordinates of the packet, so they cannot split the exported classifier class or create a second downstream seal view. □

Theorem 450.22 (RealCompleteness shared-window handoff from CauchySequenceSpace)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet whose regular-family row is presented by an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet*

$$T = (F, \sigma, W_c, \epsilon, Q_c, H_c, C_c, P_c, N_c).$$

Then every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer read of the seal row* S *factors first through the displayed* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *rows*

$$F, \sigma, W_c, \epsilon, Q_c, H_c, C_c, P_c, N_c$$

and then through the displayed $\mathsf{RealCompleteness}^{\uparrow}$ *rows*

$$\mu, \Delta, D, W, Q, S, H, C, P, N.$$

The handoff exports no quotient stream, selected host limit, host real equality, or ambient completeness coordinate.

Proof. Begin with the shared finite Cauchy window. By Definition 406.11, any completion-facing comparison supplied by T reaches its handoff only after reading the regular-family row, schedule, cofinal-window ledger, dyadic tolerance row, completion handoff, transports, continuations, provenance, and local naming row.

Next exhaust the regular-family presentation. Theorem 406.15 makes every regular-family consumer of T factor through the same rows, and Definition 406.19 records that a uniform-criterion presentation contributes only the shared threshold and real-seal handoff already displayed by T.

Now pass to the diagonal seal. Apply Definition 450.20 and Definition 450.21. The seal row S is read through the explicit modulus, diagonal selector, dyadic ledger, stream windows, regular readback, $\mathsf{Real}^{\uparrow}$ seal, transported routes, continuations, package provenance, and local $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ row.

Finally use Definition 450.17. Since neither the CauchySequenceSpace packet nor the RealCompleteness packet contains quotient streams, selected host limits, host real equality, or an ambient completeness row, a consumer cannot bypass the two-stage displayed route. □

Theorem 450.23 (RealCompleteness diagonal seal totality)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of the diagonal seal row S factors through exactly the displayed regular family row X, explicit modulus row μ, diagonal selector Δ,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q,* $\mathsf{Real}^{\uparrow}$ *seal row S,* hsame *transport row H,* Cont *route row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *row N. The consumer surface contains no quotient stream, host equality on completed reals, selected external limit, choice principle, or ambient completeness row.*

Proof. First identify the seal route. Apply Definition 450.13 and Definition 450.15. The candidate read is the seal row S, formed from the diagonal selector Δ, bounded by μ, and read through the dyadic ledger, stream windows, and regular-sequence readback rows D, W, Q.

Next lock the completion-facing endpoint. By Definition 450.16, a $\mathsf{Real}^{\uparrow}$ consumer sees the candidate only through the seal ledger carried by R and the displayed H, C, P, N rows.

For downstream seal consumers, use Definition 450.18 and Definition 450.20. They make the $\mathsf{CauchyLimitSeal}^{\uparrow}$ route consume the same explicit modulus, selected diagonal observations, dyadic ledger, stream windows, readback row, seal row, transports, continuations, package provenance, and local naming row.

Finally apply Definition 450.17. The excluded objects are absent from the accepted packet, so no consumer can replace the displayed diagonal-seal route with a quotient stream, host equality, selected external limit, choice principle, or ambient completeness row. □

Theorem 450.24 (RealCompleteness transport-tail-selector fusion)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Let*

$$T = (q, s, r, d, e, w, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet whose selected-tail rows provide the same precision,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows used by μ, Δ, D, W, Q, S. Let*

$$U = (R_U, W_U, D_U, S_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *packet transporting those rows to the same* R *surface, and let*

$$F = (q_F, s_F, r_F, w_F, e_F, h_F, c_F, p_F, n_F)$$

be an accepted $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *packet consuming the selected finite tail supplied by* T *and transported by* U*. Then every completion-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read of* R *factors through exactly the finite route*

$$\mu, \Delta, D, W, Q, S, H, C, P, N$$

together with the selected-tail witness row w*, the transport packet rows* $R_U, W_U, D_U, S_U, H_U, C_U, P_U, N_U$*, and the finite-tail seal rows* $q_F, s_F, r_F, w_F, e_F, h_F, c_F, p_F, n_F$*. The fused route exports no quotient stream, host real equality, selected external limit, countable-choice selector, or ambient completeness theorem.*

Proof. First expose the RealCompleteness side. Apply Definition 450.3 and Definition 450.4. They force each completion-facing precision request to read the displayed modulus, diagonal selector, dyadic ledger, stream windows, regular readback, seal row, transports, continuations, package provenance, and local naming row of R.

Next attach the selected tail. By Definition 458.3, the tail-selector packet supplies an admissible row w for the same precision, window, regularity, dyadic, and seal reads. The row is finite before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reaches the seal.

Now transport the selected row into the RealCompleteness surface. Apply Definition 600.9; since the transport packet presents the same selected-window, dyadic-ledger, and seal rows, every diagonal-seal consumer of R factors through the displayed transport rows before reading S.

Finally seal the finite tail. The handoff theorem Definition 559.5 routes the selected finite tail through $q_F, s_F, r_F, w_F, e_F, h_F, c_F, p_F, n_F$ before the real-facing row is consumed. Combine this handoff with Definition 450.23 and Definition 450.17; because none of the packets has a coordinate for quotient streams, host real equality, selected external limits, countable-choice selection, or ambient completeness, the fused consumer route cannot escape the displayed finite rows. □

Theorem 450.25 (RealCompleteness scoped consumer seal exhaustion)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. The diagonal-limit surface of* R *is scoped exactly by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *consumer rows, together with the* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the packet. It contains no quotient stream, selected host limit, host real equality, countable-choice row, or ambient completeness theorem.*

Proof. Start with the local naming surface. Apply Definition 450.2. Its carrier, classifier, diagonal readback, and package-closure clauses expose only the rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$.

Next bind the diagonal construction to finite observations. Apply Definition 450.15; the selected diagonal reads are exactly the reads justified by the explicit modulus, dyadic ledger, finite stream windows, and regular-sequence readback rows. Apply Definition 450.16 to route the resulting seal through the displayed $\mathsf{Real}^{\uparrow}$ row and the same transport, continuation, provenance, and naming rows.

For seal consumers, apply Definition 450.20. It keeps the $\mathsf{CauchyLimitSeal}^{\uparrow}$ route attached to the same diagonal seal and the same H, C, P, N ledger. For completion-facing consumers whose regular family is supplied through $\mathsf{CauchySequenceSpace}^{\uparrow}$, apply Definition 450.22; the handoff

contributes only its displayed finite window rows before returning to the same $\mathsf{RealCompleteness}^{\uparrow}$ packet.

Finally apply Definition 450.23. The diagonal-seal route is exhausted by the displayed rows, so no consumer can replace it by a quotient stream, selected host limit, host real equality, countable-choice row, or ambient completeness theorem. □

Theorem 450.26 (RealCompleteness scoped consumer seal exhaustion)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion-facing consumer of R factors through exactly the displayed finite seal surface*

$$X, \mu, \Delta, D, W, Q, S, H, C, P, N.$$

When the regular-family row is supplied through a $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff, the consumer route also factors through the displayed handoff rows of that packet before returning to the same* $\mathsf{RealCompleteness}^{\uparrow}$ *rows. No such consumer exports a quotient stream, host equality on completed reals, selected external limit, choice principle, or ambient completeness row.*

Proof. Start with the finite carrier surface. The local obligations in Definition 450.2 identify the accepted packet as the displayed rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$, and Definition 450.15 pins every diagonal observation used by the seal to μ, Δ, D, W, Q.

Next bind the seal consumers. Apply Definition 450.20 and Definition 450.21. These results make both the $\mathsf{Real}^{\uparrow}$ classifier and the downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ route consume the same seal row S through the displayed transports, continuations, package provenance, and local naming row.

For a completion-facing route whose family row is presented by $\mathsf{CauchySequenceSpace}^{\uparrow}$, use Definition 450.22. The handoff contributes only its displayed finite window rows before the read returns to $\mu, \Delta, D, W, Q, S, H, C, P, N$.

Finally, Definition 450.23 exhausts the diagonal-seal surface for all $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers. Apply Definition 450.17 to exclude quotient streams, host equality, selected external limits, choice principles, and ambient completeness from every completion-facing export. □

Theorem 450.27 (Real completeness finite limit-existence witness)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. For every precision request made by a* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion-facing consumer, the packet supplies a BEDC finite witness for the diagonal limit-existence obligation: the request reads the modulus row μ, selects a displayed diagonal observation through Δ, exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows D, W, Q, reaches the* $\mathsf{Real}^{\uparrow}$ *seal row S, and records the transport, continuation, package, and local naming rows H, C, P, N. The witness contains no quotient stream, selected host limit, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier limit existence finite witness`

Proof. Project the accepted packet through Definition 450.1. The rows named in the statement are exactly the rows of the carrier, so the precision request has no source outside the displayed BHist packet.

Use Definition 450.9 to bind each consumed tolerance request to μ. Then apply Definition 450.15 to route the selected diagonal observation through Δ, D, W, Q.

It remains to identify the endpoint and public ledger. Apply Definition 450.23. The selected route reaches S through H, C, P, N, and Definition 450.17 rules out quotient streams, selected host limits, countable choice, and ambient completeness as witness coordinates. □

Theorem 450.28 (Real completeness limit uniqueness classifier)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

and

$$R' = (X', \mu', \Delta', D', W', Q', S', H, C, P, N)$$

be accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packets whose* X, μ, Δ, D, W, Q *rows are related by the displayed transports named in the shared* H *row, and whose consumer routes use the same* C, P, N *data. Then the two packets export the same finite* $\mathsf{Real}^{\uparrow}$ *seal-ledger classifier for the candidate limit. The classifier is determined by the transported diagonal rows and the shared* H, C, P, N *ledger, not by quotient equality of streams, host real equality, a selected external limit, or ambient completeness.*

Proof. Begin with classifier stability in Definition 450.2. It transports the family, modulus, selector, dyadic ledger, stream windows, readback row, and seal row componentwise through the displayed H row.

Next use Definition 450.10. Since the transported selector reads use the same C, P, N route, they reach the same finite $\mathsf{Real}^{\uparrow}$ seal ledger.

The public classifier is then fixed by Definition 450.21. Finally apply Definition 450.17; no external completion object or quotient stream is present to refine or split the displayed seal-ledger classification. □

Theorem 450.29 (Real completeness LimitSelector handoff)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$L = (R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L)$$

be an accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet. Assume that the selected precision* n*, modulus* μ_L*, selected index* k_L*, finite window* W_L*, dyadic readback* D_L*, and* $\mathsf{Real}^{\uparrow}$ *seal row* S *agree through the displayed transports with the corresponding diagonal rows of* R *along* $\mu, \Delta, D, W, Q, S, H, C, P, N$*. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of* R *whose route reads that selected diagonal observation factors through the* $\mathsf{LimitSelector}^{\uparrow}$ *finite witness route*

$$n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$$

before reaching the same seal row S*. The factorization contains no host-selected limit, quotient-stream equality, countable-choice selector, or ambient completeness witness.*

Proof. First project the selector packet. The local obligations of Definition 673.2 expose $R_L, n, \mu_L, k_L, W_L, D_L, S, H$ and Definition 673.3 makes every diagonal-limit read pass through the displayed selector, window, readback, seal, transport, continuation, provenance, and naming rows.

Next compare this selector route with the $\mathsf{RealCompleteness}^{\uparrow}$ packet. The agreement assumptions identify the selected precision and modulus with the diagonal rows of R, while Definition 450.15 pins the corresponding observation of R to μ, Δ, D, W, Q.

It remains to identify the endpoint. Apply Definition 450.23 to route every $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of the selected observation to the same seal row S, and apply Definition 450.28 to keep the transported seal-ledger classifier fixed by the displayed rows.

Finally use selector exhaustion and seal non-escape. By Definition 673.7 and Definition 673.5, the selected observation has no remaining route outside $n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$. The boundary theorem Definition 450.17 excludes host-selected limits, quotient streams, countable choice, and ambient completeness from the $\mathsf{RealCompleteness}^{\uparrow}$ side as well. □

Theorem 450.30 (Real completeness limit handoff package)**.** *For every accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

the concrete paper-side package corresponding to the vision-level limit target consists of the finite limit-existence witness of Definition 450.27, the uniqueness classifier of Definition 450.28, the $\mathsf{LimitSelector}^{\uparrow}$ *finite-route handoff of Definition 450.29, the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *shared-window handoff of Definition 450.22, the diagonal seal totality of Definition 450.23, and the scoped consumer seal exhaustion of Definition 450.26. This package is a finite* BHist *handoff to* $\mathsf{Real}^{\uparrow}$ *and downstream seal consumers; it is not a theorem asserting* $\mathsf{Real}^{\uparrow}$ *completeness as a closed public result.*

Proof. First assemble the existence side. Apply Definition 450.27; it supplies the displayed precision-to-seal route through $\mu, \Delta, D, W, Q, S, H, C, P, N$.

Next assemble the classifier side. Apply Definition 450.28; transported packets with the same public ledger export the same finite $\mathsf{Real}^{\uparrow}$ seal-ledger classifier.

Now attach the selector handoff. Apply Definition 450.29; every selected diagonal observation consumed by a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ route factors through the independent $\mathsf{LimitSelector}^{\uparrow}$ finite witness route before it reaches S.

Then attach the downstream handoff. Use Definition 450.22 when the regular-family row is supplied through $\mathsf{CauchySequenceSpace}^{\uparrow}$, and use Definition 450.23 to keep every seal consumer on the same displayed route.

Finally apply Definition 450.26. The assembled package is exactly the scoped finite consumer surface. Since the chapter boundary excludes quotient streams, selected host limits, host equality, countable choice, and ambient completeness, the package does not close $\mathsf{Real}^{\uparrow}$ completeness as a public theorem. □

Theorem 450.31 (Real completeness shared-tail modulus limit lock)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$T = (R_0, R_1, M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1, E, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet. Assume that* M_0 *and* M_1 *present the same finite precision request read by* μ*, that the shared threshold* τ *presents the diagonal tail threshold* Δ*, that* D_T *and* D *are the same dyadic ledger row, that* W_0 *and* W_1 *present the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of* W*, that* Q_0 *and* Q_1 *present the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and that the* $\mathsf{Real}^{\uparrow}$ *seal*

row E is transported to S by the displayed rows H_T, C_T, P_T, N_T and H, C, P, N. Then every $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion-facing consumer of R whose route reads this diagonal limit observation factors through the shared-tail-modulus route*

$$M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1, E, H_T, C_T, P_T, N_T$$

before it reaches the seal row S. The locked route exports no quotient stream, selected host limit, countable-choice selector, host real equality, or ambient completeness theorem.

Proof. Start with the finite handoff package. By Definition 450.30, every completion-facing read of the displayed diagonal limit observation is already confined to the finite BHist route through $\mu, \Delta, D, W, Q, S, H, C, P, N$ and its downstream seal consumers.

Next identify the shared threshold side. Apply Definition 983.3. The two tail schedules M_0 and M_1 are read before the single threshold τ, and both $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks are forced through τ, D_T. The agreement assumptions transport this route to the Real-Completeness rows μ, Δ, D, W, Q.

Now attach the Real seal. By Definition 983.4, every Real-facing read of T reaches E only after the shared route through $M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1$. The assumed transport from E to S therefore places the $\mathsf{RealCompleteness}^{\uparrow}$ seal row after the same shared-tail-modulus route.

It remains to rule out a second completion-facing path. The uniqueness classifier of Definition 450.28 fixes the seal-ledger classifier once the displayed rows $\mu, \Delta, D, W, Q, S, H, C, P, N$ have been identified, and Definition 450.17 removes quotient streams, selected host limits, countable-choice selectors, host real equality, and ambient completeness from the packet boundary. Hence any $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or completion-facing consumer must pass through the shared-tail-modulus route before reaching S. □

Theorem 450.32 (Real completeness finite-window open phase exit)**.** *For every accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N),$$

the completion-facing open-phase boundary factors through exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row D, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row S, and the structural rows H, C, P, N. This gives the paper-side finite-window witness for the Phase Real and Completion object chain without claiming* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of streams, selected host limits, countable choice, or* $\mathsf{bridgeChecked}$ *status.*

Proof. Begin with the dependency lattice of Definition 450.8. It identifies the completion-facing object chain through the dyadic ledger, stream-window, regular-sequence readback, and RealUp seal rows already present in R.

Next apply Definition 450.7. The obligation surface exposes exactly D, W, Q, S together with the structural rows H, C, P, N, so the open-phase witness is finite and packet-local.

Use Definition 450.11 to route every completion-facing consumer through those displayed rows. Then apply Definition 450.17; no quotient stream, selected host limit, countable-choice selector, ambient completeness proof, or bridge-checked object lies outside the finite-window packet. □

Theorem 450.33 (Real completeness exact-boundary handoff)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$B = (L, K, J, S_B, W_B, R_B, D_B, E_B, H_B, C_B, P_B, N_B)$$

be an accepted RealCompletionExactBoundaryUp *packet. Suppose that the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D_B *matches* D*, the* $\mathsf{StreamName}^{\uparrow}$ *row* W_B *matches* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R_B *matches* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *row* E_B *matches* S *through the displayed componentwise* hsame *transports, the displayed* Cont *replays, the* Pkg *provenance rows, and the local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ *support. Then every completion-facing consumer of* R *that asks for an exact-boundary witness factors through* B *after the finite-window open-phase route. The handoff supplies only the same-route exact-boundary witness, and it exports no quotient stream, selected limit, host equality, countable choice, ambient completeness, metric completeness, or bridge token.*

Proof. Start with Definition 450.32. It confines the open Phase Real and Completion exit of R to the finite route through D, W, Q, S, H, C, P, N and excludes quotient streams, selected host limits, countable choice, ambient completeness, and bridge-checked status from that route.

Next apply Definition 1120.6. The accepted boundary packet B exposes exactly the four completion faces $\mathsf{DyadicRatCore}^{\uparrow}(D_B)$, $\mathsf{StreamName}^{\uparrow}(W_B)$, $\mathsf{RegSeqRat}^{\uparrow}(R_B)$, and $\mathsf{Real}^{\uparrow}(E_B)$, together with its structural rows H_B, C_B, P_B, N_B.

The matching assumptions identify these four faces with the rows D, W, Q, S already used by the finite-window route. Because the comparison is carried only by displayed hsame transports, Cont replays, Pkg provenance, and local NameCert support on both packets, the exact-boundary witness is a repacking of the same route rather than a second completion object.

Finally use Definition 1120.20 and Definition 1120.14. The first theorem keeps the witness inside the exact-boundary obligation surface, and the second theorem removes quotient streams, selected limits, host equality, countable choice, ambient completeness, metric completeness, and bridge tokens from the exported boundary. Hence every exact-boundary request made by a completion-facing consumer of R factors through B after the finite-window open-phase route and receives no additional completion witness. □

Theorem 450.34 (Real completeness exact-boundary finite induction)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$B = (L, K, J, S_B, W_B, R_B, D_B, E_B, H_B, C_B, P_B, N_B)$$

be an accepted RealCompletionExactBoundaryUp *packet whose four exact-boundary faces match the four open Real and Completion faces of* R*. Then induction on the finite consumer route of* R *reduces every exact-boundary request to exactly the matched faces* $\mathsf{DyadicRatCore}^{\uparrow}(D_B)$*,* $\mathsf{StreamName}^{\uparrow}(W_B)$*,* $\mathsf{RegSeqRat}^{\uparrow}(R_B)$*, and* $\mathsf{Real}^{\uparrow}(E_B)$*, followed only by the displayed* hsame *transports,* Cont *replays,* Pkg *provenance rows, and local* NameCert *rows of the two packets.*

Consequently the exact-boundary handoff is a finite BEDC-internal derivation over the four Phase Real and Completion faces. It has no quotient-stream, selected-limit, host-equality, countable-choice, ambient-completeness, metric-completeness, or bridge-token case.

Proof. Begin with Definition 450.32. The open-phase consumer route of R is finite, and its only completion-facing object rows are D, W, Q, S together with the structural rows H, C, P, N.

The base reads of the induction are therefore the four displayed face reads. Apply Definition 1120.6 to identify the matching boundary witnesses as $\mathsf{DyadicRatCore}^{\uparrow}(D_B)$, $\mathsf{StreamName}^{\uparrow}(W_B)$, $\mathsf{RegSeqRat}^{\uparrow}(R_B)$, and $\mathsf{Real}^{\uparrow}(E_B)$. Each base read is transported by the corresponding hsame comparison, replayed through the relevant Cont row, and recorded by the Pkg and local NameCert entries.

For the induction step, append one accepted consumer edge to a route already reduced to the four boundary faces. The edge can only read one of the structural rows H, C, P, N or one of the matched faces, because Definition 450.32 has no other open-phase exit. The handoff theorem Definition 450.33 then repacks the extended route through B without creating a second completion object.

Finally apply Definition 1120.14. The escape theorem removes every non-finite alternative case: there is no quotient-stream branch, no selected-limit branch, no host-equality branch, no countable-choice branch, no ambient-completeness branch, no metric-completeness branch, and no bridge-token branch. Thus the finite induction closes exactly on the four matched faces followed by H, C, P, N and the displayed local certificate rows. □

Theorem 450.35 (Real completeness exact-boundary standard bridge obligation)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. The packet R is eligible for a standard bridge only through a matching accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S_B, W_B, R_B, D_B, E_B, H_B, C_B, P_B, N_B)$$

whose four exposed faces match the four completion-facing rows of R: D_B matches D, W_B matches W, R_B matches Q, and E_B matches S through the displayed hsame *transports,* Cont *replays,* Pkg *provenance rows, and local* NameCert *support. The accepted bridge face must be the bridge-checked target*

BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryUp_StdBridge.

Thus any standard-bridge claim for $\mathsf{RealCompleteness}^{\uparrow}$ *factors through the already bridge-checked exact-boundary face, not through a separate completion object attached to R.*

Proof. First apply Definition 450.33. Any exact-boundary request made by a completion-facing consumer of R is already routed through a matching accepted RealCompletionExactBoundaryUp packet B after the finite-window open-phase route.

Next use Definition 450.34. The route reduces to the four matched faces $\mathsf{DyadicRatCore}^{\uparrow}(D_B)$, $\mathsf{StreamName}^{\uparrow}(W_B)$, $\mathsf{RegSeqRat}^{\uparrow}(R_B)$, and $\mathsf{Real}^{\uparrow}(E_B)$, followed only by displayed hsame transports, Cont replays, Pkg provenance rows, and local NameCert rows.

Finally invoke Definition 1120.21. That result identifies the public bridge factorization of the exact-boundary packet with the bridge-checked RealCompletionExactBoundaryUp standard-bridge target. Therefore the only admissible standard-bridge path for R is through the matched exact-boundary packet B and its bridge-checked face.

This obligation adds no quotient stream, selected limit, host equality, countable choice, ambient completeness, metric completeness, or bridge token to the $\mathsf{RealCompleteness}^{\uparrow}$ carrier. It records only the required factorization route for any later $\mathsf{RealCompleteness}^{\uparrow}$ standard-bridge claim. □

Theorem 450.36 (Real completeness selector route coverage)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$L = (R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L)$$

be an accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet whose selected observation is the displayed diagonal observation consumed from* R*. Every selected diagonal observation consumed by a* $\mathsf{Real}^{\uparrow}$ *reader or a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *reader factors through the finite* $\mathsf{LimitSelector}^{\uparrow}$ *route*

$$n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$$

and through the $\mathsf{RealCompleteness}^{\uparrow}$ *rows*

$$\mu, \Delta, D, W, Q, S, H, C, P, N.$$

The coverage is a finite route statement: it names the reader path that reaches the common seal row S*, and it does not export a host-selected limit, quotient-stream equality, countable-choice selector, host equality, or ambient completeness theorem.*

Proof. First expose the selector side. Apply Definition 673.2 to read the packet rows of L, and use Definition 673.3 to force every selected diagonal-limit read through $n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$.

Next compare the selected observation with the $\mathsf{RealCompleteness}^{\uparrow}$ diagonal rows. The handoff theorem Definition 450.29 identifies the selector route with the diagonal read of R, while Definition 450.15 pins that read to μ, Δ, D, W, Q.

Finally bind the consumer endpoint. Apply Definition 450.28 to keep the transported seal-ledger classifier fixed, and apply Definition 450.17 to rule out the listed external completion objects. □

Theorem 450.37 (Real completeness modulus-selector factorization)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. The modulus row and the selector row are not independently readable by any selected-observation consumer. Every selected index is justified first by the displayed modulus row* μ*, then by the selector row* Δ*, and then by the finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows* D, W, Q, S*. The remaining transport, continuation, package, and local naming rows* H, C, P, N *record the same route for downstream seal consumers.*

Proof. Begin with the modulus obligation. The RealCompleteness obligations bind every precision request to the displayed modulus row, and Definition 450.15 turns the selected diagonal read into a route through μ, Δ, D, W, Q.

The selector cannot be detached from that modulus route. When the observation is presented through $\mathsf{LimitSelector}^{\uparrow}$, Definition 673.2 and Definition 673.3 expose only the finite selector rows, and Definition 450.29 identifies those rows with the same μ, Δ, D, W, Q diagonal read.

It remains to seal the endpoint. Apply Definition 450.28 and Definition 450.17. The first fixes the seal-ledger classifier along S, H, C, P, N, and the second excludes an independent quotient, host, choice, or ambient-completeness interpretation of the selected index. □

Theorem 450.38 (Real completeness seal consumer exhaustion)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet, and let*

$$L = (R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L)$$

be an accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet reading a selected diagonal observation from* R*. Every downstream consumer of the shared seal exhausts the same finite route: it consumes the selector rows* $n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$ *when the observation is selector-presented, consumes the RealCompleteness rows* $\mu, \Delta, D, W, Q, S, H, C, P, N$ *when the observation is read from* R*, and exports no quotient stream, selected host limit, countable-choice selector, host equality, or ambient completeness theorem.*

Proof. First route the selector-presented observation. Theorems 673.2 and 673.3 expose the selector rows and give the finite witness route to the common seal row S.

Now route the same observation from the $\mathsf{RealCompleteness}^{\uparrow}$ packet. Use Definition 450.15 for the diagonal rows and Definition 450.29 for the comparison between the selector packet and R.

Finally exhaust the downstream consumer surface. Apply Definition 450.28 to keep the seal-ledger classifier determined by the displayed rows, and apply Definition 450.17 to exclude quotient streams, host-selected limits, countable-choice selectors, host equality, and ambient completeness from the export surface. □

Theorem 450.39 (RealCompleteness diagonal seal factorization public boundary)**.** *Every accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit packet consumed by a* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing reader factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *row. The public read exports no quotient stream, no selected host limit, no host real equality, no choice datum, and no ambient completeness theorem.*

Proof. Project the accepted $\mathsf{RealCompleteness}^{\uparrow}$ carrier. Its diagonal-limit rows expose the dyadic ledger, stream windows, regular-sequence readback, Cauchy-limit seal boundary, transports, routes, provenance, and local naming row as the only public coordinates.

Apply the diagonal witness schedule and finite witness rows already displayed in the chapter. They route every precision request through the modulus and diagonal selector before a $\mathsf{Real}^{\uparrow}$- or $\mathsf{CauchyLimitSeal}^{\uparrow}$-facing reader can see the observation.

Now route the seal-facing read. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary and $\mathsf{RegSeqRat}^{\uparrow}$ readback carry the observation back to the displayed finite ledger, while the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows record the same transport and provenance for the public consumer.

The non-escape conclusion is immediate from the carrier inventory. Since no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness theorem is a coordinate of the packet, none can be exported by the public factorization route. □

Theorem 450.40 (Real completeness finite approximation LimitSelector handoff)**.** *Let*

$$F = (R_F, n_F, k_F, W_F, D_F, E_F, S_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet, let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit packet consuming the finite approximation row, and let*

$$L = (R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L)$$

be a compatible accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet reading the same selected diagonal observation. If the finite approximation route of* F *supplies the source row, selected precision, threshold, finite*

window, dyadic observation ledger, explicit error row, and handoff row consumed by R, and if the selected observation of L agrees through the displayed transports with the same diagonal rows of R, then every $\mathsf{Real}^{\uparrow}$*-,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-, selector-facing, and completion-facing consumer factors through the same displayed rows*

$$X, \mu, \Delta, D, W, Q, S, H, C, P, N$$

and the selector rows

$$n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L.$$

The handoff contains the finite approximation route of F, the diagonal seal route of R, and the selected finite witness route of L only. It exports no quotient stream, no selected host limit, no countable-choice selector, no host equality, no hidden completion coordinate, and no ambient completeness theorem.

Proof. Begin with the finite approximation packet. Theorems 601.7 and 601.11 force every consumer of F to read the displayed source, precision, threshold, finite window, dyadic ledger, error row, handoff row, transports, continuation routes, provenance, and local naming rows before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary is visible.

Next attach the diagonal-limit packet. The finite approximation handoff is consumed by the $\mathsf{RealCompleteness}^{\uparrow}$ rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$, and the diagonal seal totality theorem routes the completion-facing read through the same seal row S rather than through an independent limit object.

Now attach the selector packet. Apply Definition 450.29 and Definition 450.36: the selected observation reaches S only through $n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$, and these rows are identified with the same diagonal read of R through the displayed transports.

Finally exhaust the consumer surface. Theorems 450.26 and 450.39 leave no remaining route outside the finite approximation, diagonal, and selector rows already displayed. Hence quotient streams, selected host limits, countable-choice selectors, host equality, hidden completion coordinates, and ambient completeness are not exported by the handoff. □

Theorem 450.41 (RealCompleteness diagonal example consumer lattice)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N$ be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Suppose a* $\mathsf{LimitSelector}^{\uparrow}$ *packet reads a selected diagonal observation from R, and suppose, when available, a* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *presentation supplies the same observation through a shared finite Cauchy window. Then the accepted data generate a finite consumer lattice over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{LimitSelector}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows: every completion-facing, seal-facing, selector-facing, and shared-window read factors through the displayed diagonal schedule, witness windows, readback rows, seal row, classifier transports, continuation routes, package provenance, and local naming rows.*

The lattice exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, no quotient stream, no selected host limit, no countable-choice selector, no host equality, and no ambient completeness row.*

Proof. First route the selected diagonal observation. Theorems 450.29 and 450.23 bind the $\mathsf{LimitSelector}^{\uparrow}$ read and the $\mathsf{RealCompleteness}^{\uparrow}$ diagonal seal to the same finite rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$.

When the observation is also presented through a shared finite Cauchy window, apply Definition 450.22. This places the $\mathsf{CauchySequenceSpace}^{\uparrow}$ presentation on the same window and seal route rather than adding an independent completion surface.

Now exhaust the downstream consumer routes. Theorem 450.38 says that selector-facing and RealCompleteness-facing consumers use exactly the displayed finite rows, while Definition 450.39 identifies the public boundary seen by $\mathsf{Real}^{\uparrow}$-, $\mathsf{CauchyLimitSeal}^{\uparrow}$-, and completion-facing readers.

The stated non-export claims follow from that factorization. None of a $\mathsf{Real}^{\uparrow}$ completeness theorem, quotient stream, selected host limit, countable-choice selector, host equality, or ambient completeness row is a coordinate of the factorized lattice, so none is exported by the consumer package. □

Theorem 450.42 (Real completeness shared-tail modulus handoff)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$ be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *carrier. Suppose a completion-facing consumer presents the selected diagonal observation through an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet. Then the consumer route factors first through the shared threshold row τ, then through the dyadic ledger, both* $\mathsf{StreamName}^{\uparrow}$ *windows, both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and finally through the same* $\mathsf{Real}^{\uparrow}$ *seal row S displayed by R.*

The route contains no pair of independent thresholds, no quotient stream equality, no host real equality, no selected host limit, no countable-choice selector, and no ambient completeness theorem.

Proof. Begin with the diagonal-limit side. Theorems 450.39 and 450.41 force every completion-facing read of R through $X, \mu, \Delta, D, W, Q, S, H, C, P, N$ before the $\mathsf{Real}^{\uparrow}$ seal row can be consumed.

Now attach the shared-tail packet. Apply Definition 983.3: both regular-Cauchy presentations are synchronized by the single threshold row τ before either window is read. Apply Definition 983.4: the synchronized tails then pass through the dyadic ledger, the two $\mathsf{StreamName}^{\uparrow}$ windows, and the two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks before reaching the real-facing seal row.

It remains to compare endpoints. The RealCompleteness factorization identifies that seal-facing endpoint with the displayed row S of R. Finally, Definition 450.17 excludes independent thresholds, quotient stream equality, host real equality, selected limits, countable choice, and ambient completeness from the exported route. □

Theorem 450.43 (Real completeness selector-window-seal factorization)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N$ be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Every selected diagonal observation consumed from R factors in the strict order*

$$\mu, \Delta, D, W, Q, S.$$

The modulus row μ supplies the finite request, the selector row Δ selects the displayed diagonal observation, the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D records the dyadic value, the* $\mathsf{StreamName}^{\uparrow}$ *windows W expose the finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q returns the regular-sequence row, and the* $\mathsf{Real}^{\uparrow}$ *seal S is reachable only after those rows have been displayed. The rows H, C, P, N carry transport, route replay, provenance, and local naming for this same factorization and carry no independent selector, window, limit, quotient, choice, host-equality, or ambient-completeness object.*

Proof. Start with selector route coverage. Theorem 450.36 says that every selected diagonal observation reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ reader only through the displayed selector route and the $\mathsf{RealCompleteness}^{\uparrow}$ rows $\mu, \Delta, D, W, Q, S, H, C, P, N$.

Next factor the front of the route. The modulus-selector factorization theorem forces the selected index to pass first through μ, then through Δ, and then through the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows D, W, Q, S.

Now seal the consumer boundary. The seal consumer exhaustion theorem leaves no downstream route outside the selector-presented rows and the displayed $\mathsf{RealCompleteness}^{\uparrow}$ rows, while the nonescape boundary excludes quotient streams, host-selected limits, countable-choice selectors, host equality, and ambient completeness.

Thus H, C, P, N can only transport, replay, authenticate, and name the same ordered route. They do not supply another selected observation or another seal endpoint. □

Theorem 450.44 (Real completeness selector-seal four-face uniqueness)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$ and $R' = (X', \mu', \Delta', D', W', Q', S', H', C', P', N')$ be accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packets. Suppose their selector rows, dyadic ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer rows agree through the displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$ *support. Then every selector-window seal route read from either packet factors through the same four visible faces: selector source, dyadic-window ledger, regular-sequence readback, and seal-facing consumer.*

Consequently the two routes determine the same public endpoint. No quotient stream, host-selected limit, countable-choice selector, host real equality, hidden completion coordinate, or ambient completeness row can distinguish the two selector-window seal routes.

Proof. First expose the route on each side. Theorem 450.36 forces every selected observation through the displayed selector route and the $\mathsf{RealCompleteness}^{\uparrow}$ rows, while Definition 450.37 fixes the front face by making the modulus row μ and selector row Δ the only source of the selected observation.

Next identify the middle faces. The finite-observation boundary for $\mathsf{StreamName}^{\uparrow}$, the common-tail window bridge for $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius-window handoff make the dyadic ledger, stream windows, and regular-sequence readback a single finite observation surface under the stated hsame, Cont, Pkg, and NameCert support.

Now compare seal-facing consumers. Theorem 450.38 exhausts all downstream seal reads, and Theorem 450.43 identifies the ordered route $\mu, \Delta, D, W, Q, S, H, C, P, N$ as the only selector-window-seal factorization. The CauchyLimitSeal selector nonescape and route-lock theorems then forbid a second seal-facing path outside these rows.

The four visible faces therefore determine the route endpoint. Since none of a quotient stream, host-selected limit, countable-choice selector, host real equality, hidden completion coordinate, or ambient completeness row occurs as a coordinate of the route, no such object can distinguish the two accepted presentations. □

Theorem 450.45 (Real completeness selector-window source determinacy)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P,$ be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Every accepted selector-window-seal read of R reads the modulus row μ before it reads the selector row Δ, and it reads Δ before any of the dyadic, stream-window, regular-sequence, or real-seal rows D, W, Q, S. The transport, route, package, and local naming rows H, C, P, N can replay and certify that order, but they cannot supply an alternate source for the selected observation.*

Proof. Start with selector route coverage. Theorem 450.36 makes every selected diagonal observation pass through the displayed selector route before it can be read by a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer.

Now apply the modulus-selector factorization. Theorem 450.37 fixes the front of that route: the finite request is supplied by μ, the selected diagonal row is supplied by Δ, and the middle certificate rows D, W, Q, S are reached only after that source pair is displayed.

Finally compare with the selector-window-seal factorization. Theorem 450.43 identifies H, C, P, N as transport, replay, provenance, and naming rows for the same ordered packet. Hence no accepted read can swap the source order or introduce a second source row. □

Theorem 450.46 (Real completeness selector-window dyadic-window nonescape)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H$ be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet. In every selector-window-seal read of R, the middle rows D, W, Q expose only the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$

readback attached to the source rows μ, Δ. They cannot export quotient streams, host equality, choice selectors, selected host limits, or ambient completeness.

Proof. Use Definition 450.45 first. It places D, W, Q downstream of the displayed source rows μ, Δ, so the middle rows are read as certificate rows for that selected observation rather than as independent completion data.

Next apply the nonescape boundary. Theorem 450.17 excludes quotient streams, host-selected limits, countable-choice selectors, host equality, and ambient completeness from the public $\mathsf{RealCompleteness}^{\uparrow}$ export.

The selector-window-seal factorization then pins the same exclusion to the middle rows. Since D, W, Q are the dyadic ledger, finite stream-window display, and regular-sequence readback inside the ordered route, they have no coordinate from which any of the excluded objects could be exported. □

Theorem 450.47 (Real completeness selector-window consumer readback)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P,$ be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ packet. Every $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, selector-facing, and completion-facing consumer of the selector-window-seal surface reads back the same displayed route*

$$\mu, \Delta, D, W, Q, S, H, C, P, N.$$

No such consumer obtains a second modulus row, selector row, dyadic ledger, stream window, regular-sequence readback, seal row, transport, route replay, package provenance, or local naming row.

Proof. First exhaust the seal-facing consumers. Theorem 450.38 says that $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, selector-facing, and completion-facing reads use the selector-presented rows and the displayed $\mathsf{RealCompleteness}^{\uparrow}$ rows.

Next attach the shared-tail handoff. Theorem 450.42 identifies the completion-facing route through the shared threshold, dyadic ledger, stream windows, regular-sequence readbacks, and the same real-facing seal row S.

Finally apply terminal export. Theorem 450.48 exports exactly the selector-window-seal packet, and Definition 450.46 removes the possible middle-row escape routes. Thus every listed consumer reads back the same displayed $\mu, \Delta, D, W, Q, S, H, C, P, N$ route. □

Theorem 450.48 (Real completeness selector-window terminal export)**.** *Let $R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$ be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ packet. Once the ordered route*

$$\mu, \Delta, D, W, Q, S, H, C, P, N$$

is displayed, every terminal completion-facing export of R factors through the same selector-window-seal packet. The export cannot introduce quotient streams, host-selected limits, countable choice, host equality, hidden completion coordinates, or ambient completeness beyond the rows already displayed by the selector-window-seal factorization.

Proof. Start with selector route coverage. Theorem 450.36 routes selected observations through the displayed $\mathsf{RealCompleteness}^{\uparrow}$ rows before a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ reader can consume the seal endpoint.

Next use the modulus-selector factorization. Theorem 450.37 forces the front of the terminal route through μ and Δ, then through the dyadic, stream-window, regular-sequence, and real-seal rows D, W, Q, S.

Now exhaust the seal-facing consumer. Theorem 450.38 leaves no downstream route outside the selector-presented rows, and Definition 450.43 identifies the ordered packet used by the terminal reader.

Finally apply Definition 450.17. It excludes quotient streams, host-selected limits, countable-choice selectors, host equality, hidden completion coordinates, and ambient completeness, so the terminal export is exactly the displayed selector-window-seal packet. □

Theorem 450.49 (Real completeness tail-fiber selector compatibility)**.** *Let*

$$R = (X, \mu, \Delta, D, W, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RealCompleteness}^{\uparrow}$ *packet. Let*

$$L = (R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L)$$

be a compatible accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet reading the selected diagonal observation of* R*. Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet whose paired windows and dyadic ledgers present that same selected observation to* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*. Then every completion-facing consumer of this combined presentation factors through one finite route: on the* $\mathsf{RealCompleteness}^{\uparrow}$ *side it reads*

$$\mu, \Delta, D, W, Q, S, H, C, P, N,$$

on the $\mathsf{LimitSelector}^{\uparrow}$ *side it reads*

$$R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L,$$

and on the tail-fiber side it reads

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H_F, C_F, P_F, N_F.$$

The compatibility exports no quotient stream, host real equality, countable-choice selector, selected host limit, or ambient completeness theorem.

Proof. First route the selected diagonal observation through the $\mathsf{LimitSelector}^{\uparrow}$ packet. Apply Definition 673.2 and Definition 673.3 to expose $R_L, n, \mu_L, k_L, W_L, D_L, S, H_L, C_L, P_L, N_L$, then use Definition 450.29 to identify that selector read with the displayed diagonal rows of R.

Next route the same observation from the $\mathsf{RealCompleteness}^{\uparrow}$ side. Apply Definition 450.36, Definition 450.37, and Definition 450.43 to force the read through $\mu, \Delta, D, W, Q, S, H, C, P, N$.

Use the terminal export theorem. By Definition 450.48, a completion-facing consumer cannot leave the displayed selector-window-seal packet after that route has been exposed.

Now compare the shared-tail presentation. Theorem 893.4 makes the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ route and the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ route share the row T before the seal row A is read, while Definition 893.8 fixes their common support to $R_0, R_1, W_0, W_1, D_0, D_1, T$.

Finally exhaust all consumer surfaces. Apply Definition 893.9 to keep every tail-fiber consumer inside $R_0, R_1, W_0, W_1, D_0, D_1, T, A, H_F, C_F, P_F, N_F$, then apply Definition 450.17 and Definition 893.5. The two non-escape boundaries exclude quotient streams, host real equality, countable-choice selectors, selected host limits, and ambient completeness, so the combined presentation has exactly three displayed finite routes: $\mathsf{RealCompleteness}^{\uparrow}$, $\mathsf{LimitSelector}^{\uparrow}$, and $\mathsf{RegularCauchyTailFiber}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompleteness}^{\uparrow}$ packet is generated by a displayed regular Cauchy family, explicit modulus row, diagonal selector, dyadic observation ledger, stream windows, regular-sequence readback, RealUp seal, transport routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 450.30 binds the public finite handoff package for every $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and completion-facing consumer route over the displayed rows $X, \mu, \Delta, D, W, Q, S, H, C, P, N$, with $\mathsf{CauchySequenceSpace}^{\uparrow}$ handoff rows only when present.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompleteness}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealCompletenessUp.RealCompletenessBHistCarrier\_regseq\_seal\_handoff`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, a public standard bridge, or bridge checking.

Upgrade path: The remaining upgrade is a bridge-checked standard bridge for the displayed RealCompleteness handoff target.

Chapter 451

A Concrete Naming Certificate for $\mathsf{FinitePhaseSampling}^{\uparrow}$

The $\mathsf{FinitePhaseSampling}^{\uparrow}$ packet is a BEDC-native finite certificate surface. It records only finite phase rows, finite coefficient reads, and explicit aliasing ledgers, so Real and Completion consumers may cite coefficient recovery without importing Haar measure, root-of-unity analysis, quotient streams, or host complex Fourier theory.

It is a downstream consumer of the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window readback in chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal surface in chapter 38. Its coefficient row is useful only through that Real-and-completion route, and it does not introduce an independent analytic carrier.

Definition 451.1 (Finite phase sampling carrier)**.** *A* $\mathsf{FinitePhaseSampling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, K, M, A, E, H, C, P, N)$$

with the following visible coordinates:

1. *G is the finite phase-grid row, listing the sampled phase marks and the cyclic grid schedule used by the observation packet;*
2. *K is the mode-bound row, recording the finite integer modes whose coefficient reads are admitted and the displayed barrier that keeps the requested modes below the grid scale;*
3. *M is the finite coefficient-sample row, storing the observed values and the grid sums used to read cosine, sine, or character coefficients;*
4. *A is the aliasing ledger, listing exactly which translated mode rows would collide with each requested mode at the chosen grid scale;*
5. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal row through which downstream real-name consumers read the recovered coefficient data;*
6. *H stores the componentwise* hsame *transports for G, K, M, A, E;*
7. *C stores the displayed* Cont *routes by which consumers may inspect the finite rows and the real handoff;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the local finite observation rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePhaseSampling}^{\uparrow}}$ *provenance row.*

The classifier compares two carriers only by these displayed finite rows and their stored transports. It does not compare functions on a circle, choose a Haar integral, form a quotient stream, or import a host Fourier object.

Theorem 451.2 (Finite phase sampling NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FinitePhaseSampling}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the packet of Definition 451.1;*
2. *classifier laws by componentwise* hsame *on the grid row, mode-bound row, coefficient-sample row, aliasing ledger, real handoff row, consumer routes, package provenance, and local certificate row;*
3. *grid exactness: every coefficient sample in M is obtained from the displayed phase marks of G and an admitted mode of K by a finite sum in the same packet;*
4. *handoff exactness: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row and the* $\mathsf{Real}^{\uparrow}$ *seal in E read only the finite coefficient rows accepted by M together with the aliasing ledger A;*
5. *non-escape: accepted carriers export no Haar-measure object, no root-of-unity theorem, no quotient-stream equality, and no host complex Fourier theory.*

Lean4 ✓ `BEDC.Derived.FinitePhaseSamplingUp.FinitePhaseSamplingNameCertObligations`

Proof. Carrier admission is the displayed tuple $S = (G, K, M, A, E, H, C, P, N)$. Each coordinate is a finite BHist row or a route between finite rows.

For the classifier laws, compare the coordinates through H and then read the permitted continuations through C. Reflexivity, symmetry, and transitivity are inherited coordinate by coordinate because the local certificate has no hidden coordinate beyond the displayed packet.

Grid exactness is finite ledger inspection. A coefficient row is accepted only when its mode is present in K and its summation entries are the phase observations already listed in G and M.

The handoff obligation repacks those finite coefficient rows as the $\mathsf{RegSeqRat}^{\uparrow}$ source for E, and the $\mathsf{Real}^{\uparrow}$ seal reads through that source. The aliasing ledger A remains attached to the same request, so the handoff does not introduce an external limiting argument.

Non-escape follows from the carrier definition. Since the packet contains no Haar measure, no analytic root-of-unity theorem, no quotient stream, and no host Fourier space, no consumer route can export one. □

Theorem 451.3 (Finite phase sampling grid recovery)**.** *Let $S = (G, K, M, A, E, H, C, P, N)$ be an accepted* $\mathsf{FinitePhaseSampling}^{\uparrow}$ *carrier. If a requested coefficient mode k is admitted by K below the displayed grid barrier and the aliasing ledger A has no nonzero colliding translated mode for k, then the coefficient row read from M is determined by the finite grid row G alone. Two accepted carriers with matching G, K, A rows therefore transport the same recovered coefficient row through their stored* hsame *and* Cont *routes.*

Proof. Project the grid row G, the mode-bound row K, the sample row M, and the aliasing ledger A from the carrier. The mode k is accepted only through the finite admission row in K.

The no-collision entry of A states that the requested row has no displayed nonzero translated mode at the chosen grid scale. Thus the coefficient read in M is the finite grid sum over the phase marks in G for that single admitted mode.

When two carriers match on G, K, A, they have the same admitted mode, the same displayed phase rows, and the same absence entry in the aliasing ledger. The transports in H and consumer routes in C carry this finite read to the matching coefficient row, so no Haar averaging or external character orthogonality is used. □

Theorem 451.4 (Finite phase sampling real handoff)**.** *For every accepted* $\mathsf{FinitePhaseSampling}^{\uparrow}$ *carrier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the finite coefficient-sample row* M *and the aliasing ledger* A*. Any Real and Completion consumer of* E *can inspect the recovered coefficient only through* G, K, M, A, H, C, P, N*, and receives no selected real limit, quotient-stream equality, Haar integral, or host complex coefficient.*

Proof. Start with an accepted read of E. By the handoff exactness obligation in Definition 451.2, the regular rational source row is assembled from the coefficient samples in M together with the aliasing information in A.

The $\mathsf{Real}^{\uparrow}$ seal then reads that $\mathsf{RegSeqRat}^{\uparrow}$ source through the same stored transports H and continuations C. The package row P and local certificate row N identify the finite observation source but do not add a further analytic object.

Therefore every consumer of E factors through the finite rows G, K, M, A, H, C, P, N. The carrier has no coordinate from which a selected limit, quotient equality, Haar integral, or host complex coefficient could be exported. □

Theorem 451.5 (Finite phase sampling RegSeqRat Real lattice)**.** *Let* $S = (G, K, M, A, E, H, C, P, N)$ *be an accepted* $\mathsf{FinitePhaseSampling}^{\uparrow}$ *carrier. If a Real or Completion consumer reads the handoff row* E*, then that read first factors through the regular-rational source surface of chapter 301. The coefficient observations are read as dyadic ledger rows from chapter 291, and only after those finite rows have been exposed does the route reach the real-name seal of chapter 38.*

Proof. Project the handoff row E from the accepted carrier. The handoff exactness obligation of Definition 451.2 says that E reads only the coefficient samples in M and the aliasing ledger A.

By Definition 451.4, every Real or Completion consumer of E is routed through exactly the displayed rows G, K, M, A, H, C, P, N before it reaches the downstream seal. Thus the regular-rational source surface is the first consumer-facing source, and the coefficient observations are visible only as the finite dyadic ledger rows attached to that surface.

The real-name seal is therefore reached only after the RegSeqRat route and its dyadic coefficient ledger have been read. Since the carrier exports no selected real limit, quotient-stream equality, Haar integration datum, or host complex coefficient, no consumer can bypass this lattice through an analytic object outside the packet. □

Theorem 451.6 (Finite phase sampling ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{FinitePhaseSampling}^{\uparrow}$ *packet factors through the finite phase-grid row, mode-bound row, coefficient-sample row, aliasing ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport row, continuation row, package provenance, and local* $\mathsf{NameCert}$ *row. The packet exports no ambient circle, no Haar measure, no root-of-unity analysis, no quotient stream, no infinite Fourier series, and no external theorem.*

Proof. Take an exported consumer route and project the carrier of Definition 451.1. The available coordinates are exactly $G, K, M, A, E, H, C, P, N$.

The NameCert obligations account for carrier admission, classifier transport, grid exactness, real handoff exactness, and non-escape. Each obligation is stated over those displayed rows and is preserved by the stored hsame and Cont routes.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient circle object, Haar measure, root-of-unity proof, quotient stream, infinite Fourier series, or imported external theorem, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePhaseSampling}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePhaseSampling}^{\uparrow}$ packet is generated from a finite phase-grid row, a finite mode-bound row, finite coefficient samples, an aliasing ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, Cont routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePhaseSampling}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite carrier, local NameCert obligations, grid recovery, real handoff, and ledger non-escape for finite phase sampling packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePhaseSampling}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Haar integration, root-of-unity analysis, quotient streams, infinite Fourier series, host complex Fourier theory, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for grid recovery, real handoff, and non-escape.

Chapter 452

A Concrete Naming Certificate for $\mathsf{FiniteLebesgueNumber}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the finite Lebesgue-number route used here is a displayed dyadic finite-cover modulus. The $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ packet records a finite dyadic cover, a positive rational radius ledger, and the finite-window witnesses needed by total-bounded and compact-metric consumers. It is not an open-cover compactness principle and it does not choose a subcover from an arbitrary family.

The finite radius ledger is consumed by the compact-metric route of chapter 129 and is built over the explicit dyadic-cover surface of chapter 446.

Definition 452.1 (Finite Lebesgue-number carrier)**.** *A* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier over a displayed* $\mathsf{DyadicCover}^{\uparrow}$ *row is a finite* BHist *packet*

$$L = (D, W, \lambda, M, H, C, P, N)$$

with the following visible coordinates:

1. *D is the accepted* $\mathsf{DyadicCover}^{\uparrow}$ *row containing the finite list of dyadic centers, rational ball rows, dyadic mesh cells, and finite coverage witnesses;*
2. *W is the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row whose inspected points are covered by the dyadic cells of D;*
3. *λ is the positive finite radius ledger, read through* $\mathsf{RationalBall}^{\uparrow}$ *and bounded by the radii recorded in D;*
4. *M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh ledger that records which mesh cells refine into the λ-balls and into the cover cells of D;*
5. *H records componentwise* hsame *transports for D, W, λ, M, the* $\mathsf{StreamName}^{\uparrow}$ *reads, the* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation rows, and the* $\mathsf{Real}^{\uparrow}$ *consumer rows;*
6. *C records the displayed* Cont *routes by which consumers inspect finite-window membership, rational-ball containment, dyadic-mesh refinement, stream-name reads, regular-rational approximations, and real rows;*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local certificate rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}$ *row.*

The carrier accepts only packets whose positive finite radius ledger is displayed beside the finite dyadic cover and whose mesh witnesses keep every admitted finite-window read inside a named cover cell.

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier`

Theorem 452.2 (Finite Lebesgue-number NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier: the packet of Definition 452.1;*
2. *classifier: componentwise comparison of the dyadic cover row, finite-window row, positive radius ledger, mesh ledger, transports, continuations, package provenance, and local certificate row;*
3. *radius-modulus stability: displayed* hsame *and* Cont *transport preserves the positive ledger* λ *together with the* $\mathsf{RationalBall}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$ *rows that witness its refinement into the finite cover;*
4. *finite-window coverage exactness: every accepted point read from* W *factors through a* $\mathsf{StreamName}^{\uparrow}$ *row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation row, a* $\mathsf{Real}^{\uparrow}$ *row, a mesh cell from* M*, and a named cover cell in* D*;*
5. *non-escape: accepted packets export no open-cover compactness theorem, no arbitrary subcover selector, no host compactness principle, and no hidden cover family outside the displayed dyadic finite-cover data.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier namecert obligations`

Proof. Carrier habitation is the displayed tuple $L = (D, W, \lambda, M, H, C, P, N)$. The dyadic cover, finite-window row, radius ledger, mesh ledger, transports, continuations, package provenance, and local certificate are all finite coordinates.

The classifier is componentwise on those coordinates. Radius-modulus stability follows by transporting the positive rational radius ledger and the mesh-refinement rows along H, then replaying only the routes stored in C.

Coverage exactness is a finite witness read. Each admitted point of W is observed through its stream and regular-rational approximation rows, placed in a real row, assigned to a mesh cell of M, and then routed to a named cover cell in D.

The carrier has no coordinate for an arbitrary open family, an open-cover compactness theorem, or a subcover selector. Thus every accepted export remains inside the displayed dyadic finite-cover packet. □

Theorem 452.3 (Finite Lebesgue-number dyadic cover handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives only the displayed* $\mathsf{DyadicCover}^{\uparrow}$ *finite cover, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *coverage row, the positive finite radius ledger* λ*, the* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement ledger* M*, and the listed* hsame, Cont, Pkg, *and* NameCert *provenance rows. The handoff supplies a finite dyadic-cover modulus and a positive finite radius ledger, not open-cover compactness or subcover choice.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberDyadicCoverHandoff`

Proof. Project the carrier of Definition 452.1. The only consumer-facing cover data are D, W, λ, M, H, C, P, and N.

Use Definition 452.2. Its classifier, radius-modulus stability, and finite-window coverage exactness rows keep every consumer read on the displayed dyadic cover, finite-window witnesses, rational-ball radius ledger, and mesh-refinement ledger.

A total-bounded or compact-metric consumer therefore receives the finite dyadic cover together with the positive radius ledger and provenance rows. Since the carrier contains no arbitrary open family and no selected subcover coordinate, the handoff cannot assert open-cover compactness. □

Theorem 452.4 (Finite Lebesgue-number radius transport)**.** *Let $L = (D, W, \lambda, M, H, C, P, N)$ be an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier. For every componentwise* hsame *transport recorded in H, the transported packet preserves the positive finite radius ledger λ, the* $\mathsf{RationalBall}^{\uparrow}$ *radius readback attached to λ, and the* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement rows in M as one finite packet. In particular, a consumer cannot transport the radius coordinate without simultaneously transporting the rational-ball and mesh-refinement evidence by which the radius is read.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusTransport`

Proof. Project the accepted carrier of Definition 452.1. The radius coordinate is not stored as an isolated scalar: it is bundled with the rational-ball readback, the mesh ledger, the componentwise transport rows, and the displayed continuation routes.

Use the radius-modulus stability row of Definition 452.2. That row transports λ only together with the $\mathsf{RationalBall}^{\uparrow}$ and $\mathsf{DyadicMesh}^{\uparrow}$ rows witnessing refinement into the finite cover.

Therefore every transported radius read remains in the finite packet $D, W, \lambda, M, H, C, P, N$. No additional radius selector, host metric radius, or open-cover radius coordinate is exposed. □

Theorem 452.5 (Finite Lebesgue-number finite-window coverage exactness)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier $L = (D, W, \lambda, M, H, C, P, N)$, every accepted finite-window read factors through the displayed window W, then through a* $\mathsf{StreamName}^{\uparrow}$ *row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation row, a* $\mathsf{Real}^{\uparrow}$ *row, a mesh cell of M, and a named cover cell of D. The factorization has no coordinate for an arbitrary open family, no host compactness witness, and no selected subcover outside the displayed dyadic finite-cover packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberWindowCoverageExactness`

Proof. Start with an accepted read from W. By the finite-window coverage exactness row of Definition 452.2, the read is observed through a named stream row, its regular-rational approximation row, and the corresponding real row.

The same exactness row then routes the real read into a displayed mesh cell of M and from that mesh cell into a named cover cell of D. These are coordinates of the carrier, so the factorization is finite and internal to L.

The non-escape row of Definition 452.2 excludes open-family, compactness, and selected-subcover coordinates. Hence every accepted finite-window read is exhausted by the displayed route. □

Theorem 452.6 (Finite Lebesgue-number compact consumer route)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives the carrier route in the order $D, W, \lambda, M, H, C, P, N$: the dyadic cover row, finite-window row, positive radius ledger, mesh refinement ledger, transport rows, continuation routes, package provenance, and local naming row. The consumer cannot read open-cover compactness, host compactness, or a selected subcover from the packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformModulusConsumerExactness`

Proof. Use Definition 452.3. It identifies the public consumer-facing data as $D, W, \lambda, M, H, C, P, N$ and states that the handoff supplies a finite dyadic-cover modulus and positive finite radius ledger.

Next use Definition 452.4. It keeps the positive radius read tied to the rational-ball and mesh-refinement evidence, so a compact consumer cannot detach the radius from the displayed packet.

Finally use Definition 452.5. Every finite-window read is routed through W, stream and regular-rational rows, a real row, a mesh cell of M, and a named cover cell of D.

The non-escape clauses in these three rows leave no coordinate for open-cover compactness, host compactness, or selected subcover data. Thus the compact consumer route is exactly the ordered carrier route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier compact consumer route`

Theorem 452.7 (Finite radius nonchoice obligation)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the positive radius ledger* λ *is readable only through the displayed* $\mathsf{RationalBall}^{\uparrow}$ *radius rows of* D *and the* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement rows of* M*. No consumer route in the packet reads* λ *by invoking an open-cover choice principle, an arbitrary subcover selector, or a host compactness witness outside* $D, W, \lambda, M, H, C, P, N$*.*

Proof. Start with the carrier of Definition 452.1. Its radius coordinate is listed between the finite-window row W and the mesh ledger M, and its admissible readback is explicitly through the rational-ball and dyadic-mesh rows.

Apply the radius-modulus stability row of Definition 452.2 and the transport statement of Definition 452.4. They keep λ attached to the $\mathsf{RationalBall}^{\uparrow}$ rows and to the $\mathsf{DyadicMesh}^{\uparrow}$ refinement evidence under every stored hsame transport and every replayed Cont route.

The compact consumer route of Definition 452.6 exposes the ordered packet $D, W, \lambda, M, H, C, P, N$ and no other cover coordinate. Since the packet contains no open-cover family and no subcover selector, the radius obligation is finite and nonchoice. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberFiniteRadiusNonchoice`

Theorem 452.8 (Mesh refinement nonchoice obligation)**.** *Let* $L = (D, W, \lambda, M, H, C, P, N)$ *be an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier. Refinement from a mesh cell recorded in* M *to a cover cell recorded in* D *is stable under the stored* hsame *transports in* H *and under the replayed* Cont *routes in* C*. Thus the mesh-to-cover handoff is determined by the displayed* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, not by a separately chosen open subcover.*

Proof. Use the finite-window coverage exactness theorem Definition 452.5. An accepted read is first routed through W, then through a $\mathsf{StreamName}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ approximation row, a $\mathsf{Real}^{\uparrow}$ row, a mesh cell of M, and finally a named cover cell of D.

The NameCert obligations of Definition 452.2 make this route componentwise. The same rows transport the mesh ledger and the cover-cell witness along H, while the continuations in C replay the finite-window membership, rational-ball containment, and mesh-refinement reads.

It follows that changing representatives by hsame does not detach the mesh cell from its named cover cell. The handoff remains inside $D, W, \lambda, M, H, C, P, N$, so no open-cover compactness theorem or arbitrary subcover coordinate can supply a different refinement route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier mesh refinement nonchoice`

Theorem 452.9 (Total-bounded handoff obligation)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives exactly the ordered packet*

$$D, W, \lambda, M, H, C, P, N.$$

The consumer may inspect the displayed DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows in that packet, but cannot recover open-cover compactness, host compactness, arbitrary subcover selection, or a hidden cover family from it.*

Proof. The carrier definition fixes the visible order $D, W, \lambda, M, H, C, P, N$. The compact consumer route of Definition 452.6 states that this is the consumer-facing order for both TotallyBounded$^{\uparrow}$ and CompactMetric$^{\uparrow}$ reads.

The finite radius obligation of Definition 452.7 keeps the radius ledger tied to the displayed rational-ball and mesh rows. The mesh refinement obligation of Definition 452.8 keeps the mesh-to-cover read stable under stored transports and replayed continuations.

Together with the dyadic cover handoff of Definition 452.3, these rows exhaust the public data a total-bounded or compact-metric consumer receives. The non-escape row of Definition 452.2 excludes open-cover compactness and arbitrary subcover selection, so the handoff is exactly the finite BEDC packet. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier total bounded handoff`

Theorem 452.10 (Compact-uniform selector handoff)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier consumed by a compact-uniform route supplies exactly the positive radius ledger read by* CompactNetModulusSelector$^{\uparrow}$, *and then by* UniformModulus$^{\uparrow}$, *through the displayed* DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows. It supplies no open-cover compactness, host minimum, selected subcover, quotient equality, or external modulus function.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformSelectorHandoff`

Proof. Project the finite Lebesgue-number carrier $L = (D, W, \lambda, M, H, C, P, N)$. The positive ledger λ is displayed between the finite-window row and the mesh row, so any compact-uniform consumer reads it as part of the same finite packet.

Invoke the dyadic cover handoff, radius transport, and finite-window coverage exactness theorems. These route the radius through D, W, the RationalBall$^{\uparrow}$ readback, the DyadicMesh$^{\uparrow}$ refinements, and the stored hsame and Cont rows.

Apply the total-bounded handoff theorem above. It identifies $D, W, \lambda, M, H, C, P, N$ as the exact packet delivered to total-bounded and compact-metric consumers, so the compact-uniform route receives no hidden compactness witness in addition to the finite radius ledger.

The compact-net selector and uniform-modulus chapters are the downstream consumers of this displayed packet. The selector reads the positive radius ledger as its compact-net input, and the uniform-modulus surface reads the resulting tolerance-only route. Since open-cover compactness, host minima, selected subcovers, quotient equality, and external modulus functions are absent from L, the handoff is exactly the finite BEDC radius-ledger route. □

Theorem 452.11 (Window-radius ledger coherence)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$, *every finite-window read in* W *is coherent with the positive radius ledger* λ *only through the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows. A compact consumer cannot read a finite window first and then attach an unrelated positive radius outside* $D, W, \lambda, M, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberWindowRadiusLedgerCoherence`

Proof. Project the displayed carrier. The finite-window row W and positive radius ledger λ are coordinates of the same packet, and the local NameCert obligations name them together.

Use radius transport and finite-window coverage exactness. The radius row is preserved by the stored hsame transport and replayed Cont route, while the finite-window row is read through the dyadic cover and mesh rows already present in D and M.

Apply the compact-uniform selector handoff above. The downstream compact route receives the ordered finite packet, so any radius attached after the window read must be the displayed ledger λ rather than an external compactness choice. □

Theorem 452.12 (Cover-cell readback totality)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, each read from the finite-window row* W *reaches a named dyadic cover cell through* $D, W, \lambda, M, H, C, P, N$*. The readback first consumes the window row, then the positive radius ledger and mesh row, then the componentwise transport, replay, provenance, and local naming rows; no open-cover family or arbitrary cell selector supplies the endpoint.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCoverCellReadbackTotality`

Proof. Start from finite-window coverage exactness. It sends every accepted window read through the displayed stream, regular-rational, real, mesh, and cover rows.

The mesh refinement and radius transport obligations keep that endpoint tied to M and λ under the same H and C routes. Thus the cover cell is not selected by a separate open-cover principle.

Finally, P and N package and name the readback as part of the same FiniteLebesgueNumber$^{\uparrow}$ carrier. This gives total cover-cell readback for the finite dyadic route. □

Theorem 452.13 (L10 source-lock for compact radius reads)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *public finite-window read locks its radius route through* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*, and* DyadicCover$^{\uparrow}$ *before any* CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$ *consumer may read the positive radius ledger. The consumer-facing read is therefore a read of the displayed packet* $D, W, \lambda, M, H, C, P, N$*, not a radius read supplied by an external compactness or modulus interface.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberL10SourceLock`

Proof. Begin with finite-window coverage exactness. It forces each accepted finite-window read to pass through the displayed stream, regular-rational, real, mesh, and cover rows before it can be used as a cover read.

The radius transport and mesh-refinement nonchoice theorems keep the positive ledger λ tied to that same window and mesh route. Hence a compact consumer cannot detach the radius from W and M and then attach a different compactness witness.

Finally apply the compact-uniform selector handoff and cover-cell readback totality. They identify the downstream compact and uniform readers as consumers of the named finite packet $D, W, \lambda, M, H, C, P, N$, so the L10 source route is locked before any positive radius ledger is exposed. □

Theorem 452.14 (Stream-regular window order)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, each public finite-window read is ordered as follows: first the window row* W*, then the* StreamName$^{\uparrow}$ *row, then the* RegSeqRat$^{\uparrow}$ *approximation row, then the* Real$^{\uparrow}$ *row, then a mesh cell in* M*, then a cover cell in* D*, and finally the support rows* H, C, P, N*. Componentwise* hsame *transport and the displayed* Cont *replay preserve that order.*

Proof. Use finite-window coverage exactness to read the window row through the displayed stream, regular-rational, real, mesh, and cover rows. This fixes the forward route from W to the named cell of D through M.

Use radius transport and mesh-refinement nonchoice to preserve the same order under componentwise hsame transport. The positive ledger λ remains between the finite window and mesh rows, so the mesh cell in M cannot be read before the regular and real source rows have been displayed.

The compact-uniform selector handoff and cover-cell readback totality then package the route with H, C, P, N. The Cont replay is therefore an order-preserving replay of the already displayed finite packet, not a new path that bypasses the stream or regular-Cauchy source. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberStreamRegularWindowOrder`

Theorem 452.15 (Real source nonchoice boundary)**.** *For every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *row used by a compact or uniform consumer is read only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows. It is not read by host real equality, an arbitrary point selector, open-cover compactness, a subcover choice principle, or any external real representative outside* $D, W, \lambda, M, H, C, P, N$.

Proof. Start from finite-window coverage exactness. The $\mathsf{Real}^{\uparrow}$ row appears as part of the displayed source chain following the stream-name and regular-rational rows, so the real read is already anchored before the mesh and cover cells are consumed.

Radius transport and mesh-refinement nonchoice keep that real source attached to the positive radius and mesh rows under the same hsame and Cont data. Thus host real equality and arbitrary point selection cannot replace the displayed source rows.

The compact-uniform selector handoff and cover-cell readback totality give the downstream boundary. Compact and uniform consumers receive the named finite route, while open-cover compactness, subcover choice, and external representatives are absent from the carrier. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealSourceNonchoice`

Theorem 452.16 (Compact consumer nonchoice boundary)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or* CompactNetModulusSelecto *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives only the finite dyadic route through* $D, W, \lambda, M, H, C, P, N$. *The consumer may use the named cover cells, finite-window reads, positive radius ledger, mesh refinements, transports, continuations, provenance, and local naming, but it receives no open-cover compactness theorem, host compactness witness, arbitrary subcover choice, or external selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactConsumerNonchoiceBoundary`

Proof. Use the compact-uniform selector handoff and the total-bounded handoff. Together they identify the consumer-facing data as the displayed finite carrier $D, W, \lambda, M, H, C, P, N$.

The window-radius coherence theorem keeps window reads and radius reads in the same packet. The cover-cell readback totality theorem then gives named dyadic cover cells without invoking a free subcover selector.

The non-escape row of the local NameCert obligations excludes open-cover compactness, host compactness, and arbitrary selectors. Therefore compact consumers receive only the finite dyadic route. □

Theorem 452.17 (Finite Lebesgue-number positive-radius compact-net export)**.** *For every accepted* FiniteLebesgueNumber$^\uparrow$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the positive-radius ledger exported to* CompactNetModulusSelector$^\uparrow$ *is exactly the finite route through* DyadicCover$^\uparrow$*,* FiniteWindowEnvelope$^\uparrow$*,* RationalBall$^\uparrow$*,* DyadicMesh$^\uparrow$*,* StreamName$^\uparrow$*,* RegSeqRat$^\uparrow$*,* Real$^\uparrow$*,* TotallyBounded$^\uparrow$*,* CompactMetric$^\uparrow$*, and* CompactNetModulusSelector$^\uparrow$ *displayed by* $D, W, \lambda, M, H, C, P, N$*. The export has no open-cover compactness theorem, host minimum, arbitrary subcover selector, quotient equality, or choice coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPositiveRadiusCompactNetExport`

Proof. Project the carrier. The dyadic cover row D, finite-window row W, positive radius ledger λ, mesh row M, and support rows H, C, P, N are the only rows available before a compact-net consumer can read the packet.

Use the compact consumer nonchoice boundary. It keeps CompactMetric$^\uparrow$ and CompactNetModulusSelector$^\uparrow$ consumers inside the finite dyadic route and excludes open-cover compactness, host compactness, arbitrary subcover choice, and external selectors.

The positive finite-radius totality theorem places λ before the compact read, while the total-bounded handoff routes the finite cover data through TotallyBounded$^\uparrow$ and CompactMetric$^\uparrow$. Hence the compact-net export is the displayed finite radius ledger and nothing stronger. □

Theorem 452.18 (Finite Lebesgue-number dyadic-cover uniform-modulus handoff)**.** *For every accepted* FiniteLebesgueNumber$^\uparrow$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, every* UniformModulus$^\uparrow$ *consumer receives the dyadic-cover radius route only after the same displayed packet has passed through* DyadicCover$^\uparrow$*,* FiniteWindowEnvelope$^\uparrow$*,* RationalBall$^\uparrow$*,* DyadicMesh$^\uparrow$*,* StreamName$^\uparrow$*,* RegSeqRat$^\uparrow$*,* Real$^\uparrow$*,* TotallyBounded$^\uparrow$*,* CompactMetric$^\uparrow$*, and* CompactNetModulusSelector$^\uparrow$*. The handoff is a finite radius-to-modulus route, not an open-cover compactness argument, selected subcover, host minimum, or external modulus selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberDyadicCoverUniformModulusHandoff`

Proof. Start from the compact-uniform selector handoff. It identifies the downstream compact-net selector and uniform-modulus chapters as consumers of the finite packet $D, W, \lambda, M, H, C, P, N$.

Next apply positive-radius compact-net export. The selector receives the positive-radius ledger only through the displayed dyadic cover, rational-ball, mesh, window, regular-readback, Real, total-bounded, and compact-metric rows.

Finally use the finite radius nonchoice and mesh-refinement nonchoice obligations. These keep the radius and mesh attached to the dyadic packet under H, C, P, N, so a UniformModulus$^\uparrow$ consumer cannot substitute an open-cover compactness theorem, host minimum, selected subcover, or external modulus row for the finite handoff. □

Theorem 452.19 (Finite Lebesgue-number compact-consumer radius-ledger totality)**.** *For every accepted* FiniteLebesgueNumber$^\uparrow$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, compact and uniform consumers that preserve the displayed classifier receive a total positive-radius ledger inside the same finite route* $D, W, \lambda, M, H, C, P, N$*. The ledger is total for the displayed* DyadicCover$^\uparrow$*,* FiniteWindowEnvelope$^\uparrow$*,* RationalBall$^\uparrow$*,* DyadicMesh$^\uparrow$*,* StreamName$^\uparrow$*,* RegSeqRat$^\uparrow$*,* Real$^\uparrow$*,* TotallyBounded$^\uparrow$*,* CompactMetric$^\uparrow$*,* CompactNetModulusSelector$^\uparrow$*, and* UniformModulus$^\uparrow$ *consumer chain, and it excludes open-cover compactness, host compactness, selected subcovers, hidden minima, quotient equality, and choice.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactConsumerRadiusLedgerTotality`

Proof. Begin with window-radius ledger coherence. It ties every finite-window read to the same positive-radius ledger λ through the displayed carrier rows, so the compact consumer cannot attach an unrelated radius after reading W.

Use cover-cell readback totality and dyadic-cover uniform-modulus handoff. The first theorem gives named dyadic cover cells through $D, W, \lambda, M, H, C, P, N$, and the second carries the same positive-radius route to $\mathsf{CompactNetModulusSelector}^{\uparrow}$ and $\mathsf{UniformModulus}^{\uparrow}$ consumers.

The compact consumer nonchoice boundary supplies the exclusion clause. Since open-cover compactness, host compactness, selected subcovers, hidden minima, quotient equality, and choice coordinates are absent from the packet, the compact-consumer radius ledger is total exactly on the displayed finite route. □

Theorem 452.20 (Positive finite-radius totality)**.** *For every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the packet exposes a positive finite radius ledger* λ *before any* $\mathsf{CompactMetric}^{\uparrow}$ *consumer may read a Lebesgue-number route. The radius is consumed only together with the displayed* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*, transport, continuation, provenance, and local naming rows.*

Proof. Project the carrier $L = (D, W, \lambda, M, H, C, P, N)$. The radius coordinate is not a host scalar outside the packet; it is placed between the finite-window row and the mesh row.

Use Definition 452.7 and Definition 452.24. They keep λ attached to rational-ball readback and dyadic-mesh refinement under stored transports and replayed continuations.

Finally apply Definition 452.9. A compact consumer receives the ordered finite packet only after the positive radius ledger has been displayed, so the Lebesgue-number route has finite-radius totality before it reaches $\mathsf{CompactMetric}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPositiveRadiusTotality`

Theorem 452.21 (Dyadic cover cell selection)**.** *For every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, each finite-window read in* W *lands in a displayed dyadic cover cell of* D *through the mesh row* M*. The packet supplies no arbitrary subcover selector and no open-cover compactness coordinate outside the displayed dyadic cover, finite-window, radius, mesh, transport, continuation, provenance, and local naming rows.*

Proof. Use Definition 452.5. It routes a finite-window read through stream naming, regular-rational readback, a Real row, a mesh cell of M, and then a named cover cell of D.

The mesh refinement nonchoice theorem, Definition 452.8, keeps that mesh-to-cover handoff stable under the stored hsame transports and displayed Cont routes.

Since the carrier contains no open-cover family and no subcover selector, cell selection is exactly the finite dyadic route already displayed by $D, W, \lambda, M, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberDyadicCoverCellSelection`

Theorem 452.22 (Compact-metric handoff exactness)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives only the dyadic cover row* D*, finite-window row* W*, positive-radius ledger* λ*, mesh row* M*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}$ *row* N*. It receives no open-cover compactness, host compactness, arbitrary subcover selector, or hidden cover family.*

Proof. Start from the total-bounded handoff theorem Definition 452.9. It identifies $D, W, \lambda, M, H, C, P, N$ as the exact packet delivered to total-bounded and compact-metric consumers.

The positive-radius totality theorem above exposes the finite radius ledger before the compact route is read. The dyadic cover cell-selection theorem above identifies every finite-window read with a displayed dyadic cover cell through M.

The non-escape row of Definition 452.2 excludes open-cover compactness, host compactness, and arbitrary subcover selection. Hence $\mathsf{CompactMetric}^{\uparrow}$ receives exactly the finite-radius dyadic packet and nothing outside it. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.CompactMetricHandoffExactness.FiniteLebesgueNumberCompactMe`

Theorem 452.23 (Finite-window coverage obligation row)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the finite-window coverage row factors through* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicMesh}^{\uparrow}$ *in that order. A point read from* W *is accepted only when its stream-name row, regular-rational row, real row, mesh cell, and named dyadic cover cell are all displayed in the same packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberObligationCoverageRow`

Proof. Start with the finite-window coordinate W in the carrier. The carrier definition already places W beside the dyadic cover D and the mesh ledger M, while the finite-window coverage exactness theorem routes each accepted read through stream, regular-rational, real, mesh, and cover rows.

The obligation row records that route as the BEDC consumer contract. Since every coordinate is read from $D, W, \lambda, M, H, C, P, N$, no open-cover family or host compactness witness supplies the factorization. □

Theorem 452.24 (Radius mesh stability obligation row)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, every stored* hsame *transport preserves the positive radius ledger* λ *only together with the displayed* $\mathsf{RationalBall}^{\uparrow}$ *rows and the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh-refinement rows. Thus radius transport is not a scalar transport separate from the rational-ball and mesh evidence that names it.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusMeshStabilityRow`

Proof. Project the radius coordinate from the carrier. It is located between the finite-window row and the mesh ledger, and its admissible readback is tied to the rational-ball rows of the dyadic cover.

Apply the radius transport and finite radius nonchoice obligations. They keep λ attached to rational-ball readback and mesh refinement under every stored hsame row and replayed Cont route, so a transported radius remains part of the same finite packet. □

Theorem 452.25 (Compact consumer boundary obligation row)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives only the displayed dyadic finite-cover packet* $D, W, \lambda, M, H, C, P, N$*. The consumer may inspect the finite cover, finite-window row, positive radius ledger, mesh-refinement row, transports, continuations, provenance, and local naming, but receives no arbitrary open-cover family, host compactness theorem, or selected subcover coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactConsumerBoundaryRow`

Proof. Use the total-bounded handoff obligation. It identifies $D, W, \lambda, M, H, C, P, N$ as the exact packet delivered to total-bounded and compact-metric consumers.

The coverage and radius-mesh rows above keep finite-window reads and radius reads inside that packet. The non-escape row of the local NameCert obligations excludes open-cover compactness and arbitrary subcover selection, so the compact consumer boundary is precisely the displayed dyadic finite-cover packet. □

Theorem 452.26 (Window-radius source lock)**.** *For every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the positive radius ledger* λ *is locked to the same source packet as the finite-window row* W*. The lock is witnessed only through the displayed* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by* $D, W, \lambda, M, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberWindowRadiusSourceLock`

Proof. Project the carrier $L = (D, W, \lambda, M, H, C, P, N)$. The window row W and radius ledger λ occur in the same finite BHist packet, with D and M recording the dyadic cover and mesh rows that read them.

Apply Definition 452.24. It keeps λ attached to the rational-ball rows and dyadic-mesh refinement rows under the stored hsame transports and replayed Cont routes.

Apply Definition 452.23. It routes every accepted finite-window read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, the mesh row M, and the cover row D. Therefore the radius source cannot be read apart from the same displayed source packet. □

Theorem 452.27 (Total-bounded consumer factorization)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$ *factors through the displayed dyadic finite-cover packet before reading any compact route. The factorization first reads* $\mathsf{DyadicCover}^{\uparrow}$ *and* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*, then reads* $\mathsf{RationalBall}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$*, then reads* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and finally replays only the carried* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTotalBoundedConsumerFactorization`

Proof. Start with Definition 452.25. It identifies $D, W, \lambda, M, H, C, P, N$ as the exact packet delivered to total-bounded and compact-metric consumers.

Use Definition 452.26. The radius ledger and finite-window reads share the same dyadic source packet, so the consumer cannot detach λ from W, D, or M.

Use Definition 452.9. The consumer-facing order is exhausted by the displayed carrier rows and their hsame, Cont, Pkg, and NameCert support, so the compact read factors through the finite dyadic packet. □

Theorem 452.28 (Open-cover refusal boundary)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, any request for an arbitrary open-cover family, a selected subcover, host compactness, or open-cover compactness is refused by the carrier boundary. The accepted request surface is only the finite route through* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Read the non-escape row of Definition 452.2. It excludes arbitrary open families, open-cover compactness, host compactness, and selected subcover coordinates from the carrier.

The compact consumer boundary theorem then identifies the only admitted consumer packet as $D, W, \lambda, M, H, C, P, N$. The total-bounded factorization theorem above keeps every compact route

inside the finite dyadic cover, window, radius, mesh, stream, regular-rational, real, transport, replay, provenance, and naming rows.

Thus an open-cover request asks for data absent from the carrier. It is refused as a boundary crossing rather than answered by a hidden compactness principle. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenCoverRefusalBoundary`

Definition 452.29 (Finite Lebesgue number root coverage ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier* $L = (D, W, \lambda, M, H, C, P, N)$*, the root coverage ledger is the finite* BHist *row*

$$\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$$

whose entries expose the DyadicCover$^{\uparrow}$ *row* D*,* DyadicMesh$^{\uparrow}$ *row* M*,* RationalBall$^{\uparrow}$ *radius row* R*,* StreamName$^{\uparrow}$ *window row* S*,* RegSeqRat$^{\uparrow}$ *source row* G*,* Real$^{\uparrow}$ *endpoint row* E*,* CompactMetric$^{\uparrow}$ *and* CompactNetModulusSelector$^{\uparrow}$ *consumer row* K*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. The ledger has no coordinate for arbitrary open-cover choice, selected subcover choice, or host compactness.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootCoverageLedger`

Theorem 452.30 (Finite Lebesgue number root dyadic radius obligation)**.** *Every accepted root coverage ledger of Definition 452.29 must display the rational radius row* R *as a* RationalBall$^{\uparrow}$ *row read through the same finite* DyadicCover$^{\uparrow}$ *and* DyadicMesh$^{\uparrow}$ *packet that supplies* D *and* M*. The* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows* S, G, E *may consume the radius only through the transported* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not by a hidden open-cover compactness witness.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootDyadicRadiusObligation`

Proof. Project the ledger $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$. The radius row R is listed between the dyadic cover and dyadic mesh rows, so its only source packet is the finite dyadic one.

Apply the radius-mesh stability row, Definition 452.24, and the window-radius source lock, Definition 452.26. Together they keep the rational-ball radius attached to D, M under componentwise hsame transport and displayed Cont replay.

The stream, regular-rational, and real rows then read the transported finite packet. Since Λ has no open-cover or selected-subcover coordinate, the radius obligation remains finite and nonchoice. □

Theorem 452.31 (Finite Lebesgue number root compact consumer nonchoice)**.** *Every* CompactMetric$^{\uparrow}$ *or* CompactNetModulusSelector$^{\uparrow}$ *consumer of an accepted root coverage ledger factors through the displayed* DyadicCover$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows before it may read* K*. The consumer uses only the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the ledger and cannot request an arbitrary open cover or selected subcover.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootCompactConsumerNonchoice`

Proof. Start from total-bounded consumer factorization, Definition 452.27. It routes every compact-metric consumer through the finite dyadic packet before any compact read is exposed.

Use the open-cover refusal boundary, Definition 452.28. It blocks arbitrary open covers, host compactness, and selected subcover coordinates, leaving only the finite cover, mesh, radius, stream, regular-rational, real, transport, replay, provenance, and naming rows.

Thus K is a consumer of the displayed finite ledger rather than a compactness selector outside it. The route is nonchoice because every consumed row is already present in Λ. □

Theorem 452.32 (Finite Lebesgue number root window cover totality)**.** *For every accepted root coverage ledger $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$, each window named by the* $\mathsf{StreamName}^{\uparrow}$ *row S and each regular-rational source named by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row G is covered by one of the finitely displayed* $\mathsf{RationalBall}^{\uparrow}$ *rows of R over the* $\mathsf{DyadicCover}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$ *rows D, M. The* $\mathsf{Real}^{\uparrow}$ *endpoint row E and compact-consumer row K may read totality only through that finite window-cover route.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootWindowCoverTotality`

Proof. Read finite-window coverage exactness through Definition 452.23. It sends every accepted finite-window read through the displayed dyadic cover, mesh, stream, regular-rational, and real rows.

The root dyadic radius obligation above attaches each rational ball to the same D, M source packet. Therefore the windows named by S and sources named by G are covered by listed rational-ball rows rather than by an external subcover.

Finally the compact consumer nonchoice theorem keeps E and K downstream of that same finite route. The totality claim is exactly the finite ledger totality, not an open-cover compactness assertion. □

Theorem 452.33 (Finite Lebesgue number root unblock package)**.** *The accepted root coverage ledger of Definition 452.29 forms a finite root-unblock package for* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$*: the* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *rows are all exposed inside one* BHist *packet, preserved by* hsame*, replayed by* Cont*, sourced by* Pkg*, and sealed by* $\mathsf{NameCert}$*. Consequently the later root consumer may target finite cover, radius, window totality, and compact-consumer obligations without relying on arbitrary open-cover compactness or choice.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootUnblockPackage`

Proof. Assemble the package from the root coverage ledger. The definition lists the dyadic cover, dyadic mesh, rational ball, stream, regular-rational, real endpoint, compact-metric, compact-net selector, transport, replay, provenance, and local naming rows in one finite BHist packet.

The dyadic radius obligation supplies the radius row over the same finite dyadic packet. The compact consumer nonchoice theorem keeps compact consumers inside the displayed finite rows. The window cover totality theorem then supplies totality for the StreamName and RegSeqRat windows through the listed rational-ball rows.

Repacking these three obligations with H, C, P, N gives the finite root-unblock package. No step introduces open-cover compactness, selected subcover choice, host compactness, or a consumer row outside the displayed ledger. □

Theorem 452.34 (Finite Lebesgue number selector-radius domain)**.** *The positive radius ledger consumed by* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *is readable only through the displayed* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root coverage ledger. It has no separate open-cover compactness, selected-subcover, host-minimum, or choice coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectorRadiusDomain`

Proof. Project the root coverage ledger $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$. The positive radius row is the rational-ball row R, and it is displayed with the dyadic cover and dyadic mesh rows D, M before any compact-net selector reads K.

Use the root dyadic radius obligation and window-radius source lock. Definition 452.30 keeps R over the same D, M packet, while Definition 452.26 ties finite-window reads to the same dyadic source.

Finally attach S, G, E, H, C, P, N. The StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows are the only rows that can carry the radius onward to CompactNetModulusSelecto so no hidden compactness or choice coordinate is available. □

Theorem 452.35 (Finite Lebesgue number compact-net selector factorization)**.** *Every* CompactNetModulusSelecto *consumer of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *root coverage ledger factors through the root coverage ledger before reading its selector row. The factorization route is*

$$D, M, R, S, G, E, H, C, P, N \to K,$$

so the selector receives finite cover, mesh, radius, stream, regular-rational, real endpoint, transport, replay, provenance, and local naming data before any compact-net selection occurs.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactNetSelectorFactorization`

Proof. Begin with the root unblock package, Definition 452.33. It exposes the compact-net selector row inside the same finite BHist packet as the dyadic cover, mesh, radius, stream, regular-rational, and real rows.

Next use root compact consumer nonchoice and total-bounded consumer factorization. Definition 452.31 and Definition 452.27 both route compact consumers through the finite dyadic packet before any compact read is exposed.

The selector-radius domain theorem supplies the remaining radius constraint. Since R is readable only through the displayed dyadic and transport rows, the selector row K factors through the root coverage ledger rather than an external subcover selector. □

Theorem 452.36 (Finite Lebesgue number uniform-modulus radius exhaustion)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *packet receives only the finite Lebesgue-number radius packet carried by the displayed root coverage ledger. It receives no open-cover compactness, no selected subcover, no host minimum, and no countable or finite choice coordinate beyond the finite rows already named by* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. First read the selector-radius domain theorem. It identifies the positive radius ledger as the rational-ball row over the finite dyadic cover and mesh packet, transported only by the displayed hsame, Cont, Pkg, and NameCert rows.

Apply compact-net selector factorization. Any compact-net route to a modulus selector must pass through $D, M, R, S, G, E, H, C, P, N$ before it reaches K, so the modulus-facing consumer receives a finite packet rather than an open-cover object.

Finally use the open-cover refusal boundary, Definition 452.28. It blocks arbitrary open families, host compactness, and selected-subcover coordinates, leaving UniformModulus$^{\uparrow}$ with only the finite Lebesgue-number radius packet. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformModulusRadiusExhaustion`

Definition 452.37 (Finite Lebesgue-number radius lower-bound fold)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *root coverage ledger*

$$\Lambda = (D, M, R, S, G, E, K, H, C, P, N),$$

the radius lower-bound fold is the finite BHist *packet*

$$F_\Lambda = (D, R, M, S, G, E, K, H, C, P, N)$$

whose read order first displays the DyadicCover$^\uparrow$ *row* D*, then the* RationalBall$^\uparrow$ *radius row* R*, then the* DyadicMesh$^\uparrow$ *row* M*, and then the* StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, CompactNetModulusSelector$^\uparrow$, hsame, Cont, Pkg, *and* NameCert *rows. The fold is a finite lower-bound packet for the displayed radius rows only: it has no coordinate for a host minimum, arbitrary open-cover compactness, selected subcover choice, quotient equality, or external modulus selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTasteGate single carrier alignment`

Theorem 452.38 (Finite Lebesgue-number cover-shrinkage stability)**.** *Let* $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$ *be an accepted* FiniteLebesgueNumber$^\uparrow$ *root coverage ledger. Shrinking each displayed cover cell along the folded positive radius packet* F_Λ *preserves the same* DyadicCover$^\uparrow$, RationalBall$^\uparrow$, DyadicMesh$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, hsame, Cont, Pkg, *and* NameCert *route. In particular, every shrunken-cell read remains a read through the folded finite lower-bound packet and not through an open-cover compactness theorem, selected subcover, host minimum, quotient equality, or choice coordinate.*

Proof. Project the root coverage ledger of Definition 452.29. Its rows already place $D, R, M, S, G, E, K, H, C, P, N$ in one finite BHist packet, and the lower-bound fold only changes the read order so the rational-ball radius is inspected before the mesh row.

Use the selector-radius domain theorem, Definition 452.34. It says the positive radius ledger is readable only through the displayed dyadic cover, rational-ball, dyadic mesh, stream, regular-rational, real, transport, replay, provenance, and naming rows.

Apply compact-net selector factorization and uniform-modulus radius exhaustion. Definition 452.35 keeps the selector downstream of the folded packet, and Definition 452.36 prevents a modulus consumer from replacing that radius with open-cover compactness or a selected subcover.

Thus cover-cell shrinkage is stable under the same displayed transports and replays. The shrinkage endpoint is the folded finite lower-bound packet, not a host minimum or external equality coordinate. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCoverShrinkageStability`

Theorem 452.39 (Finite Lebesgue-number UniformModulus consumer exactness)**.** *Every* UniformModulus$^\uparrow$ *consumer of an accepted* FiniteLebesgueNumber$^\uparrow$ *root coverage ledger factors through the radius lower-bound fold* F_Λ*. The consumer receives exactly the finite radius packet*

$$D, R, M, S, G, E, H, C, P, N \to K$$

carried by DyadicCover$^\uparrow$, RationalBall$^\uparrow$, DyadicMesh$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, hsame, Cont, Pkg, *and* NameCert *rows, and it receives no open-cover compactness theorem, selected subcover, host minimum, quotient equality, choice coordinate, or external modulus row.*

Proof. Begin with the root coverage ledger. By Definition 452.29, every candidate consumer row is already displayed inside $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$, and Definition 452.37 repacks those rows as the finite lower-bound packet F_Λ.

Apply Definition 452.34. It identifies R as the positive radius ledger over the displayed D, M dyadic packet and keeps all later reads inside S, G, E, H, C, P, N.

Next apply Definition 452.35 and Definition 452.36. The first theorem routes K through the finite packet before any selector read, and the second theorem states that a $\mathsf{UniformModulus}^{\uparrow}$ consumer receives only the finite Lebesgue-number radius packet.

Finally use cover-shrinkage stability. It shows that shrinking cover cells to the folded positive radius remains stable under the same displayed transport and replay rows, so the consumer receives exactly $D, R, M, S, G, E, H, C, P, N \to K$ and no open-cover compactness, selected-subcover, host-minimum, quotient-equality, or choice coordinate. □

Theorem 452.40 (Finite Lebesgue number radius-modulus transport scope)**.** *In every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root coverage ledger, the positive radius ledger is transported toward a modulus only together with the displayed* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows. Thus the radius-to-modulus route has no separate carrier for a compact minimum, an open-cover selector, or a choice coordinate outside the finite dyadic-cover packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusModulusTransportScope`

Proof. Project the accepted ledger $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$. The selector-radius domain theorem identifies R as the positive rational-ball radius row over the displayed dyadic cover and mesh rows D, M.

Apply the compact-net selector factorization theorem and the uniform-modulus radius exhaustion theorem. They force every modulus-facing consumer to pass through $D, M, R, S, G, E, H, C, P, N$ before any selector row can be read.

The remaining rows are exactly the StreamName, RegSeqRat, Real, transport, replay, provenance, and NameCert rows already named in the ledger. Since no other row is present, radius-modulus transport is scoped by the displayed finite packet. □

Theorem 452.41 (Finite Lebesgue number open-phase root package)**.** *Every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root coverage ledger presents its finite Lebesgue-number root surface as the finite-window route*

$$D, M, R, S, G, E, H, C, P, N$$

inside one BHist *packet. The route is anchored by* hsame, Cont, Pkg, *and* NameCert *rows, starts from the displayed* $\mathsf{DyadicCover}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$ *evidence, and transports the positive* $\mathsf{RationalBall}^{\uparrow}$ *radius through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers without exposing open-cover compactness, a selected subcover, a host minimum, quotient equality, or a choice coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier open phase root package`

Proof. Read the dyadic-cover handoff and the root coverage ledger. They display the cover, mesh, rational-radius, stream, regular-rational, real, transport, replay, provenance, and local naming rows in one finite BHist packet.

Apply the finite-window coverage exactness, root compact consumer nonchoice, and radius-modulus transport scope rows. Together they route every root-surface read through $D, M, R, S, G, E, H, C, P, N$ before any compact or uniform consumer can inspect the packet.

The open-cover refusal boundary supplies the negative side of the package. Since no arbitrary open family, selected subcover, host compactness minimum, quotient equality, or external selector row is present, the root surface is precisely the displayed finite-window route. □

Theorem 452.42 (Finite Lebesgue number compact cover-selection row)**.** *Every* CompactMetric$^{\uparrow}$ *cover-selection consumer of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *packet consumes the dyadic cover row D, total-bounded finite-window rows, dyadic mesh row M, rational radius row R, and the* hsame, Cont, Pkg, *and* NameCert *ledger before any compact read. Thus compact cover selection is a finite-route read from* DyadicCover$^{\uparrow}$, DyadicMesh$^{\uparrow}$, TotallyBounded$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *data, not a read from open-cover compactness or a choice-like selected subcover.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactCoverSelectionRow`

Proof. Start from the finite-window route of Definition 452.41. It places the dyadic cover, mesh, radius, stream, regular-rational, real, transport, replay, provenance, and naming rows before the compact consumer.

Use the total-bounded consumer factorization and compact-net selector factorization rows. These factor the compact-metric read through the displayed finite cover and mesh packet, so the compact route consumes the finite dyadic data already present in the ledger.

Finally apply the open-cover refusal boundary. The compact consumer therefore has no coordinate from which to read an arbitrary open cover, a selected subcover, or a compactness minimum; it receives only the finite cover-selection row anchored in BHist, hsame, Cont, Pkg, and NameCert. □

Theorem 452.43 (Finite Lebesgue number ContinuousMap uniform route row)**.** *Every* ContinuousMap$^{\uparrow}$ *or* UniformModulus$^{\uparrow}$ *consumer reading an accepted* FiniteLebesgueNumber$^{\uparrow}$ *packet may read only the explicit radius ledger and the compact-net route*

$$D, M, R, S, G, E, H, C, P, N \to K.$$

The consumer may replay Cont *and transport along* hsame *inside the stored* BHist *packet, and may cite the displayed* Pkg *and* NameCert *rows, but it cannot replace that route by an external continuity modulus, open-cover compactness, selected subcover, host minimum, quotient equality, or choice coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier continuousmap uniform route row`

Proof. Begin with radius-modulus transport scope. It already restricts the positive radius ledger to the displayed dyadic cover, mesh, stream, regular-rational, real, transport, replay, provenance, and naming rows.

Attach the compact-net selector factorization and UniformModulus consumer exactness rows. They place K downstream of $D, M, R, S, G, E, H, C, P, N$ and identify the modulus-facing packet as the radius lower-bound fold rather than an independent continuity object.

The replay row for ContinuousMap$^{\uparrow}$ is therefore a consumer of the stored finite route. It can read Cont only as part of the same BHist packet, so the uniform route remains the explicit radius ledger and compact-net route, with no outside open-cover or choice coordinate. □

Theorem 452.44 (Finite Lebesgue number compact uniform handoff scope)**.** *Every* CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *consumer of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *packet factors through the same displayed finite dyadic-cover packet before receiving compact or uniform-continuity data. The handoff route is the finite row family*

$$D, M, R, S, G, E, H, C, P, N \to K,$$

with D from DyadicCover$^{\uparrow}$, *M from* DyadicMesh$^{\uparrow}$, *R from* RationalBall$^{\uparrow}$, *S from* StreamName$^{\uparrow}$, *G from* RegSeqRat$^{\uparrow}$, *E from* Real$^{\uparrow}$, *and H, C, P, N from* hsame, Cont, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformHandoffScope`

Proof. Start with the root compact consumer nonchoice theorem. It sends compact-metric and compact-net consumers through the finite dyadic cover, mesh, radius, stream, regular-rational, real, transport, replay, provenance, and naming rows before any compact read is exposed.

Next use compact-net selector factorization. The displayed route $D, M, R, S, G, E, H, C, P, N \to K$ makes the selector row downstream of the same finite packet rather than a source of independent compactness.

Finally apply uniform-modulus radius exhaustion. The uniform-modulus consumer receives only the radius packet carried by those same rows, so compactness and uniform continuity meet at the displayed handoff and nowhere outside it. □

Theorem 452.45 (Finite Lebesgue number scoped closure package)**.** *The finite dyadic-cover obligations displayed in the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root coverage ledger form a scoped closure package: the root unblock package, selector-radius domain, compact-net selector factorization, uniform-modulus radius exhaustion, radius lower-bound fold, cover-shrinkage stability, UniformModulus consumer exactness, radius-modulus transport scope, and compact uniform handoff scope all use the same finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet. Hence the chapter closes the finite Lebesgue-number scope for explicit dyadic-cover radius transport and compact-to-uniform handoff.*

Proof. First assemble the existing root route. The root unblock package exposes the dyadic cover, dyadic mesh, rational radius, stream, regular-rational, real endpoint, compact-metric, compact-net selector, transport, replay, provenance, and local naming rows in one finite packet.

The selector-radius domain, compact-net selector factorization, and uniform-modulus radius exhaustion theorems then route the positive radius from that packet toward selector and uniform-modulus consumers. No step introduces an external open cover, selected subcover, host minimum, or choice coordinate.

Insert the radius lower-bound fold and cover-shrinkage stability rows. They turn the displayed rational-ball radius into the finite lower-bound packet and keep cover shrinkage inside the same transported and replayed rows.

Add UniformModulus consumer exactness, radius-modulus transport scope, and compact uniform handoff scope. These rows state that both the modulus transport and the compact-to-uniform handoff consume the same displayed dyadic-cover packet, so the finite Lebesgue-number scope is closed at the displayed rows. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberScopedClosurePackage`

Theorem 452.46 (Finite Lebesgue number radius-cover monotone readback)**.** *In every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root coverage ledger* $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$*, the positive radius row* R *is readable only after the displayed* $\mathsf{DyadicCover}^{\uparrow}$ *row* D *and the* $\mathsf{RationalBall}^{\uparrow}$ *readback rows attached to* D *have been exposed. Any* hsame *transport in* H *or* Cont *replay in* C *may extend the read toward* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *consumers, but it cannot detach* R *from the finite cover row or reverse the finite read order.*

Proof. Start with the carrier and root coverage ledger. The carrier of Definition 452.1 places the finite cover, radius ledger, mesh ledger, transport, replay, provenance, and local naming rows in one packet, while Definition 452.29 displays them as $\Lambda = (D, M, R, S, G, E, K, H, C, P, N)$.

Use the NameCert obligations, radius transport, and finite-radius nonchoice rows. Definition 452.2, Definition 452.4, and Definition 452.7 require every radius read to pass through the rational-ball rows tied to the displayed dyadic cover and mesh evidence.

Next apply the selector-radius domain and radius-modulus transport scope. They keep R downstream of D and upstream of every compact-net or modulus-facing consumer, with H and C acting only as stored transport and replay rows.

The readback is therefore monotone in the finite ledger order: transport and replay can carry the displayed radius to later consumers, but cannot create an earlier host radius, open-cover compactness row, selected subcover, or external modulus function. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusCoverMonotoneReadback`

Theorem 452.47 (Finite Lebesgue number mesh-refinement cover exhaustion)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *root coverage ledger, each mesh-refinement read from* DyadicMesh$^{\uparrow}$ *exhausts into a named cover cell through the displayed* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The exhaustion is finite: no mesh cell may request an arbitrary open cover, a selected subcover, host compactness, quotient equality, or a cover-cell selector outside the ledger.*

Proof. Begin with finite-window coverage exactness and the obligation coverage row. Definition 452.5 and Definition 452.23 send every accepted window read through a StreamName$^{\uparrow}$ row, a RegSeqRat$^{\uparrow}$ approximation row, a Real$^{\uparrow}$ row, a mesh cell, and a named dyadic cover cell.

Attach the mesh-refinement nonchoice and radius-mesh stability rows. Definition 452.8 keeps the mesh-to-cover handoff stable under the stored hsame transports and replayed Cont routes, while Definition 452.24 keeps the mesh evidence tied to the positive rational-ball radius readback.

Now read the root window cover totality theorem and the root unblock package. They place the StreamName, RegSeqRat, Real, mesh, cover, transport, replay, provenance, and local naming rows in the same finite BHist packet before compact consumers inspect the ledger.

Thus every mesh-refinement read exhausts into a named cover cell already present in the displayed packet. Since the packet has no coordinate for open-cover compactness, arbitrary subcover selection, host compactness, quotient equality, or an external cell selector, the exhaustion is nonchoice. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberMeshRefinementCoverExhaustion`

Theorem 452.48 (Finite Lebesgue number compact-route dyadic modulus package)**.** *The compact route from an accepted* FiniteLebesgueNumber$^{\uparrow}$ *ledger to* CompactMetric$^{\uparrow}$ *and* CompactNetModulusSelector$^{\uparrow}$ *consumers is exactly the finite dyadic modulus packet*

$$D, R, M, S, G, E, H, C, P, N \to K.$$

It packages the displayed DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows for compact consumers, and it exports no open-cover compactness theorem, host compactness witness, arbitrary subcover selector, quotient equality, or external modulus function.*

Proof. Start from the compact consumer route and compact consumer nonchoice boundary. Definition 452.6 and Definition 452.16 identify the compact-facing data as the displayed finite carrier and exclude open-cover compactness, host compactness, arbitrary subcover choice, and external selectors.

Use the root compact consumer nonchoice theorem with compact-net selector factorization. Definition 452.31 sends both compact-metric and compact-net consumers through $D, M, R, S, G, E, H, C, P, N$, and Definition 452.35 records K as downstream of that finite packet.

Apply radius-cover monotone readback and mesh-refinement cover exhaustion. The first keeps the positive radius row readable only after the displayed cover and rational-ball rows, and the second exhausts every mesh-refinement read into a named cover cell through the StreamName, RegSeqRat, Real, transport, replay, provenance, and naming rows.

Finally add the compact uniform handoff scope and scoped closure package. These results state that compact and uniform consumers meet at the same finite dyadic-cover packet, so the compact route is precisely $D, R, M, S, G, E, H, C, P, N \to K$ and not an open-cover compactness or choice-sensitive subcover route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactRouteDyadicModulusPackage certificate`

Theorem 452.49 (Finite Lebesgue number public compact-uniform export)**.** *The public* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *export for compact-uniform consumers is exactly the finite packet*

$$D, R, M, S, G, E, H, C, P, N \to K.$$

Here D is the displayed $\mathsf{DyadicCover}^{\uparrow}$ *row, R the* $\mathsf{RationalBall}^{\uparrow}$ *radius readback, M the* $\mathsf{DyadicMesh}^{\uparrow}$ *refinement ledger, S the* $\mathsf{StreamName}^{\uparrow}$ *row, G the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, E the* $\mathsf{Real}^{\uparrow}$ *row, H, C, P, N the* hsame, Cont, Pkg, *and* NameCert *rows, and K the compact-net selector handoff read by* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$ *consumers. The export has no coordinate for open-cover compactness, selected subcovers, host minima, quotient equality, choice, or external modulus functions.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPublicCompactUniformExport`

Proof. Begin with the scoped closure package. Definition 452.45 already binds the root unblock package, selector-radius domain, compact-net selector factorization, uniform-modulus radius exhaustion, lower-bound fold, cover-shrinkage stability, consumer exactness, and compact uniform handoff to the same finite BHist, hsame, Cont, Pkg, and NameCert packet.

Next attach radius-cover monotone readback and mesh-refinement cover exhaustion. Definition 452.46 keeps R downstream of the displayed cover and rational-ball rows, while Definition 452.47 sends every mesh read into a named cover cell through $D, R, M, S, G, E, H, C, P, N$.

Finally apply the compact-route dyadic modulus package. Definition 452.48 identifies the compact-facing route as $D, R, M, S, G, E, H, C, P, N \to K$. Since the preceding results place both compact and uniform-modulus reads inside that same finite route, the public export is the displayed packet and nothing outside it: no open-cover compactness, selected subcover, host minimum, quotient equality, choice, or external modulus function is exported. □

Theorem 452.50 (Finite Lebesgue-number compact-radius bridge)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives the positive finite-radius ledger only through the displayed route*

$$D, W, \lambda, M, H, C, P, N.$$

The bridge route passes through $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before the compact or compact-net consumer reads the radius ledger. It exports no open-cover compactness, selected subcover, host minimum, quotient equality, or external radius selector.*

Proof. Start with radius transport and mesh-refinement nonchoice. Definition 452.4 keeps λ tied to the displayed rational-ball readback, and Definition 452.8 keeps the mesh-to-cover route inside the same stored transport and replay rows.

Next use the total-bounded handoff and compact-uniform selector handoff. Definition 452.9 identifies $D, W, \lambda, M, H, C, P, N$ as the compact-facing finite packet, while Definition 452.10 sends the same positive radius ledger to compact-net and uniform consumers.

Finally apply the L10 source-lock. Definition 452.13 forces the compact-radius read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, and $\mathsf{DyadicCover}^{\uparrow}$ before any consumer inspects the ledger. Hence the bridge is exactly the displayed dyadic finite-radius route and has no open-cover or choice coordinate. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactRadiusBridge`

Theorem 452.51 (Finite Lebesgue-number uniform-modulus radius scope)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *handoff from an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier may read only the displayed positive finite-radius row in* $D, W, \lambda, M, H, C, P, N$*. The handoff cannot add open-cover compactness, a selected subcover, a host minimum,* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, or an external modulus function to the radius scope.*

Proof. Begin with the compact-uniform selector handoff. It already places the uniform-modulus read downstream of the positive radius ledger carried by the displayed dyadic cover, finite-window, rational-ball, mesh, transport, replay, provenance, and local naming rows.

Use total-bounded handoff and window-radius ledger coherence. The total-bounded handoff fixes the compact-facing packet as $D, W, \lambda, M, H, C, P, N$, and window-radius ledger coherence prevents a window read from acquiring a separate positive radius outside the same packet.

The compact consumer nonchoice boundary supplies the negative side. It excludes open-cover compactness, selected subcovers, host minima, quotient equality, and external selectors from the compact route, so the uniform-modulus handoff inherits only the displayed positive finite-radius row and not a stronger compactness or completeness principle. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformModulusRadiusScope`

Theorem 452.52 (Finite Lebesgue-number dyadic-cover Real route)**.** *Every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *finite-window read reaches* $\mathsf{Real}^{\uparrow}$ *only through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *before mesh and cover-cell readback. The resulting dyadic-cover route then continues through* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicCover}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*; no host real representative, open cover, selected subcover, or arbitrary cover-cell selector supplies the endpoint.*

Proof. Start from finite-window coverage exactness and stream-regular window order. These results force each finite-window read to pass first through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, then through $\mathsf{Real}^{\uparrow}$, before any mesh cell or dyadic cover cell is read.

Apply the Real source nonchoice boundary. It rules out host real equality, arbitrary point selection, open-cover compactness, subcover choice, and external real representatives as sources for the $\mathsf{Real}^{\uparrow}$ row.

Finally use cover-cell readback totality. The mesh and cover-cell endpoint is packaged by the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the Real route reaches the dyadic cover only through the stated stream and regular-rational sources. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberDyadicCoverRealRoute`

Theorem 452.53 (Finite Lebesgue-number source-chain admission)**.** *An accepted* FiniteLebesgueNumber$^{\uparrow}$ *route is admitted only after the displayed* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, *and* DyadicMesh$^{\uparrow}$ *rows pass through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *source rows, all carried by the same* BHist, hsame, Cont, Pkg, *and* NameCert *packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSourceChainAdmission`

Proof. Use finite-window coverage exactness to place every finite read inside the displayed window packet.

Then apply Definition 452.52. It forces the source chain through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before any dyadic mesh or cover-cell endpoint is admitted.

Finally use Definition 452.48. The compact-facing package is exactly the same BHist, hsame, Cont, Pkg, and NameCert route, so admission cannot bypass the displayed L10 source rows. □

Theorem 452.54 (Finite Lebesgue-number ledger transport)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier,* hsame *transport preserves the positive radius ledger together with the dyadic cover, mesh,* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows. The radius is never transported as a detached scalar outside the displayed finite source packet.*

Proof. Start with Definition 452.24. It keeps the positive rational-ball read attached to the dyadic mesh evidence under stored hsame transport.

Next use Definition 452.26. The finite-window and radius ledgers share the same dyadic source packet, including the stream, regular-rational, and Real rows.

Apply Definition 452.46. Every transported radius read remains downstream of the displayed cover and rational-ball rows, so no scalar radius separates from the finite ledger. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberLedgerTransport`

Theorem 452.55 (Finite Lebesgue-number consumer nonescape triad)**.** *The* CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *consumers of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier receive only the displayed finite route. They cannot import open-cover compactness, a host minimum, a selected subcover, quotient equality, choice, or external modulus data.*

Proof. Use Definition 452.25 to fix the compact and compact-net consumer boundary at the displayed finite packet.

Then apply Definition 452.44. The uniform-modulus handoff reads the same radius and cover route rather than a stronger compactness principle.

Finally use Definition 452.49. The public export has no coordinate for open-cover compactness, host minima, selected subcovers, quotient equality, choice, or external modulus data, so the three consumers cannot escape the finite route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberConsumerNonescapeTriad`

Definition 452.56 (Finite Lebesgue-number selected-tail bridge ledger)**.** *The selected-tail bridge ledger for an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier is the finite* BHist *packet*

$$J = (D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N).$$

Here D, W, λ, M are the displayed FiniteLebesgueNumber$^{\uparrow}$ *dyadic-cover, finite-window, rational-ball radius, and dyadic-mesh rows. The rows A_T, W_T, K_T, S_T, R_T are the selected finite-tail admission, stream-window, regular-rational readback, dyadic tolerance ledger, and Real seal rows supplied by*

the current SelectedTailSeed$^{\uparrow}$ *seed. The structural rows* H, C, P, N *record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the compatible local* NameCert *rows.*

The ledger is accepted only when the finite Lebesgue radius read and the selected-tail Real-window read are carried by the same structural support. Thus the compact-uniform route may consume λ *only together with* $D, W, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N$.

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectedTailBridgeLedger`
Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectedTailBridgeLedger certificate`

Theorem 452.57 (Selected-tail radius admission)**.** *Any accepted finite Lebesgue radius read entering the compact-uniform route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *and the selected-tail seed handoff before* CompactNetModulusSelector$^{\uparrow}$ *consumes the radius. More precisely, if the read is admitted by the ledger* J *of Definition 452.56, then the consumer-facing radius is the same finite packet route*

$$D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N$$

and not a radius obtained from open-cover compactness, host real equality, selected subcovers, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectedTailRadiusAdmission`

Proof. Begin with the FiniteLebesgueNumber$^{\uparrow}$ carrier. The carrier definition Definition 452.1 and the root package theorem Definition 452.33 identify the radius read as the displayed dyadic-cover, finite-window, rational-ball, and mesh packet D, W, λ, M with the same H, C, P, N support used by compact consumers.

Next read the selected-tail seed scope. The stream-regular readback theorem Definition 1207.18 binds the W_T, K_T part of the route to StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$. The dyadic Real-seal theorem Definition 1207.19 keeps S_T, R_T as the finite DyadicRatCore$^{\uparrow}$-to-Real$^{\uparrow}$ seal.

Finally apply the finite-tail Real-boundary theorem Definition 1207.20. It identifies A_T, W_T, K_T, S_T, R_T as the selected-tail boundary consumed downstream. Since the bridge ledger stores this boundary beside D, W, λ, M, the compact-net selector receives the finite radius only after the selected-tail source route has been displayed. □

Theorem 452.58 (Selected-tail uniform consumer route)**.** *Every* UniformModulus$^{\uparrow}$ *consumer downstream of an accepted selected-tail bridge ledger receives the same finite radius packet as* CompactNetModulusSelector

$$D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N.$$

The consumer cannot replace this packet by open-cover compactness, host minima, selected subcovers, quotient equality, choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectedTailUniformConsumerRoute`

Proof. Use Definition 452.57 to fix the compact-net input. The admitted radius already factors through the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ selected-tail route before the compact-net selector can consume it.

The compact-uniform handoff Definition 452.44 and the consumer nonescape triad Definition 452.55 then pass the same displayed radius packet from CompactNetModulusSelector$^{\uparrow}$ to UniformModulus$^{\uparrow}$. These rows expose the finite cover, mesh, rational-ball radius, selected-tail

source boundary, and structural support; they do not expose a second compactness or modulus interface.

The exclusions are coordinate exclusions. Open-cover compactness, host minima, selected subcovers, quotient equality, choice, and ambient Real completeness are absent from J, while H, C, P, N only transport, replay, source, and name the rows already listed. Hence the uniform consumer route is the same finite selected-tail bridge packet. □

Theorem 452.59 (Finite Lebesgue-number total-bounded radius source lock)**.** *Any* $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer that preserves the selected-tail bridge ledger of Definition 452.56 can read the positive radius only through the same finite packet*

$$D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N.$$

Equivalently, the consumer-facing radius is unavailable unless the selected finite-tail admission, stream-window row, regular-rational readback, dyadic tolerance ledger, Real-seal row, finite Lebesgue radius row, dyadic-cover row, finite-window row, mesh row, componentwise transport, continuation replay, provenance, and local naming rows are all present. The route contains no open-cover compactness, host minimum, selected subcover, quotient equality, choice, or ambient Real completeness coordinate.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTotalBoundedRadiusSourceLock`

Proof. Start with the selected-tail radius admission theorem, Definition 452.57. It identifies the compact-net input radius with the displayed selected-tail packet and prevents the radius from being read before the stream, regular-rational, dyadic tolerance, and Real-seal rows have appeared.

Next use the selected-tail uniform consumer route, Definition 452.58. That theorem passes the same packet from $\mathsf{CompactNetModulusSelector}^{\uparrow}$ to $\mathsf{UniformModulus}^{\uparrow}$ and records that the uniform consumer receives no second compactness or modulus interface.

The root finite Lebesgue-number NameCert obligations, Definition 452.2, together with the root unblock package Definition 452.33, supply the finite dyadic-cover, finite-window, rational-ball radius, mesh, transport, continuation, provenance, and naming rows. These obligations keep λ attached to D, W, M, H, C, P, N before the selected-tail rows are joined to the route.

Finally apply the compact consumer nonchoice boundary, Definition 452.16. It keeps $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{CompactNetModulusSelector}^{\uparrow}$ consumers inside the displayed finite route and excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and ambient Real completeness. Therefore any preserving compact or uniform consumer reads the positive radius exactly through $D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N$. □

Theorem 452.60 (Finite Lebesgue-number bridge packet exactness)**.** *The* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *bridge packet displayed to compact and uniform consumers is exactly*

$$D, W, \lambda, M, H, C, P, N.$$

The route reads D *through* $\mathsf{DyadicCover}^{\uparrow}$, λ *through* $\mathsf{RationalBall}^{\uparrow}$, M *through* $\mathsf{DyadicMesh}^{\uparrow}$, *the finite-window sources through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and the consumer endpoints through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$. *No coordinate outside this tuple contributes to the bridge packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberBridgePacketExactness`

Proof. Start with the compact-radius bridge. Definition 452.50 identifies the compact and compact-net radius route as the displayed tuple $D, W, \lambda, M, H, C, P, N$ and forces the read through $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next use source-chain admission and ledger transport. Definition 452.53 admits the route only after the displayed source rows pass through the same BHist, hsame, Cont, Pkg, and NameCert packet, while Definition 452.54 prevents the positive radius ledger from separating from the cover, mesh, stream, regular-rational, and real rows.

Finally apply uniform-modulus radius scope. Definition 452.51 sends the uniform consumer through the same positive finite-radius row and excludes any added compactness, completeness, quotient, or external modulus coordinate. Therefore the bridge packet is exactly the displayed tuple. □

Theorem 452.61 (Finite Lebesgue-number compact-uniform bridge soundness)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives the same finite radius route*

$$D, W, \lambda, M, H, C, P, N.$$

The compact and uniform endpoints may inspect the displayed dyadic cover, rational-ball radius, mesh ledger, stream source, regular-rational approximation, real row, transport rows, continuations, package provenance, and local NameCert *row; they do not receive an open-cover coordinate, a subcover-choice coordinate, or an external radius-modulus coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformBridgeSoundness`

Proof. First fix the compact side. Definition 452.50 sends $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{CompactNetModulusSelector}^{\uparrow}$ consumers through the displayed positive finite-radius route and excludes open-cover compactness, selected subcovers, host minima, quotient equality, and external radius selectors.

Now fix the uniform side. Definition 452.51 states that a $\mathsf{UniformModulus}^{\uparrow}$ handoff reads only the displayed radius row in the same tuple and cannot add host compactness, real completeness, quotient equality, or an external modulus function.

The dyadic-cover Real route supplies the shared source order. Definition 452.52 forces finite-window reads through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before mesh and cover-cell readback. Thus compact, compact-net, and uniform consumers receive one finite radius route rather than separate compactness or choice surfaces. □

Theorem 452.62 (Finite Lebesgue-number bridge nonescape)**.** *The* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *bridge packet cannot import host compactness, selected subcovers, quotient equality,* $\mathsf{Real}^{\uparrow}$ *completeness, or external modulus data. Every bridge-facing read remains inside the displayed dyadic finite-cover route*

$$D, W, \lambda, M, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberBridgeNonescape`

Proof. Use the consumer nonescape triad. Definition 452.55 excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and external modulus data for compact, compact-net, and uniform consumers.

Next use compact-uniform bridge soundness. Definition 452.61 identifies the only consumer route as $D, W, \lambda, M, H, C, P, N$ and lists the permitted cover, radius, mesh, stream, regular-rational, real, transport, continuation, provenance, and naming rows.

Finally use bridge packet exactness. Definition 452.60 leaves no bridge coordinate outside the displayed tuple. Hence none of host compactness, selected subcovers, quotient equality, real completeness, or external modulus data can enter the bridge packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ packet is generated from an accepted $\mathsf{DyadicCover}^{\uparrow}$ finite cover, a $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ row, a positive rational radius ledger, a $\mathsf{DyadicMesh}^{\uparrow}$ refinement ledger, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public finite-cover radius route is closed by the carrier, local NameCert obligations, root unblock package, selector-radius domain, compact-net selector factorization, uniform-modulus radius exhaustion, radius lower-bound fold Definition 452.37, cover-shrinkage stability Definition 452.38, UniformModulus consumer exactness Definition 452.39, radius-modulus transport scope Definition 452.40, compact uniform handoff scope Definition 452.44, scoped closure package Definition 452.45, radius-cover monotone readback Definition 452.46, mesh-refinement cover exhaustion Definition 452.47, compact-route dyadic modulus package Definition 452.48, public compact-uniform export Definition 452.49, bridge packet exactness Definition 452.60, compact-uniform bridge soundness Definition 452.61, and bridge nonescape Definition 452.62.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLebesgueNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close open-cover compactness, arbitrary subcover choice, host real compactness, total-bounded construction, Lean verification, or bridge checking.

Upgrade path: The remaining obligation is Lean verification and checked standard-bridge audit for the displayed finite dyadic-cover packet.

Theorem 452.63 (Compact radius source-accountability)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *radius read from an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier factors through the displayed source-accountability surface*

$$\mathsf{DyadicCover}^{\uparrow},\ \mathsf{DyadicMesh}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The compact or compact-net consumer may read the positive radius only as part of that finite dyadic packet. It receives no open-cover compactness, selected subcover, host minimum, quotient equality, choice, or radius selector outside the displayed BHist *ledger.*

Proof. Start with the compact-uniform selector handoff. Definition 452.10 sends every compact-net and uniform-facing radius read through the displayed finite cover, finite window, rational radius, dyadic mesh, transport, continuation, provenance, and local naming rows.

Now apply the L10 source-lock. Definition 452.13 forces the consumer-facing radius route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, and $\mathsf{DyadicCover}^{\uparrow}$ before any compact or compact-net consumer can inspect the positive ledger.

Finally use the compact-route dyadic modulus package. Definition 452.48 identifies the compact-facing packet with the finite dyadic modulus route carried by BHist, hsame, Cont, Pkg, and NameCert. Since the packet has no open-cover, subcover-selection, host-minimum, quotient-equality, choice, or detached selector coordinate, every compact radius read has the stated source-accountability surface. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactRadiusSourceAccountability`

Theorem 452.64 (Uniform modulus source factorization)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *packet reads the same positive finite-radius ledger used by* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *consumers. The factorization runs through the displayed* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, rather than through an external modulus function or an independently supplied continuity surface.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformModulusSourceFactorization`

Proof. First read the compact radius source-accountability theorem above. It fixes the compact and compact-net radius source as the displayed dyadic packet and forbids a detached radius selector before the uniform consumer appears.

Next apply the compact-uniform selector handoff and the compact-route dyadic modulus package. These rows pass the same positive ledger from the compact route to the compact-net selector and keep the ledger inside the finite dyadic modulus packet.

Use the consumer nonescape triad for the final boundary. Definition 452.55 states that $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers receive only the displayed finite route. Hence the uniform modulus consumer reads the already displayed positive radius ledger and cannot import an external modulus function, a separate continuous-map modulus, open-cover compactness, quotient equality, or choice. □

Theorem 452.65 (Total-bounded ledger readback)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *handoff downstream of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier receives the displayed finite dyadic cover ledger*

$$D, W, \lambda, M, H, C, P, N.$$

Its readback may inspect the $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows already present in that ledger, but it cannot recover open-cover compactness, selected subcovers, quotient equality, choice, host compactness, or an external modulus surface.*

Proof. Begin with the total-bounded handoff obligation. Definition 452.9 identifies $D, W, \lambda, M, H, C, P, N$ as the exact packet delivered to $\mathsf{TotallyBounded}^{\uparrow}$ and compact-metric consumers.

Attach the L10 source-lock and the compact radius source-accountability theorem above. Together they force the radius, mesh, stream, regular-rational, real, transport, continuation, provenance, and local naming reads to occur inside the displayed finite dyadic ledger before any total-bounded or compact-facing consumer can use the handoff.

Finally invoke the consumer nonescape triad. Definition 452.55 excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and external modulus data from the downstream consumer route. Therefore the total-bounded handoff reads back exactly the displayed ledger and cannot reconstruct any stronger compactness or selection principle from it. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTotalBoundedLedgerReadback`

Definition 452.66 (Finite Lebesgue-number compact-continuous unblock ledger)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *carrier, the compact-continuous unblock ledger is the displayed finite packet*

$$U_{\mathrm{cc}} = (D, W, \lambda, M, H, C, P, N; K, F, Q)$$

where $D, W, \lambda, M, H, C, P, N$ *are the dyadic cover row, finite-window row, positive rational-ball radius ledger, dyadic mesh ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row of the finite Lebesgue-number carrier. The row* K *is the compact-net consumer row over* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{CompactNetModulusSelector}^{\uparrow}$, F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus consumer row over the same finite-window source, and* Q *is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consuming the radius read from* λ.

The ledger is accepted only when K, F, Q *read the same finite dyadic-radius packet* $D, W, \lambda, M, H, C, P, N$ *through the same* BHist, hsame, Cont, Pkg, *and* NameCert *support. It has no coordinate for open-cover compactness, selected subcovers, host minima, quotient equality, choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or an external modulus selector.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactContinuousUnblockLedger`

Theorem 452.67 (Finite Lebesgue-number compact-continuous triad)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer admitted by the compact-continuous unblock ledger of Definition 452.66 reads the same finite dyadic-radius packet*

$$D, W, \lambda, M, H, C, P, N.$$

The compact row, compact-net row, continuous-map pointwise row, and uniform-modulus handoff row are therefore a single BHist *route rather than four independent parameter surfaces.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactContinuousTriad`

Proof. Begin with finite-window coverage exactness. Definition 452.5 sends every accepted window read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, the mesh row M, and the named dyadic cover row D before any consumer can inspect the radius ledger.

Next attach the compact consumer nonchoice boundary and compact-uniform selector handoff. Definition 452.16 keeps compact and compact-net consumers inside the displayed finite packet, while Definition 452.10 sends the same positive radius ledger from that packet to the compact-net and uniform endpoints.

Use source-chain admission and ledger transport for the shared support. Definition 452.53 admits the route only through the common BHist, hsame, Cont, Pkg, and NameCert rows, and Definition 452.54 prevents λ from separating from the dyadic cover, finite window, mesh, stream, regular-rational, and real rows.

Finally apply the consumer nonescape triad. Definition 452.55 excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and external modulus data. The continuous-map row F in the ledger is therefore read only as a pointwise consumer of the same finite radius packet, and the four consumer rows form the stated compact-continuous triad. □

Theorem 452.68 (Finite Lebesgue-number source-radius unblock)**.** *In every accepted compact-continuous unblock ledger* U_{cc}*, a source-side radius request reaches* λ *only after the displayed finite-window source has passed through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, *and* $\mathsf{DyadicCover}^{\uparrow}$ *rows carried by* H, C, P, N*. Thus the root unblock surface for compact, compact-net, continuous-map, and uniform-modulus consumers is the finite dyadic-radius packet itself, not an open-cover or host-completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSourceRadiusUnblock`

Proof. Start with the source-chain admission theorem. It requires the dyadic cover, rational-ball radius, and mesh rows to pass through the stream-name, regular-rational, and real rows before the route is admitted into the shared support.

Now use finite-window coverage exactness. Each accepted read from W lands in a named mesh and cover cell before the radius ledger is visible, so a source request cannot enter through a detached real representative or an arbitrary open family.

Transport the radius with ledger transport. The hsame and Cont rows preserve the positive radius together with the cover, mesh, stream, regular-rational, and real rows; λ is never a scalar outside the finite packet.

The compact consumer nonchoice boundary supplies the negative side. It refuses open-cover compactness, selected subcovers, host minima, quotient equality, choice, detached $\mathsf{Real}^{\uparrow}$ completeness, and external selectors, leaving the finite dyadic-radius packet as the only root unblock surface. □

Theorem 452.69 (Finite Lebesgue-number uniform-continuity handoff)**.** *Every* UniformModulus$^{\uparrow}$ *handoff accepted by the compact-continuous unblock ledger consumes the* ContinuousMap$^{\uparrow}$ *pointwise row only after* CompactMetric$^{\uparrow}$ *and* CompactNetModulusSelector$^{\uparrow}$ *have exposed the same positive finite-radius packet* $D, W, \lambda, M, H, C, P, N$*. The handoff cannot replace that packet with a selected subcover, a host minimum, quotient equality, choice, detached* Real$^{\uparrow}$ *completeness, or an external modulus selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformContinuityHandoff`

Proof. First use compact-uniform selector handoff. It places the uniform-modulus read downstream of the positive radius ledger already carried by $D, W, \lambda, M, H, C, P, N$.

Next use the compact radius source-accountability theorem. Definition 452.63 identifies the compact and compact-net source of the radius as the displayed dyadic cover, mesh, stream-name, regular-rational, real, transport, continuation, provenance, and naming rows.

Read the continuous-map row through the ledger definition. The pointwise row F is admitted only as a consumer row over the same finite-window source and the same BHist, hsame, Cont, Pkg, and NameCert support, so it cannot supply an independent global modulus surface.

Finally invoke uniform modulus source factorization and consumer nonescape. Definition 452.64 sends the uniform consumer through the same source packet, and Definition 452.55 removes open-cover compactness, selected subcovers, host minima, quotient equality, choice, detached real completeness, and external modulus selectors from the handoff. □

Theorem 452.70 (Finite Lebesgue-number L10 sibling route)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *consumer route that exposes the positive finite-radius ledger to a compact, compact-net, continuous-map, or uniform-modulus consumer factors through the L10 finite-name chain chapter 126, chapter 301, chapter 291, and chapter 38, and then through the dyadic mesh and cover surfaces of chapter 358 and chapter 446. Thus a compact-continuous consumer can read the positive radius only as part of the displayed finite route*

$$D, W, \lambda, M, H, C, P, N.$$

The route has no coordinate for open-cover compactness, host minima, selected subcovers, quotient equality, choice, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Start with the root carrier and NameCert obligations. Definition 452.1 displays $D, W, \lambda, M, H, C, P, N$, while Definition 452.2 makes every radius, mesh, window, transport, continuation, provenance, and local naming read a finite component of that packet.

Next attach the source-accountability rows. Definition 452.63 forces compact and compact-net radius reads through the visible StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, DyadicMesh$^{\uparrow}$, and DyadicCover$^{\uparrow}$ surface. Definition 452.57 inserts the selected-tail DyadicRatCore$^{\uparrow}$ seal before the compact-net selector consumes the radius.

The selected-tail bridge ledger then keeps the source route finite. Its packet J records the stream-window, regular-rational readback, dyadic tolerance ledger, real seal, finite Lebesgue radius row, and the same H, C, P, N support; no row in J supplies an open family, a host minimum, a quotient equality principle, or an ambient completeness theorem.

Finally apply the compact-continuous triad. Definition 452.67 and Definition 452.69 make CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, and UniformModulus$^{\uparrow}$ consumers read the same finite dyadic-radius packet. Hence every accepted compact-continuous route factors through the displayed L10 sibling chain before the positive radius is visible, and the excluded compactness, selection, equality, choice, and completeness data remain absent. □

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberL10SiblingRoute

Definition 452.71 (Finite Lebesgue-number radius-consumer determinacy ledger)**.** *The radius-consumer determinacy ledger for* FiniteLebesgueNumber$^{\uparrow}$ *is the finite row family*

$$\Delta_\lambda = (D, R, M, S, G, E, H, C, P, N, K, Q)$$

whose displayed entries are the DyadicCover$^{\uparrow}$ *cover row D, the* RationalBall$^{\uparrow}$ *positive radius row R, the* DyadicMesh$^{\uparrow}$ *mesh row M, the* StreamName$^{\uparrow}$ *source-window row S, the* RegSeqRat$^{\uparrow}$ *regular-rational readback row G, the* Real$^{\uparrow}$ *endpoint row E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, local* NameCert *naming N, the compact and compact-net consumer row K, and the* UniformModulus$^{\uparrow}$ *consumer row Q. The ledger determines the radius read available to* CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *consumers only through these finite rows. It has no coordinate for an arbitrary open cover, selected subcover, host minimum, quotient equality, choice principle, detached* Real$^{\uparrow}$ *representative, or external modulus selector.*

Lean4 (definition) BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusConsumerDeterminacyLedger

Theorem 452.72 (Finite Lebesgue-number radius-row determinacy)**.** *Let L and L' be accepted* FiniteLebesgueNumber$^{\uparrow}$ *routes whose radius-consumer ledgers display the same* DyadicCover$^{\uparrow}$ *cover row,* RationalBall$^{\uparrow}$ *positive radius row,* DyadicMesh$^{\uparrow}$ *mesh row,* StreamName$^{\uparrow}$ *source-window row,* RegSeqRat$^{\uparrow}$ *regular-rational row,* Real$^{\uparrow}$ *endpoint row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming row. Then every compact, compact-net, or uniform consumer admitted by those routes exposes the same positive finite-radius row.*

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusRowDeterminacy

Proof. Start with radius transport. Definition 452.4 keeps the positive radius tied to the rational-ball and mesh evidence under the displayed transport rows.

Next apply the window-radius source lock. Definition 452.26 identifies the finite-window source of the radius through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, DyadicMesh$^{\uparrow}$, and DyadicCover$^{\uparrow}$ before a consumer can inspect the radius.

Use radius-cover monotone readback and ledger transport for the comparison. Definition 452.46 fixes the read order from the cover row to the rational-ball row, while Definition 452.54 prevents the radius from separating from the cover, mesh, source, transport, replay, provenance, and naming rows.

Since L and L' agree on exactly those displayed rows, no compact, compact-net, or uniform consumer has another radius coordinate to read. The exposed positive radius row is therefore the same row in both routes. □

Theorem 452.73 (Finite Lebesgue-number uniform-consumer determinacy)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *radius-consumer determinacy ledger Δ_λ, a* UniformModulus$^{\uparrow}$ *consumer cannot obtain a modulus radius different from the positive finite-radius row displayed in Δ_λ. Any uniform-modulus handoff preserving the displayed cover, rational-ball, mesh, source, transport, replay, provenance, and naming rows reads the same radius as the compact and compact-net consumers.*

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformConsumerDeterminacy

Proof. Begin with the uniform-modulus radius scope theorem. Definition 452.51 places every $\mathsf{UniformModulus}^{\uparrow}$ handoff inside the displayed positive finite-radius row.

Now use the consumer nonescape triad. Definition 452.55 excludes open-cover compactness, selected subcovers, host minima, quotient equality, choice, and external modulus data from compact, compact-net, and uniform consumers alike.

Finally apply the public compact-uniform export. Definition 452.49 identifies the public export as the same finite cover, rational-ball, mesh, source, transport, replay, provenance, and naming route. Thus the uniform consumer has no second modulus-radius source and must read the radius already determined by Δ_λ. □

Theorem 452.74 (Finite Lebesgue-number compact-consumer determinacy package)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius-consumer determinacy ledger factors through one deterministic finite route*

$$D, R, M, S, G, E, H, C, P, N, K, Q.$$

The compact metric read, compact-net selector read, and uniform-modulus handoff therefore expose the same positive radius row through the displayed $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *data.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactConsumerDeterminacyPackage`

Proof. First package the compact side. Definition 452.50 sends compact and compact-net consumers through the displayed positive finite-radius route, and Definition 452.72 makes the radius row fixed by the shared ledger rows.

Next package the uniform side. Definition 452.73 states that a $\mathsf{UniformModulus}^{\uparrow}$ consumer preserving the same ledger cannot read a different modulus radius.

It remains to identify the shared finite route. Definition 452.71 lists exactly the cover, radius, mesh, source, transport, replay, provenance, naming, compact-consumer, and uniform-consumer rows, while Definition 452.55 removes all outside compactness, equality, choice, and modulus coordinates. Hence the three consumer surfaces are packaged by the single deterministic route displayed above. □

Theorem 452.75 (Finite Lebesgue-number Phase Real source lock)**.** *Every accepted public radius read of* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *that is visible to a compact or continuous consumer factors through the displayed Phase Real and Completion route*

$$\mathsf{DyadicCover}^{\uparrow},\ \mathsf{RationalBall}^{\uparrow},\ \mathsf{DyadicMesh}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Thus the consumer-facing radius is not a host scalar and not an open-cover compactness witness: it is the finite dyadic-radius packet carried by the BHist, hsame, Cont, Pkg, *and* NameCert *rows of the finite Lebesgue-number carrier before any* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *or* $\mathsf{ContinuousMap}^{\uparrow}$ *endpoint can inspect it.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealSourceLock`

Proof. Start with the root obligations. Definition 452.2 fixes the dyadic cover, finite-window, rational-ball radius, mesh, transport, continuation, provenance, and local naming rows as the only public rows of the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ packet.

Next apply the L10 source lock. Definition 452.13 forces the radius route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, and $\mathsf{DyadicCover}^{\uparrow}$ before the compact side can read the radius.

Use the selected-tail bridge ledger and bridge packet exactness for the Phase Real and Completion handoff. Definition 452.56 records the finite selected-tail source rows, while Definition 452.60 leaves no bridge coordinate outside the displayed dyadic, stream, regular-rational, and real rows.

Finally use compact-continuous unblock. Definition 452.67 sends compact, compact-net, continuous-map, and uniform-modulus consumers through the same finite dyadic-radius packet. Hence every accepted public read factors through the stated Phase Real and Completion route. □

Theorem 452.76 (Finite Lebesgue-number selected-tail modulus exhaustion)**.** *In an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *route, every selected-tail witness and every finite-window witness used to expose a radius to a modulus consumer is exhausted by the displayed carrier rows*

$$D, W, \lambda, M, A_T, W_T, K_T, S_T, R_T, H, C, P, N.$$

No hidden modulus selector, quotient stream, countable choice coordinate, host real equality, or detached $\mathsf{Real}^{\uparrow}$ *representative can be inserted between the selected-tail source rows and the finite dyadic-radius packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberSelectedTailModulusExhaustion`

Proof. Read the finite-window side first. Definition 452.2 and Definition 452.13 make every window and radius read pass through the dyadic cover, rational-ball radius, mesh, stream-name, regular-rational, and real rows with the displayed structural support.

Now read the selected-tail side. Definition 452.56 lists exactly the selected finite-tail admission, stream-window, regular-rational readback, dyadic tolerance, and Real-seal rows joined to the finite Lebesgue-number packet.

Apply bridge packet exactness. Definition 452.60 states that the bridge packet has no coordinate beyond the displayed tuple. The excluded coordinates include quotient equality, host real equality, choice, detached real completeness, and external modulus data.

The modulus consumer therefore sees only the selected-tail and finite-window rows already named in the carrier. Since there is no further selector or stream coordinate in the packet, the listed rows exhaust the modulus-facing witness data. □

Theorem 452.77 (Finite Lebesgue-number compact-continuous nonchoice exit)**.** *Any* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer admitted downstream of* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *receives exactly the finite dyadic-radius packet*

$$\mathsf{DyadicCover}^{\uparrow}, \mathsf{RationalBall}^{\uparrow}, \mathsf{DyadicMesh}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCer}$$

The exit to these consumers does not supply open-cover compactness, an arbitrary subcover principle, a host minimum, countable choice, quotient stream equality, or an external modulus selector.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactContinuousNonchoiceExit`

Proof. Begin with source accountability. Definition 452.13 and Definition 452.75 identify the only source of the consumer-facing radius as the finite dyadic, stream, regular-rational, and real route.

Next use the selected-tail exhaustion theorem above. It prevents selected-tail or finite-window witnesses from contributing a hidden modulus selector, quotient stream, countable choice coordinate, host real equality, or detached real representative.

Finally apply the downstream consumer package. Definition 452.67 sends compact, compact-net, continuous-map, and uniform-modulus endpoints through the same finite packet, and Definition 452.60 removes all bridge coordinates outside that packet.

Therefore the consumer exit is nonchoice: the consumers receive the finite dyadic-radius packet and no open-cover compactness or arbitrary subcover principle. □

Theorem 452.78 (Finite Lebesgue-number real-phase coverage export)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *public read used by a* CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *endpoint factors through exactly the L10 Real-phase route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the finite dyadic cover, rational-ball radius, dyadic mesh, BHist, hsame, Cont, Pkg, *and* NameCert *rows of the finite Lebesgue-number carrier. The exported handoff supplies no host real equality, quotient stream equality, countable-choice coordinate, selected limit, open-cover compactness principle, or ambient-completeness primitive.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealPhaseCoverageExport`

Proof. Start with the Phase Real source lock. Definition 452.75 routes every compact- or continuous-visible radius read through the dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, and Real rows, with BHist, hsame, Cont, Pkg, and NameCert carrying the public BEDC packet.

Next use selected-tail exhaustion. Definition 452.76 rules out any hidden modulus selector, quotient stream, countable choice coordinate, host real equality, or detached Real$^{\uparrow}$ representative between the selected-tail source rows and the finite dyadic-radius packet.

Finally apply the compact-continuous nonchoice exit. Definition 452.77 says that the compact, compact-net, continuous-map, and uniform-modulus endpoints receive exactly that finite packet and no open-cover compactness, arbitrary subcover principle, host minimum, quotient stream equality, or external modulus selector.

Thus the L10 handoff exported to the downstream endpoints is the finite Real-phase coverage route through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$, together with the named finite carrier rows and no completed-real or choice-bearing primitive. □

Definition 452.79 (Finite Lebesgue-number root radius-window obligation ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier, the root radius-window obligation ledger is the finite* BHist *row*

$$\Omega_{\lambda W} = (D, W, \lambda, M, S, Q, R, H, C, P, N)$$

whose entries are the DyadicCover$^{\uparrow}$ *cover row D, the finite window row W, the positive radius row λ, the* DyadicMesh$^{\uparrow}$ *mesh row M, the* StreamName$^{\uparrow}$ *source row S, the* RegSeqRat$^{\uparrow}$ *regular-rational row Q, the* Real$^{\uparrow}$ *endpoint row R, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N. The ledger records the root obligation surface only through* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows; it has no coordinate for open-cover compactness, a host minimum, choice, quotient equality, selected limit data, or a separate completeness primitive.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusWindowObligationLedge`

Theorem 452.80 (Finite Lebesgue-number root radius-window obligation surface)**.** *Every accepted root radius-window obligation ledger of Definition 452.79 discharges the positive radius and finite-window obligations only through the displayed dyadic cover, rational-ball radius, dyadic mesh,*

stream-name, regular-rational, Real, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. In particular, the root obligation exposed to the Real-and-completion route is the finite packet*

$$D, W, \lambda, M, S, Q, R, H, C, P, N$$

and not an open-cover compactness principle, host minimum, choice coordinate, quotient equality, selected limit, or Real-completeness coordinate.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusWindowObligationSurface`

Proof. Begin with the carrier. Definition 452.1 displays the finite dyadic cover, finite window, radius, mesh, transport, continuation, provenance, and local naming rows of the FiniteLebesgueNumber$^{\uparrow}$ packet.

Apply the NameCert obligations and finite-window coverage exactness. Definition 452.2 keeps each public read inside the carrier rows, while Definition 452.5 routes every accepted window through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, the mesh row, and the named dyadic cover row.

Now apply the L10 and Phase Real source locks. Definition 452.13 and Definition 452.75 force the positive-radius read through the dyadic, stream, regular-rational, and Real rows before any consumer can inspect it.

Thus the ledger of Definition 452.79 is exactly the root radius-window surface. Since the cited rows contain no open-cover compactness, host minimum, choice, quotient equality, selected limit, or Real-completeness coordinate, none is available at the root. □

Theorem 452.81 (Finite Lebesgue-number radius carrier source)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *root radius-window ledger projects its carrier source exactly to the finite* BHist *rows*

$$D, W, \lambda, M, S, Q, R, H, C, P, N$$

of Definition 452.79. The only source faces visible in this projection are DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*, together with the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No host minimum, choice coordinate, quotient equality, selected limit, or Real-completeness coordinate is part of the carrier source.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusCarrierSource`

Proof. Project the carrier through the ledger definition. Definition 452.79 lists the finite rows $D, W, \lambda, M, S, Q, R, H, C, P, N$ and names the dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, Real, transport, continuation, provenance, and local naming faces.

Now use the root radius-window obligation surface. Definition 452.80 states that an accepted ledger discharges the positive-radius and finite-window obligations only through those displayed rows.

The L10 source faces are therefore read as the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ part of the same finite packet, while DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, and DyadicMesh$^{\uparrow}$ supply the local radius carrier. Since neither the definition nor the surface theorem contains a row for a host minimum, choice, quotient equality, selected limit, or Real completeness, no such coordinate can appear in the projected carrier source. □

Theorem 452.82 (Finite Lebesgue-number radius classifier stability)**.** *Let two accepted* FiniteLebesgueNumber$^{\uparrow}$ *root radius-window ledgers agree componentwise by* hsame *on the rows* $D, W, \lambda, M, S, Q, R, H, C, P, N$*. Then the positive-radius window transported between them has the same classifier at every* CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, and* UniformModulus$^{\uparrow}$ *endpoint, and the transport does not change the L10 source faces* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusClassifierStability`

Proof. Start with the componentwise agreement. Definition 452.84 transports the accepted radius-window obligation along the shared rows $D, W, \lambda, M, S, Q, R, H, C, P, N$ and keeps the consumer-facing surface finite.

Next apply open Phase exhaustion. Definition 452.86 says that the positive-radius and finite-window reads are exhausted by the root radius-window packet before a compact, compact-net, continuous-map, or uniform-modulus endpoint can add a further coordinate.

Finally use RegSeq-StreamName nonescape. Definition 452.88 keeps the stream-name and regular-rational source faces attached to the four-face route, so the transported classifier cannot replace the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ faces. The classifier is stable because every visible row is carried by the same componentwise hsame transport. □

Theorem 452.83 (Finite Lebesgue-number radius ledger exactness)**.** *The local* NameCert *ledger for the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root radius-window obligation is exactly the carrier-source and classifier-stability package of Definition 452.81 and Definition 452.82. It has no additional compactness, choice, selected-limit, quotient, metric-completeness, or Real-completeness coordinate beyond the rows of Definition 452.79.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRadiusLedgerExactness`

Proof. First read the carrier-source side. Definition 452.81 identifies the carrier projection with the finite rows $D, W, \lambda, M, S, Q, R, H, C, P, N$ and excludes off-ledger minima, choice, quotient equality, selected limits, and Real completeness.

Next read the classifier side. Definition 452.82 transports the positive-radius classifier through componentwise hsame while keeping the L10 source faces fixed.

Now apply consumer exactness and Real-source coverage. Definition 452.85 states that every downstream compact, compact-net, continuous-map, and uniform-modulus consumer receives exactly the ledger rows, while Definition 452.87 accounts for the Real-side source faces inside the same finite packet.

These two readings exhaust the local NameCert ledger: the carrier source gives the rows, classifier stability gives their transported public read, and the consumer and Real-source theorems rule out any extra compactness, choice, selected-limit, quotient, metric-completeness, or Real-completeness coordinate. □

Theorem 452.84 (Finite Lebesgue-number root radius-window transport)**.** *Let two accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *routes agree on the rows* $D, W, \lambda, M, S, Q, R, H, C, P, N$ *of the root radius-window obligation ledger. Then the radius-window obligation transported to* $\mathsf{CompactMetric}^{\uparrow}$, CompactNetModulusS $\mathsf{ContinuousMap}^{\uparrow}$, *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumers is the same finite obligation surface in both routes. The transport uses only* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and* NameCert *rows over* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusWindowTransport certificate`

Proof. First read the shared ledger rows. Agreement on $D, W, \lambda, M, S, Q, R$ fixes the dyadic cover, finite window, positive radius, mesh, stream-name, regular-rational, and Real endpoint data.

Use the NameCert obligations to transport componentwise. Definition 452.2 keeps the carrier rows under hsame, and Definition 452.5 keeps the window witness attached to the same stream, regular-rational, Real, mesh, and cover route.

Next use the L10 and Phase Real source locks. Definition 452.13 identifies the only source of the radius as the finite dyadic Real-phase route, and Definition 452.75 makes that route visible to compact and continuous consumers without adding an outside coordinate.

The remaining rows H, C, P, N replay and name the same finite packet for both routes. Therefore the transported consumer obligation is identical, because no open-cover compactness, host minimum, choice, quotient equality, selected limit, or Real-completeness coordinate can distinguish the two routes. □

Theorem 452.85 (Finite Lebesgue-number root radius-window consumer exactness)**.** *Every* CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *consumer admitted downstream of the root radius-window obligation ledger receives exactly the finite rows of Definition 452.79. The consumer receives no additional open-cover compactness, host minimum, choice, quotient equality, selected limit, or Real-completeness coordinate beyond the* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *packet.*

Proof. Start with the root obligation surface theorem above. It states that the positive radius and finite-window obligations are already exhausted by $D, W, \lambda, M, S, Q, R, H, C, P, N$.

Apply the Phase Real source lock. Definition 452.75 makes every consumer-facing radius read pass through DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before any compact, compact-net, continuous-map, or uniform-modulus endpoint can inspect it.

Use the L10 source lock and window coverage exactness to cover the finite-window side. Definition 452.13 fixes the source route, while Definition 452.5 sends each accepted finite-window read through the named stream, regular-rational, Real, mesh, and cover rows.

Finally read consumer exactness from the absence of outside rows in the ledger. Since Definition 452.79 lists only the displayed finite packet, each downstream consumer receives exactly that packet and no compactness, equality, choice, selected-limit, or completeness coordinate outside it. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusWindowConsumerExactness`

Theorem 452.86 (Finite Lebesgue-number open-phase radius-window exhaustion)**.** *For every accepted* FiniteLebesgueNumber$^{\uparrow}$ *route exposed to the current Real-and-Completion four-face source route, the open-phase radius-window ledger is exhausted by the displayed rows*

$$\text{DyadicRatCore}^{\uparrow},\ \text{StreamName}^{\uparrow},\ \text{RegSeqRat}^{\uparrow},\ \text{Real}^{\uparrow},\ \text{BHist},\ \text{hsame},\ \text{Cont},\ \text{Pkg},\ \text{NameCert}.$$

Equivalently, every positive-radius window visible from the root obligation surface is read through the dyadic-rational core, stream-name source, regular-rational readback, Real endpoint, history transport, continuation, package provenance, and local NameCert rows. It has no coordinate for open-cover compactness, host minima, quotient equality, selected limits, choice, detached Real completeness, or a second radius source.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRadiusWindowLedgerExhaustion`

Proof. Begin with the root radius-window obligation surface. Definition 452.80 states that the positive radius and finite-window obligations are already carried by the finite dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Next attach the Phase Real source lock. Definition 452.75 forces each compact- or continuous-visible public radius read through DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before any downstream endpoint can inspect it.

Finally use the Real-completion terminal-radius exhaustion. Definition 452.95 says that the terminal compact, compact-net, continuous-map, and uniform-modulus radius reads still factor through the same L10 packet and introduce no outside compactness, equality, selected-limit, choice, or completeness coordinate. Thus the open-phase radius-window ledger is exactly the displayed four-face source route with the BEDC support rows. □

Theorem 452.87 (Finite Lebesgue-number Real-source ledger coverage)**.** *The finite Lebesgue-number Real-source ledger covers every source row used by the current radius-window packet:*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

are the only source faces, and BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *are the only BEDC transport and naming rows that carry those faces to the accepted* FiniteLebesgueNumber$^{\uparrow}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealSourceLedgerCoverage`

Proof. Read the root ledger first. Definition 452.79 lists the dyadic cover, finite window, positive radius, dyadic mesh, stream-name, regular-rational, Real endpoint, history, transport, continuation, provenance, and local naming rows, with no independent compactness or completeness coordinate.

Use the Phase Real source lock to identify the source faces. Definition 452.75 routes the consumer-facing radius through the dyadic, stream-name, regular-rational, and Real rows, and it leaves only BHist, hsame, Cont, Pkg, and NameCert as the BEDC support carrying the public packet.

The preceding open-phase exhaustion theorem then removes any remaining escape route at the radius-window boundary. Therefore the Real-source ledger covers exactly the displayed four source faces and the named support rows. □

Theorem 452.88 (Finite Lebesgue-number RegSeq-StreamName nonescape)**.** *In an accepted* FiniteLebesgueNumber$^{\uparrow}$ *open-phase radius-window ledger, the* RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *rows cannot be bypassed by a host sequence, quotient stream equality, selected limit, choice-selected representative, or ambient completeness witness. Every downstream read of the radius window that reaches* CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$ *must pass through the displayed* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *rows together with* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRegSeqStreamNameNonescape`

Proof. Start from Real-source ledger coverage. Definition 452.87 identifies StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ as source faces of the finite radius-window packet, not as optional annotations.

Now apply the root radius-window consumer exactness theorem. Definition 452.85 says that compact, compact-net, continuous-map, and uniform-modulus consumers receive exactly the displayed finite rows and no host minimum, quotient equality, selected limit, choice, open-cover compactness, or Real-completeness coordinate.

Finally attach the Real-completion terminal-radius exhaustion. Definition 452.95 keeps terminal radius reads on the same L10 packet. Hence bypassing StreamName$^{\uparrow}$ or RegSeqRat$^{\uparrow}$ would require a coordinate absent from both the root consumer boundary and the terminal Real-completion ledger, so the stated nonescape follows. □

The root radius ledger is read against the four-face Phase Real route through chapter 291, chapter 126, chapter 301, and chapter 38.

Theorem 452.89 (Finite Lebesgue-number open Phase source nonescape)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier, no public source read visible at the root can inspect host* Real$^{\uparrow}$ *equality, open-cover compactness, choice-selected subcovers, or external modulus data before the read has passed through the finite four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The only admissible source coordinates before that passage are the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows tied to the dyadic cover, rational-ball radius, dyadic mesh, stream name, regular rational, and Real endpoint rows of the* FiniteLebesgueNumber$^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseSourceNonescape`

Proof. Begin with the finite carrier. Definition 452.79 lists the root radius-window rows as a finite packet over DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Apply the source locks. Definition 452.13 forces the radius source through the stream, regular-rational, Real, mesh, and cover rows, while Definition 452.75 exposes the same route to compact and continuous consumers without adding an outside coordinate.

Now use root radius-window exactness. Definition 452.80 excludes open-cover compactness, host minima, choice coordinates, quotient equality, selected limits, and detached Real-completeness data from the root obligation surface.

Therefore any accepted public source read must pass through the four displayed Phase Real faces before a downstream consumer can inspect it. Since no host equality, open-cover compactness, choice-selected subcover, or external modulus coordinate appears in the finite packet, none can be read at the root. □

Theorem 452.90 (Finite Lebesgue-number four-face radius readback)**.** *Every* CompactMetric$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$ *radius read admitted downstream of* FiniteLebesgueNumber$^{\uparrow}$ *factors through the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the DyadicCover$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*, and* RationalBall$^{\uparrow}$ *rows of the finite Lebesgue-number carrier. Thus compact and compact-net consumers can read only the displayed finite radius ledger, not an open-cover compactness principle, host minimum, arbitrary subcover, selected limit, or external modulus selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberFourFaceRadiusReadback`

Proof. First read the public export. Definition 452.49 identifies the consumer-facing export as the finite cover, rational-ball, mesh, source, transport, replay, provenance, and naming route.

Next apply the Phase Real coverage export. Definition 452.78 routes compact, compact-net, continuous-map, and uniform-modulus endpoints through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$, together with the finite carrier rows.

Use the root radius-window consumer exactness theorem for the compact side. Definition 452.85 says that the downstream consumers receive exactly the displayed finite rows and no open-cover compactness, host minimum, choice, quotient equality, selected limit, or Real-completeness coordinate.

The remaining rows DyadicCover$^{\uparrow}$, DyadicMesh$^{\uparrow}$, and RationalBall$^{\uparrow}$ are precisely the cover, mesh, and radius rows named in the carrier. Hence every radius read factors through the four L10 faces plus those finite dyadic rows, with no separate compactness or selector interface. □

Theorem 452.91 (Finite Lebesgue-number root-unblock exit triad)**.** *Definition 452.89, Definition 452.90, and Definition 452.55 form the root-unblock exit surface for every compact-uniform consumer of* FiniteLebesgueNumber$^{\uparrow}$*. The exit surface supplies only the finite four-face Phase Real route, the dyadic cover, dyadic mesh, rational-ball radius,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it supplies no host Real equality, open-cover compactness, choice-selected subcover, selected limit, detached completeness primitive, or external modulus data.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootUnblockExitTriad`

Proof. Start with the source boundary. Definition 452.89 prevents a root read from reaching host Real equality, open-cover compactness, selected subcovers, or external modulus data before it passes through the four-face Phase Real route.

Now read the radius boundary. Definition 452.90 sends every compact or compact-net radius read through the four L10 faces and the displayed dyadic cover, dyadic mesh, and rational-ball rows.

Finally apply the existing consumer boundary. Definition 452.55 excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and external modulus data from the compact, compact-net, and uniform consumer endpoints.

Combining the three rows gives exactly the root-unblock exit surface. All exported data is already present in the finite carrier and its NameCert rows, so downstream compact-uniform consumers cannot recover any outside compactness, equality, choice, completeness, selected-limit, or modulus primitive from the handoff. □

Theorem 452.92 (Finite Lebesgue-number L10 phase consumer readiness)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *consumer route used at the Phase Real or Completion boundary factors through the finite packet over* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Continuous-map and totally-bounded consumers read the same route through the displayed compact-uniform handoff. The route exports no open-cover compactness, selected subcover, host minimum, quotient equality, choice, external modulus, or* Real$^{\uparrow}$ *completeness.*

Proof. Begin with the compact-uniform export. Definition 452.49 identifies the public handoff as the finite dyadic-cover, rational-ball, mesh, stream, regular-rational, Real, transport, replay, provenance, naming, and compact-net selector packet.

Next attach the L10 source chain. Definition 452.52 sends each finite-window read through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before mesh and cover-cell readback, and Definition 452.53 places the displayed dyadic rows in the same BHist, hsame, Cont, Pkg, and NameCert packet.

Use ledger transport to keep the route stable. Definition 452.54 preserves the positive radius ledger together with the cover, mesh, stream, regular-rational, and Real rows under hsame transport, so later compact, continuous-map, totally-bounded, uniform-modulus, or Completion-facing consumers cannot detach a scalar or host-side witness from the packet.

Finally apply the nonescape boundary. Definition 452.55 excludes open-cover compactness, host minima, selected subcovers, quotient equality, choice, and external modulus data from the compact, compact-net, and uniform consumer endpoints. Since the Phase Real and Completion handoffs read through those endpoints and the same L10 source packet, their consumer-ready route is exactly the displayed finite packet and not an additional compactness, choice, modulus, or completeness principle. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberL10PhaseConsumerReadiness`

Theorem 452.93 (Finite Lebesgue-number L10 consumer readback exactness)**.** *Every Phase Real or Completion boundary consumer of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route reads exactly the packet over*

DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg

The compact, compact-net, continuous-map, totally-bounded, and uniform-modulus endpoints add no open-cover compactness, selected subcover, host minimum, quotient equality, choice, external modulus, selected limit, or detached Real-completeness row to that consumer-visible readback.

Proof. First identify the endpoint factorization. Definition 452.92 places every Phase Real or Completion boundary consumer on the finite route through the dyadic cover, rational-ball, dyadic mesh, stream, regular-rational, Real, compact, compact-net, uniform, and structural rows, with no outside compactness, equality, choice, modulus, or completeness coordinate.

Next read the source packet through the open Phase classifier. Definition 452.97 identifies the accepted positive-radius read with the four-face route

DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$

together with the DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally assemble the terminal package. Definition 452.99 states that the open Phase pullback, compact-uniform readiness, Real-completion terminal-radius exhaustion, and L10 phase consumer readiness are one certificate over exactly the displayed packet. Since that terminal certificate has no coordinate for open-cover compactness, selected subcovers, host minima, quotient equality, choice, external moduli, selected limits, or detached Real completeness, the consumer-visible readback is exactly the finite packet named in the statement. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberL10ConsumerReadbackExactness`

Definition 452.94 (Finite Lebesgue-number Real-completion terminal-radius ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route, the Real-completion terminal-radius ledger is the finite* BHist *packet*

$$\Theta_{\lambda}^{\mathrm{term}} = (D, W, \lambda, M, S, Q, R, K, U, H, C, P, N)$$

whose entries are the DyadicCover$^{\uparrow}$ *cover row D, the* FiniteWindowEnvelope$^{\uparrow}$ *row W, the positive* RationalBall$^{\uparrow}$ *radius row λ, the* DyadicMesh$^{\uparrow}$ *row M, the* StreamName$^{\uparrow}$ *source row S, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *endpoint row R, the compact and compact-net consumer row K, the continuous-map and uniform-modulus consumer row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N.*

The ledger is terminal for the L10 Real-and-Completion source route: a downstream radius consumer may inspect only the displayed DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows, with* CompactNetModulusSelector$^{\uparrow}$ *read through the compact row K. It has no coordinate for open-cover compactness, a host minimum, quotient equality, selected limits, choice, or an ambient Real-completeness primitive.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealCompletionTerminalRadiusLed`

Theorem 452.95 (Finite Lebesgue-number Real-completion terminal-radius exhaustion)**.** *Every* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *consumer*

admitted downstream of an accepted Real-completion terminal-radius ledger of Definition 452.94 factors its positive-radius read through the displayed L10 packet

$$D, W, \lambda, M, S, Q, R, K, U, H, C, P, N.$$

Consequently the compact, compact-net, continuous-map, and uniform-modulus endpoints cannot read open-cover compactness, host minima, quotient equality, selected limits, choice, or Real completeness outside the $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *route.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealCompletionTerminalRadiusExhaustion`
Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalRadiusExhaustion`

Proof. Begin with the finite-window side. Definition 452.5 routes every accepted window read through the finite window W, a $\mathsf{StreamName}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ row, a mesh cell of M, and a named cover cell of D. Thus the window coordinate used by a downstream radius consumer is already inside the displayed terminal ledger.

Next attach the L10 source route. Definition 452.13 forces the positive-radius read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, and $\mathsf{DyadicCover}^{\uparrow}$, while Definition 452.75 exposes the same route to compact and continuous consumers without adding a host scalar or open-cover compactness witness.

Now read the root consumer boundary. Definition 452.85 states that $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelect}$ $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers receive exactly the root rows $D, W, \lambda, M, S, Q, R, H, C, P, N$ and no open-cover compactness, host minimum, quotient equality, selected limit, choice, or Real-completeness coordinate.

Finally use the L10 phase consumer readiness theorem. Definition 452.92 places the compact, compact-net, continuous-map, and uniform-modulus endpoints on the same finite packet, so the consumer rows K and U do not introduce another radius source. Therefore every terminal positive-radius read factors through $\Theta_{\lambda}^{\mathrm{term}}$, and the excluded compactness, equality, selected-limit, choice, and completeness data remain unavailable. □

Theorem 452.96 (Finite Lebesgue-number terminal-radius choice refusal)**.** *Let a downstream* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer preserve the Real-completion terminal-radius ledger*

$$\Theta_{\lambda}^{\mathrm{term}} = (D, W, \lambda, M, S, Q, R, K, U, H, C, P, N)$$

of Definition 452.94. Then the displayed rational-ball radius λ *cannot be replaced, at that terminal boundary, by a host minimum, a selected subcover radius, a choice-selected modulus, quotient* $\mathsf{Real}^{\uparrow}$ *equality, a selected limit, or an external compactness witness.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalRadiusChoiceRefusal`

Proof. The ledger definition fixes the available rows exactly. By Definition 452.94, the terminal packet exposes only $D, W, \lambda, M, S, Q, R, K, U, H, C, P, N$, with K and U serving as consumer rows and with no coordinate for open-cover compactness, a host minimum, quotient equality, selected limits, choice, or an ambient Real-completeness primitive.

Now apply the exhaustion theorem. Definition 452.95 says that every positive-radius read by the compact, compact-net, continuous-map, or uniform-modulus endpoints factors through this displayed packet. Thus a replacement for λ would have to be readable from one of the displayed rows.

Finally use the root consumer boundary cited in that proof. Definition 452.85 excludes open-cover compactness, host minima, quotient equality, selected limits, choice, and Real-completeness coordinates from exactly these root consumers. The proposed host minimum, selected subcover radius, choice-selected modulus, quotient $\mathsf{Real}^{\uparrow}$ equality, selected limit, and external compactness witness therefore have no terminal ledger coordinate through which they can replace λ. □

Theorem 452.97 (Finite Lebesgue-number open Phase radius classifier pullback)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius read visible at the open Phase boundary, the radius classifier factors through exactly the* $\mathsf{DyadicCover}^{\uparrow}$ *row, the* $\mathsf{RationalBall}^{\uparrow}$ *radius row, the* $\mathsf{DyadicMesh}^{\uparrow}$ *row, and the four-face source route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The only structural support available to the read is the displayed BHist *packet together with componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *naming. In particular the pullback has no coordinate for open-cover compactness, a selected subcover, a host minimum, quotient equality, choice, an external modulus, or a detached Real-completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRadiusClassifierPullback`

Proof. Start with the Phase Real source lock. Definition 452.75 sends every compact- or continuous-visible radius read through the dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, and Real rows, while preserving only BHist, hsame, Cont, Pkg, and NameCert support.

Next use the four-face readback. Definition 452.90 identifies the same radius read with the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. Thus the classifier cannot acquire a second source after it crosses the open Phase boundary.

Finally apply the root consumer boundary and the nonescape triad. Definition 452.85 fixes the downstream rows available to compact, compact-net, continuous-map, and uniform-modulus consumers, and Definition 452.55 removes open-cover compactness, selected subcovers, host minima, quotient equality, choice, and external modulus data from those endpoints. The stated pullback is therefore exactly the displayed finite classifier route. □

Theorem 452.98 (Finite Lebesgue-number compact-uniform four-face readiness)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CompactNetModulusSelector}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer downstream of the accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius ledger reads the same four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

together with the $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The compact-uniform handoff supplies no open-cover compactness, selected subcover, host minimum, quotient equality, choice, external modulus, or Real-completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformFourFaceReadiness`

Proof. First read the radius classifier through the open Phase pullback. Definition 452.97 gives the exact four-face source route and the dyadic cover, rational-ball, and dyadic-mesh rows available to any accepted positive-radius read.

Now attach the compact and uniform consumers. Definition 452.85 states that $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers receive exactly

the root finite rows and no outside compactness, equality, selected-limit, choice, or completeness coordinate.

Finally use the nonescape boundary. Definition 452.55 excludes open-cover compactness, selected subcovers, host minima, quotient equality, choice, and external modulus data from the compact, compact-net, and uniform endpoints. Therefore the compact-uniform route is ready precisely over the four-face packet and not over an additional compactness or completeness principle. □

Theorem 452.99 (Finite Lebesgue-number root-unblock terminal package)**.** *The accepted* FiniteLebesgueNumber$^{\uparrow}$ *root radius ledger is terminal for the current four-face route: the open Phase radius classifier pullback of Definition 452.97, the compact-uniform readiness of Definition 452.98, the Real-completion terminal-radius exhaustion of Definition 452.95, and the L10 phase consumer readiness of Definition 452.92 form one certificate over*

DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg

No downstream CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$ *consumer in this terminal certificate can import open-cover compactness, a selected subcover, a host minimum, quotient equality, choice, an external modulus, a selected limit, or a detached Real-completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootUnblockTerminalPackage`

Proof. Begin with the two classifier surfaces. Definition 452.97 fixes the accepted radius read to the dyadic cover, rational-ball, dyadic-mesh, and four-face source rows, and Definition 452.98 makes the compact, compact-net, and uniform consumers read that same packet.

Next attach the terminal Real-completion route. Definition 452.95 says that every compact, compact-net, continuous-map, or uniform-modulus positive-radius read factors through the terminal L10 packet and has no coordinate for open-cover compactness, host minima, quotient equality, selected limits, choice, or Real completeness.

Finally use L10 phase consumer readiness. Definition 452.92 places the compact, compact-net, continuous-map, and uniform-modulus endpoints on the same finite route, so the classifier and terminal-radius rows are not separate certificates. The combined root ledger is therefore the stated terminal certificate, with all listed outside compactness, equality, choice, modulus, selected-limit, and completeness data excluded. □

Definition 452.100 (Finite Lebesgue-number compact radius-window pullback packet)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier, the compact radius-window pullback packet is the finite* BHist *row*

$$\Omega_\lambda = (B, S, Q, R, K, F, U, H, C, P, N)$$

whose entries are the DyadicRatCore$^{\uparrow}$ *radius row B, the* StreamName$^{\uparrow}$ *finite-window row S, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *seal face R, the* CompactMetric$^{\uparrow}$ *and* ContinuousMap$^{\uparrow}$ *consumer row K, F, the* UniformModulus$^{\uparrow}$ *row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N. The packet is accepted only when B, S, Q, R, K, F, U are read as one radius-window obligation through the same H, C, P, N support, with no open-cover compactness, selected limit, quotient stream equality, selected subcover, host minimum, choice-like cover selection, or unbudgeted radius selector.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactRadiusWindowPullbackPack`

Theorem 452.101 (Finite Lebesgue-number compact radius-window pullback)**.** *Every compact or continuous consumer admitted downstream of an accepted compact radius-window pullback packet* Ω_λ *receives the positive radius through exactly the displayed* $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{CompactMetric}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, $\mathsf{UniformModulus}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 452.100. The pullback supplies a finite radius-window obligation for* $\mathsf{CompactMetric}^\uparrow$ *and* $\mathsf{ContinuousMap}^\uparrow$ *consumers; it does not supply open-cover compactness, selected limits, quotient stream equality, selected subcovers, host minima, choice-like cover selection, or unbudgeted radius selection.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactRadiusWindowPullback`

Proof. Begin with the open Phase radius classifier pullback. Definition 452.97 fixes every accepted positive-radius read to the dyadic radius, finite-window, regular-rational readback, and Real source faces before any compact or continuous endpoint can inspect the radius.

Next use the L10 sibling route. Definition 452.70 forces compact, compact-net, continuous-map, and uniform-modulus reads through the finite-name chain and then through the dyadic mesh and cover surfaces. The row B, S, Q, R of Ω_λ is therefore the same finite source route, not a second compactness witness.

Attach the compact-continuous unblock triad. Definition 452.67 and Definition 452.69 make the $\mathsf{CompactMetric}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, and $\mathsf{UniformModulus}^\uparrow$ rows consume the shared finite dyadic-radius packet through the same BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply root-radius coherence. Definition 452.115 prevents the consumer from distinguishing two routes whose displayed radius, stream, regular-rational, Real, transport, replay, provenance, and naming rows agree. Since none of the cited surfaces contains a coordinate for open-cover compactness, selected limits, quotient stream equality, selected subcovers, host minima, choice-like cover selection, or unbudgeted radius selection, the accepted packet is exactly the stated compact radius-window pullback. □

Theorem 452.102 (Finite Lebesgue-number uniform-modulus consumer pullback)**.** *Every* $\mathsf{UniformModulus}^\uparrow$ *row admitted by* Ω_λ *is pulled back through the same compact radius-window packet read by the* $\mathsf{CompactMetric}^\uparrow$ *and* $\mathsf{ContinuousMap}^\uparrow$ *rows. Thus the uniform consumer receives the radius only as the finite* BHist *packet over* $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming; it cannot replace that packet by an external modulus, an open-cover compactness argument, a selected limit, quotient stream equality, a selected subcover, a host minimum, choice-like cover selection, or unbudgeted radius selection.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformModulusConsumerPullback`

Proof. First read the compact-continuous triad. Definition 452.67 identifies the compact, compact-net, continuous-map, and uniform-modulus rows as consumers of one finite dyadic-radius packet.

Now apply the compact radius-window pullback theorem above. It places the $\mathsf{CompactMetric}^\uparrow$ and $\mathsf{ContinuousMap}^\uparrow$ reads on Ω_λ, whose source faces are the $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{Real}^\uparrow$ rows and whose support is exactly H, C, P, N.

Use the root radius-window consumer exactness and root-radius coherence package to rule out a detached uniform source. Definition 452.85 states that every compact, continuous, or uniform endpoint receives only the finite root rows, and Definition 452.116 reads those rows through the coherent root-radius ledger.

The uniform-modulus row therefore has no remaining coordinate for an external modulus, open-cover compactness, selected limits, quotient stream equality, selected subcovers, host minima,

choice-like cover selection, or unbudgeted radius selection. Its only admissible source is the compact radius-window packet. □

Theorem 452.103 (Finite Lebesgue-number budgeted Real-completion nonescape)**.** *The Real-completion route of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier cannot turn a compact radius-window pullback packet into an unbudgeted radius selector. Any terminal* Real$^{\uparrow}$ *or completion-facing radius read remains the finite packet* Ω_λ *transported by* hsame *and replayed by* Cont *through the displayed* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, Pkg, *and* NameCert *rows. In particular the Real-completion endpoint has no coordinate for open-cover compactness, selected limits, quotient stream equality, selected subcovers, host minima, choice-like cover selection, or unbudgeted radius selection.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberBudgetedRealCompletionNonescape`

Proof. Start with the existing Real-completion radius nonescape statement. Definition 452.117 says that the terminal Phase Real or Completion radius read remains the coherent finite root-radius packet, transported by hsame and replayed by Cont through the displayed source faces.

Next use the open Phase radius classifier pullback. Definition 452.97 identifies the radius classifier before the compact or continuous consumer sees it, so the Real-completion endpoint cannot add a new selected-limit or quotient-stream coordinate after the classifier has crossed the open Phase boundary.

Attach the compact radius-window pullback and uniform-modulus consumer pullback. The first theorem in this file places the compact and continuous consumers on Ω_λ, and the uniform theorem above makes the UniformModulus$^{\uparrow}$ row consume the same packet. Thus the Real-completion route and the compact-uniform route share the same budgeted radius-window obligation.

Finally apply root-radius coherence. Definition 452.115 makes the displayed radius, stream, regular-rational, Real, transport, replay, provenance, and naming rows determine the consumer-facing obligation. Since those rows contain no open-cover compactness, selected limits, quotient stream equality, selected subcovers, host minima, choice-like cover selection, or unbudgeted radius selection, the budgeted packet is nonescaping at the Real-completion endpoint. □

Definition 452.104 (Finite Lebesgue-number Phase Real radius audit packet)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route, the Phase Real radius audit packet is the finite* BHist *row*

$$\mathcal{A}_\lambda^{\mathrm{PhaseReal}} = (D, \rho, M, K, S, Q, R, H, C, P, N)$$

whose entries are the displayed positive radius ledger ρ*, the* DyadicCover$^{\uparrow}$ *cover row* D*, the* RationalBall$^{\uparrow}$ *radius row carried by* ρ*, the* DyadicMesh$^{\uparrow}$ *mesh row* M*, the* DyadicRatCore$^{\uparrow}$ *core row* K*, the* StreamName$^{\uparrow}$ *stream row* S*, the* RegSeqRat$^{\uparrow}$ *regular-rational row* Q*, the* Real$^{\uparrow}$ *endpoint row* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *naming* N*. The packet audits only these rows and contains no open-cover compactness, selected subcover, quotient equality, choice, external modulus, selected limit, or detached Real-completeness coordinate.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusAuditPacket`
Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusAuditPacket.certificate`

Theorem 452.105 (Finite Lebesgue-number Phase Real radius audit carrier)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *root radius read visible to the current Real and Completion route carries a*

unique audit packet of Definition 452.104. The carrier consists exactly of the positive radius ledger together with

$$\mathsf{DyadicCover}^{\uparrow}, \mathsf{RationalBall}^{\uparrow}, \mathsf{DyadicMesh}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}$$

No additional compactness, choice, quotient-equality, external-modulus, selected-limit, or Real-completeness row is part of the carrier.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusAuditCarrier`

Proof. Begin with the root radius-window obligation surface. Definition 452.80 displays the positive radius and finite-window obligations as finite dyadic cover, rational-ball, mesh, stream-name, regular-rational, Real, history, transport, continuation, provenance, and naming rows.

Next apply the open Phase radius classifier pullback. Definition 452.97 identifies the same radius read with the $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, and $\mathsf{DyadicMesh}^{\uparrow}$ rows and with the four-face route through chapter 291, chapter 126, chapter 301, and chapter 38.

The displayed rows are precisely the entries of $\mathcal{A}_{\lambda}^{\mathrm{PhaseReal}}$. Since the cited root and pullback theorems leave no row for open-cover compactness, subcover selection, quotient equality, choice, external modulus data, selected limits, or detached Real completeness, the carrier is exactly the stated audit packet. □

Theorem 452.106 (Finite Lebesgue-number Phase Real radius audit classifier)**.** *The classifier of a Phase Real radius audit packet is determined by its displayed positive radius ledger and by the ordered source route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Thus two accepted audit packets with the same $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*, four-face source route,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows expose the same radius classifier to every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CompactNetModulusSelector}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *endpoint.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusAuditClassifier`

Proof. Read the carrier theorem first. Definition 452.105 reduces every visible root radius read to the finite audit packet and names the four-face source route.

Now use the root radius-window consumer exactness theorem. Definition 452.85 states that compact, compact-net, continuous-map, and uniform-modulus endpoints receive exactly the dyadic cover, rational-ball, mesh, stream-name, regular-rational, Real, history, transport, continuation, provenance, and naming rows.

Finally apply the compact-uniform four-face readiness theorem. Definition 452.98 makes those endpoints read the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source faces. Therefore equality of the displayed audit rows determines the downstream classifier and leaves no second classifier coordinate. □

Theorem 452.107 (Finite Lebesgue-number Phase Real radius audit nonescape)**.** *No accepted Phase Real radius audit packet can be extended, without leaving the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root-unblock route, by open-cover compactness, a selected subcover, quotient equality, choice, an external modulus, selected limits, or a detached Real-completeness row. Every downstream radius read remains inside the packet of Definition 452.104.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusAuditNonescape`

Proof. Start with the classifier theorem above. It shows that all downstream compact, compact-net, continuous-map, and uniform-modulus reads are already determined by the displayed audit rows.

Attach the Real-completion terminal-radius exhaustion. Definition 452.95 keeps every terminal radius read on the L10 packet and excludes open-cover compactness, host minima, quotient equality, selected limits, choice, and Real completeness from the terminal route.

Then use the open Phase source nonescape and RegSeq-StreamName nonescape boundaries. Definition 452.89 and Definition 452.88 prevent bypassing the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ source rows through host sequences, quotient stream equality, selected representatives, choice, or ambient completeness.

Any proposed extension by one of the excluded rows would therefore have to pass through a coordinate absent from the carrier, the classifier, and the terminal Real-completion route. The extension leaves the accepted root-unblock packet, so the audit surface is nonescaping. □

Theorem 452.108 (Phase Real four-face source exactness)**.** *Every consumer-visible* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius read in the accepted root route factors through the current Phase Real and Completion faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The exact source packet exports no open-cover compactness, selected subcover, quotient equality, host minimum, countable choice, Real completeness, or external modulus coordinate.*

Proof. Begin with the L10 source lock. Definition 452.13 forces the radius route through the finite stream, regular-rational, Real, dyadic-mesh, and dyadic-cover rows before any consumer can inspect the positive-radius ledger.

Next apply the open Phase classifier. Definition 452.97 identifies the visible positive-radius classifier with the four-face route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, carried by the same finite dyadic cover, rational-ball, and mesh packet.

Finally attach the Phase Real radius audit carrier. Definition 452.105 records the BHist, hsame, Cont, Pkg, and NameCert support rows and excludes open-cover compactness, choice, quotient equality, external modulus data, selected limits, and detached Real-completeness rows. Since the accepted consumer read has no row outside these cited packets, the stated four-face source exactness follows. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealFourFaceSourceExactness`

Theorem 452.109 (Terminal-radius Phase consumer nonescape)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* CompactNetModulusSel
$\mathsf{ContinuousMap}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer that reads a terminal* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius remains inside the four-face Phase Real and Completion packet*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Such a terminal consumer cannot escape to open-cover compactness, a selected subcover, quotient equality, a host minimum, countable choice, Real completeness, or an external modulus coordinate.

Proof. Start from terminal-radius exhaustion. Definition 452.95 makes every compact, compact-net, continuous-map, or uniform-modulus terminal read factor through the displayed L10 packet and denies open-cover compactness, host minima, quotient equality, selected limits, choice, and Real completeness at that boundary.

Then use the four-face exactness theorem above. It identifies the visible source of the same radius with DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ plus the BEDC support rows BHist, hsame, Cont, Pkg, and NameCert.

The compact-uniform source triangle, Definition 452.111, places these source faces before the compact, continuous-map, and uniform-modulus endpoints. Any terminal escape would therefore require a coordinate absent from the terminal ledger and absent from the four-face source packet, so no such consumer can leave the stated route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalRadiusPhaseConsumerNonescape`

Theorem 452.110 (Phase Real root-unblock package)**.** *The* FiniteLebesgueNumber$^{\uparrow}$ *root-unblock surface for downstream compact and uniform consumers is the single finite package consisting of the four Phase Real and Completion faces* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$ *and the support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. This package is source-exact: every consumer-visible radius read factors through it, and it exports no open-cover compactness, selected subcover, quotient equality, host minimum, countable choice, Real completeness, or external modulus coordinate.*

Proof. First combine source exactness with terminal nonescape. Definition 452.108 gives the exact source packet for any consumer-visible radius read, and Definition 452.109 keeps terminal compact, compact-net, continuous-map, and uniform-modulus consumers inside the same packet.

Next use the compact-uniform source triangle. Definition 452.111 orders the four source faces before the consumer endpoints and ties them to the visible dyadic cover, rational-ball, mesh, history, transport, continuation, provenance, and naming rows.

The open Phase radius classifier pullback and Real-completion terminal-radius exhaustion supply the two boundary checks: Definition 452.97 fixes the classifier before the consumer endpoint can read it, while Definition 452.95 rules out the off-packet compactness, equality, choice, completeness, and modulus coordinates at the terminal boundary. The root-unblock surface is therefore exactly the stated finite package. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRootUnblockPackage`

Theorem 452.111 (Compact-uniform source triangle)**.** *Every accepted compact, continuous-map, or uniform-modulus consumer of a* FiniteLebesgueNumber$^{\uparrow}$ *positive-radius row factors through exactly the displayed Phase Real source triangle*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

before the route reaches CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$*. The four source faces are read through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows tied to the visible* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*, and* DyadicMesh$^{\uparrow}$ *packet, and no accepted downstream consumer may read a detached radius before that triangle has been consumed.*

Proof. Start at the open Phase boundary. Definition 452.89 keeps every public source read inside the four-face route, and Definition 452.97 identifies the positive-radius classifier with the same DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ source packet.

Next pass through the sibling route. Definition 452.70 supplies the finite-name chain that carries the radius through BHist, hsame, Cont, Pkg, and NameCert rows, while Definition 452.69 makes the compact, continuous-map, and uniform-modulus consumers read that carried packet rather than an independent consumer datum.

The source chapters chapter 291, chapter 126, chapter 301, and chapter 38 give the four faces their public NameCert readings. Combining those readings with the cited finite-lebesgue surfaces yields the displayed source triangle before any CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, or UniformModulus$^{\uparrow}$ endpoint is available. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformSourceTriangle`

Theorem 452.112 (Open-phase nonchoice cover)**.** *The open Phase route used by* FiniteLebesgueNumber$^{\uparrow}$ *contains no coordinate for open-cover compactness, selected subcovers, quotient equality, host minima, or external modulus data. Any compact-uniform consumer admitted by the route receives only the finite* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by the source triangle of Definition 452.111.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseNonchoiceCover`

Proof. The nonescape theorem rules out host Real$^{\uparrow}$ equality, open-cover compactness, selected subcovers, and external modulus reads before the four-face passage. The classifier pullback then fixes the positive-radius read to the finite dyadic cover, rational-ball, dyadic mesh, and four source faces.

The L10 sibling route and uniform-continuity handoff preserve only the displayed transport and naming rows. Since none of those rows has a field for a chosen subcover, quotient equality, host minimum, or detached modulus, the downstream compact-uniform consumer cannot obtain such data from the open Phase route. □

Theorem 452.113 (Uniform consumer radius cover)**.** *For later formalization under* BEDC.Derived.FiniteLebesgueN *the downstream radius-cover handoff of* FiniteLebesgueNumber$^{\uparrow}$ *is the single consumer statement that a* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* UniformModulus$^{\uparrow}$ *endpoint may consume the positive radius only through the source triangle of Definition 452.111 and the nonchoice cover of Definition 452.112. Thus the consumer-visible radius cover is finite, named, and exhausted by the Phase Real and Completion route.*

Proof. Apply the compact-uniform source triangle to obtain the required order: DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ are consumed before the compact, continuous-map, or uniform-modulus endpoint can read the radius.

Then apply the open-phase nonchoice cover. It removes open-cover compactness, selected subcovers, quotient equality, host minima, and external modulus data from the same handoff, so the endpoint receives exactly the finite named radius-cover route and no alternate source. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberUniformConsumerRadiusCover`

Definition 452.114 (Finite Lebesgue-number root-radius coherence ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier, the root-radius coherence ledger is the finite* BHist *row*

$$\Xi_\lambda = (D, B, \lambda, M, S, Q, R, H, C, P, N)$$

whose entries are the displayed DyadicCover$^{\uparrow}$ *cover row D, the* RationalBall$^{\uparrow}$ *radius readback row B for the positive radius λ, the* DyadicMesh$^{\uparrow}$ *mesh row M, the* StreamName$^{\uparrow}$ *source row S, the* RegSeqRat$^{\uparrow}$ *regular-rational approximation row Q, the* Real$^{\uparrow}$ *endpoint row R, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N. The ledger is coherent exactly when B, λ, M, S, Q, R are read as one named root-radius row through the same H, C, P, N support, with no separate host Real equality, quotient-stream coordinate, selected subcover, countable-choice selector, or ambient completeness field.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusCoherenceLedger`

Theorem 452.115 (Finite Lebesgue-number root-radius coherence)**.** *Let two accepted* FiniteLebesgueNumber$^{\uparrow}$ *routes carry root-radius coherence ledgers* Ξ_λ *and* Ξ'_λ*. If their* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows agree componentwise under the recorded* hsame *transports, and if their displayed* Cont *replays follow the same* Pkg *and* NameCert *support rows, then both routes expose the same root-radius obligation to every downstream consumer. The coherent obligation is the finite row* Ξ_λ *of Definition 452.114, not a detached compactness or completion datum.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier root radius coherence`

Proof. Begin with the root radius-window surface. Definition 452.80 already restricts the positive radius and finite-window obligations to the finite dyadic cover, rational-ball, mesh, stream-name, regular-rational, Real, history, transport, continuation, provenance, and local naming rows.

Next apply the componentwise hypothesis. Agreement under hsame fixes the rows $D, B, \lambda, M, S, Q, R$, and the common Cont replay fixes how those rows are inspected. The remaining P, N rows name the same package and local certificate, so there is no second route through which a consumer could distinguish Ξ_λ from Ξ'_λ.

Finally use the L10 sibling route. Definition 452.70 forces compact, compact-net, continuous-map, and uniform-modulus reads through the finite-name chain before the positive radius is visible. Therefore the two routes expose the same finite root-radius obligation and no outside compactness, equality, choice, selected-subcover, quotient-stream, or completeness coordinate. □

Theorem 452.116 (Finite Lebesgue-number compact-uniform radius admission)**.** *Every* CompactMetric$^{\uparrow}$*,* CompactNetModulusSelector$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, and* UniformModulus$^{\uparrow}$ *consumer admitted by an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route receives the positive radius only through the coherent root-radius ledger of Definition 452.114. In particular, admission to those consumers is the finite packet over* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*; it is not an independent compactness theorem, host minimum, selected subcover, countable-choice modulus, quotient-stream equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformRadiusAdmission`

Proof. Start with root-radius coherence. Definition 452.115 identifies the consumer-facing positive radius with the finite ledger Ξ_λ whenever the component rows are transported and replayed through the same support.

Attach the compact-uniform source triangle. Definition 452.111 makes every compact, continuous-map, or uniform-modulus consumer factor through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before the route reaches the consumer endpoint. The same theorem ties those four source faces to the visible dyadic cover, rational-ball, and mesh packet.

Now use the root radius-window consumer exactness supplied by Definition 452.80 together with the L10 route of Definition 452.70. The only admitted consumer data are the finite rows already named in Ξ_λ, so no consumer obtains an additional compactness, minimum, selected-subcover, choice, quotient, or completeness field. □

Theorem 452.117 (Finite Lebesgue-number Real-completion radius nonescape)**.** *The Real-and-completion route of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *carrier cannot replace the coherent root-radius ledger by host* Real$^{\uparrow}$ *equality, quotient streams, selected subcovers, countable choice, or ambient completeness. Any terminal radius read for the Phase Real or Completion route remains the*

finite BHist *packet* Ξ_λ *of Definition 452.114, transported by* hsame *and replayed by* Cont *through the displayed* DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, *and* Real$^\uparrow$ *source faces.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealCompletionRadiusNonescape`

Proof. First read the source triangle. Definition 452.111 places the Phase Real source faces before any compact-uniform consumer can inspect the radius, and it ties those faces to the same dyadic cover, rational-ball, and mesh packet.

Next apply root-radius coherence and compact-uniform admission. Definition 452.115 makes the radius stable under the displayed hsame and Cont rows, while Definition 452.116 states that admitted compact, compact-net, continuous-map, and uniform-modulus consumers receive only that finite ledger.

It remains to check the terminal exclusions. The L10 sibling route of Definition 452.70 and the root radius-window obligation surface of Definition 452.80 contain no fields for host Real equality, quotient streams, selected subcovers, countable choice, or ambient completeness. Since the terminal radius read is assembled only from those cited rows, each proposed escape has no ledger coordinate and the coherent root-radius packet remains the endpoint. □

Definition 452.118 (Finite Lebesgue-number choice-free tail-radius ledger)**.** *For an accepted* FiniteLebesgueNumber$^\uparrow$ *route, the choice-free tail-radius ledger is the finite* BHist *row family*

$$\Theta_\lambda^{\mathrm{tail}} = (D, B, \lambda, M, A_T, S_T, Q_T, R_T, H, C, P, N).$$

Here D *is the displayed* DyadicCover$^\uparrow$ *cover row,* B *is the* RationalBall$^\uparrow$ *radius readback row for the positive radius* λ*,* M *is the* DyadicMesh$^\uparrow$ *mesh row, and* A_T *is the selected finite-tail admission row. The rows* S_T, Q_T, R_T *record the* StreamName$^\uparrow$, RegSeqRat$^\uparrow$, *and* Real$^\uparrow$ *tail endpoint route;* H *records componentwise* hsame *transport,* C *records the displayed* Cont *replay,* P *records* Pkg *provenance, and* N *records the compatible local* NameCert *rows.*

The ledger is accepted only when the dyadic cover, rational-ball radius, dyadic mesh, selected finite-tail admission, stream-name row, regular-rational row, Real endpoint row, transport, replay, provenance, and local naming rows are read as one finite packet. A compact-uniform consumer may inspect λ *only as this packet and may not introduce a separate open-cover compactness, selected-subcover, host-minimum, quotient-stream, countable-choice, detached-completeness, or external-modulus coordinate.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberChoiceFreeTailRadiusLedger`

Theorem 452.119 (Choice-free tail-radius admission)**.** *Every* CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, TotallyBounded$^\uparrow$, *or* UniformModulus$^\uparrow$ *consumer admitted by the L10 route of an accepted* FiniteLebesgueNumber$^\uparrow$ *carrier factors its tail-radius read through the ledger* $\Theta_\lambda^{\mathrm{tail}}$ *of Definition 452.118. Thus the consumer-facing radius is the finite row family*

$$D, B, \lambda, M, A_T, S_T, Q_T, R_T, H, C, P, N$$

and no admitted read bypasses the selected finite-tail admission, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, hsame, Cont, Pkg, *or* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberChoiceFreeTailRadiusAdmission`

Proof. Start with selected-tail exhaustion. Definition 452.76 forces every selected-tail witness and every finite-window witness visible to a modulus consumer through the displayed finite packet, with no hidden selector, quotient stream, countable choice coordinate, host real equality, or detached $\mathsf{Real}^{\uparrow}$ representative.

Next use the root-radius coherence package. Definition 452.115 keeps the positive radius stable under the displayed hsame transports and Cont replays, while Definition 452.116 makes compact, compact-net, continuous-map, and uniform-modulus consumers read that coherent finite ledger rather than an outside compactness datum.

Finally read the L10 handoff. Definition 452.92 places the compact-uniform endpoints on the Phase Real route, and Definition 452.111 identifies the source faces as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ tied to the dyadic cover, rational-ball, and mesh rows. Combining these cited rows gives exactly $\Theta_{\lambda}^{\mathrm{tail}}$, so every admitted tail-radius read factors through the stated finite ledger. □

Theorem 452.120 (Tail-radius transport determinacy)**.** *Let two accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *routes carry choice-free tail-radius ledgers* $\Theta_{\lambda}^{\mathrm{tail}}$ *and* $\Theta_{\mu}^{\mathrm{tail}}$*. If their* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*, selected finite-tail admission,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows agree componentwise under the recorded* hsame *transports, and their* Cont *replays use the same* Pkg *provenance and local* NameCert *support, then they expose the same radius to every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer admitted by the L10 route.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTailRadiusTransportDeterminacy`

Proof. First identify the two ledgers row by row. The hypotheses make the cover, rational-ball radius, mesh, selected finite-tail admission, stream-name, regular-rational, and Real endpoint rows agree under hsame, and the shared Cont replay fixes the order in which a consumer can inspect them.

Now apply root-radius coherence. Definition 452.115 says that accepted routes agreeing on the radius, source, transport, replay, provenance, and local naming support expose the same root-radius obligation to downstream consumers.

It remains to pass from root radius to tail radius. By Definition 452.76, the selected finite-tail side has no second modulus selector, quotient stream, choice coordinate, host equality, or detached Real representative. By Definition 452.92, the compact-uniform consumers read only the L10 packet. Therefore the two accepted routes expose the same consumer radius across $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$. □

Theorem 452.121 (Compact-uniform choice refusal)**.** *The compact-uniform handoff of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *route cannot replace the choice-free tail-radius ledger of Definition 452.118 by open-cover compactness, selected subcovers, host minima, quotient stream equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or external modulus data. Any radius visible to* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *is read only from*

$$\Theta_{\lambda}^{\mathrm{tail}} = (D, B, \lambda, M, A_T, S_T, Q_T, R_T, H, C, P, N).$$

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactUniformChoiceRefusal`

Proof. Read the ledger definition first. Definition 452.118 lists the only rows available at the tail-radius handoff: dyadic cover, rational-ball radius, dyadic mesh, selected finite-tail admission, stream-name, regular-rational, Real endpoint, transport, replay, provenance, and local naming.

The selected-tail and source-triangle theorems remove the proposed outside coordinates. Definition 452.76 rules out hidden modulus selectors, quotient streams, countable choice, host real equality, and detached Real representatives, while Definition 452.111 requires the compact-uniform consumer to pass through the named Phase Real source faces before the radius is visible.

Finally apply compact-uniform admission along the L10 route. Definition 452.92 and Definition 452.116 leave no consumer coordinate outside the finite ledger. Since open-cover compactness, selected subcovers, host minima, quotient stream equality, countable choice, detached completeness, and external modulus data are absent from that ledger, the handoff refuses each of those readings. □

Definition 452.122 (Finite Lebesgue-number Real-phase source-exhaustion ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route, the Real-phase source-exhaustion ledger is the finite row family*

$$\Xi_{\lambda}^{\mathrm{real}} = (D, B, M, A, S, Q, R, H, C, P, N).$$

Here D is the displayed DyadicCover$^{\uparrow}$ *cover row, B is the* RationalBall$^{\uparrow}$ *positive-radius row, M is the* DyadicMesh$^{\uparrow}$ *mesh row, and A records the finite admissibility row through which the radius is offered to compact, continuous, totally bounded, and uniform endpoints. The rows S, Q, R are the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *source faces; H records the permitted* hsame *transports, C records the* Cont *replay, P records* Pkg *provenance, and N records the compatible local* NameCert *rows.*

The ledger is accepted only as one finite BHist *packet. A consumer may read the radius λ only through these rows and may not replace the packet by host* Real$^{\uparrow}$ *equality, quotient stream equality, selected limits, detached completeness, choice, an external modulus, or arbitrary open-cover compactness.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealPhaseSourceExhaustionLedger`

Theorem 452.123 (Finite Lebesgue-number Real-phase source exhaustion)**.** *Every* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *consumer admitted by an accepted* FiniteLebesgueNumber$^{\uparrow}$ *radius read passes through the ledger $\Xi_{\lambda}^{\mathrm{real}}$ of Definition 452.122 before it can inspect the radius. Thus every compact, continuous, totally bounded, compact-net, or uniform radius read is exhausted by the finite rows*

$$D, B, M, A, S, Q, R, H, C, P, N$$

over DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealPhaseSourceExhaustion`
↪ variant `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCarrier real phase source exhaustion`

Proof. Start with the open Phase boundary. Definition 452.89 prevents the source read from reaching host Real$^{\uparrow}$ equality, open-cover compactness, selected subcovers, or external modulus data before the four-face route is consumed.

Next identify the radius readback. Definition 452.90 says that every compact, compact-net, continuous-map, totally bounded, or uniform radius read factors through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$, together with the visible dyadic cover, rational-ball, and dyadic mesh rows.

Now pass to the L10 consumer route. Definition 452.93 makes the consumer-facing read exact at the finite packet, while Definition 452.74 keeps compact and compact-net endpoints from introducing a second source.

Combining those four cited surfaces gives precisely $\Xi_\lambda^{\mathrm{real}}$. Each named row is already in the finite BHist, hsame, Cont, Pkg, and NameCert packet, so no admitted consumer radius read has a remaining source outside the displayed ledger. □

Theorem 452.124 (Finite Lebesgue-number compact-consumer source verdict)**.** *The compact and compact-net endpoints of an accepted* FiniteLebesgueNumber$^\uparrow$ *route receive only the finite dyadic radius verdict recorded by* $\Xi_\lambda^{\mathrm{real}}$*. They receive the rows* $D, B, M, A, S, Q, R, H, C, P, N$ *and no open-cover compactness principle, selected subcover data, host minimum, quotient stream equality, chosen limit, detached completeness primitive, choice coordinate, or external modulus selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberCompactConsumerSourceVerdict`

Proof. First read the compact endpoint through determinacy. Definition 452.74 fixes the compact and compact-net consumer verdict to the finite carrier rows and their recorded transport.

Then use radius readback exactness. Definition 452.90 and Definition 452.93 identify the visible radius as the dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, Real endpoint, transport, replay, provenance, and naming packet.

Finally exclude outside compactness sources. The open Phase nonescape theorem removes open-cover compactness, selected subcovers, and external modulus data before the source is read, and the Real-phase source exhaustion theorem above leaves no second endpoint source after the finite packet is consumed.

Therefore compact and compact-net consumers receive exactly the finite dyadic radius verdict and none of the refused compactness, selection, equality, completeness, choice, or modulus data. □

Theorem 452.125 (Finite Lebesgue-number Real-phase offtable refusal)**.** *The Real-and-Completion handoff of an accepted* FiniteLebesgueNumber$^\uparrow$ *route refuses every radius source not listed in* $\Xi_\lambda^{\mathrm{real}}$*. In particular, host* Real$^\uparrow$ *equality, quotient stream equality, selected limits, detached completeness, countable choice, an external modulus, and arbitrary open-cover compactness are offtable for every compact, continuous, totally bounded, compact-net, and uniform consumer radius read.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealPhaseOfftableRefusal`

Proof. Read the ledger definition first. Definition 452.122 lists the only rows available at the Real-and-Completion handoff: dyadic cover, rational-ball radius, dyadic mesh, finite admissibility, stream-name, regular-rational, Real endpoint, transport, replay, provenance, and local naming.

The source and readback theorems close the route around those rows. Definition 452.89 prevents outside compactness or selector data from entering before the four-face source, and Definition 452.90 forces later radius reads through the same four faces.

The consumer exactness theorems close the route at the endpoint. Definition 452.93 identifies the consumer-visible read with the finite packet, while Definition 452.74 prevents compact endpoints from replacing it by a separate source.

Since host Real$^\uparrow$ equality, quotient stream equality, selected limits, detached completeness, countable choice, external modulus data, and arbitrary open-cover compactness occur in none of the accepted rows, the Real-phase handoff refuses all of them for every admitted consumer. □

Theorem 452.126 (Finite Lebesgue-number root radius choice-free tail exhaustion)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *or* $\mathsf{UniformModulus}^{\uparrow}$ *radius read admitted by an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root route factors through the choice-free tail-radius ledger* $\Theta_{\lambda}^{\mathrm{tail}}$ *and the Real-phase source-exhaustion ledger* $\Xi_{\lambda}^{\mathrm{real}}$*. The visible root radius is therefore read only from the finite rows over* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, *selected finite-tail admission,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*, and not from open-cover compactness, selected subcovers, quotient stream equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or external modulus data.*

Proof. Start with the tail ledger. Definition 452.119 places every compact, continuous, totally bounded, and uniform tail-radius read inside $\Theta_{\lambda}^{\mathrm{tail}}$, and Definition 452.120 makes that read stable under the recorded componentwise hsame transports.

Now attach the Real-phase route. Definition 452.123 forces the same radius read through $\Xi_{\lambda}^{\mathrm{real}}$ before any endpoint can inspect it, while Definition 452.125 removes host equality, quotient stream equality, selected limits, detached completeness, choice, external modulus data, and arbitrary open-cover compactness from the accepted rows.

The two ledgers share the displayed dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, Real endpoint, transport, replay, provenance, and naming rows. Hence the root route exposes the radius only through the stated finite tail-and-Real packet, so none of the refused compactness, selection, quotient, choice, completeness, or modulus coordinates remains available to an admitted consumer. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusChoiceFreeTailExhaustion`

Theorem 452.127 (Finite Lebesgue-number root radius source determinacy)**.** *Let two accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root routes carry tail-radius ledgers and Real-phase source ledgers for visible radii* λ *and* μ*. If their* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows agree componentwise under the recorded* hsame *transports, and if their selected finite-tail admission rows name the same local source, then every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$ *consumer receives the same downstream root-radius source from the two routes.*

Proof. First compare the tail-radius rows. The componentwise hypotheses identify the dyadic cover, rational-ball, dyadic mesh, stream-name, regular-rational, Real endpoint, history, replay, provenance, and local naming rows, while the shared selected finite-tail admission fixes the only finite tail through which a consumer can read the radius.

Apply Definition 452.120. The two tail ledgers therefore expose the same consumer radius along the L10 route, with no second selector, quotient stream, choice coordinate, host equality, or detached Real representative.

It remains to identify the source side. Definition 452.123 confines the source route to the same Real-phase ledger, and Definition 452.125 prevents any outside source from being substituted after the rows agree. Thus the downstream compact, continuous, totally bounded, and uniform consumers receive the same root-radius source. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusSourceDeterminacy`

Theorem 452.128 (Finite Lebesgue-number root radius NameCert obligation package)**.** *An accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root radius satisfies the NameCert obligation package for the Real-and-Completion route when its finite rows provide:*

1. *the choice-free tail-radius admission and transport determinacy of Definition 452.119 and Definition 452.120;*

2. the Real-phase source exhaustion and offtable refusal of Definition 452.123 and Definition 452.125;

3. the root tail exhaustion and source determinacy of Definition 452.126 and Definition 452.127.

Under these clauses, every accepted CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *read is accounted for by the finite* BHist, hsame, Cont, Pkg, *and* NameCert *packet over the displayed* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusNameCertObligationPackage`

Proof. Assemble the tail side first. The choice-free tail-radius admission theorem supplies the finite ledger through which each admitted consumer reads the tail radius, and tail-radius transport determinacy says that componentwise agreement on that ledger fixes the consumer-visible radius.

Assemble the Real-phase side next. The Real-phase source-exhaustion theorem supplies the finite source ledger for the same radius, and the offtable refusal theorem rules out host equality, quotient stream equality, selected limits, detached completeness, choice, external modulus data, and arbitrary open-cover compactness as substitute sources.

Finally apply the two root-radius theorems above. Root tail exhaustion identifies the common finite packet seen by consumers, and root source determinacy says that agreement on the displayed dyadic, stream, regular-rational, Real, history, replay, provenance, and naming rows determines the downstream source. These clauses are exactly the NameCert obligation package claimed for the accepted Real-and-Completion route. □

Definition 452.129 (Finite Lebesgue-number Phase Real radius-consumer ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *root route, the Phase Real radius-consumer ledger is the finite* BHist *packet*

$$\Lambda_{\lambda}^{\mathrm{cons}} = (D, B, M, E, S, Q, R, K, F, U, H, C, P, N).$$

Here D is the DyadicCover$^{\uparrow}$ *cover row, B is the* RationalBall$^{\uparrow}$ *positive-radius row, M is the* DyadicMesh$^{\uparrow}$ *mesh row, and E is the* DyadicRatCore$^{\uparrow}$ *face through which the radius is read before the stream face. The rows S, Q, R are the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *source faces. The rows K, F, U are the* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, *and* UniformModulus$^{\uparrow}$ *consumer endpoints, while H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and compatible local* NameCert *rows.*

The packet is accepted only when these rows form one finite NameCert route for the same positive radius λ. A compact, continuous, or uniform endpoint may inspect λ only through $\Lambda_{\lambda}^{\mathrm{cons}}$ and may not replace it by open-cover compactness, a host minimum, a selected subcover, quotient stream equality, a selected limit, countable choice, metric-completeness data, or a detached Real$^{\uparrow}$ *completeness claim.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusConsumerLedger`

Theorem 452.130 (Finite Lebesgue-number Phase Real radius-consumer carrier)**.** *Every admitted* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *consumer radius source downstream of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *root route projects to exactly the rows of $\Lambda_{\lambda}^{\mathrm{cons}}$ in Definition 452.129. Thus the consumer-visible source is carried by*

DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$,

together with the endpoint rows CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$ *and the* BHist, hsame, Cont, Pkg, *and* NameCert *support rows, and by no further radius-source coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusConsumerCarrier`

Proof. Start from the root route. Definition 452.126 and Definition 452.127 place every compact, continuous, totally bounded, or uniform root-radius read on the shared tail-and-Real packet over the dyadic cover, rational ball, dyadic mesh, stream-name, regular-rational, Real, history, transport, replay, provenance, and naming rows.

Next attach the Real-phase source boundary. Definition 452.123 forces admitted compact, continuous, compact-net, totally bounded, and uniform radius reads through the finite Real-phase ledger, while Definition 452.125 denies substitute source coordinates outside that ledger.

Finally identify the four faces and the consumer endpoints. Definition 452.108 adds the $\mathsf{DyadicRatCore}^{\uparrow}$ face to the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, and Definition 452.101 ties the compact, continuous, and uniform endpoints to the same radius-window packet. These cited surfaces list exactly the rows of $\Lambda_{\lambda}^{\mathrm{cons}}$, so every admitted consumer source projects to that packet and to no outside radius coordinate. □

Theorem 452.131 (Finite Lebesgue-number Phase Real radius-consumer stability)**.** *Let two accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root routes carry Phase Real radius-consumer ledgers for visible radii* λ *and* μ*. If their* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows agree componentwise under the recorded* hsame *transports, then the compact, continuous, and uniform endpoints receive the same radius-consumer source from the two routes.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusConsumerStability`

Proof. First compare the root radius sources. Definition 452.127 says that componentwise agreement on the dyadic, stream, regular-rational, Real, history, replay, provenance, and naming rows fixes the downstream root-radius source for compact, continuous, totally bounded, and uniform consumers.

Now compare the compact radius-window packets. Definition 452.101 and Definition 452.102 make the $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ rows consume one shared finite packet, so the agreed endpoint rows cannot introduce a second source after transport.

Finally use four-face exactness. Definition 452.108 fixes the consumer-visible source to the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with the same BEDC support rows. Since each of those rows is identified by the recorded hsame transports, the compact, continuous, and uniform endpoints receive the same radius-consumer source. □

Theorem 452.132 (Finite Lebesgue-number Phase Real radius-consumer exhaustion)**.** *The Phase Real radius-consumer ledger* $\Lambda_{\lambda}^{\mathrm{cons}}$ *exports no open-cover compactness, host minimum, selected subcover, quotient stream equality, selected limit, countable choice, metric-completeness datum, or* $\mathsf{Real}^{\uparrow}$ *completeness claim. Every compact, continuous, or uniform radius read admitted by the ledger is exhausted by its displayed dyadic, four-face Real, endpoint, history, transport, replay, provenance, and naming rows.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusConsumerExhaustion`

Proof. Read the ledger definition first. Definition 452.129 lists only the dyadic cover, rational-ball radius, dyadic mesh, $\mathsf{DyadicRatCore}^{\uparrow}$, stream-name, regular-rational, Real, compact, continuous, uniform, history, transport, replay, provenance, and naming rows.

The root and Real-phase exhaustion theorems remove the proposed outside coordinates at the source. Definition 452.126 excludes open-cover compactness, selected subcovers, quotient stream

equality, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, and external modulus data from the root route, while Definition 452.123 leaves every admitted endpoint read inside the finite Real-phase packet.

The compact-window and four-face theorems remove the remaining endpoint escapes. Definition 452.101 excludes selected limits, quotient stream equality, selected subcovers, host minima, choice-like cover selection, and unbudgeted radius selection for compact and continuous consumers, and Definition 452.108 excludes open-cover compactness, selected subcover, quotient equality, host minimum, countable choice, Real completeness, and external modulus coordinates from the four-face packet.

Combining these surfaces, every admitted compact, continuous, or uniform read is already accounted for by $\Lambda_{\lambda}^{\mathrm{cons}}$. None of open-cover compactness, a host minimum, a selected subcover, quotient stream equality, selected limits, countable choice, metric-completeness data, or a detached $\mathsf{Real}^{\uparrow}$ completeness claim has a ledger row through which it can be exported. □

Theorem 452.133 (Finite Lebesgue-number open Phase root-unblock radius coverage)**.** *Every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root radius read exported to the open Phase Real route factors through the finite radius-unblock packet*

$$D, B, M, E, S, Q, R, H, C, P, N$$

where D is the displayed $\mathsf{DyadicCover}^{\uparrow}$ *cover row, B is the* $\mathsf{RationalBall}^{\uparrow}$ *positive-radius row, M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh row, E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source face, S, Q, R are the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces, and the surrounding* BHist *records H, C, P, N as the* hsame *transport,* Cont *replay,* Pkg *provenance, and compatible local* NameCert *rows. Thus the root-unblock obligation exposed to compact-uniform consumers is a finite BEDC packet over these rows, not a host-side radius source.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRootUnblockRadiusCoverage`

Proof. Start with the root radius-window surface. Definition 452.80 identifies the positive radius and finite window through the dyadic cover, rational-ball radius, dyadic mesh, stream-name, regular-rational, Real, history, transport, replay, provenance, and naming rows.

Next attach the Phase Real source route. Definition 452.75 forces each public radius read through the same dyadic, stream, regular-rational, and Real rows before a consumer can inspect it, while Definition 452.78 adds the $\mathsf{DyadicRatCore}^{\uparrow}$ face to the compact, compact-net, continuous-map, and uniform-modulus export.

Finally read the exit and consumer ledgers. Definition 452.91 gives the open Phase root-unblock exit surface, and Definition 452.132 exhausts the consumer-visible radius source by the same finite rows. The displayed packet therefore covers every accepted root radius read exported to the open Phase Real route. □

Theorem 452.134 (Finite Lebesgue-number open Phase root-unblock choice refusal)**.** *The open Phase root-unblock obligation packet for* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *exports no open-cover compactness principle, selected subcover, host minimum, quotient stream equality, countable choice coordinate, detached* $\mathsf{Real}^{\uparrow}$ *completeness row, or external modulus row. Any admitted compact, continuous, totally bounded, or uniform radius use must be read from the finite dyadic, four-face Real,* BHist, hsame, Cont, Pkg, *and* NameCert *packet of Definition 452.133.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRootUnblockChoiceRefusal`

Proof. Read the positive side from the coverage theorem. Definition 452.133 leaves only the dyadic cover, rational-ball radius, dyadic mesh, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and BEDC support rows available to the open Phase root-unblock route.

The source and exit theorems supply the negative side. Definition 452.75 prevents the source read from being replaced by an outside compactness or completeness primitive, and Definition 452.91 states that the exit surface supplies no host Real equality, open-cover compactness, selected subcover, selected limit, detached completeness primitive, or external modulus data.

The consumer ledger closes the refusal at the endpoints. Definition 452.132 removes open-cover compactness, host minima, selected subcovers, quotient stream equality, selected limits, countable choice, metric-completeness data, and detached $\mathsf{Real}^{\uparrow}$ completeness from compact, continuous, and uniform reads. The same finite packet is the only admitted source, so none of the listed outside rows can be exported. □

Theorem 452.135 (Finite Lebesgue-number open Phase root-unblock consumer readback)**.** *The* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *consumers downstream of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *open Phase root-unblock route read the same finite packet. Their visible radius source is exactly the shared* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route, with no consumer-specific radius source added at the endpoint.*

Proof. First identify the shared source. Definition 452.133 factors every accepted root radius read through the finite dyadic and four-face Real packet, and Definition 452.91 presents that packet as the root-unblock exit surface for compact-uniform consumers.

Next pass through Real-phase coverage. Definition 452.78 routes compact, compact-net, continuous-map, and uniform-modulus endpoints through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the finite carrier rows. Definition 452.132 then states that every compact, continuous, or uniform radius read is exhausted by the displayed dyadic, four-face Real, endpoint, history, transport, replay, provenance, and naming rows.

Finally compare the endpoint consumers. The $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ rows have no coordinate outside the packet by Definition 452.134. Since each endpoint reads through the same source rows and the same BHist, hsame, Cont, Pkg, and NameCert support, their consumer-visible radius source is the same finite packet. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRootUnblockConsumerReadback`

Theorem 452.136 (Finite Lebesgue-number root-unblock endpoint uniformity)**.** *If two downstream endpoint reads for* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *are accepted from the same* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *open Phase root-unblock route, then their visible positive-radius source is one finite packet over* $\mathsf{DyadicCover}^{\uparrow}$*,* $\mathsf{RationalBall}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Conversely, any endpoint read displaying exactly that packet is an admissible consumer-visible radius read for the same route.*

Proof. Start from the shared source assertion. Definition 452.133 says that every accepted root radius read exported to the open Phase Real route factors through the displayed dyadic, rational-ball, mesh, dyadic-rational, stream, regular-rational, Real, history, transport, replay, provenance, and naming rows.

The refusal theorem removes every possible endpoint-specific replacement. By Definition 452.134, the route exports no consumer-specific radius source, open-cover compactness, selected subcover,

host minimum, quotient equality, choice coordinate, external modulus, selected limit, or detached $\mathsf{Real}^{\uparrow}$-completeness row. Thus two accepted endpoint reads cannot differ by importing any outside radius witness.

Now compare the endpoint reads. Definition 452.135 states that the $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers read exactly the same finite packet and add no endpoint-specific source. Hence any two accepted reads from the same root-unblock route have the same visible positive-radius source.

Conversely, an endpoint read that displays exactly the finite packet named above already presents the consumer-visible route required by Definition 452.135. Since the refusal theorem excludes only outside rows and the displayed packet is precisely the covered source, such a read is admissible for the same open Phase root-unblock route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootUnblockEndpointUniformity`

Theorem 452.137 (Finite Lebesgue-number root-radius export exactness)**.** *An accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *root radius export is admissible exactly when it displays the finite packet over* $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The admissible export has no open-cover coordinate, selected-subcover coordinate, host minimum coordinate, or detached consumer radius source.*

Proof. For the forward direction, start with an accepted root radius export. The carrier of Definition 452.1 fixes the local packet $D, W, \lambda, M, H, C, P, N$, and Definition 452.2 names the carrier, classifier, radius-modulus stability, finite-window coverage, and non-escape rows that govern the packet.

Now pass to the open Phase route. Definition 452.133 factors every accepted root radius read through the displayed dyadic, rational-ball, mesh, four-face Real, history, transport, replay, provenance, and naming rows. Definition 452.134 removes open-cover compactness, selected subcovers, host minima, quotient stream equality, detached completeness, and external modulus coordinates from that export.

For the reverse direction, suppose a radius export displays exactly those rows and no outside coordinate. Definition 452.135 identifies that displayed packet as the consumer-visible radius source for the compact, continuous, totally bounded, and uniform endpoints. Definition 452.136 then states that any endpoint read displaying exactly this finite packet is admissible for the same open Phase root-unblock route. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRootRadiusExportExactness`

Theorem 452.138 (Finite Lebesgue-number open Phase positive-radius stability)**.** *Componentwise* hsame *transport and displayed* Cont *replay preserve the positive-radius row of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *open Phase root-unblock export only together with its* $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. In particular, the positive-radius row cannot be transported as a detached scalar outside the Phase Real and Completion packet.*

Proof. Begin with the local carrier and NameCert rows. In Definition 452.1, the positive ledger λ is stored between the finite-window and mesh coordinates and is accompanied by the support rows H, C, P, N. The radius-modulus stability row of Definition 452.2 preserves λ only with the rational-ball and dyadic-mesh evidence that witnesses its refinement into the finite cover.

The open Phase coverage theorem extends this local stability to the root-unblock export. Definition 452.133 places the positive radius inside the same packet as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$,

RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. Thus any componentwise hsame transport and displayed Cont replay is a transport of the displayed packet, not of an isolated radius coordinate.

Endpoint uniformity gives the stability test. By Definition 452.135 and Definition 452.136, every compact, continuous, totally bounded, or uniform endpoint reads the same finite packet. The choice-refusal theorem excludes a second radius source, so preserving the positive-radius row requires preserving the rational-ball, mesh, stream, regular-rational, and Real rows along with it. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhasePositiveRadiusStability`

Theorem 452.139 (Finite Lebesgue-number L10 consumer radius exhaustion)**.** *Every* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *consumer read of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *open Phase root-unblock route is exhausted by the same Phase Real and Completion packet over* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *No consumer may add a consumer-specific radius witness to that route.*

Proof. First identify the common packet. The carrier of Definition 452.1 and the NameCert obligations of Definition 452.2 bind the finite radius ledger to the dyadic cover, finite-window, mesh, transport, replay, provenance, and naming rows.

Next route that packet through the open Phase root-unblock surface. Definition 452.133 supplies the Phase Real and Completion faces, while Definition 452.134 removes open-cover compactness, selected subcovers, host minima, quotient stream equality, detached completeness, and external modulus data.

Finally read the downstream consumers. Definition 452.135 says that CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, and UniformModulus$^{\uparrow}$ consume exactly the same finite packet. Definition 452.136 identifies any endpoint display of that packet as the admissible read for the route, so a consumer-specific radius witness has no remaining coordinate to occupy. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberL10ConsumerRadiusExhaustion`

Theorem 452.140 (Finite Lebesgue-number four-face radius exit coverage)**.** *Every accepted* FiniteLebesgueNumber *root radius visible to a* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *compact-net, or* UniformModulus$^{\uparrow}$ *consumer is exhausted by the four current open-phase source faces* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *together with the surrounding* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The visible exit package contains no radius source outside that four-face BEDC packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberFourFaceRadiusExitCoverage`
`semantic name certificate`

Proof. Begin with root-unblock coverage. Definition 452.133 places every accepted root radius read inside the dyadic, rational-ball, mesh, four-face Real, history, transport, replay, provenance, and naming packet.

Now pass through the exact export and L10 endpoint route. Definition 452.137 identifies admissibility with displaying exactly that finite packet, and Definition 452.139 says that compact, continuous, totally bounded, and uniform consumers cannot add a consumer-specific radius witness.

Finally attach the compact-net face to the same route. The sibling route of Definition 452.70 sends compact and compact-net reads through the finite-name chain before the positive radius

is visible. Therefore every listed consumer reads the same four-face exit package and no outside radius source remains available. □

Theorem 452.141 (Finite Lebesgue-number four-face radius exit determinacy)**.** *For accepted* FiniteLebesgueNumber$^{\uparrow}$ *root-radius exits, componentwise agreement on the* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows determines the downstream radius classifier seen by* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *compact-net, and* UniformModulus$^{\uparrow}$ *consumers. Two such exits with the same displayed rows expose the same consumer-visible classifier.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberFourFaceRadiusExitDeterminacy`

Proof. First reduce the classifier to the root packet. By Definition 452.137, an admissible root-radius export is exactly the finite packet over the four source faces and the support rows, with no independent compactness, minimum, subcover, or detached radius coordinate.

Next use endpoint uniformity. Definition 452.139 and Definition 452.136 make every compact, continuous, totally bounded, and uniform endpoint read the same displayed packet. Componentwise hsame agreement and displayed Cont replay therefore carry the endpoint classifier without changing its source rows.

The compact-net route has no separate classifier channel. Definition 452.70 routes compact-net visibility through the same finite-name chain, while Definition 452.88 prevents the stream and regular-rational faces from being replaced by quotient stream equality, selected representatives, choice, or ambient completeness. Thus equal four-face rows determine the downstream radius classifier for every listed consumer. □

Theorem 452.142 (Finite Lebesgue-number open Phase root-radius classifier totality)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *root-radius exit visible to a* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *compact-net, or* UniformModulus$^{\uparrow}$ *consumer has exactly one consumer-visible radius classifier. This classifier is determined by the shared* DyadicCover$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of the exit packet, and no endpoint can add a second classifier or detach the radius from the Phase Real and Completion packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseRootRadiusClassifierTotality`

Proof. Existence comes from the four-face exit surface. Definition 452.140 puts every accepted root radius visible to the listed consumers inside the displayed dyadic, four-face Real, history, transport, replay, provenance, and naming packet. Definition 452.137 identifies admissible export with that finite packet, so the visible classifier is read from the same rows rather than from an outside coordinate.

The endpoint ledger removes the remaining sources. Definition 452.139 exhausts compact, continuous, totally bounded, and uniform reads by the Phase Real and Completion packet, while Definition 452.70 sends compact-net visibility through the same finite-name route. Thus every accepted endpoint has a consumer-visible classifier carried by the shared packet.

Uniqueness follows from componentwise agreement. Definition 452.141 says that agreement on the four source faces and the BHist, hsame, Cont, Pkg, and NameCert rows determines the downstream classifier for every listed consumer. A second classifier would either agree with those rows and therefore be the same classifier, or differ by an outside coordinate.

The outside-coordinate case is excluded by the refusal boundary. Definition 452.134 and Definition 452.143 rule out host minima, selected subcovers, quotient stream equality, choice rows,

selected limits, and detached $\mathsf{Real}^{\uparrow}$ completeness. Hence no endpoint can add a second classifier or detach the radius from the Phase Real and Completion packet. □

Theorem 452.143 (Finite Lebesgue-number four-face radius exit nonescape)**.** *The four-face* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius exit package exports no open-cover compactness row, selected subcover row, host minimum row, quotient stream equality row, selected limit row, choice row, or detached* $\mathsf{Real}^{\uparrow}$ *completeness row. Any claimed compact, continuous, totally bounded, compact-net, or uniform-modulus use of such a row is outside the admitted open-phase root-radius exit.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberFourFaceRadiusExitNonescape`

Proof. Read the positive source from coverage and exactness. Definition 452.140 and Definition 452.137 leave only the dyadic cover, rational-ball radius, dyadic mesh, four source faces, and BEDC support rows available to an accepted root-radius exit.

Then apply the existing refusal boundary. Definition 452.134 removes open-cover compactness, selected subcovers, host minima, quotient stream equality, countable choice coordinates, detached $\mathsf{Real}^{\uparrow}$ completeness, and external modulus data from the open-phase root-unblock obligation packet.

Finally close the endpoint escapes. Definition 452.139 exhausts downstream consumer reads by the same Phase Real and Completion packet, and Definition 452.88 blocks replacement of the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ faces by quotient stream equality, selected representatives, choice, selected limits, or ambient completeness. Hence none of the listed rows can be exported from the four-face radius exit package. □

Definition 452.144 (Finite Lebesgue-number Phase Real radius-window carrier)**.** *For an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *route, the Phase Real radius-window carrier is the finite* BHist *packet*

$$\mathcal{W}_{\lambda}^{\mathrm{PhaseReal}} = (E, R, D, W, S, H, C, P, N)$$

where E is the finite Lebesgue-number request row, R is the rational-ball radius row, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ *radius readback row, W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, S is the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal route, and H, C, P, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and compatible local* $\mathsf{NameCert}$ *rows. The carrier is read against the four source chapters chapter 291, chapter 126, chapter 301, and chapter 38; it contains no quotient-stream coordinate, host-limit coordinate, countable-choice coordinate, open-cover compactness coordinate, or ambient Real-completeness coordinate.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowCarrier`

Theorem 452.145 (Finite Lebesgue-number dyadic radius-window admission)**.** *Every admitted radius-window read of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real carrier exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius readback row D before the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W. Equivalently, a downstream compact, continuous, totally bounded, or uniform consumer may read W only as part of the packet* $\mathcal{W}_{\lambda}^{\mathrm{PhaseReal}}$ *of Definition 452.144, with the displayed dyadic radius read already present.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowAdmission`
Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberDyadicRadiusWindowAdmission`

Proof. Start with the carrier definition. The packet $\mathcal{W}_{\lambda}^{\mathrm{PhaseReal}}$ places the finite Lebesgue-number request E, the rational-ball radius row R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D before the finite stream window W is available to any consumer.

The existing root and open Phase surfaces give the same order at the chapter boundary. Definition 452.80 identifies the positive radius and finite-window obligations with the dyadic, stream, regular-rational, Real, history, transport, continuation, provenance, and naming rows. Definition 452.133 then exports the root radius read through the finite dyadic and four-face Real packet.

Finally use the Phase Real audit and source exactness. Definition 452.105 and Definition 452.108 keep the consumer-visible radius inside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support. Since W has no row outside this packet, an admitted window read can occur only after D has been displayed. □

Theorem 452.146 (Finite Lebesgue-number Real seal radius readback)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal read used by an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real radius-window carrier factors through the ordered rows*

$$D, \quad W, \quad S$$

of Definition 452.144. Thus the Real-side radius is read back only through the $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, and the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal route, together with the shared* H, C, P, N *support.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberRealSealRadiusReadback`

Proof. First expose the dyadic admission. By Definition 452.145, an admitted radius-window read displays D before W. The dyadic readback is therefore already part of the carrier before the stream window is inspected.

Next attach the regular-rational and Real faces. Definition 452.108 routes every consumer-visible radius through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The definition of $\mathcal{W}_{\lambda}^{\mathrm{PhaseReal}}$ records the $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ part of this route as S.

The terminal boundary supplies the readback exclusion. Definition 452.95 keeps terminal radius reads on the same L10 packet and denies host minima, quotient equality, selected limits, countable choice, and detached Real completeness at the Real boundary. Hence any Real seal read factors through D, W, S and the common H, C, P, N rows, rather than through an off-packet Real source. □

Theorem 452.147 (Finite Lebesgue-number open Phase radius-window nonescape)**.** *No accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *open Phase radius-window carrier can escape to quotient streams, host limits, countable choice, open-cover compactness, or ambient Real completeness. Every admitted compact, continuous, totally bounded, compact-net, or uniform radius read remains inside the finite packet*

$$E, \quad R, \quad D, \quad W, \quad S, \quad H, \quad C, \quad P, \quad N$$

of Definition 452.144.

Proof. Begin with the two positive readback gates. Definition 452.145 forces the window side to expose the $\mathsf{DyadicRatCore}^{\uparrow}$ row before $\mathsf{StreamName}^{\uparrow}$ can be consumed. Definition 452.146 then factors each Real seal read through D, W, S with the shared support rows.

Now read the existing nonescape boundaries. Definition 452.89 keeps the open Phase source from bypassing the stream and regular-rational rows through host sequences, quotient stream equality, selected representatives, choice, or ambient completeness. Definition 452.88 gives the same refusal for the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ faces.

Finally apply terminal and consumer exhaustion. Definition 452.109 and Definition 452.139 keep compact, compact-net, continuous, totally bounded, and uniform consumers inside the four-face Phase Real and Completion packet. Since the carrier itself has no coordinate for quotient

streams, host limits, countable choice, open-cover compactness, or ambient Real completeness, none of those rows can be exported from an admitted radius-window read. □

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberOpenPhaseNonescape

Theorem 452.148 (Finite Lebesgue-number Phase Real radius-window no extra source)**.** *Every accepted* FiniteLebesgueNumber$^{\uparrow}$ *Phase Real radius-window packet may read only the rows*

$$D, \quad W, \quad S, \quad H, \quad C, \quad P, \quad N$$

of Definition 452.144, together with the displayed finite history BHist *and its componentwise* hsame*,* Cont*,* Pkg*, and* NameCert *support. In particular, the packet has no readable source beyond the* DyadicRatCore$^{\uparrow}$ *endpoint and radius row, the* StreamName$^{\uparrow}$ *finite-window row, the* RegSeqRat$^{\uparrow}$ *regularity and classifier row, and the* Real$^{\uparrow}$ *seal row.*

Proof. Read the packet from its carrier definition. The only source chapters named there are chapter 291, chapter 126, chapter 301, and chapter 38; the remaining coordinates are the displayed finite history, transport, continuation, provenance, and local naming rows.

The three local gates make these coordinates exhaustive. Definition 452.145 exposes D before W, Definition 452.146 factors the Real-side read through D, W, S, and Definition 452.147 denies quotient streams, host limits, countable choice, open-cover compactness, and ambient Real completeness.

Thus any admitted consumer read is already a read of the four named faces plus the displayed BEDC support rows. Since the carrier contains no further coordinate and the nonescape theorem denies the standard off-packet exits, there is no extra source available to the radius-window packet. □

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowNoExtraSource

Theorem 452.149 (Finite Lebesgue-number Phase Real radius-window face minimality)**.** *The four source faces of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *Phase Real radius-window packet are minimal for the exit route. Removing the* DyadicRatCore$^{\uparrow}$ *face loses the endpoint and radius row D; removing the* StreamName$^{\uparrow}$ *face loses the finite-window row W; removing the* RegSeqRat$^{\uparrow}$ *face loses the regularity and classifier row in S; and removing the* Real$^{\uparrow}$ *face loses the terminal seal row read through S.*

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowFaceMinimality

Proof. For the dyadic face, use Definition 452.145. The theorem makes the DyadicRatCore$^{\uparrow}$ row a prerequisite for consuming the finite window, so deleting that face removes the endpoint and radius read needed before W can be inspected.

For the stream face, the same admission theorem identifies W as the finite window available to downstream compact, continuous, totally bounded, and uniform consumers. Without StreamName$^{\uparrow}$, the packet still has a dyadic radius row, but it has no finite-window row to present at the consumer boundary.

For the regular-rational and Real faces, use Definition 452.146. The readback factors Real-side radius access through D, W, S, where S records the RegSeqRat$^{\uparrow}$-to-Real$^{\uparrow}$ seal route. Removing RegSeqRat$^{\uparrow}$ deletes the regularity and classifier part of S, while removing Real$^{\uparrow}$ deletes the terminal seal part of the same row.

Definition 452.148 supplies the converse exclusion needed for minimality: none of these losses can be repaired by an off-packet source. Hence each of the four faces is necessary for the corresponding radius-window row. □

Theorem 452.150 (Finite Lebesgue-number Phase Real radius-window exit consumer readiness)**.** *The accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real radius-window packet is ready for downstream compact, continuous, totally bounded, compact-net, and uniform consumers. Such a consumer may use the dyadic admission row, the Real seal readback row, the open Phase nonescape boundary, the no-extra-source boundary, and the four-face minimality boundary as one finite radius ledger, without quotient streams, selected limits, host real equality, ambient Real completeness, metric-completeness, or countable choice.*

Proof. Assemble the positive ledger first. Definition 452.145 provides the ordered dyadic-to-window read, and Definition 452.146 routes the Real seal through D, W, S with the shared BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next attach the boundary conditions. Definition 452.147 keeps every admitted consumer read inside the finite packet, while Definition 452.148 states that no fifth source face is readable from that packet.

Finally use Definition 452.149 to make the ledger exit-ready rather than merely admissible. Each of the four source faces is required for its corresponding endpoint, finite-window, regularity, classifier, or seal row, and the nonescape boundary prevents replacement by quotient streams, selected limits, host real equality, ambient Real completeness, metric-completeness, or countable choice. This is exactly the finite radius ledger needed by downstream Phase Real consumers. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowExitConsumerReadi`

Theorem 452.151 (Finite Lebesgue-number Phase Real local radius determinacy)**.** *Let two admitted downstream reads of an accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real exit consumer use the same rows*

$$E, \quad R, \quad D, \quad W, \quad S, \quad H, \quad C, \quad P, \quad N$$

from Definition 452.144. Then the two reads expose the same local radius-window packet $\mathcal{W}_{\lambda}^{\mathrm{PhaseReal}}$*. In particular, the consumer-visible dyadic radius, finite stream window, regular-rational seal route, Real seal row, history transport, continuation replay, provenance row, and local* $\mathsf{NameCert}$ *row are fixed by the displayed packet.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealLocalRadiusDeterminacy`

Proof. Start from the carrier definition. The Phase Real radius-window packet is exactly the finite BHist tuple $(E, R, D, W, S, H, C, P, N)$, so equality of the displayed rows identifies the packet before any consumer interpretation is applied.

The positive readback gates make the identification local at the two consumer reads. By Definition 452.145, both reads must expose the same dyadic radius row D before the same finite window row W. By Definition 452.146, the Real-side radius access in both reads factors through the same ordered D, W, S rows with the same support.

It remains to rule out a hidden source of disagreement. Definition 452.148 denies any readable source beyond the four named faces and the support rows, while Definition 452.149 makes those four faces necessary for the exit route. Hence two accepted reads with the same packet rows cannot expose distinct local radius-window data. □

Theorem 452.152 (Finite Lebesgue-number Phase Real radius-window source lock)**.** *Every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real radius-window exit read has its source locked to the four faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The rows D, W, S, H, C, P, N of Definition 452.144 may be consumed only through these faces and their displayed finite BHist, hsame, Cont, Pkg, *and* NameCert *support.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusWindowSourceLock`

Proof. The carrier definition names exactly the four source chapters DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ for the radius-window packet. The remaining entries are not new sources; they are the finite history, transport, continuation, provenance, and naming rows used to read those faces.

The admission and readback results fix the order of access. Definition 452.145 locks the finite window behind the dyadic radius row, and Definition 452.146 locks the Real-side seal behind the ordered D, W, S route.

Finally, Definition 452.148 excludes a fifth readable source, and Definition 452.149 shows that none of the four faces can be removed from the exit route. The accepted read is therefore source-locked to precisely the displayed four-face packet. □

Theorem 452.153 (Finite Lebesgue-number Phase Real radius consumer nonescape)**.** *Compact, continuous, totally bounded, compact-net, and uniform consumers of an accepted* FiniteLebesgueNumber$^{\uparrow}$ *Phase Real radius-window exit read cannot create an additional radius source. Any radius data they expose is already read from the packet*

$$E, \quad R, \quad D, \quad W, \quad S, \quad H, \quad C, \quad P, \quad N$$

of Definition 452.144, with source locked to DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealRadiusConsumerNonescape`

Proof. Use the consumer-readiness theorem as the entry point. Definition 452.150 packages the dyadic admission row, the Real seal readback row, the open Phase nonescape boundary, the no-extra-source boundary, and the four-face minimality boundary as one finite radius ledger for these consumers.

The local determinacy theorem then fixes repeated reads of that ledger. By Definition 452.151, two accepted consumer reads with the same rows expose the same local radius-window packet. By Definition 452.152, that packet is locked to the four named source faces and their support rows.

Thus a compact, continuous, totally bounded, compact-net, or uniform consumer has no constructor site for an additional radius source. Any consumer-visible radius row is already one of the displayed packet rows, and the exclusions in Definition 452.148 prevent quotient streams, selected limits, host real equality, ambient Real completeness, metric-completeness, or countable choice from supplying another source. □

Definition 452.154 (Finite Lebesgue-number Phase Real audit ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *root route, the Phase Real audit ledger is the finite* BHist *packet*

$$\mathcal{L}_{\lambda}^{\mathrm{audit}} = (D, B, M, E, S, Q, R, K, T, F, U, H, C, P, N)$$

whose entries are the DyadicCover$^{\uparrow}$ *cover row D, the* RationalBall$^{\uparrow}$ *positive-radius row B, the* DyadicMesh$^{\uparrow}$ *mesh row M, the* DyadicRatCore$^{\uparrow}$ *core-radius row E, the* StreamName$^{\uparrow}$ *finite-window row S, the* RegSeqRat$^{\uparrow}$ *regular-rational row Q, the* Real$^{\uparrow}$ *seal row R, the* CompactMetric$^{\uparrow}$ *compact consumer row K, the* TotallyBounded$^{\uparrow}$ *compact-net admission row T, the* ContinuousMap$^{\uparrow}$ *endpoint row F, the* UniformModulus$^{\uparrow}$ *endpoint row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and compatible local* NameCert *naming N. The ledger is accepted*

only when these rows read one positive radius through the same NameCert route over the four open Phase source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealAuditLedger`

Theorem 452.155 (Finite Lebesgue-number Phase Real audit-ledger source)**.** *Every accepted Phase Real audit-ledger read for* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *factors its radius source through the four open Phase faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The dyadic-cover, rational-ball, and dyadic-mesh rows of Definition 452.154 are source rows for this route, while the compact, totally bounded, continuous-map, and uniform-modulus rows are consumers of the same finite packet.

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealAuditLedgerSource`

Proof. Begin with the root radius-window obligation surface. Definition 452.80 confines the visible positive-radius and finite-window obligations to the dyadic cover, rational-ball, mesh, stream-name, regular-rational, Real, history, transport, replay, provenance, and naming rows.

Now attach the open Phase root-unblock route. Definition 452.133 exposes the same root radius through the four source faces, and Definition 452.137 says that an admissible root-radius export is exactly the finite packet over those faces and the support rows.

Finally use the exit and window certificates. Definition 452.90, Definition 452.140, Definition 452.141, and Definition 452.143 make the four-face route covering, deterministic, and nonescaping. The Phase Real radius-window face minimality theorem then identifies these faces as the necessary source faces for the accepted audit ledger, so every accepted read factors through them. □

Theorem 452.156 (Finite Lebesgue-number Phase Real audit-ledger scope)**.** *The scoped support of an accepted Phase Real audit ledger for* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *is exactly the displayed rows*

$$\mathsf{DyadicCover}^{\uparrow}, \mathsf{RationalBall}^{\uparrow}, \mathsf{DyadicMesh}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{Totall}$$

It has no coordinate for open-cover compactness, selected subcovers, quotient streams, host minima, selected limits, countable choice, detached $\mathsf{Real}^{\uparrow}$ *completeness, external modulus data, or an unbudgeted radius selector.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealAuditLedgerScope`

Proof. Read the ledger definition first. The only support rows named by $\mathcal{L}_{\lambda}^{\mathrm{audit}}$ are the dyadic cover, rational ball, mesh, four Phase Real source faces, compact and uniform consumer endpoints, finite history, transport, replay, provenance, and local naming rows.

The existing audit and window packets make this support exhaustive. Definition 452.105 identifies the current Real and Completion route with the dyadic cover, rational-ball, mesh, four-face, history, transport, replay, provenance, and naming rows. Definition 452.149 says that none of the four faces can be removed while preserving its corresponding radius-window row.

The nonescape boundaries remove the off-table rows. Definition 452.107 excludes open-cover compactness, selected subcovers, quotient equality, choice, external modulus data, selected limits, and detached Real completeness from the audit packet. Definition 452.139 keeps downstream consumers inside the same Phase Real and Completion packet, so host minima, quotient streams, selected limits, and unbudgeted radius selection have no scope row in the accepted ledger. □

Theorem 452.157 (Finite Lebesgue-number Phase Real audit-ledger readiness)**.** *The Phase Real audit ledger of Definition 452.154 is ready for* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, TotallyBounded$^{\uparrow}$, *compact-net, and* UniformModulus$^{\uparrow}$ *consumers. Each of these consumers reads the same finite ledger through the root radius-window obligation surface, the open Phase root-unblock coverage, the root-radius export, the L10 consumer exhaustion, the four-face exit package, and the Phase Real radius-window face minimality boundary; none receives a private radius source.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealAuditLedgerReadiness`

Proof. Assemble the source side first. Definition 452.155 routes every accepted audit-ledger read through the four open Phase faces, and Definition 452.156 fixes the support rows available to the route.

Now attach the downstream endpoints. Definition 452.98 places compact, compact-net, and uniform consumers on the same four-face packet. Definition 452.139 extends the same exhaustion to compact, continuous, totally bounded, compact-net, and uniform radius reads along the Phase Real and Completion route.

The readiness claim follows from the absence of a second source coordinate. The root radius-window obligation surface, the open Phase root-unblock radius coverage, the root-radius export exactness, the four-face exit coverage, determinacy, and nonescape theorems, and the Phase Real radius-window face minimality theorem leave the consumers with precisely the rows of $\mathcal{L}_{\lambda}^{\mathrm{audit}}$. Since the scoped ledger excludes open-cover compactness, selected subcovers, quotient streams, host minima, selected limits, countable choice, detached Real completeness, external modulus data, and unbudgeted radius selection, no consumer can read a private radius source. □

452.1 Finite Lebesgue-number Phase Real terminal radius readiness

Definition 452.158 (Finite Lebesgue-number Phase Real terminal radius readiness ledger)**.** *For an accepted* FiniteLebesgueNumber$^{\uparrow}$ *route, the Phase Real terminal radius readiness ledger is the finite* BHist *packet*

$$\mathcal{T}_{\lambda}^{\mathrm{ready}} = (D, B, M, E, S, Q, R, K, F, T, U, H, C, P, N)$$

whose entries are the DyadicCover$^{\uparrow}$ *cover row D, the* RationalBall$^{\uparrow}$ *positive-radius row B, the* DyadicMesh$^{\uparrow}$ *mesh row M, the* DyadicRatCore$^{\uparrow}$ *core-radius face E, the* StreamName$^{\uparrow}$ *window face S, the* RegSeqRat$^{\uparrow}$ *regular-rational face Q, the* Real$^{\uparrow}$ *endpoint face R, the* CompactMetric$^{\uparrow}$ *compact consumer row K, the* ContinuousMap$^{\uparrow}$ *continuous endpoint row F, the* TotallyBounded$^{\uparrow}$ *finite-net consumer row T, the* UniformModulus$^{\uparrow}$ *uniform endpoint row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and compatible local* NameCert *naming N.*

The ledger is accepted only when the compact, continuous, totally bounded, compact-net, and uniform consumers read one terminal positive-radius verdict through the same dyadic and four-face Phase Real packet. In particular, the verdict is carried by the displayed DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *faces and by the same* BHist, hsame, Cont, Pkg, *and* NameCert *support rows; it is not a host minimum, an open-cover compactness row, a selected subcover, a quotient stream equality, a selected limit, countable choice, detached* Real$^{\uparrow}$ *completeness, or an external modulus datum.*

Lean4 (definition) `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealTerminalRadiusReadiness`

Theorem 452.159 (Finite Lebesgue-number Phase Real terminal radius readiness)**.** *Every accepted terminal radius consumer for* FiniteLebesgueNumber$^{\uparrow}$ *factors through the ledger of Definition 452.158. Equivalently, each* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*, compact-net, or* UniformModulus$^{\uparrow}$ *consumer visible at the Phase Real terminal boundary receives the same finite packet*

$$D, B, M, E, S, Q, R, K, F, T, U, H, C, P, N$$

and receives no private positive-radius source outside that packet.

Proof. Start with the root-unblock surface. Definition 452.133 puts every accepted root radius read inside the finite dyadic cover, rational-ball, dyadic-mesh, four-face Real, history, transport, replay, provenance, and naming packet, while Definition 452.137 identifies admissible root-radius export with displaying exactly that packet.

Next attach the terminal consumer rows. Definition 452.139 exhausts compact, continuous, totally bounded, and uniform radius reads by the same Phase Real and Completion packet. Definition 452.140 adds the compact-net consumer to the same four-face exit package, so every listed endpoint reads the displayed dyadic and Phase Real rows before any consumer verdict is available.

Finally use the audit readiness boundary. Definition 452.157 states that compact, continuous, totally bounded, compact-net, and uniform consumers read the same finite ledger through the root radius-window obligation surface, open Phase root-unblock coverage, root-radius export, L10 consumer exhaustion, four-face exit package, and Phase Real radius-window face minimality boundary. These are precisely the rows of $\mathcal{T}_{\lambda}^{\mathrm{ready}}$, so every accepted terminal consumer factors through the readiness ledger and cannot add a private positive-radius source. □

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealTerminalRadiusReadiness`

Theorem 452.160 (Finite Lebesgue-number Phase Real terminal radius stability)**.** *Componentwise* hsame *transport and displayed* Cont *replay preserve the terminal radius verdict of Definition 452.158 exactly when they preserve the displayed* DyadicCover$^{\uparrow}$*,* RationalBall$^{\uparrow}$*,* DyadicMesh$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* Pkg*, and* NameCert *rows componentwise. Under such transport, the compact, continuous, totally bounded, compact-net, and uniform consumers read the same terminal verdict.*

Proof. First reduce the verdict to the root packet. By Definition 452.137, an admissible root-radius export is exactly the finite packet over the dyadic cover, rational-ball, mesh, four source faces, and BEDC support rows. Definition 452.138 then says that componentwise hsame transport and displayed Cont replay preserve the positive-radius row only together with its rational-ball, mesh, stream, regular-rational, and Real rows.

Next compare the consumer endpoints. Definition 452.141 makes agreement on the four source faces and the BHist, hsame, Cont, Pkg, and NameCert rows determine the downstream radius classifier seen by compact, continuous, totally bounded, compact-net, and uniform consumers. Thus componentwise transport of the rows listed in $\mathcal{T}_{\lambda}^{\mathrm{ready}}$ transports the consumer-visible verdict as one packet.

The converse is forced by nonescape. If the terminal verdict were preserved while one displayed source or support row changed, the endpoint would require an unrecorded replacement coordinate for the missing row. Definition 452.143 and Definition 452.156 exclude precisely those outside coordinates. Hence terminal verdict stability is exactly componentwise hsame and displayed Cont stability of the ledger rows. □

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealTerminalRadiusStability

Theorem 452.161 (Finite Lebesgue-number Phase Real terminal radius nonescape)**.** *The Phase Real terminal radius readiness ledger exports no open-cover compactness row, selected-subcover row, host-minimum row, quotient stream equality row, selected-limit row, countable-choice row, detached* $\mathsf{Real}^{\uparrow}$ *completeness row, or external-modulus row. Any compact, continuous, totally bounded, compact-net, or uniform-modulus terminal use of such a row lies outside the accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *terminal radius route.*

Proof. Read the positive source first. Definition 452.134 removes open-cover compactness, selected subcovers, host minima, quotient stream equality, countable choice coordinates, detached $\mathsf{Real}^{\uparrow}$ completeness, and external modulus data from the open Phase root-unblock obligation packet.

Now pass through the four-face exit package. Definition 452.143 excludes open-cover compactness, selected subcovers, host minima, quotient stream equality, selected limits, choice rows, and detached $\mathsf{Real}^{\uparrow}$ completeness from the four-face radius exit. Definition 452.139 keeps compact, continuous, totally bounded, and uniform consumers inside the same Phase Real and Completion packet.

Finally attach the terminal ledger. Definition 452.156 lists exactly the dyadic, four-face Real, compact, continuous, totally bounded, uniform, history, transport, replay, provenance, and naming rows available to the audit route, and Definition 452.159 makes every accepted terminal consumer factor through those rows. Since none of the excluded objects has a coordinate in $\mathcal{T}_{\lambda}^{\mathrm{ready}}$, no accepted terminal consumer can export or inspect such an object. □

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealTerminalRadiusNonescape

Theorem 452.162 (Finite Lebesgue-number Phase Real terminal radius choice-free finality)**.** *For* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$*, every accepted Phase Real terminal radius verdict is final exactly as the finite packet* $D, B, M, E, S, Q, R, K, F, T, U, H, C, P, N$ *of Definition 452.158. If another accepted terminal verdict is visible at the same boundary, then it agrees componentwise on the* $\mathsf{DyadicCover}^{\uparrow}$ *cover row,* $\mathsf{RationalBall}^{\uparrow}$ *positive-radius row,* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *core-radius face,* $\mathsf{StreamName}^{\uparrow}$ *window face,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational face,* $\mathsf{Real}^{\uparrow}$ *endpoint face,* $\mathsf{CompactMetric}^{\uparrow}$ *compact consumer row,* $\mathsf{ContinuousMap}^{\uparrow}$ *continuous endpoint row,* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row,* $\mathsf{UniformModulus}^{\uparrow}$ *uniform endpoint row,* BHist *transport ledger, displayed continuation row,* Pkg *provenance row, and compatible* $\mathsf{NameCert}$ *row. No compact, continuous, totally bounded, compact-net, or uniform endpoint may refine that final verdict by open-cover compactness, selected subcover, host minimum, quotient stream equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or an external modulus.*

Lean4 ✓ BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberPhaseRealTerminalRadiusChoiceFreeFinali

Proof. First force every terminal verdict through the readiness packet. By Definition 452.159, an accepted compact, continuous, totally bounded, compact-net, or uniform endpoint reads the terminal positive-radius verdict only through the displayed rows $D, B, M, E, S, Q, R, K, F, T, U, H, C, P, N$.

Now compare two accepted terminal verdicts. Definition 452.160 says that preservation of the terminal verdict is exactly componentwise preservation of the dyadic cover, rational ball, dyadic mesh, DyadicRatCore, StreamName, RegSeqRat, Real, compact, continuous, totally bounded, uniform, history, provenance, and NameCert rows together with the displayed transport and continuation data. Thus two accepted terminal verdicts at the same boundary agree precisely when these coordinates agree.

The remaining visible classifier is unique. Definition 452.142 supplies the total visible root-radius classifier for the open Phase boundary, so there is no second consumer-facing classifier through which the same terminal packet could be reread as a different verdict.

Finally exclude all proposed refinements outside the packet. Definition 452.161 refuses open-cover compactness, selected subcover, host minimum, quotient stream equality, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, and external modulus rows at this terminal boundary. Hence the accepted terminal radius verdict is final in the choice-free Phase Real sense: every second accepted verdict is the same componentwise packet, and every outside endpoint refinement is unavailable to the route. □

Theorem 452.163 (Finite Lebesgue-number terminal-readiness face projection)**.** *Every accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *terminal readiness ledger projects componentwise to the four displayed Phase Real exit faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and to the support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The projection keeps the dyadic-cover, rational-ball, dyadic-mesh, compact, continuous, totally bounded, compact-net, and uniform endpoint rows only as coordinates of the same finite terminal packet; it does not introduce a second radius source.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalReadinessFaceProjection`

Proof. Read the terminal ledger first. Definition 452.159 makes every accepted compact, continuous, totally bounded, compact-net, or uniform terminal consumer factor through the displayed packet $D, B, M, E, S, Q, R, K, F, T, U, H, C, P, N$.

The four source faces are then forced by the exit package. Definition 452.140 covers the accepted root radius read by the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, and Definition 452.108 identifies those faces together with BHist, hsame, Cont, Pkg, and NameCert as the exact source packet for consumer-visible reads.

Finally compare transported ledgers. Definition 452.160 says terminal verdict stability is exactly componentwise stability of these displayed coordinates. Hence every accepted terminal readiness ledger has the stated componentwise projection, and the consumer rows remain downstream coordinates of the same packet rather than private positive-radius sources. □

Theorem 452.164 (Finite Lebesgue-number terminal-readiness exit handoff)**.** *The componentwise projection of Definition 452.163 is exactly the Phase Real and Completion exit handoff consumed by the* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, compact-net, and* $\mathsf{UniformModulus}^{\uparrow}$ *endpoints. At the terminal boundary, these endpoints receive the same four-face radius packet and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalReadinessExitHandoff`

Proof. First identify the packet being handed off. The face projection theorem above gives the four Phase Real and Completion faces and the BEDC support rows as componentwise coordinates of the accepted terminal readiness ledger.

Next attach the endpoint consumers. Definition 452.139 exhausts compact, continuous, totally bounded, compact-net, and uniform radius reads by the same Phase Real and Completion packet, while Definition 452.141 makes agreement on the four source faces and support rows determine the downstream radius classifier.

The remaining boundary is the visible classifier. Definition 452.142 supplies the total root-radius classifier for the open Phase boundary. Therefore the projected terminal readiness packet is not merely compatible with the endpoints: it is the exact exit handoff those endpoints consume. □

Theorem 452.165 (Finite Lebesgue-number terminal-readiness no-completeness promotion)**.** *The terminal-readiness exit handoff of Definition 452.164 cannot promote to an open-cover compactness row, a selected subcover, a host minimum, a quotient stream equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, metric completeness, or an external modulus. Any such datum lies outside the accepted* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *Phase Real terminal radius route.*

Lean4 ✓ `BEDC.Derived.FiniteLebesgueNumberUp.FiniteLebesgueNumberTerminalReadinessNoCompletenessPromotio`

Proof. Start with the terminal boundary. Definition 452.161 excludes open-cover compactness, selected subcovers, host minima, quotient stream equality, selected limits, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, and external modulus rows from accepted terminal radius use.

The four-face exit gives the same refusal before the handoff reaches any endpoint. Definition 452.143 removes open-cover compactness, selected subcovers, host minima, quotient stream equality, selected limits, choice rows, and detached $\mathsf{Real}^{\uparrow}$ completeness from the Phase Real radius exit itself.

Metric completeness and external moduli have no independent coordinate in the projected packet. The handoff theorem above passes only the four Phase Real and Completion faces plus BHist, hsame, Cont, Pkg, and NameCert support rows to the compact, continuous, totally bounded, compact-net, and uniform endpoints. Thus the handoff is a finite BEDC exit packet, not a compactness, completeness, or external-modulus surrogate. □

Chapter 453

A Concrete Naming Certificate for $\mathsf{DoCalculus}^{\uparrow}$

Following Pearl, *Causality*, 2009, the intervention calculus used here is recorded as a finite certificate surface rather than as an external semantic oracle. The $\mathsf{DoCalculus}^{\uparrow}$ packet keeps the intervention ledger, observed variable rows, adjustment-set witnesses, distribution rows, conditional independence certificates, and conditional-expectation handoff rows visible as BEDC data. It does not claim a host probability space, a full structural causal model, graphical completeness of do-calculus, or a selected counterfactual semantics.

The finite intervention ledger consumes already certified probability-facing rows rather than creating a host causal model. Its probability-space export surface is the one isolated by chapter 184; its conditional independence certificates are read through chapter 187; its displayed distribution handoff is the surface of chapter 186; and its conditional-expectation row is routed through chapter 188. Thus $\mathsf{DoCalculus}^{\uparrow}$ only packages finite intervention reads through those certified probability-facing rows.

Definition 453.1 (DoCalculus carrier)**.** *A* $\mathsf{DoCalculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, V, A, M, R, J, E, H, C, P, N)$$

with the following visible coordinates:

1. *I is the finite intervention ledger, whose rows name the intervened variables, the displayed intervention marks, and the finite order in which the intervention rows are read;*
2. *V is the finite observed-variable row, listing the outcome, treatment, covariate, and auxiliary observation names that are visible to the certificate;*
3. *A is the adjustment-set ledger, containing only finite rows that name which observed variables are admitted as adjustment coordinates for the displayed query;*
4. *M records the intervention-modified distribution rows, including the pre-intervention observational row, the post-intervention row, and the finite conditioning rows exposed to consumers;*
5. *R records conditional-independence certificates used by the intervention ledger and the adjustment-set ledger;*
6. *J is the conditional-expectation handoff row, whose input is the displayed adjusted distribution row and whose output is the query-facing expectation row;*

7. *E records the probability-facing export row consumed by downstream probability chapters;*
8. *H records the componentwise* hsame *transports for intervention rows, observed-variable rows, adjustment ledgers, distribution rows, independence certificates, conditional-expectation rows, and probability exports;*
9. *C records the displayed* Cont *routes by which consumers read intervention admission, adjustment-set membership, independence certificates, distribution handoffs, and conditional-expectation exports;*
10. *P records* Pkg *provenance over* BHist, hsame, Cont, *finite variable rows, finite ledger rows, probability handoff rows, and local naming rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DoCalculus}^{\uparrow}}$ *provenance row.*

The carrier accepts exactly finite intervention-ledger packets whose consumers can read every causal query through the displayed adjustment, independence, distribution, and expectation rows.

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusUpTasteGate single carrier alignment`

Theorem 453.2 (DoCalculus NameCert obligation surface)**.** *Every accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet with carrier Definition 453.1 carries five local naming obligations:*

1. *carrier admission: the tuple $D = (I, V, A, M, R, J, E, H, C, P, N)$ is admitted as one finite* BHist *packet;*
2. *classifier transport: componentwise* hsame *and* Cont *transport preserves the intervention, observed-variable, adjustment, distribution, independence, expectation, package, and naming rows;*
3. *adjustment ledger exactness: the adjustment row A lists exactly the observed coordinates allowed to mediate the displayed intervention query;*
4. *intervention non-escape: every consumer-visible intervention read passes through $I, V, A, M, R, H, C, P, N$ and cannot project an unlisted intervention, hidden counterfactual, or host causal model;*
5. *probability handoff: every probability-facing read passes through the conditional-expectation row J and the export row E after the finite adjustment and independence rows have been displayed.*

These five rows are the complete local $\mathsf{NameCert}_{\mathsf{DoCalculus}^{\uparrow}}$ *surface for the finite intervention calculus packet.*

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket namecert obligation surface`

Proof. Project the accepted carrier. Its coordinates are the finite rows $I, V, A, M, R, J, E, H, C, P, N$, so carrier admission is just the displayed BHist packet.

For classifier transport, move each coordinate through its stored hsame row and then read it through the corresponding Cont route. Because the carrier is componentwise, no transported row can introduce a variable, intervention, distribution, or handoff coordinate not already listed in the tuple.

Adjustment exactness is read from A: it names only observed variables from V, and the distribution rows in M consume exactly those names for the displayed query. The intervention non-escape clause follows because every intervention-facing route has to pass through I and the same finite ledger rows before a consumer can read it.

Finally, probability-facing data are exported only through J and E. The conditional-independence certificates in R and the adjustment ledger A are therefore visible before the expectation handoff is consumed, which keeps the handoff finite and internal to the packet. □

Theorem 453.3 (DoCalculus adjustment ledger exactness)**.** *For every accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet* $D = (I, V, A, M, R, J, E, H, C, P, N)$*, the adjustment-set ledger* A *is exhausted by the finite observed-variable rows in* V *and by the conditional-independence certificates in* R *that justify their use for the displayed intervention query. No variable outside* V *and no unstored independence assertion can be read as part of the adjustment set.*

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket adjustment ledger exactness`

Proof. Read the observed-variable coordinate V. It gives the finite list of outcome, treatment, covariate, and auxiliary observation names available to the packet.

Next read the adjustment ledger A. Each adjustment entry is indexed by one of those observed names and points to an independence certificate in R used by the displayed query.

Thus the adjustment surface is exhausted by V, A, R, H, C, P, N. A hidden variable or unstored independence assertion would require a coordinate absent from Definition 453.1, so it cannot be projected from the accepted packet. □

Theorem 453.4 (DoCalculus intervention non-escape)**.** *Every intervention-facing consumer of an accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet* $D = (I, V, A, M, R, J, E, H, C, P, N)$ *factors through the intervention ledger* I*, the observed-variable row* V*, the adjustment ledger* A*, the distribution rows* M*, the independence certificates* R*, the transport rows* H*, the continuation routes* C*, the package provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*. Such a consumer cannot obtain a hidden intervention, an unlisted causal graph, a counterfactual selector, or a host structural causal model from the packet.*

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket intervention non escape`

Proof. Use the carrier definition to inspect an intervention read. The read begins at the finite ledger I, and each intervention mark is paired with visible observed-variable data in V.

The route then passes through the adjustment ledger, the modified distribution rows, and the named independence certificates. Transport and consumer access are only the displayed H and C rows, with provenance kept in P and N.

The excluded objects have no coordinate in the carrier. Since consumer routes are finite projections from the displayed packet, none of those objects can escape through an intervention-facing read. □

Theorem 453.5 (DoCalculus probability handoff)**.** *Every probability-facing read from an accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet factors through the adjusted distribution rows* M*, the conditional-independence certificates* R*, the conditional-expectation handoff row* J*, and the export row* E*. The handoff is a finite certificate route into probability chapters, not a claim of host measure theory, graphical completeness, or selected counterfactual semantics.*

Proof. Project the distribution coordinate M and the independence coordinate R. They display the observational, post-intervention, and conditioning rows available to the query together with the finite certificates that justify the adjustment route.

Read the conditional-expectation handoff J. Its input is the adjusted distribution row already exposed by M and R, and its output is the probability-facing row later exported by E.

Therefore the probability chapter receives only the finite handoff packet M, R, J, E, H, C, P, N. No ambient probability space, completeness theorem for do-calculus, or counterfactual selector is supplied by this route. □

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket probability handoff`

Definition 453.6 (DoCalculus displayed prefix subledger)**.** *Let $D = (I, V, A, M, R, J, E, H, C, P, N)$ be an accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet as in Definition 453.1. A displayed prefix subledger of D is a finite packet*

$$D_{\leq k} = (I_{\leq k}, V_{\leq k}, A_{\leq k}, M_{\leq k}, R_{\leq k}, J_{\leq k}, E_{\leq k}, H_{\leq k}, C_{\leq k}, P_{\leq k}, N_{\leq k})$$

obtained by retaining an initial segment $I_{\leq k}$ of the displayed finite order on the intervention ledger I and retaining exactly the observed-variable, adjustment, distribution, independence, conditional-expectation, export, transport, continuation, package, and local naming rows whose route entries are used by intervention reads from that initial segment. Rows of I strictly after the displayed prefix are called later intervention rows for $D_{\leq k}$.

Lean4 (definition) `BEDC.Derived.DoCalculusUp.DoCalculusDisplayedPrefixSubledger`

Theorem 453.7 (DoCalculus intervention prefix locality)**.** *Let $D = (I, V, A, M, R, J, E, H, C, P, N)$ be an accepted* $\mathsf{DoCalculus}^{\uparrow}$ *packet, and let $D_{\leq k}$ be a displayed prefix subledger of D in the sense of Definition 453.6. Every intervention-facing consumer whose displayed intervention mark lies in $I_{\leq k}$ factors through*

$$I_{\leq k}, V_{\leq k}, A_{\leq k}, M_{\leq k}, R_{\leq k}, H_{\leq k}, C_{\leq k}, P_{\leq k}, N_{\leq k}.$$

In particular, such a consumer cannot read any later intervention row of I.

Proof. By Definition 453.1, the intervention ledger I is finite and carries a displayed order for the intervention rows. The prefix subledger retains an initial segment $I_{\leq k}$ of that order together with exactly the route rows attached to interventions in that segment.

Apply the intervention non-escape theorem Definition 453.4 to the accepted packet D. Any intervention-facing read in D factors through the displayed intervention ledger, observed-variable row, adjustment ledger, distribution rows, independence certificates, transport rows, continuation routes, package provenance, and local naming row:

$$I, V, A, M, R, H, C, P, N.$$

For a read whose intervention mark lies in the prefix $I_{\leq k}$, the definition of displayed prefix subledger says that the relevant observed-variable, adjustment, distribution, independence, transport, continuation, package, and naming entries are exactly the retained rows

$$V_{\leq k}, A_{\leq k}, M_{\leq k}, R_{\leq k}, H_{\leq k}, C_{\leq k}, P_{\leq k}, N_{\leq k}.$$

Thus restricting the non-escape factorization to the displayed prefix gives the stated factorization through the retained prefix rows.

It remains to exclude later intervention rows. A later row is, by definition, a row of I outside the initial segment $I_{\leq k}$. Since the consumer route has already been restricted to the retained prefix route family $C_{\leq k}$ and its retained provenance rows, no projection in that route has a source coordinate in the later part of I. Reading such a later row would require leaving the retained prefix packet and using an intervention coordinate not present in $D_{\leq k}$. This is impossible for an intervention-facing read that factors through the displayed prefix subledger. Therefore no later intervention row can be read. □

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket intervention prefix locality`

Theorem 453.8 (DoCalculus scoped closure route)**.** *The displayed* DoCalculus$^{\uparrow}$ *packet closes the following scoped surface: finite* BHist *intervention ledgers, componentwise* hsame *transport, displayed* Cont *replay, adjustment ledgers, intervention non-escape rows, probability-facing handoff rows, displayed prefix subledgers, prefix-locality routes,* Pkg *provenance, and local* NameCert *rows. The scope is witnessed exactly by Definition 453.1, Definition 453.2, Definition 453.3, Definition 453.4, Definition 453.5, Definition 453.6, and Definition 453.7.*

Within this scope, every intervention-facing read is a finite packet route through the displayed causal ledger and its retained probability-facing rows. No read in the scoped surface supplies a hidden intervention, host causal graph, selected counterfactual, ambient probability equality, or adjustment row outside the finite ledger.

Proof. Start from the carrier of Definition 453.1. It lists exactly the intervention ledger, observed-variable rows, adjustment ledger, modified distribution rows, independence certificates, conditional-expectation row, export row, transport row, continuation row, package provenance, and local naming row. The NameCert obligation surface of Definition 453.2 then identifies carrier admission, classifier transport, adjustment exactness, intervention non-escape, and probability handoff as the local naming obligations.

The adjustment ledger is bounded by Definition 453.3: only observed rows from V and stored independence certificates from R can mediate the displayed query. The intervention-facing side is bounded by Definition 453.4, which forces every consumer route through $I, V, A, M, R, H, C, P, N$ before any intervention read is visible.

The probability-facing side is bounded by Definition 453.5. It routes adjusted distribution rows and independence certificates through the conditional-expectation row and the export row, so the probability chapter receives a finite handoff rather than an ambient model.

Finally use the prefix interface. The displayed subledger of Definition 453.6 retains an initial intervention segment together with exactly its dependent observed-variable, adjustment, distribution, independence, transport, continuation, package, and naming rows. The locality theorem Definition 453.7 says that reads from the retained prefix cannot inspect later intervention rows. These witnesses assemble the stated scoped closure over BHist, hsame, Cont, finite intervention ledgers, probability-facing rows, and NameCert. □

Lean4 ✓ `BEDC.Derived.DoCalculusUp.DoCalculusPacket scoped closure route`

Closure status for NameCert$_{\text{DoCalculus}^{\uparrow}}$.

Constructive story: A DoCalculus$^{\uparrow}$ packet is generated from a finite intervention ledger, observed-variable rows, adjustment-set ledger, modified distribution rows, conditional-independence certificates, conditional-expectation rows, export rows, componentwise hsame transport, displayed Cont routes, package provenance, and local naming. The classifier reads an intervention only through the same finite causal ledger and its adjustment and probability-facing handoff rows. The prefix subledger story is constructive locality: an initial intervention segment generates its own retained packet, and later intervention rows cannot be read through routes that were restricted to the displayed prefix. No causal oracle or hidden intervention is introduced.

Theory closure: TheoryGate(NameCert$_{\text{DoCalculus}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 453.8 binds the DoCalculus carrier, NameCert obligation surface, adjustment ledger exactness, intervention non-escape, probability handoff, displayed prefix subledger, and intervention prefix locality for DoCalculus$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{DoCalculus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.DoCalculusUp.DoCalculusPacket\_scoped\_closure\_route`

Bridge status: none

Origin: human

Not claimed: It does not claim hidden interventions, host graphical causal semantics, causal oracles, external probability equality, or adjustment rows outside the finite ledger.

Upgrade path: To reach obligationClosure, the intervention ledger and prefix-locality results must be assembled into a checked finite-carrier closure theorem.

Chapter 454

LocatedModulusCompletion NameCert Construction

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedModulusCompletion}^{\uparrow}$ interface records a completion-side read of a located real name only through explicit regular Cauchy data, a displayed modulus, and a finite limit seal. It belongs to the same Real and Completion phase as $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$: the packet exposes finite readback and seal rows, not a quotient of Cauchy streams or a selected host real value.

The packet consumes the regular source of chapter 301, the real seal of chapter 38, the modulus row of chapter 292, the limit-facing seal of chapter 386, and the located evidence of **??**. The located modulus row is not an ambient completeness principle: it transports finite $\mathsf{StreamName}^{\uparrow}$ windows through $\mathsf{RegSeqRat}^{\uparrow}$ regularity and $\mathsf{Real}^{\uparrow}$ seal rows, with $\mathsf{CauchyModulus}^{\uparrow}$ supplying the displayed threshold budget.

Definition 454.1 (LocatedModulusCompletion carrier)**.** *A* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, M, S, E, H, C, P, N)$$

with the following visible coordinates:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational observations supply the regular Cauchy name;*
2. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row read from that regular source;*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning each requested precision a finite threshold for the same source row;*
4. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row recording the finite limit-facing seal consumed by completion readers;*
5. *E is the* $\mathsf{LocatedReal}^{\uparrow}$ *evidence row, listing the located read windows and rational separation or enclosure rows attached to the same real-name read;*
6. *H stores componentwise* hsame *transports for the regular source, real seal, modulus row, limit seal, located evidence row, package provenance, and local naming row;*
7. *C stores the displayed* Cont *routes for reading the source windows, applying the modulus threshold, reaching the limit seal, and exposing the located real evidence;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}$ *row.*

The packet is a finite completion-side read certificate. It carries no quotient of Cauchy names, no selected completed value, no host equality of real numbers, and no ambient completeness principle.

Theorem 454.2 (LocatedModulusCompletion NameCert obligations)**.** *For the carrier of Definition 454.1, the local* $\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $L = (R, Q, M, S, E, H, C, P, N)$;*
2. *classifier transport by componentwise* hsame *over the* $\mathsf{RegSeqRat}^{\uparrow}$ *source,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, and* $\mathsf{LocatedReal}^{\uparrow}$ *evidence;*
3. *regular-seal routing, which requires every completion-facing read to pass through the displayed regular source R, modulus row M, and limit seal S before reaching Q or E;*
4. *ledger exactness, which records that the precision threshold used by a read is exactly the threshold displayed by M for that request;*
5. *non-escape, which excludes quotient streams, selected host limits, host real equality, trichotomy, and ambient completeness from the packet surface.*

Lean4 ✓ `BEDC.Derived.LocatedModulusCompletionUp.LocatedModulusCompletionCarrier namecert obligations`

Proof. Carrier admission is the direct projection of the finite tuple in Definition 454.1. The listed rows are all displayed in the same BHist packet and are named by the local certificate row.

Classifier transport is componentwise. The regular source, real seal, modulus row, limit seal, and located evidence each move through the stored hsame rows and are read through the corresponding Cont route.

For routing and exactness, inspect a requested precision. The continuation first reads the threshold row of M, then uses that finite threshold to reach the source window in R, the limit-facing row in S, and the visible real or located rows Q and E.

The non-escape clause follows from the carrier inventory. Since no coordinate stores a quotient stream, selected limit, host equality, trichotomy, or completeness theorem, no consumer route can project one. □

Theorem 454.3 (LocatedModulusCompletion regular seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *read exported by an accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, the same* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row M, and the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *limit-facing row S. The real seal Q and located evidence E therefore share one finite completion route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with the consumer route to Q or E. By the carrier definition, both routes are stored in C and both are packaged with the same provenance row P.

Read the requested precision through M. The threshold selected by the modulus row determines which finite source window in R is inspected before any real-seal or located-evidence row is exposed.

Finally read the limit-facing row S. It is the completion-side seal attached to the same source and threshold, so the exported real and located reads share the same finite route and do not choose an independent completed value. □

Lean4 ✓ `BEDC.Derived.LocatedModulusCompletionUp.LocatedModulusCompletionCarrier regular seal`

Theorem 454.4 (LocatedModulusCompletion classifier stability)**.** *Let $L = (R, Q, M, S, E, H, C, P, N)$ and $L' = (R', Q', M', S', E', H', C', P', N')$ be accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carriers whose displayed rows are related componentwise by the stored* hsame *transports. If the transported modulus rows assign matching threshold reads and the transported limit seals remain attached to the transported regular sources, then every* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *consumer route from L transports to the corresponding route from L'.*

Proof. Project the componentwise transports. The rows R, Q, M, S, E have separate hsame comparisons, and the package and naming rows record that those comparisons belong to one carrier.

Use the modulus comparison first. Matching threshold reads ensure that a requested precision reaches corresponding source windows in R and R'.

Next use the seal comparison. The transported $\mathsf{CauchyLimitSeal}^{\uparrow}$ row remains attached to the transported regular source, so the real-seal and located-evidence routes keep the same source-threshold-limit order.

Repack the transported rows with C', P', N'. The resulting consumer route is exactly the corresponding route from L', and no hidden equality of completed real values is used. □

Theorem 454.5 (LocatedModulusCompletion ledger exactness)**.** *For every accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier and every displayed precision request, the threshold consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *source read, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the* $\mathsf{LocatedReal}^{\uparrow}$ *evidence row is the threshold recorded by the carrier's* $\mathsf{CauchyModulus}^{\uparrow}$ *row for that request. No downstream consumer may replace it by an unrecorded modulus, a host convergence rate, or a choice-selected tail.*

Proof. Inspect the precision request in the carrier. The only threshold coordinate is the displayed row M, and the continuation route C reads that row before it reaches any source, seal, or located-evidence coordinate.

The regular source route consumes the threshold selected by M. The limit seal, real seal, and located evidence are downstream of that same source read, so their tolerance ledger is the one already selected by the modulus row.

An unrecorded modulus or host convergence rate would require an additional carrier coordinate. Since Definition 454.1 lists none, the displayed ledger is exact for every accepted route. □

Theorem 454.6 (LocatedModulusCompletion public scope)**.** *The public surface of a* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *seed packet is precisely the finite route through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *named in Definition 454.1. It exposes located modulus-controlled completion reads and excludes quotient-real construction, selected limits, host order completeness, trichotomy, and equality of completed real representatives.*

Proof. Project the carrier and collect its consumer-facing rows. The visible data are the regular source, real seal, modulus threshold row, limit seal, located evidence, transports, continuation routes, package provenance, and local naming row.

Apply the regular-seal theorem to route real and located reads through the same source, modulus, and limit-seal rows. Apply classifier stability to transport those rows componentwise, and apply ledger exactness to identify the threshold consumed by each precision request.

The public scope is exhausted by those finite rows. None of the excluded objects is present as a carrier coordinate, so the seed packet states only the located modulus-controlled completion surface. □

Theorem 454.7 (LocatedModulusCompletion obligation routing package)**.** *The carrier of Definition 454.1, the regular seal of Definition 454.3, classifier stability of Definition 454.4, ledger exactness of Definition 454.5, and public scope of Definition 454.6 assemble the five* $\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}$ *obligations for carrier admission, classifier transport, regular-seal routing, modulus ledger exactness, and public non-escape.*

Proof. First project the carrier packet $L = (R, Q, M, S, E, H, C, P, N)$. The displayed coordinates provide carrier admission, and the local naming row N names the certificate surface over that same finite packet.

Next attach transport and routing. Classifier stability supplies componentwise transport through the stored hsame rows, while the regular-seal theorem routes every $\mathsf{Real}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ consumer through R, M, S before it reaches Q or E.

Finally attach exactness and scope. Ledger exactness identifies the consumed threshold with the displayed row M for each precision request, and public scope rules out quotient-real construction, selected completed limits, host order completeness, trichotomy, and equality of completed representatives. □

Theorem 454.8 (LocatedModulusCompletion modulus seal ledger scope)**.** *For every accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier and every displayed precision request, the route to either the* $\mathsf{Real}^{\uparrow}$ *seal row* Q *or the* $\mathsf{LocatedReal}^{\uparrow}$ *evidence row* E *passes through the same* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* S*. Thus the modulus ledger and limit-facing seal are shared by all completion-side reads of the packet.*

Proof. Begin with the requested precision. The carrier exposes only one modulus coordinate, namely M, and the continuation inventory reads M before any route reaches a real or located consumer row.

After the modulus threshold is read, the same route reaches the limit-facing seal row S. The regular source row R and the seal row S are packaged with the same provenance P and local certificate row N.

Apply Definition 454.5 to identify the threshold used by every downstream read, and apply Definition 454.3 to factor the routes to Q and E through R, M, S. Hence both consumer rows share the same modulus and seal ledger. □

Theorem 454.9 (LocatedModulusCompletion located-real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer and every* $\mathsf{LocatedReal}^{\uparrow}$ *consumer exported by a* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier receives the same finite source, modulus, and limit-seal packet*

$$(R, M, S)$$

before reading Q *or* E*. The located-real handoff therefore adds located evidence to the displayed completion route without changing the source row, replacing the modulus row, or selecting an external completed value.*

Proof. Project the two consumer rows from the carrier. The $\mathsf{Real}^{\uparrow}$ row is Q and the $\mathsf{LocatedReal}^{\uparrow}$ evidence row is E, and both are downstream of the route inventory stored in C.

Use Definition 454.8. It forces any route to either consumer through the same modulus row M and limit-seal row S, with the same regular source row R read before the consumer boundary.

The located evidence row E therefore extends the real-seal route by visible located-read data only. Since the carrier has no coordinate for a selected completed value or an alternate source/modulus packet, the handoff preserves the same finite packet (R, M, S) for both consumers. □

Theorem 454.10 (LocatedModulusCompletion schedule seal)**.** *For every accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *precision request, the consumer schedule first reads the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* M*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, then the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* S*, before either the* $\mathsf{Real}^{\uparrow}$ *seal row* Q *or the* $\mathsf{LocatedReal}^{\uparrow}$ *evidence row* E *is exposed. This schedule is stable under the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows and cannot be replaced by a quotient stream, a selected completed value, host equality, trichotomy, or ambient completeness.*

Proof. Start with the precision request. By Definition 454.5, the consumed threshold is exactly the one recorded by the displayed modulus row M, so the route cannot begin from an unrecorded convergence rate.

Apply Definition 454.8. After the threshold is read, every route to Q or E passes through the same regular source row R and the same limit-facing seal row S.

Use Definition 454.3 and Definition 454.9 to expose the two possible consumer faces. The real seal and located evidence are downstream of the same packet (R, M, S).

The stability and non-escape clauses are then inherited from the carrier rows. Componentwise hsame transports, displayed Cont routes, Pkg provenance, and the local NameCert row replay the same schedule, while the carrier has no coordinate for quotient streams, selected completed values, host equality, trichotomy, or ambient completeness. □

Theorem 454.11 (LocatedModulusCompletion observation-budget handoff)**.** *Let* $L = (R, Q, M, S, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier, and let*

$$B = (E_0, W, D, R_0, S_0, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet whose regular-sequence row* R_0 *and* $\mathsf{Real}^{\uparrow}$ *seal row* S_0 *agree with the displayed rows* R *and* Q *of* L*. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *consumer route of* L *first factors through the finite budget route*

$$E_0 \longmapsto W \longmapsto D \longmapsto R_0 \longmapsto S_0$$

and then through the displayed modulus row M *and limit-seal row* S *before the consumer reads* Q *or* E*. The handoff consumes only the finite observation-budget packet and the displayed located-modulus ledger; it cannot read quotient streams, selected host limits, ambient completeness, host equality, or an unbounded schedule.*

Proof. Start from the accepted budget packet. By Definition 611.17, every admitted $\mathsf{Real}^{\uparrow}$ seal read of B is reached through E_0, W, D, R_0, S_0 with the displayed transport, route, provenance, and local naming rows.

Use the row agreement hypotheses to identify the budget handoff with the displayed source and real-seal rows of L. Then Definition 611.9 says that the regular-sequence and real-facing consumers have exactly the finite packet rows supplied by B, with no additional analytic coordinate.

Now attach the located completion route. Definition 454.10 forces every consumer route of L to read M, then R, then S, before Q or E is exposed; the agreement with R_0 and S_0 places this schedule downstream of the finite budget route.

Finally apply Definition 454.15. It preserves the same modulus and limit-seal ledger under classifier transport, so the combined route has no coordinate for quotient streams, selected host limits, ambient completeness, host equality, or an unbounded schedule. □

Theorem 454.12 (LocatedModulusCompletion real observation-budget route)**.** *Let $L = (R, Q, M, S, E, H, C, P, N$ be an accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carrier, and let*

$$B = (E_0, W, D, R_0, S_0, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet whose regular-sequence row R_0 and* $\mathsf{Real}^{\uparrow}$ *seal row S_0 agree with the displayed rows R and Q of L. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *read of L first factors through the accepted observation-budget rows*

$$E_0, W, D, R_0, S_0$$

and only then through the displayed modulus row M and limit-seal row S before reaching Q or E.

Proof. Project the carrier from Definition 454.1. The only consumer-facing rows are the $\mathsf{Real}^{\uparrow}$ seal row Q and the $\mathsf{LocatedReal}^{\uparrow}$ evidence row E, and both are attached to the same displayed packet R, M, S, H, C, P, N.

Apply Definition 454.11. It routes every real or located-real consumer through the accepted budget packet E_0, W, D, R_0, S_0 before the located-modulus rows are read.

Use Definition 454.3 and Definition 454.10. The regular source, modulus threshold, and limit-seal route are consumed in the displayed order before Q or E is exposed.

Finally apply Definition 454.5 and Definition 454.15. They identify the consumed threshold with the displayed ledger row and preserve that ledger under classifier transport, so the route is exactly the finite observation-budget packet followed by M, S and then the consumer row. □

Theorem 454.13 (LocatedModulusCompletion threshold source coherence)**.** *Under the hypotheses of Definition 454.12, the modulus threshold row consumed by R is the same threshold ledger routed through the observation-budget window. It cannot be replaced by an unrecorded convergence rate, an alternate modulus packet, or a source row outside the displayed agreement between R_0 and R.*

Proof. Start with the accepted carrier and budget packet. The observation-budget handoff fixes the regular-sequence row by the agreement between R_0 and R, and Definition 454.12 places the subsequent located-modulus read downstream of that finite packet.

Apply Definition 454.5. The threshold used by the route is the one recorded in the displayed modulus row M, not a hidden convergence rate.

Use Definition 454.10. It reads M before the regular source row is consumed by the seal-facing route and then carries the same source through S before Q or E is exposed.

The carrier of Definition 454.1, the regular-seal theorem, and the classifier-ledger seal leave no coordinate for a second threshold ledger. Hence the threshold consumed by R is coherent with the observation-budget window and cannot be exchanged for an unrecorded rate. □

Theorem 454.14 (LocatedModulusCompletion consumer-window non-escape)**.** *For every accepted pair (B, L) satisfying the agreement hypotheses of Definition 454.12, the combined consumer window exports only the observation-budget rows E_0, W, D, R_0, S_0 followed by the located-modulus rows M, S, Q, E and the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *data. It exports no quotient stream, selected completed value, host equality, trichotomy, ambient completeness principle, or unbounded schedule.*

Proof. Project the accepted carrier and budget packet. Their visible rows are the finite observation-budget packet, the located modulus and limit-seal rows, the two consumer rows, and the displayed transport, continuation, provenance, and naming data.

Apply Definition 454.11 and Definition 454.12. These theorems force each real or located-real read through E_0, W, D, R_0, S_0 and then through M, S before Q or E can be consumed.

Use Definition 454.13. The threshold row is the recorded ledger row for the same source route, so the consumer cannot switch to an unrecorded convergence schedule.

Finally combine Definition 454.3, Definition 454.5, Definition 454.10, and Definition 454.15. These results exhaust the displayed route and preserve it under classifier transport, while the carrier contains no row for quotient streams, selected completed values, host equality, trichotomy, ambient completeness, or unbounded schedules. □

Theorem 454.15 (LocatedModulusCompletion classifier ledger seal)**.** *Let $L = (R, Q, M, S, E, H, C, P, N)$ and $L' = (R', Q', M', S', E', H', C', P', N')$ be accepted* $\mathsf{LocatedModulusCompletion}^{\uparrow}$ *carriers whose displayed rows are related componentwise by the stored* hsame *transports. If the modulus row M transports to M' request by request, and the limit-seal attachment S transports to the limit-seal attachment S' for the transported regular source, then every transported route to Q' or E' consumes the transported threshold from M' and the transported seal S'. Hence classifier transport preserves the same modulus row and limit-seal ledger, and no quotient stream, selected completed value, host equality, trichotomy, or ambient completeness principle can enter through the transported classifier.*

Proof. Project the two carrier rows from Definition 454.1. The displayed classifier data are the regular source, real seal, modulus row, limit seal, located evidence, componentwise transports, continuation routes, provenance, and local naming row.

Use Definition 454.5 to identify the consumed threshold with the row supplied by the modulus ledger. Under the assumed transport from M to M', the transported route must therefore read the corresponding threshold in M'.

Next apply Definition 454.8 and Definition 454.9. These force both the real-seal route and the located-evidence route through the same regular source, transported modulus row, and transported limit seal before either Q' or E' is exposed.

Finally apply Definition 454.10. The route order remains threshold, source, limit seal, and then consumer row, so the transported classifier has no coordinate from which a quotient stream, selected completed value, host equality, trichotomy, or ambient completeness principle could be read. □

Theorem 454.16 (LocatedModulusCompletion scoped route closure)**.** *The carrier of Definition 454.1, the NameCert obligations of Definition 454.2, the regular-seal route of Definition 454.3, classifier stability of Definition 454.4, ledger exactness of Definition 454.5, public scope of Definition 454.6, modulus-seal ledger scope of Definition 454.8, located-real handoff of Definition 454.9, schedule seal of Definition 454.10, and classifier-ledger seal of Definition 454.15 assemble the complete obligation surface for $L = (R, Q, M, S, E, H, C, P, N)$.*

Every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *consumer is forced through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No quotient stream, selected completed value, host equality, trichotomy, or ambient completeness coordinate is available inside the scoped route.*

Proof. Project the carrier and local naming rows from Definition 454.1 and Definition 454.2. These provide carrier admission and the five displayed certificate obligations over $R, Q, M, S, E, H, C, P, N$.

Next assemble the consumer route. Apply Definition 454.3, Definition 454.8, Definition 454.9, and Definition 454.10 to force every real or located read through the regular source, modulus threshold, and limit-facing seal before Q or E is exposed.

Finally attach transport, exactness, and scope. The classifier stability and classifier-ledger seal theorems preserve the same displayed route under hsame, while ledger exactness and public scope

rule out every excluded completion coordinate. Hence the listed local theorems exhaust the scoped obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedModulusCompletion}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a $\mathsf{Real}^{\uparrow}$ seal row, a $\mathsf{CauchyModulus}^{\uparrow}$ threshold row, a $\mathsf{CauchyLimitSeal}^{\uparrow}$ limit-facing row, $\mathsf{LocatedReal}^{\uparrow}$ evidence, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}, \mathsf{obligation})$

Scope closed: The seed scope records the finite carrier, local naming obligations, regular-seal route, classifier stability surface, ledger exactness, and public located-modulus completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedModulusCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-real construction, selected completed limits, host order completeness, trichotomy, equality of completed real representatives, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, modulus ledger exactness, classifier stability, regular-seal routing, and public non-escape for the located completion packet.

Chapter 455

A Concrete Naming Certificate for $\mathsf{ReachabilityGramian}^{\uparrow}$

The $\mathsf{ReachabilityGramian}^{\uparrow}$ interface records the finite reachability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon controllability Gramian of Kalman, *On the General Theory of Control Systems* (1960), and the finite matrix-ledger presentation follows the control-theory notation of Kailath, Sayed, and Hassibi, *Linear Estimation* (2000). The BEDC certificate is only the finite discovery surface consumed by $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, and $\mathsf{ModelPredictiveControl}^{\uparrow}$.

The finite Gramian row is a sibling ledger consumed alongside the observability matrix surface of chapter 294 and by the optimization and control surface of chapter 296. The dependency is the shared finite transition and control matrix rows together with the finite horizon ledger, not a host analytic reachability theorem or an ambient control-system semantics.

455.1 Carrier definition

Definition 455.1 (ReachabilityGramianUp finite carrier)**.** *A* $\mathsf{ReachabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, B, n, W, L, C, P, N).$$

Here A is the displayed transition-matrix row, B is the displayed control-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand columns

$$BB^{\mathsf{T}}, \quad ABB^{\mathsf{T}}A^{\mathsf{T}}, \quad A^2BB^{\mathsf{T}}(A^{\mathsf{T}})^2, \quad \ldots, \quad A^{n-1}BB^{\mathsf{T}}(A^{\mathsf{T}})^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, control, horizon, summand, and Gramian endpoint rows, C is the family of* Cont *routes reading every multiplication, transpose, column append, and finite sum from displayed* $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, *and* $\mathsf{ControlControllability}^{\uparrow}$ *rows, P is the package provenance naming those dependency certificates together with* $\mathsf{Riccati}^{\uparrow}$ *and* $\mathsf{LQR}^{\uparrow}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{ReachabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, B, n, W, L, C, P, N rows. It does not quotient by a host reachable subspace, analytic trajectory, controllability rank predicate, covariance equation, or external control-system object.*

Lean4 ✓ `BEDC.Derived.ReachabilityGramianUp.ReachabilityGramianUp single carrier alignment`

455.2 Certificate surface

Theorem 455.2 (ReachabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ReachabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, B, n, W, L, C, P, N *packet of Definition 455.1;*
2. *classifier transport under componentwise* hsame *of the transition, control, horizon, Gramian, summand-ledger, continuation, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed summand columns inside* W*;*
4. *controllability handoff, saying that the reachability-column surface of* $\mathsf{ControlControllability}^{\uparrow}$ *can consume* A*,* B*,* n*,* W*, and* L *only through the displayed* Cont *and* Pkg *rows;*
5. *non-escape, saying that no accepted route exports a host reachable subspace, analytic trajectory, rank theorem, infinite-horizon Gramian, covariance process, or external solver.*

Lean4 ✓ `BEDC.Derived.ReachabilityGramianUp.ReachabilityGramianBHistCarrier namecert obligations`

Proof. The carrier row is Definition 455.1. It displays the transition row, control row, unary horizon, Gramian endpoint, summand ledger, continuation routes, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For the finite sum ledger, inspect W. Each summand is a displayed matrix product formed from A, B, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The continuation routes in C record every multiplication and the final sum, while L records the permitted componentwise transports, so no extra analytic Gramian object is available.

The controllability handoff is a projection to the shared A, B, and n rows and to the finite reachability columns named by W. Because those rows are already packaged with $\mathsf{ControlControllability}^{\uparrow}$ provenance and transported only through L, a downstream controllability certificate can read them without importing host semantics.

Non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host reachable subspace, analytic trajectory, rank theorem, infinite-horizon Gramian, covariance process, or external solver, so none of those objects can be projected by a consumer route. □

Theorem 455.3 (ReachabilityGramianUp controllability handoff)**.** *Every* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{Riccati}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*, or* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *consumer of an accepted* $\mathsf{ReachabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* B *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*,* Cont *routes,* Pkg *provenance, and local* NameCert *row of Definition 455.1. The handoff supplies no host reachable-space quotient, no analytic state trajectory, and no infinite-horizon controllability theorem.*

Lean4 ✓ `BEDC.Derived.ReachabilityGramianUp.ReachabilityGramianBHistCarrier controllability handoff`

Proof. Project the accepted packet to the rows A, B, n, W, L, C, P, N. The first three rows are the shared finite control-system coordinates, while W records the finite Gramian discovery ledger and L records the componentwise transport ledger over those same coordinates.

A $\mathsf{ControlControllability}^{\uparrow}$ consumer reads reachability columns through the displayed transition, control, horizon, and continuation rows. A $\mathsf{Riccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, or $\mathsf{ModelPredictiveControl}^{\uparrow}$ consumer reads only the resulting finite matrix endpoint and the package provenance that names the source certificates.

The carrier contains no row for a host reachable-space quotient, analytic state trajectory, or infinite-horizon controllability theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 455.4 (ReachabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ReachabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, B, n, W, L, C, P, N *of Definition 455.1. In particular, the packet exports no ambient reachable subspace, no analytic trajectory family, no stochastic covariance process, no infinite-horizon Gramian, no rank-completeness theorem, and no external numerical solver.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, control row, horizon row, finite Gramian-summand and endpoint row, transport ledger, continuation routes, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, controllability handoff, and non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame and Cont routes.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient reachable subspace, analytic trajectory family, stochastic covariance process, infinite-horizon Gramian, rank-completeness theorem, or external numerical solver, no accepted route can supply one downstream. □

Lean4 ✓ `BEDC.Derived.ReachabilityGramianUp.ReachabilityGramianBHistCarrier ledger nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{ReachabilityGramian}^{\uparrow}}$.

Constructive story: A $\mathsf{ReachabilityGramian}^{\uparrow}$ packet is generated from displayed transition and control matrix rows, a unary finite horizon, the finite Gramian summand and endpoint row, componentwise hsame transport ledger, Cont routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReachabilityGramian}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 455.1, Definition 455.2, Definition 455.3, and Definition 455.4 enumerate the finite Gramian carrier, NameCert obligations, control handoff, and nonescape surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReachabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host reachable-subspace semantics, analytic trajectories, stochastic covariance theory, infinite-horizon Gramians, rank-completeness, numerical solvers, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, controllability handoff, and non-escape.

Chapter 456

A Concrete Naming Certificate for $\mathsf{TransferOperator}^{\uparrow}$

The $\mathsf{TransferOperator}^{\uparrow}$ packet is a BEDC-native finite certificate surface. It records the finite-window transfer-matrix reading of golden-mean paths as visible BHist rows, with no ambient Banach space, no analytic spectrum, and no imported external theorem.

The packet consumes the finite symbolic source lattice of chapter 370, chapter 434, and chapter 401. $\mathsf{GoldenMeanShift}^{\uparrow}$ supplies the no-adjacent-one window row, $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ supplies the forbidden-pair classifier, and $\mathsf{FibonacciCube}^{\uparrow}$ supplies the finite path-support row read by every accepted transfer-matrix entry.

Definition 456.1 (Transfer operator carrier)**.** *A* $\mathsf{TransferOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, G, S, F, M, R, C, P, N)$$

with the following visible coordinates:

1. *A is the finite adjacency-matrix row, displaying the accepted source row, target row, and matrix entry for each visible transition;*
2. *G is the accepted-window source row from* $\mathsf{GoldenMeanShift}^{\uparrow}$*, specialized to the no-adjacent-one ledger;*
3. *S is the finite forbidden-adjacency and accepted-pair row from* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$*;*
4. *F is the finite path and word-support row from* $\mathsf{FibonacciCube}^{\uparrow}$*, recording the same visible words as path-support packets;*
5. *M is the finite transfer-matrix row, whose entries are read only from the displayed adjacency rows of A, G, S, F;*
6. *R stores the componentwise* hsame *classifier rows for adjacency rows, accepted windows, finite supports, and matrix entries;*
7. *C stores the* Cont *readback routes from rows of M to the corresponding golden-mean, subshift, and Fibonacci-cube source rows;*
8. *P is the* Pkg *provenance naming* $\mathsf{GoldenMeanShift}^{\uparrow}$*,* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$*,* $\mathsf{FibonacciCube}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *source rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TransferOperator}^{\uparrow}}$ *ledger.*

The classifier compares two carriers only by the finite rows $A, G, S, F, M, R, C, P, N$. No coordinate contains an infinite shift space, a Ruelle operator, a spectral radius, a circle-dimension object, or an analytic transfer theorem.

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperatorTasteGate single carrier alignment`

Theorem 456.2 (Transfer operator NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TransferOperator}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the packet of Definition 456.1;*
2. *classifier laws by componentwise* hsame *on adjacency rows, accepted-window rows, subshift ledgers, finite path-support rows, matrix entries, readback routes, package provenance, and the local certificate ledger;*
3. *matrix-entry exactness: every row of M is read from one displayed source-target adjacency row in A together with its G, S, F support rows;*
4. *readback stability: every* Cont *route in C returns to the same finite golden-mean, subshift, and Fibonacci-cube rows that supplied the entry;*
5. *non-escape: an accepted packet exports no analytic transfer operator, no infinite spectrum, no ambient topology, and no external import.*

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperatorCarrier namecert obligations`

Proof. Carrier admission is the displayed tuple $T = (A, G, S, F, M, R, C, P, N)$. Each coordinate is either a finite BHist row or a finite route between such rows.

For the classifier laws, inspect the component rows one at a time. The classifier has access only to the displayed hsame rows in R, so reflexive, symmetric, and transitive reads are inherited coordinate by coordinate.

Matrix-entry exactness projects M. An entry is accepted only when its source row, target row, and displayed weight are read from the same adjacency row and from the same G, S, F support rows.

Readback stability then projects C. Each route records where the matrix entry was read from, so returning along the route recovers the finite golden-mean window, the subshift forbidden-pair decision, and the Fibonacci-cube support row used by that entry.

Non-escape follows from the carrier definition. Since no coordinate names an analytic operator, infinite spectrum, topological carrier, or external theorem, downstream consumers can cite only the finite rows and their recorded routes. □

Theorem 456.3 (Transfer operator symbolic source lattice)**.** *Every accepted transfer-matrix entry of a* $\mathsf{TransferOperator}^{\uparrow}$ *packet factors through the symbolic source lattice formed by the* $\mathsf{GoldenMeanShift}^{\uparrow}$ *no-adjacent-one window row, the* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *forbidden-pair classifier row, and the* $\mathsf{FibonacciCube}^{\uparrow}$ *finite path-support row before it reaches the local matrix ledger M.*

Proof. Project the carrier $T = (A, G, S, F, M, R, C, P, N)$. The matrix row M is not an independent source coordinate; by Definition 456.1, it is read only from the displayed adjacency row together with the support rows G, S, F.

The row G supplies the accepted golden-mean window, so an entry whose source-target pair is one-one has no route into M. The row S supplies the finite forbidden-pair classifier, so the same acceptance decision is visible as a subshift row rather than as an external symbolic-dynamics theorem.

The row F supplies the finite path-support witness. Combining G, S, F gives the complete symbolic source lattice for an accepted entry, and the continuation row C records the route from that lattice into the local matrix ledger.

Thus every accepted matrix entry factors through those three sibling rows before it can be read as a row of M. Since no coordinate bypasses G, S, F, the packet has no kernel-only matrix entry disconnected from its finite symbolic source. □

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperator symbolic source lattice`

Theorem 456.4 (Transfer operator golden-mean matrix boundary)**.** *For the golden-mean specialization of a* $\mathsf{TransferOperator}^{\uparrow}$ *carrier, the visible transfer-matrix boundary is the finite two-state adjacency ledger whose only forbidden source-target row is the one-one transition. Hence every accepted matrix entry is indexed by one of the displayed rows zero-zero, zero-one, or one-zero, and every one-one query is rejected by the same singleton forbidden row used by* $\mathsf{GoldenMeanShift}^{\uparrow}$ *and* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$.

Proof. Specialize the G row to the $\mathsf{GoldenMeanShift}^{\uparrow}$ no-adjacent-one ledger and the S row to the $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ singleton forbidden-pair ledger. The displayed forbidden row is exactly the adjacent pair one-one.

Project the adjacency matrix row A. Its accepted source-target rows are those whose pair is absent from the singleton forbidden ledger, so the surviving visible rows are zero-zero, zero-one, and one-zero.

Finally read M through A, G, S, F. Since every matrix entry carries the same readback route in C, a one-one query cannot supply an entry; its route stops at the rejected forbidden row rather than at an accepted matrix row. □

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperator golden mean matrix boundary`

Theorem 456.5 (Transfer operator finite-window ledger)**.** *Every exported calculation of an accepted* $\mathsf{TransferOperator}^{\uparrow}$ *packet factors through the finite packet rows* $A, G, S, F, M, R, C, P, N$. *In particular, row sums, path counts, and displayed matrix products are ledger reads over the finite window and do not invoke entropy, inverse limits, quotient shift spaces, spectral measures, or external linear algebra.*

Proof. Take an exported calculation and project the carrier. The only available coordinates are the adjacency rows, golden-mean rows, subshift rows, Fibonacci-cube support rows, transfer-matrix rows, classifier rows, readback routes, package provenance, and local naming ledger.

Row sums and displayed matrix products are finite reuses of rows of M. Each row of M is already tied by C to its source adjacency and support rows.

Therefore every calculation factors through the listed finite ledger. Since the carrier contains no entropy coordinate, inverse-limit carrier, quotient shift space, spectral-measure row, or host matrix object, no exported route can cite such data. □

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperator finite window ledger`

Theorem 456.6 (Transfer operator subshift handoff)**.** *The* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *row* S *is the handoff surface from forbidden-pair ledgers to transfer-matrix entries: transporting* S *componentwise by* hsame *and the recorded* Cont *routes transports precisely the allowed adjacency rows in* A *and therefore transports the supported matrix rows in* M.

Lean4 ✓ `BEDC.Derived.TransferOperatorUp.TransferOperator subshift handoff`

Proof. Project the S coordinate and the classifier row R. The subshift row stores the finite allowed and forbidden neighboring-pair decisions, while R records the componentwise hsame routes for those decisions.

Project A and M next. A matrix row is supported exactly when its source-target pair is accepted by the transported subshift ledger and its finite path support appears in F.

The readback routes in C then repack the transported rows into the same packet shape. Thus the handoff from forbidden-pair decisions to matrix entries is constructive and finite: it is only projection, componentwise transport, and repackaging by P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{TransferOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{TransferOperator}^{\uparrow}$ packet is generated from a symbolic graph row, subshift row, finite forbidden-pair ledger, adjacency-matrix rows, path-support rows, componentwise hsame transport, displayed Cont readbacks, package provenance, and local naming. The classifier reads a transfer entry only after the source-target pair is accepted by the transported subshift ledger and its finite support appears in the path ledger. The ledger story is finite symbolic handoff: forbidden-pair decisions generate matrix entries by projection, transport, and repackaging, while analytic operators, spectra, entropy, inverse limits, ambient topology, and host matrix objects remain outside the carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TransferOperator}^{\uparrow}}, \mathsf{obligation})$

Scope closed: This chapter binds the carrier definition and the NameCert-obligation, symbolic-source-lattice, golden-mean-matrix-boundary, finite-window-ledger, and subshift-handoff theorems for $\mathsf{TransferOperator}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TransferOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim analytic transfer operators, infinite spectra, ambient topologies, inverse-limit carriers, entropy coordinates, spectral measures, or host matrix objects.

Upgrade path: To reach obligationClosure, the finite symbolic transfer carrier and subshift handoff must be encoded as checked matrix-ledger obligations.

Chapter 457

A Concrete Naming Certificate for HankelVandermondeUp

The HankelVandermondeUp packet is a BEDC-native finite certificate surface. It records a finite determinant ledger for displayed Hankel moment rows and their Vandermonde-square readback. It is consumable by FoldMomentKernel$^{\uparrow}$ and CollisionKernelSpectrum$^{\uparrow}$ without importing host matrices, analytic spectra, quotient moment spaces, or external field algebra. The determinant and Vandermonde-square rows form a finite ledger consumed by chapter 384 and chapter 447: FoldMomentKernel$^{\uparrow}$ supplies the finite fold-moment rows, while CollisionKernelSpectrum$^{\uparrow}$ reads the spectral-shadow consumer boundary through that finite ledger, and HankelVandermondeUp itself exports no analytic spectrum or host matrix carrier.

Definition 457.1 (Hankel-Vandermonde carrier)**.** *A* HankelVandermondeUp *carrier is a finite* BHist *packet*

$$V = (A, W, M, D, \Delta, Q, H, C, P, N)$$

with the following visible coordinates:

1. *A is the displayed atom row, listing the finite atom names whose pairwise differences are read by the determinant ledger;*
2. *W is the weight row, recording the finite coefficient attached to each displayed atom;*
3. *M is the Hankel moment row, whose entries are displayed finite sums over the atom and weight rows;*
4. *D is the determinant-expansion ledger row, listing the finite permutation terms used to read the selected Hankel determinant;*
5. *Δ is the pairwise-difference row, containing only displayed differences between atoms from A;*
6. *Q is the Vandermonde-square product row, recording the finite product of the displayed pairwise differences after squaring and weighting;*
7. *H stores the componentwise* hsame *transports for atom, weight, moment, determinant-ledger, pairwise-difference, and Vandermonde-square rows;*
8. *C stores the displayed* Cont *routes through which determinant and product consumers read the same finite rows;*
9. *P is the* Pkg *provenance naming* BHist, hsame, Cont, FoldMomentKernel$^{\uparrow}$, CollisionKernelSpectrum$^{\uparrow}$, *finite determinant ledgers, and local naming evidence;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HankelVandermondeUp}}$ *evidence row.*

The classifier compares two carriers only by the displayed rows $A, W, M, D, \Delta, Q, H, C, P, N$. It has no coordinate for host matrices, analytic spectra, quotient moment spaces, imported external theorems, or external field algebra.

Lean4 (definition) `BEDC.Derived.HankelVandermondeUp.HankelVandermondeUp`

Lean4 ✓ `BEDC.Derived.HankelVandermondeUp.HankelVandermondeTasteGate single carrier alignment`

Theorem 457.2 (Hankel-Vandermonde NameCert obligations)**.** *Every accepted* HankelVandermondeUp *packet with carrier Definition 457.1 carries five local naming obligations:*

1. *carrier habitation: the tuple $V = (A, W, M, D, \Delta, Q, H, C, P, N)$ is admitted as one finite* BHist *packet;*
2. *classifier transport: componentwise* hsame *and* Cont *transport preserves the displayed atom, weight, moment, determinant-ledger, pairwise-difference, Vandermonde-square, package, and naming rows;*
3. *determinant-ledger exactness: every determinant read passes through the finite permutation ledger D and the displayed Hankel moment row M;*
4. *Vandermonde-square readback: every product read passes through the finite pairwise-difference row Δ and the Vandermonde-square row Q;*
5. *non-escape: accepted carriers export no host matrix object, analytic spectrum, quotient moment space, imported external theorem, or external field algebra.*

These five clauses are the complete local $\mathsf{NameCert}_{\mathsf{HankelVandermondeUp}}$ *surface for finite Hankel-Vandermonde determinant ledgers.*

Lean4 ✓ `BEDC.Derived.HankelVandermondeUp.HankelVandermondeCarrier namecert obligations`

Proof. Carrier habitation is the displayed tuple $V = (A, W, M, D, \Delta, Q, H, C, P, N)$. Each coordinate is a finite BHist row or a finite route between such rows, so the packet is admitted without a host matrix carrier.

For classifier transport, move each visible coordinate through its stored hsame row and read it through the corresponding Cont route. Because the classifier is componentwise, transport cannot create a new atom, weight, moment, determinant term, pairwise difference, product factor, package row, or naming row.

Determinant-ledger exactness is finite ledger inspection. The selected Hankel determinant is read only by expanding along the displayed permutation terms in D, and each term cites entries already present in the moment row M.

The Vandermonde-square readback then projects Δ and Q. Pairwise differences are available only between atoms listed in A, and the squared product row records only the factors displayed by Δ together with the weights from W.

Non-escape follows from the carrier definition. Since no coordinate names a host matrix space, analytic spectral object, quotient moment space, external theorem, or external field algebra, downstream $\mathsf{FoldMomentKernel}^{\uparrow}$ and $\mathsf{CollisionKernelSpectrum}^{\uparrow}$ consumers can cite only the finite rows and routes in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HankelVandermondeUp}}$**.**

Constructive story: A HankelVandermondeUp packet is generated from displayed atom rows, weight rows, Hankel moment rows, determinant-expansion ledger rows, pairwise-difference rows, Vandermonde-square product rows, componentwise transports, continuation routes, package provenance, and local naming evidence.

Theory closure: TheoryGate(NameCert$_{\text{HankelVandermondeUp}}$, seed)
Scope closed: The seed scope records the finite carrier and five-clause NameCert surface for Hankel-Vandermonde determinant ledgers.
Formal verification: FormalStatus(NameCert$_{\text{HankelVandermondeUp}}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not close host matrix theory, analytic spectra, quotient moment spaces, external field algebra, external theorem reuse, or a checked Lean certificate.
Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the displayed determinant and Vandermonde-square rows.

Theorem 457.3 (Hankel-Vandermonde spectral-shadow bridge)**.** *Let*

$$V_H = (A, W, M, D, \Delta, Q, H_H, C_H, P_H, N_H)$$

be an accepted HankelVandermondeUp *carrier. Let*

$$K_\Omega = (W_\Omega, F_\Omega, L_\Omega, I_\Omega, Q_\Omega, T_\Omega, C_\Omega, P_\Omega, N_\Omega)$$

be an accepted FoldMomentKernel$^\uparrow$ *finite-window carrier, and let*

$$C_S = (G, F, V, I, K, S, H_S, T_S, P_S, N_S)$$

be an accepted CollisionKernelSpectrum$^\uparrow$ *carrier. Suppose the shared* FoldMomentKernel$^\uparrow$ *moment surface identifies the collision moment-index row I with the retained row I_Ω, and suppose every displayed spectral-shadow request in S is read at the same finite moment address of the Hankel row M. Suppose also that, at those addresses, the displayed atom row A, weight row W, determinant ledger D, pairwise-difference row Δ, and Vandermonde-square row Q are the rows recorded by the* CollisionKernelSpectrum$^\uparrow$ *provenance through P_S and by the* FoldMomentKernel$^\uparrow$ *provenance through P_Ω.*

Then every downstream consumer route that reads the spectral-shadow row S factors through the finite HankelVandermondeUp *determinant surface*

$$A, W, M, D, \Delta, Q, H_H, C_H, P_H, N_H.$$

More explicitly, each such read first reaches S as a finite row-sum, diagonal weight, or selected moment shadow of the displayed kernel row K, then passes through the common finite moment-index row $I = I_\Omega$, then reads the Hankel moment row M, and finally reaches the determinant ledger D and the Vandermonde-square readback rows Δ, Q. No such consumer route exports a host matrix, analytic spectrum, quotient moment space, selected eigenbasis, external field algebra, or ambient linear-algebra object.

Proof. Project the accepted CollisionKernelSpectrum$^\uparrow$ carrier through Definition 447.1. By Definition 447.3, the row S is not an independent analytic spectrum: each of its displayed entries is a row-sum, diagonal weight, or selected moment shadow of the finite kernel entries in K. By Definition 447.8, any exported spectral-shadow read must pass through the finite moment-index row I together with the visible rows $G, F, V, K, H_S, T_S, P_S, N_S$.

The hypothesis identifies that moment-index row with the retained FoldMomentKernel$^\uparrow$ row I_Ω. Apply Definition 384.5 to the accepted finite-window carrier K_Ω. A moment read over this surface can inspect only the displayed window, fold-source, fiber-ledger, moment-index, collision-count, transport, continuation, package, and local naming rows. If the route is restricted to the retained window, Definition 384.6 keeps the same conclusion inside the packet K_Ω. Thus the

spectral-shadow consumer cannot leave the finite moment surface before it reaches the shared Hankel moment address.

At that shared address, the second hypothesis names the Hankel-Vandermonde rows A, W, M, D, Δ, Q as the rows carried by the collision-spectrum provenance and by the fold-moment provenance. Now apply Definition 457.2. Determinant-ledger exactness forces every determinant read of M through the finite permutation ledger D, and Vandermonde-square readback forces the product read through the displayed pairwise-difference row Δ and the Vandermonde-square row Q, with componentwise transport and continuation stored in H_H and C_H.

Combining the three projections gives the announced factorization. A read of S factors through K, then through the common finite moment-index row $I = I_\Omega$, then through the Hankel moment row M, and finally through D, Δ, Q inside the carrier V_H. The non-escape clauses of Definition 447.4, Definition 384.5, and Definition 457.2 exclude exactly the remaining host matrix, analytic spectrum, quotient moment, selected eigenbasis, external field-algebra, and ambient linear-algebra exports. □

Chapter 458

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailSelector}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy sequence tail selection, the $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ packet names only the finite rows needed to select an inspected tail of a regular rational Cauchy name. The packet sits on the $\mathsf{Real}^{\uparrow}$ and completion path: downstream diagonal-limit and finite-window consumers can cite the selected-tail ledger without obtaining a quotient representative, a host real equality, a selected infinite subsequence, or an ambient completeness principle.

The finite route is anchored in the source-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the locator-facing Cauchy real route of chapter 3410. These siblings supply the visible faces of the selected-tail packet; none supplies an off-packet quotient, selected limit, or countable-choice operator.

The selector is the one-source face of the shared-tail threshold surface in chapter 560. Its selected-tail witness row w is consumed by the sibling meet packet whenever two regular Cauchy sources must share a dyadic threshold; Definition 560.39 is the exact sibling dependency that admits this witness into the shared-threshold route.

Definition 458.1 (Regular Cauchy tail selector carrier)**.** *A* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (q, s, r, d, e, w, H, C, P, N)$$

with the following visible rows:

1. *q is the requested rational precision row, carried by* $\mathsf{Rat}^{\uparrow}$ *and read through the dyadic comparison surface;*
2. *s is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose finite windows are inspected by the selector;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row certifying that the inspected stream windows obey the displayed Cauchy tolerance schedule;*
4. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row that turns each requested precision into the dyadic-radius comparison used by the tail ledger;*
5. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the selected tail row and regularity row have both been read;*
6. *w is the selected-tail witness row, listing the requested precision, the displayed starting index, the finite suffix window, and the dyadic bounds checked on that window;*

7. *H stores componentwise* hsame *transports for the rational precision row, stream row, regularity row, dyadic readback row, real seal row, selected-tail witness row, package provenance, and local naming row;*
8. *C stores the displayed* Cont *routes by which diagonal-limit, finite-window, and real-seal consumers may read the packet;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local selector rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient of Cauchy streams, a host-level completed real, a globally chosen infinite tail function, or an external real-completeness theorem.

Theorem 458.2 (Regular Cauchy tail selector NameCert obligations)**.** *The carrier of Definition 458.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSelector}^{\uparrow}}$ *obligation surface. Its five obligation rows are:*

1. *carrier admission for the finite tuple $T = (q, s, r, d, e, w, H, C, P, N)$;*
2. *classifier transport by the componentwise* hsame *rows stored in H and the consumer routes stored in C;*
3. *tail witness exactness, saying that w names only the displayed precision request, starting index, finite suffix window, and dyadic bounds;*
4. *regularity compatibility, saying that each selected-tail bound is read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row r and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row d before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed;*
5. *seal non-escape, saying that the real-facing route exposes only the finite selected-tail ledger and never a quotient representative, a host real equality, a choice-selected limit, or ambient completeness.*

Proof. Carrier admission is the displayed BHist tuple. Each coordinate is either a finite row, a transport row, a continuation row, package provenance, or the local naming row.

For classifier transport, move the rational precision, stream window, regularity, dyadic readback, seal, witness, package, and local naming coordinates through H, then read them through C. Componentwise transport cannot introduce a tail row outside the tuple.

Tail witness exactness is inspection of w. It records one requested precision, one displayed starting index, one finite suffix window, and the dyadic bounds checked on that window.

Regularity compatibility follows by reading r and d before e. The selected tail is certified by regular rational data and dyadic comparisons, so the real-facing seal is reached only after those rows have been consumed.

The carrier contains no quotient stream, host completed real, global selector, or ambient completeness coordinate. The local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSelector}^{\uparrow}}$ obligations are therefore exhausted by the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailSelectorUp.RegularCauchyTailSelectorPacket namecert obligation surface`

Theorem 458.3 (Regular Cauchy tail selector diagonal admissibility)**.** *Every accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet supplies an admissible selected-tail row for diagonal-limit, finite-window, and* $\mathsf{Real}^{\uparrow}$ *seal consumers. A consumer may read the requested rational precision row, the* $\mathsf{StreamName}^{\uparrow}$ *window, the*

RegSeqRat$^{\uparrow}$ *regularity row, the* DyadicRatCore$^{\uparrow}$ *comparison row, the selected-tail witness row, the displayed* hsame *and* Cont *routes, package provenance, and the local* NameCert *row. It cannot obtain a quotient representative of the stream, a host real equality, a choice-selected limiting value, or an ambient completeness theorem.*

Proof. Start with an accepted carrier $T = (q, s, r, d, e, w, H, C, P, N)$. The rows q, s, r, d, w provide the finite precision request, inspected stream window, regularity certificate, dyadic comparison surface, and selected-tail witness.

Read the witness row w through the classifier transport of Definition 458.2. The displayed hsame and Cont routes preserve the same requested precision, starting index, suffix window, and dyadic bounds for any consumer using the packet.

For a diagonal-limit consumer, the admissible input is exactly this finite selected-tail row together with its regularity and dyadic-readback support. The consumer can splice the row into a diagonal read because every bound in w has already been checked against r and d.

For a Real$^{\uparrow}$ or completion-facing consumer, the seal row e is read only after the selected-tail witness and regularity rows have been consumed. Since no coordinate of T is a quotient representative, host real equality, selected limiting value, or ambient completeness proof, the admissible handoff remains the finite selected-tail ledger. □

Theorem 458.4 (Regular Cauchy tail selector classifier transport)**.** *Let $T = (q, s, r, d, e, w, H, C, P, N)$ and $T' = (q', s', r', d', e', w', H', C', P', N')$ be accepted* RegularCauchyTailSelector$^{\uparrow}$ *packets whose rational precision, stream, regularity, dyadic readback, real seal, selected-tail witness, package, and naming rows are matched componentwise by the displayed* hsame *rows and read through the displayed* Cont *routes. Then every selector row available to a diagonal-limit, finite-window, or* Real$^{\uparrow}$ *consumer of T transports to the corresponding selector row of T'.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailSelectorUp.RegularCauchyTailSelectorPacket classifier transport`

Proof. Begin with the componentwise hsame data stored in H and H'. It pairs the rational precision row, source stream row, regularity row, dyadic readback row, seal row, selected-tail witness row, package provenance, and local naming row.

Read those paired rows through the Cont routes in C and C'. By the classifier-transport clause of Definition 458.2, this preserves the same requested precision, starting index, finite suffix window, and dyadic bounds.

The consumer surface is the one described in Definition 458.3. Hence no transported selector row can acquire a quotient representative, host real equality, choice-selected limiting value, or ambient completeness coordinate. □

Theorem 458.5 (Regular Cauchy tail selector precision-window exactness)**.** *For every accepted packet $T = (q, s, r, d, e, w, H, C, P, N)$, the witness row w exhausts the selector precision window: it records exactly the requested rational precision, the displayed starting index, the finite suffix window of s, and the dyadic bounds checked through d against the regularity row r. There is no second selected-tail ledger outside w.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailSelectorUp.RegularCauchyTailSelectorPacket precision window exactness`

↪ variant `BEDC.Derived.RegularCauchyTailSelectorUp.RegularCauchyTailSelectorCarrier precision window exactness`

Proof. Project the carrier of Definition 458.1. The coordinate w is explicitly the selected-tail witness row, and its listed fields are the precision request, starting index, finite suffix window, and dyadic bounds.

Use the tail witness exactness and regularity compatibility clauses of Definition 458.2. They force each bound in w to be read through the $\mathsf{RegSeqRat}^{\uparrow}$ row r and the $\mathsf{DyadicRatCore}^{\uparrow}$ row d before the seal row e is exposed.

The diagonal admissibility theorem Definition 458.3 gives the downstream read boundary. Since T has no alternate selector coordinate, the precision window is exactly w together with its displayed support rows. □

Theorem 458.6 (Regular Cauchy tail selector Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted packet* $T = (q, s, r, d, e, w, H, C, P, N)$ *can read only the selected-tail ledger* w*, the regularity row* r*, the dyadic readback row* d*, the displayed* Cont *routes, the* Pkg *provenance row* P*, and the local* NameCert *row* N*, with componentwise transport supplied by* H*. It cannot read a quotient representative, host real equality, chosen limit, infinite tail function, or ambient completeness principle.*

Proof. The Real-facing route starts with Definition 458.3. It allows the seal row only after the selected-tail witness, regularity row, and dyadic readback row have been consumed.

Apply Definition 458.5 to identify the selected-tail ledger with w and its support rows r and d. Apply Definition 458.4 to account for the displayed hsame and Cont routes.

The carrier of Definition 458.1 lists no quotient representative, host real equality, chosen limit, infinite tail function, or ambient completeness coordinate. The Real seal therefore cannot escape the finite selected-tail, regularity, dyadic readback, route, package, and local naming rows. □

Theorem 458.7 (Regular Cauchy tail selector tail-support coverage)**.** *Every accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet exposes exactly the tail-support rows needed by its selected-tail witness surface: the requested precision row, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row, the selected-tail witness, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It exposes no quotient stream, host equality, infinite tail function, or completeness coordinate.*

Proof. Use the carrier of Definition 458.1. Its coordinates are the finite tuple $T = (q, s, r, d, e, w, H, C, P, N)$, where $q, s, r, d, w, H, C, P, N$ are precisely the support rows named in the statement.

The obligation surface of Definition 458.2 identifies these coordinates as the admitted carrier, classifier-transport rows, witness-exactness row, regularity-compatibility row, and local seal boundary. The diagonal admissibility theorem Definition 458.3 gives the same readable boundary to diagonal-limit, finite-window, and $\mathsf{Real}^{\uparrow}$ consumers.

Finally, Definition 458.6 rules out the four omitted coordinates. Since the carrier has no additional tail-support slot, the exposed surface is exactly the displayed finite ledger. □

Theorem 458.8 (Regular Cauchy tail selector transported-bound preservation)**.** *For accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packets, componentwise* hsame *transport preserves the dyadic bounds recorded in the selected-tail witness. Transport can move the requested precision, stream window, regularity row, dyadic comparison row, witness row, route row, provenance row, and local naming row to their corresponding coordinates, but it cannot create a second bound ledger.*

Proof. Start with the witness-exactness row from Definition 458.2. It places the requested precision, starting index, finite suffix window, and dyadic bounds in the single witness coordinate w.

Apply the classifier-transport theorem Definition 458.4. Componentwise hsame moves the rational precision, stream, regularity, dyadic readback, real seal, witness, package, and naming rows through the displayed Cont routes, preserving the corresponding selector rows for each consumer.

The precision-window exactness theorem Definition 458.5 says that w exhausts the selector precision window. Therefore the transported dyadic bounds remain the bounds recorded in w, and no second ledger can be introduced by transport. □

Theorem 458.9 (Regular Cauchy tail selector diagonal consumer package)**.** *Diagonal-limit, finite-window, and* $\mathsf{Real}^{\uparrow}$ *consumers of an accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet receive a single finite package assembled from tail-support coverage, transported-bound preservation, and the admissible selected-tail row of Definition 458.3. The package contains the finite witness ledger and its support rows, and no consumer receives an additional quotient, equality, infinite-tail, or completeness coordinate.*

Proof. First assemble the readable support rows using Definition 458.7. This gives the requested precision, stream window, regularity, dyadic comparison, selected-tail witness, transport, continuation, provenance, and local naming rows.

Next apply Definition 458.8. The dyadic bounds carried by the selected-tail witness survive componentwise transport as the same finite ledger, with no second bound record.

Finally use Definition 458.3. The diagonal-limit, finite-window, and $\mathsf{Real}^{\uparrow}$ consumer routes read the assembled finite package through the admissible selected-tail row. The non-escape boundary of Definition 458.6 excludes any additional quotient, equality, infinite-tail, or completeness coordinate. □

Theorem 458.10 (Regular Cauchy tail selector finite induction)**.** *Every accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *route over a finite inspected tail window is generated by induction over the displayed finite support row. The induction preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, precision-window exactness, classifier transport, Real seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No step introduces an infinite-tail coordinate, quotient equality of regular Cauchy streams, host real equality, or ambient completeness.*

Proof. Induct on the finite support list exposed by Definition 458.7.

In the empty case, the route exposes only the base precision window. The carrier still displays the requested precision, stream row, regularity row, dyadic readback, selected-tail witness, transport, replay, provenance, and local naming, and Definition 458.5 identifies that base window as the whole selector ledger.

For the step case, append one inspected tail coordinate to the finite support list. Tail-support coverage attaches that coordinate to the same selected-tail witness row, while Definition 458.8 keeps the dyadic bounds stable under componentwise hsame transport.

It remains to check that the consumer route has not changed type. Apply Definition 458.12 to keep the new inspected coordinate on the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ handoff before the $\mathsf{Real}^{\uparrow}$ seal is read.

Finally use the diagonal consumer package theorem above and the Real seal non-escape theorem. The inductively generated package is exactly the finite support route read by diagonal-limit, finite-window, and Real-facing consumers, so no additional quotient, infinite tail, host equality, or completeness coordinate is available. □

Theorem 458.11 (Regular Cauchy tail selector scope window)**.** *For every accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet*

$$T = (q, s, r, d, e, w, H, C, P, N),$$

the selected-tail witness window is bound to the displayed BHist *carrier, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. A selector consumer can read the requested precision, starting index, finite suffix window, and dyadic bounds only through* w, H, C, P, N.

Proof. Read the carrier in Definition 458.1. The only selected-tail coordinate is w, and the only structural rows exposing it are H, C, P, N.

Use the obligation surface of Definition 458.2. Its tail witness exactness clause identifies the requested precision, starting index, finite suffix window, and dyadic bounds as the complete content of w.

Apply Definition 458.5 to exclude a second selector ledger. The selected window is therefore scoped by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 458.12 (Regular Cauchy tail selector RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff consumed by a* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet reads its dyadic bounds through the* $\mathsf{StreamName}^{\uparrow}$ *source row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The handoff route is*

$$s, r, d, w, H, C, P, N \to e,$$

where s *supplies the inspected stream windows,* r *supplies the regularity ledger,* d *supplies dyadic comparison, and* w *records the selected finite tail window.*

Proof. Start from the regularity compatibility row in Definition 458.2. It requires each selected-tail bound to be read through r and d before the seal row e is exposed.

The witness window itself is fixed by Definition 458.5. Its finite suffix is a window of the $\mathsf{StreamName}^{\uparrow}$ source row s, and its bounds are the dyadic comparisons supplied through d against the regularity row r.

Finally apply Definition 458.6. The Real-facing consumer receives the finite selected-tail package after this route and cannot bypass the StreamName, RegSeqRat, or DyadicRatCore rows. □

Theorem 458.13 (Regular Cauchy tail selector Real consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *packet can read only the finite selector package*

$$q, s, r, d, w, H, C, P, N.$$

It cannot obtain quotient equality of regular Cauchy streams, a selected infinite tail, host real equality, or ambient completeness from the selector packet.

Proof. Use the diagonal admissibility theorem Definition 458.3. It names the finite precision, stream, regularity, dyadic readback, selected-tail witness, transport, route, provenance, and local naming rows available to Real-facing consumers.

Apply the scope-window theorem Definition 458.11 and the RegSeqRat handoff theorem Definition 458.12. Together they bind the witness window to the finite carrier and force the dyadic bounds through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before the seal is read.

The non-escape theorem Definition 458.6 then rules out quotient equality, selected infinite tails, host real equality, and ambient completeness. No other carrier coordinate can supply those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ packet is generated from a rational precision row, a stream-name row, a regular rational Cauchy row, a dyadic readback row, a real seal row, a selected-tail witness row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailSelector}\uparrow}$, maturePackageClosed)

Scope closed: Definition 458.1, Definition 458.2, and Definition 458.3 record the finite selected-tail witness surface for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailSelector}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyTailSelectorUp.RegularCauchyTailSelectorPacket\_classifier\_transport`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of regular Cauchy streams, host real equality, a selected infinite tail function, ambient completeness, or a checked Lean certificate.

Upgrade path: No further paper-axis theory step remains for this tail-selector packet; bridge checking remains a separate verification-axis task.

Chapter 459

A Concrete Naming Certificate for $\mathsf{RegularCauchyNegation}^{\uparrow}$

The $\mathsf{RegularCauchyNegation}^{\uparrow}$ interface names the finite sign-flip surface for a regular rational Cauchy name. Following Bishop and Bridges, *Constructive Analysis*, the construction treats negation as a displayed transformation of dyadic observations and tolerance ledgers: a consumer may inspect the source name, the scheduled windows, the sign-flipped dyadic rows, and the $\mathsf{Real}^{\uparrow}$ seal handoff, but it cannot form a quotient stream, compare host sequence equality, select a limit, or appeal to ambient real completeness.

The packet is the unary sign-flip sibling consumed by chapter 417 and chapter 429. Those siblings read the same finite window, dyadic observation, classifier, and $\mathsf{Real}^{\uparrow}$ seal boundary rather than host completed-real arithmetic.

Definition 459.1 (RegularCauchyNegation carrier)**.** *A* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

with the following displayed coordinates:

1. *X is the* $\mathsf{RegularCauchyName}^{\uparrow}$ *source row whose regular rational observations are to be sign-flipped;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule naming the finite source reads for each requested precision;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row read from X through W;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise classifier row for the source regular Cauchy name and its scheduled rational observations;*
5. *F is the sign-flipped dyadic observation ledger, recording for each displayed row of D the dyadic row with the same precision data and the opposite signed endpoint;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row through which downstream real-name consumers receive the sign-flipped regular sequence;*
7. *H stores componentwise* hsame *transports for the source row, window schedule, dyadic observation row, classifier row, sign-flipped ledger, and seal row;*
8. *C stores the displayed* Cont *routes for reading the source window, forming the sign-flipped dyadic row, transporting the classifier, and exposing the seal handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$ *row.*

The packet is a finite naming certificate for the negated regular Cauchy name. It stores no quotient stream, no host sequence equality proof, no selected limit, and no ambient real-completeness object.

Lean4 (definition) `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier`

Theorem 459.2 (RegularCauchyNegation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$ *surface for the carrier of Definition 459.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is the displayed packet G with a* $\mathsf{RegularCauchyName}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, sign-flipped ledger,* $\mathsf{Real}^{\uparrow}$ *seal handoff, transport rows, continuation rows, provenance, and local naming data;*
2. *classifier transport: classification transports X, W, D, R, F, E componentwise through the stored* hsame *and* Cont *rows;*
3. *sign-flip stability: replacing the source by a classified regular Cauchy name preserves the finite window schedule and recomputes the same kind of sign-flipped dyadic observation ledger;*
4. *finite-window transport: every negated observation is obtained by reading the finite window named by W and then applying the displayed sign flip in* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
5. $\mathsf{Real}^{\uparrow}$ *seal handoff: every real-name consumer of the negated row first reads $X, W, D, R, F, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier namecert obligations`

Proof. Carrier admission is projection from the tuple in Definition 459.1. The source name, window schedule, dyadic observation row, classifier row, sign-flipped ledger, seal handoff, transport routes, continuation routes, package row, and local naming row are explicit coordinates of G.

Classifier transport is componentwise. The $\mathsf{RegularCauchyName}^{\uparrow}$ row transports its scheduled observations, the $\mathsf{StreamName}^{\uparrow}$ windows transport the finite reads, the $\mathsf{DyadicRatCore}^{\uparrow}$ row transports the observed endpoint data, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier transports the pointwise regular rational readback.

The sign-flip and seal clauses are finite. The ledger F is formed from the displayed dyadic row D without choosing a completed value, and the $\mathsf{Real}^{\uparrow}$ handoff E consumes that ledger only after the source window and classifier have been exposed. Thus no quotient stream, host sequence equality proof, selected limit, or ambient completeness principle is available to the consumer. □

Theorem 459.3 (RegularCauchyNegation diagonal compatibility)**.** *For every accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N),$$

the sign-flipped packet commutes with the two standard regular-Cauchy consumer routes: a $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ *consumer may first diagonalize the displayed source windows and then read the sign-flipped dyadic ledger, while a* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *consumer may first read the pointwise classifier row and then consume the same* $\mathsf{Real}^{\uparrow}$ *seal handoff. Both routes expose the same finite rows $X, W, D, R, F, E, H, C, P, N$ and no quotient stream, host sequence equality, selected limit, or ambient real-completeness object.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier diagonal limit readout`

Proof. For the diagonal route, read the source row through the $\mathsf{RegularCauchyName}^{\uparrow}$ surface and its $\mathsf{StreamName}^{\uparrow}$ schedule. The $\mathsf{RegularCauchyDiagonal}^{\uparrow}$ consumer selects only finite displayed windows, so the dyadic row D remains the observation row named by the carrier.

Apply the sign-flip ledger after the diagonal read. Since F is computed pointwise from the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ rows of D, the diagonal consumer receives the same finite sign-flipped observations and the same transported hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

For the limit-classifier route, first expose the $\mathsf{RegSeqRat}^{\uparrow}$ pointwise classifier row R. The $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ consumer then reaches the $\mathsf{Real}^{\uparrow}$ seal row E only through R and the sign-flipped ledger F.

The two routes therefore differ only in the order in which finite rows are read. The compatibility uses the same source windows, dyadic observations, classifier row, sign-flip ledger, and seal handoff, and it does not introduce quotient streams, host sequence equality, selected limits, or ambient completeness. □

Theorem 459.4 (RegularCauchyNegation algebraic handoff)**.** *Every consumer that combines* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *with* $\mathsf{RegularCauchySum}^{\uparrow}$ *or* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *factors through the displayed sign-flipped ledger F, the source classifier row R, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff E. The sum route reads the negated row as one finite summand; the difference route reads it as the displayed unary part of pointwise subtraction. Neither route may bypass $X, W, D, R, F, E, H, C, P, N$ to obtain a quotient stream, host sequence equality, selected limit, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier algebraic handoff`

Proof. Begin with the local obligations of Definition 459.2. They force every sign-flipped observation through the source window W, dyadic row D, and ledger F before any algebraic consumer sees the row.

For a $\mathsf{RegularCauchySum}^{\uparrow}$ consumer, F is simply the displayed dyadic observation row supplied for one summand, and R supplies the regular rational classifier needed by the sum readback. The consumer therefore reads the negated name as a finite summand rather than as an equality class of completed streams.

For a $\mathsf{RegularCauchyDifference}^{\uparrow}$ consumer, the same rows supply the unary sign-flip surface that pointwise subtraction may inspect. The difference packet can read the displayed opposite endpoint and its tolerance data from F, then route the result through the same $\mathsf{Real}^{\uparrow}$ seal handoff E.

The carrier inventory exhausts both algebraic handoffs. Since no coordinate of G is a quotient stream, host sequence equality proof, selected limit, or ambient completeness theorem, neither algebraic route can export such data. □

Theorem 459.5 (RegularCauchyNegation carrier transport)**.** *Let*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier. Displayed* hsame *transport of the* $\mathsf{RegSeqRat}^{\uparrow}$ *source classifier row R, the sign-flipped window row F, the dyadic tolerance ledger D, and the* $\mathsf{Real}^{\uparrow}$ *seal row E, together with the stored source, schedule, route, package, and local naming rows, preserves the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier predicate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier transport`

Proof. Project the carrier of Definition 459.1. The source row, window schedule, dyadic observation row, classifier row, sign-flipped ledger, seal row, transports, routes, provenance, and local name are all coordinates of the same finite packet.

Apply the classifier transport clause in Definition 459.2. It transports X, W, D, R, F, E componentwise through the stored H and C rows. The diagonal compatibility theorem keeps the transported finite windows attached to the same dyadic observations, and the algebraic handoff theorem keeps the seal handoff routed through the same F, R, E rows. □

Theorem 459.6 (RegularCauchyNegation sign-flip ledger stability)**.** *For an accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier, the sign-flip route records no hidden host negation. Every negated observation is obtained from the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row through the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window ledger and is then recorded in the sign-flipped ledger* F *before any* $\mathsf{Real}^{\uparrow}$ *seal handoff.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier sign flip ledger stability`

Proof. Begin with the finite-window transport obligation in Definition 459.2. It requires each negated observation to read the window schedule W and dyadic row D before the sign-flipped ledger F is consumed.

The diagonal compatibility theorem shows that changing the consumer route does not change these rows: the diagonal and limit-classifier reads both expose $X, W, D, R, F, E, H, C, P, N$. The algebraic handoff theorem then confirms that sum and difference consumers also factor through F, R, and E. Thus the sign flip is a finite ledger operation and not a hidden host-level negation. □

Theorem 459.7 (RegularCauchyNegation limit-classifier compatibility)**.** *The negated* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *packet respects the limit classifier inherited from* $\mathsf{RegSeqRat}^{\uparrow}$ *and hands off to the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. A* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *consumer may inspect the pointwise classifier row* R *and the sign-flipped ledger* F*, but it receives the real-name boundary only through the displayed seal handoff* E*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier limit classifier compatibility`

Proof. Use the $\mathsf{Real}^{\uparrow}$ seal handoff clause of Definition 459.2. It forces every real-name consumer to read the source, window, dyadic, classifier, sign-flipped, seal, transport, route, package, and naming rows.

Now apply diagonal compatibility. Its limit-classifier route exposes R before reaching E, and it reaches E only through R and the sign-flipped ledger F. The algebraic handoff theorem preserves this same boundary when the negated row is consumed by sum or difference packets, so the inherited limit classifier and the $\mathsf{Real}^{\uparrow}$ seal remain compatible. □

Theorem 459.8 (RegularCauchyNegation public RealUp seal export)**.** *For every accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N),$$

the public consumer boundary exported to $\mathsf{Real}^{\uparrow}$ *is exactly the displayed seal row* E *reached through* $X, W, D, R, F, H, C, P, N$*. Any additive-inverse, sum, difference, diagonal, or limit-classifier consumer of the negated regular Cauchy name must pass through the same seal row* E*, and it receives no quotient stream, host sequence equality, selected limit, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier public seal export`

Proof. Begin with the five finite obligations in Definition 459.2. Carrier admission exposes the rows $X, W, D, R, F, E, H, C, P, N$, and the $\mathsf{Real}^{\uparrow}$ handoff clause says that a real-name consumer reaches the public boundary only after those rows have been read.

For algebraic consumers, use Definition 459.4. Sum and difference routes read the negated packet through the sign-flipped ledger F, the source classifier row R, and the same seal handoff E; no additive-inverse interface is allowed to replace E by a host-level completed-real operation.

For schedule and classifier consumers, combine Definition 459.6 with Definition 459.7. The sign flip remains a displayed dyadic ledger operation, and the inherited limit classifier reaches the real-name boundary only through E.

Thus the public export is a finite BEDC seal handoff. It is not a completed-real inverse theorem, because the carrier has no coordinate from which a quotient stream, host sequence equality, selected limit, or ambient completeness premise could be projected. □

Theorem 459.9 (RegularCauchyNegation apartness non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *apartness, additive-inverse, sum, or difference consumer of an accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

must read the sign-flipped dyadic ledger F, the source classifier row R, and the $\mathsf{Real}^{\uparrow}$ *seal row E through the displayed rows $X, W, D, R, F, E, H, C, P, N$. It cannot replace that route by host sequence equality, quotient-stream equality, selected limits, completed-real inverse laws, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier apartness nonescape`

Proof. Start with the local obligations of Definition 459.2. They expose the source row, window schedule, dyadic observation row, classifier row, sign-flipped ledger, seal handoff, transports, continuation routes, provenance, and local naming row as coordinates of the same packet.

The algebraic handoff theorem Definition 459.4 handles additive-inverse, sum, and difference consumers. Each such route reads the sign-flipped ledger F as the unary negation surface, uses the source classifier row R as the regular rational readback boundary, and reaches the real-facing row only through the seal E.

For apartness consumers, combine sign-flip ledger stability with limit-classifier compatibility. The apartness read can inspect the displayed opposite dyadic observations and their inherited classifier data, but the route to a RealUp boundary still passes through F, R, and E in the same carrier.

Finally apply the public seal export theorem Definition 459.8. Since every public real-facing consumer receives exactly the displayed seal route and the carrier contains no host sequence equality, quotient-stream equality, selected limit, completed-real inverse law, or ambient completeness premise, none of those objects can escape through apartness or additive-inverse use. □

Theorem 459.10 (RegularCauchyNegation additive-inverse boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *additive-inverse, apartness, sum, or difference consumer of an accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

must pass through the sign-flipped dyadic ledger F, the source classifier row R, and the $\mathsf{Real}^{\uparrow}$ *seal row E displayed by G. The consumer may inspect the finite rows $X, W, D, R, F, E, H, C, P, N$ through the stored* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$ *row, but it cannot replace this boundary by quotient streams, host sequence equality, selected limits, completed-real inverse laws, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier additive inverse boundary`

Proof. Begin with the NameCert obligations of Definition 459.2. Carrier admission and classifier transport place the consumer inside the displayed packet G, with R as the source regular-rational classifier and F as the only sign-flipped dyadic observation ledger.

Next apply the algebraic handoff theorem Definition 459.4. Additive-inverse, sum, and difference consumers all read the negated row as a finite algebraic surface through F and R, not as a completed-real inverse operation.

The sign-flip ledger stability theorem Definition 459.6 and the limit-classifier compatibility theorem Definition 459.7 keep the same finite sign-flipped observations attached to the inherited $\mathsf{RegSeqRat}^{\uparrow}$ classifier. Thus any classifier-facing or apartness-facing read still reaches the real boundary only through F, R, E.

Finally combine the public seal export theorem Definition 459.8 with apartness non-escape Definition 459.9. These results exhaust the public $\mathsf{Real}^{\uparrow}$ route and rule out quotient streams, host sequence equality, selected limits, completed-real inverse laws, and ambient completeness as exported data. □

Theorem 459.11 (RegularCauchyNegation involutive seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *algebraic, apartness, sum, or difference consumer of an accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

that applies the displayed sign-flip boundary twice must factor through the original $\mathsf{RegularCauchyName}^{\uparrow}$ *source row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* R*, the first sign-flipped ledger* F*, the second displayed sign-flipped ledger, and the same* $\mathsf{Real}^{\uparrow}$ *seal route* E, H, C, P, N*. The double route exports no quotient stream, host sequence equality, selected limit, completed-real inverse law, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier involutive seal boundary`

Proof. Start with the NameCert obligations in Definition 459.2. They make $X, W, D, R, F, E, H, C, P, N$ the only available carrier coordinates, and they force any sign-flip read to pass through the source windows, dyadic row, classifier row, displayed transports, continuation routes, provenance, and local naming row.

Apply sign-flip ledger stability Definition 459.6 to the first boundary. The first flip is a finite operation on the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ observations read through W and classified by R, so the consumer obtains only the ledger F and not a host-level negation or completed real name.

Apply the same boundary form to the second displayed sign flip. The second ledger is again reached through the same source row, schedule, dyadic observations, classifier row, transports, routes, package provenance, and naming certificate; it is not an independent quotient object and it does not replace the original source row X by host sequence equality.

For algebraic consumers, use the additive-inverse boundary Definition 459.10. It already restricts additive-inverse, apartness, sum, and difference routes to the finite rows $X, W, D, R, F, E, H, C, P, N$, so a double sign-flip route can only add the second displayed ledger before returning to the same $\mathsf{Real}^{\uparrow}$ seal handoff.

Finally combine public seal export Definition 459.8 with apartness non-escape Definition 459.9. The public real-facing boundary remains the same seal route E, H, C, P, N, and the absence of

quotient streams, host sequence equality, selected limits, completed-real inverse laws, and ambient completeness premises prevents the double route from exporting stronger real-algebra data. □

Theorem 459.12 (RegularCauchyNegation bridge readback scope)**.** *Every bridge-facing consumer of the public sign-flipped* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N)$$

must factor through the displayed $\mathsf{RegularCauchyName}^{\uparrow}$ *source row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *pointwise classifier row* R*, the sign-flipped dyadic ledger* F*, the* hsame *transports* H*, the* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$ *row* N*. The readback may reach the public seal row* E *only through those rows, and it does not assert quotient equality, selected limits, ambient real completeness, or full additive inverse laws for completed real names.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier bridge readback scope`

Proof. Begin with carrier transport. The NameCert obligations expose $X, W, D, R, F, E, H, C, P, N$, and transport stability keeps the source row, window schedule, dyadic observation row, classifier row, and seal row in the same displayed packet.

Next apply sign-flip ledger stability. It makes F a pointwise read from the dyadic observation row D under the scheduled windows W, so a bridge-facing consumer cannot replace the sign-flipped observations by a completed-real operation.

Use limit-classifier compatibility for the classifier readback. The inherited $\mathsf{RegSeqRat}^{\uparrow}$ row R is read before the real-facing handoff, and the public seal export theorem routes the handoff to E only after $X, W, D, R, F, H, C, P, N$ have been exposed.

Finally apply apartness non-escape. It rules out quotient streams, host sequence equality, selected limits, completed-real inverse laws, and ambient completeness as exported bridge data, so the bridge-facing readback scope is exactly the displayed finite carrier surface. □

Theorem 459.13 (RegularCauchyNegation real-algebra bridge certificate)**.** *For every accepted* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier*

$$G = (X, W, D, R, F, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$*-algebra bridge certificate exported by* G *is exactly the finite certificate assembled from the sign-flipped dyadic ledger* F*, the inherited* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* R*, the public* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the displayed* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$ *row* N*. This certificate is the bridge-facing handoff for downstream sum, difference, additive-inverse, and apartness consumers; it exports no quotient stream, host sequence equality, selected limit, completed-real inverse law, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier real algebra bridge certificate`

Proof. The NameCert obligations of Definition 459.2 identify the carrier and classifier rows inside the single packet G. In particular, R is the inherited $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, while H, C, P, N are the displayed transport, continuation, provenance, and local naming rows available to any certified consumer.

The sign-flip ledger stability theorem Definition 459.6 pins the negated dyadic observations to F, and the algebraic handoff theorem Definition 459.4 is the route by which sum, difference, additive-inverse, and apartness consumers read that ledger. Thus the algebra-facing part of the certificate is the pair (F, R) together with the displayed rows H, C, P, N, not a completed-real inverse operation.

The public seal export theorem Definition 459.8 supplies the real-facing boundary E. The involutive seal boundary Definition 459.11 keeps double sign-flip use inside the same displayed carrier rows, so repeated algebraic reads do not create an additional public object beyond F, R, E, H, C, P, N.

Finally, the bridge readback scope theorem Definition 459.12 describes the bridge-facing consumers of the public sign-flipped seal. Combining that scope with apartness non-escape Definition 459.9 rules out quotient streams, host sequence equality, selected limits, completed-real inverse laws, and ambient completeness, which leaves exactly the stated finite bridge certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyNegation}^{\uparrow}$ packet is generated from a $\mathsf{RegularCauchyName}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ window schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, a $\mathsf{RegSeqRat}^{\uparrow}$ pointwise classifier row, a sign-flipped dyadic ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The real-algebra bridge certificate theorem Definition 459.13 binds the sign-flipped dyadic ledger, inherited RegSeqRat classifier row, RealUp seal, displayed hsame transports, Cont routes, Pkg provenance, and local NameCert row for bridge-facing algebra consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyNegation}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.RegularCauchyNegationUp.RegularCauchyNegationCarrier\_real\_algebra\_bridge\_certificat`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient equality of streams, host sequence equality, selected limits, ambient real completeness, completed-real additive inverse laws, a StdBridge object, or ambient real algebra.

Upgrade path: No further paper-side theory closure is claimed beyond standard bridge checking for the displayed real-algebra certificate.

Chapter 460

A Concrete Naming Certificate for $\mathsf{CauchyDoubleSequence}^{\uparrow}$

The $\mathsf{CauchyDoubleSequence}^{\uparrow}$ interface names the finite certificate used when a two-parameter regular Cauchy array is read through a diagonal cofinal window and then handed to the constructive real-completion path. It records the array, the selected diagonal schedule, tail tolerance data, regular readback, completion consumer, and $\mathsf{Real}^{\uparrow}$ seal provenance without asserting a quotient limit or host double-sequence convergence theorem.

The two-parameter array route consumes the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular-rational readback surface of chapter 301, and the seal-facing $\mathsf{Real}^{\uparrow}$ row of chapter 38 before the $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumer is read.

Definition 460.1 (CauchyDoubleSequence carrier)**.** *A* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, \mu, \tau, D, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. *X is a two-parameter* $\mathsf{RegSeqRat}^{\uparrow}$ *family, read only through finite stream windows and regular rational rows;*
2. *μ is the* $\mathsf{CauchyRate}^{\uparrow}$ *schedule selecting a diagonal cofinal window in the two indices;*
3. *τ is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row assigned to the selected diagonal window;*
4. *D is the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained by applying μ and τ to X;*
5. *R is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row that reads the diagonal regular name;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed after the completion consumer reads D;*
7. *H stores the componentwise* hsame *transports for the array, schedule, tolerance, diagonal readback, completion consumer, and real seal rows;*
8. *C stores the* Cont *routes for reading the array windows, schedule, tail bound, diagonal readback, completion consumer, and seal;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDoubleSequence}^{\uparrow}}$ *row.*

The carrier is a finite diagonal certificate. It contains no coordinate for an ambient double limit, quotient equality of arrays, a selected host limit, or unscoped completeness.

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceTasteGate single carrier alignment`

Theorem 460.2 (CauchyDoubleSequence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyDoubleSequence}^{\uparrow}}$ *surface for the carrier of Definition 460.1 consists exactly of carrier admission, diagonal schedule admission, tail-bound routing, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumption,* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *readback, package provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 460.1. The coordinates $X, \mu, \tau, D, R, E, H, C, P, N$ give the complete accepted packet.

The schedule, tolerance, and readback obligations are then obtained by reading μ, τ, and D in order. The schedule selects the cofinal diagonal window, the tolerance row bounds that selected window, and the diagonal readback repacks the selected rational rows as a regular Cauchy name.

Finally the completion and seal obligations are projections through R and E. The transport, continuation, package, and local naming obligations are the displayed rows H, C, P, N, so the certificate has no remaining source from which to read an external limit or quotient array equality. □

Theorem 460.3 (CauchyDoubleSequence diagonal handoff)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier factors through the diagonal route*

$$X, \mu, \tau \longmapsto D \longmapsto R \longmapsto E.$$

The consumer receives only the two-parameter $\mathsf{RegSeqRat}^{\uparrow}$ *family windows selected by* μ*, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *row* τ*, the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* D*, the completion consumer row* R*, and the seal row* E *with their displayed* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier diagonal handoff`

Proof. First project the accepted packet of Definition 460.1. The only array coordinate is X, and the only row capable of selecting a cofinal diagonal window is μ.

Next read the tolerance row τ and the diagonal readback D. These rows convert the selected finite windows of X into the regular Cauchy data consumed downstream.

It remains to route the real-facing consumer. The row R reads D as the completion-facing input, and E is the sole $\mathsf{Real}^{\uparrow}$ seal exposed after that read. Thus every handoff factors through the displayed diagonal route and cannot bypass it through an ambient double-limit object. □

Theorem 460.4 (CauchyDoubleSequence Real seal non-escape)**.** *An accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier exports no* $\mathsf{Real}^{\uparrow}$*-facing data except the seal row* E *obtained from the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* D *through the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row* R*. In particular, the carrier has no coordinate for host double-sequence convergence, quotient equality of regular arrays, selected limits, or an unscoped real-completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceSealNonEscape`

Proof. Inspect the carrier inventory in Definition 460.1. The only $\mathsf{Real}^{\uparrow}$ coordinate is E, and it appears after the diagonal readback and completion consumer rows.

The preceding handoff theorem fixes the path to E through D and R. Transport and continuation rows move this displayed path componentwise; they do not create an additional seal or an ambient comparison operation.

Therefore every real-facing read is exhausted by $X, \mu, \tau, D, R, E, H, C, P, N$. The excluded host convergence, quotient, selected-limit, and unscoped-completeness data have no coordinate in the packet. □

Theorem 460.5 (CauchyDoubleSequence scoped surface)**.** *The scoped* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *surface is the finite* BHist *package over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *whose only analytic route is the diagonal handoff of Definition 460.3 and whose only real-facing escape boundary is Definition 460.4.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceScopedSurface`

Proof. Combine the carrier and obligation statements. The carrier fixes the ten displayed coordinates, and the NameCert obligations say that accepted consumers read exactly those coordinates.

The diagonal handoff theorem identifies the unique analytic route from the two-parameter array to the completion consumer. The seal non-escape theorem identifies the unique real-facing boundary after that route.

Hence the chapter surface is scoped to the displayed BHist package and its named dependencies. No theorem in this surface claims an independent completeness result for double arrays or a public standard bridge. □

Theorem 460.6 (CauchyDoubleSequence window diagonal exhaustion)**.** *Let* $A = (X, \mu, \tau, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier. Every finite consumer read of the two-parameter array* X *that contributes to the diagonal readback* D *factors through a window selected by the schedule row* μ *and bounded by the tolerance row* τ*. Thus no downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read can inspect an array coordinate outside the visible finite windows recorded by* $X, \mu, \tau, D, H, C, P, N$.

Proof. Project the accepted carrier. The array row is X, the schedule row is μ, and the tolerance row is τ; no other coordinate can select or bound a two-parameter window.

Apply the diagonal handoff of Definition 460.3. The handoff reads X only after μ has selected the cofinal window and τ has supplied the bound used by the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ row D.

The completion consumer and real seal appear only after D. Therefore every finite consumer read contributing to D, R, or E is exhausted by the visible schedule and tolerance windows, and no ambient array coordinate can be read outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier window diagonal exhaustion`
Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceWindowDiagonalExhaustion`

Theorem 460.7 (CauchyDoubleSequence two-axis seal factorization)**.** *For an accepted carrier* $A = (X, \mu, \tau, D, R, E, H, C, P, N)$*, every real-facing read factors as a two-axis path*

$$X \xrightarrow{\mu,\tau} D \xrightarrow{R} E.$$

The first axis is finite diagonal-window selection from the array row to the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the second axis is completion-consumer access from* D *to the* $\mathsf{Real}^{\uparrow}$ *seal row. The factorization is stable under the componentwise* hsame *transports and* Cont *routes recorded in* H *and* C.

Proof. Use Definition 460.6 to isolate the first axis. It says that all array information consumed by D is already selected by μ and bounded by τ.

Use the Real seal non-escape theorem Definition 460.4 for the second axis. The only real-facing coordinate is E, and it is exposed through the completion consumer row R after the diagonal readback.

Finally transport the two axes through H and read them through C. These rows preserve the displayed path as BHist data; they do not introduce a host double limit, a quotient of arrays, or a second real seal. □

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceTwoAxisSealFactorization`

Theorem 460.8 (CauchyDoubleSequence schedule-tail exactness)**.** *For every accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier* $A = (X, \mu, \tau, D, R, E, H, C, P, N)$*, the diagonal schedule row* μ *and the tail-bound row* τ *exactly determine the finite array windows consumed by the diagonal readback* D*. The route uses the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *array rows, the* $\mathsf{CauchyRate}^{\uparrow}$ *schedule, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row, and the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CauchyDoubleSequence}^{\uparrow}}$ *provenance.*

No completion consumer may inspect an array coordinate outside the scheduled and tail-bounded windows.

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier schedule tail exactness`

Proof. Begin with window diagonal exhaustion. It says that every array read contributing to D is selected by μ and bounded by τ.

Use the diagonal handoff to align those selected windows with the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The schedule row is the only $\mathsf{CauchyRate}^{\uparrow}$ coordinate, and the tolerance row is the only $\mathsf{DyadicTailBound}^{\uparrow}$ coordinate.

Transport the route through H and read it through C. Package and naming provenance record the same finite schedule-tail path, so no unscheduled array coordinate is available to the completion side. □

Theorem 460.9 (CauchyDoubleSequence completion consumer scope)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *packet factors through the schedule-tail route*

$$(X, \mu, \tau) \longmapsto D \longmapsto R \longmapsto E.$$

The first stage is the finite $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window read controlled by* $\mathsf{CauchyRate}^{\uparrow}$ *and* $\mathsf{DyadicTailBound}^{\uparrow}$ *rows; the second stage is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer row; the final stage is the* $\mathsf{Real}^{\uparrow}$ *seal row, transported through* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CauchyDoubleSequence}^{\uparrow}}$*.*

The consumer receives no host double-limit object, quotient equality of arrays, selected limit, or unscoped completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier completion consumer scope`

Proof. First apply schedule-tail exactness. The completion input D is already exhausted by the scheduled finite windows and their tail-bound row.

Next apply seal non-escape and two-axis seal factorization. They identify R as the only completion consumer row and E as the only real-facing seal available after D.

Finally use the scoped surface. It binds the whole route to the displayed BHist, hsame, Cont, Pkg, and NameCert rows, so every completion consumer remains inside the finite packet. □

Theorem 460.10 (CauchyDoubleSequence real consumer boundary)**.** *Any downstream consumer that reads a* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *packet as real-completion input receives exactly the diagonal window data, the completion-consumer row, and the seal row displayed by* $X, \mu, \tau, D, R, E, H, C, P, N$*. Its read factors through Definition 460.7 and cannot depend on host double-sequence convergence, quotient equality of arrays, selected limits, or unscoped real completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier real consumer boundary`

Proof. Begin with the scoped surface of Definition 460.5. It restricts the public package to the named BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The two-axis factorization then identifies the only real-facing route. A consumer first receives the finite diagonal windows selected from X by μ and τ, then the diagonal readback D, then the completion row R, and finally the seal row E.

The excluded host objects are not coordinates of the carrier and are ruled out again by Definition 460.4. Hence the consumer boundary is exactly the displayed finite packet and its seal row. □

Definition 460.11 (CauchyDoubleSequence scoped route)**.** *The scoped diagonal route for an accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *carrier*

$$A = (X, \mu, \tau, D, R, E, H, C, P, N)$$

is the finite BHist *route*

$$(X, \mu, \tau, H, C, P, N) \longmapsto D \longmapsto R \longmapsto E.$$

Here X *is the two-parameter* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* μ *is the* $\mathsf{CauchyRate}^{\uparrow}$ *diagonal schedule,* τ *is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row,* D *is the scheduled diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* R *is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer, and* E *is the* $\mathsf{Real}^{\uparrow}$ *seal. The route is scoped by the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{CauchyDoubleSequence}^{\uparrow}}$ *rows; it is not an ambient convergence theorem for host double sequences.*

Theorem 460.12 (CauchyDoubleSequence scoped diagonal route)**.** *Every accepted* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *packet binds the two-parameter* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{CauchyRate}^{\uparrow}$ *diagonal schedule,* $\mathsf{DyadicTailBound}^{\uparrow}$ *tolerance row,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer, and* $\mathsf{Real}^{\uparrow}$ *seal into the scoped route of Definition 460.11. Every real-completion read of the packet factors through this route as a* BHist *dependency surface.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceScopedDiagonalRoute`

Proof. Project the accepted carrier $A = (X, \mu, \tau, D, R, E, H, C, P, N)$. The only two-parameter source row is X, the only diagonal selector is μ, and the only tolerance row assigned to the selected windows is τ.

Use schedule-tail exactness to identify the windows consumed by D. The diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback is therefore already determined by the displayed array, schedule, and tail-bound rows.

Next use completion consumer scope and two-axis seal factorization. They force every real-completion read through D, then through the completion consumer row R, and finally through the $\mathsf{Real}^{\uparrow}$ seal row E.

The transport, continuation, package, and naming rows H, C, P, N scope the whole route as finite BHist evidence. Thus the route is a displayed dependency surface, not a host double-sequence convergence principle. □

Theorem 460.13 (CauchyDoubleSequence cofinal-tail stability)**.** *Let*

$$A = (X, \mu, \tau, D, R, E, H, C, P, N)$$

be an accepted CauchyDoubleSequence$^{\uparrow}$ *carrier. If transported cofinal schedule and tail-bound rows* μ' *and* τ' *select only diagonal windows no earlier than the displayed* μ, τ *route, then the diagonal readback determined by* μ' *and* τ' *has the same* RealCauchyCompletion$^{\uparrow}$ *consumer row and the same* Real$^{\uparrow}$ *seal row, both factoring through the displayed provenance* H, C, P, N*. The statement is only a cofinal-tail stability claim for the finite route; it exports no host double-limit convergence theorem, no quotient equality of arrays, and no selected limit.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier cofinal tail stability`

Proof. Start with Definition 460.8. The transported rows μ' and τ' are allowed only because their windows lie in the displayed tail selected by μ and bounded by τ. Hence the new diagonal readback reads the same finite source family X through a later scheduled tail, not through a different array object.

Apply Definition 460.7. Once the diagonal readback is fixed by the scheduled tail, the completion-facing consumer must pass through R and the real-facing consumer must pass through E.

Finally use Definition 460.12. The transport, continuation, package, and local naming rows H, C, P, N scope both the original route and the cofinal readback route. Since the route has no coordinates for host double-limit convergence, quotient array equality, or selected limits, the stability result cannot export any of those objects. □

Theorem 460.14 (CauchyDoubleSequence diagonal-tail stability)**.** *Let*

$$A = (X, \mu, \tau, D, R, E, H, C, P, N) \qquad \text{and} \qquad A' = (X', \mu', \tau', D', R', E', H', C', P', N')$$

be accepted CauchyDoubleSequence$^{\uparrow}$ *packets. Suppose the array rows* X *and* X' *are related by componentwise* hsame *on every finite diagonal window consumed by their displayed readback rows, the schedule rows* μ, μ' *are cofinal over those consumed windows, and the tail-bound rows* τ, τ' *agree on each displayed dyadic tolerance window used by the readback rows. Then the diagonal* RegSeqRat$^{\uparrow}$ *readbacks* D, D' *expose the same* RealCauchyCompletion$^{\uparrow}$ *consumer row and the same* Real$^{\uparrow}$ *seal route, both through the scoped dependency surfaces* H, C, P, N *and* H', C', P', N'*. The statement exports no host double-limit convergence theorem, quotient equality of arrays, selected limit, or unscoped real-completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceDiagonalTailStability`

Proof. Project both accepted carriers. Their public analytic data are exactly the array, schedule, tail-bound, diagonal readback, completion-consumer, real-seal, transport, replay, provenance, and local naming rows displayed in the statement.

Apply schedule-tail exactness to each packet. The consumed coordinates of D and D' are precisely the finite diagonal windows selected by μ, μ' and bounded by τ, τ'. The hypotheses identify these consumed windows by componentwise hsame and match the dyadic tolerance rows on every displayed window used by the readback.

Use cofinal-tail stability to move later scheduled windows back to the same scoped tail route. Cofinality ensures that neither readback leaves the visible diagonal tail, and the tail-bound agreement keeps the two readbacks within the same tolerance ledger.

Finally apply two-axis seal factorization and the scoped diagonal route theorem. Once the diagonal readbacks expose the same finite tail route, the completion-facing reads must pass through

the corresponding R, R' rows and the real-facing reads through E, E', with both routes scoped by their displayed transport, replay, provenance, and local naming rows. Since neither carrier contains a coordinate for host convergence, quotient array equality, selected limits, or unscoped completeness, the conclusion is only equality of the visible completion-consumer and seal routes. □

Theorem 460.15 (CauchyDoubleSequence scoped non-escape)**.** *The scoped diagonal route of Definition 460.11 exports no host double-sequence convergence theorem, no quotient equality of two-parameter arrays, no selected double limit, and no unscoped real-completeness principle. A downstream* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives only the scheduled diagonal windows, the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the completion consumer row, the seal row, and the displayed* H, C, P, N *provenance.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceCarrier scoped nonescape`

Proof. Begin with the scoped diagonal route theorem. It says that every real-completion read factors through $X, \mu, \tau, D, R, E, H, C, P, N$.

Apply window diagonal exhaustion to the first stage. No consumer can inspect an array coordinate outside the windows selected by μ and bounded by τ.

Apply real seal non-escape to the final stage. The only real-facing coordinate is E, and it is reached through the completion consumer row R after the diagonal readback D.

Since the scoped route contains no coordinate for host convergence, quotient equality, selected limits, or unscoped completeness, those objects cannot be extracted from the packet. □

Theorem 460.16 (CauchyDoubleSequence seal-route determinacy)**.** *Let*

$$A = (X, \mu, \tau, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *packet. Any two* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads of* A *that use the same displayed rows* $X, \mu, \tau, D, R, E, H, C, P, N$ *factor through the same scoped diagonal route of Definition 460.11 and classify at the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Neither read may use host double-sequence convergence, quotient array equality, selected limits, or unscoped completeness to obtain a different seal classification.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceSealRouteDeterminacy`

Proof. First apply the scoped diagonal route theorem. It sends each accepted real-completion read through the displayed dependency surface

$$(X, \mu, \tau, H, C, P, N) \longmapsto D \longmapsto R \longmapsto E.$$

Next use two-axis seal factorization and completion consumer scope. They identify D as the only diagonal readback, R as the only completion-consumer row after D, and E as the only real-facing seal row after R.

Finally apply scoped non-escape. Since the packet has no coordinate for host convergence, quotient array equality, selected limits, or unscoped completeness, two reads that start from the same displayed rows cannot leave the route to produce different seal classifications. □

Theorem 460.17 (CauchyDoubleSequence diagonal-tail envelope)**.** *Every accepted two-parameter Cauchy array whose diagonal is consumed by a* $\mathsf{Real}^{\uparrow}$*-completion route exposes one finite tail envelope before the* $\mathsf{Real}^{\uparrow}$ *seal is read. The envelope contains the* $\mathsf{StreamName}^{\uparrow}$ *window row, the*

RegSeqRat$^{\uparrow}$ *readback row, the dyadic tail-bound row, the cofinal schedule row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. A downstream completion consumer may read the diagonal through that envelope and then through the seal row E, but it cannot bypass the envelope by host double-sequence convergence, quotient array equality, selected limits, or unscoped real completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDoubleSequenceUp.CauchyDoubleSequenceDiagonalTailEnvelope`

Proof. Start with the diagonal handoff and schedule-tail exactness theorems. They force every diagonal read to expose the scheduled window data and the dyadic tail-bound row before any regular rational readback is available.

Next apply cofinal-tail stability. Cofinal schedule changes remain inside the same displayed tail envelope, so the transported schedule row and tail-bound row still feed the same diagonal readback surface.

Use the scoped diagonal route to attach the structural rows. The componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row scope the stream window, regular rational readback, dyadic tail-bound, and cofinal schedule rows as one finite packet.

Finally apply seal-route determinacy. Once the envelope is fixed, the completion consumer reaches the same Real$^{\uparrow}$ seal row E and has no coordinate for host convergence, quotient equality, selected limits, or unscoped completeness. □

Closure status for NameCert$_{\text{CauchyDoubleSequence}^{\uparrow}}$.

Constructive story: A CauchyDoubleSequence$^{\uparrow}$ packet is generated from a two-parameter RegSeqRat$^{\uparrow}$ family, a CauchyRate$^{\uparrow}$ diagonal schedule, a DyadicTailBound$^{\uparrow}$ tolerance row, a diagonal RegSeqRat$^{\uparrow}$ readback, a RealCauchyCompletion$^{\uparrow}$ consumer, a Real$^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{CauchyDoubleSequence}^{\uparrow}}$, publicClosed)

Scope closed: Definition 460.11, Definition 460.12, Definition 460.13, Definition 460.15, and Definition 460.16 export the public StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ double-sequence route for the finite diagonal-array carrier, schedule, tolerance, readback, completion consumer, real seal, transport, continuation, package, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyDoubleSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Lean verification, bridge checking, host double-sequence convergence, quotient equality of arrays, selected limits, or unscoped real completeness.

Upgrade path: Lean verification and bridge checking require checked targets for the exported StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyRate$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, and RealCauchyCompletion$^{\uparrow}$ route.

Chapter 461

A Concrete Naming Certificate for $\mathsf{RegularCauchyProduct}^{\uparrow}$

The $\mathsf{RegularCauchyProduct}^{\uparrow}$ interface names the finite pointwise Cauchy-product surface for two regular rational Cauchy names. Bishop-Bridges constructive real arithmetic uses Cauchy products to make multiplication of regular real names visible. BEDC records the finite packet version: two regular source rows, aligned stream windows, dyadic endpoint multiplication, a product-error ledger, regular rational readback, componentwise transport, continuation routes, package provenance, and a local naming row.

The packet sits beside $\mathsf{RegularCauchySum}^{\uparrow}$ and $\mathsf{RegularCauchyDifference}^{\uparrow}$. It is not a completed-real multiplication law, not a quotient of Cauchy streams, not a selected host limit, and not an ambient completeness theorem.

Definition 461.1 (RegularCauchyProduct carrier)**.** *A* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose scheduled rational observations are multiplied pointwise;*
2. *W_A and W_B are aligned* $\mathsf{StreamName}^{\uparrow}$ *windows for a shared precision request and a shared unary observation index;*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from those windows;*
4. *D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *multiplication row obtained from the common-exponent product window for $D_A \cdot D_B$;*
5. *E is the finite* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *product ledger recording the two input tolerances, endpoint magnitude bounds, cross-error terms, and the output tolerance assigned to D;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose scheduled point row is D with the tolerance budget recorded by E;*
7. *H stores componentwise* hsame *transports for both source names, both stream windows, the three dyadic endpoint rows, the product-error ledger, and the readback row;*
8. *C stores the displayed* Cont *routes for reading the two input windows, forming the dyadic endpoint multiplication row, checking the product budget, and exposing the readback;*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}$ *row.*

The carrier is a finite multiplication certificate for regular Cauchy names. It stores no quotient stream, no selected limit, no host real equality, and no ambient completeness object.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductTasteGate single carrier alignment`

Theorem 461.2 (RegularCauchyProduct NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}$ *surface for the carrier of Definition 461.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is the displayed* BHist *packet M with two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic endpoint rows, a product-error ledger, transport rows, continuation rows, provenance, and local naming data;*

2. *classifier transport: classification transports the two source rows, the aligned windows, the endpoint multiplication row, the finite product-error ledger, and the readback row componentwise through the stored* hsame *and* Cont *rows;*

3. *stability: replacing either source by a classified regular name preserves the aligned window and recomputes the same kind of displayed dyadic multiplication row;*

4. *ledger exactness: the output tolerance row is the finite product-error budget recorded in E, including the two input tolerances and the displayed endpoint magnitude bounds;*

5. *downstream handoff: every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the product first reads R together with A, B, W_A, W_B, D_A, D_B, D, E, H, C, P, and N.*

Proof. Carrier admission is projection from the tuple in Definition 461.1. Each source row, window row, endpoint row, budget row, transport route, continuation route, provenance row, and local naming row is an explicit coordinate of M.

Classifier transport is componentwise. The two $\mathsf{RegSeqRat}^{\uparrow}$ source rows transport their scheduled rational observations, the $\mathsf{StreamName}^{\uparrow}$ rows transport the aligned window reads, and the $\mathsf{DyadicRatCore}^{\uparrow}$ rows transport the endpoint multiplication calculation at the common exponent.

The budget row is finite. It records the two input tolerances, the endpoint magnitude bounds used in the product estimate, the cross-error terms, and the output tolerance before the product readback is exposed. The downstream handoff is then the readback row R, and no omitted coordinate can add quotient streams, selected limits, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier namecert obligations`

Theorem 461.3 (RegularCauchyProduct window budget)**.** *Given accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows A and B in a* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet and a shared precision request, the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows W_A and W_B display dyadic endpoint rows D_A and D_B. The packet displays a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D for $D_A \cdot D_B$ and a finite ledger E whose output tolerance is the recorded product-error budget for the two source tolerances and endpoint magnitude bounds.*

Proof. Read the shared precision request through the two stream windows. Since W_A and W_B are aligned coordinates of the same packet, they expose the two endpoint rows at the same requested observation index.

Apply the displayed common-exponent multiplication row of $\mathsf{DyadicRatCore}^{\uparrow}$ to D_A and D_B. The result is the endpoint row D, still inside the carrier.

Finally read the finite budget ledger E. Its entries record the two input tolerances, the visible endpoint bounds, and the product-error terms assigned to the output request. This proves the window budget without invoking any completed real, quotient stream, or hidden limit. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier window budget`

Theorem 461.4 (RegularCauchyProduct RegSeqRat handoff)**.** *The budgeted pointwise-product windows of Definition 461.3 repack as the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R of Definition 461.1. A consumer of R reads only the aligned input windows, the dyadic endpoint multiplication row, the finite product-error ledger, the displayed* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

Proof. Start with the finite window budget. For each requested precision, the packet exposes the two source endpoint rows, the dyadic product endpoint, and the product-error row that bounds the output tolerance.

Repack those displayed rows as a regular rational sequence readback. The point row of R is D, and the regularity evidence visible to the consumer is the product ledger E routed through the same aligned windows.

The transport and continuation rows are inherited from the carrier. Since the repacking uses only coordinates already listed in Definition 461.1, it introduces no host sequence quotient, selected limit, or completed-real multiplication object. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier regseqrat handoff`

Theorem 461.5 (RegularCauchyProduct RealUp seal boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet reads only the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R together with the visible provenance of the two source names, aligned windows, dyadic endpoint multiplication row, finite product-error ledger,* hsame *transports,* Cont *routes,* Pkg *row, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}$ *row.*

Proof. Factor the real-facing read through the regular readback. By Definition 461.4, R is already the repacking of the budgeted pointwise-product windows.

The remaining consumer-visible data are exactly the provenance rows attached to that repacking: the two source names, aligned schedules, dyadic product endpoints, product-error ledger, transports, continuations, package row, and local naming row.

The carrier has no independent $\mathsf{Real}^{\uparrow}$ equality field and no completion witness outside the readback. Therefore the real seal can consume only the displayed regular readback and its visible provenance. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier real seal boundary`

Theorem 461.6 (RegularCauchyProduct source-window transport)**.** *For an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet $M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$, classifier transport preserves the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows A, B and the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows W_A, W_B through the displayed* hsame *and* Cont *routes. Consequently any transported packet reads the same two regular sources and the same aligned source windows before forming the dyadic product row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier source window transport`

Proof. Project the carrier of Definition 461.1. The two source rows and the two aligned stream windows are explicit coordinates of the same finite packet.

Apply the classifier-transport obligation of Definition 461.2. It transports the source rows and aligned windows componentwise through the stored hsame and Cont rows.

The product endpoint rows D_A, D_B, D are downstream of these transported windows. Since the transport first fixes A, B, W_A, W_B, the dyadic product row can only be read after the same two source rows and the same aligned window positions have been displayed. □

Theorem 461.7 (RegularCauchyProduct error-budget exactness)**.** *For an accepted* RegularCauchyProduct$^{\uparrow}$ *packet, the output tolerance assigned to the readback row R is exactly the finite product-error ledger E computed from the two input tolerances, the endpoint magnitude bounds for D_A and D_B, and the cross-error terms recorded in the packet. No additional tolerance rule is read from a quotient stream, a selected limit, or a completed real multiplication object.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier error budget exactness`

Proof. Use the window budget theorem Definition 461.3. It displays D_A and D_B from the aligned source windows and forms the dyadic endpoint product row D.

Read the ledger row E. By the NameCert obligations, E records the two input tolerances, the endpoint magnitude bounds, the cross-error terms, and the output tolerance before R is exposed.

Apply the RegSeqRat$^{\uparrow}$ handoff of Definition 461.4. The readback row R is repacked from D with the tolerance budget already recorded in E, so the output tolerance is exactly that finite ledger and nothing outside the displayed packet. □

Theorem 461.8 (RegularCauchyProduct downstream source lock)**.** *Every downstream* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* RegularCauchyProduct$^{\uparrow}$ *packet factors through*

$$(A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N).$$

Such a consumer can read the two source rows, aligned windows, dyadic endpoint rows, finite product-error ledger, regular readback, transport rows, continuation routes, package provenance, and local naming row, but cannot read quotient streams, selected limits, host real equality, or completed real multiplication.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier downstream source lock`

Proof. Start with the regular handoff. By Definition 461.4, every RegSeqRat$^{\uparrow}$ readback factors through the aligned input windows, dyadic endpoint multiplication row, finite product-error ledger, transports, continuations, package row, and local naming row.

Use Definition 461.6 to lock the source rows. The aligned windows available to any transported consumer are still the windows attached to A and B.

Finally apply the RealUp seal boundary of Definition 461.5. A real-facing consumer reads only the regular readback and its visible provenance, so neither a quotient stream nor a selected limit nor host real equality nor completed multiplication can enter downstream. □

Theorem 461.9 (RegularCauchyProduct public export certificate)**.** *The carrier, local* NameCert *obligations, window budget,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal boundary, source-window transport, error-budget exactness, and downstream source lock assemble into one public* RegularCauchyProduct$^{\uparrow}$ *export packet. Its exported rows are exactly*

$$A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N.$$

The packet exports no quotient stream, selected limit, host real equality, ambient completeness witness, or completed-real multiplication law.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier public export certificate`

Proof. Start with the displayed carrier of Definition 461.1. It supplies exactly the two regular source rows, two aligned source-window rows, dyadic endpoint rows, product row, finite error ledger, regular readback, transports, continuations, provenance, and local name.

Add the local acceptance and budget rows. The NameCert obligations of Definition 461.2 bind the accepted packet, while Definition 461.3 computes the endpoint and product rows from the aligned windows.

Next expose the two consumer boundaries. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is supplied by Definition 461.4, and the $\mathsf{Real}^{\uparrow}$ seal boundary is supplied by Definition 461.5.

Finally use Definition 461.6, Definition 461.7, and Definition 461.8. These results keep every downstream read on the displayed source windows and finite error ledger, and they exclude quotient streams, selected limits, host real equality, ambient completeness, and completed-real multiplication from the public export. □

Theorem 461.10 (RegularCauchyProduct bridge readiness)**.** *The public* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *export packet of Definition 461.9 is bridge-ready for downstream real-algebra consumers. Every such consumer factors through the carrier* $M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$*, the local* NameCert *obligations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the public export certificate before it may read product data. The bridge-ready surface exports exactly the regular-Cauchy product boundary and no host multiplication over completed reals, quotient stream equality, selected limit, ambient completeness, or unlisted algebra law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier bridge readiness`

Proof. Start with the public export certificate. It assembles the carrier, local naming obligations, window budget, regular readback handoff, real-seal boundary, source-window transport, error-budget exactness, and downstream source lock into one public packet.

Use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff to identify the product datum read by downstream consumers. The handoff row R is obtained from the aligned windows, dyadic endpoint multiplication row, and finite product-error ledger, so it is already a regular rational product route.

Next apply the $\mathsf{Real}^{\uparrow}$ seal boundary and the real algebra boundary. They force real-facing consumers to read the same regular readback and visible provenance rather than a completed-real multiplication operation.

The public export certificate then supplies the bridge boundary: all product reads pass through $A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N$. Since no excluded algebraic or completion object is a carrier coordinate, the public surface is ready to be the source side of a standard bridge for real-algebra consumers. □

Theorem 461.11 (RegularCauchyProduct real algebra boundary)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *algebra consumer of an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet*

$$M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$$

must read the source-window rows A, B, W_A, W_B*, the finite error-budget ledger* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations* D_A, D_B, D*, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, the displayed* hsame *transports* H*, the finite* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}$

row N. It receives no quotient stream, host multiplication over completed reals, selected limit, ambient completeness, or unlisted algebra law.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier real algebra boundary`

Proof. Start with the NameCert obligations of Definition 461.2. They admit only the displayed packet M and force every accepted read through the two source rows, aligned windows, endpoint rows, product ledger, transports, routes, provenance, and local name.

Apply the window budget theorem Definition 461.3. It builds the dyadic product row from the two source windows and records the product tolerance in the finite ledger E before any downstream algebra consumer sees the readback.

Next use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff Definition 461.4 together with the $\mathsf{Real}^{\uparrow}$ seal boundary Definition 461.5. The real-facing route is the regular readback R with its visible product provenance, not a host multiplication operation over completed reals.

Finally combine error-budget exactness, downstream source lock, and the public export certificate. These results exhaust the consumer-visible rows and exclude quotient streams, selected limits, ambient completeness, host real equality, completed-real multiplication, and unlisted algebra laws from the product boundary. □

Theorem 461.12 (RegularCauchyProduct bounded-source handoff)**.** *Let*

$$M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet. If the two source windows W_A and W_B expose dyadic endpoint magnitude bounds in the finite product-error ledger E, then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the product factors through W_A, W_B, the dyadic product row D, and the same ledger row E. No readback may replace those bounded windows by a quotient stream, selected limit, host real equality, ambient completeness, or completed-real multiplication law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier bounded source handoff`

Proof. Start with Definition 461.3. The product row D is computed from the displayed source windows and the dyadic endpoint rows, and the product tolerance is recorded in the finite ledger E.

Use Definition 461.6. Transported consumers still see the windows attached to the source rows A and B, so the endpoint magnitude bounds remain the bounds supplied by W_A and W_B.

Apply Definition 461.7. The error budget exposed to a readback is exactly the displayed product ledger, with no additional completed-real algebra coordinate.

Finally use Definition 461.8. Any downstream $\mathsf{RegSeqRat}^{\uparrow}$ handoff must keep the same source windows, dyadic product row, and ledger row E, which gives the stated bounded-source factorization. □

Theorem 461.13 (RegularCauchyProduct real seal product budget)**.** *Every* $\mathsf{Real}^{\uparrow}$ *algebra consumer of the product seal in an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet receives the finite product-budget row E before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R. The consumer cannot replace this budget by completed-real multiplication, quotient stream equality, selected limits, ambient completeness, host real equality, or any unlisted algebra law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier real seal product budget`

Proof. First apply Definition 461.5. The real-facing endpoint is read through the visible product provenance and regular readback route, not through completed-real multiplication.

Next use Definition 461.11. A $\mathsf{Real}^{\uparrow}$ algebra consumer must read the finite error ledger E, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, the dyadic product row, transports, continuations, provenance, and local name.

The public export certificate Definition 461.9 supplies the same row order at the public surface. The packet contains the product ledger and readback row, but no quotient stream, selected limit, ambient completeness, or host real equality.

Finally invoke Definition 461.12. Since every readback factors through the bounded source windows, dyadic product row, and ledger E, the real seal consumer receives the finite product budget before it can consume the regular rational handoff. □

Theorem 461.14 (RegularCauchyProduct window comparison)**.** *Let M be an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet, let S be an accepted* $\mathsf{RegularCauchySum}^{\uparrow}$ *packet, and let Q be an accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *packet. Suppose the three packets share the same two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows A, B and the same aligned* $\mathsf{StreamName}^{\uparrow}$ *source windows W_A, W_B. Then every product consumer must first read the common finite boundary*

$$(A, B, W_A, W_B)$$

before diverging to the product-error ledger E of M, while the sum and difference consumers diverge from that same boundary to their own finite arithmetic ledgers. The conclusion is a comparison of finite-window routes; it is not a completed-real algebra law.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier window comparison`

Proof. Start with the product carrier of Definition 461.1. Its first four rows are the two regular source rows and the two aligned source-window rows, so any product read must expose A, B, W_A, W_B before it reaches dyadic endpoint multiplication or the product ledger.

Use the source-window transport theorem Definition 461.6. Transport preserves the same source rows and aligned windows through the displayed hsame and Cont routes, so the shared boundary remains the first visible route segment for every transported product consumer.

Now apply the real algebra boundary and product-budget results, Definition 461.11 and Definition 461.13. They force the product route to diverge from the shared source-window boundary into the finite product-error ledger E before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal are consumed.

Finally compare the sibling surfaces chapter 417 and chapter 429. Their accepted packets read the same kind of finite source-window boundary for the same two source rows, but their next arithmetic coordinates are the finite sum and difference ledgers rather than E. Thus the comparison is a route comparison over shared windows and sibling ledgers, with no quotient stream, selected limit, host real equality, ambient completeness, or field multiplication law. □

Theorem 461.15 (RegularCauchyProduct coordinate regularity)**.** *For every accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet*

$$M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N),$$

the readback row R is regular by finite coordinates alone. Each requested product endpoint row D is obtained from the aligned $\mathsf{StreamName}^{\uparrow}$ *windows W_A, W_B, and the ledger E records the two source tolerances, the endpoint magnitude bounds for D_A, D_B, and the cross-error terms needed for the output tolerance. Thus the regularity of R is witnessed by the carrier coordinates, the local* $\mathsf{NameCert}$ *obligations, the window budget, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, source-window transport, error-budget exactness, and downstream source lock, with no appeal to a quotient stream, selected limit, host real equality, ambient completeness, or completed-real multiplication.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier coordinate regularity`

Proof. Project the carrier of Definition 461.1. The coordinates $A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N$ include the two regular sources, the aligned windows, the dyadic endpoint rows, the product row, and the finite budget ledger.

Use the local NameCert obligations and the window budget theorem, Definition 461.2 and Definition 461.3. They force each requested product endpoint to be read from the aligned $\mathsf{StreamName}^{\uparrow}$ windows and place the two source tolerances, endpoint bounds, and cross-error terms in E.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal boundary, Definition 461.4 and Definition 461.5. The row R is the regular rational readback of these finite product windows, while the real-facing boundary exposes only the same finite provenance.

Finally use source-window transport, error-budget exactness, and downstream source lock. These keep every transported consumer on the displayed windows and ledger, so the regularity witness for R is the finite product-coordinate route itself. □

Theorem 461.16 (RegularCauchyProduct classifier stability)**.** *Let*

$$M = (A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet. Classifier transport of either source row preserves the aligned source windows, the dyadic endpoint product row, the finite error ledger, and the readback route through the stored* hsame *and* Cont *coordinates. Consequently replacing A or B by a classified regular source does not change the finite product boundary read by R; it only replays the carrier,* NameCert *obligations, window budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, source-window transport, error-budget exactness, and downstream source lock along the displayed transport rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier classifier stability`

Proof. Start from the classifier-transport clause in Definition 461.2. It transports the two source rows, aligned windows, endpoint rows, finite product-error ledger, and readback row componentwise through the stored hsame and Cont coordinates.

Use Definition 461.6. Transported packets still display the same source-window boundary A, B, W_A, W_B before any dyadic endpoint product row can be consumed.

Next apply the window budget and error-budget exactness results. The product row D and ledger E remain the rows obtained from the transported aligned windows, and the output tolerance is still exactly the displayed product ledger.

Finally combine the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal boundary, and downstream source lock. They force the transported readback to consume the same finite product boundary, so classifier stability is a stability claim about the displayed coordinates rather than a completed-real algebra law. □

Theorem 461.17 (RegularCauchyProduct RealUp pair seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of a product pair sealed by an accepted* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *packet can read only the two source regular names A, B, the regular readback row R, and the finite product provenance carried by $W_A, W_B, D_A, D_B, D, E, H, C, P, N$. The seal exposes the carrier,* NameCert *obligations, window budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, source-window transport, error-budget exactness, and downstream source lock, but it does not expose host real multiplication, a quotient stream, a selected limit, an ambient completeness witness, or any extra algebra law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductUp.RegularCauchyProductCarrier real pair seal`

Proof. Begin with the carrier and local NameCert obligations. The accepted packet contains exactly the two source rows, aligned windows, dyadic endpoint rows, product ledger, readback, transports, continuations, package provenance, and local name.

Apply the window budget and $\mathsf{RegSeqRat}^{\uparrow}$ handoff. They show that the product readback R is assembled from the finite endpoint rows and product-error ledger, so the consumer reads a regular rational product route rather than a completed-real multiplication operation.

Use the $\mathsf{Real}^{\uparrow}$ seal boundary together with source-window transport. The real-facing consumer sees the same two source names and aligned windows after transport, and no additional real-valued coordinate is introduced.

Finally invoke error-budget exactness and downstream source lock. They exhaust the visible product provenance and rule out quotient streams, selected limits, host real equality, ambient completeness witnesses, host multiplication, and unlisted algebra laws at the pair seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProduct}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, aligned $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint multiplication rows, a finite $\mathsf{FiniteErrorBudget}^{\uparrow}$ product ledger, displayed hsame transports, Cont routes, Pkg provenance, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature public export surface records the finite carrier, local naming obligations, shared-window product budget, regular readback handoff, real-seal boundary, source-window transport, error-budget exactness, downstream source lock, public export certificate, and bridge-readiness theorem for pointwise products of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, selected limits, host real equality, ambient completeness, real-field multiplication laws, or bridge checking.

Upgrade path: Verification closure requires Lean synchronization for the mature public regular-Cauchy product boundary and any standard bridge target recorded by the formalization pipeline.

Chapter 462

A Concrete Naming Certificate for $\mathsf{CauchyModulusMeet}^{\uparrow}$

The $\mathsf{CauchyModulusMeet}^{\uparrow}$ interface packages the finite rows needed when two regular rational stream names must be read through one shared Cauchy bound. It is a naming certificate for the shared-modulus packet used by diagonal-limit and uniform Cauchy consumers; it does not assert that a quotient stream, a selected limit, or an ambient completeness principle has already been constructed.

The meet packet consumes $\mathsf{StreamName}^{\uparrow}$ finite-window rows from chapter 126, regular rational readbacks from chapter 301, dyadic tolerance ledgers from chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal surface from chapter 38. Thus $\mathsf{CauchyModulusMeet}^{\uparrow}$ is a shared-bound packet over these sibling carriers, not a kernel-only surface.

Definition 462.1 (CauchyModulusMeet carrier)**.** *A* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$$

with the following displayed coordinates:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *rows whose rational windows are read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data;*
2. μ_0 *and* μ_1 *are the two source* $\mathsf{CauchyModulus}^{\uparrow}$ *rows attached to those regular rational names;*
3. μ *is the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *meet row, displayed with a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger sufficient for both source rows;*
4. H *stores componentwise* hsame *transports for the two stream names, two source moduli, shared meet modulus, dyadic tolerance rows, real-facing reads, provenance, and local name;*
5. C *stores the displayed* Cont *routes by which consumers inspect stream windows, regular rational rows, source modulus rows, the shared meet bound, and the real-facing handoff;*
6. P *is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting these displayed coordinates to a common stream-name, modulus, and dyadic tolerance surface. The carrier has no coordinate for quotient stream equality, selected limits, host real equality, or ambient completeness.

Theorem 462.2 (CauchyModulusMeet NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$ *surface for the carrier of Definition 462.1 consists exactly of five rows:*

1. *carrier admission for the finite packet* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$*;*
2. *classifier transport to a common displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *surface;*
3. *stability of both source modulus projections and of the shared meet row under componentwise* hsame *transport;*
4. *exactness of the shared dyadic tolerance ledger as a bound readable by both source modulus rows;*
5. *ledger rows for* Cont *routes,* Pkg *provenance,* $\mathsf{Real}^{\uparrow}$ *handoff reads, and the local naming certificate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket namecert obligations`

Proof. Carrier admission is projection from Definition 462.1. The displayed coordinates provide the two source stream names, two source moduli, shared meet modulus, transport rows, continuation rows, provenance, and local name.

Classifier transport uses H to move every coordinate to a common finite surface. Once transported, the routes in C are the only ways to read stream windows, regular rational rows, source moduli, the shared meet bound, and the real-facing handoff.

The dyadic tolerance ledger is stored inside the shared row μ and its associated $\mathsf{DyadicRatCore}^{\uparrow}$ read. Since no carrier coordinate stores quotient streams, selected limits, host real equality, or an ambient completeness proof, the NameCert obligations are exhausted by the five displayed rows. □

Theorem 462.3 (CauchyModulusMeet shared-radius carrier admission)**.** *Carrier admission for a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet*

$$M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$$

exposes exactly the two $\mathsf{StreamName}^{\uparrow}$ *rows* S_0, S_1*, the two source* $\mathsf{CauchyModulus}^{\uparrow}$ *rows* μ_0, μ_1*, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance meet carried by* μ*, the componentwise* hsame *rows in* H*, the displayed* Cont *routes in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$ *row* N*. No carrier-admission route adds quotient stream equality, selected limits, host real equality, or ambient completeness.*

Proof. Start with the carrier of Definition 462.1. Its displayed coordinates are precisely $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$, with S_0, S_1 providing the stream-name rows and μ_0, μ_1, μ providing the two source modulus rows and the shared meet row.

Apply the NameCert obligation surface of Definition 462.2. The carrier-admission row is one of the five local obligations, and the remaining rows only classify, stabilize, bound, route, record provenance, and name the same finite packet.

Projection stability and shared-bound exhaustion identify the readable modulus content. Definition 462.6 keeps the source projections on the displayed transport surface, while Definition 462.9 confines the shared bound to the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger carried by μ. Hence carrier admission exposes exactly the listed BEDC rows and nothing external to the packet. □

Theorem 462.4 (CauchyModulusMeet RegSeqRat classifier transport)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier comparison between accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packets factors through transported stream windows, the two source modulus rows, the shared meet row* μ*, and the displayed* hsame *and* Cont *routes of the carrier in Definition 462.1. The comparison may use the* Pkg *and* NameCert *rows to identify the packet provenance and local name, but it has no access to quotient stream equality, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket regseqrat classifier transport`

Proof. Read the classifier row from Definition 462.2. It transports the compared packets to a common displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ surface before any classifier read is made.

Now use the carrier coordinates of Definition 462.1. The only stream-window inputs are the windows routed from S_0 and S_1, the only source modulus inputs are μ_0 and μ_1, and the only common bound input is the shared meet row μ with its dyadic ledger.

Finally apply projection stability and the shared-bound exhaustion theorem. Definition 462.6 keeps the two source modulus reads stable under the displayed hsame transports, and Definition 462.9 prevents the classifier from replacing μ by an undisplayed analytic datum. The classifier comparison therefore factors through the listed transported routes. □

Theorem 462.5 (CauchyModulusMeet Real seal consumer obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet may read only the shared-bound route exposed by*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

Such a consumer cannot extract quotient streams, selected limits, host real equality, ambient completeness, or any real-valued seal beyond the displayed shared modulus handoff.

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket real seal consumer obligation`

Proof. Begin with the carrier in Definition 462.1. The packet contains stream names, regular-rational read routes, source modulus rows, the shared meet row, transport rows, continuation routes, provenance, and a local name, but no host real object or completeness witness.

Use the NameCert obligations to identify the available consumer rows. By Definition 462.2, the real-facing handoff reads are ledger rows over the same finite packet, not extra coordinates.

Apply the Real handoff boundary and shared-bound exhaustion. Definition 462.7 forces each $\mathsf{Real}^{\uparrow}$ read through the stream-name, regular-rational, dyadic-tolerance, transport, route, provenance, and naming rows, while Definition 462.9 identifies the only readable bound as the displayed ledger carried by μ.

Thus the consumer obligation is exactly the shared-bound route. Since neither the carrier nor the cited obligations contain quotient streams, selected limits, host real equality, ambient completeness, or a separate real-valued seal, none can be extracted by a $\mathsf{Real}^{\uparrow}$ consumer. □

Theorem 462.6 (CauchyModulusMeet projection stability)**.** *Let* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$ *and* $M' = (S'_0, S'_1, \mu'_0, \mu'_1, \mu', H', C', P', N')$ *be accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carriers whose displayed coordinates are related componentwise by the transport rows in* H *and* H'*. Then the projected source modulus reads* μ_0, μ_1 *and the shared meet read* μ *remain readable through the transported packet, and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger is the same displayed ledger seen by both* $\mathsf{RegSeqRat}^{\uparrow}$ *stream-name projections.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket projection stability`

Proof. Project both accepted carriers and read the componentwise transport rows. The only modulus coordinates in either packet are the two source rows and the shared meet row.

Use the displayed hsame entries in H and H' to transport $S_0, S_1, \mu_0, \mu_1, \mu$ to the common surface. The continuation rows then read the transported stream windows and regular rational rows before reading the transported source moduli.

It remains to compare the shared bound. The dyadic tolerance ledger is part of the displayed meet row and is routed by C rather than by an external real-completeness object. Hence each projected source modulus remains readable through the meet packet after transport, and both projections see the same displayed shared bound. □

Theorem 462.7 (CauchyModulusMeet Real handoff boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *diagonal consumer of an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carrier may read the shared bound only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *rows* S_0, S_1*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readbacks of those rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carried by* μ*, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket real handoff boundary`

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ diagonal consumer route. The carrier has no real-facing coordinate other than the handoff reads reached through the stream-name, regular rational, and shared modulus routes.

The consumer first reads S_0 and S_1, then their regular rational windows, and then the shared dyadic tolerance ledger inside μ. The two source moduli μ_0 and μ_1 are available only as displayed projections through the same finite packet.

All remaining movement is supplied by the rows H, C, P, N. These rows transport, route, record provenance, and name the packet; they do not add quotient streams, selected limits, ambient completeness, or host real equality. Therefore the real-facing handoff is bounded by the displayed certificate surface. □

Theorem 462.8 (CauchyModulusMeet swap stability)**.** *Let* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carrier, and let*

$$M^{\mathrm{sw}} = (S_1, S_0, \mu_1, \mu_0, \mu, H^{\mathrm{sw}}, C^{\mathrm{sw}}, P, N)$$

be the packet obtained by exchanging the two displayed source stream/modulus rows and transporting every affected row componentwise by the hsame *entries recorded in* H*. Then* M^{sw} *is accepted by the same carrier surface, and it presents the same shared meet ledger* μ *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows to every displayed* Cont *route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket swap stability`

Proof. Project the accepted packet through Definition 462.1. The first two stream coordinates and the first two source modulus coordinates are the only ordered source-side entries of the carrier.

Exchange these two pairs and read the corresponding componentwise transport rows from H. The transported packet still has two $\mathsf{StreamName}^{\uparrow}$ rows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two source $\mathsf{CauchyModulus}^{\uparrow}$ rows, one shared meet row, the same provenance row, and the same local name row.

The shared row μ is not duplicated or replaced by the swap. Its dyadic tolerance ledger remains the displayed ledger routed by C, while H^{sw} and C^{sw} only retarget the source-side reads. Hence the swapped packet is accepted and exposes the same shared meet surface. □

Theorem 462.9 (CauchyModulusMeet shared-bound exhaustion)**.** *For an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carrier* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$*, every readable shared bound exported by the packet is one of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows carried by* μ*. No exported shared-bound route can use quotient stream equality, a selected limit, host real equality, or an ambient completeness datum.*

Proof. Start from an exported shared-bound route. By the carrier definition, the only bound coordinate common to the two source modulus rows is the meet row μ together with its displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger.

The continuation row C reads source stream windows, regular rational rows, source modulus rows, the shared meet bound, and the real-facing handoff. The transport row H only moves these displayed coordinates to a common surface.

There is no carrier coordinate for quotient streams, selected limits, host real equality, or ambient completeness. Therefore a readable shared bound must be a displayed tolerance row inside μ, recorded through H, C, P, N and nothing outside the finite packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket shared bound transport`

Theorem 462.10 (CauchyModulusMeet consumer factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer route out of an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *carrier factors through the ordered rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

In particular, such a consumer first reads the displayed stream-name rows, the source modulus rows, and the shared meet row, and then uses only the componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$ *row supplied by the packet.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket consumer factorization`

Proof. Let a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer route be given. Project its source carrier to the tuple of Definition 462.1.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, the stream-name and rational readback surface is reached through S_0 and S_1, with source bounds supplied by μ_0, μ_1 and their common bound supplied by μ. For a $\mathsf{Real}^{\uparrow}$ consumer, Definition 462.7 forces the real-facing read through the same rows before any handoff is visible.

The remaining route data are exactly H, C, P, N. These rows transport, route, record provenance, and name the packet; they do not provide an independent consumer object. Thus every such route factors through the displayed ordered carrier rows. □

Theorem 462.11 (CauchyModulusMeet obligation surface)**.** *The carrier, projection stability, swap stability, shared-bound exhaustion,* $\mathsf{Real}^{\uparrow}$ *handoff boundary, and consumer factorization of the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet constitute the full obligation surface for meeting two displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows over a shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. Downstream* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers may use only this finite surface.*

The surface exports no quotient stream equality, selected limit, ambient completeness datum, host real equality, or public standard bridge.

Proof. First project the carrier of Definition 462.1. It lists the two $\mathsf{StreamName}^{\uparrow}$ rows, their two source $\mathsf{CauchyModulus}^{\uparrow}$ rows, the shared meet row, the transport row, continuation row, provenance row, and local name.

Projection stability is supplied by Definition 462.6, and source-row exchange is supplied by Definition 462.8. Together these bind the two source moduli and their shared meet row under the permitted componentwise hsame transports.

The shared-bound and consumer-facing parts are then fixed by Definition 462.9, Definition 462.7, and Definition 462.10. These three results force $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance reads, $\mathsf{RegSeqRat}^{\uparrow}$ route reads, and $\mathsf{Real}^{\uparrow}$ handoff reads through the displayed meet packet.

Finally the NameCert obligations in Definition 462.2 record the same rows as the local naming surface. Since none of the cited blocks introduces quotient stream equality, selected limits, ambient completeness, host real equality, or a standard bridge, the listed blocks are the full obligation surface. □

Theorem 462.12 (CauchyModulusMeet public export certificate)**.** *The carrier, projection stability, swap stability, shared-bound exhaustion,* $\mathsf{Real}^{\uparrow}$ *handoff boundary, consumer factorization, and obligation surface assemble one public* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *certificate over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This certificate exports exactly the finite rows

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

It does not export quotient stream equality, selected limits, host real equality, ambient completeness, or a standard bridge.

Proof. Begin with the carrier of Definition 462.1. It supplies the two $\mathsf{StreamName}^{\uparrow}$ rows, their $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the two source $\mathsf{CauchyModulus}^{\uparrow}$ rows, the shared meet row, the transport row, the continuation row, the provenance row, and the local NameCert row.

Add the local stability surface. Projection stability is supplied by Definition 462.6, source-row exchange by Definition 462.8, and shared-bound exhaustion by Definition 462.9.

Add the consumer-facing surface. The $\mathsf{Real}^{\uparrow}$ handoff boundary is supplied by Definition 462.7, consumer factorization by Definition 462.10, and the finite obligation inventory by Definition 462.11.

Each cited block is stated over the same displayed packet and refuses quotient stream equality, selected limits, host real equality, ambient completeness, and bridge content. Therefore the assembled public certificate has exactly the named surface and no additional export. □

Theorem 462.13 (CauchyModulusMeet public export surface)**.** *The* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *public export packet is the finite shared-modulus surface over*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

It assembles the carrier admission, projection stability, $\mathsf{Real}^{\uparrow}$ *handoff boundary, swap stability, shared-bound exhaustion, consumer factorization, and obligation surface into one readable certificate for downstream* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers.*

The export packet carries no quotient stream equality, selected limit, ambient completeness datum, host real equality, or standard bridge.

Proof. Begin with the carrier of Definition 462.1. It supplies exactly the two stream-name rows, two source modulus rows, shared meet row, transport row, continuation row, provenance row, and local naming row.

Add the stability and boundary components. Projection stability comes from Definition 462.6, real-facing access from Definition 462.7, source-row exchange from Definition 462.8, and shared-bound exhaustion from Definition 462.9.

Add consumer routing and local obligations. Consumer factorization is supplied by Definition 462.10, and the complete obligation inventory is supplied by Definition 462.11.

Each cited result is stated over the same finite rows and explicitly refuses quotient streams, selected limits, ambient completeness, host real equality, and a standard bridge. Therefore the public export packet is exactly the assembled shared-modulus surface. □

Theorem 462.14 (CauchyModulusMeet export consumer lock)**.** *Every downstream consumer using the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *public export packet is locked to the shared modulus rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

A $\mathsf{RegSeqRat}^{\uparrow}$ *read,* $\mathsf{Real}^{\uparrow}$ *handoff,* $\mathsf{CauchyModulus}^{\uparrow}$ *projection, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance read must factor through the public export surface of Definition 462.13.*

Proof. Start with a downstream consumer of the export packet. By Definition 462.13, the only public rows are $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$.

For $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, apply Definition 462.10 and Definition 462.7. These force the read through the displayed stream-name, source-modulus, shared-meet, transport, route, provenance, and naming rows.

For $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ consumers, apply Definition 462.6 and Definition 462.9. These force modulus projections and tolerance reads through the same shared meet row μ and its displayed dyadic ledger. □

Theorem 462.15 (CauchyModulusMeet tailmeet handoff)**.** *An accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$ *supplies the shared-bound rows consumed by* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*. The tail-meet consumer may read the two* $\mathsf{StreamName}^{\uparrow}$ *rows, the two source* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, the shared meet row* μ*, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transports,* Cont *routes, package provenance, and local* $\mathsf{NameCert}$ *row, and it has no tail oracle outside the public export packet.*

Proof. Start with the public export surface of Definition 462.13. It exposes exactly $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ for downstream consumers.

Use projection stability and shared-bound exhaustion. Definition 462.6 keeps the two source modulus rows and the shared meet row readable under the displayed transports, while Definition 462.9 identifies the shared-bound read with μ and its dyadic tolerance ledger.

Finally apply the real handoff boundary and export consumer lock. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ read is a downstream shared-bound consumer, so it is forced through the same packet and cannot import quotient stream equality, selected limits, host real equality, ambient completeness, or an external tail oracle. □

Theorem 462.16 (CauchyModulusMeet diagonal window route)**.** *For an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$*, every diagonal-window consumer that later feeds a* $\mathsf{Real}^{\uparrow}$ *read uses the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows exposed by the public packet. The route passes through* $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ *and not through quotient streams, selected limits, host real equality, ambient completeness, or a separate completion witness.*

Proof. Begin with consumer factorization and the real handoff boundary. They route every real-facing read through the displayed stream names, source modulus rows, shared meet row, transports, continuations, provenance, and local naming row.

Then use shared-bound exhaustion. The dyadic tolerance read for the diagonal window is the ledger attached to the same meet row μ, while the source windows remain the public rows S_0 and S_1.

The public export surface and export consumer lock finish the routing claim. They prevent a diagonal-window consumer from choosing independent $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{DyadicRatCore}^{\uparrow}$ data outside M. □

Theorem 462.17 (CauchyModulusMeet comparison synchronization)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet* $M = (S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N)$ *synchronizes on the displayed meet row* μ*. In particular, tail-meet consumers and diagonal-window consumers compare source rows through the same shared-bound row, componentwise transports, continuation routes, package provenance, and local naming row, rather than through an external tail oracle.*

Proof. First compose projection stability with swap stability. These results say that either ordering of the source rows reads the same source-modulus projections and the same shared meet row under the displayed transports.

Next use shared-bound exhaustion together with the public export surface. The comparison row available to downstream consumers is the meet row μ and its dyadic tolerance ledger, packaged with H, C, P, N.

Finally apply the tailmeet handoff and diagonal-window route theorems above. Both downstream consumer classes are already locked to M, so their comparison synchronization occurs on the displayed meet row and cannot be delegated to quotient streams, selected limits, host real equality, ambient completeness, or an external tail oracle. □

Theorem 462.18 (CauchyModulusMeet tail-threshold compatibility)**.** *The public export rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$$

of a $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet supply the shared tail-threshold route consumed by* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*. The route reads the two source stream rows, the two source modulus rows, the shared meet row, the componentwise transport, continuation routes, package provenance, and local naming row, and it has no independent tail oracle outside the public packet.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket tail threshold compatibility`

Proof. Start with the carrier of Definition 462.1. It gives the two stream-name rows, the two source modulus rows, the meet modulus row, the transport row, the continuation row, the provenance row, and the local $\mathsf{NameCert}$ row.

Use the NameCert obligations of Definition 462.2 and the projection stability of Definition 462.6. They make the source moduli and the meet row stable under the displayed hsame transport.

The shared-bound exhaustion theorem Definition 462.9 then supplies the tail-threshold read through μ. By the public export surface and export consumer lock, a $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ consumer must read that threshold through the displayed packet and cannot add a quotient stream, selected limit, host real equality, ambient completeness, or independent tail oracle. □

Theorem 462.19 (CauchyModulusMeet diagonal-window consumer handoff)**.** *The public export rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$$

of a $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet supply the selected-window route consumed by* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$*. The selected-window consumer receives only the shared bound, the two source stream/modulus rows,*

the displayed dyadic tolerance ledger, componentwise transport, continuation routes, package provenance, and local naming row.

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket diagonal window consumer handoff`

Proof. Begin with carrier admission and the NameCert obligation surface. These fix the finite packet over the two stream rows, two source modulus rows, shared meet row, transport, continuation, provenance, and naming coordinates.

Apply projection stability and shared-bound exhaustion. The diagonal-window read can choose its window only through the stable source projections and the shared bound supplied by μ.

Now apply consumer factorization, the $\mathsf{Real}^{\uparrow}$ handoff boundary, the public export surface, and the export consumer lock. These route the $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ consumer through the public shared-modulus packet and exclude quotient streams, selected limits, host real equality, ambient completeness, and any independent tail oracle. □

Theorem 462.20 (CauchyModulusMeet downstream unblock package)**.** *The public* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *export is a finite downstream handoff package for the real-completion route*

$$\mathsf{RegularCauchyTailMeet}^{\uparrow}, \quad \mathsf{RealDiagonalWindowCoverage}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Its rows $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ *provide the shared tail-threshold route, the selected-window route, and the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff boundary, with no quotient stream equality, selected limit, host real equality, ambient completeness, or independent tail oracle.*

Proof. First assemble the local packet from the carrier, NameCert obligations, and projection stability. This gives the displayed rows and keeps source-modulus reads stable under componentwise transport.

Next attach the tail and window routes. Shared-bound exhaustion gives the tail-threshold route of Definition 462.18, while the same shared meet row supplies the selected-window handoff of Definition 462.19.

Finally add the consumer-facing boundary. The $\mathsf{Real}^{\uparrow}$ handoff boundary and consumer factorization route $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers through the same finite packet, and the public export surface with the export consumer lock prevents any downstream consumer from importing quotient streams, selected limits, host real equality, ambient completeness, or independent tail oracles. □

Theorem 462.21 (CauchyModulusMeet export non-escape)**.** *No consumer of the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *public export packet can extract quotient stream equality, a selected limit, ambient completeness, host real equality, or a standard bridge. Every public read is exhausted by the carrier rows, projection stability, real handoff boundary, swap stability, shared-bound exhaustion, consumer factorization, and local obligation surface assembled in Definition 462.13.*

Proof. Inspect the export packet given by Definition 462.13. Its coordinates are exactly $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ together with the cited stability, boundary, routing, and obligation facts.

Apply Definition 462.14. Any downstream consumer is forced through those same displayed rows, so no consumer route can introduce an independent quotient, limit, host-real, completion, or bridge coordinate.

The exclusions then follow from Definition 462.11. That theorem already states that the shared-modulus obligation surface exports none of the forbidden objects, and the public export packet adds no row beyond that surface. □

Theorem 462.22 (CauchyModulusMeet source-pair exposure)**.** *The public* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *export exposes exactly the two source stream/modulus rows and the shared meet row:*

$$(S_0, \mu_0), \quad (S_1, \mu_1), \quad \mu.$$

Every exposed source-pair read is accompanied by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row of the packet.*

Proof. Start with Definition 462.1. The only source-side rows in the carrier are S_0, S_1, μ_0, μ_1, and the only shared bound row is μ.

Definition 462.6 preserves the two source modulus projections and the shared meet read under componentwise transport. Definition 462.9 identifies the readable shared bound with the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger carried by μ.

Finally apply Definition 462.13. The public packet adds only the transport, continuation, provenance, and local naming rows, so the exposed source-pair surface is exactly the two source pairs plus the shared meet row. □

Theorem 462.23 (CauchyModulusMeet shared-bound consumer read)**.** *Any downstream shared-bound read from a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet factors through the public export surface of Definition 462.13. The read may use the two source stream/modulus rows, the shared meet row, the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise transports, continuation routes, package provenance, and local naming row, and no other route.*

Proof. Let a downstream shared-bound read be given. By Definition 462.14, every downstream consumer is locked to $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$.

Definition 462.22 identifies the source-pair rows and the shared meet row inside that public surface. The dyadic tolerance ledger is therefore read as part of μ, not as a separate completion or quotient object.

Definition 462.21 excludes quotient streams, selected limits, host real equality, ambient completeness, and bridge-checking oracle content. Hence a shared-bound consumer has no route outside the public export packet. □

Theorem 462.24 (CauchyModulusMeet diagonal handoff scope)**.** *A* $\mathsf{Real}^{\uparrow}$ *diagonal handoff consuming a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet may use only the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the two source stream/modulus rows, the shared meet row,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row. The handoff does not import quotient stream equality, a selected limit, ambient completeness, host real equality, or a bridge-checking oracle.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ diagonal handoff. Definition 462.7 forces the real-facing read through the stream-name rows, regular rational readbacks, shared dyadic tolerance ledger, transports, routes, provenance, and local naming row.

Definition 462.23 then factors the shared-bound part of the handoff through the public export surface. The source-pair exposure theorem identifies the exact source rows and shared meet row available to that surface.

It remains to rule out external analytic data. This is exactly Definition 462.21: the public packet supplies no quotient stream equality, selected limit, ambient completeness, host real equality, or bridge-checking oracle. □

Theorem 462.25 (CauchyModulusMeet common window coherence)**.** *Every* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *consumer that reads the shared meet row of a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet must use the same displayed packet*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$$

and the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows. In particular, a diagonal-window read and a regular-Cauchy tail read cannot choose separate source names, separate rational readbacks, or separate dyadic tolerance ledgers while claiming the same shared meet.*

Proof. Start with the carrier and the source-pair exposure theorem. Definition 462.1 gives the displayed coordinates, and Definition 462.22 identifies the two source stream/modulus rows together with the shared meet row.

Next read the shared bound through Definition 462.23. Any downstream shared-bound consumer is already forced through $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ and through no other route.

Finally apply Definition 462.24. A $\mathsf{Real}^{\uparrow}$ diagonal handoff has the same finite rows available as the public shared-bound read, so a diagonal-window consumer and a regular-Cauchy tail consumer meet on the same source, readback, tolerance, transport, route, provenance, and local naming packet. □

Theorem 462.26 (CauchyModulusMeet paired RegSeqRat readback)**.** *Both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks attached to* S_0 *and* S_1 *are exposed before any* $\mathsf{Real}^{\uparrow}$ *handoff using a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet. These readbacks are finite packet rows, not quotient stream equality, selected limits, or host real equality, and no downstream consumer may replace either readback by such an external equality.*

Proof. Use projection stability first. Definition 462.6 preserves the two source modulus projections under the componentwise transport rows, so the source-side readbacks remain attached to the displayed packet.

Now apply consumer factorization and source-pair exposure. Definition 462.10 routes every real-facing or regular-rational-facing read through the finite carrier coordinates, and Definition 462.22 names the paired source rows before the shared meet row is consumed.

The replacement exclusions are exactly the public export boundary. By Definition 462.21, no consumer route can extract quotient stream equality, selected limits, host real equality, ambient completeness, or bridge-checking oracle content from the packet. □

Theorem 462.27 (CauchyModulusMeet diagonal consumer sufficiency)**.** *Definition 462.25, Definition 462.26, and the public export surface of Definition 462.13 are sufficient for downstream diagonal-window consumers to use the shared bound of a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet. Such consumers need not import selected limits, ambient completeness, host real equality, quotient stream equality, or bridge-checking oracle content.*

Proof. First use common window coherence. It fixes the diagonal-window and regular-Cauchy tail consumers to the same $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ packet and the same finite stream-name, rational-readback, and dyadic-tolerance rows.

Next use paired readback. The two $\mathsf{RegSeqRat}^{\uparrow}$ rows are already exposed before the $\mathsf{Real}^{\uparrow}$ handoff, so the diagonal consumer can read the source rational data and the shared modulus data from the displayed packet.

Finally apply the public export surface and the export non-escape theorem. Definition 462.13 supplies exactly the finite rows needed by the diagonal consumer, while Definition 462.21 rules

out selected limits, ambient completeness, host real equality, quotient stream equality, and bridge-checking oracle content. □

Theorem 462.28 (CauchyModulusMeet standard bridge certificate)**.** *The public export packet of Definition 462.13 is exactly the standard shared-modulus bridge surface consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*. Every bridge read factors through*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The certificate exports no quotient stream, selected limit, host real equality, ambient completeness row, or bridge-checking oracle.

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetUp StdBridge`

Proof. First read the public export surface. By Definition 462.13, the available packet consists precisely of the two source names, two source modulus rows, the shared modulus row, transport rows, continuation routes, package provenance, and local name.

Next lock the downstream consumers to those rows. Apply Definition 462.14; every $\mathsf{RegSeqRat}^{\uparrow}$ read, $\mathsf{Real}^{\uparrow}$ handoff, source-modulus projection, and dyadic tolerance read must pass through the displayed shared-modulus packet.

The shared-bound content is exhausted by Definition 462.9. Thus the bridge surface has no further analytic coordinate beyond the shared meet row μ and its finite dyadic ledger.

Finally apply Definition 462.21. Since no consumer can extract quotient streams, selected limits, host real equality, ambient completeness, or an independent bridge checker, the public export packet is exactly the standard shared-modulus bridge surface available to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. □

Theorem 462.29 (CauchyModulusMeet mature shared-bound readback)**.** *Every mature shared-bound readback of a* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet is exactly the displayed surface*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The two source stream rows and their source modulus rows provide the only $\mathsf{RegSeqRat}^{\uparrow}$*-facing inputs,* μ *carries the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* H, C, P, N *supply only componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Proof. Begin with the carrier of Definition 462.1. It names precisely the two source streams, two source modulus rows, the shared meet row, transport ledger, routes, provenance, and local certificate.

The source-pair exposure and shared-bound exhaustion theorems identify the readable shared-bound data. Definition 462.22 fixes $S_0, S_1, \mu_0, \mu_1, \mu$, while Definition 462.9 binds the tolerance ledger to the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row carried by μ.

Finally use the public export surface and export consumer lock. They force every mature readback through $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ and prevent any additional source, tolerance, route, provenance, or naming coordinate from entering the shared-bound read. □

Theorem 462.30 (CauchyModulusMeet mature downstream handoff)**.** *Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *read and every downstream* $\mathsf{Real}^{\uparrow}$ *handoff consuming the mature* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *shared-bound package factors through the public export surface of Definition 462.13 and the standard bridge certificate of Definition 462.28. In particular, the downstream consumer receives only the shared-bound readback surface of Definition 462.29.*

Proof. First route the consumer through the public packet. Definition 462.14 already locks $\mathsf{RegSeqRat}^{\uparrow}$ reads, $\mathsf{Real}^{\uparrow}$ handoffs, source-modulus projections, and dyadic tolerance reads to the public export coordinates.

Next apply the diagonal handoff scope. Definition 462.24 and Definition 462.27 show that real-facing diagonal consumers use the same stream, regular-rational, shared-modulus, dyadic, transport, route, provenance, and naming rows as the public shared-bound read.

The standard bridge certificate then identifies that public packet as the bridge surface accepted by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Combining it with the mature shared-bound readback theorem gives the claimed factorization through the displayed package and through no separate downstream route. □

Theorem 462.31 (CauchyModulusMeet mature non-escape)**.** *The mature* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *downstream package exports no quotient stream, no selected limit, no host real equality, no ambient completeness principle, and no independent bridge checker. Its downstream readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *handoff remain confined to the public shared-bound export and the standard bridge certificate.*

Proof. Read the package through the mature shared-bound readback theorem. The only available surface is $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ with the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger.

Now apply the mature downstream handoff theorem. It factors the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer routes through the public export surface and the standard bridge certificate, so no consumer acquires a second source of analytic data.

Finally use export non-escape. Definition 462.21 already excludes quotient streams, selected limits, host real equality, ambient completeness, and bridge-checking oracle content from the public packet, and the standard bridge certificate repeats the same exclusion at the bridge surface. □

Theorem 462.32 (CauchyModulusMeet modulus-meet associativity)**.** *Let* M_{01}*,* M_{12}*, and* M_{02} *be accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packets whose source rows are respectively* (S_0, S_1)*,* (S_1, S_2)*, and the displayed shared meet row obtained by composing those two packets through the common* S_1 *source. Then every downstream consumer of the three packets reads the same finite shared-bound surface*

$$S_0, S_1, S_2, \mu_0, \mu_1, \mu_2, \mu_{01}, \mu_{12}, \mu_{02}, H, C, P, N$$

as the iterated meet packet. The comparison is only a public readback equality of exported rows; it does not assert quotient stream equality, host real equality, a selected limit, or ambient completeness.

Proof. Begin with the public export surface. Definition 462.13 exposes exactly the source stream rows, source modulus rows, shared meet row, transport rows, displayed routes, provenance, and local name for each packet.

Use source-pair exposure for M_{01} and M_{12}. The shared S_1 row and its modulus row are displayed before either meet row is consumed, so the two packets can be read through the common finite source-pair surface.

Now read the composed shared bound. Definition 462.29 fixes each mature readback to the displayed meet row and its $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and Definition 462.14 keeps downstream consumers on those rows.

Finally apply export non-escape. Since no packet exports quotient streams, selected limits, host real equality, ambient completeness, or a bridge-checking oracle, the two parenthesizations of the three accepted meet packets have the same downstream surface precisely because they display the same finite rows. □

Theorem 462.33 (CauchyModulusMeet meet-projection pair determinacy)**.** *Let M and M' be accepted* CauchyModulusMeet$^{\uparrow}$ *packets. If both packets expose the same two source* StreamName$^{\uparrow}$ *rows, the same two source* CauchyModulus$^{\uparrow}$ *rows, and the same shared meet row up to the displayed* hsame *transport, then their* RegSeqRat$^{\uparrow}$*-facing source-pair reads and their* DyadicRatCore$^{\uparrow}$ *tolerance ledgers have the same public readback surface.*

Proof. First apply projection stability. Definition 462.6 transports the two source modulus projections componentwise, so M and M' retain the same visible source-pair coordinates under the displayed hsame rows.

Next use source-pair exposure. Definition 462.22 identifies the RegSeqRat$^{\uparrow}$-facing reads with the two displayed source stream and modulus rows before the shared meet row is consumed.

The tolerance ledger is then determined by the mature shared-bound readback. Definition 462.29 binds the shared meet row to its DyadicRatCore$^{\uparrow}$ ledger and leaves no second ledger coordinate in the carrier.

Thus the public readback surfaces agree. The conclusion is only determinacy of the displayed source-pair and tolerance rows, not equality of quotient streams, selected limits, host real values, or ambient completeness witnesses. □

Theorem 462.34 (CauchyModulusMeet regular-limit consumer triad)**.** *Every* RealDiagonalWindowCoverage$^{\uparrow}$*,* RegularCauchyTriangleBound$^{\uparrow}$*, or* RegSeqRat$^{\uparrow}$ *consumer of an accepted* CauchyModulusMeet$^{\uparrow}$ *packet factors through the same rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The three consumer kinds may use those rows differently, but none obtains an additional quotient stream, selected limit, host real equality, ambient completeness principle, or bridge-checking oracle.

Proof. Start with common-window coherence. Definition 462.25 places diagonal-window and regular-tail consumers on the same stream, source-modulus, shared-modulus, dyadic-ledger, transport, route, provenance, and naming rows.

Next use paired RegSeqRat$^{\uparrow}$ readback. Definition 462.26 exposes the two source readbacks before the Real$^{\uparrow}$ handoff, so a regular-rational consumer reads the same source pair as a diagonal-window consumer.

Now apply diagonal consumer sufficiency. Definition 462.27 shows that the displayed packet already supplies the shared bound needed by diagonal-window and regular-Cauchy tail consumers.

Finally use mature downstream handoff. Definition 462.30 factors the downstream RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ routes through the public export surface and the standard bridge certificate, while the non-escape boundary rules out every undisplayed analytic coordinate. □

Theorem 462.35 (CauchyModulusMeet binary meet projection lock)**.** *Every accepted* CauchyModulusMeet$^{\uparrow}$ *packet exposes the two source modulus projections and the shared meet row exactly through*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

Here S_0, S_1 are the two source StreamName$^{\uparrow}$ *rows, μ_0, μ_1 are the source* CauchyModulus$^{\uparrow}$ *rows, μ is the shared meet row with its* DyadicRatCore$^{\uparrow}$ *ledger, and H, C, P, N are respectively the componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. The projection lock exports no selected limit, no quotient stream, no host real equality, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket binary projection lock`

Proof. Project the carrier of Definition 462.1. Its visible rows are the two source streams, the two source modulus rows, the shared meet row, the transport and continuation rows, package provenance, and the local certificate.

Apply source-pair exposure and projection stability. Definition 462.22 identifies the two source stream and modulus projections, while Definition 462.6 keeps those projections componentwise under the displayed hsame rows.

Use shared-bound exhaustion to identify the meet row. Definition 462.9 and Definition 462.29 leave the shared dyadic ledger inside μ and expose no second tolerance coordinate.

Finally apply the non-escape boundary. Definition 462.21 rules out selected limits, quotient streams, host real equality, ambient completeness, and any bridge-checking oracle outside the displayed packet. □

Theorem 462.36 (CauchyModulusMeet joint consumer exactness)**.** *Every joint* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet factors through the paired source rows* S_0, S_1, μ_0, μ_1*, the shared dyadic ledger carried by* μ*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*, package provenance* P*, and the local* NameCert *row* N*. No joint consumer may replace this route by a selected stream limit, quotient stream equality, host real equality, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket joint consumer exactness`

Proof. Begin with the export consumer lock. Definition 462.14 already forces every $\mathsf{RegSeqRat}^{\uparrow}$ read, $\mathsf{Real}^{\uparrow}$ handoff, source-modulus projection, and dyadic tolerance read through the public export coordinates.

Read the paired source rows next. Definition 462.26 exposes the two source readbacks before the real handoff, and Definition 462.35 fixes the corresponding modulus projections and shared meet row.

The real-facing side is the same finite route. Definition 462.24 and Definition 462.27 show that diagonal and real consumers use the same shared-bound packet and the same displayed transport, continuation, provenance, and naming rows.

Finally use mature downstream handoff and mature non-escape. They identify the downstream route with the public export surface and exclude selected limits, quotient streams, host real equality, ambient completeness, and any external analytic object. □

Theorem 462.37 (CauchyModulusMeet root-unblock export)**.** *The accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet exports the exact shared-bound surface consumed by* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The export consists of carrier admission, projection stability, swap stability, shared-bound exhaustion, diagonal handoff scope, common-window coherence, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and mature downstream handoff over the single packet*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

It is a root-unblock surface for those consumers only through displayed BEDC rows, not through quotient streams, selected limits, host real values, ambient completeness, or an external bridge checker.

Lean4 ✓ `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket root unblock export`

Proof. First assemble the packet surface. Carrier admission comes from Definition 462.1, and Definition 462.35 fixes the two source rows, the two source modulus rows, the shared meet row, transport, routes, provenance, and local name.

Next close the stability and shared-bound obligations. Definition 462.6 and Definition 462.8 keep the source projections stable, while Definition 462.9 identifies the shared dyadic ledger as the only bound exported by μ.

Now route the diagonal and common-window consumers. Definition 462.24, Definition 462.25, and Definition 462.27 put $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, and RealDiagonalWindowCoverage on the displayed shared-bound packet.

Finally route the rational and real consumers. Definition 462.26, Definition 462.36, and Definition 462.30 put $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ on the same packet, and Definition 462.31 excludes every undisplayed analytic route. □

Theorem 462.38 (CauchyModulusMeet uniform-family tail-threshold factorization)**.** *Let an accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet export its shared meet row to a finite-family* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *ledger and to an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. If both consumers read the shared tail threshold supplied by the meet packet, then both consumer routes factor through the same displayed rows*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The finite-family ledger and the tail-meet packet may transport the row through their own displayed hsame *and* Cont *routes, but neither consumer may replace μ by an independent tail selector, quotient stream, selected limit, host equality, ambient completeness witness, or any tail-threshold row not read from the displayed* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet.*

Proof. Begin with the exported meet surface. Definition 462.37 fixes the two source stream rows, the two source modulus rows, the shared meet row, transport, continuation, provenance, and local naming rows as the only root-unblock packet available to downstream consumers.

Next attach the regular tail-meet consumer. Definition 560.20 says that a finite-family threshold request absorbed by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ must pass through the displayed tail-meet route rather than a separate selector or completion object.

Now read the finite-family side. Definition 435.10 keeps the family threshold request on the shared transported threshold row, and Definition 435.47 keeps subfamily tail reads inside the same finite family ledger.

Combining these routes identifies the tail-meet threshold and the finite-family threshold as two consumer views of the same displayed row μ inside

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The exclusions follow because each cited route is a ledger-read theorem over displayed BEDC rows and none exposes quotient streams, selected limits, host equality, ambient completeness, or an external tail selector. □

Theorem 462.39 (CauchyModulusMeet finite-chain threshold induction)**.** *Let $M_0, \ldots, M_n$ be a finite chain of accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packets such that each adjacent pair shares the displayed source stream row and source modulus row needed to compose the next meet packet. Write the visible rows of the chain as*

$$S_0, \ldots, S_n, \mu_0, \ldots, \mu_n, \mu_{01}, \mu_{12}, \ldots, \mu_{n-1,n}, H, C, P, N$$

with the iterated meet row obtained by repeatedly applying the displayed shared-threshold meet. Then every $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of the chain factors through a single iterated shared-threshold surface assembled from the rows S_i, the*

modulus rows μ_i, the iterated meet row, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows of the accepted packets.*

The factorization is only a finite BEDC ledger read. It exports no quotient stream, no selected limit, no host real equality, no hidden tail oracle, and no ambient completeness principle.

Proof. The base case is the binary packet. Definition 462.35 fixes the two source rows, two source modulus rows, shared meet row, transports, continuations, provenance, and local name, while Definition 462.36 confines the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer routes to that same packet.

For the induction step, assume the initial segment already reads through its iterated shared-threshold surface. Compose that surface with the next accepted meet packet along the adjacent shared source row and shared modulus row. Definition 462.32 identifies the parenthesized readbacks as the same finite exported surface, so the next packet extends the row list without introducing a second threshold coordinate.

Now route the finite-family consumers. Definition 462.37 exposes the accepted meet packets as the root-unblock surface for $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and Definition 462.38 puts $\mathsf{UniformCauchyCriterion}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold reads on the same displayed meet row.

Finally apply mature non-escape at each accepted packet and at the iterated surface. Definition 462.31 rules out quotient streams, selected limits, host real equality, hidden analytic routes, bridge-checking oracles, and ambient completeness, so the finite chain supplies only the displayed shared-threshold ledger requested by the downstream consumers. □

Theorem 462.40 (CauchyModulusMeet regular-comparison window handoff)**.** *An accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *packet exporting*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$$

supplies the shared window and dyadic tolerance surface consumed by a $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet. The comparison rows W, D, τ must therefore be read from the displayed meet row μ and its source projections S_0, S_1, μ_0, μ_1, rather than from a second tolerance surface, quotient stream equality, host real equality, selected limit, or ambient completeness principle.*

Proof. First lock the meet projections. Apply Definition 462.35: the two source rows, the two source modulus rows, and the shared meet row are exactly the displayed $S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N$ surface.

Next read joint consumer exactness and the root-unblock export. Definition 462.36 keeps $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers on that same finite route, and Definition 462.37 identifies it as the shared-bound surface exposed to regular comparison consumers.

Now attach the $\mathsf{RegularCauchyComparison}^{\uparrow}$ packet. Its carrier and window exactness rows, Definition 411.1 and Definition 411.4, force the two regular names through one displayed window W before dyadic observations are read.

Finally apply tolerance-row exactness and the real seal boundary, Definition 411.7 and Definition 411.8. They make the comparison's D, τ rows part of the same finite comparison surface and exclude a second tolerance, host real equality, quotient stream, selected limit, or ambient completeness route. □

Theorem 462.41 (CauchyModulusMeet Real seal shared-threshold determinacy)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer that uses the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *window route and the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *shared meet row must read the same threshold packet*

$$S_0, S_1, \mu_0, \mu_1, \mu, H, C, P, N.$$

The consumer may pass through the comparison window rows W, D, τ, but those rows are determined by the displayed shared meet packet. It cannot introduce a second tolerance surface, quotient stream equality, selected limit, host real equality, or ambient completeness witness.

Proof. First lock the shared packet. Definition 462.35 fixes the two source rows, the two source modulus rows, the shared meet row, and the transport, continuation, provenance, and naming rows.

Next read the consumer side. Definition 462.36 keeps $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers on that same finite route, while Definition 462.37 exposes it as the shared-bound packet consumed downstream.

Now attach the comparison route. Definition 462.40 forces the $\mathsf{RegularCauchyComparison}^{\uparrow}$ window and dyadic tolerance rows to be read from the displayed meet row μ and its source projections.

The exclusions follow from the same route determinations. Since the cited packet, export, and handoff theorems expose no second tolerance surface, quotient stream equality, selected limit, host real equality, or ambient completeness witness, the Real-seal and regular-rational consumers read one deterministic threshold packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusMeet}^{\uparrow}$ packet is generated from two $\mathsf{StreamName}^{\uparrow}$ rows, their $\mathsf{RegSeqRat}^{\uparrow}$ rational readbacks, two source $\mathsf{CauchyModulus}^{\uparrow}$ rows, one shared $\mathsf{CauchyModulus}^{\uparrow}$ meet row with a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 462.1, Definition 462.2, Definition 462.6, Definition 462.7, Definition 462.8, Definition 462.9, Definition 462.10, Definition 462.11, Definition 462.12, Definition 462.13, Definition 462.14, Definition 462.21, Definition 462.25, Definition 462.26, Definition 462.27, Definition 462.28, Definition 462.29, Definition 462.30, and Definition 462.31 bind the finite shared-modulus carrier, source modulus projections, dyadic tolerance ledger, Real and RegSeqRat consumer boundaries, diagonal-window consumer surface, public export certificate, standard bridge surface, mature shared-bound readback, mature downstream handoff, and mature non-escape package over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusMeet}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CauchyModulusMeetUp.CauchyModulusMeetPacket\_root\_unblock\_export`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, ambient completeness, host real equality, or bridge-checking oracle content.

Upgrade path: No further formal-axis upgrade remains for the checked standard bridge.

Chapter 463

A Concrete Naming Certificate for $\mathsf{UniformLimit}^{\uparrow}$

The $\mathsf{UniformLimit}^{\uparrow}$ interface names the constructive-analysis packet for a uniformly convergent family of displayed real-valued windows. Following Bishop–Bridges constructive analysis, the limiting object is read through a uniform tail ledger, a regular rational handoff, and a visible $\mathsf{CauchyLimitSeal}^{\uparrow}$ row. The packet does not select a host function limit, quotient a function space, or appeal to ambient completeness of $\mathsf{Real}^{\uparrow}$.

The endpoint row is available only after the packet consumes the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ tail ledger of chapter 435, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal row of chapter 386. Thus the endpoint is a routed finite seal read, not a host-selected limit.

Definition 463.1 (UniformLimit carrier)**.** *A* $\mathsf{UniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

with the following displayed coordinates:

1. *F is the displayed function/window family, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{Real}^{\uparrow}$ *endpoint rows;*
2. *M is the shared modulus ledger assigning each requested tolerance to one family-wide window threshold;*
3. *T is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail ledger that records the accepted comparisons beyond the threshold selected by M;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the same windows and tail ledger;*
5. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal row attached to the regular handoff;*
6. *E is the exposed* $\mathsf{Real}^{\uparrow}$ *endpoint row read after the seal row;*
7. *H stores componentwise* hsame *transports for the function/window family, modulus ledger, uniform tail ledger, regular handoff, limit seal, endpoint row, package provenance, and local naming row;*
8. *C stores the displayed* Cont *routes for reading the windows, applying the shared threshold, consuming the uniform tail ledger, reaching the regular handoff, and exposing the seal endpoint;*

9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows, their displayed transports, their continuation routes, package provenance, and local naming provenance.

Theorem 463.2 (UniformLimit NameCert obligations)**.** *For the carrier of Definition 463.1, the local* $\mathsf{NameCert}_{\mathsf{UniformLimit}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet $U = (F, M, T, R, S, E, H, C, P, N)$;*
2. *classifier transport by componentwise* hsame *over the function/window family, modulus ledger, tail ledger, regular handoff, seal row, and endpoint row;*
3. *uniform-tail exactness, requiring each endpoint-facing read to pass through the same threshold row M and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *ledger T;*
4. *regular-seal routing, requiring every limit-facing read to pass from F, M, T through R and S before reaching E;*
5. *non-escape from pointwise convergence without a uniform ledger, open-cover compactness, selected host limits, quotient function spaces, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformLimitUp.UniformLimitCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 463.1. The displayed coordinates name the family windows, shared modulus, uniform tail ledger, regular handoff, seal row, endpoint row, transports, continuations, package provenance, and local certificate.

Classifier transport is componentwise. The rows in H move each named coordinate, and the routes in C replay the same consumer order, so a transported packet remains inside the displayed carrier surface.

For exactness, inspect a requested tolerance. The consumer route first reads M, then reads the corresponding comparisons in T, and only then reaches the regular handoff and seal rows. The packet inventory contains no coordinate for pointwise-only convergence, open-cover compactness, selected host limits, quotient function spaces, or real completeness. □

Theorem 463.3 (UniformLimit seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *endpoint read exported by an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier factors through the finite route*

$$F, M, T \longmapsto R \longmapsto S \longmapsto E.$$

Thus the endpoint is exposed only after the displayed function/window family has been read through the shared threshold ledger, the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, together with the visible* hsame, Cont, Pkg, *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.UniformLimitUp.UniformLimitCarrier seal factorization`

Proof. Begin with an endpoint-facing consumer request. By the carrier definition, the only $\mathsf{Real}^{\uparrow}$ endpoint coordinate is E, and the continuation route to E passes through the named rows of the same finite packet.

Read the family windows through F and the shared threshold through M. The uniform comparisons are then supplied by T, so the request has not used an indexwise or pointwise-only convergence witness.

Next consume the regular rational handoff R and the limit seal S. These rows mediate the passage from the uniform tail ledger to the endpoint row, while H, C, P, N record the transport, route, package, and naming provenance. Hence every endpoint read has the stated factorization. □

Theorem 463.4 (UniformLimit non-escape boundary)**.** *An accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier exports only the displayed function/window family, shared modulus ledger,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal row,* $\mathsf{Real}^{\uparrow}$ *endpoint, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 463.1. It exports no pointwise convergence object lacking the uniform ledger, no open-cover compactness principle, no selected host limit, no quotient function space, and no completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Inspect the packet coordinates. Every listed row is a finite display, transport, continuation, provenance, or local naming row, and the only endpoint coordinate is the seal-facing row E.

Apply Definition 463.2. Uniform-tail exactness keeps each read inside F, M, T, and regular-seal routing forces the route through R and S before E is exposed.

The excluded analytic objects would require additional carrier coordinates or a different consumer route. Since neither is present, the public surface is exhausted by the finite BEDC rows listed in the statement. □

Theorem 463.5 (UniformLimit finite endpoint certificate)**.** *The carrier, local* NameCert *obligations, seal factorization, and non-escape boundary assemble a finite endpoint certificate for* $\mathsf{UniformLimit}^{\uparrow}$ *over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Every downstream completion or real-consumer read is confined to that packet and its factorization route.*

Proof. Start with the carrier of Definition 463.1. It supplies exactly the function/window family, shared modulus, uniform tail ledger, regular handoff, seal row, endpoint row, transports, continuations, package provenance, and local certificate.

Use Definition 463.2 to obtain carrier admission, classifier transport, uniform-tail exactness, regular-seal routing, and the non-escape clauses. Then use Definition 463.3 to identify the endpoint route.

Finally apply Definition 463.4. The route has no coordinate outside the listed BEDC rows, so the finite endpoint certificate is the whole downstream consumer surface. □

Theorem 463.6 (UniformLimit windowwise Cauchy obligation)**.** *Every endpoint-facing read of an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier must pass through the shared threshold ledger* M *and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail row* T *before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. Thus the read is windowwise uniform: no endpoint row may be obtained from a pointwise-only convergence witness or from a window threshold selected outside the packet.*

Proof. Project the route inventory from Definition 463.1. The only endpoint-facing continuation route reads the family windows through F, then the shared threshold ledger M, then the uniform tail ledger T.

Apply the uniform-tail and regular-seal clauses of Definition 463.2. They force the consumer route to use the same M and T before the handoff row R can be reached.

Finally use Definition 463.3 and Definition 463.5. Both identify the downstream endpoint path as the finite route through F, M, T, R, S, E, so a pointwise-only or externally selected threshold route is not part of the certificate. □

Theorem 463.7 (UniformLimit limit-seal compatibility)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* S *of an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier are compatible with the same window coordinates and the same* hsame *and* Cont *rows used by the uniform tail route. Transporting the window family, shared threshold, or uniform tail ledger therefore transports the handoff and seal route without changing the endpoint surface.*

Proof. Read the carrier coordinates H and C from Definition 463.1. These rows store the componentwise transports and displayed continuations for the function/window family, threshold ledger, uniform tail ledger, handoff row, seal row, and endpoint.

Use Definition 463.3. It factors every endpoint read through the same route from F, M, T to R, then to S, and finally to E.

Apply the classifier-transport clause of Definition 463.2. Since R and S are transported by the same H and replayed by the same C as the tail route, the transported packet keeps the handoff and seal compatible with the original windows and endpoint boundary. □

Theorem 463.8 (UniformLimit ledger exactness obligation)**.** *For an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier, the packet*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

exhausts the visible ledger. It includes the function/window family, shared modulus, uniform tail ledger, regular handoff, limit seal, endpoint row, transports, continuations, package provenance, and local naming row, and it excludes pointwise-only convergence, open-cover compactness, selected host limits, quotient function spaces, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 463.1. Its tuple lists every visible coordinate of the carrier and names the dependency rows in P and N.

Use Definition 463.6 for the tail side and Definition 463.7 for the handoff and seal side. Together they show that all endpoint-facing routes are contained in the displayed packet.

Finally apply Definition 463.4 and Definition 463.5. The excluded analytic structures would require additional carrier coordinates or a different consumer route, and neither is present in U. □

Theorem 463.9 (UniformLimit tail-ledger exactness)**.** *For every accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier*

$$U = (F, M, T, R, S, E, H, C, P, N),$$

each endpoint-facing read first factors through the displayed function/window family F*, the shared threshold ledger* M*, and the unique* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail ledger* T *visible to the packet. No second tail ledger, pointwise-only convergence witness, or independently selected threshold route can be read before the regular handoff row* R*.*

Lean4 ✓ `BEDC.Derived.UniformLimitUp.UniformLimitCarrier tail ledger exactness`

Proof. Project U through Definition 463.1. The only rows before the regular handoff are F, M, T.

Apply Definition 463.2. The local NameCert surface names the function/window family, the shared threshold ledger, and the uniform Cauchy tail ledger as the finite endpoint-facing entrance to the packet.

Use Definition 463.6. Any endpoint-facing read reaches R only after the shared threshold and uniform tail rows have been read.

Finally apply Definition 463.8. Since the visible ledger has exactly one T coordinate and no pointwise-only convergence or second threshold coordinate, the displayed T is the unique tail ledger available to the packet. □

Theorem 463.10 (UniformLimit tail-seal route uniqueness)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

and

$$U' = (F', M', T', R', S', E', H', C', P', N')$$

be accepted $\mathsf{UniformLimit}^{\uparrow}$ *carriers. If the displayed classifier rows identify* F *with* F'*,* M *with* M'*,* T *with* T'*,* R *with* R'*,* S *with* S'*, and* E *with* E'*, then every endpoint-facing read follows the same finite route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E.$$

No pointwise-only convergence witness, second tail ledger, selected host limit, quotient function space, or $\mathsf{Real}^{\uparrow}$ *completeness theorem can provide a competing endpoint route.*

Proof. Project both carriers through Definition 463.1. The classifier hypotheses identify the function/window family, shared threshold ledger, uniform tail ledger, regular handoff, seal row, and endpoint row on the displayed carrier surface.

Apply Definition 463.9. Any endpoint-facing read must enter through F, M, T, and the accepted carrier has no second tail ledger or pointwise-only entrance that could reach R first.

Use Definition 463.3. Once the tail entrance is fixed, the endpoint route factors through the regular handoff R, the seal row S, and the endpoint E, so the displayed route is the only tail-to-seal path available to the consumer.

Finally invoke Definition 463.4. The excluded analytic objects are not carrier coordinates and are not exported by the transport, continuation, provenance, or local naming rows, so they cannot supply a competing endpoint read. □

Theorem 463.11 (UniformLimit seal-route transport)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

and

$$U' = (F', M', T', R', S', E', H', C', P', N')$$

be accepted $\mathsf{UniformLimit}^{\uparrow}$ *carriers related by the displayed classifier. Componentwise* hsame *transport of* F, M, T *transports the regular handoff, limit seal, and endpoint route* R, S, E *through the same* H, C, P, N *provenance. Thus the seal route remains the same finite route after transport:*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E.$$

Proof. Start with the compatibility theorem Definition 463.7. It states that the handoff row R and seal row S use the same window coordinates and the same transport and continuation rows as the uniform tail route.

Apply Definition 463.9. Transport of the endpoint-facing entrance is therefore transport of the rows F, M, T, not a change to a different tail ledger.

Use Definition 463.3. The endpoint row E is reached only through R and S, so transporting F, M, T carries the whole route through the same R, S, E boundary.

Finally invoke Definition 463.2 and Definition 463.8. The rows H, C, P, N are the displayed transport, replay, provenance, and local naming rows for this packet, and no additional seal route is present. □

Theorem 463.12 (UniformLimit endpoint readback exhaustion)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier can read only*

$$F, M, T, R, S, E, H, C, P, N.$$

Such a consumer may read the displayed window family, shared threshold ledger, uniform tail ledger, regular handoff, limit seal, endpoint row, transports, continuations, package provenance, and local naming row. It cannot extract pointwise convergence without the uniform ledger, quotient function spaces, selected host limits, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an arbitrary downstream endpoint read. By Definition 463.9, the read first factors through F, M, T.

Apply Definition 463.11 and Definition 463.3. The regular handoff, limit seal, and endpoint route are exactly R, S, E, transported and replayed by the same H, C rows.

Use Definition 463.5. The finite certificate packages the endpoint-facing route together with P and N as the visible provenance and local naming surface.

Finally apply Definition 463.4 and Definition 463.8. The excluded pointwise, quotient, selected-limit, compactness, and completeness objects are not coordinates of the carrier and cannot be recovered by readback. □

Theorem 463.13 (UniformLimit obligation closure package)**.** *For every accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier*

$$U = (F, M, T, R, S, E, H, C, P, N),$$

the carrier, local NameCert *obligations, windowwise Cauchy obligation, limit-seal compatibility, tail-ledger exactness, seal-route transport, and endpoint readback exhaustion jointly supply the obligation surface over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package exports exactly the displayed function/window family, shared threshold ledger, uniform tail ledger, regular handoff, limit seal, endpoint row, transport routes, continuation routes, provenance, and local naming data. It exports no pointwise-only convergence object, open-cover compactness principle, selected host limit, quotient function space, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the carrier and local naming obligations in Definition 463.1 and Definition 463.2. They identify the coordinates $F, M, T, R, S, E, H, C, P, N$ and bind carrier admission, classifier transport, uniform-tail exactness, regular-seal routing, and non-escape to those coordinates.

Use the windowwise Cauchy obligation of Definition 463.6 and the limit-seal compatibility theorem Definition 463.7. These force every endpoint route to read the shared threshold and the same uniform tail ledger before reaching the handoff and seal rows.

Apply tail-ledger exactness from Definition 463.9 and seal-route transport from Definition 463.11. They show that transported endpoint reads remain on the finite route from F, M, T through R, S, E.

Finally use endpoint readback exhaustion in Definition 463.12. Since all downstream consumers can read only the displayed rows and their provenance, the obligation package has no coordinate for pointwise-only convergence, open-cover compactness, selected host limits, quotient function spaces, or real completeness. □

Theorem 463.14 (UniformLimit diagonal-window handoff)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier, and let*

$$G = (q, w, r, d, e, h, c, p, n)$$

be an accepted $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *carrier. Suppose that the selected window* w *is a displayed window of* F*, and that the readback row* r *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read through* T*. Then every* $\mathsf{Real}^{\uparrow}$*-facing diagonal-window read determined by* G *and the displayed window of* U *factors simultaneously through the two finite routes*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E$$

and

$$q \longmapsto w \longmapsto r \longmapsto d \longmapsto e.$$

Consequently the diagonal-window consumer can use the uniform-limit endpoint exposure only through the selected finite window, the uniform tail ledger, the regular rational handoff, the dyadic observation ledger, and the displayed $\mathsf{Real}^{\uparrow}$ *seal; it gains no pointwise-only convergence row, host limit, quotient function space, ambient completeness theorem, or unbounded diagonal stream.*

Proof. Start from the accepted $\mathsf{UniformLimit}^{\uparrow}$ carrier. By Definition 463.3, every endpoint read exported by U reaches the endpoint only along the finite route through F, M, T, R, S, E.

Use the windowwise Cauchy obligation of Definition 463.6 and tail-ledger exactness from Definition 463.9. Since w is one of the displayed windows of F and r is the $\mathsf{RegSeqRat}^{\uparrow}$ handoff read through T, the diagonal read cannot bypass the shared threshold ledger or replace the uniform tail row with a pointwise witness.

Now turn to the accepted $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ carrier. The selected-window coverage theorem Definition 561.7 forces the seal-facing read to expose w and the dyadic ledger d before the seal row is consumed.

Apply the RealUp handoff theorem Definition 561.8. It routes every real-facing completion or continuous-map consumer through q, w, r, d, h, c, p, n before the $\mathsf{Real}^{\uparrow}$ seal row e is read, so the selected-window side is exactly the displayed route from q through e.

Finally combine the non-escape boundary for $\mathsf{UniformLimit}^{\uparrow}$ with the non-escape boundary for $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$. The first boundary excludes host limits, quotient function spaces, pointwise-only convergence, compactness principles, and real completeness from U; the second excludes quotient streams, host real equality, unbounded diagonal streams, ambient completeness, and choice-selected limits from G. Thus the only common consumer route is the paired finite factorization through the uniform-limit endpoint route and the selected diagonal-window route. □

Theorem 463.15 (UniformLimit observation-budget factorization)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier, and let*

$$B = (e, W, D, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier. Suppose the finite windows in* W *are displayed windows of* F*, the dyadic readbacks in* D *are the endpoint-facing readbacks for those windows, the handoff rows in* R_B *include the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *read through* T*, and the seal rows in*

S_B include the same endpoint-facing seal route S, E. Then every $\mathsf{Real}^{\uparrow}$ *endpoint read exported by U factors through the observation-budget route*

$$e \longmapsto W \longmapsto D \longmapsto R_B \longmapsto S_B$$

before exposing the endpoint row E.

Thus a common consumer of the uniform-limit packet and the observation-budget packet reads the shared finite windows, dyadic readbacks, regular handoff, seal rows, transports, and routes before the endpoint is available.

Proof. First identify the uniform-limit entrance. By Definition 463.3, every endpoint read of U reaches E only through F, M, T, R, S.

Next apply the windowwise Cauchy obligation and the diagonal-window handoff, Definition 463.6 and Definition 463.14. Since the windows in W are displayed windows of F and the readbacks in D are tied to those windows, the endpoint-facing read cannot bypass the shared threshold row M or the tail ledger T.

Now read the budget side. The finite-window seal-route totality theorem for $\mathsf{RealObservationBudget}^{\uparrow}$, Definition 611.17, routes every admitted seal read through e, W, D, R_B, S_B with the displayed transports, continuations, provenance, and local name.

Finally combine the two route inventories. The shared handoff hypothesis identifies the uniform-limit R with the budget handoff rows, and the shared seal hypothesis identifies S, E with the budget seal route. Therefore the common endpoint read must pass through the observation-budget route before E is exposed. □

Theorem 463.16 (UniformLimit observation-budget nonchoice boundary)**.** *Under the hypotheses of Definition 463.15, the common endpoint read cannot be replaced by pointwise convergence, a hidden selected limit, quotient function-space equality, an ambient completeness theorem, quotient stream equality, host real equality, or an unbounded observation schedule. Its only admissible route is the shared finite budget route through F, M, T, R, S, E and e, W, D, R_B, S_B with the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with the uniform-limit non-escape boundary. Definition 463.4 removes pointwise-only convergence, open-cover compactness, selected host limits, quotient function spaces, and $\mathsf{Real}^{\uparrow}$ completeness from the $\mathsf{UniformLimit}^{\uparrow}$ packet.

Then apply the observation-budget non-escape theorem, Definition 611.6. It removes quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded observation schedules from the $\mathsf{RealObservationBudget}^{\uparrow}$ packet.

The factorization theorem Definition 463.15 gives the remaining common route. Since both packets expose only their displayed rows and the shared handoff and seal hypotheses identify the overlap, a consumer has no second source from which to select the endpoint. □

Theorem 463.17 (UniformLimit shared threshold regular seal)**.** *For every accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier*

$$U = (F, M, T, R, S, E, H, C, P, N),$$

the endpoint-facing route uses one shared threshold ledger M and one $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail row T before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row S. Thus the regular handoff and seal are bound to the same windowwise threshold data that generated the uniform tail comparisons.*

Proof. Start with the accepted carrier U. By Definition 463.6, every endpoint-facing read must enter through the displayed windows, the shared threshold ledger M, and the uniform tail row T before it can reach R.

Apply Definition 463.9. The carrier has no second tail ledger and no independently selected threshold route, so the rows M and T used at the entrance are the unique rows available to the regular handoff.

Finally use Definition 463.3. The route from the regular handoff to the endpoint passes through S and then E, so the seal row is attached to the same shared threshold and tail ledger rather than to a separate pointwise or host-level limit route. □

Theorem 463.18 (UniformLimit RegSeq seal classifier scope)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

and

$$U' = (F', M', T', R', S', E', H', C', P', N')$$

be accepted $\mathsf{UniformLimit}^{\uparrow}$ *carriers related by the displayed classifier. Classifier transport of* F, M, T *transports the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* S*, and the endpoint row* E *through the displayed* hsame *rows and* Cont *routes already recorded in* H *and* C*. It does not create a second seal route.*

Proof. Use Definition 463.7. The handoff row R and seal row S are compatible with the same window coordinates, transport rows, and continuation routes that carry the uniform tail route.

Apply Definition 463.11. Once F, M, T are transported by the displayed classifier, the regular handoff, seal, and endpoint route are transported along the same finite path through R, S, E.

The carrier inventory contains only the transport ledger H and continuation ledger C for this route. Hence classifier transport preserves the displayed seal route and supplies no additional route by which a consumer could bypass the regular handoff. □

Theorem 463.19 (UniformLimit completion consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformLimit}^{\uparrow}$ *carrier factors through*

$$F, M, T, R, S, E, H, C, P, N.$$

Such a consumer cannot read pointwise-only convergence, selected host limits, quotient function spaces, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Begin with an arbitrary real-facing or completion-facing consumer. By Definition 463.12, the consumer can read only the displayed carrier rows $F, M, T, R, S, E, H, C, P, N$.

Use Definition 463.14 for the boundary handoff. Any diagonal-window read that reaches the endpoint must pass through the finite uniform route and the selected finite window route, so it gains no unbounded stream, host limit, quotient function space, or ambient completeness principle.

Finally apply Definition 463.4. The excluded analytic objects are not carrier coordinates and are not exposed by the continuation or provenance rows. Therefore the Real and completion consumers are confined to the displayed uniform-limit packet. □

Theorem 463.20 (UniformLimit continuous map handoff)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier. Any* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer, and more generally any continuity-preservation consumer of the limiting endpoint, must factor through the finite route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E$$

together with the transport, continuation, provenance, and non-escape rows H, C, P, N*. The shared threshold ledger* M *and uniform tail row* T *supply the only uniform threshold surface before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* S*. The consumer therefore has no access to pointwise-only convergence, host function limits, quotient function spaces, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{ContinuousMap}^{\uparrow}$ or continuity-preservation consumer of the limiting endpoint. By Definition 463.19, every endpoint-facing consumer is confined to the displayed carrier rows $F, M, T, R, S, E, H, C, P, N$ and cannot read an ambient function-space or completeness object.

Use Definition 463.6. The route into the endpoint is forced to pass through the finite windows, the shared threshold ledger M, and the uniform tail row T. Thus the continuity consumer receives its limiting-endpoint data only after the same uniform threshold surface has been selected.

Finally apply Definition 463.3. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row S are downstream of that shared uniform route, so the handoff presented to a continuous-map consumer is exactly the finite completion-facing surface already sealed by the packet. The listed pointwise, host-limit, quotient, compactness, and $\mathsf{Real}^{\uparrow}$-completeness readings are outside the carrier inventory and cannot be consumed here. □

Theorem 463.21 (UniformLimit finite window seal exhaustion)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier, and let*

$$G = (q, w, r, d, e, h, c, p, n)$$

be an accepted $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *carrier. Suppose* w *is a displayed window of* F*, and suppose* r *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read through* T*. Then every common* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer read of the two packets is exhausted by the paired finite route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E$$

together with

$$q \longmapsto w \longmapsto r \longmapsto d \longmapsto e.$$

Conversely, any read assembled from those two displayed routes, together with the visible hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of* U *and* G*, is an admissible finite seal read for the common endpoint-facing consumer surface. No such read can escape to pointwise-only convergence, quotient function spaces, selected host limits, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or unbounded diagonal streams.*

Proof. First identify the uniform-limit side of the common route. Apply Definition 463.14: because w is a displayed window of F and r is read through T, every diagonal-window endpoint read supplied by U factors through the finite route from F to E. Apply Definition 463.17 to bind that route to the same shared threshold row M, the same tail ledger T, the same regular handoff R, and the same seal row S.

Next transport the route through the displayed classifier surface. By Definition 463.18, classifier transport of F, M, T carries R, S, E along the same hsame and Cont rows already present in H and C. Thus the common consumer cannot obtain a second regular-seal route from a pointwise-only witness, a quotient function space, a selected host limit, or an open-cover compactness principle.

Now identify the selected-window side. The theorem Definition 561.7 forces any $\mathsf{Real}^{\uparrow}$-facing read of G to display w and the dyadic observation ledger d before the seal row e is consumed. The handoff theorem Definition 561.8 then routes every completion or continuous-map consumer through q, w, r, d, e and the visible transport, continuation, provenance, and naming rows.

Combining these two finite routes gives the forward exhaustion claim. Apply Definition 463.19 to the $\mathsf{UniformLimit}^{\uparrow}$ packet and the RealDiagonalWindowCoverage handoff just cited to remove every undisplayed common consumer source, including ambient $\mathsf{Real}^{\uparrow}$ completeness and unbounded diagonal streams.

For the converse direction, start with a read assembled from the displayed rows of the two routes. The uniform route is admissible because the accepted carrier U exposes exactly $F, M, T, R, S, E, H, C, P, N$ to endpoint-facing consumers, and the selected-window route is admissible because the accepted carrier G exposes exactly $q, w, r, d, e, h, c, p, n$ to Real-facing consumers. Since the two routes agree at the displayed window w and the handoff row r, their assembly is a finite seal read on the common endpoint-facing surface and uses no row outside the two accepted packets. □

Theorem 463.22 (UniformLimit finite-window induction)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier. Induction over the displayed finite windows of F, with the base row read through the shared threshold ledger M and successor rows read through the uniform tail ledger T, generates the finite endpoint route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E.$$

Every finite stage of this induction is an admissible carrier read using only the visible hsame*,* Cont*,* Pkg*, and* NameCert *rows of U.*

Proof. The base stage is the displayed threshold row. By Definition 463.6, each finite window of F must be read through the shared threshold ledger M before any endpoint-facing consumer can see a tail estimate.

For the successor stage, apply Definition 463.9. The uniform tail row T supplies the next finite witness without changing the selected threshold surface, so the regular handoff R and the seal row S are appended by the same carrier route rather than by a pointwise side channel.

Iterating these finite stages yields the displayed route from F through E. The exhaustion theorem Definition 463.21 then shows that no additional endpoint read is needed: every common completion-facing read is already accounted for by the finite route and the visible transport, continuation, provenance, and naming rows of U. □

Theorem 463.23 (UniformLimit Real seal nonescape)**.** *Let*

$$U = (F, M, T, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *carrier. The finite-window induction of Definition 463.22 exports only the finite route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto S \longmapsto E$$

and the visible hsame, Cont, Pkg, *and* NameCert *rows of* U. *It does not export pointwise-only convergence, a selected host limit, a quotient function space, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 463.22. The induction surface is finite at every stage and reaches the endpoint only through M, T, R, S, E and the visible carrier rows.

Now use Definition 463.4. That boundary rules out every read of U that bypasses the shared threshold ledger or treats the seal row as evidence for a pointwise, quotient, compactness, host-limit, or ambient-completeness object.

Finally apply Definition 463.19. Completion-facing consumers are confined to the same finite endpoint surface, so the induction cannot export any $\mathsf{Real}^{\uparrow}$ completeness principle or external function-space limit beyond the rows displayed by the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformLimit}^{\uparrow}$ packet is generated from a displayed function/window family, a shared threshold ledger, one $\mathsf{UniformCauchyCriterion}^{\uparrow}$ tail ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal row, a $\mathsf{Real}^{\uparrow}$ endpoint, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 463.1, Definition 463.3, Definition 463.5, Definition 463.17, and Definition 463.4 bind the finite carrier, seal factorization, public-facing endpoint certificate, tail-ledger exactness, and non-escape boundary for the displayed uniform-limit packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformLimit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.UniformLimitUp.UniformLimitCarrier\_seal\_factorization`

Bridge status: none

Origin: human

Not claimed: This chapter does not close pointwise convergence without a uniform ledger, open-cover compactness, selected host limits, quotient function spaces, $\mathsf{Real}^{\uparrow}$ completeness, or a checked Lean certificate.

Upgrade path: Formal audit and bridge checking remain verification-axis work over the same finite uniform-limit endpoint surface.

The $\mathsf{CauchyProduct}^{\uparrow}$ hub binds the product-facing Real route to the sibling rows of chapter 126, chapter 291, chapter 301, and chapter 38.

Chapter 464

A Concrete Naming Certificate for $\mathsf{CauchyProduct}^{\uparrow}$

The $\mathsf{CauchyProduct}^{\uparrow}$ interface names the finite product certificate for two regular Cauchy names. It expresses the Bishop–Bridges constructive product of regular Cauchy sequences as displayed $\mathsf{NameCert}$ rows over scheduled finite windows, dyadic observations, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal, without adding quotient streams or an ambient completeness operation.

Definition 464.1 (CauchyProduct NameCert source)**.** *A* $\mathsf{CauchyProduct}^{\uparrow}$ *source is a finite* BHist *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N)$$

with the following displayed coordinates:

1. *X and Y are the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows;*
2. *W_X and W_Y are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules for a shared precision request;*
3. *M_X and M_Y are the paired dyadic-radius rows bounding the two source windows;*
4. *Q_X and Q_Y are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows read from the scheduled windows;*
5. *P is the pointwise product observation row formed from Q_X and Q_Y inside the displayed dyadic rational core;*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row for the product observation and its finite radius ledger;*
7. *H stores the displayed* hsame *transports for the two source names, both stream windows, both radius rows, both dyadic observations, the product observation, and the regular rational readback;*
8. *C stores the displayed* Cont *routes for reading the source windows, transporting the dyadic product observation, exposing the regular rational classifier, and handing the result to the real seal;*
9. *L is the finite ledger recording the paired input radii, the product-radius row, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the* Pkg *provenance over* $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row.*

The packet stores a finite product construction. It contains no quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductTasteGate single carrier alignment`

Theorem 464.2 (CauchyProduct NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^\uparrow}$ *surface for the source packet of Definition 464.1 consists exactly of carrier admission, paired* $\mathsf{RegularCauchyName}^\uparrow$ *source admission, paired* $\mathsf{StreamName}^\uparrow$ *window scheduling, dyadic-radius routing,* $\mathsf{DyadicRatCore}^\uparrow$ *product observation, pointwise* $\mathsf{RegSeqRat}^\uparrow$ *classifier exposure,* $\mathsf{Real}^\uparrow$ *seal handoff, displayed* hsame *and* Cont *transport, package provenance, finite ledger, and the local naming row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket namecert obligations`

Proof. Carrier admission is projection from the displayed packet Z. Its coordinates list the two regular Cauchy sources, their paired stream windows, the radius rows, dyadic observations, product observation, readback classifier, transport rows, continuation routes, ledger, and local certificate.

The analytic product step is finite. The two window reads expose Q_X and Q_Y inside $\mathsf{DyadicRatCore}^\uparrow$, and the product observation row P is the only row that combines them. The classifier row E then reads this product observation with the radius information stored in the ledger.

The remaining obligations are provenance obligations. The rows H, C, L, N carry transport, continuation, package, seal handoff, and local naming data, so the certificate has no source from which to read host multiplication, quotient equality, selected limits, trichotomy, or completeness. □

Theorem 464.3 (CauchyProduct root window obligations)**.** *For an accepted* $\mathsf{CauchyProduct}^\uparrow$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the root-facing paired-window obligation is anchored in the BHist *rows* $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y$*. The two* $\mathsf{RegularCauchyName}^\uparrow$ *sources may be consumed only through the paired* $\mathsf{StreamName}^\uparrow$ *schedules and the shared finite tail threshold surface used by chapter 560; no consumer may replace those windows by quotient streams, selected limits, host multiplication, or an ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window obligations`

Proof. Begin with the carrier and NameCert obligations. Definition 464.1 displays the two source rows and paired window rows, while Definition 464.2 makes those rows part of the local certificate surface.

Use the StreamName handoff of Definition 464.20. It forces every product read to pass through W_X, W_Y before Q_X, Q_Y, M_X, M_Y, P, and E can be inspected.

Finally read the shared-tail consumer through chapter 560. The tail-meet packet consumes the same kind of finite threshold window, so the CauchyProduct root obligation is a paired finite-window row rather than a quotient or completed-real operation. □

Theorem 464.4 (CauchyProduct root budget obligations)**.** *The product-radius budget obligation of an accepted* $\mathsf{CauchyProduct}^\uparrow$ *packet is a finite* BHist *route from the paired radius rows* M_X, M_Y *and dyadic rows* Q_X, Q_Y *through the product observation* P *to the classifier row* E *and ledger row* L*. It is routed only by the displayed* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^\uparrow}$ *row, and it exposes no radius oracle, host multiplication estimate, quotient stream, selected limit, trichotomy row, or ambient completeness witness.*

Proof. Start with the source packet. The only budget-relevant coordinates are $M_X, M_Y, Q_X, Q_Y, P, E, L$ together with the structural rows H, C, N.

Apply Definition 464.2 and Definition 464.20. These make the dyadic and radius rows visible only after the paired finite windows, and they route the product observation through the same local certificate.

The budget route is then exact by Definition 464.29. Since that theorem binds the product-radius ledger to the displayed path into E, no unrecorded radius oracle or analytic primitive can enter the root obligation. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget obligations`

Theorem 464.5 (CauchyProduct root Real seal obligations)**.** *Every root-facing* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through the finite rows*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and through the shared tail-threshold consumption surface of chapter 560. The seal read cannot escape to host multiplication, RealUp completeness, quotient-stream equality, selected limits, trichotomy, independent real carriers, or an ambient completeness operation.

Proof. First combine the root window obligations with the root budget obligations. The former fixes the paired finite windows, and the latter fixes the product-radius route from those windows through the dyadic observations to the classifier and ledger.

Next apply Definition 464.21. The $\mathsf{Real}^{\uparrow}$ seal reads the product only through the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E together with the displayed ledger, transport, continuation, provenance, and local naming rows.

Finally use Definition 464.30. It exhausts every real-facing consumer route by the same finite inventory, while the sibling tail-meet packet supplies only the shared finite threshold surface. Hence the root seal obligation is finite-row based and has no escape to the excluded analytic objects. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root real seal obligations`

Theorem 464.6 (CauchyProduct root window seal compatibility)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the paired $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y *and the product-radius budget recorded through* M_X, M_Y, Q_X, Q_Y, P, L *are the only route through which a* $\mathsf{Real}^{\uparrow}$ *seal consumer reaches the classifier row* E*. The route is the one named by the root window obligations, root budget obligations, and root Real seal obligations; it admits no host multiplication, quotient stream, selected limit, trichotomy row, or ambient completeness witness.*

Proof. Begin with Definition 464.3. It places every root-facing product read downstream of the paired finite windows W_X, W_Y.

Apply Definition 464.4. The product-radius route is then fixed through M_X, M_Y, Q_X, Q_Y, P, L before the classifier row can be inspected.

Finally use Definition 464.5. The $\mathsf{Real}^{\uparrow}$ seal reads E only after the paired windows and budget route have been consumed, so none of the excluded analytic objects has a coordinate in the accepted packet. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window seal compatibility`

Theorem 464.7 (CauchyProduct root obligation package)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet supplies a root-unblock package consisting of carrier admission, paired-window source rows, product-budget exactness, Real seal compatibility,* hsame *and* Cont *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows. The package is the finite surface*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

and it excludes host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root obligation package`

Proof. First read carrier admission and NameCert obligations through Definition 464.1 and Definition 464.2. These give the two source names, paired windows, dyadic rows, product observation, classifier row, transport rows, continuation routes, ledger, provenance, and local naming row.

Next assemble the root route. The root window obligations provide the paired-window source surface, the root budget obligations provide the product-radius accounting, and Definition 464.6 identifies the Real seal route through the same rows.

Finally apply the root Real seal obligations. They keep the seal consumer inside the displayed packet and preserve the exclusions on host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, and ambient completeness. The listed rows therefore form the root-unblock package. □

Theorem 464.8 (CauchyProduct root source-pair lock)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the two $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows* X, Y *and their paired* $\mathsf{StreamName}^{\uparrow}$ *window rows* W_X, W_Y *are locked before any product observation row is read. A consumer cannot inspect* P *until the carrier admission, local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *obligations, and the* $\mathsf{StreamName}^{\uparrow}$ *handoff have exposed the two source rows as one paired finite-window route.*

Proof. Read the carrier from Definition 464.1. The first four rows of Z are the two regular Cauchy sources and their paired stream windows, before the radius, dyadic observation, product, classifier, and seal-facing rows.

Apply Definition 464.2. It makes paired source admission and paired window scheduling part of the local certificate surface rather than optional consumer data.

Finally use Definition 464.20. It routes every product read through W_X, W_Y before exposing Q_X, Q_Y, M_X, M_Y, P, or E, so the accepted packet locks the two source rows before any product observation is available. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window read stability`

Theorem 464.9 (CauchyProduct root product observation route)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the product observation row* P *is reachable only along the finite route*

$$X, Y \longmapsto W_X, W_Y \longmapsto Q_X, Q_Y \longmapsto M_X, M_Y \longmapsto P.$$

The route reads paired $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic observation rows, and radius rows, and it exposes no host multiplication primitive, quotient stream, selected limit, independent real carrier, or ambient completeness witness.*

Proof. Start from the source-pair lock in Definition 464.8. The consumer has already passed through the two source rows and their paired stream windows before any product-facing row is visible.

Use Definition 464.20. The handoff exposes Q_X, Q_Y and M_X, M_Y from the paired schedules before the product row P is formed.

Now apply Definition 464.4. The product-radius budget keeps P inside the same finite BHist route and rules out any extra radius oracle or analytic multiplication coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root product observation route`

Theorem 464.10 (CauchyProduct root downstream unblock surface)**.** *The root-unblock surface of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is the finite union of the source-pair lock, the product-observation route, the root budget obligations, the root* $\mathsf{Real}^{\uparrow}$ *seal obligations, and the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows. Downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *product consumers read only this union and cannot import quotient streams, host multiplication, selected limits, trichotomy, independent real carriers, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root downstream unblock surface`

Proof. First combine Definition 464.8 and Definition 464.9. These identify the source-window side and the finite product-observation path into P.

Next apply Definition 464.4. The same route is extended through the product-radius ledger toward E and L, with no extra analytic coordinate.

Finally use Definition 464.5 together with Definition 464.2. The seal, transport, continuation, provenance, and local naming rows are already part of the displayed packet, so the downstream unblock surface is exactly the stated finite union. □

Theorem 464.11 (CauchyProduct paired-window source admission)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the rows X, Y, W_X, W_Y *are admitted only as paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window source rows over the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *inputs. The admission route is the one exposed by the sibling surfaces chapter 126 and chapter 301; it has no quotient-stream, selected-limit, host multiplication, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket paired window source admission`

Proof. Begin with Definition 464.7. It identifies X, Y, W_X, W_Y as the source side of the root package for the product packet.

Apply Definition 464.3. The two source names can be consumed only through the paired finite windows W_X, W_Y, and the route is downstream of the displayed $\mathsf{StreamName}^{\uparrow}$ schedule rows.

The sibling StreamName and RegSeqRat surfaces provide the dependency anchors for the window and source-name reads. Because the root package excludes quotient streams, selected limits, host multiplication, and ambient completeness, no other source-admission route is available. □

Theorem 464.12 (CauchyProduct budget route exactness)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the rows*

$$M_X, M_Y, Q_X, Q_Y, P, L$$

are the exact finite budget route into the RegSeqRat$^{\uparrow}$ *classifier row* E*. The route reads dyadic observations through chapter 291, records its regular-rational classifier through chapter 301, and admits no radius oracle, quotient stream, selected limit, host multiplication estimate, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget route exactness`

Proof. Start from Definition 464.4. It lists $M_X, M_Y, Q_X, Q_Y, P, E, L$ as the finite product-radius route.

Use Definition 464.7 to keep the same route inside the carrier, transport, continuation, provenance, and local naming rows. The dyadic observation surface comes from the displayed Q_X, Q_Y rows, and the product observation P is the only combining row.

Finally apply Definition 464.6. The classifier row E is reached only after the budget ledger L has consumed those rows, so the sibling DyadicRatCore and RegSeqRat surfaces are the whole classifier-facing budget path. □

Theorem 464.13 (CauchyProduct finite budget composition)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, the paired* StreamName$^{\uparrow}$ *window route, root budget route,* RegSeqRat$^{\uparrow}$ *seal route,* RealObservationBudget$^{\uparrow}$ *seal, and* RealSeries$^{\uparrow}$ *consumer boundary compose into one finite CauchyProduct budget handoff over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The composed handoff is the displayed route through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the budget-facing rows consumed by RealObservationBudget$^{\uparrow}$ *and the terminal* RealSeries$^{\uparrow}$ *consumer row. It exposes no ambient* Real$^{\uparrow}$ *completeness theorem, quotient sequence equality, selected representative, host infinite sum, host multiplication primitive, trichotomy row, or independent real carrier.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket finite budget composition`

Proof. Start with paired-window source admission. By Definition 464.11, the source rows X, Y are read only through the paired finite StreamName$^{\uparrow}$ windows W_X, W_Y.

Next apply budget route exactness. Definition 464.12 fixes the route from M_X, M_Y, Q_X, Q_Y, P, L into the classifier row E, and Definition 464.16 ties that classifier to the same paired windows.

Now use the Real seal boundary. The root RealObservationBudget seal boundary of Definition 464.19 keeps budget-facing reads inside the displayed product and budget rows.

Finally apply the RealSeries consumer boundary of Definition 464.23. It consumes the public CauchyProduct surface only after the finite budget handoff has been assembled, so none of the excluded analytic objects has a coordinate in the composed route. □

Theorem 464.14 (CauchyProduct main surface public handoff)**.** *Every public* CauchyProduct$^{\uparrow}$ *consumer reads the paired* RegularCauchyName$^{\uparrow}$ *sources,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *budget rows,* RegSeqRat$^{\uparrow}$ *classifier,* RealObservationBudget$^{\uparrow}$ *seal, and* RealSeries$^{\uparrow}$ *consumer boundary through one displayed route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. For an accepted packet, that route is the finite surface*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the budget-facing rows consumed by RealObservationBudget$^{\uparrow}$ *and the terminal* RealSeries$^{\uparrow}$ *consumer row. It exposes no quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket main surface public handoff`

Proof. Begin with paired-window source admission. The theorem Definition 464.11 forces the two $\mathsf{RegularCauchyName}^{\uparrow}$ rows X, Y to be read only through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y.

Next assemble the budget route. By Definition 464.12, the dyadic rows M_X, M_Y, Q_X, Q_Y, P, L are the exact finite route into the classifier row E, and Definition 464.13 composes that route with the public budget handoff.

Now use the root seal and series boundaries. The root $\mathsf{RealObservationBudget}^{\uparrow}$ seal obligations of Definition 464.19 keep budget-facing reads inside the displayed product and budget rows, while Definition 464.23 permits a $\mathsf{RealSeries}^{\uparrow}$ consumer only after the public CauchyProduct surface has been read.

Finally repack the route through the structural rows. The displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows are already part of the accepted packet, so no public consumer has a coordinate for quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 464.15 (CauchyProduct budget public handoff)**.** *Every product-facing* $\mathsf{Real}^{\uparrow}$ *or series consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet must factor through the public budget handoff*

$\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRa}$

Concretely, the consumer first reads the two regular Cauchy source names, their shared-tail budget and paired $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and product rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, the* $\mathsf{Real}^{\uparrow}$ *seal, the shared-tail modulus, the diagonal-tail selector rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row before reading the product packet.*

The handoff exports no quotient streams, selected limits, host multiplication primitive, trichotomy row, countable-choice schedule, independent real carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget public handoff`

Proof. Start with the main surface public handoff of Definition 464.14. It already routes every public product consumer through the paired source names, windows, dyadic budget rows, classifier, real seal, transport, replay, provenance, and local name.

Next use the finite budget composition of Definition 464.13. It composes the paired-window route with the RealObservationBudget seal and RealSeries consumer boundary, so the product-facing Real or series consumer sees one budget route rather than separate local rows.

Finally insert the shared-tail and diagonal-tail surfaces from the root window obligations and the sibling regular Cauchy tail budget. These rows supply the modulus and selector needed before a product packet can be read, while the accepted carrier has no coordinate for quotient streams, selected limits, trichotomy, countable choice, host multiplication, or ambient completeness. □

Theorem 464.16 (CauchyProduct paired-window ledger classifier)**.** *The paired window rows* W_X, W_Y *together with the ledger route* M_X, M_Y, Q_X, Q_Y, P, L *determine the classifier-facing product row* E *consumed by the* $\mathsf{Real}^{\uparrow}$ *seal. This determination is routed through the sibling surfaces chapter 126, chapter 291, chapter 301, and chapter 38, and it exposes no independent real carrier, selected limit, quotient stream, trichotomy row, host multiplication primitive, or ambient completeness operation.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket paired window ledger classifier`

Proof. First combine Definition 464.11 with Definition 464.12. The paired windows provide the source reads, and the budget route provides the dyadic product observation and ledger feeding E.

Next use Definition 464.5. The $\mathsf{Real}^{\uparrow}$ seal consumes the product only after the classifier row E has been reached through the same finite windows, dyadic observations, product row, ledger, transport, continuation, provenance, and local naming rows.

The sibling surfaces account for each dependency edge: StreamName for finite windows, DyadicRatCore for observations and product rows, RegSeqRat for the classifier readback, and RealUp for the seal. Therefore the classifier-facing product row is determined by the paired-window ledger route and by no excluded analytic object. □

Theorem 464.17 (CauchyProduct root window seal stability)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the paired window rows W_X, W_Y *remain the unique route to every seal-facing product row under componentwise* hsame *transport and displayed* Cont *replay. A seal consumer reaches* P*,* E*, and the ledger row* L *only through the route fixed by the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the dyadic observations, the product-radius ledger, package provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window seal stability`

Proof. Start with the root window obligations. They force every root-facing product read through the paired finite windows W_X, W_Y before the dyadic rows, radius rows, product row, classifier row, or seal-facing ledger can be inspected.

Now apply the StreamName handoff and window product stability. The handoff fixes the route from W_X, W_Y to Q_X, Q_Y, M_X, M_Y, P, E, while stability keeps that route invariant under the componentwise transport row H and the continuation replay row C.

Finally use the root Real seal obligations. Since the $\mathsf{Real}^{\uparrow}$ seal can read the product only after the same window and ledger route has been consumed, no second path to a seal-facing row is available inside the accepted packet. □

Theorem 464.18 (CauchyProduct root RegSeqRat seal stability)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the product observation row P *and the product-radius ledger row* L *determine the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *before any* $\mathsf{Real}^{\uparrow}$ *seal read is available. The determining route is the finite path through the paired windows, dyadic observations, radius rows, displayed* hsame *transports, displayed* Cont *routes, package provenance, and local NameCert row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regseq seal stability`

Proof. First apply the root budget obligations. They bind $M_X, M_Y, Q_X, Q_Y, P, E, L$ as the finite product-radius route downstream of the paired windows.

Use radius ledger exactness to identify L with the displayed route from the input-radius rows and dyadic observations through the product observation P. Then apply radius product regularity to recover the classifier row E from exactly that ledgered product route.

The root Real seal obligations place every real-facing read after this classifier. Hence the $\mathsf{RegSeqRat}^{\uparrow}$ row is determined before the seal is consumed, and the seal cannot supply an independent classifier, quotient stream, selected limit, trichotomy row, host multiplication row, or ambient completeness coordinate. □

Theorem 464.19 (CauchyProduct root RealObservationBudget seal boundary)**.** *Every* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reads the product only through the union of the CauchyProduct rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The boundary exposes no quotient stream, selected limit, trichotomy row, host multiplication primitive, independent real carrier, ambient completeness coordinate, or additional real-facing product route.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root real budget seal boundary`

Proof. Begin with root window seal stability and root RegSeqRat seal stability. These place the product route through the paired windows, the product observation, the radius ledger, and the classifier row before any real-facing budget consumer can read the seal.

Now compose with the Real observation budget handoff. It routes the product seal through the accepted budget packet before the consumer reaches its real-facing rows.

Use the observation budget RegSeq handoff to identify the classifier read with the budgeted regular-sequence row. Finally apply the budgeted product consumer boundary; it exhausts the joined interface by the displayed product rows and budget rows, leaving no coordinate for any excluded analytic object. □

Theorem 464.20 (CauchyProduct StreamName handoff)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet routes a product read through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y *before exposing the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations* Q_X, Q_Y*, the dyadic-radius rows* M_X, M_Y*, the product observation* P*, and the pointwise* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*. No consumer can read a product row without first reading those finite schedules and their displayed provenance.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket streamname handoff`

Proof. Start with the two source rows X and Y. Each is read only through its displayed $\mathsf{RegularCauchyName}^{\uparrow}$ source row and the matching finite $\mathsf{StreamName}^{\uparrow}$ schedule.

The dyadic product row is downstream of the schedules. Once W_X and W_Y have exposed Q_X, Q_Y and M_X, M_Y, the row P records the finite dyadic product observation and E repacks it as a pointwise regular rational classifier.

The transport and continuation rows preserve this order. Since H, C, L, N are attached to the same paired windows, a consumer has no alternate route to a host sequence, quotient product, selected limit, or ambient real operation. □

Theorem 464.21 (CauchyProduct Real seal boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reads only the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *built from the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic-radius rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product observation, displayed* hsame *transports,* Cont *routes, finite ledger, package provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row. The seal boundary exports no RealUp completeness theorem, quotient stream, selected limit, trichotomy row, independent real carrier, or host multiplication primitive.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal boundary`

Proof. Factor the real-facing read through the pointwise regular classifier. By Definition 464.20, the classifier is already downstream of the two finite stream windows and their dyadic product observation.

The real seal is recorded in the ledger row rather than supplied by an ambient carrier. Thus a $\mathsf{Real}^{\uparrow}$ consumer receives E together with H, C, L, N, which are exactly the displayed transport, continuation, ledger, package, and naming rows.

No other coordinate in Definition 464.1 can supply completeness, quotient equality, a selected limit, trichotomy, or host multiplication. The real-facing boundary is therefore exhausted by the finite product packet. □

Theorem 464.22 (CauchyProduct public interface export)**.** *Every public consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reads the product through the paired-window source rows, the finite budget route, the ledger classifier, the* $\mathsf{StreamName}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary. Equivalently, the public export surface is the finite route through* $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$ *together with the displayed source, budget, provenance, continuation, and naming rows; it exports no quotient stream equality, selected infinite sum, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness claim.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public interface export`

Proof. First use the paired-window source surface. The StreamName handoff, Definition 464.20, places every product read behind the two finite windows W_X, W_Y and the dyadic observation rows Q_X, Q_Y, M_X, M_Y.

Next compose the budget route with the classifier route. The root RealObservationBudget seal boundary identifies the budget-facing rows, while the root RegSeqRat seal stability theorem makes the classifier row E downstream of the product observation and ledger.

Finally apply the Real seal boundary, Definition 464.21. It exhausts the public read through the displayed finite rows and excludes quotient streams, selected limits, trichotomy, host multiplication, independent real carriers, and ambient completeness. This is exactly the stated export surface. □

Theorem 464.23 (CauchyProduct RealSeries consumer boundary)**.** *Every* $\mathsf{RealSeries}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. The consumer may read the product only through the public interface of Definition 464.22; it receives no selected infinite sum, quotient stream equality, host multiplication primitive, order trichotomy, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real series consumer boundary`

Proof. Begin with a RealSeries-facing read. The public interface export theorem routes the read through the paired finite windows, dyadic ledgers, product observation, classifier, transports, continuations, provenance, and local name.

The series consumer can then inspect only the named BEDC rows. Its stream access is through $\mathsf{StreamName}^{\uparrow}$, its rational readback is through $\mathsf{RegSeqRat}^{\uparrow}$, its real-facing seal is through $\mathsf{Real}^{\uparrow}$, and its order-facing comparison surface is through $\mathsf{RealAlgOrder}^{\uparrow}$ before the $\mathsf{RealSeries}^{\uparrow}$ row is consumed.

The structural rows BHist, hsame, Cont, Pkg, and NameCert preserve that route without adding analytic data. Applying Definition 464.21 removes any remaining path to a selected infinite sum,

quotient stream equality, host multiplication primitive, order trichotomy, independent real carrier, or ambient completeness witness. □

Theorem 464.24 (CauchyProduct public RealSeries budget export)**.** *Every public* $\mathsf{RealSeries}^{\uparrow}$*-facing* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer exports a product claim only after the finite budget route through the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, dyadic product rows,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum ledger has been displayed. For an accepted packet, the export route is*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the terminal $\mathsf{RealSeries}^{\uparrow}$ *consumer row.*

The public export surface supplies no quotient stream equality, selected infinite sum, host multiplication primitive, order trichotomy, independent real carrier, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public real series budget export`

Proof. Begin with the root budget obligations of Definition 464.4. They bind $M_X, M_Y, Q_X, Q_Y, P, E, L$ as the product-radius route downstream of the paired windows.

Next use the StreamName handoff and the public interface export, Definition 464.20 and Definition 464.22. These force the stream windows and regular-rational classifier to be consumed before any public product claim is named.

Finally apply Definition 464.23. The $\mathsf{RealSeries}^{\uparrow}$ row reads the product only after the $\mathsf{Real}^{\uparrow}$ seal and the displayed public CauchyProduct surface, so the partial-sum ledger cannot import an untracked analytic object. □

Theorem 464.25 (CauchyProduct public observation budget export)**.** *Every public* $\mathsf{RealObservationBudget}^{\uparrow}$ *handoff from an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet exports the product through the same paired-window, regular-readback, dyadic-budget, and real-seal route as the RealSeries consumer. The handoff is the finite route through*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

plus the accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *budget row; it exports no radius oracle, quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness claim.*

Proof. Use the root RealObservationBudget seal boundary of Definition 464.19. It identifies the budget-facing consumer as downstream of the product rows and the observation-budget rows.

Compose this with Definition 464.4. The dyadic product observation and radius ledger are fixed before the classifier row E and ledger row L can be inspected.

The StreamName handoff and Real seal boundary, Definition 464.20 and Definition 464.21, keep the public observation-budget route inside the displayed finite rows. Therefore the handoff cannot export any excluded analytic or host-side object. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public observation budget export`

Theorem 464.26 (CauchyProduct public RealSeries export)**.** *Every public* $\mathsf{RealSeries}^{\uparrow}$*-facing export of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is the composition of paired-window source admission, budget route exactness, paired-window ledger classification, public interface export, and the RealSeries consumer boundary. The exported surface is exactly the finite route through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the $\mathsf{RealSeries}^{\uparrow}$ *row. It supplies no quotient stream, selected infinite sum, host multiplication, order trichotomy, independent real carrier, or ambient completeness.*

Proof. Start with paired-window source admission and budget route exactness. The source rows X, Y, W_X, W_Y and the budget rows M_X, M_Y, Q_X, Q_Y, P, L are fixed by Definition 464.11 and Definition 464.12.

Next apply the paired-window ledger classifier. Definition 464.16 places the classifier row E downstream of those windows, dyadic observations, product row, and ledger.

Now use the public interface export. Definition 464.22 packages the same finite route with the displayed transport, continuation, provenance, and local naming rows H, C, N.

Finally apply the RealSeries consumer boundary. It attaches the $\mathsf{RealSeries}^{\uparrow}$ consumer row only after the public CauchyProduct surface has been read, and it rules out the excluded quotient, sum-selection, host-multiplication, trichotomy, independent-carrier, and completeness objects. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public real series export`

Theorem 464.27 (CauchyProduct ledger exhaustion)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, every consumer route is exhausted by the rows named in Definition 464.1 together with the obligations of Definition 464.2, the* $\mathsf{StreamName}^{\uparrow}$ *handoff of Definition 464.20, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 464.21. The ledger contains exactly the paired input radii, product-radius row, regular rational readback route, seal handoff, displayed transports, continuations, package provenance, and local certificate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket ledger exhaustion`

Proof. Inspect the source packet. Its only product-facing data are the two regular names, paired windows, paired radius rows, dyadic observations, product observation, classifier row, transports, continuations, ledger, and local naming row.

The NameCert obligations attach every accepted read to those displayed rows. The StreamName handoff fixes the order from source windows to dyadic product observation to regular classifier, while the Real seal boundary places each real-facing read downstream of the same classifier and ledger.

Therefore the listed rows exhaust the certificate. A route seeking quotient streams, host multiplication, selected limits, trichotomy, an independent real carrier, or ambient completeness has no coordinate in the accepted packet. □

Theorem 464.28 (CauchyProduct window product stability)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the paired $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y *stably determine the dyadic product observation* P*. Componentwise* hsame *transport of the two source names, both windows, both radius rows, and both dyadic observations preserves the product row and keeps every read inside the displayed window route.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket window product stability`

Proof. Project the source packet of Definition 464.1. The product row P is downstream of the paired window rows and the dyadic observations Q_X, Q_Y.

Apply the StreamName handoff theorem. It forces a product read to pass through W_X, W_Y before exposing Q_X, Q_Y, the radius rows, and the product observation.

The transport coordinate H compares exactly those rows componentwise, and the continuation coordinate C replays the displayed route from windows to product observation. Hence stable transport preserves the paired-window product route without adding host multiplication or a quotient product. □

Theorem 464.29 (CauchyProduct radius ledger exactness)**.** *The product-radius ledger of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is exact for the displayed route from paired input radii* M_X, M_Y *through the dyadic product observation* P *to the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*. The ledger records no independent radius oracle, selected limit, trichotomy row, host multiplication primitive, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket radius ledger exactness`

Proof. Use window product stability to bind the product observation to the paired windows, input radii, and dyadic observations. This places the product-radius read downstream of $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$.

The NameCert obligations identify E as the pointwise regular rational classifier exposed from the product observation together with its finite radius information. The finite ledger L records the paired input radii, product-radius row, seal handoff, and package provenance.

Finally apply ledger exhaustion. Since every consumer route is exhausted by the source packet and its displayed obligations, the radius ledger is exact for that route and contains no unlisted analytic primitive. □

Theorem 464.30 (CauchyProduct Real seal consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through the paired windows, dyadic product observation, product-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, displayed transport and continuation rows, finite ledger, package provenance, and local naming row. It has no access to host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal consumer exhaustion`

Proof. Start with the Real seal boundary. It places each real-facing read downstream of the classifier row E, which is built from the paired windows, dyadic observations, and displayed ledger.

Use radius ledger exactness to identify the classifier input with the product-radius route recorded by L. The transport and continuation rows H, C are the only allowed movement from source windows to the product observation and then to the regular rational readback.

Apply ledger exhaustion at the end. The rows named in the source packet exhaust every consumer route, so the real seal cannot import host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness. □

Theorem 464.31 (CauchyProduct paired-window closure)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet*

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

the two $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows* X, Y *close their product read only through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y*. Those windows expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations* Q_X, Q_Y*, the paired radius rows* M_X, M_Y*, the product observation* P*, the product-radius ledger in* L*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*; no quotient stream, selected limit, trichotomy row, ambient completeness witness, or host multiplication primitive is admitted.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket paired window closure`

Proof. Begin with the StreamName handoff. It forces every product read from X, Y to pass through the paired finite windows before the dyadic observations and radius rows are visible.

Use window product stability next. The product observation P is stably determined by $W_X, W_Y, Q_X, Q_Y, M_X, M_Y$ under componentwise transport.

Finally apply radius ledger exactness and ledger exhaustion. The product-radius row and classifier exposure are exactly the rows recorded in L and E, so the paired-window closure has no coordinate for quotient streams, selected limits, trichotomy, ambient completeness, or host multiplication. □

Theorem 464.32 (CauchyProduct radius product regularity)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the product-radius ledger turns the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, and product observation P into the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row E. The regularity evidence is exactly the finite radius route recorded by L and transported by H, C; it is not a host multiplication theorem, quotient stream construction, selected limit, trichotomy decision, or ambient completeness argument.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket radius product regularity`

Proof. Use radius ledger exactness to bind the product-radius route to the displayed path from M_X, M_Y and Q_X, Q_Y through P to E.

The paired-window closure identifies the source of those rows with the two regular Cauchy names and their finite windows. Window product stability keeps the product observation invariant under the displayed transport rows.

Ledger exhaustion then prevents any further regularity source. The regular sequence classifier is therefore certified by the finite radius ledger alone, not by host multiplication, quotient streams, selected limits, trichotomy, or completeness. □

Theorem 464.33 (CauchyProduct Real multiplication seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *handoff of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reads multiplication only as the sealed* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row E built from paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the product-radius ledger, displayed* hsame *transports,* Cont *routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *evidence. The handoff exports no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real multiplication seal handoff`

Proof. Start with Real seal consumer exhaustion. It already factors every real-facing read through the paired windows, product observation, radius ledger, classifier row, transports, continuations, provenance, and local naming data.

Apply radius product regularity to identify the classifier input. The row E is supplied by the finite product-radius ledger over the dyadic observations and product row, not by an ambient multiplication operation.

Use the Real seal boundary at the end. Since the seal sees only the classifier and its finite ledger, the multiplication handoff cannot expose host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or completeness. □

Theorem 464.34 (CauchyProduct product-observation seal obligations)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *windows, paired* $\mathsf{StreamName}^{\uparrow}$ *schedules, dyadic-radius rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product observation P,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier E,* $\mathsf{Real}^{\uparrow}$ *seal handoff, displayed* hsame *transports,* Cont *routes, finite ledger,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row*

jointly supply the seed-to-obligation package for product observation. The package admits no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket product observation seal obligations`

Proof. Begin with the NameCert obligations. They expose the carrier, paired regular Cauchy sources, finite stream schedules, dyadic-radius routing, product observation P, regular rational classifier E, seal handoff, transport rows, continuation rows, package provenance, ledger, and local naming row.

Use the StreamName handoff to bind the order of observation. A product read must pass through W_X, W_Y before Q_X, Q_Y, the radius rows, P, and then E are visible.

The radius ledger exactness theorem identifies the product-radius row with that displayed route, and radius product regularity turns the route into the regular classifier row. Thus the observation supplied to the seal is the finite dyadic product packet, not an external multiplication rule.

Finally apply the Real multiplication seal handoff. Since every real-facing consumer reads multiplication only through E and the recorded ledger, the listed rows form the obligation package and no unlisted analytic primitive can enter. □

Theorem 464.35 (CauchyProduct RealCompleteness diagonal seal route)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet be consumed by a* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit packet through the real-facing seal route. The contribution from* $\mathsf{CauchyProduct}^{\uparrow}$ *is only the finite product-observation surface*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

These rows supply the paired $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic observations, product-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row. The diagonal seal route receives no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness from the product packet.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real completeness diagonal seal route`

Proof. Start with the StreamName handoff of Definition 464.20. It places the product observation downstream of the paired finite windows and their dyadic observation rows, so a diagonal-limit consumer first sees $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$ before it can read P.

Next apply radius ledger exactness and radius product regularity, Definition 464.29 and Definition 464.32. These identify the product-radius ledger as the only route from the dyadic product observation to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E.

Use the product-observation seal obligations and Real multiplication seal handoff, Definition 464.34 and Definition 464.33. They make the real-facing multiplication read consume only the displayed ledger, classifier, transports, continuations, package provenance, and local certificate.

Finally pull this finite surface back along the RealCompleteness scoped seal discipline of Definition 450.18. Since the pullback receives only the displayed product rows, the diagonal seal route cannot acquire host multiplication, quotient streams, selected limits, trichotomy, an independent real carrier, or ambient completeness from $\mathsf{CauchyProduct}^{\uparrow}$. □

Theorem 464.36 (CauchyProduct constant left factor)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose left source row X is read as a constant regular Cauchy name through the displayed* $\mathsf{RegularCauchyName}^{\uparrow}$

and StreamName$^{\uparrow}$ *schedules, multiplication by that left factor is certified only by the existing product packet. The consumer route factors through the paired windows, dyadic product observation P, product-radius ledger,* RegSeqRat$^{\uparrow}$ *classifier E, and* Real$^{\uparrow}$ *multiplication seal handoff, and it introduces no host scalar multiplication primitive or quotient equality.*

Proof. Begin with the StreamName handoff. Even when the left source is constant, the packet still reads X through its displayed RegularCauchyName$^{\uparrow}$ row and the window W_X before the dyadic observation Q_X is available.

Use paired-window closure and radius product regularity. These force the product observation P and the classifier row E through the same finite route from $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$ to the product-radius ledger.

Finally apply the Real multiplication seal handoff. The seal receives only E with its recorded ledger and transports, so the constant left factor is a specialization of the displayed product packet rather than a host scalar operation. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket constant left factor`

Theorem 464.37 (CauchyProduct constant right factor)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *packet whose right source row Y is read as a constant regular Cauchy name through the displayed* RegularCauchyName$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *schedules, multiplication by that right factor is certified only by the existing product packet. The consumer route factors through the paired windows, dyadic product observation P, product-radius ledger,* RegSeqRat$^{\uparrow}$ *classifier E, and* Real$^{\uparrow}$ *multiplication seal handoff, and it introduces no host scalar multiplication primitive or quotient equality.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket constant right factor`

Proof. Start with the right source row. The StreamName handoff makes W_Y the only route by which the constant right input exposes its dyadic observation Q_Y and radius row M_Y.

Apply paired-window closure and radius product regularity. The right constant read is paired with the left window through the product observation P, and the classifier row E is supplied by the finite product-radius ledger.

The Real multiplication seal handoff then exhausts the real-facing route. Since the seal sees only the displayed classifier and ledger, the right constant factor does not add host scalar multiplication, quotient streams, selected limits, or ambient completeness. □

Theorem 464.38 (CauchyProduct bilinear tail window)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *packet and any shared tail read admitted by its paired* StreamName$^{\uparrow}$ *windows, the two one-sided constant-factor routes and the product route are all witnessed by the same finite window surface*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

The bilinear tail window is therefore a NameCert obligation over the displayed product-radius ledger and Real$^{\uparrow}$ *seal handoff, not a field law, quotient equality, selected-limit argument, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket bilinear tail window`

Proof. First bind the shared tail read to the paired-window closure. The windows W_X, W_Y expose the two source observations, their radius rows, the dyadic product observation, and the regular rational classifier through one finite route.

Next use the two constant-factor obligations. The left and right constant reads are not new operations; each is a specialization of the same displayed product packet and therefore uses the same product observation P and classifier row E.

Finally use radius product regularity and the Real multiplication seal handoff. These identify the product-radius ledger as the only regularity evidence visible to the seal, so the bilinear tail window is exactly the finite NameCert surface and not an external algebraic law. □

Theorem 464.39 (CauchyProduct constant-factor bilinear seal)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the left constant-factor route, the right constant-factor route, and the bilinear tail window are all specializations of the same finite product packet. Every* $\mathsf{Real}^{\uparrow}$ *seal-facing multiplication read therefore factors through the paired windows, dyadic product observation, product-radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row:*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

The shared seal boundary asserts only this finite packet route. It asserts no host multiplication, quotient equality, selected limit, trichotomy, independent real carrier, or ambient completeness principle.

Proof. Start with Definition 464.36 and Definition 464.37. Each constant-factor read passes through the displayed $\mathsf{RegularCauchyName}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows before exposing the dyadic observations and product-radius ledger.

Use Definition 464.38. The bilinear tail window states that the two one-sided constant-factor routes and the product route share the same finite surface $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$.

Apply radius product regularity to identify the product-radius ledger with the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E. Then apply the $\mathsf{Real}^{\uparrow}$ multiplication seal handoff to make the seal-facing consumer read only E together with the recorded ledger, transports, continuations, provenance, and local naming row.

The assembled route is therefore one product-packet seal boundary. Since none of its rows is a host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or completeness witness, the listed exclusions remain in force. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket constant factor bilinear seal`

Theorem 464.40 (CauchyProduct left-factor classifier congruence)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet with product route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

replace the left $\mathsf{RegularCauchyName}^{\uparrow}$ *source by another source whose* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier is equivalent through the displayed* $\mathsf{StreamName}^{\uparrow}$ *handoff,* hsame *transports,* Cont *routes, package provenance, finite radius ledger, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row. Then the product observation route descends to the same displayed classifier row* E *and the same* $\mathsf{Real}^{\uparrow}$ *multiplication seal handoff. The descent introduces no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket left factor classifier congruence`

Proof. Begin with the left StreamName handoff. Classifier-equivalence of the replacement source is read only through the displayed window W_X, radius row M_X, dyadic observation Q_X, transport row H, continuation route C, provenance row L, and local naming row N.

Next pair this transported left read with the unchanged right window. Window product stability keeps the dyadic product observation P on the same finite route, so the product packet has no additional coordinate from which a host product could be read.

Finally apply radius ledger exactness, radius product regularity, and the Real multiplication seal handoff. These identify the classifier row as the same E and make the real-facing seal consume only the recorded ledger, so the left-factor replacement is a classifier descent inside the packet surface. □

Theorem 464.41 (CauchyProduct right-factor classifier congruence)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet with product route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

replace the right $\mathsf{RegularCauchyName}^{\uparrow}$ *source by another source whose* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier is equivalent through the displayed* $\mathsf{StreamName}^{\uparrow}$ *handoff,* hsame *transports,* Cont *routes, package provenance, finite radius ledger, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row. Then the product observation route descends to the same displayed classifier row E and the same* $\mathsf{Real}^{\uparrow}$ *multiplication seal handoff. The descent introduces no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket right factor classifier congruence`

Proof. Start with the right StreamName handoff. The replacement source can expose its classifier only through W_Y, M_Y, Q_Y, the displayed transport and continuation rows, the package row, and the local naming evidence.

Now pair that transported right read with the unchanged left window. Window product stability preserves the dyadic product observation P over the same finite product surface and leaves the packet with the original radius ledger.

Use radius ledger exactness and radius product regularity to recover the same regular rational classifier row E. The Real multiplication seal handoff then reads only that row and its ledger, excluding host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, and ambient completeness. □

Theorem 464.42 (CauchyProduct two-factor classifier descent)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet with product route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

simultaneously replace the left and right $\mathsf{RegularCauchyName}^{\uparrow}$ *sources by classifier-equivalent sources witnessed through the displayed* $\mathsf{StreamName}^{\uparrow}$ *handoffs,* hsame *transports,* Cont *routes, package provenance, finite product-radius ledger, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *evidence. The paired replacement descends to the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row E and therefore to the same* $\mathsf{Real}^{\uparrow}$ *multiplication seal handoff. The resulting multiplication surface contains no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket two factor classifier descent`

Proof. First transport the two source reads separately. The left congruence supplies the route through $W_X, M_X, Q_X, H, C, L, N$, and the right congruence supplies the route through $W_Y, M_Y, Q_Y, H, C, L, N$.

Next recombine the transported reads at the product observation. Window product stability keeps P on the displayed paired-window surface, while ledger exactness keeps the radius evidence tied to the same finite product-radius row.

It remains to pass from the product row to the real-facing consumer. Radius product regularity recovers the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E, and the Real multiplication seal handoff reads only E with its recorded ledger. Thus the simultaneous two-factor descent is a classifier-level handoff, not an appeal to host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness. □

Theorem 464.43 (CauchyProduct swapped-window classifier symmetry)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet with product route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

the swapped finite route

$$W_Y, W_X, M_Y, M_X, Q_Y, Q_X$$

reaches the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *and the same* $\mathsf{Real}^{\uparrow}$ *multiplication seal after the dyadic product observation is replayed through the displayed* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row. This symmetry is internal to the packet surface and contains no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket swapped window classifier symmetry certificate`

Proof. Start with the two one-factor classifier congruences. The left-factor congruence transports the first source read through W_X, M_X, Q_X, and the right-factor congruence transports the second source read through W_Y, M_Y, Q_Y, always along the displayed hsame, Cont, Pkg, and local NameCert rows.

Now read the same two transported sources in the opposite finite window order. The two-factor classifier descent applies to the paired replacement and keeps the product route at the displayed classifier row E rather than exposing a new classifier field.

It remains to replay the product observation through the real-facing seal. Radius product regularity identifies the radius ledger with the same $\mathsf{RegSeqRat}^{\uparrow}$ row E, and the Real multiplication seal handoff consumes that row together with H, C, L, N.

Thus the swapped window route has the same classifier and seal as the original route. Since every step was made through the packet rows, no host multiplication, quotient stream, selected limit, trichotomy, independent real carrier, or ambient completeness witness is introduced. □

Theorem 464.44 (CauchyProduct factor-swap observation surface)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet with product route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

swapping the two $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows swaps only the displayed source-window and radius-observation rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y.$$

The dyadic product observation route P*, classifier row* E*,* hsame *transports* H*,* Cont *continuations* C*, finite ledger* L*, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row* N *remain on the same product surface.*

The swapped read therefore preserves the $\mathsf{Real}^{\uparrow}$ *multiplication seal handoff and introduces no host multiplication primitive, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket factor swap observation surface`

Proof. Begin with the two StreamName handoffs for the source rows. Swapping the source order exchanges the displayed finite windows and their radius-observation rows, so the only rows whose names change are $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$.

Next use window product stability. The paired window route still reaches the same dyadic product observation P, because the product packet records the observation through the finite paired-window surface rather than through a host multiplication operation.

The radius ledger exactness theorem keeps the ledger L tied to that same product observation route. Radius product regularity then recovers the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E from the recorded ledger.

Finally use the two-factor classifier descent together with the Real multiplication seal handoff. The transports H, continuations C, package provenance, and local naming row N are carried along the displayed packet route, so the factor swap changes only the factor-side rows and not the public product observation surface or any forbidden ambient resource. □

Theorem 464.45 (CauchyProduct public multiplication consumer boundary)**.** *Every public multiplication consumer supplied by the* $\mathsf{CauchyProduct}^{\uparrow}$ *packet and read through the* $\mathsf{Real}^{\uparrow}$ *seal can inspect only the finite rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

It cannot inspect host multiplication, quotient streams, selected limits, trichotomy rows, independent real carriers, or ambient completeness. Thus the public multiplication boundary is exactly the displayed NameCert product packet and not any external algebraic or order-theoretic resource.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public multiplication consumer boundary`

Proof. Start from the Real multiplication seal handoff. A seal-facing consumer receives the product only through the recorded $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E and the finite ledger, not through a host product or an independently chosen real representative.

Trace that classifier row backward through the packet. Radius product regularity, radius ledger exactness, and the product-observation seal obligations force every access to E to pass through P and then through the paired window and dyadic observation rows $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$.

The remaining rows are administrative rather than ambient. The transports H, continuations C, package provenance, ledger L, and local naming evidence N exhaust the public consumer interface recorded by the product packet.

Since this exhaustion lists every row visible through the $\mathsf{Real}^{\uparrow}$ seal, there is no remaining coordinate from which a public consumer could read quotient streams, selected limits, trichotomy, an independent carrier, ambient completeness, or host multiplication. □

Theorem 464.46 (CauchyProduct RealAlgOrder multiplication export)**.** *The source packet of Definition 464.1, together with the factor-swap observation surface of Definition 464.44 and the public multiplication consumer boundary of Definition 464.45, supplies the RealAlgOrder-facing multiplication export for* $\mathsf{CauchyProduct}^{\uparrow}$*. The export is the finite product packet*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

read through the Real$^{\uparrow}$ *seal, with factor congruence, factor swap, and public consumer exhaustion all internal to the displayed* RegularCauchyName$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realalgorder multiplication export`

Proof. First assemble the source packet. The carrier source and local NameCert obligations provide the two regular Cauchy source rows, paired StreamName windows, dyadic observations, product observation, classifier row, transports, continuations, ledger, provenance, and naming evidence.

Next add factor congruence and descent. The left-factor, right-factor, and two-factor classifier congruence theorems show that changing either source by an equivalent displayed classifier leaves the product route at the same classifier row E and the same Real multiplication seal handoff.

The factor-swap surface records the symmetric product read visible to downstream algebra consumers. It preserves P, E, H, C, L, N and swaps only the source-side window and radius-observation rows.

Finally use the public multiplication consumer boundary. Since every public consumer reads only the listed packet through the Real$^{\uparrow}$ seal, the RealAlgOrder-facing export has exactly the finite multiplication surface required here and carries no host multiplication, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness witness. □

Theorem 464.47 (CauchyProduct obligation closure)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, the carrier source of Definition 464.1, the StreamName handoff, the* Real$^{\uparrow}$ *seal boundary, ledger exhaustion, radius product regularity, and product-observation seal obligations jointly supply the full* obligation *surface for* CauchyProduct$^{\uparrow}$ *over* RegularCauchyName$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

This closure surface preserves the exclusions on quotient streams, selected limits, trichotomy, host multiplication, independent real carriers, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket obligation closure`

Proof. First collect the carrier source and NameCert obligations. They list the two RegularCauchyName$^{\uparrow}$ sources, paired StreamName$^{\uparrow}$ windows, dyadic-radius rows, DyadicRatCore$^{\uparrow}$ observations, product observation, RegSeqRat$^{\uparrow}$ classifier, Real$^{\uparrow}$ seal handoff, transport, continuation, ledger, provenance, and local naming rows.

Next apply the StreamName handoff and Real seal boundary. These theorems fix the route from the source names through paired finite windows to the dyadic product observation and then to the real-facing classifier row.

Use ledger exhaustion and radius product regularity to identify the finite regularity evidence. The product-radius ledger is the only route turning the paired observations and product row into the displayed RegSeqRat$^{\uparrow}$ classifier.

Finally apply the product-observation seal obligations. They assemble the same rows into the product-observation package and exclude any coordinate for quotient streams, selected limits, trichotomy, host multiplication, independent real carriers, or ambient completeness. The named rows therefore supply the full obligation-closure surface. □

Theorem 464.48 (CauchyProduct Real observation budget handoff)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, any* RealObservationBudget$^{\uparrow}$ *consumer that reads the product through the* Real$^{\uparrow}$ *seal must first route the paired windows, dyadic product observation, product-radius ledger,* RegSeqRat$^{\uparrow}$ *classifier, and seal row through the finite budget packet*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B).$$

The rows W_B, D_B, R_B, S_B cover the product-side windows W_X, W_Y, dyadic observations Q_X, Q_Y, product observation P, radius ledger L, classifier row E, and Real$^\uparrow$ *seal handoff before the product consumer reads the seal.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real observation budget handoff`

Proof. Begin with the CauchyProduct obligation closure. It already binds the product route to the paired StreamName$^\uparrow$ windows, dyadic observations, product-radius ledger, RegSeqRat$^\uparrow$ classifier, Real$^\uparrow$ seal boundary, displayed transports, continuations, provenance, and local naming rows.

Route that surface through the observation budget. The readiness theorem Definition 611.18 requires carrier admission for $B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$ before a later consumer may use the finite windows, dyadic rows, regular handoff, or real seal.

Finally align the product rows with the budget rows. The StreamName handoff supplies the paired windows, ledger exhaustion and radius product regularity supply the product-radius and classifier route, and the Real seal boundary supplies the seal row. Thus the budget packet is read before the product seal is consumed. □

Theorem 464.49 (CauchyProduct observation budget RegSeq handoff)**.** *An accepted* CauchyProduct$^\uparrow$ *packet hands its product* RegSeqRat$^\uparrow$ *classifier to a* RealObservationBudget$^\uparrow$ *consumer only through the budget rows $E_B, W_B, D_B, R_B, H_B, C_B, P_B, N_B$. The classifier row E is reached from the paired source windows and the product observation by the CauchyProduct StreamName handoff, ledger exhaustion, radius product regularity, and the* RealObservationBudget$^\uparrow$ *regular-sequence handoff; no consumer may read E from a quotient stream, selected limit, host multiplication row, trichotomy row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket observation budget regseq handoff`

Proof. First build the product classifier route. The StreamName handoff places P downstream of $W_X, W_Y, Q_X, Q_Y, M_X, M_Y$, while ledger exhaustion and radius product regularity identify the finite radius ledger as the route from P to the classifier row E.

Next pass through the observation budget. By Definition 611.4, every regular-sequence handoff in an accepted budget packet is downstream of the precision request, finite windows, dyadic readbacks and radii, transports, continuation routes, provenance, and local naming row.

Combining the two routes gives the required handoff. The product classifier can be consumed only as the budgeted regular-sequence row R_B, carried by H_B, C_B, P_B, N_B, and all forbidden analytic objects remain outside both packets. □

Theorem 464.50 (CauchyProduct window-budget composition surface)**.** *For every accepted* CauchyProduct$^\uparrow$ *packet and every accepted* RealObservationBudget$^\uparrow$ *packet reading its product, the paired windows W_X, W_Y, radius rows M_X, M_Y, dyadic rows Q_X, Q_Y, product row P, ledger L, and budget rows $E_B, W_B, D_B, R_B, H_B, C_B, P_B, N_B$ form one finite composition surface from* StreamName$^\uparrow$ *through* DyadicRatCore$^\uparrow$ *to the* RegSeqRat$^\uparrow$ *classifier row E. The surface is obtained by the CauchyProduct StreamName handoff of Definition 464.20, the budget route exactness surface of Definition 464.12, and radius product regularity of Definition 464.32, and it exposes no quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket window budget composition surface`

Proof. First unfold the paired-window rows of the accepted product packet. The StreamName handoff of Definition 464.20 sends the two source names through W_X, W_Y to the dyadic-radius and observation rows M_X, M_Y, Q_X, Q_Y, so the first part of the surface is already finite.

Next route the same rows through the observation budget. The budget route exactness surface of Definition 464.12 identifies the consumer-visible rows $E_B, W_B, D_B, R_B, H_B, C_B, P_B, N_B$ with the finite precision, window, dyadic readback, regular-sequence, transport, continuation, provenance, and naming rows of the budget packet.

Finally compose with radius product regularity of Definition 464.32. The product row P and ledger L carry the paired dyadic observations to the classifier row E, so the combined route is a single finite surface from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$.

Every displayed coordinate belongs to one of the two packets, and both packets exclude unlisted analytic resources. Hence the composition surface contains no quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness witness. □

Theorem 464.51 (CauchyProduct budget Real seal compatibility)**.** *The window-budget composition surface of Definition 464.50 is exactly the route that a* $\mathsf{Real}^{\uparrow}$ *seal and a* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer may read. The* $\mathsf{Real}^{\uparrow}$ *side reads the classifier and seal rows supplied by the CauchyProduct Real seal boundary of Definition 464.21; the budget side reads the same classifier only through the Real observation budget handoff of Definition 464.48 and the observation-budget* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 464.49. No additional quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness witness is available at the join.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget real seal compatibility`

Proof. Begin at the Real seal boundary of Definition 464.21. It permits the product to reach a $\mathsf{Real}^{\uparrow}$ consumer only through the recorded $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E, the finite ledger, the displayed transports, continuations, provenance, and local naming evidence.

Now read the same route through the observation budget. The Real observation budget handoff of Definition 464.48 requires the budget packet before the seal is consumed, and the observation-budget $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 464.49 places the classifier row downstream of $E_B, W_B, D_B, R_B, H_B, C_B, P_B, N_B$.

The two readings therefore meet on the composition surface and nowhere else. The seal-facing route supplies the product classifier, while the budget-facing route supplies the permitted consumer access to that classifier.

Since the cited boundaries are exhaustive on both sides, there is no remaining coordinate for quotient streams, selected limits, host multiplication primitives, trichotomy rows, independent real carriers, or ambient completeness witnesses. □

Theorem 464.52 (CauchyProduct root product readiness package)**.** *The window-budget composition surface of Definition 464.50, the budget Real seal compatibility of Definition 464.51, and the budgeted product consumer boundary of Definition 464.53 supply the root product readiness package for downstream* $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{RealCompleteness}^{\uparrow}$*, and budgeted product consumers. The package consists exactly of the finite product rows* $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$ *together with the finite budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*, read through* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *evidence.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root product readiness package`

Proof. First collect the composition surface. It gives downstream readers the finite route from paired $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ observations to the $\mathsf{RegSeqRat}^{\uparrow}$ product classifier.

Next attach the Real seal compatibility. This identifies the same finite route as the only route a $\mathsf{Real}^{\uparrow}$ seal and a $\mathsf{RealObservationBudget}^{\uparrow}$ consumer may jointly read.

Finally add the budgeted product consumer boundary. That theorem lists the complete product and budget inventories visible to product consumers, including the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

The three surfaces therefore form the advertised readiness package. It is suitable for $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{RealCompleteness}^{\uparrow}$, and budgeted product consumers precisely because it exports a finite product-read interface and excludes quotient streams, selected limits, trichotomy, host multiplication primitives, independent real carriers, and ambient completeness witnesses. □

Theorem 464.53 (CauchyProduct budgeted product consumer boundary)**.** *Every product consumer that enters through an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet and exits through an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet reads exactly the union of the product rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The shared route exposes finite windows, dyadic observations, product-radius accounting, regular-rational classification, the $\mathsf{Real}^{\uparrow}$ *seal boundary, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exposes no quotient stream, host multiplication primitive, selected limit, trichotomy decision, independent real carrier, or ambient completeness coordinate.*

Proof. Start from the product side. CauchyProduct obligation closure, the StreamName handoff, the Real seal boundary, ledger exhaustion, and radius product regularity exhaust the product-facing rows before the seal is read.

Now add the observation-budget side. The budget readiness theorem, together with budget consumer exactness and non-escape, says that $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers read only $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$.

The consumer boundary is therefore the displayed union of rows. Since both packets list their complete inventories and both non-escape clauses exclude unlisted analytic objects, the joined consumer route has no coordinate for quotient streams, host multiplication, selected limits, trichotomy, independent real carriers, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget product consumer boundary`

Theorem 464.54 (CauchyProduct budget RegSeq classifier lock)**.** *Before the product row* E *of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is read as a* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier by a* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer, the finite rows* $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L$ *have already locked the route from the two source names to the product observation. The lock is the route supplied by the CauchyProduct StreamName handoff, the paired-window ledger classifier, the budget route exactness surface, and the budgeted product consumer boundary.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget regseq classifier lock`

Proof. The StreamName handoff fixes the source reads through W_X, W_Y and the radius rows M_X, M_Y before the dyadic rows Q_X, Q_Y and product observation P are visible.

The paired-window ledger classifier then routes P through L to the classifier row E. Budget route exactness and the budgeted product consumer boundary say that a consumer can read this row only through the displayed budget route, so E cannot be reached before the listed product rows have been exposed. □

Theorem 464.55 (CauchyProduct Real seal budget gate)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet that is consumed through a* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet factors through the finite budget rows* $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*. The gate is the Real observation budget handoff joined with the CauchyProduct budget Real seal compatibility and the budgeted product consumer boundary.*

Proof. First use the Real observation budget handoff. It requires the consumer to expose the precision request, window, dyadic, regular-sequence, seal, transport, continuation, provenance, and local naming rows of B.

The CauchyProduct budget Real seal compatibility identifies that budget route with the product classifier and seal route. The budgeted product consumer boundary exhausts the joined inventory, so every seal read through the budget passes through exactly $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal budget gate`

Theorem 464.56 (CauchyProduct window product nonescape)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the paired windows* W_X, W_Y *are the only route from the two source names to the product observation* P*. The route proceeds through* M_X, M_Y, Q_X, Q_Y *and the product ledger surface, and it excludes quotient streams, host multiplication, selected limits, trichotomy, independent real carriers, and ambient completeness.*

Proof. The CauchyProduct StreamName handoff sends each source name to its finite window and dyadic observation row. The paired-window ledger classifier then supplies the product observation route through the displayed dyadic and ledger rows.

Budget route exactness, the Real observation budget handoff, and the budgeted product consumer boundary exhaust all rows available to downstream readers. Since none of those rows is a quotient stream, host multiplication primitive, selected limit, trichotomy decision, independent real carrier, or ambient completeness theorem, there is no alternate route from the source names to P. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket window product nonescape`

Theorem 464.57 (CauchyProduct shared-window root admission)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet admits a root-facing product route only through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support rows* hsame*,* Cont*,* Pkg*, and* NameCert*. The admitted root surface is*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

It exposes no product before those paired finite windows and their dyadic observations have been read.

Proof. Begin with the CauchyProduct StreamName handoff. It places the product route downstream of the paired source-window rows W_X, W_Y and their modulus rows M_X, M_Y.

Next use the product-observation seal obligations and ledger exhaustion. These rows force the dyadic observations Q_X, Q_Y, the product observation P, and the product-radius ledger L to be read before the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E is available.

Finally pass through the Real multiplication seal handoff and the public consumer boundary. The rows H, C, L, N preserve, route, account for, and name the same packet, so a root-facing consumer receives the product only through the displayed finite window inventory. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared window root admission`

Theorem 464.58 (CauchyProduct root window RegSeqRat seal route)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *root route that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal in that order. The route is the finite inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and it exposes no product-facing seal before the paired windows and dyadic observations have been recorded.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window regseqrat seal route`

Proof. Begin with Definition 464.57. It admits a root-facing product route only after the paired window rows W_X, W_Y, source-radius rows M_X, M_Y, and dyadic observation rows Q_X, Q_Y have entered the product packet.

Next use the product-observation and ledger rows later in this chapter. The paired-window regularity exhaustion, root factor-source exhaustion, and root budget classifier coverage keep the product row P, classifier row E, and ledger row L downstream of the same finite window and dyadic observation route.

Finally apply the Real seal root surface and seal-ledger exactness. These rows place the $\mathsf{Real}^{\uparrow}$ seal after the $\mathsf{RegSeqRat}^{\uparrow}$ readback and preserve the same hsame, Cont, Pkg, and NameCert evidence, giving the stated order with no additional analytic coordinate. □

Theorem 464.59 (CauchyProduct root budget non-escape)**.** *An accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *root formation packet cannot export quotient streams, host real multiplication, ambient completeness, selected limits, trichotomy rows, independent real carriers, or unbounded selector rows. Every exported product route is confined to paired finite windows, dyadic observations, product-observation rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, the* $\mathsf{Real}^{\uparrow}$ *seal, displayed transport, displayed continuation, package provenance, local naming, and any explicitly displayed* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows.*

Proof. Start from the shared-window root admission route. It places the product behind the paired finite windows, dyadic observations, classifier row, seal row, and local support rows.

Use the later product-observation and budget rows to account for all consumers. The budgeted readback root boundary and budgeted product consumer boundary add only displayed $\mathsf{RealObservationBudget}^{\uparrow}$ rows, while paired-window regularity exhaustion and root budget classifier coverage keep every product read inside the same finite ledger.

The excluded objects have no coordinate in this inventory. Since quotient streams, host real multiplication, ambient completeness, selected limits, trichotomy rows, independent real carriers, and unbounded selector rows are absent from the packet and absent from the budget handoff, root formation cannot export them. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget nonescape`

Theorem 464.60 (CauchyProduct root selector-ledger exactness)**.** *Every selector or budget row used by a downstream* CauchyProduct$^{\uparrow}$ *product consumer is recorded in the local ledger before the consumer reads the* Real$^{\uparrow}$ *seal. The recorded route includes the paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations, product row,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, local* NameCert *evidence, and any displayed* RealObservationBudget$^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root selector ledger exactness`

Proof. First apply shared-window root admission. It fixes the product route as a finite sequence through paired windows, dyadic observations, product formation, classifier readback, transport, continuation, ledger, provenance, and local naming rows.

Next attach the downstream consumer. The budgeted readback root boundary, root seal-ledger exactness, and budgeted product consumer boundary require any budget-facing read to carry its displayed observation-budget rows with the same product ledger.

Finally read the seal. The Real seal root surface and Real seal multiplication non-escape rows leave the consumer no route around the local ledger, so each selector or budget row has already been recorded before the Real$^{\uparrow}$ seal is consumed. □

Theorem 464.61 (CauchyProduct budgeted readback root boundary)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *packet feeding an accepted* RealObservationBudget$^{\uparrow}$ *packet, every root-facing readback of the product factors through the product inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the budget inventory

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The readback reaches the RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *rows only after the paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows have been consumed.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budgeted readback root boundary`

Proof. First apply the shared-window root admission theorem. It binds the product route to the paired finite windows, dyadic observations, product observation, classifier row, Real seal, transport, continuation, ledger, provenance, and local naming rows.

Now enter the observation budget. The Real observation budget handoff and observation-budget RegSeq handoff require the budget rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$ before a downstream consumer may read the regular-sequence or real-facing product.

The budgeted product consumer boundary then identifies the complete readback interface. Since both inventories are finite packet rows, there is no coordinate for reading the product through a quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness witness. □

Theorem 464.62 (CauchyProduct Real seal root surface)**.** *The* Real$^{\uparrow}$ *seal surface of an accepted* CauchyProduct$^{\uparrow}$ *packet is root-visible only as the finite route composed from paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations,* RegSeqRat$^{\uparrow}$ *readback, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence. When the packet feeds an accepted* RealObservationBudget$^{\uparrow}$ *packet, the same route is read through the budgeted boundary of Definition 464.61.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal root unblock surface`
↪ variant `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal root surface`

Proof. Start with the Real multiplication seal handoff. It makes the real-facing product row depend on the RegSeqRat$^{\uparrow}$ classifier E rather than on host multiplication or a selected real representative.

Trace E back through the finite product route. The CauchyProduct StreamName handoff, product-observation seal obligations, radius ledger, and radius product regularity force the route through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L$.

Finally compose with the budgeted readback root boundary. The budget packet reads the same real seal through $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, while the support rows preserve the product route as hsame, Cont, Pkg, and NameCert data. Thus the root-visible Real seal is exactly the finite packet route. □

Theorem 464.63 (CauchyProduct RealAlgOrder product-observation export)**.** *Every* RealAlgOrder$^{\uparrow}$ *multiplication export that reads an accepted* CauchyProduct$^{\uparrow}$ *packet factors through the paired* StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *observations, the* RegSeqRat$^{\uparrow}$ *classifier, the* Real$^{\uparrow}$ *seal, the* RealObservationBudget$^{\uparrow}$ *rows when a budget packet serves the read, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The export reads multiplication only as the finite product-observation route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$$

when they are present. It exports no host multiplication, quotient stream, selected limit, trichotomy, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realalgorder product observation export`

Proof. First compose root-window admission with the budgeted readback boundary. The paired StreamName$^{\uparrow}$ windows and DyadicRatCore$^{\uparrow}$ observations must be read before the product row, classifier row, and seal row are visible to any downstream multiplication consumer.

Next apply the Real seal root surface. It keeps the real-facing product on the finite route through RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, displayed hsame transports, displayed Cont routes, Pkg provenance, and the local NameCert row.

Finally route the same surface through the observation-budget rows when the budget packet serves the read. Since the product and budget inventories contain no host multiplication primitive, quotient stream, selected limit, trichotomy branch, independent real carrier, or ambient completeness witness, the RealAlgOrder$^{\uparrow}$ export can read only the displayed product-observation route. □

Theorem 464.64 (CauchyProduct dyadic-radius ledger handoff)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *packet feeding an accepted* RealObservationBudget$^{\uparrow}$ *packet, the input-radius rows and the product-radius ledger rows are preserved when the product packet hands off to the observation-budget packet. The handoff carries*

$$M_X, M_Y, Q_X, Q_Y, L$$

through the product route

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and into the budget route

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

It preserves the displayed dyadic-radius accounting and exports no host multiplication, quotient stream, selected limit, trichotomy, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket dyadic radius ledger handoff`

Proof. Start with root-window admission. The input-radius rows and dyadic observations enter the product packet together with the paired finite windows, so the product route cannot reach P or E without first reading M_X, M_Y, Q_X, Q_Y.

Use radius budget conservation for the product side. It identifies L as the finite ledger that carries the input-radius and product-radius accounting through componentwise hsame and displayed Cont routes.

Now pass through the Real seal root surface and the budgeted readback boundary. The observation-budget packet reads the same classifier and seal route through its displayed rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, preserving the dyadic-radius ledger rather than replacing it by any excluded analytic object. □

Theorem 464.65 (CauchyProduct root-window factorization)**.** *Every accepted product consumer that enters through a* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through the root-window rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

before any $\mathsf{RealObservationBudget}^{\uparrow}$ *row is read. The factorization route first reads the two* $\mathsf{StreamName}^{\uparrow}$ *window schedules, then the paired dyadic observations and radius ledgers, then the product row* P*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*, the displayed* hsame *transports, the* Cont *routes, the ledger row* L*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row* N*. It exposes no quotient stream, selected limit, trichotomy, host multiplication primitive, independent real carrier, or ambient completeness coordinate before the budget packet is entered.*

Proof. Start with the StreamName handoff. It makes the product consumer read the paired finite windows W_X, W_Y, moduli M_X, M_Y, and dyadic observations Q_X, Q_Y as displayed packet rows rather than as ambient streams.

Next use paired-window closure and the product ledger. The pointwise product row P and classifier row E are reached only through the paired finite windows, radius accounting, and regular-rational route; the displayed hsame and Cont rows carry the same route without adding another analytic carrier.

Finally apply Definition 464.53. That boundary places every later observation-budget read downstream of the product inventory, so the root window has already factored through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$, before any budget row is available. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root window factorization`

Theorem 464.66 (CauchyProduct radius budget conservation)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet feeding an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, the paired input-radius rows and the product-radius row are conserved under componentwise* hsame *and displayed* Cont *routes through the budget packet. Thus the route from*

$$M_X, M_Y, Q_X, Q_Y, L$$

to the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$$

preserves the same finite radius ledger, the same product-radius witness, and the same regular-sequence classifier handoff; it cannot replace them by a selected limit, quotient-stream equality, host multiplication estimate, trichotomy branch, independent real carrier, or ambient completeness statement.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket radius budget conservation`

Proof. First read the radius ledger on the product side. Ledger exhaustion and radius product regularity identify the input radii, the dyadic product observation, and the product-radius row as the exact finite accounting surface for the $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

Now transport that surface componentwise. The displayed hsame rows preserve the left and right input-radius rows, while the displayed Cont routes preserve the product-radius route and package provenance inside the finite packet.

Pass the conserved ledger through the observation budget. The observation-budget RegSeq handoff reads the same classifier route as R_B together with H_B, C_B, P_B, N_B, and Definition 464.53 leaves no coordinate by which an analytic replacement could enter. □

Theorem 464.67 (CauchyProduct shared-tail budget consumer)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet be consumed by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and suppose the paired source windows are admitted through the shared threshold surface of chapter 560. Then every seal-facing product consumer factors through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, the product-radius budget rows, the shared RegularCauchyTailMeet threshold, and the* $\mathsf{Real}^{\uparrow}$ *seal rows before any product consumer may read the seal.*

The complete consumer route is the product inventory

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the budget inventory

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

It exposes no quotient stream, selected limit, host multiplication, trichotomy, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared tail budget consumer`

Proof. First compose the root-window factorization with the shared tail surface. The factorization forces every product read through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$, while chapter 560 supplies the common finite threshold used by the paired source windows.

Next apply radius budget conservation. The rows M_X, M_Y, Q_X, Q_Y, L remain the same finite radius ledger when the product route is transported through the observation-budget packet.

Now use the budgeted readback root boundary. It attaches $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$ to the same product route and makes the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and $\mathsf{Real}^{\uparrow}$ seal readable only after the paired windows, dyadic observations, budget rows, and shared threshold have been displayed.

Finally apply the sibling tail-meet threshold surface together with the budgeted product consumer boundary. The resulting route has no remaining coordinate for a quotient stream, selected limit, host multiplication, trichotomy, independent real carrier, or ambient completeness principle, so the seal consumer is exactly the shared-tail budget consumer described above. □

Theorem 464.68 (CauchyProduct tail-fusion budget route)**.** *Every accepted* CauchyProduct$^{\uparrow}$ *product consumer that also consumes* RegularCauchyTailFusion$^{\uparrow}$ *and* RegularCauchyTailBudget$^{\uparrow}$ *factors through the paired* StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *observations, the* RegSeqRat$^{\uparrow}$ *readback, the fusion shared-threshold row, and the tail-budget row before the* Real$^{\uparrow}$ *seal is read. The root-visible route is the product inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the observation-budget inventory

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$$

and the displayed tail-fusion and tail-budget rows supplied by the sibling certificates. Its support is carried only by hsame*,* Cont*,* Pkg*, and* NameCert *rows; it exports no quotient stream, host multiplication, selected limit, trichotomy row, independent real carrier, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket tail fusion budget route`

Proof. First use shared-window root admission. The accepted product consumer must read the paired StreamName$^{\uparrow}$ windows and DyadicRatCore$^{\uparrow}$ observations before the product row, classifier row, ledger row, or seal row can be visible.

Next attach the tail-fusion surface. The RegularCauchyTailFusion$^{\uparrow}$ consumer contributes its shared-threshold row as a displayed finite route over the same source windows, so the product read cannot fork into an independent stream or carrier.

Finally compose with Definition 464.67. The tail-budget packet reads the same product inventory and observation-budget inventory, while hsame, Cont, Pkg, and NameCert preserve and name the route. The listed excluded objects have no coordinate in any of the composed packets. □

Theorem 464.69 (CauchyProduct root budget-fusion non-escape)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *product consumer served through* RegularCauchyTailFusion$^{\uparrow}$ *and* RegularCauchyTailBudget$^{\uparrow}$*, the root-visible budget-fusion route is exhausted by the paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations,* RegSeqRat$^{\uparrow}$ *readback, fusion shared-threshold row, tail-budget row,* Real$^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert*. No consumer of this route can recover a quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness coordinate.*

Proof. Begin with Definition 464.68. It gives the complete product, observation-budget, tail-fusion, and tail-budget inventory seen by the root-facing consumer.

Now apply radius budget conservation. The input-radius rows, dyadic observations, product-radius ledger, and regular readback row remain the same finite accounting surface when the route enters the tail-budget consumer.

The non-escape conclusion is then coordinate exhaustion. The composed packets expose only the displayed windows, observations, readback, threshold, budget, seal, transport, route, provenance, and local naming rows, so none of the excluded analytic objects can be read from the root-visible route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget fusion nonescape`

Theorem 464.70 (CauchyProduct RealAlgOrder root handoff)**.** *Every* RealAlgOrder$^{\uparrow}$ *product consumer that reads an accepted* CauchyProduct$^{\uparrow}$ *packet through* RegularCauchyTailFusion$^{\uparrow}$ *and* RegularCauchyTailBudget$^{\uparrow}$ *receives only the root-visible budget-fusion route of Definition 464.68. Thus the RealAlgOrder multiplication observation is consumed through paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *observations,* RegSeqRat$^{\uparrow}$ *readback, the fusion shared-threshold row, the tail-budget row, the* Real$^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert*, with no host multiplication, quotient stream, selected limit, trichotomy, independent real carrier, or ambient completeness coordinate exported.*

Proof. First apply Definition 464.63. The RealAlgOrder$^{\uparrow}$ multiplication export reads multiplication only as the finite product-observation route, with observation-budget rows when the budget packet serves the read.

Next compose the export with Definition 464.68. This inserts the shared tail-fusion threshold and tail-budget rows before the seal-facing RealAlgOrder read without adding a second multiplication carrier.

Finally use Definition 464.69. Since the composed route has no coordinate for host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness, the RealAlgOrder handoff is exactly the displayed root-visible budget-fusion route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realalgorder root handoff`

Theorem 464.71 (CauchyProduct RegSeq seal non-escape)**.** *A* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *seal consumer that is reached from an accepted* CauchyProduct$^{\uparrow}$ *packet and an accepted* RealObservationBudget$^{\uparrow}$ *packet cannot replace the finite product classifier by host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness. The only available seal-facing route is the product inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the observation-budget inventory

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket regseq seal nonescape`

Proof. Begin at the product seal. The Real multiplication seal handoff reads the product classifier E through the StreamName windows, dyadic observations, radius ledger, displayed transports, continuations, provenance, and local NameCert row.

Next enter the observation budget. The Real observation budget handoff and the observation-budget RegSeq handoff make the budget consumer read the classifier only as a finite budget row, with the same transports, routes, provenance, and local naming evidence.

The budgeted product consumer boundary then seals the route. Since the displayed product and budget inventories are complete, there is no remaining packet coordinate for host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, or ambient completeness. □

Theorem 464.72 (CauchyProduct budgeted product associative route)**.** *Let two accepted* CauchyProduct$^{\uparrow}$ *packets feed a common accepted* RealObservationBudget$^{\uparrow}$ *packet, and suppose the displayed source windows, dyadic observations, product-radius ledgers,* RegSeqRat$^{\uparrow}$ *classifier rows,* Real$^{\uparrow}$ *seal rows, budget rows, transports, routes, provenance, and local naming rows are the same:*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

Then the finite seal-facing route seen by the budgeted product consumer is the same whether the product observation is grouped as first-left-then-right or first-right-then-left. The comparison is a route-composition equality inside the displayed row inventory; it does not introduce quotient streams, host multiplication primitives, selected limits, trichotomy rows, independent real carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget product associative route`

Proof. Begin with the common product inventory. The CauchyProduct StreamName handoff fixes the paired source-window reads, while ledger exhaustion and radius product regularity fix the product-radius and classifier route from the same displayed dyadic observations.

Pass both groupings through the Real-facing rows. The Real multiplication seal handoff and the Real observation budget handoff read the same $\mathsf{Real}^{\uparrow}$ seal rows and the same budget packet before any consumer-facing seal is exposed.

Finally compare the two composites. The observation budget RegSeq handoff and Definition 464.53 leave only the displayed product rows and budget rows available to the consumer. Since both groupings use exactly those rows with the same transports, routes, provenance, and local names, their seal-facing finite routes coincide within the packet inventory. □

Theorem 464.73 (CauchyProduct budgeted product symmetric route)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet feeding an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, swapping the two source-window families swaps only*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y$$

between the two product inputs. The budget rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$*, classifier row* E*, Real seal handoff, displayed transports, routes, provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *row are preserved as the same finite consumer surface. No symmetric route exports quotient streams, host multiplication primitives, selected limits, trichotomy rows, independent real carriers, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget product symmetric route`

Proof. Read the two source families through the CauchyProduct StreamName handoff. Swapping the sources exchanges only the displayed left and right window, modulus, and dyadic observation rows.

The product-radius ledger and radius product regularity are then replayed with the exchanged source rows. They still produce the same classifier-facing product ledger and route to E, because no additional analytic coordinate is available in the packet.

Route the result through the Real multiplication seal handoff, the Real observation budget handoff, and the observation budget RegSeq handoff. The budget packet, seal handoff, transports, continuations, provenance, and local naming rows are unchanged, so Definition 464.53 gives the same consumer boundary with only the two source families exchanged. □

Theorem 464.74 (CauchyProduct budgeted product unit route)**.** *If one displayed source of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is the regular-Cauchy unit source, then inserting that unit source contributes only its unit source rows to the finite packet. The budgeted product consumer still reads the same product boundary*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the same observation-budget boundary

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B,$$

with the unit source occupying the corresponding source-window, modulus, and dyadic observation coordinates. The route does not export quotient streams, host multiplication primitives, selected limits, trichotomy rows, independent real carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget product unit route`

Proof. Insert the displayed unit source as one regular-Cauchy input. Its contribution is read through the same CauchyProduct StreamName handoff as any other source: one window family, one modulus row, and one dyadic observation family.

Ledger exhaustion and radius product regularity then place the unit rows in the product-radius ledger. The route still reaches the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and $\mathsf{Real}^{\uparrow}$ seal through the existing product packet rather than through a host multiplication primitive or selected limit.

The Real observation budget handoff and observation budget RegSeq handoff carry the result into the same budget packet. By Definition 464.53, the consumer boundary is unchanged except for the displayed unit rows occupying their source slots, and all excluded analytic objects remain outside the finite inventory. □

Theorem 464.75 (CauchyProduct paired-window regularity exhaustion)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose product read is consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *route factors through exactly the finite rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

The factorization first binds the paired source windows W_X, W_Y*, the source-radius rows* M_X, M_Y*, and the dyadic observations* Q_X, Q_Y*; then it reads the product row* P *and the product-radius route that supplies the classifier row* E*; finally it transports the sealed product through* H, C, L, N *to the consumer boundary. The route excludes host multiplication, quotient streams, selected limits, trichotomy, independent real carriers, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket paired window regularity exhaustion`

Proof. Start with paired-window closure. It forces any accepted product read to pass through W_X, W_Y before exposing M_X, M_Y, Q_X, Q_Y, the product row P, and the downstream classifier route.

Next use radius product regularity. The product-radius ledger is the only route from the displayed dyadic product observation to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier E, so the regularity witness is part of the finite packet rather than an external multiplication or completeness claim.

Now pass to the seal-facing boundary. The Real multiplication seal handoff and the RealCompleteness diagonal seal route make every $\mathsf{Real}^{\uparrow}$ or diagonal consumer read only the displayed classifier, transports, continuation rows, ledger, package provenance, and local $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ evidence.

Finally apply ledger exhaustion. Since the named rows exhaust the accepted packet, there is no remaining coordinate for quotient streams, selected limits, trichotomy, independent real carriers, ambient completeness, or a host multiplication primitive. □

Theorem 464.76 (CauchyProduct root factor-source exhaustion)**.** *Every accepted* CauchyProduct$^{\uparrow}$ *route that exposes the product row P to a downstream* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or* RealObservationBudget$^{\uparrow}$ *consumer first exhausts the two factor sources through*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y.$$

The packet therefore reads the left and right StreamName$^{\uparrow}$ *windows, the two source-radius rows, and the two* DyadicRatCore$^{\uparrow}$ *observation rows before P is visible. The product route has no source-facing coordinate for quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root factor source exhaustion`

Proof. Start from the paired-window regularity exhaustion theorem. It orders every accepted product read by the displayed rows $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$, with the source windows and dyadic observations preceding the product row.

Next apply the root-window factorization and budgeted readback boundary. These keep the same source rows when the product is read through a budget packet, so P cannot be inspected before the two factor windows, radius rows, and dyadic observations have been consumed.

Finally pass through the Real seal root surface. The seal handoff reads the product only after the classifier and transport rows have inherited the exhausted source inventory, leaving no separate analytic object from which P could be imported. □

Theorem 464.77 (CauchyProduct root budget classifier coverage)**.** *The product observation P, the product-radius ledger L, and the* RegSeqRat$^{\uparrow}$ *classifier row E of an accepted* CauchyProduct$^{\uparrow}$ *packet are covered only by the dyadic-radius budget route through* DyadicRatCore$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$*. More explicitly, any consumer that reaches one of P, L, E must read the finite product inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and, when an observation budget is present, the finite budget inventory

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

No independent classifier, host multiplication estimate, quotient-stream comparison, selected limit, trichotomy row, independent real carrier, or ambient completeness witness covers P, L, E.

Proof. First use radius budget conservation. It identifies the route from the two dyadic source observations to the product-radius ledger and then to the regular-sequence classifier, so P, L, E stay on the same dyadic-radius accounting path.

Next combine this with paired-window regularity exhaustion. The product row P is reached only after the paired windows, source-radius rows, and dyadic observations have been read, and the classifier row E is supplied by the product-radius route rather than by an external real carrier.

When the read is budgeted, apply the budgeted readback root boundary and the budgeted product consumer boundary. They add exactly the displayed RealObservationBudget$^{\uparrow}$ rows and do not add a second route for P, L, E. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget classifier coverage`

Theorem 464.78 (CauchyProduct root seal-ledger exactness)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealObservationBudget}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reads exactly the joined product and budget ledger rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and, when a budget packet is present,

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The seal consumer cannot import quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, ambient completeness, or any row outside this joined ledger.

Proof. Begin with the Real seal root surface. It routes the real-facing product through paired windows, dyadic observations, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, displayed transports, displayed continuations, package provenance, and local $\mathsf{NameCert}$ evidence.

Next apply root factor-source exhaustion and root budget classifier coverage. The first fixes the factor-source rows before P can be read; the second fixes the only dyadic-radius route covering P, L, E and, when present, the exact budget rows.

Finally use the budgeted product consumer boundary. Since the product ledger and the observation-budget ledger are already the complete finite inventories, a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealObservationBudget}^{\uparrow}$ seal consumer has no remaining row from which to import quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root seal ledger exactness`

Theorem 464.79 (CauchyProduct budgeted bilinear diagonal)**.** *Let two accepted* $\mathsf{RegSeqRat}^{\uparrow}$*-backed* $\mathsf{CauchyProduct}^{\uparrow}$ *factors share a* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. Then the diagonal product window is determined by the existing paired-window and budgeted product routes over*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the observation-budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The resulting diagonal read is bilinear only as a finite product-observation certificate: it exposes the two regular source windows, the dyadic radius rows, the pointwise product row, the shared budget ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and it exports no host multiplication primitive, quotient stream equality, selected limit, independent real carrier, or ambient completeness.*

Proof. Start with Definition 464.75. It forces the product read through the two source windows, source-radius rows, dyadic observations, product row, classifier row, transports, continuations, ledger, package provenance, and local naming row.

Next apply the budgeted product route supplied by Definition 464.53. The shared observation budget contributes exactly $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$, so the product read and the budget read meet on the same diagonal request rather than on a host arithmetic operation.

Use the radius budget conservation route of Definition 464.66. The dyadic radius ledger records the bound consumed by the product window, so the diagonal product observation is determined by the displayed source windows and the displayed budget rows.

Finally use the Real multiplication seal handoff. The seal receives only the finite product and budget rows already listed, giving the stated bilinear diagonal certificate without quotient streams, selected limits, independent real carriers, host multiplication, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budgeted bilinear diagonal`

Theorem 464.80 (CauchyProduct shared-radius product coverage)**.** *Every displayed product-observation request consumed by the* RealAlgOrder$^{\uparrow}$ *multiplication export is covered by paired-window regularity plus radius-budget conservation. Concretely, a multiplication consumer can read a* CauchyProduct$^{\uparrow}$ *product observation only through the two* RegularCauchyName$^{\uparrow}$ *source rows, their shared* StreamName$^{\uparrow}$ *windows, the displayed* DyadicRatCore$^{\uparrow}$ *radius observations, the* RealObservationBudget$^{\uparrow}$ *budget rows, the* RegSeqRat$^{\uparrow}$ *classifier, the* Real$^{\uparrow}$ *seal handoff, and the visible* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Apply Definition 464.75. Any product read reaching a regular-sequence or real-facing consumer already passes through the paired windows and the dyadic product-radius rows.

Apply Definition 464.79. The shared observation budget aligns the product request with the same finite diagonal window and the same radius ledger, so a multiplication export receives a covered product observation rather than an implicit arithmetic operation.

Now use the Real seal handoff and the RegSeq seal non-escape route recorded earlier in this chapter. These rows keep the public consumer inside the product packet, budget packet, classifier row, seal row, transports, continuations, package provenance, and local naming certificate.

Thus every product-observation request displayed to the RealAlgOrder$^{\uparrow}$ multiplication export is accounted for by paired-window regularity and radius-budget conservation, with no hidden multiplication or ambient completeness route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared radius product coverage`

Theorem 464.81 (CauchyProduct Real seal multiplication non-escape)**.** *The public multiplication consumer of a* CauchyProduct$^{\uparrow}$ *packet receives only the finite* CauchyProduct$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows displayed by the product and budget certificates. It receives no host multiplication primitive, quotient stream equality, selected limit, trichotomy row, independent real carrier, ambient completeness principle, or unrecorded radius witness.*

Proof. Begin with Definition 464.80. It says that every multiplication-facing product observation is covered by the paired source windows, dyadic radius rows, shared observation budget, classifier route, and real seal handoff.

Use Definition 464.53 to identify the public budget boundary. The consumer can inspect the product packet and the observation-budget packet, but only through their displayed rows and the recorded transport, continuation, provenance, and local naming routes.

Finally apply the paired-window exhaustion theorem and the RegSeq seal non-escape route. They leave no coordinate for host multiplication, quotient stream equality, selected limits, trichotomy, independent real carriers, ambient completeness, or unrecorded radius witnesses, so the multiplication seal is exactly the finite public certificate stated above. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal multiplication nonescape`

Theorem 464.82 (CauchyProduct root source admission)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, the source rows*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y$$

admit a finite BHist *root surface over* RegularCauchyName$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *The surface is generated by the two regular-Cauchy source rows, the paired finite-window schedules, the paired radius rows, and the dyadic observation rows,*

and it exposes no quotient stream, selected limit, host multiplication primitive, ambient completeness row, trichotomy row, or independent real carrier.

Proof. Start with the packet source of Definition 464.1. Its displayed coordinates put $X, Y, W_X, W_Y, M_X, M_Y, Q_X,$ inside the same finite BHist packet as the product row, classifier row, transports, continuations, ledger, package provenance, and local naming row.

The sibling carriers supply the admissible readings. Use chapter 126 for the finite window schedules, chapter 291 for the dyadic observations and radius rows, chapter 301 for the classifier route, and chapter 38 for the real seal boundary.

Finally apply Definition 464.3, Definition 464.4, and Definition 464.5. These three obligations bind the source rows to finite windows, radius accounting, and the real-seal route, leaving no coordinate for any excluded analytic object. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root source admission`

Theorem 464.83 (CauchyProduct root classifier stability)**.** *Componentwise classifier transport of the two source rows, paired windows, radius rows, dyadic observations, product observation, and* RegSeqRat$^{\uparrow}$ *classifier preserves the same* CauchyProduct$^{\uparrow}$ *root packet*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

The transported packet is again routed only through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows over* RegularCauchyName$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$*; it admits no quotient stream and no host multiplication primitive.*

Proof. First read the source and window rows through Definition 464.82. The theorem fixes the root source surface as a finite BHist packet over the listed sibling carriers.

Next transport the classifier componentwise. The StreamName and DyadicRatCore rows from chapter 126 and chapter 291 preserve the paired windows, radius rows, dyadic observations, and product observation through the displayed hsame and Cont data.

Now read the classifier and seal. The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ rows from chapter 301 and chapter 38, together with Definition 464.75, Definition 464.66, and Definition 464.71, keep the transported packet on the same finite route. Since the route is exhausted by the displayed packet, quotient streams and host multiplication primitives have no source row. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root classifier stability`

Theorem 464.84 (CauchyProduct root product-row determinacy)**.** *For any two accepted* CauchyProduct$^{\uparrow}$ *root packets over the same displayed rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, H, C, L, N,$$

the product row P and classifier-facing row E are determined by the paired finite-window route and the dyadic-observation route before the RegSeqRat$^{\uparrow}$ *readback and the* Real$^{\uparrow}$ *seal are consumed. No host multiplication primitive, quotient stream, selected limit, trichotomy branch, independent real carrier, or ambient completeness coordinate can change P or E once those finite rows are fixed.*

Proof. First read the common source surface through Definition 464.82. The accepted packets expose the paired windows, radius rows, and dyadic observations as one finite BHist route over StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Next apply Definition 464.75. It forces every product read toward a regular-rational or real-facing consumer through

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

Thus the rows preceding P and E already determine the multiplication-facing packet route.

Finally use Definition 464.83 together with the StreamName, DyadicRatCore, RegSeqRat, and Real carrier chapters chapter 126, chapter 291, chapter 301, and chapter 38. These routes preserve the same finite packet through componentwise transport, so the later $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal can only consume the determined product and classifier-facing rows. The excluded analytic objects have no coordinate in the packet and cannot alter them. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root product row determinacy`

Theorem 464.85 (CauchyProduct root phase handoff)**.** *Every downstream L10 consumer of the* $\mathsf{CauchyProduct}^{\uparrow}$ *root surface factors through the displayed paired windows, dyadic product route, radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, and local* NameCert *rows. Equivalently, the consumer boundary is the finite route through*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and, when a budget packet is present, through the displayed budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

No downstream consumer receives a quotient stream, selected limit, host multiplication primitive, unrecorded radius witness, trichotomy branch, independent real carrier, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root phase handoff`

Proof. Begin with the root packet. The source admission and classifier stability theorems keep $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$ as one finite BHist surface over the sibling carriers named in chapter 126, chapter 291, chapter 301, and chapter 38.

Next factor the product-facing consumer. Apply Definition 464.57, Definition 464.61, Definition 464.62, and Definition 464.65; these route the consumer through paired windows, dyadic observations, the product row, the classifier row, and the seal boundary.

Finally pass through the public multiplication surface when it is requested. The diagonal, coverage, and non-escape results Definition 464.79, Definition 464.80, and Definition 464.81 identify the exact downstream boundary. Since all routes are already finite packet rows, no excluded analytic object can be handed to an L10 consumer. □

Theorem 464.86 (CauchyProduct product-window consumer handoff)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *product-window consumer factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *multiplication boundary before exporting any product read. Its consumer-facing route is the finite surface*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the displayed observation-budget rows when a budget packet serves the read. It exports no product coordinate outside those rows, and no host multiplication, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket product window consumer handoff`

Proof. Start with the paired product windows. The shared-window root admission and root-window factorization results place every product read behind $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$, so the consumer first receives the two finite $\mathsf{StreamName}^{\uparrow}$ schedules and the dyadic tolerance observations.

Next compose the readback route. The budgeted readback boundary and radius budget conservation carry P, L through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E, while the structural rows H, C, N preserve the same packet under hsame, Cont, and local NameCert replay.

Finally pass to the order-facing export. The Real seal root surface, the RealAlgOrder product-observation export, and the Real seal multiplication non-escape theorem route the same packet through the $\mathsf{Real}^{\uparrow}$ seal and the $\mathsf{RealAlgOrder}^{\uparrow}$ multiplication boundary. Since every step projects or composes rows already present in the displayed obligations, the exported product read has exactly the stated finite surface and no excluded analytic coordinate. □

Theorem 464.87 (CauchyProduct stream seal obligation)**.** *Every* $\mathsf{CauchyProduct}^{\uparrow}$ *root consumer that reads a product seal must first read the paired* $\mathsf{StreamName}^{\uparrow}$ *windows*

$$W_X, W_Y$$

together with the source-radius rows M_X, M_Y*, the dyadic observation rows* Q_X, Q_Y*, the product row* P*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*, and the* $\mathsf{Real}^{\uparrow}$ *seal route. The obligation binds the stream windows to the regular-rational and real-seal reads through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it exports no host multiplication, quotient stream, selected limit, trichotomy row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket stream seal obligation`

Proof. Begin with Definition 464.82. It places the two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows, their $\mathsf{StreamName}^{\uparrow}$ windows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ observations inside the same finite root surface.

Next apply Definition 464.75. The theorem forces the product read through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H$ before any regular-rational or real-facing consumer can inspect the product.

Finally use the sibling seal boundaries from chapter 301 and chapter 38. These chapters name the classifier and seal rows reached by the product packet, while chapter 126 and chapter 291 name the upstream finite-window and dyadic inputs. The displayed packet has no row for any excluded analytic object. □

Theorem 464.88 (CauchyProduct real classifier obligation)**.** *The product observation* P *of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reaches the real-facing classifier only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*. The classifier route is determined by the paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, product-radius ledger, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and local naming row; it is not a host multiplication primitive, quotient-stream comparison, selected limit, trichotomy branch, independent real carrier, or ambient completeness statement.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real classifier obligation`

Proof. Start with the product observation row P. By Definition 464.83, transporting the paired source rows, windows, radius rows, dyadic observations, product observation, and classifier preserves the same finite root packet.

Now read the regular-rational classifier. The $\mathsf{RegSeqRat}^{\uparrow}$ boundary of chapter 301 is reached only after the displayed window and dyadic rows have been consumed, and the radius conservation result Definition 464.66 keeps the ledger attached to the same packet.

The real-facing read follows through the $\mathsf{Real}^{\uparrow}$ seal boundary, not through an external multiplication or completeness principle. Since the carrier exposes only the displayed classifier route, the excluded analytic objects cannot serve as alternative classifier evidence. □

Theorem 464.89 (CauchyProduct ledger obligation)**.** *The* $\mathsf{CauchyProduct}^{\uparrow}$ *root ledger binds the radius rows, product row, transport rows, continuation rows, package provenance, and local NameCert row as one finite accounting surface:*

$$M_X, M_Y, Q_X, Q_Y, P, H, C, L, N.$$

Every product-seal read must preserve this ledger when it passes to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, *or* $\mathsf{RealCompleteness}^{\uparrow}$ *consumers. The ledger contains no host multiplication estimate, quotient-stream equality, selected limit, trichotomy witness, independent real carrier, or ambient completeness coordinate.*

Proof. First apply Definition 464.66. It identifies the input radii, dyadic product observations, product-radius witness, and classifier handoff as one conserved finite ledger.

Next combine the ledger with Definition 464.80. The multiplication-facing consumer can read the product only through the paired source windows, dyadic radius observations, budget rows, classifier row, real-seal handoff, and visible structural rows.

Finally route the same ledger to the completion side through chapter 450. That chapter may consume the finite product-observation surface, while the ledger obligation keeps radius, product, transport, Cont, Pkg, and $\mathsf{NameCert}$ rows displayed and leaves all excluded analytic objects outside the packet. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket ledger obligation`

Theorem 464.90 (CauchyProduct root observation-budget totality)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet feed an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. The observation-budget rows are total for the product read precisely on the finite route where every displayed product read carries the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, the* $\mathsf{Real}^{\uparrow}$ *seal, the componentwise* hsame *transports, the displayed* Cont *replays, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows:*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, S, H, C, L, N.$$

Equivalently as a certificate condition internal to the packet, a budget row may be read by a $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer only when the product ledger of Definition 464.89 and the budgeted readback boundary of Definition 464.61 display the same product route. The budget contributes no host multiplication, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root observation budget totality`

Proof. First read the product side. The stream seal obligation places the paired $\mathsf{StreamName}^{\uparrow}$ windows, dyadic observations, regular-rational classifier, real seal, transport, replay, provenance, and local naming rows on one displayed product route.

Next read the accounting side. The ledger obligation keeps the radius rows, product row, classifier row, transport rows, continuation rows, provenance, and local NameCert row as one finite surface, so a budget-facing read cannot detach one row and still claim to read the product packet.

Now pass to the budget packet. The budgeted readback root boundary says that a $\mathsf{RealObservationBudget}^{\uparrow}$ consumer reads the same product route through its displayed budget rows. It does not add a second product witness or a hidden real carrier.

Finally use the real classifier obligation. Since the classifier and seal are reached only through the finite product route, totality of the budget rows is exactly coverage of every displayed product read by those same rows. The excluded analytic objects have no coordinate in either packet. □

Theorem 464.91 (CauchyProduct downstream unblock)**.** *Downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *and* $\mathsf{RealCompleteness}^{\uparrow}$ *consumers of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet receive only the finite product-observation surface*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

When an observation-budget packet is present, they may also read the displayed budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

They receive no host multiplication, quotient stream, selected limit, trichotomy row, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket downstream unblock`

Proof. Begin with Definition 464.85. It states that every downstream L10 consumer factors through the paired windows, dyadic product route, radius ledger, regular-rational readback, real seal row, and local naming rows.

For the order-facing side, apply Definition 464.80 and Definition 464.81. The $\mathsf{RealAlgOrder}^{\uparrow}$ consumer sees a covered finite product observation, not a host multiplication primitive or quotient-stream comparison.

For the completion-facing side, read chapter 450 through the same product surface. The completion route may consume the finite product rows and, when present, the observation-budget rows named by Definition 464.61; it has no coordinate for selected limits, trichotomy, independent real carriers, or ambient completeness beyond the displayed packet. □

Corollary 464.92 (CauchyProduct shared-window root strict local obstruction)**.** *Let* $Z = (X, Y, W_X, W_Y, M_X, M_Y$ *be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *root formation packet. Suppose a displayed failure, refusal, separation, or countermodel row in the root-unblock obligations is cited against a named converse or reuse route for the shared-window root admission surface of Definition 464.57. Then the cited row is a strict local obstruction. It blocks only the named route whose consumer-facing read tries to omit, replace, detach, or bypass one of the displayed root coordinates*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

or one of the displayed support rows hsame, Cont, Pkg, NameCert, *together with any explicitly displayed* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows serving the same read. Every permitted consumer-facing use factors through those rows and rejects unlisted coordinates. The same cited row cannot be reinterpreted as a global impossibility statement for* $\mathsf{CauchyProduct}^{\uparrow}$*, as a host theorem about multiplication or completion, as hidden external validation, as a quotient-stream argument, as a selected-limit argument, as a trichotomy row, as an independent real carrier, as an ambient completeness principle, or as an unbounded selector route.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared window root strict obstruction`

Proof. First apply Definition 464.57. It admits a root-facing product route only through the paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{Real}^{\uparrow}$ seal, and the support rows hsame, Cont, Pkg, and $\mathsf{NameCert}$. Thus any consumer-facing use of a cited obstruction must first be located on the finite admitted root surface

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

In particular, a route that still factors through these displayed rows is not rejected by the corollary; only a named converse or reuse request that leaves the displayed surface has a challenged coordinate.

Next apply Definition 464.59. It states that every exported product route is confined to paired finite windows, dyadic observations, product-observation rows, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier, the $\mathsf{Real}^{\uparrow}$ seal, displayed transport, displayed continuation, package provenance, local naming, and any explicitly displayed $\mathsf{RealObservationBudget}^{\uparrow}$ rows. It also excludes quotient streams, host real multiplication, ambient completeness, selected limits, trichotomy rows, independent real carriers, and unbounded selector rows from the exported root formation packet.

Now inspect the cited failure, refusal, separation, or countermodel row. If the named converse or reuse route tries to omit the paired windows, replace the dyadic observations, detach the product row from the classifier and seal, bypass H, C, L, N, or reuse the support rows away from hsame, Cont, Pkg, and $\mathsf{NameCert}$, the cited row blocks exactly that local request. If the same row is read instead as a global impossibility, a host theorem, hidden external validation, a quotient-stream route, a selected-limit route, a trichotomy route, an independent real carrier, an ambient completeness principle, or an unbounded selector route, then Definition 464.59 supplies no coordinate through which that reading can be exported.

Therefore the obstruction has exactly the force of a strict local block against the named converse or reuse route. It factors every permitted consumer-facing use through the displayed shared-window root inventory and rejects every unlisted coordinate without asserting any global, host-level, or hidden validation claim. □

Theorem 464.93 (CauchyProduct observation-budget triangle)**.** *Any accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *product observation consumed through a* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet factors through the triangle whose sides are the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the product and budget rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The source side is the admission route of Definition 464.82; the classifier side is the stability route of Definition 464.83; and the budget side is the finite readback boundary of Definition 464.61.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket observation budget triangle`

Proof. First read the product packet through Definition 464.82. The two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows are visible only through the paired $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{DyadicRatCore}^{\uparrow}$ observation rows, so the product row P has no source-facing route outside the displayed finite window surface.

Next transport that surface through Definition 464.83. The classifier row E, the $\mathsf{Real}^{\uparrow}$ seal row, and the displayed hsame, Cont, Pkg, and NameCert rows preserve the same finite product packet rather than replacing it by a quotient stream or host product.

Finally enter the observation budget through Definition 464.61. The budget rows $E_B, W_B, D_B, R_B, S_B, H_B, C_B$, read the same product packet, so the accepted observation is routed through the stated triangle of source windows, dyadic observations, classifier and seal rows, and budget rows. □

Theorem 464.94 (CauchyProduct RealAlgOrder Completion handoff)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *multiplication consumer and every* $\mathsf{RealCompleteness}^{\uparrow}$ *completion-facing consumer that reads an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet receives the same finite product-budget route. The shared route is the phase handoff of Definition 464.85 with the ledger obligation of Definition 464.89, read by the downstream chapters chapter 766 and chapter 450.*

Proof. Begin with Definition 464.85. It places every downstream L10 product consumer behind the paired source windows, dyadic observations, product row, classifier row, real seal row, displayed transports, continuations, package provenance, and local naming evidence.

Now apply Definition 464.89. The ledger route keeps the radius rows, product-radius accounting, and product classifier visible as finite packet data, so the handoff does not create a second multiplication surface for $\mathsf{RealAlgOrder}^{\uparrow}$ or a selected limit surface for $\mathsf{RealCompleteness}^{\uparrow}$.

The two downstream chapters then consume the same row inventory. The $\mathsf{RealAlgOrder}^{\uparrow}$ chapter may use the packet as its multiplication-facing product route, while the $\mathsf{RealCompleteness}^{\uparrow}$ chapter may use the same route as a completion-facing finite seal input; neither receives any coordinate beyond the displayed product and budget rows. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realalgorder completion handoff`

Theorem 464.95 (CauchyProduct finite product-budget exhaustion)**.** *The union of the product rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$$

exhausts the downstream L10 consumer surface of an accepted $\mathsf{CauchyProduct}^{\uparrow}$ *packet consumed through* $\mathsf{RealObservationBudget}^{\uparrow}$*. That surface excludes quotient streams, selected limits, host multiplication primitives, trichotomy rows, independent real carriers, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket finite product budget exhaustion`

Proof. Read the product side first. The root source admission, classifier stability, and phase handoff theorems make every product-facing consumer pass through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$ before any downstream real-facing read is available.

Read the budget side next. The observation-budget boundary supplies exactly $E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B$ as the finite consumer rows through which a $\mathsf{RealObservationBudget}^{\uparrow}$ packet may inspect the product.

Combining the two inventories leaves no unused coordinate in the L10 consumer surface. Since quotient streams, selected limits, host multiplication primitives, trichotomy rows, independent real carriers, and ambient completeness are not rows of either packet, they cannot be exported by the accepted product-budget route. □

Theorem 464.96 (CauchyProduct budget-triangle Real seal exhaustion)**.** *Any Real-facing consumer that reads an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet through the observation-budget triangle reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic observations, product row, classifier row, product ledger, and* $\mathsf{RealObservationBudget}^{\uparrow}$ *rows have all been displayed. The exhausted route is the finite union*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

It exports no quotient stream, selected limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget triangle real seal exhaustion`

Proof. First compose the triangle of Definition 464.93. It routes the product read through paired source windows, dyadic observations, the product row, classifier row, seal row, and the displayed observation-budget rows.

Next apply the RealAlgOrder and completion handoff of Definition 464.94. The downstream $\mathsf{RealAlgOrder}^{\uparrow}$ and $\mathsf{RealCompleteness}^{\uparrow}$ consumers receive the same finite product-budget route, so the seal is not reached through a separate multiplication or completion surface.

Finally use finite product-budget exhaustion from Definition 464.95. That row inventory accounts for every product and budget coordinate available to the consumer; because quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, and ambient completeness are absent from the inventory, none can be exported by the Real-facing seal route. □

Theorem 464.97 (CauchyProduct budget-triangle selector compatibility)**.** *Any Real-facing consumer that reads an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet through the observation-budget triangle and then through a* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *route factors through the shared finite product rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

the $\mathsf{RealObservationBudget}^{\uparrow}$ *rows*

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B,$$

and the selector rows

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

The common seal route is read through the displayed $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, and it exports no quotient stream, selected host limit, countable-choice row, trichotomy row, host multiplication primitive, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget triangle selector compatibility`

Proof. Start with the observation-budget triangle of Definition 464.93. It identifies the product-facing read with the paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation rows, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, the $\mathsf{Real}^{\uparrow}$ seal row, and the displayed product and budget rows.

Next apply finite product-budget exhaustion from Definition 464.95. This accounts for every product and $\mathsf{RealObservationBudget}^{\uparrow}$ coordinate visible to the Real-facing consumer before any selector handoff occurs.

Finally pass to the $\mathsf{DiagonalTailSelector}^{\uparrow}$ route by Definition 697.6. The selector read contributes only the rows $R, n, \mu, k, W, D, T, S, H, C, P, N$ and uses the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface already identified by the product-budget triangle.

Combining the three row inventories gives the stated factorization. Since none of the product, budget, or selector rows contains a quotient stream, selected host limit, countable-choice row, trichotomy row, host multiplication primitive, or ambient completeness coordinate, the common Real seal route cannot export any of those objects. □

Theorem 464.98 (CauchyProduct shared-tail selector factorization)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet be consumed by a Real-facing route whose paired source windows are served by an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet and whose seal route then passes through a* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Then the consumer factors through the product rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

the shared-tail rows

$$R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N,$$

and the selector rows

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

The combined route exports no quotient stream, selected host limit, countable-choice row, trichotomy row, host multiplication primitive, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared tail selector factorization`

Proof. Start with the product side. Apply Definition 464.97; the Real-facing product read is already confined to the displayed product rows, the observation-budget route, and the $\mathsf{DiagonalTailSelector}^{\uparrow}$ rows.

Next identify the paired source windows used by that product read. The diagonal-consistency route of Definition 464.117 keeps the two source windows on the same finite product packet, so the shared-tail packet may serve the read without replacing it by a host product or a quotient stream.

Now insert the shared-tail modulus surface. By Definition 983.4, every completion-facing Real seal read factors through $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$. The paired-window readback and route replay rows keep the sources R_0, R_1 on the same displayed route, so the full shared-tail inventory is $R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$.

Finally use Definition 983.9 together with the diagonal selector boundary Definition 697.6. The shared threshold cannot be replaced by private thresholds, and the selector contributes only $R, n, \mu, k, W, D, T, S, H, C, P, N$.

Combining the product, shared-tail, and selector inventories gives the stated factorization. Since none of the three finite row lists contains quotient streams, selected host limits, countable-choice rows, trichotomy, host multiplication, or ambient completeness, no Real or completion consumer can obtain such a coordinate from the combined route. □

Lemma 464.99 (CauchyProduct observation-budget downstream coverage)**.** *Every accepted local consumer downstream of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet read through the* $\mathsf{RealObservationBudget}^{\uparrow}$ *observation-budget triangle is exhausted by the finite alternatives, ledgers, and coverage rows already displayed in the triangle file. Concretely, the consumer can read only the product-side rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

the observation-budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B,$$

or the downstream $\mathsf{RealAlgOrder}^{\uparrow}$ *and* $\mathsf{RealCompleteness}^{\uparrow}$ *handoff route carried by the same finite product-budget inventory. Quotient streams, selected product limits, host multiplication primitives, trichotomy rows, independent regular-sequence or real carriers, and ambient completeness witnesses remain outside the accepted BEDC packet.*

Proof. Start with the observation-budget triangle of Definition 464.93. It says that any accepted product observation consumed through $\mathsf{RealObservationBudget}^{\uparrow}$ factors through the paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, the $\mathsf{Real}^{\uparrow}$ seal row, and exactly the displayed product and budget row lists. Thus every local product-facing alternative has already entered one of the listed source-window, dyadic, classifier, seal, product-ledger, or budget-ledger coordinates.

Now pass to downstream consumers by Definition 464.94. A $\mathsf{RealAlgOrder}^{\uparrow}$ multiplication consumer and a $\mathsf{RealCompleteness}^{\uparrow}$ completion-facing consumer receive the same finite product-budget route, namely the phase handoff together with the ledger obligation. Therefore the downstream read does not add a second multiplication surface, a selected completion surface, or an independent real carrier; it only reuses the rows already supplied by the triangle.

Finally apply the finite product-budget exhaustion of Definition 464.95 and the Real seal exhaustion of Definition 464.96. These two coverage statements account for every product, budget, and seal-facing coordinate available to a local consumer. Since quotient streams, selected product limits, host multiplication primitives, trichotomy rows, independent regular-sequence or real carriers, and ambient completeness witnesses occur in none of those inventories, there is no row from which such an excluded alternative can be exported by the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket observation budget downstream coverage`

Theorem 464.100 (CauchyProduct root source window totality)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, each product read first exposes the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y*. The exposed rows are exactly the finite source-window route named by the carrier, the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *obligations, the StreamName handoff, the root window obligations, and the shared tail-window surface of chapter 560.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root source window totality`

Proof. Read the carrier of Definition 464.1. Its first source coordinates are X, Y, W_X, W_Y, with the two source rows displayed before the dyadic observations and before the product row.

Apply the NameCert obligations and the StreamName handoff. These force any consumer of the product packet to enter the two sources through W_X, W_Y before it can inspect M_X, M_Y, Q_X, Q_Y, P, or E.

Finally use the root window obligations together with chapter 560. The sibling tail-meet chapter supplies the same finite tail-window consumption pattern, so the product read has no alternate source route outside the paired StreamName windows. □

Theorem 464.101 (CauchyProduct root budget product lock)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the dyadic product row* P *is locked by the two source modulus rows* M_X, M_Y *and the product budget ledger* L *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *is read. The lock is the finite route supplied by the carrier, the local NameCert obligations, the StreamName handoff, the root window obligations, the root budget obligations, and the budget triangle package.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget product lock`

Proof. Start from the paired source-window route. By source-window totality, W_X, W_Y expose the two regular Cauchy sources before the dyadic observations and product row are available.

The root budget obligations then read M_X, M_Y, Q_X, Q_Y, P, L as the budget route into the product classifier. The budget triangle package keeps that route compatible with the downstream product and observation-budget consumers.

Thus E is reached only after P has been formed inside the displayed dyadic core and after L has recorded the two source modulus rows. No consumer can read the classifier through a separate product estimate, radius oracle, quotient stream, selected limit, host multiplication row, or ambient completeness coordinate. □

Theorem 464.102 (CauchyProduct root source-window unblock certificate)**.** *The source-window totality row, the budget product lock row, and the existing root budget triangle rows form the root-unblock certificate consumed by* $\mathsf{Real}^{\uparrow}$*-facing product consumers. The certificate is the finite* $\mathsf{CauchyProduct}^{\uparrow}$ *route from* X, Y *through* $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L$ *to* E *and then to the displayed* $\mathsf{Real}^{\uparrow}$ *seal handoff; it exports no quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root unblock certificate`

Proof. First assemble the source side. The carrier, NameCert obligations, StreamName handoff, and source-window totality theorem expose the two $\mathsf{RegularCauchyName}^{\uparrow}$ rows only through the paired StreamName windows.

Next assemble the product side. The root window obligations, root budget obligations, and budget product lock route the dyadic product row through the two source modulus rows and the product ledger before the classifier row is read.

Finally hand the joined route to the existing root budget triangle rows and the sibling finite-tail surface of chapter 560. A $\mathsf{Real}^{\uparrow}$-facing product consumer therefore receives exactly the displayed finite route and has no coordinate for any excluded analytic object. □

Theorem 464.103 (CauchyProduct source-window seal consumer exactness)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, any consumer that exports the product as a* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite route*

$$X, Y \to W_X, W_Y \to M_X, M_Y, Q_X, Q_Y, P, L \to E.$$

The factorization is forced by the source-window totality row, the budget product lock row, and the root-unblock certificate. Such a consumer exports no quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket source window seal consumer exactness`

Proof. First apply Definition 464.100. Every product read reaches the two source rows only through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y, so the consumer has no direct source coordinate outside the displayed window route.

Next apply Definition 464.101. The dyadic product row is read through M_X, M_Y, Q_X, Q_Y, P, L before the classifier row E, so the source modulus rows and product ledger are the only budgeted path into the $\mathsf{RegSeqRat}^{\uparrow}$ classifier.

Finally apply Definition 464.102 together with the sibling finite-tail surface of chapter 560. The downstream $\mathsf{Real}^{\uparrow}$-facing seal therefore receives exactly the stated finite route, and the excluded analytic objects have no exported coordinate in the accepted packet. □

Theorem 464.104 (CauchyProduct root source-window admission)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, every product read admits the two source rows* X, Y *only through the paired windows* W_X, W_Y *and the displayed* $\mathsf{StreamName}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *route. The read cannot inspect either source before the window rows, and it cannot replace the window route by a quotient stream, selected product limit, host multiplication primitive, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root source window admission`

Proof. First apply Definition 464.100. It exposes the two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows exactly through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y.

Next apply Definition 464.107. The route from the source rows to the classifier must pass through the displayed dyadic and regular-rational rows, so no source read can enter after the windows have been skipped.

Finally use Definition 464.102. The root certificate excludes quotient streams, selected limits, host multiplication, and ambient completeness from the product route, leaving the paired source-window admission as the only source entry. □

Theorem 464.105 (CauchyProduct window product readback)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the product row* P *is read back from* M_X, M_Y, Q_X, Q_Y *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *is consumed. The readback is the finite product route through* $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L, E$*, not a route through host multiplication or a selected product limit.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductWindowProductReadback`

Proof. Begin with source-window admission. It fixes the source rows behind W_X, W_Y before any modulus, dyadic, product, or classifier row can be read.

Apply Definition 464.101. The dyadic product row P is locked by the source modulus rows M_X, M_Y and the product ledger L before the classifier E is available.

The source-window seal consumer exactness theorem then gives the complete order of the readback into E. Thus P is read from M_X, M_Y, Q_X, Q_Y inside the finite BEDC route, with no host product primitive or selected limit coordinate. □

Theorem 464.106 (CauchyProduct product seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet receives only the finite route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

The consumer obtains no quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket product seal nonescape`

Proof. First use Definition 464.104. It confines the two source rows to the paired StreamName windows and the displayed dyadic route.

Next use Definition 464.105. The product row and classifier are read only after the source modulus and dyadic readback rows have been fixed.

Finally apply Definition 464.103 and Definition 464.102. These results hand the product route to a $\mathsf{Real}^{\uparrow}$-facing seal consumer and exclude quotient streams, host multiplication, selected limits, trichotomy rows, independent real carriers, and ambient completeness from the exported surface. □

Theorem 464.107 (CauchyProduct StreamName-RegSeqRat source lock)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, every product read locks both source rows through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or dyadic product row is read. The locked route is the finite path*

$$X, Y \to W_X, W_Y \to M_X, M_Y, Q_X, Q_Y, P, L \to E.$$

It exports no unpaired source row, quotient stream, selected product limit, host multiplication primitive, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket streamname regseqrat source lock`

Proof. First use Definition 464.100. It places the two source rows behind the paired $\mathsf{StreamName}^{\uparrow}$ windows, so a product read cannot inspect X or Y independently of W_X, W_Y.

Next apply Definition 464.101. The dyadic product rows and product ledger are read only after the source modulus and dyadic observation rows, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E is downstream of that product ledger.

Finally combine the root-unblock certificate with the StreamName-RegSeqRat consumer surfaces of chapter 126 and chapter 301. The route reaching E is exactly the displayed paired-window route, with no coordinate for an alternate source or host product object. □

Theorem 464.108 (CauchyProduct RealTailAgreementSeal handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing product consumer that compares product tails of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet must pass through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699. The comparison consumes the paired source-window lock, the product budget lock, and the root-unblock certificate before it reaches a RealUp seal; it cannot replace that route by a selected host limit, quotient stream, host equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real tail agreement handoff`

Proof. Begin with Definition 464.103. A RealUp-facing product seal already factors through the finite route from the two source rows through the paired windows, dyadic observations, product row, ledger, and classifier.

Use Definition 464.107 to make the source side rigid. Both sources are locked through W_X, W_Y before RegSeqRat readback or product rows can be consumed.

Now read the sibling $\mathsf{RealTailAgreementSeal}^{\uparrow}$ chapter. Its paired-window route compares tails only through displayed StreamName windows, dyadic readbacks, agreement rows, and the RealUp handoff. Composing that route with the product root-unblock certificate gives the required product-tail comparison surface.

The excluded alternatives are absent from both packets. Therefore a product-tail consumer reaches the RealUp seal through $\mathsf{RealTailAgreementSeal}^{\uparrow}$ and receives no selected limit, quotient stream, host equality, host multiplication primitive, or ambient completeness witness. □

Theorem 464.109 (CauchyProduct paired-window tail-agreement exhaustion)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, if a product-tail consumer reaches a* $\mathsf{Real}^{\uparrow}$ *seal, then its paired tail-agreement comparison is exhausted by the finite route through* W_X, W_Y*,* M_X, M_Y, Q_X, Q_Y, P, L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*, and the sibling* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. The route is the one supplied by the* $\mathsf{StreamName}^{\uparrow}$ *source lock, the product budget lock, the RealTailAgreementSeal handoff, and the sibling surfaces of chapter 126, chapter 301, and chapter 699. It exports no quotient stream, host multiplication primitive, host equality, selected limit, trichotomy row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket source window tail agreement exhaustion`

Proof. First apply Definition 464.107. This fixes the two source rows behind the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y before any $\mathsf{RegSeqRat}^{\uparrow}$ classifier or product read is available.

Next apply Definition 464.101. The product-tail consumer can reach E only through M_X, M_Y, Q_X, Q_Y, P, L, so the product comparison has no separate budget, product, or classifier route.

Now apply Definition 464.108. The Real-facing tail comparison is handed to the sibling $\mathsf{RealTailAgreementSeal}^{\uparrow}$ packet, whose chapter supplies the displayed paired-window agreement surface.

Finally read the sibling chapters chapter 126, chapter 301, and chapter 699 as the three consumer boundaries. Their surfaces account for the StreamName windows, the regular-rational classifier row, and the RealTailAgreementSeal packet, leaving no coordinate for quotient streams, host multiplication, host equality, selected limits, trichotomy, or ambient completeness. □

Corollary 464.110 (CauchyProduct root source-window strict obstruction)**.** *Let a displayed failure, refusal, separation, or countermodel row be cited against a named root source-window converse or reuse route for an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read omits, replaces, detaches, or bypasses one of the displayed finite coordinates*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

Every permitted consumer-facing use of the obstruction factors first through the paired $\mathsf{StreamName}^{\uparrow}$ *source windows* W_X, W_Y *and then through the finite product-seal route named in Definition 464.106. The obstruction has no force as a global impossibility theorem, no force as a host theorem about products or completion, and no force as hidden external validation for any route not displayed by the accepted packet.*

Proof. Start with Definition 464.100. It makes the two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows visible only through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y, and it identifies the exposed source rows with the carrier route, the local $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ obligations, the StreamName handoff, the root window obligations, and the shared finite tail-window surface. Thus any consumer-facing converse or reuse route for root source-window totality must first ask for those paired windows before it can read any downstream product coordinate.

Now apply Definition 464.106. A $\mathsf{Real}^{\uparrow}$ seal consumer receives only the finite route

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

It receives no quotient stream, host multiplication primitive, selected limit, trichotomy row, independent real carrier, or ambient completeness witness. Therefore every consumer-facing use of a displayed failure, refusal, separation, or countermodel row factors through the same finite route: the source part is forced by W_X, W_Y, and the product-seal part is exhausted by $M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$.

Consider the named converse or reuse route against which the displayed row is cited. If the route omits, replaces, detaches, or bypasses one of the listed coordinates, the row blocks that named route exactly at the missing finite coordinate. If the proposed reading instead asks for a quotient stream, host multiplication theorem, selected product limit, trichotomy row, independent real carrier, ambient completeness witness, or external validation channel, the two cited theorems supply no coordinate through which that request can be consumed. Such a request is rejected

because it is unlisted in the accepted packet, not because the displayed row proves a stronger global impossibility.

Hence the obstruction remains local to the named root source-window converse or reuse route. It is a negative certificate for a failed finite packet read, while global impossibility statements, host product theorems, completion principles, and hidden external validation remain outside the exported $\mathsf{CauchyProduct}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root source window strict obstruction`

Theorem 464.111 (CauchyProduct budgeted root source triple)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and observation rows form the admitted root source triple for the product classifier. The triple is the finite* BHist *route*

$$(X, Y), \quad (W_X, W_Y), \quad (M_X, M_Y, Q_X, Q_Y)$$

inside the packet

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N),$$

and it reaches the product row and $\mathsf{RegSeqRat}^{\uparrow}$ *classifier only through the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows.*

Proof. Project the carrier of Definition 464.1. Its displayed coordinates put the two regular Cauchy source rows, the paired windows, and the dyadic radius and observation rows in one finite packet before the product row P and classifier row E are read.

Apply the root source admission of Definition 464.82 and the root source window totality of Definition 464.100. These rows force every product read to enter through the two source rows and the paired StreamName windows.

Now use the root budget product lock of Definition 464.101. The dyadic observations and radius rows are the budgeted route into P and E, while H, C, L, N carry the displayed transport, continuation, package, and local naming evidence. Coordinate exhaustion leaves no second source route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budgeted root source triple`

Theorem 464.112 (CauchyProduct window-budget NameCert ledger)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet feeding a product observation into* $\mathsf{RegSeqRat}^{\uparrow}$*, the finite window-budget ledger is the only* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *ledger available to that observation. It is the displayed inventory*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L, H, C, N$$

together with the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E *and, when a* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet consumes the read, the budget rows*

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

No separate ledger can feed the product observation into $\mathsf{RegSeqRat}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket window budget namecert ledger`

Proof. First read the product ledger through Definition 464.89. It binds the radius rows, product row, transport rows, continuation rows, package provenance, and local naming row as one finite accounting surface.

Next apply the window-budget composition surface of Definition 464.50. This identifies the path from paired StreamName windows and DyadicRatCore rows to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier E as a single finite route.

Finally use the budgeted product consumer boundary of Definition 464.53. If a budget packet is present, it adds only the displayed budget rows, and those rows read the same product ledger. Thus the NameCert ledger feeding the RegSeqRat observation is exactly the stated inventory. □

Theorem 464.113 (CauchyProduct Real seal root consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing product seal read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the root package. The consumer boundary is the finite route from the two source rows through the paired windows, dyadic radius and observation rows, product row, ledger, classifier row, and seal handoff; it does not pass through host multiplication, quotient streams, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal root consumer boundary`

Proof. Begin with the budgeted root source triple of Definition 464.111. It fixes the source, window, radius, and dyadic observation rows before the product row and classifier can be inspected.

Use the window-budget NameCert ledger of Definition 464.112. The ledger identifies the exact route through P, L, H, C, N and, when present, through the observation-budget rows feeding the same classifier.

Finally apply the Real seal root surface of Definition 464.62 together with the budget-triangle Real seal exhaustion of Definition 464.96. These theorems place every real-facing product seal behind the same finite product and budget rows, so host multiplication, quotient streams, selected limits, and ambient completeness have no consumer coordinate. □

Theorem 464.114 (CauchyProduct root Real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read that consumes an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet factors through the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the dyadic radius and readback rows, the product row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows. In* BHist *coordinates the read is routed through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

with each transport witnessed by the displayed hsame *rows, each admissible continuation witnessed by the displayed* Cont *rows, and each package handoff witnessed by the displayed* Pkg *and* NameCert *rows. No* $\mathsf{Real}^{\uparrow}$ *consumer obtains a separate seal factor outside this finite route.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root real seal factorization`

Proof. Start from the budgeted root source triple of Definition 464.111. It supplies the two regular Cauchy sources, the paired finite windows, and the dyadic budget rows before the product and classifier rows can be inspected.

Use the window-budget NameCert ledger of Definition 464.112. The ledger binds P, E, H, C, L, N to the same finite route and records the local NameCert evidence for the product observation.

Finally apply the Real seal root consumer boundary of Definition 464.113. Since that boundary admits only the displayed root package rows for real-facing seal reads, the factorization is exactly the stated BHist, hsame, Cont, Pkg, and NameCert route. □

Theorem 464.115 (CauchyProduct root RegSeqRat classifier lock)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier input used by a product consumer is locked to the paired-window and dyadic-budget route*

$$(W_X, W_Y, M_X, M_Y, Q_X, Q_Y) \longrightarrow P \longrightarrow E.$$

The classifier row E may read the two regular Cauchy source rows only through the displayed paired $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic radius rows, dyadic readback rows, product row,* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *ledger. It cannot be fed by an alternate product stream, quotient representative, ambient multiplication operation, or selected limit.*

Proof. The budgeted root source triple in Definition 464.111 identifies the only source, window, and dyadic budget coordinates available before P and E are formed.

The window-budget NameCert ledger of Definition 464.112 then ties those coordinates to the product row and classifier row as one local accounting surface. Any classifier input that omits this ledger has no $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ row in the packet.

The Real seal root consumer boundary of Definition 464.113 excludes all non-root analytic routes at the consumer edge. Therefore the $\mathsf{RegSeqRat}^{\uparrow}$ classifier input is locked to the displayed paired-window and dyadic-budget route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regseqrat classifier lock`

Theorem 464.116 (CauchyProduct root completion non-escape)**.** *Every completion-facing product consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet remains inside the finite root package route. In particular, the consumer may use only the two regular Cauchy source rows, paired windows, dyadic radius and readback rows, product row,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row,* BHist *route,* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows displayed in the packet. Host multiplication, quotient streams, selected limits, and ambient completeness are not available as consumer witnesses for the product seal.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root completion nonescape certificate`

Proof. Apply the budgeted root source triple of Definition 464.111. It places every product consumer behind the two source rows, paired windows, and dyadic budget rows.

Apply the window-budget NameCert ledger of Definition 464.112. The ledger shows that the product observation and $\mathsf{RegSeqRat}^{\uparrow}$ classifier share the same finite BHist route, transport rows, continuation rows, package provenance, and local naming row.

Apply the Real seal root consumer boundary of Definition 464.113. That boundary already denies host multiplication, quotient streams, selected limits, and ambient completeness at the real-facing seal edge, so a completion-facing consumer has no remaining route by which to escape the displayed root package. □

Theorem 464.117 (CauchyProduct root source diagonal consistency)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *root package presents one diagonal-consistent source route from the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows through the paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier input, and the* $\mathsf{Real}^{\uparrow}$ *seal. The route is the finite handoff*

$$(X, Y) \to (W_X, W_Y) \to (M_X, M_Y, Q_X, Q_Y) \to P \to E \to \mathsf{Real}^{\uparrow}$$

inside the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ *rows. It denies host multiplication, quotient streams, selected limits, and ambient completeness as source witnesses for the product seal.*

Proof. Begin with Definition 464.111. It fixes the two source rows, paired StreamName windows, and dyadic budget rows as the only root source triple feeding the product classifier.

Use Definition 464.112. The ledger ties the windows, dyadic observations, product row, classifier row, transport rows, continuation rows, package provenance, and local naming row into one finite accounting surface.

Next apply Definition 464.113 and Definition 464.115. These rows place the Real-facing seal behind the same root package and lock the $\mathsf{RegSeqRat}^{\uparrow}$ classifier input to the paired-window and dyadic-budget route.

Finally use Definition 464.116. Since completion-facing consumers cannot leave the displayed root package, the preceding rows form one diagonal-consistent Real/Completion handoff and no host multiplication, quotient stream, selected limit, or ambient completeness coordinate remains available. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root source diagonal consistency`

Theorem 464.118 (CauchyProduct source-window budget compatibility)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *root package, the source-window route and the product-budget route are compatible on the single finite consumer path*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L, E, H, C, N.$$

The compatibility composes the root source-window totality, budget product lock, StreamName-RegSeqRat source lock, RealTailAgreementSeal handoff, and root source diagonal consistency. It leaves no route through quotient streams, host multiplication, selected limits, trichotomy, or ambient completeness.

Proof. Start with Definition 464.100. It exposes the two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows only through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y.

Apply Definition 464.101. This places the dyadic rows M_X, M_Y, Q_X, Q_Y, product row P, and budget ledger L before the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row E is read.

Now use Definition 464.107 and Definition 464.117. The source windows and budget rows feed one diagonal-consistent route to E and then to the $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 464.108. The Real-facing handoff consumes the same finite route, so the excluded analytic resources have no coordinate from which to enter the product seal. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket source window budget compatibility`

Theorem 464.119 (CauchyProduct RegSeqRat product budget exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier read used by an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *product seal is exhausted by the displayed rows* $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L, E, H, C, N$*. In particular, the classifier input is obtained from the paired source windows, dyadic radius and observation rows, product row, and budget ledger, rather than from a quotient stream, a host multiplication primitive, a selected product limit, trichotomy, or ambient completeness.*

Proof. Read the classifier lock in Definition 464.107. It fixes both source rows behind W_X, W_Y before any $\mathsf{RegSeqRat}^{\uparrow}$ or product row can be consumed.

Use Definition 464.101. The lock makes M_X, M_Y, Q_X, Q_Y, P, L the only budget route into E.

Apply Definition 464.118. The compatibility theorem joins the source windows, budget rows, classifier row, and structural rows into one route.

The non-escape part follows from Definition 464.116. Since completion-facing consumers cannot leave the root package, the classifier read is exhausted by the displayed budget route. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket regseqrat product budget exhaustion`

Theorem 464.120 (CauchyProduct Real seal source-budget non-escape)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing product seal consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet must pass through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, L, E, H, C, N$$

before exporting the product seal. The route composes the source-window totality, product-budget exhaustion, StreamName-RegSeqRat lock, RealTailAgreementSeal handoff, and root source diagonal consistency; no quotient stream, host multiplication primitive, selected limit, trichotomy row, or ambient completeness witness is exported with the seal.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal source budget nonescape`

Proof. Begin with Definition 464.118. It gives the single finite source-budget path from the two source rows through the paired windows, dyadic rows, product row, budget ledger, classifier row, and structural rows.

Apply Definition 464.119. This exhausts the $\mathsf{RegSeqRat}^{\uparrow}$ classifier input by the same product-budget path.

Next use Definition 464.108. The Real-facing seal comparison consumes the paired-window and product-budget route before reaching the sibling $\mathsf{RealTailAgreementSeal}^{\uparrow}$ surface.

Finally apply Definition 464.117. The Real/Completion handoff is diagonal-consistent inside the displayed root package, so no excluded analytic object can be repacked as a source-budget witness. □

Corollary 464.121 (CauchyProduct budgeted root source strict obstruction)**.** *For an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, any displayed failure, refusal, separation, or countermodel row attached to the budgeted root source triple is a strict local obstruction. It blocks only a proposed converse or reuse route that tries to feed the product classifier, the completion-facing consumer, or the product seal through a coordinate outside the displayed root package*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

Equivalently, the obstructed route is local to the named missing coordinate: an unlisted source reuse, an unlisted window transport, an unlisted dyadic readback, an unlisted classifier handoff, or an unlisted completion consumer witness. The same row cannot be read as a global impossibility theorem, a host multiplication theorem, a quotient-stream theorem, a selected-limit theorem, an ambient-completeness theorem, or hidden external validation of the product construction.

Proof. Start with the budgeted root source triple of Definition 464.111. It identifies the only admitted source data before the product classifier is read: the two $\mathsf{RegularCauchyName}^{\uparrow}$ source rows, the paired $\mathsf{StreamName}^{\uparrow}$ finite-window rows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius and observation rows. In packet coordinates this is the finite route

$$(X, Y), \quad (W_X, W_Y), \quad (M_X, M_Y, Q_X, Q_Y)$$

inside

$$Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N).$$

The theorem also fixes the available handoff rows to the displayed hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CauchyProduct}^{\uparrow}}$ data. Thus a proposed converse or reuse route has a consumer-facing coordinate only when it factors through one of those rows.

Now apply Definition 464.116. Every completion-facing product consumer remains inside the same finite root package, and the theorem explicitly denies host multiplication, quotient streams, selected limits, and ambient completeness as consumer witnesses for the product seal. Therefore a displayed failure, refusal, separation, or countermodel row can obstruct only the attempted escape from that package: it marks that the named converse or reuse route has no listed source, transport, readback, classifier, or completion-consumer coordinate.

The obstruction is strict because the accepted packet still retains the admitted root source triple and the finite classifier route supplied by Definition 464.111. The obstruction is local because Definition 464.116 exhausts the consumer surface by the displayed rows and rejects only unlisted coordinates. Reading the same row as a global impossibility, host theorem, quotient-stream theorem, selected-limit theorem, ambient-completeness theorem, or hidden external validation would add a witness outside the packet inventory, contradicting the non-escape boundary just cited. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budgeted root source strict obstruction`

Theorem 464.122 (CauchyProduct source-window bilinear synchronization)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, the paired source windows from* StreamName$^{\uparrow}$ *synchronize the two* RegularCauchyName$^{\uparrow}$ *inputs before any product row is read. Concretely, every product observation must first factor through the finite source route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y$$

over the sibling carriers of chapter 126 and chapter 291. The row P is therefore a bilinear readback from the displayed pair of regular-Cauchy source windows, not a host multiplication primitive, selected limit, quotient stream, or ambient product operation.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket source window bilinear synchronization`

Proof. Start with the root source admission theorem Definition 464.82. It places the two RegularCauchyName$^{\uparrow}$ source rows and the paired StreamName$^{\uparrow}$ windows inside the same finite BHist surface.

Next use the radius budget conservation theorem Definition 464.66. The dyadic rows from chapter 291 attach the source windows to the product-radius ledger before P can be inspected.

The sibling regular-rational and real chapters, chapter 301 and chapter 38, then consume only that displayed route. Since the route has no coordinate for host multiplication or quotient streams, the product row is synchronized by the paired windows before any downstream read. □

Theorem 464.123 (CauchyProduct shared budget radius exhaustion)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet and every product observation requested by a downstream reader, the shared dyadic radius and budget rows exhaust the product observation before the* RegSeqRat$^{\uparrow}$ *classifier and* Real$^{\uparrow}$ *seal are consumed. The only available route is the finite ledger*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the displayed observation-budget rows when such a budget packet is present. No second radius source, hidden selector, selected limit, quotient stream, host multiplication estimate, or ambient completeness row can feed the classifier after this ledger has been consumed.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared budget radius exhaustion`

Proof. Read the conservation surface through Definition 464.66. It identifies the paired source radii, dyadic product observations, product row, and classifier handoff as one finite accounting route.

Apply Definition 464.71. The $\mathsf{RegSeqRat}^{\uparrow}$ classifier is reached only after the displayed product route has been read, and the $\mathsf{Real}^{\uparrow}$ seal is downstream of the same classifier boundary.

Finally combine this with the handoff row Definition 464.94. The order and completion consumers receive the same exhausted ledger rather than a fresh analytic source, so the radius and budget rows close the product observation before the classifier and seal are consumed. □

Theorem 464.124 (CauchyProduct Real-completion consumer lock)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *product consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is locked to the same finite route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and, when present, to the same finite observation-budget rows. Such a consumer cannot import quotient streams, selected limits, host multiplication, trichotomy, independent real carriers, or ambient completeness as an alternative source for the product seal.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real completion consumer lock`

Proof. First use source-window synchronization from Definition 464.122. It fixes the paired $\mathsf{RegularCauchyName}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ source surface before the product row is read.

Next apply shared budget radius exhaustion from Definition 464.123. The $\mathsf{RegSeqRat}^{\uparrow}$ classifier and $\mathsf{Real}^{\uparrow}$ seal are reached only through the conserved dyadic radius and product-budget ledger.

Now read the consumer handoff through Definition 464.94 together with the sibling surfaces chapter 301 and chapter 38. The $\mathsf{RealAlgOrder}^{\uparrow}$ and $\mathsf{RealCompleteness}^{\uparrow}$ consumers therefore receive exactly the finite route already displayed by the product packet, and no excluded analytic object has a coordinate from which it could be imported. □

Theorem 464.125 (CauchyProduct finite-window real-seal factorization)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet consumed through an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, the resulting* $\mathsf{Real}^{\uparrow}$ *seal factors through exactly the paired source windows* W_X, W_Y*, dyadic rows* Q_X, Q_Y*, radius rows* M_X, M_Y*, product row* P*, classifier row* E*, product ledger* L*, and the budget rows*

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

The factorization also contains the two source rows X, Y *that feed the paired windows. No quotient stream, selected limit, host multiplication, trichotomy row, independent real carrier, or ambient completeness coordinate can enter the product seal.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket finite window real seal factorization`

Proof. Start with source-window synchronization from Definition 464.122. It fixes the source rows X, Y and the paired finite windows W_X, W_Y before the product row is read.

Apply shared budget radius exhaustion from Definition 464.123. The only product-side route to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and $\mathsf{Real}^{\uparrow}$ seal passes through $M_X, M_Y, Q_X, Q_Y, P, E, L$, so the product packet supplies no separate selector, quotient stream, or host multiplication coordinate.

Now attach the observation-budget carrier. Window coverage in Definition 611.3 sends every budget dyadic readback and displayed radius through E_B, W_B, D_B, while the real-seal boundary theorem Definition 611.5 makes S_B the completion-facing boundary reached through R_B, H_B, C_B, P_B, N_B.

Finally combine the consumer lock Definition 464.124 with budget non-escape Definition 611.6. The consumer receives the product seal only through the two displayed finite ledgers, and every excluded analytic object lacks a row on either side of the factorization. □

Theorem 464.126 (CauchyProduct synchronized source-window downstream coverage)**.** *For every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, every accepted local downstream consumer of its product observation is covered by the finite source-window and budget surface displayed in this file. More explicitly, before such a consumer can read the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, the* $\mathsf{Real}^{\uparrow}$ *seal, a* $\mathsf{RealAlgOrder}^{\uparrow}$ *product-order row, a* $\mathsf{RealCompleteness}^{\uparrow}$ *completion row, or an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *observation row, its route has been exhausted by the source rows*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y,$$

the product ledger

$$P, E, H, C, L, N,$$

and, when a budget packet is present, the budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

Consequently the local consumer has no remaining BEDC coordinate for a second radius source, hidden selector, selected limit, quotient stream, host multiplication estimate, trichotomy branch, independent real carrier, or ambient completeness row.

Proof. Start with source-window bilinear synchronization from Definition 464.122. It forces every product observation to enter through the two source rows X, Y, the paired finite windows W_X, W_Y, the paired radius rows M_X, M_Y, and the dyadic observation rows Q_X, Q_Y before the product row P is read. Thus no downstream consumer can begin from host multiplication, quotient-stream comparison, a selected limit, or an ambient product operation.

Next apply shared budget radius exhaustion from Definition 464.123. Once the product observation is requested, the finite ledger

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

exhausts the route to the $\mathsf{RegSeqRat}^{\uparrow}$ classifier and the $\mathsf{Real}^{\uparrow}$ seal, with the displayed observation-budget rows attached when such a budget packet is present. This accounts for the possible local alternatives: the classifier-facing read, the seal-facing read, and the budget-facing read are all rows of the same finite packet.

Now read the local completion and budget consumers through Definition 464.124 and Definition 464.125. The first theorem locks every $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{RealCompleteness}^{\uparrow}$ product consumer to the displayed product ledger, and the second theorem identifies the budgeted seal factorization through

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B.$$

Therefore every accepted local downstream consumer is exhausted by one of the enumerated source, ledger, or budget rows. The excluded alternatives are not delayed choices inside the packet; they have no row in the synchronized source surface, no row in the shared radius ledger, and no row in the budgeted seal factorization, so they remain outside the BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket source window downstream coverage`

Theorem 464.127 (CauchyProduct budget route package)**.** *Let* $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C$ *be an accepted* CauchyProduct$^{\uparrow}$ *packet. Suppose its product is read together with an accepted* FiniteCauchyApproximation$^{\uparrow}$ *finite-approximation packet and an accepted* RealObservationBudget$^{\uparrow}$ *observation-budget packet over the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, and* CauchyLimitSeal$^{\uparrow}$ *rows. Then these packets assemble one finite product budget route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

with the finite approximation rows and the observation-budget rows

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$$

over BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert *evidence. The assembled route is the one consumed by* Real$^{\uparrow}$ *and completion-facing Cauchy readers; it is not a loose union of separate budget lemmas.*

Proof. Begin with the CauchyProduct side. The obligation closure of Definition 464.47, the window-budget composition surface of Definition 464.50, and the root product readiness package of Definition 464.52 bind the paired StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ observations, product row, classifier row, Real seal row, transport rows, continuation rows, package provenance, ledger row, and local certificate row into one finite packet.

Attach the root-unblock and triangle rows next. The source-window unblock certificate of Definition 464.102, the observation-budget triangle of Definition 464.93, and the RealAlgOrder Completion handoff of Definition 464.94 identify the same product route as the route read by downstream Real and completion-facing consumers.

Now add the finite approximation and observation-budget chapters. The accepted FiniteCauchyApproximation$^{\uparrow}$ packet is read through chapter 601, and the accepted RealObservationBudget$^{\uparrow}$ packet is read through chapter 611. These chapters supply the finite approximation, precision, finite-window, dyadic-readback, regular-sequence, real-seal, transport, continuation, provenance, and local naming rows that match the product route.

The joined package is therefore a single route over the displayed kernel objects. Since every cited surface is finite and row-exhaustive, the assembled package exports only the listed product, finite-approximation, observation-budget, transport, continuation, provenance, and certificate rows. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget route package`

Theorem 464.128 (CauchyProduct componentwise window admission)**.** *For an accepted* CauchyProduct$^{\uparrow}$ *packet* Z*, the left and right product windows are admitted componentwise through the displayed* StreamName$^{\uparrow}$ *rows* W_X, W_Y *and the displayed* RegSeqRat$^{\uparrow}$ *route to* E *before any product seal is read. The admission proceeds through the finite window rows, dyadic rows, product row, product-radius ledger,* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* NameCert$_{\text{CauchyProduct}^{\uparrow}}$ *row, with the observation-budget rows attached only after the componentwise source windows have been exposed.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket componentwise window admission`

Proof. First read the source windows. The paired-window source admission of Definition 464.11, the StreamName handoff of Definition 464.20, and the root source window totality of Definition 464.100 expose W_X, W_Y componentwise before the dyadic observations and product row are visible.

Next route the componentwise windows into the classifier. The budget product lock of Definition 464.101, the root RegSeqRat classifier lock of Definition 464.115, and the budget RegSeq classifier lock of Definition 464.54 send the displayed window, dyadic, product, and ledger rows to E as a finite $\mathsf{RegSeqRat}^{\uparrow}$ route.

Finally attach the real-facing boundary. The Real seal root consumer boundary of Definition 464.113 and the finite-window real-seal factorization of Definition 464.125 place every seal read after the componentwise window route. Thus the product seal can read only after the left and right windows have already entered through the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. □

Theorem 464.129 (CauchyProduct Real handoff non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of the product budget route assembled in Definition 464.127 receives only the displayed product rows, finite-approximation rows, observation-budget rows, transport rows, continuation rows, provenance rows, and local* NameCert *rows. The handoff supplies no quotient stream, selected real value, host metric, host multiplication primitive, trichotomy row, independent real carrier, ambient completeness principle, or external Cauchy-limit principle.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real handoff nonescape`

Proof. Start from the product non-escape rows. The window product nonescape of Definition 464.56, the RegSeq seal non-escape of Definition 464.71, and the root completion non-escape of Definition 464.116 rule out quotient streams, selected limits, host multiplication, independent real carriers, and ambient completeness inside the accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet.

Read the budget and real-facing handoff through the route package. The finite product-budget exhaustion of Definition 464.95, the budget-triangle Real seal exhaustion of Definition 464.96, and the finite-window real-seal factorization of Definition 464.125 list every product and observation-budget row visible to the consumer.

Finally compose with the completion-facing sibling surfaces. The finite approximation chapter chapter 601 and the CauchyLimitSeal chapter chapter 386 consume the same displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. Because neither side contributes a quotient stream, selected real value, host metric, ambient completeness principle, or external Cauchy-limit principle, the Real and seal handoff remains inside the displayed finite route. □

Theorem 464.130 (CauchyProduct public budget route export)**.** *Every public* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or completion-facing read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *budget route factors through the assembled* $\mathsf{CauchyProduct}^{\uparrow}$ *rows, the* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary named by Definition 464.127, Definition 464.128, and Definition 464.129. The public export supplies the finite product route and its admitted componentwise windows, but it exports no quotient stream, selected real value, host multiplication primitive, trichotomy row, or independent completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket public budget route export`

Proof. Use the budget route package first. Definition 464.127 assembles the product rows with the finite-approximation and observation-budget packets over the displayed BHist, hsame, Cont, Pkg, and local NameCert evidence.

Next expose the componentwise admission surface. Definition 464.128 forces the $\mathsf{StreamName}^{\uparrow}$ rows and the $\mathsf{RegSeqRat}^{\uparrow}$ entry into the route before any product seal is read, so a public reader receives the product boundary through those displayed windows.

Finally apply the handoff non-escape row. Definition 464.129 states that the $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers of the assembled route receive only the listed product, finite-approximation, observation-budget, transport, continuation, provenance, and certificate rows. The same exclusions rule out quotient streams, selected real values, host multiplication primitives, trichotomy rows, and independent completeness principles from the public export. □

Corollary 464.131 (CauchyProduct budget route strict local obstruction)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X$ be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet. Let $F = (R_F, n_F, k_F, W_F, D_F, E_F, S_F, H_F, C_F, P_F, N_F)$ be the accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet serving the same finite product read, and let $B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$ be the accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet on the same route. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse or reuse route for the product budget package of Definition 464.127. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read tries to omit, replace, detach, or bypass one of the displayed coordinates*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

$$R_F, n_F, k_F, W_F, D_F, E_F, S_F, H_F, C_F, P_F, N_F,$$

$$E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B,$$

together with the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *support rows. The same row cannot be reinterpreted as a global impossibility statement for* $\mathsf{CauchyProduct}^{\uparrow}$, *as a host theorem about multiplication or completion, as an external validation channel, as a quotient-stream argument, as a selected real value, as a host metric, as a trichotomy row, as an independent real carrier, as an ambient completeness principle, or as an external Cauchy-limit principle.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget route strict obstruction`

Proof. First apply the budget route package of Definition 464.127. It assembles the accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet with the accepted finite-approximation and observation-budget packets into one finite product budget route. On the product side the route displays the source rows, paired windows, modulus rows, dyadic readbacks, product row, classifier row, real-seal support, transports, continuations, ledger, provenance, and local naming evidence. On the auxiliary sides it displays the finite approximation rows of F and the observation-budget rows of B. Thus a consumer-facing converse or reuse route has no first read except through these displayed packet coordinates.

Next apply the Real handoff non-escape theorem, Definition 464.129. Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of the assembled route receives only the displayed product rows, finite-approximation rows, observation-budget rows, transport rows, continuation rows, provenance rows, and local $\mathsf{NameCert}$ rows. In particular, quotient streams, selected real values, host metrics, host multiplication primitives, trichotomy rows, independent real carriers, ambient completeness principles, and external Cauchy-limit principles are not exported coordinates of the handoff.

Now consider the cited failure, refusal, separation, or countermodel row. If the named converse or reuse route asks to omit, replace, detach, or bypass one of the displayed product, finite-approximation, observation-budget, transport, continuation, provenance, ledger, or local naming coordinates, the cited row blocks exactly that request at the challenged coordinate of the accepted package. If the route instead asks for a global impossibility statement, a host theorem, an external validation channel, a quotient stream, a selected real value, a host metric, a trichotomy row, an

independent real carrier, an ambient completeness principle, or an external Cauchy-limit principle, Definition 464.129 gives no coordinate through which such a request can be read.

Therefore every permitted consumer-facing use of the obstruction factors through the displayed finite route, and every unlisted coordinate is rejected as outside the accepted $\mathsf{CauchyProduct}^{\uparrow}$ budget route package. The obstruction is strict and local to the named converse or reuse route, and it asserts nothing beyond the displayed package. □

Theorem 464.132 (CauchyProduct budget-selector distributive lock)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y$ be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet. Suppose the two factor-window reads W_X and W_Y are consumed through the same accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *finite observation-budget selector from chapter 782. Then the product read has a single displayed budget route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the selector rows

$$B, S, W, D, R, E_B, H_B, C_B, P_B, N_B.$$

The route is distributive only as a finite BHist *product-route certificate: it distributes the same selector window over the two displayed factor windows and then reads the product through the existing* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *rows. It does not expose host multiplication, quotient stream equality, selected limits, independent real carriers, trichotomy rows, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget selector distributive lock`

Proof. Begin with the two source windows. By Definition 464.56, the only route from the source names to the product observation passes through $W_X, W_Y, M_X, M_Y, Q_X, Q_Y$ and the product ledger before P is visible.

Now read both factor windows through the common finite observation-budget selector of chapter 782. The selector contributes one finite schedule, one finite window family, one dyadic ledger, one regular handoff, one seal row, and one set of transport, continuation, provenance, and local naming rows; it does not create separate selector routes for the two factors.

Finally apply Definition 464.53. The product consumer can inspect exactly the displayed product rows together with the displayed budget rows, so the two factor-window reads and the product read are locked to one finite route. The excluded analytic objects are absent from both accepted packets, hence they cannot enter the distributive product-route certificate. □

Theorem 464.133 (CauchyProduct commuted-budget route)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H$ be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose two factor-window reads are accepted by the same finite observation-budget selector from chapter 782. If the packet is read after swapping W_X with W_Y and swapping the dyadic product inputs, then the same finite product-budget route reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row and the* $\mathsf{Real}^{\uparrow}$ *seal. The commuted route is preserved by componentwise* $\mathsf{StreamName}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-row symmetry,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal non-escape; it does not claim host multiplication or completed-real commutativity.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket commuted budget route`

Proof. First expose the original finite route. The root source-window totality and Definition 464.132 bind both factor windows to the same displayed selector route before the product row is available.

Next commute the source read. Swapping the two factor windows only exchanges the componentwise $\mathsf{StreamName}^{\uparrow}$ handoffs, and the dyadic product row is read through the symmetric $\mathsf{DyadicRatCore}^{\uparrow}$ product observation rather than through host multiplication.

Finally reuse the downstream budget boundary. The root budget product lock and Definition 464.134 keep the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite budget rows. Thus the commuted packet reaches the same classifier and seal route while excluding host multiplication and completed-real commutativity. □

Theorem 464.134 (CauchyProduct Real budget multiplication boundary)**.** *A* $\mathsf{Real}^{\uparrow}$ *multiplication-facing seal for an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet is reached only after the shared* $\mathsf{StreamName}^{\uparrow}$ *factor windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the local* $\mathsf{CauchyProduct}^{\uparrow}$ $\mathsf{NameCert}$ *rows have all been displayed. If the factor windows are read through the finite observation-budget selector of chapter 782, then the multiplication-facing route additionally passes through the selector budget rows before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. No seal-facing consumer receives a host multiplication primitive, a quotient-stream product, a selected real value, an independent real carrier, or an ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real budget multiplication boundary`

Proof. First expose the CauchyProduct product route. The non-escape statement Definition 464.56 fixes the source-window route, and Definition 464.132 binds any shared selector read to the same finite product-route certificate.

Next pass to the real-facing seal. By Definition 464.55, every $\mathsf{Real}^{\uparrow}$ seal read consumed through a budget packet factors through the displayed budget rows before the seal is available.

The boundary follows by joining that gate with Definition 464.53. The consumer sees the shared $\mathsf{StreamName}^{\uparrow}$ windows, the dyadic radius ledger, the regular-rational handoff, the selector rows, the displayed transports and continuations, the provenance rows, and the local $\mathsf{NameCert}$ rows. There is no remaining coordinate from which a host multiplication primitive or ambient completion principle could be read. □

Theorem 464.135 (CauchyProduct RegSeqRat product seal coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing product consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose factor windows are read through the finite observation-budget selector of chapter 782 is exhausted by the rows displayed in Definition 464.132 and Definition 464.134. In particular, the consumer reads the same source windows, dyadic radius rows, product row, classifier handoff, selector rows, transports, continuations, provenance, and local naming rows before any product seal is exposed, and it receives no additional regular-sequence carrier, quotient stream, selected limit, host multiplication, trichotomy row, independent real carrier, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket regseqrat product seal coverage`

Proof. Start with the RegSeqRat side of the product packet. The window product non-escape theorem Definition 464.56 and the budgeted product consumer boundary Definition 464.53 leave only the displayed product rows and budget rows available to a regular-rational consumer.

Now attach the shared selector route. The distributive lock Definition 464.132 shows that both factor-window reads pass through one selector route rather than two unrelated budget routes.

Finally use the real budget multiplication boundary of Definition 464.134. Any product seal read occurs after the same window, dyadic, classifier, selector, transport, continuation, provenance, and local naming rows. Thus the listed rows exhaust every $\mathsf{RegSeqRat}^{\uparrow}$-facing product consumer, and no unlisted analytic or host-level object remains available. □

Lemma 464.136 (CauchyProduct budget-selector downstream coverage)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet read its factor windows through the same accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *selector. Every accepted local consumer downstream of the budget-selector distributive route is exhausted by the finite product rows*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and by the selector and coverage rows already displayed in Definition 464.132 and Definition 464.135. Thus a downstream consumer may read only a factor-window alternative, the product ledger and classifier route, the selector schedule and budget ledger, or the transport, continuation, provenance, and local NameCert *coverage rows. Quotient streams, selected product limits, host multiplication primitives, trichotomy rows, independent regular-sequence or real carriers, and ambient completeness witnesses remain outside the accepted BEDC packet.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket budget selector downstream coverage`

Proof. Start with the distributive route. The budget-selector distributive lock of Definition 464.132 says that the two factor-window reads do not create two independent budget paths. They are consumed through one finite selector route, and the product read is therefore carried by the displayed product rows together with the displayed selector rows.

Now pass to downstream consumers. The RegSeqRat product seal coverage theorem Definition 464.135 exhausts every $\mathsf{RegSeqRat}^{\uparrow}$-facing product consumer by the same source windows, dyadic radius rows, product row, classifier handoff, selector rows, transports, continuations, provenance, and local naming rows before any product seal is exposed.

These two cited statements cover the local alternatives. A factor-window consumer is already on the shared selector route, a ledger or classifier consumer is on the displayed product route, and a seal-facing consumer is on the coverage route of Definition 464.135. No fourth coordinate is exported by either packet inventory. Hence every accepted local consumer is exhausted by the enumerated alternatives, ledgers, and coverage rows, while each excluded analytic alternative remains outside the accepted BEDC packet. □

Theorem 464.137 (CauchyProduct root-selector Real-seal exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing product seal read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose factor windows pass through the shared finite observation-budget selector factors through the existing distributive lock, real budget multiplication boundary,* $\mathsf{RegSeqRat}^{\uparrow}$ *product seal coverage,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. In particular, the seal-facing route is exhausted by the displayed source windows, dyadic product observations, selector budget rows, regular-sequence handoff rows, componentwise transports, continuation replays, provenance rows, and local names; it exposes no host multiplication primitive, quotient stream equality, selected real value, independent real carrier, trichotomy row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root selector real seal exhaustion`

Proof. First read the shared selector route. The distributive lock of Definition 464.132 binds both factor windows to one finite observation-budget selector before a product-facing consumer sees the product rows.

Next pass through the real-facing boundary. The real budget multiplication boundary of Definition 464.134 forces every seal-facing read through the stream windows, dyadic radius and product rows, selector budget rows, regular handoff, transports, continuations, provenance, and local naming.

Finally use the RegSeqRat product seal coverage of Definition 464.135. The regular-sequence side and the real-seal side therefore name the same finite route, so a root-selector consumer has no unlisted coordinate from which to read host multiplication, quotient equality, selected limits, order trichotomy, separate real carriers, or ambient completeness. □

Theorem 464.138 (CauchyProduct shared-selector RegSeqRat–Real seal commutation)**.** *Let* $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N)$ *be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet whose two factor windows are read through the same* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *route. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier handoff and the* $\mathsf{Real}^{\uparrow}$ *seal handoff commute through one displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{FiniteObservationBudgetSelector}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Equivalently, every product-facing consumer first sees the same stream names, dyadic product rows, selector budget rows, componentwise transports, continuation replays, provenance rows, and local naming row before either the regular-sequence classifier or the real seal is consumed. The commutation exposes no host multiplication, quotient stream equality, selected limits, trichotomy, independent real carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared selector regseqrat real seal commutation`

Proof. Start from the shared selector route. By Definition 464.132, the two factor windows are not read by unrelated budget packets: they are displayed through the same finite selector rows and the same product-route certificate.

Now pass to the two downstream handoffs. The real-facing side is bounded by Definition 464.134, while the regular-sequence side is exhausted by Definition 464.135. Both statements name the same stream-window, dyadic, selector, transport, continuation, provenance, and local NameCert rows.

Composing these two readings gives the commutation claim. Whichever handoff a consumer cites first, the other handoff is reached only by replaying the same finite rows, so no unlisted host-level product, quotient equality, selected limit, order trichotomy, separate real carrier, or completeness principle can enter the route. □

Theorem 464.139 (CauchyProduct RealSeries shared-selector handoff)**.** *Let* $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q$ *be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet, and let* $K = (T, S, W, Q, D, M, E_R, H_R, C_R, P_R, N_R)$ *be an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet. Suppose the product factor windows of* Z *and the series tail windows of* K *are consumed through the same finite observation-budget selector from chapter 782. Then the product-facing* $\mathsf{Real}^{\uparrow}$ *seal of* Z *and the series endpoint seal of* K *read the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, selector budget rows,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before either handoff is exposed.*

The handoff is a finite shared-selector route. It excludes host multiplication, selected infinite sums, quotient stream equality, countable choice, detached real carriers, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realseries shared selector handoff`

Proof. Start with the CauchyProduct side. The shared-selector commutation theorem Definition 464.138 places the product-facing regular handoff and real seal on one route through the stream windows, dyadic product rows, selector rows, transports, continuations, provenance, and local names.

Now read the RealSeries side through Definition 1581.13 and Definition 1581.37. These results make the finite-tail consumer surface visible only through partial-sum windows, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the shared tail threshold, the endpoint seal, and the structural rows.

The common selector hypothesis identifies the selector budget rows used by the product windows with the selector budget rows used by the series tail windows. Apply Definition 464.132 to lock the product windows to that selector, Definition 464.135 to keep the regular product handoff on the same route, and Definition 464.134 to place the product seal after the same budget ledger.

Finally use the RealSeries endpoint route Definition 1581.40. The endpoint seal of K is downstream of the same finite tail windows, regular readback, dyadic ledger, threshold, transports, continuations, provenance, and local names. Since neither accepted packet contains host multiplication, an infinite-sum selector, quotient stream equality, countable choice, a detached real carrier, or ambient completeness data, the shared-selector handoff exposes none of those objects. □

Theorem 464.140 (CauchyProduct shared-selector budget totality)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y,$ be an accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet, and let $K = (T, S, W, Q, D, M, E_R, H_R, C_R, P_R, N_R)$ be an accepted $\mathsf{RealSeries}^{\uparrow}$ packet. Suppose the product factor windows of Z and the series tail windows of K are consumed through one accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ route*

$$B, S_B, W_B, D_B, R_B, E_B, H_B, C_B, P_B, N_B.$$

Then every product-facing or series-facing consumer that reaches a $\mathsf{Real}^{\uparrow}$ endpoint must read the displayed CauchyProduct rows, the selector rows, the $\mathsf{StreamName}^{\uparrow}$ window rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ product and tolerance observations, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, the RealSeries endpoint route, the componentwise hsame *transports, displayed* Cont *replays,* Pkg *provenance rows, and local* NameCert *rows. No remaining endpoint route is available outside this finite budget.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared selector budget totality`

Proof. First use the distributive lock Definition 464.132. It binds the two CauchyProduct factor windows to the single selector route before the dyadic product row and regular handoff can be consumed.

Next use the real budget multiplication boundary Definition 464.134. This places every product-facing $\mathsf{Real}^{\uparrow}$ seal downstream of the same stream windows, dyadic product rows, selector ledger, regular handoff, transport, replay, provenance, and local naming rows.

Finally apply the RealSeries shared-selector handoff Definition 464.139. The series endpoint reads the same finite selector route and therefore has no second budget, selected infinite-sum route, quotient stream equality, detached real carrier, or ambient completeness witness. The listed rows exhaust the shared-selector product route. □

Theorem 464.141 (CauchyProduct dyadic product-window inversion)**.** *For an accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N)$ whose factor windows are consumed through a shared $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ route, any accepted read of the $\mathsf{DyadicRatCore}^{\uparrow}$ product observations Q_X, Q_Y determines the preceding product-window route*

$$W_X, W_Y, M_X, M_Y, Q_X, Q_Y$$

inside the same selector budget. The inversion is only the BEDC packet readback: it recovers the displayed $\mathsf{StreamName}^{\uparrow}$ windows, dyadic product observations, selector rows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It does not recover host factors, a quotient stream equality, selected real values, or a parameter-equivalence transport.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket dyadic product window selector certificate`
↪ variant `BEDC.Derived.CauchyProductUp.CauchyProductPacket dyadic product window inversion`

Proof. Start with the window product non-escape theorem used by Definition 464.132. A dyadic product observation in an accepted packet is downstream of the displayed factor windows and radius rows; it is not a detached product token.

Now attach the shared selector. The distributive lock places the factor windows and product observations inside one finite observation-budget selector route. Reading Q_X, Q_Y therefore also reads the selector rows and the same componentwise transports, continuations, provenance, and local names.

Finally use Definition 464.135. The regular-sequence product seal is downstream of exactly these rows, so the inversion recovers only the finite BEDC product-window route. No host factorization, quotient equality, selected real value, or parametric carrier transport is part of the recovered data. □

Theorem 464.142 (CauchyProduct RealSeries budget exhaustion)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P$ be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet and $K = (T, S, W, Q, D, M, E_R, H_R, C_R, P_R, N_R)$ an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet whose windows are read through the same finite observation-budget selector. The combined product-and-series endpoint route is exhausted by the CauchyProduct displayed rows, the RealSeries tail and endpoint rows, the selector rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product and tolerance observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *endpoint seals,* hsame, Cont, Pkg*, and local* NameCert *data. In particular, no consumer can add a second selector, an infinite-sum choice row, host multiplication, quotient stream equality, detached real carrier, trichotomy, or ambient completeness after the shared budget is spent.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real series budget exhaustion`

Proof. First build the product side from the shared-selector budget totality of Definition 464.140. The product-facing route already contains the displayed CauchyProduct rows, selector rows, stream windows, dyadic product observations, regular handoff, and real seal.

Next read the product-window inversion of Definition 464.141. Any dyadic product observation consumed by the route points back to the same factor-window and selector budget, so there is no hidden second window family.

Finally compose with the RealSeries handoff Definition 464.139. The series endpoint seal is downstream of the same finite tail windows, dyadic tolerance ledger, regular readback, transports, continuations, provenance, and local names. Thus the combined route has no coordinate left for a second selector, chosen infinite sum, host multiplication, quotient equality, detached real carrier, order trichotomy, or ambient completeness principle. □

Theorem 464.143 (CauchyProduct root-budget public export)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E,$ be an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet and $K = (T, S, W, Q, D, M, E_R, H_R, C_R, P_R, N_R)$ an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet whose windows are read through the same finite observation-budget selector. The public export consumed by* $\mathsf{RealSeries}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal readers is the single shared-selector route obtained by composing the budget-selector distributive lock, the shared-selector* $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal commutation, the shared-selector budget totality, and the RealSeries budget exhaustion route. Thus every seal-facing read passes through the displayed CauchyProduct rows, RealSeries tail rows, selector budget rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product and tolerance observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *endpoint seals,* hsame, Cont, Pkg*, and local* NameCert *data, with no second selector, host multiplication, quotient stream equality, detached real carrier, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root budget public export`

Proof. First expose the CauchyProduct side. The budget-selector distributive lock of Definition 464.132 binds both factor-window reads to one finite selector route before any product-facing consumer reaches the product rows.

Next pass through the regular and real seal interface. The shared-selector commutation theorem Definition 464.138 places the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal on the same displayed route.

Now apply shared-selector budget totality from Definition 464.140. It states that product-facing and series-facing endpoint consumers spend the same selector budget and read the same stream, dyadic, transport, replay, provenance, and naming rows.

Finally use Definition 464.142. The RealSeries endpoint route is exhausted by those same rows, so the public export is exactly the shared root-budget route and no unlisted analytic coordinate remains available. □

Theorem 464.144 (CauchyProduct RealSeries selector refinement boundary)**.** *Let $Z = (X, Y, W_X, W_Y, M_X, M_Y,$ be an accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet and $K = (T, S, W, Q, D, M, E_R, H_R, C_R, P_R, N_R)$ an accepted $\mathsf{RealSeries}^{\uparrow}$ packet whose windows pass through one accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ route. Any refinement of the selector budget that preserves the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows preserves both the product-facing $\mathsf{Real}^{\uparrow}$ seal and the series endpoint seal. The refinement exposes no second selector, quotient stream equality, selected infinite sum, host multiplication, detached real carrier, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket realseries selector refinement boundary`

Proof. Start with the budget-selector distributive lock of Definition 464.132. It binds the CauchyProduct factor windows and dyadic product observations to the same finite selector route before either endpoint can be read.

Next use the shared-selector $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ seal commutation of Definition 464.138. Preserving the displayed stream, dyadic, regular-sequence, real, transport, replay, provenance, and naming rows therefore preserves the product-facing real seal.

Now apply the RealSeries shared-selector handoff of Definition 464.139. The series endpoint consumes the same selector route, so the preserved rows also preserve the series endpoint seal.

Finally use the root-budget public export of Definition 464.143. A selector refinement only replays the same finite rows before either endpoint handoff is consumed, so it has no coordinate for a second selector, quotient stream equality, selected infinite sum, host multiplication, detached real carrier, trichotomy, or ambient completeness. □

Theorem 464.145 (CauchyProduct root regular product carrier row)**.** *For every accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet, the paired $\mathsf{RegularCauchyName}^{\uparrow}$ source rows, paired $\mathsf{StreamName}^{\uparrow}$ finite-window rows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rows, and $\mathsf{RegSeqRat}^{\uparrow}$ product classifier row form one finite BHist product carrier row. In displayed coordinates the carrier route is*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

where X, Y are the two regular Cauchy source rows, W_X, W_Y are the paired finite windows, M_X, M_Y, Q_X, Q_Y are the dyadic radius and observation rows, P is the product row, and E is the regular-rational classifier row. The row is local to the product packet and carries no host multiplication, quotient stream, selected limit, independent real carrier, or ambient completeness witness. The windows W_X, W_Y are read through chapter 126, the rows M_X, M_Y, Q_X, Q_Y through chapter 291, the classifier row E through chapter 301, and the seal boundary through chapter 38.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product carrier row`

Proof. Project the accepted carrier of Definition 464.1. Its source coordinates expose X, Y before the paired finite windows W_X, W_Y, and the dyadic rows M_X, M_Y, Q_X, Q_Y are read before the product row.

Use the StreamName handoff and the product-observation seal obligations. They place the product observation P and the classifier E downstream of the displayed finite windows and dyadic ledger rows.

Finally read the support rows. The componentwise hsame rows, displayed Cont routes, Pkg provenance, and local NameCert row keep the whole product route inside one finite BHist carrier row, so none of the excluded analytic objects has an exported coordinate. □

Theorem 464.146 (CauchyProduct root shared-window product classifier)**.** *For every accepted* CauchyProduct$^{\uparrow}$ *packet, the product classifier row E is locked to the paired finite windows W_X, W_Y and to the displayed* hsame *and* Cont *route*

$$W_X, W_Y \to M_X, M_Y, Q_X, Q_Y \to P \to E.$$

Any consumer that reads E must first read the two StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *radius and observation rows, and the local support rows H, C, L, N. It cannot feed the classifier from a host multiplication operation, quotient stream product, selected limit, independent real carrier, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root shared window product classifier certificate`

Proof. Begin with the carrier row of Definition 464.145. The row lists $W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E$ in the order needed by the product read.

Apply the root source window totality and root budget product lock. These statements force the two finite windows and the dyadic budget rows to be consumed before the product row P and classifier row E are visible.

The displayed hsame and Cont rows then transport the same product route to the local naming surface. Since the accepted packet contains no host multiplication row, quotient stream, selected limit, independent real carrier, or ambient completeness coordinate, the classifier is locked to the finite windows and the displayed route. □

Theorem 464.147 (CauchyProduct root Real seal product ledger)**.** *Any* Real$^{\uparrow}$*-facing product seal read of an accepted* CauchyProduct$^{\uparrow}$ *packet consumes the local* Pkg *and* NameCert *ledger before it reaches the seal. The read factors through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

uses the product classifier route of Definition 464.146, and then reaches the Real$^{\uparrow}$ *seal only through the displayed product ledger. It cannot use host multiplication, quotient streams, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root real seal product ledger certificate`

Proof. First build the product carrier row using Definition 464.145. This supplies the two regular Cauchy sources, paired windows, dyadic ledger rows, product row, classifier row, and support rows as one finite inventory.

Next apply the shared-window product classifier theorem. The classifier row E is already downstream of the paired windows, dyadic observations, product row, and displayed hsame and Cont route.

Finally route the classifier to the Real-facing seal. The local Pkg and NameCert ledger rows L, N are part of the displayed packet before the seal is consumed, and the existing Real seal root boundary excludes host multiplication, quotient streams, selected limits, and ambient completeness from the consumer surface. □

Theorem 464.148 (CauchyProduct root regular product budget seal exhaustion)**.** *Any* Real$^{\uparrow}$*-facing product consumer that uses the root regular product carrier of an accepted* CauchyProduct$^{\uparrow}$ *packet must pass through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and through the displayed RealObservationBudget$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *seal route before a* Real$^{\uparrow}$ *seal is exposed. The consumer cannot reach the seal from host multiplication, quotient streams, selected limits, an independent real carrier, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product budget seal inventory certificate`

Proof. Begin with the root regular product carrier row. It places the paired regular Cauchy sources, finite windows, dyadic budget rows, product observation, classifier row, transport rows, route rows, package ledger, and local naming row in one finite inventory.

Apply the shared-window product classifier. The classifier E is visible only after the paired windows W_X, W_Y, the dyadic radius and observation rows M_X, M_Y, Q_X, Q_Y, and the product row P have been consumed along the displayed route.

Use the Real seal product ledger for the final boundary. It sends the Real$^{\uparrow}$-facing consumer through the local Pkg and NameCert ledger rows L, N before the seal is visible, and it identifies the exposed seal as the displayed RealObservationBudget$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ route rather than an ambient real object.

The excluded sources have no coordinate in the carrier row. Therefore the Real-facing product seal is exhausted by the displayed budgeted product route. □

Lemma 464.149 (CauchyProduct root regular product downstream coverage)**.** *Every accepted local consumer downstream of the root regular product route of an accepted* CauchyProduct$^{\uparrow}$ *packet is exhausted by the finite alternatives, ledgers, and coverage rows displayed in this file. Concretely, the consumer can read only the carrier and classifier rows*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

the Real$^{\uparrow}$*-facing product ledger through* L, N*, or the displayed* RealObservationBudget$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *seal route attached to the same product inventory. Host multiplication, quotient streams, selected product limits, independent regular-sequence or real carriers, and ambient completeness witnesses remain outside the accepted BEDC packet.*

Proof. Start with the carrier row of Definition 464.145. It displays the whole root regular product packet as one finite BHist inventory: the two RegularCauchyName$^{\uparrow}$ source rows, the paired StreamName$^{\uparrow}$ windows, the dyadic radius and observation rows, the product row, the RegSeqRat$^{\uparrow}$ classifier, and the support rows H, C, L, N. Hence every local consumer first meets one of the listed carrier, window, dyadic, product, classifier, transport, continuation, ledger, or naming coordinates.

The classifier alternatives are then fixed by Definition 464.146. Any read of E must pass through the paired windows, the dyadic rows, the product row P, and the displayed hsame and Cont route. Thus a downstream consumer cannot introduce a second classifier input, a host multiplication operation, a quotient-stream product, or a selected product limit at the classifier stage.

For a seal-facing read, apply Definition 464.147. The $\mathsf{Real}^{\uparrow}$ boundary consumes the local Pkg and NameCert ledger rows before the seal is visible, so the downstream route is still the same finite inventory with L, N as the product ledger.

Finally apply Definition 464.148. It identifies every $\mathsf{Real}^{\uparrow}$-facing product consumer using the root regular product carrier with the displayed $\mathsf{RealObservationBudget}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ seal route, and it excludes host multiplication, quotient streams, selected limits, an independent real carrier, and ambient completeness from the seal surface. Combining the carrier, classifier, ledger, and budget-seal rows accounts for all accepted local downstream reads, while the excluded alternatives occur in no coordinate of the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product downstream coverage`

Theorem 464.150 (CauchyProduct regular product window exhaustion)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing product consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet must consume both* $\mathsf{StreamName}^{\uparrow}$ *windows, both* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and budget rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *product classifier, the local* Pkg *and* NameCert *ledger, and the displayed* $\mathsf{RealObservationBudget}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal route before any product read is exported. Equivalently, the consumer factors through*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and then through the budgeted Real seal route. The packet provides no host multiplication, quotient stream product, selected limit, independent real carrier, or ambient completeness coordinate from which a product read could be exported earlier.

Proof. First apply the root regular product carrier row. It supplies the two regular Cauchy sources, the two finite windows, the two dyadic observation and budget pairs, the product row, the classifier row, and the local support rows as one finite inventory.

Next apply the shared-window product classifier. The classifier E is downstream of both windows W_X, W_Y and both dyadic rows M_X, M_Y, Q_X, Q_Y, so no product read can use E before the displayed finite-window route has been consumed.

Finally apply the budget seal exhaustion theorem. That theorem routes every $\mathsf{Real}^{\uparrow}$-facing product consumer through the local Pkg and NameCert ledger and through the displayed $\mathsf{RealObservationBudget}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ seal boundary before the Real seal is visible.

The excluded sources have no coordinate in the carrier row. Hence host multiplication, quotient streams, selected limits, independent real carriers, and ambient completeness are unavailable to the consumer, and the regular product window route is exhaustive. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket regular product window exhaustion`

Theorem 464.151 (Cauchy product regular-tail diagonal window lock)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet be read by a product-facing consumer through two regular product tails whose finite observations are served by shared* $\mathsf{StreamName}^{\uparrow}$ *windows. Before the consumer can inspect the local* NameCert *row, the read must pass through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the same diagonal-tail selector row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal:*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N \longmapsto \mathsf{DiagonalTailSelector}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The packet has no alternate route through quotient streams, a selected product limit, host equality, host multiplication, choice, or ambient Real completeness.

Proof. Start with the regular product window exhaustion theorem. It forces every Real-facing product consumer through the paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observation and budget rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier, local Pkg and $\mathsf{NameCert}$ ledger, and the displayed Real seal route before any product read is exported.

Next use the root regular product budget separation theorem. The two product tails share the same displayed tolerance budget and readback route, so the consumer cannot split one tail through a second dyadic ledger or an independent regular-rational classifier.

Attach the diagonal selector route using the product budget-triangle selector compatibility and shared-tail selector factorization rows. Those sibling rows identify the $\mathsf{DiagonalTailSelector}^{\uparrow}$ contribution with the same finite StreamName, DyadicRatCore, RegSeqRat, and Real surface already present in the product packet.

Finally compare the route with the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal boundaries. The seal-facing consumer reaches the local $\mathsf{NameCert}$ only after the displayed product inventory, diagonal-tail selector, and Real seal have been read. Since none of the cited rows contains quotient streams, selected limits, host equality, host multiplication, choice data, or ambient completeness, no such coordinate can feed the product-facing consumer. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductRegularTailDiagonalWindowLock`

Theorem 464.152 (CauchyProduct root RealSeries handoff exhaustion)**.** *Every* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer fed by a* $\mathsf{RealSeries}^{\uparrow}$ *root series surface consumes paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, the* $\mathsf{RealSeries}^{\uparrow}$ *shared tail threshold, and the local* Pkg *and* $\mathsf{NameCert}$ *ledger before any Real-facing product seal is exported. The visible route factors through the local product inventory*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and through the RealSeries rows

$$T, S, W, Q, D, M, E, H, C, P, N$$

from Definition 1581.19. It has no coordinate for host multiplication, quotient stream product, selected infinite sums, independent real carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root series handoff exhaustion`

Proof. Begin with the root regular product carrier row. It supplies $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L,$ as the only product inventory visible to the consumer.

Apply the shared-window product classifier. The classifier row E is downstream of both $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ budget and observation rows, and the product observation row, so a product read cannot be fed by a host multiplication operation or quotient stream product.

Use the budget seal exhaustion theorem for the Real-facing boundary. It sends every seal read through the local Pkg and $\mathsf{NameCert}$ rows before any $\mathsf{Real}^{\uparrow}$ seal is exported.

Finally apply Definition 1581.19. The RealSeries handoff supplies only the finite rows $T, S, W, Q, D, M, E, H, C,$ with the shared threshold row M already fixed before endpoint sealing. Thus the product consumer receives a finite paired-window route, not a selected infinite sum, independent real carrier, or ambient completeness coordinate. □

Theorem 464.153 (CauchyProduct root regular product public export)**.** *Every public* CauchyProduct$^{\uparrow}$ *export over the root regular-product packet factors through paired* RegularCauchyName$^{\uparrow}$ *sources, paired* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *budget rows, the* RegSeqRat$^{\uparrow}$ *product classifier,* RealSeries$^{\uparrow}$ *shared-tail rows, the local* Pkg *and* NameCert *ledger, and the terminal* Real$^{\uparrow}$ *seal route. In displayed coordinates the export consumes*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and the RealSeries$^{\uparrow}$ *rows*

$$T, S, W, Q, D, M, E, H, C, P, N$$

before any public product read is available. The export has no coordinate for host multiplication, quotient streams, selected product limits, independent real carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product public export`

Proof. First apply the root regular product carrier row. It places the paired RegularCauchyName$^{\uparrow}$ sources, paired finite windows, dyadic budget rows, product observation, classifier row, transport, replay, provenance, and local naming in one finite carrier inventory.

Next use the shared-window product classifier. The classifier row E can be read only after the two StreamName$^{\uparrow}$ windows and the DyadicRatCore$^{\uparrow}$ radius and observation rows have been consumed, so no host multiplication or quotient-stream product can feed the public export.

Then apply the regular product budget seal exhaustion and the Real seal product ledger. Together they route the public read through L, N and through the displayed RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ seal boundary before any Real-facing product value is exposed.

Finally use the root RealSeries handoff exhaustion. The shared-tail rows $T, S, W, Q, D, M, E, H, C, P, N$ are the only series-facing input to the product packet, and their threshold row is fixed before endpoint sealing. Combining these carrier, classifier, budget, ledger, and handoff rows gives the public export route and excludes the stated analytic alternatives by field exhaustion. □

Theorem 464.154 (CauchyProduct root regular product budget separation)**.** *Any accepted* CauchyProduct$^{\uparrow}$ *root regular product consumer that reaches a* Real$^{\uparrow}$*-facing seal factors first through the paired* StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *budget rows, the* RegSeqRat$^{\uparrow}$ *classifier row, the* RealObservationBudget$^{\uparrow}$ *rows, the displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. The Real seal is consumed only after that route has exposed*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

No host multiplication, quotient stream product, selected limit, independent real carrier, or ambient completeness coordinate is available to the consumer before the seal is read.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product budget separation certificate`

↪ variant `BEDC.Derived.CauchyProductUp.CauchyProductPacket root regular product budget separation`

Proof. First apply the root regular product carrier row. It supplies the paired regular Cauchy sources, finite windows, dyadic budget rows, product observation row, classifier row, and structural support rows as one finite carrier inventory.

Next apply the shared-window product classifier. This locks E behind the two StreamName$^{\uparrow}$ windows and the DyadicRatCore$^{\uparrow}$ radius and observation rows, so the classifier cannot be fed by a host multiplication operation or a quotient stream product.

Then apply the Real seal product ledger. The Pkg and NameCert rows are consumed before the seal, and the route passes through the displayed RealObservationBudget$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ handoff rather than through an independent real carrier.

Finally apply budget seal exhaustion. The exhausted route has no coordinate for selected limits or ambient completeness, so every accepted Real-facing product consumer remains separated from the excluded sources until the displayed finite budget route reaches the seal. □

Theorem 464.155 (CauchyProduct regular factor source minimality)**.** *Any* Real$^{\uparrow}$*-facing product consumer over the root regular product packet of an accepted* CauchyProduct$^{\uparrow}$ *carrier must display both* RegularCauchyName$^{\uparrow}$ *source rows, both* StreamName$^{\uparrow}$ *windows, all* DyadicRatCore$^{\uparrow}$ *budget and observation rows, the* RegSeqRat$^{\uparrow}$ *product classifier, the* RealObservationBudget$^{\uparrow}$ *seal rows, local* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *support before a* Real$^{\uparrow}$ *seal is exported. In coordinates the required source surface is*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N.$$

No proper subroute omitting either regular factor, either finite window, any dyadic budget row, the classifier row, the RealObservationBudget rows, or the local support rows can export the product seal.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket regular factor source minimality certificate`

Proof. Begin with the carrier row of Definition 464.145. It displays $X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L,$ as one finite product inventory, with both regular factors and both finite windows present before the product observation and classifier rows.

Apply Definition 464.146. The classifier E is locked behind the paired StreamName$^{\uparrow}$ windows, the dyadic budget and observation rows, the product row P, and the displayed hsame and Cont route, so omitting any of those rows destroys the classifier input rather than producing a smaller product read.

Next use Definition 464.148. Every Real-facing product consumer passes through the displayed RealObservationBudget$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ seal route before the Real$^{\uparrow}$ boundary is visible.

Finally combine Definition 464.152 with Definition 464.153. The RealSeries handoff and public export consume the same paired-factor inventory and the local Pkg and NameCert rows before exposing any public product seal. Since the accepted carrier contains no host multiplication, quotient stream product, selected limit, independent real carrier, or ambient completeness coordinate, the displayed route is source-minimal for regular-factor Real-facing product reads. □

Theorem 464.156 (CauchyProduct mature public factorization)**.** *Every public* CauchyProduct$^{\uparrow}$ *read over an accepted root regular product packet factors through regular product source rows, shared-tail modulus rows, finite dyadic product budgets,* RegSeqRat$^{\uparrow}$ *readback,* RealSeries$^{\uparrow}$ *handoff, and* Real$^{\uparrow}$ *seal rows before any consumer can read a product claim. In displayed coordinates the public route consumes*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

from the product packet, then the RealSeries$^{\uparrow}$ *handoff rows*

$$T, S, W, Q, D, M, E, H, C, P, N$$

from the shared-tail surface, and finally the local Pkg *and* NameCert *ledger at the seal boundary. No public product claim is available from host multiplication, quotient stream product, selected limits, independent real carriers, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket mature public factorization`

Proof. First build the regular product source surface. The root carrier theorem supplies the paired regular Cauchy source rows, shared finite windows, dyadic radius and observation rows, product row, classifier row, transport, replay, provenance, and local naming as one finite product inventory.

Next apply the shared-window classifier and the Real seal product ledger. These route any read of the product classifier through the paired windows and dyadic budgets, then through the local Pkg and NameCert rows before a Real-facing seal is visible.

Use budget exhaustion and downstream coverage for the consumer boundary. They identify every accepted Real-facing product consumer with the same finite dyadic product budget route and exclude host multiplication, quotient streams, selected limits, independent real carriers, and ambient completeness from the carrier surface.

Finally apply the RealSeries handoff and source-minimality theorems. The shared-tail rows $T, S, W, Q, D, M, E, H, C, P, N$ are consumed before the public product export, and the two regular factors, finite windows, dyadic budget rows, readback, ledger, and seal rows cannot be omitted without destroying the displayed product route. Therefore every public $\mathsf{CauchyProduct}^{\uparrow}$ read factors through exactly the stated finite source, budget, handoff, and seal surfaces. □

RegSeqRat–Real sibling route. The $\mathsf{CauchyProduct}^{\uparrow}$ carrier consumes the $\mathsf{RegSeqRat}^{\uparrow}$ regular-sequence carrier of chapter 301 and sends its product readback to the $\mathsf{Real}^{\uparrow}$ seal surface of chapter 38. This dependency is the finite product route through scheduled windows and dyadic ledgers, not an imported series convergence theorem.

Theorem 464.157 (CauchyProduct Real seal upstream dependency)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *Real-seal read consumed by a completion-facing downstream chapter first consumes the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, the product budget ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal surface. In displayed coordinates, the upstream route is the finite packet*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with any accepted observation-budget rows when the consumer uses a budget packet. The route exports no quotient stream, selected product limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal upstream dependency`

Proof. Start with paired-window source admission from Definition 464.100 and Definition 464.82. These rows force the two source names X, Y to enter the product packet through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y.

Next use budget route exactness from Definition 464.101 and Definition 464.66. The dyadic observations Q_X, Q_Y, source-radius rows M_X, M_Y, product row P, and ledger row L form the budget route before the classifier can be read.

Apply the paired-window ledger classifier results Definition 464.146 and Definition 464.88. They identify the $\mathsf{RegSeqRat}^{\uparrow}$ row E as downstream of the same finite windows, dyadic observations, and product ledger.

Finally use the root Real seal obligations and consumer exhaustion supplied by Definition 464.147, Definition 464.124, and Definition 464.85. The completion-facing consumer therefore reaches the $\mathsf{Real}^{\uparrow}$ seal only through the displayed upstream route and cannot import any excluded analytic object. □

Theorem 464.158 (CauchyProduct Real seal budget factorization)**.** *Every completion-facing product read of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *Real-seal packet factors through the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_X, W_Y*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations* Q_X, Q_Y*, the product budget rows* M_X, M_Y, P, L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* E*, the* $\mathsf{Real}^{\uparrow}$ *seal, and any accepted observation-budget rows when such rows are present. In displayed coordinates the factorization is exhausted by*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with those accepted budget rows. No quotient stream, selected product limit, host multiplication primitive, trichotomy row, independent real carrier, or ambient completeness row can bypass this finite factorization route.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal upstream budget factorization`

Proof. First apply Definition 464.157. It already places every completion-facing Real-seal read downstream of the paired StreamName windows, dyadic observations, product budget ledger, RegSeqRat classifier, Real seal surface, and accepted observation-budget rows when the consumer asks for such rows.

Next use the Real–RegSeqRat–StreamName handoff of Definition 464.159. That handoff identifies the same rows as the ordered visibility chain seen by Real-completion-facing consumers, so the factorization has the displayed source windows, observation rows, product rows, classifier row, transport rows, replay rows, ledger row, and local naming row as its whole public surface.

Finally combine the local budget locks used in the upstream proof: Definition 464.101, Definition 464.66, and Definition 464.132. These locks prevent a consumer from reading a product seal before the product budget rows and accepted observation-budget rows have been displayed.

The cited upstream dependency and handoff theorems already exclude quotient streams, selected product limits, host multiplication primitives, trichotomy rows, independent real carriers, and ambient completeness data. Therefore no completion-facing product read can bypass the finite budget factorization route stated above. □

Theorem 464.159 (Real RegSeqRat StreamName CauchyProduct handoff)**.** *An accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *route consumed by a Real-completion-facing chapter is equivalent, as a finite BEDC handoff surface, to the ordered visibility chain consisting of paired* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic observations, the product budget ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal. In coordinates, the handoff surface is exhausted by*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the accepted observation-budget rows when a budget consumer asks for them. It exports no quotient stream, host multiplication primitive, trichotomy row, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real regseqrat streamname handoff`

Proof. First read the ordered source surface. The paired-window totality of Definition 464.100 forces the two source names X, Y to be visible only through the paired $\mathsf{StreamName}^{\uparrow}$ windows W_X, W_Y before any product consumer can inspect the downstream rows.

Next read the budget and classifier surface. The budget product lock of Definition 464.101 places the dyadic observations, source-radius rows, product row, and ledger row before the classifier, and Definition 464.88 identifies the resulting E row as the finite $\mathsf{RegSeqRat}^{\uparrow}$ classifier consumed by the Real route.

Finally apply the upstream dependency theorem Definition 464.157. It states that a completion-facing Real-seal read reaches the $\mathsf{Real}^{\uparrow}$ seal only through exactly those prior rows, so the ordered chain is both sufficient for the handoff and exhaustive for accepted consumers. Coordinate exhaustion gives the exclusions: no quotient stream, host multiplication, trichotomy, or ambient completeness coordinate is present. □

Theorem 464.160 (CauchyProduct LimitSeal sibling dependency)**.** *Every accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *Real-seal route that reaches the* $\mathsf{Real}^{\uparrow}$ *seal passes through the same finite rows consumed by the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary of chapter 386: paired* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic observations, the product budget ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier, transport rows, route rows, package provenance, and local* $\mathsf{NameCert}$ *rows. The sibling dependency supplies a horizontal handoff edge only; it imports no selected product limit, quotient stream, host multiplication primitive, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket limitseal sibling dependency`

Proof. Begin with the upstream dependency theorem. It places every completion-facing Real-seal read of a $\mathsf{CauchyProduct}^{\uparrow}$ packet on the paired windows, dyadic observations, product budget ledger, classifier row, and displayed structural rows before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Next use the Real–RegSeqRat–StreamName handoff. That handoff identifies the same finite ordered surface as the route seen by a downstream Real-completion consumer.

Read this finite surface against the $\mathsf{CauchyLimitSeal}^{\uparrow}$ compatibility boundary named in chapter 386. That boundary consumes $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, dyadic observation, $\mathsf{Real}^{\uparrow}$, transport, route, package, and local naming rows, so the product route and the limit-seal route meet on those rows.

The conclusion is only a sibling dependency. Since neither cited product theorem exports a quotient stream, selected product limit, host multiplication primitive, or ambient completeness row, the horizontal edge adds no analytic coordinate beyond the displayed finite BEDC surface. □

Theorem 464.161 (CauchyProduct shared-selector Real seal stability)**.** *Let two accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packets read their factor windows through the same* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *route. Suppose the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic radius rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and* $\mathsf{Real}^{\uparrow}$ *seal boundary agree componentwise on the public packet rows. Then every multiplication-facing* $\mathsf{Real}^{\uparrow}$ *seal consumer of the two packets reads the same finite product-route surface*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the shared selector rows. The shared selector can be reused without changing the product seal surface, and the route exports no host multiplication, quotient-stream product, selected limit, trichotomy row, or ambient completeness datum.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket shared selector real seal stability`

Proof. First apply the budget-selector distributive lock Definition 464.132. It identifies both factor-window reads with one finite selector route and one displayed product-route certificate.

Next use the Real budget multiplication boundary Definition 464.134. A multiplication-facing $\mathsf{Real}^{\uparrow}$ seal read is available only after the shared windows, dyadic radius rows, regular-rational handoff, selector rows, transports, continuations, provenance, and local naming rows have been displayed.

Finally apply the RegSeqRat product seal coverage theorem Definition 464.135. Agreement on the displayed StreamName, dyadic, classifier, and Real seal boundary rows therefore forces both consumers to inspect the same finite product-route surface. The cited coverage theorem also lists the missing coordinates, so no host multiplication, quotient product, selected limit, trichotomy row, or ambient completeness object can be introduced by reusing the selector. □

Theorem 464.162 (CauchyProduct Real seal selector coherence)**.** *Let two accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packets use the same finite observation-budget selector. Suppose their paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and observation rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and* $\mathsf{Real}^{\uparrow}$ *seal boundary agree componentwise on the public packet rows. Then every Real-seal consumer and every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *sibling-seal consumer reads the same finite route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the shared selector rows. Selector reuse is coherent with the full L10 sibling seal route, and it exports no host multiplication, quotient stream, selected limit, trichotomy row, choice witness, or ambient completeness datum.

Proof. First read the Real-seal side through Definition 464.157. That theorem exhausts every completion-facing Real-seal read by the displayed route, together with accepted observation-budget rows, before the $\mathsf{Real}^{\uparrow}$ boundary can be consumed.

Next read the sibling-seal side through Definition 464.160. It identifies the $\mathsf{CauchyLimitSeal}^{\uparrow}$ compatibility boundary as a horizontal consumer of the same StreamName, dyadic, classifier, Real, transport, route, package, and local naming rows, so the sibling edge has no independent analytic coordinate.

Finally apply Definition 464.161. Since the two accepted packets share one selector and agree on the public window, radius, observation, classifier, and Real boundary rows, selector reuse fixes the finite product-route surface for both packets. Combining the three cited theorems gives the same route for Real-seal and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers, with no host multiplication, quotient streams, selected limits, trichotomy, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal selector coherence`

Theorem 464.163 (CauchyProduct limit-seal selector coherence)**.** *Let an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *packet reuse the same finite observation-budget selector for its Real-seal route and for the sibling route consumed through chapter 386. Then the selector remains coherent on the displayed product route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

and on the shared StreamName, dyadic, RegSeqRat, Real-boundary, transport, route, package, and local NameCert rows. The sibling edge adds no quotient stream, selected product limit, host multiplication coordinate, choice witness, or ambient completeness datum.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket limitseal selector coherence`

Proof. Read the Real-seal route through Definition 464.162. The selector already fixes the finite product-route surface for every Real-seal consumer that agrees on the public StreamName, dyadic, classifier, and Real-boundary rows.

Now read the sibling consumer through chapter 386. The CauchyLimitSeal boundary consumes the same finite stream windows, dyadic observations, regular classifier row, Real boundary, transports, routes, package provenance, and local naming rows.

Thus the reused selector is not a new analytic coordinate. It is a shared finite witness for the same displayed product-route surface, and the sibling handoff cannot export quotient streams, selected product limits, host multiplication, choice, or ambient completeness because none of those rows is present in either route. □

Lemma 464.164 (CauchyProduct Real seal selector minimal witness)**.** *For a completion-facing consumer of an accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *Real-seal packet, the local admission and consumption claim is witnessed by the single displayed product route*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N$$

together with the shared $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows when the consumer asks for a selector. If a consumer presents a larger reading of the same Real-seal claim, then its accepted part factors through this displayed witness: the source admission uses only* X, Y, W_X, W_Y*, the budget and product admission uses only* M_X, M_Y, Q_X, Q_Y, P, L*, the classifier admission uses only* E*, and the seal consumption uses only* H, C, N *with the displayed Real boundary. No unlisted quotient stream, selected product limit, host multiplication primitive, trichotomy row, independent real carrier, ambient choice object, or ambient completeness witness can serve as an additional admission coordinate.*

Proof. Start with the upstream dependency theorem Definition 464.157. It states that every completion-facing Real-seal read first consumes the paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the product budget ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, and the $\mathsf{Real}^{\uparrow}$ seal surface. In coordinates this is exactly the displayed packet, with accepted observation-budget rows only when a budget consumer asks for them.

Now apply the shared-selector Real-seal stability theorem Definition 464.161. When two accepted product packets read their factor windows through the same $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ route and agree on the public window, radius, classifier, and Real-boundary rows, every multiplication-facing Real-seal consumer reads the same finite product-route surface together with the shared selector rows. Thus selector reuse does not create a second product seal surface; it only names the finite selector rows already attached to the displayed route.

Combine the two readings by sorting the consumer-facing coordinates by their role in the displayed packet. The source part is forced to factor through X, Y, W_X, W_Y by the upstream source admission. The budget and product part is forced to factor through M_X, M_Y, Q_X, Q_Y, P, L before the classifier is visible. The classifier part is forced to factor through E, and the final seal handoff is carried by H, C, N together with the Real boundary already listed by the two cited theorems. The shared selector rows may be consumed as witness rows for the common observation-budget route, but they do not add any unlisted analytic object.

Therefore any larger accepted reading has the same consumer-facing content after projection to the displayed product route and the shared selector rows. Since both cited theorems explicitly exclude quotient streams, selected limits, host multiplication primitives, trichotomy rows, independent real carriers, ambient choice objects, and ambient completeness witnesses from the exported route, none of those objects can be used as an additional admission or consumption coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket real seal selector minimal witness`

Theorem 464.165 (CauchyProduct L10 dependency lattice)**.** *Every accepted* CauchyProduct$^{\uparrow}$ *packet that reaches the* Real$^{\uparrow}$ *seal factors through the displayed L10 dependency lattice: the paired* StreamName$^{\uparrow}$ *windows of chapter 126, the* DyadicRatCore$^{\uparrow}$ *observation and radius rows of chapter 291, the* RegSeqRat$^{\uparrow}$ *classifier row of chapter 301, the* Real$^{\uparrow}$ *seal of chapter 38, and the local* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. In coordinates the product-facing lattice is*

$$X, Y, W_X, W_Y, M_X, M_Y, Q_X, Q_Y, P, E, H, C, L, N,$$

with any accepted observation-budget selector rows attached only as finite budget witnesses. The lattice exports no quotient streams, host multiplication primitives, selected limits, independent real carriers, trichotomy rows, countable-choice witnesses, or ambient completeness principles.

Lean4 ✓ `BEDC.Derived.CauchyProductUp.CauchyProductPacket l10 dependency lattice`

Proof. Start from the product carrier row of Definition 464.145. It puts the two source names, paired finite windows, dyadic radius and observation rows, product observation, classifier row, transport rows, continuation rows, provenance row, and local naming row into one finite BHist carrier.

Next pass to the Real seal by Definition 464.157. The seal-facing consumer can reach Real$^{\uparrow}$ only after the displayed StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and local support rows have already been read.

Use Definition 464.135 for the regular-rational side. It shows that any product-seal consumer using a finite observation-budget selector remains on the same windows, dyadic rows, classifier row, selector rows, transports, continuations, provenance, and local NameCert rows, rather than acquiring an additional carrier.

Combining these three rows gives the stated L10 lattice. The cited carrier, Real-seal, and RegSeqRat coverage theorems list exactly the available coordinates, so quotient streams, host multiplication primitives, selected limits, independent real carriers, trichotomy rows, countable choice, and ambient completeness have no coordinate from which they could enter the accepted packet. □

Closure status for NameCert$_{\text{CauchyProduct}^{\uparrow}}$**.**

Constructive story: A CauchyProduct$^{\uparrow}$ packet is generated from two RegularCauchyName$^{\uparrow}$ source rows, paired StreamName$^{\uparrow}$ finite-window schedules, dyadic-radius rows, DyadicRatCore$^{\uparrow}$ observations, a pointwise product row, a RegSeqRat$^{\uparrow}$ classifier, displayed hsame transports, Cont routes, a finite ledger, Real$^{\uparrow}$ seal handoff, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{CauchyProduct}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 464.47, Definition 464.139, Definition 464.138, and Definition 464.140 bind the product-observation surface to its shared-selector RealSeries handoff, RegSeqRat–Real seal commutation, and budget-totality consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyProduct}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, quotient streams, selected limits, trichotomy, host multiplication as a primitive, independent real carriers, or bridge checking.

Upgrade path: Bridge checking remains the verification-side step; the theory-side route already includes the shared-selector RealSeries handoff, RegSeqRat–Real seal commutation, and budget-totality rows.

Chapter 465

A Concrete Naming Certificate for $\mathsf{RegularCauchyScale}^{\uparrow}$

The $\mathsf{RegularCauchyScale}^{\uparrow}$ interface names the finite rational-scaling surface for a regular rational Cauchy name. It sits with $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, and $\mathsf{RegularCauchyProduct}^{\uparrow}$: the packet displays a rational scalar row, one regular source name, aligned stream windows, dyadic endpoint rows for the scalar and source, a scaled dyadic endpoint row, a finite error-budget ledger, regular rational readback, transports, continuations, package provenance, and local naming data.

The packet is a constructive arithmetic certificate. It is not a quotient of streams, not a selected host limit, not a host real equality assertion, and not an ambient completeness theorem.

The $\mathsf{RegularCauchyScale}^{\uparrow}$ packet consumes the same finite arithmetic route as chapter 417, chapter 459, and chapter 461. Its scalar endpoint tolerance is recorded through chapter 385 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff.

Definition 465.1 (RegularCauchyScale carrier)**.** *A* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *scalar row whose rational value is used in every scaled endpoint;*
2. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose scheduled rational observations are scaled by Q;*
3. *W is the aligned* $\mathsf{StreamName}^{\uparrow}$ *window used to read the source observation at the requested precision and index;*
4. *D_Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row for the scalar Q at the common exponent used by the packet;*
5. *D_A is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row read from the source window W;*
6. *D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row for the scaled value $D_Q \cdot D_A$;*
7. *E is the finite* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger recording the scalar endpoint, source tolerance, scalar magnitude bound, scaled endpoint error, and output tolerance assigned to D;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose scheduled point row is D with regularity evidence supplied by E;*

9. *H stores componentwise* hsame *transports for the scalar, source name, stream window, dyadic endpoint rows, finite budget ledger, and readback row;*
10. *C stores the displayed* Cont *routes for reading the scalar, reading the source window, forming the dyadic scaled endpoint, checking the budget, and exposing the readback;*
11. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyScale}^{\uparrow}}$ *row.*

The carrier is finite rational-scaling data for regular Cauchy names. It stores no quotient stream, no selected limit, no host real equality, and no ambient completeness object.

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleTasteGate single carrier alignment`

Theorem 465.2 (RegularCauchyScale NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyScale}^{\uparrow}}$ *surface for the carrier of Definition 465.1 consists of five finite obligations:*

1. *carrier admission: the accepted object is the displayed* BHist *packet S with a* $\mathsf{Rat}^{\uparrow}$ *scalar, one* $\mathsf{RegSeqRat}^{\uparrow}$ *source, an aligned* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic endpoint rows, a finite scale-error ledger, transports, continuations, provenance, and local naming data;*
2. *classifier transport: classification transports the scalar row, source row, window row, endpoint rows, finite budget ledger, and readback row componentwise through the stored* hsame *and* Cont *rows;*
3. *stability: replacing the scalar or source by a classified row preserves the displayed window route and recomputes the same kind of dyadic scaled endpoint row;*
4. *ledger exactness: the output tolerance row is the finite scale-error budget recorded in E, including the scalar endpoint, source tolerance, scalar magnitude bound, and scaled endpoint error;*
5. *downstream handoff: every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer first reads R together with Q, A, W, D_Q, D_A, D, E, H, C, P, and N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier namecert obligations`

Proof. Carrier admission is projection from the tuple in Definition 465.1. The rational scalar, regular source, stream window, endpoint rows, finite ledger, transport rows, continuation routes, provenance row, and local naming row are explicit coordinates of S.

Classifier transport is componentwise. The $\mathsf{Rat}^{\uparrow}$ scalar row transports the rational coefficient, the $\mathsf{RegSeqRat}^{\uparrow}$ source row transports its scheduled observations, the $\mathsf{StreamName}^{\uparrow}$ row transports the source window read, and the $\mathsf{DyadicRatCore}^{\uparrow}$ rows transport the scalar endpoint, source endpoint, and scaled endpoint calculation.

The finite ledger records the scalar endpoint, the source tolerance, the scalar magnitude bound, the scaled endpoint error, and the output tolerance before the readback is exposed. Since the downstream handoff is the readback row R and its displayed provenance, no omitted coordinate can add quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 465.3 (RegularCauchyScale source factorization)**.** *For an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier $S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$, every scale read factors through the* $\mathsf{Rat}^{\uparrow}$ *scalar row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row A, the* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows D_Q, D_A, and D, and the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row E before the readback row R is exposed.*

Thus the packet presents rational scaling as a finite source factorization: scalar, source schedule, window, endpoint calculation, and budget row are all visible before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the scaled name.*

Proof. Project the carrier of Definition 465.1. The scalar row Q, source row A, stream window W, endpoint rows D_Q, D_A, D, and ledger row E are explicit coordinates of the finite packet.

Read a scale request through those coordinates in order. The scalar endpoint is supplied by Q through D_Q, the source observation is supplied by A through W and D_A, and the scaled endpoint is the displayed dyadic row D.

The budget row E then records the scalar endpoint, source tolerance, scalar magnitude bound, scaled endpoint error, and output tolerance. Only after this finite ledger is present does the packet expose the regular readback row R.

Consequently every consumer-facing scale read is factored through the displayed rational source data and finite budget rows rather than through a completed real scalar action or a quotient stream. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier source factorization`

Theorem 465.4 (RegularCauchyScale classifier route exhaustion)**.** *Classifier transport for* $\mathsf{RegularCauchyScale}^{\uparrow}$ *is exhausted by componentwise* hsame *transport and* Cont *replay over the displayed tuple*

$$(Q, A, W, D_Q, D_A, D, E, R, H, C, P, N).$$

There is no classifier route for a hidden scalar, an unrecorded source stream, a quotient representative, a selected limit, or an ambient completion object.

Proof. Start with the tuple in Definition 465.1. The visible classifier inputs are the scalar, source, window, endpoint rows, ledger, readback, transport row, continuation row, provenance row, and local naming row.

Apply the classifier transport clause of Definition 465.2. It moves those rows componentwise through the stored hsame data and replays consumer access only through the displayed Cont routes.

The source factorization of Definition 465.3 leaves no remaining coordinate to classify. Every scale read has already passed through Q, A, W, D_Q, D_A, D, E before reaching R.

Thus classifier routing is exhausted by componentwise transport and continuation replay over the displayed tuple, with no hidden stream, selected limit, quotient representative, or completion object available to inspect. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier classifier route exhaustion`

Theorem 465.5 (RegularCauchyScale RealUp scalar non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *scalar consumer of an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet sees only the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the finite scale ledger* E *together with the visible scalar, source, window, endpoint, transport, continuation, provenance, and naming rows. It does not receive a completed real scalar multiplication, a quotient stream equality, a selected limit, or an ambient vector-space operation.*

Proof. Factor the real-facing read through Definition 465.3. Before R is exposed, the packet displays the rational scalar, source row, aligned window, dyadic endpoint rows, and finite budget ledger.

Use Definition 465.4 to control transport. A classified replacement can replay only the displayed hsame and Cont routes over the same tuple.

Finally apply the existing real seal boundary in Definition 465.11. A $\mathsf{Real}^{\uparrow}$ consumer reads the regular rational sequence surface and its visible provenance, not a separate completed-real scalar field.

Therefore the scalar operation does not escape the finite $\mathsf{RegSeqRat}^{\uparrow}$ readback and budget ledger into quotient equality, selected limits, ambient completeness, or a host vector-space operation. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier real scalar non escape`

Theorem 465.6 (RegularCauchyScale carrier admission obligation)**.** *For an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier* $S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$*, carrier admission is exactly the displayed finite* BHist *packet of Definition 465.1. The admitted rows are the* $\mathsf{Rat}^{\uparrow}$ *scalar row* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source* A*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scalar endpoint* D_Q*, source endpoint* D_A*, and scaled endpoint* D*, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *scale ledger* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

No admitted coordinate is a quotient stream, host real equality, selected limit, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier scale carrier obligation`

Proof. Project the tuple in Definition 465.1. The scalar, source, stream window, endpoint rows, scale ledger, readback, transport, continuation, provenance, and naming rows are all explicit coordinates.

The $\mathsf{Rat}^{\uparrow}$ scalar row and $\mathsf{RegSeqRat}^{\uparrow}$ source row are read through the aligned $\mathsf{StreamName}^{\uparrow}$ window. The $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows display the scalar endpoint, source endpoint, and scaled endpoint used by the finite budget ledger.

The rows H, C, P, and N only transport and expose those displayed coordinates. Thus carrier admission has no coordinate for quotient streams, host real equality, selected limits, or ambient completeness. □

Theorem 465.7 (RegularCauchyScale classifier stability obligation)**.** *Classifier stability for* $\mathsf{RegularCauchyScale}^{\uparrow}$ *transports the* $\mathsf{Rat}^{\uparrow}$ *scalar row,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *scale ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* Pkg *provenance, and local* NameCert *row componentwise through the displayed* hsame *and* Cont *data.*

The transported packet remains the same kind of finite rational-scaling surface: scalar row, source window, dyadic scaled endpoint, finite scale ledger, and regular readback. It introduces no quotient stream, host real equality, selected limit, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier classifier stability obligation`

Proof. Start with the transport ledger H. It relates the scalar, source name, stream window, dyadic endpoint rows, finite budget ledger, and readback row componentwise.

Read those transported rows through the continuation routes C. The scalar endpoint and source endpoint still form the displayed scaled endpoint row, and the finite scale ledger still records the tolerance budget consumed by the readback.

The package provenance P and local naming row N bound the transported surface. Since the classifier sees only these finite rows, it cannot add quotient streams, host real equality, selected limits, or ambient completeness. □

Theorem 465.8 (RegularCauchyScale scale ledger exactness obligation)**.** *The scale ledger exactness obligation for* $\mathsf{RegularCauchyScale}^{\uparrow}$ *says that every read of the output row* R *must pass through the* $\mathsf{Rat}^{\uparrow}$ *scalar row* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source* A*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D_Q*,* D_A*, and* D*, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger* E*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*.*

The ledger must account for the scalar endpoint, source tolerance, scalar magnitude bound, scaled endpoint error, and output tolerance before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the scaled regular Cauchy name.*

Proof. Read the scalar and source rows first. The $\mathsf{Rat}^{\uparrow}$ scalar supplies D_Q, and the $\mathsf{RegSeqRat}^{\uparrow}$ source is observed through the $\mathsf{StreamName}^{\uparrow}$ window to supply D_A.

Use the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row D for the scaled endpoint. The finite ledger E records the scalar endpoint, source tolerance, scalar magnitude bound, scaled endpoint error, and output tolerance assigned to that endpoint.

Finally route the readback through H, C, P, and N. Because the ledger exactness route is fully displayed before R reaches a downstream consumer, no quotient stream, host real equality, selected limit, or ambient completeness can be used in its place. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier ledger exactness obligation`

Theorem 465.9 (RegularCauchyScale window budget)**.** *Given an accepted* $\mathsf{Rat}^{\uparrow}$ *scalar row* Q *and an accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* A *in a* $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet, every precision request reads the aligned* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the scalar endpoint row* D_Q*, and the source endpoint row* D_A*. The packet displays a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *for* $D_Q \cdot D_A$ *and a finite ledger* E *whose output tolerance is the recorded scale-error budget for the source tolerance and scalar magnitude bound.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier window budget`

Proof. Read the precision request through the scalar and source coordinates. The scalar row Q supplies the rational endpoint D_Q, while the source row A is observed through the aligned stream window W to produce D_A.

Apply the displayed dyadic multiplication row of $\mathsf{DyadicRatCore}^{\uparrow}$ to D_Q and D_A. The result is the scaled endpoint row D, still inside the same finite carrier.

Finally read the finite budget ledger E. Its entries record the source tolerance, the scalar magnitude bound, and the endpoint error assigned to the output request, so the budget is a finite BEDC ledger rather than an appeal to a completed real scalar action. □

Theorem 465.10 (RegularCauchyScale RegSeqRat handoff)**.** *The budgeted rational-scaling windows of Definition 465.9 repack as the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *of Definition 465.1. A consumer of* R *reads only the scalar row, source window, dyadic scaled endpoint row, finite scale-error ledger, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier regseqrat handoff`

Proof. Start with the finite window budget. For each requested precision, the packet exposes the scalar endpoint, the source endpoint, the scaled dyadic endpoint, and the scale-error row that bounds the output tolerance.

Repack those displayed rows as a regular rational sequence readback. The point row of R is D, and the regularity evidence visible to the consumer is the scale ledger E routed through the same scalar and source windows.

The transport and continuation rows are inherited from the carrier. Because the repacking uses only coordinates already listed in Definition 465.1, it introduces no host sequence quotient, selected limit, host real equality, or completed-real scalar operation. □

Theorem 465.11 (RegularCauchyScale RealUp seal boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet reads only the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *together with the visible provenance of the scalar, source name, aligned window, dyadic endpoint rows, finite scale-error ledger,* hsame *transports,* Cont *routes,* Pkg *row, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyScale}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier realup seal boundary`

Proof. Factor the real-facing read through the regular readback. By Definition 465.10, R is already the repacking of the budgeted rational-scaling windows.

The remaining consumer-visible data are exactly the provenance rows attached to that repacking: the scalar row, source name, source schedule, dyadic endpoint rows, scale-error ledger, transports, continuations, package row, and local naming row.

The carrier has no independent $\mathsf{Real}^{\uparrow}$ equality field and no completion witness outside the readback. Therefore the real seal can consume only the displayed regular readback and its visible provenance. □

Theorem 465.12 (RegularCauchyScale obligation scope package)**.** *For every accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N),$$

the carrier admission, classifier stability, scale ledger exactness, window budget, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal boundary jointly supply the obligation surface over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package exports exactly the rational scalar row, source row, aligned window, dyadic endpoint rows, finite scale-error ledger, regular readback, transport routes, continuation routes, provenance, and local naming data. It exports no quotient stream, host real equality, selected limit, ambient completeness object, or completed real scalar operation.

Proof. Start with carrier admission in Definition 465.6. It identifies every coordinate of the accepted packet and fixes the dependency surface over the scalar, source, window, endpoint, ledger, readback, transport, continuation, provenance, and naming rows.

Use classifier stability from Definition 465.7 and scale ledger exactness from Definition 465.8. These clauses keep transported packets componentwise inside the same finite surface and require every output read to pass through $Q, A, W, D_Q, D_A, D, E, H, C, P, N$.

Apply the window budget theorem Definition 465.9 and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff theorem Definition 465.10. They package the scaled dyadic endpoint and finite budget as the readback row R.

Finally use the $\mathsf{Real}^{\uparrow}$ seal boundary in Definition 465.11. A real-facing consumer receives only R and the displayed provenance rows, so the obligation package has no coordinate from which quotient streams, host equality, selected limits, ambient completeness, or a completed real scalar operation could be read. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier obligation scope package`

Theorem 465.13 (RegularCauchyScale scoped Real-consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of a scaled regular Cauchy packet*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

factors through the scalar row Q*, source row* A*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D_Q, D_A, D*,* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*. The scoped consumer boundary exports no quotient stream, selected limit, host real equality, ambient completeness object, or completed real scalar multiplication.*

Proof. Begin with source factorization from Definition 465.3. It routes every scaled read through the scalar row Q, source row A, aligned window W, endpoint rows D_Q, D_A, D, and finite budget ledger E.

Use classifier route exhaustion from Definition 465.4 and ledger exactness from Definition 465.8. These results keep the transported classifier route inside the same finite packet and require the budget ledger before any output read is exported.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff theorem Definition 465.10. The scaled dyadic endpoints and finite ledger are repacked as the readback row R under the displayed transport, continuation, provenance, and local naming rows.

Finally use the $\mathsf{Real}^{\uparrow}$ seal boundary in Definition 465.11. The real-facing consumer receives only R and its visible provenance, so the scoped boundary cannot export quotient streams, selected limits, host real equality, ambient completeness, or a completed real scalar operation. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier scoped real consumer boundary`

Theorem 465.14 (RegularCauchyScale cofinal budget refinement)**.** *Let*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet. Suppose a refined finite scale ledger* E' *is displayed over the same scalar row* Q*, source row* A*, stream window* W*, dyadic endpoint rows* D_Q, D_A, D*, and dyadic endpoint route, and suppose the refined output tolerance is transported by the same componentwise* hsame *rows* H *and displayed* Cont *routes* C*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer of the refined packet factors through*

$$(Q, A, W, D_Q, D_A, D, E', R, H, C, P, N).$$

The refinement introduces no completed real scalar multiplication, quotient stream, selected limit, host real equality, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier cofinal budget refinement`

Proof. Start with source factorization from Definition 465.3. The refined ledger is displayed over the same scalar row, source row, stream window, dyadic endpoint rows, and endpoint route, so the read still begins with Q, A, W, D_Q, D_A, D and the finite budget row now named E'.

Use classifier route exhaustion and ledger exactness from Definition 465.4 and Definition 465.8. These clauses keep the refined tolerance inside the same componentwise hsame and Cont rows, so no classifier route can replace the finite ledger by an ambient estimate.

Apply the window budget theorem Definition 465.9. The refined ledger is cofinal for the same finite window surface, hence every displayed scale-tail read has the same source, endpoint, and transport coordinates with E' in the budget coordinate.

Finally use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff in Definition 465.10 and the $\mathsf{Real}^{\uparrow}$ seal boundary in Definition 465.11. The regular consumer receives the refined finite readback surface, and the real-facing consumer factors through that readback and its visible provenance, leaving no coordinate for quotient streams, selected limits, host equality, ambient completeness, or completed real scalar multiplication. □

Theorem 465.15 (RegularCauchyScale tail-window carrier obligation)**.** *For every accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N),$$

scale-tail admission is exactly the displayed tail-window budget of the packet. The admitted scale tail reads the $\mathsf{Rat}^{\uparrow}$ *scalar* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source* A*, the aligned* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D_Q*,* D_A*, and* D*, and the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger* E *before exposing the readback row* R*.*

No scale-tail row is admitted without the source factorization, classifier route exhaustion, window budget, and $\mathsf{RegSeqRat}^{\uparrow}$ *handoff displayed by Definition 465.3, Definition 465.4, Definition 465.9, and Definition 465.10.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier tail window carrier obligation`

Proof. Project the carrier tuple. Its scalar, source, window, endpoint, budget, and readback rows are precisely the coordinates needed for a scale-tail read.

Use source factorization to order the read: Q and A are read through W, the dyadic endpoint rows D_Q and D_A form the scaled endpoint D, and the finite budget E records the output tolerance.

The RegSeqRat handoff then exposes R only after this budget is present. Since classifier route exhaustion leaves no hidden stream or selected limit to inspect, scale-tail admission is exactly the displayed tail-window budget. □

Theorem 465.16 (RegularCauchyScale scalar classifier stability obligation)**.** *Scalar classifier stability for* $\mathsf{RegularCauchyScale}^{\uparrow}$ *is inherited through the* $\mathsf{RegSeqRat}^{\uparrow}$ *scalar route. If a classified packet transports the scalar row* Q*, source row* A*, window* W*, endpoint rows* D_Q, D_A, D*, budget ledger* E*, and readback row* R*, then the transported packet remains a finite rational-scaling carrier with the same source factorization, classifier route exhaustion, window budget, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff shape.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier scalar classifier stability obligation`

Proof. Begin with classifier route exhaustion. It says that classified packets can move only by the componentwise hsame ledger and the displayed Cont routes over the tuple $Q, A, W, D_Q, D_A, D, E, R, H, C, P, N$.

The source factorization and window budget are stable under that route because the scalar endpoint, source endpoint, scaled endpoint, and finite error ledger are all transported as named rows. No classified replacement can bypass W or replace E by an ambient real estimate.

Finally the $\mathsf{RegSeqRat}^{\uparrow}$ handoff consumes the transported budgeted windows in the same way as the original packet. The resulting readback remains a finite regular rational sequence surface, not a quotient stream or selected host limit. □

Theorem 465.17 (RegularCauchyScale unit-scalar boundary)**.** *For an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N),$$

suppose the $\mathsf{Rat}^{\uparrow}$ *scalar row* Q *is the displayed unit row and* D_Q *is the unit dyadic endpoint on the same denominator ledger as the source endpoint row* D_A*. Then every displayed scale read exposes the same source endpoint route as* D_A *before the finite budget ledger* E *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *are exposed.*

Thus the packet supplies the identity-scalar boundary through the finite BHist *packet, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It does not use a host vector-space law, quotient stream equality, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the source factorization in Definition 465.3. Every scale read already factors through Q, A, W, D_Q, D_A, D, E before it exposes R.

Use the window budget theorem Definition 465.9. Since Q is the displayed unit scalar row and D_Q is the unit endpoint on the same denominator ledger as D_A, the dyadic scaled endpoint route records the same source endpoint route that supplied D_A.

Classifier stability from Definition 465.16 keeps this route inside the componentwise hsame ledger and the displayed Cont rows. No classified replacement can replace the unit endpoint ledger by a host scalar action.

Finally apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff in Definition 465.10 and the real seal boundary in Definition 465.11. The readback R and any $\mathsf{Real}^{\uparrow}$ consumer see only the displayed budgeted route and its provenance, so the identity-scalar boundary is a finite naming boundary rather than a quotient, selected-limit, completeness, or vector-space assertion. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier unit scalar boundary`

Theorem 465.18 (RegularCauchyScale RealUp seal handoff obligation)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal handoff for* $\mathsf{RegularCauchyScale}^{\uparrow}$ *consumes no row outside the displayed surface*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

together with the scalar row Q*, finite budget* E*, and carrier history in* BHist*. In particular, every real-facing consumer factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *and the visible provenance of Definition 465.1.*

Proof. Read the real-facing route through the RealUp seal boundary. The consumer receives R together with the scalar, source window, dyadic endpoint rows, finite budget ledger, transports, continuations, provenance, and naming row.

The tail-window carrier obligation supplies the admitted budgeted windows, and scalar classifier stability shows that classified packets preserve the same handoff shape. Thus the seal receives the displayed $\mathsf{RegSeqRat}^{\uparrow}$ route rather than a host-level scalar action.

Every coordinate consumed by the seal is named in the carrier or in the listed dependency surface. There is no row for quotient stream equality, selected limit, ambient completeness, or a real-vector-space law outside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier real seal handoff obligation`

Theorem 465.19 (RegularCauchyScale rational scalar bridge boundary)**.** *Every bridge-facing consumer of a scaled regular Cauchy packet*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

must expose the $\mathsf{Rat}^{\uparrow}$ *scalar row* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* A*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D_Q, D_A, D*, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$

readback R, the componentwise hsame *transport H, the displayed* Cont *replay C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyScale}^{\uparrow}}$ *row N before it reads the* $\mathsf{Real}^{\uparrow}$ *seal. The bridge boundary therefore exports rational scaling only through the displayed finite packet, not through quotient streams, selected limits, host real equality, ambient completeness, or an external real-scalar law.*

Proof. Start with source factorization in Definition 465.3. It orders every scaled read through the scalar row Q, the source row A, the aligned stream window W, and the dyadic endpoint rows D_Q, D_A, D.

Next apply the window budget theorem Definition 465.9. The finite error ledger E is exposed before the scale read is admitted, so the bridge-facing consumer receives a displayed tolerance account rather than a hidden real estimate.

Use the $\mathsf{RegSeqRat}^{\uparrow}$ handoff in Definition 465.10. The readback row R is available only after the scalar, source, window, endpoint, and budget rows have been read, and classifier stability keeps the same route under componentwise hsame transport and displayed Cont replay.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal boundary Definition 465.11. The seal can be consumed only through R, H, C, P, N, so the rational-scalar bridge boundary is exactly the displayed carrier route and no quotient, selected-limit, ambient-completeness, or host scalar-action coordinate remains. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier rational scalar bridge boundary`

Theorem 465.20 (RegularCauchyScale shared-tail normalization)**.** *Let*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet. Suppose two real-facing consumer reads use the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row A, the same* $\mathsf{StreamName}^{\uparrow}$ *tail-window row W, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale ledger D_Q, D_A, D, the same unit-scalar boundary when Q is the displayed unit row, and the same* $\mathsf{Real}^{\uparrow}$ *seal route through the readback row R. Then the two reads normalize to the same displayed shared-tail route through*

$$A,\ W,\ D_Q,\ D_A,\ D,\ E,\ R,\ H,\ C,\ P,\ N.$$

The normalization is internal to the finite carrier history, componentwise hsame *transport, displayed* Cont *rows,* Pkg *provenance, and local* NameCert *row. It does not appeal to quotient streams, host real equality, hidden limits, ambient completeness, or any unlisted real-scalar law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier shared tail normalization`

Proof. First use the tail-window carrier obligation in Definition 465.15. Since both consumers read the same source row, tail-window row, dyadic scale ledger, and finite budget ledger, their admitted scale-tail routes have the same displayed carrier coordinates before readback.

Next apply scalar classifier stability from Definition 465.16. Any classified transport of either read remains inside the same componentwise hsame ledger and the same displayed Cont rows, so no consumer can replace the shared tail by an ambient stream quotient or by a host-level scalar action.

When the scalar row is the displayed unit row, use the unit-scalar boundary in Definition 465.17. It identifies the unit read with the source endpoint route already present in D_A, before the budget ledger and readback row are exposed.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal handoff obligation in Definition 465.18. The real-facing surface consumes only the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback R and the visible provenance of the same packet,

hence both consumer reads present one shared-tail route and no hidden limit, host real equality, ambient completeness, quotient stream equality, or external real-scalar law is available to distinguish them. □

Theorem 465.21 (RegularCauchyScale scoped closure route)**.** *Every accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *carrier*

$$S = (Q, A, W, D_Q, D_A, D, E, R, H, C, P, N)$$

is scoped by the displayed dependency tuple over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *Every scale read factors through the scalar row Q, source row A, window row W, scalar endpoint D_Q, source endpoint D_A, scaled endpoint D, finite budget E, regular readback R, componentwise transport H, continuation replay C, provenance P, and local naming row N. The scoped route exports no quotient stream, selected limit, host real equality, ambient completeness object, or real-vector-space law.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier scoped closure route`

Proof. First use the window budget theorem Definition 465.9. It binds every precision request to the scalar row, source row, stream window, dyadic scalar endpoint, dyadic source endpoint, scaled dyadic endpoint, and finite error-budget ledger.

Next apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff theorem Definition 465.10. The budgeted endpoint rows repack as the readback row R under the same displayed transport, continuation, provenance, and local naming rows.

Use the $\mathsf{Real}^{\uparrow}$ seal boundary Definition 465.11 together with shared-tail normalization Definition 465.20. These results keep real-facing consumers on the same readback route and normalize common tails through the finite carrier history rather than through quotient equality or host-level limits.

Finally apply the NameCert obligations theorem Definition 465.2. Its carrier admission, classifier transport, ledger exactness, and downstream handoff rows exhaust the visible dependency surface, so no quotient stream, selected limit, host real equality, ambient completeness, or real-vector-space law remains outside the scoped tuple. □

Theorem 465.22 (RegularCauchyScale public consumer boundary)**.** *The named public consumer boundary of an accepted* $\mathsf{RegularCauchyScale}^{\uparrow}$ *packet is the finite route*

$$Q, A, W, D_Q, D_A, D, E, R, H, C, P, N.$$

Every $\mathsf{RegSeqRat}^{\uparrow}$ *consumer reads the same source row A, aligned* $\mathsf{StreamName}^{\uparrow}$ *window W, dyadic endpoint rows D_Q, D_A, D, finite budget E, and regular readback R before exporting scale data. Every* $\mathsf{Real}^{\uparrow}$ *consumer then reads the same readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before the terminal seal is visible. Thus the public interface is the displayed scale packet itself, not quotient stream equality, selected limits, host real equality, ambient completeness, or an external real-vector-space law.*

Proof. Project the accepted carrier and apply Definition 465.21. It already orders the scalar, source, window, dyadic endpoint, finite-budget, regular-readback, transport, replay, provenance, and local naming rows.

For $\mathsf{RegSeqRat}^{\uparrow}$ consumers, use Definition 465.10. The readback row R can be consumed only after the displayed source, window, endpoint, and budget rows have been preserved.

For $\mathsf{Real}^{\uparrow}$ consumers, use Definition 465.11. The terminal seal reads the same R, H, C, P, N rows and adds no coordinate for quotient streams, selected limits, host equality, ambient completeness, or external scalar laws. □

Lean4 ✓ `BEDC.Derived.RegularCauchyScaleUp.RegularCauchyScaleCarrier public consumer boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyScale}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyScale}^{\uparrow}$ packet is generated from a $\mathsf{Rat}^{\uparrow}$ scalar row, a $\mathsf{RegSeqRat}^{\uparrow}$ source row, an aligned $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ scalar/source/scaled endpoint rows, a finite $\mathsf{FiniteErrorBudget}^{\uparrow}$ scale ledger, displayed hsame transports, Cont routes, Pkg provenance, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyScale}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 465.21 binds the finite scalar, window, dyadic endpoint, finite-budget, regular-readback, transport, continuation, provenance, Real seal, and NameCert rows for rational scaling of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyScale}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, selected limits, host real equality, ambient completeness, real-vector-space laws, or bridge checking.

Upgrade path: No further paper-theory closure is claimed before Lean formalization or bridge checking for the displayed scalar, window, dyadic endpoint, finite-budget, regular-readback, and Real seal rows.

Chapter 466

A Concrete Naming Certificate for RegularCauchyTriangleBound$^{\uparrow}$

The RegularCauchyTriangleBound$^{\uparrow}$ interface packages the finite three-leg estimate used by the diagonal regularity step in the Real completeness capstone. It keeps the two diagonal observations, the shared intermediate observation, the two regular-sequence tail bounds, and the middle Cauchy-modulus bound in one NameCert packet. The packet is a displayed estimate surface; it is not a quotient stream, not a selected limit, and not an ambient completeness principle.

The triangle packet consumes the finite windows of chapter 126, the regular readbacks of chapter 301, the dyadic tolerance rows of chapter 291, and the downstream diagonal consumer surface of chapter 646. These four rows determine the L10 dependency route on which the three-leg estimate is later read by completion-facing consumers.

The packet consumes the shared-threshold and middle-meet surfaces of chapter 560 and chapter 462 before handing a finite three-leg bound to Real$^{\uparrow}$ consumers. Those sibling rows supply the L10 threshold lattice used by the triangle route; the present carrier only synchronizes their reads into one diagonal estimate packet.

Definition 466.1 (RegularCauchyTriangleBound carrier)**.** *A* RegularCauchyTriangleBound$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$$

with the following displayed rows:

1. *D_m and D_n are the two* StreamName$^{\uparrow}$ *diagonal observation rows for the requested indices m and n;*
2. *I_m and I_n are the two intermediate* StreamName$^{\uparrow}$ *rows used to compare the diagonal observations through a common Cauchy-modulus surface;*
3. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger carrying the two endpoint tolerances $1/(m+1)$ and $1/(n+1)$ and the summed output bound;*
4. *A stores the two* RegSeqRat$^{\uparrow}$ *regularity reads that bound the first and third triangle legs;*
5. *B stores the middle* CauchyModulusMeet$^{\uparrow}$ *read that bounds the comparison between the two intermediate observations;*
6. *H stores componentwise* hsame *transports for diagonal rows, intermediate rows, tolerance rows, regularity reads, middle-modulus reads, provenance, and the local name;*

7. *C stores the displayed* Cont *routes for diagonal inspection, intermediate-point selection, regularity reads, middle-modulus reads, dyadic tolerance addition, and local NameCert comparison;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting every displayed row to a common diagonal, intermediate-point, and dyadic tolerance surface. The carrier has no coordinate for quotient stream equality, selected limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTriangleBoundUp.RegularCauchyTriangleBoundTasteGate single carrier alignment decode encode`

Theorem 466.2 (RegularCauchyTriangleBound NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleBound}}$ *surface for the carrier of Definition 466.1 consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet $T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$;*
2. *classifier transport to a common* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *surface;*
3. *regular-leg stability for the two endpoint triangle legs stored in A;*
4. *middle-leg exactness for the shared Cauchy-modulus comparison stored in B;*
5. *ledger rows for dyadic tolerance addition,* Cont *routes,* Pkg *provenance,* $\mathsf{Real}^{\uparrow}$ *handoff reads, and the local naming certificate.*

Proof. Carrier admission is projection from Definition 466.1. The rows display the two diagonal observations, two intermediate observations, dyadic tolerance ledger, endpoint regularity reads, middle-modulus read, transports, continuations, provenance, and local name.

Classifier transport uses H to move those rows to a common finite surface. Once transported, the routes in C are the only ways to inspect diagonal rows, choose intermediate rows, read endpoint regularity, read the middle modulus, and add dyadic tolerances.

The two endpoint estimates are stored in A, while the middle comparison is stored in B. Since the carrier has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness, the obligations are exhausted by the five displayed rows. □

Theorem 466.3 (RegularCauchyTriangleBound diagonal regularity)**.** *For an accepted* RegularCauchyTriangleBound *carrier $T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$, the two diagonal observations D_m and D_n are compared by the displayed three-leg route*

$$D_m \longrightarrow I_m \longrightarrow I_n \longrightarrow D_n$$

and the output tolerance is the dyadic sum recorded in Q from the two regular endpoint bounds in A and the middle Cauchy-modulus bound in B. Every $\mathsf{Real}^{\uparrow}$ *diagonal-regularity consumer reads this finite route before receiving the bound.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTriangleBoundUp.RegularCauchyTriangleBoundCarrier diagonal regular route`

Proof. Read the accepted carrier and project the four observation rows. The continuation row C supplies the route from D_m to I_m, from I_m to I_n, and from I_n to D_n.

The first and third legs are bounded by the two $\mathsf{RegSeqRat}^{\uparrow}$ regularity reads in A. The middle leg is bounded by the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ read in B, transported to the same intermediate surface by H.

Finally read the dyadic ledger Q. It records the two endpoint tolerances and the summed output tolerance, so the consumer receives exactly the three-leg diagonal estimate and no separate completed-real object. □

Theorem 466.4 (RegularCauchyTriangleBound endpoint-leg scope)**.** *For an accepted* RegularCauchyTriangleBound *carrier* $T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$*, the first comparison leg* $D_m \longrightarrow I_m$ *and the third comparison leg* $I_n \longrightarrow D_n$ *are exactly the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity reads stored in row* A*. No additional endpoint estimate, quotient-stream comparison, selected limit, host real equality, or ambient completeness datum participates in either endpoint leg.*

Proof. Project the carrier of Definition 466.1. Its row A is defined as the pair of $\mathsf{RegSeqRat}^{\uparrow}$ regularity reads, one for the m endpoint and one for the n endpoint, while the surrounding rows only transport, route, add tolerances, record provenance, and name the packet.

Apply the diagonal route of Definition 466.3. The route reads the first triangle leg from D_m to I_m and the third triangle leg from I_n to D_n before the dyadic output bound is reported, so the only endpoint estimates available to that route are the two reads already displayed in A.

The non-escape boundary of Definition 466.7 prevents any consumer-facing route from exporting quotient streams, selected limits, host real equality, ambient completeness, or completion witnesses. Thus the endpoint-leg scope is exactly row A. □

Theorem 466.5 (RegularCauchyTriangleBound middle-meet exactness)**.** *For an accepted* RegularCauchyTriangleBo *carrier* $T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$*, the middle comparison* $I_m \longrightarrow I_n$ *is exactly the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *read stored in row* B*, transported by the displayed* hsame *rows in* H *to the common intermediate surface. No quotient stream, selected limit, host real equality, or ambient completeness datum supplies a second middle comparison.*

Proof. Project the carrier of Definition 466.1. Row B is the middle $\mathsf{CauchyModulusMeet}^{\uparrow}$ read, and row H is the componentwise transport surface that aligns the intermediate observations and middle-modulus read.

Use Definition 466.3 to inspect the route $D_m \longrightarrow I_m \longrightarrow I_n \longrightarrow D_n$. The central arrow is the only place in the route where the two intermediate rows are compared, and the theorem identifies that bound as the shared middle Cauchy-modulus read in B.

By Definition 466.7, every consumer-facing route is exhausted by the finite displayed rows. Since the carrier has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness, row B is the exact middle-meet obligation. □

Theorem 466.6 (RegularCauchyTriangleBound tolerance-addition ledger)**.** *For an accepted* RegularCauchyTriangle *carrier* $T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N)$*, the output tolerance read by the diagonal-regularity consumer is precisely the* $\mathsf{DyadicRatCore}^{\uparrow}$ *sum row in* Q*, formed from the two endpoint tolerances certified by* A *and the middle tolerance certified by* B*. No selected limit, quotient-stream equality, host real equality, or ambient completeness datum changes the reported tolerance.*

Proof. Project the carrier of Definition 466.1. Row Q is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, row A supplies the two endpoint regularity tolerances, and row B supplies the middle Cauchy-modulus tolerance.

The diagonal-regularity statement in Definition 466.3 reads the three legs in order and then reads the dyadic sum recorded in Q. Therefore the reported output tolerance is not recomputed by a hidden completion object; it is the displayed ledger row attached to the finite packet.

The non-escape boundary of Definition 466.7 rules out exported quotient streams, selected limits, host real equality, ambient completeness proofs, and host completion witnesses. Hence the tolerance-addition obligation is precisely the $\mathsf{DyadicRatCore}^{\uparrow}$ row Q. □

Theorem 466.7 (RegularCauchyTriangleBound non-escape)**.** *Every consumer-facing route out of an accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *carrier factors through the finite rows* $D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N$. *No route exports a quotient stream, selected limit, ambient completeness proof, host real equality, or host completion witness.*

Proof. Take any consumer-facing route and project the carrier. The only mathematical rows are the diagonal observations, intermediate observations, dyadic tolerance ledger, endpoint regularity reads, and middle-modulus read.

The remaining rows transport, route, record provenance, and name the packet. Thus any read by a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer is exhausted by the displayed finite packet.

Because the carrier omits quotient streams, selected limits, ambient completeness proofs, host real equality, and host completion witnesses, none of those objects can be exported by a continuation route. □

Theorem 466.8 (RegularCauchyTriangleBound phase handoff)**.** *The* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *packet is the finite handoff surface from* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *into the Real completeness diagonal-regularity argument. A downstream* $\mathsf{Real}^{\uparrow}$ *consumer receives the transported three-leg route, the endpoint regularity reads, the middle-modulus read, the dyadic tolerance ledger, provenance, and the local* $\mathsf{NameCert}$ *row, and receives no additional limit-selection data.*

Proof. The dependency anchors in P name the dyadic arithmetic, stream schedule, regular rational readback, and shared-modulus rows consumed by the packet. Projecting the carrier exposes exactly those rows together with transports, continuations, provenance, and local naming.

The handoff to $\mathsf{Real}^{\uparrow}$ is through the diagonal-regularity route of Definition 466.3. That route first displays the three triangle legs and then reads the summed dyadic bound.

No downstream row is available for a selected limit, quotient stream, host real equality, or ambient completeness object. Therefore the phase handoff is precisely the finite three-leg estimate surface. □

Theorem 466.9 (RegularCauchyTriangleBound Real consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion consumer that uses a* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *packet reads the two endpoint regularity legs, the middle* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *meet, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-addition ledger before the* $\mathsf{Real}^{\uparrow}$ *seal handoff. The consumer route exports no quotient-stream equality, selected limits, host equality, or ambient completeness row.*

Proof. Start with Definition 466.3. It fixes the three-leg route through the two endpoint observations and the two intermediate rows.

Apply Definition 466.4 and Definition 466.5. These identify the endpoint regularity reads with row A and the middle modulus meet with row B after the displayed transports.

The tolerance row is supplied by Definition 466.6. Use Definition 466.7 to exclude quotient-stream equality, selected limits, host equality, and ambient completeness, then apply Definition 466.8 to present the finite packet to the $\mathsf{Real}^{\uparrow}$ seal handoff. □

Theorem 466.10 (RegularCauchyTriangleBound transported assembly)**.** *For an accepted* RegularCauchyTriangleB *carrier*

$$T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N),$$

any transported consumer route over T is exhausted by the endpoint-leg reads in A, the middle-meet read in B, the dyadic sum ledger Q, the transport and continuation routes H, C, and the package and naming rows P, N. The assembled route is the finite transported packet that a $\mathsf{Real}^{\uparrow}$ *diagonal-regularity or completion-seal consumer receives; it is not a single-theorem variant of one leg obligation, because it combines both endpoint legs, the middle leg, tolerance addition, non-escape, and the Real-phase handoff into one consumer package.*

Proof. Start with diagonal regularity from Definition 466.3. It fixes the visible route $D_m \longrightarrow I_m \longrightarrow I_n \longrightarrow D_n$ and makes the dyadic output bound a read from the same packet.

Apply endpoint-leg scope from Definition 466.4. The first and third legs of the route are exactly the two $\mathsf{RegSeqRat}^{\uparrow}$ regularity reads stored in A after the displayed transports.

Next apply middle-meet exactness from Definition 466.5. The central comparison is exactly the $\mathsf{CauchyModulusMeet}^{\uparrow}$ read stored in B, transported by H to the same intermediate surface.

The tolerance-addition ledger of Definition 466.6 supplies the summed dyadic output row Q. Non-escape from Definition 466.7 prevents quotient streams, selected limits, host real equality, ambient completeness, and completion witnesses from entering the transported route.

Finally use the phase handoff of Definition 466.8. The $\mathsf{Real}^{\uparrow}$ consumer receives the assembled finite packet through H, C, P, N, so the transported assembly is exactly the union of the three leg obligations, the dyadic ledger, and the Real-phase handoff. □

Theorem 466.11 (RegularCauchyTriangleBound completion-seal consumer exactness)**.** *For an accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *carrier*

$$T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N),$$

every $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion-seal consumer that invokes the triangle-bound packet factors through the diagonal route theorem, endpoint-leg scope, middle-meet exactness, tolerance-addition ledger, non-escape boundary, and phase handoff. The consumer receives exactly the displayed rows $D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N$ and receives no selected limit, quotient stream, host real equality, ambient completeness proof, or completion witness.*

Proof. First apply Definition 466.3. It fixes the route from D_m through I_m and I_n to D_n as the only diagonal comparison available to a completion-seal consumer.

Next apply Definition 466.4 and Definition 466.5. These identify the two endpoint legs with row A and the middle leg with row B after the displayed transports.

The tolerance read is then forced by Definition 466.6. Apply Definition 466.7 to exclude selected limits, quotient streams, host real equality, ambient completeness proofs, and completion witnesses.

Finally use Definition 466.8. The handoff row presents exactly $D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N$ to the $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ reader, so the consumer boundary is exact for the finite triangle-bound packet. □

Theorem 466.12 (RegularCauchyTriangleBound finite induction scope)**.** *For an accepted* RegularCauchyTriangleB *carrier*

$$T = (D_m, D_n, I_m, I_n, Q, A, B, H, C, P, N),$$

the three-leg estimate is obtained by finite induction over the displayed $\mathsf{StreamName}^{\uparrow}$ *windows and the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *rows visible to the packet. The induction step preserves the two endpoint*

$\mathsf{RegSeqRat}^{\uparrow}$ *regularity reads in A, the middle-meet read in B, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-addition ledger Q, so every completion-facing consumer receives the same transported triangle route recorded by Definition 466.10.*

Proof. The base window is the diagonal regularity route of Definition 466.3. It displays the route $D_m \longrightarrow I_m \longrightarrow I_n \longrightarrow D_n$ and binds the output tolerance to the dyadic sum in Q.

For the induction step, extend only the displayed $\mathsf{StreamName}^{\uparrow}$ window rows and the visible $\mathsf{CauchyModulusMeet}^{\uparrow}$ row. Endpoint-leg scope from Definition 466.4 keeps the first and third legs inside A, while middle-meet exactness from Definition 466.5 keeps the central comparison inside B.

The tolerance-addition ledger of Definition 466.6 carries the dyadic sum through the same finite step. Non-escape from Definition 466.7 prevents quotient streams, selected limits, host real equality, ambient completeness, or completion witnesses from entering the induction.

Finally apply the phase handoff of Definition 466.8 and the transported assembly of Definition 466.10. The resulting consumer route is exactly the finite induction over displayed windows and meet rows, with no additional completion object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ packet is generated from two diagonal observation rows, two intermediate observation rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, two $\mathsf{RegSeqRat}^{\uparrow}$ regularity reads, one $\mathsf{CauchyModulusMeet}^{\uparrow}$ middle read, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleBound}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The scoped dependency surface consists exactly of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows supporting the finite triangle-bound carrier, diagonal route, endpoint legs, middle meet, tolerance ledger, handoff, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTriangleBound}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyTriangleBoundUp.RegularCauchyTriangleBoundCarrier\_diagonal\_regular\_rout`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of streams, selected limits, host real equality, ambient completeness, completion witnesses, or bridge checking.

Upgrade path: No further paper-side closure grade remains; bridge checking is recorded only on the separate formal bridge axis.

Chapter 467

A Concrete Naming Certificate for $\mathsf{DyadicEmbedding}^{\uparrow}$

Following the Bishop–Bridges constructive-analysis treatment of rational and dyadic numbers inside real names, the $\mathsf{DyadicEmbedding}^{\uparrow}$ interface records a constant dyadic real name as a finite route from a dyadic row to a constant stream, a stationary regular rational readback, and a real seal. The packet below exposes the embedding path without quotienting regular sequences, selecting a host real limit, invoking real completeness, or comparing completed real representatives by ambient equality.

The constant dyadic route depends on the finite-window surface of chapter 126, the stationary readback surface of chapter 301, the dyadic source row of chapter 291, and the seal boundary of chapter 38. The $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ row is therefore not a separate analytic shortcut; it is the named sibling route that reads the same dyadic row as a constant stream, then as a stationary regular rational row, and finally as a finite $\mathsf{Real}^{\uparrow}$ seal.

Definition 467.1 (DyadicEmbedding carrier)**.** *A* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (d, S, R, Q, H, C, P, N)$$

with the following visible coordinates:

1. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row naming the dyadic rational observation;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule whose every admitted read returns the same dyadic row d;*
3. *R is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the constant stream schedule;*
4. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the stationary regular readback;*
5. *H stores componentwise* hsame *transports for the dyadic source, constant stream, stationary readback, real seal, package provenance, and local naming row;*
6. *C stores the displayed* Cont *routes from the dyadic row to the constant stream, from the stream to the stationary regular readback, and from the readback to the real seal;*
7. *P names* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StationaryRationalDiagonal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicEmbedding}^{\uparrow}}$ *row.*

The carrier is a finite naming certificate for constant dyadic real names. It contains no quotient stream, no selected limit, no host real equality test, and no completeness theorem for arbitrary regular sequences.

Lean4 ✓ `BEDC.Derived.DyadicEmbeddingUp.DyadicEmbeddingTasteGate single carrier alignment`

Theorem 467.2 (DyadicEmbedding NameCert obligations)**.** *For the carrier of Definition 467.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicEmbedding}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet* $D = (d, S, R, Q, H, C, P, N)$*;*
2. *classifier transport by componentwise* hsame *over the dyadic source, constant stream schedule, stationary readback, and real seal;*
3. *constant-stream readback, which requires every stream window to return the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row before it is read as a* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
4. *real-seal handoff, which routes any* $\mathsf{Real}^{\uparrow}$ *consumer through the stationary readback* R *and the package rows* P, N*;*
5. *non-escape, which excludes quotient regular streams, chosen limits, host equality of real representatives, trichotomy, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.DyadicEmbeddingUp.DyadicEmbeddingCarrier namecert obligations`

Proof. Carrier admission is the direct projection of the tuple in Definition 467.1. The coordinates name the dyadic source, stream schedule, regular readback, real seal, transports, routes, package provenance, and local certificate row in one finite BHist packet.

Classifier transport is componentwise. The rows d, S, R, Q are compared only through the stored hsame entries in H, and each accepted route is replayed through the displayed Cont entries in C.

For the constant-stream and real-seal obligations, first read d into the constant stream schedule S, then read the stationary regular row R through the same schedule, and finally expose Q only through R. This proves that all downstream real-name consumers receive the displayed finite route.

The non-escape clause follows from the carrier inventory. Since the tuple contains no coordinate for quotient streams, selected limits, host real equality, trichotomy, or completeness, no projected certificate route can export one. □

Theorem 467.3 (DyadicEmbedding constant stream readback)**.** *Every accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier reads its* $\mathsf{StreamName}^{\uparrow}$ *coordinate as a constant schedule over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* d*. The resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *row is the stationary rational diagonal named by* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$*, and no separate rational stream or unrecorded convergence witness participates in the readback.*

Lean4 ✓ `BEDC.Derived.DyadicEmbeddingUp.DyadicEmbeddingCarrier constant stream readback`

Proof. Project the dyadic source and stream coordinates from D. The route from d to S is displayed in C, so every admitted window of the stream schedule is read through the same source row.

The stationary readback is then the route from S to R. Because S has no coordinate other than the constant dyadic row and its finite window ledger, R is the regular rational readback of that stationary schedule.

The package row P names $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ as the readback shape. Thus the readback is a named stationary diagonal over d, not a separate rational stream or a convergence object chosen outside the packet. □

Theorem 467.4 (DyadicEmbedding real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer exported by a* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet factors through the same dyadic source row* d*, constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, and stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *before it reaches the real seal* Q*. The seal is therefore a finite handoff from the dyadic row to a constant real name, not a proof of completeness for arbitrary regular sequences.*

Proof. Start with a consumer route to Q. The carrier definition records this route in C as downstream of the regular readback coordinate R.

Apply Definition 467.3. It identifies R as the stationary rational diagonal read from the constant schedule S, and identifies S as the schedule produced from the same dyadic row d.

Composing the displayed routes gives the handoff d to S to R to Q. The packet has no row for arbitrary completeness or a selected limit, so the exported real seal is exactly the constant-name handoff. □

Theorem 467.5 (DyadicEmbedding ledger non-escape)**.** *The dependency ledger of a* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier is exhausted by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No consumer may add a quotient of streams, a selected real value, host equality of completed representatives, trichotomy, or a completeness principle as an implicit dependency.*

Proof. Inspect the package coordinate P and the local naming coordinate N. Together they name exactly the dependency list displayed in Definition 467.1.

The readback theorem accounts for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ through the rows d, S, R. The real-seal theorem accounts for the $\mathsf{Real}^{\uparrow}$ consumer through Q and the same route inventory.

No further coordinate remains in D. Therefore any extra quotient, selected host value, equality test, trichotomy, or completeness theorem would have to be supplied from outside the accepted packet, which the local $\mathsf{NameCert}_{\mathsf{DyadicEmbedding}^{\uparrow}}$ surface does not permit. □

Theorem 467.6 (DyadicEmbedding public scope)**.** *The public surface of a* $\mathsf{DyadicEmbedding}^{\uparrow}$ *seed packet is precisely the finite constant-name route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *named in Definition 467.1. It exports constant dyadic real names and excludes quotient-real construction, arbitrary Cauchy completion, selected limits, host real equality, and real-order trichotomy.*

Proof. Project the public rows of the carrier. They are exactly the dyadic source, constant stream schedule, stationary regular readback, real seal, componentwise transports, continuation routes, package provenance, and local naming row.

Use Definition 467.2 for carrier admission and local classifier transport. Use Definition 467.3 for the stationary readback and Definition 467.4 for the real-name consumer route.

Finally apply Definition 467.5. The ledger admits only the displayed dependency rows, so the public scope is exactly the constant dyadic real-name interface and none of the excluded completion or host-real structure. □

Theorem 467.7 (DyadicEmbedding constant schedule transport)**.** *For an accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier*

$$D = (d, S, R, Q, H, C, P, N),$$

componentwise hsame *transport preserving the dyadic source row* d *preserves the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*: every admitted transported stream window still reads the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row and uses only the displayed route from* d *to* S*.*

Proof. Project the transport coordinate H from Definition 467.1. Its entries compare the dyadic source, constant stream schedule, stationary readback, real seal, package row, and local naming row componentwise.

The constant-stream readback theorem identifies S as a schedule whose admitted windows all return the same dyadic source row d. Transport of d along H therefore transports the single source read used by every stream window.

The continuation coordinate C contains the displayed route from d to S. Replaying that route after the componentwise transport gives the same constant schedule shape and adds no separate stream or convergence witness. □

Theorem 467.8 (DyadicEmbedding stationary readback exactness)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R of an accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier is exactly the stationary diagonal over the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row d. Its exactness is witnessed by the route $d \to S \to R$ and by the* $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ *provenance row in P.*

Proof. Use Definition 467.7 to keep the stream coordinate S constant under the allowed componentwise transports. Thus every read entering the regular rational row starts from the same dyadic source d.

Apply the constant-stream readback theorem. It identifies R with the stationary readback produced from S and names $\mathsf{StationaryRationalDiagonal}^{\uparrow}$ as the readback shape.

The package row P records that stationary diagonal provenance, while C records the route $d \to S \to R$. Hence the readback is exact for the displayed dyadic row and is not an arbitrary regular sequence chosen outside the carrier. □

Theorem 467.9 (DyadicEmbedding Real seal consumer non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *carrier factors through the rows d, S, R, Q, H, C, P, N. It may read the dyadic source, constant stream schedule, stationary readback, real seal, transports, continuations, package provenance, and local naming row, but it has no quotient stream, selected limit, host equality, trichotomy, or completeness datum.*

Proof. Start with the real-seal handoff theorem. It already factors any $\mathsf{Real}^{\uparrow}$ read through d, then S, then R, and finally the real seal row Q.

Use stationary readback exactness to identify the regular rational row consumed by the seal with the stationary diagonal over d. The only transport and continuation data available for that route are the displayed rows H and C.

Finally apply the ledger non-escape theorem. The package and local naming rows P, N exhaust the dependency list, so a consumer cannot obtain quotient streams, selected limits, host real equality, trichotomy, or ambient completeness from the carrier. □

Theorem 467.10 (DyadicEmbedding stationary source obligation)**.** *Every accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet reads the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row d through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule and the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. No accepted read may replace that source by a varying stream coordinate or by a regular sequence selected outside the carrier.*

Proof. Begin with the carrier of Definition 467.1. Its source row is d, its stream schedule is S, and its regular rational readback is R.

Apply Definition 467.3. It identifies every admitted stream window with the same dyadic source row and reads the resulting regular rational row as the stationary diagonal named by $\mathsf{StationaryRationalDiagonal}^{\uparrow}$.

Use Definition 467.7 and Definition 467.8. Transport preserves the constant schedule, and exactness identifies R with the stationary readback over d, so no varying or externally selected source remains available. □

Theorem 467.11 (DyadicEmbedding Real seal obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet factors through the rows* d, S, R, Q, H, C, P, N *and receives only the constant dyadic real-name route. The consumer cannot bypass the stationary readback or obtain a selected real limit outside the displayed seal row.*

Proof. Start from a consumer route to the real seal row Q. By Definition 467.4, the route first reads d, then S, then R, and only then reaches Q.

Apply Definition 467.10. The source and readback are the stationary route through the same dyadic row, so the real seal is the constant dyadic real-name route.

Finally use Definition 467.9. The remaining rows H, C, P, N record transport, continuation, package provenance, and local naming, and they do not contain a bypass to a selected host real limit. □

Theorem 467.12 (DyadicEmbedding ledger exactness obligation)**.** *The visible ledger of an accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet is exhausted by* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StationaryRationalDiagonal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *It excludes quotient streams, selected limits, host equality, trichotomy, and completeness as available dependencies.*

Proof. Project the package row P and local naming row N from Definition 467.1. They list the dyadic source, constant stream schedule, stationary regular readback, real seal, history, transport, continuation, package, and naming dependencies.

Use Definition 467.5 for the base dependency boundary. Then apply Definition 467.6 to identify that boundary as the public constant dyadic real-name surface.

The stationary-source and real-seal obligations show that every consumer route remains inside d, S, R, Q, H, C, P, N. Therefore quotient streams, selected limits, host equality, trichotomy, and completeness would require extra rows not present in the accepted packet. □

Theorem 467.13 (DyadicEmbedding source-window-seal factorization)**.** *For every accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet*

$$D = (d, S, R, Q, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *consumer factors through the same dyadic source row* d*, the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the real seal row* Q*, and the displayed transport, continuation, package, and local naming rows* H, C, P, N*. No varying stream, selected limit, host equality, trichotomy, or completeness row participates in the route.*

Proof. Project the carrier of Definition 467.1. The source, stream, readback, seal, transport, continuation, package, and naming rows are precisely the coordinates d, S, R, Q, H, C, P, N.

Use Definition 467.10 to force the stream and regular rational readback through the same dyadic source row. Then apply Definition 467.11 to route the $\mathsf{Real}^{\uparrow}$ consumer through Q only after the stationary readback has been consumed.

Finally apply Definition 467.12. The ledger is exhausted by the displayed rows, so no varying stream, selected limit, host equality, trichotomy, or completeness coordinate remains available to the consumer. □

Theorem 467.14 (DyadicEmbedding obligation closure surface)**.** *The carrier, local* $\mathsf{NameCert}_{\mathsf{DyadicEmbedding}^{\uparrow}}$ *obligations, constant stream readback, real seal handoff, ledger non-escape, public scope, stationary readback exactness, and real seal obligation of Definition 467.1, Definition 467.2, Definition 467.3, Definition 467.4, Definition 467.5, Definition 467.6, Definition 467.8, and Definition 467.11 jointly supply the obligation surface for the constant dyadic real-name route. Every accepted consumer factors through the displayed rows* d, S, R, Q, H, C, P, N *and obtains no quotient stream, selected limit, host real equality, trichotomy, or ambient completeness principle.*

Proof. First project the carrier of Definition 467.1. The displayed rows d, S, R, Q, H, C, P, N name the dyadic source, constant stream schedule, stationary readback, real seal, transports, continuation routes, package provenance, and local naming row.

Use Definition 467.2 for carrier admission, classifier transport, constant-stream readback, real-seal handoff, and non-escape as the local naming obligations. Then apply Definition 467.3 and Definition 467.8 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ row R as the stationary readback of the same dyadic source row d.

Next route real consumers through Definition 467.4 and Definition 467.11. These theorems force every $\mathsf{Real}^{\uparrow}$ read to pass through the constant schedule and stationary readback before reaching Q.

Finally combine Definition 467.5 with Definition 467.6. The public surface is exactly the constant dyadic real-name route, and the excluded quotient, selected-limit, host-equality, trichotomy, and completeness data have no row in the packet. □

Theorem 467.15 (DyadicEmbedding scoped dependency spine)**.** *For every accepted* $\mathsf{DyadicEmbedding}^{\uparrow}$ *packet*

$$D = (d, S, R, Q, H, C, P, N),$$

the scoped public dependency spine is exactly the finite route from the $\mathsf{DyadicRatCore}^{\uparrow}$ *row* d *to the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* Q*, with componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. Every constant dyadic real-name consumer factors through that spine and cannot replace any edge by a quotient stream, selected limit, host equality, trichotomy, or completeness datum.*

Proof. Begin with the accepted carrier. By Definition 467.2, the packet admits only the displayed rows d, S, R, Q, H, C, P, N as the carrier and local naming surface, with classifier transport supplied componentwise by the stored hsame entries in H.

Next identify the readback route. Apply Definition 467.3 to force the stream coordinate S to be the constant schedule over d, and use Definition 467.8 to identify R as the stationary regular rational readback of that same schedule.

The real-name handoff is then fixed by Definition 467.11. Any $\mathsf{Real}^{\uparrow}$ consumer reaches Q only after passing through the dyadic source, constant stream schedule, and stationary readback, while the continuation coordinate C records the displayed route edges.

Finally apply Definition 467.12. The package provenance P and local naming row N exhaust the available dependencies, so no consumer has a row from which to obtain a quotient stream, selected limit, host equality, trichotomy, or completeness datum in place of an edge of the spine. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicEmbedding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicEmbedding}^{\uparrow}$ packet is generated from a $\mathsf{DyadicRatCore}^{\uparrow}$ row, a constant $\mathsf{StreamName}^{\uparrow}$ schedule, a stationary $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, package provenance, and a local $\mathsf{NameCert}$ row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicEmbedding}\uparrow}$, scopedClosed)

Scope closed: The obligation scope records the finite carrier, local naming obligations, constant-stream readback, real-seal handoff, ledger non-escape, public constant dyadic real-name boundary, stationary readback exactness, real-seal obligation, and obligation closure surface.

Formal verification: FormalStatus(NameCert$_{\text{DyadicEmbedding}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.DyadicEmbeddingUp.DyadicEmbeddingCarrier\_constant\_stream\_readback`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-real construction, arbitrary Cauchy completion, selected limits, host real equality, real-order trichotomy, or a public real-object surface.

Upgrade path: Public closure requires a public-object presentation of the constant dyadic real-name route through DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert.

Chapter 468

A Concrete Naming Certificate for $\mathsf{FiniteMultiHistPacket}^{\uparrow}$

The $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ interface supplies the first concrete carrier for the multi-Hist branch of Definition 9419.25. It packages a finite family of observer histories with only local comparison rows, visible continuation routes, package provenance, and local naming data. The packet is deliberately weaker than the later $\mathsf{CrossHistCausal}^{\uparrow}$ surface: it records the configuration that later causal and maximum-rate certificates may consume, but it does not already select a causal relation, a shared time coordinate, or a global observer frame. The downstream two-history consumer is the CrossHistCausal certificate of chapter 478; the present packet supplies its finite displayed-history rows, component ledgers, pairwise hsame rows, visible Cont routes, and no-global-sync input surface, while $\mathsf{CrossHistCausal}^{\uparrow}$ adds two-history causal witness rows.

Definition 468.1 (Finite MultiHist packet carrier)**.** *A* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (H, L, S, C, B, P, N)$$

with the following visible rows:

1. $H = (H_1, \ldots, H_n)$ *is the finite observer-history row, whose entries are displayed* BHist *histories;*
2. $L = (L_1, \ldots, L_n)$ *is the component ledger, with* L_i *naming the local rows of* H_i *available to consumers;*
3. $S = (S_{ij})_{i,j}$ *is the pairwise* hsame *row, recording exactly the history-sameness comparisons admitted between displayed component rows;*
4. C *records the visible* Cont *routes by which a consumer reads a component ledger, a pairwise* hsame *entry, the package row, and the local naming row;*
5. B *is the no-global-sync boundary row, recording that the packet has no coordinate for a privileged shared now, global frame, total observer order, or selected synchronization map;*
6. P *is the* Pkg *provenance over* BHist, Hist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteMultiHistPacket}^{\uparrow}}$ *row.*

The carrier is finite and every cross-component comparison available to the packet is displayed in S *and reached through* C*. It has no cross-Hist causal witness, no maximum-rate function, no metric row, and no synchronization object.*

Lean4 (definition) `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp`
Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketTasteGate single carrier alignment`

Theorem 468.2 (Finite MultiHist packet NameCert obligations)**.** *For the carrier of Definition 468.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteMultiHistPacket}^{\uparrow}}$ *surface consists of carrier admission for* $M = (H, L, S, C, B, P, N)$*, component-ledger exactness for each displayed history row, pairwise* hsame *stability through* S*, visible read access through* C*, package provenance through* P*, local naming through* N*, and the no-global-sync boundary recorded by* B*.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp taste gate boundary`

Proof. Carrier admission is the displayed finite tuple. The rows H, L, S, C, B, P, N are the only data made available to the naming certificate, so every obligation can be projected from a named coordinate of the packet.

Component-ledger exactness follows from L. For each component H_i, the ledger L_i records the local rows a consumer may inspect before any cross-component comparison is used.

Pairwise stability is supplied by S. Each admitted comparison between component histories is an explicit hsame row S_{ij}, and no comparison outside that row is part of the carrier.

The read and provenance clauses are supplied by C, P, and N. The continuation row C exposes the component ledger, the pairwise sameness row, package provenance, and local naming row; the package row P names the BEDC kernel objects used by the certificate; N is the local NameCert surface.

Finally, B records the boundary condition. Since the carrier has no coordinate for a global frame, shared clock, total observer order, selected synchronization map, causal relation, maximum-rate function, or metric data, the local naming row cannot export any of those objects. □

Theorem 468.3 (Finite MultiHist packet hsame transport)**.** *Let* $M = (H, L, S, C, B, P, N)$ *be an accepted* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *carrier. Any transport that preserves the component ledgers in* L *and the pairwise* hsame *row* S *preserves the same finite multi-Hist packet interface, with consumer access replayed through the displayed* Cont *routes in* C*.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp hsame transport`

Proof. Project the component ledgers from L. A transported read of a component history must still pass through the displayed ledger for that component, so it cannot introduce an unrecorded history row.

Next project the pairwise comparison row S. Since the admitted cross-component comparisons are exactly the displayed hsame entries, preserving S preserves the comparison surface of the packet.

It remains to check access. The routes in C are the only visible continuations from consumer reads to component ledgers, pairwise comparison rows, package provenance, and the naming row. Replaying those routes after transport gives the same finite interface and adds no global synchronization coordinate. □

Theorem 468.4 (Finite MultiHist packet no-sync boundary)**.** *The public boundary of an accepted* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *carrier exports only the displayed finite histories, component ledgers, pairwise* hsame *rows, visible* Cont *routes, package provenance, and local* NameCert *row. It exports no privileged shared now, no global observer frame, no total ordering of observers, and no selected synchronization map between components.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp no sync boundary`

Proof. Inspect the coordinates of Definition 468.1. The public rows are H, L, S, C, B, P, N, and each row is finite and local to the packet.

The no-global-sync boundary is the row B. It records that the packet lacks a privileged shared now, global observer frame, total observer order, and selected synchronization map.

No other coordinate can supply such data. The history rows are local components, L records their local ledgers, S records only pairwise hsame comparisons, C records visible reads, P records package provenance, and N records the local naming surface. Hence the boundary exports no synchronization object. □

Theorem 468.5 (Finite MultiHist packet causal interface)**.** *Every accepted* FiniteMultiHistPacket$^{\uparrow}$ *carrier supplies the input surface required by a later* CrossHistCausal$^{\uparrow}$ *certificate: a finite family of displayed observer histories, component ledgers, pairwise* hsame *transport rows, visible* Cont *routes, package provenance, and a no-global-sync boundary. It does not itself assert any cross-Hist causal dependence or maximum causal rate.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp causal interface`

Proof. The consumer surface needed by CrossHistCausal$^{\uparrow}$ begins with displayed histories. The row H supplies the finite family of observer histories, and L supplies the local ledgers from which those histories are read.

Transport data are already present. The row S records the pairwise hsame comparisons, and C records the continuation routes through which a later causal certificate may read the selected components and their comparison rows.

Package and naming data are also present. The rows P and N identify the kernel objects and the local NameCert surface from which the later certificate may project its inputs.

The boundary condition remains negative. The row B excludes a global synchronization object, and the carrier has no coordinate for a causal witness or maximum-rate function. Thus the packet supplies the causal interface inputs without pre-solving the later causal or rate obligations. □

Theorem 468.6 (Finite MultiHist packet CrossHist forward route)**.** *Every* CrossHistCausal$^{\uparrow}$ *consumer of an accepted* FiniteMultiHistPacket$^{\uparrow}$ *carrier must enter through the finite family row H, component ledger L, pairwise* hsame *row S, visible* Cont *routes C, no-global-sync boundary B,* Pkg *provenance P, and local* NameCert *row N before adding causal witness rows. The* CrossHistCausal$^{\uparrow}$ *certificate of chapter 478 adds causal dependence rows downstream of this carrier; it does not replace the finite multi-history packet or supply a hidden global synchronization coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketCrossHistForwardRoute`

Proof. Start with the causal interface theorem, Definition 468.5. It exposes exactly the finite family of observer histories, component ledgers, pairwise transport rows, visible routes, provenance, and no-global-sync boundary required before a later causal certificate can read any input.

The forward route is therefore ordered by the carrier rows. A CrossHistCausal$^{\uparrow}$ consumer first reads H, L, S, C, B, P, N from the finite packet, and only then may append causal-witness rows in the downstream chapter named by chapter 478.

Since the finite carrier has no coordinate for a causal witness, maximum-rate function, privileged shared now, global observer frame, or selected synchronization map, the downstream causal packet extends the visible carrier route rather than replacing it. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMultiHistPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ packet is generated from a finite family of displayed BHist observer histories, component ledgers, pairwise hsame rows, displayed Cont routes, package provenance, a no-global-sync boundary row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMultiHistPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite multi-Hist carrier and its displayed component ledgers, pairwise hsame rows, visible Cont routes, package provenance, local naming row, and no-global-sync boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMultiHistPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close cross-Hist causal dependence, maximum causal rates, Lorentz geometry, curved spacetime, quantum-state certificates, or a checked Lean theorem for the carrier obligations.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the displayed finite multi-Hist packet, its transport rows, visible reads, and no-global-sync boundary.

Theorem 468.7 (Finite MultiHist packet subpacket restriction)**.** *Let*

$$M = (H, L, S, C, B, P, N)$$

be an accepted $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *carrier. Let J be a displayed finite subfamily of the observer-history row H, and suppose the subfamily ledger records every selected component read, every selected component-ledger read, and every selected pairwise comparison through the visible* Cont *routes already present in C. Write*

$$M|_J = (H_J, L_J, S_J, C_J, B, P_J, N_J)$$

for the packet obtained by projecting H to the selected histories, L to the corresponding component ledgers, S to the selected pairwise hsame *entries, C to the displayed routes that read those selected rows, and P, N to the same package-provenance and local-naming data restricted to the selected rows. Then $M|_J$ is again an accepted* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *carrier. Its public boundary has the same no-global-sync row B, so the restriction exports no privileged shared now, no global observer frame, no total observer order, and no selected synchronization map.*

Lean4 ✓ `BEDC.Derived.FiniteMultiHistPacketUp.FiniteMultiHistPacketUp subpacket restriction`

Proof. The carrier shape is read from Definition 468.1. Since H is finite and J is a displayed finite subfamily, H_J is a finite observer-history row. The component ledger L_J is obtained by keeping exactly the entries L_i whose histories H_i are selected by J, so every component row of the restricted packet remains a displayed local ledger row of the original packet.

The pairwise comparison row is closed under this projection. By definition, S contains exactly the admitted cross-component hsame comparisons of M. The row S_J contains precisely those entries S_{ij} with both endpoints selected by J. Thus the restricted packet neither invents a comparison outside S nor forgets the comparison data required for the displayed selected components.

The visible-read condition is inherited from C. By hypothesis, every selected history read, component-ledger read, and pairwise comparison read factors through a route already displayed in C. The row C_J is the subrow of those routes, so consumer access to $M|_J$ is still by visible Cont routes and has no hidden access path.

The package and naming rows are restricted, not replaced. The rows P_J and N_J are the original provenance and local NameCert rows read only on the selected packet coordinates. Applying Definition 468.2 to these projected coordinates gives carrier admission, component-ledger exactness, pairwise hsame stability, visible read access, package provenance, local naming, and the boundary

clause for the restricted packet. Equivalently, Definition 468.3 applies because the projection preserves the selected component ledgers and the selected pairwise hsame row.

It remains to check the boundary. The restriction keeps the row B unchanged and only deletes unselected histories, ledger entries, comparison entries, and routes. Deleting displayed rows cannot create a coordinate for a shared clock, global frame, total observer order, synchronization map, causal relation, maximum-rate function, or metric row. Therefore Definition 468.4 applies with the same no-global-sync boundary B, and the projected packet is an accepted $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ carrier with the same public no-sync boundary. □

Chapter 469

A Concrete Naming Certificate for $\mathsf{ContinuationTrace}^{\uparrow}$

The $\mathsf{ContinuationTrace}^{\uparrow}$ packet gives the computation-as-continuation reading of chapter 9365 a concrete finite carrier. It treats a displayed small-step computation as a BHist trace: each row records a source history, a mark-and-tail route, a target history, the hsame transports needed to compare displayed histories, the Cont route that witnesses the step, package provenance, and a local naming row. The packet does not introduce a host evaluator, a host termination predicate, quotient behaviour, or big-step semantic object.

The same trace packet supplies the step-ledger surface consumed by the unary-continuation categorical reading in chapter 471: source and target rows together with Cont tails become the hom-carrier data, while the trace ledger records the finite route support.

Definition 469.1 (ContinuationTrace carrier)**.** *A* $\mathsf{ContinuationTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, R, Q, H, C, P, N)$$

whose displayed coordinates are as follows:

1. *S is the source-history row being inspected by the small-step read;*
2. *M is the mark row and displayed tail route by which the step is named;*
3. *R is the target-history row reached by the displayed route;*
4. *Q is the finite trace ledger containing the ordered source, mark, tail, and target reads for each displayed step row;*
5. *H is the componentwise* hsame *transport row comparing source histories, tail histories, and target histories across accepted trace packets;*
6. *C is the* Cont *route witnessing that the displayed tail carries the source row to the target row;*
7. *P is the* Pkg *provenance over* BHist, BMark, hsame, Cont, *and the continuation-reading dependencies;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationTrace}^{\uparrow}}$ *row.*

Two accepted traces classify together only when the source histories, mark-and-tail route, target histories, finite trace ledger, hsame *transport,* Cont *route, package row, and local certificate row are transported componentwise by the displayed rows.*

Lean4 (definition) `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier`

Theorem 469.2 (ContinuationTrace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationTrace}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier: an accepted trace is exactly the finite packet of Definition 469.1;*
2. *classifier: two accepted traces compare only by componentwise* hsame *transport of* S*,* M*,* R*,* Q*,* H*,* C*,* P*, and* N*;*
3. *step readback: every displayed small-step row is read from the* Cont *route* C *together with the mark-and-tail coordinate* M *and the source and target rows* S *and* R*;*
4. *non-escape: the trace exports no host function, host termination object, quotient behaviour, or big-step semantic value;*
5. *public export: the packet handed to later formalization is the same finite* BHist *trace carrier with its classifier, readback, non-escape, and provenance rows.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier namecert obligations`

Proof. Carrier admission is the finite packaging operation of Definition 469.1. Every coordinate in the accepted object is one of the displayed trace rows.

Classifier stability is componentwise. A transport of accepted traces may compare source histories, target histories, route rows, and provenance rows, but it can only do so through the listed hsame and Cont fields.

The final three rows are supplied by the readback, non-escape, and public export theorems below. Together they account for every consumer-facing trace read and keep the packet inside the BEDC history, mark, sameness, continuation, package, and naming primitives. □

Theorem 469.3 (ContinuationTrace small-step readback)**.** *Let* $T = (S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{ContinuationTrace}^{\uparrow}$ *carrier. Every displayed small-step transition read from* T *factors through the source-history row* S*, the mark-and-tail row* M*, the target-history row* R*, the finite trace ledger* Q*, and the* Cont *route* C*. Conversely, any displayed* Cont *tail in* C *whose source, mark, tail, and target rows are listed in* Q *is read back as a step of the same trace classifier.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier small step readback`

Proof. Project the accepted trace through Definition 469.1. A step read has no coordinates available except the source row, the mark-and-tail row, the target row, the trace ledger, and the displayed continuation route.

For the forward direction, inspect the row of Q named by the consumer. It records the same source, mark, tail, and target rows, and the route coordinate C supplies the displayed Cont witness that the tail carries the source to the target.

For the reverse direction, begin with such a displayed Cont tail in C. Since its source, mark, tail, and target rows are listed in Q, the classifier of the carrier reads it through the same local packet and returns the corresponding trace step. The transport row H accounts only for hsame comparisons among displayed histories and does not alter the trace boundary. □

Theorem 469.4 (ContinuationTrace non-escape boundary)**.** *An accepted* $\mathsf{ContinuationTrace}^{\uparrow}$ *packet exposes no host function graph, host termination predicate, quotient behaviour relation, selected evaluator, or big-step semantic object. Its visible computational content is exactly the finite source, mark-and-tail, target, trace-ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 469.1.*

Proof. Read the carrier coordinates. None of S, M, R, Q, H, C, P, or N is a host function, a host termination object, a quotient of behaviours, or a big-step value.

The only operational row is C, and it is a BEDC continuation route. By Definition 469.3, consumer-facing steps are recovered from that route together with the finite trace ledger Q and the displayed source, mark, and target rows.

Thus every exported computation read remains in the finite continuation trace packet. There is no coordinate through which a host evaluator, host termination principle, quotient behaviour, or big-step semantics can enter the public surface. □

Theorem 469.5 (ContinuationTrace public export)**.** *The public* $\mathsf{ContinuationTrace}^{\uparrow}$ *export surface is the finite carrier of Definition 469.1 together with the five NameCert rows of Definition 469.2, the readback equivalence of Definition 469.3, and the boundary statement of Definition 469.4. This is the canonical finite packet for future Lean formalization of continuation traces.*

Proof. The carrier exposes exactly the rows S, M, R, Q, H, C, P, N. The NameCert theorem identifies the accepted carrier, its classifier, the step readback row, non-escape, and the public export obligation over those rows.

The readback theorem accounts for the computation-facing use: displayed small-step reads are precisely reads through the source, mark-and-tail, target, trace ledger, and continuation route. The non-escape theorem accounts for the boundary by excluding host functions, host termination, quotient behaviours, selected evaluators, and big-step semantic objects.

Repacking these clauses gives the public finite certificate. Since every clause points back to the same displayed carrier coordinates, the export is a concrete BEDC packet and not an external operational semantics. □

Theorem 469.6 (ContinuationTrace obligation closure package)**.** *The carrier, NameCert obligations, small-step readback, non-escape boundary, and public export of Definition 469.1, Definition 469.2, Definition 469.3, Definition 469.4, and Definition 469.5 assemble the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *obligation package over* BHist, BMark, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The package exposes source histories, mark-and-tail rows, target histories, trace ledgers, transports, continuation routes, provenance, and the local naming row as the whole public surface.*

Proof. The carrier theorem supplies the finite trace packet. The NameCert obligations bind admission, classifier comparison, step readback, non-escape, and public export to the displayed coordinates S, M, R, Q, H, C, P, N.

The small-step readback theorem accounts for every computation-facing row by reading it through the source history, mark-and-tail route, target history, trace ledger, and continuation route. The non-escape boundary prevents the same read from acquiring a host evaluator, host termination predicate, quotient behaviour, selected evaluator, or big-step semantic object.

The public export theorem repacks those rows as the visible trace interface. Because all five cited results refer back to the same carrier coordinates, they form one finite $\mathsf{ContinuationTrace}^{\uparrow}$ obligation package over the BEDC history, mark, sameness, continuation, package, and naming primitives. □

Theorem 469.7 (ContinuationTrace concatenation readback)**.** *Let*

$$T_0 = (S_0, M_0, R_0, Q_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad T_1 = (S_1, M_1, R_1, Q_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{ContinuationTrace}^{\uparrow}$ *packets. If the target row* R_0 *of* T_0 *and the source row* S_1 *of* T_1 *are matched by a displayed* hsame *row, then their concatenation is a finite trace packet whose step*

reads factor through exactly one of the two source ledgers Q_0 or Q_1, with the boundary comparison supplied by that displayed hsame *row.*

The public concatenated packet is still over BHist, BMark, hsame, Cont, Pkg, *and* NameCert. *It exports no host evaluator, host termination predicate, quotient behaviour, or big-step semantic object.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier concatenation readback`

Proof. Project both accepted packets through Definition 469.1. The only rows available for a step read in the first packet are $S_0, M_0, R_0, Q_0, H_0, C_0, P_0, N_0$, and the only rows available in the second packet are $S_1, M_1, R_1, Q_1, H_1, C_1, P_1, N_1$.

Use Definition 469.3 separately on the two packets. A step read before the boundary is read from Q_0 together with C_0, while a step read after the boundary is read from Q_1 together with C_1.

The displayed hsame match between R_0 and S_1 supplies only the boundary comparison. By Definition 469.4, that comparison cannot introduce a host evaluator, host termination predicate, quotient behaviour, selected evaluator, or big-step semantic object.

Finally apply Definition 469.5 and Definition 469.6. Repacking the two finite ledgers with the boundary hsame row keeps the concatenated public packet inside BHist, BMark, hsame, Cont, Pkg, and NameCert, and every step read is accounted for by exactly one source ledger. □

Theorem 469.8 (ContinuationTrace endpoint ledger exhaustion)**.** *Let $T = (S, M, R, Q, H, C, P, N)$ be an accepted* ContinuationTrace$^{\uparrow}$ *packet. Every endpoint-facing source or target read of T is exhausted by the displayed rows S, M, R, Q, H, C, P, N: the endpoint read is listed in the finite trace ledger Q and replayed by the continuation route C from the displayed source side to the displayed target side.*

No endpoint read of T is supplied by a host evaluator, a host termination predicate, a quotient behaviour, a selected evaluator, or a big-step semantic object.

Lean4 ✓ `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier endpoint ledger exhaustion`

Proof. Project T through Definition 469.1. The endpoint-facing coordinates are the source row S, mark-and-tail row M, target row R, finite ledger Q, transport row H, continuation route C, provenance row P, and local naming row N.

Apply Definition 469.3. A displayed endpoint read is a small-step read only when its source, mark, tail, and target rows are listed in Q and its route is supplied by C.

Apply Definition 469.4 and Definition 469.5. These two results exclude host evaluators, host termination, quotient behaviours, selected evaluators, and big-step semantic objects from the public packet, so the endpoint read is exhausted by the concrete trace rows already displayed. □

Theorem 469.9 (ContinuationTrace boundary hsame pullback)**.** *Let*

$$T_0 = (S_0, M_0, R_0, Q_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad T_1 = (S_1, M_1, R_1, Q_1, H_1, C_1, P_1, N_1)$$

be accepted ContinuationTrace$^{\uparrow}$ *packets. A displayed boundary comparison R_0hsameS_1 pulls back only through the transport rows H_0, H_1 and the displayed ledgers Q_0, Q_1. It adds no new step row and no extra endpoint outside the two accepted trace packets.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier boundary hsame pullback`

Proof. Project both packets through Definition 469.1. The boundary comparison has access to R_0 from the first packet and S_1 from the second packet, while its admissible transport data is recorded by H_0 and H_1.

Use Definition 469.3 on each side. Step rows before the boundary are read from Q_0 and C_0, and step rows after the boundary are read from Q_1 and C_1.

Now apply Definition 469.7. The displayed $R_0\mathsf{hsame}S_1$ row supplies the boundary comparison between the two ledgers rather than a third ledger row. By Definition 469.4, the comparison cannot import an external endpoint or host-level computation. □

Theorem 469.10 (ContinuationTrace route support totality)**.** *For accepted packets*

$$T_0 = (S_0, M_0, R_0, Q_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad T_1 = (S_1, M_1, R_1, Q_1, H_1, C_1, P_1, N_1),$$

every consumer route of their concatenated $\mathsf{ContinuationTrace}^{\uparrow}$ *packet factors through exactly one of the source ledgers* Q_0 *or* Q_1*, together with the displayed boundary row* $R_0\mathsf{hsame}S_1$*. The full route remains a route over* BHist*,* BMark*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Begin with the two accepted carriers of Definition 469.1. Their route support is displayed by the two finite ledgers Q_0, Q_1, the continuation rows C_0, C_1, the transport rows H_0, H_1, provenance, and local naming.

Apply Definition 469.3 separately to the two packets. A consumer route whose read lies before the boundary is supported by Q_0, and a consumer route whose read lies after the boundary is supported by Q_1.

Apply Definition 469.7 and the boundary pullback of Definition 469.9. The displayed $R_0\mathsf{hsame}S_1$ row is the only boundary support, so the concatenated route has no third source ledger.

Finally use Definition 469.5. The public packet is still the finite BEDC trace packet, hence the complete route support is over BHist, BMark, hsame, Cont, Pkg, and NameCert. □

Theorem 469.11 (ContinuationTrace route composition determinacy)**.** *For accepted packets*

$$T_0 = (S_0, M_0, R_0, Q_0, H_0, C_0, P_0, N_0), \quad T_1 = (S_1, M_1, R_1, Q_1, H_1, C_1, P_1, N_1), \quad T_2 = (S_2, M_2, R_2, Q_2, H_2, C_2, P_2, N$$

suppose the displayed boundary rows match $R_0\mathsf{hsame}S_1$ *and* $R_1\mathsf{hsame}S_2$*. Then every consumer route of the two-step concatenated* $\mathsf{ContinuationTrace}^{\uparrow}$ *packet factors through exactly one source ledger among* Q_0, Q_1, Q_2 *together with the two displayed boundary rows. The selected source ledger and the resulting public route are independent of whether the first boundary* $R_0\mathsf{hsame}S_1$ *or the second boundary* $R_1\mathsf{hsame}S_2$ *is assembled first.*

The composed packet still exports no host evaluator, host termination predicate, quotient behaviour, or big-step semantic object.

Proof. Apply Definition 469.7 to the pair (T_0, T_1) and then to the resulting packet with T_2. The first assembly contributes exactly the ledgers Q_0, Q_1 and the boundary row $R_0\mathsf{hsame}S_1$; the second assembly contributes Q_2 and the boundary row $R_1\mathsf{hsame}S_2$.

Apply the same readback in the other order, first to (T_1, T_2) and then to T_0 with that packet. The visible route inventory is again Q_0, Q_1, Q_2 plus the two displayed boundary rows.

Use Definition 469.9 for each boundary. Each hsame row pulls back only to the adjacent trace ledgers and does not add a third boundary source or an external endpoint.

Finally apply Definition 469.10 on the assembled finite packet. Every consumer route chooses exactly one of the three source ledgers, while the two boundary rows only glue adjacent endpoints. The non-escape boundary remains the one already stated for continuation traces, so host evaluators, termination predicates, quotient behaviours, and big-step semantics stay outside the public route. □

Theorem 469.12 (ContinuationTrace scoped readback)**.** *Every accepted* $\mathsf{ContinuationTrace}^{\uparrow}$ *route is scoped exactly by its* BHist *source rows,* BMark *step rows, target rows, finite ledgers, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The route cannot read host evaluator semantics, host termination, quotient behaviour, or big-step data.*

Proof. Begin with small-step readback. It reads every displayed step from the source row, mark-and-tail row, target row, trace ledger, and continuation route.

Use the non-escape boundary and public export. These identify the visible computational content with the finite trace packet and exclude host evaluators, host termination, quotient behaviours, selected evaluators, and big-step semantic objects.

Now apply route support totality. Every consumer route factors through exactly one displayed source ledger together with the relevant boundary hsame row, so no undisplayed route support can enter.

Finally use route composition determinacy. Composed routes remain independent of the order in which adjacent boundary rows are assembled, and every read still chooses one displayed ledger. The full scope is therefore exactly the BHist, BMark, hsame, Cont, Pkg, and NameCert surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuationTrace}^{\uparrow}$ packet is generated from finite source-history rows, mark-and-tail route rows, target-history rows, trace ledgers, displayed hsame transports, Cont routes, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ContinuationTrace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 469.11 and Definition 469.12, together with the displayed trace carrier, small-step readback, concatenation readback, endpoint ledger exhaustion, boundary hsame pullback, and route support rows, supply the mature paper surface over BHist, BMark, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ContinuationTrace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ContinuationTraceUp.ContinuationTraceCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean formalization, bridge checking, host evaluator semantics, host termination, quotient behaviours, or big-step semantics.

Upgrade path: The remaining step is a checked bridge over the finite $\mathsf{ContinuationTrace}^{\uparrow}$ trace packet and its composed route ledger.

Chapter 470

A Concrete Naming Certificate for $\mathsf{StreamLimitReflection}^{\uparrow}$

The $\mathsf{StreamLimitReflection}^{\uparrow}$ interface records the reflection-limit boundary in the countable-name layer. It packages a finite observation schedule for regular rational names, the RealUp seal row available at the same finite site, and a structural code row for the internal CIC* copy, while keeping runtime execution and truth classification outside the packet.

The reflection packet consumes chapter 126 for scheduled finite windows, chapter 301 for regular rational readback, and chapter 38 for the finite $\mathsf{Real}^{\uparrow}$ seal. These are substantive carrier rows of the packet; runtime execution, truth classification, quotient streams, selected limits, and host real equality remain outside the packet. The carrier is composite over the $\mathsf{StreamName}^{\uparrow}$ schedule surface, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, the $\mathsf{Real}^{\uparrow}$ finite seal row, and the structural-code boundary ledger.

Definition 470.1 (StreamLimitReflection carrier)**.** *A* $\mathsf{StreamLimitReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, E, K, T, H, C, P, N)$$

with the following visible coordinates:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, listing the finite countable-name window queried by the packet;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row read over that schedule;*
3. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the finite regular-name readback;*
4. *K is an internal* CIC* *structural-code row for formation, recursor, and certificate syntax visible at this site;*
5. *T records runtime-boundary and truth-boundary ledger entries, separating structural codes from an executing checker and from an internal truth classifier;*
6. *H stores the componentwise* hsame *routes for S, R, E, K, T;*
7. *C stores the displayed* Cont *routes for reading the schedule, regularity row, RealUp seal, structural-code row, and boundary ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamLimitReflection}^{\uparrow}}$ *row.*

The carrier exports finite structural observations and explicit non-closure ledgers. It stores no internal runtime, no truth predicate, no quotient stream, no selected limit, and no host real equality.

Lean4 (definition) `BEDC.Derived.StreamLimitReflectionUp.StreamLimitReflectionUp`
Lean4 ✓ `BEDC.Derived.StreamLimitReflectionUp.StreamLimitReflectionTasteGate single carrier alignment`

Theorem 470.2 (StreamLimitReflection NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StreamLimitReflection}^{\uparrow}}$ *surface for the carrier of Definition 470.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet $Q = (S, R, E, K, T, H, C, P, N)$;*
2. *classifier transport by componentwise* hsame *and* Cont *on the stream schedule, regularity row, RealUp seal, structural-code row, boundary ledger, package row, and local naming row;*
3. *countable-name handoff from S to the regular rational readback R and the RealUp seal E;*
4. *structural-code separation, which reads K only as* CIC* *formation and recursor syntax, not as an executing checker;*
5. *boundary non-escape, which forces runtime and truth claims to pass through the ledger T rather than through a hidden object-language predicate.*

Lean4 ✓ `BEDC.Derived.StreamLimitReflectionUp.TasteGate.StreamLimitReflectionNameCertObligations`

Proof. Carrier admission is tuple projection. The rows $S, R, E, K, T, H, C, P, N$ are all displayed coordinates of the same finite BHist packet.

Classifier transport is componentwise. The stored hsame routes compare the schedule, regularity, seal, structural-code, ledger, package, and local naming rows, while the Cont routes record the allowed reads among those rows.

The countable-name handoff is then finite. The observation schedule S exposes the requested stream window, R records the regular rational row over that window, and E records the RealUp seal that can be named from the same readback.

The structural row K is only syntax for the internal copy. The ledger T records that no coordinate executes that syntax and no coordinate classifies BEDC truth. Thus the certificate exports structural finite observations and explicit non-closure ledgers, not a runtime, truth predicate, quotient stream, selected limit, or host real equality. □

Theorem 470.3 (StreamLimitReflection runtime boundary)**.** *Every runtime-facing consumer of an accepted* $\mathsf{StreamLimitReflection}^{\uparrow}$ *carrier factors through $S, R, E, K, T, H, C, P, N$. It may read the structural* CIC* *code row and the boundary ledger, but it cannot obtain an internal evaluator, checker, execution semantics, quotient stream, selected limit, or host real equality from the packet.*

Proof. Project the accepted carrier. The only row that mentions the host calculus is K, and that row is structural syntax: formation, recursor, and certificate-code data.

The boundary ledger T records that execution of such syntax is not a coordinate of the carrier. The transport rows H and C can move and read only the displayed finite rows, so they cannot create an evaluator or checker.

The regular-name rows do not change this boundary. They expose the countable schedule, regular rational readback, and RealUp seal, but none of these rows contains a quotient stream, selected limit, host equality test, or runtime semantics. Hence every runtime-facing read remains outside the finite packet. □

Theorem 470.4 (StreamLimitReflection truth non-closure)**.** *Every truth-facing consumer of an accepted* $\mathsf{StreamLimitReflection}^{\uparrow}$ *carrier reads the structural code row* K *and the truth-boundary entries of* T *only as finite ledger data. The packet does not define a BEDC truth predicate for the represented* CIC* *sentences and does not turn the RealUp seal* E *into host real equality or completed-limit selection.*

Proof. Begin with the structural-code row. By Definition 470.1, K names formation, recursor, and certificate syntax for the internal copy, not a truth classifier over that syntax.

Next read the truth-boundary entries of T. They mark truth-total classification as external to the packet, so a consumer may cite the boundary but cannot convert it into an object-language truth predicate.

Finally inspect the countable-name side. The RealUp seal E is attached to the finite regular-name readback R over schedule S. It seals the displayed name site only; it supplies neither host real equality nor a selected completed limit. Therefore truth-facing exports remain finite ledger exports. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamLimitReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StreamLimitReflection}^{\uparrow}$ packet is generated from StreamName schedule rows, RegSeqRat regularity rows, RealUp seal rows, structural-code rows, runtime-boundary and truth-boundary ledger rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier reflects stream-limit structure only as finite observations plus explicit non-closure ledgers. The code row is syntax, the boundary row records that no evaluator or truth classifier is present, and the RealUp seal remains attached to the finite regular-name readback. Thus the packet generates a structural reflection route without quotient streams, selected limits, host real equality, internal runtime, or BEDC truth predicate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamLimitReflection}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition and the NameCert-obligation, runtime-boundary, and truth-non-closure theorems for $\mathsf{StreamLimitReflection}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamLimitReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim internal runtime, truth predicates, quotient streams, selected limits, host real equality, internal evaluators, or checkers for the represented syntax.

Upgrade path: To reach obligationClosure, the structural-code and boundary-ledger rows must be encoded with checked runtime and truth non-closure proofs.

Chapter 471

A Concrete Naming Certificate for $\mathsf{UnaryContinuationCategory}^{\uparrow}$

The $\mathsf{UnaryContinuationCategory}^{\uparrow}$ packet is the concrete category-shaped reading of the BEDC unary continuation calculus. It names the existing carrier rows used by $\mathsf{Cat}^{\uparrow}$: unary object histories, displayed $\mathsf{CategoryHomCarrier}$ rows, continuation-tail ledgers, componentwise hsame transport, Cont composition routes, Pkg provenance, and a local $\mathsf{NameCert}$ ledger. It does not introduce a host category universe or any equality outside BHist.

It consumes the finite transition-step reading of chapter 482 and the unit and associativity ledger of chapter 485. Its object and hom rows are the category-shaped repackaging of those continuation surfaces, not an isolated kernel-only construction.

Definition 471.1 (Unary-continuation category carrier)**.** *A* $\mathsf{UnaryContinuationCategory}^{\uparrow}$ *carrier is a finite packet*

$$U = (O, H, I, M, T, P, N)$$

with the following visible rows:

1. *O is the unary object row: each displayed object history a is accepted exactly through* $\mathsf{UnaryHistory}(a)$*.*
2. *H is the displayed hom row: an arrow from a to b with displayed tail f is accepted by*

$$\mathsf{CategoryHomCarrier}(a, b, f).$$

3. *I records the empty-tail identity ledgers* $\mathsf{Cont}(a, \mathsf{e}, a)$ *over unary object rows.*
4. *M records the binary composition ledgers: for displayed hom rows* $\mathsf{CategoryHomCarrier}(a, b, f)$ *and* $\mathsf{CategoryHomCarrier}(b, c, g)$*, a composite tail fg is accepted only through the displayed* $\mathsf{Cont}(f, g, fg)$ *route together with the resulting* $\mathsf{CategoryHomCarrier}(a, c, fg)$ *row.*
5. *T records componentwise* hsame *transport for object histories and for displayed hom histories over fixed endpoints.*
6. *P is the* Pkg *provenance row naming the unary object inspection, hom-carrier inspection, identity ledger, and composition ledger.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}^{\uparrow}}$ *ledger for the carrier, classifier, stability, exactness, and provenance rows.*

The carrier is therefore a category-shaped packet over BHist *only: its objects, arrows, identities, composites, and equality rows are the existing unary-continuation data.*

Lean4 ✓ `BEDC.Derived.UnaryContinuationCategoryUp.UnaryContinuationCategoryTasteGate single carrier alignment`

Theorem 471.2 (Unary-continuation category NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}^{\uparrow}}$ *obligation surface consists of five rows:*

1. *carrier habitation by the empty unary history and its empty-tail identity row;*
2. *classifier laws inherited componentwise from* hsame *on object histories and displayed hom histories;*
3. *hom admission by* $\mathsf{CategoryHomCarrier}(a, b, f)$*, equivalently by unary endpoints, a unary displayed tail, and the continuation-tail presentation of Definition 61.317;*
4. *stability under componentwise endpoint and hom-tail transport, using the unary-continuation stability law package of Definition 61.325;*
5. *ledger exactness for identity, binary composition, associativity, and unit rows, using the public* $\mathsf{Cat}^{\uparrow}$ *host-interface inventory of Definition 61.335.*

These rows bind the carrier, classifier, Cont *routes,* Pkg *provenance, and local* NameCert *ledger without adding any host category primitive.*

Proof. For habitation, take the empty unary history. The empty continuation row supplies the displayed identity carrier, so the object and identity coordinates are inhabited inside BHist.

The classifier laws are componentwise. Object comparison uses hsame on unary histories, and hom comparison uses hsame on displayed hom histories over the same endpoints, with endpoint transport supplied by the stability package.

Hom admission is the concrete correspondence of Definition 61.317. It identifies each displayed hom row with unary endpoints and a continuation morphism whose tail is classified as the displayed arrow history.

Composition and stability are exactly the rows named by Definition 61.325. The proof reads the identity, binary composition, associativity, unit, and endpoint-transport rows from the existing $\mathsf{Cat}^{\uparrow}$ surface and stores them in I, M, and T.

Finally, Definition 61.335 lists the public host-interface rows exported by the same unary-continuation carrier. Repacking those rows with the Pkg provenance and local naming ledger gives the five $\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}^{\uparrow}}$ obligations. □

Theorem 471.3 (Unary-continuation category standard bridge)**.** *The* $\mathsf{UnaryContinuationCategory}^{\uparrow}$ *packet presents the standard unary-continuation category interface over* BHist*: public objects are exactly unary histories, public arrows are exactly displayed* CategoryHomCarrier *rows, identity arrows are empty-tail continuation rows, composition is displayed* Cont *composition of tails, and equality of objects and arrows is componentwise* hsame*.*

The bridge reads back to the same rows. It exports no global category universe, no host equality of arrows, no limit or adjunction structure, and no morphism type beyond displayed BHist *histories.*

Proof. Read the carrier coordinates of Definition 471.1. The object row is UnaryHistory, the hom row is CategoryHomCarrier, the identity row is the empty-tail continuation ledger, and the composition row is the displayed Cont route.

The category laws are those already used by Definition 61.329 and inventoried in Definition 61.335. They supply associativity, left and right units, and transport compatibility on the same displayed histories.

The readback is literal because every public row is one of the carrier coordinates O, H, I, M, T, P, N. The bridge therefore exposes the unary-continuation category packet and nothing outside the BEDC history surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}\uparrow}$.

Constructive story: A unary-continuation category packet is generated from unary object histories, displayed CategoryHomCarrier arrows, empty-tail identity ledgers, displayed Cont composition tails, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}\uparrow}, \mathsf{seed})$

Scope closed: The seed surface names the concrete BEDC carrier rows for unary objects, hom carriers, identity tails, composition tails, transport rows, provenance, and local naming, but it does not close the paper axis beyond this carrier presentation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnaryContinuationCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a separate Lean namespace, host categories, arbitrary categorical constructions, limits, adjunctions, equivalences, or machine bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, classifier laws, composition ledger exactness, and transport stability for the named unary-continuation packet.

Chapter 472

A Concrete Naming Certificate for $\mathsf{TruthTotalReflection}^{\uparrow}$

The $\mathsf{TruthTotalReflection}^{\uparrow}$ interface records the finite boundary certificate associated with the truth-total reflection obstruction of Definition 9461.23 and Definition 9461.24. It packages sentence-code rows for CIC*, attempted reflection rows, diagonal obstruction rows, transport data, package provenance, and a local naming row. The truth-total reflection packet reads its extension-attempt and refusal boundary against the accepted and refused declaration surface of chapter 549, while adding the diagonal obstruction row that prevents a total internal truth classifier. The packet is a boundary witness: it records why a truth-total classifier cannot be exported as an internal BEDC object.

Definition 472.1 (Truth-total reflection NameCert carrier)**.** *A* $\mathsf{TruthTotalReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, A, D, H, C, P, N)$$

with the following displayed coordinates:

1. *S is a finite row of* CIC* *sentence codes, including the code rows inspected by a candidate reflection consumer;*
2. *A is the finite extension-attempt row recording the proposed* Ext_T *classifier reads against the sentence-code row S;*
3. *D is the diagonal obstruction row displaying the code for the sentence that reads the negation of its own attempted* Ext_T *classification;*
4. *H stores the componentwise* hsame *transports for the sentence-code, extension-attempt, and diagonal rows;*
5. *C stores the displayed* Cont *routes by which a consumer reads those rows;*
6. *P is the* Pkg *provenance over* BHist, hsame, Cont, CIC* *sentence coding, diagonal formation, and* $\mathsf{NameCert}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{TruthTotalReflection}^{\uparrow}}$ *row.*

The local classifier relates two such packets exactly when their displayed sentence-code, extension-attempt, diagonal, transport, route, provenance, and naming rows compare componentwise through H and C. It has no coordinate for a BEDC-internal truth predicate, a total truth classifier, or a metalanguage able to validate BEDC from inside BEDC.

Lean4 (definition) `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionUp`

Theorem 472.2 (Truth-total reflection boundary certificate)**.** *Every accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet records only the finite boundary evidence for truth-total reflection over* CIC^* *sentence codes. Its public surface consists of the rows* S, A, D, H, C, P, N *of Definition 472.1; therefore any consumer can read the attempted extension classifier and the diagonal obstruction through the displayed* hsame *and* Cont *routes, but cannot export an internal BEDC truth classifier for all BEDC sentence codes.*

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionUp taste gate boundary`

Proof. Project the accepted packet to the tuple $T = (S, A, D, H, C, P, N)$. The sentence-code row S supplies the finite CIC^* codes under inspection, and the extension-attempt row A records only the proposed classifier reads attached to those codes.

The diagonal row D is read through the same finite boundary. It records the obstruction used in Definition 9461.24: any truth-total reading of the attempted classifier can be turned against its own diagonal code.

Transport and consumer access are exhausted by H and C. These rows move and read the displayed sentence-code, extension-attempt, and diagonal coordinates, while P and N record provenance and local naming.

No remaining coordinate stores a BEDC-internal truth predicate or a total classifier over all represented BEDC sentences. Thus the accepted packet is a finite boundary certificate, not an internal truth-total reflection object. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionTasteGate single carrier alignment`

Theorem 472.3 (Truth-total reflection attempt-row stability)**.** *For every accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet*

$$T = (S, A, D, H, C, P, N),$$

componentwise hsame *transport through* H *preserves the sentence-code row* S*, the extension-attempt row* A*, and the diagonal row* D *as rows of the same finite boundary certificate. The transport may replay the displayed* Cont *routes in* C*, but it does not add a coordinate for a BEDC-internal truth predicate, a total truth classifier, or host validation of BEDC.*

Proof. Project the packet to its displayed rows. The transport row H is part of the carrier inventory and is specified only for the sentence-code, extension-attempt, diagonal, route, provenance, and naming coordinates.

Transporting S, A, D therefore keeps the same finite row classes and merely changes their presented history representatives. The continuation row C can replay reads of those transported rows, but it has no source for a classifier outside the packet.

Since the carrier definition contains no coordinate for internal truth, total classification, or host validation, componentwise transport cannot create one. The stabilized surface remains the finite attempted-reflection boundary. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionAttemptRowStability`

Theorem 472.4 (Truth-total reflection diagonal obstruction exactness)**.** *Every read of the diagonal row* D *in an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet factors through the displayed route*

$$S, A, H, C, P, N.$$

The obstruction is exact for the finite attempted reflection row A: it blocks only the displayed attempt to classify the sentence codes in S, and it does not assert a global theorem about all possible BEDC truth predicates or all host interpretations.

Proof. Read D from the carrier only after the sentence-code row S and the extension-attempt row A have been fixed. The diagonal row is the obstruction attached to that displayed attempt.

The rows H and C give the only transport and consumer routes by which the diagonal row can be moved or inspected. The provenance and naming rows P, N record the package in which that read occurs.

Thus a consumer reaches D through the route S, A, H, C, P, N. Because no other classifier row is present, the obstruction applies exactly to the finite attempt recorded by A and carries no broader host-level or total-truth conclusion. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionDiagonalObstructionExactness`

Theorem 472.5 (Truth-total reflection consumer surface exhaustion)**.** *Every consumer of an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet factors through*

$$S, A, D, H, C, P, N.$$

Such a consumer may inspect the sentence-code row, the attempted extension-classifier row, the diagonal obstruction row, componentwise transport, displayed continuation reads, provenance, and local naming. It cannot export BEDC-internal truth, total truth classification, host validation, or any classifier not recorded in A.

Proof. The carrier inventory lists exactly the rows S, A, D, H, C, P, N. Any accepted consumer read must therefore begin by projecting one of those coordinates and then using the displayed transport and continuation data.

The boundary theorem identifies A as the only attempted classifier row and D as the corresponding diagonal obstruction row. The stability theorem keeps these rows fixed under hsame transport, and the exactness theorem confines diagonal reads to the displayed attempt.

No remaining coordinate stores internal truth, total classification, host validation, or another classifier. Hence the consumer surface is exhausted by the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionConsumerSurfaceExhaustion`

Theorem 472.6 (Truth-total reflection carrier primitive scope)**.** *Carrier admission for an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet is exactly the finite packet*

$$S, A, D, H, C, P, N$$

over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The rows expose the finite* CIC* *sentence-code row S, the displayed extension-attempt row A, the diagonal obstruction row D, componentwise transport H, continuation reads C, provenance P, and the local naming row N. No additional carrier coordinate may supply internal truth, total truth classification, host validation, or a second reflection predicate.*

Proof. Project an accepted packet through Definition 472.1. The definition lists exactly S, A, D, H, C, P, N and identifies their primitive BEDC roles over BHist, hsame, Cont, Pkg, and NameCert.

Apply the boundary and consumer-exhaustion rows. Definition 472.2 confines the packet to finite boundary evidence, while Definition 472.5 states that every consumer read factors through the same displayed tuple.

Since no listed coordinate stores internal truth, total classification, host validation, or another reflection predicate, carrier admission is exhausted by the primitive finite packet. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionCarrierPrimitiveScope`

Theorem 472.7 (Truth-total reflection classifier transport scope)**.** *Classifier transport for* $\mathsf{TruthTotalReflection}^{\uparrow}$ *can move only sentence-code rows, extension-attempt rows, and diagonal obstruction rows through the displayed* H *and* C *rows of the accepted packet. It may replay the finite reads of* S, A, D *through componentwise* hsame *transport and displayed* Cont *routes, but it cannot transport a truth predicate, a total truth classifier, host validation of BEDC, or any classifier not recorded in* A*.*

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionClassifierTransportScope`

Proof. Start with the local classifier clause in Definition 472.1. It compares packets componentwise through the displayed transport and continuation rows, and the only classifier-attempt coordinate it can inspect is A.

Use Definition 472.3. Transport through H preserves the sentence-code row, extension-attempt row, and diagonal row as rows of the same finite boundary certificate, while C only replays their displayed reads.

The diagonal exactness and consumer-exhaustion rows then block escape from this scope. Definition 472.4 ties every diagonal read to the displayed attempt, and Definition 472.5 excludes internal truth, total classification, host validation, and classifiers absent from A. □

Theorem 472.8 (Truth-total reflection diagonal route scope)**.** *Every diagonal-obstruction read in an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet*

$$T = (S, A, D, H, C, P, N)$$

is scoped by the displayed sentence-code, extension-attempt, diagonal, transport, continuation, provenance, and naming rows, together with the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The route has no coordinate for an internal truth predicate, total truth classifier, host validator, or unrecorded classifier.*

Proof. Start from the boundary certificate Definition 472.2. It identifies the accepted packet as the finite tuple S, A, D, H, C, P, N and reads the diagonal obstruction only against the displayed sentence-code and extension-attempt rows.

Apply attempt-row stability and diagonal exactness, Definition 472.3 and Definition 472.4. Transport through H preserves the rows S, A, D, while every read of D factors through S, A, H, C, P, N.

Now add the primitive and transport scope rows, Definition 472.6 and Definition 472.7. They bind carrier admission and classifier movement to BHist, hsame, Cont, Pkg, and NameCert.

The consumer-exhaustion theorem Definition 472.5 removes all remaining escape routes. Since the packet has no row for internal truth, total truth classification, host validation, or any classifier outside A, the diagonal route is scoped exactly by the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionDiagonalRouteScope`

Theorem 472.9 (Truth-total reflection consumer route scope)**.** *Every consumer route for an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet factors through*

$$S, A, D, H, C, P, N$$

and through the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *primitives used by the diagonal-obstruction route. A consumer cannot substitute total truth classification, host validation, or an unrecorded classifier for the displayed finite packet.*

Proof. First use Definition 472.5. It states that every consumer read begins and ends inside the finite packet S, A, D, H, C, P, N.

The carrier primitive scope Definition 472.6 supplies the primitive boundary for that packet. The only admitted carrier rows are sentence codes, the extension attempt, the diagonal obstruction, componentwise transport, displayed continuations, provenance, and local naming.

Next combine Definition 472.7 with Definition 472.8. Classifier movement can replay only the recorded S, A, D rows through H and C, and the diagonal route cannot leave the same primitive scope.

Thus any consumer route factors through the displayed packet. Total truth classification, host validation, and unrecorded classifier rows are absent from the carrier and cannot be substituted for it. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionConsumerRouteScope`

Theorem 472.10 (Truth-total reflection obligation unblock package)**.** *The* $\mathsf{TruthTotalReflection}^{\uparrow}$ *obligation package consists exactly of the boundary certificate, attempt-row stability, diagonal obstruction exactness, and consumer-surface exhaustion rows for the packet*

$$S, A, D, H, C, P, N.$$

Together with the primitive carrier and classifier-transport scopes, these rows unblock the finite truth-total reflection obligation while excluding internal truth, total truth classification, host validation, and any classifier not recorded in A*.*

Proof. Assemble the previously displayed rows. Definition 472.2 supplies the finite boundary certificate, Definition 472.3 supplies stability of S, A, D under H, Definition 472.4 supplies the exact diagonal route, and Definition 472.5 exhausts consumer reads.

Attach the primitive rows from this section. Definition 472.6 fixes carrier admission as S, A, D, H, C, P, N, and Definition 472.7 fixes classifier movement through the displayed H and C rows.

The assembled package therefore closes the scoped obligation surface for the finite attempted-reflection packet. Because every row is read through the displayed carrier and no row stores internal truth, total classification, host validation, or a classifier outside A, the package has no escape route to those excluded objects. □

Theorem 472.11 (Truth-total reflection public consumer interface)**.** *Every public consumer of an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet must read the packet through the displayed rows*

$$S, A, D, H, C, P, N.$$

The consumer may inspect sentence codes, extension-attempt rows, diagonal-obstruction rows, componentwise transport, displayed continuation routes, package provenance, and local naming. It cannot project total truth, a BEDC-internal truth predicate, host validation, or any classifier outside the recorded attempt row A*.*

Proof. Begin with the carrier primitive scope. Definition 472.6 identifies the accepted carrier as the finite tuple S, A, D, H, C, P, N over the displayed BEDC primitives.

Apply consumer route scope and classifier transport scope. Definition 472.9 forces every consumer route through the same tuple, while Definition 472.7 confines classifier movement to S, A, D through H and C.

Finally use consumer-surface exhaustion. Definition 472.5 rules out internal truth, total classification, host validation, and unrecorded classifiers. Thus the public consumer interface is exactly the displayed refusal packet. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionPublicConsumerInterface`

Theorem 472.12 (Truth-total reflection root route totality)**.** *Every root consumer route for an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet factors through the finite rows*

$$S, A, D, H, C, P, N.$$

The route may inspect sentence codes, the attempted classifier row, the diagonal obstruction, componentwise transport, replay, provenance, and local naming, but it cannot export internal truth, total truth classification, host validation, or a classifier not recorded in A*.*

Proof. Begin with the boundary certificate and attempt-row stability. They expose S, A, D, H, C, P, N as the finite packet and keep the attempted classifier row attached to the same sentence-code and diagonal-obstruction rows.

Next use diagonal obstruction exactness and the public consumer interface. The diagonal read factors through the same route, and every public consumer is already confined to the displayed refusal packet.

Finally apply consumer surface exhaustion together with primitive scope and classifier transport scope. These results exhaust the carrier coordinates and leave no row for internal truth, total classification, host validation, or an unrecorded classifier. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionRootRouteTotality`

Theorem 472.13 (Truth-total reflection ledger non-escape)**.** *For an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet, the package and provenance ledgers* P, N *record only the public refusal boundary for the rows* S, A, D, H, C*. They cannot create a BEDC-internal truth predicate, total truth classification, host validation, or a classifier outside the displayed attempt row* A*.*

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionLedgerNonescape`

Proof. Start with refusal ledger exactness. The exact refusal ledger is carried by A, D over S, while P, N record provenance and local naming for that finite ledger.

Use classifier transport scope and diagonal route scope. Transport and replay return to the same displayed attempt and obstruction rows, so provenance cannot manufacture a classifier beyond the one recorded in A.

The boundary certificate, consumer surface exhaustion, and public consumer interface rule out the remaining escapes. Since internal truth, total truth classification, and host validation are absent from the carrier, the ledgers P, N cannot export them. □

Theorem 472.14 (Truth-total reflection TruthCert handoff)**.** *Any handoff from an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet to a TruthCert-style downstream boundary remains a finite refusal-boundary handoff through*

$$S, A, D, H, C, P, N.$$

It supplies the attempted classifier row and the diagonal obstruction as a public obstruction certificate, not a truth predicate, total truth classifier, host validator, or bridge certificate.

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionTruthCertHandoff`

Proof. First assemble the public boundary. The boundary certificate, attempt-row stability, and diagonal obstruction exactness place the handoff on S, A, D, H, C, P, N and identify A, D as the obstruction-bearing rows.

Next use consumer surface exhaustion and the public consumer interface. A TruthCert-style downstream boundary can consume the refusal packet only through the public consumer route, so the handoff remains finite and displayed.

Finally apply primitive scope and classifier transport scope. They keep the handoff inside the recorded carrier and prevent it from becoming a truth predicate, total truth classifier, host validator, or bridge certificate. □

Theorem 472.15 (Truth-total reflection refusal ledger exactness)**.** *In an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet, the extension-attempt row A and diagonal row D are the exact refusal ledger for truth-total reflection attempts over the displayed sentence-code row S. The ledger records the attempted classifier read and the diagonal obstruction to that attempt, and it does not include an additional total truth predicate, a second classifier row, or a host-level validation route.*

Proof. Read the attempt row through Definition 472.3. Transport preserves S, A, D as rows of the same finite boundary certificate, so A remains the displayed extension attempt.

Read the diagonal row through Definition 472.4. The diagonal obstruction factors through S, A, H, C, P, N and is exact for the finite attempted reflection row A.

The boundary and route-scope theorems then exclude hidden rows. Definition 472.2 and Definition 472.8 leave no coordinate for total truth, a second classifier, or host validation. Hence A and D are the exact refusal ledger. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionRefusalLedgerExactness`

Theorem 472.16 (Truth-total reflection downstream boundary package)**.** *The reusable downstream package exported by* $\mathsf{TruthTotalReflection}^{\uparrow}$ *consists exactly of the boundary certificate, attempt-row stability, diagonal exactness, refusal-ledger exactness, public consumer interface, and consumer-surface exhaustion for the packet*

$$S, A, D, H, C, P, N.$$

Any downstream chapter that consumes this package receives a finite refusal boundary for truth-total attempts, not a BEDC-internal truth predicate, a total truth classifier, a host validator, or a bridge certificate.

Proof. First assemble the scoped rows. Definition 472.2 gives the boundary certificate, Definition 472.3 keeps S, A, D stable, and Definition 472.4 fixes diagonal exactness for the displayed attempt.

Next add the public rows. Definition 472.15 identifies A, D as the exact refusal ledger, while Definition 472.11 forces every public consumer through S, A, D, H, C, P, N.

Finally apply consumer-surface exhaustion and the obligation package. Definition 472.5 and Definition 472.10 rule out internal truth, total classification, host validation, unrecorded classifiers, and bridge status. The remaining object is exactly the reusable downstream boundary package. □

Lean4 ✓ `BEDC.Derived.TruthTotalReflectionUp.TruthTotalReflectionDownstreamBoundaryPackage`

Theorem 472.17 (Truth-total reflection root public readiness)**.** *Every downstream public consumer of an accepted* $\mathsf{TruthTotalReflection}^{\uparrow}$ *packet reads exactly the public refusal route*

$$S, A, D, H, C, P, N$$

through the existing boundary certificate, attempt-row stability, diagonal exactness, consumer exhaustion, public interface, root route totality, ledger non-escape, TruthCert handoff, refusal ledger exactness, and downstream boundary package. No internal truth predicate, total truth classifier, host validator, or unrecorded classifier is exported by that public route.

Proof. First assemble the public boundary package. The downstream boundary package collects the boundary certificate, attempt-row stability, diagonal exactness, refusal ledger exactness, public consumer interface, and consumer-surface exhaustion for S, A, D, H, C, P, N.

Next attach root route totality and ledger non-escape. They keep every root public read on the same finite carrier, with P, N recording only the refusal boundary rather than a host-level validation route.

Finally use the TruthCert handoff. A downstream TruthCert-style consumer receives the displayed obstruction certificate and nothing else; internal truth predicates, total truth classifiers, host validators, and unrecorded classifiers have no coordinate in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TruthTotalReflection}\uparrow}$.

Constructive story: A $\mathsf{TruthTotalReflection}^{\uparrow}$ packet is generated from finite CIC^* sentence-code rows, extension-attempt rows, diagonal obstruction rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TruthTotalReflection}\uparrow}$, publicClosed)

Scope closed: Definition 472.1, Definition 472.2, Definition 472.3, Definition 472.4, Definition 472.5, Definition 472.11, Definition 472.12, Definition 472.13, Definition 472.14, Definition 472.15, Definition 472.16,
Lean4 (definition) `BEDC.Derived.TruthTotalReflectionUp.taste gate`
, and the displayed TasteGate single-carrier alignment marker bind the finite public refusal package for truth-total reflection attempts over CIC^* sentence codes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TruthTotalReflection}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close an internal BEDC truth predicate, a truth-total classifier, a host-runtime validator, or a public standard bridge.

Upgrade path: Bridge closure requires a checked public bridge for the same truth-total reflection refusal packet.

Chapter 473

A Concrete Naming Certificate for $\mathsf{CertificateCompiler}^{\uparrow}$

The $\mathsf{CertificateCompiler}^{\uparrow}$ interface reads compilation as a finite morphism between two local naming certificates. It records only the source certificate, target certificate, displayed compiler graph, edge landing, continuation compatibility, classifier transport, package provenance, and local naming row. It does not import a host compiler semantics, a denotational equality of programs, or an unbounded trace comparison.

Definition 473.1 (CertificateCompiler carrier)**.** *A* $\mathsf{CertificateCompiler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, G, E, C, H, P, N)$$

with the following visible rows:

1. *S is the source* $\mathsf{NameCert}$ *row whose carrier and classifier are the inputs read by the compiler graph;*
2. *T is the target* $\mathsf{NameCert}$ *row whose carrier and classifier receive the compiled outputs;*
3. *G is the finite compiler graph, displaying source-to-target graph edges rather than a host-level function space;*
4. *E is the edge-landing row, asserting that every displayed graph edge starts in the source carrier and lands in the target carrier;*
5. *C is the continuation-compatibility row, pairing each source* Cont *route used by the graph with the corresponding target* Cont *route;*
6. *H records the componentwise* hsame *transports for the source certificate, target certificate, graph edge, edge landing, continuation row, package row, and local certificate row;*
7. *P names* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the source certificate, the target certificate, and the local compiler graph shape;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *row.*

The packet is a finite compiler-certificate surface. It contains no ambient program equality, no host optimiser theorem, and no semantic model beyond the displayed source and target naming certificates.

Lean4 (definition) `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier`

Definition 473.2 (CertificateCompiler classifier)**.** *The classifier of a* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet compares graph edges in pairs. Two displayed edges are classifier-compatible when they read the same source classifier row in S, land through E in the target carrier of T, and their target classifier rows are connected by the* hsame *and* Cont *routes recorded in H and C.*

Thus classifier membership is not a judgment about arbitrary compiler outputs. It is a finite relation on paired graph edges with a common source classifier row and an explicitly transported target classifier row.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerClassifier semantic name certificate`

Theorem 473.3 (CertificateCompiler endpoint certificate availability)**.** *Every displayed compiler edge in an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet $K = (S, T, G, E, C, H, P, N)$ exposes both endpoint* NameCert *rows before carrier landing is consumed. The source certificate row S supplies the admitted input carrier and classifier row, the target certificate row T supplies the landing carrier and classifier row, and the displayed graph row G has no admissible edge that bypasses either endpoint certificate.*

Proof. Unfold the carrier of Definition 473.1. The first two visible rows of K are precisely the source certificate S and target certificate T, while G is only a displayed finite graph between those certificate rows.

The carrier-landing row E is read after these endpoint rows. Its role is to assert that each graph edge starts in the source carrier named by S and lands in the target carrier named by T.

Unfold the classifier of Definition 473.2. A compatible edge comparison also reads the source classifier row in S and the target classifier row in T, so endpoint certificates remain available before any carrier landing or paired-edge comparison is accepted.

Since the packet contains no host compiler semantics, program equality, or unbounded trace coordinate, endpoint availability is exhausted by the displayed rows S, T, G, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint certificate availability`

Theorem 473.4 (CertificateCompiler displayed edge witness totality)**.** *Every accepted edge comparison in a* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet $K = (S, T, G, E, C, H, P, N)$ is witnessed by the displayed graph row G, the carrier-landing row E, the componentwise* hsame *transports in H, and the continuation compatibility rows in C. No comparison can be witnessed by an external compiler meaning, ambient program equality, or an unlisted target behavior.*

Proof. Unfold Definition 473.1. The only edge source in the packet is the finite graph row G, and the only target-carrier landing witness for such an edge is the row E.

Unfold Definition 473.2. An accepted comparison must share the source classifier row, land through E, and transport the target classifier by the componentwise hsame entries recorded in H.

It remains to account for route compatibility. The classifier definition reads the paired Cont routes recorded in C, so a comparison is not accepted until the displayed source route and displayed target route have both been replayed.

The carrier has no coordinate for host compiler semantics, program equality, or unrecorded target behavior. Thus G, E, H, C are the total displayed witness surface for accepted edge comparison. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerClassifier displayed edge witness totality`

Theorem 473.5 (CertificateCompiler KernelMorphism route coverage)**.** *Every accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *edge route whose source and target certificates are read by a downstream* KernelMorphismUp *dependency must expose the source certificate row S, target certificate row T, displayed graph row G, edge-landing row E, continuation row C, componentwise* hsame *transport row H,* Pkg *provenance row P, and local* NameCert *row N before the route is consumed as a kernel-morphism route.*

Proof. Unfold the carrier and classifier of Definition 473.1 and Definition 473.2. The route inventory is exactly S, T, G, E, C, H, P, N, with graph edges landing through E and classifier transport through H and C.

Use endpoint certificate availability and displayed-edge witness totality. Theorems 473.3 and 473.4 force source and target certificates to be read before the graph edge is consumed, and force each accepted comparison to expose graph, landing, transport, and continuation witnesses.

The downstream consumer is the KernelMorphismUp chapter of chapter 620. It may consume only this finite compiler route together with its own graph-admission rows, so no host compiler semantics, ambient program equality, or unrecorded target behavior becomes a kernel-morphism route. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier kernel morphism route coverage`

Theorem 473.6 (CertificateCompiler consumer replay obligation triad)**.** *Every consumer replay of an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet must read three local obligations before package provenance or local naming is used: carrier landing through G and E, classifier transport through S, T, H, and continuation compatibility through C. Package and* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *rows may repack only this displayed triad.*

Proof. Unfold the carrier of Definition 473.1. A consumer replay first sees the graph row G and the landing row E, so the replayed edge has a displayed source and a displayed target before any package row is read.

Unfold the classifier of Definition 473.2. The replayed comparison next reads the source classifier row in S, the target classifier row in T, and the componentwise transport entries in H.

The third obligation is route compatibility. The row C pairs the source Cont route used by the graph with the target Cont route followed after compiling, and this compatibility is required before the replay can be named locally.

Only after these three obligations have been read may P and N repack the compiler packet. Because no host compiler semantics or program equality coordinate exists in the carrier, package and local naming cannot certify anything beyond carrier landing, classifier transport, and continuation compatibility. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier consumer replay obligation triad`

Theorem 473.7 (CertificateCompiler NameCert obligations)**.** *For the carrier and classifier of Definition 473.1 and Definition 473.2, the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier landing: every edge of G reads a source row admitted by S and lands in a target row admitted by T through E;*
2. *classifier transport: paired graph edges with the same source classifier row transport classifier membership to the target classifier row;*

3. *continuation compatibility: every source* Cont *route used by a graph edge is paired in C with the target* Cont *route followed after compiling;*
4. *identity compiler row: the diagonal graph on a certificate is accepted with carrier landing, classifier transport, and continuation compatibility read from the same certificate;*
5. *composition stability: composable accepted compiler packets compose by relational graph composition and preserve the same landing, classifier, and continuation rows.*

Proof. Carrier landing is the projection of E from the finite packet $K = (S, T, G, E, C, H, P, N)$. Since E names the source and target carrier rows for each displayed edge, no edge can be consumed without first reading S, T, and the corresponding graph row in G.

Classifier transport uses the paired-edge classifier. The common source classifier row is read in S, and the target classifier row is obtained only by replaying the componentwise hsame entries in H and the displayed continuation routes in C.

Continuation compatibility is exactly the second projection of C. Each source route used by a graph edge has a named target route, so a downstream consumer cannot substitute an unrecorded control-flow or semantic route.

For the identity row, take $S = T$ and let G be the displayed diagonal graph. The landing, classifier, and continuation rows are then the carrier, classifier, and route rows already present in the same certificate.

For composition, compose graph edges relationally through the middle certificate. The first packet supplies landing and classifier transport into the middle row, and the second supplies the same data out of it; composing the two continuation rows gives the target route. Repacking these rows gives the five local NameCert obligations for the composite packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier namecert obligations`

Theorem 473.8 (CertificateCompiler composition stability)**.** *Given accepted* CertificateCompiler$^{\uparrow}$ *packets from S to M and from M to T, their relational graph composite is an accepted* CertificateCompiler$^{\uparrow}$ *packet from S to T. The composite reads only the two displayed graph rows, their edge-landing rows, continuation-compatible routes, componentwise* hsame *transports, package provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier composition stability`

Proof. Start with the two accepted packets and project their carrier rows. The first packet lands graph edges in the middle certificate, and the second starts from that same middle certificate.

Classifier transport composes through the middle row. A common source classifier row is transported to the middle classifier by the first packet, then from the middle classifier to the target classifier by the second packet.

Continuation compatibility composes in the same order. The source route is paired with the middle route by the first packet, and that middle route is paired with the target route by the second packet.

The resulting graph, landing row, classifier transport, continuation row, transport row, package row, and local naming row are finite projections of the two input packets. Therefore the composite exports no host compiler semantics and no unrecorded target behavior. □

Theorem 473.9 (CertificateCompiler identity ledger)**.** *For every local naming certificate S, the diagonal graph on S is an accepted* CertificateCompiler$^{\uparrow}$ *identity packet. Its ledger is exhausted by the carrier, classifier,* hsame, Cont, Pkg, *and* NameCert *rows already displayed by S.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerIdentityLedger`

Proof. Use S as both source and target certificate. The diagonal graph has one displayed edge for each admitted source row and lands in the same row.

The classifier transport is the local classifier row of S read twice. The required hsame transport is reflexive at the displayed row, and the continuation route is the same route already recorded in S.

The package and local naming rows are inherited from S with no further coordinate. Hence the identity compiler packet has exactly the stated ledger and does not create a separate compiler semantics. □

Theorem 473.10 (CertificateCompiler edge-classifier exactness)**.** *For an accepted* CertificateCompiler$^{\uparrow}$ *packet* $K = (S, T, G, E, C, H, P, N)$*, a displayed edge comparison is accepted exactly when it reads the source classifier row in* S*, lands through* E *in the target carrier of* T*, transports the target classifier by the componentwise* hsame *entries of* H*, and follows the paired* Cont *routes recorded in* C*. No host compiler semantics, program equality, or unrecorded target behavior can witness edge-classifier compatibility.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerClassifier edge classifier exactness`

Proof. Project the accepted packet to the paired-edge classifier of Definition 473.2. The source classifier row is the first required comparison coordinate, so two edges cannot be compared after forgetting their source certificate.

Next read the edge-landing row E. It supplies the target-carrier landing for each displayed edge and fixes the target certificate row in T that the comparison may inspect.

The remaining data are the transport and route rows. The hsame entries in H move target classifier rows componentwise, and the Cont routes in C record the control path followed after compiling. These are exactly the displayed rows named by the classifier definition.

Since the carrier has no coordinate for host compiler semantics, program equality, or unrecorded target behavior, the displayed graph, landing, classifier, and continuation rows exhaust edge-classifier exactness. □

Theorem 473.11 (CertificateCompiler endpoint route determinacy)**.** *Let two accepted* CertificateCompiler$^{\uparrow}$ *packets share the source certificate row, target certificate row, displayed graph row, edge-landing row, componentwise* hsame *transports, and paired* Cont *route row. Then they determine the same endpoint route from the accepted source certificate to the accepted target certificate. No host compiler semantics, program equality, or unrecorded target behavior can distinguish the two endpoint routes.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint route determinacy`

Proof. First expose the endpoint certificates by Definition 473.3. The shared source and target certificate rows fix the admitted input carrier, source classifier row, target carrier, and target classifier row before any edge landing is consumed.

Next use displayed edge witness totality, Definition 473.4. Since the displayed graph row, edge-landing row, componentwise hsame transports, and paired Cont route row are shared, every accepted edge comparison has the same finite witness surface in both packets.

Finally apply edge-classifier exactness, Definition 473.10. The classifier accepts exactly by reading the source classifier row, landing through E, transporting the target classifier through H, and replaying the paired Cont routes in C. Therefore the endpoint route is determined by

the shared displayed rows, and host compiler semantics, program equality, and unrecorded target behavior have no coordinate on which to separate the packets. □

Theorem 473.12 (CertificateCompiler identity and composite ledger exhaustion)**.** *The identity packets of Definition 473.9 and the composite packets of Definition 473.8 exhaust their ledgers by displayed graph rows, edge-landing rows, classifier-transport rows, continuation rows,* Pkg *provenance, and local* NameCert *rows. Triple composites in Definition 473.13 add no further ledger coordinate beyond the three displayed graph edges and their two middle landing matches.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier source target landing exhaustion`

Proof. For the identity packet, use the diagonal graph on the source certificate. The graph, landing, classifier, continuation, provenance, and local naming rows are inherited from the same certificate, as stated in Definition 473.9.

For a binary composite, apply Definition 473.8. The composite graph is relational graph composition of the two displayed graphs, and its landing, classifier, and continuation rows are obtained by reading the corresponding rows through the middle certificate.

For triple composites, use Definition 473.13. Both associations expose the same triple of graph edges, the same two middle landing matches, the same ordered classifier transport, and the same continuation endpoint. Repacking with Pkg and NameCert rows therefore adds no host compiler coordinate. □

Theorem 473.13 (CertificateCompiler triple composite associativity)**.** *Let*

$$K_{01} = (S_0, S_1, G_{01}, E_{01}, C_{01}, H_{01}, P_{01}, N_{01}),$$

$$K_{12} = (S_1, S_2, G_{12}, E_{12}, C_{12}, H_{12}, P_{12}, N_{12}),$$

and

$$K_{23} = (S_2, S_3, G_{23}, E_{23}, C_{23}, H_{23}, P_{23}, N_{23})$$

be accepted $\mathsf{CertificateCompiler}^{\uparrow}$ *packets in the sense of Definition 473.1 and Definition 473.7. Suppose the displayed middle naming rows agree exactly as composability data: the target* NameCert *row of* K_{01} *is the source* NameCert *row of* K_{12}*, and the target* NameCert *row of* K_{12} *is the source* NameCert *row of* K_{23}*. Then the two iterated relational graph composites*

$$(K_{23} \circ K_{12}) \circ K_{01} \qquad \textit{and} \qquad K_{23} \circ (K_{12} \circ K_{01})$$

classify by the same displayed S_0*-to-*S_3 *compiler graph, the same edge-landing row, the same classifier-transport row, and the same continuation-compatibility row. Their common displayed graph consists exactly of triples of graph edges*

$$(g_{01}, g_{12}, g_{23})$$

with g_{01} *in* G_{01}*,* g_{12} *in* G_{12}*,* g_{23} *in* G_{23}*, the landing target of* g_{01} *equal to the source row read by* g_{12} *in* S_1*, and the landing target of* g_{12} *equal to the source row read by* g_{23} *in* S_2*.*

Proof. Project the three accepted packets to their displayed graph, landing, classifier, continuation, transport, provenance, and local naming rows. By the composability hypotheses, the middle target row of the first packet is the source row consumed by the second packet, and the middle target row of the second packet is the source row consumed by the third packet. Thus every edge of either

iterated composite is represented by the same finite triple (g_{01}, g_{12}, g_{23}) with the two displayed middle landings named in the statement.

The graph rows agree because relational composition of finite displayed graph edges is associative at the level of witnesses. The left association first records the pair (g_{01}, g_{12}) and then appends g_{23}; the right association first records (g_{12}, g_{23}) and then prefixes g_{01}. After forgetting the temporary pairing, both ledgers contain exactly the same triple and the same two middle matching rows. No extra graph edge can appear, since Definition 473.8 forms a composite only from displayed edges of the two input packets.

The edge-landing rows agree by the same projection. In the left association, E_{01} lands g_{01} in S_1, the composite of E_{12} and E_{23} lands the remaining route in S_3, and the S_2 match is displayed inside that second composite. In the right association, the first composite displays the S_1 and S_2 matches before E_{23} lands in S_3. Both descriptions therefore expose the identical landing ledger

$$S_0 \xrightarrow{g_{01}} S_1 \xrightarrow{g_{12}} S_2 \xrightarrow{g_{23}} S_3.$$

Classifier transport is also independent of the association. Starting from a source classifier row in S_0, the first packet transports it to the displayed classifier row in S_1, the second transports that row to S_2, and the third transports the result to S_3. The left association packages the last two transports before composing with the first; the right association packages the first two transports before composing with the third. Since the classifier of Definition 473.2 is the paired-edge transport read through the componentwise hsame entries and the displayed continuation rows, both packages replay the same three componentwise transport entries H_{01}, H_{12}, and H_{23} in the same source-to-target order.

For continuation compatibility, the three accepted packets provide displayed routes from the S_0 route to the S_1 route, from the S_1 route to the S_2 route, and from the S_2 route to the S_3 route. The left association composes the last two routes and then the first route; the right association composes the first two routes and then the last route. By the relational continuation associativity of Definition 21.29, the two iterated continuation rows have the same endpoint route up to the displayed hsame transport already recorded in the packets. Hence they classify as the same continuation-compatibility row.

Combining the four projections gives one finite S_0-to-S_3 compiler packet surface: the triple graph, the three-step landing ledger, the ordered classifier transport through S_1 and S_2, and the associated continuation route. Both iterated composites read exactly this surface, so they classify by the same displayed rows and introduce no host compiler semantics, no ambient program equality, and no unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier triple composite associativity`

Theorem 473.14 (CertificateCompiler left identity unit)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet, and let I_S be the diagonal identity packet on the source* NameCert *row S supplied by Definition 473.9. Then the source-side composite $K \circ I_S$ is classified by the same displayed graph G, the same edge-landing row E, the same paired-edge classifier transport, the same continuation-compatibility row C, the same* Pkg *provenance row P, and the same local* NameCert *row N as K.*

Thus precomposing an accepted compiler packet with the diagonal source identity adds no host compiler semantics, no ambient program equality, and no unrecorded target behavior.

Proof. Use Definition 473.9 to read I_S as the diagonal packet whose graph, landing, classifier, continuation, provenance, and local naming rows are all inherited from S.

Apply Definition 473.8 to the composable pair (I_S, K). The composite graph contains exactly the displayed edges of G, because the diagonal source edge contributes only the source row that G already reads.

The classifier of Definition 473.2 then transports through the same componentwise hsame entries recorded in H, and the continuation route is the original route of C with a diagonal prefix. By Definition 21.29, that prefix changes no displayed endpoint route.

Repacking the Pkg row and the local NameCert row therefore returns the same finite compiler packet surface as K. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier left identity public route certificate`

Theorem 473.15 (CertificateCompiler right identity unit)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet, and let I_T be the diagonal identity packet on the target* NameCert *row T supplied by Definition 473.9. Then the target-side composite $I_T \circ K$ is classified by the same displayed graph G, the same edge-landing row E, the same paired-edge classifier transport, the same continuation-compatibility row C, the same* Pkg *provenance row P, and the same local* NameCert *row N as K.*

Thus postcomposing an accepted compiler packet with the diagonal target identity adds no host compiler semantics, no ambient program equality, and no unrecorded target behavior.

Proof. Read the target identity packet from Definition 473.9. Its displayed graph is diagonal on T, and its landing, classifier, continuation, provenance, and local naming rows are the target-side rows already available to K.

Apply Definition 473.8 to (K, I_T). Every composite edge is an edge of G followed by the diagonal target edge, so the finite graph row is again G after forgetting the diagonal witness.

The edge landing remains E, since the target identity lands only in the same target row. The classifier transport is the row of Definition 473.2 followed by diagonal componentwise hsame transport, and the continuation route is the row of C followed by the target diagonal route.

Use Definition 21.29 to identify that diagonal suffix with the original endpoint route. The Pkg and local NameCert rows are unchanged, so the target-side identity composite classifies by the same packet surface as K. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier right identity unit`

Theorem 473.16 (CertificateCompiler finite category surface)**.** *The accepted* CertificateCompiler$^{\uparrow}$ *packets form a finite category-like surface over local* NameCert *rows: each local row has the diagonal identity packet of Definition 473.9; each composable pair has the binary composite packet of Definition 473.8; each composable triple has the common associative replay surface of Definition 473.13; and every accepted packet satisfies the source and target unit laws of Definition 473.14 and Definition 473.15.*

The surface is finite and local: its objects are the displayed local NameCert *rows and its morphism packets are exactly the displayed graph, landing, classifier-transport, continuation,* Pkg*, and local* NameCert *rows recorded by the certificate compiler carrier.*

Proof. Identities are supplied by the diagonal ledger of Definition 473.9. Binary composition is supplied by Definition 473.8, whose output is again a displayed compiler packet over the same row types.

Associativity is the finite triple replay statement of Definition 473.13. It says that the two iterated composites expose the same graph triple, landing ledger, ordered classifier transport, and continuation endpoint.

The unit laws are exactly Definition 473.14 and Definition 473.15. Both identify the identity composite with the original displayed graph, landing, classifier, continuation, Pkg, and local NameCert rows.

These rows are all read through the classifier of Definition 473.2. Hence the resulting surface is a finite local NameCert-morphism interface rather than a claim about host compiler semantics, ambient program equality, optimiser correctness, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier finite category surface`

Theorem 473.17 (CertificateCompiler route replay scope)**.** *For an accepted* CertificateCompiler$^{\uparrow}$ *packet*

$$K = (S, T, G, E, C, H, P, N),$$

every accepted compiler read replays through the displayed compiler graph G, edge-landing row E, paired-edge classifier row, continuation row C, transport row H, package provenance P, and local NameCert *row N. No consumer of K can replace that replay by host compiler semantics, ambient program equality, optimiser correctness, or unrecorded target behavior.*

Proof. Begin with the carrier of Definition 473.1. Its only displayed compiler coordinates are the source and target naming rows S, T, the graph G, the landing row E, the continuation row C, the transport row H, the package row P, and the local naming row N.

Apply the classifier of Definition 473.2 and the edge-classifier exactness theorem. They force an accepted read to inspect the graph edge, its source classifier row, its landing in the target carrier, and the componentwise hsame and Cont routes.

The identity ledger and composition stability theorems show that replay is stable under identity and binary composition, while Definition 473.13 gives the same replay surface for triple composites. Therefore every accepted compiler read remains inside the displayed finite rows of K. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier route replay scope`

Theorem 473.18 (CertificateCompiler middle-row handoff)**.** *Let $K_{01} = (S_0, S_1, G_{01}, E_{01}, C_{01}, H_{01}, P_{01}, N_{01})$ and $K_{12} = (S_1, S_2, G_{12}, E_{12}, C_{12}, H_{12}, P_{12}, N_{12})$ be composable accepted* CertificateCompiler$^{\uparrow}$ *packets. Their composite can hand off from the first compiler packet to the second only through the displayed middle* NameCert *row S_1, together with the edge-landing, classifier-transport, and continuation rows that read S_1.*

The handoff exports no independent middle compiler semantics, program equivalence witness, optimiser correctness proof, or unrecorded target behavior.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier middle row handoff`

Proof. Project the two accepted carriers. The target naming row of K_{01} and the source naming row of K_{12} are the same displayed row S_1, and this is the only middle carrier coordinate available to both packets.

Use Definition 473.10. It makes every edge comparison pass through the landing row into the target carrier, the transported classifier row, and the continuation route over the displayed middle certificate.

Composition stability then forms the composite graph only from pairs of displayed edges whose middle landing reads S_1. The identity ledger supplies the degenerate case, and triple composite associativity shows that repeated handoffs still expose the same middle rows rather than a hidden compiler coordinate. □

Theorem 473.19 (CertificateCompiler finite-graph nonescape)**.** *For an accepted* CertificateCompiler$^{\uparrow}$ *carrier* $K = (S, T, G, E, C, H, P, N)$*, every packet exported by the compiler certificate is a finite graph packet over the displayed rows* S, T, G, E, C, H, P, N*. It exports no host compiler semantics, no program equivalence relation, no optimiser correctness theorem, and no target behavior that is absent from the displayed graph, landing, classifier, continuation, provenance, and local naming rows.*

Proof. Read the carrier and classifier rows. The carrier lists only S, T, G, E, C, H, P, N, and the classifier reads graph edges through displayed landing, hsame, and Cont rows.

The edge-classifier exactness theorem prevents a consumer from accepting an edge by an external compiler semantics. The identity ledger and composition stability theorems keep identity and composite packets inside the same finite graph surface.

Finally apply Definition 473.13. Triple composites expose only triples of graph edges and their two displayed middle landing matches. Since no excluded compiler or target object appears in those rows, there is no escape from the finite graph packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier finite graph nonescape`

Theorem 473.20 (CertificateCompiler carrier dependency scope)**.** *For an accepted* CertificateCompiler$^{\uparrow}$ *carrier*

$$K = (S, T, G, E, C, H, P, N),$$

every carrier read is exhausted by the source and target NameCert *rows* S, T*, the displayed graph* G*, edge landing* E*, continuation compatibility* C*, componentwise* hsame *transport* H*,* Pkg *provenance* P*, and the local compiler* NameCert *row* N*. No additional host compiler semantics, program equality, optimiser theorem, decompiler row, or unrecorded target behavior is a dependency of the carrier surface.*

Proof. Start with Definition 473.1. Its coordinates are exactly the source certificate, target certificate, graph, edge landing, continuation compatibility, transport, package provenance, and local naming rows.

Apply the NameCert obligations of Definition 473.7. Carrier landing, classifier transport, continuation compatibility, identity, and composition all read from the same displayed rows.

Use Definition 473.10 and Definition 473.12 to exclude further witnesses. They state that accepted edges, identity packets, and composites are exhausted by graph, landing, classifier, continuation, provenance, and local naming rows, so the carrier has no hidden dependency. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier dependency scope`

Theorem 473.21 (CertificateCompiler classifier-route dependency scope)**.** *For an accepted* CertificateCompiler$^{\uparrow}$ *packet, edge classification depends only on the source classifier row read in* S*, the target landing row supplied by* E*, componentwise* hsame *transport through* H*, and the paired source and target* Cont *routes recorded in* C*. In particular, a classifier route cannot be witnessed by host compiler*

semantics, program equality, optimiser correctness, or a target behavior absent from the displayed landing route.

Proof. Read the classifier definition in Definition 473.2. It compares paired graph edges by their common source classifier row, their target landing, and the transported target classifier row.

Apply Definition 473.10. The exactness statement identifies the allowed route data as the source classifier, edge landing, componentwise hsame entries, and paired Cont routes.

Composition stability and the identity-composite ledger exhaustion theorem show that identity and composite routes use the same dependency list. Thus route classification remains inside the displayed compiler packet and cannot import an external compiler witness. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerClassifier route dependency scope`

Theorem 473.22 (CertificateCompiler composite dependency scope)**.** *Binary composites and triple composites of accepted* CertificateCompiler$^{\uparrow}$ *packets introduce no dependency beyond displayed graph edges, edge-landing rows, classifier-transport rows, continuation rows,* Pkg *provenance, local* NameCert *rows, and the middle landing matches required for composition. The composite surface therefore has no host compiler semantics, ambient program equality, optimiser correctness theorem, or unrecorded target row.*

Proof. For a binary composite, apply Definition 473.8. The composite graph is relational graph composition of the two displayed graphs, and all landing, classifier, and continuation data are read through the middle certificate.

For identities and binary ledgers, use Definition 473.9 and Definition 473.12. They identify the available ledger rows as graph, landing, classifier, continuation, package provenance, and local naming rows.

For triple composites, apply Definition 473.13. The two associations expose the same three graph edges and the same two middle landing matches, so no association can add a host compiler or target-semantic coordinate. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier left identity semantic name certificate`

Theorem 473.23 (CertificateCompiler source-target landing exhaustion)**.** *Every accepted* CertificateCompiler$^{\uparrow}$ *packet has each accepted compiler edge exhausted by the finite rows S, T, G, E, C, H, P, N. The source and target* NameCert *rows S and T are the only admission points for the graph edge, the landing row E is the only target-carrier landing, and no accepted edge can bypass those rows through host compiler semantics, ambient program equality, optimiser correctness, or unrecorded target behavior.*

Proof. Start with the carrier coordinates of Definition 473.1. An accepted packet displays the source row S, target row T, graph row G, landing row E, continuation row C, componentwise transport row H, provenance row P, and local naming row N.

Apply the classifier of Definition 473.2 and Definition 473.10. These force an accepted edge to be read through its graph entry, source classifier row, target landing, transported classifier row, and displayed continuation route.

Use Definition 473.21 and Definition 473.22 to close the route under identity, binary composition, and triple composition. The resulting accepted edge surface is exactly the displayed source-target landing route and has no external compiler coordinate. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerClassifier source target landing exhaustion`

Theorem 473.24 (CertificateCompiler public consumer boundary)**.** *Every downstream consumer of an accepted* CertificateCompiler$^{\uparrow}$ *packet factors through graph landing, classifier transport, and continuation rows. Thus a public read can inspect only the graph row G, the target landing row E, the classifier transport row H, the continuation row C, the* Pkg *provenance row P, and the local* NameCert *row N over the source and target naming rows S, T.*

Proof. Read the packet through Definition 473.1. Its public surface presents graph, landing, continuation, transport, provenance, and local naming rows over the source and target certificates.

Apply Definition 473.2, Definition 473.10, and Definition 473.21. They identify the consumer-visible classifier route with graph landing, transported classifier comparison, and continuation compatibility.

Finally use Definition 473.22. Composite consumers read only the same graph-landing and classifier-continuation rows, so downstream access factors through the displayed finite packet surface. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier public consumer boundary`

Theorem 473.25 (CertificateCompiler downstream boundary sibling link)**.** *Every adjacent-stage boundary read in chapter 574 consumes two accepted* CertificateCompiler$^{\uparrow}$ *packets through their endpoint availability, displayed edge witness totality, and continuation compatibility rows before the middle-stage compatibility row is read. Thus the downstream boundary is a finite sibling consumer of the compiler packets rather than a host compiler semantics, ambient program equivalence, optimizer correctness theorem, or unrecorded target behavior.*

Proof. First expose the endpoints of the two compiler packets. Definition 473.3 supplies the source, middle, and target NameCert rows, while Definition 473.4 forces each displayed edge to land through its graph and edge-landing rows.

Next replay the local consumer obligations. Definition 473.6 and Definition 473.24 make the boundary read pass through graph landing, classifier transport, and continuation compatibility for each packet before any shared middle row can be inspected.

Finally read the sibling boundary chapter. The adjacent-stage packet in chapter 574 consumes the two compiler packets exactly at that middle certificate interface, so its compatibility row has no coordinate from host compiler semantics, ambient program equivalence, optimizer correctness, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier downstream boundary sibling link`

Theorem 473.26 (CertificateCompiler finite-route non-escape)**.** *The finite* CertificateCompiler$^{\uparrow}$ *route exports no host compiler semantics, optimizer correctness theorem, ambient program equality, or unrecorded target behavior. Every accepted identity, edge, binary composite, triple composite, and downstream consumer read remains inside the displayed rows S, T, G, E, C, H, P, N and their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Proof. The carrier of Definition 473.1 supplies the finite row list S, T, G, E, C, H, P, N. No other compiler coordinate is part of the accepted packet.

The classifier of Definition 473.2, together with Definition 473.10, restricts edge reads to source classifier comparison, target landing, transported classifier rows, and displayed continuation routes.

Apply Definition 473.21 and Definition 473.22. Identity and composite routes keep the same finite dependency surface, so no host compiler, optimizer, ambient equality, or unrecorded target behavior is exported. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier finite route nonescape`

The endpoint route consumes the finite morphism packet of chapter 620. KernelMorphismUp supplies source-target graph admission and classifier-lift rows, while CertificateCompiler$^{\uparrow}$ adds target landing, endpoint route, and composition ledgers.

Theorem 473.27 (CertificateCompiler target endpoint route)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet. Every route by which K reaches the target certificate T is exhausted by the source* NameCert *row S, the displayed compiler graph G, the target landing row E, the classifier transport row H, the continuation row C, the* Pkg *provenance row P, and the local* NameCert *row N. In particular, target arrival is a finite* BHist *read through S, G, E, H, C, P, N, not a read of host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. Project the carrier of Definition 473.1. The accepted packet exposes exactly the source and target naming rows, graph, landing, continuation, componentwise hsame transport, package provenance, and local naming row.

Endpoint availability is supplied by Definition 473.3, so the target certificate is already one of the displayed rows. Edge-classifier exactness from Definition 473.10 then forces each graph edge to land through E and to transport classifier data only through H and the paired Cont route in C.

Finally apply the public consumer boundary of Definition 473.24. A downstream read can consume only the displayed graph, landing, classifier transport, continuation, package, and local naming rows. Thus every target endpoint route is the stated finite BHist route through S, G, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint route`

Theorem 473.28 (CertificateCompiler identity composite public boundary)**.** *For accepted* CertificateCompiler$^{\uparrow}$ *packets, identity rows, binary composites, and triple composites remain inside the same public row surface S, T, G, E, C, H, P, N. The identity packet exposes only the diagonal rows of its local* NameCert *input, a binary composite exposes only the two displayed compiler packets and their shared middle landing, and a triple composite exposes only the three displayed graph edges, the two middle landing rows, the ordered* hsame *transports, the composed* Cont *route, the* Pkg *provenance, and the local* NameCert *rows.*

Proof. For identities, apply Definition 473.9. The diagonal packet reads the carrier, classifier, hsame, Cont, Pkg, and NameCert rows already displayed by the local certificate.

For binary composites, use Definition 473.8 together with Definition 473.12. The composite graph is formed from displayed graph edges, and its landing, classifier, continuation, package, and local naming rows are projected from the two input packets and their middle NameCert row.

For triple composites, apply Definition 473.13. The two associations expose the same three graph edges, two middle landing matches, ordered classifier transports, and continuation endpoint.

The left and right unit laws, Definition 473.14 and Definition 473.15, identify identity composites with the original packet surface. Combining these rows gives the same public boundary S, T, G, E, C, H, P, N for identity, binary, and triple compiler packets. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier identity composite public boundary`

Theorem 473.29 (CertificateCompiler endpoint-route determinacy envelope)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted $\mathsf{CertificateCompiler}^{\uparrow}$ *packet. Any accepted target endpoint route for* K*, and any identity, binary composite, or triple composite route built from accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packets, is determined by the same displayed finite surface* S, T, G, E, C, H, P, N*. There is no second endpoint route through host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. First apply the target endpoint route of Definition 473.55. It places endpoint arrival inside the displayed source naming row, compiler graph, target landing row, classifier transport, continuation row, package provenance, and local naming row of the accepted carrier.

Next apply the identity composite public boundary of Definition 473.56. Identity rows, binary composites, and triple composites stay on the same public row surface S, T, G, E, C, H, P, N; their middle landings, graph edges, transports, continuation rows, package provenance, and local names are projected from displayed compiler packets.

Finally return to the carrier definition. The accepted packet has no coordinate for host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior. Endpoint arrival and the composite public boundary therefore share the same displayed rows, so the finite surface S, T, G, E, C, H, P, N determines the available endpoint and composite routes. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint route determinacy envelope`

Theorem 473.30 (CertificateCompiler endpoint determinacy)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet. If two accepted target endpoint routes for* K *agree on the source row* S*, the compiler graph* G*, the landing row* E*, the classifier transport* H*, the continuation row* C*, the provenance row* P*, and the local name* N*, then they determine the same target certificate row* T*. No host compiler semantics, optimiser equality, ambient program equality, or unrecorded target behavior can create a second endpoint.*

Proof. First apply the target endpoint route theorem, Definition 473.55. It exhausts every route to T by the displayed source, graph, landing, classifier transport, continuation, package, and local naming rows.

Endpoint certificate availability, Definition 473.3, supplies the target certificate as part of that displayed packet, and edge-classifier exactness, Definition 473.10, keeps graph edges tied to the same landing and classifier rows. Thus two accepted reads agreeing on S, G, E, H, C, P, N have no further endpoint coordinate to vary.

Now use the public consumer boundary, Definition 473.24, together with Definition 473.56. Public consumers, identities, binary composites, and triple composites remain on the same finite row surface. Since the packet contains no host compiler semantics, optimiser equality, ambient program equality, or unrecorded target behavior, the agreed displayed rows determine the same target certificate row T. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint determinacy`

Theorem 473.31 (CertificateCompiler downstream stage unblock package)**.** *The target endpoint route of Definition 473.55 and the identity/composite boundary of Definition 473.56 supply the public package consumed by* CertificatePipelineStageBoundary$^{\uparrow}$, KernelMorphismUp, *and* DecompilerRightInverse$^{\uparrow}$. *The package consists only of finite* BHist *rows, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *rows over* $K = (S, T, G, E, C, H, P, N)$*; it does not export host compiler semantics, optimiser correctness, decompiler correctness, ambient program equality, or target behavior absent from the packet.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier downstream stage unblock package`

Proof. First build the endpoint part of the package. By Definition 473.55, every accepted target route reads the source naming row, graph row, landing row, classifier transport row, continuation row, package provenance, and local naming row.

Next build the compositional part. By Definition 473.56, identities, binary composites, and triple composites keep the same public row surface and do not add a host compiler coordinate.

Now combine these with Definition 473.23 and Definition 473.26. The source and target endpoints are exhausted by the displayed packet rows, and every finite route stays inside BHist, hsame, Cont, Pkg, and NameCert coordinates.

The consumers named in the statement require exactly this public route surface: pipeline stages read adjacent compiler boundaries, kernel morphisms read source-to-target graph preservation, and decompiler right-inverse packets read the displayed compiler/decompiler interface. Since all three reads factor through the same finite package, none receives a host compiler semantics or optimiser-correctness theorem. □

Theorem 473.32 (CertificateCompiler KernelMorphism route scope)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Any downstream* KernelMorphismUp *consumer reads the compiler graph* G *only through the endpoint route* S, G, E, H, C, P, N *and the source-target graph admission and classifier-lift rows of chapter 620. It cannot import host compiler semantics, optimiser correctness, decompiler correctness, or target behavior absent from the finite packet.*

Proof. First use the target endpoint route of Definition 473.55. It identifies the graph-facing route to the target certificate with the displayed source naming row, compiler graph, target landing row, classifier transport, continuation row, package provenance, and local naming row.

Next apply the identity and composite public boundary of Definition 473.56. Identities, binary composites, and triple composites stay on the same finite endpoint surface, so a downstream morphism consumer receives graph preservation through displayed compiler rows rather than through host program semantics.

Finally combine the downstream stage package of Definition 473.59 with the KernelMorphismUp source-target scope named in chapter 620. The consumer-facing route is exactly the finite BHist handoff through S, G, E, H, C, P, N together with the KernelMorphism graph-admission and classifier-lift rows, leaving optimiser correctness, decompiler correctness, and unrecorded target behavior outside the packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier kernelmorphism route scope`

Theorem 473.33 (CertificateCompiler KernelMorphism decompiler boundary)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Any route from* K *into a* KernelMorphismUp *consumer and then into a* DecompilerRightInverse$^{\uparrow}$ *consumer factors through the displayed finite rows*

S, G, E, H, C, P, N, the KernelMorphismUp *graph-admission and classifier-lift rows, and the* DecompilerRightInverse$^{\uparrow}$ *inverse-boundary rows of Definition 614.3. The route exports no host compiler semantics, optimiser correctness, ambient program equality, host decompiler function, total decompiler, or unrecorded target behavior.*

Proof. First apply the target endpoint route of Definition 473.55. It places every accepted compiler target read on the displayed finite rows S, G, E, H, C, P, N.

Next apply the KernelMorphism route scope of Definition 473.60. The intermediate morphism consumer can add only the graph-admission and classifier-lift rows licensed by KernelMorphismUp, so the compiler leg remains a finite BHist handoff rather than a host compiler semantics.

Finally apply the decompiler right-inverse non-escape boundary of Definition 614.3. A downstream inverse consumer may read only the displayed compiler packet, decompiler graph, target optimisation row, right-inverse ledger, transport, continuation, provenance, and local naming rows. Combining the three non-escape clauses leaves no route to optimiser correctness, ambient program equality, host decompiler behavior, or target behavior absent from the packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier kernelmorphism decompiler boundary`

Theorem 473.34 (CertificateCompiler target endpoint determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Once the source row S, a displayed graph edge in G, its landing row E, the classifier transport row H, and the continuation row C are fixed, the target endpoint route to T is determined as the finite route through S, G, E, H, C, P, N. No second endpoint route can be obtained from host compiler semantics, optimiser correctness, ambient program equality, or an unrecorded target behavior.*

Proof. Start from Definition 473.55. It identifies every target arrival with the displayed source, graph, landing, classifier, continuation, package, and local naming rows.

Use Definition 473.10. With the source row, graph edge, landing row, classifier transport, and continuation row fixed, the edge has no further classifier-facing or endpoint-facing coordinate to vary.

Finally apply Definition 473.26. The carrier exports no external compiler semantics or target behavior, so the fixed displayed rows determine the endpoint route. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint determinacy`

Theorem 473.35 (CertificateCompiler endpoint-route pullback determinacy)**.** *Let a downstream route pass through* CertificateCompiler$^{\uparrow}$ *into* KernelMorphismUp *and then into* DecompilerRightInverse$^{\uparrow}$*. If two such routes have the same displayed source row, graph edge, landing row, classifier transport row, and* Cont *row, then their endpoint pullbacks are the same finite route through S, G, E, H, C, P, N. No host compiler semantics, optimiser correctness, ambient program equality, host decompiler function, or unrecorded target behavior can distinguish the pullbacks.*

Proof. First apply target endpoint determinacy, Definition 473.62. Agreement of the displayed source row, graph edge, landing row, classifier transport row, and continuation row fixes the compiler endpoint route through S, G, E, H, C, P, N.

Next use the KernelMorphism route scope, Definition 473.60. The morphism consumer can add only the graph-admission and classifier-lift rows licensed by KernelMorphismUp, so it cannot separate two already fixed compiler endpoint routes.

Finally apply the KernelMorphism decompiler boundary, Definition 473.61. The decompiler-facing leg remains on the displayed compiler packet and inverse-boundary rows, leaving no coordinate for host compiler semantics, optimiser correctness, ambient equality, host decompiler functions, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint route pullback determinacy`

Theorem 473.36 (CertificateCompiler route composition exactness)**.** *For accepted* CertificateCompiler$^{\uparrow}$ *packets, the identity route, every binary composite route, and every triple composite route use only the displayed graph rows, landing rows, classifier transport rows, and* Cont *rows, together with the visible* Pkg *and local* NameCert *provenance. These routes introduce no hidden program equality, optimiser correctness theorem, ambient compiler semantics, or target behavior outside the finite packet.*

Proof. For the identity and binary cases, apply Definition 473.56. The identity packet reads the diagonal local naming rows, and the binary composite reads the two displayed compiler packets and their shared middle landing.

For the triple case, use the same public boundary together with Definition 473.10. The three graph edges, two middle landing rows, ordered classifier transports, and composed Cont route are exactly the rows visible to the consumer.

The finite non-escape theorem Definition 473.26 excludes any additional semantic coordinate. Hence all identity, binary, and triple composition routes are exact over the displayed rows. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier route composition exactness`

Theorem 473.37 (CertificateCompiler ledger cut elimination)**.** *Let a downstream consumer read the public package supplied by Definition 473.59. An intermediate package row may be cut from that consumer route only when the same displayed endpoint rows and the same identity, binary, or triple composite rows remain preserved in the resulting route. A cut that changes the endpoint route, changes the composite route, or replaces either route by host compiler semantics, decompiler correctness, ambient program equality, or unrecorded target behavior is not a valid* CertificateCompiler$^{\uparrow}$ *consumer read.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier ledger cut elimination`

Proof. First read the downstream package through Definition 473.59. The consumer-visible packet consists only of finite BHist rows, displayed hsame transport, Cont routes, Pkg provenance, and local NameCert rows.

Endpoint preservation is forced by Definition 473.55. If a cut removes an intermediate package row but keeps the same source, graph, landing, classifier, continuation, provenance, and local naming rows, the target route is still the displayed finite route.

Composite preservation is forced by Definition 473.56 and Definition 473.64. Identity, binary, and triple composites remain valid exactly when their displayed graph, landing, classifier, and continuation rows are unchanged.

Finally use Definition 473.26. Any cut that replaces a displayed row by host semantics, decompiler correctness, ambient equality, or unrecorded target behavior leaves the finite packet surface and is therefore unavailable to downstream consumers. □

Theorem 473.38 (CertificateCompiler endpoint decompiler handoff determinacy)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* CertificateCompiler$^{\uparrow}$ *endpoint route. Once the displayed rows* S, G, E, H, C, P, N *and the* KernelMorphismUp *graph-admission and classifier-lift rows are fixed, any* DecompilerRightInverse$^{\uparrow}$ *consumer can read only the inverse-boundary rows already cited by Definition 473.61. The handoff cannot introduce host compiler semantics, optimiser correctness, ambient program equality, host decompiler functions, or unrecorded target behavior.*

Proof. First use target endpoint determinacy, Definition 473.62. With the source row, graph edge, landing row, classifier transport, continuation row, provenance, and local naming row fixed, the endpoint route is the displayed finite route through S, G, E, H, C, P, N.

Next apply route composition exactness and ledger cut elimination, Definition 473.64 and Definition 473.65. Identity, binary, and triple composites may be replayed only over the same displayed endpoint rows, and no intermediate cut can replace them by host semantics or unrecorded behavior.

Now pass through the KernelMorphism route scope, Definition 473.60. The morphism leg contributes only the graph-admission and classifier-lift rows licensed by KernelMorphismUp.

Finally apply the KernelMorphism decompiler boundary, Definition 473.61, together with the decompiler right-inverse non-escape boundary of Definition 614.3. A downstream inverse consumer therefore receives only the displayed compiler packet, the fixed KernelMorphism rows, and the inverse-boundary rows, with no coordinate for host compiler semantics, optimiser correctness, ambient equality, host decompiler functions, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint decompiler handoff determinacy`

Theorem 473.39 (CertificateCompiler endpoint-composite determinacy)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Once the displayed source row, graph edge, landing row, classifier transport, continuation row, and identity, binary, and triple composite rows are fixed, every downstream* KernelMorphismUp *or* DecompilerRightInverse$^{\uparrow}$ *consumer route is determined by the finite packet* S, G, E, H, C, P, N *together with the relevant composite rows. The determined route exports no host compiler semantics, optimiser correctness theorem, decompiler correctness theorem, ambient program equality, or target behavior outside those finite rows.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint composite determinacy`

Proof. Start with the endpoint route and public composite boundary. The target endpoint route of Definition 473.55 fixes the source, graph, landing, classifier transport, continuation, provenance, and local naming rows, while Definition 473.56 fixes identity, binary, and triple composites on the same displayed surface.

Next expose the downstream packet. Apply the downstream stage unblock package, Definition 473.59, together with KernelMorphism route scope, Definition 473.60. A morphism consumer can read only the finite compiler route and the corresponding graph-admission and classifier-lift rows.

Now attach the decompiler boundary. The KernelMorphism decompiler boundary, Definition 473.61, keeps any route into DecompilerRightInverse$^{\uparrow}$ on the same finite compiler packet before the inverse-boundary rows are read.

Finally use target endpoint determinacy, route composition exactness, and ledger cut elimination, Definition 473.62, Definition 473.64, and Definition 473.65. These identify all endpoint and composite reads after the listed rows are fixed, so every downstream morphism or decompiler route

is determined by S, G, E, H, C, P, N plus the relevant composite rows, with no remaining semantic coordinate available. □

Theorem 473.40 (CertificateCompiler endpoint KernelMorphism roundtrip)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted finite* CertificateCompiler$^{\uparrow}$ *endpoint route. If the source and target endpoint reads agree under Definition 473.62, then the route factors through the* KernelMorphismUp *graph-admission and classifier-lift scope and returns through the* DecompilerRightInverse$^{\uparrow}$ *boundary on the same displayed compiler rows.*

The roundtrip uses only S, G, E, H, C, P, N, the relevant identity, binary, or triple composite rows, the KernelMorphismUp *graph-admission and classifier-lift rows, and the* DecompilerRightInverse$^{\uparrow}$ *inverse-boundary rows. It adds no hidden host compiler semantics, optimiser correctness theorem, decompiler correctness theorem, ambient program equality, total decompiler, or unrecorded target behavior.*

Proof. First apply target endpoint determinacy, Definition 473.62. Agreement of the source and target endpoint reads fixes the displayed finite endpoint route through S, G, E, H, C, P, N.

Next pass through the morphism surface using Definition 473.60. A downstream KernelMorphismUp consumer can read only the graph-admission and classifier-lift rows attached to the same compiler packet, so the route has no host compiler coordinate at the morphism stage.

Now return through the decompiler side by Definition 473.61. The decompiler-facing consumer remains on the displayed compiler packet together with the inverse-boundary rows, rather than importing host decompiler behavior or total decompiler data.

Finally compose the route and remove intermediate package cuts. The composition exactness theorem, Definition 473.64, keeps identity, binary, and triple composites on the displayed rows, while ledger cut elimination, Definition 473.65, permits only cuts preserving those endpoint and composite rows. Thus the endpoint, KernelMorphism, and decompiler legs form a finite roundtrip with no hidden host compiler semantics. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint kernelmorphism roundtrip`

Corollary 473.41 (CertificateCompiler target endpoint consumer determinacy)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet. Fix the consumer-visible rows cited by Definition 473.57: the source row S, target row T, compiler graph G, target landing row E, continuation row C, classifier transport row H, package provenance row P, local naming row N, and, when a composite is requested, the displayed identity, binary, or triple-composite ledger rows over the same surface. Then any accepted consumer-facing use of K must select the same finite endpoint route, the same composite normal form, or the same classifier-exactness decision named by the* CertificateCompiler$^{\uparrow}$ *chapter. It cannot select an alternate hidden coordinate, a second source classifier row, host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. First apply Definition 473.55. It exhausts every target arrival by the finite BHist route through the displayed source naming row, graph row, target landing row, classifier transport row, continuation row, package provenance row, and local naming row. Thus a consumer that asks for the target endpoint has no endpoint coordinate beyond S, G, E, H, C, P, N.

Next apply Definition 473.57. It says that accepted endpoint routes and accepted identity, binary, and triple-composite routes are all determined by the same displayed finite surface

S, T, G, E, C, H, P, N. Hence fixing the cited local rows also fixes the composite normal form available to a consumer: the identity normal form is the diagonal row, a binary normal form is the displayed two-packet composite through the shared middle landing, and a triple normal form is the displayed three-edge ledger with its two middle landing rows.

It remains to rule out duplicate classifier sources and alternate exactness decisions. By Definition 473.10, an accepted classifier decision reads the source classifier row in S, lands through E, transports through the componentwise hsame row H, and follows the paired Cont routes in C. With S, E, H, C fixed, there is no second classifier-facing source from which a different decision could be accepted.

Finally use Definition 473.26. The accepted carrier exports no host compiler semantics, optimiser correctness theorem, ambient program equality, or unrecorded target behavior. Any consumer-facing use that agrees on the displayed rows therefore factors through the same finite route surface and must return the same endpoint route, composite normal form, or classifier-exactness decision named above. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint consumer determinacy`

Theorem 473.42 (CertificateCompiler root landing obligation)**.** *Every root consumer of an accepted* CertificateCompiler$^{\uparrow}$ *packet*

$$K = (S, T, G, E, C, H, P, N)$$

must first read the source NameCert *endpoint row S, the target* NameCert *endpoint row T, the displayed compiler graph G, and the edge-landing row E. The consumer may treat a displayed graph edge as landed only after S supplies the source carrier and classifier row, T supplies the target carrier and classifier row, G supplies the finite edge, and E records that the edge starts in the carrier named by S and lands in the carrier named by T.*

No root consumer can obtain a landing witness from host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, or an unrecorded target behavior outside the rows S, T, G, E.

Proof. Unfold the carrier in Definition 473.1. Its first four rows are exactly the source endpoint certificate S, the target endpoint certificate T, the displayed compiler graph G, and the edge-landing row E.

Use the classifier definition in Definition 473.2. Even a paired-edge comparison begins by reading the source classifier row in S and the target landing row in T, so classification cannot precede endpoint availability.

Apply Definition 473.7. Its carrier landing obligation is the finite row assertion that every edge of G reads a source row admitted by S and lands in a target row admitted by T through E.

The carrier has no coordinate for host compiler equality or ambient program equivalence. Therefore a root consumer that consumes landing data is forced through the displayed S, T, G, E rows and cannot source a landing witness elsewhere. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing obligation`

Theorem 473.43 (CertificateCompiler root classifier transport obligation)**.** *For every accepted* CertificateCompiler$^{\uparrow}$ *packet*

$$K = (S, T, G, E, C, H, P, N),$$

paired-edge classifier transport is exhausted by the source and target endpoint rows S, T, the componentwise hsame *transport row H, and the continuation row C. A root consumer may compare two displayed graph edges only by sharing the source classifier row in S, landing through T, moving the target classifier row through H, and following the paired* Cont *routes recorded in C.*

The transport obligation does not admit host compiler equality, source program equivalence, target program equivalence, or an external semantic relation as a classifier witness.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root classifier transport obligation`

Proof. Unfold Definition 473.1. The transport-relevant rows available to a root consumer are the endpoint certificates S, T, the componentwise transport row H, and the continuation compatibility row C, with G and E only displaying the edges whose classifiers are compared.

Now unfold Definition 473.2. It defines compatibility for paired graph edges by a common source classifier row, a target landing, componentwise hsame transport, and paired continuation routes. These are precisely the rows S, T, H, C named in the statement.

Apply the classifier-transport clause of Definition 473.7. That clause permits transport from the source classifier row to the target classifier row only by the displayed finite-row data already carried by the packet.

Since neither the carrier nor the classifier contains host compiler equality or a program-equivalence coordinate, a root consumer cannot certify paired edges by such an external equality. The finite row surface S, T, H, C exhausts classifier transport. □

Theorem 473.44 (CertificateCompiler root continuation route obligation)**.** *Every downstream root consumer of an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet, including* $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$, KernelMorphismUp*, and* $\mathsf{DecompilerRightInverse}^{\uparrow}$*, reads continuation compatibility only through the displayed* Cont *row C, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *row N together with the endpoint* NameCert *rows S, T. The consumer route factors through the recorded source and target continuation pair, the named package surface, and the local certificate row; it does not factor through an unstated compiler semantics or decompiler correctness theorem.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root continuation route obligation`

Proof. Unfold the carrier in Definition 473.1. The only continuation coordinate is C, and the rows that publish it to downstream consumers are the provenance row P and the local naming row N over the endpoint certificates S, T.

Use Definition 473.2. Its paired-edge comparison does not stop at landing; it also reads the source and target Cont routes recorded in C, so any classifier-compatible downstream route has already passed through the displayed continuation row.

Apply Definition 473.7. The continuation-compatibility clause states that each source route used by a graph edge is paired in C with the target route followed after compiling, while the package and local naming rows can only repack these displayed obligations.

The three named root consumers each read a different downstream surface: pipeline-stage boundaries read adjacent compiler routes, kernel morphisms read source-to-target preservation, and decompiler right-inverse packets read the compiler/decompiler interface. In all three cases the available route data are the finite rows C, P, N, S, T and the local NameCert obligations, so no unstated compiler semantics or decompiler correctness theorem can enter the continuation route. □

Theorem 473.45 (CertificateCompiler root landing bridge determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet. Every consumer-facing bridge use of root landing data for K, including a displayed edge route, classifier transport, endpoint readback, continuation readback, package handoff, or local naming handoff, factors through the local row surface*

$$S, T, G, E, C, H, P, N.$$

More precisely, landing is determined by S, T, G, E, paired-edge classifier transport is determined by S, T, H, C, continuation route readback is determined by C over the endpoint rows S, T, and the handoff rows P, N can only repack those displayed obligations. No host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, external semantic relation, or unrecorded target behavior supplies an additional bridge coordinate.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing bridge determinacy`

Proof. Start with root landing. By Definition 473.75, a root consumer may treat a displayed graph edge as landed only after reading the endpoint rows S, T, the displayed graph row G, and the edge-landing row E. Thus every bridge use whose first demand is endpoint readback, graph readback, or target landing is already forced through S, T, G, E.

Next read classifier transport. By Definition 473.76, paired-edge comparison is exhausted by the source and target endpoint rows S, T, the componentwise hsame row H, and the continuation row C. Hence a consumer cannot turn a landing bridge into an independent classifier witness: the only accepted transport reads the same endpoints, moves the target classifier through H, and follows the paired Cont routes in C.

Now apply Definition 473.77. A downstream route consumer reads continuation compatibility through C, package provenance through P, and local naming through N together with the endpoint certificates S, T. Therefore route readback, continuation readback, package handoff, and local naming handoff do not create a second bridge surface; they publish the same finite landing and transport rows to downstream consumers.

Combining the three obligations gives determinacy. The landing rows S, T, G, E decide which displayed edge has landed, the transport rows S, T, H, C decide which paired-edge classifier comparison is available, and the rows C, P, N decide how that route is handed to later consumers. Each listed row has a fixed role in the packet, while the carrier and classifier supply no coordinate for host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, an external semantic relation, or unrecorded target behavior. Any attempted bridge coordinate outside S, T, G, E, C, H, P, N therefore has no row to read, and every permitted consumer-facing use factors through the cited local rows. □

Corollary 473.46 (CertificateCompiler root landing consumer determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet, and fix the consumer-visible rows cited by Definition 473.78: the source endpoint row S, the target endpoint row T, the displayed compiler graph G, the edge-landing row E, the continuation row C, the componentwise* hsame *transport row H, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *row N. Then any accepted consumer-facing use of root landing data for K must pick the same finite displayed edge route, the same continuation readback route, the same package or local naming handoff, and the same classifier-exactness decision determined by those rows. It cannot pick an alternate hidden coordinate, a duplicate source classifier row, a second endpoint readback source, host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, an external semantic relation, or unrecorded target behavior.*

Proof. By Definition 473.75, root landing can be consumed only after the consumer has read S, T, G, E. Thus the landing part of any accepted use is already pinned to the source endpoint certificate, target endpoint certificate, displayed graph edge, and edge-landing row named in the packet. There is no carrier coordinate from which the consumer could obtain another landing witness, endpoint readback, or source classifier row.

Apply Definition 473.78. It lists the consumer-facing bridge uses of the same root landing data: displayed edge route, classifier transport, endpoint readback, continuation readback, package handoff, and local naming handoff. The theorem states that each such use factors through the finite row surface

$$S, T, G, E, C, H, P, N.$$

It also separates the roles of the rows: S, T, G, E determine landing, S, T, H, C determine paired-edge classifier transport, C over S, T determines continuation readback, and P, N only repack those obligations for package and local naming handoff.

The remaining alternatives are therefore ruled out by absence of a row to read. A claimed alternate finite route would have to alter the displayed graph, edge landing, continuation row, or handoff row, but those are fixed as G, E, C, P, N. A claimed different normal form would have to use an identity, composite, or readback surface outside the displayed packet, but Definition 473.78 permits only the listed bridge surface for consumer-facing use. A claimed different classifier-exactness decision would have to replace the source classifier row in S, the target classifier row in T, the componentwise transport row H, or the paired Cont row C, but the classifier transport part of the bridge determinacy theorem exhausts exactly S, T, H, C.

Consequently, once the cited local rows are fixed, every accepted consumer-facing use lands on the same displayed route, readback route, handoff, or exactness decision named by the CertificateCompiler$^{\uparrow}$ chapter. A hidden coordinate, duplicate classifier source, or external semantic witness is not an accepted consumer source for root landing data. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing consumer determinacy`

Theorem 473.47 (CertificateCompiler bridge schema handoff)**.** *Every public* CertificateCompiler$^{\uparrow}$ *packet hands a schema bridge only the finite compiler graph, target landing rows, classifier-transport rows, continuation rows, package provenance, and source, target, and local* NameCert *rows. The handoff is the displayed packet surface S, T, G, E, C, H, P, N together with its identity, composite, and triple-composite ledgers; it does not include host compiler semantics, optimizer correctness, decompiler behavior, ambient program equality, or unrecorded target behavior.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge schema handoff`

Proof. Start with the carrier of Definition 473.1. It exposes source and target certificates, graph edges, landing rows, continuation rows, componentwise transports, provenance, and a local naming row.

Use the classifier and the NameCert obligations. Definition 473.2 and Definition 473.7 identify the bridge-readable packet with the finite graph, landing, classifier transport, continuation, package, and naming rows.

Now apply edge-classifier exactness, the identity ledger, composition stability, and triple-composite associativity. These theorems show that every schema-facing edge, identity, binary composite, and triple composite is replayed through the same finite rows, so the bridge handoff has no host compiler coordinate. □

Theorem 473.48 (CertificateCompiler bridge-row non-escape)**.** *The bridge-facing rows of a public* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet export no optimizer, decompiler, program equality, ambient categorical equality, or unrecorded target behavior. Any schema bridge that consumes the packet must read the source certificate, target certificate, graph edge, landing, route, transport, provenance, and local naming rows exactly as displayed.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge rows nonescape`

Proof. Project the public packet as the finite rows S, T, G, E, C, H, P, N. These rows are the only bridge-facing coordinates of the carrier.

Apply Definition 473.10 and Definition 473.26. They force edge reads through graph, landing, classifier transport, continuation, provenance, and naming rows and exclude host compiler semantics from the packet.

The identity ledger, composition stability, and triple-composite associativity preserve the same row set under bridge replay. Since no displayed row is an optimizer, decompiler, program-equality witness, ambient categorical equality, or target-behavior oracle, none can be exported to the bridge consumer. □

Theorem 473.49 (CertificateCompiler bridge consumer exhaustion)**.** *Every schema-bridge consumer of a public* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet factors through the already displayed source certificate, target certificate, graph edge, landing row, route row, classifier-transport row, provenance row, and local naming row. The consumer can compose, reassociate, and replay certificate-compiler edges only by the identity, binary-composite, and triple-composite ledgers already present in the chapter.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge consumer exhaustion`

Proof. Begin with the public consumer boundary Definition 473.24. It already restricts downstream reads to graph landing, classifier transport, continuation, provenance, and local naming over the source and target certificates.

Add the bridge handoff of Definition 473.70. The bridge receives exactly the same finite graph and certificate rows, with no host compiler semantics.

Finally apply the identity ledger, composition stability, and triple-composite associativity. These results exhaust the bridge consumer's available operations on the packet: identity replay, binary composition, and triple reassociation all factor through the displayed rows and cannot introduce another compiler surface. □

Theorem 473.50 (CertificateCompiler bridge associativity split index)**.** *The bridge handoff of Definition 473.70, the bridge-row non-escape boundary of Definition 473.71, and the bridge consumer exhaustion row of Definition 473.72 all read the same identity, binary-composite, and triple-composite ledger for* $\mathsf{CertificateCompiler}^{\uparrow}$ *packets. A schema bridge therefore consumes one finite ledger surface rather than three unrelated compiler interfaces.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge associativity split index`

Proof. First read the identity ledger. Definition 473.9 gives the diagonal compiler packet over the same displayed source, target, graph, landing, classifier transport, continuation, provenance, and naming rows consumed by the bridge handoff.

Next read binary composition. Definition 473.8 and Definition 473.12 keep composite replay on the same public row surface and add no host compiler semantics.

Finally read triple composites. Definition 473.13 and the bridge consumer exhaustion theorem use the same triple of graph edges and middle landing matches, while bridge-row non-escape blocks optimizer, decompiler, program-equality, ambient categorical, and unrecorded target-behavior coordinates. □

Corollary 473.51 (CertificateCompiler bridge schema handoff strict obstruction)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be a public* CertificateCompiler$^{\uparrow}$ *packet read by a schema bridge. If a displayed failure, refusal, separation, or countermodel row is cited against a named bridge-handoff converse, reuse, or readback route for K, then that row is a strict local obstruction to that named route only. It blocks exactly the request that tries to read, omit, replace, detach, or bypass one of the displayed source certificate, target certificate, graph edge, landing, route, classifier-transport, provenance, or local naming rows. It cannot be reinterpreted as a global impossibility theorem, a host theorem about compilers, hidden external validation, optimizer correctness, decompiler correctness, ambient program equality, ambient categorical equality, or unrecorded target behavior.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge schema handoff strict obstruction`

Proof. Read the schema bridge handoff first. By Definition 473.70, the bridge receives only the finite compiler graph, target landing rows, classifier-transport rows, continuation rows, package provenance, and source, target, and local NameCert rows of the packet. Its full bridge surface is the displayed tuple S, T, G, E, C, H, P, N together with the identity, binary-composite, and triple-composite ledgers. The same theorem excludes host compiler semantics, optimizer correctness, decompiler behavior, ambient program equality, and unrecorded target behavior from the handoff.

Now apply bridge-row non-escape, Definition 473.71. Any schema bridge that consumes K must read the source certificate, target certificate, graph edge, landing, route, transport, provenance, and local naming rows exactly as displayed. Thus a consumer-facing use of a displayed obstruction row has no coordinate outside the listed bridge rows. Source and target admission are read through S, T; graph and landing evidence through G, E; route and replay through C; classifier transport through H; package provenance through P; and the local certificate name through N.

Use Definition 473.72 and Definition 473.73 to exhaust the remaining bridge operations. A bridge consumer may compose, reassociate, or replay only by the displayed identity, binary-composite, and triple-composite ledgers, and those ledgers read the same finite surface rather than separate compiler interfaces. Hence an obstruction row attached to a named bridge-handoff converse, reuse, or readback route can block only the local request whose reading position asks for an unlisted coordinate or bypasses one of the displayed rows.

Any stronger interpretation would require data that the two bridge theorems and the consumer ledger do not export. A global impossibility theorem would require a theorem surface beyond the finite packet; a host theorem about compilers would require host compiler semantics or host program equality; hidden external validation would require an external certificate channel beyond P and N; optimizer or decompiler correctness would require optimizer or decompiler coordinates; and ambient categorical equality or unrecorded target behavior would require exactly the off-packet data ruled out by bridge-row non-escape. Since every permitted bridge-facing use factors through S, T, G, E, C, H, P, N and the displayed ledgers, the obstruction remains strict and local to the named bridge-handoff route. □

Theorem 473.52 (CertificateCompiler mature consumer inventory)**.** *Every mature consumer read of an accepted* CertificateCompiler$^{\uparrow}$ *packet factors through endpoint* NameCert *rows, displayed graph edges, edge landing, classifier transport, continuation compatibility,* Pkg *provenance, and local*

NameCert *rows. The inventory is exhausted by the identity ledger of Definition 473.9, the binary composition stability of Definition 473.8, and the triple composite associativity of Definition 473.13; no additional compiler semantics, optimizer witness, decompiler, program equality, or unrecorded target behavior is available to the consumer.*

Proof. First read the endpoint rows. The source and target NameCert entries supply the only admission points for graph edges, and the carrier records their edge landing, classifier transport, continuation compatibility, provenance, and local naming rows.

Next read identity and binary routes. Definition 473.9 supplies the diagonal endpoint route, while Definition 473.8 composes only displayed graph edges through their middle landing and classifier-transport rows.

Finally read triple routes through Definition 473.13. The two associations preserve the same graph-edge inventory and the same endpoint landing data, so every mature consumer remains inside the finite packet surface listed in the statement. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier mature consumer inventory`

Theorem 473.53 (CertificateCompiler route-category boundary)**.** *Identity and composite* CertificateCompiler$^{\uparrow}$ *packets form only a finite route-category boundary over displayed graph composition. Their identities, binary composites, and triple composites are the packet routes described by Definition 473.9, Definition 473.8, and Definition 473.13; they do not constitute host compiler semantics, ambient categorical laws, optimizer correctness, or a theory of program equivalence.*

Proof. Use the identity ledger for objects. It gives a diagonal packet over a displayed endpoint certificate and records only graph, landing, classifier, continuation, provenance, and local naming rows.

Use composition stability for arrows. Binary composition is graph composition over displayed middle landing data, and its classifier and continuation clauses are read from the same packet rows.

Use triple composite associativity only as a boundary statement. It aligns the two finite replay routes over three displayed graph edges, but it adds no semantic interpretation of compilers, programs, optimizers, or ambient categories. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier route category boundary`

Theorem 473.54 (CertificateCompiler formal target surface)**.** *The Lean-facing formal target for* CertificateCompiler$^{\uparrow}$ *is the finite packet surface S, T, G, E, C, H, P, N together with endpoint certificate availability, displayed edge witness totality, the consumer replay obligation triad, composition stability, the identity ledger, edge-classifier exactness, triple composite associativity, the public consumer boundary, and finite-route non-escape. It excludes host compiler semantics, optimizer correctness, ambient program equality, decompiler semantics, and unrecorded target behavior.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier formal target surface`

Proof. Use Definition 473.1, Definition 473.3, and Definition 473.4 to expose S, T, G, E, C, H, P, N, endpoint certificate availability, and displayed edge witness totality.

Apply Definition 473.6, Definition 473.10, Definition 473.8, Definition 473.9, and Definition 473.13. These results keep all formal obligations inside displayed edge, landing, classifier, continuation, transport, provenance, and naming rows.

Finally apply Definition 473.24 and Definition 473.26. They state that consumers receive only the finite route packet, so host compiler semantics, optimizer correctness, ambient program equality, decompiler semantics, and unrecorded target behavior are outside the formal target. □

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateCompiler}^{\uparrow}$ packet is generated from source and target NameCert rows, a finite compiler graph, edge landing, continuation compatibility, hsame transports, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public compiler-certificate surface binds the finite graph, landing, classifier, continuation, identity, composition, export, triple-composite, BHist, hsame, Cont, Pkg, and NameCert rows already displayed for certificate compiler packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host compiler semantics, program equivalence, optimiser correctness, decompilation, categorical laws beyond the displayed finite packets, or a checked Lean certificate.

Upgrade path: No further theory-axis upgrade – chapter at top of closure axis; formal upgrade requires Lean verification of the same finite compiler-certificate graph.

The endpoint route consumes the finite morphism packet of chapter 620. KernelMorphismUp supplies source-target graph admission and classifier-lift rows, while $\mathsf{CertificateCompiler}^{\uparrow}$ adds target landing, endpoint route, and composition ledgers.

Theorem 473.55 (CertificateCompiler target endpoint route)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted $\mathsf{CertificateCompiler}^{\uparrow}$ *packet. Every route by which K reaches the target certificate T is exhausted by the source* NameCert *row S, the displayed compiler graph G, the target landing row E, the classifier transport row H, the continuation row C, the* Pkg *provenance row P, and the local* NameCert *row N. In particular, target arrival is a finite* BHist *read through S, G, E, H, C, P, N, not a read of host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. Project the carrier of Definition 473.1. The accepted packet exposes exactly the source and target naming rows, graph, landing, continuation, componentwise hsame transport, package provenance, and local naming row.

Endpoint availability is supplied by Definition 473.3, so the target certificate is already one of the displayed rows. Edge-classifier exactness from Definition 473.10 then forces each graph edge to land through E and to transport classifier data only through H and the paired Cont route in C.

Finally apply the public consumer boundary of Definition 473.24. A downstream read can consume only the displayed graph, landing, classifier transport, continuation, package, and local naming rows. Thus every target endpoint route is the stated finite BHist route through S, G, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint route`

Theorem 473.56 (CertificateCompiler identity composite public boundary)**.** *For accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packets, identity rows, binary composites, and triple composites remain inside the same public row surface S, T, G, E, C, H, P, N. The identity packet exposes only the diagonal rows of its local* NameCert *input, a binary composite exposes only the two displayed compiler packets and their shared middle landing, and a triple composite exposes only the three displayed graph edges, the two middle landing rows, the ordered* hsame *transports, the composed* Cont *route, the* Pkg *provenance, and the local* NameCert *rows.*

Proof. For identities, apply Definition 473.9. The diagonal packet reads the carrier, classifier, hsame, Cont, Pkg, and NameCert rows already displayed by the local certificate.

For binary composites, use Definition 473.8 together with Definition 473.12. The composite graph is formed from displayed graph edges, and its landing, classifier, continuation, package, and local naming rows are projected from the two input packets and their middle NameCert row.

For triple composites, apply Definition 473.13. The two associations expose the same three graph edges, two middle landing matches, ordered classifier transports, and continuation endpoint.

The left and right unit laws, Definition 473.14 and Definition 473.15, identify identity composites with the original packet surface. Combining these rows gives the same public boundary S, T, G, E, C, H, P, N for identity, binary, and triple compiler packets. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier identity composite public boundary`

Theorem 473.57 (CertificateCompiler endpoint-route determinacy envelope)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet. Any accepted target endpoint route for K, and any identity, binary composite, or triple composite route built from accepted* CertificateCompiler$^{\uparrow}$ *packets, is determined by the same displayed finite surface S, T, G, E, C, H, P, N. There is no second endpoint route through host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. First apply the target endpoint route of Definition 473.55. It places endpoint arrival inside the displayed source naming row, compiler graph, target landing row, classifier transport, continuation row, package provenance, and local naming row of the accepted carrier.

Next apply the identity composite public boundary of Definition 473.56. Identity rows, binary composites, and triple composites stay on the same public row surface S, T, G, E, C, H, P, N; their middle landings, graph edges, transports, continuation rows, package provenance, and local names are projected from displayed compiler packets.

Finally return to the carrier definition. The accepted packet has no coordinate for host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior. Endpoint arrival and the composite public boundary therefore share the same displayed rows, so the finite surface S, T, G, E, C, H, P, N determines the available endpoint and composite routes. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint route determinacy envelope`

Theorem 473.58 (CertificateCompiler endpoint determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* CertificateCompiler$^{\uparrow}$ *packet. If two accepted target endpoint routes for K agree on the source row S, the compiler graph G, the landing row E, the classifier transport H, the continuation row C, the provenance row P, and the local name N, then they determine the same target certificate row T. No host compiler semantics, optimiser equality, ambient program equality, or unrecorded target behavior can create a second endpoint.*

Proof. First apply the target endpoint route theorem, Definition 473.55. It exhausts every route to T by the displayed source, graph, landing, classifier transport, continuation, package, and local naming rows.

Endpoint certificate availability, Definition 473.3, supplies the target certificate as part of that displayed packet, and edge-classifier exactness, Definition 473.10, keeps graph edges tied to the

same landing and classifier rows. Thus two accepted reads agreeing on S, G, E, H, C, P, N have no further endpoint coordinate to vary.

Now use the public consumer boundary, Definition 473.24, together with Definition 473.56. Public consumers, identities, binary composites, and triple composites remain on the same finite row surface. Since the packet contains no host compiler semantics, optimiser equality, ambient program equality, or unrecorded target behavior, the agreed displayed rows determine the same target certificate row T. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint determinacy`

Theorem 473.59 (CertificateCompiler downstream stage unblock package)**.** *The target endpoint route of Definition 473.55 and the identity/composite boundary of Definition 473.56 supply the public package consumed by* CertificatePipelineStageBoundary$^{\uparrow}$, KernelMorphismUp, *and* DecompilerRightInverse$^{\uparrow}$. *The package consists only of finite* BHist *rows, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *rows over* $K = (S, T, G, E, C, H, P, N)$*; it does not export host compiler semantics, optimiser correctness, decompiler correctness, ambient program equality, or target behavior absent from the packet.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier downstream stage unblock package`

Proof. First build the endpoint part of the package. By Definition 473.55, every accepted target route reads the source naming row, graph row, landing row, classifier transport row, continuation row, package provenance, and local naming row.

Next build the compositional part. By Definition 473.56, identities, binary composites, and triple composites keep the same public row surface and do not add a host compiler coordinate.

Now combine these with Definition 473.23 and Definition 473.26. The source and target endpoints are exhausted by the displayed packet rows, and every finite route stays inside BHist, hsame, Cont, Pkg, and NameCert coordinates.

The consumers named in the statement require exactly this public route surface: pipeline stages read adjacent compiler boundaries, kernel morphisms read source-to-target graph preservation, and decompiler right-inverse packets read the displayed compiler/decompiler interface. Since all three reads factor through the same finite package, none receives a host compiler semantics or optimiser-correctness theorem. □

Theorem 473.60 (CertificateCompiler KernelMorphism route scope)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Any downstream* KernelMorphismUp *consumer reads the compiler graph* G *only through the endpoint route* S, G, E, H, C, P, N *and the source-target graph admission and classifier-lift rows of chapter 620. It cannot import host compiler semantics, optimiser correctness, decompiler correctness, or target behavior absent from the finite packet.*

Proof. First use the target endpoint route of Definition 473.55. It identifies the graph-facing route to the target certificate with the displayed source naming row, compiler graph, target landing row, classifier transport, continuation row, package provenance, and local naming row.

Next apply the identity and composite public boundary of Definition 473.56. Identities, binary composites, and triple composites stay on the same finite endpoint surface, so a downstream morphism consumer receives graph preservation through displayed compiler rows rather than through host program semantics.

Finally combine the downstream stage package of Definition 473.59 with the KernelMorphismUp source-target scope named in chapter 620. The consumer-facing route is exactly the finite BHist

handoff through S, G, E, H, C, P, N together with the KernelMorphism graph-admission and classifier-lift rows, leaving optimiser correctness, decompiler correctness, and unrecorded target behavior outside the packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier kernelmorphism route scope`

Theorem 473.61 (CertificateCompiler KernelMorphism decompiler boundary)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Any route from K into a* KernelMorphismUp *consumer and then into a* DecompilerRightInverse$^{\uparrow}$ *consumer factors through the displayed finite rows S, G, E, H, C, P, N, the* KernelMorphismUp *graph-admission and classifier-lift rows, and the* DecompilerRightInverse$^{\uparrow}$ *inverse-boundary rows of Definition 614.3. The route exports no host compiler semantics, optimiser correctness, ambient program equality, host decompiler function, total decompiler, or unrecorded target behavior.*

Proof. First apply the target endpoint route of Definition 473.55. It places every accepted compiler target read on the displayed finite rows S, G, E, H, C, P, N.

Next apply the KernelMorphism route scope of Definition 473.60. The intermediate morphism consumer can add only the graph-admission and classifier-lift rows licensed by KernelMorphismUp, so the compiler leg remains a finite BHist handoff rather than a host compiler semantics.

Finally apply the decompiler right-inverse non-escape boundary of Definition 614.3. A downstream inverse consumer may read only the displayed compiler packet, decompiler graph, target optimisation row, right-inverse ledger, transport, continuation, provenance, and local naming rows. Combining the three non-escape clauses leaves no route to optimiser correctness, ambient program equality, host decompiler behavior, or target behavior absent from the packet. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier kernelmorphism decompiler boundary`

Theorem 473.62 (CertificateCompiler target endpoint determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Once the source row S, a displayed graph edge in G, its landing row E, the classifier transport row H, and the continuation row C are fixed, the target endpoint route to T is determined as the finite route through S, G, E, H, C, P, N. No second endpoint route can be obtained from host compiler semantics, optimiser correctness, ambient program equality, or an unrecorded target behavior.*

Proof. Start from Definition 473.55. It identifies every target arrival with the displayed source, graph, landing, classifier, continuation, package, and local naming rows.

Use Definition 473.10. With the source row, graph edge, landing row, classifier transport, and continuation row fixed, the edge has no further classifier-facing or endpoint-facing coordinate to vary.

Finally apply Definition 473.26. The carrier exports no external compiler semantics or target behavior, so the fixed displayed rows determine the endpoint route. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint determinacy`

Theorem 473.63 (CertificateCompiler endpoint-route pullback determinacy)**.** *Let a downstream route pass through* CertificateCompiler$^{\uparrow}$ *into* KernelMorphismUp *and then into* DecompilerRightInverse$^{\uparrow}$*. If two such routes have the same displayed source row, graph edge, landing row, classifier transport row, and* Cont *row, then their endpoint pullbacks are the same finite route through S, G, E, H, C, P, N. No host compiler semantics, optimiser correctness, ambient program equality, host decompiler function, or unrecorded target behavior can distinguish the pullbacks.*

Proof. First apply target endpoint determinacy, Definition 473.62. Agreement of the displayed source row, graph edge, landing row, classifier transport row, and continuation row fixes the compiler endpoint route through S, G, E, H, C, P, N.

Next use the KernelMorphism route scope, Definition 473.60. The morphism consumer can add only the graph-admission and classifier-lift rows licensed by KernelMorphismUp, so it cannot separate two already fixed compiler endpoint routes.

Finally apply the KernelMorphism decompiler boundary, Definition 473.61. The decompiler-facing leg remains on the displayed compiler packet and inverse-boundary rows, leaving no coordinate for host compiler semantics, optimiser correctness, ambient equality, host decompiler functions, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint route pullback determinacy`

Theorem 473.64 (CertificateCompiler route composition exactness)**.** *For accepted* CertificateCompiler$^{\uparrow}$ *packets, the identity route, every binary composite route, and every triple composite route use only the displayed graph rows, landing rows, classifier transport rows, and* Cont *rows, together with the visible* Pkg *and local* NameCert *provenance. These routes introduce no hidden program equality, optimiser correctness theorem, ambient compiler semantics, or target behavior outside the finite packet.*

Proof. For the identity and binary cases, apply Definition 473.56. The identity packet reads the diagonal local naming rows, and the binary composite reads the two displayed compiler packets and their shared middle landing.

For the triple case, use the same public boundary together with Definition 473.10. The three graph edges, two middle landing rows, ordered classifier transports, and composed Cont route are exactly the rows visible to the consumer.

The finite non-escape theorem Definition 473.26 excludes any additional semantic coordinate. Hence all identity, binary, and triple composition routes are exact over the displayed rows. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier route composition exactness`

Theorem 473.65 (CertificateCompiler ledger cut elimination)**.** *Let a downstream consumer read the public package supplied by Definition 473.59. An intermediate package row may be cut from that consumer route only when the same displayed endpoint rows and the same identity, binary, or triple composite rows remain preserved in the resulting route. A cut that changes the endpoint route, changes the composite route, or replaces either route by host compiler semantics, decompiler correctness, ambient program equality, or unrecorded target behavior is not a valid* CertificateCompiler$^{\uparrow}$ *consumer read.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier ledger cut elimination`

Proof. First read the downstream package through Definition 473.59. The consumer-visible packet consists only of finite BHist rows, displayed hsame transport, Cont routes, Pkg provenance, and local NameCert rows.

Endpoint preservation is forced by Definition 473.55. If a cut removes an intermediate package row but keeps the same source, graph, landing, classifier, continuation, provenance, and local naming rows, the target route is still the displayed finite route.

Composite preservation is forced by Definition 473.56 and Definition 473.64. Identity, binary, and triple composites remain valid exactly when their displayed graph, landing, classifier, and continuation rows are unchanged.

Finally use Definition 473.26. Any cut that replaces a displayed row by host semantics, decompiler correctness, ambient equality, or unrecorded target behavior leaves the finite packet surface and is therefore unavailable to downstream consumers. □

Theorem 473.66 (CertificateCompiler endpoint decompiler handoff determinacy)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *endpoint route. Once the displayed rows* S, G, E, H, C, P, N *and the* KernelMorphismUp *graph-admission and classifier-lift rows are fixed, any* $\mathsf{DecompilerRightInverse}^{\uparrow}$ *consumer can read only the inverse-boundary rows already cited by Definition 473.61. The handoff cannot introduce host compiler semantics, optimiser correctness, ambient program equality, host decompiler functions, or unrecorded target behavior.*

Proof. First use target endpoint determinacy, Definition 473.62. With the source row, graph edge, landing row, classifier transport, continuation row, provenance, and local naming row fixed, the endpoint route is the displayed finite route through S, G, E, H, C, P, N.

Next apply route composition exactness and ledger cut elimination, Definition 473.64 and Definition 473.65. Identity, binary, and triple composites may be replayed only over the same displayed endpoint rows, and no intermediate cut can replace them by host semantics or unrecorded behavior.

Now pass through the KernelMorphism route scope, Definition 473.60. The morphism leg contributes only the graph-admission and classifier-lift rows licensed by KernelMorphismUp.

Finally apply the KernelMorphism decompiler boundary, Definition 473.61, together with the decompiler right-inverse non-escape boundary of Definition 614.3. A downstream inverse consumer therefore receives only the displayed compiler packet, the fixed KernelMorphism rows, and the inverse-boundary rows, with no coordinate for host compiler semantics, optimiser correctness, ambient equality, host decompiler functions, or unrecorded target behavior. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint decompiler handoff determinacy`

Theorem 473.67 (CertificateCompiler endpoint-composite determinacy)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet. Once the displayed source row, graph edge, landing row, classifier transport, continuation row, and identity, binary, and triple composite rows are fixed, every downstream* KernelMorphismUp *or* $\mathsf{DecompilerRightInverse}^{\uparrow}$ *consumer route is determined by the finite packet* S, G, E, H, C, P, N *together with the relevant composite rows. The determined route exports no host compiler semantics, optimiser correctness theorem, decompiler correctness theorem, ambient program equality, or target behavior outside those finite rows.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint composite determinacy`

Proof. Start with the endpoint route and public composite boundary. The target endpoint route of Definition 473.55 fixes the source, graph, landing, classifier transport, continuation, provenance, and local naming rows, while Definition 473.56 fixes identity, binary, and triple composites on the same displayed surface.

Next expose the downstream packet. Apply the downstream stage unblock package, Definition 473.59, together with KernelMorphism route scope, Definition 473.60. A morphism consumer can read only the finite compiler route and the corresponding graph-admission and classifier-lift rows.

Now attach the decompiler boundary. The KernelMorphism decompiler boundary, Definition 473.61, keeps any route into $\mathsf{DecompilerRightInverse}^{\uparrow}$ on the same finite compiler packet before the inverse-boundary rows are read.

Finally use target endpoint determinacy, route composition exactness, and ledger cut elimination, Definition 473.62, Definition 473.64, and Definition 473.65. These identify all endpoint and composite reads after the listed rows are fixed, so every downstream morphism or decompiler route is determined by S, G, E, H, C, P, N plus the relevant composite rows, with no remaining semantic coordinate available. □

Theorem 473.68 (CertificateCompiler endpoint KernelMorphism roundtrip)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted finite* CertificateCompiler$^{\uparrow}$ *endpoint route. If the source and target endpoint reads agree under Definition 473.62, then the route factors through the* KernelMorphismUp *graph-admission and classifier-lift scope and returns through the* DecompilerRightInverse$^{\uparrow}$ *boundary on the same displayed compiler rows.*

The roundtrip uses only S, G, E, H, C, P, N, the relevant identity, binary, or triple composite rows, the KernelMorphismUp *graph-admission and classifier-lift rows, and the* DecompilerRightInverse$^{\uparrow}$ *inverse-boundary rows. It adds no hidden host compiler semantics, optimiser correctness theorem, decompiler correctness theorem, ambient program equality, total decompiler, or unrecorded target behavior.*

Proof. First apply target endpoint determinacy, Definition 473.62. Agreement of the source and target endpoint reads fixes the displayed finite endpoint route through S, G, E, H, C, P, N.

Next pass through the morphism surface using Definition 473.60. A downstream KernelMorphismUp consumer can read only the graph-admission and classifier-lift rows attached to the same compiler packet, so the route has no host compiler coordinate at the morphism stage.

Now return through the decompiler side by Definition 473.61. The decompiler-facing consumer remains on the displayed compiler packet together with the inverse-boundary rows, rather than importing host decompiler behavior or total decompiler data.

Finally compose the route and remove intermediate package cuts. The composition exactness theorem, Definition 473.64, keeps identity, binary, and triple composites on the displayed rows, while ledger cut elimination, Definition 473.65, permits only cuts preserving those endpoint and composite rows. Thus the endpoint, KernelMorphism, and decompiler legs form a finite roundtrip with no hidden host compiler semantics. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier endpoint kernelmorphism roundtrip`

Corollary 473.69 (CertificateCompiler target endpoint consumer determinacy)**.** *Let*

$$K = (S, T, G, E, C, H, P, N)$$

be an accepted CertificateCompiler$^{\uparrow}$ *packet. Fix the consumer-visible rows cited by Definition 473.57: the source row S, target row T, compiler graph G, target landing row E, continuation row C, classifier transport row H, package provenance row P, local naming row N, and, when a composite is requested, the displayed identity, binary, or triple-composite ledger rows over the same surface. Then any accepted consumer-facing use of K must select the same finite endpoint route, the same composite normal form, or the same classifier-exactness decision named by the* CertificateCompiler$^{\uparrow}$ *chapter. It cannot select an alternate hidden coordinate, a second source classifier row, host compiler semantics, optimiser correctness, ambient program equality, or unrecorded target behavior.*

Proof. First apply Definition 473.55. It exhausts every target arrival by the finite BHist route through the displayed source naming row, graph row, target landing row, classifier transport row,

continuation row, package provenance row, and local naming row. Thus a consumer that asks for the target endpoint has no endpoint coordinate beyond S, G, E, H, C, P, N.

Next apply Definition 473.57. It says that accepted endpoint routes and accepted identity, binary, and triple-composite routes are all determined by the same displayed finite surface S, T, G, E, C, H, P, N. Hence fixing the cited local rows also fixes the composite normal form available to a consumer: the identity normal form is the diagonal row, a binary normal form is the displayed two-packet composite through the shared middle landing, and a triple normal form is the displayed three-edge ledger with its two middle landing rows.

It remains to rule out duplicate classifier sources and alternate exactness decisions. By Definition 473.10, an accepted classifier decision reads the source classifier row in S, lands through E, transports through the componentwise hsame row H, and follows the paired Cont routes in C. With S, E, H, C fixed, there is no second classifier-facing source from which a different decision could be accepted.

Finally use Definition 473.26. The accepted carrier exports no host compiler semantics, optimiser correctness theorem, ambient program equality, or unrecorded target behavior. Any consumer-facing use that agrees on the displayed rows therefore factors through the same finite route surface and must return the same endpoint route, composite normal form, or classifier-exactness decision named above. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier target endpoint consumer determinacy`

Theorem 473.70 (CertificateCompiler bridge schema handoff)**.** *Every public* CertificateCompiler$^{\uparrow}$ *packet hands a schema bridge only the finite compiler graph, target landing rows, classifier-transport rows, continuation rows, package provenance, and source, target, and local* NameCert *rows. The handoff is the displayed packet surface S, T, G, E, C, H, P, N together with its identity, composite, and triple-composite ledgers; it does not include host compiler semantics, optimizer correctness, decompiler behavior, ambient program equality, or unrecorded target behavior.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge schema handoff`

Proof. Start with the carrier of Definition 473.1. It exposes source and target certificates, graph edges, landing rows, continuation rows, componentwise transports, provenance, and a local naming row.

Use the classifier and the NameCert obligations. Definition 473.2 and Definition 473.7 identify the bridge-readable packet with the finite graph, landing, classifier transport, continuation, package, and naming rows.

Now apply edge-classifier exactness, the identity ledger, composition stability, and triple-composite associativity. These theorems show that every schema-facing edge, identity, binary composite, and triple composite is replayed through the same finite rows, so the bridge handoff has no host compiler coordinate. □

Theorem 473.71 (CertificateCompiler bridge-row non-escape)**.** *The bridge-facing rows of a public* CertificateCompiler$^{\uparrow}$ *packet export no optimizer, decompiler, program equality, ambient categorical equality, or unrecorded target behavior. Any schema bridge that consumes the packet must read the source certificate, target certificate, graph edge, landing, route, transport, provenance, and local naming rows exactly as displayed.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge rows nonescape`

Proof. Project the public packet as the finite rows S, T, G, E, C, H, P, N. These rows are the only bridge-facing coordinates of the carrier.

Apply Definition 473.10 and Definition 473.26. They force edge reads through graph, landing, classifier transport, continuation, provenance, and naming rows and exclude host compiler semantics from the packet.

The identity ledger, composition stability, and triple-composite associativity preserve the same row set under bridge replay. Since no displayed row is an optimizer, decompiler, program-equality witness, ambient categorical equality, or target-behavior oracle, none can be exported to the bridge consumer. □

Theorem 473.72 (CertificateCompiler bridge consumer exhaustion)**.** *Every schema-bridge consumer of a public* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet factors through the already displayed source certificate, target certificate, graph edge, landing row, route row, classifier-transport row, provenance row, and local naming row. The consumer can compose, reassociate, and replay certificate-compiler edges only by the identity, binary-composite, and triple-composite ledgers already present in the chapter.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge consumer exhaustion`

Proof. Begin with the public consumer boundary Definition 473.24. It already restricts downstream reads to graph landing, classifier transport, continuation, provenance, and local naming over the source and target certificates.

Add the bridge handoff of Definition 473.70. The bridge receives exactly the same finite graph and certificate rows, with no host compiler semantics.

Finally apply the identity ledger, composition stability, and triple-composite associativity. These results exhaust the bridge consumer's available operations on the packet: identity replay, binary composition, and triple reassociation all factor through the displayed rows and cannot introduce another compiler surface. □

Theorem 473.73 (CertificateCompiler bridge associativity split index)**.** *The bridge handoff of Definition 473.70, the bridge-row non-escape boundary of Definition 473.71, and the bridge consumer exhaustion row of Definition 473.72 all read the same identity, binary-composite, and triple-composite ledger for* $\mathsf{CertificateCompiler}^{\uparrow}$ *packets. A schema bridge therefore consumes one finite ledger surface rather than three unrelated compiler interfaces.*

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge associativity split index`

Proof. First read the identity ledger. Definition 473.9 gives the diagonal compiler packet over the same displayed source, target, graph, landing, classifier transport, continuation, provenance, and naming rows consumed by the bridge handoff.

Next read binary composition. Definition 473.8 and Definition 473.12 keep composite replay on the same public row surface and add no host compiler semantics.

Finally read triple composites. Definition 473.13 and the bridge consumer exhaustion theorem use the same triple of graph edges and middle landing matches, while bridge-row non-escape blocks optimizer, decompiler, program-equality, ambient categorical, and unrecorded target-behavior coordinates. □

Corollary 473.74 (CertificateCompiler bridge schema handoff strict obstruction)**.** *Let* $K = (S, T, G, E, C, H, P, N)$ *be a public* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet read by a schema bridge. If a displayed failure, refusal, separation, or countermodel row is cited against a named bridge-handoff*

converse, reuse, or readback route for K, then that row is a strict local obstruction to that named route only. It blocks exactly the request that tries to read, omit, replace, detach, or bypass one of the displayed source certificate, target certificate, graph edge, landing, route, classifier-transport, provenance, or local naming rows. It cannot be reinterpreted as a global impossibility theorem, a host theorem about compilers, hidden external validation, optimizer correctness, decompiler correctness, ambient program equality, ambient categorical equality, or unrecorded target behavior.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier bridge schema handoff strict obstruction`

Proof. Read the schema bridge handoff first. By Definition 473.70, the bridge receives only the finite compiler graph, target landing rows, classifier-transport rows, continuation rows, package provenance, and source, target, and local NameCert rows of the packet. Its full bridge surface is the displayed tuple S, T, G, E, C, H, P, N together with the identity, binary-composite, and triple-composite ledgers. The same theorem excludes host compiler semantics, optimizer correctness, decompiler behavior, ambient program equality, and unrecorded target behavior from the handoff.

Now apply bridge-row non-escape, Definition 473.71. Any schema bridge that consumes K must read the source certificate, target certificate, graph edge, landing, route, transport, provenance, and local naming rows exactly as displayed. Thus a consumer-facing use of a displayed obstruction row has no coordinate outside the listed bridge rows. Source and target admission are read through S, T; graph and landing evidence through G, E; route and replay through C; classifier transport through H; package provenance through P; and the local certificate name through N.

Use Definition 473.72 and Definition 473.73 to exhaust the remaining bridge operations. A bridge consumer may compose, reassociate, or replay only by the displayed identity, binary-composite, and triple-composite ledgers, and those ledgers read the same finite surface rather than separate compiler interfaces. Hence an obstruction row attached to a named bridge-handoff converse, reuse, or readback route can block only the local request whose reading position asks for an unlisted coordinate or bypasses one of the displayed rows.

Any stronger interpretation would require data that the two bridge theorems and the consumer ledger do not export. A global impossibility theorem would require a theorem surface beyond the finite packet; a host theorem about compilers would require host compiler semantics or host program equality; hidden external validation would require an external certificate channel beyond P and N; optimizer or decompiler correctness would require optimizer or decompiler coordinates; and ambient categorical equality or unrecorded target behavior would require exactly the off-packet data ruled out by bridge-row non-escape. Since every permitted bridge-facing use factors through S, T, G, E, C, H, P, N and the displayed ledgers, the obstruction remains strict and local to the named bridge-handoff route. □

Theorem 473.75 (CertificateCompiler root landing obligation)**.** *Every root consumer of an accepted* CertificateCompiler$^{\uparrow}$ *packet*

$$K = (S, T, G, E, C, H, P, N)$$

must first read the source NameCert *endpoint row S, the target* NameCert *endpoint row T, the displayed compiler graph G, and the edge-landing row E. The consumer may treat a displayed graph edge as landed only after S supplies the source carrier and classifier row, T supplies the target carrier and classifier row, G supplies the finite edge, and E records that the edge starts in the carrier named by S and lands in the carrier named by T.*

No root consumer can obtain a landing witness from host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, or an unrecorded target behavior outside the rows S, T, G, E.

Proof. Unfold the carrier in Definition 473.1. Its first four rows are exactly the source endpoint certificate S, the target endpoint certificate T, the displayed compiler graph G, and the edge-landing row E.

Use the classifier definition in Definition 473.2. Even a paired-edge comparison begins by reading the source classifier row in S and the target landing row in T, so classification cannot precede endpoint availability.

Apply Definition 473.7. Its carrier landing obligation is the finite row assertion that every edge of G reads a source row admitted by S and lands in a target row admitted by T through E.

The carrier has no coordinate for host compiler equality or ambient program equivalence. Therefore a root consumer that consumes landing data is forced through the displayed S, T, G, E rows and cannot source a landing witness elsewhere. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing obligation`

Theorem 473.76 (CertificateCompiler root classifier transport obligation)**.** *For every accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet*

$$K = (S, T, G, E, C, H, P, N),$$

paired-edge classifier transport is exhausted by the source and target endpoint rows S, T, the componentwise hsame *transport row H, and the continuation row C. A root consumer may compare two displayed graph edges only by sharing the source classifier row in S, landing through T, moving the target classifier row through H, and following the paired* Cont *routes recorded in C.*

The transport obligation does not admit host compiler equality, source program equivalence, target program equivalence, or an external semantic relation as a classifier witness.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root classifier transport obligation`

Proof. Unfold Definition 473.1. The transport-relevant rows available to a root consumer are the endpoint certificates S, T, the componentwise transport row H, and the continuation compatibility row C, with G and E only displaying the edges whose classifiers are compared.

Now unfold Definition 473.2. It defines compatibility for paired graph edges by a common source classifier row, a target landing, componentwise hsame transport, and paired continuation routes. These are precisely the rows S, T, H, C named in the statement.

Apply the classifier-transport clause of Definition 473.7. That clause permits transport from the source classifier row to the target classifier row only by the displayed finite-row data already carried by the packet.

Since neither the carrier nor the classifier contains host compiler equality or a program-equivalence coordinate, a root consumer cannot certify paired edges by such an external equality. The finite row surface S, T, H, C exhausts classifier transport. □

Theorem 473.77 (CertificateCompiler root continuation route obligation)**.** *Every downstream root consumer of an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet, including* $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$, KernelMorphismUp, *and* $\mathsf{DecompilerRightInverse}^{\uparrow}$, *reads continuation compatibility only through the displayed* Cont *row C, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *row N together with the endpoint* NameCert *rows S, T. The consumer route factors through the recorded*

source and target continuation pair, the named package surface, and the local certificate row; it does not factor through an unstated compiler semantics or decompiler correctness theorem.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root continuation route obligation`

Proof. Unfold the carrier in Definition 473.1. The only continuation coordinate is C, and the rows that publish it to downstream consumers are the provenance row P and the local naming row N over the endpoint certificates S, T.

Use Definition 473.2. Its paired-edge comparison does not stop at landing; it also reads the source and target Cont routes recorded in C, so any classifier-compatible downstream route has already passed through the displayed continuation row.

Apply Definition 473.7. The continuation-compatibility clause states that each source route used by a graph edge is paired in C with the target route followed after compiling, while the package and local naming rows can only repack these displayed obligations.

The three named root consumers each read a different downstream surface: pipeline-stage boundaries read adjacent compiler routes, kernel morphisms read source-to-target preservation, and decompiler right-inverse packets read the compiler/decompiler interface. In all three cases the available route data are the finite rows C, P, N, S, T and the local NameCert obligations, so no unstated compiler semantics or decompiler correctness theorem can enter the continuation route. □

Theorem 473.78 (CertificateCompiler root landing bridge determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* CertificateCompiler$^{\uparrow}$ *packet. Every consumer-facing bridge use of root landing data for K, including a displayed edge route, classifier transport, endpoint readback, continuation readback, package handoff, or local naming handoff, factors through the local row surface*

$$S, T, G, E, C, H, P, N.$$

More precisely, landing is determined by S, T, G, E, paired-edge classifier transport is determined by S, T, H, C, continuation route readback is determined by C over the endpoint rows S, T, and the handoff rows P, N can only repack those displayed obligations. No host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, external semantic relation, or unrecorded target behavior supplies an additional bridge coordinate.

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing bridge determinacy`

Proof. Start with root landing. By Definition 473.75, a root consumer may treat a displayed graph edge as landed only after reading the endpoint rows S, T, the displayed graph row G, and the edge-landing row E. Thus every bridge use whose first demand is endpoint readback, graph readback, or target landing is already forced through S, T, G, E.

Next read classifier transport. By Definition 473.76, paired-edge comparison is exhausted by the source and target endpoint rows S, T, the componentwise hsame row H, and the continuation row C. Hence a consumer cannot turn a landing bridge into an independent classifier witness: the only accepted transport reads the same endpoints, moves the target classifier through H, and follows the paired Cont routes in C.

Now apply Definition 473.77. A downstream route consumer reads continuation compatibility through C, package provenance through P, and local naming through N together with the endpoint certificates S, T. Therefore route readback, continuation readback, package handoff, and local

naming handoff do not create a second bridge surface; they publish the same finite landing and transport rows to downstream consumers.

Combining the three obligations gives determinacy. The landing rows S, T, G, E decide which displayed edge has landed, the transport rows S, T, H, C decide which paired-edge classifier comparison is available, and the rows C, P, N decide how that route is handed to later consumers. Each listed row has a fixed role in the packet, while the carrier and classifier supply no coordinate for host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, an external semantic relation, or unrecorded target behavior. Any attempted bridge coordinate outside S, T, G, E, C, H, P, N therefore has no row to read, and every permitted consumer-facing use factors through the cited local rows. □

Corollary 473.79 (CertificateCompiler root landing consumer determinacy)**.** *Let $K = (S, T, G, E, C, H, P, N)$ be an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet, and fix the consumer-visible rows cited by Definition 473.78: the source endpoint row S, the target endpoint row T, the displayed compiler graph G, the edge-landing row E, the continuation row C, the componentwise* hsame *transport row H, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CertificateCompiler}^{\uparrow}}$ *row N. Then any accepted consumer-facing use of root landing data for K must pick the same finite displayed edge route, the same continuation readback route, the same package or local naming handoff, and the same classifier-exactness decision determined by those rows. It cannot pick an alternate hidden coordinate, a duplicate source classifier row, a second endpoint readback source, host compiler equality, ambient program equivalence, optimizer correctness, decompiler behavior, an external semantic relation, or unrecorded target behavior.*

Proof. By Definition 473.75, root landing can be consumed only after the consumer has read S, T, G, E. Thus the landing part of any accepted use is already pinned to the source endpoint certificate, target endpoint certificate, displayed graph edge, and edge-landing row named in the packet. There is no carrier coordinate from which the consumer could obtain another landing witness, endpoint readback, or source classifier row.

Apply Definition 473.78. It lists the consumer-facing bridge uses of the same root landing data: displayed edge route, classifier transport, endpoint readback, continuation readback, package handoff, and local naming handoff. The theorem states that each such use factors through the finite row surface

$$S, T, G, E, C, H, P, N.$$

It also separates the roles of the rows: S, T, G, E determine landing, S, T, H, C determine paired-edge classifier transport, C over S, T determines continuation readback, and P, N only repack those obligations for package and local naming handoff.

The remaining alternatives are therefore ruled out by absence of a row to read. A claimed alternate finite route would have to alter the displayed graph, edge landing, continuation row, or handoff row, but those are fixed as G, E, C, P, N. A claimed different normal form would have to use an identity, composite, or readback surface outside the displayed packet, but Definition 473.78 permits only the listed bridge surface for consumer-facing use. A claimed different classifier-exactness decision would have to replace the source classifier row in S, the target classifier row in T, the componentwise transport row H, or the paired Cont row C, but the classifier transport part of the bridge determinacy theorem exhausts exactly S, T, H, C.

Consequently, once the cited local rows are fixed, every accepted consumer-facing use lands on the same displayed route, readback route, handoff, or exactness decision named by the $\mathsf{CertificateCompiler}^{\uparrow}$ chapter. A hidden coordinate, duplicate classifier source, or external semantic witness is not an accepted consumer source for root landing data. □

Lean4 ✓ `BEDC.Derived.CertificateCompilerUp.CertificateCompilerCarrier root landing consumer determinacy`

Chapter 474

A Concrete Naming Certificate for $\mathrm{MultiHistConfig}^{\uparrow}$

The $\mathrm{MultiHistConfig}^{\uparrow}$ interface records the first concrete object in the inter-Hist locality chain of chapter 9419. It packages a finite configuration of at least two BHist observer rows, componentwise hsame transport, finite ProbeBundle ledgers, displayed Cont routes, and Pkg provenance into a local naming certificate. The packet is deliberately prior to $\mathsf{CrossHistCausal}^{\uparrow}$ and $\mathsf{MaxCausalRate}^{\uparrow}$: it supplies the configuration carrier on which those later predicates may be read, while exporting no global clock, no selected synchronization history, and no observer outside the listed rows.

The $\mathrm{MultiHistConfig}^{\uparrow}$ packet is the arity-general configuration carrier extending the bilateral event-record surface of chapter 509. The $\mathsf{LocalityCell}^{\uparrow}$ packet supplies the two-observer event-record and GAP coordination row; the present packet retains only finite observer rows, finite ProbeBundle ledgers, the no-global-sync witness, hsame transport, Cont routes, Pkg provenance, and local NameCert, so downstream $\mathsf{CrossHistCausal}^{\uparrow}$ and $\mathsf{MaxCausalRate}^{\uparrow}$ chapters can consume finite configurations without a global frame.

Definition 474.1 (MultiHistConfig carrier)**.** *A* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (H, L, S, R, C, P, N)$$

with the following visible coordinates:

1. $H = (h_0, h_1, \ldots, h_{n-1})$ *is a finite row of* BHist *observer histories with* $n \geq 2$*;*
2. $L = (\ell_0, \ell_1, \ldots, \ell_{n-1})$ *records one finite* ProbeBundle *ledger for each observer row, indexed by the same finite row as* H*;*
3. S *is the displayed no-global-sync witness, stating that no extra coordinate selects a common clock, a synchronization history, or a privileged observer outside the finite index ledger;*
4. R *records componentwise* hsame *comparisons between accepted presentations of the same observer row;*
5. C *stores the displayed* Cont *visibility routes by which each finite ledger row is read from its observer history and transported to the local packet boundary;*
6. P *is the* Pkg *provenance naming* BHist*,* ProbeBundle*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the inter-Hist locality source chapter;*
7. N *is the local* $\mathsf{NameCert}_{\mathrm{MultiHistConfig}^{\uparrow}}$ *row.*

The local classifier relates two carriers exactly when they have the same finite arity and their corresponding rows H, L, S, R, C, P, N compare componentwise after the displayed hsame *and* Cont *routes. It contains no coordinate for a global frame, a shared time index, a privileged observer, or a synchronization history.*

Lean4 (definition) `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier`

Theorem 474.2 (MultiHistConfig carrier habitation)**.** *The carrier of Definition 474.1 is inhabited whenever two displayed* BHist *observer rows, their finite* ProbeBundle *ledgers, a no-global-sync witness, componentwise* hsame *witnesses, displayed* Cont *routes,* Pkg *provenance, and a local naming row are supplied. The smallest accepted carrier has arity* 2 *and already records non-degenerate inter-Hist configuration data.*

Proof. Take the two supplied observer rows as h_0 and h_1. Place their finite ledgers into L, record the no-global-sync witness as S, place their componentwise comparisons into R, and put their displayed continuation routes into C.

The package coordinate P records the listed kernel dependencies, and N records the local naming row. These seven visible coordinates assemble the packet $M = (H, L, S, R, C, P, N)$ with arity 2.

Since the definition only requires finite arity at least two and visible kernel rows, this packet is an accepted MultiHistConfig$^{\uparrow}$ carrier. No global synchronization datum is used in the assembly. □

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier habitation`
↪ variant `BEDC.Derived.MultiHistConfigUp.MultiHistConfig carrier habitation`

Theorem 474.3 (MultiHistConfig componentwise classifier stability)**.** *The* MultiHistConfig$^{\uparrow}$ *classifier is stable under componentwise* hsame *transport. If two accepted carriers have the same finite arity and each observer row, finite ledger, route table, package row, and local naming row is transported through the displayed componentwise data, then the two carriers are locally classified together.*

Proof. Project the two accepted carriers through Definition 474.1. Equal arity aligns the observer rows and ledger rows index by index.

For each component, use the displayed hsame entry in R and replay the corresponding Cont route in C. This transports the observer row and its finite ledger to the matching component of the second carrier.

The no-global-sync witness, package row, and local naming row are compared by the same componentwise rule. Therefore every visible coordinate of the first packet is matched by a transported coordinate of the second packet, which is exactly the local classifier condition. □

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier componentwise classifier stability`

Theorem 474.4 (MultiHistConfig no-global-sync ledger)**.** *No consumer of a* MultiHistConfig$^{\uparrow}$ *carrier can extract a global frame, shared clock, privileged observer, or synchronization history from the public surface. Any such datum would have to be a visible coordinate of $M = (H, L, S, R, C, P, N)$, but the carrier exposes only observer rows, finite ledgers, a no-global-sync witness, componentwise transports, continuation routes, package provenance, and the local naming row.*

Proof. Inspect the coordinate inventory in Definition 474.1. The observer list H records only the displayed histories, and the ledger list L records only finite probe observations attached to those histories.

The no-global-sync coordinate S is negative ledger data: it records the absence of a common clock, synchronization history, or privileged observer in the packet. The transport coordinate R and route coordinate C compare or route existing components; they do not create a new history standing above the configuration.

The package coordinate P names dependencies, and N names the local certificate row. There is no coordinate left from which a global frame, shared clock, privileged observer, or synchronization history could be projected. Hence any consumer that asks for such a datum must leave the public $\mathrm{MultiHistConfig}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier no global sync ledger`

Theorem 474.5 (MultiHistConfig finite ledger obligations)**.** *The public ledger of an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier is exactly the finite index row of* ProbeBundle *entries displayed in L, the no-global-sync witness S, and the routes in C that attach those entries to the observer rows in H. It exports no unrecorded observation, no host-side list equality, and no quotient of configurations by an external relation.*

Proof. Start with an accepted carrier $M = (H, L, S, R, C, P, N)$. By definition, every ledger entry visible to the public certificate is one of the finite ProbeBundle rows in L, indexed by the same finite row as the histories in H.

The only way such a ledger entry is tied to an observer row is through the displayed Cont routes in C. Componentwise hsame data in R may transport an already displayed entry, but it cannot add a new ledger coordinate.

The no-global-sync witness S records the boundary condition that no common clock, synchronization history, or privileged observer is available to complete the ledger externally. The package and naming rows contain provenance and certificate identity, not extra observations. Thus the ledger surface is exhausted by L, S, and C, with no host-side list equality, quotient carrier, or hidden observation source. □

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier finite ledger obligations`

Theorem 474.6 (MultiHistConfig handoff boundary)**.** *The* $\mathrm{MultiHistConfig}^{\uparrow}$ *surface is a handoff object for later* $\mathsf{CrossHistCausal}^{\uparrow}$ *and* $\mathsf{MaxCausalRate}^{\uparrow}$ *chapters. Downstream chapters may consume the finite observer rows, no-global-sync witness, componentwise* hsame *classifier, finite ledgers, displayed* Cont *routes, and* Pkg *provenance of Definition 474.1; they may not treat the packet itself as a cross-Hist causal predicate or as a maximum-rate bound.*

Proof. The carrier supplies exactly the data needed to state later inter-Hist predicates over a finite configuration: the observer row H, the finite observation ledgers L, the no-global-sync witness S, componentwise classifier data R, continuation routes C, and provenance P, N.

Cross-Hist causal dependence requires an additional predicate over pairs or routed windows of those rows. A maximum-rate bound requires a further unary bound extracted from such causal predicates. Neither predicate is a coordinate of M.

Therefore the accepted packet is a configuration boundary for later chapters. It provides the finite carrier and local classifier they can consume, while leaving causal dependence and rate extraction outside the present public surface. □

Theorem 474.7 (MultiHistConfig NameCert obligation package)**.** *For an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier*

$$M = (H, L, S, R, C, P, N),$$

the carrier row, habitation theorem, componentwise classifier stability, no-global-sync ledger, finite ledger obligations, and downstream handoff assemble the local $\mathsf{NameCert}_{\mathrm{MultiHistConfig}^{\uparrow}}$ *obligation package over* BHist, ProbeBundle, hsame, Cont, Pkg, *and* NameCert. *The package exposes exactly the seven visible coordinates of* M *and leaves causal predicates and maximum-rate bounds outside the present packet.*

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier namecert obligation package`

Proof. Project the accepted packet through Definition 474.1. The seven coordinates H, L, S, R, C, P, N give the finite observer rows, probe ledgers, no-global-sync witness, componentwise transport table, continuation routes, package provenance, and local naming row.

The carrier and habitation rows are supplied by Definition 474.2; the componentwise classifier row is supplied by Definition 474.3. These rows bind the local classifier to the displayed BHist and ProbeBundle data rather than to any host-side equality or synchronization object.

The no-global-sync and finite-ledger rows are supplied by Definition 474.4 and Definition 474.5. They state that the visible ledger is exhausted by L, S, and C, with no global frame, shared clock, hidden observation, external quotient, or unrecorded list equality.

Finally, Definition 474.6 identifies the downstream boundary. A later inter-Hist consumer may read the finite carrier, classifier, ledgers, routes, provenance, and naming row, but any cross-Hist causal predicate or maximum-rate bound must be supplied by that later chapter as an additional row. □

Theorem 474.8 (MultiHistConfig component transport scope)**.** *For an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier* $M = (H, L, S, R, C, P, N)$*, every classifier movement factors componentwise through the finite observer rows* H*, the finite* ProbeBundle *ledgers* L*, the displayed* hsame *transport rows* R*, and the* Cont *routes* C*. No movement of* M *is licensed by a parameter transport, a global observer frame, or an unlisted synchronization coordinate.*

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier component transport scope`

Proof. Project M through Definition 474.1. The classifier has one finite observer row family, one matching finite ledger family, one transport table, and one route table.

A classifier movement must therefore align the arity of H and carry each observer row together with its ledger row through the corresponding entries of R and C. The no-global-sync witness S prevents adding a shared clock or privileged observer as a transport source.

The package and local naming rows record provenance and certificate identity for the same finite packet. They do not add a parameter-level motion outside the listed component rows, so the componentwise route exhausts the transport scope. □

Theorem 474.9 (MultiHistConfig consumer projection boundary)**.** *Every consumer of an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier may project only the listed observer rows* H*, finite* ProbeBundle *ledgers* L*, no-global-sync witness* S*, transport rows* R*, continuation routes* C*, provenance row* P*, and local* NameCert *row* N*. The public surface exports no global clock, quotient configuration, hidden synchronization history, or observer outside the finite row.*

Lean4 ✓ `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier consumer projection boundary`

Proof. The public packet is exactly $M = (H, L, S, R, C, P, N)$. A consumer projection is a read of one of these coordinates through the routes displayed in the carrier.

The finite observer rows and ledgers give the positive projection surface. The no-global-sync witness supplies the negative boundary, while the transport and route coordinates explain how the same listed rows may be read after componentwise classifier movement.

There is no coordinate for a global clock, quotient of configurations, hidden synchronization history, or external observer. Any consumer asking for one of those objects leaves the $\mathrm{MultiHistConfig}^{\uparrow}$ NameCert surface rather than projecting a public row. □

Closure status for $\mathsf{NameCert}_{\mathrm{MultiHistConfig}^{\uparrow}}$.

Constructive story: A $\mathrm{MultiHistConfig}^{\uparrow}$ packet is generated from two or more BHist observer rows, finite ProbeBundle ledgers, a no-global-sync witness, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathrm{MultiHistConfig}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scoped packet binds the carrier, component transport scope, finite ledger obligations, and consumer projection boundary over the same BHist, ProbeBundle, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathrm{MultiHistConfig}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MultiHistConfigUp.MultiHistConfigCarrier\_finite\_ledger\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not define a cross-Hist causal predicate, a maximum causal rate, Lorentz geometry, curved spacetime gluing, quantum-state superposition, or a checked bridge certificate.

Upgrade path: The theory axis is at mature closure; remaining movement is verification-axis bridge checking for the existing public packet and its downstream $\mathsf{CrossHistCausal}^{\uparrow}$ and $\mathsf{MaxCausalRate}^{\uparrow}$ consumers.

Chapter 475

A Concrete Naming Certificate for $\mathsf{TransportedStationaryDiagonal}^{\uparrow}$

The $\mathsf{TransportedStationaryDiagonal}^{\uparrow}$ interface records the transported stationary diagonal finite-window seal used by the Real and Completion phase. It packages the transported $\mathsf{Rat}^{\uparrow}$ seed row, the induced constant $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window seal, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the receiving $\mathsf{Real}^{\uparrow}$ row as a single finite naming certificate. The packet is a concrete handoff object; it contains no quotient stream, host real equality, selected host limit, choice principle, or real-completeness theorem.

Definition 475.1 (Transported stationary diagonal carrier)**.** *A* $\mathsf{TransportedStationaryDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, S, G, D, R, F, C, P, N)$$

with the following displayed coordinates:

1. *Q is the transported* $\mathsf{Rat}^{\uparrow}$ *classifier row for the stationary rational seed pair;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier induced by Q;*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal row read from the same constant stream window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the dyadic endpoints, tolerance rows, and observation bounds inspected by the finite window;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier row receiving the transported stationary diagonal observation;*
6. *F records the exact finite-window seal equation tying Q, S, G, D, and R to the same stationary row;*
7. *C stores the displayed* hsame *comparisons and* Cont *routes for the rational seed, stream window, regular-rational seal, dyadic ledger, and real-facing row;*
8. *P is the* Pkg *provenance naming* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *source rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TransportedStationaryDiagonal}^{\uparrow}}$ *row.*

The local classifier relates two carriers exactly when the rows $Q, S, G, D, R, F, C, P, N$ compare componentwise after the displayed hsame *and* Cont *routes. No comparison may inspect a quotient stream, host equality, external limit choice, or ambient completeness datum.*

Theorem 475.2 (Transported stationary diagonal NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{TransportedStationaryDiagonal}^{\uparrow}}$ *surface has five obligations:*

1. *carrier: formation of the finite packet of Definition 475.1;*
2. *classifier: componentwise comparison of the transported* $\mathsf{Rat}^{\uparrow}$ *seed row, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *row, seal equation, route table, package provenance, and local certificate row;*
3. *stability: the classifier is preserved by the displayed* hsame *comparisons and* Cont *routes, so a transported stationary seed row continues to induce the same constant window, seal row, dyadic ledger, and real-facing row;*
4. *exact finite-window seal: the four public readings* S*,* G*,* D*, and* R *all read the same transported stationary row named by* Q*;*
5. *ledger: the public export contains only the displayed finite rows and excludes quotient streams, host real equality, selected host limits, choice principles, and real-completeness data.*

Proof. Carrier formation is the tuple assembly of Definition 475.1. The transported rational seed supplies Q, the constant-stream read supplies S, the regular-rational boundary supplies G, the dyadic observation boundary supplies D, and the real-facing boundary supplies R. The seal equation F, route table C, package row P, and local naming row N are visible coordinates of the same finite packet.

For the classifier obligation, compare the packet component by component after applying the displayed hsame and Cont routes. The transported $\mathsf{Rat}^{\uparrow}$ row determines the constant $\mathsf{StreamName}^{\uparrow}$ window, the window determines the $\mathsf{RegSeqRat}^{\uparrow}$ seal read, and the dyadic ledger supplies the observation rows consumed by the $\mathsf{Real}^{\uparrow}$ boundary.

Stability is the same componentwise comparison read as a transport statement. Since C contains only displayed hsame comparisons and Cont routes, moving an accepted carrier along those routes preserves the induced window, regular-rational seal, dyadic ledger, and real-facing classifier.

Exact finite-window sealing is the stationary diagonal calculation at the finite boundary. The row F records that S, G, D, and R are four public projections of the same transported stationary row named by Q.

The ledger clause follows by constructor inspection. The carrier has no coordinate for quotient streams, host real equality, selected host limits, choice, or a real-completeness theorem, so no public read can export one. □

Theorem 475.3 (Transported stationary diagonal window-seal exhaustion)**.** *Let* $T = (Q, S, G, D, R, F, C, P, N)$ *be an accepted* $\mathsf{TransportedStationaryDiagonal}^{\uparrow}$ *carrier. Every finite Real and Completion read of the transported stationary diagonal factors through the same five displayed rows* Q, S, G, D, R *and the same exact seal equation* F*. In particular, any two accepted seal rows over the same transported* $\mathsf{Rat}^{\uparrow}$ *seed row and displayed finite-window ledger have the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *public readings after the routes recorded in* C*.*

Proof. Project the accepted carrier through Definition 475.1. The only rows that can serve a finite Real and Completion read are the transported rational seed Q, the constant stream window S, the regular-rational seal G, the dyadic ledger D, and the real-facing row R.

Apply the exact finite-window seal clause of Definition 475.2. It identifies the four public readings S, G, D, and R as projections of the same transported stationary row named by Q.

The stability clause transports this identity through the route table C. Therefore two accepted seal rows with the same transported seed and finite-window ledger have matching public readings at the StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp boundaries.

The ledger clause excludes any further coordinate through which a second seal could distinguish itself. Thus finite-window seal exhaustion is a property of the displayed BEDC packet itself. □

Theorem 475.4 (Transported stationary diagonal Rat–Stream sibling link)**.** *Every accepted* TransportedStationaryI *packet consumes the transported* $\mathsf{Rat}^{\uparrow}$ *seed through chapter 37, the constant finite window through chapter 126, and the regular, dyadic, and real boundary rows through chapter 301, chapter 291, and chapter 38. The carrier coordinates* $Q, S, G, D, R, F, C, P, N$ *expose this sibling route as a finite projection surface, not as an ambient quotient stream, selected limit, or host real equality.*

Proof. Project the accepted packet by Definition 475.1. The coordinate Q is the transported rational seed row, S is the induced constant stream window, G is the regular-rational seal, D is the dyadic ledger, and R is the real-facing row.

Apply the window-seal exhaustion theorem Definition 475.3. It forces the Real and Completion read to pass through the same five displayed rows Q, S, G, D, R and the same seal equation F.

The sibling chapters named in the statement are therefore consumed exactly at the coordinates where the carrier stores them. The route table C, provenance row P, and local certificate N authenticate those projections without adding a quotient stream, selected limit, or host equality row. □

Closure status for $\mathsf{NameCert}_{\mathsf{TransportedStationaryDiagonal}\uparrow}$.

Constructive story: A $\mathsf{TransportedStationaryDiagonal}^{\uparrow}$ packet is generated from a transported RatUp seed row, constant StreamNameUp finite window, RegSeqRatUp seal row, DyadicRatCoreUp observation ledger, RealUp finite-window row, exact seal equation, displayed transport routes, package provenance, and local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TransportedStationaryDiagonal}\uparrow}, \mathsf{seed})$

Scope closed: The seed scope records the finite carrier, componentwise classifier, transport stability, exact finite-window seal, and public ledger boundary for transported stationary diagonal handoff packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TransportedStationaryDiagonal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close full real completeness, quotient-stream equality, host real equality, selected host limits, choice principles, or a standard bridge.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the five displayed rows.

Chapter 476

A Concrete Naming Certificate for $\mathsf{TypeClassifierMembership}^{\uparrow}$

The $\mathsf{TypeClassifierMembership}^{\uparrow}$ surface realizes type checking as a finite classifier-membership packet over displayed BEDC histories. It records a term row, a judgment ledger, an Ext membership row, a Cont reduction row, and local naming provenance, so type membership and subject-reduction reads remain inside BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data.

The packet consumes the finite classifier row of chapter 489 and the visible reduction boundary of chapter 483; $\mathsf{TypeClassifierMembership}^{\uparrow}$ combines those sibling rows into the local T, J, E, R, H, C, P, N membership packet rather than introducing a new host type checker.

Before the local membership packet reads $\mathsf{ClassifierTyping}^{\uparrow}$, its accepted and refused Ext membership rows are constrained by the finite accepted/refused boundary of chapter 595; that boundary separates membership and refusal rows from host type checking, hidden proof search, semantic equality, and quotient histories.

Definition 476.1 (TypeClassifierMembership carrier)**.** *A* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, J, E, R, H, C, P, N)$$

with the following displayed coordinates:

1. *T is the displayed term row whose finite syntax window is being checked;*
2. *J is the finite judgment ledger recording the available typing judgments and their dependency order;*
3. *E is the* Ext *membership row saying that the displayed term row is admitted by the local classifier;*
4. *R is the* Cont *reduction row recording the typed continuation steps whose source and target terms are visible in the packet;*
5. *H records the componentwise* hsame *transports for term rows, judgment ledgers, membership rows, reduction rows, dependency packages, and local provenance;*
6. *C records the* Cont *read routes for accessing the judgment ledger, the* Ext *membership row, the typed reduction row, and the subject-reduction boundary;*
7. *P packages the dependencies on* Ext, Cont, *the judgment ledger, the surrounding* $\mathsf{NameCert}$ *rows, and the BEDC carrier rows used by the local checker;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}$ *provenance row.*

The carrier contains no host type-theory semantics, no semantic equality of terms, no global normalization theorem, and no undecidable classifier outside the finite membership and reduction rows.

Definition 476.2 (TypeClassifierMembership classifier)**.** *For accepted* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *carriers* X *and* X'*, the classifier compares the judgment ledgers,* Ext *membership rows, and* Cont *reduction rows componentwise under the displayed* hsame *transports in* H *and the read routes in* C*. A comparison may inspect the term row* T*, the dependency package* P*, and the local provenance row* N *only as named carrier coordinates.*

The classifier does not compare host typing derivations, semantic equality of terms, erased representatives, or any unrecorded conversion relation.

Lean4 (definition) `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipClassifier`

Theorem 476.3 (TypeClassifierMembership NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}$ *surface for Definition 476.1 consists of five obligations:*

1. *carrier admission for the finite packet* $X = (T, J, E, R, H, C, P, N)$*;*
2. *classifier stability for the judgment,* Ext*, and reduction rows under the displayed* hsame *and* Cont *routes;*
3. Ext *stability, saying that a transported accepted judgment ledger preserves the membership row* E*;*
4. *subject-reduction boundary, saying that a visible reduction row* R *whose source is admitted by* E *can read only the displayed target membership row;*
5. *non-escape from host type theory, semantic equality of terms, undecidable conversion, unrecorded normalization, or any classifier not named by* J*,* E*,* R*,* H*,* C*,* P*, and* N*.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket namecert obligations`

Proof. Carrier admission is the finite packet of Definition 476.1. Its coordinates display the term row, judgment ledger, membership row, reduction row, transport rows, continuation routes, dependency package, and local certificate row.

Classifier stability follows from Definition 476.2. The only comparisons are componentwise reads of the judgment, Ext, and reduction rows through H and C.

Ext stability and subject reduction are boundary reads, not external semantic principles. The row E is the visible membership witness, and R is the visible typed reduction route whose target membership can only be obtained through the recorded subject-reduction boundary.

The non-escape clause is the carrier inventory. Since no coordinate names host typing semantics, semantic equality, undecidable conversion, or a hidden normalization oracle, no consumer can read those objects from the local $\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}$ surface. □

Theorem 476.4 (TypeClassifierMembership subject-reduction boundary)**.** *Let* $X = (T, J, E, R, H, C, P, N)$ *be an accepted* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *carrier. Every public subject-reduction read starts from the displayed term row* T*, the judgment ledger* J*, the source membership row* E*, and the reduction row* R*, then reaches the target membership boundary only through the recorded* hsame*,* Cont*,* Pkg*, and* NameCert *routes. It does not appeal to host reduction semantics, global normalization, or semantic equality of terms.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket subject reduction boundary`

Proof. Project the accepted carrier. The only source of type membership is E, and the only source of reduction information is R.

Use classifier stability from Definition 476.2. Transport can move the judgment, membership, and reduction rows only along the displayed hsame routes, and consumer access is mediated by C.

Finally apply the obligation surface of Definition 476.3. Its subject-reduction boundary repacks the visible source membership and reduction rows into the target membership read, so host reduction semantics and semantic equality never enter the packet. □

Theorem 476.5 (TypeClassifierMembership non-escape)**.** *No accepted* TypeClassifierMembership$^{\uparrow}$ *carrier exports a host type checker, an undecidable semantic equality oracle, a hidden conversion relation, an erased representative of a typing derivation, or an ambient normalization theorem. Every public type-checking read factors through the finite judgment ledger J, the* Ext *membership row E, the* Cont *reduction row R, displayed* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *row of Definition 476.1.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket non escape`

Proof. Inspect the carrier inventory. The public rows are exactly T, J, E, R, H, C, P, N, with J recording the finite judgments, E recording classifier membership, and R recording typed reduction.

Apply Definition 476.4 for reduction-facing consumers and Definition 476.2 for classifier-facing consumers. Both routes pass through the same displayed transport and continuation rows.

Since no coordinate of the packet names a host type checker, semantic equality oracle, hidden conversion relation, erased derivation representative, or ambient normalization theorem, those objects cannot be exported by the TypeClassifierMembership$^{\uparrow}$ surface. □

Theorem 476.6 (TypeClassifierMembership Ext membership stability)**.** *Let $X = (T, J, E, R, H, C, P, N)$ be an accepted* TypeClassifierMembership$^{\uparrow}$ *carrier. Any transported accepted judgment ledger for X preserves the displayed* Ext *membership row E through the componentwise* hsame *transport row H and the* Cont *read routes C. The transported read does not replace E by host typing semantics, semantic equality of terms, undecidable conversion, or an unrecorded membership predicate.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket ext membership stability`

Proof. Start from the carrier of Definition 476.1. Its judgment and membership information is displayed only by the rows J and E, and the only transport and read coordinates are H and C.

Apply the classifier description in Definition 476.2. Classifier comparisons of the judgment ledger and Ext row are componentwise reads through those displayed routes, so a transported accepted ledger still reads the same membership coordinate E.

The obligation surface in Definition 476.3 records this as Ext stability. Since no carrier coordinate names host typing semantics or an unrecorded predicate, the transported ledger cannot substitute any external membership witness for E. □

Theorem 476.7 (TypeClassifierMembership reduction target coverage)**.** *Let $X = (T, J, E, R, H, C, P, N)$ be an accepted* TypeClassifierMembership$^{\uparrow}$ *carrier. Every typed* Cont *reduction target membership read is covered by the displayed rows T, J, E, R, H, C, P, N. In particular, the target membership is reached from the finite judgment ledger, the visible* Ext *membership row, the reduction row, componentwise transport, continuation routes, dependency package, and local naming row, not from host reduction semantics or hidden normalization.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket reduction target coverage`

Proof. Project the accepted carrier. The reduction source and target are visible only through T, J, E, R, while H and C provide the recorded transport and continuation access to those rows.

Apply Definition 476.4. It sends each public subject-reduction read from the source membership and visible reduction row to the target membership boundary through the same hsame, Cont, Pkg, and NameCert routes.

The non-escape theorem Definition 476.5 excludes host reduction semantics, hidden conversion, erased derivations, and ambient normalization from the exported surface. Thus every typed reduction target membership read is covered by the packet rows T, J, E, R, H, C, P, N. □

Theorem 476.8 (TypeClassifierMembership obligation closure package)**.** *Every accepted* TypeClassifierMembership *packet carries the obligation package consisting of carrier admission, classifier stability,* Ext *membership stability, subject-reduction boundary coverage, reduction target coverage, and public non-escape. The package is witnessed by the carrier and classifier surfaces together with the finite rows T, J, E, R, H, C, P, N, and it uses no host type semantics, semantic equality of terms, hidden conversion relation, global normalization theorem, or unrecorded classifier.*

Proof. Carrier admission is supplied by Definition 476.1. Classifier stability is supplied by Definition 476.2, which limits comparison to the displayed judgment, membership, and reduction rows under H and C.

For membership preservation, apply Definition 476.6. For the subject-reduction boundary, apply Definition 476.4. For target coverage, apply Definition 476.7.

It remains to close the public boundary. Apply Definition 476.5; the packet exports only T, J, E, R, H, C, P, N and no host type checker, semantic equality oracle, hidden conversion relation, erased derivation representative, or ambient normalization theorem. □

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket obligation closure package`

Theorem 476.9 (TypeClassifierMembership visible target determinacy)**.** *Let $X = (T, J, E, R, H, C, P, N)$ be an accepted* TypeClassifierMembership$^{\uparrow}$ *carrier. Any two public subject-reduction reads that use the same displayed term row T, judgment ledger J, source* Ext *membership row E, reduction row R, and the same transport, continuation, provenance, and naming route H, C, P, N reach the same displayed target membership boundary. No second target membership row can be supplied by host semantics, semantic equality, hidden conversion, or normalization.*

Lean4 ✓ `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket visible target determinacy`

Proof. Project the accepted carrier and fix the displayed rows T, J, E, R, H, C, P, N. Both public reads therefore start from the same term row, judgment ledger, source membership row, and visible reduction row.

Apply Definition 476.4. It forces each read through E and R and prevents a subject-reduction boundary from entering through host reduction semantics or semantic equality.

Next apply Definition 476.7. The target boundary is identified with the same finite packet rows T, J, E, R, H, C, P, N for both reads, so there is no independent coordinate from which a second target membership row could be read.

Finally apply Definition 476.5. Hidden conversion, host semantics, erased derivations, and normalization are outside the packet, so the two public reads terminate at the same displayed target membership boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeClassifierMembership}^{\uparrow}$ packet is generated from a displayed term row, finite judgment ledger, Ext membership row, typed Cont reduction row, componentwise hsame transports, continuation routes, dependency package, and local NameCert provenance row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 476.1, Definition 476.2, Definition 476.3, Definition 476.4, Definition 476.5, Definition 476.6, Definition 476.7, Definition 476.8, and Definition 476.9 bind the finite rows T, J, E, R, H, C, P, N, Ext, Cont, hsame, Pkg, NameCert, the subject-reduction boundary, and the public packet export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeClassifierMembership}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TypeClassifierMembershipUp.TypeClassifierMembershipPacket\_obligation\_closure\_package`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host type semantics, semantic equality of terms, global normalization, Lean verification, or bridge checking.

Upgrade path: The paper closure axis is complete for this finite type-classifier membership package; Lean verification and bridge checking remain independent axes.

Chapter 477

A Concrete Naming Certificate for ObserverHistoryIdentity$^{\uparrow}$

The ObserverHistoryIdentity$^{\uparrow}$ interface names the finite certificate by which an observed history is compared with another observed history without introducing an observer-substance, host equality of histories, quotient identity, or a global subject parameter. Its public surface is a local NameCert packet over displayed BHist rows, hsame comparisons, Cont routes, licensed ProbeBundle and SigRel observations, and package provenance.

This chapter consumes the no-subject observer-history surface of chapter 479 before comparing licensed ProbeBundle and SigRel observations. The two packets differ in scope: ObserverHistoryIdentity$^{\uparrow}$ records pairwise observation-ledger comparison, while ObserverHistIdentity$^{\uparrow}$ records the single packet that excludes observer-substance.

The licensed observer comparison packet of chapter 1031 consumes this pairwise ProbeBundle/SigRel comparison row as its observer-facing observation lattice input.

Theorem 477.1 (ObserverHistoryIdentity consumes ObserverHistIdentity)**.** *Every accepted* ObserverHistoryIdentity *packet consumes the no-subject* ObserverHistIdentity$^{\uparrow}$ *packet of chapter 479 before pairwise* ProbeBundle *and* SigRel *comparison is read. The dependency projects the displayed rows $H, K, B, S, R, L, C, P, N$ through the no-subject observer-history boundary first, and only then exposes the pairwise probe and signature ledgers.*

Proof. Project the rows $H, K, B, S, R, L, C, P, N$ named by the carrier below. The history rows H and K are observed histories before they are compared, so their no-subject boundary is inherited from chapter 479.

After that boundary is visible, the pairwise rows B and S may be read as licensed ProbeBundle and SigRel comparisons. The rows R, L, C, P, N then record sameness, ledger, replay, provenance, and local naming for the same dependency route. □

Definition 477.2 (Observer history identity carrier)**.** *An* ObserverHistoryIdentity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (H, K, B, S, R, L, C, P, N)$$

with the following visible rows:

1. *H and K are the two observed* BHist *histories whose displayed rows may be compared;*
2. *B is the finite* ProbeBundle *observation ledger naming the licensed probes applied to H and K;*
3. *S is the finite* SigRel *ledger of signature observations produced by the probes in B;*

4. *R records the displayed* hsame *comparisons between the kernel rows of H and K that are available to the classifier;*
5. *L is the local observation ledger tying each displayed probe and signature row to the corresponding history row and to its local naming row;*
6. *C records the visible* Cont *routes by which a consumer reads a history row, applies a licensed probe, reads the induced signature relation, and returns to the local history-sameness comparison;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, ProbeBundle, SigRel, *and the local naming rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverHistoryIdentity}^{\uparrow}}$ *row.*

The carrier is finite and displays every observation that can affect the identity judgment. It has no coordinate for observer-substance, host equality of histories, quotient identity, or a global subject parameter.

Theorem 477.3 (Observer history identity NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObserverHistoryIdentity}^{\uparrow}}$ *surface for the carrier of Definition 477.2 consists exactly of carrier habitation by $O = (H, K, B, S, R, L, C, P, N)$, classifier comparison of the displayed observation rows, stability under the recorded kernel* hsame *rows, ledger exactness for the licensed probes and signature relations, and non-escape from observer-substance, host equality, quotient identity, and global subject parameters.*

Lean4 ✓ `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityPacket namecert obligations`

Proof. Carrier habitation is the displayed finite packet. The two histories, probe ledger, signature ledger, sameness rows, observation ledger, continuation routes, package provenance, and local naming row are all visible before a consumer can read the certificate.

Classifier comparison is restricted to B, S, R, L, C, P, N. A consumer may compare observations only after the relevant probe and signature rows have been displayed, and the resulting comparison returns to the recorded hsame rows between the kernel history entries.

The ledger clauses exhaust the certificate. Since no coordinate stores observer-substance, host equality, quotient identity, or a global subject parameter, none of those objects can be exported by the naming row. □

Theorem 477.4 (Observer history identity classifier stability)**.** *Let $O = (H, K, B, S, R, L, C, P, N)$ and $O' = (H', K', B', S', R', L', C', P', N')$ be accepted* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *carriers related by the displayed classifier. The classifier compares only the observation rows displayed by the* ProbeBundle *and* SigRel *ledgers and the kernel sameness rows recorded in R and R'. Transport may rename these rows through* hsame *and replay them through* Cont, *but it cannot add a host equality comparison or a subject-level identity row.*

Proof. Project both carriers through Definition 477.2. The available comparison data are exactly the two history rows, the probe bundles, the signature relations, the kernel sameness rows, the local observation ledgers, continuation routes, package provenance, and naming rows.

The classifier first reads the displayed probes and signatures. It then checks the corresponding hsame rows in the kernel ledger, with C and C' supplying the permitted read routes.

No further comparison generator is present. Hence stability is componentwise over the displayed observations and kernel sameness rows, with no host equality or subject-level identity added by transport. □

Lean4 ✓ `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityPacket classifier stability`

Theorem 477.5 (Observer history identity ledger exactness)**.** *Every accepted observation used by an* ObserverHistoryIdentity$^{\uparrow}$ *carrier is recorded in the local ledger L before it can affect the identity classifier. The ledger records the source history row, the licensed probe in B, the resulting signature relation in S, the relevant* hsame *row in R, and the* Cont *route in C that exposes the read to the consumer.*

Proof. Begin with an accepted observation. Carrier projection places the observation inside the finite probe ledger B and the corresponding signature ledger S.

The local ledger L then records which history row supplied the observation and which kernel sameness row in R is used by the classifier. The continuation route C records the permitted order of this read.

Thus every classifier-relevant observation is ledgered before it is compared. There is no unrecorded observation channel and no hidden parameter through which an observer-substance or host equality judgment could enter. □

Lean4 ✓ `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityPacket ledger exactness`

Theorem 477.6 (Observer history identity BHist induction coverage)**.** *Every classifier-relevant read of an accepted* ObserverHistoryIdentity$^{\uparrow}$ *carrier $O = (H, K, B, S, R, L, C, P, N)$ is obtained by induction over the displayed finite packet rows: the history rows H, K, then the* ProbeBundle *rows B, the* SigRel *rows S, the* hsame *comparison rows R, the local ledger L, the displayed* Cont *replay C, the provenance row P, and the naming row N.*

Proof. Project the carrier through Definition 477.2. The base read sees only the two finite history rows H, K and therefore has no observer-substance, host equality, quotient identity, or global subject parameter available.

For the observation step, induct over the finite local ledger L together with the displayed ProbeBundle and SigRel rows. Each added ledger entry names its source history row, licensed probe, signature relation, and kernel hsame comparison before the classifier may consume it.

Finally use Definition 477.4 and Definition 477.5. Stability keeps the classifier on the displayed observation and comparison rows, while ledger exactness rules out any unrecorded read channel. The replay, provenance, and naming rows only preserve this route, so the induction exhausts all classifier-relevant reads. □

Theorem 477.7 (Observer history identity non-escape boundary)**.** *The public boundary of an accepted* ObserverHistoryIdentity$^{\uparrow}$ *packet exports only the displayed* BHist *histories, finite* ProbeBundle *observations, finite* SigRel *rows, kernel* hsame *comparisons, local observation ledger,* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no observer-substance, no host equality of histories, no quotient identity, and no global subject parameter.*

Lean4 ✓ `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityPacket non escape boundary`

Proof. Inspect the coordinates of Definition 477.2. They are exactly $H, K, B, S, R, L, C, P, N$, and each coordinate is a finite history, observation, signature, sameness, continuation, package, or naming row.

The classifier-stability theorem keeps all comparisons inside the displayed observation rows and kernel sameness rows. The ledger-exactness theorem ensures that every observation affecting the classifier has already passed through the local observation ledger.

Therefore the boundary has no escape route beyond the finite packet. Since observer-substance, host equality, quotient identity, and global subject parameters are absent from the carrier, the public certificate cannot export them. □

Theorem 477.8 (Observer history identity bridge boundary)**.** *The public* ObserverHistoryIdentity$^{\uparrow}$ *packet bridges observed* BHist *comparison to the acceptance schema through exactly the displayed* ProbeBundle *rows,* SigRel *rows,* hsame *rows,* Cont *rows,* Pkg *provenance rows, and local* NameCert *rows. The bridge exports no observer-substance, host history equality, quotient identity, or global subject parameter.*

Lean4 ✓ `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityUp StdBridge`

Proof. Start from the carrier of Definition 477.2. The only observable comparison data are the two displayed history rows together with the probe, signature, sameness, continuation, package, and naming rows.

Use Definition 477.3. It binds the carrier, classifier, stability, ledger, and non-escape clauses to the same finite local NameCert packet.

Apply Definition 477.4 and Definition 477.5. These keep every comparison inside the displayed probe and signature rows and require the local ledger before a comparison can be consumed by the acceptance schema.

Finally apply Definition 477.7. Since observer-substance, host history equality, quotient identity, and global subject parameters have no carrier coordinate, the bridge boundary cannot export them. □

Closure status for NameCert$_{\text{ObserverHistoryIdentity}^{\uparrow}}$.

Constructive story: An ObserverHistoryIdentity$^{\uparrow}$ packet is generated from two displayed BHist histories, finite ProbeBundle observations, finite SigRel rows, kernel hsame comparisons, local observation-ledger rows, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObserverHistoryIdentity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 477.3 and Definition 477.8 bind the carrier, classifier stability, ledger exactness, non-escape obligations, and bridge boundary to the displayed finite observer-history packet.

Formal verification: FormalStatus(NameCert$_{\text{ObserverHistoryIdentity}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ObserverHistoryIdentityUp.ObserverHistoryIdentityPacket\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close observer-substance, host equality of histories, quotient identity, global subject parameters, or bridge checking.

Upgrade path: No further upgrade – chapter at top of the formal bridge axis.

Chapter 478

A Concrete Naming Certificate for $\mathsf{CrossHistCausal}^{\uparrow}$

The $\mathsf{CrossHistCausal}^{\uparrow}$ interface records the first two-history causal-dependence packet in the inter-Hist chain of Definition 9419.25. Its purpose is local: a consumer may inspect two displayed history rows, the unary slot ledger, the finite dependence witness row, componentwise stability data, read routes, package provenance, and the local naming row, but it receives no global frame, no synchronized clock, and no ambient geometry.

The two-history carrier consumes the finite multi-history configuration surface of chapter 474 and supplies the causal-route rows later bounded by chapter 594. MultiHistConfig provides the finite independent Hist slots from which the ordered pair is selected; $\mathsf{CrossHistCausal}^{\uparrow}$ refines that ordered pair with a slot ledger, dependence witness, hsame and psame stability rows, Cont routes, Pkg provenance, and local NameCert rows.

This local row is the two-history dependency surface later consumed by chapter 876. In that route, chapter 479 supplies the observer slots and chapter 469 supplies the finite replay surface, while $\mathsf{CrossHistCausal}^{\uparrow}$ supplies the dependence witness row R and unary slot ledger K that the causal-rate packet reads before imposing its maximum-rate boundary.

Definition 478.1 (CrossHistCausal carrier)**.** *A* $\mathsf{CrossHistCausal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (A, B, K, R, H, Q, C, P, N)$$

with the following visible rows:

1. *A and B are the two displayed* Hist *rows whose relation is being named;*
2. *K is the unary slot ledger marking the finite time slot at which the dependence read is evaluated;*
3. *R is the finite dependence witness row, recording the displayed source-side and target-side observations used by the cross-Hist read;*
4. *H records componentwise* hsame *comparisons for A, B, K, R, the probe-stability row, the continuation row, the package row, and the local naming row;*
5. *Q records* psame *probe-stability at both anchors, so the same dependence witness remains visible after probe-equivalent source and target reads;*
6. *C records the displayed* Cont *read routes from A and B into K, R, H, Q, P, and N;*
7. *P is the package provenance naming* CrossHistCausal*,* Hist*,* Unary*,* BHist*,* hsame*,* psame*,* Cont*,* Pkg*, and* NameCert*;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausal}^{\uparrow}}$ *row.*

The carrier is a finite two-history naming certificate for cross-Hist causal dependence. It contains no global observer frame, no total ordering of all histories, no metric structure, no maximum-rate function, and no selected synchronization map between A and B.

Lean4 (definition) `BEDC.Derived.InterHist.CrossHistCausal`

Theorem 478.2 (CrossHistCausal NameCert obligations)**.** *For the carrier of Definition 478.1, the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausal}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $X = (A, B, K, R, H, Q, C, P, N)$;*
2. *slot locality, which requires every causal-dependence read to pass through the unary slot ledger K;*
3. *witness exactness, which requires the finite dependence row R to display the source and target observations consumed by the read;*
4. *transport stability, which preserves the carrier under the componentwise* hsame *row H, the two-anchor* psame *row Q, and the displayed* Cont *routes in C;*
5. *non-escape, which excludes global frames, synchronized clocks, metric data, maximum-rate data, and ambient geometry.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalUp.CrossHistCausalUp taste gate boundary`

Proof. Carrier admission is the projection of the finite tuple in Definition 478.1. The rows $A, B, K, R, H, Q, C, P, N$ display the two histories, slot ledger, dependence witness, stability rows, read routes, package provenance, and local naming row.

Slot locality follows because the only slot coordinate is K. Every dependence read named by the carrier reaches the witness row through that unary ledger and cannot substitute a hidden global clock.

Witness exactness is read from R together with the two anchors A and B. The source-side and target-side observations used by the causal read are rows of the same finite packet, so the dependence witness is not imported from an ambient multi-Hist object.

Transport stability is componentwise. The hsame row H transports the visible carrier rows, Q records psame probe-stability at both anchors, and C replays the displayed Cont routes after transport.

The non-escape clause follows from the carrier inventory. Since the packet has no coordinate for a global frame, synchronized clock, metric tensor, maximum-rate function, or ambient geometry, no local $\mathsf{NameCert}_{\mathsf{CrossHistCausal}^{\uparrow}}$ projection can export one. □

Theorem 478.3 (CrossHistCausal hsame and psame stability)**.** *Let $X = (A, B, K, R, H, Q, C, P, N)$ be an accepted* $\mathsf{CrossHistCausal}^{\uparrow}$ *carrier. Any transported read preserving the componentwise* hsame *row H and the two-anchor* psame *row Q preserves the same finite dependence witness R at the same unary slot ledger K, with all consumer access replayed through the displayed* Cont *routes in C.*

Proof. Project the stability coordinates H and Q from the carrier. The row H compares the two history anchors, the unary slot ledger, the dependence witness, the route row, the package row, and the local naming row componentwise.

The row Q records probe-stability at both anchors. Thus a transported source read at A and a transported target read at B still inspect the same displayed dependence witness row rather than a witness chosen outside the packet.

Finally replay the routes in C. Because every consumer path to K and R is one of the displayed continuations, the transported read preserves the same slot-local dependence packet and adds no global synchronization or metric coordinate. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalUp.CrossHistCausalUp slot witness route transport`

Theorem 478.4 (CrossHistCausal slot locality)**.** *For an accepted* $\mathsf{CrossHistCausal}^{\uparrow}$ *carrier*

$$X = (A, B, K, R, H, Q, C, P, N),$$

every causal-dependence read of R is local to the displayed unary slot ledger K. The read may use the componentwise hsame *row H, the two-anchor* psame *row Q, and the displayed* Cont *routes in C, but it may not introduce a global frame, synchronized clock, metric datum, or ambient geometry.*

Proof. Apply Definition 478.2. Its slot-locality clause states that every causal-dependence read passes through K.

Apply Definition 478.3. Transported reads preserve the same finite witness R at the same unary slot ledger after using H, Q, and C.

The checked TasteGate marker attached to Definition 478.2 supports this boundary as an admitted taste-gate surface. Since the carrier rows are only $A, B, K, R, H, Q, C, P, N$, there is no projection target for global frames, synchronized clocks, metric data, or ambient geometry. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalUp.CrossHistCausalUp slot locality`

Theorem 478.5 (CrossHistCausal witness exactness)**.** *For an accepted* $\mathsf{CrossHistCausal}^{\uparrow}$ *carrier $X = (A, B, K, R, H, Q, C, P, N)$, the dependence witness consumed by any cross-Hist read is exactly the displayed row R together with the two history anchors A, B and the slot ledger K. Componentwise* hsame *and two-anchor* psame *transport preserve that same witness; they do not replace it by a witness chosen from a multi-Hist frame, an external rate structure, or an ambient geometric model.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalUp.CrossHistCausalUp witness exactness`

Proof. Use the witness-exactness clause of Definition 478.2. It requires the finite dependence row R to display the source-side and target-side observations consumed by the read.

Use Definition 478.3 for transport. The hsame row preserves the carrier coordinates componentwise, while the psame row preserves probe-stability at both anchors.

The checked TasteGate marker at the obligation theorem records that this finite boundary is the supported local surface. Because X contains no multi-Hist frame, rate row, synchronized clock, metric row, or ambient geometry coordinate, the witness remains exactly R at A, B, K. □

Theorem 478.6 (CrossHistCausal non-escape)**.** *Every consumer of an accepted* $\mathsf{CrossHistCausal}^{\uparrow}$ *carrier factors through the finite rows*

$$A, B, K, R, H, Q, C, P, N.$$

In particular, the consumer may read the two history anchors, unary slot ledger, finite dependence witness, componentwise hsame *transport, two-anchor* psame *stability, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CrossHistCausal}^{\uparrow}}$ *row, and nothing outside those rows.*

Proof. Project the accepted carrier and apply Definition 478.2. The non-escape clause there excludes global frames, synchronized clocks, metric data, maximum-rate data, and ambient geometry.

Apply Definition 478.3. Any transported consumer read is still routed through H, Q, C and preserves the same slot-local dependence witness.

The checked TasteGate marker associated with the obligation theorem supplies the recognized gate boundary for promoting this carrier past a mere seed reading. Since every visible consumer row is one of $A, B, K, R, H, Q, C, P, N$, no consumer can extract data outside the finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CrossHistCausal}\uparrow}$.

Constructive story: A $\mathsf{CrossHistCausal}^{\uparrow}$ packet is generated from two displayed Hist rows, a unary slot ledger, a finite dependence witness row, componentwise hsame transport, two-anchor psame probe-stability, displayed Cont routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.CrossHistCausalUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CrossHistCausal}\uparrow}$, maturePackageClosed)

Scope closed: The exported public two-history NameCert surface binds the finite carrier, slot-local witness, hsame and psame stability, Cont routes, Pkg provenance, local NameCert row, and non-escape theorem for downstream CrossHistCausal consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CrossHistCausal}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close multi-Hist configurations, maximum causal rates, Lorentz geometry, curved spacetime, quantum-state certificates, or a checked Lean theorem for the carrier obligations.

Upgrade path: No further paper closure grade remains; Lean verification and bridge checking remain pending.

Chapter 479

A Concrete Naming Certificate for $\mathsf{ObserverHistIdentity}^{\uparrow}$

The $\mathsf{ObserverHistIdentity}^{\uparrow}$ interface packages the finite observation surface behind observer-history identity. It names the point at which Hist rows, hsame transport, Cont routes, probe and signature observations, no-subject ledgers, package provenance, and a local naming row form one BEDC-native certificate without adding an observer-substance coordinate.

The no-subject side of this carrier is supplied by chapter 615, whose finite observation boundary is consumed before the H, O, L, C, P, N rows are read as an identity packet.

Definition 479.1 (Observer-Hist identity carrier)**.** *An* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, O, L, C, P, N)$$

with the following displayed coordinates:

1. *H is the observed* Hist *pair, or a unary* Hist *row when the consumer inspects one observer-history trace at a time;*
2. *O lists the finite mark observations, displayed* Cont *observations,* ProbeBundle *reads,* SigRel *reads, and classifier observations available to the consumer;*
3. *L stores the componentwise* hsame *transports for the observed* Hist *rows, observation rows, continuation routes, package provenance, and local name;*
4. *C stores the displayed* Cont *routes by which observations are read from the packet;*
5. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*, and* NameCert*;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverHistIdentity}^{\uparrow}}$ *row.*

The classifier compares observer-history packets only by their displayed history rows, transported observations, routes, provenance, and local name. The carrier has no coordinate for an internal truth predicate, observer-substance type, quotient identity, or host equality of observers.

Theorem 479.2 (Observer-Hist identity NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObserverHistIdentity}^{\uparrow}}$ *surface for the carrier of Definition 479.1 consists exactly of five obligations:*

1. *carrier admission for the finite packet $Q = (H, O, L, C, P, N)$;*
2. *classifier transport by componentwise* hsame *over the displayed* Hist *and observation rows;*
3. *observation-ledger exactness for mark,* Cont*,* ProbeBundle*,* SigRel*, and classifier reads;*

4. *no-subject non-escape, asserting that no consumer route introduces a subject coordinate outside* Hist*;*
5. *consumer-surface factorization through* H, O, L, C, P, N*.*

Lean4 ✓ `BEDC.Derived.ObserverHistIdentityUp.ObserverHistIdentityTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 479.1. The finite packet already displays the history row, observation ledger, transport row, continuation row, provenance row, and local naming row.

Classifier transport is componentwise. The row L moves the displayed Hist observations to the common hsame surface, and the row C supplies the only permitted observation routes.

Observation exactness and no-subject non-escape follow by inventory of the packet. Since no coordinate stores an internal truth predicate, observer-substance type, quotient identity, or host equality of observers, every consumer read is exhausted by the five displayed obligation rows. □

Theorem 479.3 (Observer-Hist identity transport stability)**.** *Let* $Q = (H, O, L, C, P, N)$ *be an accepted* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier. If each displayed history and observation coordinate is transported componentwise by the* hsame *rows stored in* L *and then read along the* Cont *routes stored in* C*, the resulting packet is accepted by the same* $\mathsf{NameCert}_{\mathsf{ObserverHistIdentity}^{\uparrow}}$ *surface and preserves the displayed observation ledger, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.ObserverHistIdentityUp.ObserverHistoryIdentityPacket transport stability`

Proof. Project the accepted packet to its six displayed rows. The movable coordinates are exactly the Hist rows, observation rows, continuation routes, provenance, and local name.

Use L to transport each displayed coordinate componentwise to the common hsame surface. The routes in C then read the transported mark, continuation, probe, signature, and classifier observations.

No extra coordinate is available after transport. Thus the transported packet is accepted by the same NameCert surface and preserves the displayed ledger, provenance, and local naming data. □

Theorem 479.4 (Observer-Hist identity ledger exactness)**.** *Every observation exported by an accepted* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier is one of the displayed rows in* O *and is read through* L, C, P, N*. No exported observation can be replaced by an internal truth predicate, quotient observer identity, host equality of observers, or subject parameter outside* Hist*.*

Proof. Start with an exported observation. By carrier admission, the observation must be listed in O as a mark, continuation, probe-bundle, signature-relation, or classifier read.

The transport row L and continuation row C account for movement and readback of that listed observation. The rows P and N record provenance and local naming only.

The carrier inventory contains no truth, quotient, host-equality, or subject coordinate. Hence every exported observation is exactly a displayed ledger entry read through the finite certificate rows. □

Theorem 479.5 (Observer-Hist identity no-subject boundary)**.** *Every consumer of an accepted* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier may inspect observer-history identity only through* Hist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* Pkg*, and* NameCert *rows displayed in* H, O, L, C, P, N*. The consumer obtains no observer-substance type, no host equality of observers, and no quotient identity outside those rows.*

Proof. Let a consumer route be given. Project the carrier to H, O, L, C, P, N and read the displayed observation ledger.

All observer-facing data are Hist rows and their licensed observations. The hsame row supplies identity transport, while Cont, ProbeBundle, and SigRel supply the finite observation routes.

Since P and N only record package provenance and local naming, there is no remaining coordinate from which a separate observer-substance, host equality, or quotient identity could be read. □

Theorem 479.6 (Observer-Hist identity consumer surface)**.** *Every consumer route out of an accepted* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier factors through the ordered rows*

$$H, O, L, C, P, N.$$

In particular, the consumer first reads the displayed Hist *and observation rows, then uses only componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ObserverHistIdentity}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ObserverHistIdentityUp.ObserverHistIdentityPacket consumer surface`

Proof. Project the accepted carrier to the ordered tuple of Definition 479.1. The tuple lists every row that can be inspected by a consumer.

The observation-ledger exactness theorem Definition 479.4 forces all exported observations through O, L, C, P, N. The no-subject boundary theorem Definition 479.5 excludes any route through a subject coordinate outside Hist.

Therefore every consumer route factors through H, O, L, C, P, N, and the finite packet is the whole public surface of the certificate. □

Theorem 479.7 (Observer-Hist identity scoped consumer interface)**.** *Any downstream observer-facing consumer of an accepted* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *carrier must read observer identity through the displayed packet*

$$H, O, L, C, P, N,$$

through the no-subject boundary supplied by chapter 615, and through the kernel rows BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* Pkg*, and* NameCert*. No such consumer may introduce observer-substance, host equality, quotient identity, or an internal truth predicate.*

Proof. Project the accepted carrier to H, O, L, C, P, N. The carrier definition lists the observed history rows, observation ledger, componentwise transports, continuation routes, provenance, and local name as the whole packet.

Apply ledger exactness and the consumer surface theorem. Observation reads factor through O, L, C, P, N, and every downstream route factors through the ordered packet H, O, L, C, P, N.

Apply the no-subject boundary and the sibling no-subject source cited in the introduction. Together they rule out a subject coordinate outside Hist and force observer-facing identity reads to remain on the displayed BHist, hsame, Cont, ProbeBundle, SigRel, Pkg, and NameCert rows.

The packet therefore has no coordinate from which observer-substance, host equality, quotient identity, or an internal truth predicate could be introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverHistIdentity}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObserverHistIdentity}^{\uparrow}$ packet is generated from displayed Hist rows, mark and continuation observations, ProbeBundle and SigRel reads, classifier observations, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObserverHistIdentity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 479.1, Definition 479.2, Definition 479.3, Definition 479.4, Definition 479.5, and Definition 479.6 with

Lean4 ✓ `BEDC.Derived.ObserverHistIdentityUp.taste gate`

bind the carrier admission, classifier transport, observation-ledger exactness, no-subject boundary, and consumer factorization rows, and the checked taste-gate consumer surface supplies the bridge from the concrete observer-history packet to the public ObserverHistIdentity$^{\uparrow}$ interface.

Formal verification: FormalStatus(NameCert$_{\text{ObserverHistIdentity}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ObserverHistIdentityUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close an internal truth predicate, observer-substance type, quotient identity, host equality of observers, or a public standard bridge.

Upgrade path: No further paper-axis upgrade remains; standard bridge checking remains a formal-axis obligation over the same observer-history consumer surface.

Chapter 480

A Concrete Naming Certificate for $\mathsf{KernelCategory}^{\uparrow}$

The $\mathsf{KernelCategory}^{\uparrow}$ packet is the finite naming-certificate surface for the unary-continuation categorical reading of BHist. It presents objects, displayed homs, identities, composition routes, endpoint transport, package provenance, and local naming as rows of the BEDC kernel rather than as a host category object.

The packet consumes the unary-continuation category surface of chapter 471 and refines the finite-kernel category packet of chapter 484. Its hom row specializes those sibling carriers to $\mathsf{CategoryHomCarrier}$ over BHist and Cont, so kernel-category reads inherit the displayed object, hom, identity, composition, and endpoint-transport rows before the local naming certificate is applied.

Definition 480.1 (KernelCategory NameCert carrier)**.** *A* $\mathsf{KernelCategory}^{\uparrow}$ *NameCert carrier is a finite* BHist *packet*

$$K = (O, H, I, C, T, P, N)$$

with the following displayed rows:

1. *O is the object row, accepting exactly unary* BHist *objects;*
2. *H is the displayed hom row, where an arrow from a to b with tail f is accepted by* $\mathsf{CategoryHomCarrier}(a, b, f)$*;*
3. *I is the identity row, storing the empty-tail continuation route for each accepted object;*
4. *C is the composition route row, storing displayed* Cont *routes that compose accepted hom tails and return the resulting* $\mathsf{CategoryHomCarrier}$ *row;*
5. *T is the endpoint* hsame *transport row for objects and displayed hom tails;*
6. *P is the* Pkg *provenance row over* BHist*,* Cont*,* hsame*,* $\mathsf{CategoryHomCarrier}$*, and* $\mathsf{Cat}^{\uparrow}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *row naming the carrier, classifier, identity, composition, transport, and non-escape clauses.*

Thus the carrier is a kernel-facing categorical packet: all accepted objects, arrows, equalities, identities, composites, and dependency names are visible BEDC rows.

Theorem 480.2 (KernelCategory NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *obligation surface for the carrier of Definition 480.1 consists of five rows:*

1. *carrier admission by the finite packet $K = (O, H, I, C, T, P, N)$;*

2. *hom classifier transport through componentwise* hsame *on endpoints and displayed hom tails;*
3. *identity route stability through the empty-tail continuation row stored in I;*
4. *composition route stability through the displayed* Cont *route stored in C and the resulting* CategoryHomCarrier *row;*
5. *non-escape from host category equality, external standard-category semantics, morphism types outside* BHist*, and hidden categorical structure not named by P and N.*

Proof. Carrier admission is immediate from the coordinates of Definition 480.1. The object, hom, identity, composition, transport, package, and local naming rows are all finite BHist data.

For hom classifier transport, unpack the displayed hom row CategoryHomCarrier(a, b, f). Endpoint and hom-tail comparison is componentwise in hsame, so transported endpoints and transported tails remain inside the same carrier coordinates.

Identity stability reads the row I. The empty-tail continuation route is stored only for accepted object rows, and its transported form is again read through T and the same local naming ledger.

Composition stability reads the row C. Given accepted displayed hom rows and a displayed Cont composition route, the resulting composite is repacked as the carrier's displayed CategoryHomCarrier row, with provenance kept in P and N.

The non-escape clause follows by inspection. No coordinate of K is a host category equality, an external standard-category semantic object, or a hidden morphism type, so every public read remains within the named BEDC rows. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier certificate surface`

Theorem 480.3 (KernelCategory Cont-hom exactness)**.** *Every public hom read of an accepted* KernelCategory$^{\uparrow}$ *carrier factors exactly through the displayed continuation route:*

$$O \longmapsto H \longmapsto C \longmapsto T \longmapsto P \longmapsto N.$$

For endpoints a, b and displayed tail f, the public hom classifier is the row CategoryHomCarrier(a, b, f) *together with its* Cont *route and endpoint* hsame *transport. Conversely, any identity or composite route accepted by I or C returns to a displayed hom row in H. The exactness surface exports no host category equality, external associativity theorem, or standard-category morphism object outside the kernel packet.*

Proof. Start with a public hom read. By the carrier definition, the read must first select accepted object endpoints from O and then a displayed hom row from H, whose classifier is CategoryHomCarrier(a, b, f).

Next read the continuation route. Identity reads pass through I, while binary composite reads pass through C and the displayed Cont route for the two hom tails. Endpoint and tail movement is then transported only through T.

It remains to check readback. Each identity or composite route named by I or C is stored precisely so that it returns to the displayed hom row H, with P and N recording the dependency and local naming provenance.

Since the route never leaves these coordinates, there is no place to export host category equality, external standard-category semantics, or a morphism object not already represented by a BHist history. □

Chapter 481

A Concrete Naming Certificate for $\mathsf{NameCertMorphism}^{\uparrow}$

The $\mathsf{NameCertMorphism}^{\uparrow}$ interface packages compiler correctness as a finite relation between two naming certificates. It keeps the source and target certificates visible, records the graph edges that move source histories to target histories, and reads correctness only as classifier and hsame preservation through displayed BEDC rows. It supplies the certificate-preserving graph relation consumed by $\mathsf{CertificateCompiler}^{\uparrow}$ in chapter 473: $\mathsf{CertificateCompiler}^{\uparrow}$ packages executable compiler edges, while $\mathsf{NameCertMorphism}^{\uparrow}$ packages the source-target certificate preservation surface those edges must respect.

Definition 481.1 (NameCert morphism carrier)**.** *A* $\mathsf{NameCertMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, G, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the source* $\mathsf{NameCert}$ *row, including its carrier, classifier, and local continuation surface;*
2. *T is the target* $\mathsf{NameCert}$ *row, including its carrier, classifier, and local continuation surface;*
3. *G is the compiler graph row over source and target carrier histories;*
4. *H stores componentwise* hsame *preservation for source-classified pairs carried by graph edges into the target certificate;*
5. *C stores displayed* Cont *routes witnessing graph lifting from source classifier membership and target-carrier transport for graph outputs;*
6. *P records* Pkg *provenance for* BHist, $\mathsf{NameCert}$, hsame, Cont, *source classifier membership, target classifier membership, and the graph row;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{NameCertMorphism}^{\uparrow}}$ *row.*

The carrier contains no host function interpreting the graph, no quotient equality between source and target histories, and no unrecorded semantic relation outside the displayed graph and certificate rows.

Definition 481.2 (NameCert morphism classifier)**.** *The classifier of a* $\mathsf{NameCertMorphism}^{\uparrow}$ *packet* $M = (S, T, G, H, C, P, N)$ *accepts exactly the graph rows whose visible obligations are witnessed inside the same packet:*

1. *carrier admission for the source row S, target row T, graph row G, preservation row H, route row C, provenance row P, and local name N;*
2. *classifier preservation, sending every source-classified graph input to a target-classified graph output;*
3. *graph lifting along source classifier membership, so every admitted source input has a displayed graph edge into the target carrier;*
4. *target carrier transport, so every graph output is read through the target certificate rather than through an external target object;*
5. *ledger non-escape from host functions, quotient equality, and compiler semantics not recorded in G, H, C, P, N.*

Thus the classifier is finite and local: a consumer can ask only whether the displayed graph preserves the displayed certificate structure.

Theorem 481.3 (NameCert morphism identity obligation)**.** *For every naming certificate row S, the diagonal graph on the carrier of S determines a* NameCertMorphism$^{\uparrow}$ *carrier from S to S. Its classifier preservation, graph lifting, target carrier transport, and* hsame *preservation are the identity rows already present in S.*

Lean4 ✓ `BEDC.Derived.NameCertMorphismUp.taste gate`

Proof. Build the carrier with $S = T$ and let G be the displayed diagonal graph. Carrier admission is projection from the single certificate row and from the local graph coordinate.

The graph-lifting and target-transport clauses use the same displayed carrier membership on both sides. A source-classified input is already a target-classified input because source and target rows coincide.

The hsame clause is the identity transport in the certificate row. No host function or quotient equality is introduced, since every output history is the same displayed history read through the same local NameCert surface. □

Theorem 481.4 (NameCert morphism composition obligation)**.** *Given accepted* NameCertMorphism$^{\uparrow}$ *carriers from S to T and from T to U, their relational graph composite is an accepted* NameCertMorphism$^{\uparrow}$ *carrier from S to U. The composed carrier preserves source classifier membership, target carrier transport, and* hsame *by concatenating the displayed graph-lifting and continuation rows.*

Lean4 ✓ `BEDC.Derived.NameCertMorphismUp.NameCertMorphismUp composition obligation`

Proof. Project the two carriers to their graph rows, preservation rows, route rows, provenance rows, and local names. The shared middle certificate is the target row of the first carrier and the source row of the second carrier.

For a source-classified input, first use the first graph-lifting row to obtain a middle graph edge. Then use the second graph-lifting row on the middle classified output to obtain a final graph edge into U.

Target carrier transport is the concatenation of the first transport into T and the second transport into U. Componentwise hsame preservation composes through the two stored preservation rows, and the composed packet still exposes only the rows displayed in its carrier. □

Theorem 481.5 (Compiler correctness as NameCert morphism bridge)**.** *A compiler graph between source and target naming certificates is correct exactly when it is an accepted* $\mathsf{NameCertMorphism}^{\uparrow}$ *carrier: it preserves source classifier membership, lifts graph edges into the target carrier, transports target rows through the target certificate, and preserves* hsame *for source-classified pairs. In this certificate, compiler correctness is read as* hsame *preservation and not as a host-level semantic function.*

Proof. Assume first that the compiler graph is accepted by the $\mathsf{NameCertMorphism}^{\uparrow}$ classifier. The classifier rows give carrier admission, source-to-target classifier preservation, graph lifting, target carrier transport, and the displayed hsame preservation row. These are exactly the correctness obligations visible to BEDC.

Conversely, assume the graph satisfies those correctness obligations using only rows in the source certificate, target certificate, graph, preservation ledger, route ledger, and package provenance. Package them as S, T, G, H, C, P, N and apply the classifier definition.

The two readings coincide because neither direction has access to a host function, quotient equality, or semantic relation outside the displayed certificate rows. Correctness is therefore the preservation of the named classifier and hsame surfaces by the finite graph packet. □

Theorem 481.6 (NameCert morphism graph-lifting obligation)**.** *Let* $M = (S, T, G, H, C, P, N)$ *be an accepted* $\mathsf{NameCertMorphism}^{\uparrow}$ *packet. For every source-classified input history displayed by* S*, the packet exposes a graph edge in* G *from that source history to a target-carrier history, and the exposure is witnessed by the route ledger* C *with provenance in* P *and local naming in* N*.*

Proof. Unpack classifier acceptance for M. The graph-lifting clause is one of the finite visible obligations of the classifier, so a source-classified input cannot be consumed directly by an external compiler operation.

The witness is read from the displayed route row C. Its source endpoint is the classifier membership row in S, its target endpoint is a carrier row admitted by T, and its edge coordinate is the displayed graph row G.

The package provenance P and local name N record that the witness is part of the same finite packet. No additional carrier, semantic map, or host-level graph interpreter is introduced. □

Theorem 481.7 (NameCert morphism target-transport obligation)**.** *Let* $M = (S, T, G, H, C, P, N)$ *be an accepted* $\mathsf{NameCertMorphism}^{\uparrow}$ *packet. Every output history reached by a graph edge in* G *is consumed through the target* NameCert *row* T*, using the stored* hsame *row* H *and the stored* Cont *route row* C*, rather than through any target object external to the packet.*

Proof. Project the accepted packet to its graph row and target certificate row. Classifier acceptance supplies carrier admission for T and the target-transport coordinate in C, so the output endpoint of an admitted graph edge is read as a target-certificate carrier history.

The preservation row H is then applied only after this target reading is available. It records componentwise hsame preservation for pairs whose source side is classified by S and whose target side is already displayed by T.

Thus the target surface used by the morphism is exactly the one stored in the packet. There is no separate target carrier, no implicit semantic codomain, and no hidden continuation object outside T, H, C, P, N. □

Theorem 481.8 (NameCert morphism ledger non-escape obligation)**.** *Let* $M = (S, T, G, H, C, P, N)$ *be an accepted* $\mathsf{NameCertMorphism}^{\uparrow}$ *packet. The correctness ledger of* M *is exhausted by the displayed source certificate* S*, target certificate* T*, graph row* G*, preservation row* H*, route row* C*,*

package provenance P, and local naming row N; in particular, acceptance cannot export a host function, quotient equality, or compiler semantics not recorded in those rows.

Proof. Inspect the carrier definition. Its only coordinates are S, T, G, H, C, P, N, and the classifier admits exactly obligations whose witnesses are stored in those coordinates.

Carrier admission, classifier preservation, graph lifting, target transport, and hsame preservation are therefore ledger projections from the same finite packet. Each proof object used by a consumer is a displayed row of the packet or a route between displayed rows.

Since no coordinate has the type of a host function, quotient equality, or semantic compiler interpretation, acceptance has no source from which to export one. The public content of the morphism is exactly the finite NameCert packet and its displayed BEDC rows. □

Theorem 481.9 (NameCert morphism source-target separation)**.** *Let $M = (S, T, G, H, C, P, N)$ be an accepted $\mathsf{NameCertMorphism}^{\uparrow}$ packet. The source certificate S, target certificate T, graph row G, same-preservation row H, route row C, and provenance row P remain displayed carrier coordinates of the accepted packet. Consequently no downstream consumer of M may replace S by T, replace T by S, or identify either endpoint through host equality or an unstated certificate coercion.*

Proof. Read the packet through Definition 481.1. The two endpoint certificates are separate coordinates of the same finite carrier, and the graph, preservation, route, provenance, and local naming rows are stored beside them rather than derived from an ambient equality between endpoints.

Now inspect the classifier in Definition 481.2. Acceptance asks for carrier admission, classifier preservation, graph lifting, target carrier transport, and ledger non-escape on those displayed coordinates. The identity row of Definition 481.3 is the special case $S = T$ inside a displayed diagonal packet; it is not a rule allowing an arbitrary packet to collapse its endpoints.

For non-diagonal transport, Definition 481.6 and Definition 481.7 force each read to pass through G, H, C, P, N and the target certificate row. Finally, Definition 481.8 rules out a hidden host equality or certificate coercion, so source and target separation is exactly the displayed-coordinate separation of the packet. □

Theorem 481.10 (NameCert morphism route-composition scope)**.** *Let $M = (S, T, G, H, C, P, N)$ and $M' = (T, U, G', H', C', P', N')$ be accepted $\mathsf{NameCertMorphism}^{\uparrow}$ packets sharing the middle certificate row T. Their composite route from S to U is read only through the displayed graph rows G, G', same-preservation rows H, H', Cont route rows C, C', Pkg provenance rows P, P', and the local $\mathsf{NameCert}$ rows N, N', with the identity and composition obligations supplying the route spine.*

Proof. Start with the two carrier decompositions given by Definition 481.1. The shared middle row is displayed as the target endpoint of M and the source endpoint of M', so the composite route has exactly the endpoint data needed for Definition 481.4.

The route spine is the finite composition proof: graph lifting from Definition 481.6 supplies the first edge into T, target transport from Definition 481.7 reads that edge through the middle certificate, and the second packet repeats the same displayed read into U. The identity obligation in Definition 481.3 covers the degenerate route at any displayed certificate row.

All preservation data used by the composite route is already named in $H, H', C, C', P, P', N, N'$. The classifier of Definition 481.2 admits no extra route oracle, and Definition 481.8 prevents the composed packet from importing a host function, quotient, or external compiler semantics. □

Theorem 481.11 (NameCert morphism ledger-boundary exactness)**.** *Every downstream read of an accepted $\mathsf{NameCertMorphism}^{\uparrow}$ packet $M = (S, T, G, H, C, P, N)$ factors through the source row*

S, target row T, graph-lifting row G, target-transport row C, same-preservation row H, provenance row P, local naming row N, and the ledger non-escape theorem. No read of M factors through a host function, quotient, external equality oracle, or compiler semantic object outside those displayed rows.

Proof. The carrier definition Definition 481.1 fixes the public boundary: the only available coordinates are S, T, G, H, C, P, N. The classifier definition Definition 481.2 then lists the admissible downstream questions as carrier admission, classifier preservation, graph lifting, target transport, and hsame preservation.

For endpoint reads, use Definition 481.9. For graph and target reads, use Definition 481.6 and Definition 481.7. For route composition, use Definition 481.10 together with Definition 481.3 and Definition 481.4.

The final boundary clause is exactly Definition 481.8: every proof object exposed to a consumer is a displayed row or a route between displayed rows. Therefore the ledger boundary is exact, and there is no remaining entry point for a host function, quotient, or external equality oracle. □

Theorem 481.12 (NameCert morphism source-target package)**.** *Every accepted* NameCertMorphism$^{\uparrow}$ *packet $M = (S, T, G, H, C, P, N)$ exposes exactly its source certificate S, target certificate T, graph row G, preservation row H, route row C, package provenance row P, and local naming row N. Downstream consumers may read source classifier membership, target carrier membership, graph lifting,* hsame *preservation, displayed* Cont *routing, and local* NameCert *provenance only through those rows.*

Proof. Project the accepted packet through Definition 481.1. The displayed coordinates are precisely S, T, G, H, C, P, N, so the endpoint certificates and every route witness are already part of the carrier boundary.

Now use Definition 481.2. Its obligations read carrier admission, classifier preservation, graph lifting, target transport, and ledger non-escape from those same rows.

The boundary exactness theorem Definition 481.11 rules out any additional compiler object. Thus the accepted package is the finite source-target graph packet and not a host-level semantic map. □

Theorem 481.13 (NameCert morphism diagonal identity boundary)**.** *For a diagonal* NameCertMorphism$^{\uparrow}$ *packet over a certificate row S, the identity graph consumes the same certificate row on the source and target sides and exports no host function. Its public read is the single displayed certificate S, the diagonal graph row, identity* hsame *preservation, identity* Cont *routing, package provenance, and local naming.*

Proof. Apply Definition 481.3. The diagonal carrier is built with $S = T$, and the graph row is the displayed diagonal graph over the same certificate carrier.

The source and target classifier reads are therefore two reads of the same certificate row. The preservation and routing data are identity rows already visible in the packet, not an external identity function.

By Definition 481.8, the diagonal packet has no coordinate for a host function, quotient equality, or compiler semantics. Its identity boundary is exactly the reuse of the same displayed certificate row on both sides. □

Theorem 481.14 (NameCert morphism composite route provenance)**.** *Let $M = (S, T, G, H, C, P, N)$ and $M' = (T, U, G', H', C', P', N')$ be accepted* NameCertMorphism$^{\uparrow}$ *packets sharing the middle certificate row T. The relational graph composite from S to U preserves the displayed* hsame, Cont,

Pkg, *and local* NameCert *provenance by concatenating the graph-lifting, target-transport, preservation, and provenance rows of the two packets.*

Proof. First use Definition 481.4. It builds the composite carrier by relational graph composition and transports source classifier membership through the shared middle row T.

Next apply Definition 481.10. The composite route reads only G, G', H, H', C, C', P, P', and the local naming rows N, N', with the shared middle certificate serving as the displayed handoff point.

Finally apply Definition 481.11. The composed packet retains the same kind of finite provenance as each input packet and gains no host function, quotient equality, or external compiler semantic relation. □

Theorem 481.15 (NameCert morphism mature consumer completeness)**.** *Every downstream consumer read of an accepted* NameCertMorphism$^{\uparrow}$ *packet factors through the source row S, target row T, graph row G, preservation row H, route row C, provenance row P, and local name N. The factorization is complete for the public morphism surface: endpoint reads, graph reads, preservation reads, route reads, provenance reads, identity reads, and composite-route reads are all accounted for by the displayed packet.*

Proof. Begin with the carrier and classifier in Definition 481.1 and Definition 481.2. They restrict a consumer to S, T, G, H, C, P, N and to the finite obligations witnessed inside those rows.

Endpoint reads are covered by Definition 481.12 and Definition 481.9. Graph and target reads are covered by Definition 481.6 and Definition 481.7.

Identity and composite-route reads are covered by Definition 481.3, Definition 481.13, Definition 481.4, Definition 481.10, and Definition 481.14. The ledger boundary theorem Definition 481.11 then rules out any remaining consumer route outside the displayed packet. □

Theorem 481.16 (NameCert morphism downstream example boundary)**.** *The identity and composite morphism examples exhaust the public example surface of* NameCertMorphism$^{\uparrow}$*. They expose diagonal packets and relational graph composites through the same carrier, classifier, source-target package, diagonal identity boundary, and composite route provenance rows, and they export no host functions or quotient equality.*

Proof. The identity example is exactly the diagonal case of Definition 481.3 read through Definition 481.13. It uses the carrier and classifier rows of Definition 481.1 and Definition 481.2, so its public data are the displayed source-target packet rows.

The composite example is exactly the relational graph composite of Definition 481.4 read through Definition 481.10 and Definition 481.14. It reuses the source-target package of Definition 481.12 and adds no external graph interpreter.

Finally apply Definition 481.8 and Definition 481.11. Since every public example is one of these displayed identity or composite packets, no downstream example surface exports host functions, quotient equality, or compiler semantics outside the local NameCertMorphism$^{\uparrow}$ rows. □

Closure status for NameCert$_{\text{NameCertMorphism}^{\uparrow}}$**.**

Constructive story: A NameCertMorphism$^{\uparrow}$ packet is generated from source and target NameCert rows, a compiler graph row, componentwise hsame preservation, displayed Cont routes, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\text{NameCertMorphism}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 481.1, Definition 481.2, Definition 481.3, Definition 481.4, Definition 481.5, Definition 481.6, Definition 481.7, Definition 481.8, Definition 481.9, Definition 481.10, Definition 481.11, Definition 481.12, Definition 481.13, Definition 481.14, Definition 481.15, and Definition 481.16 bind the finite classifier-preserving certificate morphism surface with instance BEDC.Derived.NameCertMorphismUp.taste_gate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NameCertMorphism}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.NameCertMorphismUp.NameCertMorphismTasteGate\_single\_carrier\_alignment`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close standard compiler semantics, host functions, quotient equality, or a mathlib bridge.
Upgrade path: The chapter is at the top of the paper closure axis; standard bridge work remains outside this chapter.

Chapter 482

A Concrete Naming Certificate for $\mathsf{ContinuationAutomaton}^{\uparrow}$

The $\mathsf{ContinuationAutomaton}^{\uparrow}$ interface realizes computation as a finite Cont-labelled transition surface. It packages a small-step automaton directly over BHist rows, behaviour classifiers, acceptance rows, package provenance, and a local naming row, without adding a host state space or an external operational semantics.

The automaton transition rows are the step-level input later repacked by chapter 471, while chapter 485 supplies the unit and associativity ledger for the same continuation routes.

Definition 482.1 (ContinuationAutomaton carrier)**.** *A* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, q_0, F, T, B, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the finite list of reachable state rows, each row being a* BHist *entry;*
2. *q_0 is the displayed initial state row and F is the finite accepting-state subledger of Q;*
3. *T is the finite transition table whose rows have the form (q, m, q', c) with $q, q' \in Q$, $m : \mathsf{Mark}$, and c a displayed $\mathsf{Cont}(q, m, q')$ witness;*
4. *B is the behaviour classifier ledger, read only from initial reachability, accepting membership, and the displayed transition rows;*
5. *H stores componentwise* hsame *transports for state rows, transition endpoints, accepting rows, behaviour rows, provenance, and local name;*
6. *C stores the displayed continuation routes by which consumers inspect transition steps and behaviour reads;*
7. *P is the* Pkg *provenance over* BHist, Mark, hsame, Cont, *behaviour classifiers, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationAutomaton}^{\uparrow}}$ *row.*

The classifier accepts behaviour claims only through this finite state, transition, acceptance, transport, continuation, provenance, and naming surface. The carrier has no coordinate for host configurations, an external evaluator, a quotient of traces, or an unlisted transition rule.

Lean4 (definition) `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonUp`

Theorem 482.2 (ContinuationAutomaton step-continuation correspondence)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*, every displayed automaton step* $q \xrightarrow{m} q'$ *is exactly a row of* T *carrying a* $\mathsf{Cont}(q, m, q')$ *witness, and every such displayed* $\mathsf{Cont}(q, m, q')$ *row reconstructs the automaton step from* q *to* q' *labelled by* m*. Consequently the behaviour classifier* B *reads transition behaviour only through* Cont *rows already present in the finite carrier.*

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonTasteGate single carrier alignment`

Proof. For the forward direction, project the carrier of Definition 482.1. A displayed automaton step is not an independent predicate in the packet; it is a row of the finite transition table T. Reading that row exposes endpoints q, q', a mark m, and the stored witness $c : \mathsf{Cont}(q, m, q')$.

For the reverse direction, begin with a displayed transition row (q, m, q', c) in T. The endpoints are members of the finite state ledger Q, and the row itself records that the automaton has a transition from q to q' labelled by m. Thus the same row reconstructs the step.

Finally read the classifier row B. Its behaviour observations are licensed only by initial reachability, accepting membership in F, the transition table T, and the routes in C. Since the only transition evidence in T is the displayed Cont field, the classifier cannot consult a host evaluator, quotient trace, or unlisted transition rule. □

Theorem 482.3 (ContinuationAutomaton NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationAutomaton}^{\uparrow}}$ *surface for the carrier of Definition 482.1 consists exactly of five rows:*

1. *carrier admission for the finite packet* $A = (Q, q_0, F, T, B, H, C, P, N)$*;*
2. *behaviour-classifier admission over the initial row, accepting rows, and* Cont*-labelled transition table;*
3. *stability of state rows, accepting membership, transition endpoints, and behaviour rows under componentwise* hsame *transport;*
4. *exactness of every displayed automaton step as the corresponding* Cont *witness and of every displayed* Cont *transition row as an automaton step;*
5. *ledger rows for transition inspection, accepting inspection, behaviour reads,* Pkg *provenance, and the local naming certificate.*

These obligations mention no host evaluator, external state space, trace quotient, or unlisted transition rule.

Proof. Carrier admission is projection from Definition 482.1. The finite packet already displays its state ledger, initial row, accepting ledger, transition table, behaviour classifier, transport row, continuation routes, provenance, and local name.

Classifier admission reads only the rows q_0, F, T, B. The row B classifies behaviour by inspecting initial reachability, accepting membership, and transition rows already present in the packet.

Stability is componentwise through H, while exactness is Definition 482.2. The remaining ledgers C, P, N route, record provenance, and name the packet. No carrier coordinate can introduce a separate operational semantics, so the five displayed rows exhaust the NameCert surface. □

Theorem 482.4 (ContinuationAutomaton behaviour classifier stability)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*, every behaviour read is determined by the displayed initial row* q_0*, accepting ledger* F*, transition table* T*, classifier row* B*, transports* H*, routes* C*, provenance* P*, and local name* N*. Componentwise* hsame *transport may rename these displayed rows, but it cannot import host configurations, quotient traces, or unlisted transition rules.*

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonCarrier behaviour classifier stability`

Proof. Project the finite carrier. The behaviour classifier B has access only to the initial row, accepting rows, and transition table named by q_0, F, T.

Use the NameCert obligations. Classifier admission and stability require behaviour reads to pass through the displayed transport row H and inspection routes C while keeping the provenance and local naming rows P, N attached.

The step-continuation correspondence supplies the transition exactness boundary. Since no host configuration, quotient trace, or unlisted transition rule is a coordinate of A, classifier stability cannot read one. □

Theorem 482.5 (ContinuationAutomaton acceptance nonescape)**.** *Every accepting read of an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *packet factors through* q_0, F, T, B, H, C, P, N *inside the carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*. Acceptance may inspect the finite state ledger* Q*, but it exports no host run relation, no quotient of traces, and no transition rule outside the displayed table* T*.*

Proof. Begin with the carrier and the accepting ledger F. An accepting read can only ask whether a displayed state row belongs to the finite ledger already stored in the packet.

Apply the NameCert obligations and the step-continuation correspondence. Transition reachability is replayed through rows of T carrying displayed Cont witnesses, and behaviour inspection is routed through B, H, C, P, N.

Use the carrier exclusion clause. A host run relation, quotient trace, or unlisted transition rule is absent from A, so acceptance cannot escape the finite ledger and transition table. □

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonCarrier acceptance nonescape`

Theorem 482.6 (ContinuationAutomaton ledger factorization)**.** *The unary-continuation-category and continuationmonad consumers of an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier receive only the finite packet*

$$Q, q_0, F, T, B, H, C, P, N.$$

Their object, transition, unit, composition, and associativity reads factor through the displayed state ledger, initial and accepting rows, Cont*-labelled transition table, behaviour classifier, transports, routes, provenance, and local naming row.*

Proof. Read the consumer boundary through Definition 482.1. The only exported rows are $Q, q_0, F, T, B, H, C, P, N$

Use the NameCert obligations and behaviour classifier stability. They bind classifier reads, transport stability, transition exactness, and ledger inspection to the same finite rows.

Apply acceptance nonescape and the step-continuation correspondence for the categorical and monadic routes. Object and transition data are rows of Q and T, while unit, composition, and associativity ledgers are repackings through C, H, P, N, not host operational semantics. □

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonCarrier ledger factorization`

Theorem 482.7 (ContinuationAutomaton transition determinacy)**.** *Let* $A = (Q, q_0, F, T, B, H, C, P, N)$ *be an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier. If two displayed transition rows of* T *have the same source state* q*, the same mark* m*, and the same target state* q'*, then their transition readings are identified by the stored* $\mathsf{Cont}(q, m, q')$ *witness and by the corresponding componentwise* hsame *transport row in* H*. Thus determinacy is a finite row equality inside the packet, not a claim about an external evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonCarrier transition determinacy`

Proof. Project the carrier and read both transition rows from the same finite table T. Each row exposes only q, m, q', and a displayed continuation witness.

The checked single-carrier alignment marker attached to Definition 482.2, named `ContinuationAutomatonTast` identifies transition evidence with the carrier field that stores it. Componentwise transport through H therefore compares the two rows by their displayed endpoints and their stored $\mathsf{Cont}(q, m, q')$ witnesses.

No host transition function is available in the carrier. Hence the determinacy statement is exactly the equality of the displayed transition row data under the packet's own hsame transport. □

Theorem 482.8 (ContinuationAutomaton trace exhaustion)**.** *For an accepted* ContinuationAutomaton$^{\uparrow}$ *carrier $A = (Q, q_0, F, T, B, H, C, P, N)$, every accepted finite behaviour trace is a finite list of rows from T replayed through the displayed continuation routes in C and transported through the componentwise rows in H. The trace exports no quotient of traces, host run relation, or unlisted transition rule.*

Proof. Read the behaviour classifier B through the carrier. Its accepted behaviours are licensed only by the initial row q_0, accepting membership in F, and transition rows already present in T.

Replay a finite trace by listing its transition rows in order and composing the corresponding displayed routes through C. At each endpoint comparison, use the componentwise transport rows in H to move between representatives of the same displayed state row.

The single-carrier alignment marker cited above prevents a trace step from being replaced by a host operational step outside T. Therefore the finite list of transition rows, together with C and H, exhausts the accepted trace surface. □

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonTraceExhaustion`

Theorem 482.9 (ContinuationAutomaton NameCert obligation pack)**.** *For an accepted* ContinuationAutomaton$^{\uparrow}$ *carrier, carrier admission, classifier stability, transition exactness, trace exhaustion,* Pkg *provenance, and local naming form the finite obligation package consumed by chapter 471 and chapter 485. These consumers receive only the displayed automaton rows $Q, q_0, F, T, B, H, C, P, N$ and no external state space or evaluator.*

Proof. Carrier admission and classifier stability are the first three rows of Definition 482.3. Transition exactness is Definition 482.2, with the checked single-carrier alignment marker ensuring that the transition row is the stored continuation evidence.

Trace exhaustion is Definition 482.8. It supplies the finite list-of-transition-rows surface that categorical and monadic consumers can replay through C and transport through H.

Finally P records the package provenance and N records the local naming row. Repacking these rows gives unary-continuation-category and continuationmonad their object, hom, unit, composition, and associativity inputs without granting any unlisted operational semantics. □

Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.ContinuationAutomatonCarrier namecert obligation pack`

Theorem 482.10 (ContinuationAutomaton state-source determinacy)**.** *For an accepted* ContinuationAutomaton$^{\uparrow}$ *carrier $A = (Q, q_0, F, T, B, H, C, P, N)$, every accepted state read is one of the finite rows of Q reached from q_0 through displayed transition rows in T, replayed through routes in C, and transported through componentwise rows in H.*

Proof. Project the state source from Definition 482.1. The only state inventory exposed by the carrier is the finite row family Q, together with the initial row q_0.

Use Definition 482.2 and Definition 482.3. They identify every step read with a displayed transition row in T and keep transport inside H, C.

Apply Definition 482.5, Definition 482.8, and Definition 482.9. No host state space or evaluator is exported, so each accepted state read is one of the displayed rows reached from q_0 through the finite packet. □

Theorem 482.11 (ContinuationAutomaton transition route closure)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*, every transition consumer factors through a displayed row* (q, m, q', c) *of* T*, where* q, q' *are state rows of* Q*,* m *is the read mark, and* c *is the stored* $\mathsf{Cont}(q, m, q')$ *witness. The consumer cannot add a host operational rule or a transition relation outside the packet.*

Proof. Read transition evidence from Definition 482.1. A transition row stores only its source, mark, target, and displayed continuation witness.

Apply Definition 482.2 and Definition 482.7. They make the stored $\mathsf{Cont}(q, m, q')$ witness the transition route and compare duplicate reads only by the packet's componentwise transport.

Use Definition 482.5, Definition 482.8, and Definition 482.9. The finite trace and consumer ledgers replay rows of T through C, H, P, N, so no additional operational semantics enters the route. □

Theorem 482.12 (ContinuationAutomaton acceptance ledger exactness)**.** *For an accepted* $\mathsf{ContinuationAutomaton}$ *carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*, all acceptance reads are exhausted by the rows* F, T, B, H, C, P, N*. This is the exact ledger consumed by chapter 471 and chapter 485.*

Proof. Start with Definition 482.3. It places accepted behaviours on the displayed classifier row B, accepting rows F, transition table T, componentwise transport H, replay routes C, provenance P, and local naming N.

Use Definition 482.5 and Definition 482.8. They exclude external operational semantics, host configurations, trace quotients, and unlisted transition rules from the accepted trace surface.

Finally apply Definition 482.9. The categorical and monadic siblings consume exactly the displayed finite ledger, so their object, hom, unit, composition, and associativity reads do not enlarge the acceptance boundary. □

Theorem 482.13 (ContinuationAutomaton obligation-closure consumer pack)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *carrier* $A = (Q, q_0, F, T, B, H, C, P, N)$*, carrier admission, behaviour classifier stability, step-continuation correspondence, trace exhaustion, acceptance ledger exactness, and ledger factorization assemble the obligation-closure package consumed by chapter 471 and chapter 485. Those consumers receive only* $Q, q_0, F, T, B, H, C, P, N$*: no host evaluator, external state space, trace quotient, or unlisted transition rule is exported.*

Proof. Carrier admission is Definition 482.1 together with Definition 482.3. These rows provide the finite state ledger, initial row, accepting ledger, transition table, classifier, transports, routes, provenance, and local name.

The behaviour surface is fixed by Definition 482.4. The step surface is fixed by Definition 482.2, so every transition read is the displayed $\mathsf{Cont}(q, m, q')$ row already stored in T.

Trace and acceptance consumption are exhausted by Definition 482.8 and Definition 482.12. The ledger factorization of Definition 482.6 then repacks the same rows for the unary-continuation-category and continuationmonad consumers.

Since each cited row is a projection or replay of $Q, q_0, F, T, B, H, C, P, N$, the consumer package has no coordinate for a host evaluator, external state space, trace quotient, or transition rule outside T. □

Theorem 482.14 (ContinuationAutomaton public consumer export)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *packet, the public export to* $\mathsf{UnaryContinuationCategory}^{\uparrow}$ *and* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumers is exactly the finite carrier surface*

$$Q, q_0, F, T, B, H, C, P, N.$$

The TasteGate marker preserves the same single-carrier surface: carrier rows, behaviour classifier rows, transition-route rows, acceptance-ledger rows, trace rows, provenance rows, and local naming rows are exported as one packet, with no host evaluator, trace quotient, or unlisted transition rule.

Proof. Begin with Definition 482.13. It assembles carrier admission, behaviour classifier stability, step-continuation correspondence, trace exhaustion, acceptance ledger exactness, and ledger factorization for the two named consumers.

Use Definition 482.12. The acceptance side of the export is exhausted by F, T, B, H, C, P, N and remains attached to the same state ledger Q and initial row q_0.

Finally apply the checked TasteGate marker for $BEDC.Derived.ContinuationAutomatonUp.taste_gate$. It records that the displayed carrier remains the single consumer-visible surface, so the public export is $Q, q_0, F, T, B, H, C, P, N$ and nothing outside the finite packet. □

Theorem 482.15 (ContinuationAutomaton mature consumer completeness)**.** *For every accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *packet, every mature consumer read factors through the finite carrier rows*

$$Q, q_0, F, T, B, H, C, P, N$$

and through the two sibling consumers chapter 471 and chapter 485. No mature consumer obtains a host evaluator, an external state space, a trace quotient, or a transition rule outside the displayed table T.

Proof. Begin with Definition 482.13. It assembles the carrier admission, classifier stability, step-continuation correspondence, trace exhaustion, acceptance ledger exactness, and ledger factorization rows consumed by the categorical and monadic siblings.

Use Definition 482.12 and Definition 482.14. Acceptance reads are exhausted by F, T, B, H, C, P, N over the same state ledger Q and initial row q_0, and the public export gives those rows to $\mathsf{UnaryContinuationCategory}^{\uparrow}$ and $\mathsf{ContinuationMonad}^{\uparrow}$ without adding another operational surface.

Finally apply the TasteGate marker already carried by the closure block. It fixes the single-carrier alignment for the packet, so a mature consumer read must be a replay through the finite rows and the two named sibling consumers. Since none of those rows contains a host evaluator, external state space, trace quotient, or unlisted transition rule, no mature read can acquire one. □

Theorem 482.16 (ContinuationAutomaton ContinuationMonad unit route)**.** *For an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *packet* $A = (Q, q_0, F, T, B, H, C, P, N)$*, the* $\mathsf{ContinuationMonad}^{\uparrow}$ *unit route reads the initial state* q_0*, the identity continuation row at* q_0*, the displayed routes* C*, provenance* P*, and local name* N *from the same finite packet. The route adds no host evaluator, external state space, trace quotient, or unlisted transition rule.*

Proof. Begin with the carrier and step-continuation correspondence, Definition 482.1 and Definition 482.2. The initial state row q_0 and the identity continuation evidence are read as displayed rows of the same finite automaton surface.

Use acceptance ledger exactness and public consumer export, Definition 482.12 and Definition 482.14. They keep the monadic unit read attached to C, P, N over the carrier rows and prevent the route from importing a separate operational semantics.

Finally apply Definition 482.15 together with the checked TasteGate marker $BEDC.Derived.ContinuationAut$ The marker records the single-carrier alignment, so the $\mathsf{ContinuationMonad}^{\uparrow}$ unit route is a repacking of q_0, C, P, N from A and nothing outside it. □

Theorem 482.17 (ContinuationAutomaton unary category object route)**.** *For an accepted* ContinuationAutomaton *packet* $A = (Q, q_0, F, T, B, H, C, P, N)$*, the* $\mathsf{UnaryContinuationCategory}^{\uparrow}$ *object route reads objects only from the finite state ledger* Q*, and reads morphism rows only from transition rows of* T *carrying displayed* Cont *witnesses. The categorical object and morphism surface exports no host state space, quotient trace, external evaluator, or transition rule outside* T*.*

Proof. Read objects from the carrier. The only object inventory available to the categorical consumer is the finite state ledger Q of Definition 482.1.

Read morphisms through the step-continuation correspondence. By Definition 482.2, Definition 482.11, and Definition 482.12, every transition morphism is a row of T carrying a displayed $\mathsf{Cont}(q, m, q')$ witness and is transported only through H, C, P, N.

Finally use Definition 482.14 and Definition 482.15, with the checked single-carrier alignment marker $ContinuationAutomatonTasteGate_single_carrier_alignment$. The categorical route therefore receives Q for objects and T for Cont-labelled morphisms, with no additional operational surface. □

Theorem 482.18 (ContinuationAutomaton consumer boundary readback)**.** *Every consumer shared by the* $\mathsf{ContinuationMonad}^{\uparrow}$ *and* $\mathsf{UnaryContinuationCategory}^{\uparrow}$ *surfaces of an accepted* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *packet reads only*

$$Q, q_0, F, T, B, H, C, P, N.$$

The shared readback exposes no host evaluator, external state space, trace quotient, hidden transition selector, or transition rule outside the displayed Cont*-labelled table* T*.*

Proof. First assemble the shared carrier surface. Apply Definition 482.1, Definition 482.2, and Definition 482.12 to bind state, initial, accepting, transition, behaviour, transport, route, provenance, and naming reads to $Q, q_0, F, T, B, H, C, P, N$.

Next use the public export and mature consumer completeness, Definition 482.14 and Definition 482.15. These theorems say that the monadic and categorical surfaces consume the same finite packet, not two independent copies with separate operational semantics.

Finally apply the two route rows, Definition 482.16 and Definition 482.17, together with the checked TasteGate marker $BEDC.Derived.ContinuationAutomatonUp.taste_gate$. The shared boundary is therefore exactly $Q, q_0, F, T, B, H, C, P, N$, and no hidden selector, trace quotient, host evaluator, or external state space can be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationAutomaton}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuationAutomaton}^{\uparrow}$ packet is generated from finite BHist state rows, an initial row, accepting rows, Mark-labelled transition rows carrying displayed Cont witnesses, behaviour classifier rows, componentwise hsame transports, displayed inspection routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationAutomaton}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 482.1, Definition 482.2, Definition 482.3, Definition 482.10, Definition 482.11, Definition 482.8, Definition 482.12, Definition 482.9, Definition 482.13, Definition 482.14, and the TasteGate marker bind the public consumer export, transition route closure, acceptance ledger exactness, obligation-closure consumer pack, transport rows, package provenance, local naming row, and the finite automaton carrier.

Formal verification: FormalStatus(NameCert$_{\text{ContinuationAutomaton}\uparrow}$, theoremChecked)
Lean target: `BEDC.Derived.ContinuationAutomatonUp.taste\_gate`
Bridge status: none
Origin: ai
Lean4 ✓ `BEDC.Derived.ContinuationAutomatonUp.taste gate`
Not claimed: This chapter does not close external operational semantics, host configurations, trace quotients, deterministic transition selection, or a public standard bridge.
Upgrade path: No further paper-axis grade is above matureClosure; bridgeChecked verification remains a separate formal-axis target.

Chapter 483

A Concrete Naming Certificate for $\mathsf{SubjectReduction}^{\uparrow}$

The $\mathsf{SubjectReduction}^{\uparrow}$ surface isolates subject reduction from the broader type-classifier packet. It records the finite data by which an accepted source membership row, a visible reduction row, and a target membership row form one Ext-Cont closure packet inside BHist, hsame, Pkg, and local NameCert rows.

The packet consumes classifier membership from chapter 489, the derivation-row surface of chapter 598, and the normal-form boundary rows of chapter 562. The source membership row E_0, displayed reduction row R, and target membership row E_1 are therefore ledgered sibling rows repackaged through Ext, Cont, hsame, Pkg, and local NameCert rather than a kernel-only island or freestanding typing semantics.

Definition 483.1 (SubjectReduction carrier)**.** *A* $\mathsf{SubjectReduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, E_0, R, E_1, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the source term-history row whose type membership is being consumed;*
2. *B is the target term-history row whose type membership is being produced;*
3. *E_0 is the source* Ext *membership row for A;*
4. *R is the* Cont *reduction row from A to B through the displayed mark and local read route;*
5. *E_1 is the target* Ext *membership row for B;*
6. *L is the finite subject-reduction ledger tying E_0, R, and E_1 to the same typing packet;*
7. *H records the componentwise* hsame *transports for the source row, target row, source membership row, reduction row, target membership row, ledger row, package row, and local certificate row;*
8. *C records the* Cont *read routes for source membership, typed reduction, target membership, and replay of the closure packet;*
9. *P is the* Pkg *provenance over* BHist, Ext, Cont, hsame, *the ledger L, and the surrounding* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReduction}^{\uparrow}}$ *row.*

The carrier contains no host reduction semantics, no semantic equality of terms, no unrecorded conversion procedure, and no global normalization theorem.

Theorem 483.2 (SubjectReduction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SubjectReduction}^{\uparrow}}$ *surface for Definition 483.1 consists of four obligations:*

1. *carrier admission for the finite packet* $S = (A, B, E_0, R, E_1, L, H, C, P, N)$*;*
2. Ext-Cont *closure, saying that the accepted source membership row* E_0 *and displayed reduction row* R *expose only the target membership row* E_1*;*
3. *ledger exactness, saying that every public subject-reduction read is recorded in* L *with its source membership, reduction, target membership,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows;*
4. *non-escape from host reduction semantics, semantic equality of terms, unrecorded conversion, hidden normalization, or any classifier not named by the carrier.*

Lean4 ✓ `BEDC.Derived.SubjectReductionUp.SubjectReductionTasteGate single carrier alignment`

Proof. Carrier admission is the finite packet displayed in Definition 483.1. Its rows name the source and target histories, the source and target Ext memberships, the reduction route, the local ledger, transport rows, continuation routes, package provenance, and certificate row.

The closure obligation is the only permitted bridge from E_0 and R to E_1. Since C supplies the read routes and H supplies only componentwise hsame transport, a consumer can replay the subject-reduction packet but cannot replace it by an ambient typing theorem.

The ledger and non-escape obligations follow from the same inventory. Every usable read is first recorded in L, and no carrier coordinate names host reduction semantics, semantic equality, conversion outside the packet, or a global normalization result. □

Theorem 483.3 (SubjectReduction Ext-Cont closure)**.** *Let* $S = (A, B, E_0, R, E_1, L, H, C, P, N)$ *be an accepted* $\mathsf{SubjectReduction}^{\uparrow}$ *carrier. Any public read that starts from the source membership row* E_0 *and the displayed reduction row* R *reaches target membership only as the recorded row* E_1*, transported through the displayed* hsame *rows and replayed through the recorded* Cont *routes.*

Proof. Project the accepted carrier. The source membership data are exactly E_0, and the reduction data are exactly R.

Use the read routes in C to replay the closure packet. The only target membership row named by the packet is E_1, and the only permitted renamings are the componentwise hsame transports in H.

Thus the closure read is an internal Ext-Cont packet. It cannot appeal to semantic equality, host reduction, or an unrecorded conversion procedure. □

Theorem 483.4 (SubjectReduction ledger exactness)**.** *Every accepted subject-reduction read exported by a* $\mathsf{SubjectReduction}^{\uparrow}$ *carrier is recorded in the ledger* L *before it is consumed. The ledger records the source row* A*, target row* B*, source membership* E_0*, reduction row* R*, target membership* E_1*, the relevant* hsame *transport rows, the* Cont *read routes, the* Pkg *provenance, and the local* NameCert *row.*

Proof. Begin with a public subject-reduction read. Carrier projection places the source membership in E_0, the reduction in R, and the target membership in E_1.

The ledger L records these rows together with their source and target histories. It also records the hsame and Cont rows needed to transport and replay the read.

Since P and N are the only package and naming rows, every exported read has the stated provenance. There is no remaining channel through which an unledgered subject-reduction assertion can enter the packet. □

Theorem 483.5 (SubjectReduction non-escape)**.** *An accepted* $\mathsf{SubjectReduction}^{\uparrow}$ *packet exports only the displayed* BHist *rows,* Ext *membership rows,* Cont *reduction row, subject-reduction ledger,* hsame *transports,* Pkg *provenance, and local* NameCert *row of Definition 483.1. It exports no host reduction semantics, semantic equality of terms, hidden conversion relation, erased typing derivation, or global normalization theorem.*

Proof. Inspect the carrier coordinates. They are exactly $A, B, E_0, R, E_1, L, H, C, P, N$, and each coordinate is a displayed history, membership row, reduction row, ledger row, transport row, route row, package row, or naming row.

Apply Definition 483.3 for reduction-facing consumers and Definition 483.4 for provenance-facing consumers. Both routes remain inside the finite carrier inventory.

Therefore the public boundary has no coordinate from which host reduction semantics, semantic equality, hidden conversion, erased typing derivations, or global normalization can be read. □

Theorem 483.6 (SubjectReduction bridge packet scope)**.** *The public carrier,* Ext-Cont *closure, ledger exactness, and non-escape rows form the bridge packet available to standard subject-reduction consumers. The packet exports the finite surface* $S = (A, B, E_0, R, E_1, L, H, C, P, N)$ *and still exports no host type checker, semantic term equality, global normalization, or erased derivation semantics.*

Proof. Start from Definition 483.1. It displays the public packet with source and target histories, source and target Ext rows, the Cont reduction row, ledger, transports, continuations, provenance, and local certificate.

Apply Definition 483.3 to obtain the consumer route from E_0 and R to E_1. Then apply Definition 483.4 to record that route in L with its transport, continuation, package, and local naming rows.

The bridge packet boundary is closed by Definition 483.5. Since the carrier has no host type-checker row, semantic equality row, global normalization row, or erased-derivation row, standard subject-reduction consumers receive only the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReduction}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReduction}^{\uparrow}$ packet is generated from source and target BHist rows, source and target Ext membership rows, a displayed Cont reduction row, a finite subject-reduction ledger, componentwise hsame transports, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubjectReduction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 483.1, Definition 483.2, Definition 483.3, Definition 483.4, Definition 483.5, and Definition 483.6 close the public finite carrier, Ext-Cont closure, ledger exactness, non-escape rows, bridge packet scope, and mature consumer package named for subject-reduction consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubjectReduction}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SubjectReductionUp.SubjectReductionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host type checker, semantic equality of terms, global normalization, erased derivation semantics, or a public subject-reduction bridge.

Upgrade path: No further theory-axis closure is pending for this chapter; remaining work, if any, belongs to separate bridge-checking infrastructure.

Chapter 484

A Concrete Naming Certificate for $\mathsf{FiniteKernelCategory}^{\uparrow}$

The $\mathsf{FiniteKernelCategory}^{\uparrow}$ packet records the category-shaped reading of the finite BEDC kernel as a concrete naming certificate. Its objects are finite object histories, its arrows are displayed continuation histories, its identity and composition rows are finite Cont ledgers, and its equality rows are hsame comparisons on the same displayed histories. The packet is native to BHist and does not import an external category library, quotient morphism type, or host equality of arrows.

It is the finite, displayed-history specialization of the broader kernel-category surface in chapter 488, and it also consumes the displayed continuation-composition route of chapter 471. Its object and hom readings come from the kernel-category packet, while the identity, composition, associativity, and unit ledgers repackage those rows through finite BHist, hsame, Cont, Pkg, and local NameCert evidence rather than importing an external category library.

Definition 484.1 (Finite-kernel category carrier)**.** *A* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier is a finite packet*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

with the following visible rows:

1. *O is the finite object-history row: every object is a displayed* BHist *object history accepted by the kernel classifier.*
2. *M is the displayed hom-history row: a morphism from object history a to object history b is a finite history f equipped with the displayed continuation ledger relating the endpoint histories.*
3. *I records identity rows, one for each accepted object history, using the empty continuation ledger from the object to itself.*
4. *Q records composition rows: for displayed hom histories $f : a \to b$ and $g : b \to c$, the composite history is accepted only through the finite* Cont *ledger joining the two displayed continuations.*
5. *A records associativity ledgers comparing the two displayed threefold composites by* hsame *after their* Cont *rows are expanded.*
6. *U records left and right unit ledgers comparing empty-continuation composites with the original displayed hom history.*
7. *H stores the componentwise* hsame *transports for object histories, hom histories, identity rows, composition rows, associativity rows, and unit rows.*

8. *C stores the* Cont *routes by which a consumer reads identities, composites, associativity witnesses, and unit witnesses.*
9. *P is the* Pkg *provenance row naming the consumed* BHist, hsame, Cont, *identity, composition, associativity, and unit ledgers.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteKernelCategory}^{\uparrow}}$ *registration row.*

The local classifier compares carriers only through these displayed rows. It has no coordinate for a global category universe, quotient morphisms, external functor categories, or host equality of arrows.

Lean4 ✓ `BEDC.Derived.FiniteKernelCategoryUp.FiniteKernelCategoryTasteGate single carrier alignment`

Theorem 484.2 (Finite-kernel category NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteKernelCategory}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation by the empty object history and by unary object histories with their empty-continuation identity rows;*
2. *classifier stability for object histories, displayed hom histories, identity ledgers, composition ledgers, associativity ledgers, and unit ledgers by componentwise* hsame*;*
3. *identity exactness, requiring each identity row to be the empty* Cont *continuation attached to the displayed object history;*
4. *composition exactness, requiring every public composite to factor through the displayed* Cont *row named in Q;*
5. *ledger non-escape, requiring all public category reads to pass through* BHist, hsame, Cont, Pkg, *and* NameCert, *excluding external category libraries and quotient morphisms.*

Proof. Carrier habitation uses the empty object history and the unary object histories already accepted by the kernel. The empty continuation gives the identity row, so the object and identity coordinates are inhabited inside the finite carrier.

Classifier stability is componentwise. Transport object histories and hom histories by the displayed hsame rows, then transport the identity, composition, associativity, and unit ledgers along the same endpoint comparisons.

Identity and composition exactness are projections of the carrier. The identity coordinate is I, while each composite is read through the Cont row stored in Q before it can be consumed as a displayed hom history.

The non-escape row follows by inspecting K. Since every public route is stored in H, C, P, and N, a consumer cannot read an external category library, quotient morphism, or host arrow equality from this certificate. □

Lean4 ✓ `BEDC.Derived.FiniteKernelCategoryUp.FiniteKernelCategoryNamecertObligations`

Theorem 484.3 (Finite-kernel category object and hom boundary)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Every accepted object read is exposed only through the object-history row O, and every accepted hom read is exposed only through the hom-history row M. Endpoint compatibility for those hom reads is guarded by the displayed* BHist *histories, the componentwise* hsame *rows in H, and the* Cont *rows in C.*

Proof. Project the carrier $K = (O, M, I, Q, A, U, H, C, P, N)$. The carrier definition makes O the only object-history coordinate and M the only hom-history coordinate.

Apply Definition 484.2. Carrier exposure and classifier stability transport object histories and hom histories by the displayed hsame rows, while composition exactness and ledger non-escape keep every endpoint read inside the visible BHist and Cont ledgers.

Finally repack the result as a boundary statement for consumers. Since no quotient morphism type or host arrow equality occurs in K, object and hom reads cannot escape the displayed O, M, H, C, P, N surface. □

Theorem 484.4 (Finite-kernel category identity and composition boundary)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted FiniteKernelCategory$^{\uparrow}$ *carrier. Identity reads are exactly the rows I, and composition reads are exactly the rows Q. Associativity and unit support for those reads is supplied only by A, U, H, C, P, N.*

Proof. Project the displayed packet. The identity coordinate is I and the composition coordinate is Q, so the public identity and binary-composition reads are not separate host operations.

Use the obligation surface of Definition 484.2. Identity exactness identifies each row of I with the empty Cont continuation attached to an accepted object history, and composition exactness identifies each composite with the displayed Cont ledger stored in Q.

Now apply Definition 484.6. The associativity and unit rows are precisely A and U, transported by H and consumed through C with provenance P and local name N. □

Theorem 484.5 (Finite-kernel category ledger non-escape boundary)**.** *Every* Functor$^{\uparrow}$, NatTrans$^{\uparrow}$, *or* Cat$^{\uparrow}$ *consumer of an accepted* FiniteKernelCategory$^{\uparrow}$ *carrier receives the finite object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and local naming rows $O, M, I, Q, A, U, H, C, P, N$. It receives no quotient morphism type, host arrow equality, functor-category object, or external category library.*

Proof. Start with Definition 484.3 and Definition 484.4. Together they expose the object, hom, identity, composition, associativity, and unit rows as the concrete carrier surface.

Apply Definition 484.7. Functor and natural-transformation consumers read preservation and naturality data only after the finite category rows have been exported through H, C, P, N.

Finally apply Definition 484.8. The same finite surface is the Cat$^{\uparrow}$ interface, and the carrier has no coordinate for quotient morphisms, host arrow equality, functor-category objects, or external category libraries. □

Lean4 ✓ `BEDC.Derived.FiniteKernelCategoryUp.FiniteKernelCategoryLedgerNonescapeBoundary`

Theorem 484.6 (Finite-kernel category composition laws)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted FiniteKernelCategory$^{\uparrow}$ *carrier. The identity, binary composition, associativity, and left and right unit laws exported by K are exactly the displayed ledgers I, Q, A, and U, transported through H and consumed through C. No category law is discharged by ambient category theory or by quotienting hom histories.*

Proof. Read the law rows from Definition 484.1. The identity law is stored in I, binary composition in Q, associativity in A, and the left and right unit laws in U.

Transport compatibility is supplied by H. It keeps endpoints, hom histories, and law ledgers aligned under hsame, so the same displayed rows remain the law witnesses after carrier transport.

Consumer access is supplied by C. Since the carrier contains no ambient category object and no quotient morphism coordinate, every law exported by K is one of the displayed finite ledgers. □

Theorem 484.7 (Finite-kernel functor and natural-transformation handoff)**.** *Every* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer of a* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier receives only the finite object-history row O, displayed hom-history row M, empty-continuation identities I, displayed* Cont *composites Q, associativity and unit ledgers A and U, and the* hsame*,* Cont*,* Pkg*, and* NameCert *provenance rows H, C, P, N. The handoff does not export host functor categories, quotient morphisms, naturality modulo external equality, or any library category object.*

Proof. Start with the NameCert obligations of Definition 484.2. They bind carrier habitation, classifier stability, identity exactness, composition exactness, and ledger non-escape to the same finite packet.

A $\mathsf{Functor}^{\uparrow}$ consumer reads objects and hom histories through O and M, then reads identity and composition preservation against I and Q. The associativity and unit side conditions are the displayed ledgers A and U transported by H.

A $\mathsf{NatTrans}^{\uparrow}$ consumer uses the same rows for component and naturality reads. The only public routes are the continuations in C and the provenance in P and N, so the handoff remains a BEDC-native certificate surface. □

Theorem 484.8 (Finite-kernel category interface handoff)**.** *Every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier exports a* $\mathsf{Cat}^{\uparrow}$*-readable interface only through the displayed object-history row O, hom-history row M, identity row I, composition row Q, associativity and unit ledgers A, U, and structural rows H, C, P, N of Definition 484.1. Any* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer must first read that* $\mathsf{Cat}^{\uparrow}$ *boundary before it reads preservation or naturality data. The handoff exports no quotient morphism type, host equality of arrows, functor category, or external category library.*

Proof. Project the finite carrier $K = (O, M, I, Q, A, U, H, C, P, N)$. The object and hom surfaces are the rows O and M, while the public identity and composite reads are exactly I and Q.

Use Definition 484.6 for the identity, composition, associativity, and unit rows. It identifies the category laws with I, Q, A, U transported through H and consumed through C, so the $\mathsf{Cat}^{\uparrow}$ interface is the displayed finite ledger of chapter 61.

Now apply Definition 484.7. A $\mathsf{Functor}^{\uparrow}$ consumer as in chapter 62 and a $\mathsf{NatTrans}^{\uparrow}$ consumer as in chapter 63 read only after that boundary has exposed objects, hom histories, identities, composites, associativity, and unit ledgers.

The excluded structures are not carrier coordinates. Since K contains no quotient morphism type, host arrow equality, functor category, or library category object, no continuation route or provenance row can export one. □

Theorem 484.9 (Finite-kernel category TasteGate obligation handoff)**.** *For an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, A, U, H, C, P, N),$$

the finite object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and local NameCert *rows form the paper-side obligation handoff consumed by the existing*

TasteGate single-carrier alignment marker. Any consumer of the TasteGate row must keep the displayed category rows $O, M, I, Q, A, U, H, C, P, N$ and cannot replace them by quotient morphisms, host arrow equality, or an external category library.

Proof. Project the carrier $K = (O, M, I, Q, A, U, H, C, P, N)$. The displayed rows are exactly the object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and local naming rows named in Definition 484.1.

Apply Definition 484.2. The five NameCert obligations bind habitation, classifier stability, identity exactness, composition exactness, and ledger non-escape to the same displayed rows.

Use Definition 484.6 and Definition 484.8. They identify the category laws and the public $\mathsf{Cat}^{\uparrow}$ interface with I, Q, A, U, H, C, P, N over the same object and hom rows O, M.

The TasteGate marker is attached immediately after the carrier definition, so its single-carrier alignment reads the same displayed packet. A consumer that replaces these rows by quotient morphisms, host arrow equality, or an external category library no longer consumes the $\mathsf{FiniteKernelCategory}^{\uparrow}$ carrier. □

Theorem 484.10 (Finite-kernel category carrier scope)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Object, hom, identity, and composition reads are grounded only in the displayed rows $O, M, I, Q, A, U, H, C, P, N$: objects are read from O, hom histories from M, identities from I, composites from Q, and their law and transport support from A, U, H, C, P, N.*

Proof. Project the carrier definition. The object coordinate is O, the hom-history coordinate is M, the identity coordinate is I, and the composition coordinate is Q.

Apply Definition 484.2. Carrier habitation and classifier stability keep object and hom reads inside finite BHist histories and componentwise hsame transport.

Next apply Definition 484.6. It identifies the category-law surface with I, Q, A, U transported through H and consumed through C.

Finally use Definition 484.8. The exported category interface is the same finite packet, so no object, hom, identity, or composition read is available outside the displayed rows. □

Theorem 484.11 (Finite-kernel category package and NameCert scope)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier, package provenance and local naming expose exactly the BEDC rows $O, M, I, Q, A, U, H, C, P, N$. They expose no quotient morphism, host equality of arrows, functor-category object, or external category library.*

Proof. Use the carrier scope of Definition 484.10. It binds the public object, hom, identity, composition, law, transport, and continuation reads to the displayed rows.

Apply the NameCert obligations of Definition 484.2. The provenance row P and local naming row N name the same BHist, hsame, Cont, identity, composition, associativity, and unit ledgers.

Now apply Definition 484.5. Since quotient morphisms, host arrow equality, functor-category objects, and external category libraries are not coordinates of K, package provenance and local naming cannot expose them. □

Theorem 484.12 (Finite-kernel category TasteGate scope)**.** *The existing* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *TasteGate marker consumes the same finite packet $O, M, I, Q, A, U, H, C, P, N$. It cannot be read after replacing any row by an external category object, quotient morphism, host arrow equality, or functor-category object.*

Proof. Start with Definition 484.9. The TasteGate handoff is explicitly the finite object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and local NameCert packet.

Use Definition 484.10 to identify the carrier rows consumed by that handoff. The same $O, M, I, Q, A, U, H, C, P, N$ rows ground every category-shaped read.

Then use Definition 484.11. External category objects, quotient morphisms, host arrow equality, and functor-category objects are absent from the package and naming surface.

Therefore replacing any displayed row by one of those external objects leaves the accepted carrier. The TasteGate marker has no route from such a replacement back to $\mathsf{FiniteKernelCategory}^{\uparrow}$. □

Theorem 484.13 (Finite-kernel category root obligation scope)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Its root obligation scope is exactly the BEDC object and hom boundary* O, M*, the identity and composition rows* I, Q*, the associativity and unit rows* A, U*, the componentwise* hsame *transports* H*, the displayed* Cont *consumer routes* C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*. The scope is readable by* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{NatTrans}^{\uparrow}$ *consumers only through those rows and exports no quotient morphism, host arrow equality, functor-category object, or external category library.*

Proof. Start with Definition 484.10. It binds object, hom, identity, and composition reads to O, M, I, Q, H, C, P, N over finite BHist, hsame, and Cont rows.

Apply Definition 484.6. The associativity and unit obligations are the displayed rows A and U, transported through H and consumed through C with the same provenance and local name.

Finally use Definition 484.5 and Definition 484.12. Category-shaped downstream consumers receive only the finite packet surface, and the TasteGate handoff has no route back from quotient morphisms, host equality, functor-category objects, or an external category library. □

Theorem 484.14 (Finite-kernel category scoped closure package)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, A, U, H, C, P, N),$$

the scoped package consists exactly of the object row O*, hom row* M*, identity row* I*, composition row* Q*, associativity row* A*, unit row* U*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*. These rows are bound to the same* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and existing* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *TasteGate marker. A* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, or* $\mathsf{NatTrans}^{\uparrow}$ *consumer may cite the root package only by preserving this whole row family.*

The scoped package introduces no external category object, quotient morphism, host arrow equality, functor-category object, or ambient category library.

Proof. Start with Definition 484.2. Carrier exposure, classifier stability, identity exactness, composition exactness, and ledger non-escape already bind $O, M, I, Q, A, U, H, C, P, N$ to the displayed finite BHist, hsame, Cont, Pkg, and NameCert surface.

Next apply Definition 484.6. It makes associativity and unit part of the same law ledger as identity and composition, transported through H and consumed through C with provenance P and local name N.

Finally use Definition 484.13 and Definition 484.9. The root scope is the exact row family read by downstream categorical consumers, and the TasteGate handoff consumes the same single-carrier

packet. By Definition 484.5, no external category object or quotient morphism can replace any row. □

Theorem 484.15 (Finite-kernel category row scope exhaustion)**.** *Let $K = (O, M, I, Q, A, U, H, C, P, N)$ be an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Every root-obligation read of K is exhausted by the rows $O, M, I, Q, A, U, H, C, P, N$ over the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and existing TasteGate support. In particular, object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and local naming data cannot be distributed across a larger ambient categorical carrier before a consumer reads the* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *root package.*

Proof. First apply Definition 484.13. It states that the root obligation surface is exactly the object and hom boundary, identity and composition rows, associativity and unit rows, componentwise transport, continuation, provenance, and local naming rows.

Use Definition 484.17. The handoff to $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert consumers is the single package $O, M, I, Q, A, U, H, C, P, N$, not a family of detachable ambient objects.

Now apply Definition 484.8 and Definition 484.7. Category, functor, and natural-transformation consumers read only after this finite package has exposed the row family that supports their preservation and naturality data.

The exhaustion is closed by Definition 484.5. Since the carrier exports no quotient morphism, host arrow equality, functor-category object, or external category library, no additional root row remains outside the displayed package. □

Theorem 484.16 (Finite-kernel category public consumer boundary)**.** *A public consumer may treat the* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *root package as scoped only when it reads object, hom, identity, composition, associativity, unit, transport, continuation, provenance, local naming, and the existing TasteGate marker through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. This is the consumer boundary for* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{NatTrans}^{\uparrow}$ *handoffs: the consumer receives a finite BEDC package, not a verified bridge claim and not an imported external category object.*

Proof. Begin with Definition 484.14. It identifies the scoped row package and binds it to the existing TasteGate support over the displayed BEDC support rows.

Next use Definition 484.15. A consumer that has not read all ten rows has not exhausted the root obligation surface, while a consumer that has read them has the finite package required for the downstream categorical interface.

Finally apply Definition 484.8 together with Definition 484.7. These handoffs make $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, and $\mathsf{NatTrans}^{\uparrow}$ consumers depend on the finite package before preservation or naturality data is read. The ledger non-escape theorem and TasteGate scope then exclude quotient morphisms, external category objects, aggregate bridge claims, and any imported category library from the public consumer boundary. □

Theorem 484.17 (Finite-kernel category root handoff package)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier $K = (O, M, I, Q, A, U, H, C, P, N)$, the root handoff package to* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*,* $\mathsf{NatTrans}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *consumers is the single package*

$$O, M, I, Q, A, U, H, C, P, N.$$

The package first exposes the object and hom boundary, then the identity and composition ledgers, then the associativity and unit rows, and finally the transport, continuation, provenance, and local naming rows. It does not expose any consumer route that skips P, N or replaces the local NameCert rows by a host category object.

Proof. Use Definition 484.2. The obligation surface already lists carrier habitation, classifier stability, identity exactness, composition exactness, and ledger non-escape for the same displayed rows.

Next apply Definition 484.8. The $\mathsf{Cat}^{\uparrow}$ interface is the displayed finite ledger, so every functor or natural-transformation consumer reads the object, hom, identity, composition, associativity, and unit rows before preservation or naturality data.

The package and naming rows are fixed by Definition 484.11. Since P and N name only the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ packet, the root handoff cannot be detached from the local certificate. □

Theorem 484.18 (Finite-kernel category object row totality)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Every accepted object read of K factors through the displayed object-history row O. The factorization keeps the object witness in the same* BHist *history, componentwise* hsame *transport row H, displayed* Cont *consumer row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}$ *row N; no category, functor, or natural-transformation consumer can read an object of K outside that object-history row.*

Proof. Read the carrier through Definition 484.1. The only object coordinate of the accepted packet is O, and its route is presented together with the transport, continuation, provenance, and local naming rows.

Apply Definition 484.2. Carrier habitation and classifier stability bind the accepted object read to the displayed BHist row and to the componentwise hsame transport that keeps the same packet visible to downstream consumers.

The law rows do not create another object source. By Definition 484.6, identity, composition, associativity, and unit use the displayed categorical rows over the same object boundary, and Definition 484.5 prevents replacing that boundary by host object equality, quotient objects, or an external category library. □

Theorem 484.19 (Finite-kernel category morphism route stability)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier $K = (O, M, I, Q, A, U, H, C, P, N)$, each accepted hom read and each endpoint transport used by a* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, or* $\mathsf{NatTrans}^{\uparrow}$ *consumer remains inside the displayed morphism route M, H, C, P, N. In particular, source and target movement is recorded by the same componentwise* hsame *transport, displayed* Cont *continuation, package provenance, and local name that expose the hom history, rather than by host arrow equality or a quotient morphism.*

Proof. Start from Definition 484.1. The hom row is M, and the only endpoint-transport data attached to it are the displayed rows H, C, P, N.

Use Definition 484.2. Classifier stability and composition exactness force hom histories and their endpoints to travel through the same componentwise hsame and Cont rows before the packet is consumed.

The category-law surface preserves that route. Apply Definition 484.6 to keep identity, composition, associativity, and unit reads over the same hom and endpoint rows, then use Definition 484.5 to exclude any replacement by host arrows, quotient morphisms, or external categorical equality. □

Theorem 484.20 (Finite-kernel category composition ledger nonescape)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Every identity, composition, associativity, or unit consumer reads only the displayed ledger rows* I, Q, A, U, H, C, P, N*. These consumers cannot replace the ledger by quotient morphisms, host arrow equality, functor-category equality, or an external category-library proof of the same laws.*

Proof. Project the ledger coordinates of Definition 484.1. Identities are carried by I, composites by Q, associativity by A, and unit laws by U, all with the same transport, continuation, provenance, and local naming rows.

Apply Definition 484.2. Identity exactness, composition exactness, and ledger non-escape are obligations of the same local NameCert surface, so a law consumer cannot detach I, Q, A, U from H, C, P, N.

Finally use Definition 484.6. It makes the displayed identity, composition, associativity, and unit rows the law surface consumed by categorical readers. The boundary theorem, Definition 484.5, rules out a second route through quotient morphisms, host arrow equality, or an ambient category library. □

Theorem 484.21 (Finite-kernel category root downstream unblock)**.** *Every downstream categorical consumer that asks for a finite BEDC category, functor source, natural-transformation source, or TasteGate handoff from* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *is unblocked by the root package*

$$O, M, I, Q, A, U, H, C, P, N$$

provided it consumes the object and hom boundary, identity and composition rows, ledger non-escape clause, and TasteGate handoff together. A consumer that omits one of these rows has not consumed the $\mathsf{FiniteKernelCategory}^{\uparrow}$ *root obligation surface.*

Proof. First build the categorical boundary. Apply Definition 484.13 to obtain the exact object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and naming surface.

Next build the handoff package. Apply Definition 484.17; it orders the $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ reads through the same finite rows.

Finally compose with Definition 484.9. The TasteGate side consumes the same single-carrier packet, so a downstream route that preserves all listed rows is unblocked, while a route omitting object/hom data, identity/composition data, ledger non-escape, or TasteGate handoff leaves the root obligation surface. □

Theorem 484.22 (Finite-kernel category downstream root package)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier* $K = (O, M, I, Q, A, U, H, C, P, N)$*, the downstream root package for* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{NatTrans}^{\uparrow}$ *consumers is exactly the displayed row sequence*

$$O, M, I, Q, A, U, H, C, P, N.$$

It consists of the object and hom boundary, identity and composition rows, associativity and unit rows, componentwise hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. No downstream categorical consumer receives a smaller root package by replacing those rows with quotient morphisms, host arrow equality, functor-category objects, or an external category library.*

Proof. Start with the root obligation scope, Definition 484.13. It identifies $O, M, I, Q, A, U, H, C, P, N$ as the complete finite surface read by $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, and $\mathsf{NatTrans}^{\uparrow}$ consumers.

Next apply the root handoff package and category interface handoff. Theorem 484.17 orders the object, hom, identity, composition, law, transport, continuation, provenance, and naming rows, while Definition 484.8 makes the same rows the categorical interface for downstream reads.

Finally use Definition 484.5. Since the ledger boundary exports no quotient morphism, host arrow equality, functor-category object, or external category library, the downstream package is exactly the displayed row sequence and not a larger ambient categorical object. □

Theorem 484.23 (Finite-kernel category hom endpoint transport determinacy)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Any accepted hom-history read transported by componentwise* hsame *preserves the displayed source and target object rows named by the same packet. A* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer cannot reuse the hom row* M *with different endpoints unless it also transports the corresponding object rows through the same* H, C, P, N *packet.*

Proof. Start with the object and hom boundary, Definition 484.3. It makes the object rows O and hom rows M the only displayed endpoints and arrows available to a category-shaped consumer.

Apply the identity and composition boundary together with the composition laws, Definition 484.4 and Definition 484.6. Identities and composites are stored in I, Q, A, U only after their source and target object rows have been kept aligned through the same transport and continuation surface.

Finally use the category interface handoff and ledger non-escape boundary, Definition 484.8 and Definition 484.5. A functor or natural-transformation reader therefore receives endpoint data only as part of the displayed $O, M, I, Q, A, U, H, C, P, N$ packet. Reusing M with different endpoints would require an extra host arrow equality or quotient morphism, and those coordinates are not exported by the carrier. □

Theorem 484.24 (Finite-kernel categorical row readback)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. Every* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, or* $\mathsf{NatTrans}^{\uparrow}$ *readback of* K *factors through exactly the displayed rows* $O, M, I, Q, A, U, H, C, P, N$*: category reads consume object, hom, identity, composition, associativity, and unit rows; functor reads consume the same rows before preservation data; natural-transformation reads consume the same rows before component and naturality data. No readback may replace those rows by quotient morphisms, host arrow equality, a functor-category object, or an external category library.*

Proof. Project the carrier of Definition 484.1. The accepted packet has only the rows $O, M, I, Q, A, U, H, C, P, N$ available for categorical consumers.

Apply the category interface handoff, Definition 484.8. It makes $O, M, I, Q, A, U, H, C, P, N$ the $\mathsf{Cat}^{\uparrow}$-readable boundary and forces $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ consumers to read that boundary first.

Use the functor and natural-transformation handoff together with the ledger non-escape theorem, Definition 484.7 and Definition 484.5. The handoff supplies no quotient morphism, host equality, functor-category object, or external library coordinate, so every readback factors through exactly the finite BEDC row sequence. □

Theorem 484.25 (Finite-kernel functor identity preservation)**.** *Every* $\mathsf{Functor}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier* $K = (O, M, I, Q, A, U, H, C, P, N)$ *preserves the displayed identity row* I *through the same object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and naming rows* $O, M, I, Q, A, U, H, C, P, N$*. The preservation obligation is a BEDC row obligation and is not discharged by a quotient morphism, host arrow equality, external category library, or functor-category object.*

Proof. Start with the category, functor, and natural-transformation readback theorem, Definition 484.24. It makes the accepted consumer read the finite row sequence $O, M, I, Q, A, U, H, C, P, N$ rather than an ambient category object.

Apply the functor and natural-transformation handoff, Definition 484.7. A $\mathsf{Functor}^{\uparrow}$ consumer can inspect identity preservation only after the identity row I has been exported with the carrier rows that determine its source and target histories.

Finally use the composition provenance factorization and ledger non-escape boundary, Definition 484.28 and Definition 484.5. The provenance and local naming rows keep the preservation witness inside the displayed BEDC packet, so no quotient or host equality route can replace I. □

Theorem 484.26 (Finite-kernel functor composition preservation)**.** *Every* $\mathsf{Functor}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier* $K = (O, M, I, Q, A, U, H, C, P, N)$ *preserves the displayed composition row* Q *through the same row sequence* $O, M, I, Q, A, U, H, C, P, N$*. Composition preservation may use the associativity row* A*, the unit row* U*, componentwise transport* H*, continuation consumption* C*, provenance* P*, and local naming* N*, but it has no separate quotient morphism, host arrow equality, external category library, or functor-category object.*

Proof. Read the accepted consumer through Definition 484.24. The composition preservation input is the displayed row Q in the same packet as $O, M, I, A, U, H, C, P, N$.

Use Definition 484.28. It factors composition, associativity, and unit consumption through Q, A, U, H, C, P, N, so a functor cannot preserve composition by bypassing the provenance row or the local $\mathsf{NameCert}$ row.

Now apply Definition 484.7 and Definition 484.5. The handoff exports the displayed category-law rows to $\mathsf{Functor}^{\uparrow}$ consumers, and the boundary excludes quotient morphisms, host equality, external category libraries, and functor-category objects from the preservation surface. □

Theorem 484.27 (Finite-kernel natural-transformation square boundary)**.** *Every* $\mathsf{NatTrans}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier* $K = (O, M, I, Q, A, U, H, C, P, N)$ *reads the naturality square boundary through the displayed object row* O*, hom row* M*, identity row* I*, composition row* Q*, associativity and unit rows* A, U*, componentwise transport* H*, continuation row* C*, provenance row* P*, and local naming row* N*. The square has no boundary edge supplied by quotient morphisms, host arrow equality, an external category library, or a functor-category object.*

Proof. Begin with the readback theorem, Definition 484.24. It routes $\mathsf{NatTrans}^{\uparrow}$ consumers through the finite row sequence, so the naturality square boundary is assembled from displayed object and hom histories and their identity and composition rows.

Apply the functor and natural-transformation handoff, Definition 484.7. The handoff places naturality inspection after the category rows O, M, I, Q, A, U and their transport, continuation, provenance, and naming rows have been exported.

Finish with composition provenance factorization and ledger non-escape, Definition 484.28 and Definition 484.5. These results keep every boundary edge inside $O, M, I, Q, A, U, H, C, P, N$ and block any external category-library or functor-category replacement. □

Theorem 484.28 (Finite-kernel composition provenance factorization)**.** *For every accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier* $K = (O, M, I, Q, A, U, H, C, P, N)$*, composition, associativity, and unit consumption factor through the composition row* Q*, associativity row* A*, unit row* U*, componentwise transport row* H*, displayed continuation row* C*, package provenance row* P*, and local* $\mathsf{NameCert}$ *row* N*. A consumer cannot skip the provenance row or the local NameCert row while still consuming the category-law surface of* K*.*

Proof. Start with the law rows of Definition 484.1. The composition row is Q, associativity is recorded in A, and the left and right unit laws are recorded in U.

Apply the composition laws, Definition 484.6. They require the law rows to be transported through H and consumed through C, so the laws are not detached from the displayed packet.

Finally use the interface handoff and hom endpoint transport determinacy, Definition 484.8 and Definition 484.23. These force categorical consumers to keep endpoints, hom histories, law rows, provenance, and local naming in the same Q, A, U, H, C, P, N factorization. □

Theorem 484.29 (Finite-kernel bridge consumer nonescape)**.** *Every bridge-facing categorical consumer of an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier receives only the finite BEDC packet*

$$O, M, I, Q, A, U, H, C, P, N.$$

It receives no quotient morphism, host arrow equality, functor-category object, external category library, or bridge route that reads category, functor, or natural-transformation data outside the displayed rows.

Proof. First read the carrier and category interface through Definition 484.1 and Definition 484.8. The bridge consumer starts from the same object, hom, identity, composition, associativity, unit, transport, continuation, provenance, and naming rows.

Next apply Definition 484.7. It keeps functor and natural-transformation consumers on the same finite row sequence before preservation, components, or naturality are inspected.

Finish with the ledger non-escape boundary and endpoint determinacy, Definition 484.5 and Definition 484.23. Since the carrier has no quotient, host equality, functor-category, or external library coordinate, a bridge-facing consumer cannot export one. □

Theorem 484.30 (Finite-kernel category carrier habitation scope)**.** *For an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, A, U, H, C, P, N),$$

carrier habitation is grounded exactly by the finite object-history row O*, the displayed hom-history row* M*, the empty-continuation identity row* I*, and the local* $\mathsf{NameCert}$ *row* N*. The habitation witness uses no quotient morphism, host category object, ambient arrow equality, or external category library object.*

Proof. Start from the carrier coordinates in Definition 484.1. The object carrier is presented by finite BHist histories in O, the hom carrier by displayed histories in M, identities by the empty Cont continuations recorded in I, and local naming by N.

Use the obligation surface of Definition 484.2. Its habitation row binds those object, hom, identity, and local naming coordinates to the same finite packet, so there is no second carrier source to read.

Finally apply the root obligation boundary, Definition 484.13. That boundary exports only $O, M, I, Q, A, U, H, C, P, N$ to categorical consumers; quotient morphisms, host category objects, and ambient category-library objects are not coordinates of the accepted carrier. □

Theorem 484.31 (Finite-kernel category classifier stability scope)**.** *For an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, A, U, H, C, P, N),$$

classifier transport is exactly the componentwise hsame *row* H *over the object, hom, identity, composition, associativity, and unit ledgers. The transport is consumed only through the displayed continuation row* C*, the package provenance row* P*, and the local* NameCert *row* N*.*

Proof. Read the classifier coordinates from Definition 484.1. The row H is the componentwise hsame transport for the displayed object and hom histories and for the ledger rows that depend on them.

Use Definition 484.6 to identify the dependent law surface. Identity, composition, associativity, and unit reads are the rows I, Q, A, U, so classifier stability acts on the full law ledger rather than on a detached ambient category object.

Now apply Definition 484.2 and Definition 484.13. The same componentwise transport is consumed through C and named through P, N; no consumer receives an alternate classifier route outside Cont, Pkg, and NameCert. □

Theorem 484.32 (Finite-kernel category identity-composition exactness scope)**.** *For an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, A, U, H, C, P, N),$$

identity exactness is the row I *and composition exactness is the row* Q*. Associativity and unit support enter only through* A *and* U*, transported by* H*, consumed through* C*, recorded by* P*, and named locally by* N*.*

Proof. Project the carrier of Definition 484.1. The identity row is I, and the composition row is Q; both are displayed BEDC rows over the same object and hom histories.

Apply Definition 484.2. Its identity-exactness and composition-exactness obligations attach I and Q to the same finite BHist, hsame, Cont, Pkg, and NameCert packet.

Finally use Definition 484.6 and Definition 484.13. Associativity and unit support are precisely A, U, H, C, P, N, so exactness cannot be discharged by an external composition law or by a quotient morphism outside the displayed carrier. □

Theorem 484.33 (Finite-kernel displayed source totality)**.** *Every accepted finite-kernel category consumer must display the complete row family*

$$O, M, I, Q, A, U, H, C, P, N$$

of the carrier $K = (O, M, I, Q, A, U, H, C, P, N)$*. The object and hom rows are read over finite* BHist *histories, the identity, composition, associativity, and unit rows are read through* hsame *and* Cont*, the provenance row is read through* Pkg*, and the local naming row is read through* NameCert*. No downstream* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, or* $\mathsf{NatTrans}^{\uparrow}$ *consumer has accepted the root package until all ten rows are displayed together.*

Proof. Project the carrier of Definition 484.1. Its visible source is the ten-row packet $O, M, I, Q, A, U, H, C, P, N$, with no separate coordinate for a host category object or a quotient morphism carrier.

Apply the NameCert obligations, Definition 484.2. They bind carrier habitation, classifier stability, identity exactness, composition exactness, and ledger non-escape to the same BHist, hsame, Cont, Pkg, and NameCert rows.

Finally use the root obligation scope, Definition 484.13. It identifies that same displayed row family as the only source package exported to categorical consumers, so a partial row read is not an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ consumer read. □

Theorem 484.34 (Finite-kernel composite ledger determinacy)**.** *Let $K = (O, M, I, Q, A, U, H, C, P, N)$ be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ carrier. If two composite reads have the same displayed object rows O, hom rows M, identity row I, composition row Q, continuation row C, provenance row P, and local naming row N, then they determine the same public ledger read of K. The determinacy is internal to the displayed carrier: it uses the composition ledger nonescape boundary and the identity-composition exactness scope, not an external category-library equality.*

Proof. First read the two composites through the displayed carrier. The shared object and hom rows fix the endpoints, while the shared identity and composition rows fix the BEDC-native law surface.

Apply the identity-composition exactness scope, Definition 484.32. It identifies identity exactness with I and composition exactness with Q, with associativity, unit, transport, continuation, provenance, and naming supplied by the same packet.

Finish with the composition ledger nonescape theorem, Definition 484.20. Since a public composite cannot leave Q, C, P, N for a quotient morphism, host arrow equality, or external category object, the two reads expose the same public ledger. □

Theorem 484.35 (Finite-kernel natural-transformation consumer scope)**.** *The downstream $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, and $\mathsf{NatTrans}^{\uparrow}$ consumers may read the $\mathsf{FiniteKernelCategory}^{\uparrow}$ root package only through the full displayed row family*

$$O, M, I, Q, A, U, H, C, P, N$$

together with the existing $\mathsf{FiniteKernelCategory}^{\uparrow}$ TasteGate marker. They may not read it through quotient morphisms, host arrow equality, functor-category objects, or an external category library.

Proof. Start with the displayed source totality theorem, Definition 484.33. It forces every accepted downstream consumer to display the complete ten-row source package over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Next apply the TasteGate handoff, Definition 484.9. The existing TasteGate marker consumes the same single-carrier alignment, so the consumer scope includes that marker only as a read of the displayed package.

Finally use the category, functor, and natural-transformation readback theorem, Definition 484.24. It routes all three downstream consumers through the same finite row sequence and excludes quotient morphisms, host arrow equality, functor-category objects, and external category libraries from the public read. □

Theorem 484.36 (Finite-kernel category public export interface)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ carrier. The public export of K is exactly the displayed row family $O, M, I, Q, A, U, H, C, P, N$. It is read first as the $\mathsf{Cat}^{\uparrow}$ interface carried by the finite kernel package, and then as the required source boundary for every downstream $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ consumer. No quotient morphism, host arrow equality, functor-category object, or external category library is part of that public export.

Proof. Start with the category interface handoff, Definition 484.8. It identifies the displayed rows $O, M, I, Q, A, U, H, C, P, N$ as the $\mathsf{Cat}^{\uparrow}$ interface of the accepted finite-kernel carrier.

Next apply the category, functor, and natural-transformation row readback, Definition 484.24. It makes the same row family the source boundary read by $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ consumers after the category interface has been exposed.

The downstream scope is fixed by Definition 484.35. It permits those consumers to read only the complete displayed package together with the existing TasteGate marker.

Finally apply the bridge consumer nonescape boundary, Definition 484.29. Since bridge consumers cannot extract quotient morphisms, host arrow equality, functor-category objects, or an external category library, the displayed row family is the whole public export. □

Theorem 484.37 (Finite-kernel category standard bridge surface)**.** *Let*

$$K = (O, M, I, Q, A, U, H, C, P, N)$$

be an accepted $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. The standard bridge surface for K is the single displayed BEDC packet consisting of the kernel-category object and hom rows O, M, the unary-continuation identity and composition rows I, Q, the associativity and unit ledgers A, U, and the structural rows H, C, P, N. This surface is read as a* $\mathsf{Cat}^{\uparrow}$ *interface and then as the common source boundary for* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *consumers; it has no quotient morphism coordinate, host arrow equality, functor-category object, or external category library.*

Proof. Project the public export interface of Definition 484.36. It identifies the visible export with exactly the row family $O, M, I, Q, A, U, H, C, P, N$.

The source of the bridge is fixed by the kernel-category and unary-continuation dependencies recorded in Definition 484.1. The object and hom readings are O, M, while identity and composition are routed through the empty and displayed Cont rows I, Q.

Now apply the category interface handoff and row readback, Definition 484.8 and Definition 484.24. They make the same displayed packet first the $\mathsf{Cat}^{\uparrow}$ interface and then the source boundary for $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ reads.

Finally use the bridge consumer nonescape boundary, Definition 484.29. Since every excluded ambient categorical object is absent from K, the standard bridge surface is precisely the displayed finite packet. □

Theorem 484.38 (Finite-kernel category functor and natural-transformation bridge boundary)**.** *For an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier $K = (O, M, I, Q, A, U, H, C, P, N)$, every* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer that uses K through the standard bridge surface must first consume the* $\mathsf{Cat}^{\uparrow}$ *boundary supplied by $O, M, I, Q, A, U, H, C, P, N$. Preservation of identities, preservation of composition, and naturality squares are therefore read only through the finite kernel-category rows and the shared structural rows H, C, P, N.*

Proof. Begin with the standard bridge surface, Definition 484.37. It fixes the only available source for categorical reads as the displayed finite packet.

Apply Definition 484.8. This places all functor and natural-transformation reads after the $\mathsf{Cat}^{\uparrow}$ interface has exposed objects, hom histories, identities, composites, associativity, and units.

The functor rows are then constrained by Definition 484.25 and Definition 484.26. The natural-transformation boundary is constrained by Definition 484.27.

The consumer scope theorem Definition 484.35 keeps each of these reads on the same H, C, P, N structural route, so no downstream consumer can detach preservation or naturality data from the finite bridge packet. □

Theorem 484.39 (Finite-kernel category bridge nonescape exactness)**.** *Let $K = (O, M, I, Q, A, U, H, C, P, N)$ be an accepted* $\mathsf{FiniteKernelCategory}^{\uparrow}$ *carrier. A bridge consumer has access exactly to the finite kernel-category rows O, M, I, Q, A, U and the shared structural rows H, C, P, N. Thus every* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{NatTrans}^{\uparrow}$ *read that factors through K is exhausted by the public export interface and the standard bridge surface, and no read escapes to quotient morphisms, host arrow equality, functor-category objects, or an external category library.*

Proof. Use the public export interface, Definition 484.36, to identify the complete exported row family.

Next compose it with the standard bridge surface, Definition 484.37, and the functor and natural-transformation bridge boundary, Definition 484.38. These results route the category, functor, and natural-transformation consumers through the same packet K.

Finally apply Definition 484.29. The carrier has no coordinate for any excluded ambient categorical datum, so the displayed row family is not merely sufficient for the bridge consumer; it is exact. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteKernelCategory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteKernelCategory}^{\uparrow}$ packet is generated from finite object histories, displayed hom histories, empty-continuation identity rows, displayed Cont composition rows, associativity and unit ledgers, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteKernelCategory}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public package records the BEDC-native rows $O, M, I, Q, A, U, H, C, P, N$, the finite category carrier, NameCert obligations, category-law rows, interface handoff, TasteGate obligation handoff, displayed source totality in Definition 484.33, composite ledger determinacy in Definition 484.34, functor identity preservation in Definition 484.25, functor composition preservation in Definition 484.26, natural-transformation square boundary in Definition 484.27, downstream consumer scope in Definition 484.35, public export interface in Definition 484.36, standard bridge surface in Definition 484.37, functor and natural-transformation bridge boundary in Definition 484.38, bridge nonescape exactness in Definition 484.39, and checked TasteGate single-carrier alignment over BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteKernelCategory}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FiniteKernelCategoryUp.FiniteKernelCategoryTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Not claimed: This chapter does not close arbitrary categorical constructions, quotient morphisms, host category libraries, functor categories, external category libraries, or checked bridge verification.

Upgrade path: The next verification step is a checked StandardBridge theorem; the present block records a paper bridge surface grounded by the existing TasteGate alignment target.

Chapter 485

A Concrete Naming Certificate for $\mathsf{ContinuationMonad}^{\uparrow}$

The $\mathsf{ContinuationMonad}^{\uparrow}$ packet packages the continuation-as-small-step reading as a finite Kleisli-style certificate over existing BEDC rows. It names source histories, target histories, continuation tails, the unit tail, displayed Cont routes, componentwise hsame transports, Pkg provenance, and a local NameCert ledger. It does not add quotient morphisms, host functions, or a general category primitive.

The packet consumes the authorized generator-boundary discipline of chapter 486 and the continuation hom-carrier surface of chapter 61. Generator closure supplies the refusal of unlisted continuation rules, while $\mathsf{Cat}^{\uparrow}$ supplies the CategoryHomCarrier and ContinuationMorphism reading used in the ledger row.

The dependency route is explicit: $\mathsf{ContinuationMonad}^{\uparrow}$ reads chapter 486 for the refusal of unlisted continuation generators and chapter 61 for the hom-carrier reading. Downstream unit and bind consumers must cite both sibling routes before treating the continuation packet as a monad surface.

Definition 485.1 (ContinuationMonad NameCert source)**.** *A* $\mathsf{ContinuationMonad}^{\uparrow}$ *source is a finite* BHist *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N)$$

with the following displayed coordinates:

1. *A, B, and C are the source, middle, and target history rows for a two-step continuation read.*
2. *f is the displayed continuation tail from A to B, and g is the displayed continuation tail from B to C.*
3. *u is the displayed unit tail, read as the empty continuation route at a history row.*
4. *H stores the componentwise* hsame *transports for the three history rows, the two non-unit tails, the unit tail, and the displayed composite tail.*
5. *K stores the displayed* Cont *routes for the unit read, the two small-step reads, and their composed Kleisli read.*
6. *L is the finite ledger recording continuation associativity and unit behaviour as displayed tail-routing facts, together with the* CategoryHomCarrier *and* ContinuationMorphism *readings used by* $\mathsf{Cat}^{\uparrow}$.

7. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *ledger for carrier admission, classifier stability, ledger exactness, package provenance, and the Kleisli composition boundary.*

The packet is a finite naming certificate. It has no quotient morphism coordinate, no host-function coordinate, and no general category object outside BHist.

Lean4 (definition) `BEDC.Derived.ContinuationMonadUp.ContinuationMonadNameCertSource`

Definition 485.2 (ContinuationMonad carrier)**.** *The concrete* $\mathsf{ContinuationMonad}^{\uparrow}$ *carrier is the finite* BHist *triple surface*

$$\mathcal{K}(A, B, C) = (f, g, u; H, K, L, N)$$

displayed inside Definition 485.1. The carrier rows are the source, middle, and target histories A, B, C; the two continuation tails f, g; the unit tail u; the componentwise hsame *row H; the* Cont *route row K; the unit and associativity ledger L; and the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row N.*

Admission to the carrier means that these rows form the only readable continuation object. In particular, the unit tail is a displayed tail row in the same packet, and the two-step readback is a Cont *ledger route rather than a host Kleisli category.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier finite packet scope`
Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit tail endpoint package`

Theorem 485.3 (ContinuationMonad unit laws)**.** *For every accepted carrier $\mathcal{K}(A, B, C)$ of Definition 485.2, the displayed unit tail u is left and right neutral for the finite continuation routes recorded in K. The left-unit route reads u before f and returns the same A-to-B tail row, while the right-unit route reads f before u and returns the same A-to-B tail row, both under the componentwise* hsame *transports stored in H.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit route laws`

Proof. Read the carrier through the route row K. The unit route and the two small-step routes are displayed coordinates of the packet, so the left-unit and right-unit reads are finite ledger reads rather than host equalities.

The transport row H supplies the componentwise hsame comparisons needed to identify the returned tail row with f. Since the unit tail u has no coordinate outside the packet, neutrality is exactly the ledger row for the displayed unit route. □

Theorem 485.4 (ContinuationMonad associativity)**.** *For every accepted carrier $\mathcal{K}(A, B, C)$ of Definition 485.2, the two possible two-step readbacks of f followed by g reassociate to the same displayed composite continuation route. The equality is the associativity row in L transported along H and read through the* Cont *row K.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier associativity`

Proof. The carrier contains only two non-unit continuation tails and one displayed composite readback. Thus both parenthesized route reads must land in the same Cont coordinate of K.

The ledger row L records that this shared coordinate is the associativity witness for the two-step continuation surface. Transport along H preserves the source, middle, target, and tail rows, so the reassociation remains the same finite ledger fact after componentwise hsame transport. □

Theorem 485.5 (ContinuationMonad NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *surface for Definition 485.2 consists of five finite rows: carrier admission for* $\mathcal{K}(A, B, C)$*; classifier stability under componentwise* hsame *transport of histories and tails; unit and associativity ledger exactness for* L*;* Pkg *provenance over* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*, and* ContinuationMorphism*; and local naming rows for the Kleisli composition handoff.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadNameCertObligations`

Proof. Carrier admission is the displayed tuple of Definition 485.2. It fixes the readable histories, tails, routes, ledger, and local naming row.

Classifier stability follows from H, and the route laws follow from Definition 485.3 and Definition 485.4. These rows account for the unit and associativity behaviour inside the finite packet.

The remaining provenance rows package the same data for CategoryUp and for the local naming ledger. No quotient Kleisli category, host function, or general category primitive is added by the certificate. □

Theorem 485.6 (ContinuationMonad carrier obligation)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *source packet of Definition 485.1, carrier admission is exhausted by the rows* $A, B, C, f, g, u, H, K, L, N$*. The admitted object is the displayed two-step continuation surface: source and target histories, continuation tails, the unit tail, the* Cont *routes, componentwise* hsame *transports, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier route closure`

Proof. Project the packet coordinates. The history rows A, B, C supply the BEDC source, middle, and target data, while f, g, u supply the displayed tails used by the small-step interpretation.

The continuation coordinate K records the only admissible routes: unit, first step, second step, and composed step. The transport coordinate H compares exactly those rows componentwise under hsame.

The ledger and certificate rows L, N bind the package provenance and local naming obligations. Since the definition lists no other coordinates, carrier admission cannot read a quotient morphism, host function, or external category primitive. □

Theorem 485.7 (ContinuationMonad classifier stability)**.** *The* $\mathsf{ContinuationMonad}^{\uparrow}$ *classifier is stable under componentwise* hsame *transport of* A, B, C, f, g, u *and of the displayed composite tail. Transported packets preserve the same unit route, the same two small-step* Cont *routes, and the same Kleisli readback recorded by* K *and* L*.*

Proof. Use the componentwise transport row H. It transports the three history rows and the displayed tails without changing the shape of the packet.

The continuation routes are preserved because K stores displayed Cont data rather than host functions. The CategoryUp continuation-morphism reading identifies each transported tail with the same small-step route over the transported endpoints.

Finally the ledger row L records associativity and unit behaviour as continuation-tail routing facts. Repacking those transported rows gives the same local classifier and no access to quotient arrows or host equality. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier classifier stability`

Theorem 485.8 (ContinuationMonad ledger exactness)**.** *The finite ledger of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet is exact for the route from the unit tail and the two displayed continuation tails through the composed Kleisli readback. Every ledger entry is one of the unit, associativity,* Cont*,* hsame*,* CategoryHomCarrier*,* ContinuationMorphism*,* Pkg*, or local* NameCert *rows named in Definition 485.1.*

Proof. Read the source packet. The only dynamic rows are the two displayed tails, the unit tail, and the composed continuation route stored in K.

Associativity and unit behaviour are ledger facts about those displayed tails. They are not imported from host functions or quotient arrows; they are the same continuation-tail facts used by the CategoryUp continuation morphism surface.

The package row records provenance for the finite BEDC objects named in the statement. Thus every ledger entry is accounted for by the packet, and no external category or monad equality can enter the certificate. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier ledger exactness`

Theorem 485.9 (ContinuationMonad NameCert package)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *package consists exactly of five obligation rows: carrier admission, classifier stability under componentwise* hsame*, ledger exactness for unit and associativity routes, package provenance over* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*, and* ContinuationMorphism*, and the Kleisli composition boundary of Definition 485.19.*

Proof. Carrier admission is supplied by Definition 485.6. It identifies the finite source packet and excludes all unlisted carrier data.

Classifier stability is supplied by Definition 485.7. Ledger exactness is supplied by Definition 485.8.

The remaining rows are bookkeeping rows. The Pkg coordinate records provenance over the named BEDC kernel and CategoryUp surfaces, and the local certificate coordinate records that the Kleisli boundary is the public consumer of the same displayed continuation data. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier namecert package`

Theorem 485.10 (ContinuationMonad root generator obligation)**.** *Every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet receives its unit and bind rows only through the displayed* $\mathsf{Cat}^{\uparrow}$ *hom-carrier and the authorized generator boundary. The unit row is the displayed tail* u*, the bind route is the finite composition route through* f, g, K, L*, and both are read after the generator-boundary refusal of unlisted continuation rules.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator obligation`

Proof. Start with carrier admission from Definition 485.6. The accepted packet exposes the histories A, B, C, tails f, g, u, transport row H, route row K, ledger row L, and local naming row N.

Read the categorical surface through chapter 61. The CategoryHomCarrier and ContinuationMorphism entries used by L are readings of the displayed continuation tails, not independent host morphisms.

Now apply the generator boundary of chapter 486. The unit and bind rows can be consumed only as authorized rows of the displayed packet, while unlisted continuation rules are refused at the generator boundary. □

Theorem 485.11 (ContinuationMonad bind-route classifier stability)**.** *Componentwise* hsame *transport of the unit row, bind route, continuation rows, and package rows of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet preserves the local Kleisli classifier. The transported classifier still reads the same* $\mathsf{Cat}^{\uparrow}$ *hom-carrier, the same authorized generator boundary, the same* Cont *routes, and the same* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *ledger.*

Proof. Use classifier stability from Definition 485.7. Transport along H preserves the three histories, the unit tail, the two continuation tails, and the displayed composite tail.

Use ledger exactness from Definition 485.8. The bind route remains a row of K and L, so its category-facing reading is still the CategoryHomCarrier and ContinuationMorphism surface described in chapter 61.

The generator boundary remains local because the transported route has no coordinate for an unlisted continuation rule. Repacking the transported rows with Pkg provenance and the local NameCert row gives the same Kleisli classifier. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bind route classifier stability`

Theorem 485.12 (ContinuationMonad NameCert ledger exactness)**.** *Every consumer-visible continuation-monad route is exhausted by the carrier, classifier, unit, bind, associativity,* Cont*,* Pkg*, and* NameCert *rows of the accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet. No consumer-visible route can bypass the displayed* $\mathsf{Cat}^{\uparrow}$ *hom-carrier or the authorized generator boundary.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier namecert ledger exactness`

Proof. Project the packet using Definition 485.9. The local NameCert package consists of carrier admission, classifier stability, ledger exactness, package provenance, and the Kleisli composition boundary.

The unit and associativity rows are supplied by Definition 485.3 and Definition 485.4. The bind route is the displayed two-step readback through f, g, K, L and is already bounded by the Kleisli composition surface.

Finally combine chapter 61 with chapter 486. The category-facing hom-carrier and the authorized generator boundary account for every public route into the packet, so no host function, quotient Kleisli arrow, or unlisted generator rule remains visible. □

Theorem 485.13 (ContinuationMonad root-unblock source triad)**.** *Every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet must expose the source history A, middle history B, target history C, continuation tails f, g, and unit tail u together before any Kleisli read is available. The first readable root surface is therefore the finite triad-and-tail packet $A, B, C, f, g, u, H, K, L, N$, not an external function space or quotient Kleisli category.*

Proof. Project carrier admission from Definition 485.6. The accepted packet displays the three histories, the two continuation tails, the unit tail, componentwise transports, route rows, ledger rows, and the local name as one finite carrier.

Use ledger exactness from Definition 485.12. A Kleisli read is consumer-visible only through the carrier, classifier, unit, bind, associativity, Cont, Pkg, and NameCert rows of that same packet.

Thus the read cannot begin from a host function, an external quotient, or a partial tail. The root surface is the displayed source, middle, and target triad together with f, g, u, H, K, L, N. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root unblock source triad`

Theorem 485.14 (ContinuationMonad root-unblock bind totality)**.** *Every bind-route consumer of an accepted* ContinuationMonad$^{\uparrow}$ *packet factors through the displayed* Cont *row K, the componentwise* hsame *row H, and the ledger row L. The bind route is total only as a read of $A, B, C, f, g, u, H, K, L, N$ and is not host function composition, host monad equality, or a quotient Kleisli arrow.*

Proof. Start with a bind-route consumer and project the packet rows. The only displayed route row is K, the only componentwise transport row is H, and the consumer-visible accounting row is L.

Use Definition 485.11. Transport along H preserves the local Kleisli classifier and keeps the route on the same displayed histories and continuation tails.

Use Definition 485.12 to close the readback. Since every consumer-visible continuation-monad route is exhausted by the displayed rows, the bind route is total precisely through K, H, L inside the finite packet and has no host-composition coordinate. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bind route totality`

Theorem 485.15 (ContinuationMonad root-unblock downstream package)**.** *The source triad of Definition 485.13 and the bind totality of Definition 485.14 form the downstream root-unblock package consumed by the CategoryUp surface of chapter 61 and the generator-boundary surface of chapter 486. Downstream consumers receive exactly $A, B, C, f, g, u, H, K, L, N$ together with the displayed category and generator readings; they receive no host function composition, quotient Kleisli category, or unlisted continuation rule.*

Proof. First expose the source packet. Definition 485.13 supplies the three histories, two continuation tails, unit tail, transports, routes, ledger, and local name before any Kleisli read.

Next expose the bind route. Definition 485.14 forces every bind consumer through K, H, L inside the same finite packet.

Finally read the downstream surfaces. The CategoryUp chapter reads the displayed hom-carrier and continuation-morphism rows, while the generator-boundary chapter refuses unlisted continuation rules. Together with ledger exactness, these readings package the root-unblock surface without exporting host function composition, quotient Kleisli categories, or extra continuation generators. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root unblock downstream package`

Theorem 485.16 (ContinuationMonad category hom row exposure)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet $Z = (A, B, C, f, g, u, H, K, L, N)$, the downstream* CategoryHomCarrier *and* ContinuationMorphism *readings are exposed only as reads of the displayed tails f, g, u through the visible rows A, B, C, H, K, L, N. They do not introduce an independent category object, quotient Kleisli arrow, ambient hom set, or host-function carrier.*

Proof. Start with carrier obligation. Definition 485.6 identifies the readable packet with $A, B, C, f, g, u, H, K, L, N$ and leaves no coordinate for an external hom carrier.

Use classifier stability and ledger exactness. Definition 485.7 keeps the transported tails on the same histories and route rows, while Definition 485.8 accounts for unit, bind, and associativity inside the displayed ledger.

Finally apply the CategoryUp surface of chapter 61 together with Definition 485.15. The category-facing names are therefore readings of the same tails and ledger rows already exported by the packet, not additional categorical data. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category hom row exposure`

Theorem 485.17 (ContinuationMonad generator boundary refusal)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the generator-boundary sibling refuses any unit row, bind row, continuation rule, or route generator that is not displayed in* $A, B, C, f, g, u, H, K, L, N$*. The refused object cannot re-enter through* Pkg *provenance, CategoryUp notation, or the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row.*

Proof. Project the accepted carrier and its classifier. The carrier obligation fixes the visible rows, and classifier stability keeps every transported continuation read on those rows.

Read the generator boundary through chapter 486. Its authorized generator surface accepts only listed rows, so an unlisted continuation rule has no generator coordinate in the packet.

Ledger exactness closes the refusal. Since Definition 485.12 exhausts every consumer-visible route by the displayed carrier, classifier, unit, bind, associativity, Cont, Pkg, and $\mathsf{NameCert}$ rows, there is no later provenance or category route through which the refused generator can appear. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator boundary refusal`

Theorem 485.18 (ContinuationMonad continuation route ledger totality)**.** *Every unit or bind consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet factors through the displayed* Cont *route row* K*, the componentwise* hsame *transport row* H*, the ledger row* L*, and the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row* N*. The total route is the finite read of* $A, B, C, f, g, u, H, K, L, N$*; it is not host monad evaluation, quotient Kleisli composition, or an unrecorded continuation evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier cont route ledger totality`

Proof. First use the unit and bind rows already exposed in this chapter. Definition 485.3 places the unit tail inside the displayed route, and Definition 485.11 keeps bind reads stable under the H transports.

Next use ledger exactness. Definition 485.12 says every consumer-visible continuation route is exhausted by the finite rows of the same accepted packet.

The CategoryUp and generator-closure siblings supply the two downstream checks. CategoryUp reads the hom and continuation-morphism faces from the displayed tails, while generator closure refuses any route generator outside the packet. Thus every unit or bind consumer factors through K, H, L, N and no hidden evaluator remains. □

Theorem 485.19 (ContinuationMonad Kleisli composition boundary)**.** *Every Kleisli-style consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet factors through the displayed route*

$$A \xrightarrow{f} B \xrightarrow{g} C$$

together with the unit tail u*, the componentwise* hsame *transports, the* Cont *routes in* K*, the ledger* L*, package provenance, and the local naming row* N*. The boundary exports no quotient Kleisli category, no host function composition, no standard monad laws as host equalities, and no general category primitive.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier kleisli composition boundary`

Proof. Start from the consumer route. By carrier admission, it can inspect only the displayed histories, tails, transports, continuations, ledger, package provenance, and local certificate.

Use classifier stability to move along componentwise hsame without changing the route. The displayed two-step path remains the route through f and g, with u serving only as the unit tail inside the same continuation ledger.

Use ledger exactness to identify the composed readback with the continuation route recorded in K and L. Therefore a Kleisli-style consumer is bounded by the named BEDC rows and cannot read quotient Kleisli categories, host functions, host monad equalities, or a general category primitive. □

Theorem 485.20 (ContinuationMonad Kleisli composite scope)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet, any composite consumer that reads*

$$A \xrightarrow{f} B \xrightarrow{g} C$$

must factor through the displayed tails f, g, u, the Cont *route row K, the componentwise* hsame *row H, the ledger row L, the* CategoryHomCarrier *and* ContinuationMorphism *readings,* Pkg *provenance, and the local* NameCert$_{\text{ContinuationMonad}^{\uparrow}}$ *row N. The consumer cannot replace this finite composite by host function composition, quotient Kleisli arrows, or a general category primitive.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier kleisli composite scope`

Proof. Begin with carrier admission. The accepted packet exposes only the histories A, B, C, the tails f, g, u, the transport row H, the continuation route row K, the ledger row L, package provenance, and the local name.

Use classifier stability to move through the componentwise hsame row without changing the displayed two-step route. The CategoryHomCarrier and ContinuationMorphism readings are therefore readings of those same continuation rows, not additional morphism objects.

Apply ledger exactness and the Kleisli composition boundary. They identify the composite readback with the recorded K, L route and confine every consumer of the composite to the displayed BEDC packet.

Since no coordinate names host function composition, quotient Kleisli arrows, or a general category primitive, the composite scope is exhausted by the listed rows. □

Theorem 485.21 (ContinuationMonad scoped route dependencies)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet, the composite consumer route factors through exactly the displayed* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*,* ContinuationMorphism*,* Pkg*, and* NameCert *rows. The dependency surface is the finite route through the histories A, B, C, continuation tails f, g, u, componentwise transports, continuation ledger, package provenance, and local naming row.*

Proof. Start with carrier obligation. It projects the accepted packet to the histories, tails, transport row, continuation route row, ledger, provenance, and local certificate.

Use classifier stability and ledger exactness to keep every read of the composite consumer on those rows. The CategoryHomCarrier and ContinuationMorphism readings are then readings of the same displayed continuation route, not independent categorical data.

Apply the Kleisli boundary and Kleisli composite scope. Together they identify the composite as the finite route through the listed BEDC rows and exhaust the scoped dependencies of the accepted packet. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier scoped route dependencies`

Theorem 485.22 (ContinuationMonad scope non-escape)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet, the scoped composite route exports no quotient Kleisli arrows, no host function composition, and no general category primitive. It exports only the displayed continuation tails,* Cont *route rows, componentwise* hsame *transports, ledger row, package provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier scope nonescape`

Proof. Project the same packet using carrier obligation. The visible rows are finite continuation data, transport data, ledger data, package provenance, and local naming data.

Classifier stability and ledger exactness keep transport and readback inside that finite route. The Kleisli composition boundary excludes quotient Kleisli arrows, host function composition, and general category primitives from the public boundary.

The Kleisli composite scope supplies the final scope claim: every composite consumer has already factored through the displayed rows, so there is no exported object beyond the scoped route. □

Theorem 485.23 (ContinuationMonad route readback determinacy)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the displayed* Cont *route row determines the unit readback, first-step readback, second-step readback, and composite readback under componentwise* hsame *transport. Any consumer reading those four results must use the same rows* u, f, g, H, K, L, N *and receives no second host route, host function composition, quotient Kleisli arrow, or host monad equality.*

Proof. Start with carrier obligation and classifier stability. Carrier admission supplies the displayed tails f, g, u and the route row K, while classifier stability keeps componentwise hsame transport on the same histories and continuation tails.

Use ledger exactness to read the unit, first step, second step, and composite from K and L. The ledger records the readbacks as finite continuation data attached to Z, not as host functions or quotient arrows.

Finally apply the Kleisli composition boundary and scope non-escape. The only route available to a public consumer is the displayed Cont route row, so no second host route or host monad equality can be projected. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier route readback determinacy`

Theorem 485.24 (ContinuationMonad route readback totality)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every public readback of the route is one of the displayed unit, first-step, second-step, or composite rows recorded by* K*,* L*, and* N*. The readback surface is exhausted by the histories* A, B, C*, tails* u, f, g*, componentwise transport* H*, continuation route row* K*, ledger row* L*, and local naming row* N*.*

Proof. Start with the carrier obligation. It projects the accepted packet to exactly $A, B, C, f, g, u, H, K, L, N$, so a public route read has no source coordinate outside the displayed continuation packet.

Use classifier stability and ledger exactness to keep the unit, first-step, second-step, and composite reads on H, K, and L. These rows record the route as finite continuation data rather than as host functions.

Finally apply Definition 485.19, Definition 485.21, and Definition 485.23. The boundary supplies the finite Kleisli-facing route, the dependency theorem restricts all consumers to the listed BEDC rows, and determinacy says the four displayed readbacks exhaust the public route. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier route readback totality`

Theorem 485.25 (ContinuationMonad unit-composite readback totality)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, unit-law consumers and composite-law consumers are jointly exhausted by* u, f, g, H, K, L, N*. They may read the displayed unit row, the two displayed step rows, the displayed composite row, componentwise* hsame *transport, the continuation ledger, and the local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row, but they cannot request host monad equalities.*

Proof. Project the packet by carrier obligation. The only unit input is the tail u, and the only composite inputs are the tails f and g together with the route row K.

Apply classifier stability and ledger exactness. Stability keeps all unit and composite reads under the same transport row H, while exactness records their readbacks through K and L.

Now use Definition 485.19, Definition 485.21, and Definition 485.23. The boundary excludes host monad equality, the scoped dependencies leave only u, f, g, H, K, L, N, and determinacy identifies any public unit-law or composite-law read with one of those displayed rows. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit composite readback totality`

Theorem 485.26 (ContinuationMonad category-boundary readback totality)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the* CategoryHomCarrier *and* ContinuationMorphism *handoff is total exactly on the same finite route rows* $A, B, C, f, g, u, H, K, L, N$*. A category-boundary consumer receives only the displayed* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert *surface of that packet, with no independent category object, host function, quotient Kleisli arrow, or host monad equality.*

Proof. First use the carrier obligation to expose the route rows $A, B, C, f, g, u, H, K, L, N$. The category-facing names are attached to those histories, tails, transport, route, ledger, provenance, and naming rows.

Classifier stability and ledger exactness identify the CategoryHomCarrier and ContinuationMorphism readings with the same displayed continuation data. No category-boundary read can leave the transport row H or the ledger rows K and L.

Finally apply Definition 485.19, Definition 485.21, and Definition 485.23. The category boundary is therefore total precisely on the finite route readbacks already determined by Z, and every excluded host or quotient object remains outside the public handoff. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category boundary readback totality`

Theorem 485.27 (ContinuationMonad unit route non-escape)**.** *Every public unit-law read of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet factors through the unit tail* u*, componentwise transport row* H*, continuation route row* K*, ledger row* L*, and local* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row* N*. The read may expose the finite* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert *rows of the packet, but it cannot export host monad equality, a host identity function, or a quotient Kleisli unit.*

Proof. Use carrier obligation to project the unit tail u and the visible package rows. The unit law is therefore attached to the same finite packet as the non-unit continuation tails.

Classifier stability keeps transport through H, and ledger exactness records the unit read through K and L. The public read is a ledger read of the displayed route, not a host equality proof.

The local NameCert package and Kleisli composition boundary give the non-escape conclusion. Since the accepted packet contains no host monad equality coordinate, no host identity-function

coordinate, and no quotient Kleisli unit coordinate, a public unit-law consumer cannot export any of them. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit route nonescape`

Theorem 485.28 (ContinuationMonad category handoff exactness)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet, the* CategoryHomCarrier *and* ContinuationMorphism *readings consume exactly the displayed continuation tails f, g, u together with the* Cont *route row K, componentwise* hsame *row H, ledger row L, package provenance, and local naming row. They do not consume quotient Kleisli arrows, host functions, host monad equalities, or an independent category object.*

Proof. Read the handoff through carrier obligation. The only category-facing rows exposed by the packet are the histories, tails, transport, continuation route, ledger, provenance, and local name.

Use classifier stability and ledger exactness to align the CategoryHomCarrier and ContinuationMorphism readings with the same f, g, u, H, K, L, N route. The category-facing names therefore read the continuation packet rather than a separate host category.

Apply the Kleisli composition boundary and scope non-escape. Together they rule out quotient Kleisli arrows, host functions, host monad equalities, and independent category objects as sources for the handoff. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category handoff exactness`

Theorem 485.29 (ContinuationMonad obligation closure)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *source packet, the carrier source, unit laws, associativity, classifier stability, ledger exactness, local* NameCert *package, and Kleisli composition boundary jointly supply the* obligation *surface for* ContinuationMonad$^{\uparrow}$ *over* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*,* ContinuationMorphism*,* Pkg*, and* NameCert*. The closure surface exports only the displayed continuation tails, route rows, transport rows, ledger rows, package provenance, and naming row.*

Proof. First use the carrier source and carrier obligation. They identify the finite packet rows $A, B, C, f, g, u, H, K, L, N$ and exclude quotient morphisms, host functions, and any general category primitive outside the displayed packet.

Next apply the unit laws and associativity rows. These bind the unit tail and the two-step readback as ledger facts about the displayed Cont routes rather than as host monad equalities.

Use classifier stability and ledger exactness to transport the same route data along componentwise hsame. The CategoryHomCarrier and ContinuationMorphism readings are therefore readings of the same finite continuation rows.

Finally apply the local NameCert package and the Kleisli composition boundary. They repack carrier admission, stability, exactness, provenance, and consumer boundary as the public obligation surface for ContinuationMonad$^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier obligation closure`

Definition 485.30 (ContinuationMonad kernel scope row)**.** *The* ContinuationMonad$^{\uparrow}$ *kernel scope row is the finite list*

BHist, Cont, hsame, CategoryHomCarrier, ContinuationMorphism, Pkg, NameCert.

For a packet $Z = (A, B, C, f, g, u, H, K, L, N)$, every public continuation read must name its source, middle, and target histories, displayed tails, unit tail, transport row, route row, ledger row, package provenance, and local naming row through this list.

Lean4 (definition) `BEDC.Derived.ContinuationMonadUp.ContinuationMonadKernelScopeRow`

Theorem 485.31 (ContinuationMonad route row grounding)**.** *Every route consumer of an accepted* ContinuationMonad$^{\uparrow}$ *packet factors through the finite route data*

$$A, B, C, f, g, u, H, K, L, N.$$

The histories A, B, C, tails f, g, u, componentwise transport row H, continuation route row K, ledger row L, and local naming row N are therefore the complete grounding surface for the kernel scope of Definition 485.30.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier route row grounding`

Proof. Project the accepted source packet. The carrier source lists exactly $A, B, C, f, g, u, H, K, L, N$, and the kernel scope row lists the only BEDC objects through which those coordinates can be read.

Use carrier obligation and classifier stability to bind all transported reads to H and all continuation reads to K. The CategoryUp hom-carrier and continuation-morphism readings are readings of the same displayed tails, not additional route coordinates.

Apply ledger exactness and the Kleisli composition boundary. The ledger L and local name N package the same finite route for downstream consumers, so no consumer can factor through a quotient Kleisli arrow, host function, or unlisted categorical primitive. □

Theorem 485.32 (ContinuationMonad downstream unblock surface)**.** *The scoped downstream unblock surface for* ContinuationMonad$^{\uparrow}$ *consists of carrier admission for $A, B, C, f, g, u, H, K, L, N$, classifier stability under H, ledger exactness through K and L, package provenance through* Pkg*, and the Kleisli boundary recorded by N. Any downstream Real, completion, category, or unary-continuation consumer must use those rows and cannot replace them by quotient morphisms, host functions, or external monad laws.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier downstream unblock surface`

Proof. Carrier admission supplies the finite continuation packet. Classifier stability then keeps transported histories and tails inside the componentwise hsame row H.

Ledger exactness supplies the unit, associativity, and composed-read facts through K and L. These facts are finite continuation rows, not host monad equalities.

Finally the package provenance and Kleisli boundary expose exactly the rows named in the kernel scope. Downstream consumers may depend on that surface, but they receive no route outside $A, B, C, f, g, u, H, K, L, N$. □

Theorem 485.33 (ContinuationMonad route bridge boundary)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N),$$

the bridge from the displayed finite route

$$A \xrightarrow{f} B \xrightarrow{g} C$$

to the standard monad-reading boundary is mediated only by the rows BHist, Cont, hsame, CategoryHomCarrier, ContinuationMorphism, Pkg, *and* NameCert*. The boundary exports no quotient Kleisli arrow, no host function composition, and no host monad equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier route bridge boundary`

Proof. Begin with the public package of Definition 485.9. It supplies carrier admission, classifier stability, ledger exactness, package provenance, and the Kleisli boundary for the same packet Z.

Apply Definition 485.19 to the displayed two-step route. The standard monad-reading boundary can see only the finite continuation route named by $A, B, C, f, g, u, H, K, L, N$.

Finally use Definition 485.21. The CategoryUp hom-carrier and continuation-morphism readings are readings of the same BHist, Cont, and hsame rows, so no quotient arrow, host composition, or host monad equality is exported. □

Theorem 485.34 (ContinuationMonad finite standard handoff)**.** *The public* ContinuationMonad$^{\uparrow}$ *handoff to standard monad notation is finite: the source, middle, and target histories are A, B, C; the displayed non-unit tails are f, g; the unit tail is u; the transport, route, ledger, and naming rows are H, K, L, N. Every standard-looking unit or composite read is a reading of these rows through* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*,* ContinuationMorphism*,* Pkg*, and* NameCert*, not a host-side monad structure.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier finite standard handoff`

Proof. Use Definition 485.20. It identifies every composite consumer with the displayed tails, route row, transport row, ledger row, CategoryUp readings, package provenance, and local certificate.

Next apply Definition 485.21. The standard notation has no separate carrier once the dependency surface has been restricted to the named BEDC rows.

The public package of Definition 485.9 supplies the carrier, stability, exactness, provenance, and boundary obligations for this handoff. Hence the handoff is finite and contains no host monad equality, host function composition, or quotient Kleisli arrow. □

Theorem 485.35 (ContinuationMonad public route bridge totality)**.** *The public* ContinuationMonad$^{\uparrow}$ *route bridge is total on the displayed packet $Z = (A, B, C, f, g, u, H, K, L, N)$. Every downstream Real, completion, category, or unary-continuation consumer that asks for the standard monad-reading boundary receives exactly the finite route surface over* BHist*,* Cont*,* hsame*,* CategoryHomCarrier*,* ContinuationMorphism*,* Pkg*, and* NameCert*; no consumer receives an extra quotient Kleisli arrow, host function composition, or host monad equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier public route bridge totality`

Proof. First apply Definition 485.32. It states that each downstream consumer must use carrier admission, classifier stability, ledger exactness, package provenance, and the Kleisli boundary of the same packet.

Then use Definition 485.33. The route bridge has already identified the only rows through which the standard boundary is readable.

Finally apply Definition 485.34. Since the handoff is finite and contains no host-side monad structure, totality holds for all displayed downstream consumers exactly on the route rows $A, B, C, f, g, u, H, K, L, N$ □

Theorem 485.36 (ContinuationMonad bridge package)**.** *The public* ContinuationMonad$^{\uparrow}$ *packet forms a bridge package from its displayed finite route*

$$A \xrightarrow{f} B \xrightarrow{g} C$$

and unit tail u to the standard monad-reading boundary. The package consists precisely of the BHist *rows A, B, C, the* Cont *route row K, the componentwise* hsame *row H, the* CategoryHomCarrier *and*

ContinuationMorphism *readings, the* Pkg *provenance row, and the* $\mathsf{NameCert}_{\mathsf{ContinuationMonad}^{\uparrow}}$ *row* N*. It exports no quotient Kleisli arrow, no host function composition, and no host monad equality.*

Proof. Start with Definition 485.9. It supplies the public NameCert package containing carrier admission, classifier stability, ledger exactness, package provenance, and the Kleisli composition boundary.

Use Definition 485.19 and Definition 485.20 to bind the standard-looking composite read to the displayed route through f and g, with u as the unit tail inside the same packet.

Apply Definition 485.21 to restrict every dependency to BHist, Cont, hsame, CategoryHomCarrier, ContinuationMorphism, Pkg, and NameCert.

Finally use Definition 485.32 and Definition 485.35. They show that downstream consumers receive the complete bridge package and no quotient Kleisli arrow, host function composition, or host monad equality. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bridge package`

Theorem 485.37 (ContinuationMonad bind-route coverage)**.** *Every accepted bind read of a* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet factors through the displayed source* BHist *row, the continuation route row, the Kleisli composite row, the componentwise* hsame *transport, the* Pkg *provenance row, and the local* NameCert *row. Thus a bind consumer can inspect only the rows* $A, B, C, f, g, u, H, K, L, N$ *named by the bridge package, where* A, B, C *are the* BHist *sources,* K *is the* Cont *route,* L *is the composite ledger,* H *is the* hsame *transport, and* N *is the local* NameCert *certificate with* Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bind route coverage`

Proof. Begin with the unit laws and associativity rows. They identify the accepted bind read as a replay of the displayed unit tail and the two-step Kleisli composite rather than as a host monad operation.

Next use the scoped route dependencies. The only histories visible to the read are the BHist rows A, B, C, and the only continuation data are the displayed Cont route K and ledger L.

Transport is then forced through the componentwise hsame row H. The route bridge boundary and bridge package attach the same read to Pkg provenance and the local NameCert row N, so no hidden evaluator, quotient Kleisli arrow, or host function can supply an additional bind route. □

Theorem 485.38 (ContinuationMonad Kleisli ledger associativity)**.** *Two nested Kleisli composites in an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet have the same consumer-visible ledger exactly when both composites are replayed through the existing associativity row and the same scoped-route dependencies over* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert*. The comparison is a ledger comparison inside* L*, not an equality supplied by a hidden evaluator or by host-side monad laws.*

Proof. For the forward direction, read the common consumer-visible ledger through the bridge package. The package exposes the source BHist rows, the Cont route row, the hsame transport, the Pkg provenance row, and the local NameCert row, so the common ledger must be the displayed row L.

Apply the associativity row to replay the two nested composites against that same row L. The scoped route dependencies keep both replays inside the route rows $A, B, C, f, g, u, H, K, L, N$.

For the reverse direction, assume both nested composites replay through associativity and the scoped-route dependencies. Since the route bridge boundary exports only the displayed ledger and no host evaluator, the two consumer-visible ledgers coincide exactly as reads of L. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier kleisli ledger associativity`

Theorem 485.39 (ContinuationMonad downstream consumer exhaustion)**.** *Any downstream CategoryUp or generator-closure consumer of the* ContinuationMonad$^{\uparrow}$ *bridge package can inspect only the unit, bind, route, ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already displayed in this chapter. In particular, every such consumer is exhausted by the rows* u, f, g, K, L, H, N *over the source* BHist *rows* A, B, C *with* Pkg *provenance and receives no quotient Kleisli category, ambient evaluator, host function composition, or extra monad law.*

Proof. First expose the consumer through the bridge package. The package has already bound the standard-looking route to the displayed BHist, Cont, hsame, Pkg, and NameCert rows.

Next apply the unit laws, associativity, scoped route dependencies, and route bridge boundary. These rows supply the unit read, bind read, route replay, and ledger comparison by finite continuation data.

The downstream CategoryUp and generator-closure consumers therefore have no remaining coordinate to inspect. Any extra evaluator, quotient Kleisli category, host function composition, or independent monad law would have to be a row outside $A, B, C, f, g, u, H, K, L, N$ and the Pkg provenance surface, which the bridge package does not export. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier downstream consumer exhaustion`

Theorem 485.40 (ContinuationMonad generator-category consumer triad)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every downstream consumer that jointly reads the generator-closure refusal surface and the* Cat$^{\uparrow}$ *hom route factors through exactly the displayed rows*

$A, B, C, f, g, u, H, K, L, N$, BHist, Cont, hsame, Pkg, NameCert, CategoryHomCarrier, ContinuationMo

The triad consists of the source histories A, B, C*, the continuation tails* f, g, u *with route and ledger rows* K, L*, and the local transport, package, and name rows* H, N*. It exports no quotient Kleisli arrow, host function, or host monad equality.*

Proof. Project the carrier Z. The histories A, B, C and tails f, g, u are the only source and route rows available to a downstream reader.

Route the CategoryUp read through the displayed CategoryHomCarrier and ContinuationMorphism entries in L. These entries read the same finite continuation tails and do not introduce an independent category object.

Route the generator consumer through the refusal surface named by the packet. The refusal surface admits the listed continuation rules and rejects any unlisted continuation rule before the consumer reaches K, L, H, N.

Finally combine the package row with the local NameCert row. Since no coordinate outside $A, B, C, f, g, u, H, K, L, N$ is exported, quotient Kleisli arrows, host functions, and host monad equalities have no consumer-facing route. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator category consumer triad`

Theorem 485.41 (ContinuationMonad generator refusal row)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every generator-closure consumer of the unit or bind surface reads the authorized continuation rows* f, g, u, K, L *through the same refusal ledger before it can inspect* Pkg *provenance or the local* NameCert *row. Any requested continuation rule not displayed by* $A, B, C, f, g, u, H, K, L, N$ *is refused at that ledger and cannot be supplied by a quotient Kleisli arrow, host function, or host monad equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator refusal certified row`

Proof. Start from the displayed unit and bind rows. The unit tail is u, the non-unit tails are f, g, and the accepted route and ledger are K, L.

Apply the generator-closure refusal surface. It separates authorized displayed continuation rows from unlisted continuation rules before the route reaches the package and local naming coordinates.

Use componentwise hsame transport only through H. The transport preserves the same refusal ledger and does not add a route for a hidden evaluator or a quotient Kleisli arrow.

The Pkg and NameCert rows therefore certify exactly the displayed generator-facing surface, with no host function or host monad equality available to replace the finite route. □

Theorem 485.42 (ContinuationMonad Category hom route totality)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every* Cat$^{\uparrow}$ *consumer of* CategoryHomCarrier *or* ContinuationMorphism *is total on the displayed route* $A, B, C, f, g, u, H, K, L, N$*. The consumer reads the hom surface only through* BHist*,* Cont*,* hsame*,* Pkg*,* NameCert*, the* CategoryHomCarrier *row, and the* ContinuationMorphism *row, and it receives no independent category object, quotient Kleisli arrow, host function, or host monad equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category hom route totality`

Proof. Project the CategoryUp-facing carrier rows. The hom carrier and continuation morphism readings are stored in the same finite ledger L that records the accepted route through f, g, u and K.

Use classifier stability to transport this read through H. The transported hom route still has endpoints A, B, C and still reads only the displayed continuation rows.

Package provenance and the local NameCert row then close the route for every CategoryUp consumer. Because the packet has no coordinate for an independent category object or host evaluator, the displayed hom route is total on exactly the finite packet. □

Theorem 485.43 (ContinuationMonad category-generator source triad)**.** *Every accepted* ContinuationMonad$^{\uparrow}$ *packet exposes the* Cat$^{\uparrow}$ *hom-carrier row, the* ContinuationMorphism *row, and the generator-closure refusal row before either the public unit read or the public bind read is available. The exposed triad is read from* $A, B, C, f, g, u, H, K, L, N$ *over* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert*, and not from an independent category object, quotient Kleisli arrow, host function space, or unlisted generator family.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category generator source triad`

Proof. Begin with carrier admission and the root generator obligation. The carrier obligation identifies $A, B, C, f, g, u, H, K, L, N$ as the available rows, while the root generator obligation places the refusal surface before any generator-facing read can use unit or bind.

Next apply bind-route classifier stability and the Category hom route totality theorem. These results route the Cat$^{\uparrow}$ hom-carrier and ContinuationMorphism entries through the same finite ledger L and the same transport row H.

The unit and bind reads therefore come after the triad is already visible. Since the packet has no coordinate for an independent category object, quotient Kleisli arrow, host function space, or unlisted generator family, no downstream reader can bypass the displayed triad. □

Theorem 485.44 (ContinuationMonad bind readback exhaustion)**.** *Every bind readback exposed by an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet is exhausted by the rows* $A, B, C, f, g, u, H, K, L, N$*. The readback may inspect the source histories* A, B, C*, the displayed continuation tails* f, g, u*, the transport row* H*, the route and ledger rows* K, L*, the* Pkg *provenance, and the local* NameCert *row* N*; it cannot read host functions, quotient Kleisli arrows, unlisted generators, or an external monad evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bind readback exhaustion`

Proof. First use bind-route classifier stability. It identifies every accepted bind read with the displayed continuation route through f, g, u, K, L and transports that read only through H.

Then use NameCert ledger exactness. The ledger records exactly the finite rows $A, B, C, f, g, u, H, K, L, N$ with Pkg provenance, so any consumer-visible bind readback must be a readback of that finite packet.

Finally apply the Kleisli composite scope and bind-route coverage theorems. They prevent a host function, quotient Kleisli arrow, unlisted generator, or external evaluator from supplying another bind readback outside the listed rows. □

Theorem 485.45 (ContinuationMonad root-unblock package)**.** *The root obligation package of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet consists exactly of carrier admission, classifier stability,* $\mathsf{Cat}^{\uparrow}$ *hom routing, generator refusal, ledger exactness, and the local* NameCert *rows over* $A, B, C, f, g, u, H, K, L, N$*. This package is the surface available to downstream continuation chapters before they read any standard monad boundary, and it exports no quotient Kleisli category, host function composition, ambient effect algebra, or independent generator list.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root unblock package`

Proof. Assemble the package from the existing obligation rows. Carrier admission supplies the packet rows, classifier stability fixes their hsame transport, category hom routing places the $\mathsf{Cat}^{\uparrow}$ read in the finite ledger, and generator refusal excludes unlisted continuation rules.

Add NameCert ledger exactness and Kleisli composite scope. These rows make the visible ledger exact for $A, B, C, f, g, u, H, K, L, N$ and keep every composite read inside the same finite route.

The resulting package is precisely the root obligation surface available to downstream continuation chapters. Any quotient Kleisli category, host function composition, ambient effect algebra, or independent generator list would require a row outside the displayed carrier, and no such row is exported. □

Theorem 485.46 (ContinuationMonad root unit-bind exhaustion)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every root unit read and root bind read is exhausted by the displayed rows* u, f, g, K, L, H, N *together with the source histories* A, B, C*. The unit and bind consumers may inspect only the unit tail, the two continuation tails, componentwise transport, the route and ledger rows, package provenance, and the local* NameCert *row; they receive no quotient Kleisli arrow, host function composition, or host monad equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root unit bind exhaustion`

Proof. Start with Definition 485.3 and Definition 485.4. They place the unit and bind laws inside the finite route named by u, f, g, K, L.

Use Definition 485.8 and Definition 485.37. The ledger reads those rows through the same accepted packet and accounts for every public unit or bind request.

Finally apply Definition 485.21 and Definition 485.27. The only remaining transports are the displayed H rows and the local certificate N, so the host and quotient readings have no exported route. □

Theorem 485.47 (ContinuationMonad root generator-category lock)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, generator-closure reads and* $\mathsf{Cat}^{\uparrow}$ *hom reads are locked to the same finite route* $A, B, C, f, g, u, H, K, L, N$*. A downstream consumer cannot separate the generator refusal ledger from the* CategoryHomCarrier *and* ContinuationMorphism *rows, and it cannot replace them by a quotient Kleisli category, an ambient category object, or host-function composition.*

Proof. First apply Definition 485.41. The generator surface reads the continuation tails and route rows through the same refusal ledger before it reaches package provenance or local naming.

Next apply Definition 485.42. The $\mathsf{Cat}^{\uparrow}$ hom carrier and continuation-morphism entries are total exactly on the displayed route and do not supply an independent category object.

The two reads are then combined by Definition 485.43. Since both sides use the same H, K, L, N rows over A, B, C, f, g, u, no downstream consumer can split the generator ledger from the Category hom route. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator category lock`

Theorem 485.48 (ContinuationMonad root downstream unblock package)**.** *The root downstream surface of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet is the package formed by Definition 485.46 and Definition 485.47. Downstream continuation chapters receive the finite rows* $A, B, C, f, g, u, H, K, L, N$ *as the complete unit, bind, generator-refusal, and Category hom route surface, and they receive no quotient Kleisli arrows, host-function composition, ambient category completion, or unlisted continuation generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream unblock package`

Proof. Use Definition 485.46 to exhaust the unit and bind consumers by the finite rows of Z.

Use Definition 485.47 to attach the generator-refusal and Category hom reads to the same rows. This supplies the generator and category boundary without adding a second route.

Finally apply Definition 485.39 and Definition 485.45. These theorems identify the downstream consumer surface with the finite root obligation package, so every excluded host, quotient, category-completion, or unlisted-generator object remains outside the packet. □

Theorem 485.49 (ContinuationMonad finite bind readback package)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every finite bind readback package is exactly the* BHist *source rows* A, B, C*, the continuation rows* f, g, u*, the* Cont *route row* K*, the ledger row* L*, the componentwise* hsame *row* H*, the* Pkg *provenance, and the local* NameCert *row* N*. A bind consumer reads this package as displayed BEDC data and not as host-function composition, quotient Kleisli equality, a general monad evaluator, or an unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier finite bind readback package`

Proof. First read bind-route totality. It forces every bind consumer through the displayed Cont route and ledger rows K, L over the same source histories and continuation tails.

Next read bind readback exhaustion. The readback has already been exhausted by $A, B, C, f, g, u, H, K, L, N$, with componentwise transport supplied only by H and with no external evaluator row.

The root downstream package then supplies the Pkg provenance and local NameCert wrapper. Since the package contains exactly the displayed carrier, route, ledger, and naming rows, the finite bind readback cannot be replaced by a host, quotient, or unlisted-generator reading. □

Theorem 485.50 (ContinuationMonad downstream route exhaustion)**.** *Every downstream route exported by an accepted* ContinuationMonad$^{\uparrow}$ *packet is exhausted by the root package rows* $A, B, C, f, g, u, H, K, L, N$ *over* BHist*,* Cont*, componentwise* hsame*,* Pkg*, and the local* NameCert *row. Unit reads, bind reads, generator-boundary reads, and* Cat$^{\uparrow}$ *hom reads all factor through that same package, so no downstream continuation chapter receives a host function, quotient Kleisli arrow, ambient category primitive, or continuation rule outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier downstream route exhaustion`

Proof. Begin with the root downstream package. It identifies the downstream surface with the carrier, root generator obligation, bind-route totality, ledger exactness, Pkg provenance, and local NameCert rows of the accepted packet.

Use generator-category lockdown to bind the generator and Cat$^{\uparrow}$ hom reads to the same route. Use the finite bind readback package to bind unit and bind consumers to the same K, L, H, N rows.

It remains only to account for downstream escape routes. The root downstream package already excludes host functions, quotient Kleisli arrows, ambient category completion, and unlisted continuation generators, so the displayed $A, B, C, f, g, u, H, K, L, N$ packet exhausts every downstream route. □

Closure status for NameCert$_{\text{ContinuationMonad}^{\uparrow}}$.

Constructive story: A ContinuationMonad$^{\uparrow}$ packet is generated from source, middle, and target histories, two displayed continuation tails, a unit tail, componentwise hsame transports, displayed Cont routes, CategoryUp continuation-morphism readings, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{ContinuationMonad}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 485.35 and Definition 485.36 expose the public bridge package from the finite continuation route rows $A, B, C, f, g, u, H, K, L, N$ to the standard monad-reading boundary over BHist, Cont, hsame, CategoryHomCarrier, ContinuationMorphism, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{ContinuationMonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full category theory, quotient Kleisli categories, host functions, standard monad laws as host equalities, arbitrary effects, or bridge checking.

Upgrade path: The theory closure is complete for the finite ContinuationMonad$^{\uparrow}$ bridge package; Lean verification and bridge checking remain separate verification-axis obligations.

Theorem 485.51 (ContinuationMonad root category-generator factorization)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the displayed continuation tails factor every root category-generator read through one finite surface. The* CategoryHomCarrier *and* ContinuationMorphism *rows are read from* f, g, u, K, L, H, N*, while the* GeneratorClosure$^{\uparrow}$ *refusal surface accepts no unit, bind, or continuation generator outside the same packet.*

Thus a downstream root consumer may combine the category face and the generator face only by reading $A, B, C, f, g, u, H, K, L, N$ *over* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert*.*

Proof. Begin with Definition 485.56. It ties the Cat$^{\uparrow}$ hom-carrier surface and the generator-boundary refusal surface to the same source histories, continuation tails, unit tail, and structural rows.

Next use Definition 485.65. The exhaustion theorem identifies every generator-facing or CategoryHomCarrier-facing read with the displayed continuation tails and rejects independent hom carriers, ambient hom sets, host functions, and quotient Kleisli arrows.

Finally apply Definition 485.47. The lock prevents a downstream consumer from separating the generator refusal ledger from the CategoryHomCarrier and ContinuationMorphism rows, so the factorization is exactly the finite packet surface stated above. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root category generator factorization`

Theorem 485.52 (ContinuationMonad root bind-unit boundary)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every public unit read and every public bind read requires the displayed rows* u, f, g, H, K, L, N *together with the source histories* A, B, C*. The unit and bind boundary has no coordinate for host function composition, host monad equality, quotient Kleisli arrows, or external evaluator rows.*

Proof. Start with Definition 485.46. It exhausts every root unit read and root bind read by u, f, g, K, L, H, N together with A, B, C.

Use Definition 485.49 to package the bind side as finite BEDC data. The package records the histories, continuation rows, route row, ledger row, transport row, provenance, and local certificate, not a host-side bind operation.

Finally use Definition 485.27. The unit side also stays inside the displayed continuation route, so public unit and bind reads both require the same finite packet rows and cannot be supplied by host or quotient structure. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root bind unit boundary`

Theorem 485.53 (ContinuationMonad root downstream surface exactness)**.** *The downstream root surface of an accepted* ContinuationMonad$^{\uparrow}$ *packet is exactly the finite package assembled from the category row, generator refusal row, carrier rows, classifier stability, ledger exactness, package provenance, and local* NameCert *row over* $A, B, C, f, g, u, H, K, L, N$*. It exports no host function composition, quotient Kleisli arrow, ambient hom set, independent category carrier, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream surface exactness`

Proof. Use Definition 485.48 to identify the downstream surface with the root unit-bind exhaustion and the root generator-category lock.

Add the factorization theorem above. It supplies the category-generator side through CategoryHomCarrier, ContinuationMorphism, and the GeneratorClosure$^{\uparrow}$ refusal surface on the same finite rows.

Add the bind-unit boundary above. It supplies the public unit and bind side through u, f, g, H, K, L, N and the source histories A, B, C.

The carrier, classifier, and ledger rows are already exact by Definition 485.6, Definition 485.7, and Definition 485.8. Repacking these rows with Pkg provenance and the local NameCert row leaves no exported coordinate for host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, or unlisted generators. □

Theorem 485.54 (ContinuationMonad root public export)**.** *Every public root export of an accepted* ContinuationMonad$^{\uparrow}$ *packet factors through the combined finite rows*

$$A, B, C, f, g, u, H, K, L, N.$$

On this single BEDC route, category-generator reads, public unit reads, public bind reads, carrier obligations, classifier stability, ledger exactness, package provenance, and local NameCert *naming are all read from the same packet. The export contains no host function composition, quotient Kleisli arrow, ambient hom set, independent category carrier, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public export`

Proof. First assemble the root category-generator side. Apply Definition 485.51 to put the CategoryHomCarrier, ContinuationMorphism, and GeneratorClosure$^{\uparrow}$ refusal surfaces on $A, B, C, f, g, u, H, K, L, N$.

Next assemble the public unit and bind side. Apply Definition 485.52; every public unit read and public bind read uses u, f, g, H, K, L, N together with the source histories A, B, C.

Now use Definition 485.53. It identifies the downstream root surface with the carrier rows, classifier stability, ledger exactness, package provenance, and local NameCert surface over the same finite packet.

Finally cite the carrier, classifier, and ledger theorems: Definition 485.6, Definition 485.7, and Definition 485.8. These rows leave no public coordinate for host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, or unlisted continuation generators, so the displayed route is the public root export. □

Lemma 485.55 (ContinuationMonad root category-generator bridge determinacy)**.** *Let* $Z = (A, B, C, f, g, u, H, K, L, N)$ *be an accepted* ContinuationMonad$^{\uparrow}$ *packet. Every consumer-facing use of the root category-generator bridge, whether it is presented as a displayed route, componentwise transport, public readback, or downstream handoff, is determined by the same finite rows*

$$A, B, C, f, g, u, H, K, L, N.$$

The category side may read only the CategoryHomCarrier *and* ContinuationMorphism *rows exposed by the displayed continuation tails and route ledger; the generator side may read only the* GeneratorClosure$^{\uparrow}$ *refusal surface attached to those rows; and the downstream side may read only the* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert *coordinates already carried by the packet. No bridge coordinate remains for host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, external evaluators, detached handoff witnesses, or unlisted continuation generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root category generator bridge determinacy`

Proof. Apply Definition 485.51. It places every root category-generator read on one finite surface: the CategoryHomCarrier and ContinuationMorphism rows are read from f, g, u, K, L, H, N, and the GeneratorClosure$^{\uparrow}$ refusal surface accepts no generator outside the same packet. Hence any consumer route that touches the category face or the generator face has already factored through $A, B, C, f, g, u, H, K, L, N$.

Now apply Definition 485.53. The downstream root surface is exactly the package assembled from the category row, generator refusal row, carrier rows, classifier stability, ledger exactness, package provenance, and local NameCert row over the same displayed packet. Thus a transport use is the componentwise hsame row already present in H, a route or replay use is one of the displayed Cont rows carried by f, g, u and K, a readback use is one of the listed carrier, classifier, ledger, category, or generator-refusal rows, and a handoff use is the Pkg provenance together with the local naming row N.

These two exactness statements have the same source rows and the same refusal boundary. Therefore a consumer cannot vary the bridge by changing only an unlisted coordinate: any such

variation would have to be a host composition row, quotient Kleisli arrow, ambient hom set, independent category carrier, external evaluator, detached handoff witness, or unlisted generator, and Definition 485.53 states that none of those coordinates is exported. The bridge is therefore determined by the displayed finite packet. □

Theorem 485.56 (ContinuationMonad generator category triad)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the source histories* A, B, C*, continuation tails* f, g, u*, and structural rows* H, K, L, N *form one generator-category triad. The* $\mathsf{Cat}^{\uparrow}$ *hom-carrier surface reads the displayed tails as* $\mathsf{CategoryHomCarrier}$ *and* $\mathsf{ContinuationMorphism}$ *rows, while the generator-boundary refusal surface of chapter 486 refuses every unit, bind, or continuation generator not already displayed in* Z*.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator category triad`

Proof. Start with carrier admission and classifier stability from Definition 485.6 and Definition 485.7. These rows expose $A, B, C, f, g, u, H, K, L, N$ as one finite packet and keep the transported continuation tails on the same histories.

Next read the category side through chapter 61 and Definition 485.16. The hom-carrier and continuation-morphism entries are readings of the displayed tails and ledger, not independent category data.

Finally read the generator side through chapter 486 and Definition 485.17. Since the refusal row accepts only the listed continuation rules, the category and generator faces meet on the same $A, B, C, f, g, u, H, K, L, N$ packet. □

Theorem 485.57 (ContinuationMonad root bind schedule exhaustion)**.** *Let* $Z = (A, B, C, f, g, u, H, K, L, N)$ *be an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet. Every root-level unit or bind schedule read of* Z *is exhausted by the displayed unit tail* u*, continuation tails* f, g*, componentwise transports* H*, route rows* K*, ledger rows* L*, and local name* N*. A downstream read cannot replace this schedule by host function composition, a quotient Kleisli arrow, an ambient hom set, or an unlisted generator rule.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root bind schedule exhaustion`

Proof. First apply Definition 485.3 and Definition 485.4. The unit and bind laws are ledger statements about u, f, g, K, L inside the accepted packet.

Use Definition 485.11 and Definition 485.12 to keep every transported bind read on the same H, K, L, N rows. The schedule is therefore the displayed continuation ledger, not a host composition operation.

Finally attach Definition 485.56. Its category and generator faces account for the only external readings of the schedule, and the refusal surface blocks every unlisted continuation rule. □

Theorem 485.58 (ContinuationMonad downstream unblock route package)**.** *The generator-category triad of Definition 485.56 and the bind schedule exhaustion of Definition 485.57 form the downstream unblock route package for* $\mathsf{ContinuationMonad}^{\uparrow}$*. Any downstream consumer receives exactly* $A, B, C, f, g, u, H, K, L, N$*, the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier reading of those rows, and the generator-boundary refusal of unlisted continuation rules.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier downstream unblock route package`

Proof. Use Definition 485.56 to expose the shared source. It binds the three histories, the continuation tails, the CategoryUp hom-carrier surface of chapter 61, and the generator-boundary refusal surface of chapter 486 to one packet.

Use Definition 485.57 to exhaust unit and bind reads. Every root schedule read factors through u, f, g, H, K, L, N before it can be consumed downstream.

Finally apply Definition 485.39 and Definition 485.45. They identify the downstream surface with the same finite carrier, route, ledger, package, and local naming rows, so no host function composition, quotient Kleisli category, ambient hom set, or unlisted generator is exported. □

Theorem 485.59 (ContinuationMonad formal target handoff)**.** *Any later Lean formal target for* $\mathsf{ContinuationMonad}^{\uparrow}$ *must read the same finite packet*

$$Z = (A, B, C, f, g, u, H, K, L, N)$$

through BHist, hsame, Cont, Pkg, $\mathsf{Cat}^{\uparrow}$, *and* $\mathsf{GeneratorClosure}^{\uparrow}$. *The target may project the packet into carrier, route, ledger, package, hom-carrier, or generator-refusal faces, but it may not introduce a second continuation source or a detached category-generator witness.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category generator formal handoff`

Proof. Start from Definition 485.56. It identifies the CategoryUp hom-carrier reading and the GeneratorClosureUp refusal reading with the displayed histories, continuation tails, and structural rows of Z.

Next use Definition 485.57. The root unit and bind schedule is exhausted by u, f, g, H, K, L, N, so any formal target that reads a continuation law must pass through the same BHist, hsame, Cont, and Pkg rows.

Finally apply Definition 485.58. It packages the category and generator readings for downstream consumers, leaving no coordinate for a second source, detached witness, or independent formal target carrier. □

Theorem 485.60 (ContinuationMonad Kleisli quotient refusal)**.** *No accepted read of a* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet may replace the displayed continuation tails* f, g, u *and rows* H, K, L, N *by quotient Kleisli arrows, ambient host composition, an unlisted hom set, or a host category law. Every accepted category or generator reading remains a read of the displayed BEDC packet* $Z = (A, B, C, f, g, u, H, K, L, N)$.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier kleisli quotient refusal`

Proof. Use Definition 485.57 first. It refuses host function composition, quotient Kleisli arrows, ambient hom sets, and unlisted generator rules at the root schedule boundary.

Then apply Definition 485.56. The CategoryUp side reads the displayed tails as hom-carrier and continuation-morphism rows, while the GeneratorClosureUp side refuses every generator not already displayed in Z.

The downstream package of Definition 485.58 carries the same refusal to every consumer. Thus a quotient or host-composition replacement has no accepted row through which it can enter. □

Theorem 485.61 (ContinuationMonad category generator exactness)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$, *the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier face and the* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal face are exact on the same displayed rows. The category face reads precisely the continuation tails and transport ledger exposed by* Z, *and the generator face refuses precisely the unit, bind, and continuation generators not displayed by those rows.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category generator exactness`

Proof. Apply Definition 485.56. It already places the CategoryUp hom-carrier surface and the GeneratorClosureUp refusal surface on the same $A, B, C, f, g, u, H, K, L, N$ packet.

Use Definition 485.57 to identify which unit and bind reads are available. The schedule is the displayed continuation ledger, so exactness is measured against u, f, g, H, K, L, N rather than against host composition.

Finally use Definition 485.58. The downstream route exports exactly the category hom-carrier reading and generator refusal reading of the same packet, which proves that the two faces are exact on the displayed rows. □

Theorem 485.62 (ContinuationMonad bind-Cont unit coherence)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the unit tail* u*, bind tails* f, g*,* $\mathsf{Cat}^{\uparrow}$ *hom-carrier face, and* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal face are coherent exactly when every displayed bind read is a* Cont*-route read through* H, K, L, N*. The packet exports no quotient Kleisli arrow, host function composition, ambient hom set, or detached unit law beyond those displayed routes.*

Proof. Start with root bind schedule exhaustion. It says that every root unit or bind schedule read is exhausted by u, f, g, H, K, L, N and refuses host function composition, quotient Kleisli arrows, ambient hom sets, and unlisted generator rules.

Apply the generator-category triad. The $\mathsf{Cat}^{\uparrow}$ hom-carrier face and the $\mathsf{GeneratorClosure}^{\uparrow}$ refusal face are readings of the same displayed continuation packet, so the category and generator sides do not introduce an extra bind source.

Use category-generator exactness and the downstream unblock route package. They identify the exported downstream surface with the same hom-carrier reading, generator refusal reading, and H, K, L, N route rows. Hence bind-Cont unit coherence is exactly the condition that displayed bind reads pass through those Cont routes. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier bind cont unit coherence`

Theorem 485.63 (ContinuationMonad generator category public factorization)**.** *Every public read of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet factors first through the displayed packet*

$$Z = (A, B, C, f, g, u, H, K, L, N),$$

then through the $\mathsf{Cat}^{\uparrow}$ *hom-carrier face and the* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal face. The exported public surface contains no quotient Kleisli arrow, host composition operation, detached category witness, or generator not listed by* Z*.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator category public factorization`

Proof. Start with category-generator exactness, Definition 485.61. It places the $\mathsf{Cat}^{\uparrow}$ hom-carrier face and the $\mathsf{GeneratorClosure}^{\uparrow}$ refusal face on the same displayed $A, B, C, f, g, u, H, K, L, N$ rows.

Use bind-Cont unit coherence, Definition 485.62. Every displayed bind read passes through the Cont route rows H, K, L, N, and the packet exports no quotient Kleisli arrow, host function composition, ambient hom set, or detached unit law outside those routes.

Finally apply Definition 485.58. It packages the category hom-carrier reading and generator-boundary refusal for every downstream consumer. Hence a public read factors through Z and then through the two displayed faces, with no detached category witness or unlisted generator available to the route. □

Theorem 485.64 (ContinuationMonad triad unit route package)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the triad rows* A, B, C*, the continuation tails* f, g*, the unit tail* u*, and the rows* H, K, L, N *are the complete unit-and-bind route package. The unit route is the displayed* u *row, the bind route is the displayed read through* f, g, K, L*, and the local readback has no coordinate for host function composition, quotient Kleisli arrows, or a general category primitive.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root triad unit route package`

Proof. Start with carrier admission from Definition 485.6. It exposes exactly the three histories, the two continuation tails, the unit tail, the transport row, the route row, the ledger row, and the local naming row.

Apply the unit laws of Definition 485.3 and the associativity theorem of Definition 485.4. The unit route and bind read are therefore ledger routes over u, f, g, K, L, transported componentwise by H.

Use classifier stability and ledger exactness from Definition 485.7 and Definition 485.12. These facts exhaust the consumer-visible route by the displayed packet, so no host function, quotient Kleisli arrow, or external category primitive remains available. □

Theorem 485.65 (ContinuationMonad generator hom route exhaustion)**.** *Every generator-facing or* $\mathsf{CategoryHomCarrier}$*-facing read of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet is exhausted by the displayed continuation tails* f, g, u*, the generator-boundary surface, and the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier surface. The read factors through* A, B, C, H, K, L, N *and cannot introduce an unlisted continuation rule, independent hom carrier, ambient hom set, host function, or quotient Kleisli arrow.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator hom route exhaustion`

Proof. First read the generator boundary through Definition 485.17. It refuses every unit row, bind row, continuation rule, or route generator not displayed in the packet.

Next read the category-facing surface through Definition 485.16. The $\mathsf{CategoryHomCarrier}$ and $\mathsf{ContinuationMorphism}$ entries are exposed only as reads of the displayed tails over A, B, C, H, K, L, N.

Finally combine ledger exactness from Definition 485.12 with the Kleisli boundary of Definition 485.19. The generator and hom-carrier readings are therefore exhausted by the continuation rows already present in the packet, with no unlisted rule or external hom object left to read. □

Theorem 485.66 (ContinuationMonad root downstream readback package)**.** *The rows displayed in Definition 485.64 and Definition 485.65 form the downstream root-unblock surface for* $\mathsf{ContinuationMonad}^{\uparrow}$*. A downstream consumer receives precisely* $A, B, C, f, g, u, H, K, L, N$*, the generator-boundary refusal of unlisted continuation rules, and the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier reading of the displayed tails.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream readback package`

Proof. Use the triad unit route package to expose the local source. The downstream surface begins with $A, B, C, f, g, u, H, K, L, N$ and reads unit and bind only through the displayed continuation routes.

Use the generator hom route exhaustion to expose the two external faces. The generator boundary refuses unlisted rules, while the CategoryUp surface reads the hom-carrier and continuation-morphism rows from the same displayed tails.

Together these rows are exactly the carrier, unit, associativity, classifier stability, ledger exactness, generator boundary, category hom exposure, and local NameCert data cited in the core chapter. Since each downstream face is already accounted for by those rows, the readback package exports no host function composition, quotient Kleisli category, ambient hom set, or additional continuation generator. □

Theorem 485.67 (ContinuationMonad root route source exposure)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the root route exposes its source only through the displayed histories* A, B, C*, the continuation tails* f, g*, the unit tail* u*, the transport row* H*, the route row* K*, the ledger row* L*, and the local name row* N*. Any downstream root consumer reads the source through these history, tail, and unit rows before it can read the generator-boundary or* $\mathsf{Cat}^{\uparrow}$ *hom surfaces.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root route source exposure`

Proof. Start with carrier admission from Definition 485.6. It displays exactly $A, B, C, f, g, u, H, K, L, N$ as the accepted source packet and leaves no coordinate for an external source object.

Apply the root route packages already present in Definition 485.58 and Definition 485.66. They route downstream reads through the same packet before the consumer reaches the generator-boundary refusal or the $\mathsf{Cat}^{\uparrow}$ hom-carrier surface.

Finally use Definition 485.16 and Definition 485.17. These theorems read category and generator data from the exposed rows rather than from an independent host source, so the root route source is precisely the displayed history, tail, unit, transport, route, ledger, and local-name surface. □

Theorem 485.68 (ContinuationMonad root continuation rule coverage)**.** *Let* $Z = (A, B, C, f, g, u, H, K, L, N)$ *be an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet. Every root unit read and every root bind read of* Z *is covered by the rows* K, H, L, N *together with the displayed tails* u, f, g *and histories* A, B, C*. The unit laws, associativity read, route provenance, componentwise transport, ledger accounting, and local naming all remain inside this finite row package.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root continuation rule coverage`

Proof. Use Definition 485.3 for unit reads and Definition 485.4 for bind reads. Both theorems state their laws as ledger reads over the displayed unit and continuation tails.

Next apply Definition 485.64 and Definition 485.57. They identify the route row K, transport row H, ledger row L, and local name row N as the finite coverage data for the public unit and bind schedules.

The root downstream packages then provide the handoff to consumers. Since Definition 485.66 and Definition 485.58 expose no rule outside $A, B, C, f, g, u, H, K, L, N$, every unit or bind read is covered by the named rows. □

Theorem 485.69 (ContinuationMonad root downstream consumer separation)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, downstream root consumers are separated from host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, and unlisted continuation generators. Their only available reads are the displayed source rows, unit and bind route rows, the generator-boundary refusal, and the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier exposure of the same packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream consumer separation`

Proof. Begin with Definition 485.17. It refuses every unit row, bind row, continuation rule, or route generator not displayed in the accepted packet.

Read the category side through Definition 485.16. The CategoryHomCarrier and ContinuationMorphism rows are exposed only as readings of f, g, u over A, B, C, H, K, L, N.

Finally combine the two root-unblock child packages, Definition 485.58 and Definition 485.66. They identify the downstream surface with the finite root route package, so host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, and unlisted generators have no consumer-visible coordinate. □

Theorem 485.70 (ContinuationMonad root consumer standard row)**.** *Every root consumer standard row of an accepted* ContinuationMonad$^{\uparrow}$ *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N)$$

is exhausted by the displayed histories A, B, C, continuation tails f, g, unit tail u, transport row H, route row K, ledger row L, and local name row N. Unit reads and bind reads are covered by the root continuation rule package over these rows, while category-facing and generator-facing reads remain separated from host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, and unlisted generators.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root consumer standard row`

Proof. First apply Definition 485.67. It identifies the root source surface with exactly $A, B, C, f, g, u, H, K, L, N$ and leaves no coordinate for an external source object.

Next use Definition 485.68. Unit and bind reads are already covered by the displayed tails, histories, transport, route, ledger, and local naming rows, so the standard row cannot import a second continuation rule package.

Finally apply Definition 485.69. The downstream separation theorem blocks host function composition, quotient Kleisli arrows, ambient hom sets, independent category carriers, and unlisted generators. Therefore the root consumer standard row is precisely the displayed finite packet. □

Theorem 485.71 (ContinuationMonad root formal target surface)**.** *For an accepted* ContinuationMonad$^{\uparrow}$ *packet $Z = (A, B, C, f, g, u, H, K, L, N)$, the formal target surface available at the root consists exactly of the finite rows $A, B, C, f, g, u, H, K, L, N$ together with their displayed* BHist*,* Cont*,* hsame*,* Pkg*,* Cat$^{\uparrow}$ *hom-carrier,* ContinuationMorphism*, and local* NameCert *readings. Downstream formal targets may consume unit, bind, generator-refusal, and Category hom data only through that surface, not through quotient Kleisli arrows, host-function composition, ambient category completion, or unlisted continuation generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal target surface`

Proof. Use the root unit-bind exhaustion theorem to account for the unit and bind reads. It restricts those reads to the displayed source histories, continuation tails, unit tail, route rows, ledger rows, transport row, provenance, and local naming row.

Use the root generator-category lock for the generator-refusal and Category hom faces. It ties those faces to the same $A, B, C, f, g, u, H, K, L, N$ route and prevents a separate quotient or ambient category object from entering the surface.

Combine the two readings with the root downstream unblock package. The package is already the complete downstream surface, so the formal target surface is the same finite row list with no additional Kleisli, host-composition, completion, or generator coordinates. □

Theorem 485.72 (ContinuationMonad root route formal handoff)**.** *Every downstream formal target that reads the* $\mathsf{ContinuationMonad}^{\uparrow}$ *root route receives the rows* $A, B, C, f, g, u, H, K, L, N$ *through the root unit-bind exhaustion, generator-category lock, root downstream unblock package, and bind readback exhaustion. The handoff may expose displayed continuation routes and Category hom readings, but it cannot reinterpret them as host Kleisli composition, quotient morphism equality, ambient category completion, or an unlisted continuation-generator rule.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root route formal handoff`

Proof. First apply root unit-bind exhaustion. It makes the public unit and bind entries read u, f, g, K, L, H, N over the source histories A, B, C rather than a host monad operation.

Next apply the generator-category lock. The same finite route carries generator refusal and $\mathsf{Cat}^{\uparrow}$ hom readings, so a downstream target cannot split the generator ledger from the Category hom row.

Now use bind readback exhaustion. It states that every bind readback is already exhausted by the displayed packet rows and cannot be supplied by a quotient Kleisli arrow, host function, unlisted generator, or external evaluator.

The root downstream unblock package then repacks these rows as the formal handoff. Since all consumer-visible data has already been routed through the finite packet, the refused host and quotient readings have no handoff coordinate. □

Theorem 485.73 (ContinuationMonad root ledger formal boundary)**.** *The ledger boundary for root* $\mathsf{ContinuationMonad}^{\uparrow}$ *formal targets is the finite package* $A, B, C, f, g, u, H, K, L, N$*. It records the source, continuation-tail, unit-tail, transport, route, ledger, provenance, generator-refusal, Category hom, and local* NameCert *rows needed by downstream continuation chapters, and it refuses quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, and continuation generators not listed in the packet.*

Proof. Start with the root downstream unblock package. It identifies the rows available before downstream continuation chapters read any standard monad boundary.

Add root unit-bind exhaustion and bind readback exhaustion. These theorems account for every public unit, bind, and bind-readback request through the same packet rows.

Add the generator-category lock. It places generator refusal and Category hom readings on the same ledger, preventing a separate categorical carrier or unlisted generator list from supplying downstream data.

Thus the ledger boundary is precisely the finite root package. Objects outside $A, B, C, f, g, u, H, K, L, N$ are not recorded by the ledger and cannot be consumed as formal-target input. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root ledger formal boundary`

Theorem 485.74 (ContinuationMonad root formal route saturation)**.** *Every downstream formal target consuming* $\mathsf{ContinuationMonad}^{\uparrow}$ *receives exactly the root rows*

$$A, B, C, f, g, u, H, K, L, N$$

through the existing root formal target surface, root route formal handoff, and downstream route exhaustion. The route exports displayed BHist, Cont, hsame, Pkg, $\mathsf{Cat}^{\uparrow}$ *hom-carrier,* ContinuationMorphism, *and local* NameCert *readings only; it exports no quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluator, or unlisted continuation generator.*

Proof. Start with Definition 485.71. It identifies the formal target surface with the finite root rows $A, B, C, f, g, u, H, K, L, N$ and their displayed BEDC readings.

Next apply Definition 485.72. The handoff keeps downstream formal targets on the same route and refuses host Kleisli composition, quotient morphism equality, ambient category completion, and unlisted continuation-generator rules.

Finally apply Definition 485.50. Since every downstream route is already exhausted by the same root package, the formal route is saturated by exactly those rows and no external evaluator or hidden continuation coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal route saturation`

Theorem 485.75 (ContinuationMonad root ledger saturation boundary)**.** *The root ledger boundary for any downstream formal target consuming* $\mathsf{ContinuationMonad}^{\uparrow}$ *is exactly the finite package*

$$A, B, C, f, g, u, H, K, L, N.$$

The boundary is supplied by the existing root ledger formal boundary together with downstream route exhaustion, and it admits no additional ledger entry for quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, or continuation generators outside the packet.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root ledger saturation boundary`

Proof. First use Definition 485.73. It states that the ledger available to root formal targets is the finite package $A, B, C, f, g, u, H, K, L, N$ with source, tail, unit, transport, route, provenance, generator-refusal, Category hom, and local naming rows.

Then use Definition 485.50. Every downstream unit, bind, generator-boundary, or Category hom read is already routed through that same package.

It remains to exclude extra ledger entries. The root ledger boundary records no coordinate for quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, or unlisted continuation generators, so saturation is exactly the displayed package. □

Theorem 485.76 (ContinuationMonad root formal handoff saturation)**.** *The root formal handoff for* $\mathsf{ContinuationMonad}^{\uparrow}$ *is saturated by the composition of Definition 485.71, Definition 485.72, Definition 485.73, and Definition 485.50. Consequently every downstream formal target receives exactly* $A, B, C, f, g, u, H, K, L, N$ *and receives no quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluator, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal handoff saturation`

Proof. First compose the formal target surface with the route handoff. The target surface lists the rows available to downstream formal targets, while the handoff states that the target may consume them only along the displayed continuation route.

Next compose the ledger boundary with downstream route exhaustion. The ledger boundary records the finite accounting rows, and route exhaustion shows that unit, bind, generator-refusal, and Category hom consumers cannot leave that accounting surface.

Combining the two compositions saturates the handoff. Every formal consumer receives the same $A, B, C, f, g, u, H, K, L, N$ packet, and the refused host, quotient, ambient, evaluator, and unlisted-generator objects have no exported coordinate. □

Theorem 485.77 (ContinuationMonad root formal audit factorization)**.** *Every downstream formal read of* ContinuationMonad$^{\uparrow}$ *factors through the displayed audit rows*

$$A, B, C, f, g, u, H, K, L, N$$

together with BHist*,* Cont*,* hsame*,* Pkg*, the* Cat$^{\uparrow}$ *hom-carrier, and the local* NameCert *surface. The factorization is the same for unit, bind, generator-refusal, Category hom, and naming consumers, and it uses no route outside the displayed root packet.*

Proof. Start with root formal route saturation. It identifies the downstream formal route with $A, B, C, f, g, u, H, K, L, N$ and the displayed BHist, Cont, hsame, Pkg, Cat$^{\uparrow}$ hom-carrier, ContinuationMorphism, and local NameCert readings.

Apply the root ledger saturation boundary. It records the same finite rows as the audit ledger for source, tail, unit, transport, route, provenance, generator-refusal, Category hom, and local naming.

Finally use root formal handoff saturation together with the sibling formal NameCert handoff of Definition 485.81. The handoff keeps downstream formal targets on the same displayed packet, so every formal read factors through the audit rows and their named BEDC carriers. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal audit factorization`

Theorem 485.78 (ContinuationMonad root formal route budget exhaustion)**.** *The root formal route budget exported by* ContinuationMonad$^{\uparrow}$ *is exhausted by* $A, B, C, f, g, u, H, K, L, N$*,* BHist*,* Cont*,* hsame*,* Pkg*,* Cat$^{\uparrow}$*, and the local* NameCert *surface. It exports no quotient Kleisli arrows, no host-function composition, no ambient category completion, no external evaluator, and no unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal route budget exhaustion`

Proof. Use root formal route saturation to fix the exported formal surface. The saturation theorem already lists the accepted rows and the allowed BEDC readings for downstream formal targets.

Use the root ledger saturation boundary to read the budget as finite accounting. Since the ledger has entries only for the displayed root rows, there is no remaining coordinate that could encode quotient Kleisli arrows, host-function composition, ambient completion, external evaluation, or another generator.

The sibling formal NameCert handoff confirms the same refusal at the local naming surface. Root formal handoff saturation then transfers the refusal to downstream formal consumers, exhausting the route budget. □

Theorem 485.79 (ContinuationMonad root formal NameCert consumer lock)**.** *The downstream formal consumers of* ContinuationMonad$^{\uparrow}$ *are locked to the audit factorization of Definition 485.77 and the route-budget exhaustion of Definition 485.78. Together with root formal handoff saturation and the root formal NameCert handoff, this lock permits only reads through* $A, B, C, f, g, u, H, K, L, N$*,* BHist*,* Cont*,* hsame*,* Pkg*,* Cat$^{\uparrow}$*, and local* NameCert*, and refuses all quotient, host-composition, ambient-completion, evaluator, and unlisted-generator reads.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal namecert consumer lock`

Proof. First apply the audit factorization theorem. It routes every downstream formal read through the finite audit rows and their displayed BEDC carriers.

Next apply route-budget exhaustion. It removes the quotient Kleisli, host-composition, ambient-completion, evaluator, and unlisted-generator alternatives from the exported route budget.

Finally combine root formal handoff saturation with Definition 485.81. The handoff and local NameCert route read the same finite packet, so a downstream formal consumer has no accepted route outside the audit factorization and no leftover route budget for the refused objects. □

Theorem 485.80 (ContinuationMonad root formal downstream coverage)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every accepted local consumer of the root formal target is exhausted by the finite downstream coverage surface*

$$A, B, C, f, g, u, H, K, L, N.$$

The coverage surface consists of the displayed BHist *source histories,* Cont *continuation tails, componentwise* hsame *transports,* Pkg *provenance, the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier row, the ContinuationMorphism row, the local* NameCert *row, and the ledger and readback rows named by the root formal route. Unit and bind consumers factor through the unit-bind and bind-readback rows, generator and Category consumers factor through the generator-category rows, and local naming or handoff consumers factor through the root formal handoff rows. Quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, and unlisted continuation-generator rules remain outside the packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal downstream coverage`

Proof. Start with Definition 485.72. It states that every downstream formal target reading the root route receives $A, B, C, f, g, u, H, K, L, N$ through root unit-bind exhaustion, generator-category lock, the root downstream unblock package, and bind readback exhaustion. Thus the accepted local consumers are already sorted into the unit-bind face, the generator-Category face, and the bind-readback or handoff face.

Apply Definition 485.74. The saturation theorem identifies the exported formal route with the same finite rows and lists the only readings exported by that route: BHist, Cont, hsame, Pkg, the $\mathsf{Cat}^{\uparrow}$ hom-carrier, the ContinuationMorphism row, and local NameCert. Therefore any consumer-facing use of the formal target must factor through one of those displayed rows.

It remains to check that no local consumer can enter through an accounting side door. The ledger side is closed by Definition 485.75, which records no extra entry beyond $A, B, C, f, g, u, H, K, L, N$. The coverage side is closed by Definition 485.77 and Definition 485.78, which route audit, naming, Category, unit, bind, and generator-facing reads through the same packet and exhaust the route budget. Finally Definition 485.79 transfers that exhaustion to local formal consumers. Since none of these rows contains a coordinate for quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, or unlisted continuation-generator rules, all excluded alternatives remain outside the BEDC packet. □

Theorem 485.81 (ContinuationMonad root formal NameCert handoff)**.** *Every downstream formal target consuming* $\mathsf{ContinuationMonad}^{\uparrow}$ *receives the local* NameCert *row only through the finite packet*

$$A, B, C, f, g, u, H, K, L, N.$$

The received readings are exactly the displayed BHist *source histories,* Cont *unit and bind tails,* hsame *transports,* Pkg *provenance,* $\mathsf{Cat}^{\uparrow}$ *hom-carrier row,* ContinuationMorphism *row, and local*

NameCert *row. No quotient Kleisli arrow, host-function composition, ambient category completion, external monad evaluator, or unlisted continuation generator is exported.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal namecert handoff`

Proof. First apply Definition 485.87. It forces root consumers to read the source histories A, B, C and continuation tails f, g, u through the displayed BHist, hsame, Cont, Pkg, and NameCert surface of the packet.

Next use Definition 485.74. Downstream formal targets receive exactly $A, B, C, f, g, u, H, K, L, N$ through the formal route, together with the displayed Category hom-carrier and ContinuationMorphism readings.

Finally compose with Definition 485.75. The ledger boundary is the same finite packet and admits no extra entry for quotient Kleisli arrows, host-function composition, ambient category completion, external evaluators, or unlisted continuation generators. Hence the local NameCert row reaches downstream formal targets only through the named packet and not through an external monad schema. □

Theorem 485.82 (Category ContinuationMonad hom-carrier route)**.** *Every downstream formal target reading a* ContinuationMonad$^{\uparrow}$ *packet through* $A, B, C, f, g, u, H, K, L, N$ *receives its* Cat$^{\uparrow}$ *hom-carrier exposure exactly through the displayed* CategoryHomCarrier *and* ContinuationMorphism *rows. The route is the finite continuation packet route; it is not host function composition, a quotient Kleisli arrow, an ambient category completion, or an external monad evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category hom carrier route`

Proof. Start with Definition 485.81. It fixes the downstream formal handoff to the finite packet $A, B, C, f, g, u, H, K, L, N$ and already names the Category hom-carrier and ContinuationMorphism readings as packet rows.

Next apply Definition 61.317. The Category side reads a hom carrier exactly as unary endpoint rows together with a continuation morphism whose displayed tail is the morphism history. Thus the CategoryHomCarrier row consumed downstream is the continuation-tail row of the same packet.

Finally use Definition 485.87, Definition 485.74, and Definition 485.75. These root source and ledger saturation results leave no additional coordinate for host composition, quotient Kleisli arrows, ambient category completion, external evaluators, or unlisted continuation generators, so the Category hom-carrier route is exactly the displayed continuation packet route. □

Theorem 485.83 (ContinuationMonad Category hom readback correspondence)**.** *Any* Cat$^{\uparrow}$ *hom-carrier read consumed by the* ContinuationMonad$^{\uparrow}$ *formal target is exactly the readback of the finite packet rows*

$$A, B, C, f, g, u, H, K, L, N.$$

The correspondence is carried by the displayed CategoryHomCarrier *and* ContinuationMorphism *rows of the packet. It cannot be replaced by quotient Kleisli arrows, host-function composition, ambient category completion, an external monad evaluator, or an unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier category hom readback correspondence`

Proof. First apply Definition 485.81. It fixes the formal target handoff to the finite packet $A, B, C, f, g, u, H, K, L, N$ and names the Category hom-carrier and ContinuationMorphism rows as part of that packet.

Next use Definition 485.82. The Category hom-carrier exposure reaches the formal target only through the displayed CategoryHomCarrier and ContinuationMorphism readings, so the Category-side readback is already the continuation packet readback.

Finally compose with Definition 485.84. The formal target has no remaining coordinate for quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, or unlisted generators. Hence every consumed $\mathsf{Cat}^{\uparrow}$ hom-carrier read corresponds exactly to the finite continuation packet rows. □

Theorem 485.84 (ContinuationMonad formal target readback exhaustion)**.** *Every downstream formal target consuming the* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet through*

$$A, B, C, f, g, u, H, K, L, N$$

must first read the root formal NameCert *handoff and the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier route. Therefore the target receives only the displayed* BHist *source histories,* Cont *continuation tails,* hsame *transports,* Pkg *provenance, Category hom-carrier row,* ContinuationMorphism *row, and local* NameCert *row; it receives no quotient Kleisli arrow, host-function composition, ambient category completion, external monad evaluator, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier formal target readback exhaustion`

Proof. Start with Definition 485.81. It fixes the formal handoff to $A, B, C, f, g, u, H, K, L, N$ and excludes quotient Kleisli arrows, host composition, ambient category completion, external evaluators, and unlisted generators from the local NameCert route.

Next apply Definition 485.82. The Category hom-carrier exposure is not an ambient category completion; it is the displayed CategoryHomCarrier and ContinuationMorphism reading of the same finite continuation packet.

Finally compose with Definition 485.87, Definition 485.74, and Definition 485.75. These root source, route, and ledger boundaries exhaust the readback surface, so every formal target consumes the finite packet and no external monad semantics. □

Theorem 485.85 (ContinuationMonad formal route target exhaustion)**.** *Any formal target that reads the displayed continuation packet*

$$A, B, C, f, g, u, H, K, L, N$$

must factor through the root formal NameCert *handoff, the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier route, and the formal-target readback exhaustion. Therefore the target receives only the displayed* BHist *source histories,* Cont *continuation tails,* hsame *transports,* Pkg *provenance, Category hom-carrier row,* ContinuationMorphism *row, and local* NameCert *row. It cannot obtain quotient Kleisli arrows, host-function composition, ambient category completion, external monad evaluators, or unlisted generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier formal route target exhaustion`

Proof. First apply Definition 485.81. It fixes the formal route to the finite packet $A, B, C, f, g, u, H, K, L, N$ and excludes quotient Kleisli arrows, host composition, ambient category completion, external evaluators, and unlisted generators from the local NameCert handoff.

Next use Definition 485.82. The Category hom-carrier exposure is the displayed CategoryHomCarrier and ContinuationMorphism reading of the same continuation packet, not an external category-completion route.

Finally compose with Definition 485.84. The formal target has already read the root handoff and the Category route, and the readback theorem exhausts the remaining consumer surface. Hence every formal consumer factors through those three routes and receives no unlisted continuation or monad data. □

Theorem 485.86 (ContinuationMonad root formal handoff downstream coverage)**.** *Every accepted local consumer of the* $\mathsf{ContinuationMonad}^{\uparrow}$ *root formal handoff is exhausted by the displayed finite route*

$$A, B, C, f, g, u, H, K, L, N.$$

Its consumer-facing use factors through one of the already enumerated rows: the BHist *source histories* A, B, C*, the* Cont *unit and bind tails* f, g, u*, the componentwise* hsame *transport row* H*, the formal route and ledger rows* K, L*, the* Pkg *provenance row, the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier row, the* ContinuationMorphism *row, or the local* NameCert *row* N*. No quotient Kleisli arrow, host-function composition, ambient category completion, external monad evaluator, or unlisted continuation generator becomes an accepted alternative of the BEDC packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal handoff downstream coverage`

Proof. Start with Definition 485.81. It fixes the local handoff to the finite packet $A, B, C, f, g, u, H, K, L, N$ and states that downstream formal targets receive the local NameCert row only through the displayed source-history, continuation-tail, transport, provenance, Category hom-carrier, ContinuationMorphism, and naming rows. Thus an accepted local consumer has no entry point before the finite handoff packet is read.

Apply Definition 485.84. The readback theorem forces any downstream formal target consuming the packet to read the root formal NameCert handoff together with the $\mathsf{Cat}^{\uparrow}$ hom-carrier route. It also identifies the remaining readback surface with exactly the displayed BHist, Cont, hsame, Pkg, Category hom-carrier, ContinuationMorphism, and local NameCert rows.

The ledger and coverage coordinates are therefore the same finite list. The route rows K, L account for the formal route and ledger boundary, while H transports only the displayed components and N names only that packet. Since the handoff theorem excludes the quotient, host-composition, ambient-completion, external-evaluator, and unlisted-generator alternatives from the local export, and the readback-exhaustion theorem leaves no further downstream coordinate after the handoff and Category route are read, every accepted local consumer is exhausted by the enumerated rows and every excluded alternative remains outside the BEDC packet. □

Theorem 485.87 (ContinuationMonad root source coverage)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every root consumer must read the displayed source histories* A, B, C *and the displayed continuation tails* f, g, u *before any unit or bind law can be consumed. The source coverage is exactly the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface of* Z*; it does not pass through host functions, quotient Kleisli arrows, ambient hom sets, or unlisted generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root source coverage`

Proof. Start with Definition 485.67. It identifies the source of a root read with $A, B, C, f, g, u, H, K, L, N$ and leaves no coordinate for an external source object.

Next use Definition 485.68. Unit and bind reads are covered only after the displayed histories, tails, route row, transport row, ledger row, and local name have been read.

Finally apply Definition 485.69. This excludes host functions, quotient Kleisli arrows, ambient hom sets, independent category carriers, and unlisted continuation generators from the root source route, so coverage is exactly the displayed packet. □

Theorem 485.88 (ContinuationMonad root source triad exactness)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every root consumer route is exactly the joint surface obtained from source exposure, continuation-rule coverage, and downstream consumer separation. The route consists of the displayed source histories* A, B, C*, continuation tails* f, g, u*, route row, transport row, ledger row, package provenance, and local naming row of* Z*; it contains no host function, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root source triad exactness`

Proof. Apply Definition 485.67. It identifies the root route source with the displayed packet $Z = (A, B, C, f, g, u, H, K, L, N)$ and supplies no external source coordinate.

Next use Definition 485.68. The unit and bind laws are readable only through the displayed histories, tails, route row, transport row, ledger row, and local name.

Finally apply Definition 485.69. Downstream consumers cannot receive host functions, quotient Kleisli arrows, ambient hom sets, or unlisted continuation generators, so the three component surfaces exhaust the root consumer route exactly. □

Theorem 485.89 (ContinuationMonad root generator consumer lock)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every generator-facing root consumer is locked to the generator closure surface of chapter 486 and to the existing generator-boundary refusal of* Z*. A consumer may read only the displayed source histories, continuation tails, transport rows, route rows, ledger rows, package rows, and local name rows; it may not introduce host functions, quotient Kleisli arrows, ambient hom sets, or continuation generators not listed in the packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator consumer lock`

Proof. Read the generator side through Definition 485.56. It routes the generator face of Z through chapter 486 and the same $A, B, C, f, g, u, H, K, L, N$ packet.

Then apply Definition 485.17. The refusal row blocks every unit, bind, continuation rule, or route generator that is not already displayed in the accepted packet.

Combine this with Definition 485.58. Downstream consumers receive the generator-boundary refusal as part of the finite route package, so no host function, quotient Kleisli arrow, ambient hom set, or unlisted generator can be added at the consumer boundary. □

Theorem 485.90 (ContinuationMonad root category generator obligation surface)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the root category-generator obligation surface is the shared finite package read through chapter 61, Definition 485.56, Definition 485.57, and Definition 485.58. The surface consists only of the displayed* BHist *source rows,* Cont *unit and bind tails, componentwise* hsame *transports,* Pkg *provenance,* NameCert *rows, generator-boundary refusal, and* $\mathsf{Cat}^{\uparrow}$ *hom-carrier readings.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root category generator obligation surface`

↪ variant `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli associativity surface`

Proof. First apply Definition 485.56. It binds the $\mathsf{Cat}^{\uparrow}$ hom-carrier surface of chapter 61 and the generator-boundary surface to the same accepted packet Z.

Next use Definition 485.57. The root unit and bind schedules are exhausted by u, f, g, H, K, L, N, so category and generator consumers cannot replace the schedule by host composition or quotient Kleisli arrows.

Finally apply Definition 485.58. It packages the triad and schedule exhaustion for downstream use, with no coordinate for ambient hom sets, external category completion, host functions, or unlisted generators. □

Theorem 485.91 (ContinuationMonad root NameCert ledger exactness)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every root consumer receives exactly the source histories* A, B, C*, continuation tails* f, g, u*, unit and bind rows, generator-boundary refusal,* $\mathsf{Cat}^{\uparrow}$ *hom-carrier readings, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows assembled by Definition 485.87, Definition 485.89, and Definition 485.90. The same root consumer receives no host function, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root namecert ledger exactness`

Proof. First apply Definition 485.87. It fixes the root source route to the displayed histories A, B, C, tails f, g, u, and the finite BHist, hsame, Cont, Pkg, and NameCert surface of Z.

Next apply Definition 485.89. It adds the generator-boundary refusal to the same accepted packet and blocks host functions, quotient Kleisli arrows, ambient hom sets, and unlisted continuation generators at every generator-facing root consumer.

Then apply Definition 485.90. It contributes the $\mathsf{Cat}^{\uparrow}$ hom-carrier readings and confirms that the unit and bind schedules are read through the displayed continuation rows rather than through an external category or quotient Kleisli object.

The carrier coordinates are exhausted by $A, B, C, f, g, u, H, K, L, N$ together with the named source, continuation, generator, category, transport, route, provenance, and local naming readings supplied by these three theorems. No remaining coordinate can export a host function, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator, so the root NameCert ledger is exact. □

Theorem 485.92 (ContinuationMonad root source public contract)**.** *For every accepted root-source public consumer of* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the public read factors through Definition 485.87, Definition 485.89, Definition 485.90, and Definition 485.91 before any unit, bind, generator, or category surface is consumed. The resulting contract exposes only the displayed source histories, continuation tails, unit and bind rows, generator-boundary refusal, category-generator obligation surface, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows of* Z*; it refuses host functions, quotient Kleisli arrows, ambient hom sets, and unlisted continuation generators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root source public contract`

Proof. First apply root source coverage. The public consumer must enter through the displayed histories A, B, C, continuation tails f, g, u, and finite BHist, hsame, Cont, Pkg, and NameCert rows of Z.

Next apply the generator consumer lock. This makes the generator-facing route pass through the displayed generator-boundary refusal before any generator surface can be consumed.

Then apply the category-generator obligation surface. Category readings are available only through the named finite package and its $\mathsf{Cat}^{\uparrow}$ hom-carrier readings, not through ambient hom sets or quotient Kleisli arrows.

Finally apply root NameCert ledger exactness. It assembles the four public readings into one ledgered route and leaves no coordinate for host functions, quotient Kleisli arrows, ambient hom sets, or unlisted continuation generators. □

Theorem 485.93 (ContinuationMonad root public contract factorization)**.** *For every accepted root-source public consumer of* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the public contract factors through exactly the ordered chain of root source coverage, generator consumer lock, category-generator obligation surface, and root* NameCert *ledger exactness before any unit, bind, generator, or category reading is exposed. The factorization refuses host functions, quotient Kleisli arrows, ambient hom sets, and unlisted continuation generators at each public read boundary.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public contract factorization`

Proof. First apply Definition 485.87. It fixes the source part of the public route to the displayed histories A, B, C, continuation tails f, g, u, and the finite BHist, hsame, Cont, Pkg, and NameCert surface of Z.

Next apply Definition 485.89. The generator-facing read is then locked to the displayed generator-boundary refusal, so no host function, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator can be inserted before generator exposure.

Then apply Definition 485.90. This makes the category-facing part a read of the displayed $\mathsf{Cat}^{\uparrow}$ hom-carrier and continuation rows rather than a read of an ambient category or quotient Kleisli object.

Finally apply Definition 485.91 and then Definition 485.92. The ledger assembles the three prior surfaces in the stated order, and the public-contract endpoint exposes only the resulting unit, bind, generator, and category readings. Since every excluded coordinate has already been refused along the chain, the public contract has exactly this factorization. □

Theorem 485.94 (ContinuationMonad root formal target exactness)**.** *For every root public and formal consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the accepted read factors through the source histories* A, B, C*, continuation tails* f, g, u*, unit and bind rows, category row, generator row, public read, formal read,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support displayed by* Z*. It does not factor through host Kleisli composition, quotient morphism equality, ambient category completion, or any generator absent from the packet.*

↪variant `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli formal target package`

Proof. Start with root source coverage, Definition 485.87. It fixes the consumer source to the displayed histories, continuation tails, and finite BHist, hsame, Cont, Pkg, and NameCert surface.

Next apply the generator consumer lock and the category-generator obligation surface, Definition 485.89 and Definition 485.90. These expose the generator and category rows without admitting host Kleisli composition, quotient morphism equality, ambient category completion, or an unlisted generator.

Finally use root NameCert ledger exactness and root public contract factorization, Definition 485.91 and Definition 485.93. The first assembles the formal ledger support, and the second makes the public read pass through the same ordered finite route. Thus the public/formal consumer has exactly the displayed target surface. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal target exactness`

Theorem 485.95 (ContinuationMonad root formal-public readback terminality)**.** *Every root public/formal consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *factors through the existing root source coverage, generator consumer lock, category-generator obligation surface, root* NameCert *ledger exactness, public contract factorization, and formal target exactness before it can read unit, bind, generator, category, or formal rows. After those factorizations, no host Kleisli composition, quotient morphism equality, ambient category completion, external monad evaluator, or unlisted continuation generator remains available.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root formal public readback terminality`

Proof. First apply Definition 485.87. It fixes the root public/formal source to the displayed histories A, B, C, continuation tails f, g, u, and the finite BHist, hsame, Cont, Pkg, and NameCert support of Z.

Next apply Definition 485.89 and Definition 485.90. These force generator and category readings to pass through the displayed generator-boundary refusal and $\mathsf{Cat}^{\uparrow}$ hom-carrier rows, so the route has no host Kleisli composition, quotient morphism equality, ambient category completion, external evaluator, or unlisted generator at those boundaries.

Then use Definition 485.91 and Definition 485.93. The ledger assembles the root rows into one ordered public route, and the public contract exposes unit, bind, generator, and category readings only through that route.

Finally apply Definition 485.94. The formal target is already the same finite public/formal surface, so the consumer is terminal at the displayed readback boundary and no unlisted continuation or external monad semantics remains to be read. □

Theorem 485.96 (ContinuationMonad root terminal readback coverage)**.** *Every terminal root readback of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *is covered exactly by the ordered root surface consisting of root source coverage, generator consumer lock, category-generator obligation surface, root* NameCert *ledger exactness, public contract factorization, formal target exactness, and formal-public readback terminality. The terminal readback exposes only the displayed source histories, continuation tails, unit and bind rows, generator-boundary refusal,* $\mathsf{Cat}^{\uparrow}$ *hom-carrier readings, formal/public rows,* BHist, hsame, Cont, Pkg, *and local* NameCert *support of* Z.

It excludes host Kleisli composition, quotient morphism equality, ambient category completion, external monad evaluator, and every continuation generator not displayed in the packet.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root terminal readback coverage`

Proof. First apply Definition 485.87. It fixes the terminal readback source to the displayed histories A, B, C, continuation tails f, g, u, and finite BHist, hsame, Cont, Pkg, and NameCert support.

Next apply Definition 485.89 and Definition 485.90. These identify the generator-boundary refusal and the category-generator surface without allowing host Kleisli composition, quotient morphism equality, ambient category completion, or unlisted generators.

Then use Definition 485.91 and Definition 485.93. The ledger collects the root rows into the local NameCert surface, and the public contract forces the terminal readback to pass through that ordered public route before any public unit, bind, generator, or category row is exposed.

Finally apply Definition 485.94 and Definition 485.95. The formal target and formal-public terminality make the same finite route the terminal readback boundary. Hence every covered row

is one of the displayed rows named in the statement, and every host, quotient, ambient-completion, external-evaluator, or unlisted-generator coordinate is outside the terminal coverage surface. □

Theorem 485.97 (ContinuationMonad generator category route totality)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the public generator-category route is total exactly on the displayed carrier rows* $A, B, C, f, g, u, H, K, L, N$*. The* $\mathsf{Cat}^{\uparrow}$ *hom-carrier surface reads the continuation tails as* CategoryHomCarrier *and* ContinuationMorphism *rows, the* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal surface blocks every unlisted continuation generator, and the* Cont*,* hsame*,* Pkg*, and* NameCert *rows supply the only route, transport, provenance, and naming data available to downstream consumers.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator category route totality`

Proof. Begin with the root category-generator factorization. By Definition 485.51, the category face and generator face already share the finite packet $A, B, C, f, g, u, H, K, L, N$.

Use the root CategoryHomCarrier consumer totality theorem next. It places every hom-carrier read on the displayed continuation tails and the $\mathsf{Cat}^{\uparrow}$ surface, with no ambient hom set or external category completion.

Finally apply the root generator boundary readback theorem and the generator-boundary refusal surface. They block unlisted unit, bind, continuation, or route generators. Since the carrier rows, hsame transports, Cont routes, Pkg provenance, and local NameCert row have already been listed, the route is total on the displayed packet and nowhere else. □

Theorem 485.98 (ContinuationMonad generator-category lockdown)**.** *For an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the generator boundary and the* $\mathsf{Cat}^{\uparrow}$ *hom route form one locked root obligation surface over* BHist*,* Cont*, componentwise* hsame*,* Pkg*, and the local* NameCert *row. Every downstream read of the authorized generator row, the* CategoryHomCarrier *row, or the* ContinuationMorphism *row factors through the same displayed rows* $A, B, C, f, g, u, H, K, L, N$*, and no host function space, quotient Kleisli category, ambient category carrier, or independent generator list is exported.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier generator category lockdown`

Proof. Start with the carrier and root generator obligation. Carrier admission fixes the readable rows $A, B, C, f, g, u, H, K, L, N$, while the root generator obligation places every authorized continuation generator inside the same finite packet before unit or bind data is consumed.

Use the bind-route totality and Category hom route totality theorems. They route the Cont reads and the $\mathsf{Cat}^{\uparrow}$ hom-carrier readings through K, L and transport them only by the componentwise hsame row H.

Finally apply the root downstream package. Its Pkg provenance and local NameCert row repack the generator and Category readings as one obligation surface, so a downstream continuation consumer has no coordinate for a separate host, quotient, ambient category, or generator object. □

Theorem 485.99 (ContinuationMonad displayed bind readback exhaustion)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, each displayed bind readback is exhausted by* $A, B, C, f, g, u, H, K, L, N$ *together with the* $\mathsf{Cat}^{\uparrow}$ *hom-carrier surface, the* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal surface, and the* Cont*,* hsame*,* Pkg*, and* NameCert *rows. The readback does not export host function composition, host monad equality, quotient Kleisli arrows, ambient hom sets, or external evaluator rows.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier displayed bind readback exhaustion`

Proof. Start from the root bind-unit boundary of Definition 485.52. It identifies public bind and unit reads with u, f, g, H, K, L, N over the source histories A, B, C.

Next attach the finite bind readback package and root downstream surface exactness. These package the bind read as BEDC data and keep the category and generator faces on the same finite source.

The exclusion clauses come from the Kleisli composition boundary and the root ledger saturation boundary. They admit no coordinate for host function composition, quotient Kleisli arrows, ambient hom sets, or external evaluator rows, so the displayed bind readback is exhausted by the accepted packet and its named consumer surfaces. □

Theorem 485.100 (ContinuationMonad root consumer non-escape)**.** *Every root consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *reads through the same finite generator-category route, the same displayed bind readback, and the same* Cont, hsame, Pkg, *and* NameCert *support rows. No root consumer can escape to host function composition, quotient Kleisli arrows, host monad equality, ambient category completion, an independent* $\mathsf{Cat}^{\uparrow}$ *hom carrier, or a generator outside the* $\mathsf{GeneratorClosure}^{\uparrow}$ *refusal surface.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root consumer nonescape`

Proof. First combine Definition 485.97 with Definition 485.99. The first theorem fixes the generator-category route, and the second fixes the bind readback.

Next use the root downstream unblock package and root Kleisli source totality. They identify downstream and Kleisli-facing reads with the carrier rows $A, B, C, f, g, u, H, K, L, N$ and the structural rows already displayed by the packet.

Finally apply the generator boundary readback and CategoryHomCarrier consumer totality surfaces. Since both root consumer faces are already routed through the accepted packet, there is no remaining route to host composition, quotient Kleisli arrows, host equality, ambient category completion, independent hom carriers, or unlisted generators. □

Theorem 485.101 (ContinuationMonad root Kleisli source totality)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the Kleisli source exposed to any root consumer is exhausted by the displayed* BHist *source histories* A, B, C*, the continuation tails* f, g*, the unit tail* u*, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* K*, the ledger row* L*, and the local* NameCert *row* N*.*

No host function, quotient Kleisli arrow, ambient hom set, external monad evaluator, or continuation rule outside Z *is part of that source.*

Proof. Start with carrier admission from Definition 485.6. It displays exactly $A, B, C, f, g, u, H, K, L, N$ as the accepted source packet.

Next use Definition 485.87. Every root read must pass through the displayed histories and continuation tails before unit or bind data can be consumed.

Finally apply ledger exactness from Definition 485.12. The consumer-visible Kleisli read is exhausted by the carrier, classifier, unit, bind, associativity, Cont, Pkg, and NameCert rows of the same finite packet, so no external source coordinate remains. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli source totality`

Theorem 485.102 (ContinuationMonad root generator boundary readback)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every generator-facing readback of the root Kleisli source factors through the authorized generator boundary named by chapter 486 and the displayed generator-boundary refusal for* Z*.*

Such a readback may replay only the listed histories, continuation tails, unit tail, transport rows, route rows, ledger rows, package rows, and local name rows. It cannot introduce an unlisted continuation rule or a second generator route.

Proof. Read the generator face through Definition 485.56. It attaches the generator boundary to the same packet $A, B, C, f, g, u, H, K, L, N$.

Apply Definition 485.17. The refusal row blocks every unit row, bind row, continuation rule, or route generator that is not already displayed in the accepted packet.

Close the readback with Definition 485.89. Generator-facing consumers inherit the same refusal and the same finite source surface, so the readback has no coordinate for an unlisted continuation rule. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator boundary readback`

Theorem 485.103 (ContinuationMonad root CategoryHomCarrier consumer totality)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every* CategoryHomCarrier*-facing consumer of the root Kleisli source is exhausted by the displayed continuation tails* f, g, u *and the* $\mathsf{Cat}^{\uparrow}$ *hom-row reading over* A, B, C, H, K, L, N*.*

The consumer may use the componentwise hsame *transports, displayed* Cont *routes, ledger, package provenance, and local* NameCert *row of* Z*, but it receives no ambient hom set, host composition operation, quotient Kleisli arrow, or external category completion.*

Proof. Begin with Definition 485.16. The CategoryHomCarrier and ContinuationMorphism entries are exposed only as reads of the displayed continuation tails over the same histories, transports, routes, ledger, and local name.

Next use Definition 485.90. It places the $\mathsf{Cat}^{\uparrow}$ hom-carrier reading on the shared finite root package and excludes independent category data.

Finally apply Definition 485.57 and Definition 485.75. Unit and bind schedules are exhausted by u, f, g, H, K, L, N, while the ledger boundary admits no quotient Kleisli arrow, host-function composition, ambient category completion, external evaluator, or unlisted generator. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root category hom consumer totality`

Theorem 485.104 (ContinuationMonad root Kleisli source route exhaustion)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every root Kleisli consumer factors through the displayed source histories* A, B, C*, continuation tails* f, g*, unit row* u*, generator boundary,* CategoryHomCarrier *row, componentwise* hsame *transport* H*,* Cont *replay row* K*,* Pkg *provenance row* L*, and local* NameCert *row* N*.*

The resulting source route is exhausted by the finite packet Z *and by the downstream readback surface of Definition 485.66. It contains no host evaluator, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator.*

Proof. Start with Definition 485.101. It displays the source histories, continuation tails, unit row, componentwise hsame transport, Cont route, ledger row, and local NameCert row as the complete root Kleisli source for Z.

Apply Definition 485.102. The generator-facing part of the same source route is forced through the authorized boundary and cannot introduce a second continuation rule or generator route.

Next use Definition 485.103. CategoryHom-facing consumers read only the displayed continuation tails and the $\mathsf{Cat}^{\uparrow}$ hom row over the same histories, transports, routes, ledger, package, and local name.

Finally cite Definition 485.66. Its formal target surface packages the downstream readback over the same finite carrier, so composing the three local theorems leaves no coordinate for a host evaluator, quotient Kleisli arrow, ambient hom set, or unlisted continuation generator. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli source route exhaustion`

Theorem 485.105 (ContinuationMonad root unit-bind coverage)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N),$$

every root consumer reads the unit row and bind rows only through the displayed histories A, B, C*, continuation tails* f, g*, unit tail* u*, componentwise* hsame *transport row* H*, displayed* Cont *route row* K*, ledger row* L*, and local* NameCert *row* N*. No unit or bind reading is available as a host function, quotient Kleisli arrow, ambient monad evaluator, or continuation rule outside this finite packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root unit bind coverage`

Proof. Start with carrier admission from Definition 485.6. It identifies the accepted packet with exactly the displayed rows $A, B, C, f, g, u, H, K, L, N$.

Use the root Kleisli source totality of Definition 485.101. That source exposes the histories and continuation tails before any root consumer can read unit or bind data.

Finally apply the root bind-unit boundary of Definition 485.52. It forces public unit and bind reads through u, f, g, H, K, L, N over the same histories A, B, C, leaving no coordinate for a host or quotient-level operation. □

Theorem 485.106 (ContinuationMonad root Kleisli associativity scope)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, Kleisli associativity at the root is scoped exactly by the displayed* Cont *rows in* K*, the componentwise* hsame *transports in* H*, the ledger row* L*, and the* CategoryHomCarrier *readings attached to* f, g, u *over* A, B, C*. The associativity surface is therefore a finite BEDC route reading of* Z*, not a host composition law or quotient Kleisli equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli associativity scope`

Proof. Begin with Definition 485.4. The associativity row is recorded in L, transported by H, and read through the displayed Cont row K.

Attach the category face through Definition 485.16. The CategoryHomCarrier and ContinuationMorphism readings are only readings of the displayed tails f, g, u over the same histories and structural rows.

Now combine Definition 485.113 with the unit-bind coverage theorem above. The root associativity surface has exactly the displayed scope and cannot be enlarged into host function composition or quotient Kleisli equality. □

Theorem 485.107 (ContinuationMonad root downstream generator package)**.** *The downstream* $\mathsf{Cat}^{\uparrow}$ *and generator-boundary consumers of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet receive exactly the finite package* $A, B, C, f, g, u, H, K, L, N$ *together with the displayed* CategoryHomCarrier *readings, generator-boundary refusal,* Cont *routes,* hsame *transports, ledger, provenance, and local naming rows. They receive no host function composition, quotient Kleisli arrow, ambient category completion, external evaluator, or unlisted continuation rule.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream generator package`

Proof. First use Definition 485.53. It identifies the downstream root surface with the category row, generator refusal row, carrier rows, classifier stability, ledger exactness, package provenance, and local NameCert row over $A, B, C, f, g, u, H, K, L, N$.

Next use Definition 485.115. The generator-facing consumer must pass through the same source, bind schedule, unit row, transport row, route row, provenance, and local name before any category or Kleisli surface is exported.

Finally apply Definition 485.100. Since every root consumer is routed through the finite generator-category route and displayed bind readback, the downstream package exports none of the listed host, quotient, completion, evaluator, or unlisted-rule objects. □

Theorem 485.108 (ContinuationMonad unit-bind terminal route)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N),$$

every terminal unit-bind read routes through the histories A, B, C*, continuation tails* f, g*, unit tail* u*, componentwise* hsame *transport row* H*, displayed* Cont *route row* K*, ledger row* L*, local* NameCert *row* N*, and the attached* CategoryHomCarrier *and Kleisli readings. The terminal route is a finite BEDC packet route, not host function composition or quotient Kleisli equality.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit bind terminal route`

Proof. Start with the root unit-bind coverage of Definition 485.105. It identifies every root unit or bind read with the displayed rows $A, B, C, f, g, u, H, K, L, N$.

Next apply the root Kleisli associativity scope of Definition 485.106. That theorem attaches the CategoryHomCarrier and Kleisli readings to the same displayed continuation rows rather than to host composition.

Finally use the downstream generator package of Definition 485.107. The terminal consumer receives exactly that finite package, so the unit-bind route remains the displayed packet route. □

Theorem 485.109 (ContinuationMonad unit-bind completion nonescape)**.** *No accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumer can replace the unit-bind terminal route of Definition 485.108 by host function composition, quotient Kleisli equality, ambient monad completion, or an evaluator outside the packet. Any such replacement would require a coordinate not present among* $A, B, C, f, g, u, H, K, L, N$ *or among the displayed* CategoryHomCarrier *and Kleisli rows.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit bind completion nonescape`

Proof. Use Definition 485.100. It excludes host function composition, quotient Kleisli arrows, ambient category completion, external evaluators, and unlisted continuation rules from every root consumer route.

Then apply Definition 485.108. The terminal unit-bind read has already been identified with the displayed histories, continuation tails, unit row, transport row, route row, ledger, local name, and category/Kleisli readings.

Since none of the excluded objects appears in those rows, no accepted consumer can replace the terminal route by an external completion or evaluator. □

Theorem 485.110 (ContinuationMonad unit-bind Kleisli consumer handoff)**.** *Every* $\mathsf{Cat}^{\uparrow}$ *or generator-closure consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet receives only the terminal unit-bind route displayed in Definition 485.108 and the local* NameCert *row. The consumer may replay the* CategoryHomCarrier *and Kleisli readings attached to* $A, B, C, f, g, u, H, K, L, N$*, but it receives no host composition law, quotient Kleisli equality, ambient monad completion, or external evaluator.*

Proof. Begin with Definition 485.107. It identifies the downstream $\mathsf{Cat}^{\uparrow}$ and generator-boundary package with the finite route $A, B, C, f, g, u, H, K, L, N$ plus the displayed category, generator, transport, ledger, provenance, and local naming rows.

Use Definition 485.108 to specialize that package to terminal unit-bind reads. The same rows supply the unit tail, bind tails, CategoryHomCarrier readings, and Kleisli route.

Finally apply Definition 485.109. The handoff therefore exports only the finite packet route and refuses host composition, quotient Kleisli equality, ambient completion, and external evaluators. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit bind kleisli consumer handoff`

Theorem 485.111 (ContinuationMonad unit-bind KernelMorphism compiler coherence)**.** *Every* KernelMorphismUp *or* $\mathsf{CertificateCompiler}^{\uparrow}$ *consumer of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *unit-bind terminal route reads the same finite packet*

$$A, B, C, f, g, u, H, K, L, N$$

and the attached CategoryHomCarrier *and Kleisli rows. Such a consumer cannot replace the route by host function composition, quotient Kleisli equality, ambient monad completion, or an external evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier unit bind kernelmorphism compiler coherence`

Proof. First apply root unit-bind coverage, Definition 485.105. It identifies every accepted unit or bind read with the displayed histories A, B, C, continuation tails f, g, unit tail u, transport row H, route row K, ledger L, and local naming row N.

Next use the root downstream generator package of Definition 485.107. This is the package available to downstream route consumers, so a kernel-morphism or compiler consumer receives the same displayed packet together with the category, generator, transport, ledger, provenance, and local naming rows.

Apply the terminal route theorem Definition 485.108. The CategoryHomCarrier and Kleisli rows are attached to the same packet rather than to a host operation.

Finally invoke completion nonescape, Definition 485.109. Host function composition, quotient Kleisli equality, ambient monad completion, and external evaluators are not coordinates of $A, B, C, f, g, u, H, K, L, N$, so the KernelMorphism and CertificateCompiler consumers preserve the displayed terminal route. □

Theorem 485.112 (ContinuationMonad root bind completion classifier)**.** *For every completion-facing root read of an accepted* ContinuationMonad$^{\uparrow}$ *packet*

$$Z = (A, B, C, f, g, u, H, K, L, N),$$

the classifier surface is the same as the displayed unit-bind terminal route: histories A, B, C, continuation tails f, g, unit tail u, componentwise hsame *row H, displayed* Cont *row K, ledger L, local* NameCert *row N, and the attached* CategoryHomCarrier *and Kleisli readings. No completion-facing root consumer can reread the packet as host function composition, quotient Kleisli equality, ambient monad completion, or an external evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root bind completion classifier`

Proof. First use Definition 485.108. It fixes the unit-bind terminal route as the rows $A, B, C, f, g, u, H, K, L, N$ together with the attached CategoryHomCarrier and Kleisli readings.

Next apply Definition 485.109. Any completion-facing root read that tried to use host composition, quotient Kleisli equality, ambient monad completion, or an external evaluator would require a coordinate outside the displayed packet.

Finally use Definition 485.111. KernelMorphism and CertificateCompiler consumers preserve the same finite terminal route, so the completion-facing classifier surface is exactly the displayed unit-bind packet and the attached category and Kleisli readings. □

Theorem 485.113 (ContinuationMonad root Kleisli associativity surface)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet $Z = (A, B, C, f, g, u, H, K, L, N)$, the displayed unit rows and bind rows replay through one Kleisli associativity surface over* BHist, hsame, Cont, Pkg, *and* NameCert. *The surface is the route carried by $A, B, C, f, g, u, H, K, L, N$, with the* Cat$^{\uparrow}$ *hom-carrier and* ContinuationMorphism *readings attached to those rows, and it exports no host function composition, quotient Kleisli arrow, ambient category completion, external evaluator, or unlisted continuation generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier kleisli associativity surface`

Proof. Begin with Definition 485.3 and Definition 485.4. The unit rows and nested bind rows are read as finite ledger rows over u, f, g, K, L rather than as host monad equations.

Use Definition 485.101 to expose the source histories and continuation tails before any Kleisli-facing consumer reads them. The source remains the displayed BHist, hsame, Cont, Pkg, and local NameCert surface of Z.

Finally combine Definition 485.67, Definition 485.97, and Definition 485.99. These theorems attach the Category hom and generator-boundary faces to the same route, so associativity is visible as one replay surface and not as a quotient or host-composition object. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli associativity surface`

Theorem 485.114 (ContinuationMonad BHist bind associativity induction)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet $Z = (A, B, C, f, g, u, H, K, L, N)$, nested bind replay over the displayed finite source rows is associative by induction on the finite continuation schedule. The induction uses only the* BHist *source rows A, B, C, the displayed bind schedule f, g, K, L, the unit row u, componentwise* hsame *transport H,* Cont *replay through K,* Pkg *provenance through L, and the local* NameCert *row N. It exports no quotient Kleisli arrow, host function composition, ambient category completion, or evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadBHistBindAssociativityInduction`

Proof. Induct on the displayed finite continuation schedule. In the base case the schedule has only the source and unit rows, so Definition 485.3 and the root unit-bind exhaustion identify the two nested bind replays through u, H, K, L, N.

For the step, expose the next continuation row in the finite schedule. The existing associativity surface of Definition 485.113 rewrites the two displayed nesting orders through the same BHist, hsame, Cont, Pkg, and local NameCert rows.

Transport preserves the induction hypothesis componentwise through H, and replay composes the displayed continuation rows through K without leaving the packet. Provenance through L and local naming through N record the same finite schedule after each step.

Thus the accepted packet proves associativity only as a BEDC finite replay induction. Since no induction step introduces a quotient Kleisli arrow, host function composition, ambient category completion, or evaluator, none of those objects is exported. □

Theorem 485.115 (ContinuationMonad root generator bind coverage)**.** *Every root generator consumer of an accepted* ContinuationMonad$^{\uparrow}$ *packet reads the source rows A, B, C, bind schedule f, g, K, L, unit row u, componentwise transport H, displayed route K, provenance through* Pkg*, and local naming row N before any* Cat$^{\uparrow}$ *or Kleisli surface is exported. The consumer receives no generator, bind evaluator, quotient Kleisli arrow, host function composition, ambient category object, or monad law outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root generator bind coverage`

Proof. Start from Definition 485.67 and Definition 485.87. They force every root read to pass through the displayed histories, continuation tails, unit tail, transport row, route row, ledger row, and local name.

Apply Definition 485.101. The Kleisli-facing source is exhausted by $A, B, C, f, g, u, H, K, L, N$, so a generator consumer cannot skip directly to an external Category or Kleisli object.

Use Definition 485.97 and Definition 485.102 to account for the generator face. The generator refusal is attached to the same finite route before any category-facing read is visible, leaving no coordinate for an unlisted continuation generator or external evaluator. □

Theorem 485.116 (ContinuationMonad root downstream monad package)**.** *The root downstream monad package for an accepted* ContinuationMonad$^{\uparrow}$ *packet is assembled from the displayed source rows, unit rows, bind schedule, componentwise* hsame *transport,* Cont *routes,* Pkg *provenance,* Cat$^{\uparrow}$ *hom-carrier readings, generator-boundary refusal, and local* NameCert *row carried by $A, B, C, f, g, u, H, K, L, N$. Downstream continuation chapters may consume this package as the finite monad-reading boundary, and no quotient Kleisli category, host function composition, ambient category completion, external evaluator, or unlisted continuation generator is exported.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root downstream monad package`

Proof. First use Definition 485.46 and Definition 485.113. They cover the unit, bind, and associativity readings through u, f, g, K, L, H, N over the source histories A, B, C.

Next use Definition 485.115. It places the generator-facing consumer behind the same source, route, transport, provenance, and naming rows before any category or Kleisli surface can be consumed.

Finally apply Definition 485.48 and Definition 485.97. These package the unit, bind, generator-boundary, and Category hom faces as one downstream surface, so all external host, quotient, completion, evaluator, and unlisted-generator readings remain outside the packet. □

Theorem 485.117 (ContinuationMonad root public obligation triad)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the root public surface exposes one obligation triad: the source histories* A, B, C*, the continuation tails* f, g, u*, and the NameCert rows* H, K, L, N*. Any downstream root consumer that reads the* BHist*,* Cont*,* Pkg*, or* NameCert *face of* Z *receives that triad before it can consume a Category, generator-boundary, or Kleisli route.*

Proof. Start with Definition 485.6 and Definition 485.87. They expose $A, B, C, f, g, u, H, K, L, N$ as the finite source read of the accepted packet.

Next use Definition 485.56. It identifies the Category and generator-boundary faces as readings of the same displayed source histories, continuation tails, transports, routes, ledger, and local name.

Finally apply Definition 485.58. Since every public root consumer is routed through the same BHist, Cont, Pkg, and NameCert rows, the triad is the public obligation surface and not a host Kleisli object or an ambient category object. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public obligation triad`

Theorem 485.118 (ContinuationMonad root public generator boundary)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, a public root generator read is admitted only through the displayed continuation tails* f, g, u*, route row* K*, ledger row* L*, package provenance, and local* NameCert *row* N*. The boundary exports no unit rule, bind rule, continuation generator, host evaluator, quotient Kleisli arrow, or ambient hom set outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public generator boundary`

Proof. Read the root source through Definition 485.101. It exhausts the source by $A, B, C, f, g, u, H, K, L, N$ before any generator-facing route is visible.

Apply the generator boundary readback of Definition 485.102. The generator face is the authorized refusal boundary attached to the same finite packet.

Use Definition 485.115 to close the public boundary. It places generator consumers behind the displayed source, bind schedule, unit row, transports, route, provenance, and local name, leaving no coordinate for an unlisted generator or host Kleisli operation. □

Theorem 485.119 (ContinuationMonad root public Kleisli readback)**.** *Every root public Kleisli readback of an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *is total on the displayed finite packet: it reads the source histories, continuation tails, unit tail, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, ledger, and local* NameCert *row before any downstream monad consumer receives the surface. It exports no host function composition, quotient Kleisli category, ambient category completion, external evaluator, or unlisted continuation rule.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public kleisli readback`

Proof. Begin with Definition 485.113. It identifies unit, bind, and associativity reads as the finite replay surface over $A, B, C, f, g, u, H, K, L, N$.

Next use Definition 485.117. The public source exposure, continuation tails, and NameCert rows are already fixed before a downstream consumer reads the Kleisli face.

Finally apply Definition 485.116. The package consumes the same finite BHist, hsame, Cont, Pkg, Category hom, generator boundary, and local NameCert rows, so the public Kleisli readback is total on the displayed packet and has no external evaluator or quotient-category coordinate. □

Theorem 485.120 (ContinuationMonad root Cont-associativity ledger)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, any root public consumer that reads a composite Kleisli route must ledger associativity only through the displayed continuation tails* f, g, u*, source histories* A, B, C*, transport row* H*, route row* K*, ledger row* L*, and local name* N*. The route exports no host function composition, quotient Kleisli category, ambient hom set, or external evaluator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root cont associativity ledger`

Proof. Begin with Definition 485.117. It places every root public consumer behind the displayed source histories, continuation tails, and local NameCert rows of Z.

Next apply Definition 485.118. The generator-facing route is admitted only through the same displayed tails, route row, ledger row, provenance, and local name, so the public consumer has no independent generator or host evaluator coordinate.

Use Definition 485.119 to read the composite Kleisli surface. It makes the unit row, bind row, componentwise hsame transport, displayed Cont route, Pkg provenance, ledger, and local NameCert row total on the finite packet.

Finally invoke Definition 485.116. The downstream monad package consumes the same finite BHist, hsame, Cont, Pkg, Category hom, generator boundary, and local NameCert rows. Therefore associativity is a ledger row inside $A, B, C, f, g, u, H, K, L, N$, not an export of host function composition, a quotient Kleisli category, an ambient hom set, or an external evaluator. □

Theorem 485.121 (ContinuationMonad root public route exactness)**.** *For every accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the root public route exposed to downstream readers is exactly the displayed source histories, continuation tails, unit tail, transport row, route row, ledger row, package provenance, and local naming row. It has no additional coordinate for host function composition, quotient Kleisli arrows, ambient hom sets, or external evaluators.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public route exactness`

Proof. Begin with the root public obligation triad, Definition 485.117. It exposes $A, B, C, f, g, u, H, K, L, N$ before any Category, generator, or Kleisli read is available.

Next use the generator boundary and Kleisli readback theorems, Definition 485.118 and Definition 485.119. They place both generator-facing and Kleisli-facing consumers behind the same finite root packet.

Finally apply the Cont-associativity ledger, Definition 485.120. Since associativity is recorded inside the displayed continuation and ledger rows, the public route is exact on $A, B, C, f, g, u, H, K, L, N$ and exports none of the excluded host or quotient surfaces. □

Theorem 485.122 (ContinuationMonad root consumer totality)**.** *Every downstream consumer that reads the root public face of an accepted* ContinuationMonad$^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *first consumes the displayed* BHist*,* hsame*,* Cont*,* Pkg*, ledger, and local* NameCert *rows. Category reads, generator-boundary reads, and Kleisli readbacks are therefore total only after the route* $A, B, C, f, g, u, H, K, L, N$ *is available.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root consumer totality`

Proof. Start from root public route exactness, Definition 485.121. It identifies the complete finite route exposed by the packet.

The generator-boundary side is handled by Definition 485.118; the Kleisli-readback side is handled by Definition 485.119. Together they show that no downstream consumer can read Category, generator, or Kleisli data before the root packet has supplied the displayed rows.

Finally use Definition 485.116. The downstream monad package consumes the same route, so consumer totality is a property of the public packet itself rather than of an ambient category or host evaluator. □

Theorem 485.123 (ContinuationMonad root public formal-target boundary)**.** *Any formal target for the root public face of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet must use exactly the displayed tuple* $Z = (A, B, C, f, g, u, H, K, L, N)$*. The tuple supplies the source histories, continuation tails, unit row, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, ledger row, and local* NameCert *row needed by the public face.*

The formal target may not introduce host function composition, quotient Kleisli arrows, ambient hom sets, or an external evaluator.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public formal target boundary`

Proof. First apply the root public obligation triad. It names the full public tuple $A, B, C, f, g, u, H, K, L, N$ and exposes the source histories, continuation tails, unit row, transport row, route row, package row, ledger row, and local naming row as the available formal surface.

Next use the generator boundary and Kleisli readback theorems. They show that generator-facing and Kleisli-facing reads are read back through the same tuple, not through host composition, quotient arrows, ambient hom sets, or an external evaluator.

Finally combine the Cont-associativity ledger, route exactness, and consumer totality. These results keep associativity, public routing, and downstream consumption inside the displayed BHist, hsame, Cont, Pkg, ledger, and local NameCert rows. Hence any Lean-facing formal target for the root public face is bounded by exactly the tuple stated above. □

Theorem 485.124 (ContinuationMonad root refusal exhaustion)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, every refused root public read is exhausted by absence of one of the displayed source-history, continuation-tail, transport, route, ledger, package, or local naming rows. The refusal surface cannot be discharged by host function composition, quotient Kleisli arrows, ambient hom sets, external evaluators, or an unlisted continuation rule.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root refusal exhaustion`

Proof. First bind refusals to the public route. By Definition 485.121, the only admitted route is the displayed packet $A, B, C, f, g, u, H, K, L, N$.

Next account for the two root readers. The generator boundary theorem, Definition 485.118, refuses unlisted generator data, while the Kleisli readback theorem, Definition 485.119, refuses host or quotient Kleisli data outside the finite packet.

Finally apply the Cont-associativity ledger and downstream package theorems, Definition 485.120 and Definition 485.116. They keep associativity and downstream monad consumption inside the same public route, so every failed read is exhausted by the displayed refusal surface. □

Theorem 485.125 (ContinuationMonad root public readiness package)**.** *For every accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$*, the root public obligation triad, generator boundary, Kleisli readback,* Cont*-associativity ledger, route exactness, consumer totality, and refusal exhaustion assemble a public readiness package over* $A, B, C, f, g, u, H, K, L, N$*. Downstream monad,*

category, and generator-boundary consumers therefore receive one finite public packet, not separate host composition, quotient-arrow, or evaluator surfaces.

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public readiness package`

Proof. Start with the root public obligation triad, Definition 485.117. It exposes the source histories, continuation tail, generator row, unit row, transport row, route row, ledger row, package row, and local NameCert row.

Next combine the public readers. The generator boundary theorem, Definition 485.118, and the Kleisli readback theorem, Definition 485.119, place both advertised reads behind the same finite packet.

Apply the Cont-associativity ledger, Definition 485.120, together with route exactness, Definition 485.121. These results keep associativity and route consumption on $A, B, C, f, g, u, H, K, L, N$.

Finally use consumer totality and refusal exhaustion, Definition 485.122 and Definition 485.124. Successful reads and failed reads are both classified by displayed rows of the same packet, which is the advertised readiness package. □

Theorem 485.126 (ContinuationMonad root public refusal totality)**.** *Every failed public read of an accepted* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *is exactly the absence of a displayed row among* $A, B, C, f, g, u, H, K, L, N$*. It is not a route through host function composition, quotient Kleisli arrows, ambient hom sets, external evaluators, or unlisted continuation rules.*

Proof. Use Definition 485.125 to reduce a public read to the single readiness packet. The only admissible surface is $A, B, C, f, g, u, H, K, L, N$.

Apply Definition 485.124. Its refusal classification already lists the displayed source-history, continuation-tail, transport, route, ledger, package, and naming rows as the exhaustive failure surface.

The excluded routes are not rows of the packet. Host function composition, quotient Kleisli arrows, ambient hom sets, external evaluators, and unlisted continuation rules therefore cannot turn a failed public read into an accepted one. □

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public refusal totality`

Theorem 485.127 (ContinuationMonad root public consumer totality)**.** *Every public* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumer of an accepted packet* $Z = (A, B, C, f, g, u, H, K, L, N)$ *must pass through the* $\mathsf{Cat}^{\uparrow}$ *hom row, the* $\mathsf{GeneratorClosure}^{\uparrow}$ *source row, the unit row, the bind row, the Kleisli composite row, the componentwise* hsame *transport, the* Cont *replay, the* Pkg *provenance, and the local* NameCert *row before it receives the public continuation-monad surface. It exports no host evaluator, quotient Kleisli category, ambient category completion, unlisted continuation generator, or monad law outside those displayed rows.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public consumer totality`

Proof. Begin with the root public obligation triad Definition 485.117. It fixes the public source histories, continuation tails, and NameCert rows of Z before any downstream consumer reads the monad-facing surface.

Use the root public generator boundary Definition 485.118 and the root category-generator factorization Definition 485.51. These place the $\mathsf{Cat}^{\uparrow}$ hom row and $\mathsf{GeneratorClosure}^{\uparrow}$ source row on the same displayed route as the source histories, routes, provenance, and local name.

Apply the root public Kleisli readback Definition 485.119. It routes the unit row, bind row, Kleisli composite row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row through the same finite packet.

The exclusions follow by row exhaustion. Since the public consumer has only the displayed rows of Z, host evaluators, quotient Kleisli categories, ambient completions, unlisted generators, and external monad-law objects remain outside the carrier. □

Theorem 485.128 (ContinuationMonad root public source lock)**.** *Any public* ContinuationMonad$^{\uparrow}$ *consumer that preserves the* Cat$^{\uparrow}$ *hom row and the* GeneratorClosure$^{\uparrow}$ *source row must preserve the same packet*

$$Z = (A, B, C, f, g, u, H, K, L, N)$$

before reading the unit, bind, or Kleisli composite rows. Changing either source lock invalidates the public ContinuationMonad$^{\uparrow}$ *consumer route and cannot be repaired by host function composition, quotient Kleisli arrows, or an ambient category completion.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root public source lock`

Proof. Begin with root public consumer totality. It places every public consumer of Z behind the Category hom row, GeneratorClosure source row, unit row, bind row, Kleisli row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Apply the category-generator NameCert source lock. The Category and generator faces are source locks for the same $A, B, C, f, g, u, H, K, L, N$ packet, so preserving both rows also preserves the source histories, continuation tails, transport, route, provenance, and local name before any monad-facing read.

Finally use the root Kleisli formal target package. The formal target has no coordinate for host composition, quotient Kleisli arrows, ambient completion, external evaluation, or an unlisted generator. Therefore changing either source lock leaves the displayed package and is not a public ContinuationMonad$^{\uparrow}$ consumer route. □

Theorem 485.129 (ContinuationMonad root category-generator NameCert source lock)**.** *The* Cat$^{\uparrow}$ *and* GeneratorClosure$^{\uparrow}$ *rows of an accepted* ContinuationMonad$^{\uparrow}$ *packet are source locks for the public Kleisli readback: before a consumer reads the unit, bind, or Kleisli composite surface, it must preserve the same source histories A, B, C, continuation tails f, g, u, transport row H, route row K, provenance row L, and local* NameCert *row N. A route that changes either source lock is not a public* ContinuationMonad$^{\uparrow}$ *consumer route.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root category generator namecert source lock`

Proof. First read the root category-generator factorization. It identifies the Cat$^{\uparrow}$ hom row and GeneratorClosure$^{\uparrow}$ source row as two faces of the same finite source route over $A, B, C, f, g, u, H, K, L, N$.

Next apply the root public generator boundary. Generator-facing consumers must remain behind the displayed continuation tails, route row, provenance, and local name, so the generator source cannot be swapped after the category row is public.

Use the root public Kleisli readback to reach the monad-facing rows. The unit, bind, and Kleisli composite surface is read only after the same source, transport, route, package, and naming rows have been preserved.

Therefore the Category and generator rows are source locks for the public readback. Any route changing one of them no longer follows the displayed packet and cannot be a public ContinuationMonad$^{\uparrow}$ route. □

Theorem 485.130 (ContinuationMonad root Kleisli formal target package)**.** *The formal target surface for root* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumers is the finite package consisting of the unit row, bind row, Kleisli associativity row, public consumer row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance,* $\mathsf{Cat}^{\uparrow}$ *hom row,* $\mathsf{GeneratorClosure}^{\uparrow}$ *source row, and local* NameCert *row carried by* $Z = (A, B, C, f, g, u, H, K, L, N)$*. This package has no coordinate for host composition, quotient Kleisli arrows, ambient category completion, external continuation evaluation, or an unlisted generator.*

Lean4 ✓ `BEDC.Derived.ContinuationMonadUp.ContinuationMonadCarrier root kleisli formal target package`

Proof. Start with the root Kleisli associativity surface Definition 485.113. It reads unit, bind, and associativity as finite rows over the displayed packet rather than as host monad equations.

Combine it with the root public consumer totality above. The public consumer row is forced through the Category hom, generator source, transport, replay, provenance, and local naming rows before any downstream read is available.

Finally use the category-generator source lock above. It keeps the Category and generator faces tied to the same $A, B, C, f, g, u, H, K, L, N$ packet, so the formal target surface is exactly the finite package listed in the statement.

Row exhaustion gives the negative clauses. There is no displayed coordinate for host composition, quotient Kleisli arrows, ambient completion, external evaluation, or an unlisted generator. □

Chapter 486

A Concrete Naming Certificate for $\mathsf{GeneratorClosure}^{\uparrow}$

The $\mathsf{GeneratorClosure}^{\uparrow}$ interface records the finite naming surface for an authorized generator together with the classifier that refuses continuations outside the generator's constructor list. It is the concrete BEDC certificate side of the generator-closure classifier in Definition 9389.7: the packet exposes the generator rows, the constructor-list provenance, the continuation witnesses, and the GAP ledger that records every attempted escape from the authorized closure.

Definition 486.1 (GeneratorClosure NameCert carrier)**.** *A* $\mathsf{GeneratorClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, L, A, Q, W, H, C, P, N)$$

with the following visible coordinates:

1. *G is the authorized generator row naming the closed inductive source whose outputs are admitted;*
2. *L is the finite constructor-list provenance row for G, including the displayed closure witness for each constructor;*
3. *A is the authorized-row ledger stating which observed rows are generated by L;*
4. *Q is the generator-closure classifier row comparing authorized generator rows componentwise;*
5. *W is the continuation-witness ledger, recording for each finite continuation read whether the same constructor-list witness remains present;*
6. *H stores displayed* hsame *transports for G, L, A, Q, W and the local naming rows;*
7. *C stores the finite* Cont *routes by which consumers read authorized rows, constructor provenance, continuation witnesses, and rejected non-closure attempts;*
8. *P is the* Pkg *provenance over* BHist*, authorized generator rows, constructor-list provenance,* hsame*,* Cont*, and the* GAP *ledger;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *row.*

The carrier exports only the finite generator row, constructor-list provenance, authorized-row ledger, classifier row, continuation witnesses, transport rows, continuation routes, package provenance, local naming row, and GAP *non-escape evidence. It contains no free-floating induction principle and no universal closure over objects outside the displayed generator.*

Theorem 486.2 (GeneratorClosure classifier stability)**.** *Displayed* hsame *transport preserving the authorized generator row G, constructor-list provenance L, authorized-row ledger A, continuation ledger W, transport rows H, continuation routes C, package provenance P, and local* NameCert *row N preserves the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier Q. In particular, two authorized generator rows classified through the same constructor-list provenance retain the same accepted-row and rejected-continuation boundary.*

Proof. The carrier of Definition 486.1 makes the classifier componentwise. Transport of G and L preserves the generator source and its closed constructor list, so the authorized-row ledger A is read against the same generating surface.

Transport of W preserves the witness attached to each finite continuation read. The routes H and C then replay the same classifier reads, and the package and local naming rows repack the transported packet without adding an external induction rule. □

Theorem 486.3 (GeneratorClosure continuation witness preservation)**.** *Let K be an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *carrier. Every finite* Cont *extension that reads only the displayed authorized generator row, constructor-list provenance, authorized-row ledger, classifier row, and local naming row preserves the continuation witness in W. If the extension asks for a row without the constructor-list witness required by L, the read is recorded as a* GAP *ledger boundary rather than as a continuation of K.*

Proof. Read the extension through the displayed Cont route C. If the route lands in $G, L, A, Q, W, H, C, P, N$, the continuation witness is already one of the stored rows, so extending the read only appends a finite access path to the same witness.

If the route asks for a row outside the constructor-list provenance, the carrier has no coordinate that can supply a closure witness. The only visible output is the GAP non-escape row in the package provenance, so the attempted read is recorded as a boundary and not as a generated continuation. □

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket continuation witness preservation`

Theorem 486.4 (GeneratorClosure NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *surface for Definition 486.1 consists of five BEDC-native rows:*

1. *carrier admission over the finite* BHist *packet K;*
2. Pkg *provenance naming the authorized generator row, constructor-list provenance, classifier row, continuation-witness ledger,* hsame *transports,* Cont *routes, and local naming row;*
3. NameCert *exactness stating that the local certificate classifies exactly the displayed generator-closure packet;*
4. *constructor-list provenance stating that every accepted row is generated by the finite constructor list L and every continuation witness is read against that same list;*
5. GAP *ledger non-escape stating that rows without the required constructor-list witness are recorded as boundary attempts and are not exported as generator-internal continuations.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket namecert obligations`

Proof. Carrier admission is projection of the tuple in Definition 486.1. The generator row, constructor list, authorized ledger, classifier row, continuation witnesses, transports, routes, provenance, and local naming row are all visible coordinates of one finite BHist packet.

The package and naming rows identify the same packet. Classifier stability from Definition 486.2 supplies the transport behavior, while continuation witness preservation from Definition 486.3 supplies the finite extension behavior.

The constructor-list and GAP clauses are the two remaining projections. The former says every accepted row is read through L; the latter says a read lacking that witness is recorded as non-escape evidence rather than as an admitted continuation. □

Theorem 486.5 (GeneratorClosure carrier obligation)**.** *For a finite* BHist *packet* $K = (G, L, A, Q, W, H, C, P, N)$ *as in Definition 486.1, carrier admission for* GeneratorClosure$^{\uparrow}$ *is exactly the displayed generator row* G*, constructor-list provenance* L*, authorized-row ledger* A*, generator-closure classifier row* Q*, continuation-witness ledger* W*, displayed* hsame *transports* H*, finite* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. Every generated closure witness exported by the carrier is produced by a row of* L *and recorded in* A *or* W*.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket carrier obligation`

Proof. Project the tuple in Definition 486.1. The only generator source is G, the only constructor-list provenance is L, and the only accepted-row ledger is A.

Now read any exported closure witness through the displayed route C. If the read is accepted, the route reaches L and records the result in A or W; if it is not accepted, the carrier has no other row that can supply generated closure data.

The rows H, P, N transport and name the same packet. They do not add a second generator, an external induction rule, or a closure witness outside the displayed constructor list. □

Theorem 486.6 (GeneratorClosure classifier obligation)**.** *Displayed* hsame *transport for an accepted* GeneratorClosure$^{\uparrow}$ *carrier preserves generator endpoints and generated closure rows. In particular, if two packets transport the same* $G, L, A, Q, W, H, C, P, N$ *surface, then the* GeneratorClosure$^{\uparrow}$ *classifier preserves the endpoints named by* G *and the generated rows recorded by* A *and* W*.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket classifier obligation`

Proof. Apply the classifier stability theorem Definition 486.2. Transport of G preserves the generator endpoint, and transport of L preserves the constructor-list provenance against which generated rows are checked.

The accepted rows in A and the continuation witnesses in W are transported along the same displayed hsame rows H. The continuation routes C, package provenance P, and local naming row N then repack the transported data as the same classifier-facing packet.

Thus the classifier can move only the displayed generator endpoint and generated closure rows; it cannot introduce an endpoint or generated row outside the transported carrier. □

Theorem 486.7 (GeneratorClosure ledger obligation)**.** *Every exported* GeneratorClosure$^{\uparrow}$ *read factors through the displayed generator row* G*, constructor-list provenance* L*, authorized-row ledger* A*, continuation-witness ledger* W*, displayed* hsame *transport* H*, finite* Cont *route* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. A read without the constructor-list witness required by* L *is recorded by the* GAP *non-escape boundary and is not exported as a generated closure row.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket ledger obligation`

Proof. Start with the carrier obligation of Definition 486.5. It identifies every exported closure witness with a finite read through $G, L, A, Q, W, H, C, P, N$.

Use continuation witness preservation from Definition 486.3. A route that keeps the constructor-list witness remains inside W and is therefore exported through the same packet.

For a route without that witness, the carrier exposes no generated closure coordinate. The read is therefore recorded as a GAP boundary by the package provenance and cannot be repacked by N as an accepted generated row. □

Theorem 486.8 (GeneratorClosure authorized-row exhaustion)**.** *Every exported generated row of an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet factors through the displayed generator row G, constructor-list provenance L, authorized-row ledger A, continuation-witness ledger W, and the finite constructor-list provenance carried by L. No generated row is exported from a rule outside the displayed constructor list.*

Proof. Begin with Definition 486.5. It identifies the only carrier rows that can export generated closure data and places every exported closure witness in the finite packet $G, L, A, Q, W, H, C, P, N$.

Use Definition 486.2 to transport the authorized generator row and constructor-list provenance componentwise. Since classifier transport preserves the same G and L surface, it cannot create a second source of generated rows.

Finally apply Definition 486.7. Its ledger boundary sends every accepted read through A or W and records every read without the required constructor-list witness as a GAP boundary. □

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket authorized row exhaustion`

Theorem 486.9 (GeneratorClosure Cont route GAP separation)**.** *Every displayed* Cont *route of an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet either preserves the constructor-list witness carried by L or lands in the displayed* GAP *boundary. A continuation route never becomes a free induction rule and never exports a generated row without the constructor-list witness.*

Proof. Read the route through Definition 486.3. A finite extension that keeps the displayed generator row, constructor-list provenance, authorized ledger, classifier row, and local name preserves the witness already stored in W.

If the route lacks that witness, the same theorem records the attempted read as a GAP ledger boundary. The carrier has no coordinate for an external induction rule, so the missing witness cannot be recovered from another row.

Apply Definition 486.4 to repack the two outcomes as the local certificate surface. The Cont route is therefore separated into witness-preserving reads and displayed GAP reads. □

Theorem 486.10 (GeneratorClosure NameCert public consumer boundary)**.** *The public consumer boundary for an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet is exhausted by the displayed* BHist *carrier, componentwise* hsame *transports, finite* Cont *routes,* Pkg *provenance,* GAP *non-escape rows, and the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *row. Every public read of a generated row factors through the authorized generator row G, constructor-list provenance L, authorized-row ledger A, continuation witness W, and the separated* GAP *boundary.*

Proof. First use Definition 486.8. It fixes the generated-row surface to G, L, A, W and the finite constructor-list provenance, so a public consumer has no generated-row source outside the displayed packet.

Next use Definition 486.9. It separates every continuation read into a witness-preserving route or a displayed GAP boundary, excluding free induction rules from the consumer surface.

Now combine Definition 486.4 with Definition 486.7. The NameCert rows package carrier admission, provenance, exactness, constructor-list reads, and GAP non-escape, while the ledger theorem routes every exported read through the same finite rows. □

Theorem 486.11 (GeneratorClosure public bridge schema handoff)**.** *The public* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet hands off to downstream schema readers only through the displayed rows* $G, L, A, Q, W, H, C, P, N$*. Thus a downstream schema may read the authorized generator row, constructor-list provenance, authorized-row ledger, classifier row, continuation-witness ledger, displayed* hsame *transports, finite* Cont *routes,* Pkg *provenance, local* NameCert *row, and displayed* GAP *boundary, but it receives no second generator source, no independent constructor rule, and no host-side recursor.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket public bridge schema handoff`

Proof. Start from the carrier obligation in Definition 486.5. It identifies the public packet with the finite BHist coordinates $G, L, A, Q, W, H, C, P, N$, so every schema-facing read begins from the displayed carrier.

Apply the classifier obligation in Definition 486.6. The classifier preserves only the endpoints named by G and the generated rows recorded by A and W under the same constructor-list provenance L.

Use Definition 486.3 for finite continuation reads, and then apply the ledger obligation in Definition 486.7. A read that carries the required constructor-list witness remains inside W; a read without that witness is recorded by the displayed GAP boundary. These are exactly the rows available to the downstream schema. □

Theorem 486.12 (GeneratorClosure bridge carrier obligation)**.** *The public bridge out of an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet consumes exactly the carrier row already fixed by the chapter. Its carrier-facing read is the authorized generator row* G*, constructor-list provenance* L*, authorized-row ledger* A*, classifier row* Q*, continuation-witness ledger* W*, componentwise* hsame *rows* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *row* N*, and the displayed* GAP *boundary; it receives no second generator source, independent constructor rule, or host-side recursor.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosureUp StdBridge`

Proof. Begin with the carrier obligation Definition 486.5. It identifies the public packet with the finite carrier rows $G, L, A, Q, W, H, C, P, N$ and leaves no coordinate for a second generator source.

Apply the public bridge schema handoff Definition 486.11. The handoff permits downstream schema readers to consume only the same displayed carrier, transport, continuation, provenance, local name, and boundary rows.

Thus the bridge carrier obligation is not an additional carrier construction. It is the carrier obligation read through the public bridge boundary, with the same constructor-list provenance and the same GAP non-escape row. □

Theorem 486.13 (GeneratorClosure bridge classifier obligation)**.** *The public bridge classifier for an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet consumes exactly the authorized generator row, constructor-list provenance, classifier row, continuation-witness ledger, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *row, and* GAP *rows already fixed by the chapter. Every accepted bridge classification factors through* G, L, Q, W, H, C, P, N*, and every read without the constructor-list witness remains a displayed* GAP *boundary.*

Proof. Use the classifier obligation Definition 486.6. It transports the accepted generator endpoint and generated-row surface through the same displayed constructor-list provenance and carrier rows.

The public bridge schema handoff Definition 486.11 exposes no classifier row other than Q and no constructor provenance other than L. Hence a bridge classification cannot be read from an external predicate or host recursor.

Finally use Definition 486.3 to separate witnessed and unwitnessed continuation reads. Witnessed reads remain in the continuation-witness ledger W, while unwitnessed reads stay in the displayed GAP boundary. □

Theorem 486.14 (GeneratorClosure bridge ledger obligation)**.** *The public bridge ledger for an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet consumes exactly the authorized-row ledger, continuation-witness ledger, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *row, and* GAP *rows already fixed by the chapter. A bridge read is accepted only through A or W with the constructor-list provenance supplied by L; otherwise the read is recorded as a displayed* GAP *row and is not exported as generated data.*

Proof. Start from the ledger obligation Definition 486.7. It records the accepted generated rows and continuation witnesses in A and W, and it sends reads lacking the constructor-list witness to the displayed boundary.

Combine this with the public bridge schema handoff Definition 486.11. The bridge reads only the authorized generator row, constructor-list provenance, classifier, continuation ledger, transport, continuation routes, package provenance, local name, and GAP rows named there.

Therefore the ledger exported to downstream bridge consumers is exactly the chapter ledger. The bridge does not introduce an unledgered generator continuation, a free induction rule, or an additional row from which generated data could be recovered. □

Theorem 486.15 (GeneratorClosure fixed-point consumer obligation)**.** *Every* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *consumer of a* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet must read the fixed witness through the* $\mathsf{GeneratorClosure}^{\uparrow}$ *constructor-list provenance and classifier surface. Concretely, the fixed witness is admissible only when its read factors through $G, L, A, Q, W, H, C, P, N$ and the constructor-list row L supplies the witness required by the classifier row Q.*

This fixed-witness consumer surface is the sibling handoff to chapter 556; the $\mathsf{GeneratorClosure}^{\uparrow}$ *constructor-list provenance and classifier rows are the specific dependency consumed by that chapter.*

Proof. The carrier obligation Definition 486.5 fixes the generator-closure packet available to the consumer. Hence the consumer cannot obtain a fixed witness from a row outside $G, L, A, Q, W, H, C, P, N$.

The classifier obligation Definition 486.6 transports the generator endpoint and generated rows along the same displayed surface. A fixed-point read therefore uses the classifier row Q only together with the constructor-list provenance L and the accepted-row or continuation-witness ledgers A and W.

Finally, continuation witness preservation from Definition 486.3 and the ledger obligation Definition 486.7 separate the two possible outcomes. With the constructor-list witness, the fixed witness is read from the displayed continuation surface; without that witness, the attempted read is a GAP boundary and is not a fixed witness for $\mathsf{GeneratorFixedPoint}^{\uparrow}$. □

Theorem 486.16 (GeneratorClosure downstream ledger exhaustion)**.** *Every downstream generator consumer of the public* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet factors through the displayed* BHist *carrier,* hsame *transport rows,* Cont *routes,* Pkg *provenance, local* NameCert *row, constructor-list provenance, and* GAP *ledger. No downstream consumer receives a free-floating induction principle, an unledgered generator continuation, or a host recursor exported outside those rows.*

Proof. Use Definition 486.5 to place the consumer read in the finite BHist packet. The only visible generator source is G, the only constructor-list provenance is L, and the only public ledgers for generated rows and continuation witnesses are A and W.

Apply Definition 486.6 to transport the classifier-facing endpoints and generated rows along H. The Cont routes C, package provenance P, and local certificate row N repack the same transported rows and do not expose another induction principle.

It remains to exhaust failed or boundary reads. By Definition 486.3, a continuation lacking the constructor-list witness cannot remain a generated continuation. The ledger obligation Definition 486.7 records such a read as a displayed GAP boundary, so every downstream read is accounted for by the public ledger surface. □

Theorem 486.17 (GeneratorClosure fixed-point acceptance route)**.** *Every accepted fixed witness consumed by* GeneratorFixedPoint$^{\uparrow}$ *from the public* GeneratorClosure$^{\uparrow}$ *packet routes through the displayed rows $G, L, A, Q, W, H, C, P, N$. The constructor-list provenance row L supplies the witness that the fixed point is generated by the authorized constructor list, and the classifier row Q is the only row that classifies the fixed witness as accepted rather than as a* GAP *boundary.*

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket fixedpoint acceptance route`

Proof. Begin with the fixed-point consumer obligation of Definition 486.15. It already restricts every GeneratorFixedPoint$^{\uparrow}$ read to the carrier rows $G, L, A, Q, W, H, C, P, N$ and requires L and Q to be read together.

Use the carrier, classifier, and ledger obligations of Definition 486.5, Definition 486.6, and Definition 486.7. The carrier identifies the only available rows, the classifier preserves the generated endpoint and fixed-witness surface, and the ledger sends reads without the constructor-list witness to the displayed GAP boundary.

Finally apply Definition 486.8 and Definition 486.9. These exhaust the authorized generated rows and separate witness-preserving routes from rejected routes, so an accepted fixed witness has exactly the stated acceptance route. □

Theorem 486.18 (GeneratorClosure fixed-point continuation-route exhaustion)**.** *Every* GeneratorFixedPoint$^{\uparrow}$ *read through a public* GeneratorClosure$^{\uparrow}$ *packet factors through the displayed surface*

$$G, L, A, Q, W, H, C, P, N.$$

If the continuation route carries the constructor-list witness supplied by L, then the read is accepted through the authorized-row ledger A or the continuation-witness ledger W. Otherwise the read is recorded as the displayed GAP *boundary and is not exported as a fixed-point continuation.*

Proof. Begin with Definition 486.15. It places every GeneratorFixedPoint$^{\uparrow}$ consumer inside the constructor-list provenance and classifier surface of the public GeneratorClosure$^{\uparrow}$ packet.

Use Definition 486.17. When the route carries the constructor-list witness from L, the fixed witness is classified through Q and read through the accepted ledgers A or W.

It remains to account for routes without that witness. Apply Definition 486.9 to separate witness-preserving routes from rejected routes, and then apply Definition 486.16 to place every rejected read in the displayed GAP boundary of the same public ledger. □

Theorem 486.19 (GeneratorClosure induction non-escape boundary)**.** *Every downstream generator consumer receives displayed constructor-list provenance and displayed* GAP *rejection rows from* GeneratorClosure$^{\uparrow}$*. It receives neither a free induction principle, nor a host recursor, nor an unledgered generator continuation outside $G, L, A, Q, W, H, C, P, N$.*

Proof. Start from downstream ledger exhaustion in Definition 486.16. It places every downstream consumer inside the displayed carrier, transport rows, continuation routes, package provenance, local certificate row, constructor-list provenance, and GAP ledger.

The carrier and ledger obligations of Definition 486.5 and Definition 486.7 identify the constructive boundary. Accepted generated rows are read through L, A, W, while reads lacking the constructor-list witness are recorded by the boundary ledger.

The route separation theorem Definition 486.9 excludes the remaining escape. A continuation route either preserves the displayed witness or lands in GAP, so the consumer cannot turn the route into a free induction principle or host-side recursor. □

Theorem 486.20 (GeneratorClosure authorized consumer factorization)**.** *Every authorized generator consumer of the public* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet factors through the carrier, the classifier row, the continuation-witness surface, the* GAP *ledger, the package provenance, and the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosure}^{\uparrow}}$ *rows. Concretely, its read is a finite route through* $G, L, A, Q, W, H, C, P, N$*, with* L *witnessing constructor-list provenance,* Q *classifying accepted generated rows,* W *recording accepted continuations,* P *recording package provenance, and* N *naming the local certificate.*

Proof. Use the authorized-row exhaustion theorem Definition 486.8. It fixes the generated-row source to the displayed generator row, constructor-list provenance, authorized-row ledger, and continuation-witness ledger.

Apply Definition 486.15 for fixed-witness consumers and Definition 486.16 for downstream consumers. Both routes pass through the same finite packet, so there is no separate consumer interface outside $G, L, A, Q, W, H, C, P, N$.

It remains only to name the factors. The classifier obligation Definition 486.6 supplies Q, the ledger obligation Definition 486.7 supplies the GAP boundary, and the carrier obligation Definition 486.5 supplies H, C, P, N as displayed support rows. Hence every authorized consumer factors through the stated public surface. □

Theorem 486.21 (GeneratorClosure constructor provenance determinacy)**.** *For an accepted* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet, the constructor-list provenance row* L *determines the admitted generator endpoint and the generated rows recorded in* A *and* W*. Hence two accepted reads with the same displayed* L *cannot disagree about the authorized constructor provenance while remaining inside the carrier; any unwitnessed route is recorded at the displayed* GAP *boundary.*

Proof. Project the carrier by Definition 486.5. The packet exposes the generator row G, constructor-list provenance row L, authorized generated-row ledger A, continuation-witness ledger W, classifier row Q, transport row H, continuation routes C, package provenance P, and local certificate row N.

Apply the classifier obligation Definition 486.6 and authorized-row exhaustion Definition 486.8. Once L is fixed, the accepted generator endpoint and generated rows are read through the same displayed A and W ledgers.

Finally apply the ledger obligation Definition 486.7. A route that lacks the displayed constructor-list witness is not an alternate accepted provenance read; it is forced into the GAP boundary. Therefore equal displayed L rows cannot support two different authorized constructor provenances inside the accepted packet. □

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket constructor provenance determinacy`

Theorem 486.22 (GeneratorClosure GAP boundary absorbs unwitnessed route)**.** *Every* Cont *route of the public* $\mathsf{GeneratorClosure}^{\uparrow}$ *packet that lacks the constructor-list witness supplied by* L

is absorbed by the displayed GAP *boundary. Such a route cannot later be exported as an accepted generated-row read through A or as an accepted continuation-witness read through W.*

Proof. Use the carrier obligation Definition 486.5 to confine the route to the displayed rows $G, L, A, Q, W, H, C, P, N$. There is no independent continuation surface outside the carrier packet.

The classifier obligation Definition 486.6 and ledger obligation Definition 486.7 separate witnessed and unwitnessed routes. A route with the L witness may be read by the accepted ledgers, while a route without that witness is recorded at the displayed GAP boundary.

It remains to exclude later export. The fixed-point consumer obligation Definition 486.15 and authorized consumer factorization Definition 486.20 require any downstream accepted read through A or W to carry the same displayed constructor-list provenance. The unwitnessed route has already been classified as boundary evidence, so it cannot be repacked as an accepted generated row or accepted continuation witness. □

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket gap boundary absorbs unwitnessed route`

Theorem 486.23 (GeneratorClosure fixed-point consumer route determinacy)**.** *Every* GeneratorFixedPoint$^{\uparrow}$ *witness consumed from the public* GeneratorClosure$^{\uparrow}$ *packet is determined by the displayed route through G, L, Q, W. Once the generator row G, constructor-list provenance row L, classifier row Q, and continuation-witness ledger W are fixed up to componentwise* hsame*, neither a host recursor nor a second generator can vary the consumed fixed witness.*

Proof. Start from Definition 486.15. It states that a GeneratorFixedPoint$^{\uparrow}$ consumer reads the fixed witness only through the constructor-list provenance and classifier surface of GeneratorClosure$^{\uparrow}$.

The carrier, classifier, and ledger obligations in Definition 486.5, Definition 486.6, and Definition 486.7 identify the visible route. The row G supplies the generator source, L supplies the constructor-list witness, Q classifies the accepted fixed witness, and W records the continuation witness that remains outside GAP.

Apply authorized consumer factorization Definition 486.20. Since every authorized consumer factors through the same public surface, a host recursor or second generator would have to appear as an additional public factor. No such factor is present, so the consumed witness is determined by the displayed G, L, Q, W route. □

Lean4 ✓ `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket fixedpoint consumer route determinacy`

Theorem 486.24 (GeneratorClosure mature consumer inventory)**.** *The mature public consumer inventory of the* GeneratorClosure$^{\uparrow}$ *packet consists exactly of authorized-row consumers, bridge schema consumers, fixed-point consumers, constructor-provenance consumers, and* GAP*-boundary consumers. Each such consumer is routed through the same displayed surface $G, L, A, Q, W, H, C, P, N$: G supplies the authorized generator row, L supplies constructor-list provenance, A records accepted generated rows, Q classifies the generator-closure read, W records fixed-point and continuation witnesses, H transports componentwise* hsame *evidence, C exposes finite* Cont *routes, P records package provenance, and N names the local certificate. No other public consumer kind is admitted by the chapter surface.*

Proof. Apply authorized consumer factorization Definition 486.20. It places every authorized-row consumer on the displayed route $G, L, A, Q, W, H, C, P, N$, and it leaves no independent consumer port outside the carrier, classifier, continuation-witness, provenance, and local certificate rows.

Bridge schema consumers use the same public surface. Their bridge-facing read must pass through the accepted generated-row ledger A, the classifier row Q, the continuation route ledger C, package provenance P, and the local naming row N, so they are not a separate carrier family.

For constructor-provenance consumers, use Definition 486.21. Once L is fixed, the admitted generator endpoint and the generated rows recorded in A and W are fixed inside the accepted packet.

For GAP-boundary consumers, use Definition 486.22. A route lacking the constructor-list witness is absorbed by the displayed boundary and cannot later become an accepted generated-row read or accepted continuation-witness read.

For fixed-point consumers, use Definition 486.23. The consumed fixed witness is determined by the displayed G, L, Q, W route, with H, C, P, N carrying transport, route exposure, package provenance, and the local certificate. These five cases exhaust the public consumer inventory because every admitted read has already been factored through the same packet surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorClosure}\uparrow}$.

Constructive story: A $\mathsf{GeneratorClosure}^{\uparrow}$ packet is generated from an authorized generator row, finite constructor-list provenance, authorized-row ledger, generator-closure classifier row, continuation-witness ledger, displayed hsame transport, finite Cont routes, Pkg provenance, a local NameCert row, and GAP non-escape evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorClosure}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The scoped surface binds the displayed BHist carrier, authorized generator rows, constructor-list provenance, hsame transport, Cont routes, Pkg provenance, local $\mathsf{NameCert}_{\mathsf{GeneratorClosure}\uparrow}$ rows, classifier stability, continuation witness preservation, and GAP non-escape evidence for authorized generator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratorClosure}\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.GeneratorClosureUp.GeneratorClosurePacket\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close free-floating induction, universal quantification over objects outside the displayed generator, host recursor correctness, or bridge checking.

Upgrade path: No further formal-axis bridge upgrade is pending for the authorized generator packet.

Chapter 487

A Concrete Naming Certificate for $\mathsf{ClosureWitnessLedger}^{\uparrow}$

The $\mathsf{ClosureWitnessLedger}^{\uparrow}$ interface records the finite ledger side of generator-internal closure. Where $\mathsf{GeneratorClosure}^{\uparrow}$ of chapter 486 supplies the authorized generator and classifier surface, $\mathsf{ClosureWitnessLedger}^{\uparrow}$ isolates the witness packets and GAP non-escape rows consumed by that surface: each accepted continuation row names the constructor-list witness it reuses, and each read without such a witness is stored as a GAP non-escape row.

Definition 487.1 (ClosureWitnessLedger carrier)**.** *A* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, L, W, Q, E, H, C, P, N)$$

with the following visible coordinates:

1. *G is the authorized generator row whose closure supplies the admissible constructor surface;*
2. *L is the finite constructor-list provenance row for G;*
3. *W is the closure-witness ledger, with one finite witness packet for each continuation read that remains inside L;*
4. *Q is the witness-classifier row comparing continuation reads by the displayed generator row, constructor-list provenance, and witness packet;*
5. *E is the* GAP *non-escape ledger, recording every attempted continuation read that lacks the constructor-list witness required by L;*
6. *H stores displayed* hsame *transports for G, L, W, Q, E and the local naming rows;*
7. *C stores the finite* Cont *routes by which consumers read generator rows, constructor-list provenance, witness packets, classifier rows, and non-escape rows;*
8. *P is the* Pkg *provenance over* BHist*, generator rows, constructor-list provenance, closure-witness packets,* hsame*,* Cont*, and* GAP *boundaries;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosureWitnessLedger}^{\uparrow}}$ *row.*

The carrier exports only the displayed generator row, constructor-list provenance, finite witness ledger, classifier row, non-escape ledger, transport rows, continuation routes, package provenance, and local naming row. It does not export an induction rule over all future continuations.

Lean4 (definition) `BEDC.Derived.ClosureWitnessLedgerUp.ClosureWitnessLedgerCarrier`

Theorem 487.2 (ClosureWitnessLedger classifier stability)**.** *Displayed* hsame *transport preserving the generator row G, constructor-list provenance L, closure-witness ledger W, non-escape ledger E, transport rows H, continuation routes C, package provenance P, and local* NameCert *row N preserves the* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *classifier Q. Thus two finite continuation reads classified through the same displayed generator and constructor-list provenance retain the same witness-present or* GAP*-boundary status.*

Lean4 ✓ `BEDC.Derived.ClosureWitnessLedgerUp.ClosureWitnessLedgerCarrier classifier stability`

Proof. The classifier is componentwise over the packet in Definition 487.1. Transport of G and L preserves the authorized generator surface and the finite constructor-list provenance against which every witness packet is read.

Transport of W preserves the witness attached to an accepted continuation read. Transport of E preserves the corresponding absence row for a read without the required constructor-list witness, and H, C, P, N repack the same finite routes and naming row without adding any external closure principle. □

Theorem 487.3 (ClosureWitnessLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosureWitnessLedger}^{\uparrow}}$ *surface for Definition 487.1 consists of five BEDC-native rows:*

1. *carrier admission over the finite* BHist *packet K;*
2. Pkg *provenance naming the generator row, constructor-list provenance, witness ledger, classifier row, non-escape ledger,* hsame *transports,* Cont *routes, and local naming row;*
3. NameCert *exactness stating that the local certificate classifies exactly the displayed closure-witness ledger packet;*
4. *witness coverage stating that each accepted continuation read in W carries a constructor-list witness from L for the same generator row G;*
5. GAP *ledger non-escape stating that each read in E lacks the displayed witness required by L and is not exported as a generator-internal continuation.*

Proof. Carrier admission is projection of the displayed tuple. The generator row, constructor-list provenance, witness ledger, classifier row, non-escape ledger, transports, routes, provenance, and local naming row are coordinates of one finite BHist packet.

The package and naming rows identify that same packet. Classifier stability from Definition 487.2 supplies the transport behavior required by the NameCert surface.

The remaining rows are direct ledger projections. The witness-coverage row reads accepted continuations through W and L, while the GAP row reads rejected continuations through E and records them as boundaries rather than as generated closure content. □

Theorem 487.4 (ClosureWitnessLedger handoff to GeneratorClosure)**.** *Let K be an accepted* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *carrier and let J be a* $\mathsf{GeneratorClosure}^{\uparrow}$ *carrier with the same displayed generator row G and constructor-list provenance L. If the* Cont *route of J asks for a continuation witness, then the answer is read from W when the witness is present and from E as a* GAP *boundary when it is absent. Therefore the* $\mathsf{GeneratorClosure}^{\uparrow}$ *continuation-witness coordinate can be supplied by the concrete* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *packet without changing the generator classifier.*

Proof. The shared G and L make both packets read continuation rows against the same finite constructor-list provenance. A Cont route in J that requests a continuation witness is therefore a route of the same displayed shape as the corresponding route in K.

If the route has a witness packet, the coordinate W supplies it and the classifier row Q records that the read remains inside the generator closure. If the route lacks the witness required by L, the coordinate E supplies the only visible row, namely the GAP boundary.

Repacking these two cases into the continuation-witness coordinate of J changes no generator row and no constructor-list provenance. The handoff is a finite ledger substitution, not an additional induction principle. □

Theorem 487.5 (ClosureWitnessLedger GAP non-escape)**.** *For any accepted* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *carrier K, a finite continuation read recorded in the* GAP *ledger E cannot also be exported by K as an accepted witness packet in W for the same generator row G and constructor-list provenance L. Consequently every rejected read remains a boundary row and never becomes generator-internal closure content by certificate transport alone.*

Proof. Read the attempted continuation through its displayed route in C. The carrier has exactly two ledger outputs for such a route: a witness packet in W when the constructor-list witness from L is present, or a GAP row in E when that witness is absent.

The classifier stability theorem transports this distinction without changing G or L. A row already classified through E has no witness coordinate to project, so package provenance can only preserve it as a boundary row.

Thus certificate transport cannot move an absent-witness read from E into W. Such a move would require a new constructor-list witness not present in the finite packet, and the carrier exposes no coordinate that can create one. □

Theorem 487.6 (ClosureWitnessLedger continuation-witness exactness)**.** *For an accepted* $\mathsf{ClosureWitnessLedger}^{\uparrow}$ *packet $K = (G, L, W, Q, E, H, C, P, N)$, every finite continuation read through C has exactly one public classification: either a witness packet in W against the displayed constructor-list provenance L, or a* GAP *row in E. Componentwise* hsame *transport through H preserves this split, and no certificate transport can move a row from E to W without a displayed witness from L.*

Proof. Project the carrier of Definition 487.1. The continuation routes in C read against the same generator row G and constructor-list provenance L, so the classifier row Q is evaluated on one displayed ledger surface.

Apply Definition 487.4. A continuation read with the required constructor-list witness is supplied from W, while a read without that witness is supplied from E as a GAP boundary.

The two cases are exclusive by Definition 487.5. A row already recorded in E cannot also be exported as a witness packet in W for the same G and L.

Transport through H is componentwise over G, L, W, Q, E and the local naming rows. It preserves the witness-vs-boundary split and provides no coordinate that can create the missing constructor-list witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosureWitnessLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosureWitnessLedger}^{\uparrow}$ packet is generated from an authorized generator row, finite constructor-list provenance, a closure-witness ledger, a witness classifier row, a GAP non-escape ledger, displayed hsame transport, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosureWitnessLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the carrier, classifier stability, NameCert obligations, generator-closure handoff, and GAP non-escape boundary for finite closure-witness ledgers.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClosureWitnessLedger}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close general induction, host recursor correctness, public bridge checking, or closure over continuations outside the displayed generator and constructor-list provenance.
Upgrade path: Obligation closure requires checked carrier admission, classifier stability, NameCert exactness, generator-closure handoff, and GAP non-escape for the same finite packet.

Chapter 488

A Continuation-Ledger Certificate for $\mathsf{KernelCategory}^{\uparrow}$

The $\mathsf{KernelCategory}^{\uparrow}$ packet records the category-shaped reading of the BEDC kernel without importing a host category object. Its data are finite BHist rows: unary object histories, continuation-ledger hom histories, empty continuation identities, displayed Cont composition ledgers, hsame law transports, package provenance, and one local naming row.

This packet is the kernel-level specialization of the concrete $\mathsf{Cat}^{\uparrow}$ hom-carrier surface in chapter 61. It consumes the continuation unit and associativity surface of chapter 931 and the displayed continuation identity and composition route already packaged by $\mathsf{ContinuationMonad}^{\uparrow}$ in chapter 485. Thus $\mathsf{KernelCategory}^{\uparrow}$ reuses the sibling carrier and route rows while refusing host category objects, quotient morphisms, and any morphism equality not expressed as finite continuation-ledger history.

Definition 488.1 (KernelCategory carrier)**.** *A* $\mathsf{KernelCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, H, I, C, A, U, P, N)$$

with the following visible coordinates:

1. *O is the object table, whose rows are exactly the histories admitted by* UnaryHistory*.*
2. *H is the hom table, whose rows are continuation-ledger histories presented as* CategoryHomCarrier *and* ContinuationMorphism *data between rows of O.*
3. *I records identity rows, and these are precisely the empty continuation rows over each unary object history.*
4. *C records displayed composition rows, and each row is a* Cont*-composite ledger from composable continuation-ledger hom histories.*
5. *A records associativity rows as* hsame *transports between the two displayed triple-composite ledgers with the same endpoints.*
6. *U records left and right unit rows as* hsame *transports from the empty-continuation composites back to the displayed hom row.*
7. *P names* BHist*,* UnaryHistory*,* Cont*,* hsame*,* Pkg*,* NameCert*, and* $\mathsf{Cat}^{\uparrow}$ *as the only dependency rows used by the packet.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *row.*

The classifier for accepted packets compares only these rows, their endpoint ledgers, their displayed transports, their continuation routes, and their package and naming provenance.

Theorem 488.2 (KernelCategory identity continuations)**.** *In an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *carrier, identity homs are exactly the empty continuation rows in* I*. If endpoint rows are transported by the componentwise* hsame *data of the carrier, the transported identity row is again the empty continuation row over the transported endpoint, and it remains classified as an identity hom in the same packet.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier identity continuation`

Proof. Project the object and identity coordinates from Definition 488.1. The object coordinate admits endpoints only through UnaryHistory, and the identity coordinate has no row other than the empty continuation row over such an endpoint.

Transport an endpoint by the displayed hsame rows. Since the identity row is defined by the empty continuation ledger and has no independent tail data, transporting the endpoint transports the whole identity row to the empty continuation row over the transported object.

The package and local naming rows are unchanged by this componentwise transport. Thus the transported row is still classified by the same $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ identity coordinate. □

Theorem 488.3 (KernelCategory composition continuations)**.** *Composable hom rows in an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *carrier close under the displayed* Cont *composition rows in* C*. The resulting hom row has the source endpoint of the first input row, the target endpoint of the second input row, and no endpoint ledger outside the two input ledgers and the displayed composite route.*

Proof. Read two composable hom rows from H. By the carrier definition, each is a continuation-ledger history with endpoints already listed in the unary object table O.

The composition coordinate C records exactly the displayed Cont-composite ledger for such a composable pair. Applying that row produces a hom history whose source is inherited from the first row and whose target is inherited from the second row.

No additional endpoint source is present in K. The associativity and unit rows in A and U may transport composite ledgers by hsame, but they do not introduce new object histories or quotient morphisms. Hence composition preserves the displayed endpoint ledger. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier composition continuation`

Theorem 488.4 (KernelCategory certificate surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier admission for the finite packet* $K = (O, H, I, C, A, U, P, N)$*;*
2. *hom classifier stability under componentwise* hsame *transport of endpoints and continuation-ledger hom rows;*
3. *identity law rows asserting that the empty continuation rows in* I *are the left and right unit ledgers;*
4. *composition and associativity law rows asserting closure under* C *and triple-composite transport by* A*;*
5. *ledger non-escape, excluding host category objects, quotient morphisms, external equality of arrows, and any category primitive not named by* O, H, I, C, A, U, P, N*.*

Proof. Carrier admission is the tuple projection of Definition 488.1. It supplies the object table, hom table, identity rows, composition rows, associativity rows, unit rows, package provenance, and local naming row.

Hom classifier stability follows by componentwise transport: endpoint rows are transported inside O, hom histories are transported inside H, and the displayed continuation routes in C replay the same source and target ledgers.

Use Definition 488.2 for the identity law rows and Definition 488.3 for composition closure. The remaining associativity and unit law rows are precisely the hsame transports stored in A and U.

Finally inspect the package row P and the naming row N. They name only the BEDC history, continuation, transport, package, NameCert, and $\mathsf{Cat}^{\uparrow}$ rows listed in the carrier. A host category object or quotient morphism would require an additional coordinate, so it is outside the certificate surface. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCertificateSurface`

Theorem 488.5 (KernelCategory associativity scope)**.** *Every triple composite admitted by a* $\mathsf{KernelCategory}^{\uparrow}$ *packet is scoped by the displayed* Cont *rows in C together with the* hsame *associativity row in A. The packet reads both parenthesized composites as finite continuation-ledger histories with the same endpoints, and it has no row for a host category object, quotient morphism, or external arrow equality.*

Proof. Start with the finite packet of Definition 488.1. Its hom rows live in H, its displayed binary composite rows live in C, and its associativity transports live in A.

Apply Definition 488.3 twice. The two applications build the two parenthesized triple composites using only the displayed Cont ledgers in C, so the source and target endpoints remain the ones inherited from the three input hom rows.

The comparison between the two readings is then exactly the hsame row named in A. By Definition 488.4, the certificate has no additional coordinate for a host category primitive, quotient morphism, or external equality of arrows. Thus associativity is scoped only by the displayed continuation rows and the named transport row. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier associativity scope`

Theorem 488.6 (KernelCategory unit scope)**.** *Every left or right unit read admitted by a* $\mathsf{KernelCategory}^{\uparrow}$ *packet is exhausted by the empty-continuation identity rows in I and the unit transports in U. The read never imports a host identity arrow, quotient morphism, or external arrow equality beyond the finite continuation-ledger rows already present in K.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier unit scope`

Proof. The identity surface is fixed by Definition 488.2: identities are exactly the empty continuation rows over unary object histories. Thus a unit read can begin only with the displayed identity row in I.

Compose the displayed identity row with the displayed hom row through Definition 488.3. The resulting ledger is again a finite continuation-ledger hom row whose endpoints are inherited from the input rows.

The return from this composite ledger to the original hom row is precisely the left or right unit hsame transport listed in U. The non-escape clause of Definition 488.4 excludes host category objects, quotient morphisms, and external arrow equality, so no further unit principle is available. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryUnitScope`

Theorem 488.7 (KernelCategory public NameCert surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *surface is exactly the carrier, hom classifier, identity, composition, associativity, unit, package, and non-escape rows named by* $K = (O, H, I, C, A, U, P, N)$*. Public consumers may cite the rows exposed by Definition 488.1, Definition 488.2, Definition 488.3, Definition 488.5, Definition 488.6, and Definition 488.4, but they cannot add host category objects, quotient morphisms, or external arrow equality.*

Proof. Project the tuple $K = (O, H, I, C, A, U, P, N)$ from Definition 488.1. These coordinates supply the object table, hom classifier, identity rows, composition rows, associativity transports, unit transports, package provenance, and local naming row.

Definition 488.2 and Definition 488.3 expose the identity and composition readings. Definition 488.5 and Definition 488.6 expose the law transports without adding any host-side category structure.

Finally, Definition 488.4 states the ledger non-escape boundary. Since the public surface is the displayed NameCert surface of the same packet, a consumer receives exactly those rows and no quotient morphism, external arrow equality, or host category object. □

Theorem 488.8 (KernelCategory root NameCert obligation triad)**.** *The root* $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$ *surface is generated by three displayed obligations: carrier admission for* $K = (O, H, I, C, A, U, P, N)$*, identity-continuation admission for the empty rows in* I*, and composition-continuation admission for the displayed* Cont *rows in* C*. Any accepted kernel-category read must cite all three obligations before it can consume associativity, unit, package, or local naming rows.*

Proof. Carrier admission is the tuple projection of Definition 488.1. It exposes the object, hom, identity, composition, transport, provenance, and local naming rows as the finite packet.

Identity-continuation admission is supplied by Definition 488.2. It identifies the identity rows with empty continuation rows over unary object histories.

Composition-continuation admission is supplied by Definition 488.3. It identifies the composable hom rows with displayed Cont composites whose endpoints remain inside the input ledgers.

The certificate surface theorem, Definition 488.4, then reads the associativity, unit, package, and non-escape rows as downstream rows of the same packet. Thus the root obligation surface is exactly the carrier, identity-continuation, and composition-continuation triad. □

Theorem 488.9 (KernelCategory composition locality)**.** *Every composition read admitted by a* $\mathsf{KernelCategory}^{\uparrow}$ *packet factors through the displayed* Cont *rows in* C *and the endpoint ledgers already present in* O *and* H*. No composition read may consult a host category composition, quotient morphism, external arrow equality, or endpoint object outside the finite carrier* $K = (O, H, I, C, A, U, P, N)$*.*

Proof. Start with the carrier of Definition 488.1. Hom rows live in H, endpoints live in O, and displayed composition rows live in C.

Apply Definition 488.3. It constructs the composite hom row from two displayed continuation-ledger hom rows and the Cont composite row that joins them.

The endpoint ledgers of the composite are therefore inherited from the two input hom rows. The associativity and unit transports in A and U compare displayed continuation ledgers, while the package and naming rows P, N certify the same finite packet.

Finally apply Definition 488.4. Its non-escape clause excludes host category objects, quotient morphisms, and external equality of arrows, so composition is local to C and the endpoint ledgers in O, H. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier cont composition locality`

Theorem 488.10 (KernelCategory package provenance non-escape)**.** *The provenance row P of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet exports no host category object, quotient morphism, external arrow equality, functor semantics, or limit object outside O, H, I, C, A, U, P, N. It only records the BEDC history, unary-history, continuation, transport, package, NameCert, and category-surface rows needed to replay the finite carrier.*

Proof. Project the package row from Definition 488.1. It names BHist, $\mathsf{UnaryHistory}$, Cont, hsame, Pkg, $\mathsf{NameCert}$, and $\mathsf{Cat}^{\uparrow}$ as dependency rows for the finite packet.

Use Definition 488.2 and Definition 488.3 to identify the only identity and composition routes available to consumers. Both routes remain inside the displayed continuation-ledger rows.

Apply Definition 488.4. The ledger non-escape obligation says that a host category object, quotient morphism, external equality of arrows, or category primitive not named by O, H, I, C, A, U, P, N is outside the certificate surface.

Functor semantics and limit objects would require additional carrier rows and additional laws. Since P records only provenance for the displayed finite packet, it cannot export those structures. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier package provenance nonescape`

Theorem 488.11 (KernelCategory endpoint transport closure)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet $K = (O, H, I, C, A, U, P, N)$, componentwise endpoint* hsame *transport preserves the unary object table O and the empty-continuation identity row I. Thus an endpoint read transported along O remains an admitted object row, and the corresponding identity read remains the same local empty* Cont *row rather than a host identity arrow.*

Proof. Start from the carrier in Definition 488.1. It places object endpoints in the unary table O and identity witnesses in the empty-continuation row I.

Apply Definition 488.2. The identity row is not a host identity operation; it is the displayed empty Cont route attached to the unary object history.

The certificate surface of Definition 488.4 supplies the componentwise hsame transport rows. Transporting an endpoint therefore moves only the displayed unary object row and its empty-continuation identity row, while the non-escape clauses exclude host category objects, quotient morphisms, and external arrow equality. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier endpoint transport closure`

Theorem 488.12 (KernelCategory composite endpoint determinacy)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet $K = (O, H, I, C, A, U, P, N)$, any displayed* Cont *composite admitted by C has exactly the source endpoint of the first continuation-ledger hom row and the target endpoint of the second continuation-ledger hom row. No host endpoint coordinate, quotient morphism endpoint, or external equality of arrows is available to alter those two displayed endpoints.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier composite endpoint determinacy`

Proof. Read the composite through Definition 488.3. It constructs the composite only from two hom rows in H and the displayed Cont composite row in C.

The first hom row supplies the source endpoint and the second hom row supplies the target endpoint. The carrier of Definition 488.1 has no further endpoint coordinate outside O, H, C.

Finally apply Definition 488.5, Definition 488.6, and Definition 488.4. The law transports compare displayed continuation ledgers, and the certificate surface blocks host category endpoints, quotient morphism endpoints, and external arrow equality from participating in the endpoint read. □

Theorem 488.13 (KernelCategory law transport exhaustion)**.** *Once the carrier row O, H, I, C, P, N, the identity row I, and the composition row C of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet are fixed, every associativity or unit read is exhausted by the law-transport rows A and U. These law reads do not import host associativity, host unit equality, quotient morphisms, functor semantics, natural transformations, or limit objects.*

Proof. Fix the carrier with Definition 488.1. The rows O, H, I, C, P, N determine the displayed object table, hom ledger, identity route, composition route, provenance, and local name.

Use Definition 488.2 and Definition 488.3. They ensure that identity and composition reads are already local to empty-continuation identity rows and displayed Cont composites.

The remaining category-shaped laws are exactly the rows described by Definition 488.5 and Definition 488.6. Associativity reads pass through A, and unit reads pass through U.

Finally apply Definition 488.4. Its public surface and non-escape clauses show that no host category law, quotient morphism, external equality of arrows, functorial semantics, natural transformation, or limit object remains after A and U have been read. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier law transport exhaustion`

Theorem 488.14 (KernelCategory route totality)**.** *Every accepted* $\mathsf{KernelCategory}^{\uparrow}$ *route factors through exactly the displayed rows O, H, I, C, A, U, P, N: unary* BHist *endpoints in O, continuation-ledger hom histories in H, empty* Cont *identity rows in I, displayed* Cont *composition rows in C, componentwise* hsame *law transports in A, U,* Pkg *provenance in P, and the local* NameCert *row N. No host category object, quotient morphism, external arrow equality, host composition, functorial semantic row, natural transformation, or limit object is part of the accepted route.*

Proof. Project the carrier in Definition 488.1. It exposes the finite packet O, H, I, C, A, U, P, N as the entire KernelCategory surface, so an accepted route begins with displayed BHist endpoints, continuation-ledger hom rows, provenance, and local naming.

Use Definition 488.2 and Definition 488.3. They place identity and composition reads in the empty Cont identity row and the displayed Cont composite row, rather than in a host category operation.

Finally combine Definition 488.5, Definition 488.6, Definition 488.13, Definition 488.4, and Definition 488.10. The law theorems exhaust the componentwise hsame rows, while the certificate and package rows block host objects, quotient morphisms, external arrow equality, and detached category structure outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier route totality`

Theorem 488.15 (KernelCategory law row exactness)**.** *After the identity row I and composition row C of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet have been exposed, every law read is exactly one of the associativity row A or the unit row U. There is no third law row for host associativity, host unit equality, quotient morphism equality, external arrow equality, functorial semantics, natural transformations, or limits.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier law row exactness`

Proof. First expose identity and composition with Definition 488.2 and Definition 488.3. These theorems identify the operational rows available before any law read occurs.

Next apply Definition 488.5 and Definition 488.6. The associativity comparison is read from A, and the unit comparison is read from U.

It remains to rule out further law sources. Use Definition 488.13 and Definition 488.4. Together they say that every category-shaped law comparison has already been exhausted by A and U, and that the public surface has no coordinate for host laws, quotient equality, external arrow equality, functorial semantics, natural transformations, or limits. □

Theorem 488.16 (KernelCategory downstream NameCert readiness)**.** *Every downstream* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, or* $\mathsf{ContinuationMonad}^{\uparrow}$*-facing consumer of a* $\mathsf{KernelCategory}^{\uparrow}$ *route receives only the displayed* $\mathsf{KernelCategory}^{\uparrow}$ $\mathsf{NameCert}$ *surface* O, H, I, C, A, U, P, N*. Such a consumer may read unary* BHist *endpoints, displayed* Cont *identity and composition rows, componentwise* hsame *law transports,* Pkg *provenance, and the local name, but it cannot import external arrow equality, host object equality, quotient morphisms, host composition, functor semantics, natural transformations, or limit data.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier downstream namecert readiness`

Proof. Start from Definition 488.1. The downstream consumer is handed the same finite tuple O, H, I, C, A, U, P, N, not a separate host category carrier.

Use Definition 488.22 to identify this tuple as the NameCert root surface for category-shaped consumers. Then apply Definition 488.4 and Definition 488.10 to keep provenance and naming inside the displayed packet.

Finally apply Definition 488.2, Definition 488.3, Definition 488.5, Definition 488.6, and Definition 488.13. These results account for the operation and law rows read by downstream category-shaped consumers, leaving no route by which external arrow equality or host category structure can enter the NameCert surface. □

Theorem 488.17 (KernelCategory identity Cont route exhaustion)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, every identity read factors through the unary object row* O*, the empty* Cont *identity row* I*, componentwise* hsame *transport,* Pkg *provenance in* P*, and the local* $\mathsf{NameCert}$ *row* N*. The identity route has no coordinate for host identities, quotient morphisms, external arrow equality, functorial semantics, natural transformations, limits, or a second naming source outside the displayed packet.*

Proof. Project the packet from Definition 488.1. The only object data available to an identity read are the unary object histories in O, with package and naming support confined to P, N.

Apply Definition 488.2. It identifies the identity row as the empty Cont route over the displayed unary object history, not as a host identity arrow.

Use Definition 488.4 together with Definition 488.13. The componentwise hsame rows and law transports are already local to A and U, so identity transport does not create a new category-shaped equality source.

Finally use Definition 488.5 and Definition 488.6 only as law-boundary checks. They keep later law reads attached to the displayed continuation packet, while the identity route remains exhausted by O, I, P, N and the local certificate surface. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier identity cont route exhaustion`

Theorem 488.18 (KernelCategory composition Cont route exactness)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, a composable pair of displayed hom rows is accepted exactly through the* Cont *composite row* C *whose source endpoint is inherited from the first hom row and whose target endpoint is inherited from the second hom row. This exact route uses the hom ledger* H*, componentwise* hsame *law transports in* A, U*,* Pkg *provenance in* P*, and local* NameCert *naming in* N*, and it imports no external arrow equality or host composition operation.*

Proof. Start from Definition 488.1. The carrier gives only the unary endpoint row O, hom ledger H, composition row C, law transports A, U, package row P, and local naming row N for a composite read.

Apply Definition 488.3. The accepted composite is the displayed Cont row in C, and its endpoints are inherited from the two continuation-ledger hom rows rather than supplied by a host category.

Use Definition 488.5 and Definition 488.6. They place subsequent associativity and unit comparisons in componentwise hsame rows over the same displayed continuation data.

Finally apply Definition 488.4 and Definition 488.13. The certificate and transport exhaustion clauses leave no coordinate for quotient morphisms, external arrow equality, host composition, functorial semantics, natural transformations, or limits. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier composition cont route exactness`

Theorem 488.19 (KernelCategory Functor continuation handoff)**.** *Every* $\mathsf{Functor}^{\uparrow}$*-facing consumer of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$ *receives hom transport only through the displayed rows of* K*: unary object histories in* O*, continuation-ledger hom rows in* H*, empty-continuation identities in* I*, displayed* Cont *composites in* C*, componentwise* hsame *law transports in* A, U*,* Pkg *provenance in* P*, and local naming in* N*. The handoff to chapter 62 carries no host category object, quotient morphism, or external arrow equality.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier functor continuation handoff`

Proof. Read the finite packet through Definition 488.1. The only hom data available to a functor-facing consumer are the hom ledger H and the displayed operation rows attached to it.

Use Definition 488.18. It fixes each composite as a displayed Cont route whose endpoints are inherited from the accepted hom rows, so functorial transport has no host composition source.

Now apply Definition 488.16. Together with the sibling surfaces of chapter 61, chapter 62, and chapter 485, it presents the same finite NameCert route as the consumer handoff. □

Theorem 488.20 (KernelCategory ContinuationMonad route coverage)**.** *The* $\mathsf{ContinuationMonad}^{\uparrow}$ *unit and bind coverage read by* $\mathsf{KernelCategory}^{\uparrow}$ *is exhausted by the identity row* I *and composition row* C *of the packet* $K = (O, H, I, C, A, U, P, N)$*. Unit coverage is the empty-continuation route over unary object histories, bind coverage is displayed* Cont *composition over hom ledgers, and neither coverage route imports a host monad, host category composition, quotient morphism, or external equality.*

Proof. Start from Definition 488.1. The carrier separates identity and composition as the packet rows I and C, with endpoints and hom inputs already present in O and H.

The exactness theorem Definition 488.18 fixes the bind-shaped route as displayed Cont composition. The same packet's identity-continuation rows supply the empty-continuation unit side used by the continuation sibling.

Finally use Definition 488.16. It routes downstream reads through the local NameCert packet and through chapter 485, leaving no off-packet monad or category law to supply coverage. □

Theorem 488.21 (KernelCategory Category surface readiness)**.** *The public* $\mathsf{Cat}^{\uparrow}$ *surface read from a* $\mathsf{KernelCategory}^{\uparrow}$ *packet is ready exactly when carrier admission, displayed identity, displayed composition, law transport,* Pkg *provenance, and local* NameCert *naming have all been read from* O, H, I, C, A, U, P, N*. The readiness route cites chapter 61 only through those finite BEDC rows and admits no host category carrier, quotient arrow class, global equality of morphisms, or detached categorical semantics.*

Proof. Begin with Definition 488.1. It supplies the object, hom, identity, composition, law, provenance, and naming coordinates of the category-shaped packet.

Use Definition 488.18 for the composition side. The exact route is already a displayed continuation composite, not a host operation.

Close the readiness claim with Definition 488.16. Its downstream handoff, read together with chapter 61, chapter 62, and chapter 485, keeps every consumer route inside the finite packet and excludes the host category surface. □

Theorem 488.22 (KernelCategory NameCert root route readiness)**.** *The local* NameCert *root surface of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$ *is route-ready exactly through the displayed rows: unary* BHist *object histories in* O*, continuation-ledger hom histories in* H*, empty* Cont *identities in* I*, displayed* Cont *composites in* C*, componentwise* hsame *associativity and unit transports in* A, U*,* Pkg *provenance in* P*, and the local naming row* N*. This root surface is the one consumed by downstream category-shaped BEDC packets, and it contains no host category object, quotient morphism, external arrow equality, functorial semantic row, natural transformation, limit object, or detached global certificate.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier namecert root route readiness`

Proof. Read the packet through Definition 488.1. It presents the complete root surface as the finite tuple O, H, I, C, A, U, P, N, so downstream consumers begin from displayed BEDC rows rather than a host category carrier.

Use Definition 488.2 for identity readiness and Definition 488.3 for composition readiness. These two theorems provide the empty Cont identity route and the displayed Cont composite route needed by category-shaped consumers.

Add the law rows with Definition 488.5, Definition 488.6, and Definition 488.13. They ensure that associativity and unit reads remain componentwise hsame transports over the displayed identity and composition routes.

Finish with Definition 488.4. The certificate surface combines the local NameCert row with the Pkg provenance row, and its non-escape clauses exclude every host, quotient, external-equality, functorial, natural-transformation, limit, or global-certificate route outside the packet. □

Theorem 488.23 (KernelCategory inductive object route)**.** *Every object-facing* $\mathsf{KernelCategory}^{\uparrow}$ *consumer is generated by induction over the displayed unary object histories in* O *and the continuation-ledger hom rows in* H *of the packet* $K = (O, H, I, C, A, U, P, N)$*. The base object route is the empty-continuation identity row in* I*, and the inductive step appends a displayed hom row through the* Cont *composition row in* C*. The resulting route may use only the law transports in* A, U*, package provenance in* P*, and local naming in* N*; it never imports quotient morphisms, host category equality, or an external object classifier.*

Proof. Induct on the visible route rows read by the consumer. In the base case the route exposes only a unary object history from O. Apply Definition 488.2: the corresponding object-facing identity is exactly the empty continuation row in I, not a host identity arrow.

For the inductive step, assume the already-read route is generated from displayed unary histories and continuation-ledger hom rows. Appending one more displayed hom row is possible only through the composition coordinate C. Apply Definition 488.3 to identify this append as the displayed Cont composite whose endpoints are inherited from the input ledgers.

The law rows do not enlarge the object route. Use Definition 488.13, Definition 488.14, and Definition 488.22: associativity and unit comparisons are the rows A, U, while provenance and naming remain P, N. Thus every object-facing read is generated by the finite induction over O, H, I, C, A, U, P, N, and no quotient morphism, host category equality, or external object classifier is available. □

Theorem 488.24 (KernelCategory carrier admission obligation)**.** *For a* KernelCategory$^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, carrier admission is exactly the finite obligation surface consisting of unary object histories in* O*, continuation-ledger hom rows in* H*, empty-continuation identity rows in* I*, displayed* Cont *composites in* C*, componentwise* hsame *law transports in* A *and* U*, provenance in* P*, and local* NameCert *in* N*. It admits no host category object, quotient morphism, external arrow equality, functor semantics, natural transformation, limit object, or global naming row outside that tuple.*

Proof. Start with Definition 488.1. Its carrier tuple exposes the object, hom, identity, composition, law-transport, provenance, and local naming rows as the available finite packet.

Use Definition 488.2 for the identity side. It identifies identity admission with empty continuation rows over unary object histories already present in O.

Use Definition 488.3 for the hom side. It identifies composition admission with displayed Cont composites whose endpoints remain in the continuation-ledger hom rows of H.

Finally apply Definition 488.4 and Definition 488.10. The certificate and provenance rows expose the hsame law transports, package row, and local naming row, while their non-escape clauses exclude all host category and quotient coordinates outside K. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier carrier admission obligation`

Theorem 488.25 (KernelCategory continuation route factorization)**.** *For a* KernelCategory$^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, every admitted composition route factors through unary object histories in* O*, continuation-ledger hom rows in* H*, displayed* Cont *composites in* C*, componentwise* hsame *transports in* A *and* U*,* Pkg *provenance in* P*, and the local* NameCert *row* N*. It has no factor through host category composition, quotient morphisms, external arrow equality, functorial semantics, natural transformations, limits, or a second naming source.*

Proof. Start with Definition 488.24. It fixes O, H, I, C, A, U, P, N as the complete carrier-admission surface and leaves no composition-shaped coordinate outside that tuple.

Use Definition 488.3. Any composition read is a displayed Cont composite whose endpoints and inputs already live in the unary object and continuation-ledger hom rows.

The law part of the factorization is supplied by Definition 488.4. Associativity and unit comparisons are componentwise hsame transports over the displayed rows, not host arrow equalities.

Finally apply Definition 488.10. The Pkg and local NameCert rows record only provenance and naming for the finite BEDC packet, so every admitted route has the displayed factorization and no external category route remains. □

Theorem 488.26 (KernelCategory continuation handoff for CategoryUp)**.** *Any* Cat$^{\uparrow}$ *consumer of the* KernelCategory$^{\uparrow}$ *packet receives only the finite continuation rows already displayed in the*

category, unary continuation-monoid, and continuation-monad constructions of chapter 61, chapter 931, and chapter 485. The handoff exports object histories, hom ledgers, empty-continuation identity rows, displayed Cont *composites,* hsame *law transports,* Pkg *provenance, and local* NameCert *naming, but not host category objects, quotient morphisms, or external arrow equality.*

Proof. Begin with Definition 488.24. The consumer can read only the finite packet rows O, H, I, C, A, U, P, N.

Apply Definition 488.2. The identity rows handed to the consumer are empty-continuation rows over unary object histories, matching the unary continuation-monoid and continuation-monad sources.

Apply Definition 488.3. The composition rows handed to the consumer are displayed Cont composites of continuation-ledger hom rows, matching the category-side finite continuation surface.

Finally use Definition 488.4 and Definition 488.10. These rows package the hsame law transports, provenance, and local naming while blocking any host category object, quotient morphism, or external arrow equality from entering the $\mathsf{Cat}^{\uparrow}$ read. □

Theorem 488.27 (KernelCategory downstream consumer readiness)**.** *Downstream Functor, NatTrans, and Category consumers may cite the* $\mathsf{KernelCategory}^{\uparrow}$ *carrier, identity rows, composition rows, law-transport rows, package provenance, and non-escape rows as a complete BEDC kernel surface. Such consumers need not import host category objects, quotient morphisms, external arrow equality, functorial semantics, natural transformation objects, limit objects, adjunction data, or a global naming row beyond the local* NameCert *row.*

Proof. First read the carrier through Definition 488.1 and Definition 488.24. These fix the finite rows available to every downstream consumer.

Next use Definition 488.2 and Definition 488.3. Identity and composition are already displayed as empty-continuation rows and Cont composites, so a downstream consumer has the operation rows it needs without a host category object.

The law rows are supplied by Definition 488.4. They transport associativity and units through componentwise hsame rows, which is enough for Category, Functor, and NatTrans consumers to cite the kernel packet as their operation surface.

Finally apply Definition 488.10. The package and non-escape rows keep provenance and naming local to P, N, so the consumer has no route to quotient morphisms, external arrow equality, limits, adjunctions, or a global naming row. □

Theorem 488.28 (KernelCategory root hom-law triad)**.** *Every root consumer of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$ *must read the hom ledger* H*, the identity row* I*, and the composition row* C *before it reads associativity or unit law rows in* A *and* U*. The law rows therefore compare already displayed continuation-ledger operations; they do not introduce host category composition, quotient morphisms, external arrow equality, functorial semantics, natural transformations, or limit objects.*

Proof. Start with Definition 488.24. It fixes O, H, I, C, A, U, P, N as the full root admission surface and places hom, identity, and composition data in the displayed packet before the law transports are consumed.

Apply Definition 488.2. Identity rows are empty-continuation reads over unary object histories, so a unit-law read has no host identity operation to import.

Apply Definition 488.3. Composition rows are displayed Cont composites between continuation-ledger hom rows, so an associativity read must already have passed through H and C.

Finally use Definition 488.4. It supplies A and U as componentwise hsame law transports over those displayed rows, while Definition 488.10 removes every host or quotient category coordinate outside the packet. □

Theorem 488.29 (KernelCategory root endpoint exhaustion)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, every displayed* Cont *composite through the object row* O*, hom ledger* H*, identity row* I*, and composition row* C *has endpoints exhausted by the unary object histories in* O *and the continuation-ledger hom rows in* H*. No endpoint can be supplied by a host category object, quotient morphism, external arrow equality, functor semantic row, natural transformation, or limit object.*

Proof. Read the composite through Definition 488.3. It builds each composite from continuation-ledger hom rows in H and a displayed Cont composition row in C.

Use Definition 488.2 for endpoint identities. The identity rows in I are empty-continuation reads over unary object histories, so their endpoints are already in O.

Apply Definition 488.9. Locality keeps the displayed composite inside the same endpoint and hom ledgers rather than moving through a host category composition operation.

Finish with Definition 488.4 and Definition 488.10. Their non-escape clauses exclude host endpoints, quotient endpoints, external arrow equality, functor semantics, natural transformations, and limits from the root endpoint surface. □

Theorem 488.30 (KernelCategory root non-escape surface)**.** *The root public surface of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$ *is exhausted by unary object histories, continuation-ledger hom rows, empty-continuation identities, displayed* Cont *composites, componentwise* hsame *law transports,* Pkg *provenance, and the local* NameCert *row. It exports no host category object, quotient morphism, external arrow equality, functor semantics, natural transformation, limit object, or detached global category certificate.*

Proof. Begin with Definition 488.24. The theorem identifies the displayed packet rows O, H, I, C, A, U, P, N and excludes additional category-shaped coordinates from carrier admission.

Use Definition 488.4. The certificate surface supplies exactly the local naming and law-transport reads needed by the root packet.

Use Definition 488.9. Any composite that reaches the public surface remains a displayed Cont composite over local endpoint and hom rows.

Finally apply Definition 488.10. The provenance row is local to the packet and cannot be reread as host category data, quotient morphism data, functorial semantics, natural transformations, limits, or a second global naming source. □

Theorem 488.31 (KernelCategory root public readiness)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, the public root surface is ready exactly when every root read is a displayed* BHist *read of unary object histories in* O*, continuation-ledger hom rows in* H*, empty-continuation identity rows in* I*, displayed* Cont *composites in* C*, componentwise* hsame *law transports in* A *and* U*,* Pkg *provenance in* P*, and the local* NameCert *row in* N*. The surface contains no host category object, quotient morphism, external arrow equality, functorial semantics, natural transformation, limit object, or global naming source outside those rows.*

Proof. Project from Definition 488.1. The carrier exposes only the rows O, H, I, C, A, U, P, N, with object reads carried by unary BHist histories and hom reads carried by continuation ledgers.

Apply Definition 488.2 and Definition 488.3. Identity readiness is the empty Cont route over a unary history, and composition readiness is the displayed Cont composite over hom ledger rows.

Use Definition 488.5 and Definition 488.6 for the law rows. These theorems place associativity and unit comparisons in the componentwise hsame transports A and U.

Finally apply Definition 488.4, Definition 488.10, and Definition 488.24. The package and naming rows are therefore local to P, N, and every public root read is already accounted for by the displayed packet. □

Theorem 488.32 (KernelCategory downstream consumer totality)**.** *Every downstream consumer of a* KernelCategory$^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$ *is total on the displayed BEDC rows of* K *precisely by reading unary* BHist *object histories, continuation-ledger hom histories, displayed* Cont *identity and composition routes, componentwise* hsame *law transports,* Pkg *provenance, and the local* NameCert *row. No downstream consumer may demand a host category carrier, quotient morphism, external equality of arrows, natural-transformation coordinate, limit object, or second certificate before it can read the packet.*

Proof. Start with Definition 488.24. It gives the finite admission surface on which a consumer may be total: O, H, I, C, A, U, P, N and no additional category-shaped row.

Read the operation rows next. By Definition 488.2, identity consumption is total for empty continuation rows over unary object histories. By Definition 488.3, composition consumption is total for displayed Cont composites over continuation-ledger hom rows.

The law rows are then supplied by Definition 488.5, Definition 488.6, and Definition 488.13. They make the consumer read only the componentwise hsame transports A and U.

Finally use Definition 488.4 and Definition 488.10. Since Pkg provenance and the local NameCert row are the only certificate endpoints, the downstream route is total on the displayed packet and nowhere else. □

Theorem 488.33 (KernelCategory formal-target frontier)**.** *The formal target frontier for* KernelCategory$^{\uparrow}$ *is the concrete packet* $K = (O, H, I, C, A, U, P, N)$ *together with its BEDC rows:* BHist *unary object histories, continuation-ledger hom histories, displayed* Cont *identity and composition routes, componentwise* hsame *associativity and unit transports,* Pkg *provenance, and local* NameCert *naming. Formalization of the public root claim needs exactly carrier admission, identity continuation, composition continuation, law transport, provenance non-escape, and local certificate exhaustion for those rows, and it needs no host category semantics or quotient arrow equality.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryUp StdBridge`

Proof. The frontier begins at Definition 488.1: the only data offered to Lean-side reconstruction is the packet O, H, I, C, A, U, P, N.

Use Definition 488.24 to identify the admission clauses. This separates unary object histories, hom ledgers, identity rows, composition rows, law transports, provenance, and local naming as the proof obligations.

Then compose the existing operation and law theorems. The identity and composition clauses are Definition 488.2 and Definition 488.3; the law clauses are Definition 488.5, Definition 488.6, and Definition 488.13.

The frontier closes with Definition 488.4 and Definition 488.10. These show that Pkg provenance and the local NameCert row are the final public coordinates, while every host or quotient category object remains outside the formal target. □

Theorem 488.34 (KernelCategory root transport non-escape)**.** *For a* KernelCategory$^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, every endpoint and hom transport remains inside the displayed rows* O, H, I, C, A, U, P, N*. Endpoint transport reads unary object histories in* O*, hom transport reads*

continuation-ledger hom rows in H, and law transport reads only the componentwise hsame *rows in A and U together with the displayed* Cont, Pkg, *and local* NameCert *support. No host category object, quotient morphism, external arrow equality, functorial semantic row, natural transformation, or limit object is available.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier root transport nonescape`

Proof. Start with the carrier admission obligation, Definition 488.24. It fixes the finite packet rows O, H, I, C, A, U, P, N as the full admission surface.

Apply Definition 488.2 to endpoint identities and Definition 488.3 to hom composites. These theorems keep identity and composition reads in unary object histories, continuation-ledger hom rows, and displayed Cont composites.

Finally use Definition 488.4 and Definition 488.10. The certificate rows supply the componentwise law transports, and the provenance theorem removes host category objects, quotient morphisms, external equality, and all category-shaped structure outside the packet. □

Theorem 488.35 (KernelCategory public consumer triad)**.** *Every public consumer of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet must pass through three admission gates before reading associativity or unit law rows:*

1. *carrier admission for the object, hom, identity, composition, law-transport, provenance, and local naming rows;*
2. *identity-continuation admission for empty-continuation identity rows over unary object histories;*
3. *composition-continuation admission for displayed* Cont *composites between continuation-ledger hom rows.*

Only after these three gates may the consumer read the A and U law-transport rows. The route imports no host category object, quotient morphism, external arrow equality, functorial semantics, natural transformation, limit object, or second naming source.

Proof. First use Definition 488.24. It exposes the finite carrier surface and blocks public reads outside O, H, I, C, A, U, P, N.

Next apply Definition 488.2. The identity side of a public consumer is admitted only through empty Cont rows attached to unary object histories.

Then apply Definition 488.3. The composition side is admitted only through displayed Cont composites built from continuation-ledger hom rows.

With the three gates in place, Definition 488.4 supplies the associativity and unit transports, and Definition 488.10 supplies the provenance non-escape clause. Thus the law rows are consumed only after carrier, identity, and composition admission, with no host or quotient route added. □

Theorem 488.36 (KernelCategory hom ledger exhaustion)**.** *For a* $\mathsf{KernelCategory}^{\uparrow}$ *packet $K = (O, H, I, C, A, U, P, N)$, every hom read is exhausted by unary endpoints in O, continuation-ledger hom rows in H, displayed* Cont *composites in C, componentwise* hsame *transports in A and U,* Pkg *provenance in P, and the local* NameCert *row N. A hom consumer has no extra coordinate for host morphisms, quotient morphisms, external equality of arrows, functorial semantics, natural transformations, limits, or a detached global category.*

Proof. Begin with Definition 488.24. It identifies the hom ledger as part of the same finite packet as the unary object endpoints, law transports, provenance, and local naming row.

Use Definition 488.3. Every composite hom read is a displayed Cont composite of continuation-ledger hom rows whose endpoints are already in O and H.

Use Definition 488.2 for identity hom rows. These are empty-continuation reads over unary object histories, so they add no host identity morphism.

Finally apply Definition 488.4 and Definition 488.10. The hsame, Pkg, and local NameCert rows exhaust the public certificate surface and exclude every host, quotient, functorial, limit, or detached category coordinate. □

Theorem 488.37 (KernelCategory continuation composition stability)**.** *Let $K = (O, H, I, C, A, U, P, N)$ and $K' = (O', H', I', C', A', U', P', N')$ be* $\mathsf{KernelCategory}^{\uparrow}$ *packets whose endpoints and hom rows are related by componentwise* hsame *transports. Any displayed* Cont *composite admitted by C transports to the corresponding displayed* Cont *composite admitted by C' using only the endpoint rows, hom rows, and law transports of the two packets. The stability statement has no coordinate for host category composition, quotient morphism, or external equality of arrows.*

Proof. Read composition through Definition 488.9. It says that a composition read is local to displayed Cont rows and the endpoint ledgers already present in O and H.

Apply Definition 488.3. The composite hom row is constructed from the two input continuation-ledger hom rows and the displayed Cont composite row joining them.

The componentwise hsame endpoint and hom transports move those input rows from K to K'. The associativity and unit transports in A, U, A', U' compare only displayed continuation ledgers, so the transported composite remains a displayed Cont composite rather than a host category composition.

Finally use Definition 488.4. Its non-escape clause removes quotient morphisms and external arrow equality, leaving the displayed continuation route as the entire stability witness. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier cont composition stability`

Theorem 488.38 (KernelCategory NameCert route exhaustion)**.** *Every public read of a* $\mathsf{KernelCategory}^{\uparrow}$ *packet factors through carrier admission, identity continuation, composition continuation, associativity and unit transports, package provenance, and local naming. Thus the local* NameCert *route is exhausted by the rows O, H, I, C, A, U, P, N and exports no host category universe, quotient morphism, external arrow equality, functorial semantics, natural transformation, limit object, adjunction, or second naming source.*

Proof. First apply Definition 488.24. It fixes the available carrier rows as unary object histories, continuation-ledger hom rows, identity rows, displayed composites, law transports, provenance, and local naming.

Next use Definition 488.2 and Definition 488.3. These identify the two operation routes available to a public read and keep both routes inside the displayed continuation ledger.

The law rows are supplied by Definition 488.4. They transport associativity and unit comparisons through hsame rows rather than through external equality of arrows.

Finally apply Definition 488.10. The provenance row records only the finite BEDC carrier and local NameCert route, so all host category, quotient, functorial, limit, adjunction, and second naming coordinates are outside the public read. □

Theorem 488.39 (KernelCategory scoped endpoint transport surface)**.** *Let $K = (O, H, I, C, A, U, P, N)$ be an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet. Every endpoint transport available to a kernel-category consumer is exhausted by the unary object histories in O, the continuation-ledger hom histories in H, the empty-continuation identity rows in I, the displayed* Cont *composition rows in C, the componentwise* hsame *associativity rows in A, and the componentwise* hsame *unit rows in U. Transport*

can therefore move object endpoints, hom endpoints, identity rows, composite rows, and law rows only by the componentwise hsame *data already listed in* K.

Proof. Start with carrier admission. By Definition 488.1 and Definition 488.24, the only endpoint-bearing rows of K are the unary object rows, continuation-ledger hom rows, identity rows, displayed composites, and law-transport rows.

Use the operation rows next. Definition 488.2 identifies identities with empty Cont rows over the transported unary histories, while Definition 488.3 and Definition 488.37 keep composite endpoints inside the displayed hom and composition ledgers.

It remains to account for law transport. Definition 488.5, Definition 488.6, and Definition 488.13 place every associativity or unit comparison in the componentwise hsame rows A and U. The non-escape surface of Definition 488.30 then removes host endpoints, quotient endpoints, and external arrow equality from the scoped endpoint surface. □

Theorem 488.40 (KernelCategory scoped package provenance surface)**.** *Let* $K = (O, H, I, C, A, U, P, N)$ *be an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet. The package provenance exposed by* P *is exhausted by* BHist *rows, UnaryHistory object histories, displayed* Cont *routes, componentwise* hsame *transports, the* Pkg *row, the local* NameCert *row, the concrete* $\mathsf{Cat}^{\uparrow}$ *surface, and the sibling ContinuationMonad and UnaryContMonoid surfaces used to read identity, composition, associativity, and unit routes. No provenance read may pass to a second package source or to a host category carrier outside those displayed rows.*

Proof. Project the packet through the certificate surface. Definition 488.4 and Definition 488.8 list exactly the carrier, identity, composition, law, package, and local naming obligations available to P.

Now bind the category-shaped rows to BEDC primitives. Definition 488.7, Definition 488.31, and Definition 488.32 show that downstream CategoryUp, ContinuationMonad, and UnaryContMonoid consumers read only unary histories, continuation-ledger homs, displayed Cont routes, and componentwise hsame transports.

Finally apply Definition 488.10 and Definition 488.38. The provenance and naming rows are local to P, N, so no second package source, host category carrier, quotient morphism, or detached global certificate can be appended to the scoped package surface. □

Theorem 488.41 (KernelCategory scoped non-escape surface)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, the scoped obligation surface exports no host category object, quotient morphism, external arrow equality, functor semantics, natural transformation, limit object, adjunction, or detached category certificate. All category-shaped laws are read as BEDC rows: unary object histories, continuation-ledger hom histories, empty-continuation identities, displayed* Cont *composites, componentwise* hsame *law transports,* Pkg *provenance, and local* NameCert *naming.*

Proof. First combine the operation and law rows. Definition 488.2, Definition 488.3, Definition 488.5, and Definition 488.6 account for the category-shaped identity, composition, associativity, and unit readings without leaving the displayed packet.

Next use the public route theorems. Definition 488.28, Definition 488.29, and Definition 488.32 show that every downstream consumer receives the same rows O, H, I, C, A, U, P, N and no unlisted semantic coordinate.

The remaining exclusions are exactly the provenance and readiness exclusions. Definition 488.10, Definition 488.30, and Definition 488.33 remove host category objects, quotient morphisms, ex-

ternal equality of arrows, functor semantics, natural transformations, limits, adjunctions, and detached certificates from the scoped surface. □

Theorem 488.42 (KernelCategory downstream functor route boundary)**.** *Let $K = (O, H, I, C, A, U, P, N)$ be an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet. Every* $\mathsf{Functor}^{\uparrow}$*-facing read of a KernelCategory route factors through the displayed rows O, H, I, C, A, U, P, N: unary object histories, continuation-ledger hom rows, empty-continuation identities, displayed* Cont *composites, componentwise* hsame *law transports, package provenance, and local naming. It cannot import host arrow equality, quotient morphisms, host composition, functor semantics, natural transformations, or limit data.*

Proof. Start with the carrier and certificate surface. The carrier Definition 488.1 fixes O, H, I, C, A, U, P, N as the packet, while Definition 488.4 exposes only the local certificate rows over that packet.

Use the operation routes. The identity theorem Definition 488.2 and composition theorem Definition 488.3 place the functor-facing source and target reads inside empty-continuation identities and displayed Cont composites.

Finally apply route totality and downstream readiness. By Definition 488.14 and Definition 488.16, the downstream functor handoff receives exactly the displayed KernelCategory NameCert route and has no host arrow equality, quotient morphism, host composition, semantic functor row, natural transformation, or limit coordinate to read. □

Theorem 488.43 (KernelCategory ContinuationMonad consumer exhaustion)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet $K = (O, H, I, C, A, U, P, N)$, every* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumer exhausts its unit and bind reads through the empty* Cont *identity rows in I and the displayed* Cont *composition rows in C. The consumer may also read the endpoint, hom-ledger, law-transport, provenance, and local naming rows O, H, A, U, P, N, but it cannot add a host monad, host category composition, quotient morphism, external arrow equality, or detached continuation semantics.*

Proof. Read the finite packet first. The carrier Definition 488.1 supplies the only object, hom, operation, law, provenance, and naming rows available to a continuation consumer, and Definition 488.4 keeps the certificate local to those rows.

Apply the identity and composition routes. By Definition 488.2, unit reads are empty Cont identity rows in I. By Definition 488.3, bind-shaped reads are displayed Cont composites in C over the hom ledger H.

Close with totality and readiness. The route totality theorem Definition 488.14 and downstream readiness theorem Definition 488.16 exhaust every continuation-monad consumer on O, H, I, C, A, U, P, N, excluding host monads, host composition, quotient morphisms, external equality, and detached continuation semantics. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier continuation monad consumer exhaustion`

Theorem 488.44 (KernelCategory CategoryUp public export lock)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet $K = (O, H, I, C, A, U, P, N)$, the public export to* $\mathsf{Cat}^{\uparrow}$ *is exactly the carrier, identity, composition, law transport, package, and local* NameCert *rows displayed by K. No public export adds a host category carrier, quotient arrow class, global equality of morphisms, functor semantic row, natural transformation, limit object, or detached public certificate.*

Proof. Begin with carrier and certificate. The carrier Definition 488.1 lists the finite export rows, and Definition 488.4 makes package provenance and local naming part of the same public surface.

Attach identity, composition, and law transport. The identity theorem Definition 488.2 supplies the empty-continuation identity rows, while Definition 488.3 supplies displayed composition rows.

The law transport rows are the same componentwise hsame rows accounted for by the certificate surface.

Finally apply route totality and downstream readiness. Theorems Definition 488.14 and Definition 488.16 show that every public $\mathsf{Cat}^{\uparrow}$ export is already exhausted by O, H, I, C, A, U, P, N, leaving no host category carrier, quotient arrow class, global morphism equality, semantic functor row, natural transformation, limit object, or detached public certificate. □

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier category public export lock`

Theorem 488.45 (KernelCategory root continuation dependency)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *route* $K = (O, H, I, C, A, U, P, N)$*, every* $\mathsf{Cat}^{\uparrow}$ *hom read must first expose the UnaryHistory endpoint row* O*, the displayed* Cont *identity rows* I*, the displayed* Cont *composition rows* C*, and the* $\mathsf{ContinuationMonad}^{\uparrow}$ *source rows used by those identity and composition reads. No hom read may enter through a host category carrier, quotient morphism, or external arrow equality.*

Proof. Begin with the finite carrier. The carrier definition and Definition 488.8 require carrier admission, identity-continuation admission, and composition-continuation admission before any category-shaped consumer can read the packet.

The identity and composition obligations then expose the continuation surface. By Definition 488.2 and Definition 488.3, identities are empty Cont rows over unary endpoints and composites are displayed Cont rows over the hom ledger.

Finally use Definition 488.14. Its finite route lists exactly O, H, I, C, A, U, P, N, so the CategoryUp hom read has no host category, quotient, or external arrow-equality coordinate to consult before the continuation rows are visible. □

Theorem 488.46 (KernelCategory root law dependency)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *route* $K = (O, H, I, C, A, U, P, N)$*, every associativity or unit read is exhausted by the displayed componentwise* hsame *transport rows* A *and* U *after the identity and composition rows have been read. Host associativity, host unit equality, quotient morphism equality, and external arrow equality are not law sources for* K*.*

Proof. First read the operation rows. Definition 488.45 places endpoints, identity rows, composition rows, and the ContinuationMonad source surface before any law read.

Now apply the local law results. Definition 488.5, Definition 488.6, and Definition 488.13 identify the available associativity and unit comparisons with the componentwise hsame rows A and U.

The non-escape boundary closes the claim. By Definition 488.41, no host category law, quotient equality, or external arrow equality can be added to those displayed transport rows. □

Theorem 488.47 (KernelCategory root downstream unblock)**.** *Every* $\mathsf{Cat}^{\uparrow}$*,* $\mathsf{Functor}^{\uparrow}$*, and* $\mathsf{ContinuationMonad}^{\uparrow}$ *consumer of an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *route receives exactly the finite NameCert route* O, H, I, C, A, U, P, N*. The route is unblocked by the root obligation triad, the continuation dependency, the law dependency, route totality, and downstream readiness, and it has no detached consumer-specific carrier.*

Proof. Start with the root certificate obligations. Definition 488.8, Definition 488.45, and Definition 488.46 expose the carrier, continuation operation rows, and law-transport rows needed by all three consumers.

Next use the total route statement. Definition 488.14 exhausts every accepted route on the finite rows O, H, I, C, A, U, P, N.

Finally apply downstream readiness. Definition 488.16, Definition 488.42, and Definition 488.43 give the CategoryUp, FunctorUp, and ContinuationMonadUp handoffs from the same finite NameCert route, without a detached carrier for any one consumer. □

Theorem 488.48 (KernelCategory object-hom-law separation)**.** *For an accepted* $\mathsf{KernelCategory}^{\uparrow}$ *packet* $K = (O, H, I, C, A, U, P, N)$*, object rows, hom rows, and law rows are consumed through distinct finite coordinates:* O *for unary object endpoints,* H, I, C *for hom, identity, and composition continuation rows, and* A, U *for associativity and unit transport. No accepted consumer may collapse these coordinates into quotient morphisms, host arrow equality, or a single ambient category law.*

Lean4 ✓ `BEDC.Derived.KernelCategoryUp.KernelCategoryCarrier object hom law separation`

Proof. Project the carrier of Definition 488.1. It displays the object endpoint row O, the hom ledger H, the identity and composition continuation rows I, C, and the law transport rows A, U as separate coordinates.

Apply the root dependency theorems. The continuation dependency Definition 488.45 makes hom reads pass through O, H, I, C, while the law dependency Definition 488.46 makes associativity and unit reads pass through A, U after the operation rows are visible.

Finally use Definition 488.47. The downstream route exports the same finite packet O, H, I, C, A, U, P, N to category-shaped consumers, so quotient morphisms, host arrow equality, and ambient category laws have no coordinate through which to identify objects, homs, and laws. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$, maturePackageClosed)

Constructive story: A $\mathsf{KernelCategory}^{\uparrow}$ packet is generated from the finite carrier rows O, H, I, C, A, U, P, N: unary object histories, continuation-ledger hom histories, empty-continuation identities, displayed Cont composites, componentwise hsame associativity and unit transports, package provenance, and local NameCert naming.

Scope closed: The closed scope is the finite KernelCategory carrier with identity, composition, law transport, package provenance, CategoryUp, FunctorUp, ContinuationMonadUp, and UnaryContMonoid rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KernelCategory}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.KernelCategoryUp.KernelCategoryCertificateSurface`

Bridge status: bridgeChecked

Not claimed: The chapter does not close host category objects, quotient morphisms, external arrow equality, functor semantics, natural transformations, limits, adjunctions, or detached category certificates.

Upgrade path: No further upgrade is recorded on the formal axis; the chapter is bridge checked.

Chapter 489

A Concrete Naming Certificate for $\mathsf{ClassifierTyping}^{\uparrow}$

The $\mathsf{ClassifierTyping}^{\uparrow}$ surface realizes classifier membership as a finite typing packet. It records the visible term row, the Ext membership row, the Cont reduction row, the SigRel typing answer, and a local NameCert ledger, so a typing consumer reads only the rows displayed by the certificate.

$\mathsf{ClassifierTyping}^{\uparrow}$ is the visible classifier row consumed by chapter 476 and chapter 598. The dependency is the finite Ext/Cont/SigRel answer route, not a host type checker or hidden derivation object.

Definition 489.1 (ClassifierTyping carrier)**.** *A* $\mathsf{ClassifierTyping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, E, R, S, H, C, P, N)$$

with the following displayed coordinates:

1. *T is the term row whose finite syntax window is being checked;*
2. *E is the* Ext *membership row witnessing admission of T by the local classifier;*
3. *R is the* Cont *reduction row whose source and target term windows are visible in the packet;*
4. *S is the* SigRel *typing-answer row pairing the displayed classifier question with its accepted answer ledger;*
5. *H stores the componentwise* hsame *transports for the term, membership, reduction, signature-answer, package, and local ledger rows;*
6. *C stores the displayed* Cont *routes for reading membership, reduction, subject reduction, and signature-answer data;*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, SigRel, hsame, Pkg, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *ledger.*

The carrier contains no host typing semantics, no erased derivation object, no semantic equality of terms, no normalization oracle, and no classifier beyond the displayed Ext, Cont, SigRel, hsame, Pkg, *and* NameCert *rows.*

Definition 489.2 (ClassifierTyping classifier)**.** *For accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carriers* $X = (T, E, R, S, H, C, P, N)$ *and* $X' = (T', E', R', S', H', C', P', N')$*, the classifier compares exactly the visible* Ext *membership rows, the* SigRel *typing-answer rows, and the* Cont *reduction rows under the componentwise* hsame

transports recorded in H and H'. The term rows, package rows, and local ledgers may be inspected only as the coordinates that name those visible comparisons.

Thus two accepted packets classify together precisely when the exposed membership answer, typing signature answer, and reduction route remain the same readable BEDC data under hsame*. No host typing derivation, semantic conversion relation, erased representative, or ambient normalization proof participates in the classifier.*

Theorem 489.3 (ClassifierTyping NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *surface for Definition 489.1 consists of five obligations:*

1. *carrier inhabitance by a finite packet $X = (T, E, R, S, H, C, P, N)$;*
2. *classifier equivalence for the visible* Ext*,* SigRel*, and* Cont *rows under componentwise* hsame *transport;*
3. *membership stability, saying that a transported accepted term row preserves the displayed* Ext *membership answer;*
4. *subject-reduction-as-*Cont *stability, saying that a visible typed reduction route reads its target membership through R, C, and the same local ledger;*
5. *signature-gap readback, saying that any readable typing answer factors through the* SigRel *row S and the local* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *ledger.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier namecert obligations`

Proof. Carrier inhabitance is projection from Definition 489.1. The packet displays the term, membership, reduction, signature-answer, transport, continuation, package, and ledger rows needed by the local certificate.

Classifier equivalence is the comparison of Definition 489.2. Since the comparison may inspect only E, S, and R through H and C, stability of membership and reduction is a statement about visible rows rather than a host typing semantics.

For subject reduction, use the reduction coordinate R as the continuation route from the source window to the target window. The target membership read is licensed by the same Ext row and local ledger after transport, so it remains inside the packet.

For signature-gap readback, project the SigRel row S. A consumer that asks for the typing answer receives the answer ledger named by S and authenticated by N, with P recording the dependency package. No hidden conversion or normalization datum is available. □

Theorem 489.4 (ClassifierTyping membership stability)**.** *Let $X = (T, E, R, S, H, C, P, N)$ and $X' = (T', E', R', S', H', C', P', N')$ be accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carriers that classify together. Every public membership read of X' is obtained by transporting the displayed* Ext *membership row of X through the recorded* hsame *comparison and the* Cont *routes in C and C'. The read cannot introduce a new classifier, a host typing derivation, or a semantic conversion rule.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier membership stability`

Proof. Start with the classifier comparison. By Definition 489.2, a classification of X with X' includes the visible comparison of E with E' and does not expose any other membership source.

Transport the membership row along the displayed hsame coordinate. The continuation routes in C and C' license the consumer reads of the source and target packets, so the transported row is still a finite Ext membership answer.

The carrier inventory excludes any host typing derivation or conversion rule. Hence every membership read of X' factors through E, E', H, H', C, and C', and no external classifier can enter the surface. □

Theorem 489.5 (ClassifierTyping signature-gap readback)**.** *Every readable typing answer in an accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carrier factors through the displayed* SigRel *row S, the visible* Ext *membership row E, the reduction route R, and the local* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *ledger N. A consumer therefore receives a finite signature-gap answer packet, not a host type checker, not an erased derivation, and not an unrecorded normalization theorem.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier signature gap readback`

Proof. Project the accepted carrier. The typing-answer coordinate is S, and the rows that can justify its use are the membership row E, the reduction row R, and the local ledger N.

Readback follows the displayed continuation routes in C. These routes determine how a consumer reaches E, R, and S, while H records the permitted transports and P records the dependency package.

Since the packet has no coordinate for host type checking, erased derivations, or normalization, the consumer can read only the finite signature-gap packet named by S and authenticated by N. □

Theorem 489.6 (ClassifierTyping subject-reduction Cont stability)**.** *Let $X = (T, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carrier. Every subject-reduction read of X is a* Cont*-stable read: the source membership is read through the visible* Ext *row E, the reduction step is read through R, the typing answer is read through* SigRel *row S, and the target membership is reached only by replaying the displayed* hsame*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *rows H, P, N along the continuation routes in C.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier subject reduction cont stability`

Proof. Start with the carrier inventory of Definition 489.1. The packet displays the term row T, the membership row E, the reduction row R, the signature-answer row S, the transport row H, the route row C, the package row P, and the local naming row N.

Apply Definition 489.2. A reduction-preservation read can compare only the exposed Ext, Cont, and SigRel rows under the recorded hsame transports, so the source membership and typing answer cannot be replaced by a host derivation.

Use Definition 489.4 for the membership side. It routes transported membership through E, H, C and excludes a fresh classifier or semantic conversion rule.

Use Definition 489.5 for the answer side. It forces the typing answer through S, E, R, N, with P recording the finite provenance. Hence the subject-reduction read is exactly the displayed Cont route from source data to target membership inside X. □

Theorem 489.7 (ClassifierTyping visible-answer determinacy)**.** *Let $X = (T, E, R, S, H, C, P, N)$ and $X' = (T', E', R', S', H', C', P', N')$ be accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carriers that classify together. If a public consumer reads a typing answer from both packets, then the two answers are the same displayed* SigRel *answer under the recorded* hsame *transports and* Cont *routes. No second visible answer can be produced from* Ext*,* Cont*,* SigRel*,* hsame*,* Pkg*, or* NameCert *rows outside the displayed packets.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier visible answer determinacy`

Proof. Begin with the classifier of Definition 489.2. Classification of X with X' compares the visible membership rows, signature-answer rows, and reduction rows, and the comparison is mediated only by H, H', C, C'.

Apply Definition 489.5 to each packet. Each readable answer factors through S or S' together with the corresponding membership row, reduction route, and local ledger.

Apply Definition 489.4. The membership rows used by the answer reads are transported by the same displayed hsame and Cont data that the classifier already exposes.

The package and naming rows P, N, P', N' record the provenance of those finite reads. Since neither carrier contains a host type checker, erased derivation, semantic conversion rule, or normalization theorem, the paired reads determine one visible SigRel answer and leave no route for a second answer. □

Theorem 489.8 (ClassifierTyping reduction-route ledger exhaustion)**.** *For every accepted* $\mathsf{ClassifierTyping}^{\uparrow}$ *carrier* $X = (T, E, R, S, H, C, P, N)$*, every reduction-preservation route exported by the packet is exhausted by the displayed rows* T, E, R, S, H, C, P, N*. In particular, a consumer that asks for subject reduction receives the visible* Ext *membership row,* Cont *reduction row,* SigRel *answer row, componentwise* hsame *transports,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$ *ledger, and receives no hidden typing derivation, semantic conversion, erased representative, or normalization oracle.*

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier reduction route ledger exhaustion`

Proof. Project the carrier from Definition 489.1. Its reduction-facing coordinates are precisely the term, membership, reduction, signature-answer, transport, route, package, and local ledger rows.

Use Definition 489.6. Subject-reduction reads are replayed through the visible Cont route, with source membership from E, reduction from R, and target membership authenticated by H, P, N.

Use Definition 489.7. Any answer read along that route is the displayed SigRel answer named by S, so the reduction route cannot branch to an extra answer source.

The exclusion clauses of the carrier and classifier rule out host typing semantics, erased derivations, semantic conversion, and normalization. Therefore the reduction-route ledger is exhausted by T, E, R, S, H, C, P, N. □

Theorem 489.9 (ClassifierTyping kernel scope binding)**.** *Every public* $\mathsf{ClassifierTyping}^{\uparrow}$ *typing read is scoped by the concrete rows*

$$T, E, R, S, H, C, P, N$$

and by the kernel primitives Ext*,* Cont*,* SigRel*,* hsame*,* Pkg*, and* NameCert*. The read may inspect the term row, membership row, reduction row, signature-answer row, componentwise transport, continuation routes, package provenance, and local naming ledger. It has no coordinate for a host type checker, erased derivation, normalization oracle, or semantic conversion relation.*

Proof. Project the carrier of Definition 489.1. The displayed packet lists exactly the rows T, E, R, S, H, C, P, N, with E naming the Ext membership row, R and C naming the Cont reduction and read routes, and S naming the SigRel answer row.

Apply Definition 489.2. Public comparison is restricted to the visible Ext, SigRel, and Cont rows under the recorded hsame transports, with Pkg and NameCert providing provenance and local naming.

The excluded objects are not packet coordinates. Therefore every public typing read is bound to the displayed kernel rows and cannot appeal to a host type checker, erased derivation, normalization oracle, or semantic conversion relation. □

Lean4 ✓ `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier kernel scope binding`

Theorem 489.10 (ClassifierTyping Ext-Cont-SigRel scope exhaustion)**.** *The membership stability of Definition 489.4, subject-reduction* Cont *stability of Definition 489.6, signature-gap readback of Definition 489.5, visible-answer determinacy of Definition 489.7, and reduction-route ledger exhaustion of Definition 489.8 together exhaust the scoped kernel surface for* $\mathsf{ClassifierTyping}^{\uparrow}$ *over* Ext*,* Cont*,* SigRel*,* hsame*,* Pkg*, and* NameCert*.*

Proof. Start with Definition 489.9. It confines every public typing read to T, E, R, S, H, C, P, N and to the named kernel primitives.

Membership stability accounts for all reads of the Ext row. Subject-reduction Cont stability accounts for reduction-preservation reads, while signature-gap readback accounts for the SigRel answer row.

Visible-answer determinacy prevents a second public answer route, and reduction-route ledger exhaustion prevents an extra preservation route. The five cited results therefore cover the whole scoped surface and leave no host type checker, erased derivation, normalization oracle, or semantic conversion relation to inspect. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierTyping}^{\uparrow}$ packet is generated from a visible term row, an Ext membership row, a Cont reduction row, a SigRel typing-answer row, componentwise hsame transports, displayed Cont read routes, Pkg provenance, and a local NameCert ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 489.1, Definition 489.2, Definition 489.3, Definition 489.4, Definition 489.5, Definition 489.6, Definition 489.7, Definition 489.8, Definition 489.9, and Definition 489.10 bind the carrier, classifier, NameCert obligations, membership stability, signature-gap readback, subject-reduction Cont stability, visible-answer determinacy, reduction-route ledger exhaustion, kernel scope binding, Ext-Cont-SigRel scope exhaustion, and bridge surface from the finite Ext/Cont/SigRel typing packet to the standard acceptance schema.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassifierTyping}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ClassifierTypingUp.ClassifierTypingCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host type checking, erased derivations, semantic conversion rules, Lean verification, or bridge checking.

Upgrade path: No further paper theory-axis step is recorded; formal bridge checking remains separate.

Chapter 490

A Concrete Naming Certificate for $\mathsf{HaltingObstruction}^{\uparrow}$

The $\mathsf{HaltingObstruction}^{\uparrow}$ surface realizes the halting fixed-point pressure of chapter 9368 as a finite naming certificate. It records only positive finite traces, self-reference rows, diagonal policy rows, and certificate provenance. It is not a global halting classifier and it carries no decision procedure for arbitrary continuation chains.

The $\mathsf{HaltingObstruction}^{\uparrow}$ packet supplies the finite obstruction ledger consumed by chapter 491 and the local fixed-point packet of chapter 496. This dependency is limited to finite trace, self-reference, diagonal-policy, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Definition 490.1 (Halting obstruction carrier)**.** *A* $\mathsf{HaltingObstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, I, T, R, D, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the certificate row naming the local program-or-certificate surface whose continuation behavior is being inspected;*
2. *I is the input-history row naming the starting history and its finite observation window;*
3. *T is the finite* Cont *trace row recording the positive continuation steps actually witnessed from I;*
4. *R is the self-reference row that identifies the history being fed back into the same certificate surface;*
5. *D is the diagonal policy row recording the local non-escape ledger attached to that self-application;*
6. *H stores the componentwise* hsame *transports for the certificate, input history, finite trace, self-reference row, diagonal policy row, package row, and local naming row;*
7. *C stores the displayed* Cont *routes by which a consumer reaches the trace, self-reference, and diagonal rows;*
8. *P is the* Pkg *provenance over* BHist, Cont, hsame, Pkg, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$ *row.*

The carrier has no coordinate for deciding whether an arbitrary chain halts, no total classifier over all certificates and histories, and no host-level oracle for divergence.

Theorem 490.2 (Halting obstruction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$ *surface for Definition 490.1 consists of five obligations:*

1. *carrier habitation by the finite packet* $O = (S, I, T, R, D, H, C, P, N)$*;*
2. *trace exactness, saying that every positive continuation fact exported by the packet is one of the finite* Cont *steps recorded in* T*;*
3. *self-reference readability, saying that the feedback row* R *is reached only through the displayed input-history and certificate rows* I *and* S*;*
4. *diagonal policy locality, saying that each obstruction claim is the ledger entry named by* D *after the displayed self-application route;*
5. *package and name closure, saying that all public reads pass through the stored* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Thus the packet certifies a finite obstruction ledger for a displayed self-reference route, not a halting decision for every certificate-history pair.

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionNamecertObligations`

Proof. Project the carrier of Definition 490.1. The rows S, I, and T give the only continuation facts that can be read: each exported step is one displayed finite Cont transition in the trace row.

Next read the self-reference coordinate R. It is not a classifier over all histories; it is a recorded feedback row tied to the same certificate and input-history surface by the transports in H and the routes in C.

The diagonal row D is then local to that feedback route. A consumer can read its obstruction ledger only after following the displayed Cont routes through T and R.

Finally, repack the visible rows with P and N. Since no carrier coordinate names a total halting predicate or a divergence oracle, the NameCert obligations expose only the finite trace and diagonal ledger listed above. □

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionCarrier namecert obligations`

Theorem 490.3 (Halting obstruction diagonal non-escape)**.** *Let* $O = (S, I, T, R, D, H, C, P, N)$ *be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *carrier. Every public attempt to turn the displayed self-reference route into a halting decision must factor through the finite trace row* T*, the self-reference row* R*, the diagonal policy row* D*, and the recorded* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. The exported surface therefore cannot escape into a global halting classifier, an unbounded search over continuation histories, a host divergence oracle, or quotient equality of certificate names.*

Proof. Start with the NameCert obligations of Definition 490.2. Trace exactness forces every positive continuation read to be one of the finite steps recorded in T.

Self-reference readability then routes the feedback claim through R together with the input-history and certificate rows. The consumer receives a displayed self-application route, not an external enumeration of all possible histories.

Apply diagonal policy locality. The obstruction ledger is exactly the row D attached to that route, transported only by H and read only through the continuations in C.

The exclusions follow by carrier inspection. Since O contains no total halting predicate, no unbounded search coordinate, no host divergence oracle, and no quotient equality of names, none of those objects can be exported from the public surface. □

Theorem 490.4 (Halting obstruction distinction boundary)**.** *Every accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *diagonal read consumes the HaltingDistinction certificate-history pair, finite trace, and diagonal self-reference boundary of chapter 491 before it can expose the obstruction ledger* D*. The obstruction surface is therefore a downstream finite ledger over* $S, I, T, R, D, H, C, P, N$*, not a global halting classifier and not a duplicate distinction principle.*

Proof. Project the obstruction carrier of Definition 490.1. Its diagonal read is available only after the certificate row S, input-history row I, finite trace row T, self-reference row R, and diagonal policy row D have been displayed.

Apply the local NameCert obligations of Definition 490.2. Trace exactness, self-reference readability, diagonal policy locality, and package closure force any public obstruction read through the finite rows T, R, D, H, C, P, N.

Now read the sibling boundary chapter 491. The certificate-history pair, finite trace, and diagonal self-reference packet supplied there are the distinction data consumed before the obstruction ledger is exposed.

Finally use diagonal non-escape from Definition 490.3. Since the obstruction packet has no total halting predicate, unbounded search, host divergence oracle, or quotient equality of certificate names, it remains a downstream finite ledger rather than a global classifier. □

Theorem 490.5 (Halting obstruction trace coverage)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every positive continuation fact exported by the packet is a displayed member of the finite trace row* T*. Such a fact is read through* BHist*,* Cont*,* hsame*,* Pkg*, and the local* NameCert *rows, and it is not supplied by an unbounded search over histories.*

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionTraceCoverage`

Proof. Project the carrier of Definition 490.1. The only row that records positive continuation facts is the finite trace row T.

Apply trace exactness from Definition 490.2. Any exported continuation fact must be one of the displayed Cont steps stored in T, transported by H and reached by the route row C.

The package and local naming rows P, N only repack that same finite evidence. Since the carrier has no coordinate for an unbounded continuation search, the public trace surface is exactly the displayed membership surface of T. □

Theorem 490.6 (Halting obstruction self-reference route determinacy)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, the self-reference row* R *is readable only through the certificate row* S*, input-history row* I*, finite trace row* T*, componentwise transports* H*, and displayed continuations* C*. The rows* P *and* N *certify that route, and no second route can read* R *without passing through those displayed coordinates.*

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionSelfReferenceRouteDeterminacy`

Proof. Start with the self-reference coordinate in Definition 490.1. It is a row of the same finite packet, not a free-standing self-application operator.

Use self-reference readability from Definition 490.2. A read of R must first name the certificate surface S and input-history row I, and it must consume the finite trace row T along the displayed routes.

The transports H and continuations C determine the only admissible path to the row. The package provenance P and local NameCert row N certify that path rather than adding an alternate route. □

Theorem 490.7 (Halting obstruction diagonal policy ledger exactness)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, the row* D *is the unique diagonal policy ledger row exported by the packet. It records the local obstruction attached to the displayed self-reference route and cannot supply a total halting classifier for arbitrary certificate-history pairs.*

Proof. Inspect the carrier in Definition 490.1. Among its coordinates, only D is a diagonal policy row; the remaining rows name the certificate, input history, finite trace, self-reference route, transports, continuations, provenance, and local naming surface.

Apply diagonal policy locality from Definition 490.2. A public diagonal read reaches D only after following the displayed self-reference route through S, I, T, R, H, C, P, N.

Finally use Definition 490.3. Since the packet has no coordinate for a total halting predicate, unbounded search, host divergence oracle, or quotient equality of names, the diagonal ledger remains local and cannot become a classifier for arbitrary inputs. □

Theorem 490.8 (Halting obstruction positive trace ledger)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every positive continuation fact exported by the public obstruction surface is one of the finite rows displayed in* T*. The exported fact is reached through the carrier, transport, route, package, and local naming coordinates of* O*, and it cannot be replaced by an unbounded search over continuation histories.*

Proof. Begin with the carrier of Definition 490.1. It names T as the finite Cont trace row and gives no separate coordinate for a search procedure over all histories.

Apply the trace exactness obligation in Definition 490.2. Any positive continuation fact exposed by the packet must already be one of the displayed finite Cont steps in T.

The non-escape statement of Definition 490.3 rules out reading this trace as a global halting classifier or as an unbounded continuation oracle. The distinction boundary of Definition 490.4 then identifies the downstream object as the same finite obstruction ledger. □

Theorem 490.9 (Halting obstruction finite route non-escape)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every public diagonal read factors only through the displayed rows* $S, I, T, R, D, H, C, P, N$*. In particular, no consumer can obtain a diagonal obstruction read by bypassing the finite trace, self-reference, policy, transport, route, package, and local naming rows of* O*.*

Proof. Inspect the coordinates in Definition 490.1. The public packet lists exactly the certificate row, input-history row, trace row, self-reference row, diagonal policy row, transports, continuation routes, package provenance, and local NameCert row.

Use package and name closure from Definition 490.2. Public reads must pass through the stored hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, while self-reference readability and diagonal policy locality force the route through R and D.

Apply Definition 490.3. Since the carrier contains no global classifier, unbounded search coordinate, host oracle, or quotient equality of names, the only public diagonal route is the finite route through $S, I, T, R, D, H, C, P, N$. □

Theorem 490.10 (Halting obstruction public read boundary)**.** *Every downstream HaltingDistinction or HaltingDiagonal consumer of an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet receives the finite obstruction ledger displayed by* $S, I, T, R, D, H, C, P, N$*. The consumer receives trace exactness, self-reference readability, diagonal policy locality, and package/name closure for that ledger, not a global classifier for arbitrary certificate-history pairs.*

Proof. Project the accepted carrier from Definition 490.1. Its public payload is the finite obstruction packet $S, I, T, R, D, H, C, P, N$.

Apply the five NameCert obligations of Definition 490.2. They provide the carrier row, the exact finite positive trace, the readable self-reference route, the local diagonal policy ledger, and the package/name closure surface.

Now use Definition 490.4. Downstream HaltingDistinction and HaltingDiagonal uses consume the finite ledger after the displayed certificate-history and self-reference boundary have been read, so the public object is not a global halting classifier. □

Theorem 490.11 (Halting obstruction finite counterread refusal)**.** *Let $O = (S, I, T, R, D, H, C, P, N)$ be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. Any read that presents the finite trace row T together with the diagonal policy row D as a total classifier for arbitrary continuation chains is refused by the carrier: the public packet exposes only $S, I, T, R, D, H, C, P, N$, and no global halting classifier, divergence oracle, unbounded search surface, host oracle, or quotient equality of names.*

Proof. Use trace coverage from Definition 490.5. The row T covers exactly the displayed finite positive trace facts admitted by the packet, not all possible continuation chains.

Next apply diagonal policy ledger exactness from Definition 490.7. The row D records the local diagonal policy attached to the same certificate-history and self-reference route, so it does not decide arbitrary continuations outside that route.

Finally use the public read boundary of Definition 490.10. A downstream read receives the finite obstruction ledger $S, I, T, R, D, H, C, P, N$ rather than a global classifier. Therefore a counterread that treats T and D as a total halting decision procedure is not a public read of the accepted packet. □

Theorem 490.12 (Halting obstruction local NameCert repackaging)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet $O = (S, I, T, R, D, H, C, P, N)$, the package row P and local naming row N repackage only the carrier exposure, trace exactness, self-reference readability, diagonal policy locality, and package/name closure already displayed by the packet. They add no new classifier, search surface, host oracle, quotient equality, or alternate diagonal route.*

Proof. Read P and N against the carrier in Definition 490.1. They are package provenance and local NameCert rows for the same finite obstruction packet, not independent computational operators.

Apply Definition 490.2. The content available for repackaging is exactly carrier habitation, trace exactness, self-reference readability, diagonal policy locality, and package/name closure.

Finally combine Definition 490.3 with Definition 490.4. The repackaged public surface remains the same finite obstruction ledger and therefore cannot expose a global classifier, unbounded search, host oracle, quotient equality, or second diagonal route. □

Theorem 490.13 (Halting obstruction distinction trace handoff)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet $O = (S, I, T, R, D, H, C, P, N)$, every handoff to* $\mathsf{HaltingDistinction}^{\uparrow}$ *carries only the finite distinction-trace rows displayed by T together with the self-reference, transport, route, package, and local naming rows R, H, C, P, N. The handoff exposes no global halting classifier, no unbounded continuation search, and no off-packet equality of names.*

Proof. Project the accepted carrier of Definition 490.1. The only trace coordinate available to the packet is the finite row T, and its movement is recorded through H and C.

Apply Definition 490.2. The local $\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$ row certifies carrier admission, trace exactness, self-reference readability, diagonal policy locality, and package/name closure for the same tuple $S, I, T, R, D, H, C, P, N$.

Use Definition 490.4 and Definition 490.10. The distinction consumer reads the displayed finite obstruction ledger, while the public boundary excludes a total classifier, host oracle, unbounded search, quotient equality, or alternate diagonal route.

Finally apply Definition 490.12. The package and local naming rows repackage the same trace and distinction rows, so the downstream handoff is exhausted by T, R, H, C, P, N inside the accepted packet. □

Theorem 490.14 (Halting obstruction diagonal policy refusal)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, any diagonal-policy read that bypasses one of the displayed rows* T, R, D, H, C, P, N *is refused by the public packet. The diagonal policy row* D *is readable only as the local obstruction ledger for the displayed trace and self-reference route.*

Proof. Start from the carrier in Definition 490.1. The row D is stored beside the finite trace row T, self-reference row R, transport row H, displayed route row C, provenance row P, and local naming row N.

Apply Definition 490.2. The local naming surface binds D to trace exactness, self-reference readability, diagonal policy locality, and package/name closure for the same tuple.

Use Definition 490.3 and Definition 490.10. A public diagonal read cannot be replaced by a global classifier, host divergence oracle, quotient equality, unbounded search, or a second route outside T, R, D, H, C, P, N.

Finally invoke Definition 490.12. Since P and N add no alternate classifier or diagonal route, a bypassing diagonal-policy read is not a read of the accepted obstruction packet. □

Theorem 490.15 (Halting obstruction public consumer ledger exhaustion)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every downstream public consumer is exhausted by the displayed finite ledger* $S, I, T, R, D, H, C, P, N$*. The consumer may read certificate-source, input-history, trace, self-reference, diagonal policy, transport, route, package, and local naming rows, but no additional halting, divergence, evaluator, equality, or classifier row.*

Proof. Read the packet from Definition 490.1. Its carrier inventory is the finite ledger $S, I, T, R, D, H, C, P, N$.

Apply Definition 490.2. The obligations attach the carrier, trace, self-reference, diagonal policy, package, and local name to that ledger and do not introduce a separate consumer coordinate.

Use Definition 490.4, Definition 490.3, and Definition 490.10. These rows force downstream reads through the displayed distinction and diagonal boundary while excluding global halting classifiers, host oracles, unbounded search, quotient equality, and alternate routes.

Finally apply Definition 490.12. Repackaging through P and N preserves exactly the same finite ledger, so every public consumer is exhausted by $S, I, T, R, D, H, C, P, N$. □

Theorem 490.16 (Halting obstruction root carrier admission)**.** *Every accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet exposes exactly*

$$O = (S, I, T, R, D, H, C, P, N)$$

as its finite root surface: S *is the certificate-source row,* I *is the input-history row,* T *is the finite positive-trace ledger,* R *is the self-reference route row,* D *is the diagonal-policy row,* H *is the componentwise* hsame *ledger,* C *is the displayed* Cont *route family,* P *is the* Pkg *provenance row, and* N *is the local* $\mathsf{NameCert}$ *row.*

No accepted packet exposes any additional BHist*,* Cont*,* hsame*,* Pkg*, or* NameCert *coordinate at the root boundary.*

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionRootCarrierAdmission`

Proof. Project the carrier from Definition 490.1. The displayed tuple supplies exactly the rows $S, I, T, R, D, H, C, P, N$.

Apply Definition 490.2. Its carrier, trace, self-reference, diagonal-policy, and package/name clauses are all obligations over the same tuple, so no second root row is introduced by the local naming surface.

Use Definition 490.12. Repackaging through P and N preserves the same finite obstruction surface rather than adding a classifier, search row, host oracle, quotient equality, or alternate diagonal route. □

Theorem 490.17 (Halting obstruction root transport scope)**.** *For an accepted* HaltingObstruction$^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every public self-reference read and every positive trace read is transported only through the componentwise* hsame *ledger* H *and replayed only through the displayed* Cont *routes* C*.*

The rows S, I, T, R, D *determine what is being read, while* H *and* C *determine the only visible movement of those reads across the root boundary.*

Proof. Use Definition 490.5 for trace reads. It forces each positive continuation fact to be one of the finite steps stored in T, transported by H, and reached by the route row C.

Use Definition 490.6 for self-reference reads. The row R is readable only after the certificate source S, input-history row I, finite trace row T, and displayed transport route have been consumed.

Apply Definition 490.3. Since the packet contains no global classifier, unbounded continuation search, host divergence oracle, or quotient equality, no public read can bypass H or replay outside C. □

Theorem 490.18 (Halting obstruction root ledger exactness)**.** *For an accepted* HaltingObstruction$^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, the rows* P *and* N *package exactly the finite obstruction ledger exposed by* S, I, T, R, D, H, C*. They do not export a global halting classifier, unbounded search, host oracle, quotient equality, or any second diagonal route.*

Proof. Read the finite ledger through Definition 490.16. The only rows available to the root surface are the certificate source, input history, trace ledger, self-reference route, diagonal policy, transport, continuation, package, and local naming rows.

Apply Definition 490.7. A public diagonal read reaches D only through the displayed self-reference and trace route, so the diagonal policy remains a local ledger row.

Finally use Definition 490.12 together with Definition 490.3. The package and naming rows repack the finite obstruction ledger already present, and the non-escape boundary excludes the classifier, search, host-oracle, quotient, and alternate-route exports. □

Theorem 490.19 (Halting obstruction fixed-tuple image and readback)**.** *For an accepted* HaltingObstruction$^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, the same displayed tuple supplies the image row and readback row required by the fixed finite* BHist *tuple route of Definition 9464.24. The image is the finite obstruction ledger exposed by* $S, I, T, R, D, H, C, P, N$*, and the readback is the projection back to the same carrier rows.*

Proof. Start with Definition 490.16. It admits exactly the certificate source, input history, trace, self-reference, diagonal-policy, transport, continuation, package, and local naming rows.

Apply Definition 490.2. The local NameCert obligations keep the same rows visible when the packet is read as an accepted obstruction surface.

Use Definition 490.17 and Definition 490.18. Transport and package readback both remain inside H, C, P, N, so the fixed-tuple route of Definition 9464.24 has no additional image or readback coordinate to supply. □

Theorem 490.20 (Halting obstruction component transport rows)**.** *For an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet* $O = (S, I, T, R, D, H, C, P, N)$*, every local NameCert row transports componentwise through the displayed transport, route, package, and naming rows* H, C, P, N*. In particular, carrier admission, trace coverage, self-reference readability, diagonal-policy locality, and public package reads are transported as rows of the same finite tuple.*

Proof. Read carrier admission from Definition 490.16 and the local NameCert clauses from Definition 490.2. They place every obligation on the coordinates of the single tuple $S, I, T, R, D, H, C, P, N$.

Use Definition 490.17. It restricts self-reference and positive trace movement to the componentwise hsame ledger H and displayed Cont routes C.

Finally apply Definition 490.18 and Definition 9464.24. The package and naming rows P, N identify the transported finite ledger with the same fixed tuple, rather than with a host classifier or alternate diagonal route. □

Theorem 490.21 (Halting obstruction bridge request boundary)**.** *A* $\mathsf{StandardBridge}$ *request for* $\mathsf{HaltingObstruction}^{\uparrow}$ *is licensed only after four finite surfaces have been displayed for the same tuple* $S, I, T, R, D, H, C, P, N$*: the tuple rows, the image and readback rows of Definition 490.19, the componentwise NameCert transport rows of Definition 490.20, and the finite obstruction non-escape boundary.*

Lean4 ✓ `BEDC.Derived.HaltingObstructionUp.HaltingObstructionTasteGate single carrier alignment`

Proof. The tuple rows are supplied by Definition 490.16 and the local NameCert obligations of Definition 490.2. Thus a bridge request starts with the displayed obstruction packet, not with a fresh carrier.

The image and readback rows are exactly Definition 490.19. Componentwise transport is exactly Definition 490.20.

Use Definition 490.17, Definition 490.18, and Definition 490.3. Together with Definition 9464.24, they leave the TasteGate boundary with a fixed finite packet and exclude a global halting classifier, unbounded search, host divergence oracle, quotient equality, or second diagonal route. □

Theorem 490.22 (Halting obstruction root consumer package)**.** *Let* $O = (S, I, T, R, D, H, C, P, N)$ *be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. Every root downstream consumer receives only the finite trace row* T*, self-reference row* R*, diagonal-policy row* D*, componentwise transport row* H*, displayed route row* C*, package row* P*, and local* $\mathsf{NameCert}$ *row* N *carried by* O*. The handoff supplies no global halting classifier, no unbounded continuation search, and no off-packet equality of names.*

Proof. Start with Definition 490.16. It admits the accepted packet as the fixed tuple $S, I, T, R, D, H, C, P, N$ and gives no root coordinate beyond those rows.

Use Definition 490.17. Any public trace or self-reference read is transported through H and replayed through C, so the downstream consumer cannot bypass the displayed finite route.

Finally apply Definition 490.18. The rows P and N package exactly the ledger already exposed by S, I, T, R, D, H, C, excluding a classifier, search surface, host oracle, quotient equality, or second diagonal route. □

Theorem 490.23 (Halting obstruction diagonal consumer handoff)**.** *Let $O = (S, I, T, R, D, H, C, P, N)$ be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. Any* $\mathsf{HaltingDiagonal}^{\uparrow}$ *consumer that reads the obstruction from O receives the displayed self-reference row R, the diagonal-policy row D, the finite trace row T, and their transport, route, package, and local NameCert rows H, C, P, N. The handoff is a local diagonal ledger for the displayed self-application route, not a decision surface for arbitrary histories.*

Proof. Read the fixed tuple through Definition 490.16. The diagonal-facing rows are already part of the accepted packet: T records the finite trace, R records self-reference, and D records the diagonal policy.

Apply Definition 490.17. It forces each self-reference and trace read through H and C, so the diagonal consumer receives the displayed route rather than an ambient self-application operator.

Use Definition 490.18 to package the same rows through P and N. Then Definition 490.4 identifies the consumer-facing object as the finite distinction and diagonal boundary, not a global halting classifier. □

Theorem 490.24 (Halting obstruction public downstream exactness)**.** *Let $O = (S, I, T, R, D, H, C, P, N)$ be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. Every downstream* $\mathsf{HaltingDistinction}^{\uparrow}$ *or* $\mathsf{HaltingDiagonal}^{\uparrow}$ *public read is exhausted by the finite trace, self-reference, diagonal-policy, transport, route, package, and local NameCert rows displayed in O. No downstream read obtains an additional trace row, diagonal policy row, package row, or local naming row outside T, R, D, H, C, P, N.*

Proof. First apply Definition 490.16. It fixes the public obstruction surface as the single tuple $S, I, T, R, D, H, C, P, N$.

Next use Definition 490.17. Public trace and self-reference movement is limited to the componentwise hsame ledger H and displayed Cont routes C.

Apply Definition 490.18. The package and local naming rows expose only the finite obstruction ledger already present in S, I, T, R, D, H, C.

Finally invoke Definition 490.4. The HaltingDistinction and HaltingDiagonal consumers receive the same finite boundary rows before any obstruction ledger is read, so the public downstream surface is exact and contains no additional classifier or off-packet row. □

Theorem 490.25 (Halting obstruction root trace admission)**.** *Let $O = (S, I, T, R, D, H, C, P, N)$ be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. The accepted root surface exposes the finite trace row T only through the carrier row S, input-history row I, componentwise transport row H, displayed route row C, package row P, and local* $\mathsf{NameCert}$ *row N. It supplies no root trace outside the displayed finite packet.*

Proof. Use Definition 490.16 to place the accepted obstruction on the single tuple $S, I, T, R, D, H, C, P, N$. The only trace coordinate in that tuple is T.

Apply Definition 490.5. It records the finite trace as the accepted trace surface and does not provide another trace carrier.

Finally use the NameCert obligations and Definition 490.10. The trace can be transported through H, replayed through C, packaged by P, and named by N, but no downstream root read receives a second trace row. □

Theorem 490.26 (Halting obstruction root policy exhaustion)**.** *Let $O = (S, I, T, R, D, H, C, P, N)$ be an accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet. The diagonal-policy row D exhausts the obstruction-policy surface of the root packet. It remains a local policy ledger for the displayed trace and self-reference route and cannot become a global halting classifier, host divergence oracle, or decision procedure for arbitrary histories.*

Proof. Read the root packet through Definition 490.16. The policy-facing coordinate is the displayed row D, paired with the finite trace and self-reference rows T and R.

Apply Definition 490.7. It exhausts the diagonal-policy surface by the local ledger row and gives no classifier row for arbitrary histories.

Use Definition 490.3 and Definition 490.10. They keep the diagonal policy inside the finite obstruction route, so the policy row cannot escape as a global halting classifier. □

Theorem 490.27 (Halting obstruction root consumer boundary)**.** *For every accepted* $\mathsf{HaltingObstruction}^{\uparrow}$ *packet $O = (S, I, T, R, D, H, C, P, N)$, each downstream* $\mathsf{HaltingDistinction}^{\uparrow}$ *or* $\mathsf{HaltingDiagonal}^{\uparrow}$ *consumer factors through $S, I, T, R, D, H, C, P, N$. The consumer may read the finite trace, self-reference, diagonal-policy, transport, route, package, and naming rows, but it receives no global halting classifier, unbounded continuation search, host divergence oracle, quotient equality, or off-packet name equality.*

Proof. Start with Definition 490.16. It fixes the root obstruction as the finite tuple $S, I, T, R, D, H, C, P, N$.

Apply Definition 490.25 and Definition 490.26. These bind the trace-facing and policy-facing reads to T and D inside the same tuple.

Next use Definition 490.17 and Definition 490.18. Transport, route replay, packaging, and local naming are exactly the rows H, C, P, N already carried by the packet.

Finally invoke Definition 490.24. The HaltingDistinction and HaltingDiagonal handoffs are exhausted by the displayed finite rows, so every downstream consumer factors through $S, I, T, R, D, H, C, P, N$ and obtains no global halting surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$.

Constructive story: A $\mathsf{HaltingObstruction}^{\uparrow}$ packet is generated from finite certificate rows, input-history rows, trace rows, self-reference rows, diagonal-policy rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 490.1, Definition 490.16, Definition 490.5, Definition 490.6, Definition 490.7, Definition 490.10, Definition 490.18, Definition 490.19, Definition 490.20, and Definition 490.21 bind carrier admission, trace exactness, self-reference readability, diagonal policy locality, package route, public read boundary, local NameCert rows, fixed-tuple image/readback rows, componentwise transport rows, and the bridge request boundary as the public obstruction surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HaltingObstruction}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.HaltingObstructionUp.HaltingObstructionRootCarrierAdmission`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a global halting classifier, unbounded continuation search, host divergence oracle, quotient equality, Lean verification, or bridge checking.

Upgrade path: The theory axis has no further paper-side grade; Lean verification and bridge checking remain independent.

Chapter 491

A Concrete Naming Certificate for $\mathsf{HaltingDistinction}^{\uparrow}$

The $\mathsf{HaltingDistinction}^{\uparrow}$ surface turns the halting-as-distinction interface of chapter 9368 into a finite naming packet. It does not decide halting. It records the visible certificate-history pair, the finite continuation trace being tested, the diagonal self-reference row, the transports, routes, provenance, and local naming rows through which the fixed-point obstruction can be read.

The finite trace row T is the local trace surface refined by chapter 492. $\mathsf{HaltingDistinction}^{\uparrow}$ consumes that sibling only as a finite continuation-trace row, not as a total halting predicate or host evaluator.

The diagonal pair row is also consumed together with the inscription-event record surface of chapter 519 and the trace-normal-form route of chapter 683. The finite route Q, T, Z, H, C, P, N supplies the certificate-history pair, trace segment, diagonal read, transports, routes, provenance, and local ledger; the sibling chapters supply the inscription record and normal-form trace surfaces needed by later halting-boundary and cellular closed-substrate consumers.

Definition 491.1 (HaltingDistinction carrier)**.** *A* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, T, Z, H, C, P, N)$$

with the following displayed rows:

1. *Q is the certificate-history pair row, naming a candidate program certificate* $\mathsf{NameCert}_P$ *together with the history h whose* Cont_P *chain is being inspected;*
2. *T is the finite trace row recording the visible* Cont_P *chain segment from h through its displayed marks and intermediate histories;*
3. *Z is the diagonal self-reference row, naming the self-application read in which the classifier for Q is asked about the same certificate-history pair it is currently classifying;*
4. *H records the componentwise* hsame *transports for certificate-history pair rows, trace rows, diagonal rows, package rows, and local naming rows;*
5. *C records the displayed* Cont *routes for reading the input pair, replaying the finite trace, entering the diagonal row, and returning the result to the local ledger;*
6. *P is the* Pkg *provenance over* BHist, Ext, Cont, hsame, Pkg, *the diagonal read, and the surrounding* $\mathsf{NameCert}$ *rows;*

7. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *ledger.*

The carrier contains no global truth predicate, no classical decidability assumption for termination, no oracle for divergence, and no meta-level evaluator outside the displayed BHist, Ext, Cont, hsame, Pkg, *and local* NameCert *rows.*

Theorem 491.2 (HaltingDistinction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *surface for Definition 491.1 consists of five obligations:*

1. *carrier admission for the finite packet $D = (Q, T, Z, H, C, P, N)$;*
2. *certificate-history classifier transport, saying that pair rows classify only through the displayed* hsame *transport recorded in H;*
3. *trace readability, saying that a public termination-facing read factors through the finite* Cont_P *chain segment T and the read routes in C;*
4. *diagonal boundary exactness, saying that the self-reference row Z records the classifier's self-application point without supplying an external halting decision;*
5. *non-escape from global truth predicates, classical decidability, divergence oracles, host evaluators, or classifiers not named by Q, T, Z, H, C, P, and N.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionNameCertObligations`

Proof. Carrier admission is projection from Definition 491.1. The packet names exactly the pair, trace, diagonal, transport, continuation, package, and local naming rows.

Classifier transport is componentwise in H. Since Q is the only row naming the certificate-history input, any transported public comparison must return to the same visible pair data and its local ledger.

Trace readability is controlled by T and C. A consumer may replay the finite Cont_P segment displayed by T, but the carrier has no coordinate that upgrades this finite trace into a total halting predicate.

The diagonal boundary is the row Z. It records the self-application read needed by the fixed-point obstruction, while P and N record only provenance and local naming. Thus the packet exhibits the obstruction surface without importing a global decision principle. □

Theorem 491.3 (HaltingDistinction diagonal boundary)**.** *Let $D = (Q, T, Z, H, C, P, N)$ be an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier. Any public read that reaches the diagonal self-reference point reaches it only through the displayed route*

$$Q \longmapsto T \longmapsto Z \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The row Z may identify the classifier question with its own certificate-history input under the recorded hsame *transports, but it does not produce a truth value for all certificate-history pairs and does not decide divergence outside the displayed finite packet.*

Proof. Start from a public diagonal read. Carrier projection gives the input pair row Q, the finite trace row T, and the self-reference row Z as the only rows that can name the read.

Use the routes in C to move from Q through T to Z. The permitted renamings are exactly the componentwise transports stored in H, and the dependency record is exactly P together with the local ledger N.

Since no coordinate of D is a total truth predicate, a classical halting decision, or a divergence oracle, the read stops at the displayed diagonal boundary. It exposes where the self-reference pressure occurs without converting that pressure into a global classifier. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionDiagonalBoundary`

Theorem 491.4 (HaltingDistinction consumer non-escape)**.** *Every consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet receives only the finite certificate-history pair row* Q*, the finite* Cont_P *trace row* T*, the diagonal self-reference row* Z*, the componentwise* hsame *transports* H*, the displayed read routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *ledger* N*. No consumer can extract from the packet a total halting predicate, a global truth predicate over certificates, a classical decidability principle for termination, or a host-level evaluator.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier consumer non escape`

Proof. Inspect the carrier coordinates. The exported rows are exactly Q, T, Z, H, C, P, N, and each row is finite BEDC data.

Apply Definition 491.2 to ordinary trace consumers and Definition 491.3 to self-reference consumers. Both routes remain inside the displayed packet.

Therefore every public read factors through the named rows. The carrier has no channel through which a total halting predicate, global truth predicate, classical decision principle, or host evaluator can enter the consumer surface. □

Theorem 491.5 (HaltingDistinction pair classifier transport)**.** *Let* $D = (Q, T, Z, H, C, P, N)$ *be an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier. Any public classifier read of the certificate-history pair row* Q *is transported only through the componentwise* hsame *row recorded in* H*. The packet therefore classifies pair rows by the displayed certificate-history data and the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *ledger, not by a global truth predicate or host evaluator.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionPairClassifierTransport`

Proof. Start with the carrier of Definition 491.1. The only coordinate naming the certificate-history input is Q, and the only transport coordinate for pair rows is the componentwise hsame data in H.

Apply Definition 491.2. Its classifier transport obligation says that pair rows classify through the displayed transport row, while the local ledger records only the finite naming surface.

Use Definition 491.3 for the self-reference case and Definition 491.4 for all consumer reads. Neither route adds a global truth predicate, a classical termination decision, divergence oracle data, or a host evaluator. □

Theorem 491.6 (HaltingDistinction trace readability)**.** *Let* $D = (Q, T, Z, H, C, P, N)$ *be an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier. Every termination-facing public read of* D *factors through the finite trace row* T *and the displayed route row* C*. The read may inspect the visible* Cont_P *chain segment recorded in* T*, but it cannot turn that finite segment into a total halting predicate or divergence oracle.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionTraceReadability`

Proof. The carrier definition supplies T as the only finite trace coordinate and C as the only route coordinate for replaying that trace. Thus any termination-facing read first projects to those rows.

The trace-readability clause of Definition 491.2 binds the public read to the finite Cont_P segment in T and to the routes in C.

If the read reaches the self-reference point, apply Definition 491.3; otherwise apply Definition 491.4. In both cases the read remains within the displayed finite packet and has no coordinate from which to recover a total halting classifier. □

Theorem 491.7 (HaltingDistinction finite trace inversion)**.** *Let $D = (Q, T, Z, H, C, P, N)$ be an accepted* HaltingDistinction$^{\uparrow}$ *carrier. If a public termination-facing read of D reaches a displayed* Cont *step, then the read determines its source pair row Q, finite trace row T, diagonal row Z when the read enters the self-reference branch, transport row H, route row C, provenance row P, and local ledger row N.*

This inversion is exact only for the displayed finite packet. It reconstructs no hidden total halting classifier, divergence oracle, global truth predicate, or host evaluator.

Proof. Project the accepted carrier. By Definition 491.1, the only coordinates available to a termination-facing read are Q, T, Z, H, C, P, N, and the only displayed continuation routes are recorded in C.

Apply Definition 491.6. A read that reaches a displayed Cont step is already inside the finite trace row T and route row C, so the source pair row Q, transport row H, provenance row P, and local ledger row N are fixed by the carrier projection.

If the route enters the self-reference branch, apply Definition 491.3. The diagonal row Z is then the displayed self-application row reached by the route, and no external decision principle is added at that point.

Finally apply Definition 491.4. Since every public consumer receives only Q, T, Z, H, C, P, N, the inversion cannot reconstruct a hidden total halting classifier, divergence oracle, global truth predicate, or host evaluator. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionFiniteTraceInversion`

Theorem 491.8 (HaltingDistinction provenance purity boundary)**.** *Let $D = (Q, T, Z, H, C, P, N)$ be an accepted* HaltingDistinction$^{\uparrow}$ *carrier. The provenance row P and the local naming row N certify only the displayed finite carrier, transport, route, trace, and diagonal rows. They cannot be read as a total halting predicate, a global truth predicate over certificates, a divergence oracle, or a host evaluator.*

Proof. By Definition 491.1, the provenance coordinate P ranges over the displayed BHist, Ext, Cont, hsame, Pkg, diagonal, and local naming rows. The local ledger N names that finite packet and no further classifier.

Apply Definition 491.2 to identify the finite obligations certified by P and N. Apply Definition 491.3 to the diagonal read, where the provenance records the route to the self-reference row without deciding the classifier question globally.

Finally apply Definition 491.4. Since no consumer of the packet can extract a total halting predicate, global truth predicate, classical decision principle, divergence oracle, or host evaluator, the provenance and naming rows cannot be reinterpreted as any of those objects. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier provenance purity boundary`

Theorem 491.9 (HaltingDistinction normal-form route consumption)**.** *Let $D = (Q, T, Z, H, C, P, N)$ be an accepted* HaltingDistinction$^{\uparrow}$ *carrier. Every accepted public read that reaches the trace-normal-form route consumes the* ContinuationTraceNormalForm$^{\uparrow}$ *sibling surface of chapter 683 before the diagonal row Z can be used. The read remains a finite route through Q, T, H, C, P, N into Z; it does not obtain a host evaluator, an unbounded execution semantics, or a total halting classifier.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionNormalFormRouteConsumption`

Proof. First project the carrier. By Definition 491.1, the only trace coordinate is T, the only displayed continuation routes are in C, and the only transport row for carrying the trace into later reads is H.

Apply Definition 491.6 and Definition 491.17. A termination-facing read reaches the finite Cont_P segment through T and C, and the normal-form sibling supplies the named finite normal-form surface consumed by that same route.

Only after that route has been consumed may the read enter the diagonal row. Definition 491.3 identifies Z as the displayed self-reference boundary, while Definition 491.8 keeps P and N from becoming evaluator or truth-predicate data. □

Theorem 491.10 (HaltingDistinction inscription-event boundary)**.** *Let $D = (Q, T, Z, H, C, P, N)$ be an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier. Every diagonal public read consumes the inscription-event record surface of chapter 519 before any self-reference boundary is exposed. The read records the diagonal pair and self-application event through Q, Z, H, C, P, N and exports no total halting predicate, divergence oracle, global truth predicate, or host evaluator.*

Proof. Begin with the diagonal coordinates. The carrier names the certificate-history pair in Q and the self-reference row in Z, with H carrying the componentwise hsame transports and C carrying the displayed route.

Apply Definition 491.16. The diagonal pair is consumed together with the inscription-event sibling before the public obstruction read is exposed, so the inscription-facing record is part of the route rather than an external semantic oracle.

Now apply Definition 491.3. The self-reference boundary becomes visible only at the displayed row Z after the finite inscription and route data have been read. Definition 491.4 excludes any hidden halting predicate, divergence oracle, global truth predicate, or host evaluator from that exposure. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier inscription event boundary`

Theorem 491.11 (HaltingDistinction downstream obstruction handoff)**.** *Every* $\mathsf{HaltingObstruction}^{\uparrow}$ *consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet receives only the finite rows*

$$Q, T, Z, H, C, P, N$$

together with the named sibling routes through chapter 492, chapter 683, chapter 519, and chapter 490. The handoff is a finite obstruction-facing boundary and does not supply a total halting predicate.

Proof. Project the accepted packet. Definition 491.4 says every consumer receives only the rows Q, T, Z, H, C, P, N, and Definition 491.23 locks the trace, inscription, and normal-form siblings to that same finite packet.

Next read the obstruction-facing package. Definition 491.14 combines diagonal boundary, trace readability, provenance purity, and consumer non-escape as the public obstruction surface.

Finally apply Definition 491.15 and Definition 491.18. They identify the downstream $\mathsf{HaltingObstruction}^{\uparrow}$ read as a consumer of the finite distinction boundary, not as a source of a total halting predicate, divergence oracle, global truth predicate, or host evaluator. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionDownstreamObstructionHandoff`

Theorem 491.12 (HaltingDistinction diagonal trace consumer totality)**.** *Every downstream obstruction-facing consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet reads only the finite rows*

$$Q, T, Z, H, C, P, N$$

together with the named trace, inscription-event, trace-normal-form, and obstruction sibling routes before it sees the diagonal boundary. The consumer totality is finite: it does not supply a total halting predicate, divergence oracle, global truth predicate, or host evaluator.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier diagonal trace consumer totality`

Proof. First read the finite trace surface. Apply Definition 491.6 and Definition 491.7; any termination-facing consumer that reaches a displayed Cont step has already fixed Q, T, Z, H, C, P, N when the diagonal branch is entered.

Next read the certificate boundary. Use Definition 491.8 to keep P, N as provenance and local naming rows, and use Definition 491.9 to force trace-normal-form access through the displayed finite route.

Finally apply Definition 491.10 and Definition 491.11. These place the inscription-event and obstruction sibling routes before the downstream diagonal read. The cited non-escape clauses exclude total halting predicates, divergence oracles, global truth predicates, and host evaluators from the consumer surface. □

Theorem 491.13 (HaltingDistinction finite obstruction ledger exhaustion)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet, the obstruction ledger exposed to downstream consumers is exhausted by* Q, T, Z, H, C, P, N *plus the trace, inscription-event, trace-normal-form, and* $\mathsf{HaltingObstruction}^{\uparrow}$ *sibling routes named in Definition 491.11. No further coordinate is available for a total halting predicate, divergence oracle, global truth predicate, or host evaluator.*

Proof. Start with Definition 491.12. It already identifies the complete finite consumer route before the diagonal boundary is exposed.

Read the obstruction ledger through Definition 491.11. The handoff sends each downstream obstruction-facing consumer through exactly the finite rows Q, T, Z, H, C, P, N and through the named sibling routes.

The exhaustion clause follows from Definition 491.8 together with Definition 491.7. Since the trace inversion reconstructs only the displayed finite packet and the provenance row cannot be reinterpreted as semantic evaluation data, the obstruction ledger has no coordinate for any total halting, divergence, truth, or host-evaluation object. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier finite obstruction ledger exhaustion`

Theorem 491.14 (HaltingDistinction diagonal obstruction public package)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the public diagonal obstruction is exposed only through the certificate-history pair row Q*, the finite trace row* T*, the diagonal self-reference row* Z*, componentwise* hsame *transports* H*, displayed routes* C*, provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *row* N*. The package combines the diagonal boundary, trace readability, provenance purity, and consumer non-escape rows; it exports no total halting predicate, global truth predicate, divergence oracle, or host evaluator.*

Proof. First read the diagonal boundary. Definition 491.3 forces every public diagonal read through Q, T, Z, H, C, P, N and stops the read at the self-reference boundary rather than at a global decision principle.

Next read the trace and provenance rows. Definition 491.6 keeps termination-facing inspection inside the finite trace T and route row C, while Definition 491.8 prevents P and N from being reinterpreted as a truth predicate, divergence oracle, or host evaluator.

Finally apply consumer non-escape. Definition 491.4 says every public consumer receives only Q, T, Z, H, C, P, N, so the package exposes the diagonal obstruction without making any total halting-decision claim. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionDiagonalObstructionPublicPackage`

Theorem 491.15 (HaltingDistinction obstruction sibling route)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the finite certificate-history pair row Q, finite trace row T, diagonal self-reference row Z, transports H, routes C, provenance P, and local naming row N are exactly the sibling boundary consumed by chapter 490 before an obstruction ledger is exposed. This route is a finite downstream consumer relation, not a global halting classifier.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionObstructionSiblingRoute`

Proof. Project the accepted distinction packet. The rows Q, T, Z, H, C, P, N are the only certificate-history, trace, diagonal, transport, route, provenance, and local naming coordinates visible to a consumer.

Apply Definition 491.14. It packages the diagonal boundary, trace readability, provenance purity, and consumer non-escape rows as the public obstruction-facing surface of $\mathsf{HaltingDistinction}^{\uparrow}$.

Now read the downstream consumer named in chapter 490. Its obstruction ledger is exposed only after the distinction boundary has supplied the certificate-history pair, finite trace, and diagonal self-reference rows.

Thus the sibling route is exact at the finite boundary Q, T, Z, H, C, P, N. Since neither side supplies a total halting predicate, global truth predicate, divergence oracle, or host evaluator, the route records only the finite obstruction consumer relation. □

Theorem 491.16 (HaltingDistinction inscription-event route)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the diagonal pair row Q is consumed together with the inscription-event record of chapter 519 before any public obstruction read is exposed. The route records the finite certificate-history pair, trace, diagonal row, transport, continuation, provenance, and local name; it does not export a host evaluator or total halting predicate.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier inscription event route`

Proof. Project the carrier of Definition 491.1. The diagonal pair row Q and the diagonal self-reference row Z are visible only with the finite trace, transport, route, provenance, and local naming rows.

Apply Definition 491.3 and Definition 491.6. These force the diagonal read to stay inside Q, T, Z, H, C, P, N before the route can be offered to any downstream consumer.

Now read the inscription-event sibling. The cited chapter records the event row that names the inscription-facing boundary, so the diagonal pair is consumed with that record before the public obstruction route of Definition 491.15 is exposed.

Finally use Definition 491.4. The resulting route has no coordinate for a total halting predicate, global truth predicate, divergence oracle, or host evaluator. □

Theorem 491.17 (HaltingDistinction trace normal-form scope)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the trace row T is scoped through chapter 683. A downstream consumer may read the finite normal-form trace route, transported through H and replayed through C, but it cannot replace T by a host evaluator, an unbounded execution semantics, or a total halting classifier.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionTraceNormalFormScope`

Proof. Begin with trace readability. Definition 491.6 identifies T as the finite trace segment available to a termination-facing inspection, and it keeps that read under the displayed route C.

Use the carrier and diagonal-boundary rows to keep the trace tied to Q and Z. The packet has no independent evaluator coordinate, so the trace cannot be detached from the certificate-history pair and diagonal row.

Now cite the normal-form sibling chapter 683. It supplies the finite trace-normal-form surface consumed by the downstream route, while H, C, P, N preserve the same displayed packet.

Consumer non-escape then excludes a host evaluator, unbounded execution semantics, divergence oracle, and total halting classifier from the trace route. □

Theorem 491.18 (HaltingDistinction obstruction consumer boundary)**.** *Every downstream obstruction consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet must pass through*

$$Q, T, Z, H, C, P, N$$

before reading chapter 490. The consumer receives the certificate-history pair, finite trace row, diagonal self-reference row, componentwise transport, displayed continuation, package provenance, and local naming row, and it receives no total halting predicate, host evaluator, divergence oracle, or global truth classifier.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionObstructionConsumerBoundary`

Proof. Project the accepted packet through Definition 491.1. The only obstruction-facing rows are Q, T, Z, H, C, P, N.

Apply Definition 491.3 and Definition 491.6. They place public diagonal and trace reads inside the displayed certificate-history and continuation boundary.

Use Definition 491.4 and Definition 491.15. The first excludes hidden evaluator and total-classifier rows; the second identifies the finite boundary consumed by the HaltingObstruction sibling.

Thus the downstream obstruction consumer can reach the cited obstruction ledger only after the finite packet Q, T, Z, H, C, P, N has been displayed. □

Theorem 491.19 (HaltingDistinction carrier obligation)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the public carrier exposes exactly the certificate-history pair Q, finite trace segment T, diagonal row Z, componentwise transport row H, displayed route row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *row N. These are the only rows available for certificate-history pairing, finite trace reading, diagonal inspection, transport, routing, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier carrier obligation`

Proof. Project the carrier from Definition 491.1. The accepted packet has the seven displayed coordinates Q, T, Z, H, C, P, N, and no extra row is introduced by the carrier projection.

Apply Definition 491.2. It identifies P and N as the finite provenance and local naming witnesses for exactly the displayed certificate-history, trace, diagonal, transport, and route rows.

The conclusion is therefore row-exact: every public read of the carrier must factor through one of Q, T, Z, H, C, P, N, and no hidden global classifier row is exposed. □

Theorem 491.20 (HaltingDistinction classifier stability)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

any classifier read of the certificate-history pair row Q is stable only along the displayed componentwise hsame *row H. The read may transport the finite rows named by Q, T, Z, C, P, N, but it cannot invoke a global truth predicate, a divergence oracle, a total halting predicate, or a host evaluator outside the accepted packet.*

Proof. Start with the exact carrier surface of Definition 491.19. The only certificate-history pair exposed to a classifier is Q, and its only transport witness is the displayed componentwise row H.

Apply Definition 491.2. The local $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ row certifies Q, T, Z, H, C, P, N as the finite packet rows, so a classifier read has no additional source for truth, divergence, total termination, or host evaluation.

Finally use Definition 491.8 and Definition 491.4. These prevent the provenance and consumer rows from being reinterpreted as external semantic or evaluator data. Thus stability of Q is exactly the stability transported by H. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionClassifierStability`

Theorem 491.21 (HaltingDistinction root carrier-route exhaustion)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

every root read exposed by the packet factors through the carrier row Q, the finite trace row T, the route row C, the provenance row P, and the local $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *row N, with the diagonal row Z and componentwise* hsame *transport row H used only as displayed coordinates of that same finite packet. No root read has access to a row outside the carrier, finite trace, route, provenance, and local naming surface.*

Proof. Project the packet by Definition 491.1. The carrier lists exactly Q, T, Z, H, C, P, N, so the root read begins with the displayed BHist rows and has no hidden coordinate to inspect.

Apply Definition 491.19. It turns the carrier projection into the public obligation surface and identifies the finite rows that may be consumed by root readers.

Now use Definition 491.6. A termination-facing read must pass through the finite Cont_P trace row T and the route row C, while P and N remain the Pkg provenance and local $\mathsf{NameCert}$ ledger for exactly those rows. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootCarrierRouteExhaustion`

Theorem 491.22 (HaltingDistinction root diagonal non-escape)**.** *For every accepted* HaltingDistinction$^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

any diagonal root read factors through Q, T, Z, H, C, P, N*. The self-reference row may identify the classifier question with its own certificate-history input through the displayed* hsame *transports, but the read cannot produce a total halting predicate, a global truth predicate, a divergence oracle, or a host evaluator.*

Proof. Start with the diagonal boundary of Definition 491.3. It routes the self-reference read through Q, T, Z, H, C, P, N and stops the read at the displayed diagonal row.

Apply Definition 491.8. The Pkg provenance row P and local naming row N certify only the finite carrier, trace, route, transport, and diagonal data.

Finally use Definition 491.4. Since every public consumer receives only the displayed rows, the diagonal read cannot be reinterpreted as a total halting predicate, global truth predicate, divergence oracle, or host evaluator. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootDiagonalNonescape`

Theorem 491.23 (HaltingDistinction root sibling consumer lock)**.** *The same finite boundary*

$$Q, T, Z, H, C, P, N$$

is the only sibling route consumed by HaltingDistinctionTrace$^{\uparrow}$ *in chapter 492, by* InscriptionEvent$^{\uparrow}$ *in chapter 519, and by* ContinuationTraceNormalForm$^{\uparrow}$ *in chapter 683. These consumers may read the finite* Cont *trace, diagonal inscription row, componentwise* hsame *transports,* Pkg *provenance, and local* NameCert *ledger, but none may extend the route to an external halting classifier or host evaluator.*

Proof. First read the root carrier-route exhaustion theorem. Definition 491.21 fixes the available rows as Q, T, Z, H, C, P, N and assigns trace, route, provenance, and local naming roles to them.

For the diagonal consumer, apply Definition 491.22. The inscription-facing read can use Z only inside the same finite boundary and cannot turn the diagonal row into evaluator data.

The sibling chapters named in the statement consume exactly those rows: the trace chapter consumes the finite Cont trace surface, the inscription chapter consumes the diagonal event surface, and the trace-normal-form chapter consumes the finite normal-form route. Combining those readings locks all three consumers to the same BHist, Cont, hsame, Pkg, and NameCert packet. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootSiblingConsumerLock`

Theorem 491.24 (HaltingDistinction route stability)**.** *Every termination-facing route from an accepted* HaltingDistinction$^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N)$$

factors through the displayed rows Q, T, Z, H, C, P, N*. Its finite continuation replay is recorded by* C*, with* H *transporting only componentwise row equality and with* P, N *supplying provenance and local naming. No route can bypass these rows through a global truth predicate, total halting classifier, divergence oracle, or host evaluator.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier route stability`

Proof. First project the accepted packet by Definition 491.19. The public carrier contributes exactly Q, T, Z, H, C, P, N.

Next apply Definition 491.6 and Definition 491.3. Termination-facing inspection reads the finite trace row T, and diagonal inspection reaches Z only through the displayed route row C.

The remaining rows cannot open an external path. By Definition 491.8 and Definition 491.4, P and N remain certificate data and no consumer may replace the displayed route by a host evaluator or total halting predicate. Hence every termination-facing route is the finite replay through C over Q, T, Z, H, P, N. □

Theorem 491.25 (HaltingDistinction root NameCert package)**.** *The accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *root package consists exactly of carrier admission for* Q, T, Z, H, C, P, N*, classifier stability through* H*, finite trace readability through* T*, the diagonal boundary at* Z*, provenance purity through* P, N*, and consumer non-escape along the displayed route* C*. This package is a local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *surface and does not export a total halting predicate or classical decidability claim.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionNameCertRootPackage`

Proof. Carrier admission is Definition 491.19, and the local naming obligations are those of Definition 491.2. Together they expose exactly Q, T, Z, H, C, P, N.

Classifier stability is supplied by Definition 491.20. The classifier may read Q only through the componentwise transport row H.

The route-facing rows are fixed by Definition 491.6, Definition 491.3, and Definition 491.24. These force trace reads through T, diagonal reads through Z, and continuation replay through C.

Finally apply Definition 491.8 and Definition 491.4. They keep P, N as provenance and naming data and exclude any export of a total halting predicate, classical decidability row, global truth predicate, divergence oracle, or host evaluator. □

Theorem 491.26 (HaltingDistinction continuation boundary)**.** *Every termination-facing read from an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N)$$

factors through the displayed finite Cont *segment* T *and route row* C*. The continuation boundary records finite trace readability and diagonal routing only; it does not supply a total halting predicate.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionContinuationBoundary`

Proof. Read the finite trace first. By Definition 491.6, termination-facing inspection reaches the packet through the finite trace segment T rather than through an ambient evaluator.

Now apply Definition 491.3. The diagonal row Z is reachable only along the displayed route row C, with H transporting the already named components.

Since Definition 491.8 prevents the provenance and naming rows from being reinterpreted as a truth predicate, divergence oracle, or host evaluator, the read stops at the finite continuation boundary and never becomes a total halting predicate. □

Theorem 491.27 (HaltingDistinction non-escape package)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet, the carrier obligation, trace readability, diagonal boundary, provenance purity, and obstruction sibling route together exclude any exported global truth predicate, divergence oracle, classical termination decidability row, or host evaluator row. The package exposes only the finite rows* Q, T, Z, H, C, P, N *and the downstream obstruction route that consumes them.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier decision refusal surface`

Proof. Use Definition 491.19 to fix the public rows of the accepted packet. The carrier contributes only Q, T, Z, H, C, P, N.

Apply Definition 491.6 and Definition 491.3. These force termination-facing and diagonal-facing reads through the finite trace, diagonal, transport, and route rows.

Next apply Definition 491.8. The provenance and local naming rows remain certificate data, not a global truth predicate, divergence oracle, classical decision principle, or host evaluator.

Finally use Definition 491.15. The downstream obstruction consumer receives the same finite boundary, so the whole package is non-escaping at exactly the displayed rows. □

Theorem 491.28 (HaltingDistinction root carrier surface)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

the root carrier exposes exactly the seven displayed rows Q, T, Z, H, C, P, N*. The row* Q *is the certificate-history pair,* T *is the finite* Cont_P *trace segment,* Z *is the diagonal self-reference row,* H *is the componentwise* hsame *transport row,* C *is the displayed* Cont *route row,* P *is the* Pkg *provenance row, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$ *ledger. No accepted packet contains a total halting coordinate, a global truth row, a divergence oracle row, or a host evaluator row.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootCarrierSurface`

Proof. Project the accepted carrier using Definition 491.1. Its displayed coordinates are precisely Q, T, Z, H, C, P, N, with the roles named in the statement and no further coordinate.

Apply Definition 491.2 to bind those rows as the local carrier, transport, trace, diagonal, and non-escape obligations. Apply Definition 491.3 and Definition 491.6 to locate the diagonal and trace reads inside the same packet.

Finally use Definition 491.15 and Definition 491.27. Downstream obstruction consumers receive exactly the displayed finite ledger, and the package exports no total halting coordinate, global truth row, divergence oracle row, or host evaluator row. □

Theorem 491.29 (HaltingDistinction root classifier transport)**.** *For every accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet*

$$D = (Q, T, Z, H, C, P, N),$$

any public classifier read of the root surface transports certificate-history data only through H*. If the read follows the continuation-facing route, it reaches the displayed finite segment through* T*, the diagonal row through* Z*, and the public route row through* C*; it never reads a classifier from an unrecorded transport, an ambient evaluator, or a total halting predicate.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootClassifierTransport`

Proof. Start with Definition 491.1. The only transport coordinate in the carrier is H, so every comparison of the certificate-history pair named by Q must use the componentwise hsame data recorded there.

Apply Definition 491.2. Its classifier-transport and trace-readability clauses keep public reads on the displayed transport and continuation rows, while Definition 491.6 identifies T and C as the finite continuation-facing route.

For the self-reference branch, apply Definition 491.3. The diagonal read reaches Z only along the displayed route and then returns to the finite ledger.

Use Definition 491.15 and Definition 491.27 for downstream consumers. They receive the same routed surface and no unrecorded transport, ambient evaluator, or total halting predicate. □

Theorem 491.30 (HaltingDistinction root ledger non-escape)**.** *Every downstream* HaltingObstruction$^{\uparrow}$ *or trace-normal-form consumer of an accepted* HaltingDistinction$^{\uparrow}$ *packet must read the same finite ledger* Q, T, Z, H, C, P, N*. Such a consumer can use the certificate-history pair, finite trace segment, diagonal row, componentwise transports, displayed routes, provenance, and local* NameCert *ledger, but it cannot extract a global truth predicate, divergence oracle, total evaluator, or hidden halting decision from that ledger.*

Proof. Project the ledger from Definition 491.1. The accepted packet exposes only Q, T, Z, H, C, P, N, and Definition 491.2 records those rows as the local obligations available to public readers.

Apply Definition 491.6 for trace-normal-form reads. They inspect the finite continuation segment T through the displayed route row C, with transports supplied only by H.

Apply Definition 491.3 and Definition 491.15 for obstruction reads. The route to HaltingObstruction$^{\uparrow}$ consumes the same certificate-history, trace, diagonal, transport, route, provenance, and naming rows.

Finally apply Definition 491.27. Since the packet exports only the finite ledger, downstream consumers cannot extract a global truth predicate, divergence oracle, total evaluator, or hidden halting decision. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootLedgerNonescape`

Theorem 491.31 (HaltingDistinction trace-normal-form handoff)**.** *Every trace-normal-form consumer of an accepted* HaltingDistinction$^{\uparrow}$ *packet receives only the finite trace row* T*, the route row* C*, the transport row* H*, the provenance row* P*, and the local ledger* N*. The rows* Q *and* Z *remain the surrounding certificate-history and diagonal rows for that handoff, so the consumer reads the normal-form trace surface without acquiring a total halting predicate, host evaluator, divergence oracle, or hidden classifier.*

Proof. Project the packet by Definition 491.1. The only displayed rows are Q, T, Z, H, C, P, N, and the trace-facing coordinate is the finite continuation segment T.

Apply Definition 491.6 and Definition 491.29. These force every trace-normal-form read through T and C, with componentwise comparison data supplied by H and provenance supplied by P, N.

The rows Q and Z are not exported as new evaluators. By Definition 491.3 and Definition 491.30, they remain the surrounding certificate-history and diagonal rows of the same finite ledger. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionTraceNormalFormHandoff`

Theorem 491.32 (HaltingDistinction inscription-event handoff)**.** *Every inscription-event consumer of an accepted* HaltingDistinction$^{\uparrow}$ *packet receives the diagonal pair row only through the displayed route*

$$Q, T, Z, H, C, P, N.$$

The event read may use the certificate-history pair, finite trace segment, diagonal row, componentwise transports, displayed routes, provenance, and local ledger, but it cannot turn that diagonal pair into an external evaluator, total halting predicate, divergence oracle, or host truth classifier.

Proof. Use Definition 491.1 to expose the diagonal row Z inside the finite packet Q, T, Z, H, C, P, N. There is no separate evaluator coordinate in the carrier.

Apply Definition 491.3. The diagonal pair can be read only along the route row C, with the certificate-history pair named by Q and the componentwise transports named by H.

Finally apply Definition 491.8 and Definition 491.27. The provenance and local ledger remain Pkg and NameCert rows, so the inscription-event consumer cannot reinterpret the diagonal pair as a total halting predicate or external evaluator. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionInscriptionEventHandoff`

Theorem 491.33 (HaltingDistinction downstream sibling package)**.** *The trace-normal-form route of chapter 683, the inscription-event route of chapter 519, the finite trace surface of chapter 492, and the obstruction route of chapter 490 jointly consume an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet only through the finite boundary* Q, T, Z, H, C, P, N*. This joint sibling package stays inside* BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert *data and exports no total halting predicate, external evaluator, divergence oracle, or global truth classifier.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionObstructionSiblingJointPackage`

Proof. First use Definition 491.31. It gives the trace-normal-form consumer the finite trace, route, transport, provenance, and local ledger rows while keeping the certificate-history and diagonal rows as surrounding ledger data.

Next use Definition 491.32. It routes the inscription-event read of the diagonal pair through exactly the displayed Q, T, Z, H, C, P, N boundary.

For the obstruction side, apply Definition 491.15 and Definition 491.30. The obstruction consumer receives the same finite ledger and no extra evaluator row.

Combining these routes gives the stated downstream sibling boundary. The non-escape clauses of Definition 491.27 exclude total halting predicates, external evaluators, divergence oracles, and global truth classifiers from the joint package. □

Theorem 491.34 (HaltingDistinction root obstruction handoff lock)**.** *Every downstream obstruction consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet reads only the finite packet*

$$Q, T, Z, H, C, P, N.$$

The obstruction route may inspect the certificate-history pair, finite trace row, diagonal row, componentwise transports, displayed route, provenance, and local ledger, but it cannot import a total halting coordinate, host evaluator, divergence oracle, or global truth classifier.

Proof. Start with the root carrier surface of Definition 491.28. It fixes Q, T, Z, H, C, P, N as the complete displayed packet.

Apply Definition 491.6 and Definition 491.3. These locate the trace and diagonal reads inside T and Z, with the displayed route row C and transport row H supplying the only admissible access path.

Finally compose Definition 491.30 with Definition 491.33. The obstruction consumer is one of the joint sibling routes, so it receives the same finite ledger and no external halting data. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootObstructionHandoffLock`

Theorem 491.35 (HaltingDistinction root trace-normal-form lock)**.** *Every trace-normal-form consumer downstream of* $\mathsf{HaltingDistinction}^{\uparrow}$ *reads only the finite packet*

$$Q, T, Z, H, C, P, N.$$

Its normal-form trace read is the displayed continuation segment with route and provenance data already present in that packet; no ambient evaluator, total halting predicate, hidden classifier, or divergence oracle is added at the handoff.

Proof. Project the accepted packet using Definition 491.28. The only rows available to the consumer are Q, T, Z, H, C, P, N.

Use Definition 491.6 to force the trace-normal-form read through T and C. Use Definition 491.3 to keep the diagonal row Z inside the same routed ledger rather than turning it into evaluator data.

The handoff is then the trace-normal-form component of Definition 491.33. Together with Definition 491.30, this prevents the consumer from reading any row outside the finite packet. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootTraceNormalFormLock`

Theorem 491.36 (HaltingDistinction root inscription-event lock)**.** *Every inscription-event consumer downstream of* $\mathsf{HaltingDistinction}^{\uparrow}$ *reads only the finite packet*

$$Q, T, Z, H, C, P, N.$$

The event may use the diagonal self-reference row and its displayed route, but the lock keeps that read inside the carrier, trace readability, diagonal boundary, root ledger non-escape, and joint sibling package surface.

Proof. Read the accepted carrier through Definition 491.28. The packet exposes Q, T, Z, H, C, P, N and no evaluator coordinate.

Apply Definition 491.3. The inscription event reaches Z only along the displayed route C, with Q and H supplying the certificate-history row and componentwise transports.

Now use Definition 491.6 to keep the surrounding trace data finite, and apply Definition 491.30 together with Definition 491.33. The inscription-event route is therefore locked to the same finite packet and cannot export total halting or host-truth data. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier root inscription event lock`

Theorem 491.37 (HaltingDistinction root diagonal route totality)**.** *Every diagonal consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet factors through the finite route*

$$Q, T, Z, H, C, P, N.$$

The consumer may read the certificate-history pair, the finite trace segment, the diagonal self-reference row, the componentwise transports, the displayed route, provenance, and local ledger, but it cannot read a total halting predicate, host evaluator, divergence oracle, or global truth classifier.

Proof. Start with the diagonal boundary of Definition 491.3. Any public read that reaches the self-reference point reaches Z only through the displayed route over Q, T, H, C, P, N.

Next project the root carrier surface by Definition 491.28. It identifies Q, T, Z, H, C, P, N as the full accepted packet and supplies no evaluator coordinate outside those rows.

Finally apply Definition 491.30. The local ledger cannot be reinterpreted as a total halting predicate, host evaluator, divergence oracle, or global truth classifier, so every diagonal consumer is total only as a finite route through the displayed packet. □

Lean4 ✓ BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootDiagonalRouteTotality

Theorem 491.38 (HaltingDistinction root finite trace classifier exhaustion)**.** *Every finite trace classifier read from an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet is exhausted by the displayed packet*

$$Q, T, Z, H, C, P, N.$$

The classifier may inspect the finite Cont_P *trace row and transport the certificate-history pair through* H*, but no read remains after the trace row, pair-classifier transport, provenance row, and local ledger have been accounted for.*

Proof. First use Definition 491.6. A termination-facing classifier read must pass through the finite trace row T and the displayed route row C.

Next use Definition 491.5. The certificate-history pair row Q is classified only through the componentwise hsame transport recorded in H, with the diagonal branch still bounded by the displayed row Z.

It remains to account for provenance and naming. Apply Definition 491.8 and Definition 491.30; the rows P and N certify only the finite carrier and cannot supply a hidden classifier, total halting predicate, divergence oracle, or host evaluator. □

Lean4 ✓ BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootFiniteTraceClassifierExhaustion

Theorem 491.39 (HaltingDistinction root downstream unblock package)**.** *The obstruction, trace-normal-form, inscription-event, and halting-boundary consumers of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet receive only the finite packet*

$$Q, T, Z, H, C, P, N.$$

Their shared package consists of diagonal route totality, finite trace classifier exhaustion, the root obstruction handoff lock, the trace-normal-form lock, the inscription-event lock, and the downstream sibling joint package; it exports no total halting predicate, host evaluator, divergence oracle, or global truth classifier.

Lean4 ✓ BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootDownstreamUnblockPackage

Proof. First combine Definition 491.37 with Definition 491.38. These two results bind both diagonal reads and finite trace classifier reads to Q, T, Z, H, C, P, N.

Next read the named consumer locks. Definition 491.34, Definition 491.35, and Definition 491.36 give the obstruction, trace-normal-form, and inscription-event routes the same finite packet.

Finally apply Definition 491.33. Its joint sibling surface includes the halting-facing obstruction and trace consumers, so the assembled package unblocks those downstream routes without adding any global halting, evaluator, divergence, or truth-classifier row. □

Theorem 491.40 (HaltingDistinction route obstruction triad)**.** *Every obstruction-facing consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet factors through the displayed*

$$Q, T, Z, H, C, P, N$$

route and the $\mathsf{HaltingObstruction}^{\uparrow}$ *sibling before it reads an obstruction ledger. The route consumes the certificate-history pair, finite trace row, diagonal row, componentwise transport, continuation replay, package provenance, and local naming row, and it exposes no total halting predicate, host evaluator, divergence oracle, or global truth classifier.*

Proof. Project the carrier of Definition 491.1. The only rows available to the obstruction-facing read are Q, T, Z, H, C, P, N.

Next apply Definition 491.3 and Definition 491.6. These route the self-reference and finite trace reads through Z and T along the displayed continuation row C, with transport supplied by H.

Now use the sibling route package Definition 491.15 and the joint package Definition 491.33. They attach the same finite packet to the $\mathsf{HaltingObstruction}^{\uparrow}$ ledger before any obstruction consumer is served.

Finally apply consumer non-escape from Definition 491.4. Since the displayed rows are exhausted by the routed packet and sibling handoff, no total halting predicate, host evaluator, divergence oracle, or global truth classifier remains available. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRouteObstructionTriad`

Theorem 491.41 (HaltingDistinction inscription trace gate)**.** *Every inscription-event consumer and every trace-normal-form consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet can consume only the finite trace row T and the diagonal row Z through the displayed Q, H, C, P, N support rows. Neither route can read a total evaluator, a global halting predicate, a divergence oracle, or a host truth row.*

Proof. Begin with the carrier projection of Definition 491.1. It presents T as the finite continuation trace and Z as the diagonal self-reference row, with Q, H, C, P, N supplying the surrounding support.

For trace-normal-form consumers, apply Definition 491.31. The read reaches the finite trace row through C, with componentwise comparison data supplied by H and provenance supplied by P, N.

For inscription-event consumers, apply Definition 491.32. The read reaches the diagonal row Z only along the displayed route and never through a separate evaluator coordinate.

Use Definition 491.30 and Definition 491.4 to exhaust the remaining rows. The finite trace and diagonal gates therefore export no total evaluator, global halting predicate, divergence oracle, or host truth row. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionInscriptionTraceGate`

Theorem 491.42 (HaltingDistinction consumer boundary package)**.** *The obstruction, inscription-event, and trace-normal-form consumers of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet share one finite* BHist, Cont, hsame, Pkg, *and* $\mathsf{NameCert}$ *consumer package over*

$$Q, T, Z, H, C, P, N.$$

The package is exactly the obstruction route of Definition 491.40, the inscription and trace gate of Definition 491.41, and the downstream unblock package of Definition 491.39; it adds no global halting predicate, total evaluator, divergence oracle, or truth-classifier row.

Proof. First project Definition 491.1. The carrier gives the finite BHist packet Q, T, Z, H, C, P, N and no extra consumer coordinate.

Use Definition 491.40 for obstruction-facing consumers. This routes the obstruction ledger through the displayed packet and the $\mathsf{HaltingObstruction}^{\uparrow}$ sibling.

Use Definition 491.41 for inscription-event and trace-normal-form consumers. This binds those consumers to Z and T through the same Q, H, C, P, N support rows.

Finally compose the downstream package of Definition 491.39 with consumer non-escape in Definition 491.4. Coordinate exhaustion leaves a single finite $\mathsf{BHist}/\mathsf{Cont}/\mathsf{hsame}/\mathsf{Pkg}/\mathsf{NameCert}$ package and no hidden halting, evaluator, divergence, or truth-classifier row. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionConsumerBoundaryPackage`

Theorem 491.43 (HaltingDistinction root inscription trace totality)**.** *Every root obligation read of an accepted* HaltingDistinction$^{\uparrow}$ *packet exposes the inscription-event route only through the finite rows*

$$Q, T, Z, H, C, P, N.$$

The inscription event reads the diagonal row Z with the sibling trace row T, componentwise transports H, displayed route C, package provenance P, and local ledger N; it does not expose a total halting predicate, global truth predicate, host evaluator, divergence oracle, or hidden decision procedure.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootInscriptionTraceTotality`

Proof. Project the carrier of Definition 491.1. The available rows are exactly Q, T, Z, H, C, P, N, with Z serving as the diagonal inscription row and T serving as the finite trace row.

Apply the inscription-event handoff and the inscription trace gate. They route the inscription read through Z and the surrounding support rows while keeping the trace-normal-form sibling attached to T.

Finally use the consumer boundary package. Since the package is exhausted by the finite BHist, Cont, hsame, Pkg, and NameCert rows, no total halting predicate, global truth predicate, host evaluator, divergence oracle, or hidden decision procedure can be read from the inscription route. □

Theorem 491.44 (HaltingDistinction root diagonal inscription nonescape)**.** *For an accepted* HaltingDistinction$^{\uparrow}$ *packet, the diagonal row Z cannot escape the root obligation surface as a halting decision. Any diagonal-inscription consumer reaches Z only along the displayed Q, T, H, C, P, N support, and the trace-normal-form sibling reads the same finite* Cont *route rather than a host evaluation row.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionRootDiagonalInscriptionNonescape`

Proof. Start with the diagonal boundary theorem and root ledger non-escape. They keep Z inside the local packet and forbid reinterpreting the diagonal row as evaluator data.

Use trace readability and the trace-normal-form handoff for the sibling route. The normal-form consumer reads T through the displayed continuation row C, with transports supplied by H and provenance by P, N.

The inscription-event handoff then binds the diagonal read to the same finite packet. Thus neither the diagonal row nor its trace-normal-form sibling yields a total halting predicate, global truth predicate, host evaluator, divergence oracle, or hidden decision procedure. □

Theorem 491.45 (HaltingDistinction root consumer obstruction exhaustion)**.** *Every obstruction-facing root consumer of an accepted* HaltingDistinction$^{\uparrow}$ *packet factors through the displayed carrier rows*

$$Q, T, Z, H, C, P, N$$

and the obstruction handoff route. The consumer may inspect the certificate-history pair, finite trace, diagonal inscription row, componentwise transports, displayed continuation route, provenance, and local NameCert *ledger, but it cannot add a total halting predicate, global truth predicate, host evaluator, divergence oracle, or hidden decision procedure.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionCarrier root consumer obstruction exhaustion`

Proof. Project the carrier rows and apply the route obstruction triad. This sends obstruction-facing consumption through the displayed packet and the $\mathsf{HaltingObstruction}^{\uparrow}$ sibling before any obstruction ledger is read.

Next apply the root inscription trace totality and root diagonal inscription nonescape theorems above. They bind the inscription and trace-normal-form sides to the same Q, T, Z, H, C, P, N surface.

Finally use the consumer boundary package with consumer non-escape. Coordinate exhaustion leaves only the finite obstruction handoff rows, so the downstream consumer receives no total halting predicate, global truth predicate, host evaluator, divergence oracle, or hidden decision procedure. □

Theorem 491.46 (HaltingDistinction observation boundary)**.** *Every observation-facing consumer of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *carrier must read the certificate-history pair, finite trace row, diagonal row, inscription-event row, trace-normal-form route, obstruction ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows before any obstruction conclusion is exposed. The read is bounded by the finite BEDC rows*

$$Q, T, Z, H, C, P, N$$

and by the sibling surfaces of $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$, *and* $\mathsf{HaltingObstruction}^{\uparrow}$. *It does not expose a total halting predicate, host evaluator, divergence oracle, global truth classifier, or unbounded execution semantics.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionObservationBoundary`

Proof. First apply Definition 491.43. It sends observation of the inscription event through the finite diagonal row Z, trace row T, transports H, route row C, provenance row P, and local ledger N.

Next apply Definition 491.44. This keeps the diagonal inscription and trace-normal-form route inside the same finite continuation surface, so the consumer cannot reinterpret the read as host evaluation data.

Finally apply Definition 491.45 together with the sibling trace, inscription-event, trace-normal-form, and obstruction surfaces named in chapter 491. The obstruction-facing conclusion is therefore available only after the displayed certificate-history, trace, diagonal, inscription, normal-form, obstruction, transport, continuation, package, and naming rows have been read, and no total halting predicate, divergence oracle, global truth classifier, host evaluator, or unbounded execution semantics is introduced. □

Theorem 491.47 (HaltingDistinction public export package)**.** *The public export of an accepted* $\mathsf{HaltingDistinction}^{\uparrow}$ *packet factors through the bounded trace surfaces of chapter 280 and chapter 199, the* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *row of chapter 492, the* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *route of chapter 683, the* $\mathsf{InscriptionEvent}^{\uparrow}$ *record of chapter 519, and the* $\mathsf{HaltingObstruction}^{\uparrow}$ *ledger of chapter 490. Only after those finite rows have been read may the package expose the obstruction boundary carried by* Q, T, Z, H, C, P, N.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionPublicExportPackage`

Proof. First read the machine-facing surface. The public $\mathsf{TuringMachine}^{\uparrow}$ certificate of Definition 280.13 and the bounded $\mathsf{Computable}^{\uparrow}$ certificate of Definition 199.20 provide only finite configuration traces, bounded readbacks, and displayed provenance.

Next project the accepted $\mathsf{HaltingDistinction}^{\uparrow}$ packet through Definition 491.1. The local carrier supplies the certificate-history row Q, finite trace row T, diagonal row Z, componentwise transports H, continuation routes C, package provenance P, and local ledger N.

The sibling routes then refine this packet without adding an evaluator row. Apply Definition 491.4, Definition 491.45, and Definition 491.11; these force the trace, normal-form, inscription-event, and obstruction consumers to use the same finite rows.

Finally apply Definition 491.18 and Definition 491.42. The public package therefore exposes the finite obstruction boundary only after the bounded trace, distinction trace, normal-form, inscription-event, and obstruction ledgers have been consumed, and it exports no total halting predicate, divergence oracle, host evaluator, or global truth classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}$.

Constructive story: A $\mathsf{HaltingDistinction}^{\uparrow}$ packet is generated from a certificate-history pair row, a finite continuation trace row, a diagonal self-reference row, componentwise transports, displayed continuation routes, package provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 491.1, Definition 491.2, Definition 491.3, Definition 491.4, Definition 491.45, Definition 491.42, and Definition 491.47 bind the finite carrier, naming obligations, diagonal boundary, root consumer obstruction exhaustion, downstream consumer package, and public export package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaltingDistinction}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.HaltingDistinctionUp.HaltingDistinctionConsumerBoundaryPackage`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a total halting predicate, global truth predicate, divergence oracle, host evaluator, classical termination decision, or bridge-checked standard model.

Upgrade path: No theory-axis upgrade remains; future work is only bridge verification if available.

Chapter 492

A Concrete Naming Certificate for $\mathsf{HaltingDistinctionTrace}^{\uparrow}$

The $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ surface refines the halting-as-distinction interface of chapter 9368 to the finite trace actually inspected by a certificate-level diagonal reader. It records the visible continuation segment, the distinction classifier row, the self-application row, and the finite certificate routes through which the fixed-point obstruction is read.

The trace packet consumes the broader $\mathsf{HaltingDistinction}^{\uparrow}$ certificate-history pair and diagonal obstruction surface of chapter 491, while restricting the read to the finite trace rows A, T, D, E, H, C, P, N.

Definition 492.1 (HaltingDistinctionTrace carrier)**.** *A* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, T, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the admitted certificate-history input row, naming the program certificate and the history at which its visible behavior is read;*
2. *T is the finite* Cont *trace row, listing the displayed marks and histories available before any halting-facing distinction is asserted;*
3. *D is the distinction classifier row, reading only whether the trace exposes the displayed termination mark or remains inside the finite open trace;*
4. *E is the diagonal self-application row, asking the classifier in D about the same certificate-history row named by A;*
5. *H stores componentwise* hsame *transports for input, trace, distinction, diagonal, provenance, and local naming rows;*
6. *C stores the displayed* Cont *routes for replaying T, entering D, reaching E, and returning to the local ledger;*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, hsame, Pkg, *the distinction row, the diagonal row, and the surrounding* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinctionTrace}^{\uparrow}}$ *row.*

The carrier contains no total halting predicate, no global truth predicate, no divergence detector, no host evaluator, and no classifier outside the displayed rows of R.

Lean4 ✓ BEDC.Derived.HaltingDistinctionTraceUp.HaltingDistinctionTraceTasteGate single carrier alignment

Theorem 492.2 (HaltingDistinctionTrace diagonal obstruction package)**.** *For an accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *carrier* $R = (A, T, D, E, H, C, P, N)$*, every certificate-level diagonal read of the halting-as-distinction obstruction factors through the finite route*

$$A \longmapsto T \longmapsto D \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route may expose that the classifier row D *is queried at its own certificate-history input through* E*, but it never exports a total halting predicate, a global truth predicate, a divergence detector, or a host-level evaluator. Thus the fixed-point obstruction is realized as finite* Cont*,* Ext*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Lean4 ✓ BEDC.Derived.HaltingDistinctionTraceUp.HaltingDistinctionTraceDiagonalObstructionPackage

Proof. Project the accepted carrier of Definition 492.1. The only rows that can name the input, the visible execution segment, the distinction classifier, and the diagonal query are A, T, D, E.

Replay of the trace is controlled by the Cont routes in C. These routes inspect the displayed marks and histories in T before the distinction row D is read, so a consumer cannot replace the finite trace by an ambient evaluator.

The diagonal point is the row E. Componentwise transport is supplied by H, while P records provenance and N records the local naming certificate. These rows can identify the self-application site, but they do not add a truth predicate or a total decision principle.

Therefore every diagonal read remains inside the displayed route through A, T, D, E, H, C, P, N. The obstruction is the finite certificate-level self-query itself, carried by Cont, Ext, Pkg, and $\mathsf{NameCert}$ data rather than by any global halting classifier. □

Theorem 492.3 (HaltingDistinctionTrace diagonal readback)**.** *For an accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *carrier* $R = (A, T, D, E, H, C, P, N)$*, the diagonal row* E *can be read only after the finite trace row* T*, the distinction row* D*, and the continuation route* C *have been displayed. Thus the packet exposes a finite certificate-level self-query, not a direct global halting classifier.*

Proof. Start from the carrier of Definition 492.1. The diagonal row E is a displayed coordinate of the same finite packet as A, T, D, H, C, P, N, and the carrier has no coordinate for a global classifier.

Apply the diagonal obstruction package of Definition 492.2. It factors every diagonal read through the finite route from the admitted input to the trace row, then to the distinction row, then to the diagonal self-application row.

The surrounding $\mathsf{HaltingDistinction}^{\uparrow}$ surface of chapter 491 supplies the distinction context, but the trace packet keeps the read inside T, D, E, C. Therefore a consumer reads the diagonal row only through the displayed finite trace route. □

Theorem 492.4 (HaltingDistinctionTrace provenance non-escape)**.** *For an accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *carrier* $R = (A, T, D, E, H, C, P, N)$*, the provenance row* P *and local* $\mathsf{NameCert}_{\mathsf{HaltingDistinctionTrace}^{\uparrow}}$ *row* N *package only the displayed rows* A, T, D, E, H, C*. They cannot turn the packet into a total halting predicate, divergence detector, host evaluator, or truth predicate.*

Proof. Read P and N in the carrier definition. Their provenance and naming scope is the finite packet A, T, D, E, H, C, P, N, with P naming the displayed BEDC objects and N naming the local certificate.

Use Definition 492.3. The only diagonal information available to P, N is the self-query route through the finite trace, distinction row, and continuation route.

Finally apply the non-export clause of Definition 492.2. Since the obstruction package itself exports no total halting predicate, global truth predicate, divergence detector, or host-level evaluator, the provenance and naming rows cannot export those objects either. □

Theorem 492.5 (HaltingDistinctionTrace scoped consumer factorization)**.** *For an accepted* HaltingDistinctionTrace *carrier* $R = (A, T, D, E, H, C, P, N)$*, every consumer read of the trace packet factors through the displayed rows*

$$A \to T \to D \to E \to H \to C \to P \to N.$$

The broader $\mathsf{HaltingDistinction}^{\uparrow}$ *certificate-history and diagonal obstruction context of chapter 491 is consumed only through the admitted input row* A*, the distinction classifier row* D*, and the diagonal self-application row* E*; it does not add a total halting predicate, truth predicate, divergence detector, or host evaluator.*

Proof. Project the carrier of Definition 492.1. The rows A, T, D, E, H, C, P, N are the complete finite packet visible to a consumer.

Use diagonal readback and provenance non-escape. They place every diagonal self-query behind the finite trace row T, the distinction row D, and the continuation route C, and they keep provenance and local naming inside the displayed packet.

Now insert the surrounding $\mathsf{HaltingDistinction}^{\uparrow}$ context through chapter 491. That context supplies the certificate-history and distinction surface read by A, D, E, but the trace packet exposes it only through those displayed rows. Hence every consumer read factors through A, T, D, E, H, C, P, N and cannot import global halting content. □

Theorem 492.6 (HaltingDistinctionTrace scoped transport non-escape)**.** *Let* $R = (A, T, D, E, H, C, P, N)$ *and* $R' = (A', T', D', E', H', C', P', N')$ *be accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *carriers. If the componentwise* hsame *rows in* H *and* H' *identify the admitted input, finite trace, distinction classifier, diagonal self-application, replay, provenance, and local naming rows, then transport preserves the route from trace to distinction to diagonal self-query. The transported packet still exports no total halting predicate, truth predicate, divergence detector, or host evaluation row.*

Proof. Read the componentwise transport rows. Their scope is exactly the displayed coordinates A, T, D, E, C, P, N of the two accepted trace packets, with no coordinate for an external evaluator or global classifier.

Apply scoped consumer factorization to both packets. Each consumer route first reads the admitted input and finite trace, then the distinction classifier and diagonal self-application, and finally the transport, replay, provenance, and local naming rows.

The hsame rows preserve those coordinates component by component. Therefore the transported read has the same trace, distinction, and diagonal route as the original packet. Since neither packet contains total halting, truth, divergence, or host-evaluation coordinates, transport cannot export them. □

Theorem 492.7 (HaltingDistinctionTrace bridge readback)**.** *For any bridge request addressed to* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$*, the displayed tuple*

$$R = (A, T, D, E, H, C, P, N)$$

is the fixed trace readback surface. The request may read the admitted certificate-history input, finite continuation trace, distinction classifier, diagonal self-application, componentwise transports,

displayed continuation routes, package provenance, and local naming row, but it cannot read a total halting predicate, divergence detector, truth predicate, or host evaluator.

Proof. Project the carrier of Definition 492.1. The only rows exposed by the packet are A, T, D, E, H, C, P, N.

Apply Definition 492.2 and Definition 492.3. They place the diagonal self-query behind the finite trace and distinction rows, so bridge readback reaches the obstruction only through the displayed tuple.

Use Definition 492.5 and Definition 492.6. These rows keep every consumer and transported read inside the same finite trace packet, with no coordinate for a total halting predicate or host evaluator. □

Theorem 492.8 (HaltingDistinctionTrace obstruction handoff)**.** *Every* $\mathsf{HaltingObstruction}^{\uparrow}$ *consumer of an accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *packet can read only the finite trace row* T*, distinction classifier row* D*, and diagonal self-query row* E*, together with the attached input, transport, route, provenance, and naming rows. The handoff carries the finite obstruction surface, not a total halting predicate or external evaluation oracle.*

Proof. Start with Definition 492.7. Any bridge-facing request reads the fixed tuple A, T, D, E, H, C, P, N.

The obstruction part of that tuple is T, D, E. By Definition 492.2, the diagonal obstruction is the finite route from the admitted input through the trace and distinction rows to the self-query row.

Apply Definition 492.4 and Definition 492.5. Provenance, naming, and consumer access remain attached to the displayed packet, so a $\mathsf{HaltingObstruction}^{\uparrow}$ consumer receives only the finite trace, distinction, and diagonal self-query rows with their recorded routes. □

Theorem 492.9 (HaltingDistinctionTrace mature boundary readback)**.** *Every mature consumer read of an accepted* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ *packet factors through the displayed rows*

$$A \longmapsto T \longmapsto D \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The read combines the diagonal obstruction package, bridge readback, and obstruction handoff, and still receives no total halting predicate, global truth predicate, divergence detector, or host evaluator.

Proof. Begin with Definition 492.2. It identifies the diagonal obstruction as the finite route through the admitted input, trace, distinction classifier, self-application row, transport, replay, provenance, and local naming rows.

Apply Definition 492.7. A bridge-facing mature request reads the same tuple A, T, D, E, H, C, P, N and no external halting coordinate.

Finally use Definition 492.8. The obstruction consumer receives only the finite trace, distinction, and diagonal self-query with their attached structural rows, so the mature readback boundary contains no total predicate, truth predicate, divergence detector, or host evaluator. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingDistinctionTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{HaltingDistinctionTrace}^{\uparrow}$ packet is generated from an admitted certificate-history input row, a finite continuation trace row, a distinction classifier row, a diagonal self-application row, componentwise transports, displayed continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaltingDistinctionTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 492.1, Definition 492.2, Definition 492.3, Definition 492.4, Definition 492.5, Definition 492.6, Definition 492.7, Definition 492.8, and Definition 492.9 bind the finite trace carrier, diagonal readback route, scoped consumer factorization, componentwise transport, provenance non-escape boundary, bridge readback surface, obstruction handoff, and mature boundary readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaltingDistinctionTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.HaltingDistinctionTraceUp.HaltingDistinctionTraceDiagonalObstructionPackage`
Bridge status: paperBridge
Origin: human
Not claimed: This chapter does not close a total halting predicate, a global truth predicate, divergence detection, host evaluation, or checked bridge certificate.
Upgrade path: Bridged closure requires a standard bridge for the displayed trace packet while preserving the independent formal-status axis.

Chapter 493

A Concrete Naming Certificate for $\mathsf{DecidableRefutation}^{\uparrow}$

The $\mathsf{DecidableRefutation}^{\uparrow}$ surface turns the permitted and forbidden contradiction boundary of chapter 9458 into a finite naming packet. It records the proposition row being inspected, the constructive refutation witness, the optional decidability route, the exclusion bits for forbidden schemas, and the provenance rows through which a consumer may read the packet. It consumes the permitted-route carrier of chapter 501 and adds the optional same-row decision coordinate that governs when double-negation consumption is allowed. The surface does not certify classical excluded middle, host truth, or a global decision procedure over naming certificates.

The permitted refutation witness row is the local consumer of chapter 501, while the exclusion bits are read through chapter 500. $\mathsf{DecidableRefutation}^{\uparrow}$ adds an optional decidability row and a truth-branch exclusion ledger over those sibling refutation surfaces, rather than replacing either the constructive route or the boundary packet.

Definition 493.1 (DecidableRefutation carrier)**.** *A* $\mathsf{DecidableRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, F, D, X, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition row being inspected, represented as a certificate-visible predicate read over the current finite history;*
2. *F is a permitted refutation witness, namely a displayed route from an assumed witness of A to $\bot$ through* Cont*,* hsame*,* msame*, and the local certificate rows;*
3. *D is an optional decidability row for A, present only when the packet carries a constructive branch witness for the same proposition row;*
4. *X records the exclusion bits for the forbidden schemas of Definition 9458.5: unwitnessed existential extraction, double-negation elimination without D, and excluded-middle branching over the carrier of all naming certificates;*
5. *H records componentwise* hsame *transports for A, F, D, X, package rows, and local naming rows;*
6. *C records the displayed* Cont *routes for reading the proposition row, replaying the permitted refutation, consuming the optional decidability row, and checking the forbidden-schema exclusions;*

7. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *the local proof rows, and the surrounding* NameCert *rows;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{DecidableRefutation}^{\uparrow}}$ *ledger.*

The carrier contains no host truth predicate, no global excluded-middle oracle, no selected witness for an existential obtained by contradiction, and no branch over the carrier of all naming certificates.

Lean4 ✓ `BEDC.Derived.DecidableRefutationUp.DecidableRefutationTasteGate single carrier alignment`

Definition 493.2 (DecidableRefutation classifier)**.** *Two accepted* $\mathsf{DecidableRefutation}^{\uparrow}$ *carriers*

$$R = (A, F, D, X, H, C, P, N) \quad \textit{and} \quad R' = (A', F', D', X', H', C', P', N')$$

are classified together exactly when their visible proposition rows are matched by the displayed hsame *transports, their permitted refutation witnesses replay through the displayed* Cont *routes, their optional decidability rows have the same displayed presence shape, and their forbidden-schema exclusion bits agree componentwise.*

The classifier compares only A, F, D, X, H, C, P, N and $A', F', D', X', H', C', P', N'$. It has no coordinate for ambient truth, no host-level proof search, no quotient over proofs, and no classifier over the carrier of all naming certificates.

Theorem 493.3 (DecidableRefutation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DecidableRefutation}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the finite packet $R = (A, F, D, X, H, C, P, N)$ of Definition 493.1;*

2. *classifier soundness for Definition 493.2, comparing only the displayed proposition, refutation, optional decidability, exclusion, transport, continuation, package, and naming rows;*

3. *permitted-negation transport, saying that a refutation witness F remains a proof of $\neg A$ under the stored* hsame *transports and displayed* Cont *routes;*

4. *decidable double-negation route, saying that $\neg\neg A$ may be consumed only when the row D displays a constructive decision for the same proposition row A;*

5. *non-escape from certificate-indexed truth branches, saying that the packet cannot export excluded middle over all naming certificates, unwitnessed existential extraction, or double-negation elimination without the displayed decidability row.*

Lean4 ✓ `BEDC.Derived.DecidableRefutationUp.DecidableRefutationUp taste gate boundary`

Proof. Carrier admission is projection from Definition 493.1. The only available rows are the proposition row, permitted refutation witness, optional decidability row, exclusion ledger, transport rows, continuation routes, package provenance, and local naming ledger.

Classifier soundness follows from Definition 493.2. A comparison replays only the displayed proposition-row match, the refutation route, the optional decidability shape, and the exclusion bits through the stored hsame and Cont data.

For permitted-negation transport, read the refutation witness F as a route from an assumed witness of A to $\bot$. Transporting A and F along H and replaying the route in C yields the same certificate-level negation, because no step leaves the displayed ground-loop rows.

For the decidable double-negation route, the row D is the only coordinate that supplies a constructive branch for A. If D is absent, the exclusion ledger X blocks the double-negation

schema named in Definition 9458.5; if D is present, the branch is consumed inside the same finite packet.

Finally, the non-escape clause is exactly the absence of a certificate-indexed truth branch coordinate. The packet contains no row classifying all naming certificates, no witness selector for existential-by-contradiction, and no host truth predicate, so every consumer remains inside A, F, D, X, H, C, P, N. □

Theorem 493.4 (DecidableRefutation classifier transport)**.** *Let $R = (A, F, D, X, H, C, P, N)$ and $R' = (A', F', D', X', H', C', P', N')$ be accepted* $\mathsf{DecidableRefutation}^{\uparrow}$ *packets. If the classifier of Definition 493.2 relates R and R', then every transported public read is one of the proposition rows A, A', the refutation rows F, F', the optional decidability rows D, D', the exclusion rows X, X', the displayed* hsame *rows H, H', the* Cont *routes C, C', the* Pkg *rows P, P', or the local* $\mathsf{NameCert}$ *rows N, N'.*

No transported read can introduce a parameter-transport theorem, an ambient truth row, a global decision procedure, or a classifier over all naming certificates.

Proof. Start with the checked TasteGate single-carrier row attached to Definition 493.1. It says the accepted object is the one finite carrier surface $R = (A, F, D, X, H, C, P, N)$, so a classifier comparison has no second hidden carrier from which to read extra rows.

Now unfold the classifier in Definition 493.2. The comparison names only the displayed proposition, refutation, optional decidability, exclusion, transport, continuation, package, and NameCert coordinates on the two packets.

The transported public read is therefore a projection through H, H' and C, C' followed by a read of one of A, F, D, X, P, N or A', F', D', X', P', N'. Since the carrier has no coordinate for parameter families, host truth, global decision, or a classifier over all certificates, none of those objects can be exported. □

Theorem 493.5 (DecidableRefutation truth-branch exclusion)**.** *No consumer of an accepted* $\mathsf{DecidableRefutation}^{\uparrow}$ *packet can extract either excluded middle over the carrier of all NameCerts or an existential-by-contradiction witness from the packet. The only branch row available to the consumer is the optional decidability coordinate D for the same displayed proposition row A, and the exclusion coordinate X blocks every branch not licensed by that row.*

Proof. Read the packet through the checked TasteGate single-carrier row and the carrier fields of Definition 493.1. The carrier lists A, F, D, X, H, C, P, N and no coordinate for a truth predicate over all NameCerts.

If D is present, it is a decision row for the same proposition row A. It does not range over arbitrary naming certificates and does not choose witnesses for unrelated existential statements.

If D is absent, the exclusion row X records the forbidden schemas from Definition 9458.5. Thus excluded-middle branching over all certificates and existential-by-contradiction witness extraction both stop at X rather than becoming exported data. □

Theorem 493.6 (DecidableRefutation obligation package)**.** *The* $\mathsf{DecidableRefutation}^{\uparrow}$ *paper obligation package consists of carrier admission for Definition 493.1, classifier transport by Definition 493.4, permitted-negation transport from Definition 493.3, the decidable double-negation route named by the same obligation surface, and truth-branch exclusion by Definition 493.5.*

This package is single-carrier: it cites the checked TasteGate row for the displayed packet $R = (A, F, D, X, H, C, P, N)$ and does not add a parameter-transport theorem.

Proof. Carrier admission is the tuple projection from Definition 493.1. The checked TasteGate single-carrier row fixes the admissible surface to the same eight coordinates, so the package begins with a single finite packet.

Classifier transport is supplied by Definition 493.4. It keeps all public reads inside the proposition, refutation, optional decidability, exclusion, hsame, Cont, Pkg, and NameCert rows.

The permitted-negation and decidable double-negation routes are the third and fourth obligations of Definition 493.3. They allow a displayed refutation route and a displayed decision row for A while refusing a free-standing truth branch.

Finally, Definition 493.5 supplies the boundary clause. Repacking these rows gives the obligation package without parameter transport, host truth, certificate-wide excluded middle, or witness selection by contradiction. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecidableRefutation}\uparrow}$.

Constructive story: A $\mathsf{DecidableRefutation}^{\uparrow}$ packet is generated from a proposition row, a permitted refutation witness, an optional decidability row, forbidden-schema exclusion bits, componentwise transports, displayed continuation routes, package provenance, and a local naming ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecidableRefutation}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 493.1, Definition 493.2, Definition 493.3, Definition 493.4, Definition 493.5, and Definition 493.6 bind the carrier, classifier, NameCert obligations, classifier transport, truth-branch exclusion, and paper obligation package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecidableRefutation}\uparrow}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.DecidableRefutationUp.taste gate`

Lean target: `BEDC.Derived.DecidableRefutationUp.DecidableRefutationUp\_taste\_gate\_boundary`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close classical excluded middle, global truth over naming certificates, existential-by-contradiction witness selection, or a checked Lean certificate.

Upgrade path: Public closure requires a public single-carrier export theorem over the displayed packet.

Chapter 494

A Concrete Naming Certificate for $\mathsf{SelectorLedger}^{\uparrow}$

The $\mathsf{SelectorLedger}^{\uparrow}$ interface packages the Empty-fable trace surface as a finite naming certificate. It records a generated history, the trace read from its constructor spine, the displayed selector-step witnesses that justify each realized branch, and the finite transport and continuation rows that make the ledger consumable by later NameCert packets. The packet does not assert a global branch selector, a truth oracle, a host list quotient, or a choice principle.

Definition 494.1 (Selector ledger carrier)**.** *A* $\mathsf{SelectorLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (h, T, S, H, C, P, N)$$

with the following displayed coordinates:

1. *h is the generated* Hist *row whose constructor spine is being read;*
2. *T is the finite* Trace *row for h, read only from* e, E0, E1, *and* Ext *constructors;*
3. *S is the finite list of* SelStep *witnesses, one for each non-empty trace edge, each exposing the enacted mark, the* Ext *step, and the local branch certificate;*
4. *H stores the componentwise* hsame *transports for the history row, trace row, and selector-step witnesses;*
5. *C stores the* Cont *routes that read the generated history, each trace edge, and each selector witness;*
6. *P is the* Pkg *provenance over* BHist, Hist, Ext, Trace, SelStep, hsame, Cont, Pkg, *and* NameCert*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$ *row.*

The carrier is this finite packet. It contains no global chooser, no hidden branch function, no host equality of traces, and no oracle row.

Lean4 ✓ `BEDC.Derived.SelectorLedgerUp.SelectorLedgerCarrier public trace readback`

Theorem 494.2 (SelectorLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$ *surface for the carrier of Definition 494.1 consists exactly of carrier admission for L, trace admission for T, selector-step admission for every row of S, transport of the history, trace, and selector witnesses by the displayed* hsame *rows in H, continuation reads through C, package provenance through P, and the local naming row N.*

Every obligation is read from BHist*,* Ext*,* Trace*,* SelStep*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The certificate exports no branch selector beyond the displayed selector-step witnesses.*

Proof. Carrier admission is projection from the finite packet L. The history row, trace row, selector-step list, transport rows, continuation routes, package provenance, and local naming row are all displayed coordinates.

Trace admission reads T from the constructor spine of h. Each non-empty edge in that trace is matched with one selector-step witness in S, and each such witness exposes the enacted mark together with the corresponding Ext row.

The transport and continuation obligations are exactly the rows H and C. They move the displayed history, trace, and selector witnesses componentwise while preserving the same package provenance and local naming row. Since the carrier has no further selector coordinate, the certificate has no hidden choice field to export. □

Lean4 ✓ `BEDC.Derived.SelectorLedgerUp.SelectorLedgerNameCertObligations`

Theorem 494.3 (SelectorLedger trace soundness)**.** *For every accepted* $\mathsf{SelectorLedger}^{\uparrow}$ *carrier* $L = (h, T, S, H, C, P, N)$*, the exported ledger trace* T *is sound for the generated history* h*: each mark in* T *is justified by a displayed* SelStep *row in* S*, each such selector row contains an* Ext *step from the previous history to the next history, and the final history reached by the finite chain is* hsame*-equivalent to* h *through the rows recorded in* H *and read through* C*.*

Thus every downstream consumer of the ledger receives a finite Empty-fable trace with explicit branch witnesses, not a global selector or external choice function.

Lean4 ✓ `BEDC.Derived.SelectorLedgerUp.SelectorLedgerTasteGate single carrier alignment`

Proof. Start with the trace coordinate T. By the carrier definition, T is the trace row associated with the generated history h, and no other trace coordinate is present.

Proceed along the finite trace. For the empty trace, the history row is the empty boundary row and the selector list is empty. For a non-empty trace, the next selector-step row in S exposes a mark, an Ext step, and a local branch certificate, so the next history in the chain is displayed.

After all trace entries are consumed, the stored hsame rows in H compare the reached history with h, and the continuation rows in C expose the read to downstream consumers. The argument never introduces a chooser beyond the selector witness already stored for that edge. □

Theorem 494.4 (SelectorLedger no hidden choice)**.** *Every consumer read of an accepted* $\mathsf{SelectorLedger}^{\uparrow}$ *carrier factors through the displayed tuple* h, T, S, H, C, P, N*. In particular, branch information can be read only as a finite* SelStep *witness whose mark and* Ext *row are visible, and trace information can be read only through the finite* Trace *row transported by* hsame *and* Cont*.*

The packet therefore exposes the Empty-fable selector ledger as a finite $\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$ *surface and exports no global choice function, hidden chooser, host trace quotient, or truth oracle.*

Lean4 ✓ `BEDC.Derived.SelectorLedgerUp.SelectorLedgerNoHiddenChoice`

Proof. Inspect the carrier inventory. The only branch-facing coordinate is the finite selector-step list S, and the only sequence-facing coordinate is the trace row T.

A branch read must therefore project a row of S. That projection exposes the mark, the Ext step, and the local branch certificate, so the selector is a visible witness rather than a hidden function.

A trace read must project T and then transport it through H and C. Package provenance and the naming row record that this is the whole public surface. Since no coordinate names a global chooser, host trace quotient, or oracle, none can be exported. □

Theorem 494.5 (SelectorLedger inscription-event consumer route)**.** *Every accepted* $\mathsf{SelectorLedger}^{\uparrow}$ *consumer route that is used as public record data factors as an* $\mathsf{InscriptionEvent}^{\uparrow}$*-style consumer route through displayed history, trace, selector-step witness, accepted continuation, intended consumer, and local* GAP *and provenance rows, in the sense of chapter 519.*

Thus the public record view of a selector ledger is not a standalone chooser interface. It is finite inscription-event route data whose readable branch content is already exposed by the displayed ledger packet.

Lean4 ✓ `BEDC.Derived.SelectorLedgerUp.SelectorLedgerInscriptionEventConsumerRoute`

Proof. Write the accepted selector ledger as $L = (h, T, S, H, C, P, N)$. By Definition 494.4, any consumer read of L can reach branch information only by projecting a displayed SelStep row from S, and can reach trace information only through the finite trace row T transported by the displayed hsame and Cont rows in H and C.

Compare this projected packet with the inscription-event carrier of chapter 519. The displayed history row supplies the source record state, the trace and selector-step row supply the named route witness, the accepted continuation row supplies the checkable public route, the intended consumer is the consumer of the selector-ledger read, and the local GAP, package, and naming rows are the public ledger/provenance data.

The factorization is therefore finite and public: it uses only h, T, S, H, C, P, N and the inscription-event source/name/route/check/consumer/GAP packet shape. No step introduces a global branch chooser, a hidden host trace quotient, or a record mutation outside the displayed carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SelectorLedger}^{\uparrow}$ packet is generated from a displayed Hist row, its finite Trace row, explicit SelStep witnesses for realized branch edges, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public object surface is exactly the displayed selector carrier, trace soundness, selector-step exposure, inscription-event consumer route, componentwise transport, continuation replay, and no-hidden-choice rows over BHist, Hist, Ext, Trace, SelStep, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SelectorLedger}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge status, global branch selection, host trace quotient, truth oracle, or any choice principle.

Upgrade path: No paper-side theory upgrade remains for the displayed selector ledger; Lean verification and bridge checking remain outside this chapter.

Chapter 495

A Concrete Naming Certificate for $\mathsf{HaltingDistinctionLimit}^{\uparrow}$

The $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ surface packages the halting fixed-point obstruction as finite BEDC data. It names the program certificate row, the input history row, the visible continuation trace, the diagonal obstruction row, the transport and route ledgers, the package provenance, and the local naming certificate through which the obstruction is read. It consumes the finite diagonal fixed-point package of chapter 496: program row, input history, self-reference row, fixed Cont row, diagonal policy, and hsame/Cont/Pkg/NameCert provenance are read before the non-decision boundary is exposed. It does not decide halting and does not introduce a halting oracle.

Definition 495.1 (Halting distinction limit carrier)**.** *A* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, h, T, D, H, C, K, N)$$

with the following displayed rows:

1. *P is the program and certificate row, naming the candidate program certificate together with the local* $\mathsf{NameCert}_P$ *row by which the classifier is addressed;*
2. *h is the input history row whose continuation behaviour is being inspected;*
3. *T is the finite trace row recording the displayed* Cont_P *segment from h through the marks and histories visible to the packet;*
4. *D is the diagonal obstruction row, naming the self-application read in which the classifier for (P, h) is queried at the same program-history position;*
5. *H records the componentwise* hsame *transports for program rows, input histories, trace rows, diagonal rows, provenance rows, and local naming rows;*
6. *C records the displayed* Cont *routes that read the input history, replay the finite trace, enter the diagonal obstruction row, and return the read to the local ledger;*
7. *K is the* Pkg *provenance over* BHist, Ext, Cont, hsame, *classifier transport, diagonal rows, and the surrounding certificate rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingDistinctionLimit}^{\uparrow}}$ *ledger.*

The packet has no coordinate for a total halting predicate, divergence oracle, global truth predicate, or host evaluator outside P, h, T, D, H, C, K, N.

Lean4 (definition) `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitCarrier`

Definition 495.2 (Halting distinction limit classifier)**.** *The* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ *classifier compares two accepted packets only by the displayed rows of Definition 495.1. It relates*

$$L = (P, h, T, D, H, C, K, N)$$

and

$$L' = (P', h', T', D', H', C', K', N')$$

when the program-certificate rows, input histories, trace rows, diagonal obstruction rows, transport ledgers, continuation routes, package provenance, and local naming rows are related by the recorded componentwise hsame *and* Cont *data. It does not compare hidden machine states, meta-level evaluator results, or truth values for unrecorded program-history pairs.*

Lean4 (definition) `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitClassifier`

Theorem 495.3 (Halting fixed-point boundary)**.** *For every accepted* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ *packet* $L = (P, h, T, D, H, C, K, N)$*, any public read of the diagonal obstruction reaches the fixed-point boundary only through the displayed route from the program-certificate row and input history row, through the finite trace row* T*, into the diagonal row* D*, and back through the transport, continuation, package, and local naming ledgers.*

The proof source is the BEDC reflection theorem `BEDC.Reflection.halting_as_form_of_distinction_fixe` *the self-application obstruction is a boundary on total classifier extension, not a construction of a halting decision procedure.*

Proof. Project the packet through Definition 495.1. The only rows that can name the tested computation are P, h, T, and D, with H, C, K, and N recording the allowed transports, routes, provenance, and local naming.

Use the classifier of Definition 495.2. A transported read can compare only the displayed program-certificate, input-history, trace, diagonal, route, and provenance rows. No hidden evaluator coordinate is available.

At the diagonal row D, apply the fixed-point obstruction expressed by `BEDC.Reflection.halting_as_form_of` The obstruction identifies the boundary where a classifier cannot be extended into a total halting decision over its own input surface.

Therefore the packet realizes the fixed-point boundary as a finite naming certificate. It records the obstruction route and excludes any export of a total halting predicate or divergence oracle. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitFixedPointBoundary`

Theorem 495.4 (Halting distinction limit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingDistinctionLimit}^{\uparrow}}$ *surface has five finite obligations:*

1. *carrier admission for packets* $L = (P, h, T, D, H, C, K, N)$ *of Definition 495.1;*
2. *classifier transport through the displayed* hsame *rows and* Cont *routes of Definition 495.2;*
3. *fixed-point boundary exposure through the diagonal row* D *and Definition 495.3;*
4. *non-decision ledger discipline, saying that* T *records only a finite trace and* D *records only the diagonal obstruction route;*
5. *downstream non-escape, saying that consumers cannot extract a total halting predicate, divergence oracle, global truth predicate, host evaluator, or unrecorded classifier.*

Proof. Carrier admission is the finite packet of Definition 495.1. Each obligation row is one of the displayed coordinates P, h, T, D, H, C, K, N.

Classifier transport is supplied by Definition 495.2. The only comparisons are componentwise comparisons of named program-certificate rows, histories, traces, diagonal rows, routes, provenance, and local naming ledgers.

The fixed-point boundary is Definition 495.3. It routes the self-application read to D and stops at the obstruction boundary.

The non-decision ledger follows because the trace row is finite and the diagonal row is a boundary row. Neither row supplies a total truth predicate or a divergence oracle.

For downstream consumers, inspect the carrier again. Every exported read is one of P, h, T, D, H, C, K, N, so no host evaluator, unrecorded classifier, or global halting decision can leave the packet. □

Lean4 ✓ `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitNameCertObligations`

Theorem 495.5 (Halting distinction limit sibling boundary)**.** *Every diagonal obstruction read in an accepted* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ *packet factors through the obstruction classifier boundary supplied by chapter 490 and the fixed-point vision boundary of chapter 9368. The exported surface remains only*

$$P, h, T, D, H, C, K, N,$$

and it does not become a halting decision procedure.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitPacket sibling boundary`

Proof. Project the local carrier of Definition 495.1. The only obstruction-facing rows are the program row P, input history h, finite trace T, diagonal row D, and the structural rows H, C, K, N.

The classifier of Definition 495.2 permits comparison only through those displayed rows. Thus a diagonal read cannot bypass the local packet and cannot add a total halting predicate.

Apply Definition 495.3 and Definition 495.4. They route the self-application read through D and keep the trace finite before any downstream consumer can see the obstruction.

The sibling packet of chapter 490 is exactly the downstream obstruction classifier boundary, and chapter 9368 names the fixed-point limit being exposed. Therefore the public read factors through that sibling boundary while exporting only P, h, T, D, H, C, K, N. □

Theorem 495.6 (Halting distinction limit public boundary)**.** *The public* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$ *interface exports only the finite fixed-point obstruction route*

$$P, h, T, D, H, C, K, N.$$

It names the program-certificate row, input history row, finite trace row, diagonal obstruction row, componentwise transport, displayed continuation routes, package provenance, and local naming certificate. It does not export a total halting predicate, divergence oracle, global truth predicate, host evaluator, or unrecorded classifier.

Lean4 ✓ `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitPublicBoundary`

Proof. Project the carrier of Definition 495.1. Its public coordinates are exactly P, h, T, D, H, C, K, N.

Apply Definition 495.3. The diagonal read reaches only the fixed-point obstruction boundary through those displayed rows.

Apply Definition 495.4 and Definition 495.5. The local certificate and sibling boundary preserve the same finite route, so no total halting predicate, divergence oracle, global truth predicate, host evaluator, or unrecorded classifier is available to a public consumer. □

Theorem 495.7 (Halting distinction limit bridge boundary)**.** *The public* HaltingDistinctionLimit$^{\uparrow}$ *route*

$$P, h, T, D, H, C, K, N$$

can bridge to the HaltingObstruction$^{\uparrow}$ *sibling only as a finite obstruction-classifier boundary. The bridge target may read the displayed program-certificate row, input history row, finite trace row, diagonal obstruction row, transports, routes, provenance, and local naming, but it cannot export a total halting predicate, divergence oracle, global truth predicate, host evaluator, or unrecorded classifier.*

Lean4 ✓ `BEDC.Derived.HaltingDistinctionLimitUp.HaltingDistinctionLimitCarrier bridge boundary`

Proof. Project the carrier of Definition 495.1. The only public route is the finite packet P, h, T, D, H, C, K, N.

Apply the fixed-point boundary Definition 495.3. The diagonal read reaches the obstruction point through D and stops at the boundary on total classifier extension.

Use the NameCert obligations of Definition 495.4. They keep the trace finite, keep the diagonal row as an obstruction row, and require all downstream reads to remain inside the displayed carrier.

Now apply the sibling boundary Definition 495.5 and the public boundary Definition 495.6. The HaltingObstruction$^{\uparrow}$ sibling is therefore a finite obstruction-classifier interface for the same route, not a halting oracle. □

Closure status for NameCert$_{\text{HaltingDistinctionLimit}^{\uparrow}}$ **·**

Constructive story: A HaltingDistinctionLimit$^{\uparrow}$ packet is generated from a program-certificate row, an input history row, a finite Cont$_P$ trace row, a diagonal obstruction row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert ledger.

Theory closure: TheoryGate(NameCert$_{\text{HaltingDistinctionLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 495.1, Definition 495.2, Definition 495.3, Definition 495.4, Definition 495.5, Definition 495.6, and Definition 495.7 jointly exhibit the mature paper-side HaltingDistinctionLimit$^{\uparrow}$ treatment for the finite BHist, Ext, Cont, hsame, classifier transport, diagonal rows, Pkg, and NameCert obstruction route.

Formal verification: FormalStatus(NameCert$_{\text{HaltingDistinctionLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a total halting predicate, a divergence oracle, a global truth predicate, a host evaluator, or a checked Lean certificate.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for classifier transport, fixed-point boundary exposure, non-decision ledger discipline, and downstream non-escape.

Chapter 496

A Concrete Naming Certificate for $\mathsf{HaltingDiagonal}^{\uparrow}$

The $\mathsf{HaltingDiagonal}^{\uparrow}$ packet turns the halting fixed-point obstruction of Definition 9368.3 into a finite naming-certificate frontier. It records program-certificate data, one input history, the self-reference row, the fixed continuation row, the diagonal-divergence policy row, displayed transports, local routes, package provenance, and local naming obligations. It does not add global halting decidability, universal-machine semantics, a halting oracle, or a total classifier over all certificates.

Definition 496.1 (HaltingDiagonal packet)**.** *A* $\mathsf{HaltingDiagonal}^{\uparrow}$ *packet is a finite row*

$$Z = (P, h, E, F, D, H, C, K, N)$$

with the following displayed coordinates:

1. *P is the program certificate row whose local* $\mathsf{NameCert}_P$ *supplies the* Cont_P *and* Ext_P *surface being inspected.*
2. *h is the input history row in* BHist*.*
3. *E is the* Ext *self-reference row producing the history used by the diagonal read.*
4. *F is the fixed* Cont *row witnessing the empty-fuel continuation from the self-referential history back to itself.*
5. *D is the diagonal-divergence policy row: if the local halting classifier identifies the fixed-point history with its own continuation readback, the policy records the corresponding divergence claim.*
6. *H stores the componentwise* hsame *transports for P, h, E, F, and D.*
7. *C stores the displayed* Cont *routes used by the program row, the fixed-point row, and the diagonal readback.*
8. *K is the finite* Pkg *provenance ledger tying the packet to the halting fixed-point interface.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingDiagonal}^{\uparrow}}$ *row for carrier admission, classifier stability, exactness, ledger, and non-decision boundary obligations.*

The packet is finite over $\mathsf{NameCert}$*,* BHist*,* Ext*,* Cont*,* hsame*, and* Pkg*. It has no coordinate for global halting decidability, no universal-machine semantics, and no oracle deciding termination for arbitrary programs.*

Theorem 496.2 (HaltingDiagonal fixed-point package)**.** *The rows $P, h, E, F, D, H, C, K, N$ of Definition 496.1 assemble the local* $\mathsf{HaltingDiagonal}^{\uparrow}$ *fixed-point package: the program row supplies the inspected certificate surface, E supplies self-reference, F supplies the fixed* Cont *row, D supplies the diagonal-divergence policy, H and C supply transport and continuation routes, and K, N supply package provenance and local naming obligations.*

Lean4 ✓ `BEDC.Derived.HaltingDiagonalUp.HaltingDiagonal fixed point package rows`

Proof. Project the finite packet. The rows P and h identify the certificate and input-history coordinates being inspected, while E identifies the self-reference row used by the diagonal construction.

The fixed-point row is F: it is the displayed Cont route from the self-referential history back to itself. The diagonal policy row is D: it records the local implication from classifier agreement at that fixed point to the divergence claim.

Finally, H transports the displayed rows, C records the continuation routes, and K, N record provenance and local naming obligations. Repacking these rows gives the finite fixed-point package without adding a global decision procedure. □

Theorem 496.3 (HaltingDiagonal non-decision boundary)**.** *Every consumer read of an accepted* $\mathsf{HaltingDiagonal}^{\uparrow}$ *packet factors through the finite rows $P, h, E, F, D, H, C, K, N$. The packet exposes the fixed-point obstruction boundary of Definition 9368.3; it does not decide halting for arbitrary* $\mathsf{NameCert}$ *rows, does not supply universal-machine semantics, and does not convert the diagonal-divergence policy into a total classifier.*

Proof. Carrier admission reads only the displayed packet rows. Thus a consumer may inspect the program-certificate row, the input history, the self-reference row, the fixed continuation, the diagonal policy, transports, routes, package provenance, and local certificate.

Apply the fixed-point package of Definition 496.2. It identifies exactly where the local obstruction lives: the route through E and F and the policy row D.

No row in the packet classifies all program-history pairs. Since no universal-machine coordinate or total halting classifier is present, consumer reads remain bounded by the local fixed-point obstruction. □

Theorem 496.4 (HaltingDiagonal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingDiagonal}^{\uparrow}}$ *row has five obligations: carrier admission for the finite packet of Definition 496.1, classifier stability under componentwise* hsame *transport of the displayed rows, exactness of the E and F fixed-point readback, ledger provenance through K, and the non-decision boundary of Definition 496.3.*

Proof. Carrier admission is the finite packet definition. It names every row available to the certificate.

Classifier stability is componentwise: transport P, h, E, F, D along H and route the transported continuations through C. Exactness is the statement that the self-reference row and fixed continuation row still read back as the same local diagonal configuration.

The ledger obligation is K, and the boundary obligation is Definition 496.3. Together these rows give the local naming package without adding a global halting predicate. □

Theorem 496.5 (HaltingDiagonal carrier exposure obligation)**.** *For every accepted* $\mathsf{HaltingDiagonal}^{\uparrow}$ *packet $Z = (P, h, E, F, D, H, C, K, N)$, the local carrier exposes exactly the finite rows $P, h, E, F, D, H, C, K, N$. Every consumer read of the diagonal obstruction must begin from those rows, and no extra coordinate supplies a global halting predicate, universal-machine semantics, or oracle access.*

Proof. Project the packet of Definition 496.1. Its carrier rows are precisely the program certificate P, input history h, self-reference row E, fixed continuation row F, diagonal-divergence policy D, transport rows H, continuation routes C, provenance ledger K, and local naming row N.

The carrier definition has no coordinate for a global halting predicate or universal-machine semantics. Therefore every consumer read is bounded by those displayed rows. □

Theorem 496.6 (HaltingDiagonal classifier stability obligation)**.** *Let $Z = (P, h, E, F, D, H, C, K, N)$ and $Z' = (P', h', E', F', D', H', C', K', N')$ be accepted* $\mathsf{HaltingDiagonal}^{\uparrow}$ *packets whose displayed program, history, self-reference, fixed-continuation, and diagonal-policy rows are related componentwise by the* hsame *transports recorded in H and H'. Then the local classifier read through N transports componentwise across those rows: the accepted carrier exposure, fixed-point readback, diagonal policy, continuation routes, and provenance ledger remain the same named obstruction under the transported packet.*

Proof. Use the componentwise transports recorded in H and H' for P, h, E, F, D. These transports place the program certificate, input history, self-reference row, fixed continuation row, and diagonal policy row on the same displayed naming surface.

Next read the continuation routes through C and C'. The routes expose the transported fixed-point readback and diagonal policy without adding another classifier coordinate.

Finally, package the transported rows through K, K' and the local naming rows N, N'. The result is classifier stability for the same finite diagonal obstruction, not a new decision procedure. □

Theorem 496.7 (HaltingDiagonal ledger-boundary obligation)**.** *The provenance ledger K of an accepted* $\mathsf{HaltingDiagonal}^{\uparrow}$ *packet records only the finite fixed-point obstruction generated by P, h, E, F, D, H, C, N. It does not promote the diagonal-divergence policy row D into a total halting classifier, and it does not permit a consumer to decide termination for arbitrary* NameCert *rows outside the displayed packet.*

Proof. Read K as the package provenance for the displayed packet. Its dependencies are the program row, input history, self-reference row, fixed continuation row, diagonal policy, transports, routes, and local naming certificate.

Apply Definition 496.3. That theorem identifies the consumer boundary: the packet exposes a local fixed-point obstruction but contains no universal classifier row.

Thus the ledger is a boundary ledger. It records the finite obstruction and prevents the diagonal policy from being read as a global halting decision. □

Theorem 496.8 (HaltingDiagonal to HaltingObstruction sibling route)**.** *Every* $\mathsf{HaltingDiagonal}^{\uparrow}$ *diagonal read that is consumed as a* $\mathsf{HaltingObstruction}^{\uparrow}$ *ledger must pass through the finite rows $P, h, E, F, D, H, C, K, N$ before entering the obstruction packet of chapter 490. The route therefore treats the diagonal packet as a sibling fixed-point source for the obstruction ledger, not as a global halting classifier.*

Proof. Start with the fixed-point package of Definition 496.2. It identifies the rows $P, h, E, F, D, H, C, K, N$ as the displayed diagonal source, including the self-reference row E, fixed continuation row F, and diagonal policy row D.

Next apply the ledger-boundary obligation of Definition 496.7. That boundary prevents K and D from being read as a total halting classifier; they remain finite ledger rows for the displayed packet.

The sibling chapter chapter 490 consumes exactly this kind of finite obstruction packet. Hence any obstruction-facing read factors through the diagonal rows first and only then enters the HaltingObstruction ledger surface. □

Theorem 496.9 (HaltingDiagonal scoped certificate)**.** *The source packet, fixed-point package, non-decision boundary, and NameCert obligations assemble one scoped* $\mathsf{HaltingDiagonal}^{\uparrow}$ *certificate over program certificate rows, input histories,* Ext *self-reference rows, fixed* Cont *rows, diagonal-divergence policy rows,* BHist, hsame, Cont, Pkg, *and* NameCert *provenance. The certificate excludes global halting decidability, universal-machine semantics, oracle access, and infinite searches through program space.*

Lean4 ✓ `BEDC.Derived.HaltingDiagonalUp.HaltingDiagonalTasteGate single carrier alignment`

Proof. Start with the packet definition of Definition 496.1. It fixes the finite carrier coordinates and the listed dependency surface.

Use Definition 496.2 to package the self-reference and fixed continuation rows as the local diagonal readback. Use Definition 496.4 to bind carrier, classifier, exactness, ledger, and boundary obligations into the local NameCert row.

The non-decision boundary of Definition 496.3 supplies the exclusion clauses. Therefore the certificate is scoped to the finite fixed-point obstruction and does not export a global halting decision mechanism. □

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingDiagonal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HaltingDiagonal}^{\uparrow}$ packet is generated from a program certificate row, an input history, an Ext self-reference row, a fixed Cont row, a diagonal-divergence policy row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaltingDiagonal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 496.1, Definition 496.2, Definition 496.3, Definition 496.4, Definition 496.9, and the finite obstruction bridge route of Definition 496.8 bind the fixed-point obstruction boundary over NameCert, Cont, Ext, hsame, Pkg, and local diagonal ledgers.

Lean target: `BEDC.Derived.HaltingDiagonalUp.HaltingDiagonalTasteGate\_single\_carrier\_alignment`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaltingDiagonal}^{\uparrow}}, \mathsf{axiomClean})$

Bridge status: none

Not claimed: This chapter does not close global halting decidability, universal-machine semantics, oracle access, total classification of all program-history pairs, or bridge checking.

Upgrade path: No further paper-theory closure upgrade remains for the finite fixed-point obstruction; Lean verification and bridge checking remain independent.

Chapter 497

A Concrete Naming Certificate for $\mathsf{DecidableRefutationBoundary}^{\uparrow}$

The $\mathsf{DecidableRefutationBoundary}^{\uparrow}$ surface realizes the contradiction boundary of chapter 9458 as a finite NameCert carrier. It records one closed assumption row, an optional decision row, the displayed route from that assumption to falsity, transport and continuation rows, package provenance, and the local naming row. The carrier admits refutation only as visible BEDC evidence and keeps the forbidden contradiction schemas outside the exported certificate surface.

$\mathsf{DecidableRefutationBoundary}^{\uparrow}$ consumes the visible falsity-route discipline of chapter 501 and the displayed witness-row discipline of chapter 503. Its optional decision row narrows those sibling surfaces to a single closed assumption row A and does not add excluded-middle branching.

Definition 497.1 (Decidable refutation boundary carrier)**.** *A* $\mathsf{DecidableRefutationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, D, F, H, C, P, N)$$

with the following displayed coordinates:

1. *A is a closed assumption row, read as the certificate-visible proposition whose attempted witness is being inspected;*
2. *D is an optional decidability witness for the same row A, present exactly when the packet carries a constructive decision route for that row;*
3. *F is the displayed falsity route, a finite path from an assumed witness of A to $\bot$ using only* msame, hsame, Cont, *and the local certificate rows;*
4. *H stores the componentwise* hsame *transports for A, D, F, package provenance, and the local naming row;*
5. *C stores the displayed* Cont *routes that read A, replay F, consume D when it is present, and return the result to the visible ledger;*
6. *P is the* Pkg *provenance over* BHist, msame, hsame, Cont, *the displayed refutation rows, optional decision rows, and the surrounding* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{DecidableRefutationBoundary}^{\uparrow}}$ *row.*

The packet has no coordinate for a witness selected from contradiction alone, no double-negation eliminator without the row D, no excluded-middle branch over all naming certificates, and no host truth predicate.

Lean4 ✓ `BEDC.Derived.DecidableRefutationBoundaryUp.DecidableRefutationBoundaryTasteGate single carrier alignment`

Theorem 497.2 (Decidable refutation boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DecidableRefutationBoundar}}$ *surface for the carrier* $R = (A, D, F, H, C, P, N)$ *consists of five obligations:*

1. *carrier admission exposes exactly the finite packet of Definition 497.1;*
2. *assumption closure keeps* A *as the single closed assumption row inspected by the packet and prevents consumers from replacing it by an ambient predicate;*
3. *permitted falsity replay transports the route* F *only through the stored* hsame *rows and the displayed* Cont *rows;*
4. *decidable readback consumes* $\neg\neg A$ *only when* D *displays a constructive decision for the same assumption row* A*;*
5. *boundary non-escape prevents export of unwitnessed existential extraction, undecidable double-negation elimination, excluded-middle branching over all naming certificates, and certificate-indexed truth branches.*

Thus the certificate surface is finite, local to A, D, F, H, C, P, N*, and separated from the forbidden patterns of Definition 9458.5.*

Proof. Carrier admission is projection from Definition 497.1. The only readable rows are the closed assumption row, optional decision row, falsity route, transport rows, continuation routes, package provenance, and local naming row.

Assumption closure follows because A is stored as a closed coordinate of the same packet. A consumer may transport it by the displayed hsame rows and read it by the displayed Cont rows, but no coordinate replaces it by a host predicate or by a predicate over all naming certificates.

Permitted falsity replay uses the route F. Transporting A and F along H and replaying the corresponding route in C still yields a displayed route from an assumed witness of the transported A to $\bot$.

For decidable readback, inspect D. If D is present, the readback is tied to the same row A and remains inside the packet. If D is absent, no row of R supplies the schema $\neg\neg A \Rightarrow A$.

Boundary non-escape is the remaining carrier inventory. The packet has no witness selector for existential-by-contradiction, no undecided double-negation eliminator, no branch over the carrier of all naming certificates, and no host truth predicate, so every consumer remains inside the displayed NameCert rows. □

Lean4 ✓ `BEDC.Derived.DecidableRefutationBoundaryUp.DecidableRefutationBoundary assumption falsity route unary closure`

Theorem 497.3 (Decidable refutation ground-loop exhaustion)**.** *Every accepted* $\mathsf{DecidableRefutationBoundary}^{\uparrow}$ *route stays inside the rows* msame, hsame, Cont, Pkg*, and* NameCert *displayed by* $R = (A, D, F, H, C, P, N)$*. It can export only the local falsity route for the closed assumption row* A *and, when* D *is present, the matching decidable readback for* A*. It cannot export an unwitnessed existential, an undecidable double-negation elimination, or a certificate-indexed truth branch.*

Proof. Project the accepted carrier. The proof route available to a consumer is F, and every movement of that route is accounted for by H and C.

The ground-loop rows are therefore exhausted by msame, hsame, Cont, the package row P, and the local naming row N. No additional classifier, oracle, quotient, or truth predicate is present in the packet.

When D is present, a consumer may use it only as the displayed decision row for the same assumption A. When D is absent, the carrier has no coordinate that turns $\neg\neg A$ into A.

The forbidden exports require absent data: a selected existential witness obtained from contradiction alone, an undecidable double-negation eliminator, or a family of branches over all naming certificates. Since none of these rows exists in R, accepted routes remain inside the permitted contradiction boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecidableRefutationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DecidableRefutationBoundary}^{\uparrow}$ packet is generated from one closed assumption row, an optional decidability witness, a displayed falsity route, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecidableRefutationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 497.1, Definition 497.2, and Definition 497.3 bind the finite carrier, NameCert obligation surface, and ground-loop exhaustion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecidableRefutationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, excluded-middle branching, host truth over naming certificates, unwitnessed existential extraction, or undecidable double-negation elimination.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for carrier admission, assumption closure, permitted falsity replay, decidable readback, and boundary non-escape.

Chapter 498

A Concrete Naming Certificate for $\mathsf{TranscendentalInductionSocket}^{\uparrow}$

The $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ interface names the finite BEDC surface at which an attempted universal-induction read is exposed as a GAP rather than accepted as a substrate rule. It records the source history, the finite trace already observed, the attempted induction schema, the GAP boundary row, the inscription handoff row, displayed transport, package provenance, and the local naming certificate. The packet is a concrete NameCert carrier for induction-as-boundary; it does not internalize mathematical induction, create a universal quantifier over an infinite domain, or add an axiom to the substrate.

The socket is consumed by chapter 610: the GAP boundary row and inscription handoff row of $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ are the finite source through which $\mathsf{InscriptionPoint}^{\uparrow}$ records forward commitment without turning it into a substrate derivation rule.

Definition 498.1 (TranscendentalInductionSocket carrier)**.** *A* $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (h, T, S, G, A, H, C, P, N)$$

with the following displayed rows:

1. *h is the source* Hist *row at which the attempted universal-induction read is made;*
2. *T is the finite verification trace already present in* BHist*;*
3. *S is the displayed induction-shape request, recording the base row, step row, and requested universal target as a packet to be inspected;*
4. *G is the GAP boundary row that marks the universal target as not discharged inside the substrate;*
5. *A is the inscription handoff row, naming the observer-side commitment position without turning it into a derivation rule;*
6. *H records componentwise* hsame *transports for the source history, finite trace, induction-shape request, GAP row, inscription row, package provenance, and local naming row;*
7. *C records the* Cont *routes that read the finite trace, expose the induction-shape request, route the failed universal read to G, and hand the socket to A;*
8. *P is the* Pkg *provenance over* BHist*,* Hist*, finite traces, GAP rows,* hsame*,* Cont*, and inscription handoff data;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{TranscendentalInductionSocket}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no coordinate for a substrate-internal induction axiom, no hidden dependent recursor over all predicates, no completed infinite trace, and no certificate-level predicate that closes every future observation.

Lean4 ✓ `BEDC.Derived.TranscendentalInductionSocketUp.TranscendentalInductionSocketTasteGate single carrier alignment`

Theorem 498.2 (TranscendentalInductionSocket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TranscendentalInductionSo}}$ *surface has five finite obligations:*

1. *carrier exposure: every accepted object exposes the tuple $I = (h, T, S, G, A, H, C, P, N)$ of Definition 498.1;*
2. *trace exactness: every observed premise read factors through the finite trace T and the displayed source history h;*
3. *GAP boundary: the requested universal target in S is routed to G when it is not discharged by the finite trace;*
4. *inscription handoff: the observer-side commitment is named only by A and is not re-read as a substrate derivation;*
5. *ledger policy:* Pkg *and* NameCert *expose only the listed* BHist, hsame, Cont, *GAP, and handoff rows.*

Lean4 ✓ `BEDC.Derived.TranscendentalInductionSocketUp.TranscendentalInductionSocketPacket namecert obligations`

Proof. Carrier exposure is projection from the tuple of Definition 498.1. The source history, trace, induction-shape request, GAP row, handoff row, transport rows, continuation routes, provenance, and local certificate row are all displayed coordinates.

Trace exactness follows by reading T through h. The finite trace supplies only the records already present in BHist, so it cannot supply the requested universal target by itself.

The GAP boundary obligation is the G coordinate together with the route in C from the failed universal read. The inscription handoff obligation is the A coordinate together with the route in C that exposes the observer-side commitment position without making it available as a rule.

Finally, the package and local certificate rows list no substrate-internal induction axiom, no dependent recursor over all predicates, and no completed infinite trace. The ledger policy is therefore exhausted by the finite socket packet. □

Theorem 498.3 (TranscendentalInductionSocket GAP boundary)**.** *Every consumer read of the attempted universal-induction target in an accepted* $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ *carrier factors through the finite trace T, the induction-shape request S, and the GAP row G. If the consumer asks for the universal target as a substrate theorem, the only exported row is G together with its displayed* Cont *route.*

Lean4 ✓ `BEDC.Derived.TranscendentalInductionSocketUp.TranscendentalInductionSocketPacket gap boundary`

Proof. Project the accepted carrier as $I = (h, T, S, G, A, H, C, P, N)$. The induction-shape request is visible only as the coordinate S, and its already observed premises are read only through T and h.

When the requested universal target is inspected, the continuation route in C sends that request to the boundary row G. This route records the failed substrate-internal discharge rather than producing a theorem row.

No other coordinate of I contains a universal target, completed infinite trace, or induction rule. Thus every consumer of the attempted universal read receives the GAP boundary and not an internal proof of induction. □

Theorem 498.4 (TranscendentalInductionSocket inscription handoff)**.** *Every observer-side read of an accepted* TranscendentalInductionSocket$^{\uparrow}$ *carrier is handed off through the inscription row A after the GAP boundary G has been exposed. The handoff names the position from which the forward commitment is taken, but it exports no substrate rule, no proof constructor, and no classifier branch that can discharge the universal target inside* BHist*.*

Lean4 ✓ `BEDC.Derived.TranscendentalInductionSocketUp.TranscendentalInductionSocketPacket inscription handoff`

Proof. Start with the boundary route of Definition 498.3. The attempted universal target is first exposed as the GAP row G, so the carrier has already refused substrate-internal closure.

The inscription row A is then read through its displayed route in C. This row records the handoff site for the observer-side commitment, and the componentwise transport row H carries it only as a named packet coordinate.

Since A is not a derivation rule and the package row P lists no induction constructor, the handoff cannot be re-entered as an internal proof of the requested universal target. □

Theorem 498.5 (TranscendentalInductionSocket no internal axiom)**.** *An accepted* TranscendentalInductionSocket$^{\uparrow}$ *carrier does not add an induction axiom to BEDC. Its public reads are exhausted by the source history h, finite trace T, induction-shape request S, GAP boundary G, inscription handoff A, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory of Definition 498.1. Each coordinate is a finite BHist row or a displayed transport, continuation, provenance, or naming row.

The GAP boundary theorem shows that the universal target is exported as G rather than as a theorem. The inscription handoff theorem shows that the observer-side row A names a commitment position without turning that position into a substrate constructor.

Therefore the packet has no coordinate from which an induction axiom could be projected. Every public read remains inside the finite socket surface, and the substrate boundary stays 0-axiom. □

Theorem 498.6 (TranscendentalInductionSocket public export)**.** *Every public consumer of an accepted* TranscendentalInductionSocket$^{\uparrow}$ *socket receives only the finite tuple*

$$I = (h, T, S, G, A, H, C, P, N).$$

The attempted universal induction target is exported as the GAP row G, and the observer commitment is exported only as the inscription handoff row A. The packet exports no substrate-internal induction axiom, no dependent recursor over all predicates, no completed infinite trace, and no public row that converts finite observation into a universal theorem.

Proof. Project the carrier by Definition 498.1. The public surface is the displayed tuple $I = (h, T, S, G, A, H, C, P, N)$, with source history, finite trace, induction-shape request, boundary row, handoff row, transport, route, provenance, and local naming all visible as packet coordinates.

Apply Definition 498.2. The local certificate admits only carrier exposure, trace exactness, GAP boundary, inscription handoff, and ledger policy for those coordinates.

Use Definition 498.3. It sends the requested universal target to G whenever the finite trace does not discharge it inside the substrate.

Finally apply Definition 498.4 and Definition 498.5. The handoff row A records observer commitment without becoming a derivation rule, and no remaining coordinate can supply an induction axiom, completed infinite trace, or universal theorem. □

Theorem 498.7 (TranscendentalInductionSocket bridge-facing certificate)**.** *For every accepted* $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ *packet*

$$I = (h, T, S, G, A, H, C, P, N),$$

the bridge-facing public certificate is exactly the conjunction of carrier exposure, NameCert *obligations, GAP boundary, inscription handoff, no-internal-axiom, public export, and the checked TasteGate single-carrier alignment recorded at Definition 498.1.*

Proof. Carrier exposure is supplied by Definition 498.1, which fixes the finite tuple $I = (h, T, S, G, A, H, C, P, N)$ as the whole accepted socket surface.

The certificate obligations are the five finite rows of Definition 498.2. They bind the carrier exposure, trace exactness, GAP boundary, inscription handoff, and ledger policy without adding any substrate rule.

The GAP side is fixed by Definition 498.3, and the observer-side handoff is fixed by Definition 498.4. These two theorems make the requested universal target leave the substrate only as G and make the forward commitment visible only as A.

Apply Definition 498.5 and Definition 498.6. The public package exports only the finite socket packet, and the TasteGate single-carrier alignment already attached to the carrier site records that this export stays on the single displayed carrier. Thus the bridge-facing certificate is a public certificate for the finite socket and not a substrate induction rule. □

Closure status for $\mathsf{NameCert}_{\mathsf{TranscendentalInductionSocket}^{\uparrow}}$.

Constructive story: A $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ packet is generated from a source Hist row, a finite verification trace, an induction-shape request, a GAP boundary row, an inscription handoff row, componentwise hsame transports, visible Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.TranscendentalInductionSocketUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TranscendentalInductionSocket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 498.1, Definition 498.2, Definition 498.3, Definition 498.4, Definition 498.5, Definition 498.6, and Definition 498.7 bind the carrier, NameCert obligations, GAP boundary, inscription handoff, no-internal-axiom theorem, public export, and bridge-facing certificate as the finite socket public surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TranscendentalInductionSocket}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TranscendentalInductionSocketUp.TranscendentalInductionSocketPacket\_namecert\_obligati`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge status, a substrate-internal induction axiom, a completed infinite trace, a dependent recursor over all predicates, or a universal theorem produced from finite observation alone.

Upgrade path: No further paper-side closure grade is claimed beyond bridge verification.

Chapter 499

A Concrete Naming Certificate for $\mathsf{OtherMindsCommitment}^{\uparrow}$

The $\mathsf{OtherMindsCommitment}^{\uparrow}$ surface turns the other-minds locality-cell commitment of Definition 9415.9 into finite BEDC certificate data. It records what an observer may publicly commit to about a candidate other system: visible behavioural evidence, provenance, locality-cell coherence, and naming obligations. It does not expose another system's Self row and does not turn behavioural evidence into direct first-person access.

The locality-cell row L is consumed from chapter 513 as the finite two-observer public cell that $\mathsf{OtherMindsCommitment}^{\uparrow}$ uses before adding the behavioural evidence row E.

Definition 499.1 (OtherMindsCommitment carrier)**.** *An* $\mathsf{OtherMindsCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, B, L, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the observer row, naming the conscious observational system that issues the commitment;*
2. *B is the candidate-other row, naming the external system as it appears in A's record surface;*
3. *L is the locality-cell commitment row, recording the* GAP*-tagged commitment that B is to be treated as another closed observational system for cross-perspective coordination;*
4. *E is the behavioural evidence row, containing the externally observable records that support the commitment without displaying B's* Self*;*
5. *G is the* GAP *and provenance row, recording the open commitment socket and the dependency certificates consumed by the packet;*
6. *H records the componentwise* hsame *transports for the observer row, candidate-other row, locality-cell row, behavioural evidence row, provenance row, package row, and local naming row;*
7. *C records the* Cont *routes that move the observer's public reads through L, E, and G without entering a private* Self *coordinate of B;*
8. *P is the* Pkg *row for the finite BEDC dependencies named by the commitment;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OtherMindsCommitment}^{\uparrow}}$ *row.*

The classifier compares two carriers only by these displayed rows and their hsame *and* Cont *transports.*

Lean4 (definition) `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentUp`

Theorem 499.2 (OtherMindsCommitment TasteGate obligation boundary)**.** *Every paper-side obligation beyond seed closure for* OtherMindsCommitment$^{\uparrow}$ *is read through the finite carrier rows* $A, B, L, E, G, H, C, P, N$ *together with the checked TasteGate target* `BEDC.Derived.OtherMindsCommitmentUp.tas` *No closure promotion may replace those rows by a direct candidate* Self *row, a host mental-state predicate, a private-history read, or a metaphysical identity assertion.*

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentTasteGateObligationBoundary`

Proof. Use Definition 499.1. The carrier coordinates are exactly $A, B, L, E, G, H, C, P, N$, so any public obligation is stated as a finite row read, an hsame transport, a Cont route, a Pkg dependency, or a local NameCert obligation.

The checked TasteGate target is attached to this carrier site, and it supplies the gate through which the chapter may discuss other-minds commitment beyond seed closure. The gate is therefore a discipline on the displayed rows, not a replacement for them.

Apply Definition 499.3 for the finite NameCert surface and Definition 499.7 for the non-escape clauses. Together they exclude direct candidate Self access, host mental-state predicates, private-history reads, and metaphysical identity assertions from any later closure reading. □

Theorem 499.3 (OtherMindsCommitment NameCert obligations)**.** *The local* NameCert$_{\text{OtherMindsCommitment}^{\uparrow}}$ *surface has five finite obligations: carrier exposure by Definition 499.1; classifier stability for the displayed observer, candidate-other, locality-cell, behavioural-evidence,* GAP*, provenance,* Pkg*, and naming rows; locality-cell coherence for the public commitment row* L*; behavioural-evidence accountability for every public read in* E*; and non-escape from direct* Self *access, unrecorded private history, host mental-state predicates, and ambient metaphysical identity.*

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentCarrier namecert obligations`

Proof. Carrier exposure is projection from the finite tuple $O = (A, B, L, E, G, H, C, P, N)$.

Classifier stability is componentwise. The only comparisons available to the packet are the displayed hsame rows in H and the displayed Cont routes in C.

The locality-cell and behavioural-evidence obligations are read from L and E. The row L records the GAP-tagged commitment, while E records the public behavioural evidence consumed by that commitment.

Finally inspect the coordinate list. There is no field for the candidate's private Self, for an unrecorded private history, for a host mental-state predicate, or for an ambient identity assertion beyond the displayed NameCert surface. □

Theorem 499.4 (OtherMindsCommitment locality-cell soundness)**.** *Every accepted* OtherMindsCommitment$^{\uparrow}$ *packet represents an other-minds commitment as a locality-cell row internal to the observer's finite history. A consumer may read the observer row, candidate-other row, public locality-cell commitment, behavioural evidence,* GAP *provenance,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows, but the consumer cannot read a direct* Self *row for the candidate other system.*

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentLocalityCellSoundness`

Proof. Project the accepted packet as $O = (A, B, L, E, G, H, C, P, N)$. The locality-cell commitment is the displayed coordinate L, and its source and target surfaces are the observer row A and candidate-other row B.

The public consumer route is limited by C and transported by H. These rows expose L, E, G, P, and N as finite certificate data.

The candidate's private Self is not a coordinate of the carrier. Thus the soundness claim is exactly locality-cell commitment inside the observer's finite public history, not direct access to another first-person row. □

Theorem 499.5 (OtherMindsCommitment behavioural-evidence compatibility)**.** *If two accepted* OtherMindsCommitment$^{\uparrow}$ *packets have matching observer rows, candidate-other rows, locality-cell commitment rows, and behavioural evidence rows under the displayed* hsame *and* Cont *transports, then they determine the same public other-minds commitment surface. The agreement is behavioural and certificate-local: it does not identify private* Self *rows, host mental states, or unrecorded histories.*

Proof. Use the classifier comparison on the displayed rows. Matching A, B, L, and E fixes the public observer, candidate, locality-cell commitment, and behavioural-evidence coordinates.

Transport along H and C carries the same public evidence into the same Pkg and local NameCert rows. Therefore any consumer sees the same finite commitment surface.

The agreement is not a private-state identity. By Definition 499.4, the carrier has no direct candidate Self coordinate and no host mental-state predicate. □

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentBehaviouralEvidenceCompatibility`

Theorem 499.6 (OtherMindsCommitment public evidence ledger)**.** *Every consumer of an accepted* OtherMindsCommitment$^{\uparrow}$ *packet must factor an other-minds read through the displayed rows* $A, B, L, E, G, H, C, P, N$*. In particular, the consumer reaches the candidate-other commitment only through the public evidence row* E *together with the locality-cell row* L*; no private* Self *row, host mental-state predicate, or unrecorded history is available as a consumer route.*

Proof. Project the accepted carrier as $O = (A, B, L, E, G, H, C, P, N)$. These are the only rows exposed by Definition 499.1.

Use Definition 499.4 to locate the candidate-other commitment inside the locality-cell row L. A consumer may read that commitment only as a public locality-cell coordinate of the observer's finite history.

Use Definition 499.5 to identify the evidence consumed by the read with the displayed behavioural evidence row E, transported by H and routed by C into P and N.

There is no remaining coordinate for a private Self, a host mental-state predicate, or an unrecorded history. Thus the public evidence ledger is the sole consumer route for the other-minds read. □

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentPublicEvidenceLedger`

Theorem 499.7 (OtherMindsCommitment non-escape boundary)**.** *An accepted* OtherMindsCommitment$^{\uparrow}$ *carrier exports only the observer row, candidate-other public row, locality-cell commitment row, behavioural evidence row,* GAP *and provenance row, displayed* hsame *and* Cont *transports,* Pkg *row, and local* NameCert *row. It exports no direct candidate* Self *access, no private-history read, no host mental-state predicate, no metaphysical identity assertion, and no decision procedure for whether every candidate other system is conscious.*

Proof. Inspect the carrier coordinates of Definition 499.1. Every exported row is one of A, B, L, E, G, H, C, P, or N.

The locality-cell and behavioural evidence rows are public to the observer's history. They support a finite commitment surface but do not contain the candidate's private Self or a private-history projection.

No remaining coordinate names a host mental-state predicate, a metaphysical identity row, or a global consciousness decision procedure. Hence none of those objects can escape from the certificate. □

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentNonEscapeBoundary`

Theorem 499.8 (OtherMindsCommitment obligation-closure package)**.** *The carrier of Definition 499.1, the checked TasteGate target* `BEDC.Derived.OtherMindsCommitmentUp.taste_gate`*, the obligation boundary of Definition 499.2, the finite* NameCert *obligations of Definition 499.3, locality-cell soundness from Definition 499.4, behavioural-evidence compatibility from Definition 499.5, the public evidence ledger of Definition 499.6, and the non-escape boundary of Definition 499.7 form the complete obligation-closure package for the AI-origin* OtherMindsCommitment$^{\uparrow}$ *chapter.*

The package closes only the public locality-cell commitment surface over the finite rows $A, B, L, E, G, H, C, P, N$. It introduces no direct candidate Self *access, no host mental-state predicate, no private-history read, and no metaphysical identity assertion.*

Proof. Start with the carrier of Definition 499.1. Its coordinate list is exactly $A, B, L, E, G, H, C, P, N$, so every admissible obligation must be a row projection, a displayed transport, a displayed continuation route, a package dependency, or the local naming row.

The checked TasteGate target named in the statement supplies the chapter gate for AI-origin other-minds commitments. By Definition 499.2, that gate constrains the displayed rows rather than adding a private candidate coordinate or a host predicate.

The finite obligation surface is then exhausted by the existing theorem chain. Apply Definition 499.3 for the carrier and NameCert rows, Definition 499.4 for the locality-cell reading, Definition 499.5 for public evidence compatibility, and Definition 499.6 for the consumer route through E and L.

Finally use Definition 499.7. Since the carrier contains no direct candidate Self row, no private-history coordinate, no host mental-state predicate, and no metaphysical identity row, the package cannot export any of those objects. Thus the listed carrier, gate, obligations, soundness, compatibility, ledger, and boundary clauses are complete for the public obligation-closure reading of this chapter. □

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentObligationClosurePackage`

Theorem 499.9 (OtherMindsCommitment bridge-facing locality route)**.** *Every bridge-facing consumer of an accepted* OtherMindsCommitment$^{\uparrow}$ *packet must read the public locality-cell commitment, public evidence ledger,* BHist *carrier rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row already exposed by*

$$O = (A, B, L, E, G, H, C, P, N).$$

The route is bridge-facing only as a paper-side public surface: it does not add a checked standard bridge, direct candidate Self *access, private-history read, host mental-state predicate, metaphysical identity assertion, or global consciousness decision procedure.*

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.OtherMindsCommitmentBridgeFacingLocalityRoute`

Proof. Start with the obligation-closure package. It fixes the accepted carrier as the finite tuple $O = (A, B, L, E, G, H, C, P, N)$ and routes every public other-minds read through the carrier, TasteGate boundary, NameCert obligations, locality-cell soundness, behavioural-evidence compatibility, public evidence ledger, and non-escape boundary.

The bridge-facing read can therefore inspect L only as the public locality-cell commitment and E only as the public behavioural evidence ledger. The structural rows H, C, P, N transport, replay, source, and name exactly those public rows, and the underlying BHist packet supplies no additional private coordinate.

Apply the non-escape boundary to the remaining cases. Direct candidate Self access, private histories, host mental-state predicates, metaphysical identity, and a global consciousness decision procedure are not carrier rows. Thus the bridge-facing surface is precisely the displayed locality route, while checked bridge status remains outside this paper-side theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{OtherMindsCommitment}^{\uparrow}}$.

Constructive story: An $\mathsf{OtherMindsCommitment}^{\uparrow}$ packet is generated from an observer row, a candidate-other public row, a locality-cell commitment row, behavioural evidence, GAP provenance, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.OtherMindsCommitmentUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OtherMindsCommitment}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 499.1, Definition 499.3, Definition 499.4, Definition 499.5, Definition 499.6, Definition 499.7, Definition 499.8, and Definition 499.9 bind the carrier, locality-cell row, behavioural-evidence row, refusal and non-escape rows, TasteGate marker, and bridge-facing public locality route for the finite public other-minds commitment surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OtherMindsCommitment}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, checked bridge status, direct access to another system's Self, private-history reads, host mental-state predicates, metaphysical identity, or a global consciousness decision procedure.

Upgrade path: Checked bridge status remains the next verification-side work over the displayed locality-cell commitment, behavioural-evidence ledger, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 500

A Concrete Naming Certificate for $\mathsf{RefutationBoundary}^{\uparrow}$

The $\mathsf{RefutationBoundary}^{\uparrow}$ surface records the exact BEDC boundary between permitted refutation and forbidden truth-branch extraction. It is a finite NameCert carrier for consumers that need to use contradiction as a local refutation rule while preserving the open meta loop of chapter 9458. The packet carries enough data to replay permitted negation routes, to consume a decidability row only when one is displayed, and to refuse any export of host excluded middle, nonconstructive existence, quotient truth predicates, or RealUp completeness.

The permitted-refutation route row F, optional decidability row D, and truth-branch exclusion ledger are the boundary consumer of chapter 510, chapter 511, and chapter 508. The dependency is only finite BHist, Cont, hsame, and $\mathsf{NameCert}$ data, not host excluded middle or truth-branch extraction.

When the optional decidability row is absent, the permitted-refutation route consumes the stable negation boundary of chapter 507, while $\mathsf{RefutationBoundary}^{\uparrow}$ keeps the consumer-facing packet finite through $A, F, D, S, T, H, C, P, N$.

Definition 500.1 (RefutationBoundary carrier)**.** *A* $\mathsf{RefutationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, F, D, S, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the inspected proposition row, read as a certificate-visible predicate over the current finite history;*
2. *F is the permitted-refutation route from an assumed witness of A to the displayed bottom row, using only* Cont, hsame, msame, *and the local naming rows;*
3. *D is an optional decidability row for A, present only when the same packet displays a constructive branch for A;*
4. *S is the forbidden-schema descriptor, naming the three blocked shapes of Definition 9458.5: unwitnessed existential extraction, double-negation elimination without D, and excluded-middle branching over the carrier of all naming certificates;*
5. *T is the truth-branch exclusion ledger, recording that no coordinate of the packet exports a family $Q(N) \vee \neg Q(N)$ over all naming certificates;*

6. *H records componentwise* hsame *transports for the proposition row, refutation route, optional decidability row, forbidden-schema descriptor, truth-branch ledger, package provenance, and local naming row;*
7. *C records the displayed* Cont *routes for reading A, replaying F, consuming D when present, checking S, and transporting T;*
8. *P is the* Pkg *provenance over* BHist, Hist, Ext, Cont, msame, hsame, *the local refutation rows, and the surrounding* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RefutationBoundary}^{\uparrow}}$ *ledger.*

The carrier is only this finite packet. It has no host truth predicate, no quotient over truth values, no global excluded-middle coordinate, no witness selector for an existential obtained by contradiction, and no completeness certificate for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryTasteGate single carrier alignment`

Theorem 500.2 (Permitted negation is sound inside the packet)**.** *Every accepted* $\mathsf{RefutationBoundary}^{\uparrow}$ *packet supports the permitted BEDC rule that a displayed route from an assumed witness of A to bottom yields the local negation of A. The rule is stable under the packet's* hsame *transports and displayed* Cont *routes.*

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryCarrier permitted negation sound`

Proof. Project the accepted carrier as $B = (A, F, D, S, T, H, C, P, N)$. The row F is not an ambient proof search result; it is a displayed route from an assumed witness of A through the stored continuation rows to the bottom row.

Transport stability is supplied by H and C. Moving the proposition row and the route along the componentwise hsame data, then replaying the displayed Cont chain, gives the same local map from an admitted witness of A to bottom.

Thus the only negation exported by the packet is the one accounted for by its displayed route. This is exactly the permitted pattern of Definition 9458.1. □

Theorem 500.3 (Forbidden truth branches are excluded)**.** *No consumer read of an accepted* $\mathsf{RefutationBoundary}^{\uparrow}$ *packet can produce a certificate-indexed truth branch for predicates over the carrier of all naming certificates. In particular, the packet does not implement any forbidden schema of Definition 9458.5.*

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryForbiddenTruthBranchExclusion`

Proof. The only classifier-visible coordinates of the carrier are $A, F, D, S, T, H, C, P, N$. None of them ranges over all naming certificates, and none is a family choosing $Q(N) \vee \neg Q(N)$ for arbitrary Q and arbitrary certificate N.

The descriptor S names the blocked schemas, while the ledger T records their non-export condition. A consumer may inspect those rows, transport them through H, and replay their checks through C, but that yields only the finite exclusion ledger already present in the packet.

The forbidden schemas require data outside the carrier inventory: a witness selected from $\neg\forall\neg$, a double-negation eliminator with no displayed decidability row, or excluded middle over naming certificates. Since no such coordinate exists, the truth branch of Definition 9458.10 cannot be extracted. □

Theorem 500.4 (RefutationBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RefutationBoundary}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the finite packet* $B = (A, F, D, S, T, H, C, P, N)$ *of Definition 500.1;*
2. *classifier stability for the visible proposition row, permitted-refutation route, optional decidability row, forbidden-schema descriptor, truth-branch ledger, transports, continuation routes, package provenance, and local naming row;*
3. *permitted-negation soundness, namely the transport-stable route of Definition 500.2;*
4. *decidability accountability, saying that* $\neg\neg A$ *may be consumed only when the row* D *displays a constructive decision for the same proposition row;*
5. *truth-branch exclusion, namely the non-export statement of Definition 500.3.*

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryCarrier namecert obligation surface`

Proof. Carrier admission is projection from Definition 500.1. Every accepted object exposes exactly the displayed finite tuple and no hidden ambient coordinate.

Classifier stability follows by comparing only the visible rows of the tuple. The proposition, refutation route, optional decidability row, forbidden-schema descriptor, truth-branch ledger, provenance, and local naming row are transported by H and read through C.

Permitted-negation soundness is Definition 500.2. The proof route is finite, displayed, and stable under the same componentwise transports.

For decidability accountability, inspect D. If D is present, it is consumed as the displayed constructive branch for A; if it is absent, S and T record that no undecided double-negation eliminator is exported.

Finally, truth-branch exclusion is Definition 500.3. The NameCert ledger records the boundary by carrying the exclusion row rather than a truth predicate over all naming certificates. □

Theorem 500.5 (RealUp consumers receive only boundary data)**.** *A* $\mathsf{Real}^{\uparrow}$ *or completion consumer that reads an accepted* $\mathsf{RefutationBoundary}^{\uparrow}$ *packet receives only the proposition row, displayed refutation route, optional decidability row, forbidden-schema descriptor, truth-branch exclusion ledger, transport rows, continuation routes, package provenance, and local naming row. It receives no completeness theorem, no quotient truth predicate, and no host excluded-middle principle.*

Proof. Any consumer read factors through the carrier coordinates $A, F, D, S, T, H, C, P, N$. These coordinates are finite BHist and ledger rows, so their projections remain inside the displayed NameCert surface.

The refutation data available to the consumer is the route F together with its transports and continuations. If the consumer also needs double-negation elimination, the only available route is the optional row D, and only for the same proposition row A.

The completion-side exclusions are exactly the absent coordinates of Definition 500.1. The packet does not contain a RealUp completeness witness, a quotient truth predicate, or a branch over all naming certificates, so a RealUp consumer cannot obtain any of those principles from this certificate. □

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryRealUpConsumerScope`
Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryCarrier realup consumer scope`

Theorem 500.6 (RefutationBoundary public export)**.** *The public* $\mathsf{RefutationBoundary}^{\uparrow}$ *export is exactly the finite packet*

$$B = (A, F, D, S, T, H, C, P, N)$$

together with the five named obligations of Definition 500.4: carrier admission, classifier stability, permitted-negation soundness, decidability accountability, and truth-branch exclusion. A public consumer may read those rows and obligations, and no further truth predicate, excluded-middle branch, quotient truth object, nonconstructive existential selector, or $\mathsf{Real}^{\uparrow}$ *completeness principle is part of the export.*

Proof. Project the carrier of Definition 500.1. Its visible rows are exactly $A, F, D, S, T, H, C, P, N$, and the carrier definition excludes every host truth predicate, quotient truth object, global excluded-middle coordinate, existential selector obtained by contradiction, and $\mathsf{Real}^{\uparrow}$ completeness certificate.

Now read the obligation package of Definition 500.4. Its five clauses supply precisely carrier admission, classifier stability, permitted-negation soundness, decidability accountability, and truth-branch exclusion for the same finite tuple.

The consumer-scope theorem Definition 500.5 identifies the downstream read as a projection of those rows and obligations. Hence the public export is the displayed packet together with the five obligations, and nothing outside that finite boundary is exported. □

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryCarrier public export`

Theorem 500.7 (RefutationBoundary standard-boundary export)**.** *The public* $\mathsf{RefutationBoundary}^{\uparrow}$ *packet is exactly the finite standard-boundary export for permitted negation and truth-branch refusal. A bridge-facing consumer receives the proposition row, displayed refutation route, optional decidability row, forbidden-schema descriptor, truth-branch exclusion ledger, transports, continuations, provenance, and local naming row, and every permitted-negation read and every truth-branch refusal is computed from those rows alone.*

Proof. Start with the public export of Definition 500.6. It identifies the exported object as the finite tuple $B = (A, F, D, S, T, H, C, P, N)$ together with carrier admission, classifier stability, permitted-negation soundness, decidability accountability, and truth-branch exclusion.

The permitted-negation side is the displayed route F read through H and C. By Definition 500.2, this route supplies only the local map from an assumed witness of A to bottom.

The refusal side is the pair S, T. By Definition 500.3, those rows exclude certificate-indexed truth branches rather than exporting one.

Thus the bridge-facing standard boundary has no further coordinate to expose. It is precisely the public packet already named by the chapter, read as permitted negation on one side and truth-branch refusal on the other. □

Lean4 ✓ `BEDC.Derived.RefutationBoundaryUp.RefutationBoundaryStandardBoundaryExport`

Theorem 500.8 (RefutationBoundary mature consumer route)**.** *Every mature consumer read of an accepted* $\mathsf{RefutationBoundary}^{\uparrow}$ *packet is the same finite route*

$$A, F, D, S, T, H, C, P, N.$$

Permitted negation, truth-branch refusal, $\mathsf{Real}^{\uparrow}$ *consumer scope, public export, and standard-boundary export are all projections of this packet route. No consumer receives an extra host truth predicate, excluded-middle branch, quotient truth object, nonconstructive existential selector, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier of Definition 500.1. It displays exactly the rows $A, F, D, S, T, H, C, P, N$, so every consumer route begins as a projection of the same finite packet.

The permitted-negation projection is Definition 500.2. It reads A and F through the transport and continuation rows H, C and exports only the local route from an assumed witness of A to bottom.

The truth-branch refusal projection is Definition 500.3. It reads S, T, H, C, P, N as the finite exclusion ledger and supplies no certificate-indexed truth branch.

The consumer-facing projections are then Definition 500.5, Definition 500.6, and Definition 500.7. These theorems expose the same rows to $\mathsf{Real}^{\uparrow}$, public, and bridge-facing consumers, so the mature route is the single displayed packet and no further coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RefutationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RefutationBoundary}^{\uparrow}$ packet is generated from a proposition row, a permitted-refutation route, an optional decidability row, a forbidden-schema descriptor, a truth-branch exclusion ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RefutationBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 500.1, Definition 500.2, Definition 500.3, Definition 500.4, Definition 500.5, Definition 500.6, Definition 500.7, and Definition 500.8 bind the finite refutation-boundary rows over BHist, hsame, Cont, Pkg, and local NameCert dependency data as the mature public standard-boundary packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RefutationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close host excluded middle, nonconstructive existence, quotient truth predicates, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the finite packet and checked theorems for permitted-negation soundness, forbidden truth-branch exclusion, NameCert obligations, RealUp consumer scope, public export, and standard-boundary export.

Chapter 501

Constructive Refutation NameCert Construction

The $\mathsf{ConstructiveRefutation}^{\uparrow}$ surface turns the contradiction-boundary discipline of chapter 9458 into a finite naming packet. It records an assumption row, a displayed route from that assumption to contradiction, the contradiction witness itself, the optional decidability row when such a row is supplied, the fixed certificate scope, the transport and continuation rows through which a consumer may replay the route, package provenance, and the local naming ledger. The packet consumes the boundary surface of chapter 502 before exposing permitted refutation routes. The packet certifies constructor-visible refutation only; it does not contain a certificate-indexed truth branch.

The packet consumes the same local assumption and refutation route surface as chapter 493 and the same optional-decision boundary as chapter 497, while keeping the exported route restricted to its own rows A, R, F, D, S, T, P, N.

Definition 501.1 (ConstructiveRefutation carrier)**.** *A* $\mathsf{ConstructiveRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (A, R, F, D, S, T, P, N)$$

with the following displayed rows:

1. *A is the assumption row whose local witness would be consumed by the refutation;*
2. *R is the permitted refutation route, a displayed* Cont *path from the assumption row into the fixed certificate surface of Definition 9458.1;*
3. *F is the contradiction witness produced by replaying R inside the same finite history and local certificate rows;*
4. *D is an optional decidability row for the assumption predicate, present only when the packet carries a constructive branch for that same row;*
5. *S is the scope row naming the fixed* $\mathsf{NameCert}$ *and the finite ground-loop rows in which the refutation is valid;*
6. *T stores the componentwise* hsame *transports and consumer* Cont *routes for A, R, F, D, S, package rows, and local naming rows;*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *the displayed contradiction route, and the surrounding* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRefutation}^{\uparrow}}$ *ledger.*

The carrier contains no coordinate for a host truth predicate, an unwitnessed existence selector, double-negation elimination without the row D, or a branch over the carrier of all naming certificates.

Lean4 ✓ `BEDC.Derived.ConstructiveRefutationUp.ConstructiveRefutationTasteGate single carrier alignment`

Theorem 501.2 (ConstructiveRefutation permitted soundness)**.** *For every accepted* $\mathsf{ConstructiveRefutation}^{\uparrow}$ *carrier*

$$C = (A, R, F, D, S, T, P, N),$$

the exported refutation of A is sound precisely as a certificate-local negation: any displayed witness of A is consumed by the route R, transported through T, and replayed inside the fixed scope S to produce the contradiction witness F. If the optional decidability row D is present, it may be used only for the same assumption row A and only inside the same carrier.

Proof. Project the carrier of Definition 501.1. The rows A, R, and F give the displayed assumption, the route from that assumption into the fixed certificate, and the contradiction witness obtained by replaying the route.

The scope row S fixes the local NameCert and the finite ground-loop rows used by the route. The transport-and-continuation row T moves only the displayed assumption, route, contradiction witness, optional decision row, and scope data, so a consumer cannot replace the route by a host-level truth test.

When D is present, it is listed as a row for the same assumption predicate A. Replaying R with the transported rows therefore yields a certificate-local negation of A, and no step requires an ambient excluded-middle principle. □

Theorem 501.3 (ConstructiveRefutation forbidden branch exclusion)**.** *An accepted* $\mathsf{ConstructiveRefutation}^{\uparrow}$ *carrier cannot export any forbidden pattern of Definition 9458.5. In particular, it has no coordinate for proving an existence claim by contradiction, no coordinate for eliminating double negation unless the displayed row D supplies a constructive decision for the same assumption row, and no coordinate for a certificate-indexed truth branch over all naming certificates.*

Proof. Inspect the carrier inventory in Definition 501.1. Its only rows are A, R, F, D, S, T, P, N.

The boundary sibling supplies the forbidden-branch ledger, so the $\mathsf{ConstructiveRefutation}^{\uparrow}$ carrier cannot erase that ledger when replaying A, R, F, D, S, T, P, N.

The permitted route is constructor-visible: it starts with the assumption row A, follows the displayed route R, and ends at the contradiction witness F within the fixed scope S. The rows T, P, and N record transport, consumer access, provenance, and local naming data for that route.

No listed row is a selector for witnesses of existential claims obtained from contradiction. No listed row is a truth predicate over all naming certificates. The only row that can support double-negation elimination is D, and it is scoped to the same assumption row A.

Thus the packet can export the permitted refutation pattern, but the forbidden truth branches have no carrier coordinate through which they could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRefutation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveRefutation}^{\uparrow}$ packet is generated from an assumption row, a displayed refutation route, a contradiction witness, an optional same-row decidability witness, a fixed certificate scope, transport and continuation rows, package provenance, and a local naming ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRefutation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 501.1, Definition 501.2, and Definition 501.3 bind permitted refutations to finite BHist, Cont, hsame, Pkg, and NameCert rows while excluding certificate-indexed truth branches.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConstructiveRefutation}\uparrow}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not close classical excluded middle, global truth over naming certificates, existential-by-contradiction witness selection, or a checked Lean certificate.
Upgrade path: Obligation closure requires a Lean encoding of the carrier and checked theorems for permitted-route soundness and forbidden-branch exclusion.

Chapter 502

A Concrete Naming Certificate for $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$

The $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$ surface records the proof-discipline boundary of chapter 9458 as a finite NameCert packet. Its carrier separates permitted refutation routes from certificate-indexed truth branches by storing the proof rows that may be replayed, the decidability rows that license double-negation consumption, and the forbidden-branch ledger that prevents a local certificate from exporting classical contradiction principles.

It specializes the refutation-boundary packet of chapter 500 and the stable-negation boundary of chapter 507 by adding the decidability row and forbidden-branch ledger while preserving the refusal of excluded-middle branching and witness selection from contradiction.

Definition 502.1 (Constructive refutation boundary carrier)**.** *A* $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition row being inspected inside the current certificate-visible history.*
2. *R is the permitted refutation data: a displayed route from an assumed witness of A to $\bot$ using only* Cont, hsame, msame, *and the local certificate rows.*
3. *D is the decidability row for A when such a row is present; it is the only row that permits a double-negation readback for the same proposition row.*
4. *L is the forbidden-branch ledger recording that existential extraction by contradiction, double-negation elimination without D, and excluded-middle branching over all naming certificates are unavailable to the packet.*
5. *H stores the componentwise* hsame *transports for A, R, D, L, package rows, and local naming rows.*
6. *C stores the displayed* Cont *routes for reading A, replaying R, consuming D when present, and checking L.*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *permitted proof rows, decidability rows, forbidden ledgers, and the surrounding* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRefutationBoundary}^{\uparrow}}$ *row.*

The carrier contains no host truth predicate, no excluded-middle oracle over the carrier of all naming certificates, and no witness selector for an existential obtained only from contradiction.

Theorem 502.2 (Constructive refutation stability)**.** *For an accepted* $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$ *carrier* $B = (A, R, D, L, H, C, P, N)$*, every consumer read of the permitted refutation row remains a certificate-level proof of* $\neg A$ *after the displayed* hsame *transports and* Cont *routes are replayed. If a double-negation readback is consumed, it factors through the displayed decidability row* D *for the same proposition row.*

Proof. Project the finite carrier. The refutation row R is already a route from an assumed witness of A to $\bot$ using only the displayed ground-loop data.

Transport the proposition row and the refutation route along H, then replay the corresponding continuation route in C. The transported consumer still receives a route from a transported witness of A to $\bot$, so it receives the same certificate-level negation and no ambient truth value.

For double-negation readback, inspect the displayed row D. When D is present, the readback uses the constructive branch carried by the same packet and the same proposition row. When D is absent, the ledger L blocks the double-negation schema named in Definition 9458.5. Thus every permitted route remains stable inside the finite packet. □

Theorem 502.3 (Forbidden branches do not export)**.** *An accepted* $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$ *packet cannot export a certificate-indexed truth branch. Every consumer read factors through* A, R, D, L, H, C, P, N*, and the ledger row* L *excludes existential extraction by contradiction, double-negation elimination without the displayed decidability row, and excluded-middle branching over the carrier of all naming certificates.*

Proof. Carrier admission exposes only the displayed rows. A consumer may read the proposition row, the permitted refutation route, the optional decidability row, the forbidden-branch ledger, transports, continuations, package provenance, and the local naming row.

Apply the ledger row L. Its three exclusions match the forbidden schemas of Definition 9458.5: unwitnessed existential extraction, double-negation without a decidability row, and excluded-middle branching over all naming certificates.

No row of B classifies all naming certificates, selects witnesses from contradiction alone, or internalizes the host truth predicate. Therefore the consumer cannot obtain the certificate-indexed branch of Definition 9458.10; it remains inside the constructive refutation boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRefutationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveRefutationBoundary}^{\uparrow}$ packet is generated from a proposition row, permitted refutation data, an optional decidability row, a forbidden-branch ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRefutationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 502.1, Definition 502.2, and Definition 502.3 bind the finite proof-discipline carrier separating permitted refutation routes from certificate-indexed truth branches.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRefutationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close classical excluded middle, global truth over naming certificates, witness selection from contradiction alone, or checked Lean realization.

Upgrade path: Obligation closure requires Lean encoding of the packet and checked carrier, stability, ledger, and non-export obligations for the named proof-discipline surface.

Chapter 503

A Concrete Naming Certificate for $\mathsf{WitnessedRefutation}^{\uparrow}$

The $\mathsf{WitnessedRefutation}^{\uparrow}$ surface packages the permitted contradiction boundary as finite NameCert data. It records the source history whose claim is being rejected, the continuation packet that turns an attempted witness into a displayed bottom row, the transported refutation ledger, and the local naming certificate that keeps the whole argument inside BHist, Cont, hsame, and $\mathsf{NameCert}$. The packet does not prove an unwitnessed existence statement, does not eliminate double negation for an undecided classifier, and does not branch on excluded middle for certificate-level predicates.

Definition 503.1 (Witnessed refutation carrier)**.** *A* $\mathsf{WitnessedRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (h, A, R, b, H, C, P, N)$$

with the following displayed rows:

1. *h is the source* Hist *row at which the rejected claim is read;*
2. *A is the attempted-witness packet, recording only the visible BEDC evidence whose use is being refuted;*
3. *R is the continuation/refutation packet, a finite list of* Cont *rows that sends each admitted attempted witness for A at h to the bottom row;*
4. *b is the produced bottom witness, displayed as the terminal absurdity row reached by the continuation packet;*
5. *H records the componentwise* hsame *transports for the source history, attempted-witness packet, refutation packet, bottom row, package provenance, and local naming row;*
6. *C records the* Cont *routes that read the source history, expose the attempted witness, execute the refutation packet, and return the bottom row to the ledger;*
7. *P is the* Pkg *provenance over* BHist, Hist, Cont, hsame, *explicit bottom witnesses, and the surrounding certificate rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{WitnessedRefutation}^{\uparrow}}$ *ledger.*

The carrier is exactly this finite packet. It has no coordinate for unwitnessed excluded middle, double-negation elimination on an undecided proposition, proof by contradiction into an existential conclusion, or a predicate over the carrier of all naming certificates.

Lean4 ✓ BEDC.Derived.WitnessedRefutationUp.WitnessedRefutationTasteGate single carrier alignment

Definition 503.2 (Witnessed refutation classifier)**.** *The* $\mathsf{WitnessedRefutation}^{\uparrow}$ *classifier compares two accepted packets only by transported displayed witnesses. It relates*

$$W = (h, A, R, b, H, C, P, N)$$

and

$$W' = (h', A', R', b', H', C', P', N')$$

when the source histories, attempted-witness packets, continuation/refutation packets, produced bottom witnesses, transport ledgers, continuation routes, package provenance, and local naming rows are connected by the recorded componentwise hsame *rows and by the displayed* Cont *routes. Thus classifier transport carries exactly the same refutation evidence from* W *to* W'*: the source* Hist *row, the attempted witness, the continuation packet, and the produced bottom witness.*

Theorem 503.3 (WitnessedRefutation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{WitnessedRefutation}^{\uparrow}}$ *surface has five finite obligations:*

1. *carrier exposure: every accepted object exposes the tuple* $W = (h, A, R, b, H, C, P, N)$ *of Definition 503.1;*
2. *classifier stability: classifier-related packets transport the source row, attempted-witness packet, refutation packet, bottom witness, provenance, and local naming row by the displayed* hsame *data;*
3. Cont*-locality: the refutation route uses only the continuation rows recorded in* R *and* C*;*
4. *ledger witness accountability: every produced bottom row* b *is reached from a displayed attempted witness through the recorded route, and no bottom row is accepted without that route;*
5. *separation from unwitnessed schemas: the packet exports no excluded-middle branch, no double-negation eliminator for an undecided classifier, and no proof-by-contradiction rule whose conclusion is an existential witness.*

Proof. Carrier exposure is projection from the finite tuple of Definition 503.1. Each coordinate is a displayed BHist row or a finite ledger row named by the tuple.

Classifier stability is exactly Definition 503.2. The comparison transports the same source history, attempted-witness packet, refutation packet, bottom witness, provenance, and naming row, and has no hidden coordinate to compare.

Cont-locality follows by inspecting R and C. The route from an attempted witness to b is a finite continuation chain recorded in those coordinates.

Ledger witness accountability follows because b is a coordinate only together with the route that produces it. A consumer projecting b must also project the source, attempted witness, and continuation packet that account for it.

Finally, separation from unwitnessed schemas is immediate from the carrier inventory. None of the coordinates names an excluded-middle branch over certificates, an undecided double-negation eliminator, or an existential witness produced without its constructor. □

Theorem 503.4 (Witnessed refutation separation from forbidden schemas)**.** *Every consumer read of an accepted* $\mathsf{WitnessedRefutation}^{\uparrow}$ *packet factors through the displayed source history, attempted-witness packet, continuation/refutation packet, and produced bottom witness. Consequently the packet realizes only the permitted rule "a displayed witness of* A *leads by displayed* Cont *rows to* $\bot$*, hence* A *is refuted at this certificate surface" and cannot be read as any of the forbidden contradiction schemas of Definition 9458.5.*

Proof. Project the accepted packet as $W = (h, A, R, b, H, C, P, N)$. Any consumer read is one of these displayed coordinates or a component transported through H and exposed through C.

The only negative conclusion available from the packet is the function that sends an admitted attempted witness for A at h through the recorded continuation packet R to the displayed bottom witness b. This is the permitted refutation shape of Definition 9458.1.

The three forbidden shapes require data absent from the packet: a constructor for an existential obtained from $\neg\forall\neg$, a double-negation eliminator for an undecided classifier, or a certificate-indexed excluded-middle branch. Since the carrier and classifier expose none of these coordinates, no consumer can extract them from $\mathsf{WitnessedRefutation}^{\uparrow}$. □

Theorem 503.5 (Witnessed refutation concretizes the contradiction boundary)**.** *The* $\mathsf{WitnessedRefutation}^{\uparrow}$ *certificate realizes the proof-discipline boundary of chapter 9458: permitted refutations are represented as explicit* Hist*-level evidence carried through* Cont *and* hsame*, while the unwitnessed contradiction schemas remain outside the NameCert surface.*

Proof. The permitted side is the packet $W = (h, A, R, b, H, C, P, N)$. Its source history, attempted-witness row, continuation/refutation packet, and produced bottom witness are all explicit BEDC rows, and the classifier transports them only by hsame and displayed Cont routes.

The boundary side is Definition 503.4. Since the carrier has no coordinate for certificate-level excluded middle, undecided double-negation elimination, or existential-by-contradiction, the packet cannot close the meta loop described in the vision chapter.

Thus the certificate is a concrete NameCert realization of the permitted contradiction boundary: refutation is admitted exactly when its bottom witness and route are displayed in the ledger. □

Theorem 503.6 (Witnessed refutation carrier admission obligation)**.** *For an accepted* $\mathsf{WitnessedRefutation}^{\uparrow}$ *packet, carrier admission is exactly the finite tuple*

$$W = (h, A, R, b, H, C, P, N)$$

of Definition 503.1. The admitted rows are the source Hist *row* h*, attempted-witness packet* A*, displayed continuation/refutation packet* R*, produced bottom witness* b*, componentwise* hsame *ledger* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{WitnessedRefutation}^{\uparrow}}$ *row* N*. The AI single-carrier evidence is the checked TasteGate alignment marker attached to Definition 503.1, so no second carrier coordinate is available.*

Proof. Project the accepted packet to the tuple of Definition 503.1. Each admitted coordinate is named in that tuple and is read as a displayed BHist row or a finite ledger row.

The TasteGate alignment fixes the AI surface to this one carrier inventory. Since the inventory contains only h, A, R, b, H, C, P, N, a consumer cannot admit a hidden witness source, an extra bottom row, or a second naming carrier.

Thus carrier admission is exhausted by the concrete packet rows, with no route to excluded middle or to an unwitnessed contradiction schema. □

Theorem 503.7 (Witnessed refutation route transport obligation)**.** *Classifier transport for* $\mathsf{WitnessedRefutation}^{\uparrow}$ *carries the displayed refuter route only through the rows of Definition 503.1. If*

$$W = (h, A, R, b, H, C, P, N)$$

and

$$W' = (h', A', R', b', H', C', P', N')$$

are classifier-related accepted packets, then the source history, attempted-witness packet, continuation/refutation packet, produced bottom row, provenance, and local naming row are transported by the stored hsame *data and replayed through the displayed* Cont *routes. No transport step may replace the refuter route by a non-displayed proof by contradiction.*

Proof. Start with the classifier of Definition 503.2. It relates two packets by componentwise transport of the same source row, attempted-witness packet, refutation packet, bottom witness, provenance, and naming row.

Read the refuter route through R and C. The continuation rows execute the attempted witness route to the bottom witness, while H transports the named coordinates between classified packets.

Because the classifier has no coordinate beyond the displayed tuple, the transported route remains a Cont route over the same finite ledger. It cannot be replaced by an excluded-middle branch, a double-negation eliminator, or an existential conclusion without a witness row. □

Theorem 503.8 (Witnessed refutation bottom-ledger non-escape obligation)**.** *Every bottom-ledger read of an accepted* $\mathsf{WitnessedRefutation}^{\uparrow}$ *packet factors through the source row h, attempted-witness packet A, continuation/refutation packet R, produced bottom witness b,* hsame *ledger H,* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N. Consequently the packet exports no excluded-middle branch, no double-negation eliminator for an undecided classifier, and no unwitnessed contradiction schema outside the displayed bottom route.*

Proof. Read the bottom row from the carrier. The coordinate b is stored only with the source row, attempted-witness packet, refutation packet, transport ledger, continuation routes, provenance, and local naming row.

Apply the forbidden-schema separation theorem. It already identifies the only negative conclusion available from the packet as the displayed route sending an admitted attempted witness through R to b.

The bottom ledger therefore has no escape row. A consumer may project the witnessed refutation route, but it cannot extract certificate-level excluded middle, undecided double-negation elimination, or an existential-by-contradiction rule. □

Theorem 503.9 (Witnessed refutation RefutationWitness dependency boundary)**.** *Every accepted* $\mathsf{WitnessedRefutation}^{\uparrow}$ *packet consumes the same assumption-to-bottom route shape exposed by chapter 511. The witnessed packet adds the attempted-witness row A and TasteGate single-carrier alignment for $W = (h, A, R, b, H, C, P, N)$, but its bottom-ledger non-escape row is the witnessed form of the* $\mathsf{RefutationWitness}^{\uparrow}$ *route replay surface: a displayed assumption is carried through recorded* Cont *rows to bottom, with no excluded-middle branch, undecided double-negation eliminator, or unwitnessed contradiction schema.*

Proof. Read the sibling route first. chapter 511 exposes the constructive assumption-to-bottom surface as a finite route from an admitted assumption through displayed continuation rows to a bottom witness.

Project the witnessed packet. Its carrier adds the attempted-witness row A to the source history, refuter route, bottom ledger, transport, continuation, provenance, and local naming rows, and the checked TasteGate alignment attached to Definition 503.1 fixes those rows as one carrier surface.

Apply Definition 503.8. The bottom read still factors through the displayed source, attempted witness, continuation/refutation packet, bottom row, transport, continuation, provenance, and

naming rows. Thus the dependency is substantive: the witnessed bottom-ledger row is the RefutationWitness route replay boundary with the attempted witness and single-carrier alignment made explicit. □

Closure status for $\mathsf{NameCert}_{\mathsf{WitnessedRefutation}\uparrow}$.

Constructive story: A $\mathsf{WitnessedRefutation}^{\uparrow}$ packet is generated from a source Hist row, an attempted-witness packet, a displayed Cont refutation route, the produced bottom witness, componentwise hsame transports, Pkg provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WitnessedRefutation}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 503.1, Definition 503.2, Definition 503.3, Definition 503.6, Definition 503.7, Definition 503.8, Definition 503.4, and Definition 503.5 bind the finite carrier, displayed classifier, carrier admission, refuter-route transport, bottom-ledger non-escape, and separation from unwitnessed contradiction schemas; the same displayed bottom route supplies the paper bridge from witnessed refutation to the abstract permitted-refutation schema.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WitnessedRefutation}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.WitnessedRefutationUp.WitnessedRefutationTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.WitnessedRefutationUp.taste gate`

Not claimed: This chapter does not close bridge status, excluded-middle branch, undecided double-negation eliminator, existential-by-contradiction rule, or predicate over the carrier of all naming certificates.

Upgrade path: No paper-side closure upgrade remains; bridge checking is the remaining formal-axis route.

Chapter 504

A Concrete Naming Certificate for $\mathsf{FiniteTraceGapSocket}^{\uparrow}$

The $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ surface turns the finite-trace side of chapter 9393 into a concrete BEDC packet. It records a bounded observation list, the explicit GAP socket that marks the unclosed universal claim, the finite replay route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, and the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows that keep the socket auditable. The packet accounts for finite windows only; it does not supply mathematical induction, host completeness, a selected real limit, or a universal quantifier over not-yet-observed rows.

Definition 504.1 (FiniteTraceGapSocket carrier)**.** *A* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, W, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite observation trace, a bounded list of previously recorded observation rows;*
2. *W is the GAP socket row attached to the claim that the finite trace does not close;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that replays the observation windows of T;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the replayed finite windows when the observations are consumed by the real-and-completion interface;*
5. *Q is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, present only as a consumer-facing boundary for the finite readback;*
6. *H records componentwise* hsame *transports for the trace, socket, stream schedule, regular readback, real seal, package provenance, and local naming row;*
7. *C records the displayed* Cont *routes that read the trace, expose the GAP socket, replay each admitted finite window, hand the replay to R, and expose Q only after that handoff;*
8. *P is the* Pkg *provenance over* BHist, Gap, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$ *row.*

The carrier is exactly this finite trace socket. It has no row for induction, no row for a completed infinite observation stream, no selected limit, no host real equality test, and no completeness principle.

Theorem 504.2 (FiniteTraceGapSocket NameCert obligations)**.** *For the carrier of Definition 504.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$ *surface consists of five finite rows:*

1. *trace carrier: every accepted object exposes the tuple* $G = (T, W, S, R, Q, H, C, P, N)$ *and the bounded observation list* T*;*
2. *socket classifier: classifier comparison transports the trace row, GAP socket row, replay schedule, readback row, real seal row, provenance row, and local naming row by the displayed* hsame *data;*
3. *finite observation replay: each consumer read of an observed window factors through* T*, then through the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, and then through the displayed* Cont *route into* R*;*
4. *ledger non-escape: the package row* P *exhausts the available dependencies and exposes no induction rule, hidden infinite stream, selected limit, host equality, trichotomy, or completeness datum;*
5. $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *handoff: the finite replay schedule* S *is the only source for the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and the real seal* Q *is readable only after that handoff.*

Proof. Trace carrier exposure is projection from Definition 504.1. The bounded list T, GAP socket W, replay schedule S, readback row R, seal row Q, transports, continuation routes, provenance, and local naming row are all coordinates of one finite BHist packet.

Socket classifier stability is componentwise. The stored hsame rows in H compare exactly the displayed coordinates, and the Cont rows in C replay the same trace, socket, schedule, readback, seal, package, and naming routes after transport.

For finite observation replay, a consumer first reads a window from T. The only stream-facing route then passes through the schedule S, and the only regular-readback route passes from S to R through the displayed continuation ledger.

Ledger non-escape follows from the package inventory. Since P names only the displayed BEDC rows, any induction rule, infinite stream, selected limit, host equality test, trichotomy branch, or completeness principle would require an extra coordinate absent from G.

Finally, the $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ handoff is the route T to S to R to Q. Thus the real-and-completion consumer sees only finite replay data and the visible GAP socket that records what remains unclosed. □

Theorem 504.3 (FiniteTraceGapSocket classifier transport)**.** *For an accepted* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *carrier* $G = (T, W, S, R, Q, H, C, P, N)$*, componentwise* hsame *transport preserves the displayed trace row* T*, GAP socket row* W*, stream schedule* S*, regular readback* R*, real seal boundary* Q*, transport ledger* H*, continuation ledger* C*, provenance row* P*, and local naming row* N*. The transport does not add host real equality, quotient equality, trichotomy, completeness, induction, a selected limit, or an infinite observation stream.*

Proof. Project the carrier of Definition 504.1. Its transport coordinate H is indexed by exactly the displayed rows $T, W, S, R, Q, H, C, P, N$, so classifier comparison can only move a consumer along these named component rows.

The continuation ledger C remains tied to the same coordinates after transport. Thus the trace read, the GAP socket read, the replay schedule, the regular readback, the real seal boundary, the provenance row, and the local $\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$ row are preserved component by component.

Any host equality, quotient equality, trichotomy branch, completeness principle, induction rule, selected limit, or infinite stream would require a coordinate not present in G. Since the transport is componentwise over the existing carrier rows, it cannot create such a coordinate. □

Theorem 504.4 (FiniteTraceGapSocket replay exactness)**.** *Every observed-window read of an accepted* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *packet factors through the finite route*

$$T \to S \to R.$$

The real seal row Q is visible only after the regular readback row R has been reached through that route.

Lean4 ✓ `BEDC.Derived.FiniteTraceGapSocketUp.FiniteTraceGapSocket replay route package`

Proof. An observed-window read begins with the finite trace coordinate T of the carrier. No other coordinate of G is an observation source, so the read first selects one of the bounded rows already present in T.

The stream-facing continuation in C then sends that selected finite window through the replay schedule S. The regular-readback continuation is the displayed handoff from S to R, so the read factors through $T \to S \to R$ before any real-facing row is available.

Finally, Q is listed as the downstream $\mathsf{Real}^{\uparrow}$ seal boundary, not as an independent observation source. It is therefore exposed only after the regular readback R has been obtained from the finite replay path. □

Theorem 504.5 (FiniteTraceGapSocket ledger non-escape)**.** *For an accepted* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *carrier $G = (T, W, S, R, Q, H, C, P, N)$, the package row P, the local naming row N, and the GAP socket W leave mathematical induction, selected limits, infinite streams, trichotomy, host real equality, quotient real equality, and completeness outside the packet.*

Proof. Read the package inventory P and local naming row N from the carrier. They name the finite trace, the GAP socket, the replay schedule, the regular readback, the real seal boundary, the componentwise transports, the displayed continuations, provenance, and the local naming certificate.

The GAP socket W records precisely that the finite trace leaves the universal claim open. It is a named row of the packet, not a constructor for induction, a selected limit, an infinite stream, trichotomy, host equality, quotient equality, or completeness.

Thus any consumer using P, N, or W remains inside the finite carrier rows. The listed non-finite principles are not coordinates of G, so the ledger cannot provide them by route composition. □

Theorem 504.6 (FiniteTraceGapSocket real-window boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *packet factors through the finite route*

$$T \to W \to S \to R \to Q \to H \to C \to P \to N.$$

The consumer may inspect the observed finite windows and the GAP socket that marks the unclosed universal claim, but it cannot read induction, host completeness, a selected real limit, or an infinite observation stream from the packet.

Proof. Project the accepted carrier $G = (T, W, S, R, Q, H, C, P, N)$. The only observation source is the finite trace T, and the only record of what the trace does not close is the GAP socket W.

Apply Definition 504.2. Its finite observation replay row routes every observed window through S and then into R, while its handoff row exposes Q only after that regular readback.

The ledger non-escape row rules out the unavailable structure. Therefore every real-window or completion-facing read remains inside the displayed packet and treats W as a socket rather than as an internally discharged universal claim. □

Theorem 504.7 (FiniteTraceGapSocket inscription-point sibling route)**.** *Every* $\mathsf{InscriptionPoint}^{\uparrow}$ *consumer that reads the finite GAP row through chapter 610 can do so only after the* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *rows*

$$T, W, S, R, Q, H, C, P, N$$

of Definition 504.1 are exposed. The sibling route introduces no hidden supply rule, induction rule, or infinite stream.

Proof. Project the accepted carrier $G = (T, W, S, R, Q, H, C, P, N)$ from Definition 504.1. The finite GAP socket is the displayed row W, and its replay, readback, real-window boundary, transports, continuations, provenance, and naming rows are exactly the remaining coordinates of G.

Now read the sibling $\mathsf{InscriptionPoint}^{\uparrow}$ carrier of chapter 610. Its finite GAP input is named as the $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ coordinate G before any inscription-point consumer can use that gap row.

Therefore the inscription-point route factors through this chapter's finite socket packet first. Since neither carrier contains a row for a hidden supply rule, induction rule, or infinite observation stream, the sibling route cannot introduce one while consuming the finite GAP row. □

Theorem 504.8 (FiniteTraceGapSocket scoped support package)**.** *The carrier of Definition 504.1, the NameCert obligations of Definition 504.2, classifier transport from Definition 504.3, replay exactness from Definition 504.4, ledger non-escape from Definition 504.5, the real-window boundary from Definition 504.6, and the InscriptionPoint sibling route from Definition 504.7 exhaust the scoped BEDC support for* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *over* BHist*,* Gap*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Within that scope, induction, host completeness, selected limits, quotient real equality, real-order trichotomy, and infinite observation streams have no carrier coordinate and no continuation route.

Proof. Read the support package by projecting the carrier $G = (T, W, S, R, Q, H, C, P, N)$. The carrier theorem supplies exactly the finite trace, GAP socket, stream schedule, regular readback, real seal boundary, transport rows, continuation rows, provenance row, and local naming row.

The NameCert obligations bind those rows as the available certificate surface. Classifier transport preserves the same coordinates, replay exactness forces every observed-window read through $T \to S \to R$, ledger non-escape rules out hidden supply from P, and the real-window boundary keeps each completion-facing consumer on the displayed finite route.

It remains to account for the external consumer named in the scope. The sibling-route theorem states that an $\mathsf{InscriptionPoint}^{\uparrow}$ consumer reaches the finite GAP row only through the same $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ packet, so the route adds no coordinate beyond the rows already listed.

Thus every named support route is one of the displayed carrier, certificate, classifier, replay, ledger, real-window, or sibling rows. The principles listed outside the scope would require a carrier coordinate or continuation absent from G, so they are not supplied by the package. □

Theorem 504.9 (FiniteTraceGapSocket TasteGate alignment record)**.** *The checked Lean carrier-alignment target* $BEDC.Derived.FiniteTraceGapSocketUp.FiniteTraceGapSocketTasteGate_single_carrier_$ *names the same finite packet surface as Definition 504.8. It is used here only as verification of the single-carrier surface for* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$*, not as a paper-side substitute for the carrier, NameCert obligations, classifier transport, finite replay exactness, ledger non-escape, real-window boundary, or sibling-route obligations.*

Proof. The named Lean target is attached to the $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ TasteGate carrier. Its role is to confirm that the checked carrier-alignment surface is the single finite packet described by the chapter, rather than a separate theory object.

The paper obligations remain the scoped rows assembled in Definition 504.8. Therefore the Lean target records verification of the carrier surface while the displayed theorem and its cited support rows state the paper-side closure content. □

Theorem 504.10 (FiniteTraceGapSocket public consumer export)**.** *Every public consumer of an accepted* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *packet receives exactly the finite socket export assembled by the carrier of Definition 504.1, the NameCert obligations of Definition 504.2, classifier transport from Definition 504.3, replay exactness from Definition 504.4, ledger non-escape from Definition 504.5, the real-window boundary from Definition 504.6, the InscriptionPoint sibling route from Definition 504.7, the scoped support package of Definition 504.8, and the TasteGate alignment record of Definition 504.9. The export contains no induction rule, infinite observation stream, selected limit, quotient real equality, host completeness, or real-order trichotomy.*

Proof. Start with a public consumer of an accepted carrier $G = (T, W, S, R, Q, H, C, P, N)$. The carrier definition supplies the displayed finite trace, GAP socket, stream schedule, regular readback, real seal boundary, componentwise transports, continuation routes, provenance, and local naming row.

The certificate and transport rows identify the consumer-facing export. Apply Definition 504.2 for the finite NameCert surface and Definition 504.3 for componentwise preservation of the same rows.

Next, the replay and boundary theorems determine the only finite observation route. Apply Definition 504.4, Definition 504.5, and Definition 504.6; together they force observed-window reads through the displayed finite route and keep the non-finite principles outside the packet.

It remains to account for the named public consumers. The InscriptionPoint sibling route of Definition 504.7 consumes the GAP row only through this finite socket, while Definition 504.8 bundles the same carrier, certificate, classifier, replay, ledger, boundary, and sibling rows into the scoped support package.

Finally, Definition 504.9 records that the checked TasteGate target names this same single-carrier surface. Hence the public consumer receives exactly the finite socket export described above and no induction rule, infinite stream, selected limit, quotient real equality, host completeness, or trichotomy row. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ packet is generated from a bounded finite observation trace, an explicit GAP socket row, a $\mathsf{StreamName}^{\uparrow}$ replay schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 504.1, Definition 504.2, Definition 504.3, Definition 504.4, Definition 504.5, Definition 504.6, Definition 504.7, Definition 504.8, Definition 504.9, and Definition 504.10 bind the carrier, the total NameCert obligation surface, classifier transport, finite replay exactness, ledger non-escape, the real-window boundary, the inscription-point sibling route, the scoped support package, the checked single-carrier alignment record, and the public bridge-facing finite-trace GAP socket route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteTraceGapSocket}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.FiniteTraceGapSocketUp.taste gate`

Lean target: `BEDC.Derived.FiniteTraceGapSocketUp.FiniteTraceGapSocketTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close mathematical induction, host completeness, selected real limits, quotient real equality, real-order trichotomy, or infinite observation streams.

Upgrade path: No further theory-axis upgrade is claimed for the finite-trace GAP socket export.

Chapter 505

A Concrete Naming Certificate for $\mathsf{CausalCommitment}^{\uparrow}$

The $\mathsf{CausalCommitment}^{\uparrow}$ surface packages the causality reading of Definition 9415.1 as finite NameCert data. It records observed regularity rows, the GAP socket that binds those rows to not-yet-observed continuations, transport evidence for the visible ledger, and the local naming row that prevents the packet from becoming a hidden law of nature, selected future record, or global observer frame.

The forward-GAP refinement is consumed through the locality-cell surface of chapter 509 and, when the consumer is observer-indexed, through the observer-locality surface of chapter 513. The precise dependency is asymmetric: $\mathsf{CausalCommitment}^{\uparrow}$ supplies the observed-regularity GAP row, while the locality-cell chapters restrict the consumer to local observer cells and prevent selected future observations or global observer frames.

Definition 505.1 (Causal commitment carrier)**.** *A* $\mathsf{CausalCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, R, G, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite observed-history ledger whose records support the regularity read;*
2. *R is the observed regularity ledger, listing the displayed pattern rows read from O;*
3. *G is the* GAP *socket row whose content is that the regularity rows in R remain binding for not-yet-observed continuations;*
4. *F is the forward-binding route from the observed ledger to the continuation-facing commitment row;*
5. *H records componentwise* hsame *transports for O, R, G, F, package provenance, and the local naming row;*
6. *C records the* Cont *routes that read the observed ledger, expose the regularity rows, enter the* GAP *socket, and return the commitment row to the finite packet;*
7. *P is the* Pkg *provenance over* BHist, Hist, GAP, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no selected future observation, no universal empirical law, no hidden induction rule, no synchronized global frame, and no ambient causal primitive.

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentTasteGate single carrier alignment`

Theorem 505.2 (CausalCommitment GAP socket)**.** *Every accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet reads the causal step through the displayed* GAP *socket row* G*. The observed ledger* O *and regularity ledger* R *supply only finite records; the forward-looking content is named only by* G *and cannot be projected as an internally generated universal law.*

Proof. Project the carrier of Definition 505.1. The only row whose content points beyond the observed ledger is G.

The rows O and R are finite history and regularity ledgers. They display what has been observed and what pattern has been read, but they do not contain a row for all future continuations.

Therefore any consumer that asks for the causal step must pass through the GAP socket. Since no other coordinate names future universality, the packet cannot expose an internally generated empirical law. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier gap socket`

Theorem 505.3 (CausalCommitment forward-binding route)**.** *For an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet, the route from observed regularity to causal commitment is exactly the displayed forward-binding row* F *together with the* Cont *routes in* C*. The route starts at the finite ledger* O, R*, enters the* GAP *socket* G*, and returns only the local commitment row named by* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$*.*

Proof. Start from the observed side. The packet first exposes O and R, so the route has a displayed finite source.

The forward-facing step is then recorded by F and replayed by C. These rows send the observed regularity ledger through G rather than through an unrecorded rule of induction.

At the output side the consumer receives the local commitment row N together with the same provenance and transport data. Thus the forward-binding route is finite, named, and confined to the displayed packet. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier forward binding route`

Definition 505.4 (CausalCommitment forward-GAP carrier)**.** *A forward-GAP refinement of the* $\mathsf{CausalCommitment}^{\uparrow}$ *carrier is the finite packet*

$$K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$$

where O, R, G, F, H, C, P, N *are the rows of Definition 505.1, and* L *is the locality-cell consumer row. The row* L *records the consumer-facing handoff to* $\mathsf{LocalityCell}^{\uparrow}$ *and* $\mathsf{ObserverLocalityCell}^{\uparrow}$*: it may read the displayed commitment only through the finite locality cell generated from* O, R, G, F, H, C, P, N*, and it contains no selected future observation, no global observer frame, and no ambient causal primitive.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapCarrier locality replay package`

Theorem 505.5 (CausalCommitment forward-GAP obligations)**.** *For an accepted forward-GAP carrier* $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *surface exposes exactly five consumer-facing obligations:*

1. *carrier exposure for the displayed rows $O, R, G, F, L, H, C, P, N$;*
2. GAP *accountability, requiring every forward-looking read to pass through G;*
3. *forward-route stability, requiring F and C to replay the observed-regularity route without adding a hidden induction rule;*
4. *locality-cell consumer restriction, requiring every observer-locality consumer to read through L;*
5. *non-escape for selected future observations, global observer frames, universal empirical laws, hidden induction rules, and ambient causal primitives.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapObligations`

Proof. Carrier exposure is projection from Definition 505.4. The refinement adds only the locality-cell consumer row L to the rows already displayed by the base carrier.

The GAP and forward-route obligations use Definition 505.2 and Definition 505.3. The forward-looking content is named by G, while F and C replay the route from observed regularity to local commitment.

The locality-cell restriction then reads the commitment through L. Since L is a consumer row rather than a new causal source, it can expose the commitment to observer-locality consumers only by replaying the same finite rows.

The non-escape clauses follow by inventory. The refined packet has no coordinate for a selected future record, global frame, universal law, hidden induction rule, or ambient causal primitive. □

Theorem 505.6 (CausalCommitment locality-cell handoff)**.** *Every observer-locality consumer of an accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ reads the causal commitment only through the displayed locality-cell row L. The handoff may expose the observed ledger, regularity ledger,* GAP *socket, forward-binding route, transports, continuations, provenance, and local naming row, but it cannot extract a selected future observation, a global observer frame, or an ambient causal primitive.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLocalityCellHandoff`

Proof. Begin with the consumer side of the refined carrier. The only row addressed to observer-locality consumption is L, so any such consumer first enters the finite locality-cell handoff.

Inside that handoff, the available reads are the rows already displayed by O, R, G, F, H, C, P, N. The row L does not add a further observation source or a global frame; it routes the same commitment through the locality-cell discipline.

Therefore the consumer receives a local commitment packet and nothing beyond it. A selected future observation, global observer frame, or ambient causal primitive has no carrier coordinate and no read route through L. □

Theorem 505.7 (CausalCommitment carrier visibility)**.** *For every accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$, the finite* BHist *rows $O, R, G, F, L, H, C, P, N$ are visible before any consumer reads* CausalCommitment$^{\uparrow}$*. The observed-history row, regularity row,* GAP *socket, forward-binding route, locality consumer row,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *row cannot be replaced by a hidden causal primitive.*

Proof. Project the forward-GAP carrier of Definition 505.4. It displays exactly $O, R, G, F, L, H, C, P, N$.

Use the GAP socket theorem and the forward-binding route theorem. The future-facing content is named by G, while F and C replay the observed-regularity route from O, R to the local commitment row.

The locality row L only hands the same carrier to locality-cell consumers. Since H, P, N transport, record, and name the displayed rows, the accepted carrier has no coordinate for a hidden causal primitive. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier visibility`

Theorem 505.8 (CausalCommitment classifier stability)**.** *The* $\mathsf{CausalCommitment}^{\uparrow}$ *classifier is stable under componentwise* hsame *transport of the displayed rows* $O, R, G, F, L, H, C, P, N$*. A consumer may replay the forward-GAP route through* Cont *and* Pkg *data, but it must preserve the finite* BHist *carrier, the* GAP *socket, the forward-binding route, the locality row, and the local* NameCert *row as separate coordinates.*

Proof. Begin with the componentwise transport row H. By construction it transports O, R, G, F, L together with package provenance and local naming data.

Apply Definition 505.2 and Definition 505.3. These theorems identify the forward-looking content with G and the observed-to-commitment route with F and C.

The forward-GAP obligations then force the locality consumer to pass through L. Thus classifier membership is preserved only by transporting the displayed rows componentwise; it is not preserved by collapsing the socket, route, locality row, or local NameCert into one ambient causal claim. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapCarrier transport`

Theorem 505.9 (CausalCommitment ledger non-escape)**.** *No accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *read over the forward-GAP carrier* $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ *can export a selected future observation, a universal empirical law, a hidden induction rule, a synchronized global frame, or an ambient causal primitive. The local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *row names only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, locality, GAP, and forward-binding rows.*

Proof. Start from carrier visibility. Every accepted read begins with the displayed rows $O, R, G, F, L, H, C, P, N$.

Use classifier stability to keep those rows separated under hsame transport and Cont replay. The GAP socket remains G, the forward route remains F, and the locality consumer remains L.

Finally apply the forward-GAP obligations. Their non-escape clause excludes selected future observations, global observer frames, universal empirical laws, hidden induction rules, and ambient causal primitives. Since no displayed row names one of those objects, the local ledger cannot export it. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLedgerNonescape`

Theorem 505.10 (CausalCommitment carrier obligation)**.** *Every accepted forward-GAP* $\mathsf{CausalCommitment}^{\uparrow}$ *carrier exposes exactly the displayed rows*

$$O, R, G, F, L, H, C, P, N.$$

Here O, R *are the finite observed and regularity ledgers,* G *is the* GAP *socket,* F *is the forward-binding route,* L *is the locality-cell consumer row,* H, C *are the componentwise transport and displayed replay rows,* P *is package provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *row. No other causal, observer, locality, or induction coordinate belongs to the accepted carrier.*

Proof. Project the forward-GAP carrier of Definition 505.4. It lists $O, R, G, F, L, H, C, P, N$ and describes L as the only locality-cell consumer handoff.

Apply the GAP socket theorem and the forward-binding route theorem. They account for the only forward-looking row G and the only route from observed regularity to commitment, namely F replayed by C.

Next use the forward-GAP obligations. Carrier exposure, GAP accountability, route stability, locality restriction, and non-escape are all stated over the same nine displayed rows.

Finally invoke ledger non-escape. Since selected future observations, global observer frames, universal empirical laws, hidden induction rules, and ambient causal primitives have no coordinate in the packet, the carrier obligation is exactly $O, R, G, F, L, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier obligation`

Theorem 505.11 (CausalCommitment locality transport)**.** *Every observer-locality transport of an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *forward-GAP packet factors through the locality-cell row L and the displayed* Cont *replay row C. A consumer may move the commitment into* $\mathsf{LocalityCell}^{\uparrow}$ *or* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *only by reading L after the route*

$$O, R \longrightarrow G \longrightarrow F \longrightarrow C$$

has been exposed on the same finite carrier.

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLocalityTransport`

Proof. Begin with Definition 505.5. Its locality-cell consumer restriction names L as the required handoff for every observer-locality consumer.

Use the forward-binding route theorem. The observed ledger and regularity ledger reach the commitment only through G and F, with C replaying the route inside the finite packet.

Apply classifier stability. Componentwise hsame transport keeps G, F, L and the replay row C separate, so an observer-indexed consumer cannot collapse the locality handoff into a global frame.

The locality-cell handoff theorem then supplies the endpoint. Since L is the only row that points to $\mathsf{LocalityCell}^{\uparrow}$ and $\mathsf{ObserverLocalityCell}^{\uparrow}$, every observer-locality transport factors through L and the displayed Cont replay. □

Theorem 505.12 (CausalCommitment ledger exactness)**.** *The* $\mathsf{CausalCommitment}^{\uparrow}$ *ledger excludes selected future observations, global observer frames, universal empirical laws, hidden induction rules, and ambient causal primitives. Its accepted readback is exactly the finite route through $O, R, G, F, L, H, C, P, N$, with no off-packet empirical law and no unrecorded causal constructor.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier ledger exactness`

Proof. Start from the carrier obligation. The accepted forward-GAP packet has only the nine displayed rows, so every consumer read must project one of those rows or use transport, replay, provenance, and local naming attached to them.

Apply Definition 505.2 and Definition 505.3. These keep the forward-looking content in G and the observed-to-commitment route in F and C.

Next use locality transport. Observer-locality consumption must pass through L, so it cannot introduce a synchronized global frame or selected future observation outside the finite carrier.

Finally apply Definition 505.9. The non-escape clause removes the remaining forbidden objects: universal empirical laws, hidden induction rules, and ambient causal primitives. Therefore the ledger readback is exactly the displayed finite route. □

Theorem 505.13 (CausalCommitment non-escape boundary)**.** *No consumer of an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet can extract a selected future observation, a universal empirical law, a hidden induction rule, a synchronized global frame, or an ambient causal primitive. Every consumer read factors through* O, R, G, F, H, C, P, N*.*

Proof. Inspect the carrier inventory. Its coordinates are the observed ledger, regularity ledger, GAP socket, forward-binding route, transport rows, continuation rows, package provenance, and local naming row.

The transport row H and the continuation row C preserve exactly those coordinates. They can replay the displayed route, but they do not introduce a new future record or a global frame.

The non-escape claim follows because the forbidden objects have no coordinate in the finite packet. A consumer route seeking one of them therefore has no projection target in $\mathsf{CausalCommitment}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentNonEscapeBoundary`

Theorem 505.14 (CausalCommitment NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *surface consists of carrier admission for* $K = (O, R, G, F, H, C, P, N)$*, observed-ledger exposure, regularity-ledger exposure,* GAP *socket accountability, forward-binding route accountability, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and the non-escape boundary for future observations, universal empirical laws, hidden induction rules, global frames, and ambient causal primitives.*

Proof. Carrier admission is projection from Definition 505.1. The packet exposes exactly the finite rows O, R, G, F, H, C, P, N.

Observed-ledger and regularity-ledger exposure are the first two projections. The GAP socket theorem then accounts for the only forward-looking coordinate in the carrier.

Forward-binding route accountability follows from Definition 505.3. The route from observed pattern to commitment is replayed by F and C, while H supplies componentwise stability and P records package provenance.

Finally apply the non-escape boundary. Since all consumer reads factor through the displayed rows, the local NameCert row cannot export selected future observations, universal empirical laws, hidden induction, global frames, or ambient causal primitives. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentNameCertObligations`

Definition 505.15 (CausalCommitment classifier row)**.** *The* $\mathsf{CausalCommitment}^{\uparrow}$ *classifier compares two finite packets*

$$K = (O, R, G, F, H, C, P, N) \qquad \textit{and} \qquad K' = (O', R', G', F', H', C', P', N')$$

component by component. A classifier read may identify the packets only when the observed-history ledgers O, O'*, regularity ledgers* R, R'*,* GAP *socket rows* G, G'*, forward-binding routes* F, F'*, transport rows* H, H'*, continuation rows* C, C'*, package provenance rows* P, P'*, and local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *rows* N, N' *are matched by the displayed* hsame*,* Cont*, and* Pkg *witnesses. It has no ambient equality row for causal commitment apart from these named coordinates.*

Lean4 (definition) `BEDC.Derived.CausalCommitmentUp.CausalCommitmentClassifierRow`

Theorem 505.16 (CausalCommitment carrier transport)**.** *For accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packets* $K = (O, R, G, F, H, C, P, N)$ *and* $K' = (O', R', G', F', H', C', P', N')$*, a componentwise* hsame *transport preserving the classifier row of Definition 505.15 preserves the observed ledger, regularity ledger,* GAP *row, forward route, continuation rows, provenance row, and local NameCert row. In particular, transport cannot replace* G *by an unrecorded forward rule, cannot replace* F *by a hidden route, and cannot export* N *outside the finite packet.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier transport`

Proof. Project both packets from Definition 505.1. The carrier displays exactly the rows O, R, G, F, H, C, P, N, and the classifier row asks for the corresponding displayed rows of K'.

Apply Definition 505.2 to the transported GAP coordinate. Since the only forward-looking row is G, componentwise hsame transport preserves the socket instead of manufacturing a separate future law.

Apply Definition 505.3 to the transported route coordinates. The route remains the displayed F replayed by the Cont rows in C, with P recording the same package provenance.

Finally use Definition 505.13. The transported local NameCert row still names only the finite packet, so no selected future record, global frame, hidden induction rule, universal empirical law, or ambient causal primitive is exported. □

Theorem 505.17 (CausalCommitment classifier coherence)**.** *The classifier row of Definition 505.15 cannot identify two accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packets whose* GAP *socket rows disagree, and it cannot identify two accepted packets whose forward-binding routes disagree. Any accepted classifier comparison must preserve both the socket that carries the forward-facing commitment and the route that returns it to the local NameCert row.*

Proof. Use the classifier row. Its comparison is componentwise, so a packet comparison has a separate coordinate for the GAP socket and a separate coordinate for the forward-binding route.

For the socket coordinate, apply Definition 505.2. The forward-looking content of a packet is named only by G. If G and G' disagree, the classifier has no other coordinate through which to identify the commitment-facing content.

For the route coordinate, apply Definition 505.3. The observed-regularity route must be the displayed F together with the Cont replay in C. If F and F' disagree, the comparison cannot repair that disagreement by appealing to a hidden route.

The non-escape boundary then rules out an ambient causal primitive that would identify the two packets outside their displayed coordinates. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentCarrier classifier coherence`

Theorem 505.18 (CausalCommitment observer-locality GAP readback)**.** *Every observer-locality readback of an accepted forward-GAP carrier* $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ *factors through the* GAP *socket row* G *and the locality-cell handoff row* L *before exposing the local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *row* N*. The readback may use the observed ledger* O*, regularity ledger* R*, forward-binding row* F*, transport row* H*, continuation replay* C*, and provenance row* P*, but it cannot expose* N *as an observer-locality certificate before* G *and* L *have both been replayed.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLocalityGapReadback`

Proof. Project the forward-GAP carrier of Definition 505.4. Its observer-facing rows are the displayed packet rows $O, R, G, F, L, H, C, P, N$, with G as the forward socket and L as the locality handoff.

Apply Definition 505.2 and Definition 505.3. These identify the forward-looking content with G and route the observed regularity through F and C before the local commitment row is available.

Now apply Definition 505.6. An observer-locality consumer reads the same commitment only through L, so the NameCert row N is exposed as observer-local only after both the GAP socket and the locality handoff have been replayed.

Finally use Definition 505.8 and Definition 505.9. They keep G, L, N as separate coordinates under transport and prevent the readback from replacing either row by a hidden causal primitive. □

Theorem 505.19 (CausalCommitment observer-cell non-escape)**.** *The observer-cell handoff of an accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ cannot export a selected future observation, global observer frame, or ambient causal primitive. Every observer-cell read remains a finite read of the observed ledger, regularity ledger,* GAP *socket, forward-binding route, locality row, transport, continuation replay, provenance, and local NameCert rows.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentObserverCellNonescape`

Proof. Begin with the locality-cell handoff theorem. Definition 505.6 says that observer-locality consumption enters only through L and reads only the rows displayed by the forward-GAP packet.

Use carrier visibility and the forward route. By Definition 505.7, the rows $O, R, G, F, L, H, C, P, N$ are visible before the read; by Definition 505.3, the forward commitment passes through G, F, C rather than a hidden induction rule.

The non-escape claim follows from Definition 505.9. Since the displayed packet has no coordinate for a selected future observation, global observer frame, or ambient causal primitive, the observer-cell handoff cannot export one. □

Theorem 505.20 (CausalCommitment forward-GAP scope package)**.** *For every accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$, carrier visibility, classifier stability, ledger exactness, and observer-locality handoff assemble into the dependency-ready forward-GAP scope package. Any downstream observer-locality consumer may depend on $O, R, G, F, L, H, C, P, N$ as one finite packet, with G carrying the forward-facing socket, L carrying observer-cell locality, and N naming only that displayed package.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapScopePackage`

Proof. First build the visible carrier surface. Apply Definition 505.4 and Definition 505.7 to expose exactly $O, R, G, F, L, H, C, P, N$.

Next attach the forward route and classifier stability. Apply Definition 505.2, Definition 505.3, and Definition 505.8; together they keep the socket G, route F, C, and transported packet rows distinct.

Then add ledger exactness and observer-cell locality. Use Definition 505.21, Definition 505.6, and Definition 505.18 to route downstream observer-locality reads through the same finite packet before the local NameCert row is exposed.

Finally apply Definition 505.19 and Definition 505.9. These close the package against selected future observations, global observer frames, ambient causal primitives, and hidden induction routes, so the displayed rows form the dependency-ready scope. □

Theorem 505.21 (CausalCommitment ledger route exactness)**.** *Every consumer route into an accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ factors through the displayed finite packet and then through the locality-cell handoff. The route first reads O, R, enters the* GAP

socket G, follows the forward-binding row F with the Cont *replay in C, records provenance in P, and reaches the local NameCert row N only through the locality row L when an observer-locality consumer is present.*

Proof. Project the refined carrier of Definition 505.4. Its consumer-visible coordinates are exactly $O, R, G, F, L, H, C, P, N$.

The observed side of the route is forced by Definition 505.3. A consumer begins with O, R, passes through G, and follows F together with the continuation replay in C.

The locality side is forced by Definition 505.6. An observer-locality consumer receives the commitment only through L, which hands off the same finite rows rather than adding a global frame or selected future record.

Finally apply Definition 505.13. Since every displayed route has already factored through the packet and the locality handoff, no consumer can bypass the ledger by exporting a hidden induction rule, universal empirical law, ambient causal primitive, selected future observation, or synchronized global frame. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLedgerRouteExactness`

Theorem 505.22 (CausalCommitment public forward-GAP export)**.** *Every public consumer of an accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ receives observed regularity only through the displayed route*

$$O, R, G, F, L, H, C, P, N.$$

The export reads the observed ledger O, regularity ledger R, GAP *socket G, forward-binding row F, locality row L, transport row H, continuation row C, provenance row P, and local NameCert row N; it exports no selected future observation, hidden induction rule, universal empirical law, synchronized global frame, or ambient causal primitive.*

Proof. First expose the forward-GAP carrier. Apply Definition 505.7 to make $O, R, G, F, L, H, C, P, N$ visible before a public consumer reads the commitment.

Next route the observed regularity through the forward boundary. Use Definition 505.2 and Definition 505.3; the consumer must read O, R, enter G, and replay the forward-binding row F through C.

Finally apply Definition 505.6, Definition 505.8, and Definition 505.21. These keep the locality row L, transport H, provenance P, and local name N on the same finite route, so the public export has no coordinate for a selected future observation or hidden induction rule. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentPublicForwardGapExport`

Theorem 505.23 (CausalCommitment locality consumer exactness)**.** *For every accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$, an observer-locality consumer reads the commitment exactly through the locality row L after the observed regularity has passed through O, R, G, F. The consumer may transport, replay, package, and name the read only by H, C, P, N, and it cannot replace L by a selected future observation, a global observer frame, or a hidden causal-law row.*

Proof. Begin with the locality handoff. Definition 505.6 identifies L as the only observer-locality consumer row in the forward-GAP carrier.

Then attach the forward route. Definition 505.3 and Definition 505.21 force the consumer to read O, R, pass through G, and follow F with the displayed continuation replay before N is available.

Use classifier stability for exactness. Definition 505.8 preserves $O, R, G, F, L, H, C, P, N$ as separate coordinates under transport, so the observer-locality read cannot collapse the locality row into a global frame or hidden causal law. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLocalityConsumerExactness`

Theorem 505.24 (CausalCommitment observed regularity nonchoice)**.** *Every public read of observed regularity from an accepted forward-GAP carrier $K_{\mathrm{fg}} = (O, R, G, F, L, H, C, P, N)$ is a finite read of the displayed rows $O, R, G, F, L, H, C, P, N$. The read passes through the* GAP *socket and the locality row before it is public, so it does not choose a future observation, add a hidden induction rule, or supply an unledgered regularity principle.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentObservedRegularityNonchoice`

Proof. Project the observed side first. Definition 505.2 says that O and R are finite ledgers and that forward-facing content is named only by G.

Now compose the public route. Apply Definition 505.22 and Definition 505.23. The public consumer reaches the regularity claim only after G, F, L, H, C, P, N have carried the observed ledger.

Finally apply Definition 505.9 and Definition 505.21. Since the displayed route has no coordinate for selected futures, hidden induction, or an unledgered regularity principle, the observed regularity read is nonchoice data. □

Theorem 505.25 (CausalCommitment locality obligations)**.** *Every accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *forward-GAP carrier binds its commitment row to* $\mathsf{LocalityCell}^{\uparrow}$ *and* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *through the displayed locality row L. The observer-facing commitment may be transported only through the finite rows $O, R, G, F, L, H, C, P, N$; it has no hidden causal-law row, selected future observation, synchronized global frame, or unledgered agency primitive.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentLocalityObligations`

Proof. Project the forward-GAP carrier of Definition 505.4. The row L is the only locality-facing coordinate added to the base packet O, R, G, F, H, C, P, N.

Apply Definition 505.6. It routes the commitment through the locality-cell and observer-locality surfaces without adding a global observer frame or selected future record.

Use Definition 505.21. Every consumer route must read O, R, pass through G and F, replay through C, record provenance in P, and reach N only through the displayed locality row when an observer-locality consumer is present.

Finally apply Definition 505.9. Since hidden causal laws, selected future observations, synchronized global frames, and unledgered primitives have no coordinate in the packet, the locality obligations are exactly the displayed rows. □

Theorem 505.26 (CausalCommitment observer-locality handoff)**.** *An observer-facing read of* $\mathsf{CausalCommitment}^{\uparrow}$ *factors through* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *only after the carrier has displayed the* BHist *rows O, R, G, F, L, the* hsame *row H, the* Cont *row C, the* Pkg *row P, and the local* NameCert *row N. No observer-facing route may bypass those rows by appealing to an ambient causal law, a selected future observation, or a primitive agency coordinate.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentObserverLocalityHandoff`

Proof. Start with Definition 505.7. It makes the rows $O, R, G, F, L, H, C, P, N$ visible before any consumer reads the refined packet.

Apply Definition 505.3. The commitment-facing content enters through G and returns along F with the displayed continuation replay in C.

Use Definition 505.6. The observer-locality consumer receives the commitment through L, not through a global frame or selected future record.

Finish with Definition 505.13. The route has no coordinate for hidden causal laws, unledgered agency primitives, universal empirical laws, or selected future observations, so the observer-locality handoff remains finite and local. □

Theorem 505.27 (CausalCommitment NameCert ledger)**.** *The local* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *ledger for an accepted forward-GAP carrier is exhausted by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, locality, GAP, and forward-binding rows* $O, R, G, F, L, H, C, P, N$*. It records observer-facing commitments only as ledgered transport through those rows and excludes hidden causal laws, unledgered agency primitives, selected future observations, and synchronized global frames.*

Proof. Begin with Definition 505.14. The local NameCert surface already records carrier admission, observed-ledger exposure, regularity-ledger exposure, GAP accountability, forward-binding route accountability, transport, replay, provenance, and non-escape.

Add the locality row using Definition 505.5. The refined carrier exposes L as the consumer-facing handoff and leaves all other coordinates unchanged.

Apply Definition 505.25 and Definition 505.26. These obligations bind locality and observer-locality reads to $O, R, G, F, L, H, C, P, N$.

Finally use Definition 505.9. Anything outside the displayed ledger has no NameCert coordinate, so no hidden causal law or agency primitive can be exported from the packet. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentNameCertLedger`

Theorem 505.28 (CausalCommitment GAP locality exactness)**.** *Every public causal read from an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet must preserve the displayed* GAP *socket row* G *and the locality row* L *before any observer-facing commitment is exported. The public read may transport those rows through* H*, replay them through* C*, package them through* P*, and name them through* N*, but it may not replace* G *by a hidden induction rule or* L *by a synchronized global frame.*

Proof. Start with the GAP socket. Definition 505.2 makes G the only forward-facing socket row available to the carrier, so a public read cannot name a hidden causal primitive in place of the displayed socket.

Attach the observer-locality handoff. By Definition 505.6, the locality consumer receives the commitment through L and not through an ambient observer frame.

Use classifier stability before export. Definition 505.8 preserves $O, R, G, F, L, H, C, P, N$ as separate rows under componentwise transport. Therefore the exported observer-facing commitment still carries the same GAP socket and locality row.

Finally read the ledger of Definition 505.27. Its exactness excludes hidden causal laws, selected future observations, synchronized global frames, and unledgered primitives from the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentGapLocalityExactness`

Theorem 505.29 (CausalCommitment forward GAP observer route)**.** *The route from observed regularity to observer-locality handoff in an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet factors through*

$$O, R, G, F, L, H, C, P, N.$$

Thus the observed-history row, regularity row, GAP socket, forward-binding row, locality row, transport, continuation replay, provenance, and local name are all crossed before an observer-facing commitment is available.

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapObserverRoute`

Proof. Begin with the observed and regularity rows. Definition 505.3 routes the observed pattern through O, R into the forward-facing content named by G and the displayed forward-binding row F.

Next attach the GAP socket and locality handoff. Definition 505.2 identifies the public socket row as G, while Definition 505.6 routes the observer-locality consumer through L.

It remains to preserve the support rows. Definition 505.8 keeps H, C, P, N attached to the same packet under transport, and Definition 505.27 records that no additional observer-facing row is exported. □

Theorem 505.30 (CausalCommitment kernel row exhaustion)**.** *The accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet exports only* BHist*,* GAP*,* hsame*,* Cont*,* Pkg*, locality-cell, observer-locality, and local* NameCert *rows. It exports no hidden causal law, selected future observation, synchronized global frame, universal empirical law, or unledgered agency primitive.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentKernelRowExhaustion`

Proof. Read the packet rows first. Definition 505.2 and Definition 505.3 expose the BHist observed and regularity rows, the GAP socket, and the forward route through O, R, G, F.

Add locality and observer access. Definition 505.6 supplies the locality-cell and observer-locality row L, while Definition 505.8 preserves the componentwise hsame, displayed Cont, and Pkg rows.

Finally apply Definition 505.27. The local NameCert ledger is exhausted by the displayed rows, so any attempted export outside BHist, GAP, transport, replay, provenance, locality, observer-locality, and local naming exits the packet. □

Theorem 505.31 (CausalCommitment GAP locality adjunction)**.** *For an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet, the GAP socket row G and the locality row L form the only admissible route from observed regularity to observer-locality handoff. Any observer-facing commitment read must pass from the observed and regularity rows through G, then through the forward-binding row F, and finally through L before the transport, replay, provenance, and local naming rows may expose it.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentGapLocalityAdjunction`

Proof. Start with the carrier and GAP socket. The carrier of Definition 505.1 displays the rows $O, R, G, F, L, H, C, P, N$, and Definition 505.2 makes G the public GAP row available to the packet.

Next use the forward binding route. By Definition 505.3, the observed regularity content enters the commitment-facing route through O, R, G, F and does not pass through an unledgered future selector.

Finally attach observer locality. Definition 505.6 routes the consumer through L, while Definition 505.13 excludes hidden causal laws, synchronized global frames, and selected future observations. Thus the only admissible observed-regularity to observer-locality route is the displayed G, F, L handoff inside $O, R, G, F, L, H, C, P, N$. □

Theorem 505.32 (CausalCommitment observer row factorization)**.** *Every observer-facing read of an accepted* $\mathsf{CausalCommitment}^{\uparrow}$ *packet factors through*

$$O, R, G, F, L, H, C, P, N.$$

The observer row, regularity row, GAP socket, forward-binding row, locality handoff, transport, continuation replay, provenance, and local NameCert row are all required before the read can be exported to an observer-locality consumer.

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentObserverRowFactorization`

Proof. Begin with the route already exposed by Definition 505.29. It identifies the observer-facing route with the displayed rows $O, R, G, F, L, H, C, P, N$.

Use classifier stability to keep the rows separate during transport. Definition 505.8 preserves the same carrier coordinates under the componentwise hsame and displayed Cont replay rows, so the observer read cannot collapse G, F, or L into a single hidden primitive.

The ledger then closes the factorization. Definition 505.27 and Definition 505.9 say that no row outside $O, R, G, F, L, H, C, P, N$ has a local NameCert coordinate. Hence every accepted observer-facing read factors through exactly those rows. □

Theorem 505.33 (CausalCommitment forward GAP NameCert export)**.** *The forward-GAP obligation surface of* $\mathsf{CausalCommitment}^{\uparrow}$ *exports a local NameCert row only after the GAP-locality adjunction and observer-row factorization are both satisfied. The exported certificate packages the carrier, GAP socket, forward-binding route, locality handoff, classifier stability, and ledger non-escape rows, and it exports no hidden causal primitive or selected future observation.*

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentForwardGapNameCertExport`

Proof. First package the carrier and locality route. Apply Definition 505.31; it gives the only admissible route from observed regularity through the GAP socket and forward binding row to the observer-locality handoff.

Next package the observer-facing read. Apply Definition 505.32; it requires the complete row sequence $O, R, G, F, L, H, C, P, N$ before any observer-locality consumer receives the commitment.

It remains to read the local NameCert ledger. Definition 505.27 supplies the local naming row, while Definition 505.9 and Definition 505.13 exclude hidden causal laws, unledgered agency primitives, synchronized global frames, universal empirical laws, and selected future observations. Therefore the exported NameCert is exactly the displayed forward-GAP obligation surface. □

Theorem 505.34 (CausalCommitment standard bridge surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$ *export of Definition 505.33 factors through the standard bridge surface by displaying exactly the observed-history ledger O, regularity ledger R,* GAP *socket G, forward-binding route F, locality handoff L, componentwise transport row H, displayed replay row C, provenance row P, and local name N. The bridge exports no selected future observation, ambient causal primitive, hidden induction rule, synchronized global frame, or universal empirical law.*

Proof. Start from the public forward-GAP export. It already packages the carrier, GAP socket, forward-binding route, locality handoff, classifier stability, and ledger non-escape rows as the local NameCert surface.

Read the rows through the bridge surface. The observed-history and regularity ledgers supply O, R, the GAP socket and forward-binding route supply G, F, the locality handoff supplies L, and the transport, replay, provenance, and local naming rows supply H, C, P, N.

Finally apply the ledger non-escape boundary. The forbidden causal objects have no coordinate in $O, R, G, F, L, H, C, P, N$, so the standard bridge surface receives only the displayed public NameCert export and no selected future observation or ambient causal primitive. □

Lean4 ✓ `BEDC.Derived.CausalCommitmentUp.CausalCommitmentStandardBridgeSurface`

Closure status for $\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}$.

Constructive story: A $\mathsf{CausalCommitment}^{\uparrow}$ packet is generated from a finite observed-history ledger, an observed regularity ledger, a displayed GAP socket, a forward-binding route, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 505.34 binds the standard bridge surface as the mature finite public package over observed-history, regularity, GAP socket, forward-binding, locality, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CausalCommitment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge status, selected future observation, universal empirical law, hidden induction rule, synchronized global frame, or ambient causal primitive.

Upgrade path: Lean verification, axiom-purity certification, and bridge checking remain separate formal-axis work for the standard bridge surface.

Chapter 506

A Concrete Naming Certificate for RussellSelfMembershipBoundary$^{\uparrow}$

The RussellSelfMembershipBoundary$^{\uparrow}$ surface packages the Russell self-membership discipline of chapter 9457 as finite NameCert data. It reads a contextual class expression only through the classifier that introduces it, records the attempted self-membership query as a separate row, and rejects the promotion of that expression into an object that could contain or fail to contain itself.

The packet is a boundary certificate. It does not create a universal class object, does not admit an unstratified membership predicate, and does not turn contextual readability into object-level membership.

The attempted self-membership query is read against the classifier-membership discipline of chapter 476: TypeClassifierMembership$^{\uparrow}$ supplies the finite T, J, E, R, H, C, P, N membership packet, while RussellSelfMembershipBoundary$^{\uparrow}$ refuses the additional object-level self-membership coordinate.

It consumes the replacement-and-non-export ledger of chapter 515 when rejecting object-level self-membership reads: both packets separate contextual classifier use from forbidden promotion into host-level existence, quotient, or proposition-extensionality rows.

Definition 506.1 (Russell self-membership boundary carrier)**.** *A* RussellSelfMembershipBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, Q, A, S, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the contextual-class source row: it records the expression read as a classifier-local class description, not as a kernel object;*
2. *Q is the attempted self-membership query row asking whether the contextual expression named by K can be read as a member of itself;*
3. *A is the negated-presence row saying that no accepted packet coordinate contains an object-level membership field for Q;*
4. *S is the stratification row separating the contextual classifier level of K from any object-level membership read;*
5. *G records the* Ask, Pkg, *and* Gap *provenance that identifies the query source, the package boundary, and the blocked passage from contextual reading to object membership;*

6. *H stores componentwise* hsame *transports for the contextual source, attempted query, negated-presence row, stratification row, provenance row, continuation routes, and local naming row;*
7. *C stores the displayed* Cont *routes by which consumers may inspect K, Q, A, S, and G without adding a new membership coordinate;*
8. *P is the* Pkg *provenance over* BHist, Ask, Gap, hsame, Cont, *contextual classifier rows, and the surrounding naming discipline;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RussellSelfMembershipBoundary}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. Its classifier-visible fields are the contextual source, the attempted query, the negated-presence row, the stratification row, the Ask/Pkg/Gap provenance, transport, continuation, package, and local NameCert rows.

Theorem 506.2 (Russell self-membership rejection)**.** *If $R = (K, Q, A, S, G, H, C, P, N)$ is an accepted* $\mathsf{RussellSelfMembershipBoundary}^{\uparrow}$ *carrier, then no classifier-visible coordinate of R promotes the contextual class expression recorded by K into an object containing itself or into an object not containing itself.*

Every consumer read of the attempted self-membership query Q factors through the negated-presence row A, the stratification row S, the Ask/Pkg/Gap provenance G, the componentwise hsame *transports H, the displayed* Cont *routes C, the package row P, and the local* NameCert *row N.*

Proof. Project the carrier rows of Definition 506.1. The source row K is contextual classifier data, while Q is only the attempted query against that contextual reading.

The negated-presence row A records that the packet has no classifier-visible object-membership coordinate for Q. The stratification row S records that the contextual level of K is not the object level needed by a self-membership assertion.

Route any consumer read through G, H, C, P, and N. These rows expose Ask/Pkg/Gap provenance, transport, continuation, and local naming data, but they do not add a membership field.

Thus the attempted self-reference remains a rejected query at the NameCert boundary. The packet can report the blocked route, but it cannot manufacture either side of the object-level self-membership alternative. □

Theorem 506.3 (Russell boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RussellSelfMembershipBoundary}^{\uparrow}}$ *surface has five finite obligations:*

1. *carrier admission: every accepted object exposes the tuple $R = (K, Q, A, S, G, H, C, P, N)$ of Definition 506.1;*
2. *contextual classifier reading: the row K is consumed only as a classifier-local class description and never as an admitted object containing members;*
3. *self-query rejection: the attempted query Q is routed through the negated-presence row A and cannot produce a membership or nonmembership field;*
4. *stratified transport: the row S and the componentwise* hsame *ledger H preserve the separation between contextual classifier reads and object-level membership reads;*
5. *ledger boundary: every downstream consumer passes through the displayed* Ask, Pkg, Gap, Cont, *and local* NameCert *rows, with no universal class object and no unstratified membership predicate.*

Lean4 ✓ `BEDC.Derived.RussellSelfMembershipBoundaryUp.RussellSelfMembershipBoundaryTasteGate single carrier alignment`

Proof. Carrier admission is projection from the finite tuple in Definition 506.1. Each coordinate is a displayed BHist row or a finite provenance row named by the tuple.

Contextual reading is enforced by the placement of K and S. The expression appears in the contextual classifier row, and the stratification row records that this reading is not an object-level membership carrier.

Self-query rejection is the role of Q together with A. The query is present as an attempted read, while A records the absence of a classifier-visible coordinate that could answer it as membership or nonmembership of an object by itself.

Finally route downstream use through G, H, C, P, and N. The ledger exposes Ask/Pkg/Gap provenance, transport, continuation, package, and local naming rows only, so the NameCert surface contains no universal class object or unstratified membership predicate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RussellSelfMembershipBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RussellSelfMembershipBoundary}^{\uparrow}$ packet is generated from a contextual-class source row, an attempted self-membership query row, a negated-presence row, a stratification row, Ask/Pkg/Gap provenance, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RussellSelfMembershipBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 506.1, Definition 506.2, and Definition 506.3 bind the finite carrier, self-query rejection row, stratified transport route, and local NameCert obligations for contextual class readings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RussellSelfMembershipBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge status, universal class object, unstratified membership predicate, or object-level Russell class.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for contextual classifier reading, self-query rejection, stratified transport, and ledger-boundary non-escape.

Chapter 507

A Concrete Naming Certificate for $\mathsf{StableNegationBoundary}^{\uparrow}$

The $\mathsf{StableNegationBoundary}^{\uparrow}$ surface turns the reasoning-by-contradiction boundary of chapter 9458 into a finite naming packet. It records the certificate row being inspected, the permitted refutation route, the local stability data for decidable refutations, and the ledger that blocks certificate-wide truth branching. The surface is a NameCert boundary for stable negation; it is not a proof of excluded middle over naming certificates.

The permitted refutation route and optional decision row are consumed from the decidable-refutation and refutation-witness sibling surfaces of chapter 493 and chapter 511. Those surfaces supply the local refutation and witness-or-decision rows used by $A, R, D, T, L, H, C, P, N$.

The stable-negation packet also consumes the permitted refutation route of chapter 503 and the constructive route boundary of chapter 501, then adds the optional decidability row and certificate-truth exclusion ledger.

Definition 507.1 (StableNegationBoundary carrier)**.** *A* $\mathsf{StableNegationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R, D, T, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition row being tested inside a fixed local certificate surface;*
2. *R is the permitted refutation route from an assumed witness of A to $\bot$, read only through* Cont, hsame, msame, *and the local certificate rows;*
3. *D is the optional decidability row for A, present exactly when the packet carries a constructive decision for the same proposition row;*
4. *T is the stable-negation classifier row, recording that $\neg A$ may be transported and that $\neg\neg A$ may be consumed only through D;*
5. *L is the truth-branch exclusion ledger, blocking excluded-middle branching over the carrier of all naming certificates and blocking witness extraction from a contradiction without displayed data;*
6. *H stores componentwise* hsame *transports for A, R, D, T, L and the local naming rows;*
7. *C stores the* Cont *routes for replaying the refutation, consuming the optional decision row, transporting the stable-negation classifier, and reading the exclusion ledger;*

8. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *local proof rows, and the surrounding* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{StableNegationBoundary}^{\uparrow}}$ *row.*

The carrier has no row for host truth, no branch over the carrier of all naming certificates, and no witness selector for an existential obtained by contradiction.

Lean4 ✓ `BEDC.Derived.StableNegationBoundaryUp.StableNegationBoundaryTasteGate single carrier alignment`

Theorem 507.2 (StableNegationBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StableNegationBoundary}^{\uparrow}}$ *surface for Definition 507.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet $S = (A, R, D, T, L, H, C, P, N)$;*

2. *classifier soundness for the stable-negation row T, comparing only the displayed proposition, refutation, optional decision, exclusion ledger, transport, continuation, package, and naming rows;*

3. *refutation stability, saying that a proof of $\neg A$ transported through H and replayed through C remains the same local certificate-level negation;*

4. *decidable double-negation use, saying that a $\neg\neg A$ consumer must pass through the displayed decision row D;*

5. *truth-branch non-escape, saying that L blocks certificate-wide excluded middle, witness extraction by contradiction, and host-truth reads.*

Proof. Carrier admission is projection from the displayed packet. The proposition row, refutation route, optional decision row, classifier row, exclusion ledger, transports, continuations, package row, and local name are all visible coordinates.

Classifier soundness and refutation stability are componentwise. The route R is replayed only after A and the local certificate rows have been transported through H, so a transported proof of $\neg A$ remains inside the same finite surface.

For double negation, the row D is the only coordinate that supplies a constructive decision for A. A consumer of $\neg\neg A$ therefore either reads D and stays inside the packet or is blocked by the exclusion ledger L.

Truth-branch non-escape is the absence of any carrier-wide decision coordinate. The ledger L records the forbidden routes named in Definition 9458.5, and the remaining rows provide no host truth predicate or witness selector. □

Theorem 507.3 (StableNegationBoundary decidable refutation stability)**.** *For an accepted* StableNegationBoundar *carrier $S = (A, R, D, T, L, H, C, P, N)$, every permitted refutation of A remains stable under the displayed* hsame *transports and* Cont *replay routes. If a consumer also requests $\neg\neg A \Rightarrow A$, the request is accepted only through the same displayed decision row D.*

Lean4 ✓ `BEDC.Derived.StableNegationBoundaryUp.StableNegationBoundaryCarrier decidable refutation stability`

Proof. Read the refutation route R from the carrier and transport the proposition row A through H. Replaying R through the displayed continuation routes in C again sends an assumed witness of the transported proposition to $\bot$.

Now inspect the double-negation request. The NameCert obligations name D as the sole route that turns $\neg\neg A$ into A, so the request cannot be served by an ambient law of excluded middle.

The packet inventory then gives stability. Every accepted route passes through $A, R, D, T, L, H, C, P, N$, and no row supplies a certificate-wide truth branch. □

Theorem 507.4 (StableNegationBoundary certificate-truth non-escape)**.** *No accepted* StableNegationBoundary$^{\uparrow}$ *carrier exports a decision family for predicates over the carrier of all naming certificates. Its negative information is exhausted by the local refutation route R, the optional decision row D, the classifier row T, and the exclusion ledger L.*

Lean4 ✓ `BEDC.Derived.StableNegationBoundaryUp.StableNegationBoundaryCarrier certificate truth non escape`

Proof. Assume a consumer reads an accepted carrier. The only available proposition-level rows are A, R, D, T, L together with their transports and continuation routes.

A decision family over all naming certificates would require a coordinate that ranges over certificate-indexed predicates. The carrier has no such coordinate; it contains only a fixed local proposition row and its local refutation or decision data.

The ledger L names the blocked routes. Since every export is routed through C and P, a consumer cannot replace the local negative information with host truth, certificate-wide excluded middle, or witness extraction by contradiction. □

Theorem 507.5 (StableNegationBoundary ledger exhaustion)**.** *Every accepted stable-negation read is exhausted by the rows $A, R, D, T, L, H, C, P, N$ of Definition 507.1. No downstream consumer may add an external truth predicate, a global excluded-middle branch, or an existential witness selected only from a contradiction.*

Lean4 ✓ `BEDC.Derived.StableNegationBoundaryUp.StableNegationBoundaryCarrier ledger exhaustion`

Proof. Start with an accepted stable-negation read. Carrier admission sends it to the finite packet $S = (A, R, D, T, L, H, C, P, N)$.

The refutation and decidability rows account for the permitted routes: R supplies $\neg A$, and D supplies the only accepted route from $\neg\neg A$ to A. The classifier row T and the transports H, C preserve these routes componentwise.

It remains to account for forbidden exits. The ledger L blocks the certificate-wide truth branch and witness-selection routes, while the package and naming rows only record provenance. Thus the displayed packet exhausts the read. □

Closure status for NameCert$_{\text{StableNegationBoundary}\uparrow}$**.**

Constructive story: A StableNegationBoundary$^{\uparrow}$ packet is generated from a local proposition row, a permitted refutation route, an optional decision row, a stable-negation classifier, a truth-branch exclusion ledger, displayed transports, continuation routes, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\text{StableNegationBoundary}\uparrow}$, seed)

Scope closed: Definition 507.1, Definition 507.2, Definition 507.3, Definition 507.4, and Definition 507.5 record the finite carrier, classifier, ledger, and non-escape obligations for stable negation.

Formal verification: FormalStatus(NameCert$_{\text{StableNegationBoundary}\uparrow}$, formalTarget)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.stableNegationBoundaryDeltaLedger

Lean target: `BEDC.Derived.StableNegationBoundaryUp.StableNegationBoundaryCarrier\_namecert\_obligations`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close excluded middle over naming certificates, witness extraction by contradiction, host truth, or a checked Lean certificate.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked obligations for classifier soundness, refutation stability, decidable double-negation use, certificate-truth non-escape, and ledger exhaustion.

Chapter 508

A Concrete Naming Certificate for $\mathsf{FiniteRefutationBoundary}^{\uparrow}$

The $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ surface realizes the contradiction-boundary discipline of chapter 9458 as a finite NameCert packet. It records the ground-loop refutation route that may be replayed, the decidability row that may license double-negation consumption, and the exclusion rows that block meta-level escape. The packet is intentionally finite: every accepted read is through BHist, msame, hsame, Cont, a fixed local NameCert, and a displayed route to falsity.

The permitted refutation row is supplied by the ground-loop carrier of chapter 510: $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ consumes its displayed assumption row, falsity-producing Cont route, hsame transports, Pkg provenance, and local NameCert row before adding the optional decision row and meta-escape exclusion ledger.

Definition 508.1 (Finite refutation boundary carrier)**.** *A* $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, R, D, E, H, C, P, N, \bot_R)$$

with the following displayed rows:

1. *A is the proposition row whose contradiction boundary is being inspected;*
2. *R is the permitted refutation row, read as an assumed witness of A followed by a displayed route to falsity;*
3. *D is the optional constructive decision row for A, present exactly when the packet permits double-negation consumption for A;*
4. *E is the meta-escape exclusion ledger, with separate entries for unwitnessed existential extraction, double-negation elimination without D, and excluded-middle branching over naming certificates;*
5. *H records componentwise* hsame *transports for proposition rows, refutation rows, optional decision rows, exclusion rows, package rows, and local naming rows;*
6. *C records the displayed* Cont *routes that read A, replay R, consume D when present, inspect E, and return the result to the local ledger;*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *local proof rows, exclusion rows, and the surrounding certificate rows;*
8. *N is the fixed local* $\mathsf{NameCert}_{\mathsf{FiniteRefutationBoundary}^{\uparrow}}$ *ledger through which all rows are addressed;*

9. *$\bot_R$ is the displayed falsity route witnessing that an assumed A-row reaches $\bot$ using only R, H, C, P, N and the kernel sameness primitives.*

The carrier contains no host truth predicate, no classifier over the carrier of all naming certificates, no selected existential witness obtained by contradiction, and no excluded-middle oracle beyond a displayed decision row D for the single proposition row A.

Lean4 ✓ `BEDC.Derived.FiniteRefutationBoundaryUp.FiniteRefutationBoundaryTasteGate single carrier alignment`

Theorem 508.2 (Finite refutation boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteRefutationBoundary}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the finite packet $B = (A, R, D, E, H, C, P, N, \bot_R)$ of Definition 508.1;*
2. *classifier soundness, comparing accepted packets only through the displayed proposition, refutation, decision, exclusion, transport, continuation, provenance, naming, and falsity-route rows;*
3. *ground-loop replay, saying that $\bot_R$ remains a route from an assumed A-witness to $\bot$ under the stored* hsame *transports and* Cont *routes;*
4. *decision discipline, saying that $\neg\neg A$ may be consumed only when D supplies a constructive branch for the same proposition row A;*
5. *escape exclusion, saying that no consumer may export unwitnessed existence, non-decidable double-negation elimination, or certificate-level excluded middle from the packet.*

Lean4 ✓ `BEDC.Derived.FiniteRefutationBoundaryUp.FiniteRefutationBoundaryNamecertObligations`

Proof. Carrier admission is projection from Definition 508.1. Every named coordinate is a finite BHist row or a route displayed over those rows.

Classifier soundness is componentwise. The comparison can inspect $A, R, D, E, H, C, P, N, \bot_R$ and their transported counterparts, and it has no coordinate for host truth, proof search, or a classifier over all naming certificates.

For ground-loop replay, transport the assumed A-witness and the row R along H, then replay the displayed route in C. The terminal step is the falsity route $\bot_R$, so the transported packet still proves the certificate-level negation of A.

For decision discipline, the row D is the only allowed source of a constructive branch for A. When D is absent, the exclusion ledger E blocks the non-decidable double-negation schema of Definition 9458.5; when D is present, the branch is consumed inside the same packet and does not become a global oracle.

Finally, escape exclusion follows by inspecting the carrier. The packet has no witness selector for existential-by-contradiction, no double-negation eliminator independent of D, and no branch over the carrier of all naming certificates, so the local NameCert obligations remain inside the closed ground loop. □

Theorem 508.3 (Ground-loop soundness for permitted refutations)**.** *For every accepted* $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ *packet $B = (A, R, D, E, H, C, P, N, \bot_R)$, the permitted refutation read from B proves only the local certificate-level negation of the displayed proposition row A. It factors through* BHist, msame, hsame, Cont, *the fixed ledger N, and the falsity route $\bot_R$, and it does not decide any proposition row not explicitly carried by B.*

Proof. Read the carrier through Definition 508.1. The only proposition being refuted is A, and the only route from an assumed witness to falsity is $\bot_R$ as replayed through R, H, C, P, N.

The kernel primitives available to the replay are msame, hsame, and Cont over the finite history rows. Transport changes representatives inside the displayed packet, but it never creates a new proposition row or a new classifier.

Thus the public consequence is the local negation of A. Since no coordinate names an ambient decision procedure, the packet cannot decide any row outside the displayed finite carrier. □

Theorem 508.4 (Meta-escape exclusion)**.** *An accepted* $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ *packet cannot export any of the three forbidden contradiction moves of Definition 9458.5: it cannot extract a witness from* $\neg\forall h : \mathsf{Hist}, \neg Q(h)$ *without displaying that witness, it cannot eliminate* $\neg\neg A$ *without the row* D *for the same proposition row, and it cannot supply excluded middle for predicates over the carrier of all naming certificates.*

Proof. For unwitnessed existence, the carrier has no selector coordinate. Any existential witness exported by the packet must be one of its displayed finite rows or be constructed by a displayed Cont route from those rows.

For non-decidable double-negation elimination, inspect D and E. The decision row D is the only positive branch source for A, while E records that the schema is refused when D is absent.

For certificate-level excluded middle, inspect the naming ledger N. It is fixed locally to this packet and does not range over the carrier of all naming certificates, so it cannot produce a branch family for arbitrary certificate predicates.

Therefore each forbidden move is blocked by the absence of a carrier coordinate and by the explicit exclusion ledger. □

Theorem 508.5 (Finite refutation boundary concretizes the contradiction-boundary vision)**.** *The* $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ *carrier of Definition 508.1, together with the obligations of Definition 508.2, realizes the boundary stated in chapter 9458: permitted refutations are exactly those whose falsity route remains in the fixed ground-loop packet, while the forbidden patterns are refused because they require a meta-level branch or witness not displayed by that packet.*

Proof. The permitted side is Definition 508.3. A refutation route is accepted only when it is the displayed route from A through R, H, C, P, N to $\bot_R$.

The refused side is Definition 508.4. The three forbidden patterns all require a coordinate absent from the finite packet: an unwitnessed existential selector, a double-negation eliminator without D, or a branch over naming certificates.

Combining these two facts gives the advertised boundary. The finite carrier supplies the concrete BEDC object, the local NameCert obligations supply the admissibility surface, and consumers receive a refutation route without receiving a meta-level truth predicate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRefutationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRefutationBoundary}^{\uparrow}$ packet is generated from a proposition row, a permitted refutation row, an optional constructive decision row, meta-escape exclusions, componentwise transports, displayed continuation routes, package provenance, a fixed local NameCert ledger, and a falsity route.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRefutationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 508.1, Definition 508.2, Definition 508.3, Definition 508.4, and Definition 508.5 bind the finite carrier, obligation surface, ground-loop soundness, meta-escape exclusion, and vision realization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRefutationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close classical excluded middle, a global truth predicate over naming certificates, existential witness selection by contradiction, or a checked Lean certificate.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for classifier soundness, ground-loop replay, decision discipline, escape exclusion, and the public vision realization.

Chapter 509

A Concrete Naming Certificate for $\mathsf{LocalityCell}^{\uparrow}$

The $\mathsf{LocalityCell}^{\uparrow}$ surface packages the locality-cell reading of Definition 9420.38 as finite BEDC data. It names two observer histories, two displayed event records, bilateral GAP commitments, componentwise transport, continuation routes, package provenance, and a local naming ledger. The packet supports inter-observer event coordination without adding a global frame, a shared clock, or a god's-eye event object outside the two recorded histories.

Definition 509.1 (LocalityCell carrier)**.** *A* $\mathsf{LocalityCell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$$

with the following visible rows:

1. *A and B are the two displayed observer* BHist *rows;*
2. *r_A is the event-record row read from observer history A;*
3. *r_B is the event-record row read from observer history B;*
4. *g_A is the* GAP *commitment on the A side that treats r_B as a coherent counterpart of r_A;*
5. *g_B is the* GAP *commitment on the B side that treats r_A as a coherent counterpart of r_B;*
6. *H records componentwise* hsame *transport for the two histories, the two event-record rows, the two* GAP *rows, package provenance, and local naming row;*
7. *C records the displayed* Cont *routes by which the two event records and bilateral* GAP *commitments are read from the two observer histories;*
8. *P is the* Pkg *provenance naming* BHist, Hist, hsame, Cont, Pkg, NameCert, *and the locality-cell source boundary;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalityCell}^{\uparrow}}$ *ledger.*

The carrier is exactly this bilateral packet. It contains no global event coordinate, no shared time index, no selected synchronization map, and no observer-independent space-time object.

Definition 509.2 (LocalityCell classifier)**.** *The* $\mathsf{LocalityCell}^{\uparrow}$ *classifier relates two accepted carriers*

$$L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$$

and

$$L' = (A', B', r'_A, r'_B, g'_A, g'_B, H', C', P', N')$$

exactly when the displayed hsame *rows transport* A *to* A', B *to* B', r_A *to* r'_A, r_B *to* r'_B, g_A *to* g'_A, g_B *to* g'_B, *and the provenance and naming rows to their counterparts, with every public read replayed through the displayed* Cont *routes. Thus classifier agreement preserves the same bilateral event coordination: the record on each observer side and the* GAP *commitment acknowledging the other side.*

Theorem 509.3 (LocalityCell NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocalityCell}^\uparrow}$ *surface has five finite obligations:*

1. *carrier exposure: every accepted object exposes the tuple* $L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$ *of Definition 509.1;*
2. *bilateral classifier stability: accepted comparisons transport both observer histories, both event records, both* GAP *commitments, provenance, and local naming rows as in Definition 509.2;*
3. *route locality: every public read of an event record or* GAP *commitment uses one of the displayed* Cont *routes recorded in* C*;*
4. *ledger exactness: the* Pkg *and* NameCert *rows name exactly the bilateral carrier, classifier, transport, and readback data listed above;*
5. *non-escape: the packet exports no global event coordinate, shared clock, synchronization map, ambient metric, or observer-independent space-time carrier.*

Lean4 ✓ `BEDC.Derived.LocalityCellUp.LocalityCellTasteGate single carrier alignment`

Proof. Carrier exposure is projection from the finite tuple in Definition 509.1. Each coordinate is visible as a displayed history row, event-record row, GAP row, transport ledger, route row, provenance row, or naming row.

Bilateral classifier stability is the rule of Definition 509.2. The comparison transports the two observer histories together with both event-record rows and both GAP commitments, so it cannot keep only one side of the cell.

Route locality follows by inspecting C. A consumer reaches r_A, r_B, g_A, or g_B only through the displayed continuation routes from the two observer histories.

Ledger exactness follows from P and N. These rows record the carrier, classifier, transport, and readback dependencies of the packet, and there is no additional coordinate from which another ledger source could be projected.

Non-escape is immediate from the carrier inventory. Since no coordinate names a global event, shared clock, synchronization map, ambient metric, or observer-independent space-time carrier, no local $\mathsf{NameCert}_{\mathsf{LocalityCell}^\uparrow}$ read can export one. □

Theorem 509.4 (LocalityCell hsame transport)**.** *Let* $L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$ *be an accepted* $\mathsf{LocalityCell}^\uparrow$ *carrier. Any classifier-preserving transport through the displayed* hsame *row* H *preserves both event records and both* GAP *commitments together; it cannot transport* r_A *without the paired* r_B*, nor* g_A *without the paired* g_B*.*

Proof. Project the componentwise transport row H from the carrier. Its entries compare the two observer histories, the two event records, the two GAP commitments, provenance, and local naming row as one packet.

The classifier of Definition 509.2 accepts a transported carrier only when all those components are matched. Therefore a transported read that keeps r_A must also keep the comparison data for r_B, and a transported read that keeps g_A must also keep the comparison data for g_B.

The routes in C replay the public reads after transport. Since every read goes through those routes, no consumer can split one side of the locality cell away from the bilateral transport ledger. □

Theorem 509.5 (LocalityCell inter-observer readback)**.** *Every public read of an accepted* $\mathsf{LocalityCell}^{\uparrow}$ *packet factors through the two displayed observer histories, the two event-record rows, the two bilateral* GAP *commitments, the* hsame *transport ledger, and the* Cont *routes of Definition 509.1. Consequently the packet realizes inter-observer event coordination as finite BEDC evidence and not as a primitive shared world-state.*

Lean4 ✓ `BEDC.Derived.LocalityCellUp.LocalityCellPacket interobserver readback`

Proof. Start with an accepted packet $L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$. The only event-facing coordinates are r_A and r_B, and each is attached to its observer history by the displayed routes in C.

The only cross-observer coordination coordinates are g_A and g_B. They are bilateral GAP commitments, so a public read obtains the claim that each side treats the other side's record as coherent with its own record.

Transport and replay are confined to H and C. Hence any readback of the cell factors through the two histories, two records, two commitments, transport ledger, and routes named in the carrier.

No coordinate of L is a primitive shared world-state or god's-eye event object. The readback therefore realizes the locality cell as finite BEDC coordination evidence between two histories. □

Theorem 509.6 (LocalityCell ObserverLocalityCell lattice link)**.** *An accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet of chapter 513 refines the bilateral* $\mathsf{LocalityCell}^{\uparrow}$ *surface by adding observer-identity and cross-history ledger rows. Every* $\mathsf{LocalityCell}^{\uparrow}$ *read consumed by that sibling still factors through the two observer histories, two event rows, bilateral* GAP *rows,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *rows of Definition 509.1.*

Proof. Project the $\mathsf{LocalityCell}^{\uparrow}$ carrier. Its bilateral surface is the displayed packet $L = (A, B, r_A, r_B, g_A, g_B, H, C, P$ so any read of the local cell has access only to two observer histories, two event records, two GAP commitments, transport, routes, provenance, and local naming.

The $\mathsf{ObserverLocalityCell}^{\uparrow}$ sibling cited in the statement consumes that bilateral surface and adds observer-identity plus cross-history ledger rows. Those added rows refine the consumer context; they do not replace $A, B, r_A, r_B, g_A, g_B, H, C, P, N$ with a global frame or a hidden shared event object.

Apply Definition 509.3, Definition 509.4, and Definition 509.5. The NameCert obligations fix the bilateral ledger, the transport theorem keeps both sides paired under hsame, and the readback theorem forces public event coordination through the displayed histories, event rows, bilateral commitments, and continuation routes.

Thus the sibling relation is a lattice refinement: $\mathsf{ObserverLocalityCell}^{\uparrow}$ may carry additional observer-identity and cross-history ledger data, while every locality-cell read it consumes remains the same bilateral $\mathsf{LocalityCell}^{\uparrow}$ route. □

Theorem 509.7 (LocalityCell scoped kernel route)**.** *Every accepted* $\mathsf{LocalityCell}^{\uparrow}$ *packet factors its carrier, classifier, local* NameCert *obligations,* hsame *transport, inter-observer readback, and TasteGate-visible single-carrier alignment through the displayed* BHist, hsame, Cont, Pkg, *and*

NameCert *rows of Definition 509.1. The route introduces no extra observer carrier, shared clock carrier, synchronization map, ambient metric carrier, or observer-independent space-time row.*

Proof. Begin with Definition 509.1 and Definition 509.2. They place the two observer histories, event records, bilateral GAP commitments, componentwise transport, continuation routes, package provenance, and local name in one displayed packet.

Apply Definition 509.3. The carrier, classifier, exactness, ledger, and non-escape obligations are all recorded by P and N, so there is no separate naming or observer row.

Use Definition 509.4 and Definition 509.5. The transport and readback routes keep both observers inside the same H and C rows rather than splitting them into independent carriers.

Finally use the chapter's TasteGate-visible Lean target recorded in the closure block. It records the single-carrier alignment at the Lean boundary, and the paper route above identifies the same alignment with the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 509.8 (LocalityCell standard bridge surface)**.** *The bilateral* $\mathsf{LocalityCell}^{\uparrow}$ *packet bridges to the concrete-instance schema exactly through the two observer histories* A, B*, the two event rows* r_A, r_B*, the two* GAP *commitments* g_A, g_B*, the componentwise* hsame *row* H*, the displayed* Cont *routes* C*, the package row* P*, and the local* NameCert *row* N *of Definition 509.1. The bridge exports the public locality-cell readback of Definition 509.5 and the scoped kernel route of Definition 509.7; it contains no shared clock, global frame, synchronization map, ambient metric, or observer-independent space-time carrier.*

Proof. Project the bilateral carrier $L = (A, B, r_A, r_B, g_A, g_B, H, C, P, N)$. Its bridge surface is the same finite packet: two histories, two event records, two bilateral GAP commitments, componentwise transport, continuation routes, package provenance, and local naming.

Apply Definition 509.5. Public event coordination already factors through exactly these histories, event rows, GAP rows, H, and C, so the bridge has no separate event object to expose.

Use Definition 509.7 to attach Pkg and NameCert without changing the carrier. The TasteGate-visible single-carrier alignment keeps the bridge on the same BHist, hsame, Cont, Pkg, and NameCert rows.

Since neither the carrier nor any cited route contains a clock, global frame, synchronization map, ambient metric, or observer-independent space-time coordinate, the bridge is exactly the displayed bilateral surface and nothing beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocalityCell}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalityCell}^{\uparrow}$ packet is generated from two observer BHist rows, two event-record rows, two bilateral GAP commitments, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert ledger.

Lean4 ✓ `BEDC.Derived.LocalityCellUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocalityCell}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 509.1, Definition 509.2, Definition 509.3, Definition 509.4, Definition 509.5, Definition 509.7, and Definition 509.8 bind the finite bilateral carrier, classifier, transport, ledger, inter-observer readback, scoped kernel route, bridge surface, and TasteGate-visible single-carrier alignment.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocalityCell}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LocalityCellUp.LocalityCellTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close a global frame, a shared clock, a synchronization map, ambient metric geometry, observer-independent space-time ontology, or bridgeChecked verification.

Upgrade path: No further paper-axis closure remains for the bilateral locality-cell surface; bridgeChecked verification is recorded independently.

Chapter 510

A Concrete Naming Certificate for $\mathsf{GroundRefutation}^{\uparrow}$

The $\mathsf{GroundRefutation}^{\uparrow}$ surface records the permitted contradiction pattern of chapter 9458 as a closed ground-loop naming packet. Its refutation is not a meta rule over arbitrary certificates: it is a finite BHist row whose assumption, falsity route, transports, continuation routes, provenance, and local $\mathsf{NameCert}$ row are all displayed before any negation may be exported.

It is the closed ground-loop specialization of chapter 500. It consumes the permitted-refutation route and forbidden truth-branch exclusion from that broader boundary, but removes the optional decidability and certificate-indexed boundary rows.

Definition 510.1 (Ground refutation carrier)**.** *A* $\mathsf{GroundRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, F, H, C, P, N)$$

with the following displayed rows:

1. *A is the assumption row admitted only inside the fixed ground loop;*
2. *F is the falsity-producing route that sends the displayed assumption row through recorded* Cont *steps to the bottom row;*
3. *H records the componentwise* hsame *transports for the assumption row, falsity route, package provenance, and local naming row;*
4. *C records the displayed* Cont *routes used to read A, replay F, transport the bottom row, and return the exported negation to the same finite packet;*
5. *P is the* Pkg *provenance over* BHist, Cont, hsame, *the bottom row, and the surrounding certificate surface;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a certificate-indexed truth branch, no excluded-middle split, no undecidable double-negation eliminator, and no existential witness selected by contradiction.

Theorem 510.2 (GroundRefutation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *surface for the carrier $G = (A, F, H, C, P, N)$ consists of five finite rows:*

1. *carrier exposure: an accepted object exposes exactly the packet of Definition 510.1;*

2. *classifier row: two accepted packets classify together only when their assumption rows, falsity routes, transports, continuation routes, package provenance, and local naming rows are matched by the displayed* hsame *and* Cont *data;*
3. *stability row: the exported negation remains tied to the same assumption row after every stored* hsame *transport;*
4. *ledger row: the bottom row reached by F is accountable to the displayed* Cont *route and package provenance;*
5. *non-escape row: no consumer may read from G a certificate-indexed truth branch, excluded-middle split, undecidable double-negation elimination, or existential witness by contradiction.*

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationNameCertObligations`

Proof. Carrier exposure is projection from the finite packet of Definition 510.1. The assumption row, falsity route, transport ledger, continuation routes, provenance, and local naming row are the complete public inventory.

The classifier row compares only those coordinates. A classified pair must replay the same assumption read and the same falsity-producing route through the displayed hsame and Cont data.

Stability follows because the exported negation is defined by applying the transported falsity route to the transported assumption row. The route never leaves A, F, H, C, P, N, so the negation remains a ground-loop output rather than a meta rule.

The ledger row is the accountability of F inside C and P. A bottom row can be consumed only together with the route and provenance that produced it.

Finally, the non-escape row follows from the carrier inventory. None of the six coordinates stores a truth predicate over certificates, an excluded-middle branch, an undecided double-negation eliminator, or an existential selector, so no consumer can project such data from the packet. □

Theorem 510.3 (Ground-loop refutation boundary)**.** *Every negation exported by an accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *packet factors through the fixed certificate's displayed* Cont *route and* hsame *transport. The packet therefore realizes only the ground-loop rule that the displayed assumption row A leads by the recorded route F to bottom. It does not export a certificate-indexed truth branch, excluded-middle split, undecidable double-negation elimination, or existential witness by contradiction.*

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationCarrier ground loop boundary`

Proof. Unpack the accepted packet as $G = (A, F, H, C, P, N)$. The only route from an assumed witness of A to bottom is the displayed falsity-producing route F read through C.

Transporting the packet uses H componentwise. The assumption row and falsity route therefore move together, and the exported negation is still the same finite route from the transported assumption row to the transported bottom row.

The remaining coordinates P and N record provenance and the local naming ledger for this route. Since the carrier has no field for truth over certificates, excluded middle, undecided double-negation elimination, or witness selection by contradiction, every exported negation stays inside the closed ground loop. □

Theorem 510.4 (GroundRefutation classifier transport)**.** *Let $G = (A, F, H, C, P, N)$ and $G' = (A', F', H', C', P', N')$ be accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *carriers, each displayed as a finite* BHist *packet with its* hsame *ledger,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row.*

If the classifier row matches A with A', F with F', H with H', C with C', P with P', and N with N' by the displayed hsame *and* Cont *data, then every classified negation exported from G transports to the corresponding classified negation exported from G'. The transport has no coordinate that can introduce a certificate-indexed truth branch, excluded-middle split, undecided double-negation eliminator, or existential witness by contradiction.*

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationCarrier classifier transport`

Proof. Read both carriers through Definition 510.1. The classifier has only the six displayed coordinates, so its transport data is the componentwise comparison of assumption row, falsity route, transport ledger, continuation ledger, provenance, and naming row.

Apply the hsame entries to move the assumption row and the falsity route together. Apply the Cont entries to replay the assumption read, falsity production, bottom transport, and exported-negation return inside the second packet.

The Pkg row and local NameCert row account for the same finite route after transport. Since no remaining carrier coordinate stores a truth branch, excluded-middle split, undecided double-negation eliminator, or witness selector, the transported classifier remains a ground-loop classifier. □

Theorem 510.5 (GroundRefutation route stability)**.** *For an accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *carrier $G = (A, F, H, C, P, N)$ displayed as a finite* BHist *packet, every stored* hsame *transport in H preserves the* Cont *route in C that reads A, replays F, transports the bottom row, and returns the exported negation. Thus the exported negation is stable under the recorded carrier transports, with accountability still carried by P and by the local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row N.*

Proof. Unpack the transport ledger H. Its entries are attached to the assumption row, falsity route, provenance row, and naming row rather than to an ambient meta rule.

The continuation ledger C is read in the same order before and after transport: read the assumption row, replay F, move the bottom row, and return the negation. Componentwise hsame therefore cannot detach the conclusion from its ground assumption.

The provenance row P and the local NameCert row N remain part of the transported packet. The route stability claim is exactly that the transported output is still produced by these displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationCarrier route stability`

Theorem 510.6 (GroundRefutation ledger accountability)**.** *Let $G = (A, F, H, C, P, N)$ be an accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *carrier. Any consumer that reads the bottom row produced by F must also read the displayed* BHist *assumption row A, the* hsame *transport ledger H, the* Cont *replay ledger C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row N. Consequently the bottom row is not available as an unaccounted contradiction token outside the packet.*

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationCarrier ledger accountability`

Proof. The carrier inventory has no standalone bottom-token coordinate. The bottom row appears only through the falsity-producing route F and the continuation entries in C.

A consumer of that row therefore consumes the route that produced it. The hsame ledger records how the route and assumption move, while the Pkg provenance and local NameCert row record why the route belongs to this finite packet.

It follows that the contradiction cannot be exported as anonymous evidence. It is always accompanied by the displayed BHist, hsame, Cont, Pkg, and NameCert rows of G. □

Theorem 510.7 (GroundRefutation consumer readiness)**.** *An accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *consumer may use the carrier* $G = (A, F, H, C, P, N)$ *only through the displayed finite* BHist *rows, the componentwise* hsame *transports, the* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row. Such a consumer is ready for the ground-loop negation output exactly when it requests no certificate-indexed truth branch, excluded-middle split, undecided double-negation eliminator, existential witness by contradiction, or route not listed in* C*.*

Lean4 ✓ `BEDC.Derived.GroundRefutationUp.GroundRefutationCarrier consumer readiness`

Proof. Consumer readiness is a projection discipline on the six-coordinate packet. The available data are precisely A, F, H, C, P, N, read with their BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ structure.

If a consumer asks for the ground-loop negation, the packet supplies the assumption row, the falsity route, the transport ledger, the continuation replay, the provenance row, and the naming row. These are exactly the rows required by the prior boundary and accountability statements.

If a consumer asks for a truth branch, excluded-middle split, undecided double-negation eliminator, existential selector, or unlisted route, the carrier has no such coordinate to project. The readiness condition therefore keeps every valid consumer inside the displayed ground-loop surface. □

Theorem 510.8 (GroundRefutation bridge-facing route)**.** *The public* $\mathsf{GroundRefutation}^{\uparrow}$ *packet presents a bridge-facing finite route: its assumption row* A*, falsity route* F*, componentwise* hsame *ledger* H*, continuation replay* C*, provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$ *row* N *expose exactly the rows needed for the standard reflection bridge to inspect the ground-loop refutation surface. This route adds no certificate-indexed truth branch, excluded-middle split, undecided double-negation eliminator, existential witness by contradiction, or unlisted continuation.*

Proof. Use Definition 510.2 to read the carrier exposure, classifier row, stability row, ledger row, and non-escape row from the local naming surface. These rows expose the finite packet as a public object rather than as an ambient refutation rule.

Then use Definition 510.3, Definition 510.4, and Definition 510.5. They keep the exported negation attached to the same assumption, falsity route, transport ledger, and continuation replay under every displayed classifier transport.

Finally apply Definition 510.6 and Definition 510.7. Any bridge-facing consumer must read the same A, F, H, C, P, N packet, so the route is ready for the standard reflection bridge while the excluded non-escape data remain absent. □

Theorem 510.9 (GroundRefutation standard bridge)**.** *The paper bridge interprets each finite carrier* $G = (A, F, H, C, P, N)$ *as the standard closed ground-loop refutation schema: the displayed assumption row* A *is the only input, the displayed route* F *is the only falsity route,* H *and* C *are the only transport and continuation replays, and* P, N *are the only provenance and naming rows. Conversely, every standard ground-loop read of this schema reflects back to exactly those six rows of* G*. The bridge exposes no certificate-indexed truth branch, excluded-middle split, double-negation eliminator, or existential witness.*

Proof. The forward interpretation reads the six coordinates of Definition 510.1. The standard schema receives the assumption row from A, sends it through the falsity route F, replays the required transports and continuations through H and C, and records accountability through P and N.

For the reverse readback, a standard ground-loop use has no data other than the assumption row, the falsity route, the transport and continuation replays, and the provenance and naming rows. These are exactly the six carrier coordinates, so the readback lands back in G without adding a hidden branch.

The non-escape clauses are inherited from the carrier inventory and consumer readiness. Since neither side of the bridge has a coordinate for certificate-indexed truth, excluded middle, double-negation elimination, or existential selection, the bridge remains a paper bridge for the closed ground-loop schema. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundRefutation}^{\uparrow}$ packet is generated from one displayed assumption row, a falsity-producing continuation route, componentwise hsame transports, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 510.1, Definition 510.2, Definition 510.3, Definition 510.4, Definition 510.5, Definition 510.6, Definition 510.7, Definition 510.8, and Definition 510.9 bind the finite carrier, NameCert obligation rows, classifier transport, route stability, ledger accountability, public consumer non-escape boundary, bridge-facing route, standard paper bridge, and worked ground-loop refutation consumer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundRefutation}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.GroundRefutationUp.GroundRefutationNameCertObligations`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not claim a checked standard bridge theorem, certificate-indexed truth branch, excluded-middle split, undecided double-negation eliminator, or existential witness by contradiction.

Upgrade path: A checked standard bridge theorem would move the verification axis beyond the axiom-clean NameCert obligations; the theory axis is closed by the seed, obligation, scoped, bridged, and worked ground-loop consumer rows.

Chapter 511

A Concrete Naming Certificate for $\mathsf{RefutationWitness}^{\uparrow}$

The $\mathsf{RefutationWitness}^{\uparrow}$ surface is the concrete NameCert landing for constructive reasoning by contradiction. It records an assumption row, a finite continuation route from that assumption to bottom, the produced bottom row, and the local transport and provenance rows needed to replay the route. The packet certifies refutation only as displayed assumption-to-bottom evidence; it does not export a certificate-indexed truth branch, a host-level contradiction principle, or a witness selected from an unwitnessed negation.

The packet consumes the finite trace route of chapter 512 and the stable negative boundary of chapter 507 before its assumption-to-bottom route is accepted. This dependency is limited to displayed assumption rows, route replay, bottom-row accountability, hsame transport, Cont replay, Pkg provenance, and NameCert rows.

Definition 511.1 (RefutationWitness carrier)**.** *A* $\mathsf{RefutationWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, a, \rho, b, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition or predicate row being refuted, read as a certificate-visible* BHist *object rather than as host truth;*
2. *a is the assumed witness row for A at the current history;*
3. *ρ is the finite refutation route, a list of displayed* Cont, msame, *and* hsame *steps beginning at a;*
4. *b is the bottom row reached by replaying ρ from the assumed witness;*
5. *H records componentwise* hsame *transport for A, a, ρ, b, provenance, and local naming rows;*
6. *C records the* Cont *routes used to read the assumption, replay the refutation route, and return the bottom row to a downstream consumer;*
7. *P is the* Pkg *provenance over* BHist, Hist, Cont, msame, hsame, *bottom-witness rows, and the surrounding certificate rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RefutationWitness}^{\uparrow}}$ *ledger.*

The carrier contains no coordinate for excluded middle over certificate predicates, no host truth predicate for A, no double-negation eliminator, and no existential witness selected from a contradiction argument.

Lean4 ✓ `BEDC.Derived.RefutationWitnessUp.RefutationWitnessCarrier assumption bottom replay package`

Theorem 511.2 (RefutationWitness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RefutationWitness}^{\uparrow}}$ *surface for the carrier of Definition 511.1 consists exactly of carrier exposure, assumption accountability, route replay, bottom-row accountability, classifier transport, continuation locality, package provenance, and truth-branch exclusion. A consumer accepts a refutation only by projecting*

$$(A, a, \rho, b, H, C, P, N)$$

and checking that the displayed route ρ *carries the displayed assumption row* a *to the displayed bottom row* b *through the recorded* Cont*,* msame*, and* hsame *data.*

Lean4 ✓ `BEDC.Derived.RefutationWitnessUp.RefutationWitnessNameCertObligations`

Proof. Carrier exposure is projection from Definition 511.1. The packet lists the proposition row, assumption row, route, bottom row, transport ledger, continuation ledger, provenance, and local naming row, and no other coordinate is available.

Assumption accountability and route replay are read from a, ρ, and C. A consumer cannot project b without also projecting the route that starts from the displayed assumed witness and reaches that bottom row.

Classifier transport is componentwise transport through H. The proposition row, assumption row, route, bottom row, package row, and naming row move together, so transport preserves the same finite refutation evidence rather than introducing an external proof principle.

Finally, truth-branch exclusion follows from the carrier inventory. The packet has no row for a family deciding predicates over all naming certificates, no host truth row for A, and no selected existential witness. Hence the NameCert obligations are exhausted by the displayed assumption-to-bottom route. □

Theorem 511.3 (RefutationWitness truth-branch blocker)**.** *No accepted* $\mathsf{RefutationWitness}^{\uparrow}$ *packet can be read as a certificate-indexed truth branch in the sense of Definition 9458.10. Its only negative information is the displayed map from the assumed witness row* a *to the bottom row* b *along* ρ*, and every downstream read remains inside the finite rows* $A, a, \rho, b, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.RefutationWitnessUp.RefutationWitnessCarrier truth branch blocker`

Proof. Start with an accepted packet $R = (A, a, \rho, b, H, C, P, N)$. The available projection operations return only those displayed rows, their componentwise transports, and the continuation routes that replay ρ.

A certificate-indexed truth branch would require a row assigning, for each naming certificate in a carrier of certificates, either a positive witness or a refutation. No such carrier-wide predicate row appears in R, and no coordinate ranges over all naming certificates.

The packet therefore supports only the local refutation read: assume the displayed witness row a, replay the finite route ρ, and obtain the displayed bottom row b. That read blocks any extraction of excluded middle, undecided double-negation elimination, or existential witness selection from the $\mathsf{RefutationWitness}^{\uparrow}$ surface. □

Theorem 511.4 (RefutationWitness ground-refutation route link)**.** *Every accepted* $\mathsf{GroundRefutation}^{\uparrow}$ *or* $\mathsf{RefutationBoundary}^{\uparrow}$ *consumer that uses an assumption-to-bottom row consumes the local* $\mathsf{RefutationWitness}^{\uparrow}$ *route shape through chapter 510 and chapter 500 only after the displayed rows*

$$A, a, \rho, b, H, C, P, N$$

are present. The route is a finite replay from the assumed witness row a to the bottom row b, not a host-level truth branch or a selected witness extracted from an undecided negation.

Lean4 ✓ `BEDC.Derived.RefutationWitnessUp.RefutationWitnessCarrier ground refutation route link`

Proof. Project the carrier of Definition 511.1. The local route shape has exactly the proposition row, assumed witness row, finite refutation route, bottom row, transport ledger, continuation ledger, provenance row, and naming row.

Use Definition 511.2. Its route replay and bottom-row accountability clauses force any consumer to carry the displayed assumption-to-bottom route ρ before it may consume the bottom row b in a ground-refutation or boundary-refutation packet.

Finally apply Definition 511.3. Since the local packet exports no certificate-indexed truth branch, excluded middle, double-negation eliminator, or witness selector, the GroundRefutation and RefutationBoundary consumers can only cite the finite route through the two named sibling chapters. □

Theorem 511.5 (Refutation witness mature treatment)**.** *Every mature consumer of an accepted* $\mathsf{RefutationWitness}^{\uparrow}$ *packet receives exactly the finite proposition row A, assumed-witness row a, refutation-route row ρ, bottom row b, componentwise* hsame *ledger H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The mature read factors through Definition 511.1, Definition 511.2, Definition 511.3, and Definition 511.4; it introduces no truth branch, selected witness, excluded middle, or double-negation eliminator.*

Lean4 ✓ `BEDC.Derived.RefutationWitnessUp.RefutationWitnessCarrier mature treatment`

Proof. Project the carrier of Definition 511.1. The mature consumer can read only $A, a, \rho, b, H, C, P, N$, because these are the displayed rows admitted by the finite packet.

Apply the NameCert obligations, Definition 511.2. Carrier admission, assumption accountability, route replay, bottom-row accountability, classifier transport, continuation locality, provenance, and local naming exhaust the accepted surface.

Now use the truth-branch blocker, Definition 511.3, and the ground-refutation route link, Definition 511.4. The former removes any certificate-indexed truth branch, excluded middle, double-negation eliminator, or witness selector; the latter places every ground-refutation and boundary-refutation read on the displayed assumption-to-bottom route.

Thus the mature treatment is exactly the finite refutation packet and its two sibling routes, with no additional logical row. □

Closure status for $\mathsf{NameCert}_{\mathsf{RefutationWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RefutationWitness}^{\uparrow}$ packet is generated from a proposition row, an assumed witness row, a finite assumption-to-bottom route, the produced bottom row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RefutationWitness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 511.1, Definition 511.2, Definition 511.3, Definition 511.4, and Definition 511.5 bind the displayed RefuterTrace, StableNegationBoundary, GroundRefutation, RefutationBoundary, BHist, hsame, Cont, Pkg, and NameCert rows for carrier admission, route replay, classifier transport, bottom-row accountability, provenance, local NameCert admission, truth-branch exclusion, the GroundRefutation–RefutationBoundary bridge route, and mature finite-row consumption.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RefutationWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, excluded middle over certificate predicates, host truth, undecided double-negation elimination, existential witness selection, or a public standard bridge.

Upgrade path: Mature closure requires a standard bridge check for the displayed GroundRefutation and RefutationBoundary route without adding a truth branch or selected witness.

Chapter 512

A Concrete Naming Certificate for $\mathsf{RefuterTrace}^{\uparrow}$

The $\mathsf{RefuterTrace}^{\uparrow}$ surface realizes the permitted-refutation side of chapter 9458 as a finite trace packet. It records the assumed proposition row, the route from that assumption to $\bot$, the local exclusions that keep forbidden contradiction schemas out of the consumer interface, and the exact BHist routes through which the certificate may be read.

Definition 512.1 (RefuterTrace carrier)**.** *A* $\mathsf{RefuterTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, U, F, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition row being refuted, represented only by the certificate-visible predicate row that a consumer may assume locally;*
2. *U is the assumption token for A, naming the temporary witness that the trace is allowed to inspect;*
3. *F is the finite refuter trace, a displayed route from U to $\bot$ through* Cont, Ext, hsame, msame, *and the local certificate rows;*
4. *E is the exclusion ledger for the forbidden contradiction moves of Definition 9458.5: existential extraction without a witness, double-negation elimination without a decision row, and excluded-middle branching over the carrier of all naming certificates;*
5. *H stores componentwise* hsame *transports for the proposition, assumption, refuter trace, exclusion ledger, package provenance, and local naming rows;*
6. *C stores the displayed* Cont *routes for entering the assumption, replaying the finite trace, reaching $\bot$, checking the exclusions, and returning to the local ledger;*
7. *P is the* Pkg *provenance over* BHist, Ext, Cont, msame, hsame, *the trace rows, the exclusion ledger, and the surrounding* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}$ *ledger.*

The carrier contains no proof by contradiction that exports a witness, no double-negation eliminator, no excluded-middle oracle, no host truth predicate, and no classifier over the carrier of all naming certificates.

Theorem 512.2 (RefuterTrace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission for the finite packet* $R = (A, U, F, E, H, C, P, N)$ *of Definition 512.1;*
2. *trace readability, saying that* F *is consumed only as a displayed route from the local assumption token* U *for* A *to* $\bot$*;*
3. *transport stability, saying that the stored* hsame *rows in* H *and* Cont *routes in* C *preserve the proposition row, assumption token, refuter trace, and exclusion ledger;*
4. *forbidden-branch exclusion, saying that* E *blocks the three forbidden contradiction schemas named in Definition 9458.5;*
5. *consumer boundary, saying that every public read of the packet factors through* A, U, F, E, H, C, P, N *and exports only the negation certified by the trace.*

Lean4 ✓ `BEDC.Derived.RefuterTraceUp.RefuterTraceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 512.1. The accepted packet has exactly the displayed rows A, U, F, E, H, C, P, N, so every obligation can be read from finite BHist data.

Trace readability follows by inspecting the coordinate F. Its source is the local assumption token U for the proposition row A, and its target is $\bot$; no other coordinate supplies a witness-producing contradiction rule.

Transport stability is supplied by H and C. The hsame rows preserve the named components, while the Cont rows replay the route through the same finite trace before returning to P and N.

The forbidden-branch clause is exactly the role of E. Since E records the absence of unwitnessed existential extraction, undecidable double-negation elimination, and excluded-middle branching over all naming certificates, no consumer can use the trace as a meta-level truth branch.

The consumer boundary follows by combining these projections. A public read may assume A locally, replay F to $\bot$, and therefore consume $\neg A$ inside the fixed packet, but it cannot export any branch or witness absent from A, U, F, E, H, C, P, N. □

Theorem 512.3 (RefuterTrace stable negation sibling route)**.** *Every accepted* $\mathsf{RefuterTrace}^{\uparrow}$ *packet that is consumed as stable negative information factors its assumption-to-bottom route through the* $\mathsf{StableNegationBoundary}^{\uparrow}$ *sibling surface of chapter 507 before any downstream consumer reads it as stable negation. The route starts from the RefuterTrace carrier and NameCert obligations, uses permitted refutation soundness, applies forbidden branch exclusion, and is then exhausted by the StableNegationBoundary NameCert and ledger rows.*

The sibling route exports no double-negation eliminator, excluded-middle oracle, witness by contradiction, or classifier over all naming certificates.

Proof. Start inside the $\mathsf{RefuterTrace}^{\uparrow}$ packet. The NameCert obligations of Definition 512.2 expose exactly A, U, F, E, H, C, P, N, so the assumption-to-bottom route is the displayed trace F from the local assumption token U to $\bot$.

Use permitted refutation soundness and forbidden branch exclusion before handing the route to the sibling packet. Definition 512.4 certifies only the local permitted pattern "assume A, derive $\bot$, conclude $\neg A$", while Definition 512.5 blocks witness extraction, undecidable double-negation elimination, and excluded-middle branching over all naming certificates.

Now read the stable-negation boundary. The NameCert obligations and ledger exhaustion of Definition 507.2 and Definition 507.5 force any downstream stable-negation consumer through the boundary rows $A, R, D, T, L, H, C, P, N$ of $\mathsf{StableNegationBoundary}^{\uparrow}$. Since those rows contain

only local refutation, optional decision, classifier, exclusion ledger, transport, route, provenance, and naming data, the composed sibling route still exports none of the forbidden contradiction interfaces. □

Theorem 512.4 (RefuterTrace permitted refutation soundness)**.** *For an accepted* $\mathsf{RefuterTrace}^{\uparrow}$ *carrier* $R = (A, U, F, E, H, C, P, N)$*, replaying the finite trace* F *under the displayed transports and continuation routes certifies* $\neg A$ *in the local* $\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}$ *ledger. The certification is exactly the permitted form "assume* A*, derive* $\bot$*, conclude* $\neg A$*" of Definition 9458.1.*

Lean4 ✓ `BEDC.Derived.RefuterTraceUp.RefuterTrace permitted refutation route`

Proof. Begin with the assumption token U for the proposition row A. The trace row F is a finite Cont route from that token to $\bot$, and the routes in C make each step visible to the certificate consumer.

The transports in H ensure that replaying the route does not change the proposition being refuted. Thus the endpoint $\bot$ is reached from the same local assumption for A rather than from an ambient truth predicate or an unrelated classifier.

By the permitted pattern of Definition 9458.1, a route from an assumed witness of A to $\bot$ is precisely a certificate-level proof of $\neg A$. The resulting negation is recorded in the local ledger N and nowhere outside the displayed packet. □

Theorem 512.5 (RefuterTrace forbidden branch exclusion)**.** *An accepted* $\mathsf{RefuterTrace}^{\uparrow}$ *carrier cannot be read as any of the three forbidden contradiction schemas of Definition 9458.5. In particular, the packet does not provide a witness for an existential claim obtained only by refutation, does not eliminate* $\neg\neg A$ *without a decision row, and does not branch on excluded middle for predicates over the carrier of all naming certificates.*

Proof. The carrier has no coordinate whose codomain is an existential witness selected from a contradiction. The only witness-like row is the temporary assumption token U, and that token is consumed by F to reach $\bot$ rather than exported.

The carrier also has no decision row for A. Therefore a consumer reading $\neg\neg A$ cannot obtain A from the packet; the exclusion ledger E records exactly that this route is unavailable.

Finally, no row in R ranges over the carrier of all naming certificates. The classifier-visible data are the finite trace coordinates A, U, F, E, H, C, P, N, so an excluded-middle branch over arbitrary certificates has no source inside the packet. □

Theorem 512.6 (RefuterTrace consumer boundary)**.** *Every consumer of an accepted* $\mathsf{RefuterTrace}^{\uparrow}$ *carrier factors through the finite route*

$$A \longmapsto U \longmapsto F \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The consumer may read the certified negation produced by the trace, but it cannot read a host truth predicate, a global excluded-middle oracle, a witness selector, or a classifier over all naming certificates.

Proof. Project the accepted carrier of Definition 512.1. The only public coordinates are A, U, F, E, H, C, P, N, and the displayed continuation route orders those coordinates from the local proposition row to the naming ledger.

The certified negation is obtained by replaying F from U to $\bot$, with H and C preserving the same proposition row throughout the replay. This is the permitted-refutation route of Definition 512.4.

The unavailable reads are ruled out componentwise. Host truth, excluded-middle branching, witness selection, and all-certificate classification have no coordinate in the packet, and the exclusion ledger E records that they are not consumer interfaces. □

Theorem 512.7 (RefuterTrace finite witnessed boundary)**.** *For an accepted* $\mathsf{RefuterTrace}^{\uparrow}$ *carrier* $R = (A, U, F, E, H, C, P, N)$*, any downstream consumer that reads the trace as a finite witnessed refutation or as a bounded-search refutation must first pass through the displayed rows*

$$A, U, F, E, H, C, P, N.$$

Only after that local passage may the consumer enter chapter 576 or chapter 579, and then only as finite consumer packaging for the already displayed trace.

The passage does not provide witness selection by contradiction, double-negation elimination without a decision row, excluded-middle branching, host truth, or a classifier over all naming certificates.

Proof. Project the carrier of Definition 512.1. Its certificate-visible coordinates are exactly A, U, F, E, H, C, P, N, so any consumer route begins with those rows rather than with an ambient contradiction principle.

Apply the obligation surface of Definition 512.2. The trace is readable only as the finite route F from the local assumption token U for A to $\bot$, while the rows E, H, C, P, N record exclusions, transports, continuation routes, provenance, and local naming.

Apply the consumer boundary of Definition 512.6. A downstream reader may consume the certified negation produced by the trace, but it cannot obtain host truth, a global excluded-middle oracle, a witness selector, or an all-certificate classifier from the packet.

The sibling chapters are therefore read only after the local packet has been exhausted. chapter 576 supplies finite witnessed-refutation packaging, and chapter 579 supplies bounded-search packaging; neither chapter is an extra logical principle for the RefuterTrace carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RefuterTrace}^{\uparrow}$ packet is generated from a proposition row, a local assumption token, a finite refuter trace to $\bot$, a forbidden-branch exclusion ledger, componentwise transports, displayed continuation routes, package provenance, and a local naming ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 512.1, Definition 512.2, Definition 512.4, Definition 512.5, Definition 512.6, and Definition 512.7 bind the finite carrier, obligation surface, permitted refutation route, forbidden-branch exclusion, consumer boundary, and finite consumer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RefuterTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close existential-by-refutation witness selection, undecidable double-negation elimination, excluded middle over naming certificates, host truth, or a checked Lean certificate.

Upgrade path: Obligation closure requires Lean encoding of the packet and checked theorems for trace readability, transport stability, forbidden-branch exclusion, permitted-refutation soundness, and consumer non-escape.

Chapter 513

A Concrete Naming Certificate for $\mathsf{ObserverLocalityCell}^{\uparrow}$

The $\mathsf{ObserverLocalityCell}^{\uparrow}$ interface packages the finite two-observer cell described in Definition 9420.38. It is a local naming surface: two observer-history rows, two event-description rows, paired Gap acknowledgements, componentwise transport, continuation routes, provenance, and a local certificate are visible, while global spacetime, a shared clock, a manifold coordinate chart, and an all-observer frame are not part of the carrier.

Definition 513.1 (ObserverLocalityCell carrier)**.** *An* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N)$$

with the following visible rows:

1. O_0 *and* O_1 *are the two observer-history rows being compared;*
2. R_0 *and* R_1 *are the two local event-description rows recorded inside those histories;*
3. G_0 *and* G_1 *are paired* Gap *acknowledgement rows, each committing its own observer side to the coherence of* R_0 *and* R_1 *as descriptions of one event;*
4. H *stores componentwise* hsame *transport for the observer histories, event rows, acknowledgement rows, continuation routes, package row, and local name row;*
5. C *stores the displayed* Cont *routes by which a consumer reads the two observer histories, the two event rows, and the two acknowledgement rows;*
6. P *is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{FiniteMultiHistPacket}^{\uparrow}$, Hist, BHist, Gap, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting these displayed coordinates to a common two-history surface. The carrier has no coordinate for an external event object, global observer frame, synchronized clock, metric geometry, manifold point, or instantaneous remote record.

Theorem 513.2 (ObserverLocalityCell NameCert obligations)**.** *For the carrier of Definition 513.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet* $L = (O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N)$*;*

2. *classifier transport to a common observer-history and event-row surface;*
3. *paired* Gap *acknowledgement exactness, requiring both G_0 and G_1 to name the same transported pair of event rows;*
4. *stability under the componentwise* hsame *row H and displayed* Cont *routes in C;*
5. *non-escape for global frames, synchronized clocks, manifold coordinates, ambient metric geometry, and instantaneous remote records.*

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket semantic name certificate`

Proof. Carrier admission is projection from Definition 513.1. The rows $O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N$ display the two observer histories, the two event records, the paired acknowledgements, transport, routes, provenance, and local name.

Classifier transport uses H to move every displayed coordinate to a common two-history surface. Once transported, the only consumer reads are the routes recorded in C.

The acknowledgement exactness row reads G_0 and G_1 together. Each side commits to the same transported pair (R_0, R_1), so the named cell is the paired commitment surface rather than an external event object.

The non-escape clause follows from the carrier inventory. No row supplies a global frame, synchronized clock, manifold coordinate, ambient metric, or instantaneous remote record. □

Theorem 513.3 (ObserverLocalityCell GAP coherence)**.** *Let $L = (O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N)$ be an accepted* ObserverLocalityCell$^{\uparrow}$ *carrier. Every coherence read of the cell factors through both acknowledgement rows G_0 and G_1 after transporting O_0, O_1, R_0, R_1 by H and replaying the displayed routes in C. A one-sided acknowledgement is not a locality cell.*

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket paired gap transport`

Proof. Project the accepted packet. The only rows that assert cross-observer coherence are G_0 and G_1; the observer histories and event-description rows merely supply the local records being compared.

Transport the two histories and the two event rows through H. The routes in C then read the transported event rows before reading the two acknowledgement rows.

Because the carrier requires both G_0 and G_1, a coherence read that omits one side cannot satisfy the displayed packet. Thus the cell is exactly the paired Gap commitment over the transported event rows. □

Theorem 513.4 (ObserverLocalityCell consumer boundary)**.** *Every* ObserverHistoryIdentity$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, *or* FiniteMultiHistPacket$^{\uparrow}$ *consumer of an accepted* ObserverLocalityCell$^{\uparrow}$ *packet may read exactly the two observer-history rows, the two event-description rows, the paired* Gap *acknowledgement rows, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and the local* NameCert *row. The packet exports no shared clock, global event object, metric light-cone, manifold coordinate chart, or all-observer synchronization map.*

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket consumer boundary`

Proof. Start with a consumer route into the accepted packet. The carrier projection exposes the two histories, two event rows, and paired acknowledgement rows as the only observational data.

For an $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ consumer, the relevant read is the identity-preserving history surface O_0, O_1 together with the transported event rows. For a $\mathsf{CrossHistCausal}^{\uparrow}$ or $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ consumer, the cross-history access still reaches coherence through the same G_0, G_1 pair.

The remaining data are H, C, P, N. They transport, route, record provenance, and name the local cell; they do not supply a clock, global event, metric cone, chart, or synchronization map. □

Theorem 513.5 (ObserverLocalityCell finite ledger package)**.** *The carrier, NameCert obligations, paired* Gap *coherence, and consumer boundary assemble a finite* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *package over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{FiniteMultiHistPacket}^{\uparrow}$, Hist, BHist, Gap, hsame, Cont, Pkg, *and* NameCert. *Every locality-cell read factors through this ledger, and no host spacetime, global observer frame, or infinite coherence web is exported.*

Proof. The carrier supplies the finite tuple of Definition 513.1. The NameCert obligations in Definition 513.2 bind its carrier, classifier, acknowledgement exactness, stability, and non-escape rows.

The paired Gap coherence theorem identifies the cell condition as a two-sided acknowledgement after componentwise transport. The consumer boundary theorem identifies the exact rows visible to observer-history, cross-history, and finite multi-history readers.

Repacking these rows gives the stated finite ledger package. Since each cited row belongs to the displayed packet, there is no exported host spacetime, global frame, or infinite coherence structure. □

Theorem 513.6 (ObserverLocalityCell TasteGate witness boundary)**.** *Any later* ChapterTasteGate *witness for an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet must read only the displayed two observer-history rows* O_0, O_1*, two event-description rows* R_0, R_1*, paired* Gap *rows* G_0, G_1*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. Such a witness may classify the packet by those rows, but it has no admissible read of a global frame, shared clock, metric cone, manifold coordinate, or observer-independent event object.*

Proof. Project the accepted packet through Definition 513.1. The visible rows are exactly the two observer histories, two event rows, paired acknowledgement rows, transport row, continuation routes, package row, and local naming row.

A ChapterTasteGate witness can audit only the carrier and routes it is handed. The NameCert obligations and consumer boundary already restrict those routes to the displayed locality-cell packet.

Therefore the witness boundary is the same finite read surface. Since the excluded global objects are not coordinates of the packet, no TasteGate read can import them. □

Theorem 513.7 (ObserverLocalityCell BHistCarrier flow surface)**.** *The carrier-to-flow presentation of an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet is a finite* BHistCarrier*-facing flow over* $O_0, O_1, R_0, R_1, G_0, G_1, H, C$*, Each flow edge is either a displayed* Cont *route, a componentwise* hsame *transport, a paired* Gap *acknowledgement read, or the final local* NameCert *read; no edge targets a host spacetime row or an infinite observer web.*

Proof. Read the packet as a finite BHist tuple. The carrier rows supply the only vertices available to a BHistCarrier-facing presentation.

The permitted edges are the routes recorded in C, the transports recorded in H, and the two acknowledgement reads through G_0 and G_1. The local naming row N closes the presentation as a finite certificate read.

Because there is no carrier coordinate for host spacetime or an infinite observer web, no flow edge can target such an object. The flow surface is therefore finite and exactly over the displayed locality-cell rows. □

Theorem 513.8 (ObserverLocalityCell conservativity audit surface)**.** *Every public consumer and audit read of an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet is conservative over the finite ledger of Definition 513.5. It cannot import a global frame, shared clock, metric cone, observer-independent event object, manifold coordinate chart, or all-observer synchronization map as an extra premise or exported conclusion.*

Proof. Start from any public consumer or audit route. By Definition 513.4, the route can read only the displayed observer rows, event rows, acknowledgement rows, transports, continuation routes, provenance, and local naming row.

The finite ledger package repacks those rows without adding another carrier. Thus an audit may check whether the stated rows are present and coherently paired, but it has no source for a global frame, shared clock, metric cone, or observer-independent event.

Consequently every public read is conservative over the finite locality-cell ledger. Any premise or conclusion requiring the excluded global objects lies outside the packet boundary. □

Theorem 513.9 (ObserverLocalityCell carrier-admission obligation)**.** *For an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet, carrier admission admits exactly the displayed rows*

$$O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N$$

as the BHistCarrier*-facing flow surface. The admitted coordinates are the two observer-history rows, two event-description rows, paired* Gap *acknowledgement rows, componentwise* hsame *transport row, displayed* Cont *route row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}^{\uparrow}}$ *row; no global frame, shared clock, metric row, manifold coordinate, or observer-independent event object is admitted as a carrier coordinate.*

Proof. Apply the carrier definition in Definition 513.1. It lists exactly the ten rows displayed in the theorem statement and identifies their observer, event, acknowledgement, transport, route, provenance, and local naming roles.

Use the NameCert obligations of Definition 513.2 to read carrier admission as a finite obligation over that tuple. The paired Gap coherence theorem then fixes G_0 and G_1 as acknowledgement rows over the transported event pair.

Finally apply the BHistCarrier flow surface of Definition 513.7 and the conservativity audit surface of Definition 513.8. These results exclude every non-displayed global coordinate, so the admitted flow surface is exactly the ten-row packet. □

Theorem 513.10 (ObserverLocalityCell classifier-stability obligation)**.** *Componentwise* hsame *transport along H preserves the paired acknowledgement classifier for an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet. After transporting $O_0, O_1, R_0, R_1, G_0, G_1, C, P, N$ through the displayed row H, the classifier still reads the same two-sided* Gap *acknowledgement surface and does not add a global observer frame, synchronized clock, metric comparison, manifold chart, or observer-independent event object.*

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket classifier stability obligation`

Proof. Start from the classifier transport obligation in Definition 513.2. The transport row H acts componentwise on the observer histories, event-description rows, acknowledgement rows, continuation routes, package provenance, and local name.

Apply the GAP coherence theorem Definition 513.3. The coherence read is still the paired read through G_0 and G_1 after the displayed rows are moved to a common two-history surface.

The consumer boundary of Definition 513.4 and the TasteGate witness boundary of Definition 513.6 identify the same finite read surface for downstream checks. Since the conservativity audit surface excludes all global frame, clock, metric, chart, and event-object coordinates, the transported classifier is stable without importing external data. □

Theorem 513.11 (ObserverLocalityCell consumer-exhaustion obligation)**.** *Every* ObserverHistoryIdentity$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, FiniteMultiHistPacket$^{\uparrow}$, *and* ChapterTasteGate *audit read of an accepted* ObserverLocalityCell$^{\uparrow}$ *packet factors through the displayed rows* $O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N$. *The exhausted consumer surface is the carrier definition, the local* NameCert *obligation surface, the paired* Gap *coherence surface, the consumer boundary, the TasteGate witness boundary, the* BHistCarrier *flow surface, and the conservativity audit surface; it contains no additional global data source.*

Proof. For ObserverHistoryIdentity$^{\uparrow}$ reads, apply Definition 513.4. The only visible history data are O_0 and O_1, with event rows and acknowledgement rows read through the displayed transport and continuation rows.

For CrossHistCausal$^{\uparrow}$ and FiniteMultiHistPacket$^{\uparrow}$ reads, use Definition 513.3 together with the finite ledger package of Definition 513.5. Cross-history access reaches coherence only through G_0 and G_1 over the transported event rows.

For ChapterTasteGate audits, use Definition 513.6. Because this chapter is marked with `\origin{ai}`, any promotion past seed closure must cite the checked Lean target `BEDC.Derived.ObserverLocalityCellUp.tast` once that declaration exists; this paper theorem does not add a Lean marker.

Finally combine the BHistCarrier flow surface and conservativity audit surface. They show that every allowed read has a source among the ten displayed rows and that no global frame, clock, metric, manifold coordinate, observer-independent event object, or all-observer synchronization map is available. □

Theorem 513.12 (ObserverLocalityCell scoped export surface)**.** *The carrier, local* NameCert *obligations, paired* Gap *coherence, consumer boundary, finite ledger package,* BHistCarrier *flow surface, and conservativity audit surface of an accepted* ObserverLocalityCell$^{\uparrow}$ *packet export exactly one scoped consumer surface for* ObserverHistoryIdentity$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, FiniteMultiHistPacket$^{\uparrow}$, *and* ChapterTasteGate *reads. That surface is the ten-row packet* $O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N$ *and no additional global-frame row.*

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket scoped export surface`

Proof. Begin with carrier admission and the local NameCert obligations. They identify $O_0, O_1, R_0, R_1, G_0, G_1, H, C$ as the complete two-observer locality-cell packet.

Apply paired Gap coherence and the consumer boundary. These route ObserverHistoryIdentity$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, and FiniteMultiHistPacket$^{\uparrow}$ reads through the two observer-history rows, two event rows, and two acknowledgement rows under the displayed transport and continuation rows.

Add the finite ledger package, BHistCarrier flow surface, and conservativity audit surface. They package the same rows for downstream reads and rule out global spacetime, a shared clock, a metric cone, manifold coordinates, observer-independent event objects, and all-observer synchronization maps.

The remaining TasteGate read is therefore not a second surface. It is a ground-compiler audit of the same ten rows, so the scoped export is unique for all four consumer families. □

Theorem 513.13 (ObserverLocalityCell TasteGate-cited readiness)**.** *For this AI-origin* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *chapter, promotion along the scoped consumer surface must cite the checked TasteGate declaration when it is present. The visible rows remain exactly*

$$O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N.$$

No global spacetime, shared clock, metric cone, manifold coordinate, or observer-independent event object enters the TasteGate-facing surface.

Proof. Read the AI-origin closure status and the existing TasteGate witness boundary. The paper surface is permitted to expose the TasteGate channel only through the finite locality-cell carrier.

Use Definition 513.12. It identifies the one scoped consumer surface for observer-history, cross-history causal, finite-multi-history, and TasteGate reads as the ten displayed rows.

The checked TasteGate declaration, when cited, checks that same row discipline rather than adding global geometry. The conservativity audit surface keeps global spacetime, shared clocks, metric cones, manifold coordinates, and observer-independent event objects outside the packet. □

Theorem 513.14 (ObserverLocalityCell public export certificate)**.** *An accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet has one public export certificate over*

$$O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N.$$

The certificate assembles carrier admission, local NameCert *obligations, paired* Gap *coherence, the consumer boundary, the finite ledger package, the* BHistCarrier *flow surface, the conservativity audit surface, the scoped export surface, and the TasteGate-cited readiness theorem into a single public packet for the two-observer locality-cell surface.*

Proof. Start with carrier admission and the local NameCert obligations. They identify the two observer-history rows, two local event-description rows, paired acknowledgement rows, componentwise transport row, continuation routes, package provenance, and local naming row as the full packet.

Add paired Gap coherence, the consumer boundary, and the finite ledger package. These route $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ reads through the same ten rows.

Next attach the BHistCarrier flow surface and the conservativity audit surface. They ensure that the public packet remains a finite history-carrier flow and that no global spacetime, shared clock, metric cone, manifold coordinate, observer-independent event object, or all-observer synchronization map is exported.

Finally apply the scoped export surface and Definition 513.13. The surrounding closure scope already cites the checked TasteGate marker for `BEDC.Derived.ObserverLocalityCellUp.taste_gate`, so this public certificate records the same TasteGate-facing row discipline without adding a second Lean marker. □

Falsifiable prediction: Any accepted $\mathsf{ObserverLocalityCell}^{\uparrow}$ consumer that bypasses both paired Gap acknowledgement rows G_0, G_1, or that imports a global observer frame, refutes the two-observer locality-cell packet.

Independence witness: Nearest siblings: observerhistoryidentity, crosshistcausal, finite-multihist-packet, observerlocalitytransport. None is bijective to the ten-row locality cell because each omits either the paired acknowledgement surface G_0, G_1 or the local $\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}^{\uparrow}}$ row.

Theorem 513.15 (ObserverLocalityCell atomicity rows)**.** *The AI-origin public surface of an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet is atomic at the ten displayed rows*

$$O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N.$$

Every public or TasteGate-facing consumer must preserve the paired acknowledgement surface G_0, G_1 *together with the local naming row* N*. The existing Lean TasteGate route in* `TasteGate.lean` *must be read together with its* Nontrivial *and* FieldFaithful *instances, so the TasteGate declaration checks the same non-collapsed and field-faithful carrier rather than a single global frame.*

Proof. Use the public export certificate of Definition 513.14. It assembles the carrier, local NameCert obligations, paired Gap coherence, consumer boundary, finite ledger package, BHistCarrier flow surface, conservativity audit surface, scoped export surface, and TasteGate-cited readiness into the ten displayed rows.

The falsifiable prediction fixes the rejected counter-routes: a consumer that omits both G_0, G_1 or imports a global frame has left the accepted packet. The independence witness separates this carrier from observer-history identity, cross-history causal, finite-multi-history, and locality-transport siblings because none of those siblings carries both the paired acknowledgement surface and the local locality-cell name.

Finally read the TasteGate declaration through the surrounding `TasteGate.lean` instances. Nontriviality prevents collapse of the locality-cell carrier, and field faithfulness preserves the displayed rows, so the checked route is paired with the same atomic surface rather than with an external frame object. □

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellTasteGate single carrier alignment`

Theorem 513.16 (ObserverLocalityCell sibling lattice link)**.** *Every accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *packet consumes the two-observer identity surface of chapter 477, refines the finite local cell surface of chapter 509, and exports its cross-history ledger route through chapter 759. In particular, the packet is a concrete-instance lattice node rather than a kernel-only island.*

Proof. Project an accepted packet to its carrier rows

$$O_0, O_1, R_0, R_1, G_0, G_1, H, C, P, N.$$

The rows O_0, O_1 are the two observer-history identity slots, and H is the componentwise transport that keeps those slots on the observer-history identity surface.

The locality-cell refinement is carried by R_0, R_1, G_0, G_1. The two local event-description rows are read together with the paired Gap acknowledgement rows, so the accepted packet specializes the finite local cell surface without replacing it by a global observer frame.

Finally the continuation row C, provenance row P, and local naming row N give the route exported to the inter-history locality ledger. That ledger reads the same two-observer cell as a finite cross-history packet, so the chapter participates in the concrete-instance dependency lattice through observer identity, local cell refinement, and inter-history ledger export. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}\uparrow}$.

Constructive story: An $\mathsf{ObserverLocalityCell}^{\uparrow}$ packet is generated from two observer-history rows, two local event-description rows, paired Gap acknowledgement rows, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 513.1, Definition 513.2, Definition 513.3, Definition 513.4, Definition 513.5, Definition 513.6 with

Lean4 ✓ `BEDC.Derived.ObserverLocalityCellUp.taste gate`

, Definition 513.7, Definition 513.8, Definition 513.9, Definition 513.10, Definition 513.11, Definition 513.14, Definition 513.15, Definition 513.16, and the TasteGate-cited readiness surface close the mature carrier, NameCert obligations, GAP coherence, consumer boundary, finite ledger package, TasteGate witness boundary, BHistCarrier flow surface, conservativity audit surface, public export certificate, atomic row treatment, sibling lattice link, and three obligation theorems for the two-observer locality-cell packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverLocalityCell}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ObserverLocalityCellUp.ObserverLocalityCellPacket\_semantic\_name\_certificate`

Bridge status: none

Not claimed: This chapter does not close global spacetime, a shared clock, manifold coordinates, metric light-cones, all-observer synchronization, infinite coherence webs, or a StdBridge synchronization theorem for the packet.

Upgrade path: The next step is a checked StdBridge synchronization for the same public two-observer locality-cell packet.

Chapter 514

ObserverFilterUp NameCert construction

The $\mathsf{ObserverFilter}^{\uparrow}$ surface packages observer-history identity as a finite selection of BHist rows together with a ledger for everything the observer does not export. It is a reusable filter carrier: an observer may retain a finite subhistory for downstream naming, while the omitted rows remain visible as a boundary ledger rather than disappearing into host equality, quotient identity, or an unrecorded subject parameter.

The filter consumes observer-history identity through chapter 477 and hands selected rows to the StreamName surface of chapter 126; the omitted-row ledger remains a boundary certificate rather than stream data.

Definition 514.1 (ObserverFilter BHist carrier)**.** *An* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (H, S, O, R, L, C, P, N)$$

with the following visible rows:

1. *H is the source observer-history row;*
2. *S is the finite selected subhistory row, listing exactly the* BHist *entries retained for public observation;*
3. *O is the omitted-row ledger, listing every source row not selected into S together with its local boundary reason;*
4. *R records the componentwise* hsame *transports between source rows, selected rows, omitted rows, downstream stream-name rows, continuation routes, package provenance, and the local name;*
5. *L is the observer ledger tying each retained or omitted row to its source position, observer decision, and local naming boundary;*
6. *C records the displayed* Cont *routes by which a consumer reads the source history, retained subhistory, omitted ledger, and downstream handoff rows;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$ *row.*

The classifier for F compares only the retained row S, omitted-row ledger O, transport row R, observer ledger L, continuation row C, provenance row P, and local name row N. The carrier has no coordinate for host equality of histories, quotient streams, hidden observer state, or an unledgered deletion.

Theorem 514.2 (ObserverFilter NameCert obligations)**.** *For the carrier of Definition 514.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet $F = (H, S, O, R, L, C, P, N)$;*
2. *classifier transport by the componentwise* hsame *rows in R;*
3. *observer-ledger exactness for retained and omitted* BHist *rows;*
4. *downstream handoff through the displayed* Cont *routes and* $\mathsf{StreamName}^{\uparrow}$ *provenance;*
5. *non-escape from host equality of histories, quotient streams, hidden observer state, and unledgered deletion.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 514.1. The packet displays the source history, selected subhistory, omitted-row ledger, transports, observer ledger, continuation routes, provenance, and local naming row before any classifier read is available.

Classifier transport is componentwise. The row R moves selected and omitted rows to the common hsame surface, and C supplies the only read routes through which those transported rows can be consumed.

The remaining obligations follow by finite-row cases on whether a source row is selected into S or recorded in O. The selected case is named by the observer ledger L and may be handed to downstream consumers; the omitted case is also named by L and stops at the boundary reason recorded in O.

No further coordinate is present in the carrier. Thus host history equality, quotient stream equality, hidden observer state, and deletion without a ledger row cannot be exported by the local $\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$ surface. □

Theorem 514.3 (ObserverFilter identity stability)**.** *Let $F = (H, S, O, R, L, C, P, N)$ and $F' = (H', S', O', R', L', C', P', N')$ be accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carriers. If their source rows, selected rows, omitted ledgers, observer ledgers, continuation routes, package rows, and local names are related componentwise by the displayed* hsame *transports, then the two carriers determine the same observer-filter identity judgment. The judgment is computed only from the retained subhistory and omitted-row boundary ledger.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier identity stability`

Proof. Project both carriers to their displayed rows. The comparison data are exactly H, S, O, R, L, C, P, N and $H', S', O', R', L', C', P', N'$.

Use the hsame rows in R and R' to transport corresponding selected rows and omitted rows to a common observer-history surface. The routes in C and C' replay the same source, selection, omission, provenance, and local-name reads after transport.

Now inspect each finite source row. A retained row is compared through the matching entry of S and S', while an omitted row is compared through the matching boundary entry of O and O'. These two cases exhaust the source ledger.

Therefore the classifier has the same retained public rows and the same omitted boundary rows on both sides. No host equality, quotient identity, or hidden observer-state coordinate is available to distinguish the transported packets. □

Theorem 514.4 (ObserverFilter StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier reads only the selected subhistory row S, its observer-ledger entries in L, the componentwise* hsame *transports in R, the displayed* Cont *routes in C, the package provenance P, and the local naming row N. Omitted rows in O may certify the boundary of the handoff, but they do not become stream-name data.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilter streamname handoff`
Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier streamname handoff`

Proof. Start with a consumer route from the accepted filter into $\mathsf{StreamName}^{\uparrow}$. Carrier projection exposes the finite selected row S as the only row permitted to supply stream-name entries.

The route must then pass through L, R, and C. The observer ledger names the retained entries, the transport row moves them by hsame, and the continuation row records the read order used by the downstream stream-name surface.

The omitted ledger O is still visible, but only as a boundary witness. Its entries certify which source rows were not exported; they supply no selected stream-name row and no replacement stream value.

Thus the handoff to $\mathsf{StreamName}^{\uparrow}$ is exactly the displayed selected-subhistory route through S, L, R, C, P, N, with O serving as the finite non-export boundary. □

Theorem 514.5 (ObserverFilter ObserverHistoryIdentity factorization)**.** *Every accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier factors its public identity claim through* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*: the source observer history H, selected subhistory S, omitted-row ledger O,* hsame *transports R, observer ledger L, continuation routes C, provenance P, and local name N provide exactly the observer-history identity surface seen by downstream consumers.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier observer history factorization`

Proof. Read the accepted filter as a finite observer-history packet. The source row H is the displayed history, S and O divide its rows into retained public observations and recorded boundary observations, and L records the local decision for each row.

The rows R and C supply the transport and read routes required by $\mathsf{ObserverHistoryIdentity}^{\uparrow}$. They identify how selected rows and omitted rows are compared without adding a subject coordinate outside the displayed BHist data.

Package provenance and local naming are carried by P and N. Repacking H, S, O, R, L, C, P, N along these roles gives the observer-history identity surface and uses no further identity principle. □

Theorem 514.6 (ObserverFilter ledger exactness)**.** *For every accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier, each source* BHist *row is accounted for exactly once as either a selected row in S or an omitted boundary row in O, and every consumer-facing read factors through the corresponding entry of L, R, C, P, and N. No row may be silently discarded, duplicated into both surfaces, or exported outside the displayed ledger.*

Proof. Induct over the finite source history H. The empty source row has empty selected and omitted ledgers, so the claim is immediate by carrier projection.

For a cons row, inspect the filter-row decision recorded in L. In the selected case the new row is placed in S, transported by R, and made readable only along a route in C with provenance in P and local naming in N.

In the omitted case the new row is placed in O with its boundary reason, transported by R, and made readable only as an omission witness along the displayed route in C.

The induction hypothesis accounts for the preceding source rows. The two filter-row cases are disjoint and exhaustive, so the full source history is partitioned by S and O, and every consumer read passes through the displayed ledger rows. □

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilter ledger exactness`

Theorem 514.7 (ObserverFilter selected-row totality)**.** *For every accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier* $F = (H, S, O, R, L, C, P, N)$*, every selected* BHist *row exported by the filter is witnessed by a source position in* H*, a selected-row entry in* S*, an observer-ledger entry in* L*, and a consumer route through* R, C, P, N*. No selected row can be supplied by hidden observer state, quotient identity, or a source outside the finite carrier of Definition 514.1.*

Proof. Project the accepted carrier $F = (H, S, O, R, L, C, P, N)$ through Definition 514.1. The only public selected-row coordinates are H, S, L, R, C, P, N.

Apply ledger exactness from Definition 514.6. A selected row is one of the two exhaustive cases of the finite source ledger, so it must be paired with its source position in H and its retained entry in S.

The consumer read then factors through R, C, P, N. Since the carrier contains no hidden observer-state row and no quotient identity coordinate, the selected row has no alternate source. □

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier selected row totality`

Theorem 514.8 (ObserverFilter omitted-row boundary exhaustion)**.** *For every accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier* $F = (H, S, O, R, L, C, P, N)$*, every source* BHist *row not selected into* S *appears in the omitted-row ledger* O *with its observer-ledger entry in* L *and its boundary read through* R, C, P, N*. Thus omission exhausts the non-selected part of* H *and admits no unledgered deletion.*

Proof. Read the source history through Definition 514.1. Each row of H is compared against the finite selected and omitted surfaces S and O.

Use Definition 514.6. Its two cases are selected and omitted, and the selected case is already represented in S; therefore every remaining source row must land in O with its corresponding ledger entry in L.

The boundary read is routed by R, C, P, N. Because the omitted case is an explicit carrier row, deleting a non-selected source row without an omitted-ledger entry is outside the accepted $\mathsf{ObserverFilter}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier omitted row boundary exhaustion`

Theorem 514.9 (ObserverFilter StreamName selected-subhistory non-escape)**.** *For every accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *carrier* $F = (H, S, O, R, L, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *handoff is exactly the selected subhistory* S *with its source ledger* L*, componentwise transports* R*, displayed routes* C*, package provenance* P*, and local name* N*. Omitted rows in* O *remain boundary witnesses and cannot become stream-name data, hidden observer state, or quotient history identity.*

Proof. Start from the handoff theorem Definition 514.4. It identifies S, L, R, C, P, N as the only stream-facing rows supplied by the filter.

Apply Definition 514.6 to the same carrier. The finite source rows of H are exhausted by the selected surface S and the omitted boundary O, so every stream-name read is already accounted for by the selected side of the partition.

The omitted side remains visible only as O. Since neither Definition 514.1 nor the handoff theorem carries hidden observer state or quotient history identity, no omitted row can escape into the selected subhistory route. □

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier streamname selected subhistory nonescape`

Theorem 514.10 (ObserverFilter ObserverHistoryIdentity StreamName lattice route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *packet first reads the* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *source row, then the selected subhistory S and omitted-row ledger O, and only then the* $\mathsf{StreamName}^{\uparrow}$ *handoff rows R, C, P, N. Omitted rows remain boundary data and cannot become stream data, hidden observer state, or quotient history identity.*

Proof. Apply Definition 514.5. It factors the accepted packet through the source observer-history row H, selected subhistory S, omitted ledger O, transports R, observer ledger L, routes C, provenance P, and local name N.

Next apply Definition 514.6. The selected and omitted rows exhaust the source history, so every downstream read must pass through either the selected side S or the boundary side O with its ledger entry.

Finally apply Definition 514.9. It identifies the StreamName handoff with the selected subhistory route and keeps omitted rows at the boundary. Thus no omitted row becomes stream data, hidden observer state, or quotient history identity. □

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilter observerhistoryidentity streamname lattice route`

Theorem 514.11 (ObserverFilter budget-selector L10 handoff)**.** *Let $F = (H, S, O, R, L, C, P, N)$ be an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *packet, and let $B = (B_0, S_B, W, D, R_B, E, H_B, C_B, P_B, N_B)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet whose selected schedule row S_B reads its finite* $\mathsf{StreamName}^{\uparrow}$ *windows from the same selected subhistory row S. Then every L10 finite-observation route supplied to a Real-completion consumer factors through*

$$\mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{FiniteObservationBudgetSelector}^{\uparrow}.$$

The selected row S supplies the only StreamName window source for B, while omitted rows in O remain boundary data. No omitted row can become a finite observation window, dyadic observation, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row,* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream, host real equality row, or hidden limit selector.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier budget selector l10 handoff`

Proof. First build the observer-history spine. Apply Definition 514.5 to factor F through the displayed source history H, selected subhistory S, omitted ledger O, transports R, observer ledger L, continuations C, provenance P, and local name N.

Next apply ledger exactness. By Definition 514.6, every source BHist row is accounted for exactly once as either a selected row in S or an omitted boundary row in O. Thus a budget-selector read whose schedule row is tied to S has no second source for finite windows.

Now transport the selected side to StreamName. Use Definition 514.9 and the budget-selector carrier of Definition 782.1. The only window row available to B is the finite $\mathsf{StreamName}^{\uparrow}$ family W selected from S_B, and by hypothesis S_B reads the same selected subhistory row S.

Finally use the NameCert obligations of Definition 782.2. The route from B_0 to S_B, then to W, D, R_B, E, and finally through H_B, C_B, P_B, N_B exhausts the L10 budget route. Since O is already classified as an omitted boundary ledger and never as S, W, D, R_B, E, or a host equality coordinate, coordinate exhaustion keeps every omitted row out of finite observations, dyadic observations, regular handoff rows, Real seals, quotient streams, host real equality, and hidden limit selection. □

Theorem 514.12 (ObserverFilter Real-window synchronizer handoff)**.** *Let $F = (H, S, O, R, L, C, P, N)$ be an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *packet, let $B = (B_0, S_B, W, D, R_B, E, H_B, C_B, P_B, N_B)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet whose selected schedule row S_B reads its finite* $\mathsf{StreamName}^{\uparrow}$ *windows from S, and let Z be an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window source is the same finite observation schedule carried by B. Then every Real-completion-facing synchronizer read factors through*

$$\mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{FiniteObservationBudgetSelector}^{\uparrow} \to \mathsf{RealWindowSyn}$$

The synchronizer route cannot read omitted rows in O, quotient streams, host real equality, hidden observer state, selected external limits, or ambient completeness outside the displayed finite window source.

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier real window synchronizer handoff`

Proof. First build the observer side. Apply Definition 514.5 to expose the source history H, selected row S, omitted boundary row O, transports R, ledger L, routes C, provenance P, and local name N.

Next use ledger exactness and StreamName non-escape. By Definition 514.6 and Definition 514.9, the selected row S is the only StreamName source exported by F, while O remains boundary data rather than stream data or hidden observer state.

Now pass through the budget selector. Apply Definition 514.11. It turns the selected StreamName row into the finite observation schedule carried by B and excludes omitted rows, quotient streams, host real equality, and hidden limit selection from the L10 route.

Finally read the synchronizer carrier and obligations from Definition 979.1 and Definition 979.7. The accepted $\mathsf{RealWindowSynchronizer}^{\uparrow}$ packet can consume only the displayed StreamName and RegSeqRat finite-window source that it receives from the same schedule carried by B. Since neither the budget selector nor the synchronizer carrier contains a coordinate for omitted rows, quotient streams, host real equality, hidden observer state, selected external limits, or ambient completeness, every Real-completion-facing read factors through the displayed route and nowhere else. □

Theorem 514.13 (ObserverFilter Real-window budget exhaustion)**.** *Let $F = (H, S, O, R, L, C, P, N)$ be an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *packet, let $B = (B_0, S_B, W, D, R_B, E, H_B, C_B, P_B, N_B)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet whose selected schedule row S_B reads its finite* $\mathsf{StreamName}^{\uparrow}$ *windows from the selected subhistory row S, and let Z be an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows are read from the same selected subhistory route through B. Then every Real-completion-facing consumer factors through*

$$\mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{FiniteObservationBudgetSelector}^{\uparrow} \to \mathsf{RealWindowSyn}$$

Omitted rows in O cannot become finite budget windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal data, quotient streams, host equality, or hidden limit-selection data.*

Lean4 ✓ `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier real window budget exhaustion`

Proof. First expose the observer source. Apply Definition 514.6 to the accepted packet F. The source history rows are exhausted by the selected subhistory S and the omitted boundary ledger O.

Next identify the selected StreamName route. By Definition 514.9, every $\mathsf{StreamName}^{\uparrow}$ read exported by F is read from S together with L, R, C, P, N. The omitted ledger remains a boundary witness and has no stream, quotient, or hidden observer coordinate.

Now pass from StreamName to the finite budget selector. Apply Definition 514.11. Since S_B reads its finite windows from S, the budget packet B supplies only the displayed rows $W, D, R_B, E, H_B, C_B, P_B, N_B$ to the L10 route.

Finally consume the schedule through the synchronizer. Apply Definition 514.12 to the same F, B, and Z. The synchronizer can read only the displayed StreamName and RegSeqRat finite-window source carried through B.

It remains to exclude every omitted-row escape. The carriers named above contain coordinates for the observer source, selected subhistory, omitted boundary ledger, StreamName route, finite budget schedule, RegSeqRat readback, synchronizer route, provenance, continuations, and local names. None contains a coordinate by which an element of O becomes a budget window, Real seal datum, quotient stream, host equality row, or hidden limit selector. Carrier-coordinate exhaustion therefore forces every Real-completion-facing consumer through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverFilter}^{\uparrow}$ packet is generated from a source BHist row, selected subhistory row, omitted-row ledger, componentwise hsame transports, observer ledger, displayed Cont routes, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 514.1, Definition 514.2, Definition 514.3, Definition 514.4, Definition 514.5, Definition 514.6, Definition 514.7, Definition 514.8, and Definition 514.9 specify the public BHist filter surface, selected-row totality, omitted-row boundary exhaustion, StreamName selected-subhistory non-escape, hsame transport, Cont route, Pkg provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverFilter}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ObserverFilterUp.ObserverFilterCarrier\_streamname\_handoff`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close hidden observer state, quotient identity, unledgered deletion, or bridge checking for $\mathsf{ObserverFilter}^{\uparrow}$.

Upgrade path: The next paper-side step is mature closure examples, or a machine-checked bridge for the displayed ObserverFilter surface.

Chapter 515

A Concrete Naming Certificate for $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$

The $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$ interface turns the three-stratum meta-closure analysis of chapter 9387 into a finite BEDC packet. It records the term, type, and proposition surfaces on which an unwitnessed existential could otherwise be accepted, together with the explicit replacement rows and the axiom-purity ledger that prevents those rows from escaping as Classical.choice, Quot.sound, or propext.

Definition 515.1 (UnwitnessedExistenceClosure carrier)**.** *An* $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, Q, R, C, S, L, H, E, P, N)$$

with the following visible rows:

1. *T is the term-stratum row for displayed positive existential data, read through the replacement surface of Definition 9387.6;*
2. *Q is the type-stratum row for quotient-facing classifier transport, read through the replacement surface of Definition 9387.7;*
3. *R is the proposition-stratum row for bidirectional implication transport, read through the replacement surface of Definition 9387.8;*
4. *C is the classifier row comparing packets only by the displayed term, type, and proposition replacement data;*
5. *S is the stability row saying that componentwise* hsame *transport preserves the three displayed replacement strata;*
6. *L is the axiom-purity ledger row listing exactly the forbidden non-constructive closures* Classical.choice*,* Quot.sound*, and* propext *as non-exportable;*
7. *H is the replacement handoff row routing later consumers to the explicit term recipe, classifier transport, or implication map rather than to an accepted oracle output;*
8. *E stores the finite* hsame *and* Cont *routes for the displayed replacement rows;*
9. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the three replacement surfaces of chapter 9387;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UnwitnessedExistenceClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global choice function, a quotient representative selector, a proposition equality token produced from bidirectional implication alone, or any host axiom outside the displayed ledger.

Lean4 (definition) `BEDC.Derived.UnwitnessedExistenceClosureUp.UnwitnessedExistenceClosureUp`

Theorem 515.2 (UnwitnessedExistenceClosure NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnwitnessedExistenceClosure}}$ *surface for Definition 515.1 consists of five rows:*

1. *carrier admission for the finite packet* $U = (T, Q, R, C, S, L, H, E, P, N)$*;*
2. *classifier comparison through the displayed term, type, and proposition replacement rows;*
3. *stability under componentwise* hsame *and* Cont *transport of* T*,* Q*,* R*,* C*,* S*,* L*, and* H*;*
4. *ledger non-escape for* Classical.choice*,* Quot.sound*, and* propext*;*
5. *replacement handoff to the explicit choice, quotient-transport, and implication-transport surfaces of Definition 9387.6, Definition 9387.7, and Definition 9387.8.*

These five rows are the whole local obligation surface, and every consumer-visible read factors through them.

Proof. Carrier admission is projection from the finite packet of Definition 515.1. The term, type, proposition, classifier, stability, ledger, handoff, transport, package, and naming coordinates are all displayed rows.

The classifier and stability obligations are componentwise. A comparison of two accepted packets may inspect only the three replacement strata and the stored transport rows, and a transported packet is obtained by moving those same rows through the displayed hsame and Cont evidence.

The ledger and handoff obligations are read from L and H. The ledger records the three forbidden non-constructive closure shortcuts as non-exportable, while the handoff row points later consumers to explicit term recipes, displayed classifier transport, and bidirectional implication maps. Since the carrier contains no oracle coordinate, the five listed rows exhaust the local NameCert surface. □

Theorem 515.3 (UnwitnessedExistenceClosure axiom-purity boundary)**.** *Every accepted* UnwitnessedExistenceClos *carrier exports exactly the displayed replacement rows* T*,* Q*, and* R*, the classifier and stability rows* C *and* S*, the ledger row* L*, the handoff row* H*, and the finite provenance rows* E*,* P*, and* N*. It exports no accepted term-stratum choice oracle, no quotient representative selector, no proposition equality token obtained from implication equivalence alone, and no dependency on* Classical.choice*,* Quot.sound*, or* propext*.*

Proof. Inspect the carrier coordinates. The only term-facing coordinate is T, the only quotient-facing coordinate is Q, and the only proposition-facing coordinate is R.

The classifier and stability rows do not add new witnesses; they only compare and transport the displayed replacement data. The finite transport and provenance rows similarly expose routes and package names, not additional logical principles.

Finally read the ledger row L. Its entries mark Classical.choice, Quot.sound, and propext as non-exportable boundary names. Because no carrier coordinate stores the corresponding oracle value, a consumer can receive only the explicit replacement rows and their finite provenance. □

Theorem 515.4 (UnwitnessedExistenceClosure effective replacement handoff)**.** *Every downstream consumer of an accepted* $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$ *packet factors through one of three effective*

handoff routes: the term-stratum route through the displayed positive-existential recipe of Definition 9387.6, the type-stratum route through explicit psame*-respecting classifier transport of Definition 9387.7, or the proposition-stratum route through the selected implication map of Definition 9387.8. No route factors through an unwitnessed existential output.*

Proof. For a term-stratum consumer, project the T and H rows. The handoff points to the displayed recipe and membership evidence named by the replacement-choice surface, so the consumer receives a constructed term row rather than an accepted choice output.

For a type-stratum consumer, project Q and the classifier part of H. The route is the explicit psame-respecting transport named by the quotient replacement, and it does not introduce a quotient representative selector.

For a proposition-stratum consumer, project R and the implication part of H. The route selects one displayed implication direction from the bidirectional datum and composes ordinary proof functions; it never identifies propositions by an equality token. The axiom-purity boundary of Definition 515.3 then rules out any alternate oracle route. □

Theorem 515.5 (UnwitnessedExistenceClosure refutation-boundary link)**.** *Every proposition-stratum read exported by an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet passes through the stable-negation boundary of chapter 507 before the ledger refusal is exported. The implication map in R may carry a displayed refutation direction, but the local ledger may refuse* propext *only after that direction has been read as stable negation rather than as a proposition-equality token.*

Proof. Project the proposition row R and the implication component of H from the effective handoff. These rows expose ordinary proof-function transport, not equality between propositions.

Read the same route through the stable-negation boundary. That sibling records the permitted refutation-facing interpretation of a displayed implication, so the proposition-stratum consumer receives a stable refusal surface before the non-exportable ledger is consulted.

The ledger row L then exports the refusal. Since the carrier never stores a proposition-equality token, the refusal is a boundary read over stable negation and not a hidden use of propext. □

Theorem 515.6 (UnwitnessedExistenceClosure primitive scope)**.** *For an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *carrier $U = (T, Q, R, C, S, L, H, E, P, N)$, the primitive scope is exactly the displayed finite packet: T, Q, R are the three replacement strata, C is the classifier row over those strata, S is the componentwise stability row, L is the non-exportable ledger, H is the replacement handoff, E records* hsame *and* Cont *transport, P is the* Pkg *provenance row, and N is the local* NameCert *row. The rows bind only* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the three replacement surfaces named in Definition 9387.6, Definition 9387.7, and Definition 9387.8.*

Proof. Project the carrier. The coordinate list of Definition 515.1 supplies the term, type, proposition, classifier, stability, ledger, handoff, transport, package, and naming rows directly.

Apply the NameCert obligations. Carrier admission, classifier comparison, stability transport, ledger non-escape, and replacement handoff are all read from those same coordinates, so no primitive outside the listed BHist, hsame, Cont, Pkg, NameCert, and replacement-surface rows is introduced. □

Theorem 515.7 (UnwitnessedExistenceClosure ledger refusal)**.** *For every accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *carrier, the ledger row L refuses* Classical.choice*,* Quot.sound*, and* propext *exactly at the carrier boundary: no coordinate of $U = (T, Q, R, C, S, L, H, E, P, N)$ stores a global choice function, quotient representative selector, proposition equality token obtained from implication equivalence alone, or host axiom output.*

Proof. Inspect the positive replacement rows first. The term route exposes T, the quotient-facing route exposes Q, and the proposition-facing route exposes R; each is a displayed replacement surface rather than an oracle output.

Read the administrative rows next. The classifier, stability, handoff, transport, provenance, and naming coordinates preserve or route the displayed replacement data, but none of them stores a global choice function, quotient representative selector, or proposition equality token.

The remaining coordinate is the ledger L. By the axiom-purity boundary it names Classical.choice, Quot.sound, and propext only as non-exportable rows, so the refusal is exact at the finite carrier boundary. □

Theorem 515.8 (UnwitnessedExistenceClosure scoped boundary package)**.** *The carrier definition, NameCert obligation theorem, axiom-purity boundary, effective replacement handoff, primitive-scope theorem, and ledger-refusal theorem jointly bind* UnwitnessedExistenceClosure$^{\uparrow}$ *as a scoped finite package over the three replacement strata T, Q, R and the rows C, S, L, H, E, P, N.*

The package closes only the displayed replacement boundary. It does not close a global choice principle, quotient representatives, proposition extensionality, host axioms, bridge checking, or any consumer route outside the finite BHist *packet.*

Proof. Start with the carrier and NameCert obligation theorem. They expose the ten visible rows and state that carrier admission, classifier comparison, stability, ledger non-escape, and replacement handoff are exhausted by those rows.

Add the axiom-purity boundary and effective replacement handoff. These identify the three allowed consumer routes and exclude the corresponding host-oracle exits.

Finally apply primitive scope and ledger refusal. The primitive-scope theorem fixes the rows as BHist, hsame, Cont, Pkg, NameCert, and replacement-surface data, while ledger refusal blocks the three forbidden non-constructive closures at the carrier boundary. The resulting package is scoped exactly to the displayed finite packet. □

Theorem 515.9 (UnwitnessedExistenceClosure replacement surface totality)**.** *Every accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *carrier read chooses exactly one of the three displayed replacement surfaces: the term surface T for positive existential data, the type surface Q for quotient-facing classifier transport, or the proposition surface R for implication transport. No accepted read lands in a host choice row, a host quotient row, a proposition-extensionality row, or any unlisted oracle coordinate.*

Proof. Project an accepted packet $U = (T, Q, R, C, S, L, H, E, P, N)$. Its consumer-facing logical rows are precisely T, Q, and R, while C, S, L, H, E, P, and N only classify, transport, record non-export, route handoff, or store provenance for those rows.

A term request is therefore read from T and the term component of H. A quotient or classifier request is read from Q together with C and S. A proposition request is read from R and the proposition component of H.

The axiom-purity boundary of Definition 515.3 removes every alternate row named by Classical.choice, Quot.sound, or propext. Thus the three displayed replacement surfaces are total for accepted reads and exclusive of host-oracle surfaces. □

Theorem 515.10 (UnwitnessedExistenceClosure stability transport)**.** *Componentwise* hsame *and* Cont *transport of an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet preserves the term, type, and proposition replacement strata T, Q, R, the classifier row C, the stability row S, the ledger row L, the handoff row H, and the provenance rows E, P, N.*

Proof. Use the displayed transport routes stored in E. They move each visible carrier coordinate componentwise, so the transported packet has corresponding rows $T, Q, R, C, S, L, H, E, P, N$ with the same row roles.

The classifier row C is preserved because it compares only the transported replacement strata. The stability row S is preserved because its content is exactly the componentwise hsame and Cont closure condition for those strata.

The ledger and handoff rows are transported as ledger and handoff rows, not recomputed through host principles. The provenance rows E, P, N record the same finite route, package, and local naming evidence after transport, so no displayed obligation row is lost. □

Theorem 515.11 (UnwitnessedExistenceClosure ledger handoff exhaustion)**.** *The axiom-purity ledger row L and the handoff row H exhaust all downstream reads of an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet through the explicit replacement-choice, replacement-quotient, and replacement-propext surfaces. A downstream consumer has no further route to an unwitnessed existential result, a quotient selector, or a proposition equality token.*

Proof. Begin with the effective handoff of Definition 515.4. It sends each downstream consumer to the term recipe, the explicit classifier transport, or the selected implication map.

The ledger row L then blocks the three forbidden exits. It names Classical.choice, Quot.sound, and propext only as non-exportable boundaries, and the carrier contains no coordinate storing their host-level outputs.

Combining H with L accounts for every consumer-visible logical read. The remaining rows classify, stabilize, transport, package, or name the same replacement data, so they cannot create an additional downstream oracle route. □

Theorem 515.12 (UnwitnessedExistenceClosure term replacement route)**.** *Let $U = (T, Q, R, C, S, L, H, E, P, N)$ be an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet. Every term-stratum read that asks for positive existential data is routed through the displayed row T, the handoff row H, and the replacement-choice surface of Definition 9387.6. The accepted packet therefore exposes a finite witness recipe and its membership evidence, not a global choice function and not an unwitnessed existential result.*

Proof. Project the term row T and the term-facing component of H. These rows are part of the carrier inventory, so the consumer can inspect only the displayed recipe, the named membership evidence, and the finite hsame and Cont routes stored in E.

The replacement-choice surface supplies the allowed route from the positive existential request to a constructed term row. Since L lists Classical.choice as non-exportable and the carrier contains no choice-function coordinate, the term consumer cannot bypass the displayed route. □

Theorem 515.13 (UnwitnessedExistenceClosure quotient replacement route)**.** *Let $U = (T, Q, R, C, S, L, H, E, P, N)$ be an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet. Every quotient-facing or classifier-facing read is routed through the displayed row Q, the classifier row C, the stability row S, and the replacement-quotient surface of Definition 9387.7. The route transports classifiers by explicit* psame*-respecting data and exports no quotient representative selector.*

Proof. Project Q, C, S and the quotient-facing component of H. The classifier may compare accepted packets only by the displayed replacement rows and the componentwise transports attached to them.

The replacement-quotient surface provides the permitted classifier transport. The ledger row L marks Quot.sound as non-exportable, and no coordinate of U stores a representative selector. Thus the quotient-facing consumer receives only the explicit transport row and its finite provenance. □

Theorem 515.14 (UnwitnessedExistenceClosure proposition replacement route)**.** *Let* $U = (T, Q, R, C, S, L, H, E,$ *be an accepted* UnwitnessedExistenceClosure$^{\uparrow}$ *packet. Every proposition-stratum read is routed through the displayed row* R*, the proposition-facing component of* H*, and the replacement-propext surface of Definition 9387.8. The route may compose the selected implication maps, but it exports no proposition equality token obtained from bidirectional implication alone.*

Proof. Project R and the proposition component of H. These rows name the two implication directions and the selected ordinary proof-function route available to a downstream consumer.

The replacement-propext surface permits implication transport, not proposition equality. Since L records propext as non-exportable and U has no coordinate for an equality token, the proposition-facing read stays inside the displayed implication route. □

Theorem 515.15 (UnwitnessedExistenceClosure TasteGate boundary)**.** *The* UnwitnessedExistenceClosure$^{\uparrow}$ *chapter gate accepts the finite packet only at the displayed replacement and ledger boundary: term reads pass through* T*, quotient and classifier reads pass through* Q, C, S*, proposition reads pass through* R*, and forbidden axiom ancestry is confined to the non-exportable ledger row* L*. A packet missing one of those displayed replacement rows, or asking to export* Classical.choice*,* Quot.sound*, or* propext*, is outside the gate.*

Lean4 ✓ `BEDC.Derived.UnwitnessedExistenceClosureUp.taste gate`

Proof. Read the chapter gate against the carrier coordinates. The accepted surface has exactly the term, quotient, proposition, classifier, stability, ledger, handoff, transport, provenance, and naming rows listed in Definition 515.1.

The three replacement-route theorems isolate the only consumer-facing routes. The axiom-purity boundary then prevents a consumer from replacing those routes by a choice function, quotient selector, or proposition-extensionality equality token. The Lean-checked gate records the same carrier-level acceptance boundary for this chapter. □

Theorem 515.16 (UnwitnessedExistenceClosure bridge-facing replacement surface)**.** *Every bridge-facing consumer of the public* UnwitnessedExistenceClosure$^{\uparrow}$ *replacement surface must first read the term row* T*, quotient row* Q*, proposition row* R*, classifier and stability rows* C, S*, and axiom-ledger row* L *through the TasteGate boundary grounded by* BEDC.Derived.UnwitnessedExistenceClosureUp.taste_gate*. The exported bridge route is therefore the finite replacement surface over* $T, Q, R, C, S, L, H, E, P, N$*; it exports no choice function, quotient representative selector, or proposition-extensionality equality token.*

Proof. Begin with the TasteGate boundary of Definition 515.15. It accepts exactly the displayed replacement and ledger rows, and it rejects any packet that asks to export Classical.choice, Quot.sound, or propext.

The three replacement-route theorems then fix the bridge-facing channels. Term consumers pass through Definition 515.12; quotient and classifier consumers pass through Definition 515.13; proposition consumers pass through Definition 515.14.

Thus the paper-side bridge boundary is exactly the public replacement surface already accepted by the TasteGate. It has no carrier coordinate for a global choice function, quotient representatives, or proposition equality obtained from implication transport. □

Falsifiable prediction: Deleting any one of T, Q, R, L admits a concrete rejected packet: a term consumer without the replacement-choice recipe, a quotient consumer without explicit classifier

transport, a proposition consumer without implication transport, or a ledger consumer that cannot refuse forbidden axiom ancestry.

Independence witness: propexttransportboundary, quotientsoundnessboundary, forbiddenaxiomancestry, axiomqueryledger: no listed sibling is bijective to $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$ because this carrier stores all three replacement routes together with one non-exportable axiom ledger, while each sibling isolates a single boundary, ancestry, or query surface.

Closure status for $\mathsf{NameCert}_{\mathsf{UnwitnessedExistenceClosure}^{\uparrow}}$.

Constructive story: An $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$ packet is generated from term, type, and proposition replacement rows, a componentwise classifier, stability transports, an axiom-purity ledger, a replacement handoff row, finite hsame and Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnwitnessedExistenceClosure}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The finite carrier, five-row NameCert surface, axiom-purity boundary, effective replacement handoff, replacement-surface totality theorem, refutation-boundary link, TasteGate boundary, the three term, quotient, and proposition replacement routes, and the bridge-facing replacement surface are closed as the public bridge route over $T, Q, R, C, S, L, H, E, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnwitnessedExistenceClosure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UnwitnessedExistenceClosureUp.taste\_gate`

Bridge status: none

Not claimed: This chapter does not close bridge checking, a global choice principle, quotient representatives, proposition extensionality, host axioms, or the full scoped NameCert implementation.

Upgrade path: No further theory-axis upgrade is claimed beyond maintaining the same bridge-facing replacement and ledger boundary.

Chapter 516

A Concrete Naming Certificate for $\mathsf{EmptyFableLedger}^{\uparrow}$

The $\mathsf{EmptyFableLedger}^{\uparrow}$ interface is the concrete NameCert landing surface for the Empty Fable Machine of chapter 9399. It packages the generated Hist row, the trace read from that history, the realized selector-step witnesses, the fable-ledger row, and the local naming boundary that prevents the ledger from being read as a global chooser.

Definition 516.1 (EmptyFableLedger NameCert carrier)**.** *An* $\mathsf{EmptyFableLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (H, T, S, F, C, P, N)$$

with the following displayed coordinates:

1. *H is the* Hist *source row generated from* e *by finitely many* Ext *steps;*
2. *T is the* Trace *row read from the constructor spine of H;*
3. *S is the finite witness list of* SelStep *rows, one for each non-empty trace edge, each exposing the enacted mark, the* Ext *step, and the local branch certificate;*
4. *F is the* FableLedger *row identifying the same finite selector ledger with the Empty-fable reading of H;*
5. *C records the displayed* Cont *routes by which a consumer reads H, T, each selector witness in S, and the fable-ledger row F;*
6. *P is the* Pkg *provenance naming the* Hist, Ext, Trace, SelStep, FableLedger, hsame, Cont, Pkg, *and* NameCert *source surfaces;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{EmptyFableLedger}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by the displayed history row, trace row, selector-witness list, fable-ledger row, continuation routes, package provenance, local naming row, and the componentwise hsame *transports attached to those rows. It contains no coordinate for host sequence equality, global branch selection, a truth oracle, or an undisplayed selector.*

Lean4 (definition) `BEDC.Derived.EmptyFableLedgerUp.TasteGate.EmptyFableLedgerUp`
Lean4 ✓ `BEDC.Derived.EmptyFableLedgerUp.TasteGate.emptyFableLedgerTasteGate`

Theorem 516.2 (EmptyFableLedger NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{EmptyFableLedger}^{\uparrow}}$ *surface for the carrier of Definition 516.1 consists exactly of carrier admission for E, trace admission for T, selector-step admission for every row of S, fable-ledger admission for F, continuation reads through C, package provenance through P, the local naming row N, and componentwise* hsame *transport for the displayed rows.*

Every obligation is read from the Empty Fable Machine definitions of Trace*,* SelStep*, and* FableLedger*, together with the core* Hist*,* hsame*,* Ext*,* Cont*,* Pkg*, and* NameCert *chapters. No obligation introduces a hidden branch selector.*

Proof. Carrier admission is projection from the finite packet E. The history row, trace row, selector-witness list, fable-ledger row, continuation routes, package provenance, and local naming row are all displayed coordinates of the carrier.

Trace admission uses the Empty Fable Machine trace definition, so the row T is read from the e, E0, and E1 constructor spine of H. Selector admission uses the SelStep definition: each row of S exposes a mark, an Ext step, and a local branch certificate.

The fable-ledger row F is the same trace surface read under the fable-machine name. The Cont routes in C and the provenance row P keep these reads inside the cited kernel and naming surfaces, while N records the local certificate. Since the packet has no further selector coordinate, no hidden branch chooser can be exported. □

Theorem 516.3 (EmptyFableLedger selector exactness)**.** *Let $E = (H, T, S, F, C, P, N)$ be an accepted* $\mathsf{EmptyFableLedger}^{\uparrow}$ *carrier. Every selector read exported from E is exact: the selected mark for a non-empty trace edge is read from a displayed* SelStep *row in S, that row contains the corresponding* Ext *step, and the accumulated trace agrees with the fable-ledger row F up to the* hsame *transports recorded for H.*

Thus a downstream consumer receives the concrete Empty-fable selector ledger as a finite packet over Hist*,* Ext*,* Trace*,* SelStep*,* FableLedger*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Project the selector-witness list S. For the empty trace there is no non-empty edge and the ledger read is the empty constructor spine above e.

For a non-empty trace edge, the corresponding SelStep row exposes the enacted mark, the Ext step from the previous history to the next history, and the local branch certificate. This is exactly the selector data used by the Empty Fable Machine chapter.

The trace row T and fable-ledger row F are then compared through the displayed hsame transport attached to H and read through C. Package provenance and the local NameCert row repack the same route for downstream consumers, so the exported selector is the displayed witness, not an external function. □

Theorem 516.4 (EmptyFableLedger selector non-escape)**.** *An accepted* $\mathsf{EmptyFableLedger}^{\uparrow}$ *packet cannot export a selector mark unless that mark appears in the displayed trace row. The accepted packet consumes the finite selector spine of chapter 494, the enacted local branch witness of chapter 685, and the observer-Hist identity surface of chapter 479 before any selector read is made available to a downstream consumer.*

Proof. Start with the finite carrier of Definition 516.1. The only selector-facing coordinates are the trace row T, the selector-step list S, and the fable-ledger row F.

The selector spine is the sibling row supplied by chapter 494. The enacted local branch is the one-step boundary row supplied by chapter 685. These two rows agree with the observer history through the identity reading of chapter 479.

Thus any exported selector mark must pass through the displayed trace row and one displayed selector-step witness. A mark outside T would require an additional selector coordinate, but the packet exposes only H, T, S, F, C, P, N and their componentwise transport rows. □

Theorem 516.5 (EmptyFableLedger non-chooser boundary)**.** *Every consumer read of an accepted* $\mathsf{EmptyFableLedger}^{\uparrow}$ *carrier factors through the displayed tuple* H, T, S, F, C, P, N *and the componentwise* hsame *transports for those rows. In particular, the packet exports no* Classical.choice *instance, no host sequence equality for traces, no truth oracle, and no hidden branch selector.*

The only branch-facing data are the finite SelStep *witnesses already present in* S*; the only sequence-facing data are the* Trace *and* FableLedger *rows already present in* T *and* F*.*

Proof. Inspect the carrier inventory. Branch information appears only in S, and each row of S is a visible selector-step witness rather than a function that chooses among all admissible branches.

Sequence information appears only in T and F. These rows are transported through hsame and read through the continuation routes in C, so a consumer never receives host sequence equality as an additional principle.

The package row P and certificate row N record the same local dependency surface. Since no carrier coordinate names Classical.choice, a truth oracle, host sequence equality, or an undisplayed branch selector, no such datum is exported by the NameCert boundary. □

Theorem 516.6 (EmptyFableLedger continuation-trace dependency)**.** *Let* $E = (H, T, S, F, C, P, N)$ *be an accepted* $\mathsf{EmptyFableLedger}^{\uparrow}$ *carrier. Every accepted trace row* T *and continuation-route row* C *are read through the finite trace replay discipline of chapter 469 before the selector ledger is exposed. Thus the fable ledger is a consumer of the sibling finite trace route, not a separate trace oracle.*

Proof. Project the displayed rows H, T, S, F, C, P, N. The trace row T is read from the constructor spine of H, and the continuation row C records the displayed routes by which a consumer reads that trace, the selector-step witnesses, and the fable-ledger row.

Now use the finite trace replay discipline of chapter 469. It fixes the trace consumer to the displayed history row and its continuation route before any downstream ledger can inspect selected marks.

The selector ledger is then exposed only through the already replayed trace route and the rows S and F. Since no coordinate of E stores a trace oracle or host sequence equality, the fable-ledger read is a consumer of the sibling continuation-trace surface. □

Falsifiable prediction: An accepted EmptyFableLedgerUp packet cannot export a selector mark unless that mark appears in the displayed trace row.

Independence witness: SelectorLedger, EmptyBoundarySelector, ObserverHistIdentity, InscriptionEvent: the fable ledger is not bijective to any sibling because it jointly binds trace readback, local selector enactment, and story-generation provenance in one packet, whereas each sibling owns only one of those rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EmptyFableLedger}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EmptyFableLedger}^{\uparrow}$ packet is generated from a displayed Hist row, its Trace row, finite SelStep witnesses, a FableLedger row, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmptyFableLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite carrier, classifier, selector exactness route, selector non-escape boundary, and non-chooser boundary for Empty-fable ledgers over Hist, Ext, Trace, SelStep, FableLedger, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EmptyFableLedger}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EmptyFableLedgerUp.TasteGate.emptyFableLedgerTasteGate`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close bridge checking, global branch selection, host sequence equality, truth oracles, or choice principles.
Upgrade path: Scoped closure requires checked carrier admission, selector exactness, trace-to-fable transport, continuation routing, package provenance, and non-chooser rows for the displayed finite packet.

Chapter 517

A Concrete Naming Certificate for $\mathsf{RussellBoundary}^{\uparrow}$

The $\mathsf{RussellBoundary}^{\uparrow}$ surface turns the Russell problem boundary of chapter 9457 into a finite NameCert packet. It records relation-first observation rows, description-expansion obligations, contextual class readings, stratification bridges, witnessed-descent routes, componentwise transports, package provenance, and a local naming ledger. The packet does not add a class ontology, a reducibility principle, a quotient over certificates, a host choice selector, or an identity between strata. The witnessed-descent row W is consumed as a public residue ledger in the same sense as chapter 520, while $\mathsf{RussellBoundary}^{\uparrow}$ keeps the stratification and contextual-class boundary separate from the inscription residue packet.

The packet also consumes the $\mathsf{NameCertMorphism}^{\uparrow}$ sibling route of chapter 481. Relation-first rows and witnessed-descent ledgers are read as morphism-preserving certificate boundaries, not as a standalone class ontology.

The mature consumer surface is the public certificate route itself: downstream foundation-boundary packets may cite the relation-first observation, description-expansion obligation, contextual-class reading, stratification bridge, witnessed-descent ledger, $\mathsf{NameCertMorphism}^{\uparrow}$ route, and inscription-gap residue only through the displayed rows of Definition 517.5.

Definition 517.1 (RussellBoundary carrier)**.** *A* $\mathsf{RussellBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, O, D, K, L, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-expression row whose apparent object grammar is being inspected;*
2. *O is the relation-first observation row, recording the BEDC relation that licenses the public name before any object-like reading is consumed;*
3. *D is the description-expansion row, listing the existence, uniqueness or stability, and ledger obligations attached to the definite-description use of S;*
4. *K is the contextual-class row, reading class-like grammar through the finite classifier context that admits it;*
5. *L is the stratification row, recording the source stratum, target stratum, and displayed bridge through which any level passage is consumed;*

6. *W is the witnessed-descent row, a finite list of certificate routes required before an upward name may be read downward by a consumer;*
7. *H records componentwise* hsame *transports for S, O, D, K, L, W, package rows, and the local naming row;*
8. *C records the displayed* Cont *routes that expose the source expression, replay the relation, unfold the description, read the classifier context, consume the bridge, and check the descent witnesses;*
9. *P is the* Pkg *provenance over* BHist, Hist, Cont, hsame, msame, *bridge rows, and the surrounding certificate rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RussellBoundary}^{\uparrow}}$ *ledger.*

The carrier is exactly this finite packet. It has no coordinate for a class object beyond its contextual classifier row, no primitive denoting description, no reducibility-style stratum collapse, no host quotient over certificate rows, and no choice selector for unlisted descent witnesses.

Lean4 ✓ `BEDC.Derived.RussellBoundaryUp.RussellBoundaryCarrier source relation description transport`

Theorem 517.2 (RussellBoundary fivefold obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RussellBoundary}^{\uparrow}}$ *surface binds five obligations: relation-first licensing through O, description expansion through D, contextual class reading through K, stratification by the displayed bridge row L, and witnessed descent through the route ledger W.*

Proof. Carrier exposure is projection from Definition 517.1. The rows $S, O, D, K, L, W, H, C, P, N$ are the only displayed data admitted by the packet.

Relation-first licensing is the row O. A consumer reads the public name through the relation recorded there, and the continuation routes in C replay that relation before any object-like use is available.

Description expansion is the row D. It displays the obligations attached to the description rather than treating the description as a primitive denoting term.

Contextual class reading is the row K. Class-like grammar is consumed only through the finite classifier context recorded in that row.

Stratification and witnessed descent are the rows L and W. The bridge row records the level passage, while the descent ledger records the finite certificate routes required for downward use. Together with H, P, N, these rows form the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RussellBoundaryUp.RussellBoundaryCarrier fivefold obligations`

Theorem 517.3 (RussellBoundary NameCert noncollapse)**.** *Every accepted* $\mathsf{RussellBoundary}^{\uparrow}$ *packet keeps source expressions, contextual classifier rows, bridge rows, and witnessed-descent rows as distinct displayed coordinates. Hence the packet cannot be read as an identity between a class expression and a class object, between a description and a primitive denotation, or between source and target strata.*

Proof. Project an accepted carrier as $R = (S, O, D, K, L, W, H, C, P, N)$. The source expression, relation row, description row, contextual-class row, stratum bridge, and descent ledger occupy separate coordinates.

The classifier context cannot collapse into a class object, because the only class-like coordinate is K, and K is read through the continuation routes and transports recorded in C and H. No coordinate stores an object obtained by forgetting that context.

The description row cannot collapse into primitive denotation, because D records obligations rather than a bare selected object. A consumer of the description must also consume the displayed rows that account for existence, stability, and ledger use.

The source and target strata cannot collapse, because L records the bridge as a bridge row. The packet exposes a consumable passage between strata, not an equality of the strata themselves. □

Lean4 ✓ `BEDC.Derived.RussellBoundaryUp.RussellBoundaryCarrier namecert noncollapse`

Theorem 517.4 (RussellBoundary witnessed-descent ledger)**.** *Any downward consumer read of a* $\mathsf{RussellBoundary}^{\uparrow}$ *upward name factors through the witnessed-descent row W and the displayed* Cont *routes in C. Therefore the packet admits downward use only when the relevant certificate route is present in the finite ledger.*

Proof. Let a consumer attempt a downward read of an accepted packet. The only descent coordinate in Definition 517.1 is W, and the only executable routes available to the packet are the displayed continuation rows in C.

If the read is admitted, it must project a route from W and replay it through C. The componentwise transports in H ensure that the source expression, relation row, description row, contextual classifier, and bridge row are the same displayed data after transport.

If no such route is present in W, the carrier has no substitute coordinate from which to manufacture a witness. The packet has no host choice selector, no class-level truth branch, and no quotient that can identify an absent witness with a present one. Thus downward use is ledger-bounded. □

Lean4 ✓ `BEDC.Derived.RussellBoundaryUp.RussellBoundaryCarrier witnessed descent ledger`

Theorem 517.5 (RussellBoundary public certificate)**.** *The carrier, fivefold obligations, noncollapse boundary, and witnessed-descent ledger assemble a public* $\mathsf{RussellBoundary}^{\uparrow}$ *certificate for relation-first, description-expansion, contextual-class, stratification, and witnessed-descent discipline over finite BEDC rows.*

Proof. Start with the carrier of Definition 517.1. It fixes the finite source, relation, description, contextual classifier, bridge, descent, transport, continuation, package, and naming rows.

Apply Definition 517.2. It identifies the five public obligations by their rows O, D, K, L, W.

Use Definition 517.3 for the boundary side: the packet does not identify class grammar with a class object, description grammar with primitive denotation, or bridge consumption with stratum equality.

Use Definition 517.4 for the consumer side: downward use is admitted only through recorded certificate routes.

These clauses expose a public certificate surface. It is public because every consumer-facing use is one of the displayed rows or a route through those rows, and it is a Russell boundary because the five restrictions of section 9457.1 are enforced by the NameCert packet rather than by an external ontology. □

Lean4 ✓ `BEDC.Derived.RussellBoundaryUp.RussellBoundaryCarrier public certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{RussellBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RussellBoundary}^{\uparrow}$ packet is generated from a source-expression row, a relation-first observation row, a description-expansion obligation row, a contextual-class row, a stratification bridge row, a witnessed-descent ledger, componentwise transports, displayed continuation routes, package provenance, and a local naming ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RussellBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 517.1, Definition 517.2, Definition 517.3, Definition 517.4, and the public export surface of Definition 517.5 bind the mature finite scope of relation-first rows, description expansion, contextual class readings, stratification bridge, witnessed descent, $\mathsf{NameCertMorphism}^{\uparrow}$ routing, inscription-gap ledger consumption, and downstream foundation-boundary citation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RussellBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge checking, class ontology, reducibility principle, quotient over certificates, host choice selector, identity between strata, or a global foundation ontology.

Upgrade path: The paper theory axis is mature through the public certificate route; Lean verification and bridge checking remain independent axes.

Chapter 518

A Concrete Naming Certificate for $\mathsf{RussellAnalysisLadder}^{\uparrow}$

Falsifiable prediction: Every accepted RussellAnalysisLadderUp packet exposes description, contextual-class, Russell-boundary, self-membership, type-classifier membership, stratification, and witnessed-descent rows before any consumer may read a description as a public name.

Independence witness: definitedescriptionboundary, contextualclassreading, russellboundary, russellselfmembershipboundary, typeclassifiermembership: none is bijective because the ladder carrier consumes these rows together with stratification and witnessed-descent evidence, while each sibling isolates one description, contextual, boundary, self-membership, or type-membership surface.

The $\mathsf{RussellAnalysisLadder}^{\uparrow}$ surface turns the Russell problem boundary of chapter 9457 into a finite NameCert packet. It records the five kernel-facing analysis roles as displayed rows: description expansion, contextual class reading, relation-first classifier exposure, stratification control, and witnessed descent. The packet does not import Russell's logicist axioms, class ontology, reducibility principle, host quotient principles, or any global construction of analysis.

The ladder consumes the description boundary, contextual-class reading, Russell boundary, and type-classifier membership surfaces via chapter 661, chapter 684, chapter 517, and chapter 476. It also reads the self-membership boundary of chapter 506. These dependencies are row consumption inside the finite packet, not external ontology.

Definition 518.1 (RussellAnalysisLadder carrier)**.** *A* $\mathsf{RussellAnalysisLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, Q, A, S, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the description row, recording the analysed source phrase and the local existence, stability, and ledger obligations needed before it may be read as a name;*
2. *Q is the contextual-class row, recording the context that classifies the phrase without promoting that context to a class object;*
3. *A is the relation-first row, recording the relation or pattern that licenses the behaviour of the name before any object-like reading is consumed;*
4. *S is the stratification row, recording the level at which the name is admitted and the explicit bridge by which any higher or lower read is requested;*

5. *W is the witnessed-descent row, recording the certificate evidence required before an upward name is used downward by a consumer;*
6. *H stores the componentwise* hsame *transports for D, Q, A, S, W and for the surrounding package and local naming rows;*
7. *C stores the displayed* Cont *routes by which consumers read the description, context, relation, stratum, and witnessed-descent rows;*
8. *P is the* Pkg *provenance over* BHist, Hist, Ask, msame, hsame, Cont, *relation rows, bridge rows, and local naming evidence;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RussellAnalysisLadder}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It has no coordinate for a primitive class object, an unanalysed definite description, a unary property replacing a relation, a level collapse, a reducibility-style descent, a host quotient, or a global analytic universe.

Lean4 ✓ `BEDC.Derived.RussellAnalysisLadderUp.RussellAnalysisLadderTasteGate single carrier alignment`

Theorem 518.2 (RussellAnalysisLadder NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RussellAnalysisLadder}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *description exposure: every accepted carrier exposes the description row D together with the displayed obligations that license name use;*
2. *contextual-class locality: every class-like read factors through the context row Q and does not export a class object;*
3. *relation-first classifier stability: every classifier read factors through the relation row A and is transported only by the displayed* hsame *data;*
4. *stratification accountability: every cross-level use is routed through S, C, P, and N rather than through a level collapse;*
5. *witnessed descent: every downward consumer of an upward name presents the evidence row W and the local naming row N.*

Proof. Carrier exposure is projection from the finite tuple of Definition 518.1. The rows $D, Q, A, S, W, H, C, P, N$ are the only data admitted to the packet.

The contextual and relation-first obligations are read from Q and A. A consumer of a class-like phrase must pass through the displayed context, and a consumer of the name's behaviour must pass through the relation row before obtaining any object-like public use.

Stratification and witnessed descent are then read from S and W. The only routes available to a consumer are the displayed Cont rows in C, transported through H and recorded by the provenance and local naming rows P and N.

Since the carrier has no coordinate beyond these rows, the certificate cannot export a primitive class, unanalysed description, unary replacement of the relation, level collapse, or unwitnessed downward use. □

Lean4 ✓ `BEDC.Derived.RussellAnalysisLadderUp.RussellAnalysisLadderCarrier namecert obligations`

Theorem 518.3 (RussellAnalysisLadder description factorization)**.** *Every public read of a description in an accepted* $\mathsf{RussellAnalysisLadder}^{\uparrow}$ *carrier factors through the finite rows $D, Q, A, S, W, H, C, P, N$.*

In particular, an apparent denoting phrase is consumed only after the source description, classifying context, relation row, stratification row, witnessed-descent evidence, transport rows, continuation routes, provenance row, and local naming row have been displayed.

Proof. Begin with the carrier of Definition 518.1. The description row D is present only together with Q, A, S, W, H, C, P, N.

Apply the NameCert obligations of Definition 518.2. Description exposure gives D, contextual-class locality routes class-like uses through Q, and relation-first classifier stability routes behavioural uses through A.

The remaining public read is accountable only through S and W. Transport and continuation rows H and C record how the consumer reaches the displayed name, while P and N record the package and local certificate.

Thus the denoting phrase is not a primitive coordinate. Its public use factors through the complete finite packet. □

Lean4 ✓ `BEDC.Derived.RussellAnalysisLadderUp.RussellAnalysisLadderCarrier description factorization`

Theorem 518.4 (RussellAnalysisLadder non-escape boundary)**.** *Every consumer read of an accepted* $\mathsf{RussellAnalysisLadder}^{\uparrow}$ *packet remains inside the finite description, contextual-class, relation-first, stratification, witnessed-descent, transport, continuation, provenance, and local naming rows. The packet cannot be read as Russellian class ontology, an imported reducibility principle, host quotient reasoning, proposition extensionality, proof irrelevance, or a completed host construction of analysis.*

Proof. Project the accepted packet as $R = (D, Q, A, S, W, H, C, P, N)$. A consumer can inspect only these displayed rows or a component transported through H and routed through C.

Use Definition 518.3. Any description read returns to the full packet, so a denoting phrase cannot bypass the contextual, relational, stratification, witnessed-descent, package, and local naming obligations.

The excluded principles require coordinates absent from the carrier: a primitive class object, a level-collapsing reducibility rule, a host quotient law, proposition extensionality, proof irrelevance, or a completed analytic universe. Since none of these coordinates occurs in R, no consumer can extract them from $\mathsf{RussellAnalysisLadder}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.RussellAnalysisLadderUp.RussellAnalysisLadderCarrier non escape boundary`

Theorem 518.5 (RussellAnalysisLadder public export)**.** *Every public consumer of an accepted* $\mathsf{RussellAnalysisLadder}^{\uparrow}$ *packet receives only the finite description row, contextual-class row, relation-first row, stratification row, witnessed-descent row, componentwise transport rows, continuation-route rows, package-provenance row, and local naming row of* $R = (D, Q, A, S, W, H, C, P, N)$*. The public export gives no class ontology, reducibility principle, quotient reasoning, proposition extensionality, proof irrelevance, or host analytic universe.*

Proof. Project the carrier of Definition 518.1. Its public-facing rows are exactly $D, Q, A, S, W, H, C, P, N$, so a consumer can read the description only together with the contextual, relational, stratification, witnessed-descent, transport, continuation, provenance, and local naming data.

Use the NameCert obligations of Definition 518.2. Description exposure, contextual-class locality, relation-first classifier stability, stratification accountability, and witnessed descent account for every public use of the displayed packet.

Apply description factorization and the non-escape boundary. The factorization theorem keeps each public description read inside the finite carrier rows, and the boundary theorem excludes class ontology, reducibility, quotient reasoning, proposition extensionality, proof irrelevance, and host analytic completion. Thus the exported public surface is the finite packet itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{RussellAnalysisLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{RussellAnalysisLadder}^{\uparrow}$ packet is generated from description, contextual-class, relation-first, stratification, witnessed-descent, componentwise transport, continuation-route, package-provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RussellAnalysisLadder}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 518.1, Definition 518.2, Definition 518.3, Definition 518.4, and Definition 518.5 bind the finite Russell problem boundary obligations, public export, and paper-side finite-description bridge to the NameCert and TasteGate carrier surface over BHist, hsame, Cont, Pkg, and witnessed-descent rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RussellAnalysisLadder}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.RussellAnalysisLadderUp.RussellAnalysisLadderTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Not claimed: This chapter does not close machine bridge checking, Russellian class ontology, imported reducibility, host quotient reasoning, proposition extensionality, proof irrelevance, or a completed host construction of analysis.

Upgrade path: The remaining independent upgrade is machine bridge checking of the already mature paper package for the public finite description, contextual-class, relation-first, stratification, and witnessed-descent packet.

Chapter 519

A Concrete Naming Certificate for $\mathsf{InscriptionEvent}^{\uparrow}$

The $\mathsf{InscriptionEvent}^{\uparrow}$ surface turns an inscription into finite NameCert data. It records the accepted public transition by which prior record state is carried into a named statement, a route witness, a check row, a downstream consumer, and a GAP ledger. The packet is not private intention, raw byte change, hidden agency, or an unconstrained host event.

The inscription packet consumes the kernel-acceptance trace of chapter 557, the TasteGate admission surface of chapter 545, and the audit membrane of chapter 524 before its route, check, and GAP ledger rows become public certificate data.

The public GAP row is consumed through chapter 520, while the accepted record route remains the $\mathsf{InscriptionEvent}^{\uparrow}$ carrier. The event packet owns the prior record, named statement, route witness, accepted check, and consumer rows; the ledger sibling owns the residue and non-escape accounting attached to that public gap row.

Definition 519.1 (InscriptionEvent carrier)**.** *An* $\mathsf{InscriptionEvent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, M, R, K, U, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the prior record-state row, naming the finite source history before the inscription is accepted;*
2. *M is the named statement or record row, naming the public inscription target and its interface;*
3. *R is the proof route or route-witness row, recording how the prior record state reaches the named statement;*
4. *K is the accepted check row, recording the finite checker result that admits the route as public record data;*
5. *U is the intended-consumer row, naming the downstream reader, citation, theorem site, or certificate surface that may consume the inscription;*
6. *G is the public* GAP *ledger row, recording the open commitment socket and the residue not discharged by generator replay alone;*
7. *H records componentwise* hsame *transports for the source, name, route, check, consumer,* GAP *ledger, package, and local naming rows;*

8. *C records the displayed* Cont *routes by which consumers read S, M, R, K, U, G without leaving the finite public packet;*
9. *P is the* Pkg *provenance row over* BHist, hsame, Cont, *public check data,* GAP *ledger data, and local certificate dependencies;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionEvent}^{\uparrow}}$ *row.*

The classifier for two accepted carriers compares only the public source row, named statement row, route-witness row, accepted check row, intended-consumer row, GAP *ledger row, and their displayed* hsame, Cont, Pkg, *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.InscriptionEventUp.InscriptionEventTasteGate single carrier alignment`

Theorem 519.2 (InscriptionEvent NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscriptionEvent}^{\uparrow}}$ *surface for the carrier of Definition 519.1 consists of five finite obligations:*

1. *carrier exposure: every accepted inscription event exposes exactly the packet $I = (S, M, R, K, U, G, H, C, P, N)$;*
2. *classifier locality: public comparison is restricted to the source, name, route, check, consumer, ledger, transport, continuation, package, and local naming rows;*
3. *route acceptance: the route-witness row R reaches the named statement row M from the prior record-state row S only through the accepted check row K;*
4. *consumer accountability: every downstream use of the inscription first reads the intended-consumer row U and the displayed* Cont *routes in C;*
5. *ledger exposure: the* GAP *row G is public certificate data and cannot be replaced by private intention, unrecorded agency, raw byte difference, or host-level completion.*

Proof. Carrier exposure is projection from the finite tuple in Definition 519.1. The source state, named statement, route witness, accepted check, consumer, ledger, transport, continuation, package, and local naming rows are all displayed coordinates.

Classifier locality follows from the last sentence of the carrier definition. The comparison has no coordinate beyond the public rows and their recorded hsame and Cont routes, so it cannot inspect private intention or unrecorded state.

Route acceptance and consumer accountability are read from R, K, U, and C. The route witness reaches the name only after the checker row accepts it, and every downstream consumer is one of the rows named by the consumer surface. The GAP ledger is therefore a public row in the same packet rather than an external explanation. □

Theorem 519.3 (InscriptionEvent carrier stability)**.** *For every accepted* $\mathsf{InscriptionEvent}^{\uparrow}$ *carrier, the public surface is stable under the displayed rows $S, M, R, K, U, G, H, C, P, N$. The* GAP *ledger is exactly the row G attached to the source record state, named statement, route witness, accepted check, and intended consumer, and every public comparison is transported only through the displayed* hsame, Cont, Pkg, *and* NameCert *provenance.*

Proof. Inspect the carrier coordinates. The ledger available to a consumer is G, and G appears beside $S, M, R, K, U, H, C, P, N$ in the same finite packet.

The NameCert obligations force the route witness through the accepted check row before the inscription is consumed. Thus any residue recorded by G is downstream of the public route and checker rows, not a replacement for them.

No carrier coordinate names private psychology, an agency primitive, a raw byte-difference classifier, or a host completion object. Therefore ledger exactness is exactly the displayed public GAP row. □

Theorem 519.4 (InscriptionEvent non-escape boundary)**.** *If two accepted* InscriptionEvent$^{\uparrow}$ *packets agree on the displayed source record state, named statement, route witness, accepted check row, intended consumer,* GAP *ledger, and their* hsame*,* Cont*,* Pkg*, and* NameCert *provenance, then every public consumer route sees the same inscription-event surface. The agreement does not identify private motives, host process traces, unrecorded edits, or any object outside the finite packet.*

Proof. Use classifier locality from Definition 519.2. The listed rows are exactly the rows the classifier may compare, so agreement on them fixes the public inscription surface.

The displayed Cont routes then carry each intended consumer through the same source, name, route, check, and ledger rows. Package and local certificate provenance are part of the same comparison, so downstream reads receive the same public packet.

The exclusions are immediate from the carrier inventory. Motives, host process traces, unrecorded edits, and non-packet objects are not coordinates of I and cannot be recovered by public compatibility. □

Theorem 519.5 (InscriptionEvent downstream handoff)**.** *Every accepted* InscriptionEvent$^{\uparrow}$ *packet hands public record use to downstream consumers only through the rows* $S, M, R, K, U, G, H, C, P, N$*. In particular, a theorem site, citation row, certificate checker, or audit consumer may read the named statement, route witness, accepted check, intended-consumer row, and* GAP *ledger, but it receives no private intention, host agency predicate, unchecked route, or hidden record mutation.*

Proof. Start from the carrier of Definition 519.1. Its intended-consumer coordinate U names the public reader, and its continuation coordinate C records the permitted routes into the packet.

Apply the route-acceptance and ledger-exposure obligations from Definition 519.2. They force the downstream route through R, K, and G after the source and named-statement rows have been displayed.

Finally use Definition 519.4. Any public consumer comparing inscription events is limited to the same rows, so the handoff cannot export private intention, host agency, unchecked routes, or hidden mutation outside the finite certificate. □

Theorem 519.6 (InscriptionEvent route obligation ledger)**.** *For an accepted* InscriptionEvent$^{\uparrow}$ *carrier*

$$I = (S, M, R, K, U, G, H, C, P, N),$$

the route row R*, checker row* K*, and named statement row* M *form the accepted route obligation. A public consumer may cite the inscription route only as the displayed passage from prior record state* S *to named statement* M *through the accepted checker row* K*.*

Proof. Project the carrier of Definition 519.1. The rows M, R, K are displayed coordinates, with R recording the route witness and K recording the accepted checker result.

Apply the route-acceptance clause of Definition 519.2. It forces the route witness to reach the named statement only through the accepted check row, so the route obligation is exactly the ledger formed by R, K, M inside the packet. □

Theorem 519.7 (InscriptionEvent consumer obligation ledger)**.** *For an accepted* InscriptionEvent$^{\uparrow}$ *carrier* I*, the consumer row* U*, public* GAP *row* G*, and continuation table* C *form the consumer-ledger obligation. Every downstream use of the inscription must read the intended consumer, the*

open commitment residue, and the displayed continuation routes before treating the inscription as public record data.

Proof. The carrier definition displays U, G, and C as finite rows of the same packet. The row U names the downstream consumer, G records the public GAP ledger, and C records the routes by which consumers read the packet.

Use consumer accountability and ledger exposure from Definition 519.2. They identify downstream use with reads through U and C, and they keep G as public certificate data rather than as private intention or host-level completion. □

Theorem 519.8 (InscriptionEvent public package obligation)**.** *For an accepted* $\mathsf{InscriptionEvent}^{\uparrow}$ *carrier* I*, the rows* H, C, P, N *assemble the public package obligation over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The inscription event is therefore available to public consumers only as a finite package with displayed transports, continuation routes, provenance, and local naming data.*

Proof. Project H, C, P, N from the carrier. The row H records componentwise hsame transports, C records displayed Cont routes, P records Pkg provenance over BHist and the public certificate data, and N records the local NameCert row.

The classifier locality clause in Definition 519.2 restricts public comparison to these displayed support rows together with the source, statement, route, check, consumer, and ledger rows. Thus the public package obligation is precisely the finite assembly over BHist, hsame, Cont, Pkg, and NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionEvent}^{\uparrow}}$**.**

Constructive story: An $\mathsf{InscriptionEvent}^{\uparrow}$ packet is generated from prior record state, a named statement or record, a route witness, an accepted check row, an intended consumer, a public GAP ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InscriptionEvent}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 519.1, Definition 519.2, Definition 519.3, Definition 519.4, Definition 519.5, Definition 519.6, Definition 519.7, and Definition 519.8 bind inscription events as finite public NameCert packets with the downstream consumer package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InscriptionEvent}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InscriptionEventUp.InscriptionEventTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge checking, hidden agency, private intention, raw byte-change identity, host process trace semantics, or completed record ontology.

Upgrade path: The paper-side theory closure is complete for the displayed finite inscription-event package; machine bridge checking remains a separate verification-axis obligation.

Chapter 520

A Concrete Naming Certificate for $\mathsf{InscriptionGapLedger}^{\uparrow}$

The $\mathsf{InscriptionGapLedger}^{\uparrow}$ surface isolates the ledger side of inscription. It records the residue left at the accepted route boundary as finite public NameCert data: the prior source scope, the named public record, the route witness, the accepted check row, the intended consumer, and the GAP row that accounts for what the generator replay alone does not discharge. The packet is narrower than the whole inscription event; it is the public residue ledger that downstream audit and supply-socket consumers may read. It is the residue-facing projection of the full inscription event carrier in chapter 519: the event chapter carries the broader inscription route, while this chapter keeps only source state, public record, route witness, accepted check, consumer, GAP residue, transport, route, provenance, and local name rows.

Definition 520.1 (InscriptionGapLedger carrier)**.** *An* $\mathsf{InscriptionGapLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, R, K, U, Q, H, C, P, N)$$

with the following visible rows:

1. *S is the prior source-state row at which the inscription route begins;*
2. *M is the named public record, theorem site, citation row, or interface reached by the route;*
3. *R is the displayed route witness from S to M;*
4. *K is the accepted check row that admits R as public record data;*
5. *U is the intended-consumer row naming the downstream reader of the ledger;*
6. *Q is the GAP residue row, recording the open commitment socket and the residue not discharged by generator replay alone;*
7. *H records componentwise* hsame *transport for S, M, R, K, U, Q together with package and naming rows;*
8. *C records the displayed* Cont *routes by which consumers read S, M, R, K, U, Q without leaving the finite packet;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, *accepted check data, GAP residue data, and local certificate dependencies;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionGapLedger}^{\uparrow}}$ *row.*

The carrier contains no private intention, hidden agency predicate, raw byte-change identity, unchecked route, host process trace, or complete inscription-event ontology.

Theorem 520.2 (InscriptionGapLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscriptionGapLedger}^{\uparrow}}$ *surface for the carrier of Definition 520.1 consists of five finite obligations: carrier exposure for* $G = (S, M, R, K, U, Q, H, C, P, N)$*; source-scope locality for the prior source state* S*; named route compression through* M, R, K*; consumer accountability through* U *and* C*; and GAP residue exactness through* Q, H, P, N*.*

Lean4 ✓ `BEDC.Derived.InscriptionGapLedgerUp.InscriptionGapLedgerTasteGate single carrier alignment`

Proof. Carrier exposure is projection from the displayed packet. Source-scope locality reads only S and the continuation rows in C, so the ledger begins at the public record state and not at an unstated host process.

Named route compression is the row chain S, R, K, M. The route witness reaches the named public record only through the accepted check row, and the package row records that this is the route admitted into the certificate.

Consumer accountability and residue exactness use U, Q, H, C, P, N. A downstream consumer must be one of the displayed readers and obtains the GAP residue through the same transport, continuation, package, and local naming rows. No additional coordinate can supply private intention, hidden agency, raw byte identity, or a complete inscription event. □

Theorem 520.3 (InscriptionGapLedger source boundary)**.** *Every accepted* $\mathsf{InscriptionGapLedger}^{\uparrow}$ *carrier reads source information only from the prior source-state row* S*, the route witness* R*, and the accepted check row* K*. The GAP residue row* Q *is attached to that boundary and cannot be used as a second source state, an unchecked route, or a replacement for the named public record* M*.*

Lean4 ✓ `BEDC.Derived.InscriptionGapLedgerUp.InscriptionGapLedger source boundary`

Proof. Project the carrier rows $S, M, R, K, U, Q, H, C, P, N$. The only source-facing rows are S, R, K, with M as the named endpoint reached after acceptance.

The GAP residue Q is recorded beside those rows and transported by H through routes in C. It therefore accounts for the displayed residue at the accepted boundary rather than adding another source, another route, or another public name.

Since the carrier has no further source coordinate, every public read of the ledger remains confined to the displayed source boundary. □

Theorem 520.4 (InscriptionGapLedger consumer exhaustion)**.** *Every downstream consumer of an accepted* $\mathsf{InscriptionGapLedger}^{\uparrow}$ *carrier is exhausted by the rows* $S, M, R, K, U, Q, H, C, P, N$*. A theorem site, citation row, audit surface, or supply-socket consumer may read the named route, accepted check, intended-consumer row, and GAP residue, but it receives no hidden mutation, private agency predicate, unchecked route, or host trace.*

Lean4 ✓ `BEDC.Derived.InscriptionGapLedgerUp.InscriptionGapLedger consumer exhaustion`

Proof. Start with a public consumer route and project U and C. These rows name the consumer and the displayed continuations into S, M, R, K, Q.

The package and local naming rows P and N attach the same route, check, consumer, and residue to one finite certificate. Componentwise transport through H preserves that attachment under accepted comparisons.

Thus the consumer sees exactly the displayed packet. Hidden mutation, private agency, unchecked routing, and host traces are absent from the carrier and cannot be exported by the ledger. □

Theorem 520.5 (InscriptionGapLedger non-escape)**.** *If two accepted* InscriptionGapLedger$^{\uparrow}$ *carriers agree on the displayed source state, named public record, route witness, accepted check, intended consumer, GAP residue,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *rows, then every public ledger consumer sees the same GAP-ledger surface. This agreement does not identify complete inscription events, private motives, host process traces, or objects outside the finite residue packet.*

Proof. Agreement on the listed rows is agreement on every coordinate of the carrier. The classifier has no additional public input beyond those rows and their displayed transport and continuation evidence.

Every consumer route factors through U and C, then reads the route, check, name, and residue rows transported by H and accounted for by P and N. Therefore the visible ledger surface is fixed.

The excluded objects are not coordinates of G. Complete inscription events, motives, host traces, and outside objects cannot be recovered from equality of the finite residue packet. □

Closure status for NameCert$_{\text{InscriptionGapLedger}^{\uparrow}}$**.**

Constructive story: An InscriptionGapLedger$^{\uparrow}$ packet is generated from prior source state, a named public record, a route witness, an accepted check row, an intended consumer, a GAP residue row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.InscriptionGapLedgerUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{InscriptionGapLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 520.1, Definition 520.2, Definition 520.3, Definition 520.4, and Definition 520.5 bind the GAP-ledger side of inscription as a finite NameCert residue packet with a checked TasteGate obligation surface.

Formal verification: FormalStatus(NameCert$_{\text{InscriptionGapLedger}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InscriptionGapLedgerUp.InscriptionGapLedgerTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close bridge checking, complete inscription-event ontology, private intention, hidden agency, raw byte-change identity, unchecked route semantics, or host process traces.

Upgrade path: The paper-side mature treatment is closed over the same finite residue packet, while bridge checking remains a separate Lean-side target.

Chapter 521

A Concrete Naming Certificate for $\mathsf{ContourIntegralOperation}^{\uparrow}$

The $\mathsf{ContourIntegralOperation}^{\uparrow}$ interface turns the contour-integration operation of chapter 9340 into a finite BEDC NameCert packet. It displays the piecewise-linear contour, holomorphic input, Riemann-sum ledger, convergence modulus, output complex-history row, componentwise transport, continuation routes, package provenance, and local naming row that downstream integral, residue, and RH scaffold consumers may inspect without importing a host analytic integral operator.

Definition 521.1 (Contour-integral operation carrier)**.** *A* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, F, S, M, I, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{PLContour}$ *row, including the finite list of* $\mathsf{Complex}^{\uparrow}$ *endpoints and the boundary-read data for its piecewise-linear segments;*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row on a neighbourhood containing the image of G;*
3. *S is the Riemann-sum ledger, recording subdivision counts, sample rows, segment contributions, and the finite sum row used by Definition 9340.5;*
4. *M is the explicit modulus row witnessing convergence of the Riemann sums toward the output;*
5. *I is the output* $\mathsf{Complex}^{\uparrow}$ *history row satisfying the contour-integral predicate of Definition 9340.7;*
6. *H stores componentwise* hsame *transport for G, F, S, M, I;*
7. *C stores the displayed* Cont *routes by which a consumer reads the contour, holomorphic input, Riemann-sum ledger, modulus, output, package provenance, and local name;*
8. *P is the* Pkg *provenance over* $\mathsf{PLContour}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, *Riemann sums, the modulus row,* hsame, Cont, *and the local naming surface;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourIntegralOperation}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient Lebesgue integral, a host path object, a quotient-of-paths class, a global branch choice, or an unledgered limiting operation.

Theorem 521.2 (Contour-integral operation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContourIntegralOperation}^{\uparrow}}$ *surface for Definition 521.1 consists of five finite obligations:*

1. *carrier exposure for* $T = (G, F, S, M, I, H, C, P, N)$*;*
2. *classifier stability for the* PLContour *row,* Holomorphic$^{\uparrow}$ *input row, Riemann-sum ledger, modulus row, and output* Complex$^{\uparrow}$ *history under the componentwise* hsame *row* H*;*
3. *operation-law ledger exactness for linearity, concatenation, reversal, and piecewise-linear reparameterization as recorded in the finite ledger* S *and routes* C*;*
4. *uniqueness of the output row* I *up to* hsame *using the modulus row* M *and the contour-integral uniqueness boundary;*
5. *non-escape from ambient integration operators, host path equality, quotient-of-paths classes, unrecorded sample choices, and global branch selectors.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationNameCertObligations`

Proof. Carrier exposure is projection from the finite packet of Definition 521.1. The contour, holomorphic input, Riemann-sum ledger, modulus, output, transport, routes, provenance, and local naming row are all displayed before a consumer can read the operation.

Classifier stability follows by transporting each displayed coordinate through H. The operation laws are not imported as host analytic facts; they are read through the ledger rows named by Definition 9340.20 and the routes in C.

Uniqueness is the boundary supplied by Definition 9340.9: any two accepted outputs for the same transported contour and holomorphic input are hsame-equal. Since no remaining coordinate stores an ambient integral or a host path quotient, the non-escape clause is exactly the carrier inventory. □

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationCarrier namecert obligations`

Theorem 521.3 (Contour-integral PL-contour boundary)**.** *Every accepted* ContourIntegralOperation$^{\uparrow}$ *packet reads its path data through the displayed* PLContour *row* G*. Public consumers may inspect the finite endpoint list, segment continuation rows, endpoint* Complex$^{\uparrow}$ *classifiers, and componentwise* hsame *transports for those rows, but they may not replace* G *by an ambient smooth path, a host homotopy class, or an unbounded path object.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationCarrier pl contour boundary`

Proof. Project the carrier coordinate G. Its only path data are the finite endpoint list and the segment rows licensed by the piecewise-linear contour carrier.

The routes in C read G before any Riemann-sum or output row is inspected. Transport through H can rename the displayed endpoint and segment rows, but it does not introduce a smooth path carrier or a host homotopy class.

Thus every contour read remains on the finite PLContour boundary, and any request for data outside that boundary has no coordinate in the packet. □

Theorem 521.4 (Contour-integral Riemann-sum ledger)**.** *For an accepted* ContourIntegralOperation$^{\uparrow}$ *packet* $T = (G, F, S, M, I, H, C, P, N)$*, every approximation read of the integral factors through the finite Riemann-sum ledger* S *and the modulus row* M*. The ledger records the subdivision count, sample rows, segment contributions, finite sum value, and precision comparison used to relate the sum to* I*; no unrecorded limiting process or sample-selection oracle is exported.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationCarrier riemann sum route closure`

Proof. Begin with an approximation read. The displayed routes in C first expose the contour row G and holomorphic input row F, then enter the Riemann-sum ledger S.

Inside S, the subdivision, samples, segment contributions, and finite sum are explicit rows. The modulus row M is the only row that turns these finite sums into a convergence claim about the output I.

Consequently an audit of the approximation surface reads only S and M after the input rows. Since no separate coordinate stores a limiting oracle or hidden sample selector, those objects are not public exports of the packet. □

Theorem 521.5 (Contour-integral public export)**.** *The public surface of an accepted* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *carrier exports exactly the finite operation packet: a* $\mathsf{PLContour}$ *row, a* $\mathsf{Holomorphic}^{\uparrow}$ *input row, the Riemann-sum ledger, convergence modulus, output* $\mathsf{Complex}^{\uparrow}$ *history, componentwise* hsame *transports, finite* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The exported operation laws are ledgered laws for this packet, not host-level contour-integration principles.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationPublicExport`

Proof. The carrier definition lists every row available to public consumers. The NameCert obligations bind carrier exposure, classifier stability, operation-law ledger exactness, uniqueness, and non-escape to precisely those rows.

The PL-contour boundary prevents path data from escaping the finite endpoint and segment surface. The Riemann-sum ledger theorem forces approximation reads through S and M before they reach the output row.

Therefore the public export is exactly the finite BEDC operation packet over contour, holomorphic input, sums, modulus, output, transport, routes, provenance, and local naming. No host analytic integral principle is exported. □

Theorem 521.6 (Contour-integral modulus transport)**.** *For accepted* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *packets with componentwise* hsame *transport of the contour row G, holomorphic input row F, Riemann-sum ledger S, modulus row M, and output row I, the transported modulus row remains the only public witness relating finite Riemann sums to the output* $\mathsf{Complex}^{\uparrow}$ *row. The transport excludes hidden sample selectors, ambient path equality, global branch choice, and unledgered limiting operations.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationModulusTransport`

Proof. Apply Definition 521.2 to expose the packet rows and the classifier-stability obligation. The componentwise transport preserves G, F, S, M, I as displayed coordinates, so the transported packet has the same public inventory.

Apply Definition 521.4. Every approximation read factors through the Riemann-sum ledger S and the modulus row M before it reaches the output row I.

Apply Definition 521.5. The exported operation packet has no coordinate for a hidden sample selector, ambient path equality, global branch choice, or unledgered limiting operation.

Therefore the transported modulus row is still the unique public bridge from finite Riemann sums to the transported $\mathsf{Complex}^{\uparrow}$ output row. Any requested witness outside S and M has no row in the accepted packet. □

Theorem 521.7 (Contour-integral operation-law ledger closure)**.** *For an accepted* ContourIntegralOperation$^{\uparrow}$ *packet*

$$T = (G, F, S, M, I, H, C, P, N),$$

linearity, concatenation, reversal, and piecewise-linear reparameterization are consumer-visible only as ledgered operation rows inside S and through the displayed routes C. No consumer may replace those rows by a host analytic integration law, a path-quotient principle, or an unrecorded limiting operation.

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationCarrier operation law ledger closure`

Proof. Project the local NameCert obligations of Definition 521.2. The operation-law clause names linearity, concatenation, reversal, and piecewise-linear reparameterization as ledger facts for the finite packet.

Apply Definition 521.3. The contour side is restricted to the displayed PLContour row G, so the laws cannot be read through an ambient smooth path object or path quotient.

Apply Definition 521.4 and Definition 521.5. The consumer sees the operation through S, C, P, N and the displayed input-output rows only. Hence the operation laws are closed at the ledger surface. □

Theorem 521.8 (Contour-integral operation modulus uniqueness boundary)**.** *For an accepted* ContourIntegralOperation$^{\uparrow}$ *packet $T = (G, F, S, M, I, H, C, P, N)$, the modulus row M is the unique public bridge from the Riemann-sum ledger S to the output row I. Any transported packet with the same displayed G, F, S, I, H, C, P, N must read convergence toward I through the transported M row and not through a hidden sample selector, host path equality, global branch choice, or ambient integral operator.*

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationModulusUniquenessBoundary`

Proof. Apply Definition 521.4. Approximation reads reach I only after passing through the finite ledger S and the modulus row M.

Use Definition 521.6. Componentwise transport preserves M as the only public witness relating the ledger to the output row.

Finally apply Definition 521.2 and Definition 521.5. The carrier exposes no second convergence bridge, so uniqueness is exactly the finite inventory of the accepted packet. □

Theorem 521.9 (Contour-integral operation residue-consumer boundary)**.** *Every residue-facing consumer of an accepted* ContourIntegralOperation$^{\uparrow}$ *packet*

$$T = (G, F, S, M, I, H, C, P, N)$$

may inspect only the PLContour *row G, holomorphic input row F, Riemann-sum ledger S, modulus row M, output row I, componentwise transports H, displayed routes C, provenance P, and local name N. It receives no residue theorem, pole-order classifier, host contour integral, global branch selector, quotient path class, or ambient analytic continuation principle from this operation packet.*

Proof. First apply Definition 521.2. The local NameCert surface exposes exactly the displayed carrier rows and the non-escape clause for the operation packet.

Use Definition 521.3 to keep path data inside G, and use Definition 521.4 to keep approximation data inside S and M.

Apply Definition 521.5. The public operation surface is precisely $G, F, S, M, I, H, C, P, N$. A residue-facing consumer can route through that surface, but the carrier has no coordinate for residue calculus or host analytic integration principles. □

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationCarrier residue consumer boundary`

Theorem 521.10 (Contour-integral operation classifier stability)**.** *Let $T = (G, F, S, M, I, H, C, P, N)$ and $T' = (G', F', S', M', I', H', C', P', N')$ be accepted* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *packets. If the contour row, holomorphic input, Riemann-sum ledger, modulus row, output row, route rows, package provenance, and local* $\mathsf{NameCert}$ *row are related componentwise by* hsame*, then T' is again classified as a* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *packet over the transported rows. The transport adds no host path equality, hidden sample choice, global branch choice, or ambient integral data.*

Proof. Apply Definition 521.2. Its classifier-stability clause requires exactly componentwise transport of the contour, holomorphic input, Riemann-sum ledger, modulus, output, routes, provenance, and local naming rows.

Use Definition 521.6 for the modulus side. The transported modulus remains the only public witness from finite Riemann sums to the transported output row, so hidden sample choice and unledgered limiting data cannot enter during transport.

Finally apply Definition 521.5. The exported packet after transport is still the finite operation surface $G', F', S', M', I', H', C', P', N'$, and the public surface contains no coordinate for host path equality, global branch choice, or an ambient integral operator. □

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationClassifierStability`

Theorem 521.11 (Contour-integral operation bridge envelope)**.** *The bridge-facing read of an accepted* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *packet is exactly the* $\mathsf{PLContour}$ *row, the* $\mathsf{Holomorphic}^{\uparrow}$ *input row, the Riemann-sum ledger, the modulus row, the* $\mathsf{Complex}^{\uparrow}$ *output row, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContourIntegralOperation}^{\uparrow}}$ *row. It exports no host analytic integral, path quotient, residue theorem, global branch selector, or analytic-continuation principle.*

Proof. Begin with the NameCert obligations of Definition 521.2. They expose the carrier rows and bind classifier stability, operation-law ledger exactness, uniqueness, and non-escape to the finite packet.

Use the PL-contour boundary of Definition 521.3 and the Riemann-sum ledger of Definition 521.4. These keep the path side on the displayed $\mathsf{PLContour}$ row and the approximation side on the finite ledger plus modulus row.

Finally apply the public export and classifier-stability theorems, Definition 521.5 and Definition 521.10. The bridge-facing read is therefore the same finite operation packet under transport, with no coordinate from which host analytic integration, path quotients, residue calculus, global branch selection, or analytic continuation could be exported. □

Lean4 ✓ `BEDC.Derived.ContourIntegralOperationUp.ContourIntegralOperationBridgeEnvelope`

Closure status for $\mathsf{NameCert}_{\mathsf{ContourIntegralOperation}^{\uparrow}}$.

Constructive story: A $\mathsf{ContourIntegralOperation}^{\uparrow}$ packet is generated from a displayed $\mathsf{PLContour}$ row, a $\mathsf{Holomorphic}^{\uparrow}$ input row, a finite Riemann-sum ledger, an explicit modulus row, an output $\mathsf{Complex}^{\uparrow}$ history, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourIntegralOperation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 521.1, Definition 521.2, Definition 521.5, Definition 521.6, Definition 521.7, and Definition 521.9 bind exactly the displayed PLContour row, Holomorphic$^{\uparrow}$ input row, Riemann-sum ledger, modulus row, Complex$^{\uparrow}$ output row, hsame, Cont, Pkg, and NameCert rows already named in the chapter.

Formal verification: FormalStatus(NameCert$_{\mathsf{ContourIntegralOperation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Cauchy's theorem, residue calculus, RH zero-location consequences, global branch selection, Lean verification, or bridge checking.

Upgrade path: Formal progress requires Lean verification and bridge checking for the same PL-contour, holomorphic-input, Riemann-sum, modulus, output, hsame, Cont, Pkg, and NameCert rows.

Chapter 522

A Concrete Naming Certificate for $\mathsf{RecursorInducedNameCert}^{\uparrow}$

The $\mathsf{RecursorInducedNameCert}^{\uparrow}$ surface turns a legal host-CIC recursor use into finite BEDC certificate data. It records the accepted inductive signature, the motive row, the branch package, the generated proof or definition output, the audit replay row, package provenance, and the local naming row. The packet is not an unrestricted induction principle and does not certify terms whose dependency trace reaches a forbidden host principle.

It consumes the recursor-generator family through chapter 534 and chapter 525. Those sibling surfaces supply the accepted-signature rows, branch-exhaustion rows, audit-provenance rows, and output-naming rows that this chapter repackages as an induced $\mathsf{NameCert}$.

Definition 522.1 (Recursor-induced NameCert carrier)**.** *A* $\mathsf{RecursorInducedNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, M, B, O, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted inductive-signature row, naming the finite constructor family and the kernel admission event that makes the associated recursor available;*
2. *M is the motive row, recording the target family or target judgment selected by the recursor call;*
3. *B is the branch-package row, with one checked branch for each constructor exposed by S and with recursive hypotheses only at the structurally smaller sites allowed by the recursor;*
4. *O is the generated-output row, naming the proof object, definition body, or certificate row produced by applying the recursor to M and B;*
5. *A is the audit-replay row, recording kernel replay of the recursor call together with the source-axiom and forbidden-dependency checks relevant to O;*
6. *H records componentwise* hsame *transports for S, M, B, O, A, the continuation row, the package row, and the local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the signature, motive, branch package, output, audit replay, provenance, and local certificate;*
8. *P is the* Pkg *provenance over* BHist*, accepted inductive signatures, recursor motives, branch packages, generated outputs, audit rows,* hsame*,* Cont*, and local naming;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}$ *row.*

The carrier exposes only these finite rows. In particular, it contains no free choice row, no quotient-soundness row, no propositional-extensionality row, no non-structural recursive call, and no branch for a constructor absent from the accepted signature.

Lean4 ✓ `BEDC.Derived.RecursorInducedNameCertUp.RecursorInducedNameCertTasteGate single carrier alignment`

Definition 522.2 (Recursor generator classifier)**.** *The* $\mathsf{RecursorInducedNameCert}^{\uparrow}$ *classifier compares two carriers only through their displayed signature, motive, branch, output, audit replay, transport, continuation, package, and local naming rows. A row is accepted by the classifier exactly when it is reachable from an accepted inductive signature through a kernel-accepted motive, a complete constructor-indexed branch package, a generated output row, and an audit replay row that records no forbidden-principle dependency for the generated output.*

Theorem 522.3 (Recursor generator NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}$ *surface for the carrier of Definition 522.1 consists of five BEDC-native obligations:*

1. *carrier exposure for the finite packet* $R = (S, M, B, O, A, H, C, P, N)$*;*
2. *signature admission stating that* S *names an accepted inductive constructor family and no constructor outside that family;*
3. *branch completeness stating that* B *supplies exactly the constructor-indexed branches required by* S *for the motive row* M*;*
4. *output exactness stating that* O *is the generated proof, definition, or certificate row obtained from the recursor call determined by* S, M, B*;*
5. *audit replay stating that* A *replays the generated output through kernel checking, source-axiom refusal, and forbidden-dependency refusal before* N *exports the local name.*

Proof. Carrier exposure is projection from Definition 522.1. The rows $S, M, B, O, A, H, C, P, N$ are the only admitted coordinates of the packet.

Signature admission and branch completeness are read from S and B. The accepted signature fixes the constructor list, and the branch package has one branch for each listed constructor with recursive hypotheses only where the recursor permits them.

Output exactness and audit replay are then read from O and A. The output row is named only after the recursor call has been replayed against its motive and branch rows, and the local certificate row exports no name until the audit row records the absence of the forbidden-principle channels. □

Theorem 522.4 (Recursor generator audit boundary)**.** *For an accepted* $\mathsf{RecursorInducedNameCert}^{\uparrow}$ *carrier, the audit row* A *is the exact public boundary for trusting the generated output row* O*. A downstream consumer may read* O *as a recursor-induced certificate only through the finite tuple consisting of the signature row, motive row, branch package, generated output row, audit replay row, componentwise transport, continuation route, package provenance, and local* $\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}$ *row.*

This boundary does not assert a theorem of host-kernel completeness, semantic completeness of all possible recursor calls, or trust in any output whose reported dependency trace includes Classical.choice*,* Quot.sound*, or* propext*.*

Lean4 ✓ `BEDC.Derived.RecursorInducedNameCertUp.RecursorInducedNameCertAuditBoundary replay route`

Proof. Project the audit row from the carrier. It records the replay surface attached to the generated output row, so a consumer receives the output only together with the displayed signature, motive, branch package, and audit report.

Transport and continuation access are confined by H and C. These rows can move and read the displayed packet, but they do not add host-kernel completeness or an unlisted dependency theorem.

The forbidden-principle exclusions follow from the final clause of Definition 522.2. A dependency trace reporting one of the listed principles lacks an accepted classifier row, so it is recorded outside the recursor-induced certificate surface rather than exported by N. □

Theorem 522.5 (Recursor generator non-escape boundary)**.** *If two accepted* $\mathsf{RecursorInducedNameCert}^{\uparrow}$ *carriers agree on the displayed signature, motive, branch package, generated output, audit replay,* hsame *transports,* Cont *routes,* Pkg *provenance, and local naming rows, then every public consumer route sees the same recursor-induced NameCert surface. The agreement does not identify terms generated by a different inductive signature, an incomplete branch package, a non-structural recursive call, an unreported dependency chain, or a forbidden-principle proof object.*

Proof. Use the classifier of Definition 522.2. It compares only the rows displayed by the carrier, so agreement on those rows fixes the public certificate surface.

The continuation routes then replay the same reads for every downstream consumer. Since package provenance and the local naming row are part of the agreement, the consumer receives the same recursor-induced name and the same audit-facing boundary.

The exclusions are immediate from the carrier inventory and classifier condition. Different signatures, missing branches, non-structural calls, unreported dependencies, and forbidden-principle proof objects have no coordinate that can be projected from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{RecursorInducedNameCert}^{\uparrow}$ packet is generated from an accepted inductive signature, a motive row, a complete branch package, a generated output row, an audit replay row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 522.1, Definition 522.2, Definition 522.3, Definition 522.4, and Definition 522.5 bind recursor-produced proof and definition outputs as finite NameCert packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursorInducedNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge checking, host-kernel completeness, all possible recursor calls, non-structural recursion, or outputs depending on forbidden host principles.

Upgrade path: Obligation closure requires Lean encoding of the displayed carrier and checked theorems for branch completeness, generated-output exactness, audit replay, and forbidden-principle non-escape.

Chapter 523

A Concrete Naming Certificate for $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$

The $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$ surface turns the seven socket-kind table of chapter 9394 into a finite NameCert packet. It records the socket kind, requested supply shape, audit gate, site row, and local routing data for a reported Gap boundary. The packet classifies the boundary; it does not discharge the requested supply, add a new substrate axiom, or turn the apophatic endpoint into an internal BEDC constructor.

The taxonomy packet is consumed through the concrete ledger route of chapter 528, which records the finite socket kind, audit gate, provenance, and continuation rows before any boundary classifier is read.

Definition 523.1 (TranscendentalSupplyTaxonomy carrier)**.** *A* $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, S, G, A, L, H, C, P, N)$$

with the following displayed rows:

1. *K is one of the seven socket-kind rows: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure;*
2. *S is the requested supply shape, stated as the exact missing item that the local construction would consume;*
3. *G is the visible* Gap *tag that keeps the request outside the closed substrate;*
4. *A is the audit-gate row: generator closure, witness visibility, extensional-collapse refusal, records-versus-prediction separation, locality discipline, aspect separation, or reflection stratification;*
5. *L is the concrete site row naming the paper or formal location where the request would be consumed;*
6. *H records componentwise* hsame *transport for K, S, G, A, L, package provenance, and the local naming row;*
7. *C records the displayed* Cont *routes that read the site, expose the* Gap *tag, classify the requested supply, and return the audit gate to downstream consumers;*

8. *P is the* Pkg *provenance over* BHist, Gap, *the seven socket-kind rows, requested supply shapes, audit gates, site rows,* hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a supplied witness, unrestricted induction principle, quotient equality, future empirical law, shared first-person history, agency explanation, or truth-total self-validator.

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyTasteGate single carrier alignment`

Theorem 523.2 (TranscendentalSupplyTaxonomy NameCert obligations)**.** *For the carrier of Definition 523.1, the local* $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}}$ *surface consists exactly of socket-kind exposure, requested-supply accountability, audit-gate routing, site-row locality, classifier transport, continuation locality, package provenance, and non-discharge of the requested supply. A consumer accepts the taxonomy row only by projecting*

$$(K, S, G, A, L, H, C, P, N)$$

and checking that K is one of the seven displayed socket kinds, that S names the missing shape consumed at L, and that A is the gate preventing S from being internalised as a substrate theorem.

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyNameCert obligations`

Proof. Carrier exposure is projection from Definition 523.1. The packet lists the socket kind, requested supply shape, Gap tag, audit gate, site row, transport ledger, continuation ledger, provenance, and local naming row.

Requested-supply accountability is the pair (S, L). The site row states where the missing item would be consumed, and the requested-shape row states what the missing item is.

Audit-gate routing is the pair (G, A). The Gap tag keeps the request outside the closed substrate, while the audit gate names the discipline that blocks silent internalisation of the requested shape.

Classifier transport is componentwise transport through H, and continuation locality is replay through C. These rows move the displayed taxonomy packet without replacing its socket kind or adding a discharged supply coordinate.

Package provenance and non-discharge follow from P and from the carrier inventory. Since the tuple has no row for the requested witness, induction principle, equality token, future law, shared locality cell, inscription source, or truth-total validator, the NameCert surface classifies the socket without closing it. □

Theorem 523.3 (TranscendentalSupplyTaxonomy boundary exhaustion)**.** *Every accepted* TranscendentalSupplyTa *packet exports exactly one finite taxonomy row*

$$K \to S \to G \to A \to L \to H \to C \to P \to N.$$

Any downstream consumer can read the socket kind, requested supply shape, audit gate, site row, transport data, continuation routes, provenance, and local name, but cannot extract from the packet the supply whose absence the Gap *tag records.*

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyBoundaryExhaustion`

Proof. Project the accepted carrier $T = (K, S, G, A, L, H, C, P, N)$. The only taxonomy-facing coordinates are the displayed finite rows named in the statement.

Use the NameCert obligations of Definition 523.2. They force the downstream read to pass through the requested-supply row, the Gap row, and the audit-gate row before the site can be treated as classified.

The non-discharge clause then exhausts the boundary. A consumer may learn which of the seven socket kinds has appeared and which gate governs it, but no coordinate in the packet supplies the missing witness, universal rule, identity token, empirical persistence, inter-history recognition, inscription source, or self-reference closure. □

Theorem 523.4 (TranscendentalSupplyTaxonomy socket-kind totality)**.** *For every accepted* TranscendentalSupplyT *packet*

$$T = (K, S, G, A, L, H, C, P, N),$$

the socket-kind row K is exhausted by exactly the seven displayed rows: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure. No accepted classifier read exposes an eighth socket kind, an ambient supply class, or a hidden discharge row.

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomySocketKindTotality`

Proof. Project the carrier of Definition 523.1. Its first coordinate K is declared to be one of the seven displayed socket-kind rows, and the remaining coordinates record the request, Gap, gate, site, transport, continuation, provenance, and local name.

Apply boundary exhaustion from Definition 523.3. A downstream consumer can read the socket kind and the gate that governs it, but cannot extract a supplied witness, induction rule, equality token, empirical persistence row, inter-history recognition object, inscription source, or self-reference validator.

The classifier is therefore total only over the seven named rows. Since the carrier has no coordinate for another kind or a discharge row, no accepted read can extend the taxonomy beyond that finite list. □

Theorem 523.5 (TranscendentalSupplyTaxonomy site locality)**.** *For every accepted* TranscendentalSupplyTaxonom *packet*

$$T = (K, S, G, A, L, H, C, P, N),$$

the site row L is the only locality at which the requested supply shape S is consumed. Transport through H and replay through C may expose the classified request at that same site, but they cannot move the requested supply to an unrecorded paper location, formal location, or substrate row.

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomySiteLocality`

Proof. Read the requested-supply accountability clause of Definition 523.2. It binds the requested shape and the place where it would be consumed as the pair (S, L).

Use the carrier inventory to inspect the remaining rows. The Gap tag G and audit gate A explain why S is outside the closed substrate, while H, C, P, N transport, replay, package, and name the same finite packet.

No second site coordinate is present. Hence any accepted consumer must treat L as the unique locality of the request, and any attempted relocation outside L leaves the displayed taxonomy packet. □

Theorem 523.6 (TranscendentalSupplyTaxonomy nondischarge ledger)**.** *For every accepted* TranscendentalSupplyT *packet*

$$T = (K, S, G, A, L, H, C, P, N),$$

no ledger row other than a displayed discharge of the requested supply can be converted into supplied content. In the present packet every non-discharge row remains classificatory: K classifies the socket, G marks the Gap*, A records the audit gate, L locates the request, and H, C, P, N transport, replay, package, and name the boundary.*

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyCarrier nondischarge ledger`

Proof. Start from Definition 523.3. It says a consumer may recover the socket kind, requested supply shape, audit gate, site row, transport data, continuation routes, provenance, and local name, but not the absent supply itself.

Now inspect the non-discharge rows. The classifier row, Gap row, gate row, site row, transport row, continuation row, provenance row, and naming row are all boundary data; none is a witness, induction principle, equality token, empirical law, shared locality cell, inscription source, or truth-total validator.

Therefore the ledger cannot convert a non-discharge row into supplied content. It records why the request remains classified at the boundary instead of being internalised as a substrate theorem. □

Theorem 523.7 (TranscendentalSupplyTaxonomy closure surface)**.** *The* TranscendentalSupplyTaxonomy$^{\uparrow}$ *packet supplies a seed closure surface for the seven socket-kind taxonomy: finite carrier exposure, local NameCert obligations, and boundary exhaustion. Its public meaning is classificatory only. It preserves the single* T *supply position by recording aspect-position rows without adding a substrate-visible source object for any type.*

Proof. The carrier of Definition 523.1 provides the finite packet. The NameCert obligations of Definition 523.2 state exactly which classifier, routing, provenance, and non-discharge rows are visible.

Boundary exhaustion in Definition 523.3 prevents a consumer from reading the requested supply itself. Thus the taxonomy row is not a hidden constructor for T and not a discharge of any Gap.

The closure surface is therefore seed-level: it binds the finite packet and the local classification obligations while leaving Lean encoding, checked transport, public bridge data, and any stronger paper package outside the present chapter. □

Theorem 523.8 (TranscendentalSupplyTaxonomy bridge precondition)**.** *Any standard-bridge-facing read of* TranscendentalSupplyTaxonomy$^{\uparrow}$ *must first display the seven socket-kind row, requested-supply row,* Gap *tag, audit gate, site row, componentwise transport, continuation route, package provenance, and local* NameCert *row. Only after those rows are visible may the packet be treated as a bridge candidate, and the candidate remains a classified boundary rather than a discharged supply.*

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyCarrier bridge precondition`

Proof. Project the carrier of Definition 523.1. It exposes exactly the tuple $(K, S, G, A, L, H, C, P, N)$, where K is the seven-kind socket row, S is the requested-supply row, G is the Gap tag, A is the audit gate, and L is the site row.

Apply the NameCert obligations of Definition 523.2. They require classifier transport, continuation locality, package provenance, and local naming before a consumer accepts the taxonomy packet.

Boundary exhaustion and socket-kind totality, Definition 523.3 and Definition 523.4, then keep the read finite and classificatory. Hence the displayed packet is a bridge precondition, not a supplied witness or a completed bridge. □

Theorem 523.9 (TranscendentalSupplyTaxonomy socket ledger readback)**.** *Downstream consumers read* TranscendentalSupplyTaxonomy$^{\uparrow}$ *through the* SupplySocketLedger$^{\uparrow}$ *sibling route. The finite ledger exposes the socket kind, requested supply,* Gap *tag, audit gate, site row, transport, continuation, provenance, and local naming data, and no consumer may bypass that route to read a hidden supply coordinate.*

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomySocket ledger readback`

Proof. Begin with the ledger dependency stated at the chapter opening: the taxonomy packet is consumed through chapter 528. That sibling records socket kind, audit gate, provenance, and continuation rows before the boundary classifier is read.

Use site locality from Definition 523.5. The requested supply shape S is tied to the displayed site row L, so a consumer cannot move the request to another site outside the ledger.

Finally apply the nondischarge ledger, Definition 523.6. Since non-discharge rows remain classificatory, the ledger route cannot be bypassed to obtain a witness, induction principle, equality token, or truth-total validator. □

Theorem 523.10 (TranscendentalSupplyTaxonomy GAP nondischarge)**.** *The bridge-facing* TranscendentalSupplyT *packet preserves the requested-supply absence marked by G. It does not turn the boundary into a witness, an unrestricted induction principle, a quotient equality, a future empirical law, an inter-history recognition object, an inscription source, or a truth-total validator.*

Proof. Read boundary exhaustion from Definition 523.3. A consumer can recover the finite taxonomy rows, but cannot extract the supply whose absence the Gap tag records.

Now apply the nondischarge ledger of Definition 523.6. The rows K, G, A, L, H, C, P, N classify, locate, transport, replay, package, and name the request; none is supplied content.

The bridge precondition of Definition 523.8 therefore preserves absence while making the packet readable. It gives a bridge-facing obligation surface without converting the boundary into any of the forbidden discharge objects. □

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.TranscendentalSupplyTaxonomyGap nondischarge`

Closure status for NameCert$_{\text{TranscendentalSupplyTaxonomy}^{\uparrow}}$.

Constructive story: A TranscendentalSupplyTaxonomy$^{\uparrow}$ packet is generated from a seven-kind socket row, a requested supply shape, an explicit Gap tag, an audit gate, a site row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TranscendentalSupplyTaxonomy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 523.1, Definition 523.2, Definition 523.3, Definition 523.4, Definition 523.5, Definition 523.6, and Definition 523.7 bind the finite carrier, seven-kind classifier, requested-supply accountability, audit-gate routing, site-row locality, socket-kind exhaustion, and non-discharge boundary.

Formal verification: FormalStatus(NameCert$_{\text{TranscendentalSupplyTaxonomy}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyTaxonomyUp.taste gate`

Not claimed: This chapter does not close Lean verification, any Gap discharge, unrestricted induction, hidden witness selection, quotient equality, empirical persistence, inter-history recognition, inscription-source explanation, self-reference closure, or a public standard bridge.

Upgrade path: No further paper-side closure grade remains; Lean verification and bridge checking remain verification-axis work.

Chapter 524

A Concrete Naming Certificate for $\mathsf{AuditMembrane}^{\uparrow}$

The $\mathsf{AuditMembrane}^{\uparrow}$ packet turns the audit-gate boundary into a finite local certificate. It records the gate surface, the content refused at that surface, the visible result of the refusal, the route by which paper-Lean drift is resolved, the transport and replay rows, the package provenance, and the local naming certificate available to downstream discipline chapters.

It consumes the audit-gate refusal surface of chapter 526: the membrane carrier repackages that boundary as the downstream gate, boundary-content, refusal-result, drift-route, transport, replay, provenance, and local NameCert rows used below.

Definition 524.1 (AuditMembrane carrier)**.** *An* $\mathsf{AuditMembrane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, B, R, D, H, C, P, N)$$

with the following displayed rows:

1. *G is the gate-surface row, recording the source-token scan, dependency ancestry scan, marker-resolution scan, and origin-flow scan as one finite audit face;*
2. *B is the boundary-content row, recording the proposed crossings through source axioms, forbidden host dependencies, unresolved Lean names, and untracked origin/import flow;*
3. *R is the refusal-result row, recording for each boundary item whether the membrane admits it as displayed certificate content or refuses it as non-substrate supply;*
4. *D is the drift-resolution route row, recording the named paper marker, the candidate Lean declaration, and the finite resolve-or-refuse result;*
5. *H records componentwise* hsame *transports for G, B, R, D, the continuation row, the package row, and the local naming row;*
6. *C records the displayed* Cont *routes by which consumers inspect the gate surface, boundary content, refusal result, drift route, provenance, and local certificate;*
7. *P is the* Pkg *provenance over* BHist, Hist, GAP, hsame, Cont, Pkg, $\mathsf{NameCert}$, *audit rows, marker names, and origin ledgers;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for a hidden global audit theorem, no permission to replace an unresolved name by a nearby theorem, no semantic proof that every import is conservative, and no route by which refused boundary content becomes substrate-internal theorem content.

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneTasteGate single carrier alignment`

Theorem 524.2 (AuditMembrane NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}$ *surface for the carrier of Definition 524.1 consists of carrier admission for*

$$M = (G, B, R, D, H, C, P, N),$$

gate-surface exactness, boundary-content display, refusal-result determinacy, paper-Lean drift resolution, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, and non-escape for hidden global correctness, fuzzy target resolution, source-axiom admission, forbidden host-dependency admission, and untracked origin flow.*

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 524.1. The packet admits exactly the rows G, B, R, D, H, C, P, N.

The audit obligations are then read row by row. The gate face is supplied by G, the boundary content by B, the refusal result by R, and the paper-Lean drift route by D.

Transport and consumer access are confined by H and C. These rows replay the displayed audit packet under componentwise hsame movement and expose only the recorded routes.

Finally P and N record provenance and the local naming certificate. Since the excluded objects have no coordinate in the packet, the NameCert surface cannot export them. □

Theorem 524.3 (AuditMembrane boundary refusal)**.** *For an accepted* $\mathsf{AuditMembrane}^{\uparrow}$ *packet, the refusal row R is the complete public result of applying the gate surface G to the boundary-content row B. A consumer may read admitted certificate content only through G, B, R, H, C, P, N, and any source axiom, forbidden host dependency, unresolved target name, or untracked origin edge marked as refused in R has no exported route into substrate theorem content.*

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneCarrier refusal replay closure`

Proof. Project the carrier and read the gate surface G together with the boundary-content row B. These rows list the finite audit face and the finite items presented to that face.

The refusal result is the row R. Because R is part of the same finite packet, every admitted or refused item remains tied to the displayed gate surface that classified it.

The replay rows H, C, P, N preserve and expose the classification. They transport, route, record provenance, and name the packet, but they do not add a new coordinate by which a refused item can be read as theorem content. □

Theorem 524.4 (AuditMembrane drift resolution)**.** *For an accepted* $\mathsf{AuditMembrane}^{\uparrow}$ *packet, the drift-resolution row D is the only route by which a paper marker becomes a local naming certificate. The route must display the marker text, the proposed Lean target name, the finite resolution result, the componentwise* hsame *transport row, the* Cont *replay row, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}$ *row.*

If D records refusal, the packet exports no replacement by similar spelling, neighboring namespace, prose interpretation, or consumer preference.

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneCarrier drift resolution`

Proof. Project D from the carrier. It contains the paper marker, the candidate Lean declaration, and the finite resolve-or-refuse result.

Move the row only through the displayed transport and replay data. The rows H and C preserve the same marker and the same resolution result, while P and N record provenance and the local name.

No carrier coordinate stores fuzzy matching, namespace guessing, prose reinterpretation, or consumer-selected substitution. A refused drift row therefore remains refused at every exported read. □

Theorem 524.5 (AuditMembrane visible audit induction)**.** *Let*

$$M = (G, B, R, D, H, C, P, N)$$

be an accepted $\mathsf{AuditMembrane}^{\uparrow}$ *packet, and let L be a finite list of displayed* BHist *audit-event rows. Suppose each row of L is admitted through the same audit-gate-boundary content route as B, carries the displayed* hsame, Cont, *and* Pkg *provenance required by H, C, P, and presents no object outside the boundary-content format of chapter 526. Then extending B by the rows of L preserves the membrane verdict recorded by R: admitted rows remain admitted as displayed certificate content, refused rows remain refused boundary content, and no row of L exports global audit completeness, arbitrary import conservativity, or host-kernel completeness.*

Proof. Induct on the finite list L of displayed BHist rows.

For the empty list, the extended boundary row is just the original B. The verdict is therefore the existing membrane verdict R, read through Definition 524.3.

For the successor step, write the list as one displayed audit event followed by a tail. The head event is first consumed as an audit-gate-boundary row: it is presented to the same gate surface G, with the displayed content format required by B and with the hsame, Cont, and Pkg rows required by H, C, P.

The boundary-refusal theorem applies to that one displayed row, so its admitted or refused verdict is recorded by R and not by any hidden audit theorem. The induction hypothesis then applies to the remaining finite tail, preserving the verdict for every later displayed row.

Thus every finite extension by displayed audit events remains inside the membrane packet. The argument proves only this finite induction route over visible BHist rows; it does not assert global audit completeness, conservativity for arbitrary imports, or host-kernel completeness. □

Theorem 524.6 (AuditMembrane consumer boundary)**.** *Every downstream consumer of an accepted* $\mathsf{AuditMembrane}^{\uparrow}$ *packet factors through the finite rows*

$$G, B, R, D, H, C, P, N.$$

Such a consumer may inspect the gate surface, boundary content, refusal result, paper-Lean resolution route, transport rows, replay rows, provenance, and local naming certificate, but it cannot extract a global audit-completeness theorem, a semantic conservativity theorem for arbitrary imports, a host-kernel completeness proof, or a theorem object for refused boundary content.

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneCarrier consumer boundary`

Proof. Let a downstream consumer route be given. By Definition 524.1, the source carrier is the finite tuple G, B, R, D, H, C, P, N.

The consumer first reads the gate and boundary rows, then the refusal and drift-resolution rows. The rows H and C supply all permitted transport and replay, and P, N supply provenance and the local certificate.

The non-escape claim follows from the carrier inventory. None of the forbidden objects appears among the displayed rows, so no consumer route out of the packet can project it. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMembrane}^{\uparrow}$ packet is generated from a finite audit gate surface, a displayed boundary-content row, a refusal-result row, a paper-Lean drift-resolution route, componentwise hsame transports, displayed Cont replay routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 524.1, Definition 524.2, Definition 524.3, Definition 524.4, Definition 524.6, and bridge witness Definition 524.7 specify the finite audit membrane carrier, local naming obligations, refusal surface, marker-resolution route, transport replay, package provenance, consumer boundary, and public exported refusal bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMembrane}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.AuditMembraneUp.AuditMembraneCarrier\_consumer\_boundary`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close global audit completeness, arbitrary import conservativity, host-kernel completeness, or Lean-side formal verification for $\mathsf{AuditMembrane}^{\uparrow}$.

Upgrade path: No further formal bridge-axis upgrade remains for the displayed standard bridge.

Theorem 524.7 (AuditGateBoundary to AuditMembrane refusal bridge)**.** *Let*

$$A = (S, D_A, L, O, H_A, C_A, P_A, G_A, N_A)$$

be an accepted $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, and let*

$$M = (G_M, B, R, D_M, H_M, C_M, P_M, N_M)$$

be an accepted $\mathsf{AuditMembrane}^{\uparrow}$ *packet. Suppose that the membrane gate surface G_M is the finite audit face obtained from the source-token row S, the dependency row D_A, the paper-Lean target-resolution row L, the origin/import row O, and the* GAP *row G_A of A. Then every boundary item refused by the corresponding* $\mathsf{AuditGateBoundary}^{\uparrow}$ *row remains refused by the membrane row R. In particular, a refused source axiom, forbidden host dependency, unresolved target name, untracked origin/import edge, or hidden global-correctness crossing has no exported route from M into substrate theorem content.*

Lean4 ✓ `BEDC.Derived.AuditMembraneUp.AuditMembraneUp StdBridge`

Proof. Project the gate-boundary packet by Definition 526.1. Its displayed audit face consists of the source-token row S, the dependency row D_A, the paper-Lean target row L, the origin/import row O, and the GAP row G_A, together with transport, continuation, provenance, and local naming rows.

The refusal content of these rows is fixed inside the gate-boundary packet. By Definition 526.2, source-axiom admission, forbidden host-dependency admission, unresolved marker names, untracked origin flow, global semantic conservativity, and host-kernel completeness are non-escaping objects for A. By Definition 526.3, the rows S and D_A decide the source-token and forbidden-dependency boundary displayed by A. By Definition 526.4, L permits only the named finite resolution result and gives no route through a nearby declaration or unresolved string. By Definition 526.5, O exposes only the finite origin/import ledger and gives no semantic conservativity

theorem for arbitrary import flow. The row G_A records the matching GAP boundary: the gate refuses the displayed boundary crossings and does not provide a hidden theorem of global correctness.

Now project the membrane packet by Definition 524.1. The hypothesis identifies its gate surface G_M with precisely the displayed audit face S, D_A, L, O, G_A read from A. Hence an item refused at the gate-boundary side is the same displayed boundary item when it is presented in the membrane boundary-content row B: source axioms are read through S, forbidden dependencies through D_A, target names through L, origin/import edges through O, and hidden global-correctness crossings through G_A.

By Definition 524.3, the membrane refusal row R is the complete public result of applying G_M to B. Therefore the classification already refused by the corresponding row of A cannot be changed by the membrane packet; it appears in R as refused boundary content rather than admitted substrate supply.

Finally, Definition 524.6 confines every downstream consumer of M to the finite rows $G_M, B, R, D_M, H_M, C_M, P$ These rows allow inspection, transport, replay, provenance, and local naming, but none of them exports a theorem object for refused boundary content. Thus every item refused by the instantiated $\mathsf{AuditGateBoundary}^{\uparrow}$ surface remains refused at the $\mathsf{AuditMembrane}^{\uparrow}$ boundary and cannot become substrate theorem content. □

Theorem 524.8 (AuditMembrane repeated consumer replay boundary)**.** *Let*

$$M = (G_M, B, R, D_M, H_M, C_M, P_M, N_M)$$

be an accepted $\mathsf{AuditMembrane}^{\uparrow}$ *packet. Any finite sequence of downstream reads of M factors through the same displayed rows $G_M, B, R, D_M, H_M, C_M, P_M, N_M$. Hence refused source axioms, forbidden dependencies, unresolved targets, and untracked origin edges remain refused under replay; no repeated consumer read exports them as substrate theorem content.*

Proof. Start with the refusal row. By Definition 524.3, the row R is the complete public result of applying the gate surface G_M to the boundary-content row B, and refused boundary content has no exported theorem route.

Now read a downstream consumer. The consumer boundary theorem, Definition 524.6, forces the read through the finite rows $G_M, B, R, D_M, H_M, C_M, P_M, N_M$ and provides no coordinate outside those rows.

For repeated reads, apply the same consumer-boundary factorization at each replay step. The refusal bridge of Definition 524.7 identifies the refused boundary items with the membrane refusal row, so replay cannot reclassify a refused source axiom, forbidden dependency, unresolved target, or untracked origin edge.

Thus every finite downstream replay remains inside the displayed membrane packet and preserves the refusal result. □

Chapter 525

A Concrete Naming Certificate for $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$

The $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ surface turns the recursor authorization principle of chapter 9373 into a finite NameCert packet. The packet records an accepted constructor signature, a legal eliminator name, complete branch data, structural descent evidence, output naming, audit provenance, and the visible boundary at which BEDC relies on the host kernel's acceptance of the recursor call. It does not add a new eliminator, a new induction principle, or a hidden source of proof terms outside the accepted signature.

The audit-provenance row A is the local consumer of the $\mathsf{OperationalMembrane}^{\uparrow}$ audit carrier in chapter 533. It reads source-axiom refusal, axiom-purity ancestry, and import-separation rows before the recursor output row O becomes public.

Definition 525.1 (AuthorizedGeneratorRecursor carrier)**.** *An* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

with the following displayed rows:

1. *I is the accepted constructor-signature row, naming the finite constructors of the inductive carrier and the strict-positivity gate that admits them;*
2. *E is the emitted eliminator row, naming the corresponding recursor I.rec as the only generator admitted by this packet;*
3. *M is the checked motive row, recording the output family or target judgment selected for the recursor call;*
4. *B is the branch-package row, containing exactly one accepted branch for each constructor listed by I;*
5. *D is the structural-descent row, recording the recursive arguments and the kernel-visible decrease or constructor subterm relation used by the call;*
6. *O is the output-name row, naming the term, proof object, or definition produced by the recursor call;*
7. *A is the audit-provenance row, recording the build check, source-axiom refusal, and transitive dependency refusal relevant to the generated output;*

8. *H records componentwise* hsame *transport for I, E, M, B, D, O, A, package provenance, boundary data, and the local naming row;*
9. *C records the displayed* Cont *routes by which downstream consumers read the signature, eliminator, motive, branch package, descent row, output name, and audit provenance;*
10. *P is the* Pkg *provenance over* BHist*, constructor signatures, recursor names, motive rows, branch packages, structural-descent rows, output names, audit reports,* hsame*,* Cont*,* GAP*, and* NameCert*;*
11. *G is the* GAP *boundary row recording that the packet relies on host-kernel acceptance of the displayed signature and recursor call, while refusing any unlisted branch, non-structural recursion channel, external axiom, or hidden proof-producing generator;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an extra constructor, a missing branch, an unchecked motive, a non-decreasing recursive call, an output without a name, an unaudited dependency channel, or a proof term generated outside I.rec and the accepted constructor signature.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTasteGate single carrier alignment`

Theorem 525.2 (AuthorizedGeneratorRecursor branch exhaustion)**.** *For an accepted* AuthorizedGeneratorRecursor *packet, the branch-package row B exhausts the generator surface named by E. Every public use of the packet reads one branch for each constructor displayed in I, transports those branches only through the componentwise row H, and cannot recover any additional branch or constructor case from the packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchExhaustion`

Proof. Project the carrier of Definition 525.1. The constructor signature I and emitted eliminator E are displayed together, and B is declared to contain exactly one accepted branch for each constructor listed by I.

Read downstream access through C. A consumer reaches the branch package only by the displayed continuation route, so the public surface is the finite list already paired with I.

Transport through H preserves the same component rows. Since the tuple has no coordinate for an extra constructor or unlisted branch, the branch surface is exhausted by the entries already recorded in B. □

Theorem 525.3 (AuthorizedGeneratorRecursor NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *surface for the carrier of Definition 525.1 consists exactly of signature admission, recursor naming, motive checking, branch exhaustion, structural-descent visibility, output naming, audit provenance, componentwise transport, continuation locality, package provenance, and the displayed* GAP *boundary. A downstream consumer may treat O as a recursor-generated named output only through the packet*

$$(I, E, M, B, D, O, A, H, C, P, G, N).$$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNameCertSurface`

Proof. Carrier exposure supplies the finite tuple. The rows $I, E, M, B, D, O, A, H, C, P, G, N$ are the only rows admitted to the local NameCert surface.

Signature admission and recursor naming are read from I and E. Motive checking, branch exhaustion, and structural-descent visibility are read from M, B, D, with branch exhaustion supplied by Definition 525.2.

The output becomes public through O only after the audit row A, transport row H, continuation row C, package row P, boundary row G, and local certificate row N are retained. No coordinate in the packet can replace these rows with an unnamed proof term or an external generator. □

Theorem 525.4 (AuthorizedGeneratorRecursor audit boundary)**.** *An accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet binds the audit boundary for recursor-authorized proof generation. The packet certifies that the displayed output is public only as the named result of the accepted signature, emitted eliminator, checked motive, complete branch package, structural-descent row, and audit-provenance row. It does not certify kernel completeness, global termination theory, semantic conservativity for all inductive declarations, or any output whose dependency chain escapes the displayed audit provenance.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditBoundary`

Proof. Start with the NameCert surface of Definition 525.3. It forces the output row O to be read together with the constructor signature, eliminator, motive, branch package, descent row, and audit row.

The boundary row G records the host-kernel handoff. It names the reliance on kernel acceptance of the signature and recursor call while refusing unlisted branches, non-structural recursion, external axioms, and hidden proof generators.

The excluded claims have no coordinate in the carrier. Therefore the packet gives a seed certificate for the displayed recursor-generated output and does not export global kernel completeness, unrestricted termination, general semantic conservativity, or an unaudited dependency surface. □

Theorem 525.5 (AuthorizedGeneratorRecursor structural descent locality)**.** *Every public recursive read of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet factors through the displayed constructor signature I, emitted eliminator E, checked motive M, branch package B, and structural-descent row D. The packet admits no public recursive call whose descent is not recorded in D, and it admits no branch read outside the constructor list displayed by I and exhausted by B.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorUp structural descent locality`

Proof. Project the carrier

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The only recursive generator named by the packet is the emitted eliminator row E, and it is paired with the constructor signature I, motive row M, branch package B, and descent row D.

Read access through C exposes those rows as the local public interface. By Definition 525.2, the branch package already contains exactly the public branches available for the constructors displayed in I.

The boundary row G refuses non-structural recursion and unlisted branches. Since the tuple has no coordinate for a second recursion channel, a non-decreasing recursive call, or an extra branch source, every public recursive read is local to I, E, M, B, D. □

Theorem 525.6 (AuthorizedGeneratorRecursor audit provenance factorization)**.** *For an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the output row O becomes public only after the audit-provenance row A, componentwise transports H, continuation routes C, package row P, boundary row G, and local name row N are retained. Thus public output exposure factors through the audit boundary of Definition 525.4 and cannot be read as an unaudited recursor result.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditProvenanceFactorizat`
Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor audit provenance factorization`

Proof. Project the carrier and retain the rows O, A, H, C, P, G, N. The NameCert surface of Definition 525.3 states that O is part of the local certificate only inside the displayed packet.

Apply Definition 525.4. It binds the public output to the accepted signature, emitted eliminator, checked motive, complete branch package, structural-descent row, and audit-provenance row.

The rows H, C, P, G, N keep the transports, public routes, package provenance, boundary refusal, and local naming certificate attached to that exposure. No coordinate remains from which a consumer can read O apart from A, H, C, P, G, N, so the output factors through the displayed audit provenance. □

Theorem 525.7 (AuthorizedGeneratorRecursor ground-compiler handoff)**.** *The* `\origin{ai}` *TasteGate marker for this chapter is consumed only as the single-carrier alignment witness for the finite* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet. It does not replace the paper-side obligations for branch exhaustion, structural-descent locality, audit provenance factorization, or the visible handoff to the host kernel at G.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundCompilerHandoff`

Proof. The checked TasteGate marker attached to Definition 525.1 identifies the chapter's finite carrier as the aligned object used by the recursor packet. It does not add a new theorem row to the carrier and does not alter the closure status of any paper-side claim.

Branch exhaustion is supplied by Definition 525.2. Structural-descent locality is supplied by Definition 525.5, and audit exposure is supplied by Definition 525.6.

The handoff to the ground compiler is therefore explicit: the TasteGate witness aligns the finite carrier, while the displayed rows $I, E, M, B, D, O, A, H, C, P, G, N$ continue to carry the branch, descent, provenance, and boundary obligations inside the paper. □

Theorem 525.8 (AuthorizedGeneratorRecursor host-kernel handoff)**.** *Every accepted* AuthorizedGeneratorRecursor *packet that reaches the host-kernel boundary row G first exposes the authorized generator branch row B, structural-descent row D, audit-provenance row A, visible* Cont *route C, and local* NameCert$_{\text{AuthorizedGeneratorRecurs}}$ *row N. The handoff exports no free induction principle, hidden branch source, or proof-producing generator outside the displayed constructor signature and accepted generator row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorHostKernelHandoff`

Proof. Start with branch exhaustion and branch-descent totality, Definition 525.2 and Definition 525.13. They force public recursive reads to stay on the accepted constructor signature, branch package, and displayed descent row.

Use audit provenance factorization from Definition 525.6. It keeps public output attached to the audit row, continuation route, package provenance, boundary row, and local name row.

Finally cite the TasteGate handoff only as the single-carrier alignment witness, through Definition 525.7. Since the aligned carrier has no coordinate for a free induction principle or hidden generator, the host-kernel row G receives exactly the displayed branch, descent, audit, route, and local NameCert surface. □

Theorem 525.9 (AuthorizedGeneratorRecursor kernel-boundary exhaustion)**.** *Every public recursor read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet factors through exactly*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The read cannot introduce an unlisted constructor, a missing branch, a non-structural recursive call, an external axiom, or a hidden proof generator beyond the host-kernel boundary recorded by G.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorKernelBoundaryExhaustion`

Proof. Start from the carrier definition in Definition 525.1. It displays the full packet and leaves no coordinate outside $I, E, M, B, D, O, A, H, C, P, G, N$.

Branch coverage is exhausted by Definition 525.2, while the local NameCert surface is fixed by Definition 525.3. Thus a consumer may read only the listed constructor, eliminator, motive, branch, output, transport, route, package, boundary, and naming rows.

Structural recursion is confined by Definition 525.5. Audit exposure is confined by Definition 525.4 and Definition 525.6.

Finally apply Definition 525.7. The TasteGate handoff aligns the same finite carrier and does not add a constructor, branch, recursion channel, axiom row, or proof-producing generator outside the displayed kernel boundary. □

Theorem 525.10 (AuthorizedGeneratorRecursor output-name stability)**.** *For an accepted* AuthorizedGeneratorRec *packet, the output-name row O is stable only when the accepted signature I, eliminator E, motive M, branch package B, descent row D, audit row A, componentwise transports H, displayed routes C, boundary row G, and local name row N are transported together. A consumer cannot keep the public output name while replacing any one of these supporting rows by an unlisted row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputNameStability`

Proof. Project the carrier of Definition 525.1. The row O is not a standalone output; it is one coordinate of the displayed packet with $I, E, M, B, D, A, H, C, P, G, N$.

Use Definition 525.3 to read the output through the local certificate, and use Definition 525.2 with Definition 525.5 to keep the signature, branch package, and descent row attached to that read.

Now apply Definition 525.4 and Definition 525.6. They keep the audit row, transports, routes, package provenance, boundary row, and naming row attached before O becomes public.

The TasteGate handoff of Definition 525.7 consumes the same carrier. Therefore any transported public output name preserves the whole displayed packet surface, not a detached name row. □

Theorem 525.11 (AuthorizedGeneratorRecursor audit-row completeness)**.** *The audit row A is complete for the accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet in the following finite sense: before O is public, A must expose the build check, source-axiom refusal, dependency refusal, and operational-membrane provenance consumed by the packet. This audit completeness does not prove kernel completeness, global termination, or a proof generator outside the accepted recursor boundary.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditRowCompleteness`

Proof. Begin with the carrier definition and its audit-provenance row. The displayed packet places A before public output exposure and ties it to $I, E, M, B, D, O, H, C, P, G, N$.

Apply Definition 525.4. It states that public output is certified only as the named result of the accepted signature, emitted eliminator, checked motive, complete branch package, structural-descent row, and audit-provenance row.

Next use Definition 525.6. The build check, source-axiom refusal, dependency refusal, operational provenance, transports, routes, boundary, and local name remain in the same packet before O is exposed.

Branch exhaustion, structural descent locality, and the TasteGate handoff are supplied by Definition 525.2, Definition 525.5, and Definition 525.7. Since the carrier has no coordinate for kernel completeness or global termination, the audit row is complete only for the finite packet boundary stated above. □

Theorem 525.12 (AuthorizedGeneratorRecursor operational membrane audit handoff)**.** *For an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the audit row A must consume the* OperationalMembrane$^{\uparrow}$ *provenance before the output row O becomes public. The public output route therefore factors through the displayed build check, source-axiom refusal, dependency refusal, operational-membrane provenance, output-name row, and local* NameCert *row, all inside*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOperationalMembraneAuditH`

Proof. Project the carrier of Definition 525.1. The row A records the build check, source-axiom refusal, dependency refusal, and operational-membrane provenance, while O is only the displayed output-name row of the same packet.

Apply Definition 525.4 and Definition 525.6. These rows place public exposure of O behind the audit boundary and keep A, H, C, P, G, N attached to that exposure.

Use Definition 525.10 with Definition 525.11. The output name cannot be transported apart from the audit row, and the audit row is complete only after it exposes the operational-membrane provenance consumed by the packet. □

Theorem 525.13 (AuthorizedGeneratorRecursor branch descent totality)**.** *For an accepted* AuthorizedGeneratorRe *packet, the accepted branch package B and structural-descent row D jointly exhaust all public recursive reads. A public read may reach the constructor signature, emitted eliminator, checked motive, accepted branches, descent evidence, output name, audit row, transports, routes, package provenance, boundary row, and local certificate, but it cannot read a recursive branch outside B or a recursive call whose descent is absent from D.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchDescentTotality`

Proof. Start from the carrier definition. The only public recursive generator named by the packet is the emitted eliminator E, and the rows adjacent to it are the accepted signature I, checked motive M, branch package B, and descent row D.

Apply Definition 525.2. It says that B contains exactly the public branch surface associated with the constructors listed by I.

Apply Definition 525.5. It confines every public recursive read to the displayed signature, eliminator, motive, branch package, and structural-descent row.

The audit boundary and output-name stability, supplied by Definition 525.4 and Definition 525.10, keep the resulting public output attached to the same packet. Thus B and D jointly exhaust the recursive read surface. □

Theorem 525.14 (AuthorizedGeneratorRecursor hidden generator exclusion)**.** *No public route out of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet can recover an unlisted constructor, unchecked branch, non-structural recursion channel, external axiom, or hidden proof generator. Every public route is a projection of the carrier rows $I, E, M, B, D, O, A, H, C, P, G, N$ together with the displayed* GAP *boundary.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorHiddenGeneratorExclusion`

Proof. Read the carrier definition and the boundary row G. The tuple has no coordinate for an extra constructor, unchecked branch, non-structural recursion channel, external axiom, or hidden proof-producing generator.

Use branch exhaustion and branch-descent totality. By Definition 525.2 and Definition 525.13, the public recursive surface is exhausted by the displayed constructors, accepted branches, and structural descent row.

Use audit provenance and output-name stability. By Definition 525.6, Definition 525.11, and Definition 525.10, public output remains attached to the audit row and cannot detach into an unaudited generator route.

Finally apply Definition 525.7. The checked carrier alignment is consumed only for the same finite packet, so it cannot add an external axiom or hidden generator beyond G. □

Theorem 525.15 (AuthorizedGeneratorRecursor scoped packet)**.** *The branch exhaustion row, local* NameCert *surface, structural-descent locality row, audit-provenance factorization row, operational-membrane audit handoff, branch-descent totality row, hidden-generator exclusion row,* GAP *boundary row, and checked TasteGate single-carrier handoff form the scoped finite packet for* AuthorizedGeneratorRecursor$^{\uparrow}$ *The packet exposes only the accepted constructor signature, emitted eliminator, checked motive, complete branch package, structural-descent row, named output, audit provenance, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance,* GAP *boundary, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorScopePackage`

Proof. Start with Definition 525.2 and Definition 525.3. They bind the constructor signature, emitted eliminator, motive, branch package, output name, transport, route, provenance, boundary, and local naming rows to the same finite carrier.

Next apply Definition 525.5. It keeps every public recursive read on the displayed signature, eliminator, motive, branch package, and structural-descent row.

Use Definition 525.4 and Definition 525.6 to attach the audit-provenance row and GAP boundary to the public output route.

Use Definition 525.12, Definition 525.13, and Definition 525.14. These rows make the operational audit handoff, the exhaustive recursive-read surface, and the refusal of hidden generators explicit inside the same packet.

Finally apply Definition 525.7. The checked TasteGate alignment row is read only as the existing single-carrier handoff, so the scoped packet remains the displayed finite tuple and does not export an unlisted branch, non-structural recursion channel, unaudited output, or hidden proof-producing generator. □

Theorem 525.16 (AuthorizedGeneratorRecursor signature route totality)**.** *Every public read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet is total only as a route through the accepted signature I, eliminator E, motive M, branch package B, descent row D, output name O, audit row A, componentwise transports H, displayed routes C, provenance row P,* GAP *row G, and local name N. No downstream recursor consumer can detach the public output from the accepted signature route.*

Proof. Start with Definition 525.2 and Definition 525.3. They bind the constructor signature, eliminator, motive, branch package, output name, transport, route, provenance, boundary, and local naming rows to one finite packet.

Next apply Definition 525.5. Every recursive read stays on the displayed signature, eliminator, motive, branch package, and descent row.

Finally use Definition 525.10 and Definition 525.14. The output name cannot be transported apart from the signature, descent, audit, and boundary rows, and the packet has no coordinate for a hidden generator. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorSignatureRouteTotality`

Theorem 525.17 (AuthorizedGeneratorRecursor output audit ledger)**.** *For an accepted* AuthorizedGeneratorRecurs *packet, the output name O is public only together with the audit row A, transports H, routes C, provenance P, boundary G, and local name N over the same I, E, M, B, D recursor route. The ledger refuses any public output whose dependency or audit provenance escapes the displayed packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputAuditLedger`

Proof. Begin with Definition 525.4. It places the public output behind the accepted signature, emitted eliminator, checked motive, complete branch package, structural-descent row, and audit row.

Apply Definition 525.6 and Definition 525.12. These attach build checking, source-axiom refusal, dependency refusal, operational provenance, transports, routes, package provenance, boundary, and local naming before O is public.

Use Definition 525.10. It keeps O tied to the same $I, E, M, B, D, A, H, C, P, G, N$ rows, so the audit ledger cannot be bypassed by an unaudited output route. □

Theorem 525.18 (AuthorizedGeneratorRecursor public consumer boundary)**.** *Every downstream consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet reads the public boundary through $I, E, M, B, D, O, A, H, C, P, G, N$. The boundary contains branch coverage, structural descent, audit provenance, operational membrane handoff, output-name stability, and hidden-generator refusal, and it contains no unlisted branch, detached output name, non-structural recursion channel, external axiom, or proof-producing generator outside G.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPublicConsumerBoundary`

Proof. First combine Definition 525.16 and Definition 525.17. The public consumer receives the accepted signature route and the audit ledger as one packet.

Next apply Definition 525.13 with Definition 525.12. These rows exhaust recursive branch reads and attach the operational membrane audit surface before the output is exposed.

Finally use Definition 525.14 and Definition 525.7. The checked carrier alignment is only the handoff for the same finite packet, so no unlisted branch, detached output, external axiom, or hidden proof generator can cross the public boundary. □

Theorem 525.19 (AuthorizedGeneratorRecursor root branch row totality)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet*

$$(I, E, M, B, D, O, A, H, C, P, G, N),$$

every branch row used by the recursor is exposed in the displayed branch packet B *before any output row* O *is public. The public output cannot be supplied by a hidden generator, an omitted branch, or a route outside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchRowTotality`

Proof. Start from the carrier definition and Definition 525.2. The accepted packet has one displayed constructor signature, emitted eliminator, checked motive, and complete branch package, so every recursive branch read is already accounted for in B.

Apply Definition 525.4 and Definition 525.17. These results place the output row behind the same signature, branch, descent, audit, transport, continuation, provenance, boundary, and local-name rows.

Finally use Definition 525.14 and Definition 525.18. They rule out a proof-producing generator or omitted branch coordinate outside the displayed packet, so O can become public only after branch totality has been read through B. □

Theorem 525.20 (AuthorizedGeneratorRecursor audit descent row)**.** *For an accepted* AuthorizedGeneratorRecursor *packet* $(I, E, M, B, D, O, A, H, C, P, G, N)$*, the structural descent row* D *and audit row* A *are read through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet. Audit provenance therefore cannot detach from the recursor branch route, and no descent certificate may be consumed without the displayed audit boundary.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditDescentRow`

Proof. Use the carrier definition together with Definition 525.5. The descent row belongs to the same finite tuple as the signature, eliminator, motive, and branch package, and every recursive read stays on those displayed coordinates.

Next apply Definition 525.4 and Definition 525.17. The audit row is exposed only through the continuation, transport, package, boundary, and local naming data attached to the same packet.

The non-detachment claim follows from Definition 525.14 and Definition 525.18. Since no hidden generator, omitted branch, or outside output route can cross the public boundary, the audit row and descent row remain one packet-local route. □

Theorem 525.21 (AuthorizedGeneratorRecursor NameCert root unblock package)**.** *The carrier row, branch-totality row, audit-descent row, hidden-generator exclusion, output audit ledger, and public consumer boundary form the root* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *obligation package for downstream consumers of the displayed packet* $(I, E, M, B, D, O, A, H, C, P, G, N)$*. A consumer may use the authorized recursor output only through that package.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNameCertRootUnblockPackag`

Proof. First combine the carrier definition with Definition 525.19. The root package begins with the displayed packet and the complete branch surface that precedes public output.

Next use Definition 525.20 and Definition 525.17. These bind structural descent, audit provenance, output exposure, transport, route, and package provenance into one BHist-local NameCert object.

Finally apply Definition 525.14 and Definition 525.18. The downstream consumer receives exactly the displayed root package, with no unlisted branch, hidden generator, detached audit row, or outside public-output route. □

Theorem 525.22 (AuthorizedGeneratorRecursor accepted signature cover)**.** *Every public route exposing the emitted eliminator E, checked motive M, branch package B, structural descent row D, or output name O of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet first exposes the accepted constructor-signature row I. Outside the I, E surface, the packet provides no unlisted constructor signature and no alternate eliminator.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptedSignatureCover`

Proof. Project the public carrier through Definition 525.3. The displayed surface lists $I, E, M, B, D, O, A, H, C, P, G$ as one finite packet, so the eliminator, motive, branch, descent, and output rows are not independent public entries.

Apply Definition 525.2. It keeps all branch reads on the accepted constructor signature and the emitted eliminator, so a consumer cannot obtain M, B, D, or O before the I, E route has been read.

Finally use Definition 525.14. The carrier has no coordinate for a hidden proof-producing generator, unlisted constructor, or alternate eliminator, so the accepted signature cover exhausts the public recursor route. □

Theorem 525.23 (AuthorizedGeneratorRecursor motive branch totality)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the checked motive row M and branch package row B are consumed together with the structural descent row D. No branch can become public without the motive that types it and the descent row that authorizes its recursive reads.*

Proof. Start with Definition 525.5. It places every public recursive read on the same signature, eliminator, motive, branch package, and structural-descent row.

Next apply Definition 525.13. The branch surface is exhaustive only as a package over the checked motive and the descent row, so no single branch can be projected as a public route apart from M and D.

Use Definition 525.10. The output name remains tied to the same I, E, M, B, D route, which prevents a branch consumer from exporting a detached branch result under a stable public name. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorMotiveBranchTotality`

Theorem 525.24 (AuthorizedGeneratorRecursor GAP boundary refusal)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the* GAP *boundary row G refuses external axioms, hidden proof generators, non-structural recursion, and unlisted branches. The output name O becomes public only through the audit row A, transports H, routes C, provenance P, boundary G, and local name N.*

Proof. Begin with Definition 525.4. It places public output behind the accepted signature route, checked motive, complete branch package, structural descent, and audit row.

Apply Definition 525.6. The audit provenance factors through the displayed transport, route, package, and boundary rows, so O has no public path that bypasses A, H, C, P, G, N.

Next use Definition 525.12. The operational membrane reads the same finite packet and does not add a non-structural recursion channel or an unlisted branch.

Finally apply Definition 525.14. Since the carrier has no hidden proof-generator coordinate, the GAP row is the public refusal boundary for external axioms, hidden generators, non-structural recursion, and branch escape. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGapBoundaryRefusal`

Theorem 525.25 (AuthorizedGeneratorRecursor output boundary non-escape)**.** *For an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the output row O cannot be read apart from the accepted signature I, emitted eliminator E, checked motive M, branch package B, structural-descent row D, audit row A, transports H, routes C, provenance P, boundary row G, and local name N. The packet exports no detached output, hidden generator, external proof term, or output route outside these displayed rows.*

Proof. Start from branch exhaustion and the local NameCert surface. Definition 525.2 and Definition 525.3 force the output read to retain the accepted signature, eliminator, motive, complete branch package, descent row, output naming row, audit provenance, routes, package provenance, boundary row, and local certificate.

Next use audit boundary, audit provenance, and output-name stability. Definition 525.4, Definition 525.6, and Definition 525.10 keep O attached to A, H, C, P, G, N before it is exposed.

Finally apply audit-row completeness, branch descent totality, hidden-generator exclusion, and the TasteGate handoff. Definition 525.11, Definition 525.13, Definition 525.14, and Definition 525.7 leave no coordinate for a detached output or proof-producing generator. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputBoundaryNonescape`

Theorem 525.26 (AuthorizedGeneratorRecursor GAP boundary totality)**.** *Every host-kernel handoff through the boundary row G of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet is preceded by the signature row I, branch package B, structural-descent row D, audit row A, displayed route row C, package row P, and local* NameCert *row N. The* GAP *boundary receives only the displayed recursor packet and never receives a hidden branch source, unchecked descent claim, unaudited dependency channel, or proof-producing generator outside the packet.*

Proof. Begin with the host-kernel boundary rows. Definition 525.8 sends any G-read through the branch, descent, audit, route, and local certificate rows, while Definition 525.16 attaches the signature route to the same packet.

Use the audit and branch synchronization supplied by the chapter. Definition 525.6, Definition 525.11, and Definition 525.13 place the audit, package, branch, and descent rows before any boundary exposure.

Finally use hidden-generator exclusion and the TasteGate handoff. Definition 525.14 blocks external generators, and Definition 525.7 reads the checked carrier alignment only as the finite packet handoff. Thus the G boundary is total for the displayed signature, branch, descent, audit, route, package, and naming rows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGapBoundaryTotality`

Theorem 525.27 (AuthorizedGeneratorRecursor audit branch synchronization)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, branch exhaustion and audit provenance are synchronized before any public output exposure. A consumer that reads O must first read the complete branch package B, structural-descent row D, audit row A, operational audit handoff, output-name row, transports, routes, provenance, boundary row, and local name together, and it cannot separate the audit ledger from the public branch descent route.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditBranchSynchronizatio

Proof. First synchronize the branch rows. Definition 525.2 and Definition 525.13 exhaust public recursive reads through the accepted branch package and structural descent row.

Next synchronize the audit rows. Definition 525.4, Definition 525.6, Definition 525.11, and Definition 525.12 place audit provenance before public exposure of O.

Finally attach output naming and the public handoff. Definition 525.10, Definition 525.14, and Definition 525.7 prevent a detached output name, hidden generator, or carrier-alignment read from bypassing the synchronized branch and audit rows. □

Theorem 525.28 (AuthorizedGeneratorRecursor kernel acceptance scope)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, any public host-kernel handoff through the boundary row G is scoped by the displayed rows $I, E, M, B, D, O, A, H, C, P, N$. The handoff authorizes only the finite carrier tuple of Definition 525.1; it cannot be read as an unbounded eliminator principle, a global induction rule, or a public recursion channel outside those rows.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorKernelAcceptanceScope

Proof. Project the carrier tuple $(I, E, M, B, D, O, A, H, C, P, G, N)$. The rows I, E, M, B, D give the accepted signature, emitted eliminator, checked motive, branch package, and structural descent surface, while O, A, H, C, P, N give the named output, audit row, transports, routes, provenance, and local certificate.

Use branch exhaustion and structural locality. By Definition 525.2 and Definition 525.5, every public recursive read stays on the accepted constructor signature, the checked motive, the branch package, and the structural-descent row.

Attach the audit and output rows. Apply Definition 525.4 and Definition 525.10; public exposure of O remains behind the audit surface and cannot keep a stable name after any supporting row is detached.

Finally use hidden-generator exclusion and the ground-compiler handoff. Definition 525.14 and Definition 525.7 leave no carrier coordinate for an unbounded eliminator, global induction rule, or external recursion source beyond the displayed G handoff. □

Theorem 525.29 (AuthorizedGeneratorRecursor branch motive pairing)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the motive row M and branch row B are paired constructor-by-constructor against the accepted signature row I. A downstream recursor consumer may not detach a branch from the motive that types it or from the constructor case in I that it covers.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchMotivePairing

Proof. Project the same carrier tuple and read the I, M, B, D coordinates together. The accepted signature supplies the constructor cases, the motive supplies the target family for those cases, the branch row supplies the case data, and the descent row supplies the legal recursive reads.

Apply branch exhaustion and structural descent locality. By Definition 525.2 and Definition 525.5, branch coverage is exhausted only as a package over the accepted constructors and the checked motive, with recursive reads confined to the displayed descent row.

Use the audit boundary and output-name stability to make the pairing public. Definition 525.4 places output exposure behind the same I, M, B, D, A packet, and Definition 525.10 prevents a stable public output from retaining B while dropping M or I.

Finally cite hidden-generator exclusion and the ground-compiler handoff. Definition 525.14 blocks a replacement branch source, and Definition 525.7 reads the TasteGate alignment only for the finite packet. Hence each branch remains paired with its motive and constructor case. □

Theorem 525.30 (AuthorizedGeneratorRecursor audit dependency refusal)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the audit row A refuses external axiom channels and hidden dependency surfaces before the output row O becomes public. The refusal is the same operational-membrane route used by Definition 525.12 and the same audit boundary used by the displayed packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditDependencyRefusal`

Proof. Project $(I, E, M, B, D, O, A, H, C, P, G, N)$. The audit row is not a detached label: it is packaged with the accepted signature, emitted eliminator, motive, branch data, structural descent, transports, routes, provenance, boundary, and local certificate before O is exposed.

Use branch exhaustion and structural locality first. By Definition 525.2 and Definition 525.5, any dependency surface visible to a public branch read is already confined to the accepted constructors, motive, branch package, and descent row.

Now apply the audit boundary and the operational membrane handoff. Definition 525.4 and Definition 525.12 place public exposure of O behind the build check, source-axiom refusal, dependency refusal, and operational-membrane provenance carried by A.

Close the dependency refusal with output-name stability, hidden-generator exclusion, and the ground-compiler handoff. Definition 525.10, Definition 525.14, and Definition 525.7 prevent a stable output name, hidden generator, or host-kernel handoff from bypassing the audited dependency surface. □

Theorem 525.31 (AuthorizedGeneratorRecursor branch-ledger disjointness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, a row consumed as branch data in B cannot also be consumed as an audit row in A or as a structural-descent row in D. The public recursor surface keeps branch coverage, descent reads, and audit provenance as disjoint displayed ledger roles inside the single carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchLedgerDisjointness`

Proof. Project the carrier $(I, E, M, B, D, O, A, H, C, P, G, N)$ of Definition 525.1. The branch, descent, and audit coordinates are separately named rows of the finite packet, and their consumer roles are introduced by the local NameCert surface rather than by a shared ambient row.

Use Definition 525.2 for the branch side and Definition 525.5 for the descent side. These theorems confine branch coverage to the accepted constructor and motive rows, while descent reads remain tied to the displayed structural-decrease row.

For the audit side, apply Definition 525.4 and Definition 525.6. They expose A only as the provenance and refusal surface behind public output, not as branch payload or recursive descent data.

Finally Definition 525.7 reads the same separated carrier coordinates at the TasteGate handoff. Thus no public consumer can reclassify a branch row as audit or descent while retaining an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ packet. □

Theorem 525.32 (AuthorizedGeneratorRecursor descent-output lock)**.** *For every accepted* AuthorizedGeneratorRec *packet, the public output row O remains locked to the displayed structural-descent row D. A consumer may expose O only through the same finite descent ledger that licensed the recursive reads used by the packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorDescentOutputLock`

Proof. Read O and D from the carrier of Definition 525.1. The output row is not a detached result name: it is packaged with the accepted signature, motive, branch data, structural descent, audit surface, transports, routes, provenance, and local certificate.

Apply Definition 525.5. Every recursive read used by the packet is confined to the displayed structural descent row D.

Next combine Definition 525.13 with Definition 525.10. Branch coverage supplies public output only together with its descent obligations, and the output name loses stability if the supporting descent row is detached.

The audit boundary and ground-compiler handoff preserve the same lock. Definition 525.4 keeps public exposure behind the finite packet, and Definition 525.7 transfers only that packet to the TasteGate surface. Hence O cannot be exposed apart from the structural descent row D. □

Theorem 525.33 (AuthorizedGeneratorRecursor audit-provenance cover)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, each public exposure of the output row O consumes the audit provenance rows already listed in the carrier. No output exposure may cite an external audit source or omit the displayed provenance factorization.*

Proof. Begin with the carrier inventory in Definition 525.1. The audit row A and provenance row P are displayed before any public route exposes O, and the local certificate names them as part of the same finite packet.

Apply Definition 525.4. Public output exposure is behind the audit boundary, so it must pass through the displayed audit surface.

Use Definition 525.6 to resolve that surface into the provenance rows already carried by P and A. The operational route of Definition 525.12 then presents the same provenance to the consumer side.

Finally Definition 525.7 permits the ground compiler to read only the finite carrier packet. Therefore every public exposure of O consumes the listed audit provenance rows and cannot draw authority from an external audit channel. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier audit provenance cover`

Theorem 525.34 (AuthorizedGeneratorRecursor carrier admission)**.** *Every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet is admitted only as the finite* BHist *row*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

of Definition 525.1. The rows I, E, M, B, D, O, A display the accepted signature, emitted recursor, motive, branch package, structural descent, output name, and audit provenance, while H, C, P, G, N supply the componentwise hsame *transport, displayed* Cont *routes,* Pkg *provenance, visible boundary, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row.*

Proof. Project the carrier definition. It lists exactly the rows $I, E, M, B, D, O, A, H, C, P, G, N$ and admits no additional source for an authorized generator or recursor call.

Apply Definition 525.3. The local NameCert surface reads the accepted signature, emitted eliminator, checked motive, complete branch package, structural descent, output row, and audit row only inside the same finite packet.

The remaining rows are structural rather than generative. The hsame, Cont, Pkg, boundary, and local NameCert rows record transport, replay, provenance, the visible host boundary, and naming for the displayed carrier, so carrier admission is exhausted by the finite BHist row. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier admission surface`

Theorem 525.35 (AuthorizedGeneratorRecursor ledger descent)**.** *Every public ledger read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet descends through the displayed generator and recursor rows*

$$I \to E \to M \to B \to D \to O \to A$$

before it may reach the componentwise hsame *row* H*, displayed* Cont *row* C*,* Pkg *row* P*, boundary row* G*, or local* NameCert *row* N*. The descent cannot be replaced by an unregistered branch ledger, hidden output source, or external recursor certificate.*

Proof. Start from carrier admission in Definition 525.34. The admitted BHist packet exposes the accepted signature and emitted recursor before motive checking, branch coverage, structural descent, output naming, and audit provenance.

Use Definition 525.5 and Definition 525.13. These rows force branch consumption through the structural descent coordinate D and keep the output row attached to the displayed branch package rather than to an external ledger.

Finally apply the NameCert surface. The rows H, C, P, G, N only transport, replay, package, bound, and name the same route, so no hidden branch ledger or external recursor certificate can replace the displayed descent chain. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorLedgerDescent`

Theorem 525.36 (AuthorizedGeneratorRecursor consumer nonescape)**.** *No downstream consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet can demand an unregistered recursion principle, hidden host induction, unchecked generator row, or proof-producing branch source outside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Every consumer-facing read must factor through carrier admission and ledger descent for* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorConsumerNonescape`

Proof. First use carrier admission to identify the finite packet and ledger descent to identify the only route by which the accepted generator and recursor rows reach the output and audit rows.

Next apply Definition 525.14. It rules out a hidden generator source beyond the accepted constructor signature, emitted eliminator, motive, complete branch package, descent row, output row, audit row, and local certificate.

It remains to account for consumers. The public consumer boundary of Definition 525.18 and the root consumer factorization of Definition 525.52 keep every downstream read inside the same BHist, hsame, Cont, Pkg, and NameCert surface. Hence an unregistered recursion principle or hidden host induction has no coordinate through which to enter the packet. □

Corollary 525.37 (AuthorizedGeneratorRecursor admission strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse or reuse route in the root admission surface of* R*. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read tries to add, omit, replace, detach, or bypass one of the displayed admission coordinates*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every permitted consumer-facing use factors through the finite BHist *carrier, the componentwise* hsame *transport row, the displayed* Cont *routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecurs}}$ *row. The same displayed row cannot be reinterpreted as a global impossibility, a host theorem about recursor admission, hidden external validation of the failed route, an unregistered recursion principle, hidden host induction, an unchecked generator row, or an off-packet proof-producing branch source.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAdmissionStrictObstructio`

Proof. Carrier admission, Definition 525.34, fixes the admitted object as the single finite BHist row

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The generative part of admission is exhausted by I, E, M, B, D, O, A: accepted signature, emitted recursor, checked motive, branch package, structural descent, output name, and audit provenance. The remaining rows H, C, P, G, N are structural admission data: componentwise transport, displayed continuation routes, package provenance, visible boundary, and local naming. Thus admission supplies no coordinate for a second generator, an omitted branch source, a host-side recursion theorem, an external validation channel, or an off-packet proof producer.

Consumer nonescape, Definition 525.36, turns that carrier statement into the consumer-facing constraint. Every downstream read must factor through carrier admission and ledger descent for the same R, and no consumer may demand an unregistered recursion principle, hidden host induction, unchecked generator row, or proof-producing branch source outside the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Now consider the displayed failure, refusal, separation, or countermodel row. If the named converse or reuse route asks for one of the listed coordinates and the displayed row denies that read, the obstruction blocks that named route at that coordinate. If the same route instead asks to read a global impossibility, a host recursor theorem, hidden external validation, an unregistered recursion principle, hidden host induction, an unchecked generator row, or an off-packet proof-producing branch source, it asks for a coordinate absent from carrier admission and rejected by consumer nonescape. Hence the displayed row has force only as local data against the named route, while every listed consumer read that remains on $I, E, M, B, D, O, A, H, C, P, G, N$ stays within the accepted packet. □

Theorem 525.38 (AuthorizedGeneratorRecursor root authorization downstream surface)**.** *Every downstream read of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet is authorized only through the finite carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The authorization row, output audit row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, host-boundary row, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row must remain in the same* BHist *packet before a consumer may read the emitted recursor, branch package, structural descent, or output row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAuthorizationDownstre`

Proof. Begin with Definition 525.34. It admits the root object only as the displayed finite carrier and leaves no separate authorization channel.

Apply Definition 525.35 and Definition 525.36. Together they force downstream consumers to traverse the accepted signature, emitted recursor, motive, branch package, descent row, output

row, audit row, transport, continuation, provenance, boundary, and local naming rows in one packet.

Finally use the root obligation surfaces in Definition 525.76 and Definition 525.81. The output audit and authorization ledgers therefore reach downstream consumers only through finite BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 525.39 (AuthorizedGeneratorRecursor Real stream handoff row)**.** *If an accepted* AuthorizedGeneratorRec *packet is used by an L10* Real$^{\uparrow}$, StreamName$^{\uparrow}$, *or* RegSeqRat$^{\uparrow}$ *consumer, then the handoff is the same authorized route* $I, E, M, B, D, O, A, H, C, P, G, N$. *The Real, stream-name, and regular-rational data may enter only as finite source rows carried by the displayed* BHist, *transported by* hsame, *replayed by* Cont, *packaged by* Pkg, *and named by the local* NameCert *row; no selected Real limit, quotient stream equality, host evaluation, or detached generator state is exported.*

Proof. Start with Definition 525.203. It already places every L10 Real, stream-name, and regular-rational read on the displayed packet.

Use Definition 525.86 to keep the output row behind the same authorized carrier before any downstream consumer receives it.

Finally apply Definition 525.38. The handoff row is therefore a finite authorized route with audit, transport, continuation, package, boundary, and local naming data, not a host-side stream or Real interface. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRealStreamHandoffRow`

Theorem 525.40 (AuthorizedGeneratorRecursor closed substitution gate row)**.** *The closed-substitution gate for an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet opens only after authorization, branch coverage, structural descent, output audit, Real/StreamName/RegSeqRat handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, host-boundary exposure, and local* NameCert *certification are all present in the finite route* $I, E, M, B, D, O, A, H, C, P, G, N$. *It cannot be opened by unrestricted substitution, unchecked recursion, external proof generation, host equality, or a private audit channel.*

Proof. First read the closed-substitution handoff of Definition 525.86. It keeps closed-substitution consumers on the displayed route and excludes unrestricted substitution and external proof generation.

Next attach the L10 handoff row from Definition 525.39. This supplies the Real, stream-name, and regular-rational source rows without leaving the finite carrier.

The gate is then closed by Definition 525.38 together with Definition 525.204. Authorization, audit, transport, continuation, provenance, boundary, and naming all remain packet-local, so there is no coordinate for unchecked recursion, host equality, or a private audit channel. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorClosedSubstitutionGateRow`

Lemma 525.41 (AuthorizedGeneratorRecursor root authorization bridge determinacy)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet be consumed through any downstream route, componentwise transport, readback, or L10 handoff. Then that consumer-facing bridge is determined by the finite carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The route part factors through I, E, M, B, D, O, A, *the transport and readback part factors through* H, C, P, *and the boundary-certificate part factors through* G, N. *For an L10* Real$^{\uparrow}$, StreamName$^{\uparrow}$, *or* RegSeqRat$^{\uparrow}$ *consumer, the source rows may enter only as packet-local* BHist, hsame, Cont,

Pkg, *boundary, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *data. No unlisted bridge coordinate is available for a selected Real limit, quotient stream equality, host evaluation, detached generator state, private audit row, off-packet readback, second transport, or second local naming certificate.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAuthorizationBridgeDe`

Proof. Start with the root authorization downstream surface, Definition 525.38. It authorizes every downstream read of an accepted root packet only through the finite carrier $R = (I, E, M, B, D, O, A, H, C, P, G, N)$. The same theorem requires the authorization row, output audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, host-boundary row, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row to remain in one BHist packet before the emitted recursor, branch package, structural descent, or output row can be read. Thus any consumer-facing bridge already has its route side confined to I, E, M, B, D, O, A and its audit-facing continuation confined to the displayed packet.

Apply the Real stream handoff row, Definition 525.39. It says that every L10 $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, or $\mathsf{RegSeqRat}^{\uparrow}$ use of the accepted packet is the same authorized route $I, E, M, B, D, O, A, H, C, P, G, N$. Its source data enter only as finite rows carried by BHist, transported by hsame, replayed by Cont, packaged by Pkg, and named by the local NameCert row. Hence a Real, stream-name, or regular-rational reading is not an additional bridge coordinate; it is the same route with the source rows read through H, C, P, G, N.

The two cited surfaces cover all bridge forms named in the statement. A displayed route is covered by the carrier rows and authorization ledger of Definition 525.38. A componentwise transport is exactly the retained hsame row H, a readback is exactly the displayed Cont replay and Pkg provenance rows C, P, and an L10 handoff is exactly the specialization supplied by Definition 525.39. The host boundary and local naming side is exhausted by G, N in the same packet.

It remains to exclude alternate bridges. Any selected Real limit, quotient stream equality, host evaluation, detached generator state, private audit row, off-packet readback, second transport, or second local naming certificate would have to be either a row of R or an exported interface allowed by the L10 handoff. The carrier displays no such row, and Definition 525.39 explicitly keeps the L10 data inside finite BHist, hsame, Cont, Pkg, boundary, and local NameCert rows. Therefore every consumer-facing route, transport, readback, or handoff factors through the cited local rows, while unlisted bridge coordinates remain unavailable. □

Corollary 525.42 (AuthorizedGeneratorRecursor root authorization determinacy)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be the fixed finite carrier of an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet. Once the authorization row, output audit row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, host-boundary row, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row of R are fixed, every accepted consumer-facing use of root authorization must use the same chapter-named finite route, packet-local normal-form reading, or exactness decision. The route side is fixed by I, E, M, B, D, O, A, the transport and readback side is fixed by H, C, P, and the boundary-certificate side is fixed by G, N. Hence two accepted consumers over the fixed rows cannot choose different authorization coordinates, a detached normal-form source, an off-packet readback, a private audit channel, a second transport, a second local naming certificate, or a duplicate classifier source.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier root authorization determinacy`

Proof. First apply Definition 525.38. It authorizes every downstream read of an accepted root packet only through the displayed finite carrier $R = (I, E, M, B, D, O, A, H, C, P, G, N)$, and it requires the authorization row, output audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, host-boundary row, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row to remain in the same BHist packet before a consumer may read the emitted recursor, branch package, structural descent, or output row. Thus an accepted consumer-facing route, normal-form reading, or exactness decision has no starting point outside those packet rows: it must read the recursor, descent, output, audit, transport, continuation, provenance, boundary, and local certificate as one finite surface.

Now use Definition 525.41. That lemma separates the fixed packet into the relevant roles. The route part factors through I, E, M, B, D, O, A, the transport and readback part factors through H, C, P, and the boundary-certificate part factors through G, N. It also covers the L10 $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff by forcing those source rows to enter only as packet-local BHist, hsame, Cont, Pkg, boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ data. Therefore a consumer-facing normal-form reading or exactness decision is not an independent classifier; it is the same fixed route read through the displayed transport, readback, provenance, boundary, and certificate rows.

It remains to rule out competing choices. Any alternate authorization coordinate, detached normal-form source, off-packet readback, private audit channel, second transport, second local naming certificate, or duplicate classifier source would have to supply either an additional row of R or an exported consumer interface not mediated by the rows named above. The downstream surface supplies no such row, and the bridge-determinacy lemma explicitly excludes unlisted bridge coordinates for selected Real limits, quotient stream equality, host evaluation, detached generator state, private audit rows, off-packet readbacks, second transports, and second local naming certificates. A duplicate classifier source would enter only through one of those excluded interfaces or through a carrier coordinate outside R. Since the rows of R are fixed and every accepted consumer use factors through them, all accepted consumers determine the same finite route and the same packet-local normal-form or exactness decision named by the chapter. □

Theorem 525.43 (AuthorizedGeneratorRecursor root branch row exposure)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the root branch row B is exposed before any downstream closed-normal or generator-audit consumer may read the output row O. The exposure is a projection of the carrier of Definition 525.1 through the local NameCert surface of Definition 525.3, and it is exhaustive by Definition 525.2.

Proof. Project the displayed packet $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ from Definition 525.1. The branch row B is one of its public coordinates, not a hidden generator source.

Apply the local NameCert surface. It exposes the output only together with the accepted signature, emitted eliminator, checked motive, branch package, structural descent, audit, transport, route, provenance, boundary, and local name rows.

Finally use branch exhaustion. The branch row contains exactly the public branch surface for the constructors displayed in I, so a downstream consumer cannot read O while bypassing B or inventing an additional branch row. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchRowExposure`

Theorem 525.44 (AuthorizedGeneratorRecursor root descent row exposure)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the structural-descent row D is exposed on every root-facing recursive read. The exposure is a projection of Definition 525.1 through Definition 525.3, and the locality of that row is forced by Definition 525.5.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootDescentRowExposure`

Proof. Read the carrier row D from $R = (I, E, M, B, D, O, A, H, C, P, G, N)$. It is displayed beside the accepted signature, emitted eliminator, checked motive, branch package, output name, audit row, and structural rows.

Use the NameCert surface to keep D attached to every public recursive read. The surface does not expose a recursive route whose descent evidence is absent from the packet.

Apply structural descent locality. It confines public recursion to the displayed signature, eliminator, motive, branch package, and descent row, so closed-normal and generator-audit consumers receive D before they can use O as an authorized recursor output. □

Theorem 525.45 (AuthorizedGeneratorRecursor root audit boundary exposure)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the audit-boundary rows A, H, C, P, G, N are exposed before any root consumer reads O as a recursor-authorized output. The exposure is a projection of Definition 525.1 through Definition 525.3, and it factors through Definition 525.6 together with the audit boundary of Definition 525.4.

Proof. Project A, H, C, P, G, N from the displayed packet. These rows carry audit provenance, componentwise transport, displayed continuation routes, package provenance, the GAP boundary, and the local name row.

Apply the NameCert surface. It exposes the output row only inside the same finite carrier as the branch, descent, audit, transport, route, provenance, boundary, and naming rows.

Use the audit boundary and audit provenance factorization. They place public output behind the accepted signature, emitted eliminator, checked motive, complete branch package, structural descent, audit provenance, continuation route, package row, boundary row, and local certificate, so no root consumer receives an unaudited recursor result. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAuditBoundaryExposure`

Theorem 525.46 (AuthorizedGeneratorRecursor root output audit totality)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, every public read of the output row O passes through the accepted constructor signature I, emitted eliminator E, checked motive M, branch package B, structural-descent row D, audit row A, transport row H, continuation row C, package provenance P, host-kernel boundary row G, and local name row N. Thus the public output is total only as the displayed NameCert packet*

$$(I, E, M, B, D, O, A, H, C, P, G, N),$$

not as a detached recursor result.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root output audit totality`

Proof. Start with the carrier and branch surface. The carrier of Definition 525.1 displays all rows named in the conclusion, and Definition 525.2 forces each branch read to stay on the constructor signature I, eliminator E, checked motive M, and branch package B.

Next add the recursive and audit rows. Definition 525.5 confines public recursive reads to the displayed descent row D, while Definition 525.4 places public exposure of O behind the audit row A.

It remains to keep the public output attached to the downstream rows. Definition 525.6 retains H, C, P, G, N at the output boundary, and Definition 525.10 prevents a stable public output from dropping the signature, motive, branch, descent, or audit rows. Therefore every public read of O consumes the whole displayed packet. □

Theorem 525.47 (AuthorizedGeneratorRecursor root boundary nonescape)**.** *Once an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet reaches the host-kernel boundary row G, it exports no free induction principle, no hidden proof generator, no external axiom channel, and no unaudited dependency route. The only public generator authority remaining at G is the accepted recursor call named by E over the finite rows $(I, M, B, D, O, A, H, C, P, N)$.*

Proof. First isolate the hidden-generator case. Definition 525.14 excludes any proof-producing source outside the accepted signature and recursor row.

The host-kernel handoff preserves that exclusion rather than widening it. Definition 525.7 reads the TasteGate marker only as the single-carrier alignment witness for the finite packet, so G does not export a second induction principle or a new proof generator.

Finally apply the audit refusal surface. Definition 525.30 blocks external axiom channels and hidden dependency routes before O becomes public. Hence a packet that has reached G has no non-displayed route by which generator authority, axiom authority, or audit authority can escape the carrier. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBoundaryNonescape`

Theorem 525.48 (AuthorizedGeneratorRecursor root NameCert consumer readiness)**.** *The rows*

$$(I, E, M, B, D, O, A, H, C, P, G, N)$$

of an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet form the root consumer-readiness surface for downstream closed-normal and generator-audit packets. A downstream consumer may cite this surface only as the conjunction of root output audit totality, root boundary nonescape, the* $\mathsf{OperationalMembrane}^{\uparrow}$ *audit handoff of chapter 533, and the local TasteGate alignment marker attached to Definition 525.1.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootNameCertConsumerReadi`

Proof. The output side is supplied by Definition 525.46. It requires every public read of O to consume the accepted signature, eliminator, motive, branch, descent, audit, transport, continuation, provenance, boundary, and local name rows.

The nonescape side is supplied by Definition 525.47. It prevents the same packet from exporting a free induction principle, hidden proof generator, external axiom channel, or unaudited dependency route after the boundary row G is reached.

The audit handoff is the local consumer of chapter 533. Together with Definition 525.12 and the displayed TasteGate alignment marker, it makes the packet ready for downstream closed-normal and generator-audit consumers without granting any authority beyond the finite NameCert surface. □

Theorem 525.49 (AuthorizedGeneratorRecursor obligation triad)**.** *For every accepted* AuthorizedGeneratorRecurso *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, obligation closure is reached only through the triad of branch-package exhaustion over* I, E, B*, structural-descent locality over* M, D*, and audit-provenance factorization over* O, A, H, C, P, G, N*. No output row* O *may be public when any one of these three components is absent from the same carrier packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorObligationTriad`

Proof. Read the packet coordinates from Definition 525.1. The carrier presents one finite surface, so branch data, descent data, audit provenance, routes, package provenance, boundary rows, and the local certificate are consumed as coordinates of the same accepted packet.

The first component is branch coverage. Apply Definition 525.2 to force the public branch package to exhaust the constructors displayed by I and the emitted eliminator row E.

The second component is descent locality. Apply Definition 525.5 to keep recursive reads on the motive and structural-descent rows M, D, rather than on a detached generator route.

The third component is audit provenance. Use Definition 525.6 and Definition 525.33 to attach public exposure of O to the audit, transport, continuation, provenance, boundary, and local naming rows A, H, C, P, G, N.

Finally Definition 525.7 transfers only this finite packet to the TasteGate handoff. If one triad component is missing, the handoff has no carrier coordinate licensing a public output row, so obligation closure cannot expose O. □

Theorem 525.50 (AuthorizedGeneratorRecursor root normalization boundary)**.** *Let* $R = (I, E, M, B, D, O, A, H,$ *be an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet. Any public output row* O *consumed by a normalization or closed-substitution consumer must first pass through the accepted signature rows* I, E, M*, branch coverage* B*, structural descent* D*, audit provenance* A, H, C, P*, the ground-compiler boundary* G*, and the sibling* ClosedTermSubstitutionBoundary$^{\uparrow}$ *rows named by chapter 527. The consumer then reaches* MetaCICNormalizationFrontier$^{\uparrow}$ *only as a finite boundary route, not as an output detached from the authorized generator packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootNormalizationBoundary`

Proof. Start from the local carrier and readiness rows. The carrier of Definition 525.1 displays $I, E, M, B, D, O, A, H, C, P, G, N$, while Definition 525.46 and Definition 525.49 force a public read of O through signature admission, branch coverage, structural descent, and audit provenance.

Next attach the compiler-facing handoff. Apply Definition 525.7 and Definition 525.47. These keep the public output inside the accepted recursor packet at G and exclude hidden generators, free induction principles, external axioms, and unaudited dependency routes.

Finally route the consumer through the sibling boundaries. The ClosedTermSubstitutionBoundary$^{\uparrow}$ chapter supplies the closed shift and substitution rows that a closed-substitution consumer must read, while the MetaCICNormalizationFrontier$^{\uparrow}$ chapter supplies the candidate and endpoint frontier rows for normalization consumers. Since neither sibling adds an off-carrier public output coordinate, the consumed output remains the displayed O reached through the finite authorized generator route. □

Theorem 525.51 (AuthorizedGeneratorRecursor consumer NameCert totality)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the complete consumer-facing NameCert route for normalization and closed-substitution consumers is the finite route*

$$I \longmapsto E \longmapsto M \longmapsto B \longmapsto D \longmapsto O \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto G \longmapsto N$$

together with the closed-boundary rows of $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *and the candidate-frontier rows of* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*. No consumer may replace this route by a detached recursor output, an unchecked substitution row, a blanket normalization theorem, an ambient termination principle, or a hidden generator channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorConsumerNameCertTotality`

Proof. First build the authorized generator side. Use Definition 525.50 to place every normalization or closed-substitution consumer after accepted signature, branch coverage, structural descent, audit provenance, and ground-compiler handoff rows.

Next use local consumer readiness. Definition 525.48 packages the rows of the accepted recursor packet as a downstream-ready NameCert surface, and Definition 525.14 blocks public routes outside the displayed carrier.

It remains to name the two downstream boundaries. $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ supplies the finite closed-term shift and substitution route, and $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ supplies the candidate-mediated normalization frontier. Combining these sibling surfaces with the authorized generator route yields the displayed consumer-facing NameCert totality statement and leaves no coordinate for the excluded replacements. □

Theorem 525.52 (AuthorizedGeneratorRecursor root consumer factorization)**.** *Every downstream closed-normal or generator-audit consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet factors through the finite carrier rows*

$$(I, E, M, B, D, O, A, H, C, P, G, N).$$

The consumer may read the accepted signature, emitted eliminator, motive, complete branch package, descent row, output name, audit row, transports, continuations, package provenance, boundary row, and local name, but it receives no free induction principle, hidden proof generator, external axiom, or unaudited dependency route.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootConsumerFactorizatio`

Proof. Begin with root NameCert consumer readiness from Definition 525.48. It already identifies the rows $I, E, M, B, D, O, A, H, C, P, G, N$ as the surface available to downstream closed-normal and generator-audit packets.

Use root output audit totality, Definition 525.46, to keep every public output read attached to the accepted signature, eliminator, motive, branch package, descent row, audit row, and structural rows.

Finally apply root boundary nonescape, Definition 525.47. The boundary row G exports only the accepted recursor call over the finite packet, so the downstream consumer has no off-packet generator authority. □

Theorem 525.53 (AuthorizedGeneratorRecursor root readiness exhaustion)**.** *The root readiness surface for* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *is exhausted by root output audit totality, root boundary nonescape, root NameCert consumer readiness, and the obligation triad. Any downstream root*

consumer that asks for a closed-normal or generator-audit handoff must cite those four rows and cannot demand an additional primitive outside the displayed packet.

Proof. Read the readiness theorem Definition 525.48. It packages the downstream surface for closed-normal and generator-audit consumers.

The output component is exactly Definition 525.46, and the boundary component is exactly Definition 525.47. Together they account for the public output row and the host-kernel boundary row.

It remains to account for the internal obligations. Apply Definition 525.49. Branch-package exhaustion, structural-descent locality, and audit-provenance factorization cover the remaining readiness checks, so no further primitive is available to a root consumer. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootReadinessExhaustion`

Theorem 525.54 (AuthorizedGeneratorRecursor root audit handoff lock)**.** *The* $\mathsf{OperationalMembrane}^{\uparrow}$ *and TasteGate handoff for an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet is locked to the same finite carrier*

$$(I, E, M, B, D, O, A, H, C, P, G, N).$$

The handoff refuses free induction, hidden proof generators, external axioms, and unaudited dependency routes; it can only expose the audited recursor output already covered by the root readiness surface.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAuditHandoffLock`

Proof. Start with the operational audit handoff Definition 525.12. It places the audit row A before public exposure of the output row and keeps operational provenance inside the same packet.

Next use Definition 525.52. Any downstream closed-normal or generator-audit consumer must read through the same finite carrier rows, so the handoff cannot introduce a second consumer route.

Finally apply Definition 525.53 together with Definition 525.7. The readiness surface has no remaining primitive outside the displayed packet, and the TasteGate handoff reads only the aligned finite carrier. Therefore free induction, hidden proof generators, external axioms, and unaudited dependency routes are refused at the handoff. □

Theorem 525.55 (AuthorizedGeneratorRecursor root consumer fact downstream coverage)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet, every accepted local downstream consumer fact whose handoff is closed-normal or generator-audit is exhausted by the displayed finite surface*

$$(I, E, M, B, D, O, A, H, C, P, G, N).$$

The constructor alternatives and recursive calls are covered by I, E, M, B, D*; the output, audit, transport, continuation, provenance, boundary, and local naming ledgers are covered by* O, A, H, C, P, G, N*; and any remaining readiness demand is covered by root output audit totality, root boundary nonescape, root NameCert consumer readiness, and the obligation triad. Hence an unlisted constructor, detached branch table, unchecked motive, non-structural recursion channel, hidden proof generator, external axiom channel, unaudited dependency route, off-packet substitution authority, or naming certificate outside the packet remains outside the BEDC packet exposed to that consumer.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootConsumerFactDownstrea`

Proof. Apply Definition 525.52. It forces every downstream closed-normal or generator-audit consumer of an accepted root packet to factor through the finite carrier rows $I, E, M, B, D, O, A, H, C, P, G, N$. Thus the consumer has no independent read surface beyond the accepted signature, eliminator, motive, branch package, descent row, output name, audit row, transports, continuations, package provenance, boundary row, and local name.

The finite carrier splits the possible consumer facts into the three displayed kinds. Branch alternatives and recursive-read coverage are read through I, E, M, B, D; output and audit-ledger facts are read through O, A, H, C, P, G, N; and boundary-facing refusal facts are read at G together with the same local name row N. Since the factorization theorem grants no free induction principle, hidden proof generator, external axiom, or unaudited dependency route, none of the excluded alternatives can be reintroduced as a second consumer fact.

It remains to check that no readiness demand is left over after this split. By Definition 525.53, any downstream root consumer asking for a closed-normal or generator-audit handoff must cite root output audit totality, root boundary nonescape, root NameCert consumer readiness, and the obligation triad, and cannot demand an additional primitive outside the displayed packet. These four rows are exactly the output-ledger, boundary-refusal, consumer-surface, and branch-descent-audit coverage rows named above. Therefore every accepted local consumer fact is exhausted by the finite alternatives, ledgers, or coverage rows already displayed, while all excluded alternatives remain outside the BEDC packet. □

Theorem 525.56 (AuthorizedGeneratorRecursor route-pack exactness)**.** *For every accepted* AuthorizedGeneratorRe *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every accepted recursor route factors through the finite packet rows displayed in Definition 525.1. The route may expose the accepted constructor signature, emitted eliminator, checked motive, branch package, structural descent, output name, audit provenance, componentwise hsame *transport, displayed* Cont *route,* Pkg *provenance,* GAP *boundary, and local* NameCert *row, but it cannot add a row outside that packet.*

Proof. Project the carrier of Definition 525.1. The displayed tuple is the whole route package, and each public recursor read is one of the listed rows.

Use the local NameCert surface of Definition 525.3. It identifies the same rows as the naming certificate for the authorized recursor output and requires a consumer to read O only through the packet.

Finally apply the audit and boundary results. The audit boundary of Definition 525.4 and the kernel acceptance scope of Definition 525.28 keep the host-kernel handoff at G on the displayed tuple. Since no carrier coordinate exists beyond $I, E, M, B, D, O, A, H, C, P, G, N$, route-pack exactness follows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRoutePackExactness`

Theorem 525.57 (AuthorizedGeneratorRecursor branch-budget exhaustion)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the branch package B and structural-descent row D exhaust every public branch read. A public consumer may read only constructor branches listed by I and authorized through E, M, B, D; it cannot import an unlisted constructor branch, a detached recursive call, or a second branch budget outside the displayed carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchBudgetExhaustion`

Proof. Start with branch exhaustion from Definition 525.2. The row B contains exactly the accepted branch surface for the constructors displayed by I, and downstream access reaches it only through the displayed continuation route.

Add structural locality from Definition 525.5 and Definition 525.13. These rows bind public recursive reads to the checked motive and structural-descent row, so every recursive branch read remains inside E, M, B, D.

The refusal side is supplied by Definition 525.14 and Definition 525.24. The carrier has no coordinate for an imported constructor branch, non-structural recursion channel, or hidden proof generator, so the displayed branch budget is exhausted. □

Theorem 525.58 (AuthorizedGeneratorRecursor consumer ledger cover)**.** *Every downstream* $\mathsf{RecursorGenerator}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, *and* TasteGate *consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet receives only the carrier, branch, descent, audit,* hsame, Cont, Pkg, GAP, *and local* NameCert *rows displayed by*

$$(I, E, M, B, D, O, A, H, C, P, G, N).$$

No such consumer receives an extra generator row, substitution authority, unlisted branch, unaudited dependency channel, or naming certificate outside the packet.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorConsumerLedgerCover`

Proof. The carrier side is fixed by Definition 525.56. It makes the finite tuple the exact route package available to any consumer.

The branch and descent side is fixed by Definition 525.57. Thus a $\mathsf{RecursorGenerator}^{\uparrow}$ consumer can read only the accepted signature, eliminator, motive, branch package, and structural-descent rows already present in the packet.

The audit and ledger side is fixed by Definition 525.48 and Definition 525.49. They attach the output, audit row, transports, continuation routes, package provenance, boundary row, and local certificate before downstream closed-normal or generator-audit consumers can cite the surface.

Finally the TasteGate handoff of Definition 525.7 reads only the same single carrier alignment witness. Since the ledger has no coordinate for substitution authority, an extra generator, or an outside naming certificate, the listed consumers receive exactly the displayed rows. □

Lemma 525.59 (AuthorizedGeneratorRecursor route-pack downstream coverage)**.** *Let c be an accepted local consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet with displayed route pack*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

Then the read made by c is exhausted by the finite alternatives already displayed in the packet. Its branch-facing part is one of the constructor, eliminator, motive, branch-package, or structural-descent rows I, E, M, B, D; its output-facing ledger part is one of O, A, H, C, P, G, N; and its consumer-facing coverage row is one of the carrier, branch, descent, audit, hsame, Cont, Pkg, GAP, *or local* NameCert *rows enumerated for the packet. Consequently an unlisted constructor alternative, detached recursive call, second branch budget, extra generator row, off-packet substitution authority, unaudited dependency channel, or outside naming certificate remains outside the BEDC packet and has no accepted local consumer route.*

Proof. Fix an accepted local consumer c of the displayed packet. By Definition 525.56, every accepted recursor route factors through the finite carrier of Definition 525.1. Thus the consumer-facing route of c has no coordinate except the displayed rows $I, E, M, B, D, O, A, H, C, P, G, N$.

Apply Definition 525.57 to the branch-facing part of the read. Any public branch read must use only the constructor branches listed by I and authorized through E, M, B, D. Hence the alternatives available to c on the branch side are exactly the displayed constructor, eliminator, motive, branch-package, and structural-descent rows, and there is no accepted coordinate for an unlisted constructor branch, detached recursive call, or second branch budget.

For the downstream ledger side, use Definition 525.58. It confines every downstream $\mathsf{RecursorGenerator}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, and $\mathsf{TasteGate}$ consumer to the carrier, branch, descent, audit, hsame, Cont, Pkg, GAP, and local $\mathsf{NameCert}$ rows displayed by the same tuple. Therefore the output-facing part of c can reach only O, A, H, C, P, G, N after the admitted branch and descent rows have been read.

Combining the route exactness, branch-budget exhaustion, and consumer-ledger cover leaves no remaining packet coordinate from which c could obtain an extra generator row, off-packet substitution authority, unaudited dependency channel, or outside naming certificate. All excluded alternatives are therefore refused by absence of a displayed BEDC row, and every accepted local consumer read is exhausted by the finite route pack. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRoutePackDownstreamCovera`

Theorem 525.60 (AuthorizedGeneratorRecursor root signature row totality)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, each public root read exposes the entire row family*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

as the local BHist *packet with componentwise* hsame *transport, displayed* Cont *routes, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *surface. No public root read may expose only a partial signature, branch, descent, audit, or output row while suppressing the remaining displayed rows.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootSignatureRowTotality`

Proof. Project the carrier of Definition 525.1. It names the finite BHist packet $R = (I, E, M, B, D, O, A, H, C, P, G,$ and gives no coordinate for a second root signature or a detached public route.

Use the local surface of Definition 525.3. The $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row reads the accepted signature, recursor name, motive, branch package, descent row, output name, audit provenance, transports, routes, package provenance, boundary, and local naming row as one public packet.

Branch and recursive-read coverage are supplied by Definition 525.2 and Definition 525.5. The audit and handoff side is supplied by Definition 525.6 and Definition 525.7. Together these results make every root read total over the displayed BHist, hsame, Cont, and $\mathsf{NameCert}$ rows. □

Theorem 525.61 (AuthorizedGeneratorRecursor root descent audit coupling)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the structural descent row D and audit row A must be transported together before the output row O is public. A public output exposure cannot retain D while dropping A, cannot retain A while dropping D, and cannot route O through an audit surface detached from the accepted structural-descent row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root descent audit coupling`

Proof. Start with the carrier and the structural locality theorem. The displayed packet of Definition 525.1 places D in the same tuple as $I, E, M, B, O, A, H, C, P, G, N$, while Definition 525.5

confines public recursive reads to the accepted signature, eliminator, motive, branch package, and descent row.

Next apply the audit boundary and audit-provenance factorization. Definition 525.4 requires public output to pass through the accepted signature, checked motive, complete branch package, structural descent, and audit provenance. Definition 525.6 keeps O attached to A, H, C, P, G, N before exposure.

Finally use branch exhaustion and the TasteGate handoff. By Definition 525.2, no branch read can escape the accepted constructor list; by Definition 525.7, the checked alignment witness consumes only the same finite carrier. Hence the descent row and audit row are coupled before O becomes public. □

Theorem 525.62 (AuthorizedGeneratorRecursor root output NameCert saturation)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, public output exposure is saturated by the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *surface over*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The exposure cannot add an unlisted constructor, an extra branch, a second recursion channel, an axiom row, or a hidden generator outside the displayed packet.

Proof. Use the carrier definition and the local NameCert surface. The carrier in Definition 525.1 is exactly the displayed tuple, and Definition 525.3 makes O public only through that tuple.

Branch and descent saturation follow from Definition 525.2 and Definition 525.5. These results rule out an unlisted constructor, an extra branch, or a recursive read whose descent is absent from D.

Audit saturation follows from Definition 525.4 and Definition 525.6. They keep O behind A, H, C, P, G, N and refuse an unaudited dependency channel. The TasteGate handoff of Definition 525.7 consumes only the same aligned carrier, so no axiom row or hidden generator can be introduced by the output exposure. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputNameCertSaturat`

Theorem 525.63 (AuthorizedGeneratorRecursor root signature downstream coverage)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, every accepted local consumer downstream of a public root signature read is exhausted by the finite root rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

More precisely, such a consumer can only use the signature, eliminator, motive, branch, and descent alternatives I, E, M, B, D*, the output and audit ledger rows* O, A, H, C, P, G, N*, or the coverage surface obtained by reading the displayed* BHist*, componentwise* hsame*,* Cont*, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows together. A request for an unlisted constructor, extra branch, second recursion channel, detached audit route, axiom row, hidden generator, or off-packet naming certificate remains outside the accepted* BEDC *packet.*

Proof. Fix an accepted packet and a public root signature read. By Definition 525.60, the read exposes the full row family

$$I, E, M, B, D, O, A, H, C, P, G, N$$

as one local BHist packet with componentwise hsame transport, displayed Cont routes, and the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ surface. The same theorem rules out a partial public root read:

no consumer can keep only a signature, branch, descent, audit, or output row while suppressing the remaining displayed rows.

The possible downstream uses therefore have only the displayed coordinates available. The root signature side supplies I, E, M, B, D as the accepted signature, emitted eliminator, checked motive, branch package, and structural-descent alternatives. The output side supplies O, A, H, C, P, G, N as the output, audit provenance, transport, continuation, package provenance, boundary, and local naming ledger. The coverage surface is not a further coordinate; it is the simultaneous exposure of the displayed BHist, hsame, Cont, and NameCert rows forced by the root signature totality theorem.

Apply Definition 525.62 to the output-facing part of the same packet. Public output exposure is saturated by the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ surface over exactly the displayed rows, and cannot add an unlisted constructor, an extra branch, a second recursion channel, an axiom row, or a hidden generator outside the packet. Combining this saturation with root signature row totality leaves no coordinate for a detached audit route or off-packet naming certificate either: any such request would require either a partial root read or an output exposure beyond the saturated local NameCert surface. Hence every accepted local downstream consumer is exhausted by the finite alternatives, ledger rows, and coverage rows displayed above, while every excluded alternative remains outside the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootSignatureDownstreamCo`

525.1 AuthorizedGeneratorRecursor displayed route surface

Definition 525.64 (AuthorizedGeneratorRecursor displayed route)**.** *An* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed route is a finite* BHist *packet*

$$R = (A, G, F, K, P, N)$$

whose rows are:

1. *A, the authorization row for the accepted generator signature;*
2. *G, the generator row naming the generator whose recursor is being read;*
3. *F, the recursion-fuel row bounding the displayed recursive descent;*
4. *K, the continuation route that exposes each branch, recursive call, and output row through displayed* Cont *data;*
5. *P, the package provenance over the accepted signature, generator row, fuel row, branch package, output route,* BHist, hsame, Cont, *and* NameCert *rows;*
6. *N, the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row for this displayed route.*

The route has no coordinate for an unrecorded host recursor, an unbounded recursion principle, an extra generator branch, or a detached output channel.

Theorem 525.65 (AuthorizedGeneratorRecursor ledger exhaustion)**.** *Every consumer read of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed route factors through the authorization row A, generator row G, recursion-fuel row F, continuation route K, package provenance P, and local naming row N. No consumer can import an unrecorded host recursor, bypass the displayed fuel row, or read a generator output detached from these rows.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor ledger exhaustion`

Proof. Project the displayed route of Definition 525.64. Its only public coordinates are A, G, F, K, P, N, and the route contains no second recursor channel.

The authorization and generator rows must be read first. They identify the accepted generator signature and the generator whose recursor output is public, so a consumer that lacks either row has no accepted root route to consume.

The recursion-fuel and continuation rows then exhaust recursive access. Every branch, recursive call, and output exposure is routed through F and K, with provenance and local naming supplied by P, N.

Therefore an unrecorded host recursor, an unbounded recursion principle, or a detached output channel would require a coordinate outside R, and no such coordinate is present. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorDisplayedRoute ledger exhaustion`

Theorem 525.66 (AuthorizedGeneratorRecursor NameCert route)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed route, the local* NameCert *obligation surface consists exactly of carrier exposure for* $R = (A, G, F, K, P, N)$*, classifier transport of these six rows by the displayed* hsame *data, route determinacy through* K*, ledger exhaustion through Definition 525.65, and non-escape from unrecorded host recursors or detached generator outputs.*

Proof. Carrier exposure is the finite route packet of Definition 525.64. It displays the authorization, generator, fuel, continuation, provenance, and naming rows as one BHist surface.

Classifier transport is componentwise: an accepted comparison preserves A, G, F, K, P, N through the stored hsame rows and replays the same public reads through the displayed Cont route K.

Route determinacy follows because K is the only continuation coordinate. Once A, G, F, P, N are fixed, every branch, recursive call, and output read must pass through that route.

Ledger exhaustion and non-escape are exactly the content of Definition 525.65. The route therefore packages the carrier, classifier, deterministic continuation, exhaustive ledger, and refusal of unrecorded host recursors into the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNameCertRoute`

Lemma 525.67 (AuthorizedGeneratorRecursor displayed route downstream coverage)**.** *Let* c *be an accepted local consumer of an* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed route. Then the consumer read is exhausted by the finite displayed-route packet*

$$R = (A, G, F, K, P, N)$$

of Definition 525.64. More precisely, c *factors through one of the following local alternatives:*

1. *the authorization row* A *for the accepted generator signature;*
2. *the generator row* G *naming the generator whose recursor is being read;*
3. *the recursion-fuel row* F *bounding the displayed recursive descent;*
4. *the continuation route* K *exposing each branch, recursive call, and output row through displayed* Cont *data;*
5. *the package provenance row* P *over the accepted signature, generator row, fuel row, branch package, output route,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
6. *the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N *for the displayed route.*

Consequently every request for an unrecorded host recursor, an unbounded recursion principle, an extra generator branch, or a detached output channel has no accepted coordinate in the $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed-route packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorDisplayedRoute downstream coverage`

Proof. Start with an accepted local consumer read of the displayed route. By Definition 525.65, every consumer read of an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ displayed route factors through the authorization row A, generator row G, recursion-fuel row F, continuation route K, package provenance row P, and local naming row N.

Project those rows through Definition 525.64. The displayed route is the finite BHist packet $R = (A, G, F, K, P, N)$, and the definition assigns each public coordinate a fixed role: A admits the accepted generator signature, G names the generator whose recursor is public, F bounds the recursive descent, K carries the displayed continuation data for branches, recursive calls, and output rows, P records the package provenance for the route data, and N supplies the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row. Thus the factorization supplied by ledger exhaustion lands in exactly the six alternatives listed in the statement.

It remains to exclude requests not listed among those alternatives. An unrecorded host recursor would require a second recursor coordinate, but Definition 525.64 states that the route has no coordinate for one and Definition 525.65 blocks importing one through a consumer read. An unbounded recursion principle would have to bypass F, while ledger exhaustion requires every recursive access to factor through the displayed fuel row. An extra generator branch would require a branch coordinate outside the continuation route K, but the displayed route exposes branches only through K. A detached output channel would require an output read not attached to A, G, F, K, P, N, while ledger exhaustion states that no consumer can read such an output. Hence every accepted local consumer is covered by the displayed finite rows, and every unlisted alternative remains outside the BEDC packet. □

Theorem 525.68 (AuthorizedGeneratorRecursor public route)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *displayed route*

$$R = (A, G, F, K, P, N),$$

the public finite export surface for generator-recursor consumers is exactly the displayed route of Definition 525.64, the ledger exhaustion of Definition 525.65, the NameCert route of Definition 525.66, and the downstream coverage of Definition 525.67. A consumer may read authorization, generator identity, recursion fuel, continuation data, provenance, and local naming from this surface, but it receives no unrecorded host recursor, unbounded recursion principle, extra generator branch, or detached output channel.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPublicRoute`

Proof. First expose the displayed route. The definition fixes the six public rows A, G, F, K, P, N and rules out any second recursor coordinate.

Next apply ledger exhaustion. Every consumer read factors through those six rows, with recursive access routed through the fuel and continuation rows.

Apply the NameCert route to package carrier exposure, classifier transport, route determinacy, ledger exhaustion, and non-escape as the local certificate surface.

Finally apply downstream coverage. It identifies the accepted local consumer alternatives with the same six rows and excludes every request outside them, so the public export surface is exactly the displayed route together with its ledger and NameCert route. □

Theorem 525.69 (AuthorizedGeneratorRecursor root branch-package coverage)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the branch-package surface exposes every constructor branch only through the accepted signature, emitted eliminator, checked motive, branch package, and structural-descent rows I, E, M, B, D. Any branch not listed by I and not carried by B is rejected before a downstream generator-audit or closed-normal consumer can read the output row O.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchPackageCoverage`

Proof. Project the carrier of Definition 525.1. The rows I, E, M, B, D are displayed before O, and the carrier has no coordinate for an extra constructor, missing branch, or detached branch table.

Use the local NameCert surface and the root row exposure results. By Definition 525.3, Definition 525.43, and Definition 525.60, the branch package is read as part of the same finite packet rather than through a hidden generator route.

Finally apply Definition 525.2, Definition 525.57, and Definition 525.49. They force branch coverage to stay on the constructor list displayed by I, the recursor row E, the checked motive M, the branch package B, and the descent row D. Thus an unlisted branch has no public route to O. □

Theorem 525.70 (AuthorizedGeneratorRecursor root motive descent stability)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet, structural descent through D preserves the checked motive row M before the output row O is exposed. A public recursive read cannot retain the descent row while replacing the motive, cannot read O from a motive-free descent surface, and cannot route the recursor through a branch package detached from M.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootMotiveDescentStabilit`

Proof. Start with the carrier and root descent exposure. The packet of Definition 525.1 places M and D in the same displayed tuple, while Definition 525.44 exposes the descent row on every root-facing recursive read.

The stability step is supplied by structural locality. Apply Definition 525.5, Definition 525.13, and Definition 525.61. These results keep the recursive read on the accepted signature, emitted eliminator, checked motive, branch package, descent row, and audit row together.

It remains to reach the output boundary. By Definition 525.46 and Definition 525.62, public exposure of O consumes the same packet. The TasteGate target named `BEDC.Derived.AuthorizedGeneratorRecurs` is used only as the aligned-carrier marker from Definition 525.7; the paper-side motive and descent obligations are carried by the cited root theorems. □

Theorem 525.71 (AuthorizedGeneratorRecursor root audit ledger exactness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet, public output reads factor through exactly the output, audit, transport, continuation, package, boundary, and local-name rows O, A, H, C, P, G, N after branch and descent admission. The packet exports no hidden generator, unaudited dependency route, external axiom channel, or naming certificate outside the displayed audit ledger.*

Proof. Read the audit ledger from the carrier

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The rows O, A, H, C, P, G, N are public only as coordinates of this packet, and Definition 525.4 places output exposure behind the accepted signature, motive, branch package, structural descent, and audit provenance.

Use factorization through the sibling root surfaces. Definition 525.6, Definition 525.45, Definition 525.52, and Definition 525.58 attach O to A, H, C, P, G, N before any downstream consumer can cite the recursor output.

Finally apply the refusal and readiness results. Definition 525.47, Definition 525.54, and Definition 525.53 exclude hidden generators, unaudited dependency routes, and off-packet naming. Therefore the public ledger is exact for the displayed audit route and exports nothing beyond it. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root audit ledger exactness`

Theorem 525.72 (AuthorizedGeneratorRecursor root branch audit coupling)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every public read of the output row O couples the branch package B to the audit row A through the same signature, eliminator, motive, descent, transport, continuation, package, boundary, and local-name rows. No downstream recursor or generator-audit consumer can read O from a branch package that is detached from the audit ledger.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchAuditCoupling`

Proof. Project the displayed carrier R. The branch row B and audit row A occur as coordinates of the same finite packet, with $I, E, M, D, O, H, C, P, G, N$ recording the surrounding admission, output, transport, route, provenance, boundary, and local naming rows.

Apply Definition 525.69. It forces every branch exposed to a consumer to pass through the accepted signature, emitted eliminator, checked motive, branch package, and structural-descent rows I, E, M, B, D before O is public.

Now apply Definition 525.71. That theorem attaches public output reads to O, A, H, C, P, G, N after branch and descent admission. The TasteGate marker `BEDC.Derived.AuthorizedGeneratorRecursorUp.tas` is used only as the carrier alignment cited by the chapter, not as a replacement for this branch-audit obligation. Thus every public branch read remains coupled to the audit ledger of R. □

Theorem 525.73 (AuthorizedGeneratorRecursor root descent output factorization)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

any output-facing recursive read factors through the checked motive row M and structural-descent row D before it reaches O, and then through A, H, C, P, G, N before any downstream closed-normal or generator-audit consumer can cite the output. The factorization has no hidden proof generator, non-structural recursion channel, external axiom row, or off-packet naming certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootDescentOutputFactoriz`

Proof. Read M, D, O as coordinates of the displayed packet R. The packet contains no second motive row, no detached descent row, and no output coordinate outside the same finite carrier.

Apply Definition 525.70. It keeps structural descent through D tied to the checked motive row M before output exposure, and it blocks a motive-free descent surface.

Then apply Definition 525.71. The output read continues through A, H, C, P, G, N after branch and descent admission, so the public consumer receives an audited output route rather than a hidden generator route. The TasteGate marker `BEDC.Derived.AuthorizedGeneratorRecursorUp.taste_gate` remains only the aligned-carrier witness; the paper-side descent and output factorization is carried by the cited root theorems. □

Theorem 525.74 (AuthorizedGeneratorRecursor root boundary ledger totality)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the output boundary ledger visible to downstream recursor, closed-normal, generator-audit, and TasteGate consumers is exhausted by $I, E, M, B, D, O, A, H, C, P, G, N$. The boundary row G and local name row N do not expose a hidden generator, unaudited dependency route, external axiom channel, host-kernel completeness claim, or naming certificate outside the displayed packet.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBoundaryLedgerTotalit`

Proof. Start from the full carrier R. The rows I, E, M, B, D supply branch and descent admission, O, A supply the output and audit ledger, and H, C, P, G, N supply transport, continuation, provenance, boundary, and local naming.

Use Definition 525.69 and Definition 525.70 to account for the admission side. Together they force branch coverage and descent stability to remain inside I, E, M, B, D before public output is read.

Use Definition 525.71 for the output side. It exhausts public output reads by O, A, H, C, P, G, N after branch and descent admission. The TasteGate marker `BEDC.Derived.AuthorizedGeneratorRecursorUp.tas` only aligns this finite carrier with the chapter handoff, so it does not supply an additional boundary or ledger row. Hence the visible boundary ledger is total for exactly the displayed packet. □

Theorem 525.75 (AuthorizedGeneratorRecursor root branch-audit downstream coverage)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every accepted local downstream consumer of the root branch-audit output is exhausted by the finite rows displayed in this carrier. More precisely, the consumer can read only a branch-admission alternative in I, E, M, B, D, an output-audit ledger row in O, A, H, C, P, G, N, or the boundary coverage row obtained by keeping these two surfaces coupled inside the same packet. Any candidate consumer that asks for a detached branch package, unaudited output route, hidden generator, external axiom channel, host-kernel completeness claim, or off-packet naming certificate remains outside the accepted BEDC *packet.*

Proof. Fix an accepted root packet and project the displayed carrier

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The branch side of any local consumer is finite because the exposed branch package is not a free generator surface. By Definition 525.72, every public read of O couples the branch package B to

the audit row A through the same signature, eliminator, motive, descent, transport, continuation, package, boundary, and local-name rows. Hence a local consumer cannot first read a branch alternative and then detach that alternative from the audit ledger of R.

The downstream side is total because the boundary ledger has no further public coordinates. By Definition 525.74, the output boundary visible to recursor, closed-normal, generator-audit, and TasteGate consumers is exhausted by

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Thus any accepted consumer that reaches the output must land either on the admission rows I, E, M, B, D before O, or on the audit, transport, continuation, package, boundary, and local-name rows O, A, H, C, P, G, N after O.

It remains only to exclude apparent alternatives. A detached branch package is rejected by the coupling theorem, since no downstream recursor or generator-audit consumer can read O from a branch package separated from A. An unaudited output route, hidden generator, external axiom channel, host-kernel completeness claim, or off-packet naming certificate is rejected by the boundary-ledger totality theorem, since none of those objects is a coordinate of the displayed carrier. Therefore the accepted local downstream coverage is exactly the finite branch alternatives, ledgers, and coverage rows already displayed in R, and every excluded alternative remains outside the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchAuditDownstrea`

Definition 525.76 (AuthorizedGeneratorRecursor root output boundary package)**.** *The root output boundary package for an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet is the finite* BHist *carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

of Definition 525.1, read with the following boundary roles. The rows I, E, M, B, D form the signature, eliminator, motive, branch, and structural-descent admission surface. The row O is the public output name. The rows A, H, C, P, G, N form the audit provenance, componentwise transport, continuation, package provenance, host-boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *surface. The package admits O only as a row of this displayed packet, after the branch package, descent locality, audit provenance, operational-membrane handoff, kernel-boundary exhaustion, and TasteGate carrier handoff have all been retained.*

Lean4 (definition) `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputBoundar`

Theorem 525.77 (AuthorizedGeneratorRecursor root branch output closed boundary)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, public output reads from the root output boundary package of Definition 525.76 occur only after the accepted signature row I, emitted eliminator row E, checked motive row M, branch-package row B, and structural-descent row D have been retained. No public read of O can detach the output from the constructor signature, eliminator, motive, branch package, or structural descent rows.*

Proof. Project the carrier named by Definition 525.1. The package of Definition 525.76 uses the same displayed tuple

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

Branch coverage is forced before any output read. Apply Definition 525.2; the row B contains exactly the branches associated with the accepted constructors displayed by I and the eliminator named by E.

Descent locality then binds the recursive read to the motive and descent rows. By Definition 525.5, a public recursive read cannot pass through a generator route detached from M and D.

Finally use Definition 525.7. The TasteGate handoff consumes only the same finite carrier, so it cannot supply an output route that bypasses I, E, M, B, D. Therefore public access to O is closed under the displayed branch and descent boundary. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root branch output closed boundary`

Theorem 525.78 (AuthorizedGeneratorRecursor root audit membrane source refusal)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the audit row A of the root output boundary package consumes* $\mathsf{OperationalMembrane}^{\uparrow}$ *provenance, source-axiom refusal, and dependency refusal before the output row O is public. Thus O cannot be read as a public output through an audit route that omits operational provenance, source refusal, or transitive dependency refusal.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAuditMembraneSourceRe`

Proof. Begin with the audit side of Definition 525.76. The rows O, A, H, C, P, G, N are retained together, and A is the row that records the build check, source-axiom refusal, dependency refusal, and operational-membrane provenance.

Apply Definition 525.6. It prevents O from becoming public apart from A, H, C, P, G, N, so the output row cannot be read as an unaudited recursor result.

Now apply Definition 525.12. The audit row must consume the $\mathsf{OperationalMembrane}^{\uparrow}$ provenance before public exposure of O, and the same row carries source-axiom and dependency refusal for the packet.

The carrier alignment of Definition 525.7 supplies no substitute audit channel. Hence a public output read must pass through operational provenance, source refusal, and dependency refusal before O leaves the package boundary. □

Theorem 525.79 (AuthorizedGeneratorRecursor root host-boundary NameCert exhaustion)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the host-boundary row G of the root output boundary package is exhausted by the displayed* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *packet. At G the packet cannot export a free induction principle, hidden branch source, external axiom, or proof generator outside*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootHostBoundaryNameCertP`

Proof. Read the host-boundary row from Definition 525.76. The row G is not a second generator surface; it is one coordinate of the same finite carrier displayed in Definition 525.1.

Branch and descent escape are already excluded inside the packet. Use Definition 525.2 and Definition 525.5 to keep every branch source and recursive call on I, E, M, B, D.

Audit escape is excluded before the boundary row becomes useful to a consumer. By Definition 525.6 and Definition 525.12, public output remains attached to the audit provenance, operational membrane, source refusal, dependency refusal, transport, continuation, package, and local name rows.

Finally apply Definition 525.9 together with Definition 525.7. The boundary row is exhausted by the displayed NameCert packet and the TasteGate carrier handoff adds no free induction principle, hidden branch source, external axiom, or proof generator. □

Corollary 525.80 (AuthorizedGeneratorRecursor root output boundary strict obstruction)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, any displayed failure, refusal, separation, or countermodel row attached to the root output boundary package of Definition 525.76 is a strict local obstruction. It blocks only the named converse or reuse route that would read* O *without the displayed audit and boundary rows. It does not state a global impossibility for recursors, does not become a host theorem about all generators, and does not supply hidden external validation. Equivalently, every consumer-facing use of the obstruction factors through the finite rows*

$$I, E, M, B, D, O, A, H, C, P, G, N,$$

and every unlisted coordinate is rejected as outside the local $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputBoundaryStrict(`

Proof. Start from the package definition. By Definition 525.76, the root output boundary is the single finite BHist carrier

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

The row O is public only as the output row of this carrier. The rows I, E, M, B, D supply the accepted signature, emitted eliminator, checked motive, branch package, and structural descent surface, while A, H, C, P, G, N supply audit provenance, componentwise transport, continuation, package provenance, host boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ data. Thus a consumer-facing use cannot begin from an isolated output token; it must enter through one of the displayed rows.

Apply Definition 525.78. The only obstruction supplied by the audit side is the refusal of an output read whose audit route omits $\mathsf{OperationalMembrane}^{\uparrow}$ provenance, source-axiom refusal, or transitive dependency refusal. This is a local refusal at A and its attached output ledger O, A, H, C, P, G, N. It blocks the converse route that tries to reuse O as unaudited output, but it does not assert that no accepted recursor output exists and does not certify any fact outside the displayed packet.

Apply Definition 525.79. At the host-boundary row G, the packet exports no free induction principle, hidden branch source, external axiom, or proof generator outside

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Therefore the boundary obstruction blocks only the attempted reuse of G as a host-side generator or external validation channel. It cannot be reinterpreted as a host theorem about all generator mechanisms, because the cited theorem exhausts G by the same local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ carrier.

It remains to exclude unlisted consumer coordinates. If a consumer-facing use were global, host-theoretic, or externally validated, it would need a coordinate not among the admission rows I, E, M, B, D, the output and audit rows O, A, H, C, P, or the boundary and name rows G, N. The package definition supplies no such coordinate; the audit refusal eliminates unaudited output reuse; and the host-boundary exhaustion eliminates free host proof production at G. Hence each displayed failure, refusal, separation, or countermodel row is strict in exactly the local sense: it obstructs the named converse or reuse route inside the root output boundary package, while every broader reading remains outside the BEDC packet. □

Theorem 525.81 (AuthorizedGeneratorRecursor branch output closure row)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet, the public branch output row remains behind the displayed carrier rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

In particular, a downstream public read of O *cannot detach the recursor output from the accepted signature row* I*, emitted eliminator row* E*, checked motive row* M*, branch package row* B*, structural-descent row* D*, audit row* A*, transport row* H*, continuation row* C*, package row* P*, boundary row* G*, or local* NameCert *row* N*.*

Proof. Project the root output boundary package of Definition 525.76. It presents O only as one coordinate of the displayed packet, not as a standalone generator result.

The branch side is fixed by Definition 525.77. It keeps public output reads behind I, E, M, B, D and refuses an output route detached from the constructor signature, emitted eliminator, checked motive, complete branch package, and structural descent.

The audit side is fixed by Definition 525.78 and Definition 525.79. These rows retain A, H, C, P, G, N before the boundary can expose O to a consumer.

Finally cite Definition 525.7 only as carrier alignment. The TasteGate handoff reads the same finite packet and supplies no independent branch output channel, so the public branch output remains closed behind exactly the displayed rows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchOutputClosureRow`

Theorem 525.82 (AuthorizedGeneratorRecursor audit provenance row)**.** *For every accepted* AuthorizedGeneratorR *root packet, the rows* A, H, C, P, G, N *are retained before public output exposure. The audit provenance row, componentwise transport row, continuation route row, package provenance row, host-boundary row, and local* NameCert *row therefore form the required downstream provenance package for any public read of* O*.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor audit provenance row`

Proof. Begin with audit provenance factorization, Definition 525.6. It states that O becomes public only after the audit row A, transports H, continuation routes C, package row P, boundary row G, and local name row N are retained.

The root package strengthens this retention at the output boundary. By Definition 525.78, A carries operational provenance, source refusal, and dependency refusal before O leaves the package boundary.

The boundary and local naming side is supplied by Definition 525.79. The row G is exhausted by the same displayed $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ packet and cannot export a hidden generator or off-packet certificate.

Use Definition 525.7 only to identify the carrier consumed by the TasteGate handoff. Since that handoff adds no theorem row and no substitute audit route, A, H, C, P, G, N remain the provenance package required before public exposure of O. □

Theorem 525.83 (AuthorizedGeneratorRecursor closed substitution consumer row)**.** *Every* ClosedTermSubstitutio *consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet reaches the recursor output only through the displayed branch, descent, audit, route, package, and* NameCert *rows. Such a consumer has no direct substitution authority over* O *outside the carrier*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorClosedSubstitutionConsume

Proof. Start from Definition 525.58. It already includes $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ among the downstream consumers that receive only the carrier, branch, descent, audit, transport, continuation, package, boundary, and local certificate rows displayed by the packet.

The branch and descent route is closed by Definition 525.81. A substitution-boundary consumer therefore reaches O only after I, E, M, B, D have been retained with the output boundary.

The audit and provenance route is closed by Definition 525.82. It keeps A, H, C, P, G, N attached before the public output can be consumed downstream.

Finally apply Definition 525.52. Downstream consumers factor through the finite carrier and receive no free induction principle, hidden proof generator, external axiom, unaudited dependency route, or off-packet substitution channel. Hence every closed-substitution consumer reaches the recursor output only through the displayed branch, descent, audit, route, package, and local NameCert rows. □

Theorem 525.84 (AuthorizedGeneratorRecursor output readiness)**.** *For every accepted* AuthorizedGeneratorRecurs *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the output row O is ready for downstream closed-substitution and generator-audit consumers only after the accepted signature I, emitted recursor E, checked motive M, branch package B, structural-descent row D, audit row A, componentwise hsame *row H, displayed* Cont *route C,* Pkg *provenance P, host-boundary row G, and local* NameCert *row N have all been retained in the same packet.*

Proof. Project the carrier of Definition 525.1. It exposes O only as a coordinate of the finite tuple R, not as a detached generator result.

Use the root output boundary package and root boundary ledger totality. The packet of Definition 525.76 and the theorem Definition 525.74 keep O tied to the branch, descent, audit, transport, continuation, provenance, boundary, and naming rows.

Finally compare with the closed-term substitution boundary of chapter 527 and the recursor generator of chapter 534. They may consume the output surface only through the displayed BHist, hsame, Cont, Pkg, and NameCert packet, so readiness is exactly the retained carrier R. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier output readiness

Theorem 525.85 (AuthorizedGeneratorRecursor branch audit ledger)**.** *For every accepted* AuthorizedGeneratorRec *packet, the branch package B and audit row A form one branch-audit ledger over the same* BHist *carrier. A public branch read that reaches O must pass through I, E, M, B, D and then through A, H, C, P, G, N; it cannot cite the closed-substitution boundary, recursor generator, or authorization row through a branch table detached from the audit ledger.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchAuditLedger

Proof. Start with the root branch audit coupling theorem. Theorem 525.72 already couples B and A through the same displayed carrier before O is public.

Read the authorization side next. The root authorization obligations and displayed route surface keep accepted signatures, generator rows, branch packages, and local naming inside the same $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ packet.

The closed-substitution handoff is therefore ledgered rather than ambient. Any route from a branch to a closed-substitution consumer must replay the same Cont row C and provenance row P while preserving componentwise hsame through H. A detached branch table has no carrier coordinate from which to export O. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier branch audit ledger`

Theorem 525.86 (AuthorizedGeneratorRecursor closed substitution handoff)**.** *Every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet hands its output row O to a* ClosedSubstitutionBoundary$^{\uparrow}$, ClosedTermSubstitution$^{\uparrow}$, *or generator-audit consumer only through the displayed finite route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The handoff preserves the closed-term substitution boundary, recursor generator, and authorization rows as finite BHist, hsame, Cont, Pkg, *and* NameCert *data, and it exports no unrestricted substitution calculus, subject-reduction theorem, proof generator, external axiom channel, host equality, or off-packet recursor certificate.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier closed substitution handoff`

Proof. First use output readiness. The output row is not public until all rows of R have been retained, so a closed-substitution consumer begins from the same accepted carrier as any generator-audit consumer.

Next use the branch audit ledger. It ties the branch package and audit row together before the handoff, preventing a consumer from reading branch output apart from authorization, motive checking, structural descent, and audit provenance.

Finally apply root host-boundary NameCert exhaustion and the closed-substitution boundary sibling. The boundary row G exports only the displayed NameCert packet, while ClosedSubstitutionBoundary$^{\uparrow}$ and ClosedTermSubstitution$^{\uparrow}$ consume closed substitution as finite boundary rows rather than as a general substitution theorem. Hence the handoff remains inside $I, E, M, B, D, O, A, H, C, P, G, N$ and none of the excluded external principles is exported. □

Lemma 525.87 (AuthorizedGeneratorRecursor output readiness bridge determinacy)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

any downstream consumer that reads the output row O as a route, transport, readback, or closed-substitution handoff factors through the same finite rows

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route part is I, E, M, B, D, O, A, the transport and readback part is H, C, P, and the boundary-certificate part is G, N. In particular, there is no consumer-facing bridge coordinate for an unlisted branch table, detached transport, off-packet readback, ambient substitution calculus, hidden generator, host equality, external axiom channel, or second NameCert *row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputReadinessBridgeDete`

Proof. Start with output readiness. By Definition 525.84, the row O is public for downstream closed-substitution and generator-audit consumers only after the accepted signature, emitted recursor, motive, branch package, descent row, audit row, componentwise transport, continuation route, package provenance, host-boundary row, and local naming row have all been retained in the same packet. Thus a consumer-facing read of O already begins from the displayed carrier R, not from an independent output surface.

Apply the branch audit ledger to the route side. Definition 525.85 says that any public branch read reaching O must pass through I, E, M, B, D and then through A, H, C, P, G, N. This makes the bridge deterministic on branch routes: the consumer may cite the branch package only together with the audit ledger and the later transport, continuation, provenance, boundary, and naming rows, so a detached branch table has no route to the closed-substitution boundary, recursor generator, or authorization row.

Now apply the closed-substitution handoff. Definition 525.86 sends every closed-substitution, closed-term substitution, and generator-audit consumer through the displayed finite route $I, E, M, B, D, O, A, H, C,$ It also states that the handoff preserves the boundary, recursor-generator, and authorization data only as finite BHist, hsame, Cont, Pkg, and NameCert rows. Hence the transport and readback surfaces are exactly H, C, P attached to O, A, and the boundary certificate surface is exactly G, N.

The cited rows exhaust the consumer-facing bridge. A displayed route is covered by the branch-audit prefix and audit suffix, a transport or readback is covered by the hsame, Cont, and Pkg rows, and a closed-substitution handoff is covered by the boundary and local NameCert rows of the same packet. Since the carrier of Definition 525.1 displays no coordinate outside R, an unlisted bridge coordinate has no factor through which a consumer can read O. □

Corollary 525.88 (AuthorizedGeneratorRecursor output readiness determinacy)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, and fix the displayed rows of R. Any accepted downstream use of the output readiness surface that reads O as a finite route, packet-local normal-form readiness claim, readback, closed-substitution handoff, or exactness decision must use the route, transport, readback, boundary, and local certificate already named by those rows. The route side is fixed by I, E, M, B, D, O, A, the transport and readback side is fixed by H, C, P, and the boundary-certificate side is fixed by G, N. Hence two accepted consumers over the fixed rows determine the same finite route and the same packet-local readiness or exactness decision named by the chapter; there is no alternate hidden coordinate, detached branch table, off-packet readback, ambient substitution calculus, hidden generator, host equality, external axiom channel, second* NameCert *row, or duplicate classifier source.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputReadinessDeterminac`

Proof. Apply Definition 525.87. It says that every downstream consumer reading O as a route, transport, readback, or closed-substitution handoff factors through the same finite rows

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

It also separates the rows by role: the route part is I, E, M, B, D, O, A, the transport and readback part is H, C, P, and the boundary-certificate part is G, N. Thus a consumer-facing normal-form readiness claim or exactness decision has no independent surface; it is a packet-local reading of the displayed route, transport, readback, and boundary rows attached to O.

Use Definition 525.85 on the route side. Any public branch read that reaches O must pass through I, E, M, B, D and then through A, H, C, P, G, N, and it cannot cite a branch table detached from the audit ledger. Therefore the branch package cannot supply a second source for the same output readiness decision after the rows of R have been fixed.

The remaining possible alternatives are exactly the interfaces excluded by the bridge determinacy lemma: an unlisted branch table, detached transport, off-packet readback, ambient substitution calculus, hidden generator, host equality, external axiom channel, or second NameCert row. A duplicate classifier source would have to enter through one of those excluded interfaces or through an additional carrier coordinate outside R. Since the fixed carrier displays no such coordinate, every accepted consumer use factors through the same listed rows and therefore selects the same chapter-named route, readiness readback, or exactness decision. □

Theorem 525.89 (AuthorizedGeneratorRecursor route totality lock)**.** *For every accepted* AuthorizedGeneratorRecu *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every accepted recursor route factors through exactly the displayed signature row I, eliminator row E, motive row M, branch package B, descent row D, output row O, audit row A, componentwise hsame *transport H, displayed* Cont *route C,* Pkg *provenance P, closed host-boundary row G, and local* NameCert *row N. No accepted route has a second eliminator, branch table, descent proof, output channel, audit packet, transport row, continuation, provenance row, or naming certificate outside this finite surface.*

Proof. Start from the displayed NameCert route, Definition 525.66. It identifies the public recursor route with the carrier rows $I, E, M, B, D, O, A, H, C, P, G, N$ and gives no detached route coordinate.

Apply the closed-substitution handoff, Definition 525.86. Any consumer that reaches the generated output must still pass through the same output, audit, transport, continuation, provenance, boundary, and naming rows.

Finally use root host-boundary NameCert exhaustion, Definition 525.79. The boundary row G exports only the local certificate surface of the accepted packet, so the route is total exactly over the displayed rows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier route totality lock`

Theorem 525.90 (AuthorizedGeneratorRecursor closed substitution budget lock)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the closed-substitution handoff consumes ClosedTermSubstitutionBoundary$^{\uparrow}$ *only through the displayed finite boundary rows G, H, C, P, N together with the authorized generator rows I, E, M, B, D, O, A. The handoff cannot import an unrestricted substitution calculus, subject-reduction theorem, semantic evaluator state, proof generator, host equality, or off-packet certificate as a boundary budget.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier closed substitution budget lock`

Proof. First invoke Definition 525.86. It places every ClosedSubstitutionBoundary$^{\uparrow}$ or ClosedTermSubstitution$^{\uparrow}$ consumer on the finite route $I, E, M, B, D, O, A, H, C, P, G, N$.

Use Definition 525.89 to lock that route before the boundary is consumed. The accepted recursor route has no auxiliary boundary channel beyond G and the visible hsame, Cont, Pkg, and NameCert rows.

Root host-boundary NameCert exhaustion then prevents any boundary budget from being read as host equality, subject reduction, or evaluator state. The consumed budget is precisely the displayed finite boundary surface. □

Theorem 525.91 (AuthorizedGeneratorRecursor output route audit cover)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every generated output row O is covered by the audit and provenance rows A, H, C, P, G, N before any downstream consumer reads it. An output route cannot bypass audit, provenance, closed-boundary naming, or componentwise transport, and it cannot import host equality, hidden evaluator state, subject reduction, an unrestricted substitution theorem, or an off-packet generator certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier output route audit cover`

Proof. Begin with the route totality lock. It keeps every accepted recursor route inside $I, E, M, B, D, O, A, H, C, P,$ so an output read begins with the packet's own audit and provenance coordinates.

Apply the closed-substitution handoff. It shows that output consumers, including closed-substitution and generator-audit consumers, receive O only through the displayed handoff route.

Finally use root host-boundary NameCert exhaustion. The boundary row exports no host equality, hidden evaluator state, subject-reduction theorem, or off-packet certificate, so the audit cover is exactly the packet-local A, H, C, P, G, N surface. □

Lemma 525.92 (AuthorizedGeneratorRecursor route totality local downstream coverage)**.** *Let c be an accepted local downstream consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N).$$

Then the consumer-facing read made by c is exhausted by the finite route, ledger, and coverage rows already displayed in R. More precisely, c either reads a route alternative among the accepted signature row I, eliminator row E, motive row M, branch package B, descent row D, output row O, and audit row A, or it reads the downstream output ledger covered by O, A, H, C, P, G, N, where H is the componentwise hsame *transport, C is the displayed* Cont *route, P is the* Pkg *provenance, G is the closed host-boundary row, and N is the local* NameCert *row. Consequently a second eliminator, detached branch table, off-route descent proof, second output channel, unaudited output read, hidden audit packet, detached transport row, private continuation, off-packet provenance row, host equality, hidden evaluator state, unrestricted substitution theorem, subject-reduction theorem, off-packet generator certificate, or outside naming certificate has no accepted local consumer coordinate in the* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet.*

Proof. Fix an accepted local downstream consumer c of the displayed carrier R. By Definition 525.89, every accepted recursor route factors through exactly the displayed rows

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

That theorem also rules out a second eliminator, branch table, descent proof, output channel, audit packet, transport row, continuation, provenance row, or naming certificate outside this finite surface. Thus the initial consumer-facing route of c has no coordinate except the rows listed in R.

Separate the read according to whether it reaches the generated output row. If c reads a route coordinate before output exposure, the totality lock already places it among the route alternatives I, E, M, B, D, O, A together with the attached transport, continuation, provenance, boundary, and naming rows. If c reads the generated output row O, then Definition 525.91 applies. It says that every generated output row is covered by the audit and provenance rows A, H, C, P, G, N before any downstream consumer reads it, and that the output route cannot bypass audit, provenance, closed-boundary naming, or componentwise transport.

The two cited theorems exhaust the downstream cases. The totality lock supplies the finite route surface and excludes any unlisted route coordinate. The audit-cover theorem supplies the output-facing ledger and keeps downstream output exposure behind A, H, C, P, G, N. Hence an accepted local consumer can only land on the displayed route alternatives, the displayed output ledger, or the displayed coverage rows of the same packet.

It remains to check the excluded alternatives. A second eliminator, detached branch table, off-route descent proof, second output channel, hidden audit packet, detached transport row, private continuation, off-packet provenance row, or outside naming certificate would be a route coordinate outside R, contradicting Definition 525.89. An unaudited output read, host equality, hidden evaluator state, unrestricted substitution theorem, subject-reduction theorem, or off-packet generator certificate would be an output-facing bypass of the audit and boundary surface, contradicting Definition 525.91. Therefore every accepted local downstream consumer is covered by the finite rows already enumerated in R, and every excluded alternative remains outside the accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier route totality local downstream coverage`

Theorem 525.93 (AuthorizedGeneratorRecursor route authorization)**.** *For every accepted* AuthorizedGeneratorRec *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the public recursor route exposes only the authorized generator rows carried by I, E, M, B, D, O, A, the displayed descent routes in D, the componentwise hsame *transport in H, the replayed* Cont *route in C,* Pkg *provenance in P, and the local* NameCert *row N. No consumer receives a generator row outside this carrier or a route around the displayed descent and audit rows.*

Proof. Project the carrier of Definition 525.1. The accepted signature, emitted recursor, motive, branch package, descent row, output row, and audit row are the only generator-facing coordinates.

Use the displayed route surface and root authorization obligations. They force any public read of the recursor through the same BHist packet, with D supplying descent, H transporting presentations, C replaying the continuation route, P recording provenance, and N recording the local certificate.

Therefore an exposed generator row is authorized exactly when it is one of the displayed rows of R. A route outside $I, E, M, B, D, O, A, H, C, P, G, N$ would require a carrier coordinate absent from the accepted packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier route authorization`

Theorem 525.94 (AuthorizedGeneratorRecursor output ledger exactness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, each generated output row O is read through the same finite authorized route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The output ledger has no detached generator result, off-route branch output, hidden structural descent certificate, or provenance row outside the displayed BHist, hsame, Cont, Pkg, *and* NameCert *surface.*

Proof. Begin with route authorization. It identifies the only public recursor route and fixes the generator-facing rows before any output can be consumed.

Apply output readiness and the branch audit ledger. Readiness keeps O unavailable until all rows of R are present, while the branch audit ledger ties B, D, O, A to the same audit surface.

The exact output ledger is therefore the displayed route itself. Since O is a coordinate of R and not an external result object, every output read factors through the same transport, continuation, provenance, and local naming rows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputLedgerExactness`

Theorem 525.95 (AuthorizedGeneratorRecursor substitution handoff nonescape)**.** *Every* ClosedSubstitutionBound *or* ClosedTermSubstitution$^{\uparrow}$ *consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet receives the authorized finite route*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

and no hidden recursor, host recursion principle, unledgered generator, unrestricted substitution calculus, proof generator, external axiom channel, host equality, or off-packet certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorSubstitutionHandoffNones`

Proof. Start from output ledger exactness. The generated output row is already tied to the authorized route, so a closed-substitution consumer begins with the same packet rather than with a detached generator product.

Use the closed-substitution handoff. It routes the handoff through the closed-boundary row G and the same BHist, hsame, Cont, Pkg, and NameCert coordinates used by the recursor packet.

Finally apply the root output boundary package. That boundary admits only the displayed handoff route, so there is no coordinate from which to export a hidden recursor, a host recursion principle, or an unledgered generator. □

Corollary 525.96 (AuthorizedGeneratorRecursor route-authorization obstruction locality)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet be read by a* ClosedSubstitutionBoundary$^{\uparrow}$ *or* ClosedTermSubstitutio *consumer through the displayed route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

If a displayed failure, refusal, separation, or countermodel row is cited against a converse or reuse route for this packet, then the citation is a strict local obstruction to that named route only. It blocks exactly the request that tries to leave the displayed packet by importing a hidden recursor, host recursion principle, unledgered generator, unrestricted substitution calculus, proof generator, external axiom channel, host equality, or off-packet certificate. It cannot be reinterpreted as a global impossibility theorem for recursors, as a host theorem about substitution systems, or as hidden external validation for any route not displayed in the packet.

Proof. By Definition 525.93, the public recursor route exposes only the carrier rows $I, E, M, B, D, O, A, H, C, P, G, N$. The generator-facing rows are I, E, M, B, D, O, A; the remaining public route data are the displayed descent route in D, componentwise hsame transport in H, the replayed Cont route in C, Pkg provenance in P, the closed-boundary row G, and the local NameCert row N.

By Definition 525.95, every ClosedSubstitutionBoundary$^{\uparrow}$ or ClosedTermSubstitution$^{\uparrow}$ consumer receives that same finite route and receives no hidden recursor, host recursion principle, unledgered generator, unrestricted substitution calculus, proof generator, external axiom channel, host equality, or off-packet certificate. Hence any consumer-facing use of a displayed obstruction row must factor through one of the listed coordinates: source and branch data through I, E, M, B, D, O, A, presentation transport through H, continuation replay through C, provenance through P, boundary handoff through G, and local naming through N.

Now suppose the cited obstruction is read as more than a block on the named converse or reuse route. A global impossibility reading would require a coordinate ranging over recursors outside the accepted packet; a host-theorem reading would require a host recursion, equality, or substitution principle; and a hidden-validation reading would require an external certificate not carried by N or the displayed provenance row P. Each required coordinate is absent from the route authorized by Definition 525.93 and is expressly excluded from the substitution handoff by Definition 525.95. Therefore the obstruction remains strict and local: it rejects the unlisted converse or reuse request and asserts nothing beyond the displayed packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRouteAuthorizationObstruc`

Theorem 525.97 (AuthorizedGeneratorRecursor terminal consumer readiness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every terminal normalization or closed-substitution consumer may read the recursor output only after the finite packet of Definition 525.1, the route authorization of Definition 525.93, the output ledger exactness of Definition 525.94, and the substitution handoff nonescape of Definition 525.95 have all been displayed. Such a consumer receives O as a row of R, with branch, descent, audit, boundary, and local naming data still attached, and not as a detached recursor theorem.

Proof. Project the finite BHist packet from Definition 525.1. The rows $I, E, M, B, D, O, A, H, C, P, G, N$ are the only displayed coordinates, so the terminal consumer starts from the packet rather than from an ambient generator principle.

Apply Definition 525.93 to route the public output through accepted signature, emitted recursor, motive, branch package, structural descent, audit, transport, continuation, provenance, boundary, and local NameCert rows. Then apply Definition 525.94 to keep O in the same ledger.

Finally use Definition 525.95. The handoff to the sibling boundaries of chapter 527 and chapter 892 is therefore a displayed BEDC row read, not a hidden induction source, unchecked generator, detached normalization proof, or external proof-producing channel. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalConsumerReadiness`

Theorem 525.98 (AuthorizedGeneratorRecursor boundary factorization)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

any public route from the recursor output O into a ClosedTermSubstitutionBoundary$^{\uparrow}$ *consumer or a* MetaCICNormalizationFrontier$^{\uparrow}$ *consumer factors through the displayed rows B, D, A, H, C, P, G, N after I, E, M have admitted the generator surface. In particular, the boundary route is the carrier route certified by Definition 525.1, Definition 525.93, Definition 525.94, and Definition 525.95.*

Proof. Begin with the route authorization theorem. It fixes the public recursor route at the rows of R, so a consumer path to O must pass through the branch row B, the descent row D, and the audit row A before any boundary read is available.

Use output ledger exactness to attach H, C, P, G, N to the same route. The componentwise transport, continuation replay, package provenance, boundary row, and local certificate are not auxiliary witnesses outside the packet; they are the boundary-facing coordinates of the packet itself.

The closed-term and normalization-frontier sibling chapters can therefore consume only this factored carrier route. A route that skipped B, D, A, H, C, P, G, N would need a boundary coordinate outside R, but Definition 525.1 displays no such coordinate. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBoundaryFactorization`

Theorem 525.99 (AuthorizedGeneratorRecursor normalization consumer nonescape)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet with carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

a MetaCICNormalizationFrontier$^{\uparrow}$ *consumer receives the recursor-generated output only as the authorized finite route already consumable by* ClosedTermSubstitutionBoundary$^{\uparrow}$*. The normalization consumer cannot reinterpret O as hidden induction, an unchecked generator, a detached normalization theorem, a subject-reduction source, a host equality principle, or an external proof-producing source.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNormalizationConsumerNone`

Proof. First read the route through Definition 525.97. Terminal readiness keeps the normalization consumer behind the same displayed finite packet as the closed-substitution consumer.

Next apply Definition 525.98. The route into chapter 892 must factor through the branch, descent, audit, transport, continuation, provenance, boundary, and local naming rows already exposed to chapter 527.

It remains to exclude non-displayed sources. Since Definition 525.95 exports no hidden recursor, unrestricted substitution calculus, proof generator, external axiom channel, host equality, or off-packet certificate, the normalization consumer has no coordinate from which to acquire hidden induction, an unchecked generator, detached normalization, subject reduction, or an external proof-producing source. □

Corollary 525.100 (AuthorizedGeneratorRecursor terminal consumer strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse, reuse, separation, or readback route for the terminal consumer surface of R. Then the cited row is a strict local obstruction. It blocks only the named route whose consumer-facing use tries to read the terminal output O, normalization readiness, or closed-substitution handoff away from the displayed packet rows*

$I, E, M, B, D, O, A, H, C, P, G, N$. Every admitted consumer-facing use factors through the packet carrier, route authorization, output-ledger exactness, substitution handoff, boundary factorization, and normalization-consumer nonescape rows already displayed for R. The same cited row cannot be reinterpreted as a global impossibility theorem for recursors, a host theorem about normalization or substitution, hidden external validation of the failed route, or an obstruction to any consumer route that still factors through the displayed rows of R.

Proof. Start with terminal consumer readiness, Definition 525.97. A terminal normalization or closed-substitution consumer may read the recursor output only after the finite carrier of Definition 525.1, route authorization, output ledger exactness, and substitution handoff nonescape have all been displayed. Thus the consumer receives O as a row of R, with branch, descent, audit, boundary, and local naming data still attached, and not as a detached recursor theorem.

Apply normalization consumer nonescape, Definition 525.99. It says that a $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ consumer receives the recursor-generated output only as the authorized finite route already consumable by $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$. The listed failed readings are hidden induction, an unchecked generator, a detached normalization theorem, a subject-reduction source, a host equality principle, and an external proof-producing source.

The required factorization is therefore finite and rowwise. By readiness, the consumer-facing read begins at the displayed carrier R and cannot start from an ambient generator principle. By boundary factorization, the route from O into the closed-substitution or normalization-frontier consumer passes through B, D, A, H, C, P, G, N after I, E, M have admitted the generator surface. By normalization nonescape, no further coordinate is exported for hidden induction, unchecked generation, detached normalization, subject reduction, host equality, or external proof production.

Now take the cited failure, refusal, separation, or countermodel row. If it is used against the named converse, reuse, separation, or readback route, the preceding paragraph forces that use to land at the exact displayed coordinate where the named route tries to omit, replace, detach, or bypass one of the packet rows. That is the strict local obstruction.

Any stronger reading would require data not supplied by the terminal consumer surface. A global impossibility theorem would require a recursor-wide theorem row outside the accepted packet. A host theorem about normalization or substitution would require host normalization, host equality, or host substitution data outside R. Hidden external validation would require an external certificate or proof-producing channel not carried by P, G, or N. These are exactly the readings excluded by terminal readiness and normalization consumer nonescape. Hence the obstruction blocks only the named local route and does not obstruct any consumer route that continues to factor through the displayed packet rows of R. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalConsumerStrictObs`

Lemma 525.101 (AuthorizedGeneratorRecursor terminal output MetaCIC reuse boundary)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Reusing the terminal output O inside a* $\mathsf{MetaCICNormalizationF}$ *consumer is admissible only when the reuse first factors through the same* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *carrier route*

$$B, D, A, H, C, P, G, N$$

after I, E, M have admitted the generator surface. A read that treats O as detached subject reduction, host equality, a hidden generator, or a proof-producing channel is outside the terminal consumer surface of R.

Proof. Start with terminal consumer readiness. It places every terminal normalization or closed-substitution read behind the displayed packet R, so O is consumed as a row with the branch, descent, audit, boundary, and naming data still attached.

Apply boundary factorization. Any public route from O into a ClosedTermSubstitutionBoundary$^{\uparrow}$ or MetaCICNormalizationFrontier$^{\uparrow}$ consumer passes through B, D, A, H, C, P, G, N after the generator admission rows I, E, M have fixed the authorized surface.

Use normalization consumer nonescape to restrict the target consumer. The normalization frontier receives the recursor-generated output only as the authorized finite route already consumable by the closed-substitution boundary, and not as hidden induction, unchecked generation, detached normalization, subject reduction, host equality, or external proof production.

Finally apply the strict obstruction corollary. A proposed reuse that bypasses one of the listed rows is exactly a local obstruction at the named terminal route. Therefore the admissible MetaCIC reuse is precisely the factored boundary route, while detached subject reduction, host equality, hidden generator, and proof-producing-channel readings are rejected. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputMetaCICReus`

Theorem 525.102 (AuthorizedGeneratorRecursor root carrier route)**.** *Every closed-substitution-facing route from an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet factors through the displayed packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route may expose the accepted signature, emitted recursor, motive, branch package, descent row, output row, audit row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, ground-compiler boundary, and local* NameCert *row, but it has no route coordinate outside that finite carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootCarrierRoute`

Proof. Start from the carrier of Definition 525.1. It lists exactly the packet rows $I, E, M, B, D, O, A, H, C, P, G, N$, so a closed-substitution-facing consumer begins from the same finite route used by the root recursor surface.

Apply Definition 525.50. The normalization and closed-substitution consumers receive O only after signature admission, branch coverage, structural descent, audit provenance, and the ground-compiler boundary have been displayed.

Finally use Definition 525.48 and Definition 525.7. These identify the consumer route with the same BHist, hsame, Cont, Pkg, boundary, and NameCert rows, leaving no additional carrier for an off-packet substitution read. □

Theorem 525.103 (AuthorizedGeneratorRecursor closed substitution gate)**.** *For a sibling* ClosedTermSubstitutionB *consumer, the output row O of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet is reachable only after the signature rows I, E, M, branch row B, descent row D, audit row A, componentwise transport row H, displayed continuation row C, package row P, ground-compiler boundary row G, and local naming row N have all been carried in the same finite packet. In particular, the consumer reaches O through* Cont, Pkg, *and* NameCert *rows rather than through a detached substitution principle.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorClosedSubstitutionGate`

Proof. First invoke Definition 525.102. It places every closed-substitution-facing route inside the displayed packet before any consumer reads the output row.

Next apply Definition 525.50. This forces the output read behind accepted signatures, branch coverage, structural descent, audit provenance, and the sibling closed-substitution boundary rows.

The remaining rows are the transport and naming rows. By Definition 525.48, the route exposes H, C, P, G, N as componentwise hsame transport, displayed Cont replay, Pkg provenance, ground-compiler boundary, and local NameCert data. Thus the sibling consumer reaches O only after the stated route has been displayed. □

Theorem 525.104 (AuthorizedGeneratorRecursor audit ledger nonescape)**.** *The audit ledger exported by an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet carries no hidden generator, host induction principle, external axiom, unaudited dependency route, or off-packet substitution channel. Every audit-facing closed-substitution read remains inside the route*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

and inside the local BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by that packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditLedgerNonescape`

Proof. Begin with the carrier route of Definition 525.102. Since every closed-substitution-facing route factors through the finite packet, the audit ledger has no independent source row from which to export a hidden generator or host induction principle.

Next use the closed substitution gate of Definition 525.103. The output row is reachable only after the signature, branch, descent, audit, transport, continuation, package, boundary, and local naming rows have been displayed.

Finally apply Definition 525.7. The handoff consumes the same finite carrier and supplies no external axiom, unaudited dependency route, or off-packet substitution channel. Therefore the audit ledger remains nonescaping for the displayed authorized-generator packet. □

Corollary 525.105 (AuthorizedGeneratorRecursor root route strict obstruction)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet be read by a* ClosedTermSubstitutionBoundary$^{\uparrow}$ *consumer through the root route of Definition 525.102. Any consumer-facing use of the obstruction recorded by Definition 525.104 is strict and local: it can block only the named converse or reuse route that tries to read the output row O without the displayed packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Equivalently, the obstruction is confined to the failure of an off-packet closed-substitution, hidden-generator, host-induction, unaudited-dependency, or detached-output route. It cannot be reinterpreted as a global impossibility theorem, a host theorem, or hidden external validation of any coordinate outside the accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootRouteStrictObstructio`

Proof. Read the root carrier route first. By Definition 525.102, every closed-substitution-facing route from the accepted packet factors through exactly the finite carrier

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The same statement identifies the exposed data as the accepted signature, emitted recursor, motive, branch package, descent row, output row, audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, ground-compiler boundary, and local NameCert row. Hence a consumer-facing route has no independent coordinate from which to read a second generator, a detached substitution principle, a host induction principle, or an external theorem.

Now apply Definition 525.104. The audit ledger carries no hidden generator, host induction principle, external axiom, unaudited dependency route, or off-packet substitution channel, and every audit-facing closed-substitution read remains inside the displayed carrier and inside the local BHist, hsame, Cont, Pkg, and NameCert rows. Thus a displayed failure, refusal, separation, or countermodel row can only deny the attempted converse or reuse route at the packet coordinate where that route asks to leave the carrier.

The remaining interpretations are outside the finite route. A global impossibility theorem would quantify beyond the accepted packet; a host theorem would add semantic strength not carried by $I, E, M, B, D, O, A, H, C, P, G, N$; hidden external validation would require an extra validating row not present among A, H, C, P, G, N. Since the cited carrier and ledger theorems give no such coordinate, the obstruction has exactly its local blocking force against the named converse or reuse route and no wider export surface. □

Theorem 525.106 (AuthorizedGeneratorRecursor acceptance totality)**.** *Every accepted* AuthorizedGeneratorRecurs *carrier exposes its branch, descent, audit, output, and local* NameCert *rows through the finite route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The branch row B, descent row D, audit row A, output row O, displayed BHist *carrier, componentwise* hsame *transport H, displayed* Cont *route C,* Pkg *provenance P, ground boundary row G, and local* NameCert *row N are all available from the accepted packet and no consumer may read a detached recursor surface before these rows have been displayed.*

Proof. Start from the carrier of Definition 525.1. Its accepted packet lists exactly the signature, emitted recursor, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and naming rows.

Apply Definition 525.102. It places every closed-substitution-facing read inside the same finite route and makes the branch, descent, audit, and output rows visible only as packet coordinates.

The naming and replay coordinates are supplied by Definition 525.48 and Definition 525.7. These identify H, C, P, G, N as the displayed BHist, hsame, Cont, Pkg, and NameCert support for the same accepted carrier. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier acceptance totality`

Theorem 525.107 (AuthorizedGeneratorRecursor output stability)**.** *For two accepted* AuthorizedGeneratorRecurso *packets whose displayed branch, descent, audit, transport, continuation, provenance, boundary, and local naming rows are related by the shown* hsame *and* Cont *routes, every output read of O is stable along those routes. The output cannot be separated from the branch package B, descent row D, audit row A,* Pkg *provenance P, ground boundary G, and local* NameCert *row N that present it.*

Proof. First expose the output through Definition 525.106. The read of O begins only after the branch, descent, audit, transport, continuation, provenance, boundary, and naming rows have been displayed in the same finite packet.

Next apply Definition 525.112. It prevents an output consumer from detaching O from A, G, N or from replacing the accepted audit route with a host-side read.

The displayed hsame and Cont rows then transport the same packet surface componentwise. Since no off-packet generator coordinate is exported, any output read along those routes is the same accepted output read carried by B, D, A, P, G, N. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier output stability

Theorem 525.108 (AuthorizedGeneratorRecursor ledger separation)**.** *Audit-ledger reads of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet do not import host recursion, hidden generators, or closed-substitution facts not already present in*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every audit read factors through the displayed branch, descent, output, transport, continuation, provenance, ground-boundary, and local NameCert *rows of the packet.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier ledger separation

Proof. Begin with Definition 525.104. The audit ledger has no independent source for hidden generators, host induction, external axioms, unaudited dependencies, or off-packet substitution channels.

Use Definition 525.103. Closed-substitution-facing reads reach the output only after the accepted signature, branch, descent, audit, transport, continuation, provenance, boundary, and naming rows have been carried together.

Finally apply Definition 525.110. The ground-compiler consumer reads the same finite packet, so the audit ledger cannot import a recursion principle or substitution fact absent from $I, E, M, B, D, O, A, H, C, P,$ □

Corollary 525.109 (AuthorizedGeneratorRecursor acceptance obstruction locality)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet be read through the acceptance-totality surface. Any displayed failure, refusal, separation, or countermodel row cited from the route*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

blocks only the named converse or reuse route whose read would detach O from the displayed branch, descent, audit, transport, continuation, provenance, boundary, or local naming rows. The same row cannot be reinterpreted as a global impossibility result, a host theorem, or hidden external validation for a coordinate absent from the displayed packet.

Proof. By Definition 525.106, every accepted read begins from the finite packet $I, E, M, B, D, O, A, H, C, P, G, N$. Thus a consumer-facing obstruction has access only to the listed signature, emitted recursor, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local NameCert coordinates.

Apply Definition 525.108. It says that audit-ledger reads of the same packet do not import host recursion, hidden generators, external axioms, unaudited dependencies, or closed-substitution facts beyond the displayed rows, and that every audit read factors through the branch, descent, output, transport, continuation, provenance, ground-boundary, and local NameCert rows. Therefore a cited separation or refusal row can obstruct only a read that asks for one of the excluded off-packet

coordinates. Reading it as a global impossibility theorem, a host theorem, or an external validation certificate would require exactly the kind of unlisted coordinate that ledger separation rejects.

It remains to check that the obstruction is strict for output reuse. By Definition 525.107, every accepted output read of O is stable only along the displayed hsame and Cont routes and cannot be separated from B, D, A, P, G, N. Hence a converse or reuse route that omits or replaces those rows is not another interpretation of the accepted output; it is the named failed route blocked at the displayed packet boundary. The obstruction therefore stays local to that coordinate and exports no additional impossibility, host-side theorem, or hidden validation principle. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptanceObstructionLoca`

Theorem 525.110 (AuthorizedGeneratorRecursor ground compiler authorization)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet, any ground-compiler consumer of the recursor output must read the packet through the displayed rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The consumer may reach O only after the accepted signature row I, emitted eliminator row E, motive row M, branch row B, structural descent row D, audit row A, BHist *carrier row, displayed* Cont *row C, package row* Pkg *at P, boundary row G, and local* NameCert *row N have all been retained in the same route.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor ground compiler authorization`

Proof. Start with the carrier of Definition 525.1. It presents the accepted recursor output as one row of the finite BHist packet, so the ground compiler has no initial read outside $I, E, M, B, D, O, A, H, C, P, G, N$.

The computational rows are fixed by the existing branch and descent obligations. Apply Definition 525.2 and Definition 525.5; the branch row is exhausted by the displayed constructors, and the recursive use is local to the motive and descent rows.

The audit and naming side is fixed by Definition 525.6, Definition 525.79, and Definition 525.7. These rows retain audit provenance, carrier admission, boundary exhaustion, Pkg provenance, and the local NameCert certificate before any ground-compiler consumer may read O. □

Theorem 525.111 (AuthorizedGeneratorRecursor finite route exposure)**.** *Every public route exposed by an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet is exhausted by the finite packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Consequently the public surface cannot introduce an unregistered branch, a hidden induction channel, or an off-packet generator path beyond the accepted signature, eliminator, motive, branch, descent, output, audit, transport, continuation, package, boundary, and local certificate rows.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorFiniteRouteExposure`

Proof. Read the route from Definition 525.102. It already factors every closed-substitution-facing public route through the displayed packet and leaves no carrier coordinate outside the finite row list.

Branch exposure is exhausted by Definition 525.2. A public route can name only the branch package associated with the accepted constructors and the emitted eliminator, so an unregistered branch has no row from which to enter.

The remaining escape routes are closed by descent locality and ledger descent. Use Definition 525.5, Definition 525.104, and Definition 525.7; together they keep recursive use, audit provenance, and ground-compiler consumption inside the same finite packet. □

Theorem 525.112 (AuthorizedGeneratorRecursor output audit nonescape)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet, the output row O cannot be consumed apart from the audit provenance row A, the ground-compiler boundary row G, and the local certificate row N. Any consumer of O must preserve the same displayed route through audit provenance, boundary exposure, and* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *certification.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputAuditNonescape`

Proof. Begin with Definition 525.6. It attaches public exposure of O to the audit and provenance rows, so the output cannot be read as an isolated generator result.

The boundary side is supplied by Definition 525.79. The row G is exhausted by the displayed packet and cannot export a hidden proof generator, external axiom, or detached induction channel.

Finally apply Definition 525.110. Since every ground-compiler consumer reads O only after the carrier, audit, boundary, package, and local $\mathsf{NameCert}$ rows are retained, the output row has no nonescaping consumer route apart from A, G, N and the finite packet that carries them. □

Corollary 525.113 (AuthorizedGeneratorRecursor ground-compiler authorization strict obstruction)**.** *Let an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *root packet be read by a ground-compiler consumer. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse or reuse route for the output row O. Then the row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read, omit, detach, replace, or bypass one of the displayed packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

In particular every permitted consumer-facing use of the cited row factors through the audit row A, the componentwise transport row H, the displayed Cont *row C, the package row P, the ground-compiler boundary row G, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N before it can speak about O. The same row cannot be reinterpreted as a global impossibility theorem, a host theorem, hidden external validation, or a certificate for any coordinate not displayed by the accepted packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundCompilerAuthorizati`

Proof. Start with finite route exposure. By Definition 525.111, every public route exposed by the accepted packet is exhausted by

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The theorem also excludes an unregistered branch, a hidden induction channel, and an off-packet generator path beyond the accepted signature, eliminator, motive, branch, descent, output, audit, transport, continuation, package, boundary, and local certificate rows. Hence any consumer-facing use of a displayed failure, refusal, separation, or countermodel row has no public coordinate outside that finite list.

Now apply output-audit nonescape, Definition 525.112. A consumer of O cannot detach the output row from audit provenance A, boundary exposure G, and local certification N; it must preserve the same displayed route through those rows. Together with the ground-compiler authorization route already displayed in the chapter, this forces the consumer-facing readback through

A, H, C, P, G, N, where H carries componentwise transport, C carries the displayed Cont replay, P carries package provenance, G carries the ground-compiler boundary, and N carries the local name certificate.

Consider the cited displayed row. If the named converse or reuse route asks to read O while omitting, replacing, detaching, or bypassing one of the rows above, the two cited theorems place the obstruction exactly at that missing packet coordinate. Thus the row blocks that named route and no other route that remains inside the displayed carrier.

It remains to rule out stronger readings. A global impossibility theorem would require a public conclusion not confined to the accepted packet and not indexed by the named failed route. A host theorem would require a host-facing theorem coordinate not among $I, E, M, B, D, O, A, H, C, P, G, N$. Hidden external validation would require an extra validation or certificate row outside A, H, C, P, G, N. Finite route exposure gives no such coordinate, and output-audit nonescape prevents O from being read apart from the displayed audit, boundary, transport, continuation, package, and local naming route. Therefore the obstruction remains strict and local to the named ground-compiler converse or reuse route. □

Theorem 525.114 (AuthorizedGeneratorRecursor root signature totality)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the accepted signature is total over the displayed constructor row I, emitted recursor row E, motive row M, branch package row B, structural descent row D, output row O, audit row A, componentwise hsame *row H, displayed* Cont *row C,* Pkg *row P, boundary row G, and local* NameCert *row N. Thus the signature cannot be read apart from the finite carrier described by Definition 525.1.*

Proof. Start with carrier admission and acceptance totality. They expose the finite packet $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ and require every accepted signature read to pass through the same displayed constructor, recursor, motive, branch, descent, output, audit, transport, continuation, package, boundary, and local naming rows.

Next use the finite route exposure and root authorization obligations. The generator route is not a hidden host eliminator; it is the accepted signature carried by I, checked by the local NameCert surface, and replayed through the displayed Cont and Pkg rows.

Finally attach the ground compiler authorization. Since the compiler-facing route reads only the displayed packet, no unchecked generator row or unregistered recursion principle can supply a second signature source. The accepted signature is therefore total over the displayed root rows. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootSignatureTotality`

Theorem 525.115 (AuthorizedGeneratorRecursor root branch descent ledger)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet $R = (I, E, M, B, D, O, A, H, C, P, G, N)$, every root branch descent that reaches the output row O is routed through the branch package row B and the structural descent row D before O is available. The ledger admits no proof-producing branch source, hidden host induction, or external recursive call outside the displayed carrier.*

Proof. Read the branch package first. Branch exhaustion and root branch package coverage keep every branch consumption on the accepted constructor signature, emitted recursor, motive, and branch package rows.

Now attach descent. Structural descent locality, branch descent totality, and root descent-audit coupling require recursive branch use to pass through D before the output row O and audit row A can be read.

The terminal branch descent exactness then supplies the root ledger boundary. A route that skipped B or D would need either a proof-producing branch source or a hidden host induction coordinate, but neither coordinate is present in R. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBranchDescentLedger

Theorem 525.116 (AuthorizedGeneratorRecursor root output audit exactness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, output audit exactness keeps the output row* O*, audit row* A*, transport row* H*, continuation row* C*, package row* P*, boundary row* G*, and local* NameCert *row* N *attached inside the same finite packet. No terminal or root consumer may detach* O *from* A, H, C, P, G, N *by appealing to hidden host induction, unregistered recursion, unchecked generator rows, or proof-producing branch sources.*

Proof. Begin with output-audit non-escape and root output audit totality. These results bind O to audit provenance, boundary exposure, package provenance, and local NameCert certification rather than to a detached generator authority.

Next use root host boundary NameCert exhaustion together with terminal audit NameCert non-escape. The output-facing route remains visible only through A, H, C, P, G, N and the displayed rows that carry them.

Finally apply terminal generator admission. Since terminal normalization, closed-substitution, and generator-audit consumers read O only as part of $I, E, M, B, D, O, A, H, C, P, G, N$, the excluded host-induction, unregistered-recursion, unchecked-generator, and proof-producing branch sources have no packet row through which to enter. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputAuditExactness

Theorem 525.117 (AuthorizedGeneratorRecursor terminal generator admission)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, every terminal generator-facing consumer route is admitted only through the displayed packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

In particular, a terminal normalization, closed-substitution, or generator-audit consumer may read the output row O *only as a row of this finite carrier and never as a detached generator authority.*

Proof. Start with the carrier admission and total route surface. By Definition 525.34, Definition 525.106, and Definition 525.111, the accepted packet displays exactly the signature, eliminator, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows.

Next use terminal consumer readiness. Definition 525.97 keeps terminal normalization and closed-substitution consumers behind the same displayed carrier, while Definition 525.110 gives the ground compiler no read outside the same packet rows.

Finally apply output-audit nonescape. By Definition 525.112, the terminal consumer cannot detach O from the audit, boundary, and local NameCert rows, so admission is exhausted by $I, E, M, B, D, O, A, H, C, P, G, N$. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalGeneratorAdmissio

Theorem 525.118 (AuthorizedGeneratorRecursor terminal branch descent exactness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, any terminal branch consumption that reaches the output row* O *must first pass through the branch row* B *and the structural-descent row* D*. No terminal route may consume a branch, recursive call, or output-facing generator result unless the branch is carried by* B *and the descent evidence is carried by* D *inside the same displayed packet.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalBranchDescentExac

Proof. Read the branch surface first. Definition 525.2 and Definition 525.69 force every public branch read to stay on the accepted constructor list and the displayed branch package.

Now attach descent locality. By Definition 525.5, Definition 525.13, and Definition 525.61, a recursive read is public only when its descent evidence is recorded in D together with the accepted signature, emitted eliminator, motive, and branch row.

The terminal step is supplied by Definition 525.97 and Definition 525.110. Since terminal consumers read O only after the finite packet has been displayed, a branch route that skipped either B or D would require a packet coordinate that the carrier does not expose. □

Theorem 525.119 (AuthorizedGeneratorRecursor terminal audit NameCert nonescape)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, terminal output cannot be consumed apart from the audit row A, the ground-boundary row G, and the local* $\mathsf{NameCert}$ *row N. The terminal route exports no output consumer outside the audit ledger, boundary exposure, and local certificate carried by the same packet.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalAuditNameCertNone

Proof. Begin with the output-audit row. Definition 525.112 binds any consumer of O to audit provenance, boundary exposure, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ certification.

Next use the output and boundary readiness results. Definition 525.46, Definition 525.79, and Definition 525.110 keep terminal output consumption on the displayed audit, boundary, package, and local naming rows.

Finally apply terminal consumer readiness. Definition 525.97 leaves no terminal normalization, closed-substitution, or generator-audit route outside the finite packet, so O has no consumer apart from A, G, N and the displayed rows that carry them. □

Corollary 525.120 (AuthorizedGeneratorRecursor terminal generator strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named terminal-generator converse or reuse route in the terminal generator obligation surface of R. Then the row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read, omit, detach, replace, or bypass one of the displayed packet coordinates*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every permitted terminal-generator use of the row factors through the same route block I, E, M, B, D, O, A, the componentwise transport row H, the displayed continuation readback row C, the provenance row P, the ground-boundary row G, and the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N. The same displayed row cannot be reinterpreted as a global impossibility theorem, a host theorem about generators, hidden external validation, or an obstruction to any terminal consumer route that still remains inside those displayed rows.*

Proof. Start with terminal generator admission, Definition 525.117. It admits every terminal generator-facing consumer route only through the finite packet rows

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

It also states that terminal normalization, closed-substitution, and generator-audit consumers may read O only as a row of this finite carrier and never as detached generator authority. Hence a consumer-facing use of the cited failure, refusal, separation, or countermodel row has no initial terminal-generator coordinate outside the displayed packet.

Now apply terminal audit NameCert nonescape, Definition 525.119. Terminal output cannot be consumed apart from the audit row A, the ground-boundary row G, and the local NameCert row N carried by the same packet. Thus any terminal use that speaks about O must retain the route rows that carry the output and audit, the transport and continuation rows H, C, the provenance row P, and the boundary-certificate rows G, N.

Consider the named converse or reuse route against which the displayed row is cited. If that route tries to read O before the route block I, E, M, B, D, O, A is present, or tries to omit the transport row H, the continuation readback row C, the provenance row P, the ground-boundary row G, or the local certificate row N, the two cited theorems locate the obstruction at that missing displayed coordinate. Therefore the row blocks precisely that named terminal-generator route, namely the route whose consumer-facing use fails to remain inside the admitted finite carrier and the audit-certificate nonescape route.

It remains to exclude stronger readings. A global impossibility theorem would require a conclusion not indexed by the accepted packet R and not confined to the named failed route. A host theorem about generators would require a host-facing generator coordinate outside $I, E, M, B, D, O, A, H, C, P, G, N$. Hidden external validation would require an additional validation or certificate row beyond A, G, N and the same displayed packet rows that carry them. Terminal generator admission exposes no such coordinate, and terminal audit NameCert nonescape prevents O from being read apart from the displayed audit, boundary, provenance, transport, continuation, and local naming data. Hence the obstruction is strict and local to the named terminal-generator converse or reuse route. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalGeneratorStrictOb`

Theorem 525.121 (AuthorizedGeneratorRecursor terminal authorized output cover)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, terminal authorized output is exposed only through the displayed generator, branch, output, audit, route, provenance, and local* NameCert *rows. More precisely, any terminal consumer of the output row O must remain inside the finite* BHist *carrier*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

with componentwise hsame *transport at H, displayed* Cont *replay at C, package provenance at* Pkg *on P, audit provenance at A, boundary exposure at G, and local naming evidence at N.*

Proof. Start from terminal generator admission. By Definition 525.117, every terminal generator-facing consumer route is admitted only through the displayed packet rows, and the output row is not available as detached generator authority.

Next attach the ground compiler surface. Definition 525.110 forces any ground-compiler consumer of O to retain the signature, eliminator, motive, branch, descent, audit, BHist, Cont, Pkg, boundary, and local NameCert rows in the same route.

Finally use the existing audit and terminal nonescape facts. Definition 525.112 binds O to audit provenance, boundary exposure, and local certification, while Definition 525.119 keeps terminal output consumption on the displayed audit, boundary, package, and naming rows. Thus terminal authorized output is covered exactly by the displayed rows listed above. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier terminal authorized output cover`

Theorem 525.122 (AuthorizedGeneratorRecursor terminal branch ledger lock)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the terminal branch ledger is locked to the closed-substitution and ground-compiler authorization rows already displayed by the chapter. A terminal consumer may read a branch, recursive call, or output-facing generator result only through the branch row B, descent row D, audit row A, componentwise* hsame *row H, displayed* Cont *row C, package provenance row P, ground-boundary row G, and local* NameCert *row N carried by the same finite packet.*

Proof. First read the branch ledger through the terminal branch exactness result. Definition 525.118 requires every terminal branch consumption that reaches O to pass through B and D inside the displayed packet.

Next lock this branch route to closed substitution. By Definition 525.103, a sibling closed-substitution consumer reaches O only after the signature, branch, descent, audit, transport, continuation, package, boundary, and local naming rows have all been carried in one finite route.

The ground-compiler side supplies the same authorization surface. Definition 525.110 allows the consumer to reach O only after those displayed rows have been retained. Therefore the terminal branch ledger has no independent branch authority beyond the closed-substitution and ground-compiler rows already exposed by the packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalBranchLedgerLock`

Theorem 525.123 (AuthorizedGeneratorRecursor terminal reader nonescape)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, terminal consumers cannot read hidden generator state, host evaluation, unchecked recursion, or private audit routes. Every terminal read of generator output, branch evidence, descent evidence, audit provenance, continuation replay, package provenance, or local naming data remains inside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the packet.*

Proof. Begin with the finite route surface. Definition 525.111 exhausts public routes by

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Hence a terminal reader has no carrier coordinate for hidden generator state or host evaluation outside the displayed packet.

Now use the audit ledger and terminal generator facts. Definition 525.104 blocks hidden generators, host induction principles, external axioms, unaudited dependency routes, and off-packet substitution channels, while Definition 525.117 keeps terminal generator-facing routes on the displayed carrier.

It remains to exclude private terminal audit routes. Definition 525.119 attaches terminal output consumption to the audit, boundary, package, and local NameCert rows, and Definition 525.103 keeps closed-substitution reads behind hsame, Cont, Pkg, and NameCert data. Thus terminal consumers cannot escape to hidden generator state, host evaluation, unchecked recursion, or private audit routes. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalReaderNonescape`

Corollary 525.124 (AuthorizedGeneratorRecursor terminal authorize strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is attached to a named terminal-authorize converse, reuse, or readback*

route in the terminal authorized-output surface of R. Then the row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read, omit, replace, detach, or bypass one of the displayed terminal coordinates

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every permitted terminal-authorize use factors through the displayed branch, descent, output, audit, transport, continuation, package provenance, ground-boundary, and local NameCert *rows of the same packet. The same displayed row cannot be reinterpreted as a global impossibility theorem, a host theorem about recursors, hidden external validation of the failed route, an unchecked generator, hidden generator state, host evaluation, unchecked recursion, a private audit route, or an obstruction to any terminal consumer route that still factors through those displayed rows.*

Proof. Fix the accepted packet R. By Definition 525.123, every terminal read of generator output, branch evidence, descent evidence, audit provenance, continuation replay, package provenance, or local naming data remains inside the displayed BHist, hsame, Cont, Pkg, and NameCert rows of R. The same theorem also excludes hidden generator state, host evaluation, unchecked recursion, and private audit routes. Thus a consumer-facing terminal-authorize use has no coordinate outside

$$I, E, M, B, D, O, A, H, C, P, G, N$$

from which to obtain an alternate generator, host evaluator, private audit path, or external certificate.

Apply Definition 525.122 to the branch side of the same terminal use. A terminal consumer may read a branch, recursive call, or output-facing generator result only through the branch row B, descent row D, audit row A, componentwise hsame row H, displayed Cont row C, package provenance row P, ground-boundary row G, and local NameCert row N carried by the same finite packet. Hence the terminal branch ledger supplies no independent branch authority and no second authorization route beyond the closed-substitution and ground-compiler rows already displayed by R.

Now consider the displayed failure, refusal, separation, or countermodel row. If the row is attached to a named terminal-authorize converse, reuse, or readback route, the reader nonescape theorem forces any consumer-facing use of that row to remain on the displayed reader coordinates, while the branch ledger lock forces the branch, recursive-call, and output-facing parts of the use to remain on B, D, A, H, C, P, G, N. Therefore the row can obstruct only the named local route at the coordinate where that route tries to read, omit, replace, detach, or bypass one of the displayed packet rows.

A stronger reading would require data that neither cited theorem exposes. A global impossibility theorem would require a theorem surface beyond the finite terminal packet; a host theorem about recursors would require host evaluation or host recursion data; hidden external validation would require an external certificate route; an unchecked generator, hidden generator state, unchecked recursion, or private audit route would require exactly the off-packet coordinates ruled out by terminal reader nonescape; and an independent branch authorization would contradict the terminal branch ledger lock. Since every permitted terminal consumer still factors through the displayed rows of R, the obstruction is strict and local to the named terminal-authorize route. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalAuthorizeStrictOb`

Theorem 525.125 (AuthorizedGeneratorRecursor root source)**.** *Every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *recursor route exposes the generator source, authorization token, boundary row, displayed* Cont

route, Pkg *provenance, ledger row, and local* NameCert *row before a consumer may read the output. Concretely, the route remains inside the finite* BHist *packet*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

where I names the accepted source signature, E names the emitted generator row, B names the authorized branch package, G names the boundary row, C names the displayed continuation route, P names package provenance, and N names the local certificate.

Proof. Start from carrier admission in Definition 525.34. It identifies the accepted recursor route with the displayed finite BHist packet and leaves no separate generator-source coordinate.

Use the route authorization surface of Definition 525.93. The source signature, emitted recursor, motive, branch package, descent row, output row, audit row, and local certificate are all read through the same accepted route.

Finally apply the boundary and terminal output rows. Theorems 525.74 and 525.121 force any output-facing read to pass through G, C, P, N together with the displayed ledger and package rows. Hence generator source, authorization token, boundary, continuation, provenance, ledger, and local naming are exposed before output consumption. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootSource`

Theorem 525.126 (AuthorizedGeneratorRecursor boundary closure)**.** *Every closed substitution or generator-normalization consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *output must pass through the displayed source, authorization, boundary,* Cont*,* Pkg*, ledger,* hsame*, and* NameCert *rows before reading O. The consumer may use the output only as a row of the packet $I, E, M, B, D, O, A, H, C, P, G$, not as a detached substitution rule or host generator.*

Proof. First use the closed-substitution handoff. Definition 525.86 requires every closed-substitution consumer to remain inside the displayed BHist, hsame, Cont, Pkg, and NameCert route.

Next use terminal readiness. Definition 525.84 states that O is available only after the signature, emitted recursor, motive, branch package, descent, audit, transport, continuation, provenance, boundary, and local certificate rows have all been retained.

The generator-normalization side is therefore not a separate host route. By Definition 525.122, every terminal branch read is locked to the closed-substitution and ground-compiler authorization rows already displayed by the packet. Thus closed substitution and generator normalization pass through the displayed boundary before reading the output. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBoundaryClosure`

Theorem 525.127 (AuthorizedGeneratorRecursor ledger nonescape)**.** *No ledger of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet can export an unlicensed generator, host recursion oracle, or hidden substitution step. Every ledger-facing read of generator source, authorization, boundary evidence, continuation replay, package provenance, componentwise* hsame *transport, or local naming data remains inside the displayed* BHist*,* Cont*,* Pkg*, and* NameCert *rows of the same packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorLedgerNonescape`

Proof. Begin with the root source theorem above. It exposes generator source, authorization, boundary, continuation, provenance, ledger, and local naming as displayed rows of one finite packet.

Apply boundary closure. Any closed substitution or generator-normalization consumer that reaches O has already passed through the displayed boundary, hsame, Cont, Pkg, and NameCert rows.

It remains to exclude hidden authority. Theorems 525.36 and 525.123 rule out hidden generator state, host evaluation, unchecked recursion, and private audit routes. Therefore the ledger has no coordinate from which to export an unlicensed generator, host recursion oracle, or hidden substitution step. □

Theorem 525.128 (AuthorizedGeneratorRecursor root-unblock carrier admission)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, root-unblock carrier admission exposes exactly the rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

before any output-facing read. These rows are the accepted source signature, emitted generator, motive, branch package, structural descent, output, audit, componentwise transport, continuation, provenance, boundary, and local NameCert *rows of the displayed* BHist *packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootUnblockCarrierAdmissi`

Proof. Start from the carrier admission theorem Definition 525.34. It identifies an accepted root packet with the displayed finite BHist carrier and gives no row outside $I, E, M, B, D, O, A, H, C, P, G, N$.

Use the root source theorem above to place generator source, authorization token, boundary, continuation, provenance, ledger, and local naming on the same admitted packet. Thus the output row O is not visible before the surrounding admission rows have been displayed.

Finally apply boundary closure. The boundary, hsame, Cont, Pkg, and NameCert rows stay attached to the same carrier before any closed-substitution or generator-normalization consumer may read O, so the carrier admission is exhausted by the stated finite packet. □

Theorem 525.129 (AuthorizedGeneratorRecursor root-unblock route totality)**.** *Every output-facing root-unblock route for an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet passes through the source signature, emitted generator, motive, branch package, descent, output, audit, transport, continuation, provenance, boundary, and local* NameCert *rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

No output-facing route skips one of these rows or reads O through a detached generator route.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootUnblockRouteTotality`

Proof. First read route authorization, Definition 525.93. It places accepted signature, emitted recursor, motive, branch package, descent row, output row, audit row, and local certificate on one accepted route.

Next use the root-unblock carrier admission just established. It supplies the remaining displayed transport, continuation, provenance, and boundary rows as part of the same BHist packet before output consumption.

Finally apply the boundary and terminal cover rows: Definition 525.126, Definition 525.74, and Definition 525.121. These force every output-facing route through A, H, C, P, G, N after I, E, M, B, D and keep O inside the displayed carrier rather than on a detached generator channel. □

Theorem 525.130 (AuthorizedGeneratorRecursor root-unblock ledger nonescape)**.** *No ledger read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root-unblock packet exports an unlicensed generator, host recursion oracle, detached substitution step, or hidden authorization. Every ledger-facing read remains inside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootUnblockLedgerNonescap`

Proof. Begin with ledger nonescape above. It already rules out unlicensed generators, host recursion oracles, and hidden substitution steps for ledger-facing reads of generator source, authorization, boundary evidence, continuation replay, package provenance, transport, and local naming data.

Then apply root-unblock route totality. Every output-facing route has already passed through the source signature, emitted generator, motive, branch package, descent, output, audit, transport, continuation, provenance, boundary, and local NameCert rows, so no ledger read can obtain a second authorization route.

It remains to exclude detached substitution. Boundary closure and the terminal reader nonescape theorem Definition 525.123 keep closed-substitution, generator-normalization, and terminal output reads inside the same displayed packet. Hence the ledger has no coordinate from which to export hidden authorization or a substitution step outside the root-unblock packet. □

Corollary 525.131 (AuthorizedGeneratorRecursor root source strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *recursor route. Suppose a displayed failure, refusal, separation, or countermodel row is attached to a named converse, reuse, or readback route at the root-source surface of R. Then the row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read the generator source, authorization token, boundary row, displayed* Cont *route,* Pkg *provenance, ledger row, componentwise* hsame *transport, or local* NameCert *row away from the displayed packet coordinates*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every permitted consumer-facing use factors through the root-source exposure rows and then through the ledger-facing BHist*,* Cont*,* Pkg*,* hsame*, and* NameCert *rows of the same packet. The same displayed row cannot be reinterpreted as a global impossibility theorem for recursors, a host theorem, hidden external validation of the failed route, an unlicensed generator, a host recursion oracle, a detached substitution step, a private authorization channel, or any obstruction to a consumer route that still factors through the displayed rows of R.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootSourceStrictObstructi`

Proof. Fix an accepted recursor route R. By Definition 525.125, the generator source, authorization token, boundary row, displayed Cont route, Pkg provenance, ledger row, and local NameCert row are exposed before a consumer may read the output, and they are exposed only inside the finite BHist packet

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Thus a consumer-facing use of the root source has no independent source coordinate, no authorization token outside the route, no boundary row outside G, no continuation route outside C, no provenance row outside P, and no local certificate outside N.

Apply Definition 525.127 to the ledger side of the same route. Every ledger-facing read of generator source, authorization, boundary evidence, continuation replay, package provenance, componentwise hsame transport, or local naming data remains inside the displayed BHist, Cont, Pkg, and NameCert rows of the packet. The ledger therefore exports no unlicensed generator, host recursion oracle, or hidden substitution step.

Now consider the displayed failure, refusal, separation, or countermodel row. If it is attached to a named converse, reuse, or readback route, the first cited theorem forces the attempted consumer-facing use to pass through the root-source packet coordinates, while the second cited theorem forces its ledger component to remain on the displayed transport, continuation, provenance, boundary, and local-name rows. Hence the row can block only the local route at the coordinate where that route tries to read, omit, replace, detach, or bypass one of those displayed rows.

Any stronger interpretation would require a coordinate not supplied by the packet. A global impossibility theorem would require a theorem surface beyond the finite root-source carrier; a host theorem would require host recursion or host evaluation data; hidden external validation would require an external certificate route; an unlicensed generator, host recursion oracle, detached substitution step, or private authorization channel would require exactly the exports ruled out by ledger nonescape. Since neither cited theorem exposes such a coordinate, the obstruction remains strict and local to the named converse, reuse, or readback route, and it does not block any consumer route that continues to factor through the displayed rows of R. □

Theorem 525.132 (AuthorizedGeneratorRecursor grounded acceptance surface)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *carrier, the generator-row input is admitted only as a displayed* BHist *row of the same root packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The accepted surface reads E through the source signature I, branch package B, structural descent row D, componentwise hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, boundary and ledger row G, and local* NameCert *row N. Hence no acceptance read can treat the generator as a detached host recursor, an abstract carrier equivalence, or an unstated authorization token.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor grounded acceptance surface`

Proof. Start with the root-unblock carrier admission of Definition 525.128. It identifies the accepted generator-row input with the finite BHist packet $I, E, M, B, D, O, A, H, C, P, G, N$ and gives no off-packet coordinate for generator admission.

Next use route totality, Definition 525.129. The output-facing route must pass through the source signature, emitted generator, motive, branch package, descent, output, audit, transport, continuation, provenance, boundary, and local certificate rows before any consumer may read the output.

Finally apply the root carrier and route surfaces in Definition 525.144 and Definition 525.145. These keep the ground-compiler and TasteGate-facing reads attached to the same finite packet, so acceptance is grounded in concrete BHist, hsame, Cont, Pkg, and NameCert rows rather than in carrier-equivalence transport. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorLedgerNonescapeSurface

Theorem 525.133 (AuthorizedGeneratorRecursor readback provenance surface)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, any readback of the generated output O is provenance-preserving: it must pass through the componentwise* hsame *row H, the displayed* Cont *replay row C, the* Pkg *row P, the boundary ledger G, and the local* NameCert *row N of the same* BHist *carrier. A readback therefore cannot replace O by a hidden generator output, a host normalization result, or a package whose provenance is detached from I, E, M, B, D, A.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadbackProvenanceSurface

Proof. Begin with the ledger provenance closure of Definition 525.146. It keeps audit ledger and output provenance inside A, H, C, P, G, N before any closed-substitution, normalization-frontier, or operational-membrane consumer reads O.

The readback route is then the same displayed replay route. By Definition 525.126, closed-substitution and generator-normalization consumers pass through the displayed source, authorization, boundary, Cont, Pkg, ledger, hsame, and NameCert rows before reading the output.

Use root-unblock route totality to recover the source side of the packet. Since I, E, M, B, D remain attached to O through H, C, P, G, N, the readback has a concrete provenance surface and cannot be substituted by a detached output package. □

Theorem 525.134 (AuthorizedGeneratorRecursor ledger nonescape surface)**.** *No ledger-facing read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet can export an unlicensed generator, hidden authorization, host recursion oracle, detached substitution step, or provenance-free output. Every such read remains inside the displayed* BHist *packet and its* hsame*,* Cont*,* Pkg*, boundary, and* NameCert *rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The ledger therefore records nonescape for generator-row input, readback, continuation replay, package provenance, local naming, and boundary evidence on the existing AuthorizedGenerator-Recursor root carrier surface.

Proof. First apply the root-unblock ledger nonescape theorem, Definition 525.130. It rules out unlicensed generators, host recursion oracles, detached substitution steps, and hidden authorization for ledger-facing reads of the displayed root-unblock packet.

Next attach output provenance through the readback surface above. Any read of O has already passed through H, C, P, G, N together with the source and branch rows, so the ledger cannot export a provenance-free output.

Finally combine consumer nonescape with terminal readiness. Theorems 525.36, 525.123, and 525.84 keep closed-substitution, terminal output, and ground-compiler-facing consumers inside the same displayed carrier. Thus the ledger has no escape coordinate beyond the concrete Authorized-GeneratorRecursor root carrier surface. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootLedgerNonescapeSurfac

Corollary 525.135 (AuthorizedGeneratorRecursor grounded acceptance strict obstruction)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet be read through the grounded acceptance surface. If a displayed failure, refusal, separation, or countermodel row is cited against a converse or reuse route for that packet, then the obstruction is strict local data: it blocks only the named route whose read leaves one of the displayed coordinates*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Equivalently, the obstruction applies only to a read that tries to detach the generator row E, the output row O, the componentwise hsame *row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, the boundary ledger row G, or the local* NameCert *row N from the same root packet. It cannot be read as a global impossibility theorem, a host theorem, or hidden external validation for any coordinate not displayed by the packet.*

Proof. By Definition 525.132, grounded acceptance admits the generator-row input only as a displayed BHist row of the same root packet. The accepted read of E is therefore routed through the source signature I, branch package B, structural descent row D, componentwise hsame row H, displayed Cont replay row C, Pkg provenance row P, boundary ledger row G, and local NameCert row N. This already fixes the domain of any consumer-facing objection: it may inspect the shown packet rows, but it has no separate host recursor, carrier-equivalence route, or unstated authorization token to inspect.

Apply Definition 525.134. Ledger-facing reads of the same accepted packet cannot export an unlicensed generator, hidden authorization, host recursion oracle, detached substitution step, or provenance-free output. Every such read remains inside the displayed BHist packet and its hsame, Cont, Pkg, boundary, and NameCert rows. Hence any cited failure, refusal, separation, or countermodel row has force only at the coordinate where the attempted converse or reuse route leaves those rows.

Now suppose the same displayed row were reinterpreted as a global impossibility theorem, a host theorem, or hidden external validation. Such a reinterpretation would have to supply a conclusion beyond the grounded acceptance packet: either a host-side theorem not routed through $I, E, M, B, D, O, A, H, C, P, G, N$, an impossibility claim not indexed by the named failed route, or a validation certificate not carried by H, C, P, G, N. The grounded acceptance surface gives no such coordinate, and ledger nonescape forbids exporting one from the ledger. Therefore the obstruction remains strict and local to the named converse or reuse route. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundedAcceptanceStrict(`

Theorem 525.136 (AuthorizedGeneratorRecursor acceptance budget cover)**.** *Every accepted* AuthorizedGeneratorR *output exposes source, branch, budget, audit, and local* NameCert *rows before any consumer route may read the terminal output. Concretely, the consumer-facing output route remains inside the finite carrier*

$$I, E, M, B, D, O, A, H, C, P, G, N,$$

where I, E, M name the source signature, emitted eliminator, and checked motive, B, D name branch and descent budget rows, A names audit, G names the ground-compiler boundary, and N names the local certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptanceBudgetCover`

Proof. Begin with carrier admission. By Definition 525.1 and Definition 525.144, every accepted output row O is already a row of the displayed packet $I, E, M, B, D, O, A, H, C, P, G, N$.

The source and budget rows are retained by route totality. Apply Definition 525.89: the public route cannot reach O without the accepted signature, emitted eliminator, motive, branch package, structural descent, audit, continuation, package, boundary, and local naming rows.

Finally attach the ground and public-consumer constraints. The ground-compiler handoff of Definition 525.7, the ledger nonescape theorem Definition 525.104, and the public-consumer boundary Definition 525.18 keep the consumer route on the same finite carrier. Hence accepted output is covered by source, branch, budget, audit, and NameCert rows before it is public. □

Theorem 525.137 (AuthorizedGeneratorRecursor ground-route refusal surface)**.** *Any* AuthorizedGeneratorRecurso *route that lacks the ground-compiler witness row G is retained as* GAP*-ledger content rather than exported as a public ground route. Such a route may name source, branch, output, audit, continuation, and provenance data, but it cannot be consumed as a ground-compiler handoff until G is present in the same carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundRouteRefusalSurface`

Proof. Read the ground-compiler handoff theorem. Definition 525.7 places every compiler-facing read behind the carrier row G together with the audit, continuation, package, and local naming rows.

If G is absent, the route fails the same admission test used by Definition 525.145. The route may still be recorded by the audit ledger, but the ledger nonescape theorem Definition 525.104 prevents ledger data from becoming a compiler handoff.

Therefore the route remains at the refusal surface. The public-consumer boundary of Definition 525.18 can read O only through the saturated carrier, so a carrier missing G is not exported as a ground route. □

Theorem 525.138 (AuthorizedGeneratorRecursor terminal output readback totality)**.** *The terminal output readback of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet is exhausted by the displayed carrier, ledger, audit, and boundary rows. Every terminal read of O factors through $I, E, M, B, D, O, A, H, C, P, G, N$ and no terminal consumer receives a hidden generator, detached branch budget, unaudited output, off-packet boundary witness, or external recursion principle.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputReadbackTot`

Proof. Start with the displayed carrier. Root carrier admission and Definition 525.157 place every terminal output route inside $I, E, M, B, D, O, A, H, C, P, G, N$.

Next use the ledger and audit rows. Definition 525.104 confines audit-facing reads to the displayed ledger, while Definition 525.91 keeps output exposure behind the same audit and route data.

Finally use the boundary rows. The ground-compiler handoff and public-consumer boundary theorems keep G, N attached to any terminal readback. Since the carrier has no coordinate for a hidden generator, detached branch budget, unaudited output, off-packet boundary witness, or external recursion principle, the displayed rows exhaust the terminal readback surface. □

Corollary 525.139 (AuthorizedGeneratorRecursor acceptance budget strict obstruction)**.** *In the acceptance-budget surface for* AuthorizedGeneratorRecursor$^{\uparrow}$*, a ground-route attempt missing the boundary row G is a strict local obstruction. It blocks exactly the attempted converse or reuse route that reads O as a public ground-compiler handoff through the displayed carrier*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

It does not assert that accepted recursor output is globally impossible, that the host has supplied an independent theorem, or that hidden external validation has been found. Any such reading would require a coordinate outside the accepted source, branch, budget, audit, transport, continuation, package, boundary, and local NameCert *rows.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptanceBudgetStrictObs

Proof. By Definition 525.136, every consumer-facing read of an accepted output factors through the finite carrier $I, E, M, B, D, O, A, H, C, P, G, N$ before O is public. Thus a consumer route can use source data through I, E, M, branch and descent budget data through B, D, output and audit data through O, A, transport and continuation data through H, C, package provenance through P, boundary data through G, and the local certificate through N; there is no additional consumer coordinate in the acceptance-budget cover.

Now apply Definition 525.137. When G is absent, the named route is retained as GAP-ledger content and is not exported as a public ground route. The refusal therefore attaches to the single missing boundary coordinate required for ground-compiler consumption. It leaves the listed source, branch, output, audit, continuation, provenance, and naming rows in their ledger roles, but it denies the readback that would reuse those rows as a completed ground-compiler handoff.

The obstruction is strict because both cited theorems keep the argument on the same finite packet. A global impossibility claim would need a row saying that no accepted AuthorizedGeneratorRecursor$^{\uparrow}$ output can exist, but the acceptance-budget cover instead starts from accepted output and lists its required rows. A host theorem reading would need a coordinate that exports a theorem outside the carrier, while the refusal surface exports only when G is present in that carrier. A hidden external validation reading would need an off-packet certificate or validation row, but the displayed carrier ends at the local NameCert row N. Hence the failure blocks only the named ground-route converse or reuse attempt and cannot be reinterpreted as a global impossibility, a host theorem, or hidden external validation. □

Theorem 525.140 (AuthorizedGeneratorRecursor accepted signature route)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

any public read that consumes the output row O must first expose the accepted signature row I, emitted recursor row E, checked motive row M, authorized branch package B, and structural-descent row D. The route may then read O only as the output row of the same carrier; it has no detached branch table, motive-free recursive channel, hidden generator source, or off-packet signature route.

Proof. Project the carrier of Definition 525.1. It places I, E, M, B, D before O in one finite BHist packet, so the public consumer begins with the displayed signature, recursor, motive, branch, and descent rows rather than with a separate generator source.

Use Definition 525.34 and Definition 525.93. These results identify accepted route consumption with the same finite carrier and require the route to retain the source signature, emitted recursor, motive, branch package, descent row, output row, audit row, and local certificate together.

Finally apply branch exhaustion and descent locality. Theorems 525.2 and 525.5 keep the branch package on the accepted constructor list and keep recursive use local to the motive and descent rows. Therefore O is reached only after I, E, M, B, D have become visible on the accepted signature route. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptedSignatureRoute

Theorem 525.141 (AuthorizedGeneratorRecursor output boundary totality)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the output row O becomes public only while the audit row A, componentwise hsame *transport H, displayed* Cont *route C,* Pkg *provenance P, boundary row G, and local* NameCert *row N are retained. The public output boundary exports no hidden generator, free induction principle, external axiom channel, or unaudited dependency route outside R.*

Proof. Start with the output side of the displayed packet. The rows O, A, H, C, P, G, N are not separate public surfaces; they are the output, audit, transport, continuation, provenance, boundary, and local naming coordinates of the same carrier.

Apply Definition 525.74. It exhausts the output boundary ledger by $I, E, M, B, D, O, A, H, C, P, G, N$ and rules out a hidden generator, unaudited dependency route, external axiom channel, host-kernel completeness claim, or naming certificate outside the displayed packet.

Use Definition 525.121 for terminal consumption. That theorem keeps terminal readers on the same authorized output rows, so public exposure of O cannot discard A, H, C, P, G, N while retaining an output name. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor output boundary totality`

Theorem 525.142 (AuthorizedGeneratorRecursor audit NameCert lock)**.** *Every downstream closed-substitution, ground-compiler, or TasteGate-facing consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *output is locked to the displayed packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The lock is the composite of accepted signature exposure, output boundary totality, the chapter's TasteGate handoff, and the OperationalMembrane$^{\uparrow}$ *audit route; it permits no closed-substitution read, ground-compiler read, or local* NameCert *read through a row outside that packet.*

Proof. First combine Definition 525.140 with Definition 525.141. The first theorem exposes I, E, M, B, D before output consumption, and the second retains O, A, H, C, P, G, N at the public boundary.

Next add the closed-substitution and ground-compiler consumers. Theorems 525.86 and 525.110 require both consumer families to read the output only through the displayed BHist, hsame, Cont, Pkg, boundary, and NameCert rows.

Finally synchronize the audit route with the operational membrane and the TasteGate handoff already cited by the chapter. Apply Definition 525.12, Definition 533.6, and the checked carrier handoff named by `BEDC.Derived.AuthorizedGeneratorRecursorUp.taste_gate`. These identify the audit and carrier-alignment reads with the same displayed packet, so downstream consumers cannot replace $I, E, M, B, D, O, A, H, C, P, G, N$ by an off-packet route. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditNameCertLock`

Lemma 525.143 (AuthorizedGeneratorRecursor accepted signature downstream coverage)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet with displayed carrier*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every accepted local consumer of the accepted signature output is exhausted by the finite rows displayed in R. More precisely, the consumer can read only an admission alternative among I, E, M, B, D, an output-boundary ledger row among O, A, H, C, P, G, N, or the same-packet coverage obtained by keeping those two surfaces coupled inside the carrier. A request for a detached branch table, motive-free recursive channel, hidden generator source, off-packet signature route, free induction principle, external axiom channel, unaudited dependency route, or naming certificate outside R remains outside the accepted BEDC *packet.*

Proof. Fix an accepted root packet and project the displayed carrier R. By Definition 525.140, any public read that consumes O must first expose the accepted signature row I, emitted recursor row E, checked motive row M, authorized branch package B, and structural-descent row D. The same theorem also says that the subsequent read of O is a read of the output row of this carrier, with no detached branch table, motive-free recursive channel, hidden generator source, or off-packet signature route. Thus the admission side of any accepted local consumer is already exhausted by the finite alternatives I, E, M, B, D.

Now apply Definition 525.141 to the output side. Once O is public, the audit row A, componentwise hsame transport H, displayed Cont route C, Pkg provenance P, boundary row G, and local NameCert row N remain attached to the same carrier. Hence a downstream consumer that passes the admission side can only proceed through the displayed output-boundary ledger O, A, H, C, P, G, N.

Combining the two cited theorems leaves no third source of consumer-facing data. A local consumer either stops on the admission alternatives I, E, M, B, D, reads the output-boundary ledger O, A, H, C, P, G, N, or uses their same-packet coupling as the coverage row for R. The alternatives excluded by the accepted signature route are absent before O, and the alternatives excluded by output boundary totality are absent after O. Therefore every unlisted coordinate remains outside the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAcceptedSignatureDownstre`

Theorem 525.144 (AuthorizedGeneratorRecursor root carrier admission)**.** *Every accepted* AuthorizedGeneratorRec *root packet exposes only the carrier rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

before any root-unblock, ground-compiler, or output-facing consumer may read it. Here I is the accepted signature, E the emitted eliminator, M the motive, B the branch package, D the structural descent row, O the output row, A the audit row, H the componentwise transport row, C the displayed continuation route, P the package provenance, G the boundary row, and N the local NameCert *row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootCarrierAdmission`

Proof. Project the carrier of Definition 525.1. Its displayed BHist packet is exactly $I, E, M, B, D, O, A, H, C, P, G, N$ and the definition gives no coordinate for an off-packet generator, hidden branch source, or detached output name.

Branch and descent exposure are already exhausted by the local root obligations. Theorems 525.2 and 525.5 keep every public recursive read on the signature, emitted eliminator, motive, branch package, and structural descent rows.

Finally attach audit and handoff data. The audit boundary of Definition 525.4 keeps output exposure behind audit provenance, and Definition 525.7 reads the checked carrier alignment only as the same finite packet. Hence the accepted root carrier exposes no rows beyond the displayed list. □

Theorem 525.145 (AuthorizedGeneratorRecursor ground-compiler route totality)**.** *Every ground-compiler or TasteGate-facing route out of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet factors through the signature row I, eliminator row E, motive row M, branch row B, descent row D, audit row A, continuation row C, package row P, boundary row G, and local certificate row N, together with the output row O and componentwise transport row H of the same carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundCompilerRouteTotali`

Proof. Start from root carrier admission above. The ground-compiler route begins with the same finite packet $I, E, M, B, D, O, A, H, C, P, G, N$ and has no separate compiler-facing carrier.

The computational part is forced through the signature, eliminator, motive, branch, and descent rows. Apply Definition 525.2 and Definition 525.5; together they rule out an unlisted constructor case, detached branch table, or non-structural recursion channel.

The public handoff is then routed through audit, continuation, package, boundary, and naming rows. Theorems 525.4 and 525.7 place the TasteGate-facing read behind the finite NameCert packet rather than behind a separate proof-producing generator. Thus every ground-compiler route is total over the displayed rows. □

Theorem 525.146 (AuthorizedGeneratorRecursor ledger provenance closure)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the audit ledger and output provenance close inside* A, H, C, P, G, N *before any* ClosedTermSubstitutionBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, *or* OperationalMembrane$^{\uparrow}$ *consumer reads* O*. Such a consumer receives the output only as a row of the finite packet* $I, E, M, B, D, O, A, H, C, P$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorLedgerProvenanceClosure`

Proof. Begin with the root carrier admission theorem. It places the output row O inside the admitted carrier and keeps the audit, transport, continuation, provenance, boundary, and local certificate rows attached to that output.

Next use the branch and descent obligations. Branch exhaustion and structural-descent locality keep the recursive source of O on I, E, M, B, D, so a downstream consumer cannot replace the output by a detached generator or hidden substitution step.

Finally apply audit boundary and ground-compiler handoff. The audit ledger is closed by Definition 525.4, while Definition 525.7 confines the compiler-facing handoff to the same NameCert carrier. Therefore closed-substitution, normalization-frontier, and operational-membrane consumers read only the closed provenance packet, not an off-route output. □

Lemma 525.147 (AuthorizedGeneratorRecursor root admission downstream coverage)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet, every local downstream consumer admitted by the root carrier is covered by the finite consumer interface of Definition 525.145. More explicitly, a consumer-facing use is exhausted by one of the displayed roles: the accepted signature row* I*, emitted eliminator row* E*, motive row* M*, branch package row* B*, structural descent row* D*, output row* O*, audit row* A*, componentwise transport row* H*, displayed continuation row* C*, package provenance row* P*, boundary row* G*, or local* NameCert *row* N*. An unlisted constructor alternative, detached branch source, non-structural recursion channel, unaudited ledger, detached output name, or external compiler certificate is therefore not an accepted coordinate of the* AuthorizedGeneratorRecursor$^{\uparrow}$ *BEDC packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAdmissionDownstreamCo`

Proof. Start with root carrier admission, Definition 525.144. It presents the accepted root carrier as the displayed packet of signature, eliminator, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local certificate rows, and it gives no additional coordinate for an off-packet generator, hidden branch source, or detached output name.

Now apply ground-compiler route totality, Definition 525.145. Any ground-compiler or TasteGate-facing route out of the accepted packet factors through exactly those named roles. The computational alternatives are therefore exhausted by the accepted signature row I, emitted eliminator row

E, motive row M, branch package row B, and structural descent row D; a purported constructor case or recursive branch not admitted by those rows has no consumer-facing route.

It remains to account for ledger, readback, and output use. The same totality theorem keeps public output and readback on O, A, H, C, P, G, N: the output row O is read only together with audit row A, componentwise transport row H, displayed continuation row C, package provenance row P, boundary row G, and local NameCert row N. Thus every accepted local consumer factors through the finite alternatives, ledger rows, and coverage rows already displayed by the root carrier admission theorem. Since the admitted carrier has no coordinate beyond those roles, every excluded alternative remains outside the BEDC packet rather than becoming a hidden consumer route. □

Definition 525.148 (AuthorizedGeneratorRecursor readiness route obligation packet)**.** *An* AuthorizedGeneratorRec *readiness route is the finite* BHist *packet*

$$Q = (U, J, K, O, L, H, C, P, N)$$

read from the admitted carrier of Definition 525.1. The authorization row U records the accepted constructor signature, emitted eliminator, checked motive, complete branch package, and structural descent rows I, E, M, B, D; the generator row J records the authorized recursor call; the ground-compiler recognition row K records that the route reaches the compiler boundary only through G; the output row O is the displayed public name; the refusal ledger L records the audit row A together with the refusal of unlisted branches, non-structural recursion, external axioms, hidden generators, and unaudited dependency channels; H is componentwise hsame *transport; C is the displayed* Cont *replay route; P is the* Pkg *provenance row; and N is the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessRoute`

Theorem 525.149 (AuthorizedGeneratorRecursor ground compiler admission)**.** *Every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *readiness route factors its generator output through the authorization row U and the ground-compiler recognition row K. Equivalently, no route may expose O to a ground-compiler consumer unless the same finite packet retains the accepted signature, emitted eliminator, motive, branch package, structural descent, audit provenance, boundary row, transports, continuation replay, package provenance, and local NameCert row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor ground compiler admission`

↪ variant `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorGroundCompilerAdmission`

Proof. Start with root carrier admission. By Definition 525.144, the accepted packet exposes only $I, E, M, B, D, O, A, H, C, P, G, N$ before a ground-compiler or output-facing consumer may read it.

The authorization row U is obtained by grouping the signature, eliminator, motive, branch, and descent rows. Theorems 525.2 and 525.5 keep the authorized generator call on those rows, so J cannot be read as a detached proof-producing source.

Now attach compiler recognition. The ground-compiler authorization theorem and the handoff theorem, Definition 525.110 and Definition 525.7, force any compiler-facing read of O to retain G, H, C, P, N with the audit row. Thus the output reaches the compiler only through U and K inside the displayed readiness packet. □

Theorem 525.150 (AuthorizedGeneratorRecursor output refusal route)**.** *Any request against an* AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route that omits either the authorization row U or the refusal ledger L remains* GAP*-ledger content. Such a request is not exported as the public output row O and is not consumed as a ground-compiler route.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor output refusal`

Proof. First read the refusal side of the carrier. The GAP boundary refusal theorem, Definition 525.24, places external axioms, hidden generators, non-structural recursion, and unlisted branches behind the boundary row G and the audit row A.

If U is absent, the request has no accepted signature, eliminator, motive, branch package, or structural descent row from which a recursor output can be formed. Root authorization obligations and root ground-compiler authorization therefore leave the request at the boundary rather than at O.

If L is absent, the request has no source-refusal, dependency-refusal, or hidden-generator ledger attached to the output route. The audit dependency refusal theorem and the root output boundary package keep such a request in the GAP ledger until the refusal rows are restored.

Finally apply root output saturation. Since Definition 525.157 exports O only through the saturated carrier route, a request missing U or L cannot be the exported output and cannot be read as a compiler-recognized route. □

Theorem 525.151 (AuthorizedGeneratorRecursor NameCert obligation exhaustion)**.** *The rows of the readiness route obligation packet exhaust the local carrier, classifier, stability, ledger, and non-escape obligations for* AuthorizedGeneratorRecursor$^{\uparrow}$*. Carrier admission is supplied by U, J, K, O; classifier visibility is supplied by C, N; stability is supplied by H and P; the refusal ledger is supplied by L; and non-escape is supplied by the joint retention of U, K, L, H, C, P, N around O.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNameCertObligationExhaust`

Proof. Carrier exhaustion is already finite. The carrier admission packet supplies $I, E, M, B, D, O, A, H, C, P, G, N$, and the readiness route only repackages these rows as authorization, generator, ground-compiler recognition, output, refusal, transport, replay, provenance, and local naming rows.

Classifier and stability obligations are retained by the public surface. The NameCert surface of Definition 525.3 classifies the route through the same packet, while output-name stability keeps O attached to H, C, P, G, N and to the authorization rows.

The ledger obligations are exhausted by refusal and audit rows. Use Definition 525.30, Definition 525.78, and Definition 525.24; together they place source refusal, dependency refusal, and hidden-generator refusal in L before output exposure.

It remains to rule out escape. Root carrier admission, root output route saturation, and ground-compiler admission above keep every consumer-facing read inside the same finite packet. Therefore the readiness rows exhaust the local carrier, classifier, stability, ledger, and non-escape obligations without adding an off-packet route. □

Corollary 525.152 (AuthorizedGeneratorRecursor readiness-route public-admission strict obstruction)**.** *Let*

$$Q = (U, J, K, O, L, H, C, P, N)$$

be an AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named public-admission converse or reuse route for the*

output row O. Then the cited row is a strict local obstruction: it blocks only the named route whose consumer-facing read omits, replaces, or bypasses one of the displayed readiness coordinates

$$U, J, K, O, L, H, C, P, N.$$

In particular, its force is confined to the authorization row U, the generator-call row J, the ground-compiler recognition row K, the output row O, the refusal ledger L, componentwise hsame *transport H, the* Cont *replay route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N. It cannot be reinterpreted as a global impossibility theorem, a host theorem, hidden external validation, or an off-packet certificate for any coordinate not displayed by Q.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessRoutePublicAdmis`

Proof. Read the packet from Definition 525.148. The readiness route has exactly the displayed finite coordinates $U, J, K, O, L, H, C, P, N$. The row U is the only authorization package for the accepted signature, eliminator, motive, branch package, and structural descent rows; J is only the authorized recursor call; K reaches the compiler boundary only through the displayed boundary row; L is the refusal ledger for unlisted branches, non-structural recursion, external axioms, hidden generators, and unaudited dependency channels; and H, C, P, N are respectively the displayed transport, replay, provenance, and local naming rows. Thus a consumer-facing use of the public output has no coordinate from which to read a detached converse route, a second generator route, a host theorem, or an external validation certificate.

Now apply Definition 525.150. Any request against the readiness route that omits U or L remains GAP-ledger content. It is not exported as O and is not consumed as a ground-compiler route. Hence a displayed refusal, failure, separation, or countermodel row can affect a public-admission converse or reuse attempt only at the precise coordinate where that attempt omits, replaces, or bypasses the authorization package or the refusal ledger.

It remains to exclude stronger readings. Interpreting the same row as a global impossibility theorem would require a theorem surface not indexed by the named failed route. Interpreting it as a host theorem would require a host-facing recursion, induction, equality, or completion coordinate outside the packet. Interpreting it as hidden external validation would require a certificate channel not carried by H, C, P, N and not audited by L. None of these coordinates appears in Q, and the output-refusal route leaves every request missing U or L at the local GAP boundary. Therefore the obstruction is strict and local to the named public-admission converse or reuse route. □

Theorem 525.153 (AuthorizedGeneratorRecursor root carrier totality)**.** *Every accepted* AuthorizedGeneratorRecur *readiness route exposes its root carrier only through the finite* BHist *rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

and through the repackaged readiness rows

$$U, J, K, O, L, H, C, P, N.$$

The accepted carrier, generator authorization row, boundary row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row are therefore read together. No hidden generator, external axiom, host induction principle, detached branch table, unchecked recursive call, or off-packet substitution source is a carrier coordinate.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootCarrierTotality`

Proof. Start with root carrier admission. By Definition 525.144, the admitted packet is exactly the finite route through $I, E, M, B, D, O, A, H, C, P, G, N$ before any compiler, normalization, or closed-substitution consumer may read the output.

Next use the readiness route definition. Definition 525.148 repackages the signature, emitted eliminator, motive, branch, descent, output, audit, boundary, transport, replay, provenance, and naming rows as $U, J, K, O, L, H, C, P, N$ without adding a further carrier source.

Finally apply NameCert obligation exhaustion, Definition 525.151. The carrier, classifier, stability, refusal-ledger, and non-escape clauses are all discharged by the displayed rows, so any route omitting one of them remains outside the accepted root carrier. □

Theorem 525.154 (AuthorizedGeneratorRecursor closed-substitution source exposure)**.** *Every closed-substitution consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *route must preserve the source rows that authorize the generator output: the accepted signature row I, emitted recursor row E, checked motive row M, branch package B, structural descent row D, audit row A, ground-boundary row G, componentwise* hsame *row H, displayed* Cont *route C,* Pkg *provenance P, refusal ledger L, and local* NameCert *row N. A closed-substitution read that omits any of these rows remains boundary content and is not an admitted output handoff.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorClosedSubstitutionSourceE`

Proof. First read the closed-substitution gate. Definition 525.103 places every output read for a closed-substitution consumer after the signature, branch, descent, audit, transport, continuation, package, boundary, and local naming rows have been displayed.

Next attach the closed-substitution handoff. Definition 525.86 keeps the handoff inside $I, E, M, B, D, O, A, H, C$ and treats the sibling closed-substitution boundary as finite BEDC boundary rows rather than as an unrestricted substitution calculus.

Finally use output refusal and closed-substitution admission, Definition 525.150 and Definition 525.169. The refusal ledger and generator authorization rows determine whether the request reaches O or remains in the GAP ledger, so the closed-substitution source exposure is exactly the displayed finite packet. □

Theorem 525.155 (AuthorizedGeneratorRecursor terminal output non-escape)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route, terminal normalization, closed-substitution, and ground-compiler consumers may read the output row O only through the displayed generator authorization row U, refusal ledger L, ground-compiler recognition row K, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The terminal output exports no hidden generator state, host evaluator, unchecked recursion principle, external axiom, detached proof producer, or off-packet substitution channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputNonescape`

Proof. Begin with readiness determinacy. Definition 525.168 says that two accepted readiness routes agreeing on U, J, K, L, H, C, P, N expose the same public output row O.

Now attach public output coverage. Definition 525.170 routes normalization, closed-substitution, and ground-compiler public consumers through the same readiness packet before any of them reads O.

Finally combine terminal authorized output cover with root output route saturation, Definition 525.121 and Definition 525.157. These theorems keep terminal output consumption on the displayed audit, boundary, transport, continuation, provenance, and local naming rows, leaving no carrier coordinate for an escaping output route. □

Corollary 525.156 (AuthorizedGeneratorRecursor root-totality strict obstruction)**.** *Let*

$$Q = (U, J, K, O, L, H, C, P, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route, read from the root carrier rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named root-totality converse or reuse route for Q. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read omits, replaces, detaches, or bypasses one of the displayed output, source, transport, replay, provenance, refusal, boundary, or local naming coordinates

$$I, E, M, B, D, O, A, H, C, P, G, N$$

or their readiness packaging

$$U, J, K, O, L, H, C, P, N.$$

In particular, the obstruction has no force as a global impossibility theorem, no force as a host theorem about recursors or totality, and no force as hidden external validation for any coordinate not displayed by the accepted packet.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessRoute root totality strict obstruction`

Proof. Start with terminal output non-escape, Definition 525.155. It lists the only rows through which terminal normalization, closed-substitution, and ground-compiler consumers may read the public output O: the authorization row U, refusal ledger L, ground-compiler recognition row K, componentwise hsame transport H, displayed Cont replay C, Pkg provenance P, and local NameCert row N. The same theorem excludes hidden generator state, host evaluation, unchecked recursion, external axioms, detached proof production, and off-packet substitution as output coordinates.

Now attach closed-substitution source exposure, Definition 525.154. A closed-substitution consumer must preserve the source rows that authorize the output: $I, E, M, B, D, A, G, H, C, P, L, N$. Thus any consumer-facing use of the displayed obstruction factors through the source rows I, E, M, B, D, A, G, the refusal ledger L, the transport and readback rows H, C, P, and the local naming row N before it can speak about the output row O.

Consider the cited failure, refusal, separation, or countermodel row. If the named converse or reuse route asks to read O while omitting the authorization row U, the refusal ledger L, the ground-recognition row K, the source rows I, E, M, B, D, A, G, or the transport and readback rows H, C, P, N, the two cited theorems place the failure exactly at that missing coordinate. The row therefore blocks the named route at the point where the route leaves the displayed packet. If a consumer instead keeps all listed rows, this corollary supplies no rejection of that consumer; the obstruction is tied to the named omitted, replaced, detached, or bypassed coordinate.

It remains to rule out stronger readings. A global impossibility theorem would require a conclusion not indexed by the named failed route and not confined to Q. A host theorem about recursors or totality would require a host evaluator, host induction rule, unchecked recursive call, or external axiom, each of which is excluded by terminal output non-escape. Hidden external validation would require a certificate or substitution channel outside H, C, P, L, N, which closed-substitution source exposure does not export. Since no such coordinate is present in the root carrier or in the readiness packaging, the displayed row remains a strict local obstruction to the named root-totality converse or reuse route only. □

Theorem 525.157 (AuthorizedGeneratorRecursor root output route saturation)**.** *Every public output route of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet factors through the accepted signature row I, emitted eliminator row E, checked motive row M, branch package B, structural descent row D, output row O, audit row A, componentwise transport row H, displayed continuation row C, provenance row P, ground-compiler boundary row G, and local* NameCert *row N. Thus the output is public only as the saturated finite carrier route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputRouteSaturation
↪ variant BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root output route saturation

Proof. Begin with the carrier and root output audit rows. Definition 525.1 displays the accepted signature, eliminator, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and naming rows, while Definition 525.46 places every public read of O behind the same packet.

Next attach boundary and compiler visibility. Definition 525.47 prevents the boundary row from exporting a hidden generator or host induction principle, and Definition 525.7 reads the ground-compiler boundary only through the accepted carrier.

Finally use consumer readiness. Definition 525.48 packages the output route for downstream consumers without dropping branch, descent, audit, boundary, provenance, or local naming rows. Hence every public output route factors through the displayed saturated route. □

Theorem 525.158 (AuthorizedGeneratorRecursor root boundary nonescape saturation)**.** *The saturated public output route of Definition 525.157 exports no hidden generator, host induction principle, external axiom, unaudited dependency route, or off-packet substitution channel. Any consumer that reaches O must keep the accepted signature, eliminator, motive, branch, descent, audit, transport, continuation, provenance, ground-compiler boundary, and local NameCert rows inside the same* BHist *packet.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootBoundaryNonescapeSatu

Proof. Project the saturated route

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route already contains the only output, audit, ground-boundary, provenance, and local naming rows supplied by the accepted carrier.

Apply root boundary nonescape and the ground-compiler handoff. Definition 525.47 excludes hidden generator, host induction, external axiom, and unaudited dependency exports at the boundary, while Definition 525.7 keeps the compiler-facing read on the same finite packet.

It remains to rule out substitution escape. Definition 525.48 and Definition 525.46 force closed-normal and generator-audit consumers to retain the output audit route before using O. Since no off-packet substitution coordinate appears in the carrier, the saturated route has no channel through which such a substitution can be exported. □

Corollary 525.159 (AuthorizedGeneratorRecursor root output-route strict obstruction)**.** *Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse or reuse route in the root output-route surface of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet.*

Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read omits, replaces, detaches, or bypasses one of the displayed finite carrier coordinates

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route coordinates are I, E, M, B, D, O, A, the transport coordinate is H, the readback and provenance coordinates are C, P, and the boundary-certificate coordinates are G, N. The same displayed row cannot be reinterpreted as a global impossibility theorem for authorized generator recursors, a host theorem, hidden external validation of the failed route, or evidence for any unlisted consumer coordinate.

Proof. Start with output-route saturation. By Definition 525.157, every public output route of an accepted packet factors through the finite carrier

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Thus a consumer-facing read of O has only the listed packet coordinates available. Its route side must pass through the accepted signature, emitted eliminator, checked motive, branch package, structural descent row, output row, and audit row I, E, M, B, D, O, A. Its transport side is the componentwise row H, its continuation and provenance readback is C, P, and its boundary and local certificate side is G, N.

Apply boundary nonescape saturation. Definition 525.158 says that the same saturated route exports no hidden generator, host induction principle, external axiom, unaudited dependency route, or off-packet substitution channel, and that any consumer reaching O must keep the accepted signature, eliminator, motive, branch, descent, audit, transport, continuation, provenance, ground-compiler boundary, and local NameCert rows inside the same BHist packet. Therefore a negative row can act only at one of those displayed coordinates: it may deny the named converse or reuse route when that route tries to detach O from its branch, descent, audit, transport, readback, boundary, or local naming support.

No stronger reading has a carrier. A global impossibility theorem would require a theorem coordinate not indexed by the named route and not confined to the accepted packet. A host theorem would require a host-facing induction or proof row outside $I, E, M, B, D, O, A, H, C, P, G, N$. Hidden external validation would require an additional validating certificate, dependency route, or substitution channel outside A, H, C, P, G, N. The two cited saturation theorems provide no such coordinate and explicitly keep every public read of O on the displayed finite route. Hence the obstruction is strict and local: it blocks the named converse or reuse route at the displayed packet coordinate that the row denies, and it rejects every unlisted interpretation as outside the accepted AuthorizedGeneratorRecursor$^{\uparrow}$ output-route surface. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputRouteStrictObst`

Theorem 525.160 (AuthorizedGeneratorRecursor output route readiness)**.** *For an accepted* AuthorizedGeneratorRe *packet, the output row O is route-ready only inside the finite packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The packet must display the accepted signature row I, emitted eliminator row E, checked motive row M, authorized branch row B, structural descent row D, audit row A, componentwise hsame *transport row H, displayed* Cont *route C,* Pkg *provenance row P, ground-boundary row G, and local* NameCert *row N in the same* BHist *carrier. Output exposure is admitted only through those authorized signature, branch, descent, audit, route, package, boundary, and local naming rows.*

Proof. Begin with the carrier route. The finite packet $I, E, M, B, D, O, A, H, C, P, G, N$ is the only displayed BHist surface on which the accepted signature, emitted eliminator, motive, branch package, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows are simultaneously present.

Next apply Definition 525.157 and Definition 525.126. They keep O attached to the accepted branch, descent, audit, hsame, Cont, Pkg, boundary, and NameCert rows before any public output read is allowed.

Thus output readiness is not a detached generator fact. It is the admission of O through the same authorized signature, branch, descent, audit, route, package, boundary, and local naming rows of the packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor output route readiness`

Theorem 525.161 (AuthorizedGeneratorRecursor boundary ledger totality)**.** *Every boundary-ledger read of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *output is total only when it preserves all rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

as displayed rows of the same BHist *packet. In particular,* hsame*,* Cont*,* Pkg*, and* NameCert *support cannot be detached from the accepted signature, branch, descent, audit, route, package, boundary, and local naming surface.*

The ledger admits no output exposure through an unauthorized signature, unchecked branch, missing descent row, absent audit row, private continuation route, off-packet package, hidden boundary, or external naming certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBoundaryLedgerTotality`

Proof. Project the packet onto its boundary and ledger rows. The displayed carrier contains G only together with the accepted source signature, emitted eliminator, checked motive, branch package, descent, output, audit, transport, continuation, provenance, and local certificate.

Use Definition 525.157 to recover the complete route through $I, E, M, B, D, O, A, H, C, P, G, N$. The same theorem forces every public output read to retain the BHist, hsame, Cont, Pkg, and NameCert support rows.

It remains to exclude off-packet ledger exposure. Definition 525.126 and Definition 525.127 keep closed-substitution, generator-normalization, and output-facing consumers on the same displayed packet, so no unauthorized signature, unchecked branch, missing descent row, private route, hidden boundary, or external naming surface can supply O. □

Theorem 525.162 (AuthorizedGeneratorRecursor downstream readback exhaustion)**.** *Every downstream readback of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *output is exhausted by the finite rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. A downstream consumer may read output exposure only through the authorized signature, branch, descent, audit, route, package, boundary, and local naming rows already present in that packet.*

No detached generator route, host recursion oracle, hidden substitution channel, unlicensed output row, or external naming witness remains for a downstream consumer after the displayed packet has been read.

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorDownstreamReadbackExhaust

Proof. First use Definition 525.160. It places O behind the accepted signature, emitted eliminator, motive, branch package, descent, audit, transport, continuation, provenance, boundary, and local certificate rows of the same BHist packet.

Next use Definition 525.161. Any route that reaches the boundary or ledger must preserve $I, E, M, B, D, O, A, H, C, P, G, N$ together with the hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 525.178 and Definition 525.179. They allow downstream consumers to read the authorized output only through the displayed packet, so detached generators, host recursion oracles, hidden substitution channels, unlicensed output rows, and external naming witnesses are exhausted by the same finite route. □

Lemma 525.163 (AuthorizedGeneratorRecursor output route downstream coverage)**.** *Let c be an accepted local consumer of an* AuthorizedGeneratorRecursor$^{\uparrow}$ *output route. If c reads the output row through the readiness surface, then its consumer-facing use is exhausted by the finite packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

More precisely, the route alternatives are the accepted signature row I, emitted eliminator row E, checked motive row M, authorized branch row B, structural descent row D, output row O, and audit row A; the transport and readback ledger is the componentwise hsame *row H, displayed* Cont *route C, and* Pkg *provenance row P; and the boundary-certificate ledger is the ground-boundary row G together with the local* NameCert *row N. Hence an accepted downstream consumer has no coordinate for an unauthorized signature, unchecked branch, missing descent row, absent audit row, private continuation route, off-packet package, hidden boundary, external naming certificate, detached generator route, host recursion oracle, or hidden substitution channel.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputRouteDownstreamCove

Proof. Start with the readiness side. By Definition 525.160, the output row O is route-ready only inside the displayed BHist packet $I, E, M, B, D, O, A, H, C, P, G, N$. That theorem also identifies the rows that must be present before output exposure is admitted: the accepted signature, emitted eliminator, checked motive, authorized branch, structural descent, audit, hsame transport, Cont route, Pkg provenance, ground-boundary, and local NameCert rows. Thus any accepted local consumer reading O from the readiness surface already factors through the route alternatives I, E, M, B, D, O, A, the transport and provenance rows H, C, P, and the boundary-certificate rows G, N.

Apply Definition 525.161 to the same consumer-facing read. Once the read reaches the boundary ledger, totality requires the same rows $I, E, M, B, D, O, A, H, C, P, G, N$ to be preserved as displayed rows of one BHist packet. In particular, the hsame, Cont, Pkg, and NameCert support cannot be detached from the accepted signature, branch, descent, audit, route, package, boundary, and local naming surface.

Combining these two constraints leaves no further accepted coordinate. Readiness admits O only through the displayed packet, and boundary-ledger totality says every ledger read preserving that output remains on exactly that packet. Therefore every accepted downstream consumer is covered by the listed route alternatives, transport and readback ledger, or boundary-certificate ledger, while all unauthorized signatures, unchecked branches, missing descent or audit rows, private continuation routes, off-packet packages, hidden boundaries, external naming certificates, detached generator routes, host recursion oracles, and hidden substitution channels remain outside the AuthorizedGeneratorRecursor$^{\uparrow}$ packet. □

Theorem 525.164 (AuthorizedGeneratorRecursor root output consumer triad)**.** *Every root output consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet receives the output row only through the finite* BHist *route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route displays the accepted authorization surface I, E, M, B, D*, the generator state witnessed by the output row* O*, the audit row* A*, the componentwise* hsame *row* H*, the displayed* Cont *route* C*, the* Pkg *provenance row* P*, the ground-boundary row* G*, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N*.*

Thus a root consumer has no hidden generator authority, host recursion oracle, off-packet output row, detached readiness row, private continuation, external package source, or outside naming certificate.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root output consumer triad`

Proof. Begin with route-pack exactness, Definition 525.56. It makes $I, E, M, B, D, O, A, H, C, P, G, N$ the complete packet available to a recursor route.

Use the downstream consumer ledger cover, Definition 525.58. It states that generator, substitution-boundary, and TasteGate consumers receive only the carrier, branch, descent, audit, transport, continuation, package, boundary, and local naming rows displayed by the packet.

Finally apply output route readiness, Definition 525.160. The output row O is route-ready only inside the same finite packet, so a root consumer cannot replace the displayed authorization, readiness, continuation, package, ground-boundary, or local naming rows by hidden generator authority or host recursion. □

Theorem 525.165 (AuthorizedGeneratorRecursor readiness route totality)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the readiness route from authorization to output is total exactly on the displayed rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The accepted constructor signature, emitted eliminator, motive, branch package, and structural descent rows provide authorization; the output row and audit row provide generator-state exposure and refusal evidence; the hsame*,* Cont*,* Pkg*, ground-boundary, and local* NameCert *rows provide the only admissible route support.*

No consumer may complete the readiness route by importing an unlisted branch, non-structural recursive call, hidden generator state, host recursion rule, private output channel, or off-packet naming witness.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessRouteTotality`

Proof. Start from the readiness route obligation packet of Definition 525.148. It repackages the accepted carrier rows as authorization, generator, ground-recognition, output, refusal, transport, replay, provenance, and local naming data.

Apply ground compiler admission, Definition 525.149. The generator output reaches any ground-compiler or output-facing consumer only through the authorization row and ground-boundary recognition carried by the same finite packet.

Use NameCert obligation exhaustion, Definition 525.151. The carrier, classifier, stability, ledger, and non-escape obligations are exhausted by these readiness rows. Therefore the route is total on the displayed packet and cannot be completed by hidden generator state, host recursion, private output exposure, or an external naming row. □

Theorem 525.166 (AuthorizedGeneratorRecursor output route naming exhaustion)**.** *Every accepted output route naming surface for* AuthorizedGeneratorRecursor$^{\uparrow}$ *is exhausted by the same finite route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The public name of O must retain the authorization rows, generator-state row, readiness and audit rows, displayed Cont *route,* Pkg *source, and local* NameCert *row already present in the packet.*

Consequently the output route cannot be named through a hidden generator, a host-recursive eliminator, an external continuation, an unaudited package source, a missing readiness row, or a naming certificate outside the displayed BHist *carrier.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier output route naming exhaustion`

Proof. Project the root output consumer triad above. Any root consumer reaches O only after the authorization, generator-state, audit, transport, continuation, package, boundary, and local naming rows are displayed.

Apply readiness route totality. Since the readiness route is total only on $I, E, M, B, D, O, A, H, C, P, G, N$, an output name cannot be certified by a partial route missing authorization, audit, continuation, package, boundary, or local NameCert support.

Finally use downstream readback exhaustion, Definition 525.162. It removes detached generator routes, host recursion oracles, hidden substitution channels, unlicensed output rows, and external naming witnesses from downstream reads. Hence the displayed finite carrier exhausts the output route naming surface. □

Lemma 525.167 (AuthorizedGeneratorRecursor root output consumer downstream coverage)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root output packet with displayed route*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every accepted local consumer-facing use of the root output row O is exhausted by the finite downstream surfaces already displayed in R. More precisely, the use must factor through the authorization and branch alternatives I, E, M, B, D, the output and audit ledger rows O, A, the transport, continuation, package, boundary, and local naming rows H, C, P, G, N, or the coverage row obtained by reading readiness and output-route naming through the same carrier.

Consequently a request for an unlisted branch, non-structural recursive call, hidden generator state, host recursion rule, private output channel, external continuation, unaudited package source, missing readiness row, or naming certificate outside the displayed BHist *packet is not an accepted local consumer of the* AuthorizedGeneratorRecursor$^{\uparrow}$ *root output.*

Proof. Start with readiness route totality, Definition 525.165. It states that the route from authorization to output is total exactly on

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Thus a consumer-facing use of O cannot begin from a free generator surface. Its admission side is forced through the accepted constructor signature, emitted eliminator, motive, branch package, and structural descent rows I, E, M, B, D, and its output side is forced to retain O, A, H, C, P, G, N as route support in the same packet.

Next use output route naming exhaustion, Definition 525.166. Any public name of the output route must keep the authorization rows, generator-state row, readiness and audit rows, displayed

Cont route, Pkg source, and local NameCert row already present in R. Hence naming O supplies no fourth downstream surface: it only records the same finite carrier as a named route and keeps the local consumer inside the displayed BHist, hsame, Cont, Pkg, boundary, and NameCert coordinates.

Combine these two constraints. If a proposed local consumer is accepted, readiness totality places the read on the displayed authorization-to-output route, while naming exhaustion keeps the public name of that read on the same carrier. The consumer therefore lands on one of the finite alternatives, ledger rows, or carrier-coverage rows listed in the statement.

It remains to exclude the apparent alternatives. An unlisted branch or non-structural recursive call would bypass I, E, M, B, D; hidden generator state or a host recursion rule would add a generator coordinate not present in the readiness route; a private output channel would detach O from A, H, C, P, G, N; an external continuation or unaudited package source would replace C or P; a missing readiness row would contradict route totality; and an outside naming certificate would contradict output route naming exhaustion. None of these requests has a coordinate in R, so each remains outside the accepted AuthorizedGeneratorRecursor$^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputConsumerDownstr`

Theorem 525.168 (AuthorizedGeneratorRecursor readiness determinacy)**.** *Let Q and Q' be accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness routes read from the same admitted carrier rows. If they agree on the authorization row U, generator row J, ground-compiler recognition row K, refusal ledger L, componentwise transport H, continuation replay C, provenance row P, and local name row N, then they expose the same public output row O. Thus the readiness packet has no hidden branch, detached generator, or off-packet substitution channel from which a second public output can be read.*

Proof. Start from carrier admission. Definition 525.144 exposes the accepted route only through the finite rows $I, E, M, B, D, O, A, H, C, P, G, N$, and Definition 525.148 repackages those rows as $Q = (U, J, K, O, L, H, C, P, N)$.

The shared rows determine the authorization side. Agreement on U and J fixes the accepted signature, emitted eliminator, checked motive, branch package, structural descent, and authorized generator call. The ground-compiler admission theorem keeps any compiler-facing read of the output on the same U and K rows.

It remains to rule out a second output route. The output refusal theorem blocks any request missing the refusal ledger L, while the root output route saturation theorem exposes public output only through the displayed finite carrier. NameCert obligation exhaustion then packages the carrier, classifier, stability, ledger, and non-escape obligations in the same readiness route, so two such routes must expose the same O. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessDeterminacy`

Theorem 525.169 (AuthorizedGeneratorRecursor closed substitution admission)**.** *Every closed-substitution consumer of the public output row O of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route retains the branch package B, structural descent row D, audit row A, boundary row G, and refusal ledger L. A route that omits one of those rows remains* GAP*-ledger content and is not admitted as a closed-substitution handoff.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorClosedSubstitutionAdmissi`

Proof. First read the closed-substitution handoff. Definition 525.86 sends O to closed-substitution and generator-audit consumers only through the finite route $I, E, M, B, D, O, A, H, C, P, G, N$.

The branch, descent, audit, and boundary rows are therefore not optional. The carrier admission theorem supplies those rows before the output is public, and the readiness route definition records their authorization, refusal, transport, replay, provenance, and local naming projections inside Q.

Now apply output refusal. By Definition 525.150, any request that omits the authorization row or the refusal ledger remains at the GAP boundary. Since B, D, A, G are the carrier coordinates through which U, K, L are read by closed-substitution consumers, a route missing one of them cannot be the admitted handoff. □

Theorem 525.170 (AuthorizedGeneratorRecursor public output coverage)**.** *Normalization consumers, closed-substitution consumers, and ground-compiler public consumers of an accepted* AuthorizedGeneratorRe *output all factor through the readiness packet*

$$Q = (U, J, K, O, L, H, C, P, N).$$

No public consumer reads O through a route outside that packet.

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPublicOutputCoverage

Proof. Begin with public output saturation. Definition 525.157 says that every public output route factors through $I, E, M, B, D, O, A, H, C, P, G, N$, and the readiness route definition records exactly the authorization, generator, compiler-recognition, output, refusal, transport, replay, provenance, and local naming view of those rows.

Closed-substitution consumers are covered by the admission theorem above and by the closed-substitution handoff. They retain the branch, descent, audit, boundary, refusal, transport, continuation, provenance, and naming rows before reading O.

Ground-compiler consumers are covered by ground compiler admission. Normalization consumers use the same public output row because NameCert obligation exhaustion leaves no extra classifier, stability, ledger, or non-escape obligation outside Q. Hence all three public consumer families factor through the readiness packet. □

Corollary 525.171 (AuthorizedGeneratorRecursor readiness consumer determinacy)**.** *Fix the admitted local rows of an* AuthorizedGeneratorRecursor$^{\uparrow}$ *readiness route*

$$Q = (U, J, K, O, L, H, C, P, N).$$

*Every accepted consumer-facing use of the public output that is admitted as a closed-substitution handoff, a normalization read, or a ground-compiler read is determined by those rows. More precisely, once the authorization row U, generator row J, ground-compiler recognition row K, refusal ledger L, componentwise transport row H, continuation replay row C, provenance row P, and local name row N are fixed, the consumer must use the same finite readiness route, the same public normal-output row O, and the same admission-versus-*GAP *decision supplied by the chapter. There is no accepted alternate coordinate, duplicate classifier source, detached branch package, or hidden substitution channel from which a second consumer result can be read.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorReadinessConsumerDetermin

Proof. Read the fixed readiness route through Definition 525.148. The row U packages the accepted signature, emitted eliminator, checked motive, complete branch package, and structural descent data; J is the authorized recursor call; K is the ground-compiler recognition row; L is the refusal ledger; and H, C, P, N are respectively the displayed transport, replay, provenance, and

local naming rows. Thus the named consumer interface already exposes the finite route on which public output can be read.

Let a consumer-facing use be accepted after these rows have been fixed. If it is a closed-substitution handoff, Definition 525.169 requires it to retain the branch package B, structural descent row D, audit row A, boundary row G, and refusal ledger L. Since B and D are read through U, G is read through K, A is recorded by L, and the handoff also keeps the displayed transport, replay, provenance, and local naming rows, the consumer determines an accepted readiness route with the same U, J, K, L, H, C, P, N as Q. A normalization or ground-compiler read has the same row support by Definition 525.170: it factors through the readiness packet rather than through a detached generator or off-packet substitution rule.

Apply Definition 525.168 to Q and to the readiness route just obtained from the accepted consumer use. The two routes agree on U, J, K, L, H, C, P, N, so they expose the same public output row O. Hence a normalizing consumer cannot choose a second normal-output row, and a ground-compiler or closed-substitution consumer cannot cite an alternate output coordinate after the displayed local rows have been fixed.

It remains to identify the decision boundary. By Definition 525.169, a purported handoff omitting B, D, A, G, or L remains GAP-ledger content and is not admitted as a closed-substitution handoff. By the determinacy theorem, a purported handoff retaining the fixed rows but exposing a different output would have to be a second readiness route over the same U, J, K, L, H, C, P, N, which is impossible. Therefore every accepted consumer use makes the same finite-route, normal-output, and admission-versus-GAP decision named by the readiness packet, and every alternate coordinate or duplicate classifier source is rejected before it becomes a public consumer result. □

Theorem 525.172 (AuthorizedGeneratorRecursor admission exhaustion)**.** *Every accepted* AuthorizedGeneratorRec *root packet is exhausted by the displayed* BHist *carrier rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

together with the readiness view

$$U, J, K, O, L, H, C, P, N.$$

An output-facing read is admitted only after the accepted signature, emitted recursor, checked motive, branch package, structural descent, audit row, boundary row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, refusal ledger, and local* NameCert *row have all been retained. No route into chapter 538, chapter 892, or chapter 534 may read a detached generator output outside those rows.*

Proof. Start with carrier admission and root carrier totality. Theorems 525.144 and 525.153 display the admitted root packet and leave no carrier coordinate outside $I, E, M, B, D, O, A, H, C, P, G, N$.

Next pass to the readiness view. The definition of the readiness route, Definition 525.148, repackages the same admitted data as $U, J, K, O, L, H, C, P, N$, while Definition 525.151 keeps carrier, classifier, stability, ledger, and non-escape obligations on that packet.

Finally attach the sibling consumers. The closed-substitution gate, terminal consumer readiness, and root output route saturation theorems force every closed-substitution, MetaCIC normalization frontier, and recursor-generator read to retain the boundary, replay, provenance, refusal, and local naming rows before reading O. Thus admission is exhausted by the two displayed finite views. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAdmissionExhaustion`

Theorem 525.173 (AuthorizedGeneratorRecursor route refusal)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *consumer request a generator output not listed in the authorized signature, branch, descent, and audit rows of the packet. Then the request remains in the refusal and boundary rows L, G, A, N and is not admitted as an output row O. In particular, no downstream closed-substitution, MetaCIC normalization frontier, or recursor-generator consumer may reinterpret an unlisted generator result as an authorized branch, recursive call, host induction principle, proof producer, or off-packet substitution channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRouteRefusal`

Proof. Read the authorization side first. The accepted signature route, branch budget exhaustion, and root signature totality theorems place every listed generator route behind I, E, M, B, D and the audit row A.

Now apply output refusal. Theorem 525.150 sends any request missing the authorization row or refusal ledger to the GAP boundary, and Definition 525.137 keeps ground-facing refusals on the displayed packet.

It remains to block downstream reinterpretation. The closed-substitution budget lock, MetaCIC normalization boundary route, and RecursorGenerator authorization obligations only consume output through the displayed finite route; terminal reader nonescape then removes hidden generator state, host evaluation, unchecked recursion, proof-producing branches, and off-packet substitution channels. □

Theorem 525.174 (AuthorizedGeneratorRecursor downstream NameCert package exactness)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the downstream NameCert package is exactly the finite package consisting of the carrier rows $I, E, M, B, D, O, A, H, C, P, G, N$, the readiness classifier rows $U, J, K, O, L, H, C, P, N$, displayed* Cont *replay through C,* Pkg *provenance through P, and local* NameCert *rows through N. Downstream closed-substitution, MetaCIC normalization frontier, and recursor-generator consumers may specialize this package, but may not add a carrier, classifier, continuation, provenance, or naming row outside it.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorDownstreamNameCertPackage`

Proof. Begin with the downstream consumer surface. The root consumer factorization and public output coverage theorems force public output consumers to factor through the displayed carrier and readiness packet before a specialization is read.

Next identify the structural rows. Componentwise transport is carried by H, replay by C, provenance by P, and local naming by N; the NameCert obligation exhaustion theorem states that these rows discharge the carrier, classifier, stability, refusal-ledger, and non-escape clauses for the same packet.

Finally use the sibling locks. Closed-substitution handoff keeps substitution consumers inside the boundary rows, MetaCIC normalization frontier reads the output only as finite boundary data, and RecursorGenerator consumes the result as authorized generator provenance rather than as a second generator. Therefore the downstream NameCert package is exact. □

Corollary 525.175 (AuthorizedGeneratorRecursor admission exhaustion strict obstruction)**.** *Let an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *root packet be read through the admission-exhaustion surface. If a displayed failure, refusal, separation, or countermodel row is cited against a converse or reuse route for that packet, then the obstruction is strict local data. It blocks only the named consumer request whose generator output is not listed in the authorized signature, branch, descent, and audit rows, so that the rejected request remains in the refusal and boundary rows L, G, A, N*

and is not admitted as an output row O. The obstruction does not state a global impossibility for AuthorizedGeneratorRecursor$^{\uparrow}$*, does not supply a host theorem about recursors, and does not validate or invalidate any hidden external route.*

Proof. By Definition 525.172, every accepted root packet is exhausted by the displayed carrier rows

$$I, E, M, B, D, O, A, H, C, P, G, N$$

together with the readiness view

$$U, J, K, O, L, H, C, P, N.$$

The same theorem also states that any output-facing read is admitted only after the accepted signature, emitted recursor, checked motive, branch package, structural descent, audit row, boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal ledger, and local NameCert row have all been retained. Hence every consumer-facing use factors through the shown finite packet, and a proposed obstruction has no coordinate outside those rows from which to speak about a global theory, a host recursor theorem, or an external validator.

Now apply Definition 525.173. A consumer request for a generator output not listed in the authorized signature, branch, descent, and audit rows is not promoted to O; it remains in L, G, A, N. The same refusal theorem blocks downstream reinterpretation of that unlisted result as an authorized branch, recursive call, host induction principle, proof producer, or off-packet substitution channel. Combining this refusal with admission exhaustion fixes the exact scope of any displayed failure, refusal, separation, or countermodel row: it marks the named converse or reuse route as leaving the displayed authorization packet, while all listed reads that retain the carrier, readiness, boundary, replay, provenance, refusal, and local naming rows remain governed by the accepted packet.

Therefore the obstruction is strict and local. It rejects only the unlisted consumer coordinate identified by the refusal row, and it cannot be reinterpreted as a global impossibility, a host theorem, or a hidden external validation. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAdmissionExhaustionStrict`

Theorem 525.176 (AuthorizedGeneratorRecursor root-admission public export)**.** *The admission exhaustion theorem, route refusal theorem, downstream* NameCert *package theorem, and strict obstruction corollary jointly export the finite MetaCIC-facing root-admission boundary for* AuthorizedGeneratorRecurso *Every authorized-generator recursor consumer must expose closed-substitution boundary rows, operational membrane rows,* Real$^{\uparrow}$/RegSeqRat$^{\uparrow}$/StreamName$^{\uparrow}$ *dependency rows, and recursor-generator rows before requesting a root admission. No consumer may request root admission from an unlisted generator output, host recursor theorem, off-packet substitution channel, or hidden external route.*

Proof. Start with admission exhaustion. Theorem 525.172 requires every output-facing read to retain the accepted signature, emitted recursor, checked motive, branch package, structural descent, audit row, boundary row, componentwise transport, replay, provenance, refusal ledger, and local naming row.

Apply route refusal. Theorem 525.173 keeps unlisted generator outputs in the refusal and boundary rows, so a consumer cannot treat an unauthorized result as an authorized branch, recursive call, host induction principle, proof producer, or off-packet substitution channel.

Use the downstream NameCert package theorem. It identifies the finite package consumed by closed-substitution, MetaCIC normalization frontier, and recursor-generator consumers, including the carrier rows, readiness rows, displayed Cont replay, Pkg provenance, and local NameCert rows.

Finally apply the strict obstruction corollary. Any displayed obstruction blocks only the named request that leaves the authorization packet; it does not become a global impossibility or a host theorem. Thus a MetaCIC-facing root-admission consumer must display the closed-substitution boundary, operational membrane, real, regular-rational, stream-name, and recursor-generator dependency rows before the admission request is accepted. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootAdmissionPublicExport`

Theorem 525.177 (AuthorizedGeneratorRecursor output-route determinacy)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

any accepted output read of O is determined by the displayed signature row I, eliminator row E, motive row M, branch row B, descent row D, audit row A, route row C, package row P, boundary row G, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N. There is no accepted output route for O detached from those rows, and no downstream consumer may recover a second output meaning from the packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier output route determinacy`

Proof. Begin with the branch and descent rows. Theorems 525.2 and 525.5 force the output route to keep the accepted constructor signature, emitted eliminator, checked motive, complete branch package, and displayed structural descent.

Next attach audit and package provenance. Theorem 525.6 retains A, H, C, P, G, N before O is exposed, while Definition 525.7 identifies the TasteGate handoff only as carrier alignment rather than as another generator route.

Finally apply admission exhaustion, Definition 525.172. It exhausts accepted output reads by the finite packet R and its readiness view. Since neither view has a coordinate for a detached output channel, the route to O is determined by the displayed rows. □

Theorem 525.178 (AuthorizedGeneratorRecursor closed-substitution consumer lock)**.** *Every downstream consumer passing an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet to chapter 527 or chapter 538 must retain the authorized output route of Definition 525.177. The consumer may read O only through*

$$I, E, M, B, D, O, A, H, C, P, G, N,$$

and cannot reinterpret closed substitution as an off-packet generator result, an unrestricted substitution calculus, or a proof-producing branch outside the authorized recursor packet.

Proof. Project the output route determinacy theorem. It gives the closed-substitution consumer the output row only as part of the accepted signature, eliminator, motive, branch, descent, audit, route, package, boundary, and local naming rows.

Read the closed-substitution chapters as consumers, not as producers of the recursor output. Their boundary surface can specialize finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data, but it receives no coordinate from which to manufacture a generator result outside R.

The exclusion follows by row projection. If a consumer tries to interpret substitution as an off-packet generator result, the interpretation would need a second output route or branch source. The packet has neither coordinate, and admission exhaustion keeps such requests on the displayed boundary and refusal rows. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier closed substitution consumer lock

Theorem 525.179 (AuthorizedGeneratorRecursor normalization-frontier consumer lock)**.** *Every route from an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet into chapter 892 and chapter 534 consumes the output row* O *only through the same finite packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Such a route cannot add a second generator, a hidden induction principle, or a proof-producing branch source beyond the authorized signature, branch, descent, audit, route, package, boundary, and local NameCert *rows.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorNormalizationFrontierCons

Proof. First retain the authorized route. The output-route determinacy theorem fixes O behind $I, E, M, B, D, A, C, P, G, N$, with H carrying componentwise transport across the same rows.

Next pass to the normalization-frontier and recursor-generator consumers. They may consume the named output as finite boundary data and generator provenance, but only through the continuation and package rows already displayed in the accepted packet.

It remains to rule out hidden producers. A second generator, hidden induction principle, or proof-producing branch would require an additional constructor, eliminator, branch, descent, or output coordinate. Branch exhaustion, structural-descent locality, audit provenance factorization, and admission exhaustion leave no such coordinate, so the consumers remain locked to the finite packet. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier normalization frontier consumer lock

Corollary 525.180 (AuthorizedGeneratorRecursor output-route strict local obstruction)**.** *Let an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be consumed by a normalization-frontier or recursor-generator downstream route as in Definition 525.179. If that consumer displays a failure, refusal, separation, or countermodel row against a converse output reading, a reuse of the recursor output, or a second generator interpretation, then the displayed row obstructs only that named consumer-facing route through

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

It cannot be reinterpreted as a global impossibility theorem, a host theorem about all recursors, a hidden external validation of the failed route, or a second meaning of O *outside the authorized packet.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputRouteStrictLocalObs

Proof. Apply Definition 525.177 first. Any accepted read of O is determined by the displayed signature, eliminator, motive, branch, descent, audit, route, package, boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecurso}}$ rows. In particular, there is no accepted output route detached from these rows and no downstream consumer may recover a second output meaning from the packet.

Now pass to the normalization-frontier and recursor-generator consumers. By Definition 525.179, every such route consumes O only through the same finite packet and cannot add a second generator, hidden induction principle, or proof-producing branch source beyond the displayed rows.

Consider a displayed failure, refusal, separation, or countermodel row in such a consumer. For the row to bear on the authorized output, the two cited theorems force its consumer-facing use to factor through the named packet coordinates. Therefore the row can block the advertised converse reading, reuse route, or second-generator interpretation only at those coordinates. Reading it as a global impossibility would require a theorem coordinate outside the packet; reading it as a host theorem would require a host-facing induction or proof row outside the packet; reading it as hidden external validation would require an external certificate route outside the packet. The determinacy theorem and the consumer lock exclude all three additional coordinates, so the obstruction remains strict and local to the named route. □

Theorem 525.181 (AuthorizedGeneratorRecursor boundary audit exhaustion)**.** *Every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet that exposes the boundary row G must retain the audit row A, componentwise transport row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N before any output row O is public. The retained root-boundary packet is exactly the finite* BHist *surface*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Thus an output-facing consumer may reach G only through the same audited carrier, not through a detached boundary witness or a public output name alone.

Proof. Begin with the carrier definition Definition 525.1. It places G in one finite BHist packet with O, A, H, C, P and the local certificate row N, and it gives no separate coordinate for a boundary witness outside that packet.

Apply the local surface theorem Definition 525.3. It allows a downstream consumer to treat O as recursor-generated only through the displayed packet, while Definition 525.2 and Definition 525.5 keep the signature, branch package, and descent rows attached to that read.

Finally use the root-unblock rows already available in this chapter. Theorems 525.128 and 525.129 force every output-facing route through $I, E, M, B, D, O, A, H, C, P, G, N$, and Definition 525.130 excludes a hidden authorization route. Therefore the audit, transport, continuation, provenance, and local naming rows are retained before O becomes public. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBoundaryAuditExhaustion`

Theorem 525.182 (AuthorizedGeneratorRecursor output provenance coverage)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, public provenance for the output row O factors through exactly the displayed rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The provenance cannot be replaced by an unaudited generator route, an off-packet Pkg *witness, or a local name that omits the audit row A and boundary row G.*

Proof. Project the finite carrier of Definition 525.1. The output row O is listed with the constructor signature, emitted eliminator, motive, branch package, descent row, audit row, transport row, continuation row, package provenance, boundary row, and local certificate.

Use Definition 525.3 to read this list as the local NameCert surface. Branch coverage and structural descent are not optional provenance side data: by Definition 525.2 and Definition 525.5, they remain part of every public recursive read.

Now attach the root-boundary audit exhaustion just proved. It keeps A, H, C, P, G, N on the same route before O is public. The root-unblock route totality theorem Definition 525.129 supplies the matching output-facing path, and Definition 525.130 rules out a ledger export of an unaudited generator route.

Since the displayed tuple has no coordinate for an off-packet package witness, hidden generator, or local name without audit provenance, the output provenance is covered exactly by $I, E, M, B, D, O, A, H, C, P, G, N$. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOutputProvenanceCoverage`

Theorem 525.183 (AuthorizedGeneratorRecursor recursion-channel refusal)**.** *Any attempted use of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet as a non-structural recursion channel, missing-branch repair, detached substitution step, or hidden proof-producing generator is routed to the displayed boundary row G and the audited rows A, H, C, P, N rather than exported as a public output. In particular, the packet exports only the structural recursor route witnessed by $I, E, M, B, D, O, A, H, C, P, G, N$.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRecursionChannelRefusal`

Proof. Start with branch exhaustion and structural descent locality. Theorems 525.2 and 525.5 leave no public branch outside B and no recursive call whose descent is absent from D.

Next read the carrier and NameCert surface. By Definition 525.1 and Definition 525.3, the boundary row G is the displayed place where unlisted branches, non-structural recursion, external axioms, and hidden proof generators are refused.

Detached substitution is handled by the root-unblock packet rows. Theorems 525.129 and 525.130 force output-facing and ledger-facing reads to stay inside the same BHist, Cont, Pkg, and $\mathsf{NameCert}$ rows, so a substitution step cannot become an off-packet generator.

Finally apply boundary audit exhaustion and output provenance coverage above. They keep A, H, C, P, G, N attached before O is public. Therefore every rejected recursion channel is recorded at the displayed boundary rather than exported as a public proof-producing route. □

Corollary 525.184 (AuthorizedGeneratorRecursor boundary audit strict obstruction)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the obstruction recorded by the boundary-audit exhaustion and recursion-channel refusal is strictly local. It blocks only the attempted non-structural recursion channel, missing-branch repair, detached substitution step, or hidden proof-producing generator named in Definition 525.183. Its consumer-facing use must factor through the displayed packet*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Consequently the same obstruction cannot be reread as a global impossibility theorem, a host theorem, or an external validation route for any coordinate not displayed in that packet.

Proof. Apply Definition 525.181. It identifies the output-facing boundary audit with the finite BHist surface $I, E, M, B, D, O, A, H, C, P, G, N$ and says that a consumer may reach the boundary row G only through the audited carrier retaining A, H, C, P and the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row N. Thus the boundary row is not a detached witness and is not available through the public output name alone.

Now apply Definition 525.183. The only failed uses routed to G are the named attempts: non-structural recursion, missing-branch repair, detached substitution, and hidden proof production. The theorem sends those attempts back to the audited rows A, H, C, P, N and states that the packet exports only the structural recursor route witnessed by the same displayed rows.

These two statements exhaust the consumer-facing cases. A listed failed route factors through G together with A, H, C, P, N, so it is blocked as a local misuse of the accepted recursor packet. An unlisted use would require an additional public coordinate, such as a global impossibility certificate, host theorem row, or external validation row. The carrier displayed in Definition 525.181 has no such coordinate, and Definition 525.183 exports no route besides the structural recursor route. Therefore the obstruction is exactly the local obstruction stated above. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBoundaryAuditStrictObstru`

Theorem 525.185 (Authorized generator recursor branch descent audit totality)**.** *Every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet reads the branch package B, descent row D, output row O, and audit row A only through the displayed carrier*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The branch, descent, output, and audit coordinates therefore form one root boundary: a consumer cannot read a branch package without the matching descent row, cannot read an output row outside the audit row, and cannot replace the displayed carrier by host induction, an external axiom route, an unaudited dependency route, or an off-packet substitution.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor branch descent audit totality`

Proof. Begin with the carrier, Definition 525.1. It displays exactly the input, authorization, measure, branch, descent, output, audit, transport, continuation, provenance, ground-compiler, and local naming rows.

Apply carrier admission, Definition 525.34. Admission exposes B, D, O, A as rows of the same accepted packet, not as detachable host data.

Now use root boundary ledger totality, Definition 525.74. It forces every public read of the root boundary through the audit ledger attached to the displayed route.

Finally apply the ground-compiler handoff, Definition 525.7. The handoff keeps branch choice, descent evidence, output naming, and audit replay inside $I, E, M, B, D, O, A, H, C, P, G, N$, excluding host induction, external axioms, unaudited dependency routes, and off-packet substitutions. □

Theorem 525.186 (Authorized generator recursor output NameCert non-escape)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, public output from the displayed output row O remains inside the local* $\mathsf{NameCert}$ *row N and the ground-compiler boundary G. It cannot escape into host induction, external axioms, unaudited dependency routes, detached branch packages, off-packet substitutions, or an output certificate not carried by*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor output namecert nonescape`

Proof. Project the accepted carrier of Definition 525.1. The only public output coordinate is O, and it is packaged together with the audit row A, ground-compiler row G, and local NameCert row N.

Use carrier admission, Definition 525.34, to identify the admitted output as a row of the same finite packet. Then apply root boundary ledger totality, Definition 525.74, so every public output read is replayed through the audit boundary rather than through an off-packet dependency route.

The ground-compiler handoff, Definition 525.7, supplies the final boundary. It sends the output read through G and N, so public output cannot be reinterpreted as host induction, external axioms, a detached branch package, an unaudited substitution, or any output certificate outside the displayed carrier. □

Corollary 525.187 (Authorized generator recursor branch descent local obstruction)**.** *For every accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, any displayed failure, refusal, separation, or countermodel row attached to branch descent blocks only the local route that tries to read one of the following coordinates outside the displayed carrier:*

$$B \text{ without } D, \quad O \text{ outside } A, \quad O \text{ outside } G \text{ and } N, \quad \text{or a certificate not carried by } I, E, M, B, D, O, A, H, C, P, G$$

The obstruction cannot be reinterpreted as a global impossibility theorem for recursors, as a host induction theorem, as hidden external validation, or as a rejection of any consumer route that still factors through the displayed branch, descent, output, audit, ground-compiler, and local NameCert *rows.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorBranchDescentLocalObstruc`

Proof. Fix an accepted packet. By Definition 525.185, every consumer-facing read of branch, descent, output, and audit data is a read through the single displayed carrier

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

In particular, B is not available as a detached branch package, D is the matching descent row for that branch package, O is not available outside A, and the carrier cannot be replaced by host induction, an external axiom route, an unaudited dependency route, or an off-packet substitution.

Apply Definition 525.186 to the output side of the same read. Public output from O remains inside the local NameCert row N and the ground-compiler boundary G, and it has no escape route to a detached branch package, an external certificate, host induction, external axioms, unaudited dependencies, or an off-packet substitution.

Therefore the only coordinates on which a displayed obstruction can act are the named local coordinates of the carrier: branch detachment from D, output detachment from A, output escape from G and N, or certificate reuse outside $I, E, M, B, D, O, A, H, C, P, G, N$. Since both cited theorems exhaust consumer-facing reads by those rows, the obstruction has no additional coordinate from which to assert a global impossibility, a host theorem, or hidden external validation. Conversely, a consumer route that still factors through the displayed rows is not blocked by this corollary; the corollary blocks only the named converse or reuse route that leaves the local carrier. □

Theorem 525.188 (Authorized generator recursor root-output totality lock)**.** *Let* $R = (I, E, M, B, D, O, A, H, C,$ *be an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *carrier. Every public exposure of the root output row O factors through the same displayed packet: branch exhaustion in B, structural descent in D, audit provenance in A, componentwise transport in H, displayed* Cont *routes in C,* Pkg *provenance in P, the* GAP *boundary in G, and the local* NameCert *row N.*

The checked TasteGate marker BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTaste *is used only as the single-carrier alignment witness for this packet. It does not expose host induction, an external axiom route, an unaudited dependency route, off-packet substitution, or a second output certificate.*

Proof. Start from the carrier definition, Definition 525.1. It displays the root packet R and leaves no output coordinate outside $I, E, M, B, D, O, A, H, C, P, G, N$.

Apply branch exhaustion and structural descent locality, Definition 525.2 and Definition 525.5. These force every recursive output read to retain the constructor signature, branch package, and descent row before O can be public.

Next attach audit and route provenance. Theorems 525.4 and 525.6 keep A, H, C, P, G, N attached to the output exposure.

Finally use the ground-compiler handoff, Definition 525.7. The TasteGate marker is read only as the aligned-carrier witness, so no hidden generator, host induction principle, external axiom, unaudited route, or off-packet substitution can replace the displayed root-output packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputTotalityLock`

Theorem 525.189 (Authorized generator recursor root audit route exactness)**.** *For the accepted carrier $R = (I, E, M, B, D, O, A, H, C, P, G, N)$, the root audit route is exact: every audit-facing consumer that reads O must also read the branch-exhaustion row B, structural-descent row D, audit-provenance row A, displayed* Cont *route C,* Pkg *row P,* GAP *boundary G, and local* NameCert *row N from the same packet.*

Thus the audit route cannot be reconstructed from output naming alone, and it cannot be replaced by host recursion, a detached proof-producing generator, external kernel equality, or an unaudited dependency channel.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root audit route exactness`

Proof. Project the accepted packet through Definition 525.1. The row A is displayed next to O and the boundary rows H, C, P, G, N, so the audit route begins inside R.

Use the root-output audit totality theorem, Definition 525.46. It places every public output read behind the branch, descent, audit, route, package, boundary, and local naming rows.

The audit provenance factorization, Definition 525.6, then records that the output row becomes public only after the audit provenance and route rows remain attached.

Finally apply the root boundary nonescape and ground-compiler handoff, Definition 525.47 and Definition 525.7. They keep the GAP boundary and TasteGate alignment on the same carrier, excluding the detached audit routes listed in the statement. □

Theorem 525.190 (Authorized generator recursor root consumer readiness surface)**.** *The accepted packet $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ is ready for downstream root consumers precisely as the displayed branch-exhaustion, structural-descent, audit-provenance,* Cont*,* Pkg*,* GAP*, and* NameCert *surface. Any closed-normal, substitution-boundary, generator-audit, or ground-compiler consumer must read O through those rows and through the checked single-carrier TasteGate alignment marker already attached to the chapter.*

The readiness surface exports no unlisted branch, non-structural recursion channel, hidden generator, host induction principle, external axiom route, unaudited dependency route, or output row outside R.

Proof. Begin with downstream consumer factorization, Definition 525.52. It makes every downstream consumer read through the same finite carrier before using the output row.

Next use terminal consumer readiness, Definition 525.97, and downstream NameCert packaging, Definition 525.174. These route closed-normal, substitution-boundary, and generator-audit consumers through the branch, descent, audit, continuation, package, boundary, and local naming rows.

Attach the root-output totality lock above. It ensures that the output row remains the displayed coordinate O of R after branch exhaustion, structural descent, audit provenance, Cont, Pkg, GAP, and NameCert have all been read.

Finally apply the ground-compiler handoff, Definition 525.7. The checked TasteGate marker is consumed only as the single-carrier alignment witness, so the downstream readiness surface contains exactly the rows stated above. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor root consumer readiness surface`

Lemma 525.191 (Authorized generator recursor root-output totality downstream coverage)**.** *Let $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ be an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *carrier. Every accepted local downstream consumer that reads the root output row O is exhausted by the finite surface displayed in the root-output packet: constructor and branch alternatives are read through I, E, M, B, D, audit and output ledgers are read through O, A, H, C, P, and boundary, coverage, and local naming rows are read through G, N together with the displayed* Cont *route C and componentwise transport row H.*

Consequently an unlisted constructor alternative, detached branch table, unchecked motive, nonstructural recursion channel, hidden generator, host induction principle, external axiom route, unaudited dependency route, off-packet substitution authority, second output certificate, or naming certificate outside R remains outside the BEDC packet exposed to that consumer.

Proof. Start with the root-output totality lock, Definition 525.188. It says that every public exposure of O factors through the displayed packet: branch exhaustion in B, structural descent in D, audit provenance in A, componentwise transport in H, displayed Cont routes in C, Pkg provenance in P, the GAP boundary in G, and the local NameCert row N. Since the carrier itself is the finite tuple $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ of Definition 525.1, this first step leaves no output coordinate from which a downstream consumer could obtain a second route to O.

Now apply root audit route exactness, Definition 525.189. Any audit-facing consumer that reads O must also read the branch-exhaustion row B, structural-descent row D, audit-provenance row A, displayed Cont route C, Pkg row P, GAP boundary G, and local NameCert row N from the same packet. Thus the consumer-facing use of O is not merely named by the output row; it is forced to carry the route, transport, package, boundary, and local naming rows that the totality lock lists.

These two constraints exhaust the downstream cases. A constructor or recursive-call demand is covered by the admitted signature, emitted eliminator, motive, branch package, and descent row I, E, M, B, D. An output-ledger or audit demand is covered by O, A, H, C, P. A boundary, refusal, or local-readback demand is covered by G, N together with the same displayed transport and continuation rows. Any proposed consumer fact outside those alternatives would need either a public exposure of O not factored by the totality lock or an audit-facing route not exact in the sense of Definition 525.189. Both possibilities are excluded, so all unlisted alternatives remain outside the BEDC packet. □

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRootOutputTotalityDownstr

The L10 route reads the authorized recursor packet through chapter 126, chapter 301, and chapter 38, while the audit row remains tied to chapter 533.

Theorem 525.192 (AuthorizedGeneratorRecursor L10 source triad)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Any L10-facing finite-source consumer for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *may read the packet only through the displayed rows* $I, E, M, B, D, O, A, H, C, P$ *signature admission in* I*, emitted recursor naming in* E*, motive checking in* M*, branch coverage in* B*, structural descent in* D*, audit provenance in* A*, boundary refusal in* G*, continuation locality in* C*, package provenance in* P*, componentwise* hsame *transport in* H*, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N*. No L10 source route may read the packet as a host choice, a detached induction rule, a selected real witness, a quotient stream, or an external proof producer.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10OpenPhaseSourceTriad

Proof. Project the carrier of Definition 525.1. The displayed packet lists exactly the signature, eliminator, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows available to a consumer.

Use branch exhaustion, Definition 525.2, together with structural descent locality, Definition 525.5. These keep every recursive read on the accepted constructor signature, complete branch package, and displayed descent row before any finite-source route can use the output name.

Attach the output and audit locks. Theorems 525.177, 525.181, and 525.188 keep O with A, H, C, P, G, N on the same finite route.

The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers therefore receive only a BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route over the displayed packet. The carrier has no coordinate for host choice, detached induction, selected real data, quotient stream data, or an external proof producer. □

Theorem 525.193 (AuthorizedGeneratorRecursor consumer handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet must preserve the output row* O *together with* A, H, C, P, G, N*. The consumer may specialize the packet to a finite real-source or completion-source route, but it cannot reinterpret the recursor packet as a host choice principle, hidden induction principle, selected limit, quotient stream, or external proof-producing channel.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOpenPhaseConsumerHandoff

Proof. Start with the L10 source triad above. It gives downstream consumers access to O only through the same finite packet rows, with the audit, transport, continuation, provenance, boundary, and local naming rows still attached.

Read the existing downstream locks. Theorems 525.177, 525.178, and 525.179 treat downstream consumers as readers of the authorized output route rather than as producers of a second generator result.

Now specialize the route to real and completion consumers. A $\mathsf{Real}^{\uparrow}$ or completion-facing route may consume finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ evidence, but the row O remains attached to A, H, C, P, G, N by output provenance coverage and root-output totality.

The excluded interpretations would require a separate host choice coordinate, hidden induction row, selected-limit witness, quotient-stream equality, or proof-producing channel. None is present in the carrier, and the boundary row G records their refusal. □

Theorem 525.194 (AuthorizedGeneratorRecursor ledger non-escape)**.** *For L10-facing use of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet, the audit row A and boundary row G exhaust the ledger exported to consumers. Every consumer route is a finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route over*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

No route exports a completion object, a real-completeness datum, a quotient completion equality, a selected limit, or a host proof of completeness.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorOpenPhaseLedgerNonescape`

Proof. Project the carrier and retain the audit and boundary rows. The definition of the carrier places A and G in the same finite packet as the branch, descent, output, transport, continuation, provenance, and local naming rows.

Use the local boundary results. Theorems 525.4, 525.181, and 525.183 route every rejected generator, recursion, or dependency request to the displayed boundary and audit rows.

Next use root-output totality and output provenance coverage. They keep L10-facing reads inside the same BHist, hsame, Cont, Pkg, and NameCert surface before the output row becomes public.

Thus the ledger exported to StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and completion-facing consumers is finite and local to the packet. Completion objects, real-completeness data, quotient completion equality, selected limits, and host completeness proofs are not rows of R, so they cannot escape from the ledger. □

Theorem 525.195 (AuthorizedGeneratorRecursor L10 readiness closure)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted AuthorizedGeneratorRecursor$^{\uparrow}$ *packet. If the L10 source triad, consumer handoff, and ledger non-escape rows are displayed, then R exports exactly the readiness surface required by* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*, and the MetaCIC normalization boundary: finite reads through $I, E, M, B, D, O, A, H, C, P, G, N$, componentwise* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *transport, and the boundary refusal row G. It exports no global recursion theorem, host induction principle, hidden completion selector, selected real witness, quotient stream equality, or proof-producing channel outside the displayed packet.*

Proof. First cite the L10 source triad, Definition 525.192. It identifies the rows available to StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and completion-facing readers before any consumer may treat the recursor output as a source.

Next apply the consumer handoff theorem, Definition 525.193. It keeps the output row O attached to A, H, C, P, G, N when a Real, completion, or normalization-boundary consumer specializes the finite route.

Finally apply ledger non-escape, Definition 525.194. The audit and boundary rows exhaust the exported ledger, so the readiness surface is exactly the displayed finite packet with its BHist, hsame, Cont, Pkg, and NameCert access. A global recursion theorem, host induction principle, hidden completion selector, selected real witness, quotient stream equality, or external proof-producing channel would require an additional coordinate not present in R. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10ReadinessClosure`

Theorem 525.196 (AuthorizedGeneratorRecursor public L10 source handoff)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet whose L10 source triad, consumer handoff, and ledger non-escape rows are displayed. Then the public L10 source handoff exported to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers is exactly the finite route through* $I, E, M, B, D, O, A, H, C, P, G, N$ *with* BHist, hsame, Cont, Pkg, *and* NameCert *access. The exported handoff cannot be read as host choice, detached induction, selected real witness, quotient stream equality, or a proof-producing channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10PublicSourceHandoff`

Proof. First use Definition 525.192. It identifies the public source rows available to the three L10 consumers and keeps the route inside the accepted packet.

Next use Definition 525.193. It preserves the output row O together with A, H, C, P, G, N when the route is specialized to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$.

Finally apply Definition 525.194 and Definition 525.195. The ledger refusal and readiness closure leave only the displayed BHist, hsame, Cont, Pkg, and NameCert surface, so the excluded host-choice, detached-induction, selected-witness, quotient-equality, and proof-producing readings have no exported coordinate. □

Theorem 525.197 (AuthorizedGeneratorRecursor L10 source-triad public export)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet whose L10 open-phase source triad is exposed to a public consumer. Any accepted authorized-generator recursor consumer of that source triad must factor through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it exposes generator authorization, recursion scheduling, or ledger replay. The exported route remains the finite* BHist, hsame, Cont, Pkg, *and* NameCert *surface over* $I, E, M, B, D, O, A, H, C, P, G, N$ *and cannot reopen host choice, detached induction, selected real witnesses, quotient stream equality, or proof-producing channels.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10SourceTriadPublicExpor`

Proof. First use the L10 source triad, Definition 525.192. It lists the only rows available before a public consumer may read the authorized recursor as a source.

Next apply the consumer handoff, Definition 525.193. It keeps the output row O attached to the audit, transport, continuation, provenance, boundary, and local naming rows when the consumer specializes the route.

Now combine ledger non-escape and readiness closure, Definition 525.194 and Definition 525.195. These results force the public route to pass through the same finite ledger and readiness surface before generator authorization, recursion scheduling, or replay is exposed.

Finally use the public source handoff, Definition 525.196. The handoff identifies the consumer-facing route for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; adding the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance surface only records the finite tolerance row used before the regular and real-facing consumers. No unlisted host-choice, detached-induction, selected-witness, quotient-equality, or proof-producing coordinate can be exported. □

Corollary 525.198 (AuthorizedGeneratorRecursor L10 source-triad strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, and let a* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing L10 consumer display a failure, refusal, separation, or countermodel row against a named converse or reuse route for the source triad. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing read factors through the displayed finite route, transport, and readback rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

with BHist, hsame, Cont, Pkg, *and* NameCert *access. It cannot be reinterpreted as a global impossibility theorem, a host theorem about recursors or real completion, hidden external validation of the failed route, a selected real witness, a quotient stream equality, a completion object, or a proof-producing channel outside the accepted packet.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10SourceTriadObstructio`

Proof. Start with the L10 source triad in Definition 525.192. It identifies the only rows available to an L10-facing finite-source consumer: signature admission I, emitted recursor naming E, motive checking M, branch coverage B, structural descent D, output naming O, audit provenance A, componentwise hsame transport H, continuation locality C, package provenance P, boundary refusal G, and the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row N. The same theorem excludes host choice, detached induction, selected real data, quotient stream data, and external proof production as possible source reads.

Now apply ledger non-escape, Definition 525.194. For L10-facing use, the exported ledger is exhausted by the audit row A and boundary row G, and every consumer route is a finite BHist, hsame, Cont, Pkg, and NameCert route over the displayed packet. The ledger exports no completion object, real-completeness datum, quotient completion equality, selected limit, or host proof of completeness.

Consider the displayed failure, refusal, separation, or countermodel row. If it is used by the named consumer against the authorized source triad, the two cited results force that use to pass through exactly the displayed packet rows and their BHist, hsame, Cont, Pkg, and NameCert readback surface. Thus the row can obstruct only the advertised converse or reuse route at the packet coordinate where that route asks to read, replace, or bypass one of those rows.

Any stronger interpretation asks for an unlisted coordinate. A global impossibility theorem would require a theorem surface beyond the finite packet; a host theorem would require host choice, detached induction, or a host completeness proof; hidden external validation would require an external certificate route; a selected real witness, quotient stream equality, completion object, or proof-producing channel would require one of the objects that the triad and the ledger explicitly refuse. Since none of those coordinates is exported by R, the obstruction remains strict and local to the named L10 source-triad route. □

Theorem 525.199 (AuthorizedGeneratorRecursor RegSeqRat route factorization)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of the recursor output factors through the displayed signature, eliminator, motive, branch, descent, output, audit,*

transport, continuation, provenance, boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows. In particular, the consumer may read* O *only after the route retains* $I, E, M, B, D, A, H, C, P, G, N$ *in the same finite packet, and it obtains no regular-rational source outside that route.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRegSeqRatRouteFactorizati`
Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCarrier regseqrat route factorization`

Proof. Start from the L10 source triad, Definition 525.192. It already identifies the $\mathsf{RegSeqRat}^{\uparrow}$ consumer as a finite-source reader of $I, E, M, B, D, O, A, H, C, P, G, N$ rather than as a separate source producer.

Use the open-phase consumer handoff, Definition 525.193. The handoff keeps O attached to A, H, C, P, G, N, so a regular-rational consumer cannot detach the output from audit, transport, continuation, provenance, boundary, and local naming rows.

Finally apply the ground-compiler authorization row, Definition 525.110. The ground-facing read of O is admitted only after the same finite packet has been retained, which is exactly the required factorization route for $\mathsf{RegSeqRat}^{\uparrow}$ consumption. □

Theorem 525.200 (AuthorizedGeneratorRecursor Real seal non-escape)**.** *For an accepted* AuthorizedGeneratorRec *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the recursor output preserves the same finite route through* $I, E, M, B, D, O, A, H, C, P, G, N$ *The route supplies only displayed* BHist*,* hsame*,* Cont*,* Pkg*, boundary, audit, and local* NameCert *data; it supplies no selected real witness, quotient stream, host equality, detached induction rule, or external proof producer.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRealSealRouteNonescape`

Proof. Read the output route through Definition 525.192. The theorem places $\mathsf{Real}^{\uparrow}$ use on the same displayed packet as $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, with no coordinate for a selected limit or quotient stream.

The consumer side is controlled by Definition 525.193. It allows specialization to a real-source route while keeping O with A, H, C, P, G, N and excluding host choice, hidden induction, selected-limit, quotient-stream, and proof-producing interpretations.

The ledger side is closed by Definition 525.194. Since the audit and boundary rows exhaust what can be exported to the real-facing consumer, the real seal does not escape the AuthorizedGeneratorRecursor packet. □

Theorem 525.201 (AuthorizedGeneratorRecursor completion boundary exhaustion)**.** *Let* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ *be an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. A completion-facing consumer of the recursor output has exactly two admissible boundary shapes: it either displays a separate completion-seal row before using completion data, or it remains bounded by the AuthorizedGeneratorRecursor packet through* $I, E, M, B, D, O, A, H, C, P, G, N$*. Without the separate completion-seal row, the consumer cannot read a completion object, quotient completion equality, selected limit, host completeness proof, or off-packet proof producer.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCompletionBoundaryExhaust`

Proof. Begin with Definition 525.193. Completion-facing use is treated as a downstream specialization of the same finite output route, so O remains attached to A, H, C, P, G, N.

Apply Definition 525.194. The ledger exported by the packet is finite and local to $I, E, M, B, D, O, A, H, C, P, G,$ and it does not export completion objects, real-completeness data, quotient completion equality, selected limits, or host completeness proofs.

If the consumer displays a separate completion-seal row, that row is outside the AuthorizedGeneratorRecursor packet and must carry its own boundary obligation. If it does not, the ground-compiler authorization theorem Definition 525.110 keeps the consumer on the accepted packet surface, exhausting the completion boundary at G, N and the retained audit route. □

Corollary 525.202 (AuthorizedGeneratorRecursor RegSeqRat route strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is used against a named converse, reuse, or readback route for the* $\mathsf{RegSeqRat}^{\uparrow}$ *factorization of* R*. Then the row is a strict local obstruction: it blocks only that named route at the finite packet surface*

$$I, E, M, B, D, O, A, H, C, P, G, N,$$

with readback through the displayed BHist*,* hsame*,* Cont*,* Pkg*, boundary, audit, transport, provenance, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows. It cannot be reinterpreted as a global impossibility theorem, a host theorem about recursors or real completion, hidden external validation of the failed route, a selected real witness, a quotient stream equality, a detached induction rule, a completion object, or an external proof producer.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRegSeqRatRouteStrictObstr`

Proof. Apply the $\mathsf{RegSeqRat}^{\uparrow}$ route factorization theorem, Definition 525.199. It states that every $\mathsf{RegSeqRat}^{\uparrow}$ consumer of the recursor output factors through the displayed signature, eliminator, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows. In particular, the consumer may read O only while retaining $I, E, M, B, D, A, H, C, P, G, N$ in the same finite packet, and it obtains no regular-rational source outside that route.

Now apply the Real seal non-escape theorem, Definition 525.200. The real-facing endpoint preserves the same finite route through $I, E, M, B, D, O, A, H, C, P, G, N$ and supplies only displayed BHist, hsame, Cont, Pkg, boundary, audit, and local NameCert data. It supplies no selected real witness, quotient stream, host equality, detached induction rule, or external proof producer.

Consider the displayed failure, refusal, separation, or countermodel row. If it is cited against a named converse, reuse, or readback route, the first theorem forces that citation to live on the $\mathsf{RegSeqRat}^{\uparrow}$ factorization surface, and the second theorem prevents the same citation from being lifted through a real-facing escape coordinate. Thus the row can obstruct only the advertised route at the coordinate where that route asks to read, reuse, detach, or bypass one of the displayed packet rows.

Any stronger interpretation asks for data that neither theorem exports. A global impossibility theorem would require a theorem surface beyond the accepted finite packet; a host theorem would require host equality, detached induction, or host completeness data; hidden external validation would require a proof-producing channel outside the packet; a selected real witness, quotient stream equality, or completion object would require one of the objects explicitly absent from the real seal. Since the accepted packet exports none of these coordinates, the obstruction remains strict and local to the named $\mathsf{RegSeqRat}^{\uparrow}$ route. □

Theorem 525.203 (AuthorizedGeneratorRecursor Real stream route)**.** *For every accepted* AuthorizedGeneratorRec *packet that is read as an L10 source route, the public generator output may depend on* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *source rows only after those rows are carried through the same finite route*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The route exposes source rows as BHist, hsame, Cont, Pkg, *boundary, and local* NameCert *data, and it exports no selected Real limit, quotient stream equality, host evaluation, or detached generator state.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor real stream route`

Proof. Begin with the closed-substitution handoff of Definition 525.86. It keeps every output-facing consumer inside the displayed packet before the row O is public.

Use the ground-compiler authorization theorem, Definition 525.110. It forces the same retained route for any consumer that compiles the generator output into a ground-facing surface.

Finally apply Definition 525.121. Terminal authorized output is read only from the finite packet rows, so the Real, regular-rational, and stream-name source rows remain packet data rather than external completion objects. □

Theorem 525.204 (AuthorizedGeneratorRecursor substitution membrane lock)**.** *The L10 route of* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *locks closed-substitution and operational-membrane rows before any generator output is public. A consumer may read substitution, branch, descent, audit, transport, continuation, package, boundary, and local naming rows from the packet, but it cannot cross the membrane through unrestricted substitution, unchecked recursion, external proof generation, host equality, or a private audit channel.*

Proof. Read the output boundary through Definition 525.86. The handoff places closed-substitution consumption behind the displayed carrier route rather than behind a general substitution calculus.

Next use Definition 525.110. The operational side of the membrane retains signature, eliminator, motive, branch, descent, audit, transport, continuation, provenance, boundary, and local naming rows together.

The terminal cover theorem, Definition 525.121, closes the membrane at the output row. It leaves no route for unchecked recursion, host equality, external proof generation, or a private audit channel outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor substitution membrane lock`

Theorem 525.205 (AuthorizedGeneratorRecursor generator normalization readiness)**.** *Generator-normalization readiness for the current L10* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet is available only after the Real, regular-rational, stream-name, closed-substitution, operational-membrane, audit, continuation, package, boundary, and local NameCert rows have been retained in one displayed route. The readiness claim does not publicize any output through host normalization, unchecked recursion, selected Real limits, quotient stream equality, or an off-packet generator certificate.*

Proof. Start from Definition 525.86. It makes the output row readable only through the retained closed-substitution and generator-audit route.

Use Definition 525.110 to attach the normalization-facing route to the same accepted signature, emitted recursor, motive, branch, descent, audit, transport, continuation, provenance, boundary, and local naming rows.

Finally apply Definition 525.121. Since terminal output remains covered by the displayed packet, generator normalization is ready only as a packet-local consumer surface and not as a host-normalizer or selected-limit interface. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor generator normalization readiness`

Corollary 525.206 (AuthorizedGeneratorRecursor Real stream route determinacy)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet that is read as an L10* $\mathsf{Real}^{\uparrow}$ *stream route, every accepted consumer-facing use of the route, packet-local normal-form readiness, or exactness decision factors through the same finite packet rows*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

The source side is determined by the $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *rows carried as* BHist*,* hsame*,* Cont*,* Pkg*, boundary, and local* NameCert *data. The normalization side is determined by the retained closed-substitution, operational-membrane, audit, continuation, package, boundary, and local* NameCert *rows of that same packet. Hence two accepted consumers over those displayed rows choose the same finite route and the same packet-local normal-form or exactness decision named by the chapter; there is no alternate hidden coordinate, selected Real limit, quotient stream equality, host normalizer, off-packet generator certificate, or duplicate classifier source.*

Proof. Apply the Real stream route theorem, Definition 525.203. It places every public $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ reading behind the displayed finite route and states that the exported source data are only the packet rows BHist, hsame, Cont, Pkg, boundary, and local NameCert. Thus a consumer-facing Real stream route has no source coordinate outside $I, E, M, B, D, O, A, H, C, P, G, N$.

Now apply generator-normalization readiness, Definition 525.205. Readiness is available only after the Real, regular-rational, stream-name, closed-substitution, operational-membrane, audit, continuation, package, boundary, and local NameCert rows have been retained in one displayed route. Therefore a packet-local normal-form readiness claim or exactness decision is not a second interface: it is the same route read with the normalization-facing rows retained.

The two cited theorems have the same exclusions. The Real stream route exports no selected Real limit, quotient stream equality, host evaluation, or detached generator state, while generator-normalization readiness exports no host normalization, unchecked recursion, selected Real limit, quotient stream equality, or off-packet generator certificate. Any alleged alternate route, hidden coordinate, or duplicate classifier source would have to enter through one of those excluded interfaces or through a packet coordinate not displayed in the finite row list. Since every accepted consumer use has already been forced through the displayed source rows and the displayed normalization rows of the same packet, no such entry point remains.

Consequently the consumer may choose only the finite route already named by the chapter. Its normal-form readiness and exactness decision are packet-local readings of that route, so any two accepted consumers over the fixed displayed rows determine the same route and the same named decision. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRealStreamRouteDeterminac`

Theorem 525.207 (AuthorizedGeneratorRecursor L10 source-row admission)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Every L10-facing accepted output packet exposes its* $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows only as displayed finite* BHist *rows inside the authorized output route. The source rows are visible only together with the displayed authorization, branch, descent, audit, componentwise* hsame, Cont, Pkg, *boundary, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows of* R.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10RealStreamOutputFactor`
Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor l10 source row admission`

Proof. Project the accepted carrier through Definition 525.192. That theorem already identifies $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source use as finite packet access rather than as host source selection.

Next retain the route side. Theorems 525.199 and 525.203 keep regular-rational, stream-name, and Real reads on the displayed packet $I, E, M, B, D, O, A, H, C, P, G, N$ before the output is public.

Finally attach the handoff rows. The closed-substitution handoff Definition 525.86 and terminal-output readback Definition 525.225 preserve A, H, C, P, G, N with the output row. Thus source admission is exactly admission through the displayed finite BHist route with hsame, Cont, Pkg, boundary, and local $\mathsf{NameCert}$ evidence attached. □

Theorem 525.208 (AuthorizedGeneratorRecursor L10 Real-stream output factorization)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$, *each* $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of the output row* O *factors through the already displayed authorization, branch, descent, audit, componentwise* hsame, Cont, Pkg, *boundary, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows. No such consumer receives an output route that omits* $I, E, M, B, D, A, H, C, P, G, N$ *while retaining* O.

Proof. Begin with L10 source-row admission above. It fixes the consumer's input surface as the finite displayed packet, so a Real, stream-name, or regular-rational consumer begins from R rather than from a detached output token.

Use the route-factorization layer next. Theorems 525.199, 525.200, and 525.221 keep O attached to the source, authorization, audit, transport, continuation, provenance, boundary, and local naming rows.

Now pass to terminal readback. Theorem 525.225 requires every terminal read by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ to expose the same BHist packet, replay through C, preserve H, retain P, and name the read by N.

Therefore the consumer map factors through the displayed BEDC route. Since the carrier has no coordinate for a bare public output token, every consumer that reaches O has already retained the named rows of R. □

Theorem 525.209 (AuthorizedGeneratorRecursor L10 completion-boundary nonescape)**.** *Let* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ *be an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. No* $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or completion-facing consumer of* R *can recover a selected Real limit, quotient stream equality, host evaluation, detached generator state, ambient completion principle, or off-packet proof channel from the packet. Every such read remains bounded by the displayed* BHist, hsame, Cont, Pkg, *boundary, audit, and local* $\mathsf{NameCert}$ *route.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10CompletionBoundaryNone`

Proof. First use L10 source-row admission and L10 Real-stream output factorization. Together they put every Real, stream-name, regular-rational, or completion-facing read on the displayed packet before any output row can be consumed.

Next read the boundary facts. Theorems 525.201 and 525.194 exhaust the completion-facing ledger by A, H, C, P, G, N on the same finite route, with no exported completion object, quotient completion equality, selected limit, host completeness proof, or off-packet generator certificate.

Finally apply terminal-output readback and terminal authorized output cover. Theorems 525.225 and 525.121 keep output consumption inside $I, E, M, B, D, O, A, H, C, P, G, N$ and leave no route for host evaluation or hidden generator state.

The excluded data would require an additional carrier coordinate or a separate completion-seal row. Neither is supplied by the AuthorizedGeneratorRecursor packet, so the completion boundary cannot escape the displayed BEDC route. □

Corollary 525.210 (AuthorizedGeneratorRecursor L10 source-admission strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Suppose an L10-facing* $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or completion-facing consumer displays a failure, refusal, separation, or countermodel row against a named converse or reuse route for source admission of* R. *Then the displayed row is a strict local obstruction: it blocks only that named converse or reuse route as a consumer-facing read through the finite rows*

$$I, E, M, B, D, O, A, H, C, P, G, N$$

with the displayed BHist, *componentwise* hsame, Cont, Pkg, *boundary, audit, terminal readback, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *evidence. It cannot be reinterpreted as a global impossibility result, a host theorem about recursors or completion, hidden external validation of the failed route, a selected real limit, quotient stream equality, detached generator state, ambient completion principle, or off-packet proof channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10SourceAdmissionStrictO`

Proof. Apply source-row admission, Definition 525.207. It fixes the source surface for every L10-facing accepted output packet: the $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ source rows are exposed only as displayed finite BHist rows inside the authorized output route, and only together with authorization, branch, descent, audit, componentwise hsame, Cont, Pkg, boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ rows of R.

Apply completion-boundary nonescape, Definition 525.209. It says that no $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or completion-facing consumer of R can recover a selected Real limit, quotient stream equality, host evaluation, detached generator state, ambient completion principle, or off-packet proof channel from the packet. Every such read remains bounded by the displayed BHist, hsame, Cont, Pkg, boundary, audit, and local $\mathsf{NameCert}$ route.

Now consider the displayed failure, refusal, separation, or countermodel row. If the row is used against the named source-admission converse or reuse route, the admission theorem makes its consumer-facing use factor through the listed finite source rows, and the nonescape theorem prevents that use from acquiring any unlisted completion, host, generator, or external proof coordinate. The row therefore obstructs exactly the requested route at the displayed packet coordinate where the route asks to read, replace, or bypass the source-admission evidence.

A stronger reading would require a coordinate not exported by R. A global impossibility result would require a theorem surface beyond the finite packet; a host theorem would require host evaluation or host completeness data; hidden external validation would require an off-packet certificate channel; and selected limits, quotient stream equalities, detached generator states, ambient completion principles, or proof-producing channels are exactly the data excluded by Definition 525.209. Since the accepted packet supplies none of those coordinates, the obstruction remains strict and local to the named L10 source-admission route. □

Theorem 525.211 (AuthorizedGeneratorRecursor terminal-output readiness)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N),$$

the terminal output row O is dependency-ready only as a read of the same displayed carrier rows. Any terminal-output consumer must first preserve signature admission I, emitted recursor naming E, motive checking M, branch coverage B, structural descent D, audit provenance A, componentwise transport H, displayed routes C, package provenance P, boundary row G, and the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N. No terminal-output read may detach O into a hidden generator, host induction rule, external axiom, or proof-producing channel.*

Proof. Project the carrier of Definition 525.1. The output row O is one coordinate of $I, E, M, B, D, O, A, H, C, P, G$, not a second carrier.

Use audit provenance factorization and audit boundary. Theorems 525.6 and 525.4 keep public exposure of O behind the audit row, branch package, descent row, transports, routes, package provenance, boundary, and local name.

Finally attach output-name stability and ground-compiler handoff. Theorems 525.10 and 525.7 preserve the name of O only on the same finite carrier, so no hidden generator or host proof channel can replace the displayed packet. □

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputReadiness`

Theorem 525.212 (AuthorizedGeneratorRecursor audit-gated output coverage)**.** *Every public terminal-output read of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet is covered by the audit-gated rows A, H, C, P, G, N together with the source rows I, E, M, B, D. In particular, if a consumer reads O, it also reads the audit boundary, output-name stability, and ground-compiler handoff for the same carrier surface.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorAuditGatedOutputCoverage`

Proof. Begin with Definition 525.4. It places public output exposure behind accepted signature, motive, branch coverage, structural descent, and audit provenance.

Next use Definition 525.6. The audit, transport, continuation, package, boundary, and local naming rows remain attached before O is exported.

Output-name stability then prevents the terminal row from being renamed away from this audit package. The ground-compiler handoff consumes the same carrier only as a finite handoff, so terminal-output coverage is exactly the displayed rows and not an external generator route. □

Theorem 525.213 (AuthorizedGeneratorRecursor local-name output exhaustion)**.** *For every accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N *exhausts terminal-output naming together with* O, A, H, C, P, G*. Any two terminal-output reads preserving the carrier rows* $I, E, M, B, D, O, A, H, C, P, G, N$ *name the same authorized output surface, and no consumer can add a detached local name, substitute branch table, unaudited proof object, or external compiler certificate.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorLocalNameOutputExhaustion`

Proof. Read the carrier and retain N with the output package. The local name row is attached to the same finite packet as O, A, H, C, P, G.

Apply output-name stability. Theorem 525.10 keeps terminal naming componentwise stable under the displayed hsame transports and continuation routes.

Use audit provenance factorization and ground-compiler handoff to close the consumer boundary. Theorems 525.6 and 525.7 leave no coordinate for a detached name, substitute branch table, unaudited proof object, or external compiler certificate outside the carrier. □

Theorem 525.214 (AuthorizedGeneratorRecursor Phase Real carrier readiness)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Any Phase Real handoff that reads* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *dependency rows for* R *must first admit the displayed carrier as the finite* BHist *route* $I, E, M, B, D, O, A, H, C, P, G, N$*, with* H *carrying componentwise* hsame*,* C *carrying displayed* Cont *replay,* P *carrying package provenance,* G *carrying the boundary ledger, and* N *carrying the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPhaseRealCarrierReadiness`

Proof. Project the carrier of Definition 525.1. The generator route exposes exactly the signature, emitted recursor, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows.

Use the L10 source triad Definition 525.192. It already confines $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads to the displayed packet and rejects host choice, detached induction, selected real data, quotient stream data, and external proof production.

The Phase Real handoff therefore begins with carrier admission for the same finite BHist route. No dependency row can be read before $I, E, M, B, D, O, A, H, C, P, G, N$ are retained. □

Theorem 525.215 (AuthorizedGeneratorRecursor Phase Real RegSeqRat handoff)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, every Phase Real* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is mediated by the same authorized generator route as the* $\mathsf{StreamName}^{\uparrow}$ *source and the* $\mathsf{Real}^{\uparrow}$ *endpoint. The handoff preserves* O, A, H, C, P, G, N *together with the signature, motive, branch, and descent rows, and it does not expose a quotient stream, selected limit, host equality, hidden generator, or off-packet proof channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPhaseRealRegSeqRatHandoff`

Proof. Begin with Definition 525.214. It fixes the carrier route and makes O, A, H, C, P, G, N part of every Phase Real dependency read.

Now apply the Real stream route Definition 525.203. That theorem identifies the public generator output as a finite read of $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ source rows through the same packet.

Finally use Definition 525.193. A downstream real or completion-facing consumer may specialize the finite source route, but it cannot reinterpret the recursor as quotient stream data, selected real data, host equality, hidden generator state, or an off-packet proof channel. □

Theorem 525.216 (AuthorizedGeneratorRecursor Phase Real ledger non-escape)**.** *Every Phase Real ledger exported by an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet is exhausted by the finite route* $I, E, M, B, D, O, A, H, C, P, G, N$*. The audit row* A*, boundary row* G*, componentwise* hsame *row* H*, displayed* Cont *row* C*, package row* P*, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N *are the only ledger surface visible to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumers.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPhaseRealLedgerNonescape`

Proof. Read the carrier and retain the audit and boundary rows. By Definition 525.214, every Phase Real dependency read already uses the displayed finite route.

Use Definition 525.215 for the regular-rational route. It keeps the $\mathsf{RegSeqRat}^{\uparrow}$ handoff attached to the stream-name source, real endpoint, output row, audit row, transport, continuation, package provenance, boundary, and local naming rows.

Close with Definition 525.194. The exported ledger has no coordinate for completion objects, real-completeness data, quotient completion equality, selected limits, host completeness proofs, or off-packet generator certificates. □

Theorem 525.217 (AuthorizedGeneratorRecursor Real output factorization lock)**.** *Let* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$ *be an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Every Real-facing output read from* R *factors through the displayed finite packet before it can be consumed as* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *data. In particular, no consumer may read the output row* O *without also retaining the audit row* A*, the componentwise transport row* H*, the displayed continuation row* C*, the provenance row* P*, the boundary ledger row* G*, and the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRealOutputFactorizationLo`

Proof. Project the carrier by Definition 525.214. The Phase Real surface first admits the finite route $I, E, M, B, D, O, A, H, C, P, G, N$, so the output row is not available as a detached Real object.

Next apply Definition 525.203. The public Real route already makes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readings pass through the same displayed packet.

Finally use Definition 525.193. Downstream consumers may specialize the route to their own readback surface, but the specialization still preserves O together with A, H, C, P, G, N and has no coordinate for quotient stream equality, selected limits, host Real equality, or external proof production. □

Theorem 525.218 (AuthorizedGeneratorRecursor L10 consumer readback totality)**.** *For an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, every L10 consumer readback of the Real-facing output is total only as a readback of the packet rows* O, A, H, C, P, G, N*. The consumer may request* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *data, but the request preserves the output row, audit row, transport row, continuation row, provenance row, boundary ledger, and local naming row as one finite* BHist *route.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10ConsumerReadbackTotali`

Proof. Start with Definition 525.214. It supplies the carrier route and makes H, C, P, G, N part of the Phase Real admission data.

Use Definition 525.215. The regular-rational handoff is mediated by the same authorized generator route as the stream-name source and the Real endpoint, so an L10 consumer cannot drop the output row or its audit, transport, continuation, provenance, boundary, and naming rows.

Apply Definition 525.216 for the ledger side. The exported ledger is exhausted by A, H, C, P, G, N on the finite packet.

Finally compose these obligations with Definition 525.203 and Definition 525.193. The result is a total consumer readback of the displayed packet, not a separate source of Real evidence or proof-producing data. □

Theorem 525.219 (AuthorizedGeneratorRecursor completion boundary saturation)**.** *Every completion-facing consumer of an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet receives only the finite route* $I, E, M, B, D, O, A, H, C, P, G, N$*. The route saturates the boundary: it provides the rows needed for* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readback, while leaving no channel for quotient stream equality, selected limits, host Real equality, ambient completeness, countable choice, or external proof production.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorCompletionBoundarySaturat`

Proof. First read the carrier and the regular-rational handoff through Definition 525.214 and Definition 525.215. These theorems place the completion-facing consumer on the same finite route as the stream source and Real endpoint.

Next apply Definition 525.216. The only visible ledger is A, H, C, P, G, N, so a completion consumer has no boundary coordinate beyond the packet.

Close with Definition 525.203 and Definition 525.193. Their route and consumer clauses allow downstream readback, but only as packet-preserving specialization; they do not introduce quotient stream equality, selected limits, host Real equality, ambient completeness, countable choice, or an external proof-production channel. □

Corollary 525.220 (AuthorizedGeneratorRecursor Phase Real readiness strict obstruction)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Suppose a Phase Real consumer of* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or a completion-facing read displays a failure, refusal, separation, or counter-model row against a named converse, reuse, or readback route in the handoff surface of* R*. Then that row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read, omit, replace, detach, or bypass one of the displayed packet coordinates*

$$I, E, M, B, D, O, A, H, C, P, G, N.$$

Every permitted use factors through the finite BHist *carrier and through the visible ledger rows* A, G, H, C, P, N*, with* H *carrying componentwise* hsame*,* C *carrying displayed* Cont *replay,* P *carrying package provenance,* G *carrying the boundary ledger, and* N *carrying the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}}$ *row. The same displayed row cannot be reinterpreted as a global impossibility theorem, a host theorem about recursors or real completion, hidden external validation of the failed route, a selected real witness, a quotient stream equality, a completion object, a host completeness proof, or an off-packet proof-producing channel.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorPhaseRealReadinessStrict(

Proof. Start with carrier readiness, Definition 525.214. Any Phase Real handoff that reads $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ data must first admit the displayed finite route $I, E, M, B, D, O, A, H, C, P, G, N$. Thus a consumer-facing negative row can bear on the packet only after the carrier has been read, and its possible coordinates are already limited to the signature, emitted recursor, motive, branch, descent, output, audit, transport, continuation, provenance, boundary, and local naming rows.

Apply ledger non-escape, Definition 525.216. The exported Phase Real ledger is exhausted by the audit row A, boundary row G, componentwise hsame row H, displayed Cont row C, package row P, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row N. The same theorem also says that this is the only ledger surface visible to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or completion-facing consumers.

Now consider the displayed failure, refusal, separation, or countermodel row. If it is used against a named converse, reuse, or readback route, the first cited theorem forces the use to pass through the full finite carrier, and the second cited theorem forces its ledger component to remain on A, G, H, C, P, N. Hence the row can obstruct only the advertised route at the coordinate where that route asks to read, omit, replace, detach, or bypass one of the displayed rows.

Any stronger reading asks for a coordinate that the two cited theorems do not export. A global impossibility theorem would require a theorem surface beyond the finite packet; a host theorem about recursors or real completion would require host choice, detached induction, host equality, or host completeness data; hidden external validation would require an external certificate route; a selected real witness, quotient stream equality, completion object, or proof-producing channel would require one of the off-packet objects excluded by the ledger non-escape clause. Since no such coordinate is visible from R, the obstruction remains strict and local to the named Phase Real route. □

Theorem 525.221 (AuthorizedGeneratorRecursor L10 output authorization closure)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet. Any L10 public output read of O that is visible to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumers must retain the full finite route $I, E, M, B, D, O, A, H, C, P, G, N$ before those consumer rows are public. The output row is therefore authorized only together with the displayed signature, emitted recursor, motive, branch, descent, audit, transport, continuation, package provenance, boundary, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows; it is not a host-recursive output, a quotient stream datum, a selected real witness, or an off-packet proof channel.*

Lean4 ✓ BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor l10 output authorization closure

Proof. Start with Definition 525.214. It admits the Phase Real carrier only as the finite BHist route $I, E, M, B, D, O, A, H, C, P, G, N$, with H, C, P, G, N carrying displayed equality, continuation, package, boundary, and local naming data.

Use Definition 525.215 and Definition 525.216. They keep the regular-rational handoff and the exported ledger attached to the same packet, excluding quotient stream data, selected limits, host equality, hidden generator state, and off-packet proofs.

Finally apply Definition 525.203. The sibling consumer chapters chapter 126, chapter 301, and chapter 38 may therefore see the recursor output only after the route has retained every row of R. □

Theorem 525.222 (AuthorizedGeneratorRecursor StreamName finite-request publicity)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *finite-request section that reads an* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *output must read it through the displayed output row* O*, audit row* A*, transport row* H*, continuation row* C*, package row* P*, boundary row* G*, and local NameCert row* N*, with* I, E, M, B, D *retained as the upstream generator route. The section publicizes finite requests only as packet-local stream-name access; it does not expose an unbounded stream, detached recursion rule, selected real value, quotient stream equality, host equality, or private audit channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorStreamNameFiniteRequestPu`

Proof. Begin with Definition 525.214. It makes the $\mathsf{StreamName}^{\uparrow}$ dependency a read of the displayed carrier rather than a separate source of stream data.

Next use Definition 525.203. The Real stream route already places $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers behind the same finite output route, so a finite-request section cannot bypass O, A, H, C, P, G, N.

The public sibling surface is then exactly chapter 126. Since that chapter consumes finite stream-name rows and not host streams, the recursor output reaches it only through the displayed output, audit, transport, continuation, package, boundary, and NameCert rows. □

Theorem 525.223 (AuthorizedGeneratorRecursor RegSeqRat and Real seal output exhaustion)**.** *For any accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumer and every* $\mathsf{Real}^{\uparrow}$ *seal consumer of the recursor output exhausts the output route through the same finite packet. The only available public data are the displayed output, audit, transport, continuation, package, boundary, and local NameCert rows together with the upstream generator rows* I, E, M, B, D*. No consumer obtains a selected limit, quotient stream equality, host equality, detached completion object, hidden generator state, or proof channel outside* R*.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorRegSeqRatRealSealOutputEx`

Proof. Use Definition 525.215. It mediates the $\mathsf{RegSeqRat}^{\uparrow}$ handoff through the same authorized generator route as the stream-name source and the Real endpoint.

Then apply Definition 525.216. The ledger exported to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion-facing consumers is exhausted by A, H, C, P, G, N on the finite packet, with no coordinate for quotient completion equality, selected limits, host completeness proofs, or off-packet generator certificates.

Finally appeal to Definition 525.203 and the sibling chapters chapter 301 and chapter 38. Their readback and seal surfaces consume packet rows; they do not turn the AuthorizedGeneratorRecursor output into selected Real evidence, quotient stream equality, host equality, or an external proof producer. □

Lemma 525.224 (AuthorizedGeneratorRecursor L10 output downstream coverage)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet, and let* c *be an accepted local L10 consumer of the public output row* O*. If* c *is visible to a* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer surface, then* c *is exhausted by the finite alternatives and ledger rows already carried by* R*:*

1. *a* $\mathsf{StreamName}^{\uparrow}$ *finite-request read of the displayed output row* O*;*
2. *a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback handoff through the same authorized output route;*

3. a $\mathsf{Real}^{\uparrow}$ *seal consumer of the same output route;*

4. the ledger surface A, H, C, P, G, N*, with* I, E, M, B, D *retained as the upstream generator route.*

Consequently every request for a host-recursive output, quotient stream datum, selected Real witness, detached completion object, host equality, hidden generator state, or off-packet proof channel remains outside the BEDC packet.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorL10OutputDownstreamCovera`

Proof. Start with the authorization closure in Definition 525.221. It states that any L10 public output read visible to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumers must retain the full finite route $I, E, M, B, D, O, A, H, C, P, G, N$ before the consumer rows are public. Thus an accepted local consumer of O cannot begin from a bare output token: it already carries the output row together with the audit, transport, continuation, package provenance, boundary, and local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ rows.

For the stream-name alternative, apply Definition 525.222. A finite-request section reads the recursor output only through O, A, H, C, P, G, N, with I, E, M, B, D retained as the upstream generator route. Hence the $\mathsf{StreamName}^{\uparrow}$ case lands in the first listed alternative and has no unbounded stream or detached recursion channel.

For the regular-rational and Real-facing alternatives, apply Definition 525.223. It says that every $\mathsf{RegSeqRat}^{\uparrow}$ readback consumer and every $\mathsf{Real}^{\uparrow}$ seal consumer exhausts the output route through the same finite packet. The only available public data are O, A, H, C, P, G, N together with I, E, M, B, D, so these two cases land in the second and third alternatives, with the common ledger surface recorded in the fourth.

The exclusion clause is the intersection of the two cited closures. The authorization closure rules out a host-recursive output, quotient stream datum, selected real witness, and off-packet proof channel at the point where O becomes public. The readback-and-seal exhaustion theorem also rules out selected limits, quotient stream equality, host equality, detached completion objects, hidden generator state, and proof channels outside R. Since every accepted local consumer visible to the three named consumer surfaces has already been placed in one of the displayed finite alternatives, any excluded request would require a coordinate not present in $I, E, M, B, D, O, A, H, C, P, G, N$. Such a coordinate is not part of the accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ packet. □

Theorem 525.225 (AuthorizedGeneratorRecursor terminal-output readback)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *finite packet. Every terminal readback of the output row* O *by a* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer must first expose the same* BHist *packet, preserve the componentwise* hsame *row* H*, replay through the displayed* Cont *row* C*, retain the* Pkg *provenance row* P*, and name the read by the local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row* N*. The readback is therefore a packet-local output route, not a selected real value, quotient stream datum, host equality, detached generator state, or off-packet proof channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputReadbackPac`

Proof. Begin with terminal-output readiness, Definition 525.211. It places O inside the displayed carrier R and keeps the output row behind the audit, transport, continuation, package, boundary, and local naming rows.

Next use the Phase Real handoff rows, Definition 525.214 and Definition 525.215. They force $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads to consume the same finite output route.

Finally apply the L10 output authorization closure, Definition 525.221. That closure leaves no exported coordinate for a selected limit, quotient stream equality, host equality, hidden generator state, or external proof producer. □

Theorem 525.226 (AuthorizedGeneratorRecursor finite-tail authorization)**.** *For an accepted* AuthorizedGeneratorR *packet* $R = (I, E, M, B, D, O, A, H, C, P, G, N)$*, every finite tail of a public output route read by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *is authorized by the same rows* O, A, H, C, P, G, N *with* I, E, M, B, D *retained as the upstream generator surface. A tail consumer cannot shorten the route to a bare output token, external recursion rule, quotient completion object, or unledgered Real endpoint.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorFiniteTailAuthorization`

Proof. Start from the StreamName finite-request publicity row, Definition 525.222. It treats finite request sections as packet-local reads of the output row rather than as an unbounded stream.

Use the RegSeqRat and Real seal output exhaustion row, Definition 525.223. The regular-rational and Real-facing consumers see only the same finite packet rows and do not receive a detached completion witness.

The local-name output exhaustion theorem, Definition 525.213, then keeps every finite tail named by N on the same O, A, H, C, P, G surface. Thus the finite tail is authorized precisely by the displayed output packet. □

Theorem 525.227 (AuthorizedGeneratorRecursor consumer ledger exhaustion)**.** *Every consumer ledger for terminal output from an accepted* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *packet is exhausted by the finite rows* $I, E, M, B, D, O, A, H, C, P, G, N$ *together with the existing* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff rows. No downstream L10 consumer obtains a ledger extension for hidden generator state, selected limits, quotient stream equality, host completeness, host induction, or an off-packet proof channel.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursor consumer ledger exhaustion`

Proof. First project the Phase Real ledger non-escape row, Definition 525.216. It identifies the exported ledger with A, H, C, P, G, N on the same finite carrier.

Next combine terminal-output readback with finite-tail authorization above. The first theorem fixes the readback route for the output row, and the second keeps every finite tail on that same route for the stream-name, regular-rational, and Real-facing consumers.

Finally use Definition 525.203 and the L10 output authorization closure. These surfaces route all downstream L10 reads through the displayed packet, so the consumer ledger has no additional coordinate beyond R and the named handoff rows. □

Lemma 525.228 (AuthorizedGeneratorRecursor terminal-output downstream coverage)**.** *Let*

$$R = (I, E, M, B, D, O, A, H, C, P, G, N)$$

be an accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *finite packet, and let* c *be an accepted local consumer of the terminal output row* O*. If the consumer-facing use of* c *is visible through a* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *surface, then the use is exhausted by the displayed terminal-output coverage already carried by the packet:*

1. *the terminal readback route through the same* BHist *packet;*
2. *the componentwise transport row H and displayed* Cont *replay row C;*
3. *the package provenance row P, boundary ledger row G, and local* $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ *row N;*
4. *the finite consumer ledger over $I, E, M, B, D, O, A, H, C, P, G, N$ together with the named* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff rows.*

Consequently every attempted downstream read that asks for hidden generator state, a selected limit, quotient stream equality, host completeness, host induction, host equality, a detached completion object, or an off-packet proof channel remains outside the BEDC packet.

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTerminalOutputDownstream(`

Proof. Start with terminal-output readback, Definition 525.225. It applies directly to the output row O of the accepted packet R and forces every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ readback to expose the same BHist packet before the read is available. The same theorem also preserves the componentwise hsame row H, replays the displayed Cont row C, retains the Pkg provenance row P, and names the read by the local $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ row N. Thus the consumer cannot begin from a bare output token or from a detached generator state; its first three coverage coordinates are already the displayed packet-local readback, transport, continuation, provenance, boundary, and naming rows.

Now apply consumer ledger exhaustion, Definition 525.227. That theorem identifies the downstream terminal-output ledger with the finite rows $I, E, M, B, D, O, A, H, C, P, G, N$ together with the existing $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff rows. Hence an accepted local consumer visible through one of the three named surfaces has no independent ledger beyond the rows listed in the statement: the stream-name read, regular-rational readback, and Real-facing read are all ledger entries over the same finite route.

It remains only to check the excluded alternatives. The readback theorem says that the terminal read is packet-local and not a selected real value, quotient stream datum, host equality, detached generator state, or off-packet proof channel. The ledger-exhaustion theorem adds that no downstream L10 consumer obtains a ledger extension for hidden generator state, selected limits, quotient stream equality, host completeness, host induction, or an off-packet proof channel. A request for any excluded object would therefore require either a readback coordinate not supplied by H, C, P, G, N or a consumer ledger coordinate not present in $I, E, M, B, D, O, A, H, C, P, G, N$ and the three named handoff rows. Since both cited theorems exhaust those coordinates, the request remains outside the accepted $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ packet. □

The $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ public export is the displayed route through generator source, authorization, motive, branch package, descent, output, audit, transport, replay, provenance, boundary, and local naming rows. Public consumers may inspect the emitted output only through that finite route, and the mature treatment is the paper-side route exhaustion of the mature child together with the L10 downstream coverage rows; full bridge verification and host generator semantics remain outside the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}}$ **·**

Constructive story: An $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ packet is generated from an accepted constructor signature, an emitted eliminator, a checked motive, complete branch data, structural descent, an output name, audit provenance, componentwise hsame transports, displayed Cont routes, Pkg provenance, a GAP boundary row, and a local NameCert row. Its carrier surface composes the signature, branch, descent, audit, boundary, compiler-handoff, and local-name rows already displayed by the sibling authorization packets.

Theory closure: TheoryGate(NameCert$_{\text{AuthorizedGeneratorRecursor}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 525.1, Definition 525.2, Definition 525.3, Definition 525.4, Definition 525.5, Definition 525.6, Definition 525.12, Definition 525.13, Definition 525.14, and Definition 525.7 with

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.taste gate`

expose authorized recursor output as a public finite NameCert packet over accepted signatures, branch coverage, structural descent, audit provenance, boundary, compiler handoff, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{AuthorizedGeneratorRecursor}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorTasteGate\_single\_carrier\_al`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, a public standard bridge, host generator semantics, unrestricted substitution, host normalization, hidden recursion, ambient equality, or any proof-producing route outside the displayed recursor packet.

Upgrade path: Verification closure requires Lean-checked carrier admission, authorization, branch coverage, descent, output audit, downstream handoff, transport, replay, provenance, boundary, and local naming for the public recursor route.

Theorem 525.229 (Authorized generator recursor mature route exhaustion)**.** *Every terminal generator-output consumer of an accepted* AuthorizedGeneratorRecursor$^{\uparrow}$ *packet factors through the root authorization rows, signature totality, acceptance totality, terminal output readiness, the L10* StreamName$^{\uparrow}$/RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *source triad, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert$_{\text{AuthorizedGeneratorRecursor}^{\uparrow}}$ *rows before export. The mature route obtains no hidden generator, host recursion principle, selected output stream, quotient equality, subject-reduction theorem, or ambient* Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.AuthorizedGeneratorRecursorUp.AuthorizedGeneratorRecursorMatureRouteExhaustion`

Chapter 526

A Concrete Naming Certificate for AuditGateBoundary$^{\uparrow}$

The AuditGateBoundary$^{\uparrow}$ surface packages the manuscript and Lean audit gates as a finite naming certificate. It is not a proof that the entire repository is complete. It records the exact boundary packets by which source-token scans, axiom-purity reports, paper-Lean target audits, origin and import conservativity ledgers, transport rows, continuation routes, package provenance, GAP boundary rows, and the local naming row remain visible to downstream governance consumers. The source-token and forbidden-dependency rows share their refusal ledger with chapter 861, while the marker-resolution row is read against the audit-map interface of chapter 806. The displayed audit consumer routes use the finite continuation-trace reading of chapter 469 as the sibling surface that makes replay and route inspection explicit.

Definition 526.1 (AuditGateBoundary carrier)**.** *An* AuditGateBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, D, L, O, H, C, P, G, N)$$

with the following displayed rows:

1. *S is the source-token scan row, recording the finite file set, token classifier, and refusal result for substrate-internal axiom declarations;*
2. *D is the axiom-purity dependency report row, recording the theorem names inspected, the reported proof ancestry, and the forbidden dependency classifier for* Classical.choice*,* Quot.sound*, and* propext*;*
3. *L is the paper-Lean target audit row, recording each manuscript marker, its named Lean target, and the finite resolution result used by the drift gate;*
4. *O is the origin and import conservativity report row, recording the chapter-origin ledger, the protected module prefixes, and the import-edge classifier that separates baseline and non-baseline surfaces;*
5. *H records componentwise* hsame *transports for S, D, L, O, G, the continuation row, the package row, and the local naming row;*
6. *C records the displayed* Cont *routes by which consumers read the token scan, dependency report, target audit, origin/import report, boundary row, provenance, and local certificate;*

7. *P is the* Pkg *provenance over* BHist, Hist, GAP, hsame, Cont, Pkg, NameCert, *Lean target names, source-file scans, dependency reports, and origin ledgers;*
8. *G is the* GAP *boundary row recording that the gates refuse only the displayed boundary crossings and do not provide a hidden theorem of global correctness;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditGateBoundary}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no unrestricted semantic conservativity theorem, no host-kernel completeness proof, no hidden permission to use source axioms, no selected Lean declaration beyond the named audit rows, and no global guarantee that paper prose carries a stronger interpretation than the resolving markers display.

Lean4 (definition) `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryUp`

Theorem 526.2 (AuditGateBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AuditGateBoundary}^{\uparrow}}$ *surface for the carrier of Definition 526.1 consists of carrier admission for*

$$A = (S, D, L, O, H, C, P, G, N),$$

source-token refusal, axiom-purity dependency refusal, paper-Lean target exactness, origin/import conservativity separation, componentwise hsame *stability, displayed* Cont *locality,* Pkg *provenance, the* GAP *boundary row, and non-escape for global semantic conservativity, host-kernel completeness, source axioms, unresolved marker names, and untracked origin flow.*

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 526.1. The rows $S, D, L, O, H, C, P, G, N$ are the only data admitted to the packet.

The first four rows expose the gate faces. The source-token scan is read from S, the dependency report from D, the paper-Lean target audit from L, and the origin/import conservativity report from O.

Transport and consumer access are then confined by H and C. These rows replay the displayed audits under componentwise hsame movement and do not add any unrecorded proof principle or import edge.

Finally P, G, N record provenance, the boundary socket, and the local certificate. Since the excluded objects have no coordinate in the carrier, no NameCert projection can export them. □

Theorem 526.3 (AuditGateBoundary axiom-purity soundness)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the source-token row S and dependency report row D are sufficient to decide the audit-facing axiom boundary claimed by the packet. A passed source-token scan means no displayed BEDC source file contributes a substrate-internal axiom declaration, and a passed dependency report means no inspected public theorem row is reported with a forbidden dependency among* Classical.choice, Quot.sound, *and* propext.

This soundness is local to the reported rows. It does not assert host-kernel completeness, semantic completeness of the theorem inventory, or any theorem outside the finite report carried by D.

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier axiom purity soundness`

Proof. Read the source side first. The row S contains the finite file set, token classifier, and refusal result, so the source-axiom boundary claimed by the packet is exactly the boundary displayed by S.

Read the theorem-dependency side next. The row D contains the inspected theorem names, their reported proof ancestry, and the forbidden dependency classifier. Therefore the axiom-purity claim is the finite absence result recorded by D.

The GAP row G fixes the scope. The packet certifies only these displayed reports and refuses to turn them into a hidden theorem about all possible host behaviour or every unlisted declaration. □

Theorem 526.4 (AuditGateBoundary paper-Lean drift exactness)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the paper-Lean audit row L is the exact public surface for resolving manuscript markers against Lean declarations. A consumer may read a marker as a naming certificate only through the tuple consisting of the marker text, the target name, the finite resolution result, the componentwise transport row, the continuation route, the package provenance, and the local* $\mathsf{NameCert}_{\mathsf{AuditGateBoundary}^{\uparrow}}$ *row.*

No consumer of L may replace a missing target with a similar theorem, a nearby namespace, a paper-side interpretation, or an unresolved string.

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier paper lean drift exactness`

Proof. Project L from the carrier. By definition it records each manuscript marker, the target name being claimed, and the finite resolution result.

Use H and C to transport and read the row. The marker remains the same displayed marker after transport, and the continuation route exposes only the named resolution result.

The non-escape clause follows from the absence of any coordinate for fuzzy matching, namespace guessing, or prose interpretation. A marker therefore functions as a certificate only when L records that the named target resolves. □

Theorem 526.5 (AuditGateBoundary origin conservativity)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the origin/import row O is the exact finite ledger by which non-baseline surfaces are kept from silently entering baseline theorem space. A consumer may use O to read chapter origins, protected module prefixes, inspected import edges, and the pass-or-refuse result, but it may not extract a semantic conservativity theorem for arbitrary extensions or a global proof that all recognized content is baseline content.*

Proof. Project O from the carrier. It contains origin rows, protected prefixes, inspected import edges, and the classifier result for those edges.

Transport and replay through H and C. The consumer sees the same finite ledger and the same route to the origin/import classifier, with provenance recorded by P and local naming recorded by N.

The boundary row G prevents overreading. Since O has no coordinate for arbitrary model-theoretic conservativity or for reclassifying non-baseline material as baseline, the packet certifies only the displayed import separation. □

Theorem 526.6 (AuditGateBoundary source-token refusal)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the row S is the only public source-token refusal face. Every consumer read of a source-token refusal factors through S together with the componentwise transport row H, the displayed continuation routes C, the package provenance P, and the local naming row N.

The read does not expose a hidden source proof, a host-kernel axiom inventory, or a refusal predicate outside the finite file set and token classifier recorded by S.

Proof. Project the accepted packet. The source-token scan row is the coordinate S, while D, L, O, G record different audit faces and cannot be substituted for the source-token refusal row.

Route the consumer access through H and C. The transport row preserves the displayed scan componentwise, and the continuation row exposes exactly the route from the packet to the token classifier and refusal result carried by S.

The package row P and local name row N bind this route to the $\mathsf{AuditGateBoundary}^{\uparrow}$ certificate. The GAP row G then fixes the non-escape boundary: the packet refuses only the displayed source-token crossing and supplies no host-wide source theorem. □

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier source token refusal`

Theorem 526.7 (AuditGateBoundary forbidden-dependency ledger)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the dependency row D fixes the finite forbidden-dependency classifier for Classical.choice, Quot.sound, *and* propext. *A consumer may read only the theorem names inspected, the reported proof ancestry, and the pass-or-refuse result carried by D, transported through H and routed through C.*

This ledger does not claim host-kernel completeness, proof-irrelevance completeness, or a dependency result for declarations absent from the finite report.

Proof. Project D from the packet. Its coordinates are the inspected theorem names, reported proof ancestry, and the classifier for Classical.choice, Quot.sound, and propext.

Transport is componentwise through H. The dependency report remains the same finite classifier row after the accepted hsame movement, and the continuation row C is the only consumer route back to that report.

Finally use P, N, G to close the boundary. Provenance and local naming identify the certificate surface, while the GAP row prevents the finite dependency report from becoming a hidden host-kernel completeness theorem. □

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier dependency forbidden set`

Theorem 526.8 (AuditGateBoundary inductive audit row)**.** *Every accepted audit-gate consumer of an* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet is obtained by finite row induction over*

$$S, D, L, O, H, C, P, G, N.$$

The source-token refusal, forbidden-dependency refusal, marker-resolution exactness, and origin conservativity reads are consumed one row at a time. The transport, replay, provenance, GAP boundary, and naming rows may preserve and route those four audit faces, but they cannot add host-kernel completeness, theorem-inventory completeness, fuzzy marker resolution, or semantic conservativity for arbitrary imports.

Proof. Project the carrier of Definition 526.1. Its displayed rows are exactly $S, D, L, O, H, C, P, G, N$, so the audit consumer has no hidden coordinate beyond the finite packet.

Read the four audit faces in order. Use Definition 526.6 for the source-token row S, Definition 526.7 for the forbidden-dependency row D, Definition 526.4 for the marker row L, and Definition 526.5 for the origin and import row O.

It remains to inspect the support rows. Componentwise transport through H, replay through C, provenance through P, the GAP boundary G, and local naming through N preserve the already displayed rows and do not create a new audit face. Therefore the consumer is obtained by finite

row induction over the displayed packet, with host-kernel completeness and semantic conservativity outside the export. □

Theorem 526.9 (AuditGateBoundary marker-resolution non-escape)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the paper-Lean row L resolves manuscript markers only by exact target names. After componentwise transport through H and consumer routing through C, the public certificate may expose the marker text, the named Lean target, and the finite resolution result recorded by L.

It cannot replace a non-resolving marker by a nearby namespace, an unresolved string, a prose interpretation, or a theorem whose name is not the exact target recorded in L.

Proof. Project the marker row L from the accepted packet. The row records the manuscript marker, the claimed Lean target name, and the finite resolution result.

Replay the read through H and C. Componentwise transport preserves the displayed marker row, and the continuation route exposes only the exact target-name resolution already present in L.

The remaining coordinates P, G, N supply provenance, the boundary row, and local naming. Since the carrier has no coordinate for fuzzy namespace search, unresolved-string substitution, or prose interpretation, marker resolution cannot escape the exact-name ledger. □

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier marker resolution nonescape`

Theorem 526.10 (AuditGateBoundary origin-gate replay)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the displayed audit replay of origin conservativity is exactly the origin/import row O transported through H, routed through C, named by N, and packaged by P. A governance consumer may replay chapter-origin rows, protected module prefixes, inspected import edges, and the pass-or-refuse classifier, but it receives no semantic conservativity theorem beyond the finite BHist *row carried by O.*

Proof. Project the carrier row O. By Definition 526.1, this row is the packet's origin and import conservativity report.

Apply Definition 526.5. The consumer may inspect exactly the displayed origin rows, protected prefixes, import edges, and classifier result.

Replay the row through the common support surface. The transport row H preserves O componentwise, the continuation row C exposes it, and P, N bind provenance and local naming.

The replay ledger of Definition 526.13 keeps this read inside the audit packet. Therefore the replay exports the finite origin gate row and not a host-wide conservativity theorem. □

Theorem 526.11 (AuditGateBoundary strict axiom-purity replay)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

strict axiom-purity replay is the joint read of the source-token row S and forbidden-dependency row D through H, C, P, G, N. The replay exposes the finite source-token refusal, the inspected theorem names, their reported proof ancestry, and the forbidden-dependency classifier for Classical.choice, Quot.sound, *and* propext, *with no hidden host-kernel completeness claim.*

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier strict axiom purity replay`

Proof. Project the rows S and D from the accepted packet. The source-token row records file and token refusal data, while the dependency row records theorem names, ancestry, and forbidden dependency classification.

Use Definition 526.3 and Definition 526.6. These theorems bind the audit-facing axiom boundary to the finite source-token and dependency rows.

Use Definition 526.7 for the strict forbidden set. The only named forbidden dependencies are Classical.choice, Quot.sound, and propext inside the displayed report.

Finally replay through H, C, P, G, N. Transport, continuation, provenance, GAP boundary, and naming keep the result inside the same BHist packet and block any host-wide completeness reading. □

Theorem 526.12 (AuditGateBoundary marker-resolution closure)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the marker-resolution row L is closed under audit replay through the displayed ledger. A governance consumer replaying L may recover only the manuscript marker text, exact Lean target name, finite resolution result, componentwise hsame *transport, continuation route,* Pkg *provenance, GAP boundary, and local* NameCert *row.*

Proof. Project the paper-Lean row L from the accepted carrier. This row contains the marker text, named Lean target, and finite resolution result.

Apply Definition 526.9. Exact target names are preserved, and the row cannot be replaced by namespace search, unresolved strings, prose interpretation, or a different theorem name.

Replay L through the ledger of Definition 526.13. The same support rows H, C, P, G, N transport, expose, package, delimit, and name the marker-resolution read.

Thus marker resolution is closed only as a finite audit-row replay. It does not become a broader manuscript-validity theorem or a substitute for resolving the exact target recorded in L. □

Theorem 526.13 (AuditGateBoundary replay ledger)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

any governance consumer replaying the packet must read the source-token row S, forbidden-dependency row D, marker-resolution row L, and origin/import row O through the same transport, continuation, provenance, GAP-boundary, and naming support rows H, C, P, G, N. The replay cannot replace exact marker resolution, forbidden dependency reports, source-token refusal, or origin/import separation by a host-wide correctness claim.

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier replay ledger`

Proof. Project the carrier of Definition 526.1. Its four audit-facing rows are exactly S, D, L, O, and the shared support rows are exactly H, C, P, G, N.

Apply the local NameCert obligations to bind those rows to one accepted packet. Source-token refusal uses S through the same transport and continuation coordinates, while the dependency forbidden set uses D to report only the displayed ancestry and forbidden names.

Next replay marker resolution and origin separation. The marker-resolution non-escape theorem keeps L on exact target names, and origin conservativity keeps O on the displayed origin/import ledger.

Finally close the replay through P, G, N. Provenance, the GAP boundary, and local naming prevent the four audit faces from becoming a global repository-correctness theorem, so a governance consumer receives only the finite replay ledger carried by A. □

Theorem 526.14 (AuditGateBoundary origin-import replay exactness)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

origin and import replay is exact only as the row O transported through H, routed through C, packaged by P, delimited by G, and named by N. A governance consumer may replay chapter-origin rows, protected module prefixes, inspected import edges, and their pass-or-refuse classifier, but it may not replace them by a host-wide semantic conservativity claim.

Proof. Project the carrier row O. It is the only coordinate that records chapter origins, protected module prefixes, inspected import edges, and the origin/import classifier.

Replay proceeds through the shared support rows. The transport row H preserves the displayed origin/import ledger, the continuation row C exposes it, and the package and local naming rows P, N bind the replay to the same certificate.

The boundary row G fixes the scope of the replay. Since the carrier contains no coordinate for host-wide semantic conservativity, the exact replay exports only the finite origin/import row. □

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier origin import replay exactness`

Theorem 526.15 (AuditGateBoundary drift row transport)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet, the paper-Lean drift row L is transported exactly as a finite marker-resolution ledger. Every transported read of L exposes only the manuscript marker text, the exact Lean target name, the finite resolution result, the componentwise* hsame *row H, the displayed* Cont *route C, the* Pkg *provenance row P, the* GAP *boundary row G, and the local* NameCert *row N.*

Proof. Project L from the accepted packet. The row records marker text, target name, and finite resolution result, and it contains no fuzzy target selector.

Transport the row componentwise through H. The transported ledger is still the same exact-name marker-resolution row, so the consumer cannot replace it by a nearby namespace or prose interpretation.

Route and package the transported row through C, P, G, N. The replay ledger and marker-resolution non-escape theorems then keep the read inside the finite audit packet, with no host-wide correctness claim attached to the transported drift row. □

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier drift row transport`

Theorem 526.16 (AuditGateBoundary axiom-purity report non-escape)**.** *For an accepted* AuditGateBoundary$^{\uparrow}$ *packet, the axiom-purity report is the joint finite read of the source-token row S and dependency row D through H, C, P, G, N. It reports only the displayed source-token refusal, inspected theorem names, reported proof ancestry, and forbidden-dependency classifier for* Classical.choice*,* Quot.sound*, and* propext*. It cannot be replaced by host-kernel completeness, theorem inventory completeness, global proof purity, or a semantic repository-correctness claim.*

Proof. Project the source and dependency rows. The source-token refusal is carried by S, while the dependency report and forbidden-dependency classifier are carried by D.

Use source-token refusal, forbidden-dependency ledger, and strict axiom-purity replay. Together they identify the axiom-facing report as exactly the finite pair of rows S, D plus the shared support rows.

Finally inspect the carrier boundary. The rows H, C, P, G, N preserve, expose, package, delimit, and name the finite report, and no coordinate records host-kernel completeness, theorem inventory completeness, global proof purity, or semantic repository correctness. The report therefore cannot escape the exact audit rows. □

Theorem 526.17 (AuditGateBoundary obligation closure surface)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the local obligation surface is the joint assembly of the source-token scan S, strict axiom-purity report D, paper-Lean drift exactness row L, origin-conservativity row O, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, GAP boundary G, and local* $\mathsf{NameCert}_{\mathsf{AuditGateBoundary}^{\uparrow}}$ *row N. Every governance-facing audit claim for the packet factors through these rows.*

Proof. Project the carrier of Definition 526.1. The rows $S, D, L, O, H, C, P, G, N$ are the complete public packet, so carrier admission contributes no additional audit face.

Use the existing source and dependency rows for the axiom side. The source-token refusal theorem binds S, while strict axiom-purity replay and axiom-purity report non-escape bind D to the displayed forbidden-dependency classifier.

Use the existing drift and origin rows for the manuscript side. Marker-resolution closure and drift row transport bind L, while origin-import replay exactness binds O to the finite origin/import ledger.

Finally replay the shared support rows. The replay ledger keeps H, C, P, G, N as transport, route, provenance, boundary, and naming support for the same packet, so no hidden global correctness claim is exported. □

Theorem 526.18 (AuditGateBoundary ledger exactness)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the audit ledger is exact: S is the only source-token refusal row, D is the only strict axiom-purity report row, L is the only exact marker-resolution row, and O is the only origin/import conservativity row. The rows H, C, P, G, N may transport, route, package, delimit, and name these four ledgers, but they cannot replace them by fuzzy Lean target matching, source-axiom admission, forbidden host dependency acceptance, or a host-wide semantic correctness theorem.*

```
Lean4 ✓  BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier ledger exactness
```

Proof. Read the four audit ledgers from the accepted carrier. Source-token refusal uses S, forbidden-dependency classification uses D, marker resolution uses L, and origin/import separation uses O.

Apply the corresponding exactness theorems. Source-token refusal, forbidden-dependency ledger, marker-resolution non-escape, and origin-import replay exactness identify the public content of the four rows.

The support rows do not create substitute ledgers. They only preserve, expose, package, delimit, and name the displayed rows; the GAP boundary blocks fuzzy target matching, source-axiom admission, forbidden host dependency acceptance, and hidden semantic correctness. □

Theorem 526.19 (AuditGateBoundary obligation non-escape)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the obligation closure surface exports no hidden global correctness claim. Its public reads are exactly source-token refusal, strict axiom-purity reporting, paper-Lean exact target resolution, origin/import separation,* hsame *transport,* Cont *replay,* Pkg *provenance, GAP boundary, and local naming; no read yields fuzzy Lean target matching, source-axiom admission, forbidden host dependency acceptance, theorem-inventory completeness, host-kernel completeness, or semantic conservativity for arbitrary imports.*

Proof. Start with the obligation surface of Definition 526.17. It assembles only the carrier rows $S, D, L, O, H, C, P, G, N$.

Use ledger exactness to inspect every public read. Each governance claim is forced back to the displayed source-token, dependency, marker-resolution, or origin/import row, with the shared support rows providing only transport and replay.

The excluded claims have no coordinate in the carrier. Since the GAP boundary is part of the same packet, the surface cannot export fuzzy target matching, source-axiom admission, forbidden host dependency acceptance, inventory completeness, host-kernel completeness, or arbitrary-import conservativity. □

Theorem 526.20 (AuditGateBoundary kernel-query scope)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet*

$$A = (S, D, L, O, H, C, P, G, N),$$

the source-token row S and the dependency report row D are the only rows by which source-token reads and axiom-query reads enter the boundary. Every such read must then pass through the shared support rows H, C, P, G, N before it can be consumed as an $\mathsf{AuditGateBoundary}^{\uparrow}$ *verdict surface.*

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryCarrier kernel query scope`

Proof. Project the carrier of Definition 526.1. The row S is the only source-token scan row, and the row D is the only axiom-purity dependency report row.

Apply Definition 526.3. It binds the audit-facing axiom boundary to the finite pair consisting of the source-token scan and the dependency report.

Apply Definition 526.6 for the source side and Definition 526.7 for the query side. These theorems route the two reads through H, C, P, G, N and leave L and O as distinct audit faces.

Finally use Definition 526.19. Since no other carrier row can introduce source-token or axiom-query content, the kernel-query scope is exactly S, D plus the shared support rows. □

Theorem 526.21 (AuditGateBoundary forbidden-dependency scope)**.** *For an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet, the forbidden-dependency classifier is exactly the displayed row for* Classical.choice*,* Quot.sound*, and* propext *transported through H and routed through C. No consumer may replace that finite classifier by a broader host-dependency predicate, a theorem-inventory completeness claim, or a different forbidden-name set.*

Proof. Start from the dependency row D in Definition 526.1. Its displayed classifier names exactly Classical.choice, Quot.sound, and propext.

Use Definition 526.7. It fixes the inspected theorem names, reported proof ancestry, and pass-or-refuse result as the content carried by D.

Now apply Definition 526.3. The strict axiom-facing conclusion is local to the finite dependency report, so transport through H and replay through C cannot widen the forbidden set.

The non-escape theorem Definition 526.19 closes the boundary. It excludes host-kernel completeness and theorem-inventory completeness, leaving only the displayed three-name classifier. □

Theorem 526.22 (AuditGateBoundary verdict scope)**.** *Every verdict read from an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet factors through*

$$S, D, L, O, H, C, P, G, N.$$

The verdict may combine source-token refusal, strict dependency reporting, exact marker resolution, and origin/import separation, but it cannot be replaced by an ambient host claim about repository correctness, theorem inventory completeness, source admission, or semantic conservativity.

Proof. Project the carrier. The public audit faces are S, D, L, O, and the support rows are H, C, P, G, N.

Use Definition 526.6 and Definition 526.7 for the source and dependency verdict components. Use Definition 526.3 to bind their joint axiom-facing reading.

The remaining rows stay inside the same accepted packet. The carrier definition provides marker-resolution and origin/import rows, while the support rows transport, replay, package, delimit, and name the verdict surface.

Apply Definition 526.19. It blocks ambient host claims and leaves the verdict as a finite read through $S, D, L, O, H, C, P, G, N$ alone. □

Theorem 526.23 (AuditGateBoundary standard bridge)**.** *The accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet forms the standard audit-boundary schema: the public source-token row, axiom-purity dependency row, paper-Lean drift row, origin/import separation row, replay ledger, GAP boundary row, and local* NameCert *row are carried by one finite* BHist *surface and transported through the shared* H, C, P *support rows. The schema exports exactly that audit-boundary verdict surface. It exports no repository completeness theorem, theorem-inventory completeness claim, host semantic correctness statement, or semantic conservativity result for arbitrary imports.*

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryUp StdBridge`

Proof. Project the carrier of Definition 526.1. Its public audit faces are the source-token scan, axiom-purity dependency report, paper-Lean target audit, and origin/import separation ledger, and its support faces are the replay, transport, package, GAP, and local naming rows.

Apply Definition 526.22. Every verdict read is therefore a finite read through the displayed audit rows and the shared support rows.

Now use Definition 526.20 and Definition 526.21 for the source-token and forbidden-dependency components. Use the replay and origin/import rows named in the carrier for the remaining audit faces. These are precisely the rows required by the standard audit-boundary schema.

Finally apply Definition 526.19. The excluded host claims have no carrier coordinate, so the bridge is only the finite audit-boundary schema and not a repository-completeness or host-correctness theorem. □

Theorem 526.24 (AuditGateBoundary visible verdict exhaustion)**.** *Every audit verdict read of an accepted* $\mathsf{AuditGateBoundary}^{\uparrow}$ *packet* $A = (S, D, L, O, H, C, P, G, N)$ *is exhausted by the displayed source-token, dependency, marker-resolution, origin/import, transport, continuation, provenance, GAP, and local* NameCert *rows. The verdict may report only the finite surface carried by those rows. It does not assert repository completeness, theorem-inventory completeness, host-kernel correctness, or a semantic correctness theorem for arbitrary imports.*

Lean4 ✓ `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryVisibleVerdictExhaustion`

Proof. First apply Definition 526.22. It factors every verdict read through $S, D, L, O, H, C, P, G, N$, so the visible verdict already has a finite carrier surface.

Next apply Definition 526.20 and Definition 526.21. These identify the source-token query and the forbidden-dependency report as the displayed audit rows, rather than as host-wide kernel assertions.

Use Definition 526.23 to package the source-token, dependency, marker-resolution, origin/import, transport, continuation, provenance, GAP, and local naming rows as one standard audit-boundary surface.

Finally apply Definition 526.19. The non-escape row removes repository completeness, theorem-inventory completeness, host-kernel correctness, and semantic correctness for arbitrary imports, so the visible verdict is exhausted by the displayed packet alone. □

Theorem 526.25 (AuditGateBoundary formal-axis refusal)**.** *An* AuditGateBoundary$^{\uparrow}$ *mature paper packet exports only the displayed audit-gate carrier, classifier, refusal, transport,* Cont*,* Pkg*, and* NameCert *rows, together with the named Lean theorem target*

`BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryUp_StdBridge`.

This supports a bridgeChecked *verification-axis row for that declaration alone. It does not create host-kernel completeness, semantic conservativity for arbitrary imports, or theorem-inventory completeness.*

Proof. First read the mature paper surface through the chapter theorems. The carrier, NameCert obligations, source-token refusal, forbidden-dependency scope, marker-resolution boundary, origin-import replay, standard bridge row, non-escape row, and visible verdict exhaustion theorem all operate on the displayed packet rows.

Next apply the two-axis discipline of chapter 9203. The theory-axis grade records what this paper packet closes, while the formal verification axis is supplied only by Lean markers and closurestatus targets that resolve to real declarations.

The closurestatus target names the existing Lean declaration for the standard audit-boundary bridge. Therefore the packet may report the corresponding checked-bridge row, while host-kernel completeness, semantic conservativity for arbitrary imports, and theorem-inventory completeness remain outside the packet. □

Closure status for NameCert$_{\text{AuditGateBoundary}^{\uparrow}}$.

Constructive story: An AuditGateBoundary$^{\uparrow}$ packet is generated from a source-token scan, an axiom-purity dependency report, a paper-Lean target audit, an origin/import separation ledger, finite hsame transport, visible Cont routes, package provenance, a GAP boundary row, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AuditGateBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public packet binds the carrier, NameCert obligations, source-token refusal, forbidden-dependency scope, marker-resolution non-escape, origin-import replay exactness, replay ledger, Definition 526.22, Definition 526.23, Definition 526.24, and the replay and non-escape rows to one finite audit-boundary surface.

Formal verification: FormalStatus(NameCert$_{\text{AuditGateBoundary}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.AuditGateBoundaryUp.AuditGateBoundaryUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close host-kernel completeness, semantic conservativity for arbitrary imports, or theorem inventory completeness.

Upgrade path: No further formal-axis upgrade remains for the checked standard bridge.

The ClosedTermSubstitutionBoundary$^{\uparrow}$ hub reads its closed substitution route through chapter 537, chapter 765, chapter 532, and chapter 525 before the child files spell out the route.

Chapter 527

Closed-Term Substitution Boundary NameCert Construction

The $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet records the finite naming surface by which MetaCIC closed terms pass through shift and substitution without exporting a free-variable leak. It packages Term, $\mathsf{ClosedAt}$, shift, $\mathsf{substitute}$, closedness ledgers, audit-visible non-escape rows, transport data, continuation routes, package provenance, and the local $\mathsf{NameCert}$ row. The packet is not a host compiler theorem and does not assert subject reduction beyond the displayed closed-term boundary.

Definition 527.1 (Closed-term substitution boundary carrier)**.** *A* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N)$$

with the following visible rows:

1. *T is the displayed* Term *row for the term being substituted into;*
2. *V is the displayed* Term *row for the closed substituting value;*
3. *d is the de Bruijn depth row at which the boundary is inspected;*
4. *C_{src} is the* $\mathsf{ClosedAt}(d, T)$ *source-closedness witness row;*
5. *C_{val} is the* $\mathsf{ClosedAt}(0, V)$ *value-closedness witness row;*
6. *Q is the displayed* shift *boundary row recording that closed terms spend no future depth budget at the inspected boundary;*
7. *R is the displayed* $\mathsf{substitute}$ *boundary row recording that substitution at a closed boundary returns the same closed source row;*
8. *L is the finite closedness ledger recording which source, value, shift, and substitution rows have been read;*
9. *A is the audit-visible non-escape row recording that no free variable, host equality witness, or untracked compiler state is exported by the packet;*
10. *H stores componentwise* hsame *transports for the term rows, depth row, closedness rows, boundary-operation rows, ledger, audit row, package row, and local naming row;*
11. *C stores the displayed* Cont *routes by which consumers read those finite rows;*

12. *P is the* Pkg *provenance over* Term, ClosedAt, shift, substitute, BHist, hsame, Cont, Pkg, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}}$ *row.*

The carrier contains no hidden typing derivation, no unrestricted compiler correctness theorem, no host substitution semantics, and no coordinate for an unregistered free variable.

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionBoundaryUp.ClosedTermSubstitutionBoundaryTasteGate`
`single carrier alignment`

Definition 527.2 (Closed-term substitution boundary classifier)**.** *For accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packets S and S', the classifier* $\mathsf{ClosedTermSubstitutionBoundaryClassifier}(S, S')$ *holds exactly when the displayed* Term *rows, inspected depth row,* ClosedAt *witnesses,* shift *boundary row,* substitute *boundary row, closedness ledger, audit-visible non-escape row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row transport componentwise to the same finite boundary surface.*

The classifier does not compare host syntax trees by an unrecorded equality test, does not inspect non-displayed typing derivations, and does not import a compiler-wide semantic preservation theorem.

Lean4 (definition) `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClassifier`
Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClassifier`
`component transport`

Theorem 527.3 (Closed-term substitution boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutio}}$ *surface consists of five finite obligations:*

1. *carrier admission: accepted packets are exactly the finite* BHist *carriers of Definition 527.1;*
2. *classifier stability: Definition 527.2 is preserved by displayed* hsame *transports and* Cont *routes on the term rows, depth row, closedness witnesses, shift row, substitution row, ledger, audit row, package provenance, and local naming row;*
3. *shift boundary exactness: the* shift *row may be consumed only after the source closedness witness has been displayed, and it records that the closed boundary does not spend a future depth budget;*
4. *substitution boundary exactness: the* substitute *row may be consumed only after the source and value closedness witnesses have been displayed, and it records that substitution cannot manufacture a free-variable coordinate;*
5. *audit non-escape: the ledger and audit rows expose every closedness dependency required by the packet and export no unregistered free variable, hidden host equality, unchecked compiler state, or ambient subject-reduction theorem.*

Proof. Carrier admission is projection from Definition 527.1. The tuple displays the term rows, depth row, closedness witnesses, operation rows, ledger, audit row, transport and continuation data, package provenance, and local naming certificate.

Classifier stability follows from Definition 527.2. The classifier is defined only on the displayed rows, so componentwise hsame movement and recorded Cont reads cannot introduce a second substitution semantics or an untracked compiler state.

The shift and substitution obligations are the two operation rows Q and R. Their prerequisites are recorded in C_{src} and C_{val}, and the finite ledger L records that those witnesses have been read before the operation boundary is consumed.

Finally the audit row A fixes the non-escape claim. Since the carrier has no coordinate for a free-variable leak, hidden host equality, or ambient subject-reduction theorem, no consumer route can export such data from the local certificate. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryNamecertObligations`

Theorem 527.4 (Closed-term substitution boundary source closedness admission)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the source closedness row C_{src} *must be exposed before either the shift boundary row* Q *or the substitution boundary row* R *can be consumed.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySourceClosednessAdm`

Proof. Project the carrier of Definition 527.1. The displayed packet places C_{src} in the same finite BHist surface as Q and R.

Use the classifier of Definition 527.2 and the NameCert obligations of Definition 527.3. The classifier permits only componentwise transport of the displayed rows, while the operation obligations require source closedness before either boundary operation is read. □

Theorem 527.5 (Closed-term substitution boundary value closedness admission)**.** *For every accepted substitution-row read of a* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the value closedness row C_{val} *must be exposed before the substitution boundary row* R *can be consumed.*

Proof. Read the carrier coordinates from Definition 527.1. The value witness C_{val} is displayed as the closedness row for the substituting value, and the substitution row R is part of the same finite packet.

Apply Definition 527.2 and Definition 527.3. Their substitution boundary clause permits a read of R only after the source and value closedness witnesses have been displayed and recorded in the local ledger. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryValueClosednessAdmi`

Theorem 527.6 (Closed-term substitution boundary closed-value preservation)**.** *Any accepted substitution-row read that preserves the* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must keep the value closedness row* C_{val} *displayed through* R, L, A, H, C, P, N*. The read cannot replace* C_{val} *by host substitution semantics, hidden equality, or an ambient subject-reduction theorem.*

Proof. Project the carrier of Definition 527.1. The row C_{val} is a named coordinate of the same finite packet as the substitution row R, ledger L, audit row A, transport H, route C, provenance P, and local name N.

Use Definition 527.5 and Definition 527.7. The first admits the displayed value-closedness row, and the second requires any substitution-row read to expose that row before R is consumed.

Thus preservation of the boundary preserves the displayed C_{val} row. No carrier coordinate supplies host substitution semantics, hidden equality, or subject reduction. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosedValuePreserva`

Theorem 527.7 (Closed-term substitution boundary substitution-row exactness)**.** *In an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the substitution row R is read only through the finite surface $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, L, A, H, C, P, N$. It exports no free variable, host equality, unchecked compiler state, or subject-reduction theorem.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionRowExac`

Proof. Start from the carrier and classifier. Definition 527.1 lists every row available to the substitution boundary, and Definition 527.2 allows only displayed hsame transports and Cont routes across those rows.

Use the NameCert obligations of Definition 527.3. They force the read of R through $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ and the ledger and audit rows before any consumer may cite the substitution boundary.

The export exclusions are absence-of-coordinate consequences. The packet has no row for a free variable, host equality, unchecked compiler state, or subject-reduction theorem, and the local certificate cannot reconstruct one from L, A, H, C, P, N. □

Theorem 527.8 (Closed-term substitution boundary ledger non-escape)**.** *Every accepted* ClosedTermSubstitutionBo *packet exposes closed-term substitution only through the rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The ledger L and audit row A force every consumer read through the displayed $\mathsf{ClosedAt}$ *witnesses and boundary-operation rows before it may cite the packet. No consumer receives an unregistered free variable, a hidden host equality witness, unchecked compiler state, or a subject-reduction theorem not carried by the packet.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryLedgerNonEscape`

Proof. Read the carrier coordinates from Definition 527.1. The only rows available to a consumer are the two terms, the depth row, the closedness witnesses, the shift and substitution boundary rows, the ledger, the audit row, and the displayed provenance data.

Use the NameCert obligations of Definition 527.3. They require the closedness witnesses to be displayed before either operation row is consumed, and they place the non-escape check in the same finite packet.

The exclusions follow from absence of coordinates. A free-variable leak, hidden host equality witness, unchecked compiler state, or ambient subject-reduction theorem is not among the packet rows and cannot be reconstructed by hsame transport or a Cont read. □

Theorem 527.9 (Closed-term substitution boundary shift-ledger exhaustion)**.** *Every shift read consumed through an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is exhausted by the finite rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, L, A, H, C, P, N.$$

The shift boundary row Q may be read only after the term rows, inspected depth, source and value closedness witnesses, ledger row, audit row, transports, continuation routes, package provenance, and local $\mathsf{NameCert}$ *row have been displayed. It exports no unregistered free variable, host equality witness, unchecked compiler state, subject-reduction theorem, or host shift semantics.*

Proof. Project the carrier of Definition 527.1. The displayed rows include the two terms, inspected depth, source and value closedness witnesses, shift row, ledger, audit row, transports, continuation routes, package provenance, and local name.

Apply the classifier and NameCert obligations. The classifier permits only componentwise hsame transport and recorded Cont reads of those rows, while the shift boundary exactness clause requires the closedness witness and ledger entry before Q is consumed.

Finally use ledger non-escape. Since no row carries an unregistered free variable, host equality witness, unchecked compiler state, subject-reduction theorem, or host shift semantics, every shift read is exhausted by the displayed packet rows. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryShiftLedgerExhausti

Theorem 527.10 (Closed-term substitution boundary substitution-ledger exhaustion)**.** *Every substitution read consumed through an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet is exhausted by the finite rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The substitution boundary row R may be read only after the source and value closedness witnesses, the shift boundary row, the ledger row, and the audit row are visible inside the same carrier. It exports no unregistered free variable, hidden host equality, unchecked compiler state, subject-reduction theorem, or host substitution semantics.

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionLedgerE

Proof. Start with the carrier and classifier. The local packet displays the substitution row R only as part of the same finite surface as $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, L, A, H, C, P, N$, and the classifier admits only recorded transport of that surface.

Use the NameCert obligations for substitution boundary exactness. They require the source and value closedness witnesses to be displayed, and the ledger L records the closedness and operation reads before the substitution row can be consumed.

Apply ledger non-escape for the audit boundary. The audit row A rules out a free-variable leak, hidden host equality, unchecked compiler state, subject-reduction theorem, or host substitution semantics; none of these can be reconstructed from hsame transport or a Cont route. □

Theorem 527.11 (Closed-term substitution boundary self-compile route)**.** *A self-compilation consumer may use an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet only as the finite closed-boundary route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The consumer may replay the packet through BHist, hsame, Cont, Pkg, *and* NameCert *rows, but it cannot import subject reduction, compiler correctness, host substitution semantics, host syntax quotients, or an unchecked semantic model.*

Proof. Begin with the shift-ledger and substitution-ledger exhaustion rows. They identify the only finite routes by which the consumer may read closed shift and closed substitution through the packet.

Use the carrier, classifier, and NameCert obligations to repack those routes as a single closed-boundary service. The rows Q and R are both guarded by the displayed closedness witnesses, ledger, audit row, transports, continuations, package provenance, and local name.

Finally apply ledger non-escape. Subject reduction, compiler correctness, host substitution semantics, host syntax quotients, and unchecked semantic models have no coordinate in the packet, so a self-compilation consumer receives only the finite closed-boundary route. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySelfCompileRoute`

Theorem 527.12 (Closed-term substitution boundary consumer boundary)**.** *A self-compilation consumer may use an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet only as the finite boundary surface for closed* shift *and* substitute *reads. The consumer must factor through the displayed* Term *rows,* ClosedAt *witnesses, depth row, operation rows, closedness ledger, audit-visible non-escape row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

This boundary supplies the closed-term substitution step needed by the self-compilation story, but it does not export a full compiler correctness theorem, a subject-reduction package, a host syntax quotient, or a hidden semantic model.

Proof. Start with the carrier and classifier. The carrier of Definition 527.1 displays the finite rows, and the classifier of Definition 527.2 permits only componentwise transport and recorded consumer reads of those rows.

Apply Definition 527.3. The shift and substitution rows become available only after the required closedness witnesses and ledger entries have been displayed.

Finally use Definition 527.8. The consumer route cannot leave the finite packet to obtain compiler correctness, subject reduction, host quotient syntax, or a hidden semantic model, so the self-compilation use is exactly the closed-term substitution boundary. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerBoundary`

Theorem 527.13 (Closed-term substitution boundary BinderBudgetSeal factorization)**.** *Every* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *shift or substitution read that is used by a closed-term substitution consumer factors through the displayed* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Thus the binder-budget seal of chapter 532 may consume the closed shift and substitution facts only through the finite BHist *packet, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row of Definition 527.1.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryBinderSealFactoriza`

Proof. Start with a $\mathsf{BinderBudgetSeal}^{\uparrow}$ read that asks for closed shift or closed substitution at the inspected binder depth. The local carrier theorem Definition 527.12 restricts that read to the finite closed-term boundary surface.

The needed closedness witnesses are exactly C_{src} and C_{val}, and the permitted operation rows are exactly Q and R. By Definition 527.3, these rows are available only after the ledger L records the corresponding closedness read.

Finally apply Definition 527.8. Since the audit row A and the continuation row C admit no hidden binder convention, unregistered free-variable coordinate, or ambient compiler theorem, the binder-budget consumer factors through the displayed packet rows and no other substitution surface. □

Theorem 527.14 (Closed-term substitution boundary authorized-generator inversion)**.** *Any* AuthorizedGeneratorR *consumer of the closed-term substitution boundary must expose* $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ *before citing the boundary. Thus the read displays the term rows, source and value closedness witnesses, shift row, substitution row, ledger, audit row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row, while refusing full subject reduction, global compiler correctness, host substitution semantics, and quotient syntax.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuthorizedGenerato`

Proof. Project the carrier, NameCert obligations, source/value closedness admission, substitution-ledger exhaustion, and BinderBudgetSeal factorization. These rows exhaust the authorized-generator read and leave no coordinate for the refused global compiler data. □

Theorem 527.15 (ClosedTermSubstitutionCompiler boundary factorization)**.** *The* ClosedTermSubstitutionCompile *schedule of chapter 540 reads the same closed finite boundary before exposing any compiler-facing* substitute_closed *row. In particular, every accepted schedule read passes through the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

of ClosedTermSubstitutionBoundary$^{\uparrow}$*, then repacks the result through the compiler packet's own continuation and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryCompilerFactorizati`

Proof. Project the compiler-facing read to its closed source and closed value premises. The consumer boundary theorem Definition 527.12 identifies those premises with a read of the closed-term substitution boundary carrier rather than with an evaluator theorem.

The schedule can expose substitute_closed only after it reads C_{src}, C_{val}, and the substitution row R. The BHist ledger L and the Cont routes C record this order, while the hsame transports H keep the same finite rows under accepted packet movement.

The final compiler packet is therefore a repackaging step. The Pkg provenance P and local naming row N certify that the compiler-facing schedule has consumed the closed boundary, but the audit row A prevents the schedule from importing open-term subject reduction, host syntax equality, or a hidden semantic evaluator. □

Theorem 527.16 (Closed-term substitution boundary RecursorGenerator handoff)**.** *Every generated branch output of* RecursorGenerator$^{\uparrow}$ *in chapter 534 consumes closed-term substitution only through the finite* ClosedTermSubstitutionBoundary$^{\uparrow}$ *boundary. The handoff route is the displayed* BHist *packet with rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

followed by the recursor generator's own branch-output packaging.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecursorHandoff`

Proof. Consider a generated branch that invokes closed substitution while assembling its output. The branch has access to the boundary only through recorded Cont routes, so Definition 527.12 forces the substitution read to enter the closed-term boundary carrier.

Inside that carrier, Definition 527.3 supplies the required order: source closedness, value closedness, ledger entry, and then the displayed substitution boundary row. Componentwise hsame transport preserves the same rows when the branch output is moved across accepted packets.

The handoff cannot bypass the finite packet. By Definition 527.8, the packet exports no unregistered free variable, host equality witness, unchecked compiler state, or subject-reduction theorem, so the generated branch output receives only the certified closed-term substitution boundary. □

Theorem 527.17 (Closed-term substitution boundary authorized recursor audit route)**.** *Every* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *output that consumes closed substitution reaches it only through the displayed* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

together with the authorized recursor audit provenance and local naming rows. The authorized recursor may attach its own branch-output boundary, but it cannot read closed substitution through host equality, unchecked compiler state, hidden evaluator data, or subject reduction outside the boundary packet.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuthorizedRecursor`

Proof. Start with the carrier and NameCert obligations of Definition 527.1 and Definition 527.3. They display the closed source, closed value, operation, ledger, audit, transport, route, provenance, and naming rows that any substitution consumer must read.

Apply ledger non-escape and the self-compile route, Definition 527.8 and Definition 527.11. These results force closed shift and closed substitution to remain inside the finite boundary packet before a self-compilation consumer can cite them.

Now use the consumer boundary and recursor handoff, Definition 527.12 and Definition 527.16. The authorized recursor output is a downstream branch-output package over the same rows, so its audit provenance augments the route without replacing the closed substitution boundary. □

Theorem 527.18 (Closed-term substitution boundary output non-escape)**.** *Generated output rows downstream of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet cannot export free variables, host equality, unchecked compiler state, or subject reduction outside the packet. Every such output that cites closed shift or closed substitution must carry the displayed boundary rows, the audit row* A*, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryOutputNonescape`

Proof. Read the carrier and the local NameCert obligations. The rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ are the whole closed-term substitution surface, with no coordinate for an exported free variable, host equality witness, unchecked compiler state, or ambient subject-reduction theorem.

Use Definition 527.8 and Definition 527.12. They make the ledger and audit rows prerequisites for any consumer-facing closed shift or closed substitution read.

Finally apply Definition 527.16 and Definition 527.17. Generated branch output and authorized recursor output therefore inherit the same finite non-escape boundary instead of exporting a detached substitution or compiler theorem. □

Theorem 527.19 (Closed-term substitution boundary self-compile public package)**.** *The public self-compilation package supplied by* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *is exactly the assembly of closed shift, closed substitution, recursor handoff, and audit-visible non-escape rows over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

It exposes the closed-boundary service to downstream self-compilation consumers without exporting host substitution semantics, subject reduction, host syntax quotienting, unchecked compiler state, or an external evaluator.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySelfCompilePublicPa`

Proof. Begin with the carrier, NameCert obligations, and ledger non-escape theorem. They identify the displayed closedness witnesses, operation rows, ledger, audit row, transports, route, provenance, and local name as the complete boundary packet.

Next use the self-compile route and consumer boundary, Definition 527.11 and Definition 527.12. These rows make the closed shift and closed substitution service public only as a finite packet read, not as a host substitution semantics.

Finally add the recursor and audit-output routes. By Definition 527.16, Definition 527.17, and Definition 527.18, branch-output consumers receive the same packet with audit provenance and no export route for free variables, unchecked compiler state, subject reduction, or hidden evaluator data. □

Theorem 527.20 (Closed-term substitution boundary normalization-frontier handoff)**.** *Every MetaCIC normalization-frontier consumer of chapter 892 reads closed substitution from* ClosedTermSubstitutionBou *only through the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The handoff supplies the closed source row, closed value row, inspected depth, closedness witnesses, shift and substitution boundary rows, closedness ledger, audit row, componentwise transport, continuation route, provenance, and local NameCert *row, but it supplies no subject-reduction theorem, host substitution semantics, evaluator, or ambient normalization principle.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryNormalizationFronti`

Proof. Start with a normalization-frontier read that asks for closed substitution before exposing its frontier row. The consumer boundary of Definition 527.12 forces that read into the finite closed-term substitution packet rather than an external MetaCIC evaluator.

Inside the packet, the compiler factorization theorem identifies the same rows as the only schedule-facing substitution source. The frontier consumer therefore receives $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ as a closed boundary and not as a subject-reduction package.

Finally apply the recursor handoff theorem. It shows that generated branch output can carry the substitution read downstream only as the displayed boundary packet, so the normalization-frontier surface inherits no host substitution semantics or ambient normalization theorem beyond those rows. □

Theorem 527.21 (Closed-term substitution boundary compiler schedule lock)**.** *Every* ClosedTermSubstitutionComp *schedule read of chapter 540 locks onto the displayed* ClosedTermSubstitutionBoundary$^{\uparrow}$ *rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

before exposing any compiler-side substitute_closed *surface. The schedule may repack the read through its own compiler rows, but the closed substitution input remains the same finite boundary packet.*

Proof. Project the schedule read to the closed source, closed value, depth, and substitution rows it needs before substitute_closed can be exposed.

Apply Definition 527.12 and Definition 527.15. Together they make the compiler schedule a consumer of the displayed boundary rows, with C_{src}, C_{val}, Q, R, and L read before the compiler-facing surface appears.

The remaining rows A, H, C, P, N keep the schedule audit-visible, transported componentwise, routed by displayed continuations, and locally named. Thus the compiler schedule can expose substitute_closed only after locking onto the finite boundary, not by importing host substitution semantics or an ambient compiler-correctness theorem. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryCompilerScheduleLoc`

Theorem 527.22 (Closed-term substitution boundary recursor audit route)**.** *Every* AuthorizedGeneratorRecursor$^{\uparrow}$ *branch output of chapter 525 that consumes closed substitution must pass through the audit row A, the recursor-side boundary row G, and the local* NameCert *rows of both packets. The substitution read remains the* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

and the authorized recursor side may only attach its displayed generator boundary and local naming data.

Proof. Begin with the recursor handoff of Definition 527.16. Any branch output that invokes closed substitution already receives the displayed finite boundary packet, including the audit row A and local naming row N.

Next route the authorized recursor consumer through chapter 525. Its branch output adds the recursor-side boundary row G and its own local NameCert row, but it does not replace the closed substitution packet or bypass the audit row.

Finally use the consumer boundary and compiler factorization theorems. They keep the substitution component attached to $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ while the authorized recursor packet contributes only its displayed branch boundary and certificate rows. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecursorAuditRoute`

Theorem 527.23 (Closed-term substitution boundary audit route)**.** *Every audit-visible read of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet factors through the displayed audit surface A, H, C, P, N. In particular, a read that cites closed shift, closed substitution, ledger non-escape, or consumer admissibility must replay the audit row A, componentwise* hsame *transport H, displayed* Cont *route C,* Pkg *provenance P, and local* NameCert$_{\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}}$ *row N.*

The audit route exports no free-variable coordinate, host equality witness, unchecked compiler state, hidden evaluator, or ambient subject-reduction theorem.

Proof. Start with the carrier and classifier of Definition 527.1 and Definition 527.2. They make A, H, C, P, N the only audit, transport, route, provenance, and naming rows available to the packet.

Use the NameCert obligations of Definition 527.3. The local certificate requires the shift and substitution rows to be consumed only after their displayed closedness witnesses and ledger entries have appeared.

Finally apply ledger non-escape and the consumer boundary from Definition 527.8 and Definition 527.12. Since every consumer read is already confined to the finite closed-term boundary, the audit-visible part of that read is exactly the route through A, H, C, P, N. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditRoute

Theorem 527.24 (Closed-term substitution boundary shift-budget exhaustion)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the shift row* Q *consumes only the displayed source closedness witness* C_{src} *at depth* d*. Once* C_{src} *is read through the ledger* L*, the row* Q *records that closed shifting spends no hidden future depth budget and does not consult any second binder-depth source.*

Proof. Project the source term row T, inspected depth row d, source closedness witness C_{src}, shift row Q, and ledger row L from the carrier.

The NameCert obligations identify shift boundary exactness as a local obligation of the same packet. The row Q is available only after C_{src} has been displayed and recorded by L.

Use ledger non-escape to rule out any hidden budget source. The packet has no coordinate for a second depth counter, a host binder convention, or an external compiler theorem, so the closed shift consumes exactly the displayed source closedness witness. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryShiftBudgetExhausti

Theorem 527.25 (Closed-term substitution boundary free-variable non-escape)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the substitution row* R *cannot manufacture an unregistered free-variable coordinate after the displayed source and value closedness witnesses* C_{src} *and* C_{val} *have been read. Any substitution consumer receives the same finite boundary surface and no open-term residue outside* $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$*.*

Proof. Read the substitution prerequisites from the carrier. The row R is paired with the source term T, the closed value V, the inspected depth d, and the two closedness witnesses C_{src} and C_{val}.

Apply the NameCert substitution boundary obligation. It permits R to be consumed only after both closedness witnesses are displayed and after the ledger has recorded that read.

The ledger non-escape theorem then removes the only remaining escape route. Because no carrier coordinate names an unregistered free variable or open-term residue, neither hsame transport nor a displayed Cont read can create one after substitution. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionFreeVar

Theorem 527.26 (Closed-term substitution boundary depth row stability)**.** *Transporting an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet by componentwise* hsame *preserves the inspected depth row* d*. The transported packet may move the displayed term rows, closedness witnesses, operation rows, ledger, audit row, continuation routes, package provenance, and local name through the rows listed in Definition 527.1, but it cannot replace* d *by an unlisted binder budget.*

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryDepthRowStability

Proof. Project the carrier row of Definition 527.1. The inspected depth d is a named coordinate of the same packet as $T, V, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$.

Use the classifier definition and the NameCert obligations. Componentwise hsame transport is available only through H, and the classifier compares the displayed depth row together with the displayed term, closedness, operation, ledger, audit, route, provenance, and naming rows.

Apply ledger non-escape from Definition 527.8. Since no coordinate names a second binder budget or a host binder convention, transported consumers keep the same inspected depth row and cannot read an unlisted depth source. □

Theorem 527.27 (Closed-term substitution boundary closedness ledger coverage)**.** *Every consumer of the displayed shift row Q or substitution row R must read C_{src} and C_{val} through the closedness ledger L before the corresponding operation row is available. The shift consumer may ignore the value witness only as unused data, but it still receives a packet whose local ledger records the closedness surface for both source and value rows. The route exports no host equality, no open-term subject reduction, and no hidden compiler state.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosednessLedgerCov`

Proof. Begin with the NameCert obligations of Definition 527.3. They make shift boundary exactness depend on the displayed source closedness row and substitution boundary exactness depend on the displayed source and value closedness rows.

Read the ledger non-escape theorem. The ledger L is the finite row that records which closedness witnesses have been consumed before Q or R is exposed to a consumer route.

Use shift-budget exhaustion and substitution free-variable non-escape. The shift row cannot spend a hidden depth budget, and the substitution row cannot manufacture an unregistered free-variable coordinate after the ledger has displayed the closedness witnesses.

The exclusions are then coordinate exclusions. Host equality, open-term subject reduction, and hidden compiler state are absent from the carrier inventory, so the operation rows become visible only through C_{src}, C_{val}, L, and the displayed route rows. □

Theorem 527.28 (Closed-term substitution boundary operation row determinacy)**.** *For fixed rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, L, A, H, C, P, N$ of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the displayed shift row Q and substitution row R are the unique operation rows visible to any closed-term consumer. A consumer may transport or replay those rows through H and C, but it cannot substitute a host equality proof, an open-term subject-reduction theorem, or hidden compiler state for either operation row.*

Proof. First project the carrier. The only operation coordinates in Definition 527.1 are the shift boundary row Q and the substitution boundary row R.

Use the NameCert obligations and closedness ledger coverage. The local certificate exposes Q and R only after the closedness witnesses have been read through L, and it exposes no second operation coordinate under the same fixed rows.

Apply depth row stability, audit route, and ledger non-escape. Transport through H preserves the inspected depth row, audit-visible replay remains inside A, H, C, P, N, and the ledger excludes unregistered free variables, hidden host equality, unchecked compiler state, and ambient subject-reduction data.

Thus a closed-term consumer with the fixed carrier rows can see exactly the displayed Q and R rows. Any different operation row would require a carrier coordinate not present in the packet. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryOperationRowDetermi`

Theorem 527.29 (Closed-term substitution boundary consumer factorization)**.** *Every self-compilation consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet factors through the full displayed surface*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The consumer may use the packet for closed shift and closed substitution only by reading the term rows, depth row, source and value closedness witnesses, operation rows, closedness ledger, audit row, componentwise transports, continuation routes, package provenance, and local naming row listed in the carrier.

Proof. Begin with the carrier and classifier. They expose exactly the rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ and restrict accepted movement to componentwise transport and displayed route replay.

Use the NameCert obligations for the two operation rows. The shift row Q depends on the displayed source closedness witness, while the substitution row R depends on the displayed source and value closedness witnesses.

Apply the audit-route theorem, shift-budget exhaustion, and substitution free-variable non-escape. These three obligations force the consumer to spend no hidden depth budget, manufacture no unregistered free variable, and route every audit-visible read through A, H, C, P, N.

Finally invoke the consumer boundary of Definition 527.12. The self-compilation consumer therefore factors through the full finite boundary surface rather than through a detached compiler theorem, host syntax quotient, evaluator, or subject-reduction package. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerFactorizati`

Theorem 527.30 (Closed-term substitution boundary obligation route triad)**.** *The carrier, classifier, and* NameCert *obligations of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet route closed shift, closed substitution, and audit non-escape through the same finite packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The shift row Q, substitution row R, and audit row A may be consumed only after the displayed closedness witnesses and ledger rows have been read through this packet.

Proof. Project the carrier and classifier. They expose the term rows, depth row, closedness witnesses, operation rows, ledger row, audit row, componentwise transports, continuation routes, package provenance, and local naming row as one accepted packet.

Use the NameCert obligations to route the operations. Closed shift reads the source closedness witness, closed substitution reads the source and value closedness witnesses, and both operation rows remain tied to the same ledger row L.

Finally apply ledger non-escape and the audit consumer boundary. The audit row A reads the same displayed history, transport, route, provenance, and naming rows, so the three obligations cannot split into independent compiler, substitution, or audit surfaces. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryObligationRouteTria`

Theorem 527.31 (Closed-term substitution boundary kernel scope package)**.** *The only scoped primitives consumed by the* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet are* Term*,* ClosedAt*,* shift*,* substitute*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. In particular, the scoped package contains no host syntax quotient, evaluator, subject-reduction theorem, compiler state, semantic environment, or external substitution relation.*

Proof. Read the finite carrier rows of Definition 527.1. The term, value, depth, closedness, shift, substitution, ledger, audit, transport, continuation, package, and naming rows exhaust the local inventory.

Then apply the obligation route triad of Definition 527.30. It forces closed shift, closed substitution, and audit non-escape to use the same displayed packet rather than an additional kernel primitive.

The exclusions follow from the consumer boundary and operation row determinacy. A host quotient, evaluator, subject-reduction theorem, compiler state, semantic environment, or external substitution relation would require a further scoped primitive not listed in the carrier. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryKernelScopePackage

Theorem 527.32 (Closed-term substitution boundary closedness consumer export)**.** *Every downstream self-compilation consumer may cite closed shift and closed substitution from an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet only through the displayed witnesses* C_{src}*,* C_{val}*, the operation rows* Q *and* R*, and the closedness ledger* L*. The exported route is therefore the displayed closedness boundary, not a host substitution semantics or ambient compiler-correctness theorem.*

Proof. Start with consumer factorization from Definition 527.29. Every self-compilation read already factors through the rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$.

Use closedness ledger coverage and operation row determinacy. They make C_{src}, C_{val}, Q, R, and L the only visible closedness and operation rows available to such a consumer.

Finally invoke the kernel scope package of Definition 527.31. Since no host substitution semantics or ambient compiler-correctness theorem is present in the scoped primitive list, downstream citation of closed shift and closed substitution must pass through the displayed witnesses and ledger. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosednessConsumerE

Theorem 527.33 (Closed-term substitution boundary compiler consumer lock)**.** *Every* ClosedTermSubstitutionCom *consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must pass through the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The compiler consumer may cite the compiler-side sibling surface of chapter 540 only after these boundary rows have been displayed; it cannot replace them by a host substitution semantics, ambient compiler-correctness theorem, or detached compiler carrier.

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryCompilerConsumerLoc

Proof. Project the accepted boundary carrier. The projection exposes the term row T, value row V, depth row d, source and value closedness witnesses, operation rows Q, R, ledger row L, audit row A, transport row H, route row C, provenance row P, and local naming row N as one finite packet.

Route the consumer through the local NameCert obligations. They make the closed shift and closed substitution rows readable only through the displayed closedness witnesses and ledger, so a compiler consumer cannot detach Q or R from $C_{\mathsf{src}}, C_{\mathsf{val}}$, and L.

Finally compare with the sibling compiler chapter. The route into chapter 540 is a consumer route out of the displayed boundary packet, not a replacement source for the packet, because no host substitution semantics or ambient compiler-correctness theorem is present among the boundary rows. □

Theorem 527.34 (Closed-term substitution boundary binder-budget depth lock)**.** *In every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the inspected depth row* d *and the source/value closedness witnesses* $C_{\mathsf{src}}, C_{\mathsf{val}}$ *lock the binder-budget surface before either operation row* Q *or* R *can be consumed. The resulting budget route cites the sibling binder-budget surface of chapter 532 only through the displayed boundary rows.*

Proof. Read the depth and closedness coordinates from the carrier. The row d is not a free ambient natural-number budget; it is the displayed depth coordinate paired with T, V, C_{src}, and C_{val} inside the accepted packet.

Use operation-row routing. The shift row Q consumes the source closedness witness at the displayed depth, and the substitution row R consumes both the source and value closedness witnesses before producing any closedness-preserving operation read.

The binder-budget sibling is therefore reached only after this local projection. A consumer may compare the resulting budget surface with chapter 532, but the comparison is routed through $d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ and supplies no hidden binder depth or host substitution step. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryBinderBudgetDepthLo`

Theorem 527.35 (Closed-term substitution boundary recursor-generator scope lock)**.** *Every* $\mathsf{RecursorGenerator}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet may use the substitution boundary only after the closedness ledger and operation rows* Q, R, L *have been displayed with*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, A, H, C, P, N.$$

The sibling recursor-generator surface of chapter 534 is therefore a downstream scope consumer, not a host recursion principle or external substitution semantics.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecursorGeneratorSc`

Proof. Start from the closedness consumer export. It already forces downstream citation of closed shift and closed substitution through $C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R$, and L.

Add the audit and route rows. The audit row A, componentwise transport row H, continuation row C, package row P, and naming row N keep the operation read inside the same finite boundary packet.

Only then may a recursor-generator consumer cite chapter 534. Since the boundary packet contains no host recursion principle, external substitution semantics, or detached generator carrier, that sibling citation is a scoped consumer route over the displayed ledger and operation rows. □

Theorem 527.36 (Closed-term substitution boundary shift-substitution ledger lock)**.** *Every closed shift or closed substitution read from an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet passes through the displayed source closedness witness* C_{src}*, value closedness witness* C_{val}*, shift row* Q*, substitution row* R*, closedness ledger* L*, and audit row* A*. The route may transport these rows by* H *and replay them through* C*, but it cannot expose an operation row before the corresponding closedness witness, ledger entry, and audit row have been displayed.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryShiftSubstitutionLe`

Proof. Project the carrier and classifier of Definition 527.1 and Definition 527.2. The only closedness and operation coordinates available to the packet are C_{src}, C_{val}, Q, R, L, and A together with the transport, route, provenance, and naming rows.

Use closedness ledger coverage and operation-row determinacy, Definition 527.27 and Definition 527.28. They make Q and R readable only after the displayed closedness witnesses and ledger row have fixed the operation surface.

Finally apply the audit route and consumer factorization theorems, Definition 527.23 and Definition 527.29. The audit row A remains on the same finite route as the operation rows, so every shift or substitution read is locked to C_{src}, C_{val}, Q, R, L, and A before any consumer can cite it. □

Theorem 527.37 (Closed-term substitution boundary audit route coverage)**.** *The audit route of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet covers every consumer read of the closedness witnesses, shift row, substitution row, closedness ledger, and non-escape row. In particular, any consumer of* C_{src}*,* C_{val}*,* Q*,* R*,* L*, or* A *factors through the finite surface*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditRouteCoverage`

Proof. Begin with the carrier/classifier definitions. They expose the closedness, operation, ledger, audit, transport, route, provenance, and naming rows as the complete consumer-visible surface of the packet.

Apply the audit route theorem together with ledger non-escape, Definition 527.23 and Definition 527.8. The audit-visible route records the same closedness and ledger dependencies and has no coordinate for an unregistered free variable, hidden host equality, unchecked compiler state, or ambient subject-reduction theorem.

Use operation-row determinacy and consumer factorization. Operation reads of Q and R are the displayed rows under fixed closedness and ledger data, and every self-compilation consumer reaches them only through the listed finite packet. Hence the audit route covers all closedness, shift, substitution, ledger, and non-escape reads. □

Theorem 527.38 (Closed-term substitution boundary NameCert export totality)**.** *The local* $\mathsf{NameCert}$ *export of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is exactly the finite packet rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

It exports no host substitution semantics, no ambient subject-reduction theorem, no host syntax quotient, no hidden evaluator, and no compiler state outside those rows.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryNameCertExportTotal`

Proof. Read the carrier, classifier, and NameCert obligation rows. The definitions and Definition 527.3 make the local certificate range over the displayed term, closedness, operation, ledger, audit, transport, continuation, package, and naming coordinates.

Apply the shift-substitution ledger lock and audit route coverage. Together they show that every operation read and every audit-visible consumer read has already passed through the same finite packet rows, including C_{src}, C_{val}, Q, R, L, and A.

Finally use operation-row determinacy, consumer factorization, and recursor-generator scope lock, Definition 527.28, Definition 527.29, and Definition 527.35. These theorems leave no export route for host substitution semantics, an ambient subject-reduction theorem, a host syntax quotient, a hidden evaluator, or compiler state outside the displayed packet. Thus the local $\mathsf{NameCert}$ export is total for exactly the finite boundary rows. □

Theorem 527.39 (Closed-term substitution boundary consumer route determinacy)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, any consumer route preserving*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

determines the same closed-term substitution boundary surface. The route may read the displayed term rows, closedness witnesses, operation rows, ledger, audit row, componentwise hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, but it cannot switch to host substitution semantics, a hidden evaluator, a host syntax quotient, or an ambient subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerRouteDeterm`

Proof. Project the carrier and classifier. They expose exactly the displayed term, depth, closedness, operation, ledger, audit, transport, continuation, provenance, and naming rows.

Apply Definition 527.28. The operation rows Q and R are fixed once $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ and the ledger row are fixed.

Use Definition 527.29 and Definition 527.38. Every consumer route preserving the displayed rows therefore factors through the same finite boundary surface, and the local certificate exports no host substitution semantics or ambient subject-reduction coordinate. □

Theorem 527.40 (Closed-term substitution boundary ledger-route separation)**.** *In an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, ledger reads of* L, A *are separate from operation-row reads of* Q, R *and must precede any exported operation boundary. The ledger rows record closedness and audit visibility for the displayed source and value witnesses, while* Q *and* R *remain the closed shift and closed substitution operation rows routed through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryLedgerRouteSeparati`

Proof. Begin with Definition 527.36. It places every closed shift or closed substitution read behind the source and value closedness witnesses, the closedness ledger L, and the audit row A.

Apply Definition 527.37. The audit-visible route covers reads of C_{src}, C_{val}, Q, R, L, and A, but it keeps the ledger and audit rows as the consumer-visible precondition for operation export.

The separation follows by coordinate inspection. The rows L, A are ledger and audit coordinates, while Q, R are operation coordinates; because the exported boundary factors through the displayed route, no consumer can expose Q or R before the ledger and audit reads have been fixed. □

Theorem 527.41 (Closed-term substitution boundary closed-value shift totality)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the closed-value shift read is total only along the displayed C_{val}*,* Q*,* L*, and* A *rows inside the same* BHist *packet. The read may use componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row, but it has no route through host substitution semantics, host syntax equality, hidden compiler state, or an ambient subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosedValueShiftTot`

Proof. Project the carrier and classifier of Definition 527.1 and Definition 527.2. They expose the closed value row C_{val} and the shift row Q as coordinates of the same finite BHist surface, transported only by the displayed hsame and Cont data.

Use the NameCert obligations. The shift boundary exactness clause makes the closedness rows and ledger row prerequisites for the operation read, while the package and naming rows record that the consumer is reading the local packet rather than an external substitution theorem.

Finally apply Definition 527.40. The ledger row L and audit row A must be fixed before Q is exported, so closed-value shift totality is exactly totality through the displayed closedness and audit rows and through no host semantic route. □

Theorem 527.42 (Closed-term substitution boundary substitute-closed totality)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the substitute-closed read is total only after the source and value closedness rows C_{src} *and* C_{val}*, the substitution row* R*, the closedness ledger* L*, and the audit row* A *have been displayed inside the same* BHist *packet. The only transport and consumer routes available are the componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row of the packet.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstituteClosedTot`

Proof. Read the carrier. The rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ are the whole substitution-boundary surface, and the classifier admits only componentwise movement of those rows.

Apply the NameCert substitution boundary obligation. It allows a read of R only after both closedness witnesses have been displayed and recorded in the finite ledger.

Use Definition 527.40 to separate the ledger and audit rows from the operation row. Since L and A precede the export of R, the substitute-closed totality claim is a closedness route through the packet and not a subject-reduction theorem or host substitution semantics. □

Theorem 527.43 (Closed-term substitution boundary audit-scope closure)**.** *The audit scope of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is closed by exactly the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Every audit-visible closed shift or substitute-closed consumer factors through the finite BHist *carrier, the classifier, the closedness rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and no consumer escapes to host substitution semantics, host syntax equality, hidden compiler state, or an ambient subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditScopeClosure`

Proof. Start from the carrier, classifier, and NameCert obligations. They list the term rows, depth row, closedness witnesses, operation rows, ledger, audit row, transport rows, continuation routes, package provenance, and naming row as the complete packet surface.

Apply the two totality rows above. Closed-value shift reads and substitute-closed reads are available only through C_{src}, C_{val}, Q, R, L, and A inside the same finite packet.

Finish with Definition 527.40. The ledger and audit rows cannot be bypassed before an operation boundary is exported. Therefore the audit scope closes over the displayed packet rows and does not import host substitution semantics, host equality, compiler state, or subject reduction. □

Theorem 527.44 (Closed-term substitution boundary audit-boundary factorization)**.** *Every audit-visible consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet factors through the carrier, ledger non-escape, shift-substitution ledger lock, and* $\mathsf{NameCert}$ *export totality. Consequently the audit boundary is the finite route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and not an appeal to host substitution semantics, subject reduction, a host syntax quotient, hidden evaluator state, or compiler state outside the displayed packet.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditBoundaryFactor`

Proof. First read the carrier of Definition 527.1. It supplies the finite BHist packet and the surrounding hsame, Cont, Pkg, and NameCert rows.

Apply Definition 527.8. The ledger route has no coordinate for unregistered free variables, hidden host equality, unchecked compiler state, or ambient subject reduction.

Next apply Definition 527.36. It forces operation reads through C_{src}, C_{val}, Q, R, L, and A before any exported closed shift or closed substitution boundary is visible.

Finally apply Definition 527.38. The local certificate exports exactly the displayed finite rows, so every audit-visible consumer factors through that boundary and no host substitution surface is added. □

Theorem 527.45 (Closed-term substitution boundary root audit totality)**.** *Every root audit consumer of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet factors through the finite route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The root audit surface is therefore exhausted by the displayed term rows, closed-value rows, depth row, closedness rows, closed shift and substitution rows, ledger rows, audit row, componentwise transports, continuation routes, package provenance, and local NameCert *row.*

Proof. Begin with Definition 527.37. It identifies every audit-visible closed shift or substitute-closed read with the finite carrier, classifier, closedness rows, transport rows, route rows, package row, and naming row of the accepted packet.

Next use Definition 527.38. The local NameCert exports exactly $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ and has no additional coordinate for host substitution semantics, host equality, hidden compiler state, or ambient subject reduction.

Finally apply Definition 527.44. Since every audit-visible consumer factors through the same carrier and export boundary, every root audit consumer is total over the displayed route and nowhere else. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootAuditTotality`

Theorem 527.46 (Closed-term substitution boundary root substitution scope)**.** *For every accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the root substitution-scope obligation is the finite read of the substitution row R only after the source and value closedness rows C_{src} and C_{val} have been displayed in the same BHist *packet. The read is transported only by the recorded* hsame *row H and consumed only through the displayed* Cont *route C under the* Pkg *provenance P and local* NameCert *row N.*

Thus a downstream root consumer receives the substitution boundary through

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, R, L, A, H, C, P, N$$

and not through host substitution semantics, subject reduction, hidden compiler state, or an unregistered free-variable coordinate.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootSubstitutionSco`

Proof. Start with the local NameCert obligations of Definition 527.3. They make a substitution-row read depend on the displayed source and value closedness rows before R can be consumed.

Next apply ledger non-escape, Definition 527.8. The ledger L and audit row A keep the substitution read inside the finite carrier and prevent a route to host equality, hidden compiler state, or a subject-reduction theorem.

Finally use Definition 527.45 and Definition 527.113. Their root routes package the same BHist, hsame, Cont, Pkg, and NameCert rows for downstream consumers, so the substitution-scope obligation is exactly the displayed finite read of R. □

Theorem 527.47 (Closed-term substitution boundary root shift scope)**.** *For every accepted* ClosedTermSubstitution *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the root shift-scope obligation is the finite read of the shift row Q only after the source closedness row C_{src} has been displayed in the same BHist *packet. The read is transported only by the recorded* hsame *row H and consumed only through the displayed* Cont *route C under the* Pkg *provenance P and local* NameCert *row N.*

Thus a downstream root consumer receives the shift boundary through

$$T, V, d, C_{\mathsf{src}}, Q, L, A, H, C, P, N$$

and not through a hidden depth convention, host substitution semantics, subject reduction, hidden compiler state, or an unregistered free-variable coordinate.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootShiftScope`

Proof. Begin with the NameCert obligations of Definition 527.3. Their shift exactness clause makes C_{src} the prerequisite for consuming the closed shift row Q.

Use ledger non-escape from Definition 527.8. Since the ledger and audit rows have no coordinate for a hidden depth convention, host substitution semantics, compiler state, or subject reduction, the shift read remains inside the displayed packet.

Close with Definition 527.45 and Definition 527.113. The root handoff exports the same BHist, hsame, Cont, Pkg, and NameCert route, so the shift-scope obligation is separated from the substitution-scope obligation while sharing the same finite packet boundary. □

Theorem 527.48 (Closed-term substitution boundary root-unblock obligation package)**.** *The root-unblock surface of* ClosedTermSubstitutionBoundary$^{\uparrow}$ *is the pair of operation-specific obligations carried by the displayed packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N).$$

The shift side is the finite BHist, hsame, Cont, Pkg, *and* NameCert *route of Definition 527.47. The substitution side is the finite* BHist, hsame, Cont, Pkg, *and* NameCert *route of Definition 527.46.*

Consequently downstream self-compilation, recursor, and compiler-schedule consumers may cite the root-unblock package only as the finite obligation packet

$$(C_{\mathsf{src}}, Q, L, A, H, C, P, N) \quad \textit{and} \quad (C_{\mathsf{src}}, C_{\mathsf{val}}, R, L, A, H, C, P, N),$$

with both sides grounded in Definition 527.3, Definition 527.8, Definition 527.45, and Definition 527.113.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootUnblockObligati`

Proof. First isolate the shift side. Apply Definition 527.47. It packages the Q read through C_{src}, L, A, H, C, P, and N.

Next isolate the substitution side. Apply Definition 527.46. It packages the R read through C_{src}, C_{val}, L, A, H, C, P, and N.

The shared root boundary is supplied by Definition 527.45 and Definition 527.113. These theorems ensure that both operation-specific routes remain in the same displayed BHist carrier with componentwise hsame transport, displayed Cont consumption, Pkg provenance, and local NameCert authority.

Finally use Definition 527.8. Since neither side has a coordinate for host substitution semantics, hidden compiler state, subject reduction, or an unregistered free variable, the combined package is exactly the finite root-unblock obligation surface. □

Theorem 527.49 (Closed-term substitution boundary root closedness gate)**.** *For every accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the source closedness row C_{src} gates every closed shift read through Q, and the pair $C_{\mathsf{src}}, C_{\mathsf{val}}$ gates every closed substitution read through R. Both operation reads remain in the displayed carrier, classifier, ledger, audit, hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Read the carrier and classifier of Definition 527.1 and Definition 527.2. They display the term row, value row, depth row, source and value closedness rows, shift row, substitution row, ledger row, audit row, transports, routes, package, and local naming row as one finite packet.

Apply Definition 527.3. Its closed-shift clause makes C_{src} the gate for Q, while its closed-substitution clause makes C_{src} together with C_{val} the gate for R.

Finally use Definition 527.8 and Definition 527.12. They keep both operation reads inside the displayed ledger and consumer boundary, with no appeal to host substitution semantics, subject reduction, or unchecked compiler state. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootClosednessGate`

Theorem 527.50 (Closed-term substitution boundary root substitution non-escape)**.** *Every root substitution read of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet stays inside*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

It may consume the substitution row R only through the displayed closedness gates, the finite ledger L, the audit row A, componentwise hsame *transport H, displayed* Cont *route C, package provenance P, and local* NameCert *row N.*

Proof. Begin with Definition 527.49. It supplies the source and value closedness gates required before R can be read.

Use Definition 527.8. The ledger and audit rows have no coordinate for an unregistered free variable, host equality, unchecked compiler state, or ambient subject reduction.

Next apply Definition 527.12. It forces every consumer-facing closed substitution read through the same finite boundary packet.

Finally cite the existing root package rows Definition 527.46 and Definition 527.48. They identify the downstream root route as the same $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ surface, so the substitution read cannot escape to host substitution semantics. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootSubstitutionNor`

Theorem 527.51 (Closed-term substitution boundary root audit gate)**.** *The audit row A gates every root read of the closed-term substitution boundary. A read that is not accepted by the displayed audit route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

cannot be consumed as a free-variable leak check, host equality, unchecked compiler state, subject reduction theorem, or host substitution semantics.

Proof. Project the accepted packet through Definition 527.1 and the local obligations of Definition 527.3. The audit row A is part of the same finite carrier as the closedness, shift, substitution, ledger, transport, route, package, and naming rows.

Apply Definition 527.8. It states that the finite ledger and audit surface do not export unregistered free variables, hidden host equality, unchecked compiler state, or subject reduction.

Use Definition 527.44 and Definition 527.12. Together they make every audit-visible consumer factor through the displayed boundary rows.

Finally apply Definition 527.50. Since substitution reads themselves remain inside the audited packet, the root audit gate prevents host substitution semantics from appearing as an additional consumer coordinate. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootAuditGate`

Corollary 527.52 (Closed-term substitution boundary root closedness strict local obstruction)**.** *Let O be a displayed failure, refusal, separation, or countermodel row attached to a consumer-facing converse or reuse attempt for a root substitution read of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N).$$

If the attempt uses the root closedness boundary, then O is a strict local obstruction. It can block only the named route that tries to consume R without the displayed source and value closedness gates, or the named reuse route that tries to read the closed-substitution row outside

$$C_{\mathsf{src}}, C_{\mathsf{val}}, R, L, A, H, C, P, N.$$

Every consumer-facing admissible use factors through those rows inside the displayed packet, and every unlisted coordinate is rejected. Hence O cannot be reinterpreted as a global impossibility statement, a host theorem, ambient subject reduction, host substitution semantics, unchecked compiler state, or hidden external validation.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootClosednessStric`

Proof. Apply Definition 527.49. It gives the only closedness gates available at the root boundary: C_{src} gates a closed shift read through Q, and the pair $C_{\mathsf{src}}, C_{\mathsf{val}}$ gates a closed substitution read through R. The same theorem keeps both operation reads inside the displayed carrier, classifier, ledger, audit, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Thus a converse or reuse attempt that cites the root closedness boundary has no independent closedness source and no operation coordinate before these gates are exposed.

Now use Definition 527.50. Every root substitution read stays inside

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

and it may consume R only through the displayed closedness gates, the ledger L, the audit row A, componentwise hsame transport H, displayed Cont route C, package provenance P, and local NameCert row N. Therefore any consumer-facing use of the alleged converse or reuse route must factor through exactly the listed closedness, substitution, ledger, audit, transport, route, provenance, and naming rows.

The obstruction row O has only this local target. If the attempted route asks to read R before C_{src} and C_{val} have gated it, then O blocks that named closedness failure. If the attempted route asks to reuse R beyond L, A, H, C, P, N, then O blocks that named non-escape failure. If the attempted route asks for a global impossibility statement, a host theorem, ambient subject reduction, host substitution semantics, unchecked compiler state, hidden external validation, or any other unlisted coordinate, the two cited root rows supply no projection for that request. The row is therefore a strict local obstruction to the named converse or reuse route and not an external validation principle. □

Theorem 527.53 (Closed-term substitution boundary source totality)**.** *Every accepted* ClosedTermSubstitutionBou *root-unblock read used by* AuthorizedGeneratorRecursor$^{\uparrow}$*,* BinderBudgetSeal$^{\uparrow}$*, or* ClosedTermSubstitutionCompiler$^{\uparrow}$ *must pass through the complete finite source surface*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The read exposes the BHist *carrier, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row for the closed source, closed value, inspected binder depth, shift boundary, substitution boundary, closedness ledger, and audit non-escape row. It exposes no open-term source coordinate, host syntax quotient, unchecked evaluator, or subject-reduction theorem.*

Proof. Start with the carrier and classifier of Definition 527.1 and Definition 527.2. They list the only rows available to a root-unblock consumer and restrict movement of those rows to componentwise hsame transport and displayed Cont replay.

Use the source triad and operation route from Definition 527.60 and Definition 527.61. These rows force the closed source, closed value, and inspected depth to be fixed before either Q or R is consumed by a generator, binder-budget, or compiler-facing consumer.

Finally apply the audit non-escape surface of Definition 527.62. Since no open source, host quotient, evaluator, or subject-reduction coordinate is present in the packet, totality of the root-unblock source surface is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySourceTotality`

Theorem 527.54 (Closed-term substitution boundary recursive readback)**.** *An* AuthorizedGeneratorRecursor$^{\uparrow}$ *branch may read a* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet only by replaying the finite route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The recursive readback must cite the BinderBudgetSeal closed-boundary factorization before it may expose the ClosedTermSubstitutionCompiler schedule row, and the compiler-facing row remains a repackaging of the same closed source, value, shift, substitution, ledger, and audit surface.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecursiveReadback`

Proof. First read the generated branch through the recursor handoff of Definition 527.16. That handoff already names the finite closed-term substitution packet as the only source available to a generated branch output.

Next apply the BinderBudgetSeal factorization from Definition 527.13. The binder-budget read can consume closed shift and closed substitution only after the closedness witnesses and operation rows are present in the same finite ledger.

Finally use the ClosedTermSubstitutionCompiler factorization of Definition 527.15. The compiler schedule repacks the already displayed boundary rows through its own route; it does not add a host evaluator, open-term substitution theorem, unchecked compiler state, or subject-reduction package. □

Theorem 527.55 (Closed-term substitution boundary downstream non-escape)**.** *Every downstream consumer that combines* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{BinderBudgetSeal}^{\uparrow}$, *and* ClosedTermSubstitutionComp *dependencies receives only the displayed* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The BHist, hsame, Cont, Pkg, *and* NameCert *rows carry the accepted closedness, operation, ledger, and audit data across those dependencies, while the downstream surface exports no unregistered free variable, hidden binder convention, host substitution semantics, host syntax quotient, unchecked compiler correctness theorem, or ambient subject reduction.*

Proof. Package the source-totality row of Definition 527.53. It fixes the complete finite source surface before any downstream dependency is allowed to consume the packet.

Then package the recursive readback row of Definition 527.54. The generator branch, binder-budget factorization, and compiler schedule all replay the same closed-boundary rows through displayed transports, continuations, provenance, and local naming data.

Apply the ledger non-escape theorem Definition 527.8. The excluded objects have no coordinate in the carrier, and none can be produced by hsame transport, Cont replay, Pkg provenance, or a local NameCert row. Hence the downstream handoff cannot leave the displayed closed-term substitution boundary. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryDownstreamNonescape`

Corollary 527.56 (Closed-term substitution boundary source-total strict obstruction)**.** *Let O be a displayed failure, refusal, separation, or countermodel row attached to a named converse or reuse route for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *root-unblock read. If the route is used by* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{BinderBudgetSeal}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$, *or by a downstream consumer combining those dependencies, then O is a strict local obstruction. It can block only the named attempt that factors through*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and then tries to omit, replace, detach, or bypass one of those displayed rows or one of the structural BHist, hsame, Cont, Pkg, *or* NameCert *rows. The same row cannot be reinterpreted as a global impossibility statement, host theorem, hidden external validation, unregistered free-variable route, hidden binder convention, host substitution semantics, host syntax quotient, unchecked evaluator, unchecked compiler correctness theorem, or ambient subject reduction.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySourceTotalStrictOb`

Proof. Apply the source-totality theorem Definition 527.53. Every accepted root-unblock read used by the generator, binder-budget, or compiler-facing consumers must first pass through the complete finite source surface

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

That surface exposes the carrier, componentwise transport rows, displayed continuation routes, package provenance, and local naming row for the closed source, closed value, inspected depth, operation rows, ledger, and audit row. It exposes no open source coordinate, host syntax quotient, unchecked evaluator, or subject-reduction theorem. Thus a displayed obstruction row attached to a root-unblock attempt has an available domain only over the listed finite rows and their displayed BEDC transport, route, provenance, and naming structure.

Now apply downstream non-escape, Definition 527.55. After the generator, binder-budget, and compiler dependencies are combined, the downstream consumer still receives only the same displayed $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ route. The carried data are exactly the accepted closedness, operation, ledger, and audit rows transported by BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, while the downstream surface exports no unregistered free variable, hidden binder convention, host substitution semantics, host syntax quotient, unchecked compiler correctness theorem, or ambient subject reduction.

Therefore any consumer-facing use of O must factor through the source-total route and the downstream non-escape route. If the named converse or reuse route asks to omit a displayed coordinate, replace it by a host coordinate, detach it from the closed source surface, or bypass the listed structural rows, then O blocks that named local attempt. Every stronger interpretation asks for a coordinate that neither source totality nor downstream non-escape exports. Hence O has force only as the strict local obstruction to the named converse or reuse route, and it cannot supply a global impossibility statement, host theorem, hidden external validation, or any external validation route outside the displayed closed-term substitution boundary. □

Theorem 527.57 (Closed-term substitution boundary root closedness budget exhaustion)**.** *Every root consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must expose the source and value closedness witnesses* C_{src} *and* C_{val} *before any closed-shift row or substitute-closed row is exported. The root read is therefore budgeted over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and not over host closedness, host substitution semantics, hidden evaluator state, or a subject-reduction theorem.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootClosednessBudge`

Proof. Start from the carrier and classifier of Definition 527.1 and Definition 527.2. They expose the displayed term rows, depth row, closedness witnesses, operation rows, audit row, transports, continuation routes, package provenance, and local naming row as one finite packet.

Use Definition 527.3. The $\mathsf{NameCert}$ obligations certify the closedness witnesses as packet coordinates before any exported operation boundary can be consumed.

Apply Definition 527.8, Definition 527.9, and Definition 527.10. These rows force closed-shift and substitute-closed reads through the displayed closedness budget, so no root consumer can reach Q or R without first passing through C_{src} and C_{val}. □

Theorem 527.58 (Closed-term substitution boundary root shift-substitution totality)**.** *Every root-facing closed shift or substitute-closed request for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is accounted for by the displayed route*

$$Q, R, L, A, H, C, P, N.$$

The route is total for root shift/substitution reads once the term rows, depth row, and closedness budget have been exposed.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootShiftSubstituti`

Proof. First apply Definition 527.57. It places every root read behind $T, V, d, C_{\mathsf{src}}$, and C_{val} before operation export.

Next use Definition 527.36. The shift row Q and substitution row R are consumed only through the ledger row L and audit row A.

The carrier transports, continuation routes, package provenance, and local naming row are then fixed by Definition 527.38 and Definition 527.113. Together they repack every root shift/substitution read into the displayed Q, R, L, A, H, C, P, N route and leave no remaining root operation request outside it. □

Theorem 527.59 (Closed-term substitution boundary root consumer NameCert separation)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the root* NameCert *route remains separate from subject reduction, host substitution semantics, host syntax quotients, hidden evaluator state, and hidden compiler state. A consumer may receive the finite route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

but receives no additional host-level substitution boundary from that certificate.

Proof. Read the packet through the carrier, classifier, and NameCert obligations. They enumerate exactly the finite BHist, hsame, Cont, Pkg, and NameCert rows available to a root consumer.

Apply Definition 527.40. The ledger route does not export host substitution semantics, host equality, hidden compiler state, or subject reduction.

Next combine Definition 527.58 with Definition 527.113. The operation surface and the root handoff are the same finite certificate route.

It remains to exclude additional host syntax or evaluator data. That exclusion is exactly the non-escape part of Definition 527.44, so the root NameCert route stays separate from all host-level substitution and reduction surfaces. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootConsumerNamecer`

Theorem 527.60 (Closed-term substitution boundary source triad)**.** *For every accepted* ClosedTermSubstitutionBo *packet*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N),$$

the displayed rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ *form the complete source, value, and depth admission triad before any shift row* Q *or substitution row* R *is consumed. The triad is read through the carrier, classifier, local* NameCert *obligations, and closedness ledger, and it exposes no operation row before the source and value closedness witnesses are fixed.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySourceTriad`

Proof. Read the carrier of Definition 527.1. It displays the source term row T, value row V, inspected depth row d, source closedness row C_{src}, value closedness row C_{val}, operation rows, ledger, audit row, transports, routes, provenance, and local name as one finite packet.

Use the classifier and local NameCert obligations of Definition 527.2 and Definition 527.3. They make the source/value closedness rows the admission surface for closed shift and closed substitution.

Finally apply source/value closedness admission through Definition 527.27. The ledger records the closedness reads before Q or R is consumed, so the source, value, and depth rows form the complete admission triad for the operation boundary. □

Theorem 527.61 (Closed-term substitution boundary operation route)**.** *Every accepted operation read of a* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet factors through*

$$Q, R, L, H, C, P, N$$

after the closedness witnesses C_{src} and C_{val} have been displayed. The route is the finite operation surface of the packet, not host substitution semantics, unchecked compiler state, subject reduction, or a hidden evaluator.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryOperationRoute`

Proof. Start from the source triad of Definition 527.60. It supplies the displayed term rows, depth row, and closedness witnesses that must be fixed before any operation read.

Apply the substitution-row exactness and shift-substitution ledger lock theorems, Definition 527.7 and Definition 527.36. These results force reads of Q and R through the closedness ledger L and the audit-visible packet boundary.

The remaining operation route is componentwise transport, continuation consumption, package provenance, and local naming. Thus H, C, P, N are the only admissible transport and consumer rows after Q, R, L, and no host substitution semantics, unchecked compiler state, subject reduction, or hidden evaluator is exported. □

Theorem 527.62 (Closed-term substitution boundary audit non-escape)**.** *The audit row A of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet excludes free-variable leaks, host equality, unchecked compiler state, subject reduction, and host substitution semantics from the source/value/depth admission surface and from the operation route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, H, C, P, N.$$

Every audit-visible consumer must replay the displayed finite packet rather than importing an external substitution theorem or compiler fact.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditNonescape`

Proof. Project the audit row from the carrier and route it through the classifier. The rows A, H, C, P, N belong to the same finite packet as the source triad and operation rows, so audit visibility cannot be detached from the displayed boundary.

Apply the local NameCert obligations and ledger non-escape theorem, Definition 527.3 and Definition 527.8. They rule out unregistered free variables, hidden host equality, unchecked compiler state, ambient subject reduction, and host substitution semantics.

Finally combine the source triad with the operation route from Definition 527.60 and Definition 527.61. Every operation read already passes through the closedness witnesses, ledger, transports, continuation route, provenance, and local name, so the audit row covers the complete finite route and no external substitution or compiler fact is available. □

Corollary 527.63 (Closed-term substitution boundary source triad strict obstruction)**.** *Let*

$$S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N)$$

be an accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, and let O be a displayed failure, refusal, separation, or countermodel row attached in the root-unblock source surface to a named consumer-facing converse or reuse attempt. If the named attempt asks to read the closed-term substitution boundary through the source, value, and depth admission triad, then O is a strict local obstruction: it blocks only that named attempt at one of the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

or at the corresponding refusal to supply an unlisted coordinate. It does not imply a global impossibility statement, a host theorem, hidden external validation, host substitution semantics, subject reduction, unchecked compiler state, host equality, or a free-variable leak outside the accepted packet.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySourceTriadStrictOb`

Proof. By Definition 527.60, the rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ form the complete admission triad before any shift row Q or substitution row R is consumed. Thus a consumer-facing converse or reuse attempt has no access to an operation row until the source term, value term, inspected depth, and both closedness witnesses have already been fixed.

By Definition 527.62, every audit-visible consumer must replay the finite packet route

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, H, C, P, N,$$

with the audit row A excluding free-variable leaks, host equality, unchecked compiler state, subject reduction, and host substitution semantics from both the admission surface and the operation route. Hence any consumer-facing use of O either factors through one of the displayed packet rows or asks for a coordinate that the packet does not export.

If the named attempt factors through a displayed row, then the displayed row O can obstruct only that requested converse or reuse read: the packet still exposes the remaining listed rows only through the same finite route. If the named attempt asks for an unlisted coordinate, then the obstruction is the local refusal of that coordinate, since the source triad and audit non-escape surface provide no projection to a host theorem, hidden external validation, global impossibility claim, unchecked compiler state, subject reduction theorem, host substitution semantics, host equality, or free-variable leak. Therefore O has exactly strict local force against the named converse or reuse route and cannot be reinterpreted as any stronger external or global claim. □

Theorem 527.64 (Closed-term substitution boundary recognizer landing surface)**.** *Any recognizer-side read of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must first pass through* GroundCompilerRecognizerF *acceptance, the generated recognizer row, the* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *tuple image, and the* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *readback row. Only after that route has recovered the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

may a closed-substitution consumer read the packet. The route supplies a recognizer-to-finite-tuple landing surface and no direct closedness readback outside the accepted finite packet.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecognizerLandingSu`

Proof. Start with the carrier and classifier of Definition 527.1 and Definition 527.2. They make $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ the complete public packet and give the recognizer read no coordinate outside those rows.

Next apply the root source and operation rows of Definition 527.60, Definition 527.57, and Definition 527.58. These rows force source, value, depth, closedness, shift, substitution, ledger, audit, transport, route, provenance, and local naming data to appear before any closed-substitution consumer can proceed.

The recognizer side is supplied by Definition 1048.3. Its event-flow, source-channel, generated-recognizer, and certificate-gate rows ensure that acceptance is mediated by the generated recognizer rather than by source text or an unmediated compiler export.

Finally use Definition 993.2 and Definition 993.3. They force the accepted recognizer export to land in a displayed tuple image and to recover each field through the readback row. Therefore the closed-substitution consumer can read the boundary only after recognizer acceptance, tuple-image landing, and componentwise readback of the displayed packet. □

Theorem 527.65 (Closed-term substitution boundary finite tuple readback)**.** *For any accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet routed through the recognizer landing surface of Definition 527.64, the finite tuple readback must recover the source term T, value term V, depth row d, closedness rows C_{src} and C_{val}, shift row Q, substitution row R, ledger L, and audit row A as* BHist *components before R can be consumed by a downstream closed-substitution read.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryFiniteTupleReadback`

Proof. Project the accepted boundary packet through the carrier. The source term, value term, depth, closedness witnesses, shift row, substitution row, ledger, audit row, transports, continuation routes, provenance, and local naming row are all displayed BHist coordinates of the same packet.

Use Definition 527.57. It prevents the substitution row from being consumed before $T, V, d, C_{\mathsf{src}}$, and C_{val} have been exposed.

Next apply Definition 527.58. Once the closedness budget is visible, the operation read is still forced through Q, R, L, A, H, C, P, N, so R is not available as a standalone row.

The tuple route is then componentwise by Definition 993.2 and Definition 993.3. Since the recognizer landing theorem already places the accepted packet in the tuple image/readback surface, the readback must recover $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A$ as BHist components before any consumer can use R. □

Theorem 527.66 (Closed-term substitution boundary substitution recognizer non-escape)**.** *The combined* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *and* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *route for an accepted* ClosedTermSubstitutionB*packet exports no open-term coordinate, free-variable leak, host syntax quotient, unchecked evaluator, compiler correctness theorem, or ambient subject reduction. Its public substitution read is exactly the recognizer-accepted tuple readback of*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionRecogni`

Proof. Begin with the downstream non-escape boundary of Definition 527.62. It excludes free-variable leaks, host equality, unchecked compiler state, subject reduction, and host substitution semantics from the accepted boundary packet.

Add the root consumer separation of Definition 527.59. That theorem keeps the root NameCert route separate from subject reduction, host substitution semantics, host syntax quotients, hidden evaluator state, and hidden compiler state.

Now compose the recognizer and tuple routes. By Definition 1048.4, report-like source, host compiler state, proof text, and raw classifier quotients remain non-evidence inside $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$. By Definition 993.3, the reflected packet can be read only through the displayed tuple image and readback.

Finally invoke Definition 527.64. The recognizer side reaches the boundary only after acceptance, generated recognition, tuple-image landing, and readback of the displayed rows. Therefore no open-term coordinate, free-variable leak, host syntax quotient, unchecked evaluator, compiler correctness theorem, or ambient subject reduction escapes the combined route. □

Corollary 527.67 (Closed-term substitution boundary recognizer strict local obstruction)**.** *Let O be a displayed failure, refusal, separation, or countermodel row attached to a consumer-facing converse or reuse attempt for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet. If the attempt is routed through the recognizer side, then O is a strict local obstruction: it can block only the named read that asks for one of the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

through $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *acceptance, the generated recognizer row, the* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *tuple image, and the* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *readback row. The same row cannot be reinterpreted as a global impossibility statement, a host theorem, compiler correctness evidence, ambient subject reduction, or hidden external validation for any unlisted route.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRecognizerStrictLoc`

Proof. Apply Definition 527.64. Every recognizer-side consumer of the accepted packet first passes through $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ acceptance, the generated recognizer row, tuple-image landing, and tuple readback. Only after that route has recovered

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

may the closed-substitution consumer read the packet. Thus a consumer-facing converse or reuse attempt has no initial access point except the displayed finite route.

Now apply Definition 527.66. The combined recognizer and finite-tuple route exports no open-term coordinate, free-variable leak, host syntax quotient, unchecked evaluator, compiler correctness theorem, or ambient subject reduction. Its public substitution read is exactly the recognizer-accepted tuple readback of the displayed rows.

Therefore the obstruction row O has only two possible relations to the attempted consumer read. If the attempt asks for one of the displayed coordinates through the accepted recognizer and tuple route, then O blocks that named coordinate read and hence blocks that particular converse or reuse route. If the attempt asks for a global impossibility conclusion, host theorem, compiler correctness result, ambient subject-reduction theorem, hidden evaluator, external validation channel, or any other unlisted coordinate, the landing theorem supplies no readback coordinate for the request and the non-escape theorem explicitly excludes it from the exported packet. In neither case can O be promoted beyond the named local obstruction. □

Theorem 527.68 (Closed-term substitution boundary root depth-value lock)**.** *Every root operation consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must read the source term*

row T, value row V, depth row d, source closedness row C_{src}, and value closedness row C_{val} before either the closed shift row Q or the substitute-closed row R is exported. The lock is carried by the same finite route

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and uses the carrier, classifier, local NameCert *obligations, root audit totality, and root* NameCert *handoff of the parent packet.*

Proof. Project the accepted packet through the carrier and classifier. The displayed rows list the source term, value, inspected depth, closedness witnesses, operation rows, ledger, audit row, transports, continuations, provenance, and local naming row as one finite surface.

Apply the local NameCert obligations. They place C_{src} and C_{val} in the packet before the operation rows may be consumed as accepted closed-boundary data.

Finally use root audit totality and the root NameCert handoff. Both root-facing rows factor through the same displayed carrier, so the depth-value surface is locked before Q or R can be exported. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootDepthValueLock`

Theorem 527.69 (Closed-term substitution boundary root operation readback lock)**.** *Every root read of the closed shift row Q or substitute-closed row R of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet factors through*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The operation readback is admitted only after the depth-value lock, the local classifier, the NameCert *obligation surface, the audit row A, and the root* NameCert *handoff agree on the same finite packet. It exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, or hidden compiler state.*

Proof. Begin with Definition 527.68. It fixes $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ before either operation row is available.

Read the operation rows through the existing ledger and audit surface. The rows Q and R are not standalone host operations; they are displayed packet coordinates routed through L and checked by A.

Now apply the root audit totality and root NameCert handoff. They force every root consumer back through the same H, C, P, N structural rows, so no operation readback can leave the finite carrier for a host substitution or compiler surface. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootOperationReadba`

Theorem 527.70 (Closed-term substitution boundary root consumer audit factorization)**.** *Every root consumer that uses the operation readback surface of* ClosedTermSubstitutionBoundary$^{\uparrow}$ *factors as a finite audit route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The factorization first exposes the term, value, depth, and closedness rows; then reads the closed shift and substitute-closed rows through Q, R, L, A; and finally packages the same rows through componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Use the root depth-value lock for the first stage of the factorization. It fixes the source, value, inspected depth, and closedness budget before the root operation rows are visible.

Use the root operation readback lock for the second stage. It routes Q and R through the ledger and audit rows and prevents a consumer from replacing them by host substitution semantics or an ambient subject-reduction theorem.

The final stage is the parent root NameCert handoff. The structural rows H, C, P, N carry exactly the same finite packet to downstream consumers, so the root audit factorization has no extra host syntax, evaluator, compiler-state, or quotient coordinate. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootConsumerAuditFa`

Lemma 527.71 (Closed-term substitution boundary root depth-value bridge determinacy)**.** *For an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet, every consumer-facing route, transport, readback, or handoff that uses the root depth-value operation boundary is determined by the finite displayed surface*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The determining rows are the source term row T, value row V, inspected depth row d, source and value closedness rows C_{src} and C_{val}, closed shift row Q, substitute-closed row R, ledger row L, audit row A, componentwise transport row H, displayed continuation row C, package provenance row P, and local NameCert *row N. A consumer may use Q or R only through this surface, and any route, transport, readback, or handoff coordinate not listed in the surface is unavailable to the root depth-value bridge.*

Proof. Start with Definition 527.68. It fixes $T, V, d, C_{\mathsf{src}}$, and C_{val} before either operation row can be exported. Thus a consumer-facing bridge cannot start from Q or R alone; it must first pass through the displayed source, value, depth, and closedness rows of the accepted packet.

Apply Definition 527.69. Every root read of Q or R factors through the same finite route

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The readback condition ties the operation rows to the local classifier, the NameCert obligation surface, the ledger, the audit row, and the root NameCert handoff. Hence any displayed operation readback used by a consumer is already a readback of these named rows, not a separate host substitution operation.

It remains to account for transports, continuations, provenance, and handoff. The same readback lock routes them through H, C, P, N, while the depth-value lock prevents the operation rows from appearing before $T, V, d, C_{\mathsf{src}}$, and C_{val} have been exposed. Therefore a consumer-facing transport is componentwise transport over the displayed packet, a continuation is replay over the displayed packet, provenance is the displayed package row, and handoff is the local NameCert row attached to that packet.

No further bridge coordinate is present in either lock. The operation readback lock explicitly exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, or hidden compiler state, and the depth-value lock supplies only the listed carrier, classifier, local NameCert obligations, root audit totality, and root NameCert handoff. Consequently every admissible route, transport, readback, or handoff factors through the cited rows, while every unlisted bridge coordinate remains unavailable. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootDepthValueBridg`

Corollary 527.72 (Closed-term substitution boundary root depth-value consumer determinacy)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet whose displayed root depth-value rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

are fixed, every accepted consumer-facing use of the root depth-value operation boundary has the same public decision source. Its finite route is the displayed route above; its transport, readback, normal-form handoff, or exactness decision named by the chapter is read through Q, R, L, A, H, C, P, N *after* $T, V, d, C_{\mathsf{src}}$*, and* C_{val} *have been exposed. No accepted consumer may choose an alternate finite route, hidden normal-form coordinate, duplicate classifier source, host substitution semantics, subject-reduction theorem, host syntax quotient, hidden evaluator, or hidden compiler state.*

Proof. Begin with Definition 527.68. It fixes the source term row T, value row V, inspected depth row d, source closedness row C_{src}, and value closedness row C_{val} before either operation row Q or R can be exported. Thus an accepted consumer cannot start its root operation use from a detached operation row, a second classifier source, or a hidden depth-value coordinate.

Apply Definition 527.71. Every consumer-facing route, transport, readback, or handoff using the root depth-value operation boundary is determined by the single displayed surface

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The same lemma identifies Q and R as usable only through that surface and states that any route, transport, readback, or handoff coordinate not listed there is unavailable to the bridge.

It follows that, once the displayed rows are fixed, there is no remaining choice point for a consumer-facing decision. A finite route must be the displayed route; a transport must be the componentwise row H over that packet; a readback or normal-form handoff must pass through Q, R, L, A and then through C, P, N; and an exactness decision named by the chapter must cite the same closedness, ledger, audit, provenance, and local naming rows. Any alternate finite route, hidden normal-form coordinate, duplicate classifier source, host substitution semantics, subject-reduction theorem, quotient syntax, hidden evaluator, or compiler state would be an unlisted coordinate. The bridge determinacy lemma makes such a coordinate unavailable, while the depth-value lock prevents it from being inserted before the operation rows are exported. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootDepthValueConsu`

Theorem 527.73 (Closed-term substitution boundary root value visibility)**.** *Every root consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must expose*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$$

before either the closed-shift row Q *or the substitute-closed row* R *is used. The exposed rows are the displayed source term, substituting value, inspected depth, source closedness witness, and value closedness witness of the same finite* BHist *packet.*

Proof. Read the root consumer through the carrier and local certificate rows of Definition 527.1 and Definition 527.3. They make $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ packet coordinates rather than hidden preconditions.

Use source and value admission from Definition 527.4 and Definition 527.5. These rows place the closedness witnesses on the visible route before the shift or substitution boundary can be read.

Finally apply Definition 527.8 and Definition 527.12. The ledger and consumer boundary prevent a root consumer from obtaining Q or R from host substitution semantics, a hidden evaluator, or an ambient compiler theorem. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootValueVisibility

Theorem 527.74 (Closed-term substitution boundary root operation dependency)**.** *In every root read of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the rows* Q *and* R *are dependent operation rows over the displayed witnesses* C_{src} *and* C_{val}*. They may be consumed only as the packet's certified closed-shift and substitute-closed boundary rows, not as host shift semantics, host substitution semantics, subject reduction, or an evaluator equation.*

Proof. Start from the operation inventory of Definition 527.28. It identifies Q and R as the only operation rows of the finite packet.

Use Definition 527.27 and Definition 527.36. These rows force the operation read through the displayed closedness witnesses and ledger before any downstream citation can mention Q or R.

The exclusion of host semantics follows from Definition 527.40 and Definition 527.44. Hence the operation dependency is a packet dependency over the displayed closedness witnesses, not an imported shift or substitution semantics. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootOperationDepend

Theorem 527.75 (Closed-term substitution boundary root consumer exhaustion)**.** *Downstream self-compilation and normalization consumers exhaust their closed-substitution read through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$*. No remaining consumer route may add an unregistered free-variable coordinate, host equality, hidden evaluator, hidden compiler state, subject-reduction theorem, or host substitution semantics.*

Proof. First package the root-visible rows by Definition 527.73 and Definition 527.74. Together they expose the source, value, depth, closedness, and operation surface before any root read continues.

Next use the existing consumer rows. The self-compilation route of Definition 527.29 and the normalization-frontier handoff of Definition 527.20 both factor through the same finite boundary packet.

Finally apply Definition 527.38 and Definition 527.44. The displayed BHist, hsame, Cont, Pkg, and NameCert rows therefore exhaust the downstream read, with no route to hidden substitution or compiler data. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootConsumerExhaust

Theorem 527.76 (Closed-term substitution boundary root value downstream coverage)**.** *Every accepted local consumer of the root value row* V *in a* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is exhausted by the displayed finite coverage alternatives*

$$(T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}), \qquad (Q, R), \qquad (\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}).$$

The first alternative covers the visible value and closedness read, the second covers the certified closed-shift and substitute-closed operation rows, and the third covers downstream self-compilation or normalization consumption through the packet ledger. No excluded route through an unregistered free-variable coordinate, host equality, hidden evaluator, hidden compiler state, subject-reduction theorem, or host substitution semantics belongs to the accepted BEDC packet.

Proof. Let an accepted local consumer ask for the root value row V. By Definition 527.73, the consumer must first expose

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$$

inside the same finite BHist packet before either operation row can be used. Any consumer whose read stops at the value, inspected depth, or closedness witnesses is therefore covered by the first displayed alternative.

If the consumer continues to an operation read, Definition 527.74 identifies Q and R as dependent operation rows over the displayed witnesses C_{src} and C_{val}. Thus the operation continuation is covered by the certified closed-shift and substitute-closed rows, not by a host shift rule, host substitution rule, evaluator equation, or subject-reduction theorem.

It remains to cover consumers that pass the operation rows downstream. This is exactly the exhaustion statement of Definition 527.75: downstream self-compilation and normalization consumers factor through the displayed BHist, hsame, Cont, Pkg, and NameCert packet, with no route to an unregistered free-variable coordinate, host equality, hidden evaluator, hidden compiler state, subject reduction, or host substitution semantics. The three displayed alternatives therefore cover every accepted local root-value consumer, and all excluded alternatives remain outside the accepted BEDC packet. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootValueDownstrean`

Theorem 527.77 (Closed-term substitution boundary root closed boundary cut)**.** *Every downstream self-compilation consumer of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet must enter the root closed-boundary cut through the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The rows C_{src} and C_{val} gate every read of the closed shift row Q and substitute-closed row R, and the cut exports only the finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route of the packet. It exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or unregistered operation coordinate.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootClosedBoundaryC`

Proof. Read the packet through Definition 527.1, Definition 527.2, and Definition 527.3. These rows make the source term, value, depth, closedness witnesses, operation rows, ledger, audit, transport, continuation, provenance, and local naming data visible as one finite BEDC packet.

Next apply Definition 527.73 and Definition 527.69. They force C_{src} and C_{val} to be visible before Q or R is consumed by any root read.

Finally use Definition 527.70. The downstream consumer is exhausted by the displayed audit route, so the cut has no remaining access to host substitution semantics, subject reduction, quotient syntax, hidden evaluation, or compiler state. □

Theorem 527.78 (Closed-term substitution boundary closed value route determinacy)**.** *In the displayed root packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

the closed-value route is determined by the value row V, the depth row d, and the witness row C_{val} before any read of Q or R. A downstream self-compilation consumer may therefore cite the substitute-closed boundary only through the same finite packet route, not through a host value semantics, hidden evaluator, subject reduction theorem, or hidden compiler state.

Proof. Start with Definition 527.5. It places C_{val} on the displayed closedness route for the same value and depth rows that the accepted packet exposes.

Apply Definition 527.73. The theorem makes V, d, C_{val} visible before either operation row is available to a root consumer.

The determinacy of the route follows by Definition 527.69. Since Q and R are read only after the displayed closedness rows and through L, A, H, C, P, N, no host value semantics or hidden compiler object can supply an alternate route. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosedValueRouteDet

Theorem 527.79 (Closed-term substitution boundary shift-substitution source synchronization)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *root packet, the closed shift row Q and substitute-closed row R are synchronized to the same displayed source rows T, d, C_{src} in*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Thus a downstream consumer cannot read the shift boundary from one source witness and the substitution boundary from another; both reads are gated by the same C_{src} and then carried through the common ledger, audit, transport, continuation, provenance, and NameCert *rows.*

Proof. Use Definition 527.4 to identify C_{src} as the displayed source closedness witness for the accepted packet. The source term row T and depth row d are therefore part of the same finite carrier surface as the closedness witness.

Next apply Definition 527.36. It locks the closed shift and substitute-closed rows behind the same ledger route, so Q and R cannot be split across independent source witnesses.

Finally read the root operation through Definition 527.69. The common route through L, A, H, C, P, N synchronizes both operation rows to the same source witness and blocks replacement by host shift or substitution semantics. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryShiftSubstitutionSo

Theorem 527.80 (Closed-term substitution boundary consumer ledger exhaustion)**.** *Every downstream self-compilation consumer of the displayed root packet exhausts its operation access through the ledger row L and audit row A of*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

After the ledger has admitted the closed-source and closed-value witnesses and the operation rows, the remaining consumer surface is exactly H, C, P, N; there is no additional host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or implicit read channel.

Proof. Begin with Definition 527.8. It states that the ledger does not let a consumer escape the displayed finite packet when reading closed-substitution data.

Combine this with Definition 527.12 and Definition 527.70. The consumer boundary and audit factorization route all downstream access through L and A before the structural rows H, C, P, N are packaged.

The ledger is therefore exhaustive for the consumer. Any attempted host semantics, hidden evaluator, subject-reduction theorem, quotient syntax, compiler state, or implicit read channel would be an additional row outside the displayed packet, contradicting the ledger and audit boundary. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerLedgerExhau

Theorem 527.81 (Closed-term substitution boundary NameCert root unblock package)**.** *The root closed-boundary cut, closed-value route determinacy, shift-substitution source synchronization, and consumer ledger exhaustion form a NameCert root-unblock package for downstream self-compilation consumers of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$*. The package is precisely the finite BEDC route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

together with the carrier, classifier, BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows already displayed in the chapter. It exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, hidden compiler pass, or extra root operation beyond that finite route.*

Lean4 ✓ `BEDC.Derived.Closedtermsubstitutionboundaryup.ClosedTermSubstitutionBoundaryNameCertRootUnbloc`

Proof. First package the closed-boundary cut by Definition 527.77. It identifies the finite route and states that the closed-source and closed-value witnesses gate the operation rows.

Next add value determinacy and source synchronization by Definition 527.78 and Definition 527.79. These two rows make the value and source witnesses part of the same closed operation surface.

Finally apply Definition 527.80 and Definition 527.113. The downstream self-compilation consumer receives the finite NameCert package, and every excluded host semantic, evaluator, quotient, compiler-state, compiler-pass, or extra operation route lies outside the exported BEDC boundary. □

Corollary 527.82 (Closed-term substitution boundary root closed determinacy)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *root packet whose displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

are fixed, every accepted downstream self-compilation use of the root closed boundary has the same public decision source. Its closed-value route is the route over V, d, C_{val}*, its shift and substitute-closed source is the synchronized route over* T, d, C_{src}*, and its consumer-facing route, normal-form handoff, or exactness decision is read only through* Q, R, L, A, H, C, P, N*. Hence no accepted consumer may replace the displayed packet by an alternate finite route, a hidden normal-form coordinate, a duplicate classifier source, a host substitution semantics, a subject-reduction theorem, a host syntax quotient, a hidden evaluator, or hidden compiler state.*

Lean4 ✓ `BEDC.Derived.Closedtermsubstitutionboundaryup.ClosedTermSubstitutionBoundaryRootClosedDetermina`

Proof. Apply Definition 527.78. It fixes the closed-value side before any operation read: once V, d, and C_{val} are the displayed rows of the accepted packet, a consumer can cite the substitute-closed boundary only through that finite packet route and not through a host value semantics or hidden evaluator.

Apply Definition 527.79. It fixes the source side of the same root read: both Q and R are synchronized to T, d, C_{src}, so the shift boundary and the substitute-closed boundary cannot be split across two classifier sources or two closedness witnesses.

The remaining consumer surface is then forced by Definition 527.69 and Definition 527.80. Every root read of Q or R passes through the common ledger and audit rows and is then packaged by H, C, P, N. Therefore any accepted consumer-facing route, normal-form handoff, or exactness decision named by the chapter is a read of the same displayed packet. An alternate route, hidden

normal-form coordinate, duplicate classifier source, host substitution semantics, subject-reduction theorem, quotient syntax, hidden evaluator, or compiler state would require a row outside the displayed packet, which the cited boundary rows do not export. □

Theorem 527.83 (Closed-term substitution boundary root route lock)**.** *Every downstream self-compilation route that consumes an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must pass through the displayed finite packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The route may cite the carrier, classifier, local NameCert *obligations, ledger non-escape row, consumer boundary, audit totality row, compiler non-escape row, and root* NameCert *handoff, but it exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or additional operation coordinate.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootRouteLock`

Proof. Start with Definition 527.1 and Definition 527.2. They display the term rows, depth row, closedness witnesses, closed shift and substitute-closed rows, ledger, audit, transport, continuation, provenance, and local naming data as the only coordinates of the accepted packet.

Apply Definition 527.3, Definition 527.8, and Definition 527.12. These rows force every operation read through the same closedness witnesses and ledger boundary before any downstream consumer can see Q or R.

Finally use Definition 527.45, Definition 527.112, and Definition 527.113. The audit, compiler-facing, and naming handoff surfaces all factor through the displayed packet, so the root route has no coordinate outside $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. □

Theorem 527.84 (Closed-term substitution boundary root operation export lock)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the closed shift row Q and substitute-closed row R are exportable to a root consumer only as rows of*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The export is locked by the carrier, local NameCert *obligations, ledger non-escape, consumer boundary, root audit totality, root compiler non-escape, and root* NameCert *handoff; it does not expose a host substitution relation, subject reduction proof, evaluator, compiler state, or unregistered source/value convention.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootOperationExport`

Proof. Read Q and R from the carrier. The carrier places both operation rows beside $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$, so neither row is available apart from the displayed closedness and depth data.

The local NameCert obligations and ledger non-escape theorem then supply the export order. A consumer must see the source and value closedness witnesses, pass through L, and remain inside the audit-visible packet before the operation rows can be used.

Now apply the consumer boundary together with the root audit totality and root compiler non-escape rows. They make any root operation export a packet read through A, H, C, P, N and prevent replacement by host substitution semantics, subject reduction, evaluator state, compiler state, or an unregistered convention.

The root NameCert handoff repacks the same route for downstream self-compilation consumers. Hence the operation export is locked to the finite packet and nowhere else. □

Theorem 527.85 (Closed-term substitution boundary root self-compile handoff readiness)**.** *The root route lock of Definition 527.83 and the root operation export lock of Definition 527.84 make* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *ready for downstream self-compilation handoff exactly through*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The ready handoff is the finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet witnessed by the carrier, local obligations, ledger non-escape, consumer boundary, audit totality, compiler non-escape, and root naming rows, not a host-level substitution or compiler-correctness object.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootSelfCompileHand`

Proof. First collect the route side. By Definition 527.83, every downstream self-compilation route is already confined to the displayed carrier, classifier, ledger, audit, transport, continuation, provenance, and naming rows.

Next collect the operation side. By Definition 527.84, the closed shift and substitute-closed rows reach a root consumer only after the closedness, ledger, audit, compiler-facing, and NameCert handoff rows have been exposed.

Finally combine these two rows with Definition 527.113. The downstream consumer receives one finite handoff packet over $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$, and no host substitution semantics, subject reduction theorem, hidden evaluator, hidden compiler state, or compiler-correctness object is added. □

Lemma 527.86 (Closed-term substitution boundary root route bridge determinacy)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, every consumer-facing bridge use of the root route l is determined by the displayed finite packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

In particular, a route use, operation transport, closed-shift readback, substitute-closed readback, audit handoff, compiler-facing handoff, or root NameCert *handoff factors through the carrier, classifier, local* NameCert *obligations, ledger non-escape row, consumer boundary, root audit totality row, root compiler non-escape row, and root* NameCert *handoff cited by Definition 527.83, with Q and R exported only as in Definition 527.84. A bridge request for host substitution semantics, a subject-reduction theorem, a host syntax quotient, a hidden evaluator, hidden compiler state, or an operation coordinate outside the displayed packet is therefore unavailable to the consumer.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootRouteBridgeDete`

Proof. Start with the route inventory supplied by Definition 527.83. That theorem already lists the complete downstream self-compilation route: the accepted packet may expose the carrier, classifier, local naming obligations, ledger row, consumer boundary, audit row, compiler-facing row, and naming handoff, and all of them live on the single displayed tuple $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. Thus a bridge use of root route l has only the displayed finite coordinates before any consumer can read it.

Now separate the possible consumer-facing uses. A pure route use reads the carrier and classifier rows. A transport use reads the componentwise hsame row H and the displayed Cont row C. A readback of closed shift or substitute-closed data must pass through Q or R together with the closedness witnesses and ledger boundary. An audit or handoff use reads L, A, H, C, P, N through the root audit, compiler non-escape, and root NameCert handoff rows. These are exactly the rows

named by the route lock, so two bridge presentations that agree on this finite packet have no remaining coordinate on which their consumer-visible route could differ.

It remains to rule out a detached operation bridge. Apply Definition 527.84. It says that the operation rows Q and R reach a root consumer only as rows of the same displayed packet and only after the carrier, local NameCert obligations, ledger non-escape, consumer boundary, root audit totality, root compiler non-escape, and root naming handoff have locked the export. Hence a consumer cannot replace the finite bridge route by host substitution semantics, subject reduction, a host syntax quotient, a hidden evaluator, hidden compiler state, an unregistered source/value convention, or any extra operation coordinate.

Combining the route lock with the operation export lock gives determinacy. Every listed consumer-facing use factors through the same local rows, while every unlisted bridge coordinate has no projection from the accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet and therefore remains unavailable. □

Corollary 527.87 (Closed-term substitution boundary root route l determinacy)**.** *For an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet whose displayed local rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

are fixed, every accepted consumer-facing use of the root route l has the same public decision source. Its finite route is the displayed route above, and any operation transport, closed-shift readback, substitute-closed readback, audit handoff, compiler-facing handoff, root NameCert *handoff, normal-form handoff, or exactness decision named by the chapter is read only through the carrier, classifier, local* NameCert *obligations, ledger non-escape row, consumer boundary, root audit totality row, root compiler non-escape row, and root* NameCert *handoff. No accepted consumer may pick an alternate finite route, hidden normal-form coordinate, duplicate classifier source, host substitution semantics, subject-reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or additional operation coordinate.*

Proof. Start with Definition 527.83. It fixes the only downstream self-compilation route available to a consumer of the accepted packet: the route must pass through the displayed carrier, classifier, local naming obligations, ledger row, consumer boundary, audit row, compiler-facing row, and naming handoff, all on the single tuple

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The same theorem excludes host substitution semantics, subject reduction, a host syntax quotient, hidden evaluator state, hidden compiler state, and any additional operation coordinate from the exported root route.

Apply Definition 527.86. It lists the consumer-facing bridge uses of the root route l: route use, operation transport, closed-shift readback, substitute-closed readback, audit handoff, compiler-facing handoff, and root NameCert handoff. Each of those uses factors through the same carrier, classifier, local NameCert obligations, ledger non-escape row, consumer boundary, root audit totality row, root compiler non-escape row, and root NameCert handoff cited by the route lock. The lemma also states that a bridge request for host substitution semantics, subject reduction, a host syntax quotient, hidden evaluator state, hidden compiler state, or an operation coordinate outside the displayed packet is unavailable to the consumer.

The fixed-row determinacy follows from those two facts. Once the displayed rows are fixed, the route lock leaves no second packet from which a consumer could source a finite route, and the bridge

determinacy lemma leaves no unlisted coordinate through which a consumer-facing use could vary. Thus a finite route must be the displayed route; a transport must be the componentwise row over that packet; a closed-shift or substitute-closed readback must pass through Q or R inside the same ledger and audit surface; and any normal-form handoff or exactness decision named by the chapter must be a handoff or decision read through L, A, H, C, P, N and the local NameCert row. An alternate finite route, hidden normal-form coordinate, duplicate classifier source, or hidden host-side classifier would require a coordinate outside the accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet, and the cited rows do not export such a coordinate. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootRouteLDetermina`

Definition 527.88 (Closed-term substitution boundary root operation surface)**.** *The root operation surface of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is the finite displayed row list*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Here T is the source term row, V is the substituting closed value row, d is the inspected depth row, C_{src} and C_{val} are the displayed closedness witnesses, Q is the closed shift boundary row, R is the substitute-closed boundary row, L is the ledger row, A is the audit row, H is the componentwise hsame *transport row, C is the displayed* Cont *route row, P is the* Pkg *provenance row, and N is the local* NameCert *row.*

Lean4 (definition) `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootOperati`

Theorem 527.89 (Closed-term substitution boundary root operation totality)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the closed shift row Q and the substitute-closed row R are both exhausted by the root operation surface of Definition 527.88. Neither operation can be read from a host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or any row outside*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootOperationTotali`

Proof. Project the accepted packet through Definition 527.1 and Definition 527.2. These two rows expose the source, value, depth, closedness witnesses, operation rows, ledger, audit row, transport, route, provenance, and local naming data as one finite packet.

Apply Definition 527.8 and Definition 527.45. The ledger blocks escape from the displayed packet, while the root audit row makes every audit-visible operation read pass through $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A,$

Finally use Definition 527.112. Since the compiler-facing read is admitted only after the same closed-boundary packet is visible, both Q and R are total over the displayed root operation surface and nowhere else. □

Theorem 527.90 (Closed-term substitution boundary root audit compiler coherence)**.** *The audit route of Definition 527.45 and the compiler-facing route of Definition 527.112 cite the same closed-boundary packet. Their common cited object is the surface of Definition 527.88, not two separate substitution semantics and not a host-level compiler correctness witness.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootAuditCompilerCo`

Proof. Start with the root audit route. By Definition 527.45, every audit-visible consumer is already routed through the displayed closedness, operation, ledger, audit, transport, continuation, provenance, and naming rows.

Now read the compiler route through Definition 527.112. That theorem permits a citation of $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ only after the same closed shift and substitute-closed boundary rows have been exposed by the boundary packet.

The two routes therefore meet at the packet of Definition 527.88. The audit side and compiler side share one closed-boundary citation, so no second host substitution semantics, hidden evaluator, or ambient compiler theorem is present. □

Theorem 527.91 (Closed-term substitution boundary root compiler route exhaustion)**.** *Every downstream compiler read that cites chapter 540 factors through the root operation surface of Definition 527.88 before the compiler-facing citation is available. The factorization exports exactly the accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet rows and no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or additional operation coordinate.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootCompilerRouteEx`

Proof. First apply Definition 527.89. It exhausts closed shift and closed substitution over the single displayed packet, so a compiler consumer cannot obtain either operation from a separate host surface.

Next apply Definition 527.90. The audit route and compiler-facing route cite the same closed-boundary packet, which keeps the operation rows tied to $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$.

Finally use Definition 527.112. The downstream read may mention the sibling compiler chapter only after the packet has exposed the closed-boundary rows, so the compiler route is exhausted by the packet before the compiler citation is formed. □

Theorem 527.92 (Closed-term substitution boundary root downstream coverage)**.** *For every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, every consumer-facing use of the root operation surface is covered by the finite downstream alternatives*

$$\mathsf{operation}(Q, R), \qquad \mathsf{ledger}(L, A), \qquad \mathsf{transport}(H, C), \qquad \mathsf{provenance}(P, N),$$
$$\mathsf{compiler_citation}(\textit{chapter } 540) \text{ after } T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Equivalently, any accepted local consumer of closed shift, substitute-closed readback, audit replay, or compiler-facing citation factors through the displayed surface of Definition 527.88. Host substitution semantics, subject reduction, host syntax quotients, hidden evaluators, hidden compiler state, detached compiler correctness, and any operation coordinate outside

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

remain outside the $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootDownstreamCover`

Proof. First consider a direct operation consumer. By Definition 527.89, both root operation rows Q and R are exhausted by the finite surface of Definition 527.88. Thus a consumer cannot obtain closed shift or substitute-closed readback from a host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or unlisted row.

Next consider a ledger, audit, transport, provenance, or naming consumer. The totality theorem routes every audit-visible operation read through the same rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$, and Definition 527.45 makes the audit side total over that finite route. The rows L and A record the ledger and audit boundary, while H, C, P, N are only componentwise transport, displayed continuation, provenance, and local naming rows; none of them contains a coordinate for an external substitution source.

Finally consider a downstream compiler-facing consumer. By Definition 527.91, every read that cites chapter 540 first factors through the same root operation surface. By Definition 527.90, this compiler route and the audit route cite one closed-boundary packet rather than two separate semantics. The non-escaping compiler condition of Definition 527.112 then keeps the sibling compiler citation downstream of the displayed packet, so the excluded alternatives remain outside the BEDC packet. □

Theorem 527.93 (Closed-term substitution boundary root transport accountability)**.** *Every root consumer that makes closed substitution transport accountable for* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *must first expose the displayed rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Here $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$ are the term, value, depth, and closedness inputs; Q and R are the closed shift and substitute-closed operation rows; L and A are the ledger and audit rows; and H, C, P, N are the transport, continuation, provenance, and local naming rows. No accountable closed substitution transport is available from a host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or operation row outside this packet.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootTransportAccoun`

Proof. Read the root consumer through Definition 527.45. That row makes every audit-visible root read total over the displayed packet and prevents the consumer from starting at a detached host operation.

Apply Definition 527.8 and Definition 527.38. The ledger fixes the closed shift and substitute-closed boundary rows inside the packet, while the NameCert export keeps the same carrier, classifier, transport, continuation, provenance, and naming rows visible to downstream consumers.

Finally use Definition 527.112. A compiler-facing citation is admitted only after the same rows have been exposed, so accountable transport is exactly transport over $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ and not over a host substitution semantics. □

Theorem 527.94 (Closed-term substitution boundary root consumer coverage)**.** *Every self-compilation, proof-erasure, or compiler-facing consumer of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *is covered by the displayed closed-boundary packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Such a consumer may read the closed shift, substitute-closed, ledger, audit, transport, continuation, provenance, and naming rows only through that packet, and it receives no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or extra compiler object.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootConsumerCoverag`

Proof. Start with self-compilation and proof-erasure consumers. By Definition 527.45, any root audit consumer factors through the closed-boundary packet before it can inspect the operation rows.

For compiler-facing consumers, apply Definition 527.112. It allows the sibling compiler chapter to be cited only after the accepted packet has exposed the same term, closedness, operation, ledger, audit, transport, continuation, provenance, and naming rows.

The non-escape and export rows close the remaining alternatives. Use Definition 527.8 to rule out hidden operation sources, and use Definition 527.38 to keep the consumer attached to the displayed NameCert carrier. Thus each listed consumer is covered by the closed-boundary packet and by no host substitution semantics. □

Theorem 527.95 (Closed-term substitution boundary root-unblock certificate)**.** *The root transport accountability row of Definition 527.93, the root consumer coverage row of Definition 527.94, the root audit totality row of Definition 527.45, and the root compiler non-escape row of Definition 527.112 assemble a downstream root-unblock certificate for* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$*. The certificate is a finite handoff over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and exports no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or additional compiler-facing route.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootUnblockCertific`

Proof. First build the accountable transport surface. Apply Definition 527.93 to force every accountable closed substitution transport through the displayed term, closedness, operation, ledger, audit, transport, continuation, provenance, and naming rows.

Next cover the downstream consumers. By Definition 527.94, self-compilation, proof-erasure, and compiler-facing reads all use the same closed-boundary packet and no host substitution semantics.

Finally combine the root audit and compiler rows. The audit side is total by Definition 527.45, while the compiler side is non-escaping by Definition 527.112. Together with Definition 527.8 and Definition 527.38, these rows provide the downstream root-unblock certificate over the single displayed packet. □

Theorem 527.96 (Closed-term substitution boundary root transport downstream coverage)**.** *Every accepted local downstream consumer of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *whose root use is self-compilation, proof-erasure, or compiler-facing accountable transport is exhausted by the displayed packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The accepted alternatives are only the closed shift row Q, substitute-closed row R, ledger row L, audit row A, transport row H, continuation row C, provenance row P, and local naming row N, together with the term, value, depth, and closedness inputs $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}$. Host substitution semantics, a subject reduction theorem, a host syntax quotient, a hidden evaluator, hidden compiler state, an extra compiler object, and any operation row outside this packet remain unavailable to the downstream consumer.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootTransportDownst`

Proof. Apply Definition 527.94 to the accepted downstream consumer. It covers the consumer-facing cases named in the statement: self-compilation, proof-erasure, and compiler-facing reads. In each case the consumer may read closed shift, substitute-closed, ledger, audit, transport, continuation, provenance, and naming data only through the displayed closed-boundary packet.

Now apply Definition 527.93 to the same root use when it makes closed substitution transport accountable. Accountability forces the root consumer to expose exactly the rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C,$ before transport is available. Thus the transport coordinate does not create a second route; it is one of the displayed rows already covered by the consumer-coverage packet.

The exclusions are common to the two rows. Consumer coverage blocks host substitution semantics, subject reduction, a host syntax quotient, hidden evaluator state, hidden compiler state, and extra compiler objects from entering the accepted consumer read. Transport accountability blocks accountable transport from any host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or operation row outside this packet. Therefore every accepted local downstream use factors through the enumerated finite rows, and every unlisted alternative remains outside the $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet. □

Theorem 527.97 (Closed-term substitution boundary closed-source budget surface)**.** *Every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet exposes the finite rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The source closedness row C_{src} *and the value closedness row* C_{val} *are read before any closed shift budget* Q *or closed substitution budget* R *is consumed. Thus the operation budget is a closed-source and closed-value surface over the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a host substitution semantics or hidden evaluator route.*

Proof. Project the accepted carrier from Definition 527.1 and the classifier from Definition 527.2. The displayed packet already separates the term row T, value row V, depth row d, source closedness row C_{src}, value closedness row C_{val}, shift row Q, substitution row R, ledger row L, audit row A, transport row H, route row C, provenance row P, and local naming row N.

Apply Definition 527.3. Its carrier exposure and classifier obligations force every public read to begin from the same finite packet, so no consumer can acquire Q or R from an unlisted operation source.

Finally use Definition 527.36. The ledger lock places C_{src} and C_{val} before the shift and substitution rows in the public operation budget, and Definition 527.8 rules out a hidden source for closedness or operation state. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosedSourceBudgetS`

Theorem 527.98 (Closed-term substitution boundary substitution budget totality)**.** *Every substitution consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet factors through source closedness, value closedness, the closed shift row, the closed substitution row, the finite ledger, the audit row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. Equivalently, the consumer route is exhausted by*

$$C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

inside the carrier over T, V, d*, and it receives no subject reduction theorem, host syntax quotient, hidden compiler state, or host substitution semantics.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionBudgetT`

Proof. Start with the closed-source budget surface of Definition 527.97. It provides the entry condition: C_{src} and C_{val} are exposed before the operation rows Q and R can be spent.

Next apply Definition 527.12. It forces downstream consumers to read the closed shift and substitution boundary through the displayed carrier, transports, routes, provenance, and local naming row.

The audit and ledger rows close the remaining path. By Definition 527.8 and Definition 527.36, the finite ledger and audit row exhaust the operation budget; by Definition 527.38, the exported surface is exactly $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. □

Theorem 527.99 (Closed-term substitution boundary consumer audit triad)**.** *For* ClosedTermSubstitutionBoundary *the closed-source budget surface of Definition 527.97, the substitution budget totality row of Definition 527.98, and the non-escape audit rows of the chapter assemble the consumer-facing triad. The triad says that every substitution-facing consumer first reads closedness, then spends only the displayed operation boundary, and finally remains audit-visible through L, A, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerAuditTriad`

Proof. First take closedness as the entry surface. The closed-source budget theorem exposes C_{src} and C_{val} before any operation row is consumed.

Second take operation totality as the consumer boundary. The substitution budget theorem factors every consumer through Q, R, L, A, H, C, P, N over the same T, V, d carrier rows.

Third attach the audit witness. Apply Definition 527.37, Definition 527.44, and Definition 527.45. These rows keep the consumer inside the displayed BHist, hsame, Cont, Pkg, and NameCert packet and prevent escape to host substitution semantics, hidden compiler state, or an untracked audit route. □

Theorem 527.100 (Closed-term substitution boundary closed-source downstream coverage)**.** *Every accepted local consumer of the closed-source downstream surface of a* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet is exhausted by the displayed alternatives*

$$C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

over the carrier rows T, V, d. The only admissible operation alternatives are the closed shift row Q, the closed substitution row R, the ledger rows L, A, the componentwise hsame *transport row H, the displayed* Cont *route row C, the* Pkg *provenance row P, and the local* NameCert *row N. Host substitution semantics, ambient subject reduction, host syntax quotients, hidden evaluator state, hidden compiler state, external substitution relations, unregistered free-variable coordinates, and additional operation rows remain outside the BEDC packet.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryClosedSourceDownstr`

Proof. Start with the closed-source budget surface of Definition 527.97. It exposes the finite packet rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ and places C_{src} and C_{val} before any use of the closed shift budget Q or closed substitution budget R. Thus a local consumer cannot enter the downstream operation boundary through a source other than the displayed closedness rows.

Apply substitution budget totality from Definition 527.98. It says that every substitution consumer factors through source closedness, value closedness, the closed shift row, the closed substitution row, the finite ledger, the audit row, componentwise transports, displayed continuation routes, package provenance, and the local naming row. Therefore the downstream consumer has exactly the listed alternatives $C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ over T, V, d.

Finally use the consumer-facing triad of Definition 527.99. The triad identifies the same closedness entry, operation budget, and audit-visible route as one finite BEDC packet. Since the closed-source surface and the substitution budget totality contain no coordinate for host substitution semantics, ambient subject reduction, host syntax quotients, hidden evaluator state, hidden compiler state, external substitution relations, unregistered free-variable coordinates, or an extra operation row, those excluded alternatives remain outside the packet for every accepted local consumer. □

Theorem 527.101 (Closed-term substitution boundary finite readiness surface)**.** *The scoped finite packet for* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *is the paper-side readiness surface for downstream local consumption. Its readable object is exactly the displayed packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N,$$

where the carrier, classifier, local NameCert *obligations, ledger non-escape row, consumer boundary, audit-boundary factorization, and root* NameCert *handoff already identify the same finite BEDC surface.*

Proof. Start with Definition 527.1 and Definition 527.2. They display the term, value, depth, closedness, operation, ledger, audit, transport, continuation, provenance, and naming rows as one finite packet.

Next apply Definition 527.3, Definition 527.8, and Definition 527.12. These rows force the local naming obligations, closed operation reads, and downstream consumers to stay inside that packet.

Finally use Definition 527.44 and Definition 527.113. The audit route and root naming handoff repack the same rows for the local paper-side consumer, so the readiness surface is the scoped finite packet and not an additional host object. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryFiniteReadinessSurf`

Theorem 527.102 (Closed-term substitution boundary formal surface exhaustion)**.** *Every formal consumer of the scoped* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet may read only*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The consumer receives no host substitution semantics, subject reduction theorem, host syntax quotient, hidden evaluator, unchecked compiler state, or unregistered operation row outside the displayed carrier, ledger, audit, transport, route, provenance, and local NameCert *surface.*

Proof. Read the formal consumer through the carrier and classifier of Definition 527.1 and Definition 527.2. Those definitions expose exactly the displayed rows and no host-level substitution relation.

Apply Definition 527.8, Definition 527.12, and Definition 527.44. The ledger blocks operation escape, the consumer boundary blocks detached downstream reads, and the audit factorization keeps each admissible read inside L, A, H, C, P, N.

The excluded alternatives are therefore outside the formal surface. A subject reduction theorem, host syntax quotient, hidden evaluator, unchecked compiler state, or unregistered operation row would add a coordinate not present in $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryFormalSurfaceExhaus`

Theorem 527.103 (Closed-term substitution boundary downstream root handoff)**.** *The downstream roots* BinderBudgetSeal$^{\uparrow}$, ClosedTermSubstitutionCompiler$^{\uparrow}$, RecursorGenerator$^{\uparrow}$, AuthorizedGeneratorRecursor *and* MetaCICNormalizationFrontier$^{\uparrow}$ *consume the accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet only through the displayed carrier, ledger, audit, transport, route, provenance, and local* NameCert *rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Their common handoff is the root NameCert *packet produced by the carrier, classifier, local obligations, ledger non-escape, consumer boundary, audit-boundary factorization, root route locks, and root transport accountability rows; it exports no separate substitution semantics or compiler state.*

Proof. First collect the base packet. By Definition 527.1 and Definition 527.2, the finite rows are displayed as one carrier surface, and Definition 527.3 keeps the same local naming surface attached to them.

Next collect the binder and compiler restrictions. Use Definition 527.13 and Definition 527.15 to route BinderBudgetSeal$^{\uparrow}$ and ClosedTermSubstitutionCompiler$^{\uparrow}$ through the closed-boundary packet.

The recursor and normalization roots use the same packet. Apply Definition 527.16, Definition 527.17, and Definition 527.20 to keep RecursorGenerator$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, and MetaCICNormalizationFrontier$^{\uparrow}$ on the displayed route.

Finally apply Definition 527.83, Definition 527.93, and Definition 527.113. These root rows package the shared carrier, ledger, audit, transport, route, provenance, and local naming data as the only handoff surface for the downstream roots. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryDownstreamRootHando`

Theorem 527.104 (Closed-term substitution boundary route determinacy)**.** *Let* $S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A,$ *be an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet. If two downstream closed-boundary reads agree on the displayed carrier rows and on the continuation row* C*, then they reach the same closed shift row* Q *and the same substitute-closed row* R*. The determined route is the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet, not a separate host substitution semantics.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRouteDeterminacy`

Proof. Start with Definition 527.1 and Definition 527.2. They make $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ the only accepted packet rows, so agreement on the carrier rows already fixes the visible source, value, depth, closedness, ledger, audit, transport, provenance, and local naming data.

Next use Definition 527.69. Once the closed-boundary read has the same displayed rows and the same continuation row C, the operation readback cannot choose a different source/value convention or an unregistered operation coordinate.

Finally apply Definition 527.83. The downstream route must pass through the displayed packet before any self-compilation consumer can read it. Hence two such reads with the same carrier rows and continuation row reach the same Q and R, and no host substitution semantics is available to split the route. □

Theorem 527.105 (Closed-term substitution boundary context stack non-escape)**.** *Every recognizer, tuple, and formal-readiness route for an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet recovers only*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The route cannot add an open-term coordinate, host syntax quotient, hidden evaluator, compiler correctness theorem, or subject-reduction theorem to the context stack.

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryContextStackNonesca

Proof. Read the recognizer side through Definition 527.64. The recognizer route reaches the boundary only after acceptance, generated recognition, tuple-image landing, and readback of the displayed packet rows.

The tuple and substitution readback restrictions are then supplied by Definition 527.65 and Definition 527.66. They keep the closed-substitution read inside the finite tuple image and exclude open-term coordinates, free-variable leaks, host syntax quotients, unchecked evaluator state, compiler correctness, and ambient subject reduction.

Finally use Definition 527.101 and Definition 527.102. The finite readiness surface is exactly the same finite packet for the cited closed-term substitution boundary object, so the context stack has no coordinate outside $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. □

Theorem 527.106 (Closed-term substitution boundary substitution-order exhaustion)**.** *Every substitution-row consumer for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must read the source closedness row* C_{src}*, value closedness row* C_{val}*, ledger row* L*, and audit row* A *before consuming* R*. After those rows are visible, the route is exhausted by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySubstitutionOrderEx

Proof. First apply Definition 527.57. It places every root substitution read behind the source and value closedness witnesses, so R cannot be consumed before C_{src} and C_{val}.

Next use Definition 527.58 and Definition 527.89. These rows force the substitute-closed request through Q, R, L, A, H, C, P, N and make both operation rows total only over the displayed root operation surface.

Finally combine the ledger side with Definition 527.8. The ledger and audit rows block escape to host substitution semantics, hidden evaluator state, compiler state, or subject reduction. Therefore the remaining substitution-order route is exhausted by the displayed BHist, hsame, Cont, Pkg, and NameCert packet. □

Corollary 527.107 (Closed-term substitution boundary route-determinacy strict obstruction)**.** *Let* $S = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N)$ *be an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet. Any displayed failure, refusal, separation, or countermodel row used against a closed-term substitution boundary converse or reuse route is a strict local obstruction only to the named route that tries to read* Q *or* R *without the displayed packet rows and the continuation row* C*. It cannot be read as a global impossibility theorem, as a host substitution theorem, or as hidden external validation for a syntax quotient, evaluator, compiler-correctness theorem, or subject-reduction theorem.*

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRouteDeterminacyStr

Proof. The available packet surface is fixed by Definition 527.1 and Definition 527.2. A consumer-facing closed-boundary read has only the displayed rows

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Thus the route, transport, and readback inventory is the finite BHist packet together with componentwise hsame transport, the displayed Cont row C, the Pkg provenance row P, and the local NameCert row N.

Apply Definition 527.105. It says that every recognizer, tuple, and formal-readiness route recovers only the listed packet rows and cannot add an open-term coordinate, host syntax quotient, hidden evaluator, compiler correctness theorem, or subject-reduction theorem to the context stack. Hence any consumer-facing use of a displayed failure, refusal, separation, or countermodel row must factor through the same finite inventory. If the proposed converse or reuse route asks for an unlisted coordinate, the route is rejected before it can reach Q or R.

Now use Definition 527.104. When two downstream reads do agree on the displayed carrier rows and on C, the closed shift row Q and substitute-closed row R are already determined by the displayed BHist, hsame, Cont, Pkg, and NameCert packet. Therefore a displayed obstruction can block only the named attempt to bypass, reverse, or reuse that local route. It has no remaining target from which to assert an absolute impossibility, a host-level substitution semantics, or an external validation theorem outside the accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet. □

Theorem 527.108 (Closed-term substitution boundary root route exhaustion)**.** *Every root route read of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is exhausted by the finite carrier rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The route may read closedness through C_{src} *and* C_{val}*, shift through* Q*, substitution through* R*, ledger accounting through* L*, audit non-escape through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. It has no root coordinate for host substitution semantics, subject reduction, host syntax quotienting, hidden evaluator state, hidden compiler state, or an unlisted operation row.*

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootRouteExhaustion

Proof. Start with the carrier and the NameCert obligations. Definition 527.1 displays exactly the rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$, and Definition 527.3 makes the closedness, operation, ledger, audit, transport, continuation, package, and naming rows the local certificate surface.

Next apply the ledger and root route rows. Definition 527.8 blocks free-variable, host-equality, compiler-state, and subject-reduction escape, while Definition 527.113 packages the root handoff as the same finite BHist, hsame, Cont, Pkg, and NameCert route.

Finally use the root route determinacy package. Since any downstream route is already locked to the displayed operation and handoff rows, the root read is exhausted by the stated carrier coordinates and cannot be replaced by an unlisted host or compiler surface. □

Theorem 527.109 (Closed-term substitution boundary root non-escape)**.** *No root consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet can escape from the displayed route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

to obtain a free-variable coordinate, host substitution semantics, subject reduction, host syntax quotienting, hidden evaluator state, hidden compiler state, or a compiler-correctness theorem. The closed shift and substitution reads remain guarded by the displayed closedness, ledger, audit, hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. First read the root consumer through Definition 527.108. That theorem fixes every accepted root route to the displayed carrier rows before any operation boundary is consumed.

Apply Definition 527.8. The ledger and audit row have no projection for free variables, host equality, unchecked compiler state, or ambient subject reduction, so those objects cannot be exported by a root read.

Now use Definition 527.113. The handoff gives downstream consumers one finite packet over BHist, hsame, Cont, Pkg, and NameCert, not a second host substitution or compiler-correctness surface. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootNonescape`

Theorem 527.110 (Closed-term substitution boundary root downstream readiness)**.** *The root route exhaustion and root non-escape rows make* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *ready for downstream self-compilation consumers exactly as a finite closed-boundary packet. A downstream chapter may consume the displayed* $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ *route, together with the NameCert obligation surface, ledger non-escape row, and root NameCert handoff, but it may not treat the packet as a host compiler theorem, a subject-reduction package, or an untracked substitution semantics.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootDownstreamReadi`

Proof. Begin with Definition 527.108. It supplies the finite operation route that downstream consumers may read.

Next apply Definition 527.109. The same route blocks host substitution semantics, subject reduction, host syntax quotienting, hidden evaluator state, hidden compiler state, and compiler-correctness import.

Finally compose Definition 527.3, Definition 527.8, and Definition 527.113. These rows repack the closedness, shift, substitution, ledger, audit, transport, continuation, provenance, and naming data as the precise downstream readiness surface. □

Corollary 527.111 (Closed-term substitution boundary root route strict local obstruction)**.** *Let* O *be a displayed failure, refusal, separation, or countermodel row cited against a named consumer-facing converse or reuse route in the root route exhaustion package for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet. Then* O *is a strict local obstruction. It blocks only the named route whose consumer-facing read factors through the finite packet rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and then tries to omit, replace, detach, or bypass one of those displayed coordinates. The same row cannot be reinterpreted as a global impossibility statement, a host theorem, hidden external validation, host substitution semantics, subject reduction, host syntax quotienting, hidden evaluator state, hidden compiler state, compiler-correctness evidence, or an obstruction to any consumer route that still remains inside the listed rows.

Proof. Start with Definition 527.108. Every root route read of the accepted packet is already exhausted by the listed rows. The theorem also identifies the only permitted readings of those rows: closedness is read through C_{src} and C_{val}, shift through Q, substitution through R, ledger accounting through L, audit non-escape through A, componentwise hsame transport through H, displayed Cont replay through C, Pkg provenance through P, and local naming through N. Thus a consumer-facing converse or reuse route has no root entry point except the displayed finite route.

Apply Definition 527.109. It excludes escape from the same route to a free-variable coordinate, host substitution semantics, subject reduction, host syntax quotienting, hidden evaluator state, hidden compiler state, or a compiler-correctness theorem. The closed shift and substitution reads therefore remain guarded by the displayed closedness, ledger, audit, transport, continuation, provenance, and local naming rows.

Now consider the cited row O. If the named converse or reuse route asks for one of the displayed coordinates and O records the failure, refusal, separation, or countermodel at that coordinate, then O blocks precisely that named local route. If the attempted reading asks instead for a global impossibility statement, a host theorem, hidden external validation, host substitution semantics, subject reduction, a host syntax quotient, hidden evaluator state, hidden compiler state, compiler-correctness evidence, or any unlisted operation row, the route exhaustion theorem supplies no coordinate for the request and the non-escape theorem explicitly excludes it. Hence O has force only as the strict local obstruction to the named converse or reuse route, while every consumer read that continues to factor through the displayed packet rows remains a read of the accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ boundary. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootRouteStrictLoca`

Theorem 527.112 (Closed-term substitution boundary root compiler non-escape)**.** *Any compiler-facing root read may cite* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *only after the accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet has exposed the displayed boundary rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The citation exports no subject reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, or host substitution semantics beyond the displayed closed shift and closed substitution boundary rows.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootCompilerNonesca`

Proof. Read the compiler-facing consumer through Definition 527.12. It allows a downstream consumer to see closed shift and substitute-closed rows only through the accepted finite packet.

Apply Definition 527.8 and Definition 527.36. Together they force the operation boundary through C_{src}, C_{val}, Q, R, L, and A before any compiler-facing citation can be formed.

The remaining coordinates are closed by Definition 527.44. The compiler citation is therefore a read of the boundary packet, not an exported subject-reduction proof, host syntax quotient, hidden evaluator, hidden compiler state, or host substitution semantics. □

Theorem 527.113 (Closed-term substitution boundary root NameCert handoff)**.** *The root audit totality row of Definition 527.45, the root compiler non-escape row of Definition 527.112, and the scoped closure surface of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *form the root-unblock handoff for downstream self-compilation consumers. The handoff is the displayed finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route, not a host substitution semantics or hidden compiler object.*

Proof. First package the audit side. By Definition 527.45, every root audit consumer factors through $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$.

Next package the compiler side. By Definition 527.112, a compiler-facing root read can cite the compiler only after the same closed shift and substitution boundary rows are visible.

Finally combine these rows with the scoped closure packet recorded below. The downstream self-compilation consumer receives one finite handoff over the displayed carrier, classifier, ledger,

audit, transport, route, provenance, and local naming rows; no subject reduction theorem, host syntax quotient, hidden evaluator, or hidden compiler state is exported. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootNamecertHandoff

Theorem 527.114 (Closed-term substitution boundary root audit replay totality)**.** *For every accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet, each root audit replay read through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *is total on the displayed finite rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The replay may inspect carrier rows, classifier rows, ledger non-escape rows, root audit totality rows, and root compiler non-escape rows, but it exports no host substitution semantics, subject-reduction theorem, host syntax quotient, hidden evaluator state, or hidden compiler state.

Proof. Start with the carrier and classifier of Definition 527.1 and Definition 527.2. They give the replay a finite packet over term rows, closed-value rows, depth, closedness witnesses, operation rows, ledger rows, audit rows, transports, continuations, package provenance, and naming.

Next apply Definition 527.8. The ledger row prevents an audit replay from leaving the displayed packet for unregistered free variables, host equality, unchecked compiler state, or ambient subject reduction.

Use Definition 527.45 to make the root replay total. Every root audit consumer is already routed through the same $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$ coordinates.

Finally apply Definition 527.112. Compiler-facing citations are allowed only after the same boundary rows are visible, so the replay remains a BHist, hsame, Cont, Pkg, and NameCert read of the accepted packet. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootAuditReplayTota

Theorem 527.115 (Closed-term substitution boundary root substitution route exactness)**.** *In an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet, a root read of closed substitution is exact precisely when it factors through the carrier, classifier, closedness ledger, substitution row R, audit row A, componentwise* hsame *transport rows H, displayed* Cont *routes C, package row P, and local* NameCert *row N. The exact route is*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

No root substitution route may be replaced by host substitution semantics, an ambient compiler theorem, a subject-reduction package, a host syntax quotient, or an untracked evaluator state.

Proof. Read the operation surface through Definition 527.36. It locks the closed shift and substitute-closed rows to C_{src}, C_{val}, Q, R, L, and A before any root consumer can export them.

Next use Definition 527.12. A downstream consumer sees the substitution boundary only as an accepted packet read through the carrier and classifier, not as an external substitution relation.

Apply Definition 527.113. The handoff packages the audit and compiler sides as one finite BHist, hsame, Cont, Pkg, and NameCert route.

The route is exact by Definition 527.114: every admissible replay has the displayed rows, and ledger non-escape blocks every unlisted host or compiler coordinate. □

Lean4 ✓ BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootSubstitutionRou

Theorem 527.116 (Closed-term substitution boundary root ledger readiness)**.** *The root ledger of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet is ready for downstream self-compilation consumers exactly when the carrier, classifier, ledger non-escape, root audit replay totality, root substitution route exactness, root audit totality, and root compiler non-escape rows are all read through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

This readiness row exports a finite closed-term substitution route only; it does not export host substitution semantics, full compiler correctness, subject reduction, a host syntax quotient, hidden evaluator state, or hidden compiler state.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootLedgerReadiness`

Proof. First assemble the base packet from the carrier, classifier, and local NameCert obligations. These rows determine the displayed closed-source, closed-value, operation, ledger, audit, transport, continuation, package, and naming coordinates.

Next add non-escape through Definition 527.8. The ledger cannot gain a hidden free-variable row, host equality row, compiler-state row, or subject-reduction row.

Use Definition 527.114 and Definition 527.115. The first makes the root audit replay total on the packet, and the second fixes closed substitution to the same finite route.

Finally combine the original root audit and compiler rows, Definition 527.45 and Definition 527.112. Downstream self-compilation consumers therefore receive one ready ledger packet over BHist, hsame, Cont, Pkg, and NameCert, with no ambient compiler semantics attached. □

Corollary 527.117 (Closed-term substitution boundary root compiler strict local obstruction)**.** *For an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet, the obstruction supplied by the root compiler non-escape row is strict and local. It blocks only a compiler-facing root read that either cites* ClosedTermSubstitutionCompiler$^{\uparrow}$ *before the displayed boundary rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

are exposed, or reuses that citation as a subject-reduction theorem, host syntax quotient, hidden evaluator, hidden compiler state, host substitution semantics, or hidden external validation. After the read factors through the root NameCert *handoff, every consumer-facing use is only the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route of the packet. Thus the obstruction cannot be reinterpreted as a global impossibility claim, a host theorem, or a validation source outside the displayed closed-term substitution boundary.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRootCompilerStrictL`

Proof. Start with Definition 527.112. It gives the exact admission condition for a compiler-facing root read: the citation of ClosedTermSubstitutionCompiler$^{\uparrow}$ is allowed only after the accepted ClosedTermSubstitutionBoundary$^{\uparrow}$ packet has exposed the displayed rows $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$. The same theorem also lists the forbidden exports: subject reduction, a host syntax quotient, hidden evaluator state, hidden compiler state, and host substitution semantics. Therefore the obstruction applies to this citation boundary and to those reuse routes, not to an ambient host theory.

Now apply Definition 527.113. The handoff identifies the downstream self-compilation surface with the finite BHist, hsame, Cont, Pkg, and NameCert route obtained from the root audit totality

row and the root compiler non-escape row. Hence a consumer-facing use of the compiler citation factors through the named packet rows, and an unlisted coordinate has no handoff projection.

Finally use the replay and readiness rows in this body file. By Definition 527.114, replay is total on exactly the displayed finite route and exports no host substitution semantics, subject-reduction theorem, host syntax quotient, hidden evaluator state, or hidden compiler state. By Definition 527.115 and Definition 527.116, the downstream root ledger remains the same closed-term substitution packet rather than an external compiler or validation object. Thus the non-escape row is a local obstruction to the named bad reuse of this packet, and it carries no claim of global impossibility or hidden host validation. □

Theorem 527.118 (Closed-term substitution boundary primitive ledger scope)**.** *Every accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *read that exposes closed shift or closed substitution binds its primitive ledger scope to the displayed* Term *rows,* ClosedAt *rows,* shift *row,* substitute *row,* BHist *packet, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. No accepted read obtains a closed substitution boundary from host syntax, host equality, unchecked compiler state, subject reduction, quotient representation, hidden evaluator state, or an operation row outside that primitive ledger.*

Proof. Start with the carrier and classifier. They display the source term, closed value, inspected depth, closedness rows, closed shift, closed substitution, ledger, audit row, transport rows, continuation routes, package provenance, and local naming row as one finite packet.

Apply the NameCert obligations and ledger non-escape theorem. The closedness rows must be read before the operation rows, and the ledger has no projection for host substitution semantics, hidden equality, unchecked compiler state, or subject reduction.

Finally use the root route exhaustion package. Every root read is already routed through the same BHist, hsame, Cont, Pkg, and NameCert surface, so the primitive ledger scope is exactly the listed boundary tuple. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryPrimitiveLedgerScop`

Theorem 527.119 (Closed-term substitution boundary downstream factorization triad)**.** *Every downstream consumer in* BinderBudgetSeal$^{\uparrow}$, ClosedTermSubstitutionCompiler$^{\uparrow}$, *and* RecursorGenerator$^{\uparrow}$ *that reads closed-term substitution factors through the same finite* ClosedTermSubstitutionBoundary$^{\uparrow}$ *tuple*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

before it may cite a substitution row. The binder-budget, compiler-schedule, and recursor-output routes may repack that tuple for their own local certificates, but none of them receives an independent substitution semantics, hidden binder convention, compiler theorem, evaluator state, or subject-reduction package.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryDownstreamFactoriza`

Proof. First use the BinderBudgetSeal factorization theorem. It forces every binder-budget shift or substitution read through the displayed closedness witnesses, operation rows, ledger, audit row, transport, route, provenance, and local naming row.

Next use the ClosedTermSubstitutionCompiler factorization theorem. The compiler schedule sees a closed substitution row only after the same boundary packet has exposed the source and value closedness rows together with Q, R, L, A, H, C, P, N.

Finally use the RecursorGenerator handoff theorem. The generated branch output may consume closed substitution only through the displayed continuation route, so the three downstream consumers share the same finite boundary tuple before any substitution read is visible. □

Theorem 527.120 (Closed-term substitution boundary NameCert threshold package)**.** *The primitive ledger scope of Definition 527.118, downstream factorization triad of Definition 527.119, consumer boundary of Definition 527.12, and ledger non-escape theorem of Definition 527.8 assemble the* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *root threshold package. The package exports the finite* Term*,* ClosedAt*,* shift*,* substitute*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route required by downstream self-compilation consumers, and it exports no hidden host substitution, compiler theorem, quotient representative, evaluator state, free-variable coordinate, or subject-reduction row.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryNameCertThresholdPa`

Proof. Begin with primitive ledger scope. It identifies the accepted closed-boundary read with the BEDC primitive rows that can be projected from the finite packet.

Add the consumer boundary and ledger non-escape rows. They require every downstream closed substitution read to remain inside the displayed packet and rule out escape to host syntax, compiler state, subject reduction, or unregistered free variables.

Finally compose the downstream factorization triad. Since the binder-budget, compiler-schedule, and recursor-output consumers all pass through the same boundary tuple, the four rows form the root threshold package counted by downstream self-compilation consumers. □

Theorem 527.121 (Closed-term substitution boundary route saturation)**.** *Every accepted route of a* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is exhausted by*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. A route may read the term row, value row, inspected depth, source and value closedness rows, closed shift row, substitute-closed row, finite ledger, audit row, componentwise transport, displayed continuation, package provenance, and local naming row, but it has no additional coordinate for a host substitution relation, host equality witness, unchecked compiler state, subject-reduction theorem, or unregistered free variable.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRouteSaturation`

Proof. Start from the carrier and classifier of Definition 527.1 and Definition 527.2. They make the displayed rows the complete carrier surface and allow only componentwise hsame transport and recorded Cont reads across that surface.

Next apply the local NameCert obligations of Definition 527.3. Source closedness, value closedness, substitution-row exactness, ledger accounting, audit visibility, package provenance, and local naming are all attached to the same finite packet.

Finally use source/value admission, substitution-row exactness, and ledger non-escape from Definition 527.4, Definition 527.5, Definition 527.7, and Definition 527.8. These rows leave no route outside the listed BEDC primitives, so the accepted route is saturated by the displayed packet. □

Theorem 527.122 (Closed-term substitution boundary NameCert source lock)**.** *In every accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet, the source closedness row* C_{src} *and the value closedness row* C_{val} *remain locked before any closed shift or substitute-closed read. Thus a consumer must read the source/value closedness rows, the local* NameCert *obligations, and the finite ledger before reading* Q *or* R *through a shift/substitution route.*

Proof. Read the accepted packet through Definition 527.1. The rows C_{src} and C_{val} are explicit carrier coordinates, not hidden side conditions.

Use the classifier and local NameCert obligations from Definition 527.2 and Definition 527.3. They make closed-source admission and closed-value admission prerequisites for the operation rows.

Apply Definition 527.4, Definition 527.5, Definition 527.7, and Definition 527.36. The shift/substitution read is therefore locked behind the source/value rows and the finite ledger before Q or R can be consumed. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryNamecertSourceLock`

Theorem 527.123 (Closed-term substitution boundary consumer non-escape)**.** *No consumer of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet can recover an unregistered free variable, host equality, unchecked compiler state, subject reduction, or host substitution semantics. Every consumer-facing read is confined to the carrier, classifier, source/value admission rows, substitution-row exactness, finite ledger, audit non-escape row, and local* NameCert *route over*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryConsumerNonescape`

Proof. First inspect the carrier, classifier, and NameCert obligations. The only available rows are the term, value, depth, closedness, operation, ledger, audit, transport, continuation, package, and local naming rows displayed in the accepted packet.

Next use source/value admission and substitution-row exactness. By Definition 527.4, Definition 527.5, and Definition 527.7, the operation rows are readable only through the displayed closedness and ledger surface.

Finally apply Definition 527.8, together with the consumer boundary supplied by Definition 527.12. The ledger and consumer rows contain no coordinate for free variables, host equality, unchecked compiler state, subject reduction, or host substitution semantics, so no consumer can recover those objects from the packet. □

Corollary 527.124 (Closed-term substitution boundary strict local obstruction)**.** *Let O be a displayed failure, refusal, separation, or countermodel row cited against a named converse or reuse route for an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet. If the named route is consumer-facing, then O is a strict local obstruction: it blocks only the attempted read that factors through*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

and then tries to omit, replace, detach, or bypass one of the displayed packet coordinates or one of the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The same row cannot be reinterpreted as a global impossibility statement, host theorem, hidden external validation, host substitution semantics, host equality witness, unchecked compiler state, subject-reduction theorem, or unregistered free-variable route.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryRouteSaturationStri`

Proof. By Definition 527.121, every accepted route of the packet is already exhausted by the displayed finite carrier rows and by the listed BEDC primitives. Thus a converse or reuse route has no independent entry point for a host substitution relation, host equality witness, unchecked compiler state, subject-reduction theorem, or unregistered free variable. Its consumer-facing content

must first be a read of the term row, value row, inspected depth, source and value closedness rows, operation rows, finite ledger, audit row, componentwise transport, displayed continuation, package provenance, and local naming row.

Now apply Definition 527.123. A consumer cannot recover an unregistered free variable, host equality, unchecked compiler state, subject reduction, or host substitution semantics from those rows. Therefore any consumer-facing use of O can reach only the named local route whose read has already factored through the displayed packet. If that route asks to omit, replace, detach, or bypass one of the listed coordinates, the row O blocks that named attempt.

The remaining interpretations ask for something outside the accepted route surface: a global impossibility statement, a host theorem, hidden external validation, host substitution semantics, host equality, unchecked compiler state, subject reduction, or an unregistered free-variable route. Route saturation supplies no coordinate for such a request, and consumer non-escape explicitly prevents recovering it from the packet. Hence O has force only as the strict local obstruction to the named converse or reuse route, while every consumer read that stays inside the displayed rows remains a read of the accepted $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet. □

Theorem 527.125 (Closed-term substitution boundary MetaCIC public handoff)**.** *Every* AuthorizedGeneratorRecu *or* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *consumer that cites an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must pass through the displayed public handoff rows*

$$\mathsf{Term}, \mathsf{ClosedAt}, \mathsf{shift}, \mathsf{substitute}, L, A, H, C, P, N.$$

The rows are read as the source term and value term entries, source and value closedness entries, closed shift and substitute-closed operation entries, finite ledger L, audit row A, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The handoff exports no subject-reduction theorem, host substitution semantics, compiler correctness theorem, host syntax quotient, hidden evaluator state, or unregistered normalization route.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryMetacicPublicHandof`

Proof. Project the accepted carrier of Definition 527.1 and the classifier of Definition 527.2. They display the term rows, closedness rows, closed shift row, substitute-closed row, ledger, audit, transport, continuation, provenance, and local naming rows as one finite packet.

Apply Definition 527.3, Definition 527.8, and Definition 527.43. These rows force every closed operation read to pass through the ledger and audit surface before the operation is exported.

Now use Definition 527.103 and Definition 527.126 when the consumer is MetaCIC-facing. The downstream root handoff lists $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ and $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ among the roots that consume only the displayed packet, while the MetaCIC consumer boundary blocks confluence or subject-reduction reads from leaving the same rows.

Thus the public handoff for these two consumers is exactly the displayed Term, ClosedAt, shift, substitute, ledger, audit, hsame, Cont, Pkg, and NameCert route. Any subject-reduction theorem, host substitution semantics, compiler correctness theorem, host syntax quotient, hidden evaluator state, or unregistered normalization route would require a coordinate outside the accepted packet. □

Theorem 527.126 (Closed-term substitution boundary MetaCIC consumer nonescape)**.** *Every MetaCIC confluence or subject-reduction consumer of an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet must read the source closedness row, value closedness row, shift boundary row, substitution boundary row, finite ledger, audit row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg

provenance, and local NameCert *row before citing the boundary. The consumer-facing route is confined to*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

It supplies no full subject-reduction theorem, compiler correctness theorem, host substitution semantics, selected term representative, or unregistered free-variable data.

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryMetacicConsumerNone`

Proof. Start from source and value closedness admission. By Definition 527.4 and Definition 527.5, the rows C_{src} and C_{val} are displayed before either operation row can serve a downstream consumer.

Next apply substitution-row exactness. The theorem Definition 527.7 forces the substitution boundary row R to remain attached to the same source row, value row, depth row, and closedness witnesses, while the shift row Q stays inside the same operation surface.

Finally use ledger non-escape, together with the route-saturation package. The rows L, A, H, C, P, N record the closedness and operation reads, transport them componentwise, replay them through displayed continuation, and name the local boundary. By Definition 527.8 and Definition 527.121, a MetaCIC confluence or subject-reduction consumer receives only that finite packet and no host substitution semantics, compiler correctness theorem, full subject-reduction theorem, selected representative, or unregistered free-variable data. □

Theorem 527.127 (Closed-term substitution boundary MetaCIC dependency)**.** *Every MetaCIC consumer of an accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet must route through the closed source row, closed value row, shift boundary row, substitution boundary row, ledger row, audit row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before the packet can feed a normalization or confluence surface. In particular, any handoff to the critical-path consumer surface of chapter 765 and to a confluence-facing sibling such as chapter 7346 is mediated by the existing closed rows; it does not export subject reduction, compiler correctness, host substitution semantics, a selected representative, or unregistered free-variable data.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryMetacicDependency`

Proof. Start with Definition 527.126. It already forces every MetaCIC confluence or subject-reduction consumer to expose the source closedness, value closedness, shift boundary, substitution boundary, finite ledger, audit, transport, replay, provenance, and local naming rows.

Next read the MetaCIC critical-path surface. The critical-path packet of chapter 765 consumes closed substitution support only as a displayed dependency row before normalization or confluence traffic can continue; it does not turn that dependency into subject reduction or compiler correctness.

Finally pass to the confluence-facing sibling. A local confluence frontier such as chapter 7346 may receive the closed substitution boundary as a premise row, but the nonescape theorem keeps the route attached to the same closed source and value rows. Thus the dependency is horizontal and finite, not a new exported metatheorem. □

Theorem 527.128 (Closed-term substitution boundary socket separation)**.** *For every accepted* ClosedTermSubstitutionBoundary$^{\uparrow}$ *packet, the source and value closedness rows, shift boundary row, substitution boundary row, ledger row, and audit row keep the free-variable socket separate from every subject-reduction consumer. Any consumer that asks for a confluence, normalization, or subject-reduction theorem must first read the finite boundary route*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

through Definition 527.40, Definition 527.121, and Definition 527.126; the packet exports no unregistered free-variable coordinate, host substitution semantics, selected term representative, or subject-reduction theorem.

Proof. Project the accepted carrier and classifier from Definition 527.1 and Definition 527.2. The only public operation reads are the displayed shift and substitution boundary rows, and both are guarded by the source/value closedness witnesses and the finite ledger.

Apply Definition 527.40. The ledger and audit rows are read before any operation boundary is exported, so a free-variable socket cannot be recovered by bypassing C_{src}, C_{val}, L, or A.

Next use Definition 527.121. Every accepted route is exhausted by the displayed rows and the BEDC primitives BHist, hsame, Cont, Pkg, and NameCert, leaving no coordinate for host substitution semantics or selected term representatives.

Finally apply Definition 527.126. MetaCIC confluence and subject-reduction consumers receive only the finite boundary packet, so the free-variable socket remains separate from any subject-reduction theorem. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundarySocketSeparation`

Theorem 527.129 (Closed-term substitution boundary audit route totality)**.** *Every audit-visible consumer route for an accepted* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *packet is total only after it displays*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N.$$

The route may then replay the carrier, classifier, local NameCert *obligations, ledger separation, root audit totality, and socket-separation rows, but it cannot start from a subject-reduction theorem, host syntax quotient, hidden compiler state, unregistered free-variable socket, or host substitution semantics.*

Proof. Begin with the root audit totality theorem Definition 527.45. It already routes every audit-visible consumer through the displayed boundary rows before the packet can be consumed.

Use the NameCert obligations and the ledger route separation. By Definition 527.3 and Definition 527.40, the source/value closedness rows, operation rows, ledger row, audit row, transport rows, continuation row, provenance row, and local naming row are all read inside the same finite packet.

Now apply Definition 527.128. The free-variable socket is not a hidden entry point into subject reduction; it is blocked unless the same displayed boundary route has already been consumed.

Therefore every audit-visible consumer route is total exactly on the listed rows. Since none of those rows contains a subject-reduction theorem, host syntax quotient, hidden compiler state, unregistered free-variable socket, or host substitution semantics, no audit-visible route can read such data before reading the packet. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryAuditRouteTotality`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ packet is generated from displayed Term rows, ClosedAt witnesses, a depth row, closed shift and substitute boundary rows, a finite closedness ledger, an audit-visible non-escape row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 527.1, Definition 527.2, Definition 527.3, Definition 527.8, Definition 527.12, Definition 527.30, Definition 527.31, Definition 527.32, Definition 527.45, Definition 527.125, Definition 527.126, and Definition 527.130 bind closed-term substitution as a mature finite audit route over Term, ClosedAt, shift, substitute, BHist, hsame, Cont, Pkg, and NameCert, with the paper-side bridge route recorded by the sibling status ledger.

Formal verification: FormalStatus(NameCert$_{\text{ClosedTermSubstitutionBoundary}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close full compiler correctness, subject reduction, host syntax quotienting, hidden semantic models, or bridge checking.
Upgrade path: Further movement is on the verification and bridge-checking axes; the paper-side treatment is mature for the displayed closed-term substitution boundary.

Theorem 527.130 (Closed-term substitution boundary MetaCIC socket factorization)**.** *Every MetaCIC critical-path consumer of closed-term substitution that reaches candidate-mediated strong-normalization or confluence readiness must first factor through the displayed binder-budget, recursor-generator, shift-substitution, audit-route, and subject-reduction socket rows of the accepted* ClosedTermSubstitutionB *packet. Concretely, the consumer route must expose the finite boundary rows*

$$T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, A, H, C, P, N$$

together with the binder-budget and authorized-generator rows cited by the MetaCICCriticalPath$^{\uparrow}$ *route of chapter 765. Only after those rows are visible may the consumer read candidate-mediated SN or confluence readiness. The factorization exports no global subject reduction, full confluence, beta-decision procedure, host equality, quotient term equality, hidden substitution semantics, or selected term representative.*

Proof. Start with the closed-term substitution carrier and classifier, Definition 527.1 and Definition 527.2. They display the source term, value term, depth, source/value closedness, closed shift row, substitute-closed row, ledger, audit, transport, replay, provenance, and local naming rows before any substitution consumer can proceed.

Next lock the operation side. Apply Definition 527.122, Definition 527.121, and Definition 527.128. These rows force the shift-substitution read through the source/value closedness witnesses, the finite ledger, and the same BHist, hsame, Cont, Pkg, and NameCert route.

Now attach the MetaCIC side. The MetaCIC critical path of chapter 765 reads binder-budget, authorized recursor-generator, candidate-mediated SN, confluence, and subject-reduction socket data as separate rows. The nonescape theorem Definition 527.126 and the audit route totality theorem Definition 527.129 keep the closed-substitution part of that path inside the displayed boundary packet.

Finally use the candidate frontier rows from the critical-path chapter, including Definition 765.278 and Definition 765.281. Candidate-mediated SN and confluence readiness can consume the closed-substitution boundary only as one socket row among the displayed packet rows. Since neither chapter supplies a closed subject-reduction certificate, full confluence theorem, beta-decision procedure, host equality proof, quotient term equality proof, hidden substitution semantics, or selected representative, no such export is available at the factorized route. □

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryMetacicSocketFacto`

Theorem 527.131 (ClosedTermSubstitutionBoundary sibling dependency route)**.** *Every* ClosedTermSubstitutionB *consumer that reads the boundary as MetaCIC substitution data must first expose the sibling substitution surface of chapter 537 and the critical-path readback boundary of chapter 765. The* BinderBudgetSeal$^{\uparrow}$ *row of chapter 532 supplies the finite binder budget, and the* AuthorizedGeneratorRecursor$^{\uparrow}$ *row of chapter 525 supplies the permitted generator boundary before the substitution readback is consumed. The* ClosedTermSubstitution$^{\uparrow}$ *row supplies the operation-facing substitution surface, while the* MetaCICCriticalPath$^{\uparrow}$ *row supplies the critical-path socket where binder budgets, closedness ledgers, and substitution reads remain audit-visible. The route is finite: it records only the*

displayed substitution operation row, closedness boundary row, transport, replay, provenance, and local naming data. It does not claim beta-conversion, quotient term equality, unrestricted normalization, or a hidden subject-reduction theorem.

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionBoundaryUp.ClosedTermSubstitutionBoundarySiblingDependencyRo`

Proof. Project the closed-term substitution boundary carrier from the core construction. Its operation row is available only through the displayed source and value closedness witnesses, ledger row, audit row, transport, replay, provenance, and local naming certificate.

Read the sibling anchors as route boundaries. The ClosedTermSubstitution$^{\uparrow}$ chapter names the substitution operation surface consumed by the boundary, the BinderBudgetSeal$^{\uparrow}$ chapter supplies the finite binder budget, the AuthorizedGeneratorRecursor$^{\uparrow}$ chapter supplies the permitted generator boundary, and the MetaCICCriticalPath$^{\uparrow}$ chapter names the critical-path socket in which the readback remains visible to MetaCIC consumers.

Since the carrier and the two sibling routes expose no beta-conversion theorem, quotient syntax equality, unrestricted normalization theorem, or hidden subject-reduction certificate, the dependency route remains exactly the displayed finite handoff. □

Theorem 527.132 (ClosedTermSubstitutionBoundary normalization-router dependency route)**.** *Every* ClosedTermSubstitutionBoundary$^{\uparrow}$ *consumer that sends closed substitution data toward MetaCIC normalization must expose the candidate-mediated router of chapter 1108. The boundary supplies the closed projection row, while the router supplies candidate evidence, Pi and application adequacy, authorized generator, compiler recognizer, kernel acceptance, replay, provenance, and local naming rows before any normalization-facing read is consumed. Thus the dependency is horizontal and finite: it records a displayed route from closed-term substitution into the normalization router, not a host substitution theorem, evidence-free normalization theorem, subject-reduction theorem, or quotient reduction equality.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionBoundaryUp.ClosedTermSubstitutionBoundaryNormalizationRoute`

Theorem 527.133 (ClosedTermSubstitutionBoundary sibling lattice route)**.** *Every* ClosedTermSubstitutionBounda *consumer that treats the boundary as a closed substitution lattice must read four sibling rows before the local boundary is exported. The substitution operation surface is supplied by chapter 537; the MetaCIC socket and critical-path readback are supplied by chapter 765; the finite binder budget is supplied by chapter 532; and the permitted generator boundary is supplied by chapter 525. The route therefore exposes a finite lattice of operation, socket, budget, generator, transport, replay, provenance, and local naming rows, without exporting quotient term equality, unrestricted substitution, host normalization, or a hidden subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionBoundaryUp.ClosedTermSubstitutionBoundarySiblingLatticeRoute`

Proof. Read the existing sibling dependency route and the normalization-router dependency route. They already force the boundary carrier to pass through the closed substitution operation row, MetaCIC critical-path socket, binder-budget seal, and authorized-generator boundary before any normalization-facing consumer reads the substitution data.

The present route repacks those sibling reads as the local lattice boundary. Since each cited chapter contributes a displayed NameCert row and none contributes quotient syntax equality, unrestricted substitution, host normalization, or erased subject reduction, the lattice exports exactly the finite operation, socket, budget, generator, transport, replay, provenance, and local naming rows. □

Theorem 527.134 (ClosedTermSubstitutionBoundary MetaCIC sibling lattice route)**.** *Every MetaCIC critical-path consumer of* ClosedTermSubstitutionBoundary$^{\uparrow}$ *keeps the closed-substitution operation row, the binder-budget seal, the authorized-generator recursor row, the residual socket, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming visible before it exports the boundary to a critical-path read. The route refuses quotient term equality, unrestricted substitution, host normalization, hidden subject reduction, and kernel consistency.*

Lean4 ✓ `BEDC.Derived.ClosedtermsubstitutionboundaryUp.ClosedTermSubstitutionBoundaryMetaCICSiblingLatti`

Proof. Apply Definition 527.133. It already exposes the closed-substitution operation row, MetaCIC socket, binder-budget seal, authorized-generator boundary, transport, replay, provenance, and local naming.

Now read the MetaCIC critical-path dependency through chapter 765. The critical-path consumer can only reuse the displayed residual socket and the rows supplied by the sibling lattice; no new quotient term equality, unrestricted substitution, host normalization, hidden subject-reduction theorem, or kernel-consistency assertion is present in the packet. □

Proposition 527.135 (ClosedTermSubstitutionBoundary MetaCIC normalization dependency lattice)**.** *Every normalization-facing consumer of* ClosedTermSubstitutionBoundary$^{\uparrow}$ *reads the MetaCIC closed substitution route through chapter 537 and the closed substitution seal route through chapter 538 before it enters the normalization-router dependency route of Definition 527.132. The MetaCIC critical-path socket of chapter 765 and the normalization certificate router of chapter 1108 therefore form the displayed dependency lattice for the boundary packet: closed substitution supplies the operation row, the seal fixes the closed fragment, the critical path supplies the socket, and the router supplies the finite normalization-facing handoff. This lattice exports no unrestricted substitution theorem, subject-reduction theorem, host normalization theorem, quotient term equality, or hidden compiler state.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionBoundaryUp.ClosedTermSubstitutionBoundaryMetacicNormalizatio`

The ClosedTermSubstitutionBoundary$^{\uparrow}$ dossier records its bridged-closure route as a paper-side ledger row: the bridge route is read through the displayed closed-source, shift, substitution, audit, transport, replay, provenance, and local NameCert rows, while the verification axis remains bound only by the closurestatus block in `namecert_construction_core.tex`. This ledger row does not add a subject-reduction theorem, host syntax quotient, compiler-correctness theorem, or hidden substitution semantics.

Chapter 528

A Concrete Naming Certificate for $\mathsf{SupplySocketLedger}^{\uparrow}$

The $\mathsf{SupplySocketLedger}^{\uparrow}$ surface is the concrete NameCert ledger for the seven T-supply socket kinds of chapter 9394. It records which kind is being exposed, which site consumes the request, which audit gate governs the refusal, and which provenance and continuation rows make the refusal available to later concrete-instance chapters. The ledger is a finite classifier over displayed rows; it is not a hidden source of the requested supply.

The socket-site row S is the ledger projection of the supply socket exposed by chapter 498, the audit row A consumes the admission and refusal discipline of chapter 545, and downstream public-record consumers such as chapter 519 may cite only the displayed GAP, kind, site, audit, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Definition 528.1 (SupplySocketLedger carrier)**.** *A* $\mathsf{SupplySocketLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, K, S, A, C, P, N)$$

with the following displayed rows:

1. *G is the* GAP *tag row naming the boundary socket under inspection;*
2. *K is one of the seven supply kinds: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure;*
3. *S is the displayed socket site, recording where the requested supply would be consumed and which concrete theorem, definition, audit gate, or consumer route would use it;*
4. *A names the audit gate that refuses silent internalisation of the requested supply kind;*
5. *C records the* Cont *consumption routes by which downstream consumers read G, K, S, and A without obtaining the missing supply;*
6. *P records* Pkg *provenance over* BHist*, the* GAP *row, the seven supply kinds, socket sites, audit gates,* Cont *routes, and the local naming surface;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unrestricted induction principle, selected witness, quotient equality, empirical persistence law, shared host history, inscription source, or truth-total self-description.

Theorem 528.2 (SupplySocketLedger NameCert obligations)**.** *For the carrier of Definition 528.1, the local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (G, K, S, A, C, P, N),$$

classifier exposure for the row K, kind exactness against the seven displayed supply kinds, audit-gate routing through A, continuation locality through C, package provenance through P, local naming through N, and ledger non-escape for any requested supply not present in the finite packet.

Proof. Carrier admission is projection from Definition 528.1. The packet exposes the GAP tag, supply-kind row, socket site, audit gate, continuation routes, provenance, and local naming row as its complete visible inventory.

Classifier and kind exactness are read from K. The row is required to be one of the seven displayed supply kinds, and the NameCert surface accepts no classifier outside that finite list.

Audit routing and consumer locality are the rows A and C. They say which gate blocks the request and how downstream consumers may inspect the blocked route without replacing it by an internal theorem.

Provenance and local naming are supplied by P and N. Since the carrier inventory contains no row for the requested supply itself, the NameCert obligations expose the socket while preserving ledger non-escape. □

Theorem 528.3 (SupplySocketLedger kind exhaustion)**.** *Every accepted* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row classifies its socket as exactly one of the seven displayed supply kinds: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure. The ledger exports no eighth hidden kind, no unlisted classifier branch, and no catch-all route that can discharge a* GAP *tag without passing through one of these seven rows.*

Proof. Project the accepted carrier $L = (G, K, S, A, C, P, N)$. The only kind coordinate is K, and Definition 528.1 restricts that coordinate to the seven named supply kinds.

Apply the NameCert obligations of Definition 528.2. They make kind exactness a certificate row rather than a prose convention, so consumer access to the socket classifier must pass through the displayed finite list.

The absence claim follows from the carrier inventory. No coordinate names an additional kind, a default classifier branch, or a discharge route outside the seven rows. □

Theorem 528.4 (SupplySocketLedger audit non-escape)**.** *The* $\mathsf{SupplySocketLedger}^{\uparrow}$ *packet records refusal of* Classical.choice, Quot.sound, propext, *host equality, unbounded induction, and truth-total self-description as explicit socket rows rather than as substrate theorems. Each refusal is visible only through its* GAP *tag, supply-kind classifier, socket site, audit gate, continuation route, package provenance, and local NameCert row.*

Proof. Read each refusal through the carrier coordinates. The GAP row names the boundary, the kind row classifies the requested supply, the site row says where the request would be consumed, and the audit row names the gate that keeps the request external.

Continuation and provenance then preserve the same boundary. The Cont routes let consumers inspect the refusal, while P records that the route belongs to the finite ledger over BHist and local naming data.

No substrate theorem is produced by these rows. The items Classical.choice, Quot.sound, propext, host equality, unbounded induction, and truth-total self-description are recorded as refused socket requests, and no coordinate in L turns them into available BEDC principles. □

Theorem 528.5 (SupplySocketLedger TasteGate carrier boundary)**.** *The local* ChapterTasteGate-*relevant carrier boundary for* $\mathsf{SupplySocketLedger}^{\uparrow}$ *is exactly the displayed finite packet*

$$L = (G, K, S, A, C, P, N).$$

It exposes the GAP *tag row, supply-kind row, socket site, audit gate,* Cont *consumption routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row, and it contains no coordinate carrying the refused supply itself.*

Proof. Project the carrier of Definition 528.1. Its complete displayed inventory is the seven-tuple $L = (G, K, S, A, C, P, N)$, so the TasteGate-relevant boundary has precisely those coordinates.

Apply Definition 528.2. The NameCert surface reads carrier admission, classifier exposure, audit routing, continuation locality, package provenance, local naming, and ledger non-escape from the same seven rows.

Finally use Definition 528.4. Refused supplies are represented as socket requests behind the displayed audit rows, not as coordinates of L. □

Theorem 528.6 (SupplySocketLedger route coverage)**.** *Every consumer route that reads a* $\mathsf{SupplySocketLedger}^{\uparrow}$ *supply-kind row factors through the displayed* GAP *tag G, socket site S, audit gate A, and* Cont *routes C. No consumer can reach the classifier row K through an unnamed route, a hidden eighth kind, or a route that bypasses the audit gate.*

Proof. Start with an accepted packet $L = (G, K, S, A, C, P, N)$. By Definition 528.1, the only classifier coordinate is K, and the carrier displays the boundary tag, socket site, audit gate, and continuation rows beside it.

Apply Definition 528.3. Any read of K lands in one of the seven displayed supply kinds, with no hidden classifier branch available to the consumer.

The route factorisation then follows from Definition 528.2: audit-gate routing is through A and continuation locality is through C. The non-escape result of Definition 528.4 rules out replacing that factored route by the refused supply itself. □

Theorem 528.7 (SupplySocketLedger consumer provenance)**.** *Downstream concrete-instance chapters may cite a* $\mathsf{SupplySocketLedger}^{\uparrow}$ *packet only through the displayed* Pkg *provenance row P and the local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row N, together with the visible classifier, socket, audit, and continuation rows that they certify. They may not cite the requested supply as a produced theorem, selected witness, quotient equality, persistence law, shared host history, inscription source, or truth-total self-description.*

Proof. Project the provenance and naming rows from Definition 528.1. The rows P and N are the only coordinates whose role is to make the ledger citeable by later concrete-instance chapters.

Use Definition 528.2 to identify what those rows certify: the finite carrier, the supply-kind classifier, audit-gate routing, continuation locality, package provenance, local naming, and ledger non-escape.

The forbidden citations have no carrier coordinate. By Definition 528.4, they remain refused socket requests visible through the ledger surface, so a downstream chapter may cite the provenance and NameCert rows but not the missing supply. □

Theorem 528.8 (SupplySocketLedger kind-route determinacy)**.** *For an accepted* $\mathsf{SupplySocketLedger}^{\uparrow}$ *packet $L = (G, K, S, A, C, P, N)$, the* BHist *row K determines the visible socket-kind route consumed by any* Cont *read of the ledger. The route is read with the socket-site row S, audit row A,* Pkg

provenance row P, and local $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row N; no second route can classify the same accepted packet as a different supply kind.*

Proof. Project the carrier of Definition 528.1. The packet has one kind coordinate K, one socket-site row S, one audit row A, one family of consumption routes C, one provenance row P, and one local naming row N.

Apply Definition 528.3 to the projected kind row. Since K ranges over the seven displayed supply kinds and the carrier has no further classifier coordinate, a consumer route cannot obtain a different kind without leaving the packet.

Finally use Definition 528.6. Every classifier read factors through the displayed Cont routes and audit gate, so the route determined by K is the only route certified by P and N for the accepted row. □

Theorem 528.9 (SupplySocketLedger refused-row non-escape)**.** *If a* BHist *supply row is outside the displayed* $\mathsf{SupplySocketLedger}^{\uparrow}$ *packet $L = (G, K, S, A, C, P, N)$, then no consumer can read it through the local ledger. The only readable refusal data are the displayed G, K, S, A rows, their* Cont *consumption routes C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row N.*

Proof. Start from the inventory in Definition 528.1. A row outside G, K, S, A, C, P, N is absent from the finite packet and therefore is not a coordinate that a local consumer can project.

Apply Definition 528.4. The refused principles and host supplies remain recorded as socket requests behind the displayed audit row; they are not converted into substrate theorems or available carrier fields.

The NameCert obligations then preserve the boundary. By Definition 528.2, provenance and local naming certify exactly the displayed refusal rows and their Cont routes, so an outside supply row has no certified path into the ledger. □

Theorem 528.10 (SupplySocketLedger consumption coverage)**.** *Every accepted* $\mathsf{SupplySocketLedger}^{\uparrow}$ *socket route consumed by a downstream chapter is covered by the displayed* BHist *rows of the same packet: the* GAP *tag G, socket-kind row K, socket-site row S, audit row A,* Cont *routes C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$ *row N. No accepted route may cite the ledger while omitting the local NameCert row or bypassing provenance.*

Lean4 ✓ `BEDC.Derived.SupplySocketLedgerUp.SupplySocketLedgerUp StdBridge`

Proof. Project an accepted packet $L = (G, K, S, A, C, P, N)$. The displayed carrier gives the complete list of rows from which a consumer may read a socket route.

Use Definition 528.6 for reads of the classifier and Definition 528.7 for downstream citations. Together they force each accepted consumption route to pass through C, cite P, and retain the local naming row N.

It remains to rule out partial consumption. The non-escape result of Definition 528.4 says that bypassing the audit or provenance rows would replace the displayed refusal by the refused supply itself, which is not a coordinate of the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SupplySocketLedger}^{\uparrow}$ packet is generated from one GAP tag row, one of the seven supply-kind classifiers, one displayed socket site, one audit-gate row, displayed Cont consumption routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SupplySocketLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 528.1, Definition 528.2, Definition 528.3, Definition 528.4, Definition 528.5 with

Lean4 ✓ `BEDC.Derived.SupplySocketLedgerUp.taste gate`

, Definition 528.6, Definition 528.7, and Definition 528.10 bind the finite carrier, seven-kind classifier, audit-gate routing, continuation locality, package provenance, consumer provenance, bridge-facing consumer route, and explicit non-escape rows.

Formal verification: FormalStatus(NameCert$_{\text{SupplySocketLedger}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.SupplySocketLedgerUp.SupplySocketLedgerTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close public export, standard bridge data, or any internal theorem supplying the refused socket contents.

Upgrade path: No further formal bridge-axis upgrade remains for the seven-kind supply-socket ledger.

Chapter 529

A Concrete Naming Certificate for $\mathsf{RecursorAuthorization}^{\uparrow}$

The $\mathsf{RecursorAuthorization}^{\uparrow}$ interface records the finite certificate surface by which an accepted Lean `inductive` declaration authorizes its emitted recursor as a BEDC generator. The packet is built from the accepted constructor signature, the emitted eliminator, the branch package supplied to the eliminator, the structural descent ledger for recursive occurrences, and the NameCert row naming the generated output. It is a naming certificate for authorized generator use; it is not a host-side trust principle for arbitrary recursion.

The $\mathsf{RecursorAuthorization}^{\uparrow}$ row is the signature-and-recursion authorization row consumed by chapter 525 and refined by chapter 534. The first sibling supplies the audit boundary for host-kernel accepted recursor calls, while the second packages legal recursor calls and closed-term handoffs.

Definition 529.1 (RecursorAuthorization carrier)**.** *A* $\mathsf{RecursorAuthorization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, \Sigma, R, M, B, D, H, C, P, N)$$

with the following visible rows:

1. *I is the accepted inductive name and Σ is its closed constructor signature, including the constructor rows for instances such as* `BMark` *and* `BHist`*;*
2. *R is the emitted eliminator row, read as the kernel-generated recursor for Σ, including concrete rows such as* `BMark.rec` *and* `BHist.rec`*;*
3. *M is the motive row accepted for the recursor call;*
4. *B is the finite branch package, with one branch row for each constructor of Σ and branch completeness checked against that signature;*
5. *D is the structural descent ledger recording that recursive calls occur only on constructor-subdata accepted by the kernel;*
6. *H stores componentwise* hsame *transports for constructor signatures, eliminator rows, motive rows, branch rows, descent rows, and generated outputs;*
7. *C stores displayed* Cont *routes by which consumers read the constructor, recursor, branch, and output rows;*
8. *P is the* Pkg *provenance over* `BMark`*,* BHist*, closed constructor signatures, emitted recursors, branch packages,* hsame*,* Cont*, and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RecursorAuthorization}^{\uparrow}}$ *row naming the generated output.*

The carrier contains no unlisted constructor, no extra branch, no unrestricted recursion channel, and no external proof-trust coordinate.

Theorem 529.2 (RecursorAuthorization NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RecursorAuthorization}^{\uparrow}}$ *obligation surface consists of the accepted constructor signature Σ, the emitted recursor row R, the motive row M, the branch package B, the structural descent ledger D, and the displayed* hsame, Cont, Pkg, *and naming rows H, C, P, N. Every authorized generator output must be read through these rows before it is available to a downstream BEDC consumer.*

Proof. Project the carrier of Definition 529.1. The constructor signature, emitted recursor, motive, branch package, descent ledger, transport rows, continuation routes, provenance, and local naming row are all coordinates of the same finite packet.

The obligation rows are therefore carrier projections. A consumer cannot read the generated output until it has passed through the signature row, the recursor row, the accepted motive and branch data, and the descent ledger.

Since the carrier has no coordinate for an external trust principle or unrestricted recursion channel, the local NameCert surface is exactly the displayed recursor-authorization packet. □

Theorem 529.3 (RecursorAuthorization branch exhaustion)**.** *For every accepted* $\mathsf{RecursorAuthorization}^{\uparrow}$ *packet $A = (I, \Sigma, R, M, B, D, H, C, P, N)$, the branch package B contains exactly the branch rows indexed by the constructors in Σ. A recursor consumer has no route to an omitted constructor case and no route to an additional branch not named by the accepted constructor signature.*

Lean4 ✓ `BEDC.Derived.RecursorAuthorizationUp.RecursorAuthorizationBranchExhaustion`

Proof. Read the constructor signature row Σ from the carrier. It is the finite list of constructors accepted for I.

Read the branch package B through the emitted recursor row R. The package is attached to the same signature row, so its branch inventory is indexed by the visible constructors of Σ.

The transport and continuation rows H and C preserve that same indexing for downstream reads. Because the carrier lists no further branch coordinate, branch exhaustion is exactly the displayed finite constructor coverage. □

Theorem 529.4 (RecursorAuthorization generator non-escape)**.** *Every generated output of an accepted* $\mathsf{RecursorAuthorization}^{\uparrow}$ *packet factors through the emitted recursor R, the accepted motive M, the branch package B, and the structural descent ledger D. The packet exports no hidden generator, no non-structural recursive call, no unlisted constructor, and no external recursion principle.*

Proof. Start with the local obligation surface of Definition 529.2. It places each generated output downstream of the accepted signature, recursor, motive, branch, and descent rows.

Use branch exhaustion from Definition 529.3. Every case consumed by the recursor is one of the constructor cases in Σ, and every recursive occurrence is governed by the same descent ledger D.

The remaining rows H, C, P, N only transport, route, package, and name that finite surface. They do not add a generator coordinate, so no output can escape the accepted recursor packet. □

Theorem 529.5 (RecursorAuthorization public handoff)**.** *The public* $\mathsf{NameCert}_{\mathsf{RecursorAuthorization}^{\uparrow}}$ *handoff of an accepted carrier is exactly the tuple of constructor signature, emitted recursor, accepted motive, complete branch package, structural descent ledger, transport rows, continuation*

routes, package provenance, and local naming row displayed in Definition 529.1. It can be cited by downstream generator-closure, TasteGate, and self-compilation consumers without exporting unrestricted recursion, hidden constructor data, or host-side proof trust.

Lean4 ✓ `BEDC.Derived.RecursorAuthorizationUp.RecursorAuthorizationTasteGate single carrier alignment`

Proof. First assemble the local authorization surface using Definition 529.2. This gives the carrier rows consumed by any recursor-generated output.

Next apply Definition 529.3 and Definition 529.4. These isolate the finite branch coverage and rule out generator routes outside the accepted signature and descent ledger.

Finally repack the displayed rows with H, C, P, N. The public handoff is the same finite packet under its local NameCert row, and no additional recursion or proof-trust coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorAuthorization}\uparrow}$.

Constructive story: A $\mathsf{RecursorAuthorization}^{\uparrow}$ packet is generated from an accepted inductive declaration row, constructor signature row, emitted recursor row, motive row, complete branch package, structural descent ledger, componentwise transport, replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursorAuthorization}\uparrow}, \mathsf{seed})$

Scope closed: Definition 529.1, Definition 529.2, Definition 529.3, Definition 529.4, and Definition 529.5 bind the accepted constructor signature, emitted recursor, motive, branch package, structural descent ledger, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursorAuthorization}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close unrestricted recursion, hidden constructors, arbitrary generator authorization, non-structural recursive calls, or host-side proof trust.

Upgrade path: Obligation closure requires the displayed branch exhaustion, generator non-escape, public handoff, and structural descent rows to be assembled without changing the formal axis.

Chapter 530

A Concrete Naming Certificate for $\mathsf{UnaryDirectionBridge}^{\uparrow}$

The $\mathsf{UnaryDirectionBridge}^{\uparrow}$ interface packages the finite bridge by which the unary $\mathsf{Nat}^{\uparrow}$ spine and the zero-spine $\mathsf{AxisNat}^{\uparrow}$ surface can be compared without identifying their kernel carriers. Its local certificate records the two source rows, the bridge witness, the one-point kernel intersection, standard-boundary interpretation, ledger policy, and public naming row. The packet is a comparison certificate over displayed BEDC rows; it is not a kernel equality between $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$.

The dependency anchors are the $\mathsf{Nat}^{\uparrow}$ source carrier of chapter 28 and the $\mathsf{AxisNat}^{\uparrow}$ source carrier of chapter 276, kept as distinct BEDC rows throughout the bridge. The $\mathsf{Add}^{\uparrow}$ and $\mathsf{AxisAdd}^{\uparrow}$ surfaces of chapter 29 and chapter 277 enter only at the standard-boundary consumer row, together with BHist, hsame, Cont, Pkg, NameCert, and the Lean-side bridge object UnaryDirectionBridge.

Definition 530.1 (UnaryDirectionBridge NameCert carrier)**.** *A* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, A, B, K, S, L, H, C, P, M)$$

with the following visible rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *unary-spine source row, read through the* Unary *predicate over* E1 *histories;*
2. *A is the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source row, read through the* ZeroSpine *predicate over* E0 *histories;*
3. *B is the local bridge row naming* UnaryDirectionBridge *and its two displayed source carriers;*
4. *K is the kernel-distinctness ledger asserting that the unary and zero-spine carriers intersect only at* e*;*
5. *S is the standard-boundary row, matching nonempty unary histories with nonempty zero-spine histories by length and preserving the* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumer route at the standard boundary;*
6. *L is the ledger policy row, recording that bridge reads must cite N, A, B, K, S before exporting any comparison;*
7. *H stores componentwise* hsame *transports for the two source rows, bridge row, kernel-distinctness witness, standard-boundary row, ledger policy row, and local naming row;*
8. *C stores displayed* Cont *routes for unary inspection, zero-spine inspection, bridge comparison, standard-boundary readback, ledger audit, and NameCert comparison;*

9. *P is the* Pkg *provenance row naming* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and* UnaryDirectionBridge*;*
10. *M is the local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ *row.*

The classifier compares accepted bridge packets only after transporting these rows componentwise. It has no coordinate for a kernel equality between $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*, no quotient identifying unary and zero-spine histories, and no host-level natural-number equality beyond the displayed standard-boundary row.*

Theorem 530.2 (UnaryDirectionBridge kernel-distinct obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ *obligation surface must expose the* $\mathsf{Nat}^{\uparrow}$ *unary source row N, the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source row A, the bridge row B, and the kernel-distinctness ledger K. Every accepted bridge comparison first reads the two source classifiers, then cites K to rule out a kernel-level carrier collapse except at* e*.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier kernel distinct obligations`

Proof. Project the carrier of Definition 530.1. The source rows N and A are separate coordinates, so a comparison cannot begin from a single shared carrier.

The bridge row B names the comparison object, while K records the only permitted kernel intersection. Reading K before the standard-boundary row forces the comparison to treat unary and zero-spine histories as distinct BEDC predicates.

The transport rows H and continuation routes C preserve this separation for downstream consumers. Since the carrier contains no equality coordinate identifying N and A, the obligations close exactly as a kernel-distinct bridge surface. □

Theorem 530.3 (UnaryDirectionBridge standard boundary)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier $U = (N, A, B, K, S, L, H, C, P, M)$, the public standard boundary is the displayed row S: nonempty unary histories are matched with nonempty zero-spine histories by the same finite length, and the* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumer route is read only after the source rows and the kernel-distinctness ledger have been transported through H and C.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier standard boundary readout`

Proof. Begin with an accepted bridge packet and project the standard-boundary row S. Its inputs are the two already displayed source rows, not an ambient natural-number object.

The row K is read before S, so the boundary comparison is licensed only after the paper has recorded the one-point kernel intersection. The transports in H and the routes in C then carry the unary and zero-spine rows to the same public comparison site.

The additive consumers $\mathsf{Add}^{\uparrow}$ and $\mathsf{AxisAdd}^{\uparrow}$ are reached through this transported boundary. Because no carrier coordinate identifies the two source predicates, the standard boundary is a public interpretation row rather than a kernel collapse. □

Theorem 530.4 (UnaryDirectionBridge ledger policy)**.** *Every consumer-facing route out of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet factors through the displayed rows $N, A, B, K, S, L, H, C, P, M$. The ledger policy row L rejects any route that compares* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$ *by host equality, by quotienting histories, or by skipping the kernel-distinctness witness.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier ledger policy`

Proof. Take any consumer-facing route and inspect the finite packet. The available source, bridge, kernel, standard-boundary, policy, transport, continuation, provenance, and naming rows are exactly $N, A, B, K, S, L, H, C, P, M$.

The policy row L orders the read: source rows first, bridge row second, kernel-distinctness ledger third, and standard-boundary comparison only after those rows are present. The continuation routes in C expose no alternate branch.

Thus the route is exhausted by the displayed ledger. Since the carrier has no coordinate for host equality or quotient identification of histories, neither can be exported by a bridge consumer. □

Theorem 530.5 (UnaryDirectionBridge NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ *surface is the finite bridge packet of Definition 530.1 together with the kernel-distinctness obligations of Definition 530.2, the standard boundary of Definition 530.3, and the ledger policy of Definition 530.4. It realizes the unary-direction bridge as a NameCert packet while keeping* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$ *distinct inside the kernel.*

Proof. Assemble the carrier rows $N, A, B, K, S, L, H, C, P, M$. These rows provide the two source certificates, the bridge object, kernel-distinctness witness, standard-boundary readback, policy ledger, transports, routes, provenance, and local name.

Apply Definition 530.2 to bind the two source rows before any comparison. Apply Definition 530.3 to expose the length-preserving public boundary. Apply Definition 530.4 to rule out routes that bypass the displayed rows.

Repacking these rows gives the local $\mathsf{NameCert}$ surface. The packet therefore realizes the bridge at the paper-certificate level and leaves the kernel carriers separate. □

Theorem 530.6 (UnaryDirectionBridge source obligation inventory)**.** *Every accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet exposes exactly the source rows needed before any bridge comparison: the* $\mathsf{Nat}^{\uparrow}$ *unary-spine row* N*, the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine row* A*, the* $\mathsf{Add}^{\uparrow}$ *and* $\mathsf{AxisAdd}^{\uparrow}$ *standard-boundary consumer rows read through* S*, the carrier history row* BHist*, componentwise* hsame *transport through* H*, displayed* Cont *routes through* C*, package provenance through* P*, and the local* $\mathsf{NameCert}$ *row* M*.*

Proof. Project the carrier of Definition 530.1. The source coordinates are N and A, while S is the only displayed row through which $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumers may read the public boundary.

Use Definition 530.2 to keep the source rows separate before the boundary is inspected. Then use Definition 530.3 and Definition 530.4 to account for the boundary, route, and policy rows.

Finally, Definition 530.5 repacks the same inventory as the local certificate. Since the carrier has no further coordinate, the listed rows exhaust the source obligation surface. □

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier source obligation inventory`

Theorem 530.7 (UnaryDirectionBridge kernel-boundary obligation)**.** *Any* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *read that reaches the standard-boundary row* S *must first pass through the kernel-distinctness ledger* K*. Thus a consumer cannot interpret the displayed* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *boundary until the unary* $\mathsf{Nat}^{\uparrow}$ *row, the zero-spine* $\mathsf{AxisNat}^{\uparrow}$ *row, and the one-point kernel-intersection policy have been recorded in the same packet.*

Proof. Start from the kernel-distinctness obligations. By Definition 530.2, the rows N, A, B, K are read before any comparison is exported.

The standard boundary of Definition 530.3 is therefore downstream of K. It reads nonempty unary and zero-spine histories through the displayed boundary, but it does not identify the two kernel carriers.

The ledger policy of Definition 530.4 rules out a route that skips K. Repacking through Definition 530.5 gives the required boundary obligation for every accepted packet. □

Theorem 530.8 (UnaryDirectionBridge standard handoff obligation)**.** $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumers of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet read only the displayed standard-boundary row S, the componentwise transport rows H, the continuation routes C, package provenance P, the local* $\mathsf{NameCert}$ *row M, and the source and ledger rows required by N, A, B, K, L. They receive no host natural-number equality, quotient of histories, or carrier identity between* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*.*

Proof. First expose the public boundary. Definition 530.3 places every $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ handoff on S after the source rows and the kernel ledger have been transported.

Next apply the ledger policy. Definition 530.4 requires the route to cite $N, A, B, K, S, L, H, C, P, M$ before it can export a comparison to additive consumers.

Finally use the NameCert surface. Definition 530.5 packages exactly those rows and keeps $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$ distinct inside the kernel, so the standard handoff cannot export host equality, quotient identification, or carrier identity. □

Theorem 530.9 (UnaryDirectionBridge AxisAdd handoff obligations)**.** *Every* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumer using the standard-boundary handoff of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet must first read the rows $N, A, B, K, S, L, H, C, P, M$. The handoff is licensed only after the source rows, bridge row, kernel-distinctness ledger, standard-boundary row, policy row, transport row, continuation row, provenance row, and local* $\mathsf{NameCert}$ *row are all present.*

Proof. Start from the carrier of Definition 530.1. Its displayed rows are exactly $N, A, B, K, S, L, H, C, P, M$, so a consumer route has no additional coordinate from which to read a boundary.

Apply Definition 530.2 before the public boundary is used. This binds the source rows and the kernel-distinctness ledger.

Apply Definition 530.3 and then Definition 530.4. The first theorem supplies the length-preserving handoff row, and the second requires the policy and route rows before export.

Finally, use Definition 530.5 to repack the read rows as the local certificate. Thus $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumers reach the handoff only through the displayed packet. □

Theorem 530.10 (UnaryDirectionBridge empty-intersection readback)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet, the only kernel intersection exposed by the bridge is the empty-history row named by K. No consumer-facing readback from B, S, L, H, C, P, or M exports a second intersection between the* $\mathsf{Nat}^{\uparrow}$ *unary source row and the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source row.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier empty intersection readback`

Proof. Read the kernel-distinctness obligations. By Definition 530.2, the bridge must cite K before it compares the source rows.

The standard-boundary theorem Definition 530.3 then interprets nonempty rows only at the public boundary. It does not insert a kernel equality between the two predicates.

The ledger-policy theorem Definition 530.4 rules out host equality, quotienting, and routes that skip K. Together with Definition 530.5, this leaves the empty-history row named by K as the sole exposed kernel intersection. □

Theorem 530.11 (UnaryDirectionBridge consumer normal form)**.** *Every consumer route out of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet normalizes to the source rows N, A, the bridge row B, the*

kernel-distinctness ledger K, the standard-boundary row S, and the local NameCert *row M, with transport, continuation, provenance, and policy reads occurring only as displayed rows of the same packet. No normalized route contains quotient identification or host-level natural-number equality.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier consumer normal form`

Proof. Inspect the carrier rows given by Definition 530.1. A consumer can mention only the displayed source, bridge, kernel, standard-boundary, policy, transport, route, provenance, and naming rows.

Apply Definition 530.4. It orders every consumer-facing route through the source rows, the bridge row, the kernel-distinctness ledger, and the standard-boundary row.

Apply Definition 530.3 for the public handoff and Definition 530.5 for the local name. The resulting normal form is therefore N, A, B, K, S, M with L, H, C, P as the displayed support rows, and it has no quotient or host natural-number equality coordinate. □

Theorem 530.12 (UnaryDirectionBridge carrier admission)**.** *For the finite packet $U = (N, A, B, K, S, L, H, C, P, M$ over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *admission to the* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *surface is exactly admission of the ten displayed rows of Definition 530.1. The admitted packet exposes the unary source row, zero-spine source row, bridge row, kernel-distinctness ledger, standard-boundary row, ledger policy, componentwise transports, continuation routes, package provenance, and local naming row, and it exposes no additional carrier coordinate.*

Proof. Start with the carrier description of Definition 530.1. It lists $N, A, B, K, S, L, H, C, P, M$ as the complete finite packet over the named source, bridge, history, transport, continuation, package, and naming surfaces.

Apply the kernel-distinct obligations of Definition 530.2 and the standard boundary of Definition 530.3. These identify the source rows, the bridge row, the kernel ledger, and the public boundary as displayed rows of the same packet.

Use the ledger policy of Definition 530.4, the NameCert surface of Definition 530.5, and the AxisAdd handoff of Definition 530.9. Together they force every admitted read through the packet rows and leave no extra carrier coordinate.

Finally, the empty-intersection readback of Definition 530.10 and the consumer normal form of Definition 530.11 exclude host equality, quotient identification, and a second kernel carrier. Thus admission is exactly the finite ledger packet. □

Theorem 530.13 (UnaryDirectionBridge classifier stability)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet $U = (N, A, B, K, S, L, H, C, P, M)$, componentwise* hsame *transport by the stored row H preserves the unary source classifier, the zero-spine source classifier, the bridge classifier, the kernel-distinctness ledger, the standard-boundary route, the ledger policy, the continuation routes, the package provenance, and the local* NameCert *row. After transport, the packet still presents the same kernel-distinctness ledger and the same standard-boundary route.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier classifier stability`

Proof. Project H from the carrier of Definition 530.1. By construction it stores componentwise transports for the two source rows, bridge row, kernel-distinctness witness, standard-boundary row, ledger policy row, and local naming row.

Apply Definition 530.2. Since K remains a transported row rather than an equality coordinate, the transported classifiers cannot collapse the $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$ carriers.

Use Definition 530.3 and Definition 530.9 for the boundary route. The transported standard-boundary row is the only route available to $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumers.

The ledger policy, NameCert surface, empty-intersection readback, and consumer normal form of Definition 530.4, Definition 530.5, Definition 530.10, and Definition 530.11 then keep every classifier read inside the displayed packet. Hence componentwise transport preserves both the kernel-distinctness ledger and the standard-boundary route. □

Theorem 530.14 (UnaryDirectionBridge ledger exactness)**.** *Every consumer route out of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet is exhausted by* $N, A, B, K, S, L, H, C, P, M$*. There is no consumer route through host equality of natural numbers, no quotient identification of histories, and no collapse of the* $\mathsf{Nat}^{\uparrow}$ *unary carrier with the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine carrier.*

Proof. Begin with the finite carrier in Definition 530.1. Its rows are exactly $N, A, B, K, S, L, H, C, P, M$, so any consumer route must be assembled from these rows.

Apply the kernel-distinct obligations and standard boundary from Definition 530.2 and Definition 530.3. They require the source rows and K before any public boundary read.

Next use the ledger policy, NameCert surface, and AxisAdd handoff of Definition 530.4, Definition 530.5, and Definition 530.9. These force consumer access through the displayed ledger and then through the local naming row.

Finally apply the empty-intersection readback of Definition 530.10 and the consumer normal form of Definition 530.11. They rule out any second intersection, quotient route, or host equality coordinate, so the displayed finite ledger is exact. □

Theorem 530.15 (UnaryDirectionBridge source-classifier transport)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier* $U = (N, A, B, K, S, L, H, C, P, M)$*, componentwise* hsame *transport of* $N, A, B, K, S, L, H, C, P, M$ *preserves the separate* $\mathsf{Nat}^{\uparrow}$ *unary source classifier and the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source classifier. The transported packet keeps the two source classifiers as distinct rows and does not collapse them into a kernel equality.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier source classifier transport`

Proof. Project the carrier of Definition 530.1. Rows N and A are separate source rows, while H supplies componentwise transport for those rows together with the bridge, ledger, route, provenance, and local name.

Apply Definition 530.2. The kernel-distinctness row K must be read before any comparison, so transport cannot replace the two classifiers by a shared kernel carrier.

Use Definition 530.10 to keep the empty-history row as the only exposed kernel intersection. Thus the transported source classifiers remain the unary $\mathsf{Nat}^{\uparrow}$ row and the zero-spine $\mathsf{AxisNat}^{\uparrow}$ row separately. □

Theorem 530.16 (UnaryDirectionBridge additive consumer exactness)**.** *Every* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumer of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier reads the standard-boundary row only after the source rows, bridge row, kernel-distinctness ledger, ledger policy, transports, routes, provenance, and local* NameCert *row are present. The consumer receives the rows* $N, A, B, K, S, L, H, C, P, M$ *in that order of dependence and receives no kernel equality between* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier additive consumer exactness`

Proof. Start with the consumer normal form of Definition 530.11. It limits every consumer route to the source rows, bridge row, kernel-distinctness ledger, standard-boundary row, local name, and the displayed support rows.

Apply Definition 530.9. This requires $N, A, B, K, S, L, H, C, P, M$ to be present before the $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ handoff is licensed.

The standard-boundary theorem Definition 530.3 supplies the public boundary only after those rows have been transported. The ledger policy of Definition 530.4 rules out any consumer route that skips the source rows or exports a host equality. □

Theorem 530.17 (UnaryDirectionBridge NameCert obligation closure)**.** *The carrier, kernel-distinctness obligations, standard boundary, ledger policy, consumer normal form, source-classifier transport, and additive consumer exactness assemble the local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ *obligation closure for* $U = (N, A, B, K, S, L, H, C, P, M)$*. The closure keeps the unary and zero-spine kernels distinct while presenting the finite bridge packet required by downstream additive consumers.*

Proof. The carrier of Definition 530.1 supplies the finite packet. The kernel-distinctness obligations, standard boundary, ledger policy, and NameCert surface of Definition 530.2, Definition 530.3, Definition 530.4, and Definition 530.5 bind the basic certificate rows.

Add the handoff obligations and empty-intersection readback from Definition 530.9 and Definition 530.10. These keep additive consumers on the displayed standard boundary and prevent a second kernel intersection from appearing.

Finally apply Definition 530.11, Definition 530.15, and Definition 530.16. The resulting local NameCert obligation closure is exactly the displayed bridge packet, with no asserted kernel equality. □

Theorem 530.18 (UnaryDirectionBridge scoped source bindings)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier* $U = (N, A, B, K, S, L, H, C, P, M)$*, the rows* $N, A, B, K, S, L, H, C, P, M$ *bind the bridge packet to* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{AxisAdd}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The binding is scoped to the displayed carrier rows:* N *and* A *are the two source classifiers,* B, K, S, L *are the bridge, kernel-distinctness, boundary, and policy rows,* H *and* C *are the transport and continuation rows,* P *is the package provenance row, and* M *is the local naming row.*

Proof. Project the carrier of Definition 530.1. The row list already names the two source surfaces, the bridge object, the kernel ledger, the boundary, the policy, the transport, the route, the package row, and the local $\mathsf{NameCert}$ row.

Apply Definition 530.2. This keeps N and A as separate source classifiers and forces every comparison to cite K before a public boundary is read.

Use Definition 530.3 for the S row and Definition 530.4 for the L row. The transport and route rows H, C carry these displayed rows without creating a host equality or quotient coordinate.

Finally apply Definition 530.5. The provenance row P and local name M therefore bind the packet to the listed BEDC dependencies as scoped rows of the same finite carrier. □

Theorem 530.19 (UnaryDirectionBridge additive route scope)**.** *Every* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *handoff route out of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet factors through* K, S, H, C, P, M *together with the source rows* N, A *and the bridge row* B*. Such a route cannot use host equality of natural numbers, quotient identification of histories, or a second kernel carrier for the unary and zero-spine sources.*

Proof. Start with the additive consumer exactness of Definition 530.16. It says that an $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumer receives the displayed source, bridge, kernel, policy, transport, route, provenance, and local naming rows before the boundary is used.

Apply the source-classifier transport theorem Definition 530.15. The transported packet still has separate $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$ source rows, so the additive route cannot be reinterpreted as a single kernel carrier.

Use the empty-intersection readback of Definition 530.10 and the consumer normal form of Definition 530.11. These exclude a second kernel intersection, quotient identification, and host equality from the normalized route.

The remaining route is therefore exactly the displayed additive handoff through K, S, H, C, P, M, with N, A, B supplying the source and bridge context. □

Theorem 530.20 (UnaryDirectionBridge scoped NameCert package)**.** *The scoped source bindings of Definition 530.18, the additive route scope of Definition 530.19, and the obligation closure of Definition 530.17 assemble the scoped* $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ *package. The package exposes the carrier rows, the obligation closure, the scoped BEDC dependency bindings, and the additive handoff route, while preserving the separation of* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$ *inside the kernel.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier namecert scoped package`

Proof. Begin with Definition 530.17. It supplies the local closure of the carrier, kernel-distinctness obligations, standard boundary, ledger policy, consumer normal form, source-classifier transport, and additive consumer exactness.

Apply Definition 530.18. This binds the same finite packet to $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ through displayed carrier rows.

Apply Definition 530.19. The additive route is then confined to the scoped bridge packet and cannot pass through host equality, quotient identification, or a second kernel carrier.

Repacking these rows gives the scoped local $\mathsf{NameCert}$ package for $\mathsf{UnaryDirectionBridge}^{\uparrow}$. Its public commitment is the displayed BEDC bridge packet, not an identification of the two source kernels. □

Theorem 530.21 (UnaryDirectionBridge kernel-axis composition)**.** *For an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *carrier* $U = (N, A, B, K, S, L, H, C, P, M)$*, the kernel-distinct obligations, standard boundary, and* $\mathsf{AxisAdd}^{\uparrow}$ *handoff assemble one* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *route from the unary direction row* N *to the additive consumer row. The route reads* $A, B, K, S, L, H, C, P, M$ *as displayed support and never identifies the unary and zero-spine classifiers as one kernel carrier.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier kernel axis composition`

Proof. Project the carrier and begin with the kernel-distinct obligations of Definition 530.2. They force the route to read the $\mathsf{Nat}^{\uparrow}$ row N, the $\mathsf{AxisNat}^{\uparrow}$ row A, the bridge row B, and the kernel ledger K before any additive handoff is considered.

Next apply the standard boundary of Definition 530.3. The boundary row S is reached only after the source rows and K have been transported through H and replayed through C.

Finally use Definition 530.9 and Definition 530.16. These results place the $\mathsf{AxisAdd}^{\uparrow}$ consumer behind the same displayed packet and repack the route through L, P, M as policy, provenance, and local naming data. The resulting composition is a projection-and-route composition, not a carrier equivalence. □

Theorem 530.22 (UnaryDirectionBridge bidirectional readback)**.** *For accepted left and right direction reads of a* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet, if their displayed* hsame *rows and* Cont *rows agree, then both reads transport to the same empty-intersection readback surface. The common surface is the* K*-licensed intersection row together with the standard boundary and local NameCert rows, not an equivalence between the source classifiers.*

Proof. Start with the two direction reads and project their carrier rows. Agreement of the displayed hsame and Cont rows means both reads use the same transport H and the same route family C.

Apply Definition 530.10. The only kernel intersection exposed by either read is the empty-history row licensed by K.

Use Definition 530.15 to keep the unary and zero-spine classifiers separate after transport. Then apply Definition 530.3 and Definition 530.5 to place both reads on the same public boundary and local certificate surface.

Thus the left and right reads agree at the empty-intersection readback surface while preserving the source-classifier separation. □

Theorem 530.23 (UnaryDirectionBridge downstream seal)**.** *Every downstream* $\mathsf{AxisAdd}^{\uparrow}$ *or* $\mathsf{AxisZeckendorfCannot}$ *consumer reads an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet only through the carrier, kernel-distinct classifier rows, ledger policy, source-classifier transport, and additive consumer exactness rows. The downstream seal exports the bridge as the displayed packet* $U = (N, A, B, K, S, L, H, C, P, M)$ *and exports no quotient identification, host natural-number equality, or carrier equivalence between* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*.*

Proof. Begin with carrier admission and ledger policy, Definition 530.12 and Definition 530.4. They restrict every downstream read to the ten displayed rows and order the read through the source, bridge, kernel, boundary, policy, transport, route, provenance, and naming data.

For the classifier side, apply Definition 530.2, Definition 530.13, and Definition 530.15. These keep N and A as separate source classifiers under componentwise transport.

For the consumer side, use Definition 530.16 and Definition 530.21. The $\mathsf{AxisAdd}^{\uparrow}$ route and any $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ consumer that depends on it can therefore read only the displayed bridge packet.

The empty-intersection readback excludes any second kernel overlap, and the consumer normal form excludes host equality and quotient identification. Hence the downstream seal is the packet route itself, not a carrier/classifier equivalence transport. □

Theorem 530.24 (UnaryDirectionBridge scoped source anchors)**.** *Every accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet* $U = (N, A, B, K, S, L, H, C, P, M)$ *binds the* $\mathsf{Nat}^{\uparrow}$ *unary source, the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source, the kernel-distinctness ledger, and the standard-boundary row to the displayed support*

$$\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The binding is a source-route obligation: N *and* A *remain separate source classifiers,* K *records their kernel separation,* S *is the only standard-boundary row, and* L, H, C, P, M *are the ledger, transport, continuation, provenance, and local certificate rows through which downstream consumers read the packet.*

Proof. Start with the carrier and carrier admission results, Definition 530.1 and Definition 530.12. They make $N, A, B, K, S, L, H, C, P, M$ the complete admitted packet surface and place the source rows, bridge row, ledger row, boundary row, transport rows, route row, package row, and local name in one carrier.

Apply the kernel-distinct obligations and standard boundary, Definition 530.2 and Definition 530.3. These force every source comparison to read K before S and prevent the unary and zero-spine source classifiers from being merged.

Use the ledger policy, classifier stability, and ledger exactness of Definition 530.4, Definition 530.13, and Definition 530.14. The only stable transport route is the displayed H, C, P, M route, so every accepted source read is anchored to the listed BEDC support.

Finally apply the obligation closure of Definition 530.17. It repacks these rows as the local $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ obligation surface, with no additional kernel equality or host natural-number route. □

Theorem 530.25 (UnaryDirectionBridge scoped additive route)**.** *Every* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumer route out of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet factors through the scoped source anchors of Definition 530.24 before reaching the standard boundary. Thus the additive route receives the unary source, zero-spine source, bridge row, kernel-distinctness ledger, ledger policy, componentwise* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row before any standard-boundary read.*

Proof. First apply the scoped source anchors. They bind $N, A, B, K, S, L, H, C, P, M$ to the finite BEDC support and make K available before the standard-boundary row is consumed.

Next use the additive consumer exactness and additive route scope, Definition 530.16 and Definition 530.19. These results identify the $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ route as a consumer of the displayed packet, not as a separate operation over host equality or quotient histories.

The carrier admission theorem and ledger policy keep the route inside the admitted rows. The standard-boundary theorem then places S after the source and kernel-distinctness anchors, while classifier stability transports the same route through H, C, P, M.

Finally the obligation closure repacks the route as a downstream obligation of the same local certificate. The consumer therefore reaches the boundary only through the scoped source anchors and the displayed bridge packet. □

Theorem 530.26 (UnaryDirectionBridge scoped NameCert transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet preserves the scoped bridge packet of Definition 530.24 and the additive route of Definition 530.25. The transported packet cannot collapse the* $\mathsf{Nat}^{\uparrow}$ *unary source classifier and the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source classifier into one kernel row.*

Proof. Begin with classifier stability and source-classifier transport, Definition 530.13 and Definition 530.15. They state that componentwise transport preserves the source rows, bridge row, kernel ledger, standard-boundary row, route rows, package provenance, and local naming row.

Apply the scoped source anchors to the transported packet. The anchors still require N and A as separate source classifiers and still require K before any comparison reaches S.

Use ledger exactness, ledger policy, and the standard boundary. These exclude host natural-number equality, quotient identification of histories, and any second kernel carrier as sources for the transported route.

Finally combine the scoped additive route with the obligation closure. The resulting transported $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$ packet is the same finite bridge surface over BHist, hsame, Cont, Pkg, NameCert, so transport preserves the route without identifying the two source classifiers. □

Theorem 530.27 (UnaryDirectionBridge standard-boundary consumer factorization)**.** *Every* $\mathsf{Add}^{\uparrow}$ *or* $\mathsf{AxisAdd}^{\uparrow}$ *consumer of the standard boundary of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet*

$$U = (N, A, B, K, S, L, H, C, P, M)$$

factors through the displayed unary source row N, zero-spine source row A, bridge row B, kernel-distinctness ledger K, standard-boundary row S, ledger policy L, componentwise transports H, continuation routes C, package provenance P, and local name M. The consumer receives no carrier equivalence between $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*, no host natural-number equality, and no quotient identification of histories.*

Proof. Start from carrier admission and the standard-boundary theorem. They make S a row of the accepted packet and place it after the source rows, bridge row, and kernel-distinctness ledger.

Apply the scoped source anchors. They require $N, A, B, K, S, L, H, C, P, M$ before any boundary consumer can read the packet, and they keep the unary and zero-spine source classifiers separate.

Now use the scoped additive route and additive consumer exactness. These route every $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ read through the displayed bridge packet rather than through a host-side equality or a quotient transport.

Finally apply scoped NameCert transport. Componentwise transport preserves the same packet surface, so any standard-boundary consumer factors through the displayed rows and through no alternate carrier/classifier equivalence. □

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier standard boundary consumer factorization`

Theorem 530.28 (UnaryDirectionBridge public axis readback)**.** *Every public consumer of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet*

$$U = (N, A, B, K, S, L, H, C, P, M)$$

reads the $\mathsf{Nat}^{\uparrow}$ *source carrier, the* $\mathsf{AxisNat}^{\uparrow}$ *source carrier, the* $\mathsf{Add}^{\uparrow}$ *surface, and the* $\mathsf{AxisAdd}^{\uparrow}$ *surface through the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route. The axis row cannot be replaced by the ordinary unary row, and the ordinary unary row cannot be replaced by the axis row.*

Lean4 ✓ `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier public axis readback`

Proof. Begin with the source obligation inventory of Definition 530.6. It lists the unary source row N, the zero-spine source row A, the additive surfaces read through S, and the support rows H, C, P, M as one finite packet.

Next apply kernel-axis composition from Definition 530.21. The kernel-distinctness ledger K must be read before the standard-boundary row S, so the ordinary unary and axis rows remain separate while they enter one public route.

Finally use the standard-boundary consumer factorization of Definition 530.27. It routes every $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumer through $N, A, B, K, S, L, H, C, P, M$ and excludes carrier equivalence, host natural-number equality, and quotient identification of histories. Hence the public readback composes the Nat/Add and AxisNat/AxisAdd surfaces without substituting one source row for the other. □

Theorem 530.29 (UnaryDirectionBridge AxisZeckendorf ledger gate)**.** *Any* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *consumer of an accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet must read the kernel-distinctness ledger K and the additive route through $N, A, B, S, L, H, C, P, M$ before it can see the bridge as a downstream obstruction. In particular, the consumer cannot bypass the ledger by identifying the* $\mathsf{Nat}^{\uparrow}$ *unary source classifier with the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine source classifier.*

Proof. Project the accepted packet. The carrier and kernel-distinct obligations expose K as the row that separates the unary and zero-spine source classifiers.

Use the ledger policy and ledger exactness. They make K and L mandatory for downstream reads and exclude host equality, quotient identification, or a second kernel carrier as substitute evidence.

Apply additive consumer exactness, kernel-axis composition, and the downstream seal. These results route $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ consumers through the additive bridge packet before the obstruction row is visible.

Therefore the bridge can be consumed only after the kernel-distinctness ledger and additive route have been read from the same displayed packet. □

Theorem 530.30 (UnaryDirectionBridge root-unblock readback package)**.** *The accepted* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *packet* $U = (N, A, B, K, S, L, H, C, P, M)$ *forms a root-unblock readback package for downstream* $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, *and* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *consumers. The package consists exactly of carrier admission, kernel-distinct classifier rows, ledger exactness, the standard-boundary consumer factorization of Definition 530.27, the AxisZeckendorf ledger gate of Definition 530.29, scoped NameCert transport, and the downstream seal.*

Proof. Begin with carrier admission. It fixes the readback packet as the ten displayed rows $N, A, B, K, S, L, H, C, P, M$.

Add the kernel-distinct obligations and ledger exactness. These make the source classifiers separate at the kernel row and make the ledger the public witness for that separation.

Use standard-boundary consumer factorization to bind $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ consumers to the same packet. Use the AxisZeckendorf ledger gate to ensure that obstruction-facing consumers read the ledger and additive route before the bridge can be consumed.

Finally combine scoped NameCert transport with the downstream seal. The assembled package exports the bridge as the displayed finite readback surface and does not export carrier equivalence transport, host natural-number equality, or quotient identification. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{UnaryDirectionBridge}^{\uparrow}$ packet is generated by a $\mathsf{Nat}^{\uparrow}$ unary source row, an $\mathsf{AxisNat}^{\uparrow}$ zero-spine source row, the displayed bridge object, kernel-distinctness witness, standard-boundary row, ledger policy, hsame transports, Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public surface is the $\mathsf{Nat}^{\uparrow}/\mathsf{AxisNat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ readback through Definition 530.27 and Definition 530.28, with the kernel-distinct ledger and displayed BHist, hsame, Cont, Pkg, and NameCert route still visible.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnaryDirectionBridge}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.UnaryDirectionBridgeUp.UnaryDirectionBridgeCarrier\_public\_axis\_readback`

Bridge status: none

Origin: human

Not claimed: This chapter does not identify $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$ inside the kernel, close the full standard-model isomorphism, or assert bridge checking for the unary-direction bridge packet.

Upgrade path: No paper-side theory-grade upgrade remains for the displayed public-axis readback; bridge checking is still not claimed.

Chapter 531

ClosedSubstitutionSeal NameCert Construction

The $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ interface packages the closed-term substitution boundary as a finite naming certificate over the MetaCIC syntax rows. It records the source term, the closure depth, the substitution payload, the closedness classifier, the inert substitution row, the inert shift row, the audit row, and the self-compilation handoff. The audit row A is read through chapter 779 before the self-compilation handoff is consumed. A consumer can cite the packet as a local compiler boundary, but it cannot read semantic substitution correctness for open terms, subject reduction for the full evaluator, or a host truth predicate.

Definition 531.1 (Closed-term substitution seal carrier)**.** *A* $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, D, V, Q, L, M, A, K, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC* `Term` *row consumed by the compiler boundary;*
2. *D is the de Bruijn depth row at which the boundary is read;*
3. *V is the substitution payload term carried through the traversal;*
4. *Q is the* `ClosedAt` *classifier row certifying that T is closed at D;*
5. *L is the inert substitution row displaying* $\texttt{substitute}(D, V, T) = T$*;*
6. *M is the inert shift row displaying the matching closed-boundary shift equality for T;*
7. *A is the axiom-pure audit row tying the displayed equalities to the records-side MetaCIC proof surface;*
8. *K is the self-compilation handoff row by which downstream compiler stages read only T, D, V, Q, L, M, A;*
9. *H stores componentwise* hsame *transports for the term, depth, payload, closedness classifier, inert rows, audit row, and handoff row;*
10. *C stores displayed* Cont *routes for reading the classifier, the inert substitution row, the inert shift row, the audit row, and the handoff row;*
11. *P is the* Pkg *provenance over* `Term`*,* `ClosedAt`*, substitution, shift, audit, self-compilation,* hsame*,* Cont*, and* NameCert*;*

12. N is the local $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSeal}^{\uparrow}}$ *row.*
The carrier contains no open-term semantic substitution theorem, no full evaluator subject-reduction theorem, no compiler completeness theorem, and no host truth predicate.

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionSealUp.ClosedSubstitutionSealTasteGate single carrier alignment`

Theorem 531.2 (Closed substitution seal boundary)**.** *For every accepted* $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ *packet* $S = (T, D, V, Q, L, M, A, K, H, C, P, N)$*, the only substitution-facing boundary exported to a downstream consumer is the displayed route through the closedness classifier* Q*, the inert substitution row* L*, the inert shift row* M*, the audit row* A*, the self-compilation handoff* K*, and the transport, continuation, package, and local naming rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionSealUp.ClosedSubstitutionSealBoundary`

Proof. Project the carrier of Definition 531.1. The term, depth, payload, closedness classifier, inert substitution row, inert shift row, audit row, handoff row, transports, continuations, provenance, and local naming row are all explicit coordinates of the same finite packet.

The substitution-facing consumer must therefore read Q before reading L. The row L is displayed only at the closed boundary certified by Q, while M records the corresponding binder-shift boundary for the same term row.

The remaining rows A, K, H, C, P, N route and name that finite surface. Since the carrier lists no semantic substitution theorem for open terms and no full evaluator theorem, the exported boundary is exactly the closed substitution seal. □

Theorem 531.3 (Closed substitution compile safety)**.** *Any self-compilation consumer of an accepted* $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ *packet factors through the handoff row* K*. That row exposes only the source term* T*, depth* D*, payload* V*, closedness classifier* Q*, inert substitution row* L*, inert shift row* M*, audit row* A*, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It does not expose a route by which the payload* V *becomes a free name in* T*.*

Proof. Start with the handoff coordinate K. By definition it is the self-compilation row attached to the same source term, depth, payload, classifier, inert rows, and audit row as the carrier.

The consumer reads Q first. Once Q is fixed, the inert substitution row L states that the substitution attempt at depth D returns T, and the inert shift row M states that the corresponding boundary shift spends no extra name budget for that closed term.

Finally route the result through A, H, C, P, N. These rows certify audit visibility, componentwise transport, continuation replay, provenance, and local naming; none is a payload-injection coordinate. Thus the compile-facing safety claim is a finite non-escape property of the displayed packet. □

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionSealUp.ClosedSubstitutionSeal compile safety`

Theorem 531.4 (Closed substitution audit surface)**.** *The audit surface of a* $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ *packet is exhausted by the MetaCIC* `Term` *row, the* `ClosedAt` *classifier, the displayed substitution and shift equalities, the self-compilation handoff, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and the local* NameCert *row. It supplies no unchecked semantic equality, no global compiler theorem, no full subject-reduction theorem, and no host-side truth predicate.*

Proof. Read the packet through Definition 531.2. The boundary exposes the term, classifier, inert substitution row, inert shift row, audit row, handoff row, and the rows that transport, replay, package, and name them.

Read the compiler-facing route through Definition 531.3. The consumer of self-compilation sees only the closed-boundary handoff and cannot project a free-name injection route for the substitution payload.

These two readings exhaust the carrier inventory from Definition 531.1. Since none of the displayed coordinates is an unchecked semantic equality, a global compiler theorem, a full subject-reduction theorem, or a host truth predicate, the audit surface is exactly the finite NameCert packet. □

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionSealUp.ClosedSubstitutionSealAudit surface`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSeal}\uparrow}$.

Constructive story: A $\mathsf{ClosedSubstitutionSeal}^{\uparrow}$ packet is generated from a MetaCIC term row, a depth row, a substitution payload, a `ClosedAt` classifier, displayed inert substitution and shift rows, an audit row, a self-compilation handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSeal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 531.1, Definition 531.2, Definition 531.3, and Definition 531.4 specify the finite closed-substitution seal carrier, non-escape boundary, compiler handoff, and audit surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionSeal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close open-term semantic substitution, full evaluator subject reduction, compiler completeness, host truth predicates, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the packet and checked theorems for classifier transport, inert substitution, inert shift, audit visibility, and self-compilation handoff.

Chapter 532

A Concrete Naming Certificate for $\mathsf{BinderBudgetSeal}^{\uparrow}$

The $\mathsf{BinderBudgetSeal}^{\uparrow}$ surface turns closed-term substitution into finite NameCert data. It records the binder-depth boundary at which a MetaCIC term is closed, the shift route at that boundary, the substitution route at the same boundary, and the consumer handoff that lets self-compilation read the boundary without receiving an unregistered free variable.

The upstream boundary is the closed-term substitution surface of chapter 527. Its depth row, closed term row, shift and substitution boundary rows, and non-escape ledger supply the exact boundary data that $\mathsf{BinderBudgetSeal}^{\uparrow}$ repackages as a binder-budget seal.

Definition 532.1 (BinderBudgetSeal carrier)**.** *A* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, T, V, S, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the binder-depth budget row naming the finite boundary at which the term is certified closed;*
2. *T is the MetaCIC term row whose free variables are contained below the boundary named by D;*
3. *V is the displayed substitution payload row, allowed to be inspected only through the boundary route;*
4. *S is the shift route row recording that shifting at the closed boundary returns the same term;*
5. *U is the substitution route row recording that substitution at the closed boundary returns the same term;*
6. *H records the componentwise* hsame *transports for the depth, term, payload, shift, substitution, package, and local naming rows;*
7. *C records the displayed* Cont *routes by which self-compilation consumers read D, T, V, S, U without leaving the finite packet;*
8. *P is the* Pkg *provenance row over* BHist, hsame, Cont, *the MetaCIC closed-term syntax rows, and local certificate dependencies;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BinderBudgetSeal}^{\uparrow}}$ *row.*

The local classifier compares two accepted carriers only by the displayed binder-depth row, term row, payload row, shift route, substitution route, and their recorded hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance.*

Theorem 532.2 (BinderBudgetSeal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BinderBudgetSeal}^{\uparrow}}$ *surface for Definition 532.1 has five finite obligations: expose the binder-depth carrier* $B = (D, T, V, S, U, H, C, P, N$ *restrict comparison to the displayed depth, term, payload, route, transport, package, and local naming rows; route shift reads through the closed-boundary row* S*; route substitution reads through the closed-boundary row* U*; and exclude any export of fresh free variables, host capture-avoidance primitives, quotient syntax equality, or compiler-stage equality outside the packet.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealTasteGate single carrier alignment`

Proof. Carrier exposure is projection from the finite tuple. The binder-depth row, term row, payload row, shift route, substitution route, transport rows, continuation routes, package provenance, and local naming row are all displayed coordinates.

Classifier locality follows from the last sentence of the carrier definition. A comparison cannot inspect a hidden syntax quotient, a host-level capture-avoidance predicate, or a compiler-stage equality because no such coordinate is present.

The two route obligations are read from S and U. The shift route records the closed-boundary non-escape law, and the substitution route records the matching non-escape law for the displayed payload. Thus every consumer reaches the closed-term boundary before it can read binder movement or substitution data. □

Theorem 532.3 (BinderBudgetSeal shift non-escape)**.** *For an accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *carrier* $B = (D, T, V, S, U, H, C, P, N)$*, every shift request at the boundary named by* D *factors through the displayed route* S *before the term row* T *is returned to a consumer. The route exports the same MetaCIC term row and no fresh free variable, depth extension, quotient syntax equality, or host binder operation.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealShiftNonescape`

Proof. Begin with the binder-depth coordinate. The row D names the finite closed boundary for the term row T, so a shift request at that boundary must first read D.

Next read the shift route S. The route is the packet coordinate that records the closed-boundary identity for binder movement, so the consumer receives T through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally apply the non-escape clause of the NameCert obligations. Since the carrier has no coordinate for a new free variable, depth extension, syntax quotient, or host binder operation, the shift request cannot export any such object. □

Theorem 532.4 (BinderBudgetSeal substitution non-escape)**.** *For an accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *carrier* $B = (D, T, V, S, U, H, C, P, N)$*, every substitution request at the boundary named by* D *factors through the displayed route* U *before the term row* T *is returned to a consumer. The payload row* V *is read only as boundary data, and the route exports no substituted free variable, host capture-avoidance primitive, quotient syntax equality, or compiler-stage equality outside the finite packet.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealSubstitutionNonescape`

Proof. Start from the closed-boundary row D and the term row T. The substitution request is accepted only at the boundary named by D, so the payload row V is not a separate source of free-variable data.

Read the substitution route U. This row records the closed-boundary return of the original term, and the continuation routes in C let a consumer inspect that result without leaving the packet.

Use the NameCert boundary obligation. Because the carrier contains no field for host capture avoidance, quotient syntax equality, compiler-stage equality, or a fresh substituted free variable, the substitution request is exhausted by D, T, V, U and the displayed provenance rows. □

Theorem 532.5 (BinderBudgetSeal self-compile handoff)**.** *Every self-compilation consumer that receives an accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *carrier receives a finite packet over the same binder-depth row, MetaCIC term row, substitution payload row, shift route, substitution route,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The handoff is exactly the closed-boundary budget for shift and substitution non-escape, and it does not claim full compiler closure, normalization, definitional equality, quotient syntax equality, or ambient host soundness.*

Proof. First bind the shift side. By Definition 532.3, a boundary shift request reads D and S before returning the displayed term row.

Next bind the substitution side. By Definition 532.4, a boundary substitution request reads D, T, V, and U before any consumer can inspect the result.

Now assemble the consumer packet. The displayed transports, continuation routes, package provenance, and local certificate rows are the only comparison and handoff data named by Definition 532.2.

The remaining exclusions follow from the same carrier inventory. No coordinate supplies full compiler closure, normalization, definitional equality, quotient syntax equality, or host soundness, so the handoff is exactly the finite binder-budget boundary for shift and substitution non-escape. □

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealSelfCompileHandoff`

Definition 532.6 (BinderBudgetSeal obligation surface)**.** *The* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *obligation surface is the single-carrier packet*

$$B = (D, T, V, S, U, H, C, P, N)$$

of Definition 532.1, equipped with four displayed rows: carrier admission for the binder-depth boundary and closed term; route admission for the shift row S and substitution row U; compiler-boundary handoff through the same hsame*,* Cont*,* Pkg*, and local* NameCert *rows; and non-escape for fresh variables, host capture-avoidance primitives, quotient syntax equality, compiler-stage equality, and ambient host soundness. The TasteGate alignment marker above is read as a single-carrier constraint on this packet, not as permission to introduce a second compiler carrier.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealObligationSurface`

Theorem 532.7 (BinderBudgetSeal carrier admission)**.** *An accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *obligation surface admits exactly the carrier rows $D, T, V, S, U, H, C, P, N$. The binder-depth row D, term row T, and payload row V are inspected only through the shift and substitution routes S, U and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows; the AI-origin TasteGate marker keeps this admission on the same carrier.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealCarrierAdmission`

Proof. Project the carrier of Definition 532.1. The rows named by the obligation surface are precisely the displayed finite coordinates.

Use the TasteGate alignment recorded after Definition 532.2 as the carrier discipline. It aligns the checked marker with the same packet B, so carrier admission cannot allocate a separate compiler input, output, or ambient syntax object.

The remaining access paths are the route rows and certificate rows already named by the carrier. Thus admission is exactly the single finite surface $D, T, V, S, U, H, C, P, N$. □

Theorem 532.8 (BinderBudgetSeal compiler boundary)**.** *Every compiler or self-compilation read of an accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *obligation surface factors through the closed-boundary routes S and U before it may inspect the term row T or payload row V. The boundary exports the same carrier $B = (D, T, V, S, U, H, C, P, N)$ and no second carrier for compiler closure, normalization, definitional equality, quotient syntax equality, or host soundness.*

Proof. Start with the self-compile handoff of Definition 532.5. It already requires a consumer to receive the binder-depth, term, payload, shift, substitution, transport, continuation, package, and naming rows as one packet.

The shift and substitution non-escape theorems force the two compiler-facing reads through S and U. These rows both reference the same binder-depth boundary D, so the compiler boundary is not a separate stage carrier.

Finally apply the obligation surface. Since it lists no coordinate for compiler closure, normalization, definitional equality, quotient syntax equality, or host soundness, such data cannot be exported by the boundary. □

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealCompilerBoundary`

Theorem 532.9 (BinderBudgetSeal non-escape)**.** *Every exported read of an accepted* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *obligation surface is exhausted by $D, T, V, S, U, H, C, P, N$. A request for a fresh free variable, a host capture-avoidance primitive, a quotient syntax equality, a compiler-stage equality, a normalization theorem, or ambient host soundness leaves the carrier and is not returned as BinderBudgetSeal evidence.*

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealNonEscape`

Proof. Read the request through the continuation routes in C. If it asks for the displayed boundary, term, payload, shift, substitution, transport, provenance, or naming rows, the finite packet supplies that row.

The shift and substitution non-escape theorems handle the two route cases. They return the same term row through S or U and do not allocate new free-variable or host binder data.

All remaining requests are absent from the obligation surface. The single-carrier TasteGate discipline and the compiler-boundary theorem keep those absent requests from being repacked as a second carrier or as compiler-stage evidence. □

Theorem 532.10 (BinderBudgetSeal TasteGate carrier grounding)**.** *The carrier rows $D, T, V, S, U, H, C, P, N$ used by the obligation surface, compiler boundary, and non-escape theorem are the same finite packet. Any TasteGate instance for* $\mathsf{BinderBudgetSeal}^{\uparrow}$ *must therefore be discharged on that single carrier and not on a compiler-stage carrier, normalization carrier, quotient syntax carrier, or host soundness carrier.*

Proof. Start with the carrier definition and the obligation surface, Definition 532.1 and Definition 532.6. Both display the same tuple $B = (D, T, V, S, U, H, C, P, N)$ and name no second carrier.

Use carrier admission from Definition 532.7. It reads D, T, V only through S, U, H, C, P, N, so the admitted surface is the same packet named by the obligation theorem.

Apply the compiler boundary and non-escape theorems, Definition 532.8 and Definition 532.9. They rule out compiler-stage, normalization, quotient syntax, and host soundness rows outside the displayed tuple.

The TasteGate carrier must therefore be grounded on row equality of this packet. Since every cited surface names $D, T, V, S, U, H, C, P, N$, a separate carrier would fail to be the accepted $\mathsf{BinderBudgetSeal}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealTasteGateCarrierGrounding`

Closure status for $\mathsf{NameCert}_{\mathsf{BinderBudgetSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BinderBudgetSeal}^{\uparrow}$ packet is generated from a binder-depth boundary, a MetaCIC term row, a substitution payload row, shift and substitution non-escape routes, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BinderBudgetSeal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 532.1, Definition 532.6, Definition 532.2, Definition 532.7, Definition 532.8, Definition 532.9, and Definition 532.10 bridge the public packet through the same finite carrier-alignment route over binder depth, MetaCIC term, substitution payload, shift and substitution nonescape, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BinderBudgetSeal}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BinderBudgetSealUp.BinderBudgetSealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim StdBridge checking, full compiler closure, normalization, definitional equality, quotient syntax equality, or host soundness.

Upgrade path: Further verification work requires purity auditing and, separately, a StdBridge theorem for the same finite carrier-alignment route.

Chapter 533

A Concrete Naming Certificate for OperationalMembrane$^{\uparrow}$

The OperationalMembrane$^{\uparrow}$ packet is the concrete NameCert surface that turns the audit-gate membrane into a finite carrier. It does not claim that audit scripts are mathematical generators by themselves. It records the rows by which source syntax, proof ancestry, import separation, paper-Lean name resolution, displayed transport, consumer continuation, package provenance, and the boundary ledger remain visible to downstream governance chapters.

The membrane consumes the public GAP-ledger residue of chapter 520 and the finite kernel acceptance trace of chapter 557 before packaging the source-scan, proof-ancestry, import-separation, and paper-Lean drift rows. The closure witness ledger of chapter 487 supplies the downstream accounting surface for accepted and refused boundary crossings, so the packet is a lattice-facing audit carrier rather than an isolated kernel row.

Definition 533.1 (OperationalMembrane carrier)**.** *An* OperationalMembrane$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, A, I, D, H, C, P, L, N)$$

with the following displayed rows:

1. *S is the source-axiom scan row, recording the finite BEDC source window, the source-token classifier, and the refusal result for internal axiom declarations;*
2. *A is the strict axiom-purity row, recording the inspected public theorem names, the reported proof ancestry, and the forbidden-dependency classifier for* Classical.choice*,* Quot.sound*, and* propext*;*
3. *I is the conservativity import-separation row, recording protected baseline prefixes, non-baseline origins, inspected import edges, and the pass-or-refuse import classifier;*
4. *D is the paper-Lean drift-resolution row, recording each manuscript marker, its named Lean target, and the finite resolution result;*
5. *H records componentwise* hsame *transports for the source, ancestry, import, drift, ledger, package, continuation, and naming rows;*
6. *C records the displayed* Cont *routes by which consumers read the source scan, axiom-purity report, import boundary, drift resolution, ledger, provenance, and local certificate;*
7. *P is the* Pkg *provenance over* BHist*, source files, proof-ancestry reports, import ledgers, paper markers, Lean target names,* hsame*,* Cont*,* Pkg*, and* NameCert*;*

8. *L is the boundary ledger row recording that accepted reads must pass through exactly one of the displayed audit faces and that rejected reads remain outside the packet;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}$ *row.*

The carrier contains no source axiom permission, no hidden host axiom, no baseline import exemption, no fuzzy marker match, no theorem of global semantic conservativity, and no host-kernel completeness principle.

Definition 533.2 (OperationalMembrane classifier)**.** *For accepted* $\mathsf{OperationalMembrane}^{\uparrow}$ *carriers M and M', the classifier* $\mathsf{OperationalMembraneClassifier}(M, M')$ *compares exactly the rows of Definition 533.1. It transports source scans through S, proof-ancestry reports through A, import separation through I, paper-Lean name resolution through D, boundary-ledger entries through L, and provenance through H, C, P, N.*

The classifier has no component for asserting that a rejected source token is permitted, that a forbidden proof dependency is harmless, that a non-baseline import edge is baseline, that an unresolved paper marker names a nearby theorem, or that the audit surface proves global semantic completeness.

Theorem 533.3 (OperationalMembrane NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (S, A, I, D, H, C, P, L, N),$$

source-axiom scan exposure, strict axiom-purity exposure, conservativity import-separation exposure, paper-Lean drift-resolution exposure, visible hsame *transport, displayed* Cont *locality,* Pkg *provenance, boundary-ledger accounting, and non-escape for source axioms, forbidden host proof ancestry, hidden import exemptions, unresolved Lean names, and global semantic completeness.*

Proof. Carrier admission is projection from Definition 533.1. The packet has exactly the rows $S, A, I, D, H, C, P, L, N$, so every admitted consumer read begins at one of those coordinates.

The four audit faces are then read separately. The source face is S, the proof-ancestry face is A, the import-separation face is I, and the paper-Lean name-resolution face is D.

Transport and locality are displayed by H and C. These rows move and replay the four audit faces componentwise without adding a source axiom, a hidden proof dependency, an import exemption, or a substitute Lean name.

Finally P, L, N package the provenance, ledger, and local certificate. Since the excluded objects have no carrier coordinate, no $\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}$ projection can export them. □

Lean4 ✓ `BEDC.Derived.OperationalMembraneUp.OperationalMembranePacket namecert obligation surface`

Theorem 533.4 (OperationalMembrane ledger exactness)**.** *For an accepted* $\mathsf{OperationalMembrane}^{\uparrow}$ *packet, the boundary ledger L is exact for the displayed carrier: a consumer may read a source-scan decision through S, an axiom-purity decision through A, an import-separation decision through I, or a paper-Lean resolution decision through D, together with the visible transport, continuation, package, and local naming rows. No consumer may use L to manufacture a fifth audit face or to erase a rejected boundary crossing.*

Proof. Project L from the carrier and compare it with the carrier coordinates. Its entries name the same four audit faces S, A, I, D and the same structural rows H, C, P, N.

Apply the classifier of Definition 533.2. It transports exactly the displayed rows and contains no comparison component for an additional audit face.

The non-escape clause of Definition 533.1 fixes the rejected side. Because no coordinate permits source axioms, forbidden dependencies, import exemptions, fuzzy marker matches, or global completeness principles, the ledger cannot turn a refused crossing into accepted content. □

Lean4 ✓ `BEDC.Derived.OperationalMembraneUp.OperationalMembranePacket ledger exactness`

Theorem 533.5 (OperationalMembrane drift resolution)**.** *For an accepted* $\mathsf{OperationalMembrane}^{\uparrow}$ *packet, the paper-Lean row D is the complete public route by which manuscript markers are resolved against Lean names. A consumer may read the marker text, the target name, the finite resolution result, the componentwise transport row, the continuation route, package provenance, boundary-ledger entry, and local* $\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}$ *row. It may not replace a missing target by a similar declaration, a neighboring namespace, a prose interpretation, or an unchecked string.*

Lean4 ✓ `BEDC.Derived.OperationalMembraneUp.OperationalMembranePacket drift resolution`

Proof. Read D from Definition 533.1. The row contains the marker text, the claimed Lean target, and the finite resolution result.

Use H and C to transport and expose that row to consumers. The target name remains the displayed target name, and the continuation route does not introduce fuzzy matching or namespace search.

The ledger row L records whether the read belongs to the accepted boundary surface. Together with P and N, it packages the result as a local naming certificate and leaves missing or unchecked names outside the packet. □

Theorem 533.6 (OperationalMembrane scoped boundary)**.** *The scoped* $\mathsf{OperationalMembrane}^{\uparrow}$ *boundary consists exactly of the source scan S, axiom-purity row A, import-separation row I, paper-Lean drift row D, componentwise* hsame *transports H, displayed* Cont *routes C, package provenance P, boundary ledger L, and local* NameCert *row N of the carrier $M = (S, A, I, D, H, C, P, L, N)$. It excludes source axiom permission, hidden proof ancestry, import exemptions, fuzzy Lean-name matching, and global semantic completeness.*

Proof. Start with Definition 533.1 and Definition 533.2. They make $S, A, I, D, H, C, P, L, N$ the complete carrier and comparison surface.

Apply Definition 533.3 to expose those rows as the local naming certificate. Then use Definition 533.4 to keep accepted and rejected boundary crossings on the displayed ledger.

Finally apply Definition 533.5 for the paper-Lean face. The same carrier has no row for source axiom permission, hidden proof ancestry, import exemption, fuzzy name replacement, or global semantic completeness, so the scoped boundary is exactly the displayed membrane. □

Theorem 533.7 (OperationalMembrane standard schema bridge)**.** *The public* $\mathsf{OperationalMembrane}^{\uparrow}$ *surface bridges to the standard operational schema exactly through the displayed source scan S, strict axiom-purity row A, import-separation row I, paper-Lean drift row D, boundary ledger L, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *provenance row P, and local* NameCert *row N of the carrier $M = (S, A, I, D, H, C, P, L, N)$.*

The bridge has no extra audit face. It does not permit source axioms, forbidden proof ancestry, baseline-import exemptions, fuzzy Lean-name resolution, host-kernel completeness, or global semantic conservativity beyond the displayed membrane rows.

Lean4 ✓ `BEDC.Derived.OperationalMembraneUp.OperationalMembraneUp StdBridge`

Proof. Start with the carrier and classifier. The definitions display the exact rows $S, A, I, D, H, C, P, L, N$ and compare only source scans, proof-ancestry reports, import separation, paper-Lean name resolution, boundary-ledger entries, and structural rows.

Apply the NameCert obligation surface and ledger exactness. These theorems expose the four audit faces and ensure the boundary ledger cannot create a fifth face or erase a refused crossing.

Use drift resolution for the paper-Lean component. It keeps marker resolution tied to the displayed target names and rejects nearby declarations, prose interpretation, or unchecked strings as replacements.

Finally apply the scoped boundary theorem. Since all standard operational reads are already routed through the displayed membrane rows, the public surface supplies a paper-level standard schema bridge without adding host-kernel completeness or global conservativity. □

Theorem 533.8 (OperationalMembrane finite-route induction)**.** *Every governance-facing route from an accepted* $\mathsf{OperationalMembrane}^{\uparrow}$ *carrier is obtained by finite induction over the displayed rows* $S, A, I, D, H, C, P, L, N$ *of* $M = (S, A, I, D, H, C, P, L, N)$*. Each induction step either projects one audit face or replays it through* hsame*,* Cont*,* Pkg*, and local* NameCert *data, and no step introduces source axiom permission, forbidden dependency acceptance, import exemption, fuzzy marker matching, or global semantic completeness.*

Proof. The base route is carrier projection. Apply Definition 533.1 and Definition 533.2 to expose exactly $S, A, I, D, H, C, P, L, N$ as the finite inventory and comparison surface.

For the audit-face steps, use Definition 533.3. It projects source scans through S, proof ancestry through A, import separation through I, and paper-Lean name resolution through D without creating any additional audit face.

For the replay steps, apply Definition 533.4 and Definition 533.5. The ledger row L preserves accepted and rejected crossings, while the drift row keeps marker resolution tied to displayed target names.

Finally use Definition 533.6 and Definition 533.7. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the induction has no constructor for source axiom permission, forbidden dependency acceptance, import exemption, fuzzy marker matching, or global semantic completeness. □

The operational membrane's consumer route reads the audit-gate boundary rows of chapter 526 before exporting membrane-visible operational data. Thus the membrane packet is a downstream operational view of the same finite boundary ledger, not an isolated kernel surface.

Closure status for $\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OperationalMembrane}^{\uparrow}$ packet is generated from source-axiom scans, strict proof-ancestry reports, import-separation ledgers, paper-Lean name-resolution rows, visible transport, continuation routes, package provenance, a boundary ledger, and a local naming certificate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 533.1, Definition 533.2, Definition 533.3, Definition 533.4, Definition 533.5, Definition 533.6, and Definition 533.7 bind the carrier, classifier, ledger exactness, drift resolution, scoped operational boundary, and paper-level standard schema bridge rows for the displayed audit membrane.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OperationalMembrane}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.OperationalMembraneUp.OperationalMembranePacket\_namecert\_obligation\_surface`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close global semantic conservativity, host-kernel completeness, bridge checking, or a Lean implementation of the operational membrane carrier.

Upgrade path: No further formal-axis upgrade is pending because the displayed standard schema bridge is checked.

Chapter 534

A Concrete Naming Certificate for RecursorGenerator$^{\uparrow}$

The RecursorGenerator$^{\uparrow}$ surface packages the authorized use of inductive recursors as finite NameCert data. It records an accepted inductive signature, legal calls to the generated eliminator, branch packages, audit rows, MetaCIC closed-term substitution routes, componentwise transports, continuation access, provenance, and the local naming row. The packet is not an ambient reflection principle, an unchecked code generator, or an escape from the displayed inductive boundary.

The packet consumes the authorized eliminator surface of chapter 525 and the generator-boundary closure of chapter 486. AuthorizedGeneratorRecursor supplies the legal recursor-call and branch-package rows, while GeneratorClosure supplies the refusal of unlisted generator routes.

Definition 534.1 (RecursorGenerator carrier)**.** *A* RecursorGenerator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, E, B, A, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted inductive-signature row, naming the constructors, positivity check, parameter telescope, motive boundary, and generated recursor symbol;*
2. *E is the legal eliminator-call row, recording the displayed I.rec call, its motive, scrutinee, parameters, and the type-checker acceptance row for the call;*
3. *B is the branch-package row, listing one branch package for each constructor of I, with branch inputs, recursive hypotheses, output row, and branch-local* Pkg *provenance;*
4. *A is the audit row, recording kernel elaboration, source axiom refusal, and dependency-purity refusal for* Classical.choice, Quot.sound, *and* propext*;*
5. *M is the MetaCIC closed-term substitution row, recording that every generated branch output is handed off only through closed terms and substitution routes accepted by the displayed classifier;*
6. *H records componentwise* hsame *transports for I, E, B, A, M, the continuation row, the package row, and the local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the signature, eliminator call, branch packages, audit row, closed-term handoff, provenance, and local certificate;*
8. *P is the* Pkg *provenance over* BHist, Hist, *accepted inductive signatures, legal recursor calls, branch packages, audit rows, MetaCIC closed terms,* hsame, Cont, *and local* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RecursorGenerator}^{\uparrow}}$ *row.*

The carrier contains no host-level code generation oracle, no untyped eliminator, no branch outside the constructor list of I, no hidden theorem of global consistency, and no imported equality principle beyond the displayed audit row.

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorCarrierSemanticNameCertificate`

Theorem 534.2 (RecursorGenerator authorization obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RecursorGenerator}^{\uparrow}}$ *surface for the carrier of Definition 534.1 consists of the following finite obligations:*

1. *carrier exposure: every accepted packet exposes exactly $R = (I, E, B, A, M, H, C, P, N)$;*
2. *signature authorization: the eliminator call in E is licensed only by the accepted inductive-signature row I;*
3. *branch coverage: the branch-package row B contains exactly the constructor-indexed branches required by $I.\mathsf{rec}$;*
4. *audit confinement: the audit row A is read before any generated term is exported as a public certificate row;*
5. *closed-term handoff: the MetaCIC row M exports generated outputs only through closed-term substitution routes;*
6. *transport locality:* hsame *and* Cont *access are restricted to the displayed rows and cannot add another generator.*

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorAuthorizationObligations`

Proof. Carrier exposure is projection from Definition 534.1. The tuple R lists every coordinate a downstream consumer may inspect.

Signature authorization and branch coverage are read from I, E, B. The accepted signature row names the generated recursor, the eliminator-call row records a legal $I.\mathsf{rec}$ invocation, and the branch-package row supplies exactly the constructor-indexed branch data demanded by that invocation.

Audit confinement is read from A. A generated output can be consumed only after the kernel elaboration row and the source and dependency refusals have been recorded in the same finite packet.

Closed-term handoff is read from M. The MetaCIC substitution route exposes generated branch outputs as closed terms under the displayed classifier, so the handoff does not add an external computation principle.

Finally H, C, P, N record transport, continuation, provenance, and the local naming certificate. Since no coordinate names another generator or an unchecked eliminator, the authorization surface is exactly the displayed packet. □

Theorem 534.3 (RecursorGenerator audit non-escape)**.** *Every public output of an accepted* $\mathsf{RecursorGenerator}^{\uparrow}$ *packet factors through the audit row A, the accepted inductive-signature row I, the legal eliminator-call row E, and the branch-package row B. A consumer cannot recover a host code generator, a non-positive eliminator, an unlisted branch,* Classical.choice, Quot.sound, propext, *or an unrecorded axiom channel from the packet.*

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGenerator audit non escape`

Proof. Begin with the legal route. The only eliminator available in the carrier is the call recorded by E, and E is licensed by the accepted signature row I.

The branch data are then exhausted by B. Each constructor branch is part of the displayed branch-package row, and there is no coordinate for an additional branch or for a different eliminator.

The audit boundary is supplied by A. Since the packet exports generated rows only after the kernel, source-token, and dependency-purity checks have been read, the forbidden host principles have no public coordinate.

The remaining rows H, C, P, N preserve and expose the same finite data. They transport and package the accepted rows; they do not provide a hidden generator or axiom channel. □

Theorem 534.4 (RecursorGenerator closed-term handoff)**.** *For an accepted* $\mathsf{RecursorGenerator}^{\uparrow}$ *packet, every downstream theorem, definition, or certificate row consuming a generated recursor output receives it through the MetaCIC closed-term substitution row M together with the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{RecursorGenerator}^{\uparrow}}$ *provenance. The handoff preserves the accepted inductive-signature classifier and does not assert host reflection, semantic completeness, or global normalization.*

Proof. Project M from the carrier. It records the closed-term substitution route for generated branch outputs and ties that route to the classifier attached to the accepted signature row I.

Use H and C to move and read the generated output. Componentwise transport preserves the displayed signature, call, branch, audit, and closed-term rows, while continuation access exposes only those rows to the downstream consumer.

Package provenance is then supplied by P and the local naming row N. The consumer receives a finite certificate over the displayed dependencies, not a host reflection theorem or a global normalization result. □

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorCarrier closed term handoff`

Theorem 534.5 (RecursorGenerator authorized output factorization)**.** *Every public generated output of an accepted* $\mathsf{RecursorGenerator}^{\uparrow}$ *packet factors through the displayed rows*

$$(I, E, B, A, M, H, C, P, N).$$

The factorization reads the legal eliminator call, the constructor-indexed branch package, the audit row, and the MetaCIC closed-term handoff before any output is exposed. Hence no public output can bypass the branch coverage supplied by the $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *surface in chapter 525.*

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorAuthorizedOutputFactorization`

Proof. Start from the carrier of Definition 534.1. It exposes exactly the rows $I, E, B, A, M, H, C, P, N$, so a public generated output has no coordinate outside this tuple.

Apply the authorization surface of Definition 534.2. The signature row I licenses the legal eliminator call E, and the branch row B supplies exactly the constructor-indexed branches required by that call.

Use the audit and handoff rows next. By Definition 534.3, the output must pass through A, I, E, B before becoming public; by Definition 534.4, the exported generated term is delivered through M with the displayed transport, continuation, package, and local naming provenance.

The sibling $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ reference supplies the public branch-coverage boundary consumed by this chapter. Since the carrier has no extra branch or generator coordinate, the displayed factorization is exhaustive. □

Theorem 534.6 (RecursorGenerator branch and descent exhaustion)**.** *For an accepted* RecursorGenerator$^{\uparrow}$ *packet, every branch read and every descent read used by a generated recursor output is exhausted by the displayed constructor-indexed branch package B together with the MetaCIC closed-term substitution row M. The packet cannot expose an unlisted constructor case, a hidden recursive call, or a descent route outside the authorized recursor surface consumed from chapter 525.*

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorBranchDescentExhaustion`

Proof. Project the branch package from the carrier of Definition 534.1. The row B lists one branch package for each constructor of I, including branch inputs, recursive hypotheses, output row, and branch-local Pkg provenance.

Apply Definition 534.2. Branch coverage is one of the finite local obligations, and transport locality prevents hsame or Cont access from adding another generator.

The descent read is then tied to the closed-term handoff. By Definition 534.4, every downstream generated output is exported through M, so each branch output is read only as a MetaCIC closed-term substitution route accepted by the displayed classifier.

Finally use Definition 534.3. Since unlisted branches, hidden recursive calls, and unaudited dependency channels have no public coordinate, the constructor-indexed branch package and the closed-term substitution row exhaust the visible branch and descent surface. □

Theorem 534.7 (RecursorGenerator downstream scope package)**.** *Every downstream self-compilation,* TasteGate*, or generated-certificate consumer of an accepted* RecursorGenerator$^{\uparrow}$ *packet receives only the finite* RecursorGenerator$^{\uparrow}$ *scope package*

$$(I, E, B, A, M, H, C, P, N).$$

Such a consumer may inspect the authorized signature, eliminator call, branch package, audit row, closed-term handoff, transport, continuation, provenance, and local certificate rows, but it receives no host reflection principle, no unchecked generator, and no certificate outside this finite scope.

Proof. Read the consumer interface from the carrier of Definition 534.1. The continuation row C exposes precisely the signature, eliminator call, branch package, audit row, closed-term handoff, provenance, and local certificate routes.

The authorized output factorization of Definition 534.5 forces every public generated output through the same rows. The branch and descent exhaustion of Definition 534.6 prevents the consumer from replacing B or M with an external recursor route.

Apply Definition 534.2, Definition 534.3, and Definition 534.4. Together they bind signature authorization, audit confinement, and MetaCIC closed-term delivery inside the displayed packet.

Therefore a self-compilation row, a TasteGate row, or a generated-certificate row receives the finite RecursorGenerator$^{\uparrow}$ scope package and nothing beyond it. □

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorDownstreamScopePackage`

Theorem 534.8 (RecursorGenerator bridge route status)**.** *The public scope package* $(I, E, B, A, M, H, C, P, N)$ *bridges the concrete* RecursorGenerator$^{\uparrow}$ *packet to the abstract recursor-generator interface through the accepted inductive signature row, legal eliminator-call row, constructor-indexed branch package, audit row, closed-term handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The bridge exposes no host reflection principle, unchecked generator execution, unlisted branch, global normalization theorem, semantic completeness theorem, or dependency channel outside the displayed audit row.*

Proof. Start with the downstream scope package of Definition 534.7. It gives every public consumer exactly the rows $I, E, B, A, M, H, C, P, N$ and no external generator interface.

Use authorization, audit, and handoff. The accepted signature row I licenses the legal eliminator call E, the branch package B exhausts constructor cases, the audit row A bars the forbidden dependency channels, and the closed-term row M supplies the generated outputs as displayed handoffs.

Finally replay the structural rows. Componentwise hsame transport, displayed Cont access, Pkg provenance, and the local NameCert row connect the concrete packet to the abstract interface without adding another coordinate, so the bridge route is exactly the public scope package. □

Lean4 ✓ `BEDC.Derived.RecursorGeneratorUp.RecursorGeneratorBridgeRouteStatus`

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorGenerator}\uparrow}$.

Constructive story: A $\mathsf{RecursorGenerator}^{\uparrow}$ packet is generated from an accepted inductive signature, a legal recursor call, constructor-indexed branch packages, audit rows, MetaCIC closed-term substitution, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursorGenerator}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 534.1, Definition 534.2, Definition 534.3, Definition 534.4, Definition 534.5, Definition 534.6, Definition 534.7, and Definition 534.8 bind recursor generation as a finite authorization packet and bridge route for accepted inductive signatures, legal eliminator calls, audit non-escape, closed-term handoff, authorized output factorization, branch and descent exhaustion, downstream scope packaging, and abstract interface access.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursorGenerator}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge checking, host reflection, global normalization, semantic completeness, non-positive eliminators, or unchecked generator execution.

Upgrade path: No further paper-theory upgrade remains for this chapter; formal verification and bridge checking remain on the verification axis.

Chapter 535

A Concrete Naming Certificate for BoundedMonotoneCauchyWitness$^{\uparrow}$

The BoundedMonotoneCauchyWitness$^{\uparrow}$ interface records the finite surface through which a bounded monotone source row becomes a witnessed Cauchy real-name packet. It separates the displayed generator range, the explicit Cauchy witness, the real-history classifier transport, and the Real$^{\uparrow}$ seal handoff before any completion-facing consumer reads the packet.

This packet is the bounded-monotone consumer of the located monotone Cauchy surface in chapter 438 and the shared finite tail route exposed by chapter 560; the dependency is the displayed tail-window and located-enclosure route through which the witness becomes available to the Real seal.

The downstream BoundedMonotoneConvergenceSeal$^{\uparrow}$ consumer of chapter 918 reads this witness surface as its finite monotone source, regular rational handoff, stream schedule, Cauchy modulus, dyadic ledger, located trap, and Real$^{\uparrow}$ seal precursor. The convergence seal packages the public convergence handoff without selected limits, quotient equality, host real equality, trichotomy, or ambient completeness.

Theorem 535.1 (Bounded monotone witness root monotone-window totality)**.** *Every accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet exposes its bounded monotone source rows and the finite* StreamName$^{\uparrow}$ *windows before the Cauchy witness is read. The read is carried by the displayed* BHist *rows, transported only by* hsame*, routed only through* Cont*, packaged only by* Pkg*, and named only by the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root monotone window totality`

Proof. Project the accepted carrier described by the opening dependency surface. Its first visible rows are the bounded monotone source, the regular rational handoff, and the stream-window data shared with chapter 438 and chapter 560.

Use the sibling certificate surfaces for RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, and CauchyModulus$^{\uparrow}$. They make the finite window and modulus rows available through the same BHist, hsame, Cont, Pkg, and NameCert support before the witness-facing coordinate can be consumed.

The non-escape clauses listed for the chapter then block any read through selected limits, quotient stream equality, host real equality, trichotomy, or ambient completeness. Thus the witness route begins with the displayed monotone source and finite-window rows. □

Theorem 535.2 (Bounded monotone witness root located Real-seal route)**.** *Every accepted* BoundedMonotoneCauc *packet routes the witness to the* $\mathsf{Real}^{\uparrow}$ *seal through the located interval trap, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row. The route is a finite* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *route, not a selected-limit or host-completion route.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root located real seal route`

Proof. Start from the located-enclosure dependency named in the chapter opening. The located interval trap supplies the finite enclosure row, while the dyadic ledger and Cauchy modulus rows supply the tolerance and tail data read by the witness.

Apply the sibling surfaces for $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyModulus}^{\uparrow}$. They keep the enclosure, tolerance, and tail rows inside the same package provenance and local naming boundary before any $\mathsf{Real}^{\uparrow}$ seal row is reached.

Finally use the $\mathsf{Real}^{\uparrow}$ seal handoff already named by the bounded-monotone consumer paragraph. Since the packet exports no selected limit, quotient equality, host real equality, trichotomy, or ambient completeness coordinate, the seal route is exhausted by the displayed finite trap, ledger, modulus, transport, continuation, package, and name rows. □

Theorem 535.3 (Bounded monotone witness root completion-source factorization)**.** *Every* BoundedMonotoneConv *consumer of an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before it reads the completion-facing convergence handoff.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root completion source factorization`

Proof. Project the accepted carrier and list the source-facing rows. The bounded monotone source is exposed through the regular rational row, stream schedule, Cauchy modulus, dyadic ledger, located trap, and Real-seal precursor named in the dependency paragraphs.

Compose the monotone Cauchy surface of chapter 438 with the regular tail route of chapter 560. This identifies the source, window, tail, and located-seal rows as the finite route available to the convergence-seal consumer.

Close with the consumer boundary of chapter 918. Its read is a package read through the displayed BHist, hsame, Cont, Pkg, and NameCert rows, while selected limits, quotient equality, host real equality, trichotomy, and ambient completeness remain outside the packet. □

Theorem 535.4 (Bounded monotone Cauchy witness window-selector factorization)**.** *Every accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet consumed by a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *route factors its bounded monotone source through the same finite selector surface used by* FiniteObservationBudgetSe *Before the* $\mathsf{Real}^{\uparrow}$ *seal is readable, the route passes through finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the located trap, the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows.*

Thus the completion-facing read is exhausted by the displayed selector chain

$$\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization carries no quotient stream, selected limit, host real equality, trichotomy row, choice principle, or ambient completeness principle.

Proof. First expose the bounded monotone window rows by Definition 535.1. This places the source and finite $\mathsf{StreamName}^{\uparrow}$ windows inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support before any witness-consuming coordinate is read.

Next route the witness to the seal by Definition 535.2. The located trap, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{CauchyModulus}^{\uparrow}$ tail row give the finite enclosure and tail data needed by the seal-facing route.

Now apply Definition 535.3. It says that a $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ consumer reads the witness only through the displayed regular, stream, modulus, dyadic, located, Real, transport, continuation, provenance, and naming rows.

Finally compare this route with the terminal selector surface of Definition 782.44. The selector theorem exhausts every terminal read through its budget, schedule, finite window, dyadic, regular, seal, transport, continuation, provenance, and local naming rows; the bounded monotone witness route supplies the same window, ledger, handoff, trap, tail, and seal-facing rows. Since neither route exports quotient streams, selected limits, host real equality, trichotomy, choice, or ambient completeness, the convergence-seal consumer factors through the finite selector surface before the $\mathsf{Real}^{\uparrow}$ seal is readable. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessWindowSelectorCertifica`

Definition 535.5 (Bounded monotone Cauchy witness carrier)**.** *A* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

with the following visible coordinates:

1. *S is the bounded monotone source row, displaying the generator history, monotone comparison rows, and the finite upper-bound ledger for the source observations;*
2. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row extracted from S and read as the regular rational name exposed to completion-facing consumers;*
3. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that selects the inspected source and regular-sequence windows;*
4. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail witness, sending each dyadic tolerance request to a concrete finite tail window;*
5. *λ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the dyadic tolerance rows, the source upper-bound rows, and the monotone comparison checks consumed by μ;*
6. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap row recording the lower observation, upper guard, and tolerance enclosure read before the real seal;*
7. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed regular sequence, selected window, Cauchy witness, dyadic ledger, and located trap have been exposed;*
8. *H records componentwise* hsame *transports for the bounded source, regular sequence, stream schedule, Cauchy witness, dyadic ledger, located trap, and real-seal rows;*
9. *C records the displayed* Cont *routes by which consumers inspect those rows;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and source certificates;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row.*
The bounded monotone source row is displayed as

$$S_{\mathrm{gen}}(i) : \textit{the } i\textit{-th generated source observation},$$
$$S_{\mathrm{mono}}(i,j) : i \le j \Rightarrow S_{\mathrm{gen}}(i) \le S_{\mathrm{gen}}(j),$$
$$S_{\mathrm{ub}}(i) : S_{\mathrm{gen}}(i) \le U$$

for a dyadic upper guard U recorded in λ. The Cauchy witness is the explicit row

$$\mu(\varepsilon) : \textit{a scheduled tail window on which } F \textit{ has diameter below } \varepsilon.$$

The carrier contains no choice-selected limit point, quotient-name equality, host real equality, unrestricted real-history range, ambient completeness theorem, trichotomy row, or open-cover compactness row.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier route local name closure`

Theorem 535.6 (Bounded monotone Cauchy witness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCa}}$ *surface consists exactly of five rows:*

1. *carrier admission for the displayed packet of Definition 535.5;*
2. *generator range discipline, stating that every source observation read by a consumer is routed through S_{gen} and the bounded monotone ledgers of S;*
3. *explicit Cauchy witness discipline, stating that every tail request is answered by the row μ and the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window σ;*
4. *located trap and* $\mathsf{Real}^{\uparrow}$ *seal handoff, stating that the real-seal row Q is reached only through $F, \sigma, \mu, \lambda, I, H, C, P, N$;*
5. *ledger exhaustion, stating that all public reads of bounded monotone convergence are exhausted by the displayed generator, witness, classifier transport, package provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier namecert obligations`

Proof. Carrier admission is projection of the finite packet in Definition 535.5. The packet lists the bounded source row, regular rational row, stream schedule, Cauchy witness, dyadic ledger, located trap, real seal, transports, continuations, package provenance, and local naming row.

Generator range discipline uses the displayed row S_{gen} and its monotone and upper-bound ledgers. Type I range pressure is discharged by that displayed generator, because no consumer ranges over an unrestricted intended class of real histories.

Explicit Cauchy witness discipline uses μ and σ to answer each tolerance request with a concrete tail window. Type II selection pressure is discharged by this visible Cauchy witness, not by choosing a limit point from nonempty completeness data.

The seal handoff and ledger exhaustion then use I, Q, H, C, P, and N. Type III collapse pressure is discharged by $\mathsf{Real}^{\uparrow}$ classifier transport and componentwise hsame rows, not by host equality or quotient-name identification. □

Theorem 535.7 (Bounded monotone Cauchy witness generator range exactness)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every source observation exported from* W *is exactly a displayed generator read through* S_{gen}*, with its monotonicity and upper-bound certificates read through* S_{mono} *and* S_{ub} *and with dyadic upper guard data read through* λ*. No consumer may replace this generator range by an unrestricted real-history range, a selected limit, trichotomy data, quotient equality, or ambient completeness.*

Proof. Project the source row S from the carrier. The definition displays S_{gen}, S_{mono}, and S_{ub} as the generator, monotonicity, and upper-bound ledgers, with the dyadic guard recorded in λ.

Use the generator range discipline row of Definition 535.6. It requires every source observation read by a consumer to pass through those displayed source ledgers.

Finally replay the transport and route rows H and C and attach the provenance rows P and N. Since the packet has no coordinate for unrestricted real-history range, selected limit, trichotomy, quotient equality, or ambient completeness, the displayed generator range is exact. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier generator range exactness`

Theorem 535.8 (Bounded monotone Cauchy witness tail modulus scope)**.** *For an accepted* BoundedMonotoneCauc *carrier* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, every tail request is scoped by the Cauchy witness row* μ*, the scheduled StreamName windows* σ*, and the dyadic tolerance ledger* λ*. The request may inspect* S_{gen}*,* S_{mono}*,* S_{ub}*, and the regular-sequence row* F*, but it cannot use an existential completeness theorem, a selected limiting representative, quotient equality, host real equality, or trichotomy.*

Proof. Read the tail witness from the carrier. The row μ answers tolerance requests, σ chooses the inspected windows, and λ records the dyadic tolerance and upper-bound ledgers consumed by the witness.

Apply the explicit Cauchy witness discipline from Definition 535.6. It sends every tail request through μ and σ rather than through an external completion assertion.

The source rows S_{gen}, S_{mono}, S_{ub} and the regular-sequence row F supply the displayed observations. Transport through H, continuation through C, and provenance through P, N do not add a selected limit, quotient equality, host equality, ambient completeness, or trichotomy coordinate. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail modulus scope`

Theorem 535.9 (Bounded monotone Cauchy witness Real seal boundary)**.** *For an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read through* Q *is routed through* $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *and remains tied to the displayed bounded source row* S*. The boundary exports no selected limit, no trichotomy row, no quotient equality, no host real equality, and no ambient completeness principle.*

Proof. Project the seal row Q and the pre-seal rows $F, \sigma, \mu, \lambda, I$ from the carrier. The real-facing consumer reaches Q only after the regular sequence, scheduled windows, tail witness, dyadic ledger, and located trap have been displayed.

Apply the located trap and $\mathsf{Real}^{\uparrow}$ seal handoff row of Definition 535.6. It binds the seal to $F, \sigma, \mu, \lambda, I, H, C, P, N$ and keeps the source row S visible through the generator range discipline.

Use the tail modulus scope and generator range exactness rows to rule out alternate inputs. The remaining transport, continuation, package, and local naming rows replay the same finite packet, so no selected limit, trichotomy, quotient equality, host real equality, or ambient completeness coordinate can cross the seal boundary. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal boundary`

Theorem 535.10 (Bounded monotone Cauchy witness socket separation)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The three analysis sockets in bounded monotone convergence are separated inside* W*: Type I range pressure is read only through the displayed generator row* S*, Type II selection pressure is read only through the explicit Cauchy witness* μ *with the scheduled windows* σ*, and Type III collapse pressure is read only through* $\mathsf{Real}^{\uparrow}$ *classifier transport and the componentwise rows in* H *and* C*.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier socket separation`

Proof. For the range site, inspect the bounded monotone source row. The generator S_{gen}, monotone ledger S_{mono}, and upper-bound ledger S_{ub} are carrier coordinates, so source reads have a displayed finite route before any regular-sequence or real-seal row is consumed.

For the selection site, inspect the Cauchy witness row. A tolerance request is routed to $\mu(\varepsilon)$ and then through the scheduled window $\sigma(\mu(\varepsilon))$, so the tail witness is present as data rather than selected from an existential completion statement.

For the collapse site, compare accepted packets only through the transported regular sequence, located trap, and real-seal classifier rows. The hsame transports and Cont routes preserve the displayed rows, while the $\mathsf{Real}^{\uparrow}$ seal is reached through classifier transport rather than host equality. □

Theorem 535.11 (Bounded monotone Cauchy witness classifier transport)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

and

$$W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$$

be accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carriers. If the displayed classifier relates* W *to* W'*, then the replacement transports the packet componentwise:*

1. *bounded source rows* S *to* S'*;*
2. $\mathsf{RegSeqRat}^{\uparrow}$ *rows* F *to* F' *and* $\mathsf{StreamName}^{\uparrow}$ *schedules* σ *to* σ'*;*
3. $\mathsf{CauchyModulus}^{\uparrow}$ *witnesses* μ *to* μ' *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers* λ *to* λ'*;*
4. $\mathsf{LocatedInterval}^{\uparrow}$ *traps* I *to* I' *and* $\mathsf{Real}^{\uparrow}$ *seals* Q *to* Q'*;*
5. hsame *rows* H *to* H'*,* Cont *routes* C *to* C'*,* Pkg *provenance* P *to* P'*, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}}$ *rows* N *to* N'*.*

No quotient real equality, selected limit, ambient completeness theorem, trichotomy row, or open-cover compactness coordinate can enter the transported packet.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier classifier transport`

Proof. Project the two accepted carriers. The source, regular sequence, stream schedule, Cauchy witness, dyadic ledger, located trap, real seal, transport, continuation, provenance, and local naming rows are exactly the displayed coordinates of Definition 535.5.

Invoke the five rows of Definition 535.6. Carrier admission fixes the packet shape, generator range discipline keeps source observations inside S_{gen} and S'_{gen}, explicit Cauchy witness discipline keeps tolerance requests inside μ, σ and μ', σ', the seal handoff keeps real-facing reads routed through the located trap and $\mathsf{Real}^{\uparrow}$ seal, and ledger exhaustion lists the remaining public routes.

Finally use Definition 535.10. Type I range, Type II selection, and Type III collapse sites occupy disjoint carrier sockets, so classifier replacement can only replay the listed componentwise rows. The excluded completion objects are absent from the carrier definition and from the NameCert rows, hence there is no coordinate through which they could be transported into the accepted packet. □

Theorem 535.12 (Bounded monotone Cauchy witness Real seal handoff)**.** *For an accepted* BoundedMonotoneCauc *carrier* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read factors through the displayed bounded source row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail witness, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and upper-bound ledger, and the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap before reaching* Q*.*

The handoff exports a witnessed monotone Cauchy packet, not a choice-selected limiting representative, quotient equality, host real equality, ambient completeness, trichotomy, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal handoff`

Proof. Project the carrier. The source and regular-sequence rows S and F display the observations available to the real-facing consumer, and σ records the finite windows on which those observations are inspected.

Next use the Cauchy witness row μ together with the dyadic ledger λ. Each tolerance request reaches a displayed tail window with its monotone comparison and upper-bound data before any located trap is read.

The located trap I is the final enclosure row before the seal. Transport through H and continuation through C replay only those displayed rows, with P and N recording provenance and local naming.

Thus the seal coordinate Q receives exactly the witnessed packet described above. None of the excluded completion objects is present as a carrier coordinate or as a handoff route. □

Theorem 535.13 (Bounded monotone Cauchy witness ledger exhaustion)**.** *Every public read of an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier is exhausted by the bounded monotone source rows, the regular-sequence row, the stream windows, the Cauchy witness row, the dyadic tolerance and upper-bound ledger, the located trap, the* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transports,* Cont *routes,* Pkg *provenance, and the local* NameCert *row of Definition 535.5.*

Proof. Inspect the carrier coordinates. The source observations, monotone comparisons, upper-bound rows, regular rational name, window schedule, Cauchy witness, dyadic ledger, located trap, and real-seal row are all fields of the same finite BHist packet.

The transport and continuation fields account for the only permitted movement of those rows. The package and local naming fields record provenance and certificate membership, but they do not add analytic data.

Consequently every public read has already appeared in the displayed ledger. The packet has no remaining coordinate for a hidden range over all real histories, an unwitnessed existential limit choice, quotient-name collapse, host real equality, ambient completeness, trichotomy, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier ledger exhaustion`

Theorem 535.14 (Bounded monotone Cauchy witness obligation closure)**.** *The carrier of Definition 535.5 and the local theorems of this chapter assemble the* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *obligations for displayed bounded-source range, explicit Cauchy witness,* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{LocatedInterval}^{\uparrow}$ *trap,* $\mathsf{Real}^{\uparrow}$ *seal handoff, classifier transport, and non-escape.*

The resulting packet admits only the rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$*. It has no coordinate for selected limits, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness.*

Proof. Begin with the carrier and the NameCert obligation theorem. They provide carrier admission and split the public surface into generator range discipline, explicit Cauchy witness discipline, located trap with $\mathsf{Real}^{\uparrow}$ handoff, and ledger exhaustion.

For the bounded-source and Cauchy-witness parts, apply Definition 535.7 and Definition 535.8. These force source reads through $S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$ and force tolerance requests through μ, σ, λ.

For the seal and transport parts, apply Definition 535.9, Definition 535.10, Definition 535.11, and Definition 535.12. These route the located trap and real seal through the displayed rows and keep the three analysis sockets separate.

Finally apply Definition 535.13. Since every public read is already exhausted by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, the excluded completion and compactness coordinates cannot be projected from the accepted packet. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier obligation closure`

Theorem 535.15 (Bounded monotone Cauchy witness convergence-seal factorization)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any completion-facing consumer using a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet whose witness coordinate is* W *must first factor through the accepted witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and then through the convergence-seal rows

$$W, M, K, F', \sigma', D, L, R, H', C', P', N'.$$

Thus the completion-facing route is a two-stage BEDC package. The first stage is the local bounded monotone Cauchy witness certificate, and the second stage is the convergence-seal certificate consuming that witness row. No consumer can skip directly from the witness coordinate to a completion object, selected limit, quotient stream equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or detached tail oracle.

Proof. Start with the witness carrier. By Definition 535.5, the accepted witness packet exposes exactly $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as its public rows.

The local NameCert obligations and closure theorem bind those rows before the witness can be exported. Apply Definition 535.6 and Definition 535.14; they force source reads, window requests, Cauchy witness reads, dyadic ledgers, located traps, seal reads, transports, continuations, provenance, and local naming to stay inside the displayed witness packet. The Real seal handoff of Definition 535.12 supplies the last local route to Q.

Now read the convergence-seal packet. Definition 918.4 makes the witness row W the first consumed row before the seal packet may read $M, K, F', \sigma', D, L, R, H', C', P', N'$. Definition 918.5 then routes tolerance-facing reads through the monotone-tail and Cauchy-criterion rows before the finite regular, stream, dyadic, and CauchyLimitSeal rows are reached.

Finally apply Definition 918.12. It packages the second-stage rows as the displayed convergence-seal obligation surface. Since both stages have exhausted their public ledgers, the combined route has no remaining coordinate for any excluded completion, equality, order, compactness, or choice-selected tail. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal factorization`

Theorem 535.16 (Bounded monotone Cauchy witness shared budget meet)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose* W *is consumed by a* $\mathsf{CauchySealBudgetSynchronizer}$ *budget handoff and by either a tail-schedule route or a real-tail-agreement route. Then every* $\mathsf{Real}^{\uparrow}$ *seal read on the combined route factors through the same finite rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

together with the synchronizer budget row before any downstream seal is readable.

Consequently the witness row, synchronizer row, and tail schedule share one finite budget route. The route exports no second tail budget, selected limit, quotient-stream equality, host real equality, ambient completeness, trichotomy, or hidden tail oracle.

Proof. First use the local convergence-seal factorization above. It forces every convergence-facing read to consume the bounded monotone Cauchy witness rows before the downstream seal rows are available, so the witness cannot be replaced by a completion object or by an independently selected tail.

Next attach the schedule and agreement handoffs. The tail-window service and Real observation-budget handoff force tolerance requests and finite Real-facing reads through $S, F, \sigma, \mu, \lambda$ before the located trap and $\mathsf{Real}^{\uparrow}$ seal are reached. The root synchronizer handoff then places every $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ consumer on the same displayed route through $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ rather than on a separate budget coordinate.

The sibling root-tail and completion routes keep the two consumers aligned. Root tail-budget exactness binds the $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, and $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ rows to the same source, tail, trap, seal, transport, continuation, provenance, and naming rows. Completion-consumer non-escape and the public seal handoff then send each Real-facing continuation through the public envelope, finite observation route, regular handoff, located trap, and seal row of W.

Finally apply ledger exhaustion. Since the accepted packet has no public coordinate beyond $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and the displayed synchronizer budget row, a downstream seal read has no second route from which it could obtain another tail budget, selected limit, quotient-stream equality, host real equality, ambient completeness, trichotomy split, or hidden tail oracle. □

Lean4 ✓ BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier shared budget meet

Theorem 535.17 (Bounded monotone Cauchy witness minimal convergence-seal witness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer use W as its witness coordinate. The local admission and consumption claim is already witnessed by the finite two-stage surface*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

followed by

$$W, M, K, F', \sigma', D, L, R, H', C', P', N'.$$

Equivalently, the named witness row W, together with the shared budget route of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, is sufficient for every consumer-facing seal read admitted by this chapter.

If a larger reading claims to consume the same bounded monotone Cauchy witness while also exposing an additional completion object, selected limit, quotient stream equality, host real equality, trichotomy row, ambient completeness row, open-cover compactness row, detached tail oracle, or hidden selector, then its consumer-facing part factors back through the displayed two-stage surface and the added coordinate is not part of the admitted witness. Thus the minimal witness is the displayed finite witness-and-seal packet, not an ambient choice object.

Proof. First read the local witness stage. By Definition 535.15, a completion-facing consumer using a convergence-seal packet with witness coordinate W must pass through the accepted witness rows

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

Those rows are not merely parameters of a later construction: they are the public source, schedule, modulus, dyadic ledger, located trap, Real seal, transport, continuation, provenance, and local naming rows through which the witness is admitted.

The same factorization then forces the second stage through the convergence-seal rows

$$W, M, K, F', \sigma', D, L, R, H', C', P', N'.$$

At that point the consumer has exactly the witness coordinate W, the public monotone and Cauchy rows, the regular-rational readback, stream and dyadic ledgers, Cauchy-limit compatibility boundary, Real seal, transports, continuations, provenance, and local name. This proves sufficiency: the local admission and consumption claim has a displayed finite route before any consumer-facing seal read is available.

It remains to rule out enlargement of the witness. Apply Definition 535.16. Whenever the same W is consumed by the seal-budget synchronizer and by a tail-schedule or real-tail-agreement route, every $\mathsf{Real}^{\uparrow}$ seal read factors through the same finite rows of W together with the synchronizer budget row. Hence a downstream consumer cannot split off a second tail budget, a detached selector, or an independent completion-facing coordinate while still claiming to consume the same admitted witness.

Combining the two statements gives the minimality claim. The factorization theorem lists the only two stages through which the convergence-seal consumer may read, and the shared-budget meet theorem keeps all Real-facing consumption on the same finite budget surface. Any larger reading therefore has its admitted consumer-facing part already factored through the displayed rows. The listed excluded objects are not rows of either displayed stage and are not supplied by the shared budget meet, so they cannot enlarge the local witness. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier minimal convergence seal witness`

Theorem 535.18 (Bounded monotone Cauchy witness convergence-seal selector synchronization)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer using W synchronizes its selected tail window through the witness rows μ, σ, λ and the seal-facing rows before it reads the convergence-seal selector.*

Concretely, a tolerance request is first answered by μ on the scheduled window σ and recorded in the dyadic ledger λ. The resulting window is then routed through I, Q, H, C, P, N and only then through the convergence-seal rows

$$W, M, K, F', \sigma', D, L, R, H', C', P', N'.$$

Thus the selected limit, quotient equality, host real equality, ambient completeness, and detached tail-oracle readings are unavailable unless they are already exposed by the displayed witness and seal packets.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal selector synchronization`

Proof. Begin with the local carrier. Definition 535.5 exposes exactly $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the consumer has no selector coordinate outside those rows.

Next apply Definition 535.8. It sends each tolerance request to μ, ties the answer to the scheduled windows σ, and records the requested dyadic tolerance in λ.

Now route the synchronized tail into the seal-facing side. By Definition 535.12, the read of Q passes through the regular-sequence row, the window schedule, the tail modulus, the dyadic ledger, the located trap, transport, continuation, provenance, and local naming rows.

Finally use Definition 535.15. The convergence-seal consumer reads the witness packet before it reads $M, K, F', \sigma', D, L, R, H', C', P', N'$. Consequently the selected window is synchronized across μ, σ, λ and the seal rows before the convergence-seal selector is available, and the excluded completion and equality coordinates remain outside the combined route. □

Theorem 535.19 (Bounded monotone Cauchy witness scoped dependency package)**.** *The scoped dependency package of Definition 535.5 consists exactly of the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Within this package, generator range reads are controlled by S, tail requests are controlled by μ, σ, λ, located enclosures are controlled by I, classifier transport is controlled by H, C, the real seal is controlled by Q, and provenance and naming are controlled by P, N. Selected limits, quotient equality, host real equality, trichotomy, ambient completeness, and open-cover compactness remain outside the scoped package.

Proof. Start with generator range exactness. By Definition 535.7, every source observation is read through S_{gen} together with the monotone and upper-bound ledgers of S and the dyadic guard data in λ.

Next use tail modulus scope. By Definition 535.8, every tolerance request is answered by μ on the scheduled windows σ with the displayed dyadic tolerance ledger λ.

For the seal-facing rows, apply Definition 535.9 and Definition 535.12. These route every $\mathsf{Real}^{\uparrow}$ read through $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ while keeping the bounded source row visible.

Classifier movement is supplied by Definition 535.11. It transports only the displayed carrier coordinates and does not add any coordinate for selected limits, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness.

Finally apply Definition 535.13. Every public read is exhausted by the listed carrier rows, so the scoped dependency package is exactly the displayed package and no excluded completion or compactness object belongs to it. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier scoped dependency package`

Theorem 535.20 (Bounded monotone Cauchy witness L10 exit-consumer package)**.** *Any Phase Real and Completion consumer of an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet must read the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{CauchyModulus}^{\uparrow}$ *tail witness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{LocatedInterval}^{\uparrow}$ *trap,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *carrier,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before exporting a real-facing convergence claim.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier l10 exit consumer package`

Proof. Apply generator range exactness and tail modulus scope. These force the source reads through $S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$ and force each tolerance request through μ, σ, λ.

Next apply the real-seal boundary and real-seal handoff. The located trap and $\mathsf{Real}^{\uparrow}$ seal are read only through the displayed I, Q, H, C, P, N rows.

Finally apply classifier transport, ledger exhaustion, and the scoped dependency package. The consumer receives exactly the displayed witness packet before any real-facing convergence export, with no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, or compactness coordinate added at the exit. □

Theorem 535.21 (Bounded monotone Cauchy witness source-to-seal triad)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing bounded monotone witness read from* W *factors through exactly three stages:*

1. *the bounded source generator rows* $S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$ *over the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *interface;*
2. *the scheduled finite tail-window service* σ, μ, λ *over the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *interface;*
3. *the located* $\mathsf{Real}^{\uparrow}$ *seal route* I, Q, H, C, P, N *through the displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows.*

No selected limit, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness row enters between these stages or supplies an additional $\mathsf{Real}^{\uparrow}$*-facing route.*

Proof. Start with the bounded source. By Definition 535.7, every generator-facing observation is already confined to $S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$ together with the displayed regular-sequence, stream-name, and dyadic guard data.

Next pass to the finite tail service. Definition 535.8 sends each tolerance request through μ on the windows scheduled by σ and records the tolerance ledger in λ.

For the seal-facing route, use Definition 535.12. It routes every $\mathsf{Real}^{\uparrow}$ read through the located interval row I, the seal row Q, the transport row H, the continuation row C, provenance P, and the local naming row N.

Now compare the three pieces with Definition 535.19. That package lists $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as the whole public surface of the accepted carrier, with the source rows, scheduled tail service, and located seal route as the only BEDC packet stages visible to consumers.

Finally apply Definition 535.13. Since the public ledger is exhausted by the displayed rows, no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness row can enter between the three stages or provide a further seal route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessSourceToSealTriad`

Theorem 535.22 (Bounded monotone Cauchy witness source row scope)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every source read exposed by* W *is read through the generator, monotonicity, and upper-bound rows*

$$S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$$

together with the dyadic guard data carried by λ*.*

The source interface exposes no selected limit, quotient equality, host real equality, ambient completeness coordinate, trichotomy split, open-cover compactness row, or choice-selected tail.

Proof. Start with generator range exactness. By Definition 535.7, any source observation available to a consumer is already confined to S_{gen}, S_{mono}, and S_{ub}, with the tolerance guard supplied by λ.

Next compare this row with the scoped dependency package. Definition 535.19 lists S as the only bounded-source coordinate of the accepted carrier, while the remaining rows $F, \sigma, \mu, I, Q, H, C, P, N$ are handoff, witness, seal, transport, route, provenance, and naming rows rather than additional source coordinates.

Finally apply classifier transport and ledger exhaustion. Definition 535.11 moves only displayed coordinates, and Definition 535.13 leaves no public row from which a selected limit, host equality, quotient collapse, ambient completeness object, trichotomy split, compactness principle, or choice-selected tail could be read. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier source row scope`

Theorem 535.23 (Bounded monotone Cauchy witness window modulus row scope)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every tolerance request made to* W *factors through the displayed window schedule* σ*, Cauchy modulus answer* μ*, and dyadic tolerance ledger* λ *before any located trap* I *or* $\mathsf{Real}^{\uparrow}$ *seal row* Q *is read.*

Thus the window-modulus interface is the finite service σ, μ, λ over the displayed source and regular handoff rows. It exports no located trap or Real seal read that bypasses that service, and it exposes no selected limit, quotient equality, host real equality, ambient completeness, trichotomy, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window modulus row scope`

Proof. Begin with tail modulus scope. Definition 535.8 sends each tolerance request through μ on the scheduled windows σ and records the dyadic tolerance data in λ.

Use the scoped dependency package to place this service inside the whole packet. It identifies $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as the public surface, so the window-modulus rows have to be read before the later located-trap and seal-facing rows can be consumed.

Now apply the Real seal boundary and Real seal handoff. Definition 535.9 and Definition 535.12 route I and Q through the displayed witness rows rather than around them.

Ledger exhaustion gives the exclusion. Since Definition 535.13 exhausts public reads by the displayed carrier rows, no completion coordinate, quotient collapse, host equality, trichotomy split, compactness principle, or choice-selected tail can answer a tolerance request independently of σ, μ, λ. □

Theorem 535.24 (Bounded monotone Cauchy witness Real seal row scope)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *seal read through Q factors through the displayed route*

$$F, \sigma, \mu, \lambda, I, H, C, P, N$$

and remains tied to the bounded source row S.

The Real seal interface therefore exports only the witnessed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. It does not export a selected limit, quotient real equality, host real equality, ambient completeness, trichotomy, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal row scope`

Proof. Start with the seal boundary and handoff rows. By Definition 535.9 and Definition 535.12, the seal coordinate Q is available only through the regular handoff, scheduled windows, Cauchy witness, dyadic ledger, located trap, classifier transport, continuation, provenance, and local naming rows.

The source row remains attached to this route. Generator range exactness keeps source observations inside $S_{\text{gen}}, S_{\text{mono}}, S_{\text{ub}}$, and tail modulus scope keeps the tolerance route inside σ, μ, λ before the seal-facing suffix is read.

Classifier transport supplies the only allowed movement of the seal packet. Definition 535.11 transports the displayed rows componentwise, so the Real seal cannot be replaced by a host equality or quotient-name collapse.

Finally apply ledger exhaustion. The accepted carrier has no public coordinate beyond $S, F, \sigma, \mu, \lambda, I, Q, H, C, P$, so every seal read through Q remains a read of the displayed witnessed packet and cannot import completeness, trichotomy, compactness, selected limits, or choice-selected tails. □

Theorem 535.25 (Bounded monotone Cauchy witness tail-window service)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every tolerance request made to* W *factors through the bounded source row* S*, the regular rational handoff* F*, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows* σ*, the Cauchy modulus answer* μ*, and the dyadic tolerance ledger* λ *before any located trap or completion-facing row can be read.*

The service exposes exactly the displayed $\mathsf{StreamName}^{\uparrow}$ *windows and their dyadic tolerance rows. It does not supply a selected limit, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness.*

Proof. Project the accepted carrier and use the NameCert obligation theorem. The rows $S, F, \sigma, \mu, \lambda$ are part of the displayed packet, so a tolerance request has no independent input surface outside the carrier.

Apply Definition 535.7. It keeps source observations inside the bounded generator, monotone, and upper-bound rows of S together with the regular rational handoff F.

Next apply Definition 535.8. The requested tolerance is answered by μ only after the scheduled windows σ and the dyadic ledger λ have been displayed.

Finally use Definition 535.13. Since every tolerance read has already passed through $S, F, \sigma, \mu, \lambda$, no completion coordinate or compactness object remains available to the service. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail window service`

Theorem 535.26 (Bounded monotone Cauchy witness located trap refinement)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The located trap row* I *refines the bounded source row* S*, the dyadic ledger* λ*, and the* $\mathsf{Real}^{\uparrow}$ *seal route* Q, H, C, P, N *through the same displayed packet. It does not add a selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.*

The refinement is local to the carrier: a trap consumer may inspect I *only after the generator range, tail-modulus, dyadic tolerance, and NameCert rows of* W *have fixed the route from* $S, F, \sigma, \mu, \lambda$ *toward* Q*.*

Proof. Project the accepted carrier using Definition 535.5. The trap row I is displayed beside the bounded source, regular rational handoff, scheduled windows, Cauchy witness, dyadic ledger, seal, transports, continuations, provenance, and local naming row.

Use the NameCert obligations of Definition 535.6. They make carrier admission, explicit witness access, trap access, seal access, classifier transport, and ledger non-escape part of the same local package.

Now apply generator range exactness and tail modulus scope. By Definition 535.7, the source read remains inside the bounded generator and upper-bound rows of S; by Definition 535.8, the tolerance request reaches I only after σ, μ, λ have supplied the displayed tail service.

Finally use Definition 535.9, Definition 535.11, and Definition 535.13. The trap is therefore a refinement of the bounded source and dyadic ledger on the path to the Real seal, and every excluded completion or compactness coordinate remains absent from the carrier. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier located trap refinement`

Theorem 535.27 (Bounded monotone Cauchy witness tail-seal synchronization)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every tail request answered by the scheduled windows* σ*, Cauchy witness* μ*, and dyadic ledger* λ *reaches the same* $\mathsf{Real}^{\uparrow}$ *seal row* Q *through the displayed route*

$$I, H, C, P, N.$$

The synchronization supplies one tail-to-seal path through $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *and cannot create a second hidden tail service or an ambient completion read.*

Proof. Begin with the tail-window service of Definition 535.25. It sends each tolerance request through $S, F, \sigma, \mu, \lambda$ before any located trap or completion-facing row can be consumed.

The carrier and NameCert rows then identify the seal-facing suffix. By Definition 535.5 and Definition 535.6, the only rows downstream of the tail service are the located trap I, the seal Q, the transports H, continuations C, provenance P, and naming row N.

Use generator range exactness and tail modulus scope to keep the source and tolerance sides fixed. Definition 535.7 fixes the bounded source read, while Definition 535.8 fixes the σ, μ, λ answer that the seal-facing route receives.

Finally apply Definition 535.9, Definition 535.11, and Definition 535.13. They force the read of Q through the same I, H, C, P, N suffix and leave no carrier coordinate for a second tail source, selected limit, quotient equality, host equality, trichotomy, ambient completeness, or compactness principle. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail seal synchronization`

Theorem 535.28 (Bounded monotone Cauchy witness completion-consumer refusal)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer receives only the witnessed packet displayed by* W*: the bounded source* S*, regular rational row* F*, window schedule* σ*, witness* μ*, dyadic ledger* λ*, located trap* I*, seal* Q*, transports* H*, continuations* C*, provenance* P*, and local naming row* N*.*

The consumer cannot read ambient completeness, quotient equality, trichotomy, open-cover compactness, host real equality, selected limiting representatives, or a choice-selected tail from this packet.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion consumer refusal`

Proof. Project the consumer-facing surface from the carrier of Definition 535.5. The displayed tuple $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ is the whole accepted packet.

Apply the NameCert obligations and the generator range row. Definition 535.6 identifies the admissible read coordinates, and Definition 535.7 keeps source observations inside the bounded generator and upper-bound data.

Next use Definition 535.8, Definition 535.25, and Definition 535.26. The consumer therefore reaches the seal only after the scheduled witness, dyadic ledger, and located trap rows have been read.

The final refusal is ledger exhaustion. By Definition 535.9, Definition 535.11, and Definition 535.13, every completion-facing read stays inside the witnessed packet and the excluded completion, equality, order, compactness, and host-real principles have no row to project. □

Theorem 535.29 (Bounded monotone Cauchy witness located trap handoff)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The* $\mathsf{LocatedInterval}^{\uparrow}$ *trap row I is reached only after the dyadic tolerance ledger λ and the source upper-bound rows in S have been displayed through the scheduled tail-window service.*

Thus any handoff from W to a located interval consumer preserves the same $S, F, \sigma, \mu, \lambda$ route before I and carries forward only Q, H, C, P, N as the seal, transport, continuation, provenance, and local naming rows.

Proof. Start with the source rows. By Definition 535.7, the upper-bound data are read from the displayed bounded source packet S and not from an ambient order-completeness assumption.

Use the tail-window service supplied by Definition 535.25. It routes the tolerance request through F, σ, μ, λ, so the dyadic ledger is already visible before the trap row is consumed.

Now apply Definition 535.9. The located trap I belongs to the seal-facing route that also carries Q, H, C, P, N, and it cannot be read before the source and tolerance ledgers have been displayed.

Finally apply Definition 535.13. The handoff has no coordinate for a hidden trap, selected limit, quotient equality, host real equality, trichotomy, ambient completeness, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier located trap handoff`

Theorem 535.30 (Bounded monotone Cauchy witness completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier reads only the displayed packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The boundary supplies the bounded source rows, regular rational handoff, scheduled tail windows, modulus answer, dyadic tolerance ledger, located trap, Real seal, transport rows, continuation rows, package provenance, and local naming row. It supplies no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion boundary`

Proof. Use the scoped dependency package. By Definition 535.19, the whole public surface of the accepted carrier is the displayed package over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 535.25 and Definition 535.29. These rows force completion-facing reads to pass through the source, window, modulus, dyadic tolerance, and located-trap route before the seal is available.

Next apply Definition 535.12. It sends every real-facing read through I, Q, H, C, P, N while keeping the source and witness rows visible.

Finally apply Definition 535.13. Since the packet is exhausted by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, the excluded completion and equality principles cannot be projected by any consumer. □

Theorem 535.31 (Bounded monotone witness bridge to public tail and criterion packets)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Suppose the bounded source row S is read through the finite tail-window surface of chapter 438, and the witness rows σ, μ, λ supply the explicit modulus and tail-bound route of chapter 392. Then every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of W factors through the public* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail-window certificate and the public* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet before the local* $\mathsf{Real}^{\uparrow}$ *seal handoff.*

The factorization preserves only the displayed rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. It exports no quotient real equality, selected representative, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice principle.

Proof. Start from Definition 535.19. It exposes the bounded source row, regular rational row, stream schedule, Cauchy modulus witness, dyadic ledger, located trap, real seal, transports, continuations, package provenance, and local naming row as the whole dependency surface of W.

Read the bounded source through the finite route supplied by Definition 438.21. That public certificate consumes the displayed tail-window schedule, monotone rows, located interval trap, real-seal row, componentwise transports, continuation routes, package provenance, and local $\mathsf{NameCert}_{\mathsf{MonotoneCauchy}^{\uparrow}}$ row, so the source socket of W is routed through the existing $\mathsf{MonotoneCauchy}^{\uparrow}$ public surface.

Read the witness rows through Definition 392.20. The rows σ, μ, λ provide the explicit modulus, dyadic tolerance, finite tail ledger, regular rational handoff, and real-seal route required by the public $\mathsf{CauchyCriterion}^{\uparrow}$ packet, so the witness socket of W is routed through the existing constructive criterion surface.

Finally apply Definition 535.12 and Definition 535.13. The handoff sends each real-facing read through $F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ while keeping the bounded source row visible, and ledger exhaustion rules out every remaining coordinate. Therefore the composed route passes through the MonotoneCauchy public tail-window certificate and the CauchyCriterion public packet before the local seal, with no route to quotient real equality, selected representatives, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier monotone cauchy criterion bridge`

Theorem 535.32 (Bounded monotone Cauchy witness Real observation-budget handoff)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. If a real-facing consumer asks W for finite observation data, then the request factors through the finite observation route of chapter 611: the windows are read through σ, dyadic readback and radius data are read through λ, the regular rational handoff is read through F, and the* $\mathsf{Real}^{\uparrow}$ *seal is read through Q.*

The handoff preserves the displayed bounded source row S, the Cauchy witness μ, the located trap I, transports H, continuations C, provenance P, and the local naming row N. It exports no quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real observation budget handoff`

Proof. Start from the scoped dependency package of Definition 535.19. It lists $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ as the whole public surface of the accepted packet.

Read the finite observation request through chapter 611. That route accepts finite window rows, dyadic readback and radius ledgers, regular-rational handoff rows, and a $\mathsf{Real}^{\uparrow}$ seal boundary; these are supplied here by σ, λ, F, Q.

Finally apply Definition 535.12 and Definition 535.13. The seal remains tied to the displayed bounded source and witness rows, while ledger exhaustion rules out every absent completion coordinate. □

Theorem 535.33 (Bounded monotone Cauchy witness Cauchy modulus-extraction factorization)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Before W exports a tail threshold to any Real-completion route, the witness rows σ, μ, λ factor through the modulus-indexed schedule of chapter 641. The schedule reads σ as its finite window source, μ as the tail-threshold answer for each dyadic tolerance request, and λ as the dyadic tolerance and radius ledger that justifies the answer.*

The factorization may pass along the regular sequence row F, the source row S, and the seal-facing rows I, Q, H, C, P, N, but it cannot replace the displayed modulus extraction by quotient stream equality, host real equality, a selected limit, trichotomy, ambient completeness, or a choice-selected tail.

Proof. Project the witness rows from the carrier. The rows σ, μ, λ already describe a window schedule, a tolerance-indexed tail answer, and the dyadic ledger that the tail answer consumes.

Route those rows through chapter 641. Its modulus-indexed schedule is exactly the public surface that receives a finite window source, a requested tolerance, and the ledger rows needed to produce a tail threshold.

Then reattach the source, regular-sequence, located-trap, seal, transport, continuation, provenance, and naming rows by Definition 535.8 and Definition 535.9. These rows carry the extracted threshold toward the Real-completion route without adding any excluded completion coordinate. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier cauchy modulus extraction factorization`

Theorem 535.34 (Bounded monotone Cauchy witness downstream unblock package)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The downstream Real and Completion consumers of W receive a finite package through four public routes: the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail-window certificate, the* $\mathsf{CauchyCriterion}^{\uparrow}$ *criterion packet, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *finite observation route, and the* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *modulus-indexed schedule.*

The package consists only of the displayed rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. It gives downstream consumers a bounded-source route, an explicit Cauchy witness route, a finite observation route, and a modulus-extraction route, with no quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, or choice-selected tail.

Proof. First apply Definition 535.31. It routes the source and witness sides of W through the public $\mathsf{MonotoneCauchy}^{\uparrow}$ tail-window certificate and the public $\mathsf{CauchyCriterion}^{\uparrow}$ packet before the local $\mathsf{Real}^{\uparrow}$ seal handoff.

Next apply Definition 535.32. It routes every real-facing finite observation read through σ, λ, F, Q and keeps the remaining carrier rows visible as the same displayed packet.

Then apply Definition 535.33. It routes σ, μ, λ through the modulus-indexed schedule before any tail threshold is exported to the Real-completion route.

Finally use Definition 535.13. Since every public read is exhausted by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, the four-route package exports only finite displayed data and none of the excluded equality, limit, completeness, trichotomy, or tail-selection objects. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier downstream unblock package`

Theorem 535.35 (Bounded monotone Cauchy witness budget regularity lock)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. The regular-sequence row F and the Cauchy witness row μ are locked to the same finite budget rows σ and λ: any downstream read of regularity, tail control, or tolerance response must pass through the displayed window schedule σ and dyadic ledger λ before it may read either F or μ.*

No quotient equality, host real equality, selected limit, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail can supply an independent budget for F or μ.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier budget regularity lock`

Proof. Start with the carrier of Definition 535.5. The only budget-facing coordinates are the stream schedule σ, the witness row μ, and the dyadic ledger λ, with F as the regular rational handoff.

Apply the NameCert obligations of Definition 535.6 and the tail-window service of Definition 535.25. These force every tolerance request through F, σ, μ, λ as one finite service.

Finally use the downstream package of Definition 535.34. Since every exported route still uses the displayed rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, no external budget row can separate F from μ or replace σ, λ. □

Theorem 535.36 (Bounded monotone Cauchy witness located trap factorization)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every read of the located trap I factors through $S, F, \sigma, \mu, \lambda, H, C, P, N$ before any* $\mathsf{Real}^{\uparrow}$ *seal read is available. The trap therefore records a bounded-source, regular-handoff, scheduled-window, Cauchy-witness, dyadic-budget, transport, continuation, provenance, and naming route, rather than an independent interval selector.*

The factorization exports no host order-completeness argument, quotient equality, selected limit, trichotomy split, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier located trap factorization`

Proof. Begin with the carrier admission and local NameCert obligations. They make I one coordinate of the accepted packet and place it after the bounded source row, regular sequence row, schedule row, witness row, and dyadic ledger.

Use Definition 535.29. It is the direct handoff theorem for the trap row and routes reads of I through $S, F, \sigma, \mu, \lambda$ with the structural rows H, C, P, N still visible.

Then apply the Real observation-budget handoff and the Cauchy modulus-extraction factorization from Definition 535.32 and Definition 535.33. These keep the trap route inside the finite observation and modulus schedule before the Real seal can be consumed. □

Theorem 535.37 (Bounded monotone Cauchy witness Real seal schedule exhaustion)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of* W *reads the displayed schedule* σ*, dyadic ledger* λ*, Cauchy witness* μ*, and located trap* I *through the same finite route* $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$*. Such a consumer receives no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.*

Proof. First invoke the budget regularity lock and located trap factorization above. These place F, μ over the same schedule and dyadic budget, and they force the trap row I through the displayed source, regularity, witness, budget, transport, continuation, provenance, and naming rows.

Next apply Definition 535.12. The seal row Q is readable only after the located trap and witness-facing rows have been displayed.

Finally use Definition 535.13 together with Definition 535.33. The ledger exhausts every public read, and the modulus factorization keeps tolerance responses inside σ, μ, λ. Therefore the seal schedule has no remaining coordinate for any excluded equality, limit, completeness, trichotomy, compactness, or choice object. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal schedule exhaustion`

Theorem 535.38 (Bounded monotone Cauchy witness seal-trap-budget determinacy)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose two Real-completion-facing consumers agree on the bounded source row* S*, regular handoff row* F*, schedule* σ*, Cauchy modulus row* μ*, dyadic ledger* λ*, located trap* I*, and structural rows* H, C, P, N*. Then they determine the same* $\mathsf{Real}^{\uparrow}$ *seal read* Q *and the same completion-facing export of* W*.*

The determinacy is only a row-level determinacy for the displayed carrier route. It does not assert a selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier seal trap budget determinacy`

Proof. First apply the tail-window service. It makes each tolerance request pass through the same displayed $S, F, \sigma, \mu, \lambda$ service, so the two consumers have no separate tail budget to read.

Next use located trap factorization. Since both consumers agree on I and on the structural rows H, C, P, N, every trap-facing read is already fixed by the shared source, handoff, schedule, modulus, ledger, and structural route.

Now apply Real seal schedule exhaustion. The seal row Q is readable only through that shared route, hence both consumers determine the same Real seal read.

Finally invoke the Real completion export theorem. The completion-facing export has no coordinate beyond $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so equality of the shared route and of the seal read forces the same export while excluding all selected-limit, quotient, host-equality, trichotomy, completeness, compactness, and choice-tail data. □

Theorem 535.39 (Bounded monotone Cauchy witness source-tail synchronization)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. A tolerance request to W cannot read the modulus answer μ or the scheduled windows σ unless the bounded source row S and the regular handoff row F are already displayed in the same packet.*

Thus the source, handoff, window, modulus, and dyadic-budget rows form one synchronized service. No consumer may replace that service by quotient equality, host real equality, a selected limit, trichotomy, ambient completeness, or a choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier source tail synchronization`

Proof. Start from the tail-window service of Definition 535.25. It routes every tolerance request through $S, F, \sigma, \mu, \lambda$ before any trap or completion-facing row can be read.

Use the downstream package of Definition 535.34. Its four public routes keep the bounded-source route and the explicit Cauchy-witness route inside the same displayed packet.

Finally invoke ledger exhaustion. Since the accepted carrier has only the rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, a request that reads μ or σ has no separate route around S and F. □

Theorem 535.40 (Bounded monotone Cauchy witness trap-budget monotonicity)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The located trap row I is monotone in the displayed dyadic budget route λ: tightening or transporting the tolerance budget can only be read through the same source row S, regular handoff F, scheduled windows σ, modulus answer μ, and located-trap handoff.*

Consequently the trap route exports no ambient order-completeness argument, trichotomy split, hidden interval selector, host real equality, quotient equality, selected limit, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier trap budget monotonicity`

Proof. First read the budget through Definition 535.25. The dyadic row λ is displayed with the window schedule and modulus answer, so budget movement is not an external operation.

Next use Definition 535.29. It places I after the route $S, F, \sigma, \mu, \lambda$ and keeps the upper-bound data from S visible at the handoff.

The classifier-transport and Real-seal rows then preserve the same displayed budget route under H, C, Q, P, N. Ledger exhaustion excludes any additional selector or completeness coordinate, so the trap can vary only along the displayed λ route. □

Theorem 535.41 (Bounded monotone Cauchy witness seal factorization triad)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. Any* Real$^{\uparrow}$ *seal read of W factors through the triad of public routes already cited by this chapter: the* MonotoneCauchy$^{\uparrow}$ *tail-window certificate, the* CauchyCriterion$^{\uparrow}$ *criterion packet, and the* RealObservationBudget$^{\uparrow}$ *finite observation route.*

The factorized seal still reads only $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. It does not import quotient equality, host real equality, selected limits, trichotomy, ambient completeness, or choice-selected tails.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier seal factorization triad`

Proof. Begin with Definition 535.31. It routes the bounded source through MonotoneCauchy$^{\uparrow}$ and the witness rows through CauchyCriterion$^{\uparrow}$ before the local seal is consumed.

Then use Definition 535.32. The finite observation request reads windows, dyadic data, regular-rational handoff, and Real seal data through σ, λ, F, Q.

Finally apply the downstream package and ledger exhaustion. The package contains the same carrier rows throughout the triad, and exhaustion rules out every absent quotient, host-equality, limit, completeness, trichotomy, and tail-selection coordinate. □

Theorem 535.42 (Bounded monotone Cauchy witness located interval trap certificate)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. The located-trap row I is certified only as the displayed* LocatedInterval$^{\uparrow}$ *trap carried by the dyadic tolerance ledger λ and by the hsame transport rows H. Any consumer that reads I as a trap certificate must therefore keep the source row S, regular handoff F, scheduled windows σ, Cauchy witness μ, dyadic ledger λ, transport rows H, continuation row C, provenance row P, and NameCert row N in the same package.*

The certificate exports no host interval selector, quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier located interval trap certificate`

Proof. Begin with Definition 535.36. It already places every read of I after the displayed rows $S, F, \sigma, \mu, \lambda, H, C, P, N$, so the trap cannot be separated from the finite BoundedMonotone source, regular-rational handoff, scheduled window, and Cauchy witness data.

The LocatedInterval$^{\uparrow}$ certificate is then read through the same dyadic tolerance ledger. The local trap row is not an interval chosen outside the carrier; it is the row paired with λ and transported through H as part of the displayed packet.

Finally use the sibling DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, and LocatedInterval$^{\uparrow}$ handoffs cited by the scoped dependency package. Together with ledger exhaustion, these rows leave no additional coordinate from which a quotient equality, host equality, selected limit, completeness principle, trichotomy split, compactness argument, or choice-selected tail could enter. □

Theorem 535.43 (Bounded monotone Cauchy witness monotone tail coherence)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. The monotone tail schedule of* W *factors through the public* MonotoneCauchy$^{\uparrow}$ *tail-window route, the* CauchyCriterion$^{\uparrow}$ *criterion packet, and the* CauchyModulusExtraction$^{\uparrow}$ *modulus schedule. A downstream tail request may read only the displayed* σ, μ, λ *service over* S *and* F*, not an independently selected tail.*

This coherence exports no quotient equality, host real equality, selected limit, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier monotone tail coherence`

Proof. First apply Definition 535.39. It forces every tolerance request that reads σ or μ to keep the bounded source row S and the regular handoff row F visible in the same packet.

Next use Definition 535.31. This is the chapter's bridge from the bounded monotone source and explicit witness rows to the sibling MonotoneCauchy$^{\uparrow}$ and CauchyCriterion$^{\uparrow}$ certificates.

Finally apply Definition 535.33. It routes the schedule through σ, μ, λ before any tail threshold is exported. Since ledger exhaustion gives no separate coordinate for a tail choice, the monotone tail service remains a finite displayed route rather than a selected subsequence. □

Theorem 535.44 (Bounded monotone Cauchy witness Real completion export)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. Every Real-completion-facing consumer of* W *must pass through the finite observation budget, the regular rational handoff, and the Real seal rows before using the witness. Concretely, such a consumer reads the route* $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *through the finite observation service of* RealObservationBudget$^{\uparrow}$*, the regular-sequence service of* RegSeqRat$^{\uparrow}$*, and the seal handoff to* Real$^{\uparrow}$*.*

The exported witness therefore carries only the displayed finite observation, regular-rational, trap, seal, transport, continuation, provenance, and naming data. It does not close a quotient real construction, host real equality, selected limit, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Proof. Start with Definition 535.32. It routes every real-facing observation through the finite rows σ, λ, F, Q while keeping the carrier rows S, μ, I, H, C, P, N visible.

Next use Definition 535.41. The Real seal read factors through MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, and RealObservationBudget$^{\uparrow}$, so the consumer cannot skip the finite observation budget before reaching the seal row.

It remains to account for the regular handoff and trap data used by the witness. The located interval trap certificate above keeps I tied to LocatedInterval$^{\uparrow}$, λ, and H, while source-tail synchronization keeps σ, μ, λ tied to S and F.

Finally apply ledger exhaustion. The displayed packet has no additional completion coordinate, quotient representative, host equality, selected limit, completeness principle, trichotomy split, compactness proof, or choice-selected tail, so every Real-completion-facing export is forced through the finite observation, regular rational, and Real seal route already named. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real completion export`

Theorem 535.45 (Bounded monotone Cauchy witness formal handoff package)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The paper-side formal handoff for this carrier is exactly the finite package consisting of the bounded source row* S*, regular rational row* F*,* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, Cauchy modulus witness* μ*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* λ*,* $\mathsf{LocatedInterval}^{\uparrow}$ *trap* I*,* $\mathsf{Real}^{\uparrow}$ *seal* Q*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row* N*.*

Every Lean-side consumer that claims Real-completion-facing use of $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *must formalize those rows before it consumes the Real route. The handoff contains no selected limit, quotient real, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.*

Proof. The carrier definition displays exactly the coordinates $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. The local NameCert obligations then read carrier admission, regular rational exposure, schedule exposure, Cauchy witness exposure, dyadic ledger exposure, trap exposure, seal exposure, transport, continuation, provenance, and local naming from those same rows.

The scoped dependency package identifies the sibling sources of the rows. Monotone tail coherence keeps σ, μ, λ tied to the bounded source and regular handoff, while the located interval trap certificate keeps I tied to λ and the componentwise transports H.

Apply the Real completion export theorem. It forces every Real-completion-facing consumer through the finite observation budget, regular rational handoff, trap, seal, transport, continuation, provenance, and naming rows already displayed by the carrier.

Ledger exhaustion finishes the package comparison. Since the displayed carrier has no remaining coordinate for a selected limit, quotient real, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail, the listed rows are precisely the handoff to be formalized by downstream Lean work. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier formal handoff route`

Theorem 535.46 (Bounded monotone Cauchy witness convergence-seal terminal synchronization)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any convergence-seal-facing or Real-completion-facing consumer of* W *synchronizes the same finite tail answer through*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

before it can read the convergence-seal rows. In particular, the terminal tail service is the displayed route through the bounded source, regular handoff, schedule, Cauchy witness, dyadic ledger, located trap, Real seal, transport, continuation, provenance, and local naming rows.

The synchronized route exports no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal terminal synchronization`

Proof. First use Definition 535.15. It gives the two-stage route from the bounded monotone witness through the finite tail service and only then to the convergence-seal rows.

Next apply Definition 535.18. The selector rows μ, σ, λ are read as one shared terminal service, so a seal-facing consumer cannot replace the scheduled tail answer by an independently selected threshold.

For consumers that continue to the Real route, use Definition 535.44. It forces the Real-completion-facing read through the same finite observation, regular handoff, trap, seal, transport, continuation, provenance, and naming rows before any terminal export is visible.

Ledger exhaustion closes the synchronization claim. The accepted packet contains no coordinate for a selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-selected tail; hence every seal or completion-facing consumer reaches the terminal convergence-seal read only through the displayed finite route above. □

Definition 535.47 (Bounded monotone Cauchy witness scheduled tail envelope)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The scheduled tail envelope of* W *is the finite* BHist *subpacket*

$$E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$$

read before the $\mathsf{Real}^{\uparrow}$ *seal row* Q*.*

Its rows are ordered as follows: S *supplies the bounded monotone source generator and its range, monotonicity, and upper-bound rows;* F *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff read from that source;* σ *is the* $\mathsf{StreamName}^{\uparrow}$ *family of inspected source and regular-sequence windows;* μ *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *answer to each dyadic tolerance request;* λ *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and upper-bound budget;* I *is the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap reached from the same budget;* H *records the componentwise* hsame *transports;* C *records the displayed* Cont *routes;* P *records* Pkg *provenance over the sibling concrete-instance rows; and* N *records the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *naming surface.*

Thus E *is the public tail-serving part of* W*: every row needed to answer a tolerance request is displayed before* I *or* Q *is consumed, and* Q *is the seal row read after this envelope rather than a coordinate inside it.*

Lean4 (definition) `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessScheduledTailEn`

Theorem 535.48 (Bounded monotone Cauchy witness window coverage)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let* $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ *be its scheduled tail envelope. Every tolerance request made to* W *factors through* E *and, more specifically, through the ordered service* $S, F, \sigma, \mu, \lambda$ *before the located trap* I *or the* $\mathsf{Real}^{\uparrow}$ *seal* Q *can be read.*

Consequently the request is answered by the source generator, regular rational handoff, scheduled windows, Cauchy modulus answer, and dyadic budget rows of the envelope. It cannot be answered by quotient equality, host real equality, a selected limit, trichotomy, ambient completeness, open-cover compactness, or a choice-selected tail.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window coverage`

Proof. Project the carrier using Definition 535.5. The rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ are the whole accepted packet, and the scheduled envelope removes only the seal row Q from the pre-seal tail service.

Use Definition 535.25. It sends each tolerance request through $S, F, \sigma, \mu, \lambda$ before any located-trap or completion-facing row is available.

Next apply Definition 535.33. The modulus answer is read through the same σ, μ, λ schedule, so the request has no alternate tail threshold outside the envelope.

Finally invoke Definition 535.13. Since the carrier has no further public read coordinate, a consumer cannot reach I or Q without first passing through the displayed source, handoff, window, modulus, and budget rows. □

Theorem 535.49 (Bounded monotone Cauchy witness trap-seal exactness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Any read of the located trap I or the* $\mathsf{Real}^{\uparrow}$ *seal Q factors through E before the read is exported.*

The exported trap or seal is therefore exact for the displayed bounded-source, regular-handoff, scheduled-window, modulus-answer, dyadic-budget, transport, continuation, provenance, and naming rows. It cannot import quotient equality, host real equality, selected limits, trichotomy, ambient completeness, open-cover compactness, or choice-selected tails.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier trap seal exactness`

Proof. Start with the located trap route of Definition 535.36. It places every read of I after $S, F, \sigma, \mu, \lambda$ and keeps the transport, continuation, provenance, and naming rows visible.

Then use Definition 535.37. The seal row Q is available only through the same schedule, dyadic ledger, Cauchy witness, and located trap route.

Combine this with Definition 535.41. The seal factors through the public monotone-tail, Cauchy-criterion, and finite-observation routes, all of which still read the displayed packet rows rather than an external completion object.

Ledger exhaustion supplies the final exclusion. After E has been read, the only remaining row is the seal row Q, so there is no carrier coordinate for quotient equality, host equality, selected limits, trichotomy, completeness, compactness, or choice-selected tails to enter the trap or the seal. □

Theorem 535.50 (Bounded monotone Cauchy witness finite observation lock)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Every finite observation request exported to a* $\mathsf{Real}^{\uparrow}$ *consumer, or to any completion-facing consumer reading the* $\mathsf{Real}^{\uparrow}$ *seal, factors through the displayed bounded source row S, the* $\mathsf{StreamName}^{\uparrow}$ *window schedule σ, the* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *threshold route carried by σ, μ, λ, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *finite-read route, the located trap I, and finally the* $\mathsf{Real}^{\uparrow}$ *seal Q.*

Thus the exported observation is locked to the finite rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. No selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector is exposed by the accepted packet.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier finite observation lock`

Proof. Begin with the finite observation route. By Definition 535.32, every Real-facing finite read passes through the displayed source, stream window, dyadic budget, regular-rational handoff, and seal rows of W.

Next apply Definition 535.33. The threshold service used by such a read is routed through σ, μ, λ, so a completion-facing consumer receives a modulus-indexed finite schedule rather than a separate tail selector.

The trap and seal are then fixed by the local exactness chain. Use Definition 535.36 for the located interval route, Definition 535.37 for the seal schedule, and Definition 535.49 for the fact that reads of I and Q remain exact for the same envelope E.

Finally invoke Definition 535.13. Since the accepted packet has no further public coordinate beyond $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, the composed route exposes no selected limit, quotient real equality, host equality, trichotomy, completeness, compactness, or choice-like tail selector. □

Theorem 535.51 (Bounded monotone Cauchy witness root unblock)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. The root consumer for the Real-and-completion route is unblocked exactly by the finite BEDC package*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

It receives the bounded-source rows from S, the regular rational handoff from F, the selected windows from σ, the Cauchy modulus answer from μ, the dyadic and upper-bound budget from λ, the located trap from I, and the $\mathsf{Real}^{\uparrow}$ *seal from Q, with H, C, P, N carrying transport, route, provenance, and naming data.*

No additional completion object is exposed. In particular, the route exports no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root unblock`

Proof. Project the carrier of Definition 535.5. The only rows available to a consumer are $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and the NameCert obligations require the generator range, explicit Cauchy witness, located trap, seal, and ledger exhaustion rows to be read through that packet.

Use Definition 535.48 and Definition 535.50. These place every tolerance and finite-observation request on the path through $S, F, \sigma, \mu, \lambda$ before I and Q can be consumed.

Finally apply Definition 535.49. The trap and seal stay exact for the same scheduled tail envelope E, so the root consumer has a single finite package rather than an independent completion-facing input. □

Theorem 535.52 (Bounded monotone Cauchy witness window exhaustion)**.** *For any accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$, the scheduled windows σ exhaust the public tail service used by the Real-and-completion route. Each dyadic tolerance request is answered by the Cauchy modulus row μ on a displayed window of σ, read with the dyadic budget λ and the regular rational row F before the located trap I or* $\mathsf{Real}^{\uparrow}$ *seal Q is exported.*

Thus the Cauchy modulus extraction and the $\mathsf{RealObservationBudget}^{\uparrow}$ *finite-read route share the same window ledger. A consumer cannot replace that ledger by a hidden subsequence, an unrecorded tail threshold, a selected limit, quotient equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window exhaustion`

Proof. Begin with the tail-window service. By Definition 535.25, a tolerance request has to pass through $S, F, \sigma, \mu, \lambda$ before any trap or completion-facing row is available.

Next use Definition 535.33. It identifies the threshold route used by the request with the same σ, μ, λ schedule, so the modulus answer cannot be read from a second tail service.

The finite observation side is locked by Definition 535.32 and Definition 535.50. These force the Real observation budget to consume the displayed windows and dyadic rows before the seal route is reached.

Ledger exhaustion closes the exclusion. Since the accepted packet contains no further tail coordinate, no consumer can add a hidden threshold, selected limit, quotient equality, host equality, completeness principle, compactness row, or choice-like selector. □

Theorem 535.53 (Bounded monotone Cauchy witness seal handoff)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The public handoff from W to the Real-and-completion route is the single finite-row package that first serves the tail windows, then extracts the Cauchy modulus, then reads the located trap, and finally exports the* $\mathsf{Real}^{\uparrow}$ *seal:*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The route is public precisely as a BEDC package: $S, F, \sigma, \mu, \lambda$ answer the tail and finite-observation requests, I supplies the located trap, Q supplies the seal, and H, C, P, N keep classifier transport, continuation, provenance, and NameCert naming visible.

Consequently any later RealUp consumer can cite this theorem as the exported BoundedMonotoneCauchyWitness handoff. The theorem does not close host real equality, quotient real construction, selected representatives, trichotomy, ambient completeness, open-cover compactness, or choice-selected tails.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail service seal handoff`

Proof. First build the pre-seal service from the carrier and NameCert rows. The carrier displays $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, and the NameCert obligations require every generator, witness, trap, seal, and ledger read to stay inside that displayed packet.

Apply Definition 535.51. It gives the root Real-and-completion consumer a single finite package instead of separate source, modulus, trap, and seal inputs.

Next apply Definition 535.52. The Cauchy modulus extraction and finite observation budget use the same σ, μ, λ window ledger over S and F, so the package has one public tail route.

Finally use Definition 535.49 together with Definition 535.13. The located trap and $\mathsf{Real}^{\uparrow}$ seal are exact for the scheduled envelope, and the ledger has no remaining coordinate for any excluded completion, equality, compactness, or choice principle. □

Theorem 535.54 (Bounded monotone Cauchy witness scheduled tail downstream coverage)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Every accepted local consumer of the scheduled tail is covered by the displayed finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

More precisely, a tolerance consumer is exhausted by the source, regular-rational handoff, scheduled-window, modulus, and dyadic-ledger rows $S, F, \sigma, \mu, \lambda$; a finite-observation consumer is exhausted by those rows together with the located trap and seal rows I, Q; a trap or seal consumer is exhausted by E followed by Q; and any transport, continuation, provenance, or naming consumer is exhausted by H, C, P, N.

Thus the scheduled tail has no accepted downstream coordinate for quotient equality, host real equality, selected limits, trichotomy, ambient completeness, open-cover compactness, hidden tail

thresholds, or choice-selected tails. Those alternatives remain outside the accepted BoundedMonotoneCauchyWitness *packet.*

Proof. Project the scheduled envelope from Definition 535.47. The envelope displays exactly the pre-seal rows $S, F, \sigma, \mu, \lambda, I, H, C, P, N$, and the only post-envelope row of the accepted carrier is the $\mathsf{Real}^{\uparrow}$ seal row Q.

First consider a tolerance consumer. By Definition 535.48, every tolerance request factors through E and specifically through $S, F, \sigma, \mu, \lambda$ before either I or Q can be read. Hence the tolerance side is already exhausted by the finite service rows of the envelope, with no hidden tail selector or completion input.

Next consider a finite-observation, trap, or seal consumer. The finite-observation route of Definition 535.50 reads the same source, window, modulus, dyadic-budget, located-trap, and seal rows. For direct trap and seal reads, Definition 535.49 says that every read of I or Q factors through E before export and is exact for the displayed source, handoff, schedule, modulus, budget, transport, continuation, provenance, and naming rows.

The remaining local consumers in this file are the root, window-exhaustion, and seal-handoff consumers. By Definition 535.51, the root consumer receives the finite package $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. By Definition 535.52, the scheduled windows and dyadic ledger exhaust the public tail service. By Definition 535.53, the public handoff is the same finite-row package, first serving the tail windows, then extracting the modulus, then reading the located trap, and finally exporting Q.

Combining these cases, every accepted local consumer reads one of the enumerated finite service, ledger, coverage, trap, seal, transport, continuation, provenance, or naming rows. Since the carrier has no coordinate beyond those rows, the excluded quotient-equality, host-equality, selected-limit, trichotomy, completeness, compactness, hidden-threshold, and choice-selected-tail alternatives cannot be accepted downstream reads of the packet. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier scheduled tail downstream coverage`

Theorem 535.55 (Bounded monotone Cauchy witness Real route non-escape)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any Real-and-completion consumer of* W *factors through the finite package*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The consumer must read the bounded monotone source S*, the regular rational handoff* F*, the scheduled windows* σ*, the Cauchy modulus answer* μ*, the dyadic and upper-bound budget* λ*, the located trap* I*, the* $\mathsf{Real}^{\uparrow}$ *seal* Q*, and the transport, continuation, provenance, and naming rows* H, C, P, N *before it can use the packet.*

Consequently the Real-and-completion route has no escape to a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real route nonescape`

Proof. Start with the carrier. By Definition 535.5, the accepted packet exposes exactly the displayed rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ to later consumers.

The tail service is exhausted before the Real seal is reached. Apply Definition 535.48 and Definition 535.52; every tolerance and finite-window request passes through $S, F, \sigma, \mu, \lambda$ before the trap or seal can be read.

The trap and seal remain exact for the same displayed envelope. Use Definition 535.49 to keep reads of I and Q attached to the scheduled tail envelope, and then use Definition 535.53 to identify the exported Real-and-completion handoff as the single package $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Finally invoke Definition 535.13. Since the ledger has no remaining public coordinate after those rows are read, a consumer cannot project a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector from the accepted packet. □

Theorem 535.56 (Bounded monotone Cauchy witness terminal-window split exactness)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every terminal tail service exported by* W *for an accepted packet factors in the exact finite order*

$$S, F, \sigma, \mu, \lambda \longmapsto I \longmapsto Q \longmapsto H, C, P, N.$$

The first stage is the bounded-source, regular-handoff, finite-observation, Cauchy-witness, and dyadic-ledger window. The second stage is the located trap I*. The third stage is the* $\mathsf{Real}^{\uparrow}$ *seal row* Q*. Only after these three stages are visible may the consumer read componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

Thus a terminal tail answer is not a detached threshold, selected subsequence, quotient real, host equality, trichotomy split, ambient completeness principle, open-cover compactness proof, or choice-selected limit. It is exactly the displayed route from finite observation through regular handoff and Real seal to the structural suffix.

Proof. Start with the finite observation stage. By Definition 535.39 and Definition 535.33, a terminal request that reads σ, μ, or λ keeps the bounded source row S and regular handoff row F in the same accepted packet.

Next pass to the located trap. The located interval trap certificate fixes I as the pre-seal enclosure obtained after the finite observation, modulus, and dyadic ledger rows have been consumed. No terminal service can replace that row by an ambient interval choice because the carrier has no such coordinate.

Then read the Real seal. Apply Definition 535.44: any Real-completion-facing consumer must pass through the finite observation budget, regular rational handoff, trap, seal, transport, continuation, provenance, and naming rows before a completion-facing export is visible.

Finally compare the suffix. The convergence-seal terminal synchronization theorem forces the same displayed route for seal-facing consumers, while ledger exhaustion leaves H, C, P, N as transport, replay, provenance, and naming only. Therefore every accepted terminal tail service splits exactly through the finite observation stage, the regular handoff and located trap, the Real seal, and the structural suffix named in the statement. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier terminal window split exactness`

Theorem 535.57 (Bounded monotone Cauchy witness tail envelope public export)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let*

$$E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$$

be its scheduled tail envelope. The envelope E exports the single public pre-seal service for completion-facing consumers: source reads pass through S, regular rational reads pass through F, window requests pass through σ, tolerance answers pass through μ, dyadic and upper-bound budget reads pass through λ, trap reads pass through I, and the transport, continuation, provenance, and naming routes pass through H, C, P, N.

Thus every public tail-envelope read is already a read through $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ before the $\mathsf{Real}^{\uparrow}$ *seal row Q is consumed. The export contains no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail envelope public export`

Proof. Start with the scheduled windows. By Definition 535.48, the source, regular-sequence, Cauchy witness, dyadic ledger, and trap-facing reads all pass through $S, F, \sigma, \mu, \lambda, I$ on the same displayed window schedule.

Next use the finite-read route. By Definition 535.50, Real-facing finite observations are locked to the displayed rows of W, so the public pre-seal route cannot replace $S, F, \sigma, \mu, \lambda, I$ by an unrecorded completion object.

The exactness and handoff theorems identify the public boundary. Apply Definition 535.49 to keep the located trap exact for the same envelope E, and apply Definition 535.53 to locate the later $\mathsf{Real}^{\uparrow}$ seal after the displayed pre-seal service.

Finally, Definition 535.52 rules out any second tail service. Since the accepted packet exposes no coordinate beyond the listed rows, the envelope exports exactly the public service $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ before Q is read. □

Theorem 535.58 (Bounded monotone Cauchy witness completion consumer non-escape)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Every completion-facing consumer of the bounded monotone Cauchy witness factors through the public envelope E, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *finite-read route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap, and the* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently the consumer has no escape route to a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector. Its entire public dependency surface is the finite package $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion consumer nonescape`

Proof. First build the envelope route. The public export theorem above identifies E as the only pre-seal service, and Definition 535.48 keeps every inspected source and tail window on that service.

Next attach finite observation. By Definition 535.50, every Real-facing finite read is routed through the displayed finite rows of W. The regular rational handoff is the F coordinate of that same packet, and the window ledger is the shared σ, μ, λ route.

The trap and seal then close the route. Apply Definition 535.49 to keep I exact for the scheduled envelope, and apply Definition 535.53 to pass from the trap side to the $\mathsf{Real}^{\uparrow}$ seal row Q without introducing a second completion-facing input.

It remains to exclude escapes. Definition 535.51 gives the root consumer exactly the finite package $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, while Definition 535.52 rules out a hidden tail service. Therefore

any completion-facing consumer factors through the displayed envelope, finite-read route, regular-sequence handoff, located trap, and seal, and no selected limit, equality principle, completeness principle, compactness row, or choice-like selector is exposed. □

Theorem 535.59 (Bounded monotone Cauchy witness completion consumer apartness budget)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Every completion-facing consumer of W must consume the scheduled tail envelope E together with the finite apartness and budget rows σ, μ, λ, I before it consumes the* $\mathsf{Real}^{\uparrow}$ *seal row Q.*

Consequently the route cannot be read as a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector. The only public pre-seal completion data are the scheduled envelope and the finite apartness-budget rows already displayed by W.

Proof. Start with the public export theorem. Definition 535.57 identifies E as the exported pre-seal service and lists σ, μ, λ, I among the rows read before the seal.

Next use the finite-observation lock. By Definition 535.50, every Real-facing finite read remains on the displayed rows of W, so no consumer can replace the apartness and budget schedule by an unrecorded completion object.

The trap and seal then fix the order of consumption. Apply Definition 535.49 to keep I on the same located trap surface, and apply Definition 535.53 to place the $\mathsf{Real}^{\uparrow}$ seal row Q after that finite surface.

Finally apply Definition 535.52 and Definition 535.58. The first rules out a second tail service, and the second rules out escape through limit selection, equality principles, completeness, compactness, or choice-like selection. Thus every completion-facing consumer must pass through the scheduled envelope and the finite apartness-budget rows before the seal row is consumed. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion consumer apartness budget`

Theorem 535.60 (Bounded monotone Cauchy witness envelope uniqueness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. If two completion-facing consumer routes for W are admitted before the* $\mathsf{Real}^{\uparrow}$ *seal row is read, then both routes consume the same scheduled tail envelope rows*

$$S, F, \sigma, \mu, \lambda, I, H, C, P, N$$

and only then read the seal row Q.

Any claimed difference between the two pre-seal envelopes is not a second completion route for W. It is routed to the refusal boundary for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selectors.

Proof. First identify the exported envelope. By Definition 535.57, the only public pre-seal service is the displayed tuple $S, F, \sigma, \mu, \lambda, I, H, C, P, N$.

Next apply completion non-escape. Definition 535.58 forces every completion-facing consumer of W through that public envelope, the finite-read route, the regular handoff, the located trap, and the seal row.

The apartness and budget rows are fixed before the seal by Definition 535.59. Thus two admitted routes cannot differ on σ, μ, λ, I while remaining completion-facing routes for the same accepted packet.

It remains to rule out a second finite envelope with the same visible role. The finite-observation lock Definition 535.50 keeps Real-facing finite reads on the displayed rows of W, and Definition 535.52 rules out a hidden tail service. Therefore any disagreement is outside the finite envelope and is handled by the existing refusal boundary for the listed non-envelope principles. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier envelope uniqueness`

Theorem 535.61 (Bounded monotone Cauchy witness completion observation route)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every completion-facing observation route for* W *first consumes the rows*

$$S, F, \sigma, \mu, \lambda, I, H, C, P, N,$$

then passes through the finite $\mathsf{RealObservationBudget}^{\uparrow}$ *route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap, and only then reads the* $\mathsf{Real}^{\uparrow}$ *seal row* Q*.*

The route exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector. Thus completion consumers read the finite observation surface before any Real-facing seal is consumed.

Proof. First expose the scheduled envelope. By Definition 535.57, the public pre-seal rows of W are exactly $S, F, \sigma, \mu, \lambda, I, H, C, P, N$.

Next route the completion consumer through the finite observation surface. Apply Definition 535.58 to force the read through the public envelope, the finite observation route, the regular rational handoff, the located trap, and the later seal row.

The apartness and budget rows fix the order of the finite segment. By Definition 535.59, σ, μ, λ, I are consumed before the $\mathsf{Real}^{\uparrow}$ seal row Q, so the seal cannot precede the finite observation route.

Finally use envelope uniqueness. Definition 535.60 rules out a second pre-seal envelope, and the non-escape conclusion excludes selected limits, equality principles, trichotomy, completeness, compactness, and choice-like selectors from the exported route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion observation route`

Theorem 535.62 (Bounded monotone Cauchy witness Real observation seal factorization)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every Real-completion-facing consumer of* W *factors in the following order: first through the scheduled tail envelope*

$$S, F, \sigma, \mu, \lambda, I, H, C, P, N,$$

then through the finite $\mathsf{RealObservationBudget}^{\uparrow}$ *route, then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{LocatedInterval}^{\uparrow}$ *trap, and only then through the* $\mathsf{Real}^{\uparrow}$ *seal row* Q*.*

Consequently the consumer never receives a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector. The only public seal-facing dependency surface is the finite package $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *with the pre-seal rows consumed before* Q*.*

Proof. Start with the envelope rows. Theorem 535.57 identifies the scheduled tail envelope as the displayed pre-seal service and places $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ before the seal row.

Next apply the completion-consumer non-escape theorem. It forces every Real-completion-facing consumer through the public envelope, the finite observation route, the regular rational handoff, the located trap, and the later $\mathsf{Real}^{\uparrow}$ seal, with no additional completion object exposed.

The pre-seal order is fixed by the apartness-budget theorem. Theorem 535.59 requires the rows σ, μ, λ, I to be consumed before the seal row Q, so the finite budget cannot be read after the Real seal.

Finally use the completion observation route. Theorem 535.61 binds the observation route to the same envelope, budget, handoff, trap, and seal order. Its non-escape conclusion rules out selected limits, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, and choice-like selectors from the exported surface. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real observation seal factorization`

Theorem 535.63 (Bounded monotone Cauchy witness completion envelope determinacy)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. If two completion-facing observation routes for W agree on the scheduled envelope rows*

$$S, F, \sigma, \mu, \lambda, I, H, C, P, N$$

before the $\mathsf{Real}^{\uparrow}$ *seal row Q is read, then they determine the same accepted completion-facing envelope.*

The two routes cannot differ by selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector. The only accepted pre-seal completion surface is the displayed scheduled envelope.

Proof. First identify the pre-seal rows. The public export theorem Definition 535.57 presents $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ as the scheduled tail envelope read before Q.

Next force both routes through that envelope. By Definition 535.58, any completion-facing observation of W must pass through the public envelope, the finite observation route, the regular handoff, and the located trap before it reaches the seal row.

The finite order is fixed by the apartness-budget theorem. Definition 535.59 places σ, μ, λ, I before Q, so the agreed pre-seal rows cannot be bypassed by a later seal read.

Finally use envelope uniqueness. Definition 535.60 rules out a second pre-seal route for the same packet, and Definition 535.61 keeps the completion-facing read on the finite observation surface. Hence the two routes determine the same accepted envelope and cannot differ by any of the excluded completion or equality principles. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion envelope determinacy`

Corollary 535.64 (Bounded monotone Cauchy witness tail envelope strict local obstruction)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet, and let $E = (S, F, \sigma, \mu, \lambda, I, H, C, P, N)$ be its scheduled tail envelope. Any completion-facing use of the tail envelope factors through the displayed pre-seal rows*

$$S, F, \sigma, \mu, \lambda, I, H, C, P, N,$$

then through the finite $\mathsf{RealObservationBudget}^{\uparrow}$ *route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap, and finally the* $\mathsf{Real}^{\uparrow}$ *seal row Q.*

Consequently the obstruction carried by the completion-consumer non-escape row is strict and local. It blocks exactly the attempted converse or reuse route that reads the finite tail envelope as a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, choice-like tail selector, or unlisted validation coordinate. It does not state a global impossibility theorem, does not supply a host-real theorem, and does not certify any external completion principle outside the displayed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail envelope strict local obstruction`

Proof. Apply Definition 535.58. It sends every completion-facing consumer of W through the public envelope E, the finite observation route, the regular rational handoff, the located trap, and the seal row, and it explicitly excludes selected limits, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selectors from the consumer surface.

Apply Definition 535.61. This fixes the order of any completion-facing observation: the rows $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ are consumed before the finite observation route and before the seal row Q. Therefore a claimed converse or reuse route must either factor through those rows or ask for a coordinate not exported by the accepted packet.

If the claimed route asks only for the listed rows, it is not obstructed by the non-escape conclusion; it is just the admitted local observation route. If it asks for a selected limit, an equality principle, trichotomy, completeness, compactness, a choice-like selector, or an external validation coordinate, then the request is rejected precisely because that coordinate is absent from the finite dependency surface. The rejection is therefore local to the named converse or reuse attempt.

Nothing in the two cited theorems quantifies over all host real constructions, all completion theorems, or all external validation mechanisms. Their common conclusion is only that the accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ packet exposes the displayed finite rows and no additional completion-facing coordinate. Hence the obstruction cannot be reinterpreted as a global impossibility, a host theorem, or a hidden external validation principle. □

Theorem 535.65 (Bounded monotone Cauchy witness convergence-seal source)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The witness rows of* W *supply the finite source row consumed by* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *in chapter 918: the bounded source row* S*, regular rational handoff* F*, stream schedule* σ*, Cauchy witness* μ*, dyadic ledger* λ*, located trap* I*, Real seal row* Q*, transport rows* H*, continuation rows* C*, package row* P*, and local* $\mathsf{NameCert}$ *row* N *are the whole source surface.*

The supplied source row exports no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal source`

Proof. Project the accepted carrier and the scoped dependency package of Definition 535.19. These clauses expose exactly the displayed source, handoff, stream, witness, dyadic, trap, seal, transport, continuation, provenance, and naming rows.

The convergence-seal carrier of chapter 918 reads a finite bounded monotone source together with a regular handoff, stream schedule, Cauchy evidence, dyadic ledger, compatibility route, and Real-facing seal. Each of those coordinates is already one of the rows of W.

Since the carrier has no coordinate for a selected limit, quotient equality, host equality, trichotomy, ambient completeness, compactness principle, or choice-like selector, the handoff to the seal cannot export such data. □

Theorem 535.66 (Bounded monotone Cauchy witness finite-tail-filter handoff)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every convergence-seal or Real-completion consumer of* W *first factors through the displayed bounded witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through $\mathsf{MonotoneCauchy}^{\uparrow}$ *and* $\mathsf{CauchyCriterion}^{\uparrow}$*. The tail-window part of that route lands in the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet of chapter 1119 before the consumer continues through*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The handoff exports no selected limit, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector outside the displayed witness and finite-tail packets.

Proof. First read the source boundary. The convergence-seal source theorem Definition 535.65 confines every seal-facing source to the bounded witness rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and rules out a separate completion coordinate.

Next apply the criterion-tail factorization Definition 535.69. It forces every completion-facing read through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ before any Real-facing seal is available, so the tail data are read as a public finite window rather than as an unrecorded selector.

Now pass through the finite-tail interface. FiniteTailFilter tail-window exactness Definition 1119.4 identifies the admissible stream window, dyadic ledger, regular readback, tail budget, common-tail row, and Real seal row as one $\mathsf{FiniteTailFilter}^{\uparrow}$ packet, and FiniteTailFilter Real seal handoff Definition 1119.5 sends that packet to the Real-facing consumer without adding a completion principle.

Finally use Real-seal public route exhaustion Definition 535.70. The remaining route is exactly through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the structural BHist, hsame, Cont, Pkg, and NameCert rows already displayed by the carrier. Hence no selected limit, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector can escape the bounded witness and finite-tail handoff. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier finite tail filter handoff`

Theorem 535.67 (Bounded monotone Cauchy witness tail-filter lock)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Any completion-facing consumer that reads* W *through* $\mathsf{FiniteTailFilter}^{\uparrow}$ *must first pass through the displayed witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through $\mathsf{MonotoneCauchy}^{\uparrow}$ *and* $\mathsf{CauchyCriterion}^{\uparrow}$*, and then through the finite-tail packet rows supplied by Definition 1119.4 and Definition 1119.5. Only after those rows have been read may the consumer continue to*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The route contains no selected limit, quotient stream equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector outside the displayed witness and finite-tail packets.

Proof. Begin with the convergence-seal source theorem, Definition 535.65. It fixes the accepted source of every seal-facing read as the finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and excludes a detached completion coordinate at the source boundary.

Next apply criterion-tail factorization, Definition 535.69. This forces the route through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ before any completion-facing consumer can inspect a Real seal.

Now use the finite-tail interface. Tail-window exactness, Definition 1119.4, identifies the stream window, dyadic ledger, regular readback, tail budget, common-tail row, and Real-seal row as one $\mathsf{FiniteTailFilter}^{\uparrow}$ packet. The Real seal handoff, Definition 1119.5, then sends exactly that packet to the Real-facing consumer.

Finally apply Real-seal public route exhaustion, Definition 535.70. It lists the remaining public route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the structural BHist, hsame, Cont, Pkg, and NameCert rows. Since no listed carrier exports the excluded completion data, every finite-tail consumer is locked to the stated L10 route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail filter lock`

Theorem 535.68 (Bounded monotone Cauchy witness selector terminal pullback)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every terminal consumer preserving the L10 route across*

$$\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegS}$$

pulls back to the same finite selector packet before any convergence seal, limit seal, real carrier, criterion route, stream name, dyadic core, regular sequence, or diagonal comparison is read. The pullback rows are exactly

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The pullback exports no quotient stream, selected limit, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.*

Proof. First identify the source packet. The source-readback equalizer fixes the accepted packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ before the convergence-seal readback can expose any public completion coordinate.

Next move to terminal consumers. The terminal-source equalizer transports the same packet to the $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ endpoints, while terminal located-trap preservation keeps the row I and its attached Q, H, C, P, N support inside that packet.

Now use the L10 route. Terminal route exhaustion brings in the $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ consumers and records that their only public support rows are the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply refusal completeness. Any terminal request outside those rows is off-route, so the consumer cannot read a seal, diagonal comparison, quotient stream, selected limit, host equality,

trichotomy, compactness, ambient completeness, or choice-like selector before returning to the finite selector packet. Thus every terminal consumer preserving the L10 route pulls back to exactly the stated packet and support rows. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier selector terminal pullback`

Theorem 535.69 (Bounded monotone Cauchy witness criterion-tail factorization)**.** *Every* BoundedMonotoneCauch *route from the bounded source row to the convergence-seal handoff factors through the public* MonotoneCauchy$^{\uparrow}$ *row and the public* CauchyCriterion$^{\uparrow}$ *row before any* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *consumer reads it.*

Equivalently, a downstream seal consumer first sees the monotone Cauchy bridge of Definition 535.31, then the displayed Cauchy criterion tail package, and only then the finite Real-seal handoff.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier criterion tail factorization`

Proof. Start with the bridge theorem Definition 535.31. It identifies the bounded monotone witness surface as a public monotone Cauchy row whose tail data are read through the displayed Cauchy criterion route.

Next apply the convergence-seal input package of Definition 535.90. That package routes the same carrier through MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, and Real$^{\uparrow}$ before any completion-facing consumer can inspect the seal.

Thus the witness route has no direct jump from boundedness and monotonicity to a Real seal. Its only public path is the criterion-tail factorization already displayed in the accepted carrier and the downstream convergence-seal chapter. □

Theorem 535.70 (Bounded monotone Cauchy witness Real-seal public route exhaustion)**.** *The public Real-seal route exposed by* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *is exhausted by the rows supplied by* BoundedMonotoneCauchyWitness$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

No part of that public route contains a selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. Use the source theorem above to restrict the public source to the finite witness rows of W. Use the factorization theorem above to force every seal-facing read through MonotoneCauchy$^{\uparrow}$ and CauchyCriterion$^{\uparrow}$ before the convergence-seal handoff.

The convergence-seal carrier at chapter 918 then records the remaining rows: stream names, dyadic tolerances, regular rational handoff, Cauchy-limit compatibility, Real seal, history transport, continuations, package provenance, and local naming.

The listed rows are the complete public route. Since none of the carriers includes a coordinate for selected limits, quotient equality, host real equality, trichotomy, ambient completeness, compactness, or choice-like tail selection, the public Real-seal handoff is exhausted by the displayed BEDC rows. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal public route exhaustion`

Theorem 535.71 (Bounded monotone Cauchy witness seal-route naturality)**.** *Any downstream consumer that reads the bounded monotone witness through the convergence-seal chapter obtains the same public route whether it starts from the witness source theorem, the criterion-tail factorization, or the Real-seal public route exhaustion. The common route factors through* $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The public handoff exports no selected limit, quotient equality, host real equality, trichotomy, open-cover compactness, or choice-like selector.

Proof. Begin with the source surface of Definition 535.65. It fixes the finite witness rows that the convergence-seal chapter may read and excludes selected limits, quotient equalities, host equalities, compactness principles, and choice-like tail selectors.

Next apply the criterion-tail factorization of Definition 535.69. This is the only public bridge from the bounded witness source to a completion-facing seal route, so every downstream read must pass through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ before it can inspect the Real-facing rows.

Finally apply the Real-seal public route exhaustion of Definition 535.70. It lists the remaining public rows $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. Since all three starting descriptions are forced through the same factorization and the same exhausted route, the downstream handoff is natural across the displayed source, criterion-tail, and Real-seal readings. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal route naturality`

Theorem 535.72 (Bounded monotone Cauchy witness convergence-seal source totality)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every convergence-seal source read exposed from* W *is one of the displayed witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The read has no extra source coordinate for a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal source totality`

Proof. Apply the checked source theorem Definition 535.65. It identifies the finite convergence-seal source surface with exactly the bounded source, regular rational handoff, stream schedule, Cauchy witness, dyadic ledger, located trap, Real seal, transport, continuation, package, and local naming rows of W.

The criterion-tail factorization of Definition 535.69 shows that a seal-facing source read cannot bypass those rows by jumping directly to a completion-facing object. The factorization first routes through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$, then through the displayed schedule, ledger, handoff, trap, and seal rows.

Finally use the Real-seal public route exhaustion of Definition 535.70. Since the exhausted route contains only the displayed finite rows and structural BEDC rows, the source read is total on the witness package and exports none of the excluded completion data. □

Theorem 535.73 (Bounded monotone Cauchy witness convergence-seal readback)**.** *Every readback from a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer to an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$

carrier $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *factors through the displayed source/schedule/regular/trap/seal route*

$$S, \sigma, F, I, Q, H, C, P, N$$

together with the Cauchy witness and dyadic ledger rows μ, λ*. Thus the readback returns only to the finite source surface of* W *and never to an unrecorded completion principle.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal readback`

Proof. Begin with seal-route naturality above. It says that a downstream consumer obtains the same route whether it starts from the source theorem, the criterion-tail factorization, or the Real-seal route exhaustion.

Read the route in the reverse direction from the convergence-seal consumer. Naturality forces the readback through the same source row S, stream schedule σ, regular rational handoff F, located trap I, Real seal Q, and structural rows H, C, P, N, with μ, λ supplying the Cauchy witness and dyadic budget already present in the source theorem.

The completion-consumer non-escape theorem Definition 535.58 keeps completion-facing consumers inside the public envelope, finite-read route, regular handoff, trap, and seal. Therefore the readback has no branch to selected limits, quotient equality, host equality, trichotomy, ambient completeness, compactness, or choice-like tail selection. □

Theorem 535.74 (Bounded monotone Cauchy witness convergence-seal non-escape)**.** *Every completion consumer reached through the convergence-seal source surface of an accepted* BoundedMonotoneCauchyWitnes *carrier remains inside the finite route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

It cannot select a limit, assert quotient real equality, assert host real equality, decide trichotomy, invoke ambient completeness, invoke open-cover compactness, or acquire a tail selector outside the displayed carrier.

Proof. First apply the convergence-seal source totality theorem above. It confines every source read to the displayed witness rows of W.

Next apply convergence-seal readback. Any consumer-side return route is the same finite source/schedule/regular/trap/seal route, with the Cauchy witness and dyadic ledger still supplied by μ, λ.

Finally apply Definition 535.58. That checked non-escape row says completion-facing consumers factor through the public envelope, finite-read route, regular handoff, located trap, and $\mathsf{Real}^{\uparrow}$ seal. Since all of those rows are already in the displayed carrier and no excluded completion coordinate occurs in the packet, the convergence-seal consumer remains inside the finite route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal nonescape`

Theorem 535.75 (Bounded monotone Cauchy witness convergence-seal source lock)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any* BoundedMonotoneCon *consumer whose source is supplied by* W *must first factor through the displayed witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through MonotoneCauchy$^{\uparrow}$ *and* CauchyCriterion$^{\uparrow}$*, and then through the finite* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$ *route before it is read as a Real-facing convergence seal.*

The consumer cannot replace that source by a selected limit, quotient stream equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. First apply Definition 535.72. It locks every source read supplied by W to the displayed finite witness rows and leaves no separate completion-facing coordinate.

Next use Definition 535.73. Any read from a convergence-seal consumer back to the witness must return along the same source, schedule, regular-rational, Cauchy-witness, dyadic-ledger, trap, seal, transport, continuation, package, and local naming rows.

Finally invoke Definition 535.74. The non-escape result keeps the consumer inside the finite route through MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, and Real$^{\uparrow}$. Since the route has no coordinate for the excluded analytic or choice-like objects, no downstream consumer can substitute such an object for the source supplied by W. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal source lock`

Theorem 535.76 (Bounded monotone Cauchy witness convergence-seal route exhaustion)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet. Every convergence-seal or Real-completion consumer of* W *factors through the displayed finite route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through MonotoneCauchy$^{\uparrow}$*,* CauchyCriterion$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

The route exports no selected limit, quotient stream equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal route exhaustion`

Proof. First apply the checked source theorem Definition 535.65. It identifies the source surface available to a convergence-seal consumer with exactly the finite witness rows of W.

Next use the criterion-tail factorization and the Real-seal public route exhaustion. These force the downstream handoff through MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, and Real$^{\uparrow}$ before any Real-completion-facing consumer can read the seal.

Finally combine the readback, non-escape, and source-lock theorems above. Readback returns consumer-side information to the same finite packet, non-escape prevents a branch to an external completion principle, and source-lock prevents replacement of W by any unrecorded selected limit, quotient equality, host equality, trichotomy, compactness, completeness, or selector coordinate. □

Theorem 535.77 (Bounded monotone Cauchy witness convergence-seal selector readback)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet. Every convergence-seal selector readback of* W *first returns to the displayed witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

and then factors through

$$\mathsf{MonotoneCauchy}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The readback exports no quotient stream, host real equality, selected limit, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.

Proof. Begin with source totality, Definition 535.72. It confines every convergence-seal source read of W to the displayed witness rows and gives the selector no separate completion-facing coordinate.

Next apply convergence-seal readback and non-escape, Definition 535.73 and Definition 535.74. The readback returns consumer-side information to the same finite packet, while non-escape keeps completion consumers inside the public route through the bounded source, criterion tail, stream, dyadic, regular, limit-seal, and Real rows.

Finally use source lock and route exhaustion, Definition 535.75 and Definition 535.76. These two statements identify the only available selector route with the displayed packet followed by $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since none of the excluded analytic or choice-like objects is a row of that route, the selector readback cannot export one. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal selector readback`

Theorem 535.78 (Bounded monotone Cauchy witness root-window convergence-seal lock)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Any convergence-seal or Real-completion consumer of W must first read the root finite window rows*

$$S, F, \sigma, \mu, \lambda,$$

as the scheduled source, regular handoff, stream schedule, Cauchy witness, and dyadic budget of the accepted packet. Only after that root-window read may the consumer continue through the displayed route

$$\mathsf{MonotoneCauchy}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The consumer cannot replace the scheduled window or its refusal boundary by selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or a choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window convergence seal lock`

Proof. First apply the convergence-seal source theorem Definition 535.65. It identifies the source rows available to the seal with the finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, and it already excludes selected limits, quotient equality, host equality, trichotomy, completeness, compactness, and choice-like tail selection.

Next apply the criterion-tail factorization Definition 535.69. This forces the consumer to pass from the root window through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ before it can read any seal-facing completion route.

The readback theorem Definition 535.73 returns every consumer-side read to the same displayed source, schedule, regular handoff, Cauchy witness, dyadic ledger, trap, seal, transport, continuation, package, and local naming rows. Thus the scheduled root window remains the boundary that a consumer must read back through.

Finally combine non-escape Definition 535.74 with route exhaustion Definition 535.76. Non-escape rules out any branch to an external completion principle, and route exhaustion lists the only public path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The convergence-seal consumer is therefore locked to the scheduled root window and its refusal boundary. □

Theorem 535.79 (Bounded monotone Cauchy witness located-trap convergence-seal cofinality)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Any* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of W must preserve the located interval row I cofinally with the root window rows*

$$S, F, \sigma, \mu, \lambda$$

and the terminal rows

$$Q, H, C, P, N.$$

Thus every completion-facing read that reaches the convergence seal still carries the same located-trap row I as part of the bounded monotone witness packet.

The consumer cannot bypass I through selected limits, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or a choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier located trap convergence seal cofinality`

Proof. First apply source totality, Definition 535.72. It identifies every seal-facing source read with the displayed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the located interval row I is present before any completion consumer inspects the route.

Next use convergence-seal readback, Definition 535.73. Readback returns consumer-side information to the same bounded monotone witness rows, which keeps I in the packet whenever a $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads the seal.

Now apply the root-window convergence-seal lock, Definition 535.78. The scheduled rows $S, F, \sigma, \mu, \lambda$ cannot be detached from the same packet as the located trap I and the terminal rows Q, H, C, P, N.

Finally use route exhaustion, Definition 535.76. The only completion-facing route runs through $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Carrier-coordinate exhaustion then leaves no coordinate from which a consumer could replace the located trap by selected limits, quotient equality, host equality, trichotomy, completeness, compactness, or selector data. □

Theorem 535.80 (Bounded monotone Cauchy witness convergence-seal diagonal readback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *route. Any* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of W whose terminal read is compared through* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *first factors through the same finite witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through

$$\mathsf{MonotoneCauchy}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow},$$

and finally through the diagonal terminal-consumer rows

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The comparison exports no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal diagonal readback`

Proof. First lock the source side. Apply source totality Definition 535.72, convergence-seal readback Definition 535.73, and convergence-seal non-escape Definition 535.74. These results keep the consumer read inside the finite witness rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next attach the public seal route. The source lock Definition 535.75, route exhaustion Definition 535.76, and root-window lock Definition 535.78 force the read through $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any terminal comparison is available.

Finally compare the terminal consumer through $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$. By Definition 646.84, every diagonal Real or completion-seal read passes through $R, T, S, D, W, Q, L, H, C, P, N$; by Definition 646.85, this route is exhausted by the displayed diagonal rows, Pkg provenance, and local NameCert row. Therefore the bounded-monotone convergence seal and the diagonal-limit terminal consumer meet at the stated finite boundary without importing any excluded analytic or choice-like coordinate. □

Theorem 535.81 (Diagonal-limit compatibility bounded monotone window-seal handoff)**.** *Let* $W_{\mathrm{bm}} = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose a terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read of* W_{bm} *is compared through an accepted* DiagonalLimitCompatibil *packet. Then the consumer first factors through the bounded-monotone rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through the diagonal terminal rows

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The two routes meet only through the shared BHist, hsame, Cont, Pkg, *and* NameCert *surfaces. The handoff exports no selected limit, quotient equality, host* $\mathsf{Real}^{\uparrow}$ *equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.*

Proof. Start with the existing diagonal comparison. The convergence-seal diagonal readback theorem Definition 535.80 already forces any bounded-monotone convergence-seal or $\mathsf{Real}^{\uparrow}$ consumer compared through $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ through the finite witness rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ before it reaches the diagonal terminal rows.

Next synchronize the prefix and suffix inside the bounded-monotone witness. By Definition 535.152, the scheduled root window $S, F, \sigma, \mu, \lambda$ and the seal-facing suffix I, Q, H, C, P, N belong to the same accepted packet. Thus the terminal read cannot detach the diagonal comparison from the bounded-monotone source packet.

Now attach the diagonal budget-seal route. The factorization theorem Definition 646.84 routes every diagonal terminal consumer through $R, T, S, D, W, Q, L, H, C, P, N$ before it is offered to a completion-facing read.

Finally combine the non-escape clauses already used in the diagonal readback paragraph. The bounded-monotone side contributes only the finite witness rows and its BHist, hsame, Cont, Pkg, and NameCert surfaces; the diagonal side contributes only the displayed terminal rows and the same shared surfaces. Therefore the two routes meet at that finite handoff boundary, with no coordinate from which to read a selected limit, quotient equality, host $\mathsf{Real}^{\uparrow}$ equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window seal handoff`

Theorem 535.82 (Bounded monotone Cauchy witness convergence-seal terminal source pullback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer that reaches the convergence-seal source factors through the same finite source pullback*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

then through $\mathsf{MonotoneCauchy}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Thus the completion-facing consumers identify the same accepted source packet before reading any seal-facing route.*

The pullback exports no selected limit, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal terminal source pullback`

Proof. Start with source totality Definition 535.72. It identifies every convergence-seal source read supplied by W with the finite rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and gives no additional completion-facing coordinate.

Next apply readback Definition 535.73, non-escape Definition 535.74, and source lock Definition 535.75. Readback returns consumer-side information to the same packet, non-escape prevents a branch to external completion data, and source lock prevents replacing the source by any unrecorded analytic or choice-like object.

Now use route exhaustion Definition 535.76 together with root-window convergence-seal lock Definition 535.78. The former lists the whole route through $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; the latter fixes the scheduled root window $S, F, \sigma, \mu, \lambda$ inside that route.

It remains only to exhaust the coordinates of the accepted carrier. The source, handoff, stream, Cauchy witness, dyadic ledger, trap, Real seal, transport, continuation, package, and local naming rows are exactly $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Hence every listed consumer pulls back to the same finite packet before it reaches the public completion route, and no excluded completion principle is exported. □

Theorem 535.83 (Bounded monotone Cauchy witness source-readback equalizer)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. The source totality row and the convergence-seal readback row determine the same finite source packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

for every seal-facing consumer. Thus a read that starts from the bounded source and a read that returns from BoundedMonotoneConvergenceSeal$^{\uparrow}$ *are equalized before any completion-facing route is inspected.*

The equalizer has no branch for selected limits, quotient equality, host equality, trichotomy, open-cover compactness, ambient completeness, or choice-like selectors.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier source readback equalizer`

Proof. First compare the forward source read with the reverse readback. Source totality, Definition 535.72, exposes exactly the rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Convergence-seal readback, Definition 535.73, returns every consumer-side read to the same rows.

Next lock the common route. Non-escape Definition 535.74 prevents a detour to external completion data, source lock Definition 535.75 prevents replacement of the source packet, and route exhaustion Definition 535.76 lists the only completion-facing path.

Finally use the root-window lock, Definition 535.78. The scheduled root window $S, F, \sigma, \mu, \lambda$ remains inside the same packet as I, Q, H, C, P, N, so the source and readback rows form a single finite equalizer. Since the listed theorems provide no coordinate for selected limits, quotient equality, host equality, trichotomy, compactness, completeness, or selector data, those reads are ruled out. □

Theorem 535.84 (Bounded monotone Cauchy witness terminal source equalizer)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet. The source-readback equalizer composes with the diagonal terminal-source pullback, so every* BoundedMonotoneConvergenceSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, *or* DiagonalLimitCompatibility$^{\uparrow}$ *terminal consumer reads the same finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

before any seal-facing or terminal comparison route is available.

No terminal consumer obtains a selected limit, quotient equality, host equality, trichotomy, open-cover compactness, ambient completeness, or choice-like selector from this comparison.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier terminal source equalizer`

Proof. Begin with the source-readback equalizer above. It identifies the forward source route and the reverse convergence-seal readback as the same finite packet, with no extra completion coordinate.

Now compose with the terminal-source pullback, Definition 535.82. That pullback already sends BoundedMonotoneConvergenceSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, and Real$^{\uparrow}$ consumers through the same source packet before the public seal route.

Finally attach the diagonal comparison. The diagonal readback theorem Definition 535.80 brings DiagonalLimitCompatibility$^{\uparrow}$ terminal reads to the same boundary after the root-window and route-exhaustion locks have fired. Therefore all four terminal consumers meet the identical finite source packet before reading any seal-facing route, and the excluded analytic or choice-like objects have no available row in the comparison. □

Theorem 535.85 (Bounded monotone Cauchy witness terminal located trap preservation)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet. Every* BoundedMonotoneConvergenceSeal$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*,* Real$^{\uparrow}$*, or* DiagonalLimitCompatibility$^{\uparrow}$ *terminal consumer that preserves the convergence-seal source also preserves the located trap row* I *as part of the finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

before any seal-facing or terminal comparison route is available.

No terminal route may replace I *by a selected limit, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier terminal located trap preservation`

Proof. First identify the source packet. Source totality, readback, non-escape, and source lock give the same finite carrier rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ for every convergence-seal source read and provide no external completion coordinate.

Next preserve the root window and the terminal route. The root-window lock keeps $S, F, \sigma, \mu, \lambda$ attached to the same accepted packet, while the terminal-source pullback and terminal-source equalizer force BoundedMonotoneConvergenceSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, and DiagonalLimitCompatibility$^{\uparrow}$ consumers to read that packet before any public completion comparison.

It remains to isolate the trap row. Since I is a displayed row of the packet equalized by Definition 535.84, a consumer that preserves the convergence-seal source preserves I along with Q, H, C, P, N. The excluded analytic and choice-like objects have no row in the equalized packet, so they cannot replace the located trap. □

Theorem 535.86 (Bounded monotone Cauchy witness L10 terminal route exhaustion)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet. Every terminal consumer preserving the convergence-seal source across*

BoundedMonotoneConvergenceSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegS

factors through exactly the finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

The exhausted route exports no quotient stream, selected limit, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier l10 terminal route exhaustion`

Proof. First equalize the source. The source-readback equalizer, Definition 535.83, identifies the source totality row with the convergence-seal readback row on the same finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next attach the terminal source route. By Definition 535.84, every BoundedMonotoneConvergenceSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, and DiagonalLimitCompatibility$^{\uparrow}$ terminal consumer reaches that packet before any terminal comparison is available.

Now preserve the located trap and support rows. The located-trap preservation theorem Definition 535.85 keeps I inside the same source packet, while the convergence-seal route exhaustion and diagonal readback theorems display the BHist, hsame, Cont, Pkg, and NameCert rows used by the public seal and diagonal compatibility routes.

Finally compare the listed consumers. The public seal route already passes through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the diagonal readback adds $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ without adding an external coordinate. Therefore the whole terminal consumer chain is exhausted by the displayed finite packet and support rows, with no available row for any excluded analytic, quotient, compactness, or choice-like object. □

Theorem 535.87 (Bounded monotone Cauchy witness L10 terminal route refusal completeness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. A terminal consumer request across*

$\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegS}$

is admissible exactly when it factors through the finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Any request for quotient stream equality, selected limits, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection is off-route, because no displayed carrier row supplies it.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier l10 terminal route refusal completeness`

Proof. The forward direction is the route-exhaustion direction. Source-readback equalization first identifies every convergence-seal source read with the same accepted packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next the terminal-source equalizer transports that packet to the $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ consumers before any terminal comparison can be read. Terminal located-trap preservation keeps I in the same packet and keeps Q, H, C, P, N attached to the displayed source.

Now apply the L10 terminal route exhaustion theorem. It adds the $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route and records that the only exported support rows are the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Hence an accepted terminal consumer is admissible only by factoring through that finite route.

For the converse, any consumer already factored through the finite packet and the displayed support rows is one of the terminal reads permitted by the exhausted route. It therefore uses only rows supplied by the accepted packet, the history/support equality rows, the continuity row, the package row, and the certificate row.

Finally consider the refused requests. Quotient stream equality, selected limits, host real equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selection are not entries of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and are not supplied by BHist, hsame, Cont, Pkg, or NameCert. The terminal chain has no further carrier row, so each such request is off-route. □

Theorem 535.88 (BoundedMonotoneCauchyWitness convergence-seal Real exit exhaustion)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*, diagonal-limit, finite-tail-filter, or completion-facing read of the bounded monotone witness convergence seal factors through the displayed source row, finite-tail-filter handoff, selector terminal pullback, criterion tail factorization, Real seal public unblock, source-readback equalizer, and terminal-route refusal-completeness rows.*

No such read may use a selected limit, quotient stream, host real equality, countable choice, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal real exit exhaustion`

Proof. Begin at the convergence-seal source. The source row and finite-tail-filter handoff attach the accepted packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ to the displayed convergence-seal route before any Real-facing or completion-facing consumer can read it.

Next use the selector terminal pullback and criterion-tail factorization. These force the selector, $\mathsf{CauchyCriterion}^{\uparrow}$, finite-tail-filter, and tail-factor reads to return to the same packet rather than detaching a selected limit or a quotient stream.

Now apply the Real-seal public unblock. It identifies the public $\mathsf{Real}^{\uparrow}$ route as a read of the same convergence seal and keeps the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows attached to the finite packet.

Finally compose the source-readback equalizer with terminal-route refusal completeness. The equalizer makes the source and readback routes the same displayed packet, while refusal completeness accepts exactly the finite terminal route and rejects selected limits, quotient streams, host real equality, countable choice, and ambient completeness rows. □

Corollary 535.89 (Bounded monotone Cauchy witness convergence-seal strict local obstruction)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ packet. Suppose a displayed failure, refusal, separation, or countermodel row is cited against a named converse or reuse route for the convergence-seal consumer surface of W. Then the cited row is a strict local obstruction: it blocks only the named route whose consumer-facing use tries to read the convergence seal, finite-tail handoff, selector readback, or completion-facing output away from the displayed route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and away from the public route through

$$\mathsf{MonotoneCauchy}^{\uparrow}, \mathsf{CauchyCriterion}^{\uparrow}, \mathsf{FiniteTailFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyLimitSeal}^{\uparrow},$$

Every admitted consumer-facing use factors through the source boundary, criterion-tail factorization, finite-tail-filter handoff, Real-seal public route, readback equalizer, and terminal-route refusal-completeness rows already displayed for W. The same cited row cannot be reinterpreted as a global impossibility theorem for bounded monotone convergence, a host theorem about real completion, hidden external validation of the failed route, or an obstruction to any consumer route that still factors through the displayed rows of W.

Proof. Start with the source boundary, Definition 535.65. It confines every convergence-seal source read to the finite witness rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and states that the source exports no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector. Thus any consumer-facing negative use has to be attached to one of the displayed packet coordinates, not to an ambient completion object.

Apply the finite-tail and route-factorization rows. By Definition 535.66, every convergence-seal or Real-completion consumer first reads the displayed witness packet, then passes through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$, then through the $\mathsf{FiniteTailFilter}^{\uparrow}$ packet, and only then continues through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. By Definition 535.76, this is the whole public convergence-seal route.

Now read the cited negative row against a named converse or reuse route. If the named route tries to bypass the source row, detach the Cauchy witness or dyadic ledger, reuse the finite-tail handoff as an unrecorded selector, or replace the Real-facing seal by a completion principle, Definition 535.74 forces the attempted consumer back into the same finite route $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. The cited failure, refusal, separation, or countermodel row therefore blocks exactly the named coordinate where that route tries to omit, replace, detach, or bypass the displayed packet.

The readback side gives the same conclusion. By Definition 535.83, the forward source read and the reverse convergence-seal readback are equalized on the displayed packet before any completion-facing route is inspected. By Definition 535.87, an admissible terminal request is exactly one that factors through the finite packet and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows; requests for quotient stream equality, selected limits, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection are off-route.

Any stronger reading would require a coordinate absent from the accepted packet. A global impossibility theorem for bounded monotone convergence would require a theorem row not indexed by the local packet W. A host theorem about real completion would require host equality, ambient completeness, or quotient data outside the displayed route. Hidden external validation would require an external certificate channel beyond P, N, and the displayed consumer rows. The source theorem and non-escape theorem exclude exactly those coordinates. Hence the obstruction is strict and local: it blocks only the named converse or reuse route and leaves every consumer route that remains inside the displayed packet available. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal strict local obstruction`

Theorem 535.90 (Bounded monotone Cauchy witness convergence-seal input package)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. Every downstream convergence-seal input exported by W factors through the displayed rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

before it is read by the BoundedMonotoneConvergenceSeal$^{\uparrow}$ *consumer of chapter 918.*

The factorization presents the bounded source through the public MonotoneCauchy$^{\uparrow}$ *row, the explicit witness through the public* CauchyCriterion$^{\uparrow}$ *row, the window schedule through* StreamName$^{\uparrow}$*, the dyadic tolerance and upper-bound ledger through* DyadicRatCore$^{\uparrow}$*, the regular rational handoff through* RegSeqRat$^{\uparrow}$*, the compatibility boundary through* CauchyLimitSeal$^{\uparrow}$*, and the sealed real-facing row through* Real$^{\uparrow}$*. It exports no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.*

Proof. Begin with the local dependency package. By Definition 535.19, every public read of W is already confined to the displayed bounded-source, regular-rational, stream, Cauchy-witness, dyadic, located-trap, Real-seal, transport, continuation, provenance, and naming rows.

Next pass through the public tail envelope. Definition 535.57 identifies $S, F, \sigma, \mu, \lambda, I, H, C, P, N$ as the pre-seal service and places Q after that service as the Real$^{\uparrow}$ seal row. The Real-completion export of Definition 535.44 keeps the same rows visible to Real-facing consumers.

Finally compare with the downstream seal chapter. Its carrier reads a bounded witness, public monotone tail-window row, public Cauchy criterion row, regular-rational handoff, stream schedule, dyadic ledger, limit-seal compatibility boundary, and Real seal. The local package above supplies exactly those inputs from W, so the convergence-seal consumer has no route around the accepted witness packet. □

Lean4 ✓ BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal input package

Theorem 535.91 (Bounded monotone Cauchy witness seal-input transport)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P,$ *and* $W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$ *be accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carriers. If the component rows are related by the displayed* hsame *transports in* H, H' *and the continuation routes in* C, C' *preserve the same* Cont *reads, then the convergence-seal input package of Definition 535.90 is preserved componentwise.*

Thus transport may replace a row only by its corresponding transported row $S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N'$*. It cannot create a detached convergence-seal input, a second compatibility boundary, a selected limiting representative, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or a choice-like tail selector.*

Proof. Start from classifier transport. The local transport theorem for W moves only the displayed carrier coordinates and keeps the source, witness, trap, Real-seal, continuation, provenance, and naming sockets componentwise aligned.

Apply the convergence-seal input package to both carriers. For W, every exported seal input is read through $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$; for W', every exported seal input is read through $S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N'$.

Use the assumed hsame and Cont preservation to compare matching coordinates. Since the package has no coordinate outside the displayed rows, componentwise transport preserves the whole package and has no source from which an extra seal input or excluded completion principle could be produced. □

Lean4 ✓ BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier seal input transport

Theorem 535.92 (Bounded monotone Cauchy witness completion-route exhaustion)**.** *For every accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, any completion-facing route that can be consumed by* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *is exhausted by the witness package of Definition 535.90. In particular, the route cannot bypass* W *to selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selectors.*

The only available route to the public convergence handoff is the finite chain from the bounded source and witness rows through the public MonotoneCauchy$^{\uparrow}$*,* CauchyCriterion$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows recorded by the downstream seal chapter.*

Proof. First use completion-consumer non-escape. Definition 535.58 says that every completion-facing consumer factors through the public envelope, finite-read route, regular-sequence handoff, located trap, and Real$^{\uparrow}$ seal of the accepted witness packet.

Then apply the convergence-seal input package. It identifies the portion of that route read by chapter 918: the bounded source, public monotone tail-window certificate, public Cauchy criterion packet, stream and dyadic ledgers, regular handoff, compatibility boundary, and Real seal are all supplied from $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Finally use the Real-completion export and public tail-envelope export. Together they leave no public row outside the accepted witness package. A route asking for a selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector therefore asks for data absent from the completion-facing surface. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier completion route exhaustion`

Theorem 535.93 (Bounded monotone Cauchy witness finite seal-route determinacy)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *and* $W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$ *be accepted* BoundedMonotoneCauchy *carriers. Suppose the displayed source, witness, monotonicity, modulus, localization, interval, seal, transport, continuation, provenance, and naming rows agree componentwise through the* hsame *data in* H, H'*, and suppose the displayed* Cont *routes in* C, C' *preserve the same reads of those rows. Then every* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *consumer obtains the same finite seal-read surface from* W *and* W'*.*

The determined surface is exactly the finite route through MonotoneCauchy$^{\uparrow}$*,* CauchyCriterion$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$ *recorded by the displayed rows. It does not compare selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selectors.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier finite seal route determinacy`

Proof. First apply Definition 535.90 to both carriers. It confines every downstream convergence-seal read to the displayed rows of the accepted packet and identifies the public route through the monotone, Cauchy, stream, dyadic, regular-rational, limit-seal, and Real rows.

Next use Definition 535.91. The assumed componentwise hsame data and the matching displayed Cont routes transport each row of W to the corresponding row of W' without adding a detached convergence-seal input or a second compatibility boundary.

Finally apply Definition 535.92. Since every completion-facing route consumable by BoundedMonotoneConvergen is exhausted by the same finite chain, there is no remaining coordinate on which the two accepted packets can differ at the seal-read surface.

The excluded objects are outside the route just used. A selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector would require data absent from both displayed packets, so none can distinguish the two downstream reads. □

Theorem 535.94 (Bounded monotone Cauchy witness convergence-seal read determinacy)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *and* $W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$ *be accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carriers. Suppose the displayed bounded source, monotone window, Cauchy criterion, stream schedule, dyadic ledger, regular-rational handoff,* CauchyLimitSeal$^{\uparrow}$ *compatibility,* Real$^{\uparrow}$ *seal, transports, continuations, provenance, and names agree componentwise through the* hsame *data in* H, H' *and the matching* Cont *reads in* C, C'*. Then every* BoundedMonotoneConvergenceSe *consumer receives the same downstream finite seal-read surface from* W *and* W'*.*

Equivalently, the read is determined by the finite witness-to-seal route through MonotoneCauchy$^{\uparrow}$*,* CauchyCriterion$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$*. No selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector can refine or distinguish that read.*

Proof. First confine the consumer route. Apply Definition 535.90 to both accepted carriers. The theorem says that any downstream seal input exported from either carrier is already read through the displayed witness rows and then through the public monotone, Cauchy, stream, dyadic, regular-rational, limit-seal, and Real rows.

Next transport the displayed rows. By Definition 535.91, the assumed hsame comparisons and matching Cont reads preserve the rows of the package componentwise. Thus the two carriers present

the same bounded source, window, criterion, schedule, ledger, handoff, compatibility boundary, Real seal, provenance, and naming surface to the downstream consumer.

It remains to rule out a second place where the consumer could read. Use Definition 535.92 to exhaust all completion-facing routes by the same witness package, and then apply Definition 535.93 to identify the finite seal-read surfaces. The excluded completion objects are not coordinates of that route, so they cannot separate the two downstream reads. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal read determinacy`

Theorem 535.95 (Bounded monotone Cauchy witness seal-route minimality)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. A* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer route is an accepted seal-input route from* W *only when it carries the displayed source row, monotone window row, Cauchy criterion row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row,* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, continuation rows, provenance row, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row.*

Consequently any proper subroute that omits one of those rows is not an accepted seal-input route for the downstream convergence seal. This minimality claim is only about the finite witness-to-seal input route; it does not assert Real completeness, a selected limit, quotient real equality, trichotomy, open-cover compactness, or any choice-like tail selection.

Proof. First take the accepted read inventory from Definition 535.90. It lists the bounded source, public monotone and Cauchy rows, stream schedule, dyadic ledger, regular-rational handoff, compatibility boundary, Real seal, transports, continuations, provenance, and naming row as the complete input package read by the convergence-seal consumer.

Next use componentwise preservation. Definition 535.91 says that accepted transport may replace a displayed row only by its matching transported row, so a route missing one coordinate cannot be repaired by a detached replacement.

Now rule out escape routes. By Definition 535.92, every completion-facing route consumable by $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ is exhausted by the same finite witness package. There is no second path from which an omitted source, window, criterion, schedule, ledger, handoff, compatibility, seal, transport, continuation, provenance, or naming row could be supplied.

Finally apply read determinacy from Definition 535.94. Accepted reads of the displayed route are equal at the downstream finite seal surface, so removing a displayed row leaves no accepted read of that surface. The excluded completion principles are not coordinates of the route and therefore play no role in the minimality statement. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier seal route minimality`

Theorem 535.96 (Bounded monotone Cauchy witness seal-route universal property)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any finite consumer route from the accepted packet* W *to a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *seal input factors through the displayed witness-to-seal route*

$$\mathsf{MonotoneCauchy}^{\uparrow},\ \mathsf{CauchyCriterion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The factorization is unique at the finite seal-read surface: two accepted factorizations with the same displayed source, monotone window, Cauchy criterion, stream schedule, dyadic ledger, regular-rational handoff, limit-seal compatibility row, Real seal, transport rows, continuation rows,

provenance row, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row induce the same downstream read, and any candidate omitting one of these rows is not an accepted route. The statement does not assert Real completeness, a selected limit, quotient real equality, host equality, trichotomy, open-cover compactness, or any choice-like tail selector.*

Proof. Existence of the factorization is the input-package statement. By Definition 535.90, every convergence-seal input exported by W is first read through the displayed packet rows and then through the public monotone, Cauchy, stream, dyadic, regular-rational, limit-seal, and Real rows.

The factorization is stable under the only permitted replacements. Apply Definition 535.91: componentwise hsame transport and matching Cont reads preserve exactly the same displayed rows and do not create a detached seal input or a second compatibility boundary.

There is no competing completion-facing route. Definition 535.92 exhausts every route consumable by $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ by the witness package just described.

Uniqueness at the finite seal-read surface follows by combining Definition 535.93 with Definition 535.94. These identify any two accepted reads of the displayed route after componentwise agreement, so a second accepted factorization has the same downstream finite surface.

Finally use Definition 535.95. A route that omits a displayed row is not accepted, and the excluded completion principles are not coordinates of the finite route. Hence the displayed route has the stated universal property among finite seal-input consumers of W. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier seal route universal property`

Corollary 535.97 (Bounded monotone Cauchy witness convergence-seal consumer determinacy)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Fix the displayed bounded-source, monotone-window, Cauchy-witness, stream-schedule, dyadic-ledger, regular-rational, located-trap,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *rows of* W*. Then any accepted consumer-facing use of* W *by a* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer has the same finite seal route, the same downstream normal-form read, and the same exactness decision named by this chapter.*

Equivalently, every accepted use is forced through the finite route

$$\mathsf{MonotoneCauchy}^{\uparrow},\ \mathsf{CauchyCriterion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

There is no alternate hidden coordinate, duplicate classifier source, selected-limit coordinate, quotient real equality, host real equality, trichotomy, ambient completeness principle, open-cover compactness principle, or choice-like tail selector that can refine the accepted read.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal consumer determinacy`

Proof. Start with the displayed input package. By Definition 535.90, every downstream convergence-seal input exported by W factors through

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

before the $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ consumer reads it. The same theorem identifies the public route as the monotone row, Cauchy-criterion row, stream schedule, dyadic ledger, regular-rational handoff, limit-seal compatibility boundary, and $\mathsf{Real}^{\uparrow}$ seal.

Now compare two accepted consumer-facing uses of the fixed rows. Treat the second use as another presentation of the same displayed packet: the fixed rows give the componentwise hsame

data, and the fixed continuation rows give the matching Cont reads. The transport theorem Definition 535.91 then permits replacement only by the corresponding transported displayed row. It does not create a detached seal input, a second compatibility boundary, or a duplicate classifier source.

Apply finite route determinacy from Definition 535.93. Since the source, witness, monotonicity, modulus, localization, interval, seal, transport, continuation, provenance, and naming rows are fixed, every accepted convergence-seal consumer obtains the same finite seal-read surface. Therefore a consumer that names the route directly, reads it as a normal form, or records its exactness decision is naming the same determined surface.

Finally use the exhaustion clauses already present in the local route. Definition 535.92 and Definition 535.94 rule out a second completion-facing place where the consumer could read. The excluded selected-limit, equality, trichotomy, completeness, compactness, and choice-like objects are not coordinates of the displayed packet, so none can supply an alternate decision or a hidden classifier source. □

Theorem 535.98 (Bounded monotone Cauchy witness source-row exhaustion)**.** *For an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N),$$

the bounded monotone source row, regular rational handoff, stream schedule, Cauchy modulus, dyadic ledger, located trap, and Real$^{\uparrow}$ *seal precursor exhaust the witness source surface. Every source-facing read is one of* $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$*, with no additional range over host real histories, selected limit, quotient equality, trichotomy row, ambient completeness witness, or open-cover compactness row.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier source row exhaustion`

Proof. Project the carrier surface of Definition 535.5. Its rows list the bounded source, regular rational handoff, stream schedule, Cauchy witness, dyadic ledger, located trap, Real seal precursor, componentwise transport, continuation route, package provenance, and local name.

Apply the local NameCert and scoped-dependency packages, Definition 535.6 and Definition 535.19. These force generator reads through S, tail requests through μ, σ, λ, located enclosures through I, classifier transport through H, C, and provenance and naming through P, N.

The child certificate-surface theorems then close the source boundary. Generator range exactness, tail modulus scope, Real seal boundary, and ledger exhaustion leave no carrier coordinate outside the displayed witness package, so the listed rows exhaust the source surface. □

Theorem 535.99 (Bounded monotone Cauchy witness tail-window lock)**.** *Every tail-window read exported by an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier factors through* MonotoneCauchy$^{\uparrow}$ *and* RegularCauchyTailMeet$^{\uparrow}$ *before any* Real$^{\uparrow}$ *seal row is read. The tail-window route is therefore the public monotone Cauchy source together with the regular rational tail handoff, not a direct jump from bounded monotonicity to a real-seal coordinate.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail window lock`

Proof. Start from the dependency paragraphs of this chapter. They place BoundedMonotoneCauchyWitness$^{\uparrow}$ as the bounded-monotone consumer of chapter 438 and the shared finite tail route of chapter 560.

Use the scoped-dependency package and the root extraction route. The displayed witness first exposes $S, F, \sigma, \mu, \lambda$ as the monotone source, regular handoff, schedule, Cauchy witness, and dyadic ledger before the located trap and Real seal suffix can be consumed.

Finally apply the convergence-seal source and input-package theorems already input by this chapter. They identify the downstream seal source with the same public monotone Cauchy and regular-tail rows, so a Real-facing consumer cannot read Q before the MonotoneCauchy and RegularCauchyTailMeet routes have been displayed. □

Theorem 535.100 (Bounded monotone Cauchy witness convergence-seal input totality)**.** *The* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *consumer of chapter 918 consumes exactly the displayed witness package*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

from an accepted BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier. It receives no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or alternate tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal input totality`

Proof. Read the downstream dependency paragraph together with the scoped-dependency package. The convergence seal asks for a finite monotone source, regular rational handoff, stream schedule, Cauchy modulus, dyadic ledger, located trap, Real seal precursor, and the structural rows carrying transport, continuation, provenance, and naming.

Apply Definition 535.90. It states that every downstream convergence-seal input exported by W factors through precisely the displayed witness rows before the seal chapter reads it.

The completion-route exhaustion and Real-seal public route exhaustion theorems rule out bypasses. Since selected limits, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, and alternate tail selectors are absent from the witness carrier and from the downstream route, the displayed package is total for the convergence-seal input. □

Theorem 535.101 (Bounded monotone Cauchy witness convergence-seal route determinacy)**.** *For an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N),$$

there is exactly one public convergence-seal route determined by its source, tail, and seal rows. The route first reads the MonotoneCauchy$^{\uparrow}$ *source and the* RegularCauchyTailMeet$^{\uparrow}$ *tail window, then passes through* CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, *and finally* Real$^{\uparrow}$. *A consumer-visible convergence-seal route for W cannot skip any one of those rows, nor can it replace them by selected limits, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or an alternate tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence seal route determinacy`

Proof. Start with source-row exhaustion. By Definition 535.98, every source-facing read of W is one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the convergence-seal route has no second source socket.

Next use the tail-window lock. It forces every tail-window read through the public MonotoneCauchy$^{\uparrow}$ and RegularCauchyTailMeet$^{\uparrow}$ surfaces before the Real$^{\uparrow}$ seal row is available.

Now attach convergence-seal input totality. The theorem Definition 535.100 says that the $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ consumer receives exactly the displayed witness package. Together with Definition 535.31 and Definition 535.90, this inserts the $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows in the displayed order.

It remains to rule out alternate public routes. The completion-route exhaustion theorem Definition 535.92 and ledger exhaustion in Definition 535.13 leave no consumer-visible coordinate outside the named rows. Any route omitting one of them therefore fails to be a convergence-seal route exported by the accepted packet W. □

Theorem 535.102 (Bounded monotone Cauchy witness tail Real-seal route totality)**.** *For an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-seal consumer of* W *passes through the source row, the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail source, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *window,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and the final* $\mathsf{Real}^{\uparrow}$ *row. No such consumer may replace this route by a selected limit, quotient equality, host real equality, trichotomy, open-cover compactness, or alternate tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail real seal route totality`

Proof. Begin with source-row exhaustion. The theorem Definition 535.98 identifies every source-facing read of W with one of the displayed coordinates $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next apply the tail-window lock. By Definition 535.99, a Real-facing tail read must first expose the $\mathsf{MonotoneCauchy}^{\uparrow}$ source and the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ window, so the tail surface cannot be selected outside the displayed packet.

Now use convergence-seal input totality and route determinacy. Theorems Definition 535.100 and Definition 535.101 force the downstream seal route through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in the named order.

It remains to exhaust the visible coordinates of the route. The source theorem accounts for the displayed BHist rows, the tail and seal theorems account for the hsame and Cont rows, and the local package and certificate rows provide exactly the Pkg and $\mathsf{NameCert}$ witnesses attached to P and N. Since the carrier contains no coordinate for a selected limit, quotient equality, host real equality, trichotomy, open-cover compactness, or alternate tail selector, none of those objects can serve as a Real-seal route for W. □

Theorem 535.103 (Bounded monotone Cauchy witness tail seal row determinacy)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose two downstream convergence-seal reads preserve the same source row* S*, the same public* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail source, the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *window, the same* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and the same structural rows* H, C, P, N*. Then the two reads determine the same visible* $\mathsf{Real}^{\uparrow}$ *seal row* Q*.*

Equivalently, the displayed tail route fixes the downstream seal row: no accepted read preserving those rows may replace Q *by a selected limit, quotient equality, host real equality, trichotomy branch, open-cover compactness witness, ambient completeness object, or alternate tail selector.*

Proof. First use source-row exhaustion. By Definition 535.98, every source-facing coordinate of W is one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Thus two reads preserving the same source row cannot choose a second source socket from outside the displayed packet.

Next apply the tail-window lock and route determinacy. The tail-window lock forces both reads through the same public $\mathsf{MonotoneCauchy}^{\uparrow}$ source and the same $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ window, while Definition 535.101 fixes the sibling route through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use the Real-seal route totality. The terminal $\mathsf{Real}^{\uparrow}$ read is a carrier coordinate of the same accepted packet, and the structural rows H, C, P, N are preserved by hypothesis. The coordinate exhaustion in Definition 535.102 leaves Q as the only visible terminal seal row and excludes selected limits, quotient equality, host real equality, trichotomy, open-cover compactness, ambient completeness, and alternate tail selectors. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail seal row determinacy`

Theorem 535.104 (Bounded monotone Cauchy witness Cauchy-ledger readback exactness)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Any* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *readback that consumes the bounded monotone witness factors through the same* $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route fixed by Definition 535.103. Such a readback cannot introduce a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, or alternate completion route.*

Proof. First read the source surface. By Definition 535.98, every source-facing coordinate available to the ledger is one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next lock the tail and seal route. The tail-window theorem forces the read through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, while convergence-seal input totality and route determinacy insert $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in the displayed order.

Finally apply tail seal row determinacy. Once the ledger preserves the same source, tail, seal, transport, route, provenance, and naming rows, the visible $\mathsf{Real}^{\uparrow}$ seal row is fixed as Q. Coordinate exhaustion leaves no socket for a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, or alternate completion route, so the ledger readback is exact. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier cauchy ledger readback exactness`

Theorem 535.105 (Bounded monotone Cauchy witness criterion-ledger route totality)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{CauchyCriterion}^{\uparrow}$ *consumer or* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *consumer that reads the bounded monotone witness factors through the same route fixed by Definition 535.104: the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *source, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *tail window,* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the structural rows* H, C, P, N.

Consequently the criterion-ledger route for W is total on the displayed packet coordinates. It has no selected limit, quotient stream equality, host real equality, choice row, hidden tail oracle, ambient completeness object, open-cover compactness row, or alternate completion route.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier criterion ledger route totality`

Proof. Start with source-row exhaustion. Theorem 535.98 identifies every source-facing read of W with one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so a criterion or ledger consumer has no carrier coordinate outside the displayed packet.

Next lock the tail window. Theorem 535.99 routes every tail-facing read through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ before any terminal $\mathsf{Real}^{\uparrow}$ seal row is available.

Now use convergence-seal route determinacy and tail seal row determinacy. Theorems 535.101 and 535.103 fix the passage through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and they fix the visible seal row once H, C, P, N are preserved.

Finally apply the Cauchy-ledger readback exactness theorem. The readback theorem identifies the ledger-facing route with the same displayed route, while the preceding coordinate exhaustion accounts for all rows of W. There is therefore no remaining socket for a selected limit, quotient stream equality, host real equality, choice row, hidden tail oracle, ambient completeness object, open-cover compactness row, or alternate completion route. □

Theorem 535.106 (Bounded monotone Cauchy witness selector-free Real route)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing route through the bounded monotone witness is exhausted by the displayed source row, the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail source, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *tail window,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the structural rows H, C, P, N. No such route can introduce a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, or alternate completion route.*

Proof. Start with source-row exhaustion. Theorem 535.98 identifies every source-facing read of W with one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so a Real-facing consumer has no second carrier surface outside the displayed packet.

Next apply the tail-window lock. Theorem 535.99 routes each tail-facing read through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ before any terminal $\mathsf{Real}^{\uparrow}$ row can be read.

Now use convergence-seal route determinacy. Theorem 535.101 fixes the sibling chain through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order.

Then apply tail seal row determinacy. Theorem 535.103 fixes the visible $\mathsf{Real}^{\uparrow}$ seal row once the same source, tail, seal, transport, continuation, provenance, and naming rows are preserved.

Finally use Cauchy-ledger readback exactness. Theorem 535.104 identifies the ledger-facing route with the same displayed route, and Definition 535.105 accounts for the criterion side. The named source, tail, seal, and structural rows therefore exhaust the Real handoff, leaving no socket for a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, or alternate completion route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier selector free real route`

Theorem 535.107 (Bounded monotone Cauchy witness dyadic-radius budget preservation)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing route preserving the bounded monotone source* S *and the regular tail rows* σ, μ, λ *also preserves the dyadic radius budget through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and the final* $\mathsf{Real}^{\uparrow}$ *seal row* Q*. Such a route cannot replace that budget by a selected limit, host real equality, quotient equality, trichotomy branch, open-cover compactness row, or alternate completion route.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier dyadic radius budget preservation`

Proof. Start with source-row exhaustion. Theorem 535.98 confines every source-facing read to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so preservation of S and the regular tail rows already fixes the carrier surface on which the dyadic budget can be read.

Next apply the tail-window lock. Theorem 535.99 routes the preserved tail data through $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ before a $\mathsf{Real}^{\uparrow}$-facing read can reach the terminal seal.

Use convergence-seal route determinacy and tail seal row determinacy. Theorems 535.101 and 535.103 fix the passage through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and they fix the visible seal row Q once the source, tail, transport, continuation, provenance, and naming rows are preserved.

Finally apply the selector-free Real route. Theorem 535.106 exhausts every $\mathsf{Real}^{\uparrow}$-facing route by the same displayed source, tail, dyadic, regular-sequence, limit-seal, Real-seal, and structural rows. Coordinate exhaustion therefore leaves no route on which a selected limit, host real equality, quotient equality, trichotomy branch, open-cover compactness row, or alternate completion route could replace the dyadic radius budget. □

Theorem 535.108 (Bounded monotone Cauchy witness root tail-budget exactness)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every accepted root-source tail-seal read of* W *factors through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *rows before any* $\mathsf{Real}^{\uparrow}$ *or completion-facing endpoint is public.*

Thus the monotone witness rows determine the same finite L10 tail-budget chain as the root source rows: no accepted read may replace that chain by a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, open-cover compactness row, or alternate completion route.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root tail budget exactness`

Proof. Start with the local source rows. Theorems 535.98, 535.99, and 535.107 bind the source row S, regular handoff F, scheduled windows σ, Cauchy modulus μ, dyadic ledger λ, located trap I, and Real seal row Q to one displayed packet.

Next read the witness and criterion ledgers. Theorems 535.104 and 535.105 force every $\mathsf{CauchyWitnessLedger}^{\uparrow}$ or $\mathsf{CauchyCriterion}^{\uparrow}$ consumer through the same $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Now attach the root sibling routes listed by the chapter. The root unblock and tail-service rows identify the $\mathsf{CauchyModulusExtraction}^{\uparrow}$ schedule, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ synchronizer, $\mathsf{RealObservationBudget}^{\uparrow}$ finite observation route, and $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ handoff as consumers of the same source, tail, trap, seal, transport, continuation, provenance, and local naming rows H, C, P, N.

Finally apply selector-free Real-route exhaustion. Theorem 535.106 leaves no endpoint coordinate outside the displayed source, tail, ledger, limit-seal, Real-seal, and structural rows. Therefore any root-source tail-seal read reaches $\mathsf{Real}^{\uparrow}$ or a completion-facing endpoint only after traversing the stated finite tail-budget chain. □

Theorem 535.109 (Bounded monotone Cauchy witness convergence-grid fork coherence)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose one downstream read consumes* W *through* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *and another downstream read consumes* W *through* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$*. If both reads preserve the displayed source, tail, dyadic-radius, transport, continuation, provenance, and local naming rows, then they determine the same visible* $\mathsf{Real}^{\uparrow}$ *seal row* Q *and the same finite route through* $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$.

No such pair of reads can fork into selected limits, quotient equality, host real equality, choice rows, hidden tail oracles, ambient completeness objects, open-cover compactness rows, or alternate completion routes.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier convergence grid fork coherence`

Proof. Start with source-row exhaustion. Theorem 535.98 confines both consumers to the same packet coordinates $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so neither the convergence-seal side nor the grid side has an additional carrier coordinate on which a second terminal seal could be named.

Next lock the seal and tail rows. Theorems 535.101 and 535.103 force every preserved convergence-seal read and every preserved tail-seal read through the same public source, tail, dyadic, regular-sequence, limit-seal, and $\mathsf{Real}^{\uparrow}$ rows. In particular, the visible terminal row is Q once the transport, continuation, provenance, and local naming rows H, C, P, N are fixed.

Now compare the two consumer routes. Theorem 535.106 excludes a second $\mathsf{Real}^{\uparrow}$-facing route outside the displayed source, tail, dyadic, limit-seal, Real-seal, and structural rows, while Definition 535.107 preserves the dyadic-radius budget along the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ passage.

Finally apply root tail-budget exactness. Theorem 535.108 identifies the root-source tail-seal read with the finite route through $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$. Since both consumers read the same carrier coordinates and the same row Q, there is no remaining coordinate from which they could fork into distinct terminal $\mathsf{Real}^{\uparrow}$ seals or any of the excluded completion mechanisms. □

Corollary 535.110 (Bounded monotone Cauchy witness source-exhaust determinacy)**.** *Let*

$$W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$$

be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Once the displayed source rows* $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *and structural rows* H, C, P, N *are fixed, every accepted consumer-facing use of* W *that asks for a convergence-seal route, a* $\mathsf{Real}^{\uparrow}$*-seal normal form, a criterion-ledger readback, a root-source tail-seal chain, or a grid comparison named by this chapter determines the same finite source-exhaust normal form:*

$$\mathsf{MonotoneCauchy}^{\uparrow}, \quad \mathsf{RegularCauchyTailMeet}^{\uparrow}, \quad \mathsf{CauchyCriterion}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}$$

with terminal visible seal row Q *and with the same displayed tail-budget rows when the root-source tail-seal chain is requested. No accepted use preserving the cited local rows may introduce a hidden coordinate, a duplicate classifier source, a selected limit, quotient equality, host real equality, choice row, hidden tail oracle, ambient completeness object, open-cover compactness row, or alternate completion route as a competing normal form or exactness decision.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier source exhaust determinacy`

Proof. Start with source-row exhaustion. By Definition 535.98, every source-facing read of W is one of $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Hence an accepted consumer use whose local rows are fixed has no second source socket from which to read a hidden coordinate or duplicate classifier source.

Apply convergence-seal route determinacy. The theorem Definition 535.101 fixes the public convergence-seal route through $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It also excludes routes that skip one of these rows or replace the route by selected limits, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or an alternate tail selector.

The same fixed route controls the named consumer decisions. The $\mathsf{Real}^{\uparrow}$-seal normal form is exhausted by Definition 535.102, and Definition 535.103 fixes the visible terminal row as Q once the source, tail, seal, transport, continuation, provenance, and local naming rows are preserved. The criterion-ledger readback is exact by Definition 535.104, and the root-source tail-seal chain is the finite budget route of Definition 535.108. For paired convergence-seal and grid consumers, Definition 535.109 identifies the same visible $\mathsf{Real}^{\uparrow}$ seal row and the same finite dyadic, regular-sequence, and limit-seal passage.

Therefore every accepted consumer-facing use that preserves the cited local rows factors through the displayed source-exhaust normal form and the corresponding exactness decision already named by the chapter. Any proposed competing use would require either an additional source coordinate, ruled out by source-row exhaustion, or an alternate public route, ruled out by convergence-seal route determinacy and the named readback and seal determinacy theorems. □

Theorem 535.111 (Bounded monotone Cauchy witness root extraction)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The root extraction route read by any downstream* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *consumer is the finite* BHist *witness packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

Here S *is the displayed bounded monotone source,* F *is the regular rational row,* σ *is the stream schedule,* μ *is the Cauchy tail witness,* λ *is the dyadic tolerance and upper-bound ledger,* I *is the*

located interval trap, and Q is the $\mathsf{Real}^{\uparrow}$ *seal row. The rows H, C, P, N keep the extraction inside* hsame *transport,* Cont *continuation,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *naming.*

Consequently the root consumer reads a finite witness packet. It does not read a selected limit, quotient real, host real equality, trichotomy, ambient completeness claim, open-cover compactness row, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessRootExtraction`

Proof. Project the carrier of Definition 535.5. Its visible coordinates are exactly $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, and the local NameCert obligations force the source, witness, trap, seal, transport, continuation, provenance, and naming reads through those coordinates.

Use the window and finite-observation locks. By Definition 535.48 and Definition 535.50, any tail request first enters the displayed source S, regular handoff F, schedule σ, witness μ, and dyadic ledger λ before it can reach I or Q.

Finally apply the convergence-seal input package of Definition 535.90. That theorem identifies the downstream seal inputs with the same rows of W, so the root extraction has no coordinate from which an excluded completion object or choice-like selector could be produced. □

Theorem 535.112 (Bounded monotone Cauchy witness limit-seal handoff)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The handoff from W to every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The source row S, regular rational row F, stream schedule σ, tail witness μ, dyadic ledger λ, located trap I, and Real seal row Q are transported only through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *routes of the carrier.*

Thus a limit-seal consumer receives the compatibility boundary carried by the witness packet. It receives no selected limiting representative, quotient-real construction, host equality proof, trichotomy split, ambient completeness principle, or compactness certificate.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier limit seal handoff`

Proof. Start with the Real seal boundary and tail-seal synchronization. Theorems Definition 535.9 and Definition 535.27 place the seal-facing suffix after the same source, regular, schedule, witness, dyadic, and located-trap rows.

Next use the trap-seal exactness and seal-route naturality. The exactness theorem fixes I and Q to the scheduled tail envelope, while Definition 535.71 compares the source, criterion-tail, and Real-seal readings through the same public route.

The local transport rows close the handoff. Since H, C, P, N are the only carrier rows for hsame, Cont, Pkg, and NameCert movement, the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers can only read the finite witness packet and cannot import any excluded completion principle. □

Theorem 535.113 (Bounded monotone Cauchy witness monotone-bound route)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The monotone-bound route used by the bounded monotone convergence surface is the route from the displayed bounded source S through the regular rational row F, stream schedule σ, Cauchy tail witness μ, dyadic tolerance ledger λ, located*

interval trap I, and Real seal row Q, with H, C, P, N supplying hsame *transport,* Cont *continuation,* Pkg *provenance, and local* NameCert *naming.*

Every downstream consumer that asks for bounded monotone convergence through W therefore reads the same finite BHist *route. The route supplies monotonicity, upper-bound, regularity, Cauchy-tail, tolerance, trap, and seal evidence, but it does not supply selected limits, quotient real equality, host real equality, trichotomy, ambient completeness, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier monotone bound route`

Proof. Begin with the source row scope and generator range exactness. These identify S as the bounded monotone source coordinate and keep generator, monotonicity, and upper-bound reads inside the displayed source row.

Pass from the source to the tail service. The tail-window service and Cauchy modulus-extraction factorization route tolerance requests through F, σ, μ, λ, so the bounded source is connected to a regular rational Cauchy witness by displayed rows rather than by a hidden subsequence or selected limiting object.

Finally use the located trap factorization and Real seal schedule exhaustion. They place I and Q after the same dyadic and scheduled-tail route, while the ledger exhaustion theorem leaves no public coordinate beyond $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. The monotone-bound route is therefore a finite BEDC witness route and not an ambient completeness claim. □

Theorem 535.114 (Bounded monotone Cauchy witness root seal route exhaustion)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of W factors through exactly the displayed witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and through the sibling route chain

$$\mathsf{MonotoneCauchy}^{\uparrow} \longmapsto \mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyLimitSe}$$

Thus the root seal route is exhausted by the extraction, limit-seal handoff, monotone-bound route, and convergence-seal input package. It contains no selected limit, quotient real equality, host equality, trichotomy, compactness, ambient completeness, or choice-like tail selector.

The sibling route is the concrete L10 chain of chapter 126, chapter 291, chapter 301, chapter 386, and chapter 38; the witness packet consumes their finite-window, dyadic-ledger, regular-source, limit-seal, and Real-seal rows before any convergence-seal consumer reads it.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root seal route exhaustion`

Proof. First apply Definition 535.111. It projects the accepted carrier to the finite witness packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and rules out any root coordinate outside that packet.

Next apply Definition 535.112 and Definition 535.113. The handoff theorem places every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ read on the same finite packet, while the monotone-bound theorem routes the bounded monotone convergence surface through $S, F, \sigma, \mu, \lambda, I, Q$ with H, C, P, N supplying transport, continuation, provenance, and naming.

Finally apply Definition 535.90. That package identifies the completion-facing suffix with the sibling chain from $\mathsf{MonotoneCauchy}^{\uparrow}$ through $\mathsf{Real}^{\uparrow}$. Since all three root-facing consumer types

enter through the preceding finite rows and the package supplies the only seal-facing continuation, no selected limit, quotient equality, host equality, trichotomy, compactness, ambient completeness, or choice-like tail selector remains available. □

Theorem 535.115 (Bounded monotone Cauchy witness root tail-window totality)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing consumer of* W *that asks for a convergence-seal, limit-seal, or real-seal input must first consume the displayed* $\mathsf{StreamName}^{\uparrow}$ *tail window carried by* σ *together with the Cauchy witness row* μ *and the dyadic ledger* λ*.*

Thus the root route through W *has a total tail-window obligation row. It passes through the displayed source* S*, regular rational row* F*, scheduled window* σ*, witness* μ*, and ledger* λ *before any located trap or* $\mathsf{Real}^{\uparrow}$ *seal threshold can be read, and it contains no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root tail window totality`

Proof. Start with the carrier definition. The accepted packet exposes σ as the $\mathsf{StreamName}^{\uparrow}$ window schedule, μ as the Cauchy tail witness, and λ as the dyadic tolerance and upper-bound ledger; no separate tail service is a carrier coordinate.

Apply root extraction and the monotone-bound route. Definition 535.111 projects every root read to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, while Definition 535.113 places the bounded monotone source before the same scheduled tail service.

Finally use the limit-seal handoff and root seal route exhaustion. The handoff theorem keeps seal-facing consumers on the displayed packet, and Definition 535.114 leaves no route around the sibling chain through $\mathsf{StreamName}^{\uparrow}$. Therefore tail-window consumption is total for the root route and the excluded completion objects are not available. □

Theorem 535.116 (Bounded monotone Cauchy witness root located-trap stability)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing consumer that has passed through the displayed tail window must preserve the* $\mathsf{LocatedInterval}^{\uparrow}$ *trap row* I *as the next pre-seal enclosure. Transport through* H *and replay through* C *may move the displayed rows componentwise, but the root route cannot replace* I *by an ambient interval choice, host equality, quotient real equality, trichotomy, completeness, compactness, or a selected limiting representative.*

Consequently the located-trap obligation is stable under the accepted root route for $\mathsf{BoundedMonotoneConvergence}$ $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root located trap stability`

Proof. Read the located row from the carrier. The definition makes I the unique $\mathsf{LocatedInterval}^{\uparrow}$ trap row and places it after $S, F, \sigma, \mu, \lambda$ and before the seal row Q.

Use the root extraction theorem to confine the consumer surface. Any root consumer starts from the displayed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the only available movement of the trap row is the componentwise transport and continuation data already carried by H and C.

Next apply the limit-seal handoff and monotone-bound route. They both route seal-facing and convergence-facing reads through the same scheduled tail and located-trap suffix, so preserving the root route also preserves I as the pre-seal enclosure.

The root seal route exhaustion theorem then rules out replacement rows. Since no selected limit, quotient equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like trap selector is present in the exhausted packet, the located trap remains stable for all three root-facing consumer types. □

Theorem 535.117 (Bounded monotone Cauchy witness root Real-seal threshold exhaustion)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{Real}^{\uparrow}$ *threshold consumed from W is reached only after the displayed tail window σ, μ, λ and the located trap I have been consumed. The seal row Q is therefore a threshold route over the finite witness packet, not a selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.*

Equivalently, a root consumer cannot expose a $\mathsf{Real}^{\uparrow}$ *seal threshold from W unless it has already traversed the accepted packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

Proof. Begin with the carrier and the Real seal boundary. The carrier lists Q only after the regular rational row, scheduled window, Cauchy witness, dyadic ledger, and located trap, and the existing seal boundary keeps every $\mathsf{Real}^{\uparrow}$ read tied to that order.

Apply root extraction, then attach the limit-seal handoff. Root extraction supplies the finite packet, and the handoff theorem says that $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers read the same packet through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ routes.

Now use the monotone-bound route and root tail-window totality. The monotone-bound route places the source and bound data before F, σ, μ, λ, while Definition 535.115 makes the tail-window obligation unavoidable before any pre-seal or seal row is reached.

Finally apply located-trap stability and root seal route exhaustion. The trap theorem fixes I as the pre-seal enclosure, and the exhaustion theorem leaves the seal-facing suffix with no coordinate outside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Hence the $\mathsf{Real}^{\uparrow}$ threshold consumption is exhausted by the displayed root route and exports none of the excluded completion principles. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessRootRealSealThresholdEx`

Theorem 535.118 (Bounded monotone Cauchy witness root selector bridge)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Any root-facing consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal row Q must traverse, in order, the selector budget supplied by the* $\mathsf{StreamName}^{\uparrow}$ *schedule σ, the dyadic tolerance ledger λ, the located trap I, and the* $\mathsf{Real}^{\uparrow}$ *seal threshold Q.*

Thus the root selector bridge is the ordered finite route through the StreamName schedule, dyadic ledger, located trap, and Real seal threshold. It exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. Begin with root extraction. Theorem 535.111 confines every root-facing read of W to the displayed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the bridge has no coordinate outside the accepted carrier.

Next force the selector budget. By Definition 535.115, every root-facing seal consumer must pass through the $\mathsf{StreamName}^{\uparrow}$ schedule σ together with the Cauchy witness row μ and the dyadic ledger λ before any located-trap or seal row is read.

Now stabilize the trap. The located-trap stability theorem, Definition 535.116, keeps I as the pre-seal enclosure after the scheduled tail and dyadic tolerance rows have been consumed.

Finally apply Definition 535.117. Since the Real-seal threshold is reached only at Q after σ, μ, λ and I, any consumer reaching Q has traversed the selector budget, dyadic tolerance ledger, located

trap, and threshold in that order. The same exhaustion removes selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selectors from the bridge. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root selector bridge`

Theorem 535.119 (Bounded monotone Cauchy witness grid route compatibility)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and suppose its root route consumes an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet* $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N$ *Then the root route must read the selector budget* B*, witness ledger* L*, completion-seal classifier* K*, and tail-witness envelope* T *of* G *before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* Q *of* W*.*

Thus the bounded monotone witness does not run a completion route parallel to the grid. Its Real-facing threshold is compatible with the grid carrier of Definition 905.1: selector exactness, witness-ledger accounting, classifier exposure, and tail-envelope readback are consumed before the final $\mathsf{Real}^{\uparrow}$ *row is available.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid route compatibility`

Proof. Project the accepted witness packet by Definition 535.111. This confines the root route to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and leaves no completion-facing coordinate outside the displayed packet.

Attach the limit-seal handoff and monotone-bound route. Theorems Definition 535.112 and Definition 535.113 place every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ read after the source, regular, schedule, witness, dyadic, and located-trap rows.

Now read the grid packet. By Definition 905.3 and Definition 905.4, the grid route reaches $\mathsf{Real}^{\uparrow}$ only after $B, S_G, R_G, L, T, K, E_G, H_G, C_G, P_G, N_G$ have been consumed.

Finally use Definition 535.117. Since the threshold row Q is reached only at the seal end of the bounded-monotone route, a route that consumes G must first pass through the grid selector budget, witness ledger, completion classifier, and tail envelope. □

Theorem 535.120 (Bounded monotone Cauchy witness grid-seal cofinal route)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and suppose its root route consumes an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet* $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$ *and reaches the* $\mathsf{Real}^{\uparrow}$ *seal threshold* Q*. Then the selector budget* B*, witness ledger* L*, completion-seal classifier* K*, tail envelope* T*, located trap* I*, and Real seal* Q *are read through one cofinal finite* BHist *route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N, \quad B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G.$$

No selected limit, quotient real equality, host equality, trichotomy, ambient completeness, compactness, or choice-like selector is exported by this cofinal route.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid seal cofinal route`

Proof. First extract the root route. Apply Definition 535.111; every root-facing read is confined to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next attach the grid route. By Definition 535.119, the grid selector budget, witness ledger, completion-seal classifier, and tail envelope are consumed before the bounded-monotone route reaches Q.

Now force the tail and trap order. The root tail-window totality theorem places σ, μ, λ before the pre-seal rows, and Definition 535.116 keeps I as the displayed located trap rather than an ambient interval choice.

Finally apply Definition 535.117. The Real seal threshold is exhausted at Q after the tail and trap rows have been traversed, so the composed grid-seal readback is one finite cofinal route and exports none of the excluded completion, equality, compactness, or choice-like data. □

Theorem 535.121 (Bounded monotone Cauchy witness grid ledger non-escape)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier whose root route consumes an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$. The grid handoff exports only the displayed selector-budget, witness-ledger, completion-seal classifier, tail-witness, StreamName, RegSeqRat, RealUp, transport, continuation, provenance, and local naming rows of G together with the displayed rows of W.*

It exposes no selected limit, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, open-cover compactness, hidden selector enlargement, or choice-like tail selector.

Proof. Start from the grid non-escape theorem, Definition 905.5. It says that an accepted grid exports only $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ and none of the excluded completion or host-level objects.

Use grid route compatibility to attach that non-escape result to the bounded monotone witness. The compatibility theorem forces the grid selector, ledger, classifier, and tail rows to be read before the witness route reaches its $\mathsf{Real}^{\uparrow}$ seal row.

Now apply root seal route exhaustion and Real-seal threshold exhaustion for W. These theorems leave the bounded-monotone route inside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and exclude selected limits, quotient equality, host equality, trichotomy, compactness, ambient completeness, and choice-like selectors from the witness side.

The two packets therefore compose without adding an escape coordinate. The grid contributes only its displayed ledger rows, and the bounded monotone witness contributes only its displayed root rows. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid ledger nonescape`

Theorem 535.122 (Bounded monotone Cauchy witness budget-cofinal readback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let its root route consume an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G).$$

Every terminal $\mathsf{Real}^{\uparrow}$ *readback from the composed route must pass through both the scheduled tail budget rows σ, μ, λ of W and the selector-budget rows B, L, K, T of G before the Real seal row Q is public. The readback is exhausted by*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N, \quad B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G,$$

and exports no selected limit, source-equivalence parameter, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, compactness, hidden selector state, or choice-like tail selection.

Proof. First extract the bounded-monotone root. Apply Definition 535.111; every terminal readback from W starts inside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next force the scheduled tail budget. By Definition 535.115, the rows σ, μ, λ are consumed before any located-trap, pre-seal, or terminal $\mathsf{Real}^{\uparrow}$ read can inspect Q.

Attach the grid packet through the public route. Definition 535.119 places the selector budget B, witness ledger L, completion-seal classifier K, and tail envelope T before the route reaches the Real seal row of W.

Now use grid ledger non-escape. Definition 535.121 says that the grid contributes only $B, L, K, T, S_G, R_G, E_G, H$ together with the displayed rows of W.

Finally apply Definition 535.117. The Real seal row Q is public only after the scheduled tail budget, located trap, and grid selector-budget rows have been traversed, so the terminal readback has no coordinate for source-equivalence data, selected limits, quotient equality, host equality, completeness, compactness, or choice-like tail selection. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier budget cofinal readback`

Theorem 535.123 (Bounded monotone Cauchy witness grid synchronizer terminal envelope)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier whose root route consumes an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$. Every root-facing* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *terminal read from the composed route factors through the finite envelope*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N, \quad B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G.$$

The envelope reaches the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling packets only through the displayed budget, ledger, classifier, seal, transport, continuation, provenance, and local naming rows.*

Proof. First attach the synchronizer-facing route of W. Root seal route exhaustion and Real-seal threshold exhaustion keep every terminal read inside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ before the $\mathsf{Real}^{\uparrow}$ threshold is available.

Next attach the grid route compatibility. The grid selector budget, witness ledger, completion-seal classifier, and tail envelope are consumed before the root route reaches the Real seal row Q.

Now apply grid ledger non-escape. It identifies $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ as the only rows contributed by the grid handoff, with no selected limit, host equality, quotient equality, trichotomy, compactness, completeness, or hidden selector state.

The finite observation-budget selector handoff is therefore read as part of the same displayed budget ledger. Since neither W nor G has an additional terminal coordinate, every synchronizer, observation-budget, or Real consumer factors through the stated envelope. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid synchronizer terminal envelope`

Theorem 535.124 (Bounded monotone Cauchy witness tail budget seal correspondence)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let its root route consume an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ The scheduled tail budget σ, μ, λ, the grid selector-budget rows B, L, K, T, and the Real-seal row Q*

correspond on one finite route: a terminal Real$^{\uparrow}$ *read may consume Q only after the tail-window rows of W and the selector-budget rows of G have both been traversed.*

Thus the composed route is a correspondence between tail budget and seal threshold, not a route through a selected limit, quotient real equality, host equality, countable choice, trichotomy, ambient completeness, compactness, or hidden selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier tail budget seal correspondence`

Proof. Begin with the tail-window side. Root tail-window totality forces σ, μ, λ to occur after the source rows and before any located-trap or seal-facing read from W.

Place the seal threshold next. Real-seal threshold exhaustion says that Q is reached only after the displayed tail window and located trap have been consumed, while root seal route exhaustion leaves no bypass around $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Attach the grid budget by route compatibility. The selector budget B, witness ledger L, completion-seal classifier K, and tail envelope T of G are read before the route exposes Q.

Finally use grid ledger non-escape and the finite observation-budget selector handoff. The selector side contributes only its displayed budget rows, so the same finite route accounts for the tail-window obligation and the terminal seal threshold without importing any excluded completion or choice principle. □

Theorem 535.125 (Bounded monotone Cauchy witness cofinal grid readback exhaustion)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *carrier whose root route consumes accepted sibling packets*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

from RealCauchyWitnessGrid$^{\uparrow}$ *together with the displayed* FiniteObservationBudgetSelector$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows. Every cofinal grid readback from this route is exhausted by the rows of W and G listed above. In particular, the readback cannot expose a selected cofinal subsequence, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, compactness, hidden grid state, or unlisted observation selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier cofinal grid readback exhaustion`

Proof. Project the bounded monotone witness first. Root seal route exhaustion and Real-seal threshold exhaustion constrain every terminal readback to the displayed packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Read the grid packet through route compatibility. The grid budget, ledger, classifier, tail envelope, StreamName, RegSeqRat, RealUp, transport, continuation, provenance, and naming rows are consumed before the terminal Real read.

Apply grid ledger non-escape to the composed route. The grid contributes exactly $B, L, K, T, S_G, R_G, E_G, H_G, C$ and the bounded monotone witness contributes exactly its displayed root rows.

The finite observation-budget selector handoff supplies only the selected budget row already named in B, L, K, T. Therefore a cofinal readback has no extra carrier coordinate from which to recover an unlisted selector, cofinal subsequence, equality principle, completeness principle, compactness row, or choice-like tail object. □

Theorem 535.126 (Bounded monotone Cauchy witness diagonal-limit compatibility pullback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet.*

Suppose its root route consumes an accepted RealCauchyWitnessGrid$^{\uparrow}$ *packet* $G = (B, L_G, K_G, T_G, S_G, R_G, E_G, H_G, C$ *and an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$D = (R_D, T_D, S_D, D_D, W_D, Q_D, L_D, H_D, C_D, P_D, N_D)$$

through the budget-seal factorization route of Definition 646.84. Then every terminal Real$^{\uparrow}$ *or completion-seal read from the root route factors through one finite pullback surface consisting of the bounded source* S*, scheduled* StreamName$^{\uparrow}$ *window* σ*, dyadic ledger* λ*,* RegSeqRat$^{\uparrow}$ *handoff* F*, located trap* I*, grid selector and witness ledger rows* $B, L_G, K_G, T_G, S_G, R_G, E_G$*, diagonal budget rows* $R_D, T_D, S_D, D_D, W_D, Q_D, L_D$*, the* CauchyLimitSeal$^{\uparrow}$ *boundary, the terminal* Real$^{\uparrow}$ *seal rows* Q, E_G, L_D*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows*

$$H, H_G, H_D, \quad C, C_G, C_D, \quad P, P_G, P_D, \quad N, N_G, N_D.$$

The pullback exports no selected limit, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, open-cover compactness, hidden selector enlargement, or unlisted tail selector.

Proof. First project the bounded monotone root packet. By Definition 535.111, every root-facing read of W is confined to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, with no external completion coordinate.

Next attach the grid route. The compatibility theorem Definition 535.119 places the selector budget, witness ledger, classifier, and tail rows of G before the Real$^{\uparrow}$ seal row of W, and Definition 535.121 says that the grid contributes only the displayed rows $B, L_G, K_G, T_G, S_G, R_G, E_G, H_G, C_G, P_G, N_G$.

Now read the diagonal-limit compatibility packet. The budget-seal factorization of Definition 646.84 sends terminal Real and completion-seal consumers through $R_D, T_D, S_D, D_D, W_D, Q_D, L_D, H_D, C_D, P_D,$ while Definition 646.85 keeps the same route inside its displayed provenance and local naming rows.

Finally apply terminal pullback non-escape for the diagonal route, Definition 646.136. The bounded source, grid ledger, diagonal budget route, CauchyLimitSeal boundary, RealUp seal, and all transport, continuation, package, and naming rows have already been listed. Since the exhausted finite packet has no coordinate for selected limits, quotient equality, host equality, trichotomy, choice, completeness, compactness, hidden selector growth, or tail-selector enlargement, every terminal read factors through exactly the displayed pullback surface. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier diagonal limit compatibility pullback`

Theorem 535.127 (Bounded monotone Cauchy witness grid root package)**.** *An accepted* BoundedMonotoneCauchy *carrier whose root route consumes an accepted* RealCauchyWitnessGrid$^{\uparrow}$ *packet forms a finite L10 root package through chapter 905, chapter 126, chapter 291, chapter 301, chapter 386, and chapter 38.*

The package reads the grid selector budget, witness ledger, completion-seal classifier, and tail-witness envelope before the bounded-monotone root route reaches the StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*, and* Real$^{\uparrow}$ *sibling rows. It remains a finite root package and not an import of completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid root package`

Proof. First project the bounded monotone root packet by root extraction. The visible route is $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so the package has a finite witness side.

Next apply the limit-seal handoff, monotone-bound route, and root seal route exhaustion. These results identify the sibling chain through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the only seal-facing continuation of the bounded-monotone packet.

Attach the Real Cauchy witness grid by Definition 535.119. The grid rows are consumed before the Real seal row, with selector, ledger, classifier, and tail-envelope reads supplied by the carrier of chapter 905.

Finally apply Definition 535.121. The composed L10 package has no coordinate for selected limits, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, compactness, hidden selector state, or choice-like tail selection. □

Theorem 535.128 (Bounded monotone Cauchy witness root source-seal totality)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing consumer from the bounded monotone source to a* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The route consumes the $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *sibling rows, and it exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector.*

Proof. Start with root extraction. Definition 535.111 confines every root read of W to the displayed carrier rows, so no root-facing consumer can begin from an external completion object.

Next attach the bounded monotone source and the seal handoff. The monotone-bound route places the source data before the scheduled tail, while the limit-seal handoff sends $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers through the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The tail-window and trap theorems close the middle of the route. Root tail-window totality forces consumption of σ, μ, λ before any seal-facing row is reached, and located-trap stability keeps I as the pre-seal enclosure rather than replacing it by an ambient interval choice.

Finally apply Real-seal threshold exhaustion. The threshold theorem consumes the seal row Q only after the displayed source, tail, and trap rows, and root seal route exhaustion leaves no coordinate outside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Therefore the full source-to-seal route is total on the finite packet and exports none of the excluded completion principles. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root source seal totality`

Theorem 535.129 (Bounded monotone Cauchy witness root synchronizer handoff)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$*,* $\mathsf{RealObservationBud}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of* W *factors through the same finite route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The synchronizer handoff uses the sibling L10 chain

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

after the displayed bounded monotone source, schedule, tail witness, dyadic ledger, located trap, seal row, transport, continuation, provenance, and local naming rows have been consumed.

Consequently the root packet supplies the shared synchronizer-facing surface for the limit-seal and Real-observation-budget routes. It supplies no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, hidden synchronizer state, or choice-like selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root synchronizer handoff`

Proof. Begin with root extraction and source-seal totality. Definition 535.111 confines every root read of W to the displayed packet, while Definition 535.128 carries that packet from the bounded monotone source through the $\mathsf{Real}^{\uparrow}$ seal.

Next attach the limit-seal and synchronizer-facing legs. The limit-seal handoff theorem places $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers on the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, and root seal route exhaustion identifies the sibling L10 chain ending at $\mathsf{Real}^{\uparrow}$ as the only seal-facing continuation.

The middle rows are forced by the tail and trap obligations. Root tail-window totality consumes σ, μ, λ before any seal-facing read, and located-trap stability keeps I as the pre-seal enclosure rather than replacing it by an ambient interval choice.

Finally use Real-seal threshold exhaustion. The $\mathsf{Real}^{\uparrow}$ threshold is reached only after $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ have been traversed, so a $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ or $\mathsf{RealObservationBudget}^{\uparrow}$ consumer has no hidden synchronizer coordinate beyond the displayed route. The excluded completion and choice principles are absent from the packet and cannot be exported by the handoff. □

Theorem 535.130 (Bounded monotone Cauchy witness synchronizer-grid route)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$ be an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet consumed on the root route of W. Every root-facing synchronizer consumer of W that also consumes G must route through the same finite witness rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and through the grid rows $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ before reaching any $\mathsf{Real}^{\uparrow}$ *seal.*

The route exports no selected limit, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, open-cover compactness, hidden synchronizer state, or detached grid selector.

Proof. Begin with the root synchronizer handoff. It sends every $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer through the displayed rows of W and leaves no hidden synchronizer coordinate outside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Attach the grid by grid route compatibility. Any accepted route that consumes G reaches the grid selector budget, witness ledger, completion classifier, and tail envelope before the $\mathsf{Real}^{\uparrow}$ seal row Q of W is read.

Use grid ledger non-escape to identify the grid contribution. The grid packet contributes only $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ together with the displayed rows of W, and it does not supply a detached selector or ambient completion principle.

Finally apply root Real-seal threshold exhaustion. The $\mathsf{Real}^{\uparrow}$ seal can be consumed only after the finite bounded-source, tail-window, located-trap, synchronizer, and grid rows have already been traversed, so none of the excluded selected-limit, equality, choice, completeness, compactness, hidden-state, or detached-selector data is exported. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier synchronizer grid route`

Theorem 535.131 (Bounded monotone Cauchy witness root tail-seal nesting)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of W is forced through the nested finite order*

$$S, F \longrightarrow \sigma, \mu, \lambda \longrightarrow I \longrightarrow Q \longrightarrow H, C, P, N.$$

The S, F stage supplies the bounded monotone source and dyadic bound data; the σ, μ, λ stage supplies the scheduled tail-window data; the I stage supplies the located trap; the Q stage supplies the Real-seal threshold; and H, C, P, N carry transport, route replay, provenance, and local naming.

Consequently the L10 source-to-seal route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *remains inside the displayed packet. It exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, hidden synchronizer state, or choice-like selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessTailSealNesting`

Proof. First confine the packet. Root extraction, Definition 535.111, says that every root-facing read of W is a read of the displayed carrier rows $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next force the tail-window stage. By Definition 535.115, any root-facing pre-seal or seal consumer must pass through σ, μ, λ after the source and bound rows S, F.

Now place the trap and threshold rows. Located-trap stability, Definition 535.116, keeps I as the pre-seal enclosure, and Real-seal threshold exhaustion, Definition 535.117, places Q before the $\mathsf{Real}^{\uparrow}$ threshold is consumed.

Finally apply the synchronizer handoff, Definition 535.129. It sends $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ and $\mathsf{RealObservationBudget}^{\uparrow}$ consumers through the same finite surface as the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, with H, C, P, N carrying transport, route replay, provenance, and naming rather than an external synchronizer coordinate. □

Theorem 535.132 (Bounded monotone Cauchy witness grid budget precedes seal)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let its root route consume an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$. Along that finite route the selector budget B, witness ledger L, completion-seal classifier K, and tail envelope T of G must be read before the route exposes the* $\mathsf{Real}^{\uparrow}$ *seal row Q of W.*

Thus the grid budget is a pre-seal obligation on the same root route. The composed packet exports no selected limit, quotient real equality, host equality, trichotomy, countable choice, ambient completeness, hidden grid selector, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid budget precedes seal`

Proof. First attach the grid route to the bounded-monotone root packet. By Definition 535.119, any route consuming G reaches the grid selector budget, witness ledger, completion classifier, and tail envelope before it reaches the $\mathsf{Real}^{\uparrow}$ seal row Q of W.

Next use the ledger non-escape theorem. Definition 535.121 says that the grid handoff contributes only $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ together with the displayed rows of W, and does not add selected limits, quotient equality, host equality, trichotomy, countable choice, ambient completeness, or hidden selector state.

Now place this order at the Real threshold. Definition 535.117 forces any $\mathsf{Real}^{\uparrow}$ threshold read from W to occur only after the finite source, tail-window, and located-trap rows have already been consumed.

Finally apply tail-seal nesting, Definition 535.131. It nests the route as a single finite chain ending in Q and then H, C, P, N, so the grid budget rows cannot be exposed on a separate completion route after the seal. The same route therefore reads B, L, K, T before exposing Q, and none of the excluded completion principles is exported. □

Theorem 535.133 (Bounded monotone Cauchy witness route tightening)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and let its root route consume an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$. The accepted witness route factors through the monotone-bound rows, the tail-window rows, the grid-budget rows, and the* $\mathsf{Real}^{\uparrow}$ *seal row in the finite order*

$$S, F \longrightarrow \sigma, \mu, \lambda \longrightarrow B, L, K, T \longrightarrow I \longrightarrow Q \longrightarrow H, C, P, N.$$

The route is therefore a tightening of the displayed BHist *packet: every Real-seal read is preceded by bounded-source data, scheduled Cauchy-modulus data, selector-budget and witness-ledger data, and the located trap. It does not import a selected limit, host real equality, quotient real equality, completeness, compactness, trichotomy, or choice data.*

Proof. First apply root extraction and the monotone-bound route. Theorems 535.111 and 535.113 confine the accepted carrier to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ and place the bounded-source rows before the scheduled tail service.

Next use root tail-window totality. It makes σ, μ, λ unavoidable before any located-trap or Real-seal row can be consumed, so the Cauchy-modulus route is already finite inside the displayed packet.

Now attach the grid side by grid budget precedence. The theorem Definition 535.132 places B, L, K, T before Q on the same root route, not on a parallel completion route.

Finally use the limit-seal handoff and synchronizer handoff. These hand the resulting finite route to $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers through H, C, P, N, leaving no carrier coordinate for the excluded completion or choice principles. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier route tightening`

Theorem 535.134 (Bounded monotone Cauchy witness schedule factorization)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Before any downstream* $\mathsf{Real}^{\uparrow}$ *seal read from W, the schedule row σ, the Cauchy-modulus witness row μ, and the dyadic tolerance ledger λ appear as finite* BHist *rows of the same accepted packet, transported only by H, replayed only by C, packaged only by P, and named only by N.*

Consequently the Real-seal consumer receives finite schedule and modulus data rather than a selected tail, host sequence equality, ambient Cauchy-completeness principle, or choice-like modulus selector.

Proof. Start from root extraction. It identifies σ, μ, λ as visible carrier rows and leaves no separate schedule service outside W.

Use root tail-window totality to force the order. Every convergence-seal, limit-seal, synchronizer, observation-budget, or Real-seal consumer must consume σ, μ, λ before the located-trap and seal rows.

Apply route tightening to the resulting packet. The tightened route keeps the schedule, modulus, dyadic ledger, grid-budget rows, located trap, and seal row on one finite BHist route.

The limit-seal handoff and synchronizer handoff then move that finite route through H, C, P, N. Since those are the only transport, continuation, package, and naming rows, no selected tail, host sequence equality, completeness principle, or choice-like modulus selector is available to the Real-seal consumer. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier schedule factorization`

Theorem 535.135 (Bounded monotone Cauchy witness Real-seal consumer exhaustion)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*-seal consumer of an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, together with any accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet consumed on its root route, is exhausted by the displayed rows*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and by the grid-budget rows $B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G$ *when such a grid packet is present. No downstream Real-seal consumer can import selected limits, host real equality, quotient real equality, real completeness, compactness, trichotomy, countable choice, hidden grid selector state, or choice-like tail data.*

Proof. First confine the witness route by root extraction and limit-seal handoff. These theorems identify the finite rows of W and make every $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read pass through them.

Next apply schedule factorization. It records that σ, μ, λ are finite BHist rows before the Real-seal read, with H, C, P, N carrying only displayed transport, route replay, provenance, and naming.

If the root route consumes a grid packet, use grid budget precedence. The rows B, L, K, T are read before Q, and the remaining grid rows are exactly the carrier, handoff, transport, continuation, provenance, and naming rows already listed by the grid package.

Finally apply the synchronizer handoff and route tightening. They exhaust the Real-seal, synchronizer, and observation-budget consumers by the displayed witness and grid rows, so none of the excluded selected-limit, host-equality, completeness, compactness, trichotomy, choice, or hidden-selector data can be imported. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal consumer exhaustion`

Theorem 535.136 (Bounded monotone Cauchy witness four-face budget correspondence)**.** *Every accepted bounded monotone Cauchy witness consumed by the Real-seal route factors through exactly the four licensed L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

For an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *face is the finite observation schedule* σ*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the dyadic tolerance and upper-bound ledger* λ *together with the grid selector budget when a* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet is consumed, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the regular rational readback row* F *together with the grid readback rows, and the* $\mathsf{Real}^{\uparrow}$ *face is the seal row* Q*.*

The correspondence is exhausted by displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows. It has no coordinate for quotient streams, selected host limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First confine the consumer to the witness packet. By Definition 535.111 and Definition 535.113, every root-facing read stays inside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, with the bounded source and regular rational readback before the scheduled tail service.

Next identify the licensed faces. Theorems 535.115 and 535.134 make σ, μ, λ unavoidable before a seal read and read those rows as finite BHist data transported only by H, replayed only by C, packaged only by P, and named only by N.

Now attach the budget side. The grid-budget theorem Definition 535.132 places the selector budget, witness ledger, completion-seal classifier, and tail envelope before Q whenever a grid packet is present, and Definition 535.133 keeps those rows on the same Real-facing chain rather than on a parallel completion route.

Finally apply Definition 535.129 and Definition 535.135. These keep observation-budget and seal-budget consumers on the same finite surface and list all rows available to the downstream consumer. Therefore the four faces above are the entire budget correspondence, and the excluded quotient-stream, selected-limit, choice, and ambient-completeness data have no displayed carrier row from which to be read. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier four face budget correspondence`

Theorem 535.137 (Bounded monotone Cauchy witness uniform criterion handoff)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier, and suppose a downstream* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer reads the root route of W through an accepted packet*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U).$$

Then the bounded-source route of W factors through the finite chain

$$S, F \longrightarrow \sigma, \mu, \lambda \longrightarrow I, Q \longrightarrow I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U.$$

The first stage supplies the displayed bounded monotone source and regular rational readback; the second supplies the $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, Cauchy tail witness, and dyadic tolerance ledger; the third supplies the located trap and* $\mathsf{Real}^{\uparrow}$ *seal row; and the final stage is exactly the UniformCauchyCriterion family-test surface over its finite index, shared window, shared modulus, dyadic tolerance, tail ledger, seal handoff, transport, continuation, package, and local naming rows.*

The StreamName finite-window handoff and the $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal rows remain explicit throughout the chain. Thus the handoff exports no quotient streams, selected host limits, host real equality, countable choice, ambient completeness, open-cover compactness, hidden family limit, or direct* $\mathsf{Real}^{\uparrow}$ *bypass.*

Proof. First project the bounded monotone witness. Root extraction confines every root read of W to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, and root tail-window totality forces the consumer to pass through σ, μ, λ before any located-trap or seal row is available.

Next factor the schedule route. The schedule factorization theorem records σ, μ, λ as finite BHist rows transported only by H, replayed only by C, packaged only by P, and named only by N. The four-face budget correspondence then identifies the licensed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces that the downstream consumer may read.

Now attach the UniformCauchyCriterion packet. By Definition 435.3, the family request preserves the same finite index ledger, shared window family, and shared modulus before a tail comparison is inspected. By Definition 435.115, the full criterion-facing surface is exhausted by $I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U$.

Finally bind the StreamName and regular-real handoff rows. The StreamName finite-window handoff of Definition 126.79 keeps the finite observation window and $\mathsf{RegSeqRat}^{\uparrow}$ seed rows visible, while the bounded monotone root seal route and Real-seal consumer exhaustion keep the I, Q suffix tied to the same regular-source and seal route. Since neither side contributes quotient streams, selected limits, host real equality, countable choice, ambient completeness, open-cover compactness, or a direct $\mathsf{Real}^{\uparrow}$ bypass, their composite handoff is exactly the finite chain displayed above. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier uniform criterion handoff`

Theorem 535.138 (Bounded monotone Cauchy witness observation-budget grid pullback)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let*

$$O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet reading the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows used by the root route of* W*, the grid packet* G*, and the uniform criterion packet* U*. Then the bounded-source route, grid-budget route, uniform-criterion route, and observation-budget route determine one pullback surface before any* $\mathsf{Real}^{\uparrow}$ *seal is public:*

$$S, F, \sigma, \mu, \lambda, I, Q, B, L, K, T, S_G, R_G, E_G, I_U, W_U, M_U, Q_U, T_U, E_U, E_O, W_O, D_O, R_O, S_O, H, H_G, H_U, H_O, C, C_G, C_U$$

Every consumer of this surface first reads the root source and tail-window rows of W*, then the grid budget rows of* G*, then the uniform-criterion rows of* U*, and finally the finite observation-budget rows that name the same windows, dyadic ledgers, regular readbacks, and seal boundary. The surface exports no quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness theorem, open-cover compactness principle, hidden family limit, or direct* $\mathsf{Real}^{\uparrow}$ *bypass.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier observation budget grid pullback`

Proof. First project the bounded-source route. Root extraction confines all reads of W to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P,$ and schedule factorization records σ, μ, λ as finite BHist rows transported only by H, replayed only by C, packaged only by P, and named only by N.

Next attach the grid side. The grid-budget precedence theorem places B, L, K, T before the seal row Q on the same root route, while the grid readback rows S_G, R_G, E_G sit on the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces. Thus the grid packet contributes budget rows to the same surface rather than a second completion route.

Now attach the uniform criterion. The uniform-criterion handoff sends the family-test packet U through the already displayed finite windows, dyadic tolerance ledger, regular readback, and

seal handoff. Its rows $I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U$ therefore meet the bounded-source and grid rows along the shared L10 faces.

Finally read the observation-budget route. By Definition 611.1, Definition 611.3, Definition 611.4, Definition 611.5, and Definition 611.6, the observation-budget consumer exposes only its finite request, window, dyadic, regular-handoff, seal, transport, continuation, provenance, and naming rows. Since those rows are assumed to be the same finite windows, ledgers, readbacks, and seal rows already displayed by W, G, and U, the four routes share exactly the pullback surface above before any $\mathsf{Real}^{\uparrow}$ seal can be public. □

Theorem 535.139 (Bounded monotone Cauchy witness criterion-grid-synchronizer correspondence)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Its public route to* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, *and* $\mathsf{RealObservationBudget}^{\uparrow}$ *is a single displayed correspondence before Q is read as the* $\mathsf{Real}^{\uparrow}$ *seal: the criterion tail exposes the scheduled window and modulus ledger, the grid packet exposes the finite observation and witness-budget rows, and the synchronizer exposes the seal budget.*

The correspondence is exhausted by the rows of W, the grid-budget rows, the criterion-tail rows, the synchronizer seal-budget rows, the observation-budget rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *names. It exports no quotient streams, selected limits, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection.*

Proof. Apply root extraction to the accepted packet W. This confines the public route to the bounded source, regular readback, schedule, modulus, dyadic ledger, located trap, seal, transport, route, provenance, and naming rows.

Apply root tail-window totality. The scheduled window σ, modulus witness μ, and dyadic ledger λ are read before any located-trap, synchronizer, observation-budget, or $\mathsf{Real}^{\uparrow}$ seal consumer can inspect Q.

Apply grid budget precedence. The $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ packet contributes its finite observation and witness-budget rows before the seal row, so the grid route is a budgeted face of the same root route rather than a hidden completion route.

Apply the synchronizer handoff and the observation-budget grid pullback. The synchronizer exposes the seal budget on the same displayed chain, and the observation-budget route reads the same finite windows, dyadic ledgers, regular readbacks, and seal boundary.

Finally apply Real-seal consumer exhaustion. The downstream consumer has only the displayed witness, criterion, grid, synchronizer, observation-budget, transport, continuation, provenance, and naming rows, so the excluded quotient-stream, selected-limit, host-equality, trichotomy, completeness, compactness, and choice-like tail-selection routes have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier criterion grid synchronizer correspondence`

Theorem 535.140 (Bounded monotone Cauchy witness root Real-seal determinacy)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let $G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$ be an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let $U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$ be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let $O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$ be an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet reading the same displayed finite-window, dyadic-ledger, regular-readback, and seal rows as the root route of W and the grid and criterion packets.*

Then any public $\mathsf{Real}^{\uparrow}$ *seal read on the shared surface is determined by the common displayed rows of the bounded-source route, grid-budget route, uniform-criterion route, synchronizer route, and observation-budget route. If two public seal reads preserve*

$$S, F, \sigma, \mu, \lambda, I, Q, B, L, K, T, S_G, R_G, E_G, I_U, W_U, M_U, Q_U, T_U, E_U, E_O, W_O, D_O, R_O, S_O, H, H_G, H_U, H_O, C, C_G, C_U$$

then they expose the same $\mathsf{Real}^{\uparrow}$ *seal boundary. The shared surface carries no quotient stream, selected host limit, host equality, choice principle, ambient completeness theorem, open-cover compactness principle, hidden family limit, or direct* $\mathsf{Real}^{\uparrow}$ *bypass.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root real seal determinacy`

Proof. Begin with root extraction and tail-window totality. Root extraction confines every public read of W to $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, and tail-window totality forces σ, μ, λ to be consumed before any located-trap, synchronizer, observation-budget, or $\mathsf{Real}^{\uparrow}$ seal read can inspect Q.

Next apply grid budget precedence. The grid rows B, L, K, T occur before the seal row on the same root route, while S_G, R_G, E_G are the displayed StreamName, RegSeqRat, and Real-seal faces of that route. Thus the grid packet cannot alter the seal boundary without altering a displayed row.

Now use the uniform criterion handoff. The packet U reads the same finite windows, shared modulus, dyadic tolerance, tail ledger, and seal handoff already exposed by the bounded-source and grid routes, so a public criterion read preserving those rows preserves the same seal-facing boundary.

Apply the observation-budget grid pullback and the criterion-grid-synchronizer correspondence. These identify the observation request, window, dyadic, regular-handoff, seal, synchronizer, transport, continuation, provenance, and naming rows as one shared pullback surface before the $\mathsf{Real}^{\uparrow}$ seal is public.

Finally use Real-seal consumer exhaustion. A downstream seal consumer has only the displayed witness, grid, criterion, synchronizer, observation-budget, transport, continuation, provenance, and naming rows listed above. Therefore two reads that preserve those rows determine the same public $\mathsf{Real}^{\uparrow}$ seal boundary, and the excluded quotient-stream, selected-limit, host-equality, choice, completeness, compactness, and bypass routes have no carrier coordinate. □

Theorem 535.141 (Bounded monotone Cauchy witness terminal budget extraction)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let* D *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reading the same root windows, regular-rational readbacks, uniform criterion rows, and Real-seal boundary.*

If the route from W *has passed root extraction, limit-seal handoff, monotone-bound routing, grid-route compatibility, uniform-criterion handoff, and root Real-seal determinacy, then it exposes one terminal budget row consumable by* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, and* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$*. The row is exhausted by*

$$S, F, \sigma, \mu, \lambda, I, Q, B, L, K, T, S_G, R_G, E_G, I_U, W_U, M_U, Q_U, T_U, E_U, H, H_G, H_U, C, C_G, C_U, P, P_G, P_U, N, N_G, N_U$$

together with the displayed diagonal-limit compatibility rows of D. It exports no selected limit, quotient stream, host real equality, countable-choice selector, ambient completeness theorem, open-cover compactness principle, hidden family limit, or direct $\mathsf{Real}^{\uparrow}$ *bypass.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier terminal budget extraction`

Proof. First extract the root route. Apply Definition 535.111, Definition 535.112, and Definition 535.113. These place the bounded source, regular readback, scheduled window, modulus ledger, dyadic ledger, located trap, Real seal, transport, continuation, provenance, and local name on the finite packet W.

Next attach the grid route. Use Definition 535.119, Definition 535.132, and Definition 535.121. The grid contributes selector-budget, witness-ledger, classifier, tail, regular-readback, and seal-facing rows before the $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Now attach the uniform criterion. Apply Definition 535.137; it sends U through the same finite index, window, modulus, dyadic tolerance, tail ledger, exactness, transport, continuation, provenance, and local naming surface. The criterion row is therefore part of the same terminal budget rather than a separate family-limit principle.

Finally use Definition 535.140 together with the diagonal-limit compatibility package of chapter 646. The Real seal is determined by the common displayed rows, and the diagonal-limit consumer reads only its displayed compatibility rows. Thus $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, and $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ consume the same terminal budget row, while the excluded completion, equality, choice, compactness, hidden-limit, and bypass data have no carrier coordinate. □

Theorem 535.142 (Bounded monotone Cauchy witness grid selector exactness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, let Y be an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, and let*

$$O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose these packets read the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal row displayed by the root route of W.*

Then the public selector route from W, G, U, Y, O is exact. Any two selector-facing reads that preserve the bounded-source rows $S, F, \sigma, \mu, \lambda, I, Q$, the grid-budget rows $B, L, K, T, S_G, R_G, E_G$, the uniform-criterion rows $I_U, W_U, M_U, Q_U, T_U, E_U$, the synchronizer seal-budget rows of Y, the observation-budget rows E_O, W_O, D_O, R_O, S_O, the componentwise transport rows H, H_G, H_U, H_O, the displayed continuation rows C, C_G, C_U, C_O, the provenance rows P, P_G, P_U, P_O, and the local NameCert rows N, N_G, N_U, N_O expose the same selector-facing $\mathsf{Real}^{\uparrow}$ *seal boundary.*

The exact selector surface has no coordinate for quotient streams, selected host limits, host real equality, choice selectors, ambient completeness, open-cover compactness, hidden family limits, or a direct $\mathsf{Real}^{\uparrow}$ *bypass.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier grid selector exactness`

Proof. First extract the root route. Root extraction and root tail-window totality force every selector-facing read of W through the displayed bounded source, regular readback, finite window, modulus, dyadic ledger, located trap, seal row, transport, continuation, provenance, and local naming rows.

Next attach the grid route. Grid budget precedence and grid route compatibility keep G on the same finite-window, dyadic-ledger, regular-readback, and seal boundary as W. The grid rows therefore refine the selector budget on the displayed route rather than adding a second completion stream.

Now add the criterion, synchronizer, and observation faces. The observation-budget grid pullback identifies the observation rows with the same window, ledger, readback, and seal surface, while the criterion-grid-synchronizer correspondence places the uniform criterion and synchronizer rows on that shared surface before the seal is public.

Finally apply root Real-seal determinacy. Once the bounded-source, grid-budget, uniform-criterion, synchronizer, observation-budget, transport, continuation, provenance, and NameCert rows are preserved, any two public selector reads expose the same $\mathsf{Real}^{\uparrow}$ seal boundary. Since the preceding route inventory contains no quotient, selected-limit, host-equality, choice, completeness, compactness, hidden-family, or bypass coordinate, no such data can enter the selector. □

Theorem 535.143 (Bounded monotone Cauchy witness terminal grid non-escape)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, let* Y *be an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, and let*

$$O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose the route also names the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal row.*

Then every terminal grid read from W, G, U, Y, O *remains inside the displayed bounded-source, grid-budget, uniform-criterion, synchronizer, observation-budget, stream-window, dyadic-ledger, regular-readback, Real-seal, transport, continuation, provenance, and NameCert rows. It exports no quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness theorem, open-cover compactness principle, hidden family limit, or direct* $\mathsf{Real}^{\uparrow}$ *bypass.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier terminal grid nonescape`

Proof. First extract the root route. Apply Definition 535.111 and Definition 535.115. These force every terminal read through $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ before the Real-facing seal is available.

Next attach the grid route. Use Definition 535.119 and Definition 535.121. The grid rows are therefore consumed as selector budget, witness ledger, classifier, tail envelope, regular readback, and seal-facing data on the same route.

Now add the synchronizer and observation-budget faces. The observation-budget grid pullback and the criterion-grid-synchronizer correspondence place U, Y, O on the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface.

Finally apply Definition 535.140 and Definition 535.142. The terminal grid read is determined by the displayed route, so the excluded completion, equality, choice, compactness, hidden-limit, and bypass data have no coordinate from which to escape. □

Theorem 535.144 (Bounded monotone Cauchy witness root budget consumer exactness)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, let* Y *be an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, and let*

$$O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose they read the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows displayed by the root route.*

Then any two root-budget consumers preserving $S, F, \sigma, \mu, \lambda, I, Q$, *the rows of* G, U, Y, O, *the shared* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff rows, componentwise transport, displayed continuation, provenance, and local NameCert data expose the same root budget boundary. The consumer has no coordinate for a second budget, a selected limit, quotient-stream equality, host real equality, countable choice, ambient completeness, compactness, hidden family data, or a direct* $\mathsf{Real}^{\uparrow}$ *route.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root budget consumer exactness`

Proof. Start with the bounded root. Root extraction and root tail-window totality display the bounded source, regular readback, scheduled window, modulus witness, dyadic ledger, located trap, Real-seal row, transport, continuation, provenance, and local name before a budget consumer can inspect the seal.

Next consume the grid and criterion route. Grid route compatibility and uniform-criterion handoff identify the grid rows and criterion rows with the same finite windows, dyadic ledgers, regular readbacks, and seal boundary.

Then use the budget services. The synchronizer route and observation-budget grid pullback place Y and O on the same root budget rather than on a detached observation schedule. The route therefore has one displayed consumer boundary for W, G, U, Y, O.

Finally apply root Real-seal threshold exhaustion and grid selector exactness. Once the listed rows are preserved, the public root-budget read is fixed, and the excluded completion, equality, choice, compactness, hidden-limit, and bypass data cannot be projected. □

Theorem 535.145 (Bounded monotone Cauchy witness RealUp threshold lock)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Let*

$$G = (B, L, K, T, S_G, R_G, E_G, H_G, C_G, P_G, N_G)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet, let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, let* Y *be an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, and let*

$$O = (E_O, W_O, D_O, R_O, S_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose the route reads the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal boundary.*

Then the terminal $\mathsf{Real}^{\uparrow}$ *threshold is locked by the displayed route through* W, G, U, Y, O*. Any Real-facing consumer must first consume the root extraction, root tail-window, grid route, grid ledger, synchronizer, observation-budget, and grid-selector rows named above, and no ambient completeness principle, selected host limit, quotient equality, countable-choice selector, open-cover compactness principle, hidden family limit, or direct* $\mathsf{Real}^{\uparrow}$ *bypass can unlock the threshold.*

Proof. First force the root order. Root extraction and root tail-window totality put $S, F, \sigma, \mu, \lambda$ before the located trap and seal rows, while root Real-seal threshold exhaustion places the $\mathsf{Real}^{\uparrow}$ threshold after the displayed root packet.

Next force the grid order. Grid route compatibility and grid ledger non-escape consume G as a finite selector-budget and witness-ledger route before the Real seal is public. The grid cannot introduce a second completion stream.

Now identify the terminal service rows. The criterion handoff, synchronizer route, observation-budget grid pullback, and grid selector exactness place U, Y, O on the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ boundary.

Finally compose the locked route. Since every Real-facing consumer has already traversed the displayed root, grid, criterion, synchronizer, observation, transport, continuation, provenance, and NameCert rows, the threshold has no remaining coordinate through which the excluded completion or choice data could enter. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier realup threshold lock`

Theorem 535.146 (Bounded monotone Cauchy witness root window extraction)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing consumer that reads* W *for a convergence-seal, limit-seal, or real-seal input first extracts the finite window route*

$$S, F, \sigma, \mu, \lambda$$

before it reads the located trap I*, the Real seal row* Q*, or the transport, continuation, provenance, and local naming rows* H, C, P, N*. The extraction is a finite* BHist *route through the bounded source, regular rational handoff, stream schedule, Cauchy witness, and dyadic ledger, not an appeal to selected limits, quotient real equality, host real equality, trichotomy, ambient completeness, or open-cover compactness.*

Proof. Project the accepted carrier through Definition 535.5. Its root-facing rows are exactly $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, with no hidden coordinate for a limiting representative or ambient completion principle.

Use the local NameCert obligations and scoped dependency package, Definition 535.6 and Definition 535.19. They place source reads in S, regular handoff in F, stream windows in σ, Cauchy witnesses in μ, and dyadic ledger data in λ before the located and seal rows are available.

Finally apply root extraction and root seal route exhaustion, Definition 535.111 and Definition 535.114. Those theorems identify every root consumer with the same finite packet, so the window route $S, F, \sigma, \mu, \lambda$ is the first extracted segment of the route. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window extraction`

Theorem 535.147 (Bounded monotone Cauchy witness root window readiness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion-facing seal consumer of W must first read the displayed packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

as its explicit readiness surface. The prefix

$$S, F, \sigma, \mu, \lambda$$

is the readiness prefix, and the suffix

$$I, Q, H, C, P, N$$

is the seal suffix. The surface exports the bounded source row, regular rational handoff, scheduled window, tail modulus, dyadic ledger, located trap, Real seal row, transport, continuation, provenance, and local naming row, and it exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window readiness`

Proof. First read the root window extraction row. Theorem 535.146 forces every root-facing consumer through $S, F, \sigma, \mu, \lambda$ before any located trap, Real seal, transport, continuation, provenance, or naming suffix is available.

Next read the tail-modulus ledger. Theorem 535.148 keeps every tail request on the displayed μ, σ, λ ledger, so the readiness prefix is not merely a carrier projection but the public tail-service surface consumed before a seal route can be read.

Now pull the route back to the public criterion-tail boundary. Theorem 535.149 identifies any such $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or completion-facing consumer with the finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ packet displayed by W.

Finally apply root Real-seal nonescape. Theorem 535.150 identifies the remaining rows as exactly I, Q, H, C, P, N. The accepted carrier has no coordinate for a selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, so the displayed prefix and suffix together form the whole readiness surface consumed by Real-facing and completion-facing seal routes. □

Theorem 535.148 (Bounded monotone Cauchy witness root tail modulus ledger)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing tail request made to W is ledgered by the displayed triple*

$$\mu, \sigma, \lambda$$

before any located-trap row I, Real seal row Q, or completion-facing handoff is read. The modulus answer μ, scheduled window σ, and dyadic ledger λ therefore form the public tail-modulus ledger for the root route, and no consumer may replace that ledger by a selected tail, quotient representative, host equality, trichotomy, ambient completeness, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root tail modulus ledger`

Proof. Start from the carrier and NameCert rows. The carrier makes μ, σ, λ displayed components, while the NameCert obligations keep Cauchy-tail, window, and dyadic-budget reads inside those components.

Apply Definition 535.146. It forces the root consumer through $S, F, \sigma, \mu, \lambda$ before I or Q, so the tail request is already tied to the scheduled window and dyadic ledger before a trap or seal can be consumed.

Now use the monotone-bound route and the limit-seal handoff, Definition 535.113 and Definition 535.112. The bounded monotone source reaches the seal-facing suffix only through the same modulus ledger, and the handoff to $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ preserves that finite route.

The root seal route exhaustion theorem then rules out every alternate tail service. Since the exhausted packet contains no selected tail, quotient representative, host equality, trichotomy, ambient completeness, or compactness row, the displayed triple is the whole public root tail-modulus ledger. □

Theorem 535.149 (Bounded monotone Cauchy witness root criterion-tail pullback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{Real}^{\uparrow}$ *seal or convergence-seal consumer that uses the ledger triple*

$$\mu, \sigma, \lambda$$

must factor through the public criterion-tail route of Definition 535.69 before it reads the located-trap row I or the Real-seal row Q. The pullback of such a consumer is the finite packet already displayed by the carrier: the BHist *rows,* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row.*

Consequently the root route exports no selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. Start with root window extraction and the root tail modulus ledger. Theorems 535.146 and 535.148 place every root-facing read through $S, F, \sigma, \mu, \lambda$ and then through the displayed ledger triple μ, σ, λ before any access to I or Q.

Next apply the criterion-tail factorization of Definition 535.69. That factorization is the public route from the bounded monotone source into the Cauchy criterion tail surface, so a consumer using the root ledger cannot jump directly from μ, σ, λ to a Real-facing seal read.

Finally use the limit-seal handoff and the root Real-seal nonescape boundary. The handoff keeps the route inside $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ as a displayed BEDC packet, while nonescape identifies the remaining suffix as I, Q, H, C, P, N rather than an unledgered completion object.

The resulting pullback is therefore exactly the finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ packet displayed by the accepted carrier. Since this packet has no coordinate for selected limits, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, none of those objects is available to the root-facing consumer. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root criterion tail pullback`

Theorem 535.150 (Bounded monotone Cauchy witness root Real-seal nonescape)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. The rows*

I, Q, H, C, P, N hand off only the displayed packet

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

to $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$. *This handoff exports the located trap, Real seal route, componentwise transport, continuation, package provenance, and local naming rows, but it exports no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or unledgered completion principle.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root real seal nonescape`

Proof. Read the suffix through the accepted carrier. The rows I, Q, H, C, P, N are the located trap, Real seal, transport, continuation, provenance, and naming coordinates of the same finite packet, not a second source of completion data.

Use the root window extraction and tail modulus ledger rows proved above. They force every root-facing request through $S, F, \sigma, \mu, \lambda$ and through the ledger μ, σ, λ before any read of I, Q, H, C, P, N.

Apply the limit-seal handoff and monotone-bound route, Definition 535.112 and Definition 535.113. These identify the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers with the same bounded-source, regular-handoff, scheduled-tail, located-trap, and seal route.

Finally cite root seal route exhaustion. Once the route is exhausted by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, compactness, or unledgered completion principle remains for the Real-seal handoff to export. □

Theorem 535.151 (Bounded monotone Cauchy witness window-meet route)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P,$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing convergence consumer or Real-seal consumer using the displayed window route must pass through the common finite sequence*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The route composes root window extraction, root tail modulus ledgering, criterion-tail pullback, and root Real-seal nonescape. It exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window meet route`

Proof. Begin with root window extraction. Theorem 535.146 forces the root-facing consumer through $S, F, \sigma, \mu, \lambda$ before the located trap or seal suffix can be read.

Next apply the root tail modulus ledger. Theorem 535.148 keeps every tail request ledgered by μ, σ, λ inside the same finite packet.

Use criterion-tail pullback. Theorem 535.149 pulls any Real-facing or convergence-facing consumer back to the public criterion-tail route and its displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally apply root Real-seal nonescape. Theorem 535.150 identifies the remaining suffix as I, Q, H, C, P, N and excludes every selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selector. The carrier of Definition 535.5 has no alternate coordinate, so the displayed sequence is the whole window-meet route. □

Theorem 535.152 (Bounded monotone Cauchy witness root window seal synchronizer)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *carrier. Every root-facing* $\mathsf{Real}^{\uparrow}$ *seal or convergence-seal consumer that uses the window route must synchronize the prefix*

$$S, F, \sigma, \mu, \lambda$$

with the suffix

$$I, Q, H, C, P, N$$

through the same finite BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *route. The synchronized packet is*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

and it exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. Begin with root window extraction and the root tail modulus ledger. Theorems 535.146 and 535.148 place the prefix rows $S, F, \sigma, \mu, \lambda$ and the ledger triple μ, σ, λ before any root-facing read of the located trap, Real seal, transport, continuation, provenance, or naming suffix.

Next use the criterion-tail pullback. Theorem 535.149 identifies every such Real-facing or convergence-facing consumer with the public criterion-tail route and its displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Apply root Real-seal nonescape and the window-meet route. Theorems 535.150 and 535.151 identify the suffix as I, Q, H, C, P, N and show that prefix and suffix are consumed as the single finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

The accepted carrier has no further coordinate for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection. Hence any root-facing seal consumer using the window route must synchronize through the displayed finite BEDC packet. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window seal synchronizer`

Theorem 535.153 (Bounded monotone Cauchy witness Real-seal selector exhaustion)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or completion-facing consumer that asks for a selector must factor through the displayed prefix*

$$S, F, \sigma, \mu, \lambda$$

and the displayed suffix

$$I, Q, H, C, P, N.$$

After those rows are fixed, the accepted packet has no coordinate for a second selector, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection.

Proof. First expose the selector prefix. Root window extraction and the root tail modulus ledger, Definition 535.146 and Definition 535.148, place every root-facing selector request on $S, F, \sigma, \mu, \lambda$ and on the displayed μ, σ, λ tail ledger before any Real seal is read.

Next pull the selector request back to the public tail boundary. The criterion-tail pullback Definition 535.149 identifies the consumer with the finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ route already displayed by the accepted packet.

Then read the seal suffix. Root Real-seal nonescape and the window-meet route, Definition 535.150 and Definition 535.151, identify the remaining rows as exactly I, Q, H, C, P, N and exclude any unshown completion or choice service.

Finally apply the root window seal synchronizer Definition 535.152. It binds the prefix and suffix into the single packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$, so a consumer that asks for selector data has exhausted every selector-facing coordinate before the Real seal is read. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal selector exhaustion`

Theorem 535.154 (Bounded monotone Cauchy witness window Real-seal pullback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer using the synchronized window factors through the displayed packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The pullback exports the bounded source row, regular rational handoff, scheduled window, tail modulus, dyadic ledger, located trap, Real seal, componentwise transport, continuation, package provenance, and local naming row, and it exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector.

Proof. Begin with root window extraction. Theorem 535.146 forces any root-facing Real or limit-seal consumer to read the prefix $S, F, \sigma, \mu, \lambda$ before it reaches a trap, seal, transport, continuation, provenance, or naming suffix.

Next apply the root tail modulus ledger and criterion-tail pullback. Theorems 535.148 and 535.149 keep the tail request on the displayed μ, σ, λ ledger and pull the consumer back through the public criterion-tail route.

Now use root Real-seal nonescape. Theorem 535.150 identifies the seal-facing suffix as I, Q, H, C, P, N and rules out every unledgered completion object at that boundary.

Finally apply the window-meet route and root window seal synchronizer. Theorems 535.151 and 535.152 combine the prefix and suffix into the single finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Since the accepted carrier has no further coordinate, every synchronized root-facing Real or limit-seal read factors through exactly that displayed BEDC packet. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window real seal pullback`

Theorem 535.155 (Bounded monotone Cauchy witness window completion pullback)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing completion consumer or* $\mathsf{Real}^{\uparrow}$*-seal consumer using the synchronized window route factors through the finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

The pullback is completion-facing only through the displayed bounded source row, regular rational handoff, scheduled window, tail modulus, dyadic ledger, located trap, Real seal, componentwise transport, continuation, package provenance, and local naming row. It exports no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window completion pullback`

Proof. Begin with root window extraction. Theorem 535.146 places every root-facing completion read on the displayed prefix $S, F, \sigma, \mu, \lambda$ before the consumer can reach a seal or naming suffix.

Next apply the root tail modulus ledger and criterion-tail pullback. Theorems 535.148 and 535.149 keep the tail request inside μ, σ, λ and route the consumer back to the public criterion-tail packet.

Use root Real-seal nonescape. Theorem 535.150 identifies the seal-facing suffix as I, Q, H, C, P, N and blocks any reading that would replace the displayed rows by an unshown completion object.

Finally compose the window-meet route, root window seal synchronizer, and window Real-seal pullback. Theorems 535.151, 535.152, and 535.154 make the synchronized completion-facing route the single packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Since the accepted carrier contains no coordinate for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, every such consumer factors through exactly the displayed BEDC packet. □

Theorem 535.156 (Bounded monotone Cauchy witness window completion terminality)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every completion-facing consumer,* $\mathsf{Real}^{\uparrow}$*-seal-facing consumer, or convergence-seal-facing consumer that factors through the synchronized window-completion route factors through exactly the displayed packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

This packet is terminal for such consumers: it exposes the bounded source row, regular rational handoff, scheduled window, tail modulus, dyadic ledger, located trap, Real seal, componentwise transport, continuation, package provenance, and local naming row, and it exposes no selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, hidden tail selector, or choice-like tail service.

Proof. First extract the finite window route from the root carrier. The root window extraction and root tail modulus ledger, Definition 535.146 and Definition 535.148, force the consumer through $S, F, \sigma, \mu, \lambda$ and through the ledger μ, σ, λ before any located trap, Real seal, transport, continuation, provenance, or naming row is read.

Next pull the route back to the public criterion-tail and Real-seal boundaries. Apply criterion-tail pullback, root Real-seal nonescape, and the window-meet route, Definition 535.149, Definition 535.150, and Definition 535.151. These identify every completion-facing, Real-seal-facing, or convergence-seal-facing read with the same finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWi}}$ packet.

Then synchronize the seal suffix with the window prefix. The root window seal synchronizer and window Real-seal pullback, Definition 535.152 and Definition 535.154, make the prefix $S, F, \sigma, \mu, \lambda$ and suffix I, Q, H, C, P, N a single route rather than two independently selectable services.

Finally use the window completion pullback, Definition 535.155. It states that every synchronized completion-facing consumer already factors through $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Coordinate exhaustion of the accepted carrier now gives terminality. The carrier of Definition 535.5 has exactly the displayed coordinates and has no coordinate for a selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, hidden tail selector, or choice-like tail service. Hence all completion-facing consumers of the synchronized route factor through exactly the displayed packet, and there is no further completion-facing datum through which they can factor. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window completion terminality`

Theorem 535.157 (Bounded monotone Cauchy witness Real-seal modulus determinacy)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *and* $W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$ *be accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packets. Suppose the bounded source rows, regular rational handoff rows, scheduled window rows, tail modulus rows, dyadic ledger rows, located trap rows, Real seal rows,* hsame *transport rows,* Cont *route rows,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *rows agree componentwise as displayed carrier coordinates. Then every root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer that reads either packet reads the same finite Real-seal route*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$$

and hence the displayed tail-modulus packet uniquely determines the Real-seal readback. The determinacy is internal to the finite carrier route; it does not select a limit, identify quotient reals, assert host equality, use trichotomy, invoke ambient completeness, appeal to open-cover compactness, or install any choice-like tail selector.

Proof. First extract the shared root window. The root window extraction and root tail modulus ledger, Definition 535.146 and Definition 535.148, force both consumers through the componentwise-agreeing rows $S, F, \sigma, \mu, \lambda$ and the same ledger triple μ, σ, λ before either consumer can read a located trap or Real seal row.

Next pull the two reads back to the public criterion-tail boundary. The criterion-tail pullback and root Real-seal nonescape, Definition 535.149 and Definition 535.150, identify the remaining data as the displayed located trap, Real seal, transport, continuation, provenance, and local naming suffix rather than any unshown completion object.

Now synchronize the suffix with the window route. The window-meet route and root window seal synchronizer, Definition 535.151 and Definition 535.152, make the agreeing prefix and agreeing suffix a single finite route for both root-facing consumers.

Finally use the window Real-seal pullback, window completion pullback, and window completion terminality, Definition 535.154, Definition 535.155, and Definition 535.156. These theorems leave no completion-facing datum outside the accepted carrier.

Coordinate exhaustion of Definition 535.5 completes the comparison. Since the two accepted packets agree in every displayed coordinate and the carrier has no coordinate for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, every root-facing $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reads the same finite Real-seal route from W and W'. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal modulus determinacy`

Theorem 535.158 (Bounded monotone Cauchy witness monotone-source seal bridge)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet, and let* $M = (F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be the accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *monotone source component read by* W*. Every root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of* W *factors first through the monotone-source finite tail-window and cofinality handoff of* M*, then through the bounded-witness root window seal synchronizer for* W*, and finally through the bounded-witness Real-seal modulus determinacy route. The composed bridge is the displayed finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N,$$

with the monotone-source window rows $F, \sigma, \mu, \lambda, I, Q$ read before the bounded source row S is allowed to expose any root-facing seal consumption. The bridge supplies no quotient real equality, selected representative, trichotomy, open-cover compactness, ambient completeness, host real equality, or choice-like tail selector.

Proof. First read the monotone source through M. Theorem 438.15 forces any $\mathsf{Real}^{\uparrow}$ seal consumer using the monotone Cauchy source and a cofinal tail schedule through $F, \sigma, \mu, \lambda, I, Q$ before the seal row is readable, and it excludes quotient equality, selected representatives, host real order, ambient completeness, open-cover compactness, and choice.

Next attach the bounded-witness root row. Theorem 535.152 synchronizes the bounded source prefix $S, F, \sigma, \mu, \lambda$ with the seal suffix I, Q, H, C, P, N as one finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ route.

Finally apply Definition 535.157. Once the monotone-source handoff and bounded-witness synchronizer have displayed the same rows, the tail-modulus packet uniquely determines the root-facing Real-seal readback. The accepted packets contain no additional coordinate for quotient real equality, selected representatives, trichotomy, open-cover compactness, ambient completeness, host real equality, or choice-like tail selection, so every root-facing $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer factors through exactly the composed finite bridge. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier monotone source seal bridge`

Theorem 535.159 (Bounded monotone Cauchy witness window seal completeness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing* $\mathsf{Real}^{\uparrow}$ *consumer, root-facing* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer, or completion consumer that uses the synchronized window route factors through exactly the finite packet*

$$S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N.$$

Conversely, every displayed window-seal read of that packet supplies the whole readiness surface for those consumers: bounded source, regular rational handoff, scheduled window, tail modulus, dyadic ledger, located trap, Real seal, componentwise hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *naming row. No selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, hidden tail selector, or choice-like tail service is part of the readiness surface.*

Proof. First build the finite route. The window-meet route and root window seal synchronizer, Definition 535.151 and Definition 535.152, identify the root window prefix $S, F, \sigma, \mu, \lambda$ with the seal suffix I, Q, H, C, P, N as one packet.

Next pass to the Real and completion boundaries. The window Real-seal pullback and window completion pullback, Definition 535.154 and Definition 535.155, make every root-facing $\mathsf{Real}^{\uparrow}$, root-facing $\mathsf{CauchyLimitSeal}^{\uparrow}$, and completion read factor through that synchronized packet.

Then use terminality. By Definition 535.156, the displayed coordinates are the terminal readiness packet for completion-facing and seal-facing consumers, so no consumer can require data outside $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Finally apply Real-seal modulus determinacy, Definition 535.157. It states that componentwise agreement of the displayed packet determines the root-facing Real-seal readback, which gives the converse: reading the displayed window-seal packet supplies exactly the readiness surface used by the Real, limit-seal, and completion consumers. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window seal completeness`

Theorem 535.160 (Bounded monotone Cauchy witness window readback stability)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *and* $W' = (S', F', \sigma', \mu', \lambda', I', Q', H', C', P', N')$ *be accepted* BoundedMonotoneCauchy *packets. Suppose the displayed rows* $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$ *are preserved componentwise by* hsame *and that both packets replay the same displayed* Cont *window-seal route. Then every root-facing* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion consumer reads the same window-seal readiness surface from* W *and* W'. *The stability claim uses only the finite window-meet route, root window seal synchronizer, window Real-seal pullback, window completion pullback, terminality, and Real-seal modulus determinacy; it does not select limits, identify quotient reals, assert host equality, use trichotomy, invoke ambient completeness, appeal to open-cover compactness, or install choice-like tail services.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window readback stability`

Proof. First compose the shared window route. The window-meet route and root window seal synchronizer, Definition 535.151 and Definition 535.152, force both packets through the same displayed prefix and suffix under the assumed componentwise hsame transport and Cont replay.

Next pull the read to the Real and completion boundaries. The window Real-seal pullback and window completion pullback, Definition 535.154 and Definition 535.155, make every root-facing $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and completion consumer factor through the finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Finally apply terminality and Real-seal modulus determinacy. Theorem 535.156 leaves no completion-facing datum outside the displayed carrier, and Theorem 535.157 makes the componentwise-preserved window packet determine the Real-seal readback. The carrier has no coordinate for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail services, so the readiness surface read from W and W' is the same. □

Theorem 535.161 (Bounded monotone Cauchy witness CauchyCriterion selector handoff)**.** *Let* $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ *be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet, and let* $K = (W_K, M, T, Q_K, E, H_K, C_K, P_K, N_K)$ *be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet. Suppose the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows displayed by the two packets agree at the selector-seal boundary. Then every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing, or completion-facing read of the bounded monotone window-seal packet factors through the same finite* $\mathsf{CauchyCriterion}^{\uparrow}$ *budget-selector surface*

$$M, \quad Q_K, \quad T, \quad W_K, \quad E, \quad H_K, \quad C_K, \quad P_K, \quad N_K$$

before any $\mathsf{Real}^{\uparrow}$ *seal is exposed. The handoff uses only the shared StreamName–RegSeqRat–DyadicRatCore–Real rows and the finite support rows* BHist, hsame, Cont, Pkg, *and* NameCert*; it does not introduce quotient streams, selected limits, host real equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selectors.*

Proof. First close the bounded monotone window packet. The window seal completeness and window readback stability theorems, Definition 535.159 and Definition 535.160, put every $\mathsf{Real}^{\uparrow}$-facing, $\mathsf{CauchyLimitSeal}^{\uparrow}$-facing, and completion-facing read of W on the displayed finite packet $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

Next expose the selector surface. The CauchyCriterion StreamName Real budget-selector boundary, Definition 392.37, makes every consumer sharing the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows read through $M, Q_K, T, W_K, E, H_K, C_K, P_K, N_K$ before selector output is exposed.

Then use uniform selector-seal compatibility, Definition 392.39. It records that the selector-facing CauchyCriterion route and any matching seal route factor through the same finite StreamName–RegSeqRat–DyadicRatCore–Real surface with the support rows BHist, hsame, Cont, Pkg, and NameCert.

Finally match the shared boundary rows. By hypothesis the bounded monotone packet and the CauchyCriterion packet display the same four public rows at the selector-seal boundary, so the terminal window-seal read of W must pass through the CauchyCriterion budget-selector packet before a $\mathsf{Real}^{\uparrow}$ seal is exposed. Neither accepted carrier has an extra completion coordinate, and the listed exclusions have no displayed row in either packet. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier cauchycriterion selector handoff`

Theorem 535.162 (Bounded monotone Cauchy witness window-seal readback completeness)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. A current Phase Real or Completion readback through* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or any completion-facing seal is complete exactly when it exposes the bounded source row S, regular rational handoff F, scheduled window σ, λ, tail modulus μ, dyadic ledger, located trap I, Real seal Q, componentwise* hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row N. Conversely, any accepted packet exposing those rows supplies the whole readback surface for the bounded monotone witness window. The statement excludes selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selection.*

Proof. First expose the bounded source prefix. The root window extraction and root tail modulus ledger, Definition 535.146 and Definition 535.148, place every Phase Real or Completion read on $S, F, \sigma, \mu, \lambda$ and the dyadic tail ledger before a trap, seal, transport, route, provenance, or naming row can be read.

Next pull the readback to the public seal boundary. The criterion-tail pullback and root Real-seal nonescape, Definition 535.149 and Definition 535.150, identify the remaining surface as I, Q, H, C, P, N rather than a selected limit or host completion object.

The route is then synchronized as one finite packet. Apply the window-meet route and root window seal synchronizer, Definition 535.151 and Definition 535.152, to bind the source prefix, tail ledger, located trap, Real seal, transports, continuations, provenance, and local name into the same displayed BHist packet.

For the forward direction, window completion terminality and Real-seal modulus determinacy, Definition 535.156 and Definition 535.157, leave no completion-facing or Real-facing coordinate outside the displayed rows. Thus a complete $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or completion-facing read must expose exactly the listed surface.

For the converse, start from an accepted packet exposing all listed rows. The CauchyCriterion selector handoff, Definition 535.161, supplies the selector-facing budget route before any Real seal is exposed, while the synchronized window packet supplies the completion-facing suffix. Since the carrier contains no row for selected limits, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, these rows supply the whole readback surface. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window seal readback completeness`

Theorem 535.163 (Bounded monotone Cauchy witness root window terminal factorization)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every root-facing* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion consumer of W factors through the single displayed root-window route*

$$S, \quad F, \quad \sigma, \quad \mu, \quad \lambda, \quad I, \quad Q, \quad H, \quad C, \quad P, \quad N.$$

This route is the only completion-facing factorization surface for the accepted packet: any consumer preserving the bounded source, regular rational handoff, scheduled root window, tail modulus ledger, located trap, Real seal, componentwise hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row preserves the same root-window readback. No selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selector is part of the factorization.*

Proof. First build the root-window spine. Root window extraction and the root tail modulus ledger, Definition 535.146 and Definition 535.148, put every root-facing read on the displayed bounded source, rational handoff, scheduled window, and tail modulus rows.

Next pull the spine to the public seal boundary. Criterion-tail pullback and root Real-seal nonescape, Definition 535.149 and Definition 535.150, force the located trap, Real seal, componentwise transports, continuation replay, provenance, and local naming row to remain attached to that same root route.

Finally close the completion-facing side. Window completion terminality, Definition 535.156, says a completion consumer has no further route outside the displayed packet, and window-seal readback completeness, Definition 535.162, identifies the whole exposed surface as $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. Therefore every root-facing $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or completion consumer factors through exactly this route, and the excluded objects have no carrier coordinate through which a second factorization could pass. □

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window terminal factorization`

Theorem 535.164 (Bounded monotone Cauchy witness finite-window factorization package)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read of W factors through the finite prefix*

$$S, \quad F, \quad \sigma, \quad \mu, \quad \lambda$$

and then through the finite suffix

$$I, \quad Q, \quad H, \quad C, \quad P, \quad N.$$

The prefix records the bounded source, regular rational handoff, scheduled root window, tail modulus ledger, and window selector. The suffix records the located trap, Real seal, componentwise hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *row. Thus any* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *RegSeqRat, StreamName, or completion-seal consumer preserving these eleven rows reads the same finite window package.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier window factorization`

Proof. First isolate the prefix. Root window extraction and the root tail-modulus ledger, Definition 535.146 and Definition 535.148, place every source-facing and tail-facing read on $S, F, \sigma, \mu, \lambda$ before a located trap or Real seal can be exposed.

Next bind the suffix to that same prefix. Criterion-tail pullback and root Real-seal nonescape, Definition 535.149 and Definition 535.150, keep the trap, seal, transport, continuation, provenance, and local naming rows attached to the displayed root window.

The window-meet route and root window seal synchronizer, Definition 535.151 and Definition 535.152, identify the two halves as one finite BHist packet. The terminal factorization theorem, Definition 535.163, then rules out any further completion-facing route outside these eleven rows. □

Theorem 535.165 (Bounded monotone Cauchy witness Real-seal route)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ packet. The $\mathsf{Real}^{\uparrow}$ seal of W is readable only through the displayed finite route*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$$

carried by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$. This route excludes quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, and choice-like tail selection. In particular, a Real-seal consumer receives the finite BHist row, the componentwise hsame transports, the Cont replay, the Pkg provenance row, and the local NameCert row, and receives no additional real-completion witness.

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier real seal route`

Proof. Start from root Real-seal nonescape, Definition 535.150. It says the Real-facing row cannot leave the finite root-window packet for a selected limit, a host equality judgment, or an ambient completion object.

The window-meet route and window-seal synchronizer, Definition 535.151 and Definition 535.152, then display the seal route as the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support carried by W.

Finally use terminal factorization, Definition 535.163. Since the accepted packet has no coordinate for quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like tail selection, no Real-seal route can factor through those objects. □

Theorem 535.166 (Bounded monotone Cauchy witness modulus ledger exhaustion)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ packet. The modulus ledger μ, σ, λ is exhausted before either located-trap row I or Real-seal row Q is readable. Equivalently, every located-trap, Real-seal, $\mathsf{CauchyLimitSeal}^{\uparrow}$, or completion-facing read first passes through the scheduled window and tail-modulus budget, and only then reaches I, Q, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier modulus ledger exhaustion`

Proof. The root tail-modulus ledger, Definition 535.148, records the selector and modulus rows σ, μ, λ as the finite budget consumed by the root window. Thus a trap or seal row cannot be read before that budget is displayed.

Criterion-tail pullback, Definition 535.149, transports any criterion-facing tail read back to the same budget rows. Root Real-seal nonescape, Definition 535.150, keeps the Real-seal side on the finite packet rather than on a selected limit or ambient completion object.

The finite-window factorization package above and the terminal factorization theorem, Definition 535.164 and Definition 535.163, identify every downstream read as a prefix read through $S, F, \sigma, \mu, \lambda$ followed by a suffix read through I, Q, H, C, P, N. Hence the modulus ledger is exhausted before the located-trap and seal rows are readable. □

Corollary 535.167 (Bounded monotone Cauchy witness root-window extraction strict local obstruction)**.** *Let $W = (S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N)$ be an accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row in the root-window extraction, tail-modulus ledger, or Real-seal handoff surface of W is cited against a named converse or reuse route. Then the row is a strict local obstruction: it blocks only the named route whose consumer-facing read tries to omit, replace, detach, or bypass one of the displayed packet coordinates*

$$S, \quad F, \quad \sigma, \quad \mu, \quad \lambda, \quad I, \quad Q, \quad H, \quad C, \quad P, \quad N.$$

Equivalently, every permitted consumer-facing use of the row factors through the finite root-window prefix

$$S, F, \sigma, \mu, \lambda$$

and the seal suffix

$$I, Q, H, C, P, N,$$

with support only from the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ *rows. The same row cannot be reinterpreted as a global impossibility theorem for* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, *a host theorem about real completion or monotone convergence, hidden external validation of the failed route, a selected limit, quotient real equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or a choice-like tail selector.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier root window strict obstruction`

Proof. Start with root-window extraction. By Definition 535.146, every root-facing convergence-seal, limit-seal, or Real-seal consumer first reads the finite prefix $S, F, \sigma, \mu, \lambda$ before it reaches the located trap, the Real seal, transport, continuation, provenance, or naming rows. Thus a displayed negative row attached to the extraction surface has a named input coordinate inside that prefix; it is not a statement about arbitrary bounded monotone witnesses or arbitrary completion procedures.

Next use the tail-modulus ledger, Definition 535.148. Any root-facing tail request is ledgered by μ, σ, λ before I, Q, or any completion-facing handoff is read. Hence a converse or reuse route that tries to replace the scheduled window, tail modulus, or dyadic ledger by a selected tail, an unledgered completion object, or a choice-like selector has no admitted coordinate. The obstruction blocks precisely that attempted replacement of the displayed ledger and does not acquire wider force.

Now pass to the seal boundary. The root Real-seal nonescape theorem, Definition 535.150, identifies the remaining suffix as exactly I, Q, H, C, P, N: the located trap, Real seal route, componentwise transport, continuation, package provenance, and local naming rows. It also says that the handoff to $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ exports no selected limit, quotient equality, host real equality, trichotomy, ambient completeness, open-cover compactness, or unledgered completion principle.

The consumer-facing factorization is therefore finite. By Definition 535.149, Definition 535.151, and Definition 535.163, any admitted Real-facing, limit-seal-facing, or completion-facing use of the

root window is pulled back to the same BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$ packet carried by $S, F, \sigma, \mu, \lambda, I, Q, H, C, P, N$.

It remains to exclude stronger readings. A global impossibility theorem would require a conclusion not indexed by the accepted packet W and not confined to the named failed route. A host theorem would require a host-facing real-completion, monotone-convergence, equality, or compactness coordinate outside the packet. Hidden external validation would require an extra certificate channel beyond P, N, and the displayed continuation and transport rows. A selected limit, quotient real equality, host equality, trichotomy, ambient completeness, open-cover compactness, or choice-like selector would require one of the coordinates explicitly absent from the extraction, ledger, and nonescape theorems. Since the accepted carrier exposes none of those coordinates, the displayed row blocks only the named local converse or reuse route and rejects every unlisted interpretation. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ packet is generated from bounded monotone source rows, a regular rational row, a stream schedule, an explicit Cauchy tail witness, dyadic tolerance and upper-bound ledgers, a located interval trap, a real-seal row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 535.19 binds the carrier, socket separation, classifier transport, Real seal handoff, and ledger exhaustion rows to the displayed RegSeqRatUp, StreamNameUp, CauchyModulusUp, DyadicRatCoreUp, LocatedIntervalUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close monotone convergence as a public theorem, quotient real construction, selected limiting representatives, host real equality, ambient completeness, trichotomy, or open-cover compactness.

Upgrade path: The paper theory axis is mature for the displayed bounded-monotone Cauchy witness packet, while Lean verification and bridge checking remain independent obligations.

Chapter 536

A Concrete Naming Certificate for $\mathsf{ApophaticSocket}^{\uparrow}$

The $\mathsf{ApophaticSocket}^{\uparrow}$ packet realizes apophatic naming as a finite certificate surface. It names the socket kind, the requested external supply shape, the audit gate that keeps the request visible, the site row where the request is made, the transport and replay rows that preserve the boundary, the package provenance, and the local NameCert row. This socket specializes the boundary-name surface of chapter 543: ApophaticName supplies the negative naming row, while ApophaticSocket adds the requested-supply shape, audit gate, site row, and noninternalization ledger. The packet does not make the requested supply internally available.

Definition 536.1 (ApophaticSocket ledger carrier)**.** *An* $\mathsf{ApophaticSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, S_0, G, A, L, Q, P, N)$$

with the following displayed rows:

1. *K is the socket-kind row, recording whether the named boundary concerns an unwitnessed selector, quotient identification, proposition-extensionality collapse, causal-continuity assumption, inscription novelty, outermost* T *supply, or another displayed externality class;*
2. *S_0 is the requested-supply-shape row, recording the content requested from the boundary without supplying that content as an internal constructor, definition, theorem, or setup field;*
3. *G is the audit-gate row, recording the finite question that detects visible socket use and rejects hidden internalization of the requested supply;*
4. *A is the site row, recording the manuscript, Lean, or governance location at which the socket is invoked and the local claim that depends on it;*
5. *L records componentwise* hsame *transport for K, S_0, G, A, the continuation row, the package row, and the local naming row;*
6. *Q records the displayed* Cont *routes by which consumers inspect the socket kind, requested supply shape, audit gate, site row, provenance, and local certificate;*
7. *P is the* Pkg *provenance over* BHist*,* Hist*,* GAP*,* hsame*,* Cont*,* Pkg*,* NameCert*, apophatic names, audit-gate rows, site rows, and origin ledgers;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for the requested external supply itself, no hidden theorem converting that supply into substrate content, no anonymous import channel, and no route by which the apophatic name becomes a positive internal description.

Theorem 536.2 (ApophaticSocket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$ *surface for the carrier of Definition 536.1 consists of carrier admission for*

$$S = (K, S_0, G, A, L, Q, P, N),$$

socket-kind exposure, requested-supply-shape display, audit-gate exactness, site-row locality, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, and non-internalization for the requested external supply.*

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 536.1. The packet exposes exactly the rows K, S_0, G, A, L, Q, P, N.

The naming obligations are read row by row. The socket kind is supplied by K, the requested supply shape by S_0, the gate question by G, and the invocation site by A.

Transport and consumer access are confined by L and Q. These rows replay the displayed socket packet under componentwise hsame movement and expose only the recorded continuation routes.

Finally P and N record provenance and the local naming certificate. Since the requested external supply has no coordinate in the carrier, the NameCert surface can name the boundary without exporting the supply as internal content. □

Theorem 536.3 (ApophaticSocket gate correspondence)**.** *For an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet, the audit-gate row* G *is the complete public gate corresponding to the socket-kind row* K *and the requested-supply-shape row* S_0*. A downstream consumer may treat a use of the socket as visible exactly when it factors through the rows* K, S_0, G, A, L, Q, P, N *and may reject any use of the same requested supply that has no displayed gate row.*

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier gate correspondence`

Proof. Read the socket side first. The rows K and S_0 record the kind of externality and the shape of supply being requested.

Read the enforcement side next. The row G records the finite audit question attached to that socket, while A fixes the site at which the question is asked.

The transport and replay rows L and Q preserve the same gate for downstream consumers, and P, N package and name it. Therefore visible socket use is exactly use through the displayed packet, and an un-gated use of the requested supply has no accepted route through this carrier. □

Theorem 536.4 (ApophaticSocket noninternalization boundary)**.** *For an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet, the requested supply shape* S_0 *remains boundary data. The packet may name the supply shape, expose the site where it is requested, attach an audit gate, transport the displayed rows by* hsame*, replay them by* Cont*, and record* Pkg *and* NameCert *provenance, but it cannot produce the requested external supply as an internal constructor, definition, theorem, setup field, or imported host principle.*

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier noninternalization boundary`

Proof. Project the requested-supply-shape row S_0 from the carrier. It records the shape of the requested externality, not an inhabitant or proof of that externality inside the substrate.

The gate and site rows G and A keep the request visible. They classify and locate the boundary use, but they do not add a coordinate for the requested supply itself.

The remaining rows L, Q, P, N only transport, replay, package, and name the displayed boundary packet. Because the carrier inventory contains no constructor, definition, theorem, setup field, or host-principle coordinate for the requested supply, the packet preserves the noninternalization boundary. □

Theorem 536.5 (ApophaticSocket consumer surface)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet factors through the finite rows*

$$K, S_0, G, A, L, Q, P, N.$$

Such a consumer may inspect the socket kind, requested supply shape, audit gate, invocation site, transport rows, replay routes, provenance, and local naming certificate. It may not extract a positive description of the external source, a hidden witness for the requested supply, an unmarked dependency, or a theorem object internalizing the socket.

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier consumer surface`

Proof. Let a downstream consumer route be given. By Definition 536.1, the source carrier is the finite tuple K, S_0, G, A, L, Q, P, N.

The consumer first reads the socket, supply-shape, gate, and site rows. These rows provide the full public boundary surface.

The rows L and Q supply all permitted transport and replay, and P, N supply provenance and the local certificate. The non-extraction claim follows from the carrier inventory: none of the forbidden objects appears among the displayed rows, so no consumer route out of the packet can project it. □

Theorem 536.6 (ApophaticSocket obligation closure package)**.** *The* $\mathsf{ApophaticSocket}^{\uparrow}$ *obligation surface is the finite package consisting of the carrier row of Definition 536.1, the local NameCert obligations of Definition 536.2, the gate correspondence of Definition 536.3, the noninternalization boundary of Definition 536.4, and the consumer surface of Definition 536.5. Closing* $\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$ *requires exactly carrier admission, socket-kind exposure, requested-supply-shape display, audit-gate exactness, site-row locality, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming exactness for* K, S_0, G, A, L, Q, P, N.

Proof. The carrier definition lists all rows available to the package. The NameCert theorem records the local naming obligations on those rows, while the gate, noninternalization, and consumer theorems identify the public uses that must remain visible.

Reading these clauses together gives the obligation surface. Each required proof target is tied to one displayed coordinate of K, S_0, G, A, L, Q, P, N, and no additional supply coordinate is introduced. □

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier obligation closure package`

Theorem 536.7 (ApophaticSocket gated supply route exhaustion)**.** *Every visible use of the requested supply shape in an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet factors through the finite rows*

$$K, S_0, G, A, L, Q, P, N.$$

In particular, the socket kind is read through K, the requested supply shape through S_0, the gate through G, the invocation site through A, transport and replay through L, Q, provenance through P, and local naming through N.

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier gated supply route exhaustion`

Proof. Let a visible supply use be given. By the gate correspondence theorem, visibility means that the use has the displayed audit-gate row G attached to the socket kind and requested supply shape.

The only carrier coordinates that can support such a use are K, S_0, G, A, L, Q, P, N. The rows L and Q preserve and replay the same displayed packet, and P, N package and name it.

Therefore every visible use is exhausted by these rows. A route outside them has no accepted coordinate in the carrier and so cannot count as a visible socket use. □

Theorem 536.8 (ApophaticSocket boundary noninternalization ledger)**.** *For an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet, no requested supply named by S_0 is internalized by the ledger. The ledger may record the supply shape, the audit question, the invocation site, permitted transport, replay, provenance, and local naming, but it cannot produce the requested supply as a constructor, definition, theorem, setup field, anonymous import, or positive internal description.*

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketUp StdBridge`

Proof. Project the requested-supply-shape row S_0. It records the boundary request as ledger data and not as a substrate object satisfying the request.

The gate and site rows G, A keep the request inspectable. The transport and replay rows L, Q preserve the same visible request, while P, N record provenance and local naming.

Since the carrier inventory contains no row for the requested supply itself, no ledger projection can internalize it. The only available outputs are the displayed boundary rows and their permitted read routes. □

Theorem 536.9 (ApophaticSocket scoped supply ledger)**.** *Every visible supply request in an accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *packet is scoped by the concrete rows*

$$K, S_0, G, A, L, Q, P, N.$$

The socket kind is read through K, the requested external supply shape through S_0, the audit gate through G, the invocation site through A, componentwise hsame *transport through L, displayed* Cont *replay through Q,* Pkg *provenance through P, and local* NameCert *naming through N. This scoped ledger records the request without internalizing the requested external supply.*

Proof. Project the carrier of Definition 536.1. Its visible coordinates are exactly K, S_0, G, A, L, Q, P, N, so every accepted packet already carries the supply request as a finite displayed ledger.

Apply the gated supply route exhaustion of Definition 536.7. Any visible use of the requested supply shape must pass through the socket kind, supply shape, gate, site, transport, replay, provenance, and local naming rows.

Finally apply the boundary noninternalization ledger of Definition 536.8. The scoped packet can expose the request and its audit route, but it has no coordinate from which the requested external supply itself can be projected. □

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier scoped supply ledger`

Theorem 536.10 (ApophaticSocket kernel-scope obligation package)**.** *For an accepted* ApophaticSocket$^{\uparrow}$ *packet, the rows*

$$K, S_0, G, A, L, Q, P, N$$

are the complete scoped kernel surface for carrier admission, gate correspondence, noninternalization, and consumer factoring. Carrier admission reads exactly these rows; gate correspondence binds K, S_0, G, A to visible socket use; noninternalization is recorded by the absence of a requested-supply coordinate; and every consumer factors through L, Q, P, N together with the same displayed socket rows.

Proof. Start from the carrier and local NameCert obligations. Definition 536.1 lists exactly K, S_0, G, A, L, Q, P, N, and Definition 536.2 reads those rows as carrier admission, socket-kind exposure, requested-supply display, gate exactness, site locality, transport, replay, provenance, and local naming.

Next bind the gate and consumer routes. Definition 536.3 makes G the public gate for the socket kind K and requested supply shape S_0, while Definition 536.5 forces every downstream read through the displayed packet.

Finally apply the boundary and route ledgers. Definition 536.7 and Definition 536.9 exhaust visible supply use through the same rows, and Definition 536.8 keeps the requested external supply outside the carrier. These clauses give the scoped kernel surface without adding an internal supply coordinate. □

Theorem 536.11 (ApophaticSocket positive supply refusal scope)**.** *For an accepted* ApophaticSocket$^{\uparrow}$ *packet with visible boundary request rows*

$$K, S_0, G, A, L, Q, P, N,$$

any visible request through these rows remains a request row. It cannot be consumed as a constructor, theorem, setup field, import permission, or positive internal description of the requested supply.

Lean4 ✓ `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier positive supply refusal scope`

Proof. Use the gate correspondence of Definition 536.3. Visibility of the request is exposed only through the socket kind K, requested supply shape S_0, audit gate G, and site row A.

Next apply the noninternalization boundary of Definition 536.4 and the boundary noninternalization ledger of Definition 536.8. These results say that the carrier records the request shape and its audit route without supplying a constructor, theorem, setup field, import permission, or positive internal description.

Finally use the scoped supply ledger of Definition 536.9 together with the kernel-scope obligation package of Definition 536.10. They exhaust every visible consumer route through K, S_0, G, A, L, Q, P, N, so no downstream consumer can reinterpret the visible boundary request as positive internal supply. □

Theorem 536.12 (ApophaticSocket refusal ledger boundary)**.** *For an accepted* ApophaticSocket$^{\uparrow}$ *packet with visible request rows*

$$K, S_0, G, A, L, Q, P, N,$$

any socket request through those rows remains a refusal-ledger boundary row. It cannot be consumed as positive supply, anonymous import, theorem object, setup field, or constructor.

Proof. Apply the gate correspondence of Definition 536.3. The visible request is attached to the socket kind K, requested supply shape S_0, audit gate G, and site row A.

Next use Definition 536.4 and Definition 536.8. These place the request in the boundary ledger and exclude an internal coordinate that could act as positive supply, anonymous import, theorem object, setup field, or constructor.

Finally apply Definition 536.9 and Definition 536.10. Every consumer of the request factors through K, S_0, G, A, L, Q, P, N, so the ledger row remains a visible refusal boundary rather than a consumable supply object. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticSocket}^{\uparrow}$ packet is generated from a socket-kind row, a requested-supply-shape row, an audit-gate row, a site row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public export is the consumer surface and gated supply route over K, S_0, G, A, L, Q, P, N, as displayed by Definition 536.5 and Definition 536.7, with the carrier, NameCert rows, positive supply refusal, refusal ledger boundary, scoped supply ledger, and kernel-scope obligation package of Definition 536.1, Definition 536.2, Definition 536.11, Definition 536.12, Definition 536.9, and Definition 536.10 fixing the same finite socket ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ApophaticSocket}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ApophaticSocketUp.ApophaticSocketCarrier\_consumer\_surface`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close bridge checking, a positive description of the external source, hidden witnesses for requested supply, or global closure for every apophatic name.

Upgrade path: No further formal-axis upgrade remains for this bridge-checked chapter.

Chapter 537

A Concrete Naming Certificate for $\mathsf{ClosedTermSubstitution}^{\uparrow}$

The $\mathsf{ClosedTermSubstitution}^{\uparrow}$ surface packages the MetaCIC boundary law for substitution as finite NameCert data. It records the term row, the closedness boundary, the substituting value, the shift and substitution witnesses, componentwise transport, continuation reads, provenance, and the local naming row that prevents a substitution payload from becoming an unregistered free variable.

The displayed boundary rows are consumed from chapter 527. The $\mathsf{ClosedTermSubstitution}^{\uparrow}$ surface specializes that finite boundary to the inert value-payload route $Q = (T, d, V, S, H, C, P, N)$, without adding subject reduction, evaluator correctness, host CIC truth, or an unregistered variable source.

The packet is the local closed-term substitution surface consumed by chapter 527 and by the compiler-facing schedule of chapter 540. The dependency is the displayed term row, closedness boundary, substitution and shift witness row, hsame transport, Cont route, Pkg provenance, and local NameCert ledger, not semantic evaluator correctness or subject reduction.

Definition 537.1 (Closed-term substitution carrier)**.** *A* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, d, V, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC term row, carrying a closed generator tree for the inspected term;*
2. *d is the boundary row, recording the local de Bruijn depth at which closedness is certified;*
3. *V is the substituting value row, displayed as payload data but not consumed by the closed term at boundary d;*
4. *S is the substitution and shift witness row, containing the boundary equations for* $\mathsf{substitute}(d, V, T)$ *and* $\mathsf{shift}(d, 1, T)$ *on the classified slice;*
5. *H records componentwise* hsame *transports for T, d, V, S, package provenance, and the local naming row;*
6. *C records the* Cont *read routes that expose the term, boundary, value, witness row, and returned term equality;*
7. *P is the* Pkg *provenance over the MetaCIC term generator,* ClosedAt *classifier, substitution and shift generators,* BHist, hsame, Cont, *and* NameCert*;*

8. N is the local $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}$ *row.*
The carrier is exactly this finite packet. It contains no open-term subject-reduction theorem, no semantic evaluator correctness proof, no host CIC truth predicate, no future binder budget, and no unregistered free-variable source.

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionUp.ClosedTermSubstitutionTasteGate single carrier alignment`

Theorem 537.2 (Closed-term substitution boundary inertness)**.** *For an accepted* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *carrier $Q = (T, d, V, S, H, C, P, N)$, every substitution read at the displayed boundary is inert on the classified term row. The witness row S records the boundary equation*

$$\mathsf{ClosedAt}(d, T) \Rightarrow \mathsf{substitute}(d, V, T) = T$$

and the companion shift equation at the same boundary. The value row V remains displayed payload data; it is not projected as a consumed variable of T.

Proof. Project the term, boundary, and witness rows of the carrier. The classifier row says that every free de Bruijn index visible in T lies below d.

The substitution generator can change a variable only at the displayed boundary, or adjust a variable above that boundary. Both branches require an index at or beyond d, and the closedness witness excludes such an index inside T.

The compound cases are transported componentwise through H and replayed by the read routes in C. Thus the only public output of the substitution read is the equality stored in S, while V remains a displayed but unconsumed payload row. □

Theorem 537.3 (Closed-term substitution NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}$ *surface consists of carrier admission for $Q = (T, d, V, S, H, C, P, N)$, term-row exposure, boundary-row exposure, value-row display without consumption, substitution and shift witness accountability, componentwise* hsame *stability, displayed* Cont *read locality,* Pkg *provenance, and non-escape for open-term subject reduction, semantic evaluator correctness, host truth, future binder budgets, and unregistered free variables.*

Proof. Carrier admission is projection from Definition 537.1. The packet exposes exactly the MetaCIC term row, boundary row, value row, witness row, transport rows, continuation routes, provenance, and local naming row.

Boundary accountability follows from Definition 537.2. The substitution and shift witnesses are read from S, and the continuation routes in C expose those witnesses without adding a further generator.

Finally inspect the excluded objects. None appears as a carrier coordinate, and the transport row H preserves only the displayed coordinates. Therefore a consumer of $\mathsf{ClosedTermSubstitution}^{\uparrow}$ cannot extract an open-term theorem, evaluator semantics, host truth predicate, future binder budget, or unregistered free variable from this packet. □

Theorem 537.4 (Closed-term substitution closure-preservation provenance)**.** *Every* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *boundary-inert read consumed by closure-preservation or beta-substitution routes remains the displayed packet*

$$Q = (T, d, V, S, H, C, P, N).$$

Downstream closure-preservation consumers may cite this substitution surface only through the provenance rows of chapter 794, chapter 716, and chapter 483. Those consumers add closure-preservation and typing-route context, but they do not add an open-term theorem, evaluator semantics, host truth predicate, or free-variable source to the closed substitution packet.

Proof. Begin with the carrier of Definition 537.1. It exposes only T, d, V, S, H, C, P, N as the term, boundary, value, witness, transport, route, provenance, and local naming rows.

Apply Definition 537.2 and Definition 537.3. The inertness theorem keeps the substitution read at the displayed boundary, and the NameCert obligations keep every public read inside the same packet.

The sibling closure-preservation, beta-substitution, and subject-reduction chapters consume this packet through their own provenance rows. Their additional rows record closure-preservation and typing-route context downstream; they do not enlarge Q with open-term semantics, evaluator truth, host truth, or a new free-variable source. □

Theorem 537.5 (Closed substitution source boundary)**.** *Every accepted* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *carrier* $Q = (T, d, V, S, H, C, P, N)$ *exposes only the displayed MetaCIC term row, the closedness boundary row, the value payload row, the substitution and shift witness row, the* hsame *transport row, the* Cont *read routes, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}$ *row. No other source row is public from this carrier.*

Proof. Project Q through Definition 537.1. The eight coordinates are exactly the displayed term, boundary, value, witness, transport, route, provenance, and naming rows.

The source boundary is the one imported from chapter 527. That chapter supplies the boundary law consumed here, while this carrier only specializes the law to the inert value-payload route.

The excluded sources are therefore absent by coordinate inspection. An accepted read may expose T, d, V, S, H, C, P, N, but it has no coordinate from which to extract an open-term theorem, evaluator semantics, host truth predicate, future binder budget, or unregistered variable source. □

Theorem 537.6 (Closed substitution ledger exhaustion)**.** *Every public substitution read from an accepted* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *carrier factors through* S*,* H*,* C*, and* P*. Consequently the ledger records exactly boundary inertness together with substitution and shift witness accountability, and no further substitution authority is exported.*

Proof. Apply Definition 537.2. The only boundary equation made public for substitution is the equation stored in the witness row S, together with the companion shift equation at the same boundary.

Apply Definition 537.3. Public access to the witness row is routed through the displayed continuation row C, stabilized through the componentwise hsame row H, and sourced through the package provenance row P.

Thus a substitution consumer exhausts the ledger by reading S, transporting through H, following C, and citing P. No additional row remains from which to read a second substitution law, a semantic evaluator claim, or a host-level truth predicate. □

Theorem 537.7 (Closed substitution binder-scope nonescape)**.** *For every accepted* $\mathsf{ClosedTermSubstitution}^{\uparrow}$ *carrier* $Q = (T, d, V, S, H, C, P, N)$*, binder depth, value payload, and continuation reads cannot export open-term subject reduction, evaluator correctness, host CIC truth, or an unregistered variable source.*

Proof. Inspect the carrier coordinates. The boundary coordinate d records only the local de Bruijn depth at which closedness is certified, and the value coordinate V is displayed payload data rather than a consumed variable of T.

The continuation coordinate C records routes that expose the term, boundary, value, witness row, and returned term equality. By Definition 537.3, those routes do not add a further generator.

The non-escape clauses stated in the carrier and NameCert rows exclude open-term subject reduction, evaluator correctness, host CIC truth, future binder budgets, and unregistered free variables. Since none of those objects is a coordinate of Q, the binder scope and payload reads cannot export them. □

The mature package for $\mathsf{ClosedTermSubstitution}^{\uparrow}$ is the closed-substitution boundary package consumed by the boundary, compiler, closure-preservation, beta-substitution, and subject-reduction rows. Its completeness statement is coordinate exhaustion: every accepted substitution-boundary consumer reads T, d, V, S, H, C, P, N, with S carrying the inert substitution and companion shift witnesses, and no consumer receives evaluator correctness, host CIC truth, future binder allocation, or an unregistered variable source.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{ClosedTermSubstitution}^{\uparrow}$ packet is generated from a MetaCIC term row, a closedness boundary row, a displayed substitution value, substitution and shift witness rows, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 537.1, Definition 537.2, Definition 537.3, Definition 537.5, Definition 537.6, Definition 537.7, and

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionUp.taste gate`

bind the carrier, boundary inertness, NameCert obligations, source boundary, ledger exhaustion, binder-scope nonescape, and closed-substitution consumer completeness surface to the mature package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedTermSubstitution}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ClosedTermSubstitutionUp.ClosedTermSubstitutionTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close open-term subject reduction, semantic evaluator correctness, host CIC truth, future binder allocation, or a complete self-compilation theorem.

Upgrade path: Bridge checking remains the verification-axis upgrade for the closed-substitution package.

Chapter 538

A Concrete Naming Certificate for $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$

The $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$ surface packages the MetaCIC closed-term substitution boundary as a finite naming certificate. It records the exact packet by which a `Term` row, a `ClosedAt` classifier row, the `shift` and `substitute` operation rows, displayed hsame transport, Cont routes, Pkg provenance, and local NameCert rows show that substitution is inert on the closed-at-boundary slice.

The sibling closed-term substitution boundary of chapter 527 supplies the finite `Term`, `ClosedAt`, `shift`, and `substitute` surface that this chapter specializes to the closed-at-boundary inertness route.

Definition 538.1 (ClosedSubstitutionBoundary carrier)**.** *A* $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, D, V, K, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC* `Term` *generator row, recording that terms are built only from* `var`*,* `app`*,* `lam`*,* `pi`*, and* `sort`*;*
2. *D is the depth row for the boundary at which the classifier is read;*
3. *V is the substitution payload term row, carried as data but not inspected by the inertness route;*
4. *K is the* `ClosedAt` *classifier row certifying that the term row is closed at depth D;*
5. *R is the* `shift` *operation row, including the boundary-budget route witnessed by the closed slice;*
6. *Q is the* `substitute` *operation row, including the route that attempts substitution at depth D with payload V;*
7. *H records componentwise* hsame *transports for the term row, depth row, payload row, classifier row,* `shift` *row,* `substitute` *row, continuation row, package row, and local naming row;*
8. *C records the displayed* Cont *routes by which consumers read the term, classifier, shift route, substitute route, provenance, and local certificate;*
9. *P is the* Pkg *provenance over* BHist*,* `Term`*,* `ClosedAt`*,* `shift`*,* `substitute`*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionBoundary}^{\uparrow}}$ *row.*

The carrier contains no semantic substitution theorem for arbitrary open terms, no subject-reduction theorem for an evaluator, no host equality of syntax quotients, and no untracked binder name.

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundaryTasteGate single carrier alignment`

Theorem 538.2 (ClosedSubstitutionBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionBoundary}^{\uparrow}}$ *surface for the carrier of Definition 538.1 consists of carrier admission for*

$$S = (T, D, V, K, R, Q, H, C, P, N),$$

`ClosedAt` classifier stability, boundary-budget stability for `shift`, inertness of the displayed `substitute` route on the closed-at-boundary slice, componentwise hsame *transport, displayed* Cont *locality,* Pkg *provenance, and non-escape for unregistered free variables, arbitrary open-term substitution, evaluator subject reduction, host syntax quotients, and ambient metatheoretic truth.*

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary namecert obligations`
↪ variant `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundaryNameCertObligations`

Proof. Carrier admission is projection from Definition 538.1. The rows $T, D, V, K, R, Q, H, C, P, N$ are the only rows admitted to the packet.

The classifier row K fixes the boundary. It records that the term row has no free de Bruijn index at or above the displayed depth D, so the `shift` route R and the `substitute` route Q are read only through that closed slice.

Transport and consumer access are confined by H and C. These rows replay the same term, boundary, payload, classifier, and operation routes under componentwise hsame movement and do not add a hidden binder name or a second substitution surface.

Finally P and N record provenance and the local certificate. Since the excluded objects have no coordinate in the carrier, no NameCert projection can export arbitrary open-term substitution, evaluator subject reduction, host syntax quotient equality, or ambient metatheoretic truth. □

Theorem 538.3 (ClosedSubstitutionBoundary substitute inertness route)**.** *For an accepted* ClosedSubstitutionBoun *carrier* $S = (T, D, V, K, R, Q, H, C, P, N)$*, every public read of the `substitute` operation at boundary* D *factors through the displayed `ClosedAt` classifier row* K *and returns the original term row. The payload row* V *is carried through the recursive operation route, but it has no access point on the closed-at-boundary slice.*

Proof. Read the classifier first. The row K says that every free index visible in the term row of T lies below the boundary D.

Now inspect the displayed substitution route Q. At a variable row, the equality branch for the boundary index cannot fire, and the decrement branch above the boundary cannot fire, because K supplies the strict below-boundary reading.

For compound rows, the route follows the constructor tree of T. The binder cases raise the depth and carry the payload row through the operation, but the classifier row continues to block any access point at the active boundary.

Transport through H and consumer reading through C preserve that same route. Thus the public substitute read is inert on the `ClosedAt` slice and does not export an unregistered free variable, a second payload access channel, or a host-level semantic substitution claim. □

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary substitute inertness route`

Theorem 538.4 (ClosedSubstitutionBoundary public nonescape)**.** *For an accepted* ClosedSubstitutionBoundary$^{\uparrow}$ *carrier* $S = (T, D, V, K, R, Q, H, C, P, N)$*, every public consumer of the packet reads only the displayed rows* $T, D, V, K, R, Q, H, C, P, N$ *and their registered* Cont *projections. In particular, the public surface cannot export arbitrary open-term substitution, evaluator subject reduction, host syntax quotient equality, or ambient metatheoretic truth.*

Proof. Start with the carrier coordinates of Definition 538.1. The only admitted rows are the term row, boundary row, payload row, `ClosedAt` classifier, `shift` and `substitute` routes, componentwise transport, continuation routes, provenance, and local naming row.

Now read a public consumer through C. Each continuation has a named source coordinate in S, so the consumer can project the displayed term, boundary, payload, classifier, operation route, transport route, provenance, or certificate, but no additional semantic object.

The excluded claims have no carrier coordinate and no continuation route. Thus arbitrary open substitution, evaluator subject reduction, quotient equality for host syntax, and ambient truth cannot be reached from an accepted packet. □

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary public nonescape`

Theorem 538.5 (ClosedSubstitutionBoundary closed-slice transport)**.** *Let* $S = (T, D, V, K, R, Q, H, C, P, N)$ *and* $S' = (T', D', V', K', R', Q', H', C', P', N')$ *be accepted* ClosedSubstitutionBoundary$^{\uparrow}$ *carriers. If the componentwise* hsame *row* H *transports* $T, D, V, K, R, Q, C, P, N$ *to* $T', D', V', K', R', Q', C', P', N'$ *and preserves the displayed* `ClosedAt` *row, then the inert* `substitute` *route on the closed slice transports from* Q *to* Q'*.*

Proof. First transport the boundary data. The hsame row identifies the term row, depth row, payload row, and `ClosedAt` classifier row with their primed counterparts while keeping the classifier attached to the same closed-slice reading.

Next apply Definition 538.3 to the source packet. The substitute route Q is inert precisely because each public read factors through K at boundary D.

Transport the operation row along H. Since the `ClosedAt` row is preserved, the same boundary-blocking argument applies after transport, and the primed route Q' remains inert on the transported closed slice. □

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary closed slice transport`

Theorem 538.6 (ClosedSubstitutionBoundary consumer exhaustion)**.** *The carrier of Definition 538.1, the NameCert obligations of Definition 538.2, the inert substitute route of Definition 538.3, the public nonescape theorem of Definition 538.4, and the closed-slice transport theorem of Definition 538.5 jointly give the consumer-exhaustion surface used by the sibling closed-term substitution boundary.*

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary consumer exhaustion`

Proof. The carrier and NameCert obligations provide the finite packet and its admissible obligations. They account for the displayed term, boundary, payload, classifier, operation, transport, continuation, provenance, and naming rows.

The inertness theorem fixes the only substitute route available to consumers on the closed slice. The transport theorem shows that componentwise hsame movement cannot change that route into an open-substitution surface.

Finally apply public nonescape. Since every consumer is exhausted by the displayed rows and their continuations, the sibling closed-term substitution boundary receives exactly the finite closed-substitution packet and no ambient substitution theorem, evaluator theorem, syntax quotient equality, or metatheoretic truth object. □

Theorem 538.7 (ClosedSubstitutionBoundary public consumer surface)**.** *Every public consumer of an accepted* ClosedSubstitutionBoundary$^{\uparrow}$ *carrier is exhausted by the finite rows*

$$T, D, V, K, R, Q, H, C, P, N.$$

The consumer may read the closed-at-boundary classifier, inert substitute route, componentwise transport, displayed continuation replay, provenance, and local naming row, but it cannot export open-term substitution, evaluator subject reduction, host syntax quotient equality, semantic preservation for arbitrary compiler passes, or bridge checking.

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.ClosedSubstitutionBoundary public consumer surface`

Proof. Start from Definition 538.6. It already identifies the carrier, NameCert obligations, inert substitute route, public nonescape clause, and closed-slice transport as the complete consumer surface.

The public consumer reads only through the displayed continuation row C. Hence every read has a named source among $T, D, V, K, R, Q, H, P, N$ and remains attached to the same local certificate.

Finally apply Definition 538.4. The excluded open-substitution, evaluator, quotient, compiler-preservation, and bridge-checking objects have no carrier coordinate and no continuation route. □

Theorem 538.8 (ClosedSubstitutionBoundary mature consumer package)**.** *The carrier of Definition 538.1, the NameCert obligations of Definition 538.2, the inertness route of Definition 538.3, the public nonescape theorem of Definition 538.4, the closed-slice transport theorem of Definition 538.5, the consumer-exhaustion theorem of Definition 538.6, and the public consumer surface of Definition 538.7 jointly form the mature consumer package for* ClosedSubstitutionBoundary$^{\uparrow}$.

Every consumer of that package reads the finite rows $T, D, V, K, R, Q, H, C, P, N$, their local NameCert *obligations, the inert closed-slice substitution route, and the public exhaustion boundary before it can treat the packet as a closed-substitution boundary surface. The package exports no arbitrary open-term substitution, evaluator subject reduction, host syntax quotient equality, semantic preservation for arbitrary compiler passes, or bridge checking.*

Proof. Start with the carrier and NameCert obligations. They display the term, depth, payload, classifier, shift, substitute, transport, continuation, provenance, and naming rows, and they bind all consumer reads to those rows.

Use the inertness route and closed-slice transport theorem for the substitution operation. They ensure that substitution at the displayed boundary remains an inert closed-slice read under componentwise hsame movement.

Finally apply public nonescape, consumer exhaustion, and the public consumer surface. These theorems exhaust the available continuation routes and exclude each ambient object named in the statement, so the combined package is mature on the paper-side consumer surface without changing the formal verification axis. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubstitutionBoundary}\uparrow}$.

Constructive story: A $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$ packet is generated from a MetaCIC `Term` row, a boundary depth row, a payload term row, a `ClosedAt` classifier row, `shift` and `substitute` operation rows, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionBoundary}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: The public surface binds the finite BHist packet, `ClosedAt` classifier, `shift` and `substitute` rows, componentwise hsame, displayed Cont, Pkg provenance, local NameCert, TasteGate marker, and public consumer exhaustion for the closed-substitution boundary slice.

Lean4 ✓ `BEDC.Derived.ClosedSubstitutionBoundaryUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedSubstitutionBoundary}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close arbitrary open-term substitution, evaluator subject reduction, host syntax quotient equality, semantic preservation for all compiler passes, or bridge checking.

Upgrade path: Higher closure requires checked public-surface formalization and bridge checking for the displayed closed-substitution boundary packet.

Chapter 539

A Concrete Naming Certificate for $\mathsf{CountableObservationSchedule}^{\uparrow}$

The $\mathsf{CountableObservationSchedule}^{\uparrow}$ surface realizes the Type IV finite-record schedule socket of section 9394.5 as a finite BHist packet. It records observed prefix rows, request windows, StreamName-facing source rows, transport, continuation, provenance, and a local naming certificate. The packet organizes countable observation names before $\mathsf{Real}^{\uparrow}$-level consumers; it does not turn observed regularity into an unobserved future law.

The schedule row is the finite-prefix and request-window source consumed by chapter 126; Real-facing budget consumers such as chapter 611 may read it only through the displayed StreamName source row and finite window ledger.

Definition 539.1 (CountableObservationSchedule carrier)**.** *A* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O, W, S, H, C, P, N)$$

with the following displayed rows:

1. *O is the observed prefix row, a finite list of already recorded observation entries indexed by their displayed finite positions;*
2. *W is the request-window row, recording the finite windows in which a downstream consumer asks for the next named observation, comparison, or bounded continuation step;*
3. *S is the StreamName source row, recording the finite source names by which the schedule is addressed without asserting that an unobserved tail is already present;*
4. *H records componentwise* hsame *transports for O, W, S, the continuation row, the package row, and the local naming row;*
5. *C records the displayed* Cont *routes that read prefix entries, expose request windows, hand off StreamName rows, and return only finite schedule data to downstream consumers;*
6. *P is the* Pkg *provenance over* BHist*, observed-prefix rows, request-window rows, StreamName source rows,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no completed countable stream, no proof that every future observation exists, no hidden causal-continuity principle, and no rule converting a successful prefix into a universal law.

Lean4 ✓ BEDC.Derived.CountableObservationScheduleUp.CountableObservationScheduleTasteGate
single carrier alignment

Theorem 539.2 (CountableObservationSchedule NameCert obligations)**.** *For the carrier of Definition 539.1, the local* $\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (O, W, S, H, C, P, N),$$

prefix exposure, request-window locality, StreamName source accountability, componentwise hsame *stability, displayed* Cont *routing,* Pkg *provenance, and non-escape for completed streams, unobserved observations, hidden persistence laws, and tail-totality claims.*

Proof. Carrier admission is projection from Definition 539.1. The packet exposes only the seven displayed rows O, W, S, H, C, P, N.

Prefix exposure is read from O, request locality from W, and StreamName accountability from S. These rows state the finite material available to a consumer and the finite name by which the consumer addresses it.

Transport, continuation, and provenance are then confined by H, C, P. The local certificate N can therefore export only the finite schedule data carried by those rows.

The non-escape clauses follow from the missing coordinates. Since Q has no row for a completed countable stream, a future observation, or a persistence law, no NameCert projection can supply such an object. □

Theorem 539.3 (CountableObservationSchedule prefix totality)**.** *Every accepted* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *packet is total on its displayed prefix: for each finite position listed in O, the packet contains exactly the observation row assigned to that position, together with the transport and continuation routes needed to read it as a schedule entry.*

This totality is prefix-local. It does not assert that positions outside O have observations, that every request window in W is eventually met by new data, or that the prefix determines a unique future tail.

Proof. Project the observed prefix row O from the carrier. By definition O is a finite list of already recorded entries indexed by displayed finite positions.

Use H to move the prefix row componentwise and C to read each displayed entry as a schedule entry. These rows preserve the prefix entries without adding further positions.

The scope restriction is forced by the carrier inventory. Positions not listed in O have no observation coordinate in Q, so prefix totality cannot be read as tail totality. □

Theorem 539.4 (CountableObservationSchedule window stability)**.** *For an accepted* CountableObservationSchedule *packet, each request window recorded in W remains the same finite request window under the componentwise* hsame *transports recorded in H. A transported consumer may ask for the same displayed finite prefix segment, next-name slot, or bounded continuation step, but it may not extend the window by transport alone.*

Proof. Project W and H from the carrier. The row W records the finite request windows, and H records componentwise transport for that row.

Replay the consumer route through C. The route exposes the transported request as the same finite window data, with provenance tracked by P and local naming tracked by N.

No window extension follows from transport. The carrier has no coordinate for enlarging a request window or for certifying an unobserved tail, so a transported window remains finite and displayed. □

Theorem 539.5 (CountableObservationSchedule StreamName handoff)**.** *For an accepted* CountableObservationSch *packet, the StreamName source row S is the only handoff surface by which downstream countable-name consumers address the schedule. The handoff consists of the displayed source name, the currently recorded prefix row, the finite request window, the transport row, the continuation route, the package provenance, and the local* $\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}$ *row.*

No consumer may replace the StreamName handoff with an unnamed countable choice function, an implicit infinite stream, or an empirical law asserting persistence beyond the displayed record.

Proof. Project the StreamName row S together with O, W, H, C, P, N. These are the only handoff-facing rows in the carrier.

The downstream consumer reads the name through C, transports the displayed rows through H, and records provenance through P. The local certificate N binds this finite handoff as the public schedule surface.

The excluded replacements have no carrier coordinate. An unnamed choice function, a completed infinite stream, and a persistence law would each add data beyond O, W, S, H, C, P, N, so they cannot be exported by the handoff. □

Theorem 539.6 (CountableObservationSchedule non-escape)**.** *The* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *packet does not discharge the Type IV causal-continuity socket. It may certify finite prefix rows, finite request windows, StreamName handoff, transport, continuation, provenance, and local naming, but it cannot certify that observed regularity persists into not-yet-observed history.*

Proof. The carrier of Definition 539.1 lists only finite schedule rows. Its prefix, window, source, transport, continuation, provenance, and naming data are all explicit coordinates of Q.

The Type IV socket of Definition 9394.17 asks for persistence from an observed record to unobserved history. That requested content is not among the coordinates of Q.

Therefore the packet can serve finite schedule consumers without closing the causal-continuity socket. Any claim of future persistence must remain outside the packet or be marked by the Type IV audit gate. □

Theorem 539.7 (CountableObservationSchedule finite request exhaustion)**.** *Every request in the W row of an accepted* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *packet is exhausted by displayed prefix entries in O and displayed* Cont *routes in C. A request may read a named* BHist *prefix entry, transport that entry through the corresponding* hsame *row in H, and record provenance through P and* $\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}$*; it has no route to an unnamed observation outside the displayed prefix.*

Proof. Project the request-window row W and the observed prefix row O from the carrier. The only entries available to satisfy a request are the finite entries already listed in O.

Replay the request through C. The continuation row reads the displayed prefix entry, while H transports that entry componentwise and P records the package provenance.

The local NameCert row names the same finite route. Since the carrier has no coordinate for an unnamed observation beyond O, every request in W is exhausted by displayed prefix entries and displayed continuation routes. □

Theorem 539.8 (CountableObservationSchedule prefix-window nonchoice)**.** *For an accepted* CountableObservation *packet, a transported request window remains a finite window over the named prefix data in O, W, S, H, C, P, N. The* hsame *transport in H cannot select an unnamed later observation, cannot choose a tail for the schedule, and cannot install a countable choice function for* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumers.*

Proof. Start with the window-stability theorem, Definition 539.4. It confines transport to the displayed finite request window.

Use the StreamName handoff of Definition 539.5. The downstream name is the displayed row S, routed through C and tracked by P and N.

The excluded selections have no carrier row. A later unnamed observation, a schedule tail, and a countable choice function would require data beyond O, W, S, H, C, P, N, so the transported window remains nonchoice and prefix-local. □

Theorem 539.9 (CountableObservationSchedule StreamName prefix consumer)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *packet factors through the displayed rows* O, W, S, H, C, P, N *before any* $\mathsf{Real}^{\uparrow}$*-facing schedule row is used. The consumer first reads the finite* BHist *prefix, finite request window, StreamName source row,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row; only that named prefix packet can be handed to a Real-facing consumer.*

Proof. Project the StreamName source row S and apply Definition 539.5. The handoff exposes only the displayed source name together with O, W, H, C, P, N.

Next apply finite request exhaustion from Definition 539.7. The consumer route reads a displayed prefix entry through C, transported through H, and recorded under P and N.

Only after those rows are assembled can a Real-facing schedule row read the packet. By Definition 539.6, that handoff still carries no completed stream, future observation law, or choice-like tail. □

Theorem 539.10 (CountableObservationSchedule TasteGate surface)**.** *Any later TasteGate promotion for* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *must certify the single carrier*

$$Q = (O, W, S, H, C, P, N)$$

and its finite prefix, request-window, $\mathsf{StreamName}^{\uparrow}$ *handoff,* hsame *transport,* Cont *route,* Pkg *provenance, and* NameCert *rows. No proof may duplicate the carrier as a proof step or replace a missing row with a completed stream, future observation law, causal-continuity assertion, or choice-like tail.*

Proof. Start with the carrier and NameCert obligation theorem. They identify $Q = (O, W, S, H, C, P, N)$ as the unique packet surface and bind prefix exposure, request-window locality, StreamName accountability, transport, continuation, provenance, and non-escape to that packet.

Next apply Definition 539.3, Definition 539.4, and Definition 539.5. These provide the finite prefix, stable request window, and named StreamName handoff without adding another carrier.

Finally apply Definition 539.6, Definition 539.7, Definition 539.8, and Definition 539.9. The consumer route is exhausted by the displayed rows, so the TasteGate surface cannot use a completed stream, future observation law, causal-continuity assertion, or choice-like tail as a replacement row. □

Theorem 539.11 (CountableObservationSchedule obligation closure package)**.** *The carrier of Definition 539.1, the NameCert obligations of Definition 539.2, prefix totality from Definition 539.3, window stability from Definition 539.4, StreamName handoff from Definition 539.5, non-escape from Definition 539.6, finite request exhaustion from Definition 539.7, prefix-window nonchoice from Definition 539.8, StreamName prefix consumption from Definition 539.9, and the TasteGate surface of Definition 539.10 form the complete paper-side obligation package over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The package is finite and local: it records exactly the displayed prefix rows, request windows, StreamName source rows, transports, continuations, provenance, and local naming row of $Q = (O, W, S, H, C, P, N)$.

Proof. The carrier theorem fixes the only admissible packet as $Q = (O, W, S, H, C, P, N)$, and the NameCert obligation theorem identifies carrier admission, prefix exposure, request locality, StreamName accountability, transport, continuation, provenance, and local naming as the public certificate rows.

Apply prefix totality, window stability, and StreamName handoff in file order. These theorems show that every accepted consumer first reads the finite prefix, the transported request window, and the displayed StreamName source row through the same packet.

Next apply non-escape, finite request exhaustion, prefix-window nonchoice, and StreamName prefix consumption. These bind each finite request to displayed prefix entries, forbid choice-like tail selection, and force any $\mathsf{Real}^{\uparrow}$-facing schedule read through the named prefix packet.

Finally, the TasteGate surface theorem packages those rows as the admissible promotion surface. Because the carrier has no coordinate for quotient stream equality, host real equality, ambient completeness, hidden choice, or an unbounded schedule, none of those objects is introduced by the obligation package. □

Theorem 539.12 (CountableObservationSchedule Real window boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing schedule row exported by an accepted* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *packet factors through the finite rows* O, W, S, H, C, P, N*. The exported row may read the displayed observations, request window, StreamName source, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, but it cannot assert tail totality, future observation existence, a completed countable stream, or a* $\mathsf{Real}^{\uparrow}$ *limit outside the displayed finite window.*

Proof. Project the carrier $Q = (O, W, S, H, C, P, N)$. The only observation data available to a Real-facing row is the finite prefix row O and the finite request window W.

Apply Definition 539.4 and Definition 539.9. They force transported Real-facing reads to keep the same finite request window and to pass through the displayed StreamName source row before a schedule row is consumed.

Finally apply Definition 539.6 and Definition 539.8. Since no carrier coordinate names a tail-total stream, future observation law, or completed $\mathsf{Real}^{\uparrow}$ limit, the Real-facing boundary is exactly the finite window. □

Theorem 539.13 (CountableObservationSchedule TasteGate ground compiler alignment)**.** *The TasteGate-facing ground compiler for* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *is aligned with the single carrier* $Q = (O, W, S, H, C, P, N)$ *and with the displayed event-flow and no-hidden-input rows. Its admissible surface is the finite prefix, request-window, StreamName handoff, transport, continuation, provenance, and local naming packet; it is not hidden causal-continuity data, an unobserved future law, a second schedule carrier, or a completed stream.*

Proof. Begin with the TasteGate surface of Definition 539.10. It fixes the admissible promotion surface as the single carrier $Q = (O, W, S, H, C, P, N)$.

The event-flow rows are then read through prefix totality, finite request exhaustion, and StreamName handoff. These expose exactly the displayed prefix entries, request windows, and source-name route, with transports in H and continuations in C.

No-hidden-input alignment follows from non-escape. Because the carrier has no coordinate for causal-continuity data, an unobserved future law, a second carrier, or a completed stream, the ground compiler can only certify the displayed finite packet. □

Theorem 539.14 (CountableObservationSchedule ground-compiler event-flow lock)**.** *Any TasteGate promotion for the finite* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *surface must route the rows*

$$O, W, S, H, C, P, N$$

through the displayed ground-compiler event-flow and no-hidden-input rows of Definition 539.13. It cannot replace those rows by hidden causal-continuity data, a future-law row, a completed stream, or a second carrier.

Proof. Start with the carrier and the TasteGate surface. Definition 539.1 fixes the single packet $Q = (O, W, S, H, C, P, N)$, while Definition 539.10 makes that packet the admissible promotion surface.

Next read the event-flow rows through prefix totality, finite request exhaustion, and StreamName handoff. These theorems expose only displayed prefix entries, finite request windows, and the source-name route, with transport in H and continuation replay in C.

The ground-compiler alignment then locks the same rows to the event-flow and no-hidden-input discipline. By Definition 539.13, a compiler-facing consumer may use the finite prefix, request-window, StreamName handoff, transport, continuation, provenance, and local naming packet, and no hidden input row.

Finally apply StreamName prefix consumption and non-escape. The route through Definition 539.9 and Definition 539.6 gives no coordinate for hidden causal-continuity data, a future-law row, a completed stream, or another carrier, so those replacements are outside the TasteGate-facing event-flow lock. □

Theorem 539.15 (CountableObservationSchedule scoped obligation package)**.** *The carrier of Definition 539.1, the NameCert obligations of Definition 539.2, prefix totality from Definition 539.3, window stability from Definition 539.4, StreamName handoff from Definition 539.5, non-escape from Definition 539.6, finite request exhaustion from Definition 539.7, prefix-window nonchoice from Definition 539.8, StreamName prefix consumption from Definition 539.9, the Real window boundary of Definition 539.12, and the TasteGate ground compiler alignment of Definition 539.13 assemble the scoped* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *obligation package over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The package is scoped to finite displayed windows. It does not close tail totality, hidden causal continuity, completed countable streams, or unbounded $\mathsf{Real}^{\uparrow}$ *limit data.*

Proof. The carrier and NameCert obligation theorems fix $Q = (O, W, S, H, C, P, N)$ as the only public packet. They bind carrier admission, finite prefix exposure, request locality, StreamName accountability, transport, continuation, provenance, and local naming to those rows.

Apply prefix totality, window stability, StreamName handoff, finite request exhaustion, prefix-window nonchoice, and StreamName prefix consumption. These results show that each consumer read is exhausted by displayed prefix entries and finite request windows transported through H, C, P, N.

The Real window boundary adds the Real-facing scope condition. It prevents the finite schedule row from asserting tail totality, a completed stream, or a hidden limit outside the displayed request window.

The TasteGate ground compiler alignment supplies the promotion boundary. It ties the same carrier to displayed event-flow and no-hidden-input rows, so the package contains no hidden causal-continuity data and no second schedule carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{CountableObservationSchedule}^{\uparrow}$ packet is generated from observed prefix rows, request windows, StreamName source rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 539.1, Definition 539.2, Definition 539.3, Definition 539.4, Definition 539.5, Definition 539.6, Definition 539.7, Definition 539.8, Definition 539.9, Definition 539.10 with
Lean4 ✓ `BEDC.Derived.CountableObservationScheduleUp.taste gate`
, Definition 539.12, Definition 539.13, Definition 539.14, and Definition 539.15 bind the finite carrier, prefix totality, window stability, StreamName handoff, Type IV non-escape boundary, finite request exhaustion, nonchoice window transport, prefix consumer factorization, TasteGate instance surface, Real window boundary, ground-compiler alignment, event-flow lock, and scoped package as one displayed finite schedule surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableObservationSchedule}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CountableObservationScheduleUp.CountableObservationScheduleTasteGate\_single\_carrier\_`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close completed countable streams, future observation existence, causal continuity, Real limits outside displayed windows, or a public standard bridge.

Upgrade path: Public closure requires a public export theorem consumed by $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ schedules together with a later bridge route.

Chapter 540

ClosedTermSubstitutionCompiler NameCert Construction

The $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ surface packages the MetaCIC closed-term substitution boundary as a finite compiler-facing naming certificate. Its carrier is not a full evaluator correctness theorem. It records the concrete rows by which ClosedAt, shift, substitute, substitute_closed, BHist, Cont, Pkg, and NameCert form the local boundary used by self-compilation consumers.

The compiler-facing packet consumes the finite closed-term boundary of chapter 527 before it exposes the substitute_closed and shift_closed witness rows. This dependency is boundary reuse: it records that the compiler schedule reads the closed substitution and shift surface through the registered packet, not that the packet proves evaluator correctness or subject reduction.

Definition 540.1 (ClosedTermSubstitutionCompiler carrier)**.** *A* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, D, V, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC term-generator row, recording the closed constructor surface for variables, applications, binders, and universes;*
2. *D is the* ClosedAt *boundary row, recording the depth classifier used to admit closed or context-closed source terms;*
3. *V is the operation row containing the syntactic* shift *and* substitute *traversals;*
4. *F is the fixed-boundary witness row recording* shift_closed *and* substitute_closed *as inertness facts on the classified slice;*
5. *H records componentwise* hsame *transports for T, D, V, F, the continuation row, the package row, and the local naming row;*
6. *C records the displayed* Cont *routes by which a compiler consumer reads the closedness boundary, applies the syntactic operation, and returns to the fixed-boundary witness;*
7. *P is the* Pkg *provenance over MetaCIC terms,* ClosedAt*,* shift*,* substitute*,* substitute_closed*,* BHist*,* Cont*, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}$ *row.*

The carrier contains no subject-reduction theorem for open evaluation, no semantic substitution theorem for arbitrary host syntax, no truth predicate for CIC, and no unrecorded binder budget.

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket carrier admission`

Theorem 540.2 (ClosedTermSubstitutionCompiler boundary)**.** *Every accepted* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *packet exposes the closed-term substitution boundary as the finite route*

$$\mathsf{ClosedAt}(d, t) \longrightarrow \mathsf{substitute}(d, v, t) = t.$$

The same packet also exposes the companion shift boundary

$$\mathsf{ClosedAt}(n, t) \longrightarrow \mathsf{shift}(n, 1, t) = t.$$

Both routes are read through the displayed BHist*,* Cont*,* Pkg*, and* NameCert *rows, and neither route may be replaced by semantic evaluation correctness, a host binder convention, or an ambient compiler theorem.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket boundary`

Proof. Project the carrier $K = (T, D, V, F, H, C, P, N)$. The term row T supplies the authorized MetaCIC constructors, the boundary row D supplies the displayed ClosedAt classifier, and the operation row V supplies only the syntactic traversals shift and substitute.

Read the witness row F through the continuation row C. The substitution witness is the fixed-boundary equality $\mathsf{substitute}(d, v, t) = t$ under $\mathsf{ClosedAt}(d, t)$, and the shift witness is the matching equality at the shift cutoff. The arbitrary value v is not consumed because the classifier shows that the target variable is absent at the boundary.

The non-escape clause is carrier exactness. Since no coordinate stores evaluator correctness, host-level binder semantics, or a global compiler theorem, a consumer can extract only the displayed closed-boundary route. □

Theorem 540.3 (ClosedTermSubstitutionCompiler NameCert obligations)**.** *For the carrier of Definition 540.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for* $K = (T, D, V, F, H, C, P, N)$*;*
2. *classifier transport for the* ClosedAt *row and the operation-witness rows under componentwise* hsame*;*
3. *substitution-boundary exactness, requiring every accepted substitution read to cite the displayed* substitute_closed *witness before exposing a compiler-facing boundary claim;*
4. *shift-boundary compatibility, requiring binder-depth movement to cite the displayed* shift_closed *witness before a deeper-context read is accepted;*
5. *non-escape for open-term semantics, subject reduction, host CIC truth, unregistered variables, and any compiler correctness theorem not stored in the packet.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket semantic name certificate`

Proof. Carrier admission is projection from Definition 540.1. The packet lists the term generator, closedness classifier, operation row, fixed-boundary witnesses, transports, continuation routes, package provenance, and local certificate row.

Transport is componentwise. The rows T, D, V, F move only through the displayed hsame entries, and the continuation row replays the same reads after transport.

For exactness, inspect a substitution consumer. The route must read the ClosedAt row and then the substitute_closed witness before it may expose the compiler-facing equality. The corresponding shift consumer reads the same boundary classifier and the shift_closed witness before using binder-depth movement.

The excluded principles have no carrier coordinate. Thus the local NameCert admits only the five displayed obligations and cannot project an unrecorded semantic theorem. □

Theorem 540.4 (ClosedTermSubstitutionCompiler self-compile route)**.** *In any compiler-facing read of an accepted* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *packet, the self-compilation substitution step passes through the schedule*

$$\mathsf{ClosedAt} \longrightarrow \mathsf{substitute} \longrightarrow \mathsf{substitute_closed} \longrightarrow \mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}.$$

The route keeps the source term inside the registered boundary ledger. It may be composed with the displayed shift_closed *row for binder movement, but it does not close evaluator soundness, subject reduction, or total self-compilation.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket self compile route`

Proof. Begin with the compiler consumer's closed source row. The carrier exposes this row only through D, so the consumer first reads the displayed ClosedAt boundary.

Next read the operation row V and the witness row F. The syntactic operation is substitute, and the accepted boundary witness is substitute_closed, so the attempted substitution returns the same registered term at the boundary.

Finally repack the route through C, P, N. The continuation, provenance, and local naming rows record a self-compilation boundary step, while the absence of evaluator-soundness and subject-reduction coordinates fixes the stated scope. □

Theorem 540.5 (ClosedTermSubstitutionCompiler classifier transport)**.** *For accepted* ClosedTermSubstitutionCom *packets, componentwise* hsame *transport preserves the displayed* ClosedAt *row, the* shift *row, the* substitute *row, the witness row containing* shift_closed *and* substitute_closed*, the self-compile route, provenance, and the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket classifier transport`

Proof. Project the carrier of Definition 540.1. The rows T, D, V, F, H, C, P, N expose the closedness classifier, syntactic operation rows, fixed-boundary witnesses, continuation route, provenance, and naming certificate.

Use the transport obligation in Definition 540.3. Its classifier clause moves D and the operation-witness rows through the displayed entries of H.

The self-compile route of Definition 540.4 is then replayed through C, while P and N keep the transported read inside the same local certificate surface. □

Theorem 540.6 (ClosedTermSubstitutionCompiler shift-substitute compatibility)**.** *In every accepted* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *packet, the displayed* shift_closed *witness and the displayed* substitute_closed *witness share the same closed-boundary classifier before any compiler-facing read may consume either equality.*

Proof. Read the carrier row D before the witness row F. By Definition 540.2, the substitution and shift routes are both exposed as closed-boundary routes rather than as evaluator facts.

The NameCert obligations of Definition 540.3 require substitution-boundary exactness and shift-boundary compatibility. Thus both witnesses are admitted only after the same displayed ClosedAt classifier has been read.

The self-compile route of Definition 540.4 may compose with the shift row, but the composition stays inside the shared classifier boundary. □

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket shift substitute compatibility`

Theorem 540.7 (ClosedTermSubstitutionCompiler closed-composition handoff)**.** *Every compiler-facing read that consumes the checked closed composition slice* substitute_compose_closed_both *must first display the local* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *boundary ledger: the* ClosedAt *rows for both closed inputs, the* substitute_closed *witness row, the* shift_closed *witness row, and the local* $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}$ *packet. The handoff is a closed-fragment route through* $K = (T, D, V, F, H, C, P, N)$ *and not an unrestricted substitute-substitute commutation theorem, subject-reduction theorem, or evaluator-soundness claim.*

Proof. Project the compiler carrier $K = (T, D, V, F, H, C, P, N)$. The boundary row D is the only source of ClosedAt data, and the witness row F is the only source of substitute_closed and shift_closed data.

Apply Definition 540.9. A compiler-facing substitution or shift equality must read the displayed closedness classifier, the corresponding syntactic operation row, and the corresponding fixed-boundary witness before any equality is exported.

Next apply Definition 540.6. The substitution and shift witnesses share the same closed-boundary classifier, so a closed composition read may compose those witnesses only after both closed inputs have been admitted through D.

Finally route the read through Definition 540.3. The local NameCert packet stores the closed-fragment boundary and excludes evaluator soundness, subject reduction, open-term substitution semantics, and unrestricted substitution calculus. Therefore the checked composition slice is handed off as a closed compiler boundary only. □

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket closed composition handoff`

Theorem 540.8 (ClosedTermSubstitutionCompiler public non-escape)**.** *No public consumer of an accepted* $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ *packet can read evaluator soundness, subject reduction, host CIC truth, or open-term semantics from the packet. Public reads are limited to the displayed* ClosedAt*,* shift*,* substitute*, witness, route, provenance, and* NameCert *rows.*

Proof. Inspect the carrier of Definition 540.1. It contains rows for the MetaCIC term generator, closedness classifier, syntactic operations, fixed-boundary witnesses, transports, routes, provenance, and local naming.

Definition 540.2 and Definition 540.4 describe exactly the public routes: closedness, syntactic operation, fixed-boundary witness, and local NameCert.

The excluded principles are absent from the carrier and are rejected by the non-escape obligation in Definition 540.3. Hence no public projection can recover evaluator soundness, subject reduction, host CIC truth, or open-term semantics. □

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket public nonescape`

Theorem 540.9 (ClosedTermSubstitutionCompiler boundary-ledger exactness)**.** *In every accepted* ClosedTermSubstitutionCompiler$^{\uparrow}$ *packet, each compiler-facing substitution read and shift read passes through the displayed boundary ledger*

$$\mathsf{ClosedAt},\ \mathsf{shift},\ \mathsf{substitute},\ \mathsf{shift_closed},\ \mathsf{substitute_closed}.$$

No read may expose a substitution equality, a shift equality, or a self-compilation boundary step unless it first cites the ClosedAt *classifier row and then the corresponding operation and witness row stored in the packet.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket boundary ledger exactness`

Proof. Project the carrier $K = (T, D, V, F, H, C, P, N)$ of Definition 540.1. The row D is the only displayed ClosedAt source, V is the only operation row for shift and substitute, and F is the only row containing shift_closed and substitute_closed.

For a substitution read, the route described in Definition 540.4 must enter through D, read substitute from V, and consume substitute_closed from F before reaching the local NameCert row.

For a shift read, Definition 540.6 forces the same closed-boundary classifier before shift and shift_closed are consumed. The continuation and package rows then record only that displayed route.

Since no other coordinate stores substitution or shift evidence, any compiler-facing equality that skips this ledger is outside the accepted packet. □

Theorem 540.10 (ClosedTermSubstitutionCompiler binder-budget non-escape)**.** *An accepted* ClosedTermSubstitut *packet does not export binder-budget semantics, open-term substitution semantics, evaluator soundness, subject reduction, host CIC truth, or any unregistered variable convention. Public reads are confined to the displayed* ClosedAt*,* shift*,* substitute*,* shift_closed*,* substitute_closed*,* BHist*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket binder budget nonescape`

Proof. Inspect the carrier of Definition 540.1. It names the term generator, closedness classifier, syntactic operation row, fixed-boundary witness row, transports, routes, provenance, and local certificate row.

Binder-budget semantics would require a separate row describing how budgets are allocated or consumed across arbitrary open binders. Open-term substitution semantics would require a row for variables not admitted by the displayed ClosedAt boundary. Neither row occurs in K.

Definition 540.8 already excludes evaluator soundness, subject reduction, host CIC truth, and open-term semantics. The boundary-ledger exactness of Definition 540.9 further confines every public read to the displayed closed-term ledger.

Thus a consumer may use the packet as a closed substitution compiler boundary, but it cannot project binder-budget or open-term semantics from the same certificate. □

Theorem 540.11 (ClosedTermSubstitutionCompiler obligation-closure package)**.** *The* ClosedTermSubstitutionCom *chapter reaches obligation closure at the paper surface: its carrier admission, classifier transport, boundary-ledger exactness, shift compatibility, and non-escape obligations are all stated as finite* NameCert *rows over the same packet* $K = (T, D, V, F, H, C, P, N)$*.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket obligation closure package`

Proof. Carrier admission is Definition 540.1. It fixes the eight displayed rows and excludes evaluator correctness, host binder conventions, and unrecorded binder budgets from the carrier.

Classifier transport is Definition 540.5. The theorem records componentwise hsame preservation of the ClosedAt row, operation rows, witness row, route, provenance, and local NameCert row.

Boundary-ledger exactness is Definition 540.9. Shift compatibility is Definition 540.6. Together they force substitution and shift reads through the same displayed closedness classifier and their corresponding witness rows.

Non-escape is supplied by Definition 540.10 and Definition 540.8. These theorems identify the absent semantics that no public projection can recover. The existing Lean TasteGate marker on Definition 540.2 supplies the current checked alignment witness, while the formal status of this package remains unadvanced. □

Theorem 540.12 (ClosedTermSubstitutionCompiler bridge route)**.** *Every compiler-facing read of an accepted* ClosedTermSubstitutionCompiler$^{\uparrow}$ *packet bridges the concrete* ClosedAt*,* shift*,* substitute*,* shift_closed*, and* substitute_closed *rows to the public* ClosedTermSubstitutionCompiler$^{\uparrow}$ *interface through the displayed* BHist*,* Cont*,* Pkg*, and* NameCert *rows. The bridge route excludes evaluator correctness, subject reduction, host CIC truth, and unregistered binder budgets.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket bridge route`

Proof. Project the packet $K = (T, D, V, F, H, C, P, N)$ from Definition 540.1. The concrete compiler rows are exactly D for ClosedAt, V for shift and substitute, and F for shift_closed and substitute_closed.

Apply the boundary-ledger exactness theorem to force every compiler-facing substitution or shift read through those concrete rows. Then use the public non-escape and binder-budget non-escape theorems to restrict the exported interface to the displayed BHist, Cont, Pkg, and local NameCert route.

The bridge therefore reaches the public ClosedTermSubstitutionCompiler$^{\uparrow}$ surface only through the named packet rows. Evaluator correctness, subject reduction, host CIC truth, and unregistered binder budgets have no coordinate in K, so they cannot be recovered by the bridge. □

Theorem 540.13 (ClosedTermSubstitutionCompiler mature route)**.** *Closed substitution compiler consumers are exhausted by the closed-term carrier, substitution replay, compiler handoff, and refusal rows already displayed by the bridge route. Every accepted* ClosedTermSubstitutionCompiler$^{\uparrow}$ *packet therefore reaches the mature paper surface through* $K = (T, D, V, F, H, C, P, N)$*: the carrier row fixes the closed-term generator and classifier, the replay rows expose* shift *and* substitute *only through* ClosedAt *witnesses, the compiler handoff reads the public interface through* BHist*,* Cont*,* Pkg*, and* NameCert*, and the refusal rows exclude evaluator correctness, subject reduction, open-term substitution semantics, host CIC truth, and unregistered binder budgets.*

Lean4 ✓ `BEDC.Derived.ClosedTermSubstitutionCompilerUp.ClosedTermSubstitutionCompilerPacket mature route`

Proof. Project the carrier of Definition 540.1. It supplies exactly the rows T, D, V, F, H, C, P, N, so every consumer begins from the closed-term generator, the ClosedAt classifier, the operation row, and the fixed-boundary witnesses.

Apply the boundary theorem, self-compile route, classifier transport, and boundary-ledger exactness. These results force substitution replay and shift replay through the same displayed closed-boundary rows before any compiler-facing read can export a public interface point.

Finally apply the bridge route together with public non-escape and binder-budget non-escape. The bridge exposes the public $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ route only through BHist, Cont, Pkg, and NameCert, while the refusal theorems exhaust every absent consumer coordinate. No mature consumer remains outside the closed-term carrier, substitution replay, compiler handoff, and refusal rows already displayed in this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$ packet is generated from MetaCIC term constructors, the ClosedAt classifier, the shift and substitute operation rows, the shift_closed and substitute_closed witness rows, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridged scope records the finite carrier, closed-boundary substitution and shift routes, classifier transport surface, NameCert obligations, self-compilation boundary handoff, boundary-ledger exactness, shift compatibility, public and binder-budget non-escape, the obligation-closure package, and the bridge route from concrete compiler rows to the public interface over the same packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close evaluator soundness, subject reduction, open-term substitution semantics, host CIC truth, total self-compilation, or a checked Lean certificate for the whole packet.

Upgrade path: A higher theory grade requires mature examples and completeness of consumer coverage for the same compiler-facing bridge route, while checked Lean formalization remains separate.

Chapter 541

A Concrete Naming Certificate for $\mathsf{AuthorizedRecursor}^{\uparrow}$

The $\mathsf{AuthorizedRecursor}^{\uparrow}$ interface records legal eliminator calls as finite naming packets. Its role is not to add a recursor beyond CIC. It names the BEDC surface already present when the kernel accepts an inductive constructor signature, checks a motive, checks one branch package for that signature, and emits an output whose computation cannot escape the displayed constructor spine.

The recursor packet consumes the accepted constructor-signature boundary of chapter 915, where constructor signature acceptance supplies Σ, and exports its generated output through the generator seal of chapter 553 or the ledger of chapter 555, where generator-induction sealing supplies the downstream non-escape route for O.

Definition 541.1 (AuthorizedRecursor carrier)**.** *An* $\mathsf{AuthorizedRecursor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, \Sigma, M, B, O, A, H, C, P, N)$$

with the following visible coordinates:

1. *I is the inductive-family name whose kernel-accepted constructor signature is being consumed;*
2. *Σ is the finite constructor-signature row listing the accepted constructors and their recursive argument positions;*
3. *M is the motive row checked over the closed carrier generated by Σ;*
4. *B is the branch-package row, with one displayed branch for each constructor in Σ;*
5. *O is the output row produced by the legal eliminator call over the selected input object;*
6. *A is the authorization ledger stating that the recursor call uses exactly the accepted signature, motive, branches, and structural recursive subcalls;*
7. *H stores displayed* hsame *transports for the signature, motive, branch package, output row, authorization ledger, and local naming row;*
8. *C stores the finite* Cont *routes by which consumers read signature rows, motive rows, branch rows, output rows, and rejected non-structural attempts;*
9. *P is the* Pkg *provenance over the inductive declaration, constructor signature, motive, branch package, output row,* hsame *transports,* Cont *routes, and audit ledger;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AuthorizedRecursor}^{\uparrow}}$ *row.*

The carrier exports only these displayed rows. It contains no extra induction principle, no branch outside Σ, no negative self-use hidden behind the motive, and no recursive call whose descent is not recorded in A.

Theorem 541.2 (AuthorizedRecursor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AuthorizedRecursor}^{\uparrow}}$ *surface for Definition 541.1 consists of five finite obligations: carrier admission for R; constructor-signature exactness for Σ; motive-and-branch coverage for M and B; authorization-ledger exactness for A; and non-escape of negative self-use, missing branches, unrecorded recursion, quotient equality, selected witnesses, and ambient classical principles.*

Proof. Carrier admission is projection of the displayed packet R. The inductive name, signature row, motive row, branch package, output row, authorization ledger, transports, routes, provenance, and local certificate are all explicit coordinates.

Signature exactness and coverage are read from Σ, M, and B. Since B has one branch row for each constructor row in Σ, every legal eliminator case is paid for by a visible branch and no additional branch is exported.

The authorization and boundary clauses are read from A and P. A recursive subcall is admitted only when A records its structural descent through the constructor spine; every missing branch, negative self-use, unrecorded recursion, quotient equality, selected witness, or classical principle has no carrier coordinate and is therefore recorded only as a boundary attempt. □

Theorem 541.3 (AuthorizedRecursor generator ledger exactness)**.** *Every consumer read of an accepted* $\mathsf{AuthorizedRecursor}^{\uparrow}$ *packet factors through the displayed rows $I, \Sigma, M, B, O, A, H, C, P, N$. The generator ledger is exact: a read of the recursor output O determines the constructor signature row Σ, the motive row M, the branch-package row B, and the authorization row A used to produce it, and no consumer can replace them by a different hidden eliminator call.*

Proof. Begin with the carrier of Definition 541.1. The output row O is stored together with the signature, motive, branch package, and authorization ledger that generated it.

Transport through H preserves those same rows, and continuation through C exposes them as the only legal consumer routes. Thus a consumer that reads O also reads the finite provenance of the eliminator call.

If a consumer asks for a different signature, motive, branch package, or recursive-call ledger, the packet has no coordinate that can supply it. The request is outside the displayed route and cannot be exported as a generated recursor output. □

Theorem 541.4 (AuthorizedRecursor non-escape boundary)**.** *An accepted* $\mathsf{AuthorizedRecursor}^{\uparrow}$ *packet does not export any object obtained from a constructor outside Σ, a branch not listed in B, a motive not named by M, a recursive call not justified by A, or an external principle such as quotient soundness, propositional extensionality, choice, or ambient classical completeness.*

Proof. The constructor boundary is finite. Every admissible constructor read is one of the rows listed in Σ, and every admissible branch read is the corresponding row in B.

The motive and recursion boundaries are also finite. The motive row M is the only motive exported by the packet, and the authorization ledger A is the only place where recursive subcalls can be justified.

The provenance row P records the absence of external principles. Since none of quotient soundness, propositional extensionality, choice, or ambient completeness is a coordinate of R, no downstream route can receive such a principle from this certificate. □

Theorem 541.5 (AuthorizedRecursor downstream handoff)**.** *Every downstream BEDC consumer of an* $\mathsf{AuthorizedRecursor}^{\uparrow}$ *output receives a finite handoff packet consisting of the output row* O*, its constructor-signature row* Σ*, motive row* M*, branch-package row* B*, authorization ledger* A*, transport row* H*, continuation route* C*, package row* P*, and local naming row* N*. The handoff is sufficient for later TasteGate or kernel-recursion certificates to inspect the legal eliminator call without reopening the host declaration.*

Lean4 ✓ `BEDC.Derived.AuthorizedRecursorUp.AuthorizedRecursorTasteGate single carrier alignment`

Proof. Project the handoff rows from the carrier R. The output row is not transmitted alone; it is transmitted with the exact signature, motive, branch, authorization, transport, route, package, and naming rows that make the eliminator call visible.

TasteGate-style consumers inspect the same finite rows to decide whether the call stays inside the accepted constructor signature. Kernel-recursion consumers inspect the authorization ledger to replay the structural descent boundary.

No downstream consumer needs a fresh host-side induction principle. The handoff carries the already named BEDC certificate surface, so later packets can consume the recursor output through finite NameCert rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuthorizedRecursor}^{\uparrow}}$.

Constructive story: An $\mathsf{AuthorizedRecursor}^{\uparrow}$ packet is generated from an inductive-family name, accepted constructor-signature row, checked motive row, branch package, output row, authorization ledger, displayed hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuthorizedRecursor}^{\uparrow}}$, seed)

Scope closed: The seed scope records the carrier, five NameCert obligations, generator-ledger exactness, non-escape boundary, and downstream handoff for legal eliminator calls.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AuthorizedRecursor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean kernel soundness, all inductive-family positivity checks, all termination checking, TasteGate verification, kernel-recursion bridge checking, or any external classical principle.

Upgrade path: Obligation closure requires a checked finite carrier, signature coverage theorem, motive-and-branch ledger theorem, structural recursion ledger theorem, and downstream handoff theorem for the same packet.

Chapter 542

A Concrete Naming Certificate for $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$

The $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$ surface turns a reported seven-kind Gap socket into a finite ledger packet that can be consumed by NameCert readers without treating the vision taxonomy itself as a certificate. It records the socket site, requested supply, seven-kind classifier, audit gate, transport, continuation, provenance, and local name. The packet is a ledger for the request boundary; it does not provide the requested supply.

The ledger consumes the audit boundary of chapter 526 and the forbidden-ancestry refusal surface of chapter 622 only as finite gate and refusal rows.

Definition 542.1 (TranscendentalSupplyLedger carrier)**.** *A* $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Lambda = (S, R, T, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the socket-site row naming the concrete paper or formal site at which a* Gap *request is reported;*
2. *R is the requested-supply row, stated as the exact missing discharge the site would consume;*
3. *T is the seven-kind classifier row, choosing one of the socket kinds of section 9394.11;*
4. *G is the audit-gate row that prevents the request from being read as an internal BEDC theorem;*
5. *H records componentwise* hsame *transport for the site, requested supply, classifier, audit gate, provenance, and local naming rows;*
6. *C records the displayed* Cont *routes through which consumers read the site row, classify the socket, inspect the requested supply, and return the audit gate;*
7. *P is the* Pkg *provenance over* BHist, Gap, *the seven socket-kind rows, requested supply rows, audit-gate rows,* hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyLedger}^{\uparrow}}$ *row.*

The carrier exports no row for an unrestricted induction rule, hidden witness, quotient equality, empirical persistence law, shared first-person history, inscription-source explanation, or truth-total self-validator. Each of those shapes may be named in R and classified by T, but none is discharged by the packet.

Lean4 ✓ BEDC.Derived.TranscendentalSupplyLedgerUp.TranscendentalSupplyLedgerTasteGate single carrier alignment

Theorem 542.2 (TranscendentalSupplyLedger taxonomy exhaustion)**.** *For every accepted* TranscendentalSupplyLe *carrier*

$$\Lambda = (S, R, T, G, H, C, P, N),$$

the classifier row T exhausts the socket taxonomy by routing the site row S and requested-supply row R through exactly one of the seven displayed kinds: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure. Every downstream taxonomy read factors through

$$S \to R \to T \to G \to H \to C \to P \to N$$

and no consumer receives an eighth classifier coordinate or a discharge of the request named in R.

Proof. Project the carrier of Definition 542.1. The only classifier-facing row is T, and the only request-facing row is R at the socket site S.

Read T against the seven rows of section 9394.11. Since T is a displayed finite row, accepting the packet selects one of those seven cases and provides no further case coordinate.

Now route the accepted row through G, H, C, P, N. The audit gate records why the requested supply remains outside the closed substrate, while transport, continuation, provenance, and local naming preserve the same finite packet for downstream readers.

Because the carrier contains no row for the requested discharge itself, taxonomy exhaustion is classificatory only. The packet can say which socket kind has appeared, but it cannot provide the witness, rule, equality token, law, recognition, inscription source, or self-reference closure named by the request. □

Lean4 ✓ BEDC.Derived.TranscendentalSupplyLedgerUp.TranscendentalSupplyLedger taxonomy exhaustion
Lean4 ✓ BEDC.Derived.TranscendentalSupplyLedgerUp.TranscendentalSupplyLedgerTaxonomyExhaustion

Theorem 542.3 (TranscendentalSupplyLedger NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{TranscendentalSuppl}}$ *surface consists exactly of the carrier row of Definition 542.1, the socket-site row S, the requested-supply row R, the seven-kind classifier row T, the audit-gate row G, componentwise* hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and local naming row N.*

Thus a consumer may accept the ledger only by reading the finite packet

$$(S, R, T, G, H, C, P, N)$$

as one NameCert boundary. It may not replace the audit gate with an internal theorem, replace the classifier with a hidden source object, or read the requested supply as available substrate data.

Proof. Carrier exposure comes first. The packet in Definition 542.1 lists the exact coordinates available to the local classifier and to any downstream NameCert consumer.

The site and request rows are S and R. They identify where the socket appears and what missing discharge is being requested, without adding a row that supplies that discharge.

The classifier and audit rows are T and G. By Definition 542.2, T selects exactly one of the seven socket kinds, while G keeps the selected request behind the corresponding audit gate.

Finally H, C, P, N bind the transport, continuation, provenance, and local naming boundaries. Repacking these rows gives the whole local $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyLedger}^{\uparrow}}$ surface, and there is no remaining coordinate from which a consumer could import an internal theorem, hidden source object, or supplied request. □

Closure status for $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$ packet is generated from a socket site, requested supply row, seven-kind classifier, audit gate, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TranscendentalSupplyLedger}^{\uparrow}}, \mathsf{obligation})$

Scope closed: The obligation surface is exactly the carrier row, taxonomy exhaustion row, audit gate, componentwise transport, displayed continuation, provenance, and local NameCert rows of Definition 542.1, Definition 542.2, and Definition 542.3.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TranscendentalSupplyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, any Gap discharge, unrestricted induction, hidden witness selection, quotient equality, empirical persistence, inter-history recognition, inscription-source explanation, self-reference closure, or a public standard bridge.

Upgrade path: Scoped closure requires binding the displayed ledger obligations to explicit BEDC primitives for socket sites, requested supply rows, seven-kind classifiers, audit gates, transport, continuation replay, provenance, and local naming.

Chapter 543

A Concrete Naming Certificate for $\mathsf{ApophaticName}^{\uparrow}$

The $\mathsf{ApophaticName}^{\uparrow}$ packet is the concrete NameCert surface for a named externality. It records a socket-name handle, the requested supply shape, the refusal gate that keeps the supply outside the generated substrate, the ledger row that makes the boundary auditable, and the provenance rows through which a downstream chapter may cite the socket without turning it into an internal object. The packet consumes the socket surface of chapter 536 and the refusal-gate surface of chapter 715 before packaging the boundary as a named externality.

Definition 543.1 (ApophaticName carrier)**.** *An* $\mathsf{ApophaticName}^{\uparrow}$ *carrier is a finite* BHist *socket-name packet*

$$A = (S, R, G, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the socket-name row, recording the named boundary position, the local history window, and the chapter site at which the external supply is requested;*
2. *R is the requested supply-shape row, recording the finite interface that a caller would have to provide without asserting that the supply is generated by the substrate;*
3. *G is the refusal-gate row, recording that the requested supply is not admitted as an axiom, quotient representative, hidden theorem, host object, or substrate constructor;*
4. *L is the refusal-ledger row, recording each permitted consumer read and each barred conversion from socket to internal content;*
5. *H records componentwise* hsame *transports for the socket name, requested supply shape, refusal gate, ledger row, continuation route, package provenance, and local naming row;*
6. *C records the displayed* Cont *routes by which consumers read the socket, the request, the refusal decision, and the ledger entry;*
7. *P is the* Pkg *provenance over* BHist*, socket names, requested supply shapes, refusal gates, refusal ledgers,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row.*

The carrier contains no supplied object behind the socket, no theorem asserting that the requested supply exists, no quotient equality for external positions, and no host-side realization of the named dependency.

Lean4 (definition) `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier`

Theorem 543.2 (ApophaticName refusal ledger)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *carrier*

$$A = (S, R, G, L, H, C, P, N),$$

the local refusal ledger exposes exactly the socket-name row S*, the requested supply-shape row* R*, the refusal-gate row* G*, the ledger row* L*, the componentwise* hsame *transport row* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*. It records that consumers may cite the named external dependency only as a visible boundary request, and it supplies no internal axiom, quotient, hidden theorem, or host object.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger surface`
↪ variant `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger`

Proof. Project the accepted carrier of Definition 543.1. The only rows available to a consumer are S, R, G, L, H, C, P, N.

Read the socket and request through S and R. These rows make the external dependency referable and describe the finite supply shape, but they do not provide the supply itself.

Use the refusal gate G and ledger row L. They record that an attempted conversion from named socket to internal content is refused, with separate entries for axioms, quotient representatives, hidden theorems, and host objects.

Finally route the read through H, C, P, N. Transport, continuation, package provenance, and local naming preserve the same boundary packet, so the ledger exposes the dependency without creating a substrate-internal witness. □

Theorem 543.3 (ApophaticName socket carrier)**.** *Every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet exposes its socket row, boundary-request row, refusal row,* hsame *transport row,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row as finite* BHist *data. For*

$$A = (S, R, G, L, H, C, P, N),$$

the displayed surface is exactly S, R, G, L, H, C, P, N*:* S *names the socket,* R *gives the boundary request,* G *and* L *record the refusal decision and ledger,* H *transports the components,* C *gives the consumer route,* P *records provenance, and* N *is the local naming row.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket carrier`

Proof. Project the accepted carrier through Definition 543.1. Its tuple has only the eight displayed coordinates S, R, G, L, H, C, P, N, and each coordinate is introduced there as finite BHist data.

Apply the refusal-ledger surface of Definition 543.2. It identifies the same coordinates as the complete consumer ledger for the named boundary request.

The socket-facing side is compatible with the sibling socket surface of Definition 536.5. The gate-facing side is compatible with the visible refusal boundary of Definition 715.3. Thus the accepted packet exposes the socket and refusal rows as finite boundary data and not as a hidden supply. □

Theorem 543.4 (ApophaticName refusal classifier stability)**.** *Componentwise* hsame *transport of accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets preserves the refusal classifier. In particular, transport may move the displayed socket-name row, boundary-request row, refusal gate, refusal ledger, continuation route, provenance row, and local naming row, but it cannot replace the socket by a generated witness, an assumed proposition, a quotient representative, a concealed theorem, or a host-side object.*

Proof. Start from the carrier rows exposed by Definition 543.3. The transport row H is componentwise over the same socket, request, refusal, ledger, continuation, provenance, and naming coordinates.

Use the local classifier stability theorem Definition 543.50. It states that transported packets still read only the displayed coordinates S, R, G, L, H, C, P, N.

Finally compare the refusal gate with Definition 536.3 and the gate-question boundary Definition 715.3. Both sibling surfaces keep the refused supply outside the generated substrate, so a componentwise transport has no coordinate in which to install a generated witness. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal classifier stability`

Theorem 543.5 (ApophaticName ledger nonescape)**.** *Every downstream read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the displayed refusal ledger. The exported ledger consists only of the socket row, boundary-request row, refusal row,* hsame *transport row,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row, and it exports no hidden proof, host equality, quotient representative, generated witness, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier ledger nonescape`

Proof. Begin with Definition 543.2. It names the complete refusal ledger and records the permitted consumer reads together with the barred conversions.

Next apply Definition 543.6. It sends every downstream read through the displayed socket, request, refusal, ledger, transport, continuation, provenance, and naming rows.

The sibling socket boundary Definition 536.4 and the gate-question boundary Definition 715.3 rule out the same hidden supply coordinates. Repacking these rows through the local NameCert surface leaves no path to a hidden proof, host equality, quotient representative, generated witness, or closure principle. □

Theorem 543.6 (ApophaticName non-escape certificate)**.** *Every downstream read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the displayed socket-name, requested supply-shape, refusal-gate, ledger,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows of Definition 543.1. No such read may repackage the named externality as a generated BEDC object, an assumed proposition, a quotient class, a concealed theorem, or a host-side datum.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier non escape certificate`

Proof. Begin with the refusal ledger of Definition 543.2. It identifies the complete consumer surface of the packet and records the barred conversions.

Inspect the requested supply row R together with the refusal gate G. The request tells a caller what kind of external supply would be needed, while the gate prevents that request from becoming an internal source of truth.

The provenance rows H, C, P, N then carry the same boundary read to downstream chapters. Since none of these rows contains a constructor, witness, quotient equality, theorem assertion, or host realization of the supply, there is no escape route from named externality to internal objecthood. □

Theorem 543.7 (ApophaticName obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the socket-name packet* $A = (S, R, G, L, H, C, P, N)$*;*

2. *classifier stability for the socket row, requested supply shape, refusal gate, and refusal ledger under the displayed* hsame *transports;*
3. *ledger exactness for the permitted reads and barred conversions recorded by L;*
4. *continuation and package provenance through C and P;*
5. *non-escape for axioms, quotients, hidden theorems, generated objects, and host-side realizations.*

These rows make the dependency auditable while keeping the named supply outside the substrate's closure claim.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier obligation surface`

Proof. Carrier admission is the tuple displayed in Definition 543.1. Classifier stability is componentwise transport along H.

Ledger exactness is Definition 543.2. It names the only permitted reads and the barred conversions from socket to internal content.

Continuation and package provenance are the rows C and P. They expose the same packet to downstream consumers without adding a new object coordinate.

Non-escape is Definition 543.6. Together these five rows form the local NameCert surface for an apophatic socket name. □

Theorem 543.8 (ApophaticName root socket exposure)**.** *Every root consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet first sees the finite socket packet*

$$A = (S, R, G, L, H, C, P, N)$$

as displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The consumer receives no root-facing object outside the socket row, requested-supply row, refusal gate, refusal ledger, transport row, continuation row, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root socket exposure`

Proof. Read the carrier in Definition 543.1. It displays exactly the rows S, R, G, L, H, C, P, N and places them in one finite BHist packet.

Apply classifier stability from Definition 543.4. The displayed hsame transports preserve the socket row, requested supply shape, refusal gate, and ledger as the same consumer-facing packet.

Finally use the obligation surface, Definition 543.7. Its carrier, continuation, package, and local naming obligations are all attached to the same packet, so a root consumer first encounters the finite socket exposure rather than a hidden internal supply. □

Theorem 543.9 (ApophaticName root refusal gate totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, each attempted conversion from socket request to internal supply is either a displayed permitted ledger read or is routed through the refusal gate. No attempted conversion can bypass G and L to produce a generated object, host datum, quotient representative, hidden theorem, assumed proposition, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal gate totality`

Proof. Start with the refusal ledger, Definition 543.2. It lists the permitted socket and request reads and records the barred conversions governed by G and L.

The classifier stability theorem, Definition 543.4, transports this ledger without changing the displayed request or refusal coordinates. Thus a conversion attempt remains on the same finite socket route under hsame.

Apply ledger nonescape, Definition 543.5. Since downstream reads factor through the displayed refusal ledger and the carrier has no coordinate for internal supply, every non-permitted conversion is routed through the refusal gate. □

Theorem 543.10 (ApophaticName root downstream boundary package)**.** *The root-unblock surface exported by an accepted* ApophaticName$^{\uparrow}$ *packet is exactly the socket carrier, refusal classifier, ledger non-escape boundary, and downstream citation boundary over*

$$S, R, G, L, H, C, P, N.$$

Any downstream chapter may cite this package only as the finite socket/refusal surface, not as a generated witness, supplied object, host theorem, quotient representative, or global closure row.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root downstream boundary package`

Proof. First assemble the carrier from Definition 543.1. It gives the socket, request, gate, ledger, transport, continuation, provenance, and local naming rows as one finite packet.

Next attach the refusal classifier and ledger boundary using Definition 543.4 and Definition 543.2. These theorems identify the stable request/gate coordinates and the permitted ledger reads.

It remains to package the downstream boundary. Apply ledger nonescape, Definition 543.5, and the obligation surface, Definition 543.7. Together they expose the finite NameCert route while excluding generated witnesses, supplied objects, host theorems, quotient representatives, and global closure rows. □

Theorem 543.11 (ApophaticName socket request coherence)**.** *Every accepted* ApophaticName$^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N)$$

that exposes the socket row S also exposes the requested supply row R before the refusal gate G is consumed. A downstream consumer may therefore cite the socket only together with the displayed request, refusal ledger, transports, routes, provenance, and local naming rows.

Proof. Project the carrier of Definition 543.1. The socket and request rows are adjacent displayed coordinates of the same finite packet, not separate resources.

Apply the refusal ledger theorem, Definition 543.2. It records each permitted socket read together with the requested supply shape and the barred conversions governed by G and L.

Thus any consumer that reads S before the gate is consumed remains on the route through S, R, G, L, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket request coherence`

Theorem 543.12 (ApophaticName refusal-gate ledger totality)**.** *For every accepted* ApophaticName$^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, the refusal gate G and ledger L exhaust the permitted conversions from named externality to boundary request. They bar generated witnesses, host objects, quotient representatives, hidden theorems, and closure principles from being read out of the packet.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal gate ledger totality`

Proof. Use the obligation surface of Definition 543.7. It lists carrier admission, classifier stability, ledger exactness, continuation provenance, and non-escape as the finite obligations of the local NameCert row.

The non-escape certificate Definition 543.6 then rules out every conversion that would turn the named externality into internal content. Since G and L are the only refusal coordinates in the carrier, their ledger is total for the permitted boundary request conversions. □

Theorem 543.13 (ApophaticName downstream noninternalization package)**.** *Downstream consumers of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet receive only the finite package*

$$S, R, G, L, H, C, P, N.$$

They may transport and cite the socket request through the displayed rows, but they cannot internalize the named supply as a generated object, host datum, quotient representative, hidden theorem, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream noninternalization package`

Proof. Compose the socket carrier theorem Definition 543.3 with the refusal ledger Definition 543.2. The composite consumer surface is exactly the displayed finite packet.

Finally apply ledger nonescape, Definition 543.5. It sends downstream reads through the same socket, request, gate, ledger, transport, continuation, provenance, and local naming rows, so no consumer obtains an internal supply coordinate. □

Theorem 543.14 (ApophaticName root-unblock refusal boundary)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N)$$

factors through the displayed socket-name row S, requested supply-shape row R, refusal gate G, refusal ledger L, componentwise hsame *transports H,* Cont *routes C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N. No such consumer can read a generated witness, host object, quotient representative, hidden theorem, or closure principle from the packet.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root unblock refusal boundary`

Proof. Start with the carrier of Definition 543.1. It displays exactly the socket, request, refusal gate, ledger, transport, continuation, provenance, and local naming rows.

Apply the refusal ledger theorem, Definition 543.2. The ledger names the permitted socket reads and records the barred conversions from requested supply to internal content.

Use the socket carrier theorem, Definition 543.3. It identifies the downstream socket surface as the finite rows S, R, G, L, H, C, P, N rather than an internal witness or supplied object.

Finally apply ledger nonescape, Definition 543.5. Since every downstream read already factors through the displayed refusal ledger and the carrier has no extra coordinate for generated witnesses, host objects, quotient representatives, hidden theorems, or closure principles, carrier inspection leaves no other route. □

Theorem 543.15 (ApophaticName root socket-gate factorization)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, any root citation of the named externality factors through the socket row S, the requested supply-shape row R, the refusal gate G, and the refusal ledger L before it can reach the structural rows H, C, P, N. The factorization is compatible with the sibling* $\mathsf{ApophaticSocket}^{\uparrow}$ *and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *boundary surfaces and exports no generated witness, host datum, quotient representative, hidden theorem, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root socket gate factorization`

Proof. Begin with the carrier of Definition 543.1. It displays the socket, request, refusal gate, and ledger rows as separate coordinates before the transport, continuation, provenance, and local naming rows.

Use the root-unblock refusal boundary, Definition 543.14. It already sends every downstream root citation through the same eight-row packet S, R, G, L, H, C, P, N.

Now read the sibling boundaries. The socket surface of chapter 536 supplies the visible requested-supply shape, and the gate-question surface of chapter 715 supplies the visible refusal question. Together they force the citation through S, R, G, L before any structural replay row is used. □

Theorem 543.16 (ApophaticName root refusal-ledger exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, the refusal ledger row* L *exhausts the root-level barred conversions for the named externality. Any root consumer that asks for a supplied object, generated witness, host datum, quotient representative, hidden theorem, or closure principle is classified by the displayed gate* G *and refused by* L*, with only* H, C, P, N *available for transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal ledger exhaustion`

Proof. Start from the refusal ledger theorem, Definition 543.2. It records the permitted socket reads and the barred conversions from requested supply to internal content.

Apply the obligation surface, Definition 543.7. The local NameCert rows include carrier admission, classifier stability, ledger exactness, continuation provenance, and non-escape for the same packet.

Use the root socket-gate factorization, Definition 543.15. Since every root citation reaches G and L before structural replay, no separate root-level conversion remains outside the displayed refusal ledger. □

Theorem 543.17 (ApophaticName root downstream socket non-escape)**.** *Every downstream consumer that reads the root socket of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the displayed package*

$$S, R, G, L, H, C, P, N.$$

The read may cite the sibling $\mathsf{ApophaticSocket}^{\uparrow}$ *and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *boundaries as public socket and gate surfaces, but it cannot turn either sibling citation into a supplied object, generated witness, host datum, quotient representative, hidden theorem, or closure principle.*

Proof. First use the root-unblock refusal boundary, Definition 543.14. It confines downstream root consumers to the displayed socket, request, gate, ledger, transport, route, provenance, and naming rows.

Next use root refusal-ledger exhaustion, Definition 543.16. It says that root-level attempts to internalize the named externality are already classified by G and refused by L.

Finally compare with the sibling socket and gate-question boundaries. They make the socket and gate publicly citeable, but the ApophaticName packet supplies only the refusal package S, R, G, L, H, C, P, N. Thus a downstream socket read cannot escape into a supplied object or hidden proof. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root downstream socket nonescape`

Theorem 543.18 (ApophaticName root refusal boundary exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every root conversion attempt from the named externality into internal content is exhausted by the displayed refusal gate G, refusal ledger L, and the structural rows S, R, H, C, P, N. The attempt is either a socket request carried by S and R, a refusal-gate read classified by G, a barred conversion recorded by L, or a replay through H, C, P, N that preserves the same boundary refusal. There is no additional root route to a generated witness, quotient representative, host object, hidden theorem, supplied external object, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal boundary exhaustion`

Proof. Start with the carrier of Definition 543.1. It displays the accepted packet as exactly the eight rows S, R, G, L, H, C, P, N.

Apply the obligation surface of Definition 543.7. It records carrier admission, classifier stability, ledger exactness, continuation provenance, and non-escape as the local NameCert obligations for this packet.

Next use the root socket-gate factorization, Definition 543.15, together with root refusal-ledger exhaustion, Definition 543.16. These force a root read through the socket, requested supply-shape row, refusal gate, and ledger before any structural replay row is consumed.

Finally apply the non-escape certificate, Definition 543.6. Since the carrier has no coordinate for the listed internal objects and every conversion attempt has already been routed through G and L, the displayed refusal gate and ledger exhaust the root boundary. □

Theorem 543.19 (ApophaticName supply-shape noninternalization)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, the requested supply-shape row R remains a boundary request. It may be exposed to the socket row S, classified by the refusal gate G, recorded in the refusal ledger L, transported through H, replayed through C, authenticated by P, and named by N, but it cannot be internalized as a generated witness, quotient representative, host object, hidden theorem, supplied external object, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier supply shape noninternalization`

Proof. Project the carrier and the refusal ledger. The carrier Definition 543.1 gives R only as the requested supply-shape row, while Definition 543.2 records the barred conversions from requested supply into internal content.

Use the supply-shape refusal exactness theorem, Definition 543.67. It identifies reads of the requested supply shape with the visible boundary row rather than with a generated or host-side object.

Now add the root boundary. By Definition 543.18, any root attempt to turn R into internal content has already been classified by G and refused by L.

The non-escape certificate Definition 543.6 removes the remaining alternatives. Thus R is preserved as a boundary request through S, G, L, H, C, P, N and is not an internal witness, representative, object, theorem, supplied externality, or closure principle. □

Theorem 543.20 (ApophaticName downstream socket ledger factorization)**.** *Every downstream socket consumer that cites* $\mathsf{ApophaticName}^{\uparrow}$ *factors through the finite packet*

$$S, R, G, L, H, C, P, N$$

and through the sibling ApophaticSocket$^{\uparrow}$ *and* ApophaticGateQuestion$^{\uparrow}$ *socket and gate surfaces before the citation reaches the ApophaticNameUp boundary. The citation therefore consumes a ledgered refusal interface, not an unrecorded supply channel or hidden closure object.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream socket ledger factorization`

Proof. Begin with downstream socket route exactness, Definition 543.61. It sends downstream reads through S, R, G, L, H, C, P, N and leaves no route outside the displayed packet.

Next use root downstream socket non-escape, Definition 543.17. It permits public citation of the sibling socket and gate-question boundaries while keeping the consumer inside the same refusal packet.

Apply the root socket-gate factorization, Definition 543.15. This places the socket row S, requested supply-shape row R, refusal gate G, and refusal ledger L before the transport, continuation, provenance, and local naming rows.

Finally combine refusal boundary exhaustion and supply-shape noninternalization from Definition 543.18 and Definition 543.19. The downstream consumer may cite the sibling surfaces and the ApophaticNameUp boundary only through the ledgered packet, so it cannot obtain an unrecorded supply channel, generated witness, host object, quotient representative, hidden theorem, or closure principle. □

Theorem 543.21 (ApophaticName root refusal terminal nonpromotion)**.** *Let $A = (S, R, G, L, H, C, P, N)$ be an accepted* ApophaticName$^{\uparrow}$ *packet. Any terminal root conversion attempt that has already passed through the downstream socket ledger remains a refusal-boundary read of S, R, G, L, H, C, P, N. The requested supply-shape row R cannot be promoted into an internal generated witness, quotient representative, host object, hidden theorem, supplied external object, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal terminal nonpromotion`

Proof. Begin with downstream socket ledger factorization, Definition 543.20. A terminal downstream consumer citing the ApophaticNameUp boundary must pass through S, R, G, L, H, C, P, N and the sibling socket and gate surfaces before the citation reaches the refusal packet.

Apply root refusal boundary exhaustion and supply-shape noninternalization, Definition 543.18 and Definition 543.19. These results classify every root conversion attempt by G, record the barred conversion in L, and keep the requested supply-shape row R as boundary data.

Finally use the non-escape certificate, Definition 543.6. Since the carrier has no coordinate for generated witnesses, quotient representatives, host objects, hidden theorems, supplied external objects, or closure principles, passing through the downstream socket ledger cannot turn the refusal boundary into any of those internal objects. □

Corollary 543.22 (ApophaticName root refusal boundary strict obstruction)**.** *Let $A = (S, R, G, L, H, C, P, N)$ be an accepted* ApophaticName$^{\uparrow}$ *packet. Any consumer-facing use of the root refusal boundary as an obstruction is strictly local to the displayed finite route*

$$S, R, G, L, H, C, P, N.$$

It blocks exactly the converse or reuse attempt that would reread the named externality as a generated witness, quotient representative, host object, hidden theorem, supplied external object, closure principle, unrecorded supply channel, or hidden external validation. It cannot be reinterpreted as a global impossibility claim, a host theorem, an ambient countermodel, or an external validation result outside the accepted packet.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal boundary strict obstruction`

Proof. Begin with root refusal boundary exhaustion, Definition 543.18. It classifies every root conversion attempt into the displayed alternatives: a socket request through S and R, a refusal-gate read classified by G, a barred conversion recorded by L, or a replay through H, C, P, N that preserves the same boundary refusal. The same theorem also excludes any additional root route to a generated witness, quotient representative, host object, hidden theorem, supplied external object, or closure principle.

Now apply downstream socket ledger factorization, Definition 543.20. Any downstream consumer that cites the $\mathsf{ApophaticName}^{\uparrow}$ boundary must first factor through the finite packet S, R, G, L, H, C, P, N and through the sibling socket and gate surfaces before the citation reaches the boundary. Thus a consumer-facing obstruction has no coordinate except the displayed socket, request, refusal, ledger, transport, continuation, provenance, and local naming rows.

The strictness is the separation between the permitted forward citation and the barred converse. In the permitted direction, the consumer may cite the ledgered refusal interface. In the barred converse or reuse direction, the consumer tries to turn that interface into supplied content, a generated witness, a quotient representative, a host object, a hidden theorem, a closure principle, an unrecorded supply channel, or hidden external validation. Root exhaustion sends that attempt back to G and L, and downstream factorization keeps every public use inside the same packet.

Therefore the obstruction is not a theorem about all ambient hosts, not a hidden host proof, not an external countermodel, and not a validation source outside A. It is exactly the local refusal-boundary obstruction carried by S, R, G, L, H, C, P, N against the named converse or reuse route. □

Theorem 543.23 (ApophaticName root request classifier totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every socket request at the root boundary is classified by the displayed route S, R, G, L, H, C, P, N *before any downstream read. The classifier exposes the socket row, request row, refusal gate, refusal ledger, transport row, continuation row, provenance row, and local naming row as one finite package.*

Proof. Project the carrier in Definition 543.1. It gives the socket, request, refusal gate, ledger, transport, continuation, provenance, and naming rows as the complete packet available at the root boundary.

Apply the refusal ledger and non-escape certificate, Definition 543.2 and Definition 543.6. These identify the request classifier with the displayed refusal rows rather than with a hidden supply channel.

Use the root-unblock refusal boundary, Definition 543.14. It sends every downstream consumer through the same package, so the request is classified before downstream citation begins. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root request classifier totality`

Theorem 543.24 (ApophaticName boundary ledger exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the ledger row L *exactly separates permitted socket citation from barred conversion into a generated witness, host object, quotient representative, hidden theorem, assumed proposition, or closure*

principle. Permitted citation stays on S, R, G, L, H, C, P, N; every barred conversion is answered by the refusal gate and ledger rows.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary ledger exactness`

Proof. Read the carrier and refusal ledger, Definition 543.1 and Definition 543.2. They display G, L as the rows that govern the named externality request.

Apply the non-escape certificate, Definition 543.6. It rules out generated witnesses, host objects, quotient representatives, hidden theorems, assumed propositions, and closure principles as exported contents of the packet.

Finally use the root-unblock refusal boundary, Definition 543.14. Since downstream consumers receive only the ledgered boundary package, the separation made by L is exact at the boundary. □

Theorem 543.25 (ApophaticName downstream socket citation completeness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every downstream chapter consuming $\mathsf{ApophaticName}^{\uparrow}$ *receives only the socket and refusal citation package S, R, G, L, H, C, P, N. It receives no generated witness, host object, quotient representative, hidden theorem, assumed proposition, closure principle, or other internal supply from the named externality.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream socket citation completeness`

Proof. Start with the carrier and refusal ledger, Definition 543.1 and Definition 543.2. They bind downstream citation to the same finite socket, request, refusal, transport, continuation, provenance, and naming rows.

Use the non-escape certificate, Definition 543.6. It bars the internal supply cases from becoming rows exported by the accepted packet.

Apply the root-unblock refusal boundary, Definition 543.14. Every downstream consumer is already routed through S, R, G, L, H, C, P, N, so the citation package is complete and no additional internal supply remains. □

Theorem 543.26 (ApophaticName boundary-request obligation triad)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

any consumer that cites the packet before downstream use must expose the socket row S, the requested supply row R, and the refusal gate G as one boundary-request triad. The triad is inseparable from the ledger, transport, continuation, package, and local naming rows: reading S without R, reading R without G, or reading G outside L, H, C, P, N is not a permitted $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *obligation route.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary request obligation triad`

Proof. Begin with the refusal ledger of Definition 543.2. It identifies S, R, G, L, H, C, P, N as the complete visible consumer surface and records the permitted socket and request reads.

Apply the obligation surface of Definition 543.7. Carrier admission, classifier stability, ledger exactness, continuation provenance, and non-escape are all obligations for the same finite packet, so the socket row, requested supply row, and refusal gate cannot be separated into independent downstream resources.

Finally compare the sibling socket and gate surfaces, Definition 536.5 and Definition 715.3. They keep socket citation and gate refusal on the displayed boundary. Thus a downstream consumer receives the triad only as the route through S, R, G, L, H, C, P, N. □

Theorem 543.27 (ApophaticName refusal-gate route locality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the refusal gate G and ledger row L localize every permitted read to the displayed socket-request ledger. They cannot be transported to an unrecorded host object, hidden theorem, quotient representative, generated witness, assumed proposition, or closure principle outside the rows S, R, G, L, H, C, P, N.

Proof. Use the root-unblock refusal boundary, Definition 543.14. It sends downstream consumers through the displayed socket, request, gate, ledger, transport, continuation, package, and local naming rows.

The refusal ledger Definition 543.2 records that G and L are the refusal coordinates for permitted reads and barred conversions. The obligation surface Definition 543.7 makes ledger exactness and continuation provenance part of the same local NameCert surface.

Apply the non-escape certificate, Definition 543.6. Since every downstream read factors through the displayed carrier rows and those rows contain no host realization, hidden theorem, quotient representative, generated witness, assumed proposition, or closure principle, the gate and ledger remain local to the visible socket-request route. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal gate route locality`

Theorem 543.28 (ApophaticName downstream ledger nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the finite refusal route*

$$S, R, G, L, H, C, P, N.$$

The route exports the socket, request, gate, ledger, transport, continuation, provenance, and local naming rows, and it exports no generated witness, quotient representative, hidden theorem, host datum, supplied external object, assumed proposition, or closure principle.

Proof. Start from the non-escape certificate, Definition 543.6. It already sends downstream reads through the displayed socket-name, requested supply-shape, refusal-gate, ledger, hsame, Cont, Pkg, and local NameCert rows.

Use the refusal ledger and obligation surface, Definition 543.2 and Definition 543.7. The ledger names the permitted reads, while the local NameCert obligations bind carrier admission, classifier stability, ledger exactness, continuation provenance, and non-escape to the same packet.

Finally cite the sibling socket and refusal-gate boundaries, Definition 536.4 and Definition 715.3. Together they leave downstream consumers with the displayed route S, R, G, L, H, C, P, N and no coordinate for generated witnesses, quotient representatives, hidden theorems, host data, supplied external objects, assumed propositions, or closure principles. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream ledger nonescape`

Theorem 543.29 (ApophaticName socket refusal)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N)$$

that cites the socket row S must also pass through the requested supply row R, refusal gate G, refusal ledger L, componentwise hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N. The socket cannot be converted into a generated witness, quotient representative, hidden theorem, host object, or closure principle.*

Proof. Start with the socket carrier theorem, Definition 543.3. It identifies the socket-facing surface as the finite rows S, R, G, L, H, C, P, N.

Apply the refusal ledger theorem, Definition 543.2. The ledger records the permitted socket read and the barred conversions from requested supply to internal content.

Now use the root-unblock refusal boundary, Definition 543.14. It says that a downstream consumer cannot bypass the ledger rows when citing the named externality.

Finally apply ledger nonescape, Definition 543.5. Since the packet has no coordinate for generated witnesses, quotient representatives, hidden theorems, host objects, or closure principles, a socket citation remains a refused boundary read. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket refusal`

Theorem 543.30 (ApophaticName externality handoff)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the only handoff available to a downstream chapter is the finite refusal packet S, R, G, L, H, C, P, N. A downstream chapter may cite the named externality as a socket request with an attached refusal ledger, but it cannot receive an internal object, assumed proposition, quotient representative, hidden theorem, host-side realization, or closure principle from that handoff.

Proof. Project the carrier of Definition 543.1. The handoff rows are exactly the socket, request, refusal gate, ledger, transport, continuation, provenance, and local naming coordinates.

Use the non-escape certificate, Definition 543.6. It sends every downstream read through the displayed carrier rows and excludes internal objecthood for the named dependency.

Apply the root-unblock refusal boundary and socket refusal theorem, Definition 543.14 and Definition 543.29. Together they force the downstream citation to retain the socket and refusal rows rather than replacing them by supplied content.

The componentwise hsame, Cont, Pkg, and NameCert rows then transport and name the same refusal packet. They do not add a coordinate for an assumed proposition, quotient representative, hidden theorem, host-side realization, or closure principle. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier externality handoff`

Theorem 543.31 (ApophaticName socket request totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every socket read of A carries the request row R on the same finite surface. A downstream consumer may cite the socket S only together with the requested supply-shape row R, the refusal gate G, refusal ledger L, componentwise hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N.*

Proof. Project the carrier of Definition 543.1. The socket and request rows are adjacent coordinates of the accepted packet $A = (S, R, G, L, H, C, P, N)$.

Use the refusal ledger theorem, Definition 543.2. It records that permitted socket reads are ledgered together with the requested supply shape and the barred conversions governed by G and L.

Apply non-escape, Definition 543.6. Since downstream reads remain inside the displayed carrier, no socket read can drop the request row or replace it by generated content, an assumed proposition, a quotient representative, a hidden theorem, host data, or a closure principle. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket request totality`

Theorem 543.32 (ApophaticName gate ledger exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the gate row G and ledger row L exhaust all barred conversions attached to the socket request. The packet has no second gate, hidden permission row, supplied external object, assumed proposition, quotient representative, host realization, or closure principle outside the displayed carrier.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier gate ledger exactness`

Proof. Read the accepted carrier from Definition 543.1. The only refusal coordinates in A are G and L, and the structural coordinates H, C, P, N only transport, route, package, and name that same finite packet.

Apply the refusal ledger theorem, Definition 543.2. It identifies the permitted socket reads and the barred conversions as the ledgered content of G and L.

Use the ledger nonescape and non-escape certificate, Definition 543.5 and Definition 543.6. Because no extra permission or supplied-content coordinate exists, the gate and ledger rows exhaust the barred conversions. □

Theorem 543.33 (ApophaticName boundary request nonescape)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

a downstream boundary request that reads S and R cannot become internal generated content. It remains a socket request with refusal gate, refusal ledger, componentwise transport, displayed continuation, provenance, and local naming rows, and it exports no internal object, generated witness, assumed proposition, quotient representative, hidden theorem, host-side realization, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary request nonescape`

Proof. Start from the carrier and refusal ledger, Definition 543.1 and Definition 543.2. They bind the socket request to the same packet $A = (S, R, G, L, H, C, P, N)$ and record the refusal rows for downstream reads.

Use the root-unblock refusal boundary and externality handoff, Definition 543.14 and Definition 543.30. These force downstream consumers to receive the finite refusal packet rather than a supplied internal object.

Finally apply non-escape, Definition 543.6. Since generated content, assumed propositions, quotient representatives, hidden theorems, host realizations, and closure principles are absent from the carrier, the boundary request remains external to generated BEDC content. □

Theorem 543.34 (ApophaticName socket-refusal route totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every accepted request that cites the socket row S *is routed through the displayed boundary-request row* R*, refusal-gate row* G*, refusal-ledger row* L*, componentwise* hsame *transport row* H*, displayed* Cont *route* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*. The route is total for the public socket-refusal surface and admits no generated witness, assumed proposition, quotient representative, hidden theorem, host-side object, or closure principle outside those rows.*

Proof. Start with the socket carrier and socket refusal theorems, Definition 543.3 and Definition 543.29. They expose S, R, G, L, H, C, P, N as the finite socket-facing packet and require every socket citation to retain the request and refusal rows.

Next use refusal classifier stability, Definition 543.4. Componentwise hsame transport cannot replace the refused socket by generated content or a host object.

Finally apply boundary request nonescape and ledger nonescape, Definition 543.33 and Definition 543.5. These rows exhaust the permitted request route, so no consumer has an additional path outside the displayed socket-refusal packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket refusal route totality`

Theorem 543.35 (ApophaticName downstream boundary exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

any downstream consumer of the named externality receives exactly the ledgered externality boundary S, R, G, L, H, C, P, N*. The consumer may cite the socket request and its refusal ledger, but it receives no internalized supply, generated witness, assumed proposition, quotient representative, hidden theorem, host-side realization, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream boundary exactness`

Proof. Project the carrier of Definition 543.1. The only public boundary rows available to a downstream consumer are the socket row, request row, refusal gate, ledger, transport, continuation, provenance, and local naming row.

Use downstream noninternalization through Definition 543.6 and the boundary request nonescape theorem. Together they keep the requested supply external to generated BEDC content.

Now attach the externality handoff of Definition 543.30. The handoff gives downstream chapters the same finite refusal packet and no internal object.

Finally apply socket-refusal route totality from Definition 543.34. Since all accepted reads already pass through S, R, G, L, H, C, P, N, the downstream boundary is exact. □

Theorem 543.36 (ApophaticName externality consumer totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the complete public consumer surface for the named externality is the package consisting of the socket row S*, request row* R*, refusal gate* G*, refusal ledger* L*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*. This package is the whole public route for consumers and contains no supplied external object, generated witness, assumed proposition, quotient representative, hidden theorem, host-side realization, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier externality consumer totality`

Proof. First package the socket and refusal rows. The carrier, refusal ledger, and refusal-gate ledger totality theorems identify S, R, G, L as the complete request and refusal surface attached to the named externality.

Next package the provenance rows. The obligation surface and componentwise transport theorem, Definition 543.7 and Definition 543.39, keep H, C, P, N tied to that same boundary packet.

Use externality handoff and downstream boundary exactness, Definition 543.30 and Definition 543.35, to show that every downstream consumer receives only this ledgered boundary.

Finally close by non-escape. The cited rows already rule out generated witnesses, assumed propositions, quotient representatives, hidden theorems, host realizations, and closure principles, so the displayed package exhausts the public consumer surface. □

Theorem 543.37 (ApophaticName visible request factorization)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every visible socket request factors through the displayed socket row S, request row R, refusal gate G, refusal ledger L, componentwise transport row H, continuation row C, provenance row P, and local naming row N. The factorization names a boundary request only; it does not supply generated content, an assumed proposition, a quotient representative, a hidden theorem, host-side realization, or a closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier visible request factorization`

Proof. Project the carrier of Definition 543.1. The visible request has no coordinates other than S, R, G, L, H, C, P, N in the accepted packet.

Apply the refusal ledger and obligation surface from Definition 543.2 and Definition 543.7. They bind the socket read to the request row and attach the refusal gate, refusal ledger, transport, continuation, provenance, and naming rows.

Use the boundary request nonescape theorem, Definition 543.33. It prevents the visible request from being replaced by generated content, assumed propositions, quotient representatives, hidden theorems, host realizations, or closure principles.

Finally cite the sibling socket and gate-question surfaces, Definition 536.5 and Definition 715.3. They are reached only through the displayed socket and refusal rows, so the visible request factors through the finite packet. □

Theorem 543.38 (ApophaticName request ledger classifier exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the request row R, refusal gate G, and refusal ledger L exhaust the permitted boundary reads of the visible socket S. Any downstream read of the named externality is classified as a socket request with the displayed refusal data, and no separate classifier row admits internal content, host realization, quotient representative, hidden theorem, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier request ledger classifier exhaustion`

Proof. Read the carrier and gate-ledger exactness theorem, Definition 543.1 and Definition 543.32. They identify R, G, L as the displayed request, gate, and refusal ledger rows attached to S.

Next apply socket request totality, Definition 543.31. Every visible socket read carries the request row R and remains on the same finite surface with G, L, H, C, P, N.

Use the refusal ledger theorem and non-escape certificate, Definition 543.2 and Definition 543.6. The ledger classifies the permitted boundary reads, while non-escape rules out a second classifier for internal content, host realization, quotient representatives, hidden theorems, or closure principles.

Thus the request ledger and classifier rows exhaust the visible boundary surface consumed by downstream chapters. □

Theorem 543.39 (ApophaticName socket gate componentwise transport)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

componentwise hsame *transport along H preserves the socket row S, request row R, refusal gate G, refusal ledger L, continuation row C, provenance row P, and naming row N as the same boundary package. Transport cannot introduce generated content, an assumed proposition, a quotient representative, a hidden theorem, host-side realization, or a closure principle.*

Proof. Start from the carrier and NameCert obligations, Definition 543.1 and Definition 543.7. The componentwise transport row H is part of the same packet as the socket, request, gate, ledger, continuation, provenance, and naming rows.

Use gate-ledger exactness, Definition 543.32. It says that G and L exhaust the barred conversions attached to the socket request, so transport has no permission row through which a generated witness could appear.

Apply visible request factorization and request ledger classifier exhaustion, Definition 543.37 and Definition 543.38. They keep the transported request inside S, R, G, L, H, C, P, N and keep all permitted reads classified by the displayed ledger.

Finally use boundary request nonescape, Definition 543.33. Since the excluded objects are absent before transport and are not produced by componentwise hsame, they remain absent after transport. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket gate componentwise transport`

Theorem 543.40 (ApophaticName socket request gate exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the socket row S is readable only as part of the displayed request and refusal package S, R, G, L, H, C, P, N. In particular, every accepted packet reads S together with the request row R, the refusal gate G, and the ledger row L, and the structural rows H, C, P, N only transport, route, package, and name that same boundary request. No consumer obtains a socket-only object, a supplied external object, a host proof, a quotient representative, a hidden theorem, or a closure principle from the socket row.

Proof. Start with visible request factorization, Definition 543.37. It sends every visible socket request through the displayed packet S, R, G, L, H, C, P, N and states that the factorized request supplies no internal content.

Next use request ledger classifier exhaustion, Definition 543.38. The rows R, G, L exhaust the permitted boundary reads of the visible socket S, so a socket read cannot bypass the request row or the refusal ledger.

The sibling socket and gate surfaces give the same boundary on the two sides of the package. Apply the consumer surface of Definition 536.5 and the refusal boundary of Definition 715.3. They keep the socket citation and the gate refusal visible without adding a supplied object.

Finally apply componentwise transport from Definition 543.39. The rows H, C, P, N preserve the same socket-request-gate package, so the exact public read of S is the finite route through S, R, G, L, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket request gate exactness`

Theorem 543.41 (ApophaticName consumer citation boundary)**.** *Every downstream citation of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the finite boundary package*

$$S, R, G, L, H, C, P, N.$$

The citation exports only a boundary request: it may name the socket row, request row, refusal gate, ledger row, transport row, continuation route, provenance row, and local naming row, but it exports no supplied object, no host proof, no quotient representative, and no closure principle.

Proof. Read the downstream citation through the carrier of Definition 543.1. The accepted packet has exactly the displayed rows S, R, G, L, H, C, P, N.

Apply socket request gate exactness, Definition 543.40. A citation of the socket already carries the request row and the refusal gate with the ledger, so the public citation is a boundary request rather than an object supply.

Use downstream boundary exactness and externality handoff, Definition 543.35 and Definition 543.30. These identify the handoff received by downstream chapters as the same finite refusal packet.

The non-escape certificate Definition 543.6 then removes the remaining internalization cases. Since the carrier has no coordinate for a supplied object, host proof, quotient representative, hidden theorem, or closure principle, the citation boundary exports only the displayed request package. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier consumer citation boundary`

Theorem 543.42 (ApophaticName named externality NameCert package)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the finite $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *package for named externalities consists of carrier admission, socket request gate exactness, refusal-ledger nonescape, componentwise socket-gate transport, and the consumer citation boundary. This package is sufficient for downstream chapters to cite the named externality as a boundary request, and it is not a package for supplied objects, host proofs, quotient representatives, hidden theorems, or closure principles.*

Proof. Carrier admission is the displayed tuple of Definition 543.1. The local NameCert obligations of Definition 543.7 attach classifier stability, ledger exactness, continuation provenance, and non-escape to this same packet.

The request and gate rows are fixed by socket request gate exactness, Definition 543.40. Thus the socket-facing part of the package is a visible boundary request, not an internal supply channel.

Refusal-ledger nonescape is supplied by Definition 543.5. Componentwise transport is supplied by Definition 543.39. These two results keep the refusal ledger and transport rows on the same finite route.

Finally apply the consumer citation boundary, Definition 543.41. Repacking these rows gives a single finite NameCert surface for named externalities, and every downstream citation remains outside supplied objects, host proofs, quotient representatives, hidden theorems, and closure principles. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier named externality namecert package`

Theorem 543.43 (ApophaticName refusal ledger exactness)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, the refusal ledger* L *is exact for downstream root-unblock use: it records precisely the permitted reads of* S *and* R*, the refusal decision carried by* G*, and the transports and replays through* H, C, P, N*. No consumer can add a further ledger row that converts the socket into a generated witness, quotient representative, hidden theorem, host object, supplied external dependency, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger exactness`

Proof. Begin with Definition 543.2. It identifies L as the ledger for the permitted socket and request reads and for the barred conversions.

Use Definition 543.5. Every downstream read already factors through the displayed refusal ledger, so a consumer has no unledgered path to hidden proof, host equality, quotient representative, generated witness, or closure principle.

Apply the non-escape certificate and root-unblock refusal boundary, Definition 543.6 and Definition 543.14. They confine downstream root-unblock use to S, R, G, L, H, C, P, N.

Finally cite the externality handoff theorem, Definition 543.30. The only handoff object is the same finite refusal packet, so exactness of L cannot be extended into a supplied object or closure principle. □

Theorem 543.44 (ApophaticName boundary request image row)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, every accepted ApophaticName consumer read of the boundary request has image inside the displayed finite tuple* S, R, G, L, H, C, P, N*. The image row may cite the socket, requested supply shape, refusal gate, refusal ledger, componentwise transport, continuation route, package provenance, or local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *component, and it cannot cite a supplied object, quotient representative, hidden theorem, host-side realization, generated witness, or closure principle outside that tuple.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary request image row`

Proof. Project the carrier in Definition 543.1. The accepted packet has exactly the socket, request, refusal gate, ledger, transport, continuation, provenance, and local naming coordinates.

Apply the refusal ledger and ledger exactness rows, Definition 543.2 and Definition 543.43. They identify the permitted reads of S and R, the decision carried by G, and the replay rows through L, H, C, P, N.

Finally use the sibling socket and gate boundaries already binding the chapter, Definition 543.3, Definition 543.14, and Definition 543.73. These boundaries keep every accepted consumer read inside the finite refusal tuple rather than an internal object or host-side witness. □

Theorem 543.45 (ApophaticName root bridge tuple visible-request factorization)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, every downstream boundary-request read factors through the visible request row* R*, refusal gate* G*, refusal ledger* L*, componentwise*

transport H, continuation route C, provenance P, and local $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N before any externality handoff. The read cannot bypass the tuple through a supplied object, quotient representative, hidden theorem, host datum, generated witness, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple visible request factorization`

Proof. Begin with the boundary request image row, Definition 543.44. It confines every accepted request image to the displayed tuple S, R, G, L, H, C, P, N.

Apply the refusal readback and ledger exactness rows, Definition 543.46 and Definition 543.43. The readback returns each accepted image to its corresponding finite component, while exactness keeps the refusal decision and replay rows inside G, L, H, C, P, N.

Use the local public-boundary package and socket citation scope already binding the chapter. They route the externality handoff only after the visible request and refusal rows have been read, so the handoff receives the displayed refusal tuple rather than a supplied object or host datum.

Since no coordinate of A stores a quotient representative, hidden theorem, generated witness, or closure principle, any downstream read that bypasses R, G, L, H, C, P, N leaves the accepted root bridge tuple. □

Theorem 543.46 (ApophaticName refusal readback row)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, every accepted image-row read has a readback to the corresponding finite* BHist *component: S reads back the socket, R the request, G the refusal decision, L the refusal ledger, H the componentwise transport, C the route, P the provenance, and N the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *component. The readback does not convert the socket into an internal object or supplied dependency.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal readback row`

Proof. Start with the carrier of Definition 543.1. Each displayed coordinate of S, R, G, L, H, C, P, N is finite BHist data or a public row over that data.

Use the refusal ledger theorem and exactness theorem, Definition 543.2 and Definition 543.43. They pair each permitted consumer read with the row that records it, so a read of the socket, request, gate, ledger, transport, route, provenance, or local name reads back to the corresponding component.

The socket and gate boundaries, Definition 543.3, Definition 543.14, and Definition 543.44, then exclude every readback path that would replace the displayed component by supplied content. Thus readback recovers the finite row and nothing beyond it. □

Theorem 543.47 (ApophaticName boundary request tuple totality)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, every accepted boundary-request read has both its image row and its readback row inside the same displayed tuple. Componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row exhaust the consumer surface, so the read cannot escape to a supplied object, hidden theorem, quotient representative, generated witness, host datum, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary request tuple totality`

Proof. Begin with the image and readback rows, Definition 543.44 and Definition 543.46. The image row confines each accepted boundary-request read to S, R, G, L, H, C, P, N, and the readback row returns each accepted image read to the corresponding finite BHist component of that tuple.

Apply the componentwise transport row, Definition 543.49. It says that the local $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ component replays exactly those image and readback rows over the same displayed coordinates under hsame transport.

It remains to account for the consumer surface. The continuation row C supplies only replay through the displayed boundary-request tuple, the provenance row P names the same finite packet, and the local naming row N exports no additional object. Thus the tuple already contains every accepted image, readback, transport, replay, provenance, and naming row.

The no-supplied-object conclusion follows because no coordinate of A stores a supplied object, hidden theorem, quotient representative, generated witness, host datum, or closure principle. Any consumer request for such content therefore leaves the accepted tuple surface and is rejected at the refusal boundary. □

Theorem 543.48 (ApophaticName refusal tuple bridge scope)**.** *Any bridge-facing consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N)$$

must preserve the image row, readback row, componentwise NameCert transport, refusal ledger, continuation route, package provenance, and local naming row together. The bridge scope exports no supplied object, hidden theorem, quotient representative, host datum, generated witness, or closure principle.

Proof. Begin with the image and readback rows, Definition 543.44 and Definition 543.46. They place every accepted boundary-request image inside S, R, G, L, H, C, P, N and return each accepted read to the corresponding finite BHist component.

Use boundary-request tuple totality, Definition 543.47. The continuation route, package provenance, and local naming row already exhaust the consumer surface, so a bridge-facing consumer receives the same finite refusal tuple rather than a separate boundary object.

Apply componentwise NameCert transport, Definition 543.49. It replays the image and readback rows over the same displayed coordinates under hsame transport, preserving the refusal ledger, continuation route, package provenance, and local naming row together.

Since the accepted tuple has no coordinate for a supplied object, hidden theorem, quotient representative, host datum, generated witness, or closure principle, the bridge scope cannot export any such object. It exports only the displayed refusal tuple and its componentwise transport route. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal tuple bridge scope`

Theorem 543.49 (ApophaticName componentwise NameCert transport row)**.** *For accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets transported along the displayed componentwise* hsame *row* H*, the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *component replays the image and readback rows componentwise over* S, R, G, L, H, C, P, N*. The transport keeps the socket a public boundary component and never converts it into an internal object, supplied dependency, quotient representative, hidden theorem, host datum, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier componentwise namecert transport row`

Proof. Project the carrier from Definition 543.1. The transport row H is one coordinate of the same accepted packet as the socket, request, gate, ledger, continuation, provenance, and local NameCert rows.

Apply the refusal ledger and exactness rows, Definition 543.2 and Definition 543.43. They make the rows that are eligible for transport exactly the finite refusal tuple, with no hidden coordinate for supplied content.

Now use the image and readback rows, Definition 543.44 and Definition 543.46. The local NameCert transport can replay only those componentwise rows, because each consumer image and each readback has already been confined to S, R, G, L, H, C, P, N.

Finally cite the sibling socket and gate boundaries, Definition 543.14 and Definition 543.73. They keep the transported socket on the public refusal boundary, so the transport is a replay of the displayed rows rather than an internalization of the socket. □

Theorem 543.50 (ApophaticName classifier stability)**.** *Accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets transported componentwise along the displayed* hsame *rows preserve the socket-name row, requested supply shape, refusal gate, refusal ledger, continuation routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row. For a packet*

$$A = (S, R, G, L, H, C, P, N),$$

the transported classifier still reads only S, R, G, L, H, C, P, N *and cannot gain a supplied object, axiom, quotient representative, hidden theorem, or host-side realization.*

Proof. Project the carrier tuple $A = (S, R, G, L, H, C, P, N)$ from Definition 543.1. The classifier-facing coordinates are exactly the socket, request, refusal gate, ledger, transport, continuation, package, and naming rows.

Use the refusal ledger of Definition 543.2. It records the permitted socket reads and the barred conversions from requested supply to internal object, axiom, quotient representative, hidden theorem, or host-side datum.

Apply the obligation surface of Definition 543.7. Its classifier-stability, ledger-exactness, continuation, package, and non-escape rows exhaust the local NameCert obligations for the same tuple.

Componentwise hsame transport can therefore move only the displayed coordinates. Since C and P preserve the continuation and package provenance rows, and since N remains the local naming row for that transported packet, no additional coordinate appears in which a supplied object, axiom, quotient representative, hidden theorem, or host-side realization could be stored. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier classifier stability`

Theorem 543.51 (ApophaticName source-refusal totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the concrete socket row S*, requested supply-shape row* R*, refusal gate* G*, refusal ledger* L*, componentwise* hsame *row* H*,* Cont *route row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N *exhaust the source-refusal surface over* BHist*. No consumer can ask the packet for an internal object, assumed proposition, quotient representative, hidden theorem, or host-side datum without first encountering the refusal ledger.*

Proof. Project the carrier of Definition 543.1. Its concrete packet rows are exactly S, R, G, L, H, C, P, N, and each row is finite BHist data.

Apply the refusal ledger of Definition 543.2. It records the permitted socket and request reads together with the barred conversions from a named externality to internal content.

Use classifier stability from Definition 543.50. Componentwise hsame transport preserves the same socket, request, gate, ledger, continuation, package, and local naming rows, so no new source coordinate appears during transport.

Finally apply non-escape through Definition 543.6. The displayed Cont, Pkg, and NameCert rows export the refusal surface but do not install an internal witness behind the socket. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier source refusal totality`

Theorem 543.52 (ApophaticName public boundary package)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the public root-unblock package exposed over BHist *is exactly the socket row* S*, request row* R*, refusal gate* G*, refusal ledger* L*, componentwise* hsame *row* H*, continuation route* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*. The package exports no supplied dependency, internal object, assumed proposition, quotient representative, hidden theorem, host object, or closure principle beyond these rows.*

Proof. Project the carrier from Definition 543.1. Its accepted packet has exactly the displayed rows S, R, G, L, H, C, P, N, with no extra coordinate in which supplied content could be stored.

Use the refusal ledger and source-refusal totality, Definition 543.2 and Definition 543.51. They send every requested supply read through the gate and ledger before a consumer can ask for internal content.

Apply the non-escape certificate, Definition 543.6. It rules out converting the public packet into an axiom, quotient representative, hidden theorem, host-side datum, or closure principle, so the public root-unblock package is exhausted by the eight displayed rows. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier public boundary package`

Theorem 543.53 (ApophaticName socket citation scope)**.** *Every downstream citation of an* $\mathsf{ApophaticName}^{\uparrow}$ *socket must retain the request row* R*, refusal gate* G*, refusal ledger* L*, continuation route* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N *together with the socket row* S *and transport row* H*. A citation that drops any of these rows is not a citation of the accepted public package.*

Proof. Start with the socket carrier and public boundary package, Definition 543.1 and Definition 543.52. They identify the cited public object with the finite packet S, R, G, L, H, C, P, N.

Apply source-refusal totality from Definition 543.51. The source side is not merely a socket name: each accepted request is paired with the refusal gate and ledger that block conversion into supplied content.

Finally use the downstream non-escape and provenance rows, Definition 543.6 and Definition 543.30. They force socket citations to keep the continuation route, package provenance, and local naming row that make the refusal auditable at the cited site. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket citation scope`

Theorem 543.54 (ApophaticName refusal audit package)**.** *The audit-facing refusal package of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet is exhausted by the finite rows*

$$S, R, G, L, H, C, P, N.$$

It cannot be converted by an auditor into an axiom, quotient representative, hidden theorem, host object, supplied dependency, or closure principle.

Proof. Read the audit package through the carrier and refusal ledger, Definition 543.1 and Definition 543.2. The ledger records permitted socket and request reads and records the blocked conversions at the same finite packet.

Use the socket citation scope from Definition 543.53. Any downstream audit citation still carries the request, gate, ledger, route, provenance, and local naming rows, so the auditor receives the same finite surface.

Apply source-refusal totality and the non-escape certificate, Definition 543.51 and Definition 543.6. These rule out every route from the audit package to an internal witness or host-side closure object, leaving only the displayed refusal rows. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal audit package`

Theorem 543.55 (ApophaticName refusal transport totality)**.** *For accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets*

$$A = (S, R, G, L, H, C, P, N)$$

and

$$A' = (S', R', G', L', H', C', P', N')$$

whose displayed rows are related componentwise by the hsame *transports, refusal transport preserves exactly the socket row, requested supply row, refusal gate, refusal ledger, transport row, continuation route, package provenance, and local naming row. The transported packet cannot create supplied content, an internal witness, an assumed proposition, a quotient representative, a hidden theorem, or a host object.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal transport totality`

Proof. Project both carriers through Definition 543.1. The only coordinates available before and after transport are S, R, G, L, H, C, P, N and $S', R', G', L', H', C', P', N'$.

Use the refusal ledger and ledger nonescape rows, Definition 543.2 and Definition 543.5. They keep every permitted read on the socket, request, gate, and ledger rows, and they bar every conversion into supplied or host-side content.

Apply source-refusal totality from Definition 543.51. Since the source-refusal surface is already exhausted by S, R, G, L, H, C, P, N, componentwise transport can only rename and replay those same rows through H, C, P, N.

The sibling socket and gate references preserve the same boundary shape. Definition 536.5 keeps the socket readable only as socket data, while Definition 715.3 keeps the gate attached to a visible refusal. Hence the transported packet remains exactly a refusal packet and has no coordinate in which supplied content can appear. □

Theorem 543.56 (ApophaticName socket-gate route exhaustion)**.** *Every socket read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the socket row* S*, requested supply row* R*, refusal gate* G*, refusal ledger* L*, componentwise transport* H*, continuation route* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*. No socket consumer can bypass this route to read the named externality as supplied content, an internal object, a quotient representative, a hidden theorem, a host datum, or a closure principle.*

Proof. Begin with the socket carrier and socket refusal rows, Definition 543.3 and Definition 543.29. They identify the socket-facing surface as the finite packet S, R, G, L, H, C, P, N rather than an internal witness.

Use gate-source compatibility and the sibling gate boundary. Definition 543.60 keeps S, R, G, L as one source surface, and Definition 715.3 prevents the gate from becoming an admitted supply.

Now apply refusal transport totality from Definition 543.55. Any socket read transported downstream remains on the same refusal rows and is replayed only through H, C, P, N.

Finally use source-refusal totality and downstream refusal-ledger exhaustion, Definition 543.51 and Definition 543.72. These exhaust the requested supply row, refusal ledger, continuation route, provenance, and local naming rows, leaving no socket route outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket gate route exhaustion`

Theorem 543.57 (ApophaticName downstream boundary consumption)**.** *Downstream consumers may consume the named externality of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet only as the finite refusal packet* S, R, G, L, H, C, P, N*. The boundary is not consumable as an internal witness, quotient representative, hidden theorem, host object, supplied dependency, assumed proposition, or closure principle.*

Proof. Start from externality handoff, Definition 543.30. The handoff object is already the finite refusal packet, so downstream citation begins with S, R, G, L, H, C, P, N rather than with supplied content.

Apply boundary citation exhaustion and downstream named-externality refusal, Definition 543.74 and Definition 543.77. They state that every downstream citation is exhausted by the public refusal rows.

Use socket-gate route exhaustion from Definition 543.56. A consumer that enters through the socket still passes through the requested supply row, refusal gate, ledger, transport, continuation, provenance, and local naming row.

The non-escape certificate and refusal transport totality, Definition 543.6 and Definition 543.55, rule out every attempted replacement by a witness, quotient representative, hidden theorem, host object, supplied dependency, assumed proposition, or closure principle. Therefore the only consumable downstream boundary is the finite refusal packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream boundary consumption`

Theorem 543.58 (ApophaticName classifier-refusal transport)**.** *Let* $A = (S, R, G, L, H, C, P, N)$ *and* $A' = (S', R', G', L', H', C', P', N')$ *be accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets whose socket, request, refusal-gate, ledger, continuation, package, and naming rows are related by the displayed componentwise* hsame *transports. Then the transported classifier carries the refusal ledger of* A *to the refusal ledger of* A' *through the displayed* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, and it cannot turn the named externality into an internal object, assumed proposition, quotient representative, hidden theorem, or host-side datum.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier classifier refusal transport`

Proof. Read both carriers through Definition 543.1. The only classifier-facing coordinates are the socket, request, gate, ledger, transport, continuation, provenance, and local name.

Use Definition 543.4. The componentwise hsame rows preserve the refusal classifier while moving from A to A'.

Now transport the ledger itself. The Cont routes replay the same consumer read, the Pkg row preserves provenance, and the local $\mathsf{NameCert}$ row names the transported packet as another accepted ApophaticName carrier.

The non-escape certificate and classifier stability rule out every attempted replacement by generated content. Since neither carrier has a coordinate for an internal object, assumed proposition, quotient representative, hidden theorem, or host-side datum, the transported refusal remains a refusal ledger. □

Theorem 543.59 (ApophaticName consumer-boundary exhaustion)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the displayed* BHist *rows*

$$S, R, G, L, H, C, P, N.$$

The consumer may read the socket name, requested supply shape, refusal gate, refusal ledger, componentwise hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row, and no further consumer boundary exists for host equality, quotient representatives, hidden theorem content, generated witnesses, or supplied external objects.*

Proof. Start with the ledger nonescape theorem Definition 543.5. It sends every downstream read through the displayed refusal ledger and excludes hidden proof, host equality, quotient representative, generated witness, and closure-principle routes.

Apply source-refusal totality from Definition 543.51. The source side has already been exhausted by the socket, request, gate, ledger, transport, continuation, package, and naming rows.

Apply classifier-refusal transport from Definition 543.58. Any transported consumer sees the same finite refusal surface through hsame, Cont, Pkg, and NameCert rather than a new supplied object.

Thus the consumer boundary is exhausted by S, R, G, L, H, C, P, N. There is no remaining carrier coordinate from which host equality, quotient representatives, hidden theorem content, generated witnesses, or supplied external objects could be read. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier consumer boundary exhaustion`

Theorem 543.60 (ApophaticName gate-source compatibility)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the socket row S, requested supply row R, refusal gate G, and refusal ledger L remain one displayed source surface under the componentwise hsame *row H and the displayed* Cont *route row C. A downstream consumer may transport and replay this source surface only as S, R, G, L, H, C, P, N; it cannot separate the request from the refusal gate or turn the gate into an internal supply.*

Proof. Project the carrier through Definition 543.1. The socket, request, gate, and ledger are introduced as four coordinates of the same finite BHist packet, with H and C recording only componentwise transport and displayed route replay.

Apply Definition 543.51. It exhausts the source side by S, R, G, L, H, C, P, N and sends every attempted read of the requested supply through the refusal ledger before any consumer can ask for content.

Finally compare the boundary with the sibling socket and gate surfaces. The socket consumer surface Definition 536.5 keeps the named socket visible, while the gate-question visible-socket nonescape row Definition 715.3 keeps the refusal decision attached to that socket. Thus hsame and Cont preserve one displayed source surface rather than splitting off a hidden supply coordinate. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier gate source compatibility`

Theorem 543.61 (ApophaticName downstream socket route exactness)**.** *Every downstream read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors exactly through*

$$S, R, G, L, H, C, P, N.$$

The route first reads the socket row and requested supply shape, then the refusal gate and refusal ledger, then the componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row. No downstream consumer has a route to a generated witness, assumed proposition, quotient representative, hidden theorem, host-side datum, or supplied external object outside these rows.*

Proof. Begin with Definition 543.59. It states that every downstream consumer is already confined to S, R, G, L, H, C, P, N and that no further consumer boundary exists for host equality, quotient representatives, hidden theorem content, generated witnesses, or supplied external objects.

Use Definition 543.60 to keep the first four rows as one source surface. The socket and request rows are readable only together with the refusal gate and refusal ledger, so a consumer cannot bypass G or L while still claiming to read the named dependency.

The sibling socket and gate references give the same exactness at the boundary. The socket noninternalization boundary Definition 536.4 excludes replacement by a generated object, and the visible gate-question boundary Definition 715.3 excludes replacement by an admitted supply. Replaying those rows through H, C, P, N gives precisely the stated downstream route. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream socket route exactness`

Theorem 543.62 (ApophaticName gate-ledger separation)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the socket row S, refusal gate G, and refusal ledger L remain separate readable rows while being transported together by H, C, P, N. The gate records the refusal decision, the ledger records permitted reads and barred conversions, and the socket row remains the visible downstream handle; none of these rows is absorbed into another row or converted into supplied content.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier gate ledger separation`

Proof. Project A through Definition 543.1. The socket, request, gate, and ledger are distinct coordinates of the finite carrier, while H, C, P, N are the displayed transport, replay, provenance, and local naming rows.

Use the sibling surfaces named by the chapter introduction. Definition 536.5 keeps the socket readable as a socket boundary, and Definition 715.3 keeps the gate attached to that visible boundary without admitting the requested supply.

Finally apply Definition 543.2 and Definition 543.61. They transport the separate socket, gate, and ledger rows together through hsame, Cont, Pkg, and NameCert, while excluding generated witnesses, host data, quotient representatives, hidden theorems, assumed propositions, and supplied external objects. □

Theorem 543.63 (ApophaticName socket-refusal correspondence)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

each downstream read of the socket row S corresponds to the refusal recorded by G and ledgered by L. The correspondence is transported only through the displayed hsame *row H, the* Cont *route C,*

the Pkg *provenance row P, and the local* NameCert *row N; it is not a route to the refused external object.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket refusal correspondence`

Proof. Begin with downstream socket route exactness. Definition 543.61 says that every downstream read factors through S, R, G, L, H, C, P, N and has no route outside those rows.

Compare the socket and gate sides. The socket boundary of Definition 536.4 prevents the visible socket from becoming a generated object, while Definition 715.3 prevents the refusal gate from becoming an admitted supply.

The correspondence is therefore the displayed refusal route itself. Transport through H, replay through C, provenance through P, and naming through N preserve the same socket-refusal packet and do not add host equality, quotient representatives, hidden theorem content, or supplied external content. □

Theorem 543.64 (ApophaticName ledger-refusal transport exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

transport of the refusal ledger is exact: any transported downstream consumer reads the same refusal relation between S, R, and G through L, H, C, P, N, and no transported route can replace the ledger by a generated witness, host-side datum, quotient representative, hidden theorem, assumed proposition, or supplied external dependency.

Proof. Start from classifier-refusal transport, Definition 543.58. It moves the refusal classifier componentwise through H and replays the consumer route through C, P, N.

Use ledger nonescape and source-refusal totality, Definition 543.5 and Definition 543.51. Together they exhaust every source-side and downstream read by the displayed ledgered packet.

Finally apply gate-ledger separation and socket-refusal correspondence. The first keeps S, G, L as separate finite rows, and the second identifies every socket read with the same refusal route, so the transported ledger remains exact over S, R, G, L, H, C, P, N and exposes no unlisted content. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier ledger refusal transport exactness`

Theorem 543.65 (ApophaticName closure refusal package)**.** *The accepted* $\mathsf{ApophaticName}^{\uparrow}$ *carrier assembles a five-row refusal package:*

1. *carrier admission for $A = (S, R, G, L, H, C, P, N)$;*
2. *classifier stability for the socket, request, gate, ledger, transport, route, provenance, and naming rows;*
3. *ledger exactness for permitted socket reads and barred conversions;*
4. *non-escape from generated witnesses, assumed propositions, quotient representatives, hidden theorems, host-side data, and supplied external objects;*
5. *downstream socket route exactness through S, R, G, L, H, C, P, N.*

This package is the local $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *refusal surface, and it supplies no external dependency as internal BEDC content.*

Proof. Carrier admission is the displayed tuple in Definition 543.1. Classifier stability is supplied by Definition 543.50, which preserves exactly the socket, request, gate, ledger, transport, continuation, package, and naming rows.

Ledger exactness is Definition 543.2 together with Definition 543.5. These rows identify the permitted reads and barred conversions and keep every consumer inside the displayed refusal ledger.

The non-escape certificate Definition 543.6 rules out the attempted conversions to generated objects, assumed propositions, quotient representatives, hidden theorems, and host-side data. Gate-source compatibility adds that the request and refusal gate remain one source surface under hsame and Cont.

Finally apply Definition 543.61. It packages the boundary read through S, R, G, L, H, C, P, N and the sibling ApophaticSocket and ApophaticGateQuestion surfaces, so the five listed rows form the local closure refusal package without installing the named externality as internal content. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier closure refusal package`

Theorem 543.66 (ApophaticName kernel scope triad)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the socket-name row S, requested-supply row R, and refusal-gate row G are read only through the displayed BHist *packet, the componentwise* hsame *transport row, the* Cont *replay row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row. No kernel read separates those three rows into a supplied object, assumed proposition, quotient representative, hidden theorem, or host-side datum.*

Proof. Project the accepted carrier by Definition 543.1. The rows S, R, G occur inside the same finite BHist packet as the ledger, transport, continuation, package, and naming rows.

Use the socket carrier and classifier stability theorems, Definition 543.3 and Definition 543.50. They keep socket, request, and refusal rows componentwise under hsame without adding a generated witness.

Finally apply the refusal ledger and consumer-boundary exhaustion, Definition 543.2 and Definition 543.59. These rows force every kernel read through L, H, C, P, N, so the triad remains a displayed refusal surface rather than an internal supply. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier kernel scope triad`

Theorem 543.67 (ApophaticName supply-shape refusal exactness)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, the requested supply shape R is exactly the row refused by the gate G and recorded by the ledger L. The packet introduces no generated witness, external object, host realization, quotient representative, hidden theorem, or assumed proposition in place of that refused row.*

Proof. Read R and G through the carrier definition. The request row and refusal-gate row are adjacent coordinates of the same displayed packet, and the ledger row L records the permitted reads and barred conversions.

Apply source-refusal totality from Definition 543.51. It exhausts the source-refusal surface by S, R, G, L, H, C, P, N and sends any attempted content read through the refusal ledger first.

Then use gate-source compatibility and the closure refusal package, Definition 543.60 and Definition 543.65. They keep the requested supply shape attached to the refusal gate and exclude every coordinate that would turn the request into supplied content. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier supply shape refusal exactness`

Theorem 543.68 (ApophaticName downstream refusal replay)**.** *Every downstream replay of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the refusal ledger* L*, classifier transport by* H*, the displayed* Cont *route* C*,* Pkg *provenance* P*, the local* NameCert *row* N*, and the consumer-boundary exhaustion of* S, R, G, L, H, C, P, N*. The replay exports only the refusal packet and never a generated witness, external object, host-side datum, quotient representative, hidden theorem, or assumed proposition.*

Proof. Begin with the refusal ledger of Definition 543.2. It names the visible socket, request, refusal gate, ledger, transport, continuation, package, and local naming rows.

Apply classifier-refusal transport from Definition 543.58. Transport moves the refusal ledger through the displayed hsame, Cont, Pkg, and NameCert rows while preserving the refusal classifier.

Finally use downstream socket route exactness and consumer-boundary exhaustion, Definition 543.61 and Definition 543.59. They exhaust every downstream replay by the same finite refusal packet, leaving no route to supplied or host-side content. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream refusal replay`

Theorem 543.69 (ApophaticName supply-request consumer factorization)**.** *Every consumer read that asks for the requested supply-shape row of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the displayed socket row* S*, requested supply row* R*, refusal gate* G*, refusal ledger* L*,* hsame *transport row* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N*. The read cannot skip the refusal ledger to obtain an internal object, axiom, quotient representative, hidden theorem, or host datum.*

Proof. Project the carrier $A = (S, R, G, L, H, C, P, N)$ from Definition 543.1. The requested supply-shape row is not a free-standing source; it is one displayed coordinate of the same refusal packet as the socket, gate, ledger, transport, route, provenance, and naming rows.

Use source-refusal totality from Definition 543.51. It sends every accepted source request through S, R, G, L, H, C, P, N and refuses the conversion from requested shape to supplied object.

Apply downstream refusal replay from Definition 543.68. Replay preserves the same refusal ledger and route rows for downstream consumers, so a supply-shape read is still a ledgered refusal read.

Finally cite the socket and gate boundaries already binding the chapter, Definition 543.3, Definition 543.60, and Definition 543.59. These boundaries exclude internal objects, axioms, quotient representatives, hidden theorems, and host data from the accepted packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier supply request consumer factorization`

Theorem 543.70 (ApophaticName kernel-scope refusal triad)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, every kernel-scope read of the socket-name row* S*, requested supply row* R*, and refusal-gate row* G *is confined to the concrete BEDC rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$*. The triad cannot be read as an abstract source equivalence, host object, quotient representative, hidden theorem, assumed proposition, or supplied external dependency.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier kernel scope refusal triad`

Proof. Project A through Definition 543.1. The socket, request, gate, ledger, transport, continuation, provenance, and naming coordinates are finite BHist rows.

Use Definition 543.66. It already confines the socket-name, requested-supply, and refusal-gate rows to the displayed packet and forbids their separation into supplied content.

Finally apply Definition 543.2 and Definition 543.7. The ledger and local NameCert obligations are concrete BEDC rows, so the kernel-scope triad remains a refusal surface rather than an abstract source equivalence. □

Theorem 543.71 (ApophaticName source-refusal transport)**.** *Let $A = (S, R, G, L, H, C, P, N)$ and $A' = (S', R', G', L', H', C', P', N')$ be accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets whose displayed rows are related by the componentwise* hsame *transports recorded in H and H'. If a consumer reads A only as a source-refusal packet over* BHist, Cont, Pkg, *and* NameCert, *then the transported consumer reads A' only through $S', R', G', L', H', C', P', N'$ and still receives no generated witness, host object, quotient representative, hidden theorem, assumed proposition, or supplied external dependency.*

Proof. Start from source-refusal totality, Definition 543.51. It exhausts the source side of A by the concrete rows S, R, G, L, H, C, P, N.

Transport the displayed rows with Definition 543.58. The hsame row moves the refusal classifier componentwise, while Cont, Pkg, and NameCert replay and rename the same refusal packet.

Apply Definition 543.59. The transported read has no coordinate for supplied content, so A' remains a source-refusal packet over the concrete BEDC rows rather than an abstract equivalence between sources. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier source refusal transport`

Theorem 543.72 (ApophaticName downstream refusal ledger exhaustion)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$ that cites the named externality must exhaust the displayed refusal ledger L together with the* BHist *socket and request rows, the* hsame *transport row H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N. After those rows are exhausted, no downstream route remains to a generated BEDC object, host-side datum, quotient representative, hidden theorem, assumed proposition, or supplied external dependency.*

Proof. Begin with Definition 543.68. It forces downstream replay through the refusal ledger, classifier transport, displayed continuation route, package provenance, local name, and consumer-boundary exhaustion.

Use Definition 543.69. A consumer that asks for the requested supply shape is still routed through S, R, G, L, H, C, P, N and cannot skip L.

Finally cite Definition 543.61 and Definition 543.59. Together they identify the downstream route as exactly the displayed ledgered packet and exclude every supplied or host-side object outside the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream refusal ledger exhaustion`

Theorem 543.73 (ApophaticName socket-request gate factorization)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$, every public read of the socket row S together with the requested supply-shape row R factors through the refusal gate G and refusal ledger L before it reaches the displayed* hsame, Cont, Pkg, *and* NameCert *rows. The factorization exports a public refusal boundary, not a supplied object, host datum, quotient representative, hidden theorem, assumed proposition, or generated witness.*

Proof. Project A through Definition 543.1. The socket, request, gate, ledger, transport, continuation, provenance, and local name are one finite packet, so the request row is not available as an independent source of content.

Apply the refusal-ledger and non-escape theorems, Definition 543.2 and Definition 543.6. They identify the permitted public reads and keep every attempted content read behind the refusal ledger.

Now use socket-refusal and source-refusal totality, Definition 543.29 and Definition 543.51. These force each socket-request read through G, L, H, C, P, N before a downstream consumer can cite the packet.

The resulting public surface is the displayed refusal boundary. Since no coordinate of A stores a supplied object or host realization, the factorization remains a refusal route through the local NameCert packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket request gate factorization`

Theorem 543.74 (ApophaticName boundary citation exhaustion)**.** *Every downstream citation of the named externality carried by an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$ *is exhausted by the public boundary rows*

$$S, R, G, L, H, C, P, N.$$

The citation must pass through the refusal ledger, socket-refusal route, externality handoff, source-refusal totality, and downstream refusal-ledger exhaustion; no further boundary remains for supplied external content, host-side data, quotient representatives, hidden theorem content, assumed propositions, or generated witnesses.

Proof. Start with the socket-request gate factorization of Definition 543.73. It places public socket and request reads behind the same gate and ledger rows.

Use externality handoff from Definition 543.30. The handoff lets downstream chapters cite the named dependency only as the finite refusal packet A.

Apply source-refusal totality and downstream refusal-ledger exhaustion, Definition 543.51 and Definition 543.72. Together they exhaust both the source side and the downstream side by S, R, G, L, H, C, P, N.

Finally cite non-escape through Definition 543.6. The packet has no coordinate for supplied external content or host realization, so the public boundary citation is exhausted by the displayed refusal rows. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary citation exhaustion`

Theorem 543.75 (ApophaticName refusal provenance transport)**.** *Let* $A = (S, R, G, L, H, C, P, N)$ *and* $A' = (S', R', G', L', H', C', P', N')$ *be accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packets related by the displayed componentwise* hsame *transports. If a downstream consumer cites* A *only through its public refusal boundary, then the transported citation of* A' *is again only through* $S', R', G', L', H', C', P', N'$ *and carries the same refusal provenance through* Cont, Pkg, *and* NameCert.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal provenance transport`

Proof. Begin with classifier-refusal transport, Definition 543.58. It moves the refusal classifier along the displayed socket, request, gate, ledger, route, provenance, and naming rows without creating a supplied coordinate.

Use source-refusal transport from Definition 543.71. A consumer that reads A only as a source-refusal packet is transported to a consumer that reads A' only through the corresponding displayed rows.

Now apply boundary citation exhaustion, Definition 543.74. The downstream citation is already exhausted by the public refusal rows, so transport cannot add host data, quotient representatives, hidden theorem content, assumed propositions, or generated witnesses.

The provenance rows C', P', N' therefore replay, package, and name the same refusal boundary for A'. This is refusal provenance transport, not transport of the refused external object. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameRefusalProvenanceTransport`

Theorem 543.76 (ApophaticName refusal export totality)**.** *For an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, the complete public export boundary is the refusal packet on* S, R, G, L, H, C, P, N*. Every exported read of the named externality is classified by the socket row, requested supply-shape row, refusal gate, refusal ledger, componentwise transport, continuation route, package provenance, and local* $\mathsf{NameCert}$ *row, and by no supplied external object.*

Proof. Project the accepted carrier through Definition 543.1. The socket, requested supply-shape, gate, ledger, transport, continuation, provenance, and naming rows are the only displayed coordinates.

Apply the refusal ledger and classifier transport boundaries, Definition 543.2 and Definition 543.58. They classify exported reads by the same ledgered refusal packet under componentwise transport.

Finally use consumer boundary exhaustion and downstream refusal ledger exhaustion, Definition 543.59 and Definition 543.72. Together they leave no exported route to a supplied external object outside S, R, G, L, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameRefusalExportTotality`

Theorem 543.77 (ApophaticName downstream named-externality refusal)**.** *Every downstream consumer that cites the named externality of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet receives only the displayed refusal packet* S, R, G, L, H, C, P, N*. The citation may replay the socket, request, gate, ledger, transport, continuation, provenance, and local naming rows, but it cannot convert the named externality into supplied content, a host datum, a quotient representative, a hidden theorem, or an assumed proposition.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream named externality refusal`

Proof. Start from downstream refusal ledger exhaustion, Definition 543.72. It requires every downstream consumer to exhaust the displayed refusal ledger together with the socket, request, transport, continuation, provenance, and local naming rows.

Use refusal export totality from Definition 543.76. The public export surface is already the refusal packet on S, R, G, L, H, C, P, N.

Apply boundary citation exhaustion, Definition 543.74. A downstream citation of the named externality is therefore a citation of the displayed refusal packet and has no remaining coordinate for supplied or host-side content. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal export totality`

Theorem 543.78 (ApophaticName supply request nonimport)**.** *The requested supply row R of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet cannot import a supplied object, axiom, quotient representative, hidden theorem, or host datum. It is read only as a requested supply-shape coordinate inside the refusal packet S, R, G, L, H, C, P, N, behind the refusal gate G and refusal ledger L.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier supply request nonimport`

Proof. Project the packet $A = (S, R, G, L, H, C, P, N)$. The row R is a coordinate of the finite refusal carrier, not a source of supplied content.

Apply supply-request consumer factorization, Definition 543.69. Any consumer read asking for R factors through S, R, G, L, H, C, P, N and cannot skip the refusal ledger.

Now use Definition 543.77. Downstream citation still receives only the displayed refusal packet, so the requested supply row cannot import an object, axiom, quotient representative, hidden theorem, or host datum. □

Theorem 543.79 (ApophaticName socket refusal totality)**.** *Every socket-facing consumer read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet $A = (S, R, G, L, H, C, P, N)$ must pass through the displayed refusal ledger, classifier-stability row, supply-request gate, and nonimport row before it can cite the packet. The resulting citation is a finite refusal packet over S, R, G, L, H, C, P, N, not a supplied object, host datum, quotient representative, hidden theorem, or assumed proposition.*

Proof. Begin with the refusal ledger, Definition 543.2. It identifies the visible socket, request, refusal-gate, ledger, transport, continuation, provenance, and naming rows as the consumer surface.

Use classifier stability from Definition 543.50. Componentwise hsame transport preserves that same refusal classifier and does not create another socket-facing coordinate.

Now apply the socket-request gate factorization and supply-request nonimport rows, Definition 543.73 and Definition 543.78. They force a socket-facing read through G, L, H, C, P, N before the packet can be cited downstream.

The packet has no coordinate for supplied content, host data, quotient representatives, hidden theorems, or assumed propositions. Hence every socket-facing citation is total only as the displayed refusal packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket refusal totality`

Theorem 543.80 (ApophaticName contextual class non-escape)**.** *The refusal surface of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet never supplies a contextual-class object, host selector, or imported ontology outside the displayed rows S, R, G, L, H, C, P, N. Any attempted contextual-class read is stopped by the refusal ledger, classifier stability, non-escape certificate, and supply-request nonimport boundary.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier contextual class nonescape`

Proof. Project the carrier through Definition 543.1. The only rows present are the socket, requested supply shape, refusal gate, refusal ledger, transport, continuation, provenance, and local naming row.

Apply the refusal ledger and classifier stability theorems, Definition 543.2 and Definition 543.50. They keep every transported consumer read on the same refusal rows.

Use the non-escape certificate, Definition 543.6. It rules out conversion of the named externality into a generated object, assumed proposition, quotient class, concealed theorem, or host-side datum.

Finally cite supply-request nonimport, Definition 543.78. Since the request row remains a requested supply-shape coordinate behind the refusal gate, no contextual-class object, host selector, or imported ontology can be supplied by the packet. □

Theorem 543.81 (ApophaticName boundary descent ledger)**.** *Downstream descent through the apophatic boundary of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet is ledger-bounded by the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. A consumer may descend only by replaying* S, R, G, L *through* H, C, P, N*, and it cannot add a supplied external object, host selector, imported ontology, quotient representative, hidden theorem, or assumed proposition.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary descent ledger`

Proof. Start from the carrier definition and the refusal ledger, Definition 543.1 and Definition 543.2. They place the boundary in finite BHist rows and identify L as the ledger for the permitted socket and request reads.

Use classifier stability, Definition 543.50. The descent rows are transported only componentwise along H, so the classifier remains attached to the displayed refusal packet.

Apply the non-escape certificate together with supply-request nonimport, Definition 543.6 and Definition 543.78. These results keep the requested supply shape behind the gate and rule out every conversion to supplied or host-side content.

Thus any downstream descent is exactly a replay through Cont, packaging through Pkg, and citation by the local NameCert row. The descent is ledger-bounded by S, R, G, L, H, C, P, N and has no other boundary object to export. □

Theorem 543.82 (ApophaticName downstream boundary consumer)**.** *Every downstream consumer read of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N)$$

factors through the displayed socket, requested supply-shape, refusal gate, ledger, componentwise transport, continuation, provenance, and local naming rows. The consumer receives only that refusal-boundary packet and no supplied object, quotient representative, hidden theorem, host datum, or generated witness.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream boundary consumer`

Proof. Project the carrier of Definition 543.1. The only rows presented to a downstream consumer are S, R, G, L, H, C, P, N.

Apply the refusal ledger, obligation surface, and non-escape certificate, Definition 543.2, Definition 543.7, and Definition 543.6. These rows identify the permitted reads, the local NameCert obligations, and the barred conversions for the same packet.

Finally use the sibling socket and gate surfaces already binding this chapter, Definition 543.73 and Definition 543.74. They force the downstream citation to remain a public refusal-boundary citation rather than a supplied object or host-side witness. □

Theorem 543.83 (ApophaticName refusal gate separation)**.** *The refusal gate* G *separates socket requests from generated witnesses, quotient representatives, hidden theorems, and host objects for every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet. A read may name the socket row* S *and requested supply-shape row* R *only by passing through* G, L, H, C, P, N*, and it cannot turn the request into substrate content.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal gate separation`

Proof. Begin with the refusal ledger of Definition 543.2. It records each permitted socket and request read together with the barred conversions from request to internal content.

Use the non-escape certificate, Definition 543.6. It rules out the attempted repackaging of the named externality as a generated object, assumed proposition, quotient class, hidden theorem, or host-side datum.

The socket-request gate factorization and boundary citation exhaustion, Definition 543.73 and Definition 543.74, keep the request behind G through Cont, Pkg, and NameCert. Hence the gate separates the public request from every hidden or host-side realization. □

Theorem 543.84 (ApophaticName NameCert handoff exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *handoff through* N *exhausts all public refusal-boundary exports of an accepted packet* $A = (S, R, G, L, H, C, P, N)$*. Any public export is a replay of the refusal ledger through* H, C, P, N*, not an export of supplied content, a quotient representative, a hidden theorem, a generated witness, or a host object.*

Proof. Start from Definition 543.7. Its carrier, classifier-stability, ledger-exactness, continuation, package, and non-escape rows are the local NameCert obligations for the packet.

Apply the refusal ledger and non-escape certificate, Definition 543.2 and Definition 543.6. They identify the public refusal rows and exclude every conversion to supplied or host-side content.

Finally cite downstream named-externality refusal and boundary descent, Definition 543.77 and Definition 543.81. Together they show that a downstream export is exhausted by replay through H, C, P, N, so the local NameCert handoff has no remaining hidden route to export. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameNameCertHandoffExhaustion`

Theorem 543.85 (ApophaticName boundary socket provenance)**.** *Every boundary citation of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet factors through the socket surface, refusal-gate provenance rows, and local* NameCert *row in the displayed packet* S, R, G, L, H, C, P, N*.*

Proof. Project the carrier and use Definition 543.3; the socket citation is already a read of the finite surface S, R, G, L, H, C, P, N.

Apply Definition 543.73 and Definition 543.84. The route reaches the local NameCert row only after the socket, request, refusal gate, ledger, transport, continuation, and provenance rows. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary socket provenance`

Theorem 543.86 (ApophaticName refusal gate route totality)**.** *Every accepted refusal route for* $\mathsf{ApophaticName}^{\uparrow}$ *is exhausted by the displayed socket, requested supply shape, refusal gate, ledger,* Pkg *provenance, and* NameCert *rows, together with the recorded* hsame *and* Cont *routes.*

Proof. Use Definition 543.2 and Definition 543.51. They identify the permitted refusal route with S, R, G, L, H, C, P, N.

Then apply Definition 543.59. Since no supplied object or hidden theorem is a coordinate, the displayed rows exhaust the route. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal gate route totality`

Theorem 543.87 (ApophaticName downstream nonimport)**.** *Downstream consumers may cite the named externality of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet only through its public boundary, and cannot import the refused supply as an internal BEDC object.*

Proof. Apply Definition 543.77 and Definition 543.78. A downstream citation receives only the refusal packet and not the requested supply.

The non-escape certificate Definition 543.6 excludes generated witnesses, host data, quotient representatives, hidden theorems, and assumed propositions from the packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier downstream nonimport`

Theorem 543.88 (ApophaticName bridge-route scope)**.** *The public refusal-boundary packet S, R, G, L, H, C, P, N supplies the paper bridge surface from* $\mathsf{ApophaticName}^{\uparrow}$ *to downstream named-externality consumers. The bridge route exports socket naming, requested supply shape, refusal gate, refusal ledger, componentwise transport, continuation replay, package provenance, and local* NameCert *rows, while importing no supplied object, generated witness, hidden theorem, quotient representative, host datum, or assumed proposition.*

Proof. Begin with the downstream nonimport theorem above. It states that downstream consumers may cite the named externality only through the public boundary and cannot import the refused supply as internal BEDC content.

Use Definition 543.86 and Definition 543.84. They identify the exported route with the displayed socket, request, gate, ledger, transport, continuation, provenance, and local naming rows.

Finally apply Definition 543.6. Since generated witnesses, hidden theorems, quotient representatives, host data, and assumed propositions are excluded from the packet, the bridge surface is a public refusal-boundary route rather than an imported object. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier bridge route scope`

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticName}^{\uparrow}$ packet is generated from a finite socket-name row, a requested supply-shape row, a refusal-gate row, a refusal ledger, componentwise transport, continuation routes, package provenance, and a local naming certificate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 543.1, Definition 543.44, Definition 543.46, Definition 543.49, Definition 543.50, Definition 543.52, and Definition 543.54 bind the mature paper surface as the finite socket and refusal tuple, image row, readback row, componentwise NameCert transport, classifier stability, public boundary package, and refusal audit package for named externality.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not supply the external dependency, prove its existence, internalize it as an axiom or quotient, or close a Lean implementation.

Upgrade path: The paper axis is closed at the mature finite socket and refusal surface; machine verification and standard bridge checking remain formal-axis obligations for the same public boundary route.

Theorem 543.89 (ApophaticName root citation safety)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every root consumer citation of the named externality may cite only the visible socket, request, refusal gate, refusal ledger, transport, continuation, provenance, and local naming surface S, R, G, L, H, C, P, N. Such a citation cannot cite a generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle outside the visible refusal packet.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root citation safety`

Proof. Start with the public boundary package of Definition 543.52. It identifies the root-facing object with exactly the finite rows S, R, G, L, H, C, P, N.

Next apply socket citation scope and consumer-boundary exhaustion, Definition 543.53 and Definition 543.59. These keep every downstream citation on the socket, request, refusal, transport, continuation, provenance, and naming rows.

Finally use the non-escape certificate, Definition 543.6. Since the accepted carrier has no coordinate for generated witnesses, host objects, quotient representatives, hidden theorems, supplied dependencies, assumed propositions, or closure principles, the root citation is safe exactly on the visible refusal surface. □

Theorem 543.90 (ApophaticName boundary request totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every boundary request for the named externality must expose the request row R before it exposes the refusal gate row G. The accepted route then continues through the refusal ledger L, transport row H, continuation row C, provenance row P, and local naming row N; it cannot bypass R by reading a generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary request totality`

Proof. Begin with visible request factorization, Definition 543.37. It sends every visible socket request through the displayed request row R as part of the packet S, R, G, L, H, C, P, N.

Use socket request gate exactness, Definition 543.40. The gate row G is read as the refusal response to that displayed request, so a boundary request cannot reach the gate before it has exposed R.

Finally combine source-refusal totality and ledger nonescape, Definition 543.51 and Definition 543.5. These route the accepted request through G, L, H, C, P, N and exclude every bypass through generated or host-side content. □

Theorem 543.91 (ApophaticName refusal route exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every refusal route for the named externality is exhausted by the visible rows G, L, H, C, P, N after the socket and request rows S, R have been exposed. No refusal route generates a witness, supplies a host object, selects a quotient representative, imports a hidden theorem, or invokes a closure principle outside the displayed packet.

Proof. First apply root-unblock refusal boundary and refusal ledger exactness, Definition 543.14 and Definition 543.43. They identify root use of the named externality with the displayed refusal gate and ledger rather than with supplied content.

Next use socket-gate route exhaustion, Definition 543.56. Any socket read that reaches the refusal side must pass through S, R, G, L, H, C, P, N, so the refusal route is already a finite route through the displayed rows.

Finally apply boundary citation exhaustion and downstream named-externality refusal, Definition 543.74 and Definition 543.77. They exhaust downstream citation by the same refusal packet and leave no route to generated witnesses, host objects, quotient representatives, hidden theorems, or closure principles. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal route exhaustion`

Theorem 543.92 (ApophaticName root source citation export)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

each root citation exported by the packet names only the visible socket, request, refusal gate, refusal ledger, transport, continuation, provenance, and local naming rows S, R, G, L, H, C, P, N*. The export does not expose a generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle outside that boundary packet.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root source citation export`

Proof. Start with root citation safety, Definition 543.89. It restricts every root consumer citation to the visible rows S, R, G, L, H, C, P, N.

Use boundary request totality, Definition 543.90, to identify the request row exported with the socket route. Then use refusal route exhaustion, Definition 543.91, to keep the refusal side on G, L, H, C, P, N after S, R have been exposed.

Finally apply Definition 543.52 and Definition 543.6. These boundary lemmas identify the exported packet with the finite refusal surface and exclude every off-boundary source from the citation export. □

Theorem 543.93 (ApophaticName root refusal-gate readback)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$*, a root refusal-gate readback must expose the request row* R *before the refusal gate row* G*. After* G *is read, the same readback exposes only the refusal ledger, provenance, and naming rows* L, P, N*, together with the transport and continuation rows* H, C *that replay the visible route.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal gate readback`

Proof. Apply boundary request totality, Definition 543.90. It places the request row R before any exposure of the refusal gate G.

Next use refusal route exhaustion, Definition 543.91. Once G is public, the remaining refusal route is exhausted by L, H, C, P, N.

The readback is still a root citation. Therefore Definition 543.89 and Definition 543.92 prevent the readback from adding any generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle. □

Theorem 543.94 (ApophaticName root boundary ledger package)**.** *The downstream root boundary package for an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet* $A = (S, R, G, L, H, C, P, N)$ *consists exactly of root citation safety, boundary request totality, refusal-route exhaustion, root source citation export, and root refusal-gate readback. A downstream root consumer may therefore cite the socket row* S*, require the request row* R *before the refusal gate row* G*, and read only the ledger, transport, continuation, provenance, and naming rows* L, H, C, P, N *afterward.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root boundary ledger package`

Proof. First collect the three base rows. Apply Definition 543.89, Definition 543.90, and Definition 543.91; together they give citation safety, request-before-gate order, and refusal-route exhaustion on S, R, G, L, H, C, P, N.

Next add the two export rows. The root source citation export of Definition 543.92 keeps downstream citations on the visible packet, and the refusal-gate readback of Definition 543.93 fixes the required R before G order for gate reads.

Finally use Definition 543.52 and Definition 543.77. They identify the downstream boundary with this finite refusal packet and leave no consumer-facing route outside the displayed ledger package. □

Theorem 543.95 (ApophaticName root citation readback package)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every root citation readback factors through exactly the finite socket, request, refusal-gate, refusal-ledger, transport, continuation, provenance, and local naming rows S, R, G, L, H, C, P, N*. The readback exposes the request row* R *before the refusal gate row* G*, and after the gate it reads only* L, H, C, P, N*.*

This package contains no generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle outside the displayed packet.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root citation readback package`

Proof. First apply root citation safety, Definition 543.89. It confines every root citation to the visible finite packet S, R, G, L, H, C, P, N and excludes off-packet witnesses or host-side objects.

Next use boundary request totality and refusal route exhaustion, Definition 543.90 and Definition 543.91. These place the request row R before the refusal gate row G and exhaust the post-gate route by L, H, C, P, N.

Then add root source citation export and root refusal-gate readback, Definition 543.92 and Definition 543.93. The export row keeps the cited source on the same visible packet, while the readback row records the required R before G order.

Finally collect the rows with Definition 543.94. The boundary ledger package is exactly the finite readback surface, so no generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, or closure principle can be added afterward. □

Corollary 543.96 (ApophaticName root citation safety strict obstruction)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

any root-facing obstruction carried by a displayed failure, refusal, separation, or countermodel-style row of the root citation package is strictly local to the visible route

$$S, R, G, L, H, C, P, N.$$

It blocks only the named converse or reuse attempt that would reread the exported root citation or refusal route as a generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, closure principle, or hidden external validation. It cannot be reinterpreted as a global impossibility claim, a host theorem, or an ambient validation result outside the accepted packet.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root citation safety strict obstruction`

Proof. Start with root source citation export, Definition 543.92. It says that every root citation exported by the accepted packet names only the socket, request, refusal gate, refusal ledger, transport, continuation, provenance, and local naming rows S, R, G, L, H, C, P, N. Thus a consumer-facing citation has no exported coordinate for a generated witness, host object, quotient representative, hidden theorem, supplied dependency, assumed proposition, closure principle, or hidden external validation.

Now apply refusal route exhaustion, Definition 543.91. Once the socket and request rows S, R have been exposed, every refusal route is exhausted by G, L, H, C, P, N. Therefore any displayed failure, refusal, separation, or countermodel-style row that a root consumer can cite is read as part of that same refusal route, not as a second source of theorem content or host-side evidence.

The strict obstruction is the separation between a permitted citation and a barred rereading. In the permitted direction, the consumer may cite the exported source through S, R, G, L, H, C, P, N. In the barred converse or reuse direction, the consumer tries to turn that cited route back into supplied content, a host theorem, a quotient representative, a hidden theorem, an assumed proposition, a closure principle, or hidden external validation. The export theorem removes every off-packet citation coordinate, while route exhaustion removes every post-gate escape from the refusal rows.

Finally, root citation safety and the readback package, Definition 543.89 and Definition 543.95, keep this conclusion at the root boundary. They identify the public readback surface with the finite packet and do not add any claim about all ambient hosts or all possible external theories. Hence the obstruction blocks exactly the named converse or reuse route through the accepted packet and carries no global impossibility theorem, host theorem, or hidden external validation beyond S, R, G, L, H, C, P, N. □

Theorem 543.97 (ApophaticName root-unblock public citation surface)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every downstream consumer of the packet factors through the public socket row S, request row R, refusal gate G, refusal ledger L, transport row H, continuation row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N. The consumer may cite this boundary package as a root-unblock surface, but it exports no generated witness, host datum, quotient representative, hidden theorem, assumed proposition, supplied object, or closure principle.*

Proof. Start with the root-unblock refusal boundary and downstream boundary package, Definition 543.14 and Definition 543.10. Together they put root use of the named externality on the finite refusal packet S, R, G, L, H, C, P, N.

Next use the boundary citation universal factorization, Definition 543.126. It makes every downstream citation pass through the displayed socket, request, refusal, ledger, transport, continuation, provenance, and local naming rows before any consumer can treat the packet as accepted.

The sibling surfaces chapter 536 and chapter 715 identify the socket and refusal gate as public boundary components. They therefore allow only citation of the finite rows and do not supply an internal object behind the socket.

Thus the public root-unblock citation surface is exactly the displayed package. Since no coordinate of that package is a generated witness, host datum, quotient representative, hidden theorem, assumed proposition, supplied object, or closure principle, no such item is exported to downstream consumers. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root unblock public citation surface`

Theorem 543.98 (ApophaticName root public citation selector exclusion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

any root public citation that factors through S, R, G, L, H, C, P, N *cannot acquire a generated selector, supplied object, closure principle, quotient representative, hidden theorem, or host datum unless that item is already one of the displayed rows of the packet.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root public citation selector exclusion`

Proof. Start with Definition 543.97. It identifies the root public citation surface with the displayed packet S, R, G, L, H, C, P, N and says downstream consumers receive only that boundary package.

Use the sibling surfaces cited there. Definition 543.14 and Definition 543.10 keep the root route on the refusal packet, while Definition 543.126 forces each citation through the public socket, request, refusal, ledger, transport, continuation, provenance, and naming rows.

The sibling chapters chapter 536 and chapter 715 supply only public boundary components. Since none of the displayed coordinates is an undisplayed selector, supplied object, closure principle, quotient representative, hidden theorem, or host datum, the root citation cannot acquire such an item outside the packet. □

Theorem 543.99 (ApophaticName two-step refusal-ledger induction)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

any downstream root citation admitted by the packet is admitted by a two-step finite induction. The first step exposes S, R *as the socket and request boundary. The second step exposes* G, L *as the refusal gate and refusal ledger. After those two steps,* H, C, P, N *transport, replay, source, and name only that refusal package. The resulting citation surface excludes generated witnesses, supplied objects, quotient representatives, hidden theorems, host data, assumed propositions, and closure principles outside the displayed packet.*

Proof. Start with the public citation surface of Definition 543.97. It identifies every downstream root citation with the finite packet S, R, G, L, H, C, P, N and gives the consumer only that public boundary package.

Use Definition 543.98 to remove the hidden choices that could make the citation look like an internal supply route. In particular, no generated selector, supplied object, closure principle, quotient representative, hidden theorem, or host datum can be acquired outside the displayed rows.

Now read the package in two finite steps. First expose the socket and request rows S, R; the sibling chapter chapter 536 identifies this pair as a public socket boundary rather than a supplied object.

Second expose the refusal gate and ledger rows G, L; the sibling chapter chapter 715 identifies the gate read as a public refusal question rather than a hidden host theorem or assumed proposition. The remaining rows H, C, P, N then only transport, replay, source, and name the same refusal package.

Thus the admitted downstream citation has the claimed induction shape: socket-request boundary first, refusal-gate ledger second, and only transport, replay, source, and naming afterward. Since both cited surface theorems exclude off-packet selectors and supplies, the citation cannot export generated witnesses, supplied objects, quotient representatives, hidden theorems, host data, assumed propositions, or closure principles. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger induction surface`

Theorem 543.100 (ApophaticName root consumer socket request totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every root consumer of the named externality passes through the displayed socket row S*, request row* R*, refusal gate* G*, refusal ledger* L*, transport row* H*, continuation row* C*, provenance row* P*, and naming row* N*. The consumer therefore receives a boundary request and refusal certificate, not an internal supplied object, host proof, quotient representative, hidden theorem, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameRootConsumerSocketRequestTotality`

Proof. Start from the root unblock refusal boundary, Definition 543.14. It already places root use of the named externality on the refusal boundary and prevents a bypass into generated content.

Next apply visible request factorization and socket request gate exactness, Definition 543.37 and Definition 543.40. These results identify the actual root-facing read as the finite socket/request/refusal package S, R, G, L, H, C, P, N.

Finally use the sibling socket and gate surfaces, Definition 536.5 and Definition 715.3. They expose the socket and refusal gate as public boundary rows only, so the root consumer must pass through the displayed packet. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root consumer socket request totality`

Theorem 543.101 (ApophaticName refusal ledger downstream coverage)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the refusal ledger L *covers every downstream read of the named externality. A downstream chapter may cite only the socket/request/refusal/provenance boundary* S, R, G, L, H, C, P, N*, and the ledger bars any internal supply of generated content, assumed propositions, quotient representatives, hidden theorems, host realizations, or closure principles.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger downstream coverage`

Proof. Apply the refusal ledger theorem, Definition 543.2. It records the permitted refusal rows and excludes the internal supply cases at the source of the named externality.

The downstream side is fixed by downstream boundary exactness and the consumer citation boundary, Definition 543.35 and Definition 543.41. They say that every downstream citation remains a citation of the same finite packet.

Use request ledger classifier exhaustion, Definition 543.38. Since all permitted boundary reads are classified by R, G, L, there is no separate internal row through which a downstream reader can obtain supplied content. □

Theorem 543.102 (ApophaticName socket boundary route exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every socket-boundary route for the named externality factors through the displayed finite packet S, R, G, L, H, C, P, N*. The route may use the socket, request, refusal, transport, continuation, provenance, and naming rows, but it has no additional carrier for generated content, assumed propositions, quotient representatives, hidden theorems, host realizations, or closure principles.*

Proof. First factor the route through the visible request surface using Definition 543.37. This places every socket-boundary route on the packet coordinates S, R, G, L, H, C, P, N.

Next apply componentwise transport and the named externality package, Definition 543.39 and Definition 543.42. They keep transport, continuation, provenance, and naming attached to the same socket/request/refusal rows.

Finally use externality handoff and root consumer request totality, Definition 543.30 and Definition 543.100. Any route seen by downstream or root consumers is therefore the displayed finite packet, with no additional supply route outside it. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameSocketBoundaryRouteExhaustion`

Theorem 543.103 (ApophaticName root ledger coverage)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the root-facing ledger rows G, L cover every downstream package read of the named externality. A downstream consumer may receive the socket row S, request row R, refusal gate G, refusal ledger L, componentwise transport row H, continuation row C, provenance row P, and local naming row N, but no generated content, assumed proposition, quotient representative, hidden theorem, host realization, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root ledger coverage`

Proof. Start from the refusal ledger, Definition 543.2. It records the permitted socket and request reads together with the barred conversions governed by G and L.

Apply the root-unblock refusal boundary and root consumer socket request totality, Definition 543.14 and Definition 543.100. These two results put every root citation of the named externality on the same finite packet S, R, G, L, H, C, P, N.

Finally use downstream boundary exactness and downstream noninternalization, Definition 543.35 and Definition 543.13. The downstream package therefore consumes only the ledgered boundary rows, so the barred internal objects have no receiving coordinate. □

Theorem 543.104 (ApophaticName root citation nonescape)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

each root citation exported to a downstream chapter is the finite boundary citation S, R, G, L, H, C, P, N. The citation cannot be re-read as a supplied object, host proof, quotient representative, hidden theorem, generated witness, or closure principle.

Proof. Use the consumer citation boundary, Definition 543.41. It already states that downstream citation of the accepted packet factors through the displayed boundary package and exports no supplied object.

Next apply visible request factorization and socket boundary route exhaustion, Definition 543.37 and Definition 543.102. They force the cited socket to carry the request, refusal, transport, continuation, provenance, and naming rows on the same route.

The nonescape part is supplied by ledger nonescape and downstream ledger nonescape, Definition 543.5 and Definition 543.28. Since no extra coordinate exists outside S, R, G, L, H, C, P, N, a root citation remains a boundary citation at every downstream use site. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root citation nonescape`

Theorem 543.105 (ApophaticName root boundary package exhaustion)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the three downstream package rows for root consumption are exhausted by ledger coverage, citation nonescape, and socket-boundary route exhaustion. Thus any downstream chapter consuming $\mathsf{ApophaticName}^{\uparrow}$ *receives exactly the boundary package* S, R, G, L, H, C, P, N *and no additional internal socket, hidden supply, host realization, quotient representative, assumed proposition, theorem cache, or closure row.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root boundary package exhaustion`

Proof. First consume root ledger coverage from Definition 543.103. It fixes G, L as the rows that answer every downstream package read of the named externality.

Next consume root citation nonescape from Definition 543.104. This keeps the exported citation on the same packet and prevents it from becoming a supplied object or host-side proof.

Finally combine socket-boundary route exhaustion with the named externality NameCert package, Definition 543.102 and Definition 543.42. Together they say that the only package available to downstream root consumers is the finite boundary package S, R, G, L, H, C, P, N, with all barred internal objects excluded. □

Theorem 543.106 (ApophaticName root bridge tuple source)**.** *Every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple exposes the socket row* S*, gate-question row* R*, externality refusal row* G*, refusal ledger* L*,* BHist *history rows, componentwise* hsame *row* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N *before any downstream refusal read. No consumer receives a root bridge tuple whose first readable coordinate is an internalized object, generated witness, host datum, quotient representative, hidden theorem, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple source`

Proof. Start from root boundary package exhaustion, Definition 543.105. It states that downstream root consumers receive exactly the finite packet S, R, G, L, H, C, P, N.

Apply the refusal ledger and root ledger coverage rows, Definition 543.2 and Definition 543.103. They force each refusal read to pass through G, L after the visible socket and request rows have already been displayed.

Use the sibling socket and gate surfaces, Definition 536.5 and Definition 715.3. These identify the socket and gate-question sides as public boundary rows, not supplied objects.

The structural rows H, C, P, N replay that same boundary packet through componentwise transport, continuation, provenance, and local naming. Thus the root bridge tuple exposes all source rows before any refusal read and exports no hidden supply channel. □

Theorem 543.107 (ApophaticName root bridge tuple classifier)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple, classifier comparison is exhausted by the displayed socket row* S*, gate-question row* R*, refusal gate* G*, refusal ledger* L*, transport row* H*, route row* C*, provenance row* P*, and local naming row* N*. The classifier cannot compare against hidden supply, host realization, quotient representative, generated witness, theorem cache, or closure row outside the tuple.*

Proof. Project the finite tuple from Definition 543.106. The source, gate, refusal, transport, route, provenance, and naming rows are already present before classifier comparison begins.

Use classifier stability and request ledger classifier exhaustion, Definition 543.50 and Definition 543.38. They keep comparison on the displayed socket/request/refusal rows and on their componentwise transports.

Apply the visible request and socket-boundary route rows, Definition 543.37 and Definition 543.102. These rows leave no second comparison source behind the public boundary citation.

The sibling gate chapter, Definition 715.3, supplies only the visible refusal question. Therefore classifier comparison is exhausted by the root bridge tuple itself. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple classifier`

Theorem 543.108 (ApophaticName root bridge tuple ledger)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple reads the refusal ledger L and the non-internalization rows before any externality handoff. The handoff may carry the socket, gate-question, refusal,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, but it cannot deliver a supplied dependency, internal object, host proof, quotient representative, hidden theorem, generated witness, or closure principle.*

Proof. Begin with root ledger coverage and root citation nonescape, Definition 543.103 and Definition 543.104. They make G, L the rows that answer downstream package reads and keep exported citations on the same finite packet.

Next apply downstream noninternalization and downstream nonimport, Definition 543.13 and Definition 543.87. These rows bar a downstream consumer from treating the requested supply as internal content.

Use the externality handoff theorem, Definition 543.30. Its handoff object is the displayed refusal boundary, replayed through H, C, P, N.

Combining these rows with the source and classifier bridge tuple theorems above shows that every downstream consumer reads the ledger and non-internalization boundary before the externality handoff, with no hidden supply coordinate left to export. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple ledger`

Theorem 543.109 (ApophaticName root bridge tuple noninternalization)**.** *Every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple must keep the socket row S, gate-question row R, refusal row G, transport row H, route row C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row N as boundary rows. The consumer may read the refusal ledger L through the same packet, but it cannot internalize the requested supply as an object, generated witness, host datum, quotient representative, hidden theorem, or closure principle.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple noninternalization`

Proof. Start with the source, classifier, and ledger rows above. Theorems 543.106, 543.107, and 543.108 expose the tuple as S, R, G, L, H, C, P, N and exhaust both comparison and downstream ledger reads on those displayed coordinates.

Use the sibling socket and gate nonescape surfaces. Theorems 536.5 and 715.3 keep the socket and gate-question sides visible as boundary rows rather than supplied content.

Finally apply the downstream noninternalization and externality handoff rows, Definition 543.13 and Definition 543.30. They keep the refusal answer on the displayed transport, route, provenance, and local naming rows, so no downstream consumer can reclassify the requested supply as an internal object, generated witness, host datum, quotient representative, hidden theorem, or closure principle. □

Theorem 543.110 (ApophaticName root bridge tuple source-route totality)**.** *Every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple exhausts its source route by the rows*

$$S, R, G, L, H, C, P, N$$

before any downstream refusal read. The downstream consumer may inspect the socket, gate-question, refusal, ledger, transport, route, provenance, and local naming rows, but it receives no supplied dependency, internal object, generated witness, host datum, quotient representative, hidden theorem, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple source route totality`

Proof. Start with the source and classifier tuple theorems, Definition 543.106 and Definition 543.107. They expose the tuple before downstream reading as the finite socket, gate-question, refusal, ledger, transport, route, provenance, and naming surface.

Next apply the ledger theorem and noninternalization theorem, Definition 543.108 and Definition 543.109. These force the refusal read through G, L while keeping S, R, H, C, P, N as boundary rows rather than supplied content.

Use the sibling socket and gate surfaces, Definition 536.5 and Definition 715.3. The source side is therefore already public as a socket and visible gate question before any downstream refusal consumer can ask for an externality handoff. □

Theorem 543.111 (ApophaticName root bridge tuple gate-refusal interface)**.** *Every externality handoff from an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple must read the gate-question row* R *and refusal rows* G, L *before the handoff is public. The interface is the finite route through* S, R, G, L, H, C, P, N *and not a supplied object, host proof, generated witness, quotient representative, hidden theorem, or closure principle.*

Proof. Begin with source-route totality from Definition 543.110. It fixes the handoff source route as S, R, G, L, H, C, P, N before downstream refusal reading begins.

Apply the root bridge tuple ledger theorem, Definition 543.108. It makes L and the noninternalization rows part of the route before the externality handoff, so the refusal answer is not read after the handoff as hidden content.

Finally use the sibling gate surface, Definition 715.3, together with the externality handoff theorem cited in Definition 543.108. The gate question and refusal are read as visible boundary rows, and the handoff carries only the displayed tuple. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple gate refusal interface`

Theorem 543.112 (ApophaticName root bridge tuple transport-naming exhaustion)**.** *For every downstream consumer of an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple, the transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *are exhausted as boundary infrastructure for the route* S, R, G, L, H, C, P, N*. They cannot be reread as supplied dependency content, an internal object, a generated witness, host datum, quotient representative, hidden theorem, or closure principle.*

Proof. First combine the source, classifier, ledger, and noninternalization tuple theorems above. Together they show that H, C, P, N only replay the already displayed socket, gate-question, refusal, and ledger rows.

Next use the gate-refusal interface of Definition 543.111. Since the externality handoff has already consumed R, G, L through the finite route, the structural rows cannot be reopened as a second source of supplied content.

Finally apply the sibling socket and gate surfaces, Definition 536.5 and Definition 715.3. They keep the source and gate sides visible, so transport, continuation, provenance, and local naming remain route witnesses rather than downstream payload. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple transport naming exhaustion`

Theorem 543.113 (ApophaticName root bridge tuple terminal exhaustion)**.** *Every downstream externality handoff from an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple is exhausted by the finite rows* S, R, G, L, H, C, P, N*. The handoff cannot export supplied dependency content, an internal object, a generated witness, host datum, quotient representative, hidden theorem, or closure principle outside those rows.*

Proof. Start with the source, classifier, ledger, and noninternalization tuple theorems. They expose the bridge tuple as S, R, G, L, H, C, P, N and force comparison, refusal reading, and downstream package reading through the displayed boundary rows.

Next apply Definition 543.111. The gate question and refusal ledger are consumed before the externality handoff is public, so the handoff cannot reopen G or L as supplied content.

Finally use Definition 543.112. The rows H, C, P, N only replay transport, continuation, provenance, and local naming for the same finite route. Therefore every downstream externality handoff is exhausted by S, R, G, L, H, C, P, N and has no hidden export coordinate. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple terminal exhaustion`

Theorem 543.114 (ApophaticName downstream refusal uniqueness)**.** *Every downstream externality handoff from an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple has a unique refusal-reading route through*

$$S, R, G, L, H, C, P, N.$$

If a second downstream route has the same socket row S *and gate-question row* R*, then it agrees on the refusal row* G*, refusal ledger* L*, transport row* H*, continuation route* C*, provenance row* P*, and local naming row* N*. Neither route can expose supplied content, host proof, quotient representative, hidden theorem, closure principle, internalized object, or generated witness.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple downstream refusal uniqueness`

Proof. First fix the source side. Apply Definition 543.110 to any downstream externality handoff. The handoff begins on the displayed socket and gate-question rows and then has only the route S, R, G, L, H, C, P, N available before refusal reading.

Next read the refusal interface. By Definition 543.111, the same socket and gate-question rows force the handoff through the same refusal gate and ledger rows G, L before the handoff is public.

Now compare the structural tail. The transport-naming exhaustion theorem Definition 543.112 says that H, C, P, N only replay the displayed refusal boundary. Therefore a second route with the same S and R has no alternative transport, continuation, provenance, or local naming row to select.

Finally apply terminal exhaustion, Definition 543.113, together with the sibling socket and gate surfaces Definition 536.5 and Definition 715.3. These keep both routes on the visible boundary

rows and exclude supplied content, host proof, quotient representatives, hidden theorems, closure principles, internalized objects, and generated witnesses. □

Theorem 543.115 (ApophaticName root bridge tuple audit normal form)**.** *Every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple has audit normal form exactly*

$$S, R, G, L, H, C, P, N.$$

The source and classifier read the socket, gate-question, and refusal rows; the ledger and noninternalization rows prevent supplied-content export; and terminal exhaustion leaves no hidden supply, quotient representative, generated witness, host datum, theorem cache, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple audit normal form`

Proof. First expose the source and classifier rows. Apply Definition 543.106 and Definition 543.107; both reads factor through S, R, G, L, H, C, P, N and never through hidden supply or host realization data.

Next bind the ledger and noninternalization boundary. Use Definition 543.108 and Definition 543.109. These rows force downstream package reads through the refusal ledger and prevent the requested supply from becoming an internal object, generated witness, host datum, quotient representative, hidden theorem, or closure principle.

Now assemble the route interface. The source-route, gate-refusal, and transport-naming rows in Definition 543.110, Definition 543.111, and Definition 543.112 keep H, C, P, N attached to the same visible socket, gate, refusal, and ledger rows.

Finally apply terminal exhaustion and downstream refusal uniqueness, Definition 543.113 and Definition 543.114, together with the sibling ApophaticSocket and ApophaticGateQuestion surfaces Definition 536.5 and Definition 715.3. No second route can add supplied content, quotient representatives, generated witnesses, host data, theorem caches, or closure principles, so the audit normal form is exactly S, R, G, L, H, C, P, N. □

Corollary 543.116 (ApophaticName root bridge tuple source strict obstruction)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *root bridge tuple, the obstruction carried by the gate-question and refusal rows is strictly local to the displayed route*

$$S, R, G, L, H, C, P, N.$$

It blocks exactly the downstream converse or reuse attempt that would reread the handoff source as an internalized object, supplied dependency, generated witness, host datum, quotient representative, hidden theorem, closure principle, or hidden external validation. It does not assert a global impossibility theorem, a host theorem, or an ambient validation result outside the accepted tuple.

Proof. Start with the tuple source, Definition 543.106. It exposes S, R, G, L, H, C, P, N before any downstream refusal read and states that no consumer receives a tuple whose first readable coordinate is an internalized object, generated witness, host datum, quotient representative, hidden theorem, or closure principle. Thus the only source available to a consumer-facing obstruction is the displayed finite tuple.

Next apply the gate-refusal interface, Definition 543.111. It forces every externality handoff to read the gate-question row R and refusal rows G, L before the handoff is public, and it identifies the interface as the same route S, R, G, L, H, C, P, N rather than a supplied object, host proof, generated witness, quotient representative, hidden theorem, or closure principle.

The strictness follows by separating the two directions. In the permitted direction, a downstream consumer may cite the socket, gate-question, refusal, ledger, transport, continuation, provenance, and local naming rows as the public source of the handoff. In the barred converse direction, the same consumer tries to turn that handoff back into supplied content or to reuse H, C, P, N as a second validation channel. The source theorem leaves no first coordinate for such a rereading, and the gate-refusal interface has already consumed R, G, L as refusal rows before public handoff.

Therefore the obstruction blocks only that named converse or reuse route through the accepted root bridge tuple. Since every cited row remains inside S, R, G, L, H, C, P, N, the proof supplies no statement about all possible ambient hosts, no theorem object outside the tuple, and no hidden external validation beyond the local refusal boundary. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge tuple source strict obstruction`

Theorem 543.117 (ApophaticName root StdBridge premise surface)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the fixed finite tuple route reaches a root-facing StandardBridge *request only after four displayed rows are present: the boundary-request image row, the refusal readback row, the componentwise* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *transport row, and the* ChapterTasteGate*-facing refusal tuple. Thus the bridge premise is the public tuple surface* S, R, G, L, H, C, P, N *together with those rows, not an internal object, host proof, quotient representative, hidden theorem, generated witness, or closure principle.*

Proof. First apply the boundary-request image row, Definition 543.44. It confines each accepted consumer image of the request to the finite tuple S, R, G, L, H, C, P, N.

Next use the refusal readback row, Definition 543.46. It sends every accepted image-row read back to the corresponding finite BHist component and prevents the readback from becoming supplied content.

Now apply componentwise NameCert transport, Definition 543.49. The local naming row replays the image and readback rows across the displayed transport route without adding a hidden coordinate.

The ChapterTasteGate-facing refusal tuple is therefore exactly the same displayed packet: socket, request, refusal gate, ledger, transport, continuation, provenance, and local naming. A root-facing StandardBridge request can consume that premise surface only after all these rows are visible. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root stdbridge premise surface`

Theorem 543.118 (ApophaticName root reflection-axis refusal)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the root tuple admits only the reflection-axis rows carried by the refusal boundary. It supplies the socket row S*, request row* R*, refusal gate* G*, refusal ledger* L*, componentwise transport* H*, continuation route* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *row* N*, but it supplies no object, host proof, quotient representative, hidden theorem, generated witness, or closure principle outside that boundary.*

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root reflection axis refusal`

Proof. Begin with the root bridge tuple source, Definition 543.106. It exposes all source rows before any downstream refusal read and rules out a first readable coordinate that is already an internalized object.

Use the root bridge tuple ledger, Definition 543.108. The ledger and non-internalization rows must be read before any externality handoff, so the handoff remains a refusal package rather than a supplied dependency.

Finally apply boundary citation universal factorization, Definition 543.126. Every downstream citation factors through the socket, gate, ledger, and provenance package S, R, G, L, H, C, P, N.

These three rows leave the reflection axis with the displayed refusal boundary only. There is no remaining carrier position from which a supplied object, host proof, quotient representative, hidden theorem, generated witness, or closure principle could be obtained. □

Theorem 543.119 (ApophaticName root bridge counter surface)**.** *After the boundary-request image row, refusal readback row, componentwise* $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ *transport row, and* ChapterTasteGate*-facing refusal tuple have all been displayed for an accepted packet*

$$A = (S, R, G, L, H, C, P, N),$$

any failed bridge attempt must locate an incompatibility between that refusal-axis requirement and the standard reflection schema. Absence of the tuple, image row, readback row, componentwise transport row, or refusal row is only a failed premise for the fixed finite tuple route, not a bridge counter-surface.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root bridge counter surface`

Proof. Start from the root StdBridge premise surface, Definition 543.117. It identifies the tuple, image, readback, componentwise transport, and refusal rows as the rows that must be present before the fixed finite tuple route is tested.

Next use root reflection-axis refusal, Definition 543.118. Once the rows are present, the only available reflection-axis content is the refusal boundary itself, with no supplied object or hidden theorem behind it.

Therefore a failed bridge attempt after these rows are displayed must be a disagreement with the standard reflection schema at the refusal axis. If one of the tuple, readback, transport, or refusal rows is absent, the route has not reached the bridge test and the failure is merely a missing premise. □

Corollary 543.120 (ApophaticName root StdBridge premise strict obstruction)**.** *Let* $A = (S, R, G, L, H, C, P, N)$ *be an accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet. Any displayed failure, refusal, separation, or countermodel row used against the root-facing* StandardBridge *premise is a strict local obstruction. It blocks only the named converse or reuse route whose consumer read tries to turn the displayed premise surface into an internal object, host proof, quotient representative, hidden theorem, generated witness, closure principle, ambient countermodel, or hidden external validation. It cannot be reinterpreted as a global impossibility theorem, a host theorem, or an external validation result outside the accepted packet.*

Proof. Start with the root StdBridge premise surface, Definition 543.117. A root-facing StandardBridge request is not reached from an unstructured object: it is reached only after four displayed rows are visible, namely the boundary-request image row, the refusal readback row, the componentwise $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ transport row, and the ChapterTasteGate-facing refusal tuple. Thus any

consumer-facing use of a displayed negative row against this premise must first consume the finite premise surface S, R, G, L, H, C, P, N together with those four rows.

Apply the root reflection-axis refusal theorem, Definition 543.118. Its source, ledger, and citation factorization rows leave the reflection axis with only the socket row S, request row R, refusal gate G, refusal ledger L, componentwise transport H, continuation route C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{ApophaticName}^{\uparrow}}$ row N. The same theorem excludes object supply, host proof, quotient representative, hidden theorem, generated witness, and closure principle outside that refusal boundary.

Now separate the permitted forward use from the barred converse or reuse use. In the permitted direction, the consumer may cite the visible boundary-request image, readback, transport, and refusal-tuple rows as the premise surface for the root-facing bridge request. In the barred converse direction, the consumer tries to reread that same surface as a stronger coordinate: an internal object behind S, a supplied dependency behind R, a host proof or hidden theorem behind G and L, a quotient representative or generated witness behind the readback, a closure principle behind H and C, or an external validation channel behind P and N. The premise-surface theorem gives no such coordinate, and the reflection-axis refusal theorem says that the only reflection-axis content is the displayed refusal boundary.

Therefore a displayed failure, refusal, separation, or countermodel row can obstruct exactly the named converse or reuse route attached to one of those displayed premise rows. If a proposed interpretation asks for a global impossibility theorem, a host theorem, an ambient countermodel detached from A, or hidden external validation, it asks for an unlisted coordinate. The accepted packet rejects that interpretation at the local refusal boundary, so the obstruction remains strict and local to the root StandardBridge premise surface. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root stdbridge premise strict obstruction`

Theorem 543.121 (ApophaticName root refusal ledger exactness)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the root-facing refusal rows G, L classify every barred conversion from socket request to generated content, assumed proposition, quotient representative, hidden theorem, host realization, or closure principle. A root consumer may cite the visible socket S and the request row R, but every such citation is answered by the displayed refusal gate and ledger, transported by H, routed by C, packaged by P, and named by N.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root refusal ledger exactness`

Proof. Start with the refusal ledger theorem, Definition 543.2. It records the barred internalization cases as refusal rows rather than as absent ambient assumptions.

Next apply gate ledger exactness, Definition 543.32. The gate row G and ledger row L are the rows that answer conversion attempts from the visible socket request into supplied content or hidden proof objects.

Finally use request ledger classifier exhaustion, Definition 543.38. Since every permitted boundary read of S is classified through R, G, L, no root read can leave the finite packet S, R, G, L, H, C, P, N and obtain one of the barred objects. □

Theorem 543.122 (ApophaticName root socket request totality)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every root read of the socket row S must also read the request row R, refusal gate G, refusal ledger L, transport row H, continuation row C, provenance row P, and local naming row N. Thus the root-facing socket is a complete request boundary and not a partial handle that can be completed by an external generated object, assumed proposition, quotient representative, hidden theorem, host realization, or closure principle.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameRootSocketRequestTotality`

Proof. Begin with root consumer socket request totality, Definition 543.100. It already identifies root consumption of the named externality with the complete socket/request/refusal package.

Use socket request gate exactness, Definition 543.40. This binds every visible read of S to the request row R, the gate row G, and the ledger row L.

The sibling socket and gate boundaries close the remaining routes. Apply Definition 536.5 to keep the socket citation visible, and apply Definition 715.3 to prevent the gate from turning into a supplied object. The transport, continuation, provenance, and naming rows then remain attached to the same packet. □

Theorem 543.123 (ApophaticName root boundary provenance export)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every downstream export from the root boundary receives only the finite provenance package S, R, G, L, H, C, P, N. The export may cite the socket request, refusal gate, refusal ledger, componentwise transport, continuation route, package provenance, and local name, but it does not export generated content, assumed propositions, quotient representatives, hidden theorems, host realizations, or closure principles.

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier root boundary provenance export`

Proof. First use the consumer citation boundary, Definition 543.41. It says that downstream citations of the accepted packet factor through the displayed boundary package and do not receive supplied objects.

Next apply the downstream noninternalization package, Definition 543.13. The same refusal package bars the internalization cases at the downstream interface.

Finally repack the rows with the named externality NameCert package, Definition 543.42. Carrier admission, gate exactness, ledger nonescape, transport, citation, provenance, and naming are all rows of S, R, G, L, H, C, P, N, so the exported provenance is exactly that finite package. □

Theorem 543.124 (ApophaticName socket gate ledger pullback)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the sibling $\mathsf{ApophaticSocket}^{\uparrow}$ *row and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *row pull back to the displayed request row R, refusal gate G, and refusal ledger L before any downstream read of the named externality. Thus the socket and gate consumers cite the same boundary packet S, R, G, L, H, C, P, N, and no downstream consumer receives a generated witness, host datum, quotient representative, hidden theorem, or closure principle from either sibling surface.*

Proof. Begin with socket request gate exactness, Definition 543.40. It binds every visible socket request for the accepted name to the displayed rows R, G, L.

The sibling socket row is compatible with this boundary by Definition 536.3. It identifies the socket consumer with the public gate interface rather than with an internal supplied object.

The gate-question row is forced onto the same name boundary by Definition 715.4. Hence both sibling surfaces pull back through R, G, L before downstream reading, and the remaining rows H, C, P, N only transport, route, package, and name that same refusal boundary. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier socket gate ledger pullback`

Theorem 543.125 (ApophaticName refusal ledger fiber product)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

the refusal ledger is the finite fiber product of the socket request row R and the visible refusal gate G over the common boundary data G, L, H, C, P, N. This fiber product exports only the boundary rows of the packet and exports no generated witness, host datum, quotient representative, hidden theorem, or closure principle.

Proof. First use root refusal ledger exactness, Definition 543.121. It makes G, L the rows that classify conversion attempts from the visible socket request into the barred internal objects.

Next apply root ledger coverage, Definition 543.103. Every downstream package read is already covered by the displayed ledger rows and their attached transport, continuation, provenance, and naming data.

Finally apply the socket gate ledger pullback, Definition 543.124. Since the sibling socket and gate rows meet on the same R, G, L boundary, the ledger is exactly the finite boundary fiber product over G, L, H, C, P, N, with no additional export coordinate. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier refusal ledger fiber product`

Theorem 543.126 (ApophaticName boundary citation universal factorization)**.** *For every accepted* $\mathsf{ApophaticName}^{\uparrow}$ *packet*

$$A = (S, R, G, L, H, C, P, N),$$

every downstream boundary citation of the named externality universally factors through the socket/gate/ledger pullback and then through the provenance package S, R, G, L, H, C, P, N. Consequently a downstream chapter may cite the accepted name only through this package and cannot factor the citation through generated content, host data, quotient representatives, hidden theorems, or closure principles.

Proof. Start with root boundary package exhaustion, Definition 543.105. It says that downstream root consumers receive exactly the finite packet S, R, G, L, H, C, P, N.

The provenance export theorem, Definition 543.123, identifies the exported object as the same finite provenance package and not as an internal supplied object.

Use socket gate ledger pullback and refusal ledger fiber product, Definition 543.124 and Definition 543.125. These two results give the common factorization stage for the sibling socket, visible gate, and ledger rows, so every downstream citation factors through that stage before it reaches the package S, R, G, L, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ApophaticNameUp.ApophaticNameCarrier boundary citation universal factorization`

Chapter 544

A Concrete Naming Certificate for $\mathsf{ZCarry}^{\uparrow}$

The $\mathsf{ZCarry}^{\uparrow}$ interface packages the displayed Zeckendorf carry row `word_011` $\to$ `word_100` as a finite BEDC naming surface. The packet records the carry generator, the source non-normality witness, the target normality witness, the non-hsame boundary, transport rows, continuation routes, package provenance, and the local naming row. It licenses the carry as a classifier and gives no route to treating the two histories as kernel-equal.

Definition 544.1 (ZCarry finite carry packet)**.** *A* $\mathsf{ZCarry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (h, k, G, N_h, N_k, H, C, P, N)$$

with the following displayed coordinates:

1. *h is the source history row* `word_011`*;*
2. *k is the target history row* `word_100`*;*
3. *G is the generated carry row* $\mathsf{ZCarry}(h, k)$*;*
4. *N_h is the source ledger row witnessing* $\neg\mathsf{ZNormal}(h)$*;*
5. *N_k is the target ledger row witnessing* $\mathsf{ZNormal}(k)$*;*
6. *H records the non-*hsame *boundary* $\neg\mathsf{hsame}(h, k)$ *together with the componentwise* hsame *transports that preserve the displayed rows separately;*
7. *C records the displayed* Cont *routes for source inspection, target inspection, carry readback, normality readback, non-*hsame *readback, and NameCert comparison;*
8. *P is the* Pkg *provenance row naming* $\mathsf{ZeckendorfNormal}^{\uparrow}$*,* $\mathsf{ZCarry}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, and the generated carry row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only by the displayed rows $h, k, G, N_h, N_k, H, C, P, N$. No coordinate identifies h and k, quotienting histories or replacing the carry classifier by hsame*.*

Lean4 (definition) `BEDC.Derived.ZCarryUp.ZCarryFiniteCarryPacket`

Theorem 544.2 (ZCarry finite NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *obligation surface consists exactly of carrier admission by Definition 544.1, the generated carry row G, the source*

*non-normal row N_h, the target normal row N_k, the non-*hsame *boundary row H, the displayed* Cont *routes C, the* Pkg *provenance row P, and the local naming row N.*

Proof. Project the finite packet. Each row required by the carry certificate appears as a coordinate before any consumer can read the classifier.

The generator row G fixes the only displayed carry window. The two normality rows record the source and target sides of that window, while H records that the window is not an hsame identification.

The routes C, provenance row P, and local naming row N then repack the same finite surface as a $\mathsf{NameCert}_{\mathsf{ZCarry}^\uparrow}$ obligation surface. Since the carrier has no equality or quotient coordinate, no further object can be exported by the certificate. □

Lean4 ✓ `BEDC.Derived.ZCarryUp.ZCarryFiniteCarryPacket obligation surface`

Theorem 544.3 (ZCarry carry-not-hsame boundary)**.** *For every accepted* $\mathsf{ZCarry}^\uparrow$ *packet* $Z = (h, k, G, N_h, N_k, H, C, P, N)$*, the carry row G is stable only as a classifier row. Its ledger includes* $\neg\mathsf{hsame}(h, k)$*, so the displayed carry from* `word_011` *to* `word_100` *cannot be consumed as history-sameness.*

Proof. Read the generated carry row G. By the Zeckendorf carry theorem Definition 9345.4, every instance of $\mathsf{ZCarry}(h, k)$ carries the boundary $\neg\mathsf{hsame}(h, k)$.

The packet stores this boundary in H and routes it through C. Therefore a consumer can transport the carry classifier only together with the recorded non-hsame witness, so the carry row cannot become a kernel equality. □

Theorem 544.4 (ZCarry normal-target ledger)**.** *For every accepted* $\mathsf{ZCarry}^\uparrow$ *packet* $Z = (h, k, G, N_h, N_k, H, C, P, N$ *the target history is normal and the source history is non-normal. The target row is* `word_100` *with* $N_k : \mathsf{ZNormal}(k)$*, while the source row is* `word_011` *with* $N_h : \neg\mathsf{ZNormal}(h)$*.*

Proof. The generator row G identifies the displayed carry window. Applying Definition 9345.5 reads the target normality witness, and applying Definition 9345.6 reads the source non-normality witness.

The packet stores those two reads as N_k and N_h. Transport through H and C preserves the target and source ledgers as separate rows, so normality can be reported only with the same displayed source-target orientation. □

Theorem 544.5 (ZCarry Zeckendorf normal sibling dependency)**.** *For every accepted* $\mathsf{ZCarry}^\uparrow$ *packet* $Z = (h, k, G, N_h, N_k, H, C, P, N)$*, the source non-normal row N_h and target normal row N_k are read through the Zeckendorf normal-form sibling surface of chapter 278 before the local carry packet assembles its* $\mathsf{NameCert}_{\mathsf{ZCarry}^\uparrow}$ *boundary. The* $\mathsf{ZCarry}^\uparrow$ *packet supplies the generated carry classifier, the non-*hsame *ledger, and the local transport, continuation, provenance, and naming rows; it does not duplicate the sibling normal-word evidence.*

Proof. Project the accepted packet as $h, k, G, N_h, N_k, H, C, P, N$. The two normality rows are separate fields of the local carrier, while G and H record the carry classifier and the non-hsame boundary.

Apply Definition 544.4. It reads N_k as the target normal row and N_h as the source non-normal row with the displayed source-target orientation.

The normal-word evidence used by those rows is supplied through the sibling chapter chapter 278. The local packet then contributes H, C, P, N and the generated carry row G, so the NameCert boundary consumes the sibling normal-form surface without replacing it. □

Theorem 544.6 (ZCarry NameCert assembly)**.** *The carrier of Definition 544.1, the obligation surface of Definition 544.2, the non-*hsame *boundary of Definition 544.3, and the normal-target ledger of Definition 544.4 assemble the concrete* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *packet for the displayed Zeckendorf carry classifier.*

Proof. Bind the carrier rows $h, k, G, N_h, N_k, H, C, P, N$. These rows expose the source, target, generator, normality witnesses, non-hsame boundary, transport routes, provenance, and local name.

Apply the finite obligation theorem to identify the public certificate surface. Apply the carry-not-hsame boundary theorem to keep the classifier separate from history-sameness. Apply the normal-target ledger theorem to retain the target-normal and source-non-normal rows.

Repacking these rows gives the local naming certificate. The assembly is exactly the concrete packet recorded by the Lean-side object zCarry_namecert. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorf.Carry.zCarry namecert`

Corollary 544.7 (ZCarry licensed classifier boundary)**.** *The* $\mathsf{ZCarry}^{\uparrow}$ *interface is licensed by the finite packet of Definition 544.6. Any downstream use of the displayed carry must cite the generated carry row, the normality ledgers, the non-*hsame *boundary, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row; it may not read the carry as* hsame *or as a primitive kernel rewrite.*

Proof. Use the assembled $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ packet. Its public rows are exactly the carrier coordinates and obligation rows named above.

The non-hsame boundary is part of the exported ledger. Hence a consumer that uses the carry classifier also receives the evidence preventing a kernel-equality reading, and no primitive rewrite row is available in the packet. □

Lean4 ✓ `BEDC.Derived.ZCarryUp.ZCarryUp single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZCarry}^{\uparrow}$ packet is generated from the displayed Zeckendorf carry row, source non-normal ledger, target normal ledger, non-hsame boundary, componentwise transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$, maturePackageClosed)

Scope closed: The checked zCarry_namecert assembly of Definition 544.6 and the licensed boundary of Definition 544.7 bind the public $\mathsf{ZCarry}^{\uparrow}$ classifier boundary to the bridge-readable package rows G, N_h, N_k, H, C, P, N for the displayed Zeckendorf carry.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.AxisZeckendorf.Carry.zCarry\_namecert`

Bridge status: none

Not claimed: This chapter does not identify the source and target histories by hsame, does not install quotient rewriting, and does not close arbitrary Zeckendorf carry generation.

Upgrade path: No further theory-axis upgrade is claimed here; remaining work is bridge checking for the same displayed generated-carry, normality, non-hsame, Cont, Pkg, and NameCert packet.

Chapter 545

A Concrete Naming Certificate for $\mathsf{TasteGateAdmission}^{\uparrow}$

The $\mathsf{TasteGateAdmission}^{\uparrow}$ packet is the concrete NameCert surface for admitting an AI-origin chapter into the BEDC horizon. It packages the candidate chapter identity, origin row, naming surface, chapter taste gate, event-flow witness, ground-compiler theorem references, transport rows, continuation routes, package provenance, and refusal ledger into one finite audit object.

The admission packet consumes the apophatic refusal discipline of chapter 543, the kernel acceptance trace of chapter 557, and the stability boundary of chapter 564. Those siblings supply refusal, trace, and stability rows for the visible $X, O, N, T, E, R, H, C, P, G$ admission packet.

The admission packet supplies the candidate-admission row consumed by chapter 563: the compiler witness reads the displayed admission row as its chapter-gate input, while the audit refusal surface of chapter 526 bounds the same row by refusing hidden semantic approval outside the finite witness, compiler-reference, replay, and provenance routes.

Definition 545.1 (TasteGateAdmission carrier)**.** *A* $\mathsf{TasteGateAdmission}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, O, N, T, E, R, H, C, P, G)$$

with the following displayed rows:

1. *X is the candidate chapter identity row, recording the chapter label, local carrier name, and the paper site whose admission is being requested;*
2. *O is the origin row, recording that the candidate enters through the AI-origin track rather than through inherited human-curated consensus;*
3. *N is the visible* $\mathsf{NameCert}$ *surface proposed for the candidate chapter, including its carrier, classifier, pattern, ledger, and non-escape rows;*
4. *T is the* $\mathsf{ChapterTasteGate}$ *witness row, recording the conservativity, no-hidden-input, round-trip, and layer-separation obligations exposed by the candidate;*
5. *E is the event-flow witness row, displaying the finite* $\mathsf{GroundCompiler}$ *event-flow image from which the candidate tokens are read;*
6. *R records the ground-compiler theorem references used by the witness, including the no-hidden-input, round-trip, and source-channel separation routes available to the admission check;*
7. *H stores componentwise* hsame *transports for $X, O, N, T, E, R, C, P, G$;*

8. *C records the displayed* Cont *routes by which the admission check reads the NameCert surface, the taste-gate witness, the event-flow witness, and the compiler reference row;*
9. *P is the* Pkg *provenance over* BHist, NameCert, ChapterTasteGate, GroundCompiler, *event-flow witnesses, ground-compiler theorem references,* hsame, Cont, *and local naming;*
10. *G is the GAP refusal ledger, recording failed admission attempts whose NameCert row, taste-gate row, event-flow row, compiler-reference row, transport row, continuation row, or provenance row is absent.*

The carrier has no coordinate for hidden prompt state, an oracle classifier, an unrecorded compiler theorem, host-side chapter approval, or a prose-only admission decision.

Lean4 ✓ `BEDC.Derived.TasteGateAdmissionUp.TasteGateAdmissionTasteGate single carrier alignment`

Theorem 545.2 (TasteGateAdmission NameCert obligations)**.** *For every accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *carrier*

$$A = (X, O, N, T, E, R, H, C, P, G),$$

the local $\mathsf{NameCert}_{\mathsf{TasteGateAdmission}^{\uparrow}}$ *surface consists exactly of carrier admission for A, candidate identity through X, AI-origin exposure through O, candidate NameCert exposure through N,* ChapterTasteGate *exposure through T, event-flow exposure through E, ground-compiler reference exposure through R, componentwise transport through H, replay and admission routes through C, package provenance through P, and refused-case accounting through G.*

Proof. Carrier admission is projection from Definition 545.1. The packet exposes exactly the chapter identity, origin, NameCert surface, taste-gate witness, event-flow witness, compiler references, transport, continuation, provenance, and refusal rows.

The candidate and origin obligations are the rows X and O. They identify the paper site being admitted and record that the admission is evaluated through the AI-origin track.

The naming and taste-gate obligations are the rows N and T. A downstream reader can inspect the candidate's local NameCert surface and its four taste-gate obligations without importing a second chapter object outside the packet.

The compiler-facing obligations are the rows E and R. They record the event-flow witness and the ground-compiler theorem references that make the taste-gate rows auditable.

Finally H, C, P, G preserve the same admission object under transport, replay, provenance, and refused-case accounting. Since every admissible read is routed through those rows, the packet supplies the whole local $\mathsf{NameCert}_{\mathsf{TasteGateAdmission}^{\uparrow}}$ surface and no hidden approval channel. □

Theorem 545.3 (TasteGateAdmission ground compiler witnesses)**.** *In every accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *carrier, the* ChapterTasteGate *witness row T is read only together with the event-flow row E and the ground-compiler reference row R. Thus no-hidden-input, round-trip, and layer-separation claims are admitted only through displayed finite event-flow evidence and named compiler theorem references, not through an unrecorded semantic judgment about the candidate chapter.*

Proof. Start from the carrier of Definition 545.1. The only taste-gate coordinate is T, and its compiler-facing evidence is displayed in E and R.

The no-hidden-input and round-trip reads pass through E before they can be consumed. The row E is the finite event-flow witness, so the admission check sees the displayed compiler image rather than an external account of the candidate's tokens.

The layer-separation and compiler-reference reads pass through R. This row records the named ground-compiler theorem references used by the chapter-level witness.

Transport and replay are then carried by H and C, with package provenance in P. These rows preserve the same witness surface and do not add an unrecorded compiler theorem or host-side approval step. □

Theorem 545.4 (TasteGateAdmission non-escape)**.** *Every accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *admission route factors through the finite rows*

$$(X, O, N, T, E, R, H, C, P, G).$$

The route may export that the candidate chapter has an AI-origin admission packet with a displayed NameCert surface, ChapterTasteGate *witness, ground-compiler witness, audit route, transport route, package provenance, GAP refusal ledger, and local naming data. It cannot export hidden prompt state, oracle approval, unlisted compiler facts, selected host semantics, or a chapter object not named by the packet.*

Proof. Project an accepted admission route through the carrier. The route has access only to the ten displayed rows $X, O, N, T, E, R, H, C, P, G$.

Use Definition 545.2 to account for local naming, candidate identity, origin, taste-gate inspection, compiler evidence, replay, provenance, and refused-case accounting.

Use Definition 545.3 for the compiler-facing part. It keeps every taste-gate read attached to the finite event-flow witness and named compiler-reference rows.

Since the carrier has no coordinate for hidden prompt state, oracle approval, unlisted compiler facts, selected host semantics, or an unnamed chapter object, no exported admission route can supply one. □

Theorem 545.5 (TasteGateAdmission carrier source obligation)**.** *Every accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *admission read has the finite source carrier*

$$(X, O, N, T, E, R, H, C, P, G).$$

These are exactly the candidate identity, AI-origin, candidate NameCert*,* ChapterTasteGate*, event-flow, ground-compiler reference,* hsame*,* Cont*,* Pkg*, and GAP-refusal rows displayed by the admission packet. No admission carrier is supplied by hidden prompt state, oracle approval, host-side chapter acceptance, or an unlisted compiler theorem.*

Proof. Project the read through Definition 545.1. The carrier definition displays precisely the ten finite rows $X, O, N, T, E, R, H, C, P, G$ and contains no additional admission coordinate.

Apply Definition 545.2. The local NameCert surface consumes exactly those rows as carrier, identity, origin, taste-gate, compiler, transport, replay, provenance, and refusal data.

Use Definition 545.3 for the compiler-facing coordinates and Definition 545.4 for the exclusion clause. Together they keep every accepted read inside the displayed finite source and rule out hidden prompt state, oracle approval, host-side acceptance, and unlisted compiler facts. □

Theorem 545.6 (TasteGateAdmission classifier stability obligation)**.** *If two accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *carriers are compared by the admission classifier, the comparison transports only through their displayed* ChapterTasteGate *rows, event-flow witness rows, ground-compiler reference rows,* hsame *rows, and* Cont *rows. It may carry the visible candidate* NameCert*, package provenance, and refusal-ledger coordinates along those routes, but it cannot introduce hidden prompt state, oracle approval, selected host semantics, or an unlisted compiler fact.*

Proof. Start with the carrier rows of Definition 545.1. The classifier-facing coordinates are the taste-gate row T, the event-flow row E, the compiler-reference row R, and the displayed transport and replay rows H and C.

Use Definition 545.2 to keep the candidate NameCert, package provenance, and refusal ledger attached to the same finite carrier. No second admission object is introduced during this transport.

Apply Definition 545.3. It requires the taste-gate comparison to read the displayed event-flow and compiler reference rows before the comparison can be consumed.

Finally apply Definition 545.4. Since the carrier has no hidden prompt, oracle, host-semantics, or unlisted-compiler coordinate, classifier stability cannot transport through one. □

Theorem 545.7 (TasteGateAdmission ledger descent obligation)**.** *Every refused or accepted* $\mathsf{TasteGateAdmission}^{\uparrow}$ *admission read descends only through the displayed GAP/refusal ledger G together with the carrier rows it records. Accepted reads descend through the same finite admission packet, while refused reads record which candidate* NameCert*,* ChapterTasteGate*, event-flow, compiler-reference,* hsame*,* Cont*, or* Pkg *row is absent. Neither kind of read can supply hidden prompt state, oracle approval, or unlisted compiler facts.*

Proof. Inspect the refusal coordinate in Definition 545.1. The row G is the only GAP/refusal ledger, and it is indexed by the same finite carrier rows $X, O, N, T, E, R, H, C, P$.

For accepted reads, apply Definition 545.5. The accepted route descends through the displayed source rows before it can be consumed by the admission check.

For refused reads, use Definition 545.2. The ledger records absence of one of the displayed NameCert, taste-gate, event-flow, compiler-reference, transport, continuation, provenance, or local naming rows.

The ground-compiler witness theorem accounts for compiler-facing refusals, and Definition 545.4 supplies the exclusion clause. Thus the ledger cannot manufacture hidden prompt state, oracle approval, or unlisted compiler facts. □

Closure status for $\mathsf{NameCert}_{\mathsf{TasteGateAdmission}^{\uparrow}}$.

Constructive story: A $\mathsf{TasteGateAdmission}^{\uparrow}$ packet is generated from a candidate chapter identity row, AI-origin row, candidate NameCert surface, ChapterTasteGate witness, event-flow witness, ground-compiler reference row, componentwise transport, continuation routes, package provenance, and GAP refusal ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TasteGateAdmission}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 545.1, Definition 545.2, Definition 545.3, Definition 545.4, Definition 545.5, Definition 545.6, and Definition 545.7 with

Lean4 ✓ `BEDC.Derived.TasteGateAdmissionUp.taste gate`

expose the seed carrier, local NameCert surface, ground-compiler witness rows, non-escape boundary, and explicit TasteGate obligation rows as a public AI-origin admission packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TasteGateAdmission}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TasteGateAdmissionUp.TasteGateAdmissionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove the candidate chapter's mathematical content, certify arbitrary host semantics, or close bridge checking.

Upgrade path: No further paper-side theory closure grade remains; only independent bridge checking can strengthen the recorded status.

Chapter 546

A Concrete Naming Certificate for $\mathsf{KernelAuditWitness}^{\uparrow}$

The $\mathsf{KernelAuditWitness}^{\uparrow}$ surface is the concrete NameCert realization of Lean kernel acceptance as a BEDC audit witness. It packages the term generator, the accepted candidate, the environment ledger, axiom-ancestry report, replay route, provenance, and paper-marker name into one finite packet. The packet refines the socket-boundary discipline of chapter 543 and the accepted-declaration ledger of chapter 549 into a theorem-marker audit surface. The packet does not certify the Lean kernel itself; it records the audit surface by which BEDC reads accepted Lean declarations as checked ledger rows.

Definition 546.1 (KernelAuditWitness carrier)**.** *A* $\mathsf{KernelAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, A, L, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the generator-and-candidate row, recording the source declaration, elaborated candidate, claimed type, and local build context that present a Lean object to the kernel;*
2. *A is the kernel-acceptance stamp row, recording that the candidate in G is accepted under the displayed environment and enters the checked-declaration surface;*
3. *L is the environment ledger row, recording the accepted name, its type, its body or proof object when visible, and the environment position from which later resolution reads the declaration;*
4. *R is the axiom-ancestry and report row, recording the transitive dependency report used by the strict purity audit, including the empty-report case for BEDC-internal theorem content;*
5. *H records componentwise* hsame *transport for the generator, candidate, acceptance stamp, ledger entry, ancestry report, replay route, provenance, and local naming rows;*
6. *C records the* Cont *replay routes for rebuilding the candidate, re-reading the accepted ledger row, querying the ancestry report, and resolving paper-side markers against the displayed name;*
7. *P is the* Pkg *provenance over* BHist*, Lean declaration names, build replay, kernel acceptance, environment ledgers, strict axiom-ancestry reports,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$ *row binding the paper-visible name to the finite audit packet.*

The carrier has no coordinate for an independent semantic endorsement of the surrounding prose, no coordinate for a second checker beneath the Lean kernel, and no coordinate that can turn a nonempty forbidden ancestry report into BEDC-internal theorem content.

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessUp`

Theorem 546.2 (KernelAuditWitness NameCert obligations)**.** *For every accepted* $\mathsf{KernelAuditWitness}^{\uparrow}$ *carrier*

$$K = (G, A, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for* K*, candidate exposure through* G*, kernel-stamp exposure through* A*, environment-ledger exposure through* L*, axiom-ancestry exposure through* R*, componentwise transport through* H*, replay and marker-resolution routes through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessNameCert obligations`

Proof. Carrier admission is projection from Definition 546.1. The packet exposes exactly the generator, stamp, ledger, ancestry, transport, continuation, provenance, and local naming rows.

The candidate and stamp obligations are the rows G and A. A downstream reader can inspect which object was presented to the kernel and which acceptance event is being recorded, but it receives no separate certificate outside those rows.

The ledger and ancestry obligations are the rows L and R. They record where the accepted name is resolved and what transitive dependency report accompanies it, including the strict empty-report case when the declaration is read as BEDC-internal theorem content.

Finally H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Repacking these rows gives the whole local $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$ surface, and there is no remaining coordinate from which a consumer could import an unlisted checker, hidden proof source, or prose-only endorsement. □

Theorem 546.3 (KernelAuditWitness non-contamination)**.** *If the ancestry row* R *of an accepted* $\mathsf{KernelAuditWitness}^{\uparrow}$ *carrier reports a forbidden dependency, then the packet keeps that dependency in the report row and does not classify the declaration as BEDC-internal theorem content. If* R *is the empty strict-purity report, then downstream readers may consume the accepted name through* L *and* C *without importing* Classical.choice*,* Quot.sound*,* propext*, or any other displayed forbidden ancestry from* R*.*

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessAncestry non contamination`

Proof. Read the carrier coordinates. The only ancestry coordinate is R, and Definition 546.1 makes it the row where transitive dependency reports are recorded.

In the forbidden-dependency case, the dependency remains visible as report data. The packet still records the kernel acceptance event, but the NameCert surface cannot erase the report row or reclassify it as empty ancestry.

In the empty-report case, the ledger read passes through L and the replay routes in C. Since R contains no forbidden ancestry to transport, the componentwise hsame row H preserves the same clean report for consumers of the local name N.

Thus contamination cannot move silently across the packet boundary. It is either present as explicit ancestry report data or absent from the strict-purity row used by the downstream BEDC theorem-content read. □

Theorem 546.4 (KernelAuditWitness ledger exactness)**.** *For an accepted* $\mathsf{KernelAuditWitness}^{\uparrow}$ *carrier* $K = (G, A, L, R, H, C, P, N)$*, paper-side marker resolution through* N *and* C *reaches exactly the accepted environment row* L *certified by the stamp row* A*. It cannot resolve to a different declaration, a prose-only claim, a stale candidate not accepted by the kernel, or an ancestry report other than the displayed row* R*.*

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessLedger exactness`

Proof. The local name is bound by N, and the continuation routes are C. By the carrier definition, those routes resolve the paper-visible marker against the accepted declaration row L.

The acceptance stamp A is tied to the candidate row G. A stale candidate that lacks the displayed stamp has no route from G through A into L, so it cannot be the ledger row reached by marker resolution.

The ancestry report is likewise fixed by R. Transport through H is componentwise over the displayed rows, so it may preserve the row under the accepted equality discipline but cannot substitute a different report for the one packaged in K.

Therefore the marker-facing route is exact. It reaches the one accepted environment row and its displayed ancestry report, rather than any neighboring declaration or surrounding prose claim. □

Theorem 546.5 (KernelAuditWitness acceptance gate)**.** *The* $\mathsf{KernelAuditWitness}^{\uparrow}$ *packet gates BEDC Lean-side theorem content by the conjunction of kernel acceptance, ledger resolution, strict ancestry reporting, replay continuity, and local NameCert naming. A declaration whose candidate lacks the stamp row* A*, whose name is absent from* L*, whose ancestry row* R *is unresolved, whose replay route* C *fails, or whose local naming row* N *is absent is not accepted as a complete* $\mathsf{KernelAuditWitness}^{\uparrow}$ *audit witness.*

Proof. Inspect the carrier row by row. Kernel acceptance is represented by A, ledger resolution by L, ancestry reporting by R, replay continuity by C, and local naming by N.

If any one of these rows is absent, the packet cannot supply the corresponding NameCert obligation from Definition 546.2. The missing row blocks the finite route from candidate presentation to paper-visible checked content.

When all rows are present, H and P preserve the packet as a transported and provenance-recorded audit surface. The gate is therefore conjunctive: the accepted declaration, exact ledger row, ancestry report, replay route, and local name must be read together. □

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessAcceptanceGate`

Theorem 546.6 (KernelAuditWitness kernel-scope binding)**.** *For every accepted* $\mathsf{KernelAuditWitness}^{\uparrow}$ *carrier* $K = (G, A, L, R, H, C, P, N)$*, the accepted declaration row* A*, environment ledger row* L*, axiom-ancestry report row* R*, replay route* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$ *row* N *are exactly the kernel-scope witnesses of the packet. A consumer may read the candidate through* G *and transport the rows through* H*, but it cannot replace these kernel-scope witnesses by prose endorsement, an untracked checker, a hidden proof source, or an ancestry report outside* R*.*

Proof. Read the carrier rows in Definition 546.1. The kernel-facing data are precisely the acceptance stamp, accepted environment ledger, ancestry report, replay route, provenance row, and local naming row.

Apply the NameCert obligations and ledger exactness theorems. They bind marker resolution to L through N and C, and they keep the ancestry information in the displayed row R.

The non-contamination and acceptance-gate theorems exclude hidden routes. Since no other carrier coordinate can certify the kernel scope, the listed rows are the exact witnesses for this packet. □

Lean4 ✓ `BEDC.Derived.KernelAuditWitnessUp.KernelAuditWitnessKernelScopeBinding`

Theorem 546.7 (KernelAuditWitness bridge boundary)**.** *Every standard audit-export consumer of a* $\mathsf{KernelAuditWitness}^{\uparrow}$ *packet must read exactly the accepted declaration row A, environment ledger row L, strict ancestry report row R, replay route C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$ *row N. The export route may also inspect the candidate row G and componentwise transport row H, but it cannot replace the six audit-export witnesses by a prose endorsement, hidden checker, hidden proof source, unchecked bridge, or ancestry report outside the displayed packet.*

Proof. Start from the carrier of Definition 546.1. Its audit-facing rows are the accepted declaration stamp, ledger row, ancestry report, replay route, provenance row, and local naming row.

Apply the NameCert obligation surface, ledger exactness, and kernel-scope binding theorems. These results force marker resolution through L, N, C, keep ancestry in R, and keep the kernel-scope witnesses inside the displayed packet.

The non-contamination and acceptance-gate theorems exclude hidden replacements. Thus a standard audit-export consumer can read the displayed rows, but it has no route to substitute prose, an unchecked checker, or an untracked bridge for the packet witnesses. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelAuditWitness}^{\uparrow}$ packet is generated from a Lean candidate row, kernel-acceptance stamp, environment ledger row, axiom-ancestry report, componentwise hsame transport, replay Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 546.1, Definition 546.2, Definition 546.3, Definition 546.4, Definition 546.5, Definition 546.6, and Definition 546.7 bind the mature finite audit-witness carrier, TasteGate-aligned NameCert obligation surface, ancestry non-contamination, ledger exactness, conjunctive acceptance gate, kernel-scope witness boundary, and standard audit-export boundary for downstream marker consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KernelAuditWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not certify the Lean kernel itself, close TasteGate-backed obligation discharge, provide Lean verification, supply public standard bridge data, or prove semantic adequacy of surrounding prose beyond the displayed audit packet.

Upgrade path: Formal advancement requires Lean verification or bridge checking for the same carrier, TasteGate, NameCert, ancestry, ledger, and acceptance rows without enlarging the kernel-scope witness surface.

Chapter 547

A Concrete Naming Certificate for $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$

The $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ surface turns a Lean acceptance event into a finite $\mathsf{NameCert}$ packet. It does not certify the kernel from inside BEDC. It records the generated candidate, the accepted declaration, the environment ledger, the axiom-query boundary, the build replay route, transport, continuation, provenance, and local naming row that make the acceptance event available to audit consumers.

The witness packet consumes the finite audit-gate rows of chapter 526: source-token refusal, axiom-purity dependency report, paper-Lean target audit, origin and import conservativity ledger, and GAP boundary rows. The $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ packet specializes that sibling gate surface to a single Lean acceptance event by adding generated-term, kernel-candidate, accepted-declaration, environment-ledger, query, replay, transport, continuation, provenance, and local $\mathsf{NameCert}$ rows.

Definition 547.1 (KernelAcceptanceAuditWitness carrier)**.** *A* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (G, K, A, L, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the generated-term row naming the elaborated proof term, definition body, inductive declaration, or recursor use submitted to the Lean kernel;*
2. *K is the kernel-candidate row, pairing G with its claimed type, universe data, binder context, recursion account, or constructor signature as applicable;*
3. *A is the accepted-declaration row, present only when the candidate has been accepted under the current Lean environment;*
4. *L is the environment-ledger row recording the accepted name and the module position at which the declaration is entered;*
5. *Q is the axiom-query row naming the transitive dependency query surface used by the strict purity audit;*
6. *R is the build-replay row naming the project replay that reconstructs the same acceptance event from the Lean sources;*

7. *H records componentwise* hsame *transport for the generated term, candidate, accepted declaration, ledger row, axiom query, replay row, provenance, and local naming row;*
8. *C records the displayed* Cont *routes through which consumers read kernel acceptance, inspect the ledger position, ask the axiom query, and replay the build route;*
9. *P is the* Pkg *provenance over* BHist*, generated terms, kernel candidates, accepted declarations, environment ledgers, axiom-query rows, build-replay rows,* hsame*,* Cont*, and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}\uparrow}$ *row.*

The carrier exports no row that proves the Lean kernel correct, no row that turns a weak theorem statement into a stronger paper claim, and no row that hides a dependency outside the axiom-query boundary.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier terminal row semantic certificate`

Theorem 547.2 (KernelAcceptanceAuditWitness NameCert obligations)**.** *For every accepted* KernelAcceptanceAu *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}\uparrow}$ *surface consists exactly of the generated-term row G, kernel-candidate row K, accepted-declaration row A, environment-ledger row L, axiom-query row Q, build-replay row R, componentwise* hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and local naming row N. Any audit consumer of N must read acceptance through the finite packet W and may not replace it by prose confidence, an external certificate, or an unrecorded dependency route.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier namecert obligations`

Proof. Start with the carrier in Definition 547.1. The displayed coordinates expose the candidate-generating side, the kernel acceptance side, the environment ledger, the axiom-query surface, and the replay route in one finite BHist packet.

The transport and continuation rows then bind how the packet is read. The hsame row preserves each named coordinate componentwise, while the Cont row gives only the displayed routes from generated term to candidate, accepted declaration, ledger entry, axiom query, and replay.

Finally the provenance and local naming rows package the same finite data as a $\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitnes}}$ surface. Since no additional coordinate is present, a consumer has no place from which to import prose confidence, an external certificate, or an unrecorded dependency route. □

Theorem 547.3 (KernelAcceptanceAuditWitness ledger non-escape)**.** *For every accepted carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the accepted declaration row A reaches downstream consumers only through the environment-ledger row L, the axiom-query row Q, the build-replay row R, and the local NameCert *row N. Thus a checked BEDC marker may consume kernel acceptance as a ledgered declaration with a replayable source and queryable dependency surface, but it cannot read acceptance as a certificate for the kernel's own correctness or for any paper-side claim not stated by the accepted declaration.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier ledger non escape`

Proof. Project the accepted declaration row A from the carrier. Its only ledger-facing coordinate is L, so downstream reads of the accepted name pass through the environment row rather than through an informal description of the term.

Next read the dependency side through Q and the replay side through R. These rows bind the transitive dependency query and the source replay that reconstructs the acceptance event from the project ledger.

The local naming row N packages A, L, Q, R together with G, K, H, C, P. Because the carrier contains no row for kernel correctness or for unstated paper meaning, ledgered acceptance cannot escape into either claim. □

Theorem 547.4 (KernelAcceptanceAuditWitness axiom-query boundary)**.** *For every accepted carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the axiom-query row Q is the only boundary through which an audit consumer may classify the transitive dependency ancestry of the accepted declaration. The build-replay row R reconstructs the local acceptance event, while Q classifies whether the resulting ledgered declaration imports a forbidden dependency. No consumer may infer dependency cleanliness from the generated-term row G, the candidate row K, the accepted-declaration row A, or the local naming row N alone.

Proof. The generated term, candidate, accepted declaration, and name rows identify what was submitted, accepted, ledgered, and named. These rows do not enumerate the transitive ancestry of the proof object or definition body.

The build-replay row R reconstructs the acceptance event from sources, but replay alone records that the declaration is accepted in the current environment. The dependency classification is a separate audit read.

That separate read is Q. Since the carrier designates Q as the axiom-query row and provides no second ancestry coordinate, any cleanliness conclusion must pass through Q rather than through the local name, the generated term, or the acceptance stamp by itself. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier axiom query boundary`

Definition 547.5 (KernelAcceptanceAuditWitness terminal accepted row)**.** *For an accepted* KernelAcceptanceAudi *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the terminal accepted-row surface is the finite read endpoint

$$A, L, Q, R, N.$$

It records the accepted declaration, its environment ledger entry, the axiom-query boundary, the build-replay route, and the local naming row. It contains no coordinate for kernel correctness, host runtime correctness, or a paper claim stronger than the accepted declaration.

Lean4 (definition) `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessTerminalRow`

Theorem 547.6 (KernelAcceptanceAuditWitness route determinacy)**.** *For every accepted carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the replay route from the generated row G to the accepted row A and the axiom-query route from A to Q are determined by the displayed continuation coordinate C. Any two audit reads that agree

on C and on the packet rows $G, K, A, L, Q, R, H, P, N$ reach the same replay row R and the same axiom-query row Q.

Proof. Read the carrier coordinates first. The generated row, kernel candidate, accepted declaration, ledger row, axiom-query row, replay row, transport row, provenance row, and local name are fixed as the packet W.

The only displayed continuation coordinate is C. Thus a replay consumer and an axiom-query consumer have no second route field from which to choose a different path through the same packet.

If two audit reads agree on C and on the displayed rows, their replay endpoints and axiom-query endpoints are the fixed rows R and Q. Route determinacy is therefore read directly from the finite packet. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier route determinacy`

Theorem 547.7 (KernelAcceptanceAuditWitness replay and query non-escape)**.** *For every accepted carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

every downstream replay or axiom-query consumer factors through the terminal accepted-row surface A, L, Q, R, N together with the displayed route row C. No such consumer can extract kernel correctness, compiler correctness, runtime correctness, dependency cleanliness without Q, replay success without R, or an unstated paper claim from the finite packet.

Proof. Start with the terminal accepted-row surface of Definition 547.5. It exposes only A, L, Q, R, N, so accepted-declaration access terminates at the ledger, query, replay, and naming rows.

Use route determinacy from Definition 547.6. Replay and axiom-query reads are routed by C to the displayed rows R and Q rather than to an implicit source of correctness.

Finally inspect the missing coordinates of the carrier. Since W has no row for kernel correctness, compiler correctness, runtime correctness, hidden dependency cleanliness, or stronger paper meaning, replay and query consumers cannot escape beyond the finite audit packet. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier replay query nonescape`

Theorem 547.8 (KernelAcceptanceAuditWitness public obligation surface)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the public obligation surface consists exactly of the accepted declaration row A, environment-ledger row L, axiom-query row Q, build-replay row R, componentwise hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N, together with the generated and candidate rows G, K needed to replay the acceptance event. Consumers cannot read kernel correctness, compiler correctness, runtime correctness, dependency cleanliness without Q, replay success without R, or stronger paper meaning from this packet.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier public obligation surface`

Proof. First expose the finite packet by Definition 547.2. It identifies the generated row, candidate row, accepted declaration, ledger row, axiom-query row, replay row, transport row, continuation row, provenance row, and local name as the entire NameCert surface.

Next isolate the accepted endpoint. The terminal row of Definition 547.5 gives A, L, Q, R, N, while Definition 547.3 prevents that endpoint from becoming a certificate of kernel correctness or a stronger paper claim.

Finally combine the query and route controls. The axiom-query boundary of Definition 547.4 forces dependency classification through Q, route determinacy from Definition 547.6 fixes the replay and query endpoints through C, and replay/query non-escape from Definition 547.7 excludes compiler correctness, runtime correctness, hidden dependency cleanliness, and stronger paper meaning. □

Theorem 547.9 (KernelAcceptanceAuditWitness source gate)**.** *For every accepted* KernelAcceptanceAuditWitness$^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

an accepted source row may enter the audit witness only through the generated-term row G, the kernel-candidate row K, the accepted-declaration row A, and the environment-ledger row L. The source gate is therefore the displayed route from source replay to kernel candidate to accepted declaration to ledger entry; it is not host equality, proof text, runtime evidence, or an unledgered source artifact.

Proof. Start with the carrier and NameCert obligations. The only entry coordinates for source-facing data are the generated row G, candidate row K, accepted declaration A, and ledger row L.

Use ledger non-escape. It forces downstream reads of A through L, Q, R, N, so a source-facing acceptance row cannot bypass the environment ledger or the replay and query boundaries.

Finally apply route determinacy. The route row C fixes the passage from generation to candidate, acceptance, ledger, replay, and query, while replay/query non-escape excludes host equality, proof text, runtime evidence, and unledgered source artifacts. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier source gate`

Theorem 547.10 (KernelAcceptanceAuditWitness query-ledger totality)**.** *For every accepted* KernelAcceptanceAu *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

every axiom query made by an audit consumer is accounted for by the ledgered query route A, L, Q, R, C, N. A consumer may replay the accepted declaration through R and classify its transitive dependency ancestry through Q, but it cannot obtain dependency cleanliness from G, K, A, L, H, P, N without that query row.

Proof. Read the axiom-query boundary first. It makes Q the only coordinate that classifies the transitive dependency ancestry of the accepted declaration.

Next combine the terminal accepted-row surface with route determinacy. The rows A, L, Q, R, N give the accepted declaration, ledger entry, query boundary, replay route, and local name, while C fixes the route by which the query is reached.

Finally use the public obligation surface. It includes all finite rows of the packet but still withholds dependency cleanliness unless Q is read, so every axiom query is accounted for by the displayed query ledger. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier query ledger totality`

Theorem 547.11 (KernelAcceptanceAuditWitness source-query route exhaustion)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

every public source-facing audit route from the generated row G to dependency classification passes through the kernel-candidate row K, accepted-declaration row A, environment-ledger row L, build-replay row R, axiom-query row Q, displayed continuation row C, and local $\mathsf{NameCert}$ *row N in that order of displayed obligations. No audit consumer may infer kernel correctness, replay success, or dependency cleanliness from G, K, A, L, H, P, N without the replay row R and the query row Q.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier source query route exhaustion`

Proof. Start with the carrier definition. It lists exactly the rows $G, K, A, L, Q, R, H, C, P, N$, so a public source-facing route has no coordinate outside the generated row, kernel candidate, accepted declaration, ledger row, query boundary, replay row, transport, continuation, provenance, and local name.

Apply the source gate to the entry side. Source-facing material can enter the witness only through G, K, A, L, which fixes the passage from generated row to kernel candidate, accepted declaration, and environment ledger before any dependency classification is read.

Use query-ledger totality for the audit side. Every query is accounted for by the ledgered route through A, L, Q, R, C, N, while the displayed obligations require the build replay row R and axiom-query row Q before the route can be consumed by C and named by N.

Finally apply replay/query non-escape. The finite packet exports no row for kernel correctness, replay success without R, or dependency cleanliness without Q, so the source-to-query route is exhausted by the displayed rows and cannot be replaced by G, K, A, L, H, P, N alone. □

Theorem 547.12 (KernelAcceptanceAuditWitness standard bridge preconditions)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

a standard bridge request for this audit witness must display the generated-term row G, kernel-candidate row K, accepted-declaration row A, environment-ledger row L, axiom-query row Q, build-replay row R, componentwise transport row H, displayed continuation row C, provenance row P, and local naming row N. These are the bridge preconditions for a public replay/query handoff over BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}}$.

The bridge request may not replace any of these rows by an accepted-name string, prose confidence, host runtime behavior, kernel-correctness claim, hidden dependency classification, or unledgered replay artifact.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier standard bridge preconditions`

Proof. Start with the carrier in Definition 547.1. It lists exactly the generated term, candidate, accepted declaration, environment ledger, axiom query, build replay, transport, continuation, provenance, and local name as the finite packet rows.

Apply the NameCert obligation surface of Definition 547.2. The local naming row packages precisely those coordinates, so a bridge request that consumes the NameCert must display the same rows rather than a smaller accepted-name surface.

Use the axiom-query boundary and route determinacy, Definition 547.4 and Definition 547.6. They force dependency classification through Q and route replay/query access through C to the displayed R and Q endpoints.

Finally combine query-ledger totality and source-query route exhaustion, Definition 547.10 and Definition 547.11. These results account for every bridge-facing source-to-query read by the ledgered packet $G, K, A, L, Q, R, H, C, P, N$, leaving no hidden row from which a standard bridge request could infer acceptance, replay, dependency cleanliness, or local naming. □

Theorem 547.13 (KernelAcceptanceAuditWitness bridge query replay boundary)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the bridge-facing read of A cannot infer dependency cleanliness without the axiom-query row Q and cannot infer replay success without the build-replay row R. The continuation row C, ledger row L, provenance row P, and local name N may route, ledger, package, and name the read, but they do not substitute for Q or R.

Thus the public bridge boundary separates dependency classification from build replay while keeping both rows inside the same finite $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier bridge query replay boundary`

Proof. First apply the axiom-query boundary, Definition 547.4. It makes Q the only coordinate through which a bridge-facing consumer may classify the accepted declaration's transitive dependency ancestry.

Next apply route determinacy, Definition 547.6. The displayed continuation row C fixes the replay endpoint R and the query endpoint Q, but it does not turn either endpoint into the other.

Use query-ledger totality and source-query route exhaustion, Definition 547.10 and Definition 547.11. Together they account for every source-facing bridge read through G, K, A, L, Q, R, C, N and require the query and replay rows before dependency and replay conclusions can be consumed.

Finally inspect the carrier rows through Definition 547.1. The remaining rows H, P, N transport, package, and name the finite packet, but none of them is an axiom query or build replay. The bridge boundary therefore permits no dependency cleanliness without Q and no replay success without R. □

Theorem 547.14 (KernelAcceptanceAuditWitness query replay interface)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

a public audit read of the accepted declaration A reaches replay evidence only through the build-replay row R and dependency classification only through the axiom-query row Q. The pair is routed by the displayed continuation row C, ledgered by the environment row L, packaged by Pkg *provenance P, and named by the local* NameCert *row N. Neither row can substitute for the other: replay through R is not dependency classification, and query classification through Q is not build replay.*

The interface exports no kernel correctness, compiler correctness, runtime correctness, hidden dependency cleanliness, or paper meaning stronger than the accepted declaration and its displayed audit rows.

Proof. Start with Definition 547.4. It makes Q the only coordinate through which a consumer classifies the transitive dependency ancestry of the accepted declaration, while replay through R only reconstructs the local acceptance event.

Apply Definition 547.6. The displayed continuation row C fixes both the replay endpoint and the query endpoint for reads of A, so no second route can exchange their roles.

Use Definition 547.10. The query side is accounted for by the ledgered route A, L, Q, R, C, N, with dependency cleanliness unavailable unless the query row Q is read.

Finally invoke Definition 547.16. The replay side is covered by R, L, C, Q, N as a replayed and ledgered acceptance event. Combining the two rows gives the public interface: R supplies replay, Q supplies dependency classification, and L, C, P, N ledger, route, package, and name the pair without adding kernel correctness, compiler correctness, runtime correctness, hidden dependency cleanliness, or stronger paper meaning. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier query replay interface`

Theorem 547.15 (KernelAcceptanceAuditWitness acceptance boundary)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

kernel acceptance records that the declaration row A has entered the environment ledger L with replay route R and axiom-query boundary Q. It does not promote host equality, proof text, generated prose, runtime behavior, compiler correctness, or a stronger BEDC theorem into evidence beyond the accepted declaration and its displayed audit rows.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier acceptance boundary`

Proof. Begin with the public obligation surface. It exposes the accepted declaration, environment ledger, axiom-query row, build replay, transports, continuations, provenance, local name, and the generated and candidate rows needed for replay.

Use the source gate theorem above to restrict how source-facing material reaches acceptance. The only admitted route passes through G, K, A, L and then through the displayed replay and query coordinates.

Apply replay/query non-escape and the axiom-query boundary. Replay success is read through R, dependency classification through Q, and the carrier contains no coordinate for host equality, proof text, generated prose, runtime behavior, compiler correctness, or stronger BEDC evidence. □

Theorem 547.16 (KernelAcceptanceAuditWitness build replay coverage)**.** *For every accepted* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

each accepted declaration row consumed by the witness is covered by the displayed build-replay row R, environment ledger L, route row C, query row Q, and local name row N. Thus an audit

consumer may read acceptance only as the replayed and ledgered declaration A with its query boundary and local NameCert *packaging, not as an unledgered kernel event or an external proof of correctness.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier build replay coverage`

Proof. Start with the carrier and terminal accepted-row surface. The carrier displays $G, K, A, L, Q, R, H, C, P, N$, while the terminal surface isolates A, L, Q, R, N as the endpoint available to acceptance consumers.

Apply route determinacy to the replay and query routes. The displayed continuation row C fixes the replay endpoint R and query endpoint Q for the accepted row A, so another route cannot be substituted after the declaration has been accepted.

Finally use replay/query non-escape. Since replay success is read through R, dependency classification through Q, ledger position through L, and local naming through N, the accepted declaration is covered exactly by the displayed rows and cannot be read as an unledgered kernel event or an external correctness certificate. □

Theorem 547.17 (KernelAcceptanceAuditWitness public bridge boundary)**.** *For every accepted* KernelAcceptanceAuditWitness$^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

any public bridge consumer of W may read only the accepted declaration row A, environment ledger L, axiom-query row Q, build-replay row R, displayed route row C, package provenance P, and local NameCert *row N, together with the transport row H as the componentwise equality data already displayed by the packet. Such a consumer cannot infer kernel correctness, compiler correctness, host runtime correctness, dependency cleanliness without Q, or any paper claim stronger than the accepted declaration A.*

Proof. Start with the axiom-query boundary. It makes Q the only displayed row that classifies transitive dependency ancestry, so a public bridge consumer cannot turn A, L, H, C, P, N into dependency cleanliness without reading Q.

Next use build replay coverage. The accepted declaration is covered by R, L, C, Q, N, so the bridge-facing read of acceptance remains the replayed and ledgered row A rather than an unledgered kernel event or an external correctness certificate.

Apply dependency-query exhaustion for the cleanliness side. It rules out reads of dependency cleanliness through the generated row, kernel candidate, accepted declaration, ledger, transport, provenance, or local name when the query boundary is absent.

Finally combine replay/query non-escape with the marker-consumer boundary. Replay and query consumers factor through A, L, Q, R, C, N, while marker consumers see only the accepted declaration, replay, query, ledger, local name, and route rows. The public bridge boundary therefore exposes the finite audit packet but does not export kernel correctness, compiler correctness, host runtime correctness, hidden dependency cleanliness, or a strengthened paper claim beyond A. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier public bridge boundary`

Theorem 547.18 (KernelAcceptanceAuditWitness dependency-query exhaustion)**.** *For every accepted* KernelAcceptanceAuditWitness$^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

dependency cleanliness can be read only through the axiom-query row Q. It cannot be inferred from the generated row G, kernel-candidate row K, accepted-declaration row A, environment ledger L, transport row H, provenance row P, or local name row N without the displayed query boundary.

Proof. Use the axiom-query boundary first. It makes Q the only row that classifies the transitive dependency ancestry of the accepted declaration.

Next apply query-ledger totality. Every audit query is accounted for by the ledgered route A, L, Q, R, C, N, so a cleanliness read must pass through the displayed query row rather than through a neighboring carrier coordinate.

The public obligation surface then lists all rows available to the consumer and still withholds dependency cleanliness unless Q is read. Therefore G, K, A, L, H, P, N can identify, ledger, transport, package, and name the declaration, but none of them can replace the axiom query. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier dependency query exhaustion`

Theorem 547.19 (KernelAcceptanceAuditWitness marker consumer boundary)**.** *For every accepted* KernelAcceptanceAuditWitness$^{\uparrow}$ *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

a paper checked-marker consumer may consume only the accepted declaration row A, build-replay row R, axiom-query row Q, environment-ledger row L, and local NameCert *row N, together with the displayed route C that connects them. Such a marker consumer may not read kernel correctness, a stronger paper theorem, compiler correctness, runtime behavior, or dependency cleanliness outside Q.*

Proof. Begin with ledger non-escape and the acceptance boundary. They make acceptance available as the ledgered declaration A with L, Q, R, N, while excluding kernel correctness, compiler correctness, runtime behavior, and stronger paper meaning.

Use dependency-query exhaustion for the dependency side. Cleanliness is available only through Q, so a marker cannot infer it from the generated term, candidate, accepted row, ledger row, transport, provenance, or name by itself.

Finally combine build replay coverage with route determinacy. The replay row R, query row Q, ledger row L, local name N, and route row C form the consumer boundary for the paper marker. Since no stronger coordinate is present in the carrier, the marker consumes the accepted declaration rather than a strengthened theorem. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier marker consumer boundary`

Theorem 547.20 (KernelAcceptanceAuditWitness public export)**.** *For every accepted* KernelAcceptanceAuditWitn *carrier*

$$W = (G, K, A, L, Q, R, H, C, P, N),$$

the public audit interface exported by W is exactly the finite packet consisting of the carrier, the source gate, query-ledger totality, build replay coverage, dependency-query exhaustion, and the marker-consumer boundary. A downstream audit consumer may therefore read the acceptance event only through A, L, Q, R, C, N together with the displayed provenance and transport rows P, H; it may not replace that interface by kernel correctness, compiler correctness, generated prose, host runtime behavior, or a stronger theorem than the accepted declaration.

Proof. First read the carrier and source gate. The carrier displays $G, K, A, L, Q, R, H, C, P, N$, and the source gate restricts source-facing material to the route through the generated term, kernel candidate, accepted row, ledger, replay, query, transport, continuation, provenance, and local name rows.

Next combine query-ledger totality with build replay coverage. These results force every accepted-row audit query to pass through A, L, Q, R, C, N and keep replay success on the displayed replay row R rather than on an unledgered kernel event.

Finally apply dependency-query exhaustion and the marker-consumer boundary. Cleanliness is readable only through Q, while marker consumers see only the accepted declaration, replay row, query row, ledger row, local name row, and their displayed route. The exported interface is therefore public as an audit object, but it contains no coordinate for kernel correctness, compiler correctness, generated prose, host runtime behavior, or theorem strengthening beyond A. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier public export`

Theorem 547.21 (KernelAcceptanceAuditWitness accepted-row consumer boundary)**.** *Every downstream consumer of an accepted* KernelAcceptanceAuditWitness$^{\uparrow}$ *row factors first through the terminal accepted row A and the query ledger Q before it may read package provenance P or the local* NameCert *row N. Thus provenance identifies how the already ledgered accepted row is packaged, but it cannot supply dependency cleanliness, replay success, or acceptance by itself.*

Proof. Start from the terminal accepted-row surface and the acceptance boundary. They expose A together with L, Q, R, N as the consumer endpoint and prevent a downstream read from treating provenance or naming as acceptance.

Use dependency-query exhaustion for the query side. Any cleanliness read must pass through Q, so provenance cannot replace the axiom-query row even when it packages the same accepted declaration.

Now apply build replay coverage and the marker-consumer boundary. Replay success is read through R, marker consumption is bounded by A, L, Q, R, C, N, and P only packages the ledgered accepted row. Therefore an accepted-row consumer reaches provenance and local naming only after the terminal row and query ledger have already been fixed. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier accepted row consumer boundary`

Theorem 547.22 (KernelAcceptanceAuditWitness formal-axis separation)**.** *The public paper export in Definition 547.20 is a paper-axis audit interface for the finite* KernelAcceptanceAuditWitness$^{\uparrow}$ *packet. It does not assert a Lean verification result, does not introduce a Lean marker, and leaves the chapter's formal status at* unformalized *until an independent Lean target verifies the corresponding carrier, ledger, replay, query, transport, provenance, and naming obligations.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceAuditWitnessUp.KernelAcceptanceAuditWitnessCarrier formal axis separation`

Proof. Inspect the exported rows. The public interface is built from the paper-side carrier and the finite audit route A, L, Q, R, C, N with H and P as displayed support rows.

The theorem adds no Lean-side verification marker and cites no Lean declaration as its verifier. Hence it records the paper closure of the audit interface only.

The closure block therefore keeps the formal status at unformalized while raising the theory axis to the public audit surface. Formal verification remains a separate obligation for the Lean-side pipeline. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ packet is generated from an elaborated term, kernel candidate, accepted declaration, environment ledger, axiom-query row, build-replay row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 547.1, Definition 547.2, Definition 547.3, Definition 547.4, Definition 547.5, Definition 547.6, Definition 547.7, Definition 547.8, Definition 547.9, Definition 547.10, Definition 547.11, Definition 547.15, Definition 547.16, Definition 547.17, Definition 547.18, Definition 547.19, Definition 547.20, Definition 547.21, and Definition 547.22 bind the finite audit-witness carrier, local naming surface, terminal accepted-row endpoint, ledger route, replay route, deterministic query route, dependency-query boundary, public obligation surface, source gate, query-ledger totality, source-query route exhaustion, acceptance boundary, build replay coverage, public bridge boundary, dependency-query exhaustion, marker consumer boundary, public export, accepted-row consumer boundary, and verification-axis separation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, kernel correctness, host runtime correctness, compiler correctness, theorem-content strengthening beyond the accepted declaration, or a public standard bridge.

Upgrade path: The accepted-row audit interface is mature on the paper closure axis; formal verification and bridge checking remain independent Lean-side obligations.

$\mathsf{CriticalLineWitness}^{\uparrow}$ reads its zero-strip and source-modulus rows through the $\mathsf{Zeta}^{\uparrow}$ source of chapter 71 and the Riemann-hypothesis boundary route already routed by the child files, without claiming RH, a zero-free strip theorem, analytic continuation, or a standard bridge.

Chapter 548

A Concrete Naming Certificate for $\mathsf{CriticalLineWitness}^{\uparrow}$

The $\mathsf{CriticalLineWitness}^{\uparrow}$ interface records the finite packet a constructive Riemann Hypothesis witness would expose after a zero has already been supplied in the critical strip. It does not prove RH and does not produce a zero. It only names the BEDC-native surface of the witness type in Definition 9239.13 and the restriction type in Definition 9239.17.

Definition 548.1 (Critical-line witness carrier)**.** *A* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Z, S, M, R, Q, H, C, P, N)$$

with the following visible coordinates:

1. *Z is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero row, displaying the zero history and its zero-classifier witness;*
2. *S is the supplied* $\mathsf{CritStrip}^{\uparrow}$ *critical-strip row for the same complex history, displaying the left and right strip comparisons;*
3. *M is the on-critical-line modulus row, sending each depth $n : \mathsf{Nat}^{\uparrow}$ to an inspected rational depth at which the real-part representative is within 2^{-n} of $1/2$;*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *real-part row consumed by the modulus;*
5. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger containing the half-line rational row, dyadic tolerance row, and strict comparison row used by M;*
6. *H records componentwise* hsame *transports for the zero row, critical-strip row, real-part row, rational comparison ledger, and modulus row;*
7. *C records the displayed* Cont *routes through which consumers inspect those rows;*
8. *P is the* Pkg *provenance over* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and the supplied analytic certificates;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row.*

The carrier exposes a witness packet for a supplied zero-in-strip input. It contains no proof that every zero in the critical strip has such a packet, no zero-producing search, no host complex equality, no quotient real equality, no selected representative of a Cauchy class, and no appeal to ambient analytic completeness.

Lean4 (definition) `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessUp`

Theorem 548.2 (Critical-line witness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier admission for the displayed packet of Definition 548.1;*
2. *zero-strip compatibility, stating that the* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z *and* $\mathsf{CritStrip}^{\uparrow}$ *row* S *name the same supplied complex history through displayed* hsame *transport;*
3. *modulus discipline, stating that every depth request is answered by the on-critical-line modulus row* M *using only the real-part row* R *and rational comparison ledger* Q*;*
4. *non-escape discipline, stating that consumers reach the critical-line conclusion only through* $Z, S, M, R, Q, H, C, P, N$*;*
5. *ledger exactness, stating that the public witness surface is exhausted by the displayed zero, strip, modulus, comparison, transport, continuation, package, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert obligations`

Proof. Carrier admission is projection of the finite packet in Definition 548.1. The packet lists the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, transports, continuations, package provenance, and local naming row.

Zero-strip compatibility uses the componentwise hsame rows in H to bind Z and S to the same supplied complex history. No host complex equality is read, and no independent zero search is introduced.

Modulus discipline is the row M. A depth request n is routed to the inspected real-part representative in R and to the dyadic comparison rows in Q, yielding the critical-line approximation at that depth.

The non-escape and ledger rows then use C, P, and N. Since the packet has no quotient, representative-selection, host-equality, or analytic-completeness coordinate, every public witness read is exhausted by the displayed finite rows. □

Theorem 548.3 (Critical-line witness root unblock source triad)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the supplied zero row* Z*, the supplied critical-strip row* S*, and the shared transport route through* H, C, P, N *are exposed before any witness read of the modulus row* M*. The source triad is a finite* $\mathsf{ZetaZeros}^{\uparrow}$–$\mathsf{CritStrip}^{\uparrow}$ *route over the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a host zero source or analytic continuation surface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootUnblockSourceTriad`

Proof. Project the carrier of Definition 548.1. Its first displayed source coordinates are the supplied $\mathsf{ZetaZeros}^{\uparrow}$ zero row Z and supplied $\mathsf{CritStrip}^{\uparrow}$ strip row S, with the transport and replay rows H and C carried in the same finite packet.

Use the NameCert obligations of Definition 548.2. The zero-strip compatibility row binds Z and S to the same supplied complex history through displayed hsame transport, while the non-escape row forces public reads through $Z, S, M, R, Q, H, C, P, N$.

The source chapters chapter 73 and chapter 72 supply the zero and strip surfaces that the carrier names. Thus the source triad is visible before any modulus witness read and has no route to host zero data or analytic continuation. □

Theorem 548.4 (Critical-line witness root unblock modulus ledger)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each modulus request factors through the displayed rows*

M, R, Q, H, C, P, N. *It consumes only* $\mathsf{Nat}^{\uparrow}$ *depth input, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger,* BHist *carrier data,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *obligation surface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root unblock modulus ledger`

Proof. Read the modulus coordinate from Definition 548.1. The row M is the only coordinate that answers a depth request, and the carrier displays R, Q, H, C, P, N as the rows consumed by that answer.

Apply the modulus-discipline and ledger-exactness rows of Definition 548.2. They force each request to use the real-part row R and rational comparison ledger Q, with transport, replay, provenance, and local naming supplied by H, C, P, N.

The depth and comparison sources are the public surfaces of chapter 28, chapter 38, and chapter 37. Since no other coordinate is present, the modulus ledger is exactly the finite route through M, R, Q, H, C, P, N and the listed BEDC kernel rows. □

Theorem 548.5 (Critical-line witness root unblock refusal boundary)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the exported refusal boundary contains no* RH *proof, no zero search, no analytic-continuation certificate, no quotient representative, no host equality, and no ambient completeness witness. Downstream consumers may cite only the source triad, modulus ledger, and local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by the packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root unblock refusal boundary`

Proof. Start with Definition 548.1. The carrier lists exactly $Z, S, M, R, Q, H, C, P, N$ and explicitly withholds global zero production, host complex equality, quotient real equality, selected representatives, and ambient analytic completeness.

Use Definition 548.2. Its non-escape and ledger-exactness rows say that every public witness read is exhausted by the displayed zero, strip, modulus, comparison, transport, continuation, package, and local naming rows.

The sibling source surfaces cited by the carrier, namely chapter 73, chapter 72, chapter 38, chapter 37, and chapter 28, provide only the named rows consumed above. Therefore the exported boundary refuses RH proof data, zero search, analytic-continuation certificates, quotient representatives, host equality, and ambient completeness. □

Theorem 548.6 (Critical-line witness carrier-admission row)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the carrier-admission obligation is exactly the displayed BHist *row of Definition 548.1. It admits the supplied zero row* Z*, critical-strip row* S*, on-line modulus row* M*, real-part row* R*, rational comparison ledger* Q*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*, and it admits no zero search, global zero classifier, host complex equality, quotient representative, or analytic completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier obligation carrier admission row`

Proof. Project the carrier definition. Its displayed tuple is precisely $Z, S, M, R, Q, H, C, P, N$, with each entry anchored as a finite BHist row.

Use the carrier-admission item in Definition 548.2. That item makes the displayed packet the only entry point for witness reads.

The non-escape boundary of Definition 548.13 removes the excluded zero-search, classifier, host-equality, quotient, and completeness rows. Therefore carrier admission is exactly the displayed packet. □

Theorem 548.7 (Critical-line witness modulus-discipline row)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the modulus-discipline obligation says that each depth request is answered only by the on-critical-line modulus row M using the real-part row R, the rational comparison ledger Q, the displayed transport row H, and the continuation row C. It does not appeal to an RH *proof, zero search, global zero-set classifier, host equality, quotient representative, or analytic completeness claim.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier obligation modulus discipline row`

Proof. Begin with the modulus-discipline item in Definition 548.2. It identifies M as the row that answers depth requests and names R and Q as the rows consumed by that answer.

Apply Definition 548.17. The depth route stays inside M, R, Q, H, C, P, N and uses the rational comparison rows already displayed in Q.

The non-escape theorem Definition 548.13 excludes RH, zero search, global classification, host equality, quotient representatives, and analytic completeness from the route. Thus the modulus discipline is a local BHist obligation over the displayed packet. □

Theorem 548.8 (Critical-line witness ledger-exactness row)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the ledger-exactness obligation identifies the public witness ledger with exactly the displayed rows $Z, S, M, R, Q, H, C, P, N$. Every accepted consumer read is sourced by these rows and cannot add an RH *proof, zero-producing search, global classifier, host equality, quotient representative, or ambient analytic-completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier obligation ledger exactness row`

Proof. Read ledger exactness from Definition 548.14. It identifies the public ledger with the displayed supplied zero, strip, modulus, real-part, comparison, transport, continuation, provenance, and naming rows.

Use Definition 548.2 to attach that ledger to the local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ surface. The ledger is therefore not an external analytic assertion but one row of the same finite packet.

Finally apply Definition 548.13. The excluded RH, search, classifier, host-equality, quotient, and completeness coordinates are absent, so ledger exactness is exactly the displayed BHist surface. □

Theorem 548.9 (Critical-line witness zero-strip compatibility)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the supplied ZetaZeros$^{\uparrow}$ *row* Z *and supplied* CritStrip$^{\uparrow}$ *row* S *are tied to the same complex history by displayed* hsame *transport in* H*. A public witness read may therefore consume* Z *as the zero row and* S *as the strip row for the same supplied history, but it cannot replace that displayed transport by host complex equality or by a new zero-producing search.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip zero compatibility`

Proof. Project the carrier and inspect the transport coordinate H. Its zero-strip component is the only row that connects the zero row Z and the critical-strip row S.

The row is an hsame transport between the displayed complex histories already carried by Z and S. It moves the supplied history across the two public rows without introducing a separate complex equality principle.

Since the carrier contains no search coordinate and no host equality coordinate, the compatibility read is exhausted by the displayed zero row, strip row, and transport row. □

Theorem 548.10 (Critical-line witness real-part modulus stability)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *carrier*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

transported depth requests preserve the alignment between the Real$^{\uparrow}$ *real-part row* R *and the online modulus row* M*. At each depth* $n : \mathsf{Nat}^{\uparrow}$*, the answer* $M(n)$ *is read against the transported* R *row and the comparison ledger* Q*, so the public critical-line modulus remains attached to the same supplied zero-in-strip history.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier realpart modulus stability`

Proof. Start with a depth request $n : \mathsf{Nat}^{\uparrow}$. The carrier exposes only the modulus coordinate M as the answer row for such a request.

The real-part row used by that answer is R. Transport through H keeps the displayed real-part history aligned with the zero and strip rows, and continuation through C only routes this already displayed data.

The rational comparison needed to read the modulus is then supplied by Q. Thus every transported request still reads the same R row through M and Q, with no selected Cauchy representative or ambient analytic completeness principle. □

Theorem 548.11 (Critical-line witness rational comparison ledger exactness)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *carrier*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the Rat$^{\uparrow}$ *comparison ledger* Q *is exactly the finite collection of the half-line row, the dyadic tolerance row, and the strict comparison row used by the modulus coordinate* M*. Every public rational comparison read of the witness packet factors through those three rows and cannot appeal to an external real-order decision, quotient equality, or hidden numeric oracle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rat comparison ledger exactness`

Proof. Inspect the comparison ledger coordinate Q. By carrier admission, Q exposes the half-line row, the dyadic tolerance row, and the strict comparison row used by the modulus.

The modulus coordinate M consumes exactly those rational rows when answering a depth request. The half-line row supplies the target $1/2$, the dyadic tolerance row supplies the inspected 2^{-n} bound, and the strict comparison row supplies the visible inequality read.

No other rational comparison coordinate is present in W. Therefore the public comparison ledger is exact and has no route to an external real-order decision, quotient equality, or hidden oracle. □

Theorem 548.12 (Critical-line witness modulus row)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

each request $n : \mathsf{Nat}^{\uparrow}$ *is answered by the displayed modulus coordinate* $M(n)$ *and the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*. The resulting read states that the rational representative of the* $\mathsf{Real}^{\uparrow}$ *real-part row* R *at the inspected depth lies within the dyadic tolerance* 2^{-n} *of the half-line row* $1/2$*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus row`

Proof. Project the carrier and inspect the modulus coordinate. The only field that answers a depth request is M, and its target is the inspected real-part representative carried by R.

The rational comparison is then read from Q. The half-line row, dyadic tolerance row, and strict comparison row are all visible coordinates of the same packet, so the read remains at the $\mathsf{Rat}^{\uparrow}$ level.

Transport through H and continuation through C preserve exactly these rows. The package and local naming fields P, N record provenance and membership, but they add no analytic proof of RH and no off-ledger equality principle. □

Theorem 548.13 (Critical-line witness non-escape boundary)**.** *Every public critical-line read exported by an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier factors through* $Z, S, M, R, Q, H, C, P, N$*. It cannot read a host complex equality, quotient real equality, selected Cauchy representative, zero-producing search, global zero-set classifier, or ambient analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessNonescapeBoundary`

Proof. Start with the carrier inventory. The packet contains exactly the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, transport rows, continuation rows, package provenance, and local naming row.

The critical-line conclusion is consumed through the modulus row and rational comparison ledger. The zero and strip rows provide the supplied input context, while H and C provide only displayed transport and route data.

None of the excluded objects appears as a coordinate of the carrier. Therefore a public read has no route by which host equality, quotient equality, representative selection, zero search, global zero-set classification, or ambient completeness can enter the witness packet. □

Theorem 548.14 (Critical-line witness ledger exactness)**.** *The public* $\mathsf{CriticalLineWitness}^{\uparrow}$ *ledger of an accepted carrier is exactly the finite packet*

$$Z, S, M, R, Q, H, C, P, N.$$

It exposes a BEDC-native witness that a supplied $\mathsf{ZetaZeros}^{\uparrow}$ *row in a supplied* $\mathsf{CritStrip}^{\uparrow}$ *row carries an on-critical-line modulus over* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$*, and it exposes no proof that all non-trivial zeros admit such a packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier ledger exactness`

Proof. Read the public ledger by projecting the carrier. The rows Z and S provide the supplied zero-in-strip input, and M, R, Q provide the critical-line modulus and its rational comparison surface.

The rows H and C account for the only permitted movement of the displayed data, while P and N record package provenance and local naming. These rows do not add a quantifier over all zeros and do not create a zero.

Consequently the ledger is exact: every exported witness coordinate is one of $Z, S, M, R, Q, H, C, P, N$, and every coordinate in that tuple has a public role in the witness packet. □

Theorem 548.15 (Critical-line witness zero-row source scope)**.** *An accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet consumes only the supplied zero row, critical-strip row, modulus route, real-part row, rational comparison ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *evidence:*

$$Z, S, M, R, Q, H, C, P, N.$$

This finite zero-row source scope exports no proof of RH*, zero construction, global zero-set classifier, analytic continuation, host equality, quotient representative, or analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero row source scope`

Proof. Begin with the carrier and NameCert obligation surface. The accepted packet exposes exactly the supplied zero row Z, strip row S, modulus row M, real-part row R, comparison ledger Q, transports H, continuations C, provenance P, and local naming row N.

Use Definition 548.9 and Definition 548.10 to keep the witness attached to the displayed zero-in-strip source and its inspected real-part route. The modulus row is therefore a finite witness row, not a construction of zeros or a classification of all zero rows.

Apply Definition 548.11 and Definition 548.13. They account for the half-line comparison and exclude host equality, quotient representatives, analytic continuation, and ambient completeness from the packet surface.

Finally invoke Definition 548.14. It identifies the public ledger with $Z, S, M, R, Q, H, C, P, N$, so every accepted read is sourced by the finite zero-row packet and no global RH-facing conclusion is exported. □

Theorem 548.16 (Critical-line witness critical-strip origin)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *consumes its strip row through the sibling critical-strip and zeta-zero witness surfaces of chapter 72 and chapter 73. The row* S *is therefore a supplied critical-strip certificate for the same zero row* Z*, transported only by the displayed* H *and read only through the displayed* C *route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier hsame cont ledger obligation`

Proof. Project the packet rows. The supplied zero coordinate is Z, and the supplied strip coordinate is S.

Read S through the critical-strip surface and Z through the zeta-zero surface. The carrier contains no independent strip constructor and no zero search coordinate.

The transport row H ties the two displayed histories together, while C records the public route. Thus the strip origin is exactly the sibling critical-strip and zeta-zero NameCert boundary. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier critical strip origin`

Theorem 548.17 (Critical-line witness modulus-depth routing)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each depth request factors through the Nat depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, the dyadic tolerance row inside* Q*, and the displayed* Cont *route* C *before the line-modulus read* M *is exposed.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus depth route determinacy`

Proof. Begin with the depth request. The carrier names M as the only modulus row, so the request cannot be answered outside that coordinate.

The read of M uses the real-part row R and the comparison ledger Q. The dyadic tolerance row is one of the finite rational comparison rows already recorded inside Q.

Route replay is then supplied by C, with componentwise transport in H and provenance in P, N. No quotient representative, host real equality, or ambient analytic completeness coordinate is available to answer the depth request. □

Theorem 548.18 (Critical-line witness RH nonclassification boundary)**.** *An accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *is a witness packet for the supplied zero-in-strip input* Z, S*. It cannot classify all zeros, construct zeros, export* RH*, or turn the modulus row* M *into a global zero-set theorem.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh nonclassification boundary`

Proof. Project the input coordinates Z and S. They are supplied rows, not outputs of the packet.

The modulus route M, R, Q, H, C witnesses the critical-line read for that supplied input only. It contains no quantifier over all zero rows and no constructor for new zeta zeros.

The package and local name rows P, N preserve this finite packet boundary. Since no global classifier or RH coordinate is present, the public read remains a local supplied-input witness. □

Theorem 548.19 (Critical-line witness depth-request ledger)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each Nat depth request for the critical-line modulus reads only the modulus row* M*, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, and the displayed* Cont *route* C*, transported through* H *and recorded by* P, N*. The request has no coordinate for a quotient representative, host real equality, hidden numeric oracle, global zero classifier, or ambient analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier depth request ledger`

Proof. Project the carrier coordinates. The depth request can reach the modulus only through the displayed row M, and the carrier exposes no second modulus or external oracle row.

Read the supporting rows for that request. The real-part row R supplies the line-facing real coordinate, while Q supplies the rational half-line and dyadic comparison ledger used by the request. The continuation row C records the route, and H, P, N transport and name the same finite packet.

Apply Definition 548.17 and Definition 548.14. These results identify the public depth read with the displayed modulus and ledger rows, so the excluded host and global analytic coordinates have no source in the packet. □

Theorem 548.20 (Critical-line witness strip-modulus route factorization)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the zero-strip rows* Z, S *and the modulus route* M, R, Q *factor through the same transported packet* H, C, P, N*. Thus the supplied zero row, supplied critical-strip row, real-part row, line-modulus row, and rational comparison ledger are one finite* BHist *surface, not separate host assertions.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessStripModulusRouteFactorization`

Proof. Start with the supplied zero and strip rows. Theorems Definition 548.16 and Definition 548.15 keep Z and S on the same supplied-input history, transported through the displayed row H.

Now attach the modulus route. Theorems Definition 548.10 and Definition 548.11 bind M, R, Q to the same accepted packet and the same public read route.

The continuation, provenance, and local naming rows C, P, N are shared by both projections. Therefore the zero-strip input and modulus route factor through one transported packet, with no extra host equality or analytic completion coordinate. □

Theorem 548.21 (Critical-line witness local RH boundary)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *remains a supplied-input witness for the displayed zero-in-strip rows* Z, S*. It can expose the local modulus route* M, R, Q, H, C, P, N*, but it cannot classify all zeta zeros, construct a zero, export* RH*, or convert the local modulus route into an ambient analytic completeness theorem.*

Proof. Project the witness surface through $Z, S, M, R, Q, H, C, P, N$. These are all finite rows of the accepted packet, and none is a global zero-set classifier or a zero-producing constructor.

Use Definition 548.18 and Definition 548.31. The first theorem keeps the packet local to the supplied zero-in-strip input, and the second prevents the local witness from becoming a proof of RH.

The non-escape and ledger exactness results then close the boundary. Since the packet exports only the displayed witness rows and their route data, it cannot export a universal zero classifier or an ambient analytic theorem. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier local rh boundary`

Theorem 548.22 (Critical-line witness obligation closure package)**.** *For supplied zero-in-strip inputs, Definition 548.1, Definition 548.2, Definition 548.9, Definition 548.10, Definition 548.11, Definition 548.13, Definition 548.14, and Definition 548.15 jointly form the* $\mathsf{CriticalLineWitness}^{\uparrow}$ *obligation package. The package is exactly the finite surface* $Z, S, M, R, Q, H, C, P, N$ *and preserves the exclusions: no proof of* RH*, no zero production, no global zero-set classifier, no analytic continuation, no host equality, no quotient representative, and no analytic completeness.*

Proof. Project the carrier and local naming rows first. The carrier definition and NameCert obligations display the supplied zero row, critical-strip row, modulus route, real-part row, rational comparison ledger, transports, continuations, provenance, and local certificate.

Next use the compatibility and stability rows. The zero-strip compatibility theorem keeps Z and S tied to the same supplied history, and the real-part modulus stability theorem keeps every depth request attached to R, M, Q.

The rational ledger exactness theorem accounts for the half-line and dyadic comparison rows. Non-escape then removes the host equality, quotient, zero-production, global-classifier, analytic-continuation, and completeness channels.

Finally apply ledger exactness and zero-row source scope. They identify every public witness read with $Z, S, M, R, Q, H, C, P, N$, which is precisely the stated obligation package for supplied zero-in-strip inputs. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier obligation closure package`

Theorem 548.23 (Critical-line witness supplied input carrier row)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet must expose the supplied zeta-zero row Z and supplied critical-strip row S before any public read of the modulus row M. The modulus request therefore begins from the displayed supplied-input carrier surface and cannot replace Z, S by zero search, a global zero classifier, or host complex equality.*

Proof. Project the accepted carrier of Definition 548.1. The rows Z and S are the first supplied inputs of the finite packet, while M is read only as the later modulus row.

Use Definition 548.9 and Definition 548.15. They keep the zero and strip rows on the same displayed history and keep every zero-facing read inside $Z, S, M, R, Q, H, C, P, N$.

Finally apply Definition 548.17. Since each modulus request routes through the displayed packet after the supplied rows have been exposed, the request has no route to zero search, global classification, or host complex equality. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied input carrier row`

Theorem 548.24 (Critical-line witness comparison ledger admission)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the rational comparison ledger Q admits exactly the half-line, dyadic tolerance, and strict-comparison rows needed by the modulus row M. It admits no quotient representative, host real equality, hidden numeric oracle, or ambient analytic completeness row.*

Proof. Read the rational ledger through Definition 548.11. The available comparison surface is the finite ledger Q, consisting of the half-line, dyadic tolerance, and strict-comparison rows.

Attach the ledger to modulus requests by Definition 548.17. The row M consumes Q through the displayed route and does not read an external comparison oracle.

The nonclassification and non-escape boundaries of Definition 548.18 and Definition 548.13 remove the excluded analytic and host-level coordinates. Thus ledger admission is exactly the comparison surface required by M. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier comparison ledger admission`

Theorem 548.25 (Critical-line witness public consumer factorization)**.** *Every public consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the displayed rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The factorization may expose the supplied zero-in-strip input, the modulus route, the rational comparison ledger, transport, continuation, provenance, and local naming, but it cannot export RH*, produce zeros, classify all zeros, or invoke ambient analytic completeness.*

Proof. Begin with Definition 548.23. It places every public read after the supplied Z, S rows and before any admissible modulus use.

Use Definition 548.24 for the M, R, Q part of the route. The comparison data consumed by the modulus is exactly the finite ledger available in the carrier.

Finally apply Definition 548.18, Definition 548.21, and Definition 548.22. These results keep the consumer inside $Z, S, M, R, Q, H, C, P, N$ and preserve the nonclassification boundary for RH. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier public consumer factorization`

Theorem 548.26 (Critical-line witness root carrier scope)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through exactly the displayed rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The root carrier scope consumes no global zero classifier, no proof of RH*, no zero-producing search, no host complex equality, no quotient representative, and no ambient analytic completeness principle.*

Proof. Start from the carrier of Definition 548.1 and the NameCert obligations of Definition 548.2. They expose only the supplied zero row, supplied critical-strip row, modulus row, real-part row, rational comparison ledger, componentwise transports, displayed continuations, provenance, and local naming.

Use Definition 548.9 to bind the zero and strip rows to the same displayed history. Use Definition 548.10 to keep the modulus route attached to the same real-part row.

Apply Definition 548.13 and Definition 548.14. These rows exhaust the public packet and exclude global zero classification, zero search, host equality, quotient representatives, and analytic completeness from the root carrier scope. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root carrier scope`

Theorem 548.27 (Critical-line witness modulus ledger non-escape)**.** *The modulus row* M *and rational comparison ledger* Q *of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet cannot export host equality, quotient representatives, zero search, a global zero classifier, or analytic completeness. Their public reads remain confined to the supplied zero row* Z*, strip row* S*, real-part row* R*, componentwise transports* H*, continuations* C*, provenance* P*, and local naming* N*.*

Proof. Project the modulus and ledger rows from the carrier. The row M answers depth requests, while Q stores the half-line, dyadic tolerance, and strict comparison rows used by that answer.

Invoke Definition 548.11. The rational comparison surface is exactly the finite ledger consumed by M, so it has no off-ledger route to host equality, quotient representatives, or hidden numeric oracles.

Use Definition 548.17 and Definition 548.13. The depth route factors through R, Q, H, C, P, N, and the non-escape boundary removes zero search, global zero classification, and analytic completeness from the public read. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus ledger nonescape`

Theorem 548.28 (Critical-line witness strip-zero route exactness)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the supplied zeta-zero row* Z *and supplied critical-strip row* S *are the only sibling-origin rows for the displayed history inspected by the witness. Any public strip-zero read factors through* Z, S, H, C, P, N *before it reaches the modulus and rational ledger rows, and no additional sibling row may be substituted for the same history.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip zero route exactness`

Proof. Read the sibling-origin rows from Definition 548.1. The packet has one $\mathsf{ZetaZeros}^{\uparrow}$ coordinate Z and one $\mathsf{CritStrip}^{\uparrow}$ coordinate S.

Apply Definition 548.9. The displayed hsame transport in H is the only route tying Z and S to the same supplied history, and continuation through C only replays that route.

Use Definition 548.16 to identify the sibling sources and Definition 548.15 to keep the read inside $Z, S, M, R, Q, H, C, P, N$.

Finally apply Definition 548.14. Since the ledger contains no additional zero or strip coordinate for the same history, the strip-zero route is exact. □

Theorem 548.29 (Critical-line witness supplied-zero readback)**.** *Every zero-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *uses the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, and the displayed transport row* H*. The packet performs no zero search, constructs no new zero row, and substitutes no global zero classifier for the supplied input.*

Proof. Project the carrier of Definition 548.1 and the carrier admission row of Definition 548.2. The zero-facing coordinates are the supplied zero row Z, the supplied critical-strip row S, and the componentwise transport row H.

Apply Definition 548.9. It identifies the displayed hsame row in H as the route tying Z and S to the same supplied complex history.

Use Definition 548.15 and Definition 548.13. They keep the read inside $Z, S, M, R, Q, H, C, P, N$ and exclude zero production, host equality, quotient representatives, global classifiers, and ambient analytic completeness. Hence the zero read is supplied-row readback only. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied zero readback`

Theorem 548.30 (Critical-line witness modulus request factorization)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each depth request for the line modulus factors through the Nat depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, componentwise transport* H*, and the displayed* Cont *route* C *before the value* $M(n)$ *is read.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus request factorization`

Proof. Begin with the modulus discipline row in Definition 548.2. A request at depth $n : \mathsf{Nat}^{\uparrow}$ is answered by the modulus row M and is not answered by an external analytic oracle.

Use Definition 548.10 to attach the request to the real-part row R under componentwise transport H. Then apply Definition 548.11: the rational data consumed by the request is exactly the ledger Q, including the half-line row, dyadic tolerance row, and strict comparison row.

Finally use Definition 548.12 and Definition 548.17. They place the displayed Cont route C before the public read of $M(n)$ and keep the request inside R, Q, H, C, P, N, with no quotient representative, host real equality, or ambient completeness coordinate. □

Theorem 548.31 (Critical-line witness RH nonproof boundary)**.** *An accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *is a witness for the supplied zero-in-strip input carried by* Z *and* S*. It exports no universal zero classifier, no proof of* RH*, no zero-producing search, and no ambient analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh nonproof boundary`

Proof. Project the supplied input rows from Definition 548.1. The packet starts with Z and S; these rows are inputs to the witness surface, not results produced by the packet.

Apply Definition 548.29 for the zero-facing route and Definition 548.30 for the modulus route. Together they show that every public read remains inside the supplied zero, strip, real-part, rational ledger, transport, continuation, package, and local naming coordinates.

Use Definition 548.13, Definition 548.14, and Definition 548.18. These rows exclude global zero-set classification, zero production, RH, host equality, quotient representatives, analytic continuation, and ambient completeness from the packet surface. The chapter therefore records only the supplied-input witness boundary. □

Theorem 548.32 (Critical-line witness zero-strip route totality)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every public route that reads the supplied zero row* Z *together with the critical-strip row* S *factors through the displayed transport coordinate* H *and continuation coordinate* C*. The route may consume the supplied zero and strip rows as the same displayed history, but it cannot replace that route by host complex equality or by a zero-producing search.*

Proof. Start with the carrier of Definition 548.1 and the local NameCert obligations of Definition 548.2. They list Z, S, H, C as displayed rows of the same finite packet.

Apply Definition 548.9. The zero-strip link is the componentwise hsame row inside H, and Definition 548.28 keeps every strip-zero read inside Z, S, H, C, P, N before the modulus and rational ledger are inspected.

Finally use Definition 548.13 and Definition 548.29. Since the packet has no host complex equality coordinate and no zero-search coordinate, the displayed H, C route is total for every accepted zero-strip read. □

Theorem 548.33 (Critical-line witness depth ledger coverage)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each depth request is covered by the displayed rows* M, R, Q, H, C, P, N*. In particular, the half-line row and dyadic tolerance row used by the request are rows of* Q*, and the request has no route to a quotient representative, host real equality, hidden numeric oracle, or ambient analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier depth ledger coverage`

Proof. Begin with the modulus discipline row of Definition 548.2. A depth request is answered by M and is read against the real-part row R and rational comparison ledger Q.

Apply Definition 548.11. The half-line row, dyadic tolerance row, and strict comparison row are exactly the rational rows available inside Q.

Use Definition 548.17 and Definition 548.30. These rows place H and C on the route before the public read of $M(n)$ and keep provenance and local naming in P, N.

The non-escape boundary of Definition 548.13 then excludes quotient representatives, host real equality, hidden numeric oracles, and ambient completeness. Thus M, R, Q, H, C, P, N cover every depth request. □

Theorem 548.34 (Critical-line witness downstream refusal package)**.** *Every downstream* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet may cite the local witness only as the supplied-input packet*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer cannot read from it a global RH *classifier, a zero construction, an analytic continuation proof, a completeness principle, host equality, or a quotient representative.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream refusal package`

Proof. Project the public packet through Definition 548.1. The exported rows are exactly $Z, S, M, R, Q, H, C, P, N$.

Use Definition 548.14 and Definition 548.15. They identify the ledger as a supplied zero-in-strip witness packet and rule out zero production, host equality, quotient representatives, analytic continuation, and ambient completeness.

Apply Definition 548.18 and Definition 548.31. These theorems keep the packet from becoming a global zero-set classifier or a proof of RH.

Finally use Definition 548.22. It repacks the same carrier, NameCert obligations, strip-zero compatibility, modulus-depth routing, non-escape boundary, and ledger exactness as the finite obligation surface available to downstream consumers. □

Theorem 548.35 (Critical-line witness carrier source)**.** *Every downstream read of an accepted* CriticalLineWitness$^{\uparrow}$ *carrier begins with the finite* BHist *packet*

$$W = (Z, S, M, R, Q, H, C, P, N).$$

The rows Z, S, M, R, Q supply the zero, strip, modulus, real-part, and rational comparison data, while H, C, P, N supply the displayed transports, consumer routes, package provenance, and local NameCert *row. No read starts from a host zero search, quotient representative, or ambient analytic-completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier carrier source`

Proof. Project the carrier from Definition 548.1. The tuple displays exactly the rows $Z, S, M, R, Q, H, C, P, N$.

Use the carrier-admission row of Definition 548.2. It makes the finite packet the entry point for zero, strip, modulus, real-part, comparison, transport, route, provenance, and naming reads.

The non-escape discipline in the same obligation list excludes host zero search, quotient representatives, and analytic completeness. Therefore every downstream read begins with the displayed BHist packet. □

Theorem 548.36 (Critical-line witness classifier separation)**.** *The classifier-facing content of an accepted* CriticalLineWitness$^{\uparrow}$ *carrier separates the supplied zero row Z, the critical-strip row S, and the critical-line modulus row M. The displayed* hsame *and* Cont *rows may transport and replay their alignment, but the packet does not identify these rows by host equality, collapse them into an* RH *classifier, or turn the supplied witness into a global zero-set classification.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier classifier separation`

Proof. Read Z, S, and M as separate coordinates of the carrier. The zero row, strip row, and modulus row have distinct jobs in the finite packet.

Apply Definition 548.9 for the displayed alignment of Z and S, and apply Definition 548.30 for the route from depth requests through M, R, Q.

Finally use Definition 548.18. The classifier-facing rows remain separated by their displayed routes and cannot become host equality, a global RH classifier, or a zero-set classification. □

Theorem 548.37 (Critical-line witness NameCert ledger)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet, the local* NameCert *ledger is exhausted by carrier admission, zero-strip compatibility, modulus discipline, non-escape, and ledger exactness for the rows $Z, S, M, R, Q, H, C, P, N$. A consumer may cite these rows as a finite supplied-witness certificate and may not add a hidden analytic-continuation row, host equality row, or zero-producing row.*

Proof. Start from Definition 548.2. It lists the five local obligation rows for the displayed carrier.

Use Definition 548.14 to identify the public witness surface with the zero, strip, modulus, real-part, rational comparison, transport, route, provenance, and naming rows.

The non-escape and downstream refusal theorems exclude analytic continuation, host equality, and zero production. Thus the local NameCert ledger is exactly the finite packet obligation surface. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert ledger`

Theorem 548.38 (Critical-line witness real-strip handoff)**.** *Every real-part read used by an accepted* CriticalLineWitness$^{\uparrow}$ *packet factors through the supplied critical-strip row S, the real-part row R, the rational comparison ledger Q, and the modulus row M, with transport and replay controlled by H and C. The handoff records a finite strip-to-real witness route only; it does not close analytic continuation, host complex equality, or real completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier real strip handoff`

Proof. Begin with the strip row S and the real-part row R in the carrier. The strip row supplies the displayed critical-strip context, and R is the only real-part coordinate consumed by the modulus.

Apply Definition 548.10. It keeps transported depth requests aligned with the same R row and the comparison ledger Q.

Then use Definition 548.11. The rational comparison rows are precisely the finite rows used by M, so the handoff is a finite strip-to-real route and not an analytic-continuation or completeness principle. □

Theorem 548.39 (Critical-line witness zeta-zero boundary)**.** *The supplied* ZetaZeros$^{\uparrow}$ *row Z in an accepted* CriticalLineWitness$^{\uparrow}$ *packet is a boundary row for the witness surface: downstream consumers may read it only together with S, M, R, Q, H, C, P, N as the finite supplied-zero packet, and cannot extract a zero constructor, global zeta-zero enumeration,* RH *proof, or analytic-continuation certificate from it.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta zero boundary`

Proof. Project the zero row Z from the finite carrier. It is stored as supplied input, not as an output of a search or construction coordinate.

Use Definition 548.15. The zero row remains tied to the same strip, modulus, real-part, comparison, transport, route, provenance, and naming rows.

Finally apply Definition 548.31 and Definition 548.34. These exclude zero construction, global enumeration, RH proof, and analytic-continuation export from the displayed packet. □

Theorem 548.40 (Critical-line witness zero-source totality)**.** *Every accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ reads its zero row only through the displayed source coordinates Z, S, R, Q, H, C, P, N. The zero-facing source route is therefore a finite* ZetaZeros$^{\uparrow}$–CritStrip$^{\uparrow}$–Real$^{\uparrow}$–Rat$^{\uparrow}$ *route, with componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local* NameCert *evidence, and not a host zero search or an ambient analytic-continuation coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero source totality`

Proof. Project the carrier of Definition 548.1. The accepted packet exposes the zero row Z together with the strip row S, real-part row R, rational comparison ledger Q, transport row H, route row C, package row P, and local naming row N.

Apply Definition 548.39. It already makes Z a supplied boundary row, readable only with the remaining displayed packet coordinates.

Use Definition 548.29 and Definition 548.38. These rows identify the zero read with the supplied $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ sources and carry the real and rational coordinates through R, Q, H, C, P, N.

The non-escape row of Definition 548.34 then excludes host zero search, analytic continuation, quotient representatives, and ambient completeness. Thus every zero-source read is total through the displayed finite coordinates. □

Theorem 548.41 (Critical-line witness strip-modulus ledger triad)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the critical-strip row* S*, modulus request row* M*, and rational comparison ledger* Q *form one BEDC-visible triad. The triad is transported only by the displayed* hsame *row* H*, replayed only by the displayed* Cont *row* C*, and packaged only through* P *and* N*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip modulus ledger triad`

Proof. Read the strip, modulus, and rational rows from the carrier. The packet lists S, M, Q as public coordinates of the same finite BHist witness.

Apply Definition 548.30. Every depth request for M factors through the real-part row, the rational comparison ledger, the transport row, and the continuation route before $M(n)$ is read.

Use Definition 548.32 and Definition 548.33. The first keeps the strip side on the displayed H, C route, while the second exhausts every depth request by M, R, Q, H, C, P, N.

Finally apply Definition 548.22. The carrier, naming, transport, continuation, package, and ledger rows are the finite obligation surface, so the strip row, modulus request, and rational ledger remain one visible triad under hsame, Cont, and Pkg. □

Theorem 548.42 (Critical-line witness RH consumer boundary package)**.** *Every downstream* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives a finite witness boundary consisting of the zero-source route, the strip-modulus ledger triad, the nonproof boundary, the downstream-refusal row, and the local obligation package. The consumer receives no* RH *proof, no global zero-set classifier, no analytic-continuation certificate, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer boundary package`

Proof. Begin with Definition 548.40. It confines every zero-facing read to the displayed $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ source coordinates and their H, C, P, N routes.

Next use Definition 548.41. It packages the strip row, modulus request, and rational comparison ledger as one finite BEDC-visible triad.

Apply Definition 548.31 and Definition 548.34. These rows prevent downstream consumers from reading the supplied packet as an RH proof, zero construction, analytic-continuation proof, completeness principle, host equality, or quotient representative.

Finally apply Definition 548.22. The prior rows repack into the finite carrier, NameCert, transport, continuation, package, non-escape, and ledger surface. Hence RH-facing consumers receive a finite witness boundary and no RH-level conclusion. □

Theorem 548.43 (Critical-line witness root zero-modulus carrier)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the rows* $Z, S, M, R, Q, H, C, P, N$ *are the only admitted zero/modulus packet available to root-facing consumers. The packet admits the supplied zero row, critical-strip row, finite modulus row, real-part row, rational comparison ledger, transport row, replay row, package row, and local naming row, and it admits no zero search, RH proof, host equality, quotient representative, analytic-continuation certificate, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero modulus carrier`

Proof. Project the carrier of Definition 548.1. The displayed coordinates are precisely $Z, S, M, R, Q, H, C, P, N$.

Apply Definition 548.42. It packages downstream access as the zero-source route, strip-modulus ledger triad, nonproof boundary, refusal row, and local obligation package.

The nonproof and refusal rows rule out zero search, an RH proof, host equality, quotient representatives, analytic continuation, and ambient completeness. Thus the admitted root packet is exactly the finite zero/modulus packet displayed by the carrier. □

Theorem 548.44 (Critical-line witness root strip readback)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing strip readback is bound to the supplied zero row* Z *and the critical-strip row* S*, then replayed through* M, R, Q, H, C, P, N*. The readback does not compare host complex numbers, identify quotient representatives, or construct a global zero-set classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip readback`

Proof. Start with Definition 548.40. It confines zero-facing reads to the supplied $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ source coordinates and their visible routes.

Use Definition 548.41. The critical-strip row S, modulus row M, and rational ledger Q remain one finite BEDC-visible triad under H, C, P, N.

Finally apply Definition 548.42. It keeps the readback at witness-boundary level and prevents the supplied zero and strip rows from becoming host complex equality, quotient identification, or a global zero classifier. □

Theorem 548.45 (Critical-line witness root consumer threshold)**.** *Every downstream* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *reads root information only through the finite modulus row* M *and rational comparison threshold row* Q*, together with the supplied zero, strip, real-part, transport, replay, package, and naming rows* Z, S, R, H, C, P, N*. Such a consumer cannot read an* RH *proof, perform zero search, or import an ambient analytic classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root consumer threshold`

Proof. Begin with the RH consumer boundary package. It states that downstream consumers receive a finite witness boundary and no RH-level conclusion.

Next use Definition 548.41. It places the threshold information in the strip-modulus ledger triad and routes comparison through the rational ledger Q.

The root zero-modulus carrier theorem identifies the only admitted rows as $Z, S, M, R, Q, H, C, P, N$. Therefore every root-facing downstream read passes through the finite modulus and comparison threshold rows, with no access to an RH proof, zero search, or ambient analytic classifier. □

Theorem 548.46 (Critical-line witness supplied-zero modulus obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet anchored in* BHist*. The local supplied-zero obligation binds the supplied zero row Z, the critical-strip row S, the real-part row R, the rational comparison ledger Q, and the modulus row M as one finite witness packet. Every modulus read from M is therefore sourced through the displayed Z, S, R, Q rows and the visible H, C, P, N routes, not through an ambient zero search, analytic-continuation oracle, or global* RH *classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied zero modulus obligation`

Proof. Start with Definition 548.40. It anchors the zero-facing source route in the displayed Z, S, R, Q, H, C, P, N coordinates of the accepted BHist packet.

Next apply Definition 548.41. The strip row, modulus request row, and rational comparison ledger are read as one BEDC-visible triad, with M connected to S, R, Q only through the displayed transport, continuation, package, and naming routes.

Finally use Definition 548.22. It repacks the same rows as the local finite obligation surface, so the supplied zero, strip, real-part, rational ledger, and modulus obligations remain attached to W and do not open an ambient classifier. □

Theorem 548.47 (Critical-line witness transport replay obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet anchored in* BHist*. The local transport obligation binds componentwise* hsame *evidence from H, displayed* Cont *replay from C, package provenance from* Pkg *through P, and local* NameCert *replay through N to the same zero-strip-modulus ledger. The replay may transport the finite rows, but it may not replace them with host equality, quotient representatives, analytic continuation, or an external naming certificate.*

Proof. Read H, C, P, N together with the zero, strip, modulus, real-part, and rational rows of the carrier. These are coordinates of the same accepted BHist packet, so the transport and replay evidence is local to W.

Apply Definition 548.41. It keeps the S, M, Q triad under the displayed hsame and Cont rows and packages the triad only through P and N.

Then use Definition 548.22. The theorem identifies the carrier, naming, transport, continuation, package, non-escape, and ledger rows as the available obligation surface.

The non-escape part of that package rules out host equality, quotient representatives, analytic continuation, and external naming. Thus the hsame, Cont, Pkg, and NameCert replay is exactly the local replay of the finite ledger. □

Theorem 548.48 (Critical-line witness downstream source obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet anchored in* BHist*. Every* RH*-facing downstream source read factors through the finite supplied-zero and modulus packet consisting of $Z, S, M, R, Q, H, C, P, N$. Such a consumer receives no* RH *proof, no global zero classifier, no host equality, no quotient representative, no analytic-continuation certificate, and no ambient completeness principle.*

Proof. Begin with Definition 548.42. It states the downstream consumer boundary as the zero-source route, strip-modulus ledger triad, nonproof boundary, refusal row, and local obligation package.

Use Definition 548.40 to make the zero source explicit, and use Definition 548.41 to make the modulus source explicit. Together they factor every source read through $Z, S, M, R, Q, H, C, P, N$.

Finally apply Definition 548.22. The package is finite and local to the accepted witness, so downstream reads cannot import an RH proof, global zero classifier, host equality, quotient representative, analytic-continuation certificate, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream source obligation`

Theorem 548.49 (Critical-line witness carrier admission obligation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the supplied zero row* Z*, critical-strip row* S*, real-part row* R*, rational comparison ledger* Q*, transport row* H*, continuation row* C*, package provenance* P*, and local* NameCert *row* N *combine with the modulus row* M *into the accepted* BHist *carrier surface. Carrier admission exposes exactly those finite rows and no zero search,* RH *proof, host equality, quotient representative, or ambient analytic-completeness row.*

Proof. Apply the carrier-admission row theorem to identify $Z, S, M, R, Q, H, C, P, N$ as the admitted packet. Use the supplied-input carrier row to keep the zero, strip, real-part, and rational comparison rows inside the same accepted BHist surface.

Next apply the NameCert ledger theorem and the transport replay obligation. They attach the displayed hsame, Cont, Pkg, and local NameCert rows to the same packet rather than to an external analytic certificate.

Finally use the downstream source obligation. It rules out RH proof data, zero search, host equality, quotient representatives, and analytic-completeness rows as carrier coordinates, so the admitted carrier surface is exactly the finite packet displayed above. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier admission obligation`

Theorem 548.50 (Critical-line witness zero-strip compatibility obligation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the supplied zero row* Z *and critical-strip row* S *are consumed together before any modulus row* M*, rational comparison ledger* Q*, or* RH*-facing read is available. The zero and strip rows therefore form a local compatibility obligation over the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a global zero-set classifier.*

Proof. Use strip-zero compatibility and strip-zero route exactness. These theorems bind Z and S to the same supplied complex-history route through the displayed transport and continuation rows.

Then apply zero-source totality together with root strip readback. They force every zero-facing read to pass through the supplied zero and strip rows before the modulus or rational comparison ledger can be inspected.

The RH consumer boundary package keeps the combined read at witness-boundary level. Thus the compatibility obligation is consumed before any modulus or RH-facing read and cannot become a global zero classifier. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip compatibility obligation`

Theorem 548.51 (Critical-line witness public route)**.** *Every public* $\mathsf{CriticalLineWitness}^{\uparrow}$ *read factors through the supplied-zero row* Z*, the strip-modulus ledger* S, M*, the rational comparison ledger* Q*, the zeta-zero boundary, the* RH *nonproof boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row before it can be consumed as a critical-line witness. The route exports no* RH *proof, global zero classification, quotient equality, or hidden analytic continuation.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPublicRoute`

Proof. Begin with the public consumer factorization of Definition 548.25. It requires every public witness consumer to stay inside the supplied packet rows before reading the witness conclusion.

Use strip-zero route exactness, supplied-zero readback, and the zeta-zero boundary: Definition 548.28, Definition 548.29, and Definition 548.39. These rows keep the supplied zero and critical-strip side on the finite Z, S, H, C, P, N route.

Next apply the rational and modulus side through the existing strip-modulus ledger surface, then package the downstream boundary with Definition 548.42. The consumer receives the witness as a finite zero-source, strip-modulus, rational-comparison, transport, replay, provenance, and local naming route.

The cited boundary package is explicitly nonproof-facing. Hence no RH proof, global zero classification, quotient equality, or hidden analytic-continuation certificate is exported by the public route. □

Theorem 548.52 (Critical-line witness ledger non-escape obligation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the modulus row M, rational comparison ledger Q, componentwise* hsame *row H, continuation row C, package row P, and local* NameCert *row N exhaust the local ledger together with Z, S, R. The ledger excludes an* RH *proof, zero search, analytic-continuation certificate, quotient representative, host equality row, and ambient analytic-completeness row.*

Proof. Apply the modulus-ledger non-escape theorem and the rat comparison ledger exactness theorem. They identify M and Q as the finite threshold and comparison route available to the witness.

Next use the transport replay obligation. It binds H, C, P, N to that same zero-strip-modulus ledger and prevents the replay rows from replacing the ledger with host equality, quotient data, or external naming.

Finally apply the downstream refusal package and downstream source obligation. These results exclude RH proof data, zero search, analytic continuation, quotient representatives, host equality, and ambient completeness from all downstream reads, so the displayed finite rows exhaust the local ledger. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessLedgerNonescapeObligation`

Theorem 548.53 (Critical-line witness scoped strip source)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The scoped strip-source route binds the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, the* $\mathsf{CritStrip}^{\uparrow}$ *row S, the* $\mathsf{Nat}^{\uparrow}$ *depth requests for M, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, and the local* BHist, hsame, Cont, Pkg, *and* NameCert *rows into one finite source surface. The route asserts only that the supplied zero-in-strip input is the source of the displayed witness packet; it does not assert* RH, *analytic continuation, or a completed zero classification.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessScopedStripSourcePackage`

Proof. Begin with the supplied zero-strip transport lock. It keeps the $\mathsf{ZetaZeros}^{\uparrow}$ row Z and the $\mathsf{CritStrip}^{\uparrow}$ row S attached to the same displayed history through H and C.

Next attach the depth and comparison route. The modulus comparison route and root comparison ledger force each $\mathsf{Nat}^{\uparrow}$ request for M to read the $\mathsf{Real}^{\uparrow}$ row R and the $\mathsf{Rat}^{\uparrow}$ ledger Q inside the same finite packet.

Finally apply the downstream refusal ledger. The package row P and local certificate N name the source surface, while the refusal rows exclude RH proof data, analytic continuation, host equality, quotient representatives, and global zero classification. □

Theorem 548.54 (Critical-line witness rational height ledger)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every displayed height read of the witness factors through a finite natural depth request and the rational comparison ledger. The read uses only the* $\mathsf{Nat}^{\uparrow}$ *request for* M*, the* $\mathsf{Rat}^{\uparrow}$ *rows inside* Q*, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it does not use a host real equality, quotient representative, hidden numeric oracle, analytic-continuation certificate, or completed zero-set classifier.*

Proof. Project the modulus and ledger coordinates from the packet. A height request is first a finite $\mathsf{Nat}^{\uparrow}$ depth request to M, not an infinite analytic search.

Read the answer through the rational comparison ledger. The row Q contains the half-line, dyadic tolerance, and strict comparison rows used by the modulus route, and the real-part row R is the only $\mathsf{Real}^{\uparrow}$ source visible to that comparison.

Use transport replay and ledger exactness to close the route. The rows H, C, P, N carry the natural request and rational ledger inside the same accepted packet, so no host equality, quotient representative, hidden numeric oracle, analytic continuation, or global zero classification enters the height read. □

Theorem 548.55 (Critical-line witness NameCert scope package)**.** *The scoped strip-source theorem Definition 548.53 and the rational-height ledger theorem Definition 548.54 assemble the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *scope for an accepted packet* $W = (Z, S, M, R, Q, H, C, P, N)$*. The scope consists exactly of the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *and* $\mathsf{CritStrip}^{\uparrow}$ *rows, the* $\mathsf{Nat}^{\uparrow}$ *depth route, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, the* $\mathsf{Real}^{\uparrow}$ *real-part row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *evidence.*

Proof. First use scoped strip source. It binds Z and S as supplied source rows and keeps the zero-strip route inside the displayed transport and continuation surface.

Next use the rational height ledger. It identifies each witness-height read with the finite natural request, the real-part row, and the rational ledger Q.

Finally repack the two rows into the local naming surface. The packet's BHist, hsame, Cont, Pkg, and NameCert coordinates record exactly those source and ledger routes, while the existing refusal boundary excludes RH, analytic continuation, host equality, quotient representatives, and completed zero classification. □

Theorem 548.56 (Critical-line witness scoped source lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The scoped source lock of* W *is the finite route through the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*,* $\mathsf{Real}^{\uparrow}$ *real-part row* R*,* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, accepted* BHist *carrier surface, componentwise* hsame *row* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N*. Every source read for* W *is bound to those rows and cannot be replaced by host equality, a global zero-set classifier, analytic continuation, quotient representation, or ambient completeness.*

Proof. Start with carrier admission for the accepted packet. The carrier exposes $Z, S, M, R, Q, H, C, P, N$ as the available finite rows and gives no coordinate for host equality, quotient data, analytic continuation, or ambient completeness.

Use zero-strip compatibility to bind Z and S to the same supplied source over $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$. Transport then occurs only through the displayed BHist and hsame rows, and continuation reads occur only through C.

Finally apply the modulus comparison route totality and RH-refusal ledger completeness. They attach R, Q, P, N to the same route and exclude global zero classification and RH proof data from the source surface. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier scoped source lock`

Theorem 548.57 (Critical-line witness scoped ledger exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The ledger exactness surface for W consists exactly of the modulus row M, the* $\mathsf{Real}^{\uparrow}$ *row R, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows H, C, P, N. Any downstream comparison or modulus consumer must factor through that surface after reading the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *and* $\mathsf{CritStrip}^{\uparrow}$ *rows Z, S.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier scoped ledger exactness`

Proof. Project the modulus and comparison rows from the carrier. The modulus row M, real-part row R, and rational comparison ledger Q are displayed coordinates of the same accepted packet.

Apply the modulus comparison route totality. It forces comparison reads through M, R, Q, H, C, P, N and makes the route total without adding a hidden numeric oracle, host real equality, quotient representative, or analytic-continuation certificate.

The root comparison ledger then supplies exactness for downstream consumers. Together with the scoped source lock, the comparison surface is the finite ledger following the supplied Z, S source rows and no second ledger is available. □

Theorem 548.58 (Critical-line witness scoped nonescape boundary)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the scoped nonescape boundary is the refusal row generated by the same* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *dependencies. It permits downstream use of the finite source lock and ledger exactness, but it exports no* RH *proof, no zero search, no global zero classifier, no host equality, no quotient representative, no analytic-continuation certificate, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier scoped nonescape boundary`

Proof. Begin with the scoped source lock. It identifies the only source dependencies available to a downstream consumer as Z, S, R, Q, H, C, P, N together with the accepted packet row.

Use scoped ledger exactness for every modulus or comparison read. The ledger route stays inside M, R, Q, H, C, P, N and therefore cannot become an independent analytic or quotient route.

Finally invoke RH-refusal ledger completeness and the root refusal route. These refusal rows already cover RH proof data, zero search, global zero classification, host equality, quotient representatives, analytic continuation, and ambient completeness, so the scoped boundary is closed under the stated nonescape obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CriticalLineWitness}^{\uparrow}$ packet is generated from a supplied zeta-zero row, a supplied critical-strip row, an on-critical-line modulus row, a real-part row, rational half-line and dyadic comparison rows, displayed transports, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$, publicClosed)

Scope closed: The public scope records the finite carrier, witness-ledger, and public route surface for supplied zero-in-strip inputs over $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove RH, produce zeros, close the global zero-set classifier, construct analytic continuation, assert host equality, select quotient representatives, or invoke ambient analytic completeness.

Upgrade path: Scoped closure requires grounding the carrier admission, zero-strip compatibility, and ledger nonescape obligations against the named $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Theorem 548.59 (Critical-line witness source normalization interface)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet first exposes the supplied zero row from* $\mathsf{ZetaZeros}^{\uparrow}$*, the critical-strip row from* $\mathsf{CritStrip}^{\uparrow}$*, the depth row from* $\mathsf{Nat}^{\uparrow}$*, the rational comparison row from* $\mathsf{Rat}^{\uparrow}$*, and the real-part row from* $\mathsf{Real}^{\uparrow}$ *through the finite surface*

$$Z, S, M, R, Q, H, C, P, N.$$

No downstream RH-boundary consumer may read a modulus, comparison, or naming row before this supplied source surface is visible.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source normalization interface`

Proof. Start with the local NameCert obligations of Definition 548.2. They make the accepted carrier and its naming row a finite packet over $Z, S, M, R, Q, H, C, P, N$.

Apply the root-unblock source triad of Definition 548.3. It exposes the supplied zero row, critical-strip row, and source-facing comparison row before downstream use.

Finally use the terminal zero-source totality of Definition 548.114. The terminal source route begins at the displayed $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ rows and reaches the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads only through the same structural rows H, C, P, N. □

Theorem 548.60 (Critical-line witness modulus ledger interface)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, all modulus and comparison reads factor through*

$$M, R, Q, H, C, P, N$$

after the supplied zero and strip rows Z, S *have been exposed. The modulus row* M *answers only displayed* $\mathsf{Nat}^{\uparrow}$ *depth requests, the real-part row* R *supplies the* $\mathsf{Real}^{\uparrow}$ *read, and the rational ledger* Q *supplies the* $\mathsf{Rat}^{\uparrow}$ *comparison boundary.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus classifier join`

Proof. First expose the source rows by Definition 548.59. This places every modulus request downstream of the supplied $\mathsf{ZetaZeros}^{\uparrow}$ zero row and $\mathsf{CritStrip}^{\uparrow}$ strip row.

Apply the NameCert obligations of Definition 548.2. The modulus-discipline and ledger-exactness rows attach M to the displayed depth request and attach R, Q to the real-part and rational comparison reads.

Then use Definition 548.114. It keeps terminal consumers on the same route rows H, C, P, N, so the modulus and comparison interface cannot bypass the supplied zero-strip rows. □

Theorem 548.61 (Critical-line witness root source/modulus formal boundary)**.** *The checked source-normalization row and source-modulus classifier-join row expose only the finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

Consequently the root source/modulus interface leaves RH *proof content, global zero classification, zero search, analytic continuation, ambient completeness, host equality, and quotient representatives outside the formal boundary read by* $\mathsf{CriticalLineWitness}^{\uparrow}$ *consumers.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source modulus interface formal boundary`

Proof. Apply the source-normalization interface, Definition 548.59. It exposes the supplied zero row, critical-strip row, depth row, rational comparison row, and real-part row only through the displayed packet $Z, S, M, R, Q, H, C, P, N$.

Next use the modulus ledger interface, Definition 548.60. The modulus and comparison reads remain downstream of the supplied zero-strip rows and factor through the same finite route.

Finally apply the RH refusal interface, Definition 548.62. Since the refusal interface is attached to exactly the same source/modulus rows, no downstream consumer can read RH proof content, global zero classification, zero search, analytic continuation, ambient completeness, host equality, or quotient representatives from the checked interface. □

Theorem 548.62 (Critical-line witness RH refusal interface)**.** *The combined critical-line witness interface over $Z, S, M, R, Q, H, C, P, N$ refuses* RH *proof content, global zero classification, zero search, host equality, quotient representatives, analytic continuation, and ambient completeness. It exports only the supplied zero-strip rows, modulus row, real-part row, rational comparison ledger, transports, continuations, provenance, and local naming.*

Proof. Use Definition 548.59 and Definition 548.60 to assemble the source and comparison sides of the finite packet. Together they expose exactly $Z, S, M, R, Q, H, C, P, N$.

Apply the terminal RH-refusal exactness of Definition 548.116. It identifies the RH-facing terminal boundary with the finite packet and excludes proof content, global classification, search, quotient representatives, analytic continuation, and completeness.

The same exclusion also removes host equality as a source for the packet rows. Hence downstream RH-boundary consumers receive a refusal interface rather than an analytic or global-zero certificate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal interface`

Theorem 548.63 (Critical-line witness source/modulus consumer determinacy)**.** *Let two* CriticalLineWitness$^{\uparrow}$ *consumer reads expose the same finite rows*

$$Z, S, M, R, Q, H, C, P, N.$$

Then they determine the same source/modulus boundary before any RH-boundary request is inspected: the supplied zero row Z, critical-strip row S, depth/modulus row M, real-part row R, rational comparison row Q, componentwise transport H, replay C, provenance P, and local naming row N are fixed as the shared consumer interface. The determinacy does not expose RH *proof content, global zero classification, zero search, analytic continuation, host equality, quotient representatives, or ambient completeness.*

Proof. First apply Definition 548.59 to both consumer reads. Since the two reads expose the same $Z, S, M, R, Q, H, C, P, N$ rows, the supplied zero and critical-strip source boundary is identical before any downstream request is considered.

Next use Definition 548.60. The depth/modulus, real-part, and rational comparison rows are downstream of the same source rows and are read through the same structural rows, so the source/modulus ledger cannot split into two consumer boundaries.

Apply Definition 548.61. It identifies the formal boundary with exactly the displayed finite route, preventing either read from importing a second analytic or host-side source.

Finally use Definition 548.62. Because RH proof content, global zero classification, zero search, analytic continuation, host equality, quotient representatives, and ambient completeness are refused at the same finite boundary, the two reads agree before any RH-boundary request is inspected. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus consumer determinacy`

Theorem 548.64 (Critical-line witness root source-modulus admission)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. A root consumer may admit a source-modulus read only after the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison row Q, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows are visible. The admitted route has no* RH *proof, zero-search oracle, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source modulus admission`

Proof. Begin with the source normalization interface of Definition 548.59. It exposes the supplied zero, strip, depth, rational-comparison, and real-part rows before any downstream boundary may request a modulus read.

Next apply the modulus ledger interface of Definition 548.60. It places every modulus and comparison request on M, R, Q, H, C, P, N after the supplied source rows have been exposed.

Finally use the root non-escape package of Definition 548.69. The admitted source-modulus read is therefore exactly the finite BHist, hsame, Cont, Pkg, and NameCert packet route, with no analytic or host-level escape. □

Theorem 548.65 (Critical-line witness zero-strip budget factorization)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zero-strip budget request factors through the supplied Z, S rows and the same modulus ledger M, R, Q, H, C, P, N. The factorized route reads the displayed* BHist *carrier rows, the componentwise* hsame *transport, the* Cont *replay route, the* Pkg *provenance row, and the local* NameCert *row, and it exports no external zeta-zero enumeration, zero-search route, quotient representative, host equality, analytic-continuation certificate, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip budget factorization`

Proof. Use the zero-strip carrier totality theorem Definition 548.73. It identifies the root carrier exactly with $Z, S, M, R, Q, H, C, P, N$ and removes every non-displayed analytic or host coordinate from the packet.

Apply the zero-strip classifier transport theorem Definition 548.74. Any replacement of the supplied zero-strip classifier pair must use the same H, C, P, N route, so the budget request cannot bypass hsame, Cont, Pkg, or NameCert.

The root source-modulus admission theorem Definition 548.64 then supplies the shared modulus side. Hence the zero-strip budget factors through the finite source and modulus rows rather than through an independent zeta or host-real interface. □

Theorem 548.66 (Critical-line witness zeta-continuation readback non-escape)**.** *Every zeta-continuation readback of an accepted* CriticalLineWitness$^{\uparrow}$ *root packet is exhausted by the displayed packet $Z, S, M, R, Q, H, C, P, N$ and by the local* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The readback may cite the sibling zeta-continuation boundary already named by the chapter, but it exports no proof of* RH, *no analytic-continuation theorem, no global zero classifier, no quotient representative, no host equality, and no ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta continuation readback nonescape`

Proof. Start with the zeta dependency lattice of Definition 548.169. It places every zero-strip or critical-line read behind the existing $\mathsf{ZetaZeros}^{\uparrow}$ and RH-boundary surfaces before the local modulus witness is public.

Next apply the downstream source package of Definition 548.179. That package combines the local finite rows with the sibling $\mathsf{ZetaContinuationWitness}^{\uparrow}$ source lock, while still exposing only $Z, S, M, R, Q, H, C, P, N$ to downstream consumers.

Finally combine the zero-strip budget factorization of Definition 548.65 with the RH refusal interface of Definition 548.62. The readback remains a finite BEDC packet route through BHist, hsame, Cont, Pkg, and NameCert, so no analytic-continuation theorem or global zero certificate can escape from it. □

Theorem 548.67 (Critical-line witness root zero-strip route)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The supplied zero row Z and critical-strip row S route through the displayed* hsame *row H and* Cont *row C before any modulus, rational comparison, or downstream RH-facing read is exposed. No consumer may replace this route by zero search, host equality, a quotient representative, analytic continuation, or an ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero strip route`

Proof. Project the carrier of Definition 548.1. The rows Z and S are displayed coordinates of the accepted BHist packet, while H and C are the local transport and replay rows that carry those coordinates.

Apply the supplied zero-strip route of Definition 548.82. It identifies the root-facing read of the supplied zero and strip rows with the finite route Z, S, H, C, P, N.

Use the supplied zero-strip transport lock of Definition 548.71. Since the lock rules out host equality, zero search, quotient identification, analytic continuation, and ambient completeness, the root zero-strip route remains exactly the displayed hsame and Cont route. □

Theorem 548.68 (Critical-line witness root modulus ledger)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, each natural-number depth request for the modulus row reads only the finite ledger*

$$M, R, Q, H, C, P, N.$$

The request may use the displayed modulus row, real-part row, rational comparison ledger, transport row, continuation row, package row, and local naming row, but it receives no hidden numeric oracle, host equality, quotient representative, analytic-continuation certificate, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus ledger`

Proof. Begin with the modulus ledger entry of Definition 548.83. It names M, R, Q, H, C, P, N as the finite route for the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ modulus-facing ledger.

Next apply Definition 548.17 and Definition 548.30. These results force each natural-number depth request through the displayed continuation route before any public modulus read is available.

Finally use Definition 548.77. The rational comparison part stays inside Q together with M, R, H, C, P, N, so the depth request cannot escape to a hidden numeric oracle, host equality, quotient data, analytic continuation, or ambient completeness. □

Theorem 548.69 (Critical-line witness root non-escape package)**.** *The root-facing* $\mathsf{CriticalLineWitness}^{\uparrow}$ *package exported to downstream consumers consists only of the supplied zero-strip route, the modulus ledger route, and the local* BHist, hsame, Cont, Pkg, *and* NameCert *rows. It exports no* RH

proof, no zero search, no global zero classifier, no host equality, no quotient representative, and no ambient analytic completeness.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root nonescape package`

Proof. First combine Definition 548.67 with Definition 548.68. The first theorem supplies the finite zero-strip route, and the second theorem supplies the finite modulus and rational-comparison route.

Next apply Definition 548.84. It packages the downstream RH-facing boundary as the supplied zero-strip route, modulus ledger entry, and local BEDC rows.

Finally use Definition 548.78. That route refuses RH proof data, zero search, global zero classification, host equality, quotient representatives, analytic continuation, and ambient completeness, so the exported package contains exactly the finite boundary described above. □

Theorem 548.70 (Critical-line witness modulus comparison route totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every modulus comparison route of W factors through the displayed modulus row M, real-part row R, rational comparison ledger Q, transport row H, continuation row C, package row P, and local* NameCert *row N. No comparison route reads a global zero classifier, host equality, quotient representative, analytic-continuation certificate, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus comparison route totality`

Proof. Start with the modulus request factorization of Definition 548.30. It places each comparison request on the displayed Cont route before the public read of M and keeps the request inside R, Q, H, C, P, N.

Then use the strip-modulus ledger triad of Definition 548.41. The strip row, modulus request row, and rational comparison ledger are one finite BEDC-visible triad under the same transport, continuation, package, and naming rows.

Finally apply the obligation closure package of Definition 548.22. The available obligation surface is exactly the carrier, naming, transport, continuation, package, non-escape, and ledger rows, so every modulus comparison route is total through M, R, Q, H, C, P, N. □

Theorem 548.71 (Critical-line witness supplied zero-strip transport lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The supplied zero row Z and critical-strip row S remain locked to the displayed* BHist *surface by the componentwise* hsame *transport row H, together with C, P, N. They cannot be replaced by host equality, zero search, a global zero-set classifier, quotient identification, or analytic continuation.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied zero strip transport lock`

Proof. Project the supplied input carrier row of Definition 548.23. The rows Z and S are input coordinates of the accepted packet, while modulus and comparison rows are read only after those supplied rows are exposed.

Next use supplied zero readback from Definition 548.29 and strip-zero compatibility from Definition 548.9. These rows identify the zero-facing and strip-facing routes with the displayed BHist packet and its hsame transport.

The downstream refusal package of Definition 548.34 supplies the non-escape clause. It excludes host equality, zero search, global zero classification, quotient representatives, analytic continuation,

and ambient completeness, so the zero-strip pair remains locked by the displayed transport route. □

Theorem 548.72 (Critical-line witness RH-refusal ledger completeness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The refusal ledger of W is complete for excluding an* RH *proof, a global zero classifier, an analytic-continuation certificate, a quotient representative, and ambient completeness from every downstream read. Downstream consumers can cite only $Z, S, M, R, Q, H, C, P, N$ and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal ledger completeness`

Proof. Begin with the RH consumer boundary package of Definition 548.42. It packages downstream access as the zero-source route, strip-modulus ledger triad, nonproof boundary, refusal row, and local obligation surface.

Use downstream source obligation from Definition 548.48. It factors every RH-facing downstream source read through the finite supplied-zero and modulus packet $Z, S, M, R, Q, H, C, P, N$.

Finally apply downstream refusal package from Definition 548.34 and ledger non-escape obligation from Definition 548.52. These rows exclude RH proof data, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness from the ledger, so the refusal surface is complete for the listed downstream exclusions. □

Theorem 548.73 (Critical-line witness zero-strip carrier totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The root carrier of W is total exactly over the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison ledger Q, and structural rows H, C, P, N. It has no coordinate for an* RH *proof, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness principle.*

Proof. Apply Definition 548.70. It places every modulus and rational comparison read inside M, R, Q, H, C, P, N and excludes analytic, quotient, host-equality, and completeness coordinates from the comparison route.

Next use Definition 548.71. The supplied zero row Z and critical-strip row S remain locked to the same BHist, hsame, Cont, Pkg, and NameCert route.

The root-source triad of Definition 548.76 and the root comparison ledger of Definition 548.77 assemble the carrier rows visible to downstream consumers. Finally Definition 548.72 removes every requested RH proof, global zero classifier, analytic-continuation certificate, quotient representative, host equality, and ambient completeness coordinate not displayed in W. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip carrier totality`

Theorem 548.74 (Critical-line witness zero-strip classifier transport)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Any replacement of the supplied zero-strip classifier pair must transport through the same componentwise* hsame *row H, displayed* Cont *row C,* Pkg *row P, and local* NameCert *row N. Such a replacement cannot substitute a host zero-search, global zero-set classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness coordinate for Z, S.*

Proof. Start with Definition 548.71. It binds the rows Z, S to the displayed componentwise transport route and prevents their replacement by host equality, zero search, quotient identification, or analytic continuation.

Use Definition 548.76. The zero-strip pair is part of the finite source triad seen by downstream readers, and the triad is carried only through H, C, P, N.

Now apply the root comparison ledger of Definition 548.77 and the RH-refusal ledger completeness of Definition 548.72. The comparison and refusal rows leave no public slot from which a replacement classifier could import a host zero search, analytic certificate, quotient representative, host equality, or ambient completeness witness. □

Theorem 548.75 (Critical-line witness RH refusal non-escape)**.** *Every* RH*-facing downstream read of an accepted* CriticalLineWitness$^{\uparrow}$ *packet factors through the finite refusal route over* $Z, S, M, R, Q, H, C, P, N$*. The read exports only the displayed refusal ledger: it does not export an* RH *proof, a zero-set classifier, a host zero-search route, an analytic-continuation certificate, a quotient representative, host equality, or ambient completeness.*

Proof. First apply Definition 548.72. It already exhausts downstream RH-facing access by the displayed rows $Z, S, M, R, Q, H, C, P, N$ and by the visible BHist, hsame, Cont, Pkg, and NameCert structure.

Then combine Definition 548.70 with Definition 548.71. The comparison side must read M, R, Q, H, C, P, N, while the zero-strip side must read Z, S, H, C, P, N.

Finally use Definition 548.76 and Definition 548.77. Together they identify the root-facing source and comparison rows as one finite refusal package, with no exported proof of RH and no exported zero-set classifier. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal nonescape`
Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip classifier transport`

Theorem 548.76 (Critical-line witness root source triad)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The supplied zero row* Z*, the critical-strip row* S*, and the rational comparison ledger* Q *are exposed as one finite* BHist *source triad before any downstream boundary may read the packet. The triad is carried only through the displayed transport row* H*, continuation row* C*, package row* P*, and local* NameCert *row* N*.*

Proof. Read the carrier of Definition 548.1. Its accepted packet has Z, S, and Q as displayed rows, so the root-facing source data are already finite BHist coordinates rather than host zero data or analytic structure.

Apply Definition 548.2. The local naming row requires the carrier, classifier, transport, package, and ledger obligations to be discharged on the same visible packet.

Finally combine Definition 548.70 with Definition 548.71. The comparison ledger is total through the displayed modulus and rational comparison rows, while the zero and strip rows stay locked to H, C, P, N. Hence Z, S, Q form the finite source triad available to downstream boundary chapters. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source triad`

Theorem 548.77 (Critical-line witness root comparison ledger)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every root-facing rational comparison used by a downstream consumer factors through the finite ledger* Q *together with* M, R, H, C, P, N*. The comparison route does not read a global zero set, a quotient representative, analytic continuation, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root comparison ledger`

Proof. Start with the modulus comparison route totality of Definition 548.70. It places every comparison request on the displayed modulus, real-part, rational comparison, transport, continuation, package, and naming rows.

Use Definition 548.2 to bind that comparison route to the local carrier and ledger obligations. The obligation surface is finite and names only the accepted packet coordinates.

The supplied zero-strip transport lock of Definition 548.71 prevents the comparison route from being replaced by a host zero search, quotient representative, analytic continuation certificate, or ambient completeness row. Thus the root-facing comparison ledger is exactly the displayed finite route through Q. □

Theorem 548.78 (Critical-line witness root refusal route)**.** *The root-facing route of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet refuses every downstream request for an* RH *proof, zero search, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness witness. A downstream chapter may cite only the finite source triad* Z, S, Q *and the displayed rows* M, R, H, C, P, N*.*

Proof. Begin with the root source triad of Definition 548.76. It exposes the supplied zero row, critical-strip row, and rational comparison ledger as finite BHist rows under the same transport and naming surface.

Next use the root comparison ledger of Definition 548.77. It forces every rational comparison read through the displayed Q route and rules out analytic and quotient replacements.

Finally apply Definition 548.72. The refusal ledger already excludes RH proof data, zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness, so the root-facing route is a finite refusal package rather than an analytic proof surface. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal route`

Theorem 548.79 (Critical-line witness boundary obligation surface)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The root boundary exposed by* W *is exactly the finite obligation surface consisting of the supplied source row* S*, supplied zero row* Z*, zero-strip and modulus row* M*, rational comparison row* Q*, real-part row* R*, componentwise* BHist *and* hsame *transport row* H*, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. Every boundary read of* W *factors through the root obligation rows of Definition 548.70, Definition 548.71, and Definition 548.72, and no read exports an* RH *proof, a zero-location theorem, analytic continuation, quotient equality, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier boundary obligation surface`

Proof. Project the accepted packet to its displayed rows. The source row S, zero row Z, modulus row M, comparison row Q, real-part row R, and structural rows H, C, P, N are coordinates of the same finite carrier, so the boundary read begins from the BHist packet rather than from an analytic host object.

Apply Definition 548.70 for the M, R, Q, H, C, P, N side and Definition 548.71 for the Z, S, H, C, P, N side. These two rows bind source and modulus access before any downstream read reaches the packet.

Finally apply Definition 548.72. It removes every requested RH proof, zero-location theorem, analytic-continuation certificate, quotient equality, host equality, and ambient-completeness coordinate not displayed in W. □

Theorem 548.80 (Critical-line witness source-modulus readback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. A readback that asks for both the supplied source row and the zero-strip/modulus row must expose S and M through the same finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The route uses BHist *carrier data,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it cannot replace the source row by zero search, replace the modulus row by a host analytic estimate, or infer* RH *from the coexistence of S and M.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus readback`
↪ variant `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus readback surface`

Proof. Read the source side first. The supplied zero-strip transport lock of Definition 548.71 keeps Z and S attached to the componentwise transport row H and to the displayed C, P, N rows.

Read the modulus side next. The modulus comparison route totality of Definition 548.70 forces every modulus comparison through M, R, Q, H, C, P, N and leaves no coordinate for a host analytic estimate.

The two reads share H, C, P, N, so the readback is one finite packet read over $Z, S, M, R, Q, H, C, P, N$. The refusal ledger of Definition 548.72 keeps the joint read from becoming an RH proof or a zero-location theorem. □

Theorem 548.81 (Critical-line witness RH-refusal ledger route)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every* RH*-facing request made to W is routed to the refusal ledger over the same rows $Z, S, M, R, Q, H, C, P, N$. The ledger may report the supplied zero row, the critical-strip source row, the zero-strip/modulus row, the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure, and the rational comparison boundary; it reports no* RH *theorem, no zero-location theorem, no global zero classifier, no quotient representative, no host equality, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal ledger route`

Proof. Begin with the complete refusal ledger of Definition 548.72. It already states that downstream RH-facing reads can cite only the displayed rows of the packet and the visible structural data.

Use Definition 548.76 to identify the source side of that ledger, and use Definition 548.77 to identify the rational comparison side. Together they account for Z, S, Q and for the M, R, H, C, P, N support rows needed by readback.

Since the carrier has no coordinate for an RH theorem, zero-location theorem, global classifier, quotient representative, host equality, or ambient completeness principle, the ledger route is exhausted by the displayed packet. □

Theorem 548.82 (Critical-line witness supplied zero-strip route)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every root-facing read of the supplied* ZetaZeros$^{\uparrow}$ *row and the* CritStrip$^{\uparrow}$ *row factors through the displayed finite route*

$$Z, S, H, C, P, N.$$

The route is supplied by the packet carrier and local NameCert *obligations, and it cannot be replaced by zero search, host complex equality, a global zero-set classifier, quotient representatives, analytic continuation, or ambient completeness.*

Proof. Read the carrier in Definition 548.1. The rows Z and S are displayed coordinates of the accepted finite packet, and H, C, P, N are the transport, continuation, provenance, and local naming rows that carry them.

Apply Definition 548.2 and Definition 548.23. These results make the supplied zero and strip rows the entry point for zero-facing reads before any modulus or comparison route is consumed.

Use Definition 548.9, Definition 548.28, and Definition 548.29. They keep the zero-strip read on the same displayed BHist surface and through the same hsame and Cont rows.

Finally apply Definition 548.13. Since the excluded analytic and host-level coordinates are absent from the packet, the supplied zero-strip route is exactly the finite route through Z, S, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied zero strip route`

Theorem 548.83 (Critical-line witness modulus ledger entry)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *modulus-facing ledger entry is the finite route through*

$$M, R, Q, H, C, P, N.$$

Every depth or rational-comparison read used by a downstream root consumer enters through the displayed modulus row, real-part row, rational comparison ledger, transport row, continuation row, package row, and local NameCert *row, and no hidden numeric oracle or quotient representative is supplied.*

Proof. Begin with the modulus row and NameCert obligations from Definition 548.12 and Definition 548.2. They identify M as the finite row answering depth requests and attach it to R, Q, H, C, P, N.

Next use Definition 548.17 and Definition 548.30. These theorems force every depth request through the displayed continuation route before a public read of $M(n)$ is available.

Apply Definition 548.11, Definition 548.14, and Definition 548.70. They identify the rational comparison surface with Q and keep the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ comparison data inside the same finite ledger entry.

The non-escape clauses of Definition 548.13 and Definition 548.27 then remove host equality, hidden numeric oracles, quotient representatives, analytic continuation, and ambient completeness from the ledger entry. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus ledger entry`

Theorem 548.84 (Critical-line witness RH non-escape package)**.** *The root-facing* $\mathsf{CriticalLineWitness}^{\uparrow}$ *package supplies a finite witness boundary for downstream RH-facing consumers but does not supply an* RH *proof, zero search, host equality, quotient representatives, analytic continuation, or analytic completeness. A downstream read may cite only the supplied zero-strip route* Z, S, H, C, P, N*, the modulus ledger entry* M, R, Q, H, C, P, N*, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRhNonescapePackage`

Proof. First combine Definition 548.82 with Definition 548.83. The first theorem supplies the finite zero-strip route, and the second theorem supplies the finite modulus and rational-comparison ledger.

Next apply Definition 548.42 and Definition 548.48. These results bind every downstream RH-facing read to the displayed supplied-input packet and its finite modulus ledger.

Use Definition 548.18, Definition 548.31, and Definition 548.34. They refuse global zero classification, zero search, analytic-continuation export, and any proof of RH from the accepted packet.

Finally apply Definition 548.72. The refusal ledger excludes host equality, quotient representatives, analytic completeness, and the other analytic rows, so the root-facing package is precisely the finite non-escape boundary described above. □

Theorem 548.85 (Critical-line witness root modulus-window totality)**.** *Every root modulus-window read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *factors through*

$$M, R, Q, H, C, P, N.$$

The modulus row, real-part row, rational comparison ledger, componentwise transport, displayed continuation, package provenance, and local naming row are therefore the whole finite route available to a root-facing consumer. No hidden numeric oracle, host real equality, quotient representative, analytic continuation row, or ambient completeness input can answer such a read.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus window totality`

Proof. Start with the modulus ledger entry, Definition 548.83. It identifies M, R, Q, H, C, P, N as the finite ledger used by Real and rational-comparison reads.

Next apply Definition 548.12, Definition 548.17, and Definition 548.30. These results force every depth request through the displayed modulus row, real-part row, rational comparison row, transport row, and continuation route before $M(n)$ is public.

The rational side is supplied by Definition 548.11 and Definition 548.70. They make the comparison row Q the only finite rational surface attached to the modulus window.

Finally use Definition 548.27 and Definition 548.13. The excluded host and analytic coordinates are absent, so M, R, Q, H, C, P, N exhaust every root modulus-window read. □

Theorem 548.86 (Critical-line witness root strip-comparison classifier)**.** *The supplied zero-strip row* Z, S, H, C, P, N *and the rational comparison row* M, R, Q, H, C, P, N *of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet share the same finite classifier route through the displayed carrier*

$$Z, S, M, R, Q, H, C, P, N.$$

A root-facing classifier may compare the zero-strip evidence and the half-line rational comparison only by these rows and their componentwise hsame *and* Cont *routes.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip comparison classifier`

Proof. Begin with the supplied zero-strip route, Definition 548.82. It locks every zero-facing and strip-facing read to Z, S, H, C, P, N.

Then apply the modulus ledger entry and root modulus-window totality, Definition 548.83 and Definition 548.85. Together they put every rational comparison read on M, R, Q, H, C, P, N.

Use Definition 548.23 and Definition 548.2. These results bind the supplied rows and the modulus-comparison rows to one accepted carrier and one local naming surface.

The classifier route is therefore componentwise: H transports the displayed rows, C replays their reads, P records their provenance, and N names the local packet. The non-escape boundary

excludes host equality, zero search, global zero classification, quotient representatives, analytic continuation, and ambient completeness from the comparison. □

Theorem 548.87 (Critical-line witness root refusal downstream gate)**.** *Every RH-facing downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the finite refusal package over*

$$Z, S, M, R, Q, H, C, P, N.$$

The gate may expose the supplied zero-strip route, the modulus-window route, the rational comparison ledger, transport, continuation, provenance, and local name, but it does not expose an RH *proof, zero search, global zero-set classifier, host equality, quotient representative, analytic continuation proof, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootRefusalDownstreamGate`

Proof. First apply Definition 548.84. It states the root-facing package as a finite witness boundary for downstream RH-facing consumers and refuses analytic proof content.

Next combine the strip-comparison classifier with the downstream source obligation, Definition 548.86 and Definition 548.48. The consumer source read is then forced through the displayed supplied rows, modulus rows, rational comparison ledger, and structural rows.

Apply Definition 548.42, Definition 548.18, and Definition 548.31. These boundaries separate the finite witness route from global zero classification, zero search, and any proof of RH.

Finally use Definition 548.34 and Definition 548.72. Their refusal rows remove quotient representatives, host equality, analytic continuation, and ambient completeness, so the downstream gate exports exactly the finite refusal package. □

Theorem 548.88 (Critical-line witness zero-strip depth lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every zero-strip read that reaches the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z *and* $\mathsf{CritStrip}^{\uparrow}$ *row* S *locks the subsequent modulus-depth request to the displayed row* M *under the same* BHist*, componentwise* hsame *row* H*, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N*. The read has no route to a hidden depth oracle, host equality, quotient representative, zero search, analytic continuation, or ambient completeness row.*

Proof. Begin with the supplied zero-strip route of Definition 548.82. It exposes Z, S, H, C, P, N as the finite entry route for root-facing zero-strip reads.

Next apply the root modulus ledger of Definition 548.68. The depth request is recorded through M, R, Q, H, C, P, N, so the zero-strip entry and the depth row share the same transport, continuation, provenance, and local naming surface.

Finally use the root non-escape package Definition 548.69. Since the package exports no hidden depth oracle, host equality, quotient representative, zero search, analytic continuation, or ambient completeness, the zero-strip read locks the modulus depth row to the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip depth lock`

Theorem 548.89 (Critical-line witness comparison-window totality)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every comparison-window read is exhausted by the finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The route exposes the supplied zero row, critical-strip row, modulus-depth row, real-part row, rational comparison ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, and it exposes no additional comparison source.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier comparison window totality`

Proof. Start with Definition 548.76. It displays the supplied zero row, critical-strip row, and rational comparison ledger as one finite BHist source triad carried by H, C, P, N.

Then use Definition 548.70 and Definition 548.77. These rows force comparison requests through M, R, Q, H, C, P, N and rule out global zero classifiers, host equality, quotient representatives, analytic-continuation certificates, and ambient completeness.

The zero-strip depth lock supplies the remaining boundary between Z, S and M. Therefore every comparison-window read is exactly the displayed finite route and has no second window or external comparison source. □

Theorem 548.90 (Critical-line witness RH-boundary downstream lock)**.** *Every RH-facing downstream consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the refusal boundary over*

$$Z, S, M, R, Q, H, C, P, N.$$

It may cite the supplied zero-strip rows, modulus-depth row, real-part row, rational comparison ledger, BHist *evidence,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows, but it receives no proof of* RH*, no global zero classification, no zero search, no host equality, no quotient representative, no analytic-continuation certificate, and no ambient completeness row.*

Proof. First apply the comparison-window totality theorem. It exhausts the finite route available to a downstream comparison read and leaves no external comparison window.

Next use Definition 548.84 and Definition 548.78. These theorems identify the root-facing export as a finite witness boundary rather than a proof surface for RH.

Finally apply Definition 548.72. The refusal ledger removes global zero classification, zero search, host equality, quotient representatives, analytic continuation, and ambient completeness from downstream access, so RH-facing consumers receive only the displayed refusal boundary. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary downstream lock`

Theorem 548.91 (Critical-line witness depth-comparison scope)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every comparison-window route that uses the modulus-depth row* M *and rational comparison ledger* Q *remains inside*

$$Z, S, M, R, Q, H, C, P, N.$$

Changing the depth request, the zero-strip row, or the comparison ledger leaves the local CriticalLineWitness$^{\uparrow}$ *packet rather than projecting a new RH-facing source from the same certificate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier depth comparison scope`

Proof. Begin with the zero-strip depth lock. By Definition 548.88, a zero-strip read that reaches Z and S locks the later modulus-depth request to the displayed row M under the same H, C, P, N surface.

Next apply Definition 548.89. It exhausts comparison-window reads by the finite route $Z, S, M, R, Q, H, C, P, N$, so the rational comparison ledger Q is not interchangeable with an external comparison source.

Finally use Definition 548.90. RH-facing consumers receive only the refusal boundary over these rows. Replacing the depth request, zero-strip row, or comparison ledger therefore leaves the local packet and cannot create a new RH-facing source. □

Theorem 548.92 (Critical-line witness root modulus ledger scope)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The rows M, R, Q, H, C, P, N are the complete root-facing modulus-comparison ledger: every modulus or rational-comparison read factors through the displayed modulus row, real-part row, rational comparison ledger, componentwise transport, displayed continuation, package provenance, and local naming row.*

No root-facing comparison route may replace this ledger by a global zero classifier, host equality, quotient representative, analytic-continuation certificate, or ambient completeness row.

Proof. Begin with the root source triad of Definition 548.76. It exposes the supplied zero row, critical-strip row, and rational comparison ledger as finite BHist data before downstream access.

Apply the root comparison ledger of Definition 548.77. It sends every comparison request through M, R, Q, H, C, P, N and rules out global zero-set, quotient, analytic, and ambient-completeness replacements.

Finally use Definition 548.70. The modulus row and rational comparison row are read under the same transport, continuation, package, and local naming rows, so M, R, Q, H, C, P, N exhaust the root-facing ledger. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootModulusLedgerScope`

Theorem 548.93 (Critical-line witness root zero-strip exclusion)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the supplied zero row Z and critical-strip row S cannot be replaced by host zero search, quotient representatives, analytic continuation, ambient completeness, or any detached zero-strip oracle. Downstream root-facing consumers must read Z and S only through the finite source triad and the displayed transport, continuation, package, and naming rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootZeroStripExclusion`

Proof. Read the root source triad. The zero and strip rows are supplied finite source rows, and the rational comparison ledger is exposed with them before any downstream boundary can inspect the packet.

Use the supplied zero-strip transport lock of Definition 548.71. It keeps Z and S attached to the displayed BHist route through H, C, P, N and excludes host equality, zero search, global zero classification, quotient representatives, and analytic continuation.

Now apply the root refusal route of Definition 548.78. The refusal route also removes ambient completeness and RH-facing proof data, so no detached zero-strip oracle remains available to root-facing consumers. □

Theorem 548.94 (Critical-line witness root RH-boundary refusal)**.** *Every RH-facing downstream consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the finite refusal package over*

$$Z, S, M, R, Q, H, C, P, N.$$

It receives no proof of RH*, no global zero classifier, no host zero search, no quotient representative, no host equality, no analytic-continuation certificate, and no ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh boundary refusal`

Proof. First apply the root modulus ledger scope. The modulus-comparison side is exhausted by M, R, Q, H, C, P, N and cannot read analytic or quotient replacements.

Next apply the root zero-strip exclusion. It fixes the supplied zero and strip rows as finite source data and prevents replacement by host zero search, analytic continuation, quotient representatives, or ambient completeness.

Finally use the RH-refusal ledger completeness theorem Definition 548.72. It identifies the downstream export as a refusal ledger rather than an RH proof surface, so the downstream consumer receives exactly the displayed finite package and none of the excluded proof data. □

Theorem 548.95 (Critical-line witness root public export)**.** *Every public root-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives exactly the finite package*

$$Z, S, M, R, Q, H, C, P, N.$$

This package is obtained by composing root modulus ledger scope, root zero-strip exclusion, and root RH-boundary refusal. The export contains no proof of RH*, no global zero classifier, no analytic-continuation certificate, no quotient representative, no host equality, and no ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root public export`

Proof. First apply Definition 548.92. It fixes the modulus and rational-comparison side of every root-facing read through M, R, Q, H, C, P, N.

Next apply Definition 548.93. It keeps the supplied zero row Z and critical-strip row S attached to the same finite transport, continuation, package, and naming route.

Finally apply Definition 548.94. It identifies the RH-facing boundary as a refusal package over $Z, S, M, R, Q, H, C, P, N$ rather than as proof data. The public root-facing consumer therefore receives exactly the displayed finite package and none of the refused RH, global-zero, analytic-continuation, quotient, host-equality, or ambient-completeness data. □

Theorem 548.96 (Critical-line witness scoped source boundary)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The scoped source boundary of* W *is the finite surface*

$$Z, S, M, R, Q, H, C, P, N.$$

Here Z *is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row,* S *is the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row,* M *is the* $\mathsf{Nat}^{\uparrow}$ *depth-request row,* R *is the* $\mathsf{Real}^{\uparrow}$ *real-part row,* Q *is the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, and the accepted packet itself is the finite* BHist *carrier surface with* H, C, P, N *as the* hsame *row,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The boundary supplies no proof of* RH*, zero search, host equality, quotient representative, analytic continuation, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier scoped source boundary`

Proof. Start with the root zero-strip route and the zero-strip depth lock. They expose the supplied Z and S rows before any modulus read and keep the subsequent $\mathsf{Nat}^{\uparrow}$ depth request on the displayed route through H, C, P, N.

Next attach the modulus and comparison rows. The root modulus ledger, comparison-window totality, and root comparison ledger force every request through M, R, Q under the same finite transport, continuation, provenance, and local naming surface.

Finally apply the RH-boundary downstream lock and the root non-escape package. These rows identify the export as a finite witness boundary and remove RH proof data, zero search, host equality, quotient representatives, analytic continuation, and ambient completeness from the source surface. □

Theorem 548.97 (Critical-line witness source ledger exhaustion)**.** *Every downstream source-ledger read of an accepted* CriticalLineWitness$^{\uparrow}$ *packet factors through the single finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The read may consume the supplied zero row, supplied critical-strip row, natural depth row, real-part row, rational comparison ledger, displayed history row, componentwise transport, continuation replay, package provenance, and local naming row, and it has no second source ledger outside that route.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source ledger exhaustion`

Proof. First use the scoped source boundary. It names the finite route and records which rows can appear in a downstream source-ledger read.

Then compose the root carrier projection with the root refusal consumer exactness theorem. The projection shows that downstream root evidence is exactly the displayed packet, and exactness prevents the consumer from treating any host object as an additional source.

The refusal ledger completeness theorem closes the route. Since the packet refuses RH proof data, zero search, global zero classification, host equality, quotient representatives, analytic continuation, and ambient completeness, the displayed route exhausts the source ledger. □

Theorem 548.98 (Critical-line witness zero-strip modulus source totality)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every source read that begins with the supplied zero-strip rows* Z, S *and later asks for a modulus or comparison row is total through*

$$Z, S, M, R, Q, H, C, P, N.$$

The total route binds the ZetaZeros$^{\uparrow}$ *row,* CritStrip$^{\uparrow}$ *row,* Nat$^{\uparrow}$ *depth requests,* Real$^{\uparrow}$ *real-part row,* Rat$^{\uparrow}$ *comparison ledger,* BHist *evidence,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *row without adding RH proof content, zero search, host equality, quotient representatives, analytic continuation, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus source totality`

Proof. Begin with the supplied zero-strip route. It makes Z, S, H, C, P, N the entry surface for zero-strip reads and prevents replacement by analytic or host-level data.

Next use the zero-strip depth lock and root modulus ledger. The first theorem forces later modulus requests to stay attached to the same zero-strip route, while the second theorem records those requests through M, R, Q, H, C, P, N.

Finally combine source-ledger exhaustion with the RH-boundary downstream lock. The downstream read therefore has one finite route through $Z, S, M, R, Q, H, C, P, N$ and no exit to RH proof data, zero search, host equality, quotient representatives, analytic continuation, or ambient completeness. □

Theorem 548.99 (Critical-line witness root downstream package)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every downstream root-facing read factors through the supplied zero row* Z*, critical-strip row* S*, modulus row* M*, real-part row* R*, rational comparison ledger* Q*, displayed* hsame *transport row* H*, displayed* Cont *route row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N*.*

The package excludes RH *proof data, global zero classifiers, analytic-continuation certificates, quotient representatives, host equality, and ambient completeness.*

Proof. Project the accepted packet through Definition 548.1. The downstream source has exactly the rows $Z, S, M, R, Q, H, C, P, N$ available, with Z and S supplied before the modulus and comparison ledger are read.

Use the root non-escape package of Definition 548.69. It assembles the zero-strip route, modulus ledger route, and local BEDC rows as the root-facing export.

Finally apply the RH-refusal ledger completeness theorem Definition 548.72. It removes RH proof data, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness from every downstream read of the package. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root downstream package`

Theorem 548.100 (Critical-line witness root carrier projection)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the downstream root carrier projection is exactly the finite projection to the rows* $Z, S, M, R, Q, H, C, P, N$*. The projection preserves the supplied zero row, critical-strip row, modulus row, rational comparison ledger, transport row, continuation route, provenance row, and local naming row, and it forgets no row that a downstream root consumer is allowed to inspect.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootCarrierProjectionRows`

Proof. Begin with the carrier-admission row of Definition 548.6. It identifies the accepted object with the displayed finite BHist tuple and leaves no hidden host coordinate behind the packet.

Apply the root source triad theorem Definition 548.76. The supplied zero, strip, and rational comparison rows remain exposed as the source side of the downstream boundary.

The modulus comparison route totality theorem Definition 548.70 supplies the remaining modulus, real-part, transport, continuation, package, and naming coordinates. Combining these two projections gives exactly $Z, S, M, R, Q, H, C, P, N$. □

Theorem 548.101 (Critical-line witness root refusal consumer exactness)**.** *Every downstream consumer of the root-facing* $\mathsf{CriticalLineWitness}^{\uparrow}$ *package is exact for the refusal boundary: it may cite the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, displayed* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row, and it may cite nothing else as root evidence.*

In particular, the consumer cannot extract an RH *proof, a global zero classifier, an analytic-continuation certificate, a quotient representative, host equality, or ambient completeness from the accepted packet.*

Proof. First read the downstream package of Definition 548.99. It supplies the complete row list that a downstream consumer may cite.

Next project through Definition 548.100. This shows that the consumer's root evidence is not a larger analytic object; it is the same finite carrier projection $Z, S, M, R, Q, H, C, P, N$.

Finally apply the root refusal route of Definition 548.78. The refusal route excludes RH proof data, zero search, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness, so the consumer boundary is exact for the displayed packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal consumer exactness`

Theorem 548.102 (Critical-line witness root dependency scope)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The root downstream dependency scope of W is exactly the finite route through the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, supplied* $\mathsf{CritStrip}^{\uparrow}$ *row S,* $\mathsf{Nat}^{\uparrow}$ *depth requests for the modulus row M, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N.*

The scope contains no RH *proof data, global zero classifier, analytic continuation certificate, quotient representative, host equality, detached zero oracle, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root dependency scope`

Proof. Begin with the downstream package of Definition 548.99. It gives every root-facing consumer the finite row list $Z, S, M, R, Q, H, C, P, N$ and excludes the analytic and quotient data outside that list.

Next use the carrier projection of Definition 548.100. The projection identifies those rows as the complete finite carrier surface available to the consumer, with no extra root evidence hidden behind the packet.

Finally combine the root modulus ledger scope of Definition 548.92 with the root zero-strip exclusion of Definition 548.93. The first binds M, R, Q, H, C, P, N to the modulus and rational-comparison route, while the second binds Z, S to the supplied zero-strip source. Together they give exactly the dependency scope stated above. □

Theorem 548.103 (Critical-line witness downstream refusal factorization)**.** *Every downstream refusal read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the dependency scope of Definition 548.102. A refusal consumer may cite the supplied zero row, strip row, modulus row, real-part row, rational comparison ledger, transport, continuation, provenance, and local naming row, and it may cite no additional RH-facing proof surface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream refusal factorization`

Proof. First read the exact consumer boundary of Definition 548.101. It states that a downstream consumer may cite only $Z, S, M, R, Q, H, C, P, N$ as root evidence.

Now apply the dependency scope theorem just established. The allowed rows are not an unstructured list: Z and S are the supplied $\mathsf{ZetaZeros}^{\uparrow}$–$\mathsf{CritStrip}^{\uparrow}$ source, M is read at $\mathsf{Nat}^{\uparrow}$ depths, R is the $\mathsf{Real}^{\uparrow}$ row, Q is the $\mathsf{Rat}^{\uparrow}$ comparison ledger, and H, C, P, N are the BEDC transport, route, provenance, and naming rows.

Finally use the RH-boundary refusal of Definition 548.94. It removes RH proof data, global zero classifiers, zero search, analytic continuation, quotient representatives, host equality, and ambient completeness from the downstream route, so every refusal read factors through the named scope. □

Theorem 548.104 (Critical-line witness zero-strip-modulus package exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the supplied zero row, supplied critical-strip row, modulus row, real-part row, and rational comparison ledger exhaust the root package together with H, C, P, N. There is no remaining coordinate from which a consumer can read another zero source, another strip certificate, another modulus, another rational comparison row, an RH proof, analytic continuation, host equality, quotient representatives, or ambient completeness.*

Proof. Start with Definition 548.102. It identifies the full dependency route as $Z, S, M, R, Q, H, C, P, N$ over $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Use the root modulus ledger scope to exhaust the modulus and rational comparison side. Every modulus or comparison read is already forced through M, R, Q, H, C, P, N.

Use the root zero-strip exclusion to exhaust the zero and strip side. The supplied rows Z and S stay attached to the same displayed history and cannot be replaced by a detached oracle or analytic certificate.

Finally apply the RH-boundary refusal. Since the refusal boundary exports only the finite package and none of the excluded analytic or host-level data, the zero, strip, modulus, and rational comparison rows listed above exhaust the root package. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus package exhaustion`

Theorem 548.105 (Critical-line witness downstream RH boundary refinement)**.** *Let $W = (Z, S, M, R, Q, H, C, P,$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every downstream RH-facing read of W is exhausted by the displayed finite rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The supplied zero-strip route, the modulus ledger entry, the RH non-escape package, the root source triad, and the root comparison ledger together force each read through the visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No downstream consumer receives analytic continuation, quotient representatives, host equality, ambient completeness, a global zero classifier, or an* RH *proof from this boundary.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream rh boundary refinement`

Proof. Begin with the supplied zero-strip route of Definition 548.82. It identifies the zero and strip reads with the finite route Z, S, H, C, P, N.

Next use the modulus ledger entry of Definition 548.83. It places every modulus and rational-comparison request on M, R, Q, H, C, P, N.

The RH non-escape package of Definition 548.84 then removes analytic proof data, zero search, host equality, quotient representatives, analytic continuation, and ambient completeness from the downstream interface. The root source triad and root comparison ledger of Definition 548.76 and Definition 548.77 identify the remaining source and comparison reads with the same displayed packet rows, so the boundary is exactly the finite read surface stated above. □

Theorem 548.106 (Critical-line witness zero-strip modulus route)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The zero-strip route Z, S, H, C, P, N and the modulus route M, R, Q, H, C, P, N form one downstream RH-facing route over the same componentwise* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. A consumer that reads the supplied zero or critical-strip row and then asks for the modulus or rational comparison must stay inside $Z, S, M, R, Q, H, C, P, N$.*

Proof. Apply Definition 548.82 to obtain the supplied route for Z and S. This route is carried by H, C, P, N and excludes replacement by zero search, host equality, quotient representatives, analytic continuation, or completeness.

Apply Definition 548.83 to obtain the modulus-facing route through M, R, Q, H, C, P, N. The shared rows H, C, P, N keep the zero-strip and modulus reads on the same visible packet.

Finally combine Definition 548.76, Definition 548.77, and Definition 548.84. These results make the source triad and comparison ledger the only admissible downstream reads and refuse any analytic or host-level replacement. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus route`

Theorem 548.107 (Critical-line witness root source consumer totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root source consumer for a downstream RH-facing chapter is total on the displayed finite packet: it either reads the supplied zero-strip route, the modulus ledger entry, the root source triad, or the root comparison ledger, and each case factors through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of* W*. There is no fifth consumer route requiring analytic continuation, quotient representatives, host equality, ambient completeness, a global zero classifier, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source consumer totality`

Proof. The supplied zero-strip route of Definition 548.82 covers zero and strip source reads. The modulus ledger entry of Definition 548.83 covers depth, real-part, and rational-comparison reads.

The root source triad of Definition 548.76 packages Z, S, Q as the finite source rows available before a downstream boundary reads the packet. The root comparison ledger of Definition 548.77 packages the comparison side through Q together with M, R, H, C, P, N.

Use the downstream RH boundary refinement of Definition 548.105 and the RH non-escape package of Definition 548.84. They exhaust the public reads by the displayed rows and remove the analytic, quotient, host-equality, completeness, global-zero, and RH-proof alternatives, so the listed source consumers are total for downstream use. □

Theorem 548.108 (Critical-line witness root zero-strip carrier row)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the root-facing zero, strip, modulus, route, transport, package, and NameCert coordinates are one finite* BHist *carrier row:*

$$W = (Z, S, M, R, Q, H, C, P, N).$$

Carrier admission ties Z *to the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row,* S *to the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row,* M *to the on-line modulus row,* R *to the* $\mathsf{Real}^{\uparrow}$ *real-part row,* Q *to the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger,* H *to componentwise* hsame *transport,* C *to displayed* Cont *replay,* P *to* Pkg *provenance, and* N *to the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row.*

Proof. Begin with Definition 548.6. It admits exactly the finite packet $Z, S, M, R, Q, H, C, P, N$ and excludes zero search, global zero classifiers, host equality, quotient representatives, and analytic-completeness coordinates.

Use the root carrier projection of Definition 548.100 and the downstream package of Definition 548.99 to read the same tuple from the root-facing side. They keep the supplied zero row, critical-strip row, modulus row, comparison ledger, transport, route, provenance, and naming coordinates in one visible carrier.

Finally apply Definition 548.22. The closure package repacks the admitted carrier, NameCert obligations, strip-zero compatibility, modulus routing, non-escape boundary, and ledger exactness as the finite obligation surface available to downstream root consumers. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero strip carrier row`

Theorem 548.109 (Critical-line witness root modulus route row)**.** *Every root-facing modulus comparison of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet passes through the displayed route*

$$M, R, Q, H, C, P, N.$$

The comparison must use the Cont *row C and the componentwise* hsame *row H before the modulus row M or rational comparison ledger Q can be exported; it has no route through a hidden numeric oracle, host real equality, quotient representative, analytic-continuation certificate, or ambient completeness principle.*

Proof. Start with the modulus request factorization of Definition 548.30 and the modulus comparison route totality of Definition 548.70. They force each depth or comparison request through the displayed modulus row, real-part row, rational comparison ledger, transport row, continuation row, provenance row, and local naming row.

Next use the root comparison ledger of Definition 548.77 and the root modulus ledger of Definition 548.68. These rows identify Q as the root-facing rational comparison ledger and keep the depth request inside M, R, Q, H, C, P, N.

Apply Definition 548.22 to repack the route as part of the local obligation surface. Since that package admits no hidden oracle, host equality, quotient representative, analytic-continuation certificate, or ambient completeness row, every modulus comparison remains on the displayed Cont and hsame route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus route row`

Theorem 548.110 (Critical-line witness root non-escape ledger row)**.** *The root-facing ledger of an accepted* CriticalLineWitness$^{\uparrow}$ *packet excludes an* RH *proof, analytic-continuation certificate, quotient representative, host equality, and ambient completeness from every downstream read. A downstream consumer may cite only the finite carrier row $Z, S, M, R, Q, H, C, P, N$ together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure of that packet.*

Proof. First read the root non-escape package of Definition 548.69 and the downstream refusal factorization of Definition 548.103. These rows state that root-facing consumers receive the supplied zero-strip route, modulus ledger route, and local BEDC rows, not RH proof data or analytic proof surfaces.

Next apply the root zero-strip carrier row established above. It identifies the downstream evidence as the finite packet $Z, S, M, R, Q, H, C, P, N$ and prevents the consumer from adding a second carrier behind the accepted witness.

Finally use Definition 548.22. The package closes the local obligation surface around carrier admission, naming, transport, continuation, package provenance, non-escape, and ledger rows. Therefore RH proof data, analytic-continuation certificates, quotient representatives, host equality, and ambient completeness have no admitted ledger coordinate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root nonescape ledger row`

Theorem 548.111 (Critical-line witness downstream zero-source readback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every downstream zero-source readback of W must first read the supplied zero row Z, the critical-strip row S, and the rational comparison ledger Q as the root source triad of Definition 548.76, and it may then read only the displayed rows M, R, H, C, P, N needed by the same accepted packet.*

The readback is therefore a finite BEDC row export. It is not an RH *proof, a global zero classifier, analytic continuation, a quotient representative, host equality, or ambient completeness.*

Proof. Start with the root source triad. By Definition 548.76, the downstream source side exposes Z, S, Q as finite BHist rows under the displayed transport, continuation, package, and local naming surface.

Next apply Definition 548.100. The accepted packet projects to exactly $Z, S, M, R, Q, H, C, P, N$, so a downstream readback has no additional carrier coordinate from which to extract analytic or host-level data.

Finally use Definition 548.72. The refusal ledger removes RH proof data, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness from every downstream read, leaving only the finite BEDC rows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream zero source readback`

Theorem 548.112 (Critical-line witness downstream modulus comparison totality)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every downstream modulus or rational-comparison request is total through the displayed route*

$$Z, S \longmapsto M, R, Q \longmapsto H, C, P, N.$$

The source side is the root source triad, the comparison side is the root comparison ledger of Definition 548.77, and the downstream consumer receives no RH *proof, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness row.*

Proof. Read the supplied source first. The root source triad theorem identifies Z, S, Q as the finite source rows available before any downstream boundary inspects the packet.

Now use Definition 548.77. It forces each downstream rational comparison through the ledger Q together with M, R, H, C, P, N and rules out global zero-set, quotient, analytic, and ambient-completeness replacements.

Finally compose with Definition 548.70. The modulus route and the rational comparison route share the displayed transport, continuation, provenance, and local naming rows, so the route is total through the finite packet and cannot acquire host equality or RH-facing proof data. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream modulus comparison totality`

Theorem 548.113 (Critical-line witness downstream refusal export)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The downstream export of* W *is exactly the finite refusal package consisting of the root source triad* Z, S, Q*, the modulus and real-part rows* M, R*, the displayed* hsame *and* Cont *rows* H, C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*.*

Any downstream RH-boundary chapter may cite that package only as a finite BEDC refusal export. It may not cite it as an RH *proof, a global zero classifier, analytic continuation, a quotient representative, host equality, or ambient completeness.*

Proof. First combine Definition 548.111 with Definition 548.112. The first theorem fixes the zero-source readback through the root source triad, and the second fixes the modulus and rational-comparison route through the displayed ledger.

Then apply the RH-refusal ledger completeness theorem, Definition 548.72. Its ledger clause says that downstream consumers can cite only $Z, S, M, R, Q, H, C, P, N$ and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

The export is therefore a refusal surface rather than an analytic proof surface. Since the accepted packet contains no coordinate for RH proof data, global zero classification, analytic continuation, quotient representatives, host equality, or ambient completeness, no downstream consumer can read such data from it. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream refusal export`

Theorem 548.114 (Critical-line witness terminal zero-source totality)**.** *Every terminal critical-line read from an accepted* CriticalLineWitness$^{\uparrow}$ *packet first exposes the supplied zero-source row Z together with the supplied critical-strip row S and the displayed route rows H, C, P, N. The terminal read may continue to the modulus and comparison rows only after this zero-source surface is present; it cannot begin from a zero search, global zero classifier, host complex equality, quotient representative, analytic-continuation certificate, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal zero source totality`

Proof. Project the carrier of Definition 548.1. The supplied zero row Z and supplied strip row S are visible before any modulus coordinate can be read.

Use the root zero-strip route of Definition 548.67 and the zero-row source scope of Definition 548.15. These results force the first terminal read through Z, S, H, C, P, N.

Finally apply the root zero-strip exclusion of Definition 548.93. It removes zero search, host equality, quotient representatives, analytic continuation, and ambient completeness from the terminal source route. □

Theorem 548.115 (Critical-line witness terminal modulus-comparison scope)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every terminal modulus comparison is scoped to the displayed* Rat$^{\uparrow}$, Real$^{\uparrow}$, *and* ZetaZeros$^{\uparrow}$ *rows together with the finite* BHist, hsame, Cont, Pkg, *and local* NameCert *routes. The comparison route is exactly the packet surface $Z, S, M, R, Q, H, C, P, N$, and it exports no external real-order decision, host equality, quotient real representative, hidden numeric oracle, or analytic completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessTerminalModulusComparisonScope`

Proof. First expose the supplied zero-source rows using Definition 548.114. The terminal comparison is attached to the displayed ZetaZeros$^{\uparrow}$ and CritStrip$^{\uparrow}$ source rows before it reaches any modulus data.

Next apply the root modulus ledger scope of Definition 548.92. It exhausts modulus and rational-comparison reads by M, R, Q, H, C, P, N.

Finally combine the rational comparison ledger exactness of Definition 548.11 with the non-escape boundary of Definition 548.13. Thus the terminal comparison stays inside the displayed Rat$^{\uparrow}$, Real$^{\uparrow}$, ZetaZeros$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 548.116 (Critical-line witness terminal RH-refusal exactness)**.** *Every terminal RH-facing read of an accepted* CriticalLineWitness$^{\uparrow}$ *packet exports only the finite critical-line witness boundary*

$$Z, S, M, R, Q, H, C, P, N.$$

It refuses RH *proof content, universal zero classification, zero-producing search, host complex equality, quotient real equality, selected Cauchy representatives, analytic-continuation certificates, and ambient analytic completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal rh refusal exactness`

Proof. Begin with the terminal zero-source totality theorem. It makes the supplied zero and strip rows the first visible source of the terminal read.

Then use the terminal modulus-comparison scope theorem. The modulus and rational comparison side is exhausted by the displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally apply the root RH-boundary refusal of Definition 548.94. The terminal packet is therefore a finite critical-line witness boundary, not an RH proof surface or an analytic oracle. □

Theorem 548.117 (Critical-line witness terminal source exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every terminal read first exhausts the displayed finite source rows* Z, S, H, C, P, N *before any modulus or comparison row is read. The read therefore begins inside the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero row, the supplied critical-strip row, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route; it cannot begin from host equality, a quotient representative, a hidden numeric oracle, analytic continuation, ambient completeness, or a detached zero classifier.*

Proof. Start with terminal zero-source totality, Definition 548.114. It makes Z, S, H, C, P, N the first exposed source surface of a terminal read.

Next project that source through the accepted packet carrier. The rows H, C, P, N keep the read inside the displayed BHist, hsame, Cont, Pkg, and NameCert route, so no host-level equality, quotient representative, hidden numeric oracle, analytic-continuation theorem, or ambient completeness row is available before the source surface is exhausted.

Finally use the terminal modulus-comparison scope of Definition 548.115. It allows modulus and comparison reads only after the finite source route has already been exposed, giving the stated source exhaustion order. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessTerminalSourceExhaustion`
↪ variant `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal source exhaustion`

Theorem 548.118 (Critical-line witness terminal comparison ledger lock)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the terminal comparison ledger is locked to the rows* M, R, Q, H, C, P, N *together with the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$ *source interfaces. The ledger exports only the local modulus, real-part, rational-comparison, transport, continuation, provenance, and naming rows; it excludes host equality, quotient representatives, hidden numeric oracle data, analytic continuation, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal comparison ledger lock`

Proof. First apply terminal source exhaustion. The comparison ledger is reached only after the terminal read has exposed the supplied zero and local route rows.

Then apply terminal modulus-comparison scope, Definition 548.115. It binds every terminal comparison to M, R, Q, H, C, P, N over the displayed $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

The exclusions follow from the same scope statement together with the terminal zero-source totality theorem. Since neither source theorem contains host equality, quotient representatives, hidden numeric oracles, analytic continuation, or ambient completeness, the terminal comparison ledger has no route to those objects. □

Theorem 548.119 (Critical-line witness terminal refusal package)**.** *Every terminal RH-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives the obligation package consisting of the*

source exhaustion of Definition 548.117, the comparison ledger lock of Definition 548.118, and the RH-refusal exactness of Definition 548.116. Thus the consumed package is exactly the finite BEDC row surface $Z, S, M, R, Q, H, C, P, N$ over BHist, hsame, Cont, Pkg, NameCert, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{ZetaZeros}^{\uparrow}$, *with no* RH *proof, global zero classifier, host equality, quotient representative, hidden numeric oracle, analytic-continuation theorem, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal refusal package`

Proof. Begin with source exhaustion. It supplies the terminal consumer with the finite source route Z, S, H, C, P, N and blocks every external source row.

Next use the comparison ledger lock. It adds only M, R, Q through the displayed $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ comparison surface while preserving the same BHist, hsame, Cont, Pkg, and NameCert route.

Finally apply terminal RH-refusal exactness, Definition 548.116. This identifies the exported terminal boundary with $Z, S, M, R, Q, H, C, P, N$ and removes RH proof content, global zero classification, host equality, quotient representatives, hidden numeric oracles, analytic continuation, and ambient completeness. The three rows together form the root-unblock obligation package used by downstream RH-boundary chapters. □

Theorem 548.120 (Critical-line witness terminal window factorization)**.** *Every terminal* $\mathsf{CriticalLineWitness}^{\uparrow}$ *route factors through the displayed zero strip, source window, modulus comparison,* $\mathsf{Rat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The factorization surface is the finite packet*

$$Z, S, M, R, Q, H, C, P, N$$

over the terminal source and comparison ledgers; it has no entry point from a global zero classifier, host equality, quotient representative, hidden numeric oracle, analytic-continuation theorem, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal window factorization`

Proof. Begin with terminal source exhaustion, Definition 548.117. It makes the zero strip and source window rows Z, S the first visible terminal surface and keeps the route inside H, C, P, N.

Next apply the terminal comparison ledger lock, Definition 548.118. The modulus comparison and $\mathsf{Rat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback rows are exactly M, R, Q over the same transport, continuation, provenance, and naming surface.

Finally use terminal RH-refusal exactness, Definition 548.116. It rules out the analytic, quotient, oracle, and completeness rows, so the terminal route factors only through the displayed finite packet. □

Theorem 548.121 (Critical-line witness terminal ledger nonescape)**.** *For every accepted terminal* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, analytic continuation content,* RH *proof content, quotient representatives, hidden numeric oracles, host equality, and ambient completeness are refused unless they already occur as displayed rows of the finite terminal ledger $Z, S, M, R, Q, H, C, P, N$. Since the ledger contains only the zero-strip source, source-window, modulus-comparison, readback, transport, continuation, provenance, and naming rows, none of those external objects can be recovered by a downstream terminal consumer.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal ledger nonescape`

Proof. First read the source side from Definition 548.117. Before any terminal consumer reaches a comparison row, the displayed zero strip and source window have already exhausted the available source data.

Then read the comparison side from Definition 548.118. It locks the modulus and readback ledger to M, R, Q, H, C, P, N and names no external analytic or host-level object.

Finally apply terminal RH-refusal exactness, Definition 548.116. The terminal boundary is a finite witness boundary rather than an RH proof, analytic-continuation certificate, quotient representative, oracle, or completeness surface, so the ledger cannot escape its displayed rows. □

Theorem 548.122 (Critical-line witness terminal consumer exhaustion)**.** *Every downstream RH-boundary consumer of an accepted terminal* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives exactly the terminal window factorization of Definition 548.120 together with the nonescape ledger of Definition 548.121. The exported consumer package is therefore the finite row surface* $Z, S, M, R, Q, H, C, P, N$ *over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$*, with no additional RH, analytic, quotient, oracle, host, or completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier terminal consumer exhaustion`

Proof. First apply terminal source exhaustion and terminal comparison ledger lock, Definition 548.117 and Definition 548.118. Together they give the downstream consumer exactly the source window, modulus comparison, readback, transport, continuation, provenance, and naming rows.

Next package those rows through terminal window factorization, Definition 548.120. This gives the consumer the finite packet surface $Z, S, M, R, Q, H, C, P, N$ as its only terminal route.

Finally attach terminal ledger nonescape and terminal RH-refusal exactness, Definition 548.121 and Definition 548.116. They remove every RH-proof, analytic-continuation, quotient, oracle, host-equality, and ambient-completeness coordinate not already displayed in the finite ledger. □

Definition 548.123 (Critical-line witness fixed strip source)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the fixed critical-strip source packet is the displayed finite route over the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, the depth row* M *from* $\mathsf{Nat}^{\uparrow}$*, the rational comparison ledger* Q *from* $\mathsf{Rat}^{\uparrow}$*, the real-part row* R *from* $\mathsf{Real}^{\uparrow}$*, the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows* H, C, P, N*. It contains exactly the rows* $Z, S, M, R, Q, H, C, P, N$ *and no analytic-continuation theorem, host complex-zero classifier, quotient representative, or ambient completeness coordinate.*

Lean4 (definition) `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessFixedStripSource`

Theorem 548.124 (Critical-line witness fixed-strip modulus refusal)**.** *Every zero-strip modulus request served by the fixed critical-strip source packet of Definition 548.123 may read only the displayed strip row* S*, supplied zero row* Z*, depth-lock row* M*, rational comparison row* Q*, refusal rows* R, H, C, P, N*, and the local source route connecting them. It cannot replace that route by zero search, a hidden depth oracle, host equality, quotient representatives, analytic continuation, or ambient completeness.*

Proof. Start with the fixed source packet. Its definition exposes only $Z, S, M, R, Q, H, C, P, N$, so the modulus request begins from the supplied zero, the supplied strip, and the displayed depth and comparison rows.

Use the zero-strip depth lock and root modulus ledger scope, Definition 548.88 and Definition 548.92. They keep the request attached to $Z, S, M, R, Q, H, C, P, N$ and make the comparison ledger the complete root-facing route.

Finally apply the root zero-strip exclusion. It removes zero search, hidden depth oracles, host equality, quotient representatives, analytic continuation, and ambient completeness from the request surface, so the fixed strip source supplies only the displayed refusal rows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip modulus refusal`

Theorem 548.125 (Critical-line witness RH consumer nonescape)**.** *Every RH-facing consumer of the fixed critical-strip source packet receives only the finite witness boundary $Z, S, M, R, Q, H, C, P, N$. The consumer may cite the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero,* $\mathsf{CritStrip}^{\uparrow}$ *strip,* $\mathsf{Nat}^{\uparrow}$ *depth row,* $\mathsf{Rat}^{\uparrow}$ *comparison ledger,* $\mathsf{Real}^{\uparrow}$ *real-part row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, but it cannot infer a full* RH *proof, analytic continuation theorem, host complex-zero classifier, quotient representative, host equality, or ambient completeness principle.*

Proof. First read the fixed strip source packet. It presents a local witness boundary over the supplied zero and strip rows, not a public theorem about all zeros.

Next apply the fixed-strip modulus refusal theorem. The modulus request and comparison ledger stay inside $Z, S, M, R, Q, H, C, P, N$, with no route to analytic continuation, quotient representatives, host equality, or ambient completeness.

Finally use the root RH-boundary refusal and RH-boundary downstream lock, Definition 548.94 and Definition 548.90. These rows identify the RH-facing export as a finite refusal boundary, so no full RH proof or host complex-zero classifier can be extracted from the fixed strip packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip rh consumer boundary`

Theorem 548.126 (Critical-line witness root source exposure)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root-facing packet exposes its source rows through finite* BHist *data before any downstream consumer reads a zero-strip or modulus boundary. The exposed rows are the* $\mathsf{ZetaZeros}^{\uparrow}$ *continuation source, the zero-strip row, the rational-height and real-part comparison rows, the modulus-depth row, and the structural rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The exposure route is finite, componentwise stable under hsame*, and contains no analytic-continuation oracle outside the displayed continuation rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source exposure`

Proof. Begin with the root source triad of Definition 548.76. It exposes the supplied zero row, strip row, and source-facing comparison data before any downstream boundary is read.

Use the source normalization interface of Definition 548.59. The $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are all opened through the finite packet $Z, S, M, R, Q, H, C, P, N$.

Now apply the modulus ledger interface of Definition 548.60 together with terminal source exhaustion from Definition 548.117. These rows keep the rational-height and modulus reads downstream of the same displayed zero-strip source route.

Finally use the fixed strip boundary of Definition 548.124. It prevents the exposed strip source from becoming a global analytic-continuation source, so the root exposure remains a finite BHist package. □

Theorem 548.127 (Critical-line witness root strip refusal)**.** *The root-facing* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet refuses to export an* RH *predicate, a global zero locator, a zero search procedure, host equality, quotient representatives, analytic-continuation proof content, or ambient completeness. It exports only the finite source, strip, modulus, rational-height, real-part, transport, route, provenance, and naming rows displayed by*

$$Z, S, M, R, Q, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip refusal`

Proof. First apply the RH refusal interface of Definition 548.62. It identifies the accepted packet surface with the finite rows $Z, S, M, R, Q, H, C, P, N$ and excludes proof content, global zero classification, zero search, quotient representatives, analytic continuation, and completeness.

Next use the root refusal route of Definition 548.78. The refusal is not an external side condition; it is read along the same displayed route as the source and modulus rows.

The terminal refusal package of Definition 548.119 and the fixed strip boundary of Definition 548.124 then close the remaining consumer-facing exits. A downstream consumer may inspect the finite strip and modulus rows, but it receives no RH predicate or global zero-locator coordinate. □

Theorem 548.128 (Critical-line witness root consumer package)**.** *The source exposure row of Definition 548.126 and the strip-refusal row of Definition 548.127 assemble into one downstream* $\mathsf{CriticalLineWitness}^{\uparrow}$ *consumer package. Every consumer receives a finite* BHist *route over* $Z, S, M, R, Q, H, C, P, N$*; every refused analytic or global-zero object remains outside that route.*

Proof. Use the root source exposure theorem to provide the displayed source, strip, rational-height, real-part, and modulus rows. This gives the consumer a finite entry route through $Z, S, M, R, Q, H, C, P, N$.

Use the root strip refusal theorem for the negative side of the same package. The consumer route carries no RH predicate, global zero locator, zero search procedure, host equality, quotient representative, analytic-continuation proof content, or ambient completeness row.

Finally compose the root downstream package of Definition 548.34 with the downstream RH-boundary refinement of Definition 548.105. The assembled package is therefore a consumer-facing finite route, not an analytic theorem or global zero classifier. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root consumer package`

Theorem 548.129 (Critical-line witness root modulus readback)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *zero-strip modulus read is routed through the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$ *source rows together with the* BHist *readback ledger. In particular, the root-facing consumer reads the zero source, critical-strip row, rational-height row, real-part comparison row, modulus-depth row, componentwise transport, continuation route, provenance row, and local* $\mathsf{NameCert}$ *row before it can consume the modulus boundary.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus readback`

Proof. Start with the root source exposure theorem. It identifies the displayed source route through $Z, S, M, R, Q, H, C, P, N$ before any downstream zero-strip or modulus boundary is read.

Use the source normalization interface and the modulus ledger interface. These rows place the $\mathsf{ZetaZeros}^{\uparrow}$ source, the $\mathsf{Rat}^{\uparrow}$ height read, the $\mathsf{Real}^{\uparrow}$ comparison read, and the modulus-depth row inside the same finite BHist packet.

Finally apply the root consumer package. The readback ledger is the route through H, C, P, N, so a consumer that omits those rows is not consuming the accepted root-facing packet. □

Theorem 548.130 (Critical-line witness strip-zero boundary)**.** *A* $\mathsf{CriticalLineWitness}^{\uparrow}$ *critical-line witness cannot be consumed as an* RH *conclusion unless the critical-strip boundary rows and the zero-source boundary rows have both been displayed. The accepted packet supplies only finite zero-source, strip, rational-height, real-part, modulus, transport, route, provenance, and naming rows; it does not supply a global* RH *predicate, analytic-continuation proof object, zero locator, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip zero boundary`

Proof. Apply the root strip refusal theorem. It excludes an RH predicate, global zero locator, zero search procedure, analytic-continuation proof content, and ambient completeness from the root-facing packet.

Now use the fixed strip boundary. The strip row is a displayed finite source boundary, not a global analytic statement about all zeros.

The terminal source exhaustion and downstream RH-boundary refinement keep the same boundary at every consumer site. Therefore a critical-line witness can be cited only as the displayed finite source package, not as an RH conclusion. □

Theorem 548.131 (Critical-line witness root unblock package)**.** *The modulus readback of Definition 548.129, the strip-zero boundary of Definition 548.130, the refusal ledger, the displayed* Cont *route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows form the root-unblock dependency surface for downstream* $\mathsf{CriticalLineWitness}^{\uparrow}$ *chapters. Any downstream chapter citing this package receives the finite rows*

$$Z, S, M, R, Q, H, C, P, N$$

and no hidden RH *conclusion, analytic-continuation oracle, quotient representative, host equality, zero-search procedure, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus readback unblock package`

Proof. First combine root source exposure with root modulus readback. These theorems display the $\mathsf{ZetaZeros}^{\uparrow}$, critical-strip, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, modulus, and BHist readback rows on one finite route.

Next combine root strip refusal with the strip-zero boundary. The refusal ledger blocks all global zero and RH readings while preserving the finite source package available to downstream consumers.

Finally repack through the root consumer package. The continuation route, provenance row, and local NameCert row are exactly C, P, N, and the transported finite source is carried by H, so the dependency surface is the displayed package $Z, S, M, R, Q, H, C, P, N$. □

Theorem 548.132 (Critical-line witness source modulus exposure)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet exposes the supplied source strip row, modulus row, zero witness row, transport row, route row, package row, and local* $\mathsf{NameCert}$ *row before any critical-line read is available. The visible surface is the finite packet*

$$Z, S, M, R, Q, H, C, P, N,$$

so a critical-line consumer must read the source strip, modulus ledger, zero witness, continuation route, provenance, and naming rows as displayed BHist *data.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus exposure`

Proof. Begin with the root source-modulus interface of Definition 548.59 and Definition 548.60. These rows expose the supplied zero-strip source and the modulus-comparison ledger before downstream use.

Apply Definition 548.67 and Definition 548.68. The zero-strip side is carried by Z, S, H, C, P, N, while the modulus side is carried by M, R, Q, H, C, P, N.

Finally use the root unblock rows in Definition 548.70 and Definition 548.76. Together they identify the accepted source, modulus, zero witness, transport, route, package, and local naming rows as the finite boundary preceding any critical-line read. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessSourceModulusExposure`

Theorem 548.133 (Critical-line witness zero-strip boundary)**.** *Every zero-strip read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is a finite* BHist*,* Cont*,* hsame*, and* Pkg *row read through*

$$Z, S, H, C, P, N.$$

It supplies no analytic continuation, no RH *truth value, and no global zeta-zero classification.*

Proof. Start with Definition 548.67. It places the supplied zero row Z and critical-strip row S on the displayed hsame and Cont route before any modulus or downstream read is exposed.

Use Definition 548.82 and Definition 548.69. These identify the zero-strip read as finite BHist carrier data transported and replayed by H and C, with provenance and naming recorded by P and N.

The RH boundary rows of Definition 548.84 and Definition 548.78 exclude analytic continuation, RH proof content, and global zero classification from this zero-strip boundary. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip boundary`

Theorem 548.134 (Critical-line witness RH boundary refusal)**.** *Any* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet may cite the packet only as a finite witness boundary over*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer cannot extract a proof or refutation of RH *from the supplied zero-strip rows, modulus ledger, transport, route, package, or local naming rows.*

Proof. First combine Definition 548.62 with Definition 548.84. These theorems identify the RH-facing surface with the finite witness packet and refuse proof content, zero search, global classification, quotient representatives, analytic continuation, and ambient completeness.

Next apply Definition 548.72 and Definition 548.87. They keep every downstream RH-facing read inside the displayed rows $Z, S, M, R, Q, H, C, P, N$.

Thus the consumer receives a finite boundary witness only. Since neither an RH truth coordinate nor a global zero classifier appears in the carrier, the packet cannot be read as a proof or refutation of RH. □

Theorem 548.135 (Critical-line witness strip-depth readback totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every strip-depth read that starts from the supplied zero row* Z*, the fixed critical-strip row* S*, or a bounded depth request in* M *routes through the finite rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The read may expose the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *row, the* $\mathsf{CritStrip}^{\uparrow}$ *boundary, the* $\mathsf{Nat}^{\uparrow}$ *depth budget, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, but no other strip-depth source.*

Proof. Project the accepted carrier through Definition 548.1. The rows $Z, S, M, R, Q, H, C, P, N$ are the displayed coordinates of the same finite BHist packet, so a strip-depth consumer has no carrier coordinate outside that tuple.

Use Definition 548.40, Definition 548.4, and Definition 548.161. They route supplied zero reads, fixed-strip reads, and bounded depth requests through Z, S, M before the real and rational comparison rows R, Q are exposed.

Finally apply Definition 548.159. The transport, continuation, package, and local naming rows H, C, P, N exhaust the remaining public readback surface, so every strip-depth read is confined to the displayed route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip depth readback totality`

Theorem 548.136 (Critical-line witness rational comparison ledger totality)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each rational comparison read is exhausted by the* $\mathsf{Real}^{\uparrow}$ *row* R *and the* $\mathsf{Rat}^{\uparrow}$ *ledger row* Q*, together with the local transport, continuation, package, and naming rows* H, C, P, N*. Such a read cannot import analytic-continuation data, a host zero theorem, a selected quotient representative, or proof data for* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rational comparison ledger totality`

Proof. Begin with the rational comparison surface. By Definition 548.77 and Definition 548.11, the only rows that compare the real-part and rational boundary data are R and Q.

The depth source remains finite by Definition 548.135. Thus the comparison read first consumes Z, S, M as source and depth data, then reads R, Q as the comparison ledger inside the same packet.

The refusal boundary follows from Definition 548.159 and Definition 548.162. Analytic continuation, host zero theorems, quotient representatives, and RH proof data have no coordinate in W, so the rational comparison ledger is total exactly at R, Q, H, C, P, N. □

Theorem 548.137 (Critical-line witness RH refusal ledger exactness)**.** *Every* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the finite refusal boundary carried by*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may read the supplied zero row, fixed strip row, bounded depth row, real-part row, rational comparison row, transport row, continuation row, package row, and local naming row. It receives no proof of RH*, zero generator, quotient representative, host equality row, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal ledger exactness`

Proof. Project the consumer route to the displayed carrier. The RH-facing surface is still a read of the same finite packet $W = (Z, S, M, R, Q, H, C, P, N)$, not a separate analytic object.

Apply Definition 548.72, Definition 548.78, and Definition 548.136. These rows expose only the supplied zero, strip, depth, real, rational, transport, continuation, provenance, and naming data before the refusal boundary is reached.

The excluded objects would require coordinates for an RH theorem, a zero-producing generator, quotient representatives, host equality, or ambient completeness. Since none is present in the accepted packet, the refusal ledger is exact for every downstream RH-facing consumer. □

Theorem 548.138 (Critical-line witness zero-strip source row)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier* $W = (Z, S, M, R, Q, H, C, P, N)$*, the* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z *and the* $\mathsf{CritStrip}^{\uparrow}$ *row* S *are exposed before any critical-line read. The read may proceed only after the supplied zero source and fixed strip source have been named as finite* BHist *coordinates of the same carrier.*

Proof. Project the carrier of Definition 548.1. Its first displayed source rows are Z and S, while M, R, Q, H, C, P, N are the later modulus, comparison, transport, route, provenance, and naming rows.

Use Definition 548.2. Carrier admission and local naming attach the zero row and strip row to the same accepted packet before downstream reads are available.

Apply Definition 548.3. It identifies the supplied zero source, fixed strip source, and source-facing route as the root entry surface, so no critical-line read can bypass Z or S. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip source row`

Theorem 548.139 (Critical-line witness modulus readback row)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier* $W = (Z, S, M, R, Q, H, C, P, N)$*, every modulus read factors through* M, R, Q, H, C, P, N*. The read uses the on-line modulus row, real-part row, rational comparison ledger, componentwise transport, continuation route, provenance, and local naming; it does not use host equality or a quotient representative.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus readback row`

Proof. Read the modulus coordinate through Definition 548.1. The row M answers only as part of the same finite packet that carries R, Q, H, C, P, N.

Apply Definition 548.2 and Definition 548.4. These bind the depth request to the real-part row R, rational comparison ledger Q, componentwise transport H, replay route C, provenance P, and local name N.

The carrier has no field for host equality of complex numbers and no quotient representative selector. Therefore the modulus readback remains the displayed finite row read through M, R, Q, H, C, P, N. □

Theorem 548.140 (Critical-line witness fixed-strip scope exhaustion)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier* $W = (Z, S, M, R, Q, H, C, P, N)$*, the fixed-strip source surface is exhausted by the rows*

$$Z, S, M, R, Q, H, C, P, N.$$

It exposes supplied zero, critical-strip, modulus, real-part, comparison, transport, route, provenance, and local naming data, and it does not assert RH *or produce new zeros.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip scope exhaustion`

Proof. Start with Definition 548.1. The accepted packet lists exactly $Z, S, M, R, Q, H, C, P, N$, with no coordinate for an RH truth value or a zero-producing search.

Use Definition 548.138 for the supplied zero and strip source. Use Definition 548.139 for every modulus-facing read.

Finally apply Definition 548.2. The local NameCert surface exhausts public reads through the displayed carrier and its non-escape clauses, so the fixed-strip surface remains a finite source boundary rather than an RH proof or zero generator. □

Theorem 548.141 (Critical-line witness root zero carrier)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. A root read of W exposes only the finite zero-strip carrier rows over* BHist, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, *and* $\mathsf{CritStrip}^{\uparrow}$*: the supplied zero row Z, the critical-strip row S, the displayed modulus row M, the real-part row R, the rational comparison ledger Q, and the structural rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero carrier`

Proof. Project the carrier of Definition 548.1. The rows $Z, S, M, R, Q, H, C, P, N$ are the whole accepted packet, and the root-facing rows are read as finite BHist coordinates rather than as host analytic data.

Use the supplied zero-strip route of Definition 548.82 together with the root modulus-window totality of Definition 548.85. The first row keeps Z, S inside the $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ boundary supplied by chapter 73; the second keeps M, R, Q inside the $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ comparison boundary supplied by chapter 38.

The root refusal downstream gate of Definition 548.87 then removes every coordinate not displayed in the finite packet. Hence the root read exposes exactly the zero-strip carrier rows listed above. □

Theorem 548.142 (Critical-line witness strip modulus readback)**.** *Every strip-modulus consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet reads the displayed modulus and zero-strip rows through the route*

$$Z, S, M, R, Q, H, C, P, N.$$

In particular, the consumer passes through Cont*, componentwise* hsame, Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows before any RH-facing boundary is available.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip modulus readback`

Proof. Begin with the root zero carrier of Definition 548.141. It identifies the zero row, strip row, modulus row, comparison ledger, and structural rows as the only carrier data a root consumer may read.

Apply Definition 548.67 for the Z, S part of the read, and apply Definition 548.68 for the M, R, Q part. These sibling rows place the zero-strip and modulus requests under the same displayed hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Finally use Definition 548.84. It places any RH-facing read downstream of the same finite package, so the consumer cannot reach an RH boundary before the displayed strip-modulus route has been read. □

Theorem 548.143 (Critical-line witness zero-strip consumer boundary)**.** *Every downstream RH or zero-count consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the displayed zero-strip witness package*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer receives no analytic continuation certificate, no host complex equality, no global zero oracle, no quotient representative, and no proof of RH.

Proof. First apply Definition 548.142. It forces every strip-modulus consumer through the displayed zero, strip, modulus, comparison, transport, continuation, provenance, and naming rows.

Next use the downstream refusal package of Definition 548.34 and the RH refusal ledger completeness of Definition 548.72. These rows exclude analytic continuation, host equality, quotient representatives, global zero classification, and RH proof data from downstream reads.

The carrier dependencies remain the finite $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows described by chapter 73 and chapter 38. Thus an RH or zero-count consumer receives only the displayed witness package and none of the analytic or host-level objects listed above. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip consumer boundary`

Theorem 548.144 (Critical-line witness root source lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing read of the source side is locked to the displayed* BHist *rows Z, S, M, R, Q together with the componentwise* hsame *row H, the visible* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. No analytic-continuation row, host evaluator, global zero classification, or* RH *truth value can replace this finite source route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source lock`

Proof. Start with the root source exposure package of Definition 548.126. It exposes the supplied zero row, the strip row, the modulus row, the real-part row, and the rational comparison row as the finite source surface of W.

Use the source-modulus interface of Definition 548.59 and Definition 548.60. These rows route the source read through Z, S, M, R, Q before any downstream consumer receives it.

The transport, continuation, provenance, and local naming rows H, C, P, N then authenticate the same finite route. Since the route is already exhausted by BHist, hsame, Cont, Pkg, and local NameCert rows, it has no slot for an analytic continuation, a host evaluator, a global zero classifier, or an RH truth-value oracle. □

Theorem 548.145 (Critical-line witness strip reflection boundary)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, any strip-reflection or zero-strip consumer reads only the displayed* BHist *packet rows, the componentwise* hsame *transport H, the* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The boundary refuses analytic continuation,* RH *truth values, global zero classification, and host evaluators.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip reflection boundary`

Proof. Project the zero-strip boundary through Definition 548.133. It places the strip side on the supplied Z, S rows and keeps the read inside the displayed packet.

Apply the zero-strip consumer boundary of Definition 548.143. The consumer must pass through $Z, S, M, R, Q, H, C, P, N$, including the visible hsame, Cont, Pkg, and local NameCert rows.

The root RH-boundary refusal of Definition 548.94 then removes the analytic exits. Hence strip reflection remains a finite BEDC boundary and cannot become analytic continuation, an RH truth value, a global zero classifier, or a host evaluator. □

Theorem 548.146 (Critical-line witness downstream RH unblock package)**.** *The downstream RH-facing unblock surface of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is exactly the finite package*

$$Z, S, M, R, Q, H, C, P, N.$$

It exposes the BHist *source rows, the* hsame *transport row, the* Cont *route row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row needed by downstream consumers, while refusing analytic continuation,* RH *truth values, global zero classification, and host evaluators.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream rh unblock package`

Proof. First use the root source lock of Definition 548.144. It fixes the source and modulus side of the downstream read to the displayed finite packet.

Next use the strip reflection boundary of Definition 548.145. This keeps every zero-strip consumer on the same $Z, S, M, R, Q, H, C, P, N$ surface and records the refusal rows before an RH-facing consumer can read the package.

Finally apply the downstream RH-boundary refinement of Definition 548.105 together with the root refusal downstream gate of Definition 548.87. The downstream route is therefore an unblock package for finite BEDC reads only, with no analytic continuation, RH truth value, global zero classification, or host evaluator exported. □

Theorem 548.147 (Critical-line witness root RH-boundary formal target)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every downstream RH-facing read from the root RH-boundary sibling remains inside*

$$Z, S, M, R, Q, H, C, P, N.$$

The target surface exports only the displayed zero, strip, modulus, real-part, rational-comparison, transport, continuation, provenance, and local naming rows. It cannot export analytic continuation, an RH *truth value, a global zero classifier, or a host evaluator.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier criticallinewitness root rh boundary formal target`

Proof. Project the downstream unblock package of Definition 548.146. It identifies the complete read surface as the finite packet $Z, S, M, R, Q, H, C, P, N$.

Use the root source lock and strip reflection boundary. Together Definition 548.144 and Definition 548.145 keep the source side and strip side on the same displayed rows before an RH-facing consumer can inspect the sibling.

Finally apply the downstream RH-boundary refinement and root refusal gate. Those refusal rows remove analytic continuation, RH truth values, global zero classifiers, and host evaluators from the target surface, so the formal target is exactly the finite BEDC boundary named above. □

Theorem 548.148 (Critical-line witness root refusal budget completion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing critical-line witness request factors through the finite rows*

$$Z, S, M, R, Q, H, C, P, N$$

before any RH*-facing consumer can read it. The route exposes only the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, componentwise transport, continuation replay, package provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal budget completion`

Proof. Begin with the root source triad of Definition 548.76. It exposes Z, S, Q as finite BHist source data before a downstream boundary can read the witness.

Next apply the root comparison ledger of Definition 548.77. It places every root-facing rational comparison through Q together with the displayed modulus, real-part, transport, continuation, package, and naming rows.

The remaining downstream read is governed by the RH-refusal ledger completeness theorem, Definition 548.72. That ledger keeps the consumer inside $Z, S, M, R, Q, H, C, P, N$ and refuses every stronger analytic or RH-facing export.

Thus the root-facing request is complete as a finite refusal budget before any RH-facing consumer receives the packet. □

Theorem 548.149 (Critical-line witness zeta-strip modulus totality)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the displayed route through* $\mathsf{ZetaZeros}^{\uparrow}$*,* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *is total for the zero-strip modulus packet. It reads the zeta-zero surface, critical-strip boundary, real-part modulus row, and rational comparison ledger, and it exports no analytic-continuation row, host equality row, quotient row, or ambient-completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta strip modulus totality`

Proof. Use the zeta dependency lattice of Definition 548.169. It places every zero-strip or critical-line read behind the existing $\mathsf{ZetaZeros}^{\uparrow}$ and RH-boundary surfaces before the local modulus witness is public.

The root source triad Definition 548.76 supplies the finite Z, S, Q side of this route. The root comparison ledger Definition 548.77 then supplies the M, R, Q comparison side through $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ rows rather than through a host analytic object.

Finally, the RH-refusal ledger completeness theorem Definition 548.72 excludes analytic continuation, host equality, quotient representatives, and ambient completeness. Hence the displayed zeta-strip-modulus route is total exactly over the finite packet rows. □

Theorem 548.150 (Critical-line witness RH consumer refusal exactness)**.** *Every* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives exactly the refusal ledger and finite witness boundary over*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer does not receive an RH *proof, a global zero classifier, an analytic-continuation certificate, a quotient representative, host equality, or an ambient-completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer refusal exactness`

Proof. First read the refusal ledger through Definition 548.72. It states that downstream RH-facing access is exhausted by the displayed rows and the visible BHist, hsame, Cont, Pkg, and NameCert structure.

Next use the root source triad and root comparison ledger, Definition 548.76 and Definition 548.77. Together they identify the finite witness boundary that the consumer may cite: the supplied zero and strip rows, the modulus and real-part rows, the rational comparison ledger, and the structural rows.

The zeta dependency lattice of Definition 548.169 fixes the sibling source of the zero-strip read without adding an analytic proof surface. Therefore the consumer receives the refusal ledger exactly, not an RH proof or global zero classifier. □

Theorem 548.151 (Critical-line witness candidate closed route)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before any modulus read, every candidate critical-line route exposed by* W *is the finite source route through the supplied zero row* Z*, the critical-strip row* S*, the componentwise* hsame *row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the*

local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. The route is a* BHist *candidate surface carried by the source triad and fixed-strip source rows, not a host zero source or analytic-continuation surface.*

Proof. Project the accepted packet through the carrier definition and the root source triad of Definition 548.76. These rows expose Z and S as the source side of the same finite BHist packet.

Next use the root unblock source triad and fixed-strip source boundary, Definition 548.3 and Definition 548.123. They keep the candidate route on Z, S, H, C, P, N before the modulus coordinate M is read.

The NameCert obligations then compose the displayed hsame, Cont, and Pkg rows with the local name N. Since the candidate has already been exhausted by Z, S, H, C, P, N, it exports no host zero source and no analytic-continuation route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier candidate closed route`

Theorem 548.152 (Critical-line witness finished endpoint handoff)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every finished critical-line endpoint handoff factors through* M, R, Q, H, C, P, N*: it consumes the* $\mathsf{Nat}^{\uparrow}$ *depth row* M*, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, the componentwise* hsame *transport* H*, the* Cont *replay route* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. No quotient representative selector or host-equality witness is consumed by the handoff.*

Proof. Begin with the root unblock modulus ledger of Definition 548.4. It identifies M, R, Q, H, C, P, N as the complete endpoint-facing route for each modulus request.

Apply the modulus readback row and strip reflection boundary, Definition 548.139 and Definition 548.145. These rows attach the finished endpoint handoff to the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ sources while keeping the transport and replay data inside H, C, P, N.

Finally use the NameCert obligations. The local naming and non-escape rows close the handoff over the displayed BHist, hsame, Cont, and Pkg coordinates, leaving no slot for quotient representative selection or host equality. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finished endpoint handoff`

Theorem 548.153 (Critical-line witness obstruction separation)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the obstruction row is exactly the refusal boundary of the packet. Downstream readers may cite the source triad, endpoint handoff, and local* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows, but they cannot read an* RH *proof, a zero-producing search, an analytic-continuation certificate, host equality, a quotient representative, or ambient completeness from* W*.*

Proof. Start from the root unblock refusal boundary of Definition 548.5. It states the missing analytic, host, quotient, and completeness coordinates for the accepted packet.

Use the candidate route and finished endpoint handoff above. The first route exhausts the pre-modulus candidate surface through Z, S, H, C, P, N, while the second route exhausts the finished endpoint surface through M, R, Q, H, C, P, N.

Apply the downstream RH unblock package and fixed-strip RH consumer nonescape, Definition 548.146 and Definition 548.125. Together they show that every downstream read remains inside the displayed packet rows, so the obstruction row is precisely the refusal boundary and not an implicit analytic or host-level witness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessObstructionSeparation`

Theorem 548.154 (Critical-line witness route readback exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every downstream readback accepted by W factors through the zero-strip source row fixed by Definition 548.144, the strip reflection boundary of Definition 548.145, and the downstream RH unblock package of Definition 548.146. No downstream readback can replace that finite route by analytic continuation, a host evaluator, global zero classification, an* RH *truth value, or an ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier route readback exhaustion certificate`

Proof. Start with the root source lock. It fixes the source-facing side of W to the displayed rows $Z, S, M, R, Q, H, C, P, N$ and refuses analytic or host-level replacements before a downstream consumer can read the packet.

Next apply the strip reflection boundary. Every zero-strip or strip-reflection consumer remains on the same finite packet, so the readback cannot leave the zero-strip source row for a separate analytic-continuation route.

Finally use the downstream RH unblock package. It exposes exactly the same finite rows to downstream consumers and records the missing RH truth-value and host-evaluator rows as refusals. Thus every downstream readback is exhausted by the displayed source package. □

Theorem 548.155 (Critical-line witness downstream zero-strip exactness)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zero-strip consumer receives the same modulus comparison surface M, R, Q, H, C, P, N after reading the zero-strip source rows Z, S. The consumer may transport and replay that surface through H and C, and may cite the provenance and local name through P and N, but it receives no second modulus row, no host zero classifier, and no proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream zero strip exactness`

Proof. Project the zero-strip source through Definition 548.144. The rows Z, S are the source side of the same accepted packet as the modulus, real-part, and rational comparison rows.

Apply Definition 548.145. It keeps each zero-strip consumer inside $Z, S, M, R, Q, H, C, P, N$ and refuses host classifiers and analytic exits.

Use Definition 548.146 to read the downstream surface. The package exposes the modulus comparison rows M, R, Q only through the same transport, route, provenance, and local name. Therefore all zero-strip consumers receive the same comparison surface and no separate RH proof. □

Theorem 548.156 (Critical-line witness modulus refusal ledger soundness)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, any off-route request for an* RH *proof, analytic-continuation certificate, global zero classification, host evaluator, quotient representative, host equality, or ambient completeness theorem lands in the refusal ledger attached to the displayed rows. The packet exports the zero-strip source and modulus comparison surface, not a proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus refusal ledger soundness certificate`

Proof. First use Definition 548.144 to identify the only root-facing source route. The route is finite and displayed, so an off-route analytic request has no source coordinate outside $Z, S, M, R, Q, H, C, P, N$.

Next use Definition 548.145. Strip and zero-strip consumers remain inside the same boundary, where analytic continuation, RH truth values, global zero classification, and host evaluators are refused.

Finally apply Definition 548.146. It presents the downstream surface as an unblock package for finite BEDC reads only. The listed off-route requests therefore terminate in the refusal ledger and cannot be repacked as a proof of RH. □

Theorem 548.157 (Critical-line witness strip localization)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root read that localizes the witness to the critical strip factors through the displayed* $\mathsf{CritStrip}^{\uparrow}$ *row S, the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, the* $\mathsf{Nat}^{\uparrow}$ *depth route of M, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N. It cannot replace the displayed strip row by analytic continuation, host complex equality, a quotient representative, a global zero classifier, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip localization`

Proof. Start with the root source triad of Definition 548.76. It displays the supplied zero row Z, the critical-strip row S, and the rational comparison ledger Q as finite BHist source data under H, C, P, N.

Next apply Definition 548.71. The zero row and strip row are transported componentwise through the same hsame surface, so localization cannot be replaced by host equality or a detached zero search.

Finally use Definition 548.77 and Definition 548.72. The Nat, Rat, and Real dependency rows are consumed only through M, R, Q, while the refusal ledger removes analytic continuation, quotient representatives, global zero classification, and ambient completeness from the localized strip read. □

Theorem 548.158 (Critical-line witness zero-row soundness)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zero-row read is sound only as a read of the supplied finite witness packet. The read consumes the* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z together with the* $\mathsf{CritStrip}^{\uparrow}$ *row S, the* $\mathsf{Nat}^{\uparrow}$ *depth row of M, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it does not construct a zero, enumerate zeros, prove* RH*, or import host complex analysis.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero row soundness`

Proof. Project the carrier of Definition 548.1. The zero-facing coordinate is the supplied row Z, and it is packaged with S, M, R, Q, H, C, P, N rather than with a zero-producing search.

Use Definition 548.40 and Definition 548.29. These results force every zero-facing read through the displayed $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ source rows before any modulus or comparison data are inspected.

Now apply Definition 548.38 and Definition 548.11. The real-part and rational comparison data are read through R, Q inside the same finite packet, so the zero row remains a supplied witness row and not an analytic construction.

The downstream refusal package and Definition 548.72 exhaust the remaining ledger. They exclude RH proof data, zero enumeration, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness. □

Theorem 548.159 (Critical-line witness ledger non-escape)**.** *Every ledger read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the finite route over* $Z, S, M, R, Q, H, C, P, N$ *and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The ledger may expose the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero row, the* $\mathsf{CritStrip}^{\uparrow}$ *strip row, the* $\mathsf{Nat}^{\uparrow}$ *depth requests, the* $\mathsf{Rat}^{\uparrow}$ *comparison rows, and the* $\mathsf{Real}^{\uparrow}$ *real-part row, but it exports no analytic continuation theorem, global* RH *proof, host complex-zero classifier, quotient representative, host equality row, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier ledger nonescape`

Proof. Begin with Definition 548.157. It confines root strip reads to $Z, S, M, R, Q, H, C, P, N$ and to the displayed CritStrip, ZetaZeros, Nat, Rat, and Real dependency rows.

Then use Definition 548.158. The zero row is sound only as a supplied finite packet row, so it cannot create a zero-producing route or a global zero classifier.

Finally apply Definition 548.52 together with Definition 548.78. The local ledger is exhausted by the displayed carrier, transport, continuation, package, and naming rows, and every request for analytic continuation, RH proof data, quotient representatives, host equality, or ambient completeness is refused rather than exported. □

Theorem 548.160 (Critical-line witness zero-localization obligation)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet anchored in* BHist*. The finite zero-localization row is the route that reads the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, the* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, the* $\mathsf{Real}^{\uparrow}$ *real-part row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, and the local transport, continuation, package, and naming rows* H, C, P, N*. It localizes only the displayed witness packet and does not supply analytic continuation, a global zero-set theorem, host complex equality, quotient representative data, or ambient completeness.*

Proof. Project the carrier of Definition 548.1. The rows Z, S, R, Q, H, C, P, N are displayed coordinates of the same finite BHist packet, so the zero-localization read has no source outside that packet.

Use the zero-row soundness and strip-localization surfaces already established in Definition 548.158 and Definition 548.157. They force the supplied zero row and critical-strip row to be read together, with the real and rational comparison rows consumed inside the same local route.

The sibling surfaces chapter 38 and chapter 37 provide only the displayed Real and Rat rows. The ledger therefore cannot import analytic continuation, global zero-set data, host equality, quotient representatives, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero localization source window`

Theorem 548.161 (Critical-line witness rational height-window obligation)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, each rational height-window request is answered through the modulus row* M *and the rational comparison ledger* Q *as a finite unary-count budget. The request may read the supplied zero row* Z*, strip row* S*, real-part row* R*, and local* H, C, P, N *rows, but it cannot obtain a host zero search, an analytic-continuation certificate, a global height enumeration, or an unbounded choice selector.*

Proof. Start with the modulus coordinate M. By carrier projection, it is the only row that answers depth requests, and it is packaged with R, Q, H, C, P, N inside the accepted witness packet.

Apply Definition 548.4 and Definition 548.11. These theorems route the height-window read through the displayed $\mathsf{Real}^{\uparrow}$ real-part row and $\mathsf{Rat}^{\uparrow}$ comparison ledger, using only the finite depth budget recorded by M.

The finite-count boundary is reinforced by chapter 573. That ledger supplies only bounded counting rows, so the request does not become a host zero search, analytic-continuation certificate, global height enumeration, or unbounded choice selector. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rational height window`

Theorem 548.162 (Critical-line witness critical-strip ledger obligation)**.** *Every ledger read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet remains on the displayed finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The ledger may expose the supplied zero row, critical-strip row, rational height window, Real real-part row, and local replay rows, but it refuses analytic-continuation data, global zero-set data, quotient representatives, host equality rows, selected Real representatives, and ambient completeness.

Proof. Project the accepted carrier. The route displayed in Definition 548.1 lists exactly the zero, strip, modulus, real, rational, transport, continuation, package, and naming rows.

Combine the zero-localization obligation with the rational height-window obligation above. The first binds the supplied zero and strip rows to the same finite localization route; the second binds height requests to the finite modulus and rational comparison ledger.

Finally apply Definition 548.159. The existing ledger boundary already refuses analytic continuation, global RH proof data, host complex-zero classifiers, quotient representatives, host equality, and ambient completeness, so the critical-strip ledger is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier critical strip ledger`

Theorem 548.163 (Critical-line witness public export)**.** *An accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet exports exactly the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row, the* $\mathsf{CritStrip}^{\uparrow}$ *row, the rational height-window modulus, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, and the local transport, replay, provenance, and naming rows. It exports no analytic continuation, global zero-set theorem,* RH *proof, quotient representative, host equality, or ambient completeness.*

Proof. First read the zero-localization obligation. It binds the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row, $\mathsf{CritStrip}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ real-part row, $\mathsf{Rat}^{\uparrow}$ comparison ledger, and local structural rows to the same finite packet.

Next read the rational height-window obligation. It routes every height request through the displayed modulus row and rational comparison ledger before any downstream consumer receives the finite window.

Finally combine the critical-strip ledger obligation with Definition 548.159. The former exhausts the public ledger by $Z, S, M, R, Q, H, C, P, N$, and the latter excludes analytic continuation, global zero-set data, RH proof data, quotient representatives, host equality, and ambient completeness from the carrier. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier public export certificate`

Theorem 548.164 (Critical-line witness root-zero selector exhaustion)**.** *Every selector or localization consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *is exhausted by the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*,* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, modulus row* M*,* $\mathsf{Real}^{\uparrow}$ *row* R*,* $\mathsf{Rat}^{\uparrow}$ *comparison row* Q*, and the local* H, C, P, N *transport, replay, provenance, and naming rows. No such consumer can introduce analytic continuation, global zero-set data,* RH *proof data, host complex equality, quotient representatives, selected Real representatives, or ambient completeness.*

Proof. Start with Definition 548.160. It forces every zero-localization read to consume the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row, $\mathsf{CritStrip}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ real-part row, $\mathsf{Rat}^{\uparrow}$ comparison row, and local structural rows inside the same finite packet.

Next apply Definition 548.161. It routes selector depth and height-window requests through the displayed modulus row M and rational comparison ledger Q, so a consumer cannot replace the finite budget with a host zero search or unbounded choice selector.

Now use Definition 548.162. The critical-strip ledger is exactly the displayed route $Z, S, M, R, Q, H, C, P, N$, and therefore a localization consumer has no additional analytic coordinate after those rows are read.

Finally combine this with Definition 548.163 and Definition 548.159. The public export identifies the same finite surface, while the ledger boundary excludes analytic continuation, global zero-set data, RH proof data, host complex equality, quotient representatives, selected Real representatives, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero localization selector exhaustion`

Theorem 548.165 (Critical-line witness root-zero localization scope)**.** *Every accepted root-zero localization selector read of a* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is scoped exactly by the finite rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The selector may consume the supplied $\mathsf{CritStrip}^{\uparrow}$ *row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison data carried by the rational height-window route, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the displayed* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *row, the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row, and the local transport, replay, provenance, and naming rows. This is only zero-localization evidence: it does not assert existence of a zero,* RH *truth, analytic continuation beyond the displayed witness, a classical critical-line theorem, host complex equality, quotient representatives, selected Real representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero localization scope`

Proof. The zero-localization obligation Definition 548.160 fixes the source of a localization selector read. It binds the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row, $\mathsf{CritStrip}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ real-part row, $\mathsf{Rat}^{\uparrow}$ comparison row, and local structural rows to one finite BHist packet; it does not create a zero-existence coordinate.

The rational height-window theorem Definition 548.161 adds the only admitted budget for selector depth. Together with chapter 291, chapter 301, and chapter 573, it routes bounded height information through finite dyadic, regular rational, modulus, and comparison rows rather than through an unbounded zero search.

The critical-strip ledger theorem Definition 548.162 and public export theorem Definition 548.163 identify the exported surface as exactly $Z, S, M, R, Q, H, C, P, N$. The sibling chapters chapter 73,

chapter 72, chapter 38, chapter 580, and the rational-data siblings contribute only their displayed rows to that surface.

Finally apply Definition 548.164. It exhausts every localization consumer by the same rows and refuses analytic continuation beyond the displayed witness, global zero-set data, RH proof data, classical critical-line closure, host equality, quotient representatives, selected Real representatives, and ambient completeness. Therefore the selector scope is exactly the displayed packet route, while zeta-zero existence remains outside the selector claim. □

Theorem 548.166 (Critical-line witness root source-route completion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The root source route is complete exactly when the zero-strip modulus row M, the fixed critical-line source row S, the supplied zeta-continuation witness row Z, the refusal ledger inside Q, the* BHist *carrier, the componentwise* hsame *row H, the displayed* Cont *row C, the* Pkg *row P, and the local* NameCert *row N are read as one finite packet.*

Every consumer of the root obligation surface must pass through that route before it can inspect the critical-line witness. The packet exports no analytic continuation theorem, no RH *proof, no host complex equality, and no off-packet zero localization.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source route completion`

Proof. Start with the zero-localization obligation of Definition 548.160. It binds the zero row, critical-strip row, Real row, rational comparison ledger, and local transport, continuation, package, and naming rows to one finite BHist packet.

Next use Definition 548.157 and Definition 548.158. These force the supplied zero row and fixed strip source to remain a displayed witness read rather than an analytic construction or host zero search.

Finally apply Definition 548.162 and Definition 548.159. The refusal ledger and local routing rows exhaust every public root read, so analytic continuation, RH, host complex equality, and off-packet zero localization have no carrier coordinate. □

Theorem 548.167 (Critical-line witness zero-strip localization exhaustion)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, a zero-strip localization request is exhausted by the displayed rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The request may consume the zero-strip modulus row, fixed critical-line source row, zeta-continuation witness row, refusal ledger, BHist, hsame, Cont, Pkg, *and* NameCert *rows, but it cannot obtain analytic continuation as a completed theorem,* RH, *host complex equality, or zero localization outside the packet.*

Proof. Project the carrier and read the source rows through Definition 548.166. This places the modulus, fixed source, supplied continuation witness, refusal ledger, transport, route, provenance, and local name in one finite route.

Apply Definition 548.158. The zero row is sound only as a supplied row of the accepted packet, so a localization request cannot turn it into a host search or a global zero-set theorem.

Use Definition 548.161. Height-window access is bounded by the finite modulus and rational comparison ledger. Thus the localization surface is exhausted before any analytic continuation theorem, RH proof, host complex equality row, or off-packet localization can be read. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip localization exhaustion`

Theorem 548.168 (Critical-line witness RH-boundary source lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The* RH *boundary for this packet is locked to the same zero-strip modulus row, fixed critical-line source row, zeta-continuation witness row, refusal ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows used by the root source route. A consumer can cite that boundary only as a refusal and localization surface for the displayed packet.*

In particular, the boundary does not supply analytic continuation as a completed theorem, prove RH*, identify host complex values, or localize zeros away from the accepted source packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRhBoundarySourceLock`

Proof. Begin with Definition 548.166. It identifies the finite route that every root obligation read must consume.

Then apply Definition 548.167. The same route exhausts zero-strip localization and prevents a second read outside $Z, S, M, R, Q, H, C, P, N$.

Finally use Definition 548.72. The refusal ledger records requests for RH proof data as boundary reads, not as generated theorem content. Since the accepted packet has no coordinate for analytic continuation, host complex equality, or off-packet zero localization, the boundary remains locked to the displayed source route. □

The CriticalLineWitness$^{\uparrow}$ root package is a witness surface over existing zeta and RH boundary rows, not an independent zeta-function object. Its zero-strip and critical-line reads consume the ZetaZeros$^{\uparrow}$ sibling surface and the RH horizon before any modulus witness is public.

Theorem 548.169 (Critical-line witness zeta dependency lattice)**.** *Every* CriticalLineWitness$^{\uparrow}$ *zero-strip or critical-line read consumes the* ZetaZeros$^{\uparrow}$ *zero-count surface of chapter 73 or the RH-boundary surface of chapter 9239 before it can expose a modulus witness. The packet supplies a finite witness surface through $Z, S, M, R, Q, H, C, P, N$; it does not construct a separate zeta-function object, zero oracle, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZetaLink`

Proof. Start with the root source package already displayed by the hub. The supplied zero row Z and critical-strip row S enter through the root zero-strip route, while M, R, Q remain the later modulus and rational-comparison surface.

Apply Definition 548.67 and Definition 548.141. These sibling rows project the zero-strip and critical-line data to the existing ZetaZeros$^{\uparrow}$ and critical-strip boundary before any modulus read is exposed.

Next use Definition 548.84 and Definition 548.143. They place downstream RH-facing and zero-count consumers behind the same finite route and explicitly refuse zero search, analytic-continuation export, host equality, quotient representatives, and an RH proof.

It remains only to read the modulus witness. By Definition 548.68, such a read passes through M, R, Q, H, C, P, N after the zero-strip and RH boundary rows have been supplied. Hence CriticalLineWitness$^{\uparrow}$ records a dependency lattice over the zeta and RH siblings, not a standalone zeta-function certificate. □

Theorem 548.170 (Critical-line witness root source-readiness triad)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every root consumer read is source-ready only after the zeta dependency row, the fixed strip-source rows, and the root non-escape package have all been displayed. Concretely, the consumer must read the* ZetaZeros$^{\uparrow}$ *row Z, the critical-strip row S, the*

modulus and comparison rows M, R, Q, and the transport, continuation, provenance, and naming rows H, C, P, N as one finite BEDC surface before any RH-facing or zeta-facing obligation is public.

This source-readiness triad supplies no proof of RH*, no analytic continuation theorem, no host zero selector, no quotient equality, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source readiness triad`

Proof. Begin with the zeta dependency lattice, Definition 548.169. It makes the root read consume the existing $\mathsf{ZetaZeros}^{\uparrow}$ and RH-boundary surfaces before a modulus witness can be exposed.

Next apply root-strip source totality, Definition 548.250. This places the $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source rows on the displayed route $Z, S, M, R, Q, H, C, P, N$.

Finally use the root non-escape package, Definition 548.69. Its refusal rows remove proof-of-RH content, analytic-continuation export, host zero selection, quotient equality, host equality, and ambient completeness from the source-ready route. □

Theorem 548.171 (Critical-line witness root zeta-strip non-escape triad)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the public zeta/strip/non-escape surface is exactly the triad consisting of the zeta dependency lattice, the root strip-source totality route, and the root zeta ledger non-escape row. Every zeta-facing or strip-facing downstream consumer factors through that triad before it can read the modulus side M, R, Q.

The triad is a finite BEDC refusal surface. It does not prove RH*, does not prove analytic continuation, does not choose a host zero, does not identify quotient representatives, and does not import ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zeta strip nonescape triad`

Proof. First read the zeta dependency lattice through Definition 548.169. The dependency row orders every zero-strip and critical-line read through the sibling zeta and RH-boundary surfaces before any modulus witness is public.

Now attach the strip-source route. By Definition 548.250, the public consumer must expose $Z, S, M, R, Q, H, C, P, N$ and the listed sibling source rows before a critical-line-facing read can begin.

Finally use root zeta ledger non-escape, Definition 548.198. The zeta ledger remains the existing continuation and zero-count boundary together with the local packet; it exports no RH proof and no analytic-continuation theorem beyond the named finite surface. □

Theorem 548.172 (Critical-line witness root RH-refusal boundary)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every RH-facing consumer read of the root packet factors through the source-readiness triad of Definition 548.170, the zeta/strip non-escape triad of Definition 548.171, and the existing root RH-boundary refusal rows.*

Thus the public boundary is a refusal package over the displayed $Z, S, M, R, Q, H, C, P, N$ route. It explicitly refuses an RH proof, an analytic continuation proof, a host zero selector, quotient equality, host equality, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh refusal boundary`

Proof. Start with the source-readiness triad. It places every RH-facing consumer on the displayed zeta, strip, modulus, transport, continuation, provenance, and local naming rows before the boundary can be read.

Next use the zeta/strip non-escape triad. That triad keeps zeta-facing and strip-facing demands inside the finite sibling surfaces and the local packet, so the RH-facing read has no separate analytic or quotient coordinate.

Close with the root RH-boundary refusal rows, Definition 548.94 and Definition 548.72. These rows classify RH proof requests, analytic-continuation proof requests, host zero-selector requests, quotient-equality requests, host-equality requests, and ambient-completeness requests as outside the accepted packet. □

Theorem 548.173 (Critical-line witness root zeta-strip source lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root packet read through W must jointly expose the* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *rows before any modulus-side or RH-facing consumer can read M, R, Q.*

The source lock exports no proof of RH*, no analytic-continuation proof, no selected zero, no quotient representative, and no ambient completeness principle.*

Proof. Start with the source-readiness triad of Definition 548.170. It places the zeta dependency row, the fixed strip-source rows, and the root non-escape package on the displayed packet before any RH-facing or zeta-facing obligation is public.

Apply the zeta/strip non-escape triad Definition 548.171 together with the RH-refusal boundary Definition 548.172. These two theorems keep modulus-side access downstream of the same finite route and remove the RH proof, analytic-continuation proof, selected-zero, quotient, and ambient-completeness exits.

Now use the zeta dependency lattice Definition 548.169 and root-strip source totality Definition 548.250. They expose $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the consumer reaches M, R, Q.

Finally attach the Phase Real source rows already carried by the root source window: $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ remain visible through the same $Z, S, M, R, Q, H, C, P, N$ packet route. The RH refusal rows then guarantee that this joint exposure is a source lock, not an analytic proof or an ambient completeness import. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zeta strip source lock`

Theorem 548.174 (CriticalLineWitness zeta source lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every zeta-facing read of W factors through the supplied zero row Z, the fixed critical-strip row S, the modulus row M, the rational comparison row Q, and the displayed support rows H, C, P, N before any* RH*-facing consumer can inspect it. The zeta-facing route is therefore a finite source lock over* BHist*,* hsame*,* Cont*,* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row, not an analytic continuation theorem or a host zero-classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zeta source lock package`

Proof. Start from the dependency lattice of Definition 548.169. It places every zeta or critical-line read behind the supplied zero row and the fixed strip row before any modulus witness is visible.

Use the carrier and local NameCert obligations in Definition 548.2. They bind the modulus, rational comparison, transport, replay, provenance, and local naming rows to the same accepted packet.

Finally apply the source-modulus public readiness surface, Definition 548.364. The readiness route exposes $Z, S, M, R, Q, H, C, P, N$ together and leaves no field from which a zeta-facing read could bypass the finite source lock before the RH consumer sees it. □

Theorem 548.175 (CriticalLineWitness source-modulus zeta readback)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the source surface and the modulus surface share the same displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support. A zeta-facing readback may distinguish the supplied zero row, the fixed-strip source row, the modulus row, the real-part row, and the rational comparison row, but it cannot detach any of them from the shared support route* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus zeta readback`

Proof. Project the accepted packet through the carrier and NameCert rows of Definition 548.2. The source and modulus coordinates are rows of one finite packet, with H, C, P, N carrying transport, replay, provenance, and local naming.

Apply the zeta dependency lattice Definition 548.169. It keeps zero-strip and critical-line reads on the same finite witness surface before the modulus read is exposed.

Use Definition 548.364 to read the source and modulus surfaces as one public route. Thus the zeta readback can replay the individual rows only through their common BHist, hsame, Cont, Pkg, and NameCert support. □

Theorem 548.176 (CriticalLineWitness RH consumer source nonescape)**.** *Every* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the finite source, modulus, and zeta packet*

$$Z, \quad S, \quad M, \quad R, \quad Q, \quad H, \quad C, \quad P, \quad N.$$

The consumer receives no proof of RH*, analytic-continuation theorem, host equality, quotient zero set, global zero classifier, ambient completeness row, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer source nonescape`

Proof. Begin with the source lock above. It forces zeta-facing inspection through the supplied zero, fixed strip, modulus, comparison, and displayed support rows before any RH-facing consumer sees the packet.

Next use the source-modulus zeta readback theorem. It keeps the source and modulus surfaces on the same BHist, hsame, Cont, Pkg, and NameCert route, so the consumer cannot detach a host source from the finite readback.

Finally apply Definition 548.365. That consumer boundary removes analytic continuation, host equality, quotient representatives, ambient completeness, zero search, selected host limits, and RH proof content. Since the accepted packet also has no standard bridge coordinate, the RH-facing consumer receives only the finite source, modulus, and zeta rows listed above. □

Theorem 548.177 (Critical-line witness RH-interface carrier)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Its RH-facing carrier exposes only the critical-line row read through R, Q, the zero-strip row Z, S, the modulus row M, the source-lock row determined by the displayed zero-strip source, the componentwise transport row H, the route row C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. A downstream consumer may cite the sibling* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *surface of chapter 580, but it receives no RH truth claim, no nontrivial-zero location theorem, and no completed* $\mathsf{Gamma}^{\uparrow}$ *certificate from this carrier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh interface carrier`

Proof. Begin with the source exposure package of Definition 548.126. It opens the zero-strip source, modulus, real-part, and rational-comparison rows as the finite surface of W.

Next use the source-modulus interface of Definition 548.59 and the modulus ledger of Definition 548.68. These rows keep the critical-line, zero-strip, and modulus reads attached to Z, S, M, R, Q before any downstream consumer sees the packet.

Finally use the sibling $\mathsf{ZetaContinuationWitness}^{\uparrow}$ refusal rows, especially Definition 580.6 and Definition 580.7. They allow the continuation surface to be cited as a finite input boundary while refusing RH truth, zero-location, and completed Gamma data. Thus H, C, P, N authenticate only the displayed finite route. □

Theorem 548.178 (Critical-line witness modulus ledger exhaustion)**.** *Every zero-strip or RH-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet must read the displayed modulus ledger M, R, Q, H, C, P, N before any downstream use of the zero-strip row, critical-line row, or sibling* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *route. The consumer receives the ledger as a finite BEDC source surface, not as an RH proof, a zero-location theorem, or a completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus ledger exhaustion`

Proof. Project the consumer route through Definition 548.143. It requires zero-strip consumers to pass through $Z, S, M, R, Q, H, C, P, N$, so the modulus ledger is already part of the visible packet.

Apply the downstream RH-boundary refinement of Definition 548.105. The RH-facing route is exhausted by the same displayed rows and refuses analytic continuation, quotient, host-equality, global-zero, and RH-proof exits.

The sibling $\mathsf{ZetaContinuationWitness}^{\uparrow}$ boundary contributes only its finite source lock and refusal surface. By Definition 580.4 and Definition 580.7, continuation reads are displayed before downstream use and do not supply RH truth, zero-location, or completed Gamma data. Hence the local modulus ledger is mandatory for every downstream read. □

Theorem 548.179 (Critical-line witness downstream source package)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the critical-line and zero-strip rows assemble with the displayed modulus ledger into the finite source package consumed by zeta-boundary downstream chapters:*

$$Z, S, M, R, Q, H, C, P, N.$$

This package is the downstream source surface for RH-boundary reads and zeta-continuation handoffs; it does not assert RH, locate a nontrivial zero, or close the Gamma side of the functional-equation route.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream source package`

Proof. Start with the RH-interface carrier above. It identifies the rows that can be exposed to downstream consumers and excludes RH truth, zero-location, and completed Gamma coordinates from the carrier.

Use the modulus ledger exhaustion theorem. Every zero-strip, critical-line, or RH-facing consumer must read M, R, Q, H, C, P, N together with the source rows before downstream use.

Finally combine the local package with the zeta dependency lattice of Definition 548.169 and the sibling $\mathsf{ZetaContinuationWitness}^{\uparrow}$ source lock of Definition 580.4. The resulting finite surface is exactly $Z, S, M, R, Q, H, C, P, N$, suitable for downstream zeta-boundary chapters and nothing stronger. □

Theorem 548.180 (Critical-line witness zero-strip source totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every supplied zero or critical-strip source read of W is total on the displayed finite route Z, S, H, C, P, N before any downstream* RH*-facing consumer inspects the packet. The source read may cite the root-source rows of Definition 548.76, the comparison-ledger rows of Definition 548.77, and the RH-refusal rows of Definition 548.72, but it remains a finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface rather than an* RH *proof datum or analytic-continuation import.*

Proof. Read the accepted carrier through its zero-strip source coordinates. The rows Z and S are supplied rows of the packet, while H, C, P, N give the componentwise transport, displayed route, package provenance, and local name.

Apply the root-source triad and root comparison ledger. These rows place source and comparison reads on the same packet $W = (Z, S, M, R, Q, H, C, P, N)$, so the zero-strip source is total without leaving the finite BEDC surface.

Finally use the RH-refusal ledger. It removes any route to RH proof data, host zero classification, analytic continuation, quotient representatives, host equality, or ambient completeness, leaving exactly the finite source route stated above. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip source totality`

Theorem 548.181 (Critical-line witness modulus-depth ledger)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every modulus-depth, real-part, or rational-comparison read of W factors through the displayed rows M, R, Q, H, C, P, N, with Z and S retained as the supplied zero-strip source coordinates. The read may use the root-source, comparison-ledger, and RH-refusal rows, but it is only a finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *ledger for supplied zero, strip, modulus, and rational comparison data; it is not an* RH *proof or an analytic-continuation certificate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus depth ledger certificate`

Proof. Project W to its modulus and comparison coordinates. The depth row M, real-part row R, rational comparison row Q, and local rows H, C, P, N are the only displayed coordinates that answer modulus-depth requests.

Use the root comparison ledger to identify every rational comparison with Q and every depth read with the same local route. The root-source rows keep Z and S attached to the read, so the ledger is still a row of the accepted packet.

The RH-refusal rows then block analytic-continuation imports, global zero classifiers, host equality, quotient representatives, ambient completeness, and RH proof data. Thus the ledger is total as a finite BEDC read and no stronger object is exported. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus depth ledger`

Theorem 548.182 (Critical-line witness zeta dependency readback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Any zeta-dependency readback that cites W reads only the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison row Q, and the local H, C, P, N rows already exposed by the root-source, comparison-ledger, and RH-refusal surfaces. The dependency readback remains finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data and imports no analytic continuation theorem, completed zeta-function object, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta dependency readback`

Proof. Start with the displayed packet $W = (Z, S, M, R, Q, H, C, P, N)$. A zeta-dependency readback can cite the packet only through its supplied source rows, its modulus and comparison ledger, and its local transport, route, provenance, and naming rows.

Combine the root-source triad with the comparison ledger. These force every readback to use the same finite coordinates and prevent a separate zeta-continuation carrier from being substituted for the accepted packet.

Finally apply the RH-refusal ledger. Since analytic continuation, completed zeta data, global zero classification, host equality, quotient representatives, ambient completeness, and RH truth are absent from the accepted rows, the dependency readback is exactly the finite BEDC surface described in the statement. □

Theorem 548.183 (Critical-line witness root finite strip route)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every root zero read and every root critical-strip read of W exposes only the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by Z, S, H, C, P, N before any downstream* RH*-facing consumer reads the packet. The later modulus, real-part, and rational-comparison rows may be consulted only through the same displayed carrier route $Z, S, M, R, Q, H, C, P, N$.*

Proof. Project the accepted packet through Definition 548.1. The carrier supplies exactly the displayed rows $Z, S, M, R, Q, H, C, P, N$, with H, C, P, N recording the finite transport, continuation, provenance, and local naming surface.

Apply the root source exposure theorem Definition 548.126. It forces a root-facing consumer to see the supplied zero and strip rows as displayed source data before it can use the packet downstream.

Finally use the zero-strip source totality theorem Definition 548.180. The total route remains inside finite BHist, hsame, Cont, Pkg, and NameCert rows, so no downstream RH-facing consumer receives an unexposed source coordinate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite strip route`

Theorem 548.184 (Critical-line witness root modulus read totality)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every modulus, depth, real-part, or rational-comparison read factors through the displayed rows M, R, Q, H, C, P, N and the existing* Rat$^{\uparrow}$/Real$^{\uparrow}$ *comparison surface. No hidden numeric oracle, host real equality, quotient representative, analytic-continuation row, or ambient completeness input can answer the read.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus read totality`

Proof. Start with the root modulus-window totality theorem Definition 548.85. It identifies M, R, Q, H, C, P, N as the whole finite route for modulus-window reads.

Now apply the root comparison ledger Definition 548.77. The rational and real comparison data are read through the displayed $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows, transported by H, replayed by C, recorded by P, and named by N.

Since the displayed carrier has no coordinate for a hidden numeric oracle, host equality, quotient representative, analytic continuation, or ambient completeness, the modulus read is total only through the stated route. □

Theorem 548.185 (Critical-line witness root refusal ledger scope)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every request for* RH *proof data, global zero classifiers, analytic-continuation certificates, quotient representatives, host equality, or ambient completeness is refused inside the local ledger over* $Z, S, M, R, Q, H, C, P, N$*. The refusal ledger is not exported as a proof surface or as any of the refused objects.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal ledger scope`

Proof. Apply the root refusal route Definition 548.78. It sends root-facing off-route demands to the displayed refusal boundary rather than to analytic or host-level data.

Then use RH-refusal ledger completeness Definition 548.72. That theorem accounts for RH proof requests, zero-search and global-classifier requests, quotient representatives, host equality, analytic continuation, and ambient completeness inside the same finite ledger.

The ledger rows remain $Z, S, M, R, Q, H, C, P, N$. They can be read as local refusal data, but no theorem in the route repacks them as an exported RH proof, global zero classifier, analytic-continuation certificate, quotient representative, host equality witness, or completeness principle. □

Theorem 548.186 (CriticalLineWitness root source-window route lock)**.** *Let* W *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet. Its* BHist *source rows, zero-strip source rows, modulus-window comparison rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row form one finite root source-window route. Every RH-facing consumer read of* W *factors through that route and cannot attach a second analytic source, second modulus window, hidden zero oracle, ambient zeta equality, or host-level RH decision row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source window route lock`

Proof. Start from the root source exposure package and the finite route scope for $\mathsf{CriticalLineWitness}^{\uparrow}$. These rows expose the root source, strip source, modulus comparison, transport, continuation, provenance, and local name as finite BHist coordinates.

The route lock is then componentwise. The hsame rows transport the source and modulus coordinates already displayed by the carrier, while the Cont rows replay only those transported coordinates for downstream consumers.

Since the Pkg and NameCert rows record the same finite route, a consumer cannot project an additional analytic source, hidden zero oracle, ambient zeta equality, or host RH decision from the packet. Those coordinates are not carrier rows and are therefore outside the root read. □

Theorem 548.187 (CriticalLineWitness root modulus comparison exhaustion)**.** *Every modulus-comparison read admitted by a* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet is exhausted by the displayed* BHist

source row, zero-strip row, finite modulus-window row, hsame *transport row,* Cont *replay row,* Pkg *provenance row, and local* NameCert *row. The comparison may certify the availability and transport of the displayed modulus data, but it does not assert a zero count, a zero-free strip, or the Riemann hypothesis.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus comparison exhaustion`

Proof. Use the root source-modulus interface to locate the comparison inside the source and window rows of the accepted packet. The comparison is a finite ledger read over those rows, not a theorem about the zero set itself.

Apply the root modulus readback and source-modulus closure rows. They show that transporting a displayed comparison by hsame and replaying it by Cont keeps the read inside the same finite BHist route and the same Pkg provenance.

The final boundary is the local NameCert exactness row. It permits the consumer to read the supplied modulus comparison and its provenance, while zero counts, zero-free-strip conclusions, and RH conclusions remain outside the packet. □

Theorem 548.188 (CriticalLineWitness root RH consumer refusal totality)**.** *Every RH-facing consumer request to a* CriticalLineWitness$^{\uparrow}$ *root packet is total as a finite readback and total as a refusal boundary: accepted reads return only the displayed* BHist *source, modulus comparison,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, while requests for a zero theorem, a zero-free critical strip, a hidden analytic continuation equality, a completed RH proof row, or a host RH truth value are refused.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh consumer refusal totality`

Proof. First read the accepted side through the root source-window route lock of Definition 548.186. It gives every RH-facing consumer the finite source, modulus, transport, continuation, provenance, and naming rows already present in the packet.

Next apply the modulus comparison exhaustion of Definition 548.187. The comparison row is available as a displayed finite obligation row, but it is not a zero-counting row and not an RH row.

The refusal side follows from the same inventory. A zero theorem, zero-free critical strip, hidden analytic continuation equality, completed RH proof row, or host RH truth value would require a coordinate absent from BHist, hsame, Cont, Pkg, and NameCert for this packet, so the request is refused rather than exported. □

Theorem 548.189 (Critical-line witness RH-refusal readback totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every* RH*-refusal readback of* W *factors through the displayed zero-strip, modulus, and source rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The readback may cite the zero-strip/source totality of Definition 548.180, the modulus-depth ledger of Definition 548.181, and the RH-interface carrier of Definition 548.177, but it exports only a finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *refusal row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal readback totality`

Proof. Project the accepted packet to its displayed zero, strip, modulus, real-part, rational-comparison, transport, route, provenance, and naming rows. These are exactly the coordinates available to a refusal readback.

Use Definition 548.180 for the Z, S, H, C, P, N side and Definition 548.181 for the M, R, Q, H, C, P, N side. The two routes share the same structural rows, so the readback remains one finite packet route.

Finally apply the RH-interface carrier and the existing refusal ledger. Since the packet has no coordinate for an RH theorem, analytic continuation theorem, host zero classifier, quotient representative, host equality, or ambient completeness principle, the refusal readback is exhausted by the displayed rows. □

Theorem 548.190 (Critical-line witness zeta dependency lock)**.** *Before any* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is exported as a critical-line witness, its dependency rows for* $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are locked into the same finite route* $Z, S, M, R, Q, H, C, P, N$. *The continuation source is read only as a finite boundary surface, the zero row is read only through* Z, *the rational comparison ledger is read only through* Q, *and the real-part row is read only through* R.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta dependency lock`

Proof. Start with the zeta dependency lattice of Definition 548.169. It forces zero-strip and critical-line reads to consume the zeta-zero and RH-boundary surfaces before the modulus witness is public.

Next use Definition 548.182 and Definition 548.179. These theorems identify every zeta-facing readback with the supplied zero row, critical-strip row, modulus row, real-part row, rational-comparison row, and local structural rows.

The $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ sources are therefore not late imports. They are the displayed rows Q and R already bound to the accepted packet, while $\mathsf{ZetaContinuationWitness}^{\uparrow}$ and $\mathsf{ZetaZeros}^{\uparrow}$ remain provenance and source surfaces consumed before export. □

Theorem 548.191 (Critical-line witness downstream counter-surface)**.** *Every downstream counter-surface consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the finite* BHist *refusal ledger over* $Z, S, M, R, Q, H, C, P, N$. *The consumer may inspect the supplied zero-strip source, modulus window, real-part row, rational comparison row, transport row, continuation row, provenance row, and local* NameCert *row, but it receives no proof or refutation of* RH, *no analytic continuation theorem, and no host zero classifier.*

Proof. Project the downstream read through Definition 548.87. That gate already restricts RH-facing consumers to the finite refusal package over the accepted packet.

Apply the RH-refusal readback totality theorem above. It identifies every such readback with the zero-strip, modulus, source, transport, route, provenance, and naming rows of W.

The non-export clauses in Definition 548.134 and Definition 548.177 remove proof content, analytic continuation, and host zero classification from the consumer surface. Hence downstream consumers receive a finite counter-surface ledger and nothing stronger. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream counter surface`

Theorem 548.192 (Critical-line witness modulus source totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before any zero-facing consumer reads* W, *the packet*

exposes its zeta source row, critical-strip row, rational-modulus row, zero-window row, componentwise transports, displayed routes, package provenance, and local naming row through the finite surface

$$Z, S, M, R, Q, H, C, P, N.$$

Thus the consumer first receives BHist *source data,* hsame *transport,* Cont *locality,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row, not a late analytic source or an unlisted zero-location coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus source totality`

Proof. Begin with the source exposure package of Definition 548.126. It opens the zero-strip, rational-height, real-part, and modulus rows as finite BHist data before downstream use.

Next apply the zero-strip source totality theorem, Definition 548.180. The Z, S source side and the H, C, P, N structural side are read together, so the zero-facing consumer cannot bypass the displayed hsame, Cont, Pkg, and NameCert rows.

Finally use the RH-interface carrier and the RH-refusal readback totality theorem, Definition 548.177 and Definition 548.189. These identify the same packet route as the only RH-facing readback surface, with M, R, Q carrying the modulus and rational-comparison data before any consumer sees the packet. □

Theorem 548.193 (Critical-line witness zero-strip ledger exhaustion)**.** *Every zero-strip read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is exhausted by the displayed critical-strip row, rational-modulus row, zero-window row, continuation rows, componentwise transports, package provenance, and local* NameCert *row of*

$$Z, S, M, R, Q, H, C, P, N.$$

The read cannot import a global zero locator, an ambient zero-search oracle, an analytic-continuation theorem, or a source row outside the finite BHist*,* hsame*,* Cont*, and* Pkg *ledger.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip ledger exhaustion`

Proof. Read the zero-strip request through Definition 548.180. The supplied zero and strip rows are already total on the finite route Z, S, H, C, P, N.

Use the modulus-depth ledger of Definition 548.181 together with the modulus ledger exhaustion of Definition 548.178. These force the rational-modulus, real-part, and rational-comparison reads through M, R, Q, H, C, P, N before the zero-strip request is consumed.

Apply the RH-refusal readback totality theorem. Since every refusal readback is the same finite packet route, no global zero locator or unlisted analytic branch can be projected from the zero-strip ledger. □

Theorem 548.194 (Critical-line witness RH-boundary refusal totality)**.** *Every RH-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet remains a refusal-boundary read over the finite rows*

$$Z, S, M, R, Q, H, C, P, N.$$

It exports no RH *predicate, no global zero-location theorem, no completed* $\mathsf{Gamma}^{\uparrow}$ *certificate, and no hidden analytic branch. The exported object is only the displayed* BHist *refusal surface stabilized by* hsame*, routed by* Cont*, carried by* Pkg*, and named by the local* NameCert*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary refusal totality`

Proof. Start from the RH-interface carrier, Definition 548.177. It exposes the critical-line, zero-strip, modulus, source-lock, transport, route, provenance, and naming rows while refusing RH truth, nontrivial-zero location, and completed Gamma data.

Next use Definition 548.189. Every RH-refusal readback factors through $Z, S, M, R, Q, H, C, P, N$, so the boundary read is not a separate analytic theorem.

Finally combine the source exposure package with the zero-strip ledger exhaustion theorem above. The consumer-facing route is already exhausted by the displayed source, modulus, zero-window, transport, continuation, provenance, and naming rows, leaving no coordinate for an RH predicate, global zero-location theorem, completed $\mathsf{Gamma}^{\uparrow}$ certificate, or hidden analytic branch. □

Theorem 548.195 (Critical-line witness root formal-target ledger)**.** *A future formal target for the* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root-modulus zero-strip surface must verify exactly the finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

Here Z is source exposure, S is the critical-strip row, M is the modulus-depth ledger, R is the real-part row, Q is the rational-comparison row, H is componentwise hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The target must refuse RH truth, global zero-location data, analytic continuation, completed* $\mathsf{Gamma}^{\uparrow}$ *data, and hidden analytic branches.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootFormalTargetLedger`

Proof. The source-exposure package fixes the source row Z before any zero-facing consumer reads the packet. The zero-strip ledger exhaustion theorem then identifies the displayed rows $Z, S, M, R, Q, H, C, P, N$ as the whole zero-strip surface.

The RH-boundary refusal totality theorem supplies the negative side of the ledger. It states that every RH-facing read remains a refusal-boundary read over the same route, with no RH predicate, global zero-location theorem, completed $\mathsf{Gamma}^{\uparrow}$ certificate, or hidden analytic branch.

Thus the formal target is not a global analytic theorem. It is the finite verification obligation that the listed source, strip, modulus, rational-comparison, transport, replay, provenance, and local naming rows exhaust the accepted root-modulus zero-strip surface. □

Theorem 548.196 (Critical-line witness root carrier classifier boundary)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the public carrier boundary exposes the displayed BHist *rows Z, S, M, R, Q, the componentwise* hsame *classifier-transport row H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N before any critical-line read of M or R is available. A root consumer may therefore enter the critical-line witness only through the finite packet surface of Definition 548.1 and the obligation rows of Definition 548.2.*

Proof. Project the carrier in Definition 548.1. Its displayed packet lists the zero row, strip row, modulus row, real-part row, rational comparison ledger, classifier transport, continuation route, provenance row, and local naming row as one finite BHist surface.

Apply Definition 548.2. Carrier admission exposes the tuple itself, zero-strip compatibility supplies the componentwise hsame classifier transport, modulus discipline keeps each critical-line read attached to M, R, Q, and ledger exactness keeps C, P, N inside the same accepted packet.

The downstream boundary is therefore ordered by the packet rows: H, C, P, N authenticate and route the displayed carrier before any public read reaches the critical-line modulus. The carrier has no alternate host equality, quotient, or analytic-completeness coordinate from which such a read could bypass the displayed boundary. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root carrier classifier boundary`

Theorem 548.197 (Critical-line witness root zero-strip supported handoff)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the zero-strip handoff from the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z *and* $\mathsf{CritStrip}^{\uparrow}$ *row* S *to a critical-line read is readable only through the* $\mathsf{Real}^{\uparrow}$ *row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, and the displayed modulus route* M, H, C, P, N*. The handoff may cite the sibling* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *certificate surfaces of chapter 38 and chapter 37, but it exports no* RH *proof and no zero-location theorem.*

Proof. Start with the supplied zero row and strip row. By Definition 548.39, Z is a boundary row for the accepted packet, readable only together with the remaining displayed coordinates and not as a zero constructor or global enumeration.

Next read the critical-line side through the modulus route. The modulus row of Definition 548.12 consumes the real-part row R and the rational comparison ledger Q, while Definition 548.11 identifies Q with the finite $\mathsf{Rat}^{\uparrow}$ comparison rows used by that route.

Finally apply Definition 548.31. Since the accepted packet supplies no universal zero classifier, no RH theorem, no zero-producing search, and no ambient analytic completeness, the zero-strip handoff is exactly the supported route through R, Q, M, H, C, P, N and the sibling $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ rows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero strip supported handoff`

Theorem 548.198 (Critical-line witness root zeta ledger non-escape)**.** *Every zeta-facing downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet is exhausted by the existing* ZetaContinuationWitness *continuation boundary, the local* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, the zero-strip boundary rows of Definition 548.39, and the refusal ledger of Definition 548.31. The consumer receives the named packet* $Z, S, M, R, Q, H, C, P, N$ *and the sibling continuation surface of chapter 580; it receives no proof of* RH *and no analytic-continuation theorem beyond that named packet.*

Proof. Begin with the local zeta row. The carrier of Definition 548.1 stores Z as supplied $\mathsf{ZetaZeros}^{\uparrow}$ input, and Definition 548.39 keeps that row tied to S, M, R, Q, H, C, P, N rather than turning it into a zero constructor or a global zeta-zero enumeration.

Now attach the continuation side only as a sibling boundary. The $\mathsf{ZetaContinuationWitness}^{\uparrow}$ chapter supplies its continuation handoff and consumer-ledger rows through chapter 580; the local packet may cite that surface, but the CriticalLineWitness carrier still exports only the displayed root packet.

Apply Definition 548.31. The refusal ledger excludes an RH proof, zero-producing search, universal zero classifier, and ambient analytic-completeness input. Therefore every zeta-facing downstream read is exhausted by the named continuation boundary, the local zero boundary, and the refusal rows already displayed by the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zeta ledger nonescape`

Theorem 548.199 (Critical-line witness root continuation zero-strip coherence)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, any continuation read* C *that reaches the zero-strip handoff preserves the same* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*,* $\mathsf{CritStrip}^{\uparrow}$ *row* S*,* $\mathsf{Real}^{\uparrow}$ *row* R*,* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. No consumer may replace this continuation route by analytic continuation, an* RH *proof, quotient equality, or ambient completeness.*

Proof. Project the carrier of Definition 548.1. The continuation row C is stored in the same finite packet as Z, S, M, R, Q, H, P, N.

Apply the root carrier classifier boundary, Definition 548.196. It forces every public read of the root packet through the displayed carrier, classifier transport, continuation, provenance, and local naming rows before any critical-line consumer reaches M or R.

Next apply the zero-strip supported handoff, Definition 548.197. The handoff from Z, S to a critical-line read must pass through R, Q, M, H, C, P, N, so the continuation cannot detach from the zero row, strip row, real row, rational comparison ledger, or local name.

Finally use zeta ledger non-escape, Definition 548.198. Since every zeta-facing read is exhausted by the named continuation boundary, local zero boundary, and refusal rows, no analytic-continuation theorem, RH proof, quotient equality, or ambient completeness coordinate can replace the displayed continuation route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root continuation zero strip coherence`

Theorem 548.200 (Critical-line witness phase Real readback boundary)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every Real-facing critical-line read must pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row* R*, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger* Q*, and the L10 dependency rows* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any modulus or critical-line consumer can inspect the packet. The readback route exports no* RH *proof, analytic-continuation theorem, quotient equality, selected zeros, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the root carrier classifier boundary, Definition 548.196. It forces every public read of the root packet through $Z, S, M, R, Q, H, C, P, N$ before a critical-line consumer reaches the modulus row or the Real-facing row.

Next apply the zero-strip supported handoff, Definition 548.197. The handoff from Z, S to a critical-line read is readable only through R, Q, M, H, C, P, N, so the $\mathsf{Real}^{\uparrow}$ row and $\mathsf{Rat}^{\uparrow}$ comparison ledger cannot be skipped.

Finally use zeta ledger non-escape, Definition 548.198, together with continuation zero-strip coherence, Definition 548.199. These rows keep the L10 dependencies $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ as named readback boundaries rather than completed ambient infrastructure.

Since the carrier has no coordinate for an RH proof, analytic continuation, quotient equality, selected zeros, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a modulus or critical-line consumer that bypasses those rows leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real readback boundary`

Theorem 548.201 (CriticalLineWitness root StdBridge refusal source)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet rooted in* BHist*. Before any* StandardBridge*-facing consumer reads* W *as a critical-line witness, the displayed zero-strip and refusal source is fixed by the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, critical-strip row* S*, depth row* M*, real-part row* R*, rational-comparison*

row Q, componentwise hsame *row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N.*

The fixed source is a finite refusal surface. It is not an analytic continuation theorem, host zero classifier, quotient-stream representative, ambient completeness principle, or proof of RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root stdbridge refusal source`

Proof. Project the accepted carrier to the displayed tuple $Z, S, M, R, Q, H, C, P, N$. These are the only BHist, transport, continuation, provenance, and local naming rows available before a bridge-facing consumer can read the witness.

The refusal surface consumes the zero row supplied by chapter 73 before any StandardBridge-facing consumer reads the packet, so the root source is anchored in the sibling ZetaZeros carrier rather than in a local-only witness route.

Use Definition 548.183 for the zero and strip side. It forces every root zero read and critical-strip read through the finite route before any downstream RH-facing consumer sees the packet.

Apply Definition 548.185 to the off-route demands. The refusal ledger records the absent analytic and host-level objects inside the same finite tuple, so the StandardBridge-facing source is fixed as the displayed refusal surface. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root stdbridge refusal source certificate`

Theorem 548.202 (CriticalLineWitness root zero-strip readback)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ rooted in* BHist*, every root zero-strip readback is exhausted by the supplied zero row Z, strip row S, modulus or depth row M, real-part row R, rational-comparison row Q, and the local H, C, P, N structural rows. No additional analytic continuation row, global zero classifier, host equality, quotient representative, countable selector, or ambient completeness datum can extend the readback.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zero strip readback`

Proof. Read the source side through Definition 548.204. The theorem lists $Z, S, M, R, Q, H, C, P, N$ as the full root source route for zero-strip consumers.

Next use Definition 548.184. It keeps modulus, depth, real-part, and rational-comparison reads inside the displayed numeric rows and the same structural route.

The remaining possible extensions are exactly the objects named in the refusal boundary. By Definition 548.206, analytic continuation export, global zero classification, host equality, quotient representatives, countable selector data, and ambient completeness stay absent as exported readback coordinates. □

Theorem 548.203 (CriticalLineWitness root RH refusal exhaustion)**.** *Every* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the refused finite witness surface over $Z, S, M, R, Q, H, C, P, N$. The consumer may inspect the supplied zero-strip source, modulus and comparison rows, real-part row, componentwise transport, continuation, provenance, and local* NameCert *rows, but receives no analytic continuation theorem, host critical-line fact, global zero classifier, quotient-stream equality, ambient completeness principle, or proof or refutation of* RH.

Proof. Begin with Definition 548.189. It states that every RH-refusal readback factors through the displayed zero-strip, modulus, source, transport, route, provenance, and naming rows.

Then apply Definition 548.191. Downstream consumers receive the finite refusal ledger over the same packet, not proof content or host zero classification.

Finally use Definition 548.201. The bridge-facing source has already been fixed as a finite refusal surface, so the RH consumer cannot strengthen it into analytic continuation, a host critical-line fact, a quotient-stream equality, ambient completeness, or an RH theorem. □

Theorem 548.204 (CriticalLineWitness root strip source totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet anchored in* BHist*. Every root consumer that asks for the zero-strip source must read the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, the critical-strip row S, the* $\mathsf{Nat}^{\uparrow}$ *depth row M, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, and the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows H, C, P, N. The source route may cite the sibling* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *surface, but it does not export analytic continuation, a global* $\mathsf{ZetaZeros}^{\uparrow}$ *classifier, host equality, quotient representatives, an* RH *proof, or ambient completeness.*

Proof. Project the accepted carrier. The rows $Z, S, M, R, Q, H, C, P, N$ are finite BHist coordinates, with H carrying componentwise hsame transport, C carrying the displayed Cont route, P carrying Pkg provenance, and N carrying the local naming row.

Use the supplied zero-strip route and the zero-strip source totality theorem, Definition 548.82 and Definition 548.180. They bind the zero and strip source side to Z, S, H, C, P, N before any downstream read can consult the modulus or comparison rows.

Next attach the numerical rows through the root comparison ledger and the modulus source totality theorem, Definition 548.77 and Definition 548.192. These rows force the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data to remain inside M, R, Q, H, C, P, N.

Finally read the zeta side only as a named sibling boundary. The $\mathsf{ZetaContinuationWitness}^{\uparrow}$ and $\mathsf{ZetaZeros}^{\uparrow}$ rows are consumed as displayed source coordinates, while the refusal ledger excludes analytic continuation, host equality, quotient representatives, global zero classification, RH proof data, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip source totality`

Theorem 548.205 (CriticalLineWitness root Gamma Real dependency)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet anchored in* BHist*. Any root read that mentions a* $\mathsf{Gamma}^{\uparrow}$*-facing or* $\mathsf{Real}^{\uparrow}$*-facing dependency must pass through the displayed real-part row R, rational comparison ledger Q, depth row M, supplied zero-strip rows Z, S, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *row P, and local* NameCert *row N. The dependency is a finite packet route over* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ZetaContinuationWitness}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$ *rows, not an analytic Gamma completion or a host real-equality shortcut.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root gamma real dependency`

Proof. Begin with the root carrier classifier boundary, Definition 548.196. It orders the displayed carrier rows before a critical-line read of M or R is available.

Next use the root zero-strip supported handoff, Definition 548.197. The handoff sends the supplied $\mathsf{ZetaZeros}^{\uparrow}$ and critical-strip rows through R, Q, M, H, C, P, N, so a $\mathsf{Real}^{\uparrow}$ dependency is still a finite carrier dependency.

Now apply the RH-boundary refusal totality theorem, Definition 548.194. It records that completed $\mathsf{Gamma}^{\uparrow}$ data and hidden analytic branches are not exported by the packet.

The remaining dependency is therefore exactly the finite route through $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. Since host

real equality and analytic Gamma completion have no displayed coordinate in W, they cannot replace R, Q, M, H, C, P, N. □

Theorem 548.206 (CriticalLineWitness root refusal ledger exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet anchored in* BHist*. Every root refusal ledger entry for* RH*-facing,* $\mathsf{Gamma}^{\uparrow}$*-facing,* $\mathsf{ZetaContinuationWitness}^{\uparrow}$*-facing, or* $\mathsf{ZetaZeros}^{\uparrow}$*-facing demand is exactly the finite ledger over $Z, S, M, R, Q, H, C, P, N$ with componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. The ledger records refusal of analytic continuation export, host equality, quotient representatives, ambient completeness, global zero classification, zero search, and* RH *proof data; it does not turn any refused item into a proof object.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal ledger exactness`

Proof. First project the root refusal route, Definition 548.78. It sends off-route root demands to the displayed refusal boundary rather than to analytic or host-level data.

Apply RH-refusal ledger completeness, Definition 548.72. This accounts for RH proof requests, global zero classifiers, analytic-continuation certificates, quotient representatives, host equality, and ambient completeness within the same finite ledger.

Then use the root zeta ledger non-escape theorem, Definition 548.198. It keeps $\mathsf{ZetaContinuationWitness}^{\uparrow}$ and $\mathsf{ZetaZeros}^{\uparrow}$ demands attached to the named packet and the sibling continuation boundary rather than exporting a new analytic theorem.

Together these finite carrier projections exhaust the refusal ledger. The exact rows are $Z, S, M, R, Q, H, C, P, N$, and every refused $\mathsf{Gamma}^{\uparrow}$, zeta, or RH object remains absent as an exported proof or host-level coordinate. □

Theorem 548.207 (CriticalLineWitness root strip public target)**.** *The root-strip source totality theorem, the root Gamma Real dependency theorem, and the root refusal ledger exactness theorem together provide a public target surface for formalizing accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packets. The exposed rows are exactly $Z, S, M, R, Q, H, C, P, N$: Z and S record the root-strip source, M, R, Q record the Gamma-facing and real-comparison dependencies, and H, C, P, N record transport, replay, provenance, and local naming. This target surface contains no* RH *proof data, analytic continuation export, global zero classification, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip public target`

Proof. Start with Definition 548.204. It makes the supplied zero row Z, strip row S, depth row M, real row R, rational ledger Q, and structural rows H, C, P, N visible before any root consumer can read the packet.

Next apply Definition 548.205. It keeps every $\mathsf{Gamma}^{\uparrow}$-facing and $\mathsf{Real}^{\uparrow}$-facing dependency inside the displayed M, R, Q, H, C, P, N route, with Z and S still supplying the named strip source.

Finally use Definition 548.206. The refusal ledger accounts for every off-route request as absent from the finite packet, so packaging the three theorems yields a public target surface and not an RH theorem, analytic-continuation theorem, global zero classifier, host-equality principle, quotient representative, or completeness claim. □

Theorem 548.208 (Critical-line witness source modulus classifier lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The supplied zero row Z, critical-strip row S, rational comparison ledger Q, modulus row M,* $\mathsf{Real}^{\uparrow}$ *row R,* $\mathsf{Rat}^{\uparrow}$ *comparison row,* hsame *row H,* Cont *row*

C, Pkg row P, and local $\mathsf{NameCert}$ row N are the only root-facing source and classifier surface of W.

Consequently a root consumer may classify the supplied zero-strip source only by reading those displayed rows. It receives no RH proof object, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness coordinate.

Proof. Begin with the root-source triad of Definition 548.76. It exposes Z, S, Q as the finite source triad carried by H, C, P, N.

Use the root comparison ledger of Definition 548.77. The ledger places every rational comparison through Q, M, R, H, C, P, N and refuses analytic or quotient replacement rows.

Finally apply the zero-strip carrier totality and RH-refusal non-escape theorems. They identify the accepted packet as exactly $Z, S, M, R, Q, H, C, P, N$ and exclude an RH proof object, global zero classifier, analytic-continuation certificate, quotient representative, host equality, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus classifier lock`

Theorem 548.209 (Critical-line witness zero-strip readback totality)**.** *For an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet $W = (Z, S, M, R, Q, H, C, P, N)$, every root readback of the zero-strip side is total through the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison ledger Q, and the structural rows H, C, P, N. No root readback may skip directly to a global zero-set decision, analytic-continuation certificate, host equality, quotient representative, RH proof object, or ambient completeness principle.*

Proof. Project the zero-strip transport lock. By Definition 548.71, the rows Z and S remain attached to the displayed BHist surface through H, C, P, N.

Attach the comparison side through the root comparison ledger. The ledger forces rational and real-part comparison readback through Q, M, R, H, C, P, N rather than through a host zero search or analytic continuation.

The root refusal route then supplies the exclusion. By Definition 548.78, an RH proof object, global classifier, analytic certificate, quotient representative, host equality, and ambient completeness remain outside the root packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZeroStripReadbackTotality`

Theorem 548.210 (Critical-line witness root NameCert scope completion)**.** *The root $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ scope for an accepted packet $W = (Z, S, M, R, Q, H, C, P, N)$ is completed by the supplied zero row, critical-strip row, rational comparison ledger, modulus row, $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ rows, componentwise hsame, displayed Cont, Pkg provenance, and local naming row. These rows are the whole root-facing source, readback, and classifier boundary for downstream RH-boundary consumers.*

The completed scope still does not provide an RH proof object, global zero classifier, analytic-continuation certificate, quotient representative, host equality, hidden zero search, or ambient completeness coordinate.

Proof. First use Definition 548.208. It locks the source and classifier surface to $Z, S, M, R, Q, H, C, P, N$.

Next use Definition 548.209. The zero-strip readback is total through the same displayed packet and cannot skip to an external analytic or host-level row.

Finally apply the existing root-source triad and root comparison ledger. Together they supply the NameCert-facing source and comparison obligations, while the refusal results rule out every

requested proof, classifier, continuation, quotient, host-equality, and completeness object outside the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root namecert scope completion`

Theorem 548.211 (Critical-line witness root budget obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The root budget read of W is exhausted by the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison ledger Q, componentwise* hsame *transport row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N.*

Every downstream read of the root budget factors through the existing root source triad, root comparison ledger, and RH-refusal non-escape rows. It receives no RH *proof data, global zero classifier, quotient representative, analytic-continuation certificate, host equality, or ambient completeness coordinate.*

Proof. Start with the root source triad of Definition 548.76. It displays the supplied zero row, critical-strip row, and rational comparison ledger as the source side of the accepted packet.

Use the root comparison ledger of Definition 548.77. The comparison read is routed through M, R, Q, H, C, P, N, so the root budget cannot branch into a host zero classifier or quotient representative.

Finally apply the RH-refusal non-escape surface, including Definition 548.81. The refused proof, analytic-continuation, host-equality, and ambient-completeness reads are absent from the displayed packet, so the root budget is exhausted by $Z, S, M, R, Q, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root budget obligation`

Theorem 548.212 (Critical-line witness zero-strip budget transport)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the zero-strip budget transported from the supplied zero row Z to the downstream comparison surface is exactly the finite route through S, M, R, Q, H, C, P, N. The transport may cite the root source triad, the root comparison ledger, and the RH-refusal non-escape rows, but it exports no* RH *proof data, global zero classifier, quotient representative, analytic-continuation certificate, host equality, or ambient completeness coordinate.*

Proof. Project the zero-strip source from Definition 548.76. The rows Z and S are displayed before any downstream comparison reads the packet.

Attach the comparison side by Definition 548.77. The modulus, real-part, and rational comparison rows are carried by M, R, Q and stabilized by H, C, P, N.

Use the RH-refusal non-escape rows to close the transport. Since the refusal ledger has no analytic continuation, host equality, quotient representative, global zero classifier, ambient completeness, or RH proof coordinate, the zero-strip budget remains the displayed finite route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip budget transport`

Theorem 548.213 (Critical-line witness root refusal budget exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root refusal budget for downstream* RH*-facing, zero-strip-facing, or comparison-facing reads is exhausted by the same finite carrier rows $Z, S, M, R, Q, H, C, P, N$. The refusal budget records absence of* RH *proof data, global zero classifiers, quotient representatives, analytic-continuation certificates, host equality, and ambient completeness; it does not turn any refused object into a proof row.*

Proof. Begin with the root budget obligation above. It identifies the finite root budget available to downstream consumers and leaves no hidden branch outside $Z, S, M, R, Q, H, C, P, N$.

Transport the zero-strip side by Definition 548.212. The zero row, strip row, modulus row, real-part row, and rational ledger remain one displayed route under H, C, P, N.

Finally apply the RH-refusal non-escape rows, especially Definition 548.81 and Definition 548.94. They record the refused analytic and host-level demands as absent coordinates, so the refusal budget is exhausted by the packet rather than by an external proof surface. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootRefusalBudgetExhaustion`

Theorem 548.214 (CriticalLineWitness zero-strip carrier)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The packet exposes a finite zero-strip source row over* $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, *and* $\mathsf{ZetaZeros}^{\uparrow}$, *with the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *coordinates. This row records the finite source surface; it does not import a proof of* RH *or any analytic continuation principle as carrier data.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip carrier`

Proof. Project the accepted carrier to its displayed rows $Z, S, M, R, Q, H, C, P, N$. The zero-strip source row is read through Z and S, while the numerical and comparison rows are read through M, R, Q.

The surrounding root source and strip-source obligation rows already bind those coordinates to the local BHist packet. Transport through H, continuation through C, provenance through P, and the local naming row N keep the carrier inside the finite $\mathsf{CriticalLineWitness}^{\uparrow}$ surface.

The refusal boundary is part of the same packet. It prevents RH proof data, analytic continuation principles, quotient representatives, host equality, and ambient completeness from becoming carrier fields. □

Theorem 548.215 (CriticalLineWitness zeta comparison ledger)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zeta-comparison row is a ledger entry read through the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *coordinates of W. The comparison ledger may mention* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *and* $\mathsf{ZetaZeros}^{\uparrow}$ *as named source rows, but it does not compare host complex values, choose zeros classically, or export a global zero classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta comparison ledger`

Proof. Begin with the displayed source coordinates. The rows Z and S identify the zeta and strip side, while M, R, Q provide only finite depth, real-part, and rational comparison rows.

The ledger is read through H, C, P, N. Componentwise hsame transport aligns the source rows, the displayed Cont route records the admissible readback, and Pkg with $\mathsf{NameCert}$ authenticates the local packet surface.

No host complex equality or classical zero selection is a coordinate of the accepted packet. The ledger therefore remains a finite displayed comparison row rather than an analytic zero-selection mechanism. □

Theorem 548.216 (CriticalLineWitness NameCert obligation package)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the carrier row, zero-strip source row, zeta-comparison ledger, refusal boundary, and downstream readback rows form the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *obligation surface. Every downstream root, strip, zeta, or* RH*-facing read must pass through that surface before it can consume any displayed packet coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert obligation package`

Proof. The carrier row supplies the finite packet coordinates $Z, S, M, R, Q, H, C, P, N$. The zero-strip source row and zeta-comparison ledger identify the source and comparison portions of the same packet.

The refusal boundary records what the packet does not export: RH proof data, analytic continuation as data, host equality, quotient representatives, global zero selection, and ambient completeness. These exclusions keep every off-route demand at the local ledger boundary.

Finally the downstream readback rows replay the same BHist, hsame, Cont, Pkg, and NameCert coordinates. Hence the local naming certificate surface is exactly the package assembled from the carrier, source, comparison, refusal, and readback rows. □

Theorem 548.217 (CriticalLineWitness fixed tuple image row)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Its* StandardBridge*-facing image is the fixed finite tuple of displayed* BHist *rows*

$$(Z, S, M, R, Q, H, C, P, N).$$

Every accepted carrier read of W lands in exactly one of these nine rows: the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational-comparison row Q, componentwise hsame *transport row H, displayed* Cont *row C,* Pkg *provenance row P, or local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The image row contains no* RH *proof, no host zero classifier, no analytic-continuation theorem, no quotient representative, and no ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed tuple image row`

Proof. Project the accepted carrier through Definition 548.1. The carrier is already the finite BHist packet $Z, S, M, R, Q, H, C, P, N$, so the image visible to a bridge-facing consumer has one displayed row for each coordinate and no further coordinate to inspect.

Use the carrier-admission and ledger-exactness clauses of Definition 548.2. They identify accepted public reads with the displayed packet rows and prevent an accepted read from leaving the packet.

Finally apply Definition 548.201. The root-facing source is a finite refusal surface over the same nine rows, so the image row remains a tuple of BHist data and does not become a bridge theorem or an RH proof. □

Theorem 548.218 (CriticalLineWitness fixed tuple readback row)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet whose image row is the displayed tuple of Definition 548.217. Each displayed image row reads back to the corresponding tuple coordinate through the local structural route: Z, S, M, R, Q are recovered through their componentwise* hsame *entries in H, their displayed* Cont *routes in C, their* Pkg *provenance in P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N; the structural rows H, C, P, N read back as the transport, continuation, provenance, and naming coordinates of the same carrier.*

Proof. Begin with the root zero-strip readback theorem Definition 548.202. It exhausts every source-side readback by the supplied zero row Z, strip row S, modulus or depth row M, real-part row R, rational-comparison row Q, and the structural route H, C, P, N.

For the data rows Z, S, M, R, Q, use the zero-strip compatibility, modulus-discipline, and ledger-exactness clauses of Definition 548.2. These clauses say that the rows are transported by H, replayed by C, recorded by P, and named by N before any public consumer can read them.

For the structural rows, read the same carrier projection directly. The rows H, C, P, N are not extra witnesses outside the image; they are the tuple coordinates that supply componentwise hsame, Cont, Pkg, and NameCert structure for the readback itself. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed tuple readback row`

Theorem 548.219 (CriticalLineWitness StdBridge premise surface)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the fixed-route* StandardBridge *premise surface is exactly the following finite collection:*

1. *the tuple carrier* $(Z, S, M, R, Q, H, C, P, N)$ *from Definition 548.1;*
2. *the image row of Definition 548.217;*
3. *the readback row of Definition 548.218;*
4. *componentwise* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *transport through the displayed* H, C, P, N *rows;*
5. *the* ChapterTasteGate *row required by the fixed finite* BHist *tuple surface.*

These are the premises for the fixed-route StandardBridge *surface and nothing stronger: they do not prove* RH*, construct a bridge theorem, solve recognizer-to-carrier transport, or add analytic, quotient, host-equality, or completeness data.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed tuple stdbridge premise surface`

Proof. Assemble the tuple carrier from Definition 548.1. The carrier supplies the nine displayed BHist rows and the local naming coordinate that all downstream reads must use.

Add the image and readback rows. The image theorem confines every accepted carrier read to the finite tuple, while the readback theorem returns each displayed row to its corresponding coordinate through hsame, Cont, Pkg, and NameCert structure.

Transport the naming obligations component by component using Definition 548.2. The root refusal and readback controls of Definition 548.201 and Definition 548.202 keep the route on the same finite packet.

Finally compare the assembled rows with Definition 9464.24 and Definition 9464.25. Those results name the tuple carrier, image row, readback row, componentwise NameCert transport, and ChapterTasteGate row as the exact fixed finite BHist premise surface before any StandardBridge request. The present theorem only identifies that premise surface for $\mathsf{CriticalLineWitness}^{\uparrow}$. □

Theorem 548.220 (CriticalLineWitness supplied source carrier triad exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The supplied* Z, S, Q *source triad is exhausted by the displayed carrier rows* $Z, S, M, R, Q, H, C, P, N$*:* Z *supplies the zero-facing source,* S *supplies the critical-line strip,* Q *supplies the source-facing comparison ledger, and the remaining rows* M, R, H, C, P, N *only route, transport, replay, package, and name that same finite carrier.*

No root read of the supplied source triad exports an RH *proof object, a global zero search, analytic continuation, quotient representatives, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier supplied source carrier triad exhaustion`

Proof. Start with Definition 548.76. It identifies Z, S, Q as the source triad exposed by the accepted root packet, and it reads that triad through the finite BHist, hsame, Cont, Pkg, and local NameCert surface.

Project the carrier coordinates of W. The only available rows are $Z, S, M, R, Q, H, C, P, N$, so the triad cannot acquire a second zero source, detached strip witness, hidden comparison classifier, or ambient analytic row.

Finally route the unused carrier rows as support rows. The modulus, real-part, transport, continuation, provenance, and naming coordinates preserve the same finite packet rather than adding a proof of RH or any external completion datum. □

Theorem 548.221 (CriticalLineWitness modulus comparison NameCert route)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every modulus comparison read factors through* M, R, Q, H, C, P, N *and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The route exposes the modulus row* M*, real-part row* R*, rational comparison row* Q*, componentwise transport row* H*, displayed continuation row* C*, package provenance row* P*, and local naming row* N *before any comparison-facing consumer can read the packet.*

The route does not supply a host analytic estimate, quotient representative, global zero classifier, analytic-continuation certificate, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus comparison namecert route`

Proof. Apply Definition 548.77. It places comparison reads on the finite ledger containing M, R, Q, H, C, P, N and refuses replacement by global zero, quotient, analytic, or completeness data.

Next apply Definition 548.70. The modulus row and rational comparison row are consumed under the same transport, continuation, provenance, and local naming rows, so the $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ coordinate belongs to the comparison route itself.

The carrier projection supplies no further comparison coordinate. Thus every accepted modulus comparison read factors through exactly the displayed NameCert route. □

Theorem 548.222 (CriticalLineWitness RH refusal public nonescape)**.** *Every public* RH*-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet exposes only the displayed finite refusal route through* $Z, S, M, R, Q, H, C, P, N$*. It cannot export an* RH *proof, zero search, analytic continuation, quotient representative, host equality, ambient completeness, or any detached classifier outside the packet.*

The public route is therefore a finite refusal surface over the supplied source rows, modulus comparison rows, transport rows, continuation rows, provenance row, and local NameCert *row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal public nonescape`

Proof. Use Definition 548.78 to locate every public refusal read on the root packet. The refusal is carried by the displayed finite rows rather than by an external analytic certificate or host-level equality.

Then apply Definition 548.75. This excludes RH proof data, zero search, analytic continuation, quotient representatives, host equality, and ambient completeness from the accepted packet surface.

Combining the two routes leaves only $Z, S, M, R, Q, H, C, P, N$. The public read can refuse off-route demands, but it has no coordinate from which to export the forbidden data. □

Theorem 548.223 (CriticalLineWitness finite observation route)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every downstream finite-observation read of* W *must expose the finite zero-strip row* Z*, critical-strip row* S*, rational-height and comparison rows* R, Q*,*

zeta-continuation dependency surface, and root-comparison rows before any downstream consumer reads the packet. No such read can bypass those rows through a host zero classifier, global analytic continuation object, quotient representative, ambient completeness witness, or detached RH *truth value.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finite observation route`

Proof. Begin with the root source triad of Definition 548.76. It exposes the supplied zero row, critical-strip row, and source-facing comparison ledger as finite rows before downstream access.

Use the zero-strip source completion rows, including Definition 548.180, to keep the zero-strip and critical-line sources on the displayed finite route. Use the root comparison ledger of Definition 548.77 to attach the rational-height and comparison reads through R, Q, H, C, P, N.

Finally apply the zeta dependency lattice of Definition 548.169. It places the zeta-zero and continuation-facing dependency surfaces before the modulus witness is public, so every downstream finite-observation read factors through the displayed packet rather than an analytic or host-level replacement. □

Theorem 548.224 (CriticalLineWitness StdBridge candidate refusal)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, a proposed standard bridge candidate that lacks the zero-strip source rows, modulus rows, or finite readback rows is a failed premise for the bridge route, not a counterinstance to the* $\mathsf{CriticalLineWitness}^{\uparrow}$ *witness surface. The packet refuses the missing zero-strip, modulus, and readback demands inside its displayed finite ledger.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier stdbridge candidate refusal`

Proof. Read the candidate first through the root refusal route, Definition 548.78. That route sends off-surface demands to the displayed finite refusal boundary rather than treating them as alternate bridge data.

Next apply the zero-strip source completion and downstream readback exhaustion rows, including Definition 548.180 and Definition 548.154. Missing zero-strip or readback data therefore means that the candidate has not supplied the premises required by the accepted packet route.

The root comparison ledger of Definition 548.77 supplies the same conclusion for modulus and rational comparison reads. Since the candidate lacks required displayed rows, it fails before any standard bridge surface is reached and cannot witness a bridge counterinstance. □

Theorem 548.225 (CriticalLineWitness dependency ledger exhaustion)**.** *Every downstream* $\mathsf{CriticalLineWitness}^{\uparrow}$ *read factors through the displayed dependency rows supplied by* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{Gamma}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ZetaContinuationWitness}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$*. The accepted packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *exhausts these dependencies through its finite zero-strip, modulus, rational-height, root-comparison, transport, continuation, package, and local* NameCert *rows, with no additional analytic, host-equality, quotient, or ambient-completeness route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier dependency ledger exhaustion`

Proof. Start from the root source triad and the root comparison ledger, Definition 548.76 and Definition 548.77. They expose the $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows through the displayed packet coordinates.

Use the zero-strip source completion rows for the $\mathsf{ZetaZeros}^{\uparrow}$ side and the zeta dependency lattice of Definition 548.169 for the $\mathsf{ZetaContinuationWitness}^{\uparrow}$ side. These rows force zero-strip and

continuation-facing reads to consume sibling dependency surfaces before the local modulus witness is public.

Finally combine the root refusal route with the downstream readback exhaustion theorem, Definition 548.78 and Definition 548.154. The remaining $\mathsf{Gamma}^{\uparrow}$ and readback dependencies stay inside the finite ledger, and every forbidden analytic, host-equality, quotient, or ambient-completeness demand is refused rather than exported. □

Theorem 548.226 (CriticalLineWitness root bridge readback totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root bridge readback of W is exhausted by the displayed finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The readback first consumes the root-source rows Z, S, Q, then the comparison-ledger rows M, R, Q, and finally the fixed-tuple transport, replay, provenance, and local naming rows H, C, P, N. It exports no RH *proof, host zero classifier, analytic-continuation theorem, quotient representative, host equality, or ambient completeness coordinate.*

Proof. Begin with the root source triad Definition 548.76. It places the supplied zero row, critical-strip row, and source-facing comparison row inside the same finite packet before bridge readback begins.

Next use the root comparison ledger Definition 548.77. The modulus, real-part, and rational-comparison reads are forced through M, R, Q, H, C, P, N rather than through an analytic host object.

Finally apply the fixed tuple image and readback rows, Definition 548.217 and Definition 548.218. These rows return each bridge-visible coordinate to the corresponding carrier coordinate, while the root refusal route removes the forbidden analytic and host-level exits. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root bridge readback totality`

Theorem 548.227 (CriticalLineWitness root NameCert consumer totality)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every root-facing* NameCert *consumer is total on the same finite packet. The consumer may read carrier admission, source exposure, comparison-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; it has no route to a second source packet, external comparison ledger, analytic oracle, quotient representative, host equality, or ambient completeness witness.*

Proof. Project the carrier with Definition 548.1. The local packet exposes exactly $Z, S, M, R, Q, H, C, P, N$, so a root-facing NameCert consumer has a finite coordinate for every accepted read.

Apply Definition 548.2. Carrier admission, source compatibility, modulus discipline, ledger exactness, and non-escape are all obligations of the same local naming row N.

The root-source triad and root comparison ledger then account for the source and comparison sides. The fixed-tuple premise surface repacks those rows as the bridge-readable packet, so the consumer is total exactly on the displayed route and nowhere else. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootNameCertConsumerTotality`

Theorem 548.228 (CriticalLineWitness root analytic-refusal exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Any root-facing request for an* RH *proof, zero search, global zero-set classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness row is exhausted by the root refusal route of W. The request is recorded*

as a refusal over $Z, S, M, R, Q, H, C, P, N$ and is not a bridge readback, NameCert consumer, or analytic witness outside the packet.

Proof. Start from Definition 548.78. It sends off-surface root demands to the displayed finite refusal boundary instead of to host analytic data.

Use the root-source triad and root comparison ledger to locate the positive rows that remain readable after refusal. They identify the supplied source and comparison data with $Z, S, M, R, Q, H, C, P, N$.

Finally use the fixed-tuple premise surface. Since every bridge-facing readback must return to the fixed tuple and every local NameCert consumer reads the same tuple, the analytic demand is fully exhausted by refusal rather than by a hidden analytic coordinate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootAnalyticRefusalExhaustion`

Theorem 548.229 (CriticalLineWitness source-route totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before any critical-line consumer reads W, the packet exposes its supplied* BHist *source row Z, fixed zero-strip row S, rational modulus row M, zeta-continuation witness row, comparison ledger rows R, Q, componentwise* hsame *transport row H, displayed* Cont *route C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The consumer therefore enters the packet only through the finite source route*

$$Z, S, M, R, Q, H, C, P, N.$$

Proof. Start with the root source triad of Definition 548.76. It exposes the supplied zero row, fixed critical-strip row, and source-facing comparison row as finite BHist data before a critical-line consumer can inspect the packet.

Next apply the root comparison ledger Definition 548.77 and the root modulus readback theorem Definition 548.129. These rows attach the rational modulus, real-part, and rational-comparison coordinates to M, R, Q, H, C, P, N rather than to a host analytic object.

Finally use the NameCert obligations of Definition 548.2. Carrier admission, transport, displayed continuation, provenance, and local naming are all read on the same accepted tuple, so no critical-line consumer can read W before the displayed source route has been exposed. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source route totality`

Theorem 548.230 (CriticalLineWitness zeta-boundary non-escape)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zeta-facing read of W remains inside the finite* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *ledger carried by the supplied source row and the displayed route $Z, S, M, R, Q, H, C, P, N$. Such a read cannot import an* RH *predicate, a global zero classifier, a completed* $\mathsf{Gamma}^{\uparrow}$ *certificate, or a selected non-trivial-zero locator.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta boundary nonescape`

Proof. Read the supplied source through Definition 548.229. The critical-line witness has already exposed the zeta-continuation witness row and the structural rows H, C, P, N before any zeta-facing consumer can use it.

Now apply the ZetaContinuationWitness source obligation and analytic readback obligation, Definition 580.15 and Definition 580.16. They keep zeta-facing reads inside the displayed continuation ledger and its pole, functional-equation, Gamma-boundary, transport, continuation, provenance, and local naming rows.

Close with the ZetaContinuationWitness zero ledger non-escape theorem Definition 580.17. It refuses RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma

certificates, so the CriticalLineWitness read cannot gain those objects by passing through the supplied zeta ledger. □

Theorem 548.231 (CriticalLineWitness NameCert obligation triad)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the source-route totality of Definition 548.229, the zeta-boundary non-escape of Definition 548.230, and the modulus readback of Definition 548.129 form the local root* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *obligation triad. Together they expose the finite* BHist *source, rational modulus, zeta-continuation, zero-strip,* hsame*,* Cont*,* Pkg*, and* NameCert *rows while refusing every off-packet analytic, zero-classification, and host-equality exit.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert obligation triad`

Proof. The source-route totality theorem supplies the carrier side of the triad. It projects the accepted packet to the displayed finite rows $Z, S, M, R, Q, H, C, P, N$ before a critical-line consumer can read the witness.

The zeta-boundary non-escape theorem supplies the analytic boundary side. It keeps zeta-facing reads inside the finite $\mathsf{ZetaContinuationWitness}^{\uparrow}$ ledger and refuses RH, zero-classifier, Gamma-completion, and non-trivial-zero selection exits.

The modulus readback theorem supplies the numeric side. It routes rational modulus, real-part, and rational comparison reads through M, R, Q, H, C, P, N and not through a host numeric oracle or quotient representative.

Combining these three coordinates gives exactly the local root $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ surface: source totality, boundary non-escape, and modulus readback are all obligations over the same displayed packet and no further root obligation row is introduced. □

Theorem 548.232 (CriticalLineWitness strip-modulus source exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, modulus row* M*, real-part row* R*, rational comparison ledger* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N *exhaust the root source-modulus surface. A consumer of this surface receives no* RH *proof, zero search, quotient representative, host equality, analytic-continuation certificate, or ambient completeness row.*

Proof. Start with the carrier of Definition 548.1 and the NameCert obligations of Definition 548.2. They expose the accepted packet as the finite tuple $Z, S, M, R, Q, H, C, P, N$ and bind all public reads to the local naming row.

Use the root source triad and source-route totality rows, Definition 548.76 and Definition 548.229. These rows put the $\mathsf{ZetaZeros}^{\uparrow}$ source, $\mathsf{CritStrip}^{\uparrow}$ source, BHist carrier, hsame transport, Cont replay, Pkg provenance, and NameCert surface on the same finite route.

Finally apply the root comparison ledger and root refusal route, Definition 548.77 and Definition 548.78. The modulus and comparison reads stay inside M, R, Q, H, C, P, N, while every RH-facing or analytic request is refused rather than exported. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier strip modulus source exhaustion`

Theorem 548.233 (CriticalLineWitness zeta-route carrier separation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the zeta route carried by* Z *and the strip-modulus route carried by* S, M, R, Q *remain separate carrier rows until the displayed* hsame*,* Cont*,* Pkg*, and*

NameCert *rows* H, C, P, N *relate them. The packet does not identify the* ZetaZeros$^{\uparrow}$ *source with the* CritStrip$^{\uparrow}$, Real$^{\uparrow}$, *or* Rat$^{\uparrow}$ *comparison surface by host equality, quotient representatives, analytic continuation, zero search,* RH *proof data, or ambient completeness.*

Proof. Project the source rows first. The root source triad of Definition 548.76 exposes Z, S, Q as finite source data before any downstream comparison is read.

Next route the modulus and comparison rows through Definition 548.77 and Definition 548.129. These rows attach M, R, Q to the Real$^{\uparrow}$ and Rat$^{\uparrow}$ comparison surfaces only through the displayed packet route.

It remains to account for the separation boundary. The NameCert obligations and fixed-tuple readback rows, Definition 548.2 and Definition 548.218, permit comparison only through H, C, P, N. Thus no host equality, quotient representative, analytic-continuation row, zero search, RH proof, or ambient completeness principle identifies the carrier rows outside the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta route carrier separation`

Theorem 548.234 (CriticalLineWitness RH-refusal ledger closure)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The RH-refusal ledger of* W *closes the root-unblock surface over the displayed* ZetaZeros$^{\uparrow}$, CritStrip$^{\uparrow}$, Real$^{\uparrow}$, Rat$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. Every request for an* RH *proof, zero search, quotient representative, host equality, analytic-continuation certificate, or ambient completeness witness is recorded as a refusal on the same packet and does not become a second carrier, oracle, or downstream proof row.*

Proof. Begin with the RH-refusal ledger completeness theorem Definition 548.72. It already states that downstream RH-facing access is exhausted by $Z, S, M, R, Q, H, C, P, N$ together with the visible BEDC structural rows.

Use strip-modulus source exhaustion from Definition 548.232. This supplies the root source-modulus surface over the zeta, strip, real, rational, history, transport, replay, provenance, and naming rows.

Finally combine the zeta-route carrier separation of Definition 548.233 with the root analytic-refusal exhaustion of Definition 548.228. The former prevents carrier collapse across source and comparison rows; the latter sends every analytic or RH-facing request to the displayed refusal route. Therefore the root-unblock ledger is closed by refusal on the accepted packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal ledger closure`

Theorem 548.235 (CriticalLineWitness modulus-strip source totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Before any downstream* RH*-refusal consumer reads* W*, the root read exposes the fixed critical-line source row* Z*, zero-strip row* S*, rational modulus and budget rows* M, R, Q*, componentwise* hsame *transport row* H*, displayed* Cont *route* C*,* Pkg *provenance row* P*, and local* NameCert$_{\text{CriticalLineWitness}^{\uparrow}}$ *row* N*. Thus every such consumer enters the witness through the finite root surface*

$$Z, S, M, R, Q, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus strip source totality`

Proof. Start with the source-route totality surface Definition 548.229. It places the supplied BHist source row, fixed strip row, rational modulus row, transport, route, provenance, and local naming row before any critical-line consumer can inspect the packet.

Next use the root bridge readback totality theorem Definition 548.226. The bridge readback consumes the same root-source, comparison-ledger, fixed-tuple, transport, replay, provenance, and naming coordinates, so the consumer has no second entry point outside $Z, S, M, R, Q, H, C, P, N$.

Finally apply the zero-strip NameCert completion surface Definition 548.216. The carrier, zero-strip source, zeta-comparison ledger, refusal boundary, and downstream readback rows are all obligations of the same local naming packet, giving the announced finite root surface before an RH-refusal read begins. □

Theorem 548.236 (CriticalLineWitness finite budget readback determinacy)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ and $W' = (Z', S', M', R', Q', H', C', P', N')$ be accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packets. If the displayed modulus rows, zero-strip rows, rational budget rows, componentwise* hsame *transport rows,* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows agree, then W and W' present the same finite root readback surface to every downstream* RH*-refusal consumer. In particular the consumer cannot distinguish the two reads by asking for a different source, modulus, strip, budget, transport, route, provenance, or local naming row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finite budget readback determinacy`

Proof. Read both packets through Definition 548.229. Each accepted packet is exposed only as its finite route $Z, S, M, R, Q, H, C, P, N$ or $Z', S', M', R', Q', H', C', P', N'$ before a downstream consumer reads it.

The assumed agreement identifies the displayed modulus, strip, rational budget, transport, route, provenance, and naming coordinates. The root bridge readback theorem Definition 548.226 then leaves no bridge-visible coordinate outside those displayed rows.

It remains to compare the refusal-facing surface. The zero-strip NameCert completion theorem Definition 548.216 binds the carrier, zero-strip source, comparison ledger, refusal boundary, and downstream readback rows to the same local naming packet. Hence the two accepted packets determine the same finite readback surface for every allowed consumer. □

Theorem 548.237 (CriticalLineWitness root consumer non-escape)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, no downstream root consumer that reads only the displayed finite rows $Z, S, M, R, Q, H, C, P, N$ extracts a completed* RH *proof, an analytic continuation theorem, a host zero oracle, a quotient representative, host equality, ambient completeness, or an unbounded critical-line search. Every positive read remains inside the* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, finite modulus, and zero-strip rows of W.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root consumer nonescape`

Proof. Begin with Definition 548.226. It states that root bridge readback is exhausted by $Z, S, M, R, Q, H, C, P, N$ and exports no RH proof, host zero classifier, analytic-continuation theorem, quotient representative, host equality, or ambient completeness coordinate.

The source-route totality theorem Definition 548.229 identifies the positive rows that remain readable: the supplied source, fixed strip, rational modulus, zeta-continuation witness, comparison ledger, transport, route, provenance, and local naming rows.

Finally use Definition 548.216. Its refusal boundary is part of the same local NameCert surface, while its carrier, source, comparison, and downstream readback rows account for every allowed finite read. Thus a root consumer stays inside the displayed packet and cannot turn those rows into an unbounded search or an off-packet analytic witness. □

Definition 548.238 (CriticalLineWitness root source window)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root read, the root source window is the finite* BHist *packet*

$$W_{\mathrm{src}} = (Z, S, M, H, C, P, N).$$

Here Z is the zero-strip source row, S is the critical-line strip row, M is the finite modulus and readback row, H records the componentwise hsame *transports, C records the displayed* Cont *routes, P records* Pkg *provenance over* BHist*,* hsame*,* Cont*, and* NameCert*, and N is the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. A read is accepted by the window only when every consumer-facing coordinate factors through these seven rows before it reaches the full packet of Definition 548.1.*

Lean4 (definition) `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessWindowSource`

Theorem 548.239 (CriticalLineWitness zero-strip source separation)**.** *Every accepted root source window $W_{\mathrm{src}} = (Z, S, M, H, C, P, N)$ separates zero-strip source reads from* RH *predicates, non-trivial-zero locators, global zero classifiers, and completed* $\mathsf{Gamma}^{\uparrow}$ *certificates. The only positive source reads exposed by the window are the finite* BHist *rows Z, S, M, their componentwise* hsame *transports H, their displayed* Cont *routes C, their* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip source separation`

Proof. Project the accepted read through Definition 548.238. The projection exposes only the seven displayed rows Z, S, M, H, C, P, N, so the zero-strip source side is already a finite BHist packet before any downstream request is considered.

Next apply the source-route and zero-strip surfaces already present in Definition 548.229 and Definition 548.180. These rows force the supplied source, critical strip, modulus, transport, route, provenance, and local naming coordinates to be read together.

Finally use the refusal and readback boundaries Definition 548.189 and Definition 548.194. They remove the RH predicate, non-trivial-zero locator, global zero classifier, and completed $\mathsf{Gamma}^{\uparrow}$ certificate from the root window, leaving only the finite source packet claimed above. □

Theorem 548.240 (CriticalLineWitness continuation route coverage)**.** *Every downstream critical-strip consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root source window must read the zero-strip source row Z, the critical-line strip row S, the finite modulus row M, the componentwise* hsame *transport row H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N before consuming the witness. No downstream critical-strip read has a coordinate outside*

$$Z, S, M, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier continuation route coverage`

Proof. Start with coordinate exhaustion for the root source window. By Definition 548.238, an accepted window contains only Z, S, M, H, C, P, N, and any consumer-facing coordinate must factor through one of those rows before reaching the full carrier.

Read the critical-strip side through Definition 548.235. It places the source row, fixed strip row, rational modulus data, transport, route, provenance, and naming row before an RH-refusal consumer can inspect the witness.

It remains to account for route locality. The Cont component is displayed as C and is tied to the same packet by the source-route totality and local NameCert obligations. Hence a downstream consumer cannot bypass H, C, P, N or replace them by an ambient continuation theorem; every route into the witness is covered by the finite window rows. □

Theorem 548.241 (CriticalLineWitness root public readiness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every downstream public read of W factors through the displayed finite route*

$$Z, S, M, R, Q, H, C, P, N$$

It also factors through the local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The route therefore exposes the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row before any public consumer can read the witness.*

Proof. Project the accepted carrier with Definition 548.1. The carrier is exactly the finite tuple $Z, S, M, R, Q, H, C, P, N$, so a public read has no carrier coordinate outside the displayed packet.

Use the NameCert obligations of Definition 548.2. Carrier admission, zero-strip compatibility, modulus discipline, ledger exactness, and non-escape are all attached to the same local naming row.

Finally apply the obligation closure package Definition 548.22 together with the downstream source obligation Definition 548.48. These rows repack every downstream source read through the finite route and the local NameCert row. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root public readiness`

Theorem 548.242 (CriticalLineWitness root scope binding)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the public-read route depends only on the chapter's displayed* BHist *carrier, the local* hsame *transport row, the displayed* Cont *replay row, the* Pkg *provenance row, the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row, and the already cited* $\mathsf{ZetaZeros}^{\uparrow}$ *and* $\mathsf{CritStrip}^{\uparrow}$ *source rows. No further source object is needed to bind the root public-read route.*

Proof. Begin with Definition 548.1. It lists the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row Z and $\mathsf{CritStrip}^{\uparrow}$ row S together with M, R, Q, H, C, P, N inside one BHist packet.

Apply Definition 548.2. The obligations bind classifier transport through H, continuation replay through C, package provenance through P, and local naming through N on that same packet.

Use Definition 548.22 to close the scope. The package has no second carrier, no external classifier, and no additional source row; hence the public-read route is scoped exactly by the listed BHist, hsame, Cont, Pkg, NameCert, $\mathsf{ZetaZeros}^{\uparrow}$, and $\mathsf{CritStrip}^{\uparrow}$ rows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root scope binding certificate`

Theorem 548.243 (CriticalLineWitness root ledger refusal totality)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, requests for* RH *proof data, global zero search, host equality, quotient representatives, analytic continuation, or ambient completeness have no ledger coordinate in the downstream public-read route through $Z, S, M, R, Q, H, C, P, N$ and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row.*

Proof. Start with the non-escape clauses in Definition 548.2. They make the local ledger a finite packet ledger rather than a route to host analytic data.

Apply the ledger non-escape obligation Definition 548.52. It exhausts the local ledger through M, Q, H, C, P, N together with Z, S, R and refuses RH proof data, zero search, analytic-continuation data, quotient representatives, host equality, and ambient completeness.

Finally use Definition 548.48. Every downstream public source read has already been factored through the same finite packet, so the refused objects cannot re-enter as hidden coordinates of the public-read route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root ledger refusal totality`

Theorem 548.244 (Critical-line witness root-line row exposure)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted root critical-line read of W first exposes the supplied zero row Z, the critical-strip row S, the modulus row M, the rational comparison row R, the comparison ledger Q, the componentwise transport row H, the displayed continuation row C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root line row exposure`

Proof. Project the accepted carrier using Definition 548.1. The packet has exactly the displayed finite rows $Z, S, M, R, Q, H, C, P, N$, so a root critical-line read has no earlier carrier coordinate to inspect.

Apply the root public-read package of Definition 548.241 and the zero-strip consumer boundary of Definition 548.143. These rows place the zero row, strip row, modulus row, comparison rows, transport row, continuation row, provenance row, and local naming row in the same finite readback route.

The read therefore begins with the supplied BEDC rows rather than with an ambient analytic object. Any later consumer receives the packet through this exposure order. □

Theorem 548.245 (Critical-line witness zero-strip route factorization)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every zero-strip consumer route through W factors through*

$$Z, S, H, C, P, N$$

before any modulus row or RH*-facing row is readable.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip route factorization`

Proof. Read the zero-strip side first. The supplied zero row Z and critical-strip row S are bound to the same componentwise transport, continuation replay, package provenance, and local naming rows by Definition 548.71.

Next use Definition 548.142. It places modulus access downstream of the displayed strip route, so M, R, Q are not available before the route has passed through Z, S, H, C, P, N.

Finally apply Definition 548.84. The same finite route separates zero-strip consumption from every RH-facing request. □

Theorem 548.246 (Critical-line witness analytic-continuation nonescape)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. No accepted packet exports analytic continuation,* RH *proof data, global zero classification, a quotient representative, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessAnalyticContinuationNonescape`

Proof. Start from the refusal ledger route of Definition 548.81. It exhausts every RH-facing request through the finite packet rows and reports no host analytic theorem.

Use the downstream refusal package of Definition 548.34 together with Definition 548.72. These rows exclude analytic-continuation data, RH proof data, global zero classification, quotient representatives, host equality, and ambient completeness from downstream reads.

The root-line row exposure above gives the only accepted readback surface. Hence none of the excluded objects can be exported as a hidden coordinate of the accepted packet. □

Theorem 548.247 (Critical-line witness root-route exposure)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet whose root-line surface is exposed by Definition 548.244. Then every root-facing consumer of W must enter through the displayed* BHist *rows, the componentwise* hsame *transport row H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N before it reads any critical-line conclusion.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier p11961 root route exposure`

Proof. Project the accepted carrier and apply Definition 548.244. The rows $Z, S, M, R, Q, H, C, P, N$ are the whole exposed packet, with H, C, P, N carrying transport, route replay, provenance, and local naming for the same finite object.

Next use Definition 548.245. It forces the zero-strip side of the consumer through the displayed route before any later modulus or root-facing row is read.

Thus a root-facing consumer reaches the critical-line surface only through the packet route already certified by BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 548.248 (Critical-line witness modulus ledger separation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The modulus row M and comparison ledger Q are consumed only after the supplied zero row Z, critical-strip row S, transport row H, continuation row C, provenance row P, and local* NameCert *row N have exposed the zero-strip route of Definition 548.245.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier p11961 modulus ledger separation`

Proof. Start with the zero-strip factorization theorem. It places every zero-strip consumer on the route through Z, S, H, C, P, N before the modulus row or any RH-facing row is readable.

The rows M, R, Q are therefore ledger data downstream from the displayed carrier route, not detached analytic coordinates. The componentwise hsame row and displayed Cont row preserve the same packet while P and N record provenance and local naming.

Combining this with root-line row exposure gives the stated separation between the root route and the modulus-comparison ledger. □

Theorem 548.249 (Critical-line witness RH refusal boundary)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Any consumer request for analytic continuation, an* RH *proof, global zero classification, quotient representatives, host equality, or ambient completeness is refused at the displayed boundary formed by the root-line row exposure, zero-strip route factorization, and analytic-continuation nonescape rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier p11961 rh refusal boundary`

Proof. Apply Definition 548.246. It states that the accepted packet exports none of the analytic, quotient, host-equality, or ambient-completeness objects named in the statement.

The root-line row exposure and zero-strip route factorization identify the only route a consumer may use: the finite carrier through $Z, S, M, R, Q, H, C, P, N$, with BHist data transported by hsame, replayed by Cont, recorded by Pkg, and named by the local NameCert row.

Every request outside that finite route is therefore a refusal request rather than an exported witness of the packet. □

Theorem 548.250 (Critical-line witness root-strip source totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root* $\mathsf{CriticalLineWitness}^{\uparrow}$ *consumer must expose the finite source rows for* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{Gamma}^{\uparrow}$*,* $\mathsf{ZetaContinuationWitness}^{\uparrow}$*,* $\mathsf{ZetaZeros}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *through* $Z, S, M, R, Q, H, C, P, N$ *before any critical-line-facing read is available. The consumer therefore enters the packet only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootStripSourceTotality`

Proof. Project the accepted packet through the carrier description of Definition 548.1. Its visible coordinates are the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison ledger Q, transport row H, continuation route C, provenance row P, and local naming row N.

Apply the source-route totality of Definition 548.229 and the root-line row exposure of Definition 548.244. These rows place the $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources on the finite route before a line-facing consumer can read the modulus or comparison side.

Finally use the zeta-boundary non-escape theorem Definition 548.230. The $\mathsf{ZetaContinuationWitness}^{\uparrow}$ and $\mathsf{Gamma}^{\uparrow}$ interfaces remain inside the displayed zeta ledger and its transport, continuation, provenance, and naming rows. Hence the root consumer receives the listed finite source rows before any critical-line-facing read begins. □

Theorem 548.251 (Critical-line witness root-strip Zeta-Gamma source lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. A consumer cannot expose the root-strip row of* W *until the* $\mathsf{ZetaZeros}^{\uparrow}$*,* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{ZetaContinuationWitness}^{\uparrow}$*,* $\mathsf{Gamma}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source rows have all been exposed through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip zeta gamma source lock`

Proof. Start with the root-strip source totality theorem Definition 548.250. It projects an accepted packet to the finite rows $Z, S, M, R, Q, H, C, P, N$ and places $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaContinuationWitness}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the route before any critical-line-facing read.

Next apply the modulus refusal exhaustion theorem Definition 548.252. A zero-strip modulus request remains inside the same displayed route and cannot supply analytic continuation evidence, a zero-free-strip theorem, a global zero classifier, or a host numeric oracle.

Finally use the RH-boundary nonescape theorem Definition 548.253. The packet exports no RH proof, finished critical-line theorem, or host analytic primitive. Therefore the root-strip row is source-locked to the listed Zeta, Gamma, Nat, Rat, Real, transport, replay, provenance, and local naming rows. □

Theorem 548.252 (Critical-line witness modulus refusal exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every zero-strip modulus request is exhausted by the displayed route through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$*. Such a request cannot become analytic continuation evidence, zero-free-strip evidence, a global zero classifier, or a host numeric oracle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessModulusRefusalExhaustion`

Proof. Begin with the zero-strip route factorization of Definition 548.245. It forces a zero-strip consumer through Z, S, H, C, P, N before any modulus row or RH-facing row is readable.

Next apply the root modulus-strip source totality theorem Definition 548.235. The modulus request is read against the same finite source rows $Z, S, M, R, Q, H, C, P, N$ and not against an ambient analytic continuation surface.

Close with the root bridge readback totality theorem Definition 548.226. It exports no RH proof, host zero classifier, analytic-continuation theorem, quotient representative, host equality, or ambient completeness coordinate. The modulus refusal is therefore exhausted by the displayed BEDC route. □

Theorem 548.253 (Critical-line witness RH-boundary nonescape)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the root packet exports no* RH *proof, no finished critical-line theorem, and no host analytic primitive. Its public surface is only the finite* $\mathsf{ZetaZeros}^{\uparrow}$–$\mathsf{CritStrip}^{\uparrow}$–$\mathsf{ZetaContinuationWitness}^{\uparrow}$ *source route, the* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *modulus rows, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh boundary public nonescape`

Proof. Read the packet through Definition 548.250. The available positive data are exactly the finite source, modulus, comparison, transport, continuation, provenance, and naming rows exposed before any line-facing read.

Use the analytic-continuation nonescape theorem Definition 548.246. It refuses analytic continuation, RH proof data, global zero classification, quotient representatives, host equality, and ambient completeness as exports of an accepted packet.

Finally apply Definition 548.252. Zero-strip modulus requests remain inside the displayed route and do not produce zero-free-strip evidence or a host analytic primitive. Thus the root packet cannot escape into an RH proof or a finished critical-line theorem. □

Theorem 548.254 (CriticalLineWitness source-budget carrier)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted packet exposes this finite tuple as the zero-strip source, source budget, modulus and readback route, refusal row, transport, continuation, provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. No accepted packet exports hidden analytic continuation, a host zero locator, a completed* RH *proof, or an off-packet critical-line oracle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source budget carrier`

Proof. Project the accepted packet through the carrier description of Definition 548.1. Coordinate exhaustion gives only the displayed rows $Z, S, M, R, Q, H, C, P, N$: the zero-strip source Z, source budget S, modulus and readback route M, R, Q, transport H, continuation C, provenance P, and local naming row N.

The existing root-strip source totality theorem Definition 548.250 places every line-facing read on this finite route before it can be consumed. The analytic continuation nonescape theorem Definition 548.246 then removes any ambient analytic continuation, host zero locator, completed RH proof, or off-packet critical-line oracle from the exported surface. □

Theorem 548.255 (CriticalLineWitness zero-strip refusal)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every zero-strip consumer factors through the displayed source, budget, modulus, readback, and refusal rows before downstream use. The consumer cannot replace the refusal row by a theorem about* RH*, an analytic-continuation bridge, or a completed zeta-zero classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip refusal`

Proof. Begin with the tuple projection supplied by Definition 548.254. It exposes the source, budget, modulus, readback, and refusal coordinates as components of the accepted BHist packet, rather than as host-side analytic data.

Apply the zero-strip route factorization theorem Definition 548.245. The consumer must pass through Z, S, M, R, Q and the refusal row before any downstream use can read the packet.

Finally use the modulus refusal exhaustion theorem Definition 548.252. Its nonescape clause prevents the refusal row from being replaced by an RH theorem, an analytic-continuation bridge, or a completed zeta-zero classifier. □

Theorem 548.256 (CriticalLineWitness NameCert obligation surface)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier exposure of* $Z, S, M, R, Q, H, C, P, N$*;*
2. *source-budget exactness for the zero-strip source and source budget;*
3. *determinacy of the modulus and readback route;*
4. *zero-strip refusal non-escape;*
5. *downstream stability through* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert obligation surface`

Proof. First project the accepted packet to its carrier coordinates. The source-budget carrier theorem Definition 548.254 supplies carrier exposure and fixes the zero-strip source and budget as visible packet rows.

Next read the route coordinates. The source-route totality theorem Definition 548.229 and the root bridge readback totality theorem Definition 548.226 determine the modulus and readback path through the same finite packet.

It remains to isolate the refusal and downstream clauses. The zero-strip refusal theorem Definition 548.255 gives non-escape for the refusal row, while the displayed H, C, P, N coordinates carry transport, continuation, provenance, and naming stability. These five rows exhaust the local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ surface. □

Theorem 548.257 (CriticalLineWitness source-budget public interface)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *source-budget packet. Its public interface consists exactly of the critical strip source row, dyadic rational tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{Zeta}^{\uparrow}$ *and* $\mathsf{ZetaZeros}^{\uparrow}$ *route rows, refusal budget row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. No* RH *conclusion, zero-count completeness theorem, host analytic-continuation object, or hidden limit is exported by this interface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source budget public interface`

Proof. Project the accepted packet through Definition 548.254. The projection exposes the critical strip source, dyadic tolerance budget, regular-rational readback, Real seal, zeta-route, refusal, transport, replay, provenance, and local naming coordinates as the full visible source-budget tuple.

Apply the NameCert obligation surface of Definition 548.256. Its carrier exposure, source-budget exactness, route determinacy, refusal non-escape, and downstream stability clauses identify those rows as the public interface rather than as private auxiliary data.

Finally use the zero-strip refusal theorem Definition 548.255. The non-escape clause prevents any consumer from replacing the refusal budget by an RH conclusion, zero-count completeness, host analytic continuation, or hidden limiting object. □

Theorem 548.258 (Critical-line witness RH source lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every RH-facing read of W first exposes the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row S, the shared* hsame *transport row H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The read may continue to the modulus and comparison rows only after that source lock is visible, and it receives no host zero source, analytic-continuation certificate, or* RH *proof from the packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh source lock`

Proof. Project the accepted packet through Definition 548.1. The carrier exposes $Z, S, M, R, Q, H, C, P, N$ as finite BHist rows and names Z and S as the supplied zeta-zero and critical-strip inputs.

Apply the zero-strip compatibility and non-escape rows of Definition 548.2. These rows bind the supplied $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ inputs through the displayed hsame transport and force every public read through the same local Cont, Pkg, and NameCert surface.

The sibling source chapters chapter 73 and chapter 72 supply only the named rows consumed by Z and S. Together with Definition 548.3 and Definition 548.250, this leaves no coordinate for a host zero source, analytic-continuation certificate, or RH proof before an RH-facing read begins. □

Theorem 548.259 (Critical-line witness modulus window exactness)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, each depth request to the modulus row is exhausted by the displayed* $\mathsf{Nat}^{\uparrow}$ *depth input to M, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger Q, and the displayed continuation route C. The request is not discharged by host real equality, quotient representatives, ambient analytic completeness, or any hidden numeric oracle.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus window exactness`

Proof. Read the modulus coordinate from Definition 548.1. The row M is the only depth-facing coordinate, and its answer is paired in the carrier with the real-part row R, rational ledger Q, transport row H, continuation row C, provenance row P, and local naming row N.

Apply the modulus-discipline and ledger-exactness rows of Definition 548.2. They require a depth request to consume the on-critical-line modulus row through R and Q, with replay performed by the displayed continuation route rather than by a host equality principle.

Now use the source-normalization and modulus-ledger interfaces Definition 548.59 and Definition 548.60. The source chapters chapter 28, chapter 38, and chapter 37 provide the named depth, real-part, and rational-comparison rows, so the modulus window is exact on those finite rows and has no route to host real equality, quotient representatives, ambient completeness, or a hidden oracle. □

Theorem 548.260 (Critical-line witness downstream refusal ledger)**.** *Every downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the displayed refusal ledger over $Z, S, M, R, Q, H, C, P, N$. The packet does not export an* RH *proof, a zero search, an analytic-continuation certificate, a quotient representative, or an ambient completeness witness to any downstream RH-facing, zero-facing, or modulus-facing consumer.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream refusal ledger`

Proof. Start with the refusal row in Definition 548.5. It states that the exported boundary is a finite packet ledger, not an analytic or global-zero object.

Apply Definition 548.258 for the source side and Definition 548.259 for the modulus side. These two rows force downstream consumers to read the supplied $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ source rows before the modulus window, and to replay the modulus request only through the displayed $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert data.

Finally cite the downstream RH boundary refinement Definition 548.105 and the sibling continuation refusal surface of chapter 580. They leave downstream consumers with the displayed refusal ledger alone. Since no packet coordinate names an RH proof, zero search, analytic-continuation certificate, quotient representative, or ambient completeness witness, none can be read from W. □

Theorem 548.261 (CriticalLineWitness critical-strip source surface)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted packet exposes the supplied critical-strip source row S, together with its* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *routes, before any zero-line consumer, modulus consumer, or* RH*-facing consumer row can read the witness. The exposed row is a packet source row, not a host critical-line theorem, analytic continuation primitive, quotient equality, or standard-model replacement.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier critical strip source surface`

Proof. Project W through the carrier description of Definition 548.1. The critical-strip coordinate S is one of the displayed finite BHist rows, and the only structural routes attached to it are the local H, C, P, N coordinates.

Apply the root-strip source totality theorem Definition 548.250. It places the $\mathsf{CritStrip}^{\uparrow}$ source row on the visible source route before any critical-line-facing read is available.

Use the RH source lock Definition 548.258 for downstream reads and the zero-strip compatibility obligation Definition 548.50 for zero-line reads. These rows force the supplied strip source to be visible through BHist, hsame, Cont, Pkg, and NameCert before any consumer reads the witness surface, and they leave no coordinate for a host critical-line theorem, analytic primitive, quotient equality, or standard-model replacement. □

Theorem 548.262 (CriticalLineWitness zero-refusal locality)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every zero-refusal row is local to the displayed packet. The refusal is witnessed inside the finite* BHist *carrier, the componentwise* hsame *transport row, the displayed* Cont *replay row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row; it is not a claim about an analytic zero primitive, global zero classifier, host quotient equality, analytic-continuation theorem, or* RH *truth value.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero refusal locality`

Proof. Start with the zero-strip refusal theorem Definition 548.255. It reads refusal as a row of the accepted BHist packet and requires zero-strip consumers to pass through the displayed source, budget, modulus, readback, and refusal rows.

Next apply the local transport obligation Definition 548.47. The componentwise hsame row, displayed Cont replay row, Pkg provenance, and local NameCert replay are bound to the same zero-strip-modulus ledger.

Finally use the downstream refusal ledger Definition 548.260. It excludes an analytic zero primitive, global zero classifier, quotient representative, analytic-continuation certificate, ambient completeness witness, and RH proof from downstream reads. The zero-refusal row is therefore local to W and does not export an analytic or host-level replacement. □

Theorem 548.263 (CriticalLineWitness modulus budget totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The modulus budget rows M, R, Q, H, C, P, N, read together with the supplied source rows Z, S, exhaust every consumer read before any critical-line witness is exported. A consumer can receive only the finite packet route through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$*; it cannot bypass that budget by requesting* RH *truth, analytic continuation, host equality, quotient representatives, or a standard-model replacement.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus budget totality`

Proof. Read the modulus coordinate through the modulus window exactness theorem Definition 548.259. Each depth request is exhausted by the displayed $\mathsf{Nat}^{\uparrow}$ depth input to M, the $\mathsf{Real}^{\uparrow}$ row R, the $\mathsf{Rat}^{\uparrow}$ comparison ledger Q, and the replay route through H, C, P, N.

Combine this with the modulus refusal exhaustion theorem Definition 548.252. Zero-strip modulus requests remain inside the same finite route and cannot become analytic continuation evidence, zero-free-strip evidence, a global zero classifier, or a host numeric oracle.

Finally apply the root bridge readback totality theorem Definition 548.226 and the root packet carrier theorem Definition 548.43. They exhaust public readback by the displayed packet rows $Z, S, M, R, Q, H, C, P, N$ before any critical-line witness can be exported, with no route to RH truth, host equality, quotient representatives, analytic continuation, or standard-model replacement. □

Definition 548.264 (CriticalLineWitness root obligation carrier)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet*

$$W = (Z, S, M, R, Q, H, C, P, N),$$

the root obligation carrier is the finite displayed surface consisting of the BHist *rows Z, S, M, R, Q, the componentwise* hsame *transport row H, the displayed* Cont *replay route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. A root-facing read is carried by this surface exactly when it enters through those rows and no additional analytic-continuation,* RH*, host-limit, quotient, or standard-bridge datum is supplied.*

Theorem 548.265 (CriticalLineWitness root classifier stability)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet. Any classifier read that is admitted by the root obligation carrier of Definition 548.264 is stable under the displayed route*

$$\mathsf{BHist} \xrightarrow{H} \mathsf{hsame} \xrightarrow{C} \mathsf{Cont}.$$

The read therefore has the same root-facing classifier before and after the hsame *transport and the* Cont *replay, and it remains attached to the same* Pkg *row P and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root classifier stability`

Proof. Project the accepted packet through Definition 548.196. It orders the displayed BHist rows and the H, C, P, N route before any critical-line read of M or R is available.

Next use Definition 548.100. The consumer sees only the same finite tuple $Z, S, M, R, Q, H, C, P, N$, so the classifier cannot be replaced by a detached host or quotient classifier.

Finally apply the non-escape package Definition 548.69. Since the packet exports no analytic continuation, RH proof, host equality, or ambient completion route, transport by H and replay by C preserve the root-facing classifier inside the local P, N certificate rows. □

Theorem 548.266 (CriticalLineWitness root NameCert obligation completion)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the root obligation carrier supplies the complete local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *obligation packet for a root-facing consumer. The supplied packet contains the source rows* Z, S*, the pattern rows* M, R, Q*, the classifier route* H, C*, the stability attachment to* P, N*, and the ledger refusal rows needed to keep the read inside the finite BEDC packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier obligation completion packet`
Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root namecert obligation completion`

Proof. Start with the carrier boundary Definition 548.196. It places $Z, S, M, R, Q, H, C, P, N$ on one root-facing surface and requires any critical-line read to pass through the local carrier and NameCert rows.

The classifier stability theorem Definition 548.265 then binds the source and pattern rows to the route through H and C. The same route keeps the provenance row P and the local naming row N attached to the accepted packet.

It remains to record the refusal side. Apply the non-escape package Definition 548.69 and the ledger refusal totality theorem Definition 548.243. Together they turn source, pattern, classifier, stability, and ledger obligations into one local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ packet with no external analytic or host-level witness. □

Theorem 548.267 (CriticalLineWitness root ledger nonescape)**.** *Every root-facing read of the obligation carrier Definition 548.264 is exhausted by the displayed packet rows* $Z, S, M, R, Q, H, C, P, N$*. The ledger exports no analytic-continuation theorem, no* RH *proof, no host-limit datum, no quotient representative, and no standard-bridge object.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root ledger nonescape`

Proof. Read the packet through Definition 548.100. The carrier projection is exactly $Z, S, M, R, Q, H, C, P, N$, so a root-facing consumer receives no additional coordinate from which analytic or host-level data could be extracted.

Next apply the non-escape package Definition 548.69. It identifies the root-facing export with the finite BHist, hsame, Cont, Pkg, and NameCert route.

Finally use Definition 548.243. The refusal ledger closes every request for analytic continuation, RH proof data, host limits, quotient representatives, or standard-bridge data at the local carrier boundary. □

Theorem 548.268 (CriticalLineWitness fixed-strip source readback totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet whose source is the fixed critical strip. Before any zeta-facing or* RH*-facing consumer can inspect the packet, the read must pass through the displayed* BHist *source rows* Z, S*, the rational-height and modulus rows* R, Q, M*, the zero-strip localization row* Z*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N *along the structural route* H, C, P*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip source readback totality`

Proof. Project the packet through the root obligation carrier of Definition 548.264. The carrier exposes only $Z, S, M, R, Q, H, C, P, N$, so the fixed-strip source is read as a displayed BHist row rather than as a host critical-line theorem.

Apply Definition 548.265. The same hsame and Cont route keeps the source row, rational-height row, modulus row, and zero-strip localization row attached to the accepted packet before downstream inspection.

Finally use the NameCert completion row Definition 548.266 and the zeta dependency lattice Definition 548.169. Together they force zeta and RH consumers to read the displayed source, modulus, zero-strip, provenance, and local naming rows before inspecting the witness surface. □

Theorem 548.269 (CriticalLineWitness zero-strip modulus separation)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the zero-strip refusal row is separated from the positive analytic-continuation payload by the displayed modulus ledger* M, R, Q, H, C, P, N *and by the* hsame*-stable classifier of the root obligation surface. Transport along* H *and replay along* C *therefore cannot turn a refusal row into an analytic-zero witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus separation`
↪ variant `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus separation certificate`

Proof. Start with the classifier stability row Definition 548.265. It keeps any root-facing classifier inside the displayed BHist, hsame, Cont, Pkg, and NameCert route.

Next apply the root ledger nonescape theorem Definition 548.267. The ledger has no coordinate for analytic continuation, host zero extraction, quotient representatives, or ambient completion, so its refusal row cannot be read as a positive analytic payload.

It remains to separate downstream zeta reads from the refusal row. Use Definition 548.169 together with Definition 548.189. They place zeta and RH consumers behind the same finite modulus and zero-strip readback route, leaving transport and replay unable to manufacture an analytic-zero witness. □

Theorem 548.270 (CriticalLineWitness RH-refusal consumer nonescape)**.** *Every downstream* RH*-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet may cite the fixed-strip source row* S*, the zero-strip refusal row* Z*, the modulus ledger* M, R, Q*, the* Cont *route* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. Such a consumer cannot extract a host zero, an analytic continuation theorem, a classical contradiction, a quotient stream, or an ambient completeness datum from those rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal consumer nonescape`

Proof. Read the consumer through the completed NameCert obligation Definition 548.266. It supplies the fixed-strip, zero-strip, modulus, transport, route, provenance, and naming rows as one finite local packet.

Apply Definition 548.267. This removes analytic-continuation theorems, RH proofs, host-limit data, quotient representatives, and standard-bridge objects from the root-facing export.

Finally combine the downstream counter-surface Definition 548.191 with the RH-refusal readback totality row Definition 548.189. The consumer may cite the displayed rows, but every extraction request for a host zero, analytic theorem, classical contradiction, quotient stream, or ambient completeness datum stops at the finite refusal ledger. □

Theorem 548.271 (CriticalLineWitness zero-strip source classifier lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The supplied zero row Z and the critical-strip row S determine the source/classifier surface only through the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows of W. No host zero search, quotient representative, analytic-continuation certificate, or ambient completeness row may replace the displayed Z, S, H, C, P, N route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip source classifier lock`

Proof. Project the accepted packet through Definition 548.1. The source-facing coordinates are the supplied zero row Z and critical-strip row S, and every classifier-facing read is attached to the same finite carrier.

Apply the visible source triad of Definition 548.286 and the root-line row exposure of Definition 548.244. These rows expose the BHist source before any downstream consumer can use the witness.

Use the supplied zero-strip transport lock Definition 548.71. It binds Z and S to the componentwise hsame row, the displayed Cont replay, the Pkg provenance row, and the local NameCert row.

The excluded objects are not coordinates of W and are not introduced by any of the cited rows. Thus the source/classifier surface is locked to the finite Z, S, H, C, P, N route, with no replacement by host zero search, quotient choice, analytic continuation, or ambient completeness. □

Theorem 548.272 (CriticalLineWitness modulus ledger readback exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. After the zero-strip source has been exposed through Z and S, every read of the modulus row M, the real-part row R, and the rational comparison ledger Q factors through M, R, Q, H, C, P, N. This readback exports neither a proof of* RH *nor a global zero classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus ledger readback exhaustion`

Proof. Begin with Definition 548.271. The zero-strip source is already exposed through the supplied source rows and the local transport, replay, provenance, and naming rows.

Apply the strip-modulus readback theorem Definition 548.142 and the root modulus classifier exhaustion theorem Definition 548.278. Together they place every modulus, real-part, and rational comparison read behind the displayed M, R, Q rows.

The same packet route is carried by Definition 548.47. Hence the readback cannot leave H, C, P, N after reading M, R, Q.

Finally use the modulus refusal scope of Definition 548.287. It keeps the rational modulus ledger from becoming a zero locator, analytic oracle, host equality, or RH proof. The readback is therefore exhausted by M, R, Q, H, C, P, N over the already exposed source rows. □

Theorem 548.273 (CriticalLineWitness RH boundary consumer nonescape)**.** *Every* RH*-facing downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ receives only the finite route*

$$Z, S, M, R, Q, H, C, P, N.$$

The boundary combines the zero-strip source classifier lock and the modulus ledger readback exhaustion with the refusal rows of Definition 548.72 and Definition 548.78; it exports no RH *proof,*

host zero search, global zero classifier, quotient representative, analytic-continuation certificate, or ambient completeness datum.

Proof. First apply Definition 548.271. It fixes the only source/classifier route available to the consumer before any RH-facing request is considered.

Next apply Definition 548.272. It routes the modulus row, real-part row, and rational comparison ledger through the same finite continuation, provenance, and local naming surface.

Use the refusal ledger completeness row Definition 548.72 and the root refusal route Definition 548.78. These rows classify RH proof requests, host zero search, quotient representatives, analytic-continuation certificates, global zero classifiers, and ambient completeness requests as outside the accepted packet.

Thus every downstream RH-facing consumer receives exactly the finite BEDC route $Z, S, M, R, Q, H, C, P, N$. The route is a readback boundary for the witness package, not an analytic escape hatch. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary consumer nonescape`

Theorem 548.274 (Critical-line witness source normalization lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before a root or downstream* RH*-boundary consumer reads any modulus, comparison, or refusal data, the supplied zero row Z and critical-strip row S are normalized as the source side of the same finite packet over* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$. *The source lock uses only the supplied zero row, critical-strip row, Nat depth row, Rat comparison ledger, Real-part row, H, C, P, N, and the existing root source triad.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source normalization lock`

Proof. Project the accepted packet W. The carrier exposes $Z, S, M, R, Q, H, C, P, N$ and has no coordinate for a global zero classifier, analytic-continuation theorem, host equality, quotient representative, or ambient completeness principle.

Apply the root source triad Definition 548.76 and the Lean-facing carrier root source triad. They identify the supplied zero row, the critical-strip row, and the source-facing comparison route as finite packet rows under the same transport, continuation, package, and local naming surface.

The modulus and comparison rows are therefore read only after Z and S have been exposed. Since the source side is locked before downstream inspection, an RH-boundary consumer receives a source-normalized packet rather than a zero search or analytic source theorem. □

Theorem 548.275 (Critical-line witness modulus ledger totality)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ whose source side is locked as in Definition 548.274, every modulus, Nat-depth, Rat-comparison, and Real-part read is total through the displayed ledger M, R, Q, H, C, P, N while retaining Z and S as the supplied zero-strip source rows. The ledger exports no hidden numeric oracle, host zero equality, quotient representative, analytic-continuation certificate, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus ledger totality`

Proof. Begin with the source-normalization lock. It places the supplied zero and critical-strip rows upstream of every modulus-facing read.

Apply the root comparison ledger Definition 548.77 together with the root modulus ledger Definition 548.68. These rows route Nat depth through M, Real-part data through R, and Rat comparison through Q, all under H, C, P, N.

Finally use the modulus-refusal scope already carried by the root packet. Because the ledger is finite and row-exact, every modulus request is answered by M, R, Q, H, C, P, N and cannot leave the packet for host equality, quotient data, analytic continuation, or ambient completeness. □

Theorem 548.276 (Critical-line witness RH-boundary refusal exhaustion)**.** *Every* RH*-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *is exhausted by the locked source side* Z, S*, the total modulus ledger* M, R, Q*, and the structural rows* H, C, P, N*. The consumer may cite a supplied zero row, critical-strip row, Nat depth, Rat comparison ledger, Real-part row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data, but it receives no* RH *proof, zero search, global zero classifier, analytic-continuation theorem, host equality, quotient stream, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary refusal exhaustion`

Proof. Use Definition 548.274 for the source side and Definition 548.275 for the comparison side. Together they expose exactly the finite rows $Z, S, M, R, Q, H, C, P, N$.

Apply the RH-refusal ledger completeness theorem Definition 548.72 and the RH-boundary refusal totality row Definition 548.194. These existing rows classify requests for proof content, zero search, analytic continuation, quotient data, host equality, and ambient completeness as refusal-side requests.

Thus the downstream boundary is exhausted by the packet itself. It remains a finite BEDC witness surface for supplied zero-strip and modulus data, not an analytic certificate and not a proof of RH. □

Theorem 548.277 (CriticalLineWitness root source obligation triad)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root consumer reads the supplied* BHist *source through the zero-strip row* Z *or the critical-strip row* S*, then through the modulus row* M*, the* hsame *transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N*. The consumer receives no root-facing source outside this finite triad and no* RH*-facing conclusion before those rows have been displayed.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source obligation triad`

Proof. Project the accepted packet through Definition 548.1. The only source-facing coordinates are the zero row Z and the critical-strip row S, and the only modulus-facing coordinate is M.

Apply the root source triad of Definition 548.76. It binds the source side to the displayed packet route before the comparison and refusal rows can be consumed.

Use the local transport obligation Definition 548.47. It carries the same source and modulus rows through H, C, P, N, so the root consumer must traverse the finite BHist, hsame, Cont, Pkg, and NameCert surface.

The RH boundary is then only a consumer boundary. Since no coordinate of W names an RH proof, analytic-continuation theorem, quotient representative, host equality, or standard-model replacement, the displayed triad exhausts the root source obligation. □

Theorem 548.278 (CriticalLineWitness root modulus classifier exhaustion)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the root modulus classifier is exhausted by the modulus row* M*, the real-part row* R*, the rational comparison ledger* Q*, the* hsame *transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*, with*

Z and S retained as the zero-strip or critical-strip source rows. No modulus consumer can inspect an unlisted classifier, hidden numeric oracle, quotient equality, or analytic source theorem.

Proof. Start from the root comparison ledger Definition 548.77. It routes comparison reads through the rational ledger Q while keeping the source rows Z, S visible.

Next apply the root modulus ledger Definition 548.68. It forces depth and modulus data to pass through M and the same structural rows rather than through a detached numeric oracle.

Transport and replay are fixed by Definition 548.47. The componentwise hsame row, the displayed Cont route, the provenance row, and the local name preserve the classifier route as one finite packet.

Thus the root modulus classifier has no remaining branch. Every accepted read is one of the displayed M, R, Q, H, C, P, N reads over the retained Z, S source, and every unlisted analytic or host-level classifier is outside the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus classifier exhaustion`

Theorem 548.279 (CriticalLineWitness root RH consumer boundary)**.** *Every* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ is bounded by the source obligation triad of Definition 548.277 and the modulus classifier exhaustion of Definition 548.278. The consumer may cite the supplied* BHist *source, zero-strip or critical-strip row, modulus row,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *row, but receives no RH proof, zero search, global zero classifier, analytic-continuation theorem, quotient stream, host equality, or ambient completeness datum.*

Proof. First use Definition 548.277. It places every root source read behind Z, S, M, H, C, P, N before any RH-facing consumer can inspect the packet.

Then use Definition 548.278. It exhausts modulus and comparison reads by M, R, Q, H, C, P, N while retaining the same zero-strip and critical-strip source rows.

Finally apply the RH-refusal ledger completeness theorem Definition 548.72 and the RH-boundary refusal totality row Definition 548.194. These rows classify requests for proof content, zero search, analytic continuation, quotient streams, host equality, and ambient completeness as outside the accepted packet.

The RH-facing consumer boundary is therefore a finite BEDC readback boundary. It exports the source, modulus, transport, replay, provenance, and local naming rows, not an analytic theorem and not a truth value for RH. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh consumer boundary`

Theorem 548.280 (CriticalLineWitness fixed-strip source admission completion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. The fixed-strip source admission row is complete when the consumer reads the supplied* BHist *source through the zero-strip row Z or critical-strip row S, then through the modulus row M, real-part row R, rational comparison ledger Q, transport row H, replay row C, provenance row P, and local name N. No fixed-strip consumer receives a source outside this packet before these rows have been displayed.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip source admission completion`

Proof. Begin with the source side of the root source obligation triad Definition 548.277. It places every root consumer behind the supplied BHist source, the zero-strip or critical-strip row, and the displayed modulus route.

Use modulus classifier exhaustion from Definition 548.278. Once the source row is admitted, every modulus, real-part, and rational-comparison read is exhausted by M, R, Q, H, C, P, N over the same Z, S source rows.

Thus fixed-strip source admission has no remaining hidden source branch. A consumer that skips Z, S or reads a source outside W leaves the accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. □

Theorem 548.281 (CriticalLineWitness zero-source refusal readback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. Any request to read a zero-source refusal must pass through the source obligation triad and the modulus classifier exhaustion rows, so the refusal is read back through $Z, S, M, R, Q, H, C, P, N$ and not through a hidden zero search, global zero classifier, analytic-continuation theorem, quotient stream, host equality, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero source refusal readback`

Proof. Apply the root source obligation triad Definition 548.277. It forces the zero-strip and critical-strip source rows to remain visible before a refusal boundary can be consumed.

Next apply modulus classifier exhaustion Definition 548.278. The refusal readback cannot inspect an unlisted classifier, since modulus and comparison data are already exhausted by M, R, Q, H, C, P, N.

The zero-source refusal is therefore a finite readback row of the accepted packet. Demands for hidden zero search, global zero classification, analytic continuation, quotient streams, host equality, or ambient completeness are outside the displayed carrier. □

Theorem 548.282 (CriticalLineWitness RH consumer unblock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. An RH-facing consumer is unblocked exactly to the finite readback surface supplied by the source obligation triad and modulus classifier exhaustion: it may consume $Z, S, M, R, Q, H, C, P, N$, but it receives no RH proof, no zero search, no global zero classifier, no analytic-continuation theorem, no quotient stream, no host equality, and no ambient completeness coordinate.*

Proof. Start from the RH consumer boundary Definition 548.279. It already identifies the consumer surface as the source obligation triad together with modulus classifier exhaustion.

The fixed-strip source admission completion above supplies the source-read side, and the zero-source refusal readback supplies the refusal side. Both rows keep the same accepted packet $W = (Z, S, M, R, Q, H, C, P, N)$ and the same BHist, hsame, Cont, Pkg, and NameCert support rows.

Consequently the RH consumer is unblocked only as a finite BEDC consumer. It can read the displayed root and modulus rows, but the analytic conclusion, zero search, global classifier, quotient stream, host equality, and ambient completeness demands have no coordinate in the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRhConsumerUnblock`

Theorem 548.283 (Critical-line witness source-modulus exclusion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. Every source-modulus consumer of W must read the supplied $\mathsf{ZetaZeros}^{\uparrow}$ row Z, the $\mathsf{CritStrip}^{\uparrow}$ row S, the $\mathsf{Nat}^{\uparrow}$ modulus row M, the $\mathsf{Real}^{\uparrow}$ real-part row R, the $\mathsf{Rat}^{\uparrow}$ comparison ledger Q, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows before it can inspect any downstream boundary. The route exports no global zero classification, analytic-continuation theorem, host equality, quotient representative,* RH *proof content, or hidden ambient completeness datum.*

Proof. Begin with the source normalization interface. Definition 548.59 exposes the supplied zero row, critical-strip row, Nat depth row, rational comparison ledger, and real-part row as the finite packet $Z, S, M, R, Q, H, C, P, N$.

Next apply the root source-modulus admission theorem. Definition 548.64 places every source-modulus read after the displayed zero-strip source and comparison rows, under the same transport, continuation, package, and local naming surface.

Finally use the RH refusal interface. Definition 548.62 classifies requests for global zero classification, analytic continuation, host equality, quotient representatives, proof content, and ambient completeness as refusal-side requests rather than source-modulus data. The admitted consumer is therefore confined to $Z, S, M, R, Q, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus exclusion`

Theorem 548.284 (Critical-line witness zero-strip route totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every zero-strip route used by a root, zeta-facing, or* RH*-boundary consumer is total through the displayed source-modulus packet*

$$Z, S, M, R, Q, H, C, P, N.$$

The route reads the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, componentwise transport, displayed continuation, provenance, and local naming, and it cannot read a global zero classifier, analytic-continuation certificate, host equality, quotient representative, or RH *proof.*

Proof. First use source-modulus exclusion. It confines the consumer to the packet $Z, S, M, R, Q, H, C, P, N$ before any zero-strip or boundary read is available.

Now apply the zero-strip budget factorization theorem Definition 548.65. That theorem identifies zero-strip budget requests with the supplied Z, S rows and the shared modulus ledger M, R, Q, H, C, P, N.

Finally use the zeta-consumer non-escape theorem Definition 548.327. Zeta-facing and RH-facing consumers remain on the same finite readback surface, so totality of the zero-strip route carries no analytic-continuation, host-equality, quotient, global-classifier, or proof-content coordinate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip route totality`

Theorem 548.285 (Critical-line witness RH-boundary nonescape)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every* RH*-boundary consumer whose read is served by the source-modulus and zero-strip rows of W is exhausted by $Z, S, M, R, Q, H, C, P, N$. It may read the displayed source row, strip row, modulus row, real-part row, rational comparison ledger,* BHist *history,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it cannot read global zero classification, analytic continuation, host equality, quotient representatives, or* RH *proof content.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary nonescape`

Proof. Start with zero-strip route totality. The boundary consumer has already been routed through $Z, S, M, R, Q, H, C, P, N$ and has no second zero-strip or host-real channel.

Next apply Definition 548.276. The RH-boundary side is exhausted by the locked source side Z, S, the total modulus ledger M, R, Q, and the structural rows H, C, P, N.

Finally combine this with Definition 548.312. Downstream consumers receive a refusal packet rather than an analytic claim, so global zero classification, analytic continuation, host equality, quotient representatives, and proof content have no carrier row from which to escape. □

Theorem 548.286 (CriticalLineWitness visible source triad)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *root witness read, the consumer-visible surface exposes the critical-strip source row, the rational modulus row, the zero-strip refusal row, and the local* BHist, hsame, Cont, Pkg, *and* NameCert *provenance before the witness can be used downstream.*

Proof. Project the accepted packet through Definition 548.1. The source, modulus, refusal, transport, continuation, provenance, and local naming rows are all displayed coordinates of the same finite witness carrier.

Apply the root source obligation triad of Definition 548.277. It forces every root consumer to read the supplied source through the displayed zero-strip or critical-strip row and through the modulus row before the consumer-facing boundary is reached.

Use the local transport obligation Definition 548.47. It carries the source and modulus rows through the hsame, Cont, Pkg, and NameCert surface, so no downstream read can bypass the visible provenance. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier visible source triad`

Theorem 548.287 (CriticalLineWitness modulus refusal scope)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *root witness read, the rational modulus row is a finite refusal-scoped ledger. It cannot be promoted to a zeta-zero locator, analytic-continuation oracle, or host complex equality.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus refusal scope`

Proof. Begin with Definition 548.286. The modulus row is displayed only beside the source, refusal, transport, continuation, provenance, and local naming rows.

Apply Definition 548.278. It exhausts the modulus-facing classifier by the finite packet rows and rejects hidden numeric or analytic classifiers.

Finally use the RH-refusal boundary rows of Definition 548.194. They classify requests for zero location, analytic continuation, and host equality as outside the accepted witness packet. □

Theorem 548.288 (CriticalLineWitness RH consumer handoff)**.** *Every* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *root witness receives only the displayed source row, rational modulus row, zero-strip refusal row, transport rows, continuation rows, provenance rows, and local naming rows before using the witness.*

Proof. Start from Definition 548.286. It exposes the exact source, modulus, refusal, and local provenance rows that any root witness read must traverse.

Use Definition 548.287. The modulus row remains a refusal-scoped finite ledger and does not become a zero locator, analytic-continuation oracle, or host equality.

Then apply Definition 548.279. It bounds every RH-facing consumer by the displayed source, modulus, transport, replay, provenance, and local naming rows. Therefore the handoff is a finite BEDC readback package, not an analytic conclusion about RH. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer handoff`

Theorem 548.289 (CriticalLineWitness visible source readback lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every visible source read of W is locked to the displayed zero-strip source triad and comparison ledger: the read must expose Z, S, Q as the source side and M, R, Q, H, C, P, N as the comparison and structural side before any downstream consumer can use the witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier visible source readback lock`

Proof. Start with the visible source triad, Definition 548.286. It places the critical-strip source row, rational modulus row, zero-strip refusal row, and local structural provenance on the consumer-visible surface.

Use the root source triad, Definition 548.76, to identify the source side of the read as the displayed Z, S, Q packet. Then use the root comparison ledger, Definition 548.77, to route the comparison side through M, R, Q, H, C, P, N.

Finally apply RH-refusal ledger completeness, Definition 548.72. It prevents the visible readback from replacing the displayed rows by an RH proof, a zero search, analytic continuation, quotient data, host equality, or ambient completeness. □

Theorem 548.290 (CriticalLineWitness fixed-strip modulus totality)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every fixed-strip modulus consumer factors through the finite rows $Z, S, M, R, Q, H, C, P, N$. The consumer has no accepted access to a detached modulus oracle, analytic-continuation row, zero-location classifier, quotient representative, host equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip modulus totality`

Proof. First apply Definition 548.289. It binds fixed-strip source access to the same source triad and comparison ledger.

Next use the downstream RH boundary refinement, Definition 548.105. The refinement says that downstream consumers receive only the finite package routed through the displayed source, comparison, transport, replay, provenance, and local naming rows.

The remaining possible escapes are removed by Definition 548.72. Hence every fixed-strip modulus read is total over $Z, S, M, R, Q, H, C, P, N$ and cannot import a stronger analytic or host-side witness. □

Theorem 548.291 (CriticalLineWitness RH refusal export boundary)**.** *Every* RH*-facing export of an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$ receives only the finite refusal package carried by $Z, S, M, R, Q, H, C, P, N$. It receives no* RH *proof, analytic-continuation certificate, zero-search result, quotient representative, host equality proof, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal export boundary`

Proof. Begin with fixed-strip modulus totality, Definition 548.290. It forces every fixed-strip modulus consumer through the displayed packet rows.

Now read the export boundary through Definition 548.105. The downstream interface is the finite refusal package, not a host analytic conclusion.

Finally apply RH-refusal ledger completeness, Definition 548.72. Its exclusion clauses remove RH proof data, analytic continuation, zero search, quotient representatives, host equality, and ambient completeness from the export surface, so only the displayed refusal package remains. □

Theorem 548.292 (CriticalLineWitness source-modulus refusal totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every* RH*-facing downstream read of W factors through the displayed finite source/modulus/refusal surface*

$$Z, S, M, R, Q, H, C, P, N$$

over BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert$_{\text{CriticalLineWitness}^{\uparrow}}$ *row. The read has no coordinate for analytic continuation, host zero search, quotient representative, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus refusal totality`

Proof. Project the packet through Definition 548.1. The carrier displays exactly the source rows Z, S, modulus and real rows M, R, rational comparison ledger Q, transport row H, continuation row C, package row P, and local naming row N.

Apply the root source triad Definition 548.76. It makes the zero-strip and critical-strip source side finite before any downstream consumer reads the witness. Then use the root comparison ledger Definition 548.77; the modulus and rational comparison side is read only through M, R, Q, H, C, P, N.

Finally apply the downstream refusal package Definition 548.34. It removes analytic-continuation proofs, host zero searches, quotient representatives, host equality, and ambient completeness from the downstream interface. The only remaining read surface is the displayed finite source/modulus/refusal packet. □

Theorem 548.293 (CriticalLineWitness RH boundary source nonescape)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, an* RH*-boundary consumer cannot replace the displayed source rows Z, S, Q by an analytic-continuation source, a host zero locator, a quotient stream, or a completeness coordinate. The boundary source remains the finite* BHist *route carried through H, C, P, N.*

Proof. Begin with the root source triad of Definition 548.76. It identifies the supplied zero row, critical-strip row, and rational comparison ledger as the source side available to the root and downstream boundary.

Use the root comparison ledger Definition 548.77. The comparison source is attached to M, R, Q and transported through H, C, P, N, so the boundary has no separate source coordinate outside the accepted packet.

Now apply Definition 548.34. The excluded requests are precisely the analytic source replacement, host zero search, quotient representative, host equality, and ambient completeness routes. Therefore the RH boundary cannot escape the finite BHist source surface displayed by W. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary source nonescape`

Theorem 548.294 (CriticalLineWitness modulus refusal readback)**.** *Every downstream modulus readback of an accepted* CriticalLineWitness$^{\uparrow}$ *packet is exhausted by the rows M, R, Q, H, C, P, N over the displayed source rows Z, S. It may report the finite modulus, real-part, rational comparison,* hsame*,* Cont*,* Pkg*, and* NameCert *data, but it cannot report analytic continuation, a host zero, quotient equality, or a completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus refusal readback`

Proof. Read the modulus row from the carrier Definition 548.1. The modulus-facing coordinates are M, R, Q, H, C, P, N, and the source context retained by that read is Z, S.

Apply the root comparison ledger Definition 548.77. It forces each modulus and comparison read through the rational ledger and the displayed transport, continuation, package, and naming rows.

Use the source-modulus refusal totality of Definition 548.292. The totality row places the modulus readback inside the same finite downstream refusal surface.

Finally apply Definition 548.34. It refuses analytic continuation, host zero data, quotient equality, and completeness witnesses as readback outputs, so the modulus readback is exactly the displayed finite packet route. □

Theorem 548.295 (Critical-line witness root route exposure)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted route that can feed a downstream zeta or* RH *consumer first exposes the critical-strip source, the zero-strip source and modulus row, the refusal ledger, the* BHist *carrier, the* hsame *transport row, the* Cont *replay row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The route has no accepted read before the displayed source/modulus/refusal surface is present.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root route exposure`

Proof. Project the accepted carrier of Definition 548.1. Its displayed rows are the source rows Z, S, the modulus and comparison rows M, R, Q, and the structural rows H, C, P, N.

Apply the root source triad Definition 548.76 and the zero-strip source totality row Definition 548.180. These rows expose the critical-strip and zero-strip source side as finite BHist data before any downstream read occurs.

Next use Definition 548.64 and Definition 548.292. They attach the zero-strip modulus and refusal ledger to the same displayed package, carried through hsame, Cont, Pkg, and the local NameCert row.

Finally apply Definition 548.154. Every downstream zeta or RH readback is exhausted by this displayed package, so no accepted route can bypass the exposed source/modulus/refusal surface. □

Theorem 548.296 (Critical-line witness zero-strip modulus refusal)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. If a zero-strip modulus read is absent from the displayed route or mismatches the displayed modulus row M, the read is routed to the visible refusal ledger of W. It is not reinterpreted as a hidden* RH *coordinate, an analytic-continuation coordinate, a host zero classifier, a quotient representative, or an ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip modulus refusal`

Proof. Begin with Definition 548.295. Any accepted route has already exposed the zero-strip source, the modulus row, the refusal ledger, and the structural BHist, hsame, Cont, Pkg, and NameCert rows.

Use the zero-strip source totality theorem Definition 548.180. It keeps the zero-strip source read inside the displayed finite route, rather than opening an analytic or host-level coordinate.

Apply Definition 548.294 and Definition 548.292. A modulus readback that is missing or mismatched has only the displayed refusal side available, because the carrier exports no second modulus or hidden analytic continuation coordinate.

Finally use the downstream refusal package Definition 548.34. The excluded RH proof, analytic-continuation certificate, host zero classifier, quotient representative, and ambient completeness witness are refusal-side requests, so the failed modulus read lands in the visible ledger. □

Theorem 548.297 (Critical-line witness downstream readback exhaustion)**.** *Every downstream* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, *zeta-boundary, or* RH *consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet reads only the displayed source-modulus-refusal package*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer cannot draw source, modulus, refusal, transport, continuation, package, or local naming data from a hidden analytic-continuation route, a host equality, a global zero classifier, a quotient representative, a selected completion coordinate, or an ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream readback exhaustion`

Proof. First apply Definition 548.295. It places the critical-strip source, zero-strip source, modulus, refusal ledger, BHist, hsame, Cont, Pkg, and local NameCert rows on the route before any downstream consumer reads the packet.

Next use the root RH boundary row Definition 548.144. This keeps the downstream boundary attached to the finite source/modulus surface instead of replacing it by an RH truth value or host analytic datum.

Apply Definition 548.296. Missing or mismatched modulus requests are sent to the visible refusal ledger, so they cannot produce an alternate source for downstream zeta or RH consumers.

Finally invoke Definition 548.154 and Definition 548.292. These rows exhaust downstream readback at the displayed source-modulus-refusal package, leaving no hidden coordinate from which another consumer route can be read. □

Theorem 548.298 (Critical-line witness window modulus exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted source-window read of* W *that can feed a downstream zero-location consumer factors through the displayed finite route*

$$Z, S, M, R, Q, H, C, P, N$$

over the zero-strip source, rational-height modulus, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. No downstream consumer can use the read before these source-window, modulus, package, and naming rows have been exposed.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier window modulus exhaustion`

Proof. Project the carrier of Definition 548.1. The accepted packet exposes exactly the source rows Z, S, the modulus and real rows M, R, the rational comparison row Q, and the structural rows H, C, P, N.

Apply the source-window separation row Definition 548.239 and the source-modulus refusal totality row Definition 548.292. Together they place the source-window read behind the displayed zero-strip source and the rational-height modulus route.

Finally use the fixed-strip readback surface Definition 548.268. It forces any zero-location consumer to pass through BHist, hsame, Cont, Pkg, and the local NameCert row before reading the strip. Hence the downstream consumer sees only the displayed finite route. □

Theorem 548.299 (Critical-line witness zero-strip refusal totality)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the zero-strip carrier and modulus route jointly refuse every request for an* RH *proof, analytic-continuation certificate, or host zero-location claim outside the finite displayed packet. The refusal is total for downstream zero-strip consumers whose reads pass through* $Z, S, M, R, Q, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip refusal totality`

Proof. Start with Definition 548.298. It places each downstream zero-location read behind the zero-strip source, rational-height modulus, transport, continuation, package, and naming rows of the accepted packet.

Apply the root RH source lock Definition 548.258 and the downstream refusal package Definition 548.34. These rows classify proof-of-RH requests, analytic-continuation requests, and host zero-location requests as refusal-side requests rather than carrier coordinates.

Use the zero-strip consumer boundary Definition 548.143. Since the consumer receives only the displayed zero-strip witness package, no outside proof, analytic certificate, or host zero-location claim remains available. □

Theorem 548.300 (Critical-line witness RH source ledger closure)**.** *Every* RH*-facing downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the source, modulus, and refusal ledger carried by*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer receives no proof of RH*, no zero-free strip theorem, and no analytic-continuation primitive. Its public read is the finite source/modulus/refusal ledger over* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh source ledger closure`

Proof. First use the window-modulus exhaustion theorem Definition 548.298. It gives the source and modulus side of the ledger before any downstream zero-location consumer can read the packet.

Next apply the zero-strip refusal totality theorem Definition 548.299. It removes RH proofs, analytic-continuation certificates, and host zero-location claims from the downstream zero-strip interface.

Finally compose these rows with the zeta dependency lattice Definition 548.169 and the downstream refusal package Definition 548.34. The lattice supplies only dependency information over the existing zeta and RH boundary surfaces, while the refusal package blocks the proof, zero-free strip, and analytic-continuation outputs. Therefore the RH-facing consumer receives exactly the finite source/modulus/refusal ledger. □

Theorem 548.301 (Critical-line witness root finite-source carrier)**.** *Every root-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is carried by the finite source tuple*

$$\mathsf{CritStrip}^{\uparrow},\ \mathsf{Gamma}^{\uparrow},\ \mathsf{ZetaZeros}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Equivalently, the packet exposes only the supplied zero row, fixed critical-strip row, real-part row, rational comparison row, transport row, continuation row, provenance row, and local naming row through the carrier of Definition 548.1. It does not expose analytic continuation, a global zero classifier, host complex equality, quotient representatives, or ambient analytic completeness.

Proof. Project the carrier of Definition 548.1. The accepted packet lists the zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, componentwise transport, displayed continuation, provenance, and local naming row as one finite BHist packet.

Use the root route exposure theorem Definition 548.295. It places every downstream zeta or RH route behind the same source, modulus, refusal, transport, continuation, package, and naming surface.

The $\mathsf{Gamma}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ source obligations are read only as finite source rows for the same packet; the $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows are the displayed comparison and real-part rows. Since no carrier coordinate has the type of analytic continuation, host equality, quotient data, or ambient completeness, the listed source tuple is exhaustive. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootFiniteSourceCarrier`

Theorem 548.302 (Critical-line witness root zero-strip modulus totality)**.** *Every zero-strip modulus read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet is tied to the finite source tuple of Definition 548.301. A successful read factors through the displayed* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{ZetaZeros}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; a missing or mismatched read lands in the same visible refusal ledger rather than producing a hidden modulus source.*

Proof. Start with Definition 548.138. It identifies the zero and strip source as supplied packet rows, not as an external zero-search service.

Next apply Definition 548.139 and Definition 548.292. These rows bind each modulus request to the displayed modulus, real-part, rational-comparison, transport, continuation, provenance, and naming surface, and send absent or mismatched requests to refusal.

Finally use Definition 548.296. The refusal side cannot be reinterpreted as analytic continuation, a host zero classifier, quotient representative, ambient completeness witness, or RH proof, so the zero-strip modulus read is total within the finite source tuple. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootZeroStripModulusTotality`

Theorem 548.303 (Critical-line witness root RH refusal ledger)**.** *Every* RH*-shaped consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the boundary/refusal ledger determined by Definition 548.301 and Definition 548.302. The consumer may read the fixed strip, supplied zero row, comparison rows, transport, continuation, provenance, and local naming data, but it receives no analytic-continuation theorem, no global* RH *proof datum, no host zero-location classifier, no quotient representative, and no ambient completeness row.*

Proof. First apply Definition 548.301. It fixes the carrier surface seen by a root consumer at the listed finite BEDC rows.

Then use Definition 548.302. Any modulus-facing demand needed by an RH-shaped consumer is either read through those same rows or returned as the visible refusal ledger.

Finally invoke the RH consumer nonescape theorem Definition 548.125. The downstream consumer receives only the finite witness boundary, so no analytic-continuation theorem, global proof, host classifier, quotient representative, or ambient completeness row is exported. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh refusal ledger`

Theorem 548.304 (Critical-line witness root finite-source NameCert obligations)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before any downstream zero-strip consumer reads* W*, the root surface exposes the finite zero-strip source rows* Z, S*, the rational modulus and comparison rows* M, R, Q*, the zeta and* RH *boundary rows, the classifier ledger,*

and the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows carried by* H, C, P, N*. No consumer can read the packet through an unshown analytic, quotient, host-equality, or global-zero coordinate.*

Proof. Project first to the root finite-source carrier of Definition 548.301. It identifies the zero row, fixed strip row, rational comparison row, transport, replay, provenance, and local naming row as finite displayed data of W.

Next use the fixed-strip source readback totality Definition 548.268 and the zeta dependency lattice Definition 548.169. These rows place every zeta-facing or RH-facing read behind the same supplied source, modulus, comparison, continuation, package, and naming surface.

Finally apply the classifier exactness and ledger stability rows Definition 548.358 and Definition 548.359. They keep the classifier and modulus ledger on $Z, S, M, R, Q, H, C, P, N$, so the downstream zero-strip consumer cannot bypass the displayed BHist/hsame/Cont/Pkg/NameCert route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite source namecert obligations`

Theorem 548.305 (Critical-line witness root classifier refusal boundary)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the root classifier refuses host zero selection, analytic-continuation shortcuts, quotient equality, hidden modulus selectors, and unledgered critical-line witnesses. Accepted classifier reads are exactly the packet-local reads through the visible zero-strip source, fixed-strip source, modulus, rational comparison, transport, continuation, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root classifier refusal boundary`

Proof. Start with the root classifier stability theorem Definition 548.265. It attaches classifier reads to the displayed source, modulus, transport, continuation, provenance, and local naming route of the accepted packet.

Use the root ledger nonescape row Definition 548.267 together with the root RH refusal ledger Definition 548.303. These rows remove analytic continuation, host zero extraction, quotient representatives, global RH proof data, and ambient completeness from the root-facing export.

It remains to rule out hidden modulus selectors and unledgered critical-line witnesses. The root zero-strip modulus totality Definition 548.302 and the zero-strip modulus separation Definition 548.269 force every modulus or refusal read through M, R, Q, H, C, P, N, leaving no classifier branch outside the visible ledger. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootClassifierRefusalBoundary`

Theorem 548.306 (Critical-line witness visible modulus consumer totality)**.** *Every downstream consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet that requests a zero-strip, critical-line, zeta-boundary, or* RH*-boundary modulus factors through the finite displayed rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The factorization uses the BHist *source,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *route; a failed or mismatched request returns the same visible refusal ledger rather than a hidden analytic source.*

Proof. Read the source side through Definition 548.304. It places zero-strip, critical-line, zeta-boundary, and RH-boundary access behind the displayed source and naming rows before any consumer sees the packet.

Next apply the zero-strip modulus totality Definition 548.302 and the classifier refusal boundary Definition 548.305. Together they require modulus and classifier reads to pass through M, R, Q, H, C, P, N and return off-route demands to the visible refusal ledger.

Finally combine the downstream consumer unblock row Definition 548.360 with the RH-refusal consumer nonescape theorem Definition 548.270. These consumer rows exhaust the available downstream surface by $Z, S, M, R, Q, H, C, P, N$, so every accepted modulus consumer factors through the finite displayed route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root visible modulus consumer totality`

Theorem 548.307 (Critical-line witness root finite-budget source totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root finite-budget source read of W is total on the displayed zero-strip source rows, rational modulus and comparison ledger, transport, continuation, package, and local naming surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The source read may expose the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *row, the* $\mathsf{CritStrip}^{\uparrow}$ *strip row, the* $\mathsf{Nat}^{\uparrow}$ *depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exposes no* RH *proof, analytic continuation, host zero classifier, quotient representative, host equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite budget source totality`

Proof. Start with the carrier route exactness and root source triad, Definition 548.310 and Definition 548.76. They place the supplied zero row, critical-strip row, and rational comparison ledger inside the same finite carrier.

Next apply the modulus ledger entry and root modulus-window totality, Definition 548.83 and Definition 548.85. These rows make M, R, Q, H, C, P, N the whole depth and rational-comparison route available to a root consumer.

Finally use the supplied zero-strip transport lock and RH-refusal ledger completeness. The structural rows hsame, Cont, Pkg, and NameCert carry the source and modulus faces together, while the refusal ledger removes analytic continuation, host classifiers, quotient representatives, host equality, ambient completeness, and any proof of RH. □

Theorem 548.308 (Critical-line witness root finite-budget ledger stability)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, replacing a root finite-budget presentation by another presentation with the same displayed Z, S, M, R, Q rows preserves the rational modulus ledger exactly through the same* hsame/Cont/Pkg/NameCert *transport surface. No stable ledger comparison may introduce a hidden modulus source, analytic-continuation certificate, quotient representative, host equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootFiniteBudgetLedgerStability`
↪ variant `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite budget ledger stability`

Proof. First read the finite-budget source totality just established. It fixes the available source and comparison rows at $Z, S, M, R, Q, H, C, P, N$, so a second presentation has no additional carrier coordinate to inspect.

Next apply Definition 548.77 and Definition 548.86. The comparison ledger Q is transported only with M, R, H, C, P, N, and the zero-strip side shares the same componentwise hsame and displayed Cont routes.

It remains to exclude alternate ledger sources. Use Definition 548.72. The refusal ledger has no slot for analytic continuation, quotient data, host equality, ambient completeness, or an RH proof, so the stable comparison is precisely the displayed finite-budget ledger. □

Theorem 548.309 (Critical-line witness root finite-budget RH refusal non-escape)**.** *Every* RH-*facing consumer of the root finite-budget surface of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the finite refusal package over*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may cite the zero-strip source, rational modulus ledger, comparison row, hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row, but it cannot extract* RH*, analytic continuation, a standard bridge, a global zero classifier, quotient equality, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite budget rh refusal nonescape`

Proof. Begin with Definition 548.307. It identifies the root finite-budget source as a finite packet over the displayed source, modulus, comparison, and structural rows.

Then use Definition 548.308. Any presentation of the same finite source keeps the rational ledger on the same hsame/Cont/Pkg/NameCert route.

Finally apply Definition 548.84 and Definition 548.87. These refusal rows make the RH-facing export a finite boundary only; analytic continuation, standard bridges, global zero classification, quotient equality, host equality, ambient completeness, and RH proof data are absent from the packet. □

Theorem 548.310 (Critical-line witness carrier route exactness)**.** *Let* W *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The carrier route exposed by* W *is exactly the finite tuple*

$$Z, S, M, R, Q, H, C, P, N.$$

Here Z *is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *zero row,* S *is the* $\mathsf{CritStrip}^{\uparrow}$ *strip row,* M *is the depth row,* R *is the real-part row,* Q *is the rational comparison row, and* H, C, P, N *are the displayed transport, continuation, provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. No accepted packet exposes a tenth carrier coordinate.*

Proof. Project the carrier in Definition 548.1. Its public coordinates are the zero, strip, depth, real-part, rational comparison, transport, continuation, provenance, and local naming rows.

Use the fixed-strip tuple of Definition 548.313 and the row exhaustion theorem Definition 548.314. They identify the visible fixed-strip source route with the nine displayed rows and exclude any additional analytic, quotient, host-equality, or completeness coordinate.

Finally apply Definition 548.295. Every downstream zeta or RH route first exposes the same source, modulus, refusal, transport, continuation, package, and naming surface. Hence the accepted packet exposes exactly $Z, S, M, R, Q, H, C, P, N$. □

Lean4 ✓ BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier carrier route exactness

Theorem 548.311 (Critical-line witness modulus route totality)**.** *For every accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every depth or modulus request is total as a finite route: it factors through*

$$M, R, Q, H, C, P, N$$

or it lands in the visible refusal ledger of W*. The request cannot be completed by a hidden modulus source, analytic-continuation primitive, host equality, quotient representative, ambient completeness row, or proof of* RH*.*

Lean4 ✓ BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier depth modulus route totality

Proof. Start with the carrier route exactness theorem Definition 548.310. It leaves only the displayed packet rows available to an accepted depth or modulus read.

Apply the modulus readback row Definition 548.139 and the source-modulus refusal totality theorem Definition 548.292. Accepted requests pass through M, R, Q together with the transport, continuation, package, and naming rows; absent or mismatched requests are sent to the visible refusal side.

Use the zero-strip modulus refusal theorem Definition 548.296. It blocks reinterpretation of a failed modulus read as an RH coordinate, analytic-continuation coordinate, host zero classifier, quotient representative, or ambient completeness witness. Thus the finite route is total by acceptance or refusal. □

Lean4 ✓ BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus route totality certificate

↪ variant BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus refusal totality

Theorem 548.312 (Critical-line witness downstream RH boundary nonescape)**.** *Every downstream* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the finite witness/refusal package*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer receives no proof of RH*, no zero search, no analytic-continuation primitive, no host equality, no quotient representative, and no ambient completeness datum.*

Proof. First apply Definition 548.310. It fixes the carrier route available to downstream consumers at the nine displayed rows.

Next apply Definition 548.311. Depth and modulus requests either pass through M, R, Q, H, C, P, N or land in the visible refusal ledger, so the downstream consumer has no alternate modulus or comparison source.

Finally use the RH consumer boundary theorem Definition 548.316 and the downstream refusal package Definition 548.34. These rows identify the consumer-facing object as a finite witness/refusal boundary and exclude proof, search, analytic-continuation, host-equality, quotient, and completeness exports. □

Definition 548.313 (Critical-line witness fixed-strip source tuple)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet, the fixed-strip source tuple is*

$$W = (Z, S, M, R, Q, H, C, P, N).$$

Here Z is the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *zero row, S is the supplied* $\mathsf{CritStrip}^{\uparrow}$ *fixed-strip row, M is the* $\mathsf{Nat}^{\uparrow}$ *depth row, R is the* $\mathsf{Real}^{\uparrow}$ *real-part row, Q is the* $\mathsf{Rat}^{\uparrow}$ *rational comparison row, H is the componentwise transport row, C is the displayed continuation row, P is the package provenance row, and N is the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The tuple is the row inventory of Definition 548.123 with no extra analytic-continuation, zero-search, host-equality, quotient, or completeness coordinate.*

Lean4 (definition) `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessFixedStripSourceTuple`

Theorem 548.314 (Critical-line witness fixed-strip row exhaustion)**.** *Every fixed-strip read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the nine rows*

$$Z, S, M, R, Q, H, C, P, N$$

of Definition 548.313. The read may inspect the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *row, the* $\mathsf{CritStrip}^{\uparrow}$ *strip row, the* $\mathsf{Nat}^{\uparrow}$ *depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *rational comparison row, and the displayed transport, continuation, package, and local naming rows, but it has no tenth row outside that tuple.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip row exhaustion`

Proof. Start with the fixed-strip source of Definition 548.123. It already presents the zero, strip, depth, real-part, rational comparison, transport, continuation, package, and local naming rows as the whole fixed-strip source surface.

Apply Definition 548.138 to identify the supplied zero and strip as the source rows. Apply Definition 548.139 to route every modulus-facing read through the displayed $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ rows.

The remaining rows are structural. The transport, continuation, package, and local naming coordinates preserve the same tuple rather than adding a further analytic or host-level coordinate. Thus every fixed-strip read factors through exactly $Z, S, M, R, Q, H, C, P, N$. □

Theorem 548.315 (Critical-line witness zero-route nonescape)**.** *The zero route carried by Definition 548.313 cannot become zero search, analytic continuation, host equality, quotient representative, ambient completeness, or a proof of* RH*. It remains a finite BEDC route through the supplied zero, fixed strip, depth, real-part, rational comparison, transport, continuation, package, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip zero route boundary`

Proof. First apply the fixed-strip modulus refusal theorem Definition 548.124. It keeps the modulus request inside the supplied zero and strip rows, the depth and comparison rows, and the displayed refusal route.

Next apply the fixed-strip row exhaustion theorem Definition 548.314. Any zero route that reaches a fixed-strip consumer has already been forced through $Z, S, M, R, Q, H, C, P, N$.

Finally use the RH consumer nonescape row Definition 548.125. Since the RH-facing consumer receives only the finite witness boundary, the zero route cannot acquire zero search, analytic continuation, host equality, quotient representatives, ambient completeness, or an RH proof. □

Theorem 548.316 (Critical-line witness RH consumer boundary)**.** *Every downstream* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the finite BEDC witness boundary*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may cite the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *zero row,* $\mathsf{CritStrip}^{\uparrow}$ *fixed-strip row,* $\mathsf{Nat}^{\uparrow}$ *depth row,* $\mathsf{Real}^{\uparrow}$ *real-part row,* $\mathsf{Rat}^{\uparrow}$ *rational comparison row, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows, but it does not receive an analytic theorem or a host-level zero-location certificate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer boundary`

Proof. Use the fixed-strip row exhaustion theorem to place the downstream read behind the tuple $Z, S, M, R, Q, H, C, P, N$. This supplies a finite BEDC witness boundary before the consumer can inspect any RH-facing conclusion.

Use the zero-route nonescape theorem to remove the possible escape channels. The same zero route cannot turn into zero search, analytic continuation, host equality, quotient representatives, ambient completeness, or an RH proof.

It remains only to identify what the consumer receives. The visible rows are the supplied zeta zero, fixed critical strip, depth, real-part, rational comparison, transport, continuation, package, and local naming rows. Therefore every RH-facing consumer receives exactly the finite boundary and no analytic or host-level certificate beyond it. □

Theorem 548.317 (Critical-line witness fixed-strip source totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every fixed critical-strip source read of* W *is total on the displayed source rows, zero-strip/source modulus rows, finite* BHist *observation windows, refusal ledger, transport, replay, provenance, and local naming surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The read may expose the supplied $\mathsf{ZetaZeros}^{\uparrow}$ *zero row, the supplied* $\mathsf{CritStrip}^{\uparrow}$ *strip row, the* $\mathsf{Nat}^{\uparrow}$ *depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *rational comparison row, the displayed* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *handoff. It exposes no zero-producing search, analytic-continuation theorem, host equality, quotient representative, ambient completeness witness, or proof of* RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip source totality certificate`

Proof. Project the accepted packet through the carrier of Definition 548.1. The visible coordinates are exactly the supplied zero row Z, strip row S, modulus row M, real-part row R, rational ledger Q, transport row H, continuation route C, provenance row P, and local naming row N.

Next use the fixed-strip tuple exhaustion of Definition 548.314 together with the root finite-budget source totality of Definition 548.307. These rows identify the source and modulus faces as the same finite observation window rather than as separate analytic inputs.

Finally read the refusal side through Definition 548.315. Since no other carrier coordinate is available, the displayed rows exhaust the fixed critical-strip source and the excluded analytic, host, quotient, completeness, and RH outputs have no route into the read. □

Theorem 548.318 (Critical-line witness modulus refusal ledger exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every zero-strip/source modulus request is exhausted by the finite refusal ledger over*

$$Z, S, M, R, Q, H, C, P, N.$$

The ledger records the BHist *observation window,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *handoff for the source and modulus rows. It cannot be extended by a hidden*

modulus source, analytic-continuation evidence, zero-free-strip evidence, a global zero classifier, a host numeric oracle, quotient equality, or ambient completeness.

Proof. Begin with the source totality of Definition 548.317. It fixes the only readable source and modulus coordinates at $Z, S, M, R, Q, H, C, P, N$.

Now apply the modulus refusal exhaustion row of Definition 548.252. A zero-strip modulus request is routed through the displayed BHist, hsame, Cont, Pkg, and NameCert surface, so the request remains a finite ledger entry rather than an analytic continuation claim.

It remains to rule out alternate sources. The finite-budget ledger stability of Definition 548.308 keeps the comparison row Q on the same transported route, while Definition 548.253 removes zero-free-strip evidence, global classifiers, host numeric oracles, quotient equality, ambient completeness, and RH proof data from the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus refusal ledger exhaustion`

Theorem 548.319 (Critical-line witness RH-boundary NameCert handoff)**.** *Every downstream* RH*-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet receives only the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *handoff for the fixed-strip source and modulus refusal ledger*

$$Z, S, M, R, Q, H, C, P, N.$$

The handoff may cite finite source rows, finite observation windows, componentwise hsame *transports, displayed* Cont *replay,* Pkg *provenance, and the refusal ledger. It does not deliver* RH*, analytic continuation, zero production, host zero classification, quotient presentation, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary namecert handoff`

Proof. First read the local handoff through Definition 548.2. The five obligation rows admit the carrier, bind zero-strip compatibility, discipline the modulus, enforce non-escape, and exhaust the ledger through $Z, S, M, R, Q, H, C, P, N$.

Next combine fixed-strip source totality with modulus refusal ledger exhaustion. The source face and the modulus face are the same finite packet as seen by the downstream consumer, and their transport, replay, provenance, and naming rows are already displayed.

Finally use the RH consumer boundary Definition 548.316. The downstream consumer can receive only the finite BEDC witness boundary, so the NameCert handoff cannot become analytic continuation, zero production, host classification, quotient presentation, host equality, ambient completeness, or a proof of RH. □

Theorem 548.320 (CriticalLineWitness fixed-strip window carrier)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet, the fixed critical-strip window carrier is a finite* BHist *packet over the zero-strip source rows, rational modulus rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. Its visible surface is the fixed-strip tuple*

$$Z, S, M, R, Q, H, C, P, N,$$

where Z supplies the zero-strip source row, S supplies the fixed critical-strip window, M, R, Q supply the depth, real-part, and rational comparison ledgers, and H, C, P, N supply transport, replay, provenance, and local naming.

The carrier has no coordinate for analytic continuation, a zero-producing search, host zero equality, quotient presentation, ambient completeness, or a proof of RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip window surface`

Proof. Start from the fixed-strip source tuple. It lists the zero row, strip row, depth row, real-part row, rational comparison row, transport, continuation, provenance, and local naming row as the whole fixed-strip source packet.

Apply fixed-strip source totality. Every fixed critical-strip source read is total on those displayed rows and on the finite observation window carried by the same BHist packet.

Finally use zero-route nonescape. Since the accepted route has no coordinate beyond $Z, S, M, R, Q, H, C, P, N$, analytic continuation, zero search, host equality, quotient presentation, ambient completeness, and RH proof data cannot be read from the carrier. □

Theorem 548.321 (CriticalLineWitness fixed-strip modulus ledger)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *fixed-strip window carrier, every modulus comparison is exhausted by the displayed rational-modulus and zero-strip rows*

$$Z, S, M, R, Q, H, C, P, N.$$

The comparison may read the supplied zero-strip source, fixed critical-strip window, depth row, real-part row, rational comparison ledger, componentwise transport, continuation replay, package provenance, and local naming row. It cannot import analytic continuation, zero-free-strip evidence, a global zero classifier, a host numeric oracle, quotient equality, ambient completeness, or a global RH *assertion.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip modulus ledger`

Proof. Begin with the fixed-strip window carrier above. It places every modulus-facing read on the same finite source and rational-comparison packet.

Use modulus refusal ledger exhaustion. The zero-strip/source modulus request is routed through the displayed BHist, hsame, Cont, Pkg, and NameCert surface, so the modulus comparison remains a ledger row rather than an analytic theorem.

Now apply RH-boundary nonescape through the fixed-strip source totality results. The downstream boundary receives only the displayed source and modulus rows, so analytic continuation, zero-free-strip evidence, global classifiers, host numeric oracles, quotient equality, ambient completeness, and global RH assertions are not exported. □

Theorem 548.322 (CriticalLineWitness fixed-strip RH refusal)**.** *Every downstream* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *fixed-strip window carrier receives only the displayed refusal ledger and the fixed-strip source window*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may use the zero-strip source, fixed critical-strip window, rational modulus comparison, componentwise transport, continuation replay, package provenance, and local naming rows. It receives no proof of zero exclusion, no analytic-continuation theorem, no global zero classifier, no host zero-location certificate, no quotient presentation, and no ambient-completeness witness.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip rh refusal`

Proof. First read the downstream surface through the fixed-strip modulus ledger. It confines the RH-facing request to the displayed zero-strip source, fixed-strip window, modulus comparison, and structural rows.

Next apply the RH-boundary NameCert handoff. The local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ handoff names the same fixed-strip source and modulus refusal ledger, not an analytic certificate.

Finally use the RH consumer boundary. The consumer receives only the finite BEDC witness boundary, so zero exclusion, analytic continuation, global zero classification, host zero-location certificates, quotient presentations, ambient completeness, and a proof of RH remain outside the accepted packet. □

Definition 548.323 (Critical-line witness signature-gap carrier)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Its signature-gap carrier is the finite packet readback*

$$\begin{aligned}
\mathsf{Sig}_{\mathsf{CriticalLineWitness}^{\uparrow}}(W) &= (Z, S, M, R, Q, H, C, P, N),\\
\mathsf{Gap}_{\mathsf{CriticalLineWitness}^{\uparrow}}(W) &= (Z, S) \mid (M, R, Q) \mid (H, C, P, N),\\
\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}(W) &= (Z, S, M, R, Q; \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, N).
\end{aligned}$$

The first row names the supplied zero-strip signature, the second row names the only admitted gap between source data and modulus-comparison data, and the third row records the local certificate route. All three rows are read over the same BHist*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 (definition) `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessSignatureGapCarrier`

Theorem 548.324 (Critical-line witness signature-gap exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every admitted signature-gap read of W factors through the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational comparison ledger Q, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N. Equivalently, the readback is exactly the carrier of Definition 548.323 and no further source, comparison, or naming row is available.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier signature gap exactness`

Proof. Project the accepted packet through Definition 548.1. The carrier exposes exactly $Z, S, M, R, Q, H, C, P, N$, so a signature-gap read starts from a finite BEDC tuple rather than from a host analytic object.

Use the root source triad of Definition 548.76 and the zero-strip source totality of Definition 548.180. These rows identify the supplied zero and critical-strip side with the $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ source surfaces before any modulus or comparison read is consumed.

Next apply the root comparison ledger Definition 548.77, the modulus-depth ledger Definition 548.181, and the modulus ledger totality Definition 548.275. They route the $\mathsf{Nat}^{\uparrow}$ depth, $\mathsf{Real}^{\uparrow}$ real-part, and $\mathsf{Rat}^{\uparrow}$ comparison data through M, R, Q under the same structural rows.

Finally use the transport replay obligation Definition 548.47 and the NameCert obligations of Definition 548.2. These attach hsame, Cont, Pkg, and the local NameCert row to the same carrier, so the admitted signature-gap read is exactly the displayed finite readback. □

Theorem 548.325 (Critical-line witness signature-gap non-escape)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the signature-gap readback of Definition 548.323 cannot import an* RH *proof, zero search, analytic-continuation certificate, host equality, quotient representative, or ambient completeness witness. Every downstream consumer of the readback receives only the supplied zero-strip row, critical-strip row, modulus row, rational comparison ledger, componentwise transport, continuation replay, package provenance, and local naming route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier signature gap nonescape`

Proof. Begin with the signature-gap exactness theorem above. It reduces any admitted readback to the finite tuple $Z, S, M, R, Q, H, C, P, N$ and to the displayed Sig, Gap, and NameCert rows.

Then apply the root unblock refusal boundary Definition 548.5 and the RH-refusal ledger completeness theorem Definition 548.72. These results refuse proof content, global zero classification, zero search, analytic continuation, quotient representatives, host equality, and ambient completeness from the accepted packet.

Use the source-modulus refusal totality of Definition 548.292 together with the downstream refusal export Definition 548.113. They show that downstream consumers may replay only the supplied zero-strip, modulus, comparison, transport, package, and naming rows.

Thus the signature-gap carrier is a kernel-facing readback surface, not an analytic theorem source. Its non-escape content is exactly the absence of every listed host-level coordinate from the finite packet. □

Theorem 548.326 (Critical-line witness fixed-strip obstruction exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every fixed-strip obstruction read of W factors only through the finite source-modulus row, the supplied zero-strip row, the signature-gap readback, the* BHist *carrier, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row:*

$$Z, S, M, R, Q, H, C, P, N.$$

The read cannot be supplied by analytic continuation, host real equality, a hidden zero classifier, quotient presentation, ambient completeness, or an unrecorded fixed-strip witness.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip obstruction exactness`

Proof. Project the accepted packet through Definition 548.1. The only rows available to a fixed-strip obstruction read are the displayed source rows Z, S, the modulus and comparison rows M, R, Q, and the structural rows H, C, P, N.

Use Definition 548.59, Definition 548.64, and Definition 548.67. These rows identify the source-modulus and zero-strip inputs as finite BHist data before the obstruction is read.

Next apply Definition 548.317 and Definition 548.324. The fixed-strip source and the signature-gap readback are the same finite carrier surface, transported only through H and replayed only through C.

Finally use the route exposure package Definition 548.295 together with the RH non-escape package Definition 548.84. Since the exposed route contains no analytic-continuation, host-equality, quotient, completeness, hidden-classifier, or extra witness coordinate, the fixed-strip obstruction is exact for the displayed source-modulus package. □

Theorem 548.327 (Critical-line witness zeta-consumer non-escape)**.** *Every zeta-facing or* RH-*facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet that uses the fixed-strip obstruction must factor through the same finite readback*

$$Z, S, M, R, Q, H, C, P, N.$$

Such a consumer may read the source-modulus row, zero-strip row, signature-gap carrier, BHist *history,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. It cannot import analytic continuation, a completed zeta function, host real equality, or a standard bridge row for the composite route.*

Proof. Begin with Definition 548.326. Any consumer using the fixed-strip obstruction has already been confined to the displayed finite readback.

The downstream route is then controlled by Definition 548.295 and Definition 548.297. These theorems force zeta-facing and RH-facing reads through the source, modulus, refusal, transport, replay, provenance, and naming rows already present in W.

Use the zero-strip and RH-boundary rows Definition 548.296, Definition 548.82, and Definition 548.84. They route failed or excessive zeta requests to the visible refusal side instead of to analytic continuation, completed zeta data, host equality, or proof content.

Finally apply Definition 548.325. The signature-gap carrier exports no standard bridge row for the composite route, so a downstream consumer receives exactly the finite BEDC witness boundary and no analytic or bridge-level escape. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta consumer nonescape`

Theorem 548.328 (Critical-line witness root source coverage)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted source read exposed by W is covered by the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z, the* $\mathsf{CritStrip}^{\uparrow}$ *row S, the* $\mathsf{Nat}^{\uparrow}$ *depth row M, the* $\mathsf{Real}^{\uparrow}$ *real-part row R, the* $\mathsf{Rat}^{\uparrow}$ *comparison row Q, and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *structure carried by H, C, P, N. No source read adds an analytic-continuation certificate, quotient representative, host equality, ambient completeness row, or proof of* RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source coverage`

Proof. Project the accepted packet through the carrier surface. The root source triad of Definition 548.76 exposes the supplied zero row, critical-strip row, and source-facing rational comparison row as finite data before any downstream consumer reads W.

Next attach the remaining carrier projections. The root comparison ledger Definition 548.77 accounts for M, R, Q, H, C, P, N, while the zeta dependency lattice Definition 548.169 places the zeta-facing source row on the existing $\mathsf{ZetaZeros}^{\uparrow}$ boundary rather than on a new analytic object.

The structural rows are therefore exactly the visible BHist carrier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row. The root refusal route then removes analytic continuation, quotient representatives, host equality, ambient completeness, and RH proof content from the source surface. □

Theorem 548.329 (Critical-line witness modulus-comparison transport stability)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, every modulus or rational-comparison route remains stable under componentwise* hsame *transport through the displayed H, C, P, N rows. The transported route still reads only the* $\mathsf{Nat}^{\uparrow}$ *depth row, the* $\mathsf{Real}^{\uparrow}$ *real-part row, the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, the finite* BHist *carrier, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row; it cannot import an external modulus source, analytic estimate, quotient equality, host equality, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus comparison transport stability`

Proof. Start with the root comparison ledger Definition 548.77. It places every comparison read on M, R, Q, H, C, P, N, so the comparison has no coordinate outside the accepted packet.

Now apply the root modulus ledger scope Definition 548.92. This fixes the modulus and rational-comparison surface as the complete root-facing ledger before any transport is replayed by Cont or recorded by Pkg and NameCert.

Finally use the finite-budget ledger stability row Definition 548.308. It identifies componentwise hsame transport as preservation of the same displayed finite ledger, while the refusal ledger excludes analytic estimates, quotient equality, host equality, ambient completeness, and any off-packet modulus source. □

Theorem 548.330 (Critical-line witness root refusal-boundary totality)**.** *Every* RH*-facing or zeta-facing consumer of the source and transport surface of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the finite refusal boundary over*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may read the supplied ZetaZeros$^{\uparrow}$ *and* CritStrip$^{\uparrow}$ *rows, the* Nat$^{\uparrow}$, Real$^{\uparrow}$, *and* Rat$^{\uparrow}$ *comparison surface, and the displayed* BHist/hsame/Cont/Pkg/NameCert *structure. It receives no proof of* RH*, global zero classifier, analytic-continuation theorem, quotient representative, host equality, ambient completeness row, or external zeta object.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal boundary totality`

Proof. Begin with RH-refusal ledger completeness, Definition 548.72. It already classifies requests for RH proof data, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness as refusal-side reads of the displayed packet.

Use the root RH-boundary refusal Definition 548.94 together with the zeta-consumer non-escape row Definition 548.327. The first keeps RH-facing requests on the finite refusal boundary, and the second prevents zeta-facing consumers from replacing the displayed packet by completed zeta data or an analytic continuation route.

Finally apply the RH-boundary refusal exhaustion row Definition 548.276. The source coverage and transport stability rows above account for every positive coordinate still readable by the consumer, so all remaining RH- or zeta-facing demands are total as refusals on $Z, S, M, R, Q, H, C, P, N$ and nowhere else. □

Theorem 548.331 (Critical-line witness root finite route totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every root-facing consumer of* W *reads only the supplied zero row, the critical-strip row, the modulus row, the real-part row, the rational comparison ledger, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer exports no proof of RH*, global zero classifier, analytic-continuation certificate, quotient representative, host equality, or ambient completeness datum.*

Proof. Begin with carrier route exactness and the root-source triad, Definition 548.310 and Definition 548.76. They expose the supplied zero row, critical-strip row, and source-facing rational comparison row as finite BHist data before any root-facing consumer reads the packet.

Next apply the source-normalization interface and the modulus-ledger interface, Definition 548.59 and Definition 548.60. These rows place M, R, Q downstream of the already exposed Z, S source route and attach the read to the same hsame/Cont/Pkg/NameCert surface.

Finally use the RH-refusal ledger completeness row Definition 548.72. It removes proof of RH, global zero classification, analytic continuation, quotient representatives, host equality, and ambient completeness from every root-facing export, so the route is exactly the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite route totality`

Theorem 548.332 (Critical-line witness root modulus source exactness)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every root-facing modulus or rational-comparison read is sourced by* M, R, Q *only after the supplied* Z, S *source rows have been exposed. No modulus request may be completed from a hidden analytic estimate, host equality, quotient representative, ambient completeness row, external zero search, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus source exactness`

Proof. First expose the source side by Definition 548.59 and Definition 548.76. These results place the zero row and critical-strip row in the accepted finite packet before any modulus or rational-comparison read is available to the root boundary.

Now apply the modulus-ledger interface Definition 548.60. It routes every modulus, real-part, and rational-comparison request through M, R, Q together with the displayed transport, continuation, package, and naming rows.

The exclusion of alternate sources follows from Definition 548.92 and Definition 548.292. Failed or off-route requests land in the finite refusal ledger rather than in analytic estimates, host equality, quotient representatives, ambient completeness, external zero search, or an RH proof. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootModulusSourceExactness`

Theorem 548.333 (Critical-line witness root refusal ledger non-escape)**.** *Every downstream* RH*-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet remains attached to the displayed refusal ledger over*

$$Z, S, M, R, Q, H, C, P, N.$$

The ledger cannot be bypassed by host equality, quotient representatives, zero search, analytic continuation, ambient completeness, a standard bridge, or an off-packet modulus source.

Proof. Start with the finite route totality above. It identifies the root-facing source, modulus, comparison, transport, continuation, package, and naming surface as the entire packet available to downstream consumers.

Next apply the root refusal-boundary totality Definition 548.330 and the downstream RH boundary nonescape Definition 548.312. These rows keep RH-facing reads on the finite refusal side of the same displayed packet.

Finally use Definition 548.72 and Definition 548.87. The refusal ledger has no escape coordinate for host equality, quotient representatives, zero search, analytic continuation, ambient completeness, a standard bridge, or an off-packet modulus source, so every downstream read remains attached to the displayed ledger. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal ledger nonescape`

Theorem 548.334 (Critical-line witness finite source lattice boundary)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The finite source lattice exposed by* W *is exactly the supplied zero row* Z*, critical-strip row* S*, modulus row* M*, real-part row* R*, rational comparison row* Q*,* BHist *carrier structure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row carried by* H, C, P, N*. The lattice factors through the zeta dependency lattice of Definition 548.169, the fixed-strip obstruction and zeta consumer*

rows of Definition 548.326 and Definition 548.327, and the root boundary totality rows of Definition 548.330. It exports no analytic-continuation theorem, host equality, quotient representative, ambient completeness row, or proof of RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finite source lattice boundary`

Proof. Project the accepted carrier through Definition 548.1. The finite packet already displays $Z, S, M, R, Q, H, C, P, N$, so the source lattice has no coordinate outside the BHist tuple and its structural rows.

Use the zeta dependency lattice to place the zero-facing side behind the existing $\mathsf{ZetaZeros}^{\uparrow}$ and RH-boundary surfaces. Then use the fixed-strip obstruction and zeta-consumer rows to keep fixed-strip and zeta-facing reads on the same finite packet.

Finally apply root refusal-boundary totality. Any request for analytic continuation, host equality, quotient data, ambient completeness, or an RH proof is classified as absent from the displayed source lattice. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessFiniteSourceLatticeBoundary`

Theorem 548.335 (Critical-line witness RH consumer route)**.** *Every* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet that reads the finite source lattice must factor through*

$$Z, S, M, R, Q, H, C, P, N.$$

The route may use the zeta dependency lattice, fixed-strip obstruction surface, and root boundary totality rows, but it cannot add a private zero classifier, completed zeta object, analytic-continuation certificate, host equality row, quotient representative, ambient completeness row, or proof object for RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh consumer route`

Proof. Begin with Definition 548.334. It identifies the only source lattice available to an RH-facing consumer as the displayed packet rows and the visible BHist, hsame, Cont, Pkg, and NameCert structure.

The zeta dependency lattice fixes the zero-strip side before the modulus witness is read. The fixed-strip obstruction and zeta-consumer non-escape rows then route any obstruction or zeta-facing demand through the same finite readback.

Root boundary totality closes the remaining RH-facing demands. Since proof content, analytic continuation, quotient representatives, host equality, and ambient completeness are not carrier fields, the consumer route remains a refusal-side route over $Z, S, M, R, Q, H, C, P, N$. □

Theorem 548.336 (Critical-line witness NameCert row exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows seen by finite source-lattice and* RH*-consumer reads are exhausted by carrier admission, zero-strip exposure, modulus and comparison exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No additional NameCert row can be inserted for analytic continuation, global zero classification, host equality, quotient presentation, ambient completeness, or* RH *proof export.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert row exhaustion`

Proof. First read the existing NameCert obligations through the carrier. They place carrier admission, zero-strip compatibility, modulus discipline, ledger exactness, transport, replay, provenance, and naming on the same finite packet.

The finite source lattice theorem accounts for all source-facing rows. The RH consumer route theorem accounts for every downstream RH-facing read that can consume those rows.

Thus the local NameCert surface has no further row to expose. Any attempted analytic, host-level, quotient, completeness, or proof export is a request outside $Z, S, M, R, Q, H, C, P, N$ and is refused by the same root boundary. □

Theorem 548.337 (Critical-line witness root obligation image row)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet. Before any fixed-tuple* StandardBridge*-facing request, W exposes the finite image row*

$$Z, S, M, R, Q, H, C, P, N$$

as its root obligation image: supplied zero row, critical-strip source row, modulus row, real-part row, rational comparison row, componentwise transport, continuation replay, provenance, and local NameCert row. The image row contains no RH *proof content, zero-search oracle, analytic-continuation closure, quotient stream equality, or ambient completeness witness.*

Proof. Start with the critical-line witness root source rows. The root source triad and supplied source rows expose Z and S as finite BHist data before any bridge-facing read.

Add the modulus and comparison rows. The existing modulus and root comparison surfaces place M, R, Q on the same packet as the source rows and keep them under the displayed transport, replay, provenance, and local naming rows H, C, P, N.

Finally read the refusal ledger. The RH-facing refusal rows route requests for proof content, zero search, analytic-continuation closure, quotient stream equality, and ambient completeness to the boundary rather than to the fixed tuple. Thus the image row is the displayed finite packet and nothing outside it. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root fixed tuple image row`

Theorem 548.338 (Critical-line witness root readback row)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet $W = (Z, S, M, R, Q, H, C, P, N)$, every fixed-tuple readback request must read the source side Z, S, the modulus and comparison side M, R, Q, and the support side H, C, P, N component by component. The readback may cite the root source, modulus, comparison, and refusal rows, but it cannot import* RH *proof content, zero-search, analytic-continuation closure, quotient stream equality, or ambient completeness.*

Proof. Read the source side first. The root source rows expose the supplied zero and critical-strip source as finite rows of the accepted packet.

Read the modulus and comparison side next. The modulus route and rational comparison ledger identify M, R, Q as displayed rows, so a fixed-tuple consumer has no second modulus source and no host analytic estimate.

The support side is H, C, P, N. These rows provide componentwise transport, displayed replay, package provenance, and local naming, while the refusal rows keep RH proof content, zero search, analytic continuation, quotient stream equality, and ambient completeness outside the readback. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root readback row`

Theorem 548.339 (Critical-line witness root componentwise NameCert transport)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the fixed-tuple* StandardBridge*-facing surface transports componentwise through the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. Each component of the tuple keeps its source, modulus, comparison, transport, replay, provenance, or naming role, and no transport step identifies the tuple with* RH *proof content, zero-search, analytic-continuation closure, quotient stream equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootComponentwiseNamecertTransport`

Proof. Use the root obligation image row to fix the finite tuple. It identifies the displayed image before any bridge-facing request.

Use the root readback row to read each component separately. The source rows Z, S, modulus and comparison rows M, R, Q, and support rows H, C, P, N already have distinct consumer roles.

Componentwise NameCert transport preserves those roles. Since the refusal ledger excludes proof content, zero search, analytic continuation, quotient equality, and ambient completeness, transport can only carry the finite tuple rows and cannot turn them into analytic or host-level data. □

Theorem 548.340 (CriticalLineWitness fixed-strip StdBridge route totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed zero-in-strip packet. The fixed-strip route from the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*,* $\mathsf{CritStrip}^{\uparrow}$ *row* S*,* $\mathsf{Nat}^{\uparrow}$ *depth row* M*,* $\mathsf{Real}^{\uparrow}$ *real-part row* R*,* $\mathsf{Rat}^{\uparrow}$ *comparison row* Q*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N *to the standard finite-tuple* StandardBridge *surface is total as an obligation package. Every bridge-facing read either lands in the image, readback, transport, taste-gate, and refusal rows already displayed by the packet or is refused on the same finite* BHist *boundary. The route exposes no zero search, no proof of* RH*, no host complex equality, no quotient real equality, and no ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip stdbridge route totality`

Proof. Start with the fixed tuple premise surface of Definition 548.219. It supplies the finite image row, readback row, componentwise naming transport, and ChapterTasteGate row needed by the standard fixed finite BHist tuple surface.

Attach the fixed-strip carrier rows. The carrier route exactness theorem Definition 548.310 and the fixed-strip source tuple Definition 548.313 identify the visible packet with $Z, S, M, R, Q, H, C, P, N$, with no tenth coordinate.

Finally use the root refusal boundary totality theorem Definition 548.330. It sends all RH-facing, zero-search, analytic-continuation, host-equality, quotient-equality, and completeness requests to the displayed refusal side. Hence the bridge-facing route is total exactly over the finite obligation package. □

Theorem 548.341 (CriticalLineWitness componentwise StdBridge transport)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed zero-in-strip packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the standard finite-tuple* StandardBridge *route transports componentwise through the displayed* hsame *and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}}$ *rows. The* $\mathsf{ZetaZeros}^{\uparrow}$ *coordinate* Z*,* $\mathsf{CritStrip}^{\uparrow}$ *coordinate* S*,* $\mathsf{Nat}^{\uparrow}$ *coordinate* M*,* $\mathsf{Real}^{\uparrow}$ *coordinate* R*,* $\mathsf{Rat}^{\uparrow}$ *coordinate* Q*,* BHist *carrier,* hsame *transport row* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and naming row* N *keep their roles under transport; no component is identified with an* RH

proof, a zero-search witness, host complex equality, quotient real equality, analytic continuation, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier componentwise stdbridge transport`

Proof. Use the root componentwise NameCert transport theorem Definition 548.339. It already treats the fixed tuple as a componentwise route through the local naming rows rather than as a host-level equality.

Read back the tuple with Definition 548.218. The source rows Z, S, comparison rows M, R, Q, and support rows H, C, P, N each return to their own displayed coordinate through hsame, Cont, Pkg, and NameCert structure.

Now compare this readback with the route totality package of Definition 548.340. Since every bridge-facing request is already confined to the finite BHist surface or refused there, componentwise transport cannot collapse the rows into RH content, zero search, host equality, quotient equality, analytic continuation, or completeness data. □

Theorem 548.342 (CriticalLineWitness RH-boundary StdBridge nonescape)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed zero-in-strip packet routed to the standard finite-tuple* StandardBridge *surface. Every downstream* RH*-boundary consumer of this route receives only the displayed finite obligation package over* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$. *The consumer may inspect the fixed-strip route, componentwise transport, and refusal ledger, but it receives no proof of* RH*, no zero-search method, no host complex equality, no quotient real equality, no analytic-continuation theorem, and no ambient completeness witness.*

Proof. Begin with the fixed-strip StdBridge route totality theorem Definition 548.340. It says that the bridge-facing route is exhausted by the finite image, readback, transport, taste-gate, and refusal rows of W.

Next apply componentwise StdBridge transport Definition 548.341. Transport preserves the row roles for $Z, S, M, R, Q, H, C, P, N$ and does not identify any row with host or quotient equality.

Finally use the downstream RH boundary nonescape theorem Definition 548.312 together with root refusal boundary totality Definition 548.330. These rows place every RH-facing demand on the same refusal boundary, so the downstream consumer cannot escape the standard finite-tuple package to obtain RH proof content, zero search, analytic continuation, host equality, quotient equality, or completeness data. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary stdbridge nonescape`

Theorem 548.343 (Critical-line witness modulus consumer coverage)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted critical-line consumer route that uses the source-modulus row is covered by the finite* BHist *source Z, S, the zero-strip witness, the rational modulus ledger M, R, Q, the componentwise* hsame *transport row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The route exposes those BEDC rows before any downstream read and exposes no proof of* RH*, no analytic-continuation theorem, and no completed zero classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus consumer coverage`

Proof. Begin with the root finite readback totality row Definition 548.346. It places every root readback on the displayed packet rows $Z, S, M, R, Q, H, C, P, N$ together with their finite BHist, hsame, Cont, Pkg, and local NameCert surface.

Next attach the source-modulus interface. The root source-modulus admission row Definition 548.64 and the root modulus classifier exactness row Definition 548.347 force the source side Z, S and the ledger side M, R, Q to be consumed through the same transport, replay, provenance, and naming rows.

Finally use the zeta dependency lattice Definition 548.169. It keeps the zeta-facing source as a dependency on the displayed ZetaZeros$^{\uparrow}$ and RH-boundary surfaces rather than as an analytic-continuation object. Hence a critical-line consumer using the source-modulus row sees exactly the finite BEDC coverage stated above. □

Theorem 548.344 (Critical-line witness zero-strip readback separation)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, zero-strip readback cannot be consumed as a proof of* RH*, a host analytic-continuation theorem, or a completed zero classification unless the displayed source-modulus rows* $Z, S, M, R, Q, H, C, P, N$ *and the fixed-strip rows of Definition 548.313 are present. Without those rows the readback remains a refusal-side BEDC boundary and has no consumer coordinate for RH proof content, analytic continuation, host zero search, quotient equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero strip readback separation`

Proof. Read the zero-strip side first. The zero-strip readback totality theorem Definition 548.209 routes the readback through the supplied zero and strip rows and through H, C, P, N.

Attach the source-modulus rows. The modulus consumer coverage theorem above and the fixed-strip source readback totality row Definition 548.268 require the displayed M, R, Q ledger and the fixed-strip tuple before a downstream consumer can read the zero-strip package.

It remains to exclude the forbidden interpretations. The root refusal boundary totality theorem Definition 548.330 and the zeta dependency lattice Definition 548.169 classify RH proof requests, analytic-continuation requests, host zero-search requests, quotient-equality requests, and completed-zero-classification requests as refusal-side demands unless the displayed BEDC rows are present. Thus the zero-strip readback is separated from those stronger consumers. □

Theorem 548.345 (Critical-line witness fixed-strip bridge consumer totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *fixed zero-in-strip packet. The fixed-strip bridge consumer surface is exhausted by the finite witness rows named in Definition 548.337, Definition 548.338, Definition 548.339, and Definition 548.340. Every consumer either reads one of those image, readback, componentwise transport, source-modulus, fixed-strip, provenance, naming, or refusal rows, or is refused on the same finite* BHist *boundary. No consumer receives an* RH *proof, host analytic continuation, global zero classification, host equality, quotient equality, or ambient completeness.*

Proof. Start with the root fixed-tuple bridge rows. The image, readback, and componentwise NameCert transport theorems identify the finite rows visible to a bridge-facing consumer before any standard bridge route is read.

Now apply the fixed-strip StdBridge route totality theorem Definition 548.340. It says that the bridge-facing route is total exactly over the finite obligation package built from $Z, S, M, R, Q, H, C, P, N$ and the refusal side of the same packet.

Finally combine the modulus consumer coverage and zero-strip readback separation theorems above. They account for the source-modulus and zero-strip consumers that feed the fixed-strip

bridge surface, while the root refusal boundary totality theorem refuses RH proof content, analytic continuation, global zero classification, host equality, quotient equality, and completeness outside the displayed packet. Therefore the fixed-strip bridge consumer surface is exhausted by the named finite witness rows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip bridge consumer totality`

Theorem 548.346 (Critical-line witness root finite readback totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root finite readback exposed by W is read through exactly the displayed packet rows $Z, S, M, R, Q, H, C, P, N$ together with their finite* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *surface. No root readback imports an analytic-continuation row, host equality, quotient representative, ambient completeness coordinate, global zero-search coordinate, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root finite readback totality`

Proof. Begin with the root source triad Definition 548.76. It exposes the supplied zero row Z, the critical-strip row S, and the source-facing comparison row as finite packet data before any downstream consumer reads W.

Next attach the comparison and dependency rows. The root comparison ledger Definition 548.77 accounts for M, R, Q, H, C, P, N, while the zeta dependency lattice Definition 548.169 keeps zeta-facing reads on the existing $\mathsf{ZetaZeros}^{\uparrow}$ and boundary surfaces rather than on a new analytic object.

Finally apply root refusal boundary totality Definition 548.330. The positive rows have already been exhausted by $Z, S, M, R, Q, H, C, P, N$, so every remaining request is refused on the displayed finite boundary and cannot be repackaged as analytic continuation, host equality, quotient data, ambient completeness, zero search, or RH proof content. □

Theorem 548.347 (Critical-line witness root modulus classifier exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. After the zero-strip source rows Z and S have been supplied, the modulus and rational comparison classifier of W is exhausted by the displayed rows M, R, Q, H, C, P, N. The classifier reads the* $\mathsf{Nat}^{\uparrow}$ *depth,* $\mathsf{Real}^{\uparrow}$ *real-part,* $\mathsf{Rat}^{\uparrow}$ *comparison ledger, transport, continuation, provenance, and local naming rows, and it has no additional coordinate for zero search, analytic continuation, quotient representatives, host equality, ambient completeness, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus classifier exactness`

Proof. First use the root source triad Definition 548.76. It supplies the zero-strip source side Z, S and fixes the source-facing comparison entry as a finite row of the accepted packet.

Now apply the root comparison ledger Definition 548.77. Once Z and S are available, every modulus or rational comparison read is routed through M, R, Q and carried by H, C, P, N.

The zeta dependency lattice Definition 548.169 prevents the classifier from being replaced by a completed zeta or analytic-continuation object. Combining that lattice with the root refusal route Definition 548.78 sends every off-packet classifier demand to refusal, so the displayed modulus and comparison rows are exact. □

Theorem 548.348 (Critical-line witness root RH consumer refusal)**.** *Every* RH*-facing consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet receives only the displayed finite refusal boundary over*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may read the supplied zero-strip, critical-strip, modulus, real-part, rational-comparison, transport, continuation, package, and naming rows. It receives no RH *proof, zero search, analytic-continuation row, quotient representative, host equality, or completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root rh consumer refusal certificate`

Proof. Start from the root refusal route Definition 548.78. It classifies RH-facing requests as finite refusal reads rather than as analytic proof surfaces.

Next combine the root source triad and the root comparison ledger, Definition 548.76 and Definition 548.77. These rows account for every positive packet coordinate visible to the consumer, namely $Z, S, M, R, Q, H, C, P, N$.

Finally use the zeta dependency lattice Definition 548.169 together with root refusal boundary totality Definition 548.330. The lattice supplies only dependency information over the existing zeta-facing surfaces, and the boundary theorem refuses all remaining RH proof, zero-search, analytic-continuation, quotient, host-equality, and completeness demands on the displayed finite boundary. □

Theorem 548.349 (Critical-line witness root budget-selector totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *root packet, and let* B *be an accepted finite observation-budget selector row for the same root readback surface. Every root consumer that reads* W *through* B *is exhausted by the displayed* BHist *rows* Z, S, M, R, Q*, the componentwise* hsame *support* H*, the displayed* Cont *route* C*, the* Pkg *provenance* P*, and the local* NameCert *row* N*. The selector may bound finite requests, but it cannot add host* RH *truth, analytic continuation, ambient completeness, or a hidden zero-location selector.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root budget selector totality`

Proof. Begin with root finite readback totality, Definition 548.346. It identifies the positive root readback surface as the finite packet $Z, S, M, R, Q, H, C, P, N$.

Attach the selector side through Definition 782.7 and Definition 782.14. Those rows bound the observation request and keep selector-facing reads inside finite BHist, hsame, Cont, Pkg, and NameCert support.

Finally apply the root refusal boundary, Definition 548.330. Every request for host RH truth, analytic continuation, ambient completeness, or hidden zero-location data is refused outside the displayed packet. □

Theorem 548.350 (Critical-line witness root strip-window exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* CriticalLineWitness$^{\uparrow}$ *root packet whose finite selector route displays the same fixed-strip window rows as the root source surface. Then every fixed-strip or zero-strip readback over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *factors through* $Z, S, M, R, Q, H, C, P, N$ *before any refusal row is exported. No consumer may replace the fixed-strip window by analytic continuation, ambient completeness, host* RH *truth, or an unrecorded zero-location selector.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip window exhaustion`

Proof. Use the fixed-strip source rows first. The fixed-strip source totality and source readback theorems, Definition 548.317 and Definition 548.268, place strip-facing reads on the displayed root packet.

Now route the window through the finite selector. By Definition 548.349, any selector-bound root consumer keeps the same BHist, hsame, Cont, Pkg, and NameCert support rows.

The exhaustion claim follows from the fixed-strip row exhaustion theorem Definition 548.314. Its refusal side excludes off-packet strip data, and the selector route adds no analytic-continuation, completeness, host-truth, or hidden-zero coordinate. □

Theorem 548.351 (Critical-line witness root refusal-ledger pullback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *root packet, and suppose a finite observation-budget selector and an RH-boundary sibling read the same displayed root rows over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Then their refusal ledgers pull back to the same finite packet $Z, S, M, R, Q, H, C, P, N$: every refused request for host* RH *truth, analytic continuation, ambient completeness, or hidden zero-location selection is recorded on that packet-local ledger and nowhere else.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root refusal ledger pullback`

Proof. Start with the root refusal consumer exactness theorem Definition 548.101. It makes refusal-facing consumers read the displayed finite root packet rather than a host analytic object.

Attach the RH boundary sibling through Definition 548.90 and Definition 548.348. These rows keep every RH-facing request on the same root refusal surface.

Finally apply Definition 548.349. The finite observation-budget selector bounds the read but does not change the refusal ledger. Hence the selector-facing and RH-boundary-facing refusal rows pull back to the same packet-local BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 548.352 (Critical-line witness root source totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. Every finite source row available at the root is displayed by Z, S, M, R, Q and is carried by the same* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support rows H, C, P, N. No root source row is supplied by analytic continuation, host equality, quotient representatives, an ambient completeness coordinate, global zero search, or an* RH *truth claim.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root source totality`

Proof. Begin with the supplied root source rows. The root source triad of Definition 548.76 exposes the zero row, critical-strip row, and source-facing comparison entry as finite packet data before downstream consumers inspect W.

Next use the finite readback totality row Definition 548.346. It says that root readbacks are already exhausted by $Z, S, M, R, Q, H, C, P, N$.

Finally attach the refusal boundary. The root refusal boundary totality theorem Definition 548.330 sends every off-packet request to the displayed finite refusal surface, so no analytic-continuation, host-equality, quotient, completeness, zero-search, or RH truth coordinate can serve as a source row. □

Theorem 548.353 (Critical-line witness root modulus separation)**.** *For an accepted* CriticalLineWitness$^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the zero-strip source rows Z, S and the modulus/classifier rows M, R, Q are separated by the displayed finite route. A consumer may read M, R, Q only after the source rows have been supplied, transported through H, and replayed through C; it may not fold*

the modulus row into a zero-search oracle, analytic-continuation row, quotient representative, host equality test, ambient completeness witness, or RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root modulus separation`

Proof. First read the root source totality just established in Definition 548.352. The source side of the accepted packet is finite and exhausted before any classifier demand is considered.

Now apply the modulus classifier exactness theorem Definition 548.347. Once Z and S are present, every modulus and rational-comparison read is routed through M, R, Q and the support rows H, C, P, N.

The zeta dependency lattice Definition 548.169 keeps the zeta-facing dependency surface distinct from the local modulus classifier. Combining that lattice with the root refusal route Definition 548.78 rejects every demand that treats the modulus as zero search, analytic continuation, quotient data, host equality, ambient completeness, or an RH proof. □

Theorem 548.354 (Critical-line witness root downstream totality)**.** *Every downstream critical-line, zeta-boundary, or* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the displayed finite packet*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may read the finite source rows, zero-strip row, modulus/classifier rows, rational comparison ledger, transport rows, continuation routes, provenance, and local naming row. It receives no analytic-continuation object, host equality, quotient representative, ambient completeness coordinate, global zero-search row, or RH *truth claim.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root downstream totality`

Proof. Start with root source totality from Definition 548.352. It exhausts the positive source rows of the accepted packet by $Z, S, M, R, Q, H, C, P, N$.

Next apply root modulus separation from Definition 548.353. The classifier and rational-comparison rows are downstream of the displayed zero-strip source route, not substitutes for an analytic or global zero-search surface.

Finally combine root RH consumer refusal Definition 548.348 with the zeta dependency lattice Definition 548.169. These rows route downstream consumers through the finite packet and refuse every remaining demand for analytic continuation, host equality, quotient representatives, ambient completeness, global zero search, or RH truth. □

Theorem 548.355 (Critical-line witness source modulus terminality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted source-modulus read of* W *terminates at the displayed finite source rows* Z, S*, the modulus and rational-comparison rows* M, R, Q*, and the support rows* H, C, P, N*. Thus the read is carried by the finite* BHist *source, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows already present in the packet.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus terminal determinacy`

Proof. Project the carrier in Definition 548.1. Its accepted packet coordinates are the finite source rows Z, S, the modulus and comparison rows M, R, Q, and the structural rows H, C, P, N.

Apply the source-modulus refusal totality theorem Definition 548.292 and the window-modulus exhaustion theorem Definition 548.298. Together they force every source-window read that asks

for the modulus to pass through the displayed zero-strip source, rational-height modulus, BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally use root finite readback totality Definition 548.346. Since every root readback is exhausted by $Z, S, M, R, Q, H, C, P, N$, there is no further source coordinate at which a source-modulus read can continue. The accepted read therefore terminates at the displayed finite packet surface. □

Theorem 548.356 (Critical-line witness source-window RH boundary)**.** *Every* RH*-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet must pass through the displayed source-window route and zero-strip boundary before using the critical-line witness. In particular, any such consumer reads the finite packet*

$$Z, S, M, R, Q, H, C, P, N$$

through the visible source rows, the zero-strip source boundary, the componentwise hsame *transport, the displayed* Cont *and* Pkg *rows, and the local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row.*

Proof. First apply source-window route lock Definition 548.186. It places the consumer behind the visible source-window route before any downstream critical-line use is permitted.

Next use zero-strip consumer boundary Definition 548.143 together with root RH source lock Definition 548.258. These rows require the consumer to cross the zero-strip boundary and the displayed source support before it can make an RH-facing request.

Finally compose with root downstream totality Definition 548.354. The downstream consumer is therefore routed through $Z, S, M, R, Q, H, C, P, N$ and cannot use the critical-line witness before the source-window, zero-strip, hsame, Cont, Pkg, and NameCert rows have been exposed. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source window rh boundary`

Theorem 548.357 (Critical-line witness finite source route saturation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the finite source route is exhausted by the displayed source, modulus, rational-comparison,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. The route exports no hidden analytic continuation, global zero oracle, host equality test, ambient quotient representative, or proof of* RH*.*

Proof. Begin with root source totality Definition 548.352. It identifies every finite source row visible at the root with the displayed packet coordinates $Z, S, M, R, Q, H, C, P, N$.

Then attach source modulus terminality from Definition 548.355. The source-modulus reads cannot pass beyond the finite BHist, hsame, Cont, Pkg, and local NameCert support rows of the packet.

It remains to rule out hidden exports. Apply root RH consumer refusal Definition 548.348 and root refusal boundary totality Definition 548.330. These rows refuse the off-route demands for analytic continuation, global zero search, host equality, quotient data, ambient completeness, and RH proof content, so the finite source route is saturated by the displayed rows alone. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finite source route saturation`

Theorem 548.358 (Critical-line witness classifier exactness)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The classifier available at the root is determined only by the supplied zero row* Z*, the critical-strip row* S*, and the displayed support rows* H, C, P, N*. It cannot be replaced by host equality, zero search, analytic continuation, quotient representatives, or ambient completeness; any accepted classifier read must remain on the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *surface displayed by* W*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier classifier exactness`

Proof. Project the carrier of Definition 548.1. Its source side displays exactly the supplied zero row Z, the supplied strip row S, and the support rows H, C, P, N before any public critical-line read is available.

Apply the NameCert obligations of Definition 548.2. The zero-strip compatibility row binds Z and S through displayed hsame transport, while carrier admission and non-escape keep the classifier read inside the same finite packet.

Finally use the root source triad Definition 548.76 and the root unblock refusal boundary Definition 548.5. They expose the finite source rows and refuse host equality, zero search, analytic continuation, quotient representatives, and ambient completeness as alternate classifier sources. Thus the classifier is exact only through Z, S, H, C, P, N. □

Theorem 548.359 (Critical-line witness ledger route stability)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The modulus and comparison ledger route through M, R, Q is stable under the displayed* hsame *transports,* Cont *replay rows,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. Every accepted comparison read therefore uses the same packet-local route M, R, Q, H, C, P, N and does not branch to a host comparison, quotient representative, analytic continuation row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier ledger route stability`

Proof. Read the modulus and comparison coordinates from Definition 548.1. The packet lists M as the depth-to-line modulus row, R as the consumed real-part row, and Q as the rational comparison ledger.

Use Definition 548.2. Its modulus discipline and ledger-exactness rows attach M, R, Q to the same displayed H, C, P, N support, so comparison reads are transported, replayed, recorded, and named inside the accepted packet.

It remains to rule out alternate ledger routes. The root source triad Definition 548.76 keeps the source side finite, and the root unblock refusal boundary Definition 548.5 refuses the host, quotient, analytic-continuation, and completeness replacements. Hence the M, R, Q ledger route is stable exactly under the displayed hsame/Cont/Pkg/NameCert rows. □

Theorem 548.360 (Critical-line witness downstream consumer unblock)**.** *Every downstream* RH*-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet may cite only the finite surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may use this surface as the supplied zero row, supplied strip row, modulus row, real-part row, rational comparison ledger, transport row, continuation row, provenance row, and local naming row; it may not cite an RH *proof, a global zero classifier, a zero-producing search, host equality, analytic continuation, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier downstream consumer unblock`

Proof. Start from the carrier definition Definition 548.1. It makes $Z, S, M, R, Q, H, C, P, N$ the visible packet and withholds every global analytic or host-level coordinate from the carrier.

Next apply Definition 548.2. Carrier admission, non-escape, and ledger exactness say that accepted public reads are exhausted by the displayed packet rows and their local NameCert support.

Finally combine the root source triad Definition 548.76 with the root unblock refusal boundary Definition 548.5. The triad exposes the finite zero-strip and comparison source before downstream use, and the refusal boundary removes every off-packet RH, zero-search, host-equality, analytic-continuation, quotient, and completeness demand. Therefore downstream consumers are unblocked precisely by the finite $Z, S, M, R, Q, H, C, P, N$ surface. □

Theorem 548.361 (CriticalLineWitness source-signature readback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted source-signature readback of W reads the critical-strip source only through the displayed finite source rows Z, S, the signature and gap rows of Definition 548.323, and the zeta,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *dependency routes recorded by M, R, Q, H, C, P, N. No accepted readback can replace those rows by an unledgered critical-strip conclusion, analytic-continuation certificate, host zero witness, quotient representative, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source signature readback`

Proof. Begin with the signature-gap carrier Definition 548.323. It identifies the source signature as the same finite packet $Z, S, M, R, Q, H, C, P, N$ and places the gap row between Z, S and M, R, Q under the displayed BHist, hsame, Cont, Pkg, and local NameCert surface.

Apply the signature-gap exactness theorem Definition 548.324. It forces every admitted signature-gap read to factor through the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison ledger, transport, replay, provenance, and naming rows.

Finally use the zeta dependency lattice Definition 548.169 together with the root modulus readback row Definition 548.129. These rows place the zeta-facing, $\mathsf{Real}^{\uparrow}$-facing, and $\mathsf{Rat}^{\uparrow}$-facing dependencies behind the displayed packet route, so the source-signature readback has no coordinate for a hidden critical-strip theorem or host analytic witness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source signature gap readback`

Theorem 548.362 (CriticalLineWitness GAP-policy stability)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Transport by the displayed* hsame *row H and replay by the displayed* Cont *row C preserve the refusal rows and the gap-policy rows of Definition 548.323. They do not convert an absent zero row, a refused* RH *request, or a missing analytic-continuation route into a zero witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier gap policy stability`

Proof. Read the policy row from Definition 548.323. The gap separates the source side Z, S from the modulus and comparison side M, R, Q while keeping both sides under the same transport, replay, provenance, and naming rows.

Use Definition 548.325. It records that the signature-gap readback cannot import an RH proof, zero search, analytic-continuation certificate, host equality, quotient representative, or ambient completeness witness.

Now apply the ledger route stability theorem Definition 548.359. Since H, C, P, N preserve the same M, R, Q comparison route, transport and replay can only move the displayed gap-policy row along the accepted packet route.

The refusal side is therefore stable as refusal. The root unblock refusal boundary Definition 548.5 keeps refused RH, analytic, host, quotient, and completeness requests outside the packet, so hsame/Cont transport cannot turn those refusals into a zero witness. □

Theorem 548.363 (CriticalLineWitness NameCert route exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every accepted consumer route factors through the displayed* BHist *carrier,* Cont *replay row,* hsame *transport row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The route can consume only the source, signature, gap, modulus, real-part, and rational-comparison rows already displayed in* W*; it cannot read an unledgered* RH *conclusion or an unledgered critical-strip conclusion.*

Proof. Start from the accepted carrier and the NameCert obligation surface. The carrier Definition 548.1 exposes only $Z, S, M, R, Q, H, C, P, N$, and Definition 548.2 attaches the local naming obligations to those same rows.

Next combine the route readback exhaustion theorem Definition 548.154 with the downstream consumer unblock theorem Definition 548.360. Together they say that every downstream consumer reads the finite surface already displayed by the packet and cannot branch to host equality, analytic continuation, quotient representatives, ambient completeness, global zero classification, or an RH proof.

It remains to include the source-signature and gap-policy rows. The source-signature readback theorem above confines the critical-strip source to the displayed finite signature route, and the GAP-policy stability theorem above keeps refusals stable under H and C.

Thus every accepted route factors through BHist, Cont, hsame, Pkg, and NameCert over the displayed packet rows. Any route that asks for an unledgered RH or critical-strip conclusion is not an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ consumer route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert route exhaustion`

Theorem 548.364 (CriticalLineWitness source-modulus public readiness)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The source-modulus public surface of* W *is ready exactly when the supplied zero row* Z*, fixed-strip source row* S*, modulus row* M*, real-part row* R*, rational comparison ledger* Q*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row* N *are all readable as one finite* BHist *packet. The readiness route is the one already exposed by the CriticalLineWitness NameCert obligations, the fixed-strip source totality rows, the zeta dependency lattice, and the signature-gap readback surface.*

This public route exports source-modulus data only through $Z, S, M, R, Q, H, C, P, N$*. It exports no analytic continuation theorem, host equality, quotient representative, ambient completeness coordinate, zero-search oracle, selected host limit, or* RH *proof content.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus public readiness`

Proof. Begin with the carrier and naming surface. The carrier of Definition 548.1 and the CriticalLineWitness NameCert obligations of Definition 548.2 expose exactly the rows $Z, S, M, R, Q, H, C, P, N$ and attach the local name to that finite route.

Next use the fixed-strip source rows. Fixed-strip source totality and fixed-strip source readback totality place the strip-facing source inside the same BHist, hsame, Cont, Pkg, and NameCert packet, so the public source-modulus read cannot split the source from the modulus row.

Finally combine the zeta dependency lattice Definition 548.169 with the signature-gap readback theorem Definition 548.361. These rows keep zeta-facing and signature-facing reads behind $Z, S, M, R, Q, H, C, P, N$, leaving no displayed coordinate for analytic continuation, host equality, quotient representatives, ambient completeness, zero search, selected host limits, or RH proof content. □

Theorem 548.365 (CriticalLineWitness source-modulus consumer non-escape)**.** *Every public consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *source-modulus packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *factors through the source-modulus public readiness route of Definition 548.364. The consumer may read the supplied zero row, fixed-strip source row, modulus row, real-part row, rational comparison ledger, transport, replay, provenance, and local naming row, but no consumer route escapes to analytic continuation, host equality, quotient representatives, ambient completeness, zero search, selected host limits, or* RH *proof content.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus consumer nonescape`

Proof. First apply Definition 548.364. It identifies the public source-modulus surface with the single finite packet $Z, S, M, R, Q, H, C, P, N$ and records the fixed-strip, zeta dependency, and signature-gap readback anchors that support that packet.

Now use the existing route exhaustion rows. The source-modulus refusal totality theorem Definition 548.292 and the NameCert route exhaustion theorem Definition 548.363 say that downstream consumers remain on the displayed BHist, hsame, Cont, Pkg, and NameCert route.

Thus a consumer can only replay the displayed source-modulus rows. Requests for analytic continuation, host equality, quotient representatives, ambient completeness, zero search, selected host limits, or RH proof content have no carrier row and therefore cannot form an accepted consumer route. □

Theorem 548.366 (CriticalLineWitness source-modulus NameCert readiness)**.** *The local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}}$ *row* N *certifies the source-modulus public surface of an accepted packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *precisely over the finite route displayed in Definition 548.364 and protected by Definition 548.365. The certificate binds the supplied zero row, fixed-strip source row, modulus row, real-part row, rational comparison ledger, transport, replay, and provenance rows as one public source-modulus package.*

Consequently the NameCert readiness claim is a finite BEDC obligation over BHist, hsame, Cont, Pkg, *and* NameCert, *not a claim of analytic continuation, host equality, quotient representative selection, ambient completeness, zero search, selected host limits, or* RH *proof content.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus namecert readiness`

Proof. Start with the local naming obligations in Definition 548.2. They attach N to the displayed carrier rows and make the local name depend on the same transport, replay, and provenance coordinates as the source-modulus route.

Next use public readiness and consumer non-escape. The readiness theorem supplies the public packet $Z, S, M, R, Q, H, C, P, N$, while the non-escape theorem blocks every consumer route outside that packet. Hence the local NameCert row certifies the public source-modulus surface only through the displayed finite route.

The remaining exclusions follow from the same row inventory. Since no row of W carries analytic continuation, host equality, quotient representatives, ambient completeness, zero search, selected host limits, or RH proof content, the local NameCert readiness surface cannot certify any of those objects. □

Theorem 548.367 (CriticalLineWitness route classifier exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted classifier read of* W *is exhausted by the finite zero-strip, modulus, comparison, transport, continuation, provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$

rows

$$Z, S, M, R, Q, H, C, P, N.$$

The read has no accepted coordinate for host equality, zero search, analytic-continuation certificates, quotient representatives, ambient completeness, a global zero classifier, or a proof of RH.

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRouteClassifierExhaustion`

Proof. Start with classifier exactness, Definition 548.358. It identifies the root classifier with the supplied zero row Z, the supplied critical-strip row S, and the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ surface carried by H, C, P, N.

Next apply ledger route stability, Definition 548.359. The modulus, real-part, and rational-comparison rows M, R, Q remain attached to the same transport, replay, provenance, and naming route, so the classifier cannot branch to an off-packet comparison or host representative.

Finally combine downstream consumer unblock, Definition 548.360, with source-signature readback, Definition 548.361. These rows make the finite packet the whole admitted source and consumer surface. Hence every accepted classifier read is exhausted by $Z, S, M, R, Q, H, C, P, N$ and every analytic, host, quotient, completeness, global zero-classification, or RH demand is refused outside the route. □

Theorem 548.368 (CriticalLineWitness fixed-strip zeta consumer separation)**.** *For an accepted fixed-strip* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, fixed-strip source reads and zeta-facing downstream consumers remain separated by the displayed signature, gap, and ledger rows. The fixed-strip side may read the supplied zero and strip rows* Z, S*, while the zeta-facing consumer may read only the modulus, comparison, transport, continuation, provenance, and naming rows* M, R, Q, H, C, P, N *already displayed by the same packet. No accepted consumer route imports an* RH *proof or analytic-continuation certificate across that separation.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessFixedStripZetaConsumerSeparation`

Proof. Read the source side through Definition 548.361. It places the critical-strip source behind the displayed finite signature route and the gap row, rather than behind a hidden analytic-continuation or host-zero coordinate.

Apply gap-policy stability, Definition 548.362. The displayed hsame transport and Cont replay preserve refusal rows as refusal rows, so moving along H and C cannot turn a missing analytic or RH coordinate into accepted source data.

Now use the fixed-strip zeta consumer boundary, Definition 548.375, and ledger route stability, Definition 548.359. The zeta-facing consumer receives the refusal budget and rational comparison ledger through M, R, Q, H, C, P, N on the same finite route.

Thus the fixed-strip source and downstream zeta consumer meet only at the displayed signature, gap, and ledger rows. The route supplies no RH proof, analytic-continuation certificate, host equality, quotient representative, or ambient-completeness datum. □

Theorem 548.369 (CriticalLineWitness finite NameCert consumer route)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *consumer route factors through the displayed* BHist *carrier,* hsame *transport row,* Cont *replay row,* Pkg *provenance row, and local naming row of* W*. The route may consume only the local source, signature, gap, modulus, comparison, and refusal rows already named by* $Z, S, M, R, Q, H, C, P, N$*.*

Proof. Begin with NameCert route exhaustion, Definition 548.363. It already forces accepted consumer routes through the packet-local BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ surface.

Attach the classifier and ledger rows. Classifier exactness Definition 548.358 binds classifier reads to Z, S, H, C, P, N, while ledger route stability Definition 548.359 binds comparison reads to M, R, Q, H, C, P, N.

It remains to include the source and downstream interfaces. Source-signature readback Definition 548.361 keeps the source behind the displayed signature and gap rows, and downstream consumer unblock Definition 548.360 confines every consumer to the finite packet surface.

Therefore every accepted NameCert consumer route factors through the displayed BHist, hsame, Cont, Pkg, and local naming rows. A route that asks for host equality, analytic continuation, quotient representatives, ambient completeness, global zero classification, or an RH proof is outside the accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ consumer route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessFiniteNameCertConsumerRoute`

Theorem 548.370 (Critical-line witness root classifier readiness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ root packet. The root classifier is ready exactly on the displayed finite* BHist *route through the source row Z, strip row S, transport row H, continuation row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ row N. Componentwise* hsame *transport and* Cont *replay preserve that classifier surface, while* Pkg *provenance records it as part of the same accepted packet. No host zero search, analytic-continuation certificate, quotient representative, ambient completeness witness, or proof of* RH *supplies an alternate classifier.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root classifier readiness`

Proof. Project the root source and zero-strip rows already exposed by the carrier and root source files. The source row Z and strip row S are the classifier inputs before any downstream read is available.

Use the root source, modulus, and zero-strip obligation rows from the sibling files. They bind the classifier to the displayed BHist packet and preserve it under hsame transport and Cont replay along H and C.

Finally attach the package and local name. The refusal and downstream-readback rows keep host zero search, analytic continuation, quotient representatives, ambient completeness, and RH proof objects outside the accepted route, so the classifier is ready precisely on Z, S, H, C, P, N. □

Theorem 548.371 (Critical-line witness root ledger readiness)**.** *For an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ root packet $W = (Z, S, M, R, Q, H, C, P, N)$, the modulus and comparison ledger is ready exactly through M, R, Q, H, C, P, N. The row M supplies the root modulus budget, R supplies the consumed real-part row, Q supplies the rational comparison ledger, H records componentwise* hsame *transport, C records the displayed* Cont *route, P records* Pkg *provenance, and N records the local* NameCert *row. The ledger has no accepted coordinate for host comparison, quotient equality, analytic continuation, ambient completeness, or an* RH *certificate.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root ledger readiness`

Proof. Read the modulus rows first. The sibling root modulus and zero-strip package files display M as the finite budget through which the strip source can be consumed.

Next read the comparison rows. The existing downstream-readback and classifier ledger files attach R and Q to the same route and keep them stable under the displayed hsame transport and Cont replay.

The package row and local name then close the ledger surface. Because the refusal rows exclude host comparisons, quotient equality, analytic continuation, ambient completeness, and RH proof data, every accepted ledger read is the finite route M, R, Q, H, C, P, N. □

Theorem 548.372 (Critical-line witness root consumer readiness)**.** *Every root consumer of an accepted* CriticalLineWitness$^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *is ready on the combined source, classifier, ledger, and naming surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer reads this surface through the displayed BHist *carrier, componentwise* hsame *row,* Cont *route,* Pkg *provenance, and local* NameCert *row. It cannot request a hidden zero source, off-packet modulus, global zero classifier, analytic-continuation certificate, quotient representative, ambient completeness theorem, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root consumer readiness certificate`

Proof. First combine the source and classifier surfaces. The displayed source, zero-strip, and refusal rows make Z, S, H, C, P, N the only classifier-ready root route.

Next combine the modulus and ledger surfaces. The modulus, real-part, and rational-comparison rows make M, R, Q, H, C, P, N the only ledger-ready route for downstream reads.

Finally intersect the two routes at the shared structural rows. Both are carried by the same BHist packet, transported by the same hsame row, replayed by the same Cont route, recorded by the same Pkg provenance, and named by the same local NameCert row. The refusal and downstream-readback siblings exclude every off-packet analytic, quotient, completeness, zero-search, and RH demand, so the root consumer is ready exactly on $Z, S, M, R, Q, H, C, P, N$. □

Definition 548.373 (CriticalLineWitness fixed-strip refusal-budget packet)**.** *A* CriticalLineWitness$^{\uparrow}$ *fixed-strip refusal-budget packet is a finite* BHist *packet*

$$B = (Z, S, M, R, Q, H, C, P, N)$$

over the supplied ZetaZeros$^{\uparrow}$ *zero row* Z*,* CritStrip$^{\uparrow}$ *critical-strip row* S*,* Nat$^{\uparrow}$ *line-modulus row* M*,* Real$^{\uparrow}$ *real-part row* R*, finite refusal budget and rational comparison ledger* Q*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert$_{\text{CriticalLineWitness}^{\uparrow}}$ *row* N*. The packet has no coordinate for zero search,* RH *proof, analytic-continuation certificate, host equality, quotient representative, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip refusal budget packet`

Theorem 548.374 (CriticalLineWitness fixed-strip budget route determinacy)**.** *Every fixed-strip budget read of an accepted* CriticalLineWitness$^{\uparrow}$ *packet factors through the packet of Definition 548.373. The read may inspect the supplied zero row, critical-strip row, line-modulus row, finite refusal budget, rational comparison ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it cannot choose a second budget carrier or a hidden analytic route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip budget route determinacy`

Proof. Project the accepted packet through the carrier of Definition 548.1. The displayed rows are exactly $Z, S, M, R, Q, H, C, P, N$, so a fixed-strip budget read has no off-packet coordinate available at the carrier level.

Apply the NameCert obligations, source triad, and zero-strip modulus route, Definition 548.2, Definition 548.76, and Definition 548.106. These bind the supplied zero, strip, modulus, real-part, and comparison rows to one finite route.

Finally use modulus refusal ledger exhaustion and root refusal-boundary totality, Definition 548.318 and Definition 548.330. They force every fixed-strip budget demand onto the same refusal ledger and rule out hidden analytic or host-level budget carriers. □

Theorem 548.375 (CriticalLineWitness fixed-strip zeta consumer boundary)**.** *Every downstream zeta-continuation consumer of the packet in Definition 548.373 may read the refusal budget and rational comparison ledger on $Z, S, M, R, Q, H, C, P, N$. It cannot obtain a zero search,* RH *proof, analytic-continuation certificate, host equality, quotient representative, or ambient completeness row from that fixed-strip budget surface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip zeta consumer boundary`

Proof. Begin with fixed-strip budget route determinacy. It identifies the consumer-facing read with the single finite packet $Z, S, M, R, Q, H, C, P, N$.

Use the downstream readback exhaustion and RH-boundary non-escape rows, Definition 548.297 and Definition 548.253. These theorems place zeta-facing and RH-facing requests on the displayed refusal ledger.

Finally apply zero-route non-escape and RH consumer boundary, Definition 548.315 and Definition 548.316. The consumer receives only finite BHist, hsame, Cont, Pkg, and NameCert data, so zero search, RH proof data, analytic continuation, host equality, quotient representatives, and ambient completeness remain outside the packet. □

Theorem 548.376 (CriticalLineWitness fixed-strip source refusal cover)**.** *Every accepted root-unblock read of a* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet is covered by the displayed fixed-strip source row, the supplied* BHist *ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The cover may read the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison row, and refusal-budget row, but it has no coordinate for analytic continuation, quotient zero sets, host equality, or an* RH *proof row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip source refusal cover`

Proof. Start with the fixed-strip refusal-budget packet of Definition 548.373. Its carrier is the finite row $Z, S, M, R, Q, H, C, P, N$, so every root-unblock read begins with the supplied zero, strip, modulus, real-part, comparison, transport, replay, provenance, and naming rows.

Apply the fixed-strip budget route determinacy theorem, Definition 548.374. It forces the read through the same finite budget surface and prevents a second budget carrier or hidden analytic route.

Finally use the fixed-strip NameCert source lock, Definition 548.379. The local NameCert row is tied to the same BHist, hsame, Cont, and Pkg surface, so analytic continuation, quotient zero sets, host equality, and an RH proof row remain outside the cover. □

Theorem 548.377 (CriticalLineWitness modulus-window transport)**.** *Let two accepted root fixed-strip reads share the displayed source row, fixed modulus window, refusal-budget ledger, componentwise* hsame *row, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. Then the modulus-window transport carries the first read to the second along the same* BHist *packet and preserves the visible source, strip, modulus, real-part, and rational comparison rows. It cannot replace that transport by analytic continuation, quotient zero-set equality, host equality, or any unrecorded target behavior.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus window transport`

Proof. Project both reads through the source refusal cover of Definition 548.376. The only available rows are the fixed-strip source packet and the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Use fixed-strip classifier stability, Definition 548.380. Componentwise hsame transport and displayed Cont replay preserve the accepted classifier on the same $Z, S, M, R, Q, H, C, P, N$ packet.

Now apply fixed-strip ledger exactness, Definition 548.381. Since downstream reads receive exactly the displayed ledger rows, the modulus-window transport is determined by those rows and cannot be distinguished by analytic continuation, quotient zero-set equality, host equality, or unrecorded target behavior. □

Theorem 548.378 (CriticalLineWitness RH-boundary nonclaim)**.** *The root fixed-strip refusal cover and its modulus-window transport expose a boundary for* RH*-facing consumers, not an* RH *proof row. A downstream consumer may read the supplied zero row, critical-strip source row, modulus window, refusal-budget ledger,* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows, but it receives no analytic-continuation certificate, no quotient zero-set equality, no host equality, no ambient completeness row, and no proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh boundary nonclaim`

Proof. Begin with the fixed-strip zeta consumer boundary, Definition 548.375. It places downstream zeta-continuation consumers on the displayed refusal budget and rational comparison ledger.

Apply the source refusal cover and modulus-window transport above. Together they show that root-unblock reads and transported reads remain on the same finite BHist, hsame, Cont, Pkg, and NameCert packet.

Finally use fixed-strip ledger exactness, Definition 548.381. The ledger exhausts the downstream surface by displayed fixed-strip rows, so the RH-facing boundary records a refusal surface and not analytic continuation, quotient zero-set equality, host equality, ambient completeness, or an RH proof row. □

Theorem 548.379 (CriticalLineWitness fixed-strip NameCert source lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted fixed-strip* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every accepted* NameCert *source read of* W *is locked to the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, the fixed-strip source and refusal-budget packet, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows carried by* H, C, P, N*. The read has no coordinate for zero search,* RH *proof data, analytic continuation, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip namecert source lock`

Proof. Start with the carrier of Definition 548.1. It lists the accepted witness as the finite packet $Z, S, M, R, Q, H, C, P, N$, so a NameCert source read begins on the displayed zero row, strip row, support rows, and local naming row.

Apply the NameCert obligations of Definition 548.2. These obligations bind the local naming row to the same BHist, hsame, Cont, and Pkg surface and leave no independent host or quotient naming coordinate.

Now attach the fixed-strip refusal-budget packet Definition 548.373. Its row inventory adds the fixed-strip source, line-modulus, comparison, budget, transport, replay, provenance, and local naming rows without adding an analytic or ambient completion row.

Finally use the fixed-strip budget route determinacy theorem Definition 548.374. It forces every budget-facing source read through that finite packet, so zero search, RH proof data, analytic continuation, host equality, quotient representatives, and ambient completeness are refused rather than read. □

Theorem 548.380 (CriticalLineWitness fixed-strip classifier stability)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted fixed-strip* CriticalLineWitness$^{\uparrow}$ *packet. The fixed-strip classifier determined by W is stable under componentwise* hsame *transport and displayed* Cont *replay on the same rows $Z, S, M, R, Q, H, C, P, N$. Transport or replay may move the accepted read along the displayed packet route, but it cannot replace the classifier by zero search,* RH *proof data, analytic continuation, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip classifier stability`

Proof. Begin with classifier exactness, Definition 548.358. It identifies the root classifier with the supplied zero row Z, critical-strip row S, and displayed BHist, hsame, Cont, Pkg, and local NameCert surface.

Use componentwise StdBridge transport, Definition 548.341. The transport row preserves the displayed packet components, and the replay row keeps the read on the same $Z, S, M, R, Q, H, C, P, N$ carrier.

It remains to exclude alternate classifier sources after transport. Apply Definition 548.379. Since the NameCert source itself is locked to the fixed-strip packet and refuses zero search, RH proof data, analytic continuation, host equality, quotient representatives, and ambient completeness, the transported classifier is stable on exactly the displayed fixed-strip rows. □

Theorem 548.381 (CriticalLineWitness fixed-strip ledger exactness)**.** *Every downstream zeta or* RH*-boundary consumer of an accepted fixed-strip* CriticalLineWitness$^{\uparrow}$ *packet receives exactly the fixed-strip ledger rows $Z, S, M, R, Q, H, C, P, N$ and no stronger analytic coordinate. The consumer may read the supplied zero row, critical-strip row, modulus row, real-part row, rational comparison and refusal-budget ledger, transport row, replay row, provenance row, and local naming row; it does not receive zero search,* RH *proof data, analytic continuation, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip ledger exactness`

Proof. First apply ledger route stability, Definition 548.359. It keeps the modulus and comparison ledger on the displayed M, R, Q, H, C, P, N route and prevents branching to host comparison, quotient representatives, analytic continuation, or ambient completeness.

Next use the fixed-strip zeta consumer boundary, Definition 548.375. It places every downstream zeta-facing read on the refusal budget and rational comparison ledger of the same finite packet.

Finally apply fixed-strip classifier stability, Definition 548.380. Since transport and replay preserve the classifier only on $Z, S, M, R, Q, H, C, P, N$, a downstream zeta or RH-boundary consumer receives exactly those ledger rows and no stronger analytic coordinate. □

Theorem 548.382 (CriticalLineWitness root handoff exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet whose root route is read through the fixed strip, modulus, rational comparison, refusal budget, transport, continuation, package, and local* NameCert *rows. Then every accepted root handoff of W is exhausted by those displayed rows. It has no coordinate for analytic continuation, an* RH *proof, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root handoff exhaustion`

Proof. Start with the finite-budget source route. The source-totality theorem Definition 548.307 places the accepted root read on the fixed zero row, critical-strip row, modulus row, rational comparison row, refusal-budget ledger, transport, replay, provenance, and local naming rows.

Next apply finite-budget ledger stability. The ledger-stability theorem Definition 548.308 keeps comparison and refusal data on the same displayed source route under hsame transport and Cont replay.

Finally use fixed-strip budget route determinacy. The theorem Definition 548.374 says that the fixed-strip route has no alternate analytic, host, quotient, or ambient-completeness branch. Therefore the handoff is exhausted by the finite packet rows and cannot export an RH proof or analytic-continuation coordinate. □

Theorem 548.383 (CriticalLineWitness root classifier budget lock)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* CriticalLineWitness$^{\uparrow}$ *packet. If a classifier-facing read and a finite-budget read preserve the same displayed fixed-strip source route under* hsame *transport and* Cont *replay, then the classifier row and the refusal-budget rows remain locked to that route. Transport and replay cannot replace the displayed classifier by analytic continuation, zero search, host equality, quotient representatives, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootClassifierBudgetLock`

Proof. First read the classifier from the root packet. Classifier exactness identifies the root classifier with the supplied zero row, critical-strip row, modulus and comparison rows, and the displayed BHist, hsame, Cont, Pkg, and local NameCert surface.

Now read the budget side. The finite-budget source-totality and ledger-stability theorems keep the refusal budget on the same $Z, S, M, R, Q, H, C, P, N$ route and preserve it under the same hsame and Cont rows.

It remains to compare the two reads. Fixed-strip budget route determinacy prevents the budget read from branching to analytic continuation, zero search, host equality, quotient representatives, or ambient completeness. Since both reads preserve the same displayed route, the classifier and budget rows stay locked to the same finite packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root classifier budget lock`

Theorem 548.384 (CriticalLineWitness root downstream consumer totality)**.** *Every downstream zeta-continuation or critical-line consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *root packet factors through the finite packet consisting of the fixed strip, modulus, rational comparison, refusal budget, transport, continuation, package, and local* NameCert *rows before any public read. No downstream consumer receives an analytic-continuation certificate, an* RH *proof, host equality, quotient representatives, ambient completeness, or a direct critical-line bypass.*

Proof. Begin with root handoff exhaustion. The preceding theorem confines the root handoff to the fixed-strip, modulus, rational-comparison, refusal-budget, transport, replay, provenance, and naming rows.

Next lock the classifier to the same budget surface. The classifier-budget theorem above ensures that hsame transport and Cont replay preserve the classifier and refusal ledger on exactly the same displayed route.

Finally apply the downstream fixed-strip boundary supplied by fixed-strip budget route determinacy. A zeta-continuation or critical-line consumer can only read the finite packet already exposed by the root route. Thus every public downstream read factors through that packet before it is read, and the excluded analytic, RH, host, quotient, completeness, and bypass coordinates remain absent. □

Theorem 548.385 (CriticalLineWitness fixed-strip carrier source totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet whose fixed-strip route is read through the displayed* BHist *carrier, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. Every accepted carrier-source read of* W *is exhausted by the visible finite source rows* $Z, S, M, R, Q, H, C, P, N$ *over the* $\mathsf{CritStrip}^{\uparrow}$*,* $\mathsf{Gamma}^{\uparrow}$*,* $\mathsf{ZetaZeros}^{\uparrow}$*, and zeta-dependency surface. It has no additional coordinate for zero search, analytic continuation, host equality, quotient representatives, ambient completeness, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip carrier source totality`

Proof. Project the accepted packet through the carrier of Definition 548.1. The only source-side rows exposed by the carrier are the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational-comparison row Q, and the structural rows H, C, P, N.

Use fixed-strip source totality and source-signature readback, Definition 548.317 and Definition 548.361. These identify the carrier source with the visible finite rows over BHist, hsame, Cont, Pkg, and NameCert, and they keep the signature-gap readback on the same displayed route.

Finally attach the zeta dependency surface through Definition 548.225. The $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, and zeta-continuation dependencies are already ledger rows of W, so the carrier-source read is exhausted by $Z, S, M, R, Q, H, C, P, N$ and cannot open any analytic, host, quotient, completeness, zero-search, or RH coordinate. □

Theorem 548.386 (CriticalLineWitness fixed-strip classifier separation)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. A classifier read that is anchored in the fixed-strip source rows* $Z, S, M, R, Q, H, C, P, N$ *is separated from a zero-strip witness read unless the displayed finite route supplies the same signature-gap ledger through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$*. In the absence of that matching ledger, transport and replay keep the*

two reads on their own refusal rows and do not identify them by zero search, analytic continuation, host equality, quotient representatives, ambient completeness, or an RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip classifier separation`

Proof. First read the fixed-strip side. By Definition 548.385, a classifier anchored in the fixed strip has only the visible packet rows $Z, S, M, R, Q, H, C, P, N$ available.

Now read the zero-strip side through the signature-gap carrier. The source-signature readback and GAP-policy stability theorems, Definition 548.361 and Definition 548.362, place any zero-strip witness read behind the displayed signature and gap rows and preserve those rows under H and C.

It remains to compare the two reads. Apply fixed-strip classifier stability, Definition 548.380. If the displayed finite route carries the same signature-gap ledger, the classifier and witness reads meet on that ledger; if it does not, the refusal rows remain distinct and no analytic, host, quotient, completeness, zero-search, or RH route can identify them. □

Theorem 548.387 (CriticalLineWitness zero-source ledger exhaustion)**.** *Every downstream* RH*-facing or zeta-continuation consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet factors through the visible zero-source ledger* $Z, S, M, R, Q, H, C, P, N$ *before it can read a public conclusion. The consumer reads the zero row, fixed strip, source signature, gap ledger, modulus, rational comparison, transport, replay, provenance, and local* NameCert *rows already displayed by* W*; it receives no analytic oracle, host zero classifier, quotient representative, ambient completeness datum, or direct* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZeroSourceLedgerExhaustion`

Proof. Begin with carrier-source totality. The first theorem in this packet confines the zero-source carrier to the finite rows $Z, S, M, R, Q, H, C, P, N$ over the local BHist, hsame, Cont, Pkg, and NameCert surface.

Next use the downstream consumer rows. Root downstream consumer totality and fixed-strip zeta source factorization, Definition 548.384 and Definition 548.389, force zeta-continuation and critical-line consumers to read the displayed packet before any public RH-facing read.

Finally apply classifier separation together with the NameCert route exhaustion theorem, Definition 548.386 and Definition 548.363. The classifier, signature-gap ledger, and downstream route all factor through the same visible zero-source ledger, so the consumer cannot replace that ledger by an analytic oracle, host zero classifier, quotient representative, ambient completeness datum, or direct RH proof. □

Theorem 548.388 (CriticalLineWitness fixed-strip zero-source refusal)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. If an* RH*-facing consumer reads the fixed-strip classifier without the matching zero-source ledger* $Z, S, M, R, Q, H, C, P, N$ *on the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *route, then that classifier read remains on the refusal route. It cannot be promoted to zero search, analytic continuation, host equality, quotient representatives, ambient completeness, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip zero source refusal`

Proof. First apply fixed-strip carrier source totality, Definition 548.385. It confines the classifier-side source to the finite displayed rows of W.

Next apply classifier separation, Definition 548.386. Without the matching signature-gap and zero-source ledger, the fixed-strip classifier and zero-source read stay on distinct displayed rows.

Finally use zero-source ledger exhaustion, Definition 548.387. Every RH-facing consumer must pass through the same ledger before reading a public conclusion, so the unmatched classifier read cannot become any of the refused analytic, host, quotient, completeness, zero-search, or RH endpoints. □

Theorem 548.389 (CriticalLineWitness fixed-strip zeta source factorization)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every zeta-continuation or critical-line consumer of W factors through the concrete packet $Z, S, M, R, Q, H, C, P, N$ before any* RH*-facing read. The consumer receives only the fixed strip, modulus, rational comparison, refusal, transport, continuation, provenance, and local naming rows already displayed by W.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip zeta source factorization`

Proof. Start with root handoff exhaustion, Definition 548.382. It confines the root handoff to the displayed fixed-strip, modulus, rational-comparison, refusal-budget, transport, continuation, package, and naming rows.

Next apply root downstream consumer totality, Definition 548.384. That result makes every zeta-continuation or critical-line consumer read the same finite packet before any public consumer read is accepted.

Finally use the root classifier budget lock, Definition 548.383. Since classifier and budget reads remain locked to the displayed route under hsame transport and Cont replay, no consumer can insert an RH-facing coordinate before the packet factorization. □

Theorem 548.390 (CriticalLineWitness modulus budget RH refusal exhaustion)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the modulus row, rational comparison ledger, and refusal budget exhaust every request for an* RH *proof, zero search, analytic continuation, quotient representative, host equality, or ambient completeness. Such a request has no accepted route outside the displayed M, R, Q, H, C, P, N budget surface.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier modulus budget rh refusal exhaustion`

Proof. Read the refusal side through Definition 548.72. It records the RH-facing refusals on the finite ledger rather than on an analytic, host, quotient, or ambient-completeness coordinate.

Use root handoff exhaustion, Definition 548.382, to place every handoff on the same fixed-strip, modulus, rational-comparison, refusal, transport, continuation, package, and naming rows. Thus a request that reaches the root must be read through the displayed packet.

Now apply the root classifier budget lock, Definition 548.383. The modulus, comparison, and refusal reads stay locked to M, R, Q, H, C, P, N under the displayed transport and replay rows, so each listed demand is exhausted by the finite refusal surface and receives no off-packet witness. □

Theorem 548.391 (CriticalLineWitness NameCert root-unblock consumer scope)**.** *The local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *root-unblock consumer scope of an accepted packet $W = (Z, S, M, R, Q, H, C, P, N)$ is exactly the fixed strip, modulus, rational comparison, refusal, transport, continuation, provenance, and naming rows displayed by W. It includes no analytic-continuation certificate,* RH

proof, zero-search oracle, quotient representative, host equality, ambient-completeness datum, or direct critical-line bypass.

Proof. First apply root downstream consumer totality, Definition 548.384. It gives the consumer-facing inclusion into the fixed-strip, modulus, rational-comparison, refusal, transport, continuation, package, and local NameCert rows.

For the reverse containment of the public scope, use root handoff exhaustion, Definition 548.382. Each displayed row is part of the accepted handoff surface of the packet and is therefore available to the local root-unblock consumer.

It remains to rule out extra rows. The root classifier budget lock Definition 548.383 keeps classifier and refusal reads on the same displayed route, while refusal ledger completeness Definition 548.72 accounts for the RH-facing refusals. Hence the exact local scope is the displayed finite packet surface and no excluded coordinate is admitted. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier namecert root unblock consumer scope`

Theorem 548.392 (Critical-line witness finite-strip carrier admission)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. The finite-strip carrier admission for W is the displayed* BHist *strip route through the fixed observation window Z, S, M, R, Q, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The admission reads the source-modulus route already bound to that fixed window and exports no analytic-continuation certificate, host zero classifier, quotient representative, ambient completeness coordinate, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier finite strip carrier admission`

Proof. Begin with the root source modulus classifier completion of Definition 548.208. It places the source row, strip row, modulus row, rational comparison row, transport, replay, provenance, and local name on the same accepted source-modulus surface.

Next use fixed-strip source completion, Definition 548.317. The fixed finite observation windows are the displayed BHist rows, and the source-modulus route is read only through hsame, Cont, Pkg, and NameCert.

Finally attach the zeta dependency lattice of Definition 548.169. The zeta-facing and RH-boundary dependencies are consumed before any modulus witness is public, so admission remains a finite carrier row and not an analytic theorem source. □

Theorem 548.393 (Critical-line witness source-modulus gap separation)**.** *For every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet $W = (Z, S, M, R, Q, H, C, P, N)$, the source-modulus route over the fixed finite observation windows separates the displayed strip rows Z, S from the modulus and comparison rows M, R, Q by the same gap ledger carried by* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. A downstream consumer may read the separation only as a finite BEDC obligation row; it cannot replace the gap by host equality, a global zero classifier, analytic continuation, quotient equality, ambient completeness, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus gap separation`

Proof. First read the finite-strip carrier admission above. It fixes the carrier rows and the source-modulus route before any downstream zeta or critical-line consumer can request a public conclusion.

Next use the source-modulus classifier lock of Definition 548.208. The classifier lock keeps $Z, S, M, R, Q, H, C, P, N$ on one displayed route and refuses global zero classification, host equality, quotient representatives, and ambient completeness.

Now apply fixed-strip source totality, Definition 548.317. The fixed observation windows and source-modulus rows remain finite BHist coordinates under the displayed transport and replay.

Finally use the zeta dependency lattice, Definition 548.169. Since zeta and RH boundary reads are dependency rows rather than replacement coordinates, the gap separation remains attached to the source-modulus route. □

Theorem 548.394 (Critical-line witness zeta-consumer refusal totality)**.** *Every zeta-facing or* RH*-boundary consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *that reads the finite-strip source-modulus route is total on the displayed refusal surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The consumer may inspect the fixed BHist *strip rows, source-modulus rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. It receives no completed zeta object, zero oracle, analytic-continuation certificate, host equality, quotient stream, ambient completeness coordinate, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta consumer refusal totality`

Proof. Begin with zeta dependency lattice, Definition 548.169. It orders zeta-facing and RH-boundary reads before the modulus witness and keeps the witness as the displayed finite packet.

Next apply finite-strip carrier admission and source-modulus gap separation. Together they show that the consumer route is the same fixed-window source-modulus surface over BHist, hsame, Cont, Pkg, and NameCert.

Finally use fixed-strip source completion, Definition 548.317. The completion exposes no zero-producing search, analytic-continuation theorem, host equality, quotient representative, ambient completeness witness, or proof of RH, so the zeta-facing refusal is total on $Z, S, M, R, Q, H, C, P, N$. □

Theorem 548.395 (Critical-line witness fixed-strip gap-consumer nonescape)**.** *Every accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$ *whose source-modulus gap is read through the fixed strip rows* $Z, S, M, R, Q, H, C, P, N$ *keeps that gap visible to any zeta-facing consumer. The consumer may inspect the fixed* BHist *strip rows, source-modulus route, rational comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it cannot replace the gap by analytic continuation, a zero oracle, host equality, quotient streams, ambient completeness, or an* RH *proof.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip gap consumer nonescape`

Proof. Begin with finite-strip carrier admission, Definition 548.392. It fixes $Z, S, M, R, Q, H, C, P, N$ as the complete carrier route before any zeta-facing consumer can read a conclusion.

Apply source-modulus gap separation, Definition 548.393. The gap is attached to the displayed source-modulus route and is not stored as a host equality, quotient stream, zero oracle, or ambient completeness coordinate.

Use fixed-strip source totality and zeta-consumer refusal totality, Definition 548.317 and Definition 548.394. These theorems keep the fixed-window read total on the displayed refusal surface and expose no analytic-continuation certificate or RH proof.

The conclusion follows by carrier-coordinate exhaustion. Every row available to the consumer is one of $Z, S, M, R, Q, H, C, P, N$, and none of those rows supplies a replacement for the source-modulus gap. □

Theorem 548.396 (CriticalLineWitness root zeta source obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing read that cites the zeta-continuation source row of W must pass through the displayed* BHist *source row Z, componentwise* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row N. The read cannot replace Z by host complex analysis, a hidden zero selector, quotient equality, or an analytic-continuation oracle row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root zeta source obligation`

Proof. Start from the carrier of Definition 548.1. It exposes the zeta source as a finite row of the accepted packet, not as an ambient analytic object.

Use the zeta dependency lattice and fixed-strip zeta source factorization surfaces. They route zeta-facing continuation reads through the same BHist, hsame, Cont, Pkg, and NameCert rows already displayed by W.

Finally apply the root refusal and source-exhaustion siblings. Since those rows exclude host complex analysis, hidden zero selection, quotient equality, and analytic-continuation oracle data, the zeta source obligation remains a finite BEDC root obligation. □

Theorem 548.397 (CriticalLineWitness root real-window obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing finite-window read of the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *side must pass through the displayed real-part row R, rational comparison row Q, modulus row M, transport row H, replay row C, provenance row P, and local naming row N. The read exports no ambient completeness witness, host real equality, quotient representative, hidden zero selector, or analytic-continuation oracle row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root real window obligation`

Proof. Read the real-window side through the existing RealUp and RatUp sibling surfaces. The packet exposes M, R, Q as finite rows for modulus, real-part, and rational-comparison data.

Next apply the root modulus ledger and rational comparison completion rows. They force every finite-window consumer to preserve the same H, C, P, N support while reading M, R, Q.

The refusal boundary then removes the forbidden exits. No accepted finite-window read can become host real equality, ambient completeness, a quotient representative, hidden zero selection, or analytic continuation outside the displayed packet. □

Theorem 548.398 (CriticalLineWitness root strip classifier obligation)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every root-facing critical-strip classifier read must be anchored in the displayed strip row S, zeta source row Z, finite-window rows M, R, Q, componentwise* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. It supplies a critical-strip witness boundary only, with no full* RH *proof, host complex-analysis classifier, quotient equality, hidden zero selector, or analytic-continuation oracle row.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier root strip classifier obligation`

Proof. Begin with the critical strip witness and fixed-strip source classifier siblings. They expose the strip classifier as a finite source row coupled to Z, S, M, R, Q.

Transport and replay the classifier through H and C. The provenance and naming rows P, N keep the classifier attached to the accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet rather than to a detached analytic object.

Finally combine the fixed-strip refusal and root RH-boundary rows. These surfaces make the classifier a BEDC witness boundary and exclude a full RH proof, host complex-analysis classifier, quotient equality, hidden zero selection, or analytic-continuation oracle row. □

Theorem 548.399 (CriticalLineWitness zeta real strip root handoff)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. Before any zeta-continuation, critical-line, or RH-facing consumer route is accepted, the packet must expose the zeta source row Z, the fixed critical-strip row S, the modulus row M, the $\mathsf{Real}^{\uparrow}$ finite-window row R, the $\mathsf{RegSeqRat}^{\uparrow}$ rational-comparison row Q, the componentwise BHist and hsame transport row H, the displayed Cont replay row C, the Pkg provenance row P, and the local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ row N. No accepted route may read the handoff as an analytic continuation theorem, a zero-search oracle, a quotient equality, a host real equality, or an ambient completeness datum.*

Proof. Use the zeta source obligation of Definition 548.396. It forces each zeta-continuation source read through Z, H, C, P, N and prevents the source from becoming hidden complex-analysis input.

Next use the real-window obligation of Definition 548.397. It adds the finite M, R, Q rows and keeps the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ side on the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

Finally use the strip classifier obligation of Definition 548.398 together with the downstream source package of Definition 548.179. These results attach the fixed strip classifier to $Z, S, M, R, Q, H, C, P, N$ before a consumer can inspect the route. The refusal ledger rows then remove analytic continuation, zero search, quotient equality, host real equality, and ambient completeness from the handoff. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta real strip root handoff`

Theorem 548.400 (CriticalLineWitness fixed-strip budget source totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. The admitted root source reads of W are exhausted by the fixed-strip source read through Z, S, H, C, P, N, the refusal-budget read through $Z, S, M, R, Q, H, C, P, N$, and the real-window source read through M, R, Q, H, C, P, N. A root consumer has no fourth source channel outside those finite rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip budget source totality`

Proof. The fixed-strip source side is supplied by Definition 548.398 and Definition 548.376. These bind the strip classifier and source refusal cover to the displayed packet rows.

The refusal-budget side is supplied by Definition 548.213. Its budget route is the same finite packet $Z, S, M, R, Q, H, C, P, N$, with H, C, P, N carrying the transport, replay, provenance, and local name.

The real-window side is supplied by Definition 548.397. Since the root source-window route lock and refusal ledger already reject detached analytic and host-level channels, these three reads exhaust the admitted root source surface. □

Theorem 548.401 (CriticalLineWitness RH refusal source exactness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ packet. Every RH-facing demand on W is classified exactly as*

a finite refusal/source row over $Z, S, M, R, Q, H, C, P, N$. Such a demand cannot be converted into analytic continuation, zero search, quotient equality, host real equality, or ambient completeness, and it cannot strengthen the packet beyond its zeta-continuation source rows and fixed-strip classifier rows.

Proof. Start with the combined handoff of Definition 548.399. It exposes the zeta source, real-window rows, rational comparison rows, strip classifier, transport, replay, provenance, and local naming row before any RH-facing consumer route is accepted.

Apply the fixed-strip budget source totality of Definition 548.400. It leaves only the fixed-strip source read, refusal-budget read, and real-window source read as admitted root source channels.

Finally use the RH consumer boundary of Definition 548.279 and the RH refusal readback totality of Definition 548.189. These sibling rows classify the remaining demand as finite BHist, hsame, Cont, Pkg, and NameCert refusal/source data and exclude analytic continuation, zero search, quotient equality, host real equality, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal source exactness`

Theorem 548.402 (CriticalLineWitness root zeta, Real, and Rat source readiness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Before any downstream zeta,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Rat}^{\uparrow}$ *consumer reads W, the packet exposes the zeta-facing source row Z, the modulus row M, the real-part row R, the rational-comparison row Q, the* BHist *carrier surface, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *route N. Thus every such consumer receives a finite BEDC route through the displayed packet rather than an analytic, host-level, quotient, or ambient-completeness substitute.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootZetaRealRatSourceReadiness`

Proof. Begin with the accepted carrier tuple. Its visible rows are $Z, S, M, R, Q, H, C, P, N$, so the zeta-facing source, modulus, real-part, and rational-comparison reads are already named before any downstream consumer asks for them.

Use the zeta source obligation, real-window obligation, and strip classifier obligation from the preceding root rows. They place zeta continuation, modulus, $\mathsf{Real}^{\uparrow}$ real-part, and $\mathsf{Rat}^{\uparrow}$ comparison reads on the same BHist, hsame, Cont, Pkg, and local NameCert route.

The displayed route is finite because every component is a row of W. Therefore a downstream consumer can read only through those packet rows, and no analytic continuation, host equality, quotient representative, or ambient completeness object is available as a replacement. □

Theorem 548.403 (CriticalLineWitness root gap refusal completeness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Every demand for an* RH *proof, analytic continuation, host zero search, quotient representative, host equality, or ambient-completeness witness is a refusal-side demand unless it is routed through the displayed $Z, M, R, Q,$* BHist*$, H, C, P, N$ rows of W. In particular, no such demand can become an accepted downstream read by bypassing the finite source, modulus, real-part, rational-comparison, transport, replay, provenance, and naming route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootGapRefusalCompletenessPackage`

Proof. First isolate the accepted route. The source-readiness theorem above makes the zeta-facing, modulus, real-part, and rational-comparison data available only as displayed rows of W, with their BHist, hsame, Cont, Pkg, and NameCert support.

Next classify the listed demands. Requests for RH truth, analytic continuation, host zero search, quotient representatives, host equality, or ambient completeness ask for data not displayed by the packet unless they are rewritten as reads of Z, M, R, Q, H, C, P, N.

Finally apply the root refusal boundary. It sends every off-packet request to the refusal side, so the only accepted form of such a request is the finite BEDC route already exposed by W. □

Theorem 548.404 (CriticalLineWitness root downstream unblock package)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the downstream root-unblock package is the finite conjunction of the source-readiness row of Definition 548.402, the refusal-completeness row of Definition 548.403, the zeta dependency lattice of Definition 548.169, the root refusal boundary totality of Definition 548.330, and the root modulus ledger of Definition 548.68. This package is sufficient for downstream critical-line, zeta,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *consumers that require a finite BEDC route, and it claims no* RH *proof, analytic completeness theorem, host zero selector, quotient equality, or ambient-completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRootDownstreamUnblockPackage`

Proof. Assemble the positive route first. The source-readiness row supplies $Z, M, R, Q, \mathsf{BHist}, H, C, P, N$, while the zeta dependency lattice and root modulus ledger attach the zeta-facing and real-rational reads to the same displayed packet.

Now assemble the negative boundary. The refusal-completeness row and root refusal boundary totality classify every RH, analytic-continuation, host-zero, quotient, host-equality, or ambient-completeness demand as off-packet unless it travels through the displayed finite rows.

The downstream consumer therefore receives exactly the finite conjunction stated in the theorem. Since each conjunct is a named packet row or an existing root boundary theorem, the package unblocks the finite BEDC route without adding any analytic or host-level substitute. □

Theorem 548.405 (CriticalLineWitness Phase Real root readback package)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the Phase Real root readback package is the finite conjunction of the root downstream unblock package of Definition 548.404, the Phase Real source triad of Definition 548.413, the source budget readiness row of Definition 548.420, and the source window exhaustion row of Definition 548.416. Consequently downstream* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers receive only displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows of* W*, and receive no analytic completeness theorem, host zero selector, quotient equality, or ambient real-completion witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealRootReadbackCertificate`

Proof. Project the root downstream unblock package first. It already exposes the source-readiness row, refusal-completeness row, zeta dependency lattice, root refusal boundary totality, and root modulus ledger as a finite BEDC route through the displayed packet.

Now import the three Phase Real source surfaces. The source triad binds $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads to the same root source surface; source budget readiness preserves the shared BHist, hsame, Cont, Pkg, and local NameCert support; source window exhaustion states that the visible Phase Real root read is exhausted by the displayed source-window packet.

Finally apply the refusal-completeness row contained in the root package. Analytic completeness, host zero selection, quotient equality, and ambient real-completion demands have no displayed coordinate in W, so the only downstream readback package available to the four consumers is the finite conjunction stated above. □

Theorem 548.406 (CriticalLineWitness Phase Real L10 face status pullback)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Any downstream Phase Real and Completion consumer of the readback package of Definition 548.405 must pull the four local status records from* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *objectwise. The rows of W supply only the finite source window and support route, not an aggregate verification-status row, bridge row, selected limit, quotient equality, host real equality, analytic completeness theorem, or ambient real-completion witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealL10FaceStatusPullback`

Proof. Start with the finite source route. By Definition 548.405, the Phase Real root readback package exposes the root downstream unblock package together with the Phase Real source triad, source budget readiness row, and source window exhaustion row through the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ support of W.

Now read the status side from the four local faces. Apply Definition 291.112, Definition 126.509, Definition 301.219, and Definition 38.494. These results force the Phase Real and Completion L10 status lookup to remain objectwise on $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Thus the accepted rows of W can name the finite source window and carry the shared support route into those consumers, but they cannot synthesize a route-level status coordinate or any of the listed analytic, quotient, host, bridge, selected-limit, or ambient-completion witnesses. □

Theorem 548.407 (CriticalLineWitness Phase Real consumer determinacy)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Any downstream Phase Real consumer using the readback package of Definition 548.405 must choose one and only one objectwise face among* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and must consume only that face's local status and readback row. The packet cannot merge the four faces into a route-level completion witness, bridge row, selected limit, quotient equality, or host real equality.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real consumer determinacy`

Proof. First restrict the available route. By Definition 548.405, the readback package is a finite conjunction of displayed root and Phase Real source rows of W, carried by the visible BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ support.

Next separate the status reads. By Definition 548.406, downstream Phase Real and Completion consumers can pull the L10 status records only objectwise from $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; no aggregate route-level status row is supplied by W.

It remains to check the four permitted faces. Apply Definition 291.112, Definition 126.509, Definition 301.219, and Definition 38.494. These four face results provide the local rows that the consumer may select, and they do so without identifying the rows with each other.

Therefore the consumer is deterministic at the face boundary: after selecting one of the four objectwise faces, it reads that face's local status and readback row and has no constructor or cited theorem that combines the four rows into a route-level completion witness, bridge row, selected limit, quotient equality, or host real equality. □

Theorem 548.408 (CriticalLineWitness Phase Real dependency totality)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier. Every Phase Real or Completion read that consumes W first factors through the displayed supplied zero-strip route, modulus ledger entry, zeta dependency lattice, and RH refusal boundary:*

$$Z, S, M, R, Q, H, C, P, N.$$

The read may use this packet as a finite source-modulus dependency surface, but it has no coordinate for analytic continuation, host zero search, quotient zero sets, selected host limits, ambient completeness, or an RH *proof row.*

Proof. Start with the carrier of Definition 548.1. Its only public coordinates are the supplied zero row Z, critical-strip row S, modulus row M, real-part row R, rational-comparison row Q, componentwise transport H, displayed continuation C, package provenance P, and local naming row N.

Apply the supplied zero-strip route of Definition 548.82 and the modulus ledger entry of Definition 548.83. Together they force the source side and the Real-rational comparison side of every consumer read through the same finite packet.

Next use the zeta dependency lattice of Definition 548.169. It places any zero-strip or critical-line read behind the existing $\mathsf{ZetaZeros}^{\uparrow}$ and RH-boundary surfaces before a modulus witness is public.

Finally apply the RH non-escape package of Definition 548.84. It removes analytic continuation, host zero search, quotient representatives, ambient completeness, and RH proof content from the accepted route. Hence the Phase Real and Completion dependency route is exhausted by the displayed rows of W. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real dependency totality`

Theorem 548.409 (CriticalLineWitness zero-source Real readback factorization)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier* $W = (Z, S, M, R, Q, H, C, P, N)$*, every Real-facing readback from the supplied zero source factors through the route*

$$Z, S, H, C, P, N \quad \textit{then} \quad M, R, Q, H, C, P, N.$$

Thus the zero source and the Real-rational modulus ledger are read as two faces of the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *packet; no consumer can identify the zero source with a Real endpoint or bypass the modulus ledger.*

Proof. Read the supplied source side through Definition 548.82. The rows Z, S, H, C, P, N are the accepted zero-strip route and are already tied to the local $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ row.

Read the Real-facing side through Definition 548.83. The rows M, R, Q, H, C, P, N form the finite modulus, real-part, and rational-comparison ledger available to a downstream consumer.

The carrier of Definition 548.1 stores both routes in the same packet, with shared transport, replay, provenance, and local naming rows. A readback that keeps the zero source but drops the modulus ledger therefore leaves the accepted carrier surface.

The non-escape clause follows from Definition 548.84 and Definition 548.169. They prevent the zero source from becoming a host Real endpoint, quotient representative, selected limit, or analytic certificate outside the displayed route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zero source real readback factorization`

Theorem 548.410 (CriticalLineWitness Real-window nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet must expose the supplied zero row* Z*, critical-strip row* S*, modulus row* M*, real-part row* R*, rational-comparison row* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N *before any L10 readback. No analytic-continuation row, selected zero, quotient zero set, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or* RH *proof row can be read from that window.*

Proof. Start with Definition 548.408. It says that every Phase Real or Completion read factors through the displayed rows $Z, S, M, R, Q, H, C, P, N$ before any consumer-facing use of the packet.

Apply Definition 548.409. The zero-source side Z, S, H, C, P, N and the Real-rational side M, R, Q, H, C, P, N are two faces of the same finite BHist route, so an L10 readback cannot keep one side while dropping the other.

Use Definition 548.169 to place zero-strip and critical-line reads behind the existing zeta dependency lattice. This keeps the visible Real window tied to the supplied source and modulus rows rather than to a host analytic object.

Finally apply Definition 548.84. Analytic continuation, selected zeros, quotient zero sets, ambient Real completeness, and RH proof data are classified outside the accepted carrier, so none of them can be read from the Real-facing window. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier real window nonescape`

Theorem 548.411 (CriticalLineWitness RH refusal phase boundary)**.** *Every* RH*-facing Phase Real or Completion request to an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *carrier is bounded by the same finite refusal surface*

$$Z, S, M, R, Q, H, C, P, N.$$

The boundary may expose the supplied zero-strip route, the modulus ledger entry, the zeta dependency lattice, and the local refusal package. It does not export an RH *truth value, zero-location theorem, analytic-continuation certificate, quotient zero-set equality, selected limit, or ambient-completeness row.*

Proof. Begin with the carrier row of Definition 548.1. A request to the packet can inspect only the rows $Z, S, M, R, Q, H, C, P, N$.

Use Definition 548.82 for the source side and Definition 548.83 for the modulus side. These two rows give the full positive data visible to a Phase Real or Completion consumer.

Now apply the zeta dependency lattice of Definition 548.169. It records that zero-strip and critical-line reads depend on existing zeta and RH boundary surfaces rather than on a standalone zeta-function object.

Finally use Definition 548.84. The package classifies RH proofs, zero searches, quotient representatives, analytic continuation, and analytic completeness as absent from the accepted carrier. The stated refusal boundary follows. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier rh refusal phase boundary`

548.0.1 Fixed-strip source lock

Theorem 548.412 (CriticalLineWitness fixed-strip source lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet whose fixed-strip read is consumed by an* RH*-facing route. The read is locked to the finite source tuple consisting of the* $\mathsf{CritStrip}^{\uparrow}$ *strip row* S*, the* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *continuation row read through* Z*, the* $\mathsf{ZetaZeros}^{\uparrow}$ *zero ledger* Z*, the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window rows* R, Q*, the modulus row* M*, the displayed* BHist *carrier data, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *support* N*. No accepted consumer may replace this tuple by an* RH *proof, a nontrivial-zero classification, host complex equality, ambient analytic continuation, quotient equality, or ambient completeness.*

Proof. Start from the root zeta, real, and rational source-readiness package of Definition 548.402. It exposes Z, M, R, Q, H, C, P, N before any downstream critical-line, zeta, $\mathsf{Real}^{\uparrow}$, or $\mathsf{RegSeqRat}^{\uparrow}$ consumer can read W.

Attach the fixed-strip side through the fixed-strip source totality and source classifier rows, Definition 548.317 and Definition 548.385. These place the $\mathsf{CritStrip}^{\uparrow}$ strip row, the $\mathsf{ZetaZeros}^{\uparrow}$ ledger, and the $\mathsf{ZetaContinuationWitness}^{\uparrow}$ source read on the same displayed BHist, hsame, Cont, Pkg, and local NameCert route. The real and rational finite-window side is the finite-window surface supplied by chapter 38 and chapter 301.

Finally apply the root gap refusal completeness and downstream unblock package, Definition 548.403 and Definition 548.404. They leave the consumer with exactly the finite source tuple stated above and refuse every request for RH truth, nontrivial-zero classification, host complex equality, ambient analytic continuation, quotient equality, or ambient completeness outside that tuple. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier fixed strip source lock`

Theorem 548.413 (CriticalLineWitness Phase Real source triad)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every fixed-strip source read used by this chapter binds its source rows to the current Phase Real and Completion triad*

$$\mathsf{Real}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}.$$

The visible route is the fixed-strip source row, the $\mathsf{Real}^{\uparrow}$ *endpoint row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support of* W*.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real source triad`

Proof. Start with the fixed-strip source lock of Definition 548.412. It already places the strip source, zeta ledger, continuation read, $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, modulus row, and local support rows on one finite tuple.

Attach the Phase Real dependency totality of Definition 548.408. It forces every Phase Real or Completion read through the displayed source-modulus dependency surface.

Finally use the Real L10 dependency surface of chapter 38. The fixed-strip source can therefore name the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows without importing any off-packet analytic or host coordinate. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealSourceTriad`

Theorem 548.414 (CriticalLineWitness Phase Real source totality)**.** *Every fixed-strip source read used by this chapter factors through the three displayed L10 rows* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$*, together with the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows exposed by the accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. No additional source row is available between the fixed-strip source lock and the Phase Real terminal read.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealSourceTotality`

Proof. First expose the source tuple by Definition 548.412. The accepted read is locked to the strip, zeta, continuation, Real, regular-rational, modulus, transport, replay, provenance, and local naming rows.

Next apply Definition 548.404. Its downstream package makes the finite source route sufficient for critical-line, zeta, $\mathsf{Real}^{\uparrow}$, and rational consumers.

Use Definition 548.413 to isolate the Phase Real face of that package. The only L10 rows visible to the source read are $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, supported by the displayed BEDC rows of W. □

Theorem 548.415 (CriticalLineWitness Phase Real source nonescape)**.** *The root source surface for a fixed-strip* $\mathsf{CriticalLineWitness}^{\uparrow}$ *read exports no* RH *truth value, analytic-continuation certificate, quotient stream, selected limit, or host complex equality. Such a read can only inspect the finite Phase Real source triad and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real source nonescape`

Proof. Begin with the refusal side. Definition 548.403 and Definition 548.411 classify RH truth, analytic continuation, host zero search, quotient representatives, host equality, and ambient completeness as off-packet demands.

Now apply Definition 548.414. Since every accepted fixed-strip source read has already been exhausted by the Phase Real source triad and its displayed support rows, the excluded demands have no carrier coordinate on the root source surface. □

Theorem 548.416 (CriticalLineWitness Phase Real source-window exhaustion)**.** *Let* $W = (Z, S, M, R, Q, H, C, P,$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet. Any root read of* W *using the Phase Real source window factors through the displayed* $\mathsf{Rat}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *row, zero-strip row, finite modulus row,* BHist *transport,* Cont *replay,* hsame *comparison, package provenance, and local* NameCert *row. No root read using that window can add an off-packet analytic source, host zero selector, or detached real-line certificate.*

Proof. First expose the source side. Apply Definition 548.402 and Definition 548.76. These rows place the zeta, real, rational, supplied-zero, critical-strip, and source-facing comparison data on the displayed finite packet before a root consumer reads W.

Next bind the modulus and zero-strip faces. Use Definition 548.347 for the finite modulus classifier and Definition 548.209 for the supplied zero-strip readback route. The two rows share the same H, C, P, N support surface, so the Phase Real window has no second route around the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally attach the Phase Real source totality of Definition 548.414. It identifies $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ as the only visible Phase Real endpoints for this source window, while the rational comparison row supplies the $\mathsf{Rat}^{\uparrow}$ face. Thus the root read is exhausted by the stated finite source-window packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealSourceWindowExhaustion`

Theorem 548.417 (CriticalLineWitness RegSeqRat-to-Real modulus handoff)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, every handoff from a* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-request row to a* $\mathsf{Real}^{\uparrow}$ *endpoint row inside the Phase Real source window must pass through the supplied* BHist *carrier rows, the displayed* Cont *route, and the componentwise* hsame *comparison carried by* H, C*. The handoff cannot skip directly from a regular rational sequence to a host real value, a quotient representative, or an unrecorded modulus witness.*

Proof. Begin with Definition 548.413. The Phase Real source face already names $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ through the same fixed-strip source row and the displayed support rows of W.

Now apply the modulus classifier. By Definition 548.347, any modulus read used by the root route is read through M, R, Q together with H, C, P, N. The zero-strip readback totality of Definition 548.209 fixes the source side of that route before the regular-rational-to-real handoff is consumed.

It remains to exclude detached real semantics. The source-window exhaustion row above leaves only the BHist, Cont, and hsame route between the finite request and the endpoint row. A host real value, quotient representative, or unrecorded modulus witness is not one of those coordinates, so it cannot serve as the handoff. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessRegSeqRatRealModulusHandoff`

Theorem 548.418 (CriticalLineWitness Phase Real classifier determinacy)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N)$ and $W' = (Z', S', M', R', Q', H', C', P', N')$ be accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packets. Suppose their Phase Real source windows expose the same displayed* $\mathsf{Rat}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *row, zero-strip row, finite modulus row,* BHist *transport,* Cont *replay,* hsame *comparison,* Pkg *provenance, and local* NameCert *row. Then every root or zeta-facing consumer reads the same classifier-facing source window from W and W'.*

Proof. Use Definition 548.416 to reduce both consumer reads to the displayed Phase Real source rows. After this reduction, the two reads have no visible coordinates except the shared rational, real, regular-rational, zero-strip, modulus, transport, replay, comparison, provenance, and local naming rows listed in the hypothesis.

Apply Definition 548.417 to the common $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ faces. The handoff prevents either packet from replacing the displayed finite modulus route by a detached host real, quotient representative, or unrecorded modulus witness.

Finally apply Definition 548.419. A root or zeta-facing consumer cannot import an external analytic truth primitive, analytic-continuation certificate, global zero-set classifier, quotient equality, or ambient-completeness witness. Therefore the classifier-facing source window is determined by the shared displayed rows, and both packets present the same window to every such consumer. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real classifier determinacy`

Theorem 548.419 (CriticalLineWitness zeta refusal consumer scope)**.** *Every* RH *or zeta-facing consumer of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet that reads the Phase Real source window sees only the displayed refusal ledger, source triad, rational comparison row, finite modulus row, zero-strip readback row,* BHist *route,* Cont *route,* hsame *transport,* Pkg *provenance, and local* NameCert *row. Such a consumer cannot import an external analytic truth primitive, analytic-continuation certificate, global zero-set classifier, quotient equality, or ambient-completeness witness.*

Proof. First collect the positive surface. Apply Definition 548.402, Definition 548.76, and the source-window exhaustion theorem above. These rows identify exactly the source, zeta, real, rational, modulus, zero-strip, transport, replay, provenance, and naming coordinates visible to the consumer.

Next read the refusal side through the existing root package. The Phase Real source non-escape row Definition 548.415 and the root downstream unblock package Definition 548.404 refuse RH proof data, analytic continuation, host zero search, quotient representatives, host equality, and ambient completeness outside the displayed packet.

Therefore the zeta-facing consumer has no carrier slot in which to place an external analytic truth primitive or detached analytic certificate. Its only admissible imports are the refusal ledger and the finite BEDC rows named in the statement. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZetaRefusalConsumerScope`

Theorem 548.420 (CriticalLineWitness Phase Real source-budget readiness)**.** *Let $W = (Z, S, M, R, Q, H, C, P, N$ be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet. Every Phase Real source-budget read of W is a read of the same finite* BHist *packet, the same* Cont *replay row, the same componentwise* hsame *transport, the same package provenance, and the same local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row. The read may use the displayed* $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *zero-strip, finite modulus, and refusal-ledger coordinates, but it cannot replace them by host reals, analytic continuation, selected zero data, or an ambient-completeness witness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPhaseRealSourceBudgetReadiness`

Proof. First expose the fixed source tuple. Apply Definition 548.416 and Definition 548.414. These rows keep the rational, regular-rational, real, zero-strip, and finite-modulus data inside the displayed packet $Z, S, M, R, Q, H, C, P, N$.

Next retain the source-budget face. The source-window exhaustion theorem binds any Phase Real read to the displayed BHist, Cont, hsame, Pkg, and local $\mathsf{NameCert}$ rows before the read reaches a consumer.

Finally read the refusal side through Definition 548.419. Since the refusal ledger has no coordinate for host reals, analytic continuation, selected zero data, or ambient completeness, the source-budget read is exactly the finite BEDC route stated above. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real source budget readiness`

Theorem 548.421 (CriticalLineWitness Phase Real RegSeqRat route readiness)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet $W = (Z, S, M, R, Q, H, C, P, N)$, every* $\mathsf{RegSeqRat}^{\uparrow}$ *route used by a Phase Real consumer must pass through the same finite BEDC route as the packet itself:* BHist *carrier admission,* Cont *replay, componentwise* hsame *comparison, package provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$. *The route is not a quotient stream, a selected limit, a host real equality, or an analytic continuation certificate.*

Proof. Start from the RegSeqRat-to-Real handoff of Definition 548.417. It places every regular-rational request behind the supplied carrier rows and the displayed continuation route before the Phase Real endpoint is read.

Use Definition 548.420. The budget row shares the same H, C, P, N support surface as the regular-rational route, so no second route can be inserted between the sequence data and the real-facing endpoint.

The excluded objects would require a quotient row, selected-limit row, host equality row, or analytic-continuation row. None is present in W, and the local NameCert row accepts only the finite route over $Z, S, M, R, Q, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real regseqrat route readiness`

Theorem 548.422 (CriticalLineWitness Phase Real zeta-refusal ledger)**.** *Every zeta-facing or* RH*-facing read of an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet reads the Phase Real refusal ledger through the displayed finite BEDC route* $Z, S, M, R, Q, H, C, P, N$*. In particular, the ledger combines source budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *route,* $\mathsf{Real}^{\uparrow}$ *endpoint, zero-strip readback,* BHist *transport,* Cont *replay, componentwise* hsame*, package provenance, and local NameCert data, and it exports no host analytic truth primitive.*

Proof. First collect the source and budget rows from Definition 548.420. The Phase Real budget is already a read of the fixed carrier, continuation, transport, provenance, and local naming rows.

Next attach the regular-rational handoff by Definition 548.421. This forces the $\mathsf{RegSeqRat}^{\uparrow}$ route and $\mathsf{Real}^{\uparrow}$ endpoint to remain part of the same finite BEDC packet.

Finally apply Definition 548.419. The zeta-facing consumer sees only the refusal ledger and the finite source rows, so a host analytic truth primitive has no admission point in the packet. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real zeta refusal ledger`

Theorem 548.423 (CriticalLineWitness Phase Real terminal no-completeness boundary)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet. Any terminal Phase Real or zeta-facing consumer that uses the source-budget row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *route-readiness row, and the zeta-refusal ledger factors through the same displayed finite BEDC packet: the* BHist *carrier,* Cont *replay, componentwise* hsame *comparison,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *row,* $\mathsf{Rat}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *endpoint rows, zero-strip readback, finite modulus, and refusal ledger. Consequently the route exports no quotient stream equality, selected limit, host real equality, analytic-continuation truth primitive, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or countable-choice selector.*

Proof. First apply Definition 548.420. It fixes the terminal read over the displayed BHist, Cont, hsame, Pkg, local NameCert, rational, regular-rational, real, zero-strip, finite-modulus, and refusal-ledger rows.

Next apply Definition 548.421. This keeps the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ endpoint inside the same finite source packet, rather than turning them into a quotient stream or selected-limit construction.

Finally apply Definition 548.422. The terminal zeta-facing read is only a read of the refusal ledger through those finite rows. Since no excluded row occurs in that displayed carrier or in the ledger, there is no export point for host real equality, analytic-continuation truth, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric completeness, or countable-choice selection. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real terminal nocompleteness boundary`

Theorem 548.424 (CriticalLineWitness Phase Real selector-route totality)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet. Every Phase Real selector read of* W *factors through the displayed finite route: the* BHist *source packet, the StreamName-facing source window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *route, the* $\mathsf{Real}^{\uparrow}$ *endpoint, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows. The selector route may inspect the source-budget and zeta-refusal ledgers already present in* W*, but it has no coordinate for host real equality, analytic truth, selected Cauchy representatives, or ambient completeness.*

Proof. First expose the fixed source side. Apply Definition 548.416 and Definition 548.420. They bind the Phase Real read to the same finite source packet $Z, S, M, R, Q, H, C, P, N$ and to the same BHist, hsame, Cont, Pkg, and local NameCert rows before any selector reaches a real-facing consumer.

Next route the selector through the regular-rational face. Apply Definition 548.421. The $\mathsf{RegSeqRat}^{\uparrow}$ route and $\mathsf{Real}^{\uparrow}$ endpoint are therefore read as displayed rows of the packet, not as a quotient stream, selected limit, host real equality, or analytic continuation certificate.

Finally attach the refusal ledger by Definition 548.422. The ledger combines the source budget, regular-rational route, real endpoint, zero-strip readback, carrier transport, continuation replay, provenance, and local naming rows. Since the carrier and NameCert obligation blocks Definition 548.1 and Definition 548.2 expose no additional selector coordinate, every Phase Real selector read factors through the displayed finite route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real selector route totality`

Theorem 548.425 (CriticalLineWitness RegSeqRat Real budget separation)**.** *For an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet* $W = (Z, S, M, R, Q, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *budget data and the* $\mathsf{Real}^{\uparrow}$ *endpoint remain separate visible rows of the Phase Real route. A consumer may read their shared* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *support, but it cannot collapse the budget row into host real equality, analytic continuation, selected-limit truth, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier regseqrat real budget separation`

Proof. Start with the source-budget row of Definition 548.420. It records the finite support surface for a Phase Real read and keeps the budget information inside the displayed packet.

Now apply Definition 548.421. This theorem places the $\mathsf{RegSeqRat}^{\uparrow}$ route before the $\mathsf{Real}^{\uparrow}$ endpoint while preserving the same carrier, transport, replay, provenance, and local naming rows.

The two rows are therefore adjacent but not identified. The carrier definition Definition 548.1 displays R and Q as named coordinates and keeps H, C, P, N as support rows rather than equality collapses.

Use the zeta-refusal ledger Definition 548.422 to close the boundary. Host real equality, analytic continuation, selected-limit truth, and ambient completeness have no exported row, so the budget and endpoint remain separate BEDC-visible obligations. □

Theorem 548.426 (CriticalLineWitness StreamName source-window lock)**.** *Let* $W = (Z, S, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{CriticalLineWitness}^{\uparrow}$ *fixed-strip source packet. Any StreamName window consumer that preserves the fixed-strip source must preserve the same finite source-window packet before it reads the Phase Real endpoint. In particular, the consumer reads* $Z, S, M, R, Q, H, C, P, N$ *through the displayed* BHist *packet, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *route reaches the* $\mathsf{Real}^{\uparrow}$ *endpoint.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier streamname source window lock`

Proof. Begin with the source-window side. Apply Definition 548.416. A StreamName-facing consumer preserving the fixed strip must first read the displayed finite source rows and cannot enter through an off-packet stream selector.

Add the source-budget readiness theorem Definition 548.420. It keeps the source window tied to the same BHist, hsame, Cont, Pkg, and local NameCert rows while the Phase Real budget is inspected.

Finally route toward the endpoint through Definition 548.421 and Definition 548.422. These results force the $\mathsf{RegSeqRat}^{\uparrow}$ route, $\mathsf{Real}^{\uparrow}$ endpoint, and refusal ledger to remain downstream from the preserved source-window packet. Therefore the StreamName consumer preserves the finite source window before any Real-facing read is available. □

Theorem 548.427 (CriticalLineWitness Phase Real four-face route exhaustion)**.** *Let $W = (Z, S, M, R, Q, H, C, P,$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ fixed-strip source packet. Every current Phase $\mathsf{Real}^{\uparrow}$ and Completion consumer route exported by W is exhausted by the ordered four-face route*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The route reads the displayed BHist *source packet, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *rows already carried by W. No host real equality, quotient stream, selected limit, analytic truth primitive, or ambient completeness row can enter the exported consumer route.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real four face route exhaustion`

Proof. First bind the CriticalLineWitness selector side. The selector-route totality theorem Definition 548.424 places every Phase Real selector read on the displayed source packet and on the same BHist, hsame, Cont, Pkg, and local NameCert rows.

Next lock the source window before the endpoint is read. By Definition 548.426, any StreamName-facing consumer preserving the fixed strip must preserve the finite source-window packet before the $\mathsf{RegSeqRat}^{\uparrow}$ route reaches the $\mathsf{Real}^{\uparrow}$ endpoint. The budget separation theorem Definition 548.425 keeps the $\mathsf{RegSeqRat}^{\uparrow}$ budget row and the $\mathsf{Real}^{\uparrow}$ endpoint row visible as separate BEDC rows, so the route cannot collapse them into host equality or a selected limit.

Now attach the ambient Real route. The Real four-face dependency route Definition 38.738 gives the ordered passage from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. The order uniqueness theorem Definition 38.739 rules out deletion, permutation, or shortcut coordinates for the current terminal Real surface.

Combining these packets leaves exactly the CriticalLineWitness source rows together with the Real four-face route. The classifier coherence theorem Definition 38.740 identifies the shared finite support as the same BHist, hsame, Cont, Pkg, and NameCert packet. Hence every exported Phase Real and Completion consumer route of W is exhausted by the four displayed faces, and the excluded host-equality, quotient-stream, selected-limit, analytic-truth, and ambient-completeness coordinates have no source row to read. □

Theorem 548.428 (CriticalLineWitness Phase Real selector budget nonescape)**.** *Let $W = (Z, S, M, R, Q, H, C, P,$ be an accepted $\mathsf{CriticalLineWitness}^{\uparrow}$ fixed-strip source packet. Any Phase Real selector preserving the fixed-strip source packet and the four-face L10 route preserves the source-budget and zeta-refusal ledgers as displayed rows of the same* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{CriticalLineWitness}^{\uparrow}}$ *packet. It cannot replace those ledgers by host real equality, analytic truth, quotient stream representatives, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier phase real selector budget nonescape`

Proof. Begin with selector-route totality, Definition 548.424. It places every Phase Real selector on the accepted fixed-strip packet and keeps the source-budget and refusal rows inside the displayed carrier support.

Next use the StreamName source-window lock, Definition 548.426. A selector that preserves the fixed-strip source must preserve the finite source-window packet before any $\mathsf{RegSeqRat}^{\uparrow}$ route reaches the $\mathsf{Real}^{\uparrow}$ endpoint.

Now apply the budget separation theorem Definition 548.425 and the four-face route exhaustion theorem Definition 548.427. These results keep the source-budget row, the regular-rational budget row, and the endpoint row visible as separate BEDC rows along

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Finally attach the zeta-refusal ledger by Definition 548.422. Since the refusal row is displayed in the same carrier and no excluded coordinate is exported by Definition 548.1 or Definition 548.2, the selector has no route to substitute host equality, analytic truth, quotient representatives, selected limits, or ambient completeness for the preserved budget and refusal ledgers. □

The $\mathsf{CriticalLineWitness}^{\uparrow}$ continuation lattice reads critical-line data only after the displayed $\mathsf{ZetaContinuationWitness}^{\uparrow}$ source-lock rows have been exposed. This places the local witness behind the continuation, pole, functional-equation, trivial-zero, Gamma-boundary, transport, replay, provenance, and naming surface of chapter 580.

Theorem 548.429 (CriticalLineWitness Zeta continuation route)**.** *Every* $\mathsf{CriticalLineWitness}^{\uparrow}$ *read offered to a zero-strip consumer must first pass through the* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *continuation route of chapter 580. In particular, the read displays the continuation row, pole ledger, functional-equation row, trivial-zero ledger, Gamma-boundary row, componentwise transport, replay, provenance, and local* NameCert *rows before any zero-strip consumer sees the witness as critical-line data. The route supplies no RH proof, global zero classifier, hidden analytic branch, completed Gamma certificate, host equality of zeta functions, or selected representative outside those rows.*

Proof. Start with the displayed $\mathsf{ZetaContinuationWitness}^{\uparrow}$ packet. Its source-lock surface exposes the continuation, pole, functional-equation, trivial-zero, and Gamma-boundary rows together with transport, replay, provenance, and local naming data.

The $\mathsf{CriticalLineWitness}^{\uparrow}$ consumer can then read only the continuation rows named by that packet. The witness is therefore downstream from the zeta-continuation surface, not parallel to it.

Finally inspect the excluded data. Since no RH proof, global zero classifier, hidden analytic branch, completed Gamma certificate, host equality of zeta functions, or selected representative is a coordinate of the source-lock packet, the zero-strip consumer cannot receive any of those objects through the critical-line witness route. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier zeta continuation route`

Proposition 548.430 (CriticalLineWitness prime and zeta dependency route)**.** *Every* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet that is consumed by an RH-facing refusal route first reads the arithmetic divisibility surface of chapter 64 and the zeta-facing zero-strip source rows displayed by the existing CriticalLineWitness lattice. The prime row supplies the finite divisibility and Euler-product-facing source discipline, while the zeta rows supply zero-strip localization, source windows, and refusal budgets. Only after those rows are present may the RH-facing consumer inspect terminal refusal, fixed-strip, or continuation data.*

Lean4 ✓ BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessPrimeZetaDependencyRoute
↪ variant BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier prime zeta dependency route

Proof. Project the CriticalLineWitness hub through its source, modulus, zero-strip, and zeta sibling routes. Those routes name the arithmetic and analytic source rows before any terminal RH-facing refusal packet is read.

The prime dependency is supplied by chapter 64. Its divisibility and factorization surface is the arithmetic source for zeta-facing rows, not a consequence of a later RH refusal.

The zeta-facing rows then carry fixed-strip and zero-strip source data into the CriticalLineWitness packet. Since the carrier route exposes these rows before terminal refusal, an RH-facing consumer cannot bypass PrimeUp divisibility or the zeta source ledger. □

The CriticalLineWitness$^{\uparrow}$ sibling-dependency route is read only after the zeta source, zero-locus witness, and analytic-continuation boundary have already been displayed by sibling chapters. The zeta source surface of chapter 71 supplies the finite Zeta$^{\uparrow}$ row, the zero-side witness rows are supplied by chapter 73, and the continuation boundary is supplied by chapter 9342. The local witness packet therefore consumes those rows through its displayed BHist, hsame, Cont, Pkg, and NameCert support before any critical-line consumer reads a witness route.

Lemma 548.431 (CriticalLineWitness sibling dependency route)**.** *Every accepted* CriticalLineWitness$^{\uparrow}$ *route that exports a critical-line witness read is supported by the displayed* Zeta$^{\uparrow}$ *source row, the zero-locus witness row, and the analytic-continuation boundary row already present in sibling chapters. The route exports no independent RH proof, global zero classifier, analytic-continuation primitive, quotient representative, host equality, or ambient completeness theorem.*

Lean4 ✓ BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessSiblingDependencyRoute

Proof. First read the zeta source through chapter 71. This supplies the source row before the local critical-line witness classifier is consulted.

Next read the zero-locus and continuation rows through chapter 73 and chapter 9342. These rows provide the zero-facing and boundary data consumed by the local witness packet.

Finally project the local CriticalLineWitness$^{\uparrow}$ support. Its BHist, hsame, Cont, Pkg, and NameCert rows transport and replay the sibling data, but they do not add an RH proof, global zero classifier, analytic-continuation primitive, quotient representative, host equality, or ambient completeness coordinate. □

The CriticalLineWitness$^{\uparrow}$ sibling lattice consumes only displayed zeta rows before any critical-line obstruction is read. Its visible route passes through the finite source and continuation surfaces of chapter 71, chapter 9342, and chapter 580, then returns to the local CriticalLineWitness$^{\uparrow}$ classifier, replay, provenance, and naming rows. This dependency edge supplies no Riemann-hypothesis conclusion, no analytic-continuation closure, no global zero-free region, and no host equality of zeta functions.

Theorem 548.432 (CriticalLineWitness Zeta sibling route)**.** *Every* CriticalLineWitness$^{\uparrow}$ *source-modulus read factors through the displayed* Zeta$^{\uparrow}$ *and zero-strip sibling rows before local* CriticalLineWitness$^{\uparrow}$ *naming is consumed. In particular, the source route must expose the zeta source surface of chapter 71, the continuation handoff of chapter 9342, the zero-facing witness boundary of chapter 580, and the fixed-strip rows already displayed by the local witness packet. The route exports no* RH *proof, global zero classifier, analytic-continuation primitive, host equality, quotient representative, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCarrier source modulus zeta sibling route`

Proof. Project an accepted CriticalLineWitness$^{\uparrow}$ packet as $W = (Z, S, M, R, Q, H, C, P, N)$. The source row Z is the only zeta-facing source coordinate, and the strip row S is the only local zero-strip coordinate available before a modulus read reaches M.

The sibling zeta rows are read before the local name is consumed. The basic zeta source, continuation handoff, and zero-facing witness boundary supply the upstream row data, while the local packet records only the displayed Z, S, M, R, Q route and its structural support.

The local naming row N is terminal for this route. Transport through H, replay through C, and provenance through P preserve the same source-modulus path, so no consumer can replace it by an RH proof, global zero classifier, analytic primitive, host equality, quotient representative, or ambient completeness datum. □

Theorem 548.433 (CriticalLineWitness cross-lattice route)**.** *Every accepted* CriticalLineWitness$^{\uparrow}$ *finite witness route consumed by chapter 705 or chapter 573 must expose the same displayed zeta-zero source row, strip-window row, real-part classifier row, modulus ledger row, and refusal ledger row before either downstream chapter reads the witness as a barrier or a zero-counting input. The route is therefore a finite BEDC handoff through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, not a host analytic assertion about zeros outside the displayed ledgers.*

Proof. Project the CriticalLineWitness$^{\uparrow}$ carrier rows used by the hub siblings. The zeta-zero row supplies the source event, the strip row bounds the critical-window read, the real-part row records the line classifier, and the modulus row keeps the quantitative witness visible to consumers.

For chapter 705, the refusal ledger is the row that prevents a barrier consumer from replacing the displayed witness by an ambient strip theorem. For chapter 573, the same refusal ledger prevents a zero-counting consumer from reading an undisplayed counting formula.

The structural rows hsame, Cont, Pkg, and NameCert transport and name exactly this finite route. Hence both downstream consumers receive the zeta-zero, strip, real-part, modulus, and refusal data before any barrier or counting use is available. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessCrossLatticeRoute`

The CriticalLineWitness$^{\uparrow}$ packet consumes the zero-location and continuation witness surfaces through chapter 73 and chapter 580 before it can expose any critical-line certificate row. This dependency keeps the critical-line readback downstream of displayed zeta-zero and continuation evidence rather than treating the line certificate as an isolated source. The CriticalLineWitness$^{\uparrow}$ zeta–RH dependency route makes the horizontal handoff explicit: the local witness consumes the Zeta$^{\uparrow}$ source and ZetaZeros$^{\uparrow}$ zero rows before any RH-facing boundary chapter can read it.

Theorem 548.434 (CriticalLineWitness zeta–RH dependency route)**.** *Every accepted* CriticalLineWitness$^{\uparrow}$ *packet that exports a critical-line witness to an RH-boundary consumer must first display the finite* Zeta$^{\uparrow}$ *source route of chapter 71, the zero-side witness surface of chapter 73, and the local witness rows* $Z, S, M, R, Q, H, C, P, N$*. The export may then be consumed by the* RHRouteHaltingBoundary$^{\uparrow}$ *packet of chapter 5564, but only as a finite critical-line witness row; it does not supply an* RH *proof, a global zero classifier, analytic-continuation data, quotient representatives, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZetaRhDependencyRoute`

Proof. Read the zeta source first. The sibling surface of chapter 71 supplies the displayed $\mathsf{Zeta}^{\uparrow}$ row before the local critical-line witness packet can expose its own source coordinate.

Next expose the zero-facing route through chapter 73. The local rows $Z, S, M, R, Q, H, C, P, N$ then name the finite witness, strip, modulus, rational comparison, transport, replay, provenance, and local certificate data consumed by $\mathsf{CriticalLineWitness}^{\uparrow}$.

Finally pass the exported row to chapter 5564. That consumer receives the witness as one finite boundary input, so the route remains a BEDC dependency edge rather than an RH proof, global zero classifier, analytic-continuation theorem, quotient representative, host equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.CriticalLineWitnessUp.CriticalLineWitnessZetaRhDependencyCertificate`

Chapter 549

A Concrete Naming Certificate for $\mathsf{KernelAcceptanceWitness}^{\uparrow}$

The $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ surface turns the Lean-side acceptance boundary into a finite BEDC certificate packet. It records the generated candidate declaration, the accepted declaration row, the environment ledger entry, the axiom-query result, refusal evidence for unaccepted candidates, transport and continuation rows, package provenance, and the local naming row. The packet is not a global trust principle for Lean; it is the visible witness that a specific candidate crossed the kernel boundary and that the accepted result has a readable purity ledger.

The $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ packet of chapter 547 consumes the candidate, accepted-declaration, environment-ledger, axiom-query, refusal, transport, continuation, provenance, and local naming rows of this packet and adds build replay plus audit-gate specialization for a single acceptance event.

Definition 549.1 (KernelAcceptanceWitness carrier)**.** *A* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, A, E, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the generated-candidate row, naming the proposed Lean declaration, its source module, its type surface, and the finite generator context that produced it;*
2. *A is the accepted-declaration row, present exactly when the kernel has accepted the declaration named by G;*
3. *E is the environment-ledger row, recording the module position, imported BEDC declarations, and replay boundary at which A is entered into the environment;*
4. *Q is the axiom-query row, recording the transitive dependency query for A and the reported absence or presence of forbidden dependency names;*
5. *R is the refusal-boundary row, recording failed candidates, unresolved names, rejected recursion, forbidden axioms, or dependency reports that prevent a candidate from being exported as accepted;*
6. *H records componentwise* hsame *transports for generated candidate, accepted declaration, environment ledger, axiom-query, refusal-boundary, continuation, package, and local naming rows;*

7. *C records the displayed* Cont *routes by which consumers read acceptance, environment replay, purity query, and refusal evidence;*
8. *P is the* Pkg *provenance over* BHist*, generated candidates, accepted declarations, environment ledger rows, axiom-query rows, refusal rows,* hsame*,* Cont*, and local naming;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceWitness}^{\uparrow}}$ *row.*

The carrier exposes no host-kernel completeness theorem, no hidden proof-producing channel, no acceptance for a candidate absent from A, and no purity claim outside the displayed query row Q.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessTasteGate single carrier alignment`

Theorem 549.2 (KernelAcceptanceWitness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceWitness}^{\uparrow}}$ *surface for the carrier of Definition 549.1 consists of six finite obligations:*

1. *carrier exposure for the packet $K = (G, A, E, Q, R, H, C, P, N)$;*
2. *candidate-to-acceptance matching, stating that the accepted row A names the declaration proposed by G and no unlisted candidate;*
3. *environment replay, stating that E records the module boundary and imported BEDC rows used when A is entered into the environment;*
4. *purity query visibility, stating that Q is the dependency ledger attached to A and records the outcome of the transitive axiom query;*
5. *refusal separation, stating that rows recorded in R are not exported as accepted declarations;*
6. *exactness of the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows for the acceptance and refusal surfaces.*

These obligations bind kernel acceptance as finite certificate data rather than as an ambient trust principle.

Proof. Carrier exposure is projection from Definition 549.1. The rows $G, A, E, Q, R, H, C, P, N$ are the only coordinates available to consumers.

Candidate-to-acceptance matching and environment replay are read from G, A, E. The accepted row is attached to the generated candidate named by G, and the environment ledger records where that accepted declaration is replayed against the visible BEDC imports.

Purity query visibility is read from Q. The dependency report belongs to the accepted declaration row and is carried as a finite ledger entry, so the consumer sees the query result rather than an unstated promise about the environment.

Refusal separation is read from R. Failed candidates and forbidden dependency reports are retained as boundary data but have no coordinate that exports them as accepted declarations.

The transport, continuation, package, and local naming obligations are componentwise over the displayed rows. They can move or read the packet, but they do not add a kernel-completeness theorem or a hidden proof-producing channel. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessPacket semantic name certificate`

Theorem 549.3 (KernelAcceptanceWitness ledger purity)**.** *For an accepted* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *packet, every purity-facing consumer route factors through the accepted-declaration row A, the*

environment-ledger row E, the axiom-query row Q, the displayed BHist, hsame, Cont, Pkg, *and* NameCert *provenance, and the refusal boundary R for candidates that do not cross the acceptance gate. The route exports no declaration as BEDC-internal when its query row reports* Classical.choice, Quot.sound, propext, *an unaccepted candidate, or an unresolved declaration name.*

Proof. Start with the NameCert obligations of Definition 549.2. They attach the accepted declaration to its generated candidate, environment replay row, axiom-query row, and local naming row.

Next read the purity route through Q. The consumer may inspect the transitive dependency report only as the ledger entry paired with A and E, so purity is a displayed row of the packet rather than a separate meta-level assertion.

Finally use the refusal row R. Any candidate with a forbidden dependency report, unresolved name, failed kernel replay, or absent accepted-declaration row remains on the refusal side of the packet and is not exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessLedgerPurity`

Theorem 549.4 (KernelAcceptanceWitness refusal separation)**.** *In an accepted* KernelAcceptanceWitness$^{\uparrow}$ *packet*

$$K = (G, A, E, Q, R, H, C, P, N),$$

any candidate row recorded in R because of unresolved names, forbidden dependency reports, failed kernel replay, or an absent accepted declaration cannot be exported through A, E, or Q as accepted BEDC content.

Proof. Project the carrier of Definition 549.1. The accepted-declaration, environment-ledger, axiom-query, and refusal-boundary rows are separate coordinates of the packet.

Use the NameCert obligations of Definition 549.2. Candidate-to-acceptance matching places accepted content in A, environment replay places its accepted module boundary in E, purity-query visibility places its dependency report in Q, and refusal separation keeps unresolved names, forbidden dependency reports, failed kernel replay, and absent accepted declarations in R.

Finally apply ledger purity from Definition 549.3. Purity-facing consumers read accepted BEDC content only through A, E, Q, while failed candidates remain on the refusal side of the same packet, so no route exports an R row as accepted content. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessRefusalSeparation`

Closure status for NameCert$_{\text{KernelAcceptanceWitness}^{\uparrow}}$.

Constructive story: A KernelAcceptanceWitness$^{\uparrow}$ packet is generated from a candidate declaration row, an accepted declaration row, an environment ledger row, an axiom-query row, refusal-boundary evidence, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KernelAcceptanceWitness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 549.1, Definition 549.2, Definition 549.3, Definition 549.4, Definition 549.5, and the TasteGate single-carrier alignment bridge bind Lean kernel acceptance as a finite NameCert packet over accepted declarations, environment ledger rows, dependency queries, refusal boundaries, and the public export package.

Formal verification: FormalStatus(NameCert$_{\text{KernelAcceptanceWitness}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean implementation, host-kernel completeness, semantic conservativity of all accepted declarations, bridge checking, or acceptance for candidates on the refusal side.

Upgrade path: No further paper-side closure upgrade remains; the remaining independent step is machine bridge checking.

Theorem 549.5 (KernelAcceptanceWitness public export package)**.** *For every accepted* KernelAcceptanceWitness$^{\uparrow}$ *packet*

$$K = (G, A, E, Q, R, H, C, P, N)$$

satisfying the local NameCert obligations of Definition 549.2, the public BEDC acceptance-witness export consists exactly of the generated-candidate row G, accepted-declaration row A, environment-ledger row E, axiom-query row Q, refusal-boundary row R, componentwise transport row H, displayed continuation routes C, package provenance row P, and local naming row N. The row R is exported only as refusal-boundary evidence: no failed candidate, unresolved name, forbidden dependency report, failed replay row, or absent accepted declaration recorded on the refusal side can be exported through A, E, or Q as accepted BEDC content.

Proof. Project the carrier of Definition 549.1. Its displayed packet has exactly the nine coordinates $G, A, E, Q, R, H, C, P, N$. The first four content rows expose the generated candidate, accepted declaration, environment replay, and dependency query; the row R exposes the refusal boundary; and the rows H, C, P, N provide componentwise transport, displayed routes, provenance, and local naming for that same finite packet.

Apply the NameCert obligations of Definition 549.2. Candidate-to-acceptance matching places accepted content in A only for the candidate named by G. Environment replay places the accepted declaration at the boundary recorded by E. Purity-query visibility makes Q the dependency ledger attached to A. Refusal separation keeps failed candidates, unresolved names, rejected recursion, forbidden axioms, and dependency reports on the R side rather than in the accepted-declaration row.

Now use ledger purity from Definition 549.3. Every purity-facing route factors through A, E, Q for accepted content and through R for candidates that do not cross the acceptance gate. Thus a route that tries to read a forbidden dependency report, unresolved declaration name, or unaccepted candidate as BEDC-internal accepted content has no exported coordinate through which to do so.

Finally apply refusal separation from Definition 549.4. Rows recorded in R because of unresolved names, forbidden dependency reports, failed kernel replay, or absent accepted declarations cannot be exported through A, E, or Q as accepted content. The structural rows H, C, P, N preserve, route, source, and name the displayed packet; by the exactness obligation they add no second acceptance channel. Therefore the public export is precisely the nine-row packet $G, A, E, Q, R, H, C, P, N$, with R visible only as boundary evidence and never as accepted BEDC content. □

Lean4 ✓ `BEDC.Derived.KernelAcceptanceWitnessUp.KernelAcceptanceWitnessPublicExportPackage`

Chapter 550

A Concrete Naming Certificate for $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$

The $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ interface names the finite witness packet consumed by the RH capstone. It packages a complex-history candidate together with the displayed critical-strip row, the zeta-zero row, the critical-line classifier boundary, and the local $\mathsf{NameCert}$ routes needed to read the witness as a BEDC object. The packet is a witness surface for the constructive form of Definition 9239.8; it is not a proof of RH and it does not manufacture analytic zeros outside the rows already exposed by $\mathsf{CritStrip}^{\uparrow}$ and $\mathsf{ZetaZeros}^{\uparrow}$.

The strip row is supplied by chapter 72, and the zero row is supplied by chapter 73. The witness packet only synchronizes these sibling rows through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data.

Definition 550.1 (CriticalStripZetaZeroWitness carrier)**.** *A* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, Z, L, B, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the* $\mathsf{CritStrip}^{\uparrow}$ *row carrying the complex history s and the strict-strip evidence for its real component;*
2. *Z is the* $\mathsf{ZetaZeros}^{\uparrow}$ *row carrying the zeta-value and zero-pattern evidence for the same source history s;*
3. *L is the* $\mathsf{CritStrip}^{\uparrow}$ *critical-line classifier row naming the boundary predicate* $\mathsf{OnCritLine}(s)$ *or its displayed absence;*
4. *B is the boundary ledger recording the two open-strip boundary exclusions, the $\Re(s) = 1$ non-vanishing slot, and the critical-line comparison row;*
5. *H records componentwise* hsame *transports for the strip row, zero row, critical-line row, and boundary ledger;*
6. *C records the* Cont *routes for reading the strip witness, reading the zeta-zero witness, exposing the critical-line classifier, and handing the packet to the RH predicate;*
7. *P is the* Pkg *provenance over* $\mathsf{Complex}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}$ *row.*

The carrier stores no host complex number, no analytic zero finder, no quotient equality of zero sets, no selected representative, and no proof or refutation of RH.

Theorem 550.2 (CriticalStripZetaZeroWitness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission: the accepted object is the displayed packet W with strip, zero, critical-line, boundary, transport, continuation, package, and local naming rows;*
2. *shared source binding: the* $\mathsf{CritStrip}^{\uparrow}$ *row S and the* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z expose the same complex-history source through the stored* hsame *and* Cont *routes;*
3. *line-boundary routing: the critical-line row L and boundary ledger B are read only after the strip and zero rows have been exposed;*
4. *RH handoff: any citation of the RH predicate reads the carrier through S, Z, L, B, H, C, P, N;*
5. *ledger exhaustion: the package and local certificate rows exhaust the dependency surface over* $\mathsf{Complex}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket namecert obligations`

Proof. Carrier admission is projection from Definition 550.1. The strip row, zero row, critical-line classifier, boundary ledger, transport rows, continuation rows, package row, and local naming row are explicit coordinates of the packet.

The shared source binding is componentwise. The strip row names the complex history through $\mathsf{CritStrip}^{\uparrow}$, the zero row names the same source through $\mathsf{ZetaZeros}^{\uparrow}$, and the stored hsame and Cont routes are the only routes by which a consumer compares them.

The remaining obligations are downstream reads. The critical-line row and boundary ledger are exposed after the strip and zero rows, and the RH handoff consumes the same finite packet. Since no coordinate is a host complex number, zero finder, quotient equality, selected representative, or proof of RH, the ledger is exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessCarrier namecert obligations`

Theorem 550.3 (CriticalStripZetaZeroWitness critical-line boundary)**.** *Every accepted* CriticalStripZetaZeroWitnes *packet routes the critical-line question through the displayed* $\mathsf{CritStrip}^{\uparrow}$ *classifier row L and the boundary ledger B. A downstream RH consumer may ask whether the source history carried by S and Z is on the critical line, but it receives only the displayed* $\mathsf{OnCritLine}(s)$ *row or its boundary evidence through H, C, P, and N.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket critical line boundary`

Proof. Apply the line-boundary routing obligation of Definition 550.2. It requires the source strip row and zero row to be exposed before the critical-line classifier is read.

The boundary ledger B contains the available boundary exclusions and comparison rows. Therefore a consumer cannot bypass L and B to obtain an ambient real equality, a selected representative for $\Re(s)$, or a host-level decision procedure for the critical line. □

Theorem 550.4 (CriticalStripZetaZeroWitness ledger exhaustion)**.** *For an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *carrier, every fact exported to a* $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or RH-facing consumer is one of the rows* S, Z, L, B, H, C, P, N *or a route between such rows. In particular, the witness packet exports no analytic zero search, no completed zero-set quotient, no host complex equality, no selected counterexample, and no proof of the RH predicate.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket package sameness`

Proof. List the coordinates of Definition 550.1. The only mathematical data rows are the strip row S, the zeta-zero row Z, the critical-line row L, and the boundary ledger B.

Transport can only move these rows through H, and continuation can only expose them through C. The package and local certificate rows P and N record provenance and naming, not additional analytic data.

Thus the consumer surface is exhausted by the finite packet. The excluded objects have no coordinate in the carrier and cannot be reconstructed from the exported rows. □

Theorem 550.5 (CriticalStripZetaZeroWitness concretizes RH)**.** *The* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet is the concrete NameCert surface for the constructive RH premise of Definition 9239.8. Any BEDC proof of the RH predicate may consume such a packet as the finite input carrying* $\mathsf{ZetaZero}(s)$ *and* $\mathsf{InCritStrip}(s)$, *and any constructive counterexample must export such a packet together with a displayed failure of the critical-line row. The packet itself contributes only the input surface; it does not prove the critical-line conclusion.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket concretizes rh`

Proof. The carrier supplies the two premise rows required by the constructive RH form: S gives the critical-strip row and Z gives the zeta-zero row. The shared source binding in Definition 550.2 identifies them as rows over the same complex history.

The critical-line boundary theorem identifies the only route to the conclusion-side classifier, and ledger exhaustion rules out any hidden analytic or quotient data. Hence the packet is exactly the finite input surface for the RH premise. A proof may consume it to produce the critical-line row, while a counterexample must pair it with a displayed failure of that row. □

Theorem 550.6 (CriticalStripZetaZero shared source scope)**.** *In an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet* $W = (S, Z, L, B, H, C, P, N)$, *the* $\mathsf{CritStrip}^{\uparrow}$ *row* S *and the* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z *share a source only through the displayed* hsame *and* Cont *routes recorded in* H *and* C. *A consumer may compare the strip row and zero row by those routes, but it receives no host complex equality, selected representative, quotient zero-set equality, or analytic zero search.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket shared source scope`

Proof. Project the shared source binding obligation from Definition 550.2. It names the strip row and zero row as finite packet coordinates and places their comparison inside the stored transport and continuation rows.

The carrier of Definition 550.1 has no additional coordinate for host complex equality or quotient equality of zero sets. Therefore every shared-source read is exactly a read through H and C. □

Theorem 550.7 (CriticalStripZetaZero line-route determinacy)**.** *Every critical-line read from an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet factors through the finite rows* L, B, H, C, P, N*. The row* L *supplies the displayed critical-line classifier,* B *supplies the boundary ledger,* H *transports the row componentwise,* C *records the continuation route, and* P, N *record provenance and local naming.*

Proof. Apply the critical-line boundary theorem. It requires the critical-line question to pass through L and the boundary ledger B after the strip and zero rows have already been exposed.

The only remaining routes are structural. Transport is recorded in H, continuation in C, package provenance in P, and local naming in N. Thus a critical-line read has a determined route through L, B, H, C, P, N and cannot be replaced by an ambient equality or host decision procedure. □

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket line route determinacy`

Theorem 550.8 (CriticalStripZetaZero RH input ledger)**.** *Any RH-facing consumer of an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet receives only the finite premise packet* S, Z, L, B, H, C, P, N*: the strip row, zero row, critical-line classifier row, boundary ledger, displayed transport, continuation, provenance, and local naming rows. The packet does not export a proof of RH, a zero search procedure, a quotient zero set, host complex equality, or a selected counterexample outside the displayed critical-line row.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket rh input ledger`

Proof. Use Definition 550.5. It identifies the packet as the finite input surface for the constructive RH premise, not as the conclusion itself.

The shared-source scope theorem fixes how S and Z may be compared, and the line-route determinacy theorem fixes how L and B may be read. Ledger exhaustion then states that no additional analytic or quotient coordinate is exported.

Consequently the RH-facing consumer receives exactly the finite premise packet and its displayed routes. Any proof or refutation of the RH predicate must provide further BEDC rows outside this witness packet. □

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessCarrier rh input ledger`

Theorem 550.9 (CriticalStripZetaZero RH premise exhaustion)**.** *Every RH-facing premise consumer of an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet*

$$W = (S, Z, L, B, H, C, P, N)$$

receives exactly the strip row S*, zeta-zero row* Z*, critical-line row* L*, boundary ledger* B*, displayed transports* H*, continuations* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}$ *row* N*. The consumer receives no analytic zero search, quotient zero-set equality, host complex equality, selected representative, or RH proof.*

Proof. Start with the accepted witness carrier and apply Definition 550.2. The RH handoff obligation reads the packet only through the displayed strip, zero, critical-line, boundary, transport, continuation, package, and local naming rows.

The critical-line boundary theorem fixes the route through L and B, while Definition 550.4 exhausts every exported fact by the same finite coordinate list. The concretizes-RH theorem then identifies those rows as the premise surface for the constructive RH statement.

Finally use Definition 550.6 and Definition 550.8. The source comparison between S and Z is only the displayed hsame and Cont route, so the RH-facing premise consumer receives exactly S, Z, L, B, H, C, P, N and nothing outside that packet. □

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket rh premise exhaustion`

Theorem 550.10 (CriticalStripZetaZero constructive counterexample surface)**.** *Any constructive counterexample surface for the RH predicate must extend an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet*

$$W = (S, Z, L, B, H, C, P, N)$$

by a displayed failure of the critical-line row L. The witness packet alone supplies the finite premise surface, but it never supplies analytic zero search, quotient zero-set equality, host complex equality, a selected complex representative, or an RH proof or refutation.

Proof. Apply Definition 550.5. It states that a constructive counterexample pairs the finite premise packet with a displayed failure of the critical-line row, rather than extracting such a failure from the witness packet itself.

The route to the critical-line question is already fixed by Definition 550.3 and Definition 550.7. Thus any failure surface must attach to the displayed row L and its boundary ledger B through the stored H, C, P, N routes.

It remains to exclude hidden analytic data. The ledger exhaustion theorem and Definition 550.9 leave no coordinate for zero search, quotient zero-set equality, host complex equality, selected representatives, or a proof-level RH decision. Therefore a constructive counterexample surface is precisely the premise packet plus an additional displayed failure of L. □

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket constructive counterexample surface`

Theorem 550.11 (CriticalStripZetaZero shared source binding)**.** *For every accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet*

$$W = (S, Z, L, B, H, C, P, N),$$

the CritStrip$^{\uparrow}$ *strip row S and the* ZetaZeros$^{\uparrow}$ *zero row Z share a source only through the displayed* hsame *transports in H and the* Cont *routes in C. The packet provides no host complex equality, quotient zero-set equality, selected representative, or analytic source-selection procedure beyond the finite rows S, Z, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket shared source binding`

Proof. Project the carrier and read the shared-source clause of the local naming surface. The strip row S and zero row Z are both visible, but the only rows connecting their source readings are the componentwise transport row H and the continuation row C.

Apply the existing shared-source scope and source-route exhaustion statements. They confine every comparison between the strip source and zero source to displayed hsame and Cont data.

The package and local naming rows P, N record the finite provenance of that comparison. Since the packet has no coordinate for host complex equality, quotient zero-set equality, selected representatives, or analytic source selection, the shared source is bound only by S, Z, H, C, P, N. □

Theorem 550.12 (CriticalStripZetaZero witness scope binding)**.** *Every scoped read of an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet*

$$W = (S, Z, L, B, H, C, P, N)$$

binds the strip row S, zero row Z, critical-line row L, boundary ledger B, transports H, continuations C, provenance P, and local name N to the displayed $\mathsf{Complex}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surfaces already present in the packet. The scoped read supplies no analytic zero search, quotient zero-set equality, host complex equality, selected representative, or proof of RH.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket scope binding`

Proof. Begin with the carrier and the local NameCert obligations. They identify an accepted witness as the finite packet S, Z, L, B, H, C, P, N and require every RH-facing handoff to read those rows through the displayed BEDC surfaces.

Use shared-source binding for the premise rows. The strip row S and zero row Z can be compared only through the stored hsame transports and Cont routes, so a scoped read binds them to $\mathsf{Complex}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, and $\mathsf{ZetaZeros}^{\uparrow}$ without producing host complex equality or a selected representative.

Use line-route determinacy and the RH input ledger for the conclusion-facing rows. The critical-line row L and boundary ledger B remain attached to $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ through the same packet rows, so the route cannot be replaced by a zero search or quotient zero-set equality.

Finally apply premise exhaustion. The RH-facing consumer receives exactly the finite premise packet, while any proof or refutation of RH must be supplied by an additional accepted BEDC row outside this witness surface. □

Theorem 550.13 (CriticalStripZetaZero line boundary routing)**.** *Let $W = (S, Z, L, B, H, C, P, N)$ be an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet. Every read of the critical-line row L is routed only after the strip row S and the zero row Z have been exposed, and the read is authenticated by the boundary ledger B, displayed* hsame *transports H, continuation routes C, package row P, and local* $\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket line boundary routing`

Proof. Start with the carrier rows. The line row L is not an independent analytic predicate; it appears in the same finite packet as the strip row S, zero row Z, and boundary ledger B.

Use the critical-line boundary and line-route determinacy statements. They force the public route to L through the displayed strip and zero rows before the boundary ledger is consumed.

Apply shared-source binding to the premise side. The source data for S and Z are already restricted to the stored hsame and Cont rows, so the line route cannot bypass them.

Finally, P and N record provenance and local naming for the same finite packet. Therefore every critical-line read is routed by S, Z, L, B, H, C, P, N and not by a host critical-line decision. □

Theorem 550.14 (CriticalStripZetaZero RH handoff obligation)**.** *Every RH-facing handoff from an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet consumes only the finite premise packet*

$$S, Z, L, B, H, C, P, N.$$

The handoff may use the strip row, zero row, critical-line row, boundary ledger, displayed transport, continuation, provenance, and local naming rows, but it does not prove RH, refute RH, search for zeros, close quotient zero sets, or decide host complex equality.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket rh handoff obligation`

Proof. Begin with the RH input ledger and premise exhaustion theorems. They identify the public handoff as a read of the finite packet S, Z, L, B, H, C, P, N and not as an RH-level conclusion.

Use shared-source binding for the premise source. The strip and zero rows share source data only through displayed hsame and Cont routes, so the handoff receives no selected complex representative or analytic source procedure.

Use line-boundary routing for the conclusion-facing row. The critical-line row L is read only after S and Z, and the boundary ledger B authenticates that finite route.

The remaining rows P, N record provenance and local naming. Since no row contains an RH proof, RH refutation, zero-search procedure, quotient equality, or host complex equality, the RH-facing handoff consumes only the finite premise packet. □

Theorem 550.15 (CriticalStripZetaZero boundary ledger non-escape)**.** *For every accepted* CriticalStripZetaZeroWi *packet* $W = (S, Z, L, B, H, C, P, N)$*, every critical-line failure read or boundary read factors through the displayed rows* L, B, H, C, P, N*. The boundary ledger exports only the finite classifier, boundary, transport, continuation, package, and local naming data; it exports no host complex equality, quotient zero-set equality, analytic zero search, or RH decision.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket boundary ledger nonescape`

Proof. Start with the carrier definition and the NameCert obligations theorem. The accepted packet has only the displayed rows S, Z, L, B, H, C, P, N, and the line-boundary routing obligation requires any critical-line read to pass through L and B after the strip and zero rows have been exposed.

Apply ledger exhaustion to the same carrier. It states that every exported fact is either one of the displayed rows or a route between such rows, so no host complex equality, quotient zero-set equality, analytic zero search, or proof-level RH decision can leave the packet.

Finally use shared-source scope and line-route determinacy. Shared-source scope confines comparison of S and Z to the stored hsame and Cont rows, while line-route determinacy confines every critical-line read to L, B, H, C, P, N. Thus a critical-line failure or boundary read cannot escape the finite boundary ledger. □

Theorem 550.16 (CriticalStripZetaZero source route exhaustion)**.** *For an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet* $W = (S, Z, L, B, H, C, P, N)$*, the common source route for the strip row* S *and the zeta-zero row* Z *is exhausted by the stored* hsame *and* Cont *rows in* H *and* C*. Any consumer comparing the two source readings must use those displayed routes and receives no additional complex representative, quotient equality, or analytic source-selection procedure.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket source route exhaustion`

Proof. Project the carrier and the shared source binding clause of the NameCert obligations theorem. The only rows that connect the $\mathsf{CritStrip}^{\uparrow}$ source and the $\mathsf{ZetaZeros}^{\uparrow}$ source are the componentwise hsame transports in H and the continuation routes in C.

The shared-source scope theorem gives the same restriction at the consumer boundary. A comparison between S and Z is a displayed route comparison, not a host equality of complex numbers or a quotient equality of zero sets.

Ledger exhaustion and premise exhaustion close the route inventory. Since every RH-facing premise consumer receives only S, Z, L, B, H, C, P, N, the shared source of S and Z has no route outside the stored hsame and Cont rows. □

Theorem 550.17 (CriticalStripZetaZero premise consumer separation)**.** *Every RH-facing consumer of an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet receives the premise packet only as*

$$S, Z, L, B, H, C, P, N.$$

The consumer may read the strip row, zero row, critical-line classifier, boundary ledger, displayed transport, continuation, provenance, and local naming rows, but any critical-line conclusion, RH proof, RH refutation, or analytic zero decision must be supplied outside the witness packet.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket premise consumer separation`

Proof. Begin with carrier admission and the NameCert obligations theorem. They identify the accepted object as exactly the finite packet S, Z, L, B, H, C, P, N and require the RH handoff to read that same packet rather than a host analytic object.

Use shared-source scope and source route exhaustion for the premise side. The strip and zero rows share their source only through H and C, so a consumer receives a displayed finite route and not a selected complex representative.

Use line-route determinacy and boundary ledger non-escape for the conclusion-facing side. They force every critical-line read through L, B, H, C, P, N and prevent the boundary ledger from exporting a host equality, quotient zero-set equality, zero search, or RH decision.

Finally apply premise exhaustion. The RH-facing consumer receives exactly the finite premise packet; therefore any critical-line conclusion or RH-level decision must be supplied by an additional BEDC row outside the witness packet. □

Theorem 550.18 (CriticalStripZetaZero witness zero-row admission)**.** *Let* $W = (S, Z, L, B, H, C, P, N)$ *be an accepted* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet. The displayed row* Z *may be admitted as zero-witness certificate data inside the finite* BHist *packet when its local* NameCert *row* N*,* Pkg *provenance* P*, componentwise* hsame *transports* H*, and displayed* Cont *routes* C *expose the same source boundary as* S, L, B*.*

This admission is only a certificate-data row. It asserts neither that an analytic zeta zero exists in the host complex plane nor that any RH-facing conclusion has been proved.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket zero row admission`

Proof. Project the carrier of the accepted packet. The only candidate zero-witness data available to the packet is the displayed row Z together with the finite rows S, L, B, H, C, P, N.

Inspect the local ledger. The hsame rows and Cont routes connect Z to the displayed source and boundary rows, while P and N record provenance and local naming. This is enough to admit Z as certificate data in the finite BHist packet.

No analytic existence statement is exported by that projection. The packet contains no host zero search, no quotient zero-set equality, and no RH decision row, so the admission remains a finite witness-row admission only. □

Theorem 550.19 (CriticalStripZetaZero witness classifier non-escape)**.** *For accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packets* $W = (S, Z, L, B, H, C, P, N)$ *and* $W' = (S', Z', L', B', H', C', P', N')$ *that classify together, classifier transport across the finite* BHist *rows may transport only the displayed witness rows, boundary rows,* hsame *rows,* Cont *routes,* Pkg *provenance, and* NameCert *evidence.*

Such transport cannot create analytic-continuation truth, Euler-product truth, zero-existence truth, RH truth, or RH refutation rows. Any such row must be supplied by a separate accepted packet outside the witness classifier.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket classifier nonescape`

Proof. Begin with the classifier-facing rows of the two accepted packets. Classification compares the displayed finite coordinates and routes the comparison through the stored hsame and Cont data.

Inspect the package and naming ledgers. They record provenance for the witness row, critical-strip boundary row, premise ledger, and local certificate; they do not contain analytic-continuation, Euler-product, zero-existence, RH-proof, or RH-refutation coordinates.

Therefore classifier transport can only replay the finite rows already present in W and W'. It cannot manufacture an analytic theorem row or settle RH, because no such coordinate is present in the carrier or ledger being transported. □

Theorem 550.20 (CriticalStripZetaZero witness consumer boundary)**.** *Every downstream consumer of an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet* $W = (S, Z, L, B, H, C, P, N)$ *may read only the displayed witness row* Z*, the critical-strip boundary rows* S, L, B*, the premise ledger, the local* NameCert *row* N*, the* Pkg *provenance* P*, and the connecting* hsame *and* Cont *rows.*

The consumer boundary supplies no analytic continuation theorem, no Euler product theorem, no zero search, no host complex equality, no RH proof, and no RH refutation.

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket consumer boundary`

↪ variant `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket witness consumer boundary`

Proof. Project the consumer-facing packet. The rows exposed to consumers are exactly S, Z, L, B, H, C, P, N, with the witness row, boundary rows, premise ledger, package provenance, and local certificate displayed in the finite carrier.

Use Definition 550.17. It already forces RH-facing consumers to receive the premise packet only as the listed finite rows, with any critical-line conclusion supplied outside the witness packet.

Combine this with zero-row admission and classifier non-escape. The first theorem admits Z only as certificate data, and the second prevents classifier transport from creating analytic or RH

truth rows. Thus downstream consumers remain at the displayed witness boundary and receive no claim that a zero exists or that RH is settled. □

Theorem 550.21 (CriticalStripZetaZero scoped package)**.** *For an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet*

$$W = (S, Z, L, B, H, C, P, N),$$

the carrier, NameCert *obligations, shared-source scope, line-route determinacy, premise exhaustion, zero-row admission, classifier non-escape, and consumer boundary assemble one scoped finite packet over* Complex$^{\uparrow}$, CritStrip$^{\uparrow}$, ZetaZeros$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*. The packet exposes only the strip row S, zero row Z, critical-line row L, boundary ledger B, transports H, routes C, provenance P, and local name N.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket scoped package`

Proof. Start from the carrier and NameCert obligation rows. They identify the accepted witness as the finite packet S, Z, L, B, H, C, P, N and bind the strip, zero, critical-line, boundary, transport, route, provenance, and naming data to the same local carrier.

Next use shared-source scope and line-route determinacy. These theorems force comparisons between strip and zero rows through the displayed H and C routes and force critical-line reads through L and B.

Apply premise exhaustion, zero-row admission, classifier non-escape, and the consumer boundary. The premise packet is therefore closed under the displayed rows, while zero data and classifier transport cannot create analytic zero truth, RH truth, or RH refutation.

The resulting surface is a scoped finite packet over the listed BEDC rows. It is not a host complex equality, a quotient zero-set equality, or an analytic zero decision. □

Theorem 550.22 (CriticalStripZetaZero source boundary closure)**.** *For an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet $W = (S, Z, L, B, H, C, P, N)$, source comparison is closed by S, Z, H, C, P, N. A consumer comparing the critical-strip source and the zeta-zero source receives the displayed strip row, zero row, transports, routes, provenance, and local name, and cannot turn that comparison into host complex equality or quotient zero-set equality.*

Lean4 ✓ `BEDC.Derived.CriticalStripZetaZeroWitnessUp.CriticalStripZetaZeroWitnessPacket source boundary closure`

Proof. Project the source side of the carrier. The only rows connecting the strip source S and the zero source Z are the transport rows H, continuation routes C, provenance row P, and local name N.

Apply shared-source scope and source route exhaustion. They state that every source comparison must use the displayed H and C rows rather than a selected host complex representative or quotient zero-set representative.

Use package provenance and the consumer boundary to close the comparison. The provenance and naming rows authenticate only the finite packet, so source comparison remains the S, Z, H, C, P, N boundary and exports no host or quotient equality. □

Theorem 550.23 (CriticalStripZetaZero RH handoff scope)**.** *Every RH-facing handoff from an accepted* CriticalStripZetaZeroWitness$^{\uparrow}$ *packet reads only*

$$S, Z, L, B, H, C, P, N.$$

Any critical-line conclusion, proof, refutation, or analytic zero decision must be supplied by a separate BEDC row outside the witness packet.

Proof. Begin with premise consumer separation. It already routes RH-facing consumers through the displayed strip, zero, critical-line, boundary, transport, continuation, provenance, and naming rows.

Use boundary ledger non-escape and line-route determinacy. These theorems keep the critical-line and boundary reads inside L, B, H, C, P, N and prevent a boundary read from becoming a host equality, quotient equality, analytic zero search, or RH decision.

Finally apply scoped package closure. Since the handoff packet has no coordinate for a critical-line conclusion, proof, refutation, or analytic zero decision, any such row must come from a separate accepted BEDC packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ packet is generated from a $\mathsf{CritStrip}^{\uparrow}$ strip row, a $\mathsf{ZetaZeros}^{\uparrow}$ zero row, a critical-line classifier row, a boundary ledger, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 550.21, Definition 550.22, and the RH handoff rows close the finite packet scope over $\mathsf{Complex}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove RH, refute RH, construct non-trivial zeta zeros, decide the critical-line classifier, close zero-set quotients, or bridge checking.

Upgrade path: Scoped closure requires checked carrier encoding for the shared source binding, critical-line boundary routing, RH handoff, and ledger exhaustion rows.

Chapter 551

A Concrete Naming Certificate for $\mathsf{HostDelegationSocket}^{\uparrow}$

The $\mathsf{HostDelegationSocket}^{\uparrow}$ surface records host Lean delegation as a finite naming packet rather than as an ambient trust claim. It is the certificate face at which paper markers, audit rows, host-kernel delegation, target resolution, transport, continuation, provenance, boundary ledgers, and local naming are made visible to downstream audit and MetaCIC chapters.

The socket consumes the accepted-declaration rows of chapter 557 and the displayed proof-route rows of chapter 582 before exposing a host-delegation boundary. The kernel acceptance trace supplies the accepted stamp, environment, axiom ledger, and classifier rows, while the inscribed route supplies the displayed route, GAP row, consumer row, and local $\mathsf{NameCert}$ row used by the socket-facing audit surface.

Definition 551.1 (HostDelegationSocket carrier)**.** *A* $\mathsf{HostDelegationSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, A, K, D, T, C, P, L, N)$$

with the following displayed rows:

1. *M is the manuscript marker row, recording the paper site, marker class, literal target text, and local citation context;*
2. *A is the audit-face row, recording source-token scans, marker-resolution checks, dependency-purity checks, and origin-flow checks relevant to the marker;*
3. *K is the host-kernel delegation row, recording that the named Lean declaration is delegated to the host Lean kernel as an audited boundary row;*
4. *D is the displayed target-resolution row, recording the exact declaration name, namespace path, resolution result, and refusal result when resolution fails;*
5. *T records componentwise* hsame *transports for marker, audit face, delegation row, target-resolution row, continuation row, provenance row, boundary ledger, and local naming row;*
6. *C records the displayed* Cont *routes by which consumers inspect marker resolution, audit evidence, host-kernel delegation, transport replay, provenance, and boundary refusal;*
7. *P is the* Pkg *provenance over* BHist*, manuscript marker rows, audit faces, host-kernel delegation rows, target-resolution rows,* hsame*,* Cont*, and local naming;*

8. *L is the boundary ledger, recording non-escape for host truth predicates, global CIC completeness, source axioms,* Classical.choice, Quot.sound, propext, *fuzzy marker matching, and bridge-checked status;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for a host truth predicate, no proof that the host CIC is globally complete, no permission to accept source axioms or forbidden host dependencies, no fuzzy resolver, and no bridge-checked certificate.

Lean4 (definition) `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocketCarrier`

Theorem 551.2 (HostDelegationSocket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$ *surface for the carrier of Definition 551.1 consists of carrier admission for*

$$H = (M, A, K, D, T, C, P, L, N),$$

manuscript-marker exposure, audit-face exactness, host-kernel delegation visibility, displayed target-resolution exactness, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, boundary-ledger non-escape, and local naming.*

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocketCarrier semantic name certificate`

Proof. Carrier admission is projection from Definition 551.1. The rows $M, A, K, D, T, C, P, L, N$ are the only coordinates available to the local certificate.

The marker, audit, delegation, and resolution obligations are read directly from M, A, K, D. The host Lean kernel remains visible as the audited delegation row K; it is not silently replaced by an internal BEDC truth predicate.

Transport and replay are confined to T and C, while P and N record provenance and local naming. The ledger L supplies the non-escape clauses for host truth predicates, global CIC completeness, source axioms, forbidden host dependencies, fuzzy marker matching, and bridge-checked status. □

Theorem 551.3 (HostDelegationSocket audit-face separation)**.** *For an accepted* $\mathsf{HostDelegationSocket}^{\uparrow}$ *packet, the audit face A and the host-kernel delegation row K remain separate public rows. A downstream consumer may inspect the audit evidence and may inspect the delegated host-kernel target, but it may not identify audit success with host truth, global CIC completeness, source-axiom admission, or absence of* Classical.choice, Quot.sound, *and* propext *beyond the displayed dependency ledger.*

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocket audit face separation`

Proof. Project the finite packet and read A and K as distinct coordinates. The audit face records the gate evidence, while the delegation row records the host-kernel target to which the marker points.

The two rows are transported together only through the componentwise hsame row T and replayed only through C. These routes preserve the separation rather than creating a predicate equating audit evidence with host truth.

The boundary ledger L then blocks every stronger reading. It refuses host truth predicates, global CIC completeness, source axioms, and unrecorded claims about forbidden dependency names unless those claims appear as the displayed audit result of the packet. □

Theorem 551.4 (HostDelegationSocket Lean marker boundary)**.** *For an accepted* $\mathsf{HostDelegationSocket}^{\uparrow}$ *packet, the manuscript marker row M reaches the local naming row N only through the displayed target-resolution row D, the host-kernel delegation row K, the transport row T, the continuation row C, the package provenance P, and the boundary ledger L. If D refuses the target, the packet exports no replacement by similar spelling, neighboring namespace, informal prose equivalence, or consumer-selected theorem.*

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocket marker boundary`

Proof. Start with the marker row M. The only route from the paper site to local naming is the displayed resolution row D, which records the exact declaration name and the resolve-or-refuse result.

If resolution succeeds, K records the host-kernel delegation and T, C, P, N transport, replay, package, and name the same route. No step in this chain introduces an alternate declaration name.

If resolution fails, D records refusal and L keeps that refusal at the boundary. The carrier has no coordinate for spelling similarity, namespace guessing, prose reinterpretation, or consumer-chosen substitution, so the refused marker cannot be exported as a local certificate. □

Theorem 551.5 (HostDelegationSocket non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{HostDelegationSocket}^{\uparrow}$ *packet factors through the finite rows*

$$M, A, K, D, T, C, P, L, N.$$

The consumer may read marker text, audit evidence, host-kernel delegation, target resolution, transport replay, continuation routes, package provenance, boundary refusal, and the local NameCert row, but it cannot extract a host truth predicate, global CIC completeness theorem, source axiom, forbidden host dependency, fuzzy marker match, or bridge-checked status.

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocketNonEscape`

Proof. Let a downstream consumer route be given. By Definition 551.1, its source packet is the finite tuple $M, A, K, D, T, C, P, L, N$.

The consumer reads the public certificate through those rows. The marker and resolution rows identify the target, the audit and delegation rows expose the host boundary, and the transport, continuation, provenance, and naming rows replay the certificate without adding new coordinates.

The non-escape claim is the carrier inventory plus the ledger row L. None of the excluded objects is stored in the tuple, and L explicitly records that these objects are refused as exported substrate content. □

Theorem 551.6 (HostDelegationSocket primitive scope binding)**.** *For an accepted* $\mathsf{HostDelegationSocket}^{\uparrow}$ *packet, marker exposure, audit-face separation, target-resolution exactness, host-kernel delegation visibility, transport replay, provenance, and boundary-ledger non-escape are scoped only by the finite* BHist *rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, the local* $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$ *row, and the imported* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *and* $\mathsf{InscribedRoute}^{\uparrow}$ *surfaces. No consumer may widen this scope to host truth predicates, global CIC completeness, source axioms, forbidden host dependencies, fuzzy marker matching, or bridge-checked status.*

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocketPrimitiveScopeBinding`

Proof. Project the carrier of Definition 551.1. Its rows expose the manuscript marker, audit face, host-kernel delegation, target resolution, transport, replay, provenance, ledger, and local naming surface as a single finite packet.

Apply audit-face separation and the Lean marker boundary. These force audit evidence, delegated target, and exact resolution to remain visible as rows of the same packet rather than as ambient host predicates or spelling-based substitutions.

Finally apply non-escape. The ledger row refuses every object outside BHist, hsame, Cont, Pkg, $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$, $\mathsf{KernelAcceptanceTrace}^{\uparrow}$, and $\mathsf{InscribedRoute}^{\uparrow}$, so the primitive socket scope is exactly the displayed carrier surface. □

Lean4 ✓ `BEDC.Derived.HostDelegationSocketUp.HostDelegationSocket primitive scope binding`

Theorem 551.7 (HostDelegationSocket public interface)**.** *An accepted* $\mathsf{HostDelegationSocket}^{\uparrow}$ *packet exports a consumer-facing socket interface consisting of the manuscript marker row M, audit face A, exact target-resolution row D, transport row T, continuation row C, provenance row P, boundary ledger L, and local naming row N. This interface lets downstream chapters replay marker resolution, inspect audit evidence, read the delegated host-kernel target, and transport the local certificate, while exporting no host truth predicate, global CIC completeness assertion, source axiom, forbidden host dependency, fuzzy resolver, or bridge-checked status.*

Proof. Project the carrier of Definition 551.1. The public rows visible to a consumer are exactly the displayed marker, audit, delegation, target-resolution, transport, continuation, provenance, ledger, and naming coordinates.

Use audit-face separation to keep A apart from the host-kernel delegation row, and use the marker boundary to force every marker readback through the exact target-resolution row D. The transport and continuation rows then replay the same finite route without introducing a second resolver.

Finally apply primitive scope binding and non-escape. These results confine the exported socket to the displayed BHist, hsame, Cont, Pkg, imported acceptance and route surfaces, and local $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$ row; the ledger L refuses every stronger host-level predicate or forbidden dependency claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}$.

Constructive story: A $\mathsf{HostDelegationSocket}^{\uparrow}$ packet is generated from a manuscript marker row, audit-face row, host-kernel delegation row, displayed target-resolution row, componentwise hsame transports, displayed Cont routes, Pkg provenance, a boundary ledger, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 551.1, Definition 551.2, Definition 551.3, Definition 551.4, Definition 551.5, Definition 551.6, and Definition 551.7 specify the finite host delegation socket carrier, local naming obligations, audit separation, marker boundary, transport replay, provenance, primitive scope, non-escape ledger, and consumer-facing public socket interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HostDelegationSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host truth predicates, global CIC completeness, bridge checking, source-axiom admission, forbidden host-dependency admission, fuzzy marker matching, or Lean-side formal verification for $\mathsf{HostDelegationSocket}^{\uparrow}$.

Upgrade path: Lean verification requires a checked public-interface theorem and an audit-clean bridge surface for $\mathsf{HostDelegationSocket}^{\uparrow}$.

Chapter 552

A Concrete Naming Certificate for $\mathsf{TasteGateAuditTrace}^{\uparrow}$

The $\mathsf{TasteGateAuditTrace}^{\uparrow}$ packet is the concrete audit-trace surface for the theory-stability gate of chapter 9459. It records an AI-origin chapter admission trace as a finite NameCert object whose rows display the four TasteGate obligations, replayable audit evidence, ground-compiler provenance, and a ledger preventing the trace from exporting hidden approval.

The trace consumes the compiler-facing witness of chapter 563 for its event-flow and compiler-provenance rows, and it consumes the audit-gate boundary of chapter 526 for refusal and replay boundary rows. Those sibling surfaces supply the concrete routes reread by the audit trace rather than a detached kernel-only admission fact.

Definition 552.1 (TasteGate audit trace carrier)**.** *A* $\mathsf{TasteGateAuditTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, O, Q, A, R, G, H, C, P, L)$$

with the following displayed rows:

1. *X is the candidate chapter identity row, recording the chapter label, proposed carrier name, and concrete paper site;*
2. *O is the AI-origin row, recording that the trace is evaluated on the AI-proposed admission track;*
3. *Q is the four-obligation* $\mathsf{ChapterTasteGate}$ *row, displaying conservativity, no-hidden-input, round-trip, and layer-separation witnesses for the same candidate;*
4. *A is the audit replay row, storing the finite replay route by which the admission check rereads $X, O, Q, R, G, H, C, P, L$;*
5. *R is the ground-compiler provenance row, listing the finite event-flow image and named compiler facts used by the no-hidden-input, round-trip, and layer-separation reads;*
6. *G is the GAP accounting row for absent carrier, absent classifier, absent TasteGate witness, absent replay, absent compiler provenance, absent transport, absent package, and attempted hidden approval cases;*
7. *H stores componentwise* hsame *transports for every displayed row of the packet;*
8. *C records the* Cont *routes that expose the candidate identity, AI-origin row, TasteGate obligations, audit replay, compiler provenance, and GAP accounting to a consumer;*

9. *P is the* Pkg *provenance over* BHist, NameCert, ChapterTasteGate, *audit replay rows, ground-compiler provenance rows, GAP accounting,* hsame, Cont, *and local naming;*
10. *L is the local naming row for* $\mathsf{NameCert}_{\mathsf{TasteGateAuditTrace}^{\uparrow}}$.

The carrier has no coordinate for hidden prompt state, host-side mathematical approval, an oracle classifier, an unlisted compiler theorem, or a chapter admission fact detached from the displayed audit rows.

Lean4 (definition) `BEDC.Derived.TasteGateAuditTraceUp.TasteGateAuditTraceUp`
Lean4 ✓ `BEDC.Derived.TasteGateAuditTraceUp.TasteGateAuditTraceUp single carrier alignment`

Definition 552.2 (TasteGate audit trace classifier)**.** *The classifier of a* $\mathsf{TasteGateAuditTrace}^{\uparrow}$ *packet accepts precisely those finite packets*

$$T = (X, O, Q, A, R, G, H, C, P, L)$$

whose rows display the same candidate chapter identity, whose AI-origin row O is present, whose ChapterTasteGate *row Q contains all four obligation witnesses, whose audit replay row A rereads the displayed rows only, whose ground-compiler provenance row R names the compiler evidence used by Q, and whose GAP row G accounts for every refused route seeking a missing witness or hidden approval channel.*

Theorem 552.3 (TasteGate audit trace NameCert obligations)**.** *For every accepted* $\mathsf{TasteGateAuditTrace}^{\uparrow}$ *carrier*

$$T = (X, O, Q, A, R, G, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{TasteGateAuditTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for T, candidate identity through X, AI-origin exposure through O, four-obligation* ChapterTasteGate *exposure through Q, audit replay through A, ground-compiler provenance through R, refused-route accounting through G, componentwise transport through H, consumer routes through C, package provenance through P, and local naming through L.*

Proof. Carrier admission and classification are projections from Definition 552.1 and Definition 552.2. The packet exposes exactly the candidate identity, origin row, TasteGate row, replay row, compiler provenance row, refusal ledger, transport rows, continuation routes, package provenance, and local naming row.

The four TasteGate obligations are all read from Q. Because Q belongs to the same finite carrier as X and O, a consumer cannot detach the obligation witnesses from the candidate chapter or from the AI-origin admission track.

The replay and provenance obligations are supplied by A and R. The row A records how the trace is reread, while R records the event-flow image and compiler facts supporting the no-hidden-input, round-trip, and layer-separation reads.

Finally G, H, C, P, L account for refused reads, preserve the same rows under transport, expose only named consumer routes, record package provenance, and name the certificate locally. No further NameCert obligation is available outside the displayed packet. □

Theorem 552.4 (TasteGate audit trace ground compiler boundary)**.** *Every accepted* $\mathsf{TasteGateAuditTrace}^{\uparrow}$ *route that consumes the four-obligation row Q must pass through the audit replay row A and the ground-compiler provenance row R. The route may export the displayed conservativity, no-hidden-input, round-trip, and layer-separation witnesses for the candidate chapter, but it cannot cite hidden prompt state, an unlisted compiler theorem, or host-side approval as part of the trace.*

Proof. Start with the carrier in Definition 552.1. The only row carrying TasteGate obligations is Q, and the only rows that make those obligations auditable are A and R.

The replay row A binds a consumer read to the finite packet. Any read of Q is therefore a replay of the displayed candidate identity, AI-origin row, TasteGate row, provenance row, transport rows, continuation rows, package row, and naming row.

The provenance row R supplies the compiler-facing boundary. It names the finite event-flow image and compiler facts used by the no-hidden-input, round-trip, and layer-separation witnesses.

Since the carrier contains no row for hidden prompt state, unlisted compiler facts, or host-side approval, these objects cannot be exported by an accepted audit trace. □

Theorem 552.5 (TasteGate audit trace non-escape)**.** *Every consumer of an accepted* $\mathsf{TasteGateAuditTrace}^{\uparrow}$ *packet factors through the finite rows*

$$(X, O, Q, A, R, G, H, C, P, L).$$

The consumer can read the AI-origin chapter identity, the four TasteGate obligation witnesses, audit replay, ground-compiler provenance, refused-route accounting, transport, continuation, package, and local naming data. It cannot extract hidden prompt state, oracle approval, an unnamed chapter object, a host-side semantic judgment, or a compiler fact absent from R.

Proof. Project the consumer route through the carrier. Its available coordinates are exactly $X, O, Q, A, R, G, H, C, P, L$.

Use Definition 552.3 for the NameCert surface. It accounts for the candidate identity, AI-origin row, four TasteGate witnesses, replay row, compiler provenance, refused routes, transports, continuations, package provenance, and local naming.

Use Definition 552.4 for the compiler-facing boundary. It keeps every read of the four-obligation row attached to replay and to the displayed ground-compiler provenance.

The forbidden outputs have no coordinate in the packet and no continuation route in C. Therefore any route exporting them would leave the accepted $\mathsf{TasteGateAuditTrace}^{\uparrow}$ carrier rather than consume it. □

Closure status for $\mathsf{NameCert}_{\mathsf{TasteGateAuditTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TasteGateAuditTrace}^{\uparrow}$ packet is generated from a candidate chapter identity row, AI-origin row, four-obligation ChapterTasteGate row, audit replay row, ground-compiler provenance row, GAP accounting row, componentwise transports, continuation routes, package provenance, and local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TasteGateAuditTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 552.1, Definition 552.2, Definition 552.3, Definition 552.4, and Definition 552.5 bind the seed carrier, classifier, local NameCert surface, compiler boundary, and non-escape ledger for AI-origin TasteGate audit traces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TasteGateAuditTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove the candidate chapter's mathematical content, certify host semantics, close bridge checking, or provide Lean verification for the audit-trace packet.

Upgrade path: Obligation closure requires checked carrier, classifier, NameCert-surface, compiler-boundary, replay, ledger, and non-escape theorems for the same finite audit-trace packet.

Chapter 553

A Concrete Naming Certificate for $\mathsf{GeneratorInductionSeal}^{\uparrow}$

The $\mathsf{GeneratorInductionSeal}^{\uparrow}$ surface names the generator-internal induction packet that remains inside an authorized generator while recording every attempted passage to free-floating induction as Gap residue. It combines the authorized generator row, the continuation witness row required by the generator-closure classifier, the closure classifier row itself, the residue ledger, transport rows, continuation routes, package provenance, and local naming into one finite certificate.

The seal substantively consumes the authorized-generator route of chapter 525 and the closure classifier surface of chapter 486 before it records the residue ledger. The induction seal reads an authorized generator row and a generator-closure classifier row before it can expose any free-floating-induction refusal.

Definition 553.1 (Generator induction seal carrier)**.** *A* $\mathsf{GeneratorInductionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, W, K, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the authorized generator row whose constructor-list provenance supplies the only internal induction authority;*
2. *W is the continuation witness row carried by every accepted generator-internal continuation;*
3. *K is the closure classifier row that checks whether a continuation remains inside the closure of G with the witness W;*
4. *R is the* Gap *ledger row recording every requested passage from the displayed generator-internal route to an unscoped induction claim;*
5. *H records componentwise* hsame *transports for G, W, K, R, package provenance, and the local naming row;*
6. *C records the displayed* Cont *routes for reading the generator row, replaying the witness-carrying continuation, consulting the closure classifier, emitting residue, and exporting the local packet;*
7. *P is the* Pkg *provenance over* BHist*, authorized generator rows, continuation witnesses, closure classifier rows,* Gap *ledger rows,* hsame*,* Cont*, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorInductionSeal}^{\uparrow}}$ *row.*

The carrier contains no induction rule over unnamed objects, no erased residue, and no route that turns a continuation lacking W into a continuation of G.

Lean4 ✓ `BEDC.Derived.GeneratorInductionSealUp.GeneratorInductionSealTasteGate single carrier alignment`

Theorem 553.2 (Generator induction seal NameCert obligations)**.** *For the carrier of Definition 553.1, the local* $\mathsf{NameCert}_{\mathsf{GeneratorInductionSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (G, W, K, R, H, C, P, N),$$

authorized-generator exposure, continuation-witness preservation, closure-classifier lookup, Gap *residue retention, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and NameCert export for the same finite packet.*

Proof. Carrier admission is projection from Definition 553.1. The packet exposes only the eight displayed rows G, W, K, R, H, C, P, N.

Generator authority is read from G, witness-carrying continuation from W, and closure classification from K. These rows determine which induction routes remain internal to the authorized generator.

Residue, transport, replay, and provenance are confined by R, H, C, P. The local row N exports only this packet as a naming certificate, so no projection can add a hidden universal induction principle. □

Theorem 553.3 (Generator induction seal continuation witness)**.** *Every continuation accepted by a* $\mathsf{GeneratorInductionSeal}^{\uparrow}$ *packet carries the displayed witness row W and is checked by the closure classifier row K against the authorized generator row G. A continuation that cannot replay W through the displayed* Cont *route is not a continuation of this seal.*

Proof. Read the proposed continuation through C. If it is accepted, the route reaches G, replays W, and consults K before the local certificate row is exported.

The transport row H preserves the same generator, witness, classifier, residue, package, and naming coordinates. Thus a transported accepted continuation still carries the displayed witness.

If the route cannot replay W, no carrier coordinate classifies it as a continuation of G. It is therefore outside the seal rather than an induction step of the packet. □

Theorem 553.4 (Generator induction seal GAP non-escape)**.** *Whenever a requested continuation is not classified by K as a witness-carrying continuation of the authorized generator row G, the* $\mathsf{GeneratorInductionSeal}^{\uparrow}$ *packet records the request only as* Gap *residue in R. The residue may be transported, replayed as a refusal, packaged, and named, but it cannot become an internal induction theorem.*

Proof. Inspect the classifier row K. If the request is not accepted as a continuation of G carrying W, the only coordinate that records it is the residue row R.

Transport through H preserves residue as residue. Replay through C can expose the refusal and package it through P, while N names the same boundary.

The carrier has no conversion row from residue to induction authority. Therefore the Gap ledger cannot export an internal induction theorem or a free-floating closure principle. □

Theorem 553.5 (Generator induction seal public package)**.** *The public export of a* $\mathsf{GeneratorInductionSeal}^{\uparrow}$ *packet is exactly the finite NameCert packet over the authorized generator row G, continuation witness row W, closure classifier row K,* Gap *ledger row R, componentwise* hsame *transports, displayed* Cont *routes, and* Pkg *provenance. A consumer can recover generator authority, witness status, classifier status, and residue location, but no induction object beyond those rows.*

Proof. Start with the carrier tuple $S = (G, W, K, R, H, C, P, N)$. The export row N names this tuple and no larger surface.

A consumer recovers generator authority from G, witness status from W, classifier status from K, and residue location from R. The same consumer uses H for componentwise transport, C for replay, and P for provenance.

These rows exhaust the public package. Since S has no coordinate for an unnamed induction principle, hidden eliminator, or erased residue, the NameCert packet cannot export such content. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorInductionSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{GeneratorInductionSeal}^{\uparrow}$ packet is generated from an authorized generator row, a continuation witness row, a closure classifier row, a Gap ledger row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GeneratorInductionSeal}^{\uparrow}}$, seed)

Scope closed: Definition 553.1, Definition 553.2, Definition 553.3, Definition 553.4, and Definition 553.5 bind the finite carrier, witness-carrying continuation route, classifier boundary, residue ledger, and NameCert export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GeneratorInductionSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, induction over unnamed objects, erased GAP residue, or continuations outside the displayed generator witness.

Upgrade path: Obligation closure requires a checked carrier and finite theorems for witness preservation, classifier stability, residue non-escape, provenance, and NameCert export.

Chapter 554

A Concrete Naming Certificate for $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$

This chapter packages the AxisZeckendorf non-claim surface as a finite NameCert packet. Its carrier records checked negative theorem rows, horizon refusal rows, displayed transport, continuation routes, package provenance, and a local naming row; it does not add an arithmetic interpretation of the carry classifier.

Definition 554.1 (AxisZeckendorf cannot-claim registry packet)**.** *An* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet is a finite* BHist *tuple*

$$R = (a, b, c, d, e, f, g, h, p, n)$$

with the following visible coordinates:

1. *a is the negative theorem row for* `word_011` *and* `word_100`*, citing the non-*hsame *separation of Definition 9350.1;*
2. *b is the carry-not-*hsame *row, citing Definition 9350.2;*
3. *c is the non-normality row for* `word_011`*, citing Definition 9350.3;*
4. *d is the boundary-zero-spine exclusion row, citing Definition 9350.4;*
5. *e is the horizon refusal row for* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$*, citing Definition 9350.6;*
6. *f is the horizon refusal row for* $\mathsf{AxisNat}^{\uparrow}$ *as* $\mathsf{Nat}^{\uparrow}$*, citing Definition 9350.7;*
7. *g is the horizon refusal row for primitive dimension lift, citing Definition 9350.5;*
8. *h is the componentwise* hsame *and* Cont *route ledger by which consumers inspect exactly the rows a, b, c, d, e, f, g;*
9. *p is the* Pkg *provenance over* BHist*,* hsame*,* $\mathsf{ZCarry}^{\uparrow}$*,* $\mathsf{ZeckendorfNormal}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{FullAxis}^{\uparrow}$*,* $\mathsf{DimLift}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* NameCert*;*
10. *n is the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row.*

The packet has no coordinate for a Real completion, a natural-number replacement, a geometric dimension, a kernel rewrite, a quotient of histories, or a host arithmetic equality.

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.TasteGate.AxisZeckendorfCannotClaimTasteGate`
`single carrier alignment`

Theorem 554.2 (AxisZeckendorf cannot-claim public boundary)**.** *Every public read of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$ *factors through the displayed rows* $a, b, c, d, e, f, g, h, p, n$*. The packet cannot export RealUp completion data, NatUp replacement data, DimLift primitive data, quotient data, or kernel rewrite data.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket public boundary`

Proof. Project the displayed BHist packet in Definition 554.1. Its public inventory is exactly the seven refusal rows, the route ledger, the package row, and the local naming row.

The existing non-escape theorem for the registry states that public reads cannot leave that inventory. The carrier itself contains no row for RealUp completion, NatUp replacement, primitive dimension lift, quotient formation, or kernel rewriting.

Thus every public read is a read of $a, b, c, d, e, f, g, h, p, n$, and every excluded positive structure remains outside the exported surface. □

Theorem 554.3 (AxisZeckendorf cannot-claim refusal ledger row)**.** *Every accepted refusal read of an* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet is determined by exactly one of the seven refusal rows* a, b, c, d, e, f, g *before the route row* h*, provenance row* p*, or local naming row* n *is consumed.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket refusal ledger row`

Proof. The carrier lists the refusal rows first: a, b, c, d are theorem-level refusals and e, f, g are horizon refusals. These rows are the only coordinates that state refusal content.

The route row h can transport an already selected refusal row, and the rows p, n can record package provenance and local naming. None of h, p, n is itself a refusal source.

The existing refusal transport and non-escape statements keep this selection inside the displayed packet. Therefore an accepted refusal read has exactly one source among a, b, c, d, e, f, g before any route or provenance row is read. □

Theorem 554.4 (AxisZeckendorf cannot-claim downstream unblock)**.** *Downstream* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{FullAxis}^{\uparrow}$*,* $\mathsf{ZCarry}^{\uparrow}$*,* $\mathsf{ZeckendorfNormal}^{\uparrow}$*,* $\mathsf{DimLift}^{\uparrow}$*, and Real-horizon consumers of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *packet may cite only the finite refusal ledger* $a, b, c, d, e, f, g, h, p, n$*. They receive no positive AxisNat bridge, FullAxis bridge, carry-equality bridge, normal-form bridge, primitive dimension bridge, or RealUp bridge from this registry.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket downstream unblock`

Proof. Start with the public boundary. It confines every downstream read to the displayed finite registry rows and excludes the positive structures that would be needed for bridge data.

Next use the refusal ledger row. A downstream consumer first selects one refusal source among a, b, c, d, e, f, g and then may route or record that selected row through h, p, n.

The existing refusal transport theorem keeps the selected row as a refusal under componentwise hsame and Cont movement. Hence the downstream consumer can cite the cannot-claim ledger, but it cannot obtain a positive bridge for the listed AxisNat, FullAxis, carry, normal-form, dimension, or Real horizon surfaces. □

Theorem 554.5 (AxisZeckendorf Real-horizon refusal route)**.** *For an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$*, every downstream read that asks for* $\mathsf{FullAxis}^{\uparrow}$ *as a* $\mathsf{Real}^{\uparrow}$ *completion factors through the displayed refusal row* e*, then through the componentwise route row* h*, the provenance row* p*, and the local naming row* n*. The route supplies only the refusal recorded in Definition 9350.6; it does not provide a* $\mathsf{Real}^{\uparrow}$ *carrier, a completion operation, a quotient stream, a host real equality, or a selected limit.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket real refusal route`

Proof. Project the carrier definition. Row e is the only coordinate whose content is the $\mathsf{FullAxis}^{\uparrow}$-as-$\mathsf{Real}^{\uparrow}$ horizon refusal, and the carrier has no Real completion coordinate.

Apply the refusal ledger row. It forces the downstream Real-horizon request to select one displayed refusal source before any transport or package row is consumed; for this request the selected source is exactly e.

Transport non-promotion is supplied by the refusal transport theorem. The row h can move the selected refusal through componentwise hsame and Cont routes, but it cannot turn that refusal into a positive $\mathsf{Real}^{\uparrow}$ bridge.

Finally apply the downstream unblock and public boundary theorems. The read remains inside e, h, p, n, so it exports the stated refusal and no Real completion data, quotient stream, host real equality, or selected limit. □

Theorem 554.6 (AxisZeckendorf Nat-horizon refusal route)**.** *For an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$*, every downstream read that asks for* $\mathsf{AxisNat}^{\uparrow}$ *as a replacement for* $\mathsf{Nat}^{\uparrow}$ *factors through the displayed refusal row* f*, then through the componentwise route row* h*, the provenance row* p*, and the local naming row* n*. The route supplies only the refusal recorded in Definition 9350.7; it does not provide a natural-number replacement carrier, a recursion principle, a host arithmetic equality, or a positive Zeckendorf construction.*

Proof. Project the carrier definition. Row f is the only coordinate whose content is the $\mathsf{AxisNat}^{\uparrow}$-as-$\mathsf{Nat}^{\uparrow}$ horizon refusal, and no coordinate of R stores a replacement $\mathsf{Nat}^{\uparrow}$ carrier.

Use the refusal ledger row to isolate the source. A downstream natural-number request must first choose one of a, b, c, d, e, f, g as its refusal source, and the $\mathsf{AxisNat}^{\uparrow}$ replacement request chooses f.

The refusal transport theorem gives the only permitted movement of this row. Transport through h preserves the refusal as a refusal under hsame and Cont routing; it does not promote f to induction, recursion, or host arithmetic data.

The downstream unblock theorem and the public boundary then close the route. The consumer may cite f, h, p, n, but it receives no $\mathsf{Nat}^{\uparrow}$ replacement surface and no positive Zeckendorf construction. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket nat refusal route`

Theorem 554.7 (AxisZeckendorf dimension-lift refusal route)**.** *For an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$*, every downstream read that asks for primitive* $\mathsf{DimLift}^{\uparrow}$ *interpretation factors through the displayed refusal row* g*, then through the componentwise route row* h*, the provenance row* p*, and the local naming row* n*. The route supplies only the refusal recorded in Definition 9350.5; it does not provide a primitive geometric dimension, a host coordinate system, a quotient of histories, or a positive dimensional semantics.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket dimlift refusal route`

Proof. Project the carrier definition. Row g is the only coordinate whose content is the primitive dimension-lift horizon refusal, and the packet has no coordinate for a geometric dimension or host coordinate system.

Apply the refusal ledger row. A downstream dimension request is accepted only after selecting one displayed refusal source, and the primitive $\mathsf{DimLift}^{\uparrow}$ request selects g before route, provenance, or naming data are read.

Transport non-promotion follows from the refusal transport theorem. The componentwise row h can preserve and replay the refusal through hsame and Cont, but it cannot convert the refusal into primitive dimensional semantics.

Finally use downstream unblock together with the public boundary. The read remains confined to g, h, p, n, so no quotient of histories, primitive geometric dimension, host coordinate system, or positive dimensional semantics is exported. □

Theorem 554.8 (AxisZeckendorf cannot-claim registry obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *surface for the carrier of Definition 554.1 consists exactly of four finite obligation families:*

1. *negative theorem coverage for* `word_011` *versus* `word_100`*, carry-not-*hsame*,* `word_011` *non-normality, and boundary-zero-spine exclusion;*
2. *horizon refusal coverage for* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$ *as* $\mathsf{Nat}^{\uparrow}$*, and primitive dimension lift;*
3. *componentwise transport and continuation routing for every visible refusal row;*
4. *ledger exactness through the displayed* Pkg *provenance and local* $\mathsf{NameCert}$ *row.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket semantic name certificate`

Proof. The negative theorem rows are precisely a, b, c, d in the registry packet. Each row names an already stated AxisZeckendorf refusal and therefore supplies a finite citation target rather than a new classifier.

The horizon refusal rows are e, f, g. They expose only the three named non-claims: no $\mathsf{FullAxis}^{\uparrow}$ field or completion route is read as $\mathsf{Real}^{\uparrow}$, no $\mathsf{AxisNat}^{\uparrow}$ carrier is read as $\mathsf{Nat}^{\uparrow}$, and no $\mathsf{DimLift}^{\uparrow}$ seed is read as a primitive geometric dimension.

Transport and continuation routing are carried by h. Since the route ledger is componentwise, it can move a displayed refusal row to an equivalent registry presentation without turning a refusal into a positive bridge.

Finally, p and n record the package and local name boundary. Every public read of the registry passes through the seven refusal rows and these provenance coordinates, so the obligation surface is exhausted by the displayed packet. □

Theorem 554.9 (AxisZeckendorf source row coverage)**.** *For the concrete carrier* $R = (a, b, c, d, e, f, g, h, p, n)$ *of Definition 554.1, the source rows* a, b, c, d, e, f, g *exactly cover the checked refusal surface:* a, b, c, d *are the four negative theorem rows, and* e, f, g *are the three horizon refusal rows. No source row outside* a, b, c, d, e, f, g *is available in the carrier.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket source row coverage`

Proof. Read the carrier definition coordinate by coordinate. Rows a, b, c, d are respectively the `word_011`–`word_100` non-hsame row, the carry-not-hsame row, the `word_011` non-normality row, and the boundary-zero-spine exclusion row.

Rows e, f, g are respectively the $\mathsf{FullAxis}^{\uparrow}$-as-$\mathsf{Real}^{\uparrow}$ refusal, the $\mathsf{AxisNat}^{\uparrow}$-as-$\mathsf{Nat}^{\uparrow}$ refusal, and the primitive dimension-lift refusal. These are precisely the two coverage families named in Definition 554.8.

The remaining carrier coordinates are h, p, n. They provide route, package, and local naming data, not source refusal rows. Hence the seven displayed source rows exhaust the refusal surface of the concrete registry packet. □

Theorem 554.10 (AxisZeckendorf refusal transport)**.** *Let r be any public transport accepted by an* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet for one of the refusal rows a, b, c, d, e, f, g of $R = (a, b, c, d, e, f, g, h, p, n)$. Then r factors through the componentwise* hsame *relation and the displayed* Cont *route recorded in row h; it cannot pass through a parameter-equivalence transport external to the carrier.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket refusal transport`

Proof. By Definition 554.1, row h is the only transport and continuation coordinate of the packet. It is stated componentwise over exactly the visible refusal rows a, b, c, d, e, f, g.

Apply the transport family in Definition 554.8. That theorem identifies componentwise transport and continuation routing as one of the four finite obligation families of the local NameCert surface.

Since no other transport coordinate appears in R, an accepted public movement of a refusal row must use the hsame and Cont route displayed by h. The packet therefore supplies no external equivalence route that could turn a refusal row into a positive bridge. □

Theorem 554.11 (AxisZeckendorf ledger exactness)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the provenance rows p and n exhaust the public* Pkg *and* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *ledger. Consequently the packet cannot export RealUp operations, NatUp replacement, primitive dimension, quotient of histories, kernel rewrite, or standard-model bridge data.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket ledger exactness`

Proof. The carrier definition assigns p to the displayed Pkg provenance and n to the local $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannot}}$ row. There is no second package row, no second naming row, and no hidden coordinate in the tuple R.

The ledger exactness family in Definition 554.8 reads exactly these two provenance coordinates together with the refusal rows and the route row h. Thus every public ledger read has to pass through p or n after the visible refusal surface has been selected.

RealUp operations, NatUp replacement, primitive dimension, quotient formation, kernel rewriting, and standard-model bridge data would each require a positive coordinate outside this ledger. Because the carrier lists no such coordinate, the provenance rows are exact and cannot export any of those structures. □

Theorem 554.12 (AxisZeckendorf negative theorem surface)**.** *Every theorem-level refusal exported by an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *packet is one of the four negative rows a, b, c, d: the* `word_011`–`word_100` *non-*hsame *row, the carry-not-*hsame *row, the* `word_011` *non-normality row, or the boundary-zero-spine exclusion row.*

Proof. Inspect the carrier coordinates of Definition 554.1. The only theorem-level refusal rows are a, b, c, d.

The route ledger h transports these rows componentwise. It does not introduce a fifth negative theorem, because it has no theorem coordinate apart from the transported row it is carrying.

The package and local naming rows p, n record provenance only. Thus a consumer cannot extract a further theorem-level refusal from the registry beyond the four displayed rows. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket negative theorem surface`

Theorem 554.13 (AxisZeckendorf cannot-claim non-escape)**.** *Every public read exported by an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *packet factors through* $a, b, c, d, e, f, g, h, p, n$*. In particular, the registry cannot be used to construct an* $\mathsf{hsame}(\mathtt{word_011}, \mathtt{word_100})$ *witness, a RealUp structure on* $\mathsf{FullAxis}^{\uparrow}$*, an identification of* $\mathsf{AxisNat}^{\uparrow}$ *with* $\mathsf{Nat}^{\uparrow}$*, a primitive dimension for* $\mathsf{DimLift}^{\uparrow}$*, or a kernel rewrite rule.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket non escape`

Proof. Begin with the registry inventory. The packet contains exactly four negative theorem rows, three horizon refusal rows, the componentwise transport and continuation ledger, package provenance, and a local naming row.

The negative reads factor through a, b, c, d, and the horizon reads factor through e, f, g. The only admissible movement of those reads is the componentwise hsame and Cont routing recorded by h.

The package row p and local name row n do not add mathematical structure; they record where the displayed rows come from. Since none of the excluded objects appears as a coordinate, no public consumer can extract a positive bridge, equality, dimension, or rewrite from the registry packet. □

Theorem 554.14 (AxisZeckendorf cannot-claim consumer horizon exhaustion)**.** *Any downstream consumer of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet can read only the negative theorem rows* a, b, c, d*, the horizon refusal rows* e, f, g*, the transport and route ledger* h*, the provenance row* p*, and the local* $\mathsf{NameCert}$ *row* n*. Consequently the packet cannot supply a* $\mathsf{Real}^{\uparrow}$ *completion, a* $\mathsf{Nat}^{\uparrow}$ *replacement, a primitive dimension-lift operation, a kernel rewrite, a quotient of histories, or a standard-model bridge.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket consumer horizon exhaustion`

Proof. Start with the registry packet of Definition 554.1. Its displayed rows are exactly $a, b, c, d, e, f, g, h, p, n$.

The theorem-level consumer reads are exhausted by Definition 554.12: they are precisely the negative rows a, b, c, d.

The remaining horizon reads are the refusal rows e, f, g, transported and routed through h and recorded through p, n. By Definition 554.13, this public surface cannot export any coordinate beyond the registry inventory.

Since a $\mathsf{Real}^{\uparrow}$ completion, $\mathsf{Nat}^{\uparrow}$ replacement, primitive dimension-lift operation, kernel rewrite, quotient, or standard-model bridge would require a positive coordinate not present in the registry, no downstream consumer can derive one from the packet. □

Theorem 554.15 (AxisZeckendorf refusal row determinacy)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every accepted refusal read is determined by exactly one displayed source row among a, b, c, d, e, f, g. The rows a, b, c, d determine theorem-level refusals, the rows e, f, g determine horizon refusals, and no accepted refusal read is determined by h, p, or n.*

Proof. The packet definition displays the seven refusal coordinates before any route or provenance coordinate. By Definition 554.1, a, b, c, d are the negative theorem rows and e, f, g are the horizon refusal rows.

The obligation statement in Definition 554.8 separates these two refusal families from the transport and ledger families. Thus an accepted refusal read first selects one of the displayed source rows, not a route row or a provenance row.

The non-escape theorem Definition 554.13 and the consumer exhaustion theorem Definition 554.14 leave no public read outside $a, b, c, d, e, f, g, h, p, n$. Since h only routes and p, n only record provenance, the selected refusal source is unique among a, b, c, d, e, f, g. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket refusal row determinacy`

Theorem 554.16 (AxisZeckendorf transport non-promotion)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the componentwise* hsame *and* Cont *ledger h transports refusal rows only as refusal rows. It does not promote any transported row into an* hsame *witness, a* $\mathsf{Real}^{\uparrow}$ *completion, a* $\mathsf{Nat}^{\uparrow}$ *replacement, a primitive dimension, a quotient, or a kernel rewrite.*

Proof. The packet definition assigns the only transport and continuation coordinate to h. That coordinate is componentwise over the displayed refusal rows, so its source and target data are still rows drawn from a, b, c, d, e, f, g.

The routing family in Definition 554.8 records transport and continuation as an obligation on visible refusals. It does not add a positive equality, completion, replacement, dimension, quotient, or rewrite coordinate.

Apply the non-escape boundary of Definition 554.13. Any public read exported by the packet factors through the displayed registry rows, and none of the promoted structures appears in that inventory.

Finally, Definition 554.14 says downstream consumers can read the ledger h only together with the refusal and provenance rows. Therefore componentwise hsame and Cont transport preserves the refusal status instead of converting it into a positive BEDC structure. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket transport nonpromotion`

Theorem 554.17 (AxisZeckendorf provenance exhaustion)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the provenance row p and the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row n exhaust the public provenance of the registry. They add no new mathematical coordinate beyond the seven refusal rows and the route ledger h.*

Proof. The registry packet definition names p as the displayed Pkg provenance row and n as the local $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ row. No further provenance coordinate occurs in R.

The ledger exactness family in Definition 554.8 binds the public ledger to the displayed package provenance and local naming row. The provenance read is therefore exhausted by p and n after the refusal row and route row have been selected.

By Definition 554.13, the packet cannot export a positive bridge, equality, dimension, quotient, or rewrite outside the registry inventory. By Definition 554.14, a downstream consumer has no additional public coordinate to inspect.

Thus p and n close the public provenance surface of the registry and do not contribute another mathematical coordinate. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket provenance exhaustion`

Theorem 554.18 (AxisZeckendorf refusal kernel scope)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the seven refusal rows a, b, c, d, e, f, g are scoped over the displayed* BHist*,* hsame*, and* Cont *registry rows. Every accepted refusal read is determined inside that registry surface and cannot be promoted to a positive arithmetic, RealUp, NatUp, dimension, quotient, or kernel-rewrite claim.*

Proof. Read the carrier definition. The rows a, b, c, d are the negative theorem rows, the rows e, f, g are the horizon refusal rows, and the row h is the componentwise transport and continuation route ledger for those rows.

Apply Definition 554.15. It assigns each accepted refusal read to exactly one displayed source row among a, b, c, d, e, f, g before any route or provenance row is consumed.

Use Definition 554.16. The hsame and Cont ledger transports refusal rows only as refusal rows, so the registry cannot turn them into a positive bridge, equality, completion, replacement, dimension, quotient, or rewrite coordinate. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket refusal kernel scope`

Theorem 554.19 (AxisZeckendorf registry classifier scope)**.** *Classifier reads of an accepted* AxisZeckendorfCannotC *registry packet are determined by the displayed rows $a, b, c, d, e, f, g, h, p, n$ of Definition 554.1. They are not determined by host arithmetic, a RealUp interpretation of* $\mathsf{FullAxis}^{\uparrow}$*, a NatUp interpretation of* $\mathsf{AxisNat}^{\uparrow}$*, or a primitive interpretation of the dimension-lift row.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket registry classifier scope`

Proof. The carrier definition gives the classifier only the seven refusal rows, the route ledger, package provenance, and local naming row. No host arithmetic interpretation, RealUp completion, NatUp replacement, or primitive dimension coordinate is present.

Apply Definition 554.15. It shows that an accepted refusal read selects a source row from a, b, c, d, e, f, g, not an external arithmetic or semantic interpretation.

Then use Definition 554.16. Transport through h preserves refusal status under the displayed hsame and Cont rows and cannot turn the selected registry row into any of the excluded positive structures. □

Theorem 554.20 (AxisZeckendorf ledger route scope)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the* Pkg *row p and local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row n exhaust the consumer ledger route after the refusal source and componentwise route ledger have been selected. Thus every consumer access to the cannot-claim registry factors through the displayed refusal rows, h, p, and n.*

Proof. Start with the displayed registry packet. It lists the refusal source rows a, b, c, d, e, f, g, the componentwise route ledger h, the package row p, and the local naming row n.

Use Definition 554.15 to choose the refusal row inspected by the consumer. Use Definition 554.16 to route that row only through the displayed hsame and Cont ledger h.

Finally apply Definition 554.17. The rows p and n close the public provenance after the refusal and route rows have been selected, and no second package, naming, bridge, quotient, dimension, completion, replacement, or rewrite route is available. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket ledger route scope`

Theorem 554.21 (AxisZeckendorf negative row scope)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the four negative theorem rows a, b, c, d are scoped exactly over the displayed* BHist*,* hsame*, and* Cont *route surface. They are not promoted to positive equality, quotient, dimension, completion, replacement, or kernel-rewrite data.*

Proof. The packet definition displays a, b, c, d as the four theorem-level refusal rows before the horizon, route, provenance, and naming coordinates. The registry obligations assign their NameCert role to that visible refusal surface.

Apply refusal-row determinacy to separate a, b, c, d from the horizon rows and from h, p, n. Transport non-promotion then fixes the BHist, hsame, and Cont route surface as transport of refusals only, and non-escape prevents any promoted positive structure from entering the scope. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket negative row scope`

Theorem 554.22 (AxisZeckendorf horizon refusal scope)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the horizon refusal rows e, f, g scope exactly the refusal of* $\mathsf{Real}^{\uparrow}$ *completion,* $\mathsf{Nat}^{\uparrow}$ *replacement, and primitive dimension lift. They add no positive completion row, no Nat replacement carrier, and no primitive-dimension witness.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket horizon refusal scope`

Proof. The packet definition names e, f, g as the three horizon refusal rows. The registry-obligation and consumer-horizon-exhaustion theorems keep those rows in the public refusal inventory rather than in a positive construction inventory.

Refusal-row determinacy identifies each horizon read with one of e, f, g. Transport non-promotion and provenance exhaustion then show that h, p, n can only route and record these refusals, while non-escape rules out the corresponding positive $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and primitive-dimension objects. □

Theorem 554.23 (AxisZeckendorf FullAxis Real refusal boundary)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every consumer that reads the FullAxis-as-Real horizon row e receives only the displayed* BHist *refusal row e, its componentwise* hsame *and* Cont *route through h, the* Pkg *provenance row p, and the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row n. The read supplies no promoted* $\mathsf{Real}^{\uparrow}$ *completion for* $\mathsf{FullAxis}^{\uparrow}$ *and no hidden real-valued axis reconstruction.*

Proof. The carrier definition places e among the three horizon refusal rows, and Definition 554.22 identifies it with the refusal of a $\mathsf{Real}^{\uparrow}$ completion read for the full axis surface.

Apply refusal-row determinacy to lock the consumer read to the displayed row e. Apply transport non-promotion to the componentwise route h: the hsame and Cont data can only carry the same refusal row, not a positive completion object.

Finally use provenance exhaustion. The rows p and n record the package and local name after the refusal source has already been selected, and non-escape excludes every second route to a real-valued axis carrier. Thus the consumer receives the finite refusal package and no promoted $\mathsf{Real}^{\uparrow}$ completion. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket fullaxis real refusal boundary`

Theorem 554.24 (AxisZeckendorf AxisNat Nat refusal boundary)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every consumer that reads the AxisNat-as-Nat horizon row f receives only the displayed* BHist *refusal row f, its componentwise* hsame *and* Cont *route through h, the* Pkg *provenance row p, and the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row n. The read supplies no promoted* $\mathsf{Nat}^{\uparrow}$ *replacement for* $\mathsf{AxisNat}^{\uparrow}$ *and no hidden natural-number carrier.*

Proof. The carrier definition lists f as the AxisNat-as-Nat horizon refusal row, while Definition 554.22 fixes the row as a refusal rather than a positive $\mathsf{Nat}^{\uparrow}$ replacement.

Refusal-row determinacy assigns the accepted consumer read to f before any route, package, or naming coordinate is consumed. Transport non-promotion then keeps the h route inside the same refusal status; the displayed hsame and Cont data do not create a natural-number carrier.

Provenance exhaustion and non-escape close the boundary. The rows p, n only record the accepted cannot-claim package, and the registry has no second coordinate from which a $\mathsf{Nat}^{\uparrow}$ replacement for $\mathsf{AxisNat}^{\uparrow}$ can be read. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket axisnat nat refusal boundary`

Theorem 554.25 (AxisZeckendorf dimension-lift primitive refusal boundary)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every consumer that reads the primitive dimension-lift horizon row g receives only the displayed* BHist *refusal row g, its componentwise* hsame *and* Cont *route through h, the* Pkg *provenance row p, and the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row n. The read supplies no primitive dimension witness and no promoted dimension-lift carrier.*

Proof. The carrier definition displays g as the primitive dimension-lift refusal row, and Definition 554.22 keeps that row inside the horizon-refusal surface.

Use refusal-row determinacy to identify g as the unique source of the accepted read. The componentwise route h is governed by transport non-promotion, so its hsame and Cont data can replay the refusal but cannot construct a primitive dimension witness.

The package and naming rows are exhausted by p and n. Together with non-escape, this leaves the consumer with the finite refusal row and its recorded route, provenance, and name, and with no promoted dimension-lift object. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket dimlift primitive refusal boundary`

Theorem 554.26 (AxisZeckendorf audit-gate consumption)**.** *Any audit-gate consumer of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet*

$$R = (a, b, c, d, e, f, g, h, p, n)$$

must read refusal rows through a, b, c, d, e, f, g *and the route, provenance, and naming rows* h, p, n*. The consumer cannot turn the registry into* $\mathsf{Real}^{\uparrow}$ *completion data,* $\mathsf{Nat}^{\uparrow}$ *replacement data, primitive dimension data, a quotient, a kernel rewrite, or standard-model bridge data.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket audit gate consumption`

Proof. Begin with refusal-row determinacy. It assigns every accepted refusal read to one displayed source row among a, b, c, d, e, f, g before any route or provenance row is consumed.

Apply transport non-promotion to the audit route. The componentwise hsame and Cont ledger h transports refusal rows only as refusal rows, so it cannot promote the selected source into a positive completion, replacement, dimension, quotient, or rewrite coordinate.

Finally use provenance exhaustion together with negative-row and horizon-refusal scope. The rows p, n record the refusal route without adding mathematical data, and the scoped refusal theorems exclude the corresponding positive structures. Thus an audit-gate consumer receives only the displayed cannot-claim registry. □

Theorem 554.27 (AxisZeckendorf cannot-claim audit-gate boundary)**.** *Every audit consumer of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$ *factors through the seven refusal rows* a, b, c, d, e, f, g *together with the route, provenance, and local naming rows* h, p, n*. Such a consumer cannot read* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*, primitive dimension, quotient, kernel rewrite, or standard bridge data from the packet.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket audit gate boundary`

Proof. Start with audit-gate consumption. It confines every audit-facing read to the selected refusal source and to the displayed rows h, p, n.

Use non-escape and transport non-promotion for the boundary. The registry has no coordinate for RealUp completion, NatUp replacement, primitive dimension, quotient, kernel rewrite, or standard bridge data, and the route row h transports only refusal rows.

Finally apply provenance exhaustion and consumer horizon exhaustion. The rows p, n record package and local naming data without adding a mathematical coordinate, and every downstream audit read remains inside the displayed registry packet. □

Definition 554.28 (AxisZeckendorf cannot-claim route-provenance triad)**.** *After a selected refusal source* $x \in \{a, b, c, d, e, f, g\}$ *in the carrier* $R = (a, b, c, d, e, f, g, h, p, n)$*, the route-provenance triad is*

$$(h, p, n),$$

where h *is the componentwise* hsame *and* Cont *route for the selected refusal row,* p *is the* Pkg *provenance row for the cannot-claim registry, and* n *is the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row. The triad is read only after the refusal source has been chosen, and it is not itself a new refusal source.*

Theorem 554.29 (AxisZeckendorf route-provenance triad obligations)**.** *For the carrier $R = (a, b, c, d, e, f, g, h, p, n)$ and the triad (h, p, n) of Definition 554.28, the local obligation surface consists of carrier admission, transport through the displayed* hsame *and* Cont *route, ledger exactness through p, n, non-promotion of every selected refusal source, and exhaustion of all audit and downstream consumers by the selected source row together with the triad.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket route provenance triad obligations`

Proof. Carrier admission is the registry packet of Definition 554.1. The selected refusal source is one of a, b, c, d, e, f, g, and the triad supplies only the displayed route, provenance, and local naming rows.

Transport and non-promotion come from Definition 554.16. The row h may carry the selected refusal through hsame and Cont, but it cannot promote the refusal into a positive equality, completion, replacement, dimension, quotient, rewrite, or bridge.

Ledger exactness is supplied by Definition 554.17. The rows p, n exhaust the package and local naming surface, so the triad has no hidden ledger coordinate.

Finally apply Definition 554.14 and the audit-gate boundary of Definition 554.27. Every public consumer first chooses a refusal row and then reads only the triad, which proves consumer exhaustion without adding any forbidden positive structure. □

Theorem 554.30 (AxisZeckendorf negative refusal coverage)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the negative refusal coverage consumed by the* AxisZeckendorfCannotClai *obligation package is exactly the four rows a, b, c, d. These rows cover the non-*hsame *separation of* `word_011` *and* `word_100`*, the carry-not-*hsame *refusal, the* `word_011` *non-normality refusal, and the boundary-zero-spine refusal, and they do not promote to a positive equality, quotient, completion, dimension, replacement, or kernel-rewrite row.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket negative refusal coverage`

Proof. Start with the carrier definition. The packet lists a, b, c, d as the four negative theorem rows before the horizon, route, package, and local naming coordinates.

Use refusal-row determinacy and negative-row scope. Definition 554.15 selects one source row among a, b, c, d, e, f, g for every accepted refusal read, while Definition 554.21 identifies the theorem-level source family as exactly a, b, c, d.

Finally apply transport non-promotion and audit-gate consumption. The route row h may carry these refusals through hsame and Cont, but Definition 554.16 and Definition 554.26 prevent that route from becoming a positive BEDC object. □

Theorem 554.31 (AxisZeckendorf horizon refusal coverage)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the horizon refusal coverage consumed by the* AxisZeckendorfCannotClaim$^{\uparrow}$ *obligation package is exactly the three rows e, f, g. These rows cover the refusal of* FullAxis$^{\uparrow}$ *as* Real$^{\uparrow}$*, the refusal of* AxisNat$^{\uparrow}$ *as* Nat$^{\uparrow}$*, and the refusal of primitive dimension lift, with no hidden positive carrier behind the horizon.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket horizon refusal coverage`

Proof. Read the horizon coordinates from the carrier definition. Rows e, f, g are the only rows named for the three horizon-facing non-claims.

Apply horizon-refusal scope and refusal-row determinacy. Definition 554.22 fixes e, f, g as the $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and primitive-dimension refusal surface, and Definition 554.15 keeps each accepted horizon read on one displayed source row.

Use transport non-promotion, provenance exhaustion, and audit-gate consumption for the public route. The rows h, p, n can route and record the selected horizon refusal, but they cannot promote it to a completion, replacement, dimension, quotient, rewrite, or standard bridge. □

Theorem 554.32 (AxisZeckendorf ledger consumer exactness)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every public consumer of the cannot-claim ledger routes through h, p, n after selecting one refusal source among a, b, c, d, e, f, g. The row h supplies the componentwise* hsame *and* Cont *route, the row p supplies the displayed* Pkg *provenance, and the row n supplies the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *record.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket ledger consumer exactness`

Proof. Begin with the source choice. Refusal-row determinacy selects one row among a, b, c, d, e, f, g before any ledger coordinate is consumed.

Next route the selected row through h. Transport non-promotion says that the componentwise hsame and Cont ledger carries the same refusal status and cannot create a positive equality, completion, replacement, dimension, quotient, or rewrite.

Finally read the provenance and local name through p, n. Provenance exhaustion identifies these rows as the whole public package and naming surface, and audit-gate consumption confines every consumer of the registry to the selected refusal row together with h, p, n. □

Theorem 554.33 (AxisZeckendorf root-unblock package)**.** *The* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *root-unblock package is the finite* NameCert *obligation surface obtained by combining negative refusal coverage on a, b, c, d, horizon refusal coverage on e, f, g, and ledger consumer exactness through h, p, n. This surface is over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{ZCarry}^{\uparrow}$*,* $\mathsf{ZeckendorfNormal}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{FullAxis}^{\uparrow}$*,* $\mathsf{DimLift}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Nat}^{\uparrow}$*, and it exports no Real completion, Nat replacement, primitive dimension, quotient, kernel rewrite, or standard bridge.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket root unblock package`

Proof. First assemble the refusal sources. Definition 554.30 supplies the four negative rows a, b, c, d, and Definition 554.31 supplies the three horizon rows e, f, g.

Next attach the public ledger. Definition 554.32 routes every consumer through h, p, n, so the selected refusal row is carried only by displayed BHist, hsame, Cont, Pkg, and local NameCert evidence.

Finally use audit-gate consumption and the existing scope boundaries. Negative-row scope, horizon-refusal scope, transport non-promotion, and provenance exhaustion keep the package on the cannot-claim surface over $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{DimLift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ without adding any promoted BEDC object. □

Theorem 554.34 (AxisZeckendorf ledger route scope)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the rows h, p, n are the only public transport, package, and* NameCert *route for all consumers of the registry. Every public consumer must read refusal data through the componentwise* hsame *and* Cont *ledger h, the* Pkg *provenance row p, and the local naming row n.*

Proof. The carrier definition assigns route data to h, package provenance to p, and local naming to n. The registry obligations bind these rows to the same finite refusal inventory a, b, c, d, e, f, g.

Consumer-horizon exhaustion and provenance exhaustion leave no second transport, package, or naming coordinate for a public read. Together with non-escape and transport non-promotion, this forces every consumer route through h, p, n without adding a positive bridge, quotient, dimension, completion, replacement, or rewrite row. □

Theorem 554.35 (AxisZeckendorf refusal source lock)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, every accepted refusal reader first locks onto exactly one source row among a, b, c, d, e, f, g before it consumes the route ledger h, the provenance row p, or the local naming row n.*

Proof. Carrier projection gives the only refusal source inventory: the negative theorem rows a, b, c, d and the horizon refusal rows e, f, g. The route and provenance coordinates are listed only after that refusal source surface has been displayed.

Apply refusal-row determinacy. It assigns each accepted refusal read to one displayed source row among a, b, c, d, e, f, g, and it excludes h, p, n as refusal sources.

Transport non-promotion then fixes the role of h: it may route the selected refusal through componentwise hsame and Cont data, but it cannot choose a new source row or turn the selected source into a positive BEDC object. Provenance exhaustion fixes p, n as recording rows only, so the lock onto the unique source row precedes every route or provenance read. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket source lock`

Theorem 554.36 (AxisZeckendorf refusal reader totality)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$, the displayed refusal families a, b, c, d and e, f, g cover all public refusal reads exported by an accepted* AxisZeckendorfCannotClaim$^{\uparrow}$ *packet. No public refusal reader can be sourced by h, p, n or by data outside the carrier.*

Proof. The registry carrier exposes exactly two refusal families. Rows a, b, c, d are theorem-level refusals, and rows e, f, g are horizon refusals.

Negative theorem surface covers all theorem-level refusal reads, while consumer-horizon exhaustion covers the remaining horizon-facing reads. Together they leave only transport through h and recording through p, n after a refusal source has already been selected.

Refusal-row determinacy rules out h, p, n as source rows. Non-escape rules out every external source: the packet contains no RealUp completion, NatUp replacement, primitive dimension, quotient, host arithmetic, or kernel-rewrite coordinate from which another refusal reader could be drawn. □

Theorem 554.37 (AxisZeckendorf gap ledger exhaustion)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$, the rows h, p, n exhaust all GAP-facing route and provenance reads of the cannot-claim registry. These rows cannot promote any refusal source to* Real$^{\uparrow}$, Nat$^{\uparrow}$, *primitive dimension, quotient, or kernel-rewrite data.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket gap ledger exhaustion`

Proof. Ledger route scope identifies h, p, n as the only public transport, package, and local naming route after a refusal source has been selected. Provenance exhaustion shows that p and n add no mathematical coordinate beyond the seven refusal rows and the route ledger h.

Transport non-promotion gives the decisive boundary for the route row. The componentwise hsame and Cont ledger transports refusal rows only as refusal rows, so it cannot create a positive $\mathsf{Real}^{\uparrow}$ completion, $\mathsf{Nat}^{\uparrow}$ replacement, primitive dimension, quotient, or rewrite object.

Finally, consumer-horizon exhaustion and non-escape close the public route surface. Any GAP-facing consumer reads only the displayed refusal rows together with h, p, n, and none of those rows carries the promoted structures named in the statement. □

Theorem 554.38 (AxisZeckendorf cannot-claim dependency interface)**.** *Every admitted* AxisZeckendorfCannotClaim *packet records its dependency-facing rows for* $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{DimLift}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as visible* BHist, Cont, Pkg, *and* NameCert *data. The* hsame *route row preserves those dependency rows componentwise, and no consumer may read one of the seven dependencies through an unrecorded carrier, package, or naming witness.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket dependency interface`

Proof. Start with the carrier $R = (a, b, c, d, e, f, g, h, p, n)$. The negative theorem rows and horizon refusal rows list the dependency-facing surface before any public route is exposed, while the rows h, p, n give the componentwise hsame and Cont route, package provenance, and local NameCert records.

The registry obligations and source-row coverage identify the seven visible rows that a consumer may cite for $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{DimLift}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Refusal transport then keeps those rows inside the same BHist packet; it does not replace them by host arithmetic, a primitive dimension, a completion object, or a kernel rewrite.

Ledger exactness and gap ledger exhaustion close the interface. Since the source lock, refusal reader totality, and negative theorem surface have already exhausted the accepted reads, every dependency-facing consumer must pass through the displayed BHist, hsame, Cont, Pkg, and NameCert rows and cannot appeal to a second carrier. □

Theorem 554.39 (AxisNat and ZCarry refusal rows do not promote)**.** *In every admitted* AxisZeckendorfCannotClaim *packet, the* $\mathsf{AxisNat}^{\uparrow}$ *and* $\mathsf{ZCarry}^{\uparrow}$ *readbacks may feed the refusal ledger, but they cannot be converted into a* $\mathsf{FullAxis}^{\uparrow}$ *carrier row or a* $\mathsf{Real}^{\uparrow}$ *carrier row without an explicit* NameCert *witness for that promoted carrier. The available data remain the displayed* BHist *refusal rows, componentwise* hsame *transport,* Cont *route,* Pkg *provenance, and local cannot-claim* NameCert *row.*

Proof. Read the $\mathsf{AxisNat}^{\uparrow}$ and $\mathsf{ZCarry}^{\uparrow}$ rows through the dependency interface. Their role is refusal-facing: source-row coverage and the negative theorem surface place them among the rows that explain why the cannot-claim packet has no positive axis reconstruction.

Apply refusal transport and ledger exactness. The hsame and Cont route can replay the same refusal row inside the accepted BHist packet, and p, n can package and name that refusal. None of these operations introduces a $\mathsf{NameCert}_{\mathsf{FullAxis}^{\uparrow}}$ or $\mathsf{NameCert}_{\mathsf{Real}^{\uparrow}}$ witness.

Finally use the source lock, refusal reader totality, and gap ledger exhaustion. They force every public read through the displayed refusal inventory and exclude an external $\mathsf{FullAxis}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ carrier row. Thus AxisNat and ZCarry readbacks remain ledger data unless a separate promoted NameCert witness is supplied. □

Theorem 554.40 (Real readback non-promotion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an admitted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *packet receives only the recorded refusal and readback package. It reads the visible* BHist *rows, the componentwise* hsame *and* Cont *ledger, the* Pkg *provenance row, and the*

local NameCert *row; it does not receive a real-valued axis reconstruction or a hidden completion carrier.*

Proof. The carrier and registry obligations expose the $\mathsf{Real}^{\uparrow}$-facing entry only as part of the refusal surface. Source-row coverage and the negative theorem surface identify the row as a readback boundary, not as a real-valued axis carrier.

Refusal transport carries that boundary through h. The componentwise hsame and Cont data preserve the same refusal record, while p and n package and name the same cannot-claim packet.

Gap ledger exhaustion gives the final non-promotion step. A $\mathsf{Real}^{\uparrow}$ consumer can inspect the recorded refusal package, but the packet has no row for a $\mathsf{Real}^{\uparrow}$ completion, host real arithmetic, or reconstructed axis value. The source lock and refusal reader totality leave no second read path from which such a carrier could be obtained. □

Theorem 554.41 (AxisZeckendorf root-unblock refusal package)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$ of Definition 554.1, the rows a, b, c, d, e, f, g package the refusal source consumed by downstream unary-direction and axis-carry certificates. The rows h, p, n supply the only accepted route, package provenance, and local* NameCert *data for that downstream read.*

Proof. Read the carrier inventory. The negative refusal rows a, b, c, d are fixed by negative refusal coverage, and the horizon refusal rows e, f, g are fixed by horizon refusal coverage.

Use root-unblock package. It combines those two refusal families with ledger consumer exactness through h, p, n, so downstream certificates receive a finite refusal source rather than a positive arithmetic carrier.

Transport non-promotion and provenance exhaustion give the boundary. The route row h may replay the selected refusal, while p, n record provenance and local naming, but none of the three rows creates a promoted object for unary-direction or axis-carry consumers. □

Theorem 554.42 (AxisZeckendorf root-unblock downstream boundary)**.** *Any downstream consumer of the* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *root-unblock package may read only the refusal rows a, b, c, d, e, f, g together with the route, provenance, and local naming rows h, p, n. It may cite the package as a refusal source for unary-direction and axis-carry work, but it may not inspect host arithmetic, a hidden carrier, or any row outside the displayed registry.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket root unblock downstream boundary`

Proof. Begin with consumer-horizon exhaustion and refusal reader totality. Together they cover every accepted public refusal read by the displayed refusal rows a, b, c, d, e, f, g.

Next use ledger consumer exactness and gap ledger exhaustion. After a refusal source has been selected, the only public route and provenance rows are h, p, n.

The downstream restriction follows from transport non-promotion. Since h carries refusal rows only as refusal rows and p, n add no mathematical coordinate, unary-direction and axis-carry consumers cannot obtain a hidden carrier, host arithmetic interpretation, quotient, completion, replacement, dimension, or rewrite from the package. □

Theorem 554.43 (AxisZeckendorf downstream route refusal)**.** *Any* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *consumer of an accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *packet may read the cannot-claim source only through the seven refusal rows a, b, c, d, e, f, g and the route and provenance rows h, p, n already exposed by the registry. The consumer route of chapter 530 may cite these rows as a downstream refusal source, but it cannot read a positive direction bridge, host arithmetic carrier,* $\mathsf{Real}^{\uparrow}$ *completion,* $\mathsf{Nat}^{\uparrow}$ *replacement, primitive dimension, quotient, or kernel rewrite from the packet.*

Proof. Start with the consumer horizon exhaustion theorem. Definition 554.14 says that every downstream consumer is confined to the displayed refusal rows a, b, c, d, e, f, g together with the route, provenance, and local naming rows h, p, n.

Next use the root-unblock downstream boundary of Definition 554.42. It identifies the downstream package as a refusal source for later unary-direction and axis-carry work, not as a hidden positive carrier.

Finally apply ledger consumer exactness and transport non-promotion. The $\mathsf{UnaryDirectionBridge}^{\uparrow}$ consumer named by chapter 530 can replay the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, but no row in the registry promotes a refusal into the positive structures excluded in the statement. □

Theorem 554.44 (AxisZeckendorf root-unblock ledger non-escape)**.** *The downstream* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *root-unblock ledger cannot yield a positive* $\mathsf{Real}^{\uparrow}$ *row, a positive* $\mathsf{Nat}^{\uparrow}$ *row, a primitive* $\mathsf{DimLift}^{\uparrow}$ *row, a quotient row, or a kernel rewrite row. Every such requested read is routed back to the refusal source* a, b, c, d, e, f, g *and the ledger rows* h, p, n.

Proof. Start from the existing non-escape theorem. It excludes RealUp operations, NatUp replacement, primitive dimension, quotient of histories, kernel rewrite, and standard-model bridge data from the accepted registry packet.

Apply refusal-row determinacy. Any accepted downstream refusal read first selects one row among a, b, c, d, e, f, g, so the requested positive object cannot be sourced by h, p, n.

Then apply transport non-promotion. The componentwise hsame and Cont ledger h preserves the selected row as a refusal and cannot promote it to a positive RealUp, NatUp, DimLiftUp, quotient, or rewrite row.

Finally use provenance exhaustion. The rows p, n close the public package and naming route without adding another mathematical coordinate, so the downstream ledger has no escape path beyond the displayed refusal package. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket root unblock ledger nonescape`

Theorem 554.45 (AxisZeckendorf cannot-claim root obligation surface)**.** *For the displayed* BHist *registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$ *of Definition 554.1, the root obligation surface consists exactly of the four negative theorem rows* a, b, c, d*, the three horizon refusal rows* e, f, g*, the* hsame *transport and* Cont *route ledger* h*, the* Pkg *provenance row* p*, and the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ *row* n*. Each accepted root read must expose the refusal row it consumes, its transport through* h*, its route through* h*, its provenance in* p*, and its local naming in* n.

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket root obligation surface`

Proof. Begin with refusal coverage. Negative refusal coverage accounts for a, b, c, d, and horizon refusal coverage accounts for e, f, g.

Use the root-unblock package to combine those rows with the downstream route surface. It identifies the registry as a finite refusal source whose public reads are carried by the displayed BHist packet rather than by host arithmetic.

Finally apply ledger non-escape. The rows h, p, n transport, route, package, and name the refusal surface, but they add no positive $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DimLift}^{\uparrow}$, quotient, or kernel-rewrite coordinate. Hence the root obligation surface is exactly the displayed registry packet. □

Theorem 554.46 (AxisZeckendorf cannot-claim downstream blocker scope)**.** *Every downstream blocker that cites* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *must factor through the displayed registry packet* $R = (a, b, c, d, e, f, g, h, p, n)$*. It may consume a NameCert refusal row, transport it by the componentwise* hsame *data in* h*, follow the displayed* Cont *route in* h*, and cite the* Pkg *provenance* p *and local name* n*; it may not replace those rows by a positive arithmetic construction, a hidden carrier, a host equality, or a stronger axis reconstruction.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket downstream blocker scope`

Proof. First use consumer horizon exhaustion. It says that every downstream consumer is confined to the refusal rows a, b, c, d, e, f, g together with the route, provenance, and local naming rows.

Next use the root-unblock downstream boundary. That theorem identifies the consumer-facing role of the packet as a refusal source for unary-direction and axis-carry work, not as a promoted arithmetic carrier.

The ledger non-escape theorem supplies the blocker scope. Since the registry cannot yield a positive $\mathsf{Real}^{\uparrow}$ row, $\mathsf{Nat}^{\uparrow}$ row, primitive dimension row, quotient row, or kernel rewrite row, every downstream blocker remains a read of the displayed NameCert refusal rows and their h, p, n support. □

Theorem 554.47 (AxisZeckendorf cannot-claim formalization handoff)**.** *The future formalization target for* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *is the finite registry handoff for* $R = (a, b, c, d, e, f, g, h, p, n)$*: refusal coverage for* a, b, c, d, e, f, g*,* hsame *transport and* Cont *route preservation through* h*,* Pkg *provenance through* p*, local* $\mathsf{NameCert}$ *naming through* n*, and non-escape from all requested positive* $\mathsf{FullAxis}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{DimLift}^{\uparrow}$*, quotient, and kernel-rewrite rows. The handoff is a negative registry theorem, not a positive Zeckendorf or axis construction.*

Proof. Assemble the proof surface from the existing registry theorems. Refusal coverage supplies the seven negative rows, and refusal reader totality says that accepted reads choose from those rows.

Transport and route preservation are supplied by ledger consumer exactness and transport non-promotion. They keep the selected row inside the same BHist, hsame, and Cont registry route, while p, n record provenance and local naming.

The final formalization boundary is given by consumer horizon exhaustion and root-unblock ledger non-escape. Together they rule out every listed positive structure and leave the future Lean handoff as the finite cannot-claim registry rather than an arithmetic construction. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimUp StdBridge`

Theorem 554.48 (AxisZeckendorf seven-row refusal coverage)**.** *For the concrete carrier* $R = (a, b, c, d, e, f, g, h, p, n)$ *of Definition 554.1, the rows* a, b, c, d, e, f, g *exhaust the source refusal surface of* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$*. The rows* a, b, c, d *cover the negative theorem refusals, the rows* e, f, g *cover the horizon refusals, and no accepted source refusal is supplied by* h, p, n *or by any hidden arithmetic, completion, dimension, quotient, or rewrite coordinate.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket seven row coverage`

Proof. Use the carrier definition to list the source rows before the ledger rows. It exposes exactly the four negative theorem rows a, b, c, d and the three horizon rows e, f, g.

Apply negative refusal coverage and horizon refusal coverage. These identify the theorem-level and horizon-level refusal families and leave no additional source slot in the displayed packet.

Finally apply refusal-row determinacy and non-escape. Every accepted refusal read selects one of a, b, c, d, e, f, g, while the route and provenance rows h, p, n cannot become source refusals or positive BEDC objects. □

Theorem 554.49 (AxisZeckendorf refusal transport route exhaustion)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$, the row h is the only admissible* hsame *and* Cont *transport route for the refusal rows a, b, c, d, e, f, g. Every downstream consumer that transports a refusal row must first choose one of those seven rows and then route it through h, with provenance and local naming recorded only by p and n.*

Proof. Start from ledger consumer exactness. It says that every public consumer reads the selected refusal source through the route ledger h and then records package and naming data through p, n.

Refusal-row determinacy supplies the selected source row among a, b, c, d, e, f, g. Transport non-promotion then fixes the role of h: it carries the chosen refusal only as a refusal, not as a positive carrier.

Since the carrier definition contains no second hsame or Cont coordinate, the transport surface is exhausted by h. The rows p, n record the result after transport and do not supply another route. □

Theorem 554.50 (AxisZeckendorf positive bridge exclusion)**.** *For the concrete carrier $R = (a, b, c, d, e, f, g, h, p, n)$, none of the rows p, n, h can promote a refusal row into* $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DimLift}^{\uparrow}$*, quotient data, kernel rewrite data, or standard-bridge data. They record provenance, local naming, and componentwise refusal transport for the seven refusal rows only.*

Proof. Read the three ledger rows separately. The row h is the route ledger, while p and n record the public package and the local $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$ name.

Apply transport non-promotion to h. A transported source from a, b, c, d, e, f, g remains a refusal row and cannot become a positive bridge, completion, replacement, dimension, quotient, or rewrite object.

Apply provenance exhaustion and non-escape to p, n. These rows add no mathematical coordinate beyond the refusal package, so no standard-bridge or positive BEDC data can be read from them. □

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket positive bridge exclusion`

Closure status for $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ packet records four checked negative theorem rows, three horizon refusal rows, componentwise hsame and Cont routing, Pkg provenance, and a local NameCert row, while exporting no RealUp, NatUp replacement, dimension, quotient, or kernel-rewrite coordinate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$, maturePackageClosed)

Scope closed: The AxisZeckendorf cannot-claim registry packet is mature on the paper closure axis: its semantic-certificate readback records four negative theorem rows, three horizon refusal rows, componentwise transport, ledger exactness, non-escape, consumer exhaustion, determinacy, transport non-promotion, provenance exhaustion, negative-row scope, horizon-refusal scope, and ledger-route scope, while Lean verification and StdBridge remain separate formal-axis work.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.AxisZeckendorfCannotClaimUp.AxisZeckendorfCannotClaimRegistryPacket\_semantic\_name\_ce`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not prove a Lean implementation, a standard bridge, positive RealUp data for FullAxisUp, AxisNatUp equality with NatUp, primitive dimension lift, quotient construction, host arithmetic equality, or kernel rewriting.

Upgrade path: no further upgrade – chapter at top of the formal axis.

Chapter 555

A Concrete Naming Certificate for $\mathsf{InductionClosureLedger}^{\uparrow}$

The $\mathsf{InductionClosureLedger}^{\uparrow}$ surface realizes generator-internal induction as a finite naming certificate. It records the authorized generator rows, the generator-closure classifier rows, the Gap residue left at every attempted free-floating induction claim, and the transport, continuation, provenance, and naming rows that make the ledger auditable.

The ledger consumes the authorized-generator row from chapter 525 and the generator-closure classifier row from chapter 486 before the Gap residue row is exported. Requests that fail to stay inside those displayed generator closures are recorded as Gap residue rather than as free-standing induction authority.

Definition 555.1 (InductionClosureLedger carrier)**.** *An* $\mathsf{InductionClosureLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, K, R, T, C, P, N)$$

with the following displayed rows:

1. *G is the authorized-generator row, listing the generator names whose closed constructor lists license internal induction;*
2. *K is the generator-closure classifier row, recording the finite classifier that decides whether a requested induction step stays inside the closure of a listed generator;*
3. *R is the* Gap *ledger residue row, recording every refused passage from generator-internal induction to free-floating induction over unspecified objects;*
4. *T records componentwise* hsame *transport for G, K, R, the continuation row, the provenance row, and the local naming row;*
5. *C records the displayed* Cont *routes that replay generator admission, classifier lookup, residue emission, and NameCert export;*
6. *P is the* Pkg *provenance over* BHist*, authorized generator rows, generator-closure classifier rows,* Gap *ledger residue,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no universal induction principle over unnamed objects, no hidden eliminator outside a displayed generator closure, and no route that erases the Gap *residue attached to an out-of-closure request.*

Theorem 555.2 (InductionClosureLedger NameCert obligations)**.** *For the carrier of Definition 555.1, the local* $\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (G, K, R, T, C, P, N),$$

authorized-generator exposure, generator-closure classifier lookup, Gap *residue retention, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and NameCert export for the same finite packet.*

Lean4 ✓ `BEDC.Derived.InductionClosureLedgerUp.InductionClosureLedgerTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 555.1. The packet exposes only the seven displayed rows G, K, R, T, C, P, N.

Generator authority is read from G, closure classification from K, and refused out-of-closure content from R. These three rows determine which induction requests are internal to an authorized generator and which requests leave a recorded residue.

Transport, replay, and provenance are confined by T, C, P. The local row N exports only this finite packet as a naming certificate, so no NameCert projection can add a hidden universal induction principle. □

Theorem 555.3 (InductionClosureLedger generator authority)**.** *Every induction route accepted by an* $\mathsf{InductionClosureLedger}^{\uparrow}$ *packet factors through an authorized generator row in* G *and a generator-closure classifier row in* K*. The route may replay the displayed* Cont *evidence for that generator closure, transported by* hsame *and recorded under* Pkg *provenance and* $\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}$ *naming, but it cannot claim authority from an unnamed generator.*

Proof. Project the authorized-generator row G and classifier row K. A replayable induction route is accepted only when the classifier records that the requested step stays inside the closure of a generator listed in G.

Use T to transport the listed generator and classifier rows componentwise. Use C to replay the accepted route, and record the read through P and N.

An unnamed generator has no coordinate in L. Since the carrier has no row from which such authority could be projected, every accepted route is exhausted by the displayed generator and classifier rows. □

Theorem 555.4 (InductionClosureLedger GAP non-escape)**.** *Whenever an induction request is not classified by* K *as lying inside the closure of an authorized generator in* G*, the* $\mathsf{InductionClosureLedger}^{\uparrow}$ *packet records the request only as* Gap *residue in* R*. The residue may be transported, replayed as a refusal, packaged, and named, but it cannot be converted into an internal induction theorem.*

Proof. Inspect the classifier row K. If the request is not accepted by the displayed generator-closure classifier, the only carrier coordinate that records it is the residue row R.

Transport of residue through T preserves the residue as residue. Replay through C can expose the refusal and package it through P, while N names the same boundary.

No conversion row is present in the carrier. Therefore a recorded Gap residue cannot escape into an internal induction theorem or a free-floating closure principle. □

Theorem 555.5 (InductionClosureLedger transcendental-input sibling link)**.** *Every free-floating induction request rejected into the residue row* R *of an accepted* $\mathsf{InductionClosureLedger}^{\uparrow}$ *packet is*

classified as a supply-socket kind by the taxonomy surface of chapter 523 before it can be cited downstream. The sibling classifier records why the residue remains outside substrate closure rather than becoming a free-standing induction authority.

Proof. Project the ledger carrier $L = (G, K, R, T, C, P, N)$ from Definition 555.1. The generator row G and classifier row K decide whether the requested induction step stays inside an authorized generator closure.

Apply generator authority from Definition 555.3. Any accepted induction route must be licensed by the displayed generator and closure classifier rows.

For a rejected request, apply GAP non-escape from Definition 555.4. The request is retained only as residue in R and cannot be converted into an internal induction theorem.

Now cite the sibling taxonomy carrier and boundary exhaustion of chapter 523. The free-floating induction socket is one of the displayed supply-socket kinds, and the taxonomy surface records the requested supply shape, Gap tag, audit gate, site row, transport, continuation, provenance, and local name before any downstream citation.

Thus the ledger residue is downstream-readable only after the sibling classifier has recorded why the induction request remains outside substrate closure. □

Theorem 555.6 (InductionClosureLedger NameCert export)**.** *The public export of an* $\mathsf{InductionClosureLedger}^{\uparrow}$ *packet is exactly the finite NameCert packet over authorized generator rows, generator-closure classifier rows,* Gap *ledger residue,* hsame *transport,* Cont *replay, and* Pkg *provenance. A consumer of the export can recover generator authority, classifier status, and residue location, but no object beyond those rows.*

Proof. Start with the carrier tuple $L = (G, K, R, T, C, P, N)$. The export row N names this tuple and no larger surface.

A consumer recovers generator authority from G, classifier status from K, and residue location from R. The same consumer uses T for componentwise transport, C for replay, and P for provenance.

These rows exhaust the public export. Since L has no coordinate for an unnamed induction principle, hidden eliminator, or erased residue, the NameCert packet cannot export such content. □

Closure status for $\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{InductionClosureLedger}^{\uparrow}$ packet is generated from authorized generator rows, generator-closure classifier rows, Gap ledger residue, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 555.1, Definition 555.2, Definition 555.3, Definition 555.4, and Definition 555.6 bind the finite carrier, generator authority, closure classifier, residue boundary, and NameCert export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InductionClosureLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, a TasteGate-backed carrier, universal induction over unnamed objects, erased GAP residue, or a public standard bridge.

Upgrade path: Promotion beyond seedClosure requires a TasteGate-backed Lean carrier and checked finite theorems for generator authority, classifier stability, residue non-escape, provenance, and NameCert export.

Chapter 556

A Concrete Naming Certificate for $\mathsf{GeneratorFixedPoint}^{\uparrow}$

The $\mathsf{GeneratorFixedPoint}^{\uparrow}$ packet names the finite certificate surface where an authorized generator returns a displayed output that remains fixed under its own certified continuation boundary. It refines the $\mathsf{GeneratorClosure}^{\uparrow}$ classifier of chapter 486: the constructor-list provenance and classifier row are consumed from that sibling before the fixed-output witness is read. A continuation without that witness is not a fixed point of the generator; it is recorded as a GAP boundary.

Definition 556.1 (Generator fixed-point carrier)**.** *A* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, L, Q, W, O, H, C, P, N)$$

with the following displayed rows:

1. *G is the authorized generator row whose constructor-list closure supplies the only admitted source of generated outputs;*
2. *L is the constructor-list provenance for G, including the displayed witness that the relevant continuation remains inside that list;*
3. *Q is the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row that compares generator-internal continuations with boundary attempts;*
4. *W is the fixed-output witness, recording that the displayed generator continuation returns the same output row under the certified route;*
5. *O is the displayed output row whose fixed-output status is witnessed by W;*
6. *H records componentwise* hsame *transports for G, L, Q, W, O, P, N and the local packet shape;*
7. *C records the finite* Cont *routes by which consumers read the generator row, constructor-list provenance, classifier row, fixed-output witness, output row, package provenance, and local naming row;*
8. *P is the* Pkg *provenance over* BHist*, authorized generator rows, constructor-list provenance,* $\mathsf{GeneratorClosure}^{\uparrow}$*, fixed-output witnesses, displayed outputs,* hsame*,* Cont*, and boundary ledger rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row.*

The carrier contains no host-level fixed-point theorem, no self-application oracle, and no induction rule beyond the constructor-list provenance displayed in L.

Theorem 556.2 (Generator fixed-point NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *surface for Definition 556.1 consists of five finite obligation rows:*

1. *carrier admission for the packet* $F = (G, L, Q, W, O, H, C, P, N)$*;*
2. *classifier stability stating that the* $\mathsf{GeneratorClosure}^{\uparrow}$ *row* Q *is preserved by componentwise* hsame *transport of the displayed generator, constructor-list provenance, fixed witness, output row, routes, package provenance, and local certificate;*
3. *fixed-witness persistence stating that* W *remains attached to* O *under the displayed generator-internal continuation routes;*
4. *non-escape stating that any continuation missing the constructor-list witness required by* L *is recorded as a* GAP *boundary rather than as a generator fixed point;*
5. *ledger exactness stating that every public read of the certificate factors through* $G, L, Q, W, O, H, C, P, N$ *and no unrecorded fixed-point principle is exported.*

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 556.1. The tuple lists the generator row, constructor-list provenance, closure classifier, fixed-output witness, displayed output, transports, routes, package provenance, and local naming row.

Classifier stability is read through H and C. Transport preserves the generator row and constructor-list provenance, so the $\mathsf{GeneratorClosure}^{\uparrow}$ row Q continues to classify the same generator-internal boundary.

Fixed-witness persistence is the displayed role of W. Since W is a coordinate of the same finite packet as O, a consumer route that stays inside C can read the output only together with the witness that certifies it as fixed under the displayed continuation.

The non-escape and ledger-exactness rows are the remaining package clauses. If a continuation lacks the constructor-list witness named by L, no carrier coordinate can turn it into a fixed point; the package records it as GAP boundary evidence, and every exported read remains a projection of the listed rows. □

Theorem 556.3 (Generator fixed-point non-escape)**.** *Let* F *be an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *carrier. A finite continuation read that lacks the constructor-list witness required by* L *is recorded by the package provenance* P *as a* GAP *boundary. It is not admitted as a fixed point of the generator row* G*, and it cannot export the fixed-output witness* W *for the displayed output row* O*.*

Proof. Read the continuation through the displayed route C. If the route carries the constructor-list witness from L, then it is evaluated against the $\mathsf{GeneratorClosure}^{\uparrow}$ classifier Q and may read the fixed-output witness W for O.

If the route lacks the witness from L, it has no carrier coordinate that authorizes it as generator-internal. The package provenance P therefore records the attempted read as GAP boundary evidence, and the local naming row N cannot expose it as a fixed point. □

Theorem 556.4 (Generator fixed-point witness stability)**.** *Componentwise* hsame *transport preserving* G, L, Q, W, H, C, P, N *preserves the fixed-output witness for the displayed row* O*. After such transport, the transported packet still reads* O *through the same authorized generator row, constructor-list provenance,* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier, fixed-output witness, continuation routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row.*

Proof. Use the transport row H. It preserves the generator row G, the constructor-list provenance L, the classifier Q, the fixed-output witness W, and the local routing and package rows.

The continuation row C then replays the same generator-internal read of the displayed output. Since the witness W is transported as one of the preserved coordinates, the fixed-output status of O remains attached to the transported packet.

Finally P and N repack the transported rows as the same local certificate surface. No new continuation route is introduced, so a boundary attempt remains outside the fixed-point witness. □

Theorem 556.5 (Generator fixed-point source admission row)**.** *For an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet*

$$F = (G, L, Q, W, O, H, C, P, N),$$

source admission is exactly the displayed pairing of the authorized generator row G with the constructor-list provenance L. A continuation route is admitted as generator-internal only when it reads through G and carries the witness from L; otherwise the local packet records the route as boundary evidence and does not expose it as a fixed point.

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointPacket source admission row`

Proof. Project the carrier of Definition 556.1. The only source rows are G and L, so admission cannot be supplied by an unlisted generator or by a continuation outside the constructor-list provenance.

Use Definition 556.3. A route without the displayed witness from L is a GAP boundary, not a fixed-output read. Thus the source admission row is precisely the displayed G, L pairing. □

Theorem 556.6 (Generator fixed-point transport row)**.** *Let $F = (G, L, Q, W, O, H, C, P, N)$ be an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet. Componentwise* hsame *transport preserving the displayed generator row, constructor-list provenance, closure classifier, fixed-output witness, displayed output, continuation routes, package provenance, and local naming row preserves the fixed-output row under* Cont *transport. The transported packet still classifies the same generator-internal route by Q and keeps W attached to O.*

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointPacket transport row`

Proof. Use the transport row H to move the displayed generator, constructor list, classifier, witness, output, route, package, and naming coordinates together.

Then apply Definition 556.4. The transported witness remains attached to the transported output row because W and O are coordinates of the same finite packet.

Finally use the continuation route row C. It replays the same generator-internal read after transport, so the classifier row Q still marks the route as internal and does not create a new boundary read. □

Theorem 556.7 (Generator fixed-point public obligation package)**.** *For every accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet, carrier admission, classifier stability, fixed-witness persistence, non-escape, and ledger exactness jointly form the public obligation package for $F = (G, L, Q, W, O, H, C, P, N)$. Every public read factors through the source admission row of Definition 556.5, the transport row of Definition 556.6, the fixed-witness row of Definition 556.4, and the finite ledger rows C, P, N.*

Proof. First collect carrier admission from Definition 556.1. The carrier inventory fixes the generator, provenance, classifier, witness, output, transport, route, package, and naming rows.

Next apply the source admission and transport rows. These identify the admitted generator-internal route and show that componentwise hsame and Cont transport preserve the same classifier and fixed-output witness.

Use Definition 556.3 for boundary attempts. A route without the constructor-list witness is recorded as GAP evidence and cannot export W.

Finally the ledger rows C, P, N make the public package exact. Every exported read is a projection of $G, L, Q, W, O, H, C, P, N$, and no host fixed-point principle, self-application oracle, or unlisted continuation coordinate is available. □

Theorem 556.8 (Generator fixed-point admitted packet row)**.** *Every accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet exposes exactly the rows*

$$G, L, Q, W, O, H, C, P, N$$

of Definition 556.1. A public or local read of the packet first projects one of these displayed rows: the authorized generator row G, constructor-list provenance L, $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row Q, fixed-output witness W, displayed output row O, componentwise* hsame *transport row H,* Cont *route row C,* Pkg *provenance row P, or local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row N. No accepted packet contains an additional host fixed-point principle, self-application oracle, or unlisted continuation coordinate.*

Proof. Start from the carrier definition. The accepted packet is the finite BHist tuple displayed there, so admission is projection from the nine named coordinates and not from an ambient fixed-point operator.

The route row C and provenance row P determine how consumers read the tuple. Since every such route is attached to one of the displayed coordinates, any accepted read factors through $G, L, Q, W, O, H, C, P, N$.

The last assertion is the non-escape side of the same inventory. The carrier definition lists no host theorem, oracle, or extra continuation field, and the local naming row N certifies only the displayed packet shape. □

Theorem 556.9 (Generator fixed-point classifier transport row)**.** *Let $F = (G, L, Q, W, O, H, C, P, N)$ and $F' = (G', L', Q', W', O', H', C', P', N')$ be accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packets related by componentwise* hsame *transport of the displayed generator row, constructor-list provenance, classifier row, fixed-output witness, output row, routes, package provenance, and local naming row. Then the transported packet preserves the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row and the fixed-output witness route: Q' classifies the transported generator-internal continuation under the transported constructor-list provenance, and W' remains attached to the transported output row O' through the transported* Cont *routes.*

Proof. Use the transport row H of F together with the matching row H' of F'. These rows relate the generator, constructor-list provenance, classifier, witness, output, routes, provenance, and naming coordinates component by component.

The classifier part is the transported Q row. Because the constructor-list provenance L is transported with the generator row G, the $\mathsf{GeneratorClosure}^{\uparrow}$ classifier reads the same generator-internal boundary after transport.

The fixed-output part is the transported W row. Since the output row O and continuation routes C are transported in the same packet, the witness route remains attached to O' and does not create a separate boundary read. □

Theorem 556.10 (Generator fixed-point witness ledger row)**.** *For every accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet, any public witness read factors through the fixed-output witness row* W*, displayed output row* O*, continuation route row* C*, package provenance row* P*, and local naming row* N*. A read that does not factor through this witness ledger remains a* GAP *boundary; in particular, non-witness generator routes and continuations outside the constructor-list provenance are not exported as fixed-output witnesses.*

Proof. Read a public witness request through the route row C. If the route is a witness route, it must project the displayed witness W together with the displayed output row O.

The package provenance P and local naming row N then record that projection as part of the finite certificate surface. Thus a witness consumer receives the same W, O, C, P, N ledger, not an unrecorded fixed-point principle.

If the route is not one of the displayed witness routes, Definition 556.3 applies. The attempt lacks the required constructor-list witness and is recorded as a GAP boundary rather than as a fixed-output export. □

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointPacket witness ledger row`

Theorem 556.11 (Generator fixed-point TasteGate witness route)**.** *Every TasteGate-admissible witness read of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet factors through the authorized generator row* G*, constructor-list provenance* L*,* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row* Q*, fixed-output witness* W*, displayed output row* O*, componentwise* hsame *transport row* H*,* Cont *route row* C*,* Pkg *provenance row* P*, local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row* N*, and the AI-roadmap TasteGate discipline that permits only displayed witness routes.*

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointTasteGate single carrier alignment`

Proof. Start with the admitted packet row. It exposes only $G, L, Q, W, O, H, C, P, N$, so a TasteGate witness request cannot begin from a host fixed-point theorem or a self-application oracle.

Use the classifier transport row to keep the $\mathsf{GeneratorClosure}^{\uparrow}$ classifier attached to the transported authorized generator and constructor-list provenance. The same transport keeps the fixed-output witness attached to the displayed output and to the listed continuation routes.

Finally apply the witness ledger row. TasteGate-admissible reads are exactly the reads that pass through W, O, C, P, N after the generator and classifier rows have been displayed, so the route is a finite certificate route rather than an ambient generation principle. □

Theorem 556.12 (Generator fixed-point diagonal obstruction scope)**.** *For an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet, any diagonal or self-application attempt outside the authorized generator row, constructor-list provenance,* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier, fixed-output witness,* hsame *transport,* Cont *route,* Pkg *provenance, local* NameCert *row, and TasteGate witness discipline is a* GAP *boundary. It is not a fixed-output witness and cannot be read as a generator fixed point.*

Proof. Read the attempted diagonal route through the carrier. If it lacks the displayed constructor-list provenance or classifier row, the source admission row and non-escape theorem already classify the attempt as boundary evidence.

If the attempt carries those rows but lacks the fixed-output witness route, the witness ledger row applies. It permits public witness reads only through W, O, C, P, N, so the attempted diagonal has no exported witness coordinate.

The TasteGate witness route then supplies the scope boundary. It admits displayed witness routes and excludes self-application routes that are not already carried by the finite packet. □

Theorem 556.13 (Generator fixed-point audit non-escape ledger)**.** *The public audit ledger for an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet is exhausted by the authorized generator row,* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row, fixed-output witness,* hsame *transport,* Cont *routes,* Pkg *provenance, local* NameCert *row, and the TasteGate witness-route discipline. The ledger exports no unlisted continuation, host fixed-point principle, self-application oracle, or semantic completeness claim.*

Proof. First combine the public obligation package with the admitted packet row. Together they make every public read a projection of $G, L, Q, W, O, H, C, P, N$.

Next apply the TasteGate witness route. It identifies the accepted witness route as the displayed generator, classifier, fixed-output, transport, continuation, provenance, and naming surface.

The diagonal obstruction scope handles the remaining cases. Any read outside that surface is a GAP boundary, so the audit ledger has no coordinate from which to export an unlisted continuation, a host fixed-point principle, a self-application oracle, or semantic completeness. □

Theorem 556.14 (Generator fixed-point TasteGate boundary exhaustion)**.** *Every accepted public fixed-point witness read of a* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet factors through the finite rows*

$$G, L, Q, W, O, H, C, P, N$$

and through the checked TasteGate carrier-alignment row cited at Definition 556.11. A route that attempts diagonal self-application, a non-witness generator read, a classifier-free fixed-output read, or a continuation outside the constructor-list provenance remains a GAP *boundary rather than a public fixed-point witness.*

Proof. First apply the source admission row. It admits a generator-internal read only through G together with the constructor-list provenance L.

Next transport the classifier and witness. The classifier transport row keeps Q attached to G, L, while the fixed-witness and witness-ledger rows keep W attached to O through the displayed continuation and package rows C, P, N.

Now use the checked TasteGate carrier-alignment row of Definition 556.11. It restricts AI-origin witness reads to the same finite surface $G, L, Q, W, O, H, C, P, N$.

Finally apply the audit non-escape ledger. Any non-witness diagonal route, self-application attempt, classifier-free route, or continuation outside the constructor-list provenance has no coordinate in the packet and is recorded as GAP boundary evidence. □

Theorem 556.15 (Generator fixed-point obligation-closure readiness)**.** *For an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet*

$$F = (G, L, Q, W, O, H, C, P, N),$$

carrier admission, the $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row* Q*, the fixed-output witness* W*, the* GAP *boundary row for non-admitted continuations, componentwise* hsame *transport* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row* N *form the obligation-closure surface for the fixed-point packet.*

Proof. Start with the carrier and public obligation package. The carrier displays $G, L, Q, W, O, H, C, P, N$, and Definition 556.7 names carrier admission, classifier stability, fixed-witness persistence, non-escape, and ledger exactness over exactly those rows.

Attach the fixed-witness route using Definition 556.4 and Definition 556.10. These keep W tied to the displayed output and to the route through C, P, N.

Finally use Definition 556.13. It exhausts boundary attempts as GAP evidence and leaves no coordinate beyond $G, L, Q, W, O, H, C, P, N$. Hence the listed rows are the full obligation-closure surface. □

Theorem 556.16 (Generator fixed-point TasteGate carrier surface)**.** *Any AI-origin* ChapterTasteGate *consumer of an accepted* GeneratorFixedPoint$^{\uparrow}$ *packet may inspect only the finite carrier rows*

$$G, L, Q, W, O, H, C, P, N.$$

The consumer may read the authorized generator, constructor-list provenance, GeneratorClosure$^{\uparrow}$ *classifier, fixed-output witness, displayed output, componentwise transport, continuation route, package provenance, and local naming row, and may not inspect a host fixed-point principle, self-application oracle, or unlisted continuation.*

Proof. First project the admitted packet row by Definition 556.8. It exposes exactly $G, L, Q, W, O, H, C, P, N$ and no further carrier coordinate.

Next apply the TasteGate witness route. Definition 556.11 confines TasteGate-admissible witness reads to the displayed generator, classifier, witness, output, transport, route, provenance, and naming rows.

The audit non-escape ledger handles excluded reads. By Definition 556.13, a host fixed-point principle, self-application oracle, or unlisted continuation has no audit coordinate in the packet. □

Theorem 556.17 (Generator fixed-point obligation closure upgrade)**.** *The* GeneratorFixedPoint$^{\uparrow}$ *chapter reaches obligation closure for the finite packet*

$$F = (G, L, Q, W, O, H, C, P, N).$$

The closure surface is exactly the carrier of Definition 556.1, the source-admission row of Definition 556.5, the witness-ledger row of Definition 556.10, the audit non-escape ledger of Definition 556.13, and the TasteGate witness route of Definition 556.11. No public fixed-point read is accepted unless it factors through those rows.

Proof. Start from the carrier. The accepted packet is the displayed tuple $G, L, Q, W, O, H, C, P, N$, so carrier admission contributes exactly the finite source, classifier, witness, transport, route, provenance, and naming rows.

Use source admission and the witness ledger. The source row admits only the authorized generator with constructor-list provenance, while the witness ledger keeps every fixed-output read attached to W, O, C, P, N.

Apply the TasteGate witness route. It restricts AI-origin witness reads to the same displayed surface and therefore supplies the required TasteGate path without adding a host fixed-point theorem.

Finally apply the audit non-escape ledger. Boundary attempts have no coordinate outside the carrier, so the named rows form the obligation-closure surface. □

Theorem 556.18 (Generator fixed-point GAP boundary closure)**.** *Every attempted* GeneratorFixedPoint$^{\uparrow}$ *read outside the obligation-closure surface is a* GAP *boundary. In particular, a route that omits the carrier row, the source-admission row, the witness-ledger row, the audit non-escape ledger, or the TasteGate witness route cannot export a fixed-output witness, cannot detach Q from G, L, and cannot be repacked as a downstream* GeneratorClosure$^{\uparrow}$ *certificate.*

Proof. First inspect the carrier row. It contains no coordinate for a host fixed-point principle, self-application oracle, semantic completeness claim, or continuation outside the constructor-list provenance.

Next use source admission. A route missing G, L is not generator-internal and is recorded as boundary evidence before any witness read can occur.

Then apply the witness ledger and the TasteGate witness route. A public witness must pass through W, O, C, P, N and, for the AI-origin surface, through the displayed TasteGate path.

The audit non-escape ledger closes the remaining cases. Since every omitted row removes part of the finite route, the attempt remains a GAP boundary rather than a fixed-point certificate. □

Theorem 556.19 (Generator fixed-point boundary consumer exhaustion)**.** *For every downstream consumer of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet, the only accepted fixed-point route is the one exposed by the carrier, source-admission row, witness ledger, audit non-escape ledger, and TasteGate witness route. All other consumer reads remain boundary evidence, including classifier-free witness reads, witness-ledger bypasses, diagonal self-application attempts, and routes outside the constructor-list provenance.*

Proof. Begin with a downstream consumer read. If it is accepted, carrier admission gives the finite tuple $G, L, Q, W, O, H, C, P, N$ as its only source.

The source-admission row and witness-ledger row then force the read through G, L and through W, O, C, P, N. The classifier row Q remains attached to the same source surface by the existing transport and handoff rows.

For AI-origin reads, the TasteGate witness route supplies the displayed witness path. It permits the finite carrier route and does not create a separate route for diagonal or classifier-free reads.

If the consumer omits any of these rows, the audit non-escape ledger applies. The read has no carrier coordinate from which to export a fixed-point witness, so it is exhausted as boundary evidence. □

Theorem 556.20 (Generator fixed-point boundary consumer totality)**.** *Every downstream boundary consumer of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet either reads the fixed witness through the displayed route*

$$G, L, Q, W, O, H, C, P, N$$

or remains a GAP *boundary. There is no third route that exports a fixed witness while bypassing the constructor-list provenance* L*, the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier* Q*, or the fixed-output witness ledger.*

Proof. Consider a downstream boundary consumer. If it reads the fixed witness, Definition 556.21 forces the read through $G, L, Q, W, O, H, C, P, N$.

If the route omits constructor-list provenance or the classifier row, Definition 556.25 and Definition 556.28 record the route as GAP boundary evidence.

It remains to exclude a witness-only bypass. The witness ledger theorem confines every witness read to W, O, C, P, N, and the audit non-escape ledger excludes unlisted continuation routes. Thus every downstream boundary consumer is either on the displayed fixed-witness route or remains a GAP boundary. □

Theorem 556.21 (Generator fixed-point closure-consumer lock)**.** *Every* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *consumer that reads a fixed witness must read it through the displayed* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row* Q*, authorized generator row* G*, constructor-list provenance row* L*, fixed-output witness row* W*, output row* O*, componentwise* hsame *transport row* H*, displayed* Cont *route row* C*, package provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row* N*. A consumer route that omits one of these rows is not a fixed-witness read of the accepted packet.*

Proof. Start from the source admission row. It exposes the fixed-point packet only as the finite tuple $G, L, Q, W, O, H, C, P, N$, with no ambient fixed-point operator or unlisted continuation coordinate.

The classifier transport row keeps the $\mathsf{GeneratorClosure}^{\uparrow}$ classifier attached to the authorized generator row and constructor-list provenance after any componentwise transport. Thus a consumer cannot detach Q from G and L and still read the same fixed witness.

Finally use the witness ledger row. The public witness route passes through W, O, C, P, N, so the consumer receives the displayed fixed-output route and no separate witness source. □

Theorem 556.22 (Generator fixed-point fixed-witness source determinacy)**.** *Let two accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packets present the same displayed source surface* $G, L, Q, W, O, H, C, P, N$ *for the generator row, constructor-list provenance, classifier row, fixed-output witness, output row, transport row, continuation routes, package provenance, and local naming row. Then they expose the same fixed-witness route to every public consumer. A proposed route that lacks the constructor-list provenance* L *or the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row* Q *is boundary evidence, not an alternate source for the fixed witness.*

Proof. Use the admitted packet row to reduce both accepted packets to the same finite source surface. Since the displayed rows agree, a public read begins from the same generator, provenance, classifier, witness, output, route, package, and naming coordinates.

The transport row identifies the consumer route through the shared H and C rows. Therefore the fixed-output witness exposed downstream is the same route through W and O in both packets.

If a proposed route omits L or Q, the diagonal obstruction scope applies. The route is missing part of the authorized generator-classifier surface and is recorded as a GAP boundary rather than as a competing fixed-witness source. □

Theorem 556.23 (Generator fixed-point audit ground-route exhaustion)**.** *The public audit ground route for an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet is exhausted by the finite fixed-witness route locked by Definition 556.21, the source determinacy of Definition 556.22, and the AI-roadmap TasteGate discipline cited in Definition 556.11. Together with the audit non-escape ledger, these rows leave no public audit coordinate for a host fixed-point theorem, a self-application oracle, or a witness route outside* $G, L, Q, W, O, H, C, P, N$.

Proof. First apply the closure-consumer lock. Any public fixed-witness read must pass through the displayed generator, constructor-list, classifier, witness, output, transport, continuation, package, and naming rows.

Next apply fixed-witness source determinacy. Once the displayed surface is fixed, every accepted packet exposes the same witness route, while routes lacking L or Q remain boundary evidence.

The TasteGate witness route supplies the permitted audit discipline for AI-roadmap witness reads. It accepts only the displayed finite route and rejects self-application routes outside that packet.

Finally use the audit non-escape ledger. Since the ledger has no coordinate beyond $G, L, Q, W, O, H, C, P, N$ and the TasteGate route, the public audit ground route is exhausted by the finite fixed-witness surface. □

Theorem 556.24 (Generator fixed-point GeneratorClosure handoff)**.** *Every fixed-point read of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet first factors through the displayed* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row* Q*. The read may then consume the authorized generator row* G*, constructor-list provenance* L*, fixed-output witness* W*, output row* O*, componentwise transport row* H*, continuation route row* C*, package provenance row* P*, and local naming row* N*, but it has no route to a fixed witness before the classifier row has been displayed.*

Proof. Start with carrier admission from Definition 556.1. The accepted packet displays $G, L, Q, W, O, H, C, P, N$, so the classifier row is part of the finite source surface rather than a derived host fact.

Use the source admission and transport rows next. Definition 556.5 and Definition 556.6 keep Q attached to the authorized generator and constructor-list provenance under every accepted read.

Finally apply the fixed-witness and audit-ground rows. Definition 556.4, Definition 556.21, and Definition 556.23 route the witness through Q, W, O, H, C, P, N and leave no classifier-free fixed-point export. □

Theorem 556.25 (Generator fixed-point boundary GAP dependency)**.** *For an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet, a consumer route missing constructor-list provenance L is exactly a* GAP*-boundary read for the fixed-point surface. Such a route cannot export the fixed-output witness W, cannot detach Q from the authorized generator row G, and cannot be repacked as a downstream* $\mathsf{GeneratorClosure}^{\uparrow}$ *witness.*

Proof. Read the attempted route through the carrier. The only constructor-list provenance coordinate is L, so a route that omits it has no displayed source row authorizing a generator-internal continuation.

Apply non-escape and source determinacy. By Definition 556.3 and Definition 556.22, a route without L is boundary evidence, not an alternate source for the same fixed witness.

The witness ledger and audit rows finish the dependency claim. Definition 556.10 permits W only through the displayed witness ledger, while Definition 556.13 excludes any host fixed-point principle or unlisted continuation that could replace the missing provenance. □

Theorem 556.26 (Generator fixed-point downstream unblock package)**.** *Every downstream* $\mathsf{GeneratorClosure}^{\uparrow}$ *consumer of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet receives only the finite rows*

$$G, L, Q, W, O, H, C, P, N.$$

The package exposes the authorized generator, constructor-list provenance, classifier, fixed witness, displayed output, transports, routes, provenance, and local naming row, and it exports no host fixed-point principle, self-application oracle, semantic completeness claim, or continuation outside the displayed constructor list.

Proof. First assemble the public package. The carrier definition, public obligation package, and closure-consumer lock show that every downstream read is a projection of $G, L, Q, W, O, H, C, P, N$.

Next apply the GeneratorClosure handoff. Definition 556.24 makes Q the first classifier row consumed by fixed-point readers, while the transport theorem keeps it attached to G and L.

Finally use boundary dependency and audit exhaustion. Routes missing constructor-list provenance are GAP boundaries by Definition 556.25, and Definition 556.23 leaves no downstream coordinate for a host fixed-point theorem, self-application oracle, semantic completeness, or unlisted continuation. □

Theorem 556.27 (Generator fixed-point carrier row totality)**.** *Every accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet has a total displayed carrier row*

$$F = (G, L, Q, W, O, H, C, P, N).$$

A public or downstream read of the fixed-point surface must account for all nine rows: the authorized generator row G, constructor-list provenance L, $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier row Q, fixed-output witness W, output row O, componentwise transport row H, continuation route row C, package provenance row P, and local naming row N. Omitting any one of these rows makes the read a partial boundary attempt, not an accepted carrier read.*

Proof. Start from Definition 556.1. The definition displays the accepted packet as the nine-coordinate tuple $G, L, Q, W, O, H, C, P, N$, so carrier admission is total over those rows.

Use the admitted packet row and the public obligation package. They make every public read a projection of the same finite tuple, with the source, classifier, witness, output, transport, route, provenance, and naming coordinates all present.

If a route omits one coordinate, the non-escape and audit rows apply. The attempted read no longer has the complete fixed-point source surface, so it is recorded as boundary evidence rather than as an accepted carrier projection. □

Theorem 556.28 (Generator fixed-point classifier row GAP determinacy)**.** *For an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet* $F = (G, L, Q, W, O, H, C, P, N)$*, the classifier route is determined by the displayed pair* L, Q*. A consumer route carrying both rows reads the* $\mathsf{GeneratorClosure}^{\uparrow}$ *classifier through* Q *under the constructor-list provenance* L*. A route missing* L *or missing* Q *is a* GAP *boundary and cannot be read as an alternate classifier for the same fixed-output witness* W*.*

Proof. Read the classifier route through the carrier. The constructor-list provenance and closure classifier are displayed as the two coordinates L and Q, so a valid classifier read must pass through both.

When both rows are present, the GeneratorClosure handoff applies. The row Q classifies the generator-internal continuation under the provenance supplied by L, and the witness ledger keeps W attached to that same classifier surface.

If L is absent, the boundary dependency theorem records the route as lacking constructor-list provenance. If Q is absent, the handoff theorem gives no classifier coordinate before the witness read.

In either missing-row case, the package provenance records a GAP boundary. The attempted route therefore cannot supply a competing classifier for W. □

Theorem 556.29 (Generator fixed-point closure-consumer NameCert handoff)**.** *Every downstream* $\mathsf{GeneratorClosure}^{\uparrow}$ *consumer of an accepted* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *packet reads the fixed-point certificate through*

$$Q, W, H, C, P, N$$

after the source rows G, L *have been displayed. The classifier row* Q *is the consumer-facing* $\mathsf{GeneratorClosure}^{\uparrow}$ *handoff;* W *supplies the fixed-output witness,* H *and* C *transport and replay the route,* P *records package provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{GeneratorFixedPoint}^{\uparrow}}$ *row. No consumer receives a classifier-free fixed-point witness.*

Proof. First display the source rows G, L using source admission. This fixes the authorized generator and constructor-list provenance that make the continuation generator-internal.

Next read the consumer-facing classifier. The GeneratorClosure handoff forces every fixed-point read through Q before W can be exported.

Then replay the local NameCert route. The transport row H preserves the displayed surface, the continuation row C replays the read, the package row P records provenance, and the local naming row N exposes only that finite route.

Finally use classifier-row determinacy. A route missing L or Q is a GAP boundary, so no downstream $\mathsf{GeneratorClosure}^{\uparrow}$ consumer can receive a classifier-free fixed-point witness. □

Theorem 556.30 (Generator fixed-point scoped package)**.** *The scoped* $\mathsf{GeneratorFixedPoint}^{\uparrow}$ *package is exactly the finite surface consisting of* BHist, $\mathsf{GeneratorClosure}^{\uparrow}$, hsame, Cont, Pkg, $\mathsf{NameCert}$*, the authorized generator row* G*, the constructor-list row* L*, the classifier row* Q*, the fixed-output*

witness W, and the output row O. No scoped consumer can add a host fixed-point principle, self-application oracle, or continuation outside that displayed package.

Proof. Start from Definition 556.1. The carrier exposes the packet $F = (G, L, Q, W, O, H, C, P, N)$, so the scoped source surface is the finite tuple together with the ambient BHist, hsame, Cont, Pkg, and NameCert rows named by the packet.

Use the source admission row and the transport row. They keep G, L, Q, W, O attached to the authorized generator and constructor-list provenance under every displayed route.

Apply the witness ledger row, diagonal obstruction scope, and closure-consumer NameCert handoff. These rows exhaust the witness, boundary, and downstream consumer routes, leaving no package coordinate for a host fixed-point theorem, self-application oracle, or unlisted continuation. □

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointUp StdBridge`

Theorem 556.31 (Generator fixed-point continuation fixed-output scope)**.** *Any fixed-output consumer of an accepted* GeneratorFixedPoint$^{\uparrow}$ *packet reads the witness row W only after the displayed route through G, L, Q, O, H, C, P, N has been exposed. The consumer therefore receives the fixed-output witness as a scoped continuation read of the finite package, not as a host-level fixed-point operation.*

Proof. First expose the source rows using source admission. The authorized generator row G and constructor-list provenance L are required before the classifier row can be read.

Next apply the GeneratorClosure handoff and classifier-row determinacy. The classifier row Q stays attached to G, L, and a route missing either row is boundary evidence rather than a fixed-output source.

Finally use the closure-consumer NameCert handoff. It routes the witness through W after G, L, Q, O, H, C, P, N have been displayed, so the fixed-output read remains a finite Cont route inside the scoped package. □

Theorem 556.32 (Generator fixed-point downstream non-escape scope)**.** *Every downstream* GeneratorClosure$^{\uparrow}$ *consumer of an accepted* GeneratorFixedPoint$^{\uparrow}$ *packet receives only constructor-list provenance, classifier, fixed-output witness, output, transport, route, package, and local naming data from $G, L, Q, W, O, H, C, P, N$. It receives no host fixed-point principle, no self-application oracle, and no continuation outside the displayed constructor-list provenance.*

Proof. Begin with the downstream unblock package. It gives every downstream GeneratorClosure$^{\uparrow}$ consumer only the finite rows $G, L, Q, W, O, H, C, P, N$.

Use the boundary GAP dependency and diagonal obstruction scope for excluded routes. Attempts missing constructor-list provenance, classifier access, or the displayed witness route are recorded as GAP boundary evidence.

Finally apply the audit ground-route exhaustion. The TasteGate route and audit ledger exhaust the public fixed-witness surface, so there is no downstream coordinate for a host fixed-point theorem, self-application oracle, or continuation outside L. □

Closure status for NameCert$_{\text{GeneratorFixedPoint}^{\uparrow}}$.

Constructive story: A GeneratorFixedPoint$^{\uparrow}$ packet is generated from an authorized generator row, finite constructor-list provenance, a GeneratorClosure$^{\uparrow}$ classifier row, a fixed-output witness, a displayed output row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GeneratorFixedPoint}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 556.30, Definition 556.31, and Definition 556.32 bind the scoped carrier package, fixed-output continuation route, and downstream non-escape boundary for the finite fixed-point packet.

Formal verification: FormalStatus(NameCert$_{\text{GeneratorFixedPoint}\uparrow}$, bridgeChecked)

Lean4 ✓ `BEDC.Derived.GeneratorFixedPointUp.taste gate`

Lean target: `BEDC.Derived.GeneratorFixedPointUp.GeneratorFixedPointTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge checking, host fixed-point theorems, self-application, semantic completeness, or continuations outside the displayed constructor-list provenance.

Upgrade path: No further upgrade is claimed on the formal axis; the displayed fixed-witness packet has a checked bridge route.

Chapter 557

A Concrete Naming Certificate for $\mathsf{KernelAcceptanceTrace}^{\uparrow}$

The $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ surface records kernel acceptance as a finite trace packet: a submitted candidate, the type and classifier under which it is checked, the accepted stamp, the environment row, the displayed axiom ledger, transport, continuation, provenance, and local naming row. It is an audit witness for what the kernel admitted and which dependency rows the admission exposes, not a proof of kernel correctness.

The candidate row and accepted stamp are read as an inscription-event record surface in the sense of chapter 519: the submitted candidate, checked type, classifier, stamp, and environment give the event record that the trace consumes. The axiom ledger remains $\mathsf{KernelAcceptanceTrace}^{\uparrow}$'s own displayed row, so dependency exposure is still governed by the trace packet rather than by the inscription-event chapter.

Definition 557.1 (KernelAcceptanceTrace carrier)**.** *A* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, T, C, S, E, L, H, R, P, N)$$

with the following displayed coordinates:

1. *A is the candidate row naming the term, declaration, inductive clause, or recursor instance submitted to the kernel;*
2. *T is the type row against which the candidate is checked;*
3. *C is the kernel classifier row recording the admitted declaration class and the local acceptance predicate for the candidate at type T;*
4. *S is the kernel stamp row that records the accepted name, module position, and replay boundary for the acceptance event;*
5. *E is the environment row in which the stamp is valid;*
6. *L is the displayed axiom ledger for the accepted trace, including the dependency rows exposed by the strict audit query;*
7. *H stores componentwise* hsame *transports for the candidate, type, classifier, stamp, environment row, axiom ledger, package provenance, and local naming row;*
8. *R stores the displayed* Cont *routes through which consumers read the candidate, transport the classifier, inspect the stamp, query the environment ledger, and replay the axiom ledger;*

9. *P is the* Pkg *provenance over* BHist*, candidate rows, type rows, kernel classifiers, kernel stamps, environment ledgers, axiom ledgers,* hsame*,* Cont*, and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}$ *row.*

The carrier has no hidden coordinate for ambient kernel soundness, host runtime trust, compiler trust, unqueried dependency ancestry, or a stronger paper-side theorem than the accepted declaration states.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceTraceUp.KernelAcceptanceTraceTasteGate single carrier alignment`

Theorem 557.2 (KernelAcceptanceTrace NameCert obligations)**.** *For every accepted* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier*

$$K = (A, T, C, S, E, L, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure for the displayed packet K and its candidate, type, classifier, stamp, environment, axiom ledger, transport, continuation, provenance, and naming rows;*
2. *classifier transport, requiring every accepted comparison of traces to move A, T, C, S, E, L, P, N only by the componentwise* hsame *rows in H and the displayed* Cont *routes in R;*
3. *kernel-stamp exposure, requiring consumers to read acceptance through the accepted name, module position, and replay boundary in S;*
4. *environment-ledger exposure, requiring the accepted trace to route through the environment row E and the displayed axiom ledger L before any dependency claim is consumed;*
5. *non-escape, excluding hidden kernel soundness, host runtime trust, compiler trust, unqueried dependency ancestry, and theorem-content strengthening outside the accepted declaration.*

Lean4 ✓ `BEDC.Derived.KernelAcceptanceTraceUp.KernelAcceptanceTraceCarrier namecert obligations`

Proof. Carrier exposure is projection from Definition 557.1. The packet lists the candidate, type, classifier, stamp, environment, axiom ledger, transport, continuation, provenance, and local naming rows as finite coordinates.

Classifier transport is componentwise. The hsame rows in H preserve the displayed coordinates, and the Cont rows in R give only the routes named in the packet, so the accepted trace cannot be compared through an unrecorded classifier.

The stamp and environment obligations then force every acceptance read through S and E. The axiom ledger L is the only displayed ancestry row for dependency classification.

Finally inspect the carrier inventory. Since no coordinate names kernel soundness, host runtime trust, compiler trust, unqueried ancestry, or theorem strengthening beyond the accepted declaration, no local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}$ consumer can export such data. □

Theorem 557.3 (KernelAcceptanceTrace axiom ledger non-escape)**.** *For every accepted* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier*

$$K = (A, T, C, S, E, L, H, R, P, N),$$

the exported dependency surface is exactly the displayed axiom ledger L together with the stamp, environment, transport, continuation, provenance, and local naming rows S, E, H, R, P, N. No consumer may obtain a hidden Classical *source,* Quot *source,* propext *source, or any other unlisted dependency by reading the candidate row, type row, classifier row, or acceptance stamp alone.*

Proof. Read the dependency side of the carrier through L. The ledger is a named coordinate of the finite packet and is the only row that classifies transitive dependency ancestry.

The stamp S and environment row E identify where the accepted declaration lives, while H, R, P, N preserve and package the route by transport, continuation, provenance, and local naming. These rows can expose the displayed ledger but do not add a second ancestry channel.

Therefore a hidden Classical, Quot, propext, or other unlisted source would require an extra coordinate outside K. The local trace certificate has no such coordinate, so the accepted trace exports only the displayed axiom ledger. □

Theorem 557.4 (KernelAcceptanceTrace classifier transport)**.** *For every accepted* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier*

$$K = (A, T, C, S, E, L, H, R, P, N),$$

accepted trace comparison transports only the candidate row A, type row T, classifier row C, stamp row S, environment row E, axiom ledger L, provenance row P, and local name N through the displayed hsame *rows in H and the displayed* Cont *routes in R.*

Proof. Project the carrier coordinates and the local NameCert obligations. The comparison-facing data are exactly A, T, C, S, E, L, P, N.

The transport row H is componentwise, and the route row R is displayed in the packet. Thus accepted trace comparison can move the listed rows, but it has no separate classifier oracle, host runtime row, compiler row, or hidden dependency source. □

Theorem 557.5 (KernelAcceptanceTrace stamp-environment exposure)**.** *Every acceptance read from an accepted* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier factors through the stamp row S and the environment row E before the axiom ledger L can be consumed. The ledger cannot be read as an ambient ancestry claim detached from the accepted name, module position, replay boundary, and environment ledger.*

Proof. Begin with the carrier. The stamp row S records the accepted name, module position, and replay boundary, while E records the environment in which that stamp is valid.

Apply the axiom-ledger non-escape theorem. Dependency ancestry is exposed by L, but it is exported together with S, E, H, R, P, N rather than through the candidate, type, or classifier rows alone.

Therefore any acceptance read must first locate the stamp and environment. Only then can it consume the displayed ledger as the dependency surface of that accepted trace. □

Theorem 557.6 (KernelAcceptanceTrace accepted-declaration boundary)**.** *The accepted declaration boundary of a* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *carrier cannot strengthen theorem content beyond the declared type row T and the displayed dependency ledger L. It exports the submitted candidate, accepted type, classifier, stamp, environment, ledger, transport, continuation, provenance, and local name, but no stronger theorem, kernel-correctness claim, or host-runtime trust row.*

Proof. Read the declaration through $A, T, C, S, E, L, H, R, P, N$. The type row T is the only displayed statement boundary for the accepted candidate, and L is the only displayed dependency ancestry.

Use classifier transport to keep comparisons inside the same rows. Transport may preserve the declared type and ledger, but it does not introduce a stronger theorem statement.

Finally apply stamp-environment exposure and axiom-ledger non-escape. Since the accepted trace must pass through S, E and has no hidden ancestry coordinate, the boundary cannot add kernel correctness, host runtime trust, or theorem content beyond T and L. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ packet is generated from a submitted candidate, type row, kernel classifier, acceptance stamp, environment row, displayed axiom ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 557.1, Definition 557.2, Definition 557.3, Definition 557.4, Definition 557.5, and Definition 557.6 bind the finite trace carrier, local naming obligations, classifier transport, kernel stamp, environment row, displayed axiom-ledger boundary, and accepted-declaration boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.KernelAcceptanceTraceUp.KernelAcceptanceTraceCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, kernel correctness, host runtime correctness, compiler correctness, theorem-content strengthening beyond the accepted declaration, or bridge checking.

Upgrade path: The chapter is at the top of the paper closure axis; bridge checking remains a separate verification-axis obligation.

Chapter 558

A Zeckendorf-Variant Naming Certificate for $\mathsf{ZCarry}^{\uparrow}$

The $\mathsf{ZCarry}^{\uparrow}$ surface packages the generated Zeckendorf carry classifier as a finite $\mathsf{NameCert}$ object over displayed BHist words. Its source is the single non-normal carry window `word_011`, its target is the normal window `word_100`, and its classifier row records carry transport without identifying the two histories by hsame.

The dependency anchors are $\mathsf{ZeckendorfNormal}^{\uparrow}$, ZCarry, $\mathsf{ZNormal}$, hsame, BHist, and the displayed words of Definition 278.2.

The accepted carry packet consumes the normal-target witness of chapter 278 and the classifier boundary of chapter 567 before downstream consumers read the target row. This dependency is substantive: $\mathsf{ZeckendorfNormal}^{\uparrow}$ supplies target normality, while $\mathsf{ZeckendorfCarryClassifier}^{\uparrow}$ supplies the non-hsame carry-transport boundary.

Definition 558.1 (Zeckendorf carry concrete carrier)**.** *A* $\mathsf{ZCarry}^{\uparrow}$ *concrete carrier is a finite* BHist *packet*

$$K = (h, k, G, S, T, B, L, N)$$

with the following displayed rows:

1. $h =$ `word_011` *is the non-normal source window;*
2. $k =$ `word_100` *is the normal target window;*
3. G *is the generated carry row* $\mathsf{ZCarry}(h, k)$*;*
4. S *stores the source rejection* $\neg\mathsf{ZNormal}(h)$*;*
5. T *stores the target acceptance* $\mathsf{ZNormal}(k)$*;*
6. B *stores the classifier boundary* $\neg\mathsf{hsame}(h, k)$*;*
7. L *is the ledger row recording the visible generator, source rejection, target acceptance, and non-*hsame *boundary;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *row.*

The classifier compares carry packets by the generated row G and the boundary row B. It exports no arithmetic value, Fibonacci weight, quotient of histories, or equality between the source and target words.

Theorem 558.2 (Zeckendorf carry generator boundary)**.** *Every accepted* $\mathsf{ZCarry}^{\uparrow}$ *packet is generated by the single displayed carry row*

$$\mathsf{ZCarry}(\texttt{word_011}, \texttt{word_100}).$$

Consequently the source row is forced to `word_011` *and the target row is forced to* `word_100`*; no second carry generator, hidden target, or arithmetic equality row belongs to the carrier.*

Proof. Unpack the carrier of Definition 558.1. The only generator coordinate is G, and G is the closed carry row from the source window to the target window.

The ledger row L records the same generator together with the source rejection, target acceptance, and non-hsame boundary. Since the carrier has no coordinate for another generator or for a value interpretation, an accepted packet can only expose the displayed carry row. □

Theorem 558.3 (Zeckendorf carry NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *surface has five obligations:*

1. *carrier habitation: the displayed packet* $K = (h, k, G, S, T, B, L, N)$ *is inhabited by* `word_011`*,* `word_100`*, and the generated carry row;*
2. *classifier transport: accepted packets are compared through the generated carry row* G*, not through* $\mathsf{hsame}(h, k)$*;*
3. *source boundary: the source row carries* $\neg\mathsf{ZNormal}(\texttt{word_011})$*;*
4. *target boundary: the target row carries* $\mathsf{ZNormal}(\texttt{word_100})$*;*
5. *ledger boundary: the packet records carry transport, source rejection, target acceptance, and non-*hsame *separation without exporting an arithmetic value or history quotient.*

Proof. Carrier habitation is the displayed packet of Definition 558.1. The source and target words are the two fixed BHist histories of Definition 278.2, and the generator row is the closed ZCarry constructor.

The classifier transport obligation reads the G and B rows together. The G row licenses carry movement from source to target, while B prevents that movement from being read as hsame.

The source and target boundaries are the normal-form rows already exposed by $\mathsf{ZeckendorfNormal}^{\uparrow}$: `word_011` is rejected and `word_100` is accepted. The ledger row packages exactly these witnesses, so no arithmetic value, quotient, or equality row is available to the public surface. □

Theorem 558.4 (Zeckendorf carry non-hsame boundary)**.** *For every accepted* $\mathsf{ZCarry}^{\uparrow}$ *packet* $K = (h, k, G, S, T, B, L, N)$*, the carry classifier separates transport from history sameness:* G *records* $\mathsf{ZCarry}(h, k)$*, while* B *records* $\neg\mathsf{hsame}(h, k)$*. Thus the packet can move a displayed non-normal window to a displayed normal window only as a classifier step, not as a kernel identification of histories.*

Proof. Project the G coordinate of the packet. By Definition 558.2, it is the displayed carry from `word_011` to `word_100`.

Project the boundary coordinate B. It records that the same source and target histories are not related by hsame. Hence carry transport and history sameness occupy distinct rows of the certificate.

The conclusion is a scope statement about the carrier: consumers may read the carry row, but they cannot replace it by a kernel history identification. The ledger L preserves this separation as part of the visible certificate surface. □

Theorem 558.5 (Zeckendorf carry normal-target handoff)**.** *Every downstream consumer of a* $\mathsf{ZCarry}^{\uparrow}$ *packet receives a normal target row only after the generator boundary, source rejection, and non-*hsame *separation have been fixed. The handoff to* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *is the target witness* $\mathsf{ZNormal}(\texttt{word_100})$*; it does not carry a Fibonacci value-preservation claim or a uniqueness-of-value row.*

Proof. Start with an accepted packet and read its generator row. The generator boundary fixes the source as `word_011` and the target as `word_100`.

Next read the source and target normal-form rows. The source row records rejection by the normal-form classifier, while the target row records acceptance by the same structural classifier.

Finally read the non-hsame boundary. Since the source and target are not identified as histories, the downstream $\mathsf{ZeckendorfNormal}^{\uparrow}$ consumer receives only the normal target witness and the ledger explaining how it was reached. Arithmetic value preservation remains outside this certificate surface. □

Theorem 558.6 (Concrete $\mathsf{ZCarry}^{\uparrow}$ packet surface)**.** *The carrier, generator boundary, NameCert obligations, non-*hsame *boundary, and normal-target handoff form the concrete* $\mathsf{ZCarry}^{\uparrow}$ *certificate surface. Its public content is finite carry transport from* `word_011` *to* `word_100` *together with the proof-facing separation between carry classification and history sameness.*

Proof. The carrier fixes the eight displayed rows of the packet. The generator boundary pins the only carry row, and the NameCert obligations state how the source, target, classifier, stability, and ledger fields are read.

The non-hsame boundary prevents the generator from becoming a history equality. The normal-target handoff identifies the only downstream normal-form consumer row.

Together these rows exhaust the concrete certificate surface. There is no carrier coordinate for Fibonacci values, value preservation, uniqueness of arithmetic value, or a quotient of histories, so those claims are not exported by $\mathsf{ZCarry}^{\uparrow}$. □

Corollary 558.7 (Zeckendorf carry strict local obstruction)**.** *Let* $K = (h, k, G, S, T, B, L, N)$ *be an accepted* $\mathsf{ZCarry}^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is attached to a consumer of* K *and is used against a named converse or reuse route. Then the row is a strict local obstruction: it blocks only the route whose read asks to add or replace one of the displayed carry coordinates*

$$\mathsf{ZCarry}(\texttt{word_011}, \texttt{word_100}), \quad S : \neg\mathsf{ZNormal}(\texttt{word_011}), \quad T : \mathsf{ZNormal}(\texttt{word_100}), \quad B : \neg\mathsf{hsame}(h, k), \quad L,$$

Every permitted consumer-facing use factors through these rows, with source `word_011` *and target* `word_100` *fixed. A request for a second carry generator, a hidden target, an* $\mathsf{hsame}(h, k)$ *identification, an arithmetic equality, a Fibonacci value row, a history quotient, a global normalizer, a host theorem, or hidden external validation has no coordinate in the accepted packet.*

Proof. Apply Definition 558.2. It fixes the only generated carry row as

$$\mathsf{ZCarry}(\texttt{word_011}, \texttt{word_100}),$$

forces the source row to `word_011` and the target row to `word_100`, and excludes any second generator, hidden target, or arithmetic equality row from the carrier. Thus a consumer cannot reinterpret a negative row as ranging over arbitrary carry windows or over an ambient arithmetic comparison; the only carry window available to the consumer is the displayed one.

Next apply Definition 558.3. The carrier-habitation obligation supplies the displayed packet K. The classifier-transport obligation routes comparison through G rather than through $\mathsf{hsame}(h, k)$. The source and target boundary obligations expose exactly $S : \neg\mathsf{ZNormal}(\texttt{word_011})$ and $T : \mathsf{ZNormal}(\texttt{word_100})$. The ledger obligation packages carry transport, source rejection, target acceptance, and non-hsame separation without exporting an arithmetic value or history quotient.

Now let the proposed converse or reuse route be consumer-facing. To consume K it must first read the generator, normality, separation, ledger, and local naming rows just listed. If the displayed failure, refusal, separation, or countermodel row denies one of those reads, it blocks that named route at the corresponding packet coordinate. If the route instead asks for a second generator, hidden target, source-target hsame proof, arithmetic equality, value row, quotient, global normalizer, host theorem, or external validation channel, the carrier and obligation theorems give no coordinate through which that request can be read.

Therefore the obstruction is exact and local. It is a negative certificate for the named converse or reuse route inside the displayed $\mathsf{ZCarry}^{\uparrow}$ packet, and it supplies no global impossibility statement, no host-side theorem, and no hidden validation of an unlisted route. □

Chapter 559

A Concrete Naming Certificate for $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$

The $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ packet records the finite handoff from a selected regular-Cauchy tail window to a $\mathsf{Real}^{\uparrow}$ seal. It is the concrete NameCert surface used when a diagonal-limit consumer asks for a rational precision request, a StreamName window, a RegSeqRat row, and the selected finite-tail witness that justifies the real-facing read.

Definition 559.1 (Finite-tail diagonal seal carrier)**.** *A* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (q, s, r, w, e, h, c, p, n)$$

with the following visible coordinates:

1. *q is the rational precision request, read through the displayed* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows;*
2. *s is the* $\mathsf{StreamName}^{\uparrow}$ *window selecting the finite diagonal observation interval;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regularity ledger supplies the source observations;*
4. *w is the selected finite-tail witness, displaying the bounded tail segment that answers q inside the window s;*
5. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after q, s, r, w have been exposed;*
6. *h records componentwise* hsame *transports for the precision request, StreamName window, RegSeqRat row, finite-tail witness, and real-seal row;*
7. *c records the displayed* Cont *routes by which consumers inspect the rows;*
8. *p is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and selected-tail evidence;*
9. *n is the local* $\mathsf{NameCert}_{\mathsf{FiniteTailDiagonalSeal}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no host real equality, no unbounded diagonal object, no ambient completeness theorem, and no selected limit outside the finite witness row w.

Theorem 559.2 (Finite-tail diagonal seal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteTailDiagonalSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for $D = (q, s, r, w, e, h, c, p, n)$, precision routing through q, finite-window routing through s, regular-Cauchy source routing through r, selected-tail*

witness exactness through w, $\mathsf{Real}^{\uparrow}$ *seal handoff through e, componentwise* hsame *transport through h, displayed* Cont *route replay through c, package provenance through p, and local naming through n.*

Proof. Project the carrier of Definition 559.1. The rows $q, s, r, w, e, h, c, p, n$ display the precision request, finite window, regular rational source, selected tail witness, real seal, transports, routes, package provenance, and local naming evidence.

The precision and source obligations are read before the seal. A consumer first reads q, then the selected StreamName window s, and then the RegSeqRat row r whose regularity ledger supplies the observations inside that window.

The selected witness obligation is the row w. It gives the finite tail segment that answers the request and prevents the handoff from appealing to an unbounded diagonal stream or an existential limit choice.

Finally the rows h, c, p, n account for transport, route replay, provenance, and local naming. Since the carrier has no quotient-stream coordinate, no host real equality coordinate, and no ambient-completeness coordinate, the obligation surface is exhausted by the displayed rows. □

Theorem 559.3 (Finite-tail diagonal seal window factorization)**.** *For an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier $D = (q, s, r, w, e, h, c, p, n)$, every finite-tail read factors through the StreamName window s and the selected witness row w before any* $\mathsf{Real}^{\uparrow}$ *seal row is consumed. The factorization may use the regular-Cauchy source row r, the precision row q, the componentwise transports h, and the displayed routes c, but it has no route through a quotient stream, an infinite diagonal object, or a host real equality.*

Proof. Read the finite-window coordinates from the carrier. The only window coordinate is s, and the only selected tail coordinate is w.

Use the NameCert obligations of Definition 559.2. They force precision routing through q, source routing through r, and selected-tail witness exactness through w before the real-facing row e is reached.

Transport and route replay are then confined to h and c. The package and local certificate rows p, n record that this is a finite packet, so the factorization has no coordinate for quotient streams, infinite diagonal data, or host real equality. □

Theorem 559.4 (Finite-tail diagonal seal window lock)**.** *For an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier $D = (q, s, r, w, e, h, c, p, n)$, the selected finite-tail witness row w is readable only under the selected* $\mathsf{StreamName}^{\uparrow}$ *window row s and the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row r. Any* $\mathsf{Real}^{\uparrow}$ *seal handoff through e preserves that same window lock through the componentwise transports h, the displayed routes c, the package provenance p, and the local* NameCert *row n.*

Lean4 ✓ `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealCarrier window lock`

Proof. Project the carrier of Definition 559.1. The row w is listed only as the selected finite-tail witness for the window s and the regular-rational source row r.

Apply the NameCert obligation theorem. It routes finite-window exposure through s, regular-Cauchy source exposure through r, and selected-tail witness exactness through w before the seal row e can be consumed.

Use the window factorization theorem to keep every finite-tail read under the same s, r, w boundary. The transports h, routes c, provenance p, and local name n replay that displayed boundary rather than selecting a second window.

Finally apply the non-escape boundary. Since the packet has no quotient stream, unbounded diagonal, host equality, ambient completeness, or external limit coordinate, a Real handoff through e cannot release w from the selected window and source rows. □

Theorem 559.5 (Finite-tail diagonal seal Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier receives the seal row* e *only through the displayed chain*

$$q, s, r, w, h, c, p, n.$$

Thus the real-facing handoff consumes a rational precision request, a finite StreamName window, a RegSeqRat source row, a selected finite-tail witness, componentwise transport, route replay, package provenance, and local naming. It does not consume host real equality, quotient-name equality, an ambient completeness principle, or a selected limit outside the finite witness.

Lean4 ✓ `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealCarrier real handoff`

Proof. Start with the carrier row e. It is a $\mathsf{Real}^{\uparrow}$ seal coordinate inside the same finite packet as q, s, r, w, h, c, p, n.

Apply the window factorization theorem. The seal-facing read must pass through the precision, window, source, and witness rows before it reaches e.

The rows h and c then replay the displayed transport and continuation data, while p and n provide provenance and local naming. Because the carrier lists no host equality, quotient equality, ambient completeness, or external limit coordinate, the real-facing handoff is exactly the displayed chain. □

Theorem 559.6 (Finite-tail diagonal seal non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *packet factors through the finite rows*

$$q, s, r, w, e, h, c, p, n.$$

The consumer may read the rational precision request, selected StreamName window, RegSeqRat source, finite-tail witness, $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and it cannot read quotient streams, host real equality, unbounded diagonal streams, ambient completeness, or a choice-selected limit.*

Proof. Project the accepted carrier and the NameCert obligation surface. The rows $q, s, r, w, e, h, c, p, n$ are the complete displayed packet.

Use the window factorization theorem for all finite-tail observations. It routes tail reads through s and w with the source and precision rows visible.

Use the Real handoff theorem for real-facing consumers. It routes seal reads through q, s, r, w, h, c, p, n before e is consumed.

Since the carrier has no remaining coordinate beyond the displayed rows, no downstream consumer can extract quotient-stream data, host real equality, unbounded diagonal data, ambient completeness, or a choice-selected limit from the packet. □

Theorem 559.7 (Finite-tail diagonal seal selected-tail exactness)**.** *For an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier*

$$D = (q, s, r, w, e, h, c, p, n),$$

the selected-tail witness row w *is exact for the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* s*. A consumer may read* w *only as the finite tail segment selected under* s *from the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* r *for the rational precision request* q*. The witness is not a quotient stream, an unbounded diagonal stream, a host real equality, or a selected limit outside the finite packet.*

Lean4 ✓ `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealCarrier selected tail exactness`

Proof. Project the carrier rows. The only selected-tail coordinate is w, and the carrier names it relative to the window row s, source row r, and precision request q.

Use the NameCert obligations and window lock. They force finite-window routing through s, source routing through r, and selected-tail witness exposure through w before the seal row e can be consumed.

Apply non-escape for the negative boundary. Since no quotient stream, unbounded diagonal stream, host equality, ambient completeness, or selected limit is present as a carrier coordinate, w is exact only as the displayed finite witness. □

Theorem 559.8 (Finite-tail diagonal seal RegSeqRat source coverage)**.** *For an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier*

$$D = (q, s, r, w, e, h, c, p, n),$$

the row r is the only $\mathsf{RegSeqRat}^{\uparrow}$ *source route into the selected finite-tail witness and the* $\mathsf{Real}^{\uparrow}$ *seal. Every seal-facing read passes through the precision row q, the window row s, the source row r, and the selected witness row w before reaching e, with transport, continuation, provenance, and local naming recorded by h, c, p, n.*

Proof. Start with the source coordinate of the carrier. The packet lists one $\mathsf{RegSeqRat}^{\uparrow}$ row, namely r, and no second regular-rational source route.

Apply window factorization and window lock. The selected finite-tail read is confined to s, r, w, while the precision request q fixes the finite observation demand answered by that tail.

Use the Real handoff theorem for the seal-facing step. It routes the consumer through q, s, r, w, h, c, p, n before e is consumed, so r is the only source route available to the seal. □

Theorem 559.9 (Finite-tail diagonal seal obligation package)**.** *The* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *obligation package over*

$$D = (q, s, r, w, e, h, c, p, n)$$

consists exactly of precision routing through q, selected window routing through s, $\mathsf{RegSeqRat}^{\uparrow}$ *source coverage through r, selected-tail witness exactness through w,* $\mathsf{Real}^{\uparrow}$ *seal handoff through e, componentwise transport through h, displayed route replay through c, provenance through p, and the local* NameCert *row n. This package has no quotient-stream, host-real-equality, hidden-limit, or ambient-completeness input.*

Proof. First assemble the displayed carrier rows. The carrier definition and NameCert obligation theorem give the precision, window, source, witness, seal, transport, route, provenance, and local naming coordinates.

Next add the source and witness controls. Selected-tail exactness binds w to the finite window s and source r, while RegSeqRat source coverage makes r the only source route into the witness and seal.

Finally apply Real handoff and non-escape. Real handoff places the seal row e downstream of the displayed finite chain, and non-escape rules out quotient streams, host real equality, unbounded diagonal streams, ambient completeness, and selected limits as additional inputs. □

Lean4 ✓ `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealCarrier obligation package`

Theorem 559.10 (FiniteTailDiagonalSeal obligation closure)**.** *For an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier* $D = (q, s, r, w, e, h, c, p, n)$*, the obligation surface closes over the same finite packet: carrier admission, precision routing, window routing, witness exactness, seal handoff, transport, route replay, provenance, and local naming are all read through* $q, s, r, w, e, h, c, p, n$*. The closure route is the one consumed by chapter 450, and its upstream rows are the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows displayed in the carrier.*

Proof. Assemble the carrier and local NameCert rows from Definition 559.1 and Definition 559.2. These rows provide carrier admission, precision routing, finite-window routing, source routing, selected-tail witness exactness, seal handoff, transport, route replay, provenance, and naming.

Use the exactness theorems to fix the route. Definition 559.3, Definition 559.4, Definition 559.7, and Definition 559.8 keep every witness read inside the selected $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ source row, and the precision route supplied through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows.

Finally apply the handoff and boundary theorems. Definition 559.5 sends the selected witness to the $\mathsf{Real}^{\uparrow}$ seal row, while Definition 559.6 blocks quotient streams, host real equality, unbounded diagonal data, ambient completeness, and choice-selected limits. □

Theorem 559.11 (FiniteTailDiagonalSeal RealCompleteness route)**.** *The route from* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *to chapter 450 passes through the same finite-tail seal rows* $q, s, r, w, e, h, c, p, n$*. A RealCompleteness consumer may read the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision route, and the* $\mathsf{Real}^{\uparrow}$ *seal row, but it has no route through quotient streams, ambient completeness, host real equality, or an unbounded diagonal object.*

Lean4 ✓ `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealUp StdBridge`

Proof. Start with Definition 559.10. It makes the finite-tail seal packet the entire obligation route available to downstream consumers.

Read the RealCompleteness side through chapter 450. Its finite diagonal-limit packet consumes StreamName windows, RegSeqRat readbacks, DyadicRatCore ledgers, and the final RealUp seal row, which are exactly the rows exposed by s, r, q, e together with the selected witness w.

The route is preserved by h, c, p, n. Componentwise transport, continuation replay, provenance, and local naming keep the consumer on the same finite seal route.

Apply Definition 559.6 for the negative boundary. Since the finite packet lacks quotient streams, ambient completeness, host equality, and unbounded diagonal data, the RealCompleteness route cannot acquire them. □

Theorem 559.12 (FiniteTailDiagonalSeal window obligation package)**.** *The window-specific obligation package inside an accepted* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *carrier isolates three rows: the* $\mathsf{StreamName}^{\uparrow}$ *window row* s*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* r*, and the selected finite-tail witness row* w*. Every precision request from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *side reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* e *only after these three rows have been read and transported through* h, c, p, n*.*

Proof. Project the window, source, and witness rows from the carrier. The displayed packet contains one selected window row s, one regular-rational source route r, and one selected finite-tail witness w.

Use Definition 559.4 and Definition 559.7. Together they make w readable only under s and r, with the precision request supplied by q.

Then use Definition 559.5. The seal row e is downstream of q, s, r, w and remains routed by h, c, p, n.

The non-escape theorem supplies the boundary: no quotient stream, host real equality, hidden limit, or ambient completeness row can enter the window package. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailDiagonalSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ packet is generated from a rational precision request, selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ source row, selected finite-tail witness, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTailDiagonalSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 559.1, Definition 559.9, and Definition 559.11 bind the public finite-tail seal surface over the obligation package and the RealCompleteness route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTailDiagonalSeal}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.FiniteTailDiagonalSealUp.FiniteTailDiagonalSealUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient streams, host real equality, unbounded diagonal streams, ambient completeness, or selected limits.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 560

RegularCauchyTailMeet NameCert construction

The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ interface packages the finite rows needed when several regular Cauchy consumers must share one dyadic tail threshold. It is a concrete naming certificate for the tail-meet packet used by diagonal real-completion routes; it does not assert quotient equality of streams, a selected limit, trichotomy, or an ambient completeness theorem.

The shared threshold route is the consumer side of the $\mathsf{CauchyModulusMeet}^{\uparrow}$ shared-bound export in chapter 462. It consumes exactly the two $\mathsf{StreamName}^{\uparrow}$ rows, the two source $\mathsf{CauchyModulus}^{\uparrow}$ rows, the shared meet row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transports, displayed Cont routes, package provenance, and the local $\mathsf{NameCert}$ row.

Definition 560.1 (RegularCauchyTailMeet carrier)**.** *A* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N)$$

with the following displayed coordinates:

1. R_0 *and* R_1 *are the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows whose tails must be read through one threshold;*
2. W_0 *and* W_1 *are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules exposing the two rational tails;*
3. M_0 *and* M_1 *are the source* $\mathsf{CauchyModulus}^{\uparrow}$ *or dyadic-radius rows attached to the two regular Cauchy names;*
4. τ *is the shared tail threshold row, displayed as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger sufficient for both source rows;*
5. Q *is the paired tail comparison row recording the two scheduled observations at the shared threshold;*
6. H *stores componentwise* hsame *transports for the two source names, windows, source moduli, shared threshold, paired comparison, provenance, and local name;*
7. C *stores the displayed* Cont *routes for reading source windows, source modulus rows, the shared threshold, the paired tail comparison, and the real-facing handoff;*
8. L *is the finite ledger over* $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMeet}^{\uparrow}}$ *row.*
The carrier is only the finite shared-tail packet. It has no coordinate for quotient streams, host real equality, selected limits, trichotomy, or ambient completeness.

Theorem 560.2 (RegularCauchyTailMeet NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMeet}^{\uparrow}}$ *surface for the carrier of Definition 560.1 consists exactly of carrier admission, paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source admission, paired* $\mathsf{StreamName}^{\uparrow}$ *scheduling, source modulus readback, shared dyadic tail-threshold exposure, paired tail comparison, componentwise* hsame *transport, displayed* Cont *routes, finite ledger exactness, package provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet T. Its coordinates list the two regular Cauchy sources, their windows, their source modulus rows, the shared tail threshold, the comparison row, transport, continuation, ledger, and local name.

The shared threshold is finite. The two schedules expose the rational tail windows before τ and Q can be read, and the source modulus rows certify that the same tolerance ledger is sufficient for both sides.

The remaining obligations are provenance and routing obligations. The rows H, C, L, N transport the displayed coordinates, route the reads, record package provenance, and name the packet; none of them introduces quotient equality, a selected limit, trichotomy, host real equality, or completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket namecert obligations`

Theorem 560.3 (RegularCauchyTailMeet classifier stability)**.** *For accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carriers whose displayed coordinates are related componentwise by the stored* hsame *rows, the paired tail comparison row Q remains readable through the same shared threshold τ. The classifier is stable under transport of the two source names, two window schedules, two source modulus rows, the threshold ledger, and the local certificate rows.*

Proof. Project the two accepted carriers. The only rows that determine the tail classifier are the two source names, their scheduled windows, their source modulus rows, the shared threshold, and the comparison row.

Use the componentwise transport entries in H to move these rows to a common displayed surface. The continuation rows in C then replay the reads from sources to windows, from windows and source bounds to τ, and from τ to Q.

Since the carrier contains no independent classifier coordinate, transported reads must use the same threshold ledger and paired comparison route. Stability follows from the displayed packet itself, not from host real equality or quotient stream equality. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket classifier stability`

Theorem 560.4 (RegularCauchyTailMeet Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *diagonal or completion consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier reads tail synchronization only through the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *schedules, source modulus rows, the shared dyadic threshold τ, the paired comparison row Q, displayed* hsame *transports,* Cont *routes, finite ledger, package provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMeet}^{\uparrow}}$ *row.*

Proof. Start with a real-facing consumer route. The carrier offers no real-facing data except the handoff reached through the displayed schedules, source modulus rows, and shared threshold ledger.

The consumer first reads the paired windows and source bounds. It then reads the common threshold τ and the comparison row Q, with all movement recorded by H and C.

The ledger L and local certificate N record provenance and name the handoff. Because no other coordinate is present, the seal exports tail synchronization without exporting quotient streams, selected limits, trichotomy, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket real seal handoff`

Theorem 560.5 (RegularCauchyTailMeet dyadic absolute-value handoff)**.** *Let an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier have shared threshold row* τ*. Suppose the displayed row* τ *is supplied by an accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint ledger whose nonnegative dyadic endpoint and tolerance ledger are used as the common tail bound. Then the two scheduled regular-Cauchy tails consume only that displayed nonnegative dyadic endpoint, its tolerance ledger, the componentwise* hsame *transports, the displayed* Cont *routes, the finite ledgers, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMeet}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{DyadicAbsoluteValue}^{\uparrow}}$ *rows.*

The handoff exports no quotient streams, selected limits, trichotomy witness, host real equality, ambient completeness, quotient rational, selected real value, or host integer normal form.

Proof. Begin with the $\mathsf{Real}^{\uparrow}$ seal handoff of Definition 560.4. It forces every completion-facing read of the shared tail packet through the paired source rows, paired window schedules, source modulus rows, the shared threshold τ, the comparison row Q, the stored transports, continuations, ledger, package provenance, and local name.

Now read τ through the accepted $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ endpoint ledger. By Definition 293.4, the endpoint offered to $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers is the displayed nonnegative dyadic endpoint, reached through the endpoint row, tolerance ledger, transports, continuation routes, package provenance, and local dyadic absolute-value name.

Apply Definition 293.5 to the endpoint source. The finite ledger admits no hidden quotient rational, selected real value, trichotomy proof, host integer normal form, or real-completeness witness. Substituting that endpoint ledger for the shared threshold route leaves the two scheduled tails with exactly the listed finite rows.

Thus the regular-Cauchy tail meet consumes the dyadic absolute-value endpoint as a finite threshold ledger. The route remains a composite of displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so no quotient stream, selected limit, trichotomy, host real equality, or ambient completeness coordinate enters the handoff. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket dyadic absolute value handoff`

Theorem 560.6 (RegularCauchyTailMeet dyadic endpoint meet)**.** *Let an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier have source rows* R_0, R_1*, schedules* W_0, W_1*, source modulus rows* M_0, M_1*, shared threshold coordinate* τ*, and paired comparison row* Q*. Suppose the two scheduled tails receive displayed nonnegative endpoint and tolerance ledgers from accepted* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *packets. If a displayed dyadic row* τ' *is strong enough for both endpoint tolerance ledgers through the same source modulus rows* M_0 *and* M_1*, then* τ' *is a valid shared threshold for the paired comparison row* Q *and for the* $\mathsf{Real}^{\uparrow}$ *seal handoff of the accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet.*

The meet route uses only the two $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint rows, their tolerance ledgers, the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, the schedules, the source modulus rows, the displayed threshold row* τ'*, the paired comparison row* Q*, the componentwise* hsame *transports, displayed*

Cont *routes, finite ledgers, package provenance, and the local* NameCert *rows. It exports no quotient stream, selected limit, trichotomy, host real equality, host integer normal form, or ambient completeness witness.*

Proof. First read the two endpoint packets through the dyadic absolute-value bridges. Apply Definition 293.4 to obtain displayed nonnegative dyadic endpoints for the two scheduled tails, and apply Definition 293.5 to keep each endpoint route inside its finite endpoint row, tolerance ledger, transports, continuations, provenance, and local dyadic name.

Next pass each endpoint tolerance ledger to the regular-Cauchy tail surface. Apply Definition 293.8 to make the displayed tolerance rows available to the source modulus reads, and apply Definition 560.7 to identify the only regular-Cauchy rows that can consume them: $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H,$

The displayed row τ' is assumed strong enough for both endpoint tolerance ledgers through the same source modulus rows. Apply Definition 560.8 with τ' in place of the carrier threshold route. The paired comparison row Q and the $\mathsf{Real}^{\uparrow}$ handoff therefore factor through τ' using the already displayed transports, continuations, finite ledger, package provenance, and local name.

All objects used in the construction are named rows in the accepted packets. Since the dyadic endpoint bridges and the tail-meet ledger exactness have no coordinate for quotient streams, selected limits, trichotomy, host real equality, host integer normal forms, or ambient completeness, the finite meet cannot export any such coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket dyadic endpoint meet`

Theorem 560.7 (RegularCauchyTailMeet ledger exactness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier, the finite ledger is exact for the route from paired regular Cauchy sources through their windows and source modulus rows to the shared dyadic tail threshold* τ*, the comparison row* Q*, and the* $\mathsf{Real}^{\uparrow}$ *handoff. No ledger entry records an independent tail oracle, quotient stream equality, selected limit, trichotomy row, host real equality, or ambient completeness witness.*

Proof. Inspect the carrier of Definition 560.1. The only tail-threshold data are the two source rows, two scheduled windows, two source modulus rows, the shared threshold row, and the paired comparison row.

Apply the NameCert obligations to bind every accepted read to those rows. Classifier stability then shows that transport cannot replace the shared threshold or comparison row with an external tail oracle.

Finally apply the Real seal handoff. Every real-facing use of the packet factors through the same finite ledger, so the ledger is exact for the displayed route and has no unlisted analytic coordinate. □

Theorem 560.8 (RegularCauchyTailMeet shared-threshold cofinality)**.** *Let an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *carrier have shared threshold row* τ*. If a displayed dyadic threshold row* τ' *is at least as strong as* τ *for both source modulus rows and is transported through the same paired window schedules, then the paired comparison row* Q *and the* $\mathsf{Real}^{\uparrow}$ *handoff factor through* τ' *using the existing* hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMeet}^{\uparrow}}$ *row.*

This strengthening exports no independent tail oracle, quotient stream, selected limit, host real equality, trichotomy row, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket shared threshold cofinality`

Proof. Project the accepted carrier. The two source modulus rows M_0 and M_1 are the only rows that certify threshold sufficiency for the paired tails, and the paired schedules W_0 and W_1 are the only rows that expose the corresponding observations.

Read the displayed strengthening τ' through those same source modulus rows. The hypothesis that τ' is at least as strong as τ for both sources lets each scheduled tail read use τ' without changing the source rows or window schedules.

Replay the paired schedules through the displayed continuation routes. Transport the comparison row through the componentwise hsame entries already stored in H, so the comparison remains the same carrier-facing row Q with the threshold coordinate refined to τ'.

Finally apply ledger exactness and the Real seal handoff. The finite ledger and seal route already contain the required transports, continuations, provenance, and local name, so the strengthened threshold factors through the same public handoff and adds no unlisted analytic coordinate. □

Theorem 560.9 (RegularCauchyTailMeet obligation closure package)**.** *The carrier,* NameCert *obligation surface, classifier stability,* Real$^{\uparrow}$ *seal handoff, dyadic absolute-value endpoint handoff, dyadic endpoint meet, ledger exactness, and shared-threshold cofinality jointly form the complete obligation package for the shared regular-Cauchy tail threshold packet over* RegularCauchyName$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* CauchyModulus$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* DyadicAbsoluteValue$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The package exports no quotient stream, selected limit, trichotomy coordinate, host equality row, or ambient completeness witness.

Proof. Begin with the finite carrier of Definition 560.1. It displays the paired regular Cauchy sources, paired stream schedules, source modulus rows, shared threshold ledger, paired comparison row, transports, continuations, ledger, package provenance, and local name.

Add the local obligation surface from Definition 560.2. Those obligations certify carrier admission, scheduled source reads, modulus readback, shared-threshold exposure, paired comparison, transport, continuation, ledger exactness, provenance, and naming.

Next use Definition 560.3 and Definition 560.4. They keep the classifier and the real-facing handoff on the same displayed shared threshold and comparison rows.

The dyadic endpoint route is controlled by Definition 560.5. It identifies the shared threshold consumed by the two tails with the displayed nonnegative endpoint and tolerance ledger supplied by the accepted DyadicAbsoluteValue$^{\uparrow}$ packet.

The two-endpoint meet is controlled by Definition 560.6. When two endpoint tolerance ledgers must be served by one displayed dyadic threshold row, the same source modulus rows and schedules route the paired comparison and the real-facing handoff through that shared row.

Finally apply Definition 560.7. The finite ledger exhausts the route from source rows to shared threshold, comparison, and Real$^{\uparrow}$ handoff, so the obligation package has no quotient stream, selected limit, trichotomy, host equality, or ambient completeness coordinate.

Shared-threshold cofinality from Definition 560.8 adds the permitted refinement step. A stronger displayed dyadic threshold can be routed through the same source modulus rows, schedules, transports, continuations, ledger, provenance, and local name without adding an independent tail oracle. □

Theorem 560.10 (RegularCauchyTailMeet diagonal-window threshold handoff)**.** *Let an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet supply its shared threshold row τ as the selected-window threshold for an accepted* RealDiagonalWindowCoverage$^{\uparrow}$ *packet. Then the paired regular-Cauchy tails route through the same finite threshold τ before the* Real$^{\uparrow}$ *seal of the diagonal-window packet is read.*

The consumer-facing route uses only the paired source rows, scheduled windows, source modulus rows, shared threshold, paired comparison row, componentwise hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, and the local* NameCert *rows. It exports no independent tail oracle, quotient stream, selected limit, host real equality, trichotomy row, or ambient completeness witness.*

Proof. First apply Definition 560.8. It identifies the shared threshold row consumed by both source tails and keeps the paired comparison and real-facing handoff on the same finite threshold route.

Apply Definition 560.9 to expose the complete shared-tail surface: paired sources, windows, source modulus rows, the threshold ledger, comparison row, transports, continuations, ledger, provenance, and local name.

Finally use Definition 561.8. The diagonal-window packet can read its $\mathsf{Real}^{\uparrow}$ seal only after the selected window and dyadic observation ledger have been displayed, so the shared threshold supplied by the tail-meet packet is consumed before the seal-facing row is available. □

Theorem 560.11 (RegularCauchyTailMeet public shared-threshold certificate)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the obligation closure package, shared-threshold cofinality theorem, dyadic endpoint meet,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and diagonal-window threshold handoff assemble one public shared-threshold certificate consumed by* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *and by* $\mathsf{Real}^{\uparrow}$*-facing diagonal routes.*

The certificate reads only the paired $\mathsf{RegularCauchyName}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *schedules, source modulus rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, paired comparison row, componentwise* hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, selected diagonal-window data, and local* NameCert *rows. It exports no quotient stream, selected limit, trichotomy, host equality, hidden tail oracle, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket public shared threshold certificate`

Proof. First collect Definition 560.9. It binds the carrier rows, schedules, source modulus reads, threshold ledger, paired comparison row, transports, continuations, ledger, package provenance, and local name into one obligation surface.

Use Definition 560.8. Cofinal refinement keeps both source tails on the same displayed threshold row and does not introduce an independent tail oracle.

Next apply Definition 560.6 together with Definition 560.4. The dyadic endpoint meet supplies the shared endpoint threshold, and the real-seal handoff keeps the consumer on that finite threshold route.

Finally use Definition 560.10. The selected-window consumer reads the shared threshold before the $\mathsf{Real}^{\uparrow}$ seal route is available, so the assembled rows form the public certificate for $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers while preserving all listed exclusions. □

Theorem 560.12 (RegularCauchyTailMeet limit-seal threshold route)**.** *Let* $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H,$ *be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer or diagonal* $\mathsf{Real}^{\uparrow}$ *consumer that reads the public shared threshold of* T *factors through the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *schedules, source modulus rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, paired comparison row, visible* BHist *evidence, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The route consumed by chapter 386 and by diagonal $\mathsf{Real}^{\uparrow}$ *readers is therefore the same finite TailMeet certificate route, not a selected limit, quotient stream, host real equality, hidden tail oracle, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket limit seal threshold route`

Proof. First read the accepted carrier through the obligation package. It exposes the paired source rows, scheduled windows, source modulus rows, shared threshold ledger, paired comparison row, componentwise transports, continuations, finite ledger, provenance, and local name as one finite TailMeet surface.

Next compose shared-threshold cofinality with the dyadic endpoint meet. Cofinal refinement keeps stronger precision requests on the displayed source modulus rows, while the endpoint meet supplies the threshold ledger row consumed by the real-facing route.

Use the real-seal handoff, the diagonal-window threshold handoff, and the public shared-threshold certificate. These results make the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers read the threshold only after the same finite route has been displayed.

The exclusion boundary is inherited from consumer exactness. Since the consumer receives only the listed TailMeet rows, there is no coordinate from which to export a quotient stream, selected limit, host real equality, hidden tail oracle, or ambient completeness witness. □

Theorem 560.13 (RegularCauchyTailMeet observation-budget absorption)**.** *Every* $\mathsf{RealObservationBudget}^{\uparrow}$ *request that reads the public shared threshold of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is absorbed by the same finite TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The observation route of chapter 611 may inspect its own finite request, window, dyadic readback, regular-rational handoff, and seal rows only through that displayed TailMeet entrance. It cannot add a hidden observation budget, host real equality, selected limit, or ambient completeness row to the threshold read.

Proof. Begin with the public shared-threshold certificate. It names τ as the threshold exported to real-facing consumers and keeps the export attached to the paired regular-Cauchy sources, stream schedules, source modulus rows, comparison row, transports, continuations, ledger, provenance, and local name.

Now route the request through the finite observation-budget surface. That sibling chapter admits observation reads through finite request rows, windows, dyadic readbacks, regular-rational handoff rows, and a real-seal boundary, so the TailMeet threshold can enter it only through the displayed finite entrance.

Finally apply the real-seal handoff and consumer exactness. The handoff places the budget read behind the same TailMeet threshold route, and exactness prevents the budget side from appending a host equality row, selected limit, hidden observation budget, or ambient completeness premise. □

Theorem 560.14 (RegularCauchyTailMeet selector-budget public route)**.** *Every* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *route that shares the public threshold of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet reuses the displayed selector-window and* $\mathsf{RegSeqRat}^{\uparrow}$ *route of chapter 678. Its tail-facing threshold entry is the TailMeet row* τ *transported through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

and it creates no second selector, hidden threshold, hidden tail oracle, quotient stream, host real equality, selected limit, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket selector budget public route`

Proof. Start with selector admission and the selector-budget surface. The admitted route supplies the finite selector windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before any tail-facing threshold is consumed.

Next use the shared-threshold cofinality and route determinacy rows. Cofinality allows stronger displayed thresholds only through M_0 and M_1, and determinacy keeps any shared-threshold read with the same transported source, window, and modulus rows on the same τ.

Attach the public shared-threshold certificate and the diagonal-window threshold handoff. They present the selector-facing threshold as the same public TailMeet row consumed by real-facing routes, with Q, H, C, L, N carried along the displayed package and naming surface.

Thus the selector-budget consumer reuses the existing selector-window and $\mathsf{RegSeqRat}^{\uparrow}$ route. With no second threshold coordinate and no extra selector row in the TailMeet carrier, a hidden tail oracle or ambient completion datum cannot be introduced. □

Theorem 560.15 (RegularCauchyTailMeet triple consumer route factorization)**.** *Any accepted consumer among* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *and* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *that reads a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold factors through the same finite rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

No such consumer may append selected limits, quotient stream equality, host real equality, hidden observation budgets, hidden tail oracles, or ambient completeness rows to that threshold read.

Proof. First use the limit-seal threshold route. It sends every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ threshold read through the paired regular-Cauchy sources, stream schedules, modulus rows, threshold ledger, comparison row, transports, continuations, package, and naming rows.

Next use observation-budget absorption. A $\mathsf{RealObservationBudget}^{\uparrow}$ request may inspect its own finite budget surface only after the same TailMeet entrance has been displayed, so it receives τ with the rows $R_0, R_1, W_0, W_1, M_0, M_1, Q, H, C, L, N$ attached.

Finally use the selector-budget public route. The selector-facing threshold is the same public TailMeet row, transported through the same finite packet and not through a second selector or hidden threshold coordinate.

The three routes therefore share the displayed row list. Since each preceding theorem excludes selected limits, quotient stream equality, host real equality, hidden observation budgets, hidden tail oracles, and ambient completeness rows, their common factorization excludes the same extra coordinates. □

Theorem 560.16 (RegularCauchyTailMeet Real and completion consumer frontier)**.** *Let* $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N)$ *be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads the shared threshold of* T *through* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *or* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *must first factor through the single finite TailMeet packet*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

No such consumer may bypass that frontier by using selected limits, quotient stream equality, host real equality, trichotomy rows, hidden observation budgets, hidden tail oracles, or ambient completeness.

Proof. Use the limit-seal threshold route to place every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ read behind the paired source rows, stream schedules, modulus rows, shared threshold ledger, comparison row, transports, continuations, package provenance, and local name.

Next use observation-budget absorption and the selector-budget public route. These results show that the observation and selector surfaces can consume τ only through the same displayed rows; their own budget, selector, window, and readback coordinates do not create another entrance to the threshold.

Apply the triple consumer route factorization. It already identifies the common route for the three downstream consumer surfaces, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer meeting any of them receives the same finite TailMeet packet before any seal-facing or completion-facing read.

The refused objects are not coordinates of that packet. Selected limits, quotient stream equality, host real equality, trichotomy rows, hidden observation budgets, hidden tail oracles, and ambient completeness therefore cannot be exported through the common frontier. □

Theorem 560.17 (RegularCauchyTailMeet Real-completion formal frontier)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau,$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *formal target that consumes the shared threshold of T through the real-completion route must use the same finite TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The handoff exposes the threshold only through the paired regular-Cauchy sources, stream schedules, source modulus rows, public dyadic threshold ledger, comparison row, BHist *evidence, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. It does not permit selected limits, quotient stream equality, host real equality, trichotomy, hidden observation budgets, hidden tail oracles, or ambient completeness.*

Proof. Begin with the limit-seal threshold route. It places every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ threshold read behind the accepted TailMeet packet and keeps the threshold attached to the displayed source, schedule, modulus, comparison, transport, continuation, provenance, and naming rows.

Next apply observation-budget absorption and the selector-budget public route. These results show that budget-facing and selector-facing consumers receive the same τ only after the finite TailMeet entrance has been displayed.

Use the triple consumer route factorization to align the three downstream consumer surfaces. The factorization gives a single row list for limit-seal, observation-budget, and selector-budget reads, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ target using any of them cannot select a different threshold route.

Finally apply the Real and completion consumer frontier. The excluded objects are outside the shared packet; therefore the formal handoff for later $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ work is exactly the finite TailMeet route and not an ambient real, quotient, trichotomy, oracle, or completeness assumption. □

Theorem 560.18 (RegularCauchyTailMeet completion-uniform frontier)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q,$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every* $\mathsf{Completion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer that uses the shared threshold of T must first factor through the common finite TailMeet entrance*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

This entrance is read before any completion seal, real seal, tail row, or uniform-family row is consumed. No such consumer may replace it by selected limits, quotient streams, host real equality, hidden tail oracles, or ambient completeness.

Proof. Start with the limit-seal threshold route. It exposes the shared threshold only through the paired regular-Cauchy source rows, stream schedules, source modulus rows, dyadic threshold ledger, comparison row, transports, continuations, package provenance, and local name.

Next apply the triple consumer route factorization. It aligns the limit-seal, observation-budget, and selector-budget consumers on the displayed TailMeet rows and removes selected limits, quotient stream equality, host real equality, hidden budget rows, hidden tail oracles, and ambient completeness from their threshold reads.

Use the Real-completion formal frontier to pass from those consumer surfaces to later $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ targets. The formal handoff keeps the same row list as the only threshold entrance and does not add trichotomy, quotient, host-equality, oracle, or completeness coordinates.

Finally attach the UniformCauchyCriterion finite-family seal obligation package. Its finite-family surface may read a regular-rational tail or a real seal only through its displayed finite seal route, so when that surface uses the TailMeet shared threshold it receives the same TailMeet rows before any uniform-family row is inspected.

Thus completion-facing, real-facing, and uniform-Cauchy-facing consumers share the displayed finite threshold entrance. Since each component route excludes non-displayed limits, quotient streams, host equality, hidden tail data, and ambient completeness, the common frontier excludes them as well. □

Theorem 560.19 (RegularCauchyTailMeet selector-window-seal consumer totality)**.** *Let* $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N)$ *be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every consumer route that reaches the shared threshold through* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*, or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *factors through exactly the displayed TailMeet row list*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

before any real, completion, or uniform-family read. The route is the single selector-window-seal packet determined by Definition 560.12, Definition 560.13, Definition 560.14, Definition 560.15, and Definition 560.18; it admits no selected limit, quotient stream, host real equality, trichotomy row, hidden tail oracle, or ambient completeness coordinate.

Proof. Project the accepted packet T. Its visible rows are the paired $\mathsf{RegularCauchyName}^{\uparrow}$ source certificates, paired $\mathsf{StreamName}^{\uparrow}$ schedules, source modulus rows, shared dyadic threshold ledger, comparison row, BHist evidence, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row.

First use the limit-seal threshold route, the observation-budget absorption route, and the selector-budget public route. These three component results place the $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, and $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ entrances on the same TailMeet threshold row τ, always with the surrounding rows $R_0, R_1, W_0, W_1, M_0, M_1, Q, H, C, L, N$ displayed.

Next apply the triple consumer route factorization. It aligns the three consumer entrances as one finite selector-window-seal route, so a downstream real or completion reader cannot choose a second threshold path after entering through any one of them.

Finally use the completion-uniform frontier. That result places $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumers on the same entrance before any uniform-family row is inspected, and it keeps completion-facing and real-facing reads behind the displayed packet. Since none of the displayed coordinates is a selected limit, quotient stream, host real equality, trichotomy row, hidden tail oracle, or ambient completeness witness, no such object can enter the consumer route. □

Theorem 560.20 (RegularCauchyTailMeet finite-family threshold absorption)**.** *Let a finite family tail ledger from* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *schedule component tails whose displayed component thresholds share the threshold row supplied by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Then the finite-family threshold request factors through the tail-meet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The absorbed route is the same shared-threshold row consumed by $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, and it introduces no independent tail oracle, quotient stream, selected limit, host equality, trichotomy coordinate, or ambient completeness witness.*

Proof. Begin with Definition 560.26 and Definition 560.27. They force every tail-meet threshold read through the displayed source, window, and source-modulus rows before it reaches τ.

Next apply Definition 435.17. The finite-family ledger has no hidden component window outside the scheduled family rows, so a family threshold request whose component tails share the displayed threshold cannot bypass the listed tail-meet route.

Finally use Definition 435.10 together with Definition 560.33. The two transport rows identify the family-facing threshold and the tail-meet threshold as the same transported row, with the exclusions inherited from ledger exactness. □

Theorem 560.21 (RegularCauchyTailMeet limit-seal threshold handoff)**.** *Suppose the public shared-threshold certificate of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is consumed by a finite-tail diagonal seal route. Then the* $\mathsf{Real}^{\uparrow}$ *seal read passes through the displayed threshold row* τ *before any seal row is available.*

The handoff uses only the paired regular-Cauchy source rows, scheduled windows, source modulus rows, shared threshold, comparison row, componentwise hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, the finite-tail selected witness row, and the local* $\mathsf{NameCert}$ *rows.*

Proof. First read the shared row through Definition 560.11. The certificate exposes τ as the public threshold consumed by diagonal $\mathsf{Real}^{\uparrow}$ routes while retaining the paired source, window, modulus, comparison, transport, continuation, ledger, package, and naming rows.

Then apply Definition 559.5. The finite-tail diagonal seal can read its $\mathsf{Real}^{\uparrow}$ row only after the precision request, selected window, regular sequence row, and selected finite-tail witness have been fixed.

Combining the two routes places the shared threshold before the seal read. Since neither packet carries quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness, the handoff exports none of those rows. □

Theorem 560.22 (RegularCauchyTailMeet uniform-criterion threshold export)**.** *For every accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet used by* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *and by a* $\mathsf{Real}^{\uparrow}$*-facing diagonal consumer, the exported shared-threshold row is the same row consumed by the finite-family criterion and by the diagonal seal route.*

The export is exactly the row τ *routed through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

and it exports no quotient stream, selected limit, trichotomy coordinate, host equality row, or ambient completeness witness.

Proof. Use Definition 560.20 to identify the finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ threshold request with the displayed tail-meet route through τ.

Use Definition 560.21 to identify the diagonal $\mathsf{Real}^{\uparrow}$ consumer route with the same public shared-threshold certificate before any seal row is read.

Finally apply Definition 560.34, Definition 560.35, and Definition 560.7. These rows exhaust the consumer-facing ledger and block every excluded quotient, limit, trichotomy, host-equality, and completeness coordinate, so the export has exactly the shared threshold route stated above. □

Theorem 560.23 (RegularCauchyTailMeet finite-family threshold coverage)**.** *Every finite-family threshold request routed through* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *and accepted by a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet factors through the displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The coverage is a finite BHist *coverage statement over the paired regular-Cauchy sources, paired stream windows, source modulus rows, shared dyadic threshold, comparison row, componentwise* hsame *transports, displayed* Cont *routes, ledger, and local* $\mathsf{NameCert}$ *row.*

Proof. Begin with Definition 560.22. It identifies the exported row τ as the row shared by the finite-family criterion and the diagonal-facing consumer.

Apply Definition 560.7. The ledger admits only the displayed route from the two source rows through the two windows and source modulus rows to τ and Q.

Finally use Definition 560.11. The public certificate packages exactly the same threshold route for consumers, so the finite-family request is covered by the listed rows and no hidden tail source is available. □

Theorem 560.24 (RegularCauchyTailMeet root-unblock handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *diagonal consumer admitted by the root surface of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet receives only the shared threshold row and comparison rows already present in that packet. The handoff does not add a selected limit, quotient stream equality, host real equality, trichotomy row, or ambient completeness witness.*

Proof. First apply Definition 560.8. It keeps any stronger diagonal request on the same displayed shared threshold route.

Next apply Definition 560.11. The consumer-facing package exposes the threshold, comparison row, transports, continuations, ledger, package provenance, and local name already present in the accepted carrier.

The non-escape part is inherited from Definition 560.22. Since that export has no quotient, selected-limit, trichotomy, host-equality, or completeness coordinate, the root handoff supplies only the admitted shared rows. □

Theorem 560.25 (RegularCauchyTailMeet diagonal budget consumer)**.** *Any diagonal budget consumer using an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet consumes the same finite dyadic threshold ledger*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and cannot obtain a selected limit, trichotomy, quotient stream, or completeness witness from that packet.

Proof. Read the budget request through Definition 560.23. The consumer must pass through the displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, and naming rows.

Use Definition 560.24. The downstream $\mathsf{Real}^{\uparrow}$ diagonal route receives the same shared threshold and comparison data and no additional consumer-specific evidence.

Finally apply Definition 560.7 together with Definition 560.11. The exact ledger and public certificate exhaust the finite dyadic budget surface, so selected limits, trichotomy, quotient streams, and completeness witnesses remain outside the packet. □

Theorem 560.26 (RegularCauchyTailMeet window-source coverage)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet with paired source rows*

$$R_0,\ R_1,$$

paired scheduled windows

$$W_0,\ W_1,$$

and source modulus rows

$$M_0,\ M_1,$$

every displayed threshold read for the shared tail factors through the finite route

$$R_0, R_1 \longrightarrow W_0, W_1 \longrightarrow M_0, M_1 \longrightarrow \tau.$$

The source rows are $\mathsf{RegularCauchyName}^{\uparrow}$ *rows, the windows are* $\mathsf{StreamName}^{\uparrow}$ *rows, the modulus reads are* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, and the produced threshold* τ *is the same dyadic threshold consumed by the public shared-threshold certificate.*

Proof. Read the carrier coordinates from Definition 560.1. The packet contains the paired regular Cauchy sources, the paired stream schedules, and the paired modulus rows as displayed finite coordinates.

Apply the obligation surface of Definition 560.2. It admits only source reads, scheduled-window reads, and modulus readback through the local $\mathsf{NameCert}$ rows.

Use Definition 560.7. The ledger routes the source and window data to the shared threshold row, so the threshold coordinate τ cannot be read without passing through $R_0, R_1, W_0, W_1, M_0, M_1$. □

Theorem 560.27 (RegularCauchyTailMeet source-modulus exhaustion)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet with displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

every threshold read factors through exactly the source-window-modulus route

$$R_0, R_1 \longrightarrow W_0, W_1 \longrightarrow M_0, M_1 \longrightarrow \tau.$$

No consumer can read the shared threshold directly from Q*, from a* $\mathsf{Real}^{\uparrow}$ *seal, from a dyadic endpoint row, or from an unlisted tail oracle.*

Proof. Start with the carrier of Definition 560.1. Its only threshold-facing source rows are the paired regular Cauchy names, paired stream windows, paired source modulus rows, and the shared dyadic threshold row.

Apply the local obligations of Definition 560.2. They allow source admission, scheduled-window reads, modulus readback, and shared-threshold exposure only through the displayed packet coordinates.

Use window-source coverage from Definition 560.26. It identifies the mandatory route from R_0, R_1 through W_0, W_1 and M_0, M_1 to τ.

Finally apply ledger exactness and shared-threshold transport, Definition 560.7 and Definition 560.33. The finite ledger records no alternate source for τ, and transport preserves the same displayed threshold row. □

Theorem 560.28 (RegularCauchyTailMeet threshold transport coverage)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q,$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every transport of the shared threshold row τ to a root-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer is covered by the displayed componentwise* hsame *row H, the displayed* Cont *route C, the finite ledger L, and the local* NameCert *row N, after the source rows R_0, R_1, window rows W_0, W_1, modulus rows M_0, M_1, and comparison row Q have been read.*

No accepted transport may replace τ by an unlisted tail threshold, quotient-stream equality, selected limit, host real equality, or ambient completeness witness.

Proof. Begin with the carrier of Definition 560.1. It displays exactly the two source rows, two window rows, two modulus rows, shared threshold row, comparison row, transport row, continuation row, ledger row, and local naming row.

Use the NameCert obligations of Definition 560.2. They force every accepted threshold read to pass through $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$ rather than through an external threshold oracle.

Apply classifier stability and ledger exactness. Classifier stability keeps the transported comparison attached to the same shared-threshold coordinate, while ledger exactness rules out an additional threshold entry in L.

Finally use shared-threshold cofinality and the Real seal handoff. They let stronger displayed threshold rows feed the same consumer route, but only through the already named hsame, Cont, Pkg, and NameCert surface. □

Theorem 560.29 (RegularCauchyTailMeet window-pair exhaustion)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. The paired window rows W_0 and W_1 exhaust the window evidence for the shared-tail comparison: every accepted comparison read of Q and every downstream handoff to* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *must factor through both displayed schedules, their source rows R_0, R_1, their source modulus rows M_0, M_1, the shared threshold τ, and the structural rows H, C, L, N.*

The packet has no coordinate for a third scheduled window, an independent tail source, a hidden selector witness, or a host-level stream comparison.

Proof. Read the accepted packet by projection. The carrier exposes two and only two window schedules, namely W_0 and W_1, together with their source rows and modulus rows.

Apply the NameCert obligations and paired-window ledger coverage. The local certificate requires scheduled-window readback before the shared threshold and comparison row can be consumed, and the ledger records both windows before any real-facing handoff is available.

Use source-modulus exhaustion to identify the route to τ. Every threshold read factors through R_0, R_1, then W_0, W_1, then M_0, M_1, so the comparison row Q cannot be read from a smaller or different window family.

Finally apply selector admission and Real seal handoff. Both downstream consumers inherit the same two-window route and cannot add a third window, selected tail, quotient stream, or host comparison outside the displayed packet. □

Theorem 560.30 (RegularCauchyTailMeet Real seal consumer coverage)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q,$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal consumer, including diagonal and limit-seal consumers, is covered by the same finite handoff route through the paired sources R_0, R_1, paired windows W_0, W_1, source modulus rows M_0, M_1, shared threshold τ, comparison row Q, componentwise transports H, continuations C, ledger L, and local name N.*

Thus Real-facing consumption of the packet never escapes to quotient stream equality, selected limits, independent Cauchy moduli, hidden completeness, host real equality, or an unlisted seal row.

Proof. Start with the Real seal handoff theorem. It presents the completion-facing route as a read through the paired source rows, paired schedules, source modulus rows, shared threshold row, comparison row, transports, continuations, ledger, provenance, and local name.

Use ledger exactness and shared-threshold non-escape. They confine every seal-facing read to $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$ and exclude quotient streams, selected limits, hidden tail oracles, and ambient completeness data.

Apply seal compatibility for $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers. The seal receives the same shared threshold and comparison data already carried by the finite packet.

Finally collect threshold transport coverage and window-pair exhaustion. These two obligations show that the transport of τ and the paired-window evidence are already present before the Real seal route is read, so no independent modulus, host equality, or unlisted seal row can enter. □

Theorem 560.31 (RegularCauchyTailMeet threshold-witness determinacy)**.** *Let two accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *presentations have componentwise* hsame *rows on their paired source rows, paired window rows, source modulus rows, shared threshold rows, comparison rows, ledger rows, package rows, and local naming rows. If both presentations transport the source-window-modulus route to the same public shared-threshold certificate, then they determine the same public threshold row under those componentwise transports.*

Proof. Project both accepted presentations through Definition 560.1. Each presentation exposes its threshold only through the displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, provenance, and local naming rows.

Apply source-modulus exhaustion from Definition 560.27. Any threshold witness in either presentation must pass through the same kind of route from paired sources to paired windows, then through paired source modulus rows, and finally to the shared threshold row.

Now use the shared-threshold transport theorem Definition 560.33. The componentwise hsame rows move the source, window, modulus, threshold, comparison, ledger, package, and local naming rows together, so the transported threshold row remains the public row read by the certificate.

The NameCert obligations and ledger exactness rule out a second threshold coordinate. Therefore two accepted transported presentations determine the same public threshold row, with no appeal to quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 560.32 (RegularCauchyTailMeet paired-window ledger coverage)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the finite ledger accounts for both scheduled windows W_0 and W_1 before any* $\mathsf{Real}^{\uparrow}$ *seal handoff is available. The ledger must record the paired source rows, paired window schedules, source modulus rows, shared threshold row, paired comparison row, componentwise transports, displayed continuations, package provenance, and local naming row before a seal-facing consumer can read the shared-tail certificate.*

Proof. Begin with Definition 560.26. It requires both scheduled windows to appear on the route from the paired regular Cauchy sources to the common threshold.

Apply Definition 560.7. The ledger is exact for the route from source rows through the two scheduled windows and source modulus rows to τ, Q, and the $\mathsf{Real}^{\uparrow}$ handoff.

Use the NameCert obligation surface to bind the continuation and provenance rows. The packet can replay the route only through H, C, L, N, so the paired windows have already been recorded when a real-facing consumer reaches the comparison row.

Finally apply the Real seal handoff Definition 560.4. The seal-facing read occurs after the shared threshold, comparison row, transports, continuations, ledger, provenance, and local name have been displayed, so it cannot bypass either scheduled window. □

Theorem 560.33 (RegularCauchyTailMeet shared-threshold transport)**.** *Let two accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *presentations have componentwise* hsame *comparisons on the paired source rows, paired window rows, paired modulus rows, shared threshold rows, comparison rows, ledger rows, package rows, and local naming rows. Then any displayed* Cont *route transporting one presentation to the other preserves the public shared-threshold row.*

Thus the row serving the tail threshold in the source presentation is transported to the row serving the tail threshold in the target presentation, and the public certificate still reads the same componentwise route through $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$.

Proof. Begin with the componentwise transport data in the carrier. The hsame rows identify the two source rows, two scheduled windows, two modulus rows, the threshold row, comparison row, ledger row, package row, and local naming row.

Apply Definition 560.3. It keeps the classifier boundary stable under those componentwise comparisons and does not introduce a second threshold coordinate.

Use the displayed Cont route. Since the continuation is attached to the same local packet rows and the public certificate reads its threshold through those rows, the transported certificate consumes the transported threshold row and preserves the shared-threshold route. □

Theorem 560.34 (RegularCauchyTailMeet consumer exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or diagonal-window consumer of the public* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold reads only*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

Here R_0, R_1 *are the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows,* W_0, W_1 *are the paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* M_0, M_1 *are the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus rows,* τ *is the shared dyadic threshold,* Q *is the paired comparison row,* H *gives componentwise* hsame *transport,* C *gives displayed* Cont *continuation,* L *is the finite ledger, and* N *is the local* NameCert *row.*

Proof. For a $\mathsf{Real}^{\uparrow}$-facing consumer, apply Definition 560.4. The real seal is reached only after the shared threshold, comparison row, transports, continuations, ledger, provenance, and naming rows have been displayed.

For a diagonal-window consumer, apply Definition 560.10. The selected window and dyadic observation route read the same threshold before the $\mathsf{Real}^{\uparrow}$ seal can be used.

Finally use Definition 560.26. The threshold itself descends through the paired source rows, window rows, and modulus rows, so every consumer route is exhausted by the listed finite coordinates. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket consumer exactness`

Theorem 560.35 (RegularCauchyTailMeet diagonal consumer non-escape)**.** *No diagonal-window or* $\mathsf{Real}^{\uparrow}$*-facing consumer of the public* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *certificate can export a quotient*

stream, host real equality, hidden limit selection, trichotomy decision, or ambient completeness witness. Every exported datum is routed through the finite rows

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

Proof. Start from consumer exactness in Definition 560.34. A consumer can read only the displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, and naming rows.

Inspect those rows through Definition 560.1. None of them is a quotient-stream coordinate, a host real equality coordinate, a selected-limit coordinate, a trichotomy coordinate, or an ambient completeness coordinate.

The bridge-facing and real-facing routes named by Definition 560.11 therefore preserve the same exclusion boundary. They can move the public threshold to a consumer, but they cannot manufacture any of the excluded host or quotient data. □

Theorem 560.36 (RegularCauchyTailMeet obligation-closure readiness)**.** *The* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet presents the finite obligation package needed by* $\mathsf{Real}^{\uparrow}$ *and completion-facing diagonal consumers. The package consists of the existing* NameCert *obligations, ledger exactness, shared-threshold cofinality, dyadic endpoint meet, public shared-threshold certificate, window-source coverage, shared-threshold transport, consumer exactness, and diagonal consumer non-escape.*

Consequently every downstream read of the shared tail threshold is served by the finite coordinates

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

with no quotient stream, hidden limit selector, host real equality, trichotomy decision, or ambient completeness premise.

Proof. First collect the carrier and local naming obligations from Definition 560.1 and Definition 560.2. These expose the finite packet rows and the local admission surface.

Next apply Definition 560.7, Definition 560.8, and Definition 560.6. They route the source modulus reads to one shared dyadic threshold and keep endpoint refinement inside the displayed ledger.

Use the public shared-threshold certificate together with window-source coverage and shared-threshold transport. These results identify the consumer-facing threshold row and preserve it under componentwise transport.

Finally apply consumer exactness and diagonal consumer non-escape. The downstream diagonal and $\mathsf{Real}^{\uparrow}$ consumers read exactly the listed finite coordinates, and the excluded quotient, host, selection, trichotomy, and completeness data never appear as carrier coordinates or exported rows. □

Theorem 560.37 (RegularCauchyTailMeet bridge package)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the public shared-threshold certificate, diagonal-window threshold handoff, dyadic endpoint meet, ledger exactness, and* $\mathsf{Real}^{\uparrow}$ *seal handoff assemble the bridge from the concrete finite packet to the abstract* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *interface consumed by* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, FiniteCauchyApproximat *and diagonal* $\mathsf{Real}^{\uparrow}$ *routes.*

The bridge package reads only the paired $\mathsf{RegularCauchyName}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *schedules, source modulus rows, one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, paired comparison row, componentwise* hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, selected diagonal-window data, dyadic endpoint rows, and local* NameCert *evidence. It exports no quotient stream, selected limit, trichotomy row, host equality, independent tail oracle, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetUp StdBridge`

Proof. First apply Definition 560.11. It packages the concrete shared-threshold data as a public object for diagonal-window and diagonal $\mathsf{Real}^{\uparrow}$ consumers.

Use Definition 560.10. The selected-window consumer reads the displayed shared threshold before the $\mathsf{Real}^{\uparrow}$ seal route, so the abstract interface receives the same finite threshold row that the concrete packet exposes.

Next apply Definition 560.6. The endpoint route identifies the paired dyadic tolerance ledgers with the single shared threshold row, which is the row consumed by $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ and by the diagonal routes.

Finally use Definition 560.7 together with the real-seal handoff already named in the public certificate. The finite ledger accounts for every exported route, so the bridge from the concrete packet to the abstract interface carries exactly the listed rows and none of the excluded completeness or quotient data. □

Theorem 560.38 (RegularCauchyTailMeet mature downstream examples)**.** *The shared-tail threshold packet exported by* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *is consumed by* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{FiniteCauchyApproxin}$ *and diagonal* $\mathsf{Real}^{\uparrow}$ *routes through the existing public shared-threshold certificate and bridge package. Each consumer receives the paired source rows, paired schedules, source modulus rows, one shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, paired comparison row, componentwise* hsame *transports, displayed* Cont *routes, finite ledger,* Pkg *provenance, selected diagonal-window data, dyadic endpoint rows, and local* $\mathsf{NameCert}$ *evidence, with no quotient stream, selected limit, host equality, independent tail oracle, or ambient completeness.*

Proof. Begin with the public shared-threshold certificate Definition 560.11. It presents the shared threshold as a public object for $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers.

Apply the bridge package Definition 560.37. The bridge names $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ as a consumer of the same shared threshold and keeps the route inside the concrete finite packet.

The three consumer examples therefore use one shared-tail threshold surface. Since the certificate and bridge package both preserve the exclusions on quotient streams, selected limits, host equality, independent tail oracles, and ambient completeness, none of those objects is exported to the downstream examples. □

Theorem 560.39 (RegularCauchyTailMeet selector admission)**.** *An accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet admits a* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *row exactly through the displayed shared-tail route. The selector precision request, stream window, regularity row, dyadic readback, selected-tail witness, componentwise transports, continuations, package provenance, and local* $\mathsf{NameCert}$ *row must factor through one of the paired source schedules*

$$R_0, W_0, M_0 \qquad \text{or} \qquad R_1, W_1, M_1$$

and then through the common threshold row τ *of the meet packet.*

Conversely, either paired source schedule together with the shared threshold row, comparison row Q*, componentwise* hsame *row* H*, displayed* Cont *row* C*, finite ledger* L*,* Pkg *row, and local naming row* N *supplies the complete selector-facing admission route. No quotient stream, selected limit, host real equality, independent tail oracle, or ambient completeness witness is part of this admission surface.*

Proof. For the forward direction, read the proposed selector row through Definition 560.26. The threshold read must pass through one of the displayed source schedules and then through the shared dyadic threshold τ.

Apply Definition 560.34. It bounds all selector-facing precision, window, readback, transport, continuation, package, and local naming data by the finite rows $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$.

For the reverse direction, choose either paired schedule already displayed in the carrier. The shared-threshold transport theorem keeps the selected schedule attached to the same τ, while the obligation-closure readiness theorem repacks the route as a downstream consumer surface.

The exclusion statement follows from diagonal consumer non-escape. The selector admission has no coordinate for quotient streams, selected limits, host real equality, independent tail oracles, or ambient completeness. □

Theorem 560.40 (RegularCauchyTailMeet source scope)**.** *Every accepted source read of a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet factors through the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, source modulus rows, and the displayed shared dyadic threshold before any* $\mathsf{Real}^{\uparrow}$ *seal row is available. The source-facing surface is therefore exactly the finite route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

with the shared threshold τ serving as the only passage from the two source schedules to the comparison row and the seal-facing handoff.

Proof. Start with the carrier definition and local $\mathsf{NameCert}$ obligations. Definition 560.1 lists the paired source, window, modulus, threshold, comparison, transport, continuation, ledger, package, and local naming rows, while Definition 560.2 makes those rows the admissible source-reading surface.

Next apply ledger exactness and shared-threshold cofinality. By Definition 560.7, an accepted source read reaches τ only through the paired windows and source modulus rows; by Definition 560.8, stronger displayed thresholds remain on the same finite route.

Finally use selector admission and seal compatibility together with the real-seal handoff. Definition 560.39 keeps selector reads on one paired schedule and the shared threshold, and Definition 560.43 keeps every $\mathsf{Real}^{\uparrow}$-facing seal read behind the same displayed rows. Thus no seal row is available before the source schedules, modulus rows, and shared threshold have all been displayed. □

Theorem 560.41 (RegularCauchyTailMeet classifier scope)**.** *The classifier comparison of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is determined by the shared threshold τ, the paired comparison row Q, the componentwise* hsame *transports in H, and the displayed* Cont *routes in C. It is not determined by quotient stream equality, host real equality, a selected limit, trichotomy, or an ambient completeness coordinate.*

Proof. Read the classifier row through Definition 560.1 and Definition 560.3. The carrier exposes one shared threshold and one paired comparison row, and classifier stability preserves that comparison through the componentwise transports and continuations already stored in the packet.

Use shared-threshold cofinality to compare stronger tail requests. Definition 560.8 allows refinement of the displayed threshold without introducing a second threshold coordinate, independent stream equality test, or host real equality test.

The real-seal handoff and seal compatibility keep the same comparison visible to downstream readers. By Definition 560.4 and Definition 560.43, seal-facing consumers receive the threshold,

comparison, transports, and routes as displayed rows, so the classifier surface has exactly the stated scope. □

Theorem 560.42 (RegularCauchyTailMeet ledger scope)**.** *The finite ledger of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet records only the two source schedules, source modulus rows, shared threshold, paired comparison, componentwise transports, displayed continuation routes, package provenance, and local* $\mathsf{NameCert}$ *row. It records no selected limit, trichotomy row, quotient stream, host real equality, independent tail oracle, or ambient completeness coordinate.*

Proof. Begin with the carrier and ledger exactness. Definition 560.1 identifies L as the finite ledger attached to the displayed rows, and Definition 560.7 exhausts the route from the paired source schedules and modulus rows to τ, Q, and the real-facing handoff.

Add the naming, selector, and seal rows. The local obligations of Definition 560.2, the selector route of Definition 560.39, and the seal route of Definition 560.43 all read the same ledger entries rather than appending hidden completion data.

The exclusion boundary is the one used by classifier stability and the real-seal handoff. Since Definition 560.3 and Definition 560.4 require only the displayed threshold, comparison, transports, continuations, package provenance, and local name, the ledger has no coordinate from which any selected limit, trichotomy, quotient stream, host equality, independent oracle, or ambient completeness witness could be exported. □

Theorem 560.43 (RegularCauchyTailMeet seal compatibility)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or diagonal* $\mathsf{Real}^{\uparrow}$ *consumer that reads a shared* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold must read it through the paired source rows, the common threshold row, the paired comparison row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *route of the packet.*

Thus seal-facing consumption is compatible with the same finite rows

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

used by the public shared-threshold certificate and by the bridge package. It introduces no quotient stream, selected limit, host real equality, independent tail oracle, or ambient completeness premise.

Proof. Begin with the public shared-threshold certificate. It identifies the threshold consumed by diagonal $\mathsf{Real}^{\uparrow}$ routes with the displayed common threshold row τ and keeps the consumer on the paired source schedules.

Use the real-seal handoff and the bridge package. The handoff reads the $\mathsf{Real}^{\uparrow}$ seal only through the shared threshold, comparison row, transports, continuations, ledger, provenance, and local naming row; the bridge package preserves those same rows for completion-facing consumers.

For a $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer, follow the same bridge-facing route. The consumer receives the shared threshold and comparison data already carried by the finite packet, and the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows provide the displayed compatibility path.

Non-escape is inherited from consumer exactness and diagonal consumer non-escape. Since all seal reads are confined to the listed finite rows, no excluded quotient, host, selection, oracle, or completeness datum is available. □

Theorem 560.44 (RegularCauchyTailMeet root consumer package)**.** *The selector admission theorem, seal compatibility theorem, public shared-threshold certificate, and bridge package assemble the downstream root package for* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$*, and diagonal* $\mathsf{Real}^{\uparrow}$ *consumers.*

The package exports one shared threshold route from the paired $\mathsf{RegularCauchyName}^{\uparrow}$ *sources and paired* $\mathsf{StreamName}^{\uparrow}$ *schedules through source modulus rows, the dyadic threshold ledger, comparison row, componentwise* hsame *transports, displayed* Cont *continuations, finite ledger,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, selected limit, host real equality, independent tail oracle, or ambient completeness principle.*

Proof. First collect selector admission. It shows that every selector-facing precision and selected-tail read factors through one paired source schedule and the shared threshold row.

Next collect seal compatibility. It shows that $\mathsf{CauchyLimitSeal}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers read the same threshold through the displayed BHist, hsame, Cont, Pkg, and NameCert route.

Add Definition 560.11 and Definition 560.37. The certificate serves $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and diagonal $\mathsf{Real}^{\uparrow}$ consumers, while the bridge package keeps completion-facing reads inside the same finite shared-tail packet.

The four consumer families therefore share one root package. Its rows are exactly the paired sources, schedules, modulus reads, threshold, comparison, transports, continuations, ledger, provenance, and local name already displayed by the carrier, so the excluded quotient, host, selection, oracle, and completeness data are not exported. □

Theorem 560.45 (RegularCauchyTailMeet root carrier admission)**.** *The root-facing admission surface for an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is the shared finite threshold row*

$$T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N)$$

of Definition 560.1. A downstream root consumer may admit the packet only after the paired $\mathsf{RegularCauchyName}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired source modulus rows, shared dyadic threshold, comparison row, componentwise* hsame *transports, displayed* Cont *routes, finite ledger, and local* NameCert *row are visible.*

Proof. Start with the carrier and the local NameCert obligations. They display the two source rows, two scheduled windows, source modulus rows, threshold row, comparison row, transport rows, continuation rows, ledger, and local name.

Use window-source coverage. Every threshold-facing read must pass through the paired sources, paired windows, and paired source modulus rows before it reaches τ.

Apply shared-threshold cofinality to the root-facing threshold. It keeps any strengthened threshold tied to the same paired schedules and source modulus rows, so root admission remains a finite shared-threshold row rather than an independent tail oracle. □

Theorem 560.46 (RegularCauchyTailMeet root classifier stability)**.** *For root-facing paired-window consumers, componentwise* hsame *transport of the rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

preserves the displayed comparison classifier. The transported root surface reads the same paired-window route to τ *and* Q*, and it cannot replace the shared threshold by a quotient-stream, host-limit, selected-subsequence, or ambient-completeness classifier.*

Proof. Project the paired-window route from the carrier. The classifier uses only the paired source rows, schedules, modulus rows, shared threshold, and comparison row.

Apply classifier stability and shared-threshold transport. The componentwise hsame rows move these coordinates together, and the displayed Cont routes replay the same window-to-threshold-to-comparison path.

Use paired-window ledger coverage to keep both scheduled windows recorded before a real-facing or root-facing read can consume the comparison. Since the ledger has no alternate classifier coordinate, the transported root surface preserves exactly the displayed comparison classifier. □

Theorem 560.47 (RegularCauchyTailMeet root ledger exactness)**.** *The root-facing ledger of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet records exactly two input tails and one meet threshold: the rows* R_0, W_0, M_0 *for the first tail, the rows* R_1, W_1, M_1 *for the second tail, the shared dyadic threshold row* τ*, the paired comparison row* Q*, and the structural rows* H, C, L, N*. No root-facing ledger entry records a third tail, an independent threshold, a quotient-stream equality, a selected limit, host real equality, or ambient completeness.*

Proof. Begin with ledger exactness. It confines every accepted read to the route from the paired regular Cauchy sources through their windows and source modulus rows to τ, Q, and the $\mathsf{Real}^{\uparrow}$ handoff.

Use paired-window ledger coverage. Both scheduled windows must be present in L before the shared-tail certificate is available, so the ledger has exactly the two displayed input-tail sides.

Apply real-seal route totality. The real-facing route is already exhausted by the displayed shared threshold and comparison rows, so no additional tail or threshold can be introduced at the root boundary. □

Theorem 560.48 (RegularCauchyTailMeet uniform criterion consumer link)**.** *The root-facing* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *obligation rows supply the shared-tail consumer link required by the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *root surface of Definition 435.120. A uniform-criterion consumer may read the meet packet only through the admitted shared threshold, the hsame-stable paired-window classifier, and the exact two-tail ledger of the preceding root rows.*

Proof. First collect root carrier admission. It presents the shared finite threshold row required by a uniform Cauchy consumer.

Next collect root classifier stability. The paired-window comparison is preserved under componentwise hsame and displayed Cont replay, so the uniform criterion reads a stable shared-tail classifier.

Finally collect root ledger exactness and apply Definition 435.120. The uniform criterion receives exactly the two input tails and one meet threshold already recorded by the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ ledger, with no quotient, host-limit, selection, oracle, or completeness data available outside that finite route. □

Theorem 560.49 (RegularCauchyTailMeet window-source exhaustion)**.** *Every accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *comparison reads exactly the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the shared-tail carrier. It has no additional window source beyond the paired regular Cauchy names, paired stream schedules, source modulus rows, dyadic threshold ledger, comparison row, componentwise transports, displayed continuations, package provenance, and local naming row.*

Proof. Begin with the carrier and the window-source coverage theorem. They display the two regular Cauchy sources, the two stream schedules, the source modulus rows, the dyadic threshold ledger, and the comparison row.

Apply consumer exactness. It states that comparison-facing reads are confined to the displayed carrier rows and the componentwise hsame transports, displayed Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row.

Use the root consumer package to repack the route for downstream consumers. The resulting comparison therefore exhausts the listed source rows and exposes no extra window, oracle, quotient stream, host limit, or ambient completeness row. □

Theorem 560.50 (RegularCauchyTailMeet shared-threshold non-escape)**.** *Shared-threshold reads of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet cannot introduce a host limit, quotient stream equality, or choice-selected subsequence. Every such read remains inside the shared dyadic threshold, the paired stream schedules, the source modulus rows, the comparison row, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *evidence.*

Proof. Start with the shared-threshold transport theorem. It identifies threshold transport with the displayed schedule, modulus, dyadic, comparison, and transport rows.

Next use diagonal consumer non-escape. That theorem rules out quotient stream equality, host real equality, selected subsequences, hidden tail oracles, and ambient completeness data for diagonal consumers of the same packet.

The root package sends shared-threshold consumers through the same route. Hence a shared-threshold read can preserve the finite threshold evidence, but it cannot create any of the excluded host, quotient, or selection data. □

Theorem 560.51 (RegularCauchyTailMeet real-seal route totality)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet must pass through the same tail-meet window and dyadic endpoint rows before reading the real seal. The route is total for admitted seal reads because the real-seal handoff, dyadic endpoint meet, consumer exactness, diagonal non-escape, and root consumer package share the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *boundary.*

Proof. First use the real-seal handoff. It places every $\mathsf{Real}^{\uparrow}$ seal read behind the shared threshold route, comparison row, transport rows, continuations, ledger, provenance, and local name.

Then use the dyadic endpoint meet. The endpoint data consumed by the seal route are the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ threshold and endpoint rows, not a host real limit or quotient stream witness.

Finally combine consumer exactness, diagonal non-escape, and the root consumer package. These theorems keep every admitted seal read inside the same finite tail-meet window, so the route is total for accepted consumers and exports no hidden completion principle. □

Theorem 560.52 (RegularCauchyTailMeet source-window-modulus route)**.** *Every shared-threshold read of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet factors through the paired* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, the paired* $\mathsf{StreamName}^{\uparrow}$ *window rows, and the source modulus rows before it reaches the displayed dyadic threshold ledger. The route is a finite* BHist *route recorded by the carrier, the local* NameCert *obligations, ledger exactness, shared-threshold cofinality, and real-seal route totality.*

Proof. Begin with the carrier and NameCert obligations. They expose the paired regular Cauchy source rows, stream windows, source modulus rows, dyadic threshold ledger, comparison row, transports, continuations, provenance, and local name as displayed coordinates.

Use shared-threshold cofinality and ledger exactness next. Any threshold read that refines the common tail must first choose the visible source rows, then the matching window rows, and then the modulus rows that justify the dyadic threshold coordinate.

Finally apply real-seal route totality. It routes admitted seal-facing reads through the same tail-meet window before any real-seal observation is read, so the source-window-modulus factorization is the common route for shared-threshold consumers. □

Theorem 560.53 (RegularCauchyTailMeet formal target envelope)**.** *Every accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet whose source-window-modulus route is read by a* FiniteObservationBudgetSelector$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *consumer factors through the same paired* RegSeqRat$^{\uparrow}$ *sources,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *threshold ledger, comparison row,* hsame *transports,* Cont *routes,* Pkg *provenance, and* NameCert *row before any* Real$^{\uparrow}$ *seal is exposed. The downstream budget-selector and limit-seal reads therefore consume one finite handoff, not a hidden completion route and not a quotient-stream route.*

Proof. Begin with root carrier admission, root classifier stability, and root ledger exactness. These rows identify the accepted packet as the displayed two-source, two-window, one-threshold surface and keep transported comparisons inside the same finite ledger.

Use real-seal route totality and the source-window-modulus route. They place every seal-facing read behind the paired sources, paired windows, source modulus rows, dyadic threshold, comparison, transports, continuations, provenance, and local name before the Real$^{\uparrow}$ seal can be read.

Now read the consumer rows in the FiniteObservationBudgetSelector$^{\uparrow}$ and CauchyLimitSeal$^{\uparrow}$ surfaces. The selector consumes finite window and dyadic-budget rows before its seal handoff, while the limit seal consumes the same shared threshold route before exposing a completion-facing seal.

The two downstream routes therefore meet at the same finite handoff. Since neither consumer receives an additional quotient stream, host-limit equality, selected representative, or ambient completeness coordinate, the formal target envelope is exactly the displayed RegularCauchyTailMeet$^{\uparrow}$ route. □

Theorem 560.54 (RegularCauchyTailMeet formal handoff Lean target)**.** *The root carrier admission, root classifier stability, root ledger exactness, source-window-modulus route, real-seal route totality, and formal target envelope jointly determine the exact finite handoff for a later Lean target over* RegularCauchyTailMeet$^{\uparrow}$*. That handoff consists of paired* RegSeqRat$^{\uparrow}$ *sources, paired* StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *threshold row, the comparison row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert*.*

The target contains no quotient stream, host limit, selected subsequence, or ambient completeness coordinate.

Proof. Begin with root carrier admission, root classifier stability, and root ledger exactness. These rows fix the two source rows, two window rows, threshold ledger, comparison row, transports, continuations, provenance, and local name as one finite carrier surface.

Next apply the source-window-modulus route. It identifies how the paired sources and windows reach the displayed dyadic threshold row before a shared-threshold read can proceed.

Use real-seal route totality and the formal target envelope for the downstream boundary. They require seal-facing consumers and formal consumers to read the same finite handoff before any Real$^{\uparrow}$ seal is exposed.

Thus a later Lean target has an exact object to encode: the paired sources, paired windows, threshold, comparison, transports, routes, provenance, and local name. The excluded quotient, host-limit, subsequence, and completeness data have no coordinate in that object. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetFormalHandoffLeanTarget`

Theorem 560.55 (RegularCauchyTailMeet window-budget exhaustion)**.** *Every budget-selector or* CauchyLimitSeal$^{\uparrow}$ *consumer of an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet exhausts its read through the two* RegSeqRat$^{\uparrow}$ *source rows, two* StreamName$^{\uparrow}$ *windows, the two source modulus rows, the*

shared DyadicRatCore$^{\uparrow}$ *threshold, comparison row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The consumer receives no further budget window, hidden Cauchy tail, quotient stream equality, selected limit, or ambient completeness coordinate.*

Proof. Begin with the source-window-modulus route, Definition 560.52. It sends every shared-threshold read through the paired regular Cauchy source rows, paired stream windows, and source modulus rows before the dyadic threshold ledger can be read.

Next apply root ledger exactness and real-seal route totality, Definition 560.47 and Definition 560.51. These keep the shared threshold, comparison row, transports, continuations, provenance, and local name on the same finite root surface for every admitted seal-facing read.

Finally use the formal target envelope and handoff target, Definition 560.53 and Definition 560.54. The budget-selector and CauchyLimitSeal$^{\uparrow}$ routes therefore consume exactly the displayed window-budget handoff, and the packet has no remaining coordinate for a second budget window, hidden tail, quotient stream equality, selected limit, or ambient completeness datum. □

Theorem 560.56 (RegularCauchyTailMeet shared-threshold route determinacy)**.** *Two shared-threshold reads of an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet that use the same transported* RegularCauchyName$^{\uparrow}$ *source rows, the same transported* StreamName$^{\uparrow}$ *window rows, and the same transported source modulus rows reach the same displayed dyadic threshold ledger. The determinacy is internal to the finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Proof. Project both reads through the source-window-modulus route. By hypothesis the transported source rows, window rows, and source modulus rows agree componentwise under the displayed hsame transport.

Apply classifier stability and ledger exactness. The paired-window classifier cannot distinguish two reads with the same transported source, window, and modulus coordinates, and the exact ledger has only the displayed dyadic threshold coordinate for that route.

The Cont, Pkg, and NameCert rows then report the same finite ledger entry. Hence both reads reach the same displayed dyadic threshold ledger without invoking quotient stream equality, host limits, or choice-selected subsequences. □

Theorem 560.57 (RegularCauchyTailMeet public tail-meet certificate)**.** *The source-window-modulus route, shared-threshold route determinacy, and the public shared-threshold certificate assemble a public* RegularCauchyTailMeet$^{\uparrow}$ *tail-meet certificate. Any public consumer receives the paired regular Cauchy sources, paired stream windows, source modulus rows, dyadic threshold ledger, comparison row, transports, continuations, provenance, and local NameCert evidence, and receives no quotient stream equality, selected limit, host real equality, or ambient completeness theorem.*

Proof. First use Definition 560.52. It gives the public consumer a finite route from source rows through stream windows and source modulus rows to the displayed dyadic threshold ledger.

Next use Definition 560.56. It makes the displayed threshold ledger independent of duplicate reads that carry the same transported source, window, and modulus coordinates.

Finally combine the public shared-threshold certificate with real-seal route totality. The public certificate exposes the shared threshold as finite NameCert data, while the route-totality boundary keeps seal-facing consumers inside the same tail-meet window and excludes quotient, selection, host-equality, and ambient-completeness data. □

Theorem 560.58 (RegularCauchyTailMeet root threshold source ledger)**.** *Every root-facing threshold read of an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet factors through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

before it reaches a downstream root consumer.

Proof. Project the carrier to expose the two source rows R_0, R_1, the two window rows W_0, W_1, the two modulus rows M_0, M_1, the shared threshold τ, comparison row Q, transport H, continuation C, ledger L, and local name N.

Apply the root carrier admission, root classifier stability, and root ledger exactness results. They identify those displayed rows as the root-facing threshold surface and exclude a second root threshold coordinate.

Finally use the uniform-criterion consumer link. It reads the same finite ledger before any UniformCauchyCriterionUp consumer sees the packet, so every root-facing threshold read factors through the listed rows. □

Theorem 560.59 (RegularCauchyTailMeet root shared-threshold coverage)**.** *The shared threshold τ in an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet covers both source modulus rows M_0 and M_1 before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer reads the packet.*

Proof. Use the carrier and root carrier admission to project the paired sources, windows, modulus rows, and shared threshold.

Apply root classifier stability and root ledger exactness. The threshold row is recorded once in the ledger and is classified against both source modulus rows, so it is not a one-sided witness.

Then apply the uniform-criterion consumer link together with the real-seal route. Both downstream consumers enter through the same displayed threshold ledger, hence the shared threshold covers M_0 and M_1 before either consumer reads the packet. □

Theorem 560.60 (RegularCauchyTailMeet root consumer exhaustion)**.** *Downstream consumers of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet receive only the two-tail, one-threshold finite ledger. They receive no quotient stream, selected limit, host equality, trichotomy, or ambient completeness row.*

Proof. Start with the carrier and the public tail-meet certificate. They expose exactly the paired source rows, paired window rows, source modulus rows, one shared threshold ledger, comparison, transport, continuations, package provenance, and local name.

Apply root ledger exactness and root classifier stability. These results keep root consumers on the displayed two-tail and one-threshold surface rather than on a quotient or selected-limit surface.

Finally use the uniform-criterion consumer link. It reuses the same finite ledger for downstream reads, so quotient stream, selected limit, host equality, trichotomy, and ambient completeness rows are unavailable to those consumers. □

Theorem 560.61 (RegularCauchyTailMeet root threshold provenance)**.** *Every root threshold read of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is sourced by the displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and inherits provenance only from the contributing $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{NameCert}$ *rows.*

Proof. Project the carrier and use Definition 560.57. The two source rows, two windows, two modulus rows, shared threshold, comparison, transport, continuation, ledger, and local name are the whole root-facing threshold surface.

Then apply Definition 560.52. It supplies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ provenance for that surface, while the local $\mathsf{NameCert}$ row names the same finite route. □

Theorem 560.62 (RegularCauchyTailMeet root window-modulus exactness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the paired window rows* W_0, W_1 *and source modulus rows* M_0, M_1 *are exactly the route to the shared dyadic threshold* τ*. Any root-facing threshold read reaches* τ *through the source-window-modulus route and the shared-threshold route determinacy.*

Proof. Apply Definition 560.52 to factor a threshold read through the paired window rows and source modulus rows. This gives the displayed route to the dyadic threshold ledger.

Apply Definition 560.56. A second root-facing read with the same transported windows and modulus rows reaches the same threshold ledger, so the paired window and modulus rows are the exact route to τ. □

Theorem 560.63 (RegularCauchyTailMeet root consumer export boundary)**.** *Downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumers of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet receive only the finite threshold packet*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

They receive no quotient stream, selected limit, host equality, trichotomy, or ambient completeness row.

Proof. Use Definition 560.60. It confines downstream consumers to the two-tail, one-threshold ledger and excludes quotient, selected-limit, host-equality, trichotomy, and completeness rows.

Then use Definition 560.61 and Definition 560.62. These identify the exported ledger with the displayed finite packet and route both $\mathsf{Real}^{\uparrow}$ and $\mathsf{UniformCauchyCriterion}^{\uparrow}$ reads through the same threshold surface. □

Theorem 560.64 (RegularCauchyTailMeet root seal-budget totality)**.** *Every admitted root-facing* $\mathsf{Real}^{\uparrow}$ *seal-budget read of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is total on the displayed shared-threshold route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The read must pass through the paired $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired source modulus rows, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, and the local* $\mathsf{NameCert}$ *rows before the seal-budget consumer sees the packet.*

Proof. Start with root carrier admission and root threshold source ledger. These results expose the paired sources, windows, modulus rows, shared threshold, comparison row, transports, continuations, ledger, and local name as the whole root-facing route.

Use real-seal route totality. It places every $\mathsf{Real}^{\uparrow}$ seal read behind the same tail-meet window and dyadic endpoint rows, so an admitted seal-budget read has a displayed route whenever the root packet is accepted.

Finally apply root consumer export boundary. The exported data are exactly the finite threshold packet, and no quotient stream, selected limit, host equality, trichotomy, or ambient completeness row is supplied to complete the read elsewhere. □

Theorem 560.65 (RegularCauchyTailMeet root tail-comparison admission)**.** *A root consumer may admit the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *tail comparison only through the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired source modulus rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold, comparison row* Q*, componentwise* hsame *transports, displayed* Cont *replay, finite ledger, provenance, and local* $\mathsf{NameCert}$ *rows of the carrier.*

The admission surface is stable under the same transported source, window, and modulus rows, and it has no independent tail oracle, selected subsequence, quotient stream equality, or ambient completeness premise.

Proof. Project the comparison through root classifier stability. The classifier uses the displayed paired-window route to τ and Q, and componentwise transport preserves that route.

Apply source-window-modulus route and shared-threshold route determinacy. They force a root comparison read through the paired source, window, and modulus rows before it reaches the dyadic threshold ledger, and duplicate reads with the same transported route reach the same threshold.

Use window-source exhaustion and shared-threshold non-escape. These results exclude additional window sources, independent tail oracles, selected subsequences, quotient stream equality, host limits, and ambient completeness from the admitted comparison surface. □

Theorem 560.66 (RegularCauchyTailMeet root formal handoff surface)**.** *The root consumer package, root seal-budget totality, and root tail-comparison admission assemble the formal handoff surface for downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumers. The handoff binds the shared-threshold route to* $\mathsf{Real}^{\uparrow}$ *seal budgets, admits the tail comparison through* Q*, records* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *provenance, and preserves the local* $\mathsf{NameCert}$ *rows of the accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root formal handoff surface`

Proof. First collect the root consumer package. It assembles selector, seal, public shared-threshold, and bridge-facing consumers around one finite shared-threshold route.

Next add root seal-budget totality. This makes the $\mathsf{Real}^{\uparrow}$ seal-budget side total exactly on the displayed paired-source, paired-window, source-modulus, dyadic-threshold, comparison, transport, continuation, ledger, provenance, and naming rows.

Then add root tail-comparison admission. The comparison side is admitted only through the same Q row and the same transported source, window, modulus, threshold, route, provenance, and local certificate data.

Finally use root consumer export boundary and root threshold provenance. They identify the exported handoff with the finite packet and record its $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{NameCert}$ provenance, while excluding quotient streams, selected limits, host equality, trichotomy, and ambient completeness. □

Theorem 560.67 (RegularCauchyTailMeet selector-budget route lock)**.** *Any* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *read consumed by the root-facing* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *formal handoff of Definition 560.66 must pass through the same paired* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, source modulus rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, comparison row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row displayed by the accepted TailMeet packet before it reaches any* $\mathsf{Real}^{\uparrow}$ *seal.*

The selector-budget sibling of chapter 678 contributes only finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ *readback along that route. It supplies no second selector, quotient stream, host limit, selected subsequence, or ambient completeness coordinate beyond the root seal-budget and tail-comparison surfaces of Definition 560.64 and Definition 560.65.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket selector budget route lock`

Proof. Begin with the root formal handoff surface. By Definition 560.66, the downstream handoff is already the finite packet whose source, window, modulus, threshold, comparison, transport, route, provenance, and local certificate rows are displayed by the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ carrier.

Apply Definition 560.64. Any $\mathsf{Real}^{\uparrow}$ seal-budget read behind that handoff is total only on the displayed TailMeet route, so a selector-budget read consumed by the seal side cannot bypass the paired sources, windows, modulus rows, threshold ledger, Q row, transports, continuations, provenance, or local name.

Apply Definition 560.65. The comparison side is admitted through the same transported rows, and the admission excludes an independent tail oracle, selected subsequence, quotient stream equality, and ambient completeness premise.

Finally read the sibling route through chapter 678. That chapter supplies the finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ readback needed by the selector-budget consumer, and the preceding TailMeet rows lock that contribution to the root packet before any $\mathsf{Real}^{\uparrow}$ seal receives it. □

Theorem 560.68 (RegularCauchyTailMeet root formal budget-refusal totality)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, or* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *consumer that reaches a selector-budget or quotient-stream request from the root formal handoff must first pass through the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, source modulus rows, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, comparison row,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row of the accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet.*

If the same consumer asks for quotient-stream equality, the request factors through Definition 560.95 and the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *route of chapter 998. No such consumer receives a selected limit, quotient stream equality, host real equality, trichotomy row, independent tail oracle, or ambient completeness witness outside those displayed routes.*

Proof. First apply Definition 560.66. The handoff presented to downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumers is already the finite TailMeet packet, with its paired sources, windows, modulus rows, threshold ledger, comparison row, transports, continuations, provenance, and local certificate displayed.

Next apply Definition 560.67. It forces every selector-budget read consumed by that handoff, including the $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ sibling read, through the same accepted TailMeet rows before any $\mathsf{Real}^{\uparrow}$ seal can see the request.

For a quotient-stream request, use Definition 560.95 together with chapter 998. The completion-facing request first traverses the finite TailMeet source-window route and then enters the quotient-refusal route; if either route is absent, the request is recorded by the refusal ledger.

These two consumer routes cover the selector-budget and quotient-stream branches of the root formal handoff. The cited boundaries jointly exclude selected limits, quotient stream equality, host real equality, trichotomy rows, independent tail oracles, and ambient completeness as data exported to downstream consumers. □

Theorem 560.69 (RegularCauchyTailMeet root window threshold coverage)**.** *Every root consumer request for an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet reaches the shared threshold only through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau$$

together with the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows. The request has no route to the threshold outside the paired source rows, paired window rows, source-modulus rows, and finite transport/provenance surface.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root window threshold coverage`

Proof. Start with window-source coverage. The coverage theorem forces every threshold-facing read to expose the two source rows and the two window rows before the shared threshold can be read.

Next use source-modulus exhaustion. It attaches the two modulus rows M_0, M_1 to the same route and leaves no separate modulus source for a root consumer request.

Finally apply shared-threshold non-escape and consumer exactness. The displayed hsame, Cont, Pkg, and NameCert rows are therefore the only structural rows through which the threshold is transported to a root consumer. □

Theorem 560.70 (RegularCauchyTailMeet root modulus route exactness)**.** *For an accepted* RegularCauchyTailMee *packet, the source-modulus rows are the only admissible route into the shared-threshold ledger. A root threshold read is admitted exactly when it factors through the source-window-modulus route, the shared-threshold transport, and the displayed non-escape boundary.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root modulus route exactness`

Proof. Apply ledger exactness first. The shared threshold ledger has only the displayed dyadic threshold coordinate, so a proposed route cannot enter the ledger through an unlisted analytic row.

Next compose with shared-threshold transport. Transport moves the paired source, window, modulus, threshold, comparison, ledger, package, and local naming rows together, preserving the same threshold row for root reads.

Finally use the non-escape exclusions. Since quotient streams, selected limits, trichotomy rows, host equality, and ambient completeness are absent from the carrier, the source-modulus rows are the exact route into the threshold ledger. □

Theorem 560.71 (RegularCauchyTailMeet root Real seal consumer exhaustion)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *root consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet reads the finite tail-meet packet before any seal row is consumed. The consumer receives no quotient stream, selected limit, trichotomy row, host equality row, or ambient completeness row.*

↪ variant `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root downstream seal route`

Proof. First apply root window threshold coverage. It routes the root consumer through $R_0, R_1, W_0, W_1, M_0, M_1, \tau$ and the displayed structural rows before the threshold is available.

Next apply root modulus route exactness. The source-modulus route is the only entrance into the threshold ledger, so a regular-rational or real-facing consumer cannot bypass the finite tail-meet packet.

Finally compose with the Real seal handoff and the public shared-threshold certificate. The handoff places seal consumption after the finite packet, while the public certificate exposes only the shared threshold surface and none of quotient streams, selected limits, trichotomy, host equality, or ambient completeness. □

Theorem 560.72 (RegularCauchyTailMeet root observation-budget entry)**.** *For an accepted* RegularCauchyTailMee *packet, a root threshold read can enter the finite* RealObservationBudget$^{\uparrow}$ *surface of chapter 611 only through the paired source, window, modulus, and threshold rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau$$

together with the displayed comparison, transport, continuation, provenance, ledger, and local NameCert *rows. No budget entry route is supplied outside this finite TailMeet packet.*

Proof. Start with root window threshold coverage. It forces every root-facing read to expose the two source rows, two window rows, source modulus rows, shared threshold, and displayed structural rows before the threshold can be consumed.

Next apply root modulus route exactness. The source-modulus rows are the only admissible route into the shared-threshold ledger, so a budget entry cannot bypass $R_0, R_1, W_0, W_1, M_0, M_1, \tau$.

Finally read the target surface through chapter 611. That chapter admits only finite observation-budget rows, so the TailMeet threshold enters it through the displayed packet and not through quotient streams, host equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root observation budget entry`

Theorem 560.73 (RegularCauchyTailMeet root threshold readback determinacy)**.** *For an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet, any two root threshold readbacks that use the same displayed source-window-modulus route reach the same dyadic threshold row τ. The determinacy is read through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and through the finite observation-budget entrance of chapter 611.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root threshold readback determinacy`

Proof. First use root window threshold coverage to place both readbacks on the paired source rows, paired window rows, source modulus rows, and shared threshold row.

Next use root modulus route exactness. Since the source-window-modulus route is the exact entrance to the threshold ledger, two readbacks with the same displayed route cannot reach different dyadic threshold rows inside the finite packet.

Now apply root observation-budget entry. When the readback is served to RealObservationBudget$^{\uparrow}$, the budget side receives only the same displayed TailMeet rows, so it cannot split the dyadic readback by adding a quotient stream, selected limit, host equality, or ambient completeness coordinate. □

Theorem 560.74 (RegularCauchyTailMeet root seal consumer boundary)**.** *Every* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *seal consumer of an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet is bounded by the finite observation-budget surface reached from the root threshold route. The consumer can read only the TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and the finite RealObservationBudget$^{\uparrow}$ *rows admitted by chapter 611; it receives no quotient stream, host equality, selected limit, trichotomy row, or ambient completeness row.*

Proof. Begin with root Real seal consumer exhaustion. A $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ root consumer must read the finite TailMeet packet before any seal row is consumed.

Apply root observation-budget entry and root threshold readback determinacy. Together they route the threshold through the finite observation-budget entrance and make the dyadic readback determined by the displayed source-window-modulus route.

For a $\mathsf{Completion}^{\uparrow}$ seal consumer, use the same budgeted boundary: it receives the TailMeet threshold only after the observation-budget surface has been displayed. Since both the TailMeet packet and the observation-budget packet are finite NameCert surfaces, no quotient stream, host equality, selected limit, trichotomy row, or ambient completeness row is available to the consumer. □

Theorem 560.75 (RegularCauchyTailMeet root seal budget synchronization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any root seal read synchronizes the finite threshold route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

with the finite $\mathsf{RealObservationBudget}^{\uparrow}$ *entrance before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal consumer can read the seal. The consumer therefore sees the shared dyadic threshold only through the displayed TailMeet packet and the finite observation-budget rows, with no route through quotient streams, selected limits, host equality, trichotomy, or ambient completeness.*

Proof. First apply root window threshold coverage and root modulus route exactness. These results put every root threshold read on the paired source rows, paired window rows, source-modulus rows, shared threshold row, rational comparison row, transport row, continuation row, ledger row, and local naming row.

Next apply root observation-budget entry. The same finite route is the only entrance into the $\mathsf{RealObservationBudget}^{\uparrow}$ surface, so the seal-side consumer cannot receive τ before the observation-budget packet has been displayed.

Use root threshold readback determinacy to identify all readbacks along that route with the same dyadic threshold row. The budget entrance therefore carries a single determined TailMeet threshold rather than a separate consumer-side threshold.

Finally apply root seal consumer boundary. It places both $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal reads after the finite budgeted route and excludes quotient streams, selected limits, host equality, trichotomy rows, and ambient completeness from the seal consumer surface. □

Theorem 560.76 (RegularCauchyTailMeet observation-budget selector handoff)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose root threshold route is exported to a* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *consumer, the exported route first factors through the same finite TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and then through the selector route

$$B \to S \to W \to D \to R \to E.$$

The handoff exports no quotient stream, selected limit, host real equality, trichotomy row, countable-choice row, hidden selector, or ambient completeness theorem.

Proof. First place the exported read on the finite observation-budget surface. By Definition 560.72, the TailMeet route enters the $\mathsf{RealObservationBudget}^{\uparrow}$ side only through $R_0, R_1, W_0, W_1, M_0, M_1, \tau$ and the displayed comparison, transport, continuation, provenance, ledger, and local naming rows.

Next identify the threshold consumed by the selector-facing export. By Definition 560.73, every readback along the displayed route reaches the same dyadic threshold row τ. By Definition 560.75, that determined threshold is synchronized with the finite budget entrance before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read is available.

It remains to describe the selector side of the handoff. By Definition 782.7, an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet with the displayed budget rows determines one finite route $B \to S \to W \to D \to R \to E$. By Definition 782.10, diagonal-limit and Real-completion consumers may use that finite route but receive no independent completion principle from the selector.

Combining the two surfaces gives a single finite export: the TailMeet root threshold is read through $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$ and is then consumed through $B \to S \to W \to D \to R \to E$. None of the excluded analytic rows is present on either side of the handoff, so none can be exported. □

Theorem 560.77 (RegularCauchyTailMeet ObservationBudgetLimiter cap synchronization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose root threshold route is exported through* $\mathsf{RealObservationBudget}^{\uparrow}$ *and then consumed by an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet, the limiter cap row is synchronized with the same finite TailMeet route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

Every downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *or* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *read must pass first through the TailMeet source-window-modulus threshold route and then through the limiter triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

The synchronized handoff exports no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped schedule, independent $\mathsf{Real}^{\uparrow}$ *carrier, or ambient completeness principle.*

Proof. First fix the TailMeet side of the export. Apply Definition 560.72 to place the root threshold entrance inside $\mathsf{RealObservationBudget}^{\uparrow}$ on the displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, and local naming rows. Then apply Definition 560.73 and Definition 560.75. These identify every readback along that route with the same dyadic threshold row and the same finite budget entrance.

Next read the limiter side through chapter 903. By Definition 903.9, the accepted limiter packet admits a cap row K only after the request and observation-budget rows E, B and before any served window row. By Definition 903.10, the downstream readback rows W, D, R remain tied to that same capped route.

Finally combine this limiter boundary with Definition 560.76 and Definition 903.25. The first theorem shows that selector-facing reads consume the TailMeet threshold only through the finite observation-budget route, while the second identifies the public limiter service as the triad (E, B, K), (W, D, R), and (S, H, C, P, N). Thus each listed downstream consumer must read the TailMeet route before it reads the limiter triad, and the excluded analytic rows have no carrier coordinate or local $\mathsf{NameCert}$ clause from which to be exported. □

Theorem 560.78 (RegularCauchyTailMeet budgeted refinement seal factorization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any root threshold route exported to a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *consumer or to a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer is read in the following order. The consumer first reads the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

It then reads the finite observation-budget surface and any limiter cap surface attached to that route. Only after those finite rows have been displayed may a $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer read the seal. The route exports no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped observation schedule, independent completion carrier, or ambient completeness principle.*

Proof. Start with the root threshold route. By Definition 560.72, every accepted export enters the $\mathsf{RealObservationBudget}^{\uparrow}$ surface only through the displayed TailMeet rows, together with their comparison, transport, continuation, ledger, and local naming coordinates.

The threshold row is then fixed before the seal is visible. Apply Definition 560.73 to identify all readbacks along the route with the same dyadic threshold row τ, and apply Definition 560.75 to place the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal reads after the finite budget entrance.

Now attach the capped budget surface. By Definition 560.77, every downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, or selector-facing read must pass through the TailMeet route and then through the limiter service rows before it can consume a seal.

Finally compare the two named consumers. The refinement chapter chapter 885 reads a Real or Completion route only after its displayed source, selector-budget, window, readback, and seal rows are present, while the Cauchy-limit seal chapter chapter 386 records the finite seal handoff boundary. Composing these chapter surfaces with the TailMeet budget and limiter theorems gives the stated factorization and leaves none of the excluded analytic rows with a carrier coordinate from which to be exported. □

Theorem 560.79 (RegularCauchyTailMeet root budget-limiter factorization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any root threshold route exported to* $\mathsf{RealObservationBudget}^{\uparrow}$, ObservationBudgetLim $\mathsf{CauchyModulusRefinement}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *factors in one order. The route first reads the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

It then enters the finite observation-budget surface, then passes through the limiter cap and service triad, before any $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal is read. The route exports no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped observation schedule, independent completion carrier, or ambient completeness principle.*

Proof. First place the root route on the TailMeet packet. By Definition 560.72, every threshold export reaches $\mathsf{RealObservationBudget}^{\uparrow}$ only through the paired source, window, modulus, threshold, comparison, transport, continuation, ledger, and local naming rows.

Next fix the threshold readback. The determinacy theorem Definition 560.73 identifies all readbacks along that displayed route with the same dyadic threshold row τ, and Definition 560.75 synchronizes that threshold with the finite budget entrance before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal is available.

Now attach the limiter stage. By Definition 560.77, the downstream read must pass from the TailMeet route into the limiter cap and service triad (E, B, K), (W, D, R), and (S, H, C, P, N).

Finally apply the budgeted refinement seal factorization, Definition 560.78. It places $\mathsf{CauchyModulusRefinement}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ after the displayed TailMeet rows, finite observation-budget entrance, and capped limiter service, leaving no excluded analytic coordinate from which a Real or Completion seal can be read earlier. □

Theorem 560.80 (RegularCauchyTailMeet Real tail-agreement factorization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any accepted TailMeet root threshold route consumed by* $\mathsf{RealTailAgreementSeal}^{\uparrow}$,

$\mathsf{CauchyLimitSeal}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *factors through one finite order. The route first reads the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

It then reads the finite $\mathsf{RealObservationBudget}^{\uparrow}$ *surface, then the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *cap and service triad, and only after those finite rows have been displayed may the consumer read the tail-agreement seal. The route exports no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped schedule, independent completion carrier, or ambient completeness principle.*

Proof. Start with the root observation-budget entrance. By Definition 560.72, every accepted threshold route enters the budget surface only through the displayed TailMeet source, window, modulus, threshold, comparison, transport, continuation, ledger, and local naming rows.

Next fix the threshold and synchronize the seal side. Apply threshold readback determinacy, Definition 560.73, to identify every readback with the same dyadic threshold row τ. Then apply seal budget synchronization, Definition 560.75, to put the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal reads after that finite budget entrance.

Now attach the limiter and refinement stages. The observation-budget limiter cap synchronization theorem, Definition 560.77, places the cap and service triad after the TailMeet budget entrance, while Definition 560.78 keeps refinement and limit-seal consumers after the same finite rows.

Finally use the root budget-limiter factorization, Definition 560.79. It gives the common route order for $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$. Composing this with the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ synchronizer surface keeps the tail-agreement seal behind the displayed TailMeet rows, finite budget surface, and capped limiter service, so none of the excluded analytic or completion-side rows has a coordinate from which to enter earlier. □

Theorem 560.81 (RegularCauchyTailMeet limit seal window pullback)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose root threshold route is exported to an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet and an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, every seal-facing read pulls back to the same displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The same pullback then reaches the $\mathsf{StreamName}^{\uparrow}$ *finite-window lattice meet and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Conversely, those displayed rows repack into one consumer route for the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal row and the* $\mathsf{RealObservationBudget}^{\uparrow}$ *entrance. The route contains no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, unbounded schedule, separate completion carrier, or ambient completeness principle.*

Proof. First place the root route on the TailMeet packet. By Definition 560.79, any route exported toward $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, or $\mathsf{CauchyLimitSeal}^{\uparrow}$ must read the displayed TailMeet rows, finite observation-budget entrance, and capped limiter service before a seal is available.

Next move from the finite budget route to the seal read. Apply Definition 560.78 and Definition 560.75. These place every $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{RealObservationBudget}^{\uparrow}$ consumer after the same finite threshold route and synchronize the seal-facing read with the finite budget entrance.

Now identify the shared $\mathsf{StreamName}^{\uparrow}$ window. The finite-window lattice dependency of Definition 386.160 forces the seal-facing route through the $\mathsf{StreamName}^{\uparrow}$ finite-window meet, and the tail-budget meet of Definition 386.135 identifies that meet with the displayed regular source, dyadic ledger, real seal, transport, continuation, provenance, and local naming rows.

For the converse direction, start with exactly those displayed TailMeet, stream-window, regular-source, dyadic, real, transport, continuation, package, and naming rows. The cited finite-window and tail-budget meet theorems repack them as the $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal row together with the $\mathsf{RealObservationBudget}^{\uparrow}$ entrance, while the TailMeet budget-limiter factorization supplies the root consumer route.

Finally use Definition 560.73. It identifies all readbacks along the displayed route with the same threshold row τ, so the forward pullback and the reverse repacking describe one deterministic route and none of the excluded analytic rows has a coordinate from which to enter it. □

Theorem 560.82 (RegularCauchyTailMeet observation-budget route uniqueness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any two exports into* $\mathsf{RealObservationBudget}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *that use the same displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

reach the same observation-budget entrance and the same seal-facing pullback route. No quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped observation schedule, independent completion carrier, or ambient completeness principle can separate the two exports after those rows are fixed.

Proof. First apply the root observation-budget entry theorem, Definition 560.72. It places any export into $\mathsf{RealObservationBudget}^{\uparrow}$ on the displayed TailMeet rows and their comparison, transport, continuation, ledger, and local naming coordinates.

Next use root threshold readback determinacy, Definition 560.73. Since both exports use the same displayed rows, every readback reaches the same dyadic threshold row τ.

The root seal budget synchronization and root budget-limiter factorization, Definition 560.75 and Definition 560.79, put the finite budget entrance before any seal-facing Real or Completion read.

Finally apply the limit seal window pullback of Definition 560.81. It repacks the fixed TailMeet rows with the shared finite-window and seal rows, so both exports reach one observation-budget entrance and one seal-facing pullback route with none of the excluded analytic coordinates available. □

Theorem 560.83 (RegularCauchyTailMeet RealTailAgreementSeal handoff)**.** *Every* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold exported to* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *is consumed only after the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

are present, the finite observation-budget entrance is fixed, the limiter cap and service rows are present, the $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row has been inserted, the finite* $\mathsf{StreamName}^{\uparrow}$ *window is selected as a window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback is exposed, and the agreement row is available. The exported package carries no quotient stream, selected limit, host equality, trichotomy row, or ambient completeness row.*

Proof. First place the exported threshold on the TailMeet packet. The root budget-limiter factorization of Definition 560.79 forces the route through the displayed TailMeet rows, the finite

observation-budget entrance, and the capped limiter service before a downstream seal route can use it.

Next identify the finite window and readback. The limit seal window pullback of Definition 560.81 transports that same threshold route through the shared $\mathsf{StreamName}^{\uparrow}$ finite-window surface and the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ readback, with no separate analytic coordinate available.

Route uniqueness then turns the exported threshold into a single finite package. By Definition 560.82, any consumer using those displayed rows reaches the same observation-budget entrance and the same seal-facing pullback route.

Finally apply the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ budget synchronizer factorization, Definition 699.25. It inserts the synchronizer row before the finite window, dyadic readback, and agreement row are consumed, so the handoff exports one finite threshold package and none of the excluded quotient, selected-limit, host-equality, trichotomy, or ambient-completeness rows. □

Theorem 560.84 (RegularCauchyTailMeet root downstream window exhaustion)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *root consumer that reads the TailMeet threshold must first exhaust the paired window-source route through the displayed* BHist *rows R_0, R_1, W_0, W_1, the componentwise* hsame *row H, the displayed* Cont *route C, the finite ledger L, the package row* Pkg*, and the local* NameCert *row N. No downstream read can reach τ, Q, or a Real seal handoff before those four source-window rows have been exposed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root downstream window exhaustion`

Proof. Start with TailMeet window-source coverage. It forces any threshold-facing read to expose the two regular-Cauchy source rows R_0, R_1 and the two stream-window rows W_0, W_1 before the shared dyadic threshold is available.

Next attach the structural rows. Shared-threshold transport carries the exposed window-source route through the componentwise hsame row H and the displayed Cont route C, while consumer exactness records the same route in L, Pkg, and N.

Finally read the Real seal handoff. The handoff can consume only the already displayed finite TailMeet route, so a root downstream consumer cannot bypass the window-source rows to reach τ, Q, or any completion-facing seal row. □

Theorem 560.85 (RegularCauchyTailMeet root downstream threshold stability)**.** *Let $T = (R_0, R_1, W_0, W_1, M_0, M$ be an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. If a downstream root consumer transports the paired source-window-modulus route through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, then the shared threshold row τ and comparison row Q remain the TailMeet rows read by that consumer. The consumer may request a stronger displayed dyadic threshold through M_0 and M_1, but the stabilized route still factors through $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$.*

Proof. Use the shared-threshold transport theorem to move the paired source rows, window rows, modulus rows, threshold row, comparison row, finite ledger, package row, and local name together. The transported route therefore keeps τ and Q attached to the same TailMeet packet.

Now apply shared-threshold cofinality. A stronger displayed dyadic threshold can serve the downstream request only through the same source modulus rows M_0, M_1 and the same paired schedules W_0, W_1.

Consumer exactness and the Real seal handoff close the boundary. They record the transported threshold route in H, C, L, N and prevent the downstream consumer from replacing it by a host equality, quotient stream, selected limit, independent tail oracle, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root downstream threshold stability`

Theorem 560.86 (RegularCauchyTailMeet root downstream nonescape package)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet* $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N)$*, downstream root consumers receive exactly the finite TailMeet route certified by Definition 560.84 and Definition 560.85. The package exposes the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows and exports no quotient stream, selected limit, host real equality, trichotomy row, independent tail oracle, or ambient completeness witness.*

Proof. First combine downstream window exhaustion with downstream threshold stability. The first theorem forces each root consumer through R_0, R_1, W_0, W_1 before the threshold is visible, and the second keeps the transported threshold and comparison rows on τ and Q through M_0, M_1, H, C, L, N.

Next compose with consumer exactness. That theorem identifies the listed rows as the complete consumer-facing surface of the accepted TailMeet packet, including the package provenance and local naming evidence.

Finally apply the Real seal handoff rows. Since every Real or Completion consumer reaches the seal through the same TailMeet route, the downstream package has no coordinate from which to export quotient streams, selected limits, host equality, trichotomy, hidden tail data, or ambient completeness. □

Theorem 560.87 (RegularCauchyTailMeet root downstream seal route)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet* $T = (R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C$ *first factors through the paired source windows* R_0, R_1, W_0, W_1*, then through the modulus rows* M_0, M_1*, then through the shared threshold and comparison rows* τ, Q*, and finally through* H, C, L, Pkg, N*. The route exports no quotient stream, selected limit, host equality, trichotomy row, independent tail oracle, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket root downstream seal route`

Proof. Start with downstream window exhaustion. It forces every seal-facing consumer to read the paired source rows and window rows R_0, R_1, W_0, W_1 before the threshold or seal handoff can be visible.

Next apply downstream threshold stability. Any stronger displayed dyadic request must pass through M_0, M_1 while keeping the shared threshold row τ and the comparison row Q attached to the same TailMeet packet.

Finally use the downstream nonescape package. It repacks the window, modulus, threshold, comparison, transport, continuation, ledger, package, and local naming rows as the complete seal route and excludes quotient streams, selected limits, host equality, trichotomy, independent tail oracles, and ambient completeness. □

Theorem 560.88 (RegularCauchyTailMeet downstream unblock package)**.** *For an accepted* RegularCauchyTailMee *packet, every downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *root consumer receives exactly the combined route certified by window exhaustion, threshold stability, the nonescape package, and the seal route. The public package is the finite tuple* $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N$*, and it exports no quotient stream, selected limit, host equality, trichotomy row, independent tail oracle, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket downstream unblock package`

Proof. Apply Definition 560.84. Every root consumer first exposes the paired source-window rows R_0, R_1, W_0, W_1 before a threshold-facing or seal-facing read is available.

Apply Definition 560.85. The same route then carries the modulus rows M_0, M_1, preserves the shared threshold row τ, and keeps the comparison row Q attached to the accepted TailMeet packet.

Apply Definition 560.86. This identifies the consumer-facing surface with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows and excludes every unlisted root resource.

Apply Definition 560.87. The seal consumer therefore factors through the same finite route and cannot bypass the ledger or package evidence.

Ledger exhaustion leaves only $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N$ in the exported package, while non-escape removes quotient streams, selected limits, host equality, trichotomy, independent tail oracles, and ambient completeness witnesses from the downstream interface. □

Theorem 560.89 (RegularCauchyTailMeet budget-selector handoff)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, every* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *consumer must read the paired source windows* R_0, R_1, W_0, W_1*, modulus rows* M_0, M_1*, shared threshold* τ*, comparison row* Q*,* hsame *row* H*,* Cont *row* C*, ledger* L*, package row* Pkg*, and local* $\mathsf{NameCert}$ *row* N *before selecting a downstream budget. The handoff exports no selected limit, quotient stream, host equality, trichotomy row, independent tail oracle, or ambient completeness witness.*

Proof. Start with the downstream unblock package. It identifies the complete public route of an accepted TailMeet packet as the finite tuple $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N$ and excludes every unlisted root resource.

Apply Definition 560.84 and Definition 560.85. A budget selector must first read the paired source windows and modulus rows, and the shared threshold and comparison row remain attached to the same accepted packet.

Apply Definition 560.86 and Definition 560.87. These route the selector through the displayed hsame, Cont, ledger, package, and local naming rows.

Thus a $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ or $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ consumer can select a downstream budget only after consuming the already displayed finite route. Since that route contains no selected limit, quotient stream, host equality, trichotomy row, independent tail oracle, or ambient completeness witness, none is exported by the handoff. □

Theorem 560.90 (RegularCauchyTailMeet root consumer budget uniqueness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *root consumer has exactly the displayed budget route* $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N$*. The route is the one obtained by the source-window exhaustion, threshold stability, nonescape package, seal route, and budget-selector handoff. No second downstream budget, quotient stream, selected limit, host equality, independent tail oracle, or ambient completeness witness is available to the consumer.*

Proof. Start with the downstream unblock package. It displays the public root route as $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q$ and records that the accepted packet exports only the named rows.

Apply Definition 560.84 and Definition 560.85. The source windows, modulus rows, threshold, and comparison row are forced by the accepted TailMeet packet before a downstream root request can be served.

Apply Definition 560.86 and Definition 560.87. These attach the same request to the displayed hsame, Cont, ledger, package, and local name rows and prevent a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer from leaving that route.

Finally use Definition 560.89. A downstream budget selector can act only after reading the already displayed route, so any root consumer budget is the route just constructed. The nonescape exclusions remove every alternative carrier for a second budget. □

Theorem 560.91 (RegularCauchyTailMeet root budget route lock)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*, or* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *consumer preserving the displayed rows must use the single route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N.$$

No selected limit, quotient stream, host equality, trichotomy row, independent tail oracle, or ambient completeness row can supply a second route.

Proof. Start with the downstream window exhaustion and threshold stability theorems. Definition 560.84 fixes the paired source windows and modulus rows, while Definition 560.85 keeps the shared threshold and comparison row on the same accepted packet.

Next apply the nonescape package and seal route. Definition 560.86 removes every unlisted root resource, and Definition 560.87 attaches the consumer to the displayed hsame, Cont, ledger, package, and local naming rows.

Now use the budget-selector handoff together with root consumer budget uniqueness. Definition 560.89 forces $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ and $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ consumers through the displayed route, and Definition 560.90 gives the same route to $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ root consumers.

Thus every listed consumer reads exactly $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, \mathsf{Pkg}, N$. The excluded rows have already been removed from the packet surface, so none can witness a second route. □

Theorem 560.92 (RegularCauchyTailMeet seal consumer tail-fiber factorization)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, every seal-facing consumer that later reads the* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *row of chapter 893 must first factor through the TailMeet shared threshold tuple* $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L$ *Only after that finite route is read may the tail-fiber row be made available. The factorization exports no quotient stream, selected limit, host equality, independent tail oracle, or ambient completeness witness.*

Proof. Begin with the root consumer budget uniqueness theorem. A seal-facing downstream consumer has a unique finite TailMeet route, and that route is the displayed tuple $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L$

Use the seal route and the budget-selector handoff. They force the consumer to read the same threshold tuple before any downstream budget or seal-facing read is available.

Now pass to the tail-fiber chapter named in chapter 893. A $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ row can be consumed only as a later row attached to the already routed TailMeet threshold service, because the present packet exports no independent tail oracle and no ambient completeness witness.

The tail-fiber consumer therefore factors through the TailMeet threshold tuple before the tail-fiber row is visible. Since the intermediate route contains no quotient stream, selected limit, host equality, independent tail oracle, or ambient completeness witness, none of those objects can be supplied by the factorization. □

Theorem 560.93 (RegularCauchyTailMeet tail-fiber seal exactness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose seal-facing consumer passes through* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$*, the TailMeet threshold route and the TailFiber paired-window route have the same finite support through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the two* $\mathsf{StreamName}^{\uparrow}$ *windows, the two dyadic ledger rows, the shared fiber or threshold row, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. Conversely, any accepted TailFiber paired-window seal route supplies the same TailMeet seal-consumer entrance. The exactness statement exports no quotient streams, selected limits, host real equality, hidden tail oracles, or ambient completeness.*

Proof. For the forward direction, start with Definition 560.92. It forces the seal-facing consumer through the displayed TailMeet threshold tuple before the $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ row is available.

Apply root consumer budget uniqueness together with the TailMeet root downstream seal route. These identify the source windows, threshold data, continuation rows, package row, and local name row as the only TailMeet entrance for a seal-facing read.

Now use TailFiber paired-window exactness and TailFiber seal-consumer exhaustion. The TailFiber route reads the two regular-rational source rows, two stream windows, two dyadic ledgers, shared fiber row, seal row, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows, and no further consumer coordinate is present.

For the converse direction, begin with an accepted TailFiber paired-window seal route. TailFiber paired-window exactness supplies the common finite support up to the shared fiber row, and seal-consumer exhaustion keeps every listed TailMeet, Real-tail-agreement, and Real-facing consumer inside the same packet.

Re-enter TailMeet through the seal-consumer factorization theorem. The TailFiber route therefore supplies exactly the TailMeet seal-consumer entrance already fixed by root consumer budget uniqueness and the TailMeet seal route. The exclusions in the TailMeet nonescape package and TailFiber ledger exhaustion rule out quotient streams, selected limits, host real equality, hidden tail oracles, and ambient completeness. □

Theorem 560.94 (RegularCauchyTailMeet budget window service lock)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, every* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*,* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Completion}^{\uparrow}$ *consumer that requests the TailMeet threshold must first read the paired* $\mathsf{StreamName}^{\uparrow}$ *windows* W_0, W_1*, modulus rows* M_0, M_1*, shared threshold* τ*, comparison row* Q*, transport rows* H, C*, ledger* L*, package row* Pkg*, and local* $\mathsf{NameCert}$ *row* N*. The service is locked to this displayed finite route and cannot be replaced by an uncapped observation request, quotient stream, selected limit, host equality, trichotomy row, independent tail oracle, or ambient completeness witness.*

Proof. Start with window exhaustion. Any threshold-facing consumer must expose the paired StreamName windows W_0, W_1 before τ is available, and the same source-window route keeps the request inside the accepted TailMeet packet.

Apply threshold stability and the downstream unblock package. They attach the modulus rows M_0, M_1, the shared threshold τ, the comparison row Q, the transport rows H, C, the ledger L, the package row Pkg, and the local name row N to the same finite consumer-facing route.

Use the seal route for $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers and the budget-selector handoff for $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ and $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ consumers. Both routes consume the displayed tuple before a threshold service is available.

The remaining non-escape claim is ledger exhaustion for that tuple. Since the route contains only the displayed windows, modulus rows, threshold, comparison, transport, ledger, package, and local name, no uncapped observation request, quotient stream, selected limit, host equality,

trichotomy row, independent tail oracle, or ambient completeness witness can replace the service. □

Theorem 560.95 (RegularCauchyTailMeet quotient-refusal factorization)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet that requests quotient-stream equality must first pass through the finite TailMeet source-window route and then through the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *route of chapter 998. If either route is missing, the request is routed to the refusal ledger. The handoff exports no selected limit, quotient stream equality, host real equality, trichotomy, independent tail oracle, or ambient completeness witness.*

Proof. Begin with the downstream unblock package. It exposes the paired source windows R_0, R_1, W_0, W_1, modulus rows M_0, M_1, shared threshold τ, comparison row Q, and structural rows H, C, L, Pkg, N before any completion-facing read is available.

Apply the threshold stability and budget-selector handoff theorems. They keep the selected downstream budget attached to the same finite TailMeet packet and prevent a selector from bypassing the source windows, modulus rows, threshold, comparison, ledger, package, or local name.

Next use the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ non-escape route of chapter 998. A quotient-stream equality request has a route only after it is translated into the displayed StreamName, RegSeqRat, CauchyLimitSeal, and RealUp seal rows; an off-route request is sent to the refusal ledger.

The two routes therefore factor the completion-facing consumer through the TailMeet finite source-window surface and then through the quotient-refusal surface. Since neither packet contains selected limits, quotient stream equality, host real equality, trichotomy, independent tail oracles, or ambient completeness, none of those objects is exported. □

Theorem 560.96 (RegularCauchyTailMeet root downstream budget-seal package)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, every root threshold route exported to* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{CauchyModulusRefinement}^{\uparrow}$*, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *reaches a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal read only after the route has passed through the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The route then passes through the finite observation-budget surface of chapter 611 and the limiter cap surface of chapter 903. Only after those rows are displayed may refinement or limit-seal consumers read the Real or completion seal surfaces of chapter 885 and chapter 386. The package exports no quotient stream, selected limit, host real equality, trichotomy row, hidden selector, uncapped observation schedule, independent completion carrier, or ambient completeness principle.

Proof. Start with the root budget-limiter factorization. By Definition 560.79, any route to $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, or $\mathsf{CauchyLimitSeal}^{\uparrow}$ first reads the TailMeet source, window, modulus, threshold, comparison, transport, continuation, ledger, and local naming rows.

Next use the budgeted refinement seal factorization. Definition 560.78 places the finite observation-budget entrance and the capped limiter service before the refinement and limit-seal consumers can reach a Real or completion seal.

The sibling chapter surfaces supply the downstream boundary. The observation-budget and limiter chapters expose only finite budget, window, readback, cap, service, transport, continuation, provenance, and local $\mathsf{NameCert}$ rows, while the refinement and limit-seal chapters read their Real and completion-facing seals through those finite rows.

Finally apply the root window threshold obligations from Definition 560.72, Definition 560.75, and Definition 560.77. These keep every consumer on the displayed route and leave no carrier

coordinate for quotient streams, selected limits, host real equality, trichotomy, hidden selectors, uncapped schedules, independent completion carriers, or ambient completeness. □

Theorem 560.97 (RegularCauchyTailMeet root seal exactness)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal read exported from a root threshold route to a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer is exact with respect to the displayed route: it must use the same TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

that appear in the root downstream budget-seal package, then the finite observation-budget surface, then the limiter cap surface, before the consumer may read the refinement or limit-seal interface. No downstream consumer may replace those rows by an independent threshold, budget, cap, Real carrier, completion carrier, quotient stream, selected limit, hidden selector, or ambient completeness witness.

Proof. Begin with the package statement Definition 560.96. It already places every exported route to $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, or $\mathsf{CauchyLimitSeal}^{\uparrow}$ on the displayed TailMeet rows before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read is available.

Next fix the route through the budget and limiter interfaces. The root budget-limiter factorization Definition 560.79 keeps the root threshold route on the TailMeet packet and then passes it through the limiter service, while Definition 560.78 places the refinement and limit-seal consumers after that finite budgeted route.

It remains to identify the rows read by the downstream seal. The observation-budget entry theorem Definition 560.72 supplies the finite budget entrance from the same root rows, and Definition 560.75 synchronizes the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal reads with that entrance.

Finally Definition 560.77 synchronizes the limiter cap with the same route. Hence every downstream seal consumer reads exactly the displayed TailMeet rows, followed by the finite observation-budget and limiter cap surfaces, and there is no remaining coordinate from which an independent analytic or completion-side seal can be obtained. □

Theorem 560.98 (RegularCauchyTailMeet downstream budget source lock)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, every route to* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *uses the same displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

No downstream route may substitute a second threshold source, independent budget row, independent limiter cap, quotient stream, selected limit, hidden selector, or ambient completeness witness for those rows.

Proof. Begin with the root downstream budget-seal package of Definition 560.96. It places all four downstream consumers on the same TailMeet source, window, modulus, threshold, comparison, transport, continuation, ledger, and naming rows.

Next use Definition 560.79. The factorization keeps the observation-budget and limiter entrances attached to that same root packet rather than to a second schedule or external completion carrier.

Finally apply the root seal exactness theorem Definition 560.97. Since every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read is exact with respect to the displayed route, no independent downstream source can replace the listed rows. □

Theorem 560.99 (RegularCauchyTailMeet seal package non-escape)**.** *The downstream budget-seal package for an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet exports no quotient stream, selected limit, uncapped observation schedule, independent completion carrier, host real equality, trichotomy row, hidden selector, or ambient completeness principle. The only seal-facing surface is the displayed TailMeet route followed by the finite observation-budget surface, the limiter cap surface, and the refinement or limit-seal consumer interface.*

Proof. First apply Definition 560.98. It locks every downstream budget, limiter, refinement, and seal route to the displayed TailMeet rows.

Next use Definition 560.78. This places the finite observation budget and capped limiter service before either $\mathsf{CauchyModulusRefinement}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ may expose a Real or completion-facing seal.

The sibling chapters then account for the endpoint rows. The Real observation-budget, observation-budget limiter, Cauchy modulus-refinement, and CauchyLimitSeal surfaces cited in Definition 560.96 provide finite budget, cap, refinement, and seal routes, not quotient streams, selected limits, uncapped schedules, independent completion carriers, or ambient completeness. □

Theorem 560.100 (RegularCauchyTailMeet window threshold totality)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the threshold row* τ *together with the limiter cap surface exhausts every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$*-facing window request exported through* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{CauchyModulusRefinem}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*. Such a request must pass through the displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and then through the finite observation-budget and limiter cap rows before any refinement or limit-seal endpoint is read.

Proof. Start with Definition 560.97. It identifies every exported Real or completion seal read with the same TailMeet route, followed by the finite observation-budget and limiter cap surfaces.

Apply the downstream budget source lock of Definition 560.98. This keeps the threshold row τ and comparison row Q attached to the displayed windows, moduli, transport, continuation, ledger, and naming rows.

Finally use the seal package non-escape theorem Definition 560.99. Since no uncapped schedule, independent completion carrier, quotient stream, selected limit, hidden selector, or ambient completeness datum is exported, the threshold row and limiter cap exhaust the whole window request budget. □

Theorem 560.101 (RegularCauchyTailMeet budget-seal finality)**.** *For an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, any downstream route to* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{CauchyModulusRefinement}^{\uparrow}$*, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is final for a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal consumer exactly when it reads the displayed TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

then the finite observation-budget surface, and then the limiter cap before reaching the refinement or limit-seal endpoint. No route that omits one of those stages is a final budget-seal route, and no final route introduces a quotient stream, selected limit, host real equality, uncapped schedule, independent completion carrier, hidden selector, or ambient completeness principle.

Proof. First compose the package route with seal exactness. Definition 560.96 places the four downstream consumers after the displayed TailMeet packet, and Definition 560.97 identifies every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read with that same route.

Next lock the source of the budget route. By Definition 560.98, none of the downstream consumers may substitute an independent threshold source, budget row, limiter cap, quotient stream, selected limit, hidden selector, or ambient completeness witness for the listed TailMeet rows.

Now use non-escape and window-threshold totality. Definition 560.99 rules out every off-route seal coordinate, while Definition 560.100 says the threshold row together with the limiter cap exhausts every Real- or completion-facing window request.

The forward direction follows because a final downstream seal route has no remaining coordinate after these displayed stages. Conversely, a route that reads the TailMeet rows, then the observation-budget surface, then the limiter cap, and only then the refinement or limit-seal endpoint has passed every required boundary named above, so it is already final for the budget-seal consumer. □

Theorem 560.102 (RegularCauchyTailMeet observation-budget factorization)**.** *Any* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold must first factor its observation request through the TailMeet rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

before it reaches the observation-budget rows. Thus the budget consumer sees the paired $\mathsf{RegularCauchyName}^{\uparrow}$ *sources, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, source modulus rows, shared threshold, comparison row, componentwise transports, displayed continuations, finite ledger, provenance, and local* $\mathsf{NameCert}$ *row as the complete entrance to the finite observation-budget surface of chapter 611.*

Proof. Start with the finite observation-budget route of chapter 611. A budget consumer can inspect only the precision request, finite window family, dyadic readbacks, regular-sequence handoff, real-seal boundary, transports, continuations, provenance, and local naming rows supplied by its accepted packet.

Next read the TailMeet threshold through shared-threshold non-escape. By Definition 560.50, every admitted shared-threshold read remains inside the displayed threshold, paired stream schedules, source modulus rows, comparison row, componentwise transports, continuation routes, provenance, and local $\mathsf{NameCert}$ evidence.

Finally apply real-seal route totality. Definition 560.51 places every admitted $\mathsf{Real}^{\uparrow}$ seal read behind the same TailMeet window and dyadic endpoint rows, so the observation-budget consumer cannot reach its budget rows before the TailMeet route has exposed $R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$. □

Theorem 560.103 (RegularCauchyTailMeet selector-budget compatibility)**.** *Any* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *route that shares the accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold consumes that threshold through the selector budget's finite-window and* $\mathsf{RegSeqRat}^{\uparrow}$ *route. It does not introduce a second selector, a hidden threshold, or an independent tail oracle outside the displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

Proof. Read the selector-budget side through chapter 678. Its finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ rows specify the admissible selector request before any downstream tail read is exported.

Now compose with shared-threshold route determinacy. By Definition 560.56, two shared-threshold reads with the same transported source rows, window rows, and source modulus rows reach the same displayed dyadic threshold ledger.

The selector-budget route therefore consumes the TailMeet threshold through the already displayed finite-window and regular-rational route. Since determinacy leaves no second threshold

coordinate and the selector budget supplies no extra selector outside its finite route, the shared threshold cannot be replaced by a hidden threshold or independent tail oracle. □

Theorem 560.104 (RegularCauchyTailMeet real-tail agreement consumer surface)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet receives exactly the shared threshold* τ*, paired comparison row* Q*, componentwise transports* H*, displayed continuations* C*, package provenance, finite ledger* L*, and local* $\mathsf{NameCert}$ *row* N*, together with the paired source, window, and modulus rows that produce* τ*.*

The consumer surface is therefore the finite TailMeet route

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

followed by the seal route of chapter 699. It includes no quotient stream, selected limit, host equality, independent tail oracle, or ambient completeness row.

Proof. First use the observation-budget factorization. Definition 560.102 forces every real-facing observation of the TailMeet threshold through the displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, provenance, and local naming rows before any budget-side read is available.

Next use selector-budget compatibility. Definition 560.103 keeps selector-facing reads on the same finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ route to the shared threshold, so the real-tail agreement consumer receives the same τ and Q rather than a second selector or hidden tail threshold.

Finally read the seal side through chapter 699. The agreement seal may consume the shared threshold, paired comparison, transports, continuations, provenance, ledger, and local naming evidence already displayed by the TailMeet packet, but the TailMeet carrier and the two preceding consumer theorems provide no coordinate for quotient streams, selected limits, host equality, independent tail oracles, or ambient completeness. □

Theorem 560.105 (RegularCauchyTailMeet budget-family cofinality)**.** *For any finite family of downstream budget requests accepted through a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, the shared threshold row remains cofinal. Every requested strengthening factors through the same finite route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N$$

and a stronger displayed dyadic threshold can be chosen inside that route for all members of the family at once. No request introduces a quotient stream, selected limit, host equality, independent tail oracle, trichotomy row, or ambient completeness coordinate.

Proof. Start with window-source coverage and source-modulus exhaustion. Definition 560.26 and Definition 560.27 force each family request through the two source rows, two window rows, and two source modulus rows before it reaches τ.

Use shared-threshold cofinality for the finite family. Since Definition 560.8 refines the displayed threshold without leaving the carrier route, induction over the finite family yields one stronger displayed threshold row still carried by the same ledger.

Finally apply ledger exactness and the public shared-threshold certificate. Definition 560.7 and Definition 560.11 keep the refined family threshold inside the finite certificate surface, so none of the excluded completion or quotient data can be exported. □

Theorem 560.106 (RegularCauchyTailMeet finite witness induction)**.** *Let a finite list of downstream requests enter through an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet, with each entry either a*

$\mathsf{RealObservationBudget}^{\uparrow}$ *request or a* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *request admitted by the packet. Then the whole list admits one displayed dyadic threshold row inside the same TailMeet route*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The witness is obtained without adding a quotient stream, selected limit, host equality row, independent tail oracle, trichotomy coordinate, or ambient completeness premise.

Proof. For the empty list, use the displayed accepted packet itself. Its public shared-threshold certificate already carries τ together with $R_0, R_1, W_0, W_1, M_0, M_1, Q, H, C, L, N$, so the required witness is the original finite row.

For a nonempty list, assume by induction that the tail of the list has one displayed threshold row inside the same TailMeet route. The head request is either an observation-budget request or a selector-budget request.

In the observation-budget case, apply Definition 560.102; in the selector-budget case, apply Definition 560.109. In both cases the head request reaches the shared threshold only through the displayed source, window, modulus, comparison, transport, continuation, package, ledger, and local naming rows.

Now use Definition 560.105 to refine the induction witness and the head request to one stronger displayed threshold. Source-window-modulus route determinacy, Definition 560.52 and Definition 560.56, keeps that refined row on the same TailMeet route rather than on a second threshold coordinate.

Finally apply ledger exactness and the public shared-threshold certificate. Definition 560.7 and Definition 560.11 exhaust the finite ledger surface for the refined witness, so the induction step has no coordinate from which to export quotient, selected-limit, host-equality, independent-tail, trichotomy, or completeness data. □

Theorem 560.107 (RegularCauchyTailMeet selector-budget absorption)**.** *Every* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *request that enters through an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet is absorbed by the displayed rows*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The selector-budget side may choose finite windows and regular-rational readbacks, but the tail-meet threshold it consumes is still the shared threshold row produced by the paired source schedules, source modulus rows, ledger exactness, public certificate, and seal-compatible handoff.

Proof. Read the selector-budget request through selector admission and selector-budget compatibility. Definition 560.39 supplies the admitted selector route, and Definition 560.103 keeps that route on the selector budget's finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ surface.

Next identify the threshold route. Window-source coverage, source-modulus exhaustion, and shared-threshold route determinacy force the selector-budget read through $R_0, R_1, W_0, W_1, M_0, M_1, \tau$ rather than through a second selector or hidden threshold.

The structural rows are then absorbed by consumer exactness and seal compatibility. Definition 560.34 and Definition 560.43 add exactly Q, H, C, L, N to the route and exclude quotient streams, selected limits, host equality, independent tail oracles, trichotomy, and ambient completeness. □

Theorem 560.108 (RegularCauchyTailMeet diagonal seal-budget exactness)**.** *Every diagonal* $\mathsf{Real}^{\uparrow}$ *seal-budget read of an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet passes through the same finite shared-threshold route used by the public certificate:*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The diagonal seal may read the real-facing handoff only after the paired source schedules, source modulus rows, cofinal shared threshold, exact ledger, public shared-threshold certificate, and seal compatibility rows have all been exposed.

Proof. Begin with the diagonal-facing coverage. Definition 560.10 and Definition 560.49 route the diagonal read through the displayed source schedules and window rows.

Attach the modulus and threshold rows. Source-modulus exhaustion, shared-threshold cofinality, and source-window-modulus route results identify the only admissible path to τ and keep stronger diagonal precision requests cofinal inside the same finite ledger.

Now read the seal side. Definition 560.7, Definition 560.11, and Definition 560.43 make the diagonal seal consume exactly the listed threshold, comparison, transport, continuation, ledger, provenance, and naming rows, with no additional completion premise. □

Theorem 560.109 (RegularCauchyTailMeet selector-budget shared entry)**.** *Every* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *consumer that enters through an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet reads the same shared threshold row* τ *through*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

The selector-budget request may choose its finite selector windows and $\mathsf{RegSeqRat}^{\uparrow}$ *readback, but its tail-facing threshold entry is the TailMeet shared threshold ledger row, transported with the displayed comparison,* hsame *rows, continuations, package provenance, and local* NameCert *evidence.*

Proof. Start from selector-budget compatibility. Definition 560.103 routes the admitted selector request through the finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ surface before any tail threshold is consumed.

Next use selector-budget absorption. Definition 560.107 identifies the absorbed TailMeet route as the displayed source schedules, source modulus rows, shared threshold, comparison row, transports, continuations, ledger, provenance, and local name.

Finally apply shared-threshold route determinacy. Definition 560.56 rules out a second threshold coordinate for the same transported source-window-modulus route, so the selector-budget consumer and the TailMeet packet read the same τ. □

Theorem 560.110 (RegularCauchyTailMeet budget-family row exhaustion)**.** *Let a finite family of downstream observation requests or* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *requests enter through an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. Then every family member factors through the TailMeet carrier and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *surface. No member contributes a row outside*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N.$$

In particular, the family cannot add a quotient stream, selected limit, host equality row, independent tail oracle, trichotomy coordinate, or ambient completeness witness.

Proof. First use budget-family cofinality. Definition 560.105 gives one stronger displayed threshold inside the same TailMeet route for all finite family members.

Observation requests factor through Definition 560.102. Selector-budget requests factor through Definition 560.109. Both routes therefore meet at the same shared threshold row and retain the same comparison, transport, continuation, package, and naming rows.

It remains to exclude extra family rows. Definition 560.7 exhausts the finite ledger surface, so the family has no coordinate from which to export quotient, selected-limit, host-equality, independent-tail, trichotomy, or completeness data. □

Theorem 560.111 (RegularCauchyTailMeet real-tail threshold ledger)**.** *Every* RealTailAgreementSeal$^{\uparrow}$ *read of an accepted* RegularCauchyTailMeet$^{\uparrow}$ *packet consumes exactly the TailMeet shared threshold ledger before the real-tail seal route is available. The consumed rows are*

$$R_0, R_1, W_0, W_1, M_0, M_1, \tau, Q, H, C, L, N,$$

with τ supplied by the public shared-threshold certificate and transported through the displayed hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Proof. Begin with the real-tail agreement consumer surface. Definition 560.104 lists exactly the TailMeet rows visible to the RealTailAgreementSeal$^{\uparrow}$ side and gives no separate threshold oracle.

Next apply diagonal seal-budget exactness. Definition 560.108 forces a real-facing seal-budget read through the displayed source schedules, source modulus rows, shared threshold, comparison, transports, continuations, ledger, provenance, and local name before the seal route is read.

Finally use the public shared-threshold certificate. Definition 560.11 identifies the real-tail threshold with the same public τ, so the agreement seal consumes exactly the TailMeet ledger and cannot introduce an additional real-tail threshold row. □

Closure status for NameCert$_{\text{RegularCauchyTailMeet}^{\uparrow}}$.

Constructive story: A RegularCauchyTailMeet$^{\uparrow}$ packet is generated from two RegularCauchyName$^{\uparrow}$ source rows, paired StreamName$^{\uparrow}$ schedules, source modulus rows, one shared DyadicRatCore$^{\uparrow}$ tail-threshold ledger, a paired comparison row, componentwise hsame transports, displayed Cont routes, finite ledger, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailMeet}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 560.1, Definition 560.2, Definition 560.3, Definition 560.4, Definition 560.5, Definition 560.6, Definition 560.7, Definition 560.8, Definition 560.9, Definition 560.10, Definition 560.11, Definition 560.26, Definition 560.33, Definition 560.34, Definition 560.35, Definition 560.36, Definition 560.37, Definition 560.38, Definition 560.39, Definition 560.40, Definition 560.41, Definition 560.42, Definition 560.43, Definition 560.44, Definition 560.45, Definition 560.46, Definition 560.47, Definition 560.48, Definition 560.49, Definition 560.50, Definition 560.51, Definition 560.52, Definition 560.56, Definition 560.57, Definition 560.102, Definition 560.103, Definition 560.104, Definition 560.105, Definition 560.106, Definition 560.107, Definition 560.108, Definition 560.109, Definition 560.110, and Definition 560.111 bind the finite shared-tail carrier, paired schedules, source modulus reads, dyadic threshold ledger, dyadic absolute-value endpoint route, dyadic endpoint meet, cofinal threshold refinement, comparison row, transport, continuation, package, real-seal handoff, obligation package, public shared-threshold certificate, window-source coverage, shared-threshold transport, consumer exactness, diagonal consumer non-escape, obligation-closure readiness, bridge package, representative downstream consumers, selector admission, source scope, classifier scope, ledger scope, seal compatibility, root consumer package, root carrier admission, root classifier stability, root ledger exactness, uniform-criterion consumer link, window-source exhaustion, shared-threshold non-escape, real-seal route totality, source-window-modulus route, route determinacy, public tail-meet certificate, observation-budget factorization, selector-budget compatibility, real-tail agreement consumer surface, budget-family cofinality, finite witness induction, selector-budget absorption, diagonal seal-budget exactness, selector-budget shared entry, budget-family row exhaustion, real-tail threshold ledger, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailMeet}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RegularCauchyTailMeetUp.RegularCauchyTailMeetPacket\_root\_formal\_handoff\_surface`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, host real equality, ambient completeness, or public bridge checking.

Upgrade path: No further formal-axis bridge step remains for this shared-tail packet.

Chapter 561

A Concrete Naming Certificate for $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$

The $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ interface records the finite selected-window route used when Real completeness arguments pass from a precision request to a visible diagonal observation and then to a $\mathsf{Real}^{\uparrow}$ seal. It sits over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, and it exposes only the rows needed by the diagonal-window consumer.

This packet consumes the finite-window schedule of chapter 126, the regular-rational readback of chapter 301, the dyadic observation ledger of chapter 291, and the seal boundary of chapter 38 before exposing its diagonal-window coverage row.

Definition 561.1 (Real diagonal-window coverage carrier)**.** *A* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (q, w, r, d, e, h, c, p, n)$$

with the following visible coordinates:

1. *q is the rational precision request, read through the displayed* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows;*
2. *w is the selected* $\mathsf{StreamName}^{\uparrow}$ *window that exposes the finite diagonal observation interval requested by q;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the rational observation selected by w;*
4. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the endpoint, radius, comparison, and precision rows used by the readback;*
5. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after q, w, r, d have been exposed;*
6. *h stores componentwise* hsame *transports for the precision request, selected window, readback row, dyadic ledger, seal row, provenance, and local name;*
7. *c stores the displayed* Cont *routes from q to w, from w to r, from r to d, and from d to the seal row e;*
8. *p is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *n is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCoverage}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no host real equality, no unbounded diagonal stream, no ambient completeness theorem, and no selected limit outside the displayed seal row.

Lean4 ✓ `BEDC.Derived.RealDiagonalWindowCoverageUp.RealDiagonalWindowCoverageTasteGate single carrier alignment`

Definition 561.2 (Real diagonal-window coverage classifier)**.** *For accepted carriers*

$$G = (q, w, r, d, e, h, c, p, n) \quad \textit{and} \quad G' = (q', w', r', d', e', h', c', p', n'),$$

the classifier $\mathsf{RealDiagonalWindowCoverageClassifier}(G, G')$ *compares only the displayed precision request, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It never compares host streams extensionally and never identifies completed reals by quotient equality.*

Theorem 561.3 (Real diagonal-window coverage NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCovera}}$ *surface for Definition 561.1 consists exactly of carrier admission for* G*, precision-request exposure through* q*, selected-window coverage through* w*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* r*, dyadic observation-ledger exactness through* d*,* $\mathsf{Real}^{\uparrow}$ *seal handoff through* e*, componentwise* hsame *transport through* h*, displayed* Cont *route replay through* c*,* Pkg *provenance through* p*, and local naming through* n*.*

Proof. Carrier admission is projection from the displayed packet G. The rows $q, w, r, d, e, h, c, p, n$ list the precision request, selected window, regular rational readback, dyadic observation ledger, real seal, transports, routes, package provenance, and local name.

The selected-window obligation is finite. A consumer reads q first, then uses w to inspect the requested diagonal interval, and then reads the rational observation through r and the dyadic ledger through d.

The remaining rows account for transport, route replay, provenance, and naming. Since no coordinate records quotient streams, host real equality, an unbounded diagonal stream, ambient completeness, or a selected external limit, the local NameCert obligation surface is exhausted by the displayed packet. □

Theorem 561.4 (Real diagonal-window coverage carrier admission)**.** *An accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet exposes exactly the carrier rows*

$$q, w, r, d, e, h, c, p, n.$$

These rows are, respectively, the precision request, selected $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Read Definition 561.1. Carrier admission is projection from the displayed tuple $G = (q, w, r, d, e, h, c, p, n)$.

Apply the NameCert obligations of Definition 561.3. They name each coordinate as part of the local certificate surface and provide no additional row beyond the displayed tuple.

The carrier definition excludes quotient streams, host real equality, unbounded diagonal streams, ambient completeness, and selected external limits, so the accepted packet exposes exactly the nine listed rows. □

Lean4 ✓ `BEDC.Derived.RealDiagonalWindowCoverageUp.RealDiagonalWindowCoverageCarrier carrier admission`

Theorem 561.5 (Real diagonal-window coverage induction)**.** *Every seal-facing consumer of an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *carrier* $G = (q, w, r, d, e, h, c, p, n)$ *is generated by finite induction along* q*, then* w*, then* r*, then* d*, then* e*, with* h, c, p, n *preserving the same displayed route.*

Proof. Use Definition 561.4 to expose the nine carrier rows. The base read is the precision request q, which has no access to a stream quotient, host real equality, selected limit, or completeness principle.

Induct over the finite selected-window observation ledger. A successor step either advances from the precision request q to the selected $\mathsf{StreamName}^{\uparrow}$ window w, from w to the $\mathsf{RegSeqRat}^{\uparrow}$ readback row r, from r to the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d, or from d to the $\mathsf{Real}^{\uparrow}$ seal e.

The remaining rows h, c, p, n do not create a second consumer path. They transport, replay, record provenance, and name the same route, so no quotient stream, host real equality, unbounded diagonal stream, selected limit, or ambient completeness can appear. □

Theorem 561.6 (Real diagonal-window coverage classifier stability)**.** *For accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packets, classifier comparison is stable under componentwise* hsame *transport of the precision request, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

Proof. Unfold the classifier in Definition 561.2. Its comparison fields are exactly the displayed precision request, window, readback, dyadic ledger, seal row, transports, routes, provenance, and local name.

Use the h coordinate of the carrier to transport each component. The same rows are the rows exposed by carrier admission, so no comparison field is lost or replaced during transport.

The RealUp handoff theorem keeps seal-facing comparisons on the displayed route through q, w, r, d, h, c, p, n. Hence classifier stability never appeals to host stream equality, quotient equality of completed reals, or an undisplayed completeness principle. □

Theorem 561.7 (Real diagonal-window coverage selected-window coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet reaches the seal row* e *only after the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger* d *have been displayed.*

Proof. Start from the carrier definition. The $\mathsf{Real}^{\uparrow}$ seal row e is listed after the precision request q, selected window w, $\mathsf{RegSeqRat}^{\uparrow}$ readback r, and dyadic ledger d.

Apply Definition 561.3. The selected-window and dyadic observation-ledger obligations are part of the local certificate surface before any seal-facing consumer is exported.

Now use Definition 561.8. It routes every $\mathsf{Real}^{\uparrow}$ completion or continuous-map consumer through the displayed selected-window and dyadic-ledger rows before the seal row is read. □

Theorem 561.8 (Real diagonal-window coverage RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion or continuous-map consumer of an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *carrier receives the seal row* e *only through the displayed route*

$$q, w, r, d, h, c, p, n.$$

Thus the handoff consumes a rational precision request, a selected $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, componentwise transport, route replay, package provenance, and local naming before the* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Lean4 ✓ `BEDC.Derived.RealDiagonalWindowCoverageUp.RealDiagonalWindowCoverageCarrier realup handoff`

Proof. Start with the carrier row e. It is the only $\mathsf{Real}^{\uparrow}$ seal coordinate in the finite packet.

Apply the NameCert obligations. They force a seal-facing read to pass through the precision request q, the selected window w, the readback row r, and the dyadic ledger d before the seal is consumed.

The rows h and c replay the displayed transport and continuation data, while p and n provide provenance and local naming. Because the carrier lists no host equality, quotient equality, ambient completeness, or external limit coordinate, the real-facing handoff is exactly the displayed route. □

Theorem 561.9 (Real diagonal-window coverage non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet factors through the finite rows*

$$q, w, r, d, e, h, c, p, n.$$

The consumer may read the rational precision request, selected $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *source observation,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and it cannot read quotient streams, host real equality, unbounded diagonal streams, ambient completeness, or a choice-selected limit.*

Proof. Project the accepted carrier and the local NameCert surface. The rows $q, w, r, d, e, h, c, p, n$ are the complete displayed packet.

Use the classifier of Definition 561.2. It compares only those rows under the stored transport and route data, so a transported consumer remains on the same finite window-read route.

Use the RealUp handoff theorem for seal-facing consumers. It routes every completion or continuous-map read through q, w, r, d, h, c, p, n before e is consumed.

Since the carrier has no remaining coordinate beyond the displayed rows, no downstream consumer can extract quotient-stream data, host real equality, unbounded diagonal data, ambient completeness, or a choice-selected limit from the packet. □

Theorem 561.10 (Real diagonal-window tail-meet triangle handoff)**.** *Let* $G = (q, w, r, d, e, h, c, p, n)$ *be an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet. Suppose its selected window row* w *is supplied by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet, and suppose its dyadic observation ledger* d *is supplied by an accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *three-leg estimate. Then every* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer of* G *factors through the finite route*

$$q \to w \to r \to d \to e$$

together with the tail-meet shared threshold, the triangle-bound tolerance ledger, componentwise hsame *rows, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *rows of the three packets.*

The handoff exports no quotient stream, selected limit, host real equality, ambient completeness principle, or hidden tail oracle.

Proof. Begin with the RealUp handoff of Definition 561.8. It forces every seal-facing read of G through the displayed precision request, selected window, readback row, dyadic ledger, transport rows, continuation routes, provenance, and local name before e is consumed.

Next read the selected window through the shared tail packet. Apply Definition 560.9; its shared-threshold surface supplies the paired regular-Cauchy tails, source schedules, source modulus

rows, threshold ledger, comparison row, transports, continuations, provenance, and local tail-meet name.

Now read the dyadic observation ledger through the triangle-bound packet. Apply Definition 466.8 for the three-leg estimate and Definition 466.7 for its consumer boundary.

Composing the displayed Cont routes gives the route $q \to w \to r \to d \to e$. Each leg is already guarded by componentwise hsame, Pkg provenance, and a local NameCert row, and the cited non-escape boundaries leave no coordinate for quotient streams, selected limits, host real equality, ambient completeness, or a hidden tail oracle. □

Theorem 561.11 (Real diagonal-window coverage kernel scope)**.** *The carrier, classifier, local* $\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCoverage}^{\uparrow}}$ *obligations, selected-window coverage,* $\mathsf{Real}^{\uparrow}$ *handoff, non-escape boundary, and tail-meet triangle handoff for an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet are all routed through exactly the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No step in this kernel scope adds a quotient-stream coordinate, host real equality, unbounded diagonal stream, selected limit, or ambient completeness coordinate.*

Proof. Start from Definition 561.1 and Definition 561.2. They display the packet $G = (q, w, r, d, e, h, c, p, n)$ and restrict classifier comparison to the same precision request, selected window, readback row, dyadic ledger, seal row, transports, continuations, provenance, and local name.

Apply Definition 561.3 for the local certificate surface. The selected-window coverage theorem then forces every seal-facing read to display w and d before the $\mathsf{Real}^{\uparrow}$ row e is consumed.

Use Definition 561.8 and Definition 561.9. They keep completion and downstream consumers on the finite route through $q, w, r, d, e, h, c, p, n$ and exclude every undisplayed real or stream coordinate.

Finally, Definition 561.10 composes this route with the cited $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ packets. Those packets are consumed through their displayed threshold and estimate rows, so the full kernel dependency scope is precisely the listed BEDC rows. □

Theorem 561.12 (Real diagonal-window coverage selected-window source lock)**.** *For an accepted* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *packet* $G = (q, w, r, d, e, h, c, p, n)$*, the selected window* w *and dyadic ledger* d *determine the only source-lock route from the rational precision request* q *to the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r *and the* $\mathsf{Real}^{\uparrow}$ *seal row* e*. The source lock has no quotient stream, host equality, unbounded diagonal stream, selected limit, or ambient completeness coordinate.*

Proof. Project the carrier in Definition 561.1. The route from q to r passes through the selected $\mathsf{StreamName}^{\uparrow}$ window w, and the route from r to e passes through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d.

Apply Definition 561.3 to identify the certificate rows that expose w, r, d, and e. Then apply Definition 561.7, which prevents a $\mathsf{Real}^{\uparrow}$-facing read from reaching e before the selected window and dyadic ledger have been displayed.

The handoff theorem Definition 561.8 supplies the same route for completion and continuous-map consumers. The non-escape theorem Definition 561.9 removes every other source coordinate, so w and d are the only source-lock route to r and e. □

Theorem 561.13 (Real diagonal-window coverage tail-meet triangle scope)**.** *The tail-meet and triangle-bound handoff cited by Definition 561.10 consumes only the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *estimate packet, and the local* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *route*

$$q, w, r, d, e, h, c, p, n.$$

It exports no extra source stream, no selected limit, no host equality, no ambient completeness row, and no hidden tail oracle.

Proof. Begin with Definition 561.10. Its hypotheses name exactly one accepted selected-window packet from $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and one accepted dyadic-ledger packet from $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$.

The local $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ contribution is fixed by Definition 561.1 and Definition 561.3. These give only the rows $q, w, r, d, e, h, c, p, n$ and the corresponding local certificate surface.

The $\mathsf{Real}^{\uparrow}$ handoff and non-escape theorems keep the seal-facing consumer on the displayed route through those rows. Therefore the tail-meet and triangle-bound packets are consumed as threshold and estimate witnesses for w and d, not as a new stream, limit, host-equality, completeness, or tail-oracle coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCoverage}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ packet is generated from a rational precision request, a selected $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCoverage}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 561.1, Definition 561.2, Definition 561.3, Definition 561.7, Definition 561.8, Definition 561.9, Definition 561.10, Definition 561.11, Definition 561.12, and Definition 561.13 bind the finite selected-window route from precision request through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback to $\mathsf{DyadicRatCore}^{\uparrow}$ observations, the $\mathsf{Real}^{\uparrow}$ seal, the selected-window source lock, and the tail-meet triangle scope for downstream consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDiagonalWindowCoverage}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealDiagonalWindowCoverageUp.RealDiagonalWindowCoverageCarrier\_realup\_handoff`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, continuous-map theorems, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a checked standard bridge for the selected-window $\mathsf{Real}^{\uparrow}$ seal handoff.

Chapter 562

A Concrete Naming Certificate for $\mathsf{TypedReductionNormalForm}^{\uparrow}$

The $\mathsf{TypedReductionNormalForm}^{\uparrow}$ surface binds typed reduction to the terminal normal-form boundary as finite BEDC data. It refines the type-checking packet by recording the source membership, the visible reduction route, the target membership, and the normal-form row in one local $\mathsf{NameCert}$ object, so consumers can use the endpoint without importing host conversion or a normalization oracle.

Definition 562.1 (Typed reduction normal-form packet)**.** *A* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packet is a finite* BHist *carrier*

$$Q = (T, J, E, R, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the displayed term row containing the source window, the target window, and the terminal normal-form window;*
2. *J is the finite judgment ledger for the typing assumptions, source judgment, target judgment, and terminal judgment consumed by the packet;*
3. *E is the* Ext *membership row that records the accepted source membership and the target membership read exposed by subject reduction;*
4. *R is the* Cont *reduction route from the source term row to the terminal target row, with every intermediate step named by the packet;*
5. *F is the normal-form boundary row, stating that the terminal target has no further packet-visible typed reduction route;*
6. *H records the componentwise* hsame *transports for term rows, judgment rows, membership rows, reduction routes, normal-form rows, package rows, and local certificate rows;*
7. *C records the continuation reads for source membership, typed reduction, target membership, normal-form boundary access, and replay of the terminal packet;*
8. *P is the* Pkg *provenance over* BHist, Ext, Cont, hsame, *the judgment ledger, the normal-form boundary, and the surrounding* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for host conversion, semantic equality of terms, a global normalization theorem, an erased typing derivation, or a normalization oracle.

Lean4 (definition) `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket`

Theorem 562.2 (TypedReductionNormalForm NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$ *surface for Definition 562.1 consists of five obligations:*

1. *carrier exposure for the finite packet* $Q = (T, J, E, R, F, H, C, P, N)$*;*
2. *reduction-route exactness, saying that every typed reduction read is the displayed route* R *and its named intermediate* Cont *rows;*
3. *subject-reduction seal, saying that the target membership reached after the typed reduction is read only through* E*,* R*,* H*,* C*,* P*, and* N*;*
4. *normal-form boundary, saying that the terminal target is accepted only at the boundary row* F *and no further typed reduction route can be consumed unless it is a displayed packet row;*
5. *non-escape from host conversion, semantic equality of terms, global normalization, erased derivation semantics, or any normalization oracle not named by the packet.*

Lean4 ✓ `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket namecert obligations`

Proof. Carrier exposure is the packet inventory of Definition 562.1. The rows name the term windows, judgment ledger, membership row, reduction route, normal-form boundary, transports, continuation reads, package provenance, and local certificate.

Reduction-route exactness follows by projecting R and C. A consumer can replay the typed route only through the named Cont rows, while H supplies the permitted componentwise transports.

The subject-reduction seal uses the same rows. The source membership is read from E, the route is read from R, and the target membership is the membership coordinate exposed by the local certificate after the displayed transport and package provenance have been checked.

The normal-form boundary is the row F. Since no coordinate names a host conversion relation, semantic equality, erased derivation, global normalization theorem, or normalization oracle, the terminal boundary cannot be replaced by data outside Q. □

Theorem 562.3 (TypedReductionNormalForm subject-reduction seal)**.** *Let* $Q = (T, J, E, R, F, H, C, P, N)$ *be an accepted* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packet. Every public target-membership read after a typed reduction reaches the normal-form boundary only through the displayed* Ext *row* E*, the* Cont *route* R*, the componentwise* hsame *transports* H*, the continuation reads* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$ *row* N*. The read does not use host conversion, semantic equality of terms, a global normalization theorem, or a normalization oracle.*

Lean4 ✓ `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket subject reduction seal`

Proof. Project the accepted packet. The source and target term windows are in T, the judgment context is in J, the membership data are in E, and the typed reduction route is in R.

Replay the route through C and transport the visible rows through H. The package row P records the dependencies on BHist, Ext, Cont, hsame, the ledger, the boundary row, and the local certificate, so the target membership read is still the displayed BEDC packet.

Finally read the terminal boundary from F and authenticate it through N. The carrier inventory has no row for host conversion, semantic equality, global normalization, erased derivation semantics, or a normalization oracle, so none of those objects can participate in the public read. □

Theorem 562.4 (TypedReductionNormalForm route exactness)**.** *Let $Q = (T, J, E, R, F, H, C, P, N)$ be an accepted* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packet. Every packet-visible typed reduction read is the displayed route R together with the named continuation reads in C. The read may use the term row T, judgment ledger J, membership row E, componentwise transports H, provenance row P, and local naming row N, but it has no route through host conversion, semantic equality of terms, erased derivation semantics, a global normalization theorem, or a normalization oracle.*

Proof. Project the packet using Definition 562.1. The only row that stores typed reduction is R, and the only row that replays it for consumers is C.

Apply the NameCert obligation surface of Definition 562.2. It names reduction-route exactness as a local obligation and binds every intermediate step to the displayed Cont rows.

The subject-reduction seal of Definition 562.3 keeps target membership reads on the same route. Since the carrier contains no host conversion, semantic equality, erased derivation, global normalization, or oracle coordinate, the displayed route is exact. □

Lean4 ✓ `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket route exactness`

Theorem 562.5 (TypedReductionNormalForm terminal-boundary non-escape)**.** *For an accepted* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packet $Q = (T, J, E, R, F, H, C, P, N)$, every normal-form access is bound to the terminal boundary row F. A consumer may authenticate the boundary through H, C, P, N and may read the surrounding typed route through T, J, E, R, but it cannot replace F by host conversion, semantic equality of terms, erased derivation semantics, a global normalization theorem, or a normalization oracle.*

Proof. Begin with the carrier inventory. The terminal normal-form boundary is represented by the row F, while the source, target, judgment, membership, and route data are represented by T, J, E, R.

Use the NameCert obligations to expose the boundary row. They state that terminal access is accepted only at F and that further typed-reduction consumption must be a displayed packet row.

Finally combine this with the subject-reduction seal. Target membership arrives through E and R, then the terminal boundary is read from F and authenticated through H, C, P, N; no hidden conversion, semantic equality, erased derivation, global normalization, or oracle row is available. □

Lean4 ✓ `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket terminal boundary nonescape`

Theorem 562.6 (TypedReductionNormalForm transport stability)**.** *Accepted* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packets preserve their finite route and terminal-boundary surface under the componentwise transports H and continuation reads C. If the rows T, J, E, R, F, P, N are transported componentwise, then the source membership, typed reduction route, target membership, and normal-form boundary remain the same packet surface over $Q = (T, J, E, R, F, H, C, P, N)$.*

Proof. Read the transport rows from the packet definition. The row H records componentwise hsame transports for terms, judgments, memberships, reduction routes, normal-form boundaries, package rows, and certificate rows, while C records the corresponding continuation reads.

Apply the NameCert obligations. Carrier exposure, route exactness, the subject-reduction seal, and the normal-form boundary all refer to the same displayed rows, so transporting one component cannot introduce a new route or boundary object.

The subject-reduction seal then repacks the transported source membership, typed reduction route, target membership, and terminal boundary through E, R, F, H, C, P, N. Thus transport stability preserves exactly the finite packet surface and excludes host conversion, semantic equality, global normalization, erased derivation semantics, and normalization oracles. □

Lean4 ✓ `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket transport stability`

Theorem 562.7 (TypedReductionNormalForm terminal readback bridge)**.** *The checked transport-stable* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *packet bridges the public terminal normal-form boundary to the standard readback surface only through the displayed rows* $T, J, E, R, F, H, C, P, N$*. The bridge reads source membership, the finite typed-reduction route, target membership, terminal normal-form boundary, componentwise transports, continuation replay, provenance, and the local* NameCert *row. It excludes host conversion, semantic equality of terms, erased derivation semantics, normalization oracles, and any host normalization procedure outside the packet.*

Proof. Start with the transport-stable packet of Definition 562.6. It preserves the same finite surface $Q = (T, J, E, R, F, H, C, P, N)$ under componentwise transports and continuation reads.

The terminal readback surface is reached by reading F after the route R and membership row E have been replayed through C and transported through H. The judgment ledger J, provenance row P, and local naming row N keep that read inside the displayed packet.

Because the carrier has no coordinate for host conversion, semantic equality, erased derivation semantics, global normalization, or a normalization oracle, the bridge cannot use any of those routes. It is therefore exactly the public terminal boundary read through $T, J, E, R, F, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{TypedReductionNormalForm}^{\uparrow}$ packet is generated from a term-window row, a finite typing-judgment ledger, source and target membership rows, a displayed typed-reduction route, a terminal normal-form boundary, componentwise transports, continuation reads, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 562.1, Definition 562.2, Definition 562.3, Definition 562.4, Definition 562.5, Definition 562.6, and Definition 562.7 bind the BHist, Ext, Cont, hsame, Pkg, and NameCert scope for the typed route, source membership, target membership, terminal normal-form boundary, and standard readback bridge, with chapter 683 as the representative downstream consumer of the finite terminal readback bridge.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TypedReductionNormalForm}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TypedReductionNormalFormUp.TypedReductionNormalFormPacket\_transport\_stability`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host conversion, semantic equality of terms, global normalization, erased derivation semantics, normalization oracles, or bridge checking.

Upgrade path: No further theory-axis upgrade remains; bridge checking is governed only by the verification axis.

Chapter 563

A Concrete Naming Certificate for $\mathsf{TasteGateCompilerWitness}^{\uparrow}$

The $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ packet is the finite compiler-facing NameCert surface for a chapter-level taste gate. It records the candidate chapter token, the ChapterTasteGate row, the displayed event-flow conservativity row, the no-hidden-input, round-trip, and layer-separation rows, the dependency ledger, the audit result row, transport and replay routes, package provenance, and local naming data in one visible certificate.

The $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ packet consumes the candidate-admission packet of chapter 545 and the endpoint and route packaging discipline of chapter 473. The witness is narrower than either sibling: it certifies only the displayed compiler-facing TasteGate evidence row and audit-result route, not arbitrary admission or compiler correctness.

Definition 563.1 (TasteGateCompilerWitness carrier)**.** *A* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, T, E, N, R, L, D, A, H, C, P, Z)$$

with the following displayed rows:

1. *X is the candidate chapter token, recording the chapter label, declared carrier name, and paper site whose taste-gate status is being certified;*
2. *T is the* ChapterTasteGate *row, exposing the four chapter-level obligations used by the admission check;*
3. *E is the event-flow conservativity row, recording the finite* GroundCompiler *event-flow image through which the candidate text and generated declaration surface are read;*
4. *N is the no-hidden-input row, stating that every accepted input channel for the candidate is named in the event-flow image and dependency ledger;*
5. *R is the round-trip row, recording that the displayed compiler witness can be replayed from the named source tokens to the named certificate rows;*
6. *L is the layer-separation row, keeping paper prose, compiler events, Lean declarations, and package provenance in their displayed layers;*
7. *D is the dependency ledger, listing the local BEDC rows, ground-compiler theorem references, candidate NameCert rows, and refusal-side gaps used by the witness;*
8. *A is the audit result row, recording acceptance or refusal of the witness surface against T, E, N, R, L, D;*

9. *H stores componentwise* hsame *transports for $X, T, E, N, R, L, D, A, C, P, Z$;*
10. *C records the displayed* Cont *routes by which a reader accesses the taste-gate rows, compiler witness rows, dependency ledger, and audit result;*
11. *P is the* Pkg *provenance over* BHist, NameCert, ChapterTasteGate, GroundCompiler, *compiler event-flow rows, dependency ledgers, audit results,* hsame, Cont, *and local naming;*
12. *Z is the local* $\mathsf{NameCert}_{\mathsf{TasteGateCompilerWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden prompt state, oracle taste, unlisted compiler facts, host-side chapter approval, or a dependency outside D.

Lean4 (definition) `BEDC.Derived.TasteGateCompilerWitnessUp.TasteGateCompilerWitnessUp`
Lean4 ✓ `BEDC.Derived.TasteGateCompilerWitnessUp.TasteGateCompilerWitnessTasteGate single carrier alignment`

Theorem 563.2 (TasteGateCompilerWitness NameCert obligations)**.** *For every accepted* TasteGateCompilerWitnes *carrier*

$$W = (X, T, E, N, R, L, D, A, H, C, P, Z),$$

the local $\mathsf{NameCert}_{\mathsf{TasteGateCompilerWitness}^{\uparrow}}$ *surface consists exactly of carrier exposure for W, candidate-token exposure through X,* ChapterTasteGate *exposure through T, compiler event-flow exposure through E, no-hidden-input exposure through N, round-trip exposure through R, layer-separation exposure through L, dependency-ledger exposure through D, audit-result exposure through A, componentwise transport through H, replay routes through C, package provenance through P, and local naming through Z.*

Proof. Carrier exposure is projection from Definition 563.1. The tuple lists every row available to a reader of the compiler witness.

The candidate and taste-gate obligations are X and T. They identify the chapter token and expose the chapter-level conservativity, no-hidden-input, round-trip, and layer-separation obligations without importing an unnamed chapter object.

The compiler-facing obligations are E, N, R, L, D, A. These rows record the displayed event-flow image, the input-channel boundary, replay route, layer boundary, dependency ledger, and audit result as finite data.

The structural obligations are H, C, P, Z. Transport, continuation, provenance, and local naming preserve the same witness surface and add no hidden approval channel. Hence the listed rows exhaust the local NameCert surface. □

Theorem 563.3 (TasteGateCompilerWitness ground compiler route)**.** *Every accepted* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *packet reads the* ChapterTasteGate *row only through the ground-compiler route*

$$X \longmapsto E \longmapsto (N, R, L) \longmapsto D \longmapsto A.$$

Thus conservativity, no-hidden-input, round-trip, and layer-separation claims are accepted only when the displayed event-flow row, compiler replay row, dependency ledger, and audit result row are all present in the same packet.

Proof. Begin with the carrier of Definition 563.1. The only candidate token is X, and the only taste-gate row is T.

The event-flow row E is the compiler-facing entrance to the witness. A reader cannot inspect T as an ambient judgment; the route to the obligations passes through the displayed compiler image.

The rows N, R, L split the compiler evidence into input-channel, replay, and layer-separation checks. The dependency ledger D then records the named BEDC rows and ground-compiler theorem references used by those checks.

Finally A records the audit result for the same rows. Since transport and replay are componentwise through H and C, the accepted route cannot replace the displayed compiler evidence with an unlisted theorem or host-side approval. □

Theorem 563.4 (TasteGateCompilerWitness non-escape)**.** *Every exported read of an accepted* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *packet factors through*

$$(X, T, E, N, R, L, D, A, H, C, P, Z).$$

The read may export the candidate chapter token, $\mathsf{ChapterTasteGate}$ *obligations, ground-compiler event-flow evidence, no-hidden-input row, round-trip row, layer-separation row, dependency ledger, audit result, transport routes, package provenance, and local NameCert. It cannot export hidden prompt state, oracle taste, unlisted compiler facts, selected host semantics, or a dependency not named by* D*.*

Proof. Project the exported read through the accepted carrier. The tuple gives exactly the rows $X, T, E, N, R, L, D, A, H, C, P, Z$.

Use Definition 563.2 to account for the local naming surface, candidate token, taste-gate row, compiler evidence, ledger, audit result, transport, continuation, and provenance.

Use Definition 563.3 for the route from the candidate token to the audit result. Every compiler-facing claim is tied to the displayed event-flow and dependency rows.

The carrier has no coordinate for hidden prompt state, oracle taste, unlisted compiler facts, selected host semantics, or a dependency outside D. No exported read can therefore supply one. □

Theorem 563.5 (TasteGateCompilerWitness public package)**.** *The public package exported by an accepted* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *carrier is the finite surface*

$$\mathsf{Pub}(W) = (X, T, E, N, R, L, D, A, H, C, P, Z).$$

It is a concrete NameCert packet tying $\mathsf{ChapterTasteGate}$ *obligations to ground-compiler witness rows, dependency-ledger rows, and audit-result rows. Any AI-origin chapter promotion that cites this package obtains only that finite witness surface and no external admission principle.*

Proof. Start with the carrier and NameCert obligations. They provide the exact public row list and identify the local certificate surface as finite BHist data.

Apply Definition 563.3. This binds the taste-gate obligations to the compiler event-flow row, replay row, layer-separation row, dependency ledger, and audit result.

Apply Definition 563.4. The exported package cannot exceed the displayed rows, so it cannot become an external admission principle or host-side taste judgment.

Thus $\mathsf{Pub}(W)$ is precisely the concrete compiler-witness NameCert packet available to downstream chapter-promotion checks. □

Theorem 563.6 (TasteGateCompilerWitness mature downstream package)**.** *For every accepted* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *carrier*

$$W = (X, T, E, N, R, L, D, A, H, C, P, Z),$$

every downstream AI-origin promotion, conservativity-audit consumer, or compiler-witness audit consumer of the accepted packet factors through

$$\mathsf{Pub}(W) = (X, T, E, N, R, L, D, A, H, C, P, Z).$$

The consumer receives the displayed candidate token, ChapterTasteGate *row, ground-compiler event-flow row, no-hidden-input row, round-trip row, layer-separation row, dependency ledger, audit result, componentwise transports, replay routes, package provenance, and local NameCert, and receives no hidden prompt state, oracle taste, unlisted compiler fact, host-side approval, or dependency outside D.*

Proof. Start with the carrier definition. The accepted packet has exactly the rows $X, T, E, N, R, L, D, A, H, C, P, Z$, with no carrier coordinate for hidden prompt state, oracle taste, unlisted compiler facts, host-side approval, or dependencies outside D.

Use Definition 563.2 to identify the local NameCert obligations exposed by those rows. The downstream consumer can name the candidate token, taste-gate row, compiler evidence, ledger, audit result, transports, replay routes, provenance, and local certificate only through that displayed surface.

Use Definition 563.3 to force every taste-gate and compiler-witness read through the ground-compiler event-flow, replay, layer-separation, dependency-ledger, and audit-result rows. This prevents the consumer from replacing the public route with an unlisted compiler fact or host-side approval.

Finally apply Definition 563.4 and Definition 563.5. The exported package is precisely $\mathsf{Pub}(W)$, so the downstream promotion or audit consumer factors through the same public package and obtains no additional state. □

Closure status for $\mathsf{NameCert}_{\mathsf{TasteGateCompilerWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ packet is generated from a candidate chapter token, ChapterTasteGate row, compiler event-flow row, no-hidden-input row, round-trip row, layer-separation row, dependency ledger, audit result row, componentwise transports, displayed continuations, package provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.TasteGateCompilerWitnessUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TasteGateCompilerWitness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 563.1, Definition 563.2, Definition 563.3, Definition 563.4, Definition 563.5, and Definition 563.6 bind the seed carrier, local NameCert surface, compiler witness route, non-escape boundary, public package, and representative downstream consumer package for TasteGate compiler-witness audits.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TasteGateCompilerWitness}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TasteGateCompilerWitnessUp.TasteGateCompilerWitnessTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove the candidate chapter's mathematical content, certify arbitrary host semantics, or close bridge checking for the compiler-witness packet.

Upgrade path: No further paper-axis theory step remains for this packet; verification-axis bridge checking is a separate closure task.

Chapter 564

A Concrete Naming Certificate for TasteGateStability$^{\uparrow}$

The TasteGateStability$^{\uparrow}$ packet turns the TasteGate discipline of chapter 9459 into a finite NameCert carrier. It records the proposed chapter site, displayed carrier surface, ChapterTasteGate obligations, ground-compiler witness rows, admission ledger, componentwise transports, replay routes, package provenance, and the local boundary that prevents a TasteGate pass from becoming hidden semantic approval.

The stability packet consumes the candidate-admission surface of chapter 545 and the compiler-witness route of chapter 563: admission supplies the candidate row being tested, while the compiler witness supplies the ground-compiler evidence read by W and replayed through C.

Definition 564.1 (TasteGateStability carrier)**.** *A* TasteGateStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, K, T, W, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the candidate chapter identity row, recording the chapter label, origin row, and concrete-instance site;*
2. *K is the candidate NameCert carrier row, exposing the finite source, classifier, pattern, ledger, and non-escape surface proposed by the chapter;*
3. *T is the* ChapterTasteGate *row, displaying conservativity, no-hidden-input, round-trip, and layer-separation obligations for the same candidate;*
4. *W is the ground-compiler witness row, listing the finite event-flow image and named compiler facts used to read the no-hidden-input, round-trip, and source-channel separation gates;*
5. *A is the admission ledger, recording accepted and refused TasteGate reads for the displayed candidate surface;*
6. *H stores componentwise* hsame *transports for X, K, T, W, A, C, P, N;*
7. *C records the displayed* Cont *routes that replay the candidate carrier, TasteGate row, ground-compiler witnesses, and admission ledger;*
8. *P is the* Pkg *provenance over* BHist*,* NameCert*,* ChapterTasteGate*, ground-compiler witnesses, admission ledgers,* hsame*,* Cont*, and local naming;*

9. N is the local $\mathsf{NameCert}_{\mathsf{TasteGateStability}^{\uparrow}}$ *row.*
The packet contains no hidden prompt state, no host-side theory approval, no oracle classifier, no unlisted compiler theorem, and no global horizon admission rule outside the displayed rows.

Lean4 (definition) `BEDC.Derived.TasteGateStabilityUp.TasteGateStabilityUp`
Lean4 ✓ `BEDC.Derived.TasteGateStabilityUp.TasteGateStabilityTasteGate single carrier alignment`

Theorem 564.2 (TasteGateStability NameCert obligations)**.** *For every accepted* $\mathsf{TasteGateStability}^{\uparrow}$ *carrier*

$$S = (X, K, T, W, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TasteGateStability}^{\uparrow}}$ *surface consists exactly of carrier admission for S, candidate identity through X, candidate NameCert exposure through K,* $\mathsf{ChapterTasteGate}$ *exposure through T, ground-compiler witness exposure through W, admission-ledger accounting through A, componentwise transport through H, replay routes through C, package provenance through P, and the local naming row N.*

Proof. Carrier admission is projection from Definition 564.1. The packet exposes exactly the candidate identity, candidate carrier, TasteGate obligations, ground-compiler witness rows, admission ledger, transport rows, replay routes, provenance, and local naming row.

The internal NameCert surface is the row K. It is read as a displayed candidate carrier, classifier, pattern, ledger, and non-escape surface, not as an ambient guarantee that the candidate has already joined the horizon.

The TasteGate surface is the row T, while the compiler-facing evidence is the row W. These rows keep chapter-level stability claims tied to the finite compiler witnesses that display no-hidden-input, round-trip, and layer-separation checks.

Finally A, H, C, P, N account for accepted and refused reads, transport the same rows, replay them for consumers, record provenance, and name the packet locally. No other NameCert obligation is exposed by the carrier. □

Theorem 564.3 (TasteGateStability ground-compiler witness route)**.** *Every accepted* $\mathsf{TasteGateStability}^{\uparrow}$ *route that reads the* $\mathsf{ChapterTasteGate}$ *row T must pass through the ground-compiler witness row W and the replay routes in C. The route may report the displayed no-hidden-input, round-trip, and layer-separation gates for the candidate, but it cannot cite an unlisted compiler theorem or an external semantic judgment.*

Proof. Project the carrier of Definition 564.1. The only row that carries chapter-level TasteGate obligations is T, and the only row that records compiler evidence for those obligations is W.

The replay routes in C bind the read of T to the displayed witness row W. Thus a consumer cannot detach a no-hidden-input, round-trip, or layer-separation claim from the finite event-flow image and named compiler facts that support it.

Since the carrier has no coordinate for an unlisted compiler theorem or external semantic judgment, every accepted TasteGate route remains inside the displayed witness rows. □

Theorem 564.4 (TasteGateStability non-escape boundary)**.** *No consumer of an accepted* $\mathsf{TasteGateStability}^{\uparrow}$ *packet can extract hidden prompt state, host-side theory approval, an oracle classifier, an unlisted compiler theorem, or a global horizon admission rule. Every consumer read factors through*

$$(X, K, T, W, A, H, C, P, N).$$

Proof. Inspect the carrier inventory. Its coordinates are the candidate identity row, candidate NameCert carrier row, TasteGate row, ground-compiler witness row, admission ledger, transport rows, replay routes, package provenance, and local naming row.

The transport and continuation rows preserve exactly those coordinates. They can replay a displayed candidate surface and its TasteGate evidence, but they do not introduce host approval, hidden prompt state, oracle classification, or an additional compiler fact.

Therefore the forbidden objects have no projection target in the packet. Any consumer read seeking one of them has to leave the displayed rows, so it is not a read of the accepted $\mathsf{TasteGateStability}^{\uparrow}$ carrier. □

Theorem 564.5 (TasteGateStability carrier field obligation)**.** *Every accepted* $\mathsf{TasteGateStability}^{\uparrow}$ *packet exposes candidate identity, candidate* NameCert *carrier,* ChapterTasteGate *row, ground-compiler witness row, admission ledger, componentwise transports, replay routes, package provenance, and local naming as fields of the same finite carrier*

$$S = (X, K, T, W, A, H, C, P, N).$$

The TasteGate carrier-alignment marker applies to this single carrier surface; it does not authorize a separate prompt state, host-side approval channel, oracle classifier, unlisted compiler theorem, or global horizon-admission row.

Proof. Project the carrier of Definition 564.1. Its coordinates are exactly $X, K, T, W, A, H, C, P, N$, with X carrying candidate identity, K carrying the candidate NameCert surface, T carrying ChapterTasteGate, W carrying ground-compiler witnesses, A carrying the admission ledger, and H, C, P, N carrying transport, replay, provenance, and naming.

Apply Definition 564.2. The local NameCert surface reads those coordinates as one accepted packet rather than as independent chapter-side records.

The ground-compiler witness route of Definition 564.3 binds the TasteGate row T to the witness row W and the replay routes C. The checked TasteGate carrier-alignment marker records the same single-carrier alignment for the displayed surface.

Finally use Definition 564.4. Since the carrier has no coordinate for hidden prompt state, host approval, oracle classification, unlisted compiler facts, or global horizon admission, the listed fields are the full carrier-field obligation surface. □

Theorem 564.6 (TasteGateStability admission ledger obligation)**.** *For every accepted* $\mathsf{TasteGateStability}^{\uparrow}$ *packet* $S = (X, K, T, W, A, H, C, P, N)$*, the admission ledger row* A *records accepted and refused* ChapterTasteGate *reads for the displayed candidate surface. The ledger is read through the same carrier, witness, transport, replay, provenance, and naming rows, and it never becomes host-side theory approval, semantic certification of the candidate chapter, bridge checking, or global horizon admission.*

Proof. Begin with the carrier definition. The row A is one field of the accepted packet, and its role is admission-ledger accounting for the same candidate identity X, candidate carrier K, TasteGate row T, and compiler witness row W.

Apply Definition 564.2. It includes admission-ledger accounting through A in the local NameCert surface, together with the transports H, replay routes C, provenance P, and local naming row N.

Next use Definition 564.3. Accepted or refused reads of T are tied to the ground-compiler witness row W and to the displayed replay routes, so the ledger records TasteGate access to that finite evidence rather than a host semantic judgment.

The checked TasteGate carrier-alignment marker and Definition 564.4 close the boundary. The marker aligns TasteGate with the same finite carrier, and the non-escape theorem blocks host approval, hidden prompt state, oracle classifiers, unlisted compiler theorems, and global horizon admission from entering through A. □

Theorem 564.7 (TasteGateStability scoped dependency boundary)**.** *Every accepted read of a* $\mathsf{TasteGateStability}^{\uparrow}$ *packet*

$$S = (X, K, T, W, A, H, C, P, N)$$

is scoped by the displayed BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* ChapterTasteGate*, and ground-compiler witness rows. The read cannot be replaced by host approval, hidden prompt state, oracle classification, or unlisted compiler facts.*

Proof. First project the packet through Definition 564.2. The local certificate reads candidate identity, candidate carrier, TasteGate exposure, ground-compiler witnesses, admission ledger, transport, replay, provenance, and naming through the displayed carrier.

Next use Definition 564.3. It binds the TasteGate row T to the witness row W and the replay routes C, so compiler-facing evidence is finite packet data rather than an ambient semantic judgment.

Apply Definition 564.4, Definition 564.5, and Definition 564.6. These rows keep the carrier fields and ledger reads inside S and exclude host approval, hidden prompt state, oracle classification, unlisted compiler facts, bridge checking, and global horizon admission. □

Closure status for $\mathsf{NameCert}_{\mathsf{TasteGateStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TasteGateStability}^{\uparrow}$ packet is generated from a candidate chapter identity row, candidate NameCert carrier row, ChapterTasteGate row, ground-compiler witness row, admission ledger, componentwise transports, replay routes, package provenance, and local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TasteGateStability}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 564.1, Definition 564.2, Definition 564.3, Definition 564.4, Definition 564.5, Definition 564.6, Definition 564.7, and

Lean4 ✓ `BEDC.Derived.TasteGateStabilityUp.taste gate`

export the public TasteGateStability surface over BHist, hsame, Cont, Pkg, NameCert, ChapterTasteGate, and the ground-compiler witness route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TasteGateStability}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TasteGateStabilityUp.TasteGateStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove the candidate chapter's mathematical content, certify host semantics, close bridge checking, or provide Lean verification for the TasteGateStability packet.

Upgrade path: No paper-side closure upgrade remains; bridge checking remains the formal bridge-axis task.

Chapter 565

A Concrete Naming Certificate for $\mathsf{GeneratorInductionBoundary}^{\uparrow}$

The $\mathsf{GeneratorInductionBoundary}^{\uparrow}$ packet is the concrete NameCert surface for separating authorized generator-internal induction from free-floating induction. It packages the generator signature, closure witness, continuation route, GAP socket, transport row, package provenance, and local naming row into one finite carrier, so that every induction read remains attached to the generator that licenses it.

Definition 565.1 (GeneratorInductionBoundary carrier)**.** *A* $\mathsf{GeneratorInductionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, C, G, H, P, N)$$

with the following displayed rows:

1. *S is the authorized generator signature row, recording the generator name, its closed constructor list, and the local induction pattern licensed by that constructor list;*
2. *W is the closure witness row, recording that the displayed induction step ranges only over the closure generated by S;*
3. *C is the continuation route row, recording the* Cont *paths by which a consumer may replay the generator-internal induction step and transport its conclusion to later finite histories;*
4. *G is the GAP socket row, recording attempted reads that would move from the generator closure to a free-floating universal induction principle;*
5. *H stores componentwise* hsame *transport for S, W, C, G, P, N;*
6. *P is the* Pkg *provenance over* BHist*, authorized generator signatures, closure witnesses, continuation routes, GAP sockets,* hsame*, and local naming;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a universal induction rule, an unbounded object domain, a hidden host-CIC evaluator, an oracle continuation, or an induction conclusion detached from the authorized generator signature.

Theorem 565.2 (GeneratorInductionBoundary NameCert obligations)**.** *For every accepted* $\mathsf{GeneratorInductionBou}$ *carrier*

$$B = (S, W, C, G, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, authorized generator exposure through* S*, closure-witness exposure through* W*, continuation routing through* C*, GAP socket accounting through* G*, componentwise transport through* H*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 565.1. The packet displays the authorized generator signature, closure witness, continuation route, GAP socket, transport, provenance, and naming rows as finite coordinates.

The generator and closure obligations are the rows S and W. They identify the constructor list being used and bind the induction read to the closure generated by that list.

The continuation obligation is the row C. Every replay of the induction step must pass through the displayed route, so the consumer receives a generator-internal conclusion rather than a detached induction rule.

Finally G, H, P, N preserve the same boundary object under socket accounting, transport, provenance, and naming. Since the carrier contains no coordinate for universal induction or hidden host evaluation, the local $\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}$ surface is exactly the displayed finite packet. □

Theorem 565.3 (GeneratorInductionBoundary GAP non-escape)**.** *Every accepted* $\mathsf{GeneratorInductionBoundary}^{\uparrow}$ *continuation route factors through the finite rows*

$$(S, W, C, G, H, P, N).$$

The route may export that an induction step is authorized inside the generator named by S*, carried by the closure witness* W*, replayed through* C*, transported by* H*, packaged by* P*, named by* N*, and separated from free-floating induction by the GAP socket* G*. It cannot export a universal induction principle, a conclusion over an unbounded object domain, or a host-CIC delegation that has no displayed row in the packet.*

Proof. Project the route through the carrier. The only available coordinates are S, W, C, G, H, P, N.

Use Definition 565.2 to account for the authorized generator signature, closure witness, continuation route, GAP socket, transport, provenance, and local naming rows.

The row G records attempted movement from generator-internal induction to a free-floating induction rule. Because such movement is represented only as a socket row and not as an exported conclusion, it cannot become a new induction coordinate.

Thus any exported route remains inside the displayed generator boundary. A universal induction principle, an unbounded object domain, or an unrecorded host-CIC delegation would require an extra row outside B, and the carrier has no such row. □

Theorem 565.4 (GeneratorInductionBoundary generator closure dependency)**.** *For every accepted* $\mathsf{GeneratorInductionBoundary}^{\uparrow}$ *carrier*

$$B = (S, W, C, G, H, P, N),$$

the authorized generator signature row S *and closure witness row* W *are consumed through the generator-closure surface of chapter 486. The GAP socket row* G *is compatible with the finite socket discipline of chapter 498: it records the attempted move from generator-internal induction to a free-floating induction principle without exporting such a principle.*

Proof. Project the carrier in Definition 565.1. The row S records the authorized generator signature, and W records that the displayed induction step ranges only over the closure generated by S.

Use the local obligations of Definition 565.2. They make S and W mandatory exposure rows before the continuation route C can be read, so the boundary consumes the same generator-closure data recorded by chapter 486.

Finally apply Definition 565.3. The row G remains a socket accounting row and does not become an exported induction coordinate. This is the same finite GAP discipline enforced by chapter 498, so the boundary packet links generator closure to the socket without adding universal induction. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GeneratorInductionBoundary}^{\uparrow}$ packet is generated from an authorized generator signature, closure witness, continuation route, GAP socket, componentwise hsame transport, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 565.1, Definition 565.2, and Definition 565.3 bind the finite carrier, local NameCert obligations, continuation boundary, and GAP non-escape surface for generator-internal induction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratorInductionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove a universal induction principle, close host-CIC soundness, certify arbitrary generator signatures, or provide Lean verification for the boundary packet.

Upgrade path: Obligation closure requires checked carrier admission, generator-signature exposure, closure-witness exposure, continuation routing, GAP socket accounting, and non-escape theorems for the same finite boundary packet.

Chapter 566

A Concrete Naming Certificate for $\mathsf{PropextTransportBoundary}^{\uparrow}$

The $\mathsf{PropextTransportBoundary}^{\uparrow}$ surface records the constructive replacement for proposition-stratum identification. It accepts bidirectional implication evidence between propositions only as direction-indexed transport data, then records how contexts, history sameness, package provenance, continuation replay, and the local naming row consume that transport without producing an equality token $P = Q$.

This proposition-stratum boundary is the directed-implication sibling of chapter 515. UnwitnessedExistenceClosu handles the term, type, and proposition closure family, while $\mathsf{PropextTransportBoundary}^{\uparrow}$ isolates the proposition-level transport row and refuses to collapse it into proposition equality.

Definition 566.1 (PropextTransportBoundary carrier)**.** *A* $\mathsf{PropextTransportBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, D, R, H, C, P, N)$$

with the following displayed coordinates:

1. *B is the bidirectional implication evidence for the proposition pair under comparison, recorded as the two directed rows $P \Rightarrow Q$ and $Q \Rightarrow P$;*
2. *D records the chosen direction rows through which each consumer reads the implication evidence;*
3. *R records context-read replacement routes for the admitted consumers, with no coordinate for equality tokens, global rewrite, or transport through $P = Q$;*
4. *H records the* hsame *and* psame *transport rows that move the proposition labels, implication rows, direction choices, context reads, continuation replay, provenance, and local naming row componentwise;*
5. *C records the* Cont *replay routes by which a consumer may request the forward read, the reverse read, the selected direction, and the returned context row;*
6. *P records* Pkg *provenance over the ThreeAxiomsOneClosure schema, the core* NameCert *discipline,* hsame, psame, *and* Cont*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{PropextTransportBoundary}^{\uparrow}}$ *row.*

The carrier has no hidden coordinate for proposition equality, propositional extensionality, host rewrite through $P = Q$, or arbitrary proposition-valued context transport.

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryTasteGate single carrier alignment`

Theorem 566.2 (PropextTransportBoundary NameCert obligations)**.** *For every* $\mathsf{PropextTransportBoundary}^{\uparrow}$ *carrier*

$$T = (B, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PropextTransportBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure for the displayed bidirectional implication, chosen direction, context-read replacement, transport, continuation, provenance, and naming rows;*
2. *direction discipline, requiring each consumer to name a row in D before using either implication row in B;*
3. *transport discipline, requiring every comparison of boundary packets to move only through the componentwise* hsame *and* psame *rows in H;*
4. *replay discipline, requiring every context read to pass through one of the displayed* Cont *routes in C and one of the replacement routes in R;*
5. *non-escape, excluding proposition equality, propositional extensionality, host rewrite through $P = Q$, and arbitrary proposition-valued context transport.*

Proof. Carrier exposure is projection from Definition 566.1. The packet lists the implication evidence, direction rows, context-read routes, transport rows, replay routes, provenance, and local naming row as finite coordinates.

Direction discipline is local because the only way to consume B is through a row in D. A forward consumer reads the $P \Rightarrow Q$ row together with its chosen direction; a reverse consumer reads the $Q \Rightarrow P$ row together with its chosen direction. No consumer obtains an undirected identification from the pair.

Transport discipline follows by reading equality-sensitive movement through H. The hsame and psame rows preserve the displayed coordinates componentwise, so comparison of two boundary packets can move proposition labels and classifier rows only along the named transport rows.

Replay discipline is then forced by C and R. The continuation row selects one of the admitted requests, and the replacement row returns the corresponding context-read output without invoking a global rewrite rule.

The non-escape clause is inspection of the carrier inventory. Since no coordinate names $P = Q$, propext, host rewrite, or arbitrary context transport, the local naming surface cannot export any of them. □

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryNameCertObligations`

Theorem 566.3 (PropextTransportBoundary direction determinacy)**.** *For every* $\mathsf{PropextTransportBoundary}^{\uparrow}$ *carrier*

$$T = (B, D, R, H, C, P, N),$$

any consumer of an implication row in B must first select a direction row in D. If two accepted packets agree on D, H, C, P, N, then the exposed read is determined directionwise: a forward selection exposes the same $P \Rightarrow Q$ read, and a reverse selection exposes the same $Q \Rightarrow P$ read. No unordered identification of the two implication rows, witness of $P = Q$, invocation of propext*, host rewrite, or arbitrary proposition-valued context transport is exported by this agreement.*

Proof. Project the carrier coordinates from Definition 566.1. The implication evidence is stored in B, but Definition 566.2 makes each use pass through a chosen row of D before either implication row can be read.

Now compare two accepted packets whose D, H, C, P, N coordinates agree. The common D row fixes whether the consumer asks for the forward or reverse implication read. The common H rows move proposition labels and packet coordinates componentwise, while the common C routes replay the selected request through the displayed continuation surface.

The common provenance row P and local naming row N ensure that this replay is still the same named boundary consumption, not a host-side rewrite. Thus a forward request remains the directed $P \Rightarrow Q$ read, and a reverse request remains the directed $Q \Rightarrow P$ read.

Finally inspect the carrier inventory. Since the packet has no coordinate for unordered proposition equality, propext, host rewrite through $P = Q$, or arbitrary proposition-valued context transport, direction agreement cannot export any of these operations. □

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryDirectionDeterminacy`

Theorem 566.4 (PropextTransportBoundary non-escape)**.** *For every* PropextTransportBoundary$^{\uparrow}$ *carrier*

$$T = (B, D, R, H, C, P, N),$$

the exported proposition-stratum replacement surface is exactly the finite packet T together with its displayed projection routes. In particular, reading the boundary cannot construct a witness of $P = Q$, cannot invoke propext*, cannot perform host rewrite through $P = Q$, and cannot transport an arbitrary proposition-valued context outside the routes recorded in R and C.*

Proof. Read the exported surface by projections from T. The only implication data are the two directed rows in B, and the only permitted uses of those rows are the choices recorded in D.

The context side is likewise explicit. A consumer must use a replacement route in R and replay it through C; those rows return only the named context-read outputs.

If a witness of $P = Q$, a propext invocation, a host rewrite, or an arbitrary context transport were exported, it would have to be a coordinate of the finite packet or the output of one of its routes. The carrier contains no such coordinate, and R and C are defined only for displayed replacement reads. Hence none of the excluded operations escapes the boundary. □

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryNonEscapeReadback`
Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryNonEscape`

Theorem 566.5 (PropextTransportBoundary ledger exactness)**.** *For every* PropextTransportBoundary$^{\uparrow}$ *carrier*

$$T = (B, D, R, H, C, P, N),$$

the provenance ledger of the boundary is exactly the displayed Pkg *row P together with the local naming row N and the transport and replay rows H, C. Its dependency surface is over ThreeAxiomsOneClosure, core* NameCert*,* hsame*,* psame*, and* Cont*, and it contains no proposition-equality, propositional-extensionality, or host-rewrite dependency row.*

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryLedgerExactness`

Proof. The ledger-bearing coordinates are displayed in the carrier: P records package provenance, N records the local name, H records transport, and C records replay. These rows are finite coordinates of T.

Inspect the support named by P. It ranges over the ThreeAxiomsOneClosure schema, the core NameCert discipline, hsame, psame, and Cont. These are exactly the rows needed to explain the replacement of proposition equality by directed implication transport.

No further ledger row is present. Therefore any proposition-equality, propositional-extensionality, or host-rewrite dependency would require an additional coordinate outside the packet, contradicting the displayed carrier inventory. □

Theorem 566.6 (PropextTransportBoundary three-axioms sibling link)**.** *The* PropextTransportBoundary$^{\uparrow}$ *ledger is the proposition-stratum projection of the three-stratum no-escape surface of chapter 515. It keeps only the displayed bidirectional implication, direction, replacement, transport, replay, provenance, and naming rows, and it introduces no witness of $P = Q$, no* propext *row, no host rewrite, and no arbitrary context transport.*

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryThreeAxiomsSiblingLink`

Proof. Read the local carrier inventory from Definition 566.1. The packet contains only B, D, R, H, C, P, N, so the proposition-stratum boundary is represented by directed implication evidence, chosen direction rows, local replacement routes, componentwise transport, continuation replay, provenance, and local naming.

Apply Definition 566.5. The ledger surface is exactly P together with N, H, C, over the ThreeAxiomsOneClosure schema, core NameCert rows, hsame, psame, and Cont.

The sibling chapter chapter 515 supplies the broader term, type, and proposition no-escape surface. Taking the proposition-stratum projection leaves the displayed directed transport rows here and adds no proposition equality token.

Therefore the local ledger is linked to that sibling surface without introducing $P = Q$, propext, host rewrite, or arbitrary context transport. □

Theorem 566.7 (PropextTransportBoundary TasteGate handoff)**.** *Every accepted* PropextTransportBoundary$^{\uparrow}$ *packet gives a finite handoff object for a* ChapterTasteGate *consumer: the gate may inspect B, D, R, H, C, P, N, accept the directed replacement reads recorded there, and reject any request whose consumption path requires proposition equality,* propext*, host rewrite through $P = Q$, or arbitrary proposition-valued context transport.*

Proof. Build the handoff object by projecting the seven coordinates of the carrier. The gate receives the bidirectional implication rows, the chosen direction rows, the replacement routes, the componentwise transport rows, the replay routes, provenance, and the local naming row.

Acceptance is finite because every permitted read is named by a coordinate of T. A request that follows D, R, H, and C is a directed replacement read and can be checked against the displayed provenance in P and the local name in N.

Rejection is the complement of this displayed surface. A request requiring $P = Q$, propext, host rewrite, or arbitrary proposition-valued context transport asks for a coordinate absent from T, so the ChapterTasteGate has a concrete reason to refuse it. □

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryTasteGateHandoff`

Theorem 566.8 (PropextTransportBoundary context replacement locality)**.** *For every accepted* PropextTransportBoundary$^{\uparrow}$ *carrier*

$$T = (B, D, R, H, C, P, N),$$

each context read exported by the boundary is local to the displayed replacement row R and the displayed Cont *row C. A consumer may replay only a named replacement route through C, and the replay returns only the context-read output recorded in R; it does not create a host rewrite, a witness of $P = Q$, or arbitrary transport for proposition-valued contexts.*

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryContextReplacementLocality`

Proof. Project the carrier from Definition 566.1. The replacement coordinate is R, and the replay coordinate is C; no other coordinate is assigned to context transport.

Apply Definition 566.2 to force every context read through the replay discipline. The obligation surface requires a displayed Cont route and one of the replacement routes in R before any context-read output can be returned.

The direction and non-escape theorems then keep the read local. By Definition 566.3, implication consumption remains directionwise, and by Definition 566.4, the packet has no coordinate for equality, propext, host rewrite, or arbitrary proposition-valued transport. □

Theorem 566.9 (PropextTransportBoundary direction ledger coverage)**.** *For every accepted* PropextTransportBoun *carrier*

$$T = (B, D, R, H, C, P, N),$$

every accepted implication read is covered by one of the directed rows in D. The ledger never records the two implication rows in B as an unordered proposition equality, and no public read of the packet can replace this directed coverage by $P = Q$ or propext*.*

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryDirectionLedgerCoverage`

Proof. The carrier exposes implication evidence only through B together with direction rows D. The NameCert obligations of Definition 566.2 require a consumer to name a row in D before using either implication row.

The coverage statement is therefore exactly Definition 566.3 read as a ledger condition. A forward request is covered by the forward row, and a reverse request is covered by the reverse row; there is no third unordered row.

Finally use Definition 566.5 and Definition 566.4. The provenance ledger contains only the displayed package, naming, transport, and replay rows, and the carrier inventory excludes proposition equality and propext. □

Theorem 566.10 (PropextTransportBoundary TasteGate refusal exactness)**.** *For every accepted* PropextTransportBoundary$^{\uparrow}$ *packet, the* ChapterTasteGate *refusal is exact for requests whose consumption path requires a witness of $P = Q$,* propext*, host rewrite through $P = Q$, or arbitrary proposition-valued context transport. Such a request is refused precisely because no displayed row of B, D, R, H, C, P, N supplies the requested operation, while requests following the directed rows in D, replacement row R, transport row H, replay row C, provenance row P, and naming row N remain admissible.*

Lean4 ✓ `BEDC.Derived.PropextTransportBoundaryUp.PropextTransportBoundaryTasteGateRefusalExactness`

Proof. Start from the handoff theorem Definition 566.7. The gate receives exactly the displayed carrier coordinates and accepts directed replacement reads recorded by those coordinates.

For an admitted request, Definition 566.8 supplies the local context route through R and C, while Definition 566.9 supplies the directed implication row in D. Transport and provenance are then read through H, P, N.

For a refused request, inspect the same inventory. By Definition 566.4 and Definition 566.5, the packet has no equality row, propext row, host-rewrite row, or arbitrary context-transport row. The gate therefore refuses exactly those requests whose path needs one of the absent operations. □

Theorem 566.11 (PropextTransportBoundary standard bridge route)**.** *Every accepted* PropextTransportBoundary$^{\uparrow}$ *packet exports its public boundary only through the displayed ThreeAxiomsOneClosure,* NameCert*,* hsame*,* psame*,* Cont*,* Pkg*, and* ChapterTasteGate *rows. This route gives a paper bridge to the standard refusal of proposition-extensional transport without importing proposition equality, host rewrite through $P = Q$, or arbitrary proposition-valued context transport.*

Proof. Start with the carrier of Definition 566.1 and the NameCert obligations of Definition 566.2. They force every accepted read through the displayed rows B, D, R, H, C, P, N before any public boundary can be exported.

Direction determinacy and ledger exactness then identify the bridge boundary. Apply Definition 566.3 to keep implication use direction-indexed, and Definition 566.5 to source the public ledger only from the ThreeAxiomsOneClosure, naming, transport, replay, and provenance rows.

The sibling link and gate handoff supply the standard bridge face. By Definition 566.6, the boundary is the proposition-stratum route of the three-axioms closure surface; by Definition 566.7, a ChapterTasteGate consumer receives exactly this finite handoff.

Finally use context replacement locality and refusal exactness. Theorems Definition 566.8 and Definition 566.10 keep the context route local to R, C and refuse equality, propext, host rewrite, and arbitrary proposition-valued transport. This is the stated paper bridge route. □

Closure status for NameCert$_{\text{PropextTransportBoundary}^{\uparrow}}$**.**

Constructive story: A PropextTransportBoundary$^{\uparrow}$ packet is generated from bidirectional implication evidence, chosen direction rows, context-read replacement routes, componentwise hsame and psame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PropextTransportBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 566.1 and the theorem rows of this chapter bind the displayed carrier $T = (B, D, R, H, C, P, N)$ to the ThreeAxiomsOneClosure provenance surface, core NameCert, hsame, psame, Cont, Pkg, ChapterTasteGate handoff routes, and the paper bridge of Definition 566.11, with proposition equality, propositional extensionality, host rewrite, and arbitrary proposition-valued context transport excluded from the scoped primitive boundary.

Formal verification: FormalStatus(NameCert$_{\text{PropextTransportBoundary}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not claim proposition equality, propositional extensionality, host rewrite through $P = Q$, arbitrary proposition-valued context transport, Lean verification, or bridge checking.

Upgrade path: No stronger paper-side closure is claimed beyond the directed implication transport boundary and its bridge discipline.

Chapter 567

A Zeckendorf Classifier Surface for $\mathsf{ZCarry}^{\uparrow}$

This chapter realizes the zero-marker Zeckendorf carry classifier as a finite BHist naming packet. The packet separates carry-relatedness from hsame: it records the local carry window, the normal-form target window, the non-normal source window, and the displayed transports by which consumers inspect those rows.

The classifier depends horizontally on the normal-form sibling chapter 278: the carrier rows s and t read source rejection and target admission from that surface. The downstream carry packet of chapter 544 then consumes this classifier boundary when it packages the finite carry row for $\mathsf{ZCarry}^{\uparrow}$.

Definition 567.1 (Zeckendorf carry-classifier carrier)**.** *A* $\mathsf{ZCarry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (u, v, c, s, t, h, r, p, n)$$

with the following visible coordinates:

1. *u is the displayed source carry window, with public shape* `word_011`*;*
2. *v is the displayed target carry window, with public shape* `word_100`*;*
3. *c is the local carry row witnessing* $\mathsf{ZCarry}(u, v)$*;*
4. *s is the source-normality boundary row, witnessing that u is rejected by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
5. *t is the target-normality row, witnessing that v is accepted by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
6. *h is the negative history-sameness row, witnessing that the carry relation is not* $\mathsf{hsame}(u, v)$*;*
7. *r is the finite carry-window readback ledger, exposing only u, v, c, s, t, h to downstream consumers;*
8. *p is the* Pkg *provenance over* BHist, hsame, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, *and the local carry generator;*
9. *n is the local* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *row.*

The carrier has no coordinate for Fibonacci value preservation, arithmetic equality, kernel rewriting, host numeric interpretation, or global replacement of hsame*.*

Lean4 (definition) `BEDC.Derived.ZeckendorfCarryClassifierUp.ZeckendorfCarryClassifierUp`
Lean4 ✓ `BEDC.Derived.ZeckendorfCarryClassifierUp.TasteGate.ZeckendorfCarryClassifierTasteGate single carrier alignment`

Definition 567.2 (Zeckendorf carry-classifier sameness)**.** *For accepted* $\mathsf{ZCarry}^{\uparrow}$ *carriers*

$$K = (u, v, c, s, t, h, r, p, n) \quad \text{and} \quad K' = (u', v', c', s', t', h', r', p', n'),$$

the classifier sameness judgment $\mathsf{ZCarrySame}(K, K')$ *holds exactly when the source windows are connected by displayed* $\mathsf{hsame}(u, u')$*, the target windows are connected by displayed* $\mathsf{hsame}(v, v')$*, and the carry rows* c, c' *read the same local generator from* `word_011` *to* `word_100`*. The judgment does not identify* u *with* v *inside either carrier.*

Theorem 567.3 (Zeckendorf carry-classifier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZCarry}^{\uparrow}}$ *surface for the carrier of Definition 567.1 consists exactly of five finite rows:*

1. *carrier admission for the displayed packet* K*;*
2. *source-window rejection by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
3. *target-window admission by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
4. *carry-not-*hsame *separation for the displayed source and target windows;*
5. *ledger exactness, stating that every public read of* $\mathsf{ZCarry}^{\uparrow}$ *factors through* $u, v, c, s, t, h, r, p, n$*.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryClassifierUp.ZeckendorfCarryClassifierCarrier namecert obligations`

Proof. Carrier admission is projection of the finite packet in Definition 567.1. The packet displays the source window, target window, carry row, normality rows, negative hsame row, readback ledger, package provenance, and local naming row.

The source and target normality rows are read from s and t. The source row records rejection of the adjacent-one window, while the target row records acceptance of the zero-zero-one window under the normal-form classifier.

The separation row is h. It states that the carry relation is a classifier edge between the two windows, not a history-sameness proof from the source to the target.

Finally, ledger exactness is read from r, p, n. Since no arithmetic equality, Fibonacci value row, host numeric interpretation, or kernel rewrite appears as a coordinate, the public NameCert surface is exhausted by the displayed finite rows. □

Theorem 567.4 (Zeckendorf carry-window determinacy)**.** *For an accepted* $\mathsf{ZCarry}^{\uparrow}$ *carrier*

$$K = (u, v, c, s, t, h, r, p, n),$$

the displayed carry row c *determines the local source and target windows up to the carrier's external* hsame *transports:* u *is the rejected* `word_011` *window and* v *is the accepted* `word_100` *window. Any consumer that asks for the carried window reads the same finite pair through* r *and cannot choose a second target window inside the packet.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryClassifierUp.ZeckendorfCarryClassifier window determinacy`

Proof. Inspect the carry row c. The $\mathsf{ZCarry}^{\uparrow}$ carrier admits only the local generator from the source window `word_011` to the target window `word_100`.

The source-normality and target-normality rows then fix the orientation. The row s rejects the source as a normal form, while t accepts the target as a normal form; reversing the pair would contradict those displayed roles.

Transport can only occur through the external hsame comparisons named in Definition 567.2. These comparisons preserve each side of the window pair separately and never identify the source with the target. Therefore the finite window readback has a single displayed target inside the packet. □

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryClassifierUp.ZeckendorfCarryClassifierCarrier window determinacy`

Theorem 567.5 (Zeckendorf carry non-escape boundary)**.** *Every public read exported by an accepted* $\mathsf{ZCarry}^{\uparrow}$ *carrier factors through* $u, v, c, s, t, h, r, p, n$*. In particular, the carrier cannot be used as a kernel rewrite rule, an* $\mathsf{hsame}(u, v)$ *witness, a Fibonacci value-preservation theorem, an arithmetic equality, or a global normalization procedure.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryClassifierUp.ZeckendorfCarryClassifierCarrier nonescape boundary`

Proof. Start with the carrier inventory. The packet contains exactly the source and target windows, the local carry row, the source and target normality rows, the negative hsame row, the readback ledger, package provenance, and the local naming row.

The public classifier read is routed through c and r. The normal-form evidence is routed through s and t, and the separation from history-sameness is routed through h.

None of the excluded objects appears as a packet coordinate. Hence no consumer can extract a kernel rewrite, source-target hsame proof, Fibonacci value row, host arithmetic equality, or global normalizer from the $\mathsf{ZCarry}^{\uparrow}$ NameCert surface. □

Theorem 567.6 (Zeckendorf carry-classifier scoped kernel route)**.** *Every accepted* $\mathsf{ZeckendorfCarryClassifier}^{\uparrow}$ *carrier read factors through the displayed* BHist *windows* u, v*, the side transports named by* hsame*, the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *source-rejection and target-admission rows* s, t*, the local* $\mathsf{ZCarry}^{\uparrow}$ *generator row* c*, the finite readback ledger* r*, the* Pkg *provenance row* p*, and the local* NameCert *naming row* n*. The scoped kernel route contains no coordinate for Fibonacci value preservation, global normalization, kernel rewriting, source-target* hsame*, or host arithmetic equality.*

Proof. Project the accepted packet of Definition 567.1. Its carrier rows are exactly the displayed windows, the local carry row, the source and target normal-form rows, the separation row, the readback ledger, provenance, and local naming.

The source and target reads then factor through the normal-form sibling. The row s records the $\mathsf{ZeckendorfNormal}^{\uparrow}$ rejection of the source window, while t records the corresponding target admission; any side comparison is transported only through the external hsame rows of Definition 567.2.

Finally apply the non-escape boundary of Definition 567.5. The readback ledger r, provenance p, and local name n preserve the finite route, and the absent global arithmetic or kernel-rewrite objects cannot be exported from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZeckendorfCarryClassifier}^{\uparrow}$ packet is generated from a finite carry-classifier carrier, source and target normal-form rows, sameness separation, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{ZeckendorfCarryClassifier}^{\uparrow}}$, publicClosed)

Scope closed: The public finite carry-classifier export binds the carrier, sameness rule, NameCert-obligation rows, window determinacy, normal-form dependency, and non-escape boundary over BHist, hsame, ZeckendorfNormal$^{\uparrow}$, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{ZeckendorfCarryClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This classifier block does not prove Fibonacci value preservation, global normalization, source-target hsame, or a Lean implementation.

Upgrade path: Obligation closure requires checked carrier and classifier obligation rows for the finite carry-classifier packet.

Chapter 568

A Concrete Naming Certificate for EffectiveReplacementLedger$^{\uparrow}$

The EffectiveReplacementLedger$^{\uparrow}$ interface records the finite certificate packet that realizes the three replacement directions of section 9387.3. It gathers explicit choice witnesses, classifier-respecting quotient transport, direction-indexed proposition transport, and the axiom-purity refusal row into one local NameCert surface, so later purity and boundary arguments can consume a concrete ledger rather than a schematic discipline statement.

When a replacement row is used as a finite refusal witness, EffectiveReplacementLedger$^{\uparrow}$ consumes the bounded finite-refutation boundary of chapter 579. When a proposed replacement would become a truth-total or validation-as-row query, it consumes the reflection-limit boundary of chapter 586. These sibling links are dependency rows only; they add no proof search, truth principle, quotient principle, choice principle, or propext principle beyond the displayed ledger.

Definition 568.1 (EffectiveReplacementLedger carrier)**.** *An* EffectiveReplacementLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, Q, T, A, H, C, P, N)$$

with the following visible coordinates:

1. *W is the choice-witness row, recording countable witness functions, dependent-step witnesses along displayed* Cont *chains, and finite-recursion witnesses over closed inductive lists;*
2. *Q is the quotient-transport row, recording classifier-respecting transport through typed* NameCert *data without choosing a canonical representative;*
3. *T is the propext-transport row, recording bidirectional implication as direction-indexed* hsame *and* psame *transport rather than as a proposition-equality token;*
4. *A is the axiom-purity refusal row, recording that no consumer may request* Classical.choice*,* Quot.sound*,* propext*, or a host rewrite through those operations;*
5. *H stores componentwise* hsame *and* psame *transports for the witness, quotient, proposition, refusal, provenance, and naming rows;*
6. *C stores the finite* Cont *routes by which consumers read each replacement row and receive only its displayed output;*
7. *P records* Pkg *provenance over the ThreeAxiomsOneClosure schema, typed* NameCert*,* BHist*,* hsame*,* psame*,* Cont*, and the axiom-purity audit surface;*

8. N is the local $\mathsf{NameCert}_{\mathsf{EffectiveReplacementLedger}^{\uparrow}}$ *row.*
The carrier contains no hidden coordinate for an unwitnessed selector, a quotient representative section, a proposition equality $P = Q$, or an ignored axiom-purity gate.

Lean4 (definition) `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerCarrier`

Theorem 568.2 (EffectiveReplacementLedger choice row)**.** *For every accepted carrier $E = (W, Q, T, A, H, C, P, N$ the choice replacement surface is exactly the witness row W together with the continuation routes in C that expose its entries. Countable choice is read as an explicit witness function over a displayed $\mathbb{N}$-indexed family, dependent choice is read as displayed step data along a* Cont *chain, and finite choice is read as primitive recursion over a closed inductive list.*

Proof. Project the carrier to W and C. The witness row names the three admitted forms of choice replacement, and the continuation row gives the only routes by which a consumer may read them.

Each route returns a displayed witness: a function for the countable case, a step witness for the dependent case, or a finite recursive witness for the closed-list case. None of these reads supplies an oracle body outside the carrier.

The refusal row A blocks any request for an unwitnessed selector. Therefore the choice surface of the packet is exhausted by W and its displayed routes in C. □

Lean4 ✓ `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerChoiceRow`

Theorem 568.3 (EffectiveReplacementLedger quotient transport row)**.** *For every accepted carrier $E = (W, Q, T, A, H, C, P, N)$, the quotient replacement surface is exactly the classifier-respecting transport row Q together with the* hsame *and* psame *transport data in H. A consumer may move certified content across the displayed classifier, but it may not extract a canonical representative, a quotient section, or a host-side collapse of carrier elements.*

Proof. Read Q as the quotient-facing coordinate of the packet. It names classifier-respecting transport through typed NameCert data and has no field for a representative-selection function.

The transport row H supplies the componentwise movement required to compare two certified presentations. It moves witness rows, classifier rows, provenance rows, and naming rows through displayed sameness data.

If a consumer asks for a quotient representative or a type-level collapse, the request is not routed through Q or H and is therefore refused by A. Thus quotient use remains transport-only inside the finite packet. □

Lean4 ✓ `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerQuotTransportRow`

Theorem 568.4 (EffectiveReplacementLedger propext transport row)**.** *For every accepted carrier $E = (W, Q, T, A, H, C, P, N)$, the proposition-stratum replacement surface is exactly the direction-indexed transport row T together with the continuation routes in C and the componentwise* hsame *and* psame *rows in H. Bidirectional implication may be consumed in a named direction, but the carrier exports no proposition equality and no host rewrite through $P = Q$.*

Proof. Project the packet to T. This row records bidirectional implication as directed transport data, so a consumer must name which direction it reads before using the implication evidence.

The route row C exposes only those directed reads, and the transport row H preserves them componentwise across certified presentations. Together they provide exactly the replacement behavior named in Definition 9387.8.

The refusal row A excludes proposition equality, propositional extensionality, and host rewrite. Since none of these is a coordinate of E, they cannot be exported by certificate projection. □

Lean4 ✓ `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerPropextTransportRow`

Theorem 568.5 (EffectiveReplacementLedger axiom-purity boundary)**.** *For every accepted carrier $E = (W, Q, T, A, H, C, P, N)$, the axiom-purity refusal row A is a boundary on the whole packet: any requested read that requires* Classical.choice*,* Quot.sound*,* propext*, an unwitnessed selector, a quotient representative section, or proposition equality is rejected rather than transported through W, Q, T, H, or C.*

Proof. Inspect the carrier inventory. The positive replacement rows are W, Q, and T, while H and C only transport and replay those displayed rows.

The refusal row A is attached to the same package provenance P and local name N. It therefore constrains every public read of the packet, not merely one replacement direction.

A request for one of the excluded operations has no coordinate to project and no continuation route to replay. The only available response is the refusal recorded in A, so the finite ledger preserves the axiom-purity boundary. □

Lean4 ✓ `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerAxiomPurityBoundary`

Theorem 568.6 (EffectiveReplacementLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EffectiveReplacementLedger}^{\uparrow}}$ *surface for Definition 568.1 consists of five finite obligations: carrier exposure for E, choice-witness exactness for W, quotient-transport exactness for Q, propext-transport exactness for T, and axiom-purity non-escape for A over the shared transport, continuation, provenance, and naming rows H, C, P, N.*

Proof. Carrier exposure is projection of the displayed tuple $E = (W, Q, T, A, H, C, P, N)$. The packet lists every replacement, transport, continuation, provenance, and naming row as a finite coordinate.

The three exactness obligations are supplied by the row theorems above. Definition 568.2 binds W to explicit witnesses, Definition 568.3 binds Q to classifier-respecting transport, and Definition 568.4 binds T to direction-indexed proposition transport.

The final obligation is the non-escape boundary of Definition 568.5. Repacking these rows with H, C, P, N gives one finite NameCert packet whose public surface contains replacement data and refusal data, but no classical axiom dependency. □

Lean4 ✓ `BEDC.Derived.EffectiveReplacementLedgerUp.EffectiveReplacementLedgerTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveReplacementLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveReplacementLedger}^{\uparrow}$ packet is generated from explicit choice witnesses, classifier-respecting quotient transport, direction-indexed proposition transport, an axiom-purity refusal row, componentwise hsame and psame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EffectiveReplacementLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 568.1, Definition 568.2, Definition 568.3, Definition 568.4, Definition 568.5, Definition 568.6, and the TasteGate target bind the mature paper scope of finite BHist packet rows, hsame and psame transport, displayed Cont replay, Pkg provenance, local NameCert, refusal of Classical.choice, Quot.sound, and propext, and the public replacement-ledger surface aligned with the standard bridge schema.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EffectiveReplacementLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, machine bridge checking, general quotient construction, proposition equality, classical choice, or an executable audit implementation.

Upgrade path: A machine bridge target would move only the verification axis; the present chapter already records the mature paper surface while the formal axis remains unformalized.

Chapter 569

A Concrete Naming Certificate for $\mathsf{AnalyticContinuationSocket}^{\uparrow}$

The $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ interface turns the analytic-continuation operation of chapter 9341 into a finite operation-socket certificate. It does not assert a global monodromy theorem or a host analytic continuation principle. It names the local source rows, overlap witnesses, continuation output rows, transport routes, and branch-obstruction ledger that a downstream $\mathsf{Holomorphic}^{\uparrow}$ or Riemann-hypothesis-facing consumer may inspect.

Definition 569.1 (AnalyticContinuationSocket carrier)**.** *An* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, G, U, A, O, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{Holomorphic}^{\uparrow}$ *row on the first* $\mathsf{Complex}^{\uparrow}$ *domain;*
2. *G is the overlapping* $\mathsf{Holomorphic}^{\uparrow}$ *row on the second* $\mathsf{Complex}^{\uparrow}$ *domain;*
3. *U is the finite overlap witness row, including the displayed nontrivial overlap and agreement evidence needed by the local identity route;*
4. *A is the analytic-continuation operation row that reads F, G, U and proposes an output continuation;*
5. *O is the output* $\mathsf{Holomorphic}^{\uparrow}$ *row on the displayed union domain;*
6. *B is the branch-obstruction ledger, recording every requested continuation route that leaves the displayed overlap-and-output packet as a* GAP *boundary;*
7. *H stores componentwise* hsame *transports for F, G, U, A, O, B and the local naming rows;*
8. *C stores the finite* Cont *routes by which consumers read the source rows, overlap witness, output row, branch ledger, package provenance, and local naming row;*
9. *P is the* Pkg *provenance over the same finite operation-socket packet;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *row.*

The carrier exposes exactly this local socket. It contains no unrestricted analytic-continuation operator, no global branch selector, no quotient of germs, and no host complex-analysis theorem.

Lean4 (definition) `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier`

Theorem 569.2 (AnalyticContinuationSocket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *surface for Definition 569.1 consists of five finite rows: carrier admission for S; classifier stability for the two holomorphic source rows, the overlap witness, the operation row, and the output row; ledger exactness for the branch-obstruction coordinate B;* Pkg *provenance over F, G, U, A, O, B, H, C; and local naming exactness for N.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier namecert obligations`

Proof. Carrier admission is projection of the finite tuple S. The source, overlap, operation, output, obstruction, transport, continuation, package, and local naming rows are all visible coordinates of the same BHist packet.

Classifier stability is componentwise. Transport of F, G, U preserves the two source rows and the overlap agreement evidence, transport of A, O preserves the proposed continuation output, and transport of B preserves the recorded non-escape boundary.

The package and local naming rows repack these coordinates without adding a host continuation principle. Thus the socket certificate consists exactly of the five displayed obligation rows. □

Theorem 569.3 (AnalyticContinuationSocket overlap transport)**.** *For an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier $S = (F, G, U, A, O, B, H, C, P, N)$, the local overlap witness U transports only along the displayed* hsame *row H and the displayed* Cont *routes C. The transported read preserves the two* $\mathsf{Holomorphic}^{\uparrow}$ *source rows F, G, the* $\mathsf{Complex}^{\uparrow}$ *overlap boundary named by U, the analytic-continuation operation row A, the output row O, the package row P, and the local* NameCert *row N; it does not create a monodromy theorem, global continuation operator, quotient of germs, or branch selector.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketPacket overlap transport`

Proof. Project the carrier. The overlap witness U is displayed in the same finite packet as the two source rows, operation row, output row, transport row, continuation routes, package provenance, and local naming row.

Apply the classifier-stability part of Definition 569.2. It allows the overlap evidence to move only through componentwise hsame transport and the corresponding Cont reads.

The carrier definition contains no row for monodromy, global continuation, quotient-of-germs equality, or branch selection. The transported overlap read therefore remains a local overlap-boundary read. □

Theorem 569.4 (AnalyticContinuationSocket branch ledger exhaustion)**.** *Every branch-sensitive read of an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier is exhausted by the rows $F, G, U, A, O, B, H, C, P, N$. If the read is not a source-row read, overlap-witness read, operation-row read, output-row read, transport read, continuation-route read, package read, or local naming read, then it is recorded in the branch-obstruction ledger B and is not exported as analytic-continuation evidence.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier branch ledger exhaustion`

Proof. Read the request through the finite continuation routes C. The positive cases are exactly the displayed source, overlap, operation, output, transport, package, and naming coordinates.

For a request outside those coordinates, use the branch-obstruction coordinate B. The ledger non-escape theorem records such a request as a GAP boundary rather than as another continuation row.

Since the packet has no hidden analytic bridge or host continuation principle, the two cases exhaust the carrier-visible branch reads. □

Theorem 569.5 (AnalyticContinuationSocket local output handoff)**.** *Every downstream* Holomorphic$^{\uparrow}$ *consumer of the output row O from an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier receives the local handoff packet $F, G, U, A, O, H, C, P, N$ together with the branch-obstruction ledger B. The handoff states that O is the displayed local operation output over the two source rows and overlap witness; it does not state global analytic continuation, monodromy invariance, branch selection, or any Riemann-hypothesis-facing zero-location consequence.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier local output handoff`

Proof. Start with the operation row A and output row O. The carrier places O as the displayed output of the local operation over F, G, U.

Transport and consumer access are supplied only by H and C, then repacked by P and N. Thus a downstream Holomorphic$^{\uparrow}$ consumer receives the same local packet rather than an ambient continuation operator.

The branch ledger B handles every request outside that packet. By branch-ledger exhaustion, such requests stay in the ledger and cannot become monodromy, branch-choice, or zero-location data. □

Theorem 569.6 (AnalyticContinuationSocket ledger non-escape)**.** *Every consumer read exported by an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier factors through the displayed rows $F, G, U, A, O, B, H, C, P, N$. If a read asks for continuation data outside the source rows, overlap witness, proposed output, and displayed transport routes, the request is recorded in B as a* GAP *boundary and is not exported as a holomorphic continuation row.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier ledger nonescape`

Proof. Read the consumer request through the displayed Cont route C. If the request lands in the two source rows, the overlap witness, the operation row, the output row, or the package and naming rows, the carrier supplies exactly that coordinate.

If the request leaves those coordinates, no row of S can supply a continuation witness. The branch-obstruction ledger B is the only remaining classifier-visible coordinate for such a request, so the read is recorded as non-escape evidence rather than as an output continuation. □

Theorem 569.7 (AnalyticContinuationSocket holomorphic handoff)**.** *For an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier $S = (F, G, U, A, O, B, H, C, P, N)$, every downstream* Holomorphic$^{\uparrow}$ *consumer of the output row O also reads the source rows F, G, the overlap witness U, the operation row A, and the transport, continuation, package, and local naming rows H, C, P, N. The handoff exposes no standalone continuation function and no branch choice outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier holomorphic handoff`

Proof. Begin with the output row O. In the carrier definition it appears only as the result coordinate of the operation row A over the displayed source rows F, G and overlap witness U.

Transport through H preserves the same source-overlap-output boundary, and the continuation routes in C expose the output only through that boundary. The package and local naming rows P, N then identify the same finite packet for downstream consumers.

The branch ledger B rules out the remaining escape case. A consumer that asks for a branch choice or continuation function outside the displayed packet reaches B, not another $\mathsf{Holomorphic}^{\uparrow}$ row. □

Theorem 569.8 (AnalyticContinuationSocket overlap classifier stability)**.** *Let $S = (F, G, U, A, O, B, H, C, P, N)$ be an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Transported reads of the source rows F, G and the overlap witness U preserve the local overlap-agreement classifier through the displayed componentwise* hsame *rows in H, the* Cont *routes in C, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *row N. A transported overlap read may replay the named agreement boundary, but it cannot replace that boundary by a global identity theorem, monodromy theorem, quotient-of-germs equality, or branch selector.*

Proof. Start with the carrier of Definition 569.1. The rows F, G, U are named coordinates of the same finite socket packet, and H stores their componentwise hsame transport.

Apply the classifier-stability row of Definition 569.2. Transport of F, G, U preserves the source rows and the local overlap agreement evidence rather than introducing a new continuation object.

Now read the transported overlap through C, P, N. The ledger non-escape theorem Definition 569.6 and the holomorphic handoff theorem Definition 569.7 keep downstream consumers inside the displayed packet, so the overlap classifier remains the named local classifier. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier overlap classifier stability`

Theorem 569.9 (AnalyticContinuationSocket operation-output factorization)**.** *For an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier $S = (F, G, U, A, O, B, H, C, P, N)$, every exported operation-output row O factors through the finite rows F, G, U, A, H, C, P, N. The branch ledger B records exactly the attempted reads that do not factor through those rows. Thus O is a local operation output of the displayed socket, not the value of an unrestricted analytic-continuation operator and not the result of a hidden branch-choice rule.*

Proof. Project the output coordinate O from the carrier in Definition 569.1. By definition it is paired with the operation row A, which reads only the two source rows F, G and the overlap witness U.

Use Definition 569.2 to attach the finite hsame, Cont, Pkg, and local naming rows. These rows provide the only certified route by which a consumer can identify the operation output.

The non-factorizing case is handled by Definition 569.6. Any read outside F, G, U, A, H, C, P, N is recorded in B, and Definition 569.7 prevents that ledger entry from becoming an additional $\mathsf{Holomorphic}^{\uparrow}$ output. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier operation output factorization`

Theorem 569.10 (AnalyticContinuationSocket RH handoff boundary)**.** *Every Riemann-hypothesis-facing zeta-continuation consumer of an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier receives only the finite local operation socket*

$$F, G, U, A, O, H, C, P, N$$

together with the branch-obstruction ledger B. The handoff supplies no monodromy theorem, no global analytic-continuation principle, no quotient of germs, no branch selector, and no zero-location consequence.

Proof. Begin with the holomorphic handoff of Definition 569.7. It exposes O only through the source rows, overlap witness, operation row, transport, continuation, package, and local naming rows.

Add the local NameCert surface from Definition 569.2. The Riemann-hypothesis-facing consumer can inspect carrier admission, classifier stability, branch-ledger exactness, package provenance, and local naming, all within the same finite carrier of Definition 569.1.

Finally apply Definition 569.6. Requests for monodromy, branch selection, global continuation, or zero-location consequences leave the displayed packet and are therefore recorded only as branch-obstruction data in B. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier rh handoff boundary`

Theorem 569.11 (AnalyticContinuationSocket root source overlap)**.** *For an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier $S = (F, G, U, A, O, B, H, C, P, N)$, every source-overlap read used by a root continuation consumer factors through the displayed source rows F, G, the finite overlap witness U, the componentwise transport row H, the displayed continuation routes C, the package provenance P, and the local* NameCert *row N. The read may reach the operation row A and output row O only through that source-overlap packet.*

Proof. Project the carrier and the local NameCert obligations. They list F, G, U as the only source-overlap coordinates and place H, C, P, N as the only transport, route, provenance, and naming rows available to a consumer.

Apply overlap classifier stability. Transported source and overlap reads preserve the local agreement boundary through H and are replayed only through the displayed routes in C.

Finally use holomorphic handoff and the RH handoff boundary. A downstream consumer can inspect A and O only after the same source-overlap packet has been displayed, so no hidden continuation source is introduced. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier root source overlap`

Theorem 569.12 (AnalyticContinuationSocket root branch ledger)**.** *Every nonlocal continuation request made against an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier lands in the branch-obstruction ledger B. Such a request may be transported, replayed, packaged, and named through H, C, P, N, but it is not exported as a source row, overlap witness, operation row, output row, holomorphic continuation row, branch selector, or zero-location datum.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier root branch ledger`

Proof. Start with ledger non-escape. It sends every request outside the displayed source, overlap, operation, output, transport, route, provenance, and naming coordinates to the branch-obstruction row B.

Use the ledger obligation and zeta branch-obstruction exactness. These theorems keep branch-sensitive and Riemann-hypothesis-facing requests in B rather than exporting them as continuation or zero-location evidence.

The RH handoff boundary then removes the remaining escape paths. Monodromy, branch selection, global continuation, and zero-location conclusions are not consumer-visible coordinates of the root socket. □

Theorem 569.13 (AnalyticContinuationSocket branch-obstruction consumer lock)**.** *Every nonlocal continuation request made by a downstream consumer of an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier is locked to the branch-obstruction row B. The request may be transported and named through H, C, P, N, but it cannot be exported as a holomorphic continuation row, source row, overlap witness, operation row, output row, branch selector, monodromy datum, or zero-location consequence.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier branch obstruction consumer lock`

Proof. Start with the root branch ledger. It sends every nonlocal continuation request to B and denies export as source, overlap, operation, output, holomorphic continuation, branch-selection, or zero-location data.

Apply branch-ledger exhaustion and ledger non-escape. They identify the positive carrier-visible routes and record every request outside those routes as branch-obstruction data rather than as continuation evidence.

Finally use overlap transport and the RH handoff boundary. Transport through H, C, P, N preserves the same ledger coordinate, and Riemann-hypothesis-facing consumers receive no zero-location or global-continuation consequence from that coordinate. □

Theorem 569.14 (AnalyticContinuationSocket zeta locality boundary)**.** *Every zeta-facing consumer of an accepted* AnalyticContinuationSocket$^{\uparrow}$ *carrier receives only the local socket rows*

$$F, G, U, A, O, H, C, P, N$$

together with the branch ledger B. The consumer receives no monodromy theorem, global analytic-continuation principle, branch selector, quotient-of-germs equality, or zero-location consequence.

Proof. Read the RH handoff boundary first. It exposes the finite local operation socket and the branch-obstruction ledger, while excluding monodromy, global continuation, quotient-of-germs equality, branch selection, and zero-location conclusions.

Use the root source-overlap theorem and local output handoff. They force the output row O to pass through F, G, U, A, H, C, P, N before any downstream consumer can inspect it.

Now apply the root branch ledger. A zeta-facing request outside that local socket is routed to B, where it remains branch-obstruction data rather than a zero-location consequence. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier zeta locality boundary`

Theorem 569.15 (AnalyticContinuationSocket output-branch separation)**.** *For an accepted* AnalyticContinuationS *carrier, the output row O factors through*

$$F, G, U, A, H, C, P, N,$$

while branch-sensitive requests factor through B. These two routes do not collapse into a global analytic-continuation principle, branch selector, monodromy theorem, quotient-of-germs equality, or zero-location theorem.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier output branch separation`

Proof. Project the operation-output factorization. The output coordinate O is a local operation result over F, G, U, A and is certified only through H, C, P, N.

Next apply branch-ledger exhaustion, ledger non-escape, and the root branch ledger. Requests that do not factor through the source-overlap-output route are recorded in B.

Finally combine holomorphic handoff with the RH handoff boundary. The output route supplies a local $\mathsf{Holomorphic}^{\uparrow}$ handoff, while the branch route remains obstruction data; the two routes therefore do not yield a global continuation or zero-location principle. □

Theorem 569.16 (AnalyticContinuationSocket root downstream unblock)**.** *The finite root surface exported by an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier to* $\mathsf{Holomorphic}^{\uparrow}$ *and zeta-continuation consumers is exactly*

$$F, G, U, A, O, B, H, C, P, N.$$

The package combines the source rows, overlap witness, operation row, output row, branch ledger, componentwise transports, continuation routes, package provenance, and local naming row, and it excludes global monodromy, branch selection, quotient-of-germs equality, and zero-location consequences.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier root downstream unblock`

Proof. First combine the root source-overlap theorem with holomorphic handoff. The consumer-visible output row O is reached only through F, G, U, A, H, C, P, N.

Next apply the root branch-ledger theorem. Every request outside that source-overlap-output path lands in B and remains obstruction data.

Finally repack the two paths with the RH handoff boundary. The downstream consumer receives one finite surface, $F, G, U, A, O, B, H, C, P, N$, and no unlisted analytic-continuation principle. □

Theorem 569.17 (AnalyticContinuationSocket vision concretization)**.** *The analytic-continuation operation surface of chapter 9341 is concretized by* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *as a finite* BHist *operation-socket packet over holomorphic source rows, overlap witnesses, continuation outputs, componentwise transport, continuation routes, package provenance, local naming, and branch-obstruction non-escape.*

Proof. The carrier of Definition 569.1 supplies the finite operation-socket packet. The NameCert obligations of Definition 569.2 bind carrier admission, classifier stability, branch-ledger exactness, package provenance, and local naming.

The ledger non-escape theorem prevents undisplayed continuation routes from becoming output rows, and the holomorphic handoff theorem shows how downstream $\mathsf{Holomorphic}^{\uparrow}$ consumers read the displayed continuation output. These clauses are exactly the finite BEDC concretization of the operation surface. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier vision concretization`

Theorem 569.18 (AnalyticContinuationSocket carrier source obligation)**.** *Carrier admission for an accepted* AnalyticContinuationSocket$^{\uparrow}$ *packet projects the finite rows*

$$F, G, U, A, O, B, H, C, P, N.$$

These rows expose the two Holomorphic$^{\uparrow}$ *source rows, the finite overlap witness, the operation row, the output row, the branch-obstruction ledger, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. They do not expose a global continuation operator, branch selector, quotient of germs, or host complex-analysis theorem.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier carrier source obligation`

Proof. Project the carrier of Definition 569.1. The finite tuple lists exactly the source, overlap, operation, output, obstruction, transport, continuation, package, and local naming rows.

Apply the NameCert obligations of Definition 569.2. Carrier admission is one of the displayed obligations, and the remaining obligation rows are read only through the same finite packet.

The ledger non-escape theorem Definition 569.6 rules out any continuation read outside these rows. Therefore the admitted carrier supplies the stated source surface and no host analytic-continuation principle. □

Theorem 569.19 (AnalyticContinuationSocket classifier stability obligation)**.** *Componentwise* hsame *transport for accepted* AnalyticContinuationSocket$^{\uparrow}$ *packets preserves the source rows F, G, the overlap witness U, the operation row A, the output row O, and the branch-obstruction ledger B. Under this transport the classifier may replay the displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, but it cannot replace them with an ambient analytic-continuation theorem or a hidden branch choice.*

Proof. Start with the transport row H in Definition 569.1. It stores the componentwise hsame data for the source, overlap, operation, output, obstruction, and local naming rows.

Use the classifier-stability clause of Definition 569.2. Transport of F, G, U preserves the source and overlap evidence, transport of A, O preserves the operation-output boundary, and transport of B preserves the recorded obstruction ledger.

The holomorphic handoff of Definition 569.7 then keeps downstream Holomorphic$^{\uparrow}$ consumers on the same displayed packet. Since the carrier has no branch selector or host continuation theorem, the transported classifier cannot export one. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier classifier stability obligation`

Theorem 569.20 (AnalyticContinuationSocket ledger exactness obligation)**.** *Every branch-obstruction read of an accepted* AnalyticContinuationSocket$^{\uparrow}$ *packet is recorded in the ledger row B. Such a read may be transported by H, replayed by C, named by P and N, and compared with the source, overlap, operation, and output rows, but it cannot become a new* Holomorphic$^{\uparrow}$ *output row or a continuation output outside O.*

Proof. Project the branch-obstruction coordinate B from Definition 569.1. It is the only displayed row assigned to continuation requests that leave the source, overlap, operation, and output packet.

Apply the ledger non-escape theorem Definition 569.6. Reads outside the displayed continuation packet are recorded as GAP boundary data in B rather than exported as holomorphic continuation rows.

Finally use the holomorphic handoff of Definition 569.7. Downstream $\mathsf{Holomorphic}^{\uparrow}$ consumers receive O only through F, G, U, A, H, C, P, N, so a branch-obstruction read remains ledger data and cannot turn into another output row. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier ledger exactness obligation`

Theorem 569.21 (AnalyticContinuationSocket operation landing exactness)**.** *For an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packet* $S = (F, G, U, A, O, B, H, C, P, N)$*, every exported operation landing at the output row* O *factors through the finite rows* F, G, U, A, H, C, P, N*. The row* O *may be transported, replayed, packaged, and locally named by those rows, but it cannot be replaced by a global analytic-continuation operator, hidden branch rule, monodromy theorem, or quotient-of-germs equality.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier operation landing exactness`

Proof. Project the carrier of Definition 569.1. The only output coordinate is O, and it is displayed beside the source rows F, G, overlap witness U, operation row A, transport row H, continuation routes C, package provenance P, and local naming row N.

Apply the local NameCert obligations of Definition 569.2. Carrier admission, classifier stability, package provenance, and local naming all read the same finite socket packet.

Use overlap transport and holomorphic handoff, Definition 569.3 and Definition 569.7, to keep every downstream read of O on the displayed source-overlap-operation route. The ledger exactness obligation Definition 569.20 and ledger non-escape theorem Definition 569.6 send every non-factorizing continuation request to B rather than to another output row. □

Theorem 569.22 (AnalyticContinuationSocket overlap classifier route)**.** *For an accepted* AnalyticContinuationSock *packet* $S = (F, G, U, A, O, B, H, C, P, N)$*, every accepted overlap classifier read factors through* F, G, U, H, C, P, N*. The read may identify the operation row* A *and output row* O *only after the displayed source-overlap classifier has been transported through* H*, replayed through* C*, packaged by* P*, and named by* N*.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier overlap classifier route`

Proof. Start with the carrier definition and the local NameCert obligations. They place the source rows F, G and overlap witness U in the same finite packet as the transport, route, package, and naming rows H, C, P, N.

Apply overlap transport Definition 569.3. It keeps the overlap witness on the displayed hsame and Cont paths and forbids replacement by monodromy, quotient-of-germs equality, or a branch selector.

Finally use holomorphic handoff together with ledger exactness and ledger non-escape. Reads that reach A or O do so through the same overlap route, while reads outside F, G, U, H, C, P, N are recorded in B and are not accepted as overlap-classifier output. □

Theorem 569.23 (AnalyticContinuationSocket branch obstruction nonexport)**.** *For an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packet* $S = (F, G, U, A, O, B, H, C, P, N)$*, every nonlocal continuation request is recorded in the branch-obstruction ledger* B*. Such a request may be transported by* H*, replayed by* C*, packaged by* P*, and locally named by* N*, but it cannot export a* $\mathsf{Holomorphic}^{\uparrow}$ *output, branch selector, quotient-of-germs equality, monodromy theorem, or zero-location datum.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier branch obstruction nonexport`

Proof. Project the branch-obstruction coordinate from Definition 569.1. The carrier assigns nonlocal continuation requests to B and lists no separate coordinate for branch selection, monodromy, quotient-of-germs equality, or zero-location data.

Use the local NameCert obligations and overlap transport to identify the permitted finite reads. The source rows, overlap witness, operation row, output row, transport routes, package provenance, and local naming are all read from $F, G, U, A, O, H, C, P, N$.

Apply ledger non-escape and the ledger exactness obligation. A request outside those rows remains branch-obstruction data in B rather than an exported continuation witness.

The holomorphic handoff then rules out the remaining export path: downstream $\mathsf{Holomorphic}^{\uparrow}$ consumers receive the displayed local output O only through F, G, U, A, H, C, P, N, so a nonlocal branch request cannot become a holomorphic output or a zero-location datum. □

Theorem 569.24 (AnalyticContinuationSocket carrier obligation)**.** *For an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packet*

$$S = (F, G, U, A, O, B, H, C, P, N),$$

carrier admission is exactly the projection of the source rows F, G, the overlap witness U, the operation row A, the output row O, the branch-obstruction ledger B, the componentwise transport row H, the continuation-route row C, the provenance row P, and the local naming row N.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier carrier obligation`

Proof. Project the carrier definition Definition 569.1. It lists precisely the source, overlap, operation, output, obstruction, transport, continuation, provenance, and naming rows of the finite socket.

Apply the NameCert obligations of Definition 569.2. Carrier admission is one of the five displayed obligation rows, and the remaining rows are read from the same packet.

The ledger non-escape theorem blocks any hidden continuation operator, branch selector, quotient of germs, or host complex-analysis theorem. Thus the carrier obligation is exactly the displayed finite projection. □

Theorem 569.25 (AnalyticContinuationSocket classifier obligation)**.** *Let $S = (F, G, U, A, O, B, H, C, P, N)$ and $S' = (F', G', U', A', O', B', H', C', P', N')$ be accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packets. Classifier preservation is componentwise: the source rows F, G compare with F', G', the overlap witnesses U, U' compare through the displayed agreement boundary, the operation and output rows A, O compare with A', O' through the stored* hsame *and* Cont *routes, and the branch ledgers B, B' compare only as obstruction ledger rows.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier classifier obligation`

Proof. Use the classifier-stability row of Definition 569.2. It preserves the two source rows, the overlap witness, the operation row, the output row, and the obstruction ledger by componentwise transport.

The overlap classifier stability theorem keeps the F, G, U comparison on the local overlap-agreement boundary. The operation-output factorization theorem keeps the A, O comparison on the displayed operation output rather than on an unrestricted continuation operator.

Finally, ledger non-escape and the holomorphic handoff prevent an obstruction comparison from becoming another $\mathsf{Holomorphic}^{\uparrow}$ output. Hence the classifier obligation preserves overlap and output through hsame and Cont only. □

Theorem 569.26 (AnalyticContinuationSocket ledger obligation)**.** *Every branch-sensitive consumer read of an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packet factors through the finite rows*

$$F, G, U, A, O, B, H, C, P, N.$$

If the read asks for a continuation output not displayed by O through the source, overlap, operation, transport, route, provenance, and naming rows, the request remains in the branch-obstruction ledger B and is not exported as a continuation output.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier ledger obligation`

Proof. Start from the ledger exactness obligation Definition 569.20. It assigns branch-obstruction reads to B and keeps them from becoming output rows outside O.

Apply ledger non-escape to exhaust the carrier-visible rows. The holomorphic handoff then shows that downstream $\mathsf{Holomorphic}^{\uparrow}$ consumers obtain the displayed output only through F, G, U, A, H, C, P, N.

Use overlap classifier stability and operation-output factorization for the two positive paths through the socket. All remaining branch-sensitive reads are ledger entries, so no undisplayed continuation output escapes the packet. □

Theorem 569.27 (AnalyticContinuationSocket zeta branch-obstruction exactness)**.** *Let $S = (F, G, U, A, O, B, H, C, P, N)$ be an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Every zeta-continuation or Riemann-hypothesis-facing request that asks for nontrivial zero-location data, monodromy data, a branch selector, or continuation data beyond the displayed local socket is recorded in the branch-obstruction ledger B. Such a request is not exported as continuation evidence, zero-location evidence, or a new* $\mathsf{Holomorphic}^{\uparrow}$ *output row.*

Proof. Project the carrier of Definition 569.1. Its consumer-readable coordinates are exactly $F, G, U, A, O, B, H, C, P, N$, with B assigned to branch-obstruction accounting.

Apply Definition 569.6 and Definition 569.26. A request outside the source rows, overlap witness, local operation output, displayed transports, routes, provenance, and naming rows has no output coordinate other than the ledger row B.

Finally use Definition 569.10. The Riemann-hypothesis-facing handoff excludes monodromy, branch selection, global continuation, and zero-location consequences, so the branch-sensitive request remains a ledger entry rather than exported continuation evidence. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier zeta branch obstruction exactness`

Theorem 569.28 (AnalyticContinuationSocket overlap-to-zeta handoff)**.** *Let $S = (F, G, U, A, O, B, H, C, P, N)$ be an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Every zeta-continuation consumer that reads the socket output receives only the overlap-and-output handoff*

$$F, G, U, A, O, H, C, P, N$$

together with the ledger row B. The handoff may certify that O is the displayed local output over the overlap witness U, but it supplies no nontrivial zero-location datum, monodromy theorem, branch selector, or continuation evidence outside O.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier overlap to zeta handoff`

Proof. Start from Definition 569.7. The output row O is exposed only through the source rows F, G, the overlap witness U, the operation row A, and the transport, route, provenance, and local naming rows H, C, P, N.

Use the carrier inventory in Definition 569.1. It contains no separate coordinate for a zeta branch selector, monodromy theorem, zero-location conclusion, or continuation object beyond the displayed operation output.

Apply Definition 569.10 and Definition 569.26. The zeta-facing consumer therefore reads the local overlap/output socket, while any request outside that socket is recorded in B and not exported as continuation evidence. □

Theorem 569.29 (AnalyticContinuationSocket zero-location non-read)**.** *Let $S = (F, G, U, A, O, B, H, C, P, N)$ be an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. No Riemann-hypothesis-facing consumer can read a nontrivial zero-location conclusion from S. The consumer can read only the local overlap/output socket, and every zero-location, monodromy, branch-selector, or global-continuation request is recorded in B rather than exported as continuation evidence.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier no zero location read`

Proof. The carrier definition Definition 569.1 lists no zero-location coordinate. Its output coordinate is the local $\mathsf{Holomorphic}^{\uparrow}$ row O, read through the finite overlap and operation packet.

Apply Definition 569.7 and Definition 569.10. These results give the Riemann-hypothesis-facing consumer only the local socket $F, G, U, A, O, H, C, P, N$ and the branch-obstruction ledger B.

The remaining cases are exactly the requests covered by Definition 569.6 and Definition 569.26. They stay in B, so no zero-location read can be exported by the socket. □

Theorem 569.30 (AnalyticContinuationSocket carrier admission)**.** *For every accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier*

$$S = (F, G, U, A, O, B, H, C, P, N),$$

the carrier admitted over BHist *is exactly the finite socket packet with* $\mathsf{Complex}^{\uparrow}$*-based* $\mathsf{Holomorphic}^{\uparrow}$ *source rows F, G, overlap witness U, local operation row A, output* $\mathsf{Holomorphic}^{\uparrow}$ *row O, branch ledger B, componentwise* hsame *transports H, finite* Cont *routes C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *row N. No admitted carrier supplies a quotient of germs, selected analytic continuation, global branch selector, or ambient complex-analysis theorem outside those rows.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier carrier admission`

Proof. Project the displayed carrier Definition 569.1. Its rows are already indexed by BHist and expose the two $\mathsf{Complex}^{\uparrow}$-based $\mathsf{Holomorphic}^{\uparrow}$ sources, the overlap witness, operation, output, branch ledger, transports, routes, package provenance, and local name.

Use the carrier obligation Definition 569.24. It identifies carrier admission with the same finite projection and leaves no coordinate from which an external continuation theorem can be read.

Finally apply ledger non-escape and the holomorphic handoff. The admitted carrier is therefore the local socket packet itself; any branch-sensitive request outside the displayed output remains ledger data rather than carrier data. □

Theorem 569.31 (AnalyticContinuationSocket classifier transport)**.** *Let*

$$S = (F, G, U, A, O, B, H, C, P, N)$$

be an accepted $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Classifier transport through* S *is componentwise over the* BHist *socket rows: the* $\mathsf{Complex}^{\uparrow}$*-based* $\mathsf{Holomorphic}^{\uparrow}$ *source rows* F, G*, overlap witness* U*, local operation row* A*, output row* O*, and branch ledger* B *are transported only through the displayed* hsame *row* H*, the finite* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier classifier transport`

Proof. Start with the classifier obligation Definition 569.25. It compares the source rows, overlap witness, operation row, output row, and branch ledger only componentwise.

The overlap classifier stability theorem keeps the F, G, U comparison on the displayed overlap boundary, while the operation-output factorization theorem keeps A and O attached to the same local socket route. The stored hsame and Cont rows are the only transport surfaces used in those comparisons.

The branch ledger is transported as ledger data. By the ledger non-escape and holomorphic handoff theorems, transport of B cannot become a new $\mathsf{Holomorphic}^{\uparrow}$ output, a global branch choice, or an external analytic-continuation result. □

Theorem 569.32 (AnalyticContinuationSocket branch-ledger exactness)**.** *For an accepted* AnalyticContinuationSoc *carrier*

$$S = (F, G, U, A, O, B, H, C, P, N),$$

every branch-sensitive $\mathsf{Complex}^{\uparrow}$*,* $\mathsf{Holomorphic}^{\uparrow}$*, or downstream socket consumer read is routed through the displayed ledger* B *unless it is already the local output row* O *transported from the source rows* F, G *across* U *by* A, H, C, P, N*. Thus the branch ledger is exact for the packet: it records precisely the requests not exported by the local holomorphic output, and it asserts no completed analytic-continuation theorem outside the finite socket.*

Proof. Begin with the ledger obligation Definition 569.26. It places every branch-sensitive read either in the displayed local output route or in the ledger row B.

Use carrier admission and classifier transport to keep this split inside the same BHist packet. The rows $F, G, U, A, O, H, C, P, N$ define the positive local-output route, while B records requests whose branch data is not displayed by that route.

Finally apply the zeta branch-obstruction exactness and overlap-to-zeta handoff theorems. They show that zero-location, monodromy, branch-selector, and global-continuation requests remain ledger entries, so the exactness claim does not assert analytic continuation as a completed external theorem. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier branch ledger exactness`

Theorem 569.33 (AnalyticContinuationSocket overlap source obligation)**.** *Let*

$$S = (F, G, U, A, O, B, H, C, P, N)$$

be an accepted $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Every accepted socket source overlap is read only through the* $\mathsf{Complex}^{\uparrow}$ *and* $\mathsf{Holomorphic}^{\uparrow}$ *source rows* F, G*, the overlap classifier row* U*, componentwise* hsame *transport* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$

row N. No overlap source read supplies a quotient of germs, selected branch, global continuation, or analytic-continuation oracle outside those rows.

Proof. Start with carrier admission. The carrier of Definition 569.1 exposes the source rows F, G and the overlap row U before any operation output is read.

Use the NameCert obligations and classifier transport. They keep the overlap classifier attached to the displayed $\mathsf{Complex}^{\uparrow}$ and $\mathsf{Holomorphic}^{\uparrow}$ source rows through H, C, P, N.

Finally apply branch ledger exactness and overlap-to-zeta handoff. Requests outside the displayed overlap source route are recorded in B rather than exported as source-overlap evidence, so the listed rows exhaust the obligation. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier overlap source obligation`

Theorem 569.34 (AnalyticContinuationSocket branch ledger obligation)**.** *For an accepted* AnalyticContinuationSo *carrier*

$$S = (F, G, U, A, O, B, H, C, P, N),$$

branch-ledger reads are exact for the displayed overlap row U, branch tag or obstruction row B, local operation row A, and local output row O. The ledger exports only the recorded split between local output and obstruction; it exports no analytic-continuation oracle, global monodromy theorem, branch selector, or zero-location datum.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier branch ledger obligation`

Proof. Begin with Definition 569.32. It states that branch-sensitive reads are routed either to the local output path through O or to the ledger row B.

Use ledger exhaustion and local output handoff to identify the positive route. The rows $F, G, U, A, O, H, C, P, N$ are exactly the displayed overlap, operation, output, transport, route, provenance, and naming rows.

Now use zeta branch-obstruction exactness. Zero-location, monodromy, branch-selector, and global-continuation requests remain obstruction entries in B, so the ledger does not export an analytic-continuation oracle. □

Theorem 569.35 (AnalyticContinuationSocket operation handoff obligation)**.** *Every* AnalyticContinuationOperatic *or* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *consumer of an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier*

$$S = (F, G, U, A, O, B, H, C, P, N)$$

must factor through the source-overlap ledger F, G, U, H, C, P, N and the branch ledger B before reading the operation output O. Such a consumer may receive the local operation handoff A, O over the displayed overlap, but it cannot bypass the source and branch ledgers to obtain global analytic continuation, monodromy, branch selection, or zero-location information.

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier operation handoff obligation`

Proof. First apply the overlap source obligation. It makes every accepted source-overlap read pass through F, G, U, H, C, P, N.

Next apply the branch ledger obligation. It accounts for every branch-sensitive request before the operation output is consumed, assigning nonlocal requests to B.

Finally use the holomorphic handoff and overlap-to-zeta handoff. They route the local operation output through A, O after the source and branch ledgers have been read, while zeta-facing obstruction exactness blocks any bypass to global continuation or zero-location data. □

Theorem 569.36 (AnalyticContinuationSocket public export)**.** *The public* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *export of an accepted carrier*

$$S = (F, G, U, A, O, B, H, C, P, N)$$

is exactly the finite socket packet consisting of the source rows F, G*, overlap witness* U*, operation row* A*, output row* O*, branch ledger* B*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *row* N*. The export contains no monodromy theorem, quotient-of-germs object, branch selector, or zero-location consequence beyond those rows.*

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier public export`

Proof. Start with the NameCert obligations for Definition 569.1. They expose the accepted socket as the displayed tuple $S = (F, G, U, A, O, B, H, C, P, N)$ and require classifier stability for the source, overlap, operation, output, branch, transport, route, provenance, and naming rows.

Use branch-ledger exactness together with holomorphic handoff and overlap classifier stability. These results keep every source-overlap read inside F, G, U, H, C, P, N and keep every branch-sensitive request either in the local output route through A, O or in the branch ledger B.

Next apply operation-output factorization, the RH handoff boundary, and the operation handoff obligation. They show that downstream operation and zeta-facing consumers must pass through the same finite rows before receiving the output, so the exported object is the packet itself.

The exclusions follow from the same routing boundary. Monodromy, quotient-of-germs data, branch selection, and zero-location conclusions are not coordinates of S and therefore cannot be projected from the public socket export. □

Theorem 569.37 (AnalyticContinuationSocket bridge route)**.** *The public export of Definition 569.36, the operation-output factorization of Definition 569.9, the RH handoff boundary of Definition 569.10, and the branch-ledger exhaustion of Definition 569.4 assemble the finite bridge route from local* $\mathsf{Holomorphic}^{\uparrow}$ *overlap data to zeta-facing continuation consumers. For an accepted carrier*

$$S = (F, G, U, A, O, B, H, C, P, N),$$

every such consumer must pass through the source rows, overlap witness, operation row, output row, branch ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}^{\uparrow}}$ *row before receiving continuation-facing output. The route exports no monodromy theorem, branch choice, quotient-of-germs object, or zero-location claim.*

Proof. Begin with the public export. It identifies the socket surface with the displayed rows $F, G, U, A, O, B, H, C, P, N$.

Next use operation-output factorization and the RH handoff boundary. They force every zeta-facing consumer to read the operation output only after the local overlap and branch rows have been exposed.

Finally apply branch-ledger exhaustion. It routes every branch-sensitive request either through the displayed output row or through the obstruction ledger, so the bridge route remains finite and contains no monodromy, branch selector, quotient germ, or zero-location coordinate. □

Lean4 ✓ `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier bridge route`

↪ variant `BEDC.Derived.AnalyticContinuationSocketUp.AnalyticContinuationSocketCarrier bridge route certificate`

The sibling dependency lattice is part of the socket surface: chapter 68 supplies the two Holomorphic$^{\uparrow}$ source rows and output row read by the branch and overlap ledger, while chapter 588 consumes the same finite continuation-facing route as zeta-facing operation data. Thus the dependency is the branch/overlap ledger together with the continuation operation route, not a detached bibliographic citation.

Closure status for $\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}\uparrow}$.

Constructive story: An AnalyticContinuationSocket$^{\uparrow}$ packet is generated from two Holomorphic$^{\uparrow}$ source rows, a finite overlap witness, an analytic-continuation operation row, an output Holomorphic$^{\uparrow}$ row, a branch-obstruction ledger, displayed hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}\uparrow}$, maturePackageClosed)

Scope closed: The mature finite socket packet is the carrier $S = (F, G, U, A, O, B, H, C, P, N)$ together with the public bridge route, branch ledger exactness, source-overlap and operation handoffs, hsame transport, Cont replay, Pkg provenance, local NameCert rows, and the sibling dependency lattice through Holomorphic$^{\uparrow}$ and ZetaContinuationApplication$^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AnalyticContinuationSocket}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close global monodromy, quotient-of-germs construction, branch selection, RH zero-location consequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, branch-ledger exactness, package provenance, and local naming for the same finite socket packet.

Chapter 570

A Concrete Naming Certificate for $\mathsf{ZetaContinuationSocket}^{\uparrow}$

The $\mathsf{ZetaContinuationSocket}^{\uparrow}$ packet turns the Riemann-zeta continuation narrative into a finite local certificate. It reads the already displayed $\mathsf{ZetaBasic}^{\uparrow}$ source, the eta half-plane handoff, the $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ operation boundary, the pole and functional-equation ledgers, the $\mathsf{Gamma}^{\uparrow}$ factor interface, and the trivial-zero ledger as named rows of one BHist packet. It does not assert RH, global zero classification, a complete constructive Gamma chapter, or a branch selector outside the displayed continuation routes.

The witness-facing handoff is the sibling packet of chapter 580, while this socket chapter supplies the boundary rows before the witness adds its package-provenance coordinate.

Definition 570.1 (ZetaContinuationSocket carrier)**.** *A* $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (B, E, A, P, F, T, G, H, C, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ZetaBasic}^{\uparrow}$ *source row on the half-plane $\Re(s) > 1$;*
2. *E is the eta half-plane row, including the eta-series value, the denominator boundary $1 - 2^{1-s}$, and the overlap agreement with B where both rows are defined;*
3. *A is the* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *handoff row that reads B, E and the finite overlap evidence as a local continuation socket;*
4. *P is the pole-residue row at $s = 1$, recording the simple-pole boundary and residue* $\mathsf{E1(e)}$ *without exporting a regular value there;*
5. *F is the finite functional-equation factor ledger, containing the rows for 2^s, π^{s-1}, $\sin(\pi s/2)$, the reflected zeta input $1 - s$, and their displayed product route;*
6. *T is the trivial-zero ledger, recording the negative-even inputs whose sine factor is zero and the companion rows that remain finite at those inputs;*
7. *G is the* $\mathsf{Gamma}^{\uparrow}$ *factor interface row, naming the input boundary for $\Gamma(1 - s)$ and recording poles as ledger entries rather than as zeta values;*
8. *H stores the displayed* hsame *transports among the zeta-basic source, eta row, analytic-continuation socket row, pole row, functional-equation factor rows, trivial-zero ledger, Gamma factor interface, and local naming row;*

9. *C stores the finite* Cont *routes by which consumers read the source half-plane, eta handoff, pole boundary, functional-equation factorization, Gamma interface, trivial-zero ledger, package provenance, and local naming row;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationSocket}^{\uparrow}}$ *row.*

The carrier exposes exactly this finite socket. It contains no unrestricted meromorphic-continuation principle, no proof that all nontrivial zeros lie on the critical line, no complete $\mathsf{Gamma}^{\uparrow}$ *construction, and no hidden analytic branch choice.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationSocketUp.ZetaContinuationSocketTasteGate single carrier alignment`

Theorem 570.2 (ZetaContinuationSocket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationSocket}^{\uparrow}}$ *surface for Definition 570.1 consists of five finite rows: carrier admission for Z; classifier stability for B, E, A, P, F, T, G under the displayed* hsame *transports in H; ledger exactness for the pole, branch, Gamma-boundary, and trivial-zero coordinates;* Pkg *provenance through the continuation routes in C; and local naming exactness for N.*

Proof. Carrier admission is projection of the tuple Z. Each analytic ingredient named by the socket is a visible coordinate, so the certificate never reads an unlisted continuation object.

Classifier stability is componentwise. The displayed hsame transports in H preserve the zeta-basic source, the eta handoff, the analytic-continuation socket boundary, the pole-residue row, the functional-equation factor rows, the trivial-zero ledger, and the Gamma-boundary row.

Ledger exactness is also coordinatewise. Requests at $s = 1$ land in P, Gamma inputs at a pole land in G, negative-even zero requests land in T, and branch requests outside the finite routes land in the obstruction part of the analytic-continuation socket row A.

The continuation routes in C and the local naming row N then repack the same coordinates as a finite NameCert surface. No row introduces a global continuation theorem or an RH-facing zero-location conclusion. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationSocketUp.ZetaContinuationSocketNameCertObligations`

Theorem 570.3 (ZetaContinuationSocket ledger non-escape)**.** *Every consumer read exported by an accepted* $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *carrier factors through the displayed rows $B, E, A, P, F, T, G, H, C, N$. A read that asks for a zeta value at the pole, a Gamma value at a Gamma pole, a continuation route outside the displayed eta and functional-equation routes, or a nontrivial-zero location claim is recorded in the corresponding ledger coordinate and is not exported as a zeta-continuation value.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationSocketUp.ZetaContinuationSocketLedger nonescape`

Proof. Read the consumer request through the finite routes in C. If it lands in the half-plane source row, eta handoff, analytic-continuation socket row, pole row, functional-equation factor ledger, trivial-zero ledger, Gamma-boundary row, transport row, or local naming row, the carrier returns that coordinate.

It remains to inspect requests outside those coordinates. A value at $s = 1$ is blocked by P, a Gamma-pole request is blocked by G, a branch request outside the displayed continuation routes is blocked by A, and a nontrivial-zero location assertion has no carrier coordinate. Each case is therefore a ledger read rather than an exported zeta value. □

Theorem 570.4 (ZetaContinuationSocket analytic handoff)**.** *For an accepted* ZetaContinuationSocket$^{\uparrow}$ *carrier* $Z = (B, E, A, P, F, T, G, H, C, N)$*, every downstream consumer of the continued zeta row also reads the* ZetaBasic$^{\uparrow}$ *source row* B*, the eta half-plane row* E*, the* AnalyticContinuationSocket$^{\uparrow}$ *handoff row* A*, the pole-residue row* P*, the functional-equation factor ledger* F*, the trivial-zero ledger* T*, the* Gamma$^{\uparrow}$ *interface* G*, and the transport, continuation, and naming rows* H, C, N*.*

Proof. Begin with a downstream continued-zeta read. By the carrier definition, such a read reaches the output only through the continuation socket coordinate A and its finite route in C.

The source side of A is the pair of rows B, E together with their overlap evidence. Transport through H preserves this source boundary, so the downstream read keeps the zeta-basic and eta inputs visible.

The remaining analytic boundary is supplied by P, F, T, G. These coordinates record the pole at 1, the functional-equation factors, the trivial-zero sine cancellations, and the Gamma-boundary interface. Thus the handoff exports a finite packet, not a standalone meromorphic function or a hidden branch choice. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationSocketUp.ZetaContinuationSocketCarrier analytic handoff`

Theorem 570.5 (ZetaContinuationSocket vision concretization)**.** *The zeta continuation application is concretized by* ZetaContinuationSocket$^{\uparrow}$ *as a finite* BHist *socket packet over the* ZetaBasic$^{\uparrow}$ *source, eta half-plane handoff,* AnalyticContinuationSocket$^{\uparrow}$ *route, pole-residue row, functional-equation factor ledger,* Gamma$^{\uparrow}$ *interface, trivial-zero ledger, displayed* hsame *transports, finite* Cont *routes, and local* NameCert *row.*

Proof. The carrier of Definition 570.1 supplies the finite packet. The NameCert obligations of Definition 570.2 bind its carrier, classifier, ledger, route, and naming rows.

The non-escape theorem prevents pole, Gamma-boundary, branch, and nontrivial-zero-location requests from becoming continuation values. The analytic handoff theorem identifies the exact rows a downstream zeta-continuation consumer must read. These clauses realize the zeta continuation application as a finite BEDC socket certificate. □

Closure status for NameCert$_{\text{ZetaContinuationSocket}^{\uparrow}}$**.**

Constructive story: A ZetaContinuationSocket$^{\uparrow}$ packet is generated from the ZetaBasic$^{\uparrow}$ source row, eta half-plane row, AnalyticContinuationSocket$^{\uparrow}$ handoff, pole-residue row at 1, functional-equation factor ledger, Gamma$^{\uparrow}$ boundary interface, trivial-zero ledger, displayed hsame transports, finite Cont routes, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ZetaContinuationSocket}^{\uparrow}}$, seed)

Scope closed: Definition 570.1, Definition 570.2, Definition 570.3, and Definition 570.4 define the local socket carrier, obligation rows, non-escape ledger, and downstream analytic handoff.

Formal verification: FormalStatus(NameCert$_{\text{ZetaContinuationSocket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close RH, nontrivial-zero classification, global meromorphic uniqueness, complete Gamma$^{\uparrow}$ construction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, pole and Gamma ledger exactness, functional-equation route provenance, trivial-zero ledger exactness, and local naming for the same finite socket packet.

Theorem 570.6 (ZetaContinuationSocket witness carrier bridge)**.** *Let* $Z = (B, E, A, P, F, T, G, H, C, N)$ *be an accepted* ZetaContinuationSocket$^{\uparrow}$ *carrier as in Definition 570.1. Suppose* R *is a displayed* Pkg *provenance row over the same zeta source, eta handoff, pole ledger, functional-equation ledger,*

trivial-zero ledger, Gamma-boundary row, transport row, route row, and local naming row, with package scope

$$\mathsf{ZetaBasic}^{\uparrow},\ \mathsf{ZetaCont}^{\uparrow},\ \mathsf{AnaCont}^{\uparrow},\ \mathsf{Holomorphic}^{\uparrow},\ \mathsf{Complex}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{NameCert}.$$

Then

$$W = (B, E, A, P, F, T, G, H, C, R, N)$$

is an accepted $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier as in Definition 580.1. Its downstream non-escape boundary is the same as the boundary of Z: pole requests land in P, negative-even trivial-zero requests land in T, Gamma-dependent requests outside the displayed functional-equation factors land in G, branch requests outside the displayed continuation routes land in A, and RH or non-trivial-zero-location requests have no exported carrier coordinate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationSocketUp.ZetaContinuationSocketWitnessCarrierBridge`

Proof. Unpack the socket carrier. The accepted packet Z already supplies the source row B, eta row E, analytic-continuation socket row A, pole row P, functional-equation factor ledger F, trivial-zero ledger T, Gamma interface G, componentwise transports H, finite continuation routes C, and local naming row N.

The only coordinate required by Definition 580.1 and not already present in the socket tuple is the package-provenance coordinate R. By hypothesis, R is displayed over exactly the same zeta, eta, pole, functional-equation, trivial-zero, Gamma-boundary, transport, route, and naming coordinates, and its package scope is the witness scope listed in Definition 580.1. Inserting this row between C and N therefore gives the witness tuple $W = (B, E, A, P, F, T, G, H, C, R, N)$ with every required coordinate visible.

It remains to compare the exported boundaries. The provenance row R is a package row; it records how the already displayed coordinates are named through Pkg, and it is not an analytic continuation row, a Gamma completion row, a pole-regularization row, or a zero-location row. Thus adding R cannot create a route that bypasses A, P, T, G, H or C.

Now apply the socket non-escape statement of Definition 570.3. The pole boundary is still the coordinate P, the negative-even trivial-zero ledger is still T, Gamma escape is still absorbed by G, and branch escape is still absorbed by A. The witness handoff statement of Definition 580.37 reads these same coordinates through W and exports no RH conclusion and no zero-location classifier. Hence the constructed witness has the same downstream non-escape boundary as the original socket carrier. □

Theorem 570.7 (ZetaContinuationSocket functional-equation boundary)**.** *Let $Z = (B, E, A, P, F, T, G, H, C, N)$ be an accepted* $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *carrier as in Definition 570.1. Any accepted route that reads eta data, pole data,* $\mathsf{Gamma}^{\uparrow}$ *boundary data, branch data, trivial-zero data, transport data, continuation-route data, or local naming data into the witness-facing packet must factor through the displayed functional-equation ledger F together with the visible socket rows $B, E, A, P, T, G, H, C, N$.*

The route exports no RH verdict, no nontrivial-zero classifier, no global meromorphic uniqueness principle, and no completed $\mathsf{Gamma}^{\uparrow}$ *construction. Those endpoints are not coordinates of the carrier and are not added by the witness carrier bridge.*

Proof. Project the carrier from Definition 570.1. The eta row is E, the pole row is P, the functional-equation factor ledger is F, the trivial-zero ledger is T, the $\mathsf{Gamma}^{\uparrow}$ boundary interface is G, and the structural rows are H, C, N.

Apply Definition 570.3. Reads asking for a pole value, a Gamma-pole value, a branch outside the displayed continuation routes, or a nontrivial-zero location are recorded in the corresponding ledger coordinates instead of being exported as continuation values.

Apply Definition 570.4. Every downstream continued-zeta read keeps the source, eta, socket, pole, functional-equation, trivial-zero, Gamma-boundary, transport, continuation, and naming rows visible. Thus the functional-equation boundary is a displayed ledger route, not a hidden analytic theorem.

Finally apply Definition 570.6. Passing to the witness packet only inserts the package-provenance row over the same displayed coordinates. Since RH verdicts, nontrivial-zero classifiers, global meromorphic uniqueness, and completed Gamma constructions are absent from the socket carrier, no witness-facing route can export them. □

Theorem 570.8 (ZetaContinuationSocket boundary exhaustion)**.** *Every accepted* $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *socket read factors through the analytic-continuation socket rows* B, E, A*, the pole row* P*, the functional-equation ledger* F*, the trivial-zero ledger* T*, the* $\mathsf{Gamma}^{\uparrow}$ *boundary row* G*, the witness-carrier bridge, the functional-equation boundary, the visible ledger rows, the displayed* hsame *transport, the* Cont *replay, package provenance, and the local* $\mathsf{NameCert}$ *row before any downstream witness may consume it. The exhausted boundary exports no RH verdict, no global meromorphic uniqueness principle, no nontrivial-zero classification, and no complete* $\mathsf{Gamma}^{\uparrow}$ *construction.*

Proof. Start with the NameCert obligations of Definition 570.2. They bind carrier admission, classifier stability, ledger exactness, provenance, and local naming to the visible rows of the socket packet.

Apply ledger non-escape, Definition 570.3, and analytic handoff, Definition 570.4. These force every consumer read through the source, eta, continuation socket, pole, functional-equation, trivial-zero, Gamma-boundary, transport, replay, and naming rows.

Now apply the witness carrier bridge and the functional-equation boundary, Definition 570.6 and Definition 570.7. Passing to a witness only adds package provenance over the same displayed rows, and the functional-equation route keeps the factor ledger visible.

The listed rows therefore exhaust the socket boundary. Since RH, global meromorphic uniqueness, nontrivial-zero classification, and a complete Gamma construction are absent from the carrier and from the bridge row, no downstream witness can consume them through this socket read. □

Chapter 571

A Concrete Naming Certificate for $\mathsf{DimLiftBoundary}^{\uparrow}$

The $\mathsf{DimLiftBoundary}^{\uparrow}$ packet turns the AxisZeckendorf dimension-lift refusal into a finite boundary certificate. It records the carry rows, normal-address rows, cannot-claim rows, full-axis context, real-facing refusal, transport routes, package provenance, and local naming row that a later dimension-lift acceptance flow must inspect before it may claim any primitive dimension reading.

The boundary packet consumes the refusal surface of chapter 554, the carry row of chapter 544, the finite normal-address surface of chapter 278, and the $\mathsf{Real}^{\uparrow}$ handoff of chapter 38. These sibling chapters supply displayed dependency rows for the boundary packet, not primitive dimension data.

It is also read downstream of the zero-spine AxisNat carrier in chapter 276 and the standard-boundary comparison in chapter 530. Those sibling rows explain why the ZCarry and normal-address evidence remains refusal evidence rather than a primitive dimension row.

Definition 571.1 (DimLiftBoundary carrier)**.** *A* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Z, N, A, F, R, H, C, P, Q)$$

with the following displayed rows:

1. *Z is the* $\mathsf{ZCarry}^{\uparrow}$ *carry-classifier ledger over the relevant AxisZeckendorf histories;*
2. *N is the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-address row for the same finite histories;*
3. *A is the* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *refusal row containing the dimension-lift non-claim;*
4. *F is the* $\mathsf{FullAxis}^{\uparrow}$ *context row through which the carry and normal-address data are read;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *refusal boundary, recording that no real, metric, topological, vector-space, homological, projective, or geometric dimension row is exported;*
6. *H stores componentwise* hsame *transports for the carry row, normal-address row, cannot-claim row, full-axis context, real-facing boundary, and local naming row;*
7. *C stores the finite* Cont *routes by which consumers read the displayed rows from the packet boundary;*
8. *P is the* Pkg *provenance over* $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *Q is the local* $\mathsf{NameCert}_{\mathsf{DimLiftBoundary}^{\uparrow}}$ *row.*

The carrier is a boundary packet, not a dimension object. It exposes the finite rows that explain why a dimension-lift reading is blocked, while leaving any accepted dimension interface to a separate certificate.

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryPacket single carrier alignment`

Definition 571.2 (DimLiftBoundary classifier)**.** *For accepted packets $D = (Z, N, A, F, R, H, C, P, Q)$ and $D' = (Z', N', A', F', R', H', C', P', Q')$, the classifier* $\mathsf{DimLiftBoundaryClassifier}(D, D')$ *holds exactly when the displayed* hsame *and* Cont *rows transport Z to Z', N to N', A to A', F to F', R to R', P to P', and Q to Q' componentwise. The classifier preserves carry and normal-address rows while blocking any read that treats those rows as a primitive dimension.*

Theorem 571.3 (DimLiftBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DimLiftBoundary}^{\uparrow}}$ *surface for Definition 571.1 consists of five finite rows: carrier admission for D; classifier stability for Z, N, A, F, R under the displayed transports H; route exactness for the consumer reads in C; package provenance through P; and local naming exactness for Q. These rows preserve the carry and normal-address evidence while keeping primitive dimension readings outside the certificate.*

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryCarrier namecert obligations`

Proof. Carrier admission is projection of the finite tuple D. Each dependency row named in the statement is a visible coordinate of the same BHist packet.

Classifier stability is componentwise. The hsame table H and the routes C transport the carry ledger, normal-address row, cannot-claim row, full-axis context, and real-facing refusal boundary without adding a dimension coordinate.

The package and naming rows then repack exactly those coordinates. Since the carrier has no row for geometric, topological, vector-space, homological, projective, metric, or real dimension data, the NameCert surface is exhausted by the boundary packet. □

Theorem 571.4 (DimLiftBoundary non-escape)**.** *Every consumer read exported by an accepted* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier factors through the displayed rows $Z, N, A, F, R, H, C, P, Q$. A read that asks to interpret the carry classifier or normal-address row as a primitive dimension lands in the refusal boundary A and R, not in a dimension-valued row.*

Proof. Read the consumer request through the finite routes in C. If it asks for carry evidence, normal-address data, the cannot-claim row, the full-axis context, transport, provenance, or local naming, the corresponding coordinate of D supplies that read.

If the request asks for a primitive dimension interpretation, no coordinate of D has that type. The only visible rows governing the request are the cannot-claim row A and the real-facing refusal boundary R.

Thus the request is recorded as boundary evidence. It cannot escape as a geometric, topological, vector-space, homological, projective, metric, or real dimension object. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryNonEscape`

Theorem 571.5 (DimLiftBoundary public export)**.** *The carrier of Definition 571.1, the classifier of Definition 571.2, and the local obligations of Definition 571.3 export a public boundary packet for later* $\mathsf{DimLift}^{\uparrow}$ *acceptance-flow work. Downstream consumers may read the displayed* $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, but they receive no primitive dimension interface from this chapter.*

Proof. The carrier supplies the public rows $Z, N, A, F, R, H, C, P, Q$. The classifier preserves those rows componentwise, so a downstream consumer can transport the boundary packet without changing its visible dependency surface.

The NameCert obligations identify the exact admission, transport, route, provenance, and naming rows. The non-escape theorem then blocks every request that would turn the carry or normal-address evidence into a primitive dimension object.

Therefore the public export is a boundary carrier for future acceptance-flow chapters. It supplies the refusal surface that such chapters must consume before supplying any separate dimension certificate. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryPublicExport`

Theorem 571.6 (DimLiftBoundary axis refusal obligations)**.** *For an accepted* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier* $D = (Z, N, A, F, R, H, C, P, Q)$*, the AxisZeckendorf-facing refusal surface consists exactly of the* $\mathsf{ZCarry}^{\uparrow}$ *carry rows in* Z*, the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-address rows in* N*, the* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *cannot-claim rows in* A*, the* $\mathsf{FullAxis}^{\uparrow}$ *context row in* F*, and the transport, route, provenance, and local naming rows* H, C, P, Q*. The surface refuses any export that reads* Z *or* N *as a primitive geometric, topological, vector-space, homological, projective, metric, or real dimension.*

Proof. Project the carrier and read the AxisZeckendorf coordinates Z, N, A, F. The carry and normal-address rows are visible only inside the full-axis context, and the cannot-claim row records the refusal attached to that context.

The remaining rows H, C, P, Q transport, route, package, and name this same boundary. None of them supplies a dimension-valued coordinate or a classifier into one.

Therefore every axis-facing consumer receives the carry and normal-address evidence as refusal evidence. It cannot repack the same rows as a primitive dimension interface. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryAxisRefusalObligations`

Theorem 571.7 (DimLiftBoundary Real handoff non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier factors through the displayed refusal boundary* R *together with* Z, N, A, F, H, C, P, Q*. The handoff may report that the AxisZeckendorf carry and normal-address rows have no real-dimension export, but it may not produce a real coordinate, real metric dimension, host-real equality row, or completion-facing dimension object.*

Proof. Start with the real-facing coordinate R. By definition of the carrier, R is reached only after the carry rows, normal-address rows, cannot-claim rows, and full-axis context have already been displayed.

The continuation routes in C and transports in H replay the same refusal packet for a real-facing consumer. Package provenance P and local naming Q record the dependency surface without adding any real-valued coordinate.

Thus the real handoff is a non-escape boundary. It can expose refusal of a real dimension reading, but it cannot export a real dimension object or any host-real comparison datum. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryRealHandoffNonescape`

Theorem 571.8 (DimLiftBoundary Zeckendorf carrier scope)**.** *The Zeckendorf scope of an accepted* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier is the finite packet* Z, N, A, F, H, C, P, Q *together with the real-facing refusal row* R*. It is closed under the displayed* hsame *transports and* Cont *routes for AxisZeckendorf carry rows, normal-address rows, cannot-claim rows, full-axis context, RealUp refusal, package provenance, and NameCert rows, and it contains no primitive geometric, topological, vector-space, homological, projective, metric, or real dimension carrier.*

Proof. Read the finite packet componentwise. The Zeckendorf data are exactly Z, N, A, F, and the only consumer-visible extensions are the transport, route, provenance, local naming, and real-facing refusal rows.

Closure under the displayed transports follows because H names the componentwise hsame rows for the same coordinates. Closure under consumer access follows because C lists the only continuation routes out of the packet.

Since no coordinate of D is a primitive dimension carrier, the Zeckendorf scope remains a carrier for refusal evidence. The scope ends at the listed boundary rows. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryZeckendorfCarrierScope`

Theorem 571.9 (DimLiftBoundary route-exactness refusal)**.** *For an accepted* DimLiftBoundary$^{\uparrow}$ *carrier $D = (Z, N, A, F, R, H, C, P, Q)$, every consumer request that tries to read Z or N as a primitive dimension must pass through the cannot-claim row A and the real-facing refusal row R. The route table C and transport table H may expose carry evidence, normal-address evidence, full-axis context, provenance, and local naming, but they do not expose a dimension-valued row.*

Thus route exactness for primitive-dimension requests is a refusal route. The exact target of such a request is the displayed pair A, R, not a geometric, topological, vector-space, homological, projective, metric, or real dimension object.

Proof. Read the request through the continuation table C. If it asks for carry, normal-address, full-axis, transport, package, or naming evidence, the corresponding coordinate of the carrier supplies that displayed row.

If the same request asks for a primitive dimension interpretation of Z or N, no carrier coordinate has that type. The only displayed rows governing that request are the cannot-claim row A and the real-facing refusal boundary R.

Transport through H preserves this boundary componentwise. It can move the refusal packet along accepted presentations, but it cannot add a new dimension row to the carrier. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryCarrier cannot claim route exactness`

Theorem 571.10 (DimLiftBoundary axis normal-address lock)**.** *For an accepted* DimLiftBoundary$^{\uparrow}$ *carrier $D = (Z, N, A, F, R, H, C, P, Q)$, the rows Z and N are locked to AxisZeckendorf carry and normal-address evidence only after four sibling routes are displayed: the* Real$^{\uparrow}$ *handoff row R of Definition 571.7, the* ZeckendorfNormal$^{\uparrow}$ *scope row of Definition 571.8, the* ZCarry$^{\uparrow}$ *carry row inside the full-axis context F, and the* AxisZeckendorfCannotClaim$^{\uparrow}$ *refusal route of Definition 571.9. Any accepted normal-address read must preserve this route through F, H, C, P, Q and cannot reinterpret Z or N as a primitive dimension carrier.*

The lock is local to the displayed packet. It records a boundary address, not a full dimensional lift, host equality of addresses, RealUp completeness, or completeness of the axis.

Proof. Project the carrier rows. The carry coordinate Z and normal-address coordinate N occur together with the full-axis context F, so each read of either row is already addressed inside the AxisZeckendorf packet rather than inside an ambient dimension object.

Use the Zeckendorf carrier scope of Definition 571.8. It closes the displayed packet under the hsame transports and Cont routes while keeping primitive dimension carriers outside the scope.

Use the Real handoff non-escape theorem Definition 571.7. It places the row R downstream of the carry, normal-address, cannot-claim, and full-axis rows, so an accepted normal-address read cannot bypass the real-facing refusal boundary.

Finally apply the cannot-claim route exactness of Definition 571.9. The request may expose Z, N, F, H, C, P, Q, but a primitive-dimension request is routed to A and R.

The package row P and local naming row Q therefore record the same dependency boundary. They name the axis-normal evidence already present; they do not provide a separate dimension row, host address equality, or completeness principle. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryAxisNormalAddressLock`

Theorem 571.11 (DimLiftBoundary downstream acceptance boundary)**.** *Any later* $\mathsf{DimLift}^{\uparrow}$ *acceptance certificate that exports a dimension row must first consume the displayed* $\mathsf{DimLiftBoundary}^{\uparrow}$ *refusal packet* $Z, N, A, F, R, H, C, P, Q$*. In particular, the later certificate must account for the cannot-claim row* A*, the real-facing refusal row* R*, the axis-normal lock on* Z, N*, and the transport, route, provenance, and naming boundary* H, C, P, Q *before it exports any separate dimension evidence.*

Consequently $\mathsf{DimLiftBoundary}^{\uparrow}$ *is a required refusal boundary for downstream acceptance work, not an acceptance certificate and not a primitive dimension source.*

Proof. Begin with Definition 571.9. It forces primitive-dimension requests against the boundary carrier through the refusal rows A and R.

Next use Definition 571.10. It fixes Z and N as AxisZeckendorf carry and normal-address evidence under F, H, C, P, Q rather than as dimension-valued rows.

A later acceptance certificate may add a separate dimension row only after this packet has been consumed as refusal evidence. Without that consumption, the later certificate would be reading through a row that the present carrier explicitly does not export. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryDownstreamAcceptanceBoundary`

Theorem 571.12 (DimLiftBoundary carrier admission obligation)**.** *For every displayed packet*

$$D = (Z, N, A, F, R, H, C, P, Q)$$

accepted as a $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier, carrier admission is exactly the admission of the nine rows* $Z, N, A, F, R, H, C, P, Q$ *in Definition 571.1. The row* Z *admits AxisZeckendorf carry evidence,* N *admits normal-address evidence,* A *admits the cannot-claim refusal,* F *admits the full-axis context,* R *admits the real-facing refusal boundary,* H *admits componentwise transport,* C *admits consumer routes,* P *admits package provenance, and* Q *admits the local NameCert name. No admitted row is a primitive dimension carrier.*

Proof. Project the carrier definition. The displayed tuple has exactly the rows $Z, N, A, F, R, H, C, P, Q$, and each row is named in the carrier inventory.

Apply Definition 571.3. Its carrier-admission clause is the first local NameCert row and it ranges over the same displayed packet.

Finally apply Definition 571.4. Since primitive dimension reads land in A and R rather than in a dimension-valued coordinate, carrier admission does not add any primitive dimension carrier. □

Lean4 ✓ `BEDC.Derived.DimLiftBoundaryUp.DimLiftBoundaryCarrier admission obligation`

Theorem 571.13 (DimLiftBoundary classifier stability obligation)**.** *For accepted packets* $D = (Z, N, A, F, R, H, C, P, Q)$ *and* $D' = (Z', N', A', F', R', H', C', P', Q')$*, the* $\mathsf{DimLiftBoundary}^{\uparrow}$ *classifier is stable exactly when the displayed* hsame *and* Cont *rows transport the carry, normal-address, cannot-claim, full-axis, real-facing refusal, provenance, and local naming coordinates componentwise. This stability preserves the refusal interpretation of* Z, N, A, F, R *and does not create a geometric, topological, vector-space, homological, projective, metric, or real dimension row.*

Proof. Start with Definition 571.2. It defines classifier comparison by componentwise hsame and Cont transport across Z, N, A, F, R, P, Q and their primed counterparts.

The transport table H and route table C are visible coordinates of both accepted packets. Thus classifier stability is checked inside the same finite carrier surface, not by an external dimension semantics.

Use Definition 571.8 and Definition 571.10. They keep Z and N locked to AxisZeckendorf carry and normal-address evidence under F, H, C, P, Q, so stable transport preserves the refusal interpretation. □

Theorem 571.14 (DimLiftBoundary ledger obligation)**.** *For every accepted* $\mathsf{DimLiftBoundary}^{\uparrow}$ *carrier* $D = (Z, N, A, F, R, H, C, P, Q)$*, the ledger obligations are route exactness through* C*, package provenance through* P*, local naming exactness through* Q*, and non-escape through* A *and* R*. Any consumer route that asks for carry, normal-address, full-axis, transport, provenance, or naming data receives the corresponding displayed row; any route that asks for primitive dimension data is refused by the cannot-claim and real-facing boundary rows.*

Proof. Route exactness is supplied by the continuation table C. By Definition 571.9, a request for primitive dimension data is routed to A and R, while ordinary carrier requests remain on the displayed rows.

Package provenance and local naming are the rows P and Q of Definition 571.1. The NameCert obligations of Definition 571.3 identify these rows as part of the same finite boundary packet.

Non-escape is then Definition 571.4. Together with the public boundary statement of Definition 571.5, it ensures that the ledger exports refusal evidence for downstream acceptance work and no primitive dimension interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DimLiftBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DimLiftBoundary}^{\uparrow}$ packet is generated from $\mathsf{ZCarry}^{\uparrow}$ carry rows, $\mathsf{ZeckendorfNormal}^{\uparrow}$ normal-address rows, the $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ refusal row, $\mathsf{FullAxis}^{\uparrow}$ context, $\mathsf{Real}^{\uparrow}$ refusal boundary, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DimLiftBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 571.1, Definition 571.2, Definition 571.3, Definition 571.4, and Definition 571.5 bind the finite boundary carrier, classifier, obligation rows, non-escape rule, and public export for AxisZeckendorf dimension-lift refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DimLiftBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a primitive geometric dimension, topological dimension, vector-space dimension, homological dimension, projective dimension, metric dimension, real dimension, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, route exactness, package provenance, local naming exactness, and non-escape for the same finite boundary packet.

Chapter 572

A Concrete Naming Certificate for $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$

The $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ interface records the finite BEDC packet by which the standard-boundary comparison between nonempty unary histories and nonempty zero-spine histories is made inspectable. It refines the displayed boundary row of $\mathsf{UnaryDirectionBridge}^{\uparrow}$ into a local NameCert target while preserving the kernel distinction between the $\mathsf{Nat}^{\uparrow}$ unary carrier and the $\mathsf{AxisNat}^{\uparrow}$ zero-spine carrier.

The dependency anchors are $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. The $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ packet consumes the source-boundary row of chapter 9344 before exposing its forward and backward standard-boundary routes; it exposes a standard-interface handoff, but it does not identify unary and zero-spine histories inside the finite kernel.

Definition 572.1 (Unary-zero-spine standard-isomorphism carrier)**.** *A* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (U, A, L, F, R, H, C, P, N)$$

with the following visible rows:

1. *U is the nonempty* $\mathsf{Nat}^{\uparrow}$ *unary-spine row, read as a displayed* Unary *history with at least one* E1 *step;*
2. *A is the nonempty* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine row, read as a displayed* ZeroSpine *history with at least one* E0 *step;*
3. *L is the length-match row stating that U and A have the same finite successor count at the standard boundary;*
4. *F is the forward finite continuation route from the unary row through L to the corresponding zero-spine row;*
5. *R is the backward finite continuation route from the zero-spine row through L to the corresponding unary row;*
6. *H stores componentwise* hsame *transports for U, A, L, F, R and the local naming row;*
7. *C stores displayed* Cont *routes for unary inspection, zero-spine inspection, length comparison, forward readback, backward readback, additive-interface handoff, and NameCert audit;*

8. *P is the* Pkg *provenance row naming* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}}$ *row.*

The displayed length-match row is

$$\begin{aligned} \mathsf{len}_{\mathsf{Unary}}(U) &= \mathsf{len}_{\mathsf{ZeroSpine}}(A), \\ \mathsf{len}_{\mathsf{Unary}}(U) &> 0, \\ \mathsf{len}_{\mathsf{ZeroSpine}}(A) &> 0. \end{aligned}$$

The classifier accepts only packets whose forward and backward routes are both finite continuations over the displayed rows and whose additive-interface handoff is read through C after L, H, P, N are present. It has no coordinate for host equality of histories, quotienting the two carriers, or replacing the displayed routes by an ambient standard-model function.

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoTasteGate single carrier alignment`

Theorem 572.2 (Unary-zero-spine standard-isomorphism NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryZeroSpine}}$ *surface for Definition 572.1 consists of five BEDC-native rows:*

1. *carrier exposure for the finite packet $Z = (U, A, L, F, R, H, C, P, N)$;*

2. *classifier stability under componentwise* hsame *transport of the unary row, zero-spine row, length-match row, forward route, backward route, continuation ledger, package provenance, and local naming row;*

3. *route-ledger exactness stating that every accepted forward or backward comparison is one of the displayed finite routes F or R through C and L;*

4. *standard-interface handoff stating that* $\mathsf{Add}^{\uparrow}/\mathsf{AxisAdd}^{\uparrow}$ *consumers may read the comparison only through the transported length-match and route rows;*

5. *nonidentification boundary stating that the packet does not turn nonempty unary and zero-spine histories into* hsame*-identical kernel histories.*

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoCarrier namecert obligations`

Proof. Carrier exposure is projection from the finite packet in Definition 572.1. The source rows, length-match row, forward and backward routes, transports, continuations, package provenance, and local naming row are all displayed before the classifier accepts the packet.

Classifier stability is componentwise. Transport through H preserves the two source rows, the length-match row, and both finite routes, while C, P, N repack the same public read surface.

Route-ledger exactness follows because F and R are the only comparison routes in the carrier. Any standard-interface handoff to $\mathsf{Add}^{\uparrow}$ or $\mathsf{AxisAdd}^{\uparrow}$ must pass through C after the length and route rows have been read.

The nonidentification row is inherited from the kernel-distinct bridge discipline of $\mathsf{UnaryDirectionBridge}^{\uparrow}$. Since the carrier contains no equality coordinate identifying U and A as histories, the NameCert surface exposes a standard-boundary comparison rather than a kernel identification. □

Theorem 572.3 (Unary-zero-spine standard-isomorphism nonidentification boundary)**.** *For any accepted* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ *packet* $Z = (U, A, L, F, R, H, C, P, N)$*, the displayed standard-boundary comparison preserves the existing kernel distinction between the nonempty* Unary *row* U *and the nonempty* ZeroSpine *row* A*. In particular, transporting the packet through* H *and reading the forward or backward routes through* C *does not produce an* hsame *row identifying the underlying unary and zero-spine histories as the same kernel history.*

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoCarrier nonidentification boundary`

Proof. Read the packet through its displayed coordinates. The length-match row L relates only the finite successor counts of U and A at the standard boundary; it is not a history equality row.

The forward route F and backward route R are continuation routes over the displayed rows. They witness public readback in opposite directions, but neither route contains a coordinate whose target is an hsame identification of the two underlying histories.

Transport through H is componentwise over U, A, L, F, R, N. It can rename each displayed coordinate inside its own classifier, but it cannot merge the unary and zero-spine carriers because no such merge row is present in the packet.

Thus the standard-boundary comparison remains compatible with the kernel-distinct bridge discipline. The packet makes the standard-interface handoff inspectable while leaving the unary and zero-spine histories nonidentified inside BHist. □

Theorem 572.4 (Unary-zero-spine standard-isomorphism route totality)**.** *For every accepted* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ *packet* $Z = (U, A, L, F, R, H, C, P, N)$*, every accepted forward or backward comparison route is one of the displayed finite routes* F *or* R *through the length-match row* L*, the componentwise transport row* H*, the continuation ledger* C*, the package row* P*, and the local naming row* N*. No comparison route can be read from a host equality of histories, a quotient identifying* $\mathsf{Nat}^{\uparrow}$ *with* $\mathsf{AxisNat}^{\uparrow}$*, or an ambient standard-model function outside the packet.*

Proof. Project the carrier of Definition 572.1. Its only route coordinates are the forward route F and the backward route R, both listed before any standard-interface consumer can inspect the packet.

Apply the route-ledger exactness row in Definition 572.2. It forces every accepted comparison to pass through L, H, C, P, N together with exactly one of the displayed routes.

The nonidentification boundary of Definition 572.3 excludes a hidden kernel identification. Therefore a downstream route has no source other than F or R. □

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoRoute unary closure`

Theorem 572.5 (Unary-zero-spine standard-isomorphism axis-nat handoff)**.** *Every* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$*, or* $\mathsf{AxisAdd}^{\uparrow}$ *consumer of an accepted* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ *packet reads only the nonempty source rows* U, A*, the length-match row* L*, the displayed finite routes* F, R*, the componentwise* hsame *transports, the* Cont *routes,* Pkg *provenance, and local* NameCert *row. Such a consumer receives a standard-boundary handoff, not a kernel identification of unary and zero-spine histories.*

Proof. Begin with the standard-interface handoff clause in Definition 572.2. It allows additive-interface consumers to read the comparison only after the length-match and route rows are present.

Use route totality from Definition 572.4. Any forward or backward read available to $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, or $\mathsf{AxisAdd}^{\uparrow}$ is therefore one of the displayed finite routes through L, H, C, P, N.

Finally apply Definition 572.3. Since no hsame row identifies the underlying histories, the consumer handoff is a standard-boundary route and not a kernel equality. □

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoAxisNatHandoff`

Theorem 572.6 (Unary-zero-spine standard-isomorphism downstream unblock)**.** *The NameCert obligations, route totality, axis-nat handoff, and nonidentification boundary for* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ *form a finite downstream standard-boundary package. A consumer may cite the package to inspect the nonempty* $\mathsf{Nat}^{\uparrow}$ *row, the nonempty* $\mathsf{AxisNat}^{\uparrow}$ *row, the length-match row, the displayed forward and backward routes,* BHist, hsame, Cont, Pkg, *and* NameCert *support, while receiving no host equality of histories, no quotient between the two carriers, and no ambient standard-model isomorphism row.*

Lean4 ✓ `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoDownstreamUnblock`

Proof. First assemble the local obligation surface from Definition 572.2. It supplies carrier exposure, classifier stability, route-ledger exactness, standard-interface handoff, and the nonidentification row.

Next add route totality from Definition 572.4. This makes F and R the only comparison routes available to the downstream consumer.

Then use the axis-nat handoff theorem above. It records that $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, and $\mathsf{AxisAdd}^{\uparrow}$ consumers read the length and route rows rather than a kernel identification.

The existing nonidentification boundary closes the package by excluding host equality, quotienting of the two carriers, and ambient standard-model replacement. The downstream consumer therefore receives exactly the displayed finite standard-boundary route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}}$.

Constructive story: A $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$ packet is generated by nonempty unary and zero-spine source rows, a displayed length-match row, forward and backward finite continuation routes, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 572.1, Definition 572.2, Definition 572.4, Definition 572.3, Definition 572.5, and Definition 572.6 bind the carrier, obligation, route, axis-nat handoff, boundary, and downstream unblock surface for the unary-zero-spine standard-boundary packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.UnaryZeroSpineStandardIsoUp.UnaryZeroSpineStandardIsoCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close a full standard-model isomorphism theorem or identify $\mathsf{Nat}^{\uparrow}$ with $\mathsf{AxisNat}^{\uparrow}$ inside the kernel.

Upgrade path: Public closure requires a public consumer surface beyond the finite standard-boundary package displayed here.

Chapter 573

A Concrete Naming Certificate for $\mathsf{RiemannZeroCountLedger}^{\uparrow}$

The $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ interface records the finite height-window packet needed by the zero-count witness direction of Definition 9239.15. It packages a requested imaginary-height bound, the visible critical-strip zero packets below that height, a unary count row, a Riemann-von Mangoldt comparison ledger, an explicit error row, transport and continuation routes, package provenance, and local naming data. The packet is a counting witness surface only: it does not prove RH, does not assert zero-set completeness, and does not supply analytic continuation.

The ledger consumes its zero-side inputs through existing concrete-instance surfaces. The critical-line route is supplied by chapter 548, while the zero packet source is supplied by chapter 73; the count ledger therefore records a dependency route already visible in its carrier rows rather than a standalone RH-side object.

Definition 573.1 (RiemannZeroCountLedger carrier)**.** *A* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, Z, U, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Nat}^{\uparrow}$ *height request row naming the imaginary-height window $|\Im(s)| \leq T$;*
2. *Z is the finite packet list of* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *rows whose zero and critical-strip coordinates are displayed inside the requested height window;*
3. *U is the* $\mathsf{UnaryHistory}$ *count row assigned to the visible list Z;*
4. *M is the Riemann-von Mangoldt comparison ledger, recording the main-term row, logarithmic row, normalization row, and comparison route used for the same height request;*
5. *E is the explicit error row bounding the gap between the unary count U and the comparison ledger M;*
6. *H stores componentwise* hsame *transports for the height request, zero packet list, unary count row, comparison ledger, error row, package provenance, and local naming row;*
7. *C stores the displayed* Cont *routes for reading the height request, filtering the finite zero packet list, counting the list, comparing against M, exposing the error row, and handing the result to an RH-facing consumer;*

8. *P is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, UnaryHistory, $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannZeroCountLedger}^{\uparrow}}$ *row.*

The carrier exposes only a finite height-window count ledger. It contains no completed zero-set quotient, no proof that the listed packets exhaust all analytic zeros below the height, no analytic-continuation constructor, and no proof of RH*.*

Theorem 573.2 (RiemannZeroCountLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RiemannZeroCountLedger}^{\uparrow}}$ *surface for the carrier of Definition 573.1 consists of six finite obligations:*

1. *carrier admission: the accepted object is the displayed* BHist *packet* $L = (T, Z, U, M, E, H, C, P, N)$*;*

2. *height-window binding: the height request row T is the only row used to filter the critical-strip zero packets in Z;*

3. *zero-list visibility: every counted entry is a displayed* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet with its* $\mathsf{ZetaZeros}^{\uparrow}$ *and* $\mathsf{CritStrip}^{\uparrow}$ *coordinates still visible;*

4. *unary-count exactness: the count row U is the* UnaryHistory *count of the displayed finite list Z and not a cardinality of an ambient zero set;*

5. *comparison-error ledger: the Riemann-von Mangoldt comparison row M and explicit error row E are both read before the count is exported;*

6. *non-escape: every downstream zero-count read factors through $T, Z, U, M, E, H, C, P, N$ and has no route to zero-set completeness, analytic continuation, host cardinality, or a proof of* RH*.*

Lean4 ✓ `BEDC.Derived.RiemannZeroCountLedgerUp.RiemannZeroCountLedgerCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 573.1. The height request, zero packet list, unary count, comparison ledger, error row, transports, continuations, provenance, and local naming row are explicit coordinates.

Height-window binding and zero-list visibility are componentwise reads. The route first reads T, then filters the visible $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ rows in Z, while each listed zero keeps its $\mathsf{ZetaZeros}^{\uparrow}$ and $\mathsf{CritStrip}^{\uparrow}$ coordinates visible through the stored transport and continuation rows.

The count and comparison obligations are the rows U, M, E. The count is the unary count of the displayed list, and the comparison is only the finite Riemann-von Mangoldt ledger with its explicit error row.

Since the carrier has no coordinate for an ambient zero set, analytic continuation, host cardinality, zero-set completeness, or an RH proof, every downstream read is exhausted by the displayed rows. □

Theorem 573.3 (RiemannZeroCountLedger window exhaustion)**.** *Every RH-facing zero-count read exported by an accepted* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *carrier can inspect only*

$$T, Z, U, M, E, H, C, P, N.$$

It may read the requested height, the displayed finite list of critical-strip zero packets, the unary count, the Riemann-von Mangoldt comparison ledger, the explicit error row, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local* NameCert *row. It cannot infer that the finite list contains all analytic zeros below the height, cannot construct analytic continuation, cannot read a host zero-set cardinality, and cannot close* RH*.*

Lean4 ✓ `BEDC.Derived.RiemannZeroCountLedgerUp.RiemannZeroCountLedgerCarrier window exhaustion`

Proof. Start with an RH-facing zero-count request. By the carrier definition, the only height input is T, and the only zero data available at that height is the finite packet list Z.

Apply the zero-list and unary-count obligations of Definition 573.2. They force the count read to pass through displayed $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ packets and then through the unary count row U.

Next read the comparison surface. The rows M and E record the Riemann-von Mangoldt comparison and the explicit constructive error before the result is exported.

The remaining rows H, C, P, N provide transport, replay, provenance, and local naming only. Therefore the height-window surface is exhausted by the displayed packet and has no route to zero-set completeness, analytic continuation, host cardinality, or a proof of the RH predicate. □

Theorem 573.4 (RiemannZeroCountLedger height-filter stability)**.** *For every accepted* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *carrier* $L = (T, Z, U, M, E, H, C, P, N)$*, a transported zero-count read preserves the displayed height request row* T *before the finite zero packet list* Z *is filtered. The componentwise* hsame *row* H *and displayed* Cont *replay row* C *may transport and replay the height request, but they cannot replace* T *by a new height window, a host real bound, or an untracked zero-set predicate.*

Proof. Project the carrier of Definition 573.1. The height request is the coordinate T, and the zero packet list Z is filtered only after that coordinate is read.

Apply the height-window binding obligation of Definition 573.2. It makes T the unique displayed row used by the finite filter, so a downstream read cannot introduce a second height source.

Now read the transport and replay rows. The componentwise hsame entries in H preserve the height row as the same displayed coordinate, while the Cont route in C replays the read of T before the filter on Z.

Therefore the transported packet still filters Z under the original displayed height row. Since no carrier coordinate stores a host real bound or ambient zero-set predicate, those endpoints cannot be inferred by transport. □

Theorem 573.5 (RiemannZeroCountLedger comparison-error nonescape)**.** *Every RH-facing export of an accepted* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *carrier must read the Riemann-von Mangoldt comparison ledger* M *and explicit error row* E *before the unary count row* U *is handed to a consumer. The export is only a finite comparison-error ledger for the displayed height window; it cannot imply* RH*, analytic continuation, zero-set completeness, host cardinality, or completeness of the analytic zero list.*

Lean4 ✓ `BEDC.Derived.RiemannZeroCountLedgerUp.RiemannZeroCountLedgerCarrier comparison error nonescape`

Proof. Start with the exported zero-count read of Definition 573.3. That theorem exhausts the read by the displayed rows $T, Z, U, M, E, H, C, P, N$.

Use the comparison-error ledger obligation in Definition 573.2. It requires both M and E to be opened before the count is exported, so the consumer receives a comparison row together with its explicit error boundary.

The rows M and E compare the unary count against the finite Riemann-von Mangoldt ledger for the same displayed height request. They do not add a coordinate for analytic continuation, all zeros below the height, host cardinality, or a proof of the RH predicate.

Thus the export is non-escaping: it records the comparison-error surface for the finite packet and refuses each analytic endpoint named in the statement. □

Theorem 573.6 (RiemannZeroCountLedger zero-list count transport)**.** *Let* $L = (T, Z, U, M, E, H, C, P, N)$ *and* $L' = (T', Z', U', M', E', H', C', P', N')$ *be accepted* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *carriers connected by the componentwise* hsame *transport and displayed* Cont *replay rows of their local packets. If the transported read identifies the height request, zero-list row, and unary-count row with the displayed* T, Z, U *rows of* L*, then the transported packet preserves the same finite zero list* Z *and the same unary count* U *before any comparison-error export is consumed.*

Proof. Project both accepted carriers. In each packet, the zero-list row and unary-count row are displayed coordinates, not ambient zero-set or host-cardinality data.

Apply zero-list visibility and unary-count exactness from Definition 573.2. These obligations force the count to be the UnaryHistory count of the displayed finite list and force every counted entry to remain a visible $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ packet.

Now use the hypothesis on the componentwise transport and replay rows. The hsame entries identify the transported height, zero-list, and count rows with T, Z, U, and the Cont entries replay those reads before the comparison rows are consumed.

Consequently the transported export preserves the displayed finite list and its unary count. Any attempt to read a different finite list, host cardinality, zero-set completeness theorem, or analytic continuation would bypass the named coordinates and would no longer be a transported read of the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannZeroCountLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ packet is generated from a $\mathsf{Nat}^{\uparrow}$ height request, a finite list of $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ packets, a UnaryHistory count row, a Riemann-von Mangoldt comparison ledger, an explicit error row, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannZeroCountLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite height-window zero-count carrier, local naming obligations, unary-count exactness, comparison-error ledger, and RH-facing readback boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannZeroCountLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove RH, construct analytic continuation, enumerate all analytic zeros, close zero-set completeness, assert host cardinality, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, height-window binding, zero-list visibility, unary-count exactness, comparison-error ledger exactness, and non-escape for the displayed zero-count packet.

Chapter 574

A Concrete Naming Certificate for $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$

The $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$ interface names the finite boundary between two adjacent certificate compiler stages. It keeps both compiler packets visible, records the shared middle-stage compatibility row, and exposes only the classifier, continuation, package, and local naming rows needed by downstream compiler-pipeline consumers.

The boundary is a consumer of chapter 473 because the adjacent packets K_{SM} and K_{MT} are accepted $\mathsf{CertificateCompiler}^{\uparrow}$ routes. This chapter adds only the displayed middle-row compatibility ledger J between those two compiler packets.

Definition 574.1 (CertificatePipelineStageBoundary carrier)**.** *A* $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, T, K_{SM}, K_{MT}, J, H, C, P, N)$$

with the following visible rows:

1. *S is the source* NameCert *row for the first compiler stage;*
2. *M is the middle* NameCert *row that receives the first stage and supplies the second stage;*
3. *T is the target* NameCert *row for the second compiler stage;*
4. *K_{SM} is an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet from S to M;*
5. *K_{MT} is an accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet from M to T;*
6. *J is the middle-stage compatibility row asserting that the target row read by K_{SM} is the source row read by K_{MT};*
7. *H records componentwise* hsame *transports for the source, middle, target, adjacent compiler packets, compatibility row, package row, and local naming row;*
8. *C records the displayed* Cont *routes for the first compiler stage, the middle-stage handoff, the second compiler stage, and the composed boundary read;*
9. *P names* BHist, hsame, Cont, Pkg, NameCert, $\mathsf{CertificateCompiler}^{\uparrow}$, *and the adjacent compiler-stage boundary shape;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificatePipelineStageBoundary}^{\uparrow}}$ *row.*

The packet contains no host compiler semantics, program equivalence theorem, global optimisation proof, or unrecorded intermediate-representation interpretation.

Theorem 574.2 (CertificatePipelineStageBoundary NameCert obligations)**.** *For the carrier of Definition 574.1, the local* $\mathsf{NameCert}_{\mathsf{CertificatePipelineStageBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet* $B = (S, M, T, K_{SM}, K_{MT}, J, H, C, P, N)$*;*
2. *middle landing and source compatibility, requiring the target certificate row consumed by* K_{SM} *to be the same displayed middle row* M *used as the source certificate row of* K_{MT}*;*
3. *classifier transport across the boundary, first through* K_{SM} *and then through* K_{MT}*, with no classifier row outside* S, M, T*;*
4. *continuation composition, composing the first-stage route, middle-stage handoff, and second-stage route through the displayed* C *row;*
5. *non-escape from host compiler semantics, excluding program equality, semantic models, global optimisation correctness, and unrecorded target behavior.*

Proof. Carrier admission is projection of Definition 574.1. The rows $S, M, T, K_{SM}, K_{MT}, J, H, C, P, N$ are visible coordinates of one finite BHist packet.

The middle compatibility obligation is the row J. It states that the first accepted $\mathsf{CertificateCompiler}^{\uparrow}$ packet lands in the same middle naming certificate that the second accepted compiler packet reads as its source.

Classifier transport is read stage by stage. The first packet transports source classifier rows from S to M, the second transports those middle rows from M to T, and the componentwise transports in H preserve the same displayed boundary rows.

Continuation composition is the corresponding route in C. It concatenates the first-stage continuation route, the middle-stage handoff route, and the second-stage continuation route before the package row P and local naming row N expose the boundary to consumers.

No further coordinate remains in the carrier. Hence host compiler semantics, program equality, global optimisation correctness, and unrecorded target behavior cannot witness the adjacent-stage boundary. □

Theorem 574.3 (CertificatePipelineStageBoundary middle-row determinacy)**.** *Let*

$$B = (S, M, T, K_{SM}, K_{MT}, J, H, C, P, N)$$

and

$$B' = (S', M', T', K_{SM}, K_{MT}, J', H', C', P', N')$$

be accepted $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$ *carriers in the sense of Definition 574.1 and Definition 574.2. Assume that both displayed compatibility rows identify the target* $\mathsf{NameCert}$ *row read by the shared first compiler packet* K_{SM} *with the source* $\mathsf{NameCert}$ *row read by the shared second compiler packet* K_{MT}*. Then the displayed middle rows* M *and* M' *are classifier-same as the common middle* $\mathsf{NameCert}$ *row of the adjacent compiler pair, and every composed boundary read exported by either* B *or* B' *factors through that common middle row before it reaches the target row.*

Proof. Project the shared first compiler packet as

$$K_{SM} = (S_0, M_0, G_{SM}, E_{SM}, C_{SM}, H_{SM}, P_{SM}, N_{SM}).$$

By Definition 473.3, every accepted read of K_{SM} exposes its endpoint NameCert rows before any landing or classifier comparison is consumed. Since the same packet K_{SM} occurs in both boundary carriers, the target endpoint available to both boundary reads is the single displayed row M_0.

Project the shared second compiler packet as

$$K_{MT} = (M_1, T_0, G_{MT}, E_{MT}, C_{MT}, H_{MT}, P_{MT}, N_{MT}).$$

The same endpoint-availability result says that the source endpoint available to every accepted read of K_{MT} is the single displayed row M_1.

Now read the two compatibility rows. In B, the row J asserts that the target row read from K_{SM} is the source row read from K_{MT}. Hence J identifies M_0 and M_1 as the displayed middle certificate used by the boundary carrier, so M is classified with that common middle row. The same argument with J' shows that M' is classified with the same common middle row. By transitivity of the local NameCert classifier comparison, M and M' are classifier-same.

It remains to identify the route used by a composed boundary read. The boundary obligation theorem makes the composed read the continuation row in C obtained by replaying the first compiler route, the middle-stage handoff, and the second compiler route. The first route reaches the target endpoint of K_{SM}, the second route starts from the source endpoint of K_{MT}, and the compatibility row is exactly the displayed identification between those two endpoint reads. Thus the composed read in B factors through the common middle row before any target read in T is available.

The same reasoning applies to B' with C' and T'. No alternative middle row can be inserted, because Definition 473.18 forces a composite of the two accepted compiler packets to hand off only through their displayed middle NameCert row, and Definition 473.10 makes classifier transport pass through the displayed landing, hsame, and Cont rows of those same packets. Since the boundary carrier has no coordinate for an independent middle certificate, host compiler semantics, program equivalence theorem, or unrecorded target behavior, every composed boundary read of either accepted packet factors through that common middle row. □

Closure status for $\mathsf{NameCert}_{\mathsf{CertificatePipelineStageBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$ packet is generated from finite BHist rows for source, middle, and target certificate stages, two compiler-edge packets, compatibility rows at the shared middle endpoint, and the composed boundary read. The pipeline boundary is built by replaying the first compiler route, the middle-stage handoff, and the second compiler route through Cont before any target-facing row is available. Componentwise hsame transports source, middle, target, landing, and classifier rows, while Pkg records the certificate-compiler dependencies. The common middle row is therefore a displayed boundary coordinate, not an inferred host compiler equivalence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificatePipelineStageBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the stage-boundary carrier, middle-row handoff, edge-classifier exactness, and composed-boundary factorization for $\mathsf{CertificatePipelineStageBoundary}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificatePipelineStageBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim host compiler semantics, program equivalence, an independent hidden middle certificate, or unrecorded target behavior.

Upgrade path: Obligation closure requires Lean encodings of the boundary carrier, middle compatibility row, edge classifier transport, and composed-read factorization.

Chapter 575

AspectChain NameCert Construction

The $\mathsf{AspectChain}^{\uparrow}$ interface packages the mind-matter double-aspect reading of Definition 9415.12 as one finite certificate. The records-side rows expose the public material record surface of the chain, while the inscription-event-side rows expose the accepted event surface by which the same chain point is written, checked, and handed to local consumers. The inscription-event coordinate is the concrete sibling surface of chapter 519, while the locality rows are consumer boundaries only and do not collapse $\mathsf{AspectChain}^{\uparrow}$ into a global observer frame. The packet does not identify private mental states with host physical objects, and it does not add a metaphysical identity relation outside the displayed BHist data.

Definition 575.1 (AspectChain carrier)**.** *An* $\mathsf{AspectChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, I, G, L, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the records-side* BHist *row read as the material record surface of the chain point;*
2. *I is the inscription-event-side* $\mathsf{InscriptionEvent}^{\uparrow}$ *row for the same accepted chain point;*
3. *G is the public* GAP *provenance row recording the open commitment carried by the double-aspect reading;*
4. *L is the local* $\mathsf{LocalityCell}^{\uparrow}$ *or* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row through which nearby observers may read the public record without receiving an all-observer frame;*
5. *O is the optional* $\mathsf{OtherMindsCommitment}^{\uparrow}$ *row by which a consumer may treat another inscription event as another conscious observational system;*
6. *H stores componentwise* hsame *transports for the records row, inscription-event row,* GAP *provenance, locality row, other-minds row, package row, and local name;*
7. *C stores the displayed* Cont *routes by which a consumer reads R, I, G, L, O without leaving the finite packet;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, GAP *provenance,* $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AspectChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows and their stored transports. It has no coordinate for a host mental substance, a host physical substance, a private phenomenal object, or an external identity predicate between mind and matter.

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChainTasteGate single carrier alignment`

Theorem 575.2 (AspectChain NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AspectChain}^{\uparrow}}$ *surface for the carrier of Definition 575.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet* $A = (R, I, G, L, O, H, C, P, N)$*;*
2. *records-side exactness: the material reading is exactly the* BHist *row* R *together with its displayed transports and consumer routes;*
3. *inscription-event exactness: the phenomenological-first-person reading is exactly the* $\mathsf{InscriptionEvent}^{\uparrow}$ *row* I *together with its public checker, consumer, and* GAP *ledger surface;*
4. *locality and other-minds accountability: every cross-observer read factors through* L *and, when present, through* O*;*
5. *non-collapse: the packet records two readings of one accepted chain point, but it exports neither a host mental state, nor a host physical object, nor a metaphysical identity relation between them.*

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChainNameCertObligations`

Proof. Carrier admission is projection from Definition 575.1. The rows $R, I, G, L, O, H, C, P, N$ are all displayed coordinates of one finite BHist packet.

The records-side obligation reads only R and the attached rows in H and C. Thus the material reading is a public record surface rather than an additional host object.

The inscription-event obligation reads I by applying the public carrier and downstream-handoff surface of $\mathsf{InscriptionEvent}^{\uparrow}$. Its checker, consumer, and GAP ledger data remain rows inside the same packet.

For locality and other-minds accountability, a consumer must route through L before treating the record as a shared local cell, and through O before treating another inscription event as another conscious observational system. The remaining rows H, C, P, N only transport, route, record provenance, and name this finite packet. □

Theorem 575.3 (AspectChain consumer boundary)**.** *Every consumer of an accepted* $\mathsf{AspectChain}^{\uparrow}$ *carrier may read only the records-side* BHist *row, the* $\mathsf{InscriptionEvent}^{\uparrow}$ *row, the* GAP *provenance row, the locality-cell row, the other-minds commitment row, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

In particular, a consumer may use the packet to compare the records-side and inscription-event-side readings of one chain point, but it receives no hidden mental state, no host physical substrate object, no all-observer locality frame, and no unrecorded cross-aspect bridge.

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChainConsumerBoundary`

Proof. Start with a consumer route into A. By the carrier definition, the only visible rows are $R, I, G, L, O, H, C, P, N$.

The records read is routed through R. The inscription-event read is routed through I, and Definition 519.5 bounds that read by the public source, name, route, check, consumer, ledger, transport, continuation, package, and naming rows of the inscription event.

The locality and other-minds reads are bounded by their displayed rows. The locality row prevents the consumer from importing a global observer frame, while the other-minds row records a commitment surface rather than direct access to another private state.

Consequently the consumer boundary is exactly the finite packet surface. The comparison of the two aspects is an internal two-reading relation over A, not an exported host-level identity or bridge object. □

Theorem 575.4 (AspectChain non-collapse)**.** *An accepted* $\mathsf{AspectChain}^{\uparrow}$ *carrier realizes the mind-matter double-aspect reading as one finite* BHist *packet with records-side and inscription-event-side rows. This realization does not collapse the records-side row into the inscription-event row, does not collapse the inscription-event row into the records-side row, and does not export host mental-state identity, host physical-object identity, private phenomenal access, or ambient metaphysical identity.*

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChainCarrier non collapse`

Proof. Project the carrier $A = (R, I, G, L, O, H, C, P, N)$. The records-side coordinate R and the inscription-event coordinate I are distinct displayed rows of the same finite packet.

Definition 575.2 fixes the two exactness obligations separately. The records-side reading is exhausted by R and its routes, while the inscription-event reading is exhausted by I and its public inscription-event surface.

Definition 575.3 then bounds every public consumer by the same finite rows. Since the packet contains no coordinate for private phenomenal access, host physical-object identity, or an external identity predicate, no consumer can collapse either aspect into the other or export a metaphysical identity claim. □

Theorem 575.5 (AspectChain public export)**.** *The public export of an accepted* $\mathsf{AspectChain}^{\uparrow}$ *carrier is exactly the records-side row R, inscription-event row I,* GAP *provenance row G, locality row L, other-minds row O, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{AspectChain}^{\uparrow}}$ *row N. It exports no host mind–matter identity, private phenomenal object, host physical substrate object, all-observer locality frame, or global observer frame.*

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChainPublicExport`

Proof. Project the accepted carrier through Definition 575.1. The only displayed rows are $R, I, G, L, O, H, C, P, N$, and the classifier has no coordinate outside those rows.

Apply Definition 575.2. The records-side, inscription-event, locality, other-minds, and non-collapse obligations identify the public object surface and exclude host mental or physical identities.

Finally use Definition 575.3 and Definition 575.4. Every consumer read is bounded by the same finite packet, so the public export is the listed row surface and not a global observer frame or ambient mind–matter bridge. □

Theorem 575.6 (AspectChain bridge-status boundary)**.** *The public export rows of Definition 575.5 and the non-collapse rows of Definition 575.4 do not constitute a standard bridge, host mind–matter identity, or bridge-checked interpretation. Any bridge consumer for* $\mathsf{AspectChain}^{\uparrow}$ *must add a separate bridge row beyond*

$$R, I, G, L, O, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChain bridge status boundary`

Proof. Project the accepted carrier through Definition 575.1. The rows $R, I, G, L, O, H, C, P, N$ are records, inscription, GAP provenance, locality, other-minds, transport, replay, provenance, and naming rows.

Apply Definition 575.5 to identify the public surface, and apply Definition 575.4 to keep the records-side and inscription-event-side readings from becoming host identities. These theorems bound what the packet exports; they do not add an external interpretation row.

The closure block records bridge status none. Thus no standard bridge, host mind–matter identity, or bridge-checked interpretation is named by the displayed packet, and any bridge consumer must supply an additional bridge row outside the listed carrier. □

Theorem 575.7 (AspectChain bridge handoff)**.** *Every bridge-facing consumer of an accepted* $\mathsf{AspectChain}^{\uparrow}$ *packet must read the public export of Definition 575.5 and the non-collapse boundary of Definition 575.4 before it adds any separate bridge row. The handoff is a paper-side bridge boundary: it orders the public records-side, inscription-event-side, locality, other-minds, transport, replay, provenance, and naming rows for later bridge use, but it does not itself certify a standard bridge, host mind–matter identity, or machine-checked bridge interpretation.*

Lean4 ✓ `BEDC.Derived.AspectChainUp.AspectChain bridge handoff`

Proof. Project the accepted carrier $A = (R, I, G, L, O, H, C, P, N)$. The only bridge-facing input available from the chapter is the same finite packet surface already named by the public export theorem.

Apply Definition 575.4. The records-side and inscription-event-side readings remain two displayed rows of one packet, so a later bridge row cannot treat either reading as an already exported host identity.

Apply Definition 575.6. Any standard bridge must be supplied as a separate row beyond the displayed carrier. Thus the handoff records the paper boundary for bridge consumers while leaving bridge verification outside this chapter. □

Theorem 575.8 (AspectChain mature consumer completeness)**.** *The mature consumer surface of an accepted* $\mathsf{AspectChain}^{\uparrow}$ *packet is exhausted by the public export of Definition 575.5, the non-collapse boundary of Definition 575.4, the bridge-status boundary of Definition 575.6, and the bridge handoff of Definition 575.7. In particular, every mature consumer either reads the displayed packet rows*

$$R, I, G, L, O, H, C, P, N$$

or asks for a bridge row whose absence is already recorded by the boundary and handoff theorems; no further consumer class receives private phenomenal access, host mental-state identity, host physical-object identity, global observer locality, or a bridge-checked interpretation from this packet.

Proof. Start with the public export theorem. It identifies the only rows exported by an accepted carrier: records, inscription-event, GAP provenance, locality, other-minds, transport, replay, provenance, and local naming.

Apply the non-collapse theorem to the same rows. It prevents a mature consumer from turning the records-side and inscription-event-side readings into host identities, private phenomenal access, or an ambient mind–matter identity predicate.

For bridge-facing reads, use the bridge-status boundary and then the bridge handoff. The boundary states that the displayed rows do not themselves constitute a standard bridge, and the handoff orders exactly what a later bridge consumer must read before supplying any separate bridge row.

These four surfaces cover the public, non-collapse, bridge-boundary, and bridge-handoff consumers of the packet. Since the carrier has no coordinate outside $R, I, G, L, O, H, C, P, N$ and the bridge row is explicitly external to the chapter, there is no remaining mature consumer class to close. □

Closure status for $\mathsf{NameCert}_{\mathsf{AspectChain}\uparrow}$.

Constructive story: An $\mathsf{AspectChain}^{\uparrow}$ packet is generated from one records-side BHist row, one $\mathsf{InscriptionEvent}^{\uparrow}$ row, public GAP provenance, a locality-cell row, an optional other-minds commitment row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.AspectChainUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AspectChain}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 575.1, Definition 575.2, Definition 575.3, Definition 575.4, Definition 575.5, Definition 575.6, Definition 575.7, and Definition 575.8 bind the public records-side row, inscription-event row, GAP provenance, locality and other-minds rows, hsame transport, Cont routes, Pkg provenance, local NameCert rows, bridge-facing boundary, and mature consumer completeness already listed by the packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AspectChain}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge checking, private phenomenal access, host mental-state identity, host physical-object identity, global observer frame, or ambient metaphysical identity.

Upgrade path: Bridge verification requires a separate standard bridge row over the already listed packet rows.

Chapter 576

A Concrete Naming Certificate for $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$

The $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ interface records a causal-commitment failure as a finite contrary-record packet. It turns the causality reading of section 9415.1 into a concrete naming surface: an observed regularity ledger, a forward-binding Gap commitment, a contrary record, and the GAP non-escape ledger through which downstream consumers may inspect the failure. The finite contrary-record row specializes the witness-row discipline of chapter 503 and is consumed by bounded-search refutation through chapter 579 as a finite branch surface.

Definition 576.1 (Finite witnessed refutation carrier)**.** *A* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, G, K, V, D, H, C, P, N)$$

with the following visible rows:

1. *R is the finite observed-regularity ledger that supported the causal commitment;*
2. *G is the* Gap*-tagged forward-binding commitment asserting that the recorded regularity persists beyond the currently observed history;*
3. *K is the finite commitment key naming the prediction, tolerance, domain cell, and record shape against which a contrary record is checked;*
4. *V is the contrary-record witness, displayed as a finite observation row whose classifier conflicts with the commitment key;*
5. *D is the decision ledger recording the local comparison between K and V;*
6. *H stores the displayed* hsame *transports for the regularity ledger, commitment row, commitment key, contrary witness, and decision ledger;*
7. *C stores the* Cont *routes by which consumers inspect the support record, commitment, witness, and decision row;*
8. *P is the* Pkg *provenance row over* BHist, hsame, Cont, Gap, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWitnessedRefutation}^{\uparrow}}$ *registration row.*

The local classifier compares two carriers only through these rows. It does not decide universal causality, does not choose a host law, and does not erase the Gap *commitment; it records that the named finite witness conflicts with the named finite commitment key.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier single carrier alignment`

Theorem 576.2 (Finite witnessed refutation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteWitnessedRefutation}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier formation for the finite packet of Definition 576.1;*
2. *classifier stability for the transported support ledger R, commitment row G, key K, contrary witness V, and decision ledger D;*
3. *witness exactness, requiring the contrary record consumed by the refutation to be exactly the displayed row V;*
4. *decision exactness, requiring the conflict claim to be read only through the comparison row D between K and V;*
5. *package closure, requiring the* hsame, Cont, Pkg, *and* NameCert *rows to contain exactly the provenance used by downstream causal-commitment consumers.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier namecert obligations`

Proof. Carrier formation is the tuple assembly of Definition 576.1. The support ledger, commitment, key, contrary witness, decision row, transport rows, continuation routes, package provenance, and naming row are all visible coordinates of F.

Classifier stability is componentwise. Transport the support ledger and commitment first, then transport the key and contrary witness, and finally transport the decision ledger that compares them.

Witness exactness and decision exactness are projections of the same finite packet. A consumer may cite a refutation only by reading V against K through D.

Package closure follows by projecting H, C, P, and N. No host causal law, all-future causal closure, or external failure oracle occurs as a coordinate of the carrier. □

Theorem 576.3 (Finite witnessed refutation ledger exactness)**.** *For every accepted packet*

$$F = (R, G, K, V, D, H, C, P, N),$$

the public refutation read is exactly the finite route from the support ledger R and forward-binding commitment G to the key K, contrary witness V, and comparison ledger D, transported by H, routed by C, packaged by P, and named by N. No refutation read may use a witness, comparison, or support record absent from the packet.

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationLedgerExactness`

Proof. Begin with Definition 576.2. Its witness exactness clause binds every contrary record to V, and its decision exactness clause binds the conflict claim to D.

The support side is also finite. The rows R and G state which observed regularity and which forward-binding Gap commitment are under inspection, while K states the finite prediction cell tested by V.

The remaining rows H, C, P, N carry transport, continuation, provenance, and local naming. Hence the refutation route is exactly the displayed finite ledger, not a hidden appeal to an ambient theory of causation. □

Theorem 576.4 (Finite witnessed refutation non-escape)**.** *The public boundary of* FiniteWitnessedRefutation$^{\uparrow}$ *exports only the finite contrary-record packet over* BHist*,* Gap*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It exports no host causal law, no global falsification principle, no all-future causal closure, no external observation selector, and no erasure of the original* Gap *commitment.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier non escape`

Proof. Inspect the carrier coordinates of Definition 576.1. The only failure-facing rows are the support ledger, commitment row, key, contrary witness, and decision ledger.

Apply Definition 576.3. Every public refutation read passes through those rows and their displayed transport, continuation, package, and naming data.

The excluded objects are absent from the tuple. The packet therefore records a finite witnessed conflict with a causal commitment, while the wider question of universal causal closure remains outside the exported boundary. □

Theorem 576.5 (Finite witnessed refutation public packet)**.** *An accepted* FiniteWitnessedRefutation$^{\uparrow}$ *packet is a public causal-commitment failure surface for downstream BEDC consumers. Such a consumer may inspect the supporting observations, the forward-binding* Gap *row, the commitment key, the contrary witness, and the decision ledger, but every read must factor through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by the carrier.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier public packet`

Proof. Use Definition 576.2 to obtain the carrier, stability, witness, decision, and package clauses. These clauses identify the rows available to a public consumer.

Next apply Definition 576.3. It forces the refutation read through $R, G, K, V, D, H, C, P, N$ and rules out any absent witness or comparison row.

Finally apply Definition 576.4. Since the carrier has no coordinate for a host law, global falsification principle, external observation selector, or erased Gap, the exported object is exactly the finite public packet. □

Theorem 576.6 (Finite witnessed refutation public consumer export)**.** *Every downstream consumer reading an accepted* FiniteWitnessedRefutation$^{\uparrow}$ *packet through*

$$R, G, K, V, D, H, C, P, N$$

must also expose the GapClosureBoundary$^{\uparrow}$ *and* BoundedSearchRefutation$^{\uparrow}$ *sibling routes before treating the contrary record as public refutation data. The exported surface contains only the finite carrier rows, local* NameCert *obligations, ledger exactness, non-escape, public packet route, and Gap boundary crosslink; it contains no universal causality law, host-law selector, global falsification principle, all-future causal closure, or erased* Gap *commitment.*

Proof. Begin with Definition 576.2. It identifies the carrier, local naming, stability, witness, decision, transport, continuation, and package rows available to a refutation consumer.

Apply Definition 576.3 and Definition 576.4. These force every read through the displayed rows $R, G, K, V, D, H, C, P, N$ and rule out host laws, external selectors, erased commitments, and global falsification data.

Finally combine Definition 576.5 with Definition 576.7. The public packet supplies the finite export surface, while the Gap boundary and bounded-search sibling route keep the contrary record local to the displayed branch rather than promoting it to universal causal closure. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier public export`

Theorem 576.7 (Finite witnessed refutation Gap boundary crosslink)**.** *For an accepted* FiniteWitnessedRefutation$^{\uparrow}$ *packet*

$$F = (R, G, K, V, D, H, C, P, N),$$

the forward-binding Gap *row G and the refusal/non-escape reading of the refutation are governed by the common boundary of chapter 1006. The finite witnessed refutation supplies the contrary record V and decision ledger D, while* GapClosureBoundary$^{\uparrow}$ *supplies the rule that unsupported closure requests route to refusal rather than to internal universal closure.*

The same finite contrary-record row V and decision ledger D are also the branch surface consumed by chapter 579: bounded-search refutation reads the displayed finite branch witness, while GapClosureBoundary$^{\uparrow}$ *supplies the refusal boundary that prevents the branch from becoming universal causal closure.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier gap boundary crosslink`

Proof. Project the finite refutation carrier. The rows $R, G, K, V, D, H, C, P, N$ display the support ledger, forward-binding Gap commitment, commitment key, contrary witness, decision ledger, transport, continuation, provenance, and local name.

Apply the public packet theorem above. It lets downstream consumers inspect R, G, K, V, D only through the same structural rows H, C, P, N and does not erase the original Gap commitment.

Now use the GapClosureBoundary source and refusal surface of Definition 1006.2 and Definition 1006.3. That common carrier routes unsupported attempts to turn a Gap request into internal closure to the refusal ledger, so the finite witnessed refutation remains a contrary-record packet rather than a universal closure principle. □

Theorem 576.8 (Finite witnessed refutation mature budget)**.** *The* FiniteWitnessedRefutation$^{\uparrow}$ *packet is a mature contrary-record branch surface for downstream bounded-refutation consumers. Its carrier, NameCert obligations, ledger exactness, non-escape theorem, public packet theorem, public consumer export, and Gap boundary crosslink give* GapClosureBoundary$^{\uparrow}$ *and* BoundedSearchRefutation$^{\uparrow}$ *exactly the finite support ledger, forward-binding* Gap *commitment, commitment key, contrary witness, decision ledger, transport, replay, provenance, and local naming rows they consume. The mature surface still refuses universal causality, global falsification, all-future causal closure, host-law selection, Lean verification, and standard bridge checking.*

Proof. Use Definition 576.1 and Definition 576.2. They expose the finite contrary-record branch as $R, G, K, V, D, H, C, P, N$ and bind every public read to those coordinates.

Apply Definition 576.3, Definition 576.4, and Definition 576.5. These results identify the reusable public packet while ruling out host causal laws, external selectors, erased Gap commitments, and global falsification data.

Finally combine Definition 576.6 with Definition 576.7. The downstream routes receive exactly the contrary-record branch consumed by GapClosureBoundary and bounded-search refutation, so no further paper-side theory grade remains for the finite branch surface itself. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessedRefutationUp.FiniteWitnessedRefutationCarrier mature budget`

Closure status for NameCert$_{\text{FiniteWitnessedRefutation}^{\uparrow}}$**.**

Constructive story: A FiniteWitnessedRefutation$^{\uparrow}$ packet is generated from a finite support ledger, a forward-binding Gap row, a commitment key, a contrary witness, a decision ledger, displayed transports, continuation routes, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteWitnessedRefutation}\uparrow}$, maturePackageClosed)

Scope closed: The mature surface over BHist, Gap, hsame, Cont, Pkg, and NameCert is exactly Definition 576.1, Definition 576.2, Definition 576.3, Definition 576.4, Definition 576.5, Definition 576.6, Definition 576.7, and Definition 576.8.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWitnessedRefutation}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close universal causality, global falsification, all-future causal closure, host-law selection, bridge checking, or Lean verification.

Upgrade path: No further theory-axis upgrade remains for this finite contrary-record surface; Lean verification and bridge checking remain outside the claim.

Chapter 577

A Concrete Naming Certificate for $\mathsf{AnalyticContinuationOperation}^{\uparrow}$

The $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ interface turns the operation surface of chapter 9341 into a concrete BEDC operation packet. It names the holomorphic input rows, complex-domain rows, overlap evidence, continuation predicate, output row, branch-obstruction ledger, transport rows, package provenance, and local naming row that a downstream consumer can inspect without appealing to host quotient equality or an undisplayed branch selector.

Definition 577.1 (Analytic continuation operation carrier)**.** *An* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$$

with the following displayed rows:

1. D_1, D_2 *are the two* $\mathsf{Complex}^{\uparrow}$ *domain rows;*
2. F *is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row over* D_1*;*
3. G *is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row over* D_2*;*
4. U *is the finite overlap row carrying the displayed domain-compatibility and pointwise* hsame *agreement evidence;*
5. A *is the analytic-continuation operation row, read as the* $\mathsf{AnaCont}^{\uparrow}$ *predicate over* F, G, U*;*
6. O *is the output* $\mathsf{Holomorphic}^{\uparrow}$ *row over the displayed union domain;*
7. L *is the continuation ledger, including the uniqueness row, domain-read rows, and the branch-obstruction row;*
8. H *stores componentwise* hsame *transports for* $D_1, D_2, F, G, U, A, O, L$*;*
9. C *stores the finite* Cont *routes by which consumers read the inputs, overlap evidence, output row, ledger rows, package provenance, and local naming row;*
10. P *is the* Pkg *provenance over the same finite operation packet;*
11. N *is the local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationOperation}^{\uparrow}}$ *row.*

The branch-obstruction row in L *is a recorded boundary. It is not a closed general decision procedure for monodromy, branch selection, or path-dependent continuation.*

Lean4 ✓ BEDC.Derived.AnalyticContinuationOperationUp.AnalyticContinuationOperationTasteGate single carrier alignment

Theorem 577.2 (Analytic continuation operation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AnalyticContinuationOpera}}$ *surface for Definition 577.1 consists of five finite rows: carrier admission for Q; classifier stability for the* $\mathsf{Complex}^{\uparrow}$ *domain rows, the* $\mathsf{Holomorphic}^{\uparrow}$ *input rows, the overlap row, the* $\mathsf{AnaCont}^{\uparrow}$ *operation row, and the output row; continuation-ledger exactness for L;* Pkg *provenance over $D_1, D_2, F, G, U, A, O, L, H, C$; and local naming exactness for N.*

Proof. Carrier admission is projection of the finite tuple Q. Each domain, input, overlap, operation, output, ledger, transport, continuation, package, and naming coordinate is displayed as a row of the same BHist packet.

Classifier stability is componentwise. The $\mathsf{Complex}^{\uparrow}$ rows are transported through their displayed domain classifier, the $\mathsf{Holomorphic}^{\uparrow}$ rows through their holomorphic classifier, and the overlap row through the stored pointwise hsame agreement evidence. The operation row A is stable because the $\mathsf{AnaCont}^{\uparrow}$ predicate is read only through the transported inputs, overlap evidence, and output row.

The package and naming rows do not add analytic content. They repack the same finite rows as Pkg provenance and local NameCert exactness, so the obligation surface is exactly the five displayed rows. □

Lean4 ✓ BEDC.Derived.AnalyticContinuationOperationUp.AnalyticContinuationOperationCarrier namecert obligations

Theorem 577.3 (Analytic continuation operation stability)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$ be an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier. Any componentwise* hsame *transport of the two* $\mathsf{Complex}^{\uparrow}$ *domain rows, the two* $\mathsf{Holomorphic}^{\uparrow}$ *input rows, the overlap evidence, and the output row preserves carrier admission, the* $\mathsf{AnaCont}^{\uparrow}$ *operation row, the continuation ledger L, the* Pkg *row P, and the local* NameCert *row N.*

Proof. Begin with the transports stored in H. They identify the two domain rows, the two holomorphic input rows, the overlap row, the operation row, the output row, and the ledger row component by component.

The $\mathsf{AnaCont}^{\uparrow}$ predicate has no hidden carrier outside those coordinates. After transporting D_1, D_2, F, G, U and O, the restriction clauses and overlap-agreement clause remain the same clauses under hsame.

The ledger row is transported by exact coordinate equality in H. The uniqueness and domain-read rows therefore remain attached to the same operation packet, and the branch-obstruction row remains a boundary row rather than an exported continuation witness.

Finally, Pkg provenance and local naming read the transported packet through C. Thus P and N classify the transported operation packet without changing the operation surface. □

Theorem 577.4 (Analytic continuation operation ledger exactness)**.** *For an accepted* AnalyticContinuationOperatio *carrier $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$, every exported continuation fact factors through the displayed rows $D_1, D_2, F, G, U, A, O, L, H, C, P, N$. Requests for branch choice, quotient-of-germs equality, or continuation data outside the displayed overlap and output rows are recorded only in the branch-obstruction coordinate of L and are not exported as* $\mathsf{Holomorphic}^{\uparrow}$ *rows.*

Proof. Read an exported fact through the continuation routes stored in C. If the route lands in the two domain rows, input rows, overlap row, operation row, output row, transport row, package row, or local naming row, exactness is immediate because the coordinate is part of Q.

If the route asks for a continuation object beyond those coordinates, no displayed row of Q carries such an object. The ledger L is then the only classifier-visible destination for the request.

Inside L, the branch-obstruction coordinate records the boundary. It is a ledger entry saying that this route is outside the closed packet, not a constructive decision procedure for all branch behaviour and not a hidden source of a new holomorphic output row. □

Theorem 577.5 (Analytic continuation operation public export)**.** *The public surface of an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier exports exactly the finite operation packet: two* $\mathsf{Complex}^{\uparrow}$ *domain rows, two* $\mathsf{Holomorphic}^{\uparrow}$ *input rows, the overlap and pointwise* hsame *evidence, the* $\mathsf{AnaCont}^{\uparrow}$ *operation row, the output* $\mathsf{Holomorphic}^{\uparrow}$ *row, the continuation ledger, componentwise* hsame *transports, finite* Cont *read routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *exactness.*

Proof. The carrier definition lists every row available to public consumers. The NameCert obligations identify carrier admission, classifier stability, ledger exactness, package provenance, and local naming for precisely those rows.

Stability ensures that changing representatives inside the displayed hsame classifiers does not change the exported operation packet. Ledger exactness prevents undisplayed branch data, quotient equality, or host continuation principles from entering the export.

Therefore the public export is exactly the finite BEDC operation packet over the holomorphic inputs, overlap evidence, output row, continuation ledger, transport routes, package provenance, and local naming row. □

Definition 577.6 (Analytic continuation operation socket extraction)**.** *Let* $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$ *be an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier, and suppose that the branch-obstruction coordinate of the continuation ledger* L *is displayed as* B*. The socket extracted from* Q *along* B *is the finite packet*

$$\mathrm{Sock}(Q, B) = (F, G, U, A, O, B, H, C, P, N).$$

It is read by retaining the two holomorphic source rows, the finite overlap row, the analytic-continuation operation row, the output row, the displayed branch-obstruction coordinate, the componentwise transports, the continuation routes, the package provenance, and the local naming row, while forgetting the two explicit domain rows and the non-obstruction ledger coordinates of L*.*

Theorem 577.7 (Analytic continuation operation socket extraction)**.** *Let* $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$ *be an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier. If the branch-obstruction coordinate of* L *is displayed as* B*, then the extracted packet*

$$\mathrm{Sock}(Q, B) = (F, G, U, A, O, B, H, C, P, N)$$

is an accepted $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *carrier. Every* $\mathsf{Holomorphic}^{\uparrow}$ *output read exported through the operation packet factors through this socket: the read reaches* O *only through the source rows* F, G*, the overlap witness* U*, the operation row* A*, and the visible* H, C, P, N *routes, while every branch-sensitive request outside that source-overlap-output path is recorded in* B *and is not exported as another* $\mathsf{Holomorphic}^{\uparrow}$ *row.*

Proof. Project the accepted operation carrier using Definition 577.1. The rows F, G, U, A, O are displayed coordinates of Q: F and G are the two $\mathsf{Holomorphic}^{\uparrow}$ source rows, U is the finite overlap row, A is the $\mathsf{AnaCont}^{\uparrow}$ operation row over F, G, U, and O is the output $\mathsf{Holomorphic}^{\uparrow}$ row. The ledger row L contains the displayed branch-obstruction coordinate B by hypothesis.

The domain rows D_1, D_2 are not independent socket coordinates. In the socket carrier of Definition 569.1, the domains are read through the source rows F, G and the overlap witness

U. Thus forgetting D_1, D_2 does not remove a socket field; it leaves exactly the source-overlap-operation-output surface F, G, U, A, O.

The remaining socket rows are inherited from the same finite operation packet. The transport row H already stores componentwise hsame transport for F, G, U, A, O and for the ledger containing B. The continuation row C already stores the finite Cont routes by which consumers read the inputs, overlap evidence, output row, ledger rows, package provenance, and local naming row. The package row P and the naming row N repack these retained coordinates as the local socket surface. Therefore the tuple $\mathrm{Sock}(Q, B)$ has exactly the carrier coordinates required by Definition 569.1.

Carrier admission and classifier stability are supplied by the operation NameCert obligations of Definition 577.2: they apply componentwise to the holomorphic source rows, overlap row, operation row, and output row. Ledger exactness for L restricts to ledger exactness for the displayed branch-obstruction coordinate B. These are precisely the socket obligation rows listed in Definition 569.2, so the extracted packet is accepted as an $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ carrier.

It remains to check the output read. By operation ledger exactness in Definition 577.4, every exported continuation fact from Q factors through the displayed operation rows, and every request for branch choice, quotient-of-germs equality, or continuation data outside the displayed overlap and output rows is recorded only in the branch-obstruction coordinate of L. Under the extraction, that coordinate is B.

Now apply the socket output factorization theorem Definition 569.9 to $\mathrm{Sock}(Q, B)$. The output row O factors through F, G, U, A, H, C, P, N, and the branch ledger B records exactly the attempted reads that do not factor through those rows. The holomorphic handoff theorem Definition 569.7 then prevents a ledger entry from becoming a second output row. Hence every $\mathsf{Holomorphic}^{\uparrow}$ output read through the operation packet factors through the extracted socket and no hidden branch selector or unrestricted analytic-continuation operator is exported. □

Theorem 577.8 (Analytic continuation operation socket ledger consumer)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L, H, C$ be an accepted $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ carrier, and let B be the displayed branch-obstruction coordinate of L. Every socket consumer of the operation output reads O only through the finite packet*

$$F, G, U, A, O, B, H, C, P, N.$$

The consumer may inspect the BHist ledger, componentwise hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row, but it cannot read a second $\mathsf{Holomorphic}^{\uparrow}$ output or a branch-sensitive continuation object outside the socket ledger.

Proof. Begin with socket extraction, Definition 577.7. It identifies the operation socket as $\mathrm{Sock}(Q, B) = (F, G, U, A, O, B, H, C, P, N)$ and proves that the output read factors through this packet.

Now apply ledger exactness and public export. Ledger exactness confines branch-sensitive requests to L, and public export lists no output row beyond the displayed $\mathsf{Holomorphic}^{\uparrow}$ row O.

Since B is the branch-obstruction coordinate of L, a socket consumer can inspect it only as ledger data. It cannot reclassify that ledger row as another output, so every operation-output read remains inside $F, G, U, A, O, B, H, C, P, N$. □

Theorem 577.9 (Analytic continuation operation domain-overlap boundary)**.** *For an accepted $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ carrier $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$, the $\mathsf{Complex}^{\uparrow}$ domain rows D_1, D_2 and the overlap row U must be displayed before any $\mathsf{Holomorphic}^{\uparrow}$ output row is exported. A consumer can reach O only after reading the source rows F, G, the domain-overlap evidence D_1, D_2, U, the $\mathsf{AnaCont}^{\uparrow}$ operation row A, and the structural rows H, C, P, N; branch-sensitive requests remain in the ledger and cannot become a second output.*

Lean4 ✓ BEDC.Derived.AnalyticContinuationOperationUp.AnalyticContinuationOperation domain overlap boundary

Proof. Project the operation carrier. The output row O is listed after the two domain rows, the two holomorphic source rows, the overlap row, and the operation row.

Use overlap classifier exactness for the domain-overlap side. It forces every overlap-classifier read through D_1, D_2, F, G, U, A and through the structural rows H, C, P, N before any output-facing read is available.

Finally apply the socket ledger consumer theorem. After the output route reaches the socket surface, the branch-obstruction coordinate remains ledger data. Thus no branch-sensitive request can bypass the displayed domain-overlap rows or become a second $\mathsf{Holomorphic}^{\uparrow}$ output. □

Theorem 577.10 (Analytic continuation operation overlap classifier exactness)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L,$ be an accepted $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ carrier. Every overlap-classifier read of Q factors through the displayed domain rows D_1, D_2, source rows F, G, finite overlap row U, operation row A, and structural rows H, C, P, N. The classifier cannot replace the local overlap evidence by monodromy, quotient-of-germs equality, a branch selector, or an undisplayed continuation object.*

Proof. Project the carrier of Definition 577.1. The rows $D_1, D_2, F, G, U, A, O, L, H, C, P, N$ are the complete operation packet, and U is the only displayed overlap row.

Use the NameCert obligations and stability, Definition 577.2 and Definition 577.3. They preserve the two domain rows, the two holomorphic source rows, the overlap row, the operation row, and the output row componentwise under hsame transport.

Finally apply public export and socket extraction, Definition 577.5 and Definition 577.7. Downstream overlap reads pass to the socket only through $F, G, U, A, O, H, C, P, N$, while the domain rows are read through those source and overlap coordinates. No separate branch selector or quotient continuation row is available. □

Theorem 577.11 (Analytic continuation operation branch-ledger non-escape)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L,$ be an accepted $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ carrier. Branch-sensitive reads of Q remain ledger entries in L unless they factor through the displayed source-overlap-operation-output route $F, G, U, A, O, H, C, P, N$. The branch obstruction is therefore a ledger boundary, not a hidden selector, host continuation oracle, quotient-of-germs equality, global monodromy decision, or second $\mathsf{Holomorphic}^{\uparrow}$ output.*

Proof. Begin with ledger exactness, Definition 577.4. It sends every continuation request outside the displayed overlap and output rows to the branch-obstruction coordinate of L.

Use public export to bound the available rows. The exported packet lists the two domains, two sources, overlap evidence, operation row, output row, ledger, transports, routes, package provenance, and local naming; it does not list a selector, oracle, quotient equality, global monodromy decision, or additional output row.

Apply socket extraction. When a displayed branch-obstruction coordinate is retained as B, the socket can hand off only the source-overlap-operation-output route, and branch-sensitive requests outside that route remain in B. Since B is a coordinate of L, the original operation packet also keeps those requests as ledger data. □

Theorem 577.12 (Analytic continuation operation socket handoff exactness)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L, H$ be an accepted $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ carrier, and let B be the displayed branch-obstruction coordinate of L. The handoff from Q to $\mathrm{Sock}(Q, B)$ preserves exactly the source rows F, G, overlap row U, operation row A, output row O, branch ledger B, transports H, routes C, provenance P,*

and local name N. It forgets only the explicit domain coordinates D_1, D_2 after they have been consumed by the source and overlap rows.

Proof. Start from the socket extraction theorem Definition 577.7. It defines the extracted packet as $\mathrm{Sock}(Q, B) = (F, G, U, A, O, B, H, C, P, N)$ and proves that this packet is accepted by the socket carrier.

Use overlap classifier exactness. The domain rows D_1, D_2 have already been consumed through F, G, U before the socket read reaches A or O, so forgetting them as separate socket coordinates does not lose a public operation row.

Finally use branch-ledger non-escape. The branch coordinate B remains a ledger boundary and cannot become a hidden selector or another $\mathsf{Holomorphic}^{\uparrow}$ output. Therefore the handoff preserves exactly the displayed socket coordinates and no additional continuation data. □

Theorem 577.13 (Analytic continuation operation contour socket handoff)**.** *Let $Q = (D_1, D_2, F, G, U, A, O, L, H,$ be an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier, and let B be the displayed branch-obstruction coordinate of L. Any* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *consumer of this operation must read the extracted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *packet*

$$\mathrm{Sock}(Q, B) = (F, G, U, A, O, B, H, C, P, N)$$

together with the two $\mathsf{Holomorphic}^{\uparrow}$ *source rows F, G, the overlap row U, the operation row A, the output row O, and the branch ledger B before a contour-facing route is exposed. No contour-facing read bypasses $D_1, D_2, F, G, U, A, O, L, H, C, P, N$.*

Proof. Begin with socket extraction and handoff exactness. By Definition 577.7 and Definition 577.12, the operation exports the socket only as $\mathrm{Sock}(Q, B)$, with $F, G, U, A, O, B, H, C, P, N$ as its visible coordinates.

Next apply the domain-overlap boundary. The contour-facing consumer cannot reach the output row O until the rows $D_1, D_2, F, G, U, A, H, C, P, N$ have been read, so the two holomorphic source rows and the overlap row are already part of the route before contour data can be exposed.

Finally use branch-ledger non-escape. Branch-sensitive requests remain in L, and the displayed coordinate B is retained by the socket as ledger data. Thus every contour-facing handoff passes through the finite operation packet and no hidden branch selector, quotient-of-germs equality, or second holomorphic output is available to the consumer. □

Theorem 577.14 (Analytic continuation operation holomorphic route non-escape)**.** *For an accepted* $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ *carrier $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$, every holomorphic output route remains inside the displayed coordinates $D_1, D_2, F, G, U, A, O, L, H, C, P, N$. Such a route cannot become quotient-of-germs equality, a branch selection, a monodromy decision, or a second* $\mathsf{Holomorphic}^{\uparrow}$ *row.*

Proof. Use ledger exactness and public export. The ledger theorem confines every exported continuation fact to the displayed rows of Q, while public export lists exactly the two domain rows, two holomorphic source rows, overlap evidence, operation row, output row, ledger, transports, routes, package provenance, and local naming row.

Now apply the socket ledger consumer theorem. Once a holomorphic output route is read through the extracted socket, the only output coordinate is O, and the branch-obstruction coordinate is still ledger data.

The excluded alternatives have no coordinate in the packet. Quotient-of-germs equality, branch selection, monodromy decision, and an additional $\mathsf{Holomorphic}^{\uparrow}$ output are therefore not reachable from a holomorphic output route inside Q. □

Theorem 577.15 (Analytic continuation operation zeta consumer boundary)**.** *Any* ZetaContinuationWitness$^{\uparrow}$ *consumer that reads an* AnalyticContinuationOperation$^{\uparrow}$ *carrier* $Q = (D_1, D_2, F, G, U, A, O, L, H, C, P, N)$ *must first factor through chapter 569, chapter 521, and the displayed operation packet* $D_1, D_2, F, G, U, A, O, L, H, C,$ *Only after those rows have been exposed may the consumer read a zeta-continuation witness packet, and the route exports no RH zero-location consequence.*

Proof. First apply the contour socket handoff theorem. It forces every contour-facing route from the operation to pass through the extracted AnalyticContinuationSocket$^{\uparrow}$ packet, the two holomorphic source rows, the overlap row, the operation row, the output row, and the branch ledger.

Next apply holomorphic route non-escape. The route remains inside $D_1, D_2, F, G, U, A, O, L, H, C, P, N$ and cannot turn into quotient-of-germs equality, branch selection, monodromy decision, or an additional holomorphic row.

The zeta witness is then read only after the socket and contour chapters have consumed this finite operation packet. Since neither the operation packet nor the contour-facing handoff contains an RH predicate or a zero-location coordinate, the zeta consumer receives continuation data without any RH zero-location consequence. □

Theorem 577.16 (AnalyticContinuationOperation bridge consumption)**.** *Every bridge consumer of an accepted* AnalyticContinuationOperation$^{\uparrow}$ *packet must read the displayed source rows, the domain-overlap row, the operation row, the output row, the branch ledger, the socket handoff, the contour-integral consumer route, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row before using the operation. Such a consumer cannot use quotient-of-germs equality, global monodromy, a branch selector, or an RH zero-location consequence.*

Proof. Start with public export, Definition 577.5. It exposes exactly the finite operation surface over $D_1, D_2, F, G, U, A, O, L, H, C, P, N$.

Next use socket extraction and socket handoff exactness, Definition 577.7 and Definition 577.12. They force every bridge-facing operation read through the source rows, overlap row, operation row, output row, branch ledger, transports, routes, provenance, and local name.

Finally apply the contour socket handoff and zeta consumer boundary, Definition 577.13 and Definition 577.15. The contour consumer route receives only the displayed operation packet, while the zeta-facing route carries no RH zero-location coordinate. Ledger exactness keeps quotient-of-germs equality, global monodromy, and branch selection outside the exported bridge consumption surface. □

Closure status for NameCert$_{\text{AnalyticContinuationOperation}^{\uparrow}}$**.**

Constructive story: An AnalyticContinuationOperation$^{\uparrow}$ packet is generated from two Complex$^{\uparrow}$ domain rows, two Holomorphic$^{\uparrow}$ input rows, finite overlap and pointwise hsame evidence, an AnaCont$^{\uparrow}$ operation row, an output Holomorphic$^{\uparrow}$ row, a continuation ledger, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AnalyticContinuationOperation}^{\uparrow}}$, bridgedClosed)

Scope closed: Definition 577.1, Definition 577.2, Definition 577.5, and Definition 577.16 define the local operation carrier, obligation rows, exported operation certificate, and bridge-consumption route.

Formal verification: FormalStatus(NameCert$_{\text{AnalyticContinuationOperation}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AnalyticContinuationOperationUp.AnalyticContinuationOperationCarrier\_namecert\_obligat`

Bridge status: none

Origin: human

Not claimed: This chapter does not close global monodromy, quotient-of-germs construction, general branch selection, RH zero-location consequences, or bridge checking.

Upgrade path: Mature closure requires examples or completeness-style consumer coverage; machine bridge checking remains a separate verification target.

Chapter 578

A Concrete Naming Certificate for $\mathsf{ConsciousObserverState}^{\uparrow}$

The $\mathsf{ConsciousObserverState}^{\uparrow}$ interface turns the current-state side of the consciousness, universe, and observer frame into a finite naming surface. It records the present observer state, the self-recognition row attached to that state, the observation ledger visible at that state, a Gap acknowledgement for what remains unfulfilled, the displayed consumer routes, package provenance, and a local $\mathsf{NameCert}$ row. The packet is local: it depends on observer-history and locality-cell certificates but does not export a global observer, a shared universe, an external time scalar, a spacetime manifold, or a host-level self object.

Definition 578.1 (Conscious observer-state carrier)**.** *A* $\mathsf{ConsciousObserverState}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, s, R, L, G, H, C, P, N)$$

with the following visible rows:

1. *O is the observer-history row from which the present state is read;*
2. *s is the current-state row inside O;*
3. *R is the self-recognition row, recording that the observer at s recognizes the same identity carried by the inspected history prefix;*
4. *L is the observation ledger row containing the finite records available at s;*
5. *G is the* Gap *acknowledgement row for commitments named at s whose witness rows are not yet present in L;*
6. *H records componentwise* hsame *transport for the observer history, current state, self-recognition row, observation ledger, gap acknowledgement, continuation routes, package row, and local name row;*
7. *C records the displayed* Cont *routes by which a consumer reads O, then s, then R, L, and G;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, BHist, Gap, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *source certificates: the observer-history row is the packet certified in chapter 477, and the locality-cell row is the finite local-cell packet of chapter 513;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$ *row.*

The carrier has no coordinate for a detached self, a global universe object, an external clock, a manifold point, an all-observer frame, host psychological content, or an instantaneous remote observation.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier present route package`

Theorem 578.2 (Conscious observer-state NameCert obligations)**.** *For the carrier of Definition 578.1, the local* $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the finite packet* $S = (O, s, R, L, G, H, C, P, N)$*;*
2. *current-state recognition, requiring* R *to name the identity read from* O *at* s*;*
3. *observation-ledger exposure, requiring every present observation read to pass through* L*;*
4. Gap *acknowledgement exactness, requiring* G *to name only commitments attached to* s *whose witness rows are absent from* L*;*
5. *stability and non-escape under the componentwise* hsame *row* H*, displayed* Cont *routes in* C*, package provenance* P*, and local name row* N*.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 578.1. The packet displays the observer history, present state, self-recognition row, observation ledger, acknowledgement row, transport, routes, provenance, and local naming row.

The current-state recognition clause reads O before s and reads R only as the recognition row attached to that same state. The observation-ledger clause then exposes exactly the records in L as the universe-for-the-observer at s.

The acknowledgement clause reads G after L. Thus the gap row can name only commitments not witnessed by the current ledger, while remaining attached to the same present state.

Finally, H transports the displayed rows componentwise and C replays only their listed routes. Since no detached self, global universe, external clock, manifold coordinate, or all-observer frame occurs among the coordinates, the local certificate cannot export one. □

Theorem 578.3 (Conscious observer-state recognition exactness)**.** *For every accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the self-recognition row R *is readable only after the observer-history row* O *and the current-state row* s *have been displayed. Thus self-recognition cannot detach from the observer-history row that supplies the present state.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier recognition exactness`

Proof. Project the carrier of Definition 578.1. Its visible order exposes O as the observer-history row, then s as the current-state row inside that history, and only then R as the recognition row attached to that state.

Apply Definition 578.2. The NameCert surface requires current-state recognition to name the identity read from O at s, while the same carrier excludes a detached self, a global observer, an external clock, a manifold point, and host psychological content.

The structural rows H, C, P, N transport, route, package, and name that same displayed packet. They do not add a second recognition source, so R remains exact for the observer-history row already displayed by O and s. □

Theorem 578.4 (Conscious observer-state GAP acknowledgement exactness)**.** *For every accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the Gap *acknowledgement row* G *names only commitments attached to the current state* s *whose witness rows are absent from the present ledger* L*. It cannot supply remote records or witness rows outside that ledger boundary.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap acknowledgement exactness`

Proof. Project the same carrier. The acknowledgement row G occurs after the observer history O, current state s, recognition row R, and ledger row L, so it is not an independent source of observation data.

Use Definition 578.2. Its acknowledgement clause states that G names only commitments attached to s whose witness rows are absent from L.

The exclusions in Definition 578.1 remove every route to an instantaneous remote observation, global universe, external clock, manifold point, or detached self. Hence the acknowledgement is exact for the local ledger boundary and cannot provide a remote record. □

Theorem 578.5 (Conscious observer-state locality provenance exactness)**.** *For every accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the package row P *routes the current-state read through the* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *and* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *sibling certificates. The transport row* H *and continuation row* C *replay exactly those observer-history, current-state, recognition, ledger, and acknowledgement rows.*

Proof. Project the provenance coordinate in Definition 578.1. It names the observer-history row certified by chapter 477 and the finite local-cell packet certified by chapter 513.

Apply Definition 578.2. The accepted packet reads O, then s, then R, L, G, with stability and non-escape supplied only by H, C, P, N.

The row H transports the displayed coordinates componentwise, and C replays the same displayed routes. Since the carrier excludes a global observer, shared universe, external time scalar, spacetime manifold, host psychological content, and instantaneous remote observation, provenance exactness is exhausted by the two sibling certificates named in P and replayed by H and C. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier locality provenance exactness`

Theorem 578.6 (Conscious observer-state classifier stability)**.** *Let*

$$S = (O, s, R, L, G, H, C, P, N)$$

be an accepted $\mathsf{ConsciousObserverState}^{\uparrow}$ *carrier. Any classifier comparison of* S *with another accepted carrier is stable only after both packets expose the same finite observer-history surface, current-state row, self-recognition row, observation ledger row,* Gap *acknowledgement row, componentwise transport, displayed routes, package provenance, and local naming row. The classifier does not compare detached selves, global universes, host mental states, manifold coordinates, or instantaneous remote records.*

Proof. Project the accepted carriers. The classifier can inspect only the rows listed in Definition 578.1.

Transport by H moves the observer history, current state, self-recognition row, observation ledger, and acknowledgement row together. The routes in C then replay the same finite reads on the transported surface.

The provenance row P and the local name row N bind the compared rows to the same package shape. Because the carrier contains no detached self, global universe, host mental-state, manifold-coordinate, or instantaneous-remote-record coordinate, classifier stability is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier classifier stability`

Theorem 578.7 (Conscious observer-state ledger exactness)**.** *For every accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet, the ledger visible at the current state is exactly the observation row L together with the* Gap *acknowledgement row G, read through O, s, R, H, C, P, N. A consumer can therefore read the present observer-state surface as*

$$O \quad \textit{then} \quad s \quad \textit{then} \quad R, L, G \quad \textit{then} \quad H, C, P, N.$$

No additional row records a hidden universe, a separate self-substance, a global clock, a spacetime point, an all-observer synchronization map, or a remote record available without acknowledgement.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateTasteGate single carrier alignment`

Proof. Read the carrier in order. The history row O supplies the finite observer-history surface, and s selects the present row within that surface.

The self-recognition row R is read at s. The observation ledger L then supplies the present records, and the acknowledgement row G supplies the named gaps attached to that same state.

The remaining rows H, C, P, N account for transport, continuation, provenance, and local naming. They do not add observational content. Hence the ledger is exact for the displayed current-state packet and contains no hidden global or host-level coordinate. □

Theorem 578.8 (Conscious observer-state current ledger route)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every current-state consumer reads the observer-history row O, the current-state row s, the self-recognition row R, and the present ledger row L before it can read the Gap *acknowledgement row G, the componentwise transport row H, the displayed* Cont *routes C, the package provenance P, or the local* $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$ *row N.*

The route cannot begin from a detached self, a remote observation, a global observer, an external clock, a spacetime point, or host psychological content.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier current ledger route`

Proof. Project the carrier of Definition 578.1. Its row order displays O, s, R, L before the acknowledgement and route-support rows.

Apply the NameCert obligations of Definition 578.2. They attach recognition, ledger exposure, transport, continuation, provenance, and local naming to that same finite packet.

Use ledger exactness from Definition 578.7 and locality provenance from Definition 578.5. These results keep the current-state consumer on O, s, R, L before G, H, C, P, N can be consumed, and the carrier excludes every detached, remote, global, external, manifold, or host row. □

Theorem 578.9 (Conscious observer-state GAP ledger route)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the Gap *row G is only a continuation of the present ledger route through O, s, R, L. It may acknowledge commitments whose witness rows are absent from L, but it cannot supply a remote observation, a hidden universe row, a detached self, an external clock, a spacetime point, or a host-state observation.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap ledger route`

Proof. Read the carrier and NameCert obligations first. The acknowledgement row G is listed after the observer-history, current-state, recognition, and present-ledger rows.

Apply ledger exactness from Definition 578.7. The visible present surface is L together with G, both read at the current state s in the observer-history row O.

Finally use locality provenance from Definition 578.5. The rows H, C, P, N replay and name the same local route; they do not turn the acknowledgement into a remote observation or any excluded global, external, manifold, detached, or host coordinate. □

Theorem 578.10 (Conscious observer-state locality-cell handoff)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the observer-history certificate of chapter 477 and the observer-locality-cell certificate of chapter 513 are the only provenance channels for the current-state packet. Every handoff to a current-state consumer must pass through $O, s, R, L, G, H, C, P, N$ with P naming exactly those sibling certificate channels.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier locality cell handoff`

Proof. Project P from the carrier of Definition 578.1. The provenance coordinate names the observer-history and observer-locality-cell sibling certificates and no third provenance source.

Apply Definition 578.2. The local certificate requires the current-state row, recognition row, ledger row, and acknowledgement row to remain attached to the same displayed observer-history packet.

Use ledger exactness and locality provenance exactness. The present ledger is read through O, s, R, L, G, while H and C transport and replay only that route. Thus the locality-cell handoff is exhausted by the two sibling certificate channels named in P. □

Theorem 578.11 (Conscious observer-state GAP-ledger non-escape)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the Gap *acknowledgement row G is attached only to the present ledger L at the current state s read from O. A consumer may read acknowledged commitments only by first routing through O, s, R, L and then through the displayed transport, continuation, provenance, and naming rows H, C, P, N.*

The packet cannot export a detached self, a global universe, an external clock, a spacetime point, or a remote record not present in L or acknowledged by G.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap ledger nonescape`

Proof. Project the carrier and read O before any acknowledgement. The current row s is selected inside that observer-history surface, and R records the recognition row attached to the same present state.

The ledger step is then local. The row L supplies the present records, and G names only commitments attached to s whose witness rows are not present in that ledger.

Transport through H and consumer access through C preserve this order. With P and N binding provenance and local naming, no coordinate remains for a detached self, global universe, external clock, spacetime point, or remote record outside the ledger-and-gap pair. □

Theorem 578.12 (Conscious observer-state route locality)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every public route to the current observer-state surface is one of the displayed Cont *routes in* C*. Such a route reads* O*, then* s*, then the local rows* R, L, G*, and then the stabilizing rows* H, P, N*; it does not bypass the present-state ledger.*

Consequently route locality excludes detached self rows, shared-universe rows, external-clock rows, spacetime-point rows, and all-observer synchronization rows.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier route locality`

Proof. Start with the displayed route row C. By the carrier definition, C is the public continuation surface for reading O, then s, then R, L, G.

Componentwise transport through H preserves those route endpoints. A transported consumer still reaches the same observer-history row, current-state row, recognition row, observation ledger, and acknowledgement row rather than a detached host object.

The package and naming rows P, N bind the route to the local certificate. Because no coordinate in S stores a shared universe, external time scalar, manifold point, or all-observer synchronization map, no displayed continuation can route to one. □

Theorem 578.13 (Conscious observer-state dependency handoff)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the dependency handoff to observer-history and observer-locality consumers is exactly the pair of local reads recorded by O, s, R, L, G *and routed through* C*. Componentwise transport in* H *preserves the observer-history anchor, current-state row, self-recognition row, observation ledger, and* Gap *acknowledgement as one packet.*

The handoff supplies no global observer, no global universe, no external clock, no spacetime point, and no remote record outside the finite current-state ledger.

Proof. Project the packet and read the dependency anchors from O and s. The self-recognition row R and ledger row L are attached to that same present state, with G recording only the named gaps at that state.

Use H for componentwise transport. The observer-history anchor, current-state row, self-recognition row, ledger, and gap acknowledgement move together, so the dependency handoff cannot split off a free-standing self or global frame.

Replay the handoff through C and record provenance through P, N. The non-escape boundary is then the carrier inventory: detached self, global universe, external clock, spacetime point, and remote record rows are absent from S. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier dependency handoff`

Theorem 578.14 (Conscious observer-state locality-cell readback)**.** *For an accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every accepted present-state packet reads its current state through the ObserverLocalityCell$^{\uparrow}$ *row named by the provenance coordinate P. After $O, s, R, L, G, H, C, P, N$ have been displayed, the current-state read is the locality-cell readback already certified by the same local route; it is not a free read from a global observer, shared universe, external clock, manifold point, host-state coordinate, or remote record.*

Proof. Project the carrier of Definition 578.1. The current-state row s is read only after the observer-history row O, and the provenance row P names the sibling observer-history and locality-cell certificates used by that read.

Use the route-locality result of Definition 578.12. The displayed continuation row C is the only public route to O, s, R, L, G, while H preserves the route endpoints componentwise.

The locality-cell reference in the provenance row therefore supplies the readback boundary. Since S has no coordinate for a global observer, shared universe, external clock, manifold point, host-state coordinate, or remote record, the accepted packet reads the current state only through the displayed locality-cell route. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier locality cell readback`

Theorem 578.15 (Conscious observer-state present ledger boundary)**.** *For an accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

a present-state consumer may read only the present ledger row L and the Gap *acknowledgement row G after it has read O and s. The consumer may use R, H, C, P, N only to recognize, transport, route, package, and name that same local surface; it cannot export a global universe, external clock, remote record, detached self, or manifold point.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier present ledger boundary`

Proof. Start with ledger exactness from Definition 578.7. The consumer reaches the present observation surface by reading O, then s, then R, L, G.

Apply GAP-ledger non-escape from Definition 578.11. The acknowledgement row G remains attached to commitments at the current state whose witness rows are absent from L.

Route locality from Definition 578.12 supplies the consumer boundary through C, with H, P, N preserving, packaging, and naming the same route. Because the carrier has no global-universe, external-clock, remote-record, detached-self, or manifold-point coordinate, the present ledger boundary is exactly L and G after O and s. □

Theorem 578.16 (Conscious observer-state GAP locality exhaustion)**.** *For an accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every acknowledged commitment in G remains attached to the same current state s and the same locality-cell route named by P and replayed by C. Any consumer that bypasses the present ledger row L or the acknowledgement row G is outside the displayed carrier surface.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap locality exhaustion`

Proof. Use GAP-ledger non-escape from Definition 578.11. It binds every acknowledged commitment in G to the current state s and to witness rows absent from the present ledger L.

Use dependency handoff from Definition 578.13 together with the locality-cell readback of Definition 578.14. The observer-history anchor, current-state row, ledger row, acknowledgement row, and locality-cell route remain one packet under H, C, P, N.

Finally apply the present ledger boundary of Definition 578.15. A consumer route that does not pass through L and G has no displayed coordinate in S, so it is not a consumer of the accepted carrier. □

Theorem 578.17 (Conscious observer-state present route exhaustion)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every public route to the present observer-state surface factors through the displayed rows $O, s, R, L, G, H, C, P, N$*. In particular, a route first reads the observer-history row* O*, then the current-state row* s*, then the self-recognition row* R*, the present ledger row* L*, and the* Gap *acknowledgement row* G*, and only then uses componentwise transport, continuation replay, package provenance, and local naming.*

No public route bypasses the present-state ledger or exits to a global observer, shared universe, external time scalar, spacetime manifold, host psychological-content row, or remote observation row.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier present route exhaustion`

Proof. Start from the carrier of Definition 578.1. Its visible inventory is exactly $O, s, R, L, G, H, C, P, N$, and Definition 578.2 makes that inventory the local naming surface.

Use ledger exactness from Definition 578.7 and GAP non-escape from Definition 578.11. These results force the present route to pass through L before any acknowledged commitment in G can be read.

Use route locality from Definition 578.12 and the dependency handoff of Definition 578.13. The displayed continuation row C replays the same present-state dependency path, while H, P, N transport, package, and name that path without adding a global or host-level coordinate. □

Theorem 578.18 (Conscious observer-state GAP consumer exactness)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

a Gap*-facing consumer can read only commitments attached to the current state* s *whose witness rows are absent from the present ledger* L*. The consumer reaches those commitments through* O, s, R, L, G *and the stabilizing rows* H, C, P, N*, and it reads no additional observer, universe, time, manifold, host-subject, or remote-record surface.*

Proof. Read the present state through O and s. The carrier and NameCert obligations attach R, L, G to that same state, so the consumer has no independent Gap source.

Apply ledger exactness and GAP-ledger non-escape. The ledger row L supplies the available witness rows at s, and G names only commitments whose witnesses are absent from that ledger.

Finally use route locality and dependency handoff. The continuation row C makes the consumer route explicit, while H, P, N preserve, package, and name the same route. Since the carrier has no coordinate for a global observer, shared universe, external time, spacetime manifold, host

psychological content, or remote record, the Gap consumer is exact for the current-state ledger boundary. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap consumer exactness`

Theorem 578.19 (Conscious observer-state current ledger handoff)**.** *For every accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

each present-state consumer handoff reads the rows in the order $O, s, R, L, G, H, C, P, N$*. The handoff then passes only the present ledger row* L *and the* Gap *acknowledgement row* G*, transported and named through* H, C, P, N*, to observer-history and locality consumers.*

No detached observer, external clock, spacetime point, host content row, or remote record is handed across this boundary.

Proof. Start with the carrier of Definition 578.1 and the NameCert obligations of Definition 578.2. They display the ordered packet $O, s, R, L, G, H, C, P, N$ and require the recognition, ledger, and acknowledgement rows to be read after the observer-history row and current state row.

Apply ledger exactness from Definition 578.7 together with GAP-ledger non-escape from Definition 578.11. These theorems make L the visible present ledger at s and make G only an acknowledgement of commitments whose witness rows are absent from that ledger.

Finally use route locality, dependency handoff, and GAP consumer exactness from Definition 578.12, Definition 578.13, and Definition 578.18. They keep the consumer-facing handoff inside H, C, P, N and exclude every route not carried by the displayed current-state packet. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier current ledger handoff`

Theorem 578.20 (Conscious observer-state local GAP coverage)**.** *For every accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

each acknowledged commitment in G *is either witnessed by the visible present ledger* L *at the current state* s *or remains explicitly named as a local* Gap *at* s*. The coverage adds no remote record, global universe row, external time coordinate, or host-level observer surface.*

Proof. Read the carrier and the NameCert obligations first. The only available commitment surface is the displayed current-state packet $O, s, R, L, G, H, C, P, N$.

Use ledger exactness to identify the visible witness rows at s with L. Then apply GAP-ledger non-escape and GAP consumer exactness. These results say that every acknowledged commitment read through G is scoped by the same current state and can be consumed only as a local absence from the present ledger.

Route locality and dependency handoff close the coverage. Since every route is replayed through C and packaged by P, N with componentwise transport in H, no global universe, remote record, or host-level observer row can appear as a second coverage source. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier local gap coverage`

Theorem 578.21 (Conscious observer-state route package exactness)**.** *For every accepted* ConsciousObserverState$^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the rows H, C, P, N package exactly the same current-state route from O through s, R, L, G. They may transport, replay, prove provenance for, and name that route, but they cannot add a detached self, external clock, spacetime point, all-observer synchronization row, host content row, or remote record.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier route package exactness`

Proof. The carrier and NameCert obligations determine the route inventory. The rows H, C, P, N are displayed only as transport, continuation replay, package provenance, and local naming for the same O, s, R, L, G packet.

Apply route locality and dependency handoff. They identify C as the current-state continuation route and ensure that H, P, N preserve, package, and name the route rather than widening it.

Ledger exactness, GAP non-escape, and GAP consumer exactness show that the only observer-facing data on the route are the present ledger and local acknowledgement at s. Therefore the route package is exact and contains none of the excluded detached, external, global, host, or remote rows. □

Theorem 578.22 (Conscious observer-state primitive scope)**.** *For every accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the rows $O, s, R, L, G, H, C, P, N$ are scoped only by the BEDC primitive sources they display: O is a BHist *observer-history row, s is a current-state row inside that history, R is the self-recognition row read at s, L is the present ledger row, G is the* Gap *acknowledgement row, H is componentwise* hsame *transport, C is displayed* Cont *routing, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$ *row.*

Proof. Use the carrier of Definition 578.1. Its coordinates already name the observer-history, current-state, recognition, ledger, acknowledgement, transport, route, package, and naming rows.

Apply the NameCert obligations of Definition 578.2. They read R, L, G only after O and s, and they route all stability through H, C, P, N.

Ledger exactness in Definition 578.7 and GAP non-escape in Definition 578.11 bind the observational and acknowledgement rows to the same present state. Route locality and present route exhaustion from Definition 578.12 and Definition 578.17 then exclude any primitive source outside BHist, Gap, hsame, Cont, Pkg, and $\mathsf{NameCert}$. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier primitive scope`

Theorem 578.23 (Conscious observer-state GAP scope binding)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

the Gap *row G is scoped by the present ledger L and the current state s alone. A public read of G must first expose O, s, R, L and then replay the displayed transport, continuation, package, and naming rows H, C, P, N.*

Proof. Start with GAP-ledger non-escape from Definition 578.11. It attaches G to commitments at the current state s whose witness rows are absent from L.

Use ledger exactness from Definition 578.7. The present observational surface is L together with G, both read after the observer-history row O and the self-recognition row R attached to s.

Finally apply route locality and present route exhaustion. The only public path to G is through the displayed continuation row C with componentwise transport H, package provenance P, and local naming N; no global observer, shared universe, remote record, or host-level state can widen the Gap scope. □

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier gap scope binding`

Theorem 578.24 (Conscious observer-state route-scope package)**.** *For an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

every public route to the observer-state surface factors through the displayed Cont *routes in* C *and the package provenance row* P*. The route may expose* O, s, R, L, G *only through the componentwise* hsame *transport* H *and the local* $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier route scope package`

Proof. Apply route locality from Definition 578.12. It makes C the public continuation surface for O, then s, then R, L, G.

Use dependency handoff from Definition 578.13. The handoff keeps the observer-history anchor, current-state row, recognition row, ledger, and Gap acknowledgement in one packet under H.

Present route exhaustion in Definition 578.17 supplies the route coverage, and GAP consumer exactness in Definition 578.18 supplies the consumer boundary. The provenance row P and local name row N therefore bind every public route to the displayed package rather than to an external observer, universe, clock, manifold, host content, or remote-record surface. □

Theorem 578.25 (Conscious observer-state public export)**.** *The public* $\mathsf{ConsciousObserverState}^{\uparrow}$ *export is exactly the finite current-state observer packet*

$$S = (O, s, R, L, G, H, C, P, N),$$

together with the carrier, NameCert obligations, ledger exactness, GAP non-escape, route locality, dependency handoff, locality-cell readback, present ledger boundary, GAP locality exhaustion, route exhaustion, GAP consumer surface, current-ledger handoff, local GAP coverage, route package exactness, primitive scope, GAP scope binding, and route-scope package already established in this chapter.

The export supplies no detached self, global observer, shared universe, external clock, spacetime point, host psychological content, or remote record.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier public export`

Proof. First assemble the finite packet from the carrier and NameCert obligations. The rows $O, s, R, L, G, H, C, P, N$ are the only observer-history, current-state, recognition, ledger, acknowledgement, transport, route, provenance, and naming coordinates available to the export.

Next use ledger exactness and GAP non-escape. The ledger rows bind the present observational surface to s, while the GAP row records only local acknowledgement of absent witness rows and cannot export a remote record or external observer state.

The route-locality chain then supplies the public access path. Apply route locality, dependency handoff, locality-cell readback, present route exhaustion, GAP consumer exactness, current-ledger handoff, local GAP coverage, route package exactness, primitive scope, GAP scope binding, and

the route-scope package to keep every public read inside the same displayed continuation and package surface.

Therefore the public export is the assembled packet itself. Since no theorem in the chain introduces coordinates for a detached self, global observer, shared universe, external clock, spacetime point, host psychological content, or remote record, none of those objects is supplied by the export. □

Theorem 578.26 (Conscious observer-state bridge route)**.** *Every public* $\mathsf{ConsciousObserverState}^{\uparrow}$ *export supplied by Definition 578.25 bridges only through the finite packet*

$$S = (O, s, R, L, G, H, C, P, N).$$

The observer-history provenance is the row governed by chapter 477, and the locality-cell provenance is the row governed by chapter 513. The bridge route has no coordinate for a global observer, shared universe, external clock, spacetime point, host psychological-content row, or remote observation.

Lean4 ✓ `BEDC.Derived.ConsciousObserverStateUp.ConsciousObserverStateCarrier bridge route`

Proof. Start with the public export theorem. It identifies the exported surface with the displayed packet S and excludes any detached self, global observer, shared universe, external clock, spacetime point, host psychological content, or remote record.

Apply route package exactness from Definition 578.21. The public route is therefore mediated by the displayed Pkg provenance P, the local NameCert row N, and the componentwise hsame transport H, rather than by an ambient observer object.

Next use locality-cell handoff from Definition 578.10. This pins the locality component of the route to the local cell provenance described in chapter 513, while the observer-history row remains the one described in chapter 477.

Finally apply present route exhaustion from Definition 578.17. Since every present route is exhausted by the displayed continuation and package rows of S, the bridge route cannot introduce a global observer, shared universe, external clock, spacetime point, host psychological-content row, or remote observation. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConsciousObserverState}^{\uparrow}$ packet is generated from an observer-history row, a current-state row, a self-recognition row, an observation ledger, a Gap acknowledgement, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 578.1, Definition 578.2, Definition 578.6, Definition 578.7, Definition 578.8, Definition 578.9, Definition 578.10, Definition 578.11, Definition 578.12, Definition 578.13, Definition 578.14, Definition 578.15, Definition 578.16, Definition 578.17, Definition 578.18, Definition 578.19, Definition 578.20, Definition 578.21, Definition 578.25, and Definition 578.26 bind the finite current-state observer packet, GAP ledger, locality-cell readback, present ledger boundary, route locality, dependency handoff, route exhaustion, GAP consumer surface, current-ledger handoff, local GAP coverage, route package exactness, public packet export, and bridge route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConsciousObserverState}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a global observer, a shared universe, external time, manifold spacetime, host psychological content, checked Lean verification, or bridge checking.

Upgrade path: No further paper-side closure grade remains; Lean verification and bridge checking remain on the verification axis.

Chapter 579

A Concrete Naming Certificate for $\mathsf{BoundedSearchRefutation}^{\uparrow}$

The $\mathsf{BoundedSearchRefutation}^{\uparrow}$ surface packages a finite refutation search as BEDC naming data. It records the proposition being inspected, a displayed depth budget, the finite frontier reached by that budget, a refutation witness when one is found or a Gap row when the budget is exhausted, and the provenance rows through which a consumer may replay the bounded search. The packet is not a global proof-search oracle: every accepted read is bounded by the displayed depth row and remains inside BHist, hsame, Cont, Pkg, and NameCert.

It consumes the finite contrary-record surface of chapter 576 when a bounded frontier displays an explicit witness row: the bounded search packet supplies the depth and frontier ledger, while the witnessed-refutation packet supplies the contrary record and decision ledger consumed by the refutation branch.

Definition 579.1 (BoundedSearchRefutation carrier)**.** *A* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, K, F, R, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the proposition row being searched for a certificate-level refutation;*
2. *K is the finite depth budget row, including the displayed start depth, successor ledger, and final bounded depth inspected by the packet;*
3. *F is the search frontier row, listing exactly the finite proof states reached within the budget K and the local edges by which each frontier entry was generated;*
4. *R is the displayed refutation witness row, present only when the frontier contains a route from an assumed witness of A to $\bot$;*
5. *G is the* Gap *row, present exactly when the bounded frontier ends without a displayed refutation witness and recording that the packet makes no truth claim beyond the budget;*
6. *H stores componentwise* hsame *transports for proposition rows, depth rows, frontier rows, refutation-witness rows,* Gap *rows, provenance, and local naming rows;*
7. *C stores the displayed* Cont *routes for reading the proposition row, advancing the bounded frontier, replaying a displayed refutation witness, or exposing the* Gap *row;*

8. *P is the* Pkg *provenance over* BHist, msame, hsame, Cont, Gap, *bounded frontier rows, displayed refutation witnesses, and the surrounding* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$ *row.*

The carrier contains no host-level proof search, no unbounded recursion principle, no truth predicate for all naming certificates, no excluded-middle branch, and no witness selected outside the displayed frontier.

Lean4 (definition) `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationCarrier`
Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationTasteGate single carrier alignment`

Theorem 579.2 (BoundedSearchRefutation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$ *surface for the carrier of Definition 579.1 consists of five finite obligations:*

1. *proposition row: the accepted packet displays the proposition row A whose refutation is being searched;*

2. *finite depth budget: every frontier expansion is indexed by the displayed budget row K and stops at the displayed final bounded depth;*

3. *search frontier: the row F lists the complete finite frontier generated within K, including the displayed local edges used to reach each entry;*

4. *refutation or gap: the packet exposes either a displayed refutation witness row R or the* Gap *row G, and the two rows are read through the same bounded frontier surface;*

5. *provenance: all reads of A, K, F, R, G transport through the stored* hsame *row H, the displayed* Cont *routes C, the package row P, and the local naming row N.*

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationNameCertObligations`

Proof. Carrier admission is projection from Definition 579.1. The proposition row, depth budget, frontier, refutation witness, gap row, transports, continuations, provenance, and local name are all displayed coordinates of the same finite packet.

The depth and frontier obligations are finite because K supplies the only allowed expansion budget. Each frontier entry in F is reached by a displayed local edge before the final bounded depth is reached, and no consumer can ask for a row outside that finite ledger.

The refutation-or-gap obligation reads the same frontier surface in both cases. If R is present, the packet exports the displayed route to $\bot$; if G is present, the packet exports only the bounded absence of such a displayed route. The provenance rows H, C, P, N keep both reads inside the same NameCert surface. □

Theorem 579.3 (BoundedSearchRefutation depth soundness)**.** *For an accepted* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *carrier $B = (A, K, F, R, G, H, C, P, N)$, every refutation witness exported from R is generated by a finite path in the frontier row F whose length is bounded by the displayed depth budget K. A consumer that reads G receives only the statement that no refutation witness is displayed inside that same bounded frontier; it receives no claim about depths beyond K.*

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationDepthSoundness`

Proof. Start with a consumer read of the refutation side. The witness row R is not an independent proof object; by the NameCert obligations it is read through the bounded frontier row F and the budget row K.

The continuation routes C replay only the local frontier edges recorded in F. Since each edge is indexed by the successor ledger in K, the replayed route cannot exceed the displayed final bounded depth.

For a gap read, the same bounded ledger is inspected. The row G records the absence of a displayed witness within F at budget K, not the absence of all refutations and not a truth value for A. Thus depth soundness is exactly the finite replay discipline of the carrier. □

Theorem 579.4 (BoundedSearchRefutation obligation scope)**.** *For an accepted* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *carrier* $B = (A, K, F, R, G, H, C, P, N)$*, every accepted frontier replay is derived by induction over the displayed finite depth row* K*. The base case is the initial frontier displayed in* F*, and the step case is exactly the displayed local edge extension in* F *from a depth row already admitted by* K *to its successor row. Consequently the bounded-search obligation surface contains no replay step outside* K*, no frontier entry outside* F*, and no refutation or* Gap *read outside the carrier rows of Definition 579.1.*

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationObligationScope`

Proof. The carrier inventory in Definition 579.1 supplies the only rows used by the replay: the finite budget K, the frontier F, the refutation-or-gap rows R, G, and the transport, continuation, package, and naming rows H, C, P, N.

Use the obligations of Definition 579.2. They identify K as the finite depth budget and F as the complete displayed frontier generated within that budget. The induction base reads the initial frontier row already displayed in F.

For the induction step, assume a frontier entry has been admitted at a displayed row of K. The only possible successor read is one of the local edge extensions recorded in F and indexed by the successor ledger of K. This step adds no host search and no unrecorded proof state.

Apply Definition 579.3. A refutation witness read from R is therefore a bounded replay path through those same induction steps, while a Gap read from G is only the report that no witness is displayed by the resulting finite frontier. □

Theorem 579.5 (BoundedSearchRefutation non-escape)**.** *No accepted* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *packet exports a global proof-search oracle. Every consumer read factors through* $A, K, F, R, G, H, C, P, N$*, and the packet has no coordinate for unbounded search, excluded middle over certificate predicates, host truth, quotient proof equality, or witness selection outside the displayed frontier.*

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationNonEscape`

Proof. Inspect the carrier coordinates in Definition 579.1. They name a proposition row, a finite budget, a finite frontier, a displayed refutation witness or gap row, and the BEDC transport, continuation, package, and naming rows.

Apply Definition 579.3. A witness read is a bounded frontier replay, while a gap read is only a bounded frontier report. Neither read supplies data for a search at a greater depth.

The forbidden oracle shapes require coordinates absent from the packet: an unbounded recursion route, a truth classifier over all naming certificates, an excluded-middle branch, or a selector outside F. Since no such row exists, every consumer remains inside the displayed bounded-search certificate. □

Theorem 579.6 (BoundedSearchRefutation NameCert boundary)**.** *The* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *carrier realizes a finite refutation-search boundary for contradiction and refutation consumers. A consumer may use a displayed witness row* R *as a bounded certificate-level refutation of* A*, or it*

may use the Gap *row G as a bounded search report at depth K; in both cases the read is routed through the same frontier, transport, continuation, provenance, and local naming rows. The packet never upgrades a* Gap *row into a proof of A or into a proof that A has no refutation at greater depth.*

Proof. First assemble the finite surface from Definition 579.2. The proposition row, budget, frontier, refutation-or-gap row, and provenance rows form one NameCert packet.

If a consumer reads R, depth soundness gives a replayable frontier path to $\bot$ bounded by K. That is a permitted certificate-level refutation because the assumed witness of A is discharged by displayed Cont rows inside the carrier.

If a consumer reads G, non-escape gives the boundary. The row reports only that the displayed frontier at the displayed depth contains no refutation witness, and it carries no branch that decides A or all deeper searches.

Therefore the public boundary is a finite BEDC search certificate. It supports bounded refutation and bounded gap reporting, while refusing any global proof-search, truth-branch, or unbounded-depth conclusion. □

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationNameCertBoundary`

Theorem 579.7 (BoundedSearchRefutation public export)**.** *Every downstream consumer of an accepted* BoundedSearchRefutation$^{\uparrow}$ *packet receives exactly the displayed finite frontier depth K, the contrary-record witness row R when that row is present, the bounded* Gap *row G when the frontier has no displayed witness, and the non-escape ledger H, C, P, N. The public export is the finite surface of Definition 579.1, Definition 579.2, Definition 579.3, Definition 579.5, and Definition 579.6; it exports no unbounded search, host truth predicate, excluded-middle branch, quotient proof equality, or witness outside the displayed frontier.*

Proof. Read the carrier first. The only public rows available to a consumer are the proposition row A, depth budget K, frontier F, witness-or-gap rows R, G, and the structural ledger H, C, P, N.

Apply the NameCert obligations and depth soundness. These force the contrary-record witness, when present, to be read through the finite frontier generated inside K, and force the Gap row to report only bounded absence inside that same frontier.

Finally use non-escape and the NameCert boundary. Since neither theorem supplies any coordinate beyond $A, K, F, R, G, H, C, P, N$, the public export is exactly the bounded frontier, contrary-record or Gap row, and non-escape ledger stated above. □

Lean4 ✓ `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationPublicExport`

Theorem 579.8 (BoundedSearchRefutation bounded frontier bridge)**.** *Every bridge consumer of an accepted* BoundedSearchRefutation$^{\uparrow}$ *packet must first read the finite depth budget K, the bounded frontier F, the witness-or-gap rows R, G, and the structural rows H, C, P, N before handing the packet to any standard bounded-refutation interface. The bridge read is therefore the displayed route through the same bounded frontier exported by Definition 579.7; it adds no global proof search, no truth predicate over certificates, no excluded-middle branch, and no witness outside F.*

Proof. Start with the NameCert obligations of Definition 579.2. They identify K, F, R, G, H, C, P, N as the finite rows through which any accepted refutation or bounded Gap read must pass.

Use depth soundness and obligation scope, Definition 579.3 and Definition 579.4. These force every replayed witness route to remain inside the displayed depth budget and frontier, while a Gap read remains only a bounded frontier report.

Finally combine non-escape, the NameCert boundary, and the public export, Definition 579.5, Definition 579.6, and Definition 579.7. Since the public packet has no coordinate beyond $A, K, F, R, G, H, C, P, N$, a standard bounded-refutation bridge can consume exactly those rows and cannot import any unbounded or classical proof-search endpoint. □

Theorem 579.9 (BoundedSearchRefutation mature treatment)**.** *Every mature consumer of an accepted* $\mathsf{BoundedSearchRefutation}^{\uparrow}$ *packet reads the proposition row A, the depth budget K, the finite frontier F, the witness-or-gap rows R, G, the componentwise transport row H, the displayed replay routes C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$ *row N as one bounded-search treatment. The treatment closes exactly the displayed finite frontier discipline: it exports no global proof search, no witness outside F, no excluded-middle branch, and no claim about depths beyond K.*

Proof. Use the carrier and obligations, Definition 579.1 and Definition 579.2. They make $A, K, F, R, G, H, C, P, N$ the complete public inventory of the packet.

Depth soundness and obligation scope, Definition 579.3 and Definition 579.4, force every accepted replay to remain inside the displayed finite budget and frontier.

The public export and bounded-frontier bridge, Definition 579.7 and Definition 579.8, are therefore already the mature treatment: mature consumers can only reuse the same finite rows and cannot attach an unbounded search, deeper-depth assertion, host truth predicate, or off-frontier selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedSearchRefutation}^{\uparrow}$ packet is generated from a proposition row, a finite depth budget, a bounded search frontier, a displayed refutation witness or Gap row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 579.1, Definition 579.2, Definition 579.3, Definition 579.4, Definition 579.5, Definition 579.6, Definition 579.7, Definition 579.8, and Definition 579.9 bind the finite bounded-search carrier, obligation scope, depth discipline, non-escape boundary, NameCert consumer surface, public export, bounded-frontier bridge, and mature consumer treatment.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedSearchRefutation}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BoundedSearchRefutationUp.BoundedSearchRefutationNameCertObligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close global proof search, excluded middle, host truth, quotient proof equality, or absence of refutations beyond the displayed budget.

Upgrade path: No further theory-axis grade is claimed; any verification-axis work must preserve the same bounded refutation and bounded Gap rows without adding global proof search or unbounded-depth claims.

Chapter 580

A Concrete Naming Certificate for $\mathsf{ZetaContinuationWitness}^{\uparrow}$

The $\mathsf{ZetaContinuationWitness}^{\uparrow}$ interface packages the zeta-continuation application of chapter 9342 as a finite witness certificate. It is the BEDC-native handoff object between $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, and the RH-facing critical-strip chapters. The packet records the eta-continuation route, the pole ledger at $s = 1$, the functional-equation route, the trivial-zero ledger, and the Gamma-horizon boundary without importing the RH predicate or a global zero-location conclusion.

Definition 580.1 (ZetaContinuationWitness carrier)**.** *A* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N)$$

with the following displayed coordinates:

1. *B is the source* $\mathsf{ZetaBasic}^{\uparrow}$ *row on the half-plane $\Re(s) > 1$;*
2. *E is the Dirichlet eta route, including the alternating-series row, the overlap row with B, and the displayed denominator boundary for $1 - 2^{1-s}$;*
3. *A is the analytic-continuation row from the eta route to the punctured positive half-plane;*
4. *P is the pole ledger at $s = 1$, recording the simple-pole row and residue-one readback;*
5. *F is the functional-equation route, including the exponential, sine, reflected-zeta, and Gamma-factor rows;*
6. *T is the trivial-zero ledger for negative even inputs, read through the sine-factor row and the nonzero remaining factors;*
7. *G is the Gamma-horizon boundary row, recording that the Gamma factor is consumed through its displayed interface and not through a closed Gamma chapter;*
8. *H records componentwise* hsame *transports for B, E, A, P, F, T, G;*
9. *C records the finite* Cont *routes by which consumers read the source zeta row, eta route, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, package provenance, and local naming row;*
10. *R is the* Pkg *provenance over* $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationWitness}^{\uparrow}}$ *row.*

The carrier contains no proof of RH, no non-trivial-zero locator, no selected branch outside the displayed continuation routes, and no hidden completion of the Gamma horizon.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.TasteGate.ZetaContinuationWitnessTasteGate single carrier alignment`

Theorem 580.2 (ZetaContinuationWitness NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationWitness}^{\uparrow}}$ *surface for Definition 580.1 consists of five finite rows:*

1. *carrier admission for the displayed packet W;*
2. *classifier stability for the* $\mathsf{ZetaBasic}^{\uparrow}$ *source row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, and Gamma-horizon row;*
3. *analytic-continuation exactness, stating that all exported* $\mathsf{ZetaCont}^{\uparrow}$ *reads factor through B, E, A, F, H, C, R, N;*
4. *pole and trivial-zero ledger exactness for P and T;*
5. *Gamma-horizon non-escape, stating that every Gamma-dependent read reaches G unless the read is already one of the displayed functional-equation factors.*

Proof. Carrier admission is projection from the finite tuple in Definition 580.1. The source row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transport routes, continuation routes, package row, and local naming row are explicit coordinates of the same BHist packet.

Classifier stability is componentwise. The $\mathsf{ZetaBasic}^{\uparrow}$ source row is transported through H, the eta and functional-equation routes are transported through their displayed overlap and factor rows, and the pole and trivial-zero ledgers are transported as ledger coordinates rather than as ambient analytic claims.

Analytic-continuation exactness follows because every exported $\mathsf{ZetaCont}^{\uparrow}$ read is routed through C after the source, eta, and functional-equation coordinates have been exposed. There is no additional continuation coordinate in W, so a consumer cannot bypass B, E, A, F, H, C, R, N to obtain a separate zeta continuation.

The pole and trivial-zero clauses are exact ledger reads. The pole row P records the simple-pole and residue-one slots at $s = 1$, while T records the negative-even zero slots produced by the sine factor together with the nonzero-factor side conditions.

Finally, the Gamma boundary is a non-escape row. If a consumer asks for a Gamma fact beyond the displayed functional-equation factor, the request reaches G and remains a boundary entry, not a closed Gamma certificate and not a hidden route to RH. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket namecert obligation surface`

Definition 580.3 (ZetaContinuationWitness source lock)**.** *For an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

the source lock is the finite obligation package that requires every continuation, pole, functional-equation, trivial-zero, Gamma-boundary, and critical-strip route to expose the source row B, eta route E, analytic-continuation row A, pole ledger P, functional-equation route F, trivial-zero ledger T, Gamma-horizon boundary G, componentwise transports H, displayed routes C, provenance R, and local name N before any downstream zeta-zero or critical-strip consumer reads the packet.

Lean4 (definition) `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessSourceLock`

Theorem 580.4 (ZetaContinuationWitness source lock coverage)**.** *Every continuation read from an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet first exposes the source lock of Definition 580.3. In particular, the read displays B, E, A, F and the pole, trivial-zero, and Gamma ledgers P, T, G before it can be consumed as a zeta-continuation, zeta-zero, or critical-strip input.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket source lock coverage`

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$ and apply Definition 580.2. The obligation surface lists carrier admission, classifier stability, analytic-continuation exactness, pole and trivial-zero exactness, and Gamma-horizon non-escape.

Use route-carrier admission through Definition 580.41. The exported continuation route is a read from the same displayed packet, so it cannot reach a consumer before the source, eta, analytic-continuation, and functional-equation rows have been displayed.

The exceptional ledgers are part of the same lock. Pole reads expose P, trivial-zero reads expose T, and Gamma-dependent reads expose G unless they are already displayed in F. Thus every continuation read is covered by the finite source lock. □

Theorem 580.5 (ZetaContinuationWitness pole-Gamma boundary)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, pole reads remain in the pole ledger P and Gamma-dependent reads remain in the Gamma-horizon boundary G unless the requested Gamma factor is already displayed inside the functional-equation route F. The packet exports no separate pole completion, closed Gamma certificate, or hidden analytic branch outside P, F, G, H, C, R, N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket pole gamma boundary`

Proof. Begin with the NameCert obligation surface. It treats P as the pole ledger and G as the Gamma-horizon non-escape row, while F is the only displayed functional-equation route that may already contain a Gamma factor.

Apply route-carrier admission. Any pole or Gamma request is routed through the same carrier and authenticated by H, C, R, N.

Now separate the two boundary cases. Pole data is read from P. Gamma data is read from G unless the read is one of the displayed factors in F. Since the carrier has no additional pole or Gamma coordinate, no further completion or hidden branch is exported. □

Theorem 580.6 (ZetaContinuationWitness critical-strip route separation)**.** *Critical-strip consumers of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receive the displayed route rows $B, E, A, P, F, T, G, H, C, R,$ and the source lock of Definition 580.3. They do not receive an RH predicate, a non-trivial-zero locator, a global zero classifier, or a completed Gamma certificate as input data.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip route separation`

Proof. Use Definition 580.28. It states that the critical-strip consumer receives only the displayed finite witness rows and cannot read RH predicates, non-trivial-zero locators, global zero classifiers, or completed Gamma certificates as packet input.

Combine this refusal with source-lock coverage. By Definition 580.4, every continuation read has already exposed B, E, A, F together with P, T, G and the authentication rows H, C, R, N.

Therefore the critical-strip route is separated from RH-facing predicates. The consumer can use the displayed finite route rows, but no excluded predicate or completed Gamma object is present in the carrier. □

Theorem 580.7 (ZetaContinuationWitness no-RH non-escape)**.** *An accepted* ZetaContinuationWitness$^{\uparrow}$ *packet exports no RH proof, no non-trivial-zero locator, and no completed Gamma certificate. Every consumer route remains inside the source lock $B, E, A, P, F, T, G, H, C, R, N$ and the obligation rows of Definition 580.2.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket no rh nonescape`

Proof. Start from the critical-strip input refusal and Gamma-horizon non-escape rows. They exclude RH predicates, non-trivial-zero locators, global zero classifiers, and closed Gamma certificates from the packet inventory.

Use pole-Gamma boundary from Definition 580.5. Pole and Gamma reads remain in P and G, except for Gamma factors already displayed in F, and none of those rows is an RH proof or a completed Gamma certificate.

Finally apply source-lock coverage. Every consumer route is confined to $B, E, A, P, F, T, G, H, C, R, N$, so the excluded RH-facing objects have no route out of the accepted witness. □

Theorem 580.8 (ZetaContinuationWitness dependency ledger)**.** *Every accepted* ZetaContinuationWitness$^{\uparrow}$ *packet routes through the displayed* ZetaBasic$^{\uparrow}$*,* ZetaCont$^{\uparrow}$*,* AnaCont$^{\uparrow}$*,* Holomorphic$^{\uparrow}$*,* Complex$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before any zeta-zero or critical-strip consumer can read it. No consumer route can read the packet as a zeta-zero input until those dependency rows have been exposed as coordinates of the same finite carrier.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket dependency ledger`

Proof. Project the accepted carrier as $W = (B, E, A, P, F, T, G, H, C, R, N)$. The carrier definition lists the source zeta row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transport rows, continuation routes, package provenance, and local name as the complete witness packet.

Use the NameCert obligation surface. Carrier admission and classifier stability bind the ZetaBasic$^{\uparrow}$, ZetaCont$^{\uparrow}$, AnaCont$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert dependencies to those displayed coordinates.

Coordinate exhaustion then gives the consumer boundary. A zeta-zero or critical-strip consumer can only read through the continuation routes in C after the same packet rows have been displayed, so no dependency outside the finite ledger is available. □

Theorem 580.9 (ZetaContinuationWitness zero-facing source totality)**.** *Every* ZetaZeros$^{\uparrow}$ *or critical-strip consumer that reads an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must first expose the source row B, eta route E, analytic-continuation row A, functional-equation route F, transport row H, continuation row C, provenance row R, and local name N. The exceptional rows P, T, G may also be consumed by their own ledger routes, but no zero-facing read can treat the packet as available before B, E, A, F, H, C, R, N have been displayed as coordinates of the same carrier.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero facing source totality`

Proof. Project the accepted packet as $W = (B, E, A, P, F, T, G, H, C, R, N)$. The dependency ledger identifies the ZetaBasic$^{\uparrow}$, ZetaCont$^{\uparrow}$, AnaCont$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows that a zero-facing consumer must meet before reading the packet.

Apply source-lock coverage. It fixes B, E, A, F as the source, eta, analytic-continuation, and functional-equation face of the continuation route, while H, C, R, N authenticate the transport, continuation, provenance, and local-name rows of that same route.

The pole, trivial-zero, and Gamma rows P, T, G remain visible ledger coordinates, but they do not replace the source face. Thus every zero-facing consumer reaches the packet only after the displayed source and authentication rows have been exposed. □

Theorem 580.10 (ZetaContinuationWitness Gamma functional boundary totality)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, every Gamma-dependent consumer read is exhausted by one of two displayed routes: it either uses the Gamma factor already present in the functional-equation route F, or it stops at the Gamma-horizon boundary row G. No such read exports a completed Gamma certificate, a hidden analytic branch, or a Gamma route outside F, G, H, C, R, N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma functional boundary totality`

Proof. Begin with the NameCert obligation surface. The surface lists the functional-equation route F and Gamma-horizon boundary G as explicit packet rows, and it treats Gamma-horizon non-escape as part of the accepted certificate.

Apply the pole-Gamma boundary theorem. A Gamma request that is already one of the displayed functional-equation factors is read through F; every other Gamma-dependent request reaches G and remains a boundary row.

Finally use coordinate exhaustion for W. Since the packet contains no Gamma coordinate other than the displayed functional-equation route and boundary row, and since H, C, R, N only transport and authenticate those coordinates, no completed Gamma certificate or hidden branch can be exported. □

Theorem 580.11 (ZetaContinuationWitness critical-strip consumer non-escape)**.** *Every critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receives only the finite packet rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The consumer receives no RH predicate, no non-trivial-zero locator, no global zero classifier, and no completed Gamma certificate as packet input.

Proof. Project the carrier row. The accepted witness is the finite tuple with coordinates $B, E, A, P, F, T, G, H, C, R$, so every packet read by a critical-strip consumer is a read from that displayed list.

Apply critical-strip input refusal and route separation. These rows exclude RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from the packet inventory while keeping the consumer route inside the displayed source lock.

The Gamma functional boundary closes the remaining escape route. Gamma-dependent reads either use F or stop at G, so the critical-strip consumer obtains no completed Gamma object and no hidden route to an RH-facing or zero-location predicate. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip consumer nonescape`

Theorem 580.12 (ZetaContinuationWitness public root export)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, the scoped rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

export a public root witness only through the displayed ZetaBasic$^{\uparrow}$, ZetaCont$^{\uparrow}$, AnaCont$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. No public root consumer receives a witness row outside that carrier, dependency ledger, public export packet, and public export closure surface.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public root export

Proof. Project the carrier of Definition 580.1. The packet rows $B, E, A, P, F, T, G, H, C, R, N$ are the complete displayed witness carrier, so every public root read starts from those coordinates.

Apply Definition 580.2, Definition 580.4, and Definition 580.8. These bind the source, continuation, analytic, holomorphic, complex, history, transport, package, and naming data to the same finite packet before any public consumer can read it.

Finally use Definition 580.78 and Definition 580.86. The public export and its closure surface factor through the displayed packet and leave no route to an off-carrier root witness. □

Theorem 580.13 (ZetaContinuationWitness critical-strip root handoff)**.** *Every critical-strip consumer receives the public root witness of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet only after the source lock, pole ledger, trivial-zero ledger, and Gamma boundary have been displayed inside the same packet. The handoff does not expose an RH predicate, a non-trivial-zero locator, a global zero classifier, or a completed Gamma certificate.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip root handoff

Proof. Begin with Definition 580.4. It requires the source row B, eta row E, analytic-continuation row A, functional-equation row F, and the ledger rows P, T, G to be visible before a continuation, zero-facing, or critical-strip read can consume the packet.

Use Definition 580.5 for the pole and Gamma side, and use the carrier row to keep the trivial-zero ledger T as a displayed coordinate of $W = (B, E, A, P, F, T, G, H, C, R, N)$.

Close with Definition 580.6 and Definition 580.7. The critical-strip route remains a read from the displayed public root packet, so the excluded RH, zero-location, global-classifier, and completed-Gamma data have no input position. □

Theorem 580.14 (ZetaContinuationWitness Gamma boundary public export)**.** *The public export of Gamma-facing data from an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet is exhausted by the rows* P, F, G, H, C, R, N*. This Gamma-facing export does not close a Gamma chapter, an RH predicate, a non-trivial-zero locator, or any hidden analytic branch.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessGammaBoundaryPublicExport
↪ variant BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma boundary public export

Proof. Start from Definition 580.5. A Gamma-facing read either uses the Gamma factor already displayed in the functional-equation row F or stops at the Gamma boundary row G, with P recording the adjacent pole ledger.

Apply Definition 580.2 and Definition 580.88. The transport, continuation, provenance, and local naming rows H, C, R, N authenticate the same public ledger, and the ledger has no Gamma coordinate beyond P, F, G, H, C, R, N.

Finally use Definition 580.86 together with Definition 580.7. The public export remains a finite packet boundary, not a Gamma chapter closure, RH assertion, non-trivial-zero locator, or hidden analytic branch. □

Theorem 580.15 (ZetaContinuationWitness source obligation)**.** *Every root-facing consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet must first project the displayed packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N)$$

and read the source row B, eta route E, analytic-continuation row A, functional-equation row F, transport row H, continuation row C, package provenance R, and local name N through the existing NameCert obligation surface. The pole, trivial-zero, and Gamma rows P, T, G remain ledger coordinates of the same packet; they do not supply an RH predicate, a non-trivial-zero locator, or a completed Gamma certificate.

Proof. Project $W = (B, E, A, P, F, T, G, H, C, R, N)$. The carrier exposes each row as a coordinate before any root-facing read is available.

Apply Definition 580.2 and Definition 580.4. The obligation surface binds carrier admission, analytic-continuation exactness, and ledger exactness, while source-lock coverage requires B, E, A, F and the authentication rows H, C, R, N to be visible first.

Use Definition 580.28 for the boundary. Since the packet contains no RH predicate, non-trivial-zero locator, global zero classifier, or completed Gamma certificate, the source obligation is exactly the displayed finite source lock. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket source boundary`

Theorem 580.16 (ZetaContinuationWitness analytic readback obligation)**.** *For an accepted* ZetaContinuationWitn *packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, every root-facing analytic readback factors through B, E, A, F, H, C, R, N and the pole-Gamma boundary rows P, G before it can be consumed by a zeta-zero or critical-strip route. The readback may use the functional-equation row F and the Gamma boundary row G, but it cannot import an off-packet analytic branch or a completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket analytic readback obligation`

Proof. Begin with the source obligation above. It places B, E, A, F, H, C, R, N on the route before any root-facing readback is consumed.

Apply Definition 580.5. Pole and Gamma-dependent reads are confined to the displayed pole ledger P, functional-equation row F, and Gamma boundary row G.

Finally use the dependency ledger of Definition 580.8. The $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert dependencies are already bound to W, leaving no coordinate for an off-packet branch or completed Gamma certificate. □

Theorem 580.17 (ZetaContinuationWitness zero ledger non-escape)**.** *An accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet exports no zero-location witness beyond the displayed trivial-zero ledger T and no RH-facing predicate beyond the source, analytic, pole, functional-equation, Gamma, transport, continuation, provenance, and naming rows of $W = (B, E, A, P, F, T, G, H, C, R, N)$. A root-facing consumer cannot read the packet as a completed zero ledger, a global zero classifier, or an RH proof.*

Proof. Project the packet and read the zero-facing coordinates. The only zero ledger coordinate is T, and it appears beside the source, analytic, pole, functional-equation, Gamma, transport, continuation, provenance, and naming rows.

Apply Definition 580.28 and Definition 580.11. These boundary rows exclude RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from any critical-strip or root-facing consumer.

Use the pole-Gamma boundary and dependency ledger to exhaust the remaining routes. Since P, F, G, H, C, R, N only authenticate the displayed packet and do not add a zero-location coordinate, the trivial-zero ledger cannot escape as a completed zero classifier. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero ledger nonescape`

Theorem 580.18 (ZetaContinuationWitness formal-target ledger)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

the source totality row, Gamma functional boundary totality row, and critical-strip consumer non-escape row jointly present the finite formal target surface for ZetaContinuationWitness$^{\uparrow}$*. A consumer receives only* $B, E, A, P, F, T, G, H, C, R, N$ *with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. RH predicates, non-trivial-zero locators, global zero classifiers, hidden analytic branches, and completed Gamma certificates remain outside the packet.*

Proof. Use the source obligation theorem to bind root-facing reads to the displayed source row, eta route, analytic-continuation row, functional-equation row, transport row, continuation row, package provenance, and local name. This supplies the source-totality side of the formal target surface.

Use the analytic readback obligation for the Gamma functional boundary. It routes analytic access through B, E, A, F, H, C, R, N and the pole-Gamma rows P, G, so a consumer cannot import an off-packet branch or a completed Gamma certificate.

Use zero ledger non-escape for the critical-strip boundary. It leaves the displayed trivial-zero ledger row T inside W and refuses RH predicates, non-trivial-zero locators, and global zero classifiers.

The chapter-level dependency ledger already records the packet dependencies through the Lean target $BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket_dependency_ledger$. Combining these three paper rows with that dependency ledger gives one finite formal-target surface over $B, E, A, P, F, T, G, H, C, R, N$ and no additional analytic or zero-facing coordinate. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root zero facing formal surface`

Theorem 580.19 (ZetaContinuationWitness root package)**.** *The root-facing obligation package for an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet is the finite surface consisting of the source obligation of Definition 580.15, the analytic readback obligation of Definition 580.16, and the zero ledger non-escape boundary of Definition 580.17. It exposes only* $W = (B, E, A, P, F, T, G, H, C, R, N)$ *and keeps RH predicates, non-trivial-zero locators, global zero classifiers, off-packet analytic branches, and completed Gamma certificates outside the packet.*

Proof. First assemble the source face. The source obligation projects W and binds the zero-facing consumer to the displayed source lock and NameCert obligation surface.

Next attach analytic readback. The analytic obligation routes every root-facing readback through the source rows, functional-equation row, pole-Gamma boundary, and dependency ledger.

Finally apply zero ledger non-escape. The zero-facing ledger remains the displayed T row inside W, while RH predicates, zero locators, global zero classifiers, off-packet analytic branches, and completed Gamma certificates remain outside the packet. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root package`

Theorem 580.20 (ZetaContinuationWitness root zero-ledger totality)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

every root-facing zero read is either the displayed trivial-zero ledger row T transported through H, C, R, N, or is refused as an RH-facing, non-trivial-zero, or global-zero-classifier request. No root-facing consumer can obtain a zero read outside the source, analytic readback, zero ledger non-escape, and root package surfaces already bound to W.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root zero ledger totality`

Proof. Begin with the source obligation of Definition 580.15. It fixes the packet coordinates $B, E, A, P, F, T, G, H, C, R, N$ before any root-facing consumer can read a zero-facing row.

Next use the analytic readback obligation of Definition 580.16. Any analytic access to a root-facing readback must pass through the displayed source, functional-equation, pole-Gamma, transport, continuation, provenance, and naming rows, so it cannot import an off-packet zero classifier.

Apply Definition 580.17 to the zero-facing coordinate. The only zero ledger row exported by the packet is T, and RH predicates, non-trivial-zero locators, and global zero classifiers remain outside the accepted surface.

Finally invoke Definition 580.19. The root package is exactly the source obligation, analytic readback obligation, and zero ledger non-escape boundary. Hence each root-facing zero read is the transported T row through H, C, R, N, or else it is refused at the displayed boundary. □

Theorem 580.21 (ZetaContinuationWitness zero-consumer source readiness)**.** *For an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

every zero-facing consumer first exposes the source lock B, E, A, F and the authenticated rows H, C, R, N in the same BHist*-anchored packet. The consumer may then read the pole, trivial-zero, and Gamma ledger rows P, T, G, but it receives no RH predicate, non-trivial-zero locator, global zero classifier, or completed Gamma certificate as a source input.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero consumer source readiness`

Proof. Project the accepted carrier through BHist and the displayed packet W. The carrier rows fix the source zeta row, eta route, analytic-continuation row, functional-equation row, transports, continuation routes, provenance, and local name before any zero-facing read can occur.

Apply Definition 580.2 and Definition 580.4. These two rows force B, E, A, F and the authentication rows H, C, R, N to be visible as the source-facing entrance of the same packet.

Use Definition 580.15 and Definition 580.20. The child-file obligation package already records that the zero-facing read is either the transported T row or is refused at the displayed boundary.

Finally invoke Definition 580.28 and Definition 580.8. They keep RH predicates, non-trivial-zero locators, global zero classifiers, completed Gamma certificates, and off-packet dependencies outside the exposed source input. □

Theorem 580.22 (ZetaContinuationWitness eta-pole functional readiness)**.** *For an accepted* ZetaContinuationWitn *packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, the admissible continuation branches for a zero-facing consumer are exhausted by the eta route E, pole ledger P, functional-equation row F, trivial-zero ledger T, and Gamma boundary row G, transported through H, C, R, N inside the same* BHist *surface. No branch outside those rows is available as an analytic-continuation input.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket eta pole functional readiness`

Proof. Begin with Definition 580.2. The obligation surface binds carrier admission, analytic-continuation exactness, pole exactness, trivial-zero exactness, and Gamma-boundary non-escape to the same accepted witness packet.

Apply Definition 580.19. Its source obligation, analytic readback obligation, and zero ledger non-escape boundary assemble the zero-facing package over $B, E, A, P, F, T, G, H, C, R, N$.

Now use Definition 580.16. Eta and analytic readback pass through B, E, A, F, H, C, R, N and, where a pole or Gamma boundary is requested, through the displayed rows P, G.

The remaining zero-facing branch is the trivial-zero ledger. By Definition 580.17 and Definition 580.20, this branch is exactly the row T transported through H, C, R, N or else the request is refused.

Finally apply Definition 580.8. The BHist, hsame, Cont, Pkg, and NameCert dependencies are already coordinates of W, so no hidden continuation branch remains after the eta, pole, functional-equation, trivial-zero, and Gamma-boundary rows have been enumerated. □

Theorem 580.23 (ZetaContinuationWitness Gamma critical-strip readiness)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every Gamma-facing or critical-strip consumer stops at the displayed Gamma boundary row* G *unless the requested Gamma factor is already carried by the functional-equation row* F*. The same consumer route exposes no RH predicate, no non-trivial-zero locator, no global zero classifier, and no completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma critical strip readiness`

Proof. Start from Definition 580.28. A critical-strip consumer receives only the finite packet rows and cannot read RH predicates, non-trivial-zero locators, global zero classifiers, or completed Gamma certificates as packet input.

Apply Definition 580.14. It identifies the Gamma-facing public export with P, F, G, H, C, R, N and separates that export from Gamma chapter closure, RH assertions, zero locators, and hidden analytic branches.

Use Definition 580.16 for the functional-equation side. If the requested Gamma factor is already represented by F, the route may read that displayed factor together with P, G, H, C, R, N; otherwise it reaches the boundary row G and stops.

Finally combine Definition 580.17 with Definition 580.8. The zero ledger remains T, while the BHist, hsame, Cont, Pkg, and NameCert dependencies keep every Gamma or critical-strip read inside W. □

Theorem 580.24 (ZetaContinuationWitness root readiness lock)**.** *Every root-unblock consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must read the displayed packet rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

before any zeta-zero or critical-strip handoff. The rows are read as a single BHist*-anchored* NameCert *surface with* hsame *transports,* Cont *routes,* Pkg *provenance, and local naming, so no downstream zero-facing consumer can bypass the carrier through an off-packet analytic branch, an RH predicate, a non-trivial-zero locator, or a completed Gamma certificate.*

Proof. Project the accepted packet as $W = (B, E, A, P, F, T, G, H, C, R, N)$. The NameCert obligation surface of Definition 580.2 and the dependency ledger of Definition 580.8 list exactly the carrier, dependency, transport, route, provenance, and naming rows that a consumer must expose.

Apply the existing root child surfaces next. The root-unblock obligation surface, root zero-facing package, and zero-consumer readiness surface force the source, analytic, pole, functional-equation, trivial-zero, Gamma-boundary, transport, continuation, provenance, and local naming rows to appear before the consumer can read a zero-facing route.

Finally use Definition 580.28. It removes RH predicates, non-trivial-zero locators, global zero classifiers, off-packet analytic branches, and completed Gamma certificates from the packet input, so the only root-unblock handoff is through the displayed rows of W. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root readiness lock`

Theorem 580.25 (ZetaContinuationWitness zero-facing source accountability)**.** *Every zero-facing read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet exposes the source row B, eta route E, analytic-continuation row A, functional-equation row F, pole ledger P, trivial-zero ledger T, Gamma boundary G, transport rows H, continuation row C, package provenance R, and local name N before becoming input to a* $\mathsf{ZetaZeros}^{\uparrow}$*-facing consumer. In particular, the read presents the source row, eta route, analytic-continuation row, functional-equation row, and pole/trivial-zero/Gamma ledger as visible coordinates of the same packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero facing source accountability`

Proof. Start with Definition 580.8. It binds the $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows to the displayed packet before any zero-facing consumer can read it.

Use Definition 580.21 and Definition 580.22. These readiness rows expose B, E, A, F as the source-facing route and expose P, T, G as the pole, trivial-zero, and Gamma ledger routes transported through H, C, R, N.

The handoff is then constrained by Definition 580.28. Since the packet has no RH predicate, non-trivial-zero locator, global zero classifier, hidden analytic branch, or completed Gamma certificate, the zero-facing source account is exactly the visible packet rows. □

Theorem 580.26 (ZetaContinuationWitness Gamma-boundary root non-escape)**.** *Every Gamma-dependent root consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet stops at the displayed Gamma boundary row G unless the requested Gamma factor is already inside the functional-equation row F. Such a consumer may read the adjacent pole ledger P and the authenticating rows H, C, R, N, but it receives no completed Gamma certificate and no Gamma-side route outside the finite packet $B, E, A, P, F, T, G, H, C, R, N$.*

Proof. Begin with the pole and Gamma side of the packet. The NameCert obligation surface records P, F, G as displayed rows, and the dependency ledger keeps them inside the same BHist carrier with the transport, continuation, provenance, and naming rows.

Apply Definition 580.23. A Gamma-facing or critical-strip consumer stops at G unless the requested factor is already represented in F, and it receives no completed Gamma certificate.

Close with Definition 580.28 and the root child readiness surfaces. They ensure that the same boundary is the only root-facing Gamma handoff and that no off-packet analytic branch, RH predicate, non-trivial-zero locator, or global zero classifier is exported with the Gamma request. □

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma boundary

Theorem 580.27 (ZetaContinuationWitness root readiness source seal)**.** *Every zero-facing or critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must first read the source-seal rows*

$$B, E, A, F, H, C, R, N.$$

The pole ledger P, trivial-zero ledger T, and Gamma boundary G may be read only as displayed ledger boundaries of the same packet. No such consumer receives an RH predicate, zero locator, global zero classifier, hidden analytic branch, or completed Gamma certificate.

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root readiness source seal

Proof. Start with the root readiness lock, Definition 580.24. It forces the accepted packet $W = (B, E, A, P, F, T, G, H, C, R, N)$ to be exposed before any zero-facing or critical-strip handoff can read it.

Use the sibling zero-consumer readiness surface and the zero-facing source accountability theorem, Definition 580.21 and Definition 580.25. These rows make B, E, A, F, H, C, R, N the required source, analytic, functional-equation, transport, continuation, provenance, and naming surface, while P, T, G remain visible ledger boundaries.

Finally apply the Gamma-boundary root non-escape theorem, Definition 580.26, together with critical-strip input refusal. A Gamma-dependent consumer stops at G unless the requested factor is already represented in F, and the same packet exports no RH predicate, zero locator, global zero classifier, hidden analytic branch, or completed Gamma certificate. □

Theorem 580.28 (ZetaContinuationWitness critical-strip input refusal)**.** *An accepted* ZetaContinuationWitness$^{\uparrow}$ *packet contains no RH predicate, no non-trivial-zero locator, no global zero classifier, and no completed Gamma certificate. Consequently a critical-strip consumer receives only the displayed finite witness rows $B, E, A, P, F, T, G, H, C, R, N$ and cannot read any of those excluded objects as packet input.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip input refusal

Proof. Project $W = (B, E, A, P, F, T, G, H, C, R, N)$. Each coordinate is one of the displayed continuation, ledger, boundary, transport, route, provenance, or naming rows.

Apply the carrier definition and the NameCert obligation surface. They account for all admitted rows and list the analytic-continuation exactness, pole and trivial-zero ledger exactness, and Gamma-horizon non-escape clauses, but none of their coordinates is an RH predicate, non-trivial-zero locator, global zero classifier, or completed Gamma certificate.

The conclusion follows by coordinate exhaustion. Since the critical-strip route is a Cont read from the same packet, it receives only the displayed rows and has no input slot for the excluded data. □

Theorem 580.29 (ZetaContinuationWitness bridged-boundary mature route)**.** *Every bridge-facing consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet factors through the public export, source lock, pole-Gamma boundary, critical-strip route separation, dependency ledger, and critical-strip input refusal before reading the packet as a standard bridge boundary. The bridge-facing read exports no RH predicate, non-trivial-zero locator, completed Gamma certificate, or global zero classifier.*

Proof. Start with the public export and source lock. The accepted packet is already exposed as the finite witness rows $B, E, A, P, F, T, G, H, C, R, N$, together with their BHist, hsame, Cont, Pkg, and NameCert support.

Next use the pole-Gamma boundary and critical-strip route separation. These rows keep Gamma-facing reads inside P, F, G, H, C, R, N and keep critical-strip reads inside the same displayed finite packet.

Then apply the dependency ledger and critical-strip input refusal. The dependency ledger binds the source, continuation, analytic, holomorphic, complex, history, transport, route, package, and naming rows to the same witness packet, while the refusal theorem removes RH predicates, zero locators, completed Gamma certificates, and global zero classifiers from the input surface.

Finally combine the same public export with the displayed bridge-facing boundary. Any bridge-facing consumer therefore factors through the same finite witness packet before reaching the standard bridge boundary, so no excluded object can be exported as bridge input. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZetaContinuationWitness}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{ZetaContinuationWitness}^{\uparrow}$ packet is generated from a $\mathsf{ZetaBasic}^{\uparrow}$ source row, an eta-continuation route, an analytic-continuation row, a pole ledger, a functional-equation route, a trivial-zero ledger, a Gamma-horizon boundary, displayed hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ZetaContinuationWitness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 580.1, Definition 580.2, Definition 580.41, Definition 580.45, Definition 580.46, Definition 580.37, Definition 580.51, Definition 580.52, Definition 580.50, and Definition 580.56 define the local witness carrier, obligation surface, route-carrier admission, classifier stability, pole-ledger exactness, continuation handoff, trivial-zero ledger, Gamma-horizon boundary, critical-strip handoff, and local package and naming rows. Bridge-facing access is then routed through the public export, source lock, pole-Gamma boundary, critical-strip route separation, dependency ledger, and critical-strip input refusal, so the chapter closes the bridge boundary on the displayed packet only.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ZetaContinuationWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket\_public\_export`

Bridge status: none

Origin: human

Not claimed: This chapter does not prove RH, locate non-trivial zeros, close the Gamma chapter, decide global branch behaviour, complete quotient-of-germs construction, or provide bridge checking.

Upgrade path: Bridge checking remains a later step if a standard bridge is added for the same finite witness packet without introducing RH or zero-location data.

Theorem 580.30 (ZetaContinuationWitness critical-strip bridge row)**.** *Every critical-strip consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet can read only the displayed rows* $B, E, A, P, F, T, G, H, C, R, N$. *The consumer cannot import an RH predicate, a non-trivial-zero location row, or a global zero classifier as packet input; those objects are outside the carrier, outside the NameCert obligation surface, and outside the exhausted consumer ledger.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip bridge row`

Proof. Start with the carrier and the NameCert obligation surface. The carrier lists exactly $B, E, A, P, F, T, G, H, C, R, N$, and the obligation rows account for source, eta, continuation, pole, functional-equation, trivial-zero, Gamma, transport, route, provenance, and local naming reads.

Apply the pole-Gamma boundary and critical-strip input refusal. These rows exclude RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from the packet input.

Finally use route-carrier admission and consumer-ledger exhaustion. Any critical-strip route must be replayed through the same displayed Cont row and authenticated by R, N, so there is no remaining bridge through which an excluded critical-strip object can enter. □

Theorem 580.31 (ZetaContinuationWitness source-lock triad)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

every zero-facing or critical-strip consumer must first expose the source triad B, E, A together with the functional-equation route F. Thus the mandatory source, eta, analytic-continuation, and functional-equation route is B, E, A, F before any zero consumer can read the packet.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket source lock triad`

Proof. Project the carrier of Definition 580.1. The rows B, E, A, F are displayed coordinates of the same finite packet, not ambient analytic data.

Apply source-lock coverage and dependency-ledger exhaustion. Definition 580.4 exposes B, E, A, F before continuation, zero, or critical-strip use, and Definition 580.8 keeps the consumer route inside the displayed dependency ledger. □

Theorem 580.32 (ZetaContinuationWitness Gamma-boundary triad)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, the pole, trivial-zero, and Gamma-horizon ledger is exactly the displayed triad P, T, G. Pole-facing reads remain in P, trivial-zero reads remain in T, and Gamma-dependent reads stop at G unless the requested Gamma factor is already displayed inside F; the packet carries no completed Gamma certificate.*

Proof. Use the carrier projection and the NameCert obligation surface. The only pole, trivial-zero, and Gamma-horizon rows named by Definition 580.1 and Definition 580.2 are P, T, G.

Now apply the pole-Gamma boundary and Gamma-horizon consumer row. Definition 580.5 confines pole and Gamma reads to P, F, G, while Definition 580.35 refuses any completed Gamma certificate beyond the displayed functional-equation factor and the boundary row. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma boundary triad`

Theorem 580.33 (ZetaContinuationWitness critical-strip ledger triad)**.** *Every critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receives the source-lock rows B, E, A, F, the ledger triad P, T, G, and the authentication rows H, C, R, N as one finite ledger $B, E, A, P, F, T, G, H, C, R, N$. It receives no RH predicate, non-trivial-zero locator, global zero classifier, hidden analytic branch, or completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessCriticalStripLedgerTriad`

Proof. First combine the two triads. Definition 580.31 supplies B, E, A, F, and Definition 580.32 supplies P, T, G as the only pole, trivial-zero, and Gamma-horizon ledger.

Next attach the transport, route, provenance, and naming rows. The carrier and Definition 580.30 route every critical-strip consumer through H, C, R, N after the displayed ledger rows.

Finally apply critical-strip input refusal. Definition 580.28 excludes RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from the packet input, so the consumer ledger is exactly the finite packet already displayed. □

Theorem 580.34 (ZetaContinuationWitness pole-ledger refusal row)**.** *Every pole-facing read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet stays in the displayed pole ledger P and its authentication rows H, C, R, N. The pole row cannot become a separate pole-completion object, cannot supply a second continuation carrier, and cannot bypass the eta, functional-equation, Gamma-boundary, and consumer-ledger rows already displayed by the packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket pole ledger refusal row`

Proof. Project the carrier and isolate the pole coordinate P. It is a ledger row inside $W = (B, E, A, P, F, T, G, H, C, R, N)$, not an independent completion carrier.

Use the NameCert obligation surface together with the pole-Gamma boundary. The obligation surface treats pole exactness as a finite ledger read, and the boundary keeps pole reads in P while routing Gamma-dependent reads through F or G.

Route-carrier admission and consumer-ledger exhaustion close the refusal. Every pole-facing consumer must replay the displayed route through C and authenticate it through R, N, so the pole row cannot split off as a separate object or bypass the displayed witness packet. □

Theorem 580.35 (ZetaContinuationWitness Gamma-horizon consumer row)**.** *Every Gamma-dependent consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet either reads a Gamma factor already displayed in the functional-equation row F or stops at the Gamma-horizon boundary G. The packet supplies no completed Gamma certificate and no hidden Gamma route beyond F, G, H, C, R, N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessGammaHorizonConsumerRow`

Proof. Begin with the carrier and NameCert obligation surface. The only Gamma-facing coordinates are the displayed factor inside F and the horizon boundary G.

Apply the pole-Gamma boundary and critical-strip input refusal. They prevent a Gamma-dependent read from importing a completed Gamma certificate or using Gamma data as an RH-facing critical-strip input.

Use route-carrier admission and consumer-ledger exhaustion for the final route. Once the read lands in F or G, it is transported, replayed, packaged, and named only through H, C, R, N, so no further Gamma-horizon consumer row is available. □

Theorem 580.36 (ZetaContinuationWitness eta-functional route coverage)**.** *Every exported continuation route of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet is covered by one of five displayed rows: the eta-continuation row E, the pole ledger P, the functional-equation row F, the trivial-zero ledger T, or the Gamma-horizon boundary G. There is no unrecorded continuation branch outside those rows and their accompanying* hsame, Cont, Pkg, *and* NameCert *authentication data.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket eta functional route coverage`

Proof. Project $W = (B, E, A, P, F, T, G, H, C, R, N)$. The route-bearing analytic coordinates are the eta-continuation row E, the analytic-continuation row A, and the functional-equation row F, while exceptional reads are represented by P, T, G.

Use the obligation surface. Analytic-continuation exactness routes exported $\mathsf{ZetaCont}^{\uparrow}$ reads through the source, eta, analytic-continuation, and functional-equation coordinates, while the pole, trivial-zero, and Gamma-horizon clauses account for P, T, G.

Finally replay the continuation routes in C through the transports H and authenticate them with R, N. Since every exported read has landed in E, P, F, T, G before authentication, no unrecorded branch remains. □

Theorem 580.37 (ZetaContinuationWitness continuation handoff)**.** *Every* $\mathsf{ZetaZeros}^{\uparrow}$ *or critical-strip consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier reads the continued zeta value through the displayed rows* $B, E, A, P, F, T, G, H, C, R, N$*. In particular, the handoff exposes the* $\mathsf{ZetaCont}^{\uparrow}$ *value surface, the pole boundary at* $s = 1$*, the trivial-zero ledger, and the Gamma-horizon boundary, but it exports no RH conclusion and no zero-location classifier.*

Proof. Start with the consumer route stored in C. If it asks for the basic zeta source, the eta continuation, the analytic-continuation row, or the functional-equation route, the requested coordinate is one of B, E, A, F and is transported through H.

If the consumer asks about the pole at $s = 1$ or a negative even input, the request lands in the ledger rows P or T. These rows expose the relevant boundary or zero witness as ledger data attached to the same packet.

The remaining analytic escape route is Gamma-dependent. That route is absorbed by G unless the requested factor is already one of the displayed functional-equation rows. Therefore the downstream consumer receives only the finite continuation witness packet and does not receive an RH proof, a non-trivial-zero locator, or a completed Gamma certificate. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket continuation handoff`

Theorem 580.38 (ZetaContinuationWitness zero-consumer handoff)**.** *Every* $\mathsf{ZetaZeros}^{\uparrow}$ *or critical-strip consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet first exposes the finite rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The handoff cannot present a zero-facing conclusion before carrier admission, NameCert obligation display, route-carrier admission, pole-Gamma boundary control, critical-strip input refusal, and consumer-ledger exhaustion have displayed those rows.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessZeroConsumerHandoff`

Proof. Begin with the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$ and the NameCert obligation surface. These two rows provide the only public inventory for a zero-facing or critical-strip read.

Use route-carrier admission to move from the raw carrier to the admitted continuation route. The admitted route is replayed only through the displayed Cont rows and authenticated only by the displayed provenance and naming rows.

Apply pole-Gamma boundary control and critical-strip input refusal. Pole-facing requests stay in P, Gamma-dependent requests stay in G unless already displayed in F, and critical-strip requests cannot import RH predicates, non-trivial-zero locators, global zero classifiers, or completed Gamma data.

Consumer-ledger exhaustion then closes the handoff. Since every surviving consumer read has already landed in $B, E, A, P, F, T, G, H, C, R, N$, there is no remaining route by which a zero-facing consumer can receive an off-packet conclusion. □

Theorem 580.39 (ZetaContinuationWitness critical-strip pole boundary)**.** *Every pole-facing or critical-strip route of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet must pass through the displayed pole ledger* P*, functional-equation row* F*, and Gamma-horizon row* G *before it reaches any downstream consumer. No such route can bypass* P, F, G *by using an RH predicate, a non-trivial-zero locator, a global critical-strip classifier, or a completed Gamma certificate.*

Proof. Project the carrier and isolate the boundary rows P, F, G. The pole row is the only pole-facing ledger, the functional-equation row is the only displayed factor route, and the Gamma-horizon row is the only Gamma-dependent boundary outside the factors already stored in F.

The NameCert obligation surface and pole-Gamma boundary identify these coordinates as mandatory boundary reads. A pole-facing request cannot leave P, while a Gamma-dependent request terminates at G unless the datum is already part of F.

Critical-strip input refusal removes the external shortcuts. The packet has no RH coordinate, no non-trivial-zero locator, no global classifier row, and no completed Gamma certificate.

Finally apply route-carrier admission and consumer-ledger exhaustion. Every admitted continuation route is replayed through the same displayed carrier and named by R, N, so a pole-facing or critical-strip consumer must pass through P, F, G rather than bypassing the boundary. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessCriticalStripPoleBoundary`

Theorem 580.40 (ZetaContinuationWitness Gamma-route non-escape ledger)**.** *Every Gamma-dependent read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet terminates at the displayed Gamma-horizon row G unless the requested Gamma datum is already displayed in the functional-equation row F. After that landing, the read is transported, replayed, packaged, and named only through H, C, R, N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessGammaRouteNonescapeLedger`

Proof. Start from the carrier and the NameCert obligation surface. They expose F and G as the only Gamma-facing rows of the packet.

Use the pole-Gamma boundary. It sends Gamma-dependent reads to G unless the requested factor has already appeared inside the functional-equation row F.

Critical-strip input refusal prevents the Gamma read from being reinterpreted as a critical-strip shortcut. The packet does not supply RH data, a zero-location classifier, or a completed Gamma object.

Route-carrier admission and consumer-ledger exhaustion finish the ledger check. Once the read has landed in F or G, the only remaining public rows are H, C, R, N, so the Gamma route has no off-packet escape. □

Theorem 580.41 (ZetaContinuationWitness route-carrier admission)**.** *An accepted* ZetaContinuationWitness$^{\uparrow}$ *packet admits a continuation route only when the route is carried by the displayed eta-continuation row E, pole ledger P, functional-equation row F, trivial-zero ledger T, Gamma-horizon boundary G, componentwise* hsame *transports H,* Cont *reads C,* Pkg *provenance R, and local* NameCert *row N. No continuation route is admitted from an unrecorded branch, an RH predicate, a global zero-location classifier, or a completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket route carrier admission`

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$. The eta-continuation coordinate E, the analytic-continuation coordinate A, and the functional-equation coordinate F are the only route-bearing analytic rows.

The ledger rows P and T restrict the exceptional reads, while G records the Gamma-horizon boundary. Transport through H and replay through C preserve those displayed rows and do not add a branch.

Finally authenticate the route through R and N. Since the carrier contains no RH coordinate, no non-trivial-zero locator, and no closed Gamma chapter, admission is exactly the displayed finite route carrier. □

Theorem 580.42 (ZetaContinuationWitness consumer ledger exhaustion)**.** *Every consumer read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet factors through the displayed carrier rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

No downstream zeta-zero, critical-strip, pole-facing, trivial-zero, or Gamma-facing consumer can obtain a continuation witness row outside that finite ledger.

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$. The tuple lists the source zeta row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transports, continuation reads, provenance, and local name.

Apply the NameCert obligation surface. It binds carrier admission, classifier stability, analytic-continuation exactness, pole and trivial-zero ledger exactness, and Gamma-horizon non-escape to precisely those coordinates.

Finally use route-carrier admission. Every admitted public continuation route is replayed through C and authenticated by R, N, so there is no remaining consumer-visible row outside the displayed ledger. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessConsumerLedgerExhaustion`

Theorem 580.43 (ZetaContinuationWitness critical-strip handoff scope)**.** *Every critical-strip consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet receives only the displayed continuation rows* B, E, A, F*, the pole row* P*, the trivial-zero row* T*, the Gamma-boundary row* G*, and the structural rows* H, C, R, N*. It receives no RH predicate, no non-trivial-zero locator, no global zero-classifier row, and no completed Gamma certificate.*

Proof. Start with a critical-strip read and project the accepted carrier. Its analytic route-bearing coordinates are B, E, A, F, and its exceptional boundary coordinates are P, T, G.

Use critical-strip input refusal and route-carrier admission. The first theorem excludes RH, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from the packet input; the second forces admitted continuation routes through the displayed eta, pole, functional-equation, trivial-zero, Gamma, transport, route, provenance, and naming rows.

The structural rows H, C, R, N only transport, replay, package, and name those displayed coordinates. Therefore the critical-strip handoff scope is exactly the advertised continuation, pole, trivial-zero, and Gamma-boundary surface. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip handoff scope`

Theorem 580.44 (ZetaContinuationWitness Gamma-horizon non-escape row)**.** *Every Gamma-dependent read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet lands at the displayed Gamma-horizon boundary* G *unless the requested datum is already part of the functional-equation factor row* F*. The read is then transported, replayed, packaged, and named only through* H, C, R, N*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma horizon nonescape row`

Proof. Project the carrier and isolate the Gamma-facing rows. The functional-equation route F contains the displayed Gamma factor only as part of the factor ledger, while G is the separate Gamma-horizon boundary.

Apply the NameCert obligation surface. Its Gamma-horizon clause says that a Gamma-dependent read reaches G unless it is already one of the displayed functional-equation factors.

Route-carrier admission and consumer ledger exhaustion then force the read through H, C, R, N after it lands in F or G. Since the carrier has no closed Gamma chapter, hidden branch selector, RH predicate, or zero-location coordinate, no Gamma-dependent read escapes the displayed boundary. □

Theorem 580.45 (ZetaContinuationWitness classifier stability)**.** *The* ZetaContinuationWitness$^{\uparrow}$ *classifier is stable under componentwise* hsame *and displayed* Cont *transport of the source zeta row, eta-continuation route, pole ledger, functional-equation route, trivial-zero ledger, Gamma-horizon boundary,* Pkg *provenance, and local* NameCert *rows. Transported packets keep the same finite witness surface and do not gain RH, global zero-location, or analytic-continuation data beyond the displayed witness packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket classifier stability`

Proof. Compare two accepted packets componentwise. The source row B, eta route E, analytic-continuation row A, pole ledger P, functional-equation row F, trivial-zero ledger T, and Gamma boundary G are transported only along the corresponding rows of H.

The continuation reads in C replay those transported rows in the same order. They preserve the route carrier and the exceptional ledgers because each read targets a displayed coordinate of the packet.

Package provenance R and local naming N then repack the transported surface. No classifier coordinate mentions RH, a non-trivial-zero locator, a global branch theorem, or a completed Gamma chapter, so stability remains finite-packet stability. □

Theorem 580.46 (ZetaContinuationWitness pole-ledger exactness)**.** *Every pole-facing read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet is exactly the displayed pole ledger P together with the eta-continuation, functional-equation, Gamma-horizon,* hsame, Cont, Pkg, *and* NameCert *rows needed to authenticate that ledger. The read does not decide non-trivial zeros, does not prove RH, and does not extend analytic continuation beyond the witness packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket pole ledger exactness`

Proof. Read the exceptional coordinate P from the carrier. It records the simple-pole row and residue-one readback at $s = 1$, so a pole-facing consumer has a named ledger rather than an ambient meromorphic theorem.

The surrounding rows E, F, G, H, C, R, N authenticate the ledger. The eta and functional-equation routes place the pole row inside the witness packet, the Gamma boundary limits Gamma-dependent access, and the transport and continuation rows replay only displayed data.

The trivial-zero ledger T remains separate. Therefore pole exactness does not locate non-trivial zeros, does not assert RH, and does not supply continuation data outside the finite certificate. □

Theorem 580.47 (ZetaContinuationWitness carrier admission tightening)**.** *Let $W = (B, E, A, P, F, T, G, H, C, R,$ be an accepted* ZetaContinuationWitness$^{\uparrow}$ *carrier. Every public continuation route admitted by the local* NameCert *projects through the displayed rows $B, E, A, P, F, T, G, H, C, R, N$. In particular, eta continuation, pole reads, functional-equation reads, trivial-zero reads, and Gamma-boundary reads are all carrier reads of this packet and not reads of an unrecorded analytic branch.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessCarrierAdmissionTightening

Proof. Start from the carrier definition. The packet lists the basic zeta source, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transport rows, continuation rows, provenance, and local name as the complete public inventory.

Apply the NameCert obligation surface and the continuation handoff. Carrier admission supplies the tuple, while the handoff sends every $\mathsf{ZetaZeros}^{\uparrow}$ or critical-strip consumer through the same displayed rows.

Use route-carrier admission to rule out unrecorded branches, RH predicates, global zero-location classifiers, and completed Gamma certificates. The classifier-stability and pole-ledger exactness rows then keep the exceptional pole and zero data inside the same finite carrier surface. □

Theorem 580.48 (ZetaContinuationWitness route stability budget)**.** *For accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carriers* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, componentwise* hsame *transport together with displayed* Cont *replay preserves the eta-continuation row* E*, pole ledger* P*, functional-equation row* F*, trivial-zero ledger* T*, and Gamma-boundary row* G*. A transported public route therefore has the same finite continuation budget and cannot gain an extra continuation, zero-location, or Gamma-completion row.*

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket route stability budget

Proof. Project the transport coordinate H and continuation coordinate C from the carrier. The source, eta, analytic-continuation, pole, functional-equation, trivial-zero, and Gamma-boundary rows are transported componentwise rather than through a single ambient analytic equality.

Apply classifier stability. It states that transported packets keep the same finite witness surface and that replay through C reads the displayed coordinates in the same route shape.

The continuation handoff and route-carrier admission identify the public route consumed downstream. Pole-ledger exactness keeps P separate from non-trivial-zero location, while the NameCert obligation surface keeps T and G as displayed ledger and boundary rows. Hence the route budget is stable under the advertised hsame and Cont data. □

Theorem 580.49 (ZetaContinuationWitness ledger non-escape boundary)**.** *An accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier exposes no RH predicate, no non-trivial-zero location row, no hidden Gamma completion, and no unrecorded branch data. Every public ledger read is confined to the displayed packet*

$$B, E, A, P, F, T, G, H, C, R, N.$$

Lean4 ✓ BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessCarrier ledger nonescape boundary

Proof. Inspect the carrier definition. Its coordinates include explicit eta, pole, functional-equation, trivial-zero, and Gamma-boundary rows, but no RH coordinate, non-trivial-zero locator, closed Gamma theorem, or branch selector.

Apply the NameCert obligation surface for ledger exactness and Gamma-horizon non-escape. The pole and trivial-zero ledgers are read as P and T, and every Gamma-dependent request either uses a displayed functional-equation factor or terminates at G.

Finally use continuation handoff, route-carrier admission, classifier stability, and pole-ledger exactness. These rows force downstream zeta and critical-strip consumers to replay the same finite

route and keep exceptional ledgers inside the witness packet, so no public read escapes to the excluded data. □

Theorem 580.50 (ZetaContinuationWitness critical-strip handoff)**.** *Every critical-strip or zeta-zero consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet reads only the displayed rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The handoff may expose the basic zeta source, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma-horizon boundary, transport, continuation, package provenance, and local name, but it exports no RH predicate, no non-trivial-zero locator, and no global zero-classifier row.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip handoff`

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$. These coordinates are the whole continuation witness packet available to downstream readers.

Apply the NameCert obligation surface. It lists carrier admission, classifier stability, analytic-continuation exactness, pole and trivial-zero ledger exactness, and Gamma-horizon non-escape as the finite rows exposed by the local name.

Apply the continuation handoff. It sends $\mathsf{ZetaZeros}^{\uparrow}$ and critical-strip reads through the same displayed coordinates and records every move in H, C, R, N.

Coordinate exhaustion closes the boundary. Since no coordinate stores RH, a non-trivial-zero locator, or a global zero classifier, a critical-strip consumer cannot read any such object from this packet. □

Theorem 580.51 (ZetaContinuationWitness trivial-zero ledger non-escape)**.** *For an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet, the negative-even ledger* T *remains a trivial-zero ledger read through the sine-factor part of the displayed functional-equation route. It cannot become an RH predicate, a non-trivial-zero locator, a global zero-classifier row, or a substitute for the critical-strip handoff.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket trivial zero ledger nonescape`

Proof. Project the carrier. The row T is listed as the trivial-zero ledger for negative even inputs, while the functional-equation route is stored separately at F and the downstream continuation routes are stored in C.

Use the NameCert obligation surface and pole-ledger exactness. The ledger clauses keep P and T as exceptional ledger reads, and the pole theorem already separates exceptional ledger exactness from RH, non-trivial-zero location, and continuation outside the witness packet.

Use the continuation handoff for downstream consumers. A zeta-zero or critical-strip read may see T only as part of the displayed finite witness route, with transport, provenance, and naming recorded in H, R, N.

No remaining coordinate can reinterpret the negative-even ledger as a critical-strip predicate. Thus T cannot escape into RH, a non-trivial-zero locator, or a global zero classifier. □

Theorem 580.52 (ZetaContinuationWitness Gamma-horizon consumer boundary)**.** *Every Gamma-dependent read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet terminates at the Gamma-horizon*

boundary row G unless the requested datum is already one of the displayed functional-equation factors carried by F. The packet therefore exposes the Gamma factor only through F, G, H, C, R, N and does not close a Gamma chapter, select a hidden branch, or supply an RH-facing analytic shortcut.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma horizon consumer boundary`

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$. The only Gamma-facing coordinates are the functional-equation route F and the Gamma-horizon boundary row G.

Apply the NameCert obligation surface. Its Gamma-horizon clause states that a Gamma-dependent read reaches G unless it is already a displayed functional-equation factor.

Apply the continuation handoff. The read is replayed through C, transported through H, and packaged and named by R, N, so the consumer receives the same finite boundary rather than a completed Gamma certificate.

Coordinate exhaustion excludes the analytic escape. The carrier has no hidden branch selector, closed Gamma theorem, RH predicate, or zero-location coordinate, so Gamma-dependent reads stop at the displayed horizon boundary. □

Theorem 580.53 (ZetaContinuationWitness vision concretization)**.** *The zeta-continuation application of chapter 9342 is concretized by* ZetaContinuationWitness$^{\uparrow}$ *as a finite* BHist *NameCert packet over the eta-continuation route, pole ledger, functional-equation route, trivial-zero ledger, Gamma-horizon boundary, componentwise* hsame *transports, finite* Cont *reads,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessVisionConcretization`

Proof. The carrier definition supplies the finite packet. The NameCert obligation surface binds carrier admission, classifier stability, analytic-continuation exactness, pole and trivial-zero ledger exactness, and Gamma-horizon non-escape for precisely those coordinates.

The continuation handoff theorem identifies the downstream consumer surface for ZetaZeros$^{\uparrow}$ and critical-strip chapters. Since the packet exports no RH conclusion, no non-trivial-zero locator, and no closed Gamma chapter, it concretizes the application as a witness interface rather than as a proof of any RH-facing conclusion. □

Theorem 580.54 (ZetaContinuationWitness eta-denominator boundary)**.** *For an accepted* ZetaContinuationWitnes *packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, the eta route row E is the only denominator-boundary witness for the factor $1 - 2^{1-s}$. A public read may consume that boundary through E, H, C, R, N, but it cannot export a branch selector, a zero-location classifier, or an RH predicate from the denominator row.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket eta denominator boundary`

Proof. Project the carrier. The denominator boundary is listed only inside the eta route coordinate E, while the route replay and authentication data are the displayed rows H, C, R, N.

Apply the NameCert obligation surface and route-carrier admission. These rows admit continuation data only from the displayed eta, pole, functional-equation, trivial-zero, and Gamma-boundary coordinates.

Finally use the critical-strip handoff and ledger non-escape boundary. Since the carrier has no branch selector, global zero-location classifier, or RH coordinate, a denominator read through E remains a boundary witness for $1 - 2^{1-s}$ and cannot export any of the excluded data. □

Theorem 580.55 (ZetaContinuationWitness functional-factor non-escape)**.** *Every functional-equation read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet through* F, G, H, C, R, N *exposes only the displayed sine factor, reflected-zeta route, exponential factor, and Gamma-boundary rows. It supplies no RH predicate, no non-trivial-zero locator, and no classifier for the critical strip beyond the finite witness packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket functional factor nonescape`

Proof. Start from the functional-equation coordinate F. Its public factors are the displayed sine, reflected-zeta, exponential, and Gamma-facing rows, with the Gamma-dependent boundary recorded separately at G.

Use the Gamma-horizon consumer boundary. A Gamma-dependent request terminates at G unless it is already one of the functional-equation factors carried by F.

Replay the read through H, C, R, N and then apply the critical-strip handoff. The downstream route receives only the finite continuation witness rows, so no RH predicate, non-trivial-zero locator, or independent critical-strip classifier is exported. □

Theorem 580.56 (ZetaContinuationWitness package provenance exactness)**.** *Every public read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet factors through the displayed* Pkg *provenance row* R *and the local naming row* N *over* $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, *and* Cont. *Thus the public package is exactly the finite surface* $B, E, A, P, F, T, G, H, C, R, N$ *and has no additional analytic, branch-selection, or zero-location coordinate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket package provenance exactness`

Proof. Project the carrier $W = (B, E, A, P, F, T, G, H, C, R, N)$. The provenance row R names the source zeta, continuation, analytic, holomorphic, complex, history, transport, continuation-route, package, and local certificate dependencies, while N is the local name.

Apply the dependency ledger and NameCert obligation surface. They bind carrier admission, classifier stability, analytic-continuation exactness, ledger exactness, and Gamma-horizon non-escape to the same displayed tuple.

Use route stability and the continuation handoff for downstream reads. Every public route is replayed through H, C and authenticated by R, N, so no consumer can obtain a separate analytic branch, branch selector, or zero-location row outside the package provenance. □

Theorem 580.57 (ZetaContinuationWitness root carrier exposure)**.** *Every accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carrier exposes the root-unblock surface only through*

$$B, E, A, P, F, T, G, H, C, R, N.$$

These rows bind the $\mathsf{ZetaBasic}^{\uparrow}$ *source, eta route, analytic-continuation row, complex and holomorphic dependency rows,* $\mathsf{ZetaCont}^{\uparrow}$ *continuation route,* $\mathsf{ZetaZeros}^{\uparrow}$ *consumer boundary, exceptional ledgers, Gamma boundary, transport, replay, provenance, and naming data. The carrier does not admit analytic continuation as a primitive and does not assert any critical-strip truth claim.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket root carrier exposure`

Proof. Project the carrier of Definition 580.1. Its displayed coordinates are exactly the source, continuation, ledger, boundary, transport, replay, provenance, and naming rows.

Use Definition 580.8 and Definition 580.56. They bind the $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{ZetaCont}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ dependencies to that finite packet.

Finally apply the critical-strip input refusal and ledger non-escape boundary. Since the packet has no RH coordinate, no global zero-location classifier, and no closed Gamma coordinate, the root carrier exposes only the displayed witness rows. □

Theorem 580.58 (ZetaContinuationWitness route classifier stability)**.** *For accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *carriers, the route classifier is stable only along the displayed analytic-continuation, complex, holomorphic,* $\mathsf{ZetaBasic}^{\uparrow}$*,* $\mathsf{ZetaCont}^{\uparrow}$*, and* $\mathsf{ZetaZeros}^{\uparrow}$ *dependency rows recorded in* B, E, A, F, H, C, R, N*, with pole, trivial-zero, and Gamma-boundary reads confined to* P, T, G*. It cannot stabilize a hidden analytic continuation primitive, RH predicate, or critical-strip zero-location row.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessRouteClassifierStability`

Proof. Start with classifier stability Definition 580.45. It transports the source row, eta route, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, package provenance, and local name componentwise.

Use route stability budget Definition 580.48. The public route keeps the same eta, pole, functional-equation, trivial-zero, and Gamma-boundary budget after hsame transport and Cont replay.

Apply carrier admission tightening and functional-factor non-escape. They force every classified route through the displayed dependency rows and rule out hidden branch data, RH predicates, and zero-location classifiers. □

Theorem 580.59 (ZetaContinuationWitness ledger scope)**.** *The* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *ledger scope consists exactly of the displayed* $\mathsf{ZetaBasic}^{\uparrow}$*,* $\mathsf{ZetaCont}^{\uparrow}$*,* $\mathsf{AnaCont}^{\uparrow}$*,* $\mathsf{Holomorphic}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*, pole, trivial-zero, Gamma-boundary, transport, replay, package, and naming rows of* $W = (B, E, A, P, F, T, G, H, C,$ *Downstream critical-strip and zeta-zero consumers may use this ledger as a finite witness interface, but not as a proof of RH, a non-trivial-zero locator, or an ambient analytic-continuation primitive.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessLedgerScope`

Proof. Read the finite ledger from the carrier definition and the NameCert obligation surface. The rows B, E, A, F hold the continuation-facing data, P, T, G hold the exceptional and boundary ledgers, and H, C, R, N authenticate the public route.

Apply the critical-strip handoff theorem. It sends critical-strip and zeta-zero consumers through the same displayed packet and excludes RH predicates, non-trivial-zero locators, and global zero classifiers from the handoff.

Use package provenance exactness to close the scope. The dependency rows are exactly the finite provenance surface over zeta, analytic, holomorphic, complex, history, transport, continuation, package, and naming data, so the ledger cannot be read as an off-table analytic claim. □

Theorem 580.60 (ZetaContinuationWitness carrier obligation)**.** *Every root-unblock consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet must first admit the displayed carrier*

$$W = (B, E, A, P, F, T, G, H, C, R, N)$$

as a BHist *witness over the* ZetaBasic$^{\uparrow}$, ZetaCont$^{\uparrow}$, AnaCont$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows. Carrier admission supplies only these rows and no analytic-continuation primitive outside the packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket carrier obligation`

Proof. Project the carrier definition. Its coordinates list the source, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transports, replay routes, package provenance, and local name.

Use the dependency ledger and ledger scope. They bind every root-facing read to the same BHist packet and prevent a consumer from reading analytic continuation as an independent primitive.

Thus the root-unblock surface begins with carrier admission. Any downstream continuation, zero, or critical-strip read must consume the displayed packet rows first. □

Theorem 580.61 (ZetaContinuationWitness ledger boundary)**.** *The root-unblock ledger for* ZetaContinuationWitne *is exactly the finite row sequence* $B, E, A, P, F, T, G, H, C, R, N$*. The pole row* P*, trivial-zero row* T*, and Gamma boundary* G *are ledger entries of the same packet; they cannot be read as RH data, non-trivial-zero locators, global zero classifiers, completed Gamma objects, or hidden branch selectors.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket ledger boundary`

Proof. Start from the carrier obligation. It admits one displayed packet and no off-packet analytic primitive.

Apply the pole-Gamma boundary, trivial-zero ledger non-escape, and Gamma-horizon consumer boundary. These rows keep P, T, G inside the finite ledger and separate them from RH-facing or Gamma-completion claims.

Finally use critical-strip handoff. A critical-strip or zeta-zero consumer may read the finite continuation witness interface, but the excluded objects have no row in the displayed ledger. □

Theorem 580.62 (ZetaContinuationWitness NameCert package)**.** *The root-unblock package for* ZetaContinuationWitness$^{\uparrow}$ *is the local* NameCert *export obtained by combining carrier admission, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local naming row for* $W = (B, E, A, P, F, T, G, H, C, R, N)$*. This package records the visible continuation witness and does not assert analytic-continuation strength beyond the displayed witness packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket namecert package`

Proof. Use package provenance exactness. It forces every public read through the displayed Pkg row R and local naming row N over the same carrier coordinates.

Use route stability budget for transport and replay. Componentwise hsame rows and Cont routes preserve the eta, pole, functional-equation, trivial-zero, and Gamma-boundary rows without adding another continuation branch.

Combine these rows with the carrier obligation and ledger boundary. The resulting NameCert package is a finite witness export, not a proof of RH, a selected analytic branch, or a completed Gamma surface. □

Theorem 580.63 (ZetaContinuationWitness source-route scoped totality)**.** *For every accepted* ZetaContinuationWitness$^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

the source route is scoped-total exactly over the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The source-lock coverage, critical-strip route separation, and ledger-scope theorems force every downstream critical-strip or zeta-zero read to expose B, E, A, F together with P, T, G, H, C, R, N before the packet is consumed.*

The total route does not assert RH, a non-trivial-zero location, a completed Gamma object, or analytic continuation beyond the displayed witness packet.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket source route scoped totality`

Proof. Project the packet W. Its source, eta, analytic-continuation, pole, functional-equation, trivial-zero, Gamma-boundary, transport, replay, provenance, and naming rows are all displayed coordinates of the same BHist carrier.

Apply source-lock coverage and critical-strip route separation. These theorems require the source lock and the exceptional ledgers to be visible before a critical-strip consumer receives the route.

Use ledger scope to identify the complete dependency surface. Since the surface is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert package over W, the source route is total only in that scoped sense and has no off-packet RH, zero-location, Gamma-completion, or analytic-continuation conclusion. □

Theorem 580.64 (ZetaContinuationWitness functional-boundary scoped exactness)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, every functional-equation, pole, and Gamma-boundary read is scoped-exact through the displayed rows P, F, G, H, C, R, N and through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *carrier surface. The pole-Gamma boundary, functional-factor boundary, critical-strip refusal, and ledger boundary theorems leave no route to a closed Gamma chapter, a hidden analytic branch, RH, or a non-trivial-zero locator.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessFunctionalBoundaryScopedExactness`

Proof. Read the boundary rows from the carrier. The pole row P, functional-equation route F, and Gamma-horizon row G are separate coordinates, and H, C, R, N are the only transport, replay, provenance, and naming rows available for them.

Apply the pole-Gamma boundary and the functional-equation boundary. Pole data remains in P, while Gamma-dependent data remains in G unless it is already one of the factors displayed inside F.

Finally apply critical-strip input refusal and the ledger boundary. These rows prevent a downstream consumer from turning the functional boundary into RH data, a non-trivial-zero locator, a closed Gamma certificate, or a hidden analytic-continuation branch. □

Theorem 580.65 (ZetaContinuationWitness local NameCert scoped package)**.** *The local* NameCert$_{\text{ZetaContinuationWi}}$ *package for an accepted packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N)$$

is the scoped package whose public rows are precisely the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates of W. It is obtained from source-lock coverage, pole-Gamma boundary control, critical-strip refusal, ledger scope, and package provenance exactness, and it exports only the witness packet rather than RH, zero-location data, completed Gamma, or analytic continuation beyond the displayed route.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessLocalNameCertScopedPackage`

Proof. Begin with package provenance exactness. It identifies the public package as the provenance row R and local naming row N over the displayed zeta, analytic, holomorphic, complex, history, transport, and route rows.

Add the source-lock and pole-Gamma rows. They ensure that source, eta, analytic-continuation, pole, functional-equation, trivial-zero, and Gamma-boundary data are carried by the single packet W before the local name is exported.

Use critical-strip refusal and ledger scope for downstream reads. A consumer preserving the scoped package preserves exactly the rows $B, E, A, P, F, T, G, H, C, R, N$ and receives no RH predicate, non-trivial-zero locator, completed Gamma certificate, or off-packet analytic-continuation primitive. □

Theorem 580.66 (ZetaContinuationWitness scoped route package)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

carrier admission, route-carrier admission, classifier stability, pole-ledger exactness, continuation handoff, trivial-zero non-escape, Gamma boundary control, critical-strip handoff, package provenance, and local naming assemble one scoped witness surface over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The surface exposes exactly the source row B, eta route E, analytic-continuation row A, pole ledger P, functional-equation route F, trivial-zero ledger T, Gamma boundary G, transports H, routes C, provenance R, and local name N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket scoped route package`

Proof. Start with carrier admission and route-carrier admission. They identify the accepted object as the displayed finite packet W and make the eta, analytic-continuation, pole, functional-equation, trivial-zero, Gamma, transport, route, provenance, and naming rows available as packet coordinates.

Next apply classifier stability, pole-ledger exactness, continuation handoff, and trivial-zero non-escape. These results keep the classifier, continuation route, pole row, and trivial-zero row inside the same BHist witness surface.

Then use the Gamma boundary, critical-strip handoff, and package provenance theorems. They force Gamma-facing and critical-strip-facing consumers to read the same H, C, R, N rows before receiving the witness packet.

The local naming row closes the scoped package. No RH predicate, non-trivial-zero locator, completed Gamma certificate, or off-packet analytic branch is exported, because no such row occurs in W. □

Theorem 580.67 (ZetaContinuationWitness critical-strip source boundary)**.** *Every critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must read the source row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transports, continuation routes, provenance, and local name before receiving the packet. Equivalently, the consumer boundary is the finite route*

$$B, E, A, P, F, T, G, H, C, R, N$$

inside the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *package.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip source boundary`

Proof. Use source-lock coverage and route-carrier admission. They place the source row B, eta route E, and analytic-continuation row A at the entrance of every continuation-facing read.

Use pole-ledger exactness, the functional-boundary theorem, trivial-zero non-escape, and Gamma boundary control for the exceptional rows. These results require P, F, T, G to be read as ledger coordinates of the same packet.

Finally apply critical-strip handoff and package provenance exactness. The consumer receives the packet only after the displayed transports, routes, provenance, and local name have been read, so the boundary is exactly the listed finite route. □

Theorem 580.68 (ZetaContinuationWitness Gamma ledger exhaustion)**.** *For an accepted* ZetaContinuationWitness *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every Gamma-facing or pole-facing read is exhausted by the rows* P, F, G, H, C, R, N*. Such a read cannot import RH, a non-trivial-zero locator, a closed Gamma certificate, or a hidden analytic branch into the witness packet.*

Proof. Project the boundary coordinates of W. The pole row P, functional-equation route F, Gamma boundary G, transports H, routes C, provenance R, and local name N are the only rows available to a Gamma-facing or pole-facing consumer.

Apply the pole-Gamma boundary, functional-boundary scoped exactness, and trivial-zero non-escape theorems. They separate pole, functional-equation, Gamma, and trivial-zero data while keeping all four inside the displayed ledger.

Close with package provenance exactness and critical-strip refusal. Since R and N authenticate only the finite packet, a consumer cannot attach RH, a non-trivial-zero locator, a completed Gamma certificate, or another analytic branch to the Gamma-facing or pole-facing read. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessGammaLedgerExhaustion`

Theorem 580.69 (ZetaContinuationWitness critical-strip total ledger)**.** *For every accepted* ZetaContinuationWitnes *packet*

$$W = (B, E, A, P, F, T, G, H, C, R, N),$$

each critical-strip consumer must expose the complete finite packet rows

$$B, E, A, P, F, T, G, H, C, R, N$$

before it can consume the witness. The consumer cannot read an RH predicate, a non-trivial-zero locator, a global zero classifier, a hidden analytic branch, or a completed Gamma certificate from this packet.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessCriticalStripTotalLedger`

Proof. Begin with the critical-strip ledger triad. Theorems 580.33, 580.67, and 580.66 identify the consumer-facing route with the displayed source, eta, analytic-continuation, pole, functional-equation, trivial-zero, Gamma, transport, continuation, provenance, and naming rows of W.

Next exhaust the consumer side. Theorems 580.42, 580.63, and 580.65 force every downstream critical-strip read to preserve the same finite BHist, hsame, Cont, Pkg, and NameCert surface.

Finally apply Definition 580.28 and Definition 580.68. These two boundary theorems rule out RH data, non-trivial-zero locations, a global zero classifier, an off-packet analytic branch, and a completed Gamma certificate, because none is a displayed coordinate of W. □

Theorem 580.70 (ZetaContinuationWitness eta-pole-functional coverage)**.** *For every accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, each exported continuation read is exhausted by the eta row* E*, pole ledger* P*, functional-equation route* F*, trivial-zero ledger* T*, Gamma boundary* G*, and authentication rows* H, C, R, N*. The export contains only those continuation, boundary, transport, route, provenance, and naming rows and does not add an external analytic-continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket eta pole functional coverage`

Proof. Project the continuation-facing rows of W. Eta continuation is carried by E, pole data by P, the functional-equation route by F, trivial-zero data by T, and the Gamma boundary by G.

Use Definition 580.36, Definition 580.46, and Definition 580.64. Together they keep the eta route, pole ledger, and functional factor inside the displayed packet rather than allowing a separate analytic source.

Close with Definition 580.56 and Definition 580.68. The rows H, C, R, N authenticate only the same finite route, and the Gamma boundary contributes no completed Gamma certificate or hidden continuation branch. □

Theorem 580.71 (ZetaContinuationWitness RH boundary exhaustion)**.** *Every RH-facing consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receives only the displayed zeta-continuation witness rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

Any RH predicate, non-trivial-zero locator, or zero-location object must be supplied by a separate certificate outside this packet.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket rh boundary exhaustion`

Proof. Start with Definition 580.7. Together with Definition 580.28 and Definition 580.51, it states that the packet carries no RH predicate and no non-trivial-zero locator.

Next apply Definition 580.67 and Definition 580.33. They force an RH-facing consumer to read only the finite rows $B, E, A, P, F, T, G, H, C, R, N$ from the same BHist, hsame, Cont, Pkg, and NameCert surface.

Finally use Definition 580.68. Gamma-facing rows remain P, F, G, H, C, R, N and cannot be transformed into a completed Gamma certificate or zero-location object. Any such object therefore has to arrive as a separate certificate outside W. □

Theorem 580.72 (ZetaContinuationWitness source scoped readback)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every source-facing read factors through the rows* B, E, A, H, C, R, N*. The read cannot bypass the displayed source row, eta route, analytic-continuation row, componentwise transports, continuation routes, provenance, or local name.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket source scoped readback`

Proof. Begin with source-lock coverage. It places the source row B, eta route E, and analytic-continuation row A at the entrance of any source-facing read of the witness packet.

Apply the scoped route package. The route package keeps the displayed transports H, continuation routes C, provenance R, and local name N attached to the same packet before the read reaches any consumer.

Finally use Gamma ledger exhaustion to separate source readback from the exceptional Gamma and pole rows. Since the consumer route already factors through B, E, A, H, C, R, N, no off-packet analytic branch, RH predicate, or zero-location row can be imported into the source-facing read. □

Theorem 580.73 (ZetaContinuationWitness Gamma boundary scope)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every Gamma-facing boundary read is exhausted by the rows* P, F, G, H, C, R, N*. The boundary exposes the pole ledger, functional-equation route, Gamma row, transports, continuation routes, provenance, and local name, and it exposes no completed Gamma certificate outside those rows.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma boundary scope`

Proof. Apply the pole-Gamma boundary theorem. It identifies the pole ledger P, functional-equation route F, and Gamma boundary row G as the complete exceptional surface for Gamma-facing reads.

Use the scoped route package to attach the structural rows. The same read must pass through H, C, R, N, so the Gamma-facing boundary remains a finite packet route rather than a free analytic-continuation primitive.

Close with Gamma ledger exhaustion. It rules out RH, a non-trivial-zero locator, a closed Gamma certificate, and a hidden analytic branch as extra Gamma-facing coordinates. □

Theorem 580.74 (ZetaContinuationWitness critical-strip consumer scope)**.** *Every critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receives only the finite packet rows* $B, E, A, P, F, T, G, H, C$ *In particular, the consumer receives no RH predicate, non-trivial-zero locator, completed Gamma certificate, or off-packet analytic-continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip consumer scope`

Proof. Use the critical-strip handoff theorem first. It requires the consumer to read the displayed witness packet before any critical-strip-facing conclusion is available.

Apply the scoped route package to enumerate the packet surface. The only rows passed to the consumer are $B, E, A, P, F, T, G, H, C, R, N$, together with their BHist, hsame, Cont, Pkg, and NameCert structure.

Finally combine source-lock coverage and Gamma ledger exhaustion. Source-facing reads are confined to B, E, A, H, C, R, N, Gamma-facing reads are confined to P, F, G, H, C, R, N, and their union is the finite packet already displayed. No RH predicate or zero-location witness is a packet coordinate. □

Theorem 580.75 (ZetaContinuationWitness carrier row)**.** *Every accepted* ZetaContinuationWitness$^{\uparrow}$ *witness exposes the rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

as BHist*-carrier coordinates before any zeta-zero consumer reads the packet. The rows are the source, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transports, continuation routes, provenance, and local name of the same finite witness carrier.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket carrier row certificate`

Proof. Project the carrier of Definition 580.1. The accepted witness is the displayed packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, so each listed row is a carrier coordinate.

Apply Definition 580.74. Every critical-strip or zeta-zero-facing consumer receives only the displayed rows of that packet, and it receives them before any downstream read is available. □

Theorem 580.76 (ZetaContinuationWitness contour route)**.** *Every contour-facing analytic read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet factors through the displayed eta, analytic-continuation, functional-equation, pole, Gamma-boundary,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not factor through a hidden analytic branch, an RH predicate, a non-trivial-zero locator, or a completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket contour route obligation`

Proof. Begin with source-lock coverage. By Definition 580.4, a continuation read exposes B, E, A, F together with the exceptional ledgers before the packet is consumed.

Use the functional-boundary and Gamma-ledger theorems for the contour side. Definition 580.64 confines functional, pole, and Gamma reads to P, F, G, H, C, R, N, and Definition 580.68 rules out a completed Gamma or hidden analytic branch as an additional contour coordinate.

Finally apply Definition 580.7. The contour-facing route stays inside the same displayed witness package and therefore cannot export RH data or a non-trivial-zero locator. □

Theorem 580.77 (ZetaContinuationWitness obligation triad)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, the carrier row of Definition 580.75, the contour route of Definition 580.76, and the NameCert obligation surface of Definition 580.2 assemble the root-facing obligation packet expected by downstream zeta-zero chapters. The packet exposes only the rows $B, E, A, P, F, T, G, H, C, R, N$ and their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket obligation triad`

Proof. First use Definition 580.75. It supplies the finite carrier coordinates read by every zeta-zero-facing consumer.

Next use Definition 580.76. It identifies the analytic and contour-facing reads as routes through the displayed eta, continuation, functional, pole, Gamma, transport, continuation, provenance, and naming rows.

Finally apply Definition 580.2. The obligation surface binds carrier admission, classifier stability, analytic-continuation exactness, pole and trivial-zero ledger exactness, Gamma-boundary non-escape, package provenance, and local naming for the same packet, so the triad is exactly the downstream root-facing obligation packet and not an RH or zero-location claim. □

Theorem 580.78 (ZetaContinuationWitness public export)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet, the carrier row, contour route, Gamma ledger, critical-strip scope, provenance rows, and local* NameCert *rows assemble the public finite witness export. The export surface is exactly the displayed packet rows $B, E, A, P, F, T, G, H, C, R, N$ together with their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export`

Proof. Begin with Definition 580.75. It supplies the displayed finite carrier rows $B, E, A, P, F, T, G, H, C, R, N$ before any zeta-zero-facing consumer can read the witness.

Use Definition 580.76 and Definition 580.68. The contour and Gamma-facing reads are forced through the displayed eta, continuation, functional, pole, Gamma, transport, provenance, and local naming rows.

Finally combine Definition 580.74 with Definition 580.7. The critical-strip consumer receives only the public packet and no hidden RH predicate, zero locator, completed Gamma certificate, or off-packet analytic branch, so the finite export is the complete handoff surface. □

Theorem 580.79 (ZetaContinuationWitness zero-route input package)**.** *Every* $\mathsf{ZetaZeros}^{\uparrow}$ *or critical-strip consumer reading an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet receives only the rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

and their displayed BEDC structure. No RH predicate, non-trivial-zero locator, completed Gamma certificate, or hidden analytic branch is part of the input package.

Proof. Apply Definition 580.74. It already confines every critical-strip consumer to the listed rows and their displayed BHist, hsame, Cont, Pkg, and NameCert data.

Use Definition 580.75 to read the same list as the accepted witness carrier and Definition 580.76 to route analytic and contour-facing input through the displayed continuation packet.

Finally invoke Definition 580.7. The consumer can receive the public route packet, but it cannot receive RH proof data, a non-trivial-zero locator, or an undisplayed analytic continuation coordinate. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero route input package`

Theorem 580.80 (ZetaContinuationWitness Gamma boundary public non-escape)**.** *Every Gamma-facing public read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet lands at the displayed Gamma boundary row* G*, unless the same read is already displayed as a factor of the functional-equation route row* F*. It cannot escape to a completed Gamma certificate, a hidden analytic branch, an RH predicate, or a non-trivial-zero locator.*

Proof. Start with Definition 580.68. It identifies the Gamma ledger as the displayed Gamma-boundary coordinate and rules out a completed Gamma certificate or hidden analytic branch as an extra public row.

Apply Definition 580.76. Any contour-facing read that reaches Gamma data must pass through the displayed functional-equation route F and Gamma-boundary row G.

Close with Definition 580.7 and Definition 580.74. They keep the public read inside $B, E, A, P, F, T, G, H, C, R, N$ and exclude RH proof data or zero-location data from the Gamma-facing boundary. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma boundary public nonescape`

Theorem 580.81 (ZetaContinuationWitness public zero-route factorization)**.** *Every public zero-facing read of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet factors through the displayed rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The factorization may expose the source row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transports, continuations, provenance, and local name, but it exposes no RH predicate, non-trivial-zero locator, global zero classifier, off-packet analytic branch, or completed Gamma certificate.

Proof. Begin with Definition 580.78. It assembles the carrier row, contour route, Gamma ledger, critical-strip scope, provenance, and local NameCert rows into one public export.

Apply Definition 580.79. The zero-facing input package is exactly the displayed tuple $B, E, A, P, F, T, G, H, C, R$ together with its BHist, hsame, Cont, Pkg, and NameCert structure.

Finally use Definition 580.86 and Definition 580.87. They keep the public export inside the same rows and remove RH predicates, non-trivial-zero locators, global zero classifiers, completed Gamma certificates, and off-packet analytic branches from the zero-facing handoff. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPublicZeroRouteFactorization`

Theorem 580.82 (ZetaContinuationWitness public contour-Gamma separation)**.** *Every public contour-facing read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet that reaches Gamma data separates into the contour route through $B, E, A, P, F, H, C, R, N$ and the Gamma boundary row G, unless the requested Gamma datum is already displayed as part of the functional-equation route F. No such read exports an RH predicate, non-trivial-zero locator, global zero classifier, off-packet analytic branch, or completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public contour gamma separation`

Proof. Start with Definition 580.80. It forces every Gamma-facing public read to land at G unless the same datum is already displayed inside F.

Use Definition 580.86. The contour-facing public export still factors through $B, E, A, P, F, T, G, H, C, R, N$, so the Gamma side cannot introduce a row outside the witness packet.

Close with Definition 580.87. The same public boundary excludes RH predicates, zero locators, global zero classifiers, completed Gamma certificates, and off-packet analytic branches from the contour-Gamma handoff. □

Theorem 580.83 (ZetaContinuationWitness public source-lock handoff)**.** *Every public critical-strip, zero-facing, or contour-facing consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must receive the source-locked rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

before using the packet as continuation evidence. The handoff is therefore a finite source-lock export over the displayed rows and their BEDC structure, not an RH predicate, non-trivial-zero locator, global zero classifier, off-packet analytic branch, or completed Gamma certificate.

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public source lock handoff`

Proof. First apply Definition 580.78. It supplies the public witness export as a finite packet, including the carrier, contour route, Gamma boundary, provenance, and local name.

Next apply Definition 580.79. Zero-facing and critical-strip consumers read the packet only through $B, E, A, P, F, T, G, H, C, R, N$.

Finally combine Definition 580.80, Definition 580.86, and Definition 580.87. These rows keep Gamma-facing and contour-facing reads source-locked inside the same packet and rule out RH predicates, zero locators, global zero classifiers, off-packet analytic branches, and completed Gamma certificates. □

Theorem 580.84 (ZetaContinuationWitness public export route lock)**.** *Every public critical-strip, zero-facing, contour-facing, or Gamma-facing consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet must first read the source-locked rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

and their BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. Agreement on those rows and support data determines the same public export route.*

The route lock exports no RH predicate, no non-trivial-zero locator, no global zero classifier, no off-packet analytic branch, and no completed Gamma certificate.

Proof. Start with Definition 580.75, Definition 580.76, and Definition 580.77. These results put the carrier coordinates, contour-facing route, and root-facing obligation packet on the same displayed rows $B, E, A, P, F, T, G, H, C, R, N$.

Next use Definition 580.78 and Definition 580.79. The public export and zero-route input package say that critical-strip and zero-facing consumers receive only the finite witness packet and its displayed BEDC support.

For Gamma-facing and contour-facing reads, apply Definition 580.80, Definition 580.81, and Definition 580.82. These theorems keep Gamma data and contour data on the same public rows and remove off-packet analytic branches, completed Gamma certificates, RH predicates, non-trivial-zero locators, and global zero classifiers.

Finally invoke Definition 580.83. Since every public consumer must receive the source-locked packet before using it as continuation evidence, two accepted packets that agree on the rows and support data determine the same public export route. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export route lock`

Theorem 580.85 (ZetaContinuationWitness public export totality)**.** *Every public critical-strip, zero-facing, contour-facing, or Gamma-facing read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet is exhausted by the single packet*

$$B, E, A, P, F, T, G, H, C, R, N$$

and its BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The public export contains no RH predicate, non-trivial-zero locator, global zero classifier, off-packet analytic branch, or completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export totality`

Proof. Begin with the public export theorem. It assembles the carrier row, contour route, Gamma ledger, critical-strip scope, provenance rows, and local NameCert rows into one public witness packet.

Apply the zero-route input package. It says zero-facing and critical-strip reads receive exactly $B, E, A, P, F, T, G, H, C, R, N$ together with the displayed BEDC support.

For contour-facing and Gamma-facing reads, use public contour-Gamma separation. The contour route and Gamma boundary remain inside the same packet, with Gamma data either read from G or already displayed inside F.

Finally apply the route-lock theorem. Since every public read is source-locked to the same packet before it is used as continuation evidence, there is no remaining coordinate for RH predicates, non-trivial-zero locators, global zero classifiers, off-packet analytic branches, or completed Gamma certificates. □

Theorem 580.86 (ZetaContinuationWitness public export factorization)**.** *For an accepted* ZetaContinuationWitnes *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every public export in the sense of Definition 580.78, and every zero-facing public input package in the sense of Definition 580.79, factors through the displayed rows* $B, E, A, P, F, T, G, H, C, R, N$ *and their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure. The export contains no route outside the source row, eta route, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, transport rows, continuation routes, provenance, and local name.*

Proof. Begin with the public export packet of Definition 580.78 and the zero-facing public input package of Definition 580.79. They already assemble the carrier row, contour route, Gamma ledger, critical-strip scope, provenance rows, and local NameCert rows as one finite witness export.

Next use the carrier and contour rows. Definition 580.75 supplies $B, E, A, P, F, T, G, H, C, R, N$ as the packet coordinates, while Definition 580.76 keeps analytic and contour reads inside the displayed continuation packet.

Finally apply the source-lock and provenance exactness rows. They attach every public consumer to the same BHist, hsame, Cont, Pkg, and NameCert surface, so no off-packet analytic branch or hidden witness row can be part of the export. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export factorization`

Theorem 580.87 (ZetaContinuationWitness critical-strip public refusal)**.** *Every public critical-strip consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet receives the finite witness rows* $B, E, A, P, F, T, G, H, C, R, N$ *and receives no RH predicate, non-trivial-zero locator, global zero classifier, completed Gamma certificate, or off-packet analytic-continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket critical strip public refusal`

Proof. Apply Definition 580.79. It identifies the public input package for ZetaZeros$^{\uparrow}$ and critical-strip consumers as the displayed witness rows together with their BEDC structure.

Use Definition 580.11 to remove the zero-facing escape routes. The consumer has no packet coordinate for an RH predicate, a non-trivial-zero locator, or a global zero classifier.

Close with Definition 580.80. Gamma-facing data either remains at the displayed boundary or is already present in the functional-equation route, so no completed Gamma certificate or hidden analytic branch is exported to the critical strip. □

Theorem 580.88 (ZetaContinuationWitness Gamma boundary public ledger)**.** *Every Gamma-facing public read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet stops at the displayed rows* P, F, G, H, C, R, N*, unless the requested Gamma datum is already displayed as part of the functional-equation route* F*. The public ledger therefore exposes pole, functional-equation, Gamma-boundary, transport, continuation, provenance, and local naming rows, and it exposes no completed Gamma certificate outside that route.*

Proof. Start with Definition 580.80. It states the public Gamma boundary condition: a Gamma-facing read lands at G unless the same data is already displayed inside F.

Next use Definition 580.73. The boundary read is exhausted by P, F, G, H, C, R, N, so the structural rows that authenticate the Gamma side are exactly the displayed transports, routes, provenance, and local name.

Finally apply Definition 580.86. Since the public export itself factors through $B, E, A, P, F, T, G, H, C, R, N$, the Gamma-facing subroute cannot acquire an additional completed Gamma certificate, RH predicate, zero locator, or hidden analytic branch. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessGammaBoundaryPublicLedger`

Theorem 580.89 (ZetaContinuationWitness public export route)**.** *For an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every public root-facing export factors through the finite contour, Gamma, critical-strip, provenance, and local naming route*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The route exposes the source row, eta-continuation row, analytic-continuation row, pole ledger, functional-equation route, trivial-zero ledger, Gamma boundary, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, and it exports no RH predicate, non-trivial-zero locator, host complex equality, off-packet analytic branch, or completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export route`

Proof. Begin with Definition 580.78. It assembles the carrier row, contour route, Gamma ledger, critical-strip scope, provenance rows, and local NameCert rows into one public finite witness export.

Use Definition 580.86 and Definition 580.12. These results force every public root-facing read through the displayed carrier coordinates and the same BHist, hsame, Cont, Pkg, and NameCert surface.

Finally apply Definition 580.87 and Definition 580.14. Critical-strip and Gamma-facing consumers receive only the finite route above, while RH data, zero-location data, host complex equality, completed Gamma data, and off-packet analytic branches remain outside the export. □

Theorem 580.90 (ZetaContinuationWitness consumer packet totality)**.** *Every* $\mathsf{ZetaZeros}^{\uparrow}$*, critical-strip, Gamma-facing, or public root-facing consumer of an accepted* $\mathsf{ZetaContinuationWitness}^{\uparrow}$ *packet is total over the displayed packet rows*

$$B, E, A, P, F, T, G, H, C, R, N$$

and their BHist, hsame, Cont, Pkg, *and* NameCert *structure. No accepted consumer route remains outside the source, contour, pole, trivial-zero, Gamma, transport, continuation, provenance, and local naming rows of the same packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket consumer packet totality`

Proof. First project the consumer input package by Definition 580.79. It identifies the rows received by zero-facing and critical-strip consumers as the displayed packet coordinates with their BEDC structure.

Next use Definition 580.74 and Definition 580.89. The critical-strip route and the public root-facing export route are the same finite packet surface, not two independent consumer packets.

Finally attach Definition 580.77 and Definition 580.86. The NameCert obligation surface, contour route, carrier row, provenance row, and local name exhaust the consumer-visible packet, so there is no admitted route left to classify outside the displayed rows. □

Theorem 580.91 (ZetaContinuationWitness analytic boundary non-escape)**.** *An accepted* ZetaContinuationWitness *public export keeps RH, non-trivial-zero location data, analytic continuation as an external analytic premise, and host complex equality outside the consumer packet. The only analytic boundary visible to the packet is the finite route through* $B, E, A, P, F, T, G, H, C, R, N$ *and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessAnalyticBoundaryNonescape`

Proof. Start with the analytic and Gamma boundaries. The contour route of Definition 580.76 and the Gamma boundary public ledger of Definition 580.88 restrict analytic-facing reads to the displayed continuation, pole, functional-equation, Gamma, transport, provenance, and naming rows.

Use Definition 580.7 and Definition 580.11. These theorems exclude RH predicates, non-trivial-zero locators, global zero classifiers, and completed Gamma certificates from the packet inventory.

Finally apply Definition 580.90. Since every accepted consumer route has already been exhausted by the displayed finite packet, analytic continuation outside the packet and host complex equality have no consumer-visible coordinate from which to enter. □

Theorem 580.92 (ZetaContinuationWitness public source-lock export)**.** *For an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, every continuation consumer of the public source-lock export reaches the continuation surface only through the finite rows*

$$B, E, A, P, F, T, G, H, C, R, N.$$

The export therefore routes source access through the ZetaBasic$^{\uparrow}$ *source row* B*, eta-continuation row* E*, analytic-continuation row* A*, pole ledger* P*, functional-equation route* F*, trivial-zero ledger* T*, Gamma boundary* G*, transport rows* H*, continuation routes* C*, provenance* R*, and local name* N*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public source lock export`

Proof. Begin with the public export route of Definition 580.89. It already makes the finite packet $B, E, A, P, F, T, G, H, C, R, N$ the only public root-facing route.

Apply the source scoped readback of Definition 580.72. The source-facing part of the route must pass through B, E, A, H, C, R, N, so no continuation consumer can enter through an off-packet analytic branch.

Finally use public export factorization. The pole, functional-equation, trivial-zero, and Gamma rows remain part of the same packet surface, so the public source-lock export has one displayed route for every continuation consumer. □

Theorem 580.93 (ZetaContinuationWitness public critical-strip export)**.** *For an accepted* ZetaContinuationWitness *packet, the public critical-strip export supplies only the finite witness packet*

$$B, E, A, P, F, T, G, H, C, R, N$$

with its BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure. It supplies no RH predicate, no non-trivial-zero locator, no global zero classifier, and no completed Gamma certificate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public critical strip export`

Proof. Use the critical-strip consumer scope of Definition 580.74. It confines every critical-strip consumer to the displayed witness packet.

Next apply the public critical-strip refusal theorem of Definition 580.87. The public route has no coordinate for RH, a non-trivial-zero locator, a global zero classifier, or a completed Gamma certificate.

The packet structure is then the one named by public export factorization. Its BHist, hsame, Cont, Pkg, and NameCert rows authenticate the finite export without adding any zero-location datum. □

Theorem 580.94 (ZetaContinuationWitness public root completion)**.** *An accepted* ZetaContinuationWitness$^{\uparrow}$ *public root surface is exactly the downstream packet obtained by packaging the source-lock export, the critical-strip export, the Gamma boundary, and the no-RH non-escape condition over the rows* $B, E, A, P, F, T, G, H, C, R, N$*. The surface is a finite downstream root witness route, not an RH theorem, zero locator, global analytic-continuation branch, or host complex equality principle.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public root completion`

Proof. First combine the source-lock export of Definition 580.92 with the critical-strip export of Definition 580.93. Together they make the same finite packet the route for continuation consumers and critical-strip consumers.

Add the Gamma boundary public ledger of Definition 580.88. Gamma-facing reads stop at P, F, G, H, C, R, N, unless the requested datum is already part of the displayed functional-equation route.

Finally use analytic boundary non-escape and no-RH non-escape. RH predicates, zero locators, off-packet analytic branches, completed Gamma certificates, and host complex equality have no packet coordinate, so the packaged downstream root surface is exactly the finite route displayed above. □

Theorem 580.95 (ZetaContinuationWitness public export lock)**.** *For every accepted* ZetaContinuationWitness$^{\uparrow}$ *packet* $W = (B, E, A, P, F, T, G, H, C, R, N)$*, the public export of* W *is exactly the finite row packet*

$$B, E, A, P, F, T, G, H, C, R, N$$

together with the dependency rows ZetaBasic$^{\uparrow}$*,* ZetaCont$^{\uparrow}$*,* AnaCont$^{\uparrow}$*,* Holomorphic$^{\uparrow}$*,* Complex$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. No public export row lies outside this displayed witness packet and dependency ledger.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket public export lock`

Proof. Project the carrier of Definition 580.1. The accepted object is the finite packet $W = (B, E, A, P, F, T, G, H, C, R, N)$, so the public export has no carrier coordinate beyond those rows.

Apply Definition 580.2, Definition 580.8, and Definition 580.4. These bind the source, continuation, analytic, holomorphic, complex, history, transport, route, package, and naming dependencies to the same packet before any public consumer receives it.

Finally use Definition 580.78, Definition 580.86, and Definition 580.12. The public packet, closure surface, and root export all read the same rows, so the export is precisely the displayed finite packet with the listed dependency rows. □

Theorem 580.96 (ZetaContinuationWitness zero-facing public factorization)**.** *Every zero-facing public consumer of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet factors through the source lock, eta route, pole ledger, functional-equation route, trivial-zero ledger, and Gamma boundary before it can use the packet as input for* ZetaZeros$^{\uparrow}$ *or any critical-strip read. The factorization exposes only the rows* $B, E, A, P, F, T, G, H, C, R, N$ *and their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket zero facing public factorization`

Proof. Begin with Definition 580.4 and Definition 580.92. They require a public consumer to pass through the source row B, eta route E, analytic-continuation row A, and functional-equation row F before the witness is consumed.

Next apply Definition 580.46, Definition 580.51, and Definition 580.88. These theorems keep the pole ledger P, trivial-zero ledger T, and Gamma boundary G as displayed rows of the same packet.

Close with Definition 580.79, Definition 580.81, and Definition 580.87. The zero-facing and critical-strip handoff therefore factors through the source lock and the exceptional ledgers before any downstream use, and no RH predicate, non-trivial-zero locator, global zero classifier, completed Gamma certificate, or off-packet analytic branch is exported. □

Theorem 580.97 (ZetaContinuationWitness Gamma public boundary lock)**.** *Every public Gamma-facing read of an accepted* ZetaContinuationWitness$^{\uparrow}$ *packet stops at the functional-equation route* F *or the Gamma boundary row* G*. Such a read does not supply a completed Gamma chapter, an RH predicate, a non-trivial-zero locator, a global zero classifier, or any zero-location object outside the displayed witness packet.*

Proof. Start with Definition 580.5 and Definition 580.80. A Gamma-facing public read either uses Gamma data already displayed inside F or stops at the boundary row G.

Use Definition 580.68 and Definition 580.14. These exhaust the Gamma-facing public ledger at P, F, G, H, C, R, N and prevent the boundary from becoming a completed Gamma chapter or hidden analytic branch.

Finally apply Definition 580.7, Definition 580.87, and Definition 580.71. The public Gamma boundary remains a finite packet row and supplies no RH predicate, non-trivial-zero locator, global zero classifier, or zero-location object. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationWitnessUp.ZetaContinuationWitnessPacket gamma public boundary lock`

Chapter 581

A Concrete Naming Certificate for $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$

The $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$ packet gives the functional-equation part of chapter 9342 a concrete finite NameCert surface. It records eta continuation, the displayed functional-equation factor row, the $\mathsf{Gamma}^{\uparrow}$ boundary, contour-integral provenance, and the trivial-zero witness as rows that downstream zeta and Gamma chapters can inspect without claiming a closed Gamma proof, Riemann-hypothesis consequence, or host complex-equality theorem.

Definition 581.1 (Zeta functional-equation ledger carrier)**.** *A* $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Z, A, E, F, G, K, T, H, C, P, N)$$

with the following displayed rows:

1. *Z is the* $\mathsf{ZetaBasic}^{\uparrow}$ *source row on the half-plane* $\Re(s) > 1$*;*
2. *A is the* $\mathsf{AnaCont}^{\uparrow}$ *handoff row that reads the eta-extension step and the positive-half-plane continuation;*
3. *E is the eta-extension row, including the displayed denominator boundary for* $1 - 2^{1-s}$ *and the overlap agreement with* Z*;*
4. *F is the functional-equation factor row containing* 2^s*,* π^{s-1}*,* $\sin(\pi s/2)$*, the* $\Gamma(1-s)$ *slot, and the reflected* $\zeta(1-s)$ *slot;*
5. *G is the* $\mathsf{Gamma}^{\uparrow}$ *boundary row, recording exactly the input domain, pole boundary, and nonzero-at-positive-integer reads consumed by the functional-equation ledger;*
6. *K is the* $\mathsf{ContourIntegral}^{\uparrow}$ *provenance row for the contour reorganization used to justify the displayed factor row;*
7. *T is the trivial-zero witness row for* $s = -2n$ *with unary* $n \geq \mathsf{E1}(\mathsf{e})$*, storing the sine-zero row and the finiteness reads for the remaining factors;*
8. *H stores componentwise* hsame *transports for* Z, A, E, F, G, K, T *and the local naming rows;*
9. *C stores the finite* Cont *routes by which consumers read eta continuation, factor decomposition, Gamma boundary, contour provenance, trivial-zero consumption, package provenance, and local naming;*

10. *P is the* Pkg *provenance over the same finite ledger packet;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}}$ *row.*

The carrier exposes only this ledger packet. It contains no closed proof of the Gamma factor, no full meromorphic continuation theorem, no quotient of germs, and no host equality principle for complex functions.

Definition 581.2 (Zeta functional-equation ledger classifier)**.** *Two* $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$ *carriers are classified by componentwise* hsame *on the displayed analytic rows Z, A, E, F, G, K, T, exact agreement of the finite* Cont *routes in C, and the same* Pkg *and local* NameCert *rows. The classifier treats the* $\mathsf{Gamma}^{\uparrow}$ *coordinate as a boundary row: it may transport the displayed domain and consumption reads, but it does not identify two host Gamma functions or close the missing meromorphic proof.*

Theorem 581.3 (Zeta functional-equation ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZetaFunctionalEquationLed}}$ *surface for Definition 581.1 consists of five finite rows: carrier admission for L; classifier stability for the zeta source, eta continuation, functional-equation factor, Gamma boundary, contour provenance, and trivial-zero witness rows; ledger exactness for reads exported through C;* Pkg *provenance over $Z, A, E, F, G, K, T, H, C$; and local naming exactness for N.*

Proof. Carrier admission is projection of the tuple L. Each zeta, continuation, eta, factor, Gamma-boundary, contour, trivial-zero, transport, continuation, package, and naming coordinate is a displayed row of the same finite BHist packet.

Classifier stability is componentwise. Transport through H preserves the source row Z, the eta-extension row E, the analytic-continuation handoff A, the factor row F, the Gamma boundary G, the contour provenance K, and the trivial-zero witness T under their stated classifiers.

Ledger exactness is read through C. Every exported consumer fact lands in one of the displayed coordinates, and any request for a closed Gamma theorem, a global meromorphic-continuation theorem, or host complex equality has no row in L.

The package and local naming rows repack the same finite coordinates. Thus the obligation surface is exactly the five rows stated in the theorem. □

Theorem 581.4 (Gamma boundary for the zeta functional-equation ledger)**.** *For an accepted* $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$ *carrier $L = (Z, A, E, F, G, K, T, H, C, P, N)$, every exported use of $\Gamma(1-s)$ factors through the displayed* $\mathsf{Gamma}^{\uparrow}$ *boundary row G and the functional-equation factor row F. The ledger exports the domain and consumption reads needed by the factor row and the trivial-zero witness, but it does not export a closed Gamma construction or a host meromorphic equality proof.*

Proof. Begin with a consumer read of the Gamma slot in the functional-equation row. The route must pass through C, so it lands either in the factor row F or in the boundary row G.

If it lands in F, the read sees only the formal factor slot $\Gamma(1-s)$ as part of the displayed product. If it lands in G, the read sees the boundary information: the input-domain row, the pole-exclusion row, and the finite nonzero reads used at positive integers.

No other coordinate of L carries a Gamma construction. The classifier of Definition 581.2 therefore keeps Gamma use at the boundary level and prevents the ledger from exporting a closed Gamma proof. □

Theorem 581.5 (Trivial-zero seal for the zeta functional-equation ledger)**.** *For an accepted* ZetaFunctionalEquation *carrier, the trivial-zero witness row T consumes exactly the sine-zero coordinate at $s = -2n$ with*

unary $n \geq \mathsf{E1}(\mathsf{e})$, the finite Gamma-boundary reads from G, the reflected zeta read from Z through A and E, and the factor decomposition in F. It seals only the displayed negative even zero rows and does not assert any zero-location statement in the critical strip.

Proof. Read the witness row T. Its sine-zero coordinate is the constructive row for $\sin(\pi(-2n)/2) = 0$, and this coordinate is already part of the factor decomposition F.

The remaining factor reads are routed through the displayed ledger. Gamma finiteness and nonzero consumption pass through G, while the reflected zeta input is read through the zeta source and eta-continuation rows Z, A, E.

The row T has no coordinate for the critical strip and no classifier row for Riemann-hypothesis strength. Hence the seal closes only the displayed negative even zero witnesses consumed by this ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}$ packet is generated from $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{AnaCont}^{\uparrow}$, eta-extension, functional-equation factor, $\mathsf{Gamma}^{\uparrow}$ boundary, $\mathsf{ContourIntegral}^{\uparrow}$ provenance, trivial-zero witness, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 581.1, Definition 581.2, Definition 581.3, Definition 581.4, and Definition 581.5 define the local ledger carrier, classifier, obligation rows, Gamma boundary, and trivial-zero seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZetaFunctionalEquationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Gamma proof, RH, full meromorphic continuation, host complex equality, Quot, Classical choice, propext, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, Gamma-boundary exactness, trivial-zero consumption, package provenance, and local naming for the same finite ledger packet.

Chapter 582

A Concrete Naming Certificate for $\mathsf{InscribedRoute}^{\uparrow}$

The $\mathsf{InscribedRoute}^{\uparrow}$ surface isolates the reusable proof route carried by an inscription event. It names the finite path from a visible gap through a public statement, a displayed proof route, an accepted record row, and a downstream consumer, while leaving private discovery, host trust, and unrecorded choice outside the packet.

It consumes the inscription-event carrier of chapter 519 and the visible gap ledger of chapter 520. The source scope and visible gap rows come from those sibling certificates, while this chapter packages the route, acceptance row, downstream consumer row, and public GAP ledger as the reusable route certificate.

Definition 582.1 (InscribedRoute carrier)**.** *An* $\mathsf{InscribedRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, G, M, R, A, U, L, P, N)$$

with the following displayed rows:

1. *S is the prior* BHist *source scope from which the route is read;*
2. *G is the visible gap row naming the local theorem, certificate, or consumer boundary that still requires a route;*
3. *M is the named statement row, fixing the public binders, hypotheses, and conclusion of the reusable route;*
4. *R is the proof-route row, recording the displayed constructor, projection, inversion, transitivity, symmetry, or transport steps that move from S and G to M;*
5. *A is the acceptance row, recording that the route is admitted as public record data;*
6. *U is the downstream consumer row naming the theorem site, citation, audit surface, or certificate boundary that may read the route;*
7. *L is the public* GAP *ledger row tying the visible gap, named statement, route, acceptance row, and consumer row to the same finite route packet;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, *route-checking data,* GAP *ledger data, and local certificate dependencies;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only $S, G, M, R, A, U, L, P, N$ and their displayed hsame *and* Cont *transports. It has no coordinate for hidden psychology, host trust, or an unrecorded route selector.*

Lean4 ✓ `BEDC.Derived.InscribedRouteUp.InscribedRouteTasteGate single carrier alignment`

Theorem 582.2 (InscribedRoute NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}$ *surface for the carrier of Definition 582.1 consists of five finite obligations:*

1. *carrier admission: every accepted route exposes exactly the packet $Q = (S, G, M, R, A, U, L, P, N)$;*
2. *route classifier stability: public comparison is restricted to the source scope, visible gap, named statement, proof route, acceptance row, downstream consumer, ledger, package provenance, local naming row, and displayed* hsame *and* Cont *routes;*
3. *proof-route determinacy: the route row R reaches M from S and G only through the displayed* Cont *path and the matching* hsame *rows;*
4. *acceptance-ledger exactness: the acceptance row A and* GAP *ledger row L record exactly the admitted route and the downstream consumer boundary that may cite it;*
5. *non-escape: the packet exports no hidden psychology, host trust predicate, or unrecorded choice of route.*

Lean4 ✓ `BEDC.Derived.InscribedRouteUp.InscribedRouteNameCertObligations`
↪ variant `BEDC.Derived.InscribedRouteUp.InscribedRouteCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 582.1. The source scope, visible gap, named statement, proof route, acceptance row, downstream consumer, ledger, package provenance, and local naming row are all displayed before the route can be consumed.

Classifier stability and proof-route determinacy are read from the displayed comparison surface. The classifier has access only to $S, G, M, R, A, U, L, P, N$, and the route from R to M is replayed through the recorded Cont path together with the matching hsame rows.

Acceptance-ledger exactness follows because A and L are coordinates of the same finite packet. The accepted route and the downstream consumer boundary are therefore ledgered in public data rather than inferred from a host-side process.

The non-escape clause is immediate from the carrier inventory. No row stores hidden psychology, host trust, or an unrecorded choice, so none of these can be exported by the local $\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}$ surface. □

Theorem 582.3 (InscribedRoute carrier admission)**.** *Every accepted* $\mathsf{InscribedRoute}^{\uparrow}$ *carrier exposes exactly the rows*

$$S, G, M, R, A, U, L, P, N$$

before any route consumer can read it. No accepted carrier has an additional coordinate for hidden host trust, private route choice, or external proof memory.

Proof. Project the carrier of Definition 582.1. The tuple displayed there is exactly $Q = (S, G, M, R, A, U, L, P, N)$, so acceptance can expose only those nine rows.

The classifier sentence following the row inventory confirms that public comparison is restricted to the same rows and their displayed hsame and Cont transports. Since the carrier lists no further coordinate, hidden host trust, private route choice, and external proof memory cannot be admitted as carrier data. □

Theorem 582.4 (InscribedRoute continuation obligation)**.** *For every accepted* $\mathsf{InscribedRoute}^{\uparrow}$ *packet, the route row R reaches the named statement row M through the displayed* Cont *path and the matching* hsame *rows recorded in the same carrier. The continuation obligation does not use an unrecorded route selector or a host-side proof search.*

Proof. Use the proof-route determinacy clause of Definition 582.2. It states that R reaches M from S and G only through the displayed Cont path and matching hsame rows.

Carrier admission supplies the ambient packet $Q = (S, G, M, R, A, U, L, P, N)$. Because R, M, P, and the displayed transport data are all rows of the same packet, the continuation can be replayed by the recorded data before the downstream consumer U reads it. □

Theorem 582.5 (InscribedRoute NameCert ledger obligation)**.** *For every accepted* $\mathsf{InscribedRoute}^{\uparrow}$ *packet, the acceptance row A and the public* GAP *ledger row L bind the same downstream consumer row U. A consumer may cite the route only when acceptance, ledger entry, named statement, and route row are displayed in the same finite packet.*

Proof. Apply the acceptance-ledger exactness clause of Definition 582.2. It identifies A and L as coordinates of the same route packet and requires them to record the admitted route and downstream consumer boundary.

Now project the carrier. The rows M, R, A, U, L occur together in Q, so the named statement, route, acceptance row, consumer row, and public ledger row are not separate host facts. The downstream consumer can therefore cite the route only through the single displayed packet. □

Theorem 582.6 (InscribedRoute public export)**.** *For every accepted* $\mathsf{InscribedRoute}^{\uparrow}$ *packet, the public reusable route exported to a downstream consumer is exactly the displayed finite route*

$$S, G, M, R, A, U, L, P, N.$$

The export makes the source scope, visible gap, named statement, proof route, acceptance row, downstream consumer row, public ledger, provenance, and local NameCert *row visible together; it exports no hidden host trust predicate, private route selector, or external proof memory.*

Proof. Carrier admission in Definition 582.3 identifies the accepted packet with the displayed tuple $S, G, M, R, A, U, L, P, N$. Thus any public export begins with exactly those rows.

The NameCert obligations of Definition 582.2 bind the proof-route determinacy and acceptance-ledger exactness clauses to that same tuple. The continuation obligation of Definition 582.4 then replays R to M only through the displayed Cont path and matching hsame rows.

Finally Definition 582.5 ties the acceptance row A, downstream consumer row U, and public ledger row L together. Therefore the public consumer can cite the route precisely as the finite BEDC packet above, and no unrecorded host-side route data are exported. □

Theorem 582.7 (InscribedRoute standard bridge surface)**.** *The public reusable route of an accepted* $\mathsf{InscribedRoute}^{\uparrow}$ *packet can be read as a standard bridge surface in the sense of chapter 9204: the source scope row S names the BEDC carrier, the visible gap row G and named statement row M give the encode/decode surface, the proof route row R records the finite route, the acceptance row A and consumer row U give the public-object surface, the public ledger row L records the bridge-readable export, and the provenance and naming rows P, N keep the route local to the packet. This bridge is a paper-level witness only; it does not claim machine bridge checking.*

Proof. Start from the public export theorem of Definition 582.6. It identifies the displayed finite route $S, G, M, R, A, U, L, P, N$ as the export surface for downstream consumers.

Read the same route through the standard bridge protocol of chapter 9204. The source scope, visible gap, named statement, proof route, acceptance row, downstream consumer, and public ledger provide the encode/decode and theorem-preservation shape required for a paper bridge.

The bridge remains paper-level because the chapter does not add any host translation module or machine no-host-leak witness. It therefore presents the public route as a bridge surface while keeping the verification axis separate. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}$.

Constructive story: An $\mathsf{InscribedRoute}^{\uparrow}$ packet is generated from a prior source scope, visible gap row, named statement, proof route, acceptance row, downstream consumer, public GAP ledger, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 582.1, Definition 582.2, Definition 582.3, Definition 582.4, Definition 582.5, Definition 582.6, and Definition 582.7 bind source-scope, visible-gap, route-witness, acceptance-ledger, downstream-consumer, public export, standard bridge surface, hsame/Cont transport, local NameCert scoping, and the public export surface for reusable inscribed routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscribedRoute}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.InscribedRouteUp.InscribedRouteNameCertObligations`

Bridge status: paperBridge

Not claimed: This chapter does not close machine bridge checking, complete inscription semantics, or hidden host route semantics.

Upgrade path: The paper bridge surface is mature; machine bridge checking remains a separate verification-axis obligation.

Chapter 583

A Concrete Naming Certificate for $\mathsf{ContourIntegralSocket}^{\uparrow}$

The $\mathsf{ContourIntegralSocket}^{\uparrow}$ interface turns the contour-integral operation surface of chapter 9340 into a finite BEDC operation socket. It names piecewise-linear contour rows, holomorphic input rows, Riemann-sum modulus rows, complex output rows, componentwise transport, continuation routes, package provenance, and a local naming ledger for consumers that need contour integration without a host complex-analysis oracle.

Definition 583.1 (Contour-integral socket carrier)**.** *A* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, F, M, I, L, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{PLContour}$ *row, including endpoints, segment list, concatenation boundary, reversal boundary, and reparameterization witness rows;*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row on a displayed* $\mathsf{Complex}^{\uparrow}$ *neighbourhood of the contour image;*
3. *M is the Riemann-sum modulus row, carrying tagged-partition refinement data and the convergence modulus used to read the integral;*
4. *I is the* $\mathsf{Complex}^{\uparrow}$ *output row naming the contour-integral value;*
5. *L is the operation-law ledger for linearity, concatenation, reversal, reparameterization, Cauchy triangle, polygon subdivision, and Cauchy integral formula rows;*
6. *H stores componentwise* hsame *transports for G, F, M, I, L and for the local naming rows;*
7. *C stores the finite* Cont *routes by which analytic-continuation and Cauchy-operation consumers read the contour, input, modulus, output, operation ledger, package provenance, and local naming row;*
8. *P is the* Pkg *provenance over the same finite operation-socket packet;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourIntegralSocket}^{\uparrow}}$ *ledger.*

The carrier exposes only this finite socket. It contains no host contour-integral function, no quotient path space, no use of $\mathsf{Classical.choice}$*, and no undisplayed completeness principle.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketTasteGate single carrier alignment`

Theorem 583.2 (Contour-integral socket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContourIntegralSocket}^{\uparrow}}$ *surface for Definition 583.1 consists of five finite rows: carrier admission for S; classifier stability for the* PLContour *row, the* $\mathsf{Holomorphic}^{\uparrow}$ *input row, the Riemann-sum modulus row, the* $\mathsf{Complex}^{\uparrow}$ *output row, and the operation ledger; operation-law ledger exactness for L;* Pkg *provenance over G, F, M, I, L, H, C; and local naming exactness for N.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketSemanticNameCert`

Proof. Carrier admission is projection of the tuple S. Each contour, input, modulus, output, ledger, transport, continuation, package, and local naming coordinate is a displayed row of one finite BHist packet.

Classifier stability is componentwise. Transport of G preserves the piecewise-linear contour data, transport of F preserves the holomorphic input row over the transported neighbourhood, transport of M preserves the Riemann-sum modulus, and transport of I preserves the complex output row.

The package and naming rows repack those same coordinates. Since the carrier definition lists no host contour-integral function, quotient path space, choice principle, or completeness oracle, the obligation surface is exactly the five displayed rows. □

Theorem 583.3 (Contour-integral socket transport)**.** *Let $S = (G, F, M, I, L, H, C, P, N)$ be an accepted* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *carrier. Any componentwise* hsame *transport of the contour row, holomorphic input row, Riemann-sum modulus row, and complex output row preserves carrier admission, the finite* Cont *read routes, the operation-law ledger,* Pkg *provenance, and the local* NameCert *ledger.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketCarrier transport`
↪ variant `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketCarrier component transport closure`

Proof. Begin with the transport row H. It stores the componentwise hsame comparisons for the contour, holomorphic input, modulus, output, ledger, and local naming rows.

The continuation routes in C read only those transported coordinates. Therefore a consumer that follows a route before transport reaches the corresponding transported coordinate after transport.

The package and local naming rows are attached to the same finite packet. Repacking through H preserves admission, provenance, and the local certificate without exposing any host equality beyond the displayed hsame rows. □

Theorem 583.4 (Contour-integral socket ledger exactness)**.** *For an accepted* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *carrier $S = (G, F, M, I, L, H, C, P, N)$, every exported contour-integral operation law factors through the ledger row L and one of the displayed rows G, F, M, I, H, C, P, N. The ledger entries are exactly the linearity, concatenation, reversal, reparameterization, Cauchy-triangle, polygon-subdivision, and Cauchy-integral-formula rows named by the contour-integral operation surface.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketCarrier ledger exactness`

Proof. Read an exported operation-law request through the finite routes in C. If the request asks for contour shape, holomorphic input data, modulus evidence, output value, transport, package, or local naming, the corresponding coordinate of S supplies it.

If the request asks for an operation law, the only classifier-visible destination is the ledger row L. Its entries are the displayed operation laws listed in the carrier definition.

Thus every exported law is accounted for by a row of the packet, and no additional analytic theorem, quotient-path equation, or host integral identity can enter the socket. □

Theorem 583.5 (Contour-integral socket non-escape)**.** *An accepted* ContourIntegralSocket$^{\uparrow}$ *carrier exports no host complex-analysis theorem, no quotient path equality, no* Classical.choice *witness, and no hidden completeness object. A consumer request outside the displayed contour, holomorphic input, modulus, output, operation-ledger, transport, continuation, package, and local naming rows is recorded as outside the socket boundary and is not returned as a contour-integral fact.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketNonescape`

Proof. Inspect the carrier coordinates. The packet contains the displayed rows $G, F, M, I, L, H, C, P, N$ and no further analytic carrier.

The continuation routes in C can only land in those rows. A request for a host complex-analysis theorem, quotient path equality, choice witness, or undisplayed completeness object has no coordinate to read.

The local naming ledger therefore records such a request as outside the operation socket. It does not manufacture a contour-integral fact or an additional Complex$^{\uparrow}$ output row. □

Theorem 583.6 (Contour-integral socket handoff)**.** *Every analytic-continuation or Cauchy-operation consumer of an accepted* ContourIntegralSocket$^{\uparrow}$ *carrier factors its read through the finite packet* $S = (G, F, M, I, L, H, C, P, N)$*. Analytic-continuation consumers receive the holomorphic input, contour-domain, transport, and output rows; Cauchy-operation consumers receive the Cauchy-triangle, polygon-subdivision, and Cauchy-integral-formula rows of* L *together with the same provenance and local naming rows.*

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketHandoff`

Proof. For an analytic-continuation consumer, follow the routes in C to the holomorphic input row F, the contour-domain information in G, the output row I, and the transports in H. These rows supply the finite handoff surface.

For a Cauchy-operation consumer, follow the routes in C to the operation ledger L. The Cauchy-triangle, polygon-subdivision, and Cauchy-integral-formula entries are ledger rows attached to the same contour and holomorphic input data.

Both consumers then read Pkg provenance and local naming through P and N. The handoff is therefore bounded by the displayed socket packet and does not expose any standalone host contour-integral operation. □

Theorem 583.7 (Contour-integral socket boundary route totality)**.** *Every accepted analytic-continuation, Cauchy-operation, or contour-law consumer route for* ContourIntegralSocket$^{\uparrow}$ *is exhausted by the displayed packet rows*

$$G, F, M, I, L, H, C, P, N.$$

Requests for data outside the contour row, holomorphic input row, Riemann-sum modulus row, complex output row, operation-law ledger, componentwise transport, finite route, package provenance, and local naming row are outside the socket boundary.

Proof. Start with the local obligations from Definition 583.2. They identify carrier admission, classifier stability, ledger exactness, provenance, and naming as finite rows of the same packet.

Apply ledger exactness. Any contour-law request must land in L or in one of the supporting rows G, F, M, I, H, C, P, N, so no separate analytic law row is available.

Use non-escape to handle requests outside the displayed boundary. A request for a host theorem, quotient path equality, choice witness, or hidden completeness object has no coordinate in S and is recorded as outside the socket.

Finally apply the handoff theorem. Analytic-continuation and Cauchy-operation consumers receive their data only by following the finite routes through C to G, F, M, I, L, H, P, N, which proves route totality for the socket boundary. □

Lean4 ✓ `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketBoundaryRouteTotality`

Closure status for $\mathsf{NameCert}_{\mathsf{ContourIntegralSocket}\uparrow}$.

Constructive story: A $\mathsf{ContourIntegralSocket}^{\uparrow}$ packet is generated from a PLContour row, a $\mathsf{Holomorphic}^{\uparrow}$ input row, a Riemann-sum modulus row, a $\mathsf{Complex}^{\uparrow}$ output row, operation-law ledger rows, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourIntegralSocket}\uparrow}, \mathsf{publicClosed})$

Scope closed: Definition 583.1, Definition 583.2, Definition 583.3, Definition 583.4, Definition 583.5, Definition 583.6, and Definition 583.7 define the finite operation socket, transport rows, operation-law ledger exactness, non-escape boundary, downstream handoff, and route-total boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContourIntegralSocket}\uparrow}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.ContourIntegralSocketUp.ContourIntegralSocketBoundaryRouteTotality`

Bridge status: none

Not claimed: This chapter does not close a host contour-integral function, quotient path spaces, global complex analysis, RH zero-location consequences, or bridge checking.

Upgrade path: Bridged closure requires a bridge theorem from the public socket package to the abstract contour-integral operation schema.

Chapter 584

A Concrete Naming Certificate for $\mathsf{ExternalityGate}^{\uparrow}$

The $\mathsf{ExternalityGate}^{\uparrow}$ interface is the concrete gate attached to the name-to-gate correspondence of Definition 9400.5. It turns a named externality into a finite NameCert packet: a socket-name row, a gate-question row, a classifier for visible use, a refusal ledger, an audit readback route, transport and continuation rows, package provenance, and a local naming certificate. The packet makes the gate question inspectable without turning the requested external supply into an internal BEDC object.

The packet consumes the apophatic-name row of chapter 543 and the socket-ledger row of chapter 536 before it exposes the gate question, visible-use classifier, refusal ledger, audit readback, transports, routes, provenance, and local NameCert row.

Definition 584.1 (ExternalityGate carrier)**.** *An* $\mathsf{ExternalityGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, Q, U, R, D, H, C, P, N)$$

with the following visible coordinates:

1. *A is the apophatic-name row naming the externality;*
2. *Q is the gate-question row determined by that name;*
3. *U is the visible-use classifier row distinguishing cited socket use from hidden internalization;*
4. *R is the refusal ledger for requests that would make the externality available as an internal generator, theorem, axiom, quotient, or host datum;*
5. *D is the audit readback row exposing the name, gate question, visible-use classifier, and refusal outcome to downstream checks;*
6. *H records componentwise* hsame *transports for name, gate question, classifier, refusal, readback, provenance, and local certificate rows;*
7. *C records the* Cont *routes by which consumers read the gate question, classifier, refusal ledger, and audit readback;*
8. *P records* Pkg *provenance over* $\mathsf{ApophaticName}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}$ *row.*

The carrier contains no coordinate for the requested external supply itself.

Theorem 584.2 (ExternalityGate name-to-gate obligations)**.** *For every accepted* $\mathsf{ExternalityGate}^{\uparrow}$ *carrier*

$$E = (A, Q, U, R, D, H, C, P, N),$$

the apophatic-name row A determines the gate-question row Q as the public question by which hidden internalization is tested. The local $\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}$ *surface consists of carrier admission, name readback, gate-question readback, visible-use classification, refusal-ledger exhaustion, audit readback, componentwise transport, continuation routing, package provenance, and the local naming row.*

Proof. Project the carrier. The tuple lists the name row A, gate-question row Q, visible-use classifier U, refusal ledger R, audit readback D, transport rows, continuation routes, provenance, and local certificate.

The name-to-gate obligation is the dependency from A to Q. A consumer that cites the externality must pass through the gate question before it can be classified as visible use by U or refused by R.

The remaining obligations are the ordinary NameCert routing obligations. The readback row D exposes exactly the name, question, classifier, and refusal outcome, while H, C, P, N preserve and name those rows without adding an internal supply coordinate. □

Theorem 584.3 (ExternalityGate non-internalization)**.** *Every accepted* $\mathsf{ExternalityGate}^{\uparrow}$ *packet keeps the externality named by A outside the generated substrate. A downstream consumer may cite A, ask the gate question Q, receive the visible-use classification U, read the refusal ledger R, and replay the audit readback D, but it cannot extract an internal generator, theorem, axiom, quotient representative, host datum, or unrecorded supply from the packet.*

Proof. Begin with the carrier inventory of Definition 584.1. The requested external supply is not a coordinate of E; only its name, gate question, classifier, refusal ledger, and audit readback are present.

Apply the name-to-gate obligations. Any consumer route must pass through Q, then through either the visible-use classifier U or the refusal ledger R.

The transport and continuation rows preserve the same finite surface. Since no route in C targets an internal supply coordinate, no downstream read can convert the named externality into generated substrate content. □

Theorem 584.4 (ExternalityGate NameCert surface)**.** *The public* $\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}$ *surface is exhausted by the apophatic-name row A, gate-question row Q, visible-use classifier U, refusal ledger R, audit readback row D, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local certificate row N. It has no second surface for anonymous external dependence, unmarked proof input, hidden setup fields, or host-side recognition content.*

Proof. Use the NameCert obligations of Definition 584.2. They list every row available to the local certificate.

The non-internalization theorem then removes any possible extra source for the named externality. What remains is exactly the finite public surface $A, Q, U, R, D, H, C, P, N$.

Thus any consumer that claims an externality gate must cite the displayed rows. A route through anonymous dependence or hidden host recognition has no row to project and is refused by the ledger. □

Theorem 584.5 (ExternalityGate audit readback)**.** *For an accepted* $\mathsf{ExternalityGate}^{\uparrow}$ *packet, the audit readback row D reports exactly four visible facts: the apophatic name A, the gate question Q, the visible-use classifier result U, and the refusal-ledger outcome R. The readback is stable under the displayed* hsame *and* Cont *routes and exposes no internal supply object beyond those four rows.*

Proof. Project the audit readback row D from the carrier. By definition, it is attached to A, Q, U, R and not to a hidden external supply coordinate.

Use the componentwise transport rows in H and replay routes in C. They move the name, gate question, classifier, refusal ledger, and readback together, so the audit report remains stable under displayed transport.

Finally apply the NameCert surface theorem. Since the certificate surface is exhausted by the listed rows, the audit readback cannot expose any additional internal object. □

Theorem 584.6 (ExternalityGate public consumer package)**.** *Every public consumer of an accepted* $\mathsf{ExternalityGate}^{\uparrow}$ *packet receives exactly the apophatic-name row A, gate-question row Q, visible-use classifier U, refusal ledger R, audit readback D, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row N. The public package has no internal generator, theorem, axiom, quotient representative, host datum, hidden supply, anonymous dependence row, or unmarked proof input.*

Proof. Start from the NameCert surface theorem. It exhausts the accepted externality gate by the rows $A, Q, U, R, D, H, C, P, N$ and gives no second coordinate for anonymous dependence or host recognition content.

Use the audit readback theorem. The readback reports exactly the apophatic name, gate question, classifier result, and refusal outcome, and displayed transport keeps those rows attached to the same packet.

Finally apply non-internalization. Since the named externality is never converted into an internal generator or hidden supply object, the public consumer receives exactly the displayed package and no internal theorem, axiom, quotient representative, host datum, or extra supply. □

The Lean TasteGate boundary is `lean4/BEDC/Derived/ExternalityGateUp/TasteGate.lean`. It provides the chapter TasteGate instance for the displayed name, question, classifier, refusal, audit, transport, continuation, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}$.

Constructive story: An $\mathsf{ExternalityGate}^{\uparrow}$ packet is generated from an apophatic-name row, a gate-question row, a visible-use classifier, a refusal ledger, an audit readback row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 584.1, Definition 584.2, Definition 584.3, Definition 584.4, Definition 584.5, Definition 584.6, and

Lean4 ✓ `BEDC.Derived.ExternalityGateUp.taste gate`

bind the finite name-to-gate carrier, refusal ledger, audit readback, non-internalization surface, public consumer package, and checked TasteGate instance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExternalityGate}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ExternalityGateUp.ExternalityGateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the requested external supply, host recognition content, hidden setup fields, quotient representatives, bridge scope, or bridge checking.

Upgrade path: The theory axis is mature for the displayed externality-gate carrier, refusal ledger, audit readback, transport, continuation, provenance, public consumer package, and TasteGate rows; only formal bridge checking remains on the independent verification axis.

Chapter 585

A Concrete Naming Certificate for $\mathsf{FableBranchWitness}^{\uparrow}$

The $\mathsf{FableBranchWitness}^{\uparrow}$ interface turns the local selector-step surface of section 9399.4 into a finite naming certificate. It isolates a single realized branch from the accumulated selector ledger: a consumer sees the source Hist row, selected mark, result Hist row, empty-boundary step, local admissibility certificate, transport rows, read routes, provenance, and local naming row, but it receives no global chooser and no completed branch policy.

The enacted local branch row is supplied by chapter 685, while chapter 516 is the finite selector-ledger consumer of the same branch shape. The witness packet therefore sits between the one-step empty-boundary selector and the ledger surface that records the branch without turning it into a global policy.

Definition 585.1 (FableBranchWitness carrier)**.** *A* $\mathsf{FableBranchWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (h, m, r, E, A, H, C, P, N)$$

with the following visible rows:

1. *h is the source* Hist *row at which the local branch read is made;*
2. *m is the selected Mark exposed by the selector step;*
3. *r is the result* Hist *row produced by enacting m at h;*
4. *E is the* EmptyStep *and* Ext *row recording the empty-boundary extension from h through m to r;*
5. *A is the local admissibility certificate for the displayed branch predicate at (h, m, r);*
6. *H records componentwise* hsame *transport for the source, selected mark, result row, step row, admissibility row, continuation row, provenance row, and local naming row;*
7. *C records the finite* Cont *routes by which consumers read h, m, r, E, A, H, P, and N;*
8. *P records* Pkg *provenance over the Empty-fable selector-step surface,* Hist*, Mark,* Ext*,* hsame*,* Cont*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FableBranchWitness}^{\uparrow}}$ *row.*

The carrier is finite and local. It contains no global selection function, no family of choices indexed by all branch predicates, no hidden mark selector, no infinite story ledger, and no truth oracle for admissibility.

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessTasteGate single carrier alignment`

Theorem 585.2 (FableBranchWitness NameCert obligations)**.** *For the carrier of Definition 585.1, the local* $\mathsf{NameCert}_{\mathsf{FableBranchWitness}^{\uparrow}}$ *surface consists of five finite obligations: carrier admission for* $W = (h, m, r, E, A, H, C, P, N)$*; mark exposure for the selected row* m*; step exactness for the* EmptyStep *and* Ext *row* E *from* h *to* r*; admissibility exactness for the local certificate* A*; and non-escape for hidden choosers, global branch policies, infinite ledgers, and truth oracles.*

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessNameCertObligations`

Proof. Carrier admission is projection of the displayed tuple. The source Hist, selected mark, result Hist, step row, admissibility row, transport row, continuation routes, package provenance, and local naming row are all visible coordinates of the same packet.

Mark exposure is the second coordinate m. Since the consumer reads m directly through one of the routes in C, the selected branch is not reconstructed by an outside selector.

Step exactness is supplied by E, which binds the source row h, the selected mark m, and the result row r as one empty-boundary extension. Admissibility exactness is supplied by A at the same triple.

The non-escape clause follows from the carrier inventory. There is no coordinate for a global choice function, a branch policy over all Hist rows, an infinite ledger, or a truth oracle, so no local naming projection can export one. □

Theorem 585.3 (FableBranchWitness selector exposure)**.** *Every accepted* $\mathsf{FableBranchWitness}^{\uparrow}$ *carrier exposes the realized selector mark as the displayed row* m *and routes every consumer read of the branch through the finite rows*

$$h, m, r, E, A, H, C, P, N.$$

Consequently a downstream Empty-fable consumer may inspect the local witnessed branch, but it may not infer a global chooser, a selected branch for an unrelated Hist row, or an accumulated selector ledger beyond this single witnessed step.

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessCarrier selector exposure`

Proof. Project the accepted carrier to the mark row m and the route row C. These two rows give the consumer a displayed selector and the finite routes by which that selector is read.

The rows E and A bind the selector to the same local branch: E records the empty-boundary extension from h through m to r, and A records admissibility for that displayed triple. The transport row H preserves those coordinates componentwise.

Finally apply the non-escape part of Definition 585.2. Since the packet has only the listed rows, it realizes local branch selection without exporting a global choice function or an accumulated ledger. □

Theorem 585.4 (FableBranchWitness local transport exactness)**.** *For every accepted* $\mathsf{FableBranchWitness}^{\uparrow}$ *carrier*

$$W = (h, m, r, E, A, H, C, P, N),$$

the componentwise hsame *row* H *transports exactly the displayed source row* h*, selected mark* m*, result row* r*, empty-boundary step row* E*, admissibility row* A*, continuation routes* C*, provenance row* P*, and local name* N*. This transport preserves the local selector surface exposed by Definition 585.3 and the TasteGate-aligned carrier row marked at Definition 585.1; it adds no global chooser, completed branch policy, hidden selector, infinite ledger, or truth oracle.*

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessCarrier local transport`

Proof. Start from the TasteGate-aligned carrier row of Definition 585.1. Its coordinates are exactly $h, m, r, E, A, H, C, P, N$, and the local NameCert obligations of Definition 585.2 require carrier admission, selected-mark exposure, empty-boundary step exactness, admissibility exactness, and non-escape for hidden choice principles.

Project the transport coordinate H. Componentwise hsame can only compare the displayed source, mark, result, step, admissibility, continuation, provenance, and local-name rows already present in the carrier.

Apply selector exposure from Definition 585.3. Since every consumer read of the branch routes through the same finite tuple, the transported packet remains the same local witnessed branch rather than a family of branches.

The carrier inventory contains no coordinate for a global chooser, completed branch policy, hidden selector, infinite ledger, or truth oracle. Transport exactness therefore preserves the local branch witness and exports none of those objects. □

Theorem 585.5 (FableBranchWitness empty-boundary selector dependency)**.** *Every accepted* FableBranchWitness$^{\uparrow}$ *packet reads*

$$h, m, r, E, A, H, C, P, N$$

as the same local branch shape supplied by chapter 685 and consumed by chapter 516. The source row, selected mark, result row, empty-boundary extension, admissibility witness, transport, continuation route, provenance, and local name remain a finite one-step selector surface, with no global chooser, completed branch policy, hidden selector, infinite ledger, or truth oracle.

Proof. Project the accepted carrier of Definition 585.1. Its visible rows are exactly the local branch tuple $h, m, r, E, A, H, C, P, N$.

Use Definition 585.3. The selected mark m and the route row C expose the same one-step branch to every consumer, while E binds h, m, r as the empty-boundary extension and A records local admissibility.

Compare this exposed tuple with chapter 685 and chapter 516. The former supplies the enacted local branch row, and the latter consumes the finite selector-ledger row. Since the witness carrier has no coordinate for a global chooser, completed branch policy, hidden selector, infinite ledger, or truth oracle, the dependency is exactly the displayed local branch shape. □

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessCarrier empty boundary selector dependency`

Theorem 585.6 (FableBranchWitness scoped dependency package)**.** *The* FableBranchWitness$^{\uparrow}$ *carrier, its local* NameCert *obligations, selector exposure, local transport row, and empty-boundary selector dependency scope the packet over*

$$\mathsf{EmptyBoundarySelector}^{\uparrow}, \mathsf{EmptyFableLedger}^{\uparrow}, \mathsf{BHist}, \mathit{Mark}, \mathsf{Ext}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Within that scope every consumer read of the branch factors through the displayed source row, selected mark, result row, empty-boundary extension, admissibility row, componentwise transport, continuation routes, provenance, and local naming row. The scoped package exports no global chooser, completed branch policy, hidden selector, infinite ledger, or truth oracle.

Proof. Project the carrier of Definition 585.1. Its visible rows are exactly the local branch tuple $h, m, r, E, A, H, C, P, N$, so the carrier and local naming surface already live over BHist, *Mark*, Ext, hsame, Cont, Pkg, and NameCert.

Apply Definition 585.2, Definition 585.3, and Definition 585.4. These results expose the selected mark, bind the empty-boundary step and admissibility rows, and preserve the same displayed coordinates under local transport.

Finally use Definition 585.5. The row supplied by $\mathsf{EmptyBoundarySelector}^{\uparrow}$ and consumed by $\mathsf{EmptyFableLedger}^{\uparrow}$ is the same finite one-step selector surface, and the carrier inventory contains no coordinate from which a global chooser or truth oracle could be read. □

Lean4 ✓ `BEDC.Derived.FableBranchWitnessUp.FableBranchWitnessCarrier scoped dependency package`

Closure status for $\mathsf{NameCert}_{\mathsf{FableBranchWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{FableBranchWitness}^{\uparrow}$ packet is generated from a source Hist row, selected mark, result Hist row, empty-boundary step and Ext row, local admissibility certificate, componentwise hsame transport, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FableBranchWitness}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 585.1, Definition 585.2, Definition 585.3, and Definition 585.4 bind the local witnessed branch carrier, selector exposure, step exactness, admissibility row, exact componentwise transport, continuation routes, provenance, and no-hidden-chooser boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FableBranchWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, bridge checking, global choice, an accumulated selector ledger, branch determinacy for unrelated Hist rows, or a truth oracle for admissibility.

Upgrade path: Public export requires checked carrier admission, selector exposure, empty-boundary step exactness, admissibility exactness, transport stability, provenance, and local naming for the same finite branch-witness packet.

Chapter 586

A Concrete Naming Certificate for $\mathsf{ReflectionLimitBoundary}^{\uparrow}$

The $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ interface packages the reflection-limit boundary of chapter 9461 as a finite naming certificate. It separates the rows that close the primitive ground loop from the rows that record structural meta-description, and it forces every truth-facing meta row to remain a boundary entry rather than an admitted ground-loop closure. The structural meta-description ledger is read against the self-compilation boundary of section 9235.3, while the ground-loop side stays inside the visible BMark, BHist, msame, hsame, Cont, Pkg, and NameCert rows.

The dependency row is horizontal across the concrete-instance lattice: the G row is read through chapter 674, the T row is read through chapter 472, and the local naming refusal is aligned with chapter 543. These siblings supply the closed ground-loop surface, the truth-total refusal boundary, and the apophatic naming discipline consumed by $\mathsf{ReflectionLimitBoundary}^{\uparrow}$.

Definition 586.1 (Reflection-limit boundary carrier)**.** *A* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, M, T, H, C, P, N)$$

with the following visible rows:

1. *G is the ground-loop ledger containing the raw* BMark *constructors, the primitive* msame *rows, and the closed constructor no-confusion boundary;*
2. *M is the structural meta-description ledger for coded inductive formation, dependent formation, recursor shape, and classifier closure rows over the internal* CIC^* *presentation;*
3. *T is the truth-boundary ledger recording every attempted truth-total, runtime-total, or validation-as-row query as an* GAP *boundary entry;*
4. *H stores displayed* hsame *transports for the ground, meta-description, and truth-boundary rows;*
5. *C stores the finite* Cont *routes by which consumers read the closed ground rows, the structural meta rows, and the boundary entries;*
6. *P is the* Pkg *provenance over* BMark, BHist, msame, hsame, Cont, Pkg, NameCert, *and* GAP *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectionLimitBoundary}^{\uparrow}}$ *row.*

The carrier exports a closed ground-loop packet and an open meta-loop packet at once. It contains no BEDC truth predicate, no self-validating runtime, and no row that classifies all internal sentence codes by their truth value.

Theorem 586.2 (Reflection-limit NameCert obligations)**.** *For an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier* $L = (G, M, T, H, C, P, N)$*, the local* NameCert *surface consists of five finite obligations:*

1. *carrier admission for the displayed* BHist *packet of Definition 586.1;*
2. *ground-loop closure, requiring every exported primitive distinction read to factor through* G*;*
3. *meta-loop openness, requiring every structural self-description read to factor through* M *without becoming a truth predicate;*
4. *boundary exactness, requiring each truth-total, runtime-total, or validation-as-row query to be recorded in* T *as a* GAP *boundary entry;*
5. *package closure, requiring all transports, continuations, provenance, and naming rows to be exactly the displayed rows* H, C, P, N*.*

Thus a consumer may read the finite ground closure and the finite structural reflection surface, but it cannot export a closed meta-level truth row through the same packet.

Proof. Carrier admission is projection of the tuple in Definition 586.1. The ground ledger G, meta-description ledger M, truth-boundary ledger T, transport rows H, continuation rows C, package row P, and naming row N are the complete classifier-visible packet.

The ground-loop clause reads only G. The primitive constructors, primitive sameness rows, and constructor no-confusion boundary are finite rows of the packet, so closing the ground loop does not require a truth predicate or runtime validation row.

The meta-loop clause reads only M. The structural copy of formation and recursor behavior remains a description ledger over internal codes; it is not a classifier deciding the truth of every represented BEDC sentence.

For any truth-total, runtime-total, or validation-as-row query, the carrier exposes only T. That row records the query as a boundary entry, and H, C, P, N transport and name the same boundary entry without turning it into an admitted truth row. □

Theorem 586.3 (Reflection-limit meta-loop openness obligation)**.** *For an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier* $L = (G, M, T, H, C, P, N)$*, the row* M *remains a structural description ledger for coded formation, dependent formation, recursor shape, and classifier closure. It is never a truth-classification row and never decides represented BEDC sentence codes by truth value.*

Proof. Project M from Definition 586.1. Its declared role is structural meta-description over the internal CIC* presentation.

Apply the meta-loop openness clause of Definition 586.2. That clause permits consumers to read structural self-description through M while excluding the conversion of that row into a truth predicate. □

Theorem 586.4 (Reflection-limit boundary exactness obligation)**.** *For an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier* $L = (G, M, T, H, C, P, N)$*, every request for a truth-total row, runtime-total row, or validation-as-row judgment is recorded in* T *as a* GAP *boundary entry. No row of* G *or* M *discharges that request as a closed ground-loop or closed meta-loop fact.*

Proof. Read the truth-boundary coordinate T in the carrier. The carrier statement assigns exactly the truth-total, runtime-total, and validation-as-row requests to this boundary ledger.

Use the boundary exactness clause of Definition 586.2. It keeps those requests in T as GAP entries, while G remains a closed ground-loop ledger and M remains structural description only. □

Theorem 586.5 (Reflection-limit package closure obligation)**.** *For an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier* $L = (G, M, T, H, C, P, N)$*, the rows* H, C, P, N *only transport, route, package, and name the same finite packet. They add no truth predicate, no self-validating runtime row, and no validation row beyond the boundary entries already recorded in* T*.*

Proof. Project the structural rows H, C, P, N from the carrier. Their roles are componentwise transport, finite consumer routing, package provenance, and local naming.

Apply the package-closure clause of Definition 586.2. The clause identifies those rows with the displayed packet and gives them no extra target beyond G, M, T and the boundary entries already visible there. □

Theorem 586.6 (Reflection-limit truth-boundary non-escape)**.** *Let* $L = (G, M, T, H, C, P, N)$ *be an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier. Every consumer query asking for a truth-total row, a runtime-total row, or a validation-as-row judgment factors through the displayed boundary ledger* T*. No projection from* G, M, H, C, P, N *yields a BEDC truth predicate or a row classifying all represented BEDC sentence codes by truth value.*

Proof. Read the accepted carrier through Definition 586.1. The only coordinate whose declared role contains truth-facing, runtime-facing, or validation-facing queries is T; the carrier statement identifies those entries as GAP boundary rows.

The remaining coordinates have disjoint public roles. The G row contains primitive ground-loop data, M contains structural meta-description data, and H, C, P, N transport, route, package, and name already displayed rows. Projecting any of those coordinates therefore returns one of the listed finite rows, not a truth predicate over all represented BEDC sentence codes. □

Theorem 586.7 (Reflection-limit package-route closure)**.** *Let* $L = (G, M, T, H, C, P, N)$ *be an accepted* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ *carrier. Every accepted consumer route for* L *is exhausted by the displayed rows* G, M, T, H, C, P, N*: ground-loop reads pass through* G*, structural meta-description reads pass through* M*, boundary reads pass through* T*, and the rows* H, C, P, N *only transport, replay, package, and name those same rows.*

Proof. Apply the package-closure clause of Definition 586.2. It states that all transports, continuations, provenance, and naming rows are exactly the displayed rows H, C, P, N.

The carrier definition supplies the source rows G, M, T. Since the continuation row C has no consumer target outside the closed ground rows, structural meta rows, and boundary entries, every accepted route is a route to one of G, M, T followed by the displayed transport, package, provenance, and naming data. □

Theorem 586.8 (Reflection-limit TasteGate route alignment)**.** *The paper carrier* $L = (G, M, T, H, C, P, N)$ *is the single-carrier surface used by the checked Lean TasteGate theorem* BEDC.Derived.ReflectionLimitBoundaryU *Thus the ground row, structural meta row, truth-boundary row, transport row, continuation row, provenance row, and naming row are the rows compared by the gate, with no additional paper-side carrier.*

Proof. The Lean theorem is stated for the inductive carrier with seven visible coordinates: ground, meta row, truth boundary, transport, continuation, provenance, and name. These are the same seven coordinates displayed in Definition 586.1.

The paper route therefore reads the checked gate as an alignment statement for the displayed packet itself. The theorem adds no second carrier and no extra truth or runtime coordinate; it records that the gate compares exactly the carrier rows already used by the local NameCert obligations. □

Theorem 586.9 (Reflection-limit mature consumer separation)**.** *Let $L = (G, M, T, H, C, P, N)$ be an accepted* ReflectionLimitBoundary$^{\uparrow}$ *carrier. Every mature consumer of L separates the closed ground-loop row G, the structural meta-description row M, and the truth-boundary row T before reading the transport, replay, provenance, and local naming rows H, C, P, N. The consumer may route closed ground-loop data through G, structural descriptions through M, and boundary refusals through T, but it receives no BEDC truth predicate, self-validating runtime export, or validation-as-row classifier from the combined carrier.*

Proof. Start from the displayed carrier of Definition 586.1. Its coordinates assign disjoint public roles to G, M, and T: ground-loop closure, structural meta-description, and truth-boundary recording.

Apply Definition 586.6. Any truth-total, runtime-total, or validation-as-row request factors through T and cannot be projected from G, M, H, C, P, N.

Finally apply Definition 586.7. The rows H, C, P, N only transport, replay, package, and name the same separated packet, so a mature consumer can compose the three readings without acquiring a BEDC truth predicate or self-validating runtime export. □

Closure status for NameCert$_{\text{ReflectionLimitBoundary}^{\uparrow}}$.

Constructive story: A ReflectionLimitBoundary$^{\uparrow}$ packet is generated from a closed ground-loop ledger, a structural meta-description ledger, a truth-boundary ledger, displayed hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ReflectionLimitBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 586.1, Definition 586.2, Definition 586.3, Definition 586.4, Definition 586.5, Definition 586.6, Definition 586.7, Definition 586.8, and
Lean4 ✓ `BEDC.Derived.ReflectionLimitBoundaryUp.taste gate`
bind the finite ground, structural meta-description, truth-boundary, transport, route, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{ReflectionLimitBoundary}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ReflectionLimitBoundaryUp.ReflectionLimitBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a BEDC truth predicate, self-validating runtime, runtime-validation row, or bridge checking for ReflectionLimitBoundary$^{\uparrow}$.

Upgrade path: Mature closure requires downstream examples and completeness-style cross-chapter consumers for the boundary surface.

Chapter 587

A Concrete Naming Certificate for $\mathsf{ContourSumWindow}^{\uparrow}$

The $\mathsf{ContourSumWindow}^{\uparrow}$ interface gives the contour-integral operation a finite window packet that downstream analytic chapters can inspect without importing a host integration operator. It records one piecewise-linear contour row, one holomorphic input row, a finite subdivision and sample ledger, the resulting Riemann-sum value, the output complex-history row, transport and continuation rows, package provenance, and a local naming row.

The window packet is the finite Riemann-sum face of chapter 521. It consumes the holomorphic input surface before exposing the $\mathsf{Complex}^{\uparrow}$ output row, so the analytic dependency is the displayed contour-window route rather than an ambient contour-integral operator.

Definition 587.1 (ContourSumWindow carrier)**.** *A* $\mathsf{ContourSumWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (\gamma, f, S, R, I, H, C, P, N)$$

with the following displayed rows:

1. γ *is the* PLContour *row naming the finite endpoint list and segment history used by the contour read;*
2. f *is the* $\mathsf{Holomorphic}^{\uparrow}$ *input row on the displayed neighbourhood of* γ*;*
3. S *is the finite subdivision and sample row, listing the segment subdivision, sample point, and local mesh witness for every segment;*
4. R *is the Riemann-sum ledger, recording the finite summands, the mesh comparison, and the value computed from* f *on the samples of* S*;*
5. I *is the output* $\mathsf{Complex}^{\uparrow}$ *history row classified as the window value associated to the displayed ledger* R*;*
6. H *is the componentwise* hsame *transport row for* γ*,* f*,* S*,* R*, and* I*;*
7. C *is the finite* Cont *route row carrying subdivision refinement, contour endpoint transport, holomorphic-input transport, and output readback;*
8. P *is the* Pkg *provenance over* PLContour*,* $\mathsf{Holomorphic}^{\uparrow}$*, finite subdivision rows, Riemann-sum rows,* $\mathsf{Complex}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{ContourSumWindow}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowCarrier semantic name certificate`

Theorem 587.2 (ContourSumWindow NameCert obligations)**.** *For every accepted* $\mathsf{ContourSumWindow}^{\uparrow}$ *carrier* $W = (\gamma, f, S, R, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContourSumWindow}^{\uparrow}}$ *surface consists exactly of finite carrier admission, subdivision ledger exactness, contour route stability, output readback through* I*, package provenance through* P*, and non-escape from host integration, open-cover compactness, selected limits, quotient values, or unlisted analytic function-space objects.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowCarrier namecert obligations`

Proof. Project the carrier of Definition 587.1. Carrier admission is the display of $\gamma, f, S, R, I, H, C, P, N$ as rows of one finite BHist packet, so no consumer can read a contour, integrand, subdivision, sum, or output row outside that list.

Read the subdivision and Riemann-sum coordinates next. The row S supplies the finite segment subdivision and sample data, while R supplies the summand ledger and mesh comparison attached to those samples. Exactness is therefore ledger exactness for the listed rows, not convergence in an ambient integration theory.

Finally apply the displayed transport and continuation rows. The hsame row transports the contour, input, ledger, and output componentwise, and the Cont row records only the finite routes named in the packet. Since no coordinate names host integration, compactness, selected limits, or quotient values, the local $\mathsf{NameCert}$ exports only the finite window surface. □

Theorem 587.3 (ContourSumWindow Cauchy boundary)**.** *Every Cauchy-facing read of an accepted* $\mathsf{ContourSumWindow}^{\uparrow}$ *carrier factors through the same* $\mathsf{PLContour}$ *row* γ*, the same* $\mathsf{Holomorphic}^{\uparrow}$ *row* f*, the displayed subdivision row* S*, the Riemann-sum ledger* R*, the* $\mathsf{Complex}^{\uparrow}$ *output row* I*, and the visible* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The boundary exports only finite contour-window data and does not assert a host Cauchy integral theorem, a host contour integral, open-cover compactness, selected limiting values, quotient complex histories, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowTasteGate single carrier alignment`

Proof. Start with the finite window packet. The only route from the contour and holomorphic input to an output row passes through the subdivision row S and the Riemann-sum ledger R before reaching the $\mathsf{Complex}^{\uparrow}$ row I.

The Cauchy-facing consumer can therefore inspect the displayed ledger and its transports, but it cannot replace the finite ledger by a host integral or by a selected limiting process. Those objects are not coordinates of the carrier.

The non-escape clauses of Definition 587.2 close the boundary. They keep Cauchy-style cancellation and boundary reads inside the displayed contour, subdivision, Riemann-sum, output, transport, continuation, package, and naming rows, with no open-cover compactness or quotient-value surface added. □

Theorem 587.4 (ContourSumWindow TasteGate provenance)**.** *The checked TasteGate single-carrier alignment displayed after Definition 587.3 is read only through the finite rows*

$$\gamma, f, S, R, I, H, C, P, N$$

of Definition 587.1. Consequently the taste evidence for $\mathsf{ContourSumWindow}^{\uparrow}$ *is tied to the same contour-window carrier governed by Definition 587.2 and Definition 587.3; it cannot be read as host contour integration, selected limits, quotient complex values, or a separate analytic operator.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowTasteGateProvenance`

Proof. Project the carrier definition. The tuple $\gamma, f, S, R, I, H, C, P, N$ is the only finite packet named by the chapter, so the TasteGate alignment has no independent coordinate outside that carrier.

Use the NameCert obligations theorem to bind each admissible read to carrier admission, subdivision ledger exactness, contour route stability, output readback, package provenance, and non-escape from host analytic objects. This makes the taste evidence a claim about the displayed finite window rather than about an ambient contour-integral operator.

Finally apply the Cauchy boundary theorem and the checked TasteGate marker immediately after it. The marker certifies single-carrier alignment at the same site where every Cauchy-facing read is forced through $\gamma, f, S, R, I, H, C, P, N$, so provenance remains attached to the finite contour-window rows. □

Theorem 587.5 (ContourSumWindow subdivision exactness)**.** *For every accepted* $\mathsf{ContourSumWindow}^{\uparrow}$ *carrier* $W = (\gamma, f, S, R, I, H, C, P, N)$*, each summand exposed by the window is read only through the displayed subdivision and sample row* S *together with the Riemann-sum ledger* R*. No summand may be supplied by a host contour integral, a selected limit, an unlisted mesh, or an ambient analytic function-space object.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowCarrier subdivision exactness`

Proof. Project the carrier of Definition 587.1. The row S names the finite segment subdivision, sample point, and mesh witness for every segment, while R records the finite summands and their mesh comparison.

Use Definition 587.2. Its subdivision ledger exactness clause makes S and R the only admissible readback for summand data, and its non-escape clause removes host integration, selected limits, quotient values, and unlisted analytic objects from the packet.

The Cauchy-facing boundary theorem Definition 587.3 preserves the same rows for every boundary read. Thus the subdivision and Riemann-sum ledger are the complete summand surface. □

Theorem 587.6 (ContourSumWindow refinement stability)**.** *Let* $W = (\gamma, f, S, R, I, H, C, P, N)$ *be an accepted* $\mathsf{ContourSumWindow}^{\uparrow}$ *carrier. Any subdivision refinement accepted by the window must replay through the displayed continuation row* C *after preserving* γ*,* f*,* S*,* R*, and* I *componentwise through* H*. The refinement route cannot introduce a host contour integral, a new holomorphic input, a quotient complex value, or a selected limiting process.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowCarrier refinement stability`

Proof. Read the transport coordinates in Definition 587.1. The row H transports the contour, input, subdivision, ledger, and output componentwise, and the row C records subdivision refinement as one of the finite continuation routes of the packet.

Apply Definition 587.2. Contour route stability and output readback require refinement to keep the same displayed rows before any consumer can inspect the refined window.

Finally use Definition 587.3. Since every Cauchy-facing read factors through $\gamma, f, S, R, I, H, C, P, N$, refinement has no coordinate for a host contour integral, a fresh analytic input, a quotient value, or a selected limit. □

Theorem 587.7 (ContourSumWindow residue boundary)**.** *Any residue-facing consumer of an accepted* ContourSumWindow$^{\uparrow}$ *carrier must first expose the* PLContour *row* γ*, the* Holomorphic$^{\uparrow}$ *row* f*, the subdivision and sample row* S*, the Riemann-sum ledger* R*, the* Complex$^{\uparrow}$ *output row* I*, and the structural rows* H, C, P, N *before reading a residue handoff. The boundary exports no host contour integral, no residue theorem, no quotient complex value, no selected limiting process, and no ambient analytic function-space object.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindow residue boundary`

Proof. Project the carrier of Definition 587.1. The rows available to a residue-facing consumer are exactly $\gamma, f, S, R, I, H, C, P, N$, with I read only after the finite Riemann-sum ledger R has been displayed.

Apply subdivision exactness, Definition 587.5. It forces each summand to be read from the subdivision and sample row S together with the ledger R, so a residue handoff cannot replace the window by an unlisted contour-integral value.

Apply refinement stability, Definition 587.6, and then the Cauchy boundary theorem Definition 587.3. The same contour, holomorphic input, subdivision, ledger, output, transport, replay, provenance, and naming rows remain visible at the boundary, while host contour integrals, residue theorems, quotient values, selected limits, and ambient analytic function spaces remain outside the packet. □

Theorem 587.8 (ContourSumWindow output seal nonescape)**.** *For every accepted* ContourSumWindow$^{\uparrow}$ *carrier* $W = (\gamma, f, S, R, I, H, C, P, N)$*, the* Complex$^{\uparrow}$ *output row* I *is sealed only after the* PLContour *row* γ*, the* Holomorphic$^{\uparrow}$ *row* f*, the subdivision row* S*, and the Riemann-sum ledger* R *have been displayed. The sealed output exports no quotient complex value, no selected limiting value, and no host contour integral.*

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowCarrier output seal nonescape`

Proof. Project the carrier rows. The output row I is classified as the window value associated to the displayed ledger R, and R itself is computed from the samples listed in S against the contour and input rows γ and f.

Use Definition 587.2 to read output readback through I and package provenance through P. These clauses make the output seal depend on the visible contour, holomorphic-input, subdivision, and Riemann-sum rows.

The boundary theorem Definition 587.3 keeps the same seal for Cauchy-facing consumers. It therefore exports only the finite contour-window output row and refuses quotient complex values, selected limits, and host contour integration. □

Theorem 587.9 (ContourSumWindow public finite-window certificate)**.** *The public* ContourSumWindow$^{\uparrow}$ *certificate is exactly the finite route through the* PLContour *row, holomorphic input row, subdivision row, Riemann-sum ledger,* Complex$^{\uparrow}$ *output seal, componentwise transport, continuation replay, package provenance, and local NameCert. Any consumer of the certificate must read*

$$\gamma, f, S, R, I, H, C, P, N$$

as the same finite packet before it can inspect the window output.

Proof. Start with the NameCert obligations of Definition 587.2. They identify the public surface with carrier admission, subdivision ledger exactness, contour route stability, output readback, package provenance, and non-escape from host analytic objects.

Use subdivision exactness and refinement stability next. Theorems Definition 587.5 and Definition 587.6 keep every summand and every refinement replay attached to $\gamma, f, S, R, I, H, C, P, N$.

Finally apply the residue boundary and output-seal nonescape theorems, Definition 587.7 and Definition 587.8. They force any residue-facing or output-facing read to pass through the same finite rows and therefore repack the displayed window as the public certificate. □

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowPublicFiniteWindowCertificate`

Theorem 587.10 (ContourSumWindow obligation closure package)**.** *The carrier, NameCert obligations, subdivision exactness, refinement stability, residue boundary, output-seal nonescape, and public finite-window certificate assemble the full obligation surface for* ContourSumWindow$^{\uparrow}$ *over the finite* PLContour *row,* Holomorphic$^{\uparrow}$ *input, subdivision row, Riemann-sum ledger,* Complex$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert*. The package closes exactly the displayed finite contour-window obligations and does not claim a host contour integral, Cauchy's theorem, a residue theorem, selected limits, quotient complex values, open-cover compactness, or an ambient analytic function-space object.*

Proof. Begin with the carrier of Definition 587.1 and the NameCert obligations theorem Definition 587.2. They identify $\gamma, f, S, R, I, H, C, P, N$ as the finite packet and bind admissible reads to carrier admission, subdivision ledger exactness, contour route stability, output readback, package provenance, and non-escape from host analytic objects.

Apply subdivision exactness and refinement stability next. Theorems Definition 587.5 and Definition 587.6 keep each summand and each refinement replay attached to the displayed PLContour row, Holomorphic$^{\uparrow}$ input, subdivision row, Riemann-sum ledger, Complex$^{\uparrow}$ output seal, hsame transport, and Cont route.

Finally combine the residue boundary, output-seal nonescape, and public finite-window certificate, namely Definition 587.7, Definition 587.8, and Definition 587.9. These results repack the same rows as the chapter's public obligation surface while leaving host contour integration, Cauchy's theorem, residue theorems, selected limits, quotient complex values, compactness, and ambient function spaces outside the packet. □

Lean4 ✓ `BEDC.Derived.ContourSumWindowUp.ContourSumWindowObligationClosurePackage`

Closure status for NameCert$_{\text{ContourSumWindow}^{\uparrow}}$**.**

Constructive story: A ContourSumWindow$^{\uparrow}$ packet is generated from a displayed PLContour row, a Holomorphic$^{\uparrow}$ input row, finite subdivision and sample rows, a Riemann-sum ledger, a Complex$^{\uparrow}$ output row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContourSumWindow}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 587.1, Definition 587.2, Definition 587.5, Definition 587.6, Definition 587.7, Definition 587.8, Definition 587.9, and Definition 587.10 bind the carrier, NameCert obligations, subdivision exactness, refinement stability, residue boundary, output-seal nonescape, public finite-window certificate, and obligation-closure package as the scoped finite contour-window surface.

Formal verification: FormalStatus(NameCert$_{\text{ContourSumWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host contour integration, Cauchy's integral theorem, open-cover compactness, selected limits, quotient complex values, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subdivision ledger exactness, contour route stability, Cauchy boundary, and non-escape for the same finite packet.

Chapter 588

A Concrete Naming Certificate for $\mathsf{ZetaContinuationApplication}^{\uparrow}$

The $\mathsf{ZetaContinuationApplication}^{\uparrow}$ packet records the application-level certificate that turns the zeta continuation surface into a finite NameCert object. It reads the eta continuation row, the functional-equation continuation row, pole and zero ledgers, Gamma horizon rows, application routes, transport rows, package provenance, and a local naming row. The packet is an application certificate only: it does not assert RH, complete the constructive $\mathsf{Gamma}^{\uparrow}$ chapter, classify nontrivial zeros, or export a continuation branch outside the displayed routes.

The carrier consumes the $\mathsf{Gamma}^{\uparrow}$ horizon row from chapter 75, the analytic-operation routing surface from chapter 577, and the $\mathsf{ZetaBasic}^{\uparrow}$ source row from chapter 71. These sibling rows are classifier inputs for the finite packet: G bounds Gamma reads, A routes analytic continuation through the operation surface, and E records the eta handoff from the basic zeta source without turning that source row into a global continuation theorem.

Definition 588.1 (ZetaContinuationApplication carrier)**.** *A* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (E, F, P, T, G, A, H, C, Q, N)$$

with the following displayed rows:

1. *E is the eta-continuation row, recording the η handoff from the* $\mathsf{ZetaBasic}^{\uparrow}$ *source and the denominator boundary $1 - 2^{1-s}$;*
2. *F is the functional-equation continuation row, recording the reflected input $1 - s$, the factors 2^s, π^{s-1}, $\sin(\pi s/2)$, and their displayed product route;*
3. *P is the pole ledger row at $s = 1$, recording the simple-pole boundary and residue* $\mathsf{E1(e)}$ *without exporting a regular zeta value there;*
4. *T is the zero ledger row, recording the trivial-zero inputs witnessed by the sine factor and the absence of any nontrivial-zero-location conclusion;*
5. *G is the* $\mathsf{Gamma}^{\uparrow}$ *horizon row, recording the domain boundary for $\Gamma(1 - s)$, its pole exclusions, and the fact that the full constructive Gamma carrier is not supplied here;*
6. *A is the application-certificate row, reading E, F, P, T, G as the finite application of analytic continuation to the zeta source;*
7. *H stores componentwise* hsame *transports for E, F, P, T, G, A and for the local naming row;*

8. *C stores the finite* Cont *routes by which consumers read eta continuation, functional-equation continuation, pole and zero ledgers, Gamma horizon rows, application provenance, and local naming;*
9. *Q is the* Pkg *provenance row over the same application packet;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationApplication}^{\uparrow}}$ *row.*

The carrier exposes exactly this finite application packet. It has no coordinate for a value at the pole, no complete $\mathsf{Gamma}^{\uparrow}$ *construction, no branch selector outside the displayed eta and functional-equation routes, and no RH-facing nontrivial-zero classifier.*

Lean4 (definition) `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationUp`

Theorem 588.2 (ZetaContinuationApplication NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZetaContinuationApplication}^{\uparrow}}$ *surface for Definition 588.1 consists of carrier admission for Z, componentwise classifier stability for E, F, P, T, G, A, ledger exactness for pole and zero rows, Gamma-horizon boundary exactness,* Cont *replay through C,* Pkg *provenance through Q, and local naming exactness for N.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier namecert obligations`

Proof. Carrier admission is projection from the tuple Z. Each continuation, ledger, horizon, application, transport, replay, package, and naming coordinate is displayed as a row of the same finite BHist packet.

Classifier stability is componentwise. The transports in H preserve the eta-continuation row, the functional-equation row, the pole row, the zero ledger, the Gamma horizon row, and the application-certificate row without adding a new analytic object.

The replay and provenance rows then repack the same data. The routes in C expose only the displayed application coordinates, Q records their package provenance, and N names the local certificate surface. □

Theorem 588.3 (ZetaContinuationApplication carrier admission)**.** *For the displayed carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$ of Definition 588.1, the admissible* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet is exactly the finite application certificate whose source locks are the* $\mathsf{Gamma}^{\uparrow}$ *horizon row of chapter 75, the analytic-operation routing surface of chapter 577, and the* $\mathsf{ZetaBasic}^{\uparrow}$ *eta source of chapter 71. The checked TasteGate alignment recorded at this site fixes those rows as the single carrier surface consumed by the chapter.*

Proof. Project the tuple Z. Its displayed coordinates are precisely the eta row E, functional-equation row F, pole row P, zero row T, Gamma horizon row G, application row A, transport row H, replay row C, provenance row Q, and local naming row N.

The sibling source locks identify where those coordinates may be read. The Gamma-facing data enters only through G, the analytic-operation data enters only through A, and the zeta source data enters only through E.

The TasteGate marker attached to the carrier definition records that these rows share one carrier alignment. Hence carrier admission is the displayed finite packet rather than an undisplayed analytic object. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrierAdmission`

Theorem 588.4 (ZetaContinuationApplication componentwise stability)**.** *For an accepted carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, every* hsame *or* Cont *replay that preserves the checked TasteGate alignment preserves the components E, F, P, T, G, A separately. In particular, the* $\mathsf{Gamma}^{\uparrow}$*, analytic-operation, and* $\mathsf{ZetaBasic}^{\uparrow}$ *sibling locks may be transported only through their own displayed rows and may not be exchanged for a different continuation source.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier componentwise stability`

Proof. Read the transport coordinate H componentwise. It stores preservation data for the eta, functional-equation, pole, zero, Gamma-horizon, application, and naming rows without adding a mixed row between them.

Replay through C keeps the same separation. Eta-side requests return to E, functional-equation requests return to F, ledger requests return to P or T, Gamma-facing requests return to G, and application requests return to A.

Since the TasteGate marker fixes the single-carrier alignment, these componentwise routes cannot swap the sibling locks. The accepted packet remains stable row by row. □

Theorem 588.5 (ZetaContinuationApplication ledger exactness)**.** *For an accepted carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, the pole ledger P, zero ledger T, Gamma horizon row G, application provenance Q, and local name N are exact ledgers for the displayed application certificate. They record only the pole boundary, trivial-zero surface, Gamma horizon boundary, analytic-operation application route, package provenance, and local naming already present in Z, and they export no pole value, completed* $\mathsf{Gamma}^{\uparrow}$ *construction, nontrivial-zero classifier, or continuation branch outside the finite replay routes in C.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier ledger exactness`

Proof. Inspect the ledger coordinates. The pole boundary is stored in P, the zero surface is stored in T, the Gamma boundary is stored in G, and provenance and naming are stored in Q and N.

Apply the carrier admission and componentwise stability rows. Each ledger read is routed back through the same coordinate of Z and through the same sibling source lock named in the chapter introduction.

The carrier has no coordinate for the positive objects excluded in the statement. Therefore exactness is ledger exactness for the displayed packet, not a hidden continuation theorem. □

Theorem 588.6 (Eta factorization route)**.** *For an accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, every eta-side continuation read factors through E, the application row A, the transport row H, and one of the finite replay routes in C. Such a read may use the displayed denominator boundary and overlap agreement with* $\mathsf{ZetaBasic}^{\uparrow}$*, but it cannot export a value at $s = 1$ or a continuation branch outside the eta route.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier eta factorization`

Proof. Read an eta-side request through the finite routes stored in C. If the request is inside the eta continuation surface, the route lands in E and then in the application row A.

Transport through H preserves the same eta row and denominator boundary. The pole request is not part of E as a regular value; it is routed to the pole ledger P.

Thus the eta factorization is a finite application route. The certificate exports the eta handoff and overlap evidence, while pole and outside-route requests remain ledger reads. □

Theorem 588.7 (Gamma horizon boundary)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier, every functional-equation read that touches the factor* $\Gamma(1-s)$ *must pass through the Gamma horizon row* G*. Inputs where the Gamma factor is outside the displayed horizon are recorded in* G *as boundary data and are not exported as values of the continued zeta function.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier gamma horizon boundary`

Proof. Begin with a functional-equation read. The factorization row F records the reflected input and the elementary factors, while the Gamma factor is named only by the horizon row G.

If the input is inside the displayed Gamma horizon, the read may continue through F, G, A, H, C. If the input is at a Gamma boundary, the route terminates in the ledger part of G.

The application row A therefore never converts the Gamma horizon into a completed Gamma$^{\uparrow}$ construction. It records exactly the boundary needed by the zeta continuation application. □

Theorem 588.8 (Gamma eta handoff source lock)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, every eta-factorization read that uses a Gamma-facing horizon fact consumes the* Gamma$^{\uparrow}$ *row through chapter 75. The read must pass through* E*,* G*,* A*, the componentwise* hsame *transport row* H*, and a finite route in* C*; it cannot replace the Gamma horizon row by an undisplayed Gamma construction.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier gamma eta handoff`

Proof. Start with the carrier definition. The eta side is stored in E, while the only Gamma-facing coordinate is the horizon row G.

Use Definition 588.6 to route the eta read through E, A, H, C. If the route also needs a Gamma-facing horizon fact, Definition 588.7 forces that fact to pass through G.

The sibling Gamma$^{\uparrow}$ chapter supplies the source row named by G, and the local packet only records that consumption. Since no coordinate of Z stores a completed Gamma construction, the handoff remains a finite source lock. □

Theorem 588.9 (Analytic operation boundary)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, every application-level continuation read factors through the analytic-operation surface of chapter 577. The row* A *records the application of that operation to the eta, functional-equation, pole, zero, and Gamma-horizon rows; it does not create a second continuation operation outside* E, F, P, T, G, H, C*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier analytic operation boundary`

Proof. Read an application-level continuation request through C. By the carrier definition, the only row that names application of continuation data is A, and its inputs are the displayed rows E, F, P, T, G.

The analytic-operation sibling supplies the operation boundary consumed by A. Componentwise transport through H preserves the inputs as inputs to that operation surface rather than as independent analytic continuations.

Apply Definition 588.23 to the outside cases. A request for an undisplayed branch or another operation row has no carrier coordinate, so the application route remains locked to the finite analytic-operation boundary. □

Theorem 588.10 (ZetaBasic source lock)**.** *For an accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the* $\mathsf{ZetaBasic}^{\uparrow}$ *row consumed from chapter 71 is only the finite source lock for the eta handoff. It licenses the displayed source row and its* hsame *transport into* E*, but it does not by itself license analytic continuation, pole removal, Gamma-factor transport, or any branch outside the routes displayed in* C*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier zeta basic source lock`

Proof. Project the source dependency from the carrier. The source row enters through E, where the eta handoff and denominator boundary are stored.

Use Definition 588.6 to read the source through E, A, H, C. The transport row H preserves the source handoff componentwise, and the replay row C exposes only the finite eta route.

The remaining analytic content is held in F, P, T, G, A, not in the $\mathsf{ZetaBasic}^{\uparrow}$ source row. By Definition 588.23, pole removal, Gamma transport, and outside-branch requests therefore remain outside the source lock unless the corresponding displayed row supplies them. □

Theorem 588.11 (ZetaContinuationApplication route totality)**.** *For an accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet, every eta, functional-equation, Gamma-horizon, pole-ledger, zero-ledger, or application read factors through the displayed rows*

$$E, F, P, T, G, A, H, C, Q, N.$$

The row E *carries eta input,* F *carries the functional-equation route,* P *and* T *carry the pole and trivial-zero ledgers,* G *carries the Gamma horizon,* A *carries the application route,* H *carries componentwise transport,* C *carries finite replay,* Q *carries package provenance, and* N *carries the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier route totality`

Proof. Project the carrier of Definition 588.1. It lists exactly the rows $E, F, P, T, G, A, H, C, Q, N$, so every application read starts from the finite NameCert packet rather than from an ambient analytic-continuation object.

Use the componentwise and ledger rows next. Theorem 588.4 keeps eta, functional-equation, pole, zero, Gamma, and application rows separate under H and C, while Definition 588.5 routes pole, zero, Gamma, provenance, and naming reads through their displayed ledgers.

Finally attach the sibling source locks. The Gamma eta handoff, analytic-operation boundary, and ZetaBasic source lock force source-facing reads through G, A, and E respectively. Hence every listed read factors through the ten-row carrier surface. □

Theorem 588.12 (ZetaContinuationApplication boundary exhaustion)**.** *An accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet exports no pole value, completed* $\mathsf{Gamma}^{\uparrow}$ *construction, nontrivial-zero classifier, RH statement, or off-route continuation branch. All boundary requests are exhausted by the pole ledger* P*, trivial-zero ledger* T*, Gamma horizon* G*, application row* A*, transport* H*, replay* C*, provenance* Q*, and local certificate* N *of the displayed packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier boundary exhaustion`

Proof. Start from route totality. Every boundary-facing request must already be a read of $E, F, P, T, G, A, H, C, Q,$

The ledger exactness theorem then handles the exceptional analytic surfaces. Pole requests terminate at P, trivial-zero requests at T, and Gamma-horizon requests at G; none of these rows contains a regular pole value, completed Gamma construction, or nontrivial-zero classifier.

It remains to rule out route escape. The analytic-operation boundary and ZetaBasic source lock keep A and E from becoming a second continuation theorem, and the carrier definition has no coordinate for RH or an off-route branch. Therefore the displayed boundary rows exhaust the public application packet. □

Theorem 588.13 (ZetaContinuationApplication root boundary triad)**.** *For every accepted* ZetaContinuationApplica *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, the root source locks are the Gamma horizon row G, the analytic-operation routing row A, and the* ZetaBasic$^{\uparrow}$ *source row E. Every root read of $E, F, P, T, G, A, H, C, Q, N$ must expose that triad before the application row is consumed, and the support rows H, C, Q, N may only transport, replay, package, and name the same displayed triad.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root boundary triad`

Proof. Start with carrier admission and route totality. The accepted packet is the finite carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, and every eta, functional-equation, pole, zero, Gamma, or application read factors through those rows.

Now attach the three source locks. The Gamma horizon boundary and Gamma eta handoff place Gamma-facing reads in G, the analytic-operation boundary places application routing in A, and the ZetaBasic source lock places eta-source reads in E.

Finally use ledger exactness and boundary exhaustion. The rows P, T remain pole and trivial-zero ledgers, while H, C, Q, N only preserve the displayed packet. Thus the root source locks are exactly the triad G, A, E inside the carrier. □

Theorem 588.14 (ZetaContinuationApplication root source readiness)**.** *Any eta, Gamma, or analytic-operation consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier must expose its source lock before using the application row. Eta consumers expose E, Gamma consumers expose G, and analytic-operation consumers expose A; only after that exposure may the consumer read F, P, T, H, C, Q, N as the displayed application support surface.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root source readiness`

Proof. Read the source route from Definition 588.13. The triad G, A, E is the complete root source-lock surface for the accepted packet.

Handle the three consumer classes separately. An eta consumer reaches the packet through the ZetaBasic source lock at E, a Gamma consumer reaches it through the Gamma horizon and eta handoff at G, and an analytic-operation consumer reaches it through the operation boundary at A.

The support rows then become available only as support. Route totality and ledger exactness allow F, P, T, H, C, Q, N to be read after the source lock is displayed, and boundary exhaustion prevents those support rows from supplying a hidden source. □

Theorem 588.15 (ZetaContinuationApplication root source-lock exhaustion)**.** *Every root read of an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier must expose exactly the source-lock triad E, G, A before it reads the support rows F, P, T, H, C, Q, N. Eta reads expose E, Gamma reads expose G, and analytic-operation reads expose A; no support row can stand in for this triad.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationRootSourceLockExhaustion`

Proof. Start with root source readiness. It requires source exposure before any eta, Gamma, or analytic-operation consumer may use the application support surface.

Use the root boundary triad. The only root source locks inside the carrier are E, G, A, with E supplied by the ZetaBasic source lock, G supplied by the Gamma horizon and eta handoff, and A supplied by the analytic-operation boundary.

Finally attach route totality and ledger exactness. They allow F, P, T, H, C, Q, N only as support rows after the triad is displayed, while boundary exhaustion prevents any support row from becoming a hidden source. □

Theorem 588.16 (ZetaContinuationApplication application row totality)**.** *For an accepted* ZetaContinuationApplic *carrier, the row A is the only analytic-operation consumer row. It cannot be replaced by a branch selector, off-route continuation branch, or hidden analytic-operation coordinate while preserving the same root source-lock surface.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationApplicationRowTotality`

Proof. Begin with the analytic-operation boundary. It places analytic-operation routing in the displayed row A and excludes off-route continuation branches from the carrier.

Now use root source readiness together with root source-lock exhaustion. Any analytic-operation consumer must expose A as its source lock before reading the support rows, so a branch selector cannot occupy the analytic-operation role without adding a new coordinate.

Route totality and boundary exhaustion close the argument. Every accepted route is already on $E, F, P, T, G, A, H, C, Q, N$, and no hidden branch selector or second analytic-operation row is present in that inventory. □

Theorem 588.17 (ZetaContinuationApplication boundary ledger non-escape)**.** *The rows P, T, G, Q, N exhaust all root requests for pole, trivial-zero, Gamma-horizon, provenance, and local-name data in an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet. A root consumer cannot refine the same carrier by adding a regular pole value, nontrivial-zero classifier, completed Gamma construction, detached provenance row, or second local naming row.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier boundary ledger nonescape`

Proof. Start from ledger exactness. Pole requests terminate at P, trivial-zero requests terminate at T, and the Gamma-horizon boundary is read at G.

Next use route totality. Provenance and local naming requests are displayed as Q and N inside the same carrier, with transport and replay handled by H and C rather than by additional ledger rows.

Finally apply boundary exhaustion. It excludes pole values, nontrivial-zero classifiers, completed Gamma data, off-route branches, and additional boundary ledgers, so P, T, G, Q, N are the complete root ledger boundary. □

Theorem 588.18 (ZetaContinuationApplication root non-escape)**.** *Every root-facing consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet factors through the displayed rows*

$$E, F, P, T, G, A, H, C, Q, N.$$

The root boundary excludes RH, a completed Gamma$^{\uparrow}$ *construction, a pole value, a nontrivial-zero classifier, and any off-route continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root nonescape`

Proof. Begin with root source readiness. Eta, Gamma, and analytic-operation consumers must first expose E, G, A respectively before they may use the application support rows.

Apply route totality and ledger exactness. They place all pole, trivial-zero, Gamma, application, transport, replay, provenance, and naming reads on P, T, G, A, H, C, Q, N inside the same finite carrier.

Finally invoke analytic-operation boundary, Gamma horizon boundary, ZetaBasic source locking, and boundary exhaustion. These results exclude RH, completed Gamma data, pole values, nontrivial-zero classification, and off-route continuation branches because none has a coordinate in Z. □

Theorem 588.19 (ZetaContinuationApplication carrier source obligation)**.** *Every accepted* ZetaContinuationApplic *carrier presents* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows before any root source read of* Gamma$^{\uparrow}$*,* AnalyticContinuationOperation$^{\uparrow}$*, or* ZetaBasic$^{\uparrow}$ *is available. The* Gamma$^{\uparrow}$ *row G, analytic-operation row A, and* ZetaBasic$^{\uparrow}$ *row E are displayed source rows of the packet, not analytic primitives imported outside the carrier.*

Proof. Project the accepted carrier and the root source-readiness theorem. The carrier supplies $E, F, P, T, G, A, H, C, Q, N$, while readiness requires E, G, A to be exposed before any consumer reaches the support rows.

The support rows then determine the source shape. The transport row H, replay row C, package row Q, and local naming row N keep the read inside BHist, hsame, Cont, Pkg, and NameCert data.

Thus Gamma$^{\uparrow}$, AnalyticContinuationOperation$^{\uparrow}$, and ZetaBasic$^{\uparrow}$ appear only as displayed source rows of the same packet. They are not primitives from which the carrier is reconstructed. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier carrier source obligation`

Theorem 588.20 (ZetaContinuationApplication classifier stability obligation)**.** *For any two accepted* ZetaContinuationApplication$^{\uparrow}$ *carriers whose* BHist *rows are related componentwise by* hsame *and replayed by the same* Cont *routes, classifier stability preserves the displayed* Gamma$^{\uparrow}$*,* AnalyticContinuationOpera *and* ZetaBasic$^{\uparrow}$ *source rows together with* Pkg *provenance and local* NameCert *data. The comparison classifies packet rows only; it does not classify analytic primitives outside the displayed carrier.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationClassifierStabilityObliga`

Proof. Start from componentwise stability and root source-lock exhaustion. They compare accepted packets row by row and require the source-lock triad E, G, A to remain visible before support rows are read.

Transport the comparison through the structural rows. The hsame row moves the displayed BHist coordinates, the Cont row replays the same route, and the Pkg and NameCert rows preserve provenance and local naming.

Consequently the classifier comparison preserves the $\mathsf{Gamma}^{\uparrow}$, $\mathsf{AnalyticContinuationOperation}^{\uparrow}$, and $\mathsf{ZetaBasic}^{\uparrow}$ source rows as source rows of the packet. No external analytic object is introduced by the classifier. □

Theorem 588.21 (ZetaContinuationApplication ledger route obligation)**.** *Every root ledger route of an accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet reads* $\mathsf{Gamma}^{\uparrow}$, $\mathsf{AnalyticContinuationOperation}^{\uparrow}$, *and* $\mathsf{ZetaBasic}^{\uparrow}$ *only through the displayed* BHist *carrier, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route may expose pole, trivial-zero, Gamma-horizon, provenance, and local-name ledgers, but it cannot export RH, a completed* $\mathsf{Gamma}^{\uparrow}$ *construction, an analytic-operation primitive, or a* $\mathsf{ZetaBasic}^{\uparrow}$ *source outside the packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier ledger route obligation`

Proof. Begin with boundary ledger non-escape. It exhausts pole, trivial-zero, Gamma-horizon, provenance, and local-name requests at P, T, G, Q, N inside the accepted packet.

Next use root non-escape and carrier source obligation. These rows force all root-facing reads through $E, F, P, T, G, A, H, C, Q, N$ and keep E, G, A as displayed $\mathsf{ZetaBasic}^{\uparrow}$, $\mathsf{Gamma}^{\uparrow}$, and $\mathsf{AnalyticContinuationOperation}^{\uparrow}$ source rows.

Finally apply the structural support rows. The route is carried by BHist, hsame, Cont, Pkg, and NameCert data, so a request for RH, completed Gamma data, or an external analytic primitive has no ledger coordinate to read. □

Theorem 588.22 (ZetaContinuationApplication public export)**.** *The public application packet consumed by* $\mathsf{Gamma}^{\uparrow}$, $\mathsf{AnalyticContinuationOperation}^{\uparrow}$, *and* $\mathsf{ZetaBasic}^{\uparrow}$ *is exactly the finite NameCert surface*

$$E, F, P, T, G, A, H, C, Q, N.$$

Downstream consumers may read the packet only through the displayed carrier admission, componentwise stability, ledger exactness, Gamma eta handoff, analytic-operation boundary, and ZetaBasic source lock; no additional analytic, RH-facing, or completed-Gamma row is exported.

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationPublicExport`

Proof. First combine carrier admission with route totality. The accepted packet is the ten-row carrier of Definition 588.1, and every public route factors through those rows.

Next identify the three sibling-facing entry points. The Gamma eta handoff supplies G, the analytic-operation boundary supplies A, and the ZetaBasic source lock supplies E; the remaining rows F, P, T, H, C, Q, N stay inside the same finite application packet.

Finally apply boundary exhaustion. Since pole values, completed Gamma data, nontrivial-zero classifiers, RH statements, and off-route branches have no exported coordinate, the public packet consumed by the sibling chapters is exactly the displayed NameCert surface. □

Theorem 588.23 (Application certificate non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet factors through the displayed rows* $E, F, P, T, G, A, H, C, Q, N$. *Requests for a pole value, a Gamma value beyond the horizon row, an undisplayed continuation*

branch, or a nontrivial-zero-location claim have no carrier coordinate and are recorded only as ledger or horizon boundary reads.

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier non escape`

Proof. Let a downstream consumer route be given. By the carrier definition, the route can inspect only the eta row, functional-equation row, pole row, zero ledger, Gamma horizon, application row, transport row, continuation routes, package row, and local naming row.

Pole requests land in P, Gamma-boundary requests land in G, and trivial-zero requests land in T. An undisplayed branch or a nontrivial-zero-location claim is absent from the carrier inventory.

Consequently every exported application fact is a read of the finite packet. The missing objects remain outside the certificate surface rather than becoming hidden application outputs. □

Theorem 588.24 (ZetaContinuationApplication primitive scope package)**.** *Every accepted* ZetaContinuationApplic *packet reads the* $\mathsf{Gamma}^{\uparrow}$*, analytic-operation, and* $\mathsf{ZetaBasic}^{\uparrow}$ *sibling rows only through the displayed carrier*

$$Z = (E, F, P, T, G, A, H, C, Q, N)$$

and through the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. In particular, the packet exposes the Gamma horizon by G, the analytic-operation route by A, the zeta source by E, componentwise transport by H, replay by C, package provenance by Q, and local naming by N, without importing analytic continuation as an external theorem.*

Proof. Start with the carrier admission of Definition 588.3. It fixes the visible packet as the finite BHist carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$ and identifies the Gamma, analytic-operation, and zeta-source rows as sibling inputs to that carrier.

Use componentwise stability from Definition 588.4. The hsame row H preserves E, F, P, T, G, A separately, so no sibling row can be exchanged for a different continuation source during transport.

Now apply the source-lock theorems. The Gamma eta handoff of Definition 588.8 routes Gamma-facing reads through G; the analytic-operation boundary of Definition 588.9 routes operation reads through A; and the ZetaBasic source lock of Definition 588.10 routes zeta-source reads through E.

Finally use non-escape from Definition 588.23. Since every downstream consumer factors through $E, F, P, T, G, A, H, C, Q, N$, the only remaining public structure is Cont replay through C, Pkg provenance through Q, and the local NameCert row N over the same BHist packet. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier primitive scope package`

Theorem 588.25 (ZetaContinuationApplication visible-boundary lock)**.** *Every accepted* ZetaContinuationApplicati *application read preserving the sibling locks keeps Gamma horizon reads in G, analytic-operation reads in A, ZetaBasic eta-source reads in E, and boundary exclusions in P, T, G, N. Such a read cannot merge those rows into a completed Gamma construction, an off-route continuation branch, a pole value, an RH statement, or a nontrivial-zero classifier.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier visible boundary lock`

Proof. Start with componentwise stability. Theorem 588.4 preserves $E, F, P, T, G, A, H, C, Q, N$ row by row, so a preserving read cannot exchange the Gamma, analytic-operation, eta-source, pole, trivial-zero, or naming rows.

Next apply the source locks. The Gamma eta handoff routes Gamma-facing reads through G, the analytic-operation boundary routes operation reads through A, and the ZetaBasic source lock routes eta-source reads through E.

Now attach route totality. Every accepted application route stays on the displayed packet and reads sibling data through the primitive scope package rather than through a second analytic continuation object.

Finally use boundary exhaustion. The pole, trivial-zero, Gamma horizon, and local naming boundaries remain in P, T, G, N, and the chapter exports no coordinate for completed Gamma data, an off-route continuation branch, a pole value, an RH statement, or a nontrivial-zero classifier. □

Theorem 588.26 (ZetaContinuationApplication root NameCert obligation surface)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the root NameCert obligation surface is the local package consisting of carrier admission, componentwise stability, ledger exactness, Gamma horizon boundary, eta factorization, analytic-operation routing, ZetaBasic source locking, primitive scope, and non-escape for the same displayed rows. Every root obligation is therefore a row obligation over* $E, F, P, T, G, A, H, C, Q, N$ *and the primitives* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root namecert obligation surface`

Proof. Start from the carrier and NameCert rows in Definition 588.1 and Definition 588.2. These identify the local packet and list the rows whose admission, stability, replay, provenance, and naming must be checked.

Next attach the carrier admission, componentwise stability, and ledger exactness theorems. They keep the accepted packet on the displayed rows and make the pole, zero, Gamma horizon, application, provenance, and naming ledgers exact for that packet.

Finally use the eta factorization, Gamma horizon, analytic-operation boundary, ZetaBasic source lock, primitive scope package, and non-escape theorems. Together they give the root obligation surface as a finite NameCert package rather than an off-table continuation theorem. □

Theorem 588.27 (ZetaContinuationApplication source route exactness)**.** *For an accepted* ZetaContinuationApplica *carrier, every read of the* Gamma$^{\uparrow}$ *horizon, analytic-operation surface, or* ZetaBasic$^{\uparrow}$ *source row factors through the displayed source routes* G*,* A*, and* E *respectively, then through componentwise transport* H*, finite replay* C*, provenance* Q*, and local naming* N*. No source read may replace these routes by a completed Gamma object, a second analytic operation, or an independent zeta continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier source route exactness`

Proof. Read the three source locks separately. The Gamma eta handoff theorem routes Gamma-facing requests through G, the analytic-operation boundary routes operation-facing requests through A, and the ZetaBasic source lock routes zeta-source requests through E.

Componentwise stability then prevents exchange among those rows. Transport through H preserves the source identities separately, while replay through C returns each request to the corresponding displayed route.

The primitive scope package repacks the same source routes with BHist, hsame, Cont, Pkg, and NameCert. The non-escape theorem excludes completed Gamma data, a second analytic operation, and an undisplayed continuation branch, so the displayed source routes are exact. □

Theorem 588.28 (ZetaContinuationApplication ledger non-escape)**.** *Every ledger-facing consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet factors through the pole ledger P, trivial-zero ledger T, Gamma horizon ledger G, application provenance Q, and local name N inside $Z = (E, F, P, T, G, A, H, C, Q, N)$. The ledger surface rules out RH, nontrivial-zero classification, completed Gamma construction, and undisplayed continuation branches.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier ledger nonescape`

Proof. Begin with ledger exactness. The pole boundary is recorded in P, the trivial-zero surface in T, the Gamma horizon in G, and the package and naming records in Q and N.

Use carrier admission and componentwise stability to keep these ledgers attached to the accepted packet. A ledger read transported by H and replayed by C returns to the same coordinate of Z.

Finally apply the primitive scope package and non-escape theorem. Since the carrier has no coordinate for RH, nontrivial-zero classification, completed Gamma construction, or an undisplayed continuation branch, no ledger-facing consumer can export one. □

Theorem 588.29 (ZetaContinuationApplication public export surface)**.** *For an accepted* ZetaContinuationApplicati *carrier, the public export surface is exactly the finite packet*

$$Z = (E, F, P, T, G, A, H, C, Q, N).$$

The row E exports the ZetaBasic$^{\uparrow}$ *eta handoff, F exports the functional-equation route, P exports the pole ledger, T exports the trivial-zero ledger, G exports the* Gamma$^{\uparrow}$ *horizon row, A exports the analytic-operation application row, H exports componentwise* hsame *transport, C exports finite* Cont *replay, Q exports* Pkg *provenance, and N exports the local* NameCert$_{\text{ZetaContinuationApplication}^{\uparrow}}$ *row. No public export coordinate lies outside those ten displayed rows.*

Proof. Project the accepted carrier through Definition 588.1. Its complete row inventory is the displayed tuple $E, F, P, T, G, A, H, C, Q, N$.

The source-lock theorems identify the public meaning of the rows. The ZetaBasic$^{\uparrow}$ source is read through E, the Gamma horizon through G, and the analytic-operation boundary through A; transport, replay, provenance, and local naming are read through H, C, Q, N.

Use Definition 588.23 for exclusion. Since every downstream read factors through the displayed inventory, no hidden continuation branch, pole value, completed Gamma construction, or nontrivial-zero classifier becomes part of the export surface. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier public export surface`

Theorem 588.30 (ZetaContinuationApplication bridge consumer boundary)**.** *Every bridge-facing or application-facing consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet factors through the* Gamma$^{\uparrow}$ *row G, the analytic-operation row A, the* ZetaBasic$^{\uparrow}$ *eta row E, componentwise transport H, replay C, provenance Q, and the local name N. The remaining public rows F, P, T may be inspected only as the displayed functional-equation, pole-ledger, and trivial-zero-ledger coordinates of the same packet.*

Proof. Begin with a bridge-facing consumer route. By Definition 588.24, sibling data enters the packet only through the primitive rows of Z and the BEDC rows BHist, hsame, Cont, Pkg, and NameCert.

The consumer must therefore read the sibling boundaries through their named coordinates. Gamma-facing requests pass through G, analytic-operation requests through A, and zeta-source eta requests through E.

The route then remains public only by using H for componentwise transport, C for replay, Q for package provenance, and N for the local certificate name. The rows F, P, T are available as internal displayed coordinates of the same route, not as independent bridge endpoints. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier bridge consumer boundary`

Theorem 588.31 (ZetaContinuationApplication public non-escape package)**.** *The public* ZetaContinuationApplicat *package is the conjunction of the non-escape theorem, the primitive scope package, and the sibling source locks for* Gamma$^{\uparrow}$*, analytic continuation, and* ZetaBasic$^{\uparrow}$*. Consequently every public consumer is confined to $E, F, P, T, G, A, H, C, Q, N$ and the package excludes RH, nontrivial-zero classification, a completed* Gamma$^{\uparrow}$ *construction, and undisplayed continuation branches.*

Proof. First combine Definition 588.29 with Definition 588.24. The public surface is the same finite carrier that the primitive scope theorem reads through BHist, hsame, Cont, Pkg, and NameCert.

Next attach the sibling source-lock theorems. The Gamma eta handoff, analytic-operation boundary, and ZetaBasic source lock keep the three external sources behind G, A, E respectively, while H, C, Q, N supply transport, replay, provenance, and naming.

Finally use Definition 588.23. The excluded objects have no displayed coordinate in the carrier, so the public package cannot export them through a bridge or application consumer. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier public nonescape package`

Theorem 588.32 (ZetaContinuationApplication root source lock)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, every root-level read of the* Gamma$^{\uparrow}$ *horizon, the analytic-operation application surface, or the* ZetaBasic$^{\uparrow}$ *eta source must pass through the displayed rows G, A, E and then through the transport, replay, provenance, and naming rows H, C, Q, N. The root packet has no source coordinate outside G, A, E, H, C, Q, N.*

Proof. Begin with the carrier definition and the NameCert obligation surface. The tuple $Z = (E, F, P, T, G, A, H, C, Q, N)$ displays the Gamma horizon row G, analytic-operation row A, eta source row E, transport row H, replay row C, provenance row Q, and local name N as named coordinates of one finite packet.

Apply the source-lock theorems. The Gamma eta handoff forces Gamma-facing reads through G, the analytic-operation boundary forces operation-facing reads through A, and the ZetaBasic source lock forces eta-source reads through E.

Finally use componentwise stability, primitive scope, and the public non-escape package. Transport and replay preserve the three source rows separately through H and C, while Q and N record only the package provenance and local name of the same carrier. Therefore every root source read is locked to G, A, E, H, C, Q, N. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root source lock`

Theorem 588.33 (ZetaContinuationApplication root ledger boundary)**.** *For an accepted* ZetaContinuationApplicat *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the root ledgers for the pole boundary, trivial-zero surface, Gamma horizon, application provenance, and local naming are exhausted by* P, T, G, Q, N*. They export no pole value, completed* $\mathsf{Gamma}^{\uparrow}$ *construction, nontrivial-zero classifier, RH statement, or ledger row outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root ledger boundary`

Proof. Start with ledger exactness. The pole ledger is P, the trivial-zero ledger is T, the Gamma horizon ledger is G, and the provenance and local naming ledgers are Q and N.

Use carrier admission and componentwise stability to keep these rows attached to the same accepted packet. A root ledger read transported by H and replayed by C returns to the corresponding coordinate of Z rather than to an external analytic object.

Now attach boundary exhaustion and the public non-escape package. The carrier has no coordinate for a regular pole value, completed Gamma construction, nontrivial-zero classifier, RH statement, or additional ledger surface, so the root ledger boundary is exactly P, T, G, Q, N. □

Theorem 588.34 (ZetaContinuationApplication root replay non-escape)**.** *For an accepted* ZetaContinuationApplica *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, every root consumer replay after the displayed carrier rows factors through* H, C, Q, N*. Such replay may inspect the source rows* E, G, A *and the public coordinates* F, P, T *only through the same finite package, and it exports no RH statement, completed* $\mathsf{Gamma}^{\uparrow}$ *construction, pole value, nontrivial-zero classifier, or off-route continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root replay nonescape`

Proof. Begin with route totality and the public export surface. They identify the complete replay-readable inventory as the displayed carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$.

Next apply the primitive scope package and bridge consumer boundary. A root consumer can move across the packet only by using componentwise hsame transport in H, finite Cont replay in C, package provenance in Q, and the local $\mathsf{NameCert}$ row N.

It remains to exclude replay escape. The public non-escape package combines the sibling source locks with boundary exhaustion, so RH, completed Gamma data, pole values, nontrivial-zero classifiers, and off-route branches have no carrier coordinate. Every root consumer replay therefore factors through H, C, Q, N over the displayed finite packet. □

Theorem 588.35 (ZetaContinuationApplication functional-row totality)**.** *For an accepted* ZetaContinuationApplica *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, every functional-equation read of the application surface remains on the displayed rows*

$$F, G, A, H, C, Q, N.$$

It may use the Gamma horizon row, analytic-operation row, componentwise transport, replay route, provenance, and local naming only as they occur in Z*; it exports no completed* $\mathsf{Gamma}^{\uparrow}$ *construction, pole value, RH statement, nontrivial-zero classifier, or off-packet continuation branch.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier functional row totality`

Proof. Begin with the local carrier and componentwise stability theorems. They project the rows F, G, A, H, C, Q, N from the accepted carrier and keep them attached under the displayed hsame transports and Cont replay.

Use ledger exactness and the Gamma horizon boundary next. The functional row F may read the Gamma horizon only through G, and the cited boundary prevents that read from becoming completed Gamma data or a pole-value computation.

Finally apply the analytic-operation boundary, ZetaBasic source lock, public non-escape package, and root replay non-escape. These results exhaust the application source route through G, A, E, H, C, Q, N and leave no coordinate for RH, nontrivial-zero classification, or an off-packet branch. □

Theorem 588.36 (ZetaContinuationApplication pole-zero ledger separation)**.** *For an accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the pole ledger* P *and the trivial-zero ledger* T *remain separate public ledgers. They can be transported and replayed only through* H, C, Q, N*, and they cannot export RH, nontrivial-zero classification, regular pole values, completed Gamma data, or a ledger row outside* P, T*.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier pole zero ledger separation`

Proof. First read the exact ledger rows. The carrier admission and ledger exactness theorems identify P as the pole ledger and T as the trivial-zero ledger, with no shared coordinate that could merge the two public records.

Next use componentwise stability and root ledger boundary. Transport through H and replay through C return to the displayed P, T, Q, N rows, so a pole read cannot become a trivial-zero classifier and a trivial-zero read cannot become a pole-value computation.

The public non-escape package and root replay non-escape then remove the off-table conclusions. Since RH, nontrivial-zero classification, completed Gamma data, regular pole values, and extra ledgers have no carrier coordinate, neither P nor T can export them. □

Theorem 588.37 (ZetaContinuationApplication source-triad exhaustion)**.** *For an accepted* ZetaContinuationApplic *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the Gamma, analytic-operation, and ZetaBasic source triad is exhausted by the rows* G, A, E *before any replay through* H, C, Q, N*. Every downstream functional-equation, pole-ledger, trivial-zero-ledger, or root consumer therefore reads the source triad through*

$$G, A, E, H, C, Q, N$$

and not through completed Gamma data, an ambient meromorphic continuation, RH, or an unrecorded zero classifier.

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier source triad exhaustion`

Proof. Start from the source locks. The Gamma eta handoff source lock, analytic operation boundary, and ZetaBasic source lock identify the only source rows available to the application packet as G, A, E.

Attach route totality and boundary exhaustion. They place each source read before the replay rows H, C, Q, N, so downstream consumers receive the triad through the same finite carrier rather than through a host analytic object.

Finally use the public non-escape package together with root replay non-escape. The exhausted triad has no coordinate for completed Gamma data, ambient meromorphic continuation, RH, or unrecorded zero classification, and every consumer route remains on G, A, E, H, C, Q, N. □

Theorem 588.38 (ZetaContinuationApplication row-inventory coverage)**.** *For every accepted* ZetaContinuationApp *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, every public read is covered by the eta source row E, the Gamma source row G, the analytic-operation source row A, the functional row F, the pole and trivial-zero ledger rows P, T, and the support rows H, C, Q, N. No additional analytic coordinate, RH-facing coordinate, ambient continuation branch, or zero-classifier row is part of the public inventory.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier row inventory coverage`

Proof. Begin with the carrier definition and the NameCert obligations. The carrier is the displayed ten-row packet $Z = (E, F, P, T, G, A, H, C, Q, N)$, and the obligation surface assigns those rows to eta source, functional equation, pole ledger, trivial-zero ledger, Gamma source, analytic-operation source, componentwise transport, replay, provenance, and local naming.

Apply root source lock, root ledger boundary, and root replay non-escape. These results exhaust source reads through E, G, A, ledger reads through P, T, and root consumer replay through H, C, Q, N, while keeping F as the displayed functional row of the same packet.

The inventory is therefore exactly the displayed carrier inventory. Any analytic branch, RH-facing conclusion, or zero-classifier output outside those rows lacks a coordinate in Z and cannot be a public read of the accepted carrier. □

Theorem 588.39 (ZetaContinuationApplication source-lock triad separation)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, the Gamma, analytic-operation, and ZetaBasic source reads remain separated through the rows G, A, E under* hsame *transport and* Cont *replay. No one source lock can substitute for another, and no replay route can identify the Gamma row, analytic-operation row, and eta source row as a single source coordinate.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier source lock triad visible separation`

Proof. Use the root source lock to locate the three source reads. Gamma-facing reads enter through G, analytic-operation reads through A, and ZetaBasic eta reads through E.

Now apply the NameCert obligations and componentwise stability. The transport row H carries hsame information componentwise, while the replay row C records Cont replay for the displayed packet rather than replacing one source row by another.

Finally combine source-triad exhaustion with root replay non-escape. Downstream routes may replay the triad only over G, A, E, H, C, Q, N, so substituting a single source lock for the other two would require an undisplayed coordinate. The accepted packet has no such coordinate. □

Theorem 588.40 (ZetaContinuationApplication root-obligation readiness)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier $Z = (E, F, P, T, G, A, H, C, Q, N)$, row-inventory coverage, source-lock triad separation, root ledger boundary, and root replay non-escape form the root-obligation packet available to downstream zeta and RH boundary chapters. The packet certifies only the displayed finite carrier interface and does not assert RH, nontrivial-zero classification, a completed* Gamma$^{\uparrow}$ *construction, or bridge verification.*

Proof. First collect the local inventory. Row-inventory coverage supplies the complete public read table $E, F, P, T, G, A, H, C, Q, N$, and the NameCert obligations attach those reads to the accepted carrier.

Next attach the separated source route and ledger boundary. Source-lock triad separation keeps G, A, E distinct under hsame and Cont, while root ledger boundary keeps pole, trivial-zero, Gamma, provenance, and naming ledgers on P, T, G, Q, N.

Finally apply root replay non-escape. Any downstream zeta or RH boundary consumer receives only the displayed root packet through H, C, Q, N, so the packet is ready as a boundary obligation without exporting RH, zero classification, completed Gamma data, or bridge verification. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root obligation readiness`

Theorem 588.41 (ZetaContinuationApplication root functional-row readiness)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the functional-equation continuation row* F *is root-ready for downstream zeta and RH boundary consumers exactly as a finite* BHist *row transported by the displayed* hsame *data and replayed through* Cont*,* Pkg*, and* NameCert*. The row is read together with the eta source row* E*, the separated Gamma and analytic-operation rows* G, A*, the pole and trivial-zero ledgers* P, T*, and the replay support* H, C, Q, N*; it does not assert RH, classify nontrivial zeros, construct a completed* Gamma$^{\uparrow}$ *object, or verify a bridge.*

Proof. Start with row-inventory coverage. It places F inside the same accepted carrier as $E, P, T, G, A, H, C, Q, N$, so a downstream consumer cannot read a functional-equation continuation row apart from the displayed packet.

Use source-lock triad separation and the root ledger boundary. The source rows G, A, E remain separated under hsame and Cont, while P, T carry only the pole and trivial-zero ledgers needed by the root interface.

Finally apply root replay non-escape and root-obligation readiness. The functional row is available as a finite replay row of Z, but the carrier has no coordinate for RH, nontrivial-zero classification, completed Gamma data, or bridge verification. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root functional row readiness`

Theorem 588.42 (ZetaContinuationApplication root Gamma handoff readiness)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the Gamma handoff available to root consumers is the separated source row* G *together with its finite replay through* H, C, Q, N *and its boundary interaction with* F, P, T, A, E*. The handoff is a* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route of the displayed packet; it is not a completed* Gamma$^{\uparrow}$ *construction, an RH assertion, a nontrivial-zero classifier, or a checked bridge.*

Proof. Use the Gamma eta handoff and source-lock triad separation. The Gamma source row is G, and the eta and analytic-operation rows remain E and A rather than substitutes for G.

Apply the root ledger boundary. The pole and trivial-zero rows P, T record the ledger side of the handoff, while F records the functional row adjacent to the Gamma source route.

Root replay non-escape then keeps the handoff on $G, F, P, T, A, E, H, C, Q, N$. Since no completed Gamma coordinate, RH coordinate, zero-classifier row, or bridge verification row is present, none can be exported by the handoff. □

Lean4 ✓ BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root gamma handoff readiness

Theorem 588.43 (ZetaContinuationApplication root public-export readiness)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *carrier* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the public export available to downstream zeta and RH boundary chapters is exactly the root packet formed by row-inventory coverage, source-lock triad separation, root ledger boundary, root replay non-escape, root functional-row readiness, and root Gamma handoff readiness. The export is anchored in the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows and refuses RH, nontrivial-zero classification, completed Gamma construction, and bridge verification.*

Lean4 ✓ BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root public export readiness

Proof. First collect the finite rows. Row-inventory coverage gives $E, F, P, T, G, A, H, C, Q, N$, while source-lock triad separation keeps the three source rows G, A, E distinct under transport and replay.

Next attach the two root handoffs. Functional-row readiness exposes F as the functional-equation continuation row of the same packet, and Gamma handoff readiness exposes G only through its finite replay and ledger boundary.

Finally use root ledger boundary, root replay non-escape, and root-obligation readiness. The public export is therefore the displayed root packet, with no coordinate for RH, nontrivial-zero classification, completed Gamma construction, or bridge verification. □

Theorem 588.44 (ZetaContinuationApplication open-phase root readiness)**.** *Any open-phase consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *application packet must expose the eta source lock* E*, Gamma horizon row* G*, analytic-operation row* A*, pole and trivial-zero ledgers* P, T*, componentwise transport* H*, replay row* C*, provenance row* Q*, and local name row* N *before it can read continuation application output. The readable output is therefore a finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route of the displayed packet.*

The readiness row asserts no RH conclusion, no analytic continuation beyond the displayed routes, no completed Gamma construction, and no standard bridge.

Lean4 ✓ BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier open phase root readiness

Proof. First use source route exactness. The application output can be reached only through the displayed source route, so the eta source lock, Gamma horizon row, and analytic-operation row must appear as the visible E, G, A portion of the carrier.

Next apply source-triad exhaustion. It prevents an open-phase consumer from collapsing E, G, A into an undisplayed source coordinate and keeps the pole and trivial-zero ledgers P, T attached to the same packet.

Finally invoke root-obligation readiness together with root public-export readiness. These rows add H, C, Q, N as the componentwise transport, replay, provenance, and local name surface needed by downstream root consumers. Since the same readiness package refuses RH, nontrivial-zero classification, completed Gamma data, and bridge verification, the open-phase root read is exactly the displayed finite obligation row. □

Theorem 588.45 (ZetaContinuationApplication root consumer exactness)**.** *Every root consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet* $Z = (E, F, P, T, G, A, H, C, Q, N)$ *reads exactly the displayed rows* $E, F, P, T, G, A, H, C, Q, N$ *through the Gamma horizon lock, analytic-operation*

lock, and ZetaBasic$^{\uparrow}$ *source lock. The consumer receives no RH statement, nontrivial-zero classifier, completed* Gamma$^{\uparrow}$ *construction, or bridge verification.*

Proof. Use open-phase root readiness to place the consumer on the displayed carrier rows. Then apply root public-export readiness and source-lock triad separation: G remains the Gamma-facing row, A remains the analytic-operation row, and E remains the ZetaBasic$^{\uparrow}$ source row.

Finally use root replay non-escape. Since the packet has no coordinate for RH, nontrivial-zero classification, completed Gamma data, or bridge verification, the root consumer surface is exactly the ten-row packet. □

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root consumer exactness`

Theorem 588.46 (ZetaContinuationApplication root pole-zero consumer readiness)**.** *Every pole- or zero-facing root consumer of an accepted* ZetaContinuationApplication$^{\uparrow}$ *packet* $Z = (E, F, P, T, G, A, H, C, Q, N)$ *reads the pole ledger* P *and trivial-zero ledger* T *only inside the same carrier route that exposes* E, F, G, A, H, C, Q, N*. The read may use the displayed Gamma, analytic-operation, and* ZetaBasic$^{\uparrow}$ *sibling locks, but it exports no pole value, RH conclusion, nontrivial-zero classification, completed* Gamma$^{\uparrow}$ *object, or bridge verification.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root pole zero consumer readiness`

Proof. Start with root ledger boundary and pole-zero ledger separation. They identify P, T as ledger rows of the same accepted packet rather than as external analytic conclusions.

Next attach root consumer exactness. The sibling locks keep Gamma reads in G, analytic-operation reads in A, and zeta source reads in E, so the ledger consumer cannot leave $E, F, P, T, G, A, H, C, Q, N$. □

Theorem 588.47 (ZetaContinuationApplication root Gamma operation handoff)**.** *For every accepted* ZetaContinuationApplication$^{\uparrow}$ *packet* $Z = (E, F, P, T, G, A, H, C, Q, N)$*, the root handoff between the Gamma horizon row and the analytic-operation row is the displayed pair* G, A *transported through* H, C, Q, N *and read against the eta source row* E*, functional row* F*, and ledgers* P, T*. This handoff remains a finite NameCert route and does not complete* Gamma$^{\uparrow}$*, assert RH, classify nontrivial zeros, or verify a bridge.*

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.ZetaContinuationApplicationCarrier root gamma operation handoff`

Proof. Apply root Gamma handoff readiness and root functional-row readiness. They expose G and F as finite rows adjacent to the application row A.

Then use root consumer exactness and root pole-zero consumer readiness. These keep the handoff inside $E, F, P, T, G, A, H, C, Q, N$ and leave RH, nontrivial-zero classification, completed Gamma data, and bridge verification outside the packet. □

Closure status for NameCert$_{\text{ZetaContinuationApplication}^{\uparrow}}$**.**

Constructive story: A ZetaContinuationApplication$^{\uparrow}$ packet is generated from eta continuation, functional-equation continuation, pole and zero ledgers, Gamma horizon rows, application routes, componentwise hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ZetaContinuationApplication}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 588.1, Definition 588.2, Definition 588.3, Definition 588.4, Definition 588.5, Definition 588.6, Definition 588.7, Definition 588.8, Definition 588.9, Definition 588.10, Definition 588.11, Definition 588.12,

Definition 588.22, Definition 588.23, Definition 588.24, Definition 588.29, Definition 588.30, Definition 588.31, Definition 588.32, Definition 588.33, Definition 588.34, Definition 588.35, Definition 588.36, Definition 588.37, Definition 588.38, Definition 588.39, and Definition 588.40 with

Lean4 ✓ `BEDC.Derived.ZetaContinuationApplicationUp.taste gate`

define the local application carrier, obligation rows, carrier admission, componentwise stability, ledger exactness, eta factorization route, Gamma horizon boundary, sibling source locks, analytic-operation boundary, route totality, boundary exhaustion, public export exactness, non-escape ledger, primitive scope package, public export surface, bridge consumer boundary, public non-escape package, root source lock, root ledger boundary, root replay non-escape, functional-row totality, pole-zero ledger separation, source-triad exhaustion, row-inventory coverage, source-lock triad separation, and root-obligation readiness.

Formal verification: FormalStatus(NameCert$_{\text{ZetaContinuationApplication}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RH, nontrivial-zero classification, full constructive Gamma$^{\uparrow}$ construction, global meromorphic uniqueness, Lean verification, or bridge checking.

Upgrade path: Paper-side closure is complete for the displayed public application packet; formal verification and bridge checking remain separate verification-axis work.

Chapter 589

A Concrete Naming Certificate for $\mathsf{CauchyNet}^{\uparrow}$

The $\mathsf{CauchyNet}^{\uparrow}$ surface records directed convergence data as finite observation packets. Kelley-style Cauchy nets and uniform-space convergence provide the mathematical precedent, but the BEDC packet below is not a quotient net, a selected limit, or an ambient completeness theorem. It is a finite BHist boundary that exposes the directed windows actually inspected by downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

The directed-window packet is a downstream consumer of chapter 126, chapter 301, chapter 291, and chapter 38: those sibling chapters supply the schedule, regular rational readback, dyadic tolerance ledger, and real-name handoff rows used in the carrier below.

Definition 589.1 (CauchyNet carrier)**.** *A* $\mathsf{CauchyNet}^{\uparrow}$ *carrier is a finite directed-window* BHist *packet*

$$K = (D, S, R, W, T, L, H, C, P, N)$$

over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *with the following displayed rows:*

1. *D is the finite directed index-window ledger, naming the inspected vertices and the common upper-window witnesses used by the packet;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that turns each requested directed window into displayed observation names;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row read from the scheduled rational observations on those windows;*
4. *W is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row for the dyadic tolerances and rational comparisons used inside the finite windows;*
5. *T is the eventual common-window classifier row, asserting that each displayed tolerance request is witnessed by one finite directed window in D;*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *handoff row through which downstream real consumers read the regular rational data;*
7. *H stores componentwise* hsame *transports for D, S, R, W, T, L and the local naming rows;*
8. *C stores the finite* Cont *routes that expose directed windows, observations, common-window witnesses, classifier reads, and real handoffs;*

9. *P is the* Pkg *provenance over the listed BEDC dependencies;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNet}^\uparrow}$ *row.*

The carrier exports only this finite directed-window packet. It contains no quotient of nets, no selected limit point, no ambient uniform-space completeness theorem, and no choice operator selecting cofinal windows.

Lean4 ✓ `BEDC.Derived.CauchyNetUp.CauchyNetCarrier transport semanticNameCert`
Lean4 ✓ `BEDC.Derived.CauchyNetUp.CauchyNetCarrier commonWindow semanticNameCert`

Theorem 589.2 (CauchyNet NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyNet}^\uparrow}$ *surface for Definition 589.1 consists of five finite obligations:*

1. *carrier admission for the displayed directed-window packet K;*

2. *eventual common-window classification through T, stating that each exposed tolerance request is routed to a common upper window recorded in D;*

3. *directed-window transport along H and C, preserving the scheduled* $\mathsf{StreamName}^\uparrow$ *observations,* $\mathsf{DyadicRatCore}^\uparrow$ *comparisons, and* $\mathsf{RegSeqRat}^\uparrow$ *handoff rows;*

4. *ledger non-escape, stating that a consumer read without a displayed common-window witness is recorded as outside the packet rather than exported as convergence data;*

5. $\mathsf{Real}^\uparrow$ *handoff through L, so real-facing consumers see only the regular rational data certified by the same finite directed-window rows.*

Lean4 ✓ `BEDC.Derived.CauchyNetUp.CauchyNetNameCert obligation surface`
↪variant `BEDC.Derived.CauchyNetUp.CauchyNetCarrier namecert obligation surface`

Proof. Carrier admission is projection from the finite tuple in Definition 589.1. The directed index-window ledger, stream schedule, dyadic observations, regular rational handoff, classifier row, real handoff, transports, routes, package provenance, and local naming row are all visible coordinates of one BHist packet.

The eventual common-window classifier reads T only after the finite directed ledger D has supplied a displayed upper-window witness. The schedule S and observation row W then provide the rational data consumed by R.

Directed-window transport is componentwise through H and replayed through C. These rows preserve the same windows, observations, classifier reads, and handoff rows; they do not add a cofinal choice function or a quotient relation on nets.

If a consumer asks for convergence data without a displayed common-window witness, no coordinate of K can supply it. The package provenance records the request as outside the finite packet boundary. The only real-facing route is therefore the handoff L through the same $\mathsf{RegSeqRat}^\uparrow$ row, which gives the stated $\mathsf{Real}^\uparrow$ boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNet}^\uparrow}$**.**

Constructive story: A $\mathsf{CauchyNet}^\uparrow$ packet is generated from a finite directed index-window ledger, a $\mathsf{StreamName}^\uparrow$ schedule, $\mathsf{DyadicRatCore}^\uparrow$ observation rows, a $\mathsf{RegSeqRat}^\uparrow$ handoff, an eventual common-window classifier, a $\mathsf{Real}^\uparrow$ handoff, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyNet}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 589.5 is the bridge-facing directed-window public export over $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, Pkg, and NameCert rows of the quotient-free packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyNet}^\uparrow}$, unformalized)

Bridge status: none

Origin: human
Not claimed: This chapter does not close quotient nets, selected limits, ambient completeness, cofinal choice, or a uniform-space standard bridge.
Upgrade path: Bridged closure requires a standard bridge for the same quotient-free directed-window packet.

Definition 589.3 (CauchyNet witness-complete finite subwindow)**.** *Let*

$$K = (D, S, R, W, T, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyNet}^{\uparrow}$ *carrier, and let* Ω *be a finite directed subwindow of the displayed index-window ledger* D*. The subwindow* Ω *is witness-complete for a displayed finite tolerance family* E *when the retained rows*

$$D_\Omega, S_\Omega, R_\Omega, W_\Omega, T_\Omega, L_\Omega, H_\Omega, C_\Omega, P_\Omega, N_\Omega$$

satisfy the following conditions:

1. D_Ω *is the finite directed subledger of* D *on* Ω*, and for every tolerance request in* E *the common upper-window witness named by the original classifier row* T *lies in* D_Ω*;*
2. S_Ω *contains exactly the scheduled* $\mathsf{StreamName}^{\uparrow}$ *observation names from* S *whose directed addresses are in* D_Ω *and are used by those retained common-window witnesses;*
3. W_Ω *contains exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and comparison rows from* W *read by the retained scheduled observations;*
4. T_Ω *is the common-window classifier row restricted to the requests in* E*, with each classifier entry using its displayed witness from* D_Ω*;*
5. R_Ω *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row obtained by restricting the original row* R *to the retained schedule, observation, comparison, and classifier entries;*
6. L_Ω *is the* $\mathsf{Real}^{\uparrow}$ *handoff row whose source is the retained row* R_Ω*;*
7. H_Ω *and* C_Ω *are the componentwise restrictions of the displayed* hsame *transports and* Cont *routes whose endpoints are among the retained rows;*
8. P_Ω *and* N_Ω *are the provenance and local* $\mathsf{NameCert}_{\mathsf{CauchyNet}^{\uparrow}}$ *rows restricted to the same retained coordinates.*

Lean4 (definition) `BEDC.Derived.CauchyNetUp.CauchyNetWitnessCompleteSubwindow`

Theorem 589.4 (CauchyNet finite subwindow restriction)**.** *Let*

$$K = (D, S, R, W, T, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyNet}^{\uparrow}$ *carrier in the sense of Definition 589.1. If* Ω *is witness-complete for a displayed finite tolerance family* E *in the sense of Definition 589.3, then*

$$K_\Omega = (D_\Omega, S_\Omega, R_\Omega, W_\Omega, T_\Omega, L_\Omega, H_\Omega, C_\Omega, P_\Omega, N_\Omega)$$

is an accepted $\mathsf{CauchyNet}^{\uparrow}$ *carrier. Moreover, every* $\mathsf{Real}^{\uparrow}$ *read exported by* L_Ω *factors through the retained* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R_Ω*, and* R_Ω *is the restriction of the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *used by the original packet.*

Lean4 ✓ `BEDC.Derived.CauchyNetUp.CauchyNetWitnessCompleteSubwindow carrier restriction`

Proof. Unfold the carrier of Definition 589.1. A $\mathsf{CauchyNet}^{\uparrow}$ carrier has exactly the displayed rows

$$D, S, R, W, T, L, H, C, P, N.$$

The witness-complete subwindow supplies one retained row for each of these coordinates. Thus K_Ω has the required directed index-window ledger, schedule, regular rational row, dyadic observation row, common-window classifier, real handoff, componentwise transports, continuation routes, provenance, and local naming row.

The common-window obligation remains internal to the retained packet. For every tolerance request in E, witness-completeness keeps the common upper-window witness selected by T inside D_Ω. It also keeps the scheduled observation names in S_Ω and the dyadic comparison rows in W_Ω that are read at that witness. Hence the restricted classifier row T_Ω does not appeal to any vertex, observation, or comparison outside the finite subwindow.

The regular rational row is not recomputed from an external net. By definition of witness-completeness, R_Ω is the restriction of the original $\mathsf{RegSeqRat}^{\uparrow}$ row R to the retained schedule, observation, comparison, and classifier entries. Therefore each retained regular rational read is attached to the same finite directed-window evidence that justified it in K.

The transport and continuation rows also remain closed under restriction. The rows H_Ω and C_Ω retain only componentwise hsame transports and Cont routes whose endpoints are among $D_\Omega, S_\Omega, R_\Omega, W_\Omega, T_\Omega$, and L_Ω. The package and local naming rows P_Ω and N_Ω name precisely those retained coordinates, so no hidden cofinal choice function, quotient of nets, selected limit point, or ambient completeness theorem is introduced.

Apply the obligation surface of Definition 589.2. Carrier admission is the displayed restricted tuple. Eventual common-window classification is supplied by T_Ω because all requested witnesses remain in D_Ω. Directed-window transport is supplied by H_Ω and C_Ω. Ledger non-escape holds because a request outside E or outside the retained witness ledger has no coordinate in K_Ω from which convergence data can be exported. Finally, the only real-facing row is L_Ω, and its source is exactly R_Ω. Thus every $\mathsf{Real}^{\uparrow}$ handoff factors through the same restricted $\mathsf{RegSeqRat}^{\uparrow}$ row, with R_Ω inherited from R. □

Theorem 589.5 (CauchyNet directed-window localization)**.** *Let*

$$K = (D, S, R, W, T, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyNet}^{\uparrow}$ *carrier, and let* Ω *be witness-complete for a displayed finite tolerance family* E *in the sense of Definition 589.3. Then every* $\mathsf{Real}^{\uparrow}$ *handoff exported by the restricted row* L_Ω *is determined by the retained directed-window rows*

$$D_\Omega, S_\Omega, R_\Omega, W_\Omega, T_\Omega, H_\Omega, C_\Omega, P_\Omega, N_\Omega.$$

No omitted directed vertex, cofinal choice function, quotient net, selected limit point, or ambient completeness row can change that handoff.

Proof. Unfold the witness-complete subwindow definition. The retained ledger D_Ω contains every common upper-window witness used by the displayed tolerance family E, while S_Ω and W_Ω contain exactly the scheduled observation names and dyadic comparison rows read at those witnesses. The classifier row T_Ω therefore has no address outside the retained directed window from which to read convergence evidence.

Apply Definition 589.4. The restricted tuple

$$K_\Omega = (D_\Omega, S_\Omega, R_\Omega, W_\Omega, T_\Omega, L_\Omega, H_\Omega, C_\Omega, P_\Omega, N_\Omega)$$

is itself an accepted $\mathsf{CauchyNet}^{\uparrow}$ carrier, and every real-facing read exported by L_Ω factors through the retained $\mathsf{RegSeqRat}^{\uparrow}$ row R_Ω.

Apply the obligation surface of Definition 589.2 to the accepted carrier K_Ω. Carrier admission is the displayed restricted tuple; eventual common-window classification is T_Ω; directed-window transport is supplied by H_Ω and C_Ω; ledger non-escape is enforced by the absence of any coordinate outside the retained rows; and the real-facing boundary is precisely the handoff through L_Ω from R_Ω.

It remains only to exclude external data. The subwindow definition gives no coordinate for an omitted directed vertex, a cofinal selector, a quotient of nets, a selected limit point, or an ambient completeness theorem. Since the only exported real-facing route is the projection through R_Ω and L_Ω inside K_Ω, changing any such omitted datum cannot affect the $\mathsf{Real}^{\uparrow}$ handoff. □

Lean4 ✓ `BEDC.Derived.CauchyNetUp.CauchyNetDirectedWindow localization`

Chapter 590

A Concrete Naming Certificate for $\mathsf{ClassifierMorphism}^{\uparrow}$

The $\mathsf{ClassifierMorphism}^{\uparrow}$ surface records the classifier-preserving layer between type-checking membership packets and broader naming-certificate morphisms. It keeps the source and target classifier packets visible, stores the graph that moves source histories to target histories, and exposes preservation only through finite Ext, SigRel, Cont, hsame, Pkg, and NameCert rows.

Definition 590.1 (ClassifierMorphism carrier)**.** *A* $\mathsf{ClassifierMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, G, E, S, C, H, P, N)$$

with the following displayed coordinates:

1. *A is the source classifier-membership packet, including its carrier,* Ext*,* SigRel*,* Cont*, and local naming rows;*
2. *B is the target classifier-membership packet, including the corresponding target carrier, classifier, signature-answer, continuation, and naming rows;*
3. *G is the finite graph row relating source histories in A to target histories in B;*
4. *E records* Ext *preservation along G, sending every accepted source membership row to an accepted target membership row;*
5. *S records* SigRel *preservation along G, sending every displayed source classifier answer to the corresponding target classifier answer;*
6. *C records* Cont *preservation, routing every source continuation edge through a target continuation edge over the same graph row;*
7. *H stores componentwise* hsame *preservation for source and target carrier rows, membership rows, signature rows, continuation rows, graph rows, and local certificate rows;*
8. *P is the* Pkg *provenance over* BHist*,* Ext*,* SigRel*,* Cont*,* hsame*, the two classifier packets, and the graph row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host compiler function, quotient equality between source and target histories, semantic type interpretation, or any relation not named by G, E, S, C, H, P, N.

Definition 590.2 (ClassifierMorphism classifier)**.** *For accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *carriers* $M = (A, B, G, E, S, C, H, P, N)$ *and* $M' = (A', B', G', E', S', C', H', P', N')$*, the classifier compares exactly the displayed graph row and its preservation ledgers. It may inspect source and target packets only through their visible* Ext*,* SigRel*,* Cont*,* hsame*,* Pkg*, and* NameCert *coordinates, and it accepts a comparison precisely when the preservation rows remain componentwise readable through the displayed* hsame *and* Cont *routes.*

Thus classifier morphisms are finite packets over classifier membership data. They do not identify source and target histories by host equality, and they do not collapse into the larger compiler-certificate surface unless a separate $\mathsf{NameCertMorphism}^{\uparrow}$ *packet supplies the extra compiler boundary.*

Lean4 (definition) `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacketClassifier`

Theorem 590.3 (ClassifierMorphism NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$ *surface consists of the carrier row of Definition 590.1, the classifier row of Definition 590.2, the* Ext *preservation row* E*, the* SigRel *preservation row* S*, the* Cont *preservation row* C*, the componentwise* hsame *preservation row* H*, the dependency package* P*, and the local naming row* N*.*

Every consumer-visible classifier-morphism read must therefore pass through the finite source packet A*, target packet* B*, graph row* G*, and the preservation ledgers* E, S, C, H, P, N*. The packet exports no host compiler function, semantic typing model, quotient equality, or unrecorded relation between the source and target classifier memberships.*

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismUp taste gate boundary`

Proof. Project the carrier tuple of Definition 590.1. The only rows available to the local certificate are the source packet, target packet, graph row, three preservation ledgers, transport ledger, provenance row, and local name.

Use the classifier of Definition 590.2. It compares accepted morphism packets by reading those same rows componentwise, so a consumer cannot replace the displayed graph with a host-level function or a semantic relation.

The local certificate is obtained by repacking $A, B, G, E, S, C, H, P, N$ as one $\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$ row. Since the carrier contains no other coordinate, the obligation surface is exhausted by those finite preservation packets. □

Theorem 590.4 (ClassifierMorphism Ext and SigRel preservation)**.** *Let* $M = (A, B, G, E, S, C, H, P, N)$ *be an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packet. If a source history is accepted by the source* Ext *row of* A *and is related by the graph row* G *to a target history in* B*, then every classifier-visible read of that target history factors through the target* Ext *preservation row* E *and the target* SigRel *preservation row* S*, with the same* hsame*,* Cont*,* Pkg*, and* NameCert *boundary* H, C, P, N*.*

Consequently the morphism preserves membership answers and signature-answer rows before any broader compiler certificate can consume the graph.

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismUp ext sigrel preservation`

Proof. Begin with the source membership read. The graph coordinate G is the only row relating the source and target histories, and the preservation coordinate E is the only row that turns source Ext membership into target Ext membership.

Next read the signature answer. The row S is attached to the same graph coordinate, so the target SigRel answer is reached through the same source and target classifier packets rather than through an independent typing semantics.

Finally route both reads through H, C, P, N. These rows record componentwise sameness, continuation access, dependency provenance, and local naming for the same finite packet, so the classifier-visible preservation surface is fixed before any $\mathsf{NameCertMorphism}^{\uparrow}$ compiler boundary is applied. □

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket ext sigrel preservation`

Theorem 590.5 (ClassifierMorphism source-target route lock)**.** *Let $M = (A, B, G, E, S, C, H, P, N)$ be an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packet. Any classifier-preserving read from the source packet A to the target packet B must pass through the same displayed graph row G,* Ext *row E,* SigRel *row S,* Cont *row C, componentwise* hsame *row H, package row P, and local* NameCert *row N.*

In particular, the route cannot split Ext *preservation and* SigRel *preservation into separate host functions, cannot replace the graph row by an unrecorded semantic relation, and cannot consume either preservation row outside the displayed source-target packet.*

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket source target route lock`

Proof. Project the accepted carrier using Definition 590.1. The only source-target transport row is G, and the only preservation ledgers attached to it are E, S, C, H, P, and N.

Use the classifier of Definition 590.2. It can inspect source and target packets only through their visible Ext, SigRel, Cont, hsame, Pkg, and NameCert coordinates, so a classifier-preserving read has no independent host function available.

Apply Definition 590.4. It binds target Ext and target SigRel reads to the same graph coordinate and the same support rows.

Finally use Definition 590.3. The local certificate surface is exhausted by $A, B, G, E, S, C, H, P, N$, so the accepted read is locked to that displayed packet and cannot factor through an unrecorded semantic relation. □

Theorem 590.6 (ClassifierMorphism type-membership dependency)**.** *The source packet A and target packet B of an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *carrier are classifier-membership surfaces in the sense of chapter 476. The morphism rows G, E, S, C, H, P, N preserve* Ext*,* SigRel*, and* Cont *reads between those packets, and the preservation route cannot be read as a host compiler function, semantic type interpretation, quotient equality, or unrecorded source-target relation.*

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket type membership dependency`

Proof. Read the carrier definition. The rows A and B are explicitly the source and target classifier-membership packets, including their carrier, classifier, signature-answer, continuation, and local naming rows.

Use the classifier definition. A comparison can inspect those packets only through the displayed Ext, SigRel, Cont, hsame, Pkg, and NameCert coordinates, so the graph row G is not a host-level compiler function.

Finally cite the TypeClassifierMembership chapter. It supplies the sibling membership surface for finite type-checking packets, while the Ext and SigRel preservation theorem keeps the route inside G, E, S, C, H, P, N. □

Theorem 590.7 (ClassifierMorphism composition closure)**.** *Given accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packets from a classifier-membership packet A to a packet B and from B to a packet D, their relational graph composite determines an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packet from A to D. The composed packet preserves* Ext*,* SigRel*,* Cont*, and* hsame *by composing the displayed graph rows,*

preservation ledgers, continuation routes, dependency packages, and local naming rows of the two input packets.

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket composition closure`

Proof. Project the first carrier to its source packet, middle packet, graph row, and preservation ledgers. Project the second carrier to the same kind of rows with the first target packet as its source-facing middle packet.

Compose the graph rows relationally through the shared middle packet. For an accepted source membership row, apply the first Ext preservation ledger to reach the middle packet, then apply the second Ext preservation ledger to reach the final packet.

The same concatenation applies to SigRel answers and Cont routes. Componentwise hsame rows compose through the shared middle packet, while the two Pkg and NameCert rows repack the resulting graph and preservation ledgers as one finite $\mathsf{ClassifierMorphism}^{\uparrow}$ carrier. □

Theorem 590.8 (ClassifierMorphism public composition export)**.** *For accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packets from classifier-membership packet A to packet B and from B to packet D, the two-step classifier-preserving route exports a single public* $\mathsf{ClassifierMorphism}^{\uparrow}$ *route from A to D. Its graph,* Ext *preservation,* SigRel *preservation,* Cont *preservation,* hsame *transport,* Pkg *provenance, and local* NameCert *rows are exactly the relational composites of the corresponding displayed rows in the two input packets.*

Thus a downstream naming-certificate morphism consumer may cite one public route from A to D rather than separately reconstructing graph composition, preservation of classifier reads, continuation replay, package provenance, and local naming.

Proof. First build the route. Apply Definition 590.7 to the two accepted packets through the shared middle packet B. The resulting carrier has source A, target D, and graph row given by relational composition of the two displayed graph rows.

Next identify the preservation surface. Use Definition 590.4 on each input packet and on the composed carrier. The Ext and SigRel reads are therefore the composite preservation ledgers, while the Cont and hsame rows are the corresponding composite replay and transport rows.

Finally repack the public export through the local boundary. The NameCert obligation surface and the checked TasteGate boundary recorded for Definition 590.3 expose only the rows G, E, S, C, H, P, N. Since those rows in the composed carrier are precisely the relational composites supplied by Definition 590.7, the exported route contains no extra host function, semantic interpretation, quotient equality, or unrecorded relation. □

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket public composition export`

Theorem 590.9 (ClassifierMorphism preservation non-escape scope)**.** *Every classifier-preserving consumer route for an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packet factors through exactly the displayed source packet A, target packet B, graph row G,* Ext *preservation row E,* SigRel *preservation row S,* Cont *preservation row C, componentwise* hsame *row H,* Pkg *provenance row P, local* NameCert *row N, and the checked TasteGate boundary. The route exports no host compiler function, semantic type interpretation, quotient equality between source and target histories, or unrecorded source-target relation.*

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket preservation nonescape scope`

Proof. Start with the obligation surface of Definition 590.3. It identifies the carrier rows $A, B, G, E, S, C, H, P, N$ as the complete local $\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$ inventory.

Use Definition 590.4 for classifier-preserving reads. The source membership read passes through G and E, while the signature-answer read passes through the same graph and the preservation row S.

Use Definition 590.7 for consumer routes that pass through more than one classifier morphism. Composition only concatenates displayed graph rows, preservation ledgers, continuation routes, package rows, and local naming rows.

The checked TasteGate boundary records the same finite packet as an encoded gate surface. Since neither the carrier nor the TasteGate boundary contains a host compiler function, semantic type interpretation, quotient equality, or unrecorded relation, every classifier-preserving route remains inside the displayed scope. □

Theorem 590.10 (ClassifierMorphism kernel compiler route exhaustion)**.** *Every kernel-compiler or naming-certificate-morphism consumer of an accepted* $\mathsf{ClassifierMorphism}^{\uparrow}$ *packet factors through exactly the displayed source packet A, target packet B, graph row G,* Ext *preservation row E,* SigRel *preservation row S,* Cont *preservation row C, componentwise* hsame *transport H, package provenance P, and local* NameCert *row N. The consumer receives no host compiler function, semantic type interpretation, quotient equality between source and target histories, or unrecorded source-target relation.*

Proof. Start with the local inventory. Apply Definition 590.3; it lists $A, B, G, E, S, C, H, P, N$ as the complete carrier and naming-certificate surface available to a $\mathsf{ClassifierMorphism}^{\uparrow}$ packet.

Next route classifier reads through the preservation rows. Use Definition 590.4 to force Ext and SigRel consumers through G, E, S, and use the displayed Cont and hsame rows for continuation replay and componentwise history transport.

For compiler-style consumers that compose routes, apply Definition 590.7. Composition concatenates graph rows, preservation ledgers, continuation routes, package rows, and local naming rows; it does not create an additional host compiler coordinate or semantic type interpretation.

Finally use Definition 590.9. The non-escape row removes quotient equality and unrecorded source-target relations from the consumer interface, so the kernel-compiler route is exhausted by the displayed packet rows. □

Lean4 ✓ `BEDC.Derived.ClassifierMorphismUp.ClassifierMorphismPacket kernel compiler route exhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClassifierMorphism}^{\uparrow}$ packet is generated from source and target classifier-membership packets, a finite graph row, Ext, SigRel, Cont, and hsame preservation rows, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 590.1, Definition 590.2, Definition 590.3, Definition 590.4, Definition 590.7, and Definition 590.9 bind the finite classifier-preservation packet between type-checking membership and naming-certificate morphism surfaces.

Lean target: `BEDC.Derived.ClassifierMorphismUp.taste\_gate`

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClassifierMorphism}^{\uparrow}}$, encodedDef)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host compiler semantics, quotient equality between source and target histories, semantic typing interpretation, or bridge checking.

Upgrade path: The paper-axis closure is complete; formal-status and bridge checking remain separately reported by the verification and bridge axes.

Chapter 591

A Concrete Naming Certificate for $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$

The $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ interface realizes the runtime-candidate boundary of Definition 9461.21 as a finite NameCert packet. It records object-level rows for coded CIC* expressions, proof objects, states, transitions, accepted-result classification, stability transports, consumer routes, package provenance, and local naming, while leaving validation by the host checker outside the packet. The bounded finite-trace evaluator of chapter 818 consumes this boundary by reading the object-level source, transition, accepted-result, transport, route, provenance, and local naming rows while keeping host validation external.

The packet is the object-level candidate boundary consumed by chapter 818 and sits below the runtime-request and refusal packet of chapter 890. The substantive dependency is the row split: $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ supplies the source, transition, accepted-result, transport, route, provenance, and local-name rows $R = (S, T, A, H, C, P, N)$; $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$ adds request, refusal, and report rows above that candidate surface; and $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ consumes the candidate boundary as its bounded evaluator provenance.

Definition 591.1 (Runtime-candidate boundary carrier)**.** *A* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, A, H, C, P, N)$$

with the following visible rows:

1. *S is the source row of coded* CIC* *expressions, proof objects, and states visible to the candidate;*
2. *T is the transition row recording the checking or evaluation steps claimed by the candidate on the source row S;*
3. *A is the accepted-result classifier row, grouping transition outputs that the candidate records as accepted results;*
4. *H stores componentwise* hsame *transports for the source, transition, accepted-result, package, and naming rows;*
5. *C stores the displayed* Cont *routes by which consumers read the source codes, transition rows, accepted-result classifier, transports, provenance, and local naming row;*
6. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the structural* CIC* *coding interface;*

7. *N is the local* $\mathsf{NameCert}_{\mathsf{RuntimeCandidateBoundary}^{\uparrow}}$ *row.*

The local classifier relates two packets exactly when their displayed source, transition, accepted-result, transport, route, provenance, and naming rows compare through the rows H and C. The carrier has no coordinate for the host judgment that the candidate is a valid BEDC checker.

Lean4 ✓ `BEDC.Derived.RuntimeCandidateBoundaryUp.RuntimeCandidateBoundaryTasteGate single carrier alignment`

Theorem 591.2 (Runtime-candidate validation non-entry)**.** *For every accepted* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *carrier $R = (S, T, A, H, C, P, N)$, validation of the candidate is not one of the packet rows corresponding to* $\mathsf{SourceSpec}_R$, $\mathsf{PatternSpec}_R$, $\mathsf{ClassifierSpec}_R$, $\mathsf{StabilityCert}_R$, *or* $\mathsf{LedgerPolicy}_R$. *It is therefore not obtainable as a source code in S, a transition pattern in T, an accepted-result classifier entry in A, a stability transport in H, or a ledger/provenance row in P.*

Proof. Project the accepted packet to its visible rows. The source field is exhausted by S: it contains coded CIC* expressions, proof objects, and states, but no judgment that those codes form a correct checker for BEDC.

The pattern and classifier fields are exhausted by T and A. They record transition behavior and accepted-result grouping internal to the candidate. Reading either row can expose what the candidate claims for a coded input; it cannot certify that the candidate itself is the host validation act.

The stability, route, and package fields are exhausted by H, C, P, N. These rows transport, read, name, and record provenance for the displayed finite packet. If validation were placed in any one of them, it would be another coded row of the same candidate and would again require validation as a certificate. Thus the carrier remains an object-level runtime candidate rather than a self-validating runtime. □

Theorem 591.3 (Runtime-candidate boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RuntimeCandidateBoundary}^{\uparrow}}$ *surface for the carrier of Definition 591.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet $R = (S, T, A, H, C, P, N)$;*
2. *classifier transport by componentwise* hsame *and* Cont *on the source, transition, accepted-result, package, and local naming rows;*
3. *runtime-result ledger exactness, requiring every accepted-result row to be exactly a displayed row of A reached from a displayed transition row of T;*
4. *validation non-escape, requiring every claim that the candidate is valid to remain outside S, T, A, H, C, P, N rather than appearing as a hidden packet coordinate;*
5. *reflection-boundary handoff, requiring the packet to cite the structural* CIC* *interpretation of Definition 9461.11 and the runtime-candidate boundary of Definition 9461.22 without promoting either citation to an internal host validator.*

Consequently the certificate names the object-level runtime-candidate interface and its finite result ledger, but it does not assert an internal truth predicate or a host checker inside BEDC.

Proof. Carrier admission is tuple projection. The packet $R = (S, T, A, H, C, P, N)$ is finite, and each displayed row is part of the same BHist carrier.

Classifier transport is componentwise. The hsame row H compares displayed source, transition, accepted-result, package, and naming rows, while the Cont row C records exactly the reads available to a consumer.

Runtime-result ledger exactness follows by reading T before A. A row can count as an accepted result only when the carrier displays the transition claim in T and the corresponding accepted-result classifier entry in A.

Validation non-escape is the boundary clause. By Definition 591.2, validation is not one of the source, pattern, classifier, stability, or ledger rows. The final handoff is therefore only a citation to the structural CIC* interpretation and to the runtime-candidate boundary, not an internal host validator. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuntimeCandidateBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ packet is generated from displayed source, transition, accepted-result, componentwise transport, visible route, package provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuntimeCandidateBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 591.1, Definition 591.2, Definition 591.3, and

Lean4 ✓ `BEDC.Derived.RuntimeCandidateBoundaryUp.taste gate`

bind the displayed runtime-candidate packet rows, validation boundary, local certificate surface, and public object-level runtime-candidate export consumed by finite-trace evaluators.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuntimeCandidateBoundary}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RuntimeCandidateBoundaryUp.RuntimeCandidateBoundaryTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host validation, an internal truth predicate, a self-validating runtime, or bridge checking.

Upgrade path: No paper-theory closure upgrade remains; formal bridge checking is a separate verification-axis target over the same source, transition, accepted-result, transport, route, provenance, and local naming rows.

Chapter 592

A Concrete Naming Certificate for FreeWillInscriptionCommitment$^{\uparrow}$

The FreeWillInscriptionCommitment$^{\uparrow}$ surface packages the free-will reading of Definition 9415.6 as finite NameCert data. It records the prior-record boundary, the inscription-event row, GAP provenance, classifier transport, and the local non-reduction ledger that prevents the packet from becoming a hidden deterministic reducer, a selected novelty source, or a host-side theory of agency.

The commitment packet consumes the inscription-event carrier of chapter 519 and the gap-ledger surface of chapter 520. In the displayed carrier below, I is the event row, G is the gap provenance row, and R records that consumers may compare the event with the prior boundary without extracting a closed reducer from B to I.

Theorem 592.1 (FreeWillInscriptionCommitment inscription-event route)**.** *Every accepted* FreeWillInscriptionCom *packet reads its inscription-event coordinate I through the finite* InscriptionEvent$^{\uparrow}$ *surface of chapter 519 before the non-reduction ledger R is exposed. The route records prior boundary, inscription row, GAP provenance, transport, continuation, package, and local name, and it exports no hidden agency source or deterministic reducer.*

Lean4 ✓ `BEDC.Derived.FreeWillInscriptionCommitmentUp.FreeWillInscriptionCommitment inscription event route`

Proof. Project the accepted carrier as $W = (B, I, G, T, R, H, C, P, N)$. The coordinate I is the only inscription-event row in the packet, and it is displayed together with the prior boundary B and GAP provenance G before the ledger row R is read.

Read I through the concrete InscriptionEvent$^{\uparrow}$ surface named by chapter 519. The transport row T, componentwise hsame row H, displayed Cont row C, package row P, and local naming row N replay that same finite event route.

Finally apply the non-escape boundary of Definition 592.4. Since the carrier has no coordinate for a hidden agency source or a deterministic reducer from B to I, exposing R cannot add one after the inscription-event read. □

Definition 592.2 (FreeWillInscriptionCommitment carrier)**.** *A* FreeWillInscriptionCommitment$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, I, G, T, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the prior-record boundary, listing the records available before the inscription event;*
2. *I is the inscription-event row, naming the finite record added at the event point;*
3. *G is the* GAP *provenance row, recording that the event is treated as an inscription point rather than as a substrate-internal derivation from B;*
4. *T is the classifier-transport row, carrying the visible comparison between prior boundary, inscription event, and provenance data;*
5. *R is the non-reduction ledger, recording that consumer reads may compare the event with B but may not extract a closed reducer from B to I;*
6. *H records componentwise* hsame *transports for B, I, G, T, R, package provenance, and the local naming row;*
7. *C records the* Cont *routes that expose the prior boundary, inscription event, GAP provenance, classifier transport, and non-reduction ledger without adding an unrecorded agency primitive;*
8. *P is the* Pkg *row for the finite BEDC dependencies named by the commitment;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FreeWillInscriptionCommitment}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no hidden faculty of will, no selected novelty source, no deterministic reducer from the prior boundary to the inscription event, no host psychology predicate, and no ambient metaphysical agency oracle.

Lean4 ✓ `BEDC.Derived.FreeWillInscriptionCommitmentUp.FreeWillInscriptionCommitmentTasteGate single carrier alignment`

Theorem 592.3 (FreeWillInscriptionCommitment NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FreeWillInscriptionComm}}$ *surface has five finite obligations: carrier exposure by Definition 592.2; prior-boundary accountability for the displayed record row B; inscription-event accountability for the displayed event row I and GAP provenance row G; classifier stability for the visible transport rows T, H, C and package row P; and non-escape from hidden faculties of will, selected novelty sources, deterministic reducers, host psychology predicates, and ambient metaphysical agency oracles.*

Lean4 ✓ `BEDC.Derived.FreeWillInscriptionCommitmentUp.FreeWillInscriptionCommitment namecert obligations`

Proof. Carrier exposure is projection from the finite tuple $W = (B, I, G, T, R, H, C, P, N)$. Every consumer-facing coordinate is one of the displayed rows.

Prior-boundary accountability reads B, while inscription-event accountability reads I together with the GAP provenance row G. These two obligations keep the event located at the finite boundary named by the packet.

Classifier stability is componentwise through T, H, C, P. The transport rows compare and replay the visible boundary, event, provenance, and ledger data; they do not add a hidden decision procedure or an unrecorded source of novelty.

The non-escape obligation is the inventory boundary. The carrier has no coordinate for a hidden faculty, selected novelty source, deterministic reducer, host psychology predicate, or ambient agency oracle, so the local NameCert row cannot export one. □

Theorem 592.4 (FreeWillInscriptionCommitment non-escape boundary)**.** *No consumer of an accepted* $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$ *packet can extract a hidden faculty of will, a selected*

novelty source, a deterministic reducer from the prior-record boundary to the inscription-event row, a host psychology predicate, or an ambient metaphysical agency oracle. Every consumer read factors through $B, I, G, T, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.FreeWillInscriptionCommitmentUp.FreeWillInscriptionCommitment nonescape`

Proof. Inspect the carrier coordinates of Definition 592.2. The prior-record boundary, inscription-event row, GAP provenance, classifier transport, non-reduction ledger, componentwise transports, continuations, package provenance, and local naming row are the whole exported surface.

The rows T, H, C transport only the displayed rows. They may compare the prior boundary with the inscription event and replay the GAP provenance, but they have no coordinate that computes I from B as a closed substrate-internal reducer.

The row R records the non-reduction ledger rather than a hidden source. Since no remaining row names a faculty of will, selected novelty source, host psychology predicate, or ambient agency oracle, every such attempted read has no projection target in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreeWillInscriptionCommitment}^{\uparrow}}$.

Constructive story: A $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$ packet is generated from a prior-record boundary, an inscription-event row, GAP provenance, classifier transport, a non-reduction ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeWillInscriptionCommitment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 592.2, Definition 592.3, and Definition 592.4 bind the finite inscription-event commitment surface and its non-reduction boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeWillInscriptionCommitment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge status, hidden faculty of will, selected novelty source, deterministic reducer, host psychology predicate, or ambient metaphysical agency oracle.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for NameCert obligations and the non-escape boundary.

Chapter 593

A Concrete Naming Certificate for $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$

The $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ interface records the finite witness schedule used by the real-completion diagonal construction when a regular Cauchy sequence supplies an explicit modulus. It packages the selected modulus row, the finite stream window, the dyadic observation ledger, the diagonal readback, and the $\mathsf{Real}^{\uparrow}$ seal handoff as one local naming certificate. It does not export a quotient stream, a selected completed real, host real equality, or ambient completeness.

Definition 593.1 (Regular Cauchy witness schedule carrier)**.** *A* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

with the following displayed coordinates:

1. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row for the regular Cauchy input;*
2. *μ is the explicit modulus row selecting the regular-Cauchy source index for a requested tolerance;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule at the selected modulus index;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the endpoint, radius, tolerance, and comparison rows used by the scheduled read;*
5. *R is the diagonal readback row naming the rational observation read from X_μ through W and D;*
6. *T is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached after the displayed diagonal readback;*
7. *H stores componentwise* hsame *transport for the family row, modulus row, window schedule, dyadic ledger, readback row, seal row, provenance, and local name;*
8. *C stores the displayed* Cont *routes from the modulus request to W, from W to D, from D to R, and from R to T;*
9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$ *row.*

The carrier is a finite witness packet for the modulus-selected diagonal observation route. It contains no unbounded stream object, quotient equality, selected limit, host real equality, or completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate single carrier alignment`

Theorem 593.2 (Regular Cauchy witness schedule NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSc}}$ *surface for the carrier of Definition 593.1 consists exactly of carrier admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *family admission, explicit modulus exposure, selected* $\mathsf{StreamName}^{\uparrow}$ *window scheduling,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation-ledger readback, diagonal readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleSemanticNameCert`

Proof. Carrier admission is projection from the displayed packet S. Its coordinates list the regular Cauchy family row, selected modulus row, finite window schedule, dyadic ledger, diagonal readback, seal handoff, transports, routes, provenance, and local name.

The modulus and window rows make the diagonal observation finite. A consumer first reads the tolerance through μ, then opens only the scheduled window W, and then reads the dyadic ledger D and diagonal row R.

The remaining rows transport and name the same route. Since the packet has no coordinate for quotient streams, selected completed reals, host equality, or ambient completeness, the listed rows exhaust the local naming certificate. □

Theorem 593.3 (Regular Cauchy witness schedule window-modulus alignment)**.** *For an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *carrier, every scheduled window read is aligned with the displayed modulus row* μ*. If a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer asks for the diagonal observation at a tolerance already routed through* μ*, the only admissible window is the stored finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *at that selected modulus index, transported through* H *and replayed through* C*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleCarrier window modulus alignment`

Proof. Project the accepted carrier. The only source-index selector is the modulus row μ, and the only scheduled window attached to that selector is W.

Use the local NameCert obligations to bind the consumer read to the displayed coordinates. The continuation route from the modulus request to the window has no alternative branch, and the transport rows preserve the selected index and window together.

Thus the read cannot replace W by an independent schedule or inspect an unbounded stream. The finite window is aligned with the displayed modulus row by the carrier itself. □

Theorem 593.4 (Regular Cauchy witness schedule diagonal readback)**.** *For an accepted* RegularCauchyWitnessSche *carrier, the diagonal readback row* R *is read only through the route*

$$X, \mu, W, D, H, C, P, N.$$

It records the rational observation obtained from the $\mathsf{RegSeqRat}^{\uparrow}$ *family row at the modulus-selected source index and the corresponding finite window, with the dyadic observation ledger* D *supplying the displayed tolerance and comparison rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleCarrier diagonal readback lock`

Proof. Begin with the carrier projection. The row R appears after X, μ, W, and D, so any readback must first pass through the source family, the selected modulus row, the finite window, and the dyadic ledger.

Apply window-modulus alignment to keep the scheduled read on the same selected index. Then use the displayed continuation route from W to D and from D to R.

The transport and provenance rows repack this same readback route. No carrier coordinate offers a quotient stream, a host real comparison, or a separate diagonal oracle. □

Theorem 593.5 (Regular Cauchy witness schedule non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet factors through the finite rows*

$$X, \mu, W, D, R, T, H, C, P, N.$$

The consumer may read the regular Cauchy family, the explicit modulus, the selected finite window, the dyadic observation ledger, the diagonal readback, the $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transports, continuation routes, package provenance, and local naming row; it cannot read a quotient stream, selected completed real, host real equality, unbounded diagonal stream, or ambient completeness witness.*

Proof. Project the accepted packet and the local naming surface. The displayed rows of S are exactly $X, \mu, W, D, R, T, H, C, P, N$.

Window-modulus alignment controls every scheduled observation route. Diagonal readback then forces the rational observation through the displayed dyadic ledger before any seal-facing row can be consumed.

Since no other coordinate exists, a downstream consumer has no route to unlisted analytic data. The packet therefore keeps the diagonal witness schedule inside the finite BEDC carrier. □

Theorem 593.6 (Regular Cauchy witness schedule local certificate package)**.** *The carrier, local* $\mathsf{NameCert}$ *obligations, window-modulus alignment, diagonal readback, and non-escape theorem jointly form the local certificate package for the modulus-indexed finite observation route from* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *data through* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback to the* $\mathsf{Real}^{\uparrow}$ *seal.*

The package exports no quotient stream, selected limit, host real equality, unbounded diagonal stream, or ambient completeness theorem.

Proof. Start from the finite carrier of Definition 593.1. It exposes exactly the regular Cauchy family, modulus row, scheduled window, dyadic ledger, diagonal readback, seal handoff, transports, routes, provenance, and local name.

Add the local obligation theorem. It identifies those rows as the complete $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$ surface for the witness schedule.

Window-modulus alignment and diagonal readback bind the operational route: the selected source index comes from μ, the finite window is W, the observation ledger is D, and the readback is R.

Finally apply non-escape. The package therefore presents the Real-completion diagonal observation as a finite naming certificate and not as a quotient stream or host-level real equality assertion. □

Theorem 593.7 (Regular Cauchy witness schedule finite-route correspondence)**.** *For an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N),$$

the finite observation route from the $\mathsf{RegSeqRat}^{\uparrow}$ *family row through the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal is exactly the conjunction of the following displayed rows: modulus selection by* μ*, window scheduling by* W*, dyadic ledger readback by* D*, diagonal row* R*, seal handoff* T*, componentwise* hsame *transport by* H*, displayed* Cont *replay by* C*, package provenance by* P*, and local* NameCert *naming by* N*.*

Equivalently, a consumer has access to that four-object route precisely when it reads the finite rows $X, \mu, W, D, R, T, H, C, P, N$ *under the local certificate package of Definition 593.6. The correspondence does not assert* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, host real equality, or equality of completed limits.*

Proof. Project the accepted packet S. The carrier of Definition 593.1 lists exactly the RegSeqRat family row, modulus row, selected StreamName window, DyadicRatCore ledger, diagonal readback, RealUp seal handoff, transports, routes, provenance, and local name.

Apply Definition 593.2 to identify those coordinates as the complete local naming surface. Then use Definition 593.3 to bind the requested tolerance to μ and the scheduled window W.

Apply Definition 593.4 to route the rational observation through D and R, and use Definition 593.6 to repack the handoff to T with H, C, P, and N. Finally, Definition 593.5 keeps the packet finite, so no unlisted analytic object can be read from the route.

Because this chapter is marked `\origin{ai}`, any later promotion beyond seed closure must cite the corresponding TasteGate target before the closure grade is advanced. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp taste gate boundary`

Theorem 593.8 (Regular Cauchy witness schedule public boundary)**.** *For an accepted* RegularCauchyWitnessSched *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N),$$

the public boundary exported to completion-facing consumers is exactly the conjunction of the local NameCert *obligations, window-modulus alignment, diagonal readback, non-escape, local certificate package, and finite-route correspondence. Thus any* $\mathsf{Real}^{\uparrow}$ *diagonal-seal consumer that reads this packet must factor through the finite* BHist *rows* $X, \mu, W, D, R, T, H, C, P, N$ *and through no quotient stream, host real equality, selected completed limit, unbounded diagonal stream, or ambient completeness witness.*

Proof. Apply Definition 593.2 to identify the complete local naming surface, and apply Definition 593.3 to bind every scheduled window read to the modulus row μ and the finite window W.

Next use Definition 593.4 to route the rational observation through D and R. The non-escape theorem then rules out any boundary data outside the finite rows of S.

The local certificate package of Definition 593.6 repacks those obligations as the certificate visible to downstream consumers, while Definition 593.7 identifies the same data with the route from $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ to the $\mathsf{Real}^{\uparrow}$ seal.

Because the chapter carries the displayed TasteGate marker

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali

the public boundary is exported only as this finite packet boundary. It does not supply completeness, quotient equality, or host real equality. □

Theorem 593.9 (Regular Cauchy witness schedule Real handoff dependency)**.** *Every accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *Real handoff factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family row, the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row. In coordinates, the handoff route for*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

uses X as the regular-rational family row, μ and W as the modulus-selected window, D and R as the dyadic diagonal readback ledger, and T as the Real seal handoff. The only transport surface available to this route is the finite BHist *packet together with the displayed* hsame, Cont, Pkg, *and* NameCert *rows H, C, P, N.*

Consequently no quotient stream, selected completed limit, host real equality, or ambient completeness witness can serve as an alternate source for the accepted Real handoff.

Proof. Project S through the carrier of Definition 593.1. The carrier exposes the family row X, modulus row μ, selected finite window W, dyadic ledger D, diagonal readback R, seal row T, and the rows H, C, P, N in one finite packet.

Apply Definition 593.2 and Definition 593.3. These bind local naming to the displayed rows and force the selected window to be read through μ and W before the dyadic ledger is inspected.

Next use Definition 593.4 and Definition 593.7. The diagonal observation reaches D and R, then the finite route carries the observation from $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ to the $\mathsf{Real}^{\uparrow}$ seal row.

Finally apply Definition 593.8. Since the public boundary is exactly the finite rows of S transported by H, replayed by C, packaged by P, and named by N, the handoff cannot escape to a quotient stream, selected limit, host real equality, or ambient completeness row. □

Theorem 593.10 (Regular Cauchy witness schedule seal-source determinacy)**.** *For an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *seal consumer route for the scheduled observation is determined by the displayed modulus-window-dyadic route. If two accepted consumer presentations expose the same* $\mathsf{RegSeqRat}^{\uparrow}$ *family row X, modulus row μ, selected finite window W, dyadic ledger D, diagonal readback R, seal row T, componentwise transports H, continuation routes C, package provenance P, and local name N, then they expose the same consumer-visible seal row T.*

No quotient stream, selected completed limit, host real equality, or ambient completeness row can supply an alternate seal source for the scheduled read.

Proof. Start from the carrier of Definition 593.1. It places the family row, modulus selector, finite window, dyadic ledger, diagonal readback, $\mathsf{Real}^{\uparrow}$ seal, transports, routes, provenance, and local name in one finite BHist packet.

Apply the NameCert obligations and window-modulus alignment. These force any seal-facing read to use the same modulus row μ and the same selected finite window W before it reaches the dyadic ledger.

Next use diagonal readback and finite-route correspondence. The rational observation is routed through D and R, and the seal handoff is then the displayed row T transported and replayed through H, C, P, N.

The public boundary theorem rules out any consumer-visible source outside the finite packet. Because the chapter carries the displayed TasteGate marker

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali

the `\origin{ai}` packet is consumed only through this carrier-aligned route. Thus two accepted presentations with the same displayed rows have the same seal source T, and no quotient stream, selected limit, host equality, or completeness row can provide another one. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleCarrier seal source determinacy`

Theorem 593.11 (Regular Cauchy witness schedule window coverage)**.** *For an accepted* RegularCauchyWitnessSche *packet* $S = (X, \mu, W, D, R, T, H, C, P, N)$*, every accepted schedule window is exhausted by the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row* X*, the explicit modulus row* μ*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*. No accepted window can read an additional stream coordinate, host real coordinate, or completion coordinate outside these rows.*

Proof. Project the carrier. The source family row is X, the selector for the requested tolerance is μ, and the only finite schedule attached to that selector is the row W.

The dyadic ledger D records the endpoint, radius, tolerance, and comparison data consumed by the scheduled read. Together X, μ, W, D therefore provide the complete finite window surface.

The remaining rows R, T, H, C, P, N read back, hand off, transport, route, locate, and name that same surface. Since none of them adds another stream, real, or completion coordinate, the accepted window is exhausted by the four displayed rows. □

Theorem 593.12 (Regular Cauchy witness schedule tail route)**.** *Every tail-facing consumer route for an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet factors through* H, C, P, N *after the finite rows* X, μ, W, D, R, T *have been fixed. Such a consumer can transport the scheduled observation, replay its displayed continuation route, cite its package provenance, and name its local certificate, but it cannot read an unbounded diagonal stream.*

Proof. Use window coverage to bind the finite source of the route to X, μ, W, D. Diagonal readback then fixes the rational observation row R, and seal-source determinacy fixes the consumer-visible $\mathsf{Real}^{\uparrow}$ handoff row T.

The only tail-facing rows left in the carrier are H, C, P, N. These rows preserve and replay the same finite packet; they do not introduce another observation source.

Therefore every tail-facing consumer route is administrative over the displayed finite rows. An unbounded diagonal stream is not a coordinate of the carrier and cannot be recovered from the tail route. □

Theorem 593.13 (Regular Cauchy witness schedule tail-budget factorization)**.** *Let* $S = (X, \mu, W, D, R, T, H, C, P,$ *be an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Every tail-budget consumer or* $\mathsf{Real}^{\uparrow}$ *seal consumer first reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row* X*, the selected modulus row* μ*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*. Only after these rows have been fixed can the consumer read the rational observation row* R *and the* $\mathsf{Real}^{\uparrow}$ *seal row* T*, transported through* H*, replayed through* C*, packaged by* P*, and named by* N*. The factorization exports no quotient stream, selected limit, host real equality, or ambient completeness witness.*

Proof. Start from the carrier definition for the accepted packet. Its consumer-visible coordinates are exactly $X, \mu, W, D, R, T, H, C, P, N$, so a tail-budget read has no source outside the displayed tuple.

Apply Definition 593.11. It fixes the initial finite window surface as X, μ, W, D before any seal-facing read is available.

Next use Definition 593.12. Tail-facing consumption can only proceed through the rational observation row R, the seal row T, and the administrative route H, C, P, N after the finite window has been selected.

The $\mathsf{Real}^{\uparrow}$ dependency is the handoff of Definition 593.14; it adds a seal row, not an ambient completeness object. Since the chapter carries the displayed TasteGate marker

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali

the `\origin{ai}` consumer boundary is aligned with the same carrier route. Thus every tail-budget or seal consumer factors through the explicit modulus, finite window, dyadic ledger, and seal rows, and no quotient stream, selected limit, host real equality, or completeness witness is exported. □

Theorem 593.14 (Regular Cauchy witness schedule Real handoff)**.** *For an accepted* RegularCauchyWitnessSchedule *packet, the* $\mathsf{Real}^{\uparrow}$ *seal row* T *is a finite handoff from* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *data: it is reached from* X, μ, W, D, R *through the displayed route and then transported, located, and named by* H, C, P, N. *The handoff is not a completeness theorem, quotient-equality theorem, host real equality theorem, or selected-limit theorem.*

Proof. Begin with the displayed route. The regular-rational source is X, the selected index is supplied by μ, the finite window is W, and the dyadic observation ledger is D.

Apply diagonal readback to obtain the observation row R. Then apply seal-source determinacy to identify the $\mathsf{Real}^{\uparrow}$ seal handoff as the row T attached to that same route.

Finally inspect the frontier rows H, C, P, N. They transport, replay, locate, and name the handoff without adding a quotient stream, selected limit, host equality, or completeness row. Thus T is a finite handoff, not an analytic closure claim. □

Theorem 593.15 (Regular Cauchy witness schedule shared-window lock)**.** *Let two accepted* RegularCauchyWitnessS *packets expose the same displayed rows*

$$X, \mu, W, D, R, T, H, C, P, N.$$

Then they expose the same finite schedule route from the $\mathsf{RegSeqRat}^{\uparrow}$ *family row through the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the diagonal readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row. No alternate stream source, selected limit source, or completeness source can supply a second route for the same displayed packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleCarrier shared window lock`

Proof. Project both accepted packets to the displayed carrier. The coordinates agree on the family row X, modulus selector μ, finite window W, dyadic ledger D, readback R, seal row T, transports H, routes C, provenance P, and local name N.

Apply window-modulus alignment to both packets. The requested tolerance reaches the same selector μ, so the only admissible scheduled window in either packet is the displayed row W.

Next apply diagonal readback and Real handoff. Both routes pass through D and R before reaching the same seal row T, and the rows H, C, P, N replay and name that route without adding a second source.

Because the chapter carries the displayed TasteGate target

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali

the `\origin{ai}` packet is consumed only through the carrier-aligned finite route. Non-escape and the public boundary rule out an alternate unbounded stream, selected limit, or completeness witness. □

Theorem 593.16 (Regular Cauchy witness schedule tail-threshold readback)**.** *For an accepted* RegularCauchyWitnessSchedule$^{\uparrow}$ *packet, every tail-facing threshold request must pass through the modulus row μ, the finite window row W, the dyadic ledger row D, and the diagonal readback row R before it reaches the* Real$^{\uparrow}$ *seal row T. The request cannot bypass those rows through a quotient stream, a host real equality row, or an ambient completeness row.*

Proof. Start with the accepted carrier. The only threshold selector displayed in the packet is μ, and the only scheduled stream window attached to that selector is W.

Use window coverage to bind any tail-facing threshold read to X, μ, W, D. The dyadic ledger row D records the tolerance-facing observation data consumed by the selected window.

Apply diagonal readback to force the request through R. Then apply Real handoff to identify T as the finite Real$^{\uparrow}$ seal row reached after that readback.

The displayed TasteGate boundary

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp_taste_gate_boundary

keeps this `\origin{ai}` route inside the finite packet. Non-escape excludes quotient streams, host equality, and ambient completeness as bypasses around μ, W, D, R, T. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleCarrier tail threshold readback`

Theorem 593.17 (Regular Cauchy witness schedule Real seal readiness)**.** *The carrier of Definition 593.1, the local* NameCert *obligations, window-modulus alignment, diagonal readback, non-escape, public boundary, shared-window lock, and tail-threshold readback form the finite obligation package for the* RegularCauchyWitnessSchedule$^{\uparrow}$ *route from* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*, and* DyadicRatCore$^{\uparrow}$ *data to the* Real$^{\uparrow}$ *seal row.*

The package is limited to the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not assert* Real$^{\uparrow}$ *completeness, equality of completed limits, quotient equality of streams, host real equality, or bridge checking.*

Proof. Begin with the carrier and NameCert obligation theorem. They expose the rows $X, \mu, W, D, R, T, H, C, P, N$ and identify carrier admission, family admission, modulus exposure, window scheduling, dyadic readback, diagonal readback, seal handoff, transports, routes, provenance, and local naming as the certificate surface.

Window-modulus alignment, diagonal readback, and non-escape bind the operational route. Every scheduled observation uses μ, W, D, and R, and no unlisted analytic object can be read from the packet.

The public boundary then packages the same rows for completion-facing consumers. Shared-window lock makes two accepted presentations with the same displayed rows expose the same finite

route, while tail-threshold readback forces every tail-facing threshold request through μ, W, D, R before T.

Finally cite the two TasteGate anchors

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali
BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp_taste_gate_bounda

They record that the `\origin{ai}` route is consumed through the finite carrier boundary. The resulting package is therefore an obligation package over the displayed RegSeqRat, StreamName, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows, not a completeness or quotient-equality theorem. □

Theorem 593.18 (Regular Cauchy witness schedule Real-completion handoff lock)**.** *For an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N),$$

any completion-facing consumer that reads S must first pass through the displayed analytic rows X, μ, W, D, R, T and then through the certificate rows H, C, P, N. The passage uses window-modulus alignment, diagonal readback, finite-route correspondence, the public boundary, seal-source determinacy, and the Real handoff theorem to identify T as the finite $\mathsf{Real}^{\uparrow}$ *seal handoff for the same route.*

Consequently the handoff lock excludes quotient stream equality, selected completed limits, host real equality, ambient completeness, and choice. None of those objects is a coordinate of the accepted packet, and none is created by the transport, continuation, package, or naming rows.

Proof. Project the accepted carrier. The visible analytic route is exactly X, μ, W, D, R, T, followed by the structural certificate rows H, C, P, N.

Apply window-modulus alignment to bind any tolerance-facing read to the modulus row μ and the finite window W. Apply diagonal readback to force the scheduled observation through the dyadic ledger D and the readback row R.

Use finite-route correspondence and the public boundary to keep the completion-facing consumer inside the same displayed packet. Seal-source determinacy and the Real handoff theorem then identify the consumer-visible seal as the row T reached by that route.

It remains to check non-escape. The rows H, C, P, N only transport, replay, package, and name the same finite data. Since the carrier has no coordinate for quotient stream equality, selected completed limits, host real equality, ambient completeness, or choice, the completion-facing handoff cannot export any of them. □

Theorem 593.19 (Regular Cauchy witness schedule scope-closed route)**.** *For an accepted* RegularCauchyWitnessSch *packet $S = (X, \mu, W, D, R, T, H, C, P, N)$, the scoped route from* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *data through* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback to the* $\mathsf{Real}^{\uparrow}$ *seal is exactly the finite route witnessed by the carrier, the local* NameCert *obligations, window-modulus alignment, diagonal readback, public boundary, shared-window lock, tail-threshold readback, and Real seal readiness.*

Equivalently, the route is the displayed finite passage

$$(X, \mu, W, D, R, T, H, C, P, N)$$

through BHist, hsame, Cont, Pkg, *and* NameCert *rows. It denies* $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, host real equality, Lean verification, and bridge checking.*

Proof. Project the accepted packet S to the carrier of Definition 593.1. Its coordinates are exactly the regular Cauchy family row X, modulus row μ, finite window W, dyadic ledger D, diagonal readback R, seal row T, transports H, continuation routes C, provenance P, and local name N.

Apply Definition 593.2 to identify those coordinates as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$ surface. Then use Definition 593.3 and Definition 593.4 to bind the selected tolerance, window, dyadic ledger, and diagonal observation.

The public boundary of Definition 593.8 keeps completion-facing consumption inside the same finite rows. The shared-window lock of Definition 593.15 rules out a second route for another accepted presentation with the same displayed rows, and Definition 593.16 forces every tail-facing threshold request through μ, W, D, R before T.

Finally apply Definition 593.17. The two displayed TasteGate anchors

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp_taste_gate_bounda

bind the `\origin{ai}` packet to this finite carrier boundary. Thus the scoped route is exactly the finite witness route over RegSeqRat, StreamName, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows, and it does not provide RealUp completeness, quotient streams, selected limits, host real equality, Lean verification, or bridge checking. □

Theorem 593.20 (Regular Cauchy witness schedule scoped tail-meet handoff)**.** *For an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N),$$

the same finite window, modulus, dyadic readback, and seal route consumed by the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *surface is supplied by the displayed rows of* S*. The route passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal row* T*, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows* H, C, P, N*.*

This handoff exports only the scoped tail/window surface. It exports no quotient stream, selected limit, host equality, completeness row, or independent tail-meet proof.

Proof. Project S to the carrier and apply the local NameCert obligations. The carrier gives the family, modulus, window, dyadic ledger, diagonal readback, seal, transport, continuation, package, and naming rows as one finite packet.

Use window-modulus alignment and diagonal readback to bind the tail-facing request to μ, W, D, and R. Non-escape and the public boundary keep the route inside the displayed BHist rows before the seal row T is consumed.

Apply the scope-closed route theorem. It identifies the finite passage through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows as the displayed scoped route for this chapter.

The sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ packet consumes exactly that finite tail/window surface as a downstream consumer surface. Since S has no coordinate for quotient streams, selected limits, host equality, or completeness, the handoff is only the scoped finite route and not an additional analytic closure claim. □

Theorem 593.21 (Regular Cauchy witness schedule tail-meet seal comparison)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Let an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet have its paired source, finite-window, modulus, threshold, comparison, transport, route, provenance, and local naming rows transported to the same displayed* W, D, R, T, H, C, P, N *surface. Then the consumer-visible* $\mathsf{Real}^{\uparrow}$ *seal route of* S *is the same finite route consumed by the sibling tail-meet surface.*

Equivalently, both presentations reach the seal only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No quotient stream, selected limit, host equality row, or completeness row can distinguish the schedule presentation from the transported tail-meet consumer presentation.*

Proof. Project S to the carrier of Definition 593.1. The carrier and Definition 593.2 expose exactly the rows $X, \mu, W, D, R, T, H, C, P, N$ and no analytic escape coordinate.

Use window-modulus alignment and diagonal readback to bind the route through μ, W, D, R. The Real seal readiness theorem then identifies T as the displayed $\mathsf{Real}^{\uparrow}$ seal reached from the finite $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data.

Now apply the scope-closed route theorem and the scoped tail-meet handoff. They identify the same route as the finite consumer surface shared with the sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ chapter, whose public surface is recorded in chapter 560.

The two TasteGate anchors

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_ali
BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp_taste_gate_bounda

keep the `\origin{ai}` packet inside this finite carrier boundary. Since both presentations expose the same W, D, R, T, H, C, P, N rows at the consumer surface, and since neither packet supplies quotient streams, selected limits, host equality, or completeness as coordinates, no downstream $\mathsf{Real}^{\uparrow}$ consumer can distinguish the two seal routes. □

Theorem 593.22 (Regular Cauchy witness schedule public seal export)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal export of an accepted* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

factors through the displayed finite rows $X, \mu, W, D, R, T, H, C, P, N$*. The export reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal handoff* T*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$ *row* N*.*

The same route is pinned by the two TasteGate anchors

BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleTasteGate_single_carrier_alig
BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp_taste_gate_boundar

and by the downstream tail-meet seal comparison. It exports no quotient stream, selected completed limit, host real equality, ambient completeness principle, or choice operation.

Proof. Project the accepted packet to the carrier of Definition 593.1. The carrier exposes only $X, \mu, W, D, R, T, H, C, P, N$, and Definition 593.2 identifies these rows as the local NameCert surface.

Use Definition 593.3 and Definition 593.4. They force any tolerance request and diagonal observation to pass through the finite window, modulus, dyadic ledger, and readback rows before the seal row is available.

Apply Definition 593.7, Definition 593.8, and Definition 593.17. These clauses identify the public-facing route as the same finite route from $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ to the $\mathsf{Real}^{\uparrow}$ seal handoff.

Finally use Definition 593.21 together with the two displayed TasteGate anchors. The single-carrier alignment and boundary theorem keep the `\origin{ai}` packet inside the shown carrier rows, while the tail-meet comparison shows that downstream seal consumers see only the same finite surface. Since none of the coordinates is a quotient stream, completed limit selector, host real equality, ambient completeness principle, or choice operation, the public seal export has no escape coordinate. □

Theorem 593.23 (Regular Cauchy witness schedule tail-agreement consumer)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Any paired-tail consumer that compares the scheduled observation of S with a second regular Cauchy source and then enters the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699 must first factor through the displayed modulus row μ, the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback D, the diagonal row R, and the* $\mathsf{Real}^{\uparrow}$ *seal row T of S.*

After those rows have been consumed, the paired route can add only the second source row and the displayed agreement row required by the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *carrier rows of Definition 699.1. Thus the route reaches the downstream threshold factorization of Definition 699.8 through the public export of Definition 593.22 and the comparison surface of Definition 593.21. It does not introduce quotient streams, host real equality, selected completed limits, ambient completeness, or choice-selected tails.*

Proof. Project S to the carrier of Definition 593.1. The carrier exposes exactly the rows $X, \mu, W, D, R, T, H, C, P, N$, so the scheduled observation has no coordinate outside the displayed modulus, window, dyadic readback, diagonal, seal, transport, continuation, provenance, and local naming rows.

Apply Definition 593.22. It places every public $\mathsf{Real}^{\uparrow}$ seal read of S on the finite route through μ, W, D, R, T together with H, C, P, N. Then apply Definition 593.21; the comparison route seen by a downstream tail consumer is still the same displayed finite surface.

Now pass to the paired seal chapter chapter 699. The carrier in Definition 699.1 contributes the missing paired source row and the displayed agreement row, while Definition 699.8 forces the Real or Completion consumer to read those rows through the shared threshold route before any handoff is available.

No step supplies a quotient stream, a host real equality, a selected completed limit, an ambient completeness row, or a choice-selected tail. Therefore the paired tail-agreement route consumes the single scheduled observation only by extending the already displayed witness schedule with the second source and agreement rows of $\mathsf{RealTailAgreementSeal}^{\uparrow}$. □

Theorem 593.24 (Regular Cauchy witness schedule shared-window factorization)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Every completion-facing consumer route, and every paired-tail consumer route, that reaches a* $\mathsf{Real}^{\uparrow}$ *seal through the witness schedule factors through one shared finite window surface in the following order: first the* $\mathsf{RegSeqRat}^{\uparrow}$ *source and explicit modulus-window rows X, μ, W, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the diagonal readback row R, the seal row T, and finally the componentwise* BHist, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows H, C, P, N.*

This factorization is the common surface pinned by Definition 593.3, Definition 593.4, Definition 593.22, Definition 593.21, and Definition 593.23. Conversely, any consumer route displaying exactly those rows is the same finite route already exported by the public seal and paired-tail consumer theorems. No quotient stream, selected completed limit, host real equality, ambient completeness row, or choice operation is supplied by the shared-window surface.

Proof. Project S to the carrier of Definition 593.1. The projection exposes the finite rows $X, \mu, W, D, R, T, H, C, P, N$ as the only schedule data available to a completion-facing or paired-tail consumer.

Use Definition 593.3 to fix the modulus and finite window rows μ, W over the source row X. Then use Definition 593.4 to fix the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D and diagonal readback row R as the only observation path feeding the seal.

Apply Definition 593.22. It identifies the consumer-visible $\mathsf{Real}^{\uparrow}$ seal handoff with the displayed row T after the finite X, μ, W, D, R path has been read. The remaining rows H, C, P, N supply only the componentwise transport, continuation, package provenance, and local naming surface attached to that same finite route.

For the downstream direction, apply Definition 593.21 and Definition 593.23. These theorems show that the sibling tail-meet and paired tail-agreement consumers can extend the displayed surface with their own paired-source data, but cannot distinguish another route for the scheduled observation itself.

Conversely, a consumer route that displays exactly $X, \mu, W, D, R, T, H, C, P, N$ has already chosen the rows forced by window-modulus alignment, diagonal readback, public seal export, tail-meet comparison, and tail-agreement consumption. Since none of those rows is a quotient stream, selected completed limit, host equality, completeness principle, or choice operator, there is no escape coordinate by which the route could differ from the finite public route already exported in this chapter. □

Theorem 593.25 (Regular Cauchy witness schedule budget pullback exactness)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Let*

$$B = (E, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet whose precision request row E, finite window rows W_B, dyadic ledger rows D_B, regular-source handoff rows R_B, and real-seal rows S_B agree componentwise under* hsame *with the witness-schedule rows μ, W, D, R, T after the transports displayed in H, H_B and the routes displayed in C, C_B are replayed.*

Then every $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer route from B that reaches the synchronized* $\mathsf{Real}^{\uparrow}$ *observation pulls back uniquely to the finite witness-schedule route through*

$$X, \mu, W, D, R, T, H, C, P, N.$$

The pullback identifies the two L10 consumer routes with the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. It creates no quotient stream equality, host real equality, selected completed limit, ambient completeness row, or unbounded diagonal schedule.*

Proof. Project the accepted schedule packet S to its carrier rows. The window-modulus alignment theorem fixes the requested precision at μ and the finite $\mathsf{StreamName}^{\uparrow}$ window at W, while diagonal readback fixes the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D and readback row R as the only observation path feeding the seal.

Use the Real handoff, shared-window lock, and shared-window factorization theorems in this chapter. They identify the completion-facing schedule route with the finite route through $X, \mu, W, D, R, T, H, C, P, N$ and leave only transport, continuation, provenance, and local naming rows after the seal handoff T has been reached.

Now project the observation-budget packet B. By request totality in Definition 611.12, the consumer request reads a finite path from E to W_B and D_B. By route determinacy in Definition 611.13, two budget routes with the same displayed rows determine the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ observation surface up to the componentwise hsame rows. By non-escape in Definition 611.6, the budget route has no hidden channel outside $E, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B$.

The assumed componentwise hsame agreement transports the budget rows E, W_B, D_B, R_B, S_B onto μ, W, D, R, T. Therefore the budget consumer and the witness-schedule consumer read the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, and the remaining displayed transport, route, provenance, and naming rows supply the unique pullback into $X, \mu, W, D, R, T, H, C, P, N$.

Finally apply the non-escape clauses on both sides. Since neither packet contains quotient stream equality, host real equality, a selected completed limit, ambient completeness, or an unbounded diagonal schedule as a coordinate, the pullback is exact only at the finite BEDC route and cannot manufacture any stronger analytic object. □

Theorem 593.26 (Regular Cauchy witness schedule budget selector compatibility)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. Every finite precision selector compatible with the* $\mathsf{RealObservationBudget}^{\uparrow}$ *route of chapter 611 identifies its request row with* μ*, its finite window row with* W*, its dyadic ledger row with* D*, and its seal row with* T*, after replaying the componentwise* hsame *transports in* H *and the displayed* Cont *routes in* C*.*

Thus a selector that reaches the synchronized real observation consumes the same displayed L10 packet

$$X, \mu, W, D, R, T, H, C, P, N$$

as the witness schedule. It has no second precision request, no alternate finite-window source, no hidden dyadic ledger, and no seal row outside the packet displayed by S*.*

Proof. First apply window-modulus alignment to the accepted schedule packet. This pins any finite precision request compatible with the schedule to the modulus row μ and pins the selected finite $\mathsf{StreamName}^{\uparrow}$ window to W.

Next apply diagonal readback. The observation read from the selected window must use the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D and the diagonal row R before it can reach the $\mathsf{Real}^{\uparrow}$ seal row T.

Now use budget pullback exactness for the compatible $\mathsf{RealObservationBudget}^{\uparrow}$ consumer. The budget route agrees componentwise with μ, W, D, R, T after the transports and continuations in H and C are replayed, so its precision selector is exhausted by those same rows.

The remaining rows X, H, C, P, N supply the source, transport, continuation, provenance, and local naming data attached to the already identified route. Since the budget non-escape clause supplies no additional request, window, dyadic, or seal coordinate, the selector and the witness schedule consume the same finite packet. □

Theorem 593.27 (Regular Cauchy witness schedule budget tail-agreement pullback)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted RegularCauchyWitnessSchedule$^{\uparrow}$ *packet. Let*

$$B = (E, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be a compatible RealObservationBudget$^{\uparrow}$ *route whose precision request, finite window, dyadic ledger, regular-source handoff, and real-seal rows agree componentwise with* μ, W, D, R, T *after the displayed transports and continuations are replayed. Suppose that this synchronized budget route is then consumed by a paired-tail route through* RealTailAgreementSeal$^{\uparrow}$.

Then the consumer-visible route for the scheduled side pulls back exactly to the finite row surface

$$X, \mu, W, D, R, T, H, C, P, N$$

and the paired-tail route may add only the second-source row and the agreement row required by the RealTailAgreementSeal$^{\uparrow}$ *carrier. It exports no quotient stream, selected completed limit, host real equality, ambient completeness principle, or choice-selected tail.*

Proof. First apply budget pullback exactness, Definition 593.25. The observation-budget request, window, ledger, handoff, and seal rows are transported onto μ, W, D, R, T, so the budget consumer has the same scheduled-side route through $X, \mu, W, D, R, T, H, C, P, N$.

Next apply budget selector compatibility, Definition 593.26. The selector has no second request, alternate window, hidden dyadic ledger, or seal row outside the displayed packet, so the completion-facing budget surface is already spent before the paired-tail route is entered.

Now use the tail-agreement consumer theorem, Definition 593.23. After the scheduled-side rows have been read, the RealTailAgreementSeal$^{\uparrow}$ route can contribute only the second source and the displayed agreement row from Definition 699.1.

Finally use non-escape on both packets. The witness schedule has no coordinate for quotient streams, selected completed limits, host equality, completeness, or choice-selected tails, and the observation budget has no hidden route outside its displayed request, window, ledger, handoff, seal, transport, continuation, provenance, and naming rows. The combined consumer route is therefore exactly the finite scheduled surface together with the paired source and agreement rows. □

Theorem 593.28 (Regular Cauchy witness schedule tail budget exhaustion)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted RegularCauchyWitnessSchedule$^{\uparrow}$ *packet. Every paired-tail or completion-facing budget consumer that reads the scheduled observation of* S *before entering the* RealTailAgreementSeal$^{\uparrow}$ *route of chapter 699 remains inside the displayed rows*

$$X, \mu, W, D, R, T, H, C, P, N.$$

The consumer may subsequently add the second source and agreement rows required by the tail-agreement seal, but it cannot add another schedule, tolerance ledger, diagonal readback, real-seal row, or completion principle for the scheduled side.

Proof. Begin with the tail-agreement consumer theorem for S. It forces any paired-tail route that uses the scheduled observation to pass through μ, W, D, R, T together with the transport, continuation, provenance, and local naming rows H, C, P, N.

Apply shared-window factorization. Completion-facing and paired-tail consumers share the same finite window surface over $X, \mu, W, D, R, T, H, C, P, N$, so the scheduled side has already spent its request, window, dyadic, diagonal, and seal budget before the downstream route is entered.

Finally use non-escape for the witness schedule and for the budget consumer surface. The only data available before the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ carrier is the displayed schedule packet; the seal route can append its paired source and agreement rows only after that finite packet has been consumed.

Therefore every paired-tail or completion-facing budget consumer is exhausted by $X, \mu, W, D, R, T, H, C, P, N$ on the scheduled side, and the later tail-agreement route cannot create a second scheduled observation route. □

Theorem 593.29 (Regular Cauchy witness schedule public export package)**.** *The* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *public export package is assembled by the NameCert obligations, window-modulus alignment, diagonal readback, public boundary, Real seal readiness, shared-window factorization, budget pullback, budget selector compatibility, tail budget exhaustion, and the displayed TasteGate carrier anchor for the packet*

$$X, \mu, W, D, R, T, H, C, P, N.$$

Every $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of this package reads exactly these finite rows. It reads no quotient stream, selected limit, host real equality, ambient completeness, or choice operation.*

Proof. Start from the displayed TasteGate single-carrier marker attached to Definition 593.1. The carrier and the NameCert obligations fix the packet as $X, \mu, W, D, R, T, H, C, P, N$.

The route to the public observation is then forced by window-modulus alignment, diagonal readback, public boundary, and Real seal readiness. These theorems identify the selected modulus, finite window, dyadic ledger, diagonal row, and Real seal row without adding a quotient stream or a selected completed limit.

For completion-facing consumers, use shared-window factorization, budget pullback exactness, budget selector compatibility, and tail budget exhaustion. These rows show that the shared window, observation budget, precision selector, and paired-tail route all pull back to the same displayed schedule packet.

Thus the public export package is a single finite BEDC surface. The consumer can inspect $X, \mu, W, D, R, T, H, C, P, N$ and nothing stronger: no host real equality, no ambient completeness principle, and no choice operation is present as a carrier coordinate or downstream route. □

Theorem 593.30 (Regular Cauchy witness schedule standard bridge)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet. The public BEDC export package of Definition 593.29 is equivalent to the standard finite witness-schedule record whose fields are exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row* X*, modulus selector* μ*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, diagonal readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* T*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$ *row* N*.*

The forward read projects those rows from the public export package. The reverse read rebuilds the finite carrier and the public package from exactly those standard finite rows. Both directions preserve the exclusions of $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, host real equality, and choice.*

Proof. For the forward direction, apply Definition 593.29. Its public export is already exhausted by the displayed packet $X, \mu, W, D, R, T, H, C, P, N$.

Use Definition 593.3 to identify μ with the selected tolerance route and W with the corresponding finite window. Then apply Definition 593.4 to route the scheduled observation through D and R

before the $\mathsf{Real}^{\uparrow}$ seal row T is read. The displayed TasteGate marker BEDC.Derived.RegularCauchyWitnessSchedul fixes this as a single carrier surface.

For the reverse direction, start with the standard finite witness-schedule record carrying exactly the same ten rows. The rows $X, \mu, W, D, R, T, H, C, P, N$ rebuild the carrier of Definition 593.1; the local naming row, transports, routes, and provenance then repack it as the public export package of Definition 593.29.

Since neither construction introduces a field outside those rows, the bridge has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, host real equality, or choice. The BEDC export and the standard finite witness-schedule record therefore read back to one another over exactly the displayed finite carrier. □

Theorem 593.31 (Regular Cauchy witness schedule finite-observation terminal coverage)**.** *Let*

$$S = (X, \mu, W, D, R, T, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *packet, where X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *family row, μ is the modulus-selector row, W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, R is the diagonal readback row, and T is the* $\mathsf{Real}^{\uparrow}$ *seal. Every terminal finite observation consumed from S by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing route factors through exactly the rows*

$$X, \mu, W, D, R, T, H, C, P, N.$$

The terminal consumer receives no $\mathsf{Real}^{\uparrow}$ *completeness principle, quotient stream, selected completed limit, host real equality, or choice operation, and it cannot add a second scheduled observation route downstream.*

Proof. Begin with the public export package of Definition 593.29. It exposes the terminal observation as the finite BEDC packet $X, \mu, W, D, R, T, H, C, P, N$, with the family, modulus, window, dyadic ledger, diagonal readback, and Real seal already displayed before a completion-facing consumer can read the observation.

Next use the standard bridge of Definition 593.30. The bridge identifies the same finite witness-schedule record, so a terminal consumer cannot reinterpret W, D, R, T as a quotient stream, selected limit, host real equality, or choice-selected completion object.

Finally apply tail budget exhaustion from Definition 593.28. Once the downstream route has spent the scheduled-side budget on $X, \mu, W, D, R, T, H, C, P, N$, any later completion or paired-tail consumer may only use the route already carried by those rows and cannot introduce a second scheduled observation path. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ family row, explicit modulus row, selected $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, diagonal readback row, $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed:

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessScheduleUp.taste gate`

The public export is the finite witness schedule boundary over the displayed RegSeqRat, StreamName, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows, as bound by Definition 593.1, Definition 593.8, Definition 593.22, Definition 593.29, and Definition 593.30.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyWitnessScheduleUp.RegularCauchyWitnessScheduleUp\_taste\_gate\_boundary`

Bridge status: paperBridge

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, Lean verification, or bridge checking.

Upgrade path: No further paper-axis upgrade is pending for the finite witness-schedule surface; verification and bridge checking remain on the independent formal axis.

Chapter 594

A Concrete Naming Certificate for $\mathsf{MaxCausalRate}^{\uparrow}$

Falsifiable prediction: An accepted $\mathsf{MaxCausalRate}^{\uparrow}$ packet is refuted if a consumer reads a public maximum-rate bound without the displayed $\mathrm{MultiHistConfig}^{\uparrow}$ row, $\mathsf{CrossHistCausal}^{\uparrow}$ witness family, shared Unary bound row, and finite comparison ledger.

Independence witness: multihistconfig, crosshistcausal, interhistinvariant, lorentzframerate: no sibling is bijective to $\mathsf{MaxCausalRate}^{\uparrow}$ because this carrier alone combines selected pair witnesses, one shared unary upper-bound row, comparison totality, and rate-facing consumer bridge rows.

The $\mathsf{MaxCausalRate}^{\uparrow}$ interface records the finite unary-bound packet that sits after $\mathrm{MultiHistConfig}^{\uparrow}$ and the $\mathsf{CrossHistCausal}^{\uparrow}$ witness-family rows of chapter 478 in the inter-Hist locality chain of Definition 9419.25. Its role is not to introduce a real-valued speed, a metric tensor, or a synchronized frame. It names a displayed family of cross-Hist causal witness rows and one unary bound row against which those witnesses are compared inside the local BEDC ledger.

Definition 594.1 (MaxCausalRate carrier)**.** *A* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, X, U, B, H, Q, C, P, N)$$

with the following visible coordinates:

1. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite observer-history configuration, its local ledgers, no-global-sync witness, componentwise* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local naming row;*
2. *X is a finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness rows over selected pairs of histories from M and selected unary slots;*
3. *U is the unary bound row that names the maximum displayed causal-rate cap for the listed witnesses;*
4. *B records a finite bound comparison for every displayed causal witness in X, relating its slot-local witness row to the same unary bound row U;*
5. *H records componentwise* hsame *transport for M, the witness family X, the unary bound row U, the comparison ledger B, the probe-stability row, the continuation routes, the package row, and the local naming row;*

6. *Q records* psame *stability for the displayed probe reads used by the cross-Hist witnesses and their bound comparisons;*
7. *C records the visible* Cont *routes from M and each selected witness row into U, B, H, Q, P, and N;*
8. *P is the* Pkg *provenance over* $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, Hist, Unary, BHist, hsame, psame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ *row.*

The carrier exports only a finite configuration row, finite causal witness rows, one unary bound row, displayed bound comparisons, transport rows, continuation routes, package provenance, and local naming data. It contains no global observer frame, no synchronized clock, no metric tensor, no Lorentz geometry, and no ambient real-valued rate.

Theorem 594.2 (MaxCausalRate NameCert obligations)**.** *For the carrier of Definition 594.1, the local* $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet $R = (M, X, U, B, H, Q, C, P, N)$;*
2. *configuration locality, requiring every selected causal witness in X to project its history anchors and unary slot from the same* $\mathrm{MultiHistConfig}^{\uparrow}$ *row M;*
3. *unary-bound exactness, requiring every displayed comparison row in B to compare one witness row of X against the shared unary bound row U;*
4. *transport stability through the componentwise* hsame *row H, the displayed* psame *row Q, and the visible* Cont *routes in C;*
5. *non-escape, excluding global frames, synchronized clocks, metric tensors, Lorentz geometry, and ambient real-valued rates.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRateTasteGate single carrier alignment`

Proof. Carrier admission is the projection of the finite tuple in Definition 594.1. The rows $M, X, U, B, H, Q, C, P, N$ display the configuration, causal witnesses, bound row, comparison ledger, transport rows, read routes, package provenance, and local naming row.

Configuration locality follows by reading M before any causal witness is inspected. The selected history pairs and unary slots used by X are drawn from the displayed $\mathrm{MultiHistConfig}^{\uparrow}$ surface, so no witness can appeal to an unlisted observer or a global synchronization row.

Unary-bound exactness is the finite ledger condition on B. Each comparison row has a displayed witness from X on one side and the common unary bound row U on the other, so the maximum-rate surface is a shared bound over listed witnesses rather than a host real-valued function.

Transport stability is componentwise through H, Q, and C. These rows replay the same configuration, witness family, unary bound, and comparison ledger under accepted hsame and psame movements.

The non-escape clause is then an inventory argument. Since the carrier has no coordinate for a global frame, synchronized clock, metric tensor, Lorentz geometry, or real-valued rate, no local $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ projection can export one. □

Theorem 594.3 (MaxCausalRate CrossHistCausal handoff)**.** *Every* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness consumed by an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier is read as one displayed row of the finite family X over the* $\mathrm{MultiHistConfig}^{\uparrow}$ *carrier M. The consumer obtains the selected history pair, unary slot,*

finite causal witness, hsame *and* psame *stability rows, displayed* Cont *routes, and package provenance already present in that witness row, together with the bound comparison against U recorded in B.*

The handoff supplies a finite unary-bound causal ledger only. It does not supply a global frame, synchronized clock, metric tensor, Lorentz geometry, or ambient real-valued speed.

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket crosshist handoff`

Proof. Start with the accepted $\mathsf{MaxCausalRate}^{\uparrow}$ carrier and project M and X. The configuration row M supplies the finite observer-history surface, while X lists the selected $\mathsf{CrossHistCausal}^{\uparrow}$ witness rows that may be consumed.

For a displayed witness row in X, apply the local surface of Definition 478.2. The witness already carries its two history anchors, unary slot ledger, finite dependence row, componentwise hsame transport, two-anchor psame stability, visible Cont routes, package provenance, and local naming row.

Next attach the comparison row from B. This row relates that specific witness to the shared unary bound U, so the handoff adds only the finite bound check needed by $\mathsf{MaxCausalRate}^{\uparrow}$.

Finally route the combined witness-and-bound read through H, Q, C, P, N. The resulting handoff remains inside the displayed BEDC rows and cannot introduce synchronization, metric geometry, or a host-side real-valued rate. □

Theorem 594.4 (MaxCausalRate non-escape)**.** *Every consumer of an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier factors through the finite rows*

$$M, X, U, B, H, Q, C, P, N.$$

In particular, the consumer may read the multi-Hist configuration row, the displayed cross-Hist causal witnesses, the shared unary bound row, the finite comparison ledger, componentwise hsame *transport, displayed* psame *stability, visible* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ *row, and nothing outside those rows.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket non escape`

Proof. Project the accepted carrier and apply Definition 594.2. Its non-escape clause excludes global frames, synchronized clocks, metric tensors, Lorentz geometry, and ambient real-valued rates from the local naming surface.

Apply Definition 594.3 to every displayed witness row that a consumer reads from X. Each read remains a $\mathsf{CrossHistCausal}^{\uparrow}$ row over M plus a finite comparison against the shared unary bound U.

The transport rows H, Q, C then replay only the listed configuration, witness, bound, and comparison data. Since every visible consumer row is one of $M, X, U, B, H, Q, C, P, N$, any request for a global frame, synchronized clock, metric tensor, Lorentz geometry, or real-valued speed leaves the $\mathsf{MaxCausalRate}^{\uparrow}$ carrier rather than projecting from it. □

Theorem 594.5 (MaxCausalRate bound-comparison totality)**.** *For an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier*

$$R = (M, X, U, B, H, Q, C, P, N),$$

the finite comparison ledger B contains a displayed comparison row for every causal witness row in X against the shared unary bound row U. A consumer may therefore read the maximum causal-rate surface only as a total finite ledger over the listed witnesses, not as a partial sample or an ambient real-valued speed function.

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket bound comparison totality`

Proof. Project the accepted carrier and inspect X and B. The witness family X is finite, and the ledger row B is defined as the finite comparison row for every displayed witness in that family against U.

Route each comparison through C. The consumer reaches a witness row of X, the shared unary bound U, and its corresponding comparison row in B by one of the displayed continuations.

The transport rows H and Q preserve the same witness-and-bound pairing. Since P and N only bind provenance and local naming, the totality claim remains a BEDC ledger fact and not a real-valued rate assertion. □

Theorem 594.6 (MaxCausalRate witness-family stability)**.** *For an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier*

$$R = (M, X, U, B, H, Q, C, P, N),$$

componentwise hsame *transport in H, probe stability in Q, and the displayed routes in C preserve the witness family X together with its comparison pairing in B against the shared unary bound U.*

No transported consumer may replace a listed witness by an unlisted cross-Hist row, change the unary bound, or detach the comparison from the $\mathrm{MultiHistConfig}^{\uparrow}$ *row M.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket witness family stability`

Proof. Read M first and then the finite witness family X. Each displayed witness is anchored in the same multi-Hist configuration and selected unary slot surface.

Transport through H preserves M, X, U, B componentwise, while Q preserves the probe reads used by the cross-Hist witnesses and their bound comparisons. Thus the transported family remains paired with the same unary bound row U.

Finally replay the reads through C and bind them by P, N. The carrier has no coordinate for an unlisted cross-Hist witness, a second bound row, or a comparison detached from M, so witness-family stability is exhausted by the displayed finite packet. □

Theorem 594.7 (MaxCausalRate bridge readback)**.** *Every consumer-facing bridge read of an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *packet is exactly the tuple*

$$M, X, U, B, H, Q, C, P, N.$$

Here M is the $\mathrm{MultiHistConfig}^{\uparrow}$ *row, X is the displayed* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness family, U is the shared unary bound, B is the comparison ledger, H is componentwise transport, Q is the probe-stability row, C is the continuation route surface, P is provenance, and N is the local name.*

The readback exports no global frame, synchronized clock, metric tensor, Lorentz geometry, or real-valued speed.

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket bridge readback`

Proof. Project the accepted carrier and local obligations from Definition 594.2. The carrier rows visible to a consumer are exactly $M, X, U, B, H, Q, C, P, N$.

Apply Definition 594.3. It identifies the displayed witness family X as the $\mathsf{CrossHistCausal}^{\uparrow}$ handoff over the multi-Hist configuration row M.

Next use Definition 594.5 and Definition 594.6. These keep the shared unary bound U, the comparison ledger B, and the transported witness family tied to H, Q, C.

The rows P and N then package and name the same surface. Since none of the cited rows contains a global frame, synchronized clock, metric tensor, Lorentz geometry, or real-valued speed coordinate, the bridge readback is exactly the displayed tuple. □

Theorem 594.8 (MaxCausalRate unary-bound non-escape)**.** *For an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier*

$$R = (M, X, U, B, H, Q, C, P, N),$$

the unary-bound row U is the only bound exported by the packet, and it is visible only through the finite witness family X and comparison ledger B transported by H, Q and routed by C.

The packet cannot export real-valued speed, synchronized clocks, metric tensors, Lorentz geometry, or any rate comparison not recorded as a row of B against U.

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket unary bound nonescape`

Proof. Project U from the carrier together with X and B. The bound row is a unary ledger coordinate, and every public comparison to it is recorded in the finite row B.

Use H, Q, C to transport and read the bound surface. Componentwise hsame movement, probe stability, and continuation routing preserve the same witness family and comparison ledger.

The package row P and local name row N bind the result to the $\mathsf{MaxCausalRate}^{\uparrow}$ certificate. Since the remaining excluded structures have no coordinate in R, the unary-bound surface cannot become a real-valued speed, synchronized-clock, metric-tensor, or Lorentz-geometry surface. □

Theorem 594.9 (MaxCausalRate configuration-locality obligation)**.** *For an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier*

$$R = (M, X, U, B, H, Q, C, P, N),$$

every causal witness row in X is anchored in the displayed $\mathrm{MultiHistConfig}^{\uparrow}$ *row M before it is compared with the shared unary bound row U. No consumer of R may read a witness from X as belonging to an observer history, unary slot, or probe route not supplied by M, H, Q, C, P, N.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket configuration locality obligation`

Proof. Project the carrier rows $M, X, U, B, H, Q, C, P, N$ from Definition 594.1. The configuration row M is the only source of observer-history anchors and selected unary slots.

Apply the checked TasteGate marker recorded in the closure status for this chapter, together with the single-carrier alignment theorem MaxCausalRateTasteGate_single_carrier_alignment. It aligns the single carrier surface with the displayed tuple, so the witness family X is read as part of the same carrier rather than as a second classifier surface.

The rows H, Q, C then preserve the configuration-local read under componentwise transport, probe stability, and visible continuation routes. The provenance and naming rows P, N bind the result to the same packet, so no witness can detach from M or acquire an unlisted observer-history source. □

Theorem 594.10 (MaxCausalRate comparison-ledger exactness)**.** *For an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *carrier*

$$R = (M, X, U, B, H, Q, C, P, N),$$

the comparison ledger B records exactly the finite comparisons between displayed witness rows of X and the shared unary bound row U. A consumer may transport and replay those comparisons through H, Q, C, P, N, but may not replace them by a real-valued speed function, a metric comparison, a synchronized-clock comparison, or a comparison against an unlisted bound.

Proof. Start with the finite carrier rows $M, X, U, B, H, Q, C, P, N$. The row B is specified as the comparison ledger for the witness family X against the shared unary bound row U.

Use Definition 594.5 to see that every displayed witness receives such a comparison, and use Definition 594.8 to see that no second bound coordinate is exported.

The checked TasteGate marker recorded in the closure status keeps the carrier read single-surfaced. Therefore B is not a partial external table and not an ambient numeric rate.

Finally transport through H, Q, C and bind by P, N. These rows preserve the same finite comparison ledger and add no metric, synchronized clock, Lorentz geometry, or host real-valued speed. □

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket comparison ledger exactness`

Theorem 594.11 (MaxCausalRate obligation closure package)**.** *The carrier, local* $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ *obligations, CrossHistCausal handoff, non-escape theorem, bound-comparison totality, witness-family stability, unary-bound non-escape, configuration-locality obligation, and comparison-ledger exactness assemble the obligation closure for every accepted packet*

$$R = (M, X, U, B, H, Q, C, P, N).$$

The closed obligation surface exports only the displayed $\mathrm{MultiHistConfig}^{\uparrow}$ *row* M*, witness family* X*, unary bound* U*, comparison ledger* B*, transport rows* H, Q*, routes* C*, provenance row* P*, and naming row* N*.*

Proof. The carrier definition supplies the tuple $M, X, U, B, H, Q, C, P, N$. The checked TasteGate marker recorded in the closure status confirms that this tuple is the single carrier surface used by the local naming obligations.

Apply Definition 594.3 and Definition 594.4. These results keep every consumer inside the displayed configuration, witness, bound, comparison, transport, route, provenance, and naming rows.

Next apply Definition 594.5, Definition 594.6, and Definition 594.8. They make the comparison ledger total over X, stable under H, Q, C, and tied to the single unary bound U.

Finally add the configuration-locality and comparison-ledger exactness obligations. Together they close the local $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$ obligation surface while continuing to exclude global frames, synchronized clocks, metric tensors, Lorentz geometry, and host real-valued speed measurements. □

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket obligation closure package`

Theorem 594.12 (MaxCausalRate scoped kernel boundary)**.** *Every downstream causal-rate read from an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *packet factors through the displayed rows*

$$M, X, U, B, H, Q, C, P, N$$

and their BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *ledgers. The read may consume the multi-Hist configuration, listed cross-Hist witnesses, shared unary bound, comparison ledger, transports, routes, provenance, and local name, but it cannot enter a global frame, synchronized-clock row, metric tensor, Lorentz geometry, or host real-valued speed surface.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket scoped kernel boundary`

Proof. Project the accepted packet through Definition 594.1. The rows $M, X, U, B, H, Q, C, P, N$ are the whole finite carrier surface.

Apply Definition 594.3 and Definition 594.5. These theorems route every causal witness read through the displayed configuration row, finite witness family, shared unary bound, and comparison ledger.

Next use Definition 594.6, Definition 594.9, and Definition 594.10. They keep the transported read inside H, Q, C, P, N and bind it to the same BHist, hsame, Cont, Pkg, and NameCert ledgers.

Finally apply Definition 594.4. Since the packet has no global-frame, synchronized-clock, metric, Lorentz, or real-speed coordinate, the displayed rows give the scoped kernel boundary for downstream reads. □

Theorem 594.13 (MaxCausalRate public unary-bound export)**.** *Every public* MaxCausalRate$^{\uparrow}$ *consumer reads the unary bound row U only through the single carrier tuple*

$$M, X, U, B, H, Q, C, P, N,$$

the CrossHistCausal$^{\uparrow}$ *handoff for the witness family X, and the existing TasteGate single-carrier marker* MaxCausalRateTasteGate_single_carrier_alignment*. The exported interface supplies a finite unary-bound ledger only: it exports no global frame, metric tensor, Lorentz geometry, synchronized clock, or real-valued speed.*

Proof. Project the accepted carrier. The row U is a coordinate of the displayed tuple $M, X, U, B, H, Q, C, P, N$, not a separate ambient rate surface.

Use Definition 594.3. Every consumer reaches the witness family X through displayed CrossHistCausal$^{\uparrow}$ rows before it reads the comparison ledger against U.

Now use the checked TasteGate single-carrier marker already recorded after Definition 594.2. It keeps the tuple as one carrier surface, so the public read cannot split U away from M, X, B, H, Q, C, P, N.

Finally apply Definition 594.4. Since the carrier has no coordinate for a global frame, metric tensor, Lorentz geometry, synchronized clock, or host real-valued speed, the public export is exactly the finite unary-bound row routed through the displayed packet. □

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket public unary bound export`

Theorem 594.14 (MaxCausalRate unary-bound consumer bridge)**.** *Every public causal-rate consumer bridge of an accepted* MaxCausalRate$^{\uparrow}$ *packet factors through the displayed* MultiHistConfig$^{\uparrow}$ *row M, the* CrossHistCausal$^{\uparrow}$ *witness family X, the shared unary bound U, the comparison ledger B, and the transport, provenance, continuation, and naming rows H, Q, C, P, N.*

The bridge uses the checked TasteGate marker MaxCausalRateTasteGate_single_carrier_alignment *as the single-carrier alignment guard. It exports only the same finite packet surface and does not claim Lorentz geometry, metric tensors, synchronized clocks, a global frame, or real-valued speed.*

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket unary bound consumer bridge`

Proof. Project the accepted carrier. The public consumer bridge begins with the same tuple $M, X, U, B, H, Q, C, P, N$ used by Definition 594.13.

Apply Definition 594.3. It routes every witness read in X through displayed CrossHistCausal$^{\uparrow}$ rows over the configuration M.

Next apply Definition 594.13 and Definition 594.10. These keep the shared bound U attached to the comparison ledger B and the structural rows H, Q, C, P, N.

Use the checked TasteGate marker recorded in the chapter as the single-carrier alignment guard. It prevents the consumer bridge from reading U as an abstract rate parameter detached from M, X, B, H, Q, C, P, N.

Finally apply Definition 594.4. Since the carrier has no global frame, metric tensor, Lorentz geometry, synchronized clock, or host real-valued speed coordinate, the bridge exposes only the displayed finite unary-bound packet. □

Theorem 594.15 (MaxCausalRate bridge consumer surface)**.** *Every bridge-facing public consumer of an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *packet factors through*

$$M, X, U, B, H, Q, C, P, N,$$

before any public causal-rate read. The consumer reads the $\mathrm{MultiHistConfig}^{\uparrow}$ *configuration row, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness family, the shared unary bound, the comparison ledger, and the single-carrier TasteGate guard as one finite packet surface.*

The bridge-facing surface contains no global frame, metric tensor, Lorentz geometry, synchronized clock, or real-valued speed.

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.MaxCausalRatePacket bridge consumer surface`

Proof. Project the accepted packet through Definition 594.14. This gives the displayed carrier tuple $M, X, U, B, H, Q, C, P, N$ as the only bridge-facing packet available before a public causal-rate read.

Use Definition 594.13. It keeps the shared unary bound U attached to the same public carrier tuple and prevents the consumer from reading U as an independent rate surface.

Next apply Definition 594.10. The comparison ledger B relates each displayed causal witness in X to the same unary bound U, so the public bridge consumes a finite ledger rather than a host-valued speed.

Finally apply Definition 594.4. Together with the single-carrier TasteGate guard already used in Definition 594.14, the non-escape row excludes global frames, metric tensors, Lorentz geometry, synchronized clocks, and real-valued speed from the public bridge surface. □

The Lean TasteGate surface for $\mathsf{MaxCausalRate}^{\uparrow}$ must preserve explicit Nontrivial and FieldFaithful instances for this carrier before bridge-facing claims read the unary bound as a public maximum-rate row. Those instances keep the selected pair witnesses, shared unary upper bound, comparison totality, and consumer bridge rows attached to one carrier rather than a collapsed encoding.

Closure status for $\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$.

Constructive story: A $\mathsf{MaxCausalRate}^{\uparrow}$ packet is generated from a $\mathrm{MultiHistConfig}^{\uparrow}$ row, finitely many $\mathsf{CrossHistCausal}^{\uparrow}$ witness rows over selected history pairs and unary slots, one shared Unary bound row, finite bound comparisons, displayed hsame and psame transport, visible Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 594.1, Definition 594.2, Definition 594.3, Definition 594.4, Definition 594.5, Definition 594.6, Definition 594.8, Definition 594.9, Definition 594.10, Definition 594.11, Definition 594.12, Definition 594.13, Definition 594.14, and Definition 594.15 bind the finite multi-Hist configuration, displayed cross-Hist causal witnesses, shared unary bound, finite comparison ledger, transport rows, visible routes, package provenance, local naming row, downstream read boundary, public unary-bound export, and paper bridge surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MaxCausalRate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.MaxCausalRateUp.taste gate`

Not claimed: This chapter does not close Lorentz geometry, metric light-cones, curved spacetime, quantum-state certificates, synchronized clocks, global frames, or real-valued speed measurements.

Upgrade path: Paper-side mature closure is complete for the finite unary-bound causal ledger, while Lean verification and bridge checking remain separate.

Chapter 595

A Concrete Naming Certificate for $\mathsf{ClassifierBoundary}^{\uparrow}$

The $\mathsf{ClassifierBoundary}^{\uparrow}$ surface realizes the type-checking boundary of chapter 9367 as a finite NameCert packet. It records accepted Ext-membership rows, refused non-member rows, Cont preservation routes, SigRel readback, package provenance, and local naming, while keeping host-language typing judgments and unrecorded semantic models outside the packet.

The same boundary is the finite Ext/Cont/SigRel surface consumed by chapter 476, chapter 483, and chapter 590. Those sibling uses are, respectively, accepted membership, subject-reduction preservation, and classifier-morphism transport across the displayed signature relation.

Definition 595.1 (Classifier-boundary carrier)**.** *A* $\mathsf{ClassifierBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, A, R, T, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row of candidate histories whose membership is inspected by the classifier;*
2. *A is the accepted boundary row, listing exactly the displayed histories admitted by the* Ext *classifier;*
3. *R is the refused boundary row, listing exactly the displayed histories for which the packet exposes a finite non-member refusal row;*
4. *T is the* Cont *preservation row, recording that every admitted source route followed by a displayed continuation remains in the accepted boundary;*
5. *G is the* SigRel *readback row, exposing the signature-level classifier read used by downstream type-certificate and compiler-morphism consumers;*
6. *H stores componentwise* hsame *transports for the source, accepted, refused, preservation, readback, package, and naming rows;*
7. *C stores the displayed* Cont *routes by which consumers read source histories, accepted rows, refused rows, preservation rows, readback rows, provenance, and local naming;*
8. *P is the* Pkg *provenance over* BHist*,* Ext*,* Cont*,* SigRel*,* hsame*,* Pkg*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierBoundary}^{\uparrow}}$ *row.*

The local classifier relates two packets exactly when their displayed source, accepted, refused, preservation, readback, transport, route, provenance, and naming rows compare through H and C. The carrier contains no host type checker, no external semantic model, no hidden proof search oracle, and no quotient of histories.

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryTasteGate single carrier alignment`

Theorem 595.2 (ClassifierBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClassifierBoundary}^{\uparrow}}$ *surface for Definition 595.1 consists of five finite obligations:*

1. *accepted* Ext*-membership exposure, requiring every accepted consumer read to be one of the displayed rows of A reached from S;*
2. *refused non-member exposure, requiring every refused consumer read to be one of the displayed rows of R and not a hidden semantic rejection;*
3. Cont *preservation, requiring a displayed continuation from an accepted row to land again in the accepted boundary through T;*
4. SigRel *readback, requiring downstream consumers to read the public classifier signature only through the row G and its routes in C;*
5. *NameCert provenance, requiring all accepted, refused, preservation, and readback rows to remain under the package row P and the local naming row N.*

Consequently the packet gives a concrete public boundary for classifier membership: accepted histories, refused histories, continuation preservation, signature readback, and provenance are all finite BHist *rows.*

Proof. Carrier exposure is projection from $B = (S, A, R, T, G, H, C, P, N)$. The accepted and refused sides are both visible packet rows, so membership and refusal are boundary data rather than host-side judgments.

For accepted reads, use the route from S to A recorded in C. For refused reads, use the route from S to R recorded in the same continuation ledger. These two rows exhaust the membership decision surface exposed by the packet.

Preservation is then read through T. A consumer that starts from an accepted row and follows a displayed continuation has no route except the one listed in T, and the resulting accepted row is again transported through H and read through C.

Finally read the signature-level view through G and package it through P and N. Since no coordinate of the carrier is a host checker, external semantic model, hidden proof search oracle, or quotient of histories, the local NameCert surface cannot export such data. □

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier namecert obligations`

Theorem 595.3 (ClassifierBoundary public boundary)**.** *Every public consumer of an accepted* $\mathsf{ClassifierBoundary}^{\uparrow}$ *packet factors through the displayed rows $S, A, R, T, G, H, C, P, N$. Accepted membership reads use A, refused non-member reads use R, subject-reduction-style reads use T, signature readback uses G, and all routes preserve their provenance through P and N.*

Proof. Split the consumer request by the row it asks to read. Accepted membership reaches A from S, while refused non-membership reaches R from S.

Continuation preservation reaches T only after the accepted row has been displayed. Thus the boundary uses the same finite source and accepted rows for both type-checking membership and subject-reduction-style continuation.

Signature readback reaches G, and provenance reaches P and N. These rows exhaust the carrier, so no consumer can bypass the finite packet to obtain a host type checker or an external semantic model. □

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier public boundary`

Theorem 595.4 (ClassifierBoundary scoped dependency route)**.** *For an accepted* ClassifierBoundary$^{\uparrow}$ *carrier*

$$B = (S, A, R, T, G, H, C, P, N),$$

the accepted-membership route, refused non-member route, and continuation-preservation route all factor through the displayed rows $S, A, R, T, G, H, C, P, N$. The dependency scope is exactly the finite BHist *rows, the* Ext *classifier rows, displayed* Cont *routes,* SigRel *readback, componentwise* hsame *transports,* Pkg *provenance, and the local* NameCert *row; it contains no host type checking, external semantic model, proof-search oracle, or quotient history.*

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier scoped dependency route`

Proof. Project the carrier of Definition 595.1. The rows S, A, R, T are the only coordinates that expose candidate histories, accepted boundary rows, refused boundary rows, and continuation preservation.

Use the NameCert obligations of Definition 595.2. Accepted reads are required to pass from S to A, refused reads from S to R, and displayed continuation preservation through T.

Finally use the public boundary of Definition 595.3. The consumer still reads transports, routes, provenance, and local naming only through H, C, P, N, while signature readback remains the row G. Since the carrier has no coordinate for host type checking, external semantic models, proof-search oracles, or quotient histories, none of those objects lies in the scoped dependency route. □

Theorem 595.5 (ClassifierBoundary signature consumer determinacy)**.** *For an accepted* ClassifierBoundary$^{\uparrow}$ *carrier*

$$B = (S, A, R, T, G, H, C, P, N),$$

every consumer-visible SigRel *readback is determined by the displayed row G together with its* hsame *transports in H,* Cont *read routes in C,* Pkg *provenance in P, and local* NameCert *row N. Thus two consumers with the same finite* BHist *source, accepted row, refused row, continuation row, signature row, transport row, route row, package row, and naming row receive the same public classifier signature, with no host type checking, external semantic model, proof-search oracle, or quotient history supplying another readback.*

Proof. Project $B = (S, A, R, T, G, H, C, P, N)$ and isolate the signature coordinate. The carrier definition makes G the displayed SigRel readback row and gives no second coordinate for classifier-signature output.

Invoke Definition 595.2. Its readback obligation requires downstream consumers to read the public classifier signature only through G and the routes stored in C, under the package and naming rows P, N.

Now apply Definition 595.3. Every public consumer factors through the same displayed rows, so agreeing rows force the same readback surface. The excluded host checker, semantic model, proof-search oracle, and quotient-history data are not carrier coordinates and cannot determine a second signature. □

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier sigrel consumer determinacy`

Theorem 595.6 (ClassifierBoundary Ext and Cont preservation scope)**.** *For an accepted* ClassifierBoundary$^{\uparrow}$ *carrier*

$$B = (S, A, R, T, G, H, C, P, N),$$

the Ext *membership surface and the* Cont *preservation surface are preserved exactly within the displayed dependency scope over* BHist*,* Ext*,* Cont*,* SigRel*,* hsame*,* Pkg*, and* NameCert*. Accepted membership is read through A, refused membership through R, preservation through T, signature readback through G, and all transported consumer access through H, C, P, N; no host type checking, external semantic model, proof-search oracle, or quotient history enlarges the scope.*

Proof. Start from the carrier projection. The Ext data are the source, accepted, and refused rows S, A, R, while the preservation data are the displayed continuation rows in T.

Use Definition 595.2 to bind the local obligations. It requires accepted exposure, refused exposure, continuation preservation, signature readback, and provenance to remain under the finite NameCert surface.

The public-boundary theorem Definition 595.3 then gives the consumer factorization through $S, A, R, T, G, H, C, P, N$. Because the carrier omits host type checking, external semantic models, proof-search oracles, and quotient histories, preservation stays inside the stated BEDC primitives. □

Lean4 ✓ `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier ext cont preservation scope`

Closure status for NameCert$_{\text{ClassifierBoundary}^{\uparrow}}$**.**

Constructive story: A ClassifierBoundary$^{\uparrow}$ packet is generated from source histories, accepted Ext rows, refused non-member rows, Cont preservation rows, SigRel readback, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClassifierBoundary}^{\uparrow}}$, publicClosed)

Scope closed: Definition 595.1, Definition 595.2, and Definition 595.3 bind the finite classifier-boundary NameCert surface for accepted membership, refused non-membership, continuation preservation, readback, and provenance.

Formal verification: FormalStatus(NameCert$_{\text{ClassifierBoundary}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ClassifierBoundaryUp.ClassifierBoundaryCarrier\_scoped\_dependency\_route`

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, bridge status, host type checking, external semantic models, hidden proof search, quotient histories, or compiler morphism preservation.

Upgrade path: A higher closure grade requires Lean-side bridge checking for the public classifier-boundary surface.

Chapter 596

A Concrete Naming Certificate for $\mathsf{SubordinateModulusCover}^{\uparrow}$

The $\mathsf{SubordinateModulusCover}^{\uparrow}$ packet names the finite cover returned before the compact-uniform-continuity rational lower-bound fold. It sits between the compact finite-net carrier and the $\mathsf{UniformModulus}^{\uparrow}$ consumer: the packet records bundled centers, pointwise-modulus precision rows, subordinate positive-radius rows, half-radius coverage witnesses, rational comparison ledgers, and the local provenance needed to hand the bundle to the uniform-modulus fold.

The packet consumes the finite-net coverage surface of chapter 60, the pointwise-modulus data supplied by chapter 130, and the metric and rational comparison rows of chapter 57 and chapter 37. Its output is the finite subordinate bundle handed to chapter 369, before any uniform precision row is chosen.

Definition 596.1 (SubordinateModulusCover carrier)**.** *A* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

with the following visible rows:

1. *E is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history requested by the compact-continuous consumer;*
2. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine returned by the compact finite-net certificate;*
3. *C assigns to each bundled probe its displayed center history;*
4. *R assigns to each bundled probe a positive subordinate radius history;*
5. *D assigns to each bundled probe the pointwise-modulus precision history supplied at its center;*
6. *M records the accepted pointwise-modulus row for the tolerance fraction read from E;*
7. *H records the half-radius coverage row saying that every source history is covered by a bundled center at the displayed subordinate radius;*
8. *F records the rational comparison ledger witnessing that every subordinate radius row is below its pointwise precision row;*
9. *L records componentwise* hsame *stability for the tolerance, bundle, centers, radius rows, precision rows, pointwise rows, coverage rows, and rational comparisons;*
10. *Q records the displayed* Cont *routes through which compact and continuous consumers read the packet;*

11. *P is the* Pkg *provenance over* Compact$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, Rat$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and the local naming rows;*

12. *N is the local* NameCert$_{\text{SubordinateModulusCover}^{\uparrow}}$ *row.*

The carrier exports no open cover, no external radius selector, no host minimum, and no uniform precision row. Its role is exactly the finite subordinate bundle consumed before the UniformModulus$^{\uparrow}$ *lower-bound fold.*

Lean4 (definition) `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverCarrier`

Theorem 596.2 (SubordinateModulusCover NameCert obligations)**.** *The local* NameCert$_{\text{SubordinateModulusCover}^{\uparrow}}$ *surface for Definition 596.1 consists of carrier admission for $S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$, classifier stability under the componentwise* hsame *rows in L, compact coverage exactness for the half-radius rows in H, pointwise-modulus compatibility for the rows in M and D, rational subordination exactness for the comparison rows in F, and consumer non-escape through the displayed* Cont, Pkg, *and local naming rows.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverTasteGate single carrier alignment`

Proof. Carrier admission is the displayed finite packet. The tolerance row, nonempty bundle, centers, subordinate radii, pointwise precision rows, pointwise-modulus rows, coverage witnesses, rational comparisons, transport ledger, continuation routes, package provenance, and local naming row are all visible coordinates.

Classifier stability is componentwise over those coordinates. The transport ledger L can rename bundled probes, centers, radius histories, precision histories, coverage rows, and comparison rows only through the displayed hsame entries, and the routes in Q preserve the compact and continuous reads attached to the same bundle.

Coverage exactness is the half-radius row H. Pointwise compatibility is the row M at each center together with the precision row in D. Rational subordination is the row F, which binds each radius row in R below the corresponding precision row in D.

These rows exhaust the consumer surface. Since no coordinate stores an open cover, an external selector, a host minimum, or a uniform precision history, the local naming certificate cannot export any of those objects. □

Theorem 596.3 (SubordinateModulusCover uniform-modulus handoff)**.** *Every accepted* SubordinateModulusCover *packet hands a* UniformModulus$^{\uparrow}$ *consumer exactly the nonempty* ProbeBundle *spine, bundled centers, positive subordinate radius rows, pointwise-modulus precision rows, half-radius coverage rows, and rational subordination comparisons needed by the finite lower-bound fold. The handoff does not yet choose the uniform precision; it supplies the finite input ledger from which the* UniformModulus$^{\uparrow}$ *fold computes that precision.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverUniformHandoff`

Proof. Project the accepted packet through Definition 596.1. The consumer receives $\mathcal{P}$, C, R, D, M, H, and F through the displayed routes in Q and the provenance rows P, N.

The nonempty bundle and coverage rows are the compact side of the handoff. The pointwise-modulus and precision rows are the continuous side. The rational comparison ledger records the subordination that permits the later lower-bound fold to stay below every bundled half-radius row.

No fold output is present in the carrier. Therefore the handoff is precisely the finite subordinate-cover input to the $\mathsf{UniformModulus}^{\uparrow}$ rational fold, with no external modulus selector and no host-level compactness object. □

Theorem 596.4 (SubordinateModulusCover carrier obligation)**.** *Every accepted* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *read is admitted through the single carrier*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

of Definition 596.1. The TasteGate alignment of Definition 596.2 binds the positive tolerance row, nonempty probe bundle, centers, subordinate radii, pointwise precision rows, pointwise-modulus row, half-radius coverage row, rational comparison ledger, transport rows, continuation routes, package provenance, and local naming row to this one packet.

Proof. Project an accepted read through the carrier definition. The rows available to the consumer are exactly $E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N$.

Apply Definition 596.2. Its checked TasteGate row states the local NameCert obligations on the same displayed carrier, including carrier admission and consumer non-escape.

Since the carrier exports no open cover, external radius selector, host minimum, or uniform precision row, admission cannot be transferred to a second packet. The accepted read is therefore a read of the single displayed subordinate-modulus-cover carrier. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverCarrierObligation`

Theorem 596.5 (SubordinateModulusCover classifier stability obligation)**.** *Classifier transport for an accepted* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *packet is componentwise on the rows of*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N).$$

The TasteGate alignment of Definition 596.2 permits transport only through the hsame *ledger* L *and the displayed* Cont *routes* Q*, preserving the tolerance, bundle, centers, radius rows, precision rows, pointwise-modulus rows, coverage rows, rational comparisons, package provenance, and local naming row.*

Proof. Start with the single carrier S. The only classifier-stability coordinate is the componentwise transport ledger L, and the only consumer routes are the displayed continuations in Q.

Use Definition 596.2. The TasteGate row records classifier stability under L together with compact coverage exactness, pointwise-modulus compatibility, rational subordination exactness, and consumer non-escape for the same packet.

Thus a transported packet preserves each visible coordinate rather than replacing the carrier or changing the bundle. No parameter-transport equivalence, open-cover selector, host minimum, or uniform precision row is introduced by classifier transport. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverClassifierStabilityObligation`

Theorem 596.6 (SubordinateModulusCover ledger handoff obligation)**.** *The compact, continuous, and uniform-modulus handoff ledger of an accepted* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *packet is exactly the route through*

$$\mathcal{P}, C, R, D, M, H, F, L, Q, P, N$$

inside the single carrier $S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$*. The handoff exposes the finite nonempty bundle, bundled centers, subordinate radius rows, pointwise-modulus precision rows, half-radius coverage witnesses, and rational subordination comparisons needed before the* $\mathsf{UniformModulus}^{\uparrow}$ *lower-bound fold, and it exposes no fold output.*

Proof. Read the handoff rows from the carrier definition. The compact side supplies $\mathcal{P}$, C, and H; the continuous side supplies D and M; the rational subordination ledger supplies R and F; and L, Q, P, N transport, route, and record the public read.

Apply Definition 596.2. Its TasteGate row binds coverage exactness, pointwise-modulus compatibility, rational subordination exactness, and non-escape to the same carrier S.

Finally use Definition 596.3. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives precisely the finite input ledger for the later lower-bound fold, while the uniform precision chosen by that fold remains outside the subordinate cover packet. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverLedgerHandoffObligation`

Theorem 596.7 (SubordinateModulusCover scope dependency)**.** *Every compact-uniform handoff row of an accepted* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *packet factors through* $\mathsf{Compact}^{\uparrow}$ *finite-net coverage,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus data,* $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{Rat}^{\uparrow}$ *comparison rows,* ProbeBundle *induction,* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer. The factorization stays inside*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N),$$

and it contains no open-cover compactness, host minimum, or external selector.

Proof. Project the carrier $S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$. The bundle spine $\mathcal{P}$ and center rows C supply the finite-net side, while R, D, M, H, F record subordinate radii, pointwise precision, pointwise-modulus data, half-radius coverage, and rational comparison.

Apply Definition 596.2. The local $\mathsf{NameCert}_{\mathsf{SubordinateModulusCover}^{\uparrow}}$ surface binds those rows to BHist, hsame, Cont, Pkg, and local naming, so the dependency cannot leave the displayed packet.

Use Definition 596.3 and Definition 596.6. They expose the compact, continuous, metric, rational, and finite-bundle rows to the $\mathsf{UniformModulus}^{\uparrow}$ consumer before the lower-bound fold is read.

Thus the scoped dependency is exhausted by the displayed carrier and its existing handoff obligations. Since S has no coordinate for open-cover compactness, host minima, or external selectors, none of those objects can enter the compact-uniform handoff. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverScopeDependency`

Theorem 596.8 (SubordinateModulusCover compact coverage exactness)**.** *Let*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

be an accepted $\mathsf{SubordinateModulusCover}^{\uparrow}$ *carrier. Every source history covered by the packet is covered by a bundled center from the nonempty* ProbeBundle *spine* $\mathcal{P}$*, through the half-radius coverage rows* H *and the displayed consumer routes* Q*. The coverage row is internal to the carrier and uses no host open-cover compactness, external radius selector, host minimum, or uniform precision output.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverCompactCoverageExactness`

Proof. Project the carrier by Definition 596.1. The nonempty bundle spine $\mathcal{P}$, center row C, subordinate radius row R, half-radius coverage row H, and continuation route row Q are all visible coordinates of the same packet.

Apply Definition 596.2. Its TasteGate alignment binds compact coverage exactness to the half-radius rows in H for the displayed carrier S.

The public coverage read is therefore the route through $\mathcal{P}, C, R, H, Q, P, N$. Since the carrier definition contains no open-cover object, external radius selector, host minimum, or uniform precision row, none of those objects can supply the coverage witness. □

Theorem 596.9 (SubordinateModulusCover pointwise compatibility)**.** *Let*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

be an accepted $\mathsf{SubordinateModulusCover}^{\uparrow}$ *carrier. Every precision row in D is supplied by the pointwise-modulus row M at the corresponding bundled center from $\mathcal{P}$ and is transported only by the componentwise* hsame *ledger L. The compatibility row does not choose an external modulus, host minimum, or uniform precision output.*

Proof. Read the relevant coordinates from Definition 596.1. The center row is C, the precision row is D, the pointwise-modulus row is M, and the only componentwise transport ledger is L.

Use Definition 596.2. The TasteGate alignment lists pointwise-modulus compatibility for the rows in M and D and classifier stability only through L.

Thus each precision read is attached to the corresponding bundled center and its pointwise-modulus row. The carrier has no coordinate for an external modulus selector, a host minimum, or a uniform precision result, so compatibility cannot be sourced from any of those objects. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverPointwiseCompatibility`

Theorem 596.10 (SubordinateModulusCover rational subordination exactness)**.** *Let*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

be an accepted $\mathsf{SubordinateModulusCover}^{\uparrow}$ *carrier. Each subordinate radius row in R is below its matching pointwise precision row in D through the rational comparison ledger F. This exactness statement is a finite ledger row of the packet and does not assert compact host minima, external radius selection, or a completed uniform-modulus output.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverRationalSubordinationExactness`

Proof. Start with the carrier definition. It displays subordinate radius rows R, pointwise precision rows D, and the rational comparison ledger F as separate coordinates of the same finite packet.

Apply Definition 596.2. Its TasteGate row names rational subordination exactness for the comparison rows in F, binding each radius row in R below the corresponding precision row in D.

The handoff remains finite by Definition 596.6. The comparison is consumed as part of the subordinate-cover ledger before the $\mathsf{UniformModulus}^{\uparrow}$ lower-bound fold, so no host minimum, external radius selector, or fold output is introduced here. □

Theorem 596.11 (SubordinateModulusCover carrier admission)**.** *Carrier admission for* SubordinateModulusCover *is exactly the displayed packet*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N).$$

It consumes the $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert *rows shown in Definition 596.1, and it admits no host open cover, external selector, host minimum, or uniform-modulus fold output.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverCarrierAdmission`

Proof. Project the carrier definition. The listed packet contains the tolerance row, nonempty probe bundle, centers, subordinate radii, pointwise precision rows, pointwise-modulus row, half-radius coverage row, rational comparison ledger, transport, replay, provenance, and local naming.

Apply the NameCert obligations of Definition 596.2. They bind carrier admission, classifier stability, coverage exactness, pointwise compatibility, rational subordination exactness, and consumer non-escape to that same packet.

Use compact coverage exactness, pointwise compatibility, rational subordination exactness, and the uniform handoff theorem. These results account for every analytic row needed by the L10 compact-uniform route, so admission is exhausted by S and cannot come from any host object outside the displayed carrier. □

Theorem 596.12 (SubordinateModulusCover radius stability)**.** *Transport through the* hsame *ledger L preserves the subordinate radius rows R, pointwise precision rows D, half-radius coverage rows H, and rational comparison rows F without changing the* ProbeBundle *spine $\mathcal{P}$. Thus a transported* SubordinateModulusCover$^{\uparrow}$ *packet presents the same finite subordinate bundle to every* UniformModulus$^{\uparrow}$ *consumer.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverRadiusStability`

Proof. Read the transport coordinate L from Definition 596.1. It is componentwise over the tolerance, bundle, centers, radius histories, precision histories, pointwise rows, coverage rows, and rational comparison rows.

Apply the NameCert obligations and the classifier stability obligation. They restrict transport to the displayed rows and preserve the continuation routes Q through which compact and continuous consumers read the packet.

Now use compact coverage exactness, pointwise compatibility, rational subordination exactness, and the uniform handoff theorem. Since the bundle spine $\mathcal{P}$ remains the same displayed spine and the rows R, D, H, F are transported componentwise, the handoff ledger remains stable without introducing a new bundle, host minimum, or fold output. □

Theorem 596.13 (SubordinateModulusCover ledger exactness)**.** *The handoff ledger exposed from an accepted* SubordinateModulusCover$^{\uparrow}$ *packet to* UniformModulus$^{\uparrow}$ *is exhausted by the finite bundle $\mathcal{P}$, bundled centers C, subordinate radii R, pointwise precision rows D, half-radius coverage rows H, and rational comparison rows F, together with the displayed transport, replay, provenance, and local naming rows. It contains no host minimum, no external radius selector, and no fold output.*

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverLedgerExactness`

Proof. Project the handoff rows from Definition 596.1. The compact side is $\mathcal{P}, C, H$, the pointwise side is D, M, and the rational subordination side is R, F.

Apply the NameCert obligations, compact coverage exactness, pointwise compatibility, and rational subordination exactness. These theorems attach each ledger row to the same accepted packet and prevent any read from bypassing L, Q, P, N.

Finally use the uniform handoff theorem and the carrier admission theorem above. The UniformModulus$^{\uparrow}$ consumer receives precisely the finite subordinate-cover input ledger before the lower-bound fold, while the fold result and any host minimum remain outside the packet. □

Theorem 596.14 (SubordinateModulusCover TasteGate Stage-F discipline)**.** *The single carrier*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N)$$

supplies the Stage-F discipline for SubordinateModulusCover$^{\uparrow}$*: the finite* ProbeBundle *spine is nontrivial, the visible fields are faithful to the carrier, and every* UniformModulus$^{\uparrow}$ *consumer must pass through compact coverage, pointwise-modulus rows, rational subordination,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any uniform-modulus fold is read.*

Proof. Start with carrier admission from Definition 596.11. It identifies S as the single accepted packet and keeps the nonempty ProbeBundle spine, center rows, radius rows, precision rows, pointwise rows, coverage rows, comparison rows, transport, replay, provenance, and local naming visible.

Apply the NameCert obligations of Definition 596.2. They bind the carrier fields to the same TasteGate surface and prevent consumer reads from bypassing the displayed hsame, Cont, Pkg, and local naming rows.

Use radius stability and ledger exactness from Definition 596.12 and Definition 596.13. The compact coverage row, pointwise-modulus row, and rational subordination row remain faithful to S through transport, and the UniformModulus$^{\uparrow}$ consumer sees only the finite subordinate-cover ledger before the fold output exists. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverTasteGate stageF discipline`

Theorem 596.15 (SubordinateModulusCover compact-uniform consumer boundary)**.** *Every* Compact$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*, or* Met$^{\uparrow}$ *consumer that reads an accepted* SubordinateModulusCover$^{\uparrow}$ *packet must expose the finite* ProbeBundle *spine, pointwise-modulus rows, rational subordination ledger, half-radius coverage row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before any* UniformModulus$^{\uparrow}$ *fold is read. The consumer boundary is exactly the displayed route through the carrier*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N),$$

and it exports no host compact minimum, open-cover selector, external modulus choice, or completed uniform precision.

Proof. Carrier admission identifies the only packet available to the consumer. The rows $\mathcal{P}, C, R, D, M, H, F, L, Q, P$ are displayed coordinates of the single carrier, and the tolerance row E only records the compact-uniform request that the packet answers.

Apply radius stability, ledger exactness, pointwise compatibility, and rational subordination exactness. These results keep the finite bundle spine, centerwise precision rows, subordinate radii, half-radius coverage row, rational comparison ledger, and transport rows attached to the same accepted packet through hsame and Cont replay.

Finally apply the Stage-F discipline. It forces every UniformModulus$^{\uparrow}$ consumer to read compact coverage, pointwise-modulus data, rational subordination, provenance, replay, and the local naming row before the lower-bound fold is available. Hence a compact, continuous, or metric consumer reaches the fold only through the finite subordinate-cover boundary, with no host compact minimum, selector, external modulus choice, or completed uniform precision. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverCompactUniformConsumerBoundary`

Theorem 596.16 (SubordinateModulusCover choice-free handoff)**.** *Every accepted* SubordinateModulusCover$^{\uparrow}$ *packet exposes the finite rows* $\mathcal{P}, C, R, D, M, H, F, L, Q, P, N$ *to a* UniformModulus$^{\uparrow}$ *consumer before any lower-bound fold is read. No row may be retyped as an external radius selector, host minimum, open-cover compactness witness, or completed uniform precision.*

Proof. Start from carrier admission in Definition 596.1. The rows $\mathcal{P}, C, R, D, M, H, F, L, Q, P, N$ are displayed coordinates of the same finite packet, and the tolerance row E only supplies the request read by that packet.

Compact coverage exactness, pointwise compatibility, and rational subordination exactness attach the bundle, centers, subordinate radii, precision rows, pointwise-modulus row, coverage row, and comparison ledger to the same carrier. The uniform handoff theorem then exposes those finite rows to the $\mathsf{UniformModulus}^{\uparrow}$ consumer before the lower-bound fold produces any output.

Ledger exactness rules out the remaining escape routes. Since the carrier contains no selector coordinate, no host minimum coordinate, no open-cover compactness row, and no completed uniform precision row, none of the displayed rows can be retyped as one of those objects. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverChoiceFreeHandoff`

Theorem 596.17 (SubordinateModulusCover finite-selection scope)**.** *Every accepted* $\mathsf{SubordinateModulusCover}^{\uparrow}$ *packet exposes the nonempty* ProbeBundle *spine, bundled centers, subordinate positive-radius rows, pointwise precision rows, half-radius coverage rows, and rational comparison ledger as the full finite-selection scope consumed by* $\mathsf{UniformModulus}^{\uparrow}$*. No host compact minimum, open-cover selector, or completed uniform precision row is present in that scope.*

Proof. Project the accepted packet through Definition 596.1. The finite-selection data are exactly the bundle $\mathcal{P}$, centers C, subordinate radii R, pointwise precision rows D, half-radius coverage rows H, and rational comparison ledger F, with L, Q, P, N supplying transport, replay, provenance, and local naming.

Apply Definition 596.2. Its carrier admission and consumer non-escape clauses bind those rows to the same local $\mathsf{NameCert}_{\mathsf{SubordinateModulusCover}^{\uparrow}}$ surface, while compact coverage exactness, pointwise compatibility, and rational subordination exactness account for the three analytic faces of the packet.

Use Definition 596.8, Definition 596.9, and Definition 596.10. These theorems attach coverage, centerwise precision, and radius comparison to the displayed finite carrier rather than to an ambient compact object.

Finally apply Definition 596.3 and Definition 596.16. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the finite subordinate-cover ledger before the lower-bound fold is read, so the scope contains neither a host compact minimum nor an open-cover selector nor a completed uniform precision row. □

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverFiniteSelectionScope`

Theorem 596.18 (SubordinateModulusCover obligation-closure route)**.** *The displayed carrier, TasteGate single-carrier alignment, carrier admission, classifier stability, compact coverage exactness, pointwise compatibility, rational subordination exactness, ledger exactness, choice-free handoff, and finite-selection scope form the obligation surface for* $\mathsf{SubordinateModulusCover}^{\uparrow}$*. These rows remain on the single carrier*

$$S = (E, \mathcal{P}, C, R, D, M, H, F, L, Q, P, N),$$

and they expose no host compact minimum, open-cover selector, external modulus choice, completed uniform precision, or carrier outside the packet.

Lean4 ✓ `BEDC.Derived.SubordinateModulusCoverUp.SubordinateModulusCoverObligationClosureRoute`

Proof. Start with the carrier of Definition 596.1 and the TasteGate row of Definition 596.2. Together they bind the tolerance, nonempty probe bundle, centers, subordinate radii, pointwise precision rows, pointwise-modulus row, half-radius coverage row, rational comparison ledger, transport, replay, provenance, and local naming to the same packet S.

Use carrier admission and classifier stability from Definition 596.11 and Definition 596.5. These results keep accepted reads and transported reads on the displayed carrier and prevent a consumer from replacing S by an external compact or modulus object.

Apply compact coverage exactness, pointwise compatibility, and rational subordination exactness. The theorems Definition 596.8, Definition 596.9, and Definition 596.10 account for the coverage, pointwise-modulus, and radius-comparison rows without leaving the packet.

Finally combine ledger exactness, choice-free handoff, and finite-selection scope from Definition 596.13, Definition 596.16, and Definition 596.17. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives exactly the finite subordinate-cover ledger before the fold output, so the named rows form the obligation-closure route and no hidden selection or completed uniform precision is exported. □

Falsifiable prediction: Every accepted $\mathsf{SubordinateModulusCover}^{\uparrow}$ consumer reaches $\mathsf{UniformModulus}^{\uparrow}$ only after displaying compact coverage, continuousmap pointwise-modulus rows, metric radius rows, rat comparison rows, rational subordination, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compact, continuousmap, metric, rat, uniformmodulus: no listed sibling is bijective to $\mathsf{SubordinateModulusCover}^{\uparrow}$ because this packet jointly carries a nonempty probe bundle, bundled centers, subordinate radius rows, pointwise precision rows, half-radius coverage, rational comparison, transport, replay, provenance, and local naming before the uniform-modulus fold.

Closure status for $\mathsf{NameCert}_{\mathsf{SubordinateModulusCover}^{\uparrow}}$.

Constructive story: A $\mathsf{SubordinateModulusCover}^{\uparrow}$ packet is generated from a positive tolerance row, a nonempty ProbeBundle spine, bundled centers, subordinate positive-radius rows, pointwise-modulus precision rows, half-radius coverage rows, rational comparison ledgers, transport rows, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubordinateModulusCover}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 596.1, Definition 596.2, Definition 596.8, Definition 596.9, Definition 596.10, and Definition 596.3 register the finite subordinate cover used before the compact-uniform-continuity rational lower-bound fold.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubordinateModulusCover}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the rational lower-bound fold, the final uniform-modulus ledger, compactness of host open covers, external modulus selection, or bridge checking.

Upgrade path: Scoped closure requires Lean checking of carrier admission, classifier stability, coverage exactness, pointwise compatibility, rational subordination exactness, and the uniform-modulus handoff for the same displayed packet.

Chapter 597

A Concrete Naming Certificate for $\mathsf{ContinuationTermination}^{\uparrow}$

The $\mathsf{ContinuationTermination}^{\uparrow}$ interface packages the finite termination and behaviour-classifier reading of a Cont trace. It records only displayed histories, displayed continuation rows, componentwise transport, package provenance, and a local naming certificate.

The finite trace row τ is consumed through chapter 469 before the terminal witness and behaviour-classifier rows are exposed. $\mathsf{ContinuationTrace}^{\uparrow}$ supplies finite replay for the displayed Cont trace; $\mathsf{ContinuationTermination}^{\uparrow}$ adds terminal witness, behaviour classifier, transport, provenance, and local NameCert rows.

Definition 597.1 (Continuation termination carrier)**.** *A* $\mathsf{ContinuationTermination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (s, t, \tau, u, b, h, p, n)$$

with the following visible coordinates:

1. *s is the start history;*
2. *t is the terminal history;*
3. *τ is the finite* Cont *trace from s to t, displayed as the finite list of intermediate* BHist *rows and transition marks;*
4. *u is the terminal witness row, naming the fact that no further displayed* Cont *row is exported from t inside this packet;*
5. *b is the behaviour-classifier row comparing start, terminal, and trace-visible histories only through the displayed classifier;*
6. *h stores componentwise* hsame *transports for the start history, terminal history, trace rows, terminal witness row, behaviour-classifier row, provenance row, and local name;*
7. *p is the* Pkg *provenance over* BHist, Cont, hsame, *the behaviour classifier, terminal-witness row, and* NameCert*;*
8. *n is the local* $\mathsf{NameCert}_{\mathsf{ContinuationTermination}^{\uparrow}}$ *row.*

The carrier contains no host evaluator, no quotient behaviour class, no unbounded trace, and no classical decision procedure for termination.

Lean4 ✓ `BEDC.Derived.ContinuationTerminationUp.ContinuationTerminationTasteGate single carrier alignment`

Definition 597.2 (Continuation termination classifier)**.** *For accepted carriers*

$$T = (s, t, \tau, u, b, h, p, n) \quad \textit{and} \quad T' = (s', t', \tau', u', b', h', p', n'),$$

the classifier $\mathsf{ContinuationTerminationClassifier}(T, T')$ *compares only the displayed start histories, terminal histories, finite* Cont *traces, terminal witness rows, behaviour-classifier rows, componentwise* hsame *transports,* Pkg *provenance, and local* NameCert *rows. It never compares host evaluation functions, quotient behaviour classes, infinite traces, or global termination predicates.*

Theorem 597.3 (Continuation termination NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationTermination}^{\uparrow}}$ *surface for Definition 597.1 consists exactly of carrier admission for T, start-history exposure through s, terminal-history exposure through t, finite* Cont *trace exposure through τ, terminal witness exposure through u, behaviour-classifier exposure through b, componentwise* hsame *transport through h,* Pkg *provenance through p, and local naming through n.*

Lean4 ✓ `BEDC.Derived.ContinuationTerminationUp.ContinuationTerminationCarrier namecert obligations`

Proof. Carrier admission is projection from the displayed packet T. The rows $s, t, \tau, u, b, h, p, n$ list the start history, terminal history, finite continuation trace, terminal witness, behaviour-classifier row, transports, provenance, and local name.

The terminal witness is finite because it is attached to the displayed terminal row t and the displayed trace τ. It records the boundary exported by this packet, not a host-level proof that arbitrary evaluation must halt.

The behaviour row is likewise local. It compares only the rows named by the packet and is transported by the stored hsame data. Since no coordinate contains a host evaluator, quotient behaviour class, unbounded trace, or classical termination decision, the obligation surface is exactly the listed rows. □

Theorem 597.4 (Continuation termination behaviour transport)**.** *If two accepted* $\mathsf{ContinuationTermination}^{\uparrow}$ *packets are related by the classifier of Definition 597.2, then any behaviour-facing consumer of the first packet transports to the second packet only through the displayed start history, terminal history, finite* Cont *trace, terminal witness, behaviour-classifier row, componentwise* hsame *data,* Pkg *provenance, and local* NameCert *row.*

Proof. Unfold the classifier. Its comparison fields are precisely $s, t, \tau, u, b, h, p, n$ and their primed counterparts.

Use the componentwise hsame rows to move the start and terminal histories, each displayed trace row, the terminal witness, and the behaviour-classifier row. The provenance and local name rows move through their displayed package and certificate coordinates.

There is no remaining coordinate from which a consumer could read host evaluation, quotient behaviour equality, an infinite execution trace, or a termination oracle. Therefore behaviour transport is confined to the finite packet surface. □

Lean4 ✓ `BEDC.Derived.ContinuationTerminationUp.ContinuationTerminationCarrier behavior transport`

Theorem 597.5 (Continuation termination non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{ContinuationTermination}^{\uparrow}$ *packet factors through the finite rows*

$$s, t, \tau, u, b, h, p, n.$$

The consumer may read the start history, terminal history, finite Cont *trace, terminal witness, behaviour-classifier row,* hsame *transports,* Pkg *provenance, and local* NameCert *row, and it cannot read host evaluation, quotient behaviour classes, unbounded traces, or a classical termination decision.*

Lean4 ✓ `BEDC.Derived.ContinuationTerminationUp.ContinuationTerminationCarrier nonescape`

Proof. Project the accepted carrier. The displayed packet contains exactly the rows $s, t, \tau, u, b, h, p, n$.

Apply the local NameCert obligations to identify the exported consumer surface. The terminal witness and behaviour-classifier rows are named inside the same finite carrier and are transported only by the stored componentwise hsame data.

Use behaviour transport for classifier-facing consumers. It keeps every comparison on the visible trace and witness rows.

Since the carrier exports no host evaluator, quotient behaviour class, unbounded trace, or classical termination predicate, no downstream consumer can escape the finite continuation-termination packet. □

Theorem 597.6 (Continuation termination terminal route)**.** *For every accepted* ContinuationTermination$^{\uparrow}$ *packet*

$$T = (s, t, \tau, u, b, h, p, n),$$

the terminal witness row u is readable only together with the start history s, terminal history t, finite Cont *trace τ, behaviour row b, componentwise* hsame *transport h,* Pkg *provenance p, and local* NameCert *row n. A terminal-facing consumer therefore receives a displayed finite route through these rows, and never a host total evaluator, quotient behaviour class, unbounded trace, or classical termination oracle.*

Proof. Project the carrier of Definition 597.1. The terminal witness u is one coordinate of the displayed packet T, not an isolated endpoint predicate.

Use the NameCert obligations of Definition 597.3. They expose u only after the start row s, terminal row t, trace row τ, behaviour row b, transport row h, provenance row p, and local name row n remain visible in the same packet.

Since no carrier coordinate names host evaluation, quotient behaviour equality, an unbounded trace, or a global termination decision, reading u cannot escape the finite terminal route. □

Theorem 597.7 (Continuation termination classifier stability)**.** *If two accepted* ContinuationTermination$^{\uparrow}$ *packets are related by the classifier of Definition 597.2, then every classifier-facing read transports terminal and behaviour data componentwise through the displayed rows*

$$s, t, \tau, u, b, h, p, n.$$

In particular, terminal witnesses and behaviour rows remain attached to the same finite start, terminal, trace, transport, provenance, and local naming surface.

Proof. Unfold the classifier of Definition 597.2. Its comparison fields are exactly the start histories, terminal histories, finite traces, terminal witnesses, behaviour rows, componentwise transports, provenance rows, and local names of the two packets.

Apply the behaviour transport theorem to move any classifier-facing consumer along the displayed hsame coordinates. The terminal witness moves with u, and the behaviour row moves with b, while s, t, τ, h, p, n keep the route tied to the same finite packet surface.

No hidden evaluator, quotient behaviour class, infinite trace, or termination oracle remains outside those coordinates. Hence classifier-related packets preserve terminal and behaviour reads componentwise through the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationTermination}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuationTermination}^{\uparrow}$ packet is generated from finite BHist rows for an initial state, displayed continuation trace, terminal witness, behavior classifier, and consumer-facing route ledger. Termination is not read as a global evaluator predicate; it is the existence of the shown finite trace whose endpoint is routed by Cont to the terminal row. Classifier-facing consumers use hsame to transport the visible trace, witness, and behavior rows componentwise, while Pkg records the continuation and naming dependencies. The packet therefore supplies a bounded continuation-termination certificate and refuses unbounded behavior classes or host evaluation semantics.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationTermination}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the continuation-termination carrier, behavior transport, terminal route obligations, and finite packet non-escape statement for $\mathsf{ContinuationTermination}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuationTermination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim a total evaluator, quotient behavior class, unbounded trace semantics, selected divergence evidence, or a global termination decision procedure.

Upgrade path: Obligation closure requires Lean encodings of the finite trace carrier, terminal witness route, behavior transport, and non-escape theorem.

Chapter 598

A Concrete Naming Certificate for $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$

The $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ surface records a type-checking run as a finite BEDC packet. It refines the $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{ClassifierTyping}^{\uparrow}$, and $\mathsf{SubjectReduction}^{\uparrow}$ views by making the derivation ledger, reduction route, classifier membership, transport, provenance, and local naming row visible in one certificate.

The carrier reads its classifier row through chapter 476, its typing ledger through chapter 489, and its reduction-preservation row through chapter 483; this chapter packages those sibling surfaces into a single derivation trace rather than replacing them.

Definition 598.1 (Type-checking derivation trace carrier)**.** *A* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, E, C, D, H, R, P, N)$$

with the following displayed rows:

1. *T is the term row, recording the source term, target term, typing context entries, and the finite term-history positions inspected by the checker;*
2. *E is the* Ext*-membership classifier row, recording the accepted typing judgments for the source row and the target row;*
3. *C is the* Cont *reduction route, displaying each checked step from the source term row to the target term row;*
4. *D is the derivation ledger, listing the typing-rule entries, premise addresses, and rule conclusions consumed by the checker;*
5. *H records componentwise* hsame *transports for the term row, classifier row, continuation route, derivation ledger, subject-reduction row, package provenance, and local naming row;*
6. *R is the subject-reduction row, recording that every displayed* Cont *step whose source is in E has its target in the transported* Ext *row;*
7. *P is the* Pkg *provenance over* BHist*, term rows,* Ext*,* Cont*, derivation ledgers,* hsame*, subject-reduction rows, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingDerivationTrace}^{\uparrow}}$ *row.*

The packet is finite and derivation-facing. It contains no ambient metatheory, no host equality of terms, no unrecorded typing rule, no hidden reduction step, and no proof object outside the displayed BEDC rows.

Lean4 ✓ `BEDC.Derived.TypeCheckingDerivationTraceUp.TypeCheckingDerivationTraceTasteGate single carrier alignment`

Theorem 598.2 (Type-checking derivation trace subject-reduction surface)**.** *Every accepted* TypeCheckingDerivatio *carrier routes the typed source row and the typed target row through the displayed* Ext *and* Cont *data. More precisely, if a consumer reads a source typing judgment from E and a reduction step from C, then the only available target typing judgment is the one recorded by the subject-reduction row R, transported through H and packaged by P, N.*

Thus a trace consumer may inspect T, E, C, D, H, R, P, N, but it cannot replace the derivation ledger by an unrecorded rule, replace the reduction route by a host computation, or identify source and target terms by host equality.

Proof. Project the accepted packet $Z = (T, E, C, D, H, R, P, N)$. The source and target term entries are carried by T, while the typing judgments available to the checker are carried by E.

Next read the continuation route C. Each reduction step consumed by a trace is displayed before any target typing judgment is exposed, so the route cannot be supplied by an external host computation.

Finally use the subject-reduction row R together with H, P, N. The transport rows replay the same classifier and continuation data under componentwise hsame, and the package and local naming rows bind the target typing judgment to the displayed trace. Since no other typing-rule or equality coordinate is present, the subject-reduction surface is exhausted by those rows. □

Theorem 598.3 (Type-checking derivation trace NameCert obligations)**.** *For the carrier of Definition 598.1, the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingDerivationTrace}\uparrow}$ *surface consists of six finite obligations:*

1. *carrier admission for the packet $Z = (T, E, C, D, H, R, P, N)$;*
2. Ext *membership for the displayed source and target typing rows;*
3. *subject reduction along the displayed* Cont *route;*
4. *ledger exactness for every typing-rule premise and conclusion consumed by D;*
5. *classifier transport through the componentwise* hsame *rows in H;*
6. *non-escape from the host type theory, excluding unrecorded typing rules, host equality of terms, hidden reduction steps, external proof objects, and ambient metatheoretic closure.*

Proof. Carrier admission is the projection of the finite tuple in Definition 598.1. The rows T, E, C, D, H, R, P, N are the only public coordinates named by the local certificate.

The Ext membership and subject-reduction obligations are read together. The classifier row E records the accepted typing judgments, the continuation route C records the checked reduction path, and Definition 598.2 identifies the target judgment through R.

Ledger exactness is supplied by D. Each typing-rule premise and conclusion consumed by the checker is addressed inside that row, so a consumer cannot introduce an unlisted rule application.

The classifier transport and non-escape clauses then follow from H, P, N and the carrier inventory. Componentwise hsame rows transport only the displayed data, while the missing coordinates rule out host equality, hidden reductions, external proof objects, and ambient metatheory. □

Theorem 598.4 (Type-checking derivation trace classifier membership readback)**.** *Every accepted* TypeCheckingDerivationTrace$^{\uparrow}$ *carrier exposes the* TypeClassifierMembership$^{\uparrow}$ *and* ClassifierTyping$^{\uparrow}$ *reads only through the displayed classifier row E, the derivation ledger D, the componentwise* hsame *transport H, the* Pkg *row P, and the local* NameCert *row N. A consumer may read classifier membership for the source and target typing rows, but it cannot replace that membership by a host typing judgment or an unrecorded classifier.*

Proof. Project the carrier $Z = (T, E, C, D, H, R, P, N)$. The only classifier coordinate is E, and the only derivation account that justifies the reads of E is the displayed ledger D.

Use the subject-reduction surface next. It already routes the typed source row and typed target row through Ext and Cont data, so classifier membership is read inside the same finite packet rather than through an ambient typing oracle.

The NameCert obligations then bind the readback to H, P, N. Componentwise hsame preserves the same classifier row, package provenance records the same BHist source, and local naming keeps the consumer at the TypeCheckingDerivationTrace$^{\uparrow}$ certificate boundary. □

Theorem 598.5 (Type-checking derivation trace classifier route)**.** *Every accepted classifier-facing read of a* TypeCheckingDerivationTrace$^{\uparrow}$ *packet factors through the displayed* Ext *membership row E, derivation ledger D, componentwise* hsame *transports H, package provenance P, and local* NameCert *row N. It cannot be supplied by host type equality, hidden reductions, external proof objects, or a normalizer row.*

Proof. Project the carrier $Z = (T, E, C, D, H, R, P, N)$. The classifier-facing coordinate is E, and the ledger that accounts for its typing-rule reads is D.

Apply Definition 598.4. It already routes the TypeClassifierMembership$^{\uparrow}$ and ClassifierTyping$^{\uparrow}$ reads through E, D, H, P, N and rules out host typing judgments and unrecorded classifiers.

For the non-escape clause, invoke Definition 598.7. That boundary states that the accepted packet contains no hidden normalizer, host term equality, unrecorded reduction step, or external proof object, so the classifier route remains the displayed finite row E, D, H, P, N. □

Theorem 598.6 (Type-checking derivation trace subject-reduction transport)**.** *For every accepted* TypeCheckingDerivationTrace$^{\uparrow}$ *carrier, the* SubjectReduction$^{\uparrow}$ *row R is transported only along the displayed* Cont *route C and the componentwise* hsame *row H. The transported target typing judgment is therefore the one packaged by P, N from the source classifier row E and derivation ledger D.*

Lean4 ✓ `BEDC.Derived.TypeCheckingDerivationTraceUp.TypeCheckingDerivationTracePacket subject reduction transport`

Proof. Start from the subject-reduction surface. It states that a consumer reading a source typing judgment from E and a reduction step from C reaches the target typing judgment through R.

The carrier makes C and H the only continuation and transport rows. Thus subject reduction can be replayed only over the displayed Cont path and only under componentwise hsame for the same term row, classifier row, derivation ledger, and provenance data.

Finally apply the NameCert obligations. They require subject reduction, ledger exactness, classifier transport, and non-escape in one local package, so the transported target judgment is the one recorded by R and cited by P, N, not an external proof object or a host computation. □

Theorem 598.7 (Type-checking derivation trace normal-form consumer boundary)**.** *Every normal-form-facing consumer of an accepted* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *packet receives only the finite trace*

$$T, E, C, D, H, R, P, N.$$

The consumer may use the $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{TypeClassifierMembership}^{\uparrow}$*, and* $\mathsf{SubjectReduction}^{\uparrow}$ *rows recorded in that packet, but it receives no hidden normalizer, no host equality of terms, no unrecorded reduction step, and no proof object outside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data.*

Proof. Read classifier membership through Definition 598.4. This fixes the classifier-facing rows as E, D, H, P, N inside the same accepted carrier.

Read subject reduction through Definition 598.6. This fixes the reduction-facing rows as C, H, R, P, N and prevents the target typing judgment from being supplied by host computation.

It remains to inspect the carrier inventory. The tuple T, E, C, D, H, R, P, N contains no normalizer row, host equality row, hidden reduction coordinate, or external proof-object coordinate. Therefore a normal-form-facing consumer can only consume the finite trace and its displayed typing and reduction certificates. □

Theorem 598.8 (Type-checking derivation trace rule-address totality)**.** *For every accepted* $\mathsf{TypeCheckingDerivation}$ *carrier* $Z = (T, E, C, D, H, R, P, N)$*, each typing-rule premise and each rule conclusion consumed by the checker has a displayed address in the finite derivation ledger* D*. No rule premise, conclusion, or typing judgment may be supplied by an unrecorded host checker row.*

Proof. Start with the carrier inventory in Definition 598.1. The row D is defined as the derivation ledger listing the typing-rule entries, premise addresses, and rule conclusions consumed by the checker.

Apply the NameCert obligations. Their ledger-exactness clause requires every consumed premise and conclusion to be addressed inside D, while the non-escape clause excludes unrecorded typing rules and host metatheory.

Finally use classifier membership readback. The classifier rows E can be read only with the derivation account D, H, P, N, so every rule-address read remains inside the finite ledger. □

Theorem 598.9 (Type-checking derivation trace Ext-Cont route coherence)**.** *Every subject-reduction read of an accepted* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *packet must pass through both the* Ext *membership row* E *and the* Cont *reduction route* C *before reaching the subject-reduction row* R*. The row* R *therefore reads a target typing boundary only from the displayed classifier and reduction route, transported through* H *and packaged by* P, N*.*

Proof. Begin with the subject-reduction surface. It says that a consumer reading a source typing judgment from E and a reduction step from C reaches the target typing judgment only through R.

Next apply subject-reduction transport. The row R is transported only along C and H, with package and local naming supplied by P, N.

Use classifier membership readback to bind E to the same packet. Since the classifier-facing read passes through E, D, H, P, N, the subject-reduction route cannot bypass E or replace C by host computation before reading R. □

Theorem 598.10 (Type-checking derivation trace normalizer refusal)**.** *No consumer may replace the displayed* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *packet*

$$T, E, C, D, H, R, P, N$$

by a hidden normalizer, host term equality, external proof object, or unrecorded reduction engine. Normal-form, classifier, and subject-reduction reads must remain inside those eight rows and the BEDC BHist*,* Ext*,* Cont*,* hsame*,* Pkg*, and local* $\mathsf{NameCert}$ *surfaces named by the carrier.*

Proof. First invoke the normal-form consumer boundary. It exposes only T, E, C, D, H, R, P, N to normal-form-facing consumers and rules out hidden normalizers, host equality of terms, unrecorded reduction steps, and external proof objects.

For classifier-facing reads, use Definition 598.8 together with classifier membership readback. The rule addresses and classifier membership are read through D, E, H, P, N, not through a host typing oracle.

For subject-reduction-facing reads, use Definition 598.9. The route reaches R only after E and C have been displayed, so a normalizer cannot replace the packet. □

Theorem 598.11 (TypeCheckingDerivationTrace membership-ledger exactness)**.** *For every accepted* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *carrier* $Z = (T, E, C, D, H, R, P, N)$*, every* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *and* $\mathsf{ClassifierTyping}^{\uparrow}$ *read factors through the displayed classifier row* E*, derivation ledger* D*, componentwise transports* H*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. No membership or classifier-typing consumer may replace* E *by a host typing judgment, replace* D *by an unrecorded rule ledger, or bypass the local packet route.*

Proof. Project the carrier and isolate the classifier-facing rows. The only displayed membership coordinate is E, and the only displayed account of rule use is the derivation ledger D.

Apply Definition 598.4. It already routes $\mathsf{TypeClassifierMembership}^{\uparrow}$ and $\mathsf{ClassifierTyping}^{\uparrow}$ reads through E, D, H, P, N.

Finally use Definition 598.7. That boundary excludes hidden normalizers, host equality, unrecorded reduction steps, and external proof objects, so the membership ledger is exact for the finite packet. □

Theorem 598.12 (TypeCheckingDerivationTrace subject-reduction route coverage)**.** *Every* $\mathsf{SubjectReduction}^{\uparrow}$ *consumer of an accepted* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ *carrier reads the source row, target row, and reduction route from the finite packet* $Z = (T, E, C, D, H, R, P, N)$*. The source and target term entries are read from* T*, classifier membership is read from* E*, the reduction route is read from* C*, and the target typing boundary is read through* R, H, P, N *rather than through host computation or an unrecorded reduction path.*

Proof. Start with the subject-reduction surface. It identifies the source and target typing judgments by reading T and E, then follows the displayed continuation route C.

Use Definition 598.6. The subject-reduction row R is transported only along C and H, with package and local naming supplied by P, N.

The normal-form consumer boundary of Definition 598.7 rules out a hidden normalizer or unrecorded reduction step. Thus every $\mathsf{SubjectReduction}^{\uparrow}$ consumer is covered by the finite packet rows. □

Theorem 598.13 (TypeCheckingDerivationTrace host normalizer non-escape)**.** *No accepted* TypeCheckingDerivatio *carrier exports a hidden normalizer, host equality of terms, or an unrecorded reduction step. Every normal-form-facing, classifier-facing, or subject-reduction-facing consumer remains inside the*

displayed rows T, E, C, D, H, R, P, N *and the* BHist, hsame, Cont, Pkg, *and local* NameCert *data named there.*

Lean4 ✓ `BEDC.Derived.TypeCheckingDerivationTraceUp.TypeCheckingDerivationTracePacket host normalizer nonescape`

Proof. Inspect the carrier inventory in Definition 598.1. The tuple T, E, C, D, H, R, P, N contains term rows, classifier rows, continuation routes, derivation ledgers, transports, subject-reduction rows, provenance, and local naming, but no host normalizer or host equality coordinate.

For classifier-facing consumers, apply Definition 598.11. For subject-reduction-facing consumers, apply Definition 598.12.

The remaining normal-form-facing case is exactly Definition 598.7. Together these boundaries keep every public read inside the finite trace packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingDerivationTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$ packet is generated from term rows, Ext-membership classifier rows, Cont reduction routes, derivation ledgers, componentwise hsame transports, subject-reduction rows, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingDerivationTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 598.1, Definition 598.2, Definition 598.3, Definition 598.4, Definition 598.6, and Definition 598.13 bind the exported public packet boundary for the carrier, classifier membership, subject-reduction route, derivation ledger, transport, provenance, local naming, and host-normalizer non-escape rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeCheckingDerivationTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.TypeCheckingDerivationTraceUp.TypeCheckingDerivationTracePacket\_host\_normalizer\_none`

Bridge status: none

Origin: human

Not claimed: This chapter does not close decidability of arbitrary host type checking, host term equality, hidden reductions, external proof objects, or a public standard bridge.

Upgrade path: No further paper-side theory-axis upgrade remains for the host-normalizer non-escape trace packet.

Chapter 599

A Concrete Naming Certificate for $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$

The $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$ packet gives the unary-direction bridge a concrete finite standard-boundary object. It records how a displayed $\mathsf{Nat}^{\uparrow}$ unary spine is read beside a displayed $\mathsf{AxisNat}^{\uparrow}$ zero-spine packet, with the normalizing boundary carried by BHist, hsame, Cont, Pkg, and NameCert rows rather than by an identification of the two carriers.

Its composite dependencies are the $\mathsf{Nat}^{\uparrow}$ unary-spine row of chapter 28, the $\mathsf{AxisNat}^{\uparrow}$ zero-spine row of chapter 276, the unary-zero-spine standard comparison of chapter 572, and the unary-direction bridge boundary, assembled as one finite normalizer packet.

The packet consumes the sibling standard-boundary comparison of chapter 572 and the zero-spine carrier of chapter 276 before its normalizing boundary row is read.

Definition 599.1 (AxisUnarySpineNormalizer carrier)**.** *An* $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, A, L, Z, H, C, P, R)$$

with the following visible coordinates:

1. *N is the source* $\mathsf{Nat}^{\uparrow}$ *unary-spine row, read as a finite* E1 *history together with its local naming certificate;*
2. *A is the target* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine row, read as a finite* E0 *history together with its local naming certificate;*
3. *L is the displayed length ledger comparing the two histories by their finite step counts;*
4. *Z is the standard-boundary row asserting that the unary read and the zero-spine read have the same displayed length while remaining different BEDC carrier predicates;*
5. *H stores componentwise* hsame *transports for the unary row, zero-spine row, length ledger, standard-boundary row, and local name;*
6. *C stores the displayed* Cont *routes that read the unary spine, read the zero spine, expose the length ledger, and hand the boundary row to downstream bridge consumers;*
7. *P records* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
8. *R is the local* $\mathsf{NameCert}_{\mathsf{AxisUnarySpineNormalizer}^{\uparrow}}$ *row.*

The carrier has no coordinate that identifies $\mathsf{Nat}^{\uparrow}$ *with* $\mathsf{AxisNat}^{\uparrow}$*, rewrites* E1 *constructors into* E0 *constructors inside the kernel, or imports a host arithmetic normalizer.*

Lean4 (definition) `BEDC.Derived.AxisUnarySpineNormalizerUp.AxisUnarySpineNormalizerCarrier`

Theorem 599.2 (AxisUnarySpineNormalizer NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisUnarySpineNormalizer}^{\uparrow}}$ *surface for Definition 599.1 consists exactly of source unary-spine admission, target zero-spine admission, displayed length-ledger readback, standard-boundary exposure, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row. These obligations assemble a finite bridge-facing packet over* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *without equating the unary and axis carriers.*

Lean4 ✓ `BEDC.Derived.AxisUnarySpineNormalizerUp.AxisUnarySpineNormalizerNameCertObligations`

Proof. Carrier admission is projection from the displayed packet U. The rows N and A give the two named source surfaces, while L and Z give the finite comparison and the standard-boundary read.

The classifier is componentwise. The stored hsame row H transports the unary read, zero-spine read, length ledger, boundary row, package row, and local name without replacing either carrier by the other.

The continuation and provenance clauses are the remaining finite obligations. The routes in C expose the two spine reads and then pass the length ledger into Z, while P records that every row belongs to the listed BEDC dependencies. No host normalizer or carrier identity has a coordinate in the packet. □

Theorem 599.3 (AxisUnarySpineNormalizer standard boundary)**.** *Every accepted* $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$ *carrier exposes a length-preserving standard-boundary row from its* $\mathsf{Nat}^{\uparrow}$ *unary spine to its* $\mathsf{AxisNat}^{\uparrow}$ *zero spine. Downstream uses of the unary-direction bridge may read this row through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates, but they may not treat the source and target carrier predicates as equal inside the BEDC kernel.*

Lean4 ✓ `BEDC.Derived.AxisUnarySpineNormalizerUp.AxisUnarySpineNormalizerTasteGate single carrier alignment`

Proof. Start with an accepted carrier $U = (N, A, L, Z, H, C, P, R)$. By Definition 599.2, every consumer-facing read of U factors through the displayed rows of Definition 599.1.

The standard-boundary coordinate is Z. Its input data are the unary row N, the zero-spine row A, and the length ledger L, and its public route is one of the continuation rows in C.

The non-identification boundary is part of the same finite packet. Since neither N nor A is replaced by a shared carrier coordinate, the bridge exports a length-preserving standard boundary while preserving the kernel distinction between the two spine predicates. □

Theorem 599.4 (AxisUnarySpineNormalizer scoped source binding)**.** *The* $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$ *carrier of Definition 599.1, the NameCert obligation theorem of Definition 599.2, and the standard-boundary theorem of Definition 599.3 are scoped exactly over the* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{AxisNat}^{\uparrow}$*,* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *source surfaces. The checked TasteGate marker*

`BEDC.Derived.AxisUnarySpineNormalizerUp.AxisUnarySpineNormalizerTasteGate_single_carrier_align`

is used only as the verified singleton-carrier alignment point for the standard-boundary read; it does not add a further public object, host arithmetic normalizer, or carrier equality between $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$*.*

Proof. The carrier row lists exactly the source unary-spine row, target zero-spine row, displayed length ledger, standard-boundary row, componentwise transport, continuation routes, provenance row, and local certificate. These are the named $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert dependencies.

The NameCert obligation theorem reads the same coordinates as finite obligations, and the standard-boundary theorem exposes the boundary through the checked TasteGate marker already attached to that theorem. Thus the chapter binds the dependency scope of the normalizer without claiming a public isomorphism or a kernel identification of the two spine carriers. □

Lean4 ✓ `BEDC.Derived.AxisUnarySpineNormalizerUp.AxisUnarySpineNormalizerScopeBinding`

Closure status for $\mathsf{NameCert}_{\mathsf{AxisUnarySpineNormalizer}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$ packet is generated from a $\mathsf{Nat}^{\uparrow}$ unary-spine row, an $\mathsf{AxisNat}^{\uparrow}$ zero-spine row, a displayed length ledger, a standard-boundary row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row. It is composite over the $\mathsf{Nat}^{\uparrow}$ unary-spine row, $\mathsf{AxisNat}^{\uparrow}$ zero-spine row, length ledger, and standard-boundary surfaces, bundling those existing boundaries into one normalizer surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisUnarySpineNormalizer}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 599.1, Definition 599.2, Definition 599.3, and Definition 599.4 bind the carrier, NameCert obligations, standard-boundary theorem, and exact dependency scopes for the finite unary-to-zero-spine normalizer surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisUnarySpineNormalizer}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean formalization, bridge checking, kernel equality between $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$, host arithmetic normalization, or a public additive isomorphism.

Upgrade path: Public closure requires a public-object surface beyond the scoped normalizer packet and its singleton-carrier TasteGate alignment.

Chapter 600

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$

The $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ surface records the finite handoff that carries a regular Cauchy diagonal-limit classifier from the displayed rational source rows to the $\mathsf{Real}^{\uparrow}$ seal row. It sits between the classifier packet of chapter 443 and the completion-facing seal discipline of chapter 450.

Definition 600.1 (Regular Cauchy limit transport carrier)**.** *A* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, W, D, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, recording the regular rational sequence family, the selected source indices, and the diagonal rational observations read by the limit classifier;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *selected-window row, recording the finite windows through which the diagonal observations are named;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and absolute-value ledger, recording the requested tolerance, comparison rows, absolute-value rows, and triangle-bound entries used by the transport;*
4. *S is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, recording the completion-facing seal reached from the displayed diagonal observations;*
5. *H records componentwise* hsame *transports for the source row, selected windows, dyadic ledger, seal row, continuation routes, package provenance, and local naming row;*
6. *C records the displayed* Cont *routes through which the source rows, selected-window rows, dyadic observations, and seal handoff are read by downstream consumers;*
7. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTransport}^{\uparrow}}$ *row.*

The packet is finite and transport-facing. It contains no quotient stream, no host real equality, no selected external limit, no trichotomy row, and no $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Definition 600.2 (Regular Cauchy limit transport classifier)**.** *The classifier of a carrier $T = (R, W, D, S, H, C, P, N)$ compares only the displayed rows R, W, D, S, H, C, P, N. Two carriers are classified together exactly when the componentwise* hsame *rows in H identify the* $\mathsf{RegSeqRat}^{\uparrow}$ *source*

row, the StreamName$^{\uparrow}$ *selected-window row, the* DyadicRatCore$^{\uparrow}$ *tolerance and absolute-value ledger, the* Real$^{\uparrow}$ *seal handoff, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

The classifier has no coordinate for quotient streams, host real equality, selected external limits, trichotomy of reals, or Real$^{\uparrow}$ *completeness. A consumer asking for any of those objects leaves the packet boundary.*

Lean4 (definition) `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportClassifier`

Theorem 600.3 (Regular Cauchy limit transport stability)**.** *Let $T = (R, W, D, S, H, C, P, N)$ be an accepted* RegularCauchyLimitTransport$^{\uparrow}$ *carrier. Any componentwise displayed* hsame *transport of the rows R, W, D, S, H, C, P, N that preserves the continuation routes in C, the package provenance in P, and the local naming row N yields another accepted* RegularCauchyLimitTransport$^{\uparrow}$ *carrier with the same diagonal-limit classifier surface and the same* Real$^{\uparrow}$ *seal handoff.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier stability`

Proof. Project the accepted packet through Definition 600.1. The only rows available to the transport are the regular rational source row, selected windows, dyadic tolerance and absolute-value ledger, seal handoff, componentwise transport row, continuation routes, package provenance, and local naming row.

The classifier of Definition 600.2 compares exactly those rows. Transporting them componentwise through H preserves the source observations, selected windows, tolerance ledger, absolute-value comparisons, and seal-facing handoff read by the same continuation routes.

Since no external limit, quotient stream, host equality, trichotomy row, or completeness theorem appears in the carrier, there is no additional coordinate that can change the classifier surface. The transported packet is therefore another accepted carrier with the same finite diagonal-limit handoff. □

Theorem 600.4 (Regular Cauchy limit transport ledger exactness)**.** *Every* Real$^{\uparrow}$ *consumer of an accepted* RegularCauchyLimitTransport$^{\uparrow}$ *packet reads the transported diagonal-limit handoff through the displayed ledger*

$$R, W, D, S, H, C, P, N.$$

The tolerance request, absolute-value rows, triangle-bound entries, selected windows, regular rational source observations, and seal route are exactly the rows displayed by the packet. No consumer can replace them by a quotient-stream comparison, a host real equality proof, an externally selected limit, a trichotomy argument, or a Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier ledger exactness`

Proof. The carrier inventory gives the full ledger. Rows R and W name the regular rational source observations and their selected finite windows, while D records the dyadic tolerance, absolute-value, and triangle-bound data used to justify the transported read.

The seal handoff is row S, and every route to it is recorded in C and packaged by P, N. The componentwise transport row H can replay only the displayed rows of the same packet.

Thus the consumer surface is exhausted by R, W, D, S, H, C, P, N. The excluded objects have no coordinate in the carrier or classifier, so they cannot be introduced by a Real$^{\uparrow}$ consumer of this transport packet. □

Theorem 600.5 (Regular Cauchy limit transport NameCert obligations)**.** *For the carrier of Definition 600.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTransport}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for* $T = (R, W, D, S, H, C, P, N)$*;*
2. *classifier stability under the componentwise* hsame *rows displayed in* H*;*
3. *selected-window exactness for the* $\mathsf{StreamName}^{\uparrow}$ *rows in* W *and their reads from the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*;*
4. *dyadic ledger exactness for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, absolute-value, comparison, and triangle-bound rows in* D*;*
5. *seal handoff non-escape through* S, C, P, N*, excluding quotient streams, host real equality, selected external limits, trichotomy, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier namecert obligations`

Proof. Carrier admission is the finite packet of Definition 600.1. The row list R, W, D, S, H, C, P, N is the complete public inventory.

Classifier stability is supplied by Definition 600.3. The selected-window exactness clause is the restriction that every diagonal observation read from R reaches the classifier through the displayed finite window row W and the same continuation routes C.

Dyadic ledger exactness is the row D: it records the tolerance, absolute-value, comparison, and triangle-bound entries consumed by the transported handoff. Finally Definition 600.4 identifies the seal-facing non-escape boundary through S, C, P, N. □

Theorem 600.6 (Regular Cauchy limit transport selected-window exactness)**.** *Let* $T = (R, W, D, S, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier. Every diagonal observation read from* T *reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R *only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *selected-window row* W*, is transported by the componentwise rows in* H*, and is replayed to downstream consumers through the continuation routes in* C*. No external window selector or undisplayed source-row read can replace* W*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier selected window exactness`

Proof. Project the accepted carrier through Definition 600.1. The only source-facing rows in the packet are the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{StreamName}^{\uparrow}$ selected-window row W, with their transport and replay data recorded by H and C.

Use the classifier specification of Definition 600.2. It compares carriers by the displayed rows R, W, D, S, H, C, P, N and has no coordinate for an external selector.

Thus a diagonal observation can be read from the source row only by passing through the selected window already present in W. The componentwise transport rows in H and the continuation routes in C may replay that read, but they cannot introduce a second source-row access path. □

Theorem 600.7 (Regular Cauchy limit transport dyadic ledger stability)**.** *Let* $T = (R, W, D, S, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier. Under any componentwise* hsame *transport accepted by the rows in* H*, the tolerance request, absolute-value rows, comparison rows, and triangle-bound entries used by the limit handoff remain entries of the single* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*. The transport creates no second dyadic ledger and no independent arithmetic side condition.*

Proof. Start from the stability theorem Definition 600.3. It says that componentwise transport of the displayed rows preserves the accepted carrier and the same diagonal-limit classifier surface.

Now read the ledger exactness theorem Definition 600.4. The tolerance request, absolute-value rows, comparison rows, and triangle-bound entries consumed by a $\mathsf{Real}^{\uparrow}$ handoff are exactly the rows displayed by the packet.

Combining the two statements fixes the dyadic data inside D throughout the transport. Since the carrier and classifier expose no additional arithmetic coordinate, the transported handoff cannot select a separate dyadic ledger or attach a hidden side condition. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier dyadic ledger stability`

Theorem 600.8 (Regular Cauchy limit transport seal-route determinacy)**.** *For an accepted* RegularCauchyLimitTra *carrier* $T = (R, W, D, S, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal handoff exported by* T *factors through the finite route*

$$R \to W \to D \to S$$

with provenance recorded by H, C, P, N*. The route cannot be replaced by quotient streams, host real equality, selected external limits, trichotomy, or a* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First apply selected-window exactness, Definition 600.6. It forces each diagonal source read to pass from R through the displayed selected-window row W and then through the recorded transport and continuation rows.

Next apply dyadic ledger stability, Definition 600.7. The tolerance, absolute-value, comparison, and triangle-bound data used after the window read remain in the single ledger D under componentwise transport.

Finally use ledger exactness, Definition 600.4. The seal row S is reached only through the displayed packet rows, with provenance H, C, P, N. The excluded quotient, host-equality, external-limit, trichotomy, and completeness objects have no carrier coordinate, so they cannot determine a replacement route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier seal route determinacy`

Theorem 600.9 (Regular Cauchy limit transport RealCompleteness consumer factorization)**.** *Let* $T = (R, W, D, S, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier. Let* Q *be an accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *packet whose regular-family row, selected-window row, dyadic-ledger row, and seal row are exactly the displayed rows* R, W, D, S *of* T*, transported by the same rows* H, C, P, N*. Then every* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-seal consumer of* Q *factors through*

$$R, W, D, S, H, C, P, N$$

before reading the $\mathsf{Real}^{\uparrow}$ *seal. The factorization exposes no quotient stream, selected external limit, host real equality, trichotomy row, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier real completeness consumer factorization`

Proof. First apply selected-window exactness, Definition 600.6. It forces each diagonal observation to enter the source row R through the displayed selected-window row W and through the same transport and continuation rows H, C.

Next apply dyadic ledger stability, Definition 600.7. The tolerance, absolute-value, comparison, and triangle-bound entries used by the diagonal read remain entries of D, so the $\mathsf{RealCompleteness}^{\uparrow}$ packet cannot substitute a second arithmetic ledger.

Now use seal-route determinacy, Definition 600.8. The seal handoff reaches S only through the finite route $R \to W \to D \to S$ with provenance H, C, P, N.

Finally read the finite seal surface of $\mathsf{RealCompleteness}^{\uparrow}$. Because Q identifies its regular-family, window, dyadic-ledger, and seal rows with R, W, D, S, any diagonal-seal consumer of Q has already passed through the carrier rows of T before it reads the $\mathsf{Real}^{\uparrow}$ seal. The excluded quotient, selected-limit, host-equality, trichotomy, and ambient-completeness objects are absent from both displayed packet surfaces. □

Theorem 600.10 (Regular Cauchy limit transport obligation-closure package)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the displayed rows jointly supply the obligation-closure package for the finite handoff from $\mathsf{RegSeqRat}^{\uparrow}$ *through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *to the* $\mathsf{Real}^{\uparrow}$ *seal. The package consists of carrier admission for* T*, classifier stability over the rows* R, W, D, S, H, C, P, N*, selected-window exactness through* W*, dyadic ledger stability through* D*, seal-route determinacy through* S, C, P, N*, and non-escape for every consumer of the finite handoff.*

The package has no quotient stream coordinate, no host real equality coordinate, no selected external limit coordinate, no trichotomy coordinate, and no $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is exactly the packet of Definition 600.1, and the classifier boundary is exactly the row comparison of Definition 600.2.

Use Definition 600.3 for classifier stability and Definition 600.5 for the local obligation surface. These two rows bind the accepted carrier and the local naming boundary to the same displayed packet.

Next apply Definition 600.6 and Definition 600.7. They force the source read through the selected-window row W and keep the tolerance, absolute-value, comparison, and triangle-bound data inside the dyadic ledger D.

Finally use Definition 600.4 and Definition 600.8. Every seal-facing consumer is exhausted by R, W, D, S, H, C, P, N, so the excluded quotient, host-equality, external-limit, trichotomy, and completeness objects have no route into the package. □

Theorem 600.11 (Regular Cauchy limit transport window-dyadic-seal triangle)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the finite route

$$R \to W \to D \to S$$

commutes as a single handoff from the $\mathsf{RegSeqRat}^{\uparrow}$ *source row through the* $\mathsf{StreamName}^{\uparrow}$ *selected-window row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger into the* $\mathsf{Real}^{\uparrow}$ *seal. The commutation is read only through the displayed* hsame *rows in* H*, the* Cont *routes in* C*, the* Pkg *provenance in* P*, and the local* NameCert *row* N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier window dyadic seal triangle`

Proof. Begin with selected-window exactness, Definition 600.6. It sends every accepted diagonal observation from the source row R through the selected-window row W, with the replay route recorded by H and C.

Next use dyadic ledger stability, Definition 600.7. After the window read, the tolerance request, absolute-value rows, comparison rows, and triangle-bound entries remain inside the single dyadic ledger D.

Finally apply seal-route determinacy, Definition 600.8, together with ledger exactness, Definition 600.4. The seal row S is reached through the same displayed packet rows and no other coordinate is present. Hence the selected-window, dyadic-ledger, and seal routes form one commuting finite handoff. □

Theorem 600.12 (Regular Cauchy limit transport classifier handoff coverage)**.** *Every accepted classifier handoff of a* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier is covered by the displayed rows*

$$R, W, D, S, H, C, P, N$$

and by the triangle of Definition 600.11. The classifier has no quotient-stream coordinate, no host real equality coordinate, no selected external limit coordinate, no trichotomy coordinate, and no completeness coordinate.

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier classifier handoff coverage`

Proof. Read the classifier boundary from Definition 600.2. It compares exactly the rows R, W, D, S, H, C, P, N, and componentwise stability is supplied by Definition 600.3.

The triangle theorem then identifies the only multi-row handoff inside that classifier surface: source observations pass through W, ledger data stays in D, and the seal route reaches S with provenance H, C, P, N.

The excluded objects are absent from the carrier inventory of Definition 600.1. Since the classifier has no slot for those objects, an accepted handoff cannot classify by quotient streams, host real equality, selected external limits, trichotomy, or completeness data. □

Theorem 600.13 (Regular Cauchy limit transport completion boundary nonescape)**.** *Let* $T = (R, W, D, S, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier. Any* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer that reads this transport must factor through the triangle*

$$R \to W \to D \to S$$

before reaching the $\mathsf{Real}^{\uparrow}$ *seal. The consumer cannot bypass the triangle by selecting a quotient stream, a host real equality proof, an external limit, a trichotomy row, or an ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier completion boundary nonescape`

Proof. First use the RealCompleteness consumer factorization theorem, Definition 600.9. It says that a diagonal-seal consumer whose rows match R, W, D, S must read the displayed ledger R, W, D, S, H, C, P, N before it reaches the $\mathsf{Real}^{\uparrow}$ seal.

Then apply the window-dyadic-seal triangle, Definition 600.11. That theorem compresses the relevant displayed ledger into the finite route $R \to W \to D \to S$ with provenance carried by H, C, P, N.

The non-escape part follows from seal-route determinacy and ledger exactness. Both the transport packet and the matching $\mathsf{RealCompleteness}^{\uparrow}$ consumer surface lack coordinates for quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness, so no consumer route can bypass the same triangle. □

Theorem 600.14 (Regular Cauchy limit transport obligation-closure export)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the rows R, W, D, S, H, C, P, N *export the obligation-closure package for* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$*. The export consists exactly of carrier admission, classifier stability, selected-window exactness, dyadic ledger stability, seal-route determinacy,* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer factorization, classifier handoff coverage, and completion-boundary non-escape.*

Proof. Start with the obligation-closure package of Definition 600.10. It provides carrier admission, classifier stability, selected-window exactness, dyadic ledger stability, seal-route determinacy, and the finite exclusion boundary for the displayed rows R, W, D, S, H, C, P, N.

Next apply the window-dyadic-seal triangle of Definition 600.11. This identifies the single handoff route $R \to W \to D \to S$ and keeps its provenance inside H, C, P, N.

Finally combine classifier handoff coverage, Definition 600.12, with completion-boundary non-escape, Definition 600.13. The first theorem exhausts the classifier-facing handoff through the same rows, and the second forces any $\mathsf{RealCompleteness}^{\uparrow}$ consumer to factor through the triangle before reading the $\mathsf{Real}^{\uparrow}$ seal. These clauses are precisely the exported obligation-closure surface. □

Theorem 600.15 (Regular Cauchy limit transport scoped dependency boundary)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the packet scope is exactly the route through $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. This route is witnessed by selected-window exactness, dyadic-ledger stability, seal-route determinacy, and completion-boundary non-escape.*

Proof. Read the source row R as the $\mathsf{RegSeqRat}^{\uparrow}$ input and the selected-window row W as the $\mathsf{StreamName}^{\uparrow}$ observation route. The selected-window theorem fixes this first segment of the route.

Next read D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and absolute-value ledger. Dyadic-ledger stability keeps that ledger inside the same finite packet.

Finally follow S to the $\mathsf{Real}^{\uparrow}$ seal through the displayed H, C, P, N rows over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Seal-route determinacy and completion-boundary non-escape show that no further dependency is used. □

Theorem 600.16 (Regular Cauchy limit transport visible scope exhaustion)**.** *No consumer of an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier can add quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness outside the scoped route of Definition 600.15. Any consumer read must stay inside* R, W, D, S, H, C, P, N *and the named* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *dependencies.*

Proof. Apply the scoped dependency boundary to identify the complete visible route. It accounts for the source row, selected-window row, dyadic ledger, Real seal handoff, transport, continuation, provenance, and local naming.

Now use classifier handoff coverage and completion-boundary non-escape. Those theorems exclude quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness from the carrier and from the RealCompleteness-facing consumer route.

Because every consumer read factors through the same displayed rows, there is no place where an external stream quotient, host equality proof, selected limit, order trichotomy, or ambient completeness principle can enter. □

Theorem 600.17 (Regular Cauchy limit transport public export)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the public $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *export is exactly the finite handoff*

$$R \to W \to D \to S$$

with provenance H, C, P, N*. This export is the consumer-facing packet read by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RealCompleteness}^{\uparrow}$*, and it excludes quotient streams, host equality, selected external limits, trichotomy, and ambient completeness.*

Proof. First collect the route components. Selected-window exactness fixes the $R \to W$ read, dyadic ledger stability fixes the $W \to D$ ledger passage, and seal-route determinacy fixes the $D \to S$ seal handoff with provenance retained in H, C, P, N.

Next use classifier-handoff coverage to make the packet visible to every named consumer. The $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{RealCompleteness}^{\uparrow}$ consumer all read the same displayed rows.

Finally apply completion-boundary non-escape. Since the boundary theorem excludes quotient streams, host equality, selected external limits, trichotomy, and ambient completeness, the public export contains exactly the finite handoff and its displayed provenance, with no further carrier coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier public export`

Theorem 600.18 (Regular Cauchy limit transport bridge packet)**.** *For an accepted* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *carrier*

$$T = (R, W, D, S, H, C, P, N),$$

the public export packet

$$R \to W \to D \to S$$

with provenance H, C, P, N *is exactly the bridge packet consumed by* $\mathsf{RealCompleteness}^{\uparrow}$ *finite-witness routes. The bridge preserves* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance while excluding quotient streams, host equality, selected external limits, trichotomy, and ambient completeness.*

Proof. Start with public export, Definition 600.17. It identifies the consumer-facing packet as the finite handoff $R \to W \to D \to S$ with provenance H, C, P, N.

Apply the RealCompleteness consumer factorization theorem. A matching $\mathsf{RealCompleteness}^{\uparrow}$ diagonal-seal consumer must pass through the same rows R, W, D, S, H, C, P, N before reading the $\mathsf{Real}^{\uparrow}$ seal.

Finally use completion-boundary non-escape. The excluded quotient, host-equality, external-limit, trichotomy, and ambient-completeness objects have no coordinate in the transport packet or in the finite-witness consumer route. Therefore the public export is already the bridge packet read by the finite RealCompleteness route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTransport}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ selected-window row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and absolute-value ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTransport}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 600.1, Definition 600.2, Definition 600.3, Definition 600.4, Definition 600.5, Definition 600.6, Definition 600.7, Definition 600.8, Definition 600.9, Definition 600.10, Definition 600.11, Definition 600.12, Definition 600.13, Definition 600.14, Definition 600.15, Definition 600.16, Definition 600.17, and Definition 600.18 bind the finite transported diagonal-limit carrier, classifier, stability, ledger exactness, selected-window route, dyadic ledger stability, seal-route determinacy, RealCompleteness consumer factorization, obligation package, window-dyadic-seal triangle, classifier handoff coverage, completion boundary, obligation-closure export, scoped dependency boundary, visible scope exhaustion, public export, bridge packet, and local naming obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitTransport}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyLimitTransportUp.RegularCauchyLimitTransportCarrier\_public\_export`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, host real equality, selected external limits, trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, or a public standard bridge.

Upgrade path: The paper axis is mature; further movement is on the verification and bridge axes.

Chapter 601

A Concrete Naming Certificate for $\mathsf{FiniteCauchyApproximation}^{\uparrow}$

The $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ interface isolates the finite observation packet used when a regular rational source is read at one displayed precision before a $\mathsf{Real}^{\uparrow}$ consumer crosses the Cauchy-limit boundary. It records the selected threshold, stream window, dyadic observation ledger, error row, and handoff route as finite BHist data, so later completion arguments can cite a named approximation surface without importing a selected real value or an ambient completeness principle.

Definition 601.1 (Finite Cauchy approximation carrier)**.** *A* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, n, k, W, D, E, S, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose rational stream is read through an attached* $\mathsf{StreamName}^{\uparrow}$ *surface;*
2. *n is the unary precision row naming the requested finite tolerance level;*
3. *k is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row displayed for precision n;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected at and beyond the threshold row k;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the dyadic representatives read from W;*
6. *E is the explicit dyadic error-bound row used to compare the selected observations;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row that exposes the finite approximation to a real-facing consumer boundary;*
8. *H stores componentwise* hsame *transports for the source, precision row, threshold, window, dyadic ledger, error bound, handoff, provenance, and local name;*
9. *C stores the displayed* Cont *routes from R to the stream window, from n to k, from W to D and E, and from the finite packet to the handoff row S;*
10. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected real values, infinite tails, host real metrics, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 601.2 (Finite Cauchy approximation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ *surface for Definition 601.1 consists exactly of source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, unary precision exposure,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold exactness, finite* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation-ledger exposure, explicit error-bound exposure,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff routing, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed tuple F. The rows R, n, k, W, D, E, S provide the source, precision, threshold, window, dyadic ledger, error bound, and real-facing handoff.

The threshold obligation is read from the $\mathsf{CauchyModulus}^{\uparrow}$ surface attached to the regular rational source. The finite window obligation is read from the $\mathsf{StreamName}^{\uparrow}$ rows selected by that threshold, and the dyadic observation rows are exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ entries recorded in D.

The remaining obligations are routing and naming obligations. The rows H, C, P, N transport the listed coordinates, replay the finite reads, record the dependency packages, and name the approximation packet without adding quotient stream equality, a selected limit, or an infinite-tail coordinate. □

Theorem 601.3 (Finite Cauchy approximation window soundness)**.** *For an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *carrier, every dyadic observation exported by the finite approximation lies in the displayed window* W *selected by the threshold row* k*. The error row* E *is read through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and its tolerance is bounded by the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold exactness for the unary precision row* n*.*

Proof. Start from the accepted carrier and project the rows R, n, k, W, D, E. The $\mathsf{CauchyModulus}^{\uparrow}$ threshold row k supplies the finite bound attached to the precision row n.

Read the stream window through the displayed $\mathsf{StreamName}^{\uparrow}$ surface. Since W is the only window coordinate in the packet, every observation route into D passes through that finite window before any dyadic representative is visible.

The error row is then read from the same dyadic ledger. The nonnegative tolerance rows used by $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ bind the endpoint comparison inside D, while the threshold exactness of $\mathsf{CauchyModulus}^{\uparrow}$ supplies the displayed precision bound. No exported observation can bypass W, D, E or appeal to an infinite tail. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyApproximationUp.FiniteCauchyApproximationCarrier window soundness`

Theorem 601.4 (Finite Cauchy approximation threshold route coverage)**.** *Let* $F = (R, n, k, W, D, E, S, H, C, P, N)$ *be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Every visible dyadic observation exported by* F*, and every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff route from* F*, first reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, unary precision row* n*,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* k*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*. The error row* E *is bounded by the same threshold row* k*, and the route to* S *can read no selected real value, quotient stream equality, hidden limit selector, host metric, or completeness coordinate before those finite rows have been exposed.*

This threshold route is the finite handoff surface supplied by the sibling $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *chapters, chapter 292 and chapter 126.*

Proof. Project the accepted carrier to the displayed rows $R, n, k, W, D, E, S, H, C, P, N$. The source, precision, threshold, window, dyadic ledger, and error row are ordinary carrier coordinates, while the packet has no coordinate for a selected limit, quotient stream, host metric, or completeness witness.

Apply Definition 601.3. It places every visible dyadic observation inside the finite window W, reads that observation through the ledger D, and bounds E by the threshold row k attached to the precision row n.

Next apply Definition 601.7. Any $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer must reach S only after the route has read R, n, k, W, D, E together with the transport, continuation, provenance, and local naming rows.

Finally apply Definition 601.16. The closure packages those same finite rows and backing ledgers as the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ surface. Thus neither a visible dyadic observation nor a real-facing handoff can bypass the threshold route through the sibling $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ surfaces. □

Theorem 601.5 (Finite Cauchy approximation refinement stability)**.** *Let two accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *carriers have the same transported* $\mathsf{RegSeqRat}^{\uparrow}$ *source, unary precision row,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, finite* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic observation ledger, and explicit error row. Then their exported finite approximations present the same observation ledger and the same error-bound surface to every* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{CauchyModulusMeet}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer route.*

Proof. Project the two carriers and read the componentwise transport rows in H. The hypothesis identifies the source, precision, threshold, window, dyadic ledger, and error-bound rows after transport.

Use the displayed continuation routes to replay the finite reads in the same order: source to window, precision to threshold, window to dyadic observations, and dyadic observations to the error row. The shared-bound routes supplied by $\mathsf{CauchyModulusMeet}^{\uparrow}$ preserve the common threshold surface when a consumer compares this packet with another regular rational source.

The consumer-facing surface is therefore fixed before the handoff row is read. Any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ route receives the same D and E rows, together with the same transport, continuation, provenance, and local naming data. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyApproximationUp.FiniteCauchyApproximationCarrier refinement stability`

Theorem 601.6 (Finite Cauchy approximation precision weakening route)**.** *Let* $F = (R, n, k, W, D, E, S, H, C, P, N$ *be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose* n' *is a second unary precision row whose request is weaker than* n *through a displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *comparison, and suppose the* $\mathsf{StreamName}^{\uparrow}$ *window for* n' *is obtained by restricting or reusing the displayed window* W *while retaining the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer at* n' *factors through* $R, n, k, W, D, E, H, C, P, N$ *before reaching* S*.*

The weakening route exports only the displayed finite-window surface. It does not identify quotient streams, select a real value, supply an infinite-tail witness, introduce a host metric, or invoke $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet to the rows $R, n, k, W, D, E, S, H, C, P, N$. Apply Definition 601.4 at the original precision row n to obtain the displayed threshold route through the source, precision, threshold, finite window, dyadic ledger, and error row.

The comparison in $\mathsf{CauchyModulus}^{\uparrow}$ says that the request at n' asks for no finer information than the request already served at n. The $\mathsf{StreamName}^{\uparrow}$ hypothesis either reuses W or restricts it, so Definition 601.3 keeps every dyadic observation for the weakened request inside the same finite ledger D and the same error row E.

Transport the shared rows through H and the displayed continuation routes C. By Definition 601.5, replacing the consumer-side precision request by the weaker row while keeping the source, threshold, window, dyadic ledger, and error rows fixed preserves the exported finite surface.

It remains only to cross the real-facing boundary. Apply Definition 601.7: every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reaches S after the finite rows and bookkeeping rows have been exposed, and the packet has no coordinate for quotient stream equality, a selected real value, an infinite tail, a host metric, or a completeness principle. □

Theorem 601.7 (Finite Cauchy approximation Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *carrier factors through the displayed handoff row* S*. The handoff may read only the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, unary precision row,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, explicit error-bound row, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row supplied by the packet.*

Proof. Begin with a real-facing consumer route. The carrier contains a single handoff coordinate S, and that coordinate is explicitly the $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer boundary.

The route to S first reads the regular rational source, the requested precision, the displayed modulus threshold, and the finite stream window. It then reads the dyadic observation ledger and the explicit error row before any real-facing handoff is exposed.

Apply the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary discipline to the final handoff. The consumer receives a finite approximation packet with its tolerance ledger and provenance, not a selected real value, quotient stream equality, host real metric, or completeness witness. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyApproximationUp.FiniteCauchyApproximationCarrier real handoff`

Theorem 601.8 (Finite Cauchy approximation observation surface)**.** *The carrier, NameCert obligations, window soundness, refinement stability, and real handoff of Definition 601.1, Definition 601.2, Definition 601.3, Definition 601.5, and Definition 601.7 assemble one finite-window approximation certificate over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{CauchyModulusMeet}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

The certificate exports exactly the rows

$$R, n, k, W, D, E, S, H, C, P, N.$$

It exports no quotient stream equality, selected real value, infinite-tail witness, host real metric, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The carrier definition supplies the finite packet and lists every row available to the certificate. The NameCert obligations bind admission, precision, threshold, window, dyadic ledger, error bound, handoff, transport, continuation, provenance, and naming to those rows.

Window soundness identifies the finite observations and the displayed tolerance ledger. Refinement stability shows that transported packets with the same source, precision, threshold, window, dyadic ledger, and error row expose the same approximation surface to downstream consumers.

Finally the real handoff theorem identifies the only completion-facing boundary. Since the packet contains no quotient stream equality, selected real value, infinite-tail witness, host real

metric, or completeness coordinate, the listed rows form the complete finite-window approximation certificate. □

Theorem 601.9 (Finite Cauchy approximation tail-meet triangle assembly)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P,$ be an accepted* FiniteCauchyApproximation$^{\uparrow}$ *packet. Suppose the finite approximation window W is backed by an accepted* RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold packet, and suppose the dyadic error row E is backed by an accepted* RegularCauchyTriangleBound$^{\uparrow}$ *tolerance-addition ledger. Then the Real handoff row S receives exactly the displayed finite approximation rows together with the shared tail threshold, the triangle-bound tolerance ledger, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows.*

The assembly exports no selected real value, quotient stream equality, infinite tail, host metric, or completeness principle.

Proof. Start from Definition 601.7. It routes every real-facing consumer through the source row, precision row, modulus threshold, finite window, dyadic observation ledger, error row, transport, continuation, provenance, and local name before the handoff row S is read.

Use Definition 560.9 to supply the shared-threshold backing for W. The tail-meet packet contributes only the paired source rows, scheduled windows, source modulus rows, shared threshold ledger, paired comparison row, transports, continuations, ledger, provenance, and local name.

Use Definition 466.6 to supply the tolerance-addition backing for E. Its ledger identifies the three-leg dyadic estimate consumed by the finite approximation without adding an external completion or metric row.

The composed packet therefore presents exactly the listed finite approximation rows and the two backing ledgers. Since none of the cited surfaces contains a selected real value, quotient stream equality, infinite tail, host metric, or completeness coordinate, the Real handoff row S cannot export any such data. □

Theorem 601.10 (Finite Cauchy approximation tail-meet dependency route)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P$ be an accepted* FiniteCauchyApproximation$^{\uparrow}$ *packet. If F exports a handoff row through a shared finite window, then every consumer route from S to* RealDiagonalMeshBudget$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *factors through the displayed* RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold row before reading the finite approximation seal.*

The route exports no selected real value, quotient stream, hidden tail oracle, host metric, or completeness coordinate outside the finite rows of F and the shared-threshold row.

Proof. Start from the real handoff. By Definition 601.7, every completion-facing read of F first passes through the source row, precision row, threshold row, finite window, dyadic observation ledger, error row, transports, routes, provenance, and local name before the handoff row S is visible.

Use Definition 601.9 for the shared finite window. It supplies the RegularCauchyTailMeet$^{\uparrow}$ shared-threshold row as the backing route for W, so a consumer cannot read the finite window as an independent tail oracle.

Next use Definition 601.12. A RealDiagonalMeshBudget$^{\uparrow}$ consumer receives the finite approximation only through the mesh-budget route, and that route already includes the shared meet and shared-tail rows before the finite approximation seal is read.

For a CauchyLimitSeal$^{\uparrow}$ consumer, use the seal coverage theorem. The seal reads the same finite rows and the same tail-meet backing before reaching S. Therefore every downstream route factors through the displayed RegularCauchyTailMeet$^{\uparrow}$ shared-threshold row and cannot bypass

it by selecting a real value, quotient stream, hidden tail oracle, host metric, or completeness coordinate. □

Theorem 601.11 (Finite Cauchy approximation seal coverage)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of the handoff row S factors through the finite rows $R, n, k, W, D, E, H, C, P, N$. If the packet is backed by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet and an accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle-bound ledger, then the consumer receives only the shared tail threshold and triangle-bound ledger already named by those backing rows.*

The covered seal exports no selected real value, quotient stream, infinite tail, host metric, or completeness principle.

Proof. Begin with the Real handoff theorem. It routes every real-facing consumer through the displayed handoff row S, and it lists the finite source, precision, threshold, window, dyadic ledger, error row, transport, continuation, provenance, and naming rows that may be read before the handoff.

Next apply window soundness. The only visible observations are the entries of the finite window W, the dyadic ledger D, and the error row E controlled by the threshold k for the precision row n. Refinement stability then transports the same finite surface across accepted packets with the same displayed rows.

For backed packets, apply the tail-meet triangle assembly theorem. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ backing contributes the shared tail threshold, and the $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ backing contributes the triangle-bound tolerance ledger; both are already part of the named finite packet route.

Finally use the observation surface exclusion. Since the packet has no coordinate for selected real values, quotient streams, infinite tails, host metrics, or completeness, neither a $\mathsf{Real}^{\uparrow}$ consumer nor a $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can receive such data through S. □

Theorem 601.12 (Finite Cauchy approximation mesh-budget projection)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose F is backed by accepted* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, *and* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *rows. Then every* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *consumer of the handoff row factors through the displayed finite rows*

$$R, n, k, W, D, E, H, C, P, N,$$

the shared meet row, the triangle-bound tolerance ledger, and the selected diagonal-window coverage row.

The projection exports no quotient stream equality, selected real value, infinite tail, host metric, or completeness principle.

Proof. Start with Definition 601.11. It confines the handoff boundary to the finite source, precision, modulus threshold, window, dyadic ledger, error row, transport, continuation, provenance, and naming rows of F.

Apply Definition 462.27 to the shared modulus read. The meet row is the only common-threshold input supplied to a diagonal consumer, so it cannot add a quotient stream, a chosen real value, or an infinite tail.

Next apply Definition 466.11. The triangle-bound backing contributes the tolerance-addition ledger used by the finite error row E, and no host metric or completeness coordinate is introduced by that ledger.

Finally combine Definition 561.13 with Definition 618.6. The selected diagonal-window coverage row supplies the window surface consumed by the mesh-budget packet, and the mesh-budget handoff reads it only through its finite route before the $\mathsf{Real}^{\uparrow}$ seal is exposed. □

Theorem 601.13 (Finite Cauchy approximation cofinal tail-budget route)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. If F is consumed through a cofinal tail-budget route, then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff factors through the displayed rows*

$$R, n, k, W, D, E, S, H, C, P, N$$

and through the sibling $\mathsf{CofinalTailBudget}^{\uparrow}$ *row before the seal is read.*

The route exports no selected real value, quotient stream equality, infinite tail, host metric, or completeness principle.

Proof. Start with the finite approximation handoff. By Definition 601.7, every real-facing consumer reads the source, precision, threshold, window, dyadic ledger, error row, handoff row, transports, routes, provenance, and local name before the $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary is available.

Next apply seal coverage and mesh-budget projection. The seal coverage theorem confines the handoff to the same finite rows, and Definition 601.12 keeps diagonal-mesh consumers behind the displayed finite route.

Now read the cofinal tail-budget sibling. By Definition 698.3, the tail-budget packet exposes only the finite windows listed by its cofinal row and the shared threshold recorded by its budget row. By Definition 698.4, its $\mathsf{Real}^{\uparrow}$ seal read passes through those windows and the $\mathsf{RegSeqRat}$ handoff before the seal is consumed.

Composing the two routes gives the stated factorization: a consumer first reads the finite approximation rows of F, then the sibling $\mathsf{CofinalTailBudget}^{\uparrow}$ row, and only then the seal boundary. The non-escape clauses of the finite approximation and cofinal tail-budget packets exclude selected real values, quotient stream equality, infinite tails, host metrics, and completeness. □

Theorem 601.14 (Finite Cauchy approximation window induction)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet, and let $W_0, \ldots, W_m$ be a displayed finite list of* $\mathsf{StreamName}^{\uparrow}$ *approximation windows read from the same source row R and threshold row k. Finite iteration over this window list preserves the route through*

$$R, n, k, W, D, E, S, H, C, P, N.$$

In particular, every iterated window consumer reaches the $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary only after the displayed source, precision, threshold, window, dyadic ledger, error, handoff, transport, replay, provenance, and local naming rows have been read.*

Proof. Induct on the finite displayed window list. The empty list is covered by Definition 601.3: no consumer can read a dyadic observation or handoff without first exposing the source, precision, threshold, window, ledger, and error rows of F.

For the step, assume the route has been preserved through the first m windows and append one displayed window. Threshold-route coverage keeps the appended window under the same $\mathsf{CauchyModulus}^{\uparrow}$ threshold row k, refinement stability transports the existing finite surface through H, and the tail-meet dependency route prevents the appended window from becoming an independent tail oracle.

Finally apply seal coverage. The appended list still reaches S only through the same finite rows and structural support, so the induction exports no selected real value, quotient stream equality, infinite tail, host metric, or completeness principle. □

Theorem 601.15 (Finite Cauchy approximation tail induction)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose a finite chain of tail refinements is backed at each stage by the named* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *and* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *rows used by the finite approximation route. Then the final* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *read still factors through the same finite approximation rows*

$$R, n, k, W, D, E, S, H, C, P, N$$

together with the displayed meet and tail-meet backing rows. The chain exports no selected real value, quotient stream equality, infinite tail object, host metric, or ambient completeness principle.

Proof. Induct on the finite tail-refinement chain. The base case is the original finite approximation route: Definition 601.7 and Definition 601.11 place every Real or seal consumer behind the displayed finite rows.

For the step, append one refinement. The $\mathsf{CauchyModulusMeet}^{\uparrow}$ backing row supplies only the shared threshold, while the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ backing row supplies only the displayed common-tail window. By Definition 601.9, these rows are read as support for W, D, E rather than as new carriers.

The induction hypothesis already routes the preceding chain through $R, n, k, W, D, E, S, H, C, P, N$. Composing it with the appended meet and tail-meet support keeps the terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ read inside the same finite approximation surface. □

Theorem 601.16 (Finite Cauchy approximation obligation closure)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. The carrier row, local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ *obligations, window soundness, refinement stability,* $\mathsf{Real}^{\uparrow}$ *handoff, tail-meet triangle assembly, seal coverage, and mesh-budget projection assemble the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ *obligation closure over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

The closed obligation surface exports exactly the finite rows

$$R, n, k, W, D, E, S, H, C, P, N,$$

together with the named backing ledgers for the shared meet row, tail-meet threshold, triangle-bound tolerance assembly, diagonal-window coverage, and mesh-budget route. It exports no selected real value, quotient stream equality, infinite tail, host metric, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The carrier definition supplies the finite tuple F and excludes every non-displayed coordinate. Apply Definition 601.2 to identify the local naming obligations on exactly the carrier rows, and apply Definition 601.3 to bind the window, dyadic observation ledger, and error row to the displayed threshold.

Refinement stability then fixes the exported finite surface under the componentwise transports recorded in H. Apply Definition 601.7 to force every real-facing route through the handoff row S after the finite reads have been exposed.

For packets backed by the completion-side ledgers, apply Definition 601.9 and Definition 601.11. These theorems add only the named shared threshold, triangle-bound tolerance ledger, and seal-routing rows already listed in the packet route.

Finally apply Definition 601.12. The diagonal-window coverage and mesh-budget rows are read as backing ledgers for the same finite handoff, so the resulting obligation closure contains the displayed rows and named ledgers, with no selected real value, quotient stream equality, infinite tail, host metric, or completeness principle. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyApproximationUp.FiniteCauchyApproximationCarrier obligation closure`

Theorem 601.17 (Finite Cauchy approximation observation-budget compatibility)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet, and let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet share the precision request, finite window rows, dyadic ledger rows, regular-sequence handoff rows, seal rows, componentwise* hsame *transports, and displayed* Cont *routes with F. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F factors through the same finite observation-budget route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

as the accepted observation-budget packet, with E read as the displayed precision request on the observation-budget side and as the explicit finite error row attached to the selected precision on the finite-approximation side.

The compatible handoff exports no selected real value, quotient stream equality, infinite tail, host metric or equality, ambient completeness, or hidden limit selector.

Proof. First read the finite approximation side. The seal coverage theorem confines every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer to the displayed rows of F, and the mesh-budget projection keeps the downstream mesh-budget handoff inside the same finite source, window, dyadic ledger, error-row, transport, route, provenance, and naming surface.

The obligation closure theorem then packages those rows as the complete local $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ obligation surface. In particular, the finite approximation does not contribute an extra completion coordinate beyond the route through E, W, D, R, S and the displayed bookkeeping rows.

Now read the shared observation-budget packet. The finite-window seal-route totality theorem for $\mathsf{RealObservationBudget}^{\uparrow}$ sends every admitted seal read through $E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$. Route determinacy identifies any two accepted routes with the same displayed rows up to the componentwise hsame transports, and non-escape excludes hidden quotient, host equality, ambient completeness, hidden selector, and unbounded-schedule data.

Since the packets share the precision, window, dyadic, regular-sequence, seal, hsame, and Cont rows, the two consumer surfaces have the same displayed route. The finite approximation handoff is therefore the observation-budget handoff restricted to the finite Cauchy approximation rows, with no selected real value, quotient stream equality, infinite tail, host metric or equality, ambient completeness, or hidden limit selector exported. □

Theorem 601.18 (Finite Cauchy approximation budget-route exactness)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose its precision, finite-window, dyadic-ledger, error-row, and handoff rows agree componentwise, through the displayed* hsame *and* Cont *rows, with accepted packets from chapter 385, chapter 618, and chapter 611. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F routes through the same finite budget route used by those three budget packets and by the local surfaces Definition 601.12 and Definition 601.17.*

The exact route exports no selected real value, quotient stream equality, hidden limit selector, host metric, or ambient completeness coordinate.

Proof. Begin with the local finite approximation surface. The mesh-budget projection theorem forces any diagonal-mesh consumer to read the finite source, precision row, window ledger, dyadic ledger, error row, transports, continuations, provenance, and local naming row before it reaches the handoff.

Next read the sibling budget packets. The finite error-budget chapter supplies the explicit finite error route, the real diagonal mesh-budget chapter supplies the mesh-facing route through the same finite window and tolerance rows, and the real observation-budget chapter supplies the observation route

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

Apply the observation-budget compatibility theorem. Since the precision, window, dyadic, error, handoff, hsame, and Cont rows agree, the observation route and the finite approximation handoff route are the same displayed finite route.

Finally compose the route exclusions from the sibling packets with the local non-escape surface. None of the accepted carriers contains a coordinate for a selected real value, quotient stream equality, hidden limit selector, host metric, or ambient completeness principle, so no $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can receive such data through the budget route. □

Theorem 601.19 (Finite Cauchy approximation finite-observation selector handoff)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose its precision row, finite-window row, dyadic-ledger row, error row, and* $\mathsf{Real}^{\uparrow}$ *handoff row are selected by an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet through the displayed* hsame *and* Cont *rows. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F factors first through the selector's displayed finite-observation route and then through the local finite-approximation rows $R, n, k, W, D, E, S, H, C, P, N$.*

The handoff exports no selected real value, quotient stream equality, hidden limit selector, host metric, or ambient completeness coordinate.

Proof. Apply Definition 601.18. It already forces every real-facing or limit-seal consumer of F through the finite budget route whenever the precision, window, dyadic, error, and handoff rows agree componentwise with the accepted budget packets.

Read the accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet as the displayed selector for that same finite observation budget. Its route is the unique selector-facing route available by hypothesis, so the consumer first passes through the selector's finite-observation surface before the local finite approximation rows are exposed.

Finally compose the non-escape clauses. The budget-route exactness theorem excludes selected real values, quotient stream equality, hidden limit selectors, host metrics, and ambient completeness coordinates, and the selector contributes only the displayed finite-observation route. Therefore the composed handoff reaches $R, n, k, W, D, E, S, H, C, P, N$ without exporting any non-displayed completion data. □

Theorem 601.20 (Finite Cauchy approximation CauchySequenceSpace handoff)**.** *Let $F = (R, n, k, W, D, E, S, H,$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose F shares its source row, schedule and finite-window rows, dyadic ledger, seal rows, componentwise* hsame *transports, and displayed* Cont *routes with an accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packet. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F routes through the common finite-window surface read by chapter 406, with the regular source supplied by chapter 301 and the finite stream window supplied by chapter 126.*

The handoff exports only the shared finite packet and its seal-facing route; it exports no quotient stream equality, selected real value, infinite tail, host metric, or ambient completeness coordinate.

Proof. Project the finite approximation packet $F = (R, n, k, W, D, E, S, H, C, P, N)$. The existing seal coverage and budget-route exactness theorems force every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer to read the source row, precision, threshold, window, dyadic ledger, error row, seal route, transports, continuations, provenance, and local naming row.

The shared $\mathsf{CauchySequenceSpace}^{\uparrow}$ packet reads the same source and common-window classifier through the regular sequence and stream-name surfaces. By the hypotheses, the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ window of chapter 126, and the dyadic and seal rows agree componentwise with the rows of F.

Therefore the downstream CauchySequenceSpace classifier is the consumer of the finite-window surface already exposed by F. The exclusions follow from the local non-escape and budget-route exactness rows: no accepted coordinate contains quotient stream equality, a selected real, an infinite tail, a host metric, or ambient completeness. □

Theorem 601.21 (Finite Cauchy approximation shared-window budget exhaustion)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose its finite-window rows, dyadic-ledger rows, budget rows, seal rows, componentwise* hsame *transports, and displayed* Cont *routes agree componentwise with accepted* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, *and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *packets. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F reaches the same handoff row S through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, explicit error row E, and finite budget route before any seal handoff.*

The shared finite route is the route exposed by chapter 126, chapter 611, chapter 618, and chapter 406. It exports no selected real value, quotient stream equality, infinite tail, host metric, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First apply Definition 601.4. This fixes the finite threshold, window, dyadic ledger, explicit error row, and local route support as the only displayed path from the approximation packet toward the seal-facing handoff.

Next apply Definition 601.18. Because the observation-budget and diagonal-mesh budget rows agree componentwise with F, every budget-facing read uses the same finite route through W, D, E, and S before any downstream seal read is available.

Finally apply Definition 601.20. The CauchySequenceSpace packet reads the same finite-window surface through the stream-name and regular-sequence handoff rows, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer cannot bypass the displayed budget route when it reaches S.

The exclusions are inherited from the three component route theorems. None of the participating carriers contains a coordinate for a selected real value, quotient stream equality, infinite tail, host metric, or $\mathsf{Real}^{\uparrow}$ completeness, so those objects cannot be exported by the shared finite-window route. □

Theorem 601.22 (Finite Cauchy approximation selector-budget determinacy)**.** *Let $F = (R, n, k, W, D, E, S, H, C,$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose F shares its finite-window rows, dyadic-ledger rows, error rows, seal rows, componentwise* hsame *transports, and displayed* Cont *routes with an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Then any two* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumers with the same selector route determine the same handoff row S and the same displayed finite approximation ledger of F.*

The determined ledger is the finite row package $R, n, k, W, D, E, S, H, C, P, N$ as read through the selector route. It exports no selected real value, quotient stream equality, infinite tail, host metric, ambient completeness theorem, or hidden budget selector.

Proof. Start with Definition 601.4. It forces every downstream real or limit-seal consumer to pass through the displayed threshold, finite stream window, dyadic ledger, error row, and handoff seal route of F.

Next apply Definition 601.18. Since the budget-facing rows agree componentwise with the accepted selector packet, a consumer using the fixed selector route reads the same W, D, E, S rows before any real or limit-seal handoff is available.

Use Definition 601.21 for the common finite-window surface. The shared route exhausts the $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ reads through the displayed window, dyadic ledger, explicit error row, and finite budget route, so the handoff row S and the visible finite approximation ledger cannot vary between the two consumers.

Finally use selector-route non-escape. The participating packets contain only displayed finite-window, dyadic-ledger, error, seal, transport, continuation, provenance, and naming rows; they contain no selected real value, quotient stream equality, infinite tail, host metric, ambient completeness theorem, or hidden budget selector. □

Theorem 601.23 (Finite Cauchy approximation shared-budget non-escape)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P,$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose F shares its finite-window rows, dyadic-ledger rows, error rows, handoff rows, selector route, componentwise* hsame *transports, and displayed* Cont *rows with accepted* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$*,* $\mathsf{CauchySequenceSpace}^{\uparrow}$*, and* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of F factors through the same displayed finite route*

$$W \longmapsto D \longmapsto E \longmapsto S$$

and through the same selector-budget route before any $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff is available.*

The shared route exports only the finite rows of F, the displayed budget rows, the shared selector route, the componentwise hsame *transports, the displayed* Cont *rows,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. It exports no selected real value, quotient stream equality, infinite tail, host metric or equality, ambient completeness theorem, or hidden selector data.*

Proof. First apply Definition 601.4. This identifies the threshold, finite window W, dyadic ledger D, explicit error row E, handoff row S, transport rows, continuation rows, provenance row, and naming row as the displayed route available from F.

Next apply Definition 601.18. The shared $\mathsf{RealObservationBudget}^{\uparrow}$ and $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ packets read the same finite budget surface, so any budget-facing $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer must pass through the displayed W, D, E, S route before a seal handoff is visible.

Use Definition 601.20 for the sequence-space side. Because the $\mathsf{CauchySequenceSpace}^{\uparrow}$ packet shares the source, finite-window, dyadic-ledger, seal, hsame, and Cont rows with F, it contributes only the same finite-window handoff route and no separate quotient stream equality or infinite tail.

Now apply Definition 601.21 and Definition 601.22. The first theorem exhausts all real and limit-seal reads through the shared finite budget route, and the second fixes the selector-facing read of that route to the displayed selector-budget data.

Composing the non-escape clauses from these route theorems leaves exactly the displayed finite rows, budget rows, selector route, transports, continuations, provenance, and local name. None of the accepted carriers contains a coordinate for selected real values, quotient stream equality, infinite tails, host metric or equality, ambient completeness, or hidden selector data, so no consumer can receive any such object before the seal handoff. □

Theorem 601.24 (Finite Cauchy approximation L10 window exhaustion)**.** *Let $F = (R, n, k, W, D, E, S, H, C, P, N$ be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet consumed by a current Phase* $\mathsf{Real}^{\uparrow}$ *route and by a current Completion terminal read. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary read from F first exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, unary precision row n,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row k, finite* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, explicit error row E, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows S, H, C, P, N.*

The L10 finite-window route exports no quotient stream identity, selected real value, host real equality, hidden tail selector, ambient completeness row, or independent terminal endpoint.

Proof. First apply Definition 601.4. It forces every real-facing or limit-seal handoff to read R, n, k, W, D, E before any boundary seal can be exposed.

Next apply Definition 601.11. The handoff through S reaches the current $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary only together with the displayed transport, continuation, provenance, and local naming rows H, C, P, N.

Finally use Definition 601.23. Its non-escape clause rules out quotient stream identity, selected real values, host real equality, hidden tail selectors, ambient completeness rows, and independent terminal endpoints along the shared budget route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, unary precision row, $\mathsf{CauchyModulus}^{\uparrow}$ threshold, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, explicit error-bound row, $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 601.1, Definition 601.2, Definition 601.3, Definition 601.4, Definition 601.5, Definition 601.6, Definition 601.7, Definition 601.11, Definition 601.12, Definition 601.13, Definition 601.16, Definition 601.17, Definition 601.18, Definition 601.20, Definition 601.21, Definition 601.23, and Definition 601.24 bind the paper-side mature finite approximation packet over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, while Lean verification and bridge checking remain separate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FiniteCauchyApproximationUp.FiniteCauchyApproximationCarrier\_obligation\_closure`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected real values, infinite tails, host real metrics, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification and bridge checking require separate grounding for the finite approximation packet and its standard bridge.

Chapter 602

A Concrete Naming Certificate for $\mathsf{FullAxisRealRefusal}^{\uparrow}$

The $\mathsf{FullAxisRealRefusal}^{\uparrow}$ surface packages the AxisZeckendorf refusal that a $\mathsf{FullAxis}^{\uparrow}$ structural seal may be read as $\mathsf{Real}^{\uparrow}$. It is a finite certificate over already visible seal, refusal, transport, continuation, provenance, and naming rows. It exports no $\mathsf{Real}^{\uparrow}$ algebraic operation, no Cauchy completion, and no host limit.

Definition 602.1 (FullAxis-to-Real refusal carrier)**.** *A* $\mathsf{FullAxisRealRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, R, C, H, T, P, N)$$

with the following visible rows:

1. *F is the* $\mathsf{FullAxis}^{\uparrow}$ *structural-seal inventory, namely the prefix-thread carrier, source, pattern, classifier, stability, ledger, finite boundary marker, and public seal-name rows of chapter 279;*
2. *R is the Real-facing refusal row, citing the fact that* $\mathsf{FullAxis}^{\uparrow}$ *does not entail the Cauchy, limit-uniqueness, field, compactness, metric-completion, or selected-limit operations of* $\mathsf{Real}^{\uparrow}$*;*
3. *C is the AxisZeckendorf cannot-claim row for the* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$ *horizon, including the boundary refusal of Definition 9350.6;*
4. *H records the displayed* hsame *transports for the structural seal row, the Real-facing refusal row, the cannot-claim row, the package row, and the local naming row;*
5. *T records the displayed* Cont *routes by which consumers may read the seal inventory and replay the refusal without producing a positive* $\mathsf{Real}^{\uparrow}$ *bridge;*
6. *P is the* Pkg *provenance over* $\mathsf{FullAxis}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the source* NameCert *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{FullAxisRealRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for addition, multiplication, order, Cauchy convergence, limit uniqueness, completeness, compactness, a quotient of histories, or host equality.

Lean4 ✓ `BEDC.Derived.FullAxisRealRefusalUp.FullAxisRealRefusalTasteGate single carrier alignment`

Theorem 602.2 (FullAxis-to-Real refusal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FullAxisRealRefusal}^{\uparrow}}$ *surface for Definition 602.1 consists exactly of five finite obligations:*

1. *carrier admission for the packet* $Q = (F, R, C, H, T, P, N)$*;*
2. *structural-seal coverage for the prefix-thread, boundary-finite, source-exhaustion, Real-route-separation, and seal-name rows collected in* F*;*
3. *refusal coverage for the Real-facing row* R *and the AxisZeckendorf cannot-claim row* C*;*
4. *transport and continuation exactness for the displayed* hsame *rows* H *and* Cont *routes* T*;*
5. *ledger non-escape through the* Pkg *provenance row* P *and local* NameCert *row* N*.*

Proof. Carrier admission is projection from Definition 602.1. The admitted packet has exactly seven visible rows, so every later read starts from F, R, C, H, T, P, N.

Structural-seal coverage is supplied by the first row. The row F names the prefix-thread carrier, finite boundary marker, source-exhaustion statement, Real-route separation statement, and public seal-name alignment from the $\mathsf{FullAxis}^{\uparrow}$ chapter.

Refusal coverage is supplied by R and C. The row R records the missing $\mathsf{Real}^{\uparrow}$ operations, while C points to the AxisZeckendorf cannot-claim horizon for the same boundary.

Transport and continuation exactness are the rows H and T. They move only displayed refusal and seal rows, and therefore cannot turn the refusal into an algebraic or completion operation.

Finally, P and N record package provenance and the local naming ledger. Since none of the excluded $\mathsf{Real}^{\uparrow}$ operations is a coordinate of the packet, the NameCert surface is exhausted by the five displayed obligations. □

Theorem 602.3 (FullAxis-to-Real refusal non-escape)**.** *Every public read exported by an accepted* $\mathsf{FullAxisRealRefusal}^{\uparrow}$ *packet factors through the rows* F, R, C, H, T, P, N*. In particular, no consumer can extract a* $\mathsf{Real}^{\uparrow}$ *addition, multiplication, order, Cauchy convergence proof, limit-uniqueness proof, completeness row, compactness row, selected host limit, quotient stream, or host equality from the packet.*

Proof. Inspect the packet coordinates. The structural-seal row F exposes only the $\mathsf{FullAxis}^{\uparrow}$ prefix-thread and finite-boundary surface, while the rows R and C expose refusal data.

The transport and continuation rows H and T preserve those displayed rows componentwise. A transported refusal remains a refusal row, and a transported structural-seal read remains a read of the same prefix-thread inventory.

The provenance and naming rows P and N do not add mathematical operations. Because no coordinate contains a $\mathsf{Real}^{\uparrow}$ algebraic operation, completion operation, quotient stream, selected limit, or host equality, no public read can export one. □

Lean4 ✓ `BEDC.Derived.FullAxisRealRefusalUp.FullAxisRealRefusal non escape`

Theorem 602.4 (FullAxis-to-Real refusal vision concretization)**.** *The AxisZeckendorf cannot-claim boundary for* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$ *is realized by the finite* $\mathsf{FullAxisRealRefusal}^{\uparrow}$ *packet of Definition 602.1. The realization cites* $\mathsf{FullAxis}^{\uparrow}$ *structural-seal rows, Real-facing refusal rows, AxisZeckendorf cannot-claim rows, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and a local* NameCert *row, while exporting no* $\mathsf{Real}^{\uparrow}$ *operation.*

Proof. Start from the cannot-claim theorem Definition 9350.6. It supplies the boundary statement that the AxisZeckendorf cluster does not assert $\mathsf{Real}^{\uparrow}$ field operations on $\mathsf{FullAxis}^{\uparrow}$.

The carrier of Definition 602.1 turns that boundary into finite rows. The row F records the $\mathsf{FullAxis}^{\uparrow}$ seal surface, R records the Real-facing refusal, and C records the AxisZeckendorf horizon row.

Rows H, T, P, N provide the admissible transport, continuation, package, and local naming surface. By Definition 602.3, this realization cannot be read as a positive $\mathsf{Real}^{\uparrow}$ bridge, so it concretizes the vision boundary as a finite refusal certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FullAxisRealRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{FullAxisRealRefusal}^{\uparrow}$ packet is generated from FullAxis structural-seal rows, Real-facing refusal rows, the AxisZeckendorf cannot-claim horizon, displayed hsame transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FullAxisRealRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the carrier, NameCert obligations, non-escape boundary, and AxisZeckendorf vision concretization for finite FullAxis-to-Real refusal packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FullAxisRealRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a RealUp bridge, RealUp algebraic operations for FullAxisUp, Cauchy completion, compactness, quotient streams, selected host limits, or host equality.

Upgrade path: Obligation closure requires checked carrier coverage, refusal-row coverage, transport exactness, continuation exactness, and ledger non-escape for the displayed packet.

Chapter 603

A Concrete Naming Certificate for $\mathsf{HistTimeStream}^{\uparrow}$

The $\mathsf{HistTimeStream}^{\uparrow}$ interface realizes the local-time-stream reading of chapter 9418 as a finite BEDC naming surface. A stream is not a clock shared by distinct histories; it is the displayed sequence of Cont steps by which one source history is generated from e.

Definition 603.1 (HistTimeStream NameCert carrier)**.** *A* $\mathsf{HistTimeStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, T, S, C, L, P, N)$$

with the following visible rows:

1. *H is the source* Hist *row being read as a local stream;*
2. *T is the finite sequence of* $\mathsf{E1}$ *and* $\mathsf{E0}$ *time-step rows whose terminal history is H;*
3. *S is the start-at-empty source ledger, recording that the sequence begins at* e *and reaches only the displayed source history;*
4. *C records the* Cont *route for each time-step row in T;*
5. *L records componentwise* hsame *transport for the displayed rows and the no-global-clock ledger preventing a second history from importing this stream as a synchronizing clock;*
6. *P is the* Pkg *provenance row naming* BHist, Hist, Cont, hsame, $\mathsf{ProbeBundle}$, Pkg, *and the local stream packet;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets by the displayed rows H, T, S, C, L, P, N. It has no coordinate for a host natural-number index, no ambient time parameter, and no equality synchronizing the local streams of distinct Hist values.

Lean4 (definition) `BEDC.Derived.HistTimeStreamUp.HistTimeStreamUp`

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamTasteGate single carrier alignment`

Theorem 603.2 (HistTimeStream NameCert obligation sketch)**.** *The local* $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *obligation surface has five displayed checks: carrier admission for $Q = (H, T, S, C, L, P, N)$, exactness of the start-at-empty step ledger S, stability of every* Cont *route recorded in C, componentwise* hsame *transport through L, and no-global-clock non-escape for the stream named by N.*

Proof. Project the finite carrier of Definition 603.1. The row H fixes the source history, while T and S record the finite step ledger from e to that history.

For each displayed step in T, read the corresponding route in C. These routes are the only permitted transitions, so the stream has no hidden branch outside the finite packet.

Transport is then performed rowwise through L. The same ledger also records that a second Hist cannot import this local stream as a shared clock, because the packet names only the source row H and its own finite route list.

Finally, P and N repack the source, step, route, transport, and locality rows as a local $\mathsf{NameCert}$ surface. This is exactly the concrete packet needed for the observer-history time-as-growth reading. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier namecert obligation sketch`

Theorem 603.3 (HistTimeStream observation carrier)**.** *Every accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet exposes the source* Hist *row* H*, the finite* $\mathsf{E1}/\mathsf{E0}$ *schedule* T*, the start-at-empty ledger* S*, and the displayed* Cont *replay row* C *as carrier coordinates of*

$$Q = (H, T, S, C, L, P, N).$$

The packet has no accepted observation surface apart from these coordinates, the componentwise hsame *ledger* L*, package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *row* N*.*

Proof. Project the accepted packet through Definition 603.1. The source row, schedule row, start ledger, continuation replay ledger, transport ledger, package provenance, and local naming row are exactly the displayed coordinates of Q.

Apply Definition 603.2. Its carrier-admission and step-ledger clauses identify H, T, S, C as the public observation rows, while the remaining clauses leave only L, P, N as transport, provenance, and naming data.

No other coordinate is present in the finite tuple. Therefore an accepted packet cannot export a separate observation carrier, ambient clock, or hidden route outside the listed rows. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier observation carrier`

Theorem 603.4 (HistTimeStream schedule classifier stability)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, componentwise* hsame *transport along the rows stored in* L *preserves the finite schedule* T*, the start-at-empty ledger* S*, and the displayed* Cont *replay row* C*. The transported classifier remains indexed only by the displayed carrier rows and cannot introduce a host-indexed global time coordinate.*

Proof. Begin with the classifier description in Definition 603.1. It compares packets by the visible rows H, T, S, C, L, P, N and provides no coordinate for a natural-number host index or shared ambient time parameter.

Transport is componentwise through L. Each transported schedule entry remains one of the displayed $\mathsf{E1}/\mathsf{E0}$ rows of T, the start ledger remains attached to e and the terminal source row H, and every replayed route is still read through the corresponding row of C.

The no-global-clock clause in Definition 603.2 blocks the only possible escape route: importing the local stream into a second Hist as a synchronizing clock. Thus the transported classifier preserves the finite schedule and cannot acquire a host-indexed time row. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier schedule classifier stability`

Theorem 603.5 (HistTimeStream NameCert ledger exhaustion)**.** *Every public* $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *read factors through the carrier row, the finite step ledger, route stability, componentwise transport, package provenance, and the local naming row:*

$$H, T, S, C, L, P, N.$$

No public read from the packet yields inter-Hist synchronization, a rate claim, an ambient global clock, or a host time index.

Proof. First use the observation-carrier theorem to place every public read inside the finite tuple $Q = (H, T, S, C, L, P, N)$. The source, schedule, start ledger, route replay, transport, package, and local naming rows are the complete carrier inventory.

Next use schedule classifier stability. It keeps the step ledger and Cont replay rows stable under componentwise hsame transport and prevents the classifier from acquiring a global-time coordinate.

Finally inspect the package and local naming rows. They certify only the displayed BHist, Hist, Cont, hsame, ProbeBundle, Pkg, and NameCert provenance for this one local stream, so the finite coordinate list exhausts the public NameCert ledger and leaves no synchronization or rate claim to read. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier namecert ledger exhaustion`

Theorem 603.6 (HistTimeStream observer-locality lattice link)**.** *Every accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *read of*

$$Q = (H, T, S, C, L, P, N)$$

is consumed as the local-time component of the observer-history identity and observer-locality surfaces of chapter 479 and chapter 889. The stream supplies the displayed source history, finite step ledger, continuation replay, and no-global-clock transport row, but it supplies no inter-Hist synchronization, shared clock, or imported host-time index.

Proof. Project the accepted packet $Q = (H, T, S, C, L, P, N)$. The rows H, T, S, C give the source history, start-at-empty finite schedule, and displayed Cont replay that the observer-history identity surface reads as its local growth component.

Use Definition 603.5. Its no-global-clock ledger keeps the read attached to the single source row H and blocks any second Hist from importing T as a synchronizing clock.

The sibling packets of chapter 479 and chapter 889 therefore consume this same local stream through L, P, N. They may use it to identify observer-history locality and transport surfaces, but they add no inter-Hist clock equality, host-time coordinate, or cross-history synchronization row. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamObserverLocalityLatticeLink`

Theorem 603.7 (HistTimeStream public export)**.** *An accepted packet*

$$Q = (H, T, S, C, L, P, N)$$

exports exactly the local-time stream $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *surface: the source* Hist *row* H*, the finite* E1/E0 *schedule* T*, the start-at-empty ledger* S*, the displayed* Cont *replay row* C*, the componentwise* hsame *and no-global-clock ledger* L*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*. No public export from* Q *yields an inter-Hist clock, host time index, global synchronization row, temporal-rate claim, or hidden route outside those displayed coordinates.*

Proof. Start from the carrier of Definition 603.1. It lists exactly the rows H, T, S, C, L, P, N, with H as the source history, T as the finite E1/E0 schedule, S as the start-at-empty ledger, C as the displayed replay row, and L, P, N as transport, provenance, and local naming data.

Use the observation carrier theorem, Definition 603.3, to identify H, T, S, C as the only observation rows. Then apply schedule classifier stability, Definition 603.4, so componentwise hsame transport preserves the finite schedule and cannot add a host-indexed global time coordinate.

Next apply ledger exhaustion, Definition 603.5. It forces every public $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ read through the same carrier, schedule, replay, transport, package, and naming rows.

Finally use the observer-locality lattice link, Definition 603.6. The downstream observer-history surfaces consume the packet as one local stream, so the public export is precisely the displayed local-time stream surface and no inter-Hist synchronization or hidden route is available. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier public export`

Theorem 603.8 (HistTimeStream window habitation)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Any finite window accepted by the local* $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$ *surface is inhabited only by the displayed rows*

$$H, T, S, C, L, P, N$$

and its source route begins at e*. The window has no additional row for a host time index, ambient clock coordinate, sibling Hist source, or hidden pre-start segment.*

Proof. Start from the carrier definition, Definition 603.1. It presents the accepted packet as exactly the seven rows H, T, S, C, L, P, N, with S recording that the finite route begins at e and reaches only the displayed source history.

Next apply the NameCert obligation sketch, Definition 603.2. Its carrier-admission and start-ledger clauses make the finite window a local packet window rather than an ambient time interval, and its route-stability clause reads the window only through the displayed Cont rows.

Finally use ledger exhaustion, Definition 603.5. Every public read of the window must pass through the same carrier, schedule, replay, transport, package, and naming rows, so no host index, sibling source, or hidden pre-start row can inhabit the accepted window. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier window habitation`

Theorem 603.9 (HistTimeStream Real StreamName window lock)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$*-facing finite-window read of Q must pass through the displayed H, T, S, C, L, P, N rows and then through the StreamName handoff surface. It cannot read a host time index, global clock, sibling Hist source, or hidden stream row outside the accepted local packet.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier real streamname window lock`

Proof. Start from the carrier and obligation sketch, Definition 603.1 and Definition 603.2. These identify the accepted packet as exactly the source history, finite schedule, start ledger, replay, transport, provenance, and local naming rows.

Apply schedule classifier stability and ledger exhaustion, Definition 603.4 and Definition 603.5. The finite E1/E0 schedule is preserved under componentwise hsame transport, and every public read remains inside the displayed packet rows.

Finally use the observer-locality lattice link, public export, and window habitation theorems, Definition 603.6, Definition 603.7, and Definition 603.8. They route the finite window through the

local StreamName-facing handoff and exclude every nonlocal clock or hidden stream coordinate. □

Theorem 603.10 (HistTimeStream RegSeqRat local clock separation)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer may use the finite local schedule T as an observation index for the displayed packet, but it cannot promote T to a shared host clock, an inter-Hist synchronization row, or a temporal-rate coordinate.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regseqrat local clock separation`

Proof. Begin with the carrier and obligation sketch, Definition 603.1 and Definition 603.2. They make T a finite local schedule row tied to the source history H and the displayed replay row C.

Next apply schedule classifier stability and ledger exhaustion, Definition 603.4 and Definition 603.5. The schedule survives only as a component of the same local packet under hsame transport, and the ledger has no row for shared host time or inter-Hist synchronization.

The observer-locality lattice link, public export, and window habitation theorems, Definition 603.6, Definition 603.7, and Definition 603.8, then bound the RegSeqRat-facing use of T. It remains an observation index for the finite local stream and cannot become a global clock coordinate. □

Theorem 603.11 (HistTimeStream Real completion consumer boundary)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet $Q = (H, T, S, C, L, P, N)$, every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may read only the local stream packet, its componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No such consumer obtains a host real equality row, ambient completeness principle, quotient stream, selected limit, global clock, or inter-Hist synchronization row from Q.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier real completion consumer boundary`

Proof. First expose the local stream packet through Definition 603.1 and Definition 603.2. The only available rows are H, T, S, C, L, P, N, with C as the replay route, L as componentwise hsame transport, P as provenance, and N as the local naming row.

Apply schedule classifier stability and ledger exhaustion, Definition 603.4 and Definition 603.5. These force every consumer-facing read to remain within the same finite schedule and local ledger, so completion-facing access cannot introduce detached Real or global-clock data.

Finally use the observer-locality lattice link, public export, and window habitation theorems, Definition 603.6, Definition 603.7, and Definition 603.8. Their common boundary is the displayed local packet with hsame, Cont, Pkg, and NameCert support, and carrier-coordinate exhaustion removes each listed external coordinate. □

Theorem 603.12 (HistTimeStream transport lock)**.** *For accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packets $Q = (H, T, S, C, L, P, N)$ and $Q' = (H', T', S', C', L', P', N')$, componentwise* hsame *transport from Q to Q' preserves the finite schedule rows and the displayed* Cont *replay rows. The transported packet may replay the local window through $H', T', S', C', L', P', N'$, but it cannot add a host time index, shared clock coordinate, or external rate row.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamTransportLock`

Proof. Begin with the carrier definition, Definition 603.1. The transport row L is componentwise transport for the displayed packet rows, not a map into an external clock carrier.

Apply schedule classifier stability, Definition 603.4. It keeps the finite E1/E0 schedule, the start-at-empty ledger, and the replay row stable under componentwise hsame transport, so the transported consumer sees the corresponding rows of Q'.

Use ledger exhaustion, Definition 603.5, to close the transport surface. Since every public read remains inside the same carrier, schedule, replay, transport, package, and naming inventory, there is no coordinate where a host time index, shared clock, or external rate can be inserted. □

Theorem 603.13 (HistTimeStream clock refusal obligation)**.** *No accepted consumer of a* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$ *can read a shared inter-Hist clock, synchronized external time, or temporal-rate coordinate from the packet. Every accepted time-facing read is confined to the finite local window displayed by* H, T, S, C, L, P, N *and to componentwise* hsame *transport of that same window.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier clock refusal`

Proof. First expose the accepted packet through Definition 603.1. The carrier lists the source history, finite schedule, start ledger, replay row, transport row, provenance row, and local naming row, and it has no coordinate reserved for a shared clock or rate.

Next apply the NameCert obligation sketch and schedule classifier stability, Definition 603.2 and Definition 603.4. These keep the local stream tied to the displayed finite schedule and block componentwise transport from becoming a host-indexed global time classifier.

Finally use ledger exhaustion, Definition 603.5. The public NameCert read is exhausted by the displayed local rows and their transport ledger, so an accepted consumer has no path to a shared inter-Hist clock, synchronized external time, or temporal-rate coordinate. □

Theorem 603.14 (HistTimeStream completion StreamName handoff factorization)**.** *Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet*

$$Q = (H, T, S, C, L, P, N)$$

factors first through the displayed local packet H, T, S, C, L, P, N*, then through the StreamName-facing finite-window handoff. The consumer cannot replace the local schedule by a host time index, global clock, sibling Hist source, quotient stream, selected limit, or ambient completeness row.*

Proof. Start with the carrier and NameCert obligation sketch, Definition 603.1 and Definition 603.2. They make H, T, S, C, L, P, N the whole accepted local packet and route every public read through the displayed schedule, replay, transport, provenance, and naming rows.

Apply the Real StreamName window lock and RegSeqRat local clock separation, Definition 603.9 and Definition 603.10. The first sends Real-facing finite-window reads through the StreamName handoff, while the second keeps the local schedule from becoming a host clock or shared synchronization row.

Use the Real completion consumer boundary and transport lock, Definition 603.11 and Definition 603.12. They keep completion-facing reads and componentwise transport inside the same finite local packet.

Finally apply the clock refusal obligation, Definition 603.13. Since no shared clock, host index, sibling Hist source, quotient stream, selected limit, or ambient completeness coordinate exists in the packet, the completion or Real consumer factors through the local packet before the StreamName handoff and nowhere else. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier completion streamname handoff factorization`

Theorem 603.15 (HistTimeStream observer-locality consumption bridge)**.** *Any observer-locality or observer-history-identity consumer of an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet factors through the displayed local stream rows*

$$H, T, S, C, L, P, N$$

before reading observer-locality transport. Such a consumer cannot add inter-Hist synchronization, a global clock, or a host time index to the consumed packet.

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier observer locality consumption bridge`

Proof. Start with the carrier of Definition 603.1. It presents the source history, finite step schedule, start ledger, continuation replay, transport, provenance, and local naming rows as the complete local stream packet.

Apply the NameCert obligation sketch and the observer-locality lattice link. The certificate routes every public read through the same H, T, S, C, L, P, N inventory, and the lattice link identifies that inventory as the local-time component consumed by observer-history identity and observer-locality transport.

Finally use the public export theorem. Since the exported surface has no coordinate for inter-Hist synchronization, a shared global clock, or a host time index, every observer-locality consumer must read the displayed local stream rows before transport and cannot extend them with a hidden time source. □

Theorem 603.16 (HistTimeStream observer-locality bridge route)**.** *Every observer-locality consumer reads an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet first through the displayed rows*

$$H, T, S, C, L, P, N,$$

then through the sibling $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *and* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *surfaces of chapter 479 and chapter 889. The route cannot add a global clock, host time index, inter-Hist synchronization row, or temporal-rate row.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamObserverLocalityBridgeRoute`

Proof. Start from the public export theorem, Definition 603.7. It makes H, T, S, C, L, P, N the whole local-time stream surface visible to an outside consumer.

Next apply the observer-locality lattice link. That theorem identifies the same carrier rows as the local-time component consumed by observer-history identity and observer-locality transport, so the sibling surfaces are reached only after the finite HistTimeStream packet has been read.

Finally use the consumption bridge of Definition 603.15. Since the consumed packet has no coordinate for a global clock, host time index, inter-Hist synchronization, or rate measurement, the bridge route cannot add any such row downstream. □

Theorem 603.17 (HistTimeStream observer-locality transport)**.** *Let*

$$Q = (H, T, S, C, L, P, N)$$

be an accepted $\mathsf{HistTimeStream}^{\uparrow}$ *packet. If the public export of* Q *is consumed by* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *or* $\mathsf{ObserverLocalityTransport}^{\uparrow}$*, then the transport through the sibling observer-locality surfaces carries exactly the displayed local-time rows* H, T, S, C, L, P, N*. It preserves the no-global-clock ledger* L *and cannot add inter-Hist synchronization, host-indexed time, temporal-rate rows, or hidden routes.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier observer locality transport`

Proof. Start with the carrier definition of Definition 603.1. The accepted packet has only the source Hist row H, finite E1/E0 schedule T, start-at-empty ledger S, displayed Cont replay row C, componentwise hsame and no-global-clock ledger L, package provenance row P, and local NameCert row N.

Apply the public export theorem, Definition 603.7. It identifies those seven rows as the whole public local-time stream surface and excludes a global clock, host time index, inter-Hist synchronization row, temporal-rate claim, or hidden route from the exported packet.

Next use the observer-locality lattice link, Definition 603.6. The sibling ObserverHistoryIdentity$^{\uparrow}$ and ObserverLocalityTransport$^{\uparrow}$ surfaces consume the exported local-time rows as the same finite stream, so the consumer receives no additional time coordinate before transport.

Finally apply the route order of Definition 603.16. Since the route reaches the sibling observer-locality surfaces only after H, T, S, C, L, P, N have been read, the resulting transport preserves the no-global-clock ledger and cannot attach any forbidden synchronization, host-indexed time, rate, or hidden-route row. □

Theorem 603.18 (HistTimeStream Cont prefix exactness)**.** *Let*

$$Q = (H, T, S, C, L, P, N)$$

be an accepted HistTimeStream$^{\uparrow}$ *packet. The displayed finite time stream is exactly the* Cont*-prefix route from* e *to the source* Hist *row H, read through the finite schedule T, the start-at-empty ledger S, the replay row C, the transport and locality ledger L, the package row P, and the local naming row N.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier cont prefix exactness`

Proof. Start with the carrier of Definition 603.1. It presents H, T, S, C, L, P, N as the complete local stream packet, with S recording the start-at-empty boundary and C recording the displayed replay route.

Use the observation carrier theorem, Definition 603.3, to identify H, T, S, C as the whole finite observation prefix. Schedule classifier stability, Definition 603.4, keeps that prefix tied to the finite E1/E0 schedule.

Finally apply ledger exhaustion and observer-locality transport, Definition 603.5 and Definition 603.17. These theorems force every consumer of the local stream through the same prefix and leave no hidden route between e and H. □

Theorem 603.19 (HistTimeStream hsame prefix transport)**.** *For accepted* HistTimeStream$^{\uparrow}$ *packets $Q = (H, T, S, C, L, P, N)$ and $Q' = (H', T', S', C', L', P', N')$, componentwise* hsame *transport preserves the* Cont*-prefix route of Definition 603.18. It may replay the displayed local stream rows from Q to Q', but it cannot create a host-indexed time coordinate, an inter-Hist clock, or a global synchronization row.*

Proof. Begin with Cont prefix exactness. Theorem 603.18 identifies the prefix route entirely from the displayed rows H, T, S, C, L, P, N.

Now transport the finite schedule and ledger rows. By schedule classifier stability, Definition 603.4, componentwise hsame preserves the finite E1/E0 schedule. By ledger exhaustion, Definition 603.5, the no-global-clock ledger remains part of the same packet rather than becoming a host time index.

Finally use observer-locality transport, Definition 603.17. The downstream observer-locality route consumes the transported packet as local rows only, so no inter-Hist clock or global synchronization coordinate is produced by the transport. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier hsame prefix transport`

Theorem 603.20 (HistTimeStream observer-locality totality)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Every observer-locality or observer-history consumer reads exactly the displayed local rows*

$$H, T, S, C, L, P, N$$

through the carrier, finite step ledger, replay route, no-global-clock locality row, provenance row, and local naming row. No such consumer obtains a global clock, host time index, inter-Hist synchronization row, temporal-rate coordinate, or hidden route.

Proof. Start with the carrier and observation carrier. The definition Definition 603.1 displays H, T, S, C, L, P, N, and Definition 603.3 identifies H, T, S, C as the finite observation surface.

Next apply schedule classifier stability and ledger exhaustion. Theorem 603.4 keeps the finite $\mathsf{E1/E0}$ schedule attached to T, while Definition 603.5 keeps the no-global-clock ledger and local naming rows inside the same packet.

Finally use the observer-locality lattice link, public export, and bridge route. Theorems 603.6, 603.7, and 603.16 show that observer-facing consumers receive only the local packet before any sibling route is available. Thus totality holds on H, T, S, C, L, P, N and all nonlocal time reads are excluded. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier observer locality totality`

Theorem 603.21 (HistTimeStream start-ledger uniqueness)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. The start-at-empty ledger S determines the displayed finite schedule endpoint H inside Q through the schedule row T and replay row C. No accepted read of Q carries a second start row, a second endpoint row, or a host-indexed clock coordinate from which another start ledger can be recovered.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamStartLedgerUniqueness`

Proof. Start with the carrier and observation carrier. The definition Definition 603.1 fixes Q as the displayed tuple H, T, S, C, L, P, N, and Definition 603.3 identifies the observed prefix rows as H, T, S, C.

Next apply Cont prefix exactness, Definition 603.18. The replay route starts at e, passes through the finite schedule and start ledger, and ends at the displayed source row H.

Finally use ledger exhaustion, Definition 603.5. Since the local NameCert surface has no additional start or endpoint coordinate, the start ledger determines the schedule endpoint inside Q and admits no second start row. □

Theorem 603.22 (HistTimeStream Cont-step coverage)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Every public local-time step read from Q is one of the displayed* $\mathsf{E1/E0}$ *schedule entries in T replayed by C. The packet exposes no step outside the finite schedule, no temporal-rate coordinate, and no hidden continuation branch.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier cont step coverage`

Proof. First read the schedule classifier. By Definition 603.4, the finite E1/E0 schedule is the classifier-stable step surface carried by T.

Next apply Cont prefix exactness, Definition 603.18. The continuation route C replays the displayed prefix from e to H through the schedule and start ledger, so every public step read is already a row of T.

Finally use public export and ledger exhaustion, Definition 603.7 and Definition 603.5. They confine public consumers to the displayed packet and leave no coordinate for an extra step, rate, or hidden continuation branch. □

Theorem 603.23 (HistTimeStream hsame schedule endpoint transport)**.** *For accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packets* $Q = (H, T, S, C, L, P, N)$ *and* $Q' = (H', T', S', C', L', P', N')$*, componentwise* hsame *transport through* L *preserves the finite schedule endpoint and the no-global-clock ledger. Thus transporting* Q *to* Q' *carries the endpoint determined by* S, T, C *to the endpoint determined by* S', T', C' *without adding a global clock, inter-Hist synchronization row, or host time index.*

Proof. Begin with hsame prefix transport, Definition 603.19. It preserves the displayed Cont-prefix route componentwise and cannot create a host-indexed time coordinate, inter-Hist clock, or global synchronization row.

Use start-ledger uniqueness and Cont-step coverage next. Theorems 603.21 and 603.22 identify the endpoint from the start ledger, finite schedule, and replay row, and place every step in the displayed E1/E0 schedule.

Finally apply observer-locality transport, Definition 603.17. Observer-facing transport reads only the local packet and its no-global-clock ledger, so the endpoint is preserved by componentwise hsame and no nonlocal time coordinate appears. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier hsame schedule endpoint transport`

Theorem 603.24 (HistTimeStream local transport exhaustion)**.** *For accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packets* $Q = (H, T, S, C, L, P, N)$ *and* $Q' = (H', T', S', C', L', P', N')$*, componentwise* hsame *transport preserves the local* Cont *prefix of* Q *as the local* Cont *prefix of* Q'*. The transport exhausts the displayed local rows and does not add a global clock, a host time index, an inter-Hist synchronization coordinate, or a temporal-rate row.*

Proof. First expose the prefix. The Cont-prefix exactness theorem, Definition 603.18, identifies the local route from e to H through T, S, C, L, P, N.

Now transport componentwise. The hsame prefix transport theorem, Definition 603.19, preserves this prefix across accepted packets, and schedule classifier stability keeps the transported schedule finite.

Finally apply observer-locality transport and local clock refusal. Theorem 603.17 keeps the transported packet local for observer-facing consumers, while Definition 603.26 rejects every nonlocal clock read unless it has already factored through the displayed rows. Hence the transport is exhausted by the local packet. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier local transport exhaustion`

Theorem 603.25 (HistTimeStream NameCert route exhaustion)**.** *Every* NameCert *route of an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$ *is exhausted by the carrier, finite step ledger, replay route, no-global-clock ledger, provenance row, and local naming row. The route cannot pass to a global clock, host time index, inter-Hist synchronization row, temporal-rate coordinate, or hidden observer route.*

Proof. Begin with the NameCert obligation sketch and the carrier. The sketch fixes the local naming obligation on the displayed packet, and Definition 603.1 supplies exactly the rows H, T, S, C, L, P, N.

Next assemble the local route. Observation carrier gives the finite prefix rows, schedule classifier stability keeps the step ledger finite, and ledger exhaustion keeps replay, no-global-clock locality, provenance, and naming in the same packet.

Finally pass through the observer-locality surface. The observer-locality lattice link, public export, bridge route, observer-locality transport, and hsame prefix transport show that every downstream route reads the same local packet. Therefore the NameCert surface is exhausted by those rows and has no escape to nonlocal time structure. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamNameCertRouteExhaustion`

Theorem 603.26 (HistTimeStream local clock refusal)**.** *Let*

$$Q = (H, T, S, C, L, P, N)$$

be an accepted $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Any attempted global-clock, host-indexed time, inter-Hist synchronization, or temporal-rate read is refused unless it factors through the displayed local tuple* H, T, S, C, L, P, N *and the no-global-clock ledger carried by* L*.*

Proof. First expose the public packet. The public export theorem, Definition 603.7, makes H, T, S, C, L, P, N the entire local-time stream surface and excludes global-clock and hidden-route coordinates from the export.

Next use hsame prefix transport. Theorem 603.19 shows that replay across componentwise transports preserves the local prefix and cannot manufacture host-indexed time or inter-Hist synchronization.

Finally apply observer-locality transport, Definition 603.17. Since observer-locality consumers receive only the displayed local stream and its no-global-clock ledger, every nonlocal clock read is refused unless it has already factored through H, T, S, C, L, P, N. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier local clock refusal`

Theorem 603.27 (HistTimeStream prefix transport exhaustion)**.** *For any accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, every consumer read through* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *or* $\mathsf{ObserverLocalityTransport}$ *first factors through the* Cont*-prefix route of Definition 603.18, then through the componentwise* hsame *transport of Definition 603.19. Any attempted global-clock, host-indexed time, inter-Hist synchronization, or temporal-rate read is refused by Definition 603.26 unless it preserves the displayed rows* H, T, S, C, L, P, N *and the observer-locality boundary of Definition 603.17.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier prefix transport exhaustion`

Proof. Begin with observer-locality transport. Theorem 603.17 says that observer-history and observer-locality consumers receive exactly the displayed local-time rows H, T, S, C, L, P, N.

Next expose the prefix route. Theorem 603.18 identifies that consumer surface as the displayed Cont prefix from e to the source Hist row H through the schedule, start ledger, replay, transport, provenance, and local naming rows.

Now transport the prefix componentwise. Theorem 603.19 preserves the same prefix across accepted packets and cannot create a host-indexed time coordinate, inter-Hist clock, or global synchronization row.

Finally apply local clock refusal. Theorem 603.26 rejects global-clock, rate, and nonlocal time reads unless they have already passed through and preserved the displayed packet rows. Thus the four cited theorems assemble one consumer-facing exhaustion route. □

Theorem 603.28 (HistTimeStream StreamName finite-window handoff)**.** *Let*

$$Q = (H, T, S, C, L, P, N)$$

be an accepted $\mathsf{HistTimeStream}^{\uparrow}$ *packet. The displayed* Cont*-prefix route of* Q *induces the finite* $\mathsf{StreamName}^{\uparrow}$ *window read by chapter 126: the source row* H*, finite schedule* T*, start ledger* S*, replay row* C*, componentwise* hsame *ledger* L*, package row* P*, and local name row* N *are passed as one bounded naming packet before any countable-name consumer may read the stream. The handoff carries no quotient stream, no selected host limit, no ambient completeness witness, and no host-indexed time coordinate.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier streamname finite window handoff`

Proof. Start from Cont-prefix exactness, Definition 603.18. It identifies the local stream as the finite route from e to the displayed source row H through T, S, C, L, P, N.

Next apply hsame prefix transport, Definition 603.19. The transport keeps the same displayed finite route when the packet is replayed componentwise, so the $\mathsf{StreamName}^{\uparrow}$ consumer of chapter 126 receives a finite window rather than an ambient clock.

Finally use local clock refusal, Definition 603.26. Since every nonlocal time read must already factor through H, T, S, C, L, P, N, the handoff to the StreamName surface cannot add a quotient stream, chosen limit, completeness witness, or host-indexed time row. □

Theorem 603.29 (HistTimeStream StreamName schedule classifier boundary)**.** *For accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packets* $Q = (H, T, S, C, L, P, N)$ *and* $Q' = (H', T', S', C', L', P', N')$*, componentwise* hsame *transport preserves the finite schedule and replay classifier before any* $\mathsf{StreamName}^{\uparrow}$ *read. The transported classifier remains the displayed boundary over* T, S, C, L, P, N *and cannot add a global clock, host-indexed time, inter-Hist synchronization row, or temporal-rate coordinate.*

Proof. First expose the finite classifier boundary. By Definition 603.18, the only route available to a consumer is the displayed Cont prefix through the schedule, start ledger, replay row, transport ledger, package row, and local name.

Now transport the packet. Theorem 603.19 preserves that prefix componentwise from Q to Q', so the schedule observed by chapter 126 is still the finite schedule carried by the HistTimeStream packet.

Finally apply local clock refusal, Definition 603.26. A global clock or host-indexed time row would have to appear before the StreamName read, but the refusal theorem permits only rows that already factor through the displayed local tuple. Therefore the schedule classifier boundary is stable under componentwise transport. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier streamname schedule classifier boundary`

Theorem 603.30 (HistTimeStream real-completion window nonescape)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the finite window induced by an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet factors through the* $\mathsf{HistTimeStream}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of chapter 603, chapter 126, chapter 301, and chapter 38. The consumer receives only the displayed finite window and no quotient stream, selected host limit, ambient completeness object, host-indexed time, or cross-Hist clock.*

Proof. Begin with the HistTimeStream window. Theorem 603.18 supplies the finite Cont prefix, and Definition 603.28 identifies it as the window handed to the StreamName naming surface.

Next pass through the downstream rows. The StreamName chapter, chapter 126, supplies the countable-name surface; the RegSeqRat chapter, chapter 301, consumes such naming data as regular rational sequence input; and the Real chapter, chapter 38, reads the resulting real-number surface only after those rows have been traversed.

It remains to exclude escape data. The hsame prefix transport theorem, Definition 603.19, keeps the window componentwise local, while local clock refusal, Definition 603.26, blocks quotient streams, selected host limits, ambient completeness witnesses, host-indexed time, and cross-Hist clocks unless they already appear in H, T, S, C, L, P, N. They do not appear there, so the real-completion consumer remains inside the four displayed rows. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier real completion window nonescape`

Theorem 603.31 (HistTimeStream start-ledger totality)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. The local stream starts at* e*, reaches only the displayed source row H through the finite schedule T and start ledger S, and reads no external time index.*

Proof. Start with the carrier definition, Definition 603.1. It exposes H, T, S, C, L, P, N as the whole packet and places the start-at-empty ledger inside S.

Apply Cont prefix exactness, Definition 603.18. The displayed prefix runs from e to the source Hist row H through the finite schedule and replay rows, so no outside time index is available along the route.

Finally use the observer-locality bridge route, Definition 603.16. Downstream observer-history consumers receive the same local stream only after the displayed rows have been read, which preserves start-ledger totality for Q. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier start ledger totality`

Theorem 603.32 (HistTimeStream Cont replay determinacy)**.** *For accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packets $Q = (H, T, S, C, L, P, N)$ and $Q' = (H', T', S', C', L', P', N')$, the schedule row T and replay row C determine the local prefix route up to componentwise* hsame*. Under componentwise* hsame *transport from Q to Q', the replayed prefix remains the displayed route from* e *to the transported source row and cannot branch through a hidden clock row.*

Proof. Begin with Cont prefix exactness. Theorem 603.18 makes the local prefix a route determined by the displayed schedule, start ledger, replay row, locality row, provenance row, and local name.

Next apply hsame prefix transport, Definition 603.19. Componentwise hsame carries the displayed prefix from Q to Q' while preserving the local replay route.

Finally use the observer-locality bridge route. Since downstream observer-history and observer-locality consumers read only the transported local stream, the schedule and replay rows determine the consumer-facing prefix without adding an inter-Hist clock or hidden global time coordinate. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier cont replay determinacy`

Theorem 603.33 (HistTimeStream no-global-clock nonescape)**.** *Every observer-history consumer of an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet may read the displayed local stream H, T, S, C, L, P, N, but it cannot obtain inter-Hist synchronization, ambient clock equality, temporal-rate data, or a host time index from that packet.*

Proof. First expose the local stream through the carrier definition and Cont prefix exactness. These identify the observer-facing data as the finite prefix from e to H with schedule, replay, locality, provenance, and naming rows.

Now transport along componentwise hsame using Definition 603.19. The transported prefix remains local and does not create a host-indexed time coordinate or inter-Hist clock.

Finally apply the observer-locality bridge route, Definition 603.16. The sibling observer-history surfaces consume this packet only as a local stream, so ambient clock equality, temporal-rate data, and global synchronization remain outside the export. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier no global clock nonescape`

Theorem 603.34 (HistTimeStream source observation totality)**.** *Let $Q = (H, T, S, C, L, P, N)$ be an accepted* HistTimeStream$^{\uparrow}$ *packet. Before any observer-history, observer-locality,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, or* Real$^{\uparrow}$ *consumer reads Q, the packet exposes exactly the source row H, finite schedule T, start ledger S, and replay row C through the displayed* Cont *prefix, with L, P, N recording only componentwise transport, package provenance, and local naming for that same finite observation.*

Proof. Start with the carrier definition, Definition 603.1. It displays H, T, S, C, L, P, N as the whole accepted packet and places the source observation, finite schedule, start ledger, and replay row in explicit coordinates.

Next apply Cont prefix exactness, Definition 603.18. The consumer route is the finite prefix from e to H through T, S, C, L, P, N, so no observer can read a source observation before these rows have been exposed.

Finally use the StreamName finite-window handoff and real-completion window nonescape, Definition 603.28 and Definition 603.30. These route every StreamName, RegSeqRat, and Real read through the same displayed finite window, so the source observation is total for all downstream consumers named in the statement. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier source observation totality`

Theorem 603.35 (HistTimeStream RegSeqRat reindex stability)**.** *For accepted* HistTimeStream$^{\uparrow}$ *packets $Q = (H, T, S, C, L, P, N)$ and $Q' = (H', T', S', C', L', P', N')$, componentwise* hsame *transport from Q to Q' preserves the finite window handed to* StreamName$^{\uparrow}$ *and therefore the regular rational sequence input read by* RegSeqRat$^{\uparrow}$*. The reindexed consumer sees only the transported rows $H', T', S', C', L', P', N'$ and cannot acquire a quotient stream, selected host limit, ambient completeness witness, host-indexed time, or cross-Hist clock.*

Proof. First expose the finite window by Cont prefix exactness and the StreamName handoff, Definition 603.18 and Definition 603.28. They identify the consumer-visible window as the displayed route through H, T, S, C, L, P, N before StreamName naming is read.

Now transport the packet componentwise. The hsame prefix transport theorem, Definition 603.19, carries the same finite route from Q to Q', so the StreamName surface receives the transported finite window rather than a newly chosen index source.

Finally pass to the RegSeqRat consumer. The real-completion window nonescape theorem, Definition 603.30, already routes RegSeqRat reads through HistTimeStream and StreamName; local clock refusal, Definition 603.26, blocks the listed escape data. Hence reindexing by componentwise hsame is stable for the RegSeqRat input. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regseqrat reindex stability`
↪ variant `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regseq reindex stability`

Theorem 603.36 (HistTimeStream Real readback ledger)**.** *Any* $\mathsf{Real}^{\uparrow}$ *readback of an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet factors through the HistTimeStream rows* H, T, S, C, L, P, N*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence consumer, and the* $\mathsf{Real}^{\uparrow}$ *row. The readback ledger cannot add quotient streams, selected host limits, ambient completeness objects, host-indexed time, or cross-Hist clock data.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier real readback ledger`

Proof. Begin with source observation totality, Definition 603.34. Every downstream read has already exposed the source row, finite schedule, start ledger, and replay row inside the displayed packet.

Next use RegSeqRat reindex stability, Definition 603.35. Componentwise hsame transport preserves the finite window handed to StreamName and then to RegSeqRat, so the regular-sequence input seen by a Real consumer remains tied to the HistTimeStream packet.

Finally apply real-completion window nonescape, Definition 603.30. It forces the Real readback through the HistTimeStream, StreamName, RegSeqRat, and Real rows named in the statement and excludes quotient streams, selected host limits, ambient completeness objects, host-indexed time, and cross-Hist clock data from the ledger. □

Theorem 603.37 (HistTimeStream StreamName window totality)**.** *Let* $Q = (H, T, S, C, L, P, N)$ *be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Every* $\mathsf{StreamName}^{\uparrow}$*-facing consumer of* Q *reads exactly the displayed finite window rows* H, T, S, C, L, P, N *through the carrier, the finite* Cont *prefix, and the StreamName finite-window handoff. Such a consumer cannot obtain a quotient stream, host-indexed clock, hidden route, selected host limit, ambient completeness witness, or cross-Hist clock from the packet.*

Proof. Start with the carrier definition, Definition 603.1. It displays Q as precisely the seven rows H, T, S, C, L, P, N, with no coordinate for a quotient stream, hidden route, host-indexed clock, selected limit, or ambient completeness witness.

Next apply the StreamName finite-window handoff, Definition 603.28. It identifies the consumer-visible naming input as the bounded window induced by the displayed Cont prefix before any StreamName consumer reads the stream.

Finally use prefix transport exhaustion, Definition 603.27. Every observer-facing or naming-facing read has already passed through the finite prefix and componentwise hsame transport, and every attempted nonlocal time read is refused unless it preserves the displayed rows. Hence the StreamName-facing window is total and admits no hidden quotient, clock, route, limit, or completeness coordinate. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier streamname window totality`

Theorem 603.38 (HistTimeStream RegSeqRat Real window route)**.** *Let* $Q = (H, T, S, C, L, P, N)$ *be an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read of the packet factors through* $\mathsf{HistTimeStream}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *in that order. The route reads only the finite window rows displayed by* Q *and cannot select a host limit, ambient completeness coordinate, quotient stream, host-indexed time, or cross-Hist clock.*

Proof. First apply StreamName window totality, Definition 603.37. It fixes the StreamName-facing input as the displayed finite window H, T, S, C, L, P, N and excludes hidden quotient, route, clock, limit, and completeness coordinates at the naming boundary.

Next use RegSeqRat reindex stability, Definition 603.35. The regular rational sequence consumer receives only the transported finite window handed through StreamName, so reindexing cannot choose a host limit or add an ambient completeness coordinate.

Finally apply real-completion window nonescape, Definition 603.30. It places each Real read after the HistTimeStream, StreamName, and RegSeqRat rows and blocks all listed escape data from the Real-facing route. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regseq real window route`

Theorem 603.39 (HistTimeStream finite-window readback exhaustion)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, the local source rows, StreamName window, RegSeqRat reindex stability, and Real readback ledger exhaust the finite-window readback obligations on the Real and Completion path. No further readback obligation remains outside the displayed local source rows, the StreamName window totality surface, the RegSeqRat route, and the Real ledger.*

Proof. Begin with StreamName window totality, Definition 603.37. It turns the local source rows of Q into the whole naming window available to a StreamName-facing consumer.

Next apply the RegSeqRat Real window route, Definition 603.38. This routes every RegSeqRat or Real read through HistTimeStream, StreamName, RegSeqRat, and Real in order, while preserving the finite window and excluding nonlocal escape data.

Finally use the Real readback ledger, Definition 603.36. The ledger records the same source rows, StreamName handoff, RegSeqRat input, and Real readback as the complete path, so the finite-window readback obligations for the Real and Completion path are exhausted by the three named surfaces. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier finite window readback exhaustion`

Theorem 603.40 (HistTimeStream NameCert root-unblock package)**.** *The* $\mathsf{HistTimeStream}^{\uparrow}$ *carrier exposure, finite schedule stability,* Cont*-prefix exactness, local-clock refusal,* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *reindex stability, and* $\mathsf{Real}^{\uparrow}$ *readback ledger form the root-unblock package for every downstream finite-window consumer of an accepted packet* $Q = (H, T, S, C, L, P, N)$*. The package exposes only the displayed local rows, componentwise* hsame *transport,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and it exports no quotient stream, selected host limit, ambient completeness witness, host-indexed time, or cross-Hist clock.*

Proof. Project the packet through the carrier, Definition 603.1. The rows H, T, S, C, L, P, N are the whole accepted surface, so the root consumer starts from the displayed source row, finite schedule, continuation routes, package row, and local naming row.

Next combine schedule stability, prefix exactness, and local-clock refusal. The stability and prefix theorems, Definition 603.4 and Definition 603.18, make the finite route from e through H, T, S, C, L, P, N the only consumer-visible prefix; Definition 603.26 blocks any host-indexed or cross-Hist clock read.

Finally attach the downstream readback surfaces. The StreamName handoff, RegSeqRat stability, and Real ledger, Definition 603.28, Definition 603.35, and Definition 603.36, consume the same finite window in order. Hence the listed rows are exactly the root-unblock package for finite-window consumers and no excluded host or quotient data is introduced. □

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier namecert root unblock package`

Theorem 603.41 (HistTimeStream local stream-prefix factorization)**.** *Every accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *consumer route factors first through the local prefix*

$$H, T, S, C, L, P, N$$

before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read is available. The prefix is the displayed* Cont *route of the packet, not a hidden stream quotient, host clock, selected limit, or ambient completion coordinate.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier local stream prefix factorization`

Proof. Read the route from the carrier of Definition 603.1. The carrier has no coordinate outside the seven displayed rows, so any accepted consumer must begin by projecting those rows.

Apply Cont prefix exactness, Definition 603.18. It identifies the consumer path as the finite prefix ending in the displayed packet rows before a naming, sequence, or real-facing consumer can read the packet.

Use StreamName window totality and the RegSeqRat Real window route, Definition 603.37 and Definition 603.38. They place StreamName, RegSeqRat, and Real reads strictly after the local prefix and preserve the same rows throughout the downstream route. The absence of extra carrier coordinates gives the stated exclusions. □

Theorem 603.42 (HistTimeStream StreamName Real consumer route)**.** *The ordered downstream route*

$$\mathsf{HistTimeStream}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

preserves the no-global-clock ledger for the packet $Q = (H, T, S, C, L, P, N)$*. Along this route every read is through the local prefix, StreamName finite-window handoff, RegSeqRat reindex surface, and Real readback ledger, and the route excludes quotient streams, selected host limits, ambient completeness witnesses, host-indexed time, and cross-Hist clocks.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier streamname real consumer route`

Proof. First factor the consumer through the local prefix by Definition 603.41. The route has already exposed the finite HistTimeStream rows before any downstream naming or readback surface is used.

Next pass through StreamName and RegSeqRat. StreamName window totality, Definition 603.37, fixes the naming input as the same finite rows, and RegSeqRat reindex stability, Definition 603.35, keeps the regular rational sequence consumer tied to componentwise hsame transport of those rows.

Finally apply local-clock refusal and the Real readback ledger, Definition 603.26 and Definition 603.36. The refusal theorem preserves the no-global-clock ledger, while the Real ledger records the ordered HistTimeStream, StreamName, RegSeqRat, Real route and rules out every listed nonlocal or quotient datum. □

Theorem 603.43 (HistTimeStream StreamName tail scope)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, every* $\mathsf{StreamName}^{\uparrow}$ *tail read induced by* Q *factors through the displayed finite rows*

$$H, T, S, C, L, P, N$$

and through the finite-window handoff of Definition 603.28. The tail read may consume only that bounded naming window and cannot add a quotient stream, host index, selected limit, ambient completeness witness, global clock, or cross-Hist time coordinate.

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier streamname tail scope`

Proof. Project Q through the carrier of Definition 603.1. The carrier exposes exactly the rows H, T, S, C, L, P, N, so a tail consumer has no source coordinate beyond the displayed packet.

Apply the local prefix factorization and the StreamName finite-window handoff, Definition 603.41 and Definition 603.28. Together they force every StreamName-facing tail read to pass through the finite Cont route before the naming surface is read.

Finally use real-completion window nonescape, the Real readback ledger, and the StreamName Real consumer route, Definition 603.30, Definition 603.36, and Definition 603.42. These downstream ledgers preserve the same finite rows and exclude quotient, host, limit, completeness, and clock data, so the StreamName tail scope is exactly the stated bounded window. □

Theorem 603.44 (HistTimeStream RegSeqRat window exactness)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of the induced downstream window receives exactly the local finite window together with the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity obligations already routed through componentwise* hsame *transport. It receives no selected host limit, quotient stream representative, ambient completeness witness, global clock, or Real completeness proof.*

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regseqrat window exactness`

Proof. Start with the StreamName tail scope of Definition 603.43. It fixes the input to the RegSeqRat-facing consumer as the finite window already exposed by HistTimeStream and StreamName.

Next apply RegSeqRat reindex stability and the RegSeqRat Real window route, Definition 603.35 and Definition 603.38. These theorems identify the sequence-facing obligation as regular rational reindexing over the transported finite rows, not as a choice of a host limit or completion object.

The exclusion clauses follow from real-completion window nonescape, the Real readback ledger, and the StreamName Real consumer route, Definition 603.30, Definition 603.36, and Definition 603.42. Each cited route keeps the RegSeqRat window inside the same local rows and blocks every listed nonlocal datum. □

Theorem 603.45 (HistTimeStream Real readback scope lock)**.** *For an accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet* $Q = (H, T, S, C, L, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *readback on this consumer route must traverse*

$$\mathsf{HistTimeStream}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The readback may inspect only the certified real surface produced by the carrier, StreamName handoff, RegSeqRat window, and Real readback ledger. It cannot inspect Real completeness, a quotient class, a selected host limit, host-indexed time, global time, or any cross-Hist clock.

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier real readback scope lock`

Proof. First expose the packet through the carrier and local prefix factorization, Definition 603.1 and Definition 603.41. This places every downstream read after the finite HistTimeStream rows.

Then pass through the two intermediate consumers. The StreamName tail scope and RegSeqRat window exactness, Definition 603.43 and Definition 603.44, identify the only naming and rational-sequence surfaces that can be read before RealUp.

Finally apply the Real readback ledger and the StreamName Real consumer route, Definition 603.36 and Definition 603.42. Their route is the displayed HistTimeStream, StreamName, RegSeqRat, Real path, and their exclusions block Real completeness, quotient classes, host limits, host-indexed time, global time, and cross-Hist clocks from the readback scope. □

Theorem 603.46 (HistTimeStream regular witness-ledger route)**.** *Every accepted* $\mathsf{HistTimeStream}^{\uparrow}$ *packet consumed by a regular Cauchy modulus witness route first passes through the local finite window and then through the ordered downstream surface*

$$\mathsf{HistTimeStream}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The route displays the finite HistTimeStream rows, the StreamName tail window, the RegSeqRat regular readback, the regular Cauchy modulus witness ledger, and the RealUp seal rows in that order. It has no host clock, quotient stream, selected limit, ambient completeness coordinate, or cross-Hist time parameter.

Lean4 ✓ `BEDC.Derived.HistTimeStreamUp.HistTimeStreamCarrier regular witness ledger route`

Proof. First expose the accepted packet through the local carrier and finite-prefix route. The carrier and local prefix factorization place every downstream consumer after the displayed HistTimeStream rows.

Apply the StreamName tail scope, Definition 603.43. It fixes the naming input as the finite window already carried by the HistTimeStream packet and excludes host-clock or quotient-stream expansion.

Next apply RegSeqRat window exactness, Definition 603.44. The RegSeqRat consumer receives exactly that finite window together with its regular rational-sequence obligations, not a selected limit or ambient completion datum.

Finally combine the Real readback scope lock with the RegularCauchyModulusWitnessLedger consumer-totality theorem, Definition 603.45 and Definition 977.17. The Real-facing read must pass through the HistTimeStream, StreamName, RegSeqRat, witness-ledger, and Real seal rows, so the witness route remains a finite paper-side consumer route rather than an external completeness or host-clock assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HistTimeStream}^{\uparrow}$ packet is generated by a source Hist row, a finite sequence of E1/E0 step rows, a start-at-empty ledger, displayed Cont routes, componentwise hsame transport, no-global-clock locality, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge surface records the carrier, step-ledger exactness, route stability, componentwise hsame transport, no-global-clock locality, package provenance, local NameCert surface, public export, observer-locality consumption bridge, observer-locality bridge route, observer-locality transport theorem, Cont-prefix exactness, hsame prefix transport, and local-clock refusal for one Hist time stream.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HistTimeStream}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not define inter-Hist synchronization, host-indexed time, temporal rates for analytic moduli, or Lean verification for the packet.

Upgrade path: No further paper-side closure grade remains; Lean verification and bridge checking are tracked only on the verification axis.

Chapter 604

A Concrete Naming Certificate for $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$

The $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$ surface turns the supply position and inscription point of chapter 9393 into a concrete BEDC packet. It records the named supply socket, the local inscription site, the finite GAP boundary at which a forward-binding belief is visible, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows that keep the object auditable. The packet names only the structural socket. It does not assert a positive predicate of the supply position, does not internalize induction, and does not add an axiom to the substrate.

The inscription-point row I is consumed from chapter 610 before the supply boundary is read, while this chapter adds the external supply socket and GAP boundary around that local inscription site.

Definition 604.1 (TranscendentalSupplyInscription carrier)**.** *A* $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, S, I, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the named transcendental supply position of Definition 9393.11, recorded only as an external socket and not as a substrate object;*
2. *S is the closed observational system in which a forward-binding belief is exposed as a BEDC-visible boundary;*
3. *I is the inscription point of Definition 9393.15, namely the local site at which the supply socket is read by S;*
4. *G is the GAP-boundary row that records the unavailable internal discharge of the forward-binding claim;*
5. *H records componentwise* hsame *transports for the supply-position name, closed-system row, inscription-point row, GAP boundary, package provenance, and local naming row;*
6. *C records the displayed* Cont *routes that read S, expose I, open the GAP boundary G, and keep T visible only as the far end of that boundary;*
7. *P is the* Pkg *provenance over* BHist, Gap, hsame, Cont, *and* $\mathsf{NameCert}$ *for the displayed socket;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyInscription}^{\uparrow}}$ *row.*

The carrier is exactly this finite supply-inscription socket. It has no row for a substrate-level induction rule, no internal witness for the universal claim marked by G, no selected content of T, no host equality test on supply positions, and no ambient completeness principle.

Lean4 (definition) `BEDC.Derived.TranscendentalSupplyInscriptionUp.TranscendentalSupplyInscriptionUp`
Lean4 ✓ `BEDC.Derived.TranscendentalSupplyInscriptionUp.TranscendentalSupplyInscriptionTasteGate`
`single carrier alignment`

Theorem 604.2 (TranscendentalSupplyInscription NameCert obligations)**.** *For the carrier of Definition 604.1, the local* $\mathsf{NameCert}_{\mathsf{TranscendentalSupplyInscription}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier exposure: every accepted object exposes the tuple $X = (T, S, I, G, H, C, P, N)$ and no additional supply coordinate;*
2. *classifier transport: comparison transports the supply-position name, closed-system row, inscription-point row, GAP boundary, provenance row, and local naming row by the displayed* hsame *data;*
3. *inscription route: every consumer read of the local inscription point factors through S, then through I, and only then reaches the GAP boundary G;*
4. *package exactness: the provenance row P lists the full BEDC inventory needed for the socket and exposes no internal induction rule, selected supply content, host equality, trichotomy, or completeness datum;*
5. *supply boundary: the row T is readable only as the external position named at the far end of G, never as an object that can be consumed by a substrate derivation.*

Proof. Carrier exposure is projection from Definition 604.1. The supply name T, closed-system row S, inscription point I, GAP boundary G, transports, continuation routes, provenance, and local name are coordinates of one finite BHist packet.

Classifier transport is componentwise. The stored hsame rows in H compare exactly the displayed coordinates, and the Cont rows in C replay the same closed-system, inscription, GAP, provenance, and naming routes after transport.

For the inscription route, a consumer first reads the closed observational system S. The only local site then exposed is I, and the only boundary reached from I is the GAP row G that records the unavailable internal discharge.

Package exactness follows from the inventory of P. Since P names only the displayed BEDC rows, an internal induction rule, selected content of T, host equality test, trichotomy branch, or completeness principle would require an extra coordinate absent from X.

Finally, the supply boundary is the route S to I to G to the external position named by T. The packet therefore makes the socket auditable without turning the far end of the socket into a substrate object. □

Theorem 604.3 (TranscendentalSupplyInscription non-escape boundary)**.** *Every consumer of an accepted* $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$ *packet factors through the finite route*

$$S \to I \to G \to H \to C \to P \to N.$$

The consumer may name T as the external supply position attached to the GAP boundary, but it cannot read T as internal mathematical content, an induction rule, a selected witness, host equality, or ambient completeness.

Proof. Project the accepted carrier $X = (T, S, I, G, H, C, P, N)$. The only closed-system coordinate is S, the only local inscription coordinate is I, and the only exposed unresolved boundary is G.

Apply Definition 604.2. Its inscription route sends every consumer through S and I before G, while its supply boundary permits T only as the named far end of that GAP boundary.

The package exactness row rules out the unavailable structure. Therefore every consumer remains inside the displayed packet and treats the supply position as an external socket endpoint rather than as an internally consumable rule or witness. □

Theorem 604.4 (TranscendentalSupplyInscription classifier transport)**.** *For every accepted* TranscendentalSupplyIn *carrier*

$$X = (T, S, I, G, H, C, P, N),$$

classifier comparison is transported only along the displayed hsame *row H and replayed only through the displayed* Cont *row C. The transport preserves the supply-position name, closed-system row, inscription-point row, GAP boundary, package row, and local* NameCert *row without adding substrate induction, selected supply content, host equality, or ambient completeness.*

Proof. Project the carrier. The compared rows are exactly T, S, I, G, H, C, P, N, and the only componentwise comparison row is H.

Apply the NameCert obligations. Their classifier-transport clause states that the supply-position name, closed-system row, inscription-point row, GAP boundary, provenance row, and local naming row are compared by the displayed hsame data and replayed by the displayed continuation routes.

The excluded objects follow from the carrier boundary. Since X contains no coordinate for substrate induction, selected supply content, host equality, or ambient completeness, classifier transport cannot acquire any of those objects during comparison. □

Theorem 604.5 (TranscendentalSupplyInscription package exactness)**.** *The* Pkg *row P of an accepted* TranscendentalSupplyInscription$^{\uparrow}$ *packet lists exactly the displayed* BHist, Gap, hsame, Cont, Pkg, NameCert, *and inscription-point rows needed to expose the route*

$$S \to I \to G \to T.$$

It does not package an internal induction rule, a selected witness of T, host equality on supply positions, trichotomy, or a completeness principle.

Proof. Read the package coordinate in the carrier definition. The row P names the finite BEDC inventory for the displayed socket and is attached to the same tuple $X = (T, S, I, G, H, C, P, N)$.

Use the package-exactness clause of Definition 604.2. It identifies P with the provenance over the socket rows rather than with an internal proof rule or a selected supply object.

Finally apply non-escape from Definition 604.3. Consumers can reach T only as the external supply position behind the GAP boundary, so the package cannot export host equality, trichotomy, completeness, or a substrate-level induction principle. □

Closure status for NameCert$_{\text{TranscendentalSupplyInscription}^{\uparrow}}$.

Constructive story: A TranscendentalSupplyInscription$^{\uparrow}$ packet is generated from a named supply-position socket, a closed observational system row, a local inscription-point row, a GAP boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.TranscendentalSupplyInscriptionUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{TranscendentalSupplyInscription}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 604.1, Definition 604.2, Definition 604.3, Definition 604.4, and Definition 604.5 bind the bridge-facing supply-boundary route through S, I, G, H, C, P, N for the same supply-inscription socket.

Formal verification: FormalStatus(NameCert$_{\text{TranscendentalSupplyInscription}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.TranscendentalSupplyInscriptionUp.TranscendentalSupplyInscriptionTasteGate\_single\_car`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close mathematical induction, positive predicates of the supply position, selected supply content, host equality of supply positions, ambient completeness, or machine bridge checking.
Upgrade path: The theory axis is complete for the displayed supply-inscription socket; machine bridge checking remains the separate verification-axis route over BHist, Gap, hsame, Cont, Pkg, NameCert, and inscription-point rows.

Chapter 605

A Concrete Naming Certificate for $\mathsf{StationarySealUniqueness}^{\uparrow}$

The $\mathsf{StationarySealUniqueness}^{\uparrow}$ surface isolates the finite seal packet obtained when a stationary rational classifier is transported through the constant $\mathsf{StreamName}^{\uparrow}$ row, the finite $\mathsf{RegSeqRat}^{\uparrow}$ consumption row, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ finite-window boundary. It is a naming certificate for the four public readings of the same stationary seed, not a host-real equality assertion or a completeness principle.

As a concrete-instance dependency, $\mathsf{StationarySealUniqueness}^{\uparrow}$ is the four-reading finite-window specialization of chapter 701. It consumes the $\mathsf{Rat}^{\uparrow}$ seed classifier, constant $\mathsf{StreamName}^{\uparrow}$ row, stationary $\mathsf{RegSeqRat}^{\uparrow}$ diagonal row, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and $\mathsf{Real}^{\uparrow}$ finite-window classifier before exposing the same selected-window seal.

Definition 605.1 (Stationary seal uniqueness carrier)**.** *A* $\mathsf{StationarySealUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, S, G, D, R, F, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *stationary seed row, recording two rational history carriers and their transported* $\mathsf{RatHistoryClassifier}$ *row;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *row read from Q;*
3. *G is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row that reads the same constant stream through a displayed observation window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for the finite window;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier row obtained from the same stationary constant-stream data;*
6. *F records the selected finite window and the four public projections from U to D, S, G, R;*
7. *H stores componentwise* hsame *transports for the rational seed, stream row, regular-sequence seal row, dyadic ledger, real classifier row, window selector, continuation routes, package provenance, and local name;*
8. *C stores the displayed* Cont *routes from Q to S, from S to G, from S to D, and from D, S, G to the real-facing row R;*

9. *P records* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{StationarySealUniqueness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-real equality, selected host limits, non-stationary tails, ambient completeness, or an external comparison relation beyond the displayed four readings.

Theorem 605.2 (Stationary seal uniqueness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationarySealUniqueness}^{\uparrow}}$ *surface for Definition 605.1 consists exactly of carrier admission for* $U = (Q, S, G, D, R, F, H, C, P, N)$*,* $\mathsf{Rat}^{\uparrow}$ *seed exposure, constant* $\mathsf{StreamName}^{\uparrow}$ *readback, finite* $\mathsf{RegSeqRat}^{\uparrow}$ *seal consumption,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation-ledger exposure,* $\mathsf{Real}^{\uparrow}$ *finite-window classifier exposure, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet U. The rows Q, S, G, D, R, F supply the rational seed, constant stream, regular-sequence seal consumption, dyadic ledger, real finite-window classifier, and selected window.

The stationary seed obligation is read from Q: it contains the two rational carriers and the RatHistoryClassifier row. The constant $\mathsf{StreamName}^{\uparrow}$ obligation is read from S by replaying the continuation route from Q to the constant stream.

The finite-window obligations are then forced by the packet order. Row G consumes the selected window as a $\mathsf{RegSeqRat}^{\uparrow}$ seal row, D records the dyadic observations read from that same window, and R exposes the real-facing classifier only after D, S, G have already been displayed.

The remaining data are routing and naming data. Rows H, C, P, N preserve the listed coordinates, replay the finite reads, record the dependency packages, and name the stationary seal packet without adding quotient-real equality, host completeness, or an external real comparison. □

Theorem 605.3 (Stationary seal four-row exhaustion)**.** *Let* $U = (Q, S, G, D, R, F, H, C, P, N)$ *be an accepted* $\mathsf{StationarySealUniqueness}^{\uparrow}$ *carrier. Every certified finite observation window exported by U has exactly the four public readings D, S, G, R: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger, the constant* $\mathsf{StreamName}^{\uparrow}$ *classifier, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row, and the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier. These four readings are all transported from the same* $\mathsf{Rat}^{\uparrow}$ *seed row Q, and any two accepted seal packets with the same Q, F, H, C, P, N expose the same four-row public surface.*

Proof. Start from the accepted carrier and project the rows $Q, S, G, D, R, F, H, C, P, N$. The selected window is recorded in F, and the only continuation routes out of the stationary seed are the routes displayed in C.

Read the stream-facing boundary first. The route from Q to S exposes the constant $\mathsf{StreamName}^{\uparrow}$ classifier induced by the rational seed, so every later finite-window read is already tied to the same RatHistoryClassifier row.

Next read the two finite observation rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row G consumes the selected constant-stream window, while the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the dyadic ledger for that same window. Since both rows are reached through the same selector F and the same continuation routes C, neither can introduce an independent seal coordinate.

Finally read the real-facing row R. The $\mathsf{Real}^{\uparrow}$ finite-window classifier receives only D, S, G together with the transport, provenance, and local name rows H, P, N. Thus the public surface of the seal is exhausted by D, S, G, R, and two carriers agreeing on Q, F, H, C, P, N replay the same four readings. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationarySealUniqueness}^{\uparrow}}$ **.**

Constructive story: A StationarySealUniqueness$^{\uparrow}$ packet is generated from a stationary Rat$^{\uparrow}$ seed classifier, a constant StreamName$^{\uparrow}$ row, a finite RegSeqRat$^{\uparrow}$ seal-consumption row, a DyadicRatCore$^{\uparrow}$ observation ledger, a Real$^{\uparrow}$ finite-window classifier, a selected-window projection row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{StationarySealUniqueness}^{\uparrow}}$, seed)

Scope closed: Definition 605.1, Definition 605.2, and Definition 605.3 bind the stationary finite-window carrier, local naming obligations, and four-reading exhaustion surface over Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{StationarySealUniqueness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-real equality, selected host limits, non-stationary tails, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, stationary seed exposure, constant-stream readback, finite seal consumption, dyadic ledger exactness, real finite-window exposure, and four-reading exhaustion for the same displayed packet.

Chapter 606

A Finite Value-Preservation Certificate for $\mathsf{ZeckendorfCarryValue}^{\uparrow}$

The $\mathsf{ZeckendorfCarryValue}^{\uparrow}$ surface packages the Fibonacci value reading of the displayed Zeckendorf carry as a finite $\mathsf{NameCert}$ packet. It keeps the carry classifier separate from history sameness: the packet records the source and target windows, the carry row, the two Fibonacci value rows, the equality-of-value check, and the visible provenance needed by consumers.

The dependency anchors are $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, and the displayed words of Definition 278.2.

Definition 606.1 (Zeckendorf carry-value carrier)**.** *A* $\mathsf{ZeckendorfCarryValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (h, k, G, N_h, N_k, F, H, C, P, N)$$

with the following visible rows:

1. $h =$ `word_011` *is the displayed source carry window;*
2. $k =$ `word_100` *is the displayed target carry window;*
3. G *is the local carry row* $\mathsf{ZCarry}(h, k)$ *inherited from* $\mathsf{ZCarry}^{\uparrow}$*;*
4. N_h *records the source normality boundary, namely that* h *is rejected by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
5. N_k *records the target normality boundary, namely that* k *is accepted by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
6. F *records the finite Fibonacci value calculation for the two displayed windows and the equality of those two values;*
7. H *records the negative history-sameness boundary* $\neg\mathsf{hsame}(h, k)$ *together with the displayed same-side transports allowed for consumers;*
8. C *records the* Cont *routes from the carry row to the value calculation and from the value calculation to the public read;*
9. P *is the* Pkg *provenance over* BHist, $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfNormal}^{\uparrow}$, hsame, *and the finite Fibonacci value rows;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryValue}^{\uparrow}}$ *row.*

The classifier compares accepted packets through the carry row, the two value rows, and the listed provenance. It does not identify h and k by hsame*, install a kernel rewrite, or export a host-level quotient of histories.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueTasteGate single carrier alignment`

Theorem 606.2 (Zeckendorf carry-value non-hsame boundary)**.** *For every accepted* $\mathsf{ZeckendorfCarryValue}^{\uparrow}$ *packet* $V = (h, k, G, N_h, N_k, F, H, C, P, N)$*, the value-preservation row and the history-sameness boundary occupy distinct coordinates. The row F states that the displayed Fibonacci value of h equals the displayed Fibonacci value of k, while H records* $\neg\mathsf{hsame}(h, k)$*. Thus value preservation is a finite classifier-facing calculation, not a kernel identification of the two histories.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueNonHsameBoundary`

Proof. Project the carry row G and the normality rows N_h, N_k. They fix the displayed source as `word_011` and the displayed target as `word_100` with the same orientation as the $\mathsf{ZCarry}^{\uparrow}$ packet.

Next project the value row F. It reads the finite Fibonacci value calculation for exactly those two displayed words and records equality of the computed values.

Finally project the boundary row H. Since H records the negative hsame fact for the same source and target, the equality in F cannot be read as a history-sameness witness. The public read therefore separates value preservation from kernel history identification. □

Theorem 606.3 (Zeckendorf carry-value NameCert package)**.** *The rows of Definition 606.1 form a finite* $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryValue}^{\uparrow}}$ *package. Its obligations are carrier habitation, carry-generator orientation, source and target normality boundaries, finite Fibonacci value calculation, non-*hsame *separation, continuation routing, package provenance, and the local naming row. Every consumer-visible value-preservation read factors through*

$$h, k, G, N_h, N_k, F, H, C, P, N.$$

No quotient stream, host arithmetic oracle, global normalization procedure, kernel rewrite, or source-target hsame *proof is exported.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueCarrier namecert package`

Proof. Carrier habitation is the displayed packet of Definition 606.1. The packet stores the two windows, the carry row, the normality boundary rows, the value calculation, and the provenance rows in a single finite history surface.

The carry-generator orientation is read from G, while N_h and N_k supply the source rejection and target admission rows. These coordinates determine which finite words are sent to the value calculation.

The value-preservation read is then routed through F and C. The row F stores the two displayed Fibonacci value computations and their equality check, and C records that consumers reach the public value row only along the carrier route.

The non-escape clauses are the remaining projections. The row H prevents the value equality from becoming $\mathsf{hsame}(h, k)$, while P and N account for provenance and naming. Since none of the excluded objects is a coordinate of the packet, no consumer can extract it from the package. □

Theorem 606.4 (Zeckendorf carry-value exactness obligation)**.** *For every accepted* $\mathsf{ZeckendorfCarryValue}^{\uparrow}$ *packet*

$$V = (h, k, G, N_h, N_k, F, H, C, P, N),$$

the value-row exactness obligation is exactly the finite Fibonacci value row F over the displayed carry packet `word_011` $\to$ `word_100`*. The row is read only after G, N_h, N_k identify the carry source, target, and normality boundaries, and it is exported only through H, C, P, N. It records equality of the two displayed Fibonacci values without identifying the histories by* hsame*.*

Proof. Project the accepted carrier of Definition 606.1. The rows h, k, G, N_h, N_k fix the source as `word_011`, the target as `word_100`, the displayed carry row, and the two normality boundary rows.

Now read the value coordinate. By Definition 606.3, the row F is the finite Fibonacci calculation attached to exactly those two displayed words, and the public value read factors through $h, k, G, N_h, N_k, F, H, C, P, N$.

Finally apply the non-hsame boundary from Definition 606.2. The value equality in F and the negative history-sameness row in H remain distinct coordinates, so exactness of the value row does not produce $\mathsf{hsame}(h, k)$. □

Theorem 606.5 (Zeckendorf carry-value vision realization)**.** *The finite packet $V = (h, k, G, N_h, N_k, F, H, C, P, N)$ realizes the Fibonacci value-preservation reading of the displayed carry row as a concrete* ZeckendorfCarryValue$^{\uparrow}$ *NameCert surface. It extends the carry classifier by adding the value row F and its consumer route, while preserving the boundary that carry-relatedness and history sameness are different judgments.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueCarrier value public read`
Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueVisionRealization`

Proof. Start from the displayed ZCarry$^{\uparrow}$ carrier. Its source, target, carry row, source rejection, target admission, and non-hsame boundary provide the rows h, k, G, N_h, N_k, H.

Add the finite value row F for the two displayed words and route it through C. This supplies a consumer-facing value-preservation check without changing the carry generator or the normal-form classifier.

Package the resulting rows with P and N. The package exposes value preservation only through the finite NameCert surface, so the existing separation between carry-relatedness and history sameness remains part of the public contract. □

Theorem 606.6 (Zeckendorf carry-value classifier dependency)**.** *Every accepted* ZeckendorfCarryValue$^{\uparrow}$ *packet consumes the carry row, source and target windows, normality boundary rows, and non-*hsame *separation supplied by the carry classifier of chapter 567 before it adds the Fibonacci value row. The value-preservation surface is therefore a finite row over the* ZCarry$^{\uparrow}$ *classifier, not a replacement for that classifier.*

Proof. Project the carrier $V = (h, k, G, N_h, N_k, F, H, C, P, N)$ from Definition 606.1. The rows h, k, G, N_h, N_k, H are the carry-side data: the two displayed windows, the carry row, the source and target normality boundaries, and the public separation from hsame.

Use the local non-hsame boundary and NameCert package, Definition 606.2 and Definition 606.3. They keep the value surface tied to the carry classifier while preventing the finite value equality from becoming a source-target history-sameness proof.

The additional row is F, routed through C and recorded with P and N. Thus the accepted packet first cites the sibling carry classifier of chapter 567 and then appends the finite Fibonacci value-preservation row over that cited carrier. □

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueUp classifier dependency rows`

Theorem 606.7 (Zeckendorf carry-value scoped public route)**.** *Every public carry-value read exported by* ZeckendorfCarryValue$^{\uparrow}$ *passes through* ZCarry$^{\uparrow}$, ZeckendorfNormal$^{\uparrow}$, *the displayed source and target word rows, the Fibonacci value row, the non-*hsame *boundary,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The exported surface is value preservation for the displayed carry packet, not kernel history identification, quotient arithmetic, a global normalizer, or a host-level numerical oracle.*

Proof. Project the carrier of Definition 606.1. The rows h, k, G, N_h, N_k fix the displayed words, the ZCarry$^{\uparrow}$ carry row, and the ZeckendorfNormal$^{\uparrow}$ source and target normality boundaries before the value row is read.

Next apply value-row exactness and the classifier dependency route, Definition 606.4 and Definition 606.6. They keep the Fibonacci equality in F over the carry classifier rather than as a replacement for the classifier.

Finally apply the non-hsame boundary and the NameCert package, Definition 606.2 and Definition 606.3. The rows H, C, P, N export value preservation only through the finite public route, so no kernel history identification, quotient arithmetic, global normalizer, or host-level numerical oracle is introduced. □

Closure status for NameCert$_{\text{ZeckendorfCarryValue}^{\uparrow}}$.

Constructive story: A ZeckendorfCarryValue$^{\uparrow}$ packet records the displayed carry from `word_011` to `word_100`, the finite Fibonacci value equality for those two words, the normality boundary rows, and the non-hsame separation row in one finite BHist certificate.

Theory closure: TheoryGate(NameCert$_{\text{ZeckendorfCarryValue}^{\uparrow}}$, publicClosed)

Scope closed: Definition 606.1, Definition 606.2, Definition 606.3, Definition 606.4, Definition 606.5, Definition 606.6, and Definition 606.7 bind the finite value-preservation public route over ZCarry$^{\uparrow}$, ZeckendorfNormal$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{ZeckendorfCarryValue}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ZeckendorfCarryValueUp.ZeckendorfCarryValueTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, a bridge certificate, a global Zeckendorf normalizer, uniqueness of numerical interpretation, a kernel rewrite, or source-target hsame.

Upgrade path: Bridged closure requires a bridge theorem connecting the public carry-value packet to the standard Zeckendorf value-preservation interface.

Chapter 607

A Concrete Naming Certificate for $\mathsf{ObservationConservationLedger}^{\uparrow}$

The $\mathsf{ObservationConservationLedger}^{\uparrow}$ interface names the finite monotone ledger packet that realises closed observation inside the BEDC kernel vocabulary. It binds observation states to displayed BHist records, uses hsame to transport old records into later states, routes every allowed read through Cont, records provenance through Pkg, and marks every non-recorded universal demand as a Gap row. The packet is a concrete NameCert carrier for the conservation side of closed observation; it is not an induction principle, not a universal quantifier over all future observations, and not an external oracle for unseen records.

The ledger consumes the observer-history comparison surface of chapter 477: old records are preserved only by transporting the displayed earlier record rows into the later observation state. That preservation row is the dependency edge read by the inscription-event packet of chapter 519, where an accepted inscription can cite the finite record that already survived the ledger transition rather than a host memory store or an unrecorded event.

Definition 607.1 (ObservationConservationLedger carrier)**.** *An* $\mathsf{ObservationConservationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O_t, O_{t+1}, R_t, R_{t+1}, A, G, H, C, P, N)$$

with the following visible coordinates:

1. O_t *and* O_{t+1} *are the earlier and later observation-state rows;*
2. R_t *is the finite ledger of records already visible at* O_t*;*
3. R_{t+1} *is the finite ledger visible at* O_{t+1}*;*
4. A *is the admission row saying which finite records are appended between the two states;*
5. G *is the* Gap *row assigned to any universal or future-facing assertion not witnessed by* R_t*,* R_{t+1}*, or* A*;*
6. H *records componentwise* hsame *transport for the earlier observation state, later observation state, old records, appended records, and* Gap *rows;*
7. C *records the visible* Cont *routes by which ledger consumers read old records, appended records, transport rows, and non-escape rows;*

8. *P is the* Pkg *provenance row for the observation-state, record-ledger, admission,* Gap, hsame, *and* Cont *coordinates;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationConservationLedger}^{\uparrow}}$ *row.*

The carrier exports only the two observation-state rows, the two finite record ledgers, the displayed admission row, the Gap *row, transport, continuation, package, and local naming rows. It contains no route for deleting old records, rewriting old records, manufacturing a universal record, or reading a witness absent from the displayed finite ledgers.*

Lean4 (definition) `BEDC.Derived.ObservationConservationLedgerUp.ObservationConservationLedgerCarrier`

Theorem 607.2 (ObservationConservationLedger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservationConservationL}}$ *surface consists exactly of five finite obligations:*

1. *carrier habitation by the finite packet of Definition 607.1;*

2. *old-record preservation from R_t into R_{t+1} through the displayed admission row A;*

3. *appended-record visibility for every record admitted by A;*

4. *componentwise* hsame *and* Cont *stability for the observation-state rows, record-ledger rows, admission rows, and* Gap *rows;*

5. *provenance and naming closure through P and N, forcing every consumer read to remain inside the displayed finite packet.*

No obligation exports a universal quantifier over future observations, an induction principle, a deletion route for old records, or an oracle for unseen witnesses.

Lean4 ✓ `BEDC.Derived.ObservationConservationLedgerUp.ObservationConservationLedgerCarrier namecert obligations`

Proof. Carrier habitation is exactly Definition 607.1. The two observation states, both record ledgers, the admission row, the Gap row, transport, continuation, package, and naming rows are all displayed coordinates of one finite packet.

Old-record preservation is read by comparing R_t with R_{t+1} through A. The admission row can append finite records, and the transport row H moves the earlier ledger componentwise into the later packet without changing its visible record identities.

Appended-record visibility is supplied by the same admission row. Any newly admitted record must appear in R_{t+1} and must be routed through C, P, and N before a consumer can read it.

The remaining demands are non-escape demands. If a consumer asks for a universal or future-facing assertion not witnessed by the displayed rows, the request is assigned to G rather than accepted as a record, so the local NameCert surface contains exactly the listed finite obligations. □

Theorem 607.3 (ObservationConservationLedger non-escape)**.** *Every consumer of an accepted* $\mathsf{ObservationConservationLedger}^{\uparrow}$ *carrier reads only the displayed rows $O_t, O_{t+1}, R_t, R_{t+1}, A, G, H, C, P, N$. A consumer may recover preserved old records, newly appended finite records, componentwise transports, continuation routes, provenance, and the local naming row. It cannot recover an unrecorded witness, a deletion or alteration of an old record, a universal assertion over all future observations, or an induction principle.*

Lean4 ✓ `BEDC.Derived.ObservationConservationLedgerUp.ObservationConservationLedgerCarrier gap read nonescape`

Proof. Inspect the carrier of Definition 607.1. Its only record coordinates are the finite ledgers R_t and R_{t+1} together with the finite admission row A.

Apply the NameCert obligation surface of Definition 607.2. Old records persist through H, appended records become visible through A, and every legal consumer read is routed through C, P, and N.

It remains to rule out non-displayed reads. A request for a universal or future-facing assertion has no witness coordinate in R_t, R_{t+1}, or A, so it is routed to the Gap row G. Since G is a tag rather than an accepted record, the carrier has no route from closed observation to an unrecorded witness, record deletion, universal closure, or induction. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationConservationLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservationConservationLedger}^{\uparrow}$ packet is generated by two observation-state rows, two finite record ledgers, a finite admission row, a Gap non-escape row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservationConservationLedger}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 607.1, Definition 607.2, and Definition 607.3 bind the BHist carrier row, hsame transport, Cont replay, Pkg provenance, Gap non-escape row, and local NameCert obligations for the finite monotone observation ledger packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservationConservationLedger}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ObservationConservationLedgerUp.ObservationConservationLedgerCarrier\_namecert\_obligat`

Bridge status: none

Origin: human

Not claimed: This chapter does not close induction, universal quantification over future observations, infinite observation streams, or any oracle for records absent from the displayed finite ledgers.

Upgrade path: Bridged closure requires an explicit standard bridge for the monotone finite observation ledger.

Chapter 608

A Concrete Naming Certificate for $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$

The $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ interface packages the displayed Zeckendorf carry witness as a finite naming certificate. It records the source non-normal row, the target normal row, the Fibonacci-value ledger, the non-hsame boundary, displayed transports, continuation routes, package provenance, and a local naming row. The packet therefore separates normalization-by-carry from history sameness while keeping value preservation as an explicit ledger row.

The source non-normal row and target normal row are read through chapter 278, while the carry witness row is read through chapter 544. The value ledger and non-hsame boundary are therefore sibling-carried rows, not standalone local data.

Definition 608.1 (Zeckendorf carry-normalization carrier)**.** *A* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (h, k, W, V, H, C, P, N)$$

with the following visible coordinates:

1. *h is the displayed source history row with public carry window* `word_011`*;*
2. *k is the displayed target history row with public carry window* `word_100`*;*
3. *W is the Zeckendorf carry witness row, reading* $\mathsf{ZCarry}(h, k)$ *through the local generator;*
4. *V is the Fibonacci-value ledger row, recording that the value read from h and the value read from k are the same displayed Fibonacci sum;*
5. *H records the non-*hsame *boundary between h and k together with the componentwise* hsame *transports that preserve the source row, target row, carry witness, value ledger, package row, and local naming row;*
6. *C stores the finite* Cont *routes by which consumers read the source non-normal row, target normal row, carry row, value ledger, boundary row, package provenance, and local naming row;*
7. *P is the* Pkg *provenance over* BHist, $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, *Fibonacci-value ledgers,* hsame, Cont, *and local* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryNormalization}^{\uparrow}}$ *row.*

The carrier exports only the displayed carry packet and its finite value ledger. It contains no source-target hsame *proof, no global normalization algorithm, no arithmetic model outside the Fibonacci rows, and no kernel rewrite rule.*

Lean4 (definition) `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier`

Theorem 608.2 (Zeckendorf carry-normalization NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryNormalizat}}$ *surface for Definition 608.1 consists of five finite obligations:*

1. *carrier admission for the finite packet* $Z = (h, k, W, V, H, C, P, N)$*;*
2. *source and target normality, stating that* h *is rejected by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier and* k *is accepted by it;*
3. *value-ledger exactness, stating that* V *is the unique public Fibonacci-value row exported by the packet;*
4. *carry-is-not-*hsame *preservation, stating that the carry witness* W *does not supply* $\mathsf{hsame}(h, k)$ *and that all displayed transports preserve that boundary;*
5. *local naming, stating that* P *and* N *name exactly the displayed carrier rows and no additional normalization object.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier namecert obligations`

Proof. Carrier admission is projection of the tuple in Definition 608.1. The source row, target row, carry witness, value ledger, boundary-and-transport row, continuation routes, package provenance, and local naming row are all displayed coordinates of one finite BHist packet.

The normality clause reads the source and target sides through C and the normal-form rows named by P. The source window is the non-normal `word_011` row, while the target window is the normal `word_100` row.

The value and boundary clauses are separate projections. The row V records the Fibonacci-value equality of the two displayed sums, while H records that the carry witness is not an $\mathsf{hsame}(h, k)$ proof and transports only the named components.

Finally, the package and local naming rows repack the same finite surface. Since no coordinate names a global normalizer, source-target hsame proof, or kernel rewrite rule, the five obligations exhaust the local NameCert surface. □

Theorem 608.3 (Zeckendorf carry-normalization value ledger)**.** *For an accepted* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *carrier*

$$Z = (h, k, W, V, H, C, P, N),$$

every public claim that the source and target carry rows have the same Fibonacci value factors through the displayed ledger V*. The ledger boundary is exact:* V *may be used to read equality of the two Fibonacci sums, but it cannot be repacked as* $\mathsf{hsame}(h, k)$*, as a kernel rewrite, or as a global normalization procedure.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier value ledger`

Proof. Read the accepted packet through the carrier definition. The only coordinate that mentions Fibonacci value is V, and the continuation routes in C expose that coordinate as a finite ledger row.

The carry witness W supplies the local $\mathsf{ZCarry}(h, k)$ edge, while the row H keeps the non-hsame boundary visible under componentwise transport. Thus a consumer can combine W and V to

read value-preserving carry normalization, but the consumer still lacks a source-target $\mathsf{hsame}(h, k)$ witness.

The package provenance P and local naming row N name exactly this packet. They do not add a global normalizer or a kernel rewrite coordinate, so every public value-preservation read remains inside the displayed ledger boundary. □

Theorem 608.4 (Zeckendorf carry-normalization carry and Nat dependency)**.** *For an accepted* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *packet*

$$Z = (h, k, W, V, H, C, P, N),$$

the value ledger V is readable only after the local carry row certified by chapter 558 and the unary Fibonacci-index discipline of chapter 28 have been consumed. Thus V is a sibling-dependent ledger row, not an isolated kernel-only row.

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier axis nat dependency`

Proof. Project $Z = (h, k, W, V, H, C, P, N)$ from Definition 608.1. The row W is the local carry witness, while V is the row that records equality of the displayed Fibonacci sums.

Read W through the Zeckendorf carry sibling of chapter 558. The source and target windows h, k are indexed as unary Nat positions, so the Fibonacci indexing discipline is read through chapter 28 before the sums named by V are meaningful.

Finally apply Definition 608.3. Since every public value-preservation read factors through V after the carry edge and Nat-index rows have been fixed, V records a sibling-dependent ledger rather than a detached kernel-only certificate. □

Theorem 608.5 (Zeckendorf carry-normalization source-target totality)**.** *For an accepted* ZeckendorfCarryNormaliz *packet*

$$Z = (h, k, W, V, H, C, P, N),$$

the visible source and target normality surface is total for the displayed carry move: the source row h is exposed as the public non-normal `word_011` *row, the target row k is exposed as the public normal* `word_100` *row, and every consumer route that reads the local carry edge W also reads those two source-target normality rows through the same finite packet.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier source target window totality`

Proof. Begin with the carrier definition Definition 608.1. The rows h, k, and W are coordinates of the same finite packet Z, with h displaying the non-normal carry window and k displaying the normal target window.

Apply the NameCert obligation theorem Definition 608.2. Its source and target normality clause makes the two classifier reads part of the admitted certificate surface, while its carrier-admission clause keeps the carry edge W tied to those same rows.

Finally use the value ledger theorem Definition 608.3. Since public value-preservation reads remain inside V and the displayed packet, no consumer can read the carry edge as a detached operation without also seeing the source and target normality rows. □

Theorem 608.6 (Zeckendorf carry-normalization boundary transport)**.** *For an accepted* ZeckendorfCarryNormalizat *packet*

$$Z = (h, k, W, V, H, C, P, N),$$

*the carry-is-not-*hsame *boundary is preserved by the displayed transport rows H and continuation routes C. Transporting the source row, target row, carry witness, Fibonacci-value ledger, provenance row, or local name along those routes never produces an* hsame(h, k) *proof and never turns the carry witness into a kernel rewrite rule.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier boundary transport`

Proof. Project the packet through Definition 608.1. The row H contains the non-hsame boundary together with the componentwise transports, while C records only the finite continuation routes by which consumers read the displayed rows.

The boundary clause of Definition 608.2 states that W does not supply hsame(h, k) and that all displayed transports preserve that boundary. The value ledger theorem Definition 608.3 keeps equality of Fibonacci values confined to V.

Therefore transport through H and C may preserve source, target, carry, value, package, and naming rows, but it cannot create source-target history sameness or a kernel rewrite coordinate. □

Theorem 608.7 (Zeckendorf carry-normalization local naming exhaustion)**.** *For an accepted* ZeckendorfCarryNormalization$^{\uparrow}$ *packet*

$$Z = (h, k, W, V, H, C, P, N),$$

the provenance row P and local NameCert *row N are exhausted by the displayed packet rows h, k, W, V, H, C, P, N. They name the source non-normal row, target normal row, carry witness, Fibonacci-value ledger, non-*hsame *boundary, transport and continuation routes, and no global normalizer, source-target* hsame *proof, ambient arithmetic model, or kernel rewrite rule.*

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier local name window exhaustion`

Proof. Read P and N through the carrier definition Definition 608.1. Both rows are part of the same finite packet as the source, target, carry, value, boundary, transport, and continuation data.

The local naming clause of Definition 608.2 says that P and N name exactly the displayed carrier rows and no additional normalization object. The value ledger theorem Definition 608.3 further confines all public value-preservation reads to V.

Thus the naming surface is exhausted by Z itself. Any consumer of P or N receives the displayed carry-normalization packet and none of the excluded global or kernel-level objects. □

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier local naming exhaustion`

Theorem 608.8 (Zeckendorf carry-normalization bridged export)**.** *The public* ZeckendorfCarryNormalization$^{\uparrow}$ *route exports only the finite Nat/carry rows already displayed by Definition 608.1. A bridge-facing consumer may read the source non-normal row h, target normal row k, carry witness W, value ledger V, boundary row H, continuation row C, provenance row P, and local name N, but it receives no source-target host equality, global normalization theorem, or kernel rewrite rule.*

Proof. Start from the public carrier and local naming exhaustion theorems, Definition 608.1 and Definition 608.7. They expose exactly the rows h, k, W, V, H, C, P, N and state that the local name is exhausted by those rows.

Next use the public route theorems. The NameCert obligations of Definition 608.2 bind carrier admission, source-target normality, the value ledger, the non-hsame boundary, and local naming. The Nat/carry dependency theorem Definition 608.4 keeps the value ledger downstream of the displayed carry row and Nat-index discipline.

Finally apply the value ledger and boundary transport theorems, Definition 608.3 and Definition 608.6. They keep public value preservation inside V and keep transport from creating $\mathsf{hsame}(h, k)$ or a rewrite coordinate. Thus the bridge-facing surface is exactly the finite Nat/carry route already displayed by the chapter. □

Lean4 ✓ `BEDC.Derived.ZeckendorfCarryNormalizationUp.ZeckendorfCarryNormalizationCarrier bridged export`

Theorem 608.9 (Zeckendorf carry-normalization mature package)**.** *The* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *carrier, NameCert obligations, value ledger, Nat/carry dependency route, source-target totality, boundary transport, local naming exhaustion, and bridged export assemble one mature finite Nat/carry package. The package exposes the displayed source row h, target row k, carry witness W, value ledger V, boundary row H, continuation row C, provenance row P, and local name N as the complete public surface. It still exports no source-target* hsame*, global normalization theorem, ambient arithmetic model, or kernel rewrite rule.*

Proof. Start with the carrier and NameCert obligation rows of Definition 608.1 and Definition 608.2. They admit exactly the finite packet $Z = (h, k, W, V, H, C, P, N)$ and bind source-target normality, value exactness, boundary preservation, and local naming to that packet.

Next use the value ledger, Nat/carry dependency, and source-target totality rows. Theorems 608.3, 608.4, and 608.5 keep value preservation downstream of the displayed carry row, Nat indexing discipline, and visible source-target windows.

Finally apply boundary transport, local naming exhaustion, and bridged export. Theorems 608.6, 608.7, and 608.8 preserve the non-hsame boundary and expose only the finite Nat/carry route. Hence the assembled package is mature on the paper theory axis without adding the excluded global or kernel-level objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeckendorfCarryNormalization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ packet is generated from a source non-normal row, target normal row, carry witness row, Fibonacci-value ledger row, non-hsame boundary row, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeckendorfCarryNormalization}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 608.1, Definition 608.2, Definition 608.3, Definition 608.4, and Definition 608.5 bind the displayed carrier, NameCert obligations, value ledger, and Nat/carry sibling dependency route as the scoped finite carry-normalization surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeckendorfCarryNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close source-target hsame, global normalization, kernel rewrite rules, or a checked Lean certificate.

Upgrade path: No further theory-axis upgrade is claimed; only Lean bridge verification remains outside the paper package.

Chapter 609

A Concrete Naming Certificate for $\mathsf{ObserverCoordinateFrame}^{\uparrow}$

The $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ surface names the local coordinate-frame packet that a conscious observer may use without appealing to a global observer frame. It is a finite consumer of the conscious-observer-state row of chapter 578 and the observer-locality-cell row of chapter 513, then adds $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and $\mathsf{MaxCausalRate}^{\uparrow}$ rows without forming an ambient chart over all observers.

Definition 609.1 (Observer-coordinate-frame carrier)**.** *An* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, L, M, C, B, H, R, P, N)$$

with the following displayed rows:

1. *S is an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *row naming the observer-state surface at which the frame is read;*
2. *L is an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row naming the locality cell through which nearby observer records are coordinated;*
3. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite collection of observer histories available to the frame;*
4. *C is a finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows over selected pairs in M and the locality-cell anchors in L;*
5. *B is an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *row bounding the displayed cross-Hist causal reads without adding a metric tensor or global clock;*
6. *H is the componentwise* hsame *transport ledger for S, L, M, C, B and for the package and naming rows;*
7. *R is the* Cont *route ledger by which a consumer reads local coordinates through S, L, M, C, B, H;*
8. *P is the* Pkg *provenance over* $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverCoordinateFrame}^{\uparrow}}$ *row.*

The classifier compares two carriers only by the displayed rows and their hsame *and* Cont *ledgers. It has no coordinate for a god's-eye observer, a single global chart, a shared external time scalar, a host spacetime manifold, or an observer-independent event object.*

Theorem 609.2 (ObserverCoordinateFrame NameCert obligations)**.** *For the carrier of Definition 609.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverCoordinateFrame}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier exposure for the rows* $S, L, M, C, B, H, R, P, N$*; observer-state anchoring through* S*; locality-cell and multi-Hist anchoring through* L *and* M*; causal and bound routing through* C *and* B*; and non-escape from global observer frames, shared clocks, host manifolds, metric tensors, observer-independent events, quotients of histories, and unrecorded histories.*

Lean4 ✓ `BEDC.Derived.ObserverCoordinateFrameUp.ObserverCoordinateFrameUp taste gate boundary`

Proof. Carrier exposure is projection from the finite tuple $F = (S, L, M, C, B, H, R, P, N)$.

Observer-state anchoring is read from S. Locality-cell and multi-Hist anchoring are read from L and M, so every frame coordinate begins inside the displayed observer state and its listed local comparison surface.

The causal and bound obligations are then read from C and B. The family C supplies only selected $\mathsf{CrossHistCausal}^{\uparrow}$ rows over the visible configuration, while B supplies the finite $\mathsf{MaxCausalRate}^{\uparrow}$ bound row for those reads.

Finally inspect the remaining coordinates. The transport ledger H, route ledger R, package row P, and local name row N can replay only the displayed finite rows, so no global frame, shared clock, host manifold, metric tensor, observer-independent event, quotient, or hidden history is exported. □

Theorem 609.3 (ObserverCoordinateFrame locality soundness)**.** *Every accepted* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *packet is local to the conscious observer state* S*, the locality cell* L*, and the finite multi-Hist configuration* M*. A consumer may read selected cross-Hist causal rows and their maximum-rate bound through* C *and* B*, transported by* H *and routed by* R*, but the consumer cannot read a coordinate outside the displayed locality-cell and multi-Hist rows.*

Proof. Project the accepted packet as $F = (S, L, M, C, B, H, R, P, N)$. The observer-state row S fixes the state at which the frame is read, while L and M fix the locality-cell and multi-Hist surfaces available at that state.

The selected causal reads in C are anchored in those rows. The bound row B is a finite $\mathsf{MaxCausalRate}^{\uparrow}$ consumer of the same cross-Hist surface, not a metric or global-chart object.

Transport and continuation replay through H and R preserve the same finite rows. Since no coordinate outside $S, L, M, C, B, H, R, P, N$ is exposed, every accepted frame remains locality-cell sound. □

Theorem 609.4 (ObserverCoordinateFrame non-global-frame boundary)**.** *An accepted* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *carrier exports only conscious-observer-state, observer-locality-cell, multi-Hist configuration, cross-Hist causal, maximum-bound,* hsame *transport,* Cont *route, package, and local* NameCert *rows. It exports no god's-eye observer, no global coordinate frame, no shared external time scalar, no host spacetime manifold, no metric tensor, no observer-independent event object, no quotient of histories, and no unrecorded remote history.*

Proof. Use the carrier inventory of Definition 609.1. Every public read of the frame is one of the rows $S, L, M, C, B, H, R, P, N$.

The positive frame data is therefore local: S supplies the present conscious-observer state, L supplies the locality cell, M supplies the finite history configuration, C supplies selected cross-Hist causal rows, and B supplies the finite bound row.

The remaining rows H, R, P, N transport, route, package, and name those same coordinates. None of them contains a god's-eye observer, global chart, shared time scalar, manifold, metric,

observer-independent event, quotient, or hidden remote history, so none of those objects can be projected from the packet. □

Falsifiable prediction: An accepted ObserverCoordinateFrameUp packet cannot export a global observer frame, shared external time scalar, host spacetime manifold, metric tensor, observer-independent event object, quotient of histories, or unrecorded remote history while preserving the displayed local-frame carrier rows.

Independence witness: consciousobserverstate, observerlocalitycell, multihistconfig, crosshistcausal, maxcausalrate: no sibling is bijective to $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ because this carrier simultaneously stores observer-state, locality-cell, multi-Hist configuration, selected cross-Hist causal rows, a maximum causal-rate bound, transport, route, provenance, and local naming rows; each sibling supplies only one face of that local-frame packet.

The Lean TasteGate namespace is the natural home for the `Nontrivial` and `FieldFaithful` instances for this displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverCoordinateFrame}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ packet is generated from a conscious-observer-state row, an observer-locality-cell row, a multi-Hist configuration row, selected cross-Hist causal rows, a maximum-causal-bound row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverCoordinateFrame}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 609.1, Definition 609.2, Definition 609.3, Definition 609.4, and the checked TasteGate boundary bind the finite local-frame scope over $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverCoordinateFrame}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ObserverCoordinateFrameUp.ObserverCoordinateFrameUp\_taste\_gate\_boundary`
Lean4 ✓ `BEDC.Derived.ObserverCoordinateFrameUp.taste gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close bridge status, god's-eye observer, global coordinate frame, shared external time scalar, host spacetime manifold, metric tensor, observer-independent event object, quotient of histories, or unrecorded remote-history read.

Upgrade path: The paper theory axis is mature over the displayed local-frame rows; machine bridge checking remains the formal-axis route beyond the existing audit-clean TasteGate boundary.

Chapter 610

A Concrete Naming Certificate for $\mathsf{InscriptionPoint}^{\uparrow}$

The $\mathsf{InscriptionPoint}^{\uparrow}$ surface realizes the local carrier promised by Definition 9393.15. It packages an individual inscription site as finite BEDC data: one accepted history, one visible gap socket, one supply-socket ledger row, one inscription-event row, transport and continuation routes, package provenance, and a local naming row. The packet does not treat the supply as a substrate-internal rule; it records only the finite site at which the supply is ledgered for downstream consumers.

The inscription point consumes the supply ledger of chapter 528, the induction-socket handoff of chapter 498, and the event row of chapter 519 before packaging the local site. These dependencies are the concrete rows named by S, T, I, not external supply, induction, or event principles.

Definition 610.1 (InscriptionPoint carrier)**.** *An* $\mathsf{InscriptionPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (H, G, S, T, I, L, R, C, P, N)$$

with the following displayed rows:

1. *H is the accepted* BHist *row naming the local observational history;*
2. *G is the* Gap *row governed by* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$*, recording the finite trace boundary at which no substrate-internal supply rule is provided;*
3. *S is the* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row classifying the requested supply as an inscription-novelty socket rather than as a hidden derivation rule;*
4. *T is the* $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ *handoff row, naming the forward-binding induction socket that the local history may consume without internalizing its far end;*
5. *I is the* $\mathsf{InscriptionEvent}^{\uparrow}$ *row that records the public event by which the site becomes readable as an accepted local inscription;*
6. *L is the local ledger row tying H, G, S, T, I to the same inscription point;*
7. *R records componentwise* hsame *comparisons for H, G, S, T, I, L, package provenance, and the local naming row;*
8. *C records the displayed* Cont *routes by which consumers may read the history, gap socket, supply ledger, induction handoff, inscription event, and ledger row;*
9. *P is the* Pkg *row naming* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, $\mathsf{TranscendentalInductionSoc}$ $\mathsf{SupplySocketLedger}^{\uparrow}$*, and* $\mathsf{InscriptionEvent}^{\uparrow}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only these displayed rows and their transported continuations. It contains no positive predicate for the far end of the supply, no universal induction rule, no private subject predicate, and no hidden source selector.

Lean4 (definition) `BEDC.Derived.InscriptionPointUp.InscriptionPointUp`
Lean4 ✓ `BEDC.Derived.InscriptionPointUp.InscriptionPointTasteGate single carrier alignment`

Theorem 610.2 (InscriptionPoint NameCert obligations)**.** *For the carrier of Definition 610.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission: every accepted inscription point exposes exactly $J = (H, G, S, T, I, L, R, C, P, N)$;*
2. *history locality: every consumer read begins at the displayed* BHist *row H;*
3. *gap and supply exactness: G is the finite trace socket, S is the supply-socket ledger, and T is the transcendental-induction handoff row consumed at that socket;*
4. *event and transport stability: the inscription event I, local ledger L, componentwise* hsame *row R, and* Cont *routes C preserve the same finite packet for downstream reads;*
5. *non-escape: the packet exports no substrate-internal supply rule, private subject primitive, hidden source selector, or positive predicate about the far end of the supply.*

Proof. Carrier admission is projection from the displayed tuple in Definition 610.1. The accepted history, gap socket, supply ledger, handoff row, inscription event, local ledger, transport row, continuation routes, package provenance, and local naming row are all finite coordinates.

History locality reads the first coordinate H. Every downstream route in C begins from the displayed history row and then reaches the gap, ledger, handoff, event, and naming rows through the same packet.

Gap and supply exactness are recorded by G, S, T. The finite trace gap is classified before the supply ledger is consumed, and the transcendental-induction handoff is named as a handoff rather than as an internal derivation principle.

Event and transport stability use I, L, R, C. The inscription event makes the local site publicly readable, the ledger ties the rows to the same point, and the hsame and Cont rows replay the same finite packet after transport.

The non-escape clause is the inventory boundary. Since J has no coordinate for a closed supply rule, private subject primitive, hidden source selector, or positive far-end predicate, no local $\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$ projection can export one. □

Theorem 610.3 (InscriptionPoint gap handoff)**.** *For an accepted* $\mathsf{InscriptionPoint}^{\uparrow}$ *carrier $J = (H, G, S, T, I, L, R, C, P, N)$, every consumer read of the local inscription point factors through the finite trace gap row G, the supply-socket ledger row S, the transcendental-induction handoff row T, and the inscription-event row I before it reaches the local* NameCert *row N.*

Proof. Start with the carrier coordinates of Definition 610.1. The rows G, S, T, I, N are displayed in the same finite BHist packet, and C records the permitted routes among them.

Apply the gap and supply exactness clause of Definition 610.2. It fixes G as the finite trace socket, S as the supply-socket ledger, and T as the forward-binding handoff row attached to that socket.

Apply the event and transport stability clause. The inscription event I and ledger L keep the local point readable through the same hsame and Cont routes, so the NameCert row cannot be consumed through a route that bypasses the gap and handoff rows. □

Theorem 610.4 (InscriptionPoint non-escape)**.** *No consumer of an accepted* $\mathsf{InscriptionPoint}^{\uparrow}$ *packet can extract a substrate-internal rule for the supply, a private subject primitive, a hidden source selector, a global observer frame, or a positive predicate about the far end of the transcendental supply. Every consumer read factors through*

$$H, G, S, T, I, L, R, C, P, N.$$

Proof. Inspect the carrier inventory in Definition 610.1. The exported rows are exactly the local history, finite trace gap, supply ledger, handoff row, inscription event, ledger, transport row, continuation routes, package provenance, and naming row.

Definition 610.2 supplies the non-escape clause for these rows. The classifier may compare the displayed packet and its transports, but it has no coordinate that evaluates the supply as a rule inside the substrate.

Definition 610.3 fixes the route shape for downstream reads. Since every read passes through the finite gap, ledger, handoff, and event rows before the local name is consumed, there is no projection target for a private subject primitive, hidden selector, global observer frame, or positive far-end predicate. □

Theorem 610.5 (InscriptionPoint socket surface through the canonical carrier)**.** *Every supply-socket read of an accepted* $\mathsf{InscriptionPoint}^{\uparrow}$ *carrier factors through the canonical rows*

$$H, G, S, T, I, L, R, C, P, N.$$

The socket and Gap *portion is supplied by* G, S, T*, the public event portion by* I, L*, and transport and provenance by* R, C, P, N*. The canonical carrier already absorbs the* O, S, A, G, H, C, P, N *socket surface: it exports no internally supplied theorem, hidden receiver, private source predicate, selected witness, quotient equality, or completed supply principle.*

Proof. Definition 610.1 fixes the accepted carrier as the displayed tuple $J = (H, G, S, T, I, L, R, C, P, N)$. Its socket-facing rows are not external receivers; they are the finite trace gap, supply-socket ledger, and transcendental-induction handoff rows named by G, S, T.

Definition 610.2 gives the NameCert inventory. Gap and supply exactness identify G, S, T as the only socket and handoff portion, while event and transport stability identify I, L, R, C as the public event and replay surface for the same packet.

Definition 610.3 forces every consumer read through G, S, T, I before the local $\mathsf{NameCert}$ row is consumed. Definition 610.4 supplies the boundary: the carrier has no coordinate for an internally supplied theorem, hidden receiver, private source predicate, selected witness, quotient equality, or completed supply principle.

Thus any read formerly expressible as O, S, A, G, H, C, P, N is read inside the canonical rows. Observation, socket, actuality, gap, transport, continuation, provenance, and naming are accounted for by $H, G, S, T, I, L, R, C, P, N$, with no additional exported theorem or hidden supply surface. □

Theorem 610.6 (InscriptionPoint bridge-facing preservation)**.** *The standard bridge-facing read of an accepted* $\mathsf{InscriptionPoint}^{\uparrow}$ *carrier preserves exactly*

$$H, G, S, T, I, L, R, C, P, N$$

through the existing finite gap, supply-ledger, induction-handoff, and inscription-event rows. The bridge-facing surface has no additional coordinate beyond the carrier of Definition 610.1, the NameCert obligations of Definition 610.2, the gap handoff of Definition 610.3, the non-escape boundary of Definition 610.4, and the canonical-carrier socket surface of Definition 610.5.

Proof. The carrier definition fixes $H, G, S, T, I, L, R, C, P, N$ as the only displayed rows. The finite gap, supply ledger, induction handoff, and inscription event are not reconstructed by a bridge read; they are the existing coordinates G, S, T, I.

The NameCert obligations preserve the same tuple under classifier, exactness, transport, continuation, provenance, and local naming reads. Gap handoff then forces every consumer route through G, S, T, I before the local NameCert row is consumed.

The non-escape and canonical-carrier socket theorems remove the remaining alternatives. Since the packet has no hidden supply rule, far-end predicate, private subject primitive, selected witness, quotient equality, or completed supply coordinate, a bridge-facing read can only preserve the displayed rows already present in $H, G, S, T, I, L, R, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionPoint}^{\uparrow}$ packet is generated from one accepted BHist row, a finite trace gap socket, a supply-socket ledger row, a transcendental-induction handoff row, an inscription-event row, a local ledger, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 610.1, Definition 610.2, Definition 610.3, Definition 610.4, and Definition 610.5 bind the finite inscription-point carrier, classifier, gap handoff, non-escape surface, finite packet boundary, and public export surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InscriptionPoint}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InscriptionPointUp.InscriptionPointTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.InscriptionPointUp.taste gate`

Not claimed: This chapter does not close bridge checking, a substrate-internal supply rule, private subject primitive, hidden source selector, global observer frame, positive far-end predicate, or standard bridge for $\mathsf{InscriptionPoint}^{\uparrow}$.

Upgrade path: No further paper-axis closure grade remains; only bridge checking remains on the formal and bridge axes.

Chapter 611

A Concrete Naming Certificate for $\mathsf{RealObservationBudget}^{\uparrow}$

The $\mathsf{RealObservationBudget}^{\uparrow}$ interface records the finite observation packet that a completion-facing consumer may inspect before a $\mathsf{Real}^{\uparrow}$ seal is reached. It keeps the precision request, stream windows, dyadic readbacks and radii, regular-sequence handoff rows, real-seal rows, transports, routes, provenance, and local naming row inside one BHist surface. It does not export quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded observation schedule.

This budget packet is a consumer of chapter 126, chapter 291, chapter 301, and chapter 38. Its substantive dependency path is finite window selection, dyadic readback and radius rows, regular-sequence handoff, and the real-seal boundary.

Whenever a completion-facing $\mathsf{Real}^{\uparrow}$ consumer asks for a synchronized diagonal read, this finite observation route also consumes the diagonal selector surface of chapter 853. The $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ sibling supplies the finite precision, mesh, and comparison budget that $\mathsf{RealObservationBudget}^{\uparrow}$ repackages as observation windows, dyadic readbacks, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, and $\mathsf{Real}^{\uparrow}$ seal rows.

Definition 611.1 (Real observation budget carrier)**.** *A* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, W, D, R, S, H, C, P, N)$$

with the following displayed coordinates:

1. *E is the unary precision request row naming the finite tolerance level visible to the consumer;*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *window rows selected for that request;*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks together with the displayed dyadic radii attached to the windows in W;*
4. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows that expose the regular rational sources feeding the observation budget;*
5. *S stores the* $\mathsf{Real}^{\uparrow}$ *seal rows that mark the boundary at which completion-facing consumers receive the finite packet;*
6. *H stores componentwise* hsame *transports for the precision request, windows, dyadic readbacks and radii, regular-sequence rows, seal rows, provenance, and local name;*

7. *C stores the displayed* Cont *routes from E to W, from W to D, from D to R, and from R to S;*
8. *P records* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ *row.*

The carrier is finite. It has no coordinate for quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded observation schedule.

Theorem 611.2 (Real observation budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ *surface for Definition 611.1 consists exactly of carrier admission, unary precision exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and radius exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *seal-boundary exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet B. The rows E, W, D, R, S provide the precision request, finite windows, dyadic readbacks and radii, regular-sequence handoffs, and real-seal boundary.

The window and dyadic obligations are read directly from W and D. The regular-sequence handoff rows in R expose only the rational-source data already routed through those finite observations.

The remaining obligations are the transport, route, provenance, and naming rows H, C, P, N. Since the carrier has no coordinate for quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded schedule, the listed rows exhaust the local naming certificate. □

Theorem 611.3 (Real observation budget window coverage)**.** *For an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier, every dyadic readback and displayed radius exported by D is covered by one of the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W, and every such window is selected by the unary precision request row E through the displayed routes in C.*

Lean4 ✓ `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetWindowCoverage`

Proof. Project the accepted packet B and read the precision row E, window family W, dyadic ledger D, transports H, and continuation routes C.

The route from E to W selects the finite windows visible to the budget. The route from W to D is the only displayed path by which a dyadic readback or radius enters the packet.

Transport rows in H preserve the same precision, window, readback, and radius coordinates when the packet is repacked for a consumer. Therefore every exported dyadic observation is covered by W and no observation can be imported from an unbounded stream schedule. □

Theorem 611.4 (Real observation budget regular-sequence handoff)**.** *For an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier, each* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row in R is reached only after the unary precision request, finite stream windows, and dyadic readbacks and radii have been read. A downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the handoff through the finite route*

$$E, W, D, H, C, P, N.$$

Proof. Begin with the carrier projection. The displayed order of the observation route places R after the precision request, stream windows, and dyadic ledger.

Use window coverage to bind every dyadic readback and radius to the finite windows selected from E. The continuation route from D to R then supplies the only path from the observation budget to the regular rational source rows.

The transport and provenance rows repack the same route for consumers. No handoff row can bypass E, W, D, H, C, P, N or appeal to quotient stream equality, hidden limit choice, or host real comparison. □

Theorem 611.5 (Real observation budget seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier factors through the seal rows* S*. The seal rows may read only the precision request* E*, finite windows* W*, dyadic readbacks and radii* D*, regular-sequence handoffs* R*, transports* H*, routes* C*, provenance* P*, and local naming row* N *supplied by the packet.*

Proof. Project the accepted packet and inspect the consumer-facing coordinates. The carrier contains a single real-facing boundary coordinate, namely the seal family S.

Apply the regular-sequence handoff theorem to force any route into S through E, W, D, R, H, C, P, N. The seal rows therefore receive only the finite observations, their radii, the regular rational handoffs, and the bookkeeping rows displayed by the packet.

Since S is reached after those rows and no other real-facing coordinate exists, a completion consumer cannot read a selected completed real, a host equality test, an ambient completeness principle, or a hidden infinite schedule from the budget. □

Theorem 611.6 (Real observation budget non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier factors through the finite rows*

$$E, W, D, R, S, H, C, P, N.$$

The consumer may inspect exactly the unary precision request, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and radii,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It cannot inspect quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded observation schedule.*

Lean4 ✓ `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetUp nonescape`

Proof. Start from the finite carrier of Definition 611.1. The displayed rows of B are exactly $E, W, D, R, S, H, C, P, N$.

Apply the local NameCert obligations to identify those rows as the complete naming surface. Window coverage binds the dyadic readbacks and radii to the finite windows selected by the precision request.

The regular-sequence handoff theorem routes source-facing access through the same finite observations. The seal-boundary theorem then routes every completion-facing consumer through S without exposing any unlisted analytic object.

Thus the observation budget is a finite BEDC packet. Its consumers can use the displayed rows, but the carrier gives them no route to quotient streams, host real equality, ambient completeness, hidden limit selection, or unbounded observation schedules. □

Theorem 611.7 (Real observation budget obligation package)**.** *The* $\mathsf{RealObservationBudget}^{\uparrow}$ *obligation package is the finite surface*

$$B = (E, W, D, R, S, H, C, P, N)$$

over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It packages carrier admission, unary precision exposure, finite window coverage, dyadic readback and radius exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *seal-boundary exposure, componentwise transport, displayed route replay, provenance, and the local naming row, with no quotient stream equality, host real equality, ambient completeness, hidden limit selection, or unbounded observation schedule.*

Proof. Start from the carrier and the local NameCert obligation theorem. Definition 611.1 supplies exactly $E, W, D, R, S, H, C, P, N$, and Definition 611.2 reads those rows as the local naming surface.

The window, dyadic, regular-rational, and real-facing clauses are already separated by the existing obligation rows. Definition 611.3 binds D to the finite windows W, Definition 611.4 routes R through the finite observation route, and Definition 611.5 marks S as the completion-facing boundary.

Finally apply non-escape. Definition 611.6 rules out every unlisted analytic object, so the obligation package is exhausted by the finite BEDC surface displayed above. □

Theorem 611.8 (Real observation budget transport stability)**.** *Classifier transport for the* $\mathsf{RealObservationBudget}^{\uparrow}$ *obligation package preserves the same precision request, finite window family, dyadic readbacks and radii, regular-sequence handoff rows, real-seal boundary rows, provenance, and local name. Thus any transported package remains a finite observation-budget surface over the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project the two classified budget packets to their displayed coordinates. The transport ledger H is componentwise: it transports the precision request, window rows, dyadic ledger, handoff rows, seal rows, provenance, and local name without changing the shape of the packet.

Use the displayed routes C to replay the observation order after transport. The route from E to W, from W to D, from D to R, and from R to S is part of the same finite package, so transported consumers still read the same obligation surface.

The non-escape theorem excludes hidden replacement data during transport. No transported package can insert quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded schedule because none of those rows exists in the source package. □

Theorem 611.9 (Real observation budget consumer exactness)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{RealObservationBudget}^{\uparrow}$ *obligation package reads exactly the finite packet rows required by the observation budget:* $E, W, D, R, S, H, C, P, N$*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *side factors through the handoff rows* R*, the* $\mathsf{Real}^{\uparrow}$ *side factors through the seal rows* S*, and both sides remain inside the obligation package of Definition 611.7.*

Proof. First read the source-facing route. Definition 611.4 forces every regular-sequence handoff through the precision request, finite windows, dyadic readbacks and radii, transports, routes, provenance, and local naming row.

Next read the completion-facing route. Definition 611.5 sends every $\mathsf{Real}^{\uparrow}$ consumer through the seal rows S, and those seal rows are reached only after E, W, D, R, H, C, P, N have been displayed.

Combine these routes with transport stability and non-escape. The transported package keeps the same finite rows, and Definition 611.6 blocks any consumer from reading quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded observation schedule outside the package. □

Theorem 611.10 (Real observation budget prefix pullback determinacy)**.** *Let* $B = (E, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing prefix pullback that reads the same unary precision row* E *and the same finite* $\mathsf{StreamName}^{\uparrow}$ *window family*

W determines the same dyadic readback ledger D, the same $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows R, the same* $\mathsf{Real}^{\uparrow}$ *seal rows S, and the same consumer route through H, C, P, N. Thus prefix pullbacks through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces cannot split into different finite consumer routes once E and W have been fixed.*

Proof. Project the accepted packet to the carrier of Definition 611.1. The prefix pullback is therefore read against the displayed rows $E, W, D, R, S, H, C, P, N$, with no extra analytic coordinate available to the consumer.

Use window coverage first. Definition 611.3 binds every dyadic readback and displayed radius in D to the finite windows W selected by E, so two pullbacks with the same E and W must read the same dyadic ledger.

Next pass through the downstream rows. Definition 611.4 binds the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R to D, and Definition 611.5 binds the $\mathsf{Real}^{\uparrow}$ seal S to that handoff boundary.

Finally apply consumer exactness and non-escape. Definition 611.9 forces the route through H, C, P, N, while Definition 611.6 rules out quotient stream equality, host real equality, hidden limit selection, and unbounded schedules as alternate sources of a different route. □

Theorem 611.11 (Real observation budget StreamName–RegSeq–Real exit)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the finite chain consisting of the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and radii,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and displayed seal rows. Along this route the consumer receives the packet coordinates $E, W, D, R, S, H, C, P, N$ and introduces no quotient stream equality, host real equality, hidden limit selector, or unbounded observation schedule.*

Proof. Start with the finite packet of Definition 611.1. The $\mathsf{StreamName}^{\uparrow}$ dependency is the selected window family W, and Definition 611.3 binds every $\mathsf{DyadicRatCore}^{\uparrow}$ readback and displayed radius in D to one of those finite windows selected by the precision row E.

Next pass through the rational handoff. Definition 611.4 sends every $\mathsf{RegSeqRat}^{\uparrow}$ consumer row through the already displayed E, W, D route together with the transport, continuation, provenance, and local naming rows H, C, P, N.

Now read the completion boundary. Definition 611.5 makes the seal rows S the only $\mathsf{Real}^{\uparrow}$ boundary reachable by a completion-facing consumer, and those rows may read only the finite observation and handoff surface already exposed by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Finally apply non-escape. Definition 611.6 exhausts the downstream consumer surface by $E, W, D, R, S, H, C, P, N$, so the exit route cannot import quotient stream equality, host real equality, hidden limit selection, or an unbounded observation schedule from outside the displayed packet. □

Theorem 611.12 (Real observation budget request totality)**.** *Every accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *budget request carries a finite unary precision row E, a finite selected* $\mathsf{StreamName}^{\uparrow}$ *window family W, and matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and radius rows D. These rows are routed by the displayed* Cont *coordinate C, so each accepted request exposes a finite path*

$$E \longmapsto W \longmapsto D$$

inside the same packet $B = (E, W, D, R, S, H, C, P, N)$.

Proof. Project the accepted carrier to its displayed rows. The carrier definition supplies the precision row E, the finite window family W, the dyadic ledger D, and the continuation coordinate C as components of one finite BHist packet.

Use window coverage for the request-facing content. Definition 611.3 says that every dyadic readback and displayed radius exported by D is covered by one of the finite windows in W, and that those windows are selected by E through the routes in C.

Finally apply non-escape. Since Definition 611.6 leaves no coordinate for quotient stream equality, host real equality, hidden limit selection, or an unbounded schedule, an accepted request cannot obtain its readback or radius rows from outside the displayed finite route. □

Theorem 611.13 (Real observation budget route determinacy)**.** *Let two accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer routes display the same rows* $E, W, D, R, S, H, C, P, N$*. Then they determine the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *observation-budget surface up to the componentwise* hsame *transports displayed in* H*.*

Proof. Begin with the request and dyadic part. Request totality identifies the finite path from E through W to D, and window coverage fixes the dyadic readbacks and radii as the observations selected by those windows.

Pass to the rational-source handoff. Definition 611.4 routes the $\mathsf{RegSeqRat}^{\uparrow}$ surface through the same E, W, D, H, C, P, N rows, so two routes with the same displayed packet have the same handoff surface after the hsame transports in H are applied.

Pass to the completion-facing boundary. Definition 611.5 forces every $\mathsf{Real}^{\uparrow}$ consumer through the seal rows S, and transport stability preserves those seal rows with the same provenance and local name. Non-escape rules out any hidden selector that could separate the two displayed routes. □

Theorem 611.14 (Real observation budget diagonal Cauchy handoff totality)**.** *Every synchronized diagonal completion read of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet first consumes the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *precision row, mesh row, and comparison row, and only then factors through the finite observation route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The exposed surface consists only of finite windows, dyadic readbacks and radii, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows, with the structural transport, route, provenance, and local naming rows attached. The handoff exports no quotient stream equality, host real equality, ambient completeness principle, hidden limit selection, or unbounded schedule.*

Proof. Begin with request totality and window coverage. Definition 611.12 places every accepted precision request on the displayed route through E, W, D, and Definition 611.3 makes the dyadic readbacks and radii finite window data.

Next attach the completion-facing handoff. Definition 611.4 routes the rational source through the $\mathsf{RegSeqRat}^{\uparrow}$ rows, and Definition 611.5 sends the resulting completion read through the $\mathsf{Real}^{\uparrow}$ seal rows.

Finally use route determinacy with the synchronized diagonal sibling. Definition 611.13 fixes the displayed route once $E, W, D, R, S, H, C, P, N$ are fixed, while the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ sibling of chapter 853 supplies exactly the finite precision, mesh, and comparison rows for the diagonal budget. Non-escape then refuses quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded schedules outside the displayed handoff. □

Theorem 611.15 (Real observation budget TasteGate-visible surface)**.** *Any* $\mathsf{ChapterTasteGate}$ *witness for this AI-origin* $\mathsf{RealObservationBudget}^{\uparrow}$ *chapter may inspect only the finite packet rows* $E, W, D, R, S, H, C, P, N$ *over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. It may not read quotient streams, host real equality, ambient completeness, hidden limit selection, or an unbounded schedule.*

Proof. Read the chapter origin and the carrier together. The chapter is AI-origin, and Definition 611.1 gives the complete carrier as the finite packet $B = (E, W, D, R, S, H, C, P, N)$ over the listed BEDC rows.

The request-totality and route-determinacy theorems identify the visible surface available to consumers. The first binds each accepted request to E, W, D through the displayed Cont route, while the second shows that the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ surfaces are determined by the same packet up to the displayed hsame transports.

Apply the ground-compiler alignment and non-escape results. Definition 611.16 restricts $\mathsf{ChapterTasteGate}$ consumption to displayed rows, and Definition 611.6 excludes quotient streams, host real equality, ambient completeness, hidden limit selection, and unbounded schedules from the packet. □

Theorem 611.16 (Real observation budget TasteGate ground-compiler alignment)**.** *For an AI-origin* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet*

$$B = (E, W, D, R, S, H, C, P, N),$$

the rows $E, W, D, R, S, H, C, P, N$ *are exactly the single-carrier* $\mathsf{ChapterTasteGate}$ *surface available to the chapter. The precision request, finite windows, dyadic readbacks and radii, regular-sequence handoff rows, real-seal rows, transports, continuation routes, provenance, and local name must be consumed through the ground-compiler event-flow and no-hidden-input discipline of chapter 9207. In particular, a consumer may not replace those rows by hidden prompt state, a host-side limit selector, an unlisted compiler fact, quotient stream equality, host real equality, ambient completeness, or an unbounded observation schedule.*

Proof. Start from the carrier definition and the local NameCert surface. Definition 611.1 exposes exactly $E, W, D, R, S, H, C, P, N$, while Definition 611.2 reads those coordinates as the whole local certificate.

The ground-compiler boundary is then a consumption rule for the same finite packet. The event-flow and no-hidden-input discipline of chapter 9207 permits only displayed, recognized rows to enter the compiler-facing read, so the $\mathsf{ChapterTasteGate}$ row cannot import an extra semantic channel.

Finally apply non-escape. Definition 611.6 already rules out quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded schedules; the ground-compiler reading adds that hidden prompt state and unnamed compiler facts are not admissible input rows either. □

Theorem 611.17 (Real observation budget finite-window seal-route totality)**.** *Every admitted* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet is reached by the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The route is carried with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ *row. There is no admitted seal read that bypasses the precision request, finite window coverage, dyadic ledger, regular-sequence handoff, or displayed provenance rows.*

Proof. First use window coverage. Definition 611.3 sends every exported dyadic readback and radius through the precision request E, the finite window family W, and the dyadic ledger D.

Next use the regular-sequence handoff theorem. Definition 611.4 places the handoff rows R downstream of E, W, D and carries the same transport, route, provenance, and naming data.

Finally apply the real-seal boundary theorem. Definition 611.5 makes S the only completion-facing boundary, and transport stability keeps the same hsame, Cont, Pkg, and NameCert rows attached to the route. □

Theorem 611.18 (Real observation budget obligation closure readiness)**.** *The* $\mathsf{RealObservationBudget}^{\uparrow}$ *paper-side obligation package consists of the carrier admission of* $B = (E, W, D, R, S, H, C, P, N)$*, the local* NameCert *obligations, finite window coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *seal-boundary routing, non-escape, transport stability, consumer exactness,* ChapterTasteGate *ground-compiler alignment, and finite-window seal-route totality. This package is precisely the obligation surface required before a later closure claim may promote the AI-origin observation budget beyond seed status.*

Proof. Begin with the carrier and NameCert rows. Definition 611.7 assembles carrier admission, local naming, window, handoff, seal, transport, route, provenance, and non-escape obligations into one finite surface.

Add the consumer-facing rows. Definition 611.8 preserves the same packet under classifier transport, and Definition 611.9 says the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers read exactly that packet.

Add the AI-origin readiness rows. Definition 611.16 ties the single-carrier ChapterTasteGate surface to ground-compiler event-flow and no-hidden-input consumption, while Definition 611.17 makes every admitted seal read pass through E, W, D, R, S with the displayed transport and provenance rows. □

Theorem 611.19 (Real observation budget classifier exactness)**.** *For an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet*

$$B = (E, W, D, R, S, H, C, P, N),$$

the classifier is exact on the displayed rows $E, W, D, R, S, H, C, P, N$*. Two consumer reads agree at the observation-budget classifier exactly when they agree on the visible precision request* E*, finite window family* W*, dyadic readback and radius rows* D*, regular-sequence handoff rows* R*, real-seal rows* S*, componentwise transports* H*, displayed routes* C*, provenance* P*, and local* NameCert *row* N*. The comparison does not inspect hidden prompt state, host real equality, quotient stream equality, ambient completeness, or an unlisted compiler fact.*

Proof. Start with the carrier definition and the local NameCert obligation surface. These expose precisely the rows $E, W, D, R, S, H, C, P, N$ and give no coordinate for a hidden classifier input.

Use consumer exactness and transport stability. Consumer exactness confines the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads to the displayed packet, while transport stability preserves the same componentwise rows under hsame replay.

Finally apply the ground-compiler alignment and non-escape theorems. Ground-compiler alignment forbids prompt-state or unnamed compiler facts from entering the classifier, and non-escape excludes quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded schedules. □

Theorem 611.20 (Real observation budget finite-route coverage)**.** *Every admitted seal read of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet*

$$B = (E, W, D, R, S, H, C, P, N)$$

is covered by the finite route

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The route carries the same componentwise transport row H, displayed route row C, package provenance P, and local NameCert *row N. No seal read is admitted directly from E, W, D, or R, and no seal read bypasses the displayed finite-window and regular-sequence handoff rows.*

Proof. First read the precision and finite-window obligations. Window coverage sends each dyadic readback and radius through E, W, D before any regular-sequence handoff is available.

Next apply the regular-sequence handoff theorem. It places R downstream of the precision, window, and dyadic rows, with the same transport, route, provenance, and local naming data attached.

The real-seal boundary then makes S the only completion-facing row. Since non-escape permits no hidden seal coordinate, every admitted seal read follows the displayed route $E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$. □

Theorem 611.21 (Real observation budget root-unblock package)**.** *The root-unblock package exposed by* $\mathsf{RealObservationBudget}^{\uparrow}$ *over*

$$B = (E, W, D, R, S, H, C, P, N)$$

consists exactly of carrier admission, local NameCert *obligations, non-escape, finite-route determinacy through $E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$, classifier exactness on displayed rows, and* ChapterTasteGate *ground-compiler alignment. This package is the paper-side surface available to downstream* $\mathsf{Real}^{\uparrow}$ *completion consumers, without claiming a checked TasteGate target.*

Proof. Begin with carrier admission and local naming. The carrier definition and NameCert obligation theorem give the complete finite packet $B = (E, W, D, R, S, H, C, P, N)$.

Add the route and classifier obligations. Finite-route coverage forces each seal read through E, W, D, R, S, and classifier exactness restricts comparison to the same displayed rows.

Finally add non-escape and ground-compiler alignment. These two results prevent hidden prompt state, unlisted compiler facts, quotient stream equality, host real equality, ambient completeness, and unbounded schedules from acting as completion inputs, while leaving the TasteGate Lean target as a separate verification obligation. □

Theorem 611.22 (Real observation budget TasteGate-cited obligation)**.** *Any promotion of the AI-origin* $\mathsf{RealObservationBudget}^{\uparrow}$ *chapter beyond seed closure must consume the real* ChapterTasteGate *declaration* `BEDC.Derived.RealObservationBudgetUp.taste_gate`*. The resolving declaration records the TasteGate citation for the finite carrier rows $E, W, D, R, S, H, C, P, N$ while the remaining observation-budget obligations stay on their paper-side rows.*

Lean4 ✓ `BEDC.Derived.RealObservationBudgetUp.taste gate`

Proof. Read the origin row in the closure status together with the carrier definition. The chapter is AI-origin, and the only displayed observation-budget packet is $B = (E, W, D, R, S, H, C, P, N)$.

Apply the TasteGate-visible surface and ground-compiler alignment theorems. They say that a ChapterTasteGate consumer may inspect only the displayed observation-budget rows under the ground-compiler event-flow and no-hidden-input discipline.

The paper-Lean name channel is stricter than the paper theorem site. The named Lean declaration supplies the TasteGate citation, while the observation-budget carrier, window, handoff, seal, transport, classifier, and non-escape obligations remain distinct from that citation. □

Theorem 611.23 (Real observation budget AI-origin obligation package)**.** *The AI-origin obligation package for* $\mathsf{RealObservationBudget}^{\uparrow}$ *is the finite packet consisting of carrier admission, finite window coverage,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, non-escape, transport stability, consumer exactness, and ground-compiler alignment over* $E, W, D, R, S, H, C, P, N$*. This package is the whole observation-budget surface carried alongside the resolving TasteGate declaration.*

Proof. Start with the carrier and local NameCert obligations. They expose the precision request, finite windows, dyadic readback and radius rows, regular-sequence handoff rows, real-seal rows, transport, routes, provenance, and local name.

Add the analytic-facing obligations. Window coverage, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the $\mathsf{Real}^{\uparrow}$ seal boundary place every completion-facing read on the finite route through E, W, D, R, S.

Finally apply non-escape, transport stability, consumer exactness, and ground-compiler alignment. These results preserve the same row set under hsame transport, exhaust the consumer read, and prevent hidden prompt state, quotient stream equality, host real equality, or ambient completeness from entering the AI-origin package. □

Theorem 611.24 (Real observation budget seed-to-obligation readiness)**.** *The* $\mathsf{RealObservationBudget}^{\uparrow}$ *chapter carries an obligation-closure ledger once the TasteGate target is present and the finite rows* $E, W, D, R, S, H, C, P, N$ *are the sole admissible carrier surface. The paper-side obligations display the observation request, window, dyadic, regular-sequence, real-seal, transport, route, provenance, and naming rows without asserting scoped closure or Lean verification for those rows.*

Proof. Use Definition 611.23 to assemble the finite obligation package. It gives exactly the rows $E, W, D, R, S, H, C, P, N$ and the named coverage, handoff, seal, non-escape, transport, consumer, and compiler-alignment obligations.

Use Definition 611.22 for the AI-origin gate. It requires the real TasteGate declaration before the paper can cite a checked TasteGate target.

Thus the obligation boundary has two ingredients: the finite row surface remains exhaustive, and the TasteGate target resolves. Together they record the paper-side obligation ledger while keeping the remaining proof and verification axes separate. □

Theorem 611.25 (Real observation budget finite-window ledger resolution)**.** *For an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier*

$$B = (E, W, D, R, S, H, C, P, N),$$

every finite-window ledger read used by a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer is resolved by* E, W, D *together with* H, C, P, N *before* R *or* S *is consumed. No such read may use quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded schedule.*

Proof. First read the request and window side. Request totality exposes the finite path from E to W and then to D, while window coverage binds every dyadic readback and displayed radius to the windows selected by the unary precision row.

Next use the consumer-facing handoff theorems. The regular-sequence handoff places R downstream of E, W, D, H, C, P, N, and the real-seal boundary places S downstream of the same finite route.

Finally apply classifier exactness and non-escape. Classifier exactness confines comparison to the displayed packet rows, and non-escape excludes quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded schedules from the ledger. □

Theorem 611.26 (Real observation budget window-request totalization)**.** *For an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier*

$$B = (E, W, D, R, S, H, C, P, N),$$

every admitted $\mathsf{StreamName}^{\uparrow}$ *window in* W*, every* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback in* D*, and every displayed dyadic radius in* D *is selected by the unary precision request* E *through the displayed* Cont *route*

$$E \longmapsto W \longmapsto D.$$

If such a row is consumed by a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *reader, the same finite request route continues through* R *and* S *with the stored* hsame*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ *rows.*

Proof. Start with the carrier definition. Definition 611.1 gives the only displayed packet rows $E, W, D, R, S, H, C, P, N$, and the continuation coordinate C records the route from the precision request to windows and then to dyadic readbacks and radii.

Use the window and handoff theorems. Definition 611.3 binds every dyadic readback and radius in D to a finite window in W selected by E, while Definition 611.4 sends every regular-rational read through the same request, window, and dyadic rows.

Finally apply finite-route coverage and non-escape. Definition 611.20 extends the route to S for admitted seal reads, and Definition 611.6 prevents any window, readback, radius, handoff, or seal row from being supplied by an unlisted schedule or host-side selector. □

Theorem 611.27 (Real observation budget seal-ledger exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read admitted by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier factors through the finite ledger rows*

$$E, W, D, R, S, H, C, P, N.$$

The seal ledger has no row for host real equality, hidden limit selection, ambient completeness, quotient stream equality, or an unbounded observation schedule.

Proof. Project the finite packet from Definition 611.1. The only real-facing coordinate is S, and the only rows attached to it are the request, window, dyadic, handoff, transport, route, provenance, and local naming rows.

Apply the seal boundary and route-totality results. Definition 611.5 makes S the completion-facing boundary, while Definition 611.17 and Definition 611.20 force every admitted seal read through $E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$.

Use non-escape for the exhaustion claim. Since Definition 611.6 excludes host real equality, hidden limit selection, ambient completeness, quotient stream equality, and unbounded schedules from the carrier, the displayed rows exhaust the seal ledger. □

Theorem 611.28 (Real observation budget NameCert field saturation)**.** *The carrier field, classifier field, componentwise transport field, displayed route field, provenance field, and local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}}$ *field of an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet exhaust the paper-side obligation surface. A downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer can request only the rows* $E, W, D, R, S, H, C, P, N$ *and their displayed classifier, transport, route, provenance, and naming evidence.*

Proof. First read the local certificate. Definition 611.2 identifies carrier admission, local naming, window coverage, handoff routing, seal-boundary routing, transport, route replay, and provenance as the complete local $\mathsf{NameCert}$ surface.

Next add classifier and transport exactness. Definition 611.19 confines classifier comparison to the displayed rows, and Definition 611.8 preserves the same rows under componentwise hsame replay.

Finally combine finite-route coverage with the AI-origin obligation package. Definition 611.20 routes each seal read through the finite ledger, and Definition 611.23 packages exactly these carrier, classifier, transport, route, provenance, and naming fields before any TasteGate citation is available. □

Theorem 611.29 (Real observation budget TasteGate target consumption)**.** *Any* RealObservationBudget$^{\uparrow}$ *closure claim beyond seed status must consume the resolving* ChapterTasteGate *carrier-alignment theorem named* `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetTasteGate_single_carrie` *The resolving marker is the TasteGate carrier-alignment theorem for the displayed packet*

$$E, W, D, R, S, H, C, P, N.$$

The checked TasteGate marker does not discharge the remaining observation-budget rows, and no downstream Real$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *consumer may treat prose, host equality, quotient stream equality, hidden limit selection, ambient completeness, or an unbounded observation schedule as a substitute for those obligations.*

Lean4 ✓ `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetTasteGate single carrier alignment`

Proof. Read the AI-origin row together with the carrier definition. The chapter declares its origin as AI, and Definition 611.1 gives the only carrier packet $B = (E, W, D, R, S, H, C, P, N)$.

Use the TasteGate-visible surface and the ground-compiler alignment theorem. Definition 611.16 says the ChapterTasteGate read is confined to the displayed rows under ground-compiler event-flow and no-hidden-input discipline.

Finally apply the paper-Lean marker rule. The resolving theorem named `BEDC.Derived.RealObservationBudg` supplies the TasteGate citation required for the AI-origin chapter to record its paper-side obligation surface, while the remaining carrier, window, handoff, seal, transport, classifier, and non-escape proofs stay on their separate obligation rows. □

Falsifiable prediction: A counterexample to the observation-budget route is an accepted RealObservationBudget$^{\uparrow}$ seal read reaching S without passing through the displayed rows E, W, D, R, H, C, P, N.

Independence witness: real-observation-budget, observationbudgetlimiter, regular-cauchy-tail-budget, cauchy-seal-budget-synchronizer, finite-observation-budget-selector: none is bijective to RealObservationBudget$^{\uparrow}$ because the observation-budget packet carries the joint field rows $E, W, D, R, S, H, C, P, N$, while each sibling omits at least one request, window, dyadic, regular handoff, seal, transport, route, provenance, or local naming field needed by this carrier.

Theorem 611.30 (Real observation budget field-faithful commitment)**.** *Any Lean TasteGate file that discharges the AI-origin* RealObservationBudget$^{\uparrow}$ *carrier-alignment surface must supply explicit* `Nontrivial RealObservationBudgetUp` *and* `FieldFaithful RealObservationBudgetUp` *instances over the displayed fields*

$$E, W, D, R, S, H, C, P, N.$$

The commitment is a companion obligation for the chapter template: the checked TasteGate carrier-alignment target may record the displayed carrier, but it does not by itself make the carrier non-trivial or field-faithful.

Proof. Read the AI-origin row and the carrier definition. The chapter is explicitly AI-origin, and Definition 611.1 displays the whole observation-budget packet as the fields $E, W, D, R, S, H, C, P, N$.

Use the TasteGate target-consumption theorem only as a boundary. Definition 611.29 says the resolving carrier-alignment target is consumed by closure claims, but it also keeps the remaining observation-budget obligations separate from that marker.

Therefore a non-vacuous TasteGate discharge must carry the two companion instances on the same displayed fields. Without `Nontrivial RealObservationBudgetUp`, the carrier could collapse to a one-object reading; without `FieldFaithful RealObservationBudgetUp`, the fields could be ignored while the chapter still names the carrier. □

Theorem 611.31 (Real observation budget RegSeqRat RealUp consumer factorization)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *carrier factors through the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

with the displayed hsame *row* H*,* Cont *row* C*,* Pkg *row* P*, and local* $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$ *row* N*. The factorization has no read channel for quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded schedule.*

Proof. First compose the request and window coverage. Definition 611.26 and Definition 611.3 route each admitted dyadic readback and radius through E, W, D.

Next attach the consumer handoff rows. Definition 611.4 places $\mathsf{RegSeqRat}^{\uparrow}$ consumption at R, while Definition 611.5 and Definition 611.27 place $\mathsf{Real}^{\uparrow}$ consumption at the seal row S.

Transport stability then preserves the same evidence rows. Definition 611.8 keeps the componentwise hsame, displayed Cont, Pkg provenance, and local $\mathsf{NameCert}$ data attached to the route.

Finally use non-escape and classifier exactness. Definition 611.6 and Definition 611.19 exclude quotient stream equality, host real equality, ambient completeness, hidden limit selection, and unbounded schedules from every admitted consumer read. □

Theorem 611.32 (Real observation budget L10 gate handoff exactness)**.** *Every admitted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet exposes the L10 phase handoff exactly through the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S,$$

where W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and radius surface,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff surface, and* S *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary. No* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer admitted by the packet may bypass this route by quotient stream equality, host real equality, hidden limit selection, ambient completeness, or an unbounded observation schedule. Thus the L10 gate is a finite BEDC route over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the observation route. Definition 611.3 binds every dyadic readback and radius in D to the finite $\mathsf{StreamName}^{\uparrow}$ windows W selected by the request row E.

Attach the rational handoff and the completion boundary. Definition 611.4 places the $\mathsf{RegSeqRat}^{\uparrow}$ consumer surface at R downstream of E, W, D, and Definition 611.5 makes S the only $\mathsf{Real}^{\uparrow}$ seal boundary reachable from that handoff.

Now use the consumer factorization theorem. Definition 611.31 says every admitted $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer factors through

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

with the displayed hsame, Cont, Pkg, and local NameCert rows.

Finally read the TasteGate target as a boundary condition, not as an extra analytic channel. Definition 611.29 keeps the checked target attached to the displayed packet and does not add quotient stream equality, host real equality, hidden limit selection, ambient completeness, or an unbounded observation schedule. The L10 handoff is therefore exactly the finite BEDC route over BHist, hsame, Cont, Pkg, NameCert, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$. □

Theorem 611.33 (Real observation budget finite-window seal lock)**.** *Let $B = (E, W, D, R, S, H, C, P, N)$ be an accepted* RealObservationBudget$^{\uparrow}$ *packet. Every* Real$^{\uparrow}$ *seal read admitted through S first passes through the finite* StreamName$^{\uparrow}$ *windows W selected by E, then through the* DyadicRatCore$^{\uparrow}$ *readback and radius rows D, and then through the* RegSeqRat$^{\uparrow}$ *handoff row R. The rows H, C, P, N preserve the same packet while transport, continuation replay, provenance, and local naming are consumed. Thus the seal boundary is a reusable finite-window consumer boundary for* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$*, with no channel for quotient stream equality, host real equality, ambient completeness, hidden limit selection, or an unbounded observation schedule.*

Proof. Begin with the displayed packet. By Definition 611.1, the only rows of an accepted RealObservationBudget$^{\uparrow}$ carrier are $E, W, D, R, S, H, C, P, N$.

Build the finite observation spine. Definition 611.3 places every admitted dyadic readback and radius row behind the finite StreamName$^{\uparrow}$ windows selected by E, and Definition 611.4 places the RegSeqRat$^{\uparrow}$ handoff at R after those rows.

Lock the completion-facing end of the route. Definition 611.5 makes S the Real$^{\uparrow}$ seal boundary reached from that handoff, while Definition 611.32 identifies the whole L10 route as

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The remaining rows do not create another analytic read. They preserve the same packet as componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming. Hence every seal read through S is locked to the finite-window route, and the excluded quotient, host equality, completeness, limit-selection, and unbounded-schedule channels have no coordinate in B. □

Theorem 611.34 (Real observation budget diagonal selector source lock)**.** *Let $B = (E, W, D, R, S, H, C, P, N)$ be an accepted* RealObservationBudget$^{\uparrow}$ *packet. Suppose a completion-facing consumer is synchronized with an accepted* CauchyDiagonalBudget$^{\uparrow}$ *route. Then the consumer can read the diagonal selector only through the displayed rows $E, W, D, R, S, H, C, P, N$ together with the* CauchyDiagonalBudget$^{\uparrow}$ *precision row, mesh row, and comparison row. The packet provides no read channel for a host-selected limit, quotient stream, ambient completeness witness, or hidden selector input.*

Proof. Project the finite carrier. By Definition 611.1, the only rows of B are $E, W, D, R, S, H, C, P, N$.

Use the observation route already forced by the local budget. Window coverage binds finite reads to E, W, D, the RegSeqRat$^{\uparrow}$ handoff binds rational-name consumption to R, and the Real$^{\uparrow}$ seal boundary binds completion-facing reads to S.

Now attach the synchronized diagonal sibling. The CauchyDiagonalBudget$^{\uparrow}$ route supplies the finite selector surface through its precision, mesh, and comparison rows, and the L10 handoff exactness of Definition 611.32 repackages those reads through E, W, D, R, S.

Finally apply non-escape at the seal boundary. Since neither the carrier nor the synchronized finite selector surface contains a host-selected limit, quotient stream, ambient completeness witness, or hidden selector input, the completion-facing consumer has no channel outside the displayed rows and the diagonal precision, mesh, and comparison rows. □

Theorem 611.35 (Real observation budget L10 four-face pullback)**.** *Let $B = (E, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet. Suppose the diagonal-selector row synchronized with B is consumed by a completion-facing consumer or by a* $\mathsf{Real}^{\uparrow}$ *consumer. Then the consumer route factors through the finite four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and the only supporting rows used by the factorization are the displayed BHist *observations, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *data attached to B and to those four faces.*

Consequently the combined observation-budget and diagonal-selector route has no channel for quotient stream equality, selected host limits, host real equality, countable-choice selectors, ambient completeness, metric-completeness rows, or a duplicate diagonal-budget face.

Proof. Project the carrier first. By Definition 611.1, the accepted packet exposes only the rows $E, W, D, R, S, H, C, P, N$, so any consumer read attached to B must begin with those displayed observation-budget coordinates.

Use the L10 handoff already forced by the local packet. Definition 611.32 identifies the observation-budget handoff as the finite route through E, W, D, R, S, with the displayed BHist, hsame, Cont, Pkg, and local NameCert support and with no hidden analytic route.

Now attach the diagonal selector. Definition 611.34 says that a completion-facing synchronized selector can only be read through the displayed packet rows together with the finite precision, mesh, and comparison rows of the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ route.

Finally pass to the Real L10 terminal surface. Definition 38.454 places any diagonal-budget row owned by a terminal $\mathsf{Real}^{\uparrow}$ read on the same finite packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while Definition 38.457 identifies that packet as the exact terminal boundary. The listed quotient, host-limit, host-equality, choice, completeness, metric-completeness, and duplicate-budget rows therefore have no coordinate in the combined consumer route. □

Theorem 611.36 (Real observation budget closure-axis obligation ledger)**.** *The obligation ledger for* $\mathsf{RealObservationBudget}^{\uparrow}$ *consists of carrier admission for $B = (E, W, D, R, S, H, C, P, N)$, the local* NameCert *obligations, classifier exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer factorization, non-escape, and* ChapterTasteGate *target consumption. These rows record the paper-side obligation surface without asserting scoped closure or Lean verification.*

Proof. Assemble the carrier and local certificate rows from Definition 611.1 and Definition 611.2. These rows give the finite packet and its local naming obligations.

Add the consumer and classifier obligations. Definition 611.19 confines comparison to the displayed rows, and Definition 611.31 supplies the finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ route.

Close the paper-side ledger with non-escape and TasteGate target consumption. Definition 611.6 rules out hidden analytic input, while Definition 611.29 keeps Lean-side checking separate from the obligation rows recorded here. □

Lean4 ✓ `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RealObservationBudget}^{\uparrow}$ packet is generated from a unary precision request, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks and radii, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealObservationBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 611.1, Definition 611.2, Definition 611.3, Definition 611.4, Definition 611.5, Definition 611.7, Definition 611.18, and Definition 611.29 scope the finite observation-budget carrier, precision windows, dyadic

readbacks, RegSeqRat$^{\uparrow}$ handoffs, Real$^{\uparrow}$ seal boundary, TasteGate instance `BEDC.Derived.RealObservationBudgetUp.taste_gate`, and TasteGate carrier-alignment theorem over BHist, hsame, Cont, Pkg, NameCert, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RealObservationBudget}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.RealObservationBudgetUp.RealObservationBudgetUp\_StdBridge`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, ambient completeness, hidden limit selection, unbounded observation schedules, or bridge checking.

Upgrade path: No further formal-axis upgrade remains: the standard bridge for the observation-budget packet is checked.

Chapter 612

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalFusion}^{\uparrow}$

The $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ packet is the finite naming certificate that joins the observation-budget route, the regular Cauchy witness schedule, and the schedule-fusion row before the $\mathsf{Real}^{\uparrow}$ diagonal seal is read. It presents one completion-facing handoff over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$, $\mathsf{RealScheduleFusion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while withholding quotient streams, host real equality, selected limits, and ambient completeness.

This fusion packet consumes the modulus-indexed witness-window route of chapter 593: the row W is not an independent schedule oracle, but the sibling witness-schedule carrier transported into the diagonal fusion route before the dyadic readback and $\mathsf{Real}^{\uparrow}$ seal are available.

Definition 612.1 (Cauchy diagonal fusion carrier)**.** *A* $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (E, B, W, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the precision request row whose tolerance selects the diagonal observation;*
2. *B is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row exposing the finite stream windows, dyadic readbacks, regular-sequence handoffs, and real-seal boundary for E;*
3. *W is the* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *row exposing the modulus-indexed witness window used by the diagonal read;*
4. *S is the* $\mathsf{RealScheduleFusion}^{\uparrow}$ *row that locks the schedule, window ledger, regular-rational readback, and seal handoff into one route;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row consumed by both the observation budget and the witness schedule;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the dyadic readback and before the real seal;*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *diagonal seal row exported to completion-facing consumers;*
8. *H records componentwise* hsame *transports for E, B, W, S, D, R, L and the structural rows;*
9. *C records the displayed* Cont *routes from precision request to observation budget, witness schedule, schedule fusion, dyadic readback, regular-rational handoff, and diagonal seal;*
10. *P is the* Pkg *provenance over the listed rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalFusion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, host real equality, a selected completed limit, an unbounded schedule, or an ambient completeness principle.

Lean4 (definition) `BEDC.Derived.CauchyDiagonalFusionUp.CauchyDiagonalFusionUp`

Theorem 612.2 (Cauchy diagonal fusion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalFusion}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (E, B, W, S, D, R, L, H, C, P, N)$, precision request exposure,* $\mathsf{RealObservationBudget}^{\uparrow}$ *observation routing,* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *witness-window routing,* $\mathsf{RealScheduleFusion}^{\uparrow}$ *schedule locking,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *diagonal seal exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Project the carrier of Definition 612.1. The rows E, B, W, S, D, R, L are the only mathematical coordinates, and H, C, P, N are the only transport, route, package, and naming coordinates.

The observation row B, witness row W, and schedule-fusion row S are read before any seal consumer reaches L. Thus the local certificate is assembled from the same finite route rather than from a host completion object.

Since quotient streams, host equality, selected limits, unbounded schedules, and ambient completeness have no row in F, no local NameCert projection can export them. □

Theorem 612.3 (Cauchy diagonal fusion schedule lock)**.** *For an accepted* $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ *carrier, every precision request routed through E reaches the same* $\mathsf{RealObservationBudget}^{\uparrow}$ *row B,* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ *row W, and* $\mathsf{RealScheduleFusion}^{\uparrow}$ *row S before reading the dyadic row D. The schedule fusion therefore fixes the finite window ledger and regular-rational readback used by the diagonal route.*

Proof. Start with the precision request row E. The displayed continuations in C send this request to B, W, S before D is available.

The transport row H carries the same observation budget, witness schedule, and schedule-fusion rows componentwise. Therefore a transported precision request cannot select a second window family or a second schedule after the diagonal route has begun. □

Theorem 612.4 (Cauchy diagonal fusion RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *diagonal seal read exported by an accepted* $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ *packet factors through the finite route*

$$E \longmapsto B \longmapsto W \longmapsto S \longmapsto D \longmapsto R \longmapsto L.$$

The seal receives only the precision request, observation-budget row, witness-schedule row, schedule-fusion row, dyadic readback, regular-rational handoff, transports, routes, provenance, and local naming data.

Proof. Apply the schedule-lock theorem to place every seal-facing request after E, B, W, S, D. The dyadic readback is therefore already tied to the displayed observation budget and witness schedule.

The regular-rational handoff row R is the only source-facing row between the dyadic readback and the seal. The continuation route then reaches L with H, C, P, N as visible structural data.

No further analytic object is present in the carrier, so the seal handoff cannot bypass the finite diagonal route. □

Theorem 612.5 (Cauchy diagonal fusion non-escape)**.** *An accepted* $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ *packet exports only the rows* $E, B, W, S, D, R, L, H, C, P, N$ *of Definition 612.1. It exports no quotient stream, no host real equality, no selected completed limit, no unbounded schedule, and no ambient completeness principle.*

Proof. Inspect the carrier rows. The route-visible data are the precision request, observation budget, witness schedule, schedule fusion, dyadic readback, regular-rational handoff, and diagonal seal, together with the structural rows H, C, P, N.

The NameCert obligation theorem identifies this list as the local naming surface. The schedule-lock theorem and seal-handoff theorem then force every downstream read to pass through that surface.

Because the excluded objects are not coordinates and are not produced by the displayed continuations, no consumer can recover them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalFusion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyDiagonalFusion}^{\uparrow}$ packet is generated from finite BHist rows for a precision request, observation-budget row, witness-schedule row, schedule-fusion row, dyadic readback, regular-rational handoff, and $\mathsf{Real}^{\uparrow}$ diagonal seal. The diagonal route is assembled bottom-up by Cont: the request reaches budget, witness, and schedule rows before the dyadic and seal rows can be read. Componentwise hsame keeps those route coordinates stable, and Pkg records the Real-observation, witness-schedule, dyadic, regular-rational, and seal dependencies. The fusion names one finite diagonal path, not an ambient completion theorem or selected limit.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalFusion}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the diagonal-fusion carrier, NameCert obligations, schedule lock, RealUp seal handoff, and non-escape theorem for $\mathsf{CauchyDiagonalFusion}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalFusion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient stream equality, host real equality, selected completed limits, unbounded schedules, choice selectors, or ambient completeness.

Upgrade path: Obligation closure requires Lean encodings of the diagonal carrier, schedule-lock theorem, seal-handoff route, and non-escape boundary.

Chapter 613

A Concrete Naming Certificate for $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$

The $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ packet records the finite route by which a transported stationary $\mathsf{Rat}^{\uparrow}$ classifier is read as a $\mathsf{Real}^{\uparrow}$ observation window. It keeps the rational seed, constant $\mathsf{StreamName}^{\uparrow}$ row, $\mathsf{RegSeqRat}^{\uparrow}$ seal-consumption row, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, real finite-window row, transport table, continuation routes, provenance, and local naming row in one finite boundary packet. It does not export quotient stream equality, selected limit data, host real equality, or an unbounded completion principle.

Definition 613.1 (Stationary diagonal window route carrier)**.** *A* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, S, G, D, R, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the transported stationary* $\mathsf{Rat}^{\uparrow}$ *seed row, including the two rational history carriers and their* $\mathsf{RatHistoryClassifier}$ *row;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *window row induced by Q;*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite seal-consumption row for the same window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger read from that window;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier row reached from S, G, D;*
6. *H stores componentwise* hsame *transports for Q, S, G, D, R, P, N;*
7. *C stores the displayed* Cont *routes from Q to S, from S to G, from S to D, and from D, S, G to R;*
8. *P records* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}}$ *row.*

The carrier is a finite route packet for stationary window reads. It contains no coordinate for quotient streams, selected real limits, host real equality, non-stationary tails, or ambient completeness.

Theorem 613.2 (Stationary diagonal window route NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWind}}$ *surface for Definition 613.1 consists exactly of carrier admission for* $W = (Q, S, G, D, R, H, C, P, N)$*, stationary* $\mathsf{Rat}^{\uparrow}$ *seed exposure, constant* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exposure,* $\mathsf{Real}^{\uparrow}$ *finite-window classifier exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed tuple W. The rows Q, S, G, D, R expose the stationary rational seed, constant stream window, regular-sequence seal row, dyadic observation ledger, and real-facing finite-window classifier.

The route obligations are read from C. They force every stream, regular-sequence, dyadic, and real-facing read to pass through the constant window induced by the same rational seed row Q.

The transport, provenance, and naming obligations are the rows H, P, N. Since the carrier has no coordinate for quotient streams, selected real limits, host equality, non-stationary tails, or ambient completeness, these displayed rows exhaust the local naming surface. □

Lean4 ✓ `BEDC.Derived.StationaryDiagonalWindowRouteUp.StationaryDiagonalWindowRoutePacket namecert obligations`

Theorem 613.3 (Stationary diagonal window route non-escape)**.** *Every transported stationary* $\mathsf{Real}^{\uparrow}$ *finite-window read exported by an accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier factors through the finite rows*

$$Q, S, G, D, R, H, C, P, N.$$

The consumer may inspect the $\mathsf{Rat}^{\uparrow}$ *seed classifier, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *finite-window classifier,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row, and nothing else.*

Proof. Project the accepted carrier and inspect its consumer-facing rows. The only route out of the stationary rational seed is the displayed continuation from Q to the constant stream window S.

The regular-sequence and dyadic reads are then forced by the routes from S to G and from S to D. The real-facing finite-window row R receives only those rows together with the transport, provenance, and naming rows.

Thus any certified finite observation of the transported stationary diagonal is exhausted by the displayed route packet. No quotient stream comparison, selected limit, host real equality, non-stationary tail, or ambient completeness datum can escape from the carrier. □

Theorem 613.4 (Stationary diagonal window route classifier stability)**.** *Let* $W = (Q, S, G, D, R, H, C, P, N)$ *and* $W' = (Q', S', G', D', R', H', C', P', N')$ *be accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carriers. If the component rows* Q, S, G, D, R, P, N *and* $Q', S', G', D', R', P', N'$ *are related by the displayed* hsame *transports stored in* H *and* H'*, then every classifier-visible comparison of the two route packets is componentwise: it compares the transported* $\mathsf{Rat}^{\uparrow}$ *seed rows, constant* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{Real}^{\uparrow}$ *finite-window rows, provenance rows, and local* NameCert *rows through those transports and through no other object.*

Proof. Project the two carriers with Definition 613.1. Their classifier-visible coordinates are exactly the displayed rows $Q, S, G, D, R, H, C, P, N$ and $Q', S', G', D', R', H', C', P', N'$.

Use the local NameCert obligations of Definition 613.2. They require all stationary seed exposure, constant-window exposure, seal consumption, dyadic ledger exposure, real finite-window exposure, transport, route replay, provenance, and naming reads to pass through the displayed packet rows.

Finally apply Definition 613.3. Since no quotient stream equality, selected limit, host real equality, non-stationary tail, or ambient completeness coordinate is present, the classifier stability claim is exhausted by componentwise hsame comparison of the named rows. □

Theorem 613.5 (Stationary diagonal window route constant-window exactness)**.** *For an accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier* $W = (Q, S, G, D, R, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S *is exactly the constant window induced by the transported stationary* $\mathsf{Rat}^{\uparrow}$ *seed row* Q*. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read through* G*, every* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger read through* D*, and every* $\mathsf{Real}^{\uparrow}$ *finite-window read through* R *consumes that same constant window and does not select a non-stationary tail.*

Proof. Begin with the carrier definition. The row Q is the transported stationary rational seed, and S is declared as the constant $\mathsf{StreamName}^{\uparrow}$ window induced by that seed.

Read the route row C. The route obligations in Definition 613.2 force the route from Q to S, from S to G, from S to D, and from D, S, G to R.

The non-escape theorem then rules out any alternate source for the window. A consumer cannot replace the constant window by quotient stream equality, a selected limit, host real equality, a non-stationary tail, or ambient completeness data, so the window consumed by G, D, R is the one induced by Q. □

Theorem 613.6 (Stationary diagonal window route sibling dependency)**.** *For an accepted* StationaryDiagonalWindo *carrier* $W = (Q, S, G, D, R, H, C, P, N)$*, the route is the route-level consumer of the sibling stationary-window boundary in chapter 626 and the transported stationary seed surface in chapter 475. It retains the sibling rows needed by the downstream regular-Cauchy window fusion consumer of chapter 693 while omitting any second classifier row, quotient stream, host real equality, non-stationary tail, or completeness principle.*

Proof. Project the accepted carrier W. Its displayed rows expose the transported stationary seed Q, the constant stream window S, the seal row G, the dyadic ledger D, the finite real-window row R, the component transports H, the continuation routes C, provenance P, and naming N.

Compare these rows with the stationary-window carrier. The sibling stationary-window boundary supplies the same constant-window shape at the Q and S rows, while the transported stationary diagonal surface supplies the seed table consumed before any real-facing read.

Apply Definition 613.5. It identifies the shared Q to S to G, D to R route as a constant-window route induced by the transported stationary seed.

The rows G, D, R, H, C, P, N are therefore the route-level material retained for the regular-Cauchy window fusion consumer. By Definition 613.3, no second classifier row, quotient stream, host real equality, non-stationary tail, or completeness principle is exported from the packet. □

Theorem 613.7 (Stationary diagonal window route seal-consumption handoff)**.** *An accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier* $W = (Q, S, G, D, R, H, C, P, N)$ *hands the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row* G *to the* $\mathsf{Real}^{\uparrow}$ *finite-window row* R *only through the displayed route packet. The handoff consumes* $Q, S, G, D, R, H, C, P, N$*: the stationary* $\mathsf{Rat}^{\uparrow}$ *seed, constant* $\mathsf{StreamName}^{\uparrow}$ *window, seal-consumption row, dyadic ledger, real window row, componentwise transports, continuation routes, package provenance, and local naming row.*

Proof. First project the accepted packet and isolate the handoff rows. The carrier supplies G as the regular-sequence seal-consumption row and R as the real-facing finite-window classifier row.

Next follow the displayed continuation routes. The NameCert obligations route Q to S, route S to G and D, and route D, S, G to R, with H supplying the componentwise transports and P, N supplying provenance and local naming.

The handoff has no further analytic coordinate. By non-escape, it cannot consume quotient streams, selected limits, host real equality, non-stationary tails, or ambient completeness, so the $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ handoff is exactly the finite packet $Q, S, G, D, R, H, C, P, N$. □

Theorem 613.8 (Stationary diagonal window route stationary seed exposure)**.** *For an accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier* $W = (Q, S, G, D, R, H, C, P, N)$*, the first readable analytic row is the transported stationary* $\mathsf{Rat}^{\uparrow}$ *seed row* Q*. The rows for* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *become readable only after the displayed continuation route from* Q *to the constant window row* S *has been exposed in* C*.*

Proof. Project the carrier of Definition 613.1. Its first coordinate is the transported stationary rational seed row Q, while the later analytic coordinates are S, G, D, R.

Now read the route coordinate C. The NameCert obligations in Definition 613.2 display the route from Q to S before any route to G, D, R is available.

The non-escape theorem supplies the boundary check. Since no quotient stream, selected real limit, host equality, non-stationary tail, or ambient completeness coordinate is present, a consumer cannot read a stream, regular-sequence, dyadic, or real row before passing through the stationary seed route in C. □

Theorem 613.9 (Stationary diagonal window route constant-window routing)**.** *For an accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier* $W = (Q, S, G, D, R, H, C, P, N)$*, the row* S *is the only constant* $\mathsf{StreamName}^{\uparrow}$ *window induced by the transported stationary seed* Q *inside the packet. The rows* G *and* D *are reachable only through the displayed packet coordinates* S, H, C, P, N*, and the finite real-facing row* R *consumes them only through the same route surface.*

Proof. The carrier definition identifies S as the constant stream window induced by Q. No second stream-window coordinate occurs in the tuple $W = (Q, S, G, D, R, H, C, P, N)$.

The route coordinate C then fixes the downstream access pattern. The only displayed routes out of the stream row are from S to the seal-consumption row G and from S to the dyadic ledger row D, with H carrying componentwise transport and P, N carrying provenance and naming.

Finally apply Definition 613.5. It rules out any non-stationary tail or alternate window source for the reads through G, D, R, so the constant window route is exactly the one carried by S, H, C, P, N. □

Theorem 613.10 (Stationary diagonal window route dyadic ledger exposure)**.** *Every* $\mathsf{Real}^{\uparrow}$ *finite-window read* R *exported by an accepted* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *carrier consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row* G *through the displayed route*

$$Q, S, G, D, R, H, C, P, N.$$

The read exports no quotient stream, selected real limit, host equality, non-stationary tail, or ambient completeness datum.

Proof. Begin with the handoff theorem Definition 613.7. It says that the regular-sequence seal row G reaches the real finite-window row R only through the finite route packet.

The dyadic ledger coordinate is tied to the same route. By Definition 613.2, the route row C displays S to D and D, S, G to R, while H, P, N provide transport, provenance, and local naming.

The non-escape theorem completes the exposure claim. The carrier has no coordinate from which a quotient stream, selected limit, host equality, non-stationary tail, or ambient completeness principle could be exported, so the real finite-window read is exactly the displayed dyadic-ledger and seal-consumption route. □

Theorem 613.11 (Stationary diagonal window route public export)**.** *For an accepted* StationaryDiagonalWindowRo *carrier*

$$W = (Q, S, G, D, R, H, C, P, N),$$

the rows $Q, S, G, D, R, H, C, P, N$ form the public $\mathsf{Real}^{\uparrow}$*-and-completion handoff from a transported stationary* $\mathsf{Rat}^{\uparrow}$ *seed through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier. Any consumer of the public export factors through that finite packet before reading a real-facing or completion-facing boundary.*

Proof. Start with the row inventory. The NameCert obligations of Definition 613.2 identify $Q, S, G, D, R, H, C, P, N$ as the carrier, stationary seed, constant window, seal-consumption, dyadic ledger, finite real-window, transport, route, provenance, and local naming surface.

Next route the stationary seed through the stream window. By Definition 613.5, the row S is exactly the constant $\mathsf{StreamName}^{\uparrow}$ window induced by Q, and every downstream G, D, R read consumes that same window.

Now attach the real-and-completion handoff. The seal-consumption handoff of Definition 613.7 sends G to R only through the displayed packet, while the dyadic ledger exposure of Definition 613.10 forces the D row into the same route.

Finally apply non-escape. Definition 613.3 excludes quotient streams, selected limits, host equality, non-stationary tails, and ambient completeness from the carrier, so any public real-facing or completion-facing consumer must factor through the finite rows $Q, S, G, D, R, H, C, P, N$. □

Theorem 613.12 (Stationary diagonal window route formal target)**.** *The public export of Definition 613.11 is the formal target for* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$*: every accepted carrier*

$$W = (Q, S, G, D, R, H, C, P, N)$$

exposes the stationary $\mathsf{Rat}^{\uparrow}$ *seed, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *finite-window row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row as the complete consumer surface.*

Proof. Start with the public export theorem. It identifies $Q, S, G, D, R, H, C, P, N$ as the finite real-facing and completion-facing handoff from the transported stationary rational seed through the stream, regular-sequence, dyadic, and real rows.

The dyadic ledger exposure theorem supplies the ledger side of the target surface. It forces every $\mathsf{Real}^{\uparrow}$ finite-window read to consume the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D together with the $\mathsf{RegSeqRat}^{\uparrow}$ seal row G through the displayed route.

The seal-consumption handoff supplies the regular-sequence side. It places G on the same finite route to R, with H, C, P, N carrying transport, continuation, provenance, and local naming.

Finally apply non-escape. Since the carrier exports no quotient stream, selected limit, host real equality, non-stationary tail, or ambient completeness coordinate, these displayed rows form the complete consumer surface for the later Lean target. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ packet is generated from a transported stationary $\mathsf{Rat}^{\uparrow}$ seed row, a constant $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ seal-consumption row, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation

ledger, a Real$^{\uparrow}$ finite-window classifier, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{StationaryDiagonalWindowRoute}^{\uparrow}}$, publicClosed)

Scope closed: Definition 613.1, Definition 613.2, Definition 613.3, Definition 613.8, Definition 613.9, Definition 613.10, and Definition 613.11 bind the public finite route packet for transported stationary RealUp window reads over Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{StationaryDiagonalWindowRoute}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.StationaryDiagonalWindowRouteUp.StationaryDiagonalWindowRoutePacket\_namecert\_obligati`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected real limits, host real equality, non-stationary tails, ambient completeness, or bridge checking.

Upgrade path: The next paper obligation is the bridge route for the same stationary Rat$^{\uparrow}$ to StreamName$^{\uparrow}$ to RegSeqRat$^{\uparrow}$ to DyadicRatCore$^{\uparrow}$ to Real$^{\uparrow}$ route packet.

Chapter 614

A Concrete Naming Certificate for $\mathsf{DecompilerRightInverse}^{\uparrow}$

The $\mathsf{DecompilerRightInverse}^{\uparrow}$ interface packages decompilation as a finite right-inverse packet for compiler certificates. It does not assert that every compiler is invertible. It records the special case where a target-to-source decompiler graph, followed by the displayed compiler graph, lands back in the target certificate up to the explicitly recorded optimisation row.

Definition 614.1 (Decompiler right-inverse carrier)**.** *A* $\mathsf{DecompilerRightInverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, T, K, \Psi, O, R, H, C, P, N)$$

with the following visible rows:

1. *S is the source* $\mathsf{NameCert}$ *row read by the compiler and produced by the decompiler;*
2. *T is the target* $\mathsf{NameCert}$ *row produced by the compiler and read by the decompiler;*
3. *K is the accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet carrying the finite compiler graph from S to T;*
4. *Ψ is the finite target-to-source decompiler graph row from T to S;*
5. *O is the target optimisation row witnessing that the composite $K \circ \Psi$ reads as a* $\mathsf{NameCert}_T$ *endomorphism modulo the displayed target classifier;*
6. *R is the paired right-inverse ledger matching each displayed target row with its decompile-then-compile image through O;*
7. *H records the componentwise* hsame *transports for source certificate rows, target certificate rows, compiler graph edges, decompiler graph edges, optimisation rows, right-inverse rows, package provenance, and the local naming row;*
8. *C records the displayed* Cont *routes for target inspection, decompiler graph readback, source certificate readback, compiler graph readback, optimisation readback, right-inverse comparison, and NameCert exposure;*
9. *P is the* Pkg *provenance row naming* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{NameCertMorphism}^{\uparrow}$, $\mathsf{CertificateCompiler}^{\uparrow}$, *target optimisation, and the decompiler right-inverse shape;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DecompilerRightInverse}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed compiler graph, decompiler graph, target optimisation row, paired right-inverse ledger, hsame *transports,* Cont *routes,* Pkg *provenance, and local naming row. It has no coordinate for a host decompiler function, semantic equivalence oracle, global compiler inverse, or unrecorded target behaviour. The compiler graph row K is consumed through the finite certificate-compiler surface of chapter 473; the decompiler packet adds only the reverse graph, optimisation row, and paired right-inverse ledger on top of that sibling surface.*

Theorem 614.2 (Decompiler right-inverse NameCert obligations)**.** *For the carrier of Definition 614.1, the local* $\mathsf{NameCert}_{\mathsf{DecompilerRightInverse}^{\uparrow}}$ *surface consists exactly of the source certificate row S, target certificate row T, accepted compiler packet K, finite compiler graph K, finite decompiler graph Ψ, target optimisation row O, paired right-inverse ledger R, componentwise* hsame *transports H, displayed* Cont *routes C, package provenance P, and local naming row N.*

Proof. Project the carrier. Each row required to read the decompiler packet is a coordinate of the finite BHist packet before any consumer can ask for the right-inverse comparison.

The compiler side is read through K, so the source-to-target direction is the same finite compiler-certificate graph already licensed by $\mathsf{CertificateCompiler}^{\uparrow}$. The reverse direction is read through Ψ, and it lands back in the displayed source certificate row S rather than in a host source semantics.

The right-inverse clause is the pair (O, R). The row O supplies the target optimisation surface for the composite $K \circ \Psi$, while R records the componentwise target-row comparison that consumes that optimisation row.

Finally the rows H, C, P, N expose transport, continuation, provenance, and local naming for the same packet. There is no remaining coordinate from which a host inverse, semantic equivalence oracle, or unrecorded compiler behaviour could be exported. □

Theorem 614.3 (Decompiler right-inverse non-escape boundary)**.** *Every accepted* $\mathsf{DecompilerRightInverse}^{\uparrow}$ *packet exports decompilation only as the displayed target-to-source graph Ψ together with the target optimisation row O and paired right-inverse ledger R. A consumer may use the packet only through $S, T, K, \Psi, O, R, H, C, P, N$, and cannot read it as a global compiler inverse, host semantic equality, quotient of programs, total decompiler, or optimiser theorem absent from the displayed rows.*

Proof. Inspect the carrier coordinates of Definition 614.1. The only reverse-direction data are the finite graph row Ψ, its routes through C, and its transported comparison rows through H.

The composite comparison is not a hidden inverse law. It is the displayed target optimisation row O together with the paired right-inverse ledger R, so every target read returns to the target certificate through finite packet data.

The package and local naming rows expose exactly those coordinates. Since the carrier has no row for host compiler semantics, program quotients, global invertibility, total decompiler behaviour, or unrecorded optimiser correctness, none of those objects can cross the boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecompilerRightInverse}^{\uparrow}}$.

Constructive story: A $\mathsf{DecompilerRightInverse}^{\uparrow}$ packet is generated from source and target NameCert rows, a finite compiler graph, a finite decompiler graph, target optimisation rows, paired right-inverse comparisons, displayed transport, continuation, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DecompilerRightInverse}^{\uparrow}}$, seed)

Scope closed: Definition 614.1, Definition 614.2, and Definition 614.3 name the finite right-inverse packet surface but do not yet bind a scoped public theorem.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DecompilerRightInverse}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close existence of decompilers for arbitrary compilers, global invertibility, host semantic equivalence, program quotients, total decompilation, or bridge checking.

Upgrade path: Scoped closure requires a checked finite right-inverse classifier theorem over the displayed compiler graph, decompiler graph, optimisation row, transport routes, package provenance, and local naming row.

Theorem 614.4 (Decompiler right-inverse target-row classifier)**.** *Let*

$$D_i = (S_i, T_i, K_i, \Psi_i, O_i, R_i, H_i, C_i, P_i, N_i) \quad (i = 0, 1)$$

be accepted $\mathsf{DecompilerRightInverse}^{\uparrow}$ *packets in the sense of Definition 614.1. For each i, let G_i be the displayed compiler graph read from the accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet K_i, and let $\rho_i(t_i)$ be a displayed target-row right-inverse comparison in R_i: it starts from a displayed target row t_i of T_i, follows the decompiler graph row Ψ_i to the source certificate, follows G_i back to the target certificate, applies the target optimisation row O_i, and then compares the result with t_i through the paired ledger row R_i.*

Assume the two reads expose the same classifier footprint: G_0 and G_1, Ψ_0 and Ψ_1, O_0 and O_1, and the ledger rows R_0 and R_1 are matched componentwise by the displayed hsame *transports, and the target inspection and right-inverse comparison routes in C_0 and C_1 read those matched rows. Then $\rho_0(t_0)$ is accepted by the* $\mathsf{DecompilerRightInverse}^{\uparrow}$ *target-row classifier exactly when $\rho_1(t_1)$ is accepted. Moreover, the common accepted comparison exports only the displayed compiler graph, decompiler graph, target optimisation row, paired right-inverse ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows; it exports no independent inverse, host decompiler function, semantic equality, quotient of programs, or unrecorded target behaviour.*

Proof. Project the carrier in Definition 614.1. A target-row right-inverse comparison has only one available route: target inspection through T_i, decompiler readback through Ψ_i, compiler readback through the graph displayed by K_i, optimisation readback through O_i, and paired comparison through R_i. These are precisely the rows named in the statement.

The compiler part of the route is finite graph data. By Definition 473.10, an accepted compiler-edge comparison is witnessed only by the displayed graph, landing, transported classifier, and displayed continuation rows of the $\mathsf{CertificateCompiler}^{\uparrow}$ packet. Thus the return from the decompiled source row to the target row is determined by the displayed compiler graph G_i and its recorded route data, not by a host compiler meaning.

The right-inverse part is the pair (O_i, R_i). By Definition 614.2, O_i supplies the target optimisation surface for the composite $K_i \circ \Psi_i$, while R_i records the componentwise target-row comparison that consumes that optimisation row. Hence, once G_i, Ψ_i, O_i, and R_i are fixed up to the displayed componentwise hsame transports and are read through the corresponding routes in C_i, the classifier has no further coordinate on which the two target-row comparisons could differ.

Therefore the comparison in D_0 is accepted exactly when the corresponding comparison in D_1 is accepted: both read the same finite target-row classifier footprint, transported componentwise through H_0 and H_1 and replayed through C_0 and C_1.

Finally apply the non-escape boundary Definition 614.3. It says that an accepted $\mathsf{DecompilerRightInverse}^{\uparrow}$ packet exports decompilation only as Ψ together with O and R, and that every consumer read remains inside $S, T, K, \Psi, O, R, H, C, P, N$. Since the classifier footprint above uses only those rows, the accepted comparison cannot export an independent inverse, host decompiler function, semantic equality, program quotient, or unrecorded target behaviour. □

Chapter 615

A Concrete Naming Certificate for $\mathsf{ObserverSubjectBoundary}^{\uparrow}$

The $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ carrier records the finite observation packet at which an observer-facing consumer may inspect histories without receiving a separate subject carrier. Its rows are histories, transported history-sameness, displayed continuation routes, probe and signature readback, package provenance, and a local naming row. The boundary is therefore a finite BEDC packet over observation rows, not a substance slot behind those rows.

The downstream $\mathsf{ObserverHistIdentity}^{\uparrow}$ surface of chapter 479 consumes this boundary before reading its H, O, L, C, P, N packet. The dependency is the absence of any observer-substance coordinate, so observer-history identity is read as a finite history-and-name packet rather than as a cosmetic reference to a hidden subject.

Definition 615.1 (ObserverSubjectBoundary carrier)**.** *An* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, H, C, B, S, P, N)$$

with the following displayed rows:

1. *R is the finite observation-row ledger, whose entries are* BHist *records of the histories inspected by the observer-facing consumer;*
2. *H stores componentwise* hsame *transport for the observation rows and for the local naming row;*
3. *C stores the displayed* Cont *routes by which the consumer reads observation rows, transports, probe bundles, signature relations, provenance, and local naming;*
4. *B is the finite* ProbeBundle *readback row attached to the same observation ledger;*
5. *S is the* SigRel *readback row recording the signature events visible through B;*
6. *P is the* Pkg *provenance over R, H, C, B, S;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverSubjectBoundary}^{\uparrow}}$ *row.*

There is no coordinate for an observer-substance type, subject identity carrier, hidden owner of the observation rows, or host equality principle outside BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* Pkg*, and* NameCert*.*

Lean4 (definition) `BEDC.Derived.ObserverSubjectBoundaryUp.ObserverSubjectBoundaryUp`

Theorem 615.2 (ObserverSubjectBoundary TasteGate carrier alignment)**.** *The displayed rows R, H, C, B, S, P, N of the* ObserverSubjectBoundary$^{\uparrow}$ *carrier are exactly the single-carrier surface checked by the Lean* ChapterTasteGate *theorem. The checked surface consists of* BHist *encode-decode stability, round-trip translation through the observer-subject-boundary event flow, injectivity of that event-flow view, and the empty-history normalization for the same finite observation packet.*

Lean4 ✓ `BEDC.Derived.ObserverSubjectBoundaryUp.ObserverSubjectBoundaryTasteGate single carrier alignment`

Proof. The carrier definition presents one tuple $O = (R, H, C, B, S, P, N)$ and does not add a second subject coordinate. The Lean gate reads the same single carrier through the event-flow encoder and decoder.

Encode-decode stability fixes the BHist row, the round-trip clause returns the displayed ObserverSubjectBoundary packet, and injectivity prevents two distinct displayed packets from sharing one event-flow view. The empty-history normalization identifies the empty observation row as the baseline ledger case for that same carrier. □

Theorem 615.3 (ObserverSubjectBoundary NameCert obligations)**.** *For the carrier of Definition 615.1, the local* NameCert$_{\text{ObserverSubjectBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for O, observation-row stability, componentwise* hsame *transport, displayed* Cont *route replay,* ProbeBundle *and* SigRel *readback exactness,* Pkg *provenance, and local naming exactness.*

Proof. Carrier admission is projection from the finite packet O. The observation rows R, transports H, routes C, bundle row B, signature row S, package row P, and local name N are the displayed coordinates of the carrier.

The stability obligations are componentwise. Transport through H preserves the observation rows and the local naming row, while the routes in C replay only the listed reads.

Bundle and signature exactness come from the readback rows B and S. Since the carrier has no coordinate beyond R, H, C, B, S, P, N, the local naming certificate is exhausted by the stated finite obligations. □

Theorem 615.4 (ObserverSubjectBoundary no extra subject)**.** *Every accepted* ObserverSubjectBoundary$^{\uparrow}$ *packet exposes an observer boundary only through its finite* BHist *observation rows,* hsame *transport,* Cont *routes,* ProbeBundle *readback,* SigRel *readback,* Pkg *provenance, and local* NameCert *row. No additional observer-substance carrier, subject identity row, hidden owner, or host equality coordinate is exported.*

Proof. Project the accepted packet and inspect the coordinates of Definition 615.1. The only rows available to a consumer are R, H, C, B, S, P, N.

Apply the NameCert obligations from Definition 615.3. They identify transport, route replay, bundle readback, signature readback, package provenance, and local naming as the full certificate surface.

A requested subject carrier would require a coordinate outside the displayed tuple. Because no such coordinate is part of the packet, the accepted boundary cannot export observer-substance, subject identity, hidden ownership, or host equality data. □

Theorem 615.5 (ObserverSubjectBoundary ledger exactness)**.** *For an accepted* ObserverSubjectBoundary$^{\uparrow}$ *packet, every observer-facing read is accounted for by the finite ledger route*

$$R, H, C, B, S, P, N.$$

The ProbeBundle *readback and* SigRel *readback are attached to the same* BHist *observation rows, transported by the same* hsame *row, routed by the same* Cont *row, and recorded under the same* Pkg *and* NameCert *provenance.*

Proof. Start with the observation ledger R. The routes in C read the ledger rows and then reach only the displayed bundle, signature, provenance, and local naming rows.

Use componentwise transport through H to keep the observation rows, bundle readback, and signature readback attached to the same finite packet. The package row P records provenance over those transported rows.

Finally apply the no-extra-subject theorem. Since no outside carrier is exported, every observer-facing read is exactly one of the displayed ledger reads and no hidden subject row can refine the ledger. □

Theorem 615.6 (ObserverSubjectBoundary NameCert package)**.** *The carrier, NameCert obligations, no-extra-subject boundary, and ledger exactness assemble the* $\mathsf{NameCert}_{\mathsf{ObserverSubjectBoundary}^{\uparrow}}$ *package as a finite observer-boundary packet over* BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* Pkg*, and local* NameCert *rows. Downstream observer-history consumers receive this package as a finite boundary certificate and not as a separate observer-substance object.*

Proof. Begin with the carrier and obligation theorem. They provide the finite packet and identify the complete local naming surface.

Next use ledger exactness to bind every observer-facing read to the displayed rows R, H, C, B, S, P, N. The readback rows for probes and signatures remain attached to the same observation ledger through the recorded transports and routes.

Finally apply the no-extra-subject theorem. The package delivered to downstream consumers is therefore a finite BEDC boundary certificate, and the absence of an additional subject carrier is part of the package surface rather than an external convention. □

Theorem 615.7 (ObserverSubjectBoundary nontriviality witness)**.** *The empty observation packet and a one-step observation packet are separated by the displayed observation-ledger coordinate* R*. Consequently the* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *carrier cannot collapse to* PUnit*: a consumer can distinguish the packet with no inspected* BHist *row from the packet whose ledger contains one inspected* BHist *row before reading the shared* H, C, B, S, P, N *coordinates.*

Proof. Compare the first coordinate of the two packets. In the empty packet, R contains no observation entry; in the one-step packet, R contains one displayed BHist entry.

Projection to R is part of the carrier surface of Definition 615.1. If the carrier collapsed to PUnit, these two packets would have the same observable ledger projection, contradicting the displayed difference between the empty ledger and the one-entry ledger. □

Theorem 615.8 (ObserverSubjectBoundary consumer scope)**.** *Every downstream consumer of an accepted* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *packet, including the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *surface named in this chapter's introduction, receives exactly the rows* R, H, C, B, S, P, N*. The consumer scope is therefore bound to* BHist *observation rows,* hsame *transport,* Cont *replay routes,* ProbeBundle *readback,* SigRel *readback,* Pkg *provenance, and the local* NameCert *row, without adding public export or bridge closure.*

Proof. Project the accepted packet through Definition 615.1. The only consumer-facing coordinates are R, H, C, B, S, P, N.

The introductory $\mathsf{ObserverHistIdentity}^{\uparrow}$ dependency reads this same packet before it forms its history-identity surface. Its available inputs are therefore the displayed history rows, transport rows, route rows, probe and signature readback rows, provenance, and local naming row.

Apply the no-extra-subject and ledger-exactness theorems. They rule out a hidden subject coordinate and bind every read to the finite observer ledger, so the scoped consumer receives a concrete BEDC boundary packet and not a public-object or bridge-checked surface. □

Theorem 615.9 (ObserverSubjectBoundary public nonescape)**.** *The* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *packet does not export public observer-substance, subject identity, hidden ownership, host equality, or bridge data. Its nonescaping surface is precisely the finite route*

$$R, H, C, B, S, P, N.$$

Thus a consumer may use the boundary as scoped input for observer-history identity, but may not treat it as a public subject carrier or a bridged schema object.

Proof. Start from the consumer-scope theorem. It identifies the complete consumer route as R, H, C, B, S, P, N and ties each row to BHist, hsame, Cont, ProbeBundle, SigRel, Pkg, and NameCert.

Next apply the no-extra-subject theorem. A public subject carrier, hidden owner, or host equality coordinate would need an additional row outside the displayed packet, and the carrier does not contain one.

Finally read the closure boundary. The chapter supplies scoped consumer witnesses for downstream observer-history use, but it does not introduce a public export theorem or a standard bridge row, so public and bridge closure remain unclaimed. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverSubjectBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ packet is generated from finite observation rows, componentwise hsame transport, displayed Cont routes, ProbeBundle readback, SigRel readback, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverSubjectBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 615.1, Definition 615.2 with

Lean4 ✓ `BEDC.Derived.ObserverSubjectBoundaryUp.taste gate`

, Definition 615.3, Definition 615.4, Definition 615.5, Definition 615.6, Definition 615.7, Definition 615.8, and Definition 615.9 bind the finite observer-boundary carrier, TasteGate surface, local package obligations, ledger exactness, nontrivial observation coordinate, scoped consumer surface, and public nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverSubjectBoundary}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ObserverSubjectBoundaryUp.ObserverSubjectBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close observer-substance, subject identity, a hidden owner of observation rows, host equality, public export, or bridge checking.

Upgrade path: No paper theory-axis grade remains; Lean verification and bridge checking remain independent obligations.

Chapter 616

A Concrete Naming Certificate for $\mathsf{ClosedObservationSystem}^{\uparrow}$

The $\mathsf{ClosedObservationSystem}^{\uparrow}$ carrier turns the closed-observation system of Definition 9393.1 into a finite BEDC packet. It records the current observation state, the finite observation-record ledger, the information-conservation order row, the explicit GAP socket, and the local transport, continuation, provenance, and naming rows needed for downstream inspection. The packet names closed observation as a local finite surface; it does not certify a universal future law or import host induction.

The conservation row G and finite ledger R are the closed-observation specialization of chapter 607. This chapter adds the GAP socket and local $\mathsf{NameCert}$ boundary for requests beyond the displayed finite records.

Definition 616.1 (ClosedObservationSystem carrier)**.** *A* $\mathsf{ClosedObservationSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O, R, G, H, C, P, N)$$

with the following displayed rows:

1. *O is the observation-state row naming the current finite state of the system;*
2. *R is the finite observation-record ledger accumulated at O;*
3. *G is the information-conservation order row, recording that later visible reads preserve the records already listed in R;*
4. *H stores the componentwise* hsame *transports for O, R, G and the local naming rows;*
5. *C stores the displayed* Cont *routes by which consumers read the observation state, finite ledger, conservation row, GAP socket, package provenance, and local naming row;*
6. *P is the* Pkg *provenance over the same finite observation-ledger packet;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationSystem}^{\uparrow}}$ *row.*

The GAP socket is the boundary exposed by C when a consumer asks for a universal quantifier, a substrate axiom, or a host induction rule not present in the finite ledger R.

Lean4 ✓ `BEDC.Derived.ClosedObservationSystemUp.ClosedObservationSystemCarrier gap socket package`

Theorem 616.2 (ClosedObservationSystem NameCert obligations)**.** *For the carrier of Definition 616.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationSystem}^{\uparrow}}$ *surface consists of five finite rows: carrier admission for Q; classifier stability for the observation-state row O, record ledger R, and information-conservation row G; GAP-socket exactness for requests beyond the displayed finite records;* Pkg *provenance over O, R, G, H, C; and local naming exactness for N.*

Proof. Carrier admission is projection of the finite tuple Q. The observation state, finite ledger, conservation row, transport, continuation, package, and local naming rows are all visible coordinates of the same BHist packet.

Classifier stability is componentwise. Transport through H preserves the named state O, the finite records in R, and the conservation row G, while C supplies only the displayed read routes.

The GAP-socket row handles the remaining requests. A consumer asking for a universal quantifier, substrate axiom, or host induction rule leaves O, R, G, H, C, P, N, so the local naming row records non-escape rather than an added closure principle. □

Lean4 ✓ `BEDC.Derived.ClosedObservationSystemUp.ClosedObservationSystemNameCertObligations`

Theorem 616.3 (ClosedObservationSystem GAP socket non-escape)**.** *Every accepted* $\mathsf{ClosedObservationSystem}^{\uparrow}$ *packet exports only the finite observation-state row O, finite observation-record ledger R, information-conservation row G, componentwise* hsame *transport H, displayed* Cont *routes C, package provenance P, and local naming row N. A read for universal quantifier closure, substrate axiom import, or host induction reaches the GAP socket and is not exported as an accepted observation record.*

Proof. Read the consumer request through the displayed Cont routes in Definition 616.1. If the request lands in O, R, G, H, C, P, N, the carrier supplies exactly the named finite coordinate.

Apply the NameCert obligations of Definition 616.2. GAP-socket exactness is part of the local surface, so a request outside the finite observation ledger is classified at the boundary rather than added to the ledger.

Thus universal closure, axiom import, and host induction have no exported row in the packet. They remain GAP-facing requests, while the accepted carrier stays a finite observation-ledger surface. □

Lean4 ✓ `BEDC.Derived.ClosedObservationSystemUp.ClosedObservationSystemGapSocketNonescape`

Theorem 616.4 (ClosedObservationSystem primitive scope)**.** *For every accepted* $\mathsf{ClosedObservationSystem}^{\uparrow}$ *carrier*

$$Q = (O, R, G, H, C, P, N),$$

the rows visible to a consumer are scoped by the primitives BHist, hsame, Cont, Pkg, *and* NameCert *together with the observation-conservation sibling of chapter 607. The observation row O, record ledger R, information-conservation row G, transport row H, continuation row C, provenance row P, and local name row N exhaust the primitive interface; no universal quantifier closure, substrate axiom, host induction rule, infinite observation tail, or future-record selector is exported by the packet.*

Proof. Project the carrier of Definition 616.1. Its only data rows are O, R, G, H, C, P, N, and the definition assigns these rows to finite BHist observation data, componentwise hsame transport, displayed Cont reads, Pkg provenance, and the local NameCert row.

The conservation row G is the single nonlocal dependency. It is read through the observation-conservation sibling named in the chapter introduction, so the order information remains an observation-ledger row rather than an ambient temporal law.

Finally apply the GAP socket non-escape theorem, Definition 616.3. Any request outside O, R, G, H, C, P, N reaches the boundary socket and is not reclassified as an accepted observation record. Thus the displayed primitive family is the full scope of the packet. □

Lean4 ✓ `BEDC.Derived.ClosedObservationSystemUp.ClosedObservationSystemPrimitiveScope`

Theorem 616.5 (ClosedObservationSystem bridge-facing handoff)**.** *Every bridge-facing consumer of an accepted* $\mathsf{ClosedObservationSystem}^{\uparrow}$ *carrier*

$$Q = (O, R, G, H, C, P, N)$$

may read only the observation-state row O, the finite observation-record ledger R, the information-conservation row G, the componentwise hsame *transport row H, the displayed* Cont *routes C, the* Pkg *provenance row P, the local* NameCert *row N, and the GAP socket boundary exposed by those continuation routes. The handoff does not export universal quantifier closure, substrate axioms, host induction, infinite observation tails, or future-record selectors.*

Lean4 ✓ `BEDC.Derived.ClosedObservationSystemUp.ClosedObservationSystemBridgeFacingHandoff`

Proof. Project the primitive scope theorem, Definition 616.4. It exhausts the consumer-visible packet by O, R, G, H, C, P, N over the BHist, hsame, Cont, Pkg, and NameCert primitives.

Apply GAP socket non-escape, Definition 616.3. Requests for universal closure, substrate axioms, host induction, infinite tails, or future selectors land at the boundary rather than becoming observation records.

Thus a bridge-facing reader receives exactly the displayed finite rows and the boundary classification of non-exported requests. No additional closure principle or hidden future-record selector is part of the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObservationSystem}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedObservationSystem}^{\uparrow}$ packet is generated from an observation-state row, a finite observation-record ledger, an information-conservation order row, a GAP socket, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedObservationSystem}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 616.4 and Definition 616.5 give the mature finite observation-ledger treatment: the primitive rows, GAP non-escape boundary, and bridge-facing handoff are all read over BHist, hsame, Cont, Pkg, and NameCert primitives.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedObservationSystem}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close universal quantifier closure, substrate axiom import, host induction, infinite observation tails, future-record selection, Lean verification, or bridge checking.

Upgrade path: Checked carrier admission, classifier stability, GAP-socket exactness, package provenance, and local naming upgrade the same public observation-ledger surface on the verification axis.

Chapter 617

A Concrete Naming Certificate for $\mathsf{LorentzFrameRate}^{\uparrow}$

The $\mathsf{LorentzFrameRate}^{\uparrow}$ interface records the finite packet at the boundary where $\mathsf{MaxCausalRate}^{\uparrow}$ becomes an observer-symmetric frame-rate ledger. It does not introduce Lorentz geometry, a metric tensor, synchronized clocks, global frames, or host real-valued speed. It names only the BEDC rows by which the maximum-rate surface can be read uniformly across the displayed histories of a multi-Hist configuration.

The packet consumes the multi-history configuration surface of chapter 474, the cross-Hist causal-rate witness surface of chapter 876, and the max-rate read gate of chapter 933 before it exposes the observer-symmetric rate row.

Definition 617.1 (LorentzFrameRate carrier)**.** *A* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, X, R, S, H, C, P, N)$$

with the following visible coordinates:

1. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite observer-history configuration and its no-global-sync boundary;*
2. *X is the finite family of accepted* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness rows over selected history pairs from M;*
3. *R is an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *row whose witness family is the displayed family X over the same configuration row M;*
4. *S is the observer-symmetry classifier row, recording that each selected observer read of the maximum-rate surface is the same finite unary-bound row of R after componentwise transport;*
5. *H records componentwise* hsame *transport for M, X, R, S, the continuation routes, the package row, and the local naming row;*
6. *C records the visible* Cont *routes from the displayed multi-Hist, causal, maximum-rate, and symmetry rows into the local certificate;*
7. *P is the* Pkg *provenance over* $\mathsf{LorentzFrameRate}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *row.*

The carrier exports a finite observer-symmetric maximum-rate packet. Its classifier preserves the displayed maximum-rate row under accepted observer transport while excluding global frames, synchronized clocks, metric tensors, host real-valued speed, and Lorentz geometry as primitive input.

Lean4 (definition) `BEDC.Derived.LorentzFrameRateUp.LorentzFrameRateUp`

Theorem 617.2 (LorentzFrameRate boundary)**.** *Every accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier presents the next inter-Hist locality boundary as the finite packet*

$$L = (M, X, R, S, H, C, P, N).$$

A consumer may read the multi-Hist configuration row, the selected cross-Hist causal witnesses, the maximum-rate row, the observer-symmetry classifier, componentwise hsame *transport, visible* Cont *routes, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *row. No other frame-rate surface is public through this certificate.*

Proof. Project the accepted carrier. The rows M, X, R, S, H, C, P, N are exactly the admitted coordinates of Definition 617.1.

Read M and X first. These rows keep the boundary tied to the displayed multi-Hist configuration and the selected cross-Hist causal witnesses.

Next read R and S. The maximum-rate row supplies the finite unary-bound surface, while the observer-symmetry classifier records that transported observer reads preserve that same displayed row.

Finally route all consumer access through H, C, P, N. The resulting boundary is exhausted by the listed rows, so no additional frame-rate surface is exported. □

Theorem 617.3 (LorentzFrameRate non-escape)**.** *An accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier cannot export a global frame, synchronized clock, metric tensor, host real-valued speed, or Lorentz geometry as a primitive input. Any consumer request for those objects leaves the finite rows*

$$M, X, R, S, H, C, P, N$$

rather than projecting from the local $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *surface.*

Proof. The carrier inventory contains only the displayed multi-Hist row, cross-Hist witness family, maximum-rate row, symmetry classifier, transport rows, continuation routes, provenance, and local naming row.

Apply the non-escape clauses of the upstream $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and $\mathsf{MaxCausalRate}^{\uparrow}$ certificates. Those rows already exclude global synchronization, unlisted causal witnesses, metric geometry, and ambient real-valued rates from their local surfaces.

The symmetry classifier S adds no coordinate of the excluded kind. It records preservation of the finite maximum-rate row under accepted observer transport, so it cannot become a frame, a clock, a tensor, a host speed, or Lorentz geometry. □

Theorem 617.4 (LorentzFrameRate NameCert obligations)**.** *For the carrier of Definition 617.1, the local* $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *surface consists exactly of carrier admission, configuration inheritance from* $\mathrm{MultiHistConfig}^{\uparrow}$*, causal-witness inheritance from* $\mathsf{CrossHistCausal}^{\uparrow}$*, maximum-rate inheritance from* $\mathsf{MaxCausalRate}^{\uparrow}$*, observer-symmetry preservation of the shared unary-bound row, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and non-escape for global frames, synchronized clocks, metric tensors, host real-valued speed, and Lorentz geometry as primitive input.*

Proof. Carrier admission is projection from Definition 617.1. The finite packet supplies every row named by the local certificate.

The inherited rows are read in dependency order. The configuration row M supplies the displayed histories, the family X supplies the selected cross-Hist causal witnesses over those histories, and the maximum-rate row R supplies the shared unary-bound surface over the same witnesses.

Observer-symmetry preservation is the role of S. It states that the selected observer reads of R remain the same displayed maximum-rate row after componentwise transport.

The rows H, C, P, N provide the remaining stability, locality, provenance, and local naming data. Since no coordinate of the carrier is a global frame, synchronized clock, metric tensor, host real-valued speed, or Lorentz geometry, the non-escape obligation is the displayed inventory boundary. □

Theorem 617.5 (Frame-rate observer-symmetry transport)**.** *Let*

$$L = (M, X, R, S, H, C, P, N)$$

be an accepted $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier. The observer-symmetry row S transports the displayed maximum-rate row R componentwise under the* hsame *data in H: every selected observer read named by S has the same* $\mathsf{MaxCausalRate}^{\uparrow}$ *unary-bound surface as R after applying the corresponding component of H, and the transported read remains routed through the same finite continuation packet C.*

Proof. Project the accepted carrier and read the symmetry coordinate S. By Definition 617.1, S records exactly the observer reads whose maximum-rate surface is compared with the displayed row R.

Next read the transport coordinate H. Its R- and S-components move the maximum-rate row and each observer-symmetry read together, so the comparison is componentwise rather than a global frame comparison.

Finally use the continuation coordinate C. It keeps every transported observer read on the displayed route from M and X to R and then to the local certificate, so the shared unary-bound surface is preserved as a finite $\mathsf{MaxCausalRate}^{\uparrow}$ row. □

Theorem 617.6 (Frame-rate rate-row inheritance)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the maximum-rate row R is inherited only from the same upstream $\mathrm{MultiHistConfig}^{\uparrow}$ *row M and* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness family X displayed by the carrier. No consumer of L may read R as a rate row over a different configuration, an unlisted causal-witness family, or an ambient host geometry.*

Proof. The carrier definition binds R to M and X: R is the accepted $\mathsf{MaxCausalRate}^{\uparrow}$ row whose witness family is the displayed family X over the same configuration row M.

The continuation routes preserve this dependency. A consumer reaches R only through the visible route from the multi-Hist configuration row to the selected cross-Hist causal witnesses and then to the maximum-rate row.

The provenance row P records the same dependency surface. Since neither the carrier nor the provenance row contains a different configuration, an unlisted witness family, or an ambient geometry coordinate, the inheritance of R is limited to the displayed M and X rows. □

Theorem 617.7 (Frame-rate locality non-escape)**.** *The local packet*

$$M, X, R, S, H, C, P, N$$

does not recover a global frame, synchronized clock, metric tensor, host real speed, or Lorentz geometry. These objects are not hidden consequences of observer-symmetry transport, rate-row inheritance, or the local $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *surface.*

Proof. Start with the non-escape theorem for the chapter. It excludes global frames, synchronized clocks, metric tensors, host real-valued speed, and Lorentz geometry as primitive inputs of the carrier.

Observer-symmetry transport adds no excluded coordinate. It moves the displayed maximum-rate row componentwise under hsame and keeps the read on the visible continuation route.

Rate-row inheritance also adds no excluded coordinate. It ties R only to the displayed multi-Hist configuration M and the selected cross-Hist witness family X.

Thus the local certificate remains a finite observer-symmetric maximum-rate packet. Recovering any excluded structure would require data outside M, X, R, S, H, C, P, N. □

Theorem 617.8 (Frame-rate configuration causal inheritance)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the coordinates M, X, R are consumed in dependency order. The multi-Hist row M supplies the displayed observer-history configuration, the family X supplies only $\mathsf{CrossHistCausal}^{\uparrow}$ *witness rows over selected history pairs from M, and the maximum-rate row R supplies only the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row whose witness family is that displayed X over the same M.*

Proof. Project the carrier and read the first three computational coordinates. The definition of the carrier names M before X, because the selected causal witnesses are meaningful only over the displayed multi-Hist configuration.

Read X next. Each row in the family is a $\mathsf{CrossHistCausal}^{\uparrow}$ witness row for selected history pairs from M, so the family cannot select a history outside the configuration already displayed by M.

Finally read R. The maximum-rate row is the accepted $\mathsf{MaxCausalRate}^{\uparrow}$ row over the same configuration and the same selected witness family. Thus consumer access to R factors through M and then X, and no alternate configuration or witness family is available through the local packet. □

Theorem 617.9 (Frame-rate symmetry classifier exactness)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the rows S, H, C compare only transported maximum-rate rows. The observer-symmetry classifier S may identify selected observer reads of the displayed $\mathsf{MaxCausalRate}^{\uparrow}$ *row R after componentwise* hsame *transport in H and visible routing in C; it cannot introduce a global frame, synchronized clock, metric tensor, host real-valued speed, Lorentz geometry, or a rate row not inherited from M, X, R.*

Proof. Start with the role of S. By Definition 617.1, the classifier records equality of selected observer reads only as preservation of the displayed maximum-rate surface of R after componentwise transport.

Now apply the transport and route coordinates. The H row moves the R- and S-components together by hsame, while C keeps the transported read on the visible continuation route through the same local certificate.

The excluded structures are absent from every compared coordinate. Since S, H, C mention only the transported $\mathsf{MaxCausalRate}^{\uparrow}$ row and the local route that reaches it, the classifier has no coordinate from which to manufacture a global frame, synchronized clock, metric tensor, host real-valued speed, Lorentz geometry, or an unrelated rate row. □

Theorem 617.10 (LorentzFrameRate TasteGate obligation package)**.** *The* $\mathsf{LorentzFrameRate}^{\uparrow}$ *chapter presents the obligation surface*

$$M, X, R, S, H, C, P, N$$

as a single finite package: carrier admission is Definition 617.1; inherited-row exactness is given by Definition 617.8; observer-symmetry transport is given by Definition 617.5 and Definition 617.9; provenance is the row P; locality is the row C; non-escape is Definition 617.7; and the existing TasteGate marker attached to Definition 617.1 records that this AI-origin carrier is admitted only at the displayed single-carrier surface.

Proof. Carrier admission is the finite packet definition. It exposes precisely the eight coordinates M, X, R, S, H, C, P, N and no ambient Lorentz-geometric structure.

The inherited-row obligations are discharged by reading the dependency chain in order. The configuration row M feeds the causal witness family X, and those two rows feed the maximum-rate row R.

The observer-symmetry obligations are discharged by the two classifier theorems. They compare selected observer reads only after componentwise hsame transport and visible continuation routing, so the comparison remains a statement about the same finite maximum-rate row.

The remaining obligations are local bookkeeping. The row P records provenance, C records the visible continuation routes, Definition 617.7 supplies non-escape, and the TasteGate marker already attached to the carrier records the AI-origin admission boundary without adding a second marker at this theorem site. □

Theorem 617.11 (LorentzFrameRate upstream row coherence)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the rows M, X, and R form one coherent upstream surface. Every row of the family X is a $\mathsf{CrossHistCausal}^{\uparrow}$ *witness over the displayed* $\mathrm{MultiHistConfig}^{\uparrow}$ *row M, and the row R is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *maximum-rate row over exactly that displayed X-family and the same configuration row M.*

Lean4 ✓ `BEDC.Derived.LorentzFrameRateUp.LorentzFrameRateUpstreamRowCoherence`

Proof. Project the carrier $L = (M, X, R, S, H, C, P, N)$. The first three coordinates are not independent imports: the definition binds X to selected history pairs from M and binds R to the same displayed witness family.

Apply Definition 617.6. It states that a consumer may read R only over the same upstream $\mathrm{MultiHistConfig}^{\uparrow}$ row M and $\mathsf{CrossHistCausal}^{\uparrow}$ witness family X, and not over a different configuration, unlisted witness family, or ambient host geometry.

Thus the upstream surface available through L is the single displayed chain M to X to R. □

Theorem 617.12 (LorentzFrameRate symmetry classifier totality)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

every observer-symmetry read factors through the classifier row S together with the componentwise transport row H and the displayed continuation row C. No observer-symmetry read exports a hidden global frame, synchronized clock, metric tensor, host real-valued speed, Lorentz geometry, or rate row outside the displayed upstream surface.

Proof. Read the classifier coordinate first. By Definition 617.5, S compares selected observer reads of the displayed maximum-rate row R only after componentwise hsame transport in H and along the same finite continuation packet C.

Then apply Definition 617.7. The transported comparison has no route to a global frame, synchronized clock, metric tensor, host real speed, or Lorentz geometry.

Finally combine this with Definition 617.11. Since the rate row being compared is still the displayed R over M and X, the classifier is total only over the local packet surface and not over an ambient observer frame or clock. □

Theorem 617.13 (LorentzFrameRate obligation closure package)**.** *The* $\mathsf{LorentzFrameRate}^{\uparrow}$ *chapter closes the local* $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ *obligation surface for*

$$M, X, R, S, H, C, P, N.$$

Carrier admission is Definition 617.1; upstream row coherence is Definition 617.11; symmetry classifier totality is Definition 617.12; transport locality is Definition 617.5; provenance is the row P; locality is the row C; non-escape is Definition 617.7; and the existing TasteGate marker on the AI-origin carrier records the single-carrier admission boundary.

Proof. Carrier admission is the finite packet definition. It displays exactly the rows M, X, R, S, H, C, P, N and supplies the local $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$ row without adding global geometric input.

The upstream and classifier obligations are now explicit. The upstream coherence theorem binds M, X, and R as one dependency surface, while the classifier totality theorem forces observer-symmetry reads through S, H, C over that same surface.

The remaining rows complete the certificate boundary. Transport locality is supplied by the observer-symmetry transport theorem, provenance is recorded by P, visible continuation by C, and non-escape by the locality theorem. The TasteGate marker already attached to the carrier remains the Lean-side admission marker for this AI-origin single-carrier surface. □

Theorem 617.14 (LorentzFrameRate standard bridge surface)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *bridge-facing packet*

$$L = (M, X, R, S, H, C, P, N),$$

the public bridge surface is exactly the $\mathrm{MultiHistConfig}^{\uparrow}$ *row M, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness family X, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row R, the observer-symmetry classifier S, and the support rows H, C, P, N. The surface exports no metric tensor, synchronized clock, host real-valued speed, global Lorentz frame, or Lorentz geometry as bridge input.*

Proof. Read the carrier definition first. The packet of Definition 617.1 displays only M, X, R, S, H, C, P, N, with M giving the multi-Hist configuration, X giving the selected cross-Hist causal witnesses, and R giving the inherited maximum-rate row.

Next read the classifier and support rows. Theorems 617.5 and 617.12 route observer-symmetry reads through S, H, C, while P and N record package provenance and local naming.

Finally apply Definition 617.7. It excludes metric tensors, synchronized clocks, host real-valued speed, global Lorentz frames, and Lorentz geometry, so the bridge-facing surface is exactly the displayed finite packet. □

Theorem 617.15 (LorentzFrameRate observer-clock refusal)**.** *For an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the rows M, X, R, S, H, C, P, N *expose no synchronized clock and no global observer frame. Every public observer-facing read remains a finite* BHist*-grounded read of the multi-Hist configuration, selected causal witnesses, maximum-rate row, observer-symmetry classifier, componentwise transport, continuation, provenance, and local name.*

Proof. Start from the carrier of Definition 617.1. Its displayed coordinates are the finite BHist rows M, X, R, S, H, C, P, N, with no clock row, frame row, metric tensor row, or host real-valued speed row.

Use Definition 617.11. It binds the source side of the packet to M, X, R, so the maximum-rate read is inherited from the displayed configuration and causal-witness family rather than from an ambient observer frame.

Apply Definition 617.12. The observer-symmetry classifier is routed through S, H, C over the same upstream surface, so an observer-facing read cannot introduce a synchronized clock as classifier input.

Finally invoke Definition 617.7. The non-escape boundary excludes synchronized clocks, global frames, metric tensors, host real-valued speed, and Lorentz geometry; the support rows P, N record only provenance and local naming for the same finite packet. □

Theorem 617.16 (LorentzFrameRate rate-row ledger exhaustion)**.** *Every public rate read of an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier exhausts its rate ledger at the rows* M, X, R *with support from* S, H, C, P, N*. The packet has no remaining coordinate from which a second maximum-rate row, host speed value, metric witness, synchronized clock, or global observer frame can be read.*

Proof. Project the carrier definition Definition 617.1. The only displayed rate row is R, and it is packaged together with the configuration row M, the selected causal-witness family X, and the support rows S, H, C, P, N.

Apply Definition 617.11. This forces any read of R to remain over the same M and X rows, exhausting the upstream part of the ledger at the displayed triple.

Use Definition 617.12 to account for the observer-facing support. The classifier row S, transport row H, and continuation row C only transport and replay the displayed maximum-rate comparison; they do not add a second rate coordinate.

Finally combine the support rows with Definition 617.7. Provenance and local naming record P, N, while non-escape excludes host speed values, metric witnesses, synchronized clocks, and global observer frames. Therefore the public rate ledger is exhausted by M, X, R with exactly the stated support. □

Theorem 617.17 (LorentzFrameRate multi-Hist causal-rate handoff)**.** *Every public rate read of an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier consumes* $\mathrm{MultiHistConfig}^{\uparrow}$*, then* $\mathsf{CrossHistCausal}^{\uparrow}$*, then* $\mathsf{MaxCausalRate}^{\uparrow}$ *in that order. The handoff route is*

$$M \longmapsto X \longmapsto R \longmapsto S, H, C, P, N,$$

and no consumer can read the maximum-rate row before the multi-Hist configuration and cross-Hist causal-witness rows have been consumed.

Proof. Read the carrier of Definition 617.1. It displays the source spine as M, X, R, followed by the observer-symmetry classifier, transport, continuation, provenance, and local naming rows.

Apply Definition 617.11. This theorem makes R dependent on the same multi-Hist configuration M and selected cross-Hist causal witnesses X, so the rate row cannot be consumed before those source rows.

Use Definition 617.12. Every public observer-facing rate read then passes through S, H, C over the already consumed M, X, R surface.

Finally combine the route with Definition 617.7. Since the packet exports no global frame, synchronized clock, metric tensor, host real-valued speed, or Lorentz geometry, there is no alternate handoff path around the displayed $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and $\mathsf{MaxCausalRate}^{\uparrow}$ order. □

Theorem 617.18 (LorentzFrameRate observer-symmetry bridge route)**.** *Every observer-symmetric rate read of an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *packet*

$$L = (M, X, R, S, H, C, P, N)$$

factors through S, H, C and through the inherited upstream route M, X, R before any downstream bridge consumer can cite the packet. Thus a bridge consumer cites only the transported observer-symmetry classifier over the displayed $\mathrm{MultiHistConfig}^{\uparrow}$*,* $\mathsf{CrossHistCausal}^{\uparrow}$*, and* $\mathsf{MaxCausalRate}^{\uparrow}$ *rows.*

Proof. Start with the classifier route. By Definition 617.12, every observer-symmetry read is forced through S, H, C and remains a comparison of transported maximum-rate rows.

Now attach the inherited upstream surface. Theorem 617.11 binds the compared maximum-rate row to M, X, R, so the observer-symmetric read cannot be cited without the multi-Hist configuration and causal-witness family that support R.

Finally combine this route with Definition 617.14. The bridge-facing packet has no public entry point outside M, X, R, S, H, C, P, N, so downstream citation must pass through the classifier route and the inherited upstream route first. □

Theorem 617.19 (LorentzFrameRate root route totality)**.** *For every public observer-symmetry read of an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *carrier*

$$L = (M, X, R, S, H, C, P, N),$$

the read follows the displayed root route

$$M \longmapsto X \longmapsto R \longmapsto S \longmapsto H, C \longmapsto P, N.$$

The route first reads the finite multi-Hist configuration row, then the selected cross-Hist causal witness family, then the maximum-rate row over that family, then the observer-symmetry classifier, then the componentwise hsame *and visible* Cont *rows, and finally the package and local naming rows. No observer-symmetry consumer has a public entry point outside this route.*

Lean4 ✓ `BEDC.Derived.LorentzFrameRateUp.LorentzFrameRateRootRouteTotality`

Proof. Project the carrier of Definition 617.1. Its coordinates are ordered as the root route in the statement: configuration row, selected causal witnesses, maximum-rate row, observer-symmetry classifier, transport and continuation rows, package provenance, and local name.

Use Definition 617.11 to bind M, X, R as one upstream surface. Then use Definition 617.12 to route every observer-symmetry read through S, H, C over that same surface.

Finally apply Definition 617.13. It repacks the upstream coherence, classifier totality, transport, locality, provenance, and local naming rows as the public obligation surface, so the root route is total for public observer-symmetry reads. □

Theorem 617.20 (LorentzFrameRate root non-escape ledger)**.** *The root route of Definition 617.19 exports no global frame, synchronized clock, metric tensor, host real-valued speed, or Lorentz geometry. Each public read remains inside the finite rows M, X, R, S, H, C, P, N and can consume only the displayed observer-history configuration, selected causal witnesses, maximum-rate row, symmetry classifier, transport, route, provenance, and local naming data.*

Proof. Start with Definition 617.7. It excludes the global frame, synchronized clock, metric tensor, host real speed, and Lorentz geometry coordinates from the local packet.

Then use Definition 617.19. Every public observer-symmetry read must pass through the same displayed route, so an excluded object could be exported only if one of the listed coordinates already contained it.

The carrier definition and the non-escape theorem rule out that possibility. The route therefore records a non-escape ledger for downstream root consumers, not a primitive Lorentz-geometric structure. □

Theorem 617.21 (LorentzFrameRate root public package)**.** *The* $\mathsf{LorentzFrameRate}^{\uparrow}$ *public root package is assembled from the carrier of Definition 617.1, upstream coherence Definition 617.11, symmetry classifier totality Definition 617.12, root route totality Definition 617.19, and the non-escape ledger Definition 617.20. It is the finite* BHist*-anchored surface consumed by downstream frame-rate chapters over*

$$M, X, R, S, H, C, P, N.$$

Proof. Carrier admission supplies the finite rows M, X, R, S, H, C, P, N. Upstream coherence binds M, X, R as one displayed source surface, while classifier totality binds every observer-symmetry read to S, H, C over that surface.

Route totality then orders the public read from the configuration row through the witness family, maximum-rate row, symmetry classifier, transport and continuation rows, package provenance, and local name. The non-escape ledger records that the same route exports no global frame, synchronized clock, metric tensor, host real speed, or Lorentz geometry.

Repacking these rows gives the downstream root package. It is public because every consumer-facing read is routed through the displayed carrier rows, and it remains finite because no coordinate outside the carrier participates in the package. □

Theorem 617.22 (LorentzFrameRate mature route completeness)**.** *Every downstream mature read of an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *packet factors through the displayed rows*

$$M, X, R, S, H, C, P, N.$$

The only admissible route is the one assembled from the $\mathrm{MultiHistConfig}^{\uparrow}$ *source row, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness surface, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *maximum-rate row, the observer-symmetry classifier, the*

hsame *transport row, the* Cont *replay row, package provenance, the local* NameCert *row, and the public root package of Definition 617.21. In particular, a mature consumer obtains no global frame, synchronized clock, metric tensor, host real-valued speed, Lorentz geometry, or Lean-side bridge check from this packet.*

Proof. First fix the row spine. The carrier of Definition 617.1 supplies the rows M, X, R, S, H, C, P, N, and Definition 617.11 binds the M, X, R rows as one upstream surface rather than three independent sources.

Next route every observer read. Apply Definition 617.12 to force the observer-symmetry classifier through the displayed configuration, witness, and maximum-rate rows, and apply Definition 617.19 to pass that classifier through the transport, replay, provenance, and local naming rows.

The excluded coordinates are accounted for by Definition 617.20. It rules out global frames, synchronized clocks, metric tensors, host real-valued speed, and Lorentz geometry along the same public route, while Lean-side bridge checking is not a carrier coordinate.

Finally use Definition 617.21. The public package is exactly the finite root surface made from the preceding rows, so any downstream mature read of an accepted packet factors through that surface and through no additional ambient structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$.

Constructive story: A $\mathsf{LorentzFrameRate}^{\uparrow}$ packet is generated from a $\mathrm{MultiHistConfig}^{\uparrow}$ row, finitely many $\mathsf{CrossHistCausal}^{\uparrow}$ witness rows, one $\mathsf{MaxCausalRate}^{\uparrow}$ maximum-rate row over the same configuration, an observer-symmetry classifier, finite hsame and Cont transports, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.LorentzFrameRateUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 617.1, Definition 617.11, Definition 617.12, Definition 617.13, Definition 617.5, Definition 617.7, Definition 617.14, Definition 617.15, Definition 617.16, Definition 617.17, Definition 617.18, Definition 617.21, and Definition 617.22 bind the finite observer-symmetric maximum-rate packet and its mature downstream route over MultiHistConfig, CrossHistCausal, MaxCausalRate, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LorentzFrameRate}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LorentzFrameRateUp.LorentzFrameRateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lorentz geometry, metric tensors, light-cone distance, synchronized clocks, global frames, host real-valued speed, curved spacetime, or Lean-side verification.

Upgrade path: No further paper-side theory closure grade remains for the finite Lorentz frame-rate route; verification and bridge checking remain separate axes.

Chapter 618

A Concrete Naming Certificate for $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$

The $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ interface records the finite mesh-budget route by which a diagonal Real-completion consumer turns a precision request into a bounded family of inspected windows and then into a $\mathsf{Real}^{\uparrow}$ seal read. It refines the selected-window surface of chapter 561 by adding finite mesh budget, dyadic tolerance, and aggregation rows before the $\mathsf{Real}^{\uparrow}$ seal handoff.

It also consumes the finite observation-budget boundary of chapter 611. The precision request, selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and $\mathsf{Real}^{\uparrow}$ seal are the same finite observation surface, with the extra mesh-budget and diagonal aggregation rows inserted before the seal handoff.

Definition 618.1 (Real diagonal mesh-budget carrier)**.** *A* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n)$$

with the following visible coordinates:

1. *q is the rational precision request read through the displayed* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows;*
2. *b is the finite mesh budget, listing the inspected diagonal indices and the dyadic radii assigned to them;*
3. *ω is the* $\mathsf{StreamName}^{\uparrow}$ *window family selected by b;*
4. *ρ is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to those selected windows;*
5. *δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the tolerance split, endpoint rows, radius comparisons, and mesh-bound checks;*
6. *θ is the diagonal aggregation row stating that the finite windows in ω cover the requested diagonal packet under the budget b;*
7. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $q, b, \omega, \rho, \delta, \theta$ have been displayed;*
8. *h records componentwise* hsame *transports for the precision request, mesh budget, selected windows, readback family, dyadic ledger, aggregation row, seal row, provenance, and local name;*
9. *c records the displayed* Cont *routes from q to b, from b to ω, from ω to ρ, from ρ to δ, from δ to θ, and from θ to e;*

10. *p is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *n is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$ *row.*

The carrier contains no infinite mesh, no quotient stream, no host real equality, no ambient completeness theorem, no selected limiting representative, and no open-cover compactness row.

Theorem 618.2 (Real diagonal mesh-budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$ *surface for Definition 618.1 consists exactly of carrier admission for M, precision exposure through q, finite mesh-budget exposure through b, selected* $\mathsf{StreamName}^{\uparrow}$ *window coverage through ω,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through ρ, dyadic tolerance-ledger exactness through δ, diagonal packet aggregation through θ,* $\mathsf{Real}^{\uparrow}$ *seal handoff through e, componentwise* hsame *transport through h, displayed* Cont *route replay through c,* Pkg *provenance through p, and local naming through n.*

Proof. Carrier admission is projection from the displayed packet M. The rows $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$ list the precision request, finite mesh budget, selected windows, regular rational readbacks, dyadic tolerance ledger, aggregation row, real seal, transports, continuations, provenance, and local name.

The mesh-budget obligations are finite. A consumer reads q, then reads the budget b, inspects only the window family ω selected by that budget, and then reads the rational family ρ together with the dyadic ledger δ.

The aggregation row θ records that the displayed windows cover the requested diagonal packet before the seal row e is exported. The rows h, c, p, n account for transport, route replay, provenance, and local naming, so the local surface is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier namecert obligations`

Theorem 618.3 (Real diagonal mesh-budget window coverage)**.** *Every diagonal observation exported from an accepted* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *carrier factors through the finite mesh budget b, the selected* $\mathsf{StreamName}^{\uparrow}$ *window family ω, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family ρ, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger δ, and the aggregation row θ.*

No exported observation may range over an infinite mesh, inspect an unlisted stream window, use quotient equality of completed names, or appeal to ambient Real completeness.

Proof. Project the carrier rows $b, \omega, \rho, \delta, \theta$. The budget b lists the inspected diagonal indices and radii, while ω and ρ provide the selected windows and rational readbacks.

Apply the NameCert obligations of Definition 618.2. They route every observation through the precision request, budget, selected-window family, readback family, dyadic ledger, and aggregation row before any seal-facing read is available.

Since the carrier has no coordinate for an infinite mesh, unlisted stream window, quotient equality, or ambient completeness, a downstream diagonal observation can only use the displayed finite coverage route. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier window coverage`

Theorem 618.4 (Real diagonal mesh-budget monotonicity)**.** *Let*

$$M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n)$$

and

$$M' = (q, b', \omega, \rho, \delta, \theta, e, h, c, p, n)$$

be accepted RealDiagonalMeshBudget$^{\uparrow}$ *packets whose precision request and route rows agree. Suppose the finite budget b' contains every inspected index and radius row of b, and suppose the selected* StreamName$^{\uparrow}$ *windows ω,* RegSeqRat$^{\uparrow}$ *readbacks ρ,* DyadicRatCore$^{\uparrow}$ *ledger rows δ, and aggregation row θ are preserved as the same displayed rows. Then every* Real$^{\uparrow}$ *seal read admitted by the smaller budget b is admitted by the larger displayed budget b'. The enlargement adds no infinite mesh, quotient stream, host real equality, or ambient completeness row.*

Proof. Project the two accepted carriers. The shared rows $q, \omega, \rho, \delta, \theta, e, h, c, p, n$ fix the precision request, selected windows, readbacks, dyadic ledger, aggregation row, seal endpoint, transports, continuation routes, provenance, and local name.

Use the NameCert obligations of Definition 618.2. They make the finite budget row the only place where inspected indices and radii can enter the packet. Since b' contains every inspected row of b, each budget read used by the smaller packet is still visible in the larger packet.

Apply Definition 618.3. The admitted diagonal observation for the smaller budget already factors through the preserved rows $\omega, \rho, \delta, \theta$, so replacing b by b' does not change the visible selected-window route.

Finally replay Definition 618.6 through the shared rows. The same route reaches the same seal row e through $q, b', \omega, \rho, \delta, \theta, h, c, p, n$, and the carrier has no coordinate for an infinite mesh, quotient stream, host real equality, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier budget monotonicity`

Theorem 618.5 (Real diagonal mesh-budget selected-window refinement)**.** *Every accepted* RealDiagonalMeshBudge *packet determines a* RealDiagonalWindowCoverage$^{\uparrow}$ *selected-window surface by projecting the precision row q, mesh budget b, selected* StreamName$^{\uparrow}$ *windows ω,* RegSeqRat$^{\uparrow}$ *readbacks ρ,* DyadicRatCore$^{\uparrow}$ *tolerance ledger δ, and aggregation row θ.*

The projected surface preserves componentwise hsame *transport, displayed* Cont *route replay, and* Pkg *provenance from the original packet. It adds no unlisted window, quotient stream, host real equality, or ambient completeness row.*

Proof. Project the carrier rows named in Definition 618.1. The coordinates $q, b, \omega, \rho, \delta, \theta$ are already displayed before the seal row, and the same packet records the transport, continuation, and provenance rows h, c, p.

Apply the local surface description of Definition 618.2. It exhausts the accepted packet by the precision request, finite mesh budget, selected windows, readback family, dyadic ledger, aggregation row, transport, route replay, provenance, and local naming rows.

Use Definition 618.3 on the projected window data. It identifies the visible selected-window route consumed by RealDiagonalWindowCoverage$^{\uparrow}$ and rules out any read through an unlisted window, quotient stream, host real equality, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier selected window refinement`

Theorem 618.6 (Real diagonal mesh-budget RealUp handoff)**.** *Every* Real$^{\uparrow}$ *completion or continuous-map consumer of an accepted* RealDiagonalMeshBudget$^{\uparrow}$ *carrier receives the seal row e only through the displayed route*

$$q, b, \omega, \rho, \delta, \theta, h, c, p, n.$$

Thus the handoff consumes a rational precision request, a finite mesh budget, selected StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, a* DyadicRatCore$^{\uparrow}$ *tolerance ledger, a diagonal aggregation row,*

componentwise transport, route replay, package provenance, and local naming before the $\mathsf{Real}^{\uparrow}$ *seal is read.*

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier realup handoff`

Proof. Start with the seal coordinate e. It is the only $\mathsf{Real}^{\uparrow}$ row in the carrier and is listed after the precision, mesh-budget, window, readback, dyadic-ledger, and aggregation rows.

Use Definition 618.3. It shows that the diagonal packet presented to a real-facing consumer has already been confined to the finite windows and dyadic tolerances selected by b.

The rows h and c replay the displayed transports and continuations, while p and n provide provenance and local naming. Therefore the seal is reached exactly through the displayed route and not through a hidden completeness or quotient-equality coordinate. □

Theorem 618.7 (Real diagonal mesh-budget seal-route exactness)**.** *For an accepted* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *carrier*

$$M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n),$$

a $\mathsf{Real}^{\uparrow}$ *seal read is admitted precisely when it is routed through* $q, b, \omega, \rho, \delta, \theta$ *with the displayed* h, c, p, n *rows before the row* e *is read.*

Any proposed $\mathsf{Real}^{\uparrow}$ *seal read missing one of the precision request, finite mesh budget, selected-window family, readback family, dyadic ledger, aggregation row, componentwise transport, continuation route, provenance, or local* NameCert *row is outside the carrier.*

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier seal route exactness`

Proof. Project the accepted carrier. The coordinates $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$ are the full packet, and Definition 618.2 identifies them as the complete local NameCert inventory.

For the admitted direction, use Definition 618.3 and Definition 618.5. They confine the selected-window side to the finite route through $b, \omega, \rho, \delta, \theta$ and preserve the displayed transport, continuation, and provenance rows.

Then apply Definition 618.6. It states that a real-facing consumer reaches e only after the displayed route through $q, b, \omega, \rho, \delta, \theta, h, c, p, n$ has been traversed.

For the excluded direction, remove any one of those rows from a proposed seal read. The NameCert inventory has no alternate carrier coordinate for the missing datum, and the selected-window and RealUp handoff results provide no hidden quotient, host equality, ambient completeness, or unlisted window route. Such a read is therefore not a read from the accepted carrier. □

Theorem 618.8 (Real diagonal mesh-budget observation-budget handoff)**.** *Let* $M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n)$ *be an accepted* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *packet, and let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet display the same precision request, selected-window family, dyadic ledger, regular readback rows, and seal rows as a finite observation packet*

$$E, W, D, R, S, H, C, P, N.$$

Then every $\mathsf{Real}^{\uparrow}$ *seal read of* M *factors through that observation budget before the seal row is consumed. The handoff exports no quotient stream, host real equality, ambient completeness, hidden limit selection, or unbounded observation schedule.*

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier observation budget handoff`

Proof. Start with mesh-budget seal-route exactness. It admits a seal read only through the displayed precision request, finite mesh budget, selected windows, readbacks, dyadic ledger, aggregation row, and structural rows before e is read.

Next use mesh-budget window coverage. The selected-window family in M is already finite and is tied to the same request and dyadic ledger that the observation-budget packet records.

Now read the observation-budget side. Request totality supplies the finite observation request, consumer exactness identifies the regular-rational and real-facing reads with that packet, and the seal boundary places every $\mathsf{Real}^{\uparrow}$ seal read after $E, W, D, R, S, H, C, P, N$.

Finally apply observation-budget non-escape. Since the budget packet carries no quotient stream, host equality, ambient completeness, hidden selector, or unbounded schedule coordinate, the mesh-budget seal consumer receives exactly the finite observation-budget handoff. □

Theorem 618.9 (Real diagonal mesh-budget non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *packet factors through the finite rows*

$$q, b, \omega, \rho, \delta, \theta, e, h, c, p, n.$$

The consumer may read the rational precision request, finite mesh budget, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, diagonal aggregation row,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row, and it cannot read infinite-mesh data, quotient streams, host real equality, ambient completeness, selected limits, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier nonescape`

Proof. Project the accepted carrier and the local NameCert surface. The rows $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$ are the complete displayed packet.

Use the window-coverage theorem for observation-facing consumers. It confines every diagonal read to the finite mesh budget, selected windows, regular rational readbacks, dyadic tolerance ledger, and aggregation row.

Use the RealUp handoff theorem for seal-facing consumers. It routes every completion or continuous-map read through $q, b, \omega, \rho, \delta, \theta, h, c, p, n$ before e is consumed.

No remaining carrier coordinate exists beyond those rows. Hence no downstream consumer can extract infinite-mesh data, quotient streams, host real equality, ambient completeness, selected limits, or open-cover compactness from the packet. □

Theorem 618.10 (Real diagonal mesh-budget obligation closure package)**.** *The carrier of Definition 618.1, the NameCert obligations of Definition 618.2, finite window coverage from Definition 618.3, selected-window refinement from Definition 618.5, RealUp handoff from Definition 618.6, seal-route exactness from Definition 618.7, and non-escape from Definition 618.9 constitute the finite mesh-budget obligation package over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package binds a rational precision request to a finite mesh budget, selected StreamName windows, regular rational readbacks, dyadic tolerance rows, diagonal aggregation, the RealUp seal row, componentwise transports, displayed continuations, provenance, and local naming.

Proof. The carrier theorem fixes the packet $M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n)$ and lists the only rows available to a mesh-budget consumer. The NameCert obligation theorem binds admission, precision exposure, finite budget exposure, selected-window coverage, readback, dyadic exactness, aggregation, seal handoff, transport, route replay, provenance, and local naming to that packet.

Apply finite window coverage and selected-window refinement. These results identify the visible route through $b, \omega, \rho, \delta, \theta$ and show that the same rows project to the selected-window surface without adding an unlisted window or a quotient stream.

Apply the RealUp handoff theorem, seal-route exactness, and then non-escape. The seal row e is consumed only after the precision, budget, selected windows, readbacks, dyadic ledger, aggregation row, transports, routes, provenance, and local name have been displayed.

No infinite mesh selection, quotient reals, host real equality, or ambient completeness row exists in the carrier. Therefore the cited rows form the complete finite mesh-budget obligation package and nothing stronger. □

Theorem 618.11 (Real diagonal mesh-budget scoped route)**.** *Every accepted* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *consumer route is scoped by the finite packet rows*

$$q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$$

over $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. A consumer may read the rational precision request, mesh budget, selected windows, rational readbacks, dyadic tolerance ledger, aggregation row, seal row, componentwise transports, displayed continuation routes, package provenance, and local naming row, and nothing outside that finite route.*

Proof. Project the carrier of Definition 618.1. Its displayed coordinates are exactly $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$.

Apply Definition 618.10. The cited package binds the precision request, finite mesh budget, selected windows, rational readbacks, dyadic ledger, aggregation row, seal row, transports, routes, provenance, and local naming into one mesh-budget surface.

The dependency names in p are precisely $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since the carrier has no further row, every accepted consumer route is scoped by that finite packet. □

Theorem 618.12 (Real diagonal mesh-budget consumer scope exhaustion)**.** *The NameCert obligations of Definition 618.2, finite window coverage of Definition 618.3, selected-window refinement of Definition 618.5, RealUp handoff of Definition 618.6, non-escape of Definition 618.9, and obligation-closure package of Definition 618.10 exhaust the scoped mesh-budget surface for* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$*.*

The exhausted surface excludes infinite mesh data, quotient streams, host real equality, ambient completeness, selected limits, and open-cover compactness.

Proof. Start with Definition 618.11. It confines every accepted consumer route to $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$ and to the named rational, stream, regular-sequence, dyadic, diagonal-window, real, history, transport, continuation, package, and naming surfaces.

The NameCert obligations and finite window coverage account for precision, budget, selected-window, readback, dyadic, and aggregation reads. Selected-window refinement accounts for the handoff to $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$.

RealUp handoff accounts for the seal-facing route, non-escape removes unlisted analytic data, and the obligation-closure package repacks those rows as one finite mesh-budget packet. Thus the scoped surface is exhausted and cannot supply infinite mesh data, quotient streams, host real equality, ambient completeness, selected limits, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier consumer scope exhaustion`

Theorem 618.13 (Real diagonal mesh-budget formal target surface)**.** *Let*

$$M = (q, b, \omega, \rho, \delta, \theta, e, h, c, p, n)$$

be an accepted $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *packet. The later finite formal target consumed by Real-completion Lean work is exactly the route through the rational precision request* q*, the finite mesh budget* b*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window family* ω*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family* ρ*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* δ*, the* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *aggregation row* θ*, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal row* e*, componentwise* hsame *transports* h*, displayed* Cont *routes* c*,* Pkg *provenance* p*, and the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$ *row* n*.*

It exports no infinite mesh, quotient stream, host real equality, selected limit, open-cover compactness, or ambient Real completeness.

Proof. Project the accepted packet M. The carrier definition supplies exactly the rows $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$, with no coordinate outside that finite list.

Use Definition 618.2 to identify the local admission, precision, mesh-budget, selected-window, readback, tolerance-ledger, aggregation, seal, transport, route, provenance, and naming obligations. Then use Definition 618.3 to confine every diagonal observation to the selected windows, regular readbacks, dyadic ledger, and aggregation row.

Apply Definition 618.8 for the observation-budget boundary and Definition 618.6 for the seal-facing boundary. These two results place the finite observation packet before the $\mathsf{Real}^{\uparrow}$ seal without introducing quotient streams, host real equality, or ambient completeness.

Finally apply Definition 618.12. Carrier inspection then leaves only the displayed precision request, budget, selected windows, readbacks, tolerance ledger, aggregation, observation handoff, seal, transports, continuations, provenance, and local NameCert row as the formal target surface. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier formal target surface`

Theorem 618.14 (Real diagonal mesh-budget common-refinement seal)**.** *Let an accepted* $\mathsf{RealDiagonalMeshBudget}$ *packet have selected windows and dyadic ledger rows that agree, as displayed routes, with a* $\mathsf{CauchySharedTailModulus}$ *shared-threshold route and a* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *seal-source route. Then every* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal consumer factors through one finite pullback surface whose coordinates are the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, diagonal aggregation row, shared threshold, Real seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

This is a correspondence of displayed finite routes. It is not a claim of host equality of reals, equality of quotient streams, equality of selected limits, ambient completeness, or choice data.

Proof. Start with the local mesh-budget route. By Definition 618.11, every accepted consumer route is confined to $q, b, \omega, \rho, \delta, \theta, e, h, c, p, n$ and to the rational, stream-window, regular-readback, dyadic-ledger, diagonal-window, Real-seal, transport, continuation, package, and naming surfaces. By Definition 618.12, these rows exhaust the mesh-budget surface.

Next read the shared-tail route. Apply Definition 983.3: the two tail requests pass through the same shared threshold before either window or readback is consumed. Apply Definition 983.4: the Real seal is exposed only after the shared threshold, dyadic ledger, paired windows, paired readbacks, transport, continuation, package, and naming rows have been displayed.

Now synchronize the diagonal-limit source. Apply Definition 646.69. It fixes the diagonal source, triangle row, dyadic ledger, window ledger, regular readback, Real seal, hsame transports,

Cont routes, Pkg provenance, and local NameCert row before any seal-facing read can consume the packet.

The hypothesis says that the selected mesh-budget windows and dyadic ledger agree with both adjacent routes as displayed rows. Pull the three route descriptions back along those shared coordinates. The only remaining rows are the selected StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, DyadicRatCore$^{\uparrow}$ ledger, diagonal aggregation, shared threshold, Real seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Coordinate exhaustion gives the conclusion. Since none of the three packets exports host real equality, quotient-stream equality, selected-limit equality, ambient completeness, or choice data, the common refinement is exactly a finite route correspondence and nothing stronger. □

Lean4 ✓ `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier common refinement seal`

Theorem 618.15 (Real diagonal mesh-budget terminal budget refinement)**.** *Let an accepted* RealDiagonalMeshBudget$^{\uparrow}$ *packet have selected mesh windows and dyadic ledger rows that agree with a* CauchySharedTailModulus$^{\uparrow}$ *route and a* DiagonalLimitCompatibility$^{\uparrow}$ *route as displayed rows. Then every terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *seal consumer for those routes is forced through the same finite budget-refinement packet: the selected mesh windows, regular readbacks, dyadic ledger, diagonal aggregation row, shared-tail threshold, Real seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The terminal consumer cannot introduce a selected limit, quotient stream, host real equality, or ambient completeness row.*

Proof. Begin with the local mesh-budget route. Definition 618.11 confines every accepted consumer route to the displayed precision request, finite budget, selected windows, readbacks, dyadic ledger, aggregation, Real seal, transport, continuation, provenance, and local naming rows.

Use consumer-scope exhaustion next. By Definition 618.12, those rows exhaust the mesh-budget surface, so a terminal seal consumer has no mesh-side coordinate beyond the finite packet.

Now attach the downstream budget and common-refinement surfaces. Definition 618.8 places the finite observation-budget handoff before the Real seal, and Definition 618.14 pulls the mesh-budget, shared-tail, and diagonal-limit routes back to one finite refinement surface.

Finally synchronize the two adjacent routes. Definition 983.3 keeps the shared-tail threshold attached to the same windows and readbacks, while Definition 646.69 fixes the diagonal seal source before any seal-facing read is consumed.

The terminal consumer therefore reaches only the finite budget-refinement packet named in the statement. Since the cited routes export no selected limit, quotient stream, host real equality, or ambient completeness coordinate, none can be introduced at the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ seal. □

Theorem 618.16 (Real diagonal mesh-budget terminal route readback)**.** *Let an accepted* RealDiagonalMeshBudget$^{\uparrow}$ *packet have selected mesh windows and dyadic ledger rows that agree, as displayed routes, with a* CauchySharedTailModulus$^{\uparrow}$ *route and a* DiagonalLimitCompatibility$^{\uparrow}$ *route. Then a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *seal read for this triple is admitted if and only if it factors through the finite budget-refinement packet consisting of the selected* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *ledger, diagonal aggregation row, shared-tail threshold,* RealObservationBudget$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Equivalently, the terminal Real$^{\uparrow}$ *and* Completion$^{\uparrow}$ *consumer surface is exactly the finite route displayed by the mesh-budget packet together with the shared-tail and diagonal-limit routes. It*

contains no additional selected limit, quotient stream, host real equality, ambient completeness, or choice coordinate.

Proof. For the forward implication, apply Definition 618.15. Any terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal consumer is forced through the selected mesh windows, regular readbacks, dyadic ledger, diagonal aggregation, shared-tail threshold, Real seal, BHist, hsame, Cont, Pkg, and NameCert rows. Then apply Definition 618.8 to insert the $\mathsf{RealObservationBudget}^{\uparrow}$ handoff in the displayed order before the Real seal.

For the readback implication, begin with a route that factors through the stated finite packet. Use Definition 618.7 to read the $\mathsf{Real}^{\uparrow}$ seal only from the selected windows, regular readbacks, dyadic ledger, diagonal aggregation, transport, continuation, package, and naming rows. Use Definition 618.14 to identify the same finite packet as the common refinement of the mesh-budget, shared-tail, and diagonal-limit routes.

The adjacent route synchronizations keep this readback terminal. By Definition 983.3, the shared-tail threshold is attached to the same windows and readbacks. By Definition 646.69, the diagonal-limit source reaches the seal only through the displayed diagonal aggregation, dyadic ledger, window, readback, hsame, Cont, Pkg, and NameCert rows.

It remains to exclude hidden coordinates. Apply Definition 618.12: the accepted carrier has no consumer row beyond the precision request, mesh budget, selected windows, regular readbacks, dyadic ledger, aggregation, observation handoff, Real seal, transports, continuations, provenance, and local naming row. Thus a route through the finite packet is an admitted terminal read, and the missing selected-limit, quotient-stream, host-equality, ambient-completeness, and choice coordinates cannot be supplied by the terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ packet is generated from a rational precision request, a finite mesh budget, selected stream windows, regular rational readbacks, a dyadic tolerance ledger, a diagonal aggregation row, a real-seal row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 618.1, Definition 618.2, Definition 618.3, Definition 618.5, Definition 618.6, Definition 618.7, Definition 618.9, Definition 618.11, Definition 618.12, Definition 618.13, and Definition 618.14 bind the carrier, NameCert obligations, selected-window coverage, selected-window refinement, RealUp handoff, non-escape, scoped consumer route, formal target surface, common-refinement seal route, and displayed BHist, hsame, Cont, Pkg, and NameCert rows from precision request through selected diagonal windows to the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealDiagonalMeshBudget}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RealDiagonalMeshBudgetUp.RealDiagonalMeshBudgetCarrier\_consumer\_scope\_exhaustion`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, continuous-map theorems, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: No further paper-axis theory upgrade remains; formal verification and bridge checking remain separate over the same finite diagonal mesh packet.

Chapter 619

A Concrete Naming Certificate for $\mathsf{FiniteNetMinimumFold}^{\uparrow}$

The $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ interface packages the finite rational lower-bound fold used by compact-uniform-modulus consumers. It reads a nonempty $\mathsf{ProbeBundle}$ of positive $\mathsf{Rat}^{\uparrow}$ half-radius ledgers, folds them through the $\mathsf{TotalOrder}^{\uparrow}$ finite-minimum route, and returns a single precision history whose classifier-stable lower-bound certificate is visible to the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff.

The fold is the finite rational lower-bound row consumed by chapter 388 and chapter 369. Those consumers read only the displayed $\mathsf{ProbeBundle}$, $\mathsf{Rat}^{\uparrow}$ half-radius, $\mathsf{TotalOrder}^{\uparrow}$ selector, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the handoff exposes no host minimum, open-cover selector, or external modulus coordinate.

Definition 619.1 (FiniteNetMinimumFold carrier)**.** *A* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, R, M, L, H, C, Q, N)$$

with the following visible rows:

1. *P is a nonempty* $\mathsf{ProbeBundle}$ *row indexing the compact finite-net probes;*
2. *R assigns to each bundled probe a positive* $\mathsf{Rat}^{\uparrow}$ *half-radius ledger;*
3. *M is the primitive-recursive accumulator history produced by folding the bundled radius ledgers through the* $\mathsf{TotalOrder}^{\uparrow}$ *finite-minimum selector;*
4. *L stores the lower-bound certificates saying that M is below every radius ledger in R after the displayed classifier transports;*
5. *H records componentwise* hsame *transports for probe membership, radius histories, accumulator rows, and lower-bound rows;*
6. *C records the* Cont *routes for bundle recursion, rational order readback, accumulator comparison, and downstream precision handoff;*
7. *Q records* Pkg *provenance over* $\mathsf{ProbeBundle}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteNetMinimumFold}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the bundle spine, each positive rational half-radius row, the accumulator history, and the lower-bound certificates to a common finite-net surface. The carrier has no coordinate for an external minimum operator, a selected probe representative, host-order trichotomy, or a modulus chosen outside the displayed recursion.

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.TasteGate.FiniteNetMinimumFoldTasteGate single carrier alignment`

Theorem 619.2 (FiniteNetMinimumFold NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteNetMinimumFold}^{\uparrow}}$ *surface for the carrier of Definition 619.1 consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $F = (P, R, M, L, H, C, Q, N)$*;*
2. *classifier transport of bundle membership, positive rational radius rows, accumulator rows, and lower-bound certificates to a common finite-net surface;*
3. *primitive-recursion stability for the fold over the nonempty* ProbeBundle *spine;*
4. *rational lower-bound exactness for every bundled radius ledger stored in* R*;*
5. *ledger rows for rational order readback,* Cont *routes,* Pkg *provenance, and the local naming certificate.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket namecert obligations`

Proof. Project the carrier. The displayed rows contain precisely the nonempty bundle spine, its positive rational half-radius ledgers, the fold accumulator, the lower-bound certificates, transports, continuations, provenance, and local name.

Use the $\mathsf{TotalOrder}^{\uparrow}$ finite-minimum selector at each recursive step over P. The accumulator is initialized from the nonempty bundle data and then compared with the next bundled half-radius ledger, so each recursive branch produces either the previous lower-bound row transported forward or the newly read rational row.

The componentwise transports in H and the continuation routes in C preserve bundle membership, rational positivity, and order readback. Therefore the five obligations exhaust the local certificate surface, and no external selector or host-order principle is available. □

Theorem 619.3 (FiniteNetMinimumFold radius ledger admission)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier*

$$F = (P, R, M, L, H, C, Q, N),$$

the positive $\mathsf{Rat}^{\uparrow}$ *half-radius ledger* R *is admitted only through the nonempty* ProbeBundle *row* P *and the displayed* $\mathsf{TotalOrder}^{\uparrow}$ *fold rows that build* M *and* L*. No radius ledger enters through a host compactness selector, an external enumeration, or a host minimum.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket radius ledger admission`

Proof. Project the carrier and the NameCert obligations. The packet admits radius data as the row R indexed by the nonempty bundle P, and the fold rows M, L are the only rows that compare those radius ledgers.

Apply the lower-bound certificate Definition 619.8. It reads every bundled half-radius ledger through the same finite fold and records the resulting lower-bound evidence in L.

Use nonempty probe exhaustion and selector stability, Definition 619.11 and Definition 619.16. They keep the admitted radius rows inside the displayed bundle recursion and transport surface, so no host compactness selector, external enumeration, or host minimum can introduce another radius ledger. □

Theorem 619.4 (FiniteNetMinimumFold compact-modulus cover input exactness)**.** *Every* CompactModulusCover$^{\uparrow}$ *input consumed through an accepted* FiniteNetMinimumFold$^{\uparrow}$ *packet enters only through*

$$P, R, H, C, Q, N$$

and the fold rows M, L generated from those inputs. The compact-cover consumer has no additional input channel for an unbundled probe, a separate compactness witness, a selected probe representative, or an external modulus row.

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compactmoduluscover input exactness`

Proof. Start with the compact-cover consumer theorem Definition 619.17. It lists the only rows visible to CompactModulusCover$^{\uparrow}$ consumers: P, R, M, L, H, C, Q, N.

Apply radius ledger admission to locate the input radius rows. They enter through P and R before the fold produces M and L.

Finally use nonempty probe exhaustion and order-selector transport. Theorems 619.11 and 619.12 show that the compact-cover input surface is replayed through H, C with provenance Q, N, and no unbundled probe or external modulus row is available. □

Theorem 619.5 (FiniteNetMinimumFold compact-metric probe consumption)**.** *For an accepted* FiniteNetMinimumFold$^{\uparrow}$ *packet supplied by a* CompactMetric$^{\uparrow}$ *finite-net witness, every bundled probe and positive half-radius row consumed by the fold is one of the displayed* ProbeBundle *rows of the compact metric certificate. The accumulator M and lower-bound row L are therefore available to the* UniformModulus$^{\uparrow}$ *handoff without host compactness, selected representatives, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compactmetric probe consumption`

Proof. Project the accepted fold carrier $F = (P, R, M, L, H, C, Q, N)$. The compact metric witness supplies the finite-net bundle through the displayed ProbeBundle row P and the positive half-radius ledgers through R.

Use compact-modulus input exactness, Definition 619.4. It confines the consumed compact-cover input to P, R, H, C, Q, N before the fold rows M, L are generated.

Apply the lower-bound certificate Definition 619.8. The fold produces the accumulator M and certificate L from the displayed bundle and radius rows.

Finally use output exhaustion, Definition 619.7. The UniformModulus$^{\uparrow}$ boundary receives exactly M with lower-bound row L, and the carrier has no channel for host compactness, selected representatives, or open-cover compactness. □

Theorem 619.6 (Finite-net fold compact-uniform modulus consumption)**.** *Every accepted* FiniteNetMinimumFold$^{\uparrow}$ *packet consumed by a* CompactUniformContinuity$^{\uparrow}$ *route supplies exactly the finite* ProbeBundle *centers, positive* Rat$^{\uparrow}$ *radius rows, certified lower-bound fold,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *rows needed to produce a* UniformModulus$^{\uparrow}$ *ledger. The consuming route exports no host finite minimum, selected point oracle, open-cover compactness, quotient real equality, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket root compact cover readback`

Proof. Start with the local carrier and obligation rows, Definition 619.1 and Definition 619.2. They expose the finite bundle P, positive radius ledger R, accumulator M, lower-bound row L, transports H, routes C, provenance Q, and local name N as the whole fold surface.

Apply compact-metric probe consumption and uniform-modulus output exhaustion, Definition 619.5 and Definition 619.7. These identify the centers and radius rows read by the compact-uniform route, then deliver exactly the accumulator M with certificate L to the $\mathsf{UniformModulus}^{\uparrow}$ ledger.

The sibling compact-uniform and uniform-modulus chapters, chapter 388 and chapter 369, consume the same finite handoff, while chapter 37 supplies the rational positivity boundary for the radius rows. Since the fold carrier has no coordinate for host minima, point oracles, open-cover compactness, quotient real equality, or ambient completeness, none of those objects is exported by the route. □

Theorem 619.7 (FiniteNetMinimumFold uniform-modulus output exhaustion)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *output obtained from an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier receives exactly the accumulator precision row M with lower-bound certificate L. The output receives no host minimum, no compactness selector, no selected probe representative, and no external modulus row beyond the displayed finite fold packet.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket uniformmodulus output exhaustion`

Proof. Project the carrier and apply the uniform-modulus handoff Definition 619.10. The handoff delivers the precision history M together with the lower-bound certificate L.

Use modulus consumer exactness Definition 619.13. It identifies M as the exact precision history read by the $\mathsf{UniformModulus}^{\uparrow}$ consumer and L as the only admissibility certificate for bundled pointwise-modulus rows.

Apply the compact-cover consumer theorem to close the output boundary. Because every downstream read factors through P, R, M, L, H, C, Q, N, the output cannot contain a host minimum, compactness selector, selected probe representative, or external modulus row. □

Theorem 619.8 (FiniteNetMinimumFold lower-bound certificate)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier*

$$F = (P, R, M, L, H, C, Q, N),$$

the accumulator history M is certified below every positive $\mathsf{Rat}^{\uparrow}$ *half-radius ledger assigned by R to a bundled probe of P. The certificate is the finite row L, transported through H and read by C; it is not a host-level minimum and does not depend on a selected enumeration outside the* ProbeBundle *recursion.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket lower bound certificate`

Proof. Read the nonempty ProbeBundle row P and the radius assignment row R. Primitive recursion over P produces the accumulator M by repeatedly applying the rational finite-minimum selector to the current accumulator and the next radius ledger.

At the base probe, the lower-bound row is the positive rational radius ledger itself. At each successor probe, the $\mathsf{TotalOrder}^{\uparrow}$ readback compares the current accumulator with the next radius ledger and records the selected lower-bound branch in L.

Transport stability follows from H, and the consumer-facing read is through C. Since the carrier contains no external enumeration, no selected probe representative, and no host-order trichotomy, the only exported lower-bound evidence is the finite certificate row L. □

Theorem 619.9 (FiniteNetMinimumFold positive precision obligation)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier*

$$F = (P, R, M, L, H, C, Q, N),$$

the accumulator M is not only certified below every bundled half-radius ledger by L; it is itself a positive $\mathsf{Rat}^{\uparrow}$ *precision history transported through H and read by C. This positivity and admissibility are part of the displayed finite fold packet, not consequences of a host minimum, host-order trichotomy, or an external rational selector.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket positive precision obligation`

Proof. Use the nonempty ProbeBundle row P. Its base probe supplies an initial bundled radius, and the radius assignment row R records that radius as a positive $\mathsf{Rat}^{\uparrow}$ half-radius ledger.

Run the primitive-recursive fold over the remaining probes. At each recursive step, the $\mathsf{TotalOrder}^{\uparrow}$ selector compares the current accumulator with the next bundled positive $\mathsf{Rat}^{\uparrow}$ row and chooses one of those two already positive rows.

The componentwise transports in H preserve the accumulator as a positive $\mathsf{Rat}^{\uparrow}$ history under classifier movement, and the routes in C replay the same row to downstream consumers. Thus M is an admissible positive precision history as well as a lower-bound certificate.

Because the carrier has no coordinate for a host minimum or host-order trichotomy, the positivity of M is obtained only by this finite row selection and transport. □

Theorem 619.10 (FiniteNetMinimumFold uniform-modulus handoff)**.** *Let a compact-uniform-continuity consumer provide a positive tolerance history ε, a nonempty finite-net* ProbeBundle *P, and positive half-radius ledgers R subordinate to the pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{CFun}^{\uparrow}$ *modulus reads. An accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet*

$$F = (P, R, M, L, H, C, Q, N)$$

hands off the precision history M to the $\mathsf{UniformModulus}^{\uparrow}$ *consumer with the lower-bound certificate L below every bundled half-radius ledger. The handoff exposes the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows and exposes no independent compactness, choice, or host-function-extensionality datum.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket uniform modulus handoff`

Proof. Start with the finite-net data supplied by the compact certificate. The bundle row P and radius row R identify the exact half-radius ledgers needed by the local pointwise-modulus reads.

Apply Definition 619.8. It supplies a single accumulator history M and a finite certificate L proving that M is below every bundled half-radius ledger after the displayed transports.

The $\mathsf{UniformModulus}^{\uparrow}$ consumer then reads M as its precision history and uses L to enter whichever bundled pointwise-modulus row the compact cover supplies for the source point. The package rows Q, N and the routes H, C keep the handoff inside the BEDC ledger surface, so no compactness selector, host function extensionality, or external minimum is exported. □

Theorem 619.11 (FiniteNetMinimumFold nonempty probe exhaustion)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier*

$$F = (P, R, M, L, H, C, Q, N),$$

every probe consumed by the accumulator history M is a member of the nonempty ProbeBundle *row P, and every member of P is consumed exactly once by the primitive-recursive fold that builds M.*

The exhaustion certificate is internal to the displayed bundle recursion, radius ledger R, transport row H, continuation row C, and local NameCert *row N.*

Proof. Start with the nonempty ProbeBundle row P. Its constructors supply the initial probe for the fold and the finite successor spine for the remaining probes.

Run primitive recursion over that spine. At the base probe, the accumulator reads the corresponding positive $\mathsf{Rat}^{\uparrow}$ half-radius ledger from R and records the first lower-bound row in L.

At each successor probe, the recursion extends the accumulator by exactly the next bundled radius ledger. The $\mathsf{TotalOrder}^{\uparrow}$ finite-minimum route compares the current accumulator with that ledger and records the selected branch without introducing an unbundled probe.

The transports in H and routes in C replay the same membership and radius rows under classifier movement. Thus the fold consumes precisely the bundle P, with no missing bundled probe and no probe outside P. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket nonempty probe exhaustion`

Theorem 619.12 (FiniteNetMinimumFold order-selector transport)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier $F = (P, R, M, L, H, C, Q, N)$, the* $\mathsf{TotalOrder}^{\uparrow}$ *selector used at each fold step is stable under the displayed componentwise* hsame *transports. If two transported presentations of the same bundled half-radius ledgers in R are compared through the routes in C, the selected accumulator branch and the lower-bound certificate row in L transport to the same* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *classifier surface.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket order selector transport`

Proof. Read the fold step determined by a bundled probe in P. The step compares the current accumulator row with the next positive $\mathsf{Rat}^{\uparrow}$ half-radius ledger from R.

Transport both compared rows through the corresponding component of H. The rational comparison is then read through the displayed $\mathsf{TotalOrder}^{\uparrow}$ route in C, so the selector sees the same ordered pair on the transported surface.

If the selector keeps the accumulator branch, the transported lower-bound row is the previous certificate transported forward. If the selector chooses the new radius branch, the transported lower-bound row is the radius ledger transported through the same component of H.

In both cases the selected branch and its certificate remain inside M and L. Therefore order selection is stable under the displayed classifier transport and exports no host-order trichotomy. □

Theorem 619.13 (FiniteNetMinimumFold modulus consumer exactness)**.** *Let a* $\mathsf{UniformModulus}^{\uparrow}$ *consumer read an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet*

$$F = (P, R, M, L, H, C, Q, N)$$

through the handoff of Definition 619.10. The precision history delivered to the consumer is exactly M, and the only justification that M is admissible for every bundled pointwise-modulus row is the finite lower-bound certificate L transported by H and replayed by C. No separate compactness witness, selected probe representative, host minimum, or external modulus row is exported.

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket modulus consumer exactness`

Proof. Project the handoff packet. The consumer receives the nonempty probe bundle P, half-radius ledger R, accumulator M, lower-bound certificate L, transports H, routes C, provenance Q, and local name N.

Apply Definition 619.11. It shows that every bundled pointwise-modulus row has been consumed by the same primitive-recursive fold that produced M.

Apply Definition 619.12. It shows that the order selector and lower-bound row survive the displayed classifier transports, so the consumer reads the same accumulator and the same certificate after transport.

Finally use Definition 619.10. The $\mathsf{UniformModulus}^{\uparrow}$ consumer therefore receives exactly M with certificate L, and the carrier has no remaining coordinate from which a separate compactness witness, selected probe, host minimum, or external modulus could be read. □

Theorem 619.14 (FiniteNetMinimumFold terminal readback)**.** *For any terminal consumer read of an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet*

$$F = (P, R, M, L, H, C, Q, N),$$

the delivered fold result is exactly the accumulator precision history M together with the lower-bound certificate L, transported through H and replayed through C with provenance Q and local name N. This terminal readback is equivalent to the displayed finite fold surface P, R, H, C, Q, N producing M, L; it is not a host minimum, an external compactness witness, or a selected representative outside the packet.

Proof. Start with the lower-bound certificate of Definition 619.8. It identifies L as the internal certificate that the accumulator selected by the finite fold is a lower bound for the bundled pointwise-modulus rows.

Use the positive precision obligation of Definition 619.9. This makes the accumulator row M an admissible positive precision history, so the terminal consumer does not need a separate positivity or compactness row.

Next apply the uniform-modulus handoff Definition 619.10. The handoff carries the displayed packet through P, R, M, L, H, C, Q, N, with transports and routes preserving the same fold result.

Finally use modulus consumer exactness Definition 619.13. It says that the consumer receives exactly M and that L is the only admissibility certificate for the bundled rows. Hence the terminal readback is precisely the finite fold surface, with no host minimum or selected representative outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket terminal readback`

Theorem 619.15 (FiniteNetMinimumFold selector totality)**.** *For every accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier*

$$F = (P, R, M, L, H, C, Q, N),$$

every nonempty finite probe ledger P exposes a selected lower-bound row through the displayed fold. The selected row is the accumulator M with lower-bound certificate L, read through the positive radius assignment R and the finite bundle recursion, not through an external selector or host minimum.

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket selector totality`

Proof. Start with the nonempty probe exhaustion theorem Definition 619.11. It shows that the primitive-recursive fold consumes exactly the bundled probes in P.

Use the lower-bound certificate of Definition 619.8. At the base probe, the selected row is the positive radius ledger read from R; at each successor probe, the fold compares the current accumulator with the next radius row and records the selected lower-bound branch in L.

The positive precision obligation Definition 619.9 makes the selected accumulator M an admissible positive $\mathsf{Rat}^{\uparrow}$ precision history. Thus every nonempty probe ledger exposes a selected lower-bound row through the displayed fold alone. □

Theorem 619.16 (FiniteNetMinimumFold selector stability)**.** *For an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *carrier* $F = (P, R, M, L, H, C, Q, N)$*, componentwise* hsame *transport preserves the selected accumulator row* M *and its lower-bound certificate* L*. After transporting bundle membership, positive radius rows, accumulator rows, and lower-bound rows through* H *and replaying them through* C*, the selected row remains the same* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *classifier surface.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket selector stability`

Proof. Project the componentwise transport row H and the continuation routes C. They move the bundle spine, positive rational radius rows, accumulator row, and lower-bound row together.

Apply the order-selector transport theorem Definition 619.12. Each fold step compares the transported accumulator with the transported next radius row and preserves the selected branch and lower-bound certificate.

Use Definition 619.15 to identify the transported selected row with the displayed accumulator M and certificate L. The modulus consumer exactness theorem then confirms that no external selector, host minimum, or separate compactness witness is read after transport. □

Theorem 619.17 (FiniteNetMinimumFold compact-cover consumer)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*, or* $\mathsf{CompactModulusCover}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet can read the selected lower-bound row only through the nonempty finite net* P*, positive precision ledgers* R*, fold accumulator* M*, lower-bound certificate* L*, componentwise transport* H*, continuation routes* C*, package provenance* Q*, and local* NameCert *row* N*. The consumer receives no separate compactness witness, selected probe representative, host minimum, or external modulus row.*

Proof. Begin with the uniform-modulus handoff Definition 619.10. It places compact-uniform-continuity consumers behind the finite-net row P, positive radius row R, accumulator M, and lower-bound certificate L.

Apply nonempty probe exhaustion and selector totality, Definition 619.11 and Definition 619.15. They show that the selected lower-bound row is produced by the displayed fold over the finite net and not by an unrecorded probe representative.

Use selector stability and modulus consumer exactness, Definition 619.16 and Definition 619.13. These transport the same selected row through H, C and deliver exactly M with certificate L to $\mathsf{UniformModulus}^{\uparrow}$ and to compact-cover consumers that depend on that route.

The package row Q and local name N record the provenance of the finite packet. Since the carrier contains no host minimum, separate compactness witness, selected probe representative, or external modulus row, no $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, or $\mathsf{CompactModulusCover}^{\uparrow}$ consumer can read one from this packet. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compact cover consumer`

Theorem 619.18 (FiniteNetMinimumFold compact-cover consumer completeness)**.** *An accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet supplies exactly the fold rows needed by* $\mathsf{CompactModulusCover}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{TotalOrder}^{\uparrow}$ *consumers. The supplied surface is the nonempty finite-net bundle* P*, positive rational half-radius ledger* R*, primitive-recursive accumulator* M*, lower-bound certificate* L*, componentwise transport row* H*, continuation row* C*, package provenance* Q*, and local naming row* N*.*

The packet exports no open-cover compactness theorem, no choice-selected minimizer, no host-level minimum, no selected probe representative, and no external modulus row beyond the displayed finite fold.

Proof. Start with the carrier and local obligations, Definition 619.1 and Definition 619.2. They display the nonempty bundle, rational radius ledgers, accumulator, lower-bound certificates, transports, continuations, provenance, and local naming row.

Apply the lower-bound and positive-precision theorems, Definition 619.8 and Definition 619.9. These identify M as the positive $\mathsf{Rat}^{\uparrow}$ precision row and L as the finite certificate below every bundled radius.

Use nonempty probe exhaustion, order-selector transport, and modulus consumer exactness. Theorems 619.11, 619.12, and 619.13 show that the $\mathsf{TotalOrder}^{\uparrow}$ fold consumes exactly the bundle rows and transports the selected accumulator to downstream consumers.

Finally apply the uniform-modulus handoff and compact-cover consumer theorem. They route $\mathsf{CompactModulusCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers through the same finite rows P, R, M, L, H, C, Q, N. Since the carrier has no coordinate for open-cover compactness, a choice-selected minimizer, a host minimum, a selected probe representative, or an external modulus, none of those objects can be exported. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compact cover consumer completeness`

Theorem 619.19 (FiniteNetMinimumFold UniformModulus CompactMetric handoff gate)**.** *Any* $\mathsf{UniformModulus}^{\uparrow}$ *route that consumes a* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net witness through an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet must pass through the nonempty* $\mathsf{ProbeBundle}$ *row* P*, the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger* R*, the accumulator* M*, the lower-bound certificate* L*, transports* H*, routes* C*, provenance* Q*, and local* $\mathsf{NameCert}$ *row* N*. The route receives no open-cover compactness theorem, no selected-probe representative, and no external compactness selector outside the displayed finite fold packet.*

Proof. First use compact-metric probe consumption. It places every bundled probe and positive half-radius row supplied by the compact metric witness inside P and R, then obtains M and L from the finite fold rather than from open-cover compactness.

Next apply uniform-modulus output exhaustion. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives exactly the accumulator precision row M with lower-bound certificate L, transported and replayed through the displayed rows.

Finally use compact-cover consumer completeness. It assembles the consumer-facing surface as P, R, M, L, H, C, Q, N for $\mathsf{CompactModulusCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{TotalOrder}^{\uparrow}$ consumers. Since that theorem also excludes open-cover compactness, choice-selected minimizers, selected probe representatives, host minima, and external modulus rows, the handoff gate is exactly the finite fold packet. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket uniformmodulus compactmetric handoff gate`

Theorem 619.20 (FiniteNetMinimumFold compact-modulus attainment)**.** *Let a compact-uniform-continuity consumer read an accepted* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net witness through an accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *packet*

$$F = (P, R, M, L, H, C, Q, N).$$

Its public UniformModulus$^{\uparrow}$ *precision is obtained exactly by composing the* CompactMetric$^{\uparrow}$ *finite-net probe rows in P, the* CompactUniformContinuity$^{\uparrow}$ *local pointwise modulus rows indexed by those probes, the positive* Rat$^{\uparrow}$ *half-radius rows in R, the* TotalOrder$^{\uparrow}$ *finite lower-bound selection that produces M, L, and the accepted* FiniteNetMinimumFold$^{\uparrow}$ *transport, replay, provenance, and local naming rows. The consumer receives no host compactness selector, no selected-point oracle, no open-cover compactness theorem, no quotient real equality, and no modulus coordinate outside the displayed finite packet.*

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compact modulus attainment`

Proof. First use compact-metric probe consumption, Definition 619.5. It confines the compact metric input to the displayed finite-net probes P and positive half-radius rows R.

Next apply the uniform-modulus handoff and the handoff gate, Definition 619.10 and Definition 619.19. These place the compact-uniform-continuity consumer behind the same finite packet and deliver the accumulator precision M with lower-bound certificate L to the UniformModulus$^{\uparrow}$ surface.

The finite selection step is controlled by selector totality, selector stability, and the lower-bound certificate, Definition 619.15, Definition 619.16, and Definition 619.8. Together they show that the TotalOrder$^{\uparrow}$ route chooses a lower-bound row from the displayed positive Rat$^{\uparrow}$ ledgers and transports that row through H, C.

Finally use compact-cover consumer completeness, Definition 619.18. It identifies the exact consumer-facing surface for CompactModulusCover$^{\uparrow}$, CompactMetric$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, and TotalOrder$^{\uparrow}$ consumers, so the attained public modulus precision is precisely the finite-net fold result rather than an external compactness or choice artifact. □

Theorem 619.21 (FiniteNetMinimumFold compact-uniform pointwise-radius chain)**.** *A* CompactUniformContinuit *consumer obtains its public* UniformModulus$^{\uparrow}$ *precision from an accepted* FiniteNetMinimumFold$^{\uparrow}$ *packet exactly through the chain of* CompactMetric$^{\uparrow}$ *finite-net probes,* CompactUniformContinuity$^{\uparrow}$ *pointwise modulus rows, positive* Rat$^{\uparrow}$ *radius rows, the* TotalOrder$^{\uparrow}$ *finite lower-bound fold, and the* FiniteNetMinimumFold$^{\uparrow}$ *transport, replay, provenance, and local naming rows. For a packet*

$$F = (P, R, M, L, H, C, Q, N),$$

the delivered precision is the accumulator row M with lower-bound certificate L, transported by H, replayed by C, packaged by Q, and named by N; no compactness selector, selected point oracle, external modulus row, host minimum, or quotient real equality is read outside this chain.

Proof. Start from compact-modulus attainment, Definition 619.20. It already identifies the public modulus precision with the finite-net probes, pointwise modulus rows, positive radius rows, finite lower-bound selection, and the displayed packet rows P, R, M, L, H, C, Q, N.

Use selector totality and selector stability, Definition 619.15 and Definition 619.16. These two rows make M, L the selected lower-bound surface of the finite TotalOrder$^{\uparrow}$ fold and preserve that surface under the displayed hsame transports and Cont replays.

Finally apply compact-cover consumer completeness, Definition 619.18. It says that CompactModulusCover$^{\uparrow}$, CompactMetric$^{\uparrow}$, UniformModulus$^{\uparrow}$, Rat$^{\uparrow}$, and TotalOrder$^{\uparrow}$ consumers see exactly the finite fold packet. Therefore the CompactUniformContinuity$^{\uparrow}$ pointwise-radius chain ends at the public UniformModulus$^{\uparrow}$ precision and has no route to any excluded external selector or equality datum. □

Lean4 ✓ `BEDC.Derived.FiniteNetMinimumFoldUp.FiniteNetMinimumFoldPacket compact uniform pointwise radius chain`

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetMinimumFold}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ packet is generated from a nonempty ProbeBundle, positive $\mathsf{Rat}^{\uparrow}$ half-radius ledgers, a primitive-recursive accumulator, lower-bound certificates, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetMinimumFold}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public surface consists of the ProbeBundle carrier, $\mathsf{Rat}^{\uparrow}$ half-radius ledgers, $\mathsf{TotalOrder}^{\uparrow}$ selector route, BHist histories, hsame transports, Cont routes, Pkg provenance, local NameCert obligations, rational lower-bound certificate, positive precision obligation, uniform-modulus handoff, nonempty probe exhaustion, order-selector transport, modulus consumer exactness, compact-cover consumer export, and Definition 619.18 for the finite-net fold packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetMinimumFold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close the full compact-uniform-continuity theorem, a public compactness bridge, host-level minima, selected probe representatives, or Lean verification.

Upgrade path: No further paper-axis theory step remains for this finite-net fold packet; formal verification and bridge checking remain separate verification-axis work.

Chapter 620

A Concrete Naming Certificate for KernelMorphismUp

The KernelMorphismUp surface turns a compiler graph between two certified kernels into a finite packet whose rows can be inspected without importing a host category of programs. It sits between classifier morphisms and compiler certificates: the source and target rows supply the carrier data, the graph row supplies the displayed edge relation, and the certificate rows record exactly the carrier-preserving and classifier-preserving obligations needed by a later compiler or decompiler chapter.

Definition 620.1 (KernelMorphism carrier)**.** *A* KernelMorphismUp *carrier is a finite* BHist *packet*

$$K = (S, T, G, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* NameCert *carrier row;*
2. *T is the target* NameCert *carrier row;*
3. *G is the finite compiler-graph row relating accepted source histories to accepted target histories;*
4. *E is the edge-admission row recording that every displayed graph edge starts in S and lands in T;*
5. *L is the classifier-lift row recording that every source-classified input has a displayed target edge;*
6. *H records componentwise* hsame *transport for S, T, G, E, L and the local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the source, target, graph, admission, and lift rows;*
8. *P is the* Pkg *provenance row over the same finite morphism packet;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelMorphismUp}}$ *row.*

The local classifier compares two accepted carriers only through the displayed source row, target row, graph row, edge-admission row, classifier-lift row, transport rows, continuation routes, package provenance, and local naming row.

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier source graph edge readback`

Theorem 620.2 (KernelMorphism NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{KernelMorphismUp}}$ *surface for Definition 620.1 has five finite obligations: expose the carrier* $K = (S, T, G, E, L, H, C, P, N)$*; admit only graph edges recorded by* E *as source-to-target carrier edges; lift source classifier membership only through the displayed row* L*; transport graph, edge, and lift comparisons through* H *and* C*; and exclude any export of a host compiler semantics, quotient of programs, ambient category law, or undisplayed target object.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier namecert obligations`

Proof. Carrier exposure is projection from the finite tuple. The source, target, graph, edge-admission, classifier-lift, transport, continuation, package, and local naming rows are all displayed coordinates.

The admission and lift obligations are read from E and L. The edge row prevents a graph edge from leaving the source-target pair, while the lift row records the only route from source classifier membership to a target graph edge.

Transport and non-escape follow from the inventory of the same packet. Since no coordinate carries host compiler semantics, a quotient of programs, an ambient category proof, or an undisplayed target object, the certificate surface remains the finite NameCert morphism packet. □

Theorem 620.3 (KernelMorphism edge-admission obligation)**.** *For every accepted* KernelMorphismUp *carrier*

$$K = (S, T, G, E, L, H, C, P, N),$$

edge admission is exactly the displayed source-to-target row E *of the packet. A graph edge in* G *is accepted only when* E *records its source endpoint in the carrier row* S *and its target endpoint in the carrier row* T*, with the route transported through* H*, replayed through* C*, and named by the local* NameCert *row* N*.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier edge lift transport`

Proof. Project the carrier through Definition 620.1. The graph row G and edge-admission row E are separate displayed coordinates, and E is the only coordinate that records source-to-target admission for graph edges.

Use Definition 620.2. Its admission clause says that only graph edges recorded by E as source-to-target carrier edges are admitted by the local certificate surface.

The transport and route rows H and C preserve the same displayed coordinates, while P and N record provenance and local naming. No host compiler edge or undisplayed target object is available to admit an edge outside E. □

Theorem 620.4 (KernelMorphism classifier-lift obligation)**.** *For every accepted* KernelMorphismUp *carrier*

$$K = (S, T, G, E, L, H, C, P, N),$$

classifier lift is exactly the displayed row L*. A source-classified input may be exported to a target graph edge only by reading* S*, then* L*, then the corresponding graph and admission rows* G *and* E*, with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row attached to the same packet.*

Proof. Start with the carrier definition. The source classifier information is read from S, while the lift route from a classified source input to a target edge is the displayed row L.

Apply Definition 620.2. Its lift clause identifies L as the only route from source classifier membership to a displayed target graph edge.

The rows G and E then certify that the lifted result is a graph edge admitted between S and T. Transport, continuation, package provenance, and local naming keep the lift inside $K = (S, T, G, E, L, H, C, P, N)$ and do not add a compiler-correctness theorem. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier classifier lift obligation`

Theorem 620.5 (KernelMorphism transport-stability obligation)**.** *For every accepted* KernelMorphismUp *carrier*

$$K = (S, T, G, E, L, H, C, P, N),$$

componentwise transport along H and replay along C preserves the same edge-admission and classifier-lift obligations. A transported packet remains readable only through $S, T, G, E, L, H, C, P, N$, so graph edges, admitted endpoints, and lifted classifier inputs stay on the displayed finite morphism surface.

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier transport stability obligation`

Proof. Read the transport rows from the carrier. The row H relates the source, target, graph, edge-admission, classifier-lift, continuation, package, and naming coordinates componentwise, and C replays the same routes after transport.

Use Definition 620.2 for the transported packet. Edge admission still passes through E, and classifier lift still passes through L, because those are the rows named by the certificate surface.

The package and local naming rows P, N close the transported surface as the same finite morphism packet. No ambient compiler semantics, quotient of programs, inverse data, or undisplayed target object is introduced by the transport. □

Theorem 620.6 (KernelMorphism composition surface)**.** *Given accepted* KernelMorphismUp *carriers*

$$K_1 = (S, M, G_1, E_1, L_1, H_1, C_1, P_1, N_1)$$

and

$$K_2 = (M, T, G_2, E_2, L_2, H_2, C_2, P_2, N_2),$$

the displayed composite surface is the finite packet whose source row is S, middle row is read only as the shared target-source row M, target row is T, graph row is the finite relational composite of G_1 and G_2, and obligation rows are obtained by chaining E_1, E_2 and L_1, L_2 through the displayed hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The statement supplies a composition surface only; it does not assert an ambient compiler category or global associativity theorem.*

Proof. Project the two carriers. The first packet exposes a source-to-middle graph and the second exposes a middle-to-target graph, with the same middle row visible as a named coordinate.

Build the composite graph by finite relational composition: a source row is related to a target row exactly when a displayed middle row is named by the two graph rows. The edge-admission rows chain source-to-middle and middle-to-target carrier admission, and the classifier-lift rows chain source classification through the middle row to the target row.

Finally repack the chained obligations through the displayed transport, continuation, package, and naming rows. Because the construction reads only K_1 and K_2, it supplies no ambient associativity law beyond the finite composition surface. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier composition surface`

Theorem 620.7 (KernelMorphism non-escape)**.** *Every consumer read exported by an accepted* KernelMorphismUp *carrier factors through* $S, T, G, E, L, H, C, P, N$*. A read asking for compiler semantics, an optimization proof, decompiler inverse data, quotient program equality, or a target object not named by* T *is recorded as outside the morphism packet and is not exported as NameCert evidence.*

Proof. Read the request through the displayed continuation route C. If it asks for the source row, target row, graph row, edge-admission row, classifier-lift row, transport row, package provenance, or local naming row, the carrier supplies exactly that coordinate.

All other requests leave the carrier inventory. The NameCert obligations allow graph and classifier evidence only through E and L, so a request for host compiler semantics, optimization, decompilation, quotient equality, or an undisplayed target object has no certificate row from which it can be exported. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier non escape`

Theorem 620.8 (KernelMorphism public export)**.** *The public export of an accepted* KernelMorphismUp *carrier is the finite NameCert morphism packet consisting of source carrier row* S*, target carrier row* T*, compiler-graph row* G*, edge-admission row* E*, classifier-lift row* L*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local naming row* N*. Downstream* CertificateCompiler$^{\uparrow}$*, pipeline-stage, and decompiler-right-inverse consumers may cite this packet as their morphism input, but they receive no compiler correctness theorem beyond these rows.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismTasteGate single carrier alignment`

Proof. Start with the carrier definition. It lists exactly the rows named in the public export, and the local classifier restricts comparison to those rows.

Use the NameCert obligation theorem to expose carrier admission, graph edge admission, classifier lift, transport, package provenance, and local naming. The non-escape theorem rules out hidden semantics and inverse data, so the exported object is only the finite morphism packet.

The downstream handoff is therefore a reuse of the same named rows. The first compiler-certificate consumer is chapter 473: it may read the source, target, graph, edge, lift, transport, route, provenance, and local naming rows only as its displayed morphism input. Compiler, pipeline, and decompiler certificates can read the morphism packet as input, but any stronger correctness or inverse claim must be supplied by their own certificate surfaces. □

Theorem 620.9 (KernelMorphism CertificateCompiler handoff)**.** *Any* CertificateCompiler$^{\uparrow}$*, pipeline-stage, or decompiler-right-inverse consumer of an accepted* KernelMorphismUp *carrier must first read the source row* S*, target row* T*, graph row* G*, edge-admission row* E*, classifier-lift row* L*, transports* H*, routes* C*, provenance* P*, and local name* N*. The compiler-facing consumer receives these rows as its morphism input and receives no compiler correctness theorem, inverse proof, quotient program equality, or undisplayed target object.*

Proof. Start with the local NameCert obligations. Definition 620.2 reads carrier admission, graph evidence, classifier accountability, transport, route, provenance, and local naming from the displayed packet.

Next use the edge-admission and classifier-lift obligations. They bind graph edges to the accepted source and target rows, and they expose the classifier lift only through the row L rather than through a host compiler fact.

Finally compose non-escape with public export. Definition 620.7 excludes compiler semantics, inverse data, quotient program equality, and unnamed targets, while Definition 620.8 presents $S, T, G, E, L, H, C, P, N$ as the public morphism packet. Therefore the CertificateCompiler-facing read is exactly that finite input surface and no stronger theorem is transferred. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier certificate compiler handoff`

Theorem 620.10 (KernelMorphism composite compiler handoff boundary)**.** *Given composable accepted* KernelMorphismUp *carriers*

$$K_1 = (S, M, G_1, E_1, L_1, H_1, C_1, P_1, N_1)$$

and

$$K_2 = (M, T, G_2, E_2, L_2, H_2, C_2, P_2, N_2),$$

any CertificateCompiler$^{\uparrow}$*, pipeline-stage, or decompiler-right-inverse consumer of the displayed composite must read the chained source row S, middle row M, target row T, composite graph, edge-admission rows, classifier-lift rows, transports, routes, provenance, and local names before obtaining a compiler-facing morphism input. The consumer receives no ambient compiler correctness theorem, quotient program equality, inverse proof, or category law from the composite packet.*

Proof. Start with the composition surface. Definition 620.6 projects K_1 and K_2, identifies M as the shared middle row, composes G_1 with G_2 through displayed middle witnesses, and chains the admission and classifier-lift rows.

The chained rows are still local NameCert rows. Edge admission is read from E_1, E_2, classifier lift is read from L_1, L_2, and the componentwise transports, routes, provenance, and local names are exactly the displayed H_i, C_i, P_i, N_i rows.

Apply public export and the existing compiler handoff to the repacked composite. Definition 620.8 exposes only the finite morphism packet, and Definition 620.9 makes that packet the input surface for compiler, pipeline, and decompiler-facing consumers.

Finally use non-escape. Definition 620.7 blocks compiler semantics, quotient program equality, inverse data, and unnamed target objects, while the composition theorem itself asserts no ambient category law. Hence the composite handoff supplies the chained finite input and nothing stronger. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier composite compiler handoff boundary`

Theorem 620.11 (KernelMorphism scoped consumer packet)**.** *Every accepted* KernelMorphismUp *consumer route factors through the single packet*

$$K = (S, T, G, E, L, H, C, P, N).$$

Edge admission is read from E, classifier lift is read from L, transport and replay pass through H and C, finite composition passes through displayed middle rows of composable packets, and the non-escape boundary excludes host compiler semantics, quotient program equality, inverse data, and ambient category laws.

Proof. Start with the carrier and public export theorems. They identify the consumer-facing object with the finite rows $S, T, G, E, L, H, C, P, N$, so every accepted read begins inside the same packet.

Use edge-admission and classifier-lift obligations. The row E supplies the only source-to-target admission for graph edges, and L supplies the only route from source classifier membership to a displayed target graph edge.

Transport-stability keeps these rows readable through H and C. For composable routes, the composition-surface theorem chains through the displayed middle row and repacks the composite graph, admission, and lift data into the same finite packet shape.

Finally apply non-escape and public export. They rule out host compiler semantics, quotient program equality, inverse data, and ambient category laws, leaving downstream consumers with exactly the scoped morphism packet. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier scoped consumer packet`

Theorem 620.12 (KernelMorphism source-target scope)**.** *For every accepted* KernelMorphismUp *carrier*

$$K = (S, T, G, E, L, H, C, P, N),$$

the source-target scope is exactly the finite BHist *route that reads S as the source* NameCert *carrier row, T as the target* NameCert *carrier row, G as the displayed graph row, E as the source-to-target admission row, and L as the classifier-lift row, with componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. No source or target object outside the displayed rows is in scope.*

Proof. Project the carrier rows in Definition 620.1. The source row S, target row T, graph row G, edge-admission row E, and classifier-lift row L are explicit coordinates of the same finite BHist packet.

Use the NameCert obligation theorem for the public route. Edge admission is read from E, classifier lift is read from L, and both reads are transported through H and replayed through C.

The package and local naming rows P, N record that the same finite packet is being consumed. Since no carrier coordinate names an external source object, external target object, host compiler semantics, or quotient program equality, the source-target scope is exhausted by $S, T, G, E, L, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier source target scope`

Theorem 620.13 (KernelMorphism composition scope)**.** *Given accepted* KernelMorphismUp *carriers*

$$K_1 = (S, M, G_1, E_1, L_1, H_1, C_1, P_1, N_1)$$

and

$$K_2 = (M, T, G_2, E_2, L_2, H_2, C_2, P_2, N_2),$$

the scoped composite reads only the displayed middle row M, the two graph rows G_1, G_2, the two edge-admission ledgers E_1, E_2, and the two classifier-lift ledgers L_1, L_2, together with the displayed hsame, Cont, Pkg, *and local* NameCert *rows of the two packets. It does not import an ambient category law, hidden associativity proof, external middle object, or compiler semantics.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier composition scope`

Proof. Start from the two displayed carriers. The only shared row available to connect them is the middle row M, occurring as the target row of K_1 and the source row of K_2.

Build the composite read by chaining G_1 with G_2 through that displayed middle row. The edge-admission side chains E_1 and E_2, and the classifier-lift side chains L_1 and L_2.

The transport, continuation, package, and local naming rows are then read componentwise from both packets. Because the construction never leaves those rows, it supplies only a scoped composition surface and no ambient category or compiler-correctness theorem. □

Theorem 620.14 (KernelMorphism NameCert scope)**.** *The scoped* $\mathsf{NameCert}_{\mathsf{KernelMorphismUp}}$ *consumer packet is the finite surface consisting of carrier exposure for* $K = (S, T, G, E, L, H, C, P, N)$*, edge admission through* E*, classifier lift through* L*, transport and replay through* H *and* C*, composition through displayed middle rows and the two edge/lift ledgers of composable packets, and non-escape from host compiler semantics, quotient program equality, inverse data, undisplayed target objects, and ambient category laws.*

Proof. Carrier exposure comes from Definition 620.1. The rows $S, T, G, E, L, H, C, P, N$ are the only rows a consumer can inspect.

Use source-target scope for the local read. It binds source, target, graph, admission, lift, transport, route, package, and naming data into one finite packet.

Use composition scope for composable reads. The only additional data consumed are displayed middle rows and the two edge/lift ledgers belonging to the two displayed packets.

Finally use the non-escape and public export theorems. They exclude host compiler semantics, quotient program equality, inverse data, undisplayed target objects, and ambient category laws, so the scoped consumer receives exactly the NameCert packet described above. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier namecert scope`

Theorem 620.15 (KernelMorphism scoped edge-lift consumer)**.** *Every consumer route through an accepted* KernelMorphismUp *packet*

$$K = (S, T, G, E, L, H, C, P, N)$$

that asks for a target graph edge must first read the source and target rows S, T*, then edge admission through* E*, then classifier lift through* L*. The same read is transported through* H*, replayed through* C*, recorded through* P*, and locally named through* N*; no compiler semantics, inverse data, quotient program equality, or ambient category law can replace those rows.*

Proof. Begin with edge admission. By Definition 620.3, a graph edge is accepted only through the displayed row E after the source endpoint is read from S and the target endpoint is read from T.

Next apply the classifier-lift obligation of Definition 620.4. It identifies L as the route from source classifier membership to the displayed target graph edge, so the consumer cannot read the target edge before the lift row is exposed.

Transport and replay are controlled by Definition 620.5. The rows H and C preserve the same admission and lift data while keeping the read inside the displayed finite packet.

Finally use Definition 620.7 and Definition 620.14. They exclude compiler semantics, inverse data, quotient program equality, and ambient category laws, leaving S, T, E, L, H, C, P, N as the scoped consumer route for target-edge reads. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier scoped edge lift consumer`

Theorem 620.16 (KernelMorphism CertificateCompiler source lock)**.** *Every* $\mathsf{CertificateCompiler}^{\uparrow}$ *consumer of an accepted* KernelMorphismUp *packet*

$$K = (S, T, G, E, L, H, C, P, N)$$

must first read the KernelMorphism source row S*, target row* T*, graph row* G*, edge-admission row* E*, classifier-lift row* L*, transport row* H*, route row* C*, provenance row* P*, and local name row* N*. The compiler may then add target landing and compiler-stage rows of its own, but it cannot replace the morphism packet by host compiler semantics, quotient program equality, inverse data, or an undisplayed target object.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier certificate compiler source lock`

Proof. Start with the public export theorem Definition 620.8. It identifies the downstream morphism input with the finite packet rows $S, T, G, E, L, H, C, P, N$ and grants no compiler-correctness theorem beyond those rows.

Next apply the NameCert scope theorem Definition 620.14 and the scoped edge-lift consumer theorem Definition 620.15. Together they force target-edge reads to expose source and target rows, edge admission, classifier lift, transport, route replay, provenance, and local naming before the read can leave the morphism packet.

Finally use the CertificateCompiler$^{\uparrow}$ endpoint route Definition 473.55. It lets the compiler add its own target landing and endpoint rows after the morphism input has been read. Since the KernelMorphism non-escape boundary still excludes host compiler semantics, quotient program equality, inverse data, and undisplayed target objects, the compiler-facing route remains source-locked to $S, T, G, E, L, H, C, P, N$. □

Theorem 620.17 (KernelMorphism standard reflection bridge)**.** *The edge-admission, classifier-lift, transport-stability, composition-surface, public-export, source-lock, and* NameCert*-scope rows assemble a standard reflection bridge for* KernelMorphismUp*. The bridge reads only the finite packet rows $S, T, G, E, L, H, C, P, N$ and composable middle rows, and imports no host equality, host function space, compiler semantics, inverse data, or ambient category law.*

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier standard reflection bridge`

Proof. Start with edge admission and classifier lift, Definition 620.3 and Definition 620.4. These rows expose the source-to-target graph route and the classifier read without leaving the finite carrier.

Transport stability and composition surface, Definition 620.5 and Definition 620.6, supply the bridge replay and the composable middle-row surface.

Finally apply the public export, source-lock, and NameCert scope theorems, Definition 620.8, Definition 620.16, and Definition 620.14. They package the displayed rows as the bridge-facing public object and exclude host equality, host functions, compiler semantics, inverse data, and ambient category laws. □

Theorem 620.18 (KernelMorphism CertificateCompiler route non-escape)**.** *Every* CertificateCompiler$^{\uparrow}$*-facing consumer of an accepted* KernelMorphismUp *packet*

$$K = (S, T, G, E, L, H, C, P, N)$$

factors through the displayed source row S, target row T, graph row G, edge-lift and edge-admission row E, classifier-lift row L, componentwise hsame *transport row H, displayed* Cont *route row C,* Pkg *provenance row P, and local* NameCert *row N. The consumer may use the CertificateCompiler route only after those KernelMorphism rows have been read.*

In particular, the handoff cannot introduce a second compiler endpoint, host equality of kernels, hidden decompiler inverse, off-ledger reflection tuple, quotient program equality, or ambient category law.

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier certificate compiler route nonescape`

Proof. Start with the CertificateCompiler handoff theorem, Definition 620.9. It exposes the compiler-facing route only through the displayed KernelMorphism packet rows $S, T, G, E, L, H, C, P, N$.

Next apply the source lock, Definition 620.16. It forces the compiler-facing consumer to read the source, target, graph, edge-admission, classifier-lift, transport, route, provenance, and local naming rows before the compiler endpoint can add any target landing data of its own.

The standard reflection bridge of Definition 620.17 repacks those same rows as the bridge-facing public object and imports no host equality, host function space, compiler semantics, inverse data, or ambient category law.

Finally use the sibling CertificateCompiler route scope, Definition 473.60. It permits the morphism leg to contribute only the graph-admission and classifier-lift rows licensed by KernelMorphismUp. Therefore no second compiler endpoint, hidden decompiler inverse, off-ledger reflection tuple, quotient program equality, or host equality of kernels has a route into the accepted handoff. □

Theorem 620.19 (KernelMorphism composition standard-reflection compatibility)**.** *Given composable accepted* KernelMorphismUp *packets*

$$K_1 = (S, M, G_1, E_1, L_1, H_1, C_1, P_1, N_1)$$

and

$$K_2 = (M, T, G_2, E_2, L_2, H_2, C_2, P_2, N_2),$$

the standard reflection bridge of the displayed composite factors through the two standard reflection bridge routes of K_1 and K_2 together with the existing composition scope rows. The composite bridge reads only the shared middle row M, the graph rows, edge-admission rows, classifier-lift rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance rows, and local* NameCert *rows.*

It cannot introduce host equality of kernels, decompiler inverse data, quotient program equality, off-ledger compiler semantics, or ambient category laws.

Proof. Start with composition scope, Definition 620.13. It makes the composite read pass through the shared middle row M, the two graph rows, the two edge-admission ledgers, the two classifier-lift ledgers, and the displayed transport, route, provenance, and local naming rows.

Next apply the standard reflection bridge theorem, Definition 620.17, separately to the two accepted packets. Each bridge route exposes only the finite packet rows and composable middle rows, with no host equality, host function space, compiler semantics, inverse data, or ambient category law.

Now use the composite compiler handoff boundary, Definition 620.10. It repacks the chained rows as the compiler-facing morphism input of the composite without importing compiler correctness, quotient program equality, inverse proof, or category law data.

Finally apply route non-escape, Definition 620.18. Since every compiler-facing consumer has already factored through the displayed morphism rows, the standard reflection bridge of the composite has no coordinate from which to obtain host kernel equality, decompiler inverse data, quotient program equality, or ambient category laws. □

Lean4 ✓ `BEDC.Derived.KernelMorphismUp.KernelMorphismCarrier composition standard reflection compatibility`

Closure status for $\mathsf{NameCert}_{\mathsf{KernelMorphismUp}}$.

Constructive story: A KernelMorphismUp packet is generated from source and target NameCert rows, a finite compiler graph, edge-admission and classifier-lift rows, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KernelMorphismUp}}$, maturePackageClosed)

Scope closed: Definition 620.1, Definition 620.2, Definition 620.6, Definition 620.7, Definition 620.8, Definition 620.13, Definition 620.14, Definition 620.15, Definition 620.16, Definition 620.17, and Definition 620.18 define

the finite morphism-packet surface, its public export, compiler-facing source lock, route non-escape, and the bridge-facing consumer surface.

Formal verification: FormalStatus(NameCert$_{\text{KernelMorphismUp}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close compiler correctness, optimization correctness, decompiler right inverses, ambient category laws, Lean verification, or bridge checking.

Upgrade path: The theory axis is complete at the mature finite morphism-packet surface; remaining work is Lean verification and bridge checking.

Chapter 621

A Concrete Naming Certificate for BinderContextSubstitutionSeal$^{\uparrow}$

The BinderContextSubstitutionSeal$^{\uparrow}$ interface packages the closed-boundary substitution site used by the MetaCIC self-compilation account. It records a term, a binder depth, a substitution payload, the attempted substitution result, and the boundary certificate proving that the payload has no visible target inside the classified term.

Theorem 621.1 (BinderContextSubstitutionSeal closed-term boundary dependency)**.** *Every accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet reads its* `ClosedAt` *boundary and inertness row as a refinement of the sibling* ClosedTermSubstitutionBoundary$^{\uparrow}$ *surface of chapter 527 before exporting the binder-depth payload route to compiler or evaluator consumers.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier closed term boundary dependency`

Proof. Project the packet rows as

$$S = (t, d, v, s, b, H, C, P, N).$$

The row b is the displayed closed-boundary witness: it carries `ClosedAt` d t together with the inertness equality read by the substitution seal.

The sibling surface of chapter 527 supplies exactly the closed-term boundary that makes a substitution read inert at a displayed depth. The seal adds the payload row v, the binder-depth row d, replay through $s = \texttt{substitute}(d, v, t)$, and local transport and provenance rows, but it does not replace that closed-term boundary with an independent open-substitution semantics.

Apply Definition 621.4 and Definition 621.5. They show that consumers receive the payload route only through the equality certified by b and only through the finite packet rows exported by the handoff.

Hence the closed-term substitution boundary is a real dependency of the binder-context seal. The seal refines that sibling boundary by adding depth, payload, replay, transport, and naming rows while preserving the same closed-boundary source for compiler and evaluator reads. □

Definition 621.2 (BinderContextSubstitutionSeal carrier)**.** *A* BinderContextSubstitutionSeal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (t, d, v, s, b, H, C, P, N)$$

with the following visible rows:

1. *t is a* `Term` *row;*
2. *d is the displayed binder-depth row;*
3. *v is the substitution payload row;*
4. *s is the result row with readback $s = \texttt{substitute}(d, v, t)$;*
5. *b is the boundary witness carrying* `ClosedAt` *d t and the inertness equality $s = t$;*
6. *H records componentwise* hsame *transports for the term, depth, payload, result, boundary witness, provenance, and local naming rows;*
7. *C records the* Cont *routes for term readback, closedness readback, substitution replay, inertness readback, and certificate exposure;*
8. *P records* Pkg *provenance over* `Term`*,* `ClosedAt`*,* `substitute`*,* `shift`*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BinderContextSubstitutionSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed `Term` *row, the* `ClosedAt` *boundary witness, the substitution replay row, the inertness equality, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and the local naming row. It has no coordinate for an open-term semantics, a capture-avoiding metatheory outside* `Term`*, an ambient binder context beyond d, or a payload-use route absent from b.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealTasteGate`
`single carrier alignment`

Theorem 621.3 (BinderContextSubstitutionSeal NameCert obligations)**.** *For the carrier of Definition 621.2, the local* $\mathsf{NameCert}_{\mathsf{BinderContextSubstitutionSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (t, d, v, s, b, H, C, P, N)$,* `Term` *readback,* `ClosedAt` *boundary readback, replay of $s = \texttt{substitute}(d, v, t)$, inertness readback $s = t$, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier`
`namecert obligations`

Proof. Project the carrier. The rows t, d, v, s expose the syntactic substitution attempt, while b exposes the closed-boundary witness and the equality read from that witness.

The boundary part of b is supplied by the MetaCIC witnesses `substitute_closed` and `closed_under_substitu`
The matching binder-motion budget is the `shift_closed` witness used by the surrounding self-compilation account, so the package names the same boundary where shifting and substitution are both inert on the classified slice.

The rows H, C, P, N then expose transport, continuation replay, package provenance, and local naming for exactly the same packet. Since the carrier has no further coordinate, these rows exhaust the local NameCert surface. □

Theorem 621.4 (BinderContextSubstitutionSeal non-escape boundary)**.** *Every accepted* BinderContextSubstitutio*n packet exports the substitution payload v only as an attempted route through $s = \texttt{substitute}(d, v, t)$ together with the boundary witness b. If b carries* `ClosedAt` *d t, then the exposed result row is $s = t$, and no consumer can read the packet as a payload insertion, an open-variable escape, a binder capture, or a depth borrowed from outside the displayed boundary.*

Lean4 ✓ BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier non escape boundary

Proof. Read the substitution row from the carrier. The only route from v to the accepted output is the displayed replay $s = \texttt{substitute}(d, v, t)$.

Use `substitute_closed` on the `ClosedAt` d t component of b. It identifies the replayed substitution result with the original term, so the payload is carried through the traversal but is never observed at a target variable inside t.

The classifier row `closed_under_substitute_classifier` records the same fact as a generator-closure boundary, and `shift_closed` supplies the matching binder-budget law for moving the same closed term through the surrounding context. The visible routes therefore expose no open-term escape, capture route, or extra depth coordinate. □

Theorem 621.5 (BinderContextSubstitutionSeal compiler handoff)**.** *An accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet*

$$S = (t, d, v, s, b, H, C, P, N)$$

hands the closed-term substitution site to a compiler or evaluator consumer as a finite boundary certificate over `Term`*,* `ClosedAt`*,* `substitute`*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The consumer receives the inert result row $s = t$ and the same package provenance; it receives no semantic evaluator theorem, no global subject-reduction claim, and no unrecorded binder context.*

Proof. Start with the carrier rows. The compiler-facing data are the term t, depth d, payload v, replayed result s, boundary witness b, and the transport and route rows H, C.

Apply Definition 621.4. The handoff can use the substitution site only through the equality $s = t$ certified by `substitute_closed` and exposed through the local boundary witness.

The package rows P, N bind that equality to the same `Term` and `ClosedAt` generator surface named by the self-compilation vision. Thus the consumer receives a finite substitution-seal packet, not a global evaluator correctness theorem or a hidden binder-context resource. □

Lean4 ✓ BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier compiler handoff

Theorem 621.6 (BinderContextSubstitutionSeal obligation rows)**.** *For an accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet*

$$S = (t, d, v, s, b, H, C, P, N),$$

the obligation ledger consists of the following separate rows: carrier admission for S, `Term` *readback for t,* `ClosedAt` *boundary readback for b, replay of $s = \texttt{substitute}(d, v, t)$, inertness equality $s = t$, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local* NameCert *exposure through N.*

Proof. Use the carrier of Definition 621.2. Its coordinates list exactly the term, depth, payload, result, boundary witness, transport, route, provenance, and local naming rows.

Definition 621.4 identifies the boundary row as the only place where the payload route can be read. Definition 621.5 then exposes the same rows to compiler and evaluator consumers.

Separating those coordinates gives the stated ledger rows. No additional obligation row can be projected because the carrier has no hidden binder-context coordinate. □

Lean4 ✓ BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier obligation rows

Theorem 621.7 (BinderContextSubstitutionSeal closed-boundary exactness)**.** *In every accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet* $S = (t, d, v, s, b, H, C, P, N)$*, the* `ClosedAt` *boundary witness in* b *forces the substitution result row to remain the original term row: the only exported result is* $s = t$*. The packet exports no payload insertion, no open-variable occurrence, and no binder capture route.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier closed boundary exactness`

Proof. Read the boundary witness through Definition 621.2. The witness b carries both `ClosedAt` d t and the inertness equality for the replayed substitution row.

Apply Definition 621.4. It states that the payload v is visible only through the attempted route $s = \texttt{substitute}(d, v, t)$ and that the accepted result row is $s = t$.

The compiler handoff of Definition 621.5 preserves this same equality for consumers. Hence the closed boundary is exact: the result row is the original term row, with no exported payload insertion route. □

Theorem 621.8 (BinderContextSubstitutionSeal route determinacy)**.** *Every compiler or evaluator handoff from* BinderContextSubstitutionSeal$^{\uparrow}$ *factors through the displayed packet*

$$S = (t, d, v, s, b, H, C, P, N).$$

The route is determined by the term row, depth row, payload row, substitution replay row, boundary witness, componentwise hsame *transports, displayed* Cont *routes, provenance, and local naming; it has no hidden binder context and no consumer-specific substitution channel outside* S*.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier route determinacy`

Proof. Start from the compiler handoff theorem Definition 621.5. It already states that compiler and evaluator consumers receive the finite boundary certificate over the rows of S.

Use the carrier definition to identify those rows as the complete route data. Transport is exhausted by H, replay is exhausted by C, and provenance and naming are exhausted by P, N.

Finally apply Definition 621.4. Since payload use is blocked at the displayed closed boundary, no hidden binder context or outside substitution channel can be read from an accepted handoff. □

Theorem 621.9 (BinderContextSubstitutionSeal obligation consumer)**.** *Every downstream compiler or evaluator consumer of an accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet*

$$S = (t, d, v, s, b, H, C, P, N)$$

factors first through the obligation rows of Definition 621.6 and then through the route determinacy of Definition 621.8. Such a consumer may use the displayed `Term`*,* `ClosedAt`*,* `substitute` *replay, inertness,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, but it may not use open-term semantics, global evaluator correctness, subject reduction, or a hidden binder context.*

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier obligation consumer`

Proof. Project the carrier through Definition 621.2. The rows visible to the consumer are exactly $t, d, v, s, b, H, C, P, N$.

Apply the obligation rows of Definition 621.6. They separate carrier admission, term readback, closed-boundary readback, substitution replay, inertness, transport, continuation, provenance, and local naming.

Then apply route determinacy. The consumer route is fixed by the same rows and has no consumer-specific channel outside the packet.

Finally use the non-escape boundary of Definition 621.4. Since the payload is blocked at the displayed closed boundary, the consumer cannot recover open-term semantics, subject reduction, global evaluator correctness, or an unrecorded binder context. □

Theorem 621.10 (BinderContextSubstitutionSeal scope triad)**.** *Every accepted* BinderContextSubstitutionSeal$^{\uparrow}$ *packet is scoped by exactly three primitive faces: the syntactic face consisting of the displayed* `Term`*,* `ClosedAt`*, and* `substitute` *rows; the replay face consisting of the displayed* BHist*,* hsame*, and* Cont *rows; and the provenance face consisting of the displayed* Pkg *and local* NameCert *rows. The packet exports no fourth primitive face for open-term semantics, global evaluator correctness, subject reduction, or a hidden binder context.*

Proof. Project the carrier in Definition 621.2. Its coordinates are exactly

$$S = (t, d, v, s, b, H, C, P, N).$$

The rows t, d, v, s, b supply the syntactic `Term`, `ClosedAt`, and `substitute` face; the rows H, C supply replay through BHist, hsame, and Cont; and the rows P, N supply Pkg provenance and the local NameCert face.

Apply Definition 621.6. It separates the same carrier admission, readback, replay, inertness, transport, continuation, provenance, and naming rows into the obligation ledger.

Next apply Definition 621.8. Every compiler or evaluator handoff is determined by those same rows, so no consumer-specific substitution channel or hidden binder context can be added after the packet is accepted.

Finally use Definition 621.9. It allows downstream consumers to read only the displayed syntactic, replay, and provenance rows, and explicitly excludes open-term semantics, global evaluator correctness, subject reduction, and unrecorded binder context. The three faces therefore exhaust the scoped primitive surface. □

Lean4 ✓ `BEDC.Derived.BinderContextSubstitutionSealUp.BinderContextSubstitutionSealCarrier scope triad`

Theorem 621.11 (BinderContextSubstitutionSeal bridged export)**.** *For every accepted* BinderContextSubstitutionS *packet*

$$S = (t, d, v, s, b, H, C, P, N),$$

the compiler handoff and route-determinacy rows form the bridge-facing export surface for downstream evaluator and compiler consumers. Such a consumer may read the displayed `Term`*,* `ClosedAt`*,* `substitute`*, inertness,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it receives no open-term semantics, global evaluator correctness, subject reduction, or hidden binder context outside* S*.*

Proof. Project the carrier of Definition 621.2. Its rows are exactly $t, d, v, s, b, H, C, P, N$, with no separate consumer channel.

Apply the compiler handoff theorem, Definition 621.5. It exports the closed-term substitution site to compiler and evaluator consumers as a finite boundary certificate over the displayed syntactic, replay, provenance, and local naming rows.

Use route determinacy, Definition 621.8, together with the scope triad of Definition 621.10. These fix every consumer route through the same packet and exclude open-term semantics, global evaluator correctness, subject reduction, and hidden binder context from the bridge-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinderContextSubstitutionSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BinderContextSubstitutionSeal}^{\uparrow}$ packet is generated from a `Term` row, a binder-depth row, a substitution payload, a replayed `substitute` result, a `ClosedAt` inertness witness, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BinderContextSubstitutionSeal}^{\uparrow}}, \mathsf{bridgedClosed})$

Scope closed: Definition 621.2, Definition 621.3, Definition 621.6, Definition 621.7, Definition 621.8, Definition 621.9, Definition 621.10, and Definition 621.11 bind the bridge-facing primitive surface over `Term`, `ClosedAt`, `substitute`, inertness, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BinderContextSubstitutionSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close semantic substitution correctness for open terms, global evaluator correctness, subject reduction, a public compiler bridge, or Lean verification.

Upgrade path: The next paper step is a bridge theorem connecting the closed-boundary substitution packet to the compiler interface.

Chapter 622

A Concrete Naming Certificate for $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$

The $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ packet gives the strict axiom-purity boundary a finite local certificate. It records a theorem row, its proof-ancestry report, the forbidden host-axiom dependency row, the refusal verdict, displayed audit routes, package provenance, and the local naming row. The packet is the concrete object consumed when a theorem has no source axiom under `lean4/BEDC/` but its host proof ancestry still attempts to carry `Classical.choice`, `Quot.sound`, or `propext`.

The proof-ancestry report row is consumed through the axiom-dependency tuple surface of chapter 861: that sibling supplies the finite dependency-budget packet, while $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ specializes the packet to the three forbidden host rows and the refusal verdict.

Definition 622.1 (ForbiddenAxiomAncestry carrier)**.** *A* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, A, F, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the theorem row, naming the public BEDC theorem or lemma whose proof ancestry is being inspected;*
2. *A is the proof-ancestry report returned by the host audit query for T;*
3. *F is the forbidden dependency row for `Classical.choice`, `Quot.sound`, and `propext`;*
4. *R is the refusal verdict, recording whether the ancestry report is admitted as clean or refused because it meets one of the forbidden dependency rows;*
5. *H records displayed* hsame *transport for theorem names, ancestry reports, forbidden rows, verdicts, continuation routes, provenance, and the local naming row;*
6. *C records the displayed* Cont *audit routes by which a consumer reads T, replays A, compares it with F, and exposes the verdict R;*
7. *P is the* Pkg *provenance over theorem rows, proof-ancestry reports, strict axiom-purity gates,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ForbiddenAxiomAncestry}^{\uparrow}}$ *row.*

The carrier has no coordinate for admitting a forbidden host dependency as BEDC theorem content, replacing a refused theorem by a neighboring theorem, or treating source-token purity as sufficient proof-ancestry purity.

Theorem 622.2 (ForbiddenAxiomAncestry NameCert obligations)**.** *For every* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *carrier*

$$Q = (T, A, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ForbiddenAxiomAncestry}^{\uparrow}}$ *surface consists of carrier exposure for the theorem row, proof-ancestry report, forbidden dependency row, refusal verdict, componentwise* hsame *transport, displayed* Cont *audit replay,* Pkg *provenance, and local naming row; exactness of the comparison between* A *and* F*; and non-escape for* `Classical.choice`*,* `Quot.sound`*,* `propext`*, fuzzy theorem substitution, and source-token-only acceptance.*

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 622.1. The packet lists exactly the rows T, A, F, R, H, C, P, N.

The ancestry comparison is local to the displayed audit surface. The row A supplies the reported host dependencies for the theorem row T, and the row F supplies the three forbidden dependency names.

The verdict row R records the finite comparison. Transport and replay are confined to H and C, so a consumer can inspect or move the same comparison only through the displayed routes.

Finally P and N record provenance and the local naming certificate. Since the excluded objects have no coordinate outside the refusal row, the NameCert surface cannot export them as BEDC theorem content. □

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier namecert obligations`

Theorem 622.3 (ForbiddenAxiomAncestry non-escape)**.** *For an accepted* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *packet, any occurrence of* `Classical.choice`*,* `Quot.sound`*, or* `propext` *in the ancestry report* A *is confined to the refusal verdict* R*. Downstream consumers may read the theorem row, ancestry report, forbidden dependency row, refusal row, transport, replay, provenance, and local naming certificate, but they cannot turn a refused host dependency into a BEDC theorem dependency or replace the refused row by a nearby resolving theorem name.*

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier forbidden dependency nonescape`

Proof. Project the accepted packet as $Q = (T, A, F, R, H, C, P, N)$. The theorem row and ancestry report are visible only as the displayed rows T and A.

Compare A with the forbidden dependency row F. If the report contains one of the listed host dependencies, the result of that comparison is stored in R as refusal content.

The remaining rows do not change that classification. The hsame row H moves the displayed packet componentwise, the Cont row C replays the audit read, and P, N record provenance and the local name.

No coordinate stores an admission rule for a forbidden host dependency, and no coordinate stores a fuzzy replacement for the theorem row. Thus refused ancestry cannot escape the packet as BEDC theorem content. □

Theorem 622.4 (ForbiddenAxiomAncestry comparison exactness)**.** *For an accepted* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *packet*

$$Q = (T, A, F, R, H, C, P, N),$$

the comparison between the ancestry report A and the forbidden row F is exhausted by the displayed audit route C. A consumer may replay T, read A, compare it with F, and expose the verdict R only through C, H, P, N, and it cannot read any other host dependency row as part of the comparison.

Proof. Project the accepted carrier. The theorem row T, ancestry report A, forbidden row F, verdict R, transport row H, audit route C, provenance row P, and local name N are the only displayed coordinates.

Use the NameCert obligations. They make the comparison between A and F one of the local certificate rows and place the audit replay in C.

Apply non-escape. Any attempted read of a further host dependency row would have to add a coordinate outside Q or reinterpret a refused dependency as BEDC theorem content. Both moves are excluded by the carrier and the refusal boundary, so the comparison is exact. □

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier comparison exactness`

Theorem 622.5 (ForbiddenAxiomAncestry refusal determinacy)**.** *Let*

$$Q = (T, A, F, R, H, C, P, N) \quad \textit{and} \quad Q' = (T, A, F, R', H', C, P', N')$$

be accepted $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *packets with the same theorem row, ancestry report, forbidden row, and transported audit route. Then the refusal verdicts agree as displayed verdict rows: R and R' are the same refusal boundary for the comparison of A with F.*

Proof. Project both accepted packets. Their theorem row, ancestry report, forbidden row, and audit replay route are the same displayed rows T, A, F, C.

Apply comparison exactness to each packet. In both packets the verdict is obtained only by replaying A against F through the shared route C; no extra host dependency row or substitute theorem row can participate.

Use the NameCert obligations to compare the transported structural rows. The transports, provenance rows, and local names authenticate the same finite comparison route, so the two verdict coordinates record the same refusal boundary. □

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier refusal determinacy`

Theorem 622.6 (ForbiddenAxiomAncestry transport replay)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ *packet*

$$Q = (T, A, F, R, H, C, P, N)$$

preserves the audit replay and the refusal boundary. The transported packet reads the transported theorem row, ancestry report, forbidden row, verdict, structural rows, provenance row, and local name through the transported H and the displayed route C, and it exports no additional host dependency row.

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier transport replay`

Proof. Begin with the componentwise transport row H. It transports the theorem row, ancestry report, forbidden row, verdict, audit route, provenance row, and local naming row as components of one packet.

Replay the audit through C. The transported theorem row still reads the transported ancestry report before comparison with the transported forbidden row, and the transported verdict remains attached to that replay.

Finally apply comparison exactness and non-escape. Exactness keeps the replay on the displayed route, while non-escape prevents transport from manufacturing a new host dependency row or admitting a refused dependency as BEDC theorem content. □

Theorem 622.7 (ForbiddenAxiomAncestry accepted clean criterion)**.** *For an accepted* ForbiddenAxiomAncestry$^{\uparrow}$ *packet*

$$Q = (T, A, F, R, H, C, P, N),$$

the packet is clean exactly when the comparison route from the proof-ancestry report A to the forbidden row F records no occurrence of `Classical.choice`*,* `Quot.sound`*, or* `propext` *in the refusal verdict R.*

Proof. Project the accepted packet to its ancestry report A, forbidden row F, refusal verdict R, and audit route C. The carrier exposes no other coordinate that can participate in the strict axiom-purity comparison.

For the forward direction, assume the packet is clean. By comparison exactness, the comparison of A with F is exhausted by the displayed route C and its displayed verdict R. Since F is precisely the row presented for the forbidden host names, a clean packet leaves R with no occurrence of `Classical.choice`, `Quot.sound`, or `propext`.

For the reverse direction, assume the comparison route records none of those three names in R. Apply comparison exactness to rule out any hidden comparison row beyond C, and apply non-escape to rule out admitting a forbidden host dependency through transport, provenance, or a neighboring theorem row. The displayed verdict therefore exhausts the strict axiom-purity read, so the accepted packet is clean. □

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier accepted clean criterion`

Theorem 622.8 (ForbiddenAxiomAncestry forbidden row coverage)**.** *For an accepted* ForbiddenAxiomAncestry$^{\uparrow}$ *packet*

$$Q = (T, A, F, R, H, C, P, N),$$

the forbidden row F exhausts precisely the three excluded host axiom names `Classical.choice`*,* `Quot.sound`*, and* `propext`*. A downstream consumer cannot enlarge the forbidden row without leaving the displayed packet Q.*

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier forbidden row coverage`

Proof. Read the carrier definition. The coordinate F is the forbidden dependency row, and its displayed content is exactly the three host axiom names excluded by the strict axiom-purity boundary.

Use the NameCert obligations. They expose F as one of the local certificate rows and make the comparison between A and F part of the authenticated packet surface.

If a consumer enlarged the forbidden row, the added host name would need a new displayed coordinate or a reinterpretation of F. The carrier supplies no such coordinate, and the NameCert obligations authenticate the original row, so the enlarged object is not the packet Q. □

Theorem 622.9 (ForbiddenAxiomAncestry consumer replay completeness)**.** *Every downstream strict axiom-purity consumer of an accepted* ForbiddenAxiomAncestry$^{\uparrow}$ *packet receives exactly*

$$T, A, F, R, H, C, P, N$$

and can replay the clean or refused verdict through C, H, P, N without importing a host proof dependency.

Lean4 ✓ `BEDC.Derived.ForbiddenAxiomAncestryUp.ForbiddenAxiomAncestryCarrier consumer replay completeness`

Proof. Start from the carrier exposure. A consumer receives the theorem row T, ancestry report A, forbidden row F, refusal verdict R, transport row H, audit route C, provenance row P, and local naming row N, and no additional host dependency coordinate.

The clean and refused cases are both read through the same displayed comparison. Apply the accepted clean criterion to identify the clean case by the absence of the three forbidden names in R, and apply forbidden row coverage to identify the refused case as meeting the finite row F.

Transport replay keeps the consumer on the displayed route after componentwise hsame movement. Non-escape prevents H, C, P, or N from importing a host proof dependency or replacing the theorem row by a nearby theorem, so the consumer replay is complete for the packet it receives. □

Closure status for $\mathsf{NameCert}_{\mathsf{ForbiddenAxiomAncestry}^{\uparrow}}$.

Constructive story: A $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ packet is generated from a theorem row, a proof-ancestry report, a forbidden dependency row for `Classical.choice`, `Quot.sound`, and `propext`, a refusal verdict, displayed hsame transports, displayed Cont audit routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForbiddenAxiomAncestry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 622.1, Definition 622.2, Definition 622.3, Definition 622.4, Definition 622.5, Definition 622.6, and Definition 622.9 specify the mature paper-side treatment of the carrier, forbidden-row coverage, accepted clean criterion, consumer replay completeness, and axiom-dependency sibling route over the theorem row, ancestry report, refusal verdict, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForbiddenAxiomAncestry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove global host-kernel purity, arbitrary theorem dependency completeness, source-token axiom scanning, Lean-side formal verification, or bridge checking for $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$.

Upgrade path: Lean verification and bridge checking require a checked realization of the same displayed refusal boundary, clean criterion, consumer replay, and axiom-dependency sibling route.

Chapter 623

A Concrete Naming Certificate for $\mathsf{FableMachineClock}^{\uparrow}$

The $\mathsf{FableMachineClock}^{\uparrow}$ interface realizes the local-clock reading of the Empty Fable Machine of chapter 9399. It is not a shared time stream and not a global order on histories. It packages one finite selector-ledger read as a local step-order packet for a single source Hist row.

The local-clock row consumes the finite selector ledger of chapter 516 before packaging step order, while the boundary chapter later recombines that ledger with the clock row.

Definition 623.1 (FableMachineClock carrier)**.** *A* $\mathsf{FableMachineClock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, I, M, S, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the source* Hist *row whose Empty-fable constructor spine is being read;*
2. *I is the finite step-index ledger for that source row, recording only the local order of inspected steps;*
3. *M is the selected mark row at each inspected step;*
4. *S is the selector witness row exposing the* SelStep *data that chooses the displayed mark from the local branch surface;*
5. *B is the local clock boundary row, stating that the step ledger belongs only to the source row H;*
6. *T records componentwise* hsame *transports for the source row, step-index ledger, selected marks, selector witnesses, clock boundary, and local naming row;*
7. *C records the displayed* Cont *routes by which a consumer reads H, I, M, S, B, and the local naming row;*
8. *P is the* Pkg *provenance row naming* Hist, Ext, Trace, SelStep, hsame, Cont, Pkg, *and the Empty-fable source packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FableMachineClock}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for a shared clock, global time, host sequence equality, global chooser function, or ambient temporal order.

Lean4 (definition) `BEDC.Derived.FableMachineClockUp.FableMachineClockUp`

Theorem 623.2 (FableMachineClock NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FableMachineClock}^{\uparrow}}$ *surface for the carrier of Definition 623.1 consists exactly of carrier admission for* K*, source-Hist admission for* H*, step-index admission for* I*, selected-mark admission for* M*, selector-witness admission for* S*, local clock-boundary admission for* B*, componentwise* hsame *transport through* T*, displayed* Cont *routing through* C*, package provenance through* P*, and the local naming row* N*.*

Every obligation is read from finite Empty-fable selector-ledger data and the BEDC BHist*,* Ext*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No obligation introduces a shared clock or a global chooser.*

Proof. Carrier admission is projection from the finite packet K. The source history, step ledger, selected marks, selector witnesses, local clock boundary, transports, continuations, provenance, and local name are precisely the displayed coordinates of Definition 623.1.

The Empty-fable data are read through the selector witness row S. Each selected mark in M is attached to a displayed SelStep row, and each such row belongs to the constructor spine of the source H.

Transport and continuation are componentwise. The row T moves only the displayed local-clock coordinates, while C records the reads available to consumers. Since the carrier has no shared-clock or chooser coordinate, the local NameCert row cannot export one. □

Lean4 ✓ `BEDC.Derived.FableMachineClockUp.FableMachineClockTasteGate single carrier alignment`

Theorem 623.3 (FableMachineClock step-ledger exactness)**.** *Let* $K = (H, I, M, S, B, T, C, P, N)$ *be an accepted* $\mathsf{FableMachineClock}^{\uparrow}$ *carrier. Every local-clock read exported from* K *is exact: each step index in* I *is paired with the selected mark row in* M*, that mark is justified by the corresponding selector witness in* S*, and the clock boundary* B *keeps the step order attached to the single source row* H*.*

Thus a downstream consumer receives a finite local step-order packet over Hist*,* Ext*,* Trace*,* SelStep*,* hsame*,* Cont*,* Pkg*, and* NameCert*, not a host-indexed time stream.*

Lean4 ✓ `BEDC.Derived.FableMachineClockUp.FableMachineClockStepLedgerExactness`

Proof. Begin with the local step ledger I. It is a displayed finite row of the carrier, so a consumer can inspect only the indices recorded there.

For each inspected index, read the selected mark from M and the selector witness from S. The witness exposes the corresponding Empty-fable branch step and prevents the mark from being treated as the value of an undisplayed selector function.

Finally read the boundary row B. It attaches the ledger to the source H and is transported only through T and C. Therefore exactness is local to the displayed packet and does not create a shared clock for other histories. □

Lean4 ✓ `BEDC.Derived.FableMachineClockUp.FableMachineClockCarrier step ledger exactness`

Theorem 623.4 (FableMachineClock no global clock boundary)**.** *Every consumer read of an accepted* $\mathsf{FableMachineClock}^{\uparrow}$ *carrier factors through* $H, I, M, S, B, T, C, P, N$*. The packet exports no shared clock, no global time parameter, no host sequence equality, no global chooser function, and no ambient temporal order relating unrelated* Hist *rows.*

The only clock-facing datum is the finite local step ledger I *constrained by the boundary row* B *for the single source* H*.*

Proof. Inspect the carrier inventory. Clock-facing information appears only in I and B: the former records the finite local order of displayed steps, and the latter states that this order belongs to H.

The selected marks and selector witnesses are still local rows. They expose the Empty-fable branch evidence for the inspected steps, but they do not provide a function choosing branches across all histories.

Package provenance and the local naming row repack this same surface for consumers. Since no coordinate names a shared clock, global time, host sequence equality, global chooser, or ambient temporal order, none can be exported through the NameCert boundary. □

Lean4 ✓ `BEDC.Derived.FableMachineClockUp.FableMachineClockCarrier no global clock boundary`

Theorem 623.5 (FableMachineClock downstream handoff)**.** *Every downstream Empty-fable, observer-locality, or Hist-time consumer of an accepted* $\mathsf{FableMachineClock}^{\uparrow}$ *carrier receives exactly the source Hist row H, finite step-index ledger I, selected mark row M, selector witness row S, local clock boundary B, componentwise transport row T, continuation row C, package provenance P, and local naming row N.*

The handoff preserves the no-global-clock boundary of Definition 623.4; it gives consumers a finite local step-order packet and no shared clock, global time, host sequence equality, global chooser function, or ambient temporal order.

Proof. Project the accepted carrier. The downstream packet is obtained by reading the displayed rows $H, I, M, S, B, T, C, P, N$ and no further coordinate.

Apply Definition 623.3. It identifies each local step index with a selected mark and a displayed selector witness, all under the local boundary row B for the source history.

Apply Definition 623.4. The consumer can reuse the local step-order packet, but it cannot promote that packet to a shared time axis or a chooser over unrelated histories. □

Closure status for $\mathsf{NameCert}_{\mathsf{FableMachineClock}^{\uparrow}}$.

Constructive story: A $\mathsf{FableMachineClock}^{\uparrow}$ packet is generated from a source Hist row, finite local step-index ledger, selected mark rows, selector witnesses, local clock boundary, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FableMachineClock}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 623.1, Definition 623.2, Definition 623.3, Definition 623.4, and Definition 623.5 bind one finite Empty-fable local-clock packet to BHist, hsame, Cont, Pkg, NameCert, and the existing TasteGate marker.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FableMachineClock}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close shared clocks, global time, host sequence equality, global chooser functions, ambient temporal order, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, step-ledger exactness, local-boundary transport, continuation routing, package provenance, and no-global-clock rows for the displayed packet.

Chapter 624

A Concrete Naming Certificate for $\mathsf{RegularCauchyZero}^{\uparrow}$

The $\mathsf{RegularCauchyZero}^{\uparrow}$ packet records the finite identity surface by which the rational zero seed is read as a constant stream, a regular rational row, a zero modulus row, a tail-meet identity row, and finally a $\mathsf{Real}^{\uparrow}$ seal. It gives tail-meet, diagonal, and completion-facing consumers a named zero source without quotient streams, host real equality, selected limits, or ambient completeness.

Definition 624.1 (Regular Cauchy zero carrier)**.** *A* $\mathsf{RegularCauchyZero}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (q, s, r, m, \mu, e, h, c, p, n)$$

with the following displayed rows:

1. *q is the* $\mathsf{Rat}^{\uparrow}$ *zero seed row;*
2. *s is the* $\mathsf{StreamName}^{\uparrow}$ *constant zero row generated from q;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero regular row read through the constant stream s;*
4. *m is the zero* $\mathsf{CauchyModulus}^{\uparrow}$ *row whose every displayed precision request is routed to the zero window;*
5. *μ is the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *identity row, meeting m with itself without changing the displayed tail threshold;*
6. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row exported only after q, s, r, m, μ have been displayed;*
7. *h records componentwise* hsame *transports for the zero seed, constant stream, regular row, modulus row, meet row, seal row, provenance, and local name;*
8. *c records the displayed* Cont *routes from q to s, from s to r, from r to m, from m to μ, and from μ to e;*
9. *p is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *n is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}$ *row.*

The carrier contains no quotient stream equality, no host zero equality, no selected real limit, no nonzero witness stream, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyZeroUp.TasteGate.RegularCauchyZeroTasteGate single carrier alignment`

Theorem 624.2 (Regular Cauchy zero NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^\uparrow}$ *surface for Definition 624.1 consists exactly of carrier admission for* Z*,* $\mathsf{Rat}^\uparrow$ *zero-seed exposure through* q*, constant* $\mathsf{StreamName}^\uparrow$ *routing through* s*,* $\mathsf{RegSeqRat}^\uparrow$ *zero-row readback through* r*, zero* $\mathsf{CauchyModulus}^\uparrow$ *exposure through* m*,* $\mathsf{CauchyModulusMeet}^\uparrow$ *identity exposure through* μ*,* $\mathsf{Real}^\uparrow$ *seal handoff through* e*, componentwise* hsame *transport through* h*, displayed* Cont *route replay through* c*,* Pkg *provenance through* p*, and local naming through* n*.*

Proof. Project the carrier rows of Z. The mathematical rows are exactly q, s, r, m, μ, e, and the structural rows are exactly h, c, p, n.

The route begins at the rational zero seed, passes through the constant stream and regular rational readback, and then displays the zero modulus before the meet identity is available. Thus every local naming obligation is witnessed by one coordinate of the finite packet.

Since the excluded objects are not coordinates and are not produced by the displayed continuations, the local certificate surface is exhausted by the listed rows. □

Theorem 624.3 (Regular Cauchy zero StreamName–Dyadic sibling route)**.** *Every accepted* $\mathsf{RegularCauchyZero}^\uparrow$ *packet routes its zero seed through the sibling finite-window surface of chapter 126 and the dyadic observation ledger of chapter 291 before any* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{RegularCauchyTailMeet}^\uparrow$*, or* $\mathsf{Real}^\uparrow$ *consumer may read it. In carrier coordinates*

$$Z = (q, s, r, m, \mu, e, h, c, p, n),$$

the seed is read first as q*, then as the constant window row* s*, then through the regular row* r*, the zero modulus row* m*, the identity meet row* μ*, and finally the seal row* e*; the rows* h, c, p, n *record the* hsame *transports,* Cont *route replay,* Pkg *provenance, and local* NameCert *naming for that sibling route.*

Proof. Project the carrier rows $q, s, r, m, \mu, e, h, c, p, n$ from the accepted packet. The zero seed is the row q, and the existing $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^\uparrow}$ obligations force the constant stream, regular readback, modulus, meet, seal, transport, continuation, provenance, and naming rows to be exactly the displayed coordinates.

Use Definition 624.2 to identify the finite route-visible surface. The stream coordinate s is the packet's only $\mathsf{StreamName}^\uparrow$ window row, and it is the row supplied by the sibling finite-window chapter chapter 126.

The zero modulus and meet rows m, μ expose the dyadic observation boundary before the tail-meet or real-seal consumers can inspect the seed. That boundary is the dyadic ledger supplied by chapter 291, routed through c and recorded in p.

Therefore any $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{RegularCauchyTailMeet}^\uparrow$, or $\mathsf{Real}^\uparrow$ consumer reads the zero seed only after the StreamName and DyadicRatCore sibling surfaces have been traversed. No kernel-only floating zero source remains outside the displayed packet route. □

Theorem 624.4 (Regular Cauchy zero tail-meet identity)**.** *For an accepted* $\mathsf{RegularCauchyZero}^\uparrow$ *carrier* $Z = (q, s, r, m, \mu, e, h, c, p, n)$*, the tail-meet route exported through* μ *has source modulus* m *on both sides and returns the same displayed zero tail threshold. Every* $\mathsf{RegularCauchyTailMeet}^\uparrow$ *or* $\mathsf{FiniteTailDiagonalSeal}^\uparrow$ *consumer of the zero source therefore reads the identity meet row rather than a second modulus or an external equality proof.*

Proof. Read the carrier rows m and μ. The row m is the only modulus coordinate in Z, and μ is the only meet coordinate.

The continuation row c routes the regular readback through m before it reaches μ. The transport row h preserves this same modulus row componentwise, so the two sides of the displayed meet cannot be separated into different modulus sources.

Thus a tail-meet or finite-tail diagonal consumer receives the zero threshold through the identity meet row of the packet and not through host zero equality. □

Theorem 624.5 (Regular Cauchy zero tail-meet idempotent replay)**.** *For an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet* $Z = (q, s, r, m, \mu, e, h, c, p, n)$*, replaying the exported identity meet row* μ *as both tail-meet inputs returns the same displayed zero tail threshold* m *and the same meet row* μ*. Repeated* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$*,* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumers therefore cannot obtain a second modulus, host equality proof, selected limit, nonzero witness, or ambient completeness row.*

Proof. Apply Definition 624.2 to the accepted packet. The local certificate surface has exactly one modulus-facing coordinate m and exactly one meet-facing coordinate μ.

Apply Definition 624.4 to both replay inputs. Since both inputs are the displayed identity meet row, the replay has the same source modulus on both sides and returns the same zero threshold.

For downstream reads, use Definition 624.7 together with Definition 624.6. These results exhaust the consumer-visible boundary by the displayed rows, so repeated consumers can reuse m and μ but cannot manufacture any additional modulus, equality proof, selected limit, nonzero witness, or completeness row. □

Theorem 624.6 (Regular Cauchy zero RealUp seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet factors through*

$$q \longmapsto s \longmapsto r \longmapsto m \longmapsto \mu \longmapsto e.$$

The seal receives only the zero seed, constant stream, zero regular row, zero modulus row, meet identity row, componentwise transports, continuation routes, package provenance, and local naming data.

Proof. The seal row e is the only $\mathsf{Real}^{\uparrow}$ coordinate in the carrier. The displayed continuations reach it only after the zero seed, stream, regular row, modulus row, and meet identity have been read.

Apply Definition 624.2 to identify the full route-visible surface. The structural rows h, c, p, n account for transport, route replay, provenance, and local naming.

No quotient stream, host zero equality, selected real limit, nonzero witness stream, or ambient completeness principle appears in the packet, so the seal cannot export such data. □

Lean4 ✓ `BEDC.Derived.RegularCauchyZeroUp.RegularCauchyZeroCarrier real seal nonescape`

Theorem 624.7 (Regular Cauchy zero downstream handoff)**.** *The packet of Definition 624.1 gives downstream* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$*,* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers a single zero-source boundary:*

$$(q, s, r, m, \mu, h, c, p, n).$$

Any such consumer may use the zero seed, constant stream, zero regular row, zero modulus row, meet identity row, displayed transports, continuation routes, package provenance, and local naming row, and it cannot read quotient streams, host real equality, selected limits, nonzero witnesses, or ambient completeness from the handoff.

Proof. For tail-meet and finite-tail diagonal consumers, use Definition 624.4. It confines the shared threshold to the displayed zero modulus and identity meet rows.

For limit-transport and real-seal consumers, use Definition 624.6. It routes every seal-facing read through the same finite zero-source boundary before e is exported.

The carrier has no further coordinates. Therefore every downstream handoff is exhausted by the displayed packet rows and cannot draw on an external equality or completeness object. □

Theorem 624.8 (Regular Cauchy zero terminal consumer uniqueness)**.** *Let Z and Z' be accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packets whose displayed zero-source rows*

$$(q, s, r, m, \mu, h, c, p, n)$$

agree componentwise. Then every $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *terminal consumer reads the same zero-source boundary from Z and Z'. No quotient stream equality, host zero equality, selected real limit, nonzero witness, or ambient completeness coordinate can distinguish the two terminal reads.*

Proof. Apply Definition 624.2 to both accepted packets. Their local certificate surfaces expose only the finite carrier rows, and the hypotheses identify the zero seed, constant stream, regular readback, modulus, meet, transport, continuation, provenance, and local naming rows componentwise.

Next use Definition 624.3. The common rows force both packets through the same $\mathsf{StreamName}^{\uparrow}$ constant-window route and the same dyadic observation ledger before any regular-Cauchy, tail-meet, or real-facing consumer can read the zero source.

For tail-meet and finite-tail diagonal consumers, apply Definition 624.4. Since the common boundary includes m and μ, both packets export the same zero modulus and the same identity meet row to those consumers.

For limit-transport and real-seal consumers, apply Definition 624.6 and Definition 624.7. These results exhaust the terminal consumer-visible boundary by the displayed rows, so a consumer has no remaining coordinate through which Z and Z' can be separated. □

Definition 624.9 (Regular Cauchy zero request bundle surface)**.** *For an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *carrier*

$$Z = (q, s, r, m, \mu, e, h, c, p, n),$$

a regular Cauchy zero request bundle surface is a finite $\mathsf{ProbeBundle}$ *of unary precision requests whose every member is displayed through the same route*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e.$$

Each request may read the rational zero seed, the constant $\mathsf{StreamName}^{\uparrow}$ *window, the zero* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the zero* $\mathsf{CauchyModulus}^{\uparrow}$ *row, the identity* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and the supporting* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of Z. The surface has no coordinate for quotient streams, host zero equality, selected limits, nonzero witnesses, or ambient completeness.*

Theorem 624.10 (Regular Cauchy zero ProbeBundle base window)**.** *For an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *carrier Z, the empty finite* $\mathsf{ProbeBundle}$ *of unary precision requests contributes no new route. Its request-bundle surface is covered by the already displayed carrier rows*

$$q, s, r, m, \mu, e, h, c, p, n$$

of Definition 624.1, and therefore it cannot expose quotient streams, host zero equality, selected limits, nonzero witnesses, or ambient completeness.

Proof. Use Definition 624.2. The empty bundle has no consumer request to place, so the only visible data is the carrier surface already certified by the local $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}$ row.

Apply Definition 624.3 and Definition 624.6 to read the available route. It is the displayed zero route through q, s, r, m, μ, e with support h, c, p, n.

Since no request is adjoined, the empty surface cannot introduce a second stream, a host equality witness, a selected limit, a nonzero source, or a completeness object. □

Theorem 624.11 (Regular Cauchy zero ProbeBundle cons window)**.** *Let $Z = (q, s, r, m, \mu, e, h, c, p, n)$ be an accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *carrier. If a finite* ProbeBundle *of unary precision requests for Z is covered by the regular Cauchy zero request bundle surface of Definition 624.9, then adjoining one further unary precision request preserves the same constant zero stream window, dyadic observation ledger, zero modulus row, identity meet row, and* $\mathsf{Real}^{\uparrow}$ *seal route:*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e.$$

The enlarged bundle still reads only the supporting BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Z.*

Proof. Read the adjoined unary request at the point where it asks for a precision window. By Definition 624.3, any such read is forced first through the constant $\mathsf{StreamName}^{\uparrow}$ row s and the dyadic observation ledger carried by the displayed route.

Next apply Definition 624.4. The request cannot select a second modulus, because the only modulus-facing coordinate is m and the only meet-facing coordinate is the identity row μ.

Finally apply Definition 624.6 and Definition 624.8. The terminal read still reaches e through the same zero-source boundary, so adjoining the request preserves the existing finite bundle coverage rather than adding an external route. □

Theorem 624.12 (Regular Cauchy zero ProbeBundle induction coverage)**.** *For every accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *carrier $Z = (q, s, r, m, \mu, e, h, c, p, n)$ and every finite* ProbeBundle *of unary precision requests for Z, there is a single displayed zero-source surface for the whole bundle:*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e$$

with support h, c, p, n over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Thus finite request coverage for* $\mathsf{RegularCauchyZero}^{\uparrow}$ *excludes quotient streams, host zero equality, selected limits, nonzero witnesses, and ambient completeness.*

Proof. Induct on the finite ProbeBundle.

At the empty boundary, apply Definition 624.10. The empty bundle is already covered by the carrier rows and contributes no consumer route outside them.

For the successor step, assume the smaller bundle has the displayed zero-source surface. Apply Definition 624.11 to the adjoined unary request. The successor bundle keeps the same constant stream window, dyadic ledger, zero modulus, identity meet, and $\mathsf{Real}^{\uparrow}$ seal route.

The route itself is pinned by Definition 624.2, Definition 624.3, Definition 624.4, Definition 624.6, and Definition 624.8. Therefore the finite induction produces one bundle-wide surface and leaves no coordinate through which a quotient stream, host equality, selected limit, nonzero witness, or completeness principle could be read. □

Theorem 624.13 (Regular Cauchy zero obligation closure package)**.** *The carrier definition of Definition 624.1, together with the visible rows certified in Definition 624.2, Definition 624.3, Definition 624.4, Definition 624.6, Definition 624.7, and Definition 624.12, forms the full* $\mathsf{RegularCauchyZero}^{\uparrow}$

obligation package. The package consists exactly of the rational zero seed, the constant $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero row, the* $\mathsf{CauchyModulus}^{\uparrow}$ *zero row, the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *identity row, the* $\mathsf{Real}^{\uparrow}$ *seal route, finite unary request-bundle coverage, and downstream non-escape, all as visible finite rows over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with Definition 624.1. Its coordinates display the rational zero seed, constant stream row, regular rational row, zero modulus row, meet row, seal row, transports, continuations, provenance, and local naming row.

Apply Definition 624.2 to identify those coordinates as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}$ surface. Apply Definition 624.3 to force the zero seed through the constant $\mathsf{StreamName}^{\uparrow}$ window and dyadic observation boundary before any regular-Cauchy or real-facing consumer can read it.

Apply Definition 624.4 to pin the modulus meet to the displayed zero row, and apply Definition 624.6 to route the $\mathsf{Real}^{\uparrow}$ seal through the same finite chain. Apply Definition 624.12 to cover every finite unary request bundle by the same zero-source surface.

Finally, Definition 624.7 exhausts the consumer-visible boundary, so no quotient stream, host equality, selected limit, nonzero witness, or ambient completeness object remains outside the listed obligation rows. □

Theorem 624.14 (Regular Cauchy zero scoped route)**.** *Every accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet scopes its zero source through the finite route*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e.$$

The rows h, c, p, n *provide componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *structure for that route. Consequently the zero packet is scoped by* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{CauchyModulusMeet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and still exports no quotient stream, host zero equality, selected real limit, nonzero witness stream, or ambient completeness.*

Proof. Apply Definition 624.2 to expose the carrier coordinates and the local certificate rows. The zero source begins as the rational seed q and is accepted only with the structural coordinates h, c, p, n.

Apply Definition 624.3. It forces the seed through s and then into the regular row r before any regular-Cauchy or completion-facing consumer reads the packet.

Apply Definition 624.4 to identify the zero modulus and meet segment $r \mapsto m \mapsto \mu$. Apply Definition 624.6 to extend the same route to the $\mathsf{Real}^{\uparrow}$ seal row e.

Finally use Definition 624.7. The downstream boundary is exhausted by the displayed zero-source rows and structural rows, so the scoped route has exactly the listed $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert dependencies and cannot export quotient streams, host zero equality, selected limits, nonzero witnesses, or ambient completeness. □

Theorem 624.15 (Regular Cauchy zero public zero-source export)**.** *Every accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet exports a unique public zero-source boundary*

$$(q, s, r, m, \mu, e, h, c, p, n)$$

to $\mathsf{RegSeqRat}^{\uparrow}$*, tail-meet, limit-transport, and* $\mathsf{Real}^{\uparrow}$ *consumers. The public boundary is the finite route*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e$$

with structural support h, c, p, n, and no public consumer obtains an additional quotient stream, host zero equality, selected real limit, nonzero witness stream, or ambient completeness principle from the packet.

Proof. Start from the carrier of Definition 624.1. The accepted packet has exactly the displayed coordinates, and Definition 624.2 identifies them as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}$ rows over BHist, hsame, Cont, and Pkg.

Apply Definition 624.3 to route the seed q through the constant stream row s and the regular rational row r. Apply Definition 624.4 to force the modulus and meet segment $r \mapsto m \mapsto \mu$.

Apply Definition 624.6 and Definition 624.7. These results place the terminal $\mathsf{Real}^{\uparrow}$ seal at e and exhaust the $\mathsf{RegSeqRat}^{\uparrow}$, tail-meet, limit-transport, and $\mathsf{Real}^{\uparrow}$ reads through the same public boundary.

Finally use Definition 624.12 and Definition 624.14. ProbeBundle induction shows that every finite public request keeps the same zero-source route, while the scoped route theorem excludes any external consumer coordinate. □

Theorem 624.16 (Regular Cauchy zero public consumer factorization)**.** *For every accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet $Z = (q, s, r, m, \mu, e, h, c, p, n)$, every public read by a* $\mathsf{RegSeqRat}^{\uparrow}$*, tail-meet, limit-transport, or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the chain*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e.$$

The factorization is public in the sense that h, c, p, n only replay the displayed route through hsame*,* Cont*,* Pkg*, and* NameCert*; they do not add a second zero source or a second terminal seal.*

Proof. Read the consumer at its first visible precision boundary. By Definition 624.3, the read must pass from q to s and then to r before it can become regular-rational or completion-facing data.

For tail-meet consumers, apply Definition 624.4. The only modulus-facing and meet-facing coordinates are m and μ, so the read factors through $r \mapsto m \mapsto \mu$.

For limit-transport and $\mathsf{Real}^{\uparrow}$ consumers, apply Definition 624.6. The terminal seal is the row e, and the non-escape result prevents a selected limit or host equality from replacing it.

The remaining structural fields are accounted for by Definition 624.2, Definition 624.12, and Definition 624.7. Hence every public read factors through the displayed chain and no extra zero-source or seal row is exported. □

Theorem 624.17 (Regular Cauchy zero public closure package)**.** *The scoped route of Definition 624.14, together with the public zero-source export of Definition 624.15 and the public consumer factorization of Definition 624.16, forms the* $\mathsf{RegularCauchyZero}^{\uparrow}$ *public export package. This package is the single named boundary through which* $\mathsf{RegSeqRat}^{\uparrow}$*, tail-meet, limit-transport, and* $\mathsf{Real}^{\uparrow}$ *consumers may cite the regular Cauchy zero source.*

Proof. The scoped route theorem supplies the finite dependency spine from $\mathsf{Rat}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with support in BHist, hsame, Cont, Pkg, and NameCert.

The public zero-source export theorem turns that spine into the consumer-facing boundary $(q, s, r, m, \mu, e, h, c, p, n)$. The public consumer factorization theorem then shows that every relevant public read factors through $q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e$.

Together these results package the carrier, local NameCert obligations, sibling route, tail-meet identity, RealUp non-escape, ProbeBundle induction, downstream handoff, and scoped route as one public object surface for downstream citation. □

Theorem 624.18 (Regular Cauchy zero diagonal selector handoff)**.** *Let*

$$Z = (q, s, r, m, \mu, e, h, c, p, n)$$

be an accepted $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet whose public zero-source boundary supplies the zero seed* q*, constant* $\mathsf{StreamName}^{\uparrow}$ *window* s*, zero* $\mathsf{RegSeqRat}^{\uparrow}$ *row* r*, dyadic observation route, zero modulus* m*, identity meet* μ*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* e*, and structural rows* h, c, p, n*. For any accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *source-schedule read*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

whose source row, precision request, modulus row, finite window, dyadic readback, tail route, and seal row agree with this zero-source boundary, the read factors through the displayed selector route

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

The handoff is a finite BEDC row-routing statement: it exports no quotient stream, host zero equality, selected limit, countable-choice row, hidden tail oracle, or ambient completeness principle.

Proof. First apply Definition 624.17. It packages the zero seed, constant stream window, regular rational zero row, zero modulus, identity meet, terminal seal, and structural rows as the public $\mathsf{RegularCauchyZero}^{\uparrow}$ boundary available to downstream consumers.

Apply Definition 624.16. Every public completion-facing read of Z passes through $q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e$, so the source-schedule rows supplied to G have no second zero source or second seal outside that boundary.

Now use Definition 697.21. For the accepted diagonal selector packet, every Real-completion source read must begin with the displayed prefix R, n, μ, k and may not replace it by a second selector, quotient stream, host-selected limit, countable-choice row, hidden tail oracle, or ambient completeness coordinate.

Finally apply Definition 697.22. The agreed zero-source rows are consumed through the compatible finite W, D, T route before the seal row S is read. Hence the handoff factors through $R, n, \mu, k, W, D, T, S, H, C, P, N$ as finite BEDC row routing, not as a completeness theorem or a theorem of host equality. □

Theorem 624.19 (Regular Cauchy zero abstract bridge surface)**.** *The public package of Definition 624.17 and the selector handoff of Definition 624.18 bridge every accepted concrete packet*

$$(q, s, r, m, \mu, e, h, c, p, n)$$

to the abstract regular-Cauchy zero interface read by $\mathsf{RegSeqRat}^{\uparrow}$*, tail-meet, diagonal-selector, and* $\mathsf{Real}^{\uparrow}$ *consumers. The bridge surface is exactly the finite route through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*:*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e.$$

It exports this route as finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data, and it exports no quotient stream, host zero equality, selected limit, countable choice, hidden tail oracle, or ambient completeness principle.*

Proof. Begin with Definition 624.17. It packages the concrete carrier, the zero-source export, the scoped route, and the public consumer factorization as the named boundary available to $\mathsf{RegSeqRat}^{\uparrow}$, tail-meet, limit-transport, and $\mathsf{Real}^{\uparrow}$ reads.

Use Definition 624.18 at the completion-facing selector boundary. The agreed rows of the diagonal selector read consume the same zero seed, constant stream window, regular rational row, modulus, meet, and terminal seal, so the selector interface receives the packet through the same finite route rather than through a host-selected limit.

The route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ is supplied by Definition 624.3, while Definition 624.4 and Definition 624.6 carry the route across the meet row and the terminal $\mathsf{Real}^{\uparrow}$ seal. Thus the abstract interface sees only the displayed BEDC row chain.

Finally apply Definition 624.16 and Definition 624.12. These results exhaust public reads and finite request bundles through the same packet coordinates, which refuses quotient streams, host zero equality, selected limits, countable choice, hidden tail oracles, and ambient completeness as extra bridge data. □

Theorem 624.20 (Regular Cauchy zero mature standard boundary)**.** *Every standard-facing read of the regular Cauchy zero interface factors through the finite BEDC row route*

$$q \mapsto s \mapsto r \mapsto m \mapsto \mu \mapsto e$$

together with the public consumer rows h, c, p, n. Thus a read by $\mathsf{RegSeqRat}^{\uparrow}$, tail-meet, diagonal-selector, limit-transport, or $\mathsf{Real}^{\uparrow}$ may inspect only the zero seed q, the constant stream row s, the regular rational row r, the modulus row m, the meet row μ, the seal row e, and the public handoff coordinates h, c, p, n recorded by the packet.

Proof. Apply Definition 624.17 to expose the carrier, obligation package, scoped route, and public export as the boundary available to downstream reads.

Next apply Definition 624.18. It forces the completion-facing selector read to consume the same zero-source rows and the same public handoff coordinates, so the diagonal interface cannot introduce a separate selected limit.

Finally apply Definition 624.19. The abstract bridge surface identifies the standard-facing route with the displayed q, s, r, m, μ, e chain and exports it only as finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data. Hence every standard-facing read factors through the stated finite boundary. □

Theorem 624.21 (Regular Cauchy zero terminal interface completeness)**.** *The tail-meet, diagonal-selector, limit-transport, and $\mathsf{Real}^{\uparrow}$ consumers exhaust exactly the same finite zero-source rows named by the regular Cauchy zero packet. No such consumer can demand a quotient stream, host zero equality, selected limit, countable choice, hidden tail oracle, or ambient completeness principle beyond the displayed route and the public handoff coordinates.*

Proof. The tail-meet consumer is already confined by Definition 624.4, which reads the zero route through the identity meet row μ and not through an ambient completion object.

The diagonal-selector and limit-transport consumers are confined by Definition 624.18. Their accepted rows are the same $q, s, r, m, \mu, e, h, c, p, n$ rows, so neither consumer can add a chosen real limit or a countable selection principle.

The $\mathsf{Real}^{\uparrow}$ consumer is confined by Definition 624.6 and by Definition 624.19. The seal row e is terminal for this packet, and the bridge exports only finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data. The listed external demands therefore have no coordinate in the terminal interface. □

Theorem 624.22 (Regular Cauchy zero mature closure package)**.** *The regular Cauchy zero chapter supplies a mature paper-side package consisting of the carrier, obligation package, scoped route, public export, abstract bridge, mature standard boundary, and terminal interface completeness. This package is the final paper-side L10 zero-source interface for downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, while Lean verification and bridge checking remain separate verification-axis work.*

Proof. Use the carrier and local row construction established before Definition 624.13; the obligation package records the finite NameCert duties for the zero seed, stream row, regular-rational row, modulus row, meet row, and terminal seal.

Apply Definition 624.14 and Definition 624.17. These results promote the local packet to the scoped route and public export read by downstream consumers.

Apply Definition 624.19, Definition 624.20, and Definition 624.21. Together they identify the standard-facing finite boundary and show that every terminal consumer remains inside the same zero-source rows. This assembles the stated mature paper-side package without asserting a Lean-side theorem or a checked standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyZero}^{\uparrow}$ packet is generated from finite BHist rows for the rational zero seed, constant $\mathsf{StreamName}^{\uparrow}$ row, zero $\mathsf{RegSeqRat}^{\uparrow}$ readback, zero modulus row, identity meet row, and terminal $\mathsf{Real}^{\uparrow}$ seal. The route starts at the displayed zero seed and is replayed by Cont through stream, regular-rational, modulus, meet, and seal rows before any completion-facing consumer reads it. Componentwise hsame preserves the same zero route, while Pkg records the rational, stream, regular-sequence, modulus, tail-meet, diagonal-seal, and Real dependencies. The packet names the finite zero source rather than host real equality.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: This chapter binds the regular Cauchy zero carrier, sibling route, tail-meet identity, RealUp seal non-escape, finite request-bundle coverage, downstream handoff exhaustion, the obligation package of Definition 624.13, the scoped route of Definition 624.14, the public export package of Definition 624.17, the abstract bridge surface of Definition 624.19, the mature standard boundary of Definition 624.20, the terminal interface completeness theorem of Definition 624.21, and the mature closure package of Definition 624.22.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyZero}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyZeroUp.TasteGate.RegularCauchyZeroTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: It does not claim quotient stream equality, host zero equality, selected real limits, nonzero witness streams, or ambient completeness.

Upgrade path: No further paper-side theory closure remains; Lean verification and bridge checking remain separate verification-axis obligations.

Chapter 625

A Concrete Naming Certificate for $\mathsf{TypePreservingCompiler}^{\uparrow}$

The $\mathsf{TypePreservingCompiler}^{\uparrow}$ interface joins the type-checking classifier surface with the certificate-morphism compiler surface. It records source and target type-classifier packets, a compiler graph between their displayed histories, subject-reduction rows on both sides, and preservation ledgers ensuring that every compiled output is read through target Ext membership rather than through a host semantic compiler.

Definition 625.1 (Type-preserving compiler carrier)**.** *A* $\mathsf{TypePreservingCompiler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, G, E, R, H, C, P, N)$$

with the following visible coordinates:

1. *S is the source* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *row, including its displayed term history, judgment ledger,* Ext *membership row, and typed* Cont *reduction row;*
2. *T is the target* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *row, including the target term history, judgment ledger, target* Ext *membership row, and target typed* Cont *reduction row;*
3. *G is the compiler graph row from source histories accepted by S to target histories displayed by T;*
4. *E is the classifier-preservation ledger recording that source* Ext *membership transported along G lands in target* Ext *membership;*
5. *R records source and target* $\mathsf{SubjectReduction}^{\uparrow}$ *rows and the handoff from compiled source steps to target typed reduction reads;*
6. *H stores componentwise* hsame *transports for source rows, target rows, graph edges, classifier-preservation rows, and subject-reduction rows;*
7. *C stores displayed* Cont *routes for type-checking membership, compiler graph lifting, target-carrier readback, subject-reduction handoff, and downstream NameCert-morphism consumption;*
8. *P records* Pkg *provenance over* BHist, Ext, Cont, $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{ClassifierMorphism}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, hsame, Pkg, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypePreservingCompiler}^{\uparrow}}$ *row.*

The carrier has no coordinate for host operational semantics, erased typing derivations, quotient equality of terms, a semantic target interpreter, or a compiler function outside the displayed graph.

Theorem 625.2 (TypePreservingCompiler NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TypePreservingCompiler}^{\uparrow}}$ *surface for Definition 625.1 consists exactly of carrier admission for*

$$K = (S, T, G, E, R, H, C, P, N),$$

source and target type-classifier membership, compiler graph lifting, target Ext *preservation, source and target subject-reduction rows, componentwise* hsame *transport, displayed* Cont *routing, dependency provenance, and the local naming row.*

Proof. Project the carrier. The rows $S, T, G, E, R, H, C, P, N$ are the only coordinates available to the local certificate.

The source and target classifier reads come from S and T. Graph lifting is read from G through C, and the classifier-preservation ledger E records that a source membership read transported along the graph is consumed as target membership.

The subject-reduction component is the row R, transported by H and routed by C. Since the carrier contains no host compiler function, semantic interpreter, quotient equality, or erased derivation row, the listed coordinates exhaust the local $\mathsf{NameCert}_{\mathsf{TypePreservingCompiler}^{\uparrow}}$ obligations. □

Lean4 ✓ `BEDC.Derived.TypePreservingCompilerUp.TypePreservingCompilerCarrier namecert obligations`

Theorem 625.3 (TypePreservingCompiler subject-reduction handoff)**.** *Let* $K = (S, T, G, E, R, H, C, P, N)$ *be an accepted* $\mathsf{TypePreservingCompiler}^{\uparrow}$ *carrier. Every public read that starts with source* Ext *membership in* S*, follows a source typed* Cont *step, and then compiles through* G *reaches the target typing surface only through the target* Ext *row of* T *and the subject-reduction ledger* R*. The handoff exports no host reduction semantics, erased typing derivation, or global normalization theorem.*

Proof. Read the source side through S. The source membership row and typed Cont row are already part of the source $\mathsf{TypeClassifierMembership}^{\uparrow}$ coordinate.

Apply the graph-lifting and classifier-preservation rows G, E. They move the source membership read into a target history displayed by T and record the corresponding target Ext membership.

It remains to account for reduction preservation. The row R stores the source and target $\mathsf{SubjectReduction}^{\uparrow}$ data, while H and C transport and replay the handoff. No other coordinate can supply host reduction semantics or a normalization theorem, so every public subject-reduction read factors through the displayed target membership and ledger rows. □

Theorem 625.4 (TypePreservingCompiler morphism reading)**.** *Every accepted* $\mathsf{TypePreservingCompiler}^{\uparrow}$ *carrier determines an accepted classifier-preserving NameCert-morphism read from the source type-checking certificate to the target type-checking certificate. The morphism data are exactly the compiler graph* G*, the classifier-preservation ledger* E*, the subject-reduction handoff* R*, the transport row* H*, the route row* C*, provenance* P*, and the local naming row* N*.*

Proof. Start from the source and target certificate rows S and T. These rows provide the two visible classifier surfaces required by the NameCert-morphism reading.

The compiler graph is the displayed row G. Its graph-lifting and target-carrier readback are routed by C, while hsame preservation is supplied by H and classifier preservation by E.

Finally insert R as the type-preservation component of the morphism. It binds the source and target subject-reduction rows to the same graph route, so compiler acceptance is read as

preservation of the named classifier and reduction surfaces, not as an external semantic compiler. □

Theorem 625.5 (TypePreservingCompiler ledger boundary)**.** *The public ledger boundary of an accepted* TypePreservingCompiler$^{\uparrow}$ *carrier is exactly the displayed packet*

$$S, T, G, E, R, H, C, P, N.$$

Every downstream consumer of the compiler reads source type-checking membership, target type-checking membership, graph lifting, Ext *preservation, subject-reduction handoff,* hsame *transport,* Cont *routing, package provenance, and local* NameCert *data through those rows. No read factors through host operational semantics, quotient equality of terms, semantic target interpretation, erased derivation representatives, or an unrecorded compiler function.*

Proof. The carrier definition gives the complete inventory of public rows. The NameCert-obligation theorem then binds carrier admission, classifier membership, graph lifting, target preservation, subject reduction, transport, routing, provenance, and local naming to that inventory.

For type preservation, apply the subject-reduction handoff. It forces every source reduction and membership read to reach the target through T and R after the graph route has been fixed.

For compiler consumption, apply the morphism reading. It repacks the same finite rows as a classifier-preserving NameCert morphism. The carrier contains no further coordinate, so no host semantics, quotient equality, erased derivation, or unrecorded compiler function can be exported. □

Closure status for NameCert$_{\text{TypePreservingCompiler}^{\uparrow}}$**.**

Constructive story: A TypePreservingCompiler$^{\uparrow}$ packet is generated from source and target type-classifier membership rows, a compiler graph row, classifier-preservation and subject-reduction ledgers, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TypePreservingCompiler}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 625.1, Definition 625.2, Definition 625.3, Definition 625.4, Definition 625.5, Definition 625.6, and Definition 625.7 bind the finite classifier-preserving compiler obligation surface.

Formal verification: FormalStatus(NameCert$_{\text{TypePreservingCompiler}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.TypePreservingCompilerUp.TypePreservingCompilerCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host compiler semantics, quotient equality of terms, erased typing derivations, semantic target interpretation, or bridge checking.

Upgrade path: No further theory-grade upgrade remains for the finite type-preserving compiler packet; bridge checking is a separate formal-axis obligation.

Theorem 625.6 (TypePreservingCompiler sequential composition)**.** *Let*

$$K_{ST} = (S, T, G_{ST}, E_{ST}, R_{ST}, H_{ST}, C_{ST}, P_{ST}, N_{ST})$$

and

$$K_{TU} = (T, U, G_{TU}, E_{TU}, R_{TU}, H_{TU}, C_{TU}, P_{TU}, N_{TU})$$

be accepted TypePreservingCompiler$^{\uparrow}$ *carriers sharing the same middle* TypeClassifierMembership$^{\uparrow}$ *row* T*. Define the sequential compiler graph* G_{SU} *by relational composition:*

$$G_{SU}(s, u) \quad \textit{holds exactly when there is a middle history } t \textit{ with } G_{ST}(s, t) \textit{ and } G_{TU}(t, u).$$

Then G_{SU} determines an accepted TypePreservingCompiler$^{\uparrow}$ *carrier from S to U whose public packet is*

$$K_{SU} = (S, U, G_{SU}, E_{SU}, R_{SU}, H_{SU}, C_{SU}, P_{SU}, N_{SU}).$$

*Here E_{SU} is the concatenated target-*Ext *preservation ledger $E_{TU} \circ_T E_{ST}$, R_{SU} is the middle-handoff subject-reduction ledger $R_{TU} \circ_T R_{ST}$, and $H_{SU}, C_{SU}, P_{SU}, N_{SU}$ are the componentwise transported sameness, continuation-route, package-provenance, and local naming rows obtained by composing the displayed rows of K_{ST} and K_{TU} through T. Consequently every public read from source* Ext *membership in S through a source typed* Cont *step and then through G_{SU} reaches U only through the target* Ext *row of U and the composed subject-reduction ledger R_{SU}.*

Proof. Project the two accepted carriers by Definition 625.1. The first packet exposes only $S, T, G_{ST}, E_{ST}, R_{ST}, H_{ST}, C_{ST}, P_{ST}, N_{ST}$, and the second exposes only $T, U, G_{TU}, E_{TU}, R_{TU}, H_{TU}, C_{TU}, P_{TU}, N_{TU}$. The shared coordinate is the displayed type-classifier membership row T, so a relational edge from S to U can be formed only by passing through a middle history already read in T.

The classifier-preserving part is closed by Definition 590.7. Applied to the classifier-morphism reads supplied by Definition 625.4, it turns the graph rows G_{ST} and G_{TU} into the relational composite G_{SU} and composes their Ext, SigRel, Cont, and hsame preservation rows. In particular, a source membership row accepted by the Ext coordinate of S is first consumed by E_{ST} as target membership in T, and the resulting middle membership is then consumed by E_{TU} as target membership in U. This is exactly the ledger $E_{SU} = E_{TU} \circ_T E_{ST}$.

The naming-certificate morphism part is closed by Definition 481.4 and its route provenance statement Definition 481.14. These theorems concatenate graph lifting, target transport, hsame preservation, Cont routing, package provenance, and local naming through the same middle certificate row. Thus the composed graph and its transport rows repack as one finite source-to-target morphism boundary from S to U, without adding a host compiler function or quotient equality.

It remains to check the type-preserving reduction handoff. Start with a public read consisting of source Ext membership in S, a source typed Cont step, and a composite graph edge $G_{SU}(s, u)$. By definition of G_{SU} there is a displayed middle history t with $G_{ST}(s, t)$ and $G_{TU}(t, u)$. Applying Definition 625.3 to K_{ST} sends the source membership and source typed step into the middle target surface only through the target Ext row of T and the ledger R_{ST}. Applying the same theorem to K_{TU} consumes that middle membership as the source side of the second compiler packet and reaches U only through the target Ext row of U and the ledger R_{TU}.

The two subject-reduction reads are ledgered before consumption by Definition 483.4, and their Ext-Cont closure is internal by Definition 483.3. Concatenating them through the displayed middle row T gives $R_{SU} = R_{TU} \circ_T R_{ST}$. The boundary theorem Definition 625.5 then accounts for every consumer-visible row of the composite packet: source and target type-classifier membership, graph lifting, target Ext preservation, subject-reduction handoff, hsame transport, Cont routing, package provenance, and local NameCert data. Since the carrier definition has no coordinate for host reduction semantics, erased derivations, quotient equality, semantic target interpretation, or an unrecorded compiler function, the displayed packet K_{SU} is the accepted TypePreservingCompiler$^{\uparrow}$ carrier determined by the sequential composite. □

Theorem 625.7 (TypePreservingCompiler subject-reduction socket consumption)**.** *Let $K = (S, T, G, E, R, H, C, P$ be an accepted* TypePreservingCompiler$^{\uparrow}$ *carrier. Every* SubjectReduction$^{\uparrow}$ *read consumed by K enters through the displayed row R, the classifier-preservation ledger E, the componentwise transport row H, the continuation route row C, the package provenance row P, and the local* NameCert *row N. If a* SubjectReductionDischargeSocket$^{\uparrow}$ *row is present, it is read only as a finite obligation socket*

for the same R-facing subject-reduction rows. The compiler packet exports no host compiler semantics, erased derivation semantics, quotient equality, semantic target interpreter, or unrecorded compiler function.

Proof. Project the accepted carrier K. By Definition 625.3, a public subject-reduction read that starts on the source side reaches the target typing surface only through T and the subject-reduction ledger R after the graph route has been fixed.

The graph and classifier-preservation part is controlled by Definition 625.4. It reads G, E, R, H, C, P, N as the finite morphism data and therefore places every type-preserving compiler request on the displayed certificate surface.

The ledger boundary theorem Definition 625.5 and sequential-composition theorem Definition 625.6 account for the same rows under downstream consumption and composition. They expose source and target type-classifier membership, graph lifting, target Ext preservation, subject-reduction handoff, transport, routing, provenance, and local naming, but no additional compiler coordinate.

When a $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ row is available, Definition 719.4 permits the compiler chapter to read it as the finite obligation socket feeding the R coordinate. That socket does not supply a host compiler semantics, an erased derivation semantics, quotient equality, a semantic target interpreter, or an unrecorded compiler function, so the accepted compiler packet consumes subject-reduction data only through its displayed rows. □

Theorem 625.8 (TypePreservingCompiler route-classifier sibling dependency)**.** *Let $K = (S, T, G, E, R, H, C, P, N$ be an accepted $\mathsf{TypePreservingCompiler}^{\uparrow}$ carrier. When a $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ packet from chapter 785 is present, the compiler row R consumes its route-kind ledger before packaging the subject-reduction handoff. The resulting compiler packet exports no host compiler semantics, erased derivation semantics, quotient equality, semantic target interpreter, or unrecorded invocation path.*

Proof. Project the accepted compiler carrier by Definition 625.1. The row R is the only subject-reduction coordinate, and the rows H, C, P, N are the only displayed transport, route, provenance, and local naming support for its consumption.

Apply Definition 625.7. It forces every subject-reduction read used by the compiler to enter through R, E, H, C, P, N and excludes host compiler semantics, erased derivation semantics, quotient equality, semantic target interpretation, and unrecorded compiler functions.

Use the sibling packet of chapter 785. By Definition 785.5 and Definition 785.12, the route-kind ledger is exactly the finite classifier surface that distinguishes the displayed invocation route before any compiler consumer reads it.

Finally apply Definition 785.10. A $\mathsf{TypePreservingCompiler}^{\uparrow}$ consumer may read the classifier only as an invocation-style handoff for the shared discharge rows, so the compiler packages the subject-reduction handoff after that route-kind ledger and exports no unrecorded invocation path. □

Chapter 626

A Concrete Naming Certificate for $\mathsf{StationaryDiagonalWindow}^{\uparrow}$

The $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ packet realizes the stationary diagonal finite-window classifier correspondence as a concrete $\mathsf{NameCert}$ surface. It keeps the rational seed rows, the constant streamname windows, the regular-rational readback rows, the dyadic finite-window ledger, the real-facing diagonal representatives, the three classifier faces, transports, continuations, provenance, and local naming in one finite packet. It does not export quotient stream equality, host real equality, selected limit data, non-stationary tails, or ambient completeness.

Its five source rows are sibling certificate rows rather than private coordinates: Q is read from chapter 37, S from chapter 126, G from chapter 301, D from chapter 291, and R from chapter 38. Thus the diagonal-window packet records a finite Real-completion lattice edge whose classifier row can be consumed only after those five sibling surfaces have been displayed.

Falsifiable prediction: Removing any one of Q, S, G, D, R breaks the three-face classifier correspondence, because the rational seed, finite stream window, regular readback, dyadic ledger, and real-facing diagonal row are all needed before K can be read as one correspondence.

Independence witness: Nearest siblings: rat, streamname, regseqrat, dyadicratcore, and real. None is bijective to $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ because no single sibling carries the full five-row packet Q, S, G, D, R together with the shared classifier row K, while the stationary diagonal packet does not own the full internal surfaces of those sibling chapters.

Definition 626.1 (Stationary diagonal window carrier)**.** *A* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, S, G, D, R, K, H, C, P, N)$$

with the following displayed rows:

1. *Q is the paired stationary* $\mathsf{Rat}^{\uparrow}$ *seed row, including the two rational history carriers and their* $\mathsf{RatHistoryClassifier}$ *row;*
2. *S is the paired constant* $\mathsf{StreamName}^{\uparrow}$ *window row induced by Q;*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback row attached to the same selected window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger, including dyadic endpoint reads, tolerance rows, and window-bound comparisons;*
5. *R is the paired* $\mathsf{Real}^{\uparrow}$ *diagonal representative row read only after S, G, D have been displayed;*

6. *K records the three classifier faces: the stationary* $\mathsf{Rat}^{\uparrow}$ *classifier on Q, the regular-rational finite-window classifier on G, D, and the* $\mathsf{Real}^{\uparrow}$ *diagonal finite-window classifier on R;*
7. *H records componentwise* hsame *transports for Q, S, G, D, R, K and the structural rows;*
8. *C records the displayed* Cont *routes from Q to S, from S to G, from G to D, from D to R, and from the three faces of K to the common correspondence row;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWindow}^{\uparrow}}$ *row.*

The carrier is finite. It has no coordinate for quotient streams, host real equality, selected real limits, non-stationary tails, an infinite window family, or a global completeness principle.

Theorem 626.2 (Stationary diagonal window NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWindow}^{\uparrow}}$ *surface for Definition 626.1 consists exactly of carrier admission for $W = (Q, S, G, D, R, K, H, C, P, N)$, paired stationary* $\mathsf{Rat}^{\uparrow}$ *seed exposure through Q, constant* $\mathsf{StreamName}^{\uparrow}$ *window exposure through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure through G,* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger exposure through D, paired* $\mathsf{Real}^{\uparrow}$ *diagonal representative exposure through R, the three classifier faces recorded in K, componentwise* hsame *transport through H, displayed* Cont *route replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.StationaryDiagonalWindowUp.StationaryDiagonalWindowTasteGate single carrier alignment`

Proof. Carrier admission is projection from the displayed tuple W. The rows Q, S, G, D, R expose the rational seeds, constant stream windows, regular-rational readbacks, dyadic finite-window ledger, and real-facing diagonal representatives.

The row K supplies the three classifier faces required by the correspondence: stationary rational, regular-rational finite-window, and real diagonal finite-window. The continuation row C routes these faces through the same selected window, so no second classifier source is available.

The rows H, P, N account for transport, package provenance, and local naming. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 626.3 (Stationary diagonal window classifier correspondence)**.** *For an accepted* StationaryDiagonalWindo *carrier, the following three finite classifier reads have the same public finite-window content: the stationary* $\mathsf{Rat}^{\uparrow}$ *classifier displayed by Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback classifier displayed by G, D, and the* $\mathsf{Real}^{\uparrow}$ *diagonal finite-window classifier displayed by R. The correspondence is witnessed inside the finite row K and replayed only through the continuations in C.*

Proof. Read the three classifier faces from K. The first face is anchored in the stationary rational seed row Q, the second in the regular-rational readback and dyadic ledger rows G, D, and the third in the real diagonal representative row R.

The continuations in C force all three faces to use the constant stream window S selected from the same rational seed pair. Hence the regular-rational and real-facing reads cannot change the finite window inspected by the rational classifier.

The transport row H carries the same window and classifier faces componentwise. Therefore an accepted packet presents one finite-window classifier correspondence rather than three unrelated classifications. □

Theorem 626.4 (Stationary diagonal window finite-window exhaustion)**.** *Every observation exported by an accepted* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *packet factors through the finite rows*

$$Q, S, G, D, R, K, H, C, P, N.$$

In particular, the consumer may inspect only the displayed stationary rational seed, constant stream window, regular-rational readback, dyadic finite-window ledger, real diagonal representative, classifier correspondence row, transports, continuations, provenance, and local naming.

Proof. Project the accepted carrier and its local naming surface. The mathematical rows are Q, S, G, D, R, K, and the structural rows are H, C, P, N.

Apply the classifier-correspondence theorem to the only classifier row K. It routes every rational, regular-rational, and real-facing observation through the common finite window induced by S.

No carrier coordinate remains after those rows are read. Thus a downstream observation cannot range over an infinite window family, quotient stream equality, host equality, selected limits, non-stationary tails, or ambient completeness. □

Theorem 626.5 (Stationary diagonal window public boundary)**.** *The public boundary of an accepted* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *carrier is the finite tuple consisting of* Q, S, G, D, R, K *together with the structural rows* H, C, P, N*. This boundary is sufficient for Real-completion consumers that require the stationary diagonal finite-window classifier correspondence, and it withholds quotient stream equality, host real equality, selected completed limits, non-stationary tails, and any unbounded completion principle.*

Proof. The NameCert obligations identify the rows available to public consumers. The classifier-correspondence theorem shows that the public classifier content is already present in K and routed through Q, S, G, D, R.

The finite-window exhaustion theorem then rules out any hidden observation path outside the displayed packet. Hence the public boundary is exactly the finite tuple named in the statement, with the excluded analytic objects absent from the carrier. □

Theorem 626.6 (Stationary diagonal window carrier-source obligation)**.** *For an accepted* StationaryDiagonalWindo *carrier, source admission is exactly projection of the rows*

$$Q, S, G, D, R, K, H, C, P, N$$

over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*. The admission surface has no coordinate for quotient stream equality, host real equality, selected real limits, non-stationary tails, or an ambient completeness principle.*

Proof. Project the carrier of Definition 626.1. Its displayed coordinates are precisely the rational seed row, stream-name window row, regular-rational readback row, dyadic ledger row, real-facing diagonal row, classifier row, transport row, continuation row, package row, and local naming row.

The classifier-correspondence theorem and finite-window-exhaustion theorem show that every admitted mathematical read already passes through these displayed rows. The public-boundary theorem then identifies the consumer-facing boundary with the same tuple and excludes the quotient, host-equality, selected-limit, tail, and completeness coordinates named in the statement. □

Theorem 626.7 (Stationary diagonal window classifier-stability obligation)**.** *For an accepted* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *carrier, componentwise* hsame *transport through* H *and the displayed* Cont *routes in* C *preserve the three classifier faces recorded in* K *on the same selected finite window. These routes cannot replace the classifier faces by parameter-equivalence transport or by an ambient real classifier outside the packet.*

Proof. Read the three classifier faces from K in Definition 626.1. The classifier-correspondence theorem places the rational, regular-rational, and real-facing reads on the same selected finite window.

The row H transports the displayed components only componentwise, and the row C replays only the displayed continuation routes. By finite-window exhaustion and the public boundary theorem, there is no additional parameter-equivalence carrier or ambient real-classifier coordinate available to replace the three faces in K. □

Theorem 626.8 (Stationary diagonal window ledger-exactness obligation)**.** *Every Real-completion consumer read admitted by an accepted* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *packet factors through the finite rows*

$$Q, S, G, D, R, K, H, C, P, N.$$

All hidden requests, including quotient stream requests, host-equality requests, selected-limit requests, non-stationary-tail requests, and ambient-completeness requests, remain absent from the packet.

Proof. Start from the carrier definition and expose the displayed tuple. The rows Q, S, G, D, R, K contain the mathematical source and classifier data, while H, C, P, N contain transport, continuation, provenance, and local naming.

Apply the classifier-correspondence theorem to route the Real-facing classifier read through the same finite window, and apply finite-window exhaustion to rule out every consumer path outside the tuple. The public-boundary theorem then gives the exact Real-completion-facing ledger and records the absence of every hidden request named in the statement. □

Theorem 626.9 (Stationary diagonal window sibling lattice link)**.** *Every accepted* $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ *packet obtains* Q *from* $\mathsf{Rat}^{\uparrow}$*,* S *from* $\mathsf{StreamName}^{\uparrow}$*,* G *from* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *from* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* R *from* $\mathsf{Real}^{\uparrow}$ *through the same finite* BHist/hsame/Cont/Pkg/NameCert *route. Consequently the three classifier faces in* K *form a cross-sibling finite-window correspondence rather than an isolated packet.*

Proof. Project the carrier of Definition 626.1. Its source coordinates Q, S, G, D, R are read in order from the sibling surfaces chapter 37, chapter 126, chapter 301, chapter 291, and chapter 38.

Use Definition 626.3 to identify the three faces in K as the rational, regular-rational finite-window, and real diagonal finite-window reads over that shared source route. The transport and continuation coordinates H, C keep the faces on the same finite window, while P, N record package provenance and local naming.

Finally use Definition 626.8. It rules out escape through quotient streams, host equality, selected limits, non-stationary tails, and ambient completeness, so the correspondence visible in K is exactly the cross-sibling finite-window link supplied by the five displayed source rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryDiagonalWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StationaryDiagonalWindow}^{\uparrow}$ packet is generated from paired stationary RatUp seed rows, constant StreamNameUp windows, RegSeqRatUp readbacks, a DyadicRatCoreUp finite-window ledger, RealUp diagonal

representative rows, a three-face classifier correspondence row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{StationaryDiagonalWindow}^{\uparrow}}$, seed)

Scope closed: Definition 626.1, Definition 626.2, Definition 626.3, Definition 626.4, Definition 626.5, Definition 626.6, Definition 626.7, Definition 626.8, and Definition 626.9 bind the finite NameCert packet for the stationary diagonal finite-window classifier correspondence over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{StationaryDiagonalWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, host real equality, selected real limits, non-stationary tails, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, the three-face classifier correspondence, finite-window exhaustion, and public-boundary exactness for the same stationary diagonal packet.

Chapter 627

A Concrete Naming Certificate for $\mathsf{GeneratorContinuationAudit}^{\uparrow}$

The $\mathsf{GeneratorContinuationAudit}^{\uparrow}$ packet realizes the continuation-level audit surface between an authorized generator row and a downstream consumer. It keeps the generator row, continuation row, displayed closure witness, rejected-continuation GAP row, transports, routes, provenance, and local naming in one finite BHist packet. It consumes the authorized-generator classifier surface of chapter 486 and the displayed witness/GAP ledger surface of chapter 487 before exporting a continuation-level consumer route. It is a certificate for per-continuation witness preservation and GAP non-escape, not a global induction principle, host recursion rule, quotient over continuations, or selected downstream representative.

The generator row G is consumed through chapter 525. This chapter adds only the continuation audit row, preserved witness row, rejected-continuation GAP row, transports, routes, provenance, and local name around that authorized generator surface.

Definition 627.1 (Generator-continuation audit carrier)**.** *A* $\mathsf{GeneratorContinuationAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, U, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *G is the authorized generator row, including its local generator classifier and the finite generator-history entry to be audited;*
2. *U is the continuation row selected for this audit, including its source endpoint, target endpoint, and finite consumer-facing read;*
3. *W is the closure-witness row preserved by the selected continuation;*
4. *Q is the rejected-continuation GAP row, recording every continuation branch that fails the generator classifier or attempts to read a witness outside W;*
5. *H records componentwise* hsame *transports for G, U, W, Q and the structural rows;*
6. *C records the displayed* Cont *routes from the generator row to the continuation row, from the continuation row to the closure witness, and from rejected continuations to the GAP row;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, *the generator classifier, the closure-witness ledger, the GAP ledger, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorContinuationAudit}^{\uparrow}}$ *row.*

The carrier is finite. It contains no coordinate for a free-floating induction rule, host recursion, quotient equality between continuations, selected consumer representatives, or unbounded downstream completeness.

Theorem 627.2 (Generator-continuation audit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{GeneratorContinuationAudit}^{\uparrow}}$ *surface for Definition 627.1 consists exactly of carrier admission for* $A = (G, U, W, Q, H, C, P, N)$*, authorized generator exposure through* G*, selected continuation exposure through* U*, closure-witness exposure through* W*, rejected-continuation GAP exposure through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed tuple A. The rows G, U, W, Q expose the mathematical content of the audit: generator authority, one continuation, the preserved closure witness, and the GAP destination for rejected continuation reads.

The route row C supplies the only paths by which the selected continuation can reach the witness row and by which a rejected continuation can be routed to the GAP row. Thus the audit surface has no hidden path from a non-generator continuation to a downstream witness.

The rows H, P, N account for transport, package provenance, and local naming. Since free induction, host recursion, quotient continuation equality, selected representatives, and unbounded completeness have no carrier coordinate, the listed rows exhaust the local naming obligations. □

Lean4 ✓ `BEDC.Derived.GeneratorContinuationAuditUp.GeneratorContinuationAuditTasteGate single carrier alignment`

Theorem 627.3 (Generator-continuation audit witness preservation)**.** *For an accepted* GeneratorContinuationAudit *carrier, every downstream read of the selected continuation* U *that passes the generator classifier factors through the displayed closure-witness row* W*. The factorization is witnessed only by the finite route from* G *to* U *to* W *together with the recorded* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with an accepted carrier and project the generator row G, continuation row U, and closure-witness row W. The generator classifier attached to G admits the selected continuation read named by U.

Replay the continuation route recorded in C. This route first binds the continuation to the authorized generator row and then reads the displayed closure witness. Componentwise transport in H can move the same rows between equivalent presentations, but it cannot create a second witness source.

The package and naming rows P, N certify that the routed witness is the row displayed in the finite packet. Therefore the downstream consumer sees precisely the preserved witness W through the stated finite factorization. □

Theorem 627.4 (Generator-continuation audit GAP non-escape)**.** *For an accepted* $\mathsf{GeneratorContinuationAudit}^{\uparrow}$ *carrier, every continuation branch that is not admitted by the generator classifier, or that attempts to consume a closure witness outside the displayed row* W*, is routed to the rejected-continuation GAP row* Q*. No such branch reaches the downstream consumer surface through* U*.*

Proof. Project the rejected-continuation row Q and the continuation routes C. By definition of the carrier, Q records the failed generator-classifier branches and the branches whose requested witness is not the displayed row W.

For each rejected branch, use the route component of C that sends the branch to Q. The same componentwise hsame transport that preserves accepted rows also preserves the GAP destination, so transport cannot turn a rejected branch into an accepted downstream read.

The NameCert obligation surface from Definition 627.2 lists no other continuation exit. Hence rejected continuations cannot escape the GAP row or enter the downstream consumer surface. □

Theorem 627.5 (Generator-continuation audit downstream consumer surface)**.** *Every downstream consumer read exported by an accepted* $\mathsf{GeneratorContinuationAudit}^{\uparrow}$ *packet is exhausted by the finite rows* G, U, W, H, C, P, N *together with the GAP exclusion row* Q*. Accepted reads preserve the displayed witness* W*, and rejected reads are visible only through* Q*.*

Proof. Begin with an exported downstream consumer read. If the read is accepted by the generator classifier, apply Definition 627.3 to factor it through the selected continuation and the displayed closure witness.

If the read is not accepted, or if it asks for a witness outside W, apply Definition 627.4. The read is then routed to Q and has no consumer-facing exit through U.

The NameCert obligations identify the remaining structural rows as H, C, P, N. These rows transport, replay, package, and name the same finite audit packet, so the downstream consumer surface is exactly the one stated. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorContinuationAudit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GeneratorContinuationAudit}^{\uparrow}$ packet is generated from an authorized generator row, a selected continuation row, a closure-witness row, a rejected-continuation GAP row, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorContinuationAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 627.1, Definition 627.2, Definition 627.3, Definition 627.4, and Definition 627.5 bind the finite continuation-level audit packet between generator closure and downstream consumer use.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratorContinuationAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global induction, host recursion, quotient continuation equality, selected downstream representatives, or bridge checking.

Upgrade path: Scoped closure requires checked preservation and GAP non-escape rows for the same finite continuation-audit packet.

Chapter 628

A Concrete Naming Certificate for $\mathsf{ModulusIndexedTailCover}^{\uparrow}$

The $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ packet names the finite surface through which a unary precision request selects a modulus row, a tail-window cover, and a displayed tail witness before the $\mathsf{Real}^{\uparrow}$ seal is read. It sits between the witness schedule and diagonal seal surfaces: the packet remembers the request, the modulus-indexed windows, the dyadic tolerance ledger, the regular-sequence handoff, and the local naming row without adding quotient streams, host real equality, selected limits, or ambient completeness.

This packet consumes the finite-window naming discipline of chapter 126, the tolerance ledger of chapter 291, the regular-rational handoff of chapter 301, and the seal boundary of chapter 38. These are substantive carrier rows in the tail-cover packet, not narrative analogies: the downstream consumer reads window names, dyadic tolerances, regular-rational readback, and the Real seal through those displayed siblings.

Definition 628.1 (Modulus-indexed tail-cover carrier)**.** *A* $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (e, m, w, t, d, r, s, h, c, p, n)$$

with the following visible coordinates:

1. *e is the unary precision request, read as the displayed natural-indexed tolerance demand;*
2. *m is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row answering the request by a finite tail threshold;*
3. *w is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window cover selected by m for the requested precision;*
4. *t is the selected finite tail witness, listing the inspected windows and their bounded overlap data;*
5. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic split, endpoint rows, and tail-radius comparisons attached to t;*
6. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads the selected tail windows through the regular rational sequence;*
7. *s is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after e, m, w, t, d, r have been displayed;*
8. *h records componentwise* hsame *transports for the precision request, modulus row, window cover, tail witness, dyadic ledger, regular-sequence handoff, seal row, provenance, and local name;*
9. *c records the displayed* Cont *routes from e to m, from m to w, from w to t, from t to d, from d to r, and from r to s;*

10. *p is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *n is the local* $\mathsf{NameCert}_{\mathsf{ModulusIndexedTailCover}^{\uparrow}}$ *row.*

The carrier contains no unbounded tail family, no quotient stream, no host real equality, no selected limiting representative, no ambient completeness theorem, and no open-cover compactness row.

Lean4 ✓ `BEDC.Derived.ModulusIndexedTailCoverUp.ModulusIndexedTailCoverTasteGate single carrier alignment`

Theorem 628.2 (Modulus-indexed tail-cover NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ModulusIndexedTailCover}^{\uparrow}}$ *surface for Definition 628.1 consists exactly of carrier admission for T, precision-request exposure through e, modulus-threshold exposure through m, selected* $\mathsf{StreamName}^{\uparrow}$ *tail-window cover exposure through w, finite tail-witness exposure through t, dyadic tolerance-ledger exactness through d,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through r,* $\mathsf{Real}^{\uparrow}$ *seal handoff through s, componentwise* hsame *transport through h, displayed* Cont *route replay through c,* Pkg *provenance through p, and local naming through n.*

Proof. Carrier admission is projection from the displayed packet T. The rows $e, m, w, t, d, r, s, h, c, p, n$ list the precision request, modulus threshold, selected window cover, finite tail witness, dyadic ledger, regular-sequence handoff, real seal, transports, continuations, provenance, and local name.

The cover obligations are finite. A consumer first reads e, then reads the threshold row m, inspects only the tail windows w selected by that threshold, and then reads the witness row t together with the dyadic ledger d and regular-sequence handoff r.

The seal row s is exported only after this displayed route has been traversed. The rows h, c, p, n account for transport, route replay, provenance, and local naming, so the local NameCert surface is exhausted by the packet. □

Theorem 628.3 (Modulus-indexed tail-window coverage)**.** *Every tail-window observation exported from an accepted* $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ *carrier factors through the finite route*

$$e, m, w, t, d, r.$$

Thus the observation is controlled by a unary precision request, a displayed modulus threshold, the selected $\mathsf{StreamName}^{\uparrow}$ *window cover, the finite tail witness, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Proof. Project the carrier rows e, m, w, t, d, r. The request e determines the modulus read m, the modulus row selects the window cover w, and the cover row exposes only the finite witness t.

Apply the local NameCert obligations of Definition 628.2. They force every observation-facing read through the precision request, modulus threshold, selected windows, finite witness, dyadic ledger, and regular-sequence handoff before any seal-facing read is available.

Since the carrier has no coordinate for an unbounded tail family, quotient stream, host real equality, selected limit, or ambient completeness theorem, an exported tail-window observation can only use the displayed finite coverage route. □

Theorem 628.4 (Modulus-indexed tail-cover Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion or diagonal-limit consumer of an accepted* $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ *carrier receives the seal row s only through the displayed route*

$$e, m, w, t, d, r, h, c, p, n.$$

The handoff therefore consumes the unary precision request, modulus threshold, selected tail windows, finite tail witness, dyadic tolerance ledger, regular-sequence readback, componentwise transport, route replay, package provenance, and local naming before the $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Start with the seal coordinate s. It is the only $\mathsf{Real}^{\uparrow}$ row in the carrier and is listed after the precision, modulus, window-cover, witness, dyadic-ledger, and regular-sequence rows.

Use Definition 628.3. It shows that the tail packet presented to a real-facing consumer has already been confined to the finite windows selected by the modulus row for the request e.

The rows h and c replay the displayed transports and continuations, while p and n provide provenance and local naming. Therefore the seal is reached through the displayed route rather than through hidden completeness, quotient-equality, or selected-limit coordinates. □

Theorem 628.5 (Modulus-indexed tail-cover non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ *packet factors through the finite rows*

$$e, m, w, t, d, r, s, h, c, p, n.$$

The consumer may read the precision request, modulus threshold, selected $\mathsf{StreamName}^{\uparrow}$ *tail-window cover, finite tail witness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, and it cannot read an unbounded tail family, quotient streams, host real equality, selected limits, ambient completeness, or open-cover compactness.*

Proof. Project the accepted carrier and the local NameCert surface. The rows $e, m, w, t, d, r, s, h, c, p, n$ are the complete displayed packet.

Use the window-coverage theorem for observation-facing consumers. It confines every modulus-indexed tail read to the finite threshold, selected windows, finite witness, dyadic tolerance ledger, and regular rational handoff.

Use the Real seal handoff theorem for seal-facing consumers. It routes every completion or diagonal-limit read through $e, m, w, t, d, r, h, c, p, n$ before s is consumed.

No remaining carrier coordinate exists beyond those rows. Hence no downstream consumer can extract an unbounded tail family, quotient stream, host real equality, selected limit, ambient completeness theorem, or open-cover compactness from the packet. □

Theorem 628.6 (Modulus-indexed tail-cover tail totality)**.** *Every accepted finite modulus request in a* $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ *carrier exposes a tail-window route through the displayed unary request row* e*, the modulus row* m*, the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* w*, the finite witness row* t*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* d*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r*, the* $\mathsf{Real}^{\uparrow}$ *seal handoff* s*, componentwise transports* h*, displayed* Cont *routes* c*,* Pkg *provenance* p*, and the local* $\mathsf{NameCert}$ *row* n*.*

Thus tail coverage is total for the finite request presented to the packet, and the totality is witnessed by the same non-choice route that reaches the $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the carrier of Definition 628.1. It displays the precision request, modulus row, selected windows, finite witness, dyadic ledger, regular rational readback, real seal, transports, routes, provenance, and local name.

Apply the window coverage theorem. It routes every tail-window observation through e, m, w, t, d, r, so the accepted finite request has a displayed cover before any seal-facing row is read.

Apply the Real seal handoff theorem to extend that route through h, c, p, n and then to the seal row s. The handoff does not add a second carrier or an unbounded family; it only replays the finite rows already named by the packet.

The non-escape theorem closes the boundary. Since no carrier coordinate supplies quotient streams, host real equality, selected limits, ambient completeness, open-cover compactness, or an unbounded tail family, the finite tail totality is exactly the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusIndexedTailCover}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusIndexedTailCover}^{\uparrow}$ packet is generated from a unary precision request, a Cauchy modulus threshold, selected StreamName tail windows, a finite tail witness, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, a RealUp seal row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusIndexedTailCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 628.1, Definition 628.2, Definition 628.3, Definition 628.4, and Definition 628.5 bind the finite modulus-indexed route from precision request through tail-window cover to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusIndexedTailCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, host real equality, selected limits, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window coverage, RealUp seal handoff, and non-escape for the same modulus-indexed packet.

Chapter 629

A Concrete Naming Certificate for $\mathsf{NegativeNameBoundary}^{\uparrow}$

The $\mathsf{NegativeNameBoundary}^{\uparrow}$ packet realizes apophatic negative naming as a finite NameCert surface. It records the name that marks a requested external supply, the socket shape being refused, the refusal ledger that keeps the supply outside generated content, the audit gate that makes the boundary readable, the componentwise transports, the displayed continuation routes, the package provenance, and the local naming row. The apophatic handle row is consumed from chapter 543, while $\mathsf{NegativeNameBoundary}^{\uparrow}$ adds the socket, refusal-ledger, and audit-gate rows that keep the requested supply outside generated content. The packet is a boundary certificate for named refusal, not a source of the requested supply.

The negative-name packet is the refusal-facing boundary between the socket-name surface of chapter 543 and the visible refusal question of chapter 715. The certificate-side sibling is chapter 630; this chapter controls the name boundary by keeping the requested socket visible while refusing to turn it into supplied content.

Definition 629.1 (NegativeNameBoundary carrier)**.** *A* $\mathsf{NegativeNameBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (a, s, r, g, h, c, p, n)$$

with the following displayed coordinates:

1. *a is the apophatic handle row naming the requested external supply without describing it as an internal object;*
2. *s is the requested socket shape, recording which supply interface the surrounding argument asked to consume;*
3. *r is the refusal and non-internalization ledger, recording each blocked route by which the requested supply might otherwise be treated as generated theorem content, quotient equality, host selector, axiom, or hidden witness;*
4. *g is the audit gate row, recording the finite question that separates visible boundary use from hidden internalization;*
5. *h records componentwise* hsame *transports for the handle, socket, refusal, gate, route, provenance, and local naming rows;*
6. *c records the displayed* Cont *routes by which consumers inspect the handle, requested socket shape, refusal ledger, audit gate, provenance, and local certificate;*

7. *p records* Pkg *provenance over* BHist*, apophatic names, socket rows, refusal ledgers, audit gates,* hsame*,* Cont*, and* NameCert*;*
8. *n is the local* $\mathsf{NameCert}_{\mathsf{NegativeNameBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for the requested external supply itself.

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundaryTasteGate single carrier alignment`

Theorem 629.2 (NegativeNameBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{NegativeNameBoundary}^{\uparrow}}$ *surface for Definition 629.1 consists exactly of carrier admission for*

$$B = (a, s, r, g, h, c, p, n),$$

apophatic handle exposure through a, requested socket display through s, refusal-ledger exhaustion through r, audit-gate readback through g, componentwise hsame *stability through h, displayed* Cont *replay through c,* Pkg *provenance through p, and local naming through n.*

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundaryNameCert obligations`

Proof. Carrier admission is projection from the displayed tuple. The rows a, s, r, g expose the boundary content: the name, the requested socket shape, the refusal ledger, and the audit gate.

Transport and consumer access are confined by h and c. These rows replay the same boundary packet under componentwise hsame movement and expose only the recorded continuation routes.

The provenance and local naming rows p, n account for package ancestry and the local certificate. Since the requested external supply has no coordinate in B, the NameCert surface can name the negative boundary without exporting the supply as internal content. □

Theorem 629.3 (NegativeNameBoundary refusal ledger)**.** *For an accepted* $\mathsf{NegativeNameBoundary}^{\uparrow}$ *carrier, the refusal ledger r records every public attempt to use the requested socket shape s as generated theorem content, quotient equality, host selector, axiom, hidden witness, or anonymous supply route. Each such attempt is routed through r and remains audit-visible through g, c, p, n.*

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundary refusal ledger`

Proof. Project the requested socket row s and the refusal ledger r. The ledger is attached to exactly the routes by which the requested supply would be converted into internal content.

The audit gate row g makes the refusal question public, while the continuation routes c replay that gate and ledger for downstream consumers. Componentwise transport in h preserves the same refusal rows when the packet is moved across equivalent presentations.

The provenance and local naming rows p, n identify the packet as the certified source of the refusal. Therefore a public use of the requested socket shape must either pass through the displayed refusal ledger or have no accepted coordinate in the packet. □

Theorem 629.4 (NegativeNameBoundary non-internalization)**.** *Every public use of an accepted* $\mathsf{NegativeNameBoundary}^{\uparrow}$ *packet factors through the rows*

$$a, s, r, g, h, c, p, n.$$

The packet may expose the apophatic handle, the requested socket shape, the refusal ledger, the audit gate, transports, continuation routes, provenance, and local naming, but it cannot export the requested external supply as generated theorem content, quotient equality, host selector, axiom, hidden witness, or unrecorded internal object.

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundary non internalization`

Proof. Begin with the carrier inventory of Definition 629.1. The requested external supply itself is absent; only its handle, socket shape, refusal ledger, audit gate, transport, route, provenance, and local naming rows are present.

Let a public consumer of the packet be given. By the NameCert obligations of Definition 629.2, every displayed consumer route must read the packet through c and preserve the same finite surface through h.

If the consumer tries to use the socket shape as generated theorem content, quotient equality, host selector, axiom, hidden witness, or anonymous supply, apply the refusal ledger of Definition 629.3. The attempt is routed to r and remains visible through the audit gate.

Thus every accepted public use factors through the displayed rows, and no projection from the packet can return the requested external supply as internal substrate content. □

Theorem 629.5 (NegativeNameBoundary quotient-soundness lattice link)**.** *Every* QuotientSoundnessBoundary$^{\uparrow}$ *representative-selection refusal first reads the* NegativeNameBoundary$^{\uparrow}$ *packet through its apophatic handle, requested socket shape, refusal ledger, audit gate, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. The quotient-soundness boundary of chapter 634 then specializes that refusal by replacing the blocked quotient request with local* psame *transport rather than a selected representative or a* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundary quotient soundness lattice link`

Proof. Start with the carrier of Definition 629.1. Its handle, socket, refusal ledger, audit gate, transport, continuation, provenance, and local naming rows give the visible boundary packet through which a representative-selection request must be inspected.

Apply the refusal ledger theorem. A request to treat the socket as quotient equality, a host selector, an axiom, or a hidden witness is routed through r and remains audit-visible through g, c, p, n.

Use non-internalization to remove the forbidden exits. Since the requested external supply is absent from the packet, the sibling quotient-soundness chapter can only specialize the visible refusal by routing surviving operations through its local relation and psame transport rows.

Thus the dependency on chapter 634 is a lattice dependency between concrete refusal packets: the negative-name packet supplies the handle, socket, refusal, and gate discipline, and the quotient-soundness packet supplies the classifier-respecting transport that replaces representative selection. □

Theorem 629.6 (NegativeNameBoundary obligation closure)**.** *For an accepted finite* NegativeNameBoundary$^{\uparrow}$ *packet* $B = (a, s, r, g, h, c, p, n)$*, the carrier, refusal ledger, non-internalization row, and quotient-soundness lattice link are the complete obligation surface. The packet records the apophatic handle, requested socket, refusal gate, transport, continuation, provenance, and local name, while excluding host equality,* Quot.sound*, propositional extensionality, and hidden negative-name witnesses.*

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundary obligation closure`

Proof. Start with Definition 629.1. The requested external supply is absent from the packet, while a, s, r, g, h, c, p, n are the displayed finite rows available to the certificate.

Apply the refusal ledger and non-internalization rows, Definition 629.3 and Definition 629.4. They send attempted uses of the requested socket to the visible refusal ledger and prevent the requested supply from becoming generated theorem content.

Finally use the quotient-soundness lattice link Definition 629.5. It routes representative-selection pressure to local transport rather than to host equality, Quot.sound, propositional extensionality, or hidden negative-name witnesses. These rows exhaust the obligation surface. □

Theorem 629.7 (NegativeNameBoundary public export)**.** *Every public consumer of an accepted* NegativeNameBoundary$^{\uparrow}$ *packet must read the apophatic handle a, requested socket s, refusal ledger r, audit gate g, componentwise* hsame *transports h, displayed* Cont *routes c,* Pkg *provenance p, and local* NameCert *row n. The only allowed downstream specialization is the quotient-soundness lattice link through local* psame *transport; no public export may import external supply,* Quot.sound*, host equality, a selector, an axiom, or hidden witness data.*

Proof. Start with the carrier and certificate rows of Definition 629.1 and Definition 629.2. They display a, s, r, g, h, c, p, n as the complete public packet surface.

Use the refusal ledger and non-internalization boundary, Definition 629.3 and Definition 629.4. These rows send every attempted socket use through the visible refusal route and keep the requested external supply outside generated content.

The downstream specialization is exactly the one supplied by Definition 629.5. Combining it with Definition 629.6 leaves only local psame transport as the quotient-soundness route, with no public export of host equality, Quot.sound, a selector, an axiom, or hidden witness data. □

Lean4 ✓ `BEDC.Derived.NegativeNameBoundaryUp.NegativeNameBoundary public export`

Closure status for NameCert$_{\text{NegativeNameBoundary}^{\uparrow}}$.

Constructive story: A NegativeNameBoundary$^{\uparrow}$ packet is generated from an apophatic handle row, requested socket-shape row, refusal and non-internalization ledger, audit gate row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NegativeNameBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 629.1, Definition 629.2, Definition 629.3, Definition 629.4, Definition 629.5, Definition 629.6, and Definition 629.7 bind the finite carrier, local naming obligations, refusal ledger, non-internalization boundary, quotient-soundness lattice link, complete obligation surface, and public export route for apophatic negative naming.

Formal verification: FormalStatus(NameCert$_{\text{NegativeNameBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close the requested external supply, positive characterization of the named source, Lean verification, bridge checking, quotient representatives, host selectors, axioms, or hidden witnesses.

Upgrade path: No further paper-side theory closure is pending; Lean verification and bridge checking remain on the independent verification axis.

Chapter 630

A Concrete Naming Certificate for $\mathsf{NegativeCertificateBoundary}^{\uparrow}$

The $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ packet realizes apophatic refusal as a finite NameCert surface. It names the socket row at which an external supply is requested, records the attempted internalizer that must be refused, keeps the refusal in a GAP ledger, exposes the result through an audit-gate readback, and carries the transport, continuation, provenance, and local naming rows needed by downstream checks.

The packet is a boundary certificate. It does not prove that the external supply is absent from every possible context, and it does not turn absence into a positive object. It records that the named socket has no admitted internalizer inside the displayed BEDC packet.

$\mathsf{NegativeCertificateBoundary}^{\uparrow}$ specializes the apophatic-name surface of chapter 543 by adding the refused-internalizer row, the GAP ledger row, and the audit-gate readback row. $\mathsf{ApophaticName}^{\uparrow}$ supplies the named externality and refusal discipline; $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ packages the concrete socket, refusal, ledger, and audit surface that downstream checks can read without converting the refused content into substrate data.

The socket row is read through chapter 536, the audit readback row is consumed through chapter 526, and the refusal boundary remains distinct from the negative-name surface of chapter 629: this packet certifies a concrete refused internalizer at a named socket rather than renaming the whole negative-name boundary.

Definition 630.1 (NegativeCertificateBoundary carrier)**.** *A* $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, G, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the named socket row, recording the apophatic boundary at which an external supply is requested without being internalized;*
2. *R is the refused internalizer row, recording the proposed constructor, theorem, axiom, quotient representative, host datum, or setup object whose admission would collapse the boundary;*
3. *G is the* GAP *ledger row, recording the finite refusal entry for R at socket S;*
4. *A is the audit-gate readback row, exposing the socket, refused internalizer, and* GAP *decision to downstream checks;*

5. *H records componentwise* hsame *transport for the socket, refusal,* GAP *ledger, audit readback, provenance, and local certificate rows;*
6. *C records the displayed* Cont *routes from the socket row to the refusal row, from the refusal row to the* GAP *ledger, and from the ledger to the audit readback;*
7. *P records* Pkg *provenance over* BHist, GAP, hsame, Cont, Pkg, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{NegativeCertificateBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a positive witness of the requested external supply and no route that converts the refusal into an internal object.

Theorem 630.2 (NegativeCertificateBoundary NameCert obligations)**.** *For every accepted* NegativeCertificateBour *carrier*

$$B = (S, R, G, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NegativeCertificateBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission: the accepted object is exactly the displayed* BHist *packet B with socket, refused-internalizer,* GAP *ledger, audit readback, transport, continuation, provenance, and local naming rows;*
2. *refusal classifier stability: any classified replacement of S, R, G, or A is transported componentwise through H and replayed only through the displayed routes in C;*
3. GAP *ledger exactness: the refusal recorded in G is attached to the named socket S and to the proposed internalizer R, with no anonymous refusal source and no unrecorded acceptance route;*
4. *audit-gate handoff: every downstream read of the boundary factors through A after reading the socket, refused internalizer, and* GAP *ledger rows;*
5. *non-internalization: the packet exports the refusal boundary and its audit readback, but it exports no constructor, theorem, axiom, quotient representative, host datum, setup object, or positive supply witness for the requested externality.*

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 630.1. The tuple exposes exactly S, R, G, A, H, C, P, N, so every admitted boundary row is part of the finite packet.

Classifier stability is componentwise. The transport row H moves the socket, refusal, ledger, and readback together, and the continuation row C replays only the displayed route from socket to refusal to ledger to audit readback.

Ledger exactness follows from the role of G. It records the refusal of the proposed internalizer R at the named socket S, and the audit readback A is the only downstream surface that exposes that decision.

Finally, non-internalization is a consequence of the carrier inventory. No coordinate of B supplies the requested external content or admits the refused internalizer, so the NameCert packet certifies the boundary rather than converting it into substrate content. □

Theorem 630.3 (NegativeCertificateBoundary audit surface)**.** *Every consumer-facing read of an accepted* $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ *packet factors through the finite rows*

$$S, R, G, A, H, C, P, N.$$

The audit readback exposes the named socket, the refused internalizer, the GAP *ledger entry, componentwise transport, continuation replay, provenance, and local naming certificate. It exposes no separate positive object for the requested external supply.*

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier audit surface`

Proof. Start with the carrier definition. The only readback row available to a downstream consumer is A, and A is attached to the socket row S, refused internalizer row R, and GAP ledger row G.

Use the continuation routes in C to order the read. A consumer first reaches the socket, then the refusal row, then the ledger, and only then the audit readback.

The transport and provenance rows preserve the same surface. Since the tuple has no coordinate for a positive supply object, the audit surface is exhausted by the displayed rows. □

Theorem 630.4 (NegativeCertificateBoundary refusal classifier stability)**.** *Let*

$$B = (S, R, G, A, H, C, P, N)$$

be an accepted NegativeCertificateBoundary$^{\uparrow}$ *carrier. Any componentwise* hsame *transport of* S, R, G, A, H, C, P, N *preserves the refusal classifier determined by the socket row* S *and the refused internalizer row* R*. In particular, transport cannot turn* R *into an admitted internalizer for the requested external supply.*

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier refusal classifier stability`

Proof. Project the carrier B. The transport row H moves the socket, refused internalizer, ledger, audit readback, provenance, and local certificate rows componentwise, so the transported packet still has the same displayed coordinates.

Use the continuation row C to replay only the displayed route from S to R, from R to G, and from G to A. The refusal classifier is therefore read through the same finite chain after transport.

The NameCert obligations in Definition 630.2 include refusal classifier stability and non-internalization. Since no coordinate of B exports a positive supply witness or an admitted internalizer, componentwise transport preserves the refusal boundary rather than converting R into substrate content. □

Theorem 630.5 (NegativeCertificateBoundary GAP ledger exactness)**.** *For every accepted* NegativeCertificateBour *carrier*

$$B = (S, R, G, A, H, C, P, N),$$

the GAP *row* G *is exact for the pair* (S, R)*: every accepted refusal read passes through* G*, every entry recorded by* G *is attached to the displayed socket* S *and refused internalizer* R*, and the packet contains no anonymous refusal source and no acceptance route bypassing the ledger.*

Proof. The carrier definition identifies G as the unique displayed GAP ledger row for the pair consisting of the socket S and the refused internalizer R. Thus coverage is finite and named: an accepted refusal read has a displayed ledger coordinate.

Soundness follows from the NameCert obligations. The ledger exactness row ties the entry in G to the socket S and the proposed internalizer R, and the continuation row C gives the route from socket to refusal to ledger.

Completeness is the audit consequence. By Definition 630.3, downstream readback factors through A after S, R, G and exposes no positive object for the requested external supply. Hence the carrier leaves no anonymous refusal and no unrecorded acceptance route outside the displayed ledger. □

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier gap ledger exactness`

Theorem 630.6 (NegativeCertificateBoundary audit-gate handoff)**.** *Every downstream audit read of an accepted* NegativeCertificateBoundary$^{\uparrow}$ *carrier factors through the audit-gate row A after reading S, R, G. The handoff exports the socket, refused internalizer, and ledger decision, but exports no constructor, theorem, axiom, quotient representative, host datum, setup object, or positive supply witness for the requested externality.*

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier audit gate handoff`

Proof. Apply the audit surface theorem to the accepted carrier. The only consumer-facing readback row is A, and the continuation route reaches it only after the socket row S, the refused internalizer row R, and the ledger row G.

The apophatic sibling route in Definition 630.7 fixes the boundary as a named socket and refused-internalizer route rather than as an isolated positive object. This keeps the downstream read attached to the sibling apophatic surface.

Finally, the carrier inventory contains no coordinate for a constructor, theorem, axiom, quotient representative, host datum, setup object, or positive supply witness. The audit handoff is therefore a boundary readback, not an admission route. □

Theorem 630.7 (NegativeCertificateBoundary apophatic sibling route)**.** *Every accepted* NegativeCertificateBounda *packet reads its socket row through chapter 543 and chapter 536, and reads its refused-internalizer and audit-gate route through chapter 715 before exposing the local* GAP *ledger.*

Proof. Project the accepted carrier as

$$B = (S, R, G, A, H, C, P, N).$$

The row S is the named apophatic socket: it is the place where the requested external supply is named without being admitted as substrate content.

The rows R, G, A give the refusal and audit route. The refused internalizer R is attached to the socket, the ledger row G records the finite GAP entry for that refusal, and the audit row A is the downstream readback surface.

Thus the packet first inherits the apophatic-name and apophatic-socket sibling route, then passes through the apophatic-gate question route for the refused internalizer and audit readback. The exposed boundary is therefore a downstream finite refusal boundary, not an isolated kernel-only object. □

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.NegativeCertificateBoundaryCarrier apophatic sibling route`

Closure status for NameCert$_{\text{NegativeCertificateBoundary}^{\uparrow}}$.

Constructive story: A NegativeCertificateBoundary$^{\uparrow}$ packet is generated from a named socket row, a refused internalizer row, a GAP ledger row, an audit-gate readback row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NegativeCertificateBoundary}\uparrow}$, maturePackageClosed)

Scope closed: Definition 630.1, Definition 630.2, Definition 630.4, Definition 630.5, Definition 630.6, and Definition 630.7 bind the finite BHist carrier, refusal classifier stability under hsame, GAP ledger exactness through Cont, audit-gate handoff through Pkg and NameCert, non-internalization of the refused supply, and the sibling route through the apophatic-name, apophatic-socket, apophatic-gate-question, and audit-gate-boundary chapters.

Formal verification: FormalStatus(NameCert$_{\text{NegativeCertificateBoundary}\uparrow}$, unformalized)

Lean4 ✓ `BEDC.Derived.NegativeCertificateBoundaryUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, checked bridge status, global absence of the external supply, or a positive object representing the refused content.

Upgrade path: Verification closure requires Lean implementation and checked bridge evidence for the finite refusal-boundary NameCert surface.

Chapter 631

A Concrete Naming Certificate for $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$

The $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ packet names the finite Real-and-Completion handoff where a product of regular Cauchy names is read as a regular Cauchy limit classifier. It consumes the product-observation surface of $\mathsf{CauchyProduct}^{\uparrow}$ from chapter 464 and the classifier handoff of $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ from chapter 443, while keeping every consumer route inside displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. The packet does not supply ambient completeness, quotient stream equality, a selected host limit, or host real multiplication.

Definition 631.1 (Product regular-Cauchy limit carrier)**.** *A* $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (P, Q, W, R, D, E, H, C, L, N)$$

with the following displayed rows:

1. *P is the accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *product packet for the two regular Cauchy source names;*
2. *Q is the accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *packet that classifies the diagonal limit handoff for the product read;*
3. *W is the joint* $\mathsf{StreamName}^{\uparrow}$ *window ledger, combining the paired product windows with the selected diagonal-limit windows;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the product-radius ledger and then consumed by the limit classifier;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, comparison, and precision ledger shared by the product observation and the limit handoff;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row through which a completion consumer sees the product-limit candidate;*
7. *H stores componentwise* hsame *transports for P, Q, W, R, D, E, L, N and the local packet shape;*
8. *C stores finite* Cont *routes for every public read of the product row, diagonal-window classifier, dyadic ledger, regular readback, Real seal, package provenance, and local naming row;*
9. *L is the* Pkg *provenance over* $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$ *row.*

The carrier contains no host multiplication primitive, no ambient completeness theorem, no quotient stream representative, and no independent selected limit.

Theorem 631.2 (Product regular-Cauchy limit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$ *surface for Definition 631.1 consists of six finite obligation rows:*

1. *carrier admission for the packet* $U = (P, Q, W, R, D, E, H, C, L, N)$*;*
2. *product-source admission stating that the only product observation is the accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *row* P*;*
3. *limit-classifier admission stating that the only completion-facing limit handoff is the accepted* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *row* Q*;*
4. *shared-window exactness stating that* W, R, D, E *are read in the order product windows, dyadic product observation, regular readback, diagonal-limit windows, and* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *componentwise stability stating that* H *and* C *preserve the same product row, classifier row, shared ledger, package provenance, and local naming row under* hsame *and* Cont *transport;*
6. *ledger exactness stating that every public read factors through* $P, Q, W, R, D, E, H, C, L, N$ *and no ambient completeness or host real multiplication is exported.*

Lean4 ✓ `BEDC.Derived.ProductRegularCauchyLimitUp.ProductRegularCauchyLimitTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 631.1. The tuple lists the product packet, limit classifier, windows, regular readback, dyadic ledger, Real seal, transports, routes, package provenance, and local naming row.

Product-source admission uses the $\mathsf{CauchyProduct}^{\uparrow}$ row P. A consumer cannot read a product observation except through the paired product windows and dyadic product ledger already displayed by that packet.

Limit-classifier admission uses the $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ row Q. Its diagonal-window classifier consumes the regular readback and Real seal data as displayed rows, not as a host-level selected limit.

The shared-window and stability clauses are the roles of W, R, D, E, H, C. They keep the product observation, regular readback, diagonal limit window, and seal handoff in one finite route under componentwise transport.

Finally, L and N repack those rows as the local certificate surface. Since the carrier has no coordinate for host multiplication, quotient stream equality, ambient completeness, or an independent selected limit, the public ledger exports only the displayed product-limit packet. □

Theorem 631.3 (Product regular-Cauchy limit product-source admission)**.** *In a* $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ *carrier* $U = (P, Q, W, R, D, E, H, C, L, N)$*, the row* P *is the only product-observation source. Every public product read of* U *is consumed through the* $\mathsf{CauchyProduct}^{\uparrow}$ *sibling surface of chapter 464, specifically through the paired-window source admission of Definition 464.11 and the budget-route exactness of Definition 464.12.*

Proof. Project the carrier of Definition 631.1. The only coordinate named as a product observation is P, and it is explicitly an accepted $\mathsf{CauchyProduct}^{\uparrow}$ packet.

The sibling certificate supplies the product read. By Definition 464.11, the source windows for the product read are the paired windows inside the $\mathsf{CauchyProduct}^{\uparrow}$ packet. By Definition 464.12,

the dyadic product observation and regular readback route are consumed through that same sibling budget surface.

Since U contains no second product-observation coordinate, every public product read factors through P and its displayed $\mathsf{CauchyProduct}^{\uparrow}$ surface. □

Theorem 631.4 (Product regular-Cauchy limit classifier admission)**.** *In a* $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ *carrier* $U = (P, Q, W, R, D, E, H, C, L, N)$*, the row* Q *is the only completion-facing diagonal-limit classifier. A consumer reads the product-limit handoff through the* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *sibling surface of chapter 443, not through an ambient selected limit.*

Proof. Inspect the displayed rows of the carrier. The only coordinate whose role is a completion-facing diagonal-limit handoff is Q, and its declared source is the accepted $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ packet.

The classifier is therefore read through the sibling chapter named in the statement. The surrounding rows R, D, E are inputs and seals for that classifier route; they do not create an independent completion theorem or selected host limit.

Thus every completion-facing product-limit read uses Q as its classifier row and remains inside the finite diagonal-window handoff. □

Theorem 631.5 (Product regular-Cauchy limit shared-window exactness)**.** *For a* $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ *carrier* $U = (P, Q, W, R, D, E, H, C, L, N)$*, the shared route through* W, R, D, E *is ordered through the displayed sibling surfaces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The relevant dependency chapters are chapter 126, chapter 291, chapter 301, and chapter 38.

Proof. Begin with the shared window row. The carrier states that W is the joint $\mathsf{StreamName}^{\uparrow}$ window ledger combining the product windows with the selected diagonal-limit windows, so the first read is the StreamName sibling surface.

Next the dyadic and regular readback rows are fixed. The row D supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, radius, comparison, and precision ledger, while R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback row produced by the product-radius ledger and consumed by the limit classifier.

Finally the row E is the $\mathsf{Real}^{\uparrow}$ seal through which the completion consumer sees the product-limit candidate. The route therefore has the displayed order W, D, R, E across the four sibling surfaces, with no extra route outside those finite rows. □

Theorem 631.6 (Product regular-Cauchy limit transport stability)**.** *Componentwise* hsame *transport and the displayed* Cont *replay of a* $\mathsf{ProductRegularCauchyLimit}^{\uparrow}$ *carrier preserve the same product row* P*, classifier row* Q*, shared rows* W, R, D, E*, provenance row* L*, and local naming row* N*. The transported carrier has the same finite public rows and the same product-limit read surface.*

Proof. The row H stores componentwise hsame transports for P, Q, W, R, D, E, L, N and for the local packet shape. Hence transport cannot replace the product source, classifier source, shared ledger, provenance, or name row with a row outside the displayed tuple.

The row C stores the finite Cont routes for the public reads of the product row, diagonal-window classifier, dyadic ledger, regular readback, Real seal, package provenance, and local naming row. Replaying these routes after componentwise transport therefore reads the same named coordinates in the transported packet.

Thus the transported packet preserves the public product-limit surface exactly at the level of displayed rows. □

Theorem 631.7 (Product regular-Cauchy limit ledger exactness)**.** *For a* ProductRegularCauchyLimit$^{\uparrow}$ *carrier* $U = (P, Q, W, R, D, E, H, C, L, N)$*, the ledger row* L *and local naming row* N *exhaust the public reads. The packet exports no host multiplication primitive, no quotient stream equality, no selected limit, and no ambient completeness theorem.*

Proof. The provenance row L names exactly the dependency surfaces listed in the carrier: CauchyProduct$^{\uparrow}$, RegularCauchyLimitClassifier$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, and NameCert.

The local row N is the $\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$ row for this finite tuple. Together with L, it gives the public naming ledger for every accepted read of the packet.

The carrier has no coordinate for host multiplication, quotient streams, selected limits, or ambient completeness. Any read requiring one of those objects has no source row in U, so the public surface is exhausted by $P, Q, W, R, D, E, H, C, L, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$**.**

Constructive story: A ProductRegularCauchyLimit$^{\uparrow}$ packet is generated from a CauchyProduct$^{\uparrow}$ product row, a RegularCauchyLimitClassifier$^{\uparrow}$ diagonal-window classifier, shared StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ ledger, Real$^{\uparrow}$ seal, transport routes, package provenance, and a local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$, obligation)

Scope closed: Definition 631.1, Definition 631.2, Definition 631.3, Definition 631.4, Definition 631.5, Definition 631.6, and Definition 631.7 bind the product-to-limit handoff as a finite Real-and-Completion NameCert packet.

Lean4 ✓ `BEDC.Derived.ProductRegularCauchyLimitUp.taste gate`

Lean target: `BEDC.Derived.ProductRegularCauchyLimitUp.ProductRegularCauchyLimitTasteGate\_single\_carrier\_alignm`

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProductRegularCauchyLimit}^{\uparrow}}$, axiomClean)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, selected host limits, host real multiplication, or a public standard bridge.

Upgrade path: Public closure requires a single public export assembling the product-source admission, limit-classifier admission, shared-window exactness, transport stability, and ledger exactness rows.

Chapter 632

Diagonal Cofinal Tail Name Certificate Construction

The $\mathsf{DiagonalCofinalTail}^{\uparrow}$ packet realizes the cofinal-tail reading of the transported stationary diagonal window as a finite $\mathsf{NameCert}$ surface. It keeps the rational seed classifier, the constant stream windows, the regular-rational tail readbacks, the dyadic finite-window ledgers, the real-facing seal row, transport rows, continuation routes, provenance, and local naming in one displayed packet. It does not export an external equality identification, a selected limit, an unrecorded window selector, or an ambient completeness principle.

Definition 632.1 (Diagonal cofinal-tail carrier)**.** *A* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, S, G, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *Q is the paired stationary* $\mathsf{Rat}^{\uparrow}$ *seed row, including the two rational history carriers and their* $\mathsf{RatHistoryClassifier}$ *row;*
2. *S is the paired constant* $\mathsf{StreamName}^{\uparrow}$ *window row induced by Q;*
3. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row for the selected diagonal window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger for the same selected tail window;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *diagonal representative row read only through S, G, D;*
6. *W records the transported stationary window seal whose RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp readings are the four public readings of Definition 9238.18;*
7. *H records componentwise* hsame *transports for Q, S, G, D, R, W and the structural rows;*
8. *C records the displayed* Cont *routes from Q to S, from S to G, from G to D, from D to R, and from W to each public window reading;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}$ *row.*

The carrier is finite. It has no coordinate for an infinite window family, external equality identification, selected real limit, unrecorded selector, or ambient completeness theorem.

Theorem 632.2 (Diagonal cofinal-tail NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}$ *surface for Definition 632.1 consists exactly of carrier admission for* $T = (Q, S, G, D, R, W, H, C, P, N)$*, stationary* $\mathsf{Rat}^{\uparrow}$ *seed exposure through* Q*, constant* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback exposure through* G*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger exposure through* D*,* $\mathsf{Real}^{\uparrow}$ *diagonal representative exposure through* R*, the transported stationary seal row* W*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier namecert obligations`

Proof. Carrier admission is projection from the displayed packet T. The rows Q, S, G, D, R expose the rational seeds, constant stream windows, regular-rational readbacks, dyadic window ledgers, and real-facing representatives.

The seal row W is read through the finite-window seal exhaustion theorem in Definition 9238.18. Its public readings are already the RatUp seed classifier, the StreamNameUp constant-window classifier, the RegSeqRatUp finite-window seal consumption row, the DyadicRatCoreUp observation ledger, and the RealUp finite-window classifier.

The rows H, C, P, N account for transport, route replay, package provenance, and local naming. Since the excluded analytic objects have no coordinate in T, the listed rows exhaust the local NameCert obligations. □

Theorem 632.3 (Diagonal cofinal-tail seed-window readback)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

the stationary $\mathsf{Rat}^{\uparrow}$ *seed row* Q*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row* G*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *diagonal row* R*, the transported seal row* W*, and the structural rows* H, C, P, N *form one consumer-readable route. The route starts at the paired RatUp seed classifier, reads the constant window, consumes the regular-rational cofinal tail, records the dyadic finite-window ledger, and reaches the real-facing row only through the transported stationary seal.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier seed window readback`

Proof. Project the accepted carrier of Definition 632.1. The displayed tuple already names the only rows that may be read by a consumer: $Q, S, G, D, R, W, H, C, P, N$.

Use Definition 632.2. It identifies the carrier admission, classifier exposure, componentwise hsame transport, continuation replay, package provenance, and local naming rows.

The transported seal row W is the same one used by Definition 9238.18. That theorem supplies the RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp public readings of the stationary window, so the projected rows assemble a single readback route. □

Theorem 632.4 (Diagonal cofinal-tail common-window refinement)**.** *Let two* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumers read the same accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet through the displayed* hsame *transports and* Cont *routes. If both consumers accept the packet, then their readings refine to the same finite cofinal-tail window rows*

$$Q, S, G, D, R, W, H, C, P, N.$$

Neither consumer can refine through a second selector, a quotient-stream equality datum, a selected limit, or an ambient completeness witness.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier common window refinement`

Proof. Begin with Definition 632.3. Both consumers start from the same accepted carrier and therefore read the same projected seed-window route $Q, S, G, D, R, W, H, C, P, N$.

Apply Definition 632.11. It forces every visible cofinal-tail selector for a RegSeqRatUp or RealUp consumer to be determined by the displayed stationary seed row, constant window row, tail readback row, dyadic ledger, transported seal, and structural routes.

Finally use Definition 632.13. Since no route leaves the finite packet, the two accepted readings have no alternate object through which to diverge. They refine to the same displayed finite cofinal-tail window rows. □

Theorem 632.5 (Diagonal cofinal-tail obligation closure package)**.** *The carrier of Definition 632.1, the seed-window readback of Definition 632.3, the common-window refinement of Definition 632.4, the Real seal handoff of Definition 632.12, the non-escape boundary of Definition 632.13, and the TasteGate single-carrier theorem Definition 632.14 constitute the obligation surface for* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *over* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier obligation closure package`

Proof. The carrier definition gives the finite tuple $T = (Q, S, G, D, R, W, H, C, P, N)$ and excludes every unlisted analytic or host-side coordinate.

The seed-window readback and common-window refinement identify the positive consumer route and show that any two accepted RegSeqRatUp or RealUp reads refine to that same finite window surface.

The Real seal handoff supplies the real-facing passage through the StreamNameUp, RegSeqRatUp, and DyadicRatCoreUp rows. The non-escape theorem prevents an accepted downstream route from leaving the displayed packet.

Finally, Definition 632.14 records that later TasteGate-facing consumers have this single cofinal-tail carrier source. Together these clauses are exactly the named obligation surface. □

Theorem 632.6 (Diagonal cofinal-tail RegSeqRat–StreamName handoff)**.** *Every accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *consumer factors the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback through the constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger, the transported stationary seal row, the* $\mathsf{Real}^{\uparrow}$ *diagonal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, the local* $\mathsf{NameCert}$ *row, and the TasteGate single-carrier surface of Definition 632.14. In the carrier notation*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

the readback route is exhausted by $Q, S, G, D, R, W, H, C, P, N$ *and has no second carrier source, selected limit, quotient-stream equality, external equality, unrecorded selector, hidden parameter, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier regseqrat streamname handoff`

Proof. Project the carrier of Definition 632.1. The accepted packet exposes the RatUp seed row Q, constant StreamNameUp window S, RegSeqRatUp tail row G, DyadicRatCoreUp ledger D, RealUp diagonal row R, transported stationary seal row W, and the structural rows H, C, P, N.

Apply the NameCert obligations together with the seed-window readback. These results identify the RegSeqRatUp readback as the route from the stationary seed through the constant StreamNameUp window and the finite dyadic ledger.

Use the common-window refinement and the obligation closure package. They keep every accepted RegSeqRatUp or RealUp consumer on the same finite cofinal-tail surface and route the transported stationary seal through the displayed H, C, P, N rows.

Finally apply Definition 632.14. The TasteGate-facing consumer sees the same single carrier source, so the handoff cannot use a second carrier, selected limit, quotient-stream equality, external equality, unrecorded selector, hidden parameter, or ambient completeness witness. □

Theorem 632.7 (Diagonal cofinal-tail primitive scope)**.** *The* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *route consumes the preservation surface of chapter 483 only as a finite packet boundary. Its cofinal-tail consumer reads the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows through the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 632.1, and never through an unrecorded preservation theorem or ambient completion principle.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier primitive scope`

Proof. Project the accepted cofinal-tail carrier. The visible rows are the rational seed row Q, stream window row S, regular-rational tail row G, dyadic ledger row D, real-facing row R, transported seal row W, and the structural rows H, C, P, N.

The sibling preservation surface is read only as a boundary for finite packet consumption. No coordinate in T imports a closed subject-reduction theorem, host preservation theorem, or completion principle from that sibling chapter.

Thus every consumer read remains inside the displayed hsame, Cont, Pkg, and NameCert rows of the cofinal-tail packet, while chapter 483 supplies only the named preservation boundary against which the packet is scoped. □

Theorem 632.8 (Diagonal cofinal-tail compatibility handoff)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

any downstream $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-seal consumer that uses the compatibility packet must read the same stationary seed row* Q*,* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback* G*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{Real}^{\uparrow}$ *diagonal row* R*, transported seal row* W*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N *before using that packet. The handoff is therefore the finite compatibility route from Definition 632.6 and Definition 632.7 into the carrier surface of chapter 646, not a host equality, selected limit, quotient-stream equality, or ambient completion principle.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier compatibility handoff`

Proof. Project the accepted cofinal-tail carrier. Its available rows are exactly the stationary seed, constant window, regular-rational tail readback, dyadic ledger, real diagonal row, transported seal row, and the structural rows H, C, P, N.

Apply Definition 632.6. It forces every RegSeqRatUp tail readback through the displayed tuple $Q, S, G, D, R, W, H, C, P, N$, so a downstream compatibility consumer has no independent seed, window, ledger, seal, transport, continuation, provenance, or local-name source.

Use Definition 632.7. The primitive scope permits only finite packet consumption through the displayed hsame, Cont, Pkg, and NameCert rows, and excludes unrecorded preservation theorems and ambient completion principles.

The remaining consumer route is precisely the finite compatibility handoff consumed by chapter 646. Since the route has already passed through the displayed rows of T, the compatibility

packet is received as a local finite handoff rather than as host equality, a selected limit, quotient-stream equality, or a completion theorem. □

Theorem 632.9 (Diagonal cofinal-tail public route)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

every downstream $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *or completion-seal consumer reads the public route exactly through* $Q, S, G, D, R, W, H, C, P, N$. *The row* Q *is the* $\mathsf{Rat}^{\uparrow}$ *seed row,* S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row,* G *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row,* R *is the* $\mathsf{Real}^{\uparrow}$ *diagonal row,* W *is the transported seal row,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is local* NameCert. *The route exports no selected limit, quotient-stream equality, host equality, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier public route`

Proof. Start with the NameCert obligations and seed-window readback. They identify the accepted carrier as the displayed tuple $Q, S, G, D, R, W, H, C, P, N$ and expose the RatUp seed, StreamName window, RegSeqRat tail, dyadic ledger, real-facing row, transported seal, transport, replay, provenance, and naming rows.

Use the common-window refinement and the RegSeqRat–StreamName handoff. These theorems force every RegSeqRatUp or RealUp reading to refine to the same finite window surface and to pass through the displayed tail readback before reaching any real-facing consumer.

Apply the primitive scope and compatibility handoff. The primitive scope blocks unrecorded preservation, hidden selector, and ambient completion data, while the compatibility handoff routes every DiagonalLimitCompatibilityUp or completion-seal consumer through the same finite tuple before it may consume the compatibility packet.

The only remaining rows are $Q, S, G, D, R, W, H, C, P, N$. Since the carrier has no coordinate for a selected limit, quotient-stream equality, host equality, or ambient completeness theorem, the stated public route is exhausted by those rows. □

Theorem 632.10 (Diagonal cofinal-tail window exhaustion)**.** *Every certified finite cofinal-tail observation exported by an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier factors through the finite rows*

$$Q, S, G, D, R, W, H, C, P, N.$$

In particular, the consumer may inspect only the displayed RatUp seed classifier, the constant StreamNameUp window, the RegSeqRatUp tail readback, the DyadicRatCoreUp finite-window ledger, the RealUp diagonal representative, the transported seal row, transports, continuations, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier window exhaustion`

Proof. Project the accepted carrier and its local naming surface. The observation rows are Q, S, G, D, R, W, and the structural rows are H, C, P, N.

Apply Definition 9238.18 to the window read by W. It says that the finite StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp readings all agree with the same RatUp seed classifier.

The continuations in C replay only those readings, and H transports them componentwise. No carrier coordinate remains for an unbounded window family, so every cofinal-tail observation is exhausted by the displayed finite packet. □

Theorem 632.11 (Diagonal cofinal-tail selector lock)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

any cofinal-tail selector visible to a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer is determined by the displayed stationary* $\mathsf{Rat}^{\uparrow}$ *seed row* Q*, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback row* G*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger* D*, the transported seal row* W*, and the route rows* H, C, P, N*. No unrecorded selector, quotient-stream equality, selected limit, or ambient completeness datum can select a different tail for that consumer.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier tail selector lock`

Proof. Project the accepted carrier. The only tail-bearing rows are the stationary seed row, the constant window row, the regular-rational readback row, the dyadic finite-window ledger, and the transported seal row, with H, C, P, N giving transport, route replay, provenance, and local naming.

Apply Definition 632.10. It forces every finite cofinal-tail observation read by a downstream consumer through the same displayed packet $Q, S, G, D, R, W, H, C, P, N$.

Now use Definition 632.12. The $\mathsf{Real}^{\uparrow}$-facing read is the transported stationary diagonal row named by the $\mathsf{Rat}^{\uparrow}$ seed classifier, so the real-facing selector and the $\mathsf{RegSeqRat}^{\uparrow}$ tail readback are the same finite route.

Since the carrier contains no coordinate for an unrecorded selector, quotient-stream equality, selected limit, or ambient completeness datum, no consumer-visible route remains that could choose a different cofinal tail. □

Theorem 632.12 (Diagonal cofinal-tail Real seal handoff)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier exported by* R *and* W *is reached only through the displayed StreamNameUp window, RegSeqRatUp seal consumption row, and DyadicRatCoreUp finite-window ledger. Thus the real-facing seal handoff is the same transported stationary diagonal row named by the RatUp seed classifier in* Q*.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier real seal handoff`

Proof. Read the seal row W and the representative row R. The transported stationary diagonal theorem identifies the real-facing diagonal representatives with the constant StreamNameUp row induced by Q.

Now restrict to the window selected by D. The finite-window seal exhaustion theorem forces the RegSeqRatUp seal consumption row and the DyadicRatCoreUp ledger to be the public middle rows through which the RealUp classifier is read.

Therefore the real-facing handoff cannot introduce another comparison datum. It is the same transported stationary diagonal row replayed through S, G, D, R, W and the structural routes H, C, P, N. □

Theorem 632.13 (Diagonal cofinal-tail non-escape)**.** *An accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet has no observation path outside the finite cofinal-tail rows of Definition 632.1. Any downstream route that consumes its RealUp seal, RegSeqRatUp readback, DyadicRatCoreUp ledger, StreamNameUp window, or RatUp seed classifier must factor through* $Q, S, G, D, R, W, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier nonescape`

Proof. The NameCert obligations identify every row that may be exported from the packet. The window exhaustion theorem shows that the finite cofinal-tail observations factor through those rows.

The Real seal handoff theorem then covers the only real-facing consumer path. It routes the RealUp classifier through the StreamNameUp, RegSeqRatUp, and DyadicRatCoreUp rows already named by the RatUp seed classifier.

Since no further coordinate is present, a downstream consumer cannot leave the packet to obtain an external equality identification, selected limit, unrecorded selector, or ambient completeness witness. □

Theorem 632.14 (Diagonal cofinal-tail TasteGate carrier single source)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet*

$$T = (Q, S, G, D, R, W, H, C, P, N),$$

every carrier row available to a later TasteGate-facing consumer is generated from this single displayed cofinal-tail packet. The consumer may read the RatUp seed row Q, StreamNameUp window S, RegSeqRatUp tail readback G, DyadicRatCoreUp ledger D, RealUp representative R, transported seal W, and structural rows H, C, P, N; it has no second carrier, second selector, or parameter source.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailTasteGate single carrier alignment`

Proof. Project the carrier of Definition 632.1. The displayed tuple lists every public row of the cofinal-tail packet.

Apply Definition 632.2. The local naming surface uses exactly those rows for carrier admission, classifier exposure, transport, continuation, provenance, and naming.

The window exhaustion theorem, Definition 632.10, keeps every finite cofinal-tail observation inside the same tuple. Hence a later TasteGate-facing read has only this carrier source. □

Theorem 632.15 (Diagonal cofinal-tail TasteGate hidden input refusal)**.** *An accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet refuses every TasteGate-facing hidden input that is not a coordinate of*

$$T = (Q, S, G, D, R, W, H, C, P, N).$$

In particular, consumers cannot read unrecorded selectors, quotient-stream equality, selected limits, ambient completeness, or external equality, because none of these objects is a displayed carrier row or structural route.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier tastegate hidden input refusal`

Proof. Begin with Definition 632.11. It rules out unrecorded selectors, quotient-stream equality, selected limits, and ambient completeness as alternate tail-selection data.

Use Definition 632.13 for the full consumer boundary. Any route consuming the RealUp seal, RegSeqRatUp readback, DyadicRatCoreUp ledger, StreamNameUp window, or RatUp seed classifier must remain inside $Q, S, G, D, R, W, H, C, P, N$.

External equality has no coordinate in the carrier definition and no route in C. Therefore it cannot be read by a TasteGate-facing consumer. □

Theorem 632.16 (Diagonal cofinal-tail TasteGate flow recognition)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier, the visible recognition route from the* $\mathsf{Rat}^{\uparrow}$ *seed rows through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *is the only flow available to later TasteGate wiring. Every recognized read follows the displayed rows* $Q, S, G, D, R, W, H, C, P, N$*, and no later wiring may recognize an unlisted analytic or host-side object as part of this packet.*

Proof. First identify the positive flow. The carrier definition lists Q for RatUp, S for StreamNameUp, G for RegSeqRatUp, D for DyadicRatCoreUp, R and W for RealUp-facing seal data, and H, C, P, N for transport, continuation, package, and naming.

Apply Definition 632.12. The RealUp-facing row is reached only through the StreamNameUp, RegSeqRatUp, and DyadicRatCoreUp rows named by the RatUp seed classifier.

Finally combine Definition 632.14 with Definition 632.15. The single-source result supplies the only recognized route, while the refusal result excludes hidden inputs outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier tastegate flow recognition`

Theorem 632.17 (Diagonal cofinal-tail route composition)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *carrier, every accepted route from the stationary* $\mathsf{Rat}^{\uparrow}$ *seed rows through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *factors through the displayed packet rows*

$$Q, S, G, D, R, W, H, C, P, N.$$

The composed route is obtained by reading Q *into* S*,* S *into* G*,* G *into* D*,* D *into* R*, and* W *into the same public window readings, with* H *carrying componentwise transport,* C *carrying route replay,* P *carrying provenance, and* N *carrying local naming.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier route composition`

Proof. Start with the carrier definition. The tuple $T = (Q, S, G, D, R, W, H, C, P, N)$ gives all rows available to the local $\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}$ surface, and Definition 632.2 identifies the admissible readings of those rows.

Next compose the displayed routes. The row C records the visible continuations from Q to S, from S to G, from G to D, from D to R, and from W to each public window reading.

The window exhaustion theorem forces every cofinal-tail observation through the same finite packet, while the Real seal handoff theorem makes the real-facing representative the transported stationary diagonal row named by Q.

Combining these facts leaves no accepted route outside $Q, S, G, D, R, W, H, C, P, N$. The route therefore factors through the displayed packet rows. □

Theorem 632.18 (Diagonal cofinal-tail public export)**.** *The* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *public surface is the single displayed packet*

$$T = (Q, S, G, D, R, W, H, C, P, N).$$

It exports the carrier of Definition 632.1, the NameCert obligations of Definition 632.2, seed-window readback, common-window refinement, Real seal handoff, non-escape, the TasteGate single-carrier alignment of Definition 632.14, hidden-input refusal, and route composition as one consumer-readable $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surface.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier public export`

Proof. Begin with the carrier and NameCert obligations. They expose exactly the rows $Q, S, G, D, R, W, H, C, P, N$ and the local naming surface attached to those rows.

Seed-window readback and common-window refinement then give the positive downstream route. They identify the stationary RatUp seed, constant StreamNameUp window, RegSeqRatUp tail readback, DyadicRatCoreUp ledger, and RealUp-facing rows as one finite window surface.

The Real seal handoff and non-escape theorem close the real-facing boundary. The seal is read only through the displayed StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp rows, and no selected limit, quotient-stream equality, unrecorded selector, external equality, or ambient completeness coordinate is present.

Finally apply Definition 632.14, hidden-input refusal, and route composition. The TasteGate-facing route has one carrier source, refuses inputs outside the tuple, and composes every accepted consumer read through the same displayed packet. □

Theorem 632.19 (Diagonal cofinal-tail public consumer exactness)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the public* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *surface reads exactly the rows*

$$Q, S, G, D, R, W, H, C, P, N$$

of the accepted packet. Such a consumer reads no selected limit, quotient-stream equality, unrecorded selector, external equality, ambient completeness witness, second carrier source, or hidden parameter outside the displayed cofinal-tail tuple.

Proof. Use the public export theorem to fix the only exported surface as $T = (Q, S, G, D, R, W, H, C, P, N)$. The consumer starts from that packet and not from a separate RegSeqRatUp or RealUp parameter source.

Route composition forces every accepted read through the displayed continuation chain. The hsame transport lock preserves those same rows under componentwise movement and does not add external equality or quotient-stream data.

Apply Definition 632.14 and the hidden-input refusal theorem. The later consumer has a single carrier source and any request for an unrecorded selector, selected limit, external equality, ambient completeness witness, second carrier, or hidden parameter is outside the packet.

Therefore the exact consumer surface is precisely the listed tuple and nothing beyond it. □

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailPublicConsumerExactness`

Theorem 632.20 (Diagonal cofinal-tail hsame transport lock)**.** *For an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet, componentwise* hsame *transport preserves the same cofinal-tail packet rows* $Q, S, G, D, R, W, H, C, P, N$*. In particular, transport along* H *cannot introduce an external equality identification, a selected limit, quotient-stream equality, or ambient completeness as an additional carrier row or consumer-readable route.*

Proof. Project the transport row H. By construction it transports only the component rows Q, S, G, D, R, W and the structural rows already displayed in the packet.

Apply the tail-selector lock. It says that every consumer-visible cofinal-tail selector is determined by the stationary RatUp seed row, constant StreamNameUp window row, RegSeqRatUp readback, DyadicRatCoreUp ledger, transported seal row, and the structural routes already named in the packet.

Now use non-escape. Any route consuming the RealUp seal, RegSeqRatUp readback, DyadicRatCoreUp ledger, StreamNameUp window, or RatUp seed classifier must remain inside $Q, S, G, D, R, W, H, C, P,$

Thus componentwise hsame transport preserves the displayed packet and cannot create any unlisted equality, limit, quotient-stream, or completeness datum. □

Lean4 ✓ BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier hsame transport lock

Theorem 632.21 (Diagonal cofinal-tail finite-window replay determinacy)**.** *Let T and T' be accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packets. Suppose the two packets share the transported* $\mathsf{Rat}^{\uparrow}$ *seed row Q, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger D, the transported stationary seal row W, and the structural rows H, C, P, N along the displayed componentwise transports. If both packets replay the same finite cofinal-tail window to* $\mathsf{Real}^{\uparrow}$ *consumers, then their consumer-visible R rows are classified together by the same route*

$$Q, S, G, D, R, W, H, C, P, N.$$

The determinacy is only a finite-window replay statement; it does not select a limit, identify quotient streams, assert external equality, or invoke ambient completeness.

Proof. First apply Definition 632.11. The shared Q, S, G, D, W rows determine the cofinal-tail selector visible to any replay consumer, so the two packets cannot choose different finite windows while keeping those rows fixed.

Next apply Definition 632.17. The RatUp seed, StreamNameUp window, RegSeqRatUp tail readback, DyadicRatCoreUp ledger, and RealUp-facing row are composed through the displayed continuation route rather than through an external equality or completion witness.

Now use Definition 632.20. Componentwise transport along H preserves the same packet rows and cannot add a second seal, quotient-stream equality, selected limit, or ambient completeness coordinate.

Finally apply Definition 632.23. Every downstream $\mathsf{Real}^{\uparrow}$ read is exhausted by the displayed tuple, so two accepted replays over the same finite window rows classify their consumer-visible R rows through the same route and no larger analytic surface. □

Lean4 ✓ BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier finite window replay determinacy

Theorem 632.22 (Diagonal cofinal-tail finite-window projection exactness)**.** *Let $T = (Q, S, G, D, R, W, H, C, P, N$ be an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet. Every finite-window projection exposed to a* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer is exactly the projection of the displayed S, G, D, R, W rows through the displayed H, C, P, N rows. Equivalently, each consumer-visible finite window is obtained by replaying the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger row D, the* $\mathsf{Real}^{\uparrow}$ *handoff row R, and the transported stationary seal row W along the componentwise transport, continuation, provenance, and naming rows already carried by H, C, P, N.*

No quotient-stream equality, selected limit, host real equality, ambient completeness object, or second selector can refine that finite-window projection. Such data are not part of the accepted projection surface and cannot be read by the three consumer classes.

Lean4 ✓ BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier window projection exactness

Proof. First use Definition 632.21. Shared replay of the same finite cofinal-tail window fixes the consumer-visible R row through the route $Q, S, G, D, R, W, H, C, P, N$, so the projection has no alternate replay source.

Next apply Definition 632.20. Componentwise transport along H preserves the rows $Q, S, G, D, R, W, H, C, P, N$ and therefore transports the projection row by row rather than adding quotient-stream equality, host real equality, a selected limit, or ambient completeness.

Finally apply Definition 632.23. Every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer is exhausted by the displayed tuple. Since no second selector or analytic refinement is consumer-readable, the finite-window projection exposed downstream is exactly the displayed projection through $S, G, D, R, W, H, C, P, N$. □

Theorem 632.23 (Diagonal cofinal-tail consumer exhaustion)**.** *Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet reads the displayed rows*

$$Q, S, G, D, R, W, H, C, P, N$$

and no unlisted analytic object. In particular, the consumer surface contains no hidden selector, external equality identification, quotient-stream equality, selected limit, ambient completeness witness, or second carrier source.

Proof. First apply route composition. It factors every accepted route from the RatUp seed rows through the StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, transport, continuation, provenance, and naming rows of the displayed packet.

Next apply the hsame transport lock. It prevents componentwise transport from adding external equality, quotient-stream equality, selected limits, or ambient completeness to the consumer-visible surface.

Finally use TasteGate hidden input refusal. A later consumer cannot read anything that is not a carrier row or structural route of $T = (Q, S, G, D, R, W, H, C, P, N)$.

Therefore any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer is exhausted by the displayed cofinal-tail packet rows and has no access to an unlisted analytic object. □

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier consumer exhaustion`

Theorem 632.24 (Diagonal cofinal-tail stationary seal uniqueness)**.** *Let T and T' be accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packets. Suppose they share the displayed stationary* $\mathsf{Rat}^{\uparrow}$ *seed row Q, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger D, and the transported seal row W up to the componentwise transports recorded in H. Then every* $\mathsf{Real}^{\uparrow}$ *consumer reads the same real-facing seal row R through the same route*

$$Q, S, G, D, R, W, H, C, P, N.$$

In particular, the consumer cannot distinguish T from T' by an alternate real seal, selected limit, quotient-stream equality, ambient completeness witness, or second carrier source.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier stationary seal uniqueness`

Proof. First apply Definition 632.11. The shared Q, S, G, D, W rows and the displayed H transports determine the cofinal-tail selector visible to any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer.

Next use Definition 632.12. The real-facing row is not read from an independent completeness object; it is the transported stationary diagonal row reached through the StreamNameUp, RegSeqRatUp, and DyadicRatCoreUp rows already named by Q.

Now apply Definition 632.20. Componentwise transport along H preserves the same packet rows, so transporting from T to T' cannot create a second real seal or a new equality coordinate.

Finally use Definition 632.23. Every downstream $\mathsf{Real}^{\uparrow}$ read is exhausted by $Q, S, G, D, R, W, H, C, P, N$, hence two packets with the same stationary seed, window, tail readback, dyadic ledger, and transported seal give the same real seal to every consumer on that route. □

Theorem 632.25 (Diagonal cofinal-tail terminal seal factorization)**.** *Let T and T' be accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packets. Suppose every terminal* $\mathsf{Real}^{\uparrow}$ *seal comparison or downstream completion-seal consumer visible to the two packets agrees on the stationary* $\mathsf{Rat}^{\uparrow}$ *seed row Q, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger D, the transported seal row W, and the componentwise transport row H. Then that terminal consumer factors through the same ordered finite route*

$$Q, S, G, D, R, W, H, C, P, N.$$

The consumer obtains no selected limit, quotient-stream equality, host real equality, countable-choice selector, ambient completeness witness, or second carrier source beyond this route.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier terminal seal factorization`

Proof. First apply Definition 632.11. The shared Q, S, G, D, W rows determine the cofinal tail visible to any terminal comparison, so no separate selector or countable-choice source can drive the consumer.

Next use Definition 632.12. The terminal $\mathsf{Real}^{\uparrow}$ row is forced through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows rather than through a host real equality, quotient-stream equality, selected limit, or ambient completeness witness.

Now apply Definition 632.20. Componentwise transport along H preserves the packet route $Q, S, G, D, R, W, H, C, P, N$ for both T and T', so the terminal seal consumer cannot acquire a second carrier source while moving between packets.

Finally combine Definition 632.24 with Definition 632.23. The shared real-facing row is unique on the stationary route, and every terminal $\mathsf{Real}^{\uparrow}$ or completion-seal read is exhausted by the ordered finite rows displayed above. Hence the consumer factors through the complete finite route and through no unlisted terminal object. □

Theorem 632.26 (Diagonal cofinal-tail terminal common-tail descent)**.** *Let T and T' be accepted* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packets. Suppose every terminal* $\mathsf{Real}^{\uparrow}$ *comparison or completion-seal consumer visible to T and T' agrees on the stationary* $\mathsf{Rat}^{\uparrow}$ *seed row Q, the constant* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window ledger D, the transported seal row W, and the componentwise transport row H. Then every terminal comparison descends through the same common finite tail route*

$$Q, S, G, D, R, W, H, C, P, N.$$

No quotient-stream equality, selected limit, host real equality, countable-choice selector, ambient completeness witness, metric-completeness row, or second carrier source can distinguish the terminal reads.

Proof. First apply Definition 632.25. The terminal $\mathsf{Real}^{\uparrow}$ comparison and every completion-seal consumer factor through the ordered finite route $Q, S, G, D, R, W, H, C, P, N$ once they agree on the seed, window, tail readback, dyadic ledger, transported seal, and transport rows.

Next use Definition 632.24. The shared stationary rows force the real-facing seal row R to be the same consumer-visible row on both packets, so descent does not branch through a second terminal seal.

Now apply Definition 632.23. The consumer surface has no unlisted analytic coordinate, hence quotient-stream equality, selected limits, host real equality, countable-choice selectors, ambient completeness witnesses, and metric-completeness rows are unavailable as distinguishing data.

Finally use Definition 632.20. Componentwise transport along H preserves the displayed route row by row, so the terminal reads from T and T' descend through the same common finite tail and cannot be separated by a second carrier source. □

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier terminal common tail descent`

Theorem 632.27 (Diagonal cofinal-tail exact-boundary terminal route)**.** *Let*

$$T = (Q, S, G, D, R, W, H, C, P, N)$$

be an accepted $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet. Suppose an accepted* $\mathsf{RealCompletionExactBoundaryUp}$ *terminal boundary consumes T as its completion-facing terminal route. Then the boundary's four Phase Real and Completion faces are read through the same finite route*

$$Q, S, G, D, R, W, H, C, P, N.$$

The $\mathsf{StreamName}^{\uparrow}$ *face is the constant window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face is the cofinal-tail readback row G, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face is the finite-window ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *face is the real-facing row R, all carried through the transported seal, componentwise transport, continuation, provenance, and local naming rows W, H, C, P, N. The terminal read cannot introduce quotient streams, host real equality, selected limits, countable choice, metric completeness, ambient completeness, bridge rows, or a second carrier source.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier exact boundary terminal route`

Proof. First apply Definition 632.26. Any terminal comparison or completion-seal consumer visible to the accepted $\mathsf{DiagonalCofinalTail}^{\uparrow}$ packet descends through the common route $Q, S, G, D, R, W, H, C, P, N$ and cannot distinguish the read by an unlisted analytic coordinate.

Next apply Definition 632.25. The terminal seal read factors through the same ordered rows once the seed, window, tail readback, dyadic ledger, transported seal, and transport rows have been fixed. Thus the boundary consumer sees the diagonal packet through a single finite carrier route rather than through a detached completion object.

Now use Definition 1120.6. Its accepted boundary exposes only the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ faces to Phase Real and Completion consumers. Under the diagonal consumption hypothesis, these four faces are exactly the rows S, G, D, R already present in T.

Finally apply Definition 1120.13. Every completion-facing or terminal $\mathsf{Real}^{\uparrow}$ consumer of the accepted boundary must read the same finite route and may not add quotient streams, host real equality, selected limits, choice, completeness, bridge coordinates, or a second source. Hence the exact-boundary terminal read is the displayed diagonal cofinal-tail route. □

Theorem 632.28 (Diagonal cofinal-tail zero-source terminal coincidence)**.** *Let*

$$T = (Q, S, G, D, R, W, H, C, P, N)$$

be an accepted $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *packet, and let*

$$Z = (q, s, r, m, \mu, h, c, p, n)$$

be an accepted RegularCauchyZero$^{\uparrow}$ *packet. Suppose the stationary seed row Q agrees with the zero seed q, the constant window row S agrees with the zero* StreamName$^{\uparrow}$ *window s, the cofinal-tail readback row G agrees with the zero* RegSeqRat$^{\uparrow}$ *readback r, the dyadic ledger row D agrees with the zero packet's dyadic observation route, and the terminal* Real$^{\uparrow}$ *seal row R agrees with the zero seal read exported by Z.*

Then every terminal Real$^{\uparrow}$ *consumer and every completion-seal consumer visible to T reads the same finite route as the zero packet. That route is exactly*

Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert

as displayed by the rows $Q, S, G, D, R, W, H, C, P, N$ and by the matching zero-source rows of Z. The consumer obtains no quotient stream, host real equality, selected limit, countable-choice selector, ambient completeness witness, or second terminal source.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailCarrier zero source terminal coincidence`

Proof. First apply Definition 624.8 to the accepted zero packet. The hypotheses identify the zero seed, constant stream window, regular zero readback, dyadic observation route, and real seal read with the stationary rows visible in T, so the zero-source side has a single terminal consumer boundary.

Next apply Definition 632.26. Once $Q, S, G, D, R, W, H, C, P, N$ are fixed, every terminal comparison or completion-seal consumer of the diagonal packet descends through the same common tail route and cannot branch through a quotient stream, selected limit, host equality, choice, completeness, or another carrier source.

Now use Definition 632.27. The completion-facing exact boundary reads the diagonal packet through the four displayed terminal faces S, G, D, R and through the structural rows W, H, C, P, N. Under the zero-source agreement assumptions, these are precisely the zero seed, zero constant window, zero readback, dyadic ledger, and zero real seal already fixed by Z.

Finally apply Definition 360.51. Downstream RealCauchyCompletion$^{\uparrow}$ support retains the displayed DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ faces together with BHist, hsame, Cont, Pkg, and NameCert. Hence the diagonal terminal read and the zero-source terminal read coincide on exactly the listed BEDC route, and no unlisted completion coordinate can supply a second terminal source. □

Theorem 632.29 (Diagonal cofinal-tail primitive scope package)**.** *The carrier, obligations, window exhaustion, selector lock, Real seal handoff, non-escape boundary, TasteGate single-carrier alignment, TasteGate hidden-input refusal,* hsame *transport lock, consumer exhaustion, stationary seal uniqueness, terminal seal factorization, terminal common-tail descent, exact-boundary terminal route, and zero-source terminal coincidence for* DiagonalCofinalTail$^{\uparrow}$ *are scoped exactly over* Rat$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the checked taste-gate marker. The scope contains no second carrier source and no unrecorded selector.*

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.DiagonalCofinalTailScopePackage`

Proof. The carrier and obligation package provide the primitive inventory. The rows $Q, S, G, D, R, W, H, C, P, N$ name the RatUp seed, StreamNameUp window, RegSeqRatUp tail readback, DyadicRatCoreUp finite-window ledger, RealUp handoff, transported seal, componentwise transport, continuation routes, provenance, and local naming surface.

Every consumer is then routed through the displayed route composition, the hsame transport lock, terminal seal factorization, terminal common-tail descent, the exact-boundary terminal route, and the zero-source terminal coincidence. These clauses force downstream RegSeqRatUp, DyadicRatCoreUp, RealUp, zero-source, and completion-seal readings to factor through the same finite packet and prevent transport from adding another equality, limit, completion, or selector coordinate.

Finally use the TasteGate rows. The carrier single-source theorem supplies the unique later carrier source, and the hidden-input refusal theorem excludes every TasteGate-facing input not already listed in $T = (Q, S, G, D, R, W, H, C, P, N)$. Thus the primitive scope is exactly the displayed BEDC object list and the checked taste-gate marker. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalCofinalTail}^{\uparrow}$ packet is generated from paired stationary RatUp seed rows, constant StreamNameUp windows, RegSeqRatUp cofinal-tail readbacks, DyadicRatCoreUp finite-window ledgers, RealUp diagonal representative rows, a transported stationary seal row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 632.9, Definition 632.22, Definition 632.26, and Definition 632.28 bind the mature transported stationary diagonal window over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DiagonalCofinalTailUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalCofinalTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, external equality identification, selected real limits, unbounded diagonal families, Lean verification, or bridge checking.

Upgrade path: Lean verification remains the separate path for the transported stationary diagonal window over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Chapter 633

A Concrete Naming Certificate for $\mathsf{CausalContinuitySocket}^{\uparrow}$

The $\mathsf{CausalContinuitySocket}^{\uparrow}$ packet realizes the Type IV causal-continuity socket as a finite certificate surface. It keeps record-side regularity, proposed prediction shape, and the audit gate in separate rows, so a successful finite record cannot be read as an internal future-law theorem.

The Type IV gate row G is the causal-continuity socket-kind row consumed from chapter 523 before the local socket site S is read. The taxonomy supplies the socket-kind ledger; $\mathsf{CausalContinuitySocket}^{\uparrow}$ specializes that row to the record, prediction, gate, and site packet below.

The Type IV gate row should be read as a causal-continuity specialization of the finite refusal boundary in chapter 630, with R, P, G, S replacing socket, refused-internalizer, GAP, and audit rows for the record-to-prediction passage.

This socket is consumed by the inter-Hist causal-rate chain, especially chapter 876 and chapter 656, because those chapters need finite record and prediction audit rows before any rate or invariant consumer may read a future-facing route.

Definition 633.1 (Causal continuity socket carrier)**.** *A* $\mathsf{CausalContinuitySocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R, P, G, S, H, C, Q, N)$$

with the following displayed rows:

1. *R is the bounded record-regularity row, recording the finite observation window, its accepted regularity summary, and the last record boundary;*
2. *P is the proposed prediction-shape row, recording the not-yet-observed continuation pattern as a request shape rather than as a theorem of persistence;*
3. *G is the* GAP *and audit-gate row, recording the Type IV question that separates record support from causal-continuity commitment;*
4. *S is the socket-site row, recording the manuscript, observation, or certificate location where the prediction shape is requested from the record;*
5. *H records componentwise* hsame *transport for R, P, G, S, the continuation row, the provenance row, and the local naming row;*
6. *C records the displayed* Cont *routes by which consumers inspect the record row, prediction-shape row, audit gate, socket site, provenance, and local certificate;*

7. *Q records* Pkg *provenance over* BHist*, bounded records, prediction-shape ledgers,* GAP*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CausalContinuitySocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for a universal future law, selected unseen observation, completed empirical stream, host causality principle, or theorem converting finite regularity into persistence.

Lean4 ✓ `BEDC.Derived.CausalContinuitySocketUp.CausalContinuitySocketTasteGate single carrier alignment`

Theorem 633.2 (Causal continuity record-prediction separation)**.** *For an accepted* $\mathsf{CausalContinuitySocket}^{\uparrow}$ *carrier, the record-side row R and prediction-shape row P remain distinct displayed coordinates. A downstream consumer may cite R as bounded regularity evidence and may cite P as a requested continuation shape, but no consumer may convert the pair (R, P) into a future-law commitment without passing through the Type IV gate row G.*

Proof. Project the accepted carrier X. The record evidence, prediction shape, and gate are the separate rows R, P, G.

The row R records only bounded observation data and the accepted finite regularity summary. The row P records only the requested continuation shape, not its fulfillment.

Any route from record support to prediction use must therefore pass through G and the socket-site row S. Since the carrier contains no future-law coordinate, the pair (R, P) cannot itself supply causal continuity. □

Lean4 ✓ `BEDC.Derived.CausalContinuitySocketUp.CausalContinuitySocket record prediction separation`

Theorem 633.3 (Causal continuity audit gate)**.** *The audit gate of an accepted* $\mathsf{CausalContinuitySocket}^{\uparrow}$ *packet is exactly the row G attached to the record row R, prediction row P, and socket site S. A use of record regularity for not-yet-observed history is visible precisely when it factors through*

$$R, P, G, S, H, C, Q, N.$$

Proof. Read the record and prediction sides first. The row R supplies the bounded regularity evidence, and P supplies the proposed continuation shape.

The gate row G is the displayed Type IV audit question linking those two rows at the site S. Thus visible use of the prediction route is use through the gate and site rows.

The rows H, C, Q, N transport, replay, package, and name the same visible gate. A route omitting G has no accepted coordinate that marks record-to-prediction passage, so it is not an accepted visible use. □

Theorem 633.4 (Causal continuity socket non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{CausalContinuitySocket}^{\uparrow}$ *packet factors through the finite rows*

$$R, P, G, S, H, C, Q, N.$$

The consumer may inspect bounded record regularity, proposed prediction shape, the Type IV gate, socket site, transport rows, continuation routes, package provenance, and the local NameCert *row. It cannot extract a universal future law, selected unseen observation, completed empirical stream, host causality principle, or persistence theorem from the packet.*

Proof. Start from the carrier definition. The packet exposes exactly the rows R, P, G, S, H, C, Q, N.

The record-prediction separation theorem confines the evidential read to R and the requested continuation shape to P. The audit-gate theorem then requires G and S for any visible route from one side to the other.

The remaining rows only transport, replay, package, and name this same finite packet. Because no future-law or unseen-observation coordinate exists in the carrier, no downstream consumer can project such a coordinate from it. □

Theorem 633.5 (Causal continuity socket NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CausalContinuitySocket}^{\uparrow}}$ *surface for Definition 633.1 consists exactly of carrier admission for*

$$X = (R, P, G, S, H, C, Q, N),$$

bounded record-regularity exposure, prediction-shape display, Type IV audit-gate exactness, socket-site locality, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal to internalize causal continuity as a theorem of the record.*

Proof. Carrier admission is projection from the displayed packet. The rows R, P, G, S, H, C, Q, N are the complete local inventory.

The record and prediction obligations are read from R and P. They expose bounded regularity and proposed continuation shape while keeping them separate.

The gate and site obligations are read from G and S, and the transport and replay obligations are read from H and C. Finally Q and N record provenance and local naming.

Together these rows give the NameCert surface. Since the inventory contains no causal-continuity theorem or completed future-history coordinate, the certificate names the socket without internalizing the requested persistence. □

Closure status for $\mathsf{NameCert}_{\mathsf{CausalContinuitySocket}^{\uparrow}}$.

Constructive story: A $\mathsf{CausalContinuitySocket}^{\uparrow}$ packet is generated from finite BHist rows for bounded record regularity, proposed prediction shape, Type IV audit gate, socket site, transport, route, provenance, and local naming. The record and prediction rows are deliberately separate: Cont routes any attempted record-to-prediction use through the displayed gate and site before a consumer can cite it. Componentwise hsame transports the record, prediction, gate, and site rows without turning them into a future-law theorem, while Pkg records the taxonomy and boundary dependencies. The socket therefore names a finite request surface, not completed persistence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CausalContinuitySocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the causal-continuity socket carrier, record-prediction separation, audit gate, non-escape theorem, and NameCert obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CausalContinuitySocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a universal future law, selected unseen observation, completed empirical stream, host causality principle, or persistence theorem.

Upgrade path: Obligation closure requires Lean encodings of the socket carrier, gate route, record-prediction separation, and non-escape boundary.

Chapter 634

A Concrete Naming Certificate for QuotientSoundnessBoundary$^{\uparrow}$

The QuotientSoundnessBoundary$^{\uparrow}$ interface names the finite boundary at which a quotient-style equality request is refused as an unwitnessed representative-selection step and replaced by classifier-respecting psame transport. It does not construct a quotient type, select a canonical representative, collapse equivalent source rows, or import Quot.sound. It records the visible source relation rows, requested representative rows, transport replacement, refusal verdict, locality routes, provenance, and naming certificate that downstream quotient-like chapters may consume.

The packet consumes the refusal discipline of chapter 998 together with its local NameCert boundary, so quotient equality requests are converted into displayed refusal-and-transport rows before they can enter BEDC public reads.

When a request names Quot.sound as external supply, the quotient-soundness boundary consumes the axiom-dependency tuple audit row of chapter 861 and converts that request into the displayed refusal-and-transport rows.

The refusal surface is the quotient-facing specialization of the negative-boundary discipline of chapter 629: representative-selection requests are first treated as blocked boundary reads, and only then replaced by local psame transport.

Host-primitive leakage refusal is read through chapter 1157 before any quotient-style representative request may be replaced by psame transport.

Real-and-Completion quotient requests through chapter 1708 consume the same refusal-before-transport discipline before a RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ seal can be read.

Definition 634.1 (QuotientSoundnessBoundary carrier)**.** *A* QuotientSoundnessBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, A, T, V, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite family of quotient-relation rows, recording the source carrier entries, their classifier relation, and the displayed equivalence-derivation data admitted by the packet;*
2. *A is the requested representative row family, recording each attempted demand for a canonical representative, section, quotient equality token, or erased derivation tree;*
3. *T is the* psame *transport replacement row, recording the classifier-respecting route by which an operation reads related source rows without selecting a representative;*

4. *V is the refusal verdict row, recording that the requested representative-selection surface in A is not admitted and that downstream use must pass through T;*
5. *H records componentwise* hsame *transport for the relation rows, requested representative rows, transport replacement, refusal verdict, continuation routes, provenance, and local naming row;*
6. *C records the visible* Cont *routes from the relation rows to the refusal verdict, from the relation rows to the* psame *transport replacement, and from the transport replacement to the downstream classifier-respecting consumer;*
7. *P is the* Pkg *provenance over* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$*, quotient-relation rows, requested representative rows,* psame*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}$ *row.*

The carrier exports only a finite refusal-and-transport packet. It contains no coordinate for a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, or Quot.sound *dependency.*

Theorem 634.2 (QuotientSoundnessBoundary NameCert obligations)**.** *For every accepted* QuotientSoundnessBou *carrier*

$$Q = (E, A, T, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, relation-row admission, requested-representative refusal, classifier-respecting* psame *transport, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and non-admission of quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery maps, and* Quot.sound *dependencies.*

Proof. Carrier admission is projection from Definition 634.1. The tuple exposes the relation rows, requested representative rows, transport replacement, refusal verdict, transport rows, continuation routes, provenance, and local naming row.

The requested representative rows are read through A and judged by V. The verdict attaches the refusal to the displayed demands rather than to an anonymous global prohibition, so every blocked quotient-soundness request remains audit-visible.

The replacement obligation is supplied by T. It routes classifier-respecting consumers through psame transport on the source rows named in E, while H and C keep the transported read componentwise stable and local to the displayed packet.

Since no coordinate of Q is a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, or Quot.sound dependency, the local certificate is exhausted by the listed finite obligations. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryCarrier namecert obligations`

Theorem 634.3 (QuotientSoundnessBoundary transport replacement)**.** *Every downstream operation accepted by a* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet that would otherwise request a quotient representative must instead factor through the source relation rows E, the classifier-respecting* psame *transport row T, the componentwise transport row H, the continuation routes C, and the local naming row N. The result is an operation on named source rows, not an operation on a selected representative of a quotient class.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryCarrier transport replacement`

Proof. Begin with a consumer request recorded in A. By the refusal verdict V, the request is not admitted as a representative selector, quotient carrier, quotient equality eliminator, or derivation eraser.

The only positive route left by the carrier is the transport row T. It reads the relevant relation rows in E and supplies the classifier-respecting psame transport needed by the downstream operation.

The rows H and C then make the replacement stable and local: H moves the relation, refusal, and transport data componentwise, while C displays the route from relation evidence to transport and then to the consumer. Thus the operation proceeds on named source rows and never obtains a hidden quotient representative. □

Theorem 634.4 (QuotientSoundnessBoundary negative refusal dependency)**.** *Every quotient-style representative request accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet first lands in the negative-boundary refusal discipline of chapter 629. This chapter specializes that discipline to the visible relation rows E, requested representative rows A, classifier-respecting* psame *transport row T, refusal verdict V, componentwise transport H, continuation routes C, provenance P, and local naming row N, replacing any appeal to* Quot.sound *by displayed transport through T.*

Proof. Start with the carrier of Definition 634.1. The requested representative rows A and refusal verdict V record the same kind of blocked boundary read handled by the negative-name boundary.

Apply the transport replacement theorem, Definition 634.3. It routes the surviving operation through E, T, H, C, N rather than through a quotient representative or quotient equality eliminator.

Use the sibling refusal surface chapter 629 to read the representative request as a negative boundary request before the transport replacement is consumed. The local packet then records the specialized quotient-facing rows E, A, T, V, H, C, P, N.

Finally cite non-escape, Definition 634.6. Since the excluded quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, and Quot.sound dependency are absent from the packet, the only exported replacement is the displayed psame transport route. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary negative refusal certificate`

Theorem 634.5 (QuotientSoundnessBoundary Cauchy refusal link)**.** *Every* CauchyQuotientBoundary$^{\uparrow}$ *consumer of a* QuotientSoundnessBoundary$^{\uparrow}$ *packet reads the representative request rows A, V before it consumes the displayed* psame *transport route E, T, H, C, P, N. Consequently no quotient stream equality, selected representative, or* Quot.sound *row crosses into the Real-completion boundary.*

Proof. Start with the carrier of Definition 634.1. The rows A record the requested representative surface, and the rows V record the refusal verdict attached to that surface.

Apply refusal locality, Definition 634.7. It forces every representative-selection demand to be inspected as A, V before the positive transport route is consumed.

Apply the negative refusal dependency, Definition 634.4, and then read the Cauchy quotient boundary of chapter 1708. The Real-completion consumer therefore receives the displayed route E, T, H, C, P, N only after the refusal pair has been read.

Finally use non-escape, Definition 634.6. Since quotient stream equality, selected representatives, quotient eliminators, and Quot.sound dependencies are absent from the packet, none of those rows can enter the RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ boundary through the Cauchy consumer. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary cauchy refusal link`

Theorem 634.6 (QuotientSoundnessBoundary non-escape)**.** *An accepted* QuotientSoundnessBoundary$^{\uparrow}$ *carrier cannot export a quotient carrier, canonical representative, quotient equality eliminator, erased equivalence-derivation recovery map, unrestricted quotient-recursion principle, host-level class selector, or* Quot.sound *dependency. Any consumer request for one of these objects leaves the finite packet*

$$E, A, T, V, H, C, P, N$$

and must be supplied by an explicit extra certificate outside this boundary.

Proof. Inspect the carrier inventory. The rows E, A, T, V, H, C, P, N name relation evidence, representative requests, transport replacement, refusal, stability, locality, provenance, and the local certificate.

None of these rows has the type or role of a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, unrestricted quotient-recursion principle, host-level class selector, or Quot.sound dependency. The rows A and V record such demands as refused requests, and T records only the classifier-respecting replacement.

Therefore a downstream consumer cannot project any excluded object from the packet. It may either use the displayed psame transport route or supply a separate certificate whose boundary is audited independently. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary non escape`

Theorem 634.7 (QuotientSoundnessBoundary refusal locality)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *carrier*

$$Q = (E, A, T, V, H, C, P, N),$$

every representative-selection demand is localized to the requested-representative rows A and the refusal verdict rows V before any transport replacement is consumed. No downstream consumer may treat T, H, C, P, or N as evidence that a representative selector, quotient equality eliminator, erased derivation recovery map, or Quot.sound *dependency has been admitted.*

Proof. Project the carrier of Definition 634.1. The only rows that record a demand for a canonical representative, quotient equality token, section, or erased derivation tree are the rows A.

The verdict rows V attach the refusal to those displayed demands. Thus the blocked request is not a global prohibition detached from the packet; it is a local row pair A, V that can be audited before the replacement route is used.

Apply Definition 634.3. The positive route after refusal is the classifier-respecting psame transport through T, stabilized and routed by H and C and named by P, N. Since Definition 634.6 excludes the quotient-side objects, the replacement rows cannot be reread as hidden representative selection. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal locality`

Theorem 634.8 (QuotientSoundnessBoundary psame transport exhaustion)**.** *Every surviving quotient-style operation accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet is exhausted by the displayed route through the source relation rows E, the classifier-respecting* psame *transport row T, the componentwise stability row H, the continuation row C, the package provenance row P, and the local naming row N. After the refusal pair A, V has been read, no second route remains through a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, unrestricted quotient-recursion principle, host-level class selector, or* Quot.sound *dependency.*

Proof. Begin with a surviving operation. By Definition 634.7, its representative-selection demand has already been localized to A, V and refused before any positive transport is consumed.

The carrier then exposes only the source relation rows E and the replacement row T as the operation-facing data. The row T is explicitly a classifier-respecting psame route on named source rows, not an operation on quotient classes.

The rows H, C, P, N provide stability, locality, provenance, and naming for that same route. Apply Definition 634.6 to remove the excluded quotient objects from the exported surface. Therefore the displayed E, T, H, C, P, N route exhausts the surviving operation. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame transport exhaustion certificate`

Theorem 634.9 (QuotientSoundnessBoundary formal-target frontier)**.** *The mature paper-side refusal boundary for* QuotientSoundnessBoundary$^{\uparrow}$ *exhausts the quotient-facing certificate by the finite packet*

$$E, A, T, V, H, C, P, N.$$

The remaining formal target is the Lean encoding of exactly that packet and its psame *transport route through E, T, H, C, P, N. No paper closure grade for this chapter may be read as* Quot.sound*, host quotient construction, representative selection, erased derivation recovery, quotient eliminator admission, or bridge checking.*

Proof. Begin with Definition 634.2. It identifies the carrier, representative-demand, refusal, transport, stability, locality, provenance, and local naming rows as the finite certificate surface.

Apply Definition 634.3. The positive route after refusal is the classifier-respecting psame replacement through the displayed transport row T, not a quotient eliminator or representative selector.

Use Definition 634.6. It removes host quotient construction, quotient equality elimination, erased derivation recovery, representative choice, and Quot.sound from the exported packet.

Finally apply Definition 634.8. The only surviving operation route is E, T, H, C, P, N, while A, V record the refused quotient-facing demands. Hence the paper boundary is mature as a refusal surface, and the remaining frontier is the Lean encoding of this exact finite route rather than a hidden bridge check. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryFormalTargetFrontier`

Theorem 634.10 (QuotientSoundnessBoundary core refusal host-quotient exclusion)**.** *Let*

$$Q = (E, A, T, V, H, C, P, N)$$

be an accepted QuotientSoundnessBoundary$^{\uparrow}$ *packet whose surviving quotient-style route is exhausted by the displayed rows E, T, H, C, P, N after the refusal pair A, V has been read. The exported boundary cannot be reread as host quotient equality, representative selection, a quotient eliminator, erased derivation recovery, or* Quot.sound*. Every downstream quotient-like consumer must cite the displayed* psame *route through E, T, H, C, P, N or provide a separate audited certificate outside this packet.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary core refusal host quotient exclusion`
Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryCoreRefusalHostQuotientExclus`

Proof. Begin with the carrier definition, Definition 634.1. The packet exposes only the relation rows, requested representative rows, psame transport replacement, refusal verdict, stability rows, continuation rows, provenance, and local naming certificate.

Use refusal locality, Definition 634.7. It places every representative-selection demand in A, V before the surviving transport route is consumed, so the later rows cannot be interpreted as an admitted host representative selector.

Apply non-escape, Definition 634.6. It removes quotient carriers, representatives, quotient equality eliminators, erased derivation recovery maps, unrestricted quotient recursion, host-level selectors, and Quot.sound dependencies from the exported packet.

Finally apply Definition 634.8. The positive survivor is exactly the classifier-respecting psame route through E, T, H, C, P, N. Hence any downstream quotient-like consumer either cites that displayed route or leaves the boundary by supplying a separate audited certificate. □

Theorem 634.11 (QuotientSoundnessBoundary downstream audit package)**.** *The refusal locality theorem, the* psame *transport exhaustion theorem, and the non-escape theorem assemble a downstream audit package for quotient-like BEDC chapters. A downstream chapter may consume the refusal verdict A, V and the transport route E, T, H, C, P, N as an auditable refusal-and-transport surface, but it receives no quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, unrestricted quotient-recursion principle, host-level class selector, or* Quot.sound *dependency from* QuotientSoundnessBoundary$^{\uparrow}$.

Proof. Use Definition 634.7 to bind the negative part of the package. It says every representative-selection demand is visible as A and refused by V before replacement.

Use Definition 634.8 for the positive part. It identifies the only surviving operation route as E, T, H, C, P, N, with psame transport carried by T and stabilized through H, C.

Finally apply Definition 634.6. The excluded quotient primitives cannot be projected from the packet, so downstream quotient-like chapters receive an audit surface consisting of refusal plus displayed transport, not a quotient kernel extension. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryDownstreamAuditPackage`

Theorem 634.12 (QuotientSoundnessBoundary fixed classifier)**.** *For an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *carrier $Q = (E, A, T, V, H, C, P, N)$, every accepted comparison is exhausted by the relation rows E, requested representative rows A, replacement row T, refusal verdict V, and the displayed transport rows H, C. No comparison may consult a quotient carrier, canonical representative, quotient equality eliminator, erased derivation recovery map, or* Quot.sound *dependency.*

Proof. Project the carrier of Definition 634.1. The only comparison-facing coordinates are the relation evidence E, the recorded representative demands A, the replacement row T, the verdict V, and the locality rows H, C.

Apply the transport replacement theorem, Definition 634.3. It consumes E and T as the positive classifier-respecting route, while A and V keep representative selection in the refusal ledger.

The non-escape theorem, Definition 634.6, excludes every quotient eliminator and Quot.sound route from the packet. Hence the classifier is fixed by E, A, T, V, H, C and cannot be widened by an ambient quotient principle. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary fixed classifier certificate`

Theorem 634.13 (QuotientSoundnessBoundary kernel non-escape)**.** *Every kernel-facing consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet remains inside the rows*

$$E, A, T, V, H, C, P, N.$$

The packet cannot import Quot.sound*, a host quotient eliminator, a hidden representative selector, or erased relation-derivation recovery.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary kernel nonescape`

Proof. Start with the NameCert obligations of Definition 634.2. They list carrier admission, relation admission, refused representative requests, classifier-respecting transport, componentwise stability, locality, provenance, and local naming as the full public surface.

Use the fixed classifier theorem above to identify the comparison route. The kernel-facing read is not an arbitrary quotient read; it is the finite path through E, A, T, V, H, C and then through package provenance and the local certificate.

Finally apply Definition 634.4 and Definition 634.6. The refusal dependency explains why the blocked request is represented negatively, and non-escape prevents any Quot.sound or host quotient eliminator from entering the kernel route. □

Theorem 634.14 (QuotientSoundnessBoundary root request triad)**.** *Every root quotient-style request accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet is exhausted, before any positive transport is consumed, by the relation rows E, the requested-representative rows A, and the refusal verdict rows V. The rows E name the visible relation evidence, A records the demanded representative-selection surface, and V refuses that demand as a kernel-facing object.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root request triad`

Proof. Project the carrier of Definition 634.1. Its request-facing coordinates include the relation rows E, the requested representative rows A, and the refusal verdict V.

Use the NameCert obligations of Definition 634.2. They identify requested-representative refusal as a named certificate obligation, so the representative demand is read from A and judged by V rather than by an ambient quotient principle.

Finally apply kernel non-escape, Definition 634.13. Since the kernel route cannot import Quot.sound, a host quotient eliminator, a hidden representative selector, or erased derivation recovery, the root request surface is exactly the finite triad E, A, V before the displayed transport row is used. □

Theorem 634.15 (QuotientSoundnessBoundary root relation row exposure)**.** *Every root quotient-style read accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet $Q = (E, A, T, V, H, C, P, N)$ must expose the relation rows E before it exposes any representative-demand row in A. Thus the packet records visible source relation evidence before a downstream consumer can ask for a canonical representative, section, quotient equality token, or erased derivation tree.*

Proof. Project the carrier. The row family E records the source relation evidence, while A records the demanded representative-selection surface.

Use the fixed classifier theorem. It keeps every accepted comparison inside E, A, T, V, H, C and prevents an ambient quotient principle from widening the comparison before relation evidence is visible.

Finally apply the root request triad and kernel non-escape. The triad exhausts the root request by E, A, V, and non-escape excludes hidden representative selectors, erased derivation recovery, host quotient eliminators, and Quot.sound. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryCarrier root relation row exposure`

Theorem 634.16 (QuotientSoundnessBoundary carrier exposure)**.** *Every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet exposes exactly the quotient-refusal carrier rows*

$$(E, A, T, V, H, C, P, N).$$

Here E is the visible relation evidence, A is the representative-demand surface, T is the surviving psame *transport row, V is the refusal verdict, H is the componentwise stability row, C is the displayed consumer route, P is package provenance, and N is the local* NameCert *row.*

Proof. Project the carrier of Definition 634.1. Its displayed tuple is the finite row list E, A, T, V, H, C, P, N.

Use Definition 634.2. The local certificate accounts for those rows as relation exposure, representative-request refusal, classifier-respecting transport, verdict accounting, componentwise stability, continuation, provenance, and naming.

No further quotient carrier row is present in the tuple. Thus every accepted packet exposes exactly the listed quotient-refusal carrier rows. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary carrier exposure`

Theorem 634.17 (QuotientSoundnessBoundary refusal ledger)**.** *Every representative-selection request accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet is ledgered by the rows A, V before the* psame *transport row T is available to a consumer.*

Proof. Use refusal locality, Definition 634.7. It keeps the requested representative surface inside the local refusal rows rather than in a host quotient selector.

Apply fixed classifier and the root request triad, Definition 634.12 and Definition 634.14. They place the demand in A and the negative verdict in V as part of the root request surface.

Only after that surface is visible can the transport replacement theorem supply the surviving psame row T. Hence the request is ledgered by A, V before T is consumed. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryRefusalLedger`

Theorem 634.18 (QuotientSoundnessBoundary downstream nonescape)**.** *Every downstream quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet receives only E, T, H, C, P, N after refusal has been accounted for. It receives no* Quot.sound*, host quotient eliminator, representative selector, or erased derivation recovery route.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream nonescape`

Proof. Begin with the refusal ledger above. The rows A, V account for the refused representative request before any surviving downstream route is used.

Use psame transport exhaustion, Definition 634.8. After refusal, the positive movement available to a quotient-like consumer is exhausted by the psame transport row T together with stability, continuation, provenance, and naming.

Finally apply kernel non-escape, Definition 634.13. It excludes Quot.sound, host quotient eliminators, representative selectors, and erased derivation recovery, so the downstream surface is only E, T, H, C, P, N once refusal has been recorded. □

Theorem 634.19 (QuotientSoundnessBoundary consumer route certificate)**.** *After the representative-selection refusal has been accounted for, any consumer route through the displayed rows H, C and package row P carries a* SemanticNameCert *surface for the surviving* psame *replacement. The certificate reads only the quotient-refusal carrier, the displayed relation row E, the replacement transport row T, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and the consumer row. It exports no representative selector, quotient eliminator, erased derivation recovery, host quotient recursion, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer route certificate`

Proof. Begin with the refusal ledger and downstream nonescape theorem above. They account for the negative request through A, V and leave the surviving consumer surface at E, T, H, C, P, N rather than at a representative selector or host quotient eliminator.

Next use the displayed route rows. The Cont route through H, C and the package row P are exactly the inputs of the checked consumer route certificate, while the carrier supplies the visible relation evidence and the surviving psame replacement row.

The checked Lean target then packages this route as a SemanticNameCert surface for the consumer row. Its fields use classifier-respecting transport, continuation replay, package provenance, and unary ledger soundness, so no quotient recursion, erased derivation recovery, representative selection, quotient eliminator, or Quot.sound dependency is imported. □

Theorem 634.20 (QuotientSoundnessBoundary root request accountability)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet $Q = (E, A, T, V, H, C, P, N)$, relation rows E, representative-request rows A, and refusal verdict rows V are accountable before any replacement by the transport row T. A downstream quotient-style consumer therefore sees the requested relation evidence, the demanded representative surface, and the refusal verdict before it can use T, H, C, P, N as the surviving audit route.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root request accountability`

Proof. Begin with Definition 634.15. It places the visible relation rows E before any representative-demand rows in A.

Use the root request triad, Definition 634.14. The triad binds the representative demand to A and its refusal to V, so the negative verdict is part of the root request surface.

After those rows are visible, the transport replacement theorem supplies T only as the classifier-respecting surviving route. The stability, locality, provenance, and naming rows H, C, P, N audit that route and do not erase E, A, V. □

Theorem 634.21 (QuotientSoundnessBoundary transport gate)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet $Q = (E, A, T, V, H, C, P, N)$, any quotient-like operation that survives the refused representative request must pass through T, H, C, P, N. The operation may still read the relation rows E, requested-representative rows A, and refusal verdict rows V as its accountable source surface, but its positive movement is gated by the displayed transport, stability, locality, provenance, and naming rows.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary transport gate`

Proof. First read the accountability surface from Definition 634.20. It keeps E, A, V visible before any surviving positive route is consumed.

Apply the transport replacement theorem. The only admitted positive motion after refusal is the psame transport row T on named source rows.

Finally use the NameCert obligations and non-escape theorem. They force every use of T to remain packaged with H, C, P, N and exclude any hidden positive path outside the displayed packet. □

Theorem 634.22 (QuotientSoundnessBoundary downstream nonimport)**.** *Every downstream quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$ *receives only the audit surface* E, A, T, V, H, C, P, N*. It receives no* Quot.sound *dependency, quotient carrier, representative selector, erased derivation recovery map, or host quotient eliminator.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryCarrier downstream nonimport`

Proof. Use root request accountability to expose E, A, V as the complete negative request surface. These rows record the visible relation evidence, the representative demand, and the refusal verdict before any surviving operation is available.

Next apply the transport gate. It confines every surviving positive operation to T, H, C, P, N, with T carrying classifier-respecting psame transport and the remaining rows carrying stability, locality, provenance, and local naming.

Close with kernel non-escape. Since the carrier has no coordinate for Quot.sound, quotient carriers, representative selectors, erased derivation recovery, or host quotient eliminators, the downstream consumer imports only the displayed audit surface. □

Theorem 634.23 (QuotientSoundnessBoundary root transport verdict order)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$*, the refusal verdict row* V *must be read before any positive* psame *transport row* T *is used by a quotient-like consumer. The transport row is therefore a replacement after refusal, not evidence that a representative selector or quotient equality eliminator has been admitted.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root transport verdict order`

Proof. Begin with refusal locality. It localizes every representative-selection demand to A and the refusal verdict V before the transport replacement is consumed.

Apply the transport replacement theorem. The row T is the classifier-respecting psame route on named source rows after the request has been refused, with H, C, P, N recording stability, locality, provenance, and naming.

Use the psame transport exhaustion theorem. After A, V are read, the surviving operation is exhausted by E, T, H, C, P, N and cannot recover a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, or Quot.sound dependency. □

Theorem 634.24 (QuotientSoundnessBoundary root downstream unblock surface)**.** *The root relation row exposure, the root transport verdict order, the transport replacement theorem, non-escape, and the kernel refusal ledger assemble the downstream unblock surface for quotient-like chapters. The exported surface is exactly* E, A, V, T, H, C, P, N*: relation rows, representative-demand rows, refusal verdict, classifier-respecting* psame *transport, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming. It explicitly excludes* Quot.sound *and host quotient eliminators.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root downstream unblock surface`

Proof. Use Definition 634.15. It forces every root quotient-style read to expose E before any representative-demand row in A can be consumed.

Use Definition 634.23. It places the refusal verdict V before the positive transport row T, so downstream use receives replacement transport only after the refused request is visible.

Finally combine Definition 634.3, Definition 634.6, and Definition 634.44. These rows assemble E, A, V, T, H, C, P, N as the finite consumer surface and rule out Quot.sound, host quotient eliminators, hidden representative selectors, and erased derivation recovery. □

Theorem 634.25 (QuotientSoundnessBoundary psame replacement classifier)**.** *After the refusal pair A, V has been read, the only accepted classifier-respecting route for a quotient-like consumer is the displayed path through E, T, H, C, P, N. The row T is the* psame *transport replacement on named source rows, and the packet supplies no quotient carrier, selected representative, quotient equality eliminator, erased relation-derivation recovery, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame replacement classifier`

Proof. Begin with the refusal pair supplied by Definition 634.14. The pair A, V has already recorded and refused the representative-selection demand before the positive consumer route is read.

Apply transport replacement, Definition 634.3, together with the fixed classifier theorem, Definition 634.12. The consumer route is forced through the relation rows E, the psame replacement row T, and the stability and locality rows H, C.

Use psame transport exhaustion, Definition 634.8. It adds the provenance and naming rows P, N to the same displayed route and excludes a second path through quotient carriers or selected representatives. □

Theorem 634.26 (QuotientSoundnessBoundary root transport package totality)**.** *After the root request triad has exposed E, A, V and the verdict-before-transport order has placed V before any positive transport use, every surviving quotient-like consumer route is totally packaged by E, T, H, C, P, N. The package reads the displayed relation rows, the classifier-respecting* psame *transport row, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming, and it imports no quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root transport package totality`

Proof. First expose the root request surface. By Definition 634.14 and Definition 634.15, every root quotient-style request reads the relation rows E together with the representative-demand rows A and refusal verdict V.

Next use Definition 634.23. It forces the refusal verdict to be visible before the positive transport row T is consumed.

Once the verdict is fixed, apply Definition 634.25. The surviving classifier-respecting route is exactly E, T, H, C, P, N.

Finally combine Definition 634.6 with Definition 634.8. These results rule out the quotient-side primitives and package the whole surviving route in the displayed transport rows. □

Theorem 634.27 (QuotientSoundnessBoundary transport-refusal normal form)**.** *After the refusal pair A, V has been read in an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, every surviving quotient-like operation is normalised to the displayed E, T, H, C, P, N route: relation rows E, classifier-respecting* psame *transport T, componentwise* hsame *stability H, displayed* Cont *locality C,* Pkg *provenance P, and local* NameCert *naming N. No surviving read exports a quotient carrier, representative selector, host quotient eliminator, erased relation-derivation recovery map, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary transport refusal normal form`

Proof. Start with the refused request. The root request triad Definition 634.14 exposes the relation rows E, the representative-style demand A, and the refusal verdict V.

Use Definition 634.23 to place V before any positive transport read. Thus a later transport row cannot erase or reinterpret the refusal pair.

Now apply Definition 634.25 and Definition 634.26. They identify every surviving classifier-respecting route with the displayed E, T, H, C, P, N transport package.

Finally use Definition 634.13 and the public bridge non-escape surface of Definition 634.52. The excluded quotient primitives have no exported coordinate, so the displayed transport route is the normal form. □

Theorem 634.28 (QuotientSoundnessBoundary dependent-eliminator refusal)**.** *Every downstream dependent eliminator, representative selector, or family-indexed quotient read requested from an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet is refused unless it is routed entirely through*

$$E, T, H, C, P, N.$$

The accepted route can transport named relation rows by psame*, componentwise* hsame *stability, displayed* Cont *locality, package provenance, and local naming, but it cannot export a dependent quotient eliminator, selected representative, erased relation-derivation recovery map, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary dependent eliminator refusal certificate`

Proof. Start with the carrier and refusal locality rows. The root request triad exposes E, A, V, and the non-escape surface records that representative selection and quotient eliminator requests are refused before any positive transport is consumed.

Use the transport-refusal normal form. Definition 634.27 states that every surviving quotient-like operation is normalised to E, T, H, C, P, N after A, V have been read.

Finally apply transport exhaustion and the public bridge non-escape surface. They leave only psame replacement transport over named source rows, with no coordinate for dependent quotient elimination, representative selection, erased derivation recovery, or Quot.sound. □

Theorem 634.29 (QuotientSoundnessBoundary classifier-substitution barrier)**.** *Any downstream substitution of classifiers for an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet must preserve the displayed relation rows E, the refusal pair A, V, and the positive transport package T, H, C, P, N. Such substitution may replace only along the displayed* psame *route; it cannot import* Quot.sound*, a quotient equality eliminator, a representative selector, or a hidden classifier over erased relation witnesses.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary classifier substitution barrier`

Proof. Read the fixed classifier row first. Definition 634.12 binds classifier-respecting use to the displayed relation rows rather than to an ambient quotient equality.

Next use refusal locality and verdict ordering. The request triad and Definition 634.23 keep A, V visible before any positive transport, so substitution cannot hide the refused demand.

Finally use transport replacement and normal form. Definition 634.3 and Definition 634.27 identify the surviving route as psame transport through E, T, H, C, P, N. Therefore classifier substitution preserves the displayed rows and imports no quotient-side primitive. □

Theorem 634.30 (QuotientSoundnessBoundary transport-only downstream package)**.** *The only downstream package exported by* QuotientSoundnessBoundary$^{\uparrow}$ *to quotient-like chapters is the transport-only package consisting of the refusal ledger E, A, V, the* psame *replacement row T, componentwise stability H, displayed locality C, package provenance P, local naming N, dependent-eliminator refusal, and classifier-substitution barrier. It supplies transport over named relation rows and no quotient carrier, quotient eliminator, representative selector, erased derivation recovery, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary transport only downstream package`

Proof. First assemble the refusal ledger. The root request triad exposes E, A, V, while transport verdict ordering keeps the refusal verdict visible before the positive row T is consumed.

Next assemble the positive transport surface. The transport replacement, psame exhaustion, root transport package totality, and transport-refusal normal form theorems identify every surviving route with E, T, H, C, P, N.

Finally attach the two downstream barriers. Definition 634.28 refuses dependent eliminators and representative selectors outside the transport route, and Definition 634.29 blocks classifier substitutions that import quotient-side primitives. The resulting package is transport-only. □

Theorem 634.31 (QuotientSoundnessBoundary consumer ledger coverage)**.** *Every downstream consumer read of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet is covered by the finite ledger rows E, A, V, T, H, C, P, N. The rows A, V record the refused representative-style request, while E, T, H, C, P, N record the surviving classifier-respecting* psame *transport route. The ledger exports no* Quot.sound *row and no host quotient eliminator.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer ledger coverage`

Proof. Read the negative coverage first. By Definition 634.14 and Definition 634.15, each root quotient-style request is visible through E, A, V.

Read the positive coverage next. The psame replacement classifier and root transport package totality, Definition 634.25 and Definition 634.26, cover every surviving transport consumer by E, T, H, C, P, N.

It remains to exclude hidden exits. Apply Definition 634.27, Definition 634.13, and Definition 634.52. The combined ledger therefore covers downstream consumers without exporting Quot.sound or a host quotient eliminator. □

Theorem 634.32 (QuotientSoundnessBoundary downstream handoff readiness)**.** *The transport-refusal normal form and consumer ledger coverage assemble the exact package that downstream quotient-like chapters may cite. That package is the finite surface E, A, V, T, H, C, P, N with refusal before transport, classifier-respecting* psame *replacement, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming; it contains no quotient primitive, no host quotient eliminator, no representative-selection export, and no* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream handoff readiness`

Proof. First take the ordered refusal surface from Definition 634.14 and Definition 634.23. These rows make A, V visible before any surviving transport handoff.

Next take the normal transport package from Definition 634.27. It identifies the only surviving quotient-like operation as the displayed E, T, H, C, P, N route.

Now add ledger coverage by Definition 634.31. Every downstream consumer read is therefore inside E, A, V, T, H, C, P, N.

Finally combine kernel non-escape with public bridge non-escape. The handoff surface is ready for downstream citation precisely because no quotient primitive, host quotient eliminator, representative-selection export, or Quot.sound dependency crosses the package boundary. □

↪ variant `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer route certificate`

Theorem 634.33 (QuotientSoundnessBoundary no hidden eliminator)**.** *After the representative-demand rows A and refusal verdict row V have been read, every accepted downstream consumer of a* QuotientSoundnessBoundary$^{\uparrow}$ *packet has only the displayed* psame *route*

$$E, T, H, C, P, N.$$

The consumer cannot recover a quotient eliminator, representative selector, erased derivation map, host quotient recursion principle, or Quot.sound *dependency from any later transport, replay, provenance, or naming row.*

Proof. First read consumer ledger coverage. It places every downstream quotient-like consumer inside the finite rows E, A, V, T, H, C, P, N and separates refused representative-style requests from the surviving classifier-respecting transport route.

Next use downstream handoff readiness. Once the rows A, V have recorded refusal, the only positive handoff available to a consumer is the displayed E, T, H, C, P, N route with psame transport, componentwise hsame stability, displayed Cont locality, Pkg provenance, and local naming.

Apply the root nonimport ledger and the public non-escape boundary. These results remove quotient eliminators, representative selectors, erased derivation recovery, host quotient recursion, and Quot.sound from the packet boundary, so no later structural row can reintroduce one.

Thus every accepted downstream consumer remains on the visible refusal-before-transport route and has no hidden quotient eliminator to recover. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary no hidden eliminator`

Theorem 634.34 (QuotientSoundnessBoundary refusal transport sibling handoff)**.** *Every downstream quotient-like sibling that consumes a* QuotientSoundnessBoundary$^{\uparrow}$ *packet must read the representative-demand row A and refusal verdict row V before reading the positive transport rows E, T, H, C, P, N. The handoff supplies only classifier-respecting* psame *transport, componentwise*

hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming over the finite* BHist *packet; it supplies no quotient carrier, host quotient eliminator, representative-selection export, or* Quot.sound *dependency.*

Proof. First read the refusal surface. The root request triad exposes the relation rows E, representative-style demand rows A, and refusal verdict rows V, while the verdict-order theorem places V before any positive transport row is available.

Next read the surviving transport surface. The transport-refusal normal form and downstream handoff readiness identify the only exported positive route as E, T, H, C, P, N, with T carrying classifier-respecting psame transport and H, C, P, N carrying hsame, Cont, Pkg, and NameCert structure.

Finally apply consumer ledger coverage and the public non-escape boundary. A downstream sibling therefore receives refusal before transport and cannot cite the packet as a source of Quot.sound, quotient carriers, host eliminators, or representative-selection exports. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal transport sibling handoff`

Theorem 634.35 (QuotientSoundnessBoundary socket-kind route)**.** *Every accepted* SocketKindClassifier$^{\uparrow}$ *quotient-kind read that cites a* QuotientSoundnessBoundary$^{\uparrow}$ *packet factors through the quotient boundary refusal-before-transport route. It first reads the representative-demand and refusal rows A, V, then reads the positive* psame *transport package E, T, H, C, P, N, and only then supplies the* SocketKindClassifier$^{\uparrow}$ *kind row from chapter 1058. The route cannot reinterpret the quotient boundary packet as* Quot.sound*, a quotient carrier, a host quotient eliminator, or a representative selector.*

Proof. Begin with the sibling handoff, Definition 634.34. It forces every downstream quotient-like sibling to receive the refusal rows A, V before it can use the positive transport surface, and it exports only classifier-respecting psame transport with hsame, Cont, Pkg, and NameCert structure.

Next apply representative-demand exposure. The demanded representative surface remains audit-visible inside the finite BHist packet, so the classifier read cannot treat A as an admitted selector or erase the refusal verdict V before the kind row is assigned.

Finally use consumer ledger coverage and the downstream classifier surface in chapter 1058. The ledger exhausts quotient-like consumption by the refusal rows and the displayed transport rows, while the classifier contributes only the socket-kind row. Thus the cited quotient boundary is consumed as a finite BEDC refusal kind, not as Quot.sound, a quotient carrier, a representative selector, or a host quotient principle. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary socket kind route`

Theorem 634.36 (QuotientSoundnessBoundary representative demand exposure)**.** *Under the sibling handoff of Definition 634.34, the representative-demand rows A remain audit-visible as part of the finite* BHist *ledger. Later reads of T, H, C, P, N may consume classifier-respecting* psame *transport, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming, but they cannot erase A or reread it as an admitted representative selector.*

Proof. Begin with refusal locality. The representative-style demand is recorded in A together with the refusal verdict V before the positive transport route is available.

Now apply the sibling handoff. It carries A, V forward as the visible negative part of the surface and only then exposes the positive route through E, T, H, C, P, N.

It remains to exclude erasure by later rows. Kernel non-escape and public bridge non-escape remove the quotient-side coordinates, so T, H, C, P, N have no operation by which they can hide A, recover an erased representative demand, or turn the refused demand into an admitted representative-selection export. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary representative demand exposure`

Theorem 634.37 (QuotientSoundnessBoundary refusal ledger consumer frontier)**.** *Every downstream consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet that reaches the representative-demand rows must first pass through the visible refusal rows E, A, V. Any later positive route through T, H, C, P, N remains classifier-respecting* psame *transport with componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming rather than a quotient eliminator.*

Proof. Start with representative-demand exposure, Definition 634.36. It keeps the requested representative rows A audit-visible after sibling handoff and prevents later reads from treating them as admitted representative selectors.

Use psame consumer normalization, Definition 634.39. Once the refusal rows have been read, the surviving positive route normalizes to E, T, H, C, P, N and the transport row T is a classifier-respecting psame route.

Apply downstream factorization exactness and two-surface exhaustion. Definition 634.41 separates downstream reads into the refused request surface or the displayed transport surface, while Definition 634.42 rules out a third quotient-side route.

Therefore a consumer that reaches the representative-demand rows has already crossed E, A, V, and any later positive use of T, H, C, P, N is only the displayed replacement transport. It cannot be read as Quot.sound, a quotient carrier eliminator, or a hidden representative-selection principle. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal ledger consumer frontier`

Theorem 634.38 (QuotientSoundnessBoundary refusal consumer public exactness)**.** *A public downstream quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet is accepted exactly when it factors through the visible refusal rows E, A, V followed by the positive* psame *transport package*

$$E, T, H, C, P, N.$$

The public route has no third branch through quotient carrier data, representative selection, a host quotient eliminator, erased derivation recovery, unrestricted quotient recursion, or Quot.sound*.*

Proof. For the forward direction, begin with the refusal ledger consumer frontier, Definition 634.37. Any consumer that reaches the representative-demand rows must pass through E, A, V, and any later positive use remains displayed transport through T, H, C, P, N.

Next apply psame consumer normalization, Definition 634.39. After the refusal rows are read, the surviving positive route is the classifier-respecting psame package E, T, H, C, P, N.

For the reverse direction, use downstream factorization exactness, Definition 634.41. A route that factors through the refusal surface and then through the displayed psame package is precisely a downstream quotient-like consumer route inside the accepted packet.

Finally apply two-surface exhaustion, Definition 634.42. The accepted packet has only the refusal surface and the positive transport surface, so quotient carrier data, representative selection, host quotient elimination, erased derivation recovery, unrestricted quotient recursion, and Quot.sound have no third route into the public consumer surface. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal consumer public exactness`

Theorem 634.39 (QuotientSoundnessBoundary psame consumer normalization)**.** *Every surviving quotient-like consumer route of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet normalizes, after the refusal rows A, V have been read, to the displayed transport surface E, T, H, C, P, N. The normal form is a* psame *consumer route with componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and local* NameCert *naming; it has no quotient carrier coordinate and no host eliminator coordinate.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame consumer normalization`

Proof. Read the classifier first. The psame replacement classifier identifies the positive consumer operation as the displayed transport row T over the named relation rows E.

Then use consumer ledger coverage. Once refusal has been recorded in A, V, every surviving consumer read is covered by the positive rows E, T, H, C, P, N and by no additional quotient-side row.

Finally apply the sibling handoff together with kernel non-escape. The surviving consumer route therefore normalizes to the displayed psame transport route and cannot pass through a quotient carrier, a host quotient eliminator, or a hidden Quot.sound dependency. □

Theorem 634.40 (QuotientSoundnessBoundary refusal verdict monotonicity)**.** *In an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, adding later transport rows cannot turn a refused representative-selection row A, V into an admitted quotient primitive. The rows E, T, H, C, P, N may supply the classifier-respecting* psame *replacement route after refusal, but they cannot reread A, V as a representative selector, quotient equality eliminator, host quotient eliminator, unrestricted quotient recursion principle, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal verdict monotonicity`

Proof. Start with the negative row. The root request triad, Definition 634.14, records representative-selection demand in A and refusal in V before transport is available.

The order statement Definition 634.23 separates verdict from transport. Thus later use of T is downstream of the refused request rather than a replacement for the refusal verdict.

Use Definition 634.25 for the positive route. It gives only classifier-respecting psame transport through E, T, H, C, P, N.

Finally apply Definition 634.6 and Definition 634.44. The packet has no coordinate by which later transport rows can promote a refused representative request into an admitted quotient primitive. □

Theorem 634.41 (QuotientSoundnessBoundary downstream factorization exactness)**.** *Every downstream quotient-like consumer of* QuotientSoundnessBoundary$^{\uparrow}$ *factors through exactly one of the two displayed surfaces: either the refused request pair A, V or the* psame *transport package E, T, H, C, P, N. There is no third route through a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, host-level class selector, unrestricted quotient recursion principle, or* Quot.sound *dependency.*

Proof. Read the negative surface first. By Definition 634.14, Definition 634.15, and Definition 634.23, a representative-style request must expose E, then read A, V, before any surviving transport can be used.

Read the positive surface next. By Definition 634.25 and Definition 634.26, every surviving classifier-respecting consumer route is packaged by E, T, H, C, P, N.

The classifier boundary is exact because Definition 634.6 removes the quotient-side primitives, while Definition 634.44 records the exported surface as refusal plus displayed transport.

Finally apply Definition 634.25 again to identify the positive side as the named psame replacement rather than a hidden quotient route. The downstream consumer therefore has only the refused pair or the displayed transport package. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream factorization exactness`

Theorem 634.42 (QuotientSoundnessBoundary two-surface exhaustion)**.** *Every downstream quotient-like consumer route for an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet is exhausted by exactly the refused request rows A, V and the classifier-respecting* psame *transport rows*

$$E, T, H, C, P, N.$$

There is no hidden third route through a quotient carrier, representative selector, quotient eliminator, unrestricted quotient recursion principle, erased derivation recovery, or Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary two surface exhaustion`

Proof. Read the negative request surface from the root request triad, Definition 634.14. A representative-style demand is recorded as the requested rows A together with the refusal verdict V.

Read the positive replacement surface from Definition 634.25. The only surviving operation is classifier-respecting psame transport through the displayed relation and structural rows E, T, H, C, P, N.

Apply consumer ledger coverage and downstream factorization exactness. The ledger covers all downstream quotient-like reads, and Definition 634.41 separates them into the refused request surface or the displayed transport surface.

Finally invoke kernel non-escape. Since the accepted packet has no coordinate for quotient carriers, representative selectors, quotient eliminators, unrestricted quotient recursion, erased derivation recovery, or Quot.sound, the two displayed surfaces exhaust the downstream route space. □

Theorem 634.43 (QuotientSoundnessBoundary consumer refusal exactness)**.** *Every downstream quotient-like consumer preserving an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *finite packet must remain on exactly one of two surfaces: either the refused request rows A, V, or the positive classifier-respecting* psame *transport package*

$$E, T, H, C, P, N.$$

No later consumer may reinterpret the refusal rows as representative selection, quotient equality elimination, host quotient elimination, unrestricted quotient recursion, erased derivation recovery, or a Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer refusal exactness`

Proof. First read consumer ledger coverage. The ledger records all downstream quotient-like reads before any consumer can choose a route, so a preserving consumer must use one of the rows exposed by the finite packet.

Next expose representative demand. The root request triad and representative demand exposure place representative-style requests on the visible request and verdict rows A, V before transport is available.

Now read the positive route. The psame consumer normalization theorem and the downstream factorization exactness theorem package surviving classifier-respecting reads as E, T, H, C, P, N and separate them from the refused request surface.

Finally apply two-surface exhaustion. Since the accepted packet has only the refused request surface and the displayed psame transport surface, a later consumer has no coordinate from which to turn refusal into representative selection or any hidden quotient principle. □

Theorem 634.44 (QuotientSoundnessBoundary kernel refusal ledger)**.** *The root request triad, the* psame *replacement classifier, and kernel non-escape assemble the kernel-facing refusal ledger consumed by downstream quotient-like chapters. The exported surface consists of the refused request rows E, A, V together with the classifier-respecting transport rows E, T, H, C, P, N, and no* Quot.sound *or host quotient eliminator enters that surface.*

Proof. Use Definition 634.14 for the negative part of the ledger. It binds each root quotient-style request to the visible relation evidence E, the requested representative rows A, and the refusal verdict V.

Use Definition 634.25 for the positive replacement route. It identifies the accepted classifier-respecting consumer path as E, T, H, C, P, N, with T carrying the displayed psame transport.

Finally apply Definition 634.13. The kernel-facing packet cannot project Quot.sound, a host quotient eliminator, a hidden representative selector, or erased relation-derivation recovery, so downstream quotient-like chapters receive refusal plus displayed transport rather than a hidden quotient principle. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal transport surface certificate`

Theorem 634.45 (QuotientSoundnessBoundary NameCert ledger)**.** *The carrier of Definition 634.1, the fixed classifier of Definition 634.12, the transport replacement of Definition 634.3, the negative-refusal dependency of Definition 634.4, and the kernel non-escape theorem of Definition 634.13 assemble the local* $\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}$ *ledger.*

Proof. The carrier supplies the finite inventory E, A, T, V, H, C, P, N. The fixed classifier theorem binds every accepted comparison to the displayed relation, request, replacement, verdict, and transport rows.

The transport replacement theorem supplies the positive operation available to downstream consumers. It routes classifier-respecting reads through named source rows rather than through selected quotient representatives.

The negative-refusal dependency and kernel non-escape theorems complete the ledger. They record the blocked quotient-soundness demand as a negative boundary request and keep every exported consumer inside E, A, T, V, H, C, P, N, so the local naming certificate contains exactly the quotient-boundary obligations listed in Definition 634.2. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary namecert ledger`

Theorem 634.46 (QuotientSoundnessBoundary public bridge)**.** *Every downstream quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet must read the public boundary through the finite rows*

$$E, A, T, V, H, C, P, N.$$

The fixed classifier, transport replacement, negative-refusal dependency, kernel non-escape theorem, and NameCert ledger make this the whole public bridge: it admits relation rows, requested-representative refusal, classifier-respecting psame *transport, componentwise stability, continuation, provenance, and local naming, but it excludes quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery, and* Quot.sound*.*

Proof. Project the carrier and apply the fixed classifier theorem. Every accepted comparison is already read through E, A, T, V, H, C, so no downstream consumer can widen the comparison by consulting a quotient carrier or representative selector.

Use transport replacement for the positive route. It routes quotient-like use through the source relation rows E, the classifier-respecting transport row T, and the structural rows H, C, P, N.

Use the negative-refusal dependency for blocked representative requests. The rows A, V record those requests as refused boundary reads before any surviving transport is consumed.

Finally apply kernel non-escape and the NameCert ledger. Kernel non-escape removes quotient equality eliminators, erased derivation recovery, and Quot.sound from the exported packet, while the ledger assembles the remaining rows as the finite public boundary. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryPublicBridge`

Theorem 634.47 (QuotientSoundnessBoundary bridge schema)**.** *Every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *carrier presents a bridge from the finite refusal-and-transport packet to the public quotient-refusal schema. The bridge reads the relation rows E, requested representative rows A, classifier-respecting* psame *transport row T, refusal verdict V, componentwise* hsame *transport row H, continuation rows C,* Pkg *provenance row P, and local* NameCert *row N, and it imports no* Quot.sound *dependency.*

Proof. Project the carrier of Definition 634.1. Its public quotient-facing surface is already the finite tuple E, A, T, V, H, C, P, N.

Use the NameCert ledger, Definition 634.45, to assemble the refusal row A, V and the transport route E, T, H, C, P, N as one public packet. The fixed classifier theorem keeps comparison reads inside the same rows.

Finally apply non-escape, Definition 634.6. Since the excluded quotient primitives and Quot.sound have no coordinate in the carrier, the bridge is exactly the displayed refusal-and-transport schema. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary public bridge schema`

Theorem 634.48 (QuotientSoundnessBoundary psame bridge handoff)**.** *After the public refusal row A, V has been read, the only bridge handoff available to a quotient-like consumer is the classifier-respecting* psame *route E, T, H, C, P, N. The handoff supplies named-source transport and does not supply a representative selector, quotient equality eliminator, erased derivation recovery map, or* Quot.sound *dependency.*

Proof. Start with the refusal-locality theorem, Definition 634.7. It places every representative-selection demand in A, V before the positive transport route is consumed.

Then use psame transport exhaustion, Definition 634.8. The surviving operation is exhausted by E, T, H, C, P, N, with T carrying classifier-respecting transport on named source rows.

The non-escape theorem removes the quotient-side objects from the handoff. Thus the bridge gives downstream consumers a displayed psame route and no quotient-soundness primitive. □

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame bridge handoff

Theorem 634.49 (QuotientSoundnessBoundary refusal-transport determinacy)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet*

$$Q = (E, A, T, V, H, C, P, N),$$

once the representative request recorded in A has been refused by V, any accepted quotient-like consumer route is determined by the displayed psame *route through E, T, H, C, P, N. The consumer cannot choose a second route through a quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, host-level class selector, unrestricted quotient recursion principle, or* Quot.sound *dependency.*

Proof. Begin with the carrier of Definition 634.1. Its rows display the relation evidence E, representative requests A, psame transport T, refusal verdict V, componentwise stability H, continuation routes C, provenance P, and local naming N.

Use refusal locality, Definition 634.7. It forces the representative-selection demand to be read as the local pair A, V before any positive transport is consumed.

After that refusal, apply psame transport exhaustion, Definition 634.8, together with the bridge handoff, Definition 634.48. These results identify E, T, H, C, P, N as the displayed consumer route after refusal.

Finally use kernel non-escape, Definition 634.13, and the public bridge, Definition 634.46. Since the excluded quotient primitives have no coordinate in the packet, the accepted consumer has no second quotient route to select. □

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal transport determinacy

Theorem 634.50 (QuotientSoundnessBoundary source-row consumer totality)**.** *Every surviving quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet must expose the source relation rows E before it reads the positive transport row T. All consumer-facing data in the packet is exhausted by the finite rows*

$$E, A, T, V, H, C, P, N.$$

Proof. Project the carrier of Definition 634.1. The row E is the source relation surface, while T is the classifier-respecting psame transport replacement available only as part of the displayed packet.

Apply root relation row exposure, Definition 634.15, and root transport verdict order, Definition 634.23. These force relation exposure and refusal to be visible before the positive transport row can be consumed.

Then use psame transport exhaustion and the bridge handoff. The surviving consumer route is the single path through E, T, H, C, P, N after the refusal pair A, V has been read.

Kernel non-escape and the public bridge complete the totality claim. They keep every consumer-facing read inside E, A, T, V, H, C, P, N and exclude quotient carriers, hidden representative selectors, erased derivation recovery, host quotient eliminators, and Quot.sound. □

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary source row consumer totality

Theorem 634.51 (QuotientSoundnessBoundary verdict-transport separation)**.** *In every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$*, the row* V *is a negative refusal row for the representative request in* A*, while* T *is a positive classifier-respecting* psame *transport row on the source relation surface* E*. Neither row can be reread as the other, and neither row can be reread as* Quot.sound*.*

Proof. The carrier separates the roles at the level of rows. The row V records the refusal verdict for the requested representative surface in A, while T records the positive psame transport replacement over the relation rows E.

Use refusal locality to read A, V as the negative part of the packet before transport. Use psame transport exhaustion and the bridge handoff to read E, T, H, C, P, N as the positive route made available after that refusal.

The two readings therefore have different positions and different consumer roles inside the finite packet. The verdict row blocks representative selection; the transport row carries classifier-respecting named-source transport.

Finally apply kernel non-escape and the public bridge non-escape, Definition 634.52. Since Quot.sound and the host quotient primitives are outside the public bridge, neither V nor T can be reinterpreted as a quotient-soundness primitive. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary verdict transport separation`

Theorem 634.52 (QuotientSoundnessBoundary public bridge non-escape)**.** *The public bridge supplied by* QuotientSoundnessBoundary$^{\uparrow}$ *is non-escaping: every consumer read remains inside* E, A, T, V, H, C, P, N*, and any request for quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery maps, host-level class selectors, unrestricted quotient recursion, or* Quot.sound *lies outside the bridge.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary public bridge nonescape`

Proof. Apply the bridge schema theorem above to identify the public bridge with the refusal-and-transport tuple E, A, T, V, H, C, P, N.

Apply the psame bridge handoff theorem above for the positive route after refusal. It leaves only named-source transport through E, T, H, C, P, N.

Finally apply Definition 634.13. The kernel-facing route has no quotient primitive coordinate, so any request for such an object must be supplied by a separate certificate outside this chapter. □

Theorem 634.53 (QuotientSoundnessBoundary erased derivation refusal)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet*

$$Q = (E, A, T, V, H, C, P, N),$$

any request to recover an erased quotient derivation is first routed to the representative-request row A *and the refusal-verdict row* V*. The request is refused before the positive transport row* T *is readable by a downstream consumer.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary erased derivation refusal certificate`

Proof. The carrier in Definition 634.1 separates representative requests, refusal verdicts, and positive transport into the rows A, V, T. A recovery demand for an erased derivation is therefore not a transport demand; it is a representative-side demand.

Apply refusal locality, Definition 634.7. It forces the demand through A, V before any consumer can use the positive transport route.

Then use verdict-transport separation, Definition 634.51. The refusal row V cannot be reread as the transport row T, and T cannot be used to reconstruct a denied representative witness.

Finally apply public bridge non-escape, Definition 634.52. Erased derivation recovery has no bridge coordinate, so the only public response is the local refusal recorded by A, V. □

Theorem 634.54 (QuotientSoundnessBoundary psame-only descent)**.** *After the refusal of representative recovery in an accepted* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$*, every accepted descent route factors only through* E, T, H, C, P, N *using classifier-respecting* psame *transport. No accepted descent route uses representative selection, quotient equality elimination, erased derivation recovery, host quotient recursion, or* Quot.sound*.*

Proof. Begin after the refusal supplied by Definition 634.53. At that point the consumer may read the positive route only as a route downstream of A, V.

Apply psame transport exhaustion, Definition 634.8. It identifies the surviving positive descent as classifier-respecting psame transport through E, T, H, C, P, N.

Use the psame bridge handoff, Definition 634.48, to pass this displayed route to the public bridge. Then apply source-row consumer totality, Definition 634.50, which exhausts the downstream read by the finite rows E, A, T, V, H, C, P, N.

The excluded quotient primitives have no coordinate in those rows. Hence an accepted descent route is psame-only and cannot pass through representative selection, quotient equality elimination, erased derivation recovery, host quotient recursion, or Quot.sound. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame only descent`

Theorem 634.55 (QuotientSoundnessBoundary consumer non-escape)**.** *Every downstream quotient-like consumer of an accepted* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet is exhausted by the finite rows*

$$E, A, T, V, H, C, P, N.$$

It cannot reinterpret the refusal verdict V *or the positive transport row* T *as quotient equality elimination, erased derivation recovery, host quotient recursion, or* Quot.sound*.*

Proof. First apply erased derivation refusal, Definition 634.53. Any recovery demand is already localized to A, V and refused before T is available.

Next apply psame-only descent, Definition 634.54. The surviving positive route factors through E, T, H, C, P, N as classifier-respecting psame transport.

Finally use public bridge non-escape, Definition 634.52. Since every consumer read stays inside E, A, T, V, H, C, P, N, neither V nor T can be reinterpreted as a quotient-side eliminator, recovery map, recursion principle, or Quot.sound witness. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer nonescape`

Theorem 634.56 (QuotientSoundnessBoundary representative request frontier)**.** *For every accepted* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$*, any request for a representative, representative selector, erased derivation, or host quotient witness is exhausted by the*

source row E, representative-request row A, and refusal-verdict row V before the positive transport row T is readable. The request therefore reaches a visible refusal frontier before any downstream psame *transport is exposed.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary representative request frontier`

Proof. Start with erased derivation refusal, Definition 634.53. It routes recovery demands to A, V before the positive transport row is available.

Apply refusal locality and verdict-transport separation, Definition 634.7 and Definition 634.51. These rows prevent a representative request from being reread as transport and prevent T from reconstructing a denied witness.

Finally use consumer non-escape, Definition 634.55. The public consumer remains inside E, A, T, V, H, C, P, N, so the representative-side request is exhausted at E, A, V before transport can be used. □

Theorem 634.57 (QuotientSoundnessBoundary psame consumer closure)**.** *After the representative frontier refuses the representative-side request, every surviving downstream consumer closes through E, T, H, C, P, N by classifier-respecting* psame *transport. The surviving route has no representative selector, quotient equality eliminator, erased derivation recovery map, host quotient recursion, or* Quot.sound *coordinate.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame consumer closure`

Proof. First apply the representative request frontier above. It leaves the positive route available only after E, A, V have already recorded the refusal boundary.

Use psame-only descent, Definition 634.54. The surviving route factors through E, T, H, C, P, N as classifier-respecting psame transport.

Then apply the public bridge non-escape theorem, Definition 634.52. The bridge has no coordinate for the excluded quotient primitives, so the psame-only consumer route is closed by the displayed finite rows. □

Theorem 634.58 (QuotientSoundnessBoundary Quot.sound refusal exactness)**.** *In an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, a request for* Quot.sound *is refused exactly at the visible boundary rows E, A, V and cannot be served by the positive transport row T or by the consumer rows H, C, P, N. Conversely, the admitted positive route through E, T, H, C, P, N is precisely the* psame-*only route that remains after that refusal.*

Proof. For the refusal direction, apply the representative request frontier above. A Quot.sound demand is a representative-side quotient primitive request, so it is routed to E, A, V before T is readable.

Apply kernel non-escape and public bridge non-escape, Definition 634.13 and Definition 634.52. Neither the kernel-facing route nor the bridge exposes a Quot.sound coordinate.

For the surviving route, use psame consumer closure above. Once the refusal is recorded, every admitted consumer is exactly the classifier-respecting psame route through E, T, H, C, P, N, so the refusal boundary and the positive route are exact complements inside the finite packet. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary quot sound refusal exactness`

Theorem 634.59 (QuotientSoundnessBoundary refusal stability row)**.** *In every accepted* QuotientSoundnessBound *packet $Q = (E, A, T, V, H, C, P, N)$, componentwise* hsame *transport through the history row H preserves the visible refusal pair A, V and the surviving transport row T. The stable row remains a refusal-and-*psame *ledger entry and does not admit representative selection, erased derivation recovery, host quotient recursion, or* Quot.sound*.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal stability row`

Proof. The carrier definition, Definition 634.1, makes H a row of the finite packet rather than a quotient eliminator. Hence componentwise hsame transport can only move along the displayed history coordinate of Q.

Apply the representative request frontier, Definition 634.56. It fixes the visible refusal boundary at E, A, V before the positive transport row T is available.

Then apply psame consumer closure, Definition 634.57. After the refusal boundary is recorded, the surviving consumer route is exactly the classifier-respecting psame route through E, T, H, C, P, N.

Finally use Quot.sound refusal exactness, Definition 634.58. Since Quot.sound is refused at E, A, V and the admitted route is the displayed psame route, hsame transport through H cannot create a hidden representative selector or quotient-soundness witness. □

Theorem 634.60 (QuotientSoundnessBoundary Cont route locality)**.** *Every consumer route through the continuation row C of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet reads the relation rows E, then the refusal rows A, V, and then the positive* psame *transport row T in the displayed order. There is no hidden route through a quotient eliminator, representative selector, erased derivation recovery map, host quotient recursion, or* Quot.sound*.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary cont route locality`

Proof. The carrier definition places C after the source, refusal, transport, and history rows in the finite packet $Q = (E, A, T, V, H, C, P, N)$. A continuation consumer therefore receives only the rows already exposed by the packet.

Use the representative request frontier, Definition 634.56, to locate every representative-side demand at E, A, V before any positive transport is readable.

Next apply psame consumer closure, Definition 634.57. It identifies the route available to C as the classifier-respecting psame route through E, T, H, C, P, N after the refusal frontier has been crossed.

Quot.sound refusal exactness, Definition 634.58, rules out the hidden branch. The continuation row has no coordinate that can bypass A, V or reinterpret T as Quot.sound. □

Theorem 634.61 (QuotientSoundnessBoundary Pkg NameCert ledger totality)**.** *In every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, the package row P and NameCert row N account for the whole finite ledger E, A, T, V, H, C. The exported object is exactly this ledger package and exports no quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery primitive, host quotient recursion principle, or* Quot.sound *primitive.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary pkg namecert ledger totality`

Proof. By the carrier definition, the public packet has exactly the displayed rows $Q = (E, A, T, V, H, C, P, N)$. The rows P, N package and certify the preceding ledger rows; they are not extra quotient primitives.

Apply the representative request frontier to account for the refusal ledger E, A, V. It records every representative-side request before positive transport is exposed.

Apply psame consumer closure to account for the positive ledger E, T, H, C, P, N. The downstream route that survives the refusal is the classifier-respecting psame route over the displayed packet.

Finally use Quot.sound refusal exactness. Since Quot.sound is refused exactly at the visible boundary and the surviving route is already exhausted by the packet rows, P, N are total for the finite ledger and cannot export any additional quotient primitive. □

Theorem 634.62 (QuotientSoundnessBoundary psame request determinacy)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet* $Q = (E, A, T, V, H, C, P, N)$*, a representative-side request is deterministically routed to the visible refusal rows* E, A, V *before the admitted* psame *route through* E, T, H, C, P, N *is readable. Thus a downstream consumer can read either the recorded refusal frontier or the surviving classifier-respecting* psame *transport, but it receives no ambiguous branch between representative access and positive transport.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame request determinacy`

Proof. Apply the representative request frontier, Definition 634.56. It sends every representative, selector, erased-derivation, host-quotient, and Quot.sound demand to E, A, V before T is exposed.

Next use psame consumer closure, Definition 634.57. After the refusal frontier is recorded, the only surviving positive consumer is the classifier-respecting psame route through E, T, H, C, P, N.

The two surfaces are separated by Quot.sound refusal exactness, Definition 634.58. Hence the packet presents a determined refusal surface first and a determined psame surface second, with no hidden branch that can choose a representative route. □

Theorem 634.63 (QuotientSoundnessBoundary NameCert refusal totality)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, the package row* P *and NameCert row* N *are total for the refusal-plus-transport ledger consisting of the visible refusal rows* E, A, V *and the admitted transport rows* E, T, H, C, P, N*. The certified export exposes no* Quot.sound *primitive, representative selector, erased-derivation recovery coordinate, or host quotient recursion coordinate.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary namecert refusal totality`

Proof. First apply psame request determinacy above. It fixes the order in which E, A, V are readable as refusal rows and E, T, H, C, P, N are readable as the positive psame route.

Use Cont route locality, Definition 634.60, to keep continuation consumers inside that displayed route. The continuation row cannot bypass A, V or reinterpret T as a quotient eliminator.

Finally apply Pkg NameCert ledger totality, Definition 634.61. Since P, N already certify the entire finite ledger and expose no excluded quotient primitive, the NameCert surface is total for exactly the refusal-plus-transport ledger stated here. □

Theorem 634.64 (QuotientSoundnessBoundary downstream representative non-escape)**.** *Every downstream consumer of the* QuotientSoundnessBoundary$^{\uparrow}$ *chapter factors through the deterministic refusal surface of Definition 634.62 and the total NameCert surface of Definition 634.63. Consequently the surviving* psame *route through* E, T, H, C, P, N *cannot be reinterpreted as a quotient eliminator, representative selector, erased-derivation recovery map, host quotient recursion principle, or* Quot.sound *witness.*

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream representative nonescape certificate

Proof. Apply psame request determinacy to determine the public request boundary. A downstream consumer first sees the visible refusal rows E, A, V and only then the classifier-respecting psame route through E, T, H, C, P, N.

Apply NameCert refusal totality above. The certified export is the finite refusal-plus-transport ledger, so its public rows expose no representative selector, recovery coordinate, host quotient recursion coordinate, or Quot.sound primitive.

Use Cont route locality and Pkg NameCert ledger totality, Definition 634.60 and Definition 634.61. They force every continuation and package read to stay inside the same finite ledger, so the surviving psame route is a consumer closure route rather than a quotient eliminator. □

Theorem 634.65 (QuotientSoundnessBoundary refusal exact complement)**.** *In an accepted* QuotientSoundnessBou *packet*

$$Q = (E, A, T, V, H, C, P, N),$$

every downstream consumer request is either the visible representative-side refusal through E, A, V or the admitted psame*-only route through E, T, H, C, P, N. These two surfaces are exact complements inside the finite packet: a request that asks for a representative, selector, erased-derivation recovery, host quotient recursion, quotient equality eliminator, or* Quot.sound *witness is refused at E, A, V, while a request that survives the refusal boundary factors through the classifier-respecting* psame *route and no third route remains.*

Proof. Apply the representative request frontier, Definition 634.56. It places every representative-side demand at the visible refusal surface E, A, V before a positive transport row can be consumed.

Use psame consumer closure, Definition 634.57. Once the representative-side request has been refused, the surviving consumer route is the classifier-respecting psame route through E, T, H, C, P, N.

Invoke Quot.sound refusal exactness, Definition 634.58. It separates the forbidden representative primitive from the admitted psame transport, so the refusal surface cannot be reinterpreted as a hidden quotient witness.

Finally apply NameCert refusal totality, Definition 634.63. The package and naming rows certify only the refusal-plus-transport ledger, hence the two surfaces exhaust the finite packet and form exact complements for downstream consumers. □

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal exact complement certificate

Theorem 634.66 (QuotientSoundnessBoundary consumer scope triad)**.** *Every downstream quotient-like consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet must first read the refusal surface E, A, V: the visible relation rows E, the representative-demand rows A, and the refusal verdict rows V. Only after that triad is visible may the consumer read any positive motion through the classifier-respecting* psame *route E, T, H, C, P, N. The handoff receives no quotient carrier, representative selector, erased derivation recovery map, host quotient recursion principle, or* Quot.sound *witness.*

Lean4 ✓ BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary consumer scope triad certificate

Proof. Use psame request determinacy, Definition 634.62. It routes every representative-side request to E, A, V before the positive psame route is readable.

Apply NameCert refusal totality, Definition 634.63. The certified export is the finite refusal-plus-transport ledger, so a downstream consumer reads the refusal triad before it receives the transport rows.

Finally use downstream representative non-escape above. It keeps the surviving route inside E, T, H, C, P, N and excludes quotient carriers, representative selectors, erased derivation recovery, host quotient recursion, and Quot.sound from the handoff. □

Theorem 634.67 (QuotientSoundnessBoundary psame consumer factorization)**.** *After the refusal surface E, A, V has been read, every surviving positive motion of a downstream quotient-like consumer factors only through the classifier-respecting* psame *package E, T, H, C, P, N. The factorization uses the displayed relation rows, transport replacement, componentwise* hsame *stability,* Cont *locality,* Pkg *provenance, and local* NameCert *naming, and it has no coordinate for a quotient carrier, representative selector, erased derivation recovery map, host quotient recursion principle, or* Quot.sound *witness.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary psame consumer factorization`

Proof. Begin with the consumer scope triad above. It places the downstream read at E, A, V before any positive transport row can be consumed.

Use psame consumer closure, Definition 634.57. Once the representative-side request is refused, every accepted consumer closes through the classifier-respecting route E, T, H, C, P, N.

Use Pkg NameCert ledger totality, Definition 634.61, to account for the package and naming rows in the same factorization.

Finally apply public bridge non-escape, Definition 634.52. The bridge exports no quotient carrier, representative selector, recovery coordinate, host quotient recursion, or Quot.sound witness, so the displayed psame package is the only surviving route. □

Theorem 634.68 (QuotientSoundnessBoundary downstream boundary readiness)**.** *The consumer scope triad and psame consumer factorization assemble the downstream boundary package for* QuotientSoundnessBoundary$^{\uparrow}$*. A downstream quotient-like chapter receives the refusal surface E, A, V first and then, for any surviving positive motion, the classifier-respecting* psame *route E, T, H, C, P, N; it receives no quotient carrier, representative selector, erased derivation recovery map, host quotient recursion principle, or* Quot.sound *witness.*

Proof. First apply the consumer scope triad. It fixes the downstream entry surface as E, A, V, with visible relation rows, requested representative rows, and refusal verdict rows read before positive transport.

Next apply psame consumer factorization above. It identifies every surviving positive motion with the displayed classifier-respecting route E, T, H, C, P, N.

Finally use downstream representative non-escape together with public bridge non-escape. These results prevent the boundary package from exporting quotient carriers, representative selectors, erased derivation recovery maps, host quotient recursion, or Quot.sound witnesses, so the chapter is ready as a downstream nonimport boundary. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream boundary readiness`

Theorem 634.69 (QuotientSoundnessBoundary downstream nonimport totality)**.** *Every downstream consumer of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet factors through the refused*

representative surface E, A, V and the admitted psame *transport surface E, T, H, C, P, N. This factorization is total for the downstream nonimport boundary: no downstream consumer imports quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery, host quotient recursion, or* Quot.sound.

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary downstream nonimport totality`

Proof. First apply public bridge non-escape, Definition 634.52. It keeps every consumer read inside E, A, T, V, H, C, P, N and gives no bridge coordinate for the excluded quotient primitives.

Next use erased derivation refusal and psame-only descent, Definition 634.53 and Definition 634.54. The representative-side demand is routed to E, A, V and refused before the admitted positive motion is read as classifier-respecting psame transport through E, T, H, C, P, N.

Finally cite the root nonimport ledger, Definition 634.79. The root ledger already packages the same refusal surface and positive transport surface without quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery, host quotient recursion, or Quot.sound, so the downstream consumer has no additional import route. □

Theorem 634.70 (QuotientSoundnessBoundary kernel dependency route)**.** *The downstream nonimport scope of* QuotientSoundnessBoundary$^{\uparrow}$ *is read first through chapter 629 for representative-refusal discipline and then through chapter 1157 for host-primitive leakage refusal. Only after those sibling boundaries are visible does the local quotient-soundness packet replace quotient elimination with displayed* psame *transport through E, T, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundaryKernelDependencyRoute`

Proof. Start with the downstream nonimport totality theorem above. It keeps downstream consumers inside the refused representative surface E, A, V and the admitted psame transport surface E, T, H, C, P, N.

Read the negative-name sibling first. The refusal discipline of chapter 629 classifies representative recovery as a boundary request rather than as supplied quotient content.

Read the host-primitive sibling next. The leakage refusal of chapter 1157 prevents the same request from being reintroduced as a host primitive, host quotient recursion, or Quot.sound witness.

Finally apply public bridge non-escape and psame consumer factorization. The surviving route is displayed psame transport inside the local rows, so the dependency route has no escape into quotient elimination or host-side primitive supply. □

Theorem 634.71 (QuotientSoundnessBoundary root psame route totality)**.** *For every accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet, every surviving root operation after representative refusal factors through the displayed* psame *route E, T, H, C, P, N. The packet exports no root operation through a representative selector, quotient equality eliminator, erased derivation recovery map, host quotient recursion, or* Quot.sound *witness.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root psame route totality`

Proof. Begin with the NameCert obligations and fixed classifier, Definition 634.2 and Definition 634.12. They identify the relation source and classifier rows read by any accepted packet.

Next use kernel non-escape and psame transport exhaustion, Definition 634.13 and Definition 634.8. These results leave only the classifier-respecting psame transport row and the structural rows H, C, P, N for a surviving root operation.

Finally apply Definition 634.58. Once Quot.sound is refused at the visible boundary, the positive consumer route is exactly E, T, H, C, P, N. □

Theorem 634.72 (QuotientSoundnessBoundary root psame source admission)**.** *Every root quotient-style operation accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet exposes the source relation rows E before the* psame *transport row T is readable. Thus the positive route E, T, H, C, P, N is admitted only after the relation source has been displayed, and no representative, erased derivation, host quotient witness, or* Quot.sound *token can stand in for that source admission.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root psame source admission`

Proof. Begin with the root request triad and root relation row exposure, Definition 634.14 and Definition 634.15. They make the source relation rows E visible at the root boundary before any consumer reads a surviving transport.

Next apply root transport verdict order, Definition 634.23. The refusal verdict is fixed before the positive route is consumed, so the later psame transport cannot create its own source relation.

Finally use psame transport exhaustion and non-escape, Definition 634.8 and Definition 634.13. The only surviving transport route is the displayed E, T, H, C, P, N route, and the excluded quotient primitives have no source-admission coordinate. □

Theorem 634.73 (QuotientSoundnessBoundary root representative refusal totality)**.** *Every root request for a representative, section, equality token, erased derivation, host quotient witness, or* Quot.sound *dependency is exhausted by the representative-request row A and refusal-verdict row V before any positive transport route is consumed. After A, V are read, the only admitted consumer route is the classifier-respecting* psame *route through E, T, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root representative refusal totality`

Proof. Use the root request triad, Definition 634.14. It sends representative-style demands to the visible request and refusal rows A, V while keeping the relation source row E explicit.

Apply root transport verdict order and the representative request frontier, Definition 634.23 and Definition 634.56. Together they place the refusal verdict before the transport row is available to a downstream consumer.

Then use psame transport exhaustion and non-escape, Definition 634.8 and Definition 634.13. They leave only the classifier-respecting route E, T, H, C, P, N after the refused request has been exhausted by A, V. □

Theorem 634.74 (QuotientSoundnessBoundary root Quot.sound exclusion)**.** *The combined root packet*

$$E, A, T, V, H, C, P, N$$

has no coordinate for Quot.sound*, quotient eliminators, host class selectors, representative recovery, erased derivation recovery, or hidden section choices. Relation exposure is recorded by E, representative-style refusal is recorded by A, V, and the only positive transport route is the displayed* psame *route E, T, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root quotsound nonimport certificate`

Proof. First combine the root source and refusal boundaries. By Definition 634.72 and Definition 634.73, the packet exposes E before transport and exhausts representative-style requests by A, V.

Next apply the root psame route totality, Definition 634.71. Every surviving root operation after refusal factors through the displayed E, T, H, C, P, N route.

Finally use psame transport exhaustion and kernel non-escape, Definition 634.8 and Definition 634.13. The packet contains only the displayed source, refusal, transport, stability, continuation, provenance, and naming rows, so Quot.sound and the other quotient primitives are excluded from the public surface. □

Theorem 634.75 (QuotientSoundnessBoundary refusal before transport)**.** *In every accepted* QuotientSoundnessBo *packet, the representative-request row A and refusal-verdict row V are read before the positive* psame *transport row T. Thus no surviving use of T can retroactively supply a representative selector, erased derivation, quotient equality eliminator, host quotient recursion, or* Quot.sound *witness.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary refusal before transport`

Proof. Apply the root transport verdict order and representative request frontier, Definition 634.23 and Definition 634.56. Together they place A, V at the visible refusal boundary before T is available to a downstream consumer.

Use verdict-transport separation, Definition 634.51. The positive transport row cannot reread the refusal verdict as a quotient eliminator or as evidence for a representative witness.

Finally invoke Definition 634.58. The exact complement between refusal and the surviving psame route gives the required order: refusal first, transport only after that refusal has been displayed. □

Theorem 634.76 (QuotientSoundnessBoundary kernel consumer audit triad)**.** *The kernel-facing audit surface of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet consists exactly of the relation exposure row, the refusal verdict for representative-side quotient primitives, and the surviving* psame *replacement route. Equivalently, every kernel consumer reads the triad*

$$E, A, V \quad \textit{and} \quad E, T, H, C, P, N$$

with the refusal side preceding transport, and no further quotient-soundness coordinate is exported.

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary kernel consumer audit triad`

Proof. First apply relation row exposure and the fixed classifier, Definition 634.15 and Definition 634.12. They expose the source relation and classifier rows without admitting a host quotient witness.

Next use refusal before transport above. The representative-request row and refusal-verdict row are read before the positive transport row, so the audit records the negative verdict before any surviving consumer can proceed.

Finally combine root psame route totality with quotient-sound refusal exactness. The surviving side is exactly the psame replacement route through E, T, H, C, P, N, while the refused side stays at E, A, V; kernel non-escape then rules out any additional quotient-soundness coordinate. □

Theorem 634.77 (QuotientSoundnessBoundary root representative refusal)**.** *Every root representative-side quotient request accepted by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet is read through the relation row E, the requested-representative row A, and the refusal-verdict row V before the transport row T becomes available. Thus the root request reaches the visible representative refusal frontier before any surviving* psame *replacement route can be consumed.*

Proof. Start with the representative request frontier, Definition 634.56. It exhausts representative, representative-selector, erased-derivation, and host quotient requests by E, A, V before the transport row T is readable.

Apply refusal before transport, Definition 634.75. The refusal verdict is consumed before any positive transport use, so a downstream root consumer cannot reverse the order and treat T as representative evidence.

Finally use the kernel consumer audit triad, Definition 634.76. The kernel-facing surface keeps the refusal side E, A, V distinct from the surviving transport side E, T, H, C, P, N, giving the stated root order. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root representative refusal certificate`

Theorem 634.78 (QuotientSoundnessBoundary root psame transport admission)**.** *Every surviving root operation of an accepted* QuotientSoundnessBoundary$^{\uparrow}$ *packet is admitted only through the classifier-respecting* psame *transport route E, T, H, C, P, N. The operation may read the refused representative-side rows E, A, V as its accountability surface, but its positive movement is exactly the displayed transport, stability, locality, provenance, and naming route.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root psame transport admission`

Proof. Begin after the refusal surface has been read. By Definition 634.77, every root representative-side request has already passed through E, A, V before the positive route is available.

Use psame-only descent, Definition 634.54. It states that accepted descent after refusal factors only through E, T, H, C, P, N by classifier-respecting psame transport.

Now apply root psame route totality, Definition 634.71. This makes the same E, T, H, C, P, N route total for surviving root operations and excludes representative selection, quotient equality elimination, erased derivation recovery, host quotient recursion, and Quot.sound. □

Theorem 634.79 (QuotientSoundnessBoundary root nonimport ledger)**.** *The root kernel ledger exported by a* QuotientSoundnessBoundary$^{\uparrow}$ *packet is exactly the refused representative surface E, A, V together with the admitted* psame *transport surface E, T, H, C, P, N. It imports no quotient carrier, representative selector, quotient equality eliminator, erased derivation recovery map, host quotient recursion principle, or* Quot.sound *dependency.*

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary root nonimport ledger`

Proof. First combine root representative refusal with root psame transport admission, Definition 634.77 and Definition 634.78. These two rows identify the negative ledger side E, A, V and the surviving positive side E, T, H, C, P, N.

Apply the kernel consumer audit triad, Definition 634.76. It packages those two sides as the kernel-facing audit surface and records that no further quotient-soundness coordinate is exported.

Finally cite public bridge non-escape and psame-only descent, Definition 634.52 and Definition 634.54. The bridge and descent rows exclude quotient carriers, representative selectors, erased derivation recovery, host quotient recursion, and Quot.sound, so the ledger is a nonimport ledger rather than a quotient extension. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ packet is generated from relation rows, requested representative rows, classifier-respecting psame transport, refusal verdicts, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 634.1, Definition 634.2, Definition 634.3, Definition 634.4, Definition 634.12, Definition 634.13, Definition 634.14, Definition 634.15, Definition 634.23, Definition 634.24, Definition 634.25, Definition 634.44, Definition 634.45, Definition 634.47, Definition 634.48, Definition 634.49, Definition 634.50, Definition 634.51, Definition 634.52, Definition 634.53, Definition 634.54, and Definition 634.55 bind the relation rows, representative-refusal rows, refusal-to-transport determinacy, source-row consumer totality, verdict-transport separation, erased derivation refusal, psame-only descent, root relation exposure, verdict-before-transport order, psame transport row, locality rows, provenance, public bridge, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuotientSoundnessBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not construct quotient carriers, representative selectors, quotient equality eliminators, erased derivation recovery maps, or any dependency on Quot.sound.

Upgrade path: No paper theory-axis grade remains; Lean verification and bridge checking remain independent obligations for the same finite refusal-and-transport packet.

Theorem 634.80 (QuotientSoundnessBoundary negative-boundary route)**.** *Every quotient-style representative request exposed to an accepted* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet is first read as a negative-name boundary request through chapter 629. Only after that boundary has displayed the blocked request may the quotient-soundness packet replace the attempted representative read by the local* psame *transport row over its visible relation data.*

Proof. Read the requested representative rows and refusal verdict from the carrier in Definition 634.1. These rows place the demand for a representative, quotient equality token, section, erased derivation tree, or Quot.sound dependency on the public refusal surface before any positive transport row is consumed.

The negative-name boundary of chapter 629 supplies the sibling route for that refusal. It treats the representative request as a blocked boundary read rather than as generated content.

After this refusal route is visible, apply Definition 634.3. The surviving operation is the classifier-respecting psame route through the displayed relation rows, componentwise hsame transport, visible Cont replay, Pkg provenance, and local NameCert row. No quotient representative, quotient eliminator, erased derivation recovery map, or Quot.sound dependency is exported. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary negative boundary route`

Theorem 634.81 (QuotientSoundnessBoundary negative-route visibility)**.** *Any quotient-style representative request exposed to an accepted* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *packet must display the negative-name boundary row and its refusal verdict before the local* psame *transport replacement can be read. The request may not consume quotient representatives, quotient eliminators, erased derivation recovery, or* Quot.sound *as hidden data behind the transport row.*

Proof. Apply the negative-boundary route of Definition 634.80. It places every representative request on the visible refusal surface before the packet admits any replacement transport.

Then use the transport replacement theorem, Definition 634.3. The replacement is the local psame route over displayed relation data, componentwise hsame transport, continuation replay, package provenance, and local naming.

Since the refusal verdict precedes the transport read, no consumer can reinterpret the replacement as a quotient representative, quotient eliminator, erased derivation recovery map, or Quot.sound dependency. Those requests remain visible as refused boundary rows. □

Lean4 ✓ `BEDC.Derived.QuotientSoundnessBoundaryUp.QuotientSoundnessBoundary negative route visibility`

The quotient-soundness boundary also records a sibling refusal route through chapter 998. That sibling supplies the quotient-facing refusal discipline consumed by the local QuotientSoundnessBoundary$^{\uparrow}$ rows before classifier-respecting psame transport replaces any representative-selection request.

Chapter 635

A Concrete Naming Certificate for $\mathsf{RealInverse}^{\uparrow}$

The $\mathsf{RealInverse}^{\uparrow}$ interface names the finite reciprocal packet for a real source whose nonzero domain is witnessed by apartness. It follows the Bishop–Bridges discipline: the reciprocal route is available only after an $\mathsf{ApartnessReal}^{\uparrow}$ witness supplies the constructive guard, and the resulting packet hands a product-seal route to $\mathsf{Real}^{\uparrow}$ without importing trichotomy, host division, quotient real equality, a selected limit, or completeness.

The reciprocal packet records its sibling route explicitly. It consumes the nonzero-domain witness from chapter 304, the product verification route from chapter 464, the finite-window route from chapter 126, the regular rational readback from chapter 301, and the seal surface from chapter 38. This is the dependency route for the displayed rows $x, a, p, w, r, s, h, c, l, n$; it is not another theorem and not a host division principle.

Definition 635.1 (Real inverse carrier)**.** *A* $\mathsf{RealInverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (x, a, p, w, r, s, h, c, l, n)$$

with the following displayed coordinates:

1. *x is the source* $\mathsf{Real}^{\uparrow}$ *row whose reciprocal is being named;*
2. *a is an* $\mathsf{ApartnessReal}^{\uparrow}$ *nonzero-domain witness licensing the reciprocal route for x;*
3. *p is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product route that checks the candidate inverse against the source;*
4. *w is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the guarded source and candidate reciprocal;*
5. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational window row exposed through w;*
6. *s is the* $\mathsf{Real}^{\uparrow}$ *seal row receiving the guarded reciprocal packet;*
7. *h stores componentwise* hsame *transports for the source, apartness witness, product route, windows, regular rational row, seal, provenance, and local name;*
8. *c stores the displayed* Cont *routes from the apartness witness to the reciprocal window, from the window to the regular rational row, from the product route to the seal, and from the finite packet to the local name;*
9. *l records* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. n is the local $\mathsf{NameCert}_{\mathsf{RealInverse}^\uparrow}$ *row.*

The carrier has no coordinate for trichotomy, host division, quotient real equality, selected real values, infinite tails, or $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.RealInverseUp.RealInverseCarrier`

Theorem 635.2 (Real inverse NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealInverse}^\uparrow}$ *surface for Definition 635.1 consists exactly of source* $\mathsf{Real}^\uparrow$ *admission,* $\mathsf{ApartnessReal}^\uparrow$ *nonzero-domain admission,* $\mathsf{CauchyProduct}^\uparrow$ *product-route exposure,* $\mathsf{StreamName}^\uparrow$ *finite-window readback,* $\mathsf{RegSeqRat}^\uparrow$ *regular rational exposure,* $\mathsf{Real}^\uparrow$ *seal handoff, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseNameCertObligations`

Proof. Carrier admission is projection from the displayed packet I. Its rows list the real source, apartness guard, product route, finite window, regular rational readback, seal, transport, continuation, provenance, and local certificate.

The reciprocal obligation is guarded by the apartness row. The certificate first reads a, then uses the continuation rows to expose the finite window w and the regular rational row r that describe the candidate reciprocal.

The product route p supplies the only multiplication-facing check before the seal s is exposed. The remaining rows h, c, l, n transport and name this finite route, so the certificate contains no trichotomy, host division, quotient real equality, selected limit, infinite tail, or completeness coordinate. □

Theorem 635.3 (Real inverse apartness domain)**.** *Every accepted* $\mathsf{RealInverse}^\uparrow$ *packet licenses reciprocal use only through its displayed* $\mathsf{ApartnessReal}^\uparrow$ *witness a. A consumer that cannot read a receives no reciprocal window, no regular rational candidate row, and no product-seal route.*

Proof. Project the carrier $I = (x, a, p, w, r, s, h, c, l, n)$. The only coordinate naming a nonzero domain is the apartness row a.

Read the continuation routes in c. They place the reciprocal window w and the regular rational row r downstream of the apartness witness, and the product route p is exposed only along the same guarded path.

Thus the reciprocal is not a total host operation on $\mathsf{Real}^\uparrow$. Without the constructive apartness witness, the packet has no route to a reciprocal candidate, product check, or seal handoff. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseApartnessDomain`

Theorem 635.4 (Real inverse apartness denominator handoff)**.** *For an accepted* $\mathsf{RealInverse}^\uparrow$ *carrier $I = (x, a, p, w, r, s, h, c, l, n)$, every denominator or reciprocal-candidate read must pass through the displayed* $\mathsf{ApartnessReal}^\uparrow$ *witness a, then the* $\mathsf{StreamName}^\uparrow$ *window w, then the* $\mathsf{RegSeqRat}^\uparrow$ *row r, before the* $\mathsf{CauchyProduct}^\uparrow$ *route p may certify the product seal. The handoff exposes no trichotomy, host division, quotient real equality, selected limit, infinite tail, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the accepted carrier $I = (x, a, p, w, r, s, h, c, l, n)$. The only row licensing a denominator read is the displayed apartness witness a.

Apply Definition 635.3. It forces the nonzero guard before any reciprocal window, regular rational candidate row, or product-seal route can be consumed.

The denominator read is then ordered through w and r. The $\mathsf{StreamName}^{\uparrow}$ window supplies the bounded observation of the source and candidate reciprocal, and the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback attached to that window.

Only after those rows are visible may the $\mathsf{CauchyProduct}^{\uparrow}$ route p certify the product seal. The structural rows h, c, l, n replay and name this same guarded path, so the packet refuses trichotomy, host division, quotient real equality, selected limits, infinite tails, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseApartnessDenominatorHandoff`

Theorem 635.5 (Real inverse product seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealInverse}^{\uparrow}$ *packet factors through the* $\mathsf{CauchyProduct}^{\uparrow}$ *product route* p*, the* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* r*, the displayed transport and continuation rows, the provenance ledger, and the seal row* s*. The handoff exposes reciprocal information only as the apartness-guarded product certificate named by the packet.*

Proof. Begin with the apartness-domain theorem. It places every reciprocal read downstream of the nonzero-domain witness a.

The finite observation then runs through w and r. The $\mathsf{StreamName}^{\uparrow}$ row supplies the bounded read, and the $\mathsf{RegSeqRat}^{\uparrow}$ row is the regular rational surface passed to the product check.

Finally the $\mathsf{CauchyProduct}^{\uparrow}$ route p supplies the product certificate consumed by the seal row s. The transport, continuation, provenance, and local naming rows record this route, so the real-facing handoff receives only the guarded finite packet. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseProductSealHandoff`

Theorem 635.6 (Real inverse guarded product uniqueness)**.** *Let* $I = (x, a, p, w, r, s, h, c, l, n)$ *and* $I' = (x, a', p', w', r', s, h', c', l', n')$ *be accepted* $\mathsf{RealInverse}^{\uparrow}$ *packets for the same* $\mathsf{Real}^{\uparrow}$ *source row* x*. If the apartness witnesses* a, a' *name the same* $\mathsf{ApartnessReal}^{\uparrow}$ *guard, the product routes* p, p' *expose* hsame*-compatible* $\mathsf{CauchyProduct}^{\uparrow}$ *certificates, and both packets hand off to the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row* s*, then* I *and* I' *are classifier-equal as* $\mathsf{RealInverse}^{\uparrow}$ *packets.*

Proof. Project the two carriers through Definition 635.1. The source row and seal row are shared by assumption, while the only reciprocal-domain coordinates are the apartness witnesses.

Apply the apartness-domain theorem, Definition 635.3. Since both witnesses name the same constructive guard, both reciprocal reads must pass through the same guarded route before any reciprocal window or product certificate is visible.

Next use the product-seal handoff of Definition 635.5. The hsame-compatible product routes align the $\mathsf{CauchyProduct}^{\uparrow}$ rows, the finite windows w, w', the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks r, r', and the common seal s along the displayed continuation rows.

It remains only to compare the bookkeeping rows. Componentwise transport in h, h' carries the aligned analytic rows, c, c' replay the same guarded route, l, l' record the same provenance surface, and n, n' name the local certificate. The non-escape theorem then rules out any hidden trichotomy, host division, quotient real equality, selected limit, infinite tail, or completeness coordinate, so classifier equality is componentwise. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseGuardedProductUniqueness`

Theorem 635.7 (Real inverse non-escape)**.** *The carrier, NameCert obligations, apartness domain, and product seal handoff of Definition 635.1, Definition 635.2, Definition 635.3, and Definition 635.5 assemble one constructive reciprocal certificate over* $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The certificate exports exactly the rows

$$x, a, p, w, r, s, h, c, l, n.$$

It exports no trichotomy, host division, quotient real equality, selected limit, infinite tail, or completeness principle.

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseNonEscape`

Proof. The carrier definition lists every row in the reciprocal packet. The NameCert obligations bind source admission, apartness admission, product-route exposure, finite-window readback, regular rational exposure, seal handoff, transport, continuation, provenance, and naming to those rows.

The apartness-domain theorem makes the reciprocal route partial in the constructive sense: it is available only after the nonzero-domain witness has been read. The product seal handoff then routes every real-facing consumer through the finite window, regular rational row, product certificate, and seal.

No other coordinate remains from which to read trichotomy, host division, quotient real equality, a selected limit, an infinite tail, or completeness. The displayed rows therefore exhaust the $\mathsf{RealInverse}^{\uparrow}$ observation surface. □

Theorem 635.8 (Real inverse obligation closure package)**.** *The carrier of Definition 635.1, the NameCert obligations of Definition 635.2, the apartness-domain theorem Definition 635.3, the product-seal handoff Definition 635.5, guarded product uniqueness Definition 635.6, and non-escape Definition 635.7 assemble the obligation-closure surface for* $\mathsf{RealInverse}^{\uparrow}$ *over* $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The package remains apartness-guarded and refuses trichotomy, total reciprocal on all real rows, host division, quotient equality, selected limits, infinite tails, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier and NameCert obligations. They display the source, apartness guard, product route, StreamName window, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming rows as the full reciprocal packet.

Apply the apartness-domain and product-seal handoff theorems. The reciprocal route is available only after the constructive nonzero witness is read, and the Real-facing seal is reached only through the finite product certificate, window, and regular rational rows.

Next use guarded product uniqueness to make the package stable under matching apartness witnesses, compatible product certificates, and the same terminal Real seal. The non-escape theorem then removes trichotomy, total reciprocal, host division, quotient equality, selected limits, infinite tails, and Real completeness from the available surface, so the named rows form the obligation-closure package. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseObligationClosurePackage`

Theorem 635.9 (Real inverse apartness-product scoped factorization)**.** *Every* $\mathsf{RealInverse}^{\uparrow}$ *consumer factors through the displayed* $\mathsf{ApartnessReal}^{\uparrow}$ *nonzero guard,* $\mathsf{CauchyProduct}^{\uparrow}$ *product route,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame

transport, displayed Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealInverse}^\uparrow}$ *row before any reciprocal read is public.*

The factorization is guarded by the apartness row and product route. It does not produce a total reciprocal on all real rows, trichotomy, host division, quotient real equality, selected limits, infinite tails, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 635.1 and the NameCert obligations of Definition 635.2. They make the rows $x, a, p, w, r, s, h, c, l, n$ exhaustive for an accepted reciprocal packet.

Apply the apartness-domain theorem Definition 635.3. It places every reciprocal read downstream of the displayed $\mathsf{ApartnessReal}^\uparrow$ nonzero guard, so no consumer can reach a reciprocal window without first reading a.

Next apply the product-seal handoff of Definition 635.5. The handoff orders the consumer through the $\mathsf{CauchyProduct}^\uparrow$ route, finite $\mathsf{StreamName}^\uparrow$ windows, $\mathsf{RegSeqRat}^\uparrow$ readback, and the terminal $\mathsf{Real}^\uparrow$ seal, with transport and continuation preserving the same route.

Guarded product uniqueness in Definition 635.6 identifies compatible guarded product routes, and non-escape in Definition 635.7 removes trichotomy, total reciprocal, host division, quotient equality, selected limits, infinite tails, and completeness from the available surface. The obligation-closure package of Definition 635.8 then repacks exactly this route with Pkg provenance and the local NameCert row. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInverseApartnessProductScopedFactorization`

Theorem 635.10 (Real inverse public inverse domain)**.** *The public* $\mathsf{RealInverse}^\uparrow$ *interface is a partial reciprocal domain. A consumer may read a reciprocal only after the displayed* $\mathsf{ApartnessReal}^\uparrow$ *guard,* $\mathsf{CauchyProduct}^\uparrow$ *product route,* $\mathsf{StreamName}^\uparrow$ *finite window,* $\mathsf{RegSeqRat}^\uparrow$ *row,* $\mathsf{Real}^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealInverse}^\uparrow}$ *rows are visible.*

The public route is therefore

$$\mathsf{ApartnessReal}^\uparrow \longmapsto \mathsf{CauchyProduct}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow \longmapsto \mathsf{NameCert}_{\mathsf{RealInverse}^\uparrow},$$

with the structural rows carried by h, c, l, n*. It does not claim trichotomy, total inverse on all real rows, host division, quotient real equality, selected limits, infinite tails, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Start with the apartness-domain theorem, Definition 635.3. It places every reciprocal read downstream of the constructive nonzero guard, so the public interface cannot be total on all $\mathsf{Real}^\uparrow$ rows.

Apply the denominator handoff and product-seal handoff, Definition 635.4 and Definition 635.5. Together they order the read through the $\mathsf{StreamName}^\uparrow$ window, $\mathsf{RegSeqRat}^\uparrow$ rational row, $\mathsf{CauchyProduct}^\uparrow$ certificate, and terminal $\mathsf{Real}^\uparrow$ seal.

Guarded product uniqueness, Definition 635.6, identifies compatible guarded product routes only after the same apartness guard, product certificate, and seal have been displayed. Non-escape, Definition 635.7, removes every coordinate for trichotomy, host division, quotient equality, selected limits, infinite tails, and completeness.

Finally use the scoped factorization theorem Definition 635.9. It repacks the public route with componentwise transport, continuation replay, provenance, and local naming, so the exported interface is exactly the apartness-guarded partial reciprocal domain. □

Theorem 635.11 (Real inverse public transport consumer factorization)**.** *Every public reciprocal consumer for an accepted* $\mathsf{RealInverse}^{\uparrow}$ *packet transports compatible apartness guards,* $\mathsf{CauchyProduct}^{\uparrow}$ *routes,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, and* $\mathsf{Real}^{\uparrow}$ *seals through componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealInverse}^{\uparrow}}$ *row before reading the reciprocal.*

The factorization is the public route

$$\mathsf{ApartnessReal}^{\uparrow} \longmapsto \mathsf{CauchyProduct}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{RealInverse}^{\uparrow}},$$

with the structural rows h, c, l, n *preserving the same guarded reciprocal packet. It does not export a total reciprocal, trichotomy, host division, quotient real equality, selected limits, infinite tails, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the public inverse-domain theorem, Definition 635.10. It makes the reciprocal read public only after the apartness guard, product route, finite window, regular rational row, Real seal, transport, replay, provenance, and local name are all visible.

Apply apartness-domain and denominator handoff, Definition 635.3 and Definition 635.4. They force every denominator and reciprocal-candidate read through the displayed $\mathsf{ApartnessReal}^{\uparrow}$ witness before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be inspected.

Next use product-seal handoff and guarded product uniqueness, Definition 635.5 and Definition 635.6. The $\mathsf{CauchyProduct}^{\uparrow}$ route supplies the product certificate consumed by the $\mathsf{Real}^{\uparrow}$ seal, and compatible guarded product routes identify the same public reciprocal surface.

Finally apply non-escape and scoped factorization, Definition 635.7 and Definition 635.9. These results repack the route through hsame, Cont, Pkg, and NameCert while excluding every coordinate for total reciprocals, trichotomy, host division, quotient equality, selected limits, infinite tails, or Real completeness. □

Lean4 ✓ `BEDC.Derived.RealInverseUp.RealInversePublicTransportConsumerFactorization`

Closure status for $\mathsf{NameCert}_{\mathsf{RealInverse}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealInverse}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ source, an $\mathsf{ApartnessReal}^{\uparrow}$ nonzero-domain witness, a $\mathsf{CauchyProduct}^{\uparrow}$ product route, $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ finite readback rows, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealInverse}^{\uparrow}}$, publicClosed)

Scope closed: Definition 635.1, Definition 635.2, Definition 635.3, Definition 635.5, Definition 635.6, Definition 635.7, Definition 635.8, the scoped factorization witness Definition 635.9, the public inverse-domain theorem Definition 635.10, and the public transport consumer factorization Definition 635.11 bind the apartness-guarded reciprocal carrier, product route, finite windows, regular rational row, seal handoff, uniqueness, refusal ledger, obligation package, and public consumer route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealInverse}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close trichotomy, total reciprocal on all real rows, host division, quotient real equality, selected limits, infinite tails, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridged closure requires Lean verification and a standard bridge for the public apartness guard, product route, finite windows, regular rational row, Real seal, transport, replay, provenance, and local NameCert primitives.

Chapter 636

A Concrete Naming Certificate for CauchyModulusComparison$^{\uparrow}$

The CauchyModulusComparison$^{\uparrow}$ packet names the finite comparison surface between two Cauchy-modulus schedules before a real-facing consumer reads a shared tail. It transports the stationary diagonal observation pattern into a concrete NameCert row: two CauchyModulus$^{\uparrow}$ schedules are compared through one RegularCauchyTailMeet$^{\uparrow}$ threshold, finite StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ tolerance rows, RegSeqRat$^{\uparrow}$ readbacks, and a Real$^{\uparrow}$ seal handoff. The packet exports no quotient stream, no host real equality, no selected limiting representative, and no ambient completeness theorem.

The comparison packet consumes the shared-threshold source of chapter 560 before the StreamName$^{\uparrow}$ and Real$^{\uparrow}$ seal rows are read.

Definition 636.1 (Cauchy modulus comparison carrier)**.** *A* CauchyModulusComparison$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M_0, M_1, T, W, Q, E, H, C, P, N)$$

with the following displayed coordinates:

1. M_0 *is the first accepted* CauchyModulus$^{\uparrow}$ *schedule row;*
2. M_1 *is the second accepted* CauchyModulus$^{\uparrow}$ *schedule row;*
3. T *is the shared* RegularCauchyTailMeet$^{\uparrow}$ *threshold packet reading both schedules through one tail-meet demand;*
4. W *is the finite* StreamName$^{\uparrow}$ *window ledger selected by the shared threshold for the two schedule presentations;*
5. Q *is the combined* DyadicRatCore$^{\uparrow}$ *tolerance ledger and* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows;*
6. E *is the* Real$^{\uparrow}$ *seal row reached only after the shared tail threshold, windows, tolerance ledger, and regular readback have been displayed;*
7. H *stores componentwise* hsame *transports for* $M_0, M_1, T, W, Q, E, P, N$ *and for the local packet shape;*
8. C *stores displayed* Cont *routes from each schedule to* T*, from* T *to* W*, from* W *to* Q*, from* Q *to* E*, and from the packet rows to the local naming row;*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}$ *row.*

The carrier contains no external modulus ordering, no quotient equality of streams, no host real equality, no hidden tail oracle, no selected real value, and no ambient completeness coordinate.

Lean4 (definition) `BEDC.Derived.CauchyModulusComparisonUp.CauchyModulusComparisonUp`
Lean4 ✓ `BEDC.Derived.CauchyModulusComparisonUp.CauchyModulusComparisonTasteGate single carrier alignment`

Theorem 636.2 (Cauchy modulus comparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}$ *surface for Definition 636.1 consists of six finite obligation rows:*

1. *carrier admission for $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$;*
2. *paired schedule admission exposing exactly the two* $\mathsf{CauchyModulus}^{\uparrow}$ *rows M_0 and M_1;*
3. *shared-threshold admission exposing the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row T as the only tail comparison datum;*
4. *finite-window and tolerance exactness forcing every compared read through W and Q;*
5. *componentwise stability under the displayed* hsame *rows and replay through the displayed* Cont *routes;*
6. *ledger exactness stating that every real-facing read factors through $M_0, M_1, T, W, Q, E, H, C, P, N$ and no host comparison or quotient stream is exported.*

Proof. Carrier admission is projection from the displayed packet. The tuple lists the two schedules, the shared tail-meet threshold, selected windows, tolerance and regular-readback ledger, Real seal, transports, continuations, provenance, and local name.

The paired schedule clause reads M_0 and M_1 before any comparison result is available. The packet therefore compares displayed schedule rows rather than an external ordering on functions.

The threshold clause uses the single row T. Since T is the only tail-meet coordinate, every downstream comparison has to pass through the same shared threshold before it can inspect W or Q.

Finally H, C, P, N repack the route as a local NameCert surface. With no carrier coordinate for quotient streams, host real equality, selected limits, hidden tail oracles, or ambient completeness, the six rows exhaust the public comparison packet. □

Theorem 636.3 (Cauchy modulus comparison paired schedule admission)**.** *Let $K = (M_0, M_1, T, W, Q, E, H, C, P,$ be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. Before any comparison route is read, the packet exposes exactly the two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule rows M_0 and M_1. No third schedule row, ambient order on modulus functions, quotient stream representative, or hidden tail oracle is available to the comparison consumer.*

Proof. Project the carrier tuple of Definition 636.1. The first two coordinates are precisely the displayed $\mathsf{CauchyModulus}^{\uparrow}$ rows M_0 and M_1.

Use the paired schedule row of Definition 636.2. It names those two rows before the threshold, window, tolerance, seal, transport, continuation, package, and local certificate coordinates are consumed.

The later directional route in Definition 636.6 starts from the same two rows and then passes through T, W, Q, H, C, P, N. Since the carrier contains no additional schedule coordinate or ambient ordering field, admission is exactly the displayed pair M_0, M_1. □

Theorem 636.4 (Cauchy modulus comparison finite-window tolerance exactness)**.** *Let $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. Every comparison read passes through the shared threshold T, the finite window ledger W, and the tolerance-readback ledger Q. Hence the read is mediated by* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks before any* $\mathsf{Real}^{\uparrow}$ *seal row can be consumed.*

Proof. Start with the carrier inventory and the finite-window row of Definition 636.2. The only comparison path after the two schedules proceeds through the threshold coordinate T and then the selected window coordinate W.

Apply Definition 636.6 for comparison-facing reads. Its route includes T, W, Q before any public comparison conclusion is available.

Use Definition 636.7 to identify T as the single shared tail threshold and W as the downstream finite window ledger. Then use Definition 636.8: the $\mathsf{Real}^{\uparrow}$ seal E is reached only after Q has supplied the dyadic tolerance and regular rational readback rows. This proves exactness of the displayed finite-window tolerance route. □

Theorem 636.5 (Cauchy modulus comparison transport route stability)**.** *For an accepted* CauchyModulusComparis *carrier $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$, componentwise* hsame *rows in H and displayed* Cont *routes in C preserve the schedule, threshold, window, tolerance, seal, package, and local* NameCert *coordinates. A transported read therefore remains a read of the same displayed packet rows $M_0, M_1, T, W, Q, E, P, N$.*

Proof. Project the transport and continuation coordinates of the carrier. The row H is indexed componentwise over the schedule rows, threshold, windows, tolerance-readback ledger, seal row, package provenance, local name, and packet shape; C displays the replay routes between those coordinates.

Use the stability row of Definition 636.2. It requires transported comparison reads to move along these displayed hsame and Cont rows, so transport cannot replace any schedule, threshold, window, tolerance, seal, package, or local naming coordinate by an unlisted one.

The directional, shared-tail, and seal-facing conclusions of Definition 636.6, Definition 636.7, and Definition 636.8 all use the same route inventory. Thus stability preserves the displayed packet coordinates through every public comparison read. □

Theorem 636.6 (Cauchy modulus comparison directional monotonicity)**.** *Let $K = (M_0, M_1, T, W, Q, E, H, C, P, N$ be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. Any consumer read asserting that the first schedule is no harder than the second schedule factors through the displayed route*

$$M_0, M_1, T, W, Q, H, C, P, N.$$

The directional read is therefore a finite shared-tail NameCert comparison over the two schedule rows, not an ambient order on external modulus functions.

Proof. Project the accepted carrier and read the two schedule rows M_0 and M_1. These are the only coordinates that name Cauchy-modulus schedules.

Next read the shared threshold T and its selected windows W. The threshold packet supplies the common tail at which the two schedules are compared, and the window ledger records only the finite observations selected by that threshold.

The tolerance and readback ledger Q supplies the dyadic comparison rows and the regular rational readback rows attached to the same selected windows. Transport through H and replay through C preserve this route componentwise.

Thus a directional comparison can only be the displayed shared-tail comparison. The provenance P and local name N close the certificate without adding an external modulus order, quotient stream, host real equality, or hidden tail oracle. □

Theorem 636.7 (Cauchy modulus comparison shared-tail exactness)**.** *Every finite tail observation exported by an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier uses the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold row T for both schedules. The observation cannot replace one side by an unlisted threshold, an unbounded tail family, a quotient stream representative, or an external tail oracle.*

Proof. Begin with the threshold coordinate T in the carrier. It is the only row whose role is to meet the two Cauchy-modulus schedules at a shared tail.

The selected windows W are downstream of T, and the tolerance-readback ledger Q is downstream of those windows. Hence every tail observation has already passed through the same threshold row before either side of the comparison can be read.

Componentwise hsame transport in H and displayed continuation replay in C preserve the identity of the threshold row through consumer reads. The package and local NameCert rows record this same dependency, so no unlisted threshold or hidden tail family remains available. □

Theorem 636.8 (Cauchy modulus comparison Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier receives the seal row E only after reading the two modulus schedules, the shared tail threshold, the selected stream windows, the dyadic tolerance and regular-readback ledger, the componentwise transports, the continuation routes, the provenance row, and the local NameCert row.*

Proof. Start with the seal coordinate E. It is listed after the schedule rows, shared threshold, selected windows, and tolerance-readback ledger, and no other $\mathsf{Real}^{\uparrow}$ seal coordinate appears in the carrier.

Use the directional monotonicity theorem for comparison-facing consumers. It confines the comparison route to $M_0, M_1, T, W, Q, H, C, P, N$ before any seal-facing conclusion is available.

Use the shared-tail exactness theorem for tail-facing consumers. It ensures that both sides of the seal route use the same threshold T and the same selected finite window ledger.

Consequently the Real seal handoff is the finite route through $M_0, M_1, T, W, Q, E, H, C, P, N$. The packet has no coordinate for quotient equality, host real equality, selected limits, hidden representatives, external modulus functions, or ambient completeness. □

Theorem 636.9 (Cauchy modulus comparison non-escape boundary)**.** *An accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *packet exposes only the rows*

$$M_0, M_1, T, W, Q, E, H, C, P, N.$$

It excludes quotient streams, selected limits, host real equality, ambient completeness, hidden modulus schedules, external tail oracles, and any comparison read that bypasses the shared threshold row T.

Proof. Inspect the carrier of Definition 636.1. Its coordinates are exactly the two schedule rows, one shared threshold, finite windows, dyadic tolerance and regular-readback ledger, Real seal, transports, routes, provenance, and local name.

The positive comparison route has already been constrained by Definition 636.6, Definition 636.7, and Definition 636.8. Each route passes through T, W, Q before any seal-facing read.

Since no coordinate stores a quotient stream, selected limiting representative, host real equality, ambient completeness theorem, hidden schedule, or external oracle, no accepted packet can export such data. □

Theorem 636.10 (Cauchy modulus comparison shared-threshold package)**.** *For every accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier* $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$*, the obligation package consists of the two modulus schedules* M_0, M_1*, the shared tail threshold* T*, the dyadic request and finite stream-window data stored in* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data stored in* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the componentwise* hsame *rows* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

Proof. Project the tuple K. The first two rows are the only displayed Cauchy-modulus schedules, and T is the only shared threshold that can compare them.

Read W and Q as the finite observation package. The dyadic request and selected windows live in W, while regular rational readback and dyadic tolerance evidence live in Q.

The seal-facing and certification rows are E, H, C, P, N. They hand the package to $\mathsf{Real}^{\uparrow}$, preserve componentwise agreement, replay the route, record provenance, and name the packet without adding any further comparison datum. □

Theorem 636.11 (Cauchy modulus comparison threshold-window coherence)**.** *Let* $K = (M_0, M_1, T, W, Q, E, H, C$ *be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. Every accepted packet routes both schedules through the same threshold row* T *before any finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger* Q *is read. Thus the window and tolerance reads are downstream of one displayed shared-threshold route, not independent side reads of the two schedules.*

Proof. Project the carrier of Definition 636.1. Its third coordinate is the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row T, and the finite window and tolerance-readback coordinates W and Q are listed only after that threshold row.

Apply the directional routing of Definition 636.6. A comparison-facing consumer must read $M_0, M_1, T, W, Q, H, C, P, N$, so the schedules meet at T before any window or tolerance ledger is inspected.

Use Definition 636.7 and Definition 636.10. Both name the same threshold coordinate as the sole shared tail row and place W and Q downstream of it. Hence threshold-window coherence is exactly the displayed carrier route. □

Theorem 636.12 (Cauchy modulus comparison window-ledger source exhaustion)**.** *Let* $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. Every Real-facing comparison read whose seal row is* E *is sourced exactly by the two schedule rows* M_0, M_1*, the single shared threshold row* T*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *ledger* Q*. The route cannot substitute an independent window, an unlisted tolerance row, a quotient stream, host real equality, or ambient completeness for those displayed sources.*

Proof. Project the carrier of Definition 636.1. The rows available to a seal-facing comparison read are the two schedules M_0, M_1, the shared threshold T, the window ledger W, the tolerance and regular-readback ledger Q, the seal E, the transport row H, the route row C, provenance P, and the local naming row N.

Apply Definition 636.11. It forces the route through T before W and Q are read, so the selected window and tolerance ledgers are downstream of one displayed shared threshold rather than independent side data.

Apply Definition 636.13. The only schedule sources are M_0 and M_1, and every comparison read starts from that pair before moving through the displayed threshold, window, ledger, transport, and route rows.

Finally use Definition 636.14. The seal row E is a downstream Real-facing row of this finite route, and the carrier inventory contains no coordinate for an independent window, unlisted tolerance row, quotient stream, host real equality, or ambient completeness. Hence the listed rows exhaust the source surface for the seal-facing comparison read. □

Theorem 636.13 (Cauchy modulus comparison schedule-pair source lock)**.** *Let $K = (M_0, M_1, T, W, Q, E, H, C, P,$ be an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier. The pair (M_0, M_1) is the complete source surface for comparison reads. It cannot be replaced by a hidden third modulus schedule, an external ordering of host functions, or a comparison source not transported through the displayed* hsame *and* Cont *rows.*

Proof. Read the first two coordinates of the carrier in Definition 636.1. They are the only $\mathsf{CauchyModulus}^{\uparrow}$ schedule rows in the packet.

The paired schedule admission theorem and Definition 636.6 start every comparison route from those same two rows. The route then proceeds through T, W, Q, H, C, P, N rather than through a host-level order or an unlisted schedule.

Finally apply Definition 636.9 and Definition 636.10. The non-escape boundary excludes hidden schedules and external orders, while the package theorem lists no source coordinate beyond M_0 and M_1. □

Theorem 636.14 (Cauchy modulus comparison Real seal comparison non-escape)**.** *For an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *carrier $K = (M_0, M_1, T, W, Q, E, H, C, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal E is only a downstream seal of the displayed finite comparison route*

$$M_0, M_1, T, W, Q, E, H, C, P, N.$$

It does not assert $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient stream equality, or host real equality.*

Proof. Start from the seal coordinate in Definition 636.1. The carrier places E after the two schedules, the shared threshold, the finite window ledger, and the dyadic and regular rational readback ledger.

Apply Definition 636.8. It states that every Real-facing consumer receives E only after the displayed comparison route has been read.

Use Definition 636.9 for the excluded conclusions. Since the carrier has no coordinate for completeness, selected limiting representatives, quotient stream equality, or host real equality, the seal remains a downstream row of the finite comparison packet. The shared threshold package records the same finite route and supplies no additional Real-side claim. □

Theorem 636.15 (Cauchy modulus comparison obligation triad)**.** *Every accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *packet supplies exactly three consumer-facing obligation groups: paired schedule admission for M_0 and M_1; shared threshold, window, and tolerance routing through T, W, Q; and* $\mathsf{Real}^{\uparrow}$ *seal handoff through E with non-escape from quotient streams and host real equality.*

Proof. The first group is paired schedule admission. Apply Definition 636.3: the packet exposes exactly the two $\mathsf{CauchyModulus}^{\uparrow}$ rows M_0 and M_1 before any comparison route is read.

The second group is the shared threshold, window, and tolerance route. Apply Definition 636.11 to force both schedules through T before W and Q are inspected. Then apply Definition 636.12 to exhaust the selected window ledger, dyadic tolerance data, and regular rational readback rows.

The third group is the Real seal handoff. Apply Definition 636.14: the seal row E is downstream of the displayed finite route and does not assert quotient stream equality, host real equality, selected limits, or ambient completeness.

These three groups cover the carrier rows used by comparison consumers. The transport, continuation, provenance, and local naming rows preserve and expose the same route; they do not add a fourth consumer-facing obligation group. □

Theorem 636.16 (Cauchy modulus comparison obligation closure package)**.** *The finite obligation-closure surface for* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *is assembled by the carrier of Definition 636.1, the NameCert obligations of Definition 636.2, the directional monotonicity route of Definition 636.6, the shared-tail exactness of Definition 636.7, the Real seal handoff of Definition 636.8, and the non-escape boundary of Definition 636.9.*

Proof. The carrier supplies the rows of the package, and the NameCert theorem turns those rows into the local obligation surface.

Directional monotonicity constrains comparison-facing reads. Shared-tail exactness forces both schedules through the same threshold before window and tolerance data are inspected.

The Real seal handoff constrains seal-facing consumers, and the non-escape boundary records the excluded quotient, limit, host equality, completeness, hidden schedule, and oracle coordinates. Together these clauses give the finite obligation package without changing the chapter's closure grade. □

Theorem 636.17 (Cauchy modulus comparison scope package)**.** *Every obligation row of an accepted* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *packet*

$$K = (M_0, M_1, T, W, Q, E, H, C, P, N)$$

is scoped by the displayed $\mathsf{CauchyModulus}^{\uparrow}$ *schedule rows* M_0, M_1*, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold* T*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks in* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* hsame *transports* H*, the* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}$ *row* N*.*

Proof. Carrier scoping is projection from Definition 636.1. The tuple lists exactly the two schedule rows, shared threshold, windows, tolerance-readback ledger, seal, transport rows, route rows, provenance, and local name.

Paired-schedule admission and threshold-window coherence bind the source and tail rows. Apply Definition 636.3 for M_0, M_1, then apply Definition 636.11 for the route through T before W and Q are inspected.

Window-ledger source exhaustion accounts for the finite observation rows. Apply Definition 636.12: every selected window, dyadic tolerance row, and regular rational readback is already a component of W and Q.

The Real seal comparison non-escape theorem closes the consumer boundary. Apply Definition 636.14; it keeps E downstream of the displayed comparison route and excludes quotient streams, selected limits, host real equality, hidden schedules, and ambient completeness. Thus every obligation row is scoped by the listed packet coordinates. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusComparison}^{\uparrow}$ packet is generated from two $\mathsf{CauchyModulus}^{\uparrow}$ schedule rows, one $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared threshold, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 636.1, Definition 636.2, Definition 636.6, Definition 636.7, Definition 636.8, and Definition 636.17 bind the finite comparison route from two Cauchy-modulus schedules through a shared tail threshold to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusComparison}^{\uparrow}}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.CauchyModulusComparisonUp.taste gate`

Lean target: `BEDC.Derived.CauchyModulusComparisonUp.CauchyModulusComparisonTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a public order on modulus functions, RealUp completeness, quotient stream equality, host real equality, selected limits, or bridge checking.

Upgrade path: The paper theory axis has no further closure-grade upgrade pending; bridge checking remains a separate verification-axis matter for the same displayed schedule, threshold, window, tolerance, readback, and seal rows.

Chapter 637

A Concrete Naming Certificate for $\mathsf{RealApproximationEnvelope}^{\uparrow}$

The $\mathsf{RealApproximationEnvelope}^{\uparrow}$ interface names the finite observation packet used by Real and Completion consumers before any completed real value is available. It packages a tolerance request, a finite stream window, a dyadic observation, a regular-sequence classifier row, and a $\mathsf{Real}^{\uparrow}$ seal handoff into one local $\mathsf{NameCert}$ surface. The packet is designed for diagonal representatives, stationary rational finite windows, Cauchy-modulus extraction, and RealUp classifier handoffs without importing host real equality, quotient stream equality, selected limits, or ambient completeness.

The envelope consumes the finite-window route of chapter 126, the dyadic observation ledger of chapter 291, the regular-rational classifier of chapter 301, and the seal surface of chapter 38. These sibling surfaces are the rows W, D, R, S of the carrier below, and the edge is a finite observation-envelope route rather than an independent completed real.

Definition 637.1 (Real approximation-envelope carrier)**.** *A* $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *ε is the tolerance request row naming the precision scale to be answered by the envelope;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected for that tolerance request;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row read through W, including the displayed endpoint, radius, and finite arithmetic ledger;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row that reads the same window as a regular rational observation;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row receiving the observation packet for downstream real-name consumers;*
6. *H records componentwise* hsame *transports for the tolerance request, window, dyadic observation, classifier row, seal row, provenance, and local name;*
7. *C records the displayed* Cont *routes from tolerance request to window, from window to dyadic observation, from dyadic observation to classifier row, and from classifier row to seal handoff;*
8. *P is the* Pkg *provenance over* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$ *row.*
The carrier stores no host real equality, no quotient stream equality, no selected limit, no completed-real inverse law, and no ambient completeness premise.

Theorem 637.2 (Real approximation-envelope NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$ *surface for Definition 637.1 consists exactly of tolerance-request admission, selected* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet E. Its rows list the tolerance request, finite window, dyadic observation ledger, regular rational classifier row, RealUp seal handoff, transport witnesses, continuation routes, provenance, and local certificate.

The observation side first reads ε and then follows the continuation route to W. The selected window exposes only finite stream data, and D records the dyadic endpoint, radius, and arithmetic ledger obtained from that window.

The classifier side routes the same finite read through R and then to the seal row S. The stored hsame and Cont rows ensure that the classifier handoff is tied to the same tolerance request and window, while P and N record the package provenance and local name.

No remaining coordinate can supply host real equality, quotient stream equality, selected limits, completed-real inverse laws, or ambient completeness. The obligations therefore exhaust the finite Real-and-Completion observation envelope. □

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelopePacket namecert obligations`

Theorem 637.3 (Real approximation-envelope tolerance-window admission)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. Each accepted tolerance request in the row ε reaches the selected* $\mathsf{StreamName}^{\uparrow}$ *finite window W through the same finite support rows H, C, P, N, and the admission path uses no stream coordinate outside the window row displayed in E.*

Proof. Project the rows of E from Definition 637.1. The row ε stores the tolerance request, W stores the selected $\mathsf{StreamName}^{\uparrow}$ finite window, H stores the componentwise hsame transport, C stores the displayed Cont route, P stores the Pkg provenance, and N stores the local NameCert row.

Apply Definition 637.2. Its first two obligations admit the tolerance request and then the selected finite window for the same local packet, while the transport, route, provenance, and naming obligations keep the path inside H, C, P, N.

Since the carrier lists no second stream read and no ambient stream equality, the admitted tolerance request can reach a consumer-visible stream observation only through W. Thus every accepted tolerance request reaches the selected finite window through the displayed rows $\varepsilon, W, H, C, P, N$. □

Theorem 637.4 (Real approximation-envelope dyadic observation soundness)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The selected finite window* W *determines the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* R *along the displayed route*

$$W \longrightarrow D,$$
$$D \longrightarrow R.$$

The determination is internal to the support rows H, C, P, N *and does not use quotient stream equality, host real equality, or a selected limit.*

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope dyadic observation soundness`

Proof. Start with the window admission theorem Definition 637.3. It places the accepted tolerance request at the selected $\mathsf{StreamName}^{\uparrow}$ row W through the local support rows of the envelope.

The NameCert obligations of Definition 637.2 then admit the $\mathsf{DyadicRatCore}^{\uparrow}$ observation row D and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row R for the same packet. The continuation ledger C records the route from W to D and from D to R, while H, P, N keep the route attached to the same finite carrier.

No coordinate of Definition 637.1 records quotient stream equality, host real equality, or a selected limit. The dyadic observation and classifier row are therefore sound exactly as the displayed W to D to R observation route. □

Theorem 637.5 (Real approximation-envelope seal ledger exactness)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The* $\mathsf{Real}^{\uparrow}$ *seal row* S *is admissible exactly as the downstream ledger endpoint of the finite route*

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

with support rows H, C, P, N*. It exports no host real equality, quotient stream equality, selected limit, completed-real inverse law, or ambient completeness principle.*

Proof. First apply Definition 637.3 and Definition 637.4. These identify the admitted route from the tolerance request ε through the selected window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation row D, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row R.

Apply Definition 637.2 to the downstream row. It admits the $\mathsf{Real}^{\uparrow}$ seal handoff and records, through C, that the seal is reached after the classifier row for the same carrier, with H, P, N preserving transport, provenance, and local naming.

The carrier definition lists the seal as a row of the finite packet rather than as a completed real value or an ambient equality principle. Since the negative boundary excludes host real equality, quotient stream equality, selected limits, completed-real inverse laws, and ambient completeness, the row S is exact only as the endpoint of the displayed finite ledger. □

Theorem 637.6 (Real approximation-envelope consumer boundary)**.** *Every diagonal-representative, stationary rational finite-window, Cauchy-modulus, or* $\mathsf{Real}^{\uparrow}$ *classifier consumer of an accepted* $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *packet must read the rows*

$$\varepsilon, W, D, R, S, H, C, P, N.$$

Such a consumer may inspect the selected $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff, but it cannot replace that route by host real equality, quotient stream equality, a selected limit, a completed-real inverse law, or ambient completeness.*

Proof. Begin with the NameCert obligations of Definition 637.2. They force every observation request to pass from ε through the selected window W before any dyadic or regular rational data is consumed.

Diagonal representatives and stationary rational finite-window consumers read the finite window side of the packet. The dyadic ledger D records the observed endpoint data, and the classifier row R records the regular rational readback for the same selected window.

Cauchy-modulus and $\mathsf{Real}^{\uparrow}$ classifier consumers then reach the real-facing boundary through S. The transport, continuation, provenance, and local naming rows keep the seal attached to the same finite observation route.

The carrier inventory is exhausted by the displayed rows. Since none of them is host real equality, quotient stream equality, a selected limit, a completed-real inverse law, or an ambient completeness theorem, no accepting consumer can export such data. □

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope consumer boundary`

Theorem 637.7 (Real approximation-envelope window readback totality)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. Every accepted tolerance request in* E *reaches the selected* $\mathsf{StreamName}^{\uparrow}$ *finite window by the displayed route through* ε*,* W*,* H*,* C*,* P*, and* N*, and the readback uses no stream coordinate except those recorded in that finite window row.*

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope window readback totality`

Proof. The carrier definition in Definition 637.1 displays ε as the tolerance request row, W as the selected $\mathsf{StreamName}^{\uparrow}$ finite-window row, H as the componentwise transport ledger, C as the continuation route ledger, P as provenance, and N as the local certificate.

Apply the NameCert obligation inventory of Definition 637.2. It requires tolerance-request admission, selected $\mathsf{StreamName}^{\uparrow}$ window admission, componentwise hsame transport, displayed Cont route replay, provenance, and local naming for the same packet E.

Finally apply the consumer boundary in Definition 637.6. A consumer may inspect the selected $\mathsf{StreamName}^{\uparrow}$ window but cannot replace that finite route by unrecorded stream access, so the readback is total exactly along the displayed envelope rows. □

Theorem 637.8 (Real approximation-envelope dyadic classifier handoff)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The selected* $\mathsf{StreamName}^{\uparrow}$ *finite window* W *determines the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *along the displayed continuation route*

$$\begin{aligned} \varepsilon &\longrightarrow W, \\ W &\longrightarrow D, \\ D &\longrightarrow R, \\ R &\longrightarrow S. \end{aligned}$$

Lean4 ✓ BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelopeDyadicClassifierHandoff

Proof. By Definition 637.1, the packet stores the $\mathsf{StreamName}^{\uparrow}$ row W, the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, the $\mathsf{RegSeqRat}^{\uparrow}$ row R, the $\mathsf{Real}^{\uparrow}$ row S, and the continuation ledger C in one finite BHist carrier.

The NameCert obligations in Definition 637.2 bind these rows as a single route: tolerance request to selected window, selected window to dyadic observation, dyadic observation to classifier row, and classifier row to seal handoff.

The boundary theorem Definition 637.6 then fixes the consumer-visible handoff. Any diagonal-representative, stationary rational finite-window, Cauchy-modulus, or $\mathsf{Real}^{\uparrow}$ classifier consumer sees the same W, D, R, S chain, so the dyadic, classifier, and seal rows are the displayed continuation of the selected window. □

Theorem 637.9 (Real approximation-envelope transport non-escape)**.** *For an accepted* RealApproximationEnvelope *carrier*

$$E = (\varepsilon, W, D, R, S, H, C, P, N),$$

componentwise hsame *transport of E preserves the tolerance request ε, the selected* $\mathsf{StreamName}^{\uparrow}$ *finite window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row R, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff S. The transported packet exports no host real equality, quotient stream equality, selected limit, completed-real inverse law, or ambient completeness principle.*

Proof. The transport coordinates are part of the carrier itself by Definition 637.1: H records componentwise hsame transports for the tolerance request, window, dyadic observation, classifier row, seal row, provenance, and local name.

Use Definition 637.2 to replay those transport rows together with the displayed Cont routes and Pkg provenance. This keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces attached to the same finite envelope packet after transport.

The non-escape claim is exactly the negative boundary of Definition 637.6. Since every accepted consumer must read the displayed rows of E, and since those rows do not contain host real equality, quotient stream equality, selected limits, completed-real inverse laws, or ambient completeness, componentwise transport cannot export any such object. □

Theorem 637.10 (Real approximation-envelope classifier handoff)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The accepted packet routes the tolerance request ε and the selected* $\mathsf{StreamName}^{\uparrow}$ *finite window W through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D into the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row R, using only the displayed rows $\varepsilon, W, D, R, H, C, P, N$ of the same finite packet.*

Proof. Project the carrier coordinates from Definition 637.1. The packet displays the tolerance request, selected window, dyadic observation, classifier row, componentwise transport ledger, continuation ledger, provenance, and local certificate in one finite BHist object.

Apply the NameCert obligations in Definition 637.2. They require admission of ε, W, D, and R together with replay of the displayed Cont routes for the same local certificate.

The route is therefore the finite chain ε to W, W to D, and D to R. No coordinate outside $\varepsilon, W, D, R, H, C, P, N$ is needed to hand the selected observation to the regular-rational classifier. □

Theorem 637.11 (Real approximation-envelope seal routing)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The* $\mathsf{Real}^{\uparrow}$ *seal row* S *is reachable only after the classifier row* R *for the same finite observation route, and this seal exports no selected limit, quotient stream equality, host real equality, completed-real inverse law, or ambient completeness premise.*

Proof. First apply Definition 637.10. It fixes the observation route from the tolerance request through the selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ row into the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row R.

The carrier and NameCert obligations then place the seal row after the classifier row by the displayed continuation route R to S. Thus any accepted access to S is tied to the same finite observation chain already used by the classifier.

Finally apply the consumer boundary theorem Definition 637.6. Since accepted consumers may inspect the seal handoff but cannot replace the finite route by selected limits, quotient stream equality, host real equality, completed-real inverse laws, or ambient completeness, the seal row exports none of those objects. □

Theorem 637.12 (Real approximation-envelope seal-classifier compatibility)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier handoff row* R *and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *are compatible exactly when both are read through the same displayed finite route*

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

and through the same support rows H, C, P, N*. No second classifier, selected limit, quotient stream equality, host real equality, or ambient completeness coordinate may mediate the seal.*

Proof. Apply Definition 637.10. It identifies the unique classifier-facing route from the tolerance request ε through the selected $\mathsf{StreamName}^{\uparrow}$ window W and the $\mathsf{DyadicRatCore}^{\uparrow}$ observation row D into the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row R.

Apply Definition 637.11. The seal-facing route reaches the $\mathsf{Real}^{\uparrow}$ row S only after the already displayed classifier row R, so the classifier handoff and the seal handoff share the same finite boundary

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S.$$

Finally use Definition 637.9 and Definition 637.6. These exclude selected limits, quotient stream equality, host real equality, completed-real inverse laws, ambient completeness, and unrecorded consumer coordinates, leaving only the displayed route together with H, C, P, N as the finite packet boundary. □

Theorem 637.13 (Real approximation-envelope window-classifier exactness)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The finite window row* W *is the unique public route by which the dyadic observation row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *are exported. Every accepted classifier or seal read factors through the displayed chain*

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

together with the support rows H, C, P, N, and no selected limit, quotient stream equality, host real equality, completed-real inverse law, or ambient completeness premise can replace that route.

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope window classifier exactness`

Proof. The carrier definition Definition 637.1 displays exactly the rows $\varepsilon, W, D, R, S, H, C, P, N$. Thus any public read of a dyadic observation, classifier row, or seal row must be a read of one of the named coordinates of the same finite BHist packet.

Apply Definition 637.7 to the first step. It makes W the total finite-window readback for the tolerance request ε and excludes unrecorded stream coordinates from that read.

Next apply Definition 637.8 and Definition 637.10. These identify the dyadic observation row D and the classifier row R as the displayed continuation of the same selected window, with only H, C, P, N available as support.

Apply Definition 637.11 for the real-facing end of the route. The seal row S is reached only after the classifier row R, so a seal read cannot bypass the window, dyadic observation, or classifier handoff.

Finally use Definition 637.6. It rules out selected limits, quotient stream equality, host real equality, completed-real inverse laws, and ambient completeness as replacement routes, leaving the displayed finite chain as the exact public export path. □

Theorem 637.14 (Real approximation-envelope regularity propagation)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row R propagates regularity to the* $\mathsf{Real}^{\uparrow}$ *seal row S exactly along the finite envelope route*

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

with support rows H, C, P, N. No selected limit, quotient stream equality, host real equality, completed-real inverse law, ambient completeness premise, or second classifier mediates this propagation.

Lean4 ✓ `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope regularity propagation`

Proof. The carrier definition Definition 637.1 gives the only available rows of an accepted envelope. Thus a regularity read starting at R can only move through coordinates already present in E.

Apply Definition 637.4 and Definition 637.10. They place the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row after the tolerance request, finite window, and dyadic observation rows, so the regularity obligation attached to R is already tied to the route $\varepsilon \to W \to D \to R$.

Apply Definition 637.12 and Definition 637.13. These force the real seal row S to be read after the same classifier row and rule out every route that bypasses the finite window, dyadic observation, classifier handoff, and support rows H, C, P, N.

The sibling anchors `RegSeqRatStreamCarrier_regularit_obligation_surface` and `RegSeqRatStreamCarri` name the regularity and seal faces that this envelope exposes. The preceding exactness results show that those faces meet only through the displayed finite chain, not through a selected limit, stream quotient, host equality, completed-real inverse law, ambient completeness premise, or second classifier. □

Theorem 637.15 (Real approximation-envelope obligation coverage)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The carrier definition and the obligation theorems discharge exactly the seven obligation faces for the envelope package: carrier admission, finite-window readback, dyadic observation soundness,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier handoff,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport stability, and consumer non-escape. The discharged package is the finite route*

$$\varepsilon \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow S$$

with support rows H, C, P, N*, and it claims no selected limit, quotient stream equality, host real equality, completed-real inverse law, ambient completeness premise, Lean verification, or standard bridge.*

Proof. Carrier admission is the displayed inventory of Definition 637.1. It supplies exactly the tolerance request, finite window, dyadic observation row, classifier row, seal row, componentwise transport ledger, continuation ledger, provenance row, and local certificate row.

Finite-window readback is Definition 637.7. Dyadic observation soundness and the classifier-facing handoff are supplied by Definition 637.8 and Definition 637.10; together they force the selected $\mathsf{StreamName}^{\uparrow}$ row W to feed D and then R inside the same finite packet.

The real-facing endpoint is Definition 637.11. It places the $\mathsf{Real}^{\uparrow}$ seal row S after the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row R and prevents the seal from becoming an independent completed-real object.

Transport stability is exactly the componentwise hsame boundary of Definition 637.9. The final consumer boundary is Definition 637.13: every accepted classifier or seal read factors through the same finite window route, and no unrecorded equality, selected limit, inverse law, or completeness principle can replace it. □

Theorem 637.16 (Real approximation-envelope primitive scope closure)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The seven obligation faces of Definition 637.15 are scoped exactly by the primitive rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, together with the four sibling surfaces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Every tolerance-window read, dyadic observation, classifier handoff, seal route, componentwise transport, and consumer read factors through those rows. The package cannot use selected limits, quotient stream equality, host real equality, completed-real inverse laws, ambient completeness, Lean verification, or bridge checking.*

Proof. The carrier inventory is Definition 637.1. It exposes only the tolerance request ε, finite window W, dyadic observation D, classifier row R, seal row S, componentwise transport ledger H, continuation ledger C, provenance row P, and local certificate row N.

Admission is supplied by Definition 637.2. Thus the available primitive rows are exactly the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, with the sibling surfaces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ named by the coordinates of E.

The route from tolerance to seal is forced by Definition 637.7, Definition 637.8, Definition 637.11, and Definition 637.12. These results identify the finite chain $\varepsilon \to W \to D \to R \to S$ as the only route available for tolerance-window reads, dyadic observations, classifier handoffs, and seal reads.

Transport cannot leave that primitive scope by Definition 637.9. Finally, Definition 637.15 repacks the carrier, admission, route, transport, and consumer faces into the seven-face package, with selected limits, quotient stream equality, host real equality, completed-real inverse laws, ambient completeness, Lean verification, and bridge checking outside the claim. □

Theorem 637.17 (Real approximation-envelope four-face public export)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. Its public Real-and-Completion export is exactly the four-face packet*

$$\mathsf{StreamName}^{\uparrow}(W) \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow \mathsf{Real}^{\uparrow}(S)$$

over the tolerance request ε*, the componentwise* hsame *rows* H*, the displayed* Cont *routes* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$ *row* N*. Every downstream diagonal-representative, stationary-window, Cauchy-modulus, Real-name, or Completion consumer reaches the L10 phase boundary through those rows, and no quotient stream, host real equality, selected limit, ambient completeness, metric-completeness, arithmetic/order package, or standard bridge row is exported by the envelope.*

Proof. Project the accepted carrier by Definition 637.1. The only public face rows in the packet are the selected $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation D, the $\mathsf{RegSeqRat}^{\uparrow}$ classifier R, and the $\mathsf{Real}^{\uparrow}$ seal S, with ε, H, C, P, N supplying the tolerance, transport, route, provenance, and local certificate support.

Apply Definition 637.2. It admits precisely those face rows and the support rows for the same local $\mathsf{NameCert}$ surface, so the export cannot acquire a detached quotient stream, host equality, selected limit, completeness witness, arithmetic/order package, or bridge coordinate.

Next apply Definition 637.13 and Definition 637.15. These force every classifier or seal read to factor through the finite route from ε to W, then to D, then to R, and finally to S, while preserving the componentwise hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The consumer boundary of Definition 637.6 and the primitive scope closure of Definition 637.16 place the same route at the public edge of the chapter. Downstream, Definition 1120.6 and Definition 1120.9 are the Real L10 four-face exactness and pullback boundary: they consume only the corresponding StreamName, DyadicRatCore, RegSeqRat, and Real faces through finite structural rows. Thus the envelope exports exactly the displayed four-face packet to that boundary and nothing outside it. □

Theorem 637.18 (Real approximation-envelope four-face transport stability)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

and

$$E' = (\varepsilon', W', D', R', S', H', C', P', N')$$

be accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carriers. Suppose the tolerance requests,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows,* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise transport rows, continuation routes, provenance rows, and local* $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$ *rows are matched by componentwise* hsame *through the displayed support rows* H, C, P, N *and* H', C', P', N'*. Then the public four-face export*

$$\mathsf{StreamName}^{\uparrow}(W) \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow \mathsf{Real}^{\uparrow}(S)$$

is transported to the same finite public packet

$$\mathsf{StreamName}^{\uparrow}(W') \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D') \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R') \longrightarrow \mathsf{Real}^{\uparrow}(S')$$

over the matched tolerance, transport, route, provenance, and local certificate rows. This preservation introduces no quotient stream equality, host real equality, selected limit, ambient completeness, metric-completeness, arithmetic/order package, or bridge row.

Proof. Start with Definition 637.9. The componentwise hsame rows keep the matched tolerance, window, dyadic observation, classifier, seal, provenance, and local name inside the two displayed finite carriers, so transport cannot leave the rows H, C, P, N and H', C', P', N'.

Apply Definition 637.13 to both accepted envelopes. It identifies the unique public route from the tolerance request through the selected $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation, and the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row. The matched support rows therefore carry the same route shape on both sides.

Next use Definition 637.14. Regularity reaches the $\mathsf{Real}^{\uparrow}$ seal only through the already matched classifier row, so the transported classifier route and the transported seal route remain a single finite packet rather than two independent public objects.

Finally invoke Definition 637.17. Since that theorem makes the StreamName, DyadicRatCore, RegSeqRat, and Real rows the exact public boundary of the envelope, the componentwise transport preserves precisely the displayed four-face packet and adds none of the excluded quotient, host-equality, limit, completeness, arithmetic/order, or bridge data. □

Theorem 637.19 (Real approximation-envelope public export boundary)**.** *Let*

$$E = (\varepsilon, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *carrier. The public export of E is exactly the four-face route*

$$\mathsf{StreamName}^{\uparrow}(W) \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow \mathsf{Real}^{\uparrow}(S)$$

supported by the displayed tolerance request ε, componentwise hsame *rows H, continuation rows C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$ *row N. This is the complete public Real-and-Completion boundary of the envelope: selected limits, quotient stream equality, host real equality, completed-real inverse laws, ambient completeness, arithmetic/order packages, and bridge data remain outside the export.*

Proof. The public four-face theorem Definition 637.17 identifies the visible packet as the StreamName, DyadicRatCore, RegSeqRat, and Real route over the support rows ε, H, C, P, N. The classifier exactness theorem Definition 637.13 forces every classifier or seal read through that same finite route.

Transport stability is supplied by Definition 637.18. It shows that componentwise hsame preserves the four-face packet without adding quotient, host-equality, limit, completeness, arithmetic/order, or bridge coordinates.

Finally, Definition 1120.6 and Definition 1120.9 give the Real completion boundary consumed by this envelope. They use only the matching StreamName, DyadicRatCore, RegSeqRat, and Real faces, so the accepted envelope exports exactly the displayed public route and nothing beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{RealApproximationEnvelope}^{\uparrow}$ packet is generated from a tolerance request, selected $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, $\mathsf{RegSeqRat}^{\uparrow}$ classifier row, $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 637.1, Definition 637.2, Definition 637.15, Definition 637.16, Definition 637.17, Definition 637.18, and Definition 637.19 bind the primitive finite envelope from tolerance requests through StreamNameUp windows, DyadicRatCore observations, RegSeqRat classifier rows, RealUp seal handoffs, componentwise transport, and the public Real-and-Completion export boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApproximationEnvelope}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealApproximationEnvelopeUp.RealApproximationEnvelope\_window\_classifier\_exactness`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient stream equality, host real equality, completed-real inverse laws, arithmetic/order packages, or bridge checking.

Upgrade path: The remaining route to a higher closure grade is a bridge theorem for the public StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp four-face export.

Chapter 638

A Concrete Naming Certificate for $\mathsf{FiniteWitnessRoute}^{\uparrow}$

The $\mathsf{FiniteWitnessRoute}^{\uparrow}$ interface names the finite route packet that carries closed-observation witness records into Real-and-Completion consumers. It is a concrete $\mathsf{NameCert}$ surface for the passage from a selected observation request through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks to a $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal row, with every transport, continuation, and provenance step present in the finite history.

The route is downstream of chapter 126, chapter 301, chapter 291, chapter 386, and chapter 38. $\mathsf{StreamName}^{\uparrow}$ supplies the finite window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the readback, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the comparison ledger, $\mathsf{CauchyLimitSeal}^{\uparrow}$ supplies the sealed completion boundary, and $\mathsf{Real}^{\uparrow}$ supplies the terminal real-facing seal.

Definition 638.1 (Finite witness-route carrier)**.** *A* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (q, w, r, d, s, h, c, p, n)$$

with the following displayed rows:

1. *q is the closed-observation request row naming the finite witness demand, tolerance scale, and downstream Real-and-Completion consumer;*
2. *w is the selected* $\mathsf{StreamName}^{\uparrow}$ *window inspected for that request;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed through the selected window;*
4. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger containing the dyadic comparisons, radius splits, and finite arithmetic reads used by the route;*
5. *s is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row that receives the completed finite witness route;*
6. *h records componentwise* hsame *transports for the request row, selected window, regular rational readback, dyadic ledger, seal row, provenance, and local name;*
7. *c records the displayed* Cont *routes from request to window, from window to regular rational readback, from readback to dyadic ledger, and from the dyadic ledger to the seal row;*
8. *p is the* Pkg *provenance over* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *n is the local* $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *row.*

The carrier contains no host-level infinite trace, no quotient stream equality, no selected real value, no unrecorded limit, and no consumer route outside the displayed finite packet.

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteTasteGate single carrier alignment`

Theorem 638.2 (Finite witness-route NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *surface for Definition 638.1 consists exactly of carrier admission, request exposure, selected* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger admission,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet R. The rows name the request, selected observation window, rational readback, dyadic ledger, seal, transports, continuations, provenance, and local certificate.

The request and window obligations are read through q and w. They ensure that a consumer sees a finite observation demand and exactly the selected $\mathsf{StreamName}^{\uparrow}$ window used to answer it.

The readback and ledger obligations are read through r and d. The regular rational row supplies the observed sequence data, and the dyadic row supplies the comparison and radius ledger used by the route.

The remaining rows s, h, c, p, n expose the seal handoff, componentwise transports, route replay, package provenance, and local naming. Since every coordinate of the packet is accounted for, these obligations exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteNameCert obligations`

Theorem 638.3 (Finite witness-route non-escape)**.** *Every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *carrier exports witness information only through the finite rows*

$$q, w, r, d, s, h, c, p, n.$$

It exports no host-level infinite trace, no quotient stream equality, no selected real value, no unrecorded limit, and no Real-and-Completion consumer route that bypasses the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project the accepted carrier as $R = (q, w, r, d, s, h, c, p, n)$. The definition lists every exported row of the finite witness route.

The observation side is exhausted by q, w, r, d. These rows record the request, selected stream window, regular rational readback, and dyadic comparison ledger.

The consumer side is exhausted by s, h, c, p, n. These rows record the seal handoff, transport, continuation, provenance, and local name.

No coordinate remains from which to read an infinite trace, quotient stream equality, selected real value, unrecorded limit, or bypass route. The accepted packet therefore has the stated non-escape boundary. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteNonEscape boundary`

Theorem 638.4 (Finite witness-route transport stability)**.** *Let*

$$R = (q, w, r, d, s, h, c, p, n)$$

be an accepted $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *carrier. Any* hsame *transport recorded in* h *preserves the request row, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, seal row, continuation routes, package provenance, and local name as one finite route packet for the same Real-and-Completion consumer.*

Proof. Read the transport row h. Its components are indexed by the coordinates of the carrier, so each displayed row has a matching transport witness.

Transporting q, w, r, d preserves the finite observation side of the route: the request, selected window, regular rational readback, and dyadic ledger stay attached to the same finite demand.

Transporting s, c, p, n preserves the consumer side of the route: the seal handoff, continuation replay, provenance, and local certificate stay attached to that demand. Thus the transported data is again one finite witness-route packet for the same consumer boundary. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteTransport stability`

Theorem 638.5 (Finite witness-route consumer factorization)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that accepts a* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, the seal row* s*, the displayed transports* h*, the continuation routes* c*, the package provenance* p*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *row* n*. The consumer receives the finite witness route, not an independent route ledger.*

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteConsumer factorization`

Proof. Begin with an accepted packet $R = (q, w, r, d, s, h, c, p, n)$. The request row q names the finite demand and the downstream consumer boundary.

The only observation path in the carrier runs through w, r, d. The selected stream window exposes the inspected records, the regular rational row exposes their sequence readback, and the dyadic ledger exposes the comparison data.

The handoff to the consumer then runs through s, h, c, p, n. These rows seal, transport, replay, package, and name the same finite route. Hence any accepting consumer factors through the displayed packet rather than through an independent ledger. □

Theorem 638.6 (Finite witness-route request-window admission)**.** *Let*

$$R = (q, w, r, d, s, h, c, p, n)$$

be an accepted $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet. Every finite witness demand read by a Real-and-Completion consumer must pass through the request row* q *and the selected* $\mathsf{StreamName}^{\uparrow}$ *window row* w*. No consumer may read a route from* R *without first exposing those two rows.*

Proof. Project the carrier of Definition 638.1. The row q names the finite witness demand, tolerance scale, and downstream consumer, while w names the selected stream window inspected for that request.

Apply Definition 638.2. Its request exposure and selected-window admission clauses make q and w part of the local $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ surface.

Thus any accepted consumer route that reads the finite witness demand must open q and then the selected window w. A route that omits either row is not a read of the displayed packet. □

Theorem 638.7 (Finite witness-route regular readback admission)**.** *For an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

every accepted regular rational readback in the route is exposed through the same selected $\mathsf{StreamName}^{\uparrow}$ *window* w *and through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* r*. The dyadic comparison ledger* d *records the finite comparison data, and the route imports neither quotient stream equality nor a selected real.*

Proof. Read the observation rows w, r, d from Definition 638.1. The selected window w is the inspection boundary, r is the regular rational readback exposed through that window, and d is the dyadic ledger for the finite comparisons.

Use Definition 638.2 to attach those three rows to the local certificate surface. The readback admission row is therefore tied to the same selected window that answers the request.

Finally apply Definition 638.3. It excludes quotient stream equality, selected real values, unrecorded limits, and host-level infinite traces from the accepted packet, so the regular readback remains a finite row of the displayed route. □

Theorem 638.8 (Finite witness-route seal-budget exactness)**.** *For an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

any $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal-budget consumer factors through the request row* q*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, the seal row* s*, and the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *rows before receiving the route. The packet exports no hidden observation budget, quotient stream equality, host real equality, selected host limit, countable-choice row, ambient completeness coordinate, independent dyadic ledger, or standard bridge row.*

Proof. Project the carrier in Definition 638.1. Its displayed coordinates are exactly $q, w, r, d, s, h, c, p, n$, with q naming the request, w the selected stream window, r the regular rational readback, d the dyadic ledger, s the seal row, and h, c, p, n the transport, continuation, package, and naming rows.

The local certificate surface is fixed by Definition 638.2. Its clauses expose carrier admission, request exposure, selected-window admission, readback admission, dyadic ledger admission, seal handoff, hsame transport, Cont replay, Pkg provenance, and the local naming row for the same finite packet.

Now apply non-escape, Definition 638.3. The route has no coordinate for host infinite traces, quotient stream equality, host real equality, selected host limits, ambient completeness, metric completion, countable choice, independent dyadic ledgers, or standard bridge data.

Finally use request-window admission, Definition 638.6, and regular readback admission, Definition 638.7. Every seal-budget read enters through q, w, then reads only r, d below that selected window before reaching s, h, c, p, n. Thus every budget coordinate received by a $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer is already one of the displayed packet rows. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteSeal budget exactness`

Theorem 638.9 (Finite witness-route consumer route exactness)**.** *Every Real-and-Completion consumer route that accepts a* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the displayed rows*

$$q, w, r, d, s, h, c, p, n$$

in that dependency order. The seal row s *cannot be consumed unless the request, selected window, regular readback, and dyadic ledger rows* q, w, r, d *have already been exposed in the same finite packet.*

Proof. Begin with Definition 638.5. It sends every accepting $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through the selected window, regular readback, dyadic ledger, seal, transport, continuation, provenance, and local naming rows of the packet.

Attach Definition 638.6. The consumer route cannot reach the observation side without the request row q and the selected window row w.

Next use Definition 638.8. The readback and dyadic comparison rows are the finite rows r and d, and the seal-budget consumer receives no hidden observation budget, quotient stream equality, selected real value, countable-choice row, ambient completeness coordinate, independent dyadic ledger, or standard bridge row.

The seal row s is therefore downstream of q, w, r, d, and the remaining rows h, c, p, n preserve, replay, package, and name the same route. Hence the accepting consumer factors through the displayed dependency order. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteConsumer route exactness`

Theorem 638.10 (Finite witness-route selected-window seal uniqueness)**.** *Let*

$$R = (q, w, r, d, s, h, c, p, n)$$

be an accepted $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet. Any Real-and-Completion consumer route that reads the seal row s must use the same request row q, selected* $\mathsf{StreamName}^{\uparrow}$ *window w,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d. Inside the packet there is no second selected window, readback, dyadic ledger, or seal handoff available for that consumer route.*

Proof. The request-window admission theorem fixes the route entrance. By Definition 638.6, every accepted read of the finite witness demand opens the request row q and the selected window row w.

The regular readback admission theorem then fixes the observation rows below that window. By Definition 638.7, the readback and dyadic comparison data are exactly the displayed rows r and d, and the packet imports neither quotient stream equality nor a selected real value.

Finally apply Definition 638.9. The consumer route reaches the seal row s only after q, w, r, d have been exposed in that dependency order. The non-escape boundary of Definition 638.3 excludes an external equality, selected real, or route ledger that could furnish a second handoff, so the seal read is unique to the displayed packet rows. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteSelectedWindowSeal uniqueness`

Theorem 638.11 (Finite witness-route scoped export factorization)**.** *Every Real-and-Completion consumer that accepts a* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the displayed request row q, the selected* $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row s, and the* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *support rows recorded by*

$$R = (q, w, r, d, s, h, c, p, n).$$

Thus the consumer exports the finite witness route only along the open Real-and-Completion kernel and sibling surface named by those rows.

Proof. The route entrance is the request-window admission of Definition 638.6. It forces any accepting consumer read of the finite witness demand to expose the request row q and the selected window w.

The observation layer is then fixed by Definition 638.7. The readback row is r, the comparison ledger is d, and both are tied to the same selected $\mathsf{StreamName}^{\uparrow}$ window rather than to quotient stream equality or a selected real value.

The consumer handoff is supplied by Definition 638.9. It sends the accepting route through $q, w, r, d, s, h, c, p, n$ in the displayed dependency order.

Finally, Definition 638.10 rules out a second selected window, readback, dyadic ledger, or seal handoff inside the packet. The only remaining support rows are the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ rows, so the consumer factors through exactly the stated scoped export surface. □

Theorem 638.12 (Finite witness-route kernel scope lock)**.** *The scoped route of Definition 638.11 exposes no host infinite trace, quotient stream equality, selected real value, unrecorded limit, countable-choice row, or independent dyadic ledger. Its Real-and-Completion exit surface is only the finite packet*

$$R = (q, w, r, d, s, h, c, p, n)$$

and the sibling rows named by w, r, d, s, h, c, p, n.

Proof. Apply Definition 638.3. It excludes host-level infinite traces, quotient stream equality, selected real values, unrecorded limits, and bypass consumer routes from every accepted $\mathsf{FiniteWitnessRoute}^{\uparrow}$ packet.

Now use Definition 638.11. Any accepting Real-and-Completion consumer must factor through the request, selected window, readback, dyadic ledger, seal, transport, continuation, provenance, and local certificate rows of the displayed packet.

Since the factorization already names the only dyadic ledger row and the non-escape theorem removes every bypass route, there is no remaining coordinate for a choice row or independent ledger. The kernel scope is therefore locked to the finite route and to the sibling rows carried by that route. □

Theorem 638.13 (Finite witness-route scoped support rows)**.** *For every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

the support scope is exactly the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows together with the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces read by* w, r, d, s*. The scope contains no host-level infinite trace, quotient stream equality, selected real value, ambient Real completeness, metric completeness, countable choice, or standard bridge row.*

Proof. Project the carrier through Definition 638.1. Its coordinates are the request row q, selected window w, regular rational readback r, dyadic ledger d, seal row s, transports h, routes c, provenance p, and local certificate n.

Apply Definition 638.2. The certificate surface accounts for carrier admission, request exposure, selected-window admission, readback admission, ledger admission, seal handoff, transport, route replay, package provenance, and the local naming row.

The sibling faces are the ones named by those coordinates: $\mathsf{StreamName}^{\uparrow}$ through w, $\mathsf{RegSeqRat}^{\uparrow}$ through r, $\mathsf{DyadicRatCore}^{\uparrow}$ through d, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ through s. Apply Definition 638.4 to keep those rows attached under the componentwise hsame transports.

Finally use Definition 638.3. It removes host infinite traces, quotient stream equality, selected real values, unrecorded limits, and bypass routes; the same packet has no coordinate for ambient Real completeness, metric completeness, countable choice, or a standard bridge row. The displayed rows therefore give the whole scoped support. □

Theorem 638.14 (Finite witness-route bridged standard surface)**.** *For every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

every bridge-facing consumer must first read the finite request q, the selected $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d, the seal row s, and the support rows $q, w, r, d, s, h, c, p, n$. This is the standard-surface precondition for later bridge checking. It asserts no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, no standard bridge theorem, and no bridgeChecked verification status.*

Proof. First use scoped export factorization. It orders every accepted Real-and-Completion consumer through $q, w, r, d, s, h, c, p, n$ before the route can expose any public exit surface.

Next apply kernel scope lock and scoped support rows. These theorems identify the bridge-facing support as the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows together with the named $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, while excluding host traces, quotient equality, selected real values, countable choice, metric completeness, and standard bridge rows.

Finally compare with the Real-and-Completion scope and public NameCert export. They package the same finite route as the public surface of the certificate. Therefore a later bridge checker may require this surface as input, but the theorem itself does not supply Real completeness or a checked standard bridge. □

Theorem 638.15 (Finite witness-route Real and Completion scope)**.** *Every Phase Real or Completion consumer read that accepts a* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the finite route*

$$q, w, r, d, s, h, c, p, n.$$

The read may use the named $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces carried by those rows, but it receives no independent real value, limit object, completeness principle, metric completion row, quotient stream equality, countable-choice selector, or standard bridge route.*

Proof. Start with the consumer factorization theorem, Definition 638.5. It sends every accepting $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through the selected window, readback, ledger, seal, transport, continuation, provenance, and local naming rows.

The request and observation side is fixed by Definition 638.6 and Definition 638.7. Thus the consumer first reads q, w and then the regular rational and dyadic rows r, d.

The seal side is fixed by Definition 638.9. The consumer reaches s only after q, w, r, d, and the remaining rows h, c, p, n preserve, replay, package, and name that same finite route.

Apply Definition 638.13 for the scope boundary. Since the accepted packet has only the displayed support rows and sibling faces, no Phase Real or Completion read can acquire an independent real value, limit object, completeness principle, metric completion row, quotient equality, countable-choice selector, or standard bridge route. □

Theorem 638.16 (Finite witness-route StreamName source lock)**.** *Every accepted Real-and-Completion consumer read of a* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet first exposes the request row q and the selected* $\mathsf{StreamName}^{\uparrow}$ *window w before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row or* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row is readable. The read has no route that starts from r or d, and it cannot bypass the selected window through a host trace, quotient stream equality, selected real value, countable-choice selector, or independent dyadic ledger.*

Proof. Begin with request-window admission, Definition 638.6. It makes the entrance to the finite witness packet the displayed request row q together with the selected window w.

Next apply regular readback admission, Definition 638.7. The regular rational readback r and dyadic ledger d are admitted only as rows below that selected window, so neither row can be the first readable Real-and-Completion coordinate.

Finally use non-escape, Definition 638.3. Since the carrier contains no host trace, quotient stream equality, selected real, choice selector, or separate dyadic ledger, every accepted consumer read is source-locked to q, w before it reaches r, d. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteStreamName source lock`

Theorem 638.17 (Finite witness-route RegSeq budget exhaustion)**.** *For every accepted Real-and-Completion read of*

$$R = (q, w, r, d, s, h, c, p, n),$$

the rows r and d exhaust the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget segment after the selected window w. The consumer receives no quotient stream equality, selected real, countable choice, or independent dyadic ledger beyond the displayed readback and ledger rows.*

Proof. Use regular readback admission, Definition 638.7, to identify the observation segment below w. The regular rational row is exactly r, and the dyadic comparison ledger is exactly d.

Now apply consumer route exactness, Definition 638.9. A Real-and-Completion consumer reaches the seal side only after the displayed request, window, readback, and dyadic rows have been read in that order.

The kernel scope lock, Definition 638.12, removes every attempt to extend that segment by an ambient equality or selector. Thus the budget after w is exhausted by r, d, with no remaining coordinate for quotient stream equality, a selected real, countable choice, or a second dyadic ledger. □

Lean4 ✓ `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteRegSeq budget exhaustion`

Theorem 638.18 (Finite witness-route Real seal factorization)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the ordered finite route*

$$q, w, r, d, s, h, c, p, n.$$

The consumer first reads the request and selected $\mathsf{StreamName}^{\uparrow}$ *window, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then the seal row, transport, continuation, provenance, and local naming rows. Hence the consumer receives only the finite L10 route and no independent limit object, real value, completeness principle, quotient equality, countable-choice selector, host trace, metric-completion row, or standard bridge route.*

Proof. First apply the StreamName source lock, Definition 638.16. It forces every accepted seal or real consumer to enter through q, w before any regular or dyadic row is readable.

Next use RegSeq budget exhaustion, Definition 638.17. The rows r, d are the whole observation budget below w, so the route cannot add quotient equality, a selected real, countable choice, or a second dyadic ledger before reaching the seal.

Now apply Real and Completion scope, Definition 638.15, together with consumer route exactness, Definition 638.9. These theorems place s, h, c, p, n after q, w, r, d and keep the consumer on the displayed CauchyLimitSeal and Real faces.

Finally invoke scoped support rows and kernel scope lock, Definition 638.13 and Definition 638.12. The accepted packet has no coordinate for an external limit, real, completeness principle, host trace, metric-completion row, or standard bridge, so the ordered finite route is the entire consumer factorization. □

Theorem 638.19 (Finite witness-route seal boundary scope)**.** *For every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

the seal row s *is downstream of the request row* q*, selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, regular* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, dyadic* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, componentwise transport row* h*, continuation row* c*, package provenance* p*, and local naming row* n*. No Real or Completion consumer may read* s *as a boundary supplied by host traces, quotient equality, selected real values, ambient completeness, metric completeness, countable choice, or a standard bridge row.*

Proof. The route entrance is determined by Definition 638.6. Any accepted finite witness demand read opens the request row q and the selected window row w.

The observation data below the window is determined by Definition 638.7. The readback row is r, the comparison ledger is d, and neither quotient stream equality nor a selected real value is imported into the packet.

The consumer dependency order is determined by Definition 638.9 together with Definition 638.10. The seal row s is read only after q, w, r, d, and there is no second selected window, readback, dyadic ledger, or seal handoff inside the carrier.

Finally apply Definition 638.15. The remaining rows h, c, p, n are the only transport, continuation, provenance, and naming rows available to the seal consumer, and the excluded host, quotient, completeness, choice, and bridge coordinates have no public entry point. □

Theorem 638.20 (Finite witness-route four-face exit exactness)**.** *Every current Phase Real and Completion consumer that reaches a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal through an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through exactly the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row carried by that packet. Its support is only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the same finite carrier.*

No such consumer imports host infinite traces, quotient stream equality, selected real values, ambient Real completeness, metric completeness, countable choice, independent dyadic ledgers, or standard bridge rows.

Proof. Apply Definition 638.11. It orders the route through the request row, selected window, regular readback, dyadic ledger, seal row, componentwise transport, continuation, package provenance, and local naming row.

Apply Definition 638.13. The only support rows available along that ordered route are the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows together with the named $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

Now use Definition 638.15. It restricts Phase Real and Completion consumers to the same finite route and excludes independent real values, limit objects, completeness principles, metric completion rows, quotient equality, countable-choice selectors, and standard bridge routes.

Finally apply Definition 638.19. The seal read is downstream of the selected window, readback, ledger, transport, continuation, provenance, and naming rows, so the entrance to the $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal is exactly the four-face route displayed in the statement. □

Theorem 638.21 (Finite witness-route public NameCert export)**.** *For every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

the public certificate $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ *exports exactly the finite packet* R *and the four L10 faces* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *displayed by*

w, r, d, s. Its support is only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows of the same finite carrier.*

No public consumer of this certificate receives a host infinite trace, quotient stream equality, selected real value, ambient Real completeness, metric completeness, countable-choice selector, independent dyadic ledger, or standard bridge row.

Proof. First read the public certificate obligations. By Definition 638.2, the local naming row accounts for carrier admission, request exposure, selected-window admission, readback admission, dyadic ledger admission, seal handoff, componentwise transport, route replay, and package provenance for the same finite packet R.

Next restrict the support rows. Apply Definition 638.13; it identifies the available support as the displayed BHist, hsame, Cont, Pkg, and local NameCert rows together with the named $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

Now factor the seal side through the Real and Completion boundary. By Definition 638.18, every $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer follows the ordered finite route $q, w, r, d, s, h, c, p, n$ and gains no independent limit object, real value, completeness principle, quotient equality, countable-choice selector, host trace, metric-completion row, or standard bridge route.

Finally use Definition 638.20. The public exit surface is exactly the selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal row carried by the packet. Combining these four results repacks the same finite route as the public $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$ export, with no coordinate left for the excluded host, quotient, real-value, completeness, choice, ledger, or bridge data. □

Theorem 638.22 (Finite witness-route choice-refusal boundary)**.** *Every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *Real-and-Completion consumer route remains on the displayed finite rows*

$$q, w, r, d, s, h, c, p, n.$$

No countable-choice selector, selected host limit, quotient stream equality, or independent real value can be inserted between the selected $\mathsf{StreamName}^{\uparrow}$ *window w and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row s.*

Proof. Begin with request-window admission, Definition 638.6. It forces every accepted finite witness demand to expose q and the selected $\mathsf{StreamName}^{\uparrow}$ window w before any downstream read.

Use regular readback admission and consumer route exactness next. By Definition 638.7, the route reads r, d through the same selected window; by Definition 638.9, the seal row s is downstream of q, w, r, d in the same finite packet.

Now apply public NameCert export, Definition 638.21. The exported certificate is exactly the displayed packet and the four L10 faces carried by w, r, d, s.

Finally use non-escape, Definition 638.3. Since no countable-choice selector, selected host limit, quotient stream equality, or independent real value is a coordinate of the carrier, none can be inserted between the selected window and the seal. □

Theorem 638.23 (Finite witness-route L10 terminal four-object handoff)**.** *Every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *consumer that reaches a Real-facing witness route reaches it through the displayed L10 handoff*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{CauchyLimitSeal}^{\uparrow} \text{ or } \mathsf{Real}^{\uparrow}.$$

Equivalently, for every accepted packet

$$R = (q, w, r, d, s, h, c, p, n),$$

the consumer first reads the selected $\mathsf{StreamName}^{\uparrow}$ *finite window* w*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* d*, then the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal boundary or* $\mathsf{Real}^{\uparrow}$ *terminal seal row* s*. The remaining rows* h, c, p, n *transport, replay, package, and name that same finite route, and no Real-facing witness is exported outside this four-object path.*

Proof. First apply consumer factorization, Definition 638.5. It places every accepting $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer on the packet rows w, r, d, s, h, c, p, n rather than on an external witness source.

The entrance is fixed by request-window admission, Definition 638.6. The selected finite window is w, so the route cannot begin at a quotient stream, host trace, selected real, or hidden selector.

The observation segment is fixed by regular readback admission, Definition 638.7, and RegSeq budget exhaustion, Definition 638.17. These identify r as the regular readback and d as the dyadic comparison ledger below the same selected window.

Finally use seal-budget exactness, Definition 638.8, and public NameCert export, Definition 638.21. The seal row s is the downstream CauchyLimitSeal or Real terminal face, while h, c, p, n only transport, replay, package, and name the already displayed route. □

Theorem 638.24 (Finite witness-route no-choice Real completion boundary)**.** *The L10 handoff in Definition 638.23 exports no quotient stream, hidden selector, countable-choice row, host real equality, ambient completeness row, selected host limit, independent dyadic ledger, or standard bridge route. Its Real-facing witness is only the public route through* w, r, d, s, h, c, p, n*.*

Proof. Begin with the choice-refusal boundary, Definition 638.22. It keeps every accepted Real-and-Completion consumer on the displayed finite rows and blocks countable-choice selectors, selected host limits, quotient stream equality, and independent real values between w and s.

Next apply seal boundary scope, Definition 638.19. The seal row is downstream of q, w, r, d, h, c, p, n, so it cannot be replaced by an ambient completeness row, host real equality, metric completion row, or standard bridge.

Finally use scoped support rows, Definition 638.13, together with public NameCert export, Definition 638.21. The certificate exposes only the finite packet and the four L10 faces, leaving no coordinate for the excluded quotient, selector, choice, host-equality, completeness, ledger, or bridge data. □

Theorem 638.25 (Finite witness-route cofinal-window factorization)**.** *Every cofinal-window read admitted by an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet factors through the same displayed L10 route*

$$q, w, r, d, s, h, c, p, n.$$

The read enters through the $\mathsf{StreamName}^{\uparrow}$ *finite window* w*, obtains its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* r*, checks its* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons through* d*, and reaches the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *terminal row through* s*. It cannot bypass the* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows through a quotient stream, ambient real, hidden selector, independent ledger, or host completeness principle.*

Proof. The source of a cofinal-window read is controlled by the StreamName source lock, Definition 638.16. Any accepted Real-and-Completion read must expose q, w before r or d is readable.

The regular and dyadic layers are then controlled by regular readback admission, Definition 638.7, and RegSeq budget exhaustion, Definition 638.17. They force the cofinal readback through r and the comparison ledger through d below the same selected window.

The seal side is controlled by Real seal factorization, Definition 638.18, and seal boundary scope, Definition 638.19. The route reaches only the CauchyLimitSeal or Real row s carried by the packet.

Apply the L10 terminal four-object handoff, Definition 638.23, to assemble the same ordered path. The preceding source, readback, ledger, and seal constraints leave no bypass around the $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{DyadicRatCore}^{\uparrow}$ rows. □

Theorem 638.26 (Finite witness-route CofinalModulusSeal handoff)**.** *Every accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *Real-facing packet that consumes a cofinal modulus route reads* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *through the same selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *packet, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row already carried by*

$$R = (q, w, r, d, s, h, c, p, n).$$

No cofinal modulus handoff may insert a quotient stream, selected host limit, countable-choice selector, ambient completeness row, detached window service, or separate completion operator between the finite witness route and the seal boundary.

Proof. Start with cofinal-window factorization, Definition 638.25. It places every cofinal-window read admitted by R on the displayed rows $q, w, r, d, s, h, c, p, n$, with entrance through w, readback through r, dyadic comparison through d, and seal access through s.

Next apply the L10 terminal four-object handoff, Definition 638.23. The Real-facing consumer therefore reaches the terminal side only through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, while h, c, p, n transport, replay, package, and name that same finite route.

Finally use the CofinalModulusSeal L10 exit boundary, Definition 1198.45. Its public exit is the finite cofinal modulus route and it excludes quotient streams, selected host limits, countable choice, ambient completeness, detached windows, and separate completion operators. Combining the two finite route descriptions gives the stated handoff through the same packet rows and support structures. □

Theorem 638.27 (Finite witness-route Stream-RegSeq budget factorization)**.** *For an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n),$$

the selected $\mathsf{StreamName}^{\uparrow}$ *window* w *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r *are the only budget-bearing entrance into the* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *consumer route. Any Real-facing or completion-facing cofinal modulus read admitted by* R *must pass first through* q, w*, then through* r, d*, and then through the seal and support rows* s, h, c, p, n*.*

Proof. Use request-window admission, Definition 638.6. It makes q, w mandatory before a Real-and-Completion consumer reads any finite witness demand.

Then apply regular readback admission, Definition 638.7. The regular rational readback is r, the dyadic comparison ledger is d, and both sit below the same selected $\mathsf{StreamName}^{\uparrow}$ window rather than below quotient stream equality or a selected real value.

It remains to connect this entrance with the cofinal modulus route. The handoff theorem Definition 638.26 consumes $\mathsf{CofinalModulusSeal}^{\uparrow}$ through the same w, r, d, s, h, c, p, n rows. Thus the StreamName window and RegSeqRat readback are the only budget-bearing entry coordinates available to that route. □

Theorem 638.28 (Finite witness-route Real Completion readiness)**.** *The combined* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *route through* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *is ready as a Phase Real and Completion paper target over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*: its consumer surface is the finite packet* $q, w, r, d, s, h, c, p, n$ *together with the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. This readiness does not claim quotient stream equality, selected host limits, countable choice, ambient completeness, or a* StandardBridge *route.*

Proof. First combine the CofinalModulusSeal handoff, Definition 638.26, with Stream-RegSeq budget factorization, Definition 638.27. These results identify the selected window and readback entrance and keep the cofinal modulus consumer on the finite packet rows.

Now apply the no-choice Real completion boundary, Definition 638.24. It states that the L10 handoff exports no quotient stream, hidden selector, countable-choice row, host real equality, ambient completeness row, selected host limit, independent dyadic ledger, or standard bridge route.

The two positive factorization results and the refusal boundary together give a precise paper target for later Real-and-Completion formalization: the route is exposed through the displayed finite rows and support structures, while the excluded quotient, choice, host-limit, completeness, and bridge claims remain outside the chapter. □

Theorem 638.29 (Finite witness-route Real completion exact-boundary handoff)**.** *Let an accepted* $\mathsf{FiniteWitnessRoute}^{\uparrow}$ *packet*

$$R = (q, w, r, d, s, h, c, p, n)$$

be consumed by an accepted RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N).$$

After the governing seal, classifier, witness-ledger, and budget rows L, K, J, S *of* B *have been displayed, the handoff identifies the route faces* w, r, d, s *with the exact-boundary faces* W, R, D, E*. The consumer remains on the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the two packets. It receives no quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, metric-completeness row, independent dyadic ledger, or standard bridge row.*

Proof. First use the L10 terminal four-object handoff, Definition 638.23. It places the finite route on the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ through w, r, d, s, with h, c, p, n carrying the transport, continuation, provenance, and local naming rows.

Next apply the Real Completion readiness theorem, Definition 638.28, together with the no-choice Real completion boundary, Definition 638.24. These results keep the route on the same finite packet rows and exclude quotient streams, hidden selectors, countable-choice rows, host real equality, ambient completeness, selected host limits, independent dyadic ledgers, and bridge routes.

Finally consume the route through the exact-boundary packet. The four-face exactness theorem Definition 1120.6 identifies the only boundary faces after L, K, J, S as W, R, D, E. The terminal-consumer exactness theorem Definition 1120.13 keeps the consumer on H, C, P, N, and the no-completeness-escape theorem Definition 1120.14 refuses quotient, host-equality, selected-limit, choice, completeness, and bridge rows. Composing these route descriptions gives the stated identification of w, r, d, s with W, R, D, E under the displayed structural rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$.

Constructive story: A FiniteWitnessRoute$^{\uparrow}$ packet is generated from a finite witness request, selected StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ observation ledger, CauchyLimitSeal$^{\uparrow}$ or Real$^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 638.1, Definition 638.17, Definition 638.28, and Definition 638.29 bind the mature FiniteWitnessRoute$^{\uparrow}$ scope: the existing finite carrier, RegSeq budget exhaustion, completion-facing consumer surface, downstream route integration, and the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ route through the same BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWitnessRoute}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FiniteWitnessRouteUp.FiniteWitnessRouteRegSeq\_budget\_exhaustion`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient stream equality, infinite traces, independent route ledgers, Lean verification, or bridge checking.

Upgrade path: No further paper-side theory grade remains; later work is optional bridge synchronization or downstream consumption of the same finite route.

Chapter 639

A Concrete Naming Certificate for CompilerOptimizationEndomorphism$^{\uparrow}$

The CompilerOptimizationEndomorphism$^{\uparrow}$ interface packages source-to-source compiler optimisation as a finite accepted endomorphism over one local NameCert row. It realizes the Cook compile-frontier route of chapter 9417 as a before/after compiler trace whose accepted rows can be read by the certificate compiler of chapter 473, the composition trace of chapter 679, the closed-term substitution compiler of chapter 540, and the cellular audit surfaces of chapter 934 and chapter 937. It consumes the compiler-morphism, certificate-compiler, classifier-morphism, BHist, hsame, Cont, Pkg, and local naming rows already used by compiler and decompiler certificates, but it does not import host program equality or a semantic optimisation theorem.

Falsifiable prediction: If the semantic-preservation row is removed, a frontier-improving packet can no longer certify that the before and after compiler traces decode to the same substrate observation row.

Independence witness: certificatecompiler, compilercompositiontrace, closedtermsubstitution-compiler, cellularautomaton, and cellulartrustsubstrate: none carries the joint before/after optimization endomorphism plus frontier-coordinate effect. CertificateCompiler exposes one compiler edge surface, CompilerCompositionTrace exposes sequential composition, ClosedTermSubstitutionCompiler exposes the closed syntactic boundary, CellularAutomaton exposes the local orbit ledger, and CellularTrustSubstrate exposes substrate trust rows; the optimization endomorphism alone binds the before trace, after trace, semantic-preservation row, and frontier-coordinate improvement in one NameCert packet.

Definition 639.1 (CompilerOptimizationEndomorphism carrier)**.** *A* CompilerOptimizationEndomorphism$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, B, A, G, E, L, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* NameCert *row that is also the target naming row of the optimisation;*
2. *B is the before compiler trace row, read through the same stage order as the Cook pipeline stages of section 9417.2;*
3. *A is the after compiler trace row, displaying the optimized trace over the same source and target naming row;*

4. *G is an accepted endomorphism graph on S, displaying source-to-source compiler edges from B to A rather than a host optimiser function;*
5. *E is the same-source landing row recording that every displayed graph edge starts in the carrier of S and lands back in the same carrier;*
6. *L is the classifier-preservation row recording that source classifier membership is transported along the displayed endomorphism graph;*
7. *F is the frontier-coordinate row recording that the after trace improves a displayed compile-frontier coordinate while still decoding to the same substrate observation row as the before trace;*
8. *H records componentwise* hsame *preservation for the source row, before and after trace rows, graph row, same-source landing row, classifier row, frontier row, continuation rows, package row, and local naming row;*
9. *C is the displayed* Cont *route replay row, pairing every source continuation read used by G with its routed post-optimisation continuation inside the same certificate S;*
10. *P names the dependency surface* $\mathsf{NameCertMorphism}^{\uparrow}$, $\mathsf{CertificateCompiler}^{\uparrow}$, $\mathsf{CompilerCompositionTrace}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionCompiler}^{\uparrow}$, $\mathsf{ClassifierMorphism}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompilerOptimizationEndomorphism}^{\uparrow}}$ *row.*

The carrier is an accepted NameCert endomorphism packet. It has no coordinate for host program equality, denotational semantics, hidden cost improvement, global category laws, or a semantic optimiser correctness theorem outside the displayed semantic-preservation row.

Lean4 ✓ `BEDC.Derived.CompilerOptimizationEndomorphismUp.CompilerOptimizationEndomorphismTasteGate`
`single carrier alignment`

Theorem 639.2 (CompilerOptimizationEndomorphism NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompilerOptimizat}}$ *surface for Definition 639.1 consists exactly of carrier admission, before-trace exposure through B, after-trace exposure through A, accepted same-source graph edges on S, source-to-source landing through E, classifier preservation through L, semantic-preservation and frontier-coordinate exposure through F,* hsame *preservation through H, displayed continuation replay through C, dependency provenance through P, and the local naming row N.*

Proof. Project the finite packet O. The source row, before trace, after trace, graph, same-source landing row, classifier-preservation row, frontier row, transport row, continuation replay row, provenance row, and local name are all displayed coordinates.

The endomorphism condition is also a projection. Both the edge source and edge target are read from the same NameCert row S, so no additional target certificate can be introduced by the obligation surface.

Classifier, semantic-preservation, hsame, and continuation obligations are read from L, F, H, and C. The package row P records the compiler, composition, closed-substitution, cellular, and trust-substrate dependencies, and N names the local certificate. Since no other coordinate is present, these rows exhaust the local NameCert obligations. □

Theorem 639.3 (CompilerOptimizationEndomorphism composition)**.** *Let*

$$O_1 = (S, B_1, A_1, G_1, E_1, L_1, F_1, H_1, C_1, P_1, N_1)$$

and

$$O_2 = (S, B_2, A_2, G_2, E_2, L_2, F_2, H_2, C_2, P_2, N_2)$$

be accepted CompilerOptimizationEndomorphism$^{\uparrow}$ *packets over the same source* NameCert *row* S*. Their displayed composite is the finite packet whose before trace is* B_1*, whose middle trace is* $A_1 = B_2$ *as a displayed compiler-composition handoff, whose after trace is* A_2*, whose graph row is the relational composite of* G_1 *and* G_2*, whose same-source landing row chains* E_1 *with* E_2*, whose classifier-preservation row chains* L_1 *with* L_2*, whose semantic-preservation and frontier-coordinate row chains* F_1 *with* F_2*, and whose* hsame*,* Cont*,* Pkg*, and local naming rows are obtained by replaying the displayed rows of* O_1 *and* O_2 *in order.*

Proof. Project the two accepted packets. Both expose the same source and target naming row S, so the middle landing of the first graph is exactly the source row consumed by the second graph.

Compose the graph rows relationally. A displayed source row is related to a displayed final row when a middle row of S is named by an edge of G_1 followed by an edge of G_2.

The landing, classifier, and frontier rows compose along the same middle row. The E_i rows keep both edges inside S, the L_i rows transport classifier membership first across G_1 and then across G_2, and the F_i rows keep the before and after traces semantically pinned to the same substrate observation while recording the displayed frontier-coordinate effect.

Finally replay the hsame, Cont, Pkg, and local naming rows in the displayed order. The result is again a finite endomorphism packet over S, with no appeal to an ambient optimiser semantics. □

Theorem 639.4 (CompilerOptimizationEndomorphism non-escape)**.** *Every consumer read of an accepted* CompilerOptimizationEndomorphism$^{\uparrow}$ *packet factors through*

$$S,\ B,\ A,\ G,\ E,\ L,\ F,\ H,\ C,\ P,\ N.$$

A request for host optimisation semantics, host program equality, denotational equivalence beyond the displayed semantic-preservation row, hidden cost improvement, a global compiler category law, or an undisplayed target certificate is outside the carrier and is not exported as NameCert evidence.

Proof. Read the request through the displayed continuation route C. If it asks for the source certificate, before trace, after trace, graph, same-source landing row, classifier-preservation row, semantic-preservation and frontier-coordinate row, hsame transport, continuation replay, package provenance, or local name, the carrier supplies exactly the corresponding coordinate.

Any other request leaves the tuple O. The carrier contains no coordinate for a semantic optimiser theorem outside F, host equality of programs, denotational model, hidden cost comparison, ambient category law, or separate target certificate.

Thus the accepted packet exports only the finite NameCert endomorphism surface. Optimisation correctness available to downstream compiler or decompiler certificates is the displayed preservation route through $B, A, G, E, L, F, H, C, P, N$, not a host-level theorem. □

Closure status for NameCert$_{\text{CompilerOptimizationEndomorphism}^{\uparrow}}$**.**

Constructive story: A CompilerOptimizationEndomorphism$^{\uparrow}$ packet is generated from finite BHist rows for one source NameCert, before and after compiler traces, an accepted endomorphism graph, same-source landing, classifier preservation, frontier-coordinate preservation, and structural rows. The optimisation route is replayed by Cont from before trace to after trace inside the same certificate surface, while hsame transports source, trace, graph, landing, classifier, and frontier rows componentwise. Pkg records the compiler, composition, substitution, classifier, cellular, and trust-substrate dependencies. Thus optimization correctness is the displayed preservation row inside the packet, not host program equality or denotational semantics.

Theory closure: TheoryGate(NameCert$_{\text{CompilerOptimizationEndomorphism}^{\uparrow}}$, seed)

Scope closed: This chapter binds the endomorphism carrier, NameCert obligations, composition theorem, and non-escape boundary for $\mathsf{CompilerOptimizationEndomorphism}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompilerOptimizationEndomorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim host optimization semantics, program equality, hidden cost improvement, global compiler category laws, or an undisplayed target certificate.

Upgrade path: Obligation closure requires Lean encodings of the endomorphism carrier, same-source composition, preservation rows, and non-escape theorem.

Chapter 640

A Concrete Naming Certificate for $\mathsf{ObserverKernelTrace}^{\uparrow}$

The $\mathsf{ObserverKernelTrace}^{\uparrow}$ surface realizes observer identity as a finite trace packet rather than as a subject behind the trace. It packages history rows, history-sameness transport, continuation reads, licensed probe and signature observations, provenance, and a local naming row into one NameCert boundary. A consumer may compare observer traces through the displayed packet, but it receives no observer-substance coordinate and no host equality of subjects.

$\mathsf{ObserverKernelTrace}^{\uparrow}$ supplies the no-subject finite trace surface consumed by chapter 477 before pairwise observer-history comparison is licensed.

Definition 640.1 (ObserverKernelTrace carrier)**.** *An* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (T, H, C, B, S, L, P, N)$$

with the following displayed rows:

1. *T is the finite observer-trace ledger, whose entries are* BHist *rows recording the histories available to an observer-facing consumer;*
2. *H records componentwise* hsame *transport for trace rows, observation rows, licensed readback rows, provenance rows, and the local naming row;*
3. *C records the displayed* Cont *routes by which a consumer reads trace rows, transports, probe bundles, signature events, provenance, and local naming;*
4. *B is the finite* ProbeBundle *observation row attached to the same trace ledger;*
5. *S is the* SigRel *observation row attached to the bundled probe reads;*
6. *L is the no-subject ledger recording that every exported observer read is one of the displayed trace, transport, route, probe, signature, provenance, or local naming rows;*
7. *P is the* Pkg *provenance over T, H, C, B, S, L;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for an observer-substance type, subject identity row, hidden owner of the observations, or host equality principle outside BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* Pkg*, and* NameCert*.*

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTraceTasteGate single carrier alignment`

Theorem 640.2 (ObserverKernelTrace NameCert obligations)**.** *For the carrier of Definition 640.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for* O*, trace-ledger stability, componentwise* hsame *transport, displayed* Cont *route replay, licensed* ProbeBundle *and* SigRel *observation exactness, no-subject ledger exactness,* Pkg *provenance, and local naming exactness.*

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTraceNameCert obligations`

Proof. Carrier admission is projection from the finite packet O. The trace ledger T, transports H, routes C, probe row B, signature row S, no-subject ledger L, package row P, and local naming row N are the displayed coordinates of the carrier.

The stability and replay obligations are componentwise. Transport through H preserves the trace and observation rows, while the routes in C replay only the listed reads.

Probe and signature exactness come from B and S, and the no-subject ledger L records every exported observer read before it reaches the consumer. Since the carrier has no coordinate beyond T, H, C, B, S, L, P, N, the local naming certificate is exhausted by the stated finite obligations. □

Theorem 640.3 (ObserverKernelTrace no-subject non-escape)**.** *Every accepted* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *packet exposes observer identity only through its finite* BHist *trace rows,* hsame *transport,* Cont *routes, licensed* ProbeBundle *readback, licensed* SigRel *readback, no-subject ledger,* Pkg *provenance, and local* NameCert *row. It exports no observer-substance carrier, subject identity coordinate, hidden owner of the trace, or host equality of subjects.*

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTrace non escape`
↪ variant `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTrace no subject non escape`

Proof. Project the accepted packet and inspect the coordinates of Definition 640.1. The only rows available to a consumer are T, H, C, B, S, L, P, N.

Apply the NameCert obligations from Definition 640.2. They identify transport, route replay, probe readback, signature readback, no-subject ledgering, package provenance, and local naming as the whole certificate surface.

A requested subject carrier would require a coordinate outside the displayed tuple. Because no such coordinate is part of the packet, the accepted boundary cannot export observer-substance, subject identity, hidden ownership, or host equality data. □

Theorem 640.4 (ObserverKernelTrace public export boundary)**.** *The public export of an accepted* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *packet is exactly the finite route*

$$T, H, C, B, S, L, P, N.$$

It exports trace rows, componentwise transport, displayed continuation routes, probe bundles, signature observations, the no-subject ledger, package provenance, and local naming, and it exports no subject carrier, host equality of subjects, hidden owner, or global observation equivalence.

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTrace public export boundary`

Proof. Project the carrier of Definition 640.1. Its displayed rows are precisely T, H, C, B, S, L, P, N, so the public surface starts from this finite tuple.

Apply the NameCert obligations of Definition 640.2. They identify trace-ledger stability, componentwise hsame transport, displayed Cont replay, probe and signature exactness, no-subject ledgering, package provenance, and local naming as the whole certificate route.

Finally use Definition 640.3. A subject carrier, host equality of subjects, hidden owner, or global observation equivalence would require a coordinate absent from the carrier, so the public export is exactly the displayed route. □

Theorem 640.5 (ObserverKernelTrace public bridge route)**.** *For an accepted* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *packet, the public bridge route from the packet to downstream observer-history consumers is exactly*

$$T, H, C, B, S, L, P, N.$$

The probe-bundle row B and signature row S are visible before any observer-history identity consumer reads the trace, and the route carries no subject carrier, host subject equality, hidden owner, or observer-substance coordinate outside the displayed packet.

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTrace public bridge route`

Proof. Project the accepted carrier of Definition 640.1. Its finite rows are T, H, C, B, S, L, P, N, and no additional subject coordinate is part of the carrier.

Use Definition 640.4 to identify the public export with the same finite tuple. The probe and signature rows B and S are listed in that export before the no-subject ledger, provenance, and local naming rows are consumed by downstream observers.

Finally apply Definition 640.3. Any subject carrier, host subject equality, hidden owner, or observer-substance coordinate would have to be a packet row outside T, H, C, B, S, L, P, N, which the carrier and public export do not contain. □

Theorem 640.6 (ObserverKernelTrace mature consumer coverage)**.** *Every ObserverHistoryIdentity consumer and every observer-history consumer of an accepted* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *packet receives exactly the displayed trace ledger T, componentwise transport row H,* Cont *replay row C,* ProbeBundle *row B,* SigRel *row S, no-subject ledger L,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}$ *row N. The consumer receives no subject-substance coordinate, host-subject equality row, hidden owner row, or global observation-equivalence row.*

Lean4 ✓ `BEDC.Derived.ObserverKernelTraceUp.ObserverKernelTrace mature consumer coverage`

Proof. Project the accepted carrier of Definition 640.1. The available rows are exactly T, H, C, B, S, L, P, N, and the definition lists no subject-substance or host-subject equality coordinate.

Apply Definition 640.2. It identifies trace-ledger stability, componentwise transport, displayed replay, licensed probe and signature observations, no-subject ledgering, provenance, and local naming as the complete certificate surface.

Use Definition 640.4 and Definition 640.5. The public export and bridge route both expose the same finite tuple before any ObserverHistoryIdentity or observer-history consumer reads the packet.

Finally invoke Definition 640.3. A subject-substance coordinate or host-subject equality row would have to be outside the displayed tuple, so mature consumer coverage is exactly the finite observer-kernel trace surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObserverKernelTrace}^{\uparrow}$ packet is generated from finite BHist trace rows, componentwise hsame transport, displayed Cont routes, licensed ProbeBundle and SigRel observations, a no-subject ledger, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 640.1, Definition 640.2, Definition 640.3, Definition 640.4, Definition 640.5, and Definition 640.6 bind the carrier, NameCert obligation rows, no-subject non-escape surface, public export boundary, public bridge route, and downstream observer-history consumer coverage for observer-kernel traces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverKernelTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close a Lean theorem, a host observer model, subject identity, hidden ownership, global observation equivalence, or equality of subjects.

Upgrade path: Bridge checking requires a Lean bridge target for the same trace, probe, signature, transport, replay, provenance, and local NameCert packet.

Chapter 641

A Concrete Naming Certificate for $\mathsf{CauchyModulusExtraction}^{\uparrow}$

The $\mathsf{CauchyModulusExtraction}^{\uparrow}$ packet records the finite observation-budget surface that a Real-completeness consumer uses before applying the diagonal route of chapter 9238. It turns a regular Cauchy input over $\mathsf{RegSeqRat}^{\uparrow}$ into explicit witness schedules, selected stream windows, and dyadic tolerance rows, while keeping the resulting modulus read inside displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{NameCert}$ data.

It consumes the finite modulus-indexed schedule exposed by chapter 593: $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$ supplies the scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ family row, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal handoff that this chapter repackages as an extracted modulus.

Definition 641.1 (Cauchy modulus-extraction carrier)**.** *A* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the positive tolerance request read through the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledgers;*
2. *B is the finite observation budget selected for the requested tolerance and diagonal scale;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected under that budget;*
4. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row naming the regular Cauchy witnesses consumed by the diagonal route;*
5. Mu *is the extracted modulus row, recording the natural threshold produced from E, B, W, S;*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison and half-radius ledger used to justify the threshold;*
7. *H records componentwise* hsame *transports for the tolerance request, budget, window family, schedule row, modulus row, dyadic ledger, package provenance, and local name;*
8. *C records the displayed* Cont *routes from tolerance to budget, from budget to windows, from windows to schedule reads, from schedule reads to* Mu*, and from* Mu *to the* $\mathsf{Real}^{\uparrow}$ *diagonal consumer;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the same tolerance request, finite budget, inspected windows, schedule row, extracted modulus, and dyadic ledger to a common Real-completion surface. The carrier contains no host-level limit, no quotient stream equality, no ambient completeness theorem, and no modulus chosen outside the displayed finite budget.

Theorem 641.2 (Cauchy modulus-extraction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}$ *surface for Definition 641.1 consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N)$*;*
2. *budget exposure stating that* B *is the only finite observation budget available for the tolerance request* E*;*
3. *schedule extraction stating that* W *and* S *are read through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows before the modulus row* Mu *is exported;*
4. *dyadic soundness stating that* Q *supplies the positive half-radius and comparison ledgers needed by the extracted threshold;*
5. *ledger exactness stating that all transport, continuation, package, and naming reads factor through* H, C, P, N *and no external modulus selector is present.*

Lean4 ✓ `BEDC.Derived.CauchyModulusExtractionUp.CauchyModulusExtractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 641.1. The tuple lists the tolerance request, finite budget, inspected windows, regular Cauchy schedule, extracted modulus, dyadic ledger, transports, continuations, provenance, and local name.

Budget exposure uses the row B. A consumer may inspect only the finite windows listed by that budget, so the modulus row is tied to displayed observation data rather than to an ambient convergence choice.

Schedule extraction uses W and S. The $\mathsf{StreamName}^{\uparrow}$ windows identify the inspected rational observations, while the $\mathsf{RegSeqRat}^{\uparrow}$ schedule row records the regular Cauchy witnesses consumed by the diagonal Real-completion route.

Dyadic soundness is the role of Q. It stores the half-radius split and comparison ledger that justify the threshold recorded in Mu for the requested tolerance.

Finally, H, C, P, N repack the same rows under componentwise transport, route replay, package provenance, and local naming. Since no remaining coordinate contains an external modulus selector, these five obligations exhaust the local certificate surface. □

Theorem 641.3 (Cauchy modulus-extraction budget-window coverage)**.** *For an accepted* CauchyModulusExtraction *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

every tolerance request E *that is accepted by the packet is routed through the finite budget row* B *before any* $\mathsf{StreamName}^{\uparrow}$ *window in* W *is inspected and before any* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S *is used to export* Mu*. Thus the budget-window surface covers the whole route from requested tolerance to inspected stream data.*

Proof. Project the accepted carrier to the displayed tuple. The rows available between the tolerance request and the modulus export are exactly the finite budget B, the window family W, and the schedule row S, together with the dyadic and certificate rows used later in the packet.

Use the NameCert obligations of Definition 641.2. Budget exposure identifies B as the only finite observation budget available for E, and schedule extraction states that W and S are read before Mu is exported.

It follows that a consumer cannot inspect a StreamName window or use a RegSeqRat schedule directly from the tolerance request. The accepted route first passes through B, then through W and S, so the finite budget-window surface covers the route to the extracted modulus. □

Theorem 641.4 (Cauchy modulus-extraction window induction)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

suppose a finite budget row B^+ *extends* B *by one displayed* $\mathsf{StreamName}^{\uparrow}$ *window while preserving the tolerance request* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S*, the dyadic ledger row* Q*, and the continuation route from* Mu *to the* $\mathsf{Real}^{\uparrow}$ *consumer. Then the extracted modulus route through*

$$E, B^+, W, S, \mathsf{Mu}, Q, H, C, P, N$$

remains accepted: the added window is consumed inside the finite budget-window surface, and the modulus export still passes through the same dyadic ledger, transport, replay, provenance, and local naming rows.

Proof. Use finite-window induction on the displayed extension from B to B^+. The base case is the budget-window coverage theorem, Definition 641.3, which places the accepted route through B, W, S before Mu is exported.

For the one-window step, read the added $\mathsf{StreamName}^{\uparrow}$ window as another row of the same finite observation budget. Because B^+ preserves E, S, Q and the continuation route to the $\mathsf{Real}^{\uparrow}$ consumer, schedule extraction still reads the window family through W and the regular Cauchy witnesses through S before the modulus row is exposed.

Apply dyadic ledger positivity from Definition 641.5. The threshold exported by Mu is justified by the same positive half-radius and comparison ledger Q, so the additional window supplies no second threshold selector.

Finally use the Real seal boundary of Definition 641.6. The completion-facing handoff remains downstream of $B^+, W, S, Q, \mathsf{Mu}, H, C, P, N$ and still has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, ambient completeness, or an external modulus choice. □

Theorem 641.5 (Cauchy modulus-extraction dyadic ledger positivity)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the dyadic ledger row Q *supplies the positive half-radius and comparison rows used by the extracted modulus row* Mu*. No threshold can be exported from the packet unless that export is justified through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* Q*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusExtractionUp.CauchyModulusExtractionCarrier dyadic ledger positivity`

Proof. Project the carrier. The displayed row Q is the packet coordinate assigned to the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison and half-radius ledger, and Mu is the only extracted modulus row.

Apply dyadic soundness from Definition 641.2. That obligation binds the positive half-radius split and comparison data needed by the threshold to Q.

Since the carrier contains no second dyadic ledger and no external threshold selector, any exported threshold must use the same Q row. Therefore the positivity and comparison data required by Mu are supplied by the displayed ledger and cannot be bypassed inside the packet. □

Theorem 641.6 (Cauchy modulus-extraction RealUp seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N)$$

reaches the RealUp seal only after the finite budget B, StreamName windows W, RegSeqRat schedule S, dyadic ledger Q, and extracted modulus row Mu *have been displayed by the same packet. The seal-facing route imports no quotient stream equality, host real equality, trichotomy, selected limits, ambient completeness, or threshold chosen outside the displayed rows.*

Proof. Project the carrier and read the continuation row C. The displayed routes go from tolerance to budget, from budget to windows, from windows to schedule reads, from schedule reads to Mu, and from Mu to the $\mathsf{Real}^{\uparrow}$ diagonal consumer.

Use budget-window coverage to place B, W, S before the extracted modulus, and use dyadic ledger positivity to place Q before the threshold export. These rows account for the data consumed by the RealUp-facing route before the seal is reached.

Finally apply ledger exactness from Definition 641.2. The carrier has no coordinate for quotient stream equality, host real equality, trichotomy, selected limits, ambient completeness, or an outside threshold selector, so the RealUp seal boundary is exactly the displayed finite packet boundary. □

Theorem 641.7 (Cauchy modulus-extraction budget soundness)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the extracted modulus row Mu *is sound for every diagonal Real-completion consumer that reads the same finite budget B,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness schedule S, and dyadic comparison ledger Q. The consumer receives a natural threshold justified by the displayed observation budget and receives no host-selected convergence modulus, quotient stream representative, or ambient completeness witness.*

Proof. Read the tolerance request E and the budget row B. The budget determines the finite observation family available to the extraction packet.

Read the window and schedule rows W, S. They identify the StreamName observations and RegSeqRat witnesses that the real-completeness diagonal route may consume for the requested scale.

Use the dyadic ledger Q to justify the half-radius comparisons attached to those reads. The extracted row Mu records the resulting natural threshold and is transported by H along the continuation routes in C.

The package and naming rows P, N expose exactly that finite packet. Because the carrier has no coordinate for an external modulus, a host-selected limit, or quotient equality, the handoff to the $\mathsf{Real}^{\uparrow}$ diagonal consumer is precisely the displayed budget-sound modulus row. □

Theorem 641.8 (Cauchy modulus-extraction budget threshold uniqueness)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the displayed rows E, B, W, S, Q determine the exported threshold row Mu *for every consumer that reads the same finite budget. No second modulus row, host-selected convergence modulus, quotient stream representative, or ambient completeness witness can determine a different threshold inside the packet.*

Proof. Project the displayed packet. The only tolerance request is E, the only observation budget is B, the only window family is W, the only regular schedule is S, and the only dyadic comparison ledger is Q.

Use the NameCert obligations. Budget exposure fixes B as the source of the observations, schedule extraction fixes W, S before the modulus is exported, and dyadic soundness fixes Q as the ledger attached to the threshold.

Apply budget soundness from Definition 641.7. Since the consumer receives Mu only as the threshold justified by those finite rows and the carrier has no second selector coordinate, the threshold exported inside the packet is unique for that budget route. □

Theorem 641.9 (Cauchy modulus-extraction StreamName RegSeqRat factorization)**.** *Every consumer route from the tolerance request E to the extracted modulus row* Mu *factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row S of the same packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N).$$

The factorization reads the budget row B first, then the window and schedule rows, then the dyadic ledger Q, and only then exports Mu*.*

Proof. Begin with carrier admission. The tuple presents B, W, S, Q as the only finite observation, window, schedule, and dyadic comparison rows available before Mu.

Use budget exposure and schedule extraction from Definition 641.2. These clauses force the route from E to the modulus to pass through the finite budget and then through the StreamName and RegSeqRat rows.

Finally use budget soundness. The Real-completion consumer receives the modulus only after the dyadic ledger has justified the threshold for those rows, so the StreamName and RegSeqRat factorization is part of every accepted route to Mu. □

Theorem 641.10 (Cauchy modulus-extraction RealUp seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer of an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet reads the same finite budget, window, schedule, dyadic, and modulus rows*

$$B, W, S, Q, \mathsf{Mu}$$

before reaching the RealUp handoff. The packet exports no seal-facing route through quotient stream equality, selected limits, host real equality, trichotomy, ambient completeness, or an external modulus selector.

Proof. Project the carrier and the continuation row C. The displayed continuation routes include the path from tolerance to budget, from budget to windows, from windows to schedule reads, from schedule reads to Mu, and from Mu to the $\mathsf{Real}^{\uparrow}$ diagonal consumer.

Apply the StreamName-RegSeqRat factorization and budget-threshold uniqueness established above. They show that the consumer reaches the same exported modulus only through B, W, S, Q.

Finish with ledger exactness from the NameCert obligations and budget soundness from Definition 641.7. Since the finite packet has no coordinate for quotient equality, selected limits, host real equality, trichotomy, ambient completeness, or an outside selector, no RealUp seal-facing consumer can escape that route. □

Theorem 641.11 (Cauchy modulus-extraction budget monotonicity)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

let B^+ *be a displayed finite budget row containing the same tolerance request* E*, the same* $\mathsf{StreamName}^{\uparrow}$ *windows needed by* W*, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S*, and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* Q*. Then the extracted modulus row* Mu *remains sound for every* $\mathsf{Real}^{\uparrow}$ *diagonal consumer that reads through* B^+*. The larger displayed budget may contain additional visible observations, but it cannot change the extracted threshold, introduce a second modulus selector, or supply quotient stream equality, a host limit, or ambient completeness.*

Proof. Project the accepted carrier to the tuple $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$. The local rows identify the tolerance request, the original finite budget, the inspected windows, the regular Cauchy schedule, the extracted threshold, and the dyadic comparison ledger.

Apply the NameCert obligations of Definition 641.2. Budget exposure fixes the displayed observation source for E, schedule extraction fixes W, S before Mu is exported, and dyadic soundness fixes Q as the comparison ledger attached to the threshold.

Now read the consumer through B^+. By assumption, the rows of B^+ contain exactly the tolerance request, window reads, schedule row, and dyadic ledger already used by B, W, S, Q for the extracted threshold.

Use budget soundness from Definition 641.7 on those common rows. The row Mu is therefore still the threshold justified by the displayed budget data, and the consumer receives it without any host-selected convergence modulus or quotient stream representative.

The extra rows of B^+ are exhausted by coordinate inspection. They are not read by Mu, have no selector coordinate in the carrier, and do not add a host limit or ambient completeness witness. Hence enlarging the finite observation budget preserves the same budget-sound modulus handoff. □

Theorem 641.12 (Cauchy modulus-extraction transport stability)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

any componentwise hsame *transport of* $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$ *preserves the extracted modulus row* Mu *and preserves the finite budget source* B *from which that row is read. The transported packet exports the same tolerance request, finite observation budget, StreamName windows, RegSeqRat schedule, dyadic comparison ledger, continuation routes, package provenance, and local naming row, and it exports no host-selected convergence modulus.*

Proof. Project the accepted carrier to its displayed tuple. The row H records componentwise transport for the tolerance request, budget, windows, schedule row, modulus row, dyadic ledger, provenance, and local name.

Apply the NameCert obligations of Definition 641.2. Carrier admission fixes the tuple, budget exposure fixes B as the finite source for the tolerance request, schedule extraction fixes W, S before Mu is exported, and ledger exactness routes all transport through H, C, P, N.

Finally use budget soundness from Definition 641.7. The Real-completion handoff receives Mu only as the threshold justified by the displayed budget and schedule rows, so componentwise transport preserves both the modulus row and its finite budget source. □

Theorem 641.13 (Cauchy modulus-extraction selector stability)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

componentwise hsame *transport of the tolerance request E, finite budget B,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger Q, and modulus row* Mu *preserves the same selected modulus for any* $\mathsf{Real}^{\uparrow}$ *diagonal consumer. The transported consumer reads the same finite selector surface and receives no external modulus selector, quotient stream representative, selected limit, host real equality, or ambient completeness witness.*

Proof. Start with the carrier definition. The selector data visible before the $\mathsf{Real}^{\uparrow}$ handoff are exactly $E, B, W, S, Q, \mathsf{Mu}$, and the transport row H is componentwise over those coordinates.

Apply the NameCert obligations and budget soundness. Carrier admission fixes the tuple, budget exposure fixes the finite source B, schedule extraction fixes W, S, dyadic soundness fixes Q, and budget soundness identifies Mu as the threshold justified by those displayed rows.

Use transport stability from Definition 641.12. Since the transported packet preserves both the modulus row and the finite budget source, the selected modulus seen by a diagonal consumer is the same selected row Mu.

Finally apply the StreamName–RegSeqRat factorization and the $\mathsf{Real}^{\uparrow}$ seal non-escape theorem. The consumer reaches the handoff only through the finite budget, stream windows, regular schedule, dyadic ledger, and exported modulus, so no outside selector or completion object can alter the selected threshold. □

Theorem 641.14 (Cauchy modulus-extraction finite-budget handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing handoff from an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet factors through the finite budget row B before reading the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger Q, and the exported modulus row* Mu*. Thus the completion-facing route is a finite-budget handoff rather than a direct appeal to host convergence, quotient stream equality, selected limits, trichotomy, or ambient completeness.*

Proof. Read the accepted packet through Definition 641.1. The continuation row C lists the route from tolerance to budget, from budget to windows, from windows to schedule reads, from schedule reads to Mu, and from Mu to the $\mathsf{Real}^{\uparrow}$ diagonal consumer.

Apply the NameCert obligations and budget soundness. Budget exposure makes B the only finite observation source for E, while budget soundness ties the exported modulus to the displayed budget, windows, schedule, and dyadic ledger.

Next use the StreamName–RegSeqRat factorization. It places W and S after the budget row and before the modulus export, and it places Q as the dyadic ledger required by the extracted threshold.

Finish with the $\mathsf{Real}^{\uparrow}$ seal non-escape theorem. Since every seal-facing route reads through B, W, S, Q, Mu and the carrier has no coordinate for host convergence or quotient stream equality, every completion-facing handoff factors through the finite budget before the remaining rows are consumed. □

Theorem 641.15 (Cauchy modulus-extraction real diagonal handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$ *diagonal or completion consumer of an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N)$$

may read the extracted modulus row Mu *only after the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger Q, and the finite budget row B have been displayed through the same local packet.*

Proof. Project the accepted carrier. The rows available before the modulus handoff are the tolerance request E, finite budget B, window family W, schedule row S, and dyadic ledger Q.

Apply the NameCert obligations of Definition 641.2. Budget exposure places B before any window read, schedule extraction places W, S before the export of Mu, and dyadic soundness places Q as the comparison ledger attached to the threshold.

Use budget soundness from Definition 641.7. Since the handoff to the diagonal consumer is precisely the displayed budget-sound modulus row, any $\mathsf{Real}^{\uparrow}$ completion route reads Mu only after the windows, schedule, dyadic comparison ledger, and finite budget rows have appeared. □

Theorem 641.16 (Cauchy modulus-extraction ledger exactness)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the ledger is exact for the route from tolerance request to finite budget, StreamName windows, RegSeqRat schedule, extracted modulus, dyadic comparison ledger, and local NameCert:

$$E, B, W, S, \mathsf{Mu}, Q, H, C, P, N.$$

It excludes quotient stream equality, host real equality, selected limits, trichotomy, ambient completeness, and any modulus selector not displayed by the finite packet.

Proof. Project the accepted carrier to the ten displayed rows. The route begins at the tolerance request E and reaches the finite budget B, windows W, schedule S, extracted modulus Mu, dyadic ledger Q, transports H, routes C, provenance P, and local name N.

Apply the NameCert obligations of Definition 641.2. They bind carrier admission, budget exposure, schedule extraction, dyadic soundness, and transport, continuation, package, and naming reads to the same finite tuple.

Finally use budget soundness from Definition 641.7. The exported modulus is justified by the displayed observation budget and carries no coordinate for quotient stream equality, host real equality, selected limits, trichotomy, ambient completeness, or an outside selector, so the ledger is exact. □

Theorem 641.17 (Cauchy modulus-extraction route exactness)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

every accepted consumer route from the tolerance request E to the extracted modulus row Mu *factors through exactly the displayed rows*

$$E, B, W, S, \mathsf{Mu}, Q, H, C, P, N.$$

Thus the route passes through the finite budget, StreamName window family, RegSeqRat schedule row, dyadic comparison ledger, componentwise transports, continuation routes, package provenance, and local naming row, and it has no additional Real-completion coordinate.

Proof. Project the accepted carrier to its displayed tuple. The carrier presentation supplies the tolerance request, budget row, inspected windows, schedule row, extracted modulus, dyadic ledger, transports, continuations, provenance, and local name.

Apply the local NameCert obligations of Definition 641.2. Carrier admission identifies the tuple, budget exposure fixes B, schedule extraction fixes W, S, dyadic soundness fixes Q, and ledger exactness fixes H, C, P, N as the only remaining certificate rows.

Finally use budget soundness from Definition 641.7. Since the handoff to the diagonal consumer is the displayed budget-sound modulus row and the carrier contains no extra selector, any accepted route from E to Mu is exactly the route through those ten rows. □

Theorem 641.18 (Cauchy modulus-extraction window schedule exhaustion)**.** *For an accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the inspected $\mathsf{StreamName}^{\uparrow}$ *window family* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S *exhaust all schedule data available to a Real-completion diagonal consumer. The consumer may use the StreamName surface of chapter 126, the RegSeqRat schedule surface of chapter 301, and the Real carrier surface of chapter 38, but within this extraction packet those surfaces are reached only through* W *and* S*.*

Proof. The carrier first exposes the finite budget B attached to the tolerance request E. This budget bounds which StreamName observations and regular Cauchy witnesses are visible to the diagonal consumer.

The window row W is the only StreamName row in the packet. By the StreamName naming surface cited in chapter 126, window reads available to the consumer are therefore the reads displayed in W.

The schedule row S is the only RegSeqRat schedule row in the packet. By the RegSeqRat naming surface cited in chapter 301, regular Cauchy witness reads available to the consumer are therefore the reads displayed in S.

The Real carrier surface cited in chapter 38 receives the diagonal handoff through the continuation rows in C. Because those continuations start from the displayed budget, windows, and schedule row, no further schedule data can be introduced at the Real-completion boundary. □

Theorem 641.19 (Cauchy modulus-extraction dyadic ledger non-escape)**.** *For an accepted* CauchyModulusExtractio *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the dyadic comparison row Q *is the only positive half-radius and comparison ledger exported to the modulus row* Mu*. The extraction packet exports no quotient stream, host-selected convergence modulus, selected limit, host real equality, or ambient completeness witness.*

Proof. Read the dyadic row Q from the carrier. It is the coordinate assigned to the positive half-radius split and to the comparisons used by the extracted threshold.

Apply dyadic soundness from Definition 641.2. That obligation identifies Q as the ledger needed by Mu and ties it to the displayed budget, windows, and schedule row.

Use budget soundness from Definition 641.7 to inspect the exported handoff. The consumer receives the natural threshold justified by Q and by the finite rows B, W, S.

The carrier has no coordinate for a quotient stream, host-selected convergence modulus, selected limit, host real equality, or ambient completeness witness. Hence none of those objects can escape through the modulus row. □

Theorem 641.20 (Cauchy modulus-extraction budget fiber exhaustion)**.** *For an accepted* CauchyModulusExtractio *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

a $\mathsf{Real}^{\uparrow}$ *diagonal or completion consumer lies in the budget fiber over the displayed tolerance request* E *and finite budget* B *exactly when its route first reads* B*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows* W *and*

$\mathsf{RegSeqRat}^{\uparrow}$ *schedule* S*, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* Q*, and exports* Mu *only through* H, C, P, N*. No quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, or external modulus selector supplies another element of that budget fiber.*

Proof. Start with a consumer lying in the displayed budget fiber. Apply the NameCert obligations of Definition 641.2 to identify the accepted carrier rows, the exposed budget, the schedule extraction surface, the dyadic soundness row, and the transport, continuation, package, and naming rows.

The finite route is then forced by budget soundness from Definition 641.7 and route exactness from Definition 641.17. The consumer first reads B, then uses the StreamName–RegSeqRat factorization through W and S, then reads the dyadic ledger Q before exporting Mu through H, C, P, N.

Conversely, any route with exactly that read order is accepted by the same packet because each row is one of the displayed NameCert rows and the handoff to the Real diagonal surface uses the continuation rows already present in C.

It remains to exclude other fiber elements. Apply window-schedule exhaustion from Definition 641.18 and dyadic-ledger non-escape from Definition 641.19. These results leave no coordinate through which quotient stream equality, host real equality, selected limits, trichotomy, ambient completeness, or an external modulus selector can enter the budget fiber. □

Theorem 641.21 (Cauchy modulus-extraction obligation closure package)**.** *For an accepted* $\mathsf{CauchyModulusExtract}$ *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the $\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}$ *obligation package consumed by the* $\mathsf{Real}^{\uparrow}$ *completion route is exactly the carrier row* E*, the finite budget row* B*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* Q*, the extracted modulus row* Mu*, and the structural rows* H, C, P, N*. Every* $\mathsf{Real}^{\uparrow}$ *diagonal consumer must pass through those rows before reading the modulus handoff. No such consumer obtains quotient stream equality, selected limits, trichotomy, host real equality, ambient completeness, or an external modulus selector from the package.*

Proof. Begin with the NameCert obligations of Definition 641.2. Carrier admission fixes the displayed packet, budget exposure fixes B, schedule extraction fixes W, S, dyadic soundness fixes Q, and ledger exactness fixes H, C, P, N as the structural route.

Budget soundness from Definition 641.7 then identifies Mu as the threshold justified by B, W, S, Q. The StreamName–RegSeqRat factorization of Definition 641.9 forces every route from E to Mu through the window and schedule rows.

Apply the Real diagonal handoff of Definition 641.15 and ledger exactness from Definition 641.16. These two results place the Real-facing read after the displayed budget, window, schedule, dyadic, modulus, transport, continuation, package, and naming rows.

Finally use budget-fiber exhaustion from Definition 641.20. It rules out every remaining route through quotient stream equality, selected limits, trichotomy, host real equality, ambient completeness, or an external modulus selector, so the listed rows are the whole obligation package for the Real-completion route. □

Theorem 641.22 (Cauchy modulus-extraction public export)**.** *For every accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the public $\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}$ *export is exactly the carrier admission, finite budget exposure,* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *factorization,* $\mathsf{DyadicRatCore}^{\uparrow}$ *positivity and non-escape ledger,*

$\mathsf{Real}^{\uparrow}$ *seal boundary, route exactness, window schedule exhaustion, budget fiber exhaustion, and obligation closure package already displayed by this chapter.*

Thus downstream $\mathsf{Real}^{\uparrow}$ *and Completion consumers may cite one public export instead of rebuilding the finite budget, dyadic, stream, schedule, seal, transport, continuation, provenance, and naming rows separately. The export contains no quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, or external modulus selector.*

Proof. First read the accepted carrier through Definition 641.2. This fixes $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$ as the only rows available to the local naming certificate and identifies the carrier admission and finite budget exposure.

Next assemble the stream and dyadic surfaces. Apply Definition 641.9, Definition 641.5, and Definition 641.19. These results put the window row, regular schedule, positive dyadic ledger, and dyadic non-escape boundary in the same displayed packet.

Now attach the Real-facing route. Use Definition 641.10, Definition 641.17, and Definition 641.18. They force every seal-facing consumer through the exact finite route and exhaust the window schedule before any modulus handoff is readable.

Finally combine Definition 641.20 with Definition 641.21. The first theorem exhausts the budget fiber over the displayed tolerance request, and the second repacks all named rows as the local obligation package. Therefore the public export is precisely the listed finite surface, with all excluded completion objects absent from the carrier. □

Theorem 641.23 (Cauchy modulus-extraction Real-completion consumer bridge)**.** *For every accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

every Real-completion consumer route that reads the extracted modulus row Mu *factors through exactly the displayed rows* $B, W, S, Q, \mathsf{Mu}, H, C, P, N$ *after carrier admission through* E*. The* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surfaces reached by the route are precisely the sibling surfaces cited by Definition 641.22; no route obtains a host convergence modulus, quotient stream, selected limit, host real equality, ambient completeness, or choice-selected witness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusExtractionUp.CauchyModulusExtractionUp StdBridge`

Proof. Start with the budget-window coverage theorem, Definition 641.3. It places every accepted tolerance request E behind the finite budget B before any window or schedule row can contribute to Mu.

Next use dyadic ledger positivity, Definition 641.5. The extracted modulus is justified by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger Q, so the consumer receives a ledgered threshold rather than an external selector.

Now apply the StreamName–RegSeqRat factorization of Definition 641.9. The sibling surfaces chapter 126 and chapter 301 are reached only through the local W and S rows in this packet.

Finally cite RealUp seal non-escape, Definition 641.10, together with the public export of Definition 641.22. These results identify the local Real-facing surface with the sibling surface chapter 38, keep the completion-facing read inside $B, W, S, Q, \mathsf{Mu}, H, C, P, N$, and rule out host convergence moduli, quotient streams, selected limits, host real equality, ambient completeness, and choice-selected witnesses. □

Theorem 641.24 (Cauchy modulus-extraction dyadic Real consumer factorization)**.** *For every accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-completion consumer that reads the extracted modulus row* Mu *factors first through the finite budget row* B*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule row* S*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* Q*. The handoff to the Real-facing consumer exports only this displayed dyadic ledger route with* H, C, P, N *attached; it exports no host modulus, quotient stream equality, selected limit, trichotomy, ambient completeness, or external real-completion witness.*

Proof. Start with the NameCert obligations of Definition 641.2. They identify the accepted packet with the rows $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$ and leave no additional carrier coordinate for a Real consumer.

Use budget-window coverage from Definition 641.3. Any consumer read of Mu must first pass through the finite budget B and the displayed window row W, so the request is already inside the local ledger route.

Next apply dyadic ledger positivity and StreamName–RegSeqRat factorization, Definition 641.5 and Definition 641.9. These force the route through the regular schedule row S and the dyadic ledger row Q before the extracted modulus can be exposed.

Finally use the Real-completion consumer bridge, Definition 641.23, and carrier non-escape from Definition 641.10. The bridge supplies the Real-facing handoff, while non-escape excludes host moduli, quotient stream equality, selected limits, trichotomy, ambient completeness, and external completion witnesses from the exported packet. □

Theorem 641.25 (Cauchy modulus-extraction mature package summary)**.** *For every accepted* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N),$$

the finite Real-completion consumer surface is the public export, the Real-completion consumer bridge, the dyadic Real consumer factorization, route exactness, budget-fiber exhaustion, and the no-ambient-completeness boundary of this chapter assembled as one package. A downstream $\mathsf{Real}^{\uparrow}$ *or completion consumer reads* $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$ *in that finite order, obtains the extracted modulus only through the displayed budget-window-dyadic route, and obtains no quotient stream equality, host real equality, selected limit, trichotomy, ambient completeness, or external modulus selector.*

Proof. Start with Definition 641.22. It collects carrier admission, budget exposure, window and schedule rows, dyadic ledger, Real seal boundary, route exactness, window exhaustion, budget-fiber exhaustion, and obligation closure into the public surface of M.

Use Definition 641.23. The bridge sends Real-completion consumers through $B, W, S, Q, \mathsf{Mu}, H, C, P, N$ after admission through E, so the extracted modulus is not read from an external host selector.

Apply Definition 641.24. It records the reusable factorization through B, W, S, Q before the Real-facing consumer can read Mu.

Finally apply Definition 641.17, Definition 641.20, and Definition 641.10. Route exactness fixes the order of the finite rows, budget-fiber exhaustion leaves no second budget source, and the RealUp boundary excludes quotient stream equality, host real equality, selected limits, trichotomy, ambient completeness, and external modulus data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusExtraction}^{\uparrow}$ packet is generated from a positive tolerance request, finite observation budget, StreamNameUp windows, RegSeqRatUp witness schedule, extracted modulus row, dyadic comparison ledger, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 641.1, Definition 641.2, Definition 641.7, Definition 641.9, Definition 641.19, Definition 641.15, Definition 641.21, Definition 641.22, Definition 641.23, Definition 641.24, and Definition 641.25 bind the mature finite Real-completion consumer surface for the modulus-extraction packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusExtraction}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CauchyModulusExtractionUp.CauchyModulusExtractionCarrier\_namecert\_obligations`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, a public standard bridge, or Lean verification.

Upgrade path: No further upgrade: the chapter is at the top of the formal bridge axis.

Chapter 642

A Concrete Naming Certificate for $\mathsf{ProofTermErasureTrace}^{\uparrow}$

The $\mathsf{ProofTermErasureTrace}^{\uparrow}$ surface records a type-checking run whose proof evidence is erased while its typing membership and reduction compatibility remain visible. It sits between derivation traces and compiler packets: the carrier keeps the proof-term row, the erased-term row, the Ext typing row, the erasure route, the subject-reduction compatibility row, the transport rows, provenance, and local naming in one finite certificate.

The trace consumes the derivation-route surface of chapter 598 and the preservation boundary of chapter 483. $\mathsf{ProofTermErasureTrace}^{\uparrow}$ does not create its own typing judgment; it packages the erased public row only after those sibling rows have supplied membership and reduction compatibility.

Definition 642.1 (Proof-term erasure trace carrier)**.** *A* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (p, e, T, E, R, H, C, P, N)$$

with the following visible coordinates:

1. *p is the proof-term row whose membership is checked by the typing classifier;*
2. *e is the erased-term row exported to consumers after proof evidence is forgotten;*
3. *T is the typing* Ext *row that admits p as a checked term;*
4. *E is the displayed erasure* Cont *route from p to e;*
5. *R is the subject-reduction compatibility row tying the typing membership of p to the erased public read;*
6. *H records the componentwise* hsame *transports for proof terms, erased terms, typing rows, erasure routes, and subject-reduction rows;*
7. *C records the displayed* Cont *routes consumed by the erasure and compatibility readings;*
8. *P is the* Pkg *provenance over* BHist, Ext, Cont, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$, $\mathsf{TypedReductionNormalForm}^{\uparrow}$, hsame, *and* $\mathsf{NameCert}$ *source certificates;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureTrace}^{\uparrow}}$ *row.*

The local classifier relates two carriers only when their proof-term, erased-term, typing, erasure-route, subject-reduction, transport, provenance, and local naming rows compare componentwise through the displayed routes in H and C.

Lean4 ✓ BEDC.Derived.ProofTermErasureTraceUp.ProofTermErasureTraceTasteGate single carrier alignment

Theorem 642.2 (Proof-term erasure trace NameCert obligations)**.** *For the carrier of Definition 642.1, the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureTrace}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission: the packet admits exactly the displayed rows* $p, e, T, E, R, H, C, P, N$*;*
2. Ext *membership preservation: any accepted public read of* e *is backed by the typing row* T *for the proof term* p *and the erasure route* E*;*
3. *subject-reduction compatibility: the row* R *records that the erased public read is compatible with the* Ext-Cont *closure law used by the proof-term trace;*
4. *erased-term non-escape: the carrier exports no host conversion relation, quotient term equality, normalization oracle, proof irrelevance principle, or erased proof witness outside* e*;*
5. *local ledger exactness: every consumer read factors through* H*,* C*,* P*, and* N *together with the visible proof, erasure, typing, and compatibility rows.*

Proof. Carrier admission is the projection of the displayed finite tuple. The coordinates $p, e, T, E, R, H, C, P, N$ account for the proof term, erased term, typing membership, erasure route, subject-reduction compatibility, transports, continuation reads, package provenance, and local naming row.

For Ext membership preservation, read T as the classifier admission of the proof term p and E as the displayed erasure route to e. The erased row is public only through this route, so its membership read is anchored in the original typing row rather than in a host erasure judgment.

Subject-reduction compatibility is the role of R. It records that the typing row and erasure continuation are consumed through the same Ext-Cont closure discipline used by the surrounding type-checking trace.

Erased-term non-escape follows by carrier inspection. None of the coordinates is a host conversion relation, quotient equality on terms, normalization oracle, proof irrelevance principle, or hidden proof witness exported after erasure.

Finally, local ledger exactness repacks the public reads through H, C, P, N. The transports and continuation rows state how the proof, erasure, typing, and compatibility data are replayed, while provenance and the local naming row record the finite ledger boundary. □

Theorem 642.3 (Proof-term erasure trace typed-reduction boundary)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet*

$$Z = (p, e, T, E, R, H, C, P, N),$$

the erased public row e *is exported only through the typing* Ext *row* T*, the erasure* Cont *route* E*, and the subject-reduction compatibility row* R*, replayed through* H, C, P, N*. The packet exports no host conversion relation, quotient term equality, proof irrelevance principle, semantic erasure-correctness theorem, or erased proof witness outside the displayed row* e*.*

Proof. Project the carrier of Definition 642.1. The public erasure surface contains the proof row p, erased row e, typing row T, erasure route E, subject-reduction row R, and the structural rows H, C, P, N.

Apply Definition 642.2. Its membership-preservation clause anchors e in the typing row T and erasure route E, while its subject-reduction clause keeps the public read compatible with R.

The same obligation theorem lists erased-term non-escape and local ledger exactness. Since the carrier has no host conversion, quotient equality, proof irrelevance, semantic erasure-correctness,

or hidden proof-witness coordinate, the typed-reduction boundary is exactly T, E, R, H, C, P, N together with the displayed erased row. □

Theorem 642.4 (Proof-term erasure trace Ext membership factorization)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet*

$$Z = (p, e, T, E, R, H, C, P, N),$$

every public membership read of the erased row e factors through the displayed Ext *row T, the erasure route E, and the local rows H, C, P, N of Definition 642.1. It cannot be certified by a host typing judgment, an erased proof witness, or an external conversion relation outside the packet.*

Proof. Project the carrier of Definition 642.1. The proof term p, erased row e, and typing row T are displayed together, so an erased membership read starts from the same finite packet rather than from a host typing predicate.

Apply Definition 642.2. Its Ext membership-preservation row states that accepted public reads of e are backed by T and by the erasure route E.

The remaining local rows H, C, P, N supply the transport, continuation replay, provenance, and local naming ledger. Since no coordinate names host typing, proof-witness export, or external conversion, the factorization is exhausted by the displayed rows. □

Theorem 642.5 (Proof-term erasure trace erasure-route replay)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet $Z = (p, e, T, E, R, H, C, P, N)$, the erasure route from the proof-term row p to the erased public row e is replayed only by the displayed* Cont *row E together with the route ledger C and transport row H. Replaying the route cannot introduce a second erasure function, a normalization oracle, or proof-irrelevance data outside Z.*

Proof. Read the erasure coordinate of Definition 642.1. The route from p to e is the displayed Cont row E, and the general continuation replay ledger is the coordinate C.

Transport is componentwise through H. Thus moving the proof term, erased term, typing row, and compatibility row transports the same erasure route rather than selecting a second route.

Apply the non-escape and ledger-exactness rows of Definition 642.2. They exclude normalization oracles, proof irrelevance, hidden proof witnesses, and host conversion data from the replay surface. □

Theorem 642.6 (Proof-term erasure trace public-row ledger exhaustion)**.** *For any accepted* ProofTermErasureTrace *packet $Z = (p, e, T, E, R, H, C, P, N)$, every public row exported to a downstream consumer is one of the displayed proof-term, erased-term, typing, erasure-route, subject-reduction, transport, continuation, provenance, or local naming rows of Definition 642.1. The ledger has no additional public row for host conversion, quotient term equality, semantic erasure correctness, proof irrelevance, or hidden proof evidence.*

Lean4 ✓ `BEDC.Derived.ProofTermErasureTraceUp.ProofTermErasureTrace public row ledger exhaustion`

Proof. Project the packet Z. Its public inventory is exactly $p, e, T, E, R, H, C, P, N$, with T carrying Ext membership, E and C carrying displayed Cont replay, R carrying subject-reduction compatibility, and H, P, N carrying transport, provenance, and local naming.

Apply Definition 642.2. Carrier admission gives the full inventory, erased-term non-escape removes undisplayed proof and conversion rows, and local ledger exactness forces downstream reads through the same packet.

Therefore the downstream ledger is exhausted by the displayed rows. Any requested host conversion, quotient equality, semantic erasure theorem, proof irrelevance principle, or hidden proof evidence is outside the accepted public-row surface. □

Theorem 642.7 (Proof-term erasure trace proof-row obligation)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet*

$$Z = (p, e, T, E, R, H, C, P, N),$$

the proof-term row p remains a displayed carrier coordinate. It may be used as the source of the typing Ext *row T, the erasure* Cont *route E, and the subject-reduction compatibility row R, but it is not exported after erasure as proof irrelevance, host conversion, semantic erasure correctness, or hidden erased evidence outside Z.*

Proof. Project the carrier of Definition 642.1. The row p is one of the displayed coordinates, and the rows T, E, R, H, C, P, N record exactly how it is typed, erased, transported, replayed, and named.

Apply Definition 642.2. Carrier admission keeps p in the finite tuple, while erased-term non-escape excludes proof irrelevance, host conversion, semantic erasure correctness, and hidden proof evidence from the exported surface.

Now use Definition 642.3 and Definition 642.5. These results show that the proof row is consumed only through the typing, erasure, and subject-reduction rows already displayed in the packet.

Finally combine Definition 642.4 with Definition 642.6. Public membership and downstream ledger reads have no additional proof-evidence coordinate beyond the displayed row p and its local routes. □

Theorem 642.8 (Proof-term erasure trace erasure membership lock)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet*

$$Z = (p, e, T, E, R, H, C, P, N),$$

every public membership read of the erased row e factors through the displayed Ext *row T, the erasure* Cont *route E, the subject-reduction row R, and the local rows H, C, P, N. The read cannot be certified by host conversion, quotient term equality, proof irrelevance, semantic erasure correctness, or an erased proof witness outside the packet.*

Proof. Begin with the carrier definition. The erased row e is displayed together with the typing row T, erasure route E, subject-reduction row R, and local rows H, C, P, N; no separate host membership coordinate is present.

Apply Definition 642.2. Its Ext membership-preservation and subject-reduction compatibility rows bind the erased public read to T, E, and R.

Next apply Definition 642.4 and Definition 642.5. Membership of e is read through the displayed typing row and replayed along the displayed erasure route, with H and C supplying only componentwise transport and continuation replay.

Finally use Definition 642.3 and Definition 642.6. The public ledger is exhausted by $p, e, T, E, R, H, C, P, N$, so no host conversion, quotient equality, proof irrelevance, semantic erasure theorem, or erased proof witness can certify membership of e outside the packet. □

Theorem 642.9 (Proof-term erasure trace kernel scope lock)**.** *For any accepted* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *packet*

$$Z = (p, e, T, E, R, H, C, P, N),$$

the finite route

$$\mathsf{BHist} \longmapsto \mathsf{Ext} \longmapsto \mathsf{Cont} \longmapsto \mathsf{Pkg} \longmapsto \mathsf{NameCert}$$

is the whole kernel-facing scope of the packet. The route binds the proof-term row p, erased row e, typing row T, erasure route E, subject-reduction row R, componentwise transport row H, continuation ledger C, provenance row P, and local naming row N; it exports no host conversion relation, quotient term equality, proof-irrelevance principle, or semantic erasure oracle.

Proof. Project the carrier of Definition 642.1. Each public coordinate is one of $p, e, T, E, R, H, C, P, N$, and each coordinate is typed by the displayed BHist, Ext, Cont, Pkg, or $\mathsf{NameCert}$ layer.

Apply the NameCert obligations and the erasure membership lock. They bind erased membership to T, E, R, H, C, P, N and keep the proof row p as the only proof evidence used by the route.

The public-row ledger exhaustion excludes every row not present in the carrier. Hence the kernel-facing scope is exactly the displayed finite route. □

Theorem 642.10 (Proof-term erasure trace erased-row determinacy)**.** *Let $Z = (p, e, T, E, R, H, C, P, N)$ and $Z' = (p', e', T', E', R', H', C', P', N')$ be accepted $\mathsf{ProofTermErasureTrace}^{\uparrow}$ packets. If their proof-term rows, typing Ext rows, erasure Cont routes, subject-reduction rows, componentwise transport rows, continuation ledgers, provenance rows, and local naming rows agree through the displayed route data, then the erased public rows agree: e and e' are determined by those rows. No consumer may choose a different erased row by appealing to host conversion, quotient equality, proof irrelevance, or an external erasure theorem.*

Proof. Start from the erasure-route replay theorem. It says that the route from the proof row to the erased public row is exactly the displayed Cont row E together with the local replay data.

Use the kernel scope lock on both packets. Since the agreed rows include the typing, erasure, subject-reduction, transport, continuation, provenance, and naming data, both packets expose the same public route from proof evidence to erased row.

Finally apply the erasure membership lock. Membership of the erased row is certified only through that route, so the carrier has no remaining coordinate from which a second erased public row could be selected. □

Theorem 642.11 (Proof-term erasure trace public-consumer factorization)**.** *Every public consumer of an accepted $\mathsf{ProofTermErasureTrace}^{\uparrow}$ packet factors through the displayed derivation and preservation siblings: the derivation route of chapter 598 supplies the typing row T, the preservation route of chapter 483 supplies the subject-reduction row R, and the local packet exports the erased row e only through E, H, C, P, N. The factorization exports no erased proof witness, host conversion row, quotient term equality, proof irrelevance, or semantic erasure-correctness oracle.*

Proof. First read the derivation side. The type-checking derivation trace supplies the Ext membership row T for the proof-term row p, so the erased public read is not introduced as an independent host judgment.

Next read the preservation side. The subject-reduction route supplies R, and the local erasure membership lock forces the erased row e to remain compatible with T, E, and R.

Finally use erased-row determinacy and public-row ledger exhaustion. Any downstream consumer sees the same erased row fixed by p, T, E, R, H, C, P, N, and the packet has no additional public coordinate for proof evidence or host conversion. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTermErasureTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTermErasureTrace}^{\uparrow}$ packet is generated from proof-term and erased-term rows, a typing Ext row, an erasure Cont route, a subject-reduction compatibility row, componentwise hsame transports, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.ProofTermErasureTraceUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofTermErasureTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 642.1, the TasteGate single-carrier marker, Definition 642.2, Definition 642.8, Definition 642.9, Definition 642.10, and Definition 642.11 bind the mature paper surface: carrier, NameCert obligations, erasure membership, kernel scope, public consumer factorization, and checked TasteGate single-carrier alignment.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofTermErasureTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ProofTermErasureTraceUp.ProofTermErasureTraceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host conversion, quotient term equality, normalization oracles, proof irrelevance, or semantic erasure correctness outside the displayed finite packet.

Upgrade path: Mature closure requires examples or completeness treatment for the same displayed erasure-trace route.

Chapter 643

A Concrete Naming Certificate for $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$

The $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ surface turns the other-minds locality-cell commitment of Definition 9415.9 into a finite coordination packet. It aligns public observer commitments, locality-cell evidence, and inter-history coherence rows without exposing private Self coordinates or treating observer agreement as host-level identity.

The row M is consumed from chapter 474 as the finite inter-Hist configuration surface. CrossPerspectiveAlignmen adds only locality-cell and public commitment routing on top of that sibling surface, not private observer-state access.

Definition 643.1 (CrossPerspectiveAlignment carrier)**.** *A* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, B, L, O, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-observer row issuing the alignment request;*
2. *B is the target-observer public row, read only as an other-minds commitment surface;*
3. *L is the locality-cell evidence row, recording the public* $\mathsf{LocalityCell}^{\uparrow}$ *evidence that makes the target row available for coordination;*
4. *O is the public* $\mathsf{OtherMindsCommitment}^{\uparrow}$ *row tying A's commitment to B without direct target* Self *access;*
5. *M is the accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite inter-Hist configuration in which the two observer histories are compared;*
6. *H records the componentwise* hsame *transports for the observer rows, locality-cell evidence, other-minds commitment, multi-Hist configuration, package row, and local naming row;*
7. *C records the* Cont *routes that let public consumers read the commitment row, locality-cell row, and inter-Hist coherence rows without entering private observer state;*
8. *P is the* Pkg *provenance over* $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CrossPerspectiveAlignment}^{\uparrow}}$ *row.*

The classifier compares carriers only through these displayed rows and their hsame *and* Cont *transports.*

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentTasteGate single carrier alignment`

Theorem 643.2 (CrossPerspectiveAlignment NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CrossPerspectiveAlignment}^{\uparrow}}$ *surface has five finite obligations: carrier exposure by Definition 643.1; public other-minds commitment through O; locality-cell evidence soundness through L; inter-Hist coherence through the* $\mathrm{MultiHistConfig}^{\uparrow}$ *row M and the displayed* hsame *and* Cont *rows; and non-escape from private* Self *access, hidden observer histories, host observer equality, and unrecorded cross-perspective synchronization.*

Proof. Carrier exposure is projection from the finite packet $B = (A, B, L, O, M, H, C, P, N)$.

The public commitment obligation is read from O. It uses $\mathsf{OtherMindsCommitment}^{\uparrow}$ as a public certificate surface, so the target observer row is available only through visible commitment data.

The locality and inter-Hist obligations are read next. The locality-cell evidence is the row L, while finite inter-Hist coherence is the row M transported by H and consumed through C.

Finally inspect the coordinate list. There is no field for private target Self access, hidden observer history, host observer equality, or unrecorded synchronization, so none of those data can be exported by the packet. □

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentCarrier namecert obligations`

Theorem 643.3 (CrossPerspectiveAlignment locality-cell soundness)**.** *Every accepted* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *packet aligns perspectives only after the public locality-cell evidence row L, the* $\mathsf{OtherMindsCommitment}^{\uparrow}$ *row O, and the* $\mathrm{MultiHistConfig}^{\uparrow}$ *row M are displayed in the same finite packet. A consumer may read $A, B, L, O, M, H, C, P, N$, but it cannot replace L by private target-state access, a global observer frame, or an unlisted synchronization coordinate.*

Proof. Project the accepted packet as $B = (A, B, L, O, M, H, C, P, N)$. The locality-cell evidence available to the consumer is exactly the displayed coordinate L.

The target public row is tied to the other-minds surface by O, and the inter-Hist comparison is tied to the finite configuration by M. The displayed transports H and routes C preserve those attachments as public certificate data.

No coordinate names private target-state access, a global observer frame, or an unlisted synchronization row. Thus locality-cell soundness is a finite public-evidence condition, not an ambient identification of observer states. □

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentCarrier locality cell soundness`

Theorem 643.4 (CrossPerspectiveAlignment non-escape boundary)**.** *An accepted* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *carrier exports only the source-observer row, target-observer public row, locality-cell evidence row, public other-minds commitment row, finite multi-Hist configuration row, displayed* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no private* Self *access for the target observer, no hidden observer history, no host observer equality, no global frame, and no unrecorded cross-perspective synchronization.*

Proof. Inspect the carrier coordinates of Definition 643.1. Every exported row is one of A, B, L, O, M, H, C, P, or N.

The rows L, O, and M are public certificate rows: locality-cell evidence, other-minds commitment, and finite multi-Hist configuration. The rows H and C only transport and route those displayed public rows.

There is no remaining coordinate for target private Self access, hidden observer history, host observer equality, a global frame, or unrecorded synchronization. Hence the alignment packet cannot export any of those objects. □

Theorem 643.5 (CrossPerspectiveAlignment observer-identity handoff)**.** *Every accepted* CrossPerspectiveAlignmen *alignment packet reads the target public identity only through the displayed observer-history identity and other-minds commitment rows. The target row B may be consumed as a public* ObserverHistIdentity$^{\uparrow}$/OtherMind *handoff carried by O, H, C, P, N, but no read may pass through a private target* Self *coordinate or identify the two observers by host equality.*

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentCarrier observer identity handoff`

Proof. Project the carrier of Definition 643.1. The target row is the public coordinate B, and its commitment route is the row O together with the displayed transports and continuations H and C.

Use the NameCert obligations of Definition 643.2. They require public other-minds commitment through O and make carrier exposure a finite packet read rather than a private observer-state read.

Finally apply the non-escape boundary of Definition 643.4. Since the packet has no coordinate for private target Self access or host observer equality, every accepted target-identity handoff is exhausted by the public observer-history and other-minds rows already displayed. □

Theorem 643.6 (CrossPerspectiveAlignment finite multi-Hist coherence)**.** *Every inter-history comparison accepted by* CrossPerspectiveAlignment$^{\uparrow}$ *factors through the finite* MultiHistConfig$^{\uparrow}$ *row M, the componentwise* hsame *transports recorded in H, and the* Cont *routes recorded in C. No comparison may appeal to an ambient global history, an unlisted synchronization row, or a hidden equality between observer histories.*

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentCarrier finite multihist coherence`

Proof. Read the carrier coordinates. The only inter-history configuration coordinate is M, and the only transport and consumer routes attached to it are the displayed rows H and C.

The locality-cell soundness theorem Definition 643.3 requires L, O, M, H, C, P, N to appear in the same finite packet before the alignment is consumed. Thus comparison across histories is a replay of the finite multi-Hist row through the stored transport and continuation data.

The non-escape theorem Definition 643.4 excludes hidden observer histories, host observer equality, global frames, and unrecorded synchronization. Hence finite multi-Hist coherence is exactly the route through M, H, C, not an ambient comparison outside the packet. □

Theorem 643.7 (CrossPerspectiveAlignment public consumer package)**.** *The carrier exposure of Definition 643.1, locality-cell soundness, observer-identity handoff, finite multi-Hist coherence, non-escape, and the displayed TasteGate marker assemble the public obligation package for* CrossPerspectiveAlignmen

A public consumer receives only the finite rows $A, B, L, O, M, H, C, P, N$ with their NameCert *obligations, and receives no private target* Self*, hidden observer history, host observer equality, global frame, or unrecorded synchronization.*

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.CrossPerspectiveAlignmentCarrier public consumer package`

Proof. Begin with the carrier definition. It displays exactly the source observer, target public observer, locality-cell evidence, other-minds commitment, multi-Hist configuration, componentwise transport, continuation, package provenance, and local naming rows.

Apply the existing NameCert obligation and locality-cell soundness theorems. They bind carrier exposure, public other-minds commitment through O, locality-cell evidence through L, and finite inter-Hist coherence through M, H, C.

The observer-identity handoff and finite multi-Hist coherence theorems above isolate the two consumer routes that would otherwise be ambiguous. The first keeps target public identity behind the observer-history and other-minds rows; the second keeps inter-history comparison behind the finite M, H, C route.

Finally use the non-escape theorem and the checked TasteGate alignment marker. Non-escape removes private and ambient coordinates from the package, while the marker records that the displayed carrier is the single TasteGate-aligned carrier surface. Thus the public obligation package is finite and contains only the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CrossPerspectiveAlignment}^{\uparrow}}$.

Constructive story: A $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ packet is generated from a source observer row, a target public observer row, locality-cell evidence, a public $\mathsf{OtherMindsCommitment}^{\uparrow}$ row, a finite $\mathrm{MultiHistConfig}^{\uparrow}$ row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.CrossPerspectiveAlignmentUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CrossPerspectiveAlignment}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 643.1, Definition 643.2, Definition 643.3, Definition 643.4, Definition 643.5, Definition 643.6, and Definition 643.7 bind the finite public alignment surface over ObserverHistIdentity, OtherMindsCommitment, LocalityCell, MultiHistConfig, hsame, Cont, Pkg, NameCert, and the checked TasteGate carrier marker.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CrossPerspectiveAlignment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, bridge status, private access to another observer's Self, hidden observer histories, host observer equality, a global frame, or unrecorded synchronization.

Upgrade path: No paper-side closure upgrade remains for the displayed alignment rows, while formal verification remains on its independent axis.

Chapter 644

A Concrete Naming Certificate for $\mathsf{RegularCauchyMesh}^{\uparrow}$

The $\mathsf{RegularCauchyMesh}^{\uparrow}$ packet records the shared finite mesh surface used by regular-Cauchy Real-completeness arguments before a diagonal mesh-budget consumer reads a $\mathsf{Real}^{\uparrow}$ seal. It names a finite family of regular rational observations indexed by mesh rows, together with StreamName windows, dyadic tolerance rows, transport, continuation, provenance, and a local naming certificate. The packet is a mesh-indexed observation carrier, not a completed real, a quotient class of regular sequences, or an ambient completeness principle.

The concrete dependency route is through the finite modulus-window surface of chapter 593 and the mesh-cell compatibility surface of chapter 358. The $\mathsf{RegularCauchyMesh}^{\uparrow}$ packet refines the schedule packet by replacing a single selected window with a finite mesh-index row while preserving the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ handoff, hsame, Cont, Pkg, and NameCert rows.

Definition 644.1 (Regular Cauchy mesh carrier)**.** *A* $\mathsf{RegularCauchyMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (a, j, w, r, d, m, e, h, c, p, n)$$

with the following displayed coordinates:

1. *a is the finite regular-Cauchy request row, recording the requested precision and the finite observation sites to be inspected;*
2. *j is the mesh-index row, listing the finite* $\mathsf{DyadicMesh}^{\uparrow}$ *cells and regular-Cauchy positions selected by the request;*
3. *w is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to those mesh cells;*
4. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected regular rational observations;*
5. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint comparisons, and mesh-cell tolerance bounds;*
6. *m is the* $\mathsf{DyadicMesh}^{\uparrow}$ *compatibility row that states the selected mesh cells are exactly the cells inspected by j and w;*
7. *e is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after a, j, w, r, d, m have been displayed;*
8. *h records componentwise* hsame *transports for the request, mesh indices, windows, readbacks, dyadic ledger, mesh compatibility row, handoff row, provenance, and local name;*

9. *c records the displayed* Cont *routes from a to j, from j to w, from w to r, from r to d, from d to m, and from m to e;*
10. *p is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *n is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMesh}^{\uparrow}}$ *row.*

The carrier contains no infinite mesh, no quotient equality of regular sequences, no host real equality, no chosen limit representative, and no ambient Real completeness theorem.

Definition 644.2 (Regular Cauchy mesh classifier)**.** *Two* $\mathsf{RegularCauchyMesh}^{\uparrow}$ *carriers*

$$G = (a, j, w, r, d, m, e, h, c, p, n) \quad \text{and} \quad G' = (a', j', w', r', d', m', e', h', c', p', n')$$

are classified together exactly when the displayed request rows, mesh-index rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore ledgers, DyadicMesh compatibility rows, RealUp handoff rows, componentwise transports, continuation routes, package provenance, and local NameCert rows are matched by the corresponding hsame, Cont, Pkg, *and* NameCert *evidence already carried in the two packets.*

The classifier is rowwise and finite. It does not identify completed reals, quotient regular-Cauchy streams, or mesh families outside the displayed cells.

Theorem 644.3 (Regular Cauchy mesh NameCert obligation sketch)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMesh}^{\uparrow}}$ *surface for Definition 644.1 has the following five rows:*

1. *carrier admission for $G = (a, j, w, r, d, m, e, h, c, p, n)$;*
2. *finite mesh request exposure through a and mesh-index exposure through j;*
3. *selected* $\mathsf{StreamName}^{\uparrow}$ *windows w,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks r,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger d, and* $\mathsf{DyadicMesh}^{\uparrow}$ *compatibility row m;*
4. $\mathsf{Real}^{\uparrow}$ *handoff through e, componentwise* hsame *transport through h, and displayed* Cont *route replay through c;*
5. Pkg *provenance through p, local naming through n, and non-escape for infinite meshes, quotient regular-Cauchy equality, host real equality, selected limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMeshUp.RegularCauchyMeshTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 644.1. The packet exposes the eleven displayed rows and no other mesh-indexed data.

The request and mesh-index rows a and j choose only finitely many cells. The selected windows w, readbacks r, dyadic ledger d, and mesh compatibility row m then give the finite regular-rational observations attached to those cells.

The handoff row e is reached through the displayed transports and continuations in h and c, with provenance and local naming recorded by p and n. Thus the five rows exhaust the NameCert surface.

The non-escape clauses follow from the carrier inventory. Since no row supplies an infinite mesh, quotient regular-Cauchy equality, host real equality, a selected limiting representative, or an ambient completeness theorem, none of those objects can be exported by the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMesh}^{\uparrow}$ packet is generated from a finite regular-Cauchy request, finite mesh indices, StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a DyadicMesh compatibility row, a RealUp handoff row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 644.1, Definition 644.2, and Definition 644.3 bind the seed carrier, rowwise classifier, and five-row NameCert surface for finite mesh-indexed regular-rational observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient equality of regular sequences, host real equality, selected limits, infinite mesh selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rowwise classifier stability, finite mesh coverage, RealUp handoff, provenance, and non-escape for the same RegularCauchyMeshUp packet.

Definition 644.4 (Regular Cauchy mesh displayed submesh restriction)**.** *Let*

$$G = (a, j, w, r, d, m, e, h, c, p, n)$$

be a $\mathsf{RegularCauchyMesh}^{\uparrow}$ *carrier as in Definition 644.1. A displayed finite submesh selector for* G *is a finite row* q *equipped with an inclusion read* $\iota : q \to j$ *whose cells are entries already displayed in the mesh-index row* j*.*

The restriction of G *along* q *is the packet*

$$G|_q = (a|_q, j|_q, w|_q, r|_q, d|_q, m|_q, e|_q, h|_q, c|_q, p|_q, n|_q),$$

where $j|_q$ *is the row* q*, and every other coordinate is obtained by retaining exactly the entries of the corresponding row of* G *whose mesh index lies in the image of* ι*. The handoff row* $e|_q$*, transport row* $h|_q$*, continuation row* $c|_q$*, provenance row* $p|_q$*, and local naming row* $n|_q$ *retain only the components whose source rows are retained by* $a|_q, j|_q, w|_q, r|_q, d|_q, m|_q$*.*

Theorem 644.5 (Regular Cauchy mesh finite submesh restriction)**.** *Let*

$$G = (a, j, w, r, d, m, e, h, c, p, n)$$

be an accepted $\mathsf{RegularCauchyMesh}^{\uparrow}$ *carrier, and let* q *be a displayed finite submesh selector for* G *in the sense of Definition 644.4. Suppose the retained StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, DyadicMesh compatibility row,* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *rows are exactly the rows pulled back along the inclusion read* $\iota : q \to j$*. Then*

$$G|_q = (a|_q, j|_q, w|_q, r|_q, d|_q, m|_q, e|_q, h|_q, c|_q, p|_q, n|_q)$$

is an accepted $\mathsf{RegularCauchyMesh}^{\uparrow}$ *carrier.*

Moreover, every retained consumer read of a mesh cell in q *factors through the restricted rows*

$$a|_q, j|_q, w|_q, r|_q, d|_q, m|_q, e|_q, h|_q, c|_q, p|_q, n|_q.$$

It cannot recover a cell of j *outside* q*, an infinite mesh, quotient regular-Cauchy equality, host real equality, a chosen limiting representative, or an ambient Real-completeness theorem.*

The rate/modulus surface is unchanged: every retained Cauchy read uses the same displayed StreamName window row and DyadicRatCore tolerance ledger already present in G*, restricted to* q*, and no new tail bound or completion modulus is introduced.*

Proof. Project the accepted carrier through Definition 644.1. The visible data are exactly the finite request row a, mesh-index row j, window row w, readback row r, dyadic ledger d, mesh compatibility row m, handoff row e, transport row h, continuation row c, provenance row p, and local name row n.

The selector q is finite and every entry of q is read from the displayed mesh-index row j. Therefore $j|_q = q$ is a finite mesh-index row, and $a|_q$ is the corresponding finite request row obtained by keeping only the requests whose inspected mesh cells lie in q.

For each retained mesh cell, the StreamName window, RegSeqRat readback, DyadicRatCore tolerance entry, and DyadicMesh compatibility entry are already rows of the original accepted packet. Pulling those rows back along ι gives $w|_q, r|_q, d|_q, m|_q$ without adding any analytic object. The mesh compatibility row remains a compatibility row because it now states the same cell-window agreement after both sides have been restricted to the same displayed submesh.

The rows e, h, c, p, n are administrative over the displayed finite route. Retaining only the components whose sources survive in $a|_q, j|_q, w|_q, r|_q, d|_q, m|_q$ yields the handoff, transport, continuation, provenance, and naming rows $e|_q, h|_q, c|_q, p|_q, n|_q$. These rows have exactly the roles required by Definition 644.1, so the restricted tuple is a $\mathsf{RegularCauchyMesh}^{\uparrow}$ carrier.

Acceptance follows from the local certificate surface of Definition 644.3. Carrier admission is the restricted finite tuple, finite mesh request exposure is through $a|_q$ and $j|_q$, selected windows and readbacks are $w|_q$ and $r|_q$, the dyadic and mesh rows are $d|_q$ and $m|_q$, and the handoff, transport, route, provenance, and local naming rows are $e|_q, h|_q, c|_q, p|_q, n|_q$.

Now let a retained consumer read a cell of q. In the original packet such a read factors through the displayed mesh-index row j, the attached window row w, the readback row r, the dyadic ledger d, the compatibility row m, and then the handoff and certificate rows. Since the cell lies in the image of ι, every one of those entries is present in the restricted tuple. Hence the same read factors through the restricted rows.

No omitted cell can be recovered from this factorization, because every retained row was formed by pullback along $q \to j$ and contains no component indexed outside q. The non-escape clauses are inherited from the carrier inventory in Definition 644.1: neither G nor $G|_q$ has a coordinate for an infinite mesh, quotient regular-Cauchy equality, host real equality, a chosen limiting representative, or an ambient Real-completeness theorem. □

Chapter 645

A Concrete Naming Certificate for $\mathsf{BoundedCauchyCover}^{\uparrow}$

The $\mathsf{BoundedCauchyCover}^{\uparrow}$ interface names the finite observation surface that carries a Cauchy modulus from a precision budget to the windows read by the real-completion diagonal route. It sits between the real observation budget and the $\mathsf{Real}^{\uparrow}$ seal: the packet remembers the requested budget, the modulus row, the finite stream-window cover, the dyadic tolerance readback, the regular rational consumption row, transport, continuations, provenance, and local naming. It does not export quotient streams, selected limits, host real equality, or an ambient completeness datum.

The finite cover route consumes the finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational consumption row of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The row B remains grounded in chapter 611, and the displayed budget-to-window route passes the sibling rows to the Real and Completion diagonal consumer rather than supplying an independent completeness principle.

Definition 645.1 (Bounded Cauchy cover carrier)**.** *A* $\mathsf{BoundedCauchyCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, M, W, D, R, H, C, P, N)$$

with the following displayed coordinates:

1. *B is the unary precision budget requested by the downstream real-completion consumer;*
2. *M is the Cauchy modulus row that turns B into a finite threshold for the relevant tail comparisons;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window cover selected by the modulus threshold;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback attached to the covered windows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity consumption row that reads the covered windows through the regular rational source;*
6. *H records componentwise* hsame *transports for the budget, modulus row, window cover, dyadic readback, regularity row, provenance, and local name;*
7. *C records displayed* Cont *routes from B to M, from M to W, from W to D, and from D to R;*
8. *P records* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedCauchyCover}^{\uparrow}}$ *row.*
The classifier compares two such packets only after transporting their budgets, Cauchy modulus rows, window covers, dyadic readbacks, and $\mathsf{RegSeqRat}^{\uparrow}$ *rows to a shared finite cover. The carrier has no coordinate for a quotient stream, selected limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.BoundedCauchyCoverUp.BoundedCauchyCoverTasteGate single carrier alignment`

Theorem 645.2 (Bounded Cauchy cover NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyCover}^{\uparrow}}$ *surface for Definition 645.1 consists exactly of carrier admission, unary budget exposure, Cauchy modulus exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *cover selection,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity consumption, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet K. The rows B, M, W, D, R expose the precision budget, Cauchy modulus threshold, finite observation cover, dyadic readback, and regular rational consumption data.

The cover obligations are finite. The route first reads B, then the modulus row M, then the selected windows W, and only then the dyadic tolerance and regularity rows D and R.

The rows H, C, P, N supply transport, route replay, provenance, and local naming. Since no quotient stream, selected limit, host real equality, or ambient completeness coordinate exists in the carrier, the listed rows exhaust the local certificate. □

Theorem 645.3 (Bounded Cauchy cover finite-window coverage)**.** *For an accepted* $\mathsf{BoundedCauchyCover}^{\uparrow}$ *carrier, every exported dyadic tolerance readback in D is covered by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W, and every window in W is selected from the budget B through the Cauchy modulus row M and the displayed route C.*

Proof. Project the accepted packet to the rows B, M, W, D, H, C. The first displayed route sends the unary budget to the Cauchy modulus row, and the second sends the resulting threshold to the finite window cover.

The route from W to D is the only displayed path by which a dyadic tolerance readback enters the packet. Componentwise transport in H preserves the budget, modulus, cover, and readback rows when the packet is compared with another accepted carrier.

Thus every dyadic readback exported by the packet is attached to a selected finite window. No readback can be imported from an unbounded stream schedule or a hidden completion object. □

Theorem 645.4 (Bounded Cauchy cover budget transport)**.** *Classifier transport for* $\mathsf{BoundedCauchyCover}^{\uparrow}$ *preserves the same budget-to-cover route. If two accepted packets classify together, then after applying the displayed transports in H their B, M, W, D, and R rows lie over a shared finite cover, and the route*

$$B \longmapsto M \longmapsto W \longmapsto D \longmapsto R$$

is replayed by the displayed continuations in C.

Proof. Begin with the classifier clause in the carrier definition. It compares packets only after the budget, modulus row, windows, dyadic readback, and regularity row have been transported to a common finite cover.

The continuation coordinate C then replays the same route on the transported rows. The budget row selects the modulus row, the modulus row selects the finite cover, the cover supplies the dyadic readback, and the readback feeds the regularity consumption row.

The provenance and local naming rows P, N are transported with the same packet. Since the carrier has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness, transport cannot introduce any additional comparison data. □

Theorem 645.5 (Bounded Cauchy cover Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *diagonal-seal consumer of an accepted* $\mathsf{BoundedCauchyCover}^{\uparrow}$ *packet receives exactly the bounded cover rows*

$$B, M, W, D, R, H, C, P, N.$$

The consumer reads a unary precision budget, a Cauchy modulus row, a finite $\mathsf{StreamName}^{\uparrow}$ *observation-window cover, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback, a* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity consumption row, transport, continuation, provenance, and local naming. It receives no quotient stream, selected limit, host real equality, or ambient completeness datum.*

Proof. Start from finite-window coverage. It places every dyadic readback under the finite windows selected by the budget and modulus rows.

Use budget transport to move the same route to the consumer-facing packet. The transported route is still $B \longmapsto M \longmapsto W \longmapsto D \longmapsto R$, with H, C, P, N carrying the displayed bookkeeping.

The handoff to $\mathsf{Real}^{\uparrow}$ can therefore inspect only those rows. All excluded analytic objects are absent from the carrier, so the diagonal-seal consumer receives the finite cover surface and nothing outside it. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchyCover}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchyCover}^{\uparrow}$ packet is generated from a unary precision budget, a Cauchy modulus row, finite StreamName observation-window cover, DyadicRatCore tolerance readback, RegSeqRat regularity consumption row, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedCauchyCover}^{\uparrow}}$, seed)

Scope closed: Definition 645.1, Definition 645.2, Definition 645.3, Definition 645.4, and Definition 645.5 bind the finite budget-to-modulus-to-window-to-readback-to-RegSeqRat route used by RealUp diagonal-seal consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedCauchyCover}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, budget transport, and RealUp handoff for the same bounded Cauchy cover packet.

The $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ hub routes the current Phase Real and Completion diagonal-limit packet through the L10 sibling spine of chapter 291, chapter 126, chapter 301, and chapter 38; the child files below keep the route finite and refuse quotient streams, selected limits, countable choice, host real equality, and ambient completeness.

Chapter 646

A Concrete Naming Certificate for $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$

The $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ surface records the finite compatibility packet between the Real completeness diagonal-limit route and the downstream completion-seal consumer. It ties the accepted diagonal-limit packet, the three-leg regularity estimate, the Cauchy limit seal, stream windows, regular rational readback, dyadic tolerance bookkeeping, and the final $\mathsf{Real}^{\uparrow}$ seal into one local naming certificate. The packet is a compatibility surface, not a host completion theorem and not a choice of a completed real.

Definition 646.1 (Diagonal-limit compatibility carrier)**.** *A* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit packet whose diagonal route is being consumed;*
2. *T is the accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *three-leg estimate packet controlling the selected diagonal windows;*
3. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal route through which downstream completion-facing consumers read the diagonal handoff;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the diagonal-limit row, the triangle estimate, and the seal route;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the finite observations used by the diagonal selector and by the seal consumer;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed after the window and dyadic ledger have been fixed;*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal class reached by the compatibility route;*
8. *H records componentwise* hsame *transports for R, T, S, D, W, Q, L and the structural rows;*
9. *C records the displayed* Cont *routes from the diagonal-limit packet through the triangle estimate, seal route, tolerance ledger, window family, regular rational readback, and* $\mathsf{Real}^{\uparrow}$ *seal;*
10. *P is the* Pkg *provenance over* $\mathsf{RealCompleteness}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row.*

The carrier classifier reads only these displayed rows and their hsame *and* Cont *transports.*

Lean4 (definition) `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityUp`

Theorem 646.2 (Diagonal-limit compatibility NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *surface has five finite obligations:*

1. *carrier admission for the packet $K = (R, T, S, D, W, Q, L, H, C, P, N)$ of Definition 646.1;*
2. *triangle-bound and selector alignment, requiring the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *row T to consume the same selected windows and dyadic tolerance ledger named by R, W, D;*
3. $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route compatibility, requiring the seal row S to read the regular-rational handoff Q and the* $\mathsf{Real}^{\uparrow}$ *seal class L only after the displayed diagonal route has passed through T, D, W;*
4. *classifier transport over the shared* hsame *rows in H and the displayed continuation routes in C;*
5. *non-escape for quotient streams, selected host limits, host real equality, choice data, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityPacketNameCertObligations`

Proof. Project the carrier of Definition 646.1. The rows R, T, S, D, W, Q, L are the only mathematical coordinates, and H, C, P, N are the only transport, route, provenance, and naming coordinates.

The triangle-bound and selector-alignment obligation is read from T together with R, W, D. The same window family and dyadic ledger are displayed before the estimate is consumed, so the three-leg bound cannot choose a second selector surface.

The seal compatibility obligation is read next. The route S reaches Q and L only through the displayed continuation rows in C, after the diagonal-limit packet, triangle estimate, tolerance ledger, and windows have been fixed.

Finally inspect the coordinate list. Since no quotient stream, selected host limit, host real equality row, choice datum, or ambient completeness principle is present, no classifier transport or seal consumer can export one. □

Theorem 646.3 (Diagonal-limit compatibility terminal Real route)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every terminal $\mathsf{Real}^{\uparrow}$ *seal read factors through the displayed route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Here R is read from $\mathsf{RealCompleteness}^{\uparrow}$*, T from* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$*, S from* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, D from* $\mathsf{DyadicRatCore}^{\uparrow}$*, W from* $\mathsf{StreamName}^{\uparrow}$*, Q from* $\mathsf{RegSeqRat}^{\uparrow}$*, and L from* $\mathsf{Real}^{\uparrow}$*. The terminal read has no quotient stream, host limit, choice datum, or ambient completeness principle available outside this route.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility terminal real route`

Proof. Project the accepted carrier of Definition 646.1. Its displayed rows name the diagonal-limit packet, triangle estimate, seal route, tolerance ledger, window family, rational readback, terminal Real seal, transport rows, continuation rows, provenance, and local naming row.

Apply the NameCert obligation theorem Definition 646.2. The triangle-bound clause routes T through R, W, D, and the seal-route clause reads Q and L only after R, T, D, W have been fixed.

Finally use the non-escape clause of the same obligation theorem. Since no quotient stream, host limit, choice datum, or ambient completeness principle is a coordinate of K, a terminal $\mathsf{Real}^{\uparrow}$ seal read cannot leave the displayed chain. □

Theorem 646.4 (Diagonal-limit terminal selector Real seal pullback)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

any terminal $\mathsf{Real}^{\uparrow}$ *seal read mediated by a* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *row factors through the same displayed window family* W*, dyadic ledger* D*, regular readback* Q*, seal row* S*, and terminal Real row* L *of* K*. Conversely, preserving* W, D, Q, S, L *together with the displayed* H, C, P, N *rows preserves the selector-mediated terminal seal route.*

Thus the selector route and the terminal $\mathsf{Real}^{\uparrow}$ *seal form one pullback packet over the displayed* BHist *carrier rows. The packet exports no quotient stream, selected host limit, host real equality, choice datum, hidden selector, or ambient completeness coordinate outside* K *and the finite selector boundary.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier terminal selector real seal pullback`

Proof. Project the accepted carrier of Definition 646.1. The rows W, D, Q, S, L are displayed coordinates of K, and the structural rows H, C, P, N are the only transport, route, provenance, and local naming rows available to the terminal read.

Apply Definition 646.3. It forces every terminal $\mathsf{Real}^{\uparrow}$ seal read of K through $R, T, S, D, W, Q, L, H, C, P, N$, so a selector-mediated read cannot skip the shared window, dyadic, readback, seal, or terminal Real rows.

Now use the finite selector boundary of Definition 782.10. A diagonal-limit or Real-completion consumer using a $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet must read the finite route through its displayed budget, schedule, window, dyadic, regular, and seal-facing rows before reaching the completion-facing endpoint.

The two routes meet on W, D, Q, S, L and their displayed H, C, P, N transports. Therefore preservation of these rows preserves the selector-mediated terminal seal route, while any accepted selector-mediated terminal seal read factors through the same pullback packet.

Finally use Definition 646.11 together with the non-escape clause of the finite selector boundary. Neither side exports quotient streams, selected host limits, host real equality, choice data, hidden selectors, or ambient completeness; hence no external coordinate can witness another terminal seal route. □

Theorem 646.5 (Diagonal-limit window ledger pullback)**.** *For any accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

preserving the displayed $\mathsf{StreamName}^{\uparrow}$ *window family* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* L *is exactly the finite pullback condition needed for selector-mediated terminal Real reads. A terminal selector read that preserves these four rows and the*

displayed H, C, P, N support is the same finite Real-seal route; a read that changes one of them is not the accepted selector-mediated terminal route.

Proof. Project the carrier of Definition 646.1. The rows W, D, Q, L are displayed as the window family, dyadic ledger, regular readback, and terminal Real seal of the packet, while H, C, P, N provide the only transport, route, provenance, and local naming support.

Apply Definition 646.4. It states that any terminal $\mathsf{Real}^{\uparrow}$ seal read mediated by a FiniteObservationBudgetSelecto row factors through the same displayed W, D, Q, S, L rows, and that preserving those rows together with H, C, P, N preserves the selector-mediated terminal seal route.

Next use Definition 646.3. It forces every terminal $\mathsf{Real}^{\uparrow}$ seal read of K through the finite route $R, T, S, D, W, Q, L, H, C, P, N$. The finite observation selector boundary Definition 782.10 supplies the matching selector side, where a diagonal-limit or Real-completion consumer reads only the finite budget, schedule, window, dyadic, regular, seal-facing, transport, route, provenance, and naming rows.

The forward direction follows by intersection of these two finite routes: their shared Real-facing data are exactly W, D, Q, L with support H, C, P, N. The reverse direction follows because preserving those rows preserves the selector-mediated terminal seal route already fixed by the terminal selector pullback theorem.

Finally inspect the excluded coordinates in both packets. Neither the diagonal carrier nor the finite selector boundary contains quotient streams, selected host limits, host real equality, choice data, hidden selectors, or ambient completeness. Therefore no external route can replace the displayed window-ledger pullback. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility window ledger pullback`

Theorem 646.6 (Diagonal-limit triangle selector alignment)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the triangle-bound row T consumes exactly the selected window family W and dyadic tolerance ledger D named by the $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet R. It has no second selector, no alternate dyadic ledger, and no hidden readback route outside the displayed compatibility carrier.*

Proof. Project the carrier of Definition 646.1. The rows R, T, D, W are displayed before the seal route or readback rows are inspected.

Use the NameCert obligations of Definition 646.2. The triangle-bound and selector-alignment clause forces T to read the same selected windows and the same dyadic ledger named by R, W, D.

The non-escape theorem Definition 646.11 excludes any additional selector or ledger coordinate. Hence the triangle selector surface is exactly W, D over the diagonal packet R. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility triangle selector alignment`

Theorem 646.7 (Diagonal-limit seal route compatibility)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route S reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q and the* $\mathsf{Real}^{\uparrow}$ *seal class L only through the displayed rows R, T, D, W. The seal route cannot bypass the diagonal packet, triangle estimate, dyadic ledger, or selected window family.*

Proof. Start from the compatibility carrier and project the route rows R, T, S, D, W, Q, L.

Apply the NameCert obligation theorem. Its seal-route compatibility clause places the read of Q and L after the diagonal-limit packet, triangle estimate, dyadic ledger, and windows have been fixed.

Now use Definition 646.11. Since the packet exports only the displayed coordinates, S has no continuation route to Q or L except through R, T, D, W and the continuation rows already recorded in C. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility seal route compatibility`

Theorem 646.8 (Diagonal-limit selector-observer seal correspondence)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$U = (B, W_U, Q_U, S_U, L_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet, and let*

$$O = (R_O, W_O, Q_O, D_O, S_O, L_O, H_O, C_O, P_O, N_O)$$

be an accepted $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *packet. The three packets name the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route exactly when their displayed L10 rows agree componentwise by* hsame*: the* $\mathsf{StreamName}^{\uparrow}$ *windows* W, W_U, W_O*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q_U, Q_O*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* S, S_U, S_O*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D, D_O*, and the terminal* $\mathsf{Real}^{\uparrow}$ *rows* L, L_U, L_O *are transported through the displayed* H, H_U, H_O*,* C, C_U, C_O*,* P, P_U, P_O*, and* N, N_U, N_O *rows. No quotient stream, host real equality, hidden selector, selected host limit, detached limit object, or ambient completeness principle can witness equality of the terminal route.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorObserverSealCorresp`

Proof. Project the three accepted carriers. The compatibility packet exposes W, Q, S, D, L through H, C, P, N; the selector-seal packet exposes W_U, Q_U, S_U, L_U through H_U, C_U, P_U, N_U; and the observer packet exposes W_O, Q_O, D_O, S_O, L_O through H_O, C_O, P_O, N_O.

First suppose the displayed L10 rows agree componentwise by hsame through those structural rows. Apply Definition 646.7 and Definition 646.12. They place the compatibility seal route behind the same stream-window, tolerance, readback, seal, and terminal Real rows.

Next apply Definition 1081.5 and Definition 961.3. The selector and observer packets expose the same downstream route positions, so the componentwise transport rows identify one terminal $\mathsf{Real}^{\uparrow}$ seal route.

Conversely, suppose the three accepted packets name the same terminal route. The cited seal-route and root-dependency theorems force any compatibility read of the terminal row through W, Q, S, D, L, while the selector handoff and observer exposure theorems force the sibling packets through their corresponding window, readback, tolerance, seal, and terminal rows.

Finally use the non-escape clauses of the three packets. Since none exports a quotient stream, host equality, hidden selector, selected host limit, detached limit object, or ambient completeness coordinate, the terminal-route equality has no witness except componentwise hsame through the displayed Cont, Pkg, and $\mathsf{NameCert}$ rows. □

Theorem 646.9 (DiagonalLimitCompatibility functor-uniform terminal factorization)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, let*

$$F = (S_F, T_F, M_F, W_{F,0}, W_{F,1}, R_F, D_F, K_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet, and let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. These three packets name the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route exactly when their displayed terminal rows agree componentwise: the* $\mathsf{StreamName}^{\uparrow}$ *windows* $W, W_{F,0}, W_{F,1}, W_U$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, R_F*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D, D_F, Q_U*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* S, S_F, T_F*, and the terminal* $\mathsf{Real}^{\uparrow}$ *rows* L, E_F, E_U *are identified through the displayed* hsame *rows* H, H_F, H_U*,* Cont *rows* C, C_F, C_U*,* Pkg *rows* P, P_F, P_U*, and local* NameCert *rows* N, N_F, N_U*.*

Equivalently, the diagonal compatibility route, the functor standard-reflection route, and the finite-family uniform-Cauchy consumer route factor through one terminal BEDC seal packet over those displayed rows. No quotient stream, host real equality, hidden selector, selected host limit, countable-choice schedule, detached limit object, or ambient completeness coordinate can witness another terminal route.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityFunctorUniformTerminalFacto`

Proof. First read the diagonal compatibility packet through Definition 646.8. Its terminal route is already fixed by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, together with the visible H, C, P, N structure.

Now attach the functor side. By Definition 1116.13, every standard-reflection read of F passes through its source and target seal rows, transport row, stream windows, regular readback, dyadic tolerance ledger, classifier row, terminal $\mathsf{Real}^{\uparrow}$ endpoint, and the displayed H_F, C_F, P_F, N_F rows. Under the componentwise agreements in the statement, this is the same terminal seal route as the diagonal packet.

Next attach the uniform-Cauchy consumer side. Apply Definition 435.57. The $\mathsf{RegSeqRat}^{\uparrow}$-, $\mathsf{Real}^{\uparrow}$-, and completion-facing reads of U are exhausted by the finite package $I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N$ so the seal row E_U is downstream of the same window, tolerance, readback, and structural rows.

Conversely, suppose the three accepted packets name one terminal $\mathsf{Real}^{\uparrow}$ seal route. The diagonal correspondence theorem forces the diagonal route through $W, Q, D, S, L, H, C, P, N$; the functor bridge theorem forces the standard-reflection route through $W_{F,0}, W_{F,1}, R_F, D_F, S_F, T_F, E_F, H_F, C_F, P_F, N_F$; and the uniform-Cauchy factorization forces the consumer route through $W_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U$. Since each cited theorem excludes off-packet quotient, equality, selector, choice, limit, and completeness coordinates, the only possible witness of sameness is componentwise agreement through the displayed transport, continuation, provenance, and local naming rows. □

Theorem 646.10 (Diagonal-limit compatibility classifier transport)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

componentwise hsame *rows* H *and displayed* Cont *routes* C *preserve the same compatibility classifier for the diagonal packet, triangle-bound row, seal route, dyadic ledger, selected windows, readback row, and real seal class. Transport cannot change the classifier by adding quotient streams, selected host limits, host real equality, choice data, or ambient completeness.*

Proof. Project H and C from the carrier. They are the only rows that move displayed coordinates between classifier presentations.

Apply Definition 646.2. Its classifier-transport clause states that transport is componentwise over the rows R, T, S, D, W, Q, L and their displayed continuation routes.

The non-escape theorem then fixes the transport boundary. Because no extra quotient, host-limit, equality, choice, or completeness coordinate is exported, the transported packet remains on the same $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ classifier surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityClassifierTransport`

Theorem 646.11 (Diagonal-limit compatibility non-escape)**.** *An accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet exports only the rows*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Every downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the accepted* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit packet, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *estimate, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the displayed* hsame *transports, the* Cont *routes,* Pkg *provenance, and the local* NameCert *row. The packet exports no quotient stream, no selected host limit, no host real equality, no choice datum, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityNonEscape`

Proof. Start from the accepted carrier $K = (R, T, S, D, W, Q, L, H, C, P, N)$. The displayed rows name exactly the diagonal-limit source, triangle estimate, seal route, dyadic ledger, window family, regular rational readback, real seal class, transports, continuations, provenance, and local certificate.

Use the NameCert obligation theorem. Its alignment rows force the triangle estimate and the seal route to consume the same selected windows and tolerance ledger already named by the diagonal-limit packet.

The only consumer-facing paths are the continuations in C, transported componentwise by H. None of those paths has an input coordinate for quotient streams, selected host limits, host real equality, choice data, or ambient completeness, so none of those objects can be exported from the packet. □

Theorem 646.12 (Diagonal-limit root dependency triad)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *window family W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q before the* $\mathsf{Real}^{\uparrow}$ *seal row L is available to a downstream consumer. The same packet displays the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route S, the* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal source R, and the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row T as named side rows for that read.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root dependency triad`

Proof. Project the carrier $K = (R, T, S, D, W, Q, L, H, C, P, N)$. The rows D, W, Q are listed before the seal row L, and the side rows S, R, T are displayed in the same finite packet.

Apply Definition 646.2. Its selector and seal-route clauses force the triangle row and seal route to read the shared dyadic ledger, stream windows, and regular-rational readback before the $\mathsf{Real}^{\uparrow}$ seal row can be consumed.

Finally use Definition 646.11. No ambient completion, host limit, quotient stream, equality row, or choice datum supplies another route to L. □

Theorem 646.13 (Diagonal-limit shared modulus handoff)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose its Real-completion consumer route reads an accepted* CauchyModulusExtraction$^{\uparrow}$ *packet over the same finite budget, window family, regular-rational schedule, and dyadic comparison rows, and suppose it also reads an accepted* CauchyModulusMeet$^{\uparrow}$ *packet over the same two regular-rational stream readbacks. Then the route to the* Real$^{\uparrow}$ *seal row L passes through one shared finite modulus surface: the selected budget row, the* StreamName$^{\uparrow}$ *windows, the* RegSeqRat$^{\uparrow}$ *schedule and readback, the* DyadicRatCore$^{\uparrow}$ *ledger, the extracted modulus row, and the* CauchyModulusMeet$^{\uparrow}$ *shared threshold. No quotient stream, selected host limit, host real equality, trichotomy, choice datum, or ambient completeness principle supplies another handoff to L.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier shared modulus handoff`

Proof. Start with the accepted diagonal packet K. Apply Definition 646.12; the route to L has already read the dyadic ledger D, the stream windows W, and the regular-rational readback Q, with R, T, S displayed as the source, triangle, and seal side rows.

Next read the modulus-extraction packet on that same budget and window surface. By Definition 641.22 and Definition 641.15, the Real-completion consumer can reach the extracted modulus only after the selected budget row, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ schedule, DyadicRatCore$^{\uparrow}$ ledger, and extracted modulus row have all been displayed inside the finite extraction packet.

Now attach the meet packet to the two regular-rational stream readbacks. Apply Definition 462.41. The meet row determines one shared threshold packet for the paired reads, so the seal-facing route cannot split the extracted modulus and the paired readbacks into separate threshold surfaces.

Finally use Definition 646.11. Since the diagonal compatibility packet exports no quotient stream, selected host limit, host real equality, trichotomy row, choice datum, or ambient completeness principle, the shared budget-window-schedule-dyadic-modulus-meet surface is the only handoff available before L. □

Theorem 646.14 (Diagonal-limit compatibility finite-prefix budget handoff)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Let*

$$F = (Q_F, B_F, W_F, D_F, R_F, S_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted FinitePrefixLimitBudget$^{\uparrow}$ *packet serving the same* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback, and Real or Cauchy-limit seal boundary as K under the displayed* hsame *transports. Then every diagonal-limit or Real-completion consumer of K that reads F must pass through the finite prefix route*

$$Q_F \to B_F \to W_F \to D_F$$

before reading the diagonal-limit readback row Q, the CauchyLimitSeal$^{\uparrow}$ *route S, or the* Real$^{\uparrow}$ *seal row L. The handoff exports no quotient stream, selected host limit, host real equality, countable choice datum, hidden tail schedule, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier finite prefix budget handoff`

Proof. Start with the finite-prefix packet F. Apply Definition 919.11 and Definition 919.19. These theorems place every diagonal-selector, diagonal-limit, and Real-completion read behind the finite route $Q_F \to B_F \to W_F \to D_F$ before the regular readback row R_F or seal row S_F is available.

Now attach the diagonal-limit packet K. The shared $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and Real or Cauchy-limit seal boundary identify W_F, R_F, S_F with the corresponding W, Q, S, L surface of K through the displayed hsame transports and Cont routes.

Use Definition 646.59 and Definition 646.15. The first theorem identifies the public budget route through $R, T, D, W, S, Q, L, H, C, P, N$, while the second forces any compatible tail handoff to meet the same finite window, dyadic ledger, readback, and seal rows before a completion-facing read reaches Q or L.

Finally apply Definition 646.11. Since the diagonal-limit packet exports no quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness row, and the finite-prefix packet records hidden-tail and outside-prefix refusals in E_F, there is no route to Q, S, or L except through $Q_F \to B_F \to W_F \to D_F$. □

Theorem 646.15 (Diagonal-limit shared-tail budget factorization)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let F be a compatible* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet whose finite windows, dyadic ledger, regular readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows agree with W, D, Q, L under the displayed* hsame *transports. Then every completion-facing route from K to the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *paired seal factors through the fusion shared-threshold row before the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route reads Q and L. The factorization exports no quotient stream, host limit, hidden tail oracle, choice datum, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySharedTailBudgetFactorizati`

Proof. Project the diagonal packet K. The displayed rows expose exactly the source R, triangle row T, seal route S, dyadic ledger D, window family W, regular readback Q, Real seal row L, transports H, continuations C, provenance P, and local name N.

Apply Definition 646.12. It orders the budget-facing rows D, W, Q, L before a downstream consumer can read the Real seal, so the compatible fusion packet must meet the same finite window and ledger surface through the displayed hsame transports.

Now use the fusion handoff rows Definition 658.6 and Definition 658.5. They force the paired finite-window and dyadic-budget reads through the shared-threshold row before any seal-facing read can reach the diagonal readback.

Finally apply Definition 646.11 and route the paired agreement consumer through chapter 699 only after that shared-threshold row has been displayed. The non-escape boundary excludes every quotient stream, host limit, hidden tail oracle, choice datum, and ambient completeness route outside the two finite packets. □

Theorem 646.16 (Diagonal-limit consumer threshold)**.** *A downstream completion-facing consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet may inspect the readback and seal rows Q, L only after the seven dependency rows*

$$R, T, S, D, W, Q, L$$

and the local rows

$$\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

have been displayed by the packet.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityConsumerThreshold`

Proof. Start with the carrier projection. The mathematical dependency rows R, T, S, D, W, Q, L and the local structural rows H, C, P, N are the complete displayed surface of the packet.

Use Definition 646.2. The local certificate admits only that carrier, its componentwise hsame transports, its displayed Cont routes, its Pkg provenance, and its own NameCert row.

Apply Definition 646.11. Since no undisplayed completion or host coordinate is exported, every consumer of Q, L has already crossed the displayed dependency and local rows. □

Theorem 646.17 (Diagonal-limit audit handoff)**.** *The root-unblock surface for* DiagonalLimitCompatibility$^{\uparrow}$ *is the package formed by Definition 646.12, Definition 646.16, and Definition 646.11. It exposes the triad D, W, Q, the side rows S, R, T, and the local* BHist/hsame/Cont/Pkg/NameCert *boundary without importing ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityAuditHandoff`

Proof. The root dependency triad names the required ledger, window, and readback rows and records the Cauchy seal, Real completeness, and triangle-bound side rows.

The consumer-threshold theorem then places every downstream read of Q, L behind the seven dependency rows and the local structural rows.

The non-escape theorem supplies the audit boundary. It removes quotient streams, selected host limits, host real equality, choice data, and ambient completeness from the handoff, so the three cited results form the complete root-unblock surface. □

Theorem 646.18 (Diagonal-limit obligation-closure package)**.** *An accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

forms an obligation-closure surface for the diagonal-limit handoff. The surface packages the triangle selector T, the CauchyLimitSeal$^{\uparrow}$ *route S, the shared* StreamName$^{\uparrow}$ *window ledger W, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, the* RegSeqRat$^{\uparrow}$ *readback Q, the* Real$^{\uparrow}$ *seal class L, componentwise* hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and the local* NameCert *row N as the only rows available to downstream completion consumers.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityObligationClosurePackage`
↪ variant `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTasteGate single carrier alignment`

Proof. Project the carrier of Definition 646.1. The displayed coordinates already list the diagonal source, triangle row, seal route, tolerance ledger, window ledger, readback row, real seal class, transports, continuations, provenance, and local certificate.

Apply Definition 646.2, Definition 646.6, and Definition 646.7. These results identify the obligation rows, fix the triangle selector to W, D, and force the seal route to reach Q, L only through R, T, D, W.

Finally use Definition 646.10 and Definition 646.11. Transport preserves the same classifier rows, and non-escape excludes quotient streams, selected host limits, host real equality, choice data, and ambient completeness. Thus the listed packet is the complete obligation-closure surface. □

Theorem 646.19 (Diagonal-limit shared window ledger)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the StreamName$^{\uparrow}$ *row W is the shared finite window ledger for the diagonal selector, triangle-bound estimate, dyadic tolerance read,* RegSeqRat$^{\uparrow}$ *readback, and downstream* CauchyLimitSeal$^{\uparrow}$ *consumer. No consumer may replace W after entering the displayed compatibility packet.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySharedWindowLedger`

Proof. Start from the carrier projection. The window row W is displayed together with the diagonal packet R, triangle row T, dyadic ledger D, seal route S, readback row Q, and real seal class L.

Use Definition 646.6. It fixes the triangle row to the same selected windows named by R, W, D and excludes a second selector surface.

Use Definition 646.7 and Definition 646.10. The seal route can reach Q, L only after W has been fixed, and transport preserves that same compatibility classifier.

The non-escape theorem, Definition 646.11, supplies the boundary. Since no alternate window family is exported, W remains the shared ledger for every downstream read. □

Theorem 646.20 (Diagonal-limit seal consumer threshold)**.** *In an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

a downstream CauchyLimitSeal$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer may inspect the* RegSeqRat$^{\uparrow}$ *readback Q and the* Real$^{\uparrow}$ *seal class L only after the diagonal packet R, triangle selector row T, dyadic tolerance ledger D, shared window ledger W, componentwise* hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N have been read.*

Proof. Project the accepted carrier and isolate the consumer-facing rows S, Q, L. The seal route is part of the displayed packet and is not an independent completion principle.

Apply Definition 646.2. Its seal-compatibility and classifier-transport clauses place the read of Q, L behind the diagonal route, triangle estimate, tolerance ledger, window family, transports, continuations, provenance, and local naming row.

Now apply Definition 646.7. The route S cannot bypass R, T, D, W when it reaches Q and L.

Finally use Definition 646.11. Because the packet exports no selected host limit, host real equality, choice datum, or ambient completeness principle, the displayed rows form the threshold for every downstream seal consumer. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility seal consumer threshold`

Theorem 646.21 (Diagonal-limit source ledger)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ BHist *carrier*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the source ledger is exactly the displayed RealCompleteness$^{\uparrow}$ *row R, triangle row T, seal route S, dyadic tolerance ledger D, stream-window row W, regular-rational readback row Q,* Real$^{\uparrow}$ *seal row L, componentwise* hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N of Definition 646.1. By Definition 646.11, no other source row is available to the compatibility packet.*

Proof. Project the carrier of Definition 646.1. The coordinates R, T, S, D, W, Q, L are the only source, estimate, seal, tolerance, window, readback, and real-seal rows in the packet.

Read the structural rows H, C, P, N. They do not create further source data; they transport, route, record provenance, and name the already displayed coordinates.

Apply Definition 646.11. Since the packet exports no host limit, quotient stream, choice datum, host real equality, or ambient completeness principle, the listed rows are the whole source ledger. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySourceLedger`

Theorem 646.22 (Diagonal-limit consumer transport)**.** *For accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ BHist *carriers* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *and* $K' = (R', T', S', D', W', Q', L', H', C', P', N')$*, every downstream seal or* $\mathsf{Real}^{\uparrow}$ *consumer that reads through the carrier of Definition 646.1 factors through the displayed componentwise* hsame *rows in* H, H' *and the displayed* Cont *routes in* C, C'*, with* Pkg *provenance* P, P' *and local* $\mathsf{NameCert}$ *rows* N, N' *preserving the same compatibility surface. By Definition 646.11, no consumer may bypass those rows.*

Proof. Start with the two accepted carriers and isolate the consumer-facing rows S, Q, L and S', Q', L'. The carrier definition places both reads behind the same diagonal, triangle, tolerance, and window coordinates.

Use the componentwise transport rows H, H'. They identify the source rows by hsame before a consumer reaches the seal or $\mathsf{Real}^{\uparrow}$ rows, so the classifier transported to the consumer is the displayed one.

Use the continuation rows C, C' next. Every read of the seal route, regular readback, or real-seal row is replayed through the displayed Cont routes, and P, P', N, N' record only that finite route surface.

Finally apply Definition 646.11. Since no alternate host-limit, quotient-stream, choice, or ambient-completeness route is exported, downstream consumers factor through the displayed transport and continuation rows. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityConsumerTransport`

Theorem 646.23 (Diagonal-limit ambient completeness refusal)**.** *An accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ BHist *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

from Definition 646.1 exports no host limit, no quotient stream, no choice datum, and no ambient completeness principle. Its only exported surface is the displayed source ledger, componentwise hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*; this is exactly the non-escape boundary of Definition 646.11.*

Proof. Project the packet K. The visible coordinates are the diagonal packet, triangle row, seal route, tolerance ledger, window row, readback row, real seal row, transport, continuation, provenance, and naming row.

Inspect the omitted coordinates. There is no row for a selected host limit, no row for quotienting streams, no row for a choice datum, and no row for an ambient completeness principle.

Apply Definition 646.11. The non-escape boundary therefore turns the omission into a consumer rule: a downstream chapter may read the displayed compatibility packet, but it may not treat that read as host completeness or as a hidden quotient construction. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityAmbientCompletenessRefusal`

Theorem 646.24 (Diagonal-limit source gate)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every accepted downstream read first fixes the RealCompleteness$^{\uparrow}$ *diagonal packet* R*, the* StreamName$^{\uparrow}$ *window family* W*, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* D*, and the* RegSeqRat$^{\uparrow}$ *readback row* Q *before any consumer treats the seal route* S *or the* Real$^{\uparrow}$ *seal row* L *as available. The source gate is exactly the route displayed by Definition 646.1; it is not a host limit selection and not an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySourceGate`

Proof. Project the carrier K from Definition 646.1. The diagonal source R, windows W, dyadic ledger D, readback Q, seal route S, and seal row L are all named coordinates of the same finite packet.

Apply Definition 646.6 and Definition 646.20. The selector alignment fixes W, D at the diagonal source, and the threshold theorem places every downstream seal read behind R, T, D, W, H, C, P, N.

Use Definition 646.7 to route the read of Q, L through the displayed rows. Then use Definition 646.10 to keep the same classifier under H and C.

Finally apply Definition 646.11. Since the packet exports no selected host limit, host real equality, choice datum, or ambient completeness principle, the displayed source gate is the only way a downstream consumer reaches the seal-facing rows. □

Theorem 646.25 (Diagonal-limit triangle-seal interface)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the RegularCauchyTriangleBound$^{\uparrow}$ *row* T *is the sole public interface between the diagonal selector carried by* R, W, D *and the* CauchyLimitSeal$^{\uparrow}$ *route* S*. A consumer may pass from the diagonal source to the seal route only through the triangle row together with the displayed dyadic ledger, stream-window ledger, classifier transport, continuation routes, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility triangle seal interface`

Proof. Start with the carrier. The only row naming the three-leg estimate between the diagonal source and the seal route is T, and the only ledgers available to it are D and W.

Apply Definition 646.6. It binds T to the same selected windows and dyadic tolerance ledger named by R, W, D, so the selector side has no second interface row.

Next apply Definition 646.7. The route S reaches Q, L only through R, T, D, W, so the seal side also has no bypass around the triangle estimate.

Classifier transport and non-escape close the interface. By Definition 646.10 the same interface is preserved through H, C, and by Definition 646.11 no quotient stream, selected host limit, choice datum, or ambient completeness coordinate can supply another route. □

Theorem 646.26 (Diagonal-limit downstream consumer gate)**.** *Every* Real$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

factors through the displayed rows $R, T, S, D, W, Q, L, H, C, P, N$*. The consumer may use the diagonal packet, triangle interface, seal route, dyadic ledger, stream-window ledger, regular rational readback, real seal row, componentwise transport, continuation routes, package provenance, and local naming row, but it cannot introduce host limits, quotient streams, choice data, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityDownstreamConsumerGate`

Proof. Read the consumer from the carrier. The displayed coordinates already list the only source, interface, route, ledger, readback, transport, continuation, provenance, and naming rows available to the packet.

Use Definition 646.24 to force the source-side order. The diagonal packet, stream windows, dyadic ledger, and regular rational readback are fixed before the consumer treats the seal-facing rows as available.

Use Definition 646.25 together with Definition 646.7. These theorems force the route from the diagonal selector to the seal consumer through T, D, W, S and then to Q, L.

Finally apply Definition 646.10 and Definition 646.11. Transport preserves the displayed classifier, and non-escape excludes every host-limit, quotient-stream, choice, host-equality, or ambient-completeness coordinate not carried by K. □

Theorem 646.27 (Diagonal-limit compatibility window-budget triad lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

must read the same triad (D, W, Q) *before the terminal seal row* L *is available. Here* W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. The three rows form one shared route through the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surface; no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness principle can supply a second triad.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility window budget triad lock`

Proof. Project the accepted packet from Definition 646.1. The rows D, W, Q occur as named coordinates of the same finite packet as the terminal seal row L.

Apply Definition 646.6. It binds the selector side to the displayed $\mathsf{StreamName}^{\uparrow}$ windows W and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, so the window and tolerance coordinates are fixed before any seal-facing read.

Next apply Definition 646.7. The seal route reaches the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal row L only through the same R, T, D, W route, so Q is not a separate host readback channel.

Use Definition 646.10 to carry this route through the displayed H and C rows. The transported consumer therefore sees the same (D, W, Q) triad that the carrier exposes.

Finally apply Definition 646.11. Since the packet exports no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate, no downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can replace the displayed triad before reading L. □

Theorem 646.28 (Diagonal-limit shared modulus terminal seal)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

any terminal $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal read that factors through a compatible* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *packet*

$$M = (E, B, W, S, \mathsf{Mu}, Q, H, C, P, N)$$

and through the finite-window budget envelope of $\mathsf{CauchyLimitSeal}^{\uparrow}$ *passes through one shared finite route. The route first reads* $R, T, S, D, W, Q, L, H, C, P, N$ *from the diagonal-limit packet and then*

reads the extraction rows $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$ as the same budget-window-modulus surface. It exports no quotient stream, selected host limit, host real equality, trichotomy, choice datum, or ambient completeness.

Proof. Start with the accepted diagonal packet. By Definition 646.11, every seal-facing read of K is confined to $R, T, S, D, W, Q, L, H, C, P, N$ and cannot import a quotient stream, selected host limit, host real equality, trichotomy, choice datum, or ambient completeness.

Next attach the compatible modulus-extraction packet. The public export of Definition 641.22 exposes exactly $E, B, W, S, \mathsf{Mu}, Q, H, C, P, N$, and the consumer bridge of Definition 641.23 forces Real-completion reads of Mu through those displayed rows.

Finally align the seal envelope. The finite-window budget equivalence of Definition 386.77 identifies the terminal seal read with the same finite budget-window route, and the verification handoff of Definition 386.76 repacks that route as the reusable $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface. Thus the Real-facing terminal read is the same finite budget-window-modulus route already displayed by the diagonal packet and the extraction packet, with no extra completion coordinate. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySharedModulusTerminalSeal`

Theorem 646.29 (Diagonal-limit window synchronizer pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$Z = (W_Z, Q_Z, S_Z, L_Z, D_Z, H_Z, C_Z, P_Z, N_Z)$$

be an accepted $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet whose* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *seal row, and shared threshold-budget rows agree componentwise with W, Q, S, L, D through the displayed* hsame *and* Cont *routes. Then every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of K pulls back through the synchronizer finite window-to-seal packet Z.*

The pullback uses only the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the two accepted packets. It does not introduce quotient streams, selected host limits, host real equality, choice data, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityWindowSynchronizerPullback`

Proof. First lock the diagonal side. Apply Definition 646.27 to force every terminal read of K through the common (D, W, Q) triad before L is available. Then apply Definition 646.28 to make the terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ reads use the same finite budget-window-modulus route.

Next read the synchronizer side. The budget pullback of Definition 979.3 places the synchronizer threshold and window rows over the same finite budget surface. The limit-seal compatibility of Definition 979.5 carries the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row through that surface, and the Real-seal exposure of Definition 979.6 exposes the terminal $\mathsf{Real}^{\uparrow}$ seal only after the same window-to-seal route.

It remains to identify the two finite routes. The hypotheses give componentwise hsame and Cont transport from the diagonal rows W, Q, S, L, D to the synchronizer rows W_Z, Q_Z, S_Z, L_Z, D_Z. Transport through the displayed BHist, Pkg, and $\mathsf{NameCert}$ rows repacks the consumer as a read of Z without changing the terminal object.

Finally use Definition 979.4. Since the synchronizer packet exports no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate, the pulled-back consumer has no route outside the finite window-to-seal packet. □

Theorem 646.30 (Diagonal-limit compatibility window source exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P,$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every accepted diagonal-limit consumer route that reaches the* Real$^{\uparrow}$ *seal row L first factors through the displayed source ledger, the window-budget triad (D, W, Q), the shared modulus route, the synchronizer pullback surface, and the seal route rows S, L.*

The exhausted route exports no quotient stream, selected host limit, ambient completeness principle, or independent real-equality coordinate. Its only available coordinates are the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of K together with the compatible finite-window packets through which the route is explicitly pulled back.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility window source exhaustion`

Proof. Begin with the source ledger. Apply Definition 646.21 to identify the readable source rows as $R, T, S, D, W, Q, L, H, C, P, N$ and to exclude any hidden host-limit or quotient-stream source.

Next lock the finite window route. The shared window ledger and consumer transport of Definition 646.27 force the consumer through (D, W, Q) before the seal row is available, while Definition 646.28 identifies the terminal seal read with the same finite budget-window-modulus route.

The ambient-completeness refusal is supplied by Definition 646.23. It prevents the consumer from replacing the displayed finite route by a host completeness principle, selected limit, or host real-equality coordinate.

Finally apply Definition 646.29. The terminal consumer is repacked through the synchronizer finite window-to-seal packet, and the carrier projection together with the non-escape clauses of the displayed packets leaves no coordinate outside the stated source, window, modulus, synchronizer, and seal rows. □

Theorem 646.31 (Diagonal-limit compatibility formal target packet)**.** *Every Lean-facing formal target for an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet exposes exactly the carrier rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

together with the compatible finite-window packets cited by Definition 646.30. The target may consume RealCompleteness$^{\uparrow}$*,* RegularCauchyTriangleBound$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *only through those displayed rows and their finite-window pullbacks.*

The formal target has no coordinate for quotient streams, selected host limits, host real equality, choice data, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityFormalTargetPacket`

Proof. Project the carrier of the accepted diagonal-limit packet. The projection displays exactly $R, T, S, D, W, Q, L, H, C, P, N$, and the local NameCert obligations make the displayed BHist, hsame, Cont, Pkg, and NameCert rows the only routes available to a formal target.

Next apply the terminal Real route and the window-budget triad lock. The terminal route keeps the Real-facing consumer on $R, T, S, D, W, Q, L, H, C, P, N$, while Definition 646.27 forces any seal-facing read through the displayed (D, W, Q) finite-window triad.

Now attach the shared modulus terminal seal and the synchronizer pullback. Definition 646.28 identifies the terminal Real$^{\uparrow}$ and CauchyLimitSeal$^{\uparrow}$ reads with one finite budget-window-modulus route, and Definition 646.29 repacks that route through the compatible RealWindowSynchronizer$^{\uparrow}$ packet.

Finally use Definition 646.30. It exhausts the source, window, shared modulus, synchronizer, and seal rows available to the Lean-facing packet. The named upstream chapters may be consumed only through these rows, so quotient streams, selected host limits, host real equality, choice data, and ambient completeness have no target coordinate. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility formal target packet`

Theorem 646.32 (Diagonal-limit finite-window-only consumer)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *consumer of* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is admitted only through the displayed finite rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

of its accepted packet and through the compatible finite-window packets cited by Definition 646.30. A route that uses quotient streams, host real equality, selected host limits, choice data, or ambient completeness is outside the carrier and is not an admitted consumer route.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityFiniteWindowOnlyConsumer`

Proof. Start with the source ledger. By Definition 646.21, the accepted packet exposes only $R, T, S, D, W, Q, L, H, C, P, N$ together with its displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Lock the finite route before the terminal read. The window-budget triad lock of Definition 646.27 forces the consumer through (D, W, Q), and Definition 646.28 makes the terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ reads share the same finite-window seal route.

Next pull the route through the synchronizer. Apply Definition 646.29 to repack the terminal consumer through the compatible finite window-to-seal packet without adding a host coordinate.

Finally use Definition 646.30. It exhausts the admissible source, triad, shared modulus, synchronizer, and seal coordinates, while Definition 646.11 removes quotient streams, host real equality, selected host limits, choice data, and ambient completeness from the carrier. Therefore every accepted $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer is finite-window-only. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier finite window only consumer`

Theorem 646.33 (Diagonal-limit triangle seal route)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the seal route S reaches the readback row Q and the $\mathsf{Real}^{\uparrow}$ *seal row L only after the triangle row T has been aligned with the dyadic ledger D and the shared stream-window row W. The route is the one exposed by Definition 646.1, Definition 646.2, Definition 646.6, and Definition 646.7; it cannot bypass classifier transport, local provenance, or the non-escape boundary.*

Proof. Project the carrier K. The displayed packet places R, T, S, D, W, Q, L before the transport, continuation, provenance, and local naming rows H, C, P, N.

Apply triangle-selector alignment. It fixes the triangle row T to the same dyadic tolerance ledger D and stream-window family W already named by the diagonal packet.

Now apply seal-route compatibility. The seal route can read Q and L only through R, T, D, W, so the route to the completion-facing rows has already consumed the triangle row and its two ledgers.

Classifier transport and non-escape close the boundary. Transport preserves the same route through H and C, while non-escape excludes quotient streams, selected host limits, host real equality, choice data, and ambient completeness. □

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTriangleSealTransportRoute
↪ variant BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility triangle seal route

Theorem 646.34 (Diagonal-limit window ledger exactness)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the row W is the exact shared-window ledger read by the diagonal source R, the triangle row T, the dyadic ledger D, the seal route S, the readback row Q, and the Real$^{\uparrow}$ *seal row L. A completion consumer reaches Q, L only after it has read the same W through the triangle row, dyadic ledger, componentwise* hsame *transport H, displayed* Cont *route C, package provenance P, and local* NameCert *row N.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility window ledger exactness

Proof. Begin with the shared-window ledger theorem. It identifies W as the finite window row used by the diagonal selector, triangle-bound estimate, dyadic tolerance read, regular readback, and downstream seal consumer.

Use the triangle-seal route theorem. Since the route to Q, L must pass through T, D, W, no consumer can replace W after the triangle row has been fixed.

Finally apply classifier transport and the NameCert obligations. The componentwise hsame and Cont rows preserve the displayed classifier, and the local NameCert row records that the same window ledger is the one carried by K. □

Theorem 646.35 (Diagonal-limit completion boundary)**.** *Every completion-facing or* Real$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

reaches Q, L only after reading the triangle row T, the dyadic ledger D, the stream-window ledger W, the componentwise hsame *transport H, the displayed* Cont *route C, package provenance P, and the local* NameCert *row N. The boundary is the one assembled from the carrier, local NameCert obligations, triangle selector alignment, seal route compatibility, classifier transport, non-escape, the shared-window ledger, Definition 646.33, and Definition 646.34.*

The consumer receives no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness principle.

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCompletionBoundary

Proof. First assemble the route. The carrier supplies the rows of K, the local NameCert obligations identify their order, and triangle selector alignment fixes T to the displayed D, W ledgers.

Next apply seal route compatibility together with the triangle-seal route theorem. These results force every read of Q, L through the route determined by R, T, D, W, S.

Use window ledger exactness to keep the stream-window row fixed through the same route. The componentwise hsame transport, displayed Cont route, package provenance, and local NameCert row then carry that fixed route to the completion consumer without adding another ledger.

Finally use non-escape. Since no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate is present in the packet, the displayed rows are the complete completion boundary. □

Theorem 646.36 (Diagonal-limit compatibility terminal formal target)**.** *An accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reaches any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *terminal consumer only through the displayed diagonal route, triangle row, dyadic ledger, shared stream-window ledger, readback row,* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row:*

$$K = (R, T, S, D, W, Q, L, H, C, P, N).$$

This is the exact formalization target for the present packet. It exports no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility terminal formal target`

Proof. Start with the triangle seal route and window ledger exactness. Definition 646.33 places the seal route behind R, T, D, W, and Definition 646.34 keeps the shared stream-window ledger fixed through the readback and seal rows.

Use the completion boundary, Definition 646.35. It forces any terminal $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to pass through the triangle row, dyadic ledger, stream-window ledger, transport, continuation, provenance, and local naming rows before Q or L is read.

Next apply tolerance ledger handoff and seal consumer factorization. Definition 646.37 fixes the dyadic ledger and window family, while Definition 646.38 routes the terminal seal read through the displayed packet rows.

The same rows define the formal target because every cited theorem consumes only $K = (R, T, S, D, W, Q, L, H, C, P, N)$. Non-escape supplies the final exclusion: quotient streams, selected host limits, host real equality, choice data, and ambient completeness have no coordinate in the packet and cannot be exported. □

Theorem 646.37 (Diagonal-limit tolerance ledger handoff)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every tolerance-facing read that reaches the seal route first uses the same $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *and* $\mathsf{StreamName}^{\uparrow}$ *window family* W *named by the diagonal packet* R *and triangle row* T*. No downstream tolerance consumer can replace* D *or* W *with an alternate ledger or window family after entering the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route.*

Proof. Project the carrier K. The rows R, T, D, W are displayed before the seal route S can be read.

Apply Definition 646.6. It identifies the triangle row T with the same selected window family W and dyadic tolerance ledger D already named by the diagonal packet R.

Finally use non-escape, Definition 646.11. Since the packet contains no alternate tolerance ledger or second window family, every read that reaches S keeps the displayed D, W handoff. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityToleranceLedgerHandoff`

Theorem 646.38 (Diagonal-limit seal consumer factorization)**.** *Every downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet factors through*

$$R, T, D, W, Q, L, H, C, P, N$$

and cannot bypass the triangle, tolerance, or window rows. The seal consumer may read Q *and* L *only after the packet has exposed the diagonal route* R*, the triangle row* T*, the dyadic ledger* D*, and the window family* W*.*

Proof. Begin with Definition 646.7. It places every seal-route read of Q and L behind the displayed rows R, T, D, W.

Use classifier transport, Definition 646.10. The componentwise transports H and continuation routes C preserve the same compatibility classifier and do not add another seal-consumer path.

Apply non-escape, Definition 646.11. The exported surface is exhausted by the listed rows and their provenance, so no $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can bypass the triangle, tolerance, or window coordinates. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealConsumerFactorization`

Theorem 646.39 (Diagonal-limit window route lock)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the $\mathsf{StreamName}^{\uparrow}$ *window row* W *is locked before any triangle estimate, tolerance read, seal-route read, or* $\mathsf{Real}^{\uparrow}$ *read can be consumed. Every such consumer uses the same displayed window family* W *already named by the carrier and cannot choose a second window route after the packet has been accepted.*

Proof. Project the carrier of Definition 646.1. The row W is part of the accepted packet, displayed before the downstream triangle, tolerance, seal, and real-facing reads are interpreted.

Apply the shared-window ledger theorem, Definition 646.19. It ties the diagonal source route to the same stream-window row W used by the packet.

Now apply the tolerance ledger handoff theorem, Definition 646.37. Since the tolerance route reaches the seal through the fixed D, W pair, no later triangle, seal, or $\mathsf{Real}^{\uparrow}$ consumer has a fresh window coordinate from which to build an alternate route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityWindowRouteLock`

Theorem 646.40 (Diagonal-limit terminal selector non-escape)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every terminal $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *selector read of* K *passes through the same diagonal route* R*, triangle row* T*, dyadic ledger* D*, stream-window row* W*, readback row* Q*, seal row* L*, componentwise* hsame *transport* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. The terminal read cannot introduce a second selector, quotient stream, host limit, choice datum, or ambient completeness principle.*

Proof. Start with the window route lock, Definition 646.39. It fixes the stream-window row W before any terminal seal or $\mathsf{Real}^{\uparrow}$ read is consumed.

Next use tolerance ledger handoff, Definition 646.37. The dyadic ledger D and the window row W remain the same rows named by the diagonal route and triangle row.

Apply seal consumer factorization, Definition 646.38. It routes every $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read through $R, T, D, W, Q, L, H, C, P, N$ rather than through a detached terminal selector.

Finally use the completion boundary, Definition 646.35. Since the packet exposes no quotient stream, host limit, choice datum, or ambient completeness coordinate, the terminal selector cannot escape the displayed diagonal route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTerminalSelectorNonescape`

Theorem 646.41 (Diagonal-limit seal readback exactness)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer reaches the readback row* Q *and the seal row* L *only through the displayed route*

$$R, T, D, W, H, C, P, N.$$

Thus Q *and* L *are read as the packet's own seal readback and real seal coordinates, not as independently selectable completion data.*

Proof. Start with seal-route compatibility, Definition 646.7. It places the reads of Q and L behind the diagonal source row R, triangle row T, dyadic ledger D, and window row W.

Use seal consumer factorization, Definition 646.38. The factorization adds the transport, continuation, provenance, and local naming rows H, C, P, N to the same route and excludes bypasses around the displayed triangle, tolerance, and window data.

Consequently every seal consumer reads Q and L through the listed rows, so the readback and seal coordinates are exact coordinates of the accepted packet. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealReadbackExactness`

Theorem 646.42 (Diagonal-limit Real seal non-escape)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, the* $\mathsf{Real}^{\uparrow}$ *seal row* L *exports no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness principle. A* $\mathsf{Real}^{\uparrow}$ *consumer can read* L *only as the displayed seal coordinate reached through the packet route and cannot extract host-level completion data from it.*

Proof. Apply the non-escape theorem, Definition 646.11. The exported $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ surface contains only the named rows of the accepted packet and not any quotient stream, selected host limit, host real equality, choice datum, or ambient completeness principle.

Now use seal readback exactness, Definition 646.41. It forces any $\mathsf{Real}^{\uparrow}$ read of L to pass through the displayed seal route and readback rows, so L cannot be used as an escape hatch into a host-level completion object. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility real seal nonescape`

Theorem 646.43 (Diagonal-limit compatibility source-route exactness)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibili}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

there is one displayed source route from the accepted $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet* R *through the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal class* L*. This route is the route exposed by the carrier and by the local* $\mathsf{NameCert}$ *obligations, using the sibling certificate surfaces chapter 450, chapter 466, and chapter 386. It has no second source, no alternate triangle row, and no bypass around the displayed tolerance, window, readback, or seal coordinates.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySourceRouteExactness

Proof. Project the accepted carrier. Its displayed mathematical rows are exactly R, T, S, D, W, Q, L, and its transport, continuation, provenance, and local naming rows are exactly H, C, P, N.

Use the local NameCert obligation theorem to read the source route in order. The triangle-selector alignment row fixes T to the same D, W data consumed by the accepted diagonal packet R, matching the diagonal and triangle certificate surfaces.

Next use seal-route compatibility. The route S reaches Q and L only after R, T, D, W have been fixed, so the Cauchy-limit seal route and the RealUp seal are attached to the same displayed source route.

Finally apply classifier transport and non-escape. The componentwise hsame and Cont rows preserve this route, while the exported packet contains no coordinate from which a second source or bypass route could be formed. □

Theorem 646.44 (Diagonal-limit compatibility regular readback exactness)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the RegSeqRat$^{\uparrow}$ *readback row Q is determined by the same triangle row T, dyadic ledger D, and stream-window row W used by the diagonal source route. After D, W, T are fixed, the packet cannot choose another* RegSeqRat$^{\uparrow}$ *readback row or another dyadic witness for the seal handoff. This readback is the one compatible with chapter 291, chapter 126, and chapter 301.*

Proof. Start from the carrier projection and isolate D, W, T, Q. The dyadic ledger, window family, triangle estimate, and readback row are all displayed before a completion-facing consumer can read the seal class.

Apply triangle-selector alignment and the shared-window ledger theorem. These facts identify the selected windows and tolerance ledger consumed by T with the same W, D rows already exposed by the diagonal packet.

Now apply seal-route compatibility and the tolerance-ledger handoff theorem. The route to Q passes through the fixed D, W, T rows, so the readback row has no independent dyadic or window witness to select.

The non-escape theorem supplies the boundary. Since the packet exports no alternate dyadic ledger, second window family, or separate readback coordinate, Q is the exact regular-rational readback for the displayed compatibility route. □

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRegularReadbackExactness

Theorem 646.45 (Diagonal-limit compatibility route ledger totality)**.** *Every completion-facing consumer accepted by a* DiagonalLimitCompatibility$^{\uparrow}$ *packet factors through one displayed route ledger*

$$(R, T, D, W, Q, L, H, C, P, N).$$

Here T is the triangle row, D is the DyadicRatCore$^{\uparrow}$ *ledger, W is the* StreamName$^{\uparrow}$ *window row, Q is the* RegSeqRat$^{\uparrow}$ *readback, and L is the* Real$^{\uparrow}$ *seal. The rows H, C, P, N only transport, continue, package, and locally name that ledger. They do not add quotient streams, host-selected limits, host real equality, choice data, or ambient completeness.*

Proof. Begin with source-route exactness, Definition 646.43. It places the consumer route on the displayed diagonal source row R and its triangle, dyadic, window, readback, seal, transport, continuation, provenance, and NameCert rows.

Use regular readback exactness, Definition 646.44. This binds Q to the same T, D, W rows, so the readback cannot be selected from a separate dyadic ledger or a second stream window.

Next apply seal readback exactness, Definition 646.41. The $\mathsf{Real}^{\uparrow}$ seal row L reads the same regular-rational data and does not introduce an independent host real witness.

The route lock is supplied by Definition 646.39. It keeps the window row W attached to the triangle and dyadic route rather than to a consumer-chosen stream family.

Finally use Real seal non-escape, Definition 646.42. Since the packet has no coordinates for quotient streams, host-selected limits, host real equality, choice data, or ambient completeness, the displayed ledger is total for accepted completion-facing consumers. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRouteLedger` semantic name certificate

Theorem 646.46 (Diagonal-limit compatibility uniform-window Real seal pullback)**.** *Let*

$$K = (R, T, S_D, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet and let*

$$U = (F, M, T_U, R_U, S, E, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformLimit}^{\uparrow}$ *packet. Suppose the two packets share the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row of chapter 126 and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301. Then every consumer common to the* $\mathsf{Real}^{\uparrow}$ *and completion-facing routes of chapter 38 and chapter 463 factors through the finite pullback surface made of the diagonal rows*

$$R, T, D, W, Q, L, H, C, P, N$$

and the uniform-limit rows

$$F, M, T_U, R_U, S, E, H_U, C_U, P_U, N_U.$$

The pullback exports only the common finite window, regular readback, seal rows, displayed hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, pointwise-only limit, selected host limit, host real equality, or ambient completeness coordinate.*

Proof. First read the diagonal side. Source-route exactness fixes the diagonal rows R, T, D, W before the displayed seal route can be consumed, and regular readback exactness identifies Q as the readback from the same T, D, W rows.

Next read the uniform-limit side. The seal factorization for $\mathsf{UniformLimit}^{\uparrow}$ routes F, M, T_U through the regular handoff R_U, the seal row S, the endpoint row E, and the displayed support rows H_U, C_U, P_U, N_U.

The shared-window and shared-readback hypotheses identify the two consumer surfaces at the common $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. Pulling back along those rows leaves a finite BHist surface whose coordinates are exactly the displayed diagonal and uniform-limit rows.

Finally apply the non-escape clauses on both chapters. The diagonal packet has no coordinate for a quotient stream, selected host limit, host real equality, or ambient completeness theorem,

and the uniform-limit packet has no coordinate for a pointwise-only limit or selected host limit. Therefore any common Real or Completion consumer factors through the stated pullback surface and nowhere else. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityUniformWindowRealSealPullba`

Theorem 646.47 (Diagonal-limit compatibility selector-seal pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose its displayed route reads a* DiagonalTailSelector$^{\uparrow}$ *source-schedule packet and a* CauchyLimitSeal$^{\uparrow}$ *observation-budget packet before any terminal* Real$^{\uparrow}$ *consumer reads the seal row L. Then the selected finite windows, observation budget,* RegSeqRat$^{\uparrow}$ *readback, and* Real$^{\uparrow}$ *seal row pull back to the same* RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$*-facing carrier surface*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Every terminal Real$^{\uparrow}$ *consumer factors first through the* DiagonalTailSelector$^{\uparrow}$ *source-schedule readback and the* CauchyLimitSeal$^{\uparrow}$ *observation-budget pullback, and only then through the completion-facing read of Q and L.*

Proof. Begin with the uniform-window Real seal pullback of Definition 646.46. It already identifies the finite window, regular readback, Real seal, displayed transport, continuation, provenance, and local naming rows as one finite consumer surface.

Next read the selector side. The diagonal-tail selector handoff fixes the source schedule before the seal-facing route can reach Q or L, so the selected windows cannot be replaced by a second host selector after the packet has been accepted.

Now read the seal side. The CauchyLimitSeal$^{\uparrow}$ observation-budget and Real handoff routes force the budget row and seal row to travel through the same BHist, hsame, Cont, Pkg, and NameCert support rows. Thus the selector readback and seal budget meet at the displayed RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ carrier surface.

Finally inspect a terminal Real$^{\uparrow}$ consumer. Since the packet exports no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate, the consumer must pass through the selector source-schedule readback and the seal-budget pullback before any completion-facing read of Q and L is available. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility selector seal pullback`

Theorem 646.48 (Diagonal-limit compatibility completion consumer nonescape)**.** *Every completion-facing or* Real$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet factors through*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The consumer receives the accepted diagonal source route, triangle row, dyadic ledger, stream windows, seal route, regular readback, real seal class, componentwise transports, displayed continuations, provenance, and local naming row. It receives no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness theorem. The boundary is the one witnessed by chapter 38, Definition 646.43, and Definition 646.44.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.CompletionConsumerNonescape.DiagonalLimitCompatibilit`

Proof. Project the accepted packet and read the exported rows. The only consumer-facing coordinates are the seal route S, the readback Q, the real seal class L, and their displayed transports and continuations through H, C, P, N.

Use source-route exactness to fix the route from R through T, D, W, S, Q to L. Use regular readback exactness to fix Q as the readback determined by the same D, W, T rows.

It remains to inspect the consumer boundary. The existing non-escape theorem excludes quotient streams, selected host limits, host real equality, choice data, and ambient completeness from the exported packet.

Therefore every completion-facing or $\mathsf{Real}^{\uparrow}$ consumer factors through the displayed rows alone, and no host-level completion object can enter through the compatibility certificate. □

Theorem 646.49 (Diagonal-limit compatibility finite source lock)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every compatibility read starts from the displayed $\mathsf{RealCompleteness}^{\uparrow}$ *source row* R*, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *estimate row* T*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal row* L*, and the support rows* H, C, P, N*. No compatibility consumer may introduce a second source route or an ambient completion principle outside those rows.*

Proof. Project the carrier and read the finite row list $R, T, S, D, W, Q, L, H, C, P, N$. These are the only coordinates available to a compatibility consumer.

Use source-route exactness, Definition 646.43. It identifies the unique displayed route from R through T, D, W, S, Q to L.

Use completion-consumer non-escape, Definition 646.48. It forces every completion-facing or $\mathsf{Real}^{\uparrow}$ consumer to factor through the same listed rows.

The general non-escape theorem, Definition 646.11, excludes quotient streams, selected host limits, host real equality, choice data, and ambient completeness. Therefore the source lock is exactly the finite packet. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityFiniteSourceLock`

Theorem 646.50 (Diagonal-limit compatibility Real seal source lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet factors through the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route row* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *before it reaches the seal row* L*. The read cannot bypass* S *or* Q *by appealing to host real equality, a selected host limit, a quotient stream, or an ambient completion theorem.*

Proof. Start with seal-readback exactness, Definition 646.41. It binds the $\mathsf{Real}^{\uparrow}$ seal row L to the displayed seal route and readback rows.

Apply source-route exactness, Definition 646.43. The route to L passes through S and Q after the source, triangle, dyadic, and window rows have been fixed.

Use regular-readback exactness, Definition 646.44. The readback Q is determined by the same D, W, T rows and is not an independent host real witness.

Finally apply completion-consumer non-escape, Definition 646.48. Every seal-facing read remains inside S, Q, L and the packet support rows, with no host completion bypass. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRealSealSourceLock`

Theorem 646.51 (Diagonal-limit compatibility triangle-seal scope)**.** *In an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, the triangle estimate row* T *and the seal route row* S *share the same dyadic ledger row* D *and stream-window row* W*. A consumer may not read the triangle estimate against one tolerance/window ledger and the seal route against another.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTriangleSealScope`

Proof. Project the accepted packet and isolate T, S, D, W. The carrier has one dyadic ledger row and one stream-window row for both the triangle estimate and the route to the seal.

Use regular-readback exactness, Definition 646.44. It fixes Q from the same T, D, W rows used by the diagonal source route.

Use source-route exactness, Definition 646.43. The seal route S appears on that same route, after R, T, D, W and before Q, L.

Completion-consumer non-escape, Definition 646.48, then prevents a consumer from splitting the route. The triangle row and the seal row therefore share the displayed D, W ledger rows. □

Theorem 646.52 (Diagonal-limit compatibility window fusion)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the shared $\mathsf{StreamName}^{\uparrow}$ *window row* W *simultaneously serves the* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal packet* R*, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *row* T*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*. A consumer cannot split these rows into separate window surfaces after the packet has been accepted.*

Proof. Project the carrier and read the row inventory. The packet has exactly one window coordinate W and places it beside the diagonal source, triangle row, seal route, dyadic ledger, readback row, and Real seal row.

Apply the NameCert obligations, triangle-selector alignment, and seal-route compatibility. These force the reads of R, T, S, D, Q to pass through the displayed window coordinate rather than through independently chosen window data.

Now use window-ledger exactness and seal-readback exactness. The former binds the diagonal, triangle, dyadic, seal, readback, and seal rows to the same ledger W; the latter keeps the readback and seal route attached to that route.

Finally apply Real seal non-escape and triangle-seal scope. Since the accepted packet contains no second $\mathsf{StreamName}^{\uparrow}$ coordinate and no host completion row, every consumer sees one fused window surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityWindowFusion`

Theorem 646.53 (Diagonal-limit compatibility Real seal exactness)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *seal row* L *is reachable exactly through the displayed route*

$$R, T, S, D, W, Q, H, C, P, N.$$

It exports no quotient stream, selected host limit, host equality, choice datum, or ambient completeness principle.

Proof. Begin with the carrier and local NameCert obligations. They expose the route from the diagonal packet R through the triangle row T, seal route S, dyadic ledger D, shared window row W, readback row Q, and support rows H, C, P, N before the Real seal row can be consumed.

Use triangle-selector alignment, seal-route compatibility, and window fusion. These theorems keep the triangle, seal, tolerance, window, and readback rows on the same displayed route and prevent a consumer from selecting a separate window or tolerance surface.

Apply seal-readback exactness and Real seal non-escape. The read of L passes through the packet's own seal readback route, while quotient streams, selected host limits, host equality, choice data, and ambient completeness are absent from the carrier.

Finally use triangle-seal scope. It binds the triangle estimate and seal route to the same D, W ledgers, so the listed rows are the exact Real seal boundary. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility real seal exactness`

Theorem 646.54 (Diagonal-limit compatibility tail-selector scope)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, every tail-selector read is mediated by the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W *and the route* R, T, D, W, S, Q, L*. It cannot choose a host tail, a second window surface, or an ambient completion witness.*

Proof. Source-route exactness fixes the route from R through T, D, W, S, Q to L, while window fusion and Real seal source lock keep the selector attached to that same W row. Regular readback exactness identifies Q with the displayed T, D, W data, and completion-consumer non-escape excludes any host tail or ambient completion coordinate. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility tail selector scope`

Theorem 646.55 (Diagonal-limit compatibility Real–regular source factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *readback from an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *already carried by the packet. The readback has no source outside the cited regular, dyadic, stream-window, and seal rows.*

Proof. Apply regular readback exactness to bind Q to T, D, W, then apply source-route exactness to place this readback on the same route from R to L through S. Real seal source lock forces every seal-facing read through S, Q, L, and completion-consumer non-escape removes quotient streams, selected host limits, host real equality, choice data, and ambient completeness from the factorization. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRealRegseqSourceFactorizati`

Theorem 646.56 (Diagonal-limit compatibility completion seal budget)**.** *Any* $\mathsf{Completion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is bounded by the displayed seal budget* $R, T, S, D, W, Q, L, H, C, P, N$*. It may consume the packet as a finite compatibility route, but it cannot claim ambient completeness or a host-level completion object.*

Proof. The source route and Real seal source lock identify the whole budget ending at L, and regular readback exactness accounts for the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ coordinates used by that route. Completion-consumer non-escape permits reads through the displayed packet and rules out any additional completeness, limit-selection, quotient, choice, or host-real coordinate. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCompletionSealBudget`

Theorem 646.57 (Diagonal-limit compatibility regular Cauchy sequence handoff)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet and let*

$$K = (R_0, T, S_0, D_0, W_0, Q, L, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the selected-window rows* W, W_0*, the dyadic ledgers* D, D_0*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R, Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows* E, L *are related by the displayed* hsame *transports carried by* H *and* H_0*, and suppose the continuation, provenance, and local naming rows* C, P, N *and* C_0, P_0, N_0 *expose the same consumer-facing route. Then every* $\mathsf{Completion}^{\uparrow}$*-facing read of* K *that consumes the regular-family data of* S *factors first through Definition 660.11 and Definition 660.16, then through Definition 697.6, Definition 646.38, Definition 386.5, and Definition 646.56. The resulting route exports the finite family row, modulus, selected windows, dyadic ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, tail-selector handoff, seal route,* $\mathsf{Real}^{\uparrow}$ *seal rows, displayed transports, continuations, provenance, and local* $\mathsf{NameCert}$ *rows, but exports no quotient stream, selected host limit, host real equality, choice data, hidden tail oracle, or ambient completeness principle.*

Proof. First read the accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ packet. The diagonal-window coverage handoff confines the family-facing data to $F, \mu, W, D, R, G, E, H, C, P, N$, so a downstream consumer sees the selected windows, dyadic ledger, regular readback, and Real seal handoff only through the displayed finite packet.

Next apply Definition 660.16. The matching hsame transports identify the regular packet's selected-window, dyadic, readback, and seal rows with the rows consumed by the diagonal-tail selector route.

Now pass through Definition 697.6. This attaches the selector route to the same finite windows and readbacks before any diagonal-limit or seal-facing consumer can read the compatibility packet.

Finally use the diagonal-limit compatibility and seal boundaries. The seal consumer factorization, the $\mathsf{CauchyLimitSeal}^{\uparrow}$ $\mathsf{Real}^{\uparrow}$ consumer boundary, and the completion seal budget force the remaining read through $R_0, T, S_0, D_0, W_0, Q, L, H_0, C_0, P_0, N_0$ and exclude quotient streams, selected host limits, host real equality, choice data, hidden tail oracles, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRegularCauchySequenceHandof`

Theorem 646.58 (Diagonal-limit compatibility source-route budget lock)**.** *For an accepted* DiagonalLimitCompatib *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every completion-facing or $\mathsf{Real}^{\uparrow}$ *consumer first reads the same* $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *source route before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* L*. The locked route is the finite row sequence*

$$R, T, S, D, W, Q, H, C, P, N$$

with sibling anchors chapter 697, chapter 126, chapter 291, chapter 386, and chapter 38.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility source route budget lock`

Proof. Project the carrier and isolate the source route R, T, S, D, W, Q together with the support rows H, C, P, N. The packet contains one diagonal source, one tail-selector route, one stream-window row, one dyadic ledger, one seal route, and one regular readback before the Real seal.

Apply source-route exactness and regular-readback exactness. These force the readback Q to be determined by the same T, D, W rows and keep the route from R to L inside the displayed source sequence.

Now use the finite source lock and the Real seal source lock. The first excludes a second source route, while the second forces every Real-facing read through S, Q, L after the tail-selector, stream-window, and dyadic rows have been read.

Finally apply completion-consumer non-escape. Since the packet exports no quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness row, the displayed route is the complete budget lock. □

Theorem 646.59 (Diagonal-limit compatibility public budget route)**.** *Every public Real-completion consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

factors through the displayed route

$$R, T, D, W, S, Q, L, H, C, P, N.$$

The $\mathsf{StreamName}^{\uparrow}$ *window row* W *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *are fixed before any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *readback row* Q *or* $\mathsf{Real}^{\uparrow}$ *seal row* L *is consumed. The route ties the* $\mathsf{RealCompleteness}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route, and* $\mathsf{Real}^{\uparrow}$ *seal to the same public budget surface.*

Proof. Begin with source-route budget lock, Definition 646.58. It forces every completion-facing or $\mathsf{Real}^{\uparrow}$ consumer through the finite source route before the Real seal row is read.

Use tolerance ledger handoff, Definition 646.37, and window ledger exactness, Definition 646.34. These keep the dyadic ledger D and the stream-window row W fixed for the diagonal source, triangle row, seal route, readback row, and Real seal.

Now apply seal consumer factorization, Definition 646.38. The public consumer reaches Q and L only through the displayed R, T, D, W rows together with the support rows H, C, P, N.

Finally apply Real seal non-escape, Definition 646.42. Since quotient streams, selected host limits, host equality, choice data, and ambient completeness are absent from the packet, the displayed public budget route is exhaustive. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility public budget route`

Theorem 646.60 (Diagonal-limit compatibility tail-selector consumer boundary)**.** *Let* $K = (R, T, S, D, W, Q, L, H$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any consumer that reads the tail selector used by a completion or* $\mathsf{Real}^{\uparrow}$ *route must first pass through the displayed* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route* S *before it may read* L*. It cannot choose a second tail selector, a host tail, or an ambient completion witness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTailSelectorConsumerBoundar`

Proof. Begin with source-route exactness. It places the tail-selector read on the route from the diagonal source R through T, D, W, S, Q to the Real seal row L.

Use regular-readback exactness to bind Q to the same T, D, W rows. The selected tail is therefore read through the StreamName and DyadicRatCore coordinates already present in the packet.

Apply the finite source lock and Real seal source lock. These results prevent the consumer from replacing the route by a second source path or by a direct Real seal read that bypasses S and Q.

Completion-consumer non-escape supplies the boundary. No host tail, quotient stream, selected host limit, choice datum, or ambient completion coordinate is carried by K, so the tail-selector consumer remains inside the displayed route. □

Theorem 646.61 (Diagonal-limit compatibility Real seal route exhaustion)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the Real seal route is exhausted by the source rows R, T, S, D, W, Q and the support rows H, C, P, N before L is read. Every completion-facing Real seal consumer therefore factors through the same DiagonalTailSelector, StreamName, DyadicRatCore, and CauchyLimitSeal route named by chapter 697, chapter 126, chapter 291, chapter 386, and chapter 38.

Proof. Read the accepted carrier. The Real seal row L is displayed only after the diagonal source, triangle row, seal route, dyadic ledger, stream window, and regular readback rows.

Apply source-route exactness, regular-readback exactness, and the finite source lock. Together these results identify one source route and one readback route leading to L, with no alternate dyadic ledger, window family, or selector path.

Now apply the Real seal source lock. It forces every seal-facing read through S and Q before L, so a consumer cannot read the Real seal as an independent host completion object.

Finally use completion-consumer non-escape. The packet lacks quotient streams, selected host limits, host real equality, choice data, and ambient completeness; hence the displayed source route exhausts the Real seal boundary. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility real seal route exhaustion`

Theorem 646.62 (Diagonal-limit compatibility public packet export)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the public export surface is exactly the finite carrier row K, the route ledger R, T, S, Q, L, the triangle-seal row T, S, Q, the window ledger W, the $\mathsf{Real}^{\uparrow}$ *seal row L, and the route-exhaustion rows H, C, P, N already named by the scoped closure. Every exported consumer row is read through the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *coordinates of K; there is no additional quotient stream, selected host limit, host real equality, choice datum, ambient completeness witness, hidden tail oracle, or independent route selector.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityPublicPacketExport`

Proof. Start with the carrier and NameCert rows. Definition 646.1 and Definition 646.2 expose exactly the displayed source, triangle, seal-route, dyadic-ledger, window, readback, Real-seal, transport, continuation, provenance, and local naming rows.

Next assemble the ledger surface. Definition 646.33, Definition 646.34, and Definition 646.37 identify the triangle seal, stream-window ledger, and dyadic tolerance ledger inside the same packet.

The consumer endpoint is fixed by Definition 646.38, Definition 646.53, and Definition 646.61. These force the $\mathsf{CauchyLimitSeal}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal to remain on the displayed route.

Finally use Definition 646.11 and Definition 646.48. Since the excluded analytic objects have no carrier coordinate in K, the listed rows are the entire public packet. □

Theorem 646.63 (Diagonal-limit compatibility route-ledger public exactness)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, the public route ledger is exactly the route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The rows T, D, W, Q *are not optional side ledgers: the source route, regular readback, window fusion, and Real seal exactness theorems force them to be the same rows used by every public completion-facing or* $\mathsf{Real}^{\uparrow}$ *consumer.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRouteLedgerPublicExactness`

Proof. Read source-route exactness first. Definition 646.43 places the route from R through the selector, seal, dyadic, window, readback, and Real rows, and Definition 646.44 binds Q to the same T, D, W data.

Use the ledger theorems to remove side choices. Definition 646.52 and Definition 646.34 keep the window row W fixed, while Definition 646.53 keeps L attached to that same route.

The public route then follows from Definition 646.59 and Definition 646.61. They identify the displayed ledger as the route read by public consumers and exhaust the remaining support rows H, C, P, N. □

Theorem 646.64 (Diagonal-limit compatibility downstream public boundary)**.** *Downstream* $\mathsf{RealCompleteness}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and regular-Cauchy consumers of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet may consume only the public route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

They may transport that route through the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows, but they cannot introduce quotient streams, selected limits, host equality, ambient completeness, hidden tail oracles, or independent route selectors.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityDownstreamPublicBoundary`

Proof. Begin with the public packet export, Definition 646.62. It fixes the finite carrier, route ledger, triangle seal, window ledger, Real seal, and route-exhaustion rows as one exported packet.

Attach the downstream consumers with Definition 646.35, Definition 646.38, and Definition 646.60. These theorems route completion, seal, and regular-tail reads through the same displayed packet.

Finally apply Definition 646.48 and Definition 646.42. The excluded objects are not part of the exported carrier or support rows, so no downstream consumer can supply them from this public boundary. □

Theorem 646.65 (DiagonalLimitCompatibility standard bridge surface)**.** *For an accepted* DiagonalLimitCompatibil *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, the public diagonal-limit compatibility packet bridges the displayed budget, selector, limit-seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows to the standard reflection schema. The bridge surface is the same finite public route*

$$R, T, S, D, W, Q, L, H, C, P, N,$$

with no quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityStandardBridgeSurface`

Proof. Start from the public route of Definition 646.64. It already exposes the budget, selector, seal, dyadic, window, readback, Real seal, transport, continuation, package, and local name rows as the only rows available to completion-facing consumers.

Synchronize the seal source with Definition 646.69 and Definition 646.50. These theorems keep the limit-seal route attached to the same diagonal source and force every Real-facing read through the displayed seal and readback rows.

Lock the source and budget rows with Definition 646.58 and Definition 646.49. The selector-budget package rows supplied by Definition 646.190 and Definition 646.191 then give the standard reflection input without adding a second selector or analytic coordinate.

Finally repack the same finite tuple as the bridge surface. The BHist, hsame, Cont, Pkg, and NameCert rows are already displayed by K, and the non-escape theorems exclude every quotient, selected-limit, host-equality, choice, and ambient-completeness row not present in that tuple. □

Theorem 646.66 (DiagonalLimitCompatibility bridge support minimality)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, every standard-reflection or completion-facing bridge read must retain the finite support rows*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Deleting D *loses the dyadic tolerance ledger, deleting* W *loses the* StreamName$^{\uparrow}$ *window ledger, deleting* Q *loses the* RegSeqRat$^{\uparrow}$ *readback row, deleting* L *loses the* Real$^{\uparrow}$ *seal, and deleting one of* H, C, P, N *loses componentwise transport, displayed route, package provenance, or local naming. No quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness row can replace any missing support row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBridgeSupportMinimality`

Proof. First lock the route that any bridge consumer may read. Definition 646.58 forces completion-facing and Real$^{\uparrow}$ consumers through the source, selector, seal, dyadic, window, readback, and Real rows before L is available. Definition 646.59 identifies this same route as the public budget route, and Definition 646.63 records it as exactly $R, T, S, D, W, Q, L, H, C, P, N$.

Next attach the bridge consumer to that route. Definition 646.64 makes downstream completion and regular-Cauchy reads factor through the same displayed packet, while Definition 646.65 repacks that packet as the standard reflection surface. Thus a standard-reflection or completion-facing bridge read has no support beyond the listed rows.

It remains to check that no listed row is eliminable. Removing D breaks the tolerance handoff used before readback; removing W breaks the stream-window ledger used by the seal and readback rows; removing Q removes the regular rational readback; removing L removes the Real seal read

by the bridge; and removing one of H, C, P, N removes the transport, route, provenance, or local naming row that makes the packet public.

Finally apply Definition 646.11, Definition 646.48, and Definition 646.42. These clauses exclude quotient streams, selected host limits, host equality, choice data, hidden tail oracles, and ambient completeness from the carrier, so none of them can supply a substitute for a deleted finite support row. □

Theorem 646.67 (Diagonal-limit compatibility budget-selector route meet)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, and let B be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet using the same* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and* $\mathsf{Real}^{\uparrow}$ *seal row L. The completion-facing route of K and the finite observation budget-selector route of B meet at the same displayed budget surface*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Every completion-facing, $\mathsf{Real}^{\uparrow}$*-facing, or selector-facing consumer that preserves those shared rows must read the same budget surface, with no quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, independent selector, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSelectorRouteMeet`

Proof. First lock the diagonal-limit route. Apply Definition 646.58 and Definition 646.59. These theorems force the accepted packet K to expose the completion-facing budget through $R, T, S, D, W, Q, L, H, C, P, N$

Next identify the public ledger. The route-ledger public exactness theorem Definition 646.63 states that the displayed budget surface is the exact public ledger for K, and Definition 646.65 repacks the same rows for bridge-facing consumers.

Now attach the selector side through chapter 782. Its accepted carrier and budget-route determinacy use the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal rows, so the selector route has no separate budget surface once it shares W, D, Q, L with K.

The two routes therefore meet at the displayed tuple. The non-escape clauses for the diagonal-limit packet and for the finite observation budget selector exclude quotient streams, selected host limits, host real equality, choice data, hidden tail oracles, independent selectors, and ambient completeness from the shared surface. □

Theorem 646.68 (Diagonal-limit compatibility budget-selector bridge-meet exactness)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, and let B be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet using the same* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and* $\mathsf{Real}^{\uparrow}$ *seal row L. Then every bridge-facing, completion-facing, or selector-facing consumer preserving those rows reads the same standard-reflection support surface*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

No quotient stream, host real equality, countable choice, hidden selector, independent selected limit, or ambient completeness coordinate refines this bridge meet.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSelectorBridgeMeetExa`

Proof. First apply the budget-selector route meet, Definition 646.67. It places the diagonal route and selector route on the same displayed budget surface whenever W, D, Q, L are shared.

Next identify the public ledger. The route-ledger public exactness theorem Definition 646.63 says that this surface is exactly $R, T, S, D, W, Q, L, H, C, P, N$ for accepted diagonal-limit consumers.

Now pass to bridge-facing consumers. The standard bridge surface theorem Definition 646.65 repacks the same rows as the standard-reflection support, and bridge support minimality Definition 646.66 prevents deletion or replacement of any listed row.

Finally use the non-escape clauses from the route meet and the bridge support theorem. Quotient streams, host real equality, countable choice, hidden selectors, selected limits, and ambient completeness are absent from both packets, so they cannot refine the common bridge meet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ packet is generated from an accepted $\mathsf{RealCompleteness}^{\uparrow}$ diagonal-limit packet, a $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ three-leg estimate, a $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal route, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal class, displayed hsame transports, Cont routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 646.1, Definition 646.2, Definition 646.11, Definition 646.33, Definition 646.34, Definition 646.35, Definition 646.37, Definition 646.38, Definition 646.39, Definition 646.41, Definition 646.42, Definition 646.43, Definition 646.44, Definition 646.48, Definition 646.49, Definition 646.50, Definition 646.51, Definition 646.52, Definition 646.53, Definition 646.61, Definition 646.62, Definition 646.65, and Definition 646.66 bind the finite carrier, route ledger, triangle seal, stream-window ledger, RegSeqRat readback, Real seal, completion boundary, route exhaustion, bridge support, and standard bridge surface to the displayed BHist, hsame, Cont, Pkg, and NameCert rows of the public packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected host limits, host real equality, choice, or ambient completeness.

Upgrade path: The remaining work is formal-axis checking of the displayed bridge route.

Theorem 646.69 (Diagonal-limit seal source synchronization)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

any seal-facing read first fixes the diagonal packet R, the triangle row T, the dyadic ledger D, the window ledger W, the regular readback row Q, and the seal row L through the displayed hsame *transports H,* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N. The read cannot inspect Q or L through a second source route, alternate window ledger, host limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealSourceSynchronization`

Proof. Project the carrier of Definition 646.1. The accepted packet exposes exactly the source rows R, T, S, D, W, Q, L together with H, C, P, N.

Use triangle selector alignment, Definition 646.6, to bind the triangle row T to the same dyadic ledger D and window ledger W already named by the diagonal packet R.

Now apply seal route compatibility, Definition 646.7. The seal route can read Q and L only after the displayed source, triangle, tolerance, and window rows have been fixed.

Finally apply non-escape, Definition 646.11. Since no second source route, alternate window ledger, host limit, host equality, choice datum, or ambient completeness row is exported, the seal-facing read is synchronized to the displayed packet rows. □

Theorem 646.70 (Diagonal-limit triangle readback lock)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is locked behind the same triangle row* T*, dyadic ledger* D*, and window ledger* W *that support the seal route. After* T, D, W *have been displayed,* Q *cannot be selected from a second regular readback route or from a tolerance row outside the packet.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility triangle readback lock`

Proof. Start with window ledger exactness, Definition 646.34. The row W is the exact shared-window ledger read by the diagonal source, triangle row, dyadic ledger, seal route, readback row, and real seal row.

Next use the triangle seal route, Definition 646.33. It places the seal route to Q behind the displayed T, D, W rows.

Apply regular readback exactness, Definition 646.44. That theorem identifies Q as the readback determined by the same T, D, W route and excludes another regular readback or dyadic witness.

The non-escape boundary then removes any remaining hidden source. The packet has no coordinate from which a second readback route or alternate tolerance surface could be selected. □

Theorem 646.71 (Diagonal-limit real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet factors through*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The consumer receives the displayed diagonal source, triangle row, seal route, dyadic ledger, window ledger, regular readback, real seal row, componentwise transports, continuations, provenance, and local naming row. It receives no quotient stream, selected host limit, host real equality, choice datum, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility real seal factorization`

Proof. First read the seal source synchronization theorem, Definition 646.69. It fixes the source, triangle, dyadic, window, readback, seal, transport, route, provenance, and naming rows before a seal-facing consumer can proceed.

Then apply the triangle readback lock, Definition 646.70. The regular readback row Q is tied to the same T, D, W route as the seal row, so a $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer has no independent readback surface.

Finally use completion consumer nonescape, Definition 646.48. Every completion-facing read remains inside the displayed packet rows and exports none of the omitted quotient, host, choice, or completeness data. □

Theorem 646.72 (Diagonal-limit Real seal consumer scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

first reads the synchronized source rows R, T, D, W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the real seal row* L *through* H, C, P, N*. Such a consumer cannot obtain a host limit, quotient stream, choice datum, ambient completeness row, or second source route.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility real seal consumer scope`

Proof. Apply Definition 646.69. It fixes the displayed source rows R, T, D, W, the readback row Q, the seal row L, and the structural rows H, C, P, N before any seal-facing read can continue.

Next use Definition 646.70. The readback row Q is locked behind the same triangle, dyadic, and window rows T, D, W, so the consumer has no independent readback or tolerance route.

Finally apply Definition 646.71. The factorization routes every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer through the displayed packet rows and exports none of the host limit, quotient stream, choice, ambient completeness, or second-source data excluded in the statement. □

Theorem 646.73 (Diagonal-limit seal source scope row)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

exposes a seal-facing $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer only after the finite budget-selector route, synchronized triangle row, dyadic ledger, stream window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Real seal, transports, continuations, provenance, and* $\mathsf{NameCert}$ *rows have been fixed. Thus the consumer reads only the displayed rows* $R, T, S, D, W, Q, L, H, C, P, N$ *and cannot obtain a second source route, quotient stream, host limit, choice datum, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility seal source scope row`

Proof. Project the carrier of Definition 646.1. The accepted packet displays the finite route rows R, T, S, D, W, Q, L and the structural rows H, C, P, N before any seal-facing consumer can be addressed.

Apply seal source synchronization, Definition 646.69. It fixes the source route, triangle row, dyadic ledger, stream window, readback row, seal row, transport, continuation, provenance, and local naming rows as one packet.

Use the triangle readback lock, Definition 646.70. The readback row Q is tied to the same T, D, W route as the seal row L, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer has no separate tolerance or regular-sequence source.

Finally apply the Real seal consumer scope theorem, Definition 646.72. It routes the consumer through the displayed rows and excludes second source routes, quotient streams, host limits, choice data, and ambient completeness. □

Theorem 646.74 (Diagonal-limit seal source determinacy)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. Suppose their displayed rows agree componentwise by* hsame*:*

$$\mathsf{hsame} RR', \quad \mathsf{hsame} TT', \quad \mathsf{hsame} SS', \quad \mathsf{hsame} DD', \quad \mathsf{hsame} WW', \quad \mathsf{hsame} QQ', \quad \mathsf{hsame} LL',$$
$$\mathsf{hsame} HH', \quad \mathsf{hsame} CC', \quad \mathsf{hsame} PP', \quad \mathsf{hsame} NN'.$$

If two $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumers both pass through Definition 646.73, then they determine the same seal-facing finite route. In particular, they cannot be distinguished by a host limit, quotient stream, choice datum, ambient completeness principle, or second source route.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealSourceDeterminacy

Proof. Apply seal source synchronization, Definition 646.69, to both accepted packets. Each seal-facing read is first confined to the displayed rows $R, T, S, D, W, Q, L, H, C, P, N$ and $R', T', S', D', W', Q', L', H', C',$

Use the triangle readback lock, Definition 646.70. The readback coordinate is tied to the same triangle, dyadic, and window rows as the seal coordinate, so the componentwise hsame hypotheses identify the finite route through T, D, W, Q, L on both sides.

Next apply real seal factorization, Definition 646.71, and real seal consumer scope, Definition 646.72. The consumer route factors through exactly the displayed coordinates and exports no host limit, quotient stream, choice datum, ambient completeness row, or second source route.

It remains only to exhaust the coordinates. The hypotheses identify every displayed source, structural, provenance, and naming row by hsame, while the factorization and scope theorems leave no hidden coordinate outside those rows. Thus the two consumers determine the same seal-facing finite route and admit no remaining distinguishing datum of the kinds named in the statement. □

Theorem 646.75 (Diagonal-limit four-face budget ownership)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted DiagonalLimitCompatibility$^{\uparrow}$ *packets whose displayed rows agree componentwise by* hsame*. Any* Real$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *consumer that passes through the synchronized seal source of K and K' decomposes its finite budget into exactly four owned faces. The* StreamName$^{\uparrow}$ *face owns the finite window/source row S and window ledger W; the* DyadicRatCore$^{\uparrow}$ *face owns the dyadic tolerance ledger D; the* RegSeqRat$^{\uparrow}$ *face owns the readback row Q behind the synchronized T, D, W route; and the* Real$^{\uparrow}$ *face owns the seal row L. The support rows H, C, P, N are shared BEDC support rows for* hsame*,* Cont*,* Pkg*, and* NameCert*, not a fifth mathematical face.*

Consequently the finite budget carries no quotient stream equality, host sequence equality, selected limit, ambient Real$^{\uparrow}$ *completeness theorem, metric-completeness datum, countable-choice datum, or standard bridge row. The same four faces are the Phase Real and Completion boundary faces named by Definition 126.500.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier four face budget ownership

Proof. First apply seal source synchronization, Definition 646.69. It confines every seal-facing read to the displayed rows $R, T, S, D, W, Q, L, H, C, P, N$ and their componentwise hsame partners in K'.

Next use the triangle readback lock, Definition 646.70. The readback row Q is available only behind the same synchronized T, D, W route, so it cannot supply an independent tolerance ledger, regular source, or host sequence equality.

Apply real seal factorization and real seal consumer scope, Definition 646.71 and Definition 646.72. These identify the seal consumer with the displayed finite packet and exclude quotient stream equality, selected limits, ambient completeness, choice data, and any second source route. Then apply seal source determinacy, Definition 646.74, to see that the componentwise hsame agreement leaves no additional coordinate on which two consumers can differ.

It remains to name the ownership boundary. The row S together with W is the StreamName finite window/source face, D is the DyadicRatCore tolerance ledger, Q is the RegSeqRat readback

face behind T, D, W, and L is the Real seal face. The rows H, C, P, N only transport, continue, package, and certify these four faces. Finally Definition 126.500 identifies exactly the same four faces with the current Phase Real and Completion exit route, so the diagonal seal packet contributes no fifth mathematical face or standard bridge row. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility four face budget ownership`

Theorem 646.76 (Diagonal-limit four-face budget determinacy)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *window/source face, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the* $\mathsf{Real}^{\uparrow}$ *seal face, and the support rows* H, C, P, N *agree by* hsame*:*

$$\mathsf{hsame}SS', \quad \mathsf{hsame}WW', \quad \mathsf{hsame}DD', \quad \mathsf{hsame}QQ', \quad \mathsf{hsame}LL',$$
$$\mathsf{hsame}HH', \quad \mathsf{hsame}CC', \quad \mathsf{hsame}PP', \quad \mathsf{hsame}NN'.$$

Then every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer that passes through the two accepted packets reads the same finite seal route. The route has exactly the four faces owned by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with* H, C, P, N *as support rows, and it has no second source route, quotient stream, host limit, choice datum, ambient completeness row, metric completion row, or standard bridge row.*

Proof. Start with seal source synchronization, Definition 646.69. It places each seal-facing read behind the displayed diagonal source, triangle, dyadic, window, readback, seal, transport, continuation, provenance, and naming rows.

Use the triangle readback lock, Definition 646.70. The readback row Q cannot move independently of the synchronized triangle, dyadic, and window rows, so the hsame agreement on D, W, Q fixes the tolerance and readback part of the route seen by either consumer.

Next apply real seal factorization, Definition 646.71, and real seal consumer scope, Definition 646.72. These results force the $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer to pass through the displayed seal row L and prevent any hidden host limit, quotient stream, choice datum, ambient completeness row, or second source route from entering the consumer route.

Now invoke seal source determinacy, Definition 646.74. Together with the stated hsame agreement on $S, W, D, Q, L, H, C, P, N$, it identifies all coordinates that the consumer can read after synchronization and factorization.

Finally use four-face budget ownership, Definition 646.75. The only owned mathematical faces are the $\mathsf{StreamName}^{\uparrow}$ window/source face, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the $\mathsf{Real}^{\uparrow}$ seal face, while H, C, P, N only support transport, continuation, packaging, and naming. Hence both consumers read the same finite seal route and no additional completion, bridge, choice, quotient, or host row remains. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier four face budget determinacy`

Theorem 646.77 (Diagonal-limit four-face pullback exactness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets whose displayed rows agree componentwise by* hsame*. A* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer classifies through the diagonal seal source of* K *and* K' *if and only if its four projections classify componentwise on the* $\mathsf{StreamName}^{\uparrow}$ *window/source face* (S, W)*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance face* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face* Q *behind the synchronized* T, D, W *route, and the* $\mathsf{Real}^{\uparrow}$ *seal face* L*.*

The only gluing data are the displayed support rows H, C, P, N *and* H', C', P', N'*, read respectively as* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support. The exact pullback exports no quotient stream, host equality, selected limit, ambient completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row.*

Proof. For the forward direction, start with a consumer that classifies through the diagonal seal source. Apply seal source synchronization, Definition 646.69, to confine the read to the displayed rows of K and K'.

Use the triangle readback lock, Definition 646.70, to keep the readback coordinate Q behind the same synchronized T, D, W route. Then apply real seal factorization, Definition 646.71, so the $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reads the seal row only after the displayed source, tolerance, window, and readback rows have been fixed.

Now project the synchronized packet to its four faces. Four-face budget ownership, Definition 646.75, names exactly the $\mathsf{StreamName}^{\uparrow}$ window/source face, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the $\mathsf{Real}^{\uparrow}$ seal face. The componentwise hsame hypotheses therefore give componentwise classification of those four projections, with H, C, P, N and H', C', P', N' as the only support rows.

For the reverse direction, assume the four projected faces classify componentwise through the displayed support rows. Apply Definition 301.58 to repack the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces as one finite terminal pullback surface.

Next apply Definition 38.413. The compatible open-phase faces determine a single consumer-visible seal-source handoff, and the comparison uses only hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Finally use seal source determinacy, Definition 646.74. The componentwise agreement of the accepted packets leaves no hidden coordinate outside the displayed finite route, so the repacked four-face surface is exactly the diagonal seal source classification for K and K'.

The exclusion clauses come from the same cited pullback and factorization results together with four-face budget ownership. None of the accepted rows contains quotient stream equality, host equality, selected limits, ambient completeness, metric-completeness data, countable-choice rows, or standard bridge rows, so none can be part of the exact pullback. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility four face pullback exactness`

Theorem 646.78 (Diagonal-limit tail-seal source uniqueness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any terminal* $\mathsf{Real}^{\uparrow}$ *seal read using the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row* W *and source row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the*

$\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route has a unique displayed tail-seal source up to componentwise* hsame *through the support rows H, C, P, N.*

Equivalently, two terminal seal reads whose $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *faces are accepted by the same packet determine the same consumer-visible source route after transport through H, replay through C, packaging through P, and local naming through N. The uniqueness statement exports no quotient stream, host real equality, hidden selector, selected host limit, detached completion object, or ambient completeness principle.*

Proof. Start with two terminal $\mathsf{Real}^{\uparrow}$ seal reads of the accepted packet. Seal source synchronization, Definition 646.69, places both reads behind the displayed $R, T, S, D, W, Q, L, H, C, P, N$ rows.

The triangle readback lock, Definition 646.70, fixes Q behind the same T, D, W route. Hence neither read can choose a separate regular readback, tolerance witness, or window source while still being a read of K.

Next use real seal factorization, Definition 646.71. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer reaches the terminal seal only after the displayed source, window, tolerance, readback, and seal rows have already been fixed.

Four-face budget determinacy, Definition 646.76, identifies the consumer-visible mathematical faces as the $\mathsf{StreamName}^{\uparrow}$ window/source face, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the $\mathsf{Real}^{\uparrow}$ seal face. The support rows H, C, P, N only transport, replay, package, and name those same faces.

Finally apply four-face pullback exactness, Definition 646.77. The four accepted faces repack as one terminal pullback surface, and seal source determinacy identifies the resulting source route. Thus any two terminal seal reads have the same displayed tail-seal source up to componentwise hsame through H, C, P, N.

The refusal clauses are inherited from the cited synchronization, factorization, budget, and pullback results. Since no accepted row contains a quotient stream, host real equality, hidden selector, selected host limit, detached completion object, or ambient completeness theorem, none can be used to produce another terminal tail-seal source. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTailSealSourceUniqueness`

Theorem 646.79 (Diagonal-limit compatibility root budget source)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reads its root budget source through the displayed finite rows*

$$\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The budget source is the same source exposed to $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *consumers: the diagonal row, dyadic ledger, stream window, regular-rational handoff, seal route, componentwise transport, route table, provenance, and local name must already be visible before the compatibility packet is accepted.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget source`

Proof. Project the accepted compatibility carrier. The displayed packet contains the diagonal source, triangle estimate, seal route, dyadic ledger, stream window, regular readback, Real seal row, transports, continuations, provenance, and local naming row.

Apply the NameCert obligation theorem and the window-ledger exactness theorem. They force the budget-facing reads to use the same carrier, hsame rows, Cont routes, Pkg provenance, and $\mathsf{NameCert}$ row rather than an undisplayed budget source.

Finally use completion-boundary and non-escape. Since no host limit, quotient stream, choice datum, host real equality, or ambient completeness coordinate is exported, the accepted packet can source its budget only through the displayed finite rows shared with the selector-budget route. □

Theorem 646.80 (Diagonal-limit compatibility terminal modulus-budget comparison)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *terminal route be consumed both by a* UniformCauchyCriterion$^{\uparrow}$ *route and by a* CauchyModulusRefinement$^{\uparrow}$ *selector route. Then the terminal budget comparison factors through the same displayed rows*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The two consumers therefore agree on the finite StreamName *window, dyadic ledger, regular-Cauchy readback,* Real$^{\uparrow}$ *seal handoff, route, provenance, and local* NameCert *rows. The comparison introduces no quotient stream, selected host limit, choice datum, host real equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier terminal modulus budget comparison`

Proof. First bind the diagonal terminal route. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These rows expose the finite stream window, dyadic ledger, diagonal readback, Real seal handoff, consumer route, provenance, and local NameCert data before any terminal consumer acts.

Next route the uniform consumer through Definition 646.95. The uniform finite-family seal can read the terminal prefix only after the displayed diagonal budget rows are visible, so its window and ledger data are not separate host-level selections.

Now route the selector consumer through Definition 646.91. The Cauchy-modulus-refinement selector reads the same diagonal prefix before it reaches its selector rows, while Definition 885.22 and Definition 885.23 keep the regular-Cauchy readback and Real$^{\uparrow}$ handoff tied to the displayed finite window.

Finally compare the two consumers using Definition 646.96. Both branches have already entered through $R, T, S, D, W, Q, L, H, C, P, N$, so the terminal budget comparison has no coordinate from which to extract a quotient stream, selected host limit, choice datum, host real equality, or ambient completeness claim. □

Theorem 646.81 (Diagonal-limit compatibility root window route)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet, the* StreamName$^{\uparrow}$ *finite-window route is selected before any* RegSeqRat$^{\uparrow}$ *readback or* Real$^{\uparrow}$ *seal handoff is available. A downstream consumer reaches the regular-rational row and the Real-facing seal only after reading the displayed window row, dyadic ledger, transport rows, continuation rows, provenance row, and local naming row.*

Proof. Start with the carrier projection. The window row W and dyadic ledger D are displayed before the readback row Q and seal row L are consumed.

Use window-ledger exactness together with the triangle-seal route theorem. These results place the route to Q, L behind the same StreamName$^{\uparrow}$ window row used by the diagonal source, triangle row, dyadic ledger, and seal route.

Apply completion boundary. It adds the componentwise hsame transport, displayed Cont route, Pkg provenance, and local NameCert rows to the same finite-window route, so no RegSeqRat or Real handoff precedes the window selection. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root window route`

Theorem 646.82 (Diagonal-limit compatibility root seal boundary)**.** *Every Real-facing read of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet passes through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and seal rows before it reaches the* $\mathsf{Real}^{\uparrow}$ *coordinate. The read is a route through* $R, T, S, D, W, Q, L, H, C, P, N$*, not a read through host real equality, a selected host limit, quotient stream equality, choice data, or ambient completeness.*

Proof. Read the packet through seal-route compatibility. The seal route can reach the regular readback and Real seal only after the diagonal source, triangle row, dyadic ledger, and stream window have been fixed.

Now apply Real-seal exactness. It identifies the displayed route through $R, T, S, D, W, Q, H, C, P, N$ as the boundary through which L is reached.

The Real-seal non-escape theorem removes the possible host-side bypasses. Thus the Real-facing read is exactly a finite regular-readback and seal-route read inside the accepted compatibility packet. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root seal boundary`

Theorem 646.83 (Diagonal-limit compatibility root consumer non-escape)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, or completion-seal consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet can inspect only the displayed budget, window, diagonal, seal, transport, route, provenance, and local-name rows. In packet notation the consumer-visible surface is exhausted by*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Proof. Project the accepted packet and list the consumer-facing coordinates. The budget and window information is carried by the dyadic ledger and stream window, the diagonal and seal information by the source, triangle, seal, readback, and Real rows, and the structural information by the transport, route, provenance, and local-name rows.

Use the completion-consumer non-escape theorem. It already requires every completion-facing or $\mathsf{Real}^{\uparrow}$ consumer to factor through the same listed rows.

Finally replay the root budget source, root window route, and root seal boundary results. Together they show that the consumer has no coordinate outside the displayed budget, window, diagonal, seal, transport, route, provenance, and local-name surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root consumer nonescape`

Theorem 646.84 (Diagonal-limit compatibility root budget-seal factorization)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root consumer reaches a* $\mathsf{Real}^{\uparrow}$ *or completion-seal read only after the displayed budget source,* $\mathsf{StreamName}^{\uparrow}$ *finite-window route,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal rows, transports, continuations, provenance, and local* $\mathsf{NameCert}$ *rows have been read. In packet notation the consumer route is*

$$R, T, S, D, W, Q, L, H, C, P, N,$$

and it excludes host real equality, selected host limits, quotient stream equality, choice data, and ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget seal factorization`

Proof. Start with the root budget source. Definition 646.79 identifies the budget-facing source with the displayed diagonal row, dyadic ledger, stream window, regular-rational handoff, Real seal route, transports, continuations, provenance, and local name.

Next apply the root window route. Definition 646.81 forces the $\mathsf{StreamName}^{\uparrow}$ finite-window row to be read before the $\mathsf{RegSeqRat}^{\uparrow}$ readback or the $\mathsf{Real}^{\uparrow}$ seal handoff is available.

Then use the root seal boundary. Definition 646.82 places each Real-facing read behind the regular readback and seal rows Q, L on the displayed route through $R, T, S, D, W, Q, L, H, C, P, N$.

Finally apply root consumer non-escape. Definition 646.83 exhausts downstream Real, completion, and completion-seal consumers by the displayed packet rows, so none of the excluded host or quotient objects has a route into the accepted consumer surface. □

Theorem 646.85 (Diagonal-limit compatibility root provenance exhaustion)**.** *Every root budget, window, readback, seal, and consumer route in an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is exhausted by the displayed rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

together with the Pkg *provenance and the local* $\mathsf{NameCert}$ *row. No host limit, quotient stream equality, choice datum, host real equality, or ambient completeness coordinate is added to the accepted root-budget surface.*

Proof. First use root budget source, root window route, and root seal boundary. These three route theorems identify the budget source, the $\mathsf{StreamName}^{\uparrow}$ finite-window route, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal boundary with the same displayed packet rows.

Next apply root consumer non-escape. It says that every downstream Real, completion, or completion-seal consumer is confined to the visible budget, window, diagonal, seal, transport, route, provenance, and local-name rows.

Finally use root budget-seal factorization. The factorization composes the preceding routes into the consumer-facing tuple $R, T, S, D, W, Q, L, H, C, P, N$, and the tuple already contains the Pkg provenance row and local $\mathsf{NameCert}$ row as P, N. Therefore the accepted packet has no remaining route through host limits, quotient stream equality, choice data, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root provenance exhaustion`

Theorem 646.86 (Diagonal-limit compatibility root-window selector exhaustion)**.** *Every root finite-window selector read of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is exhausted by the displayed rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

before any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-seal consumer can read the route. The selector cannot add quotient stream equality, selected host limits, hidden selector data, host real equality, choice data, or ambient completeness.*

Proof. Begin with the root budget source and root window route, Definition 646.79 and Definition 646.81. These identify the budget source, dyadic ledger, stream window, diagonal row, window row, and regular handoff as the finite selector-facing packet.

Next apply the root seal boundary and root consumer non-escape, Definition 646.82 and Definition 646.83. The seal boundary keeps $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion-facing reads behind the displayed Q, L, H, C, P, N rows, while non-escape prevents a branch to host limits, quotient streams, selected host data, choice data, host equality, or ambient completeness.

Finally use budget-seal factorization and root provenance exhaustion, Definition 646.84 and Definition 646.85. Their composite exhausts the selector route by $R, T, S, D, W, Q, L, H, C, P, N$ together with the displayed provenance and local naming rows. Since the accepted packet has no hidden selector coordinate, every root finite-window selector read is consumed before the downstream regular, Real, or completion-seal route is available. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root window selector exhaustion`

Theorem 646.87 (Diagonal-limit compatibility selector-budget common refinement)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose terminal route is consumed by a* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *route, a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *selector route, and a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal route factors all three reads through the same displayed finite rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

before any selector or seal consumer can inspect the route. The shared route first exposes the diagonal root budget, stream-name window, regular-rational readback, dyadic ledger, Real seal boundary, componentwise transport, continuation replay, package provenance, and local $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. It admits no hidden selector budget, no quotient stream equality, no selected host limit, no host real equality, no choice datum, and no ambient completeness row.*

Proof. Start with the root budget source, root window route, and root seal boundary. Theorems 646.79, 646.81, and 646.82 identify the diagonal source, $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, dyadic ledger, and $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal boundary as one displayed packet.

Next attach the selector-facing consumers. The root-window selector exhaustion theorem, Definition 646.86, forces every selector-budget read to pass through the same $R, T, S, D, W, Q, L, H, C, P, N$ rows before any regular, Real, or completion-seal route is visible.

Finally close the route against escape. Root consumer non-escape, budget-seal factorization, and root provenance exhaustion, Definition 646.83, Definition 646.84, and Definition 646.85, keep Pkg provenance and the local $\mathsf{NameCert}$ row attached to the same carrier. Thus the $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumers share the common refinement before any selector or seal consumer inspects the route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget common refinement`

Theorem 646.88 (DiagonalLimitCompatibility finite-tail-filter handoff)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$F = (S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted FiniteTailFilter$^{\uparrow}$ *packet whose stream-window, dyadic, regular-readback, and Real-seal rows agree with the displayed rows of* K *by* hsame(S_F, W), hsame(D_F, D), hsame(R_F, Q), *and* hsame(E_F, L).

Then every finite-tail-filter consumer of F *reads the same* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Cont, Pkg, *and* NameCert *route supplied by* K *before any completion-facing read. The handoff introduces no hidden tail selector, quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget finite tail filter handoff`

Proof. First apply the selector-budget common refinement of Definition 646.87. It places the diagonal root budget, stream-name window, regular-rational readback, dyadic ledger, Real seal boundary, continuation replay, package provenance, and local naming row on the single displayed route $R, T, S, D, W, Q, L, H, C, P, N$.

Next read the finite-tail-filter carrier through the four shared rows. The hsame identifications force the FiniteTailFilter$^{\uparrow}$ window, dyadic, readback, and Real-seal faces to be W, D, Q, L as seen by the diagonal packet, while the carrier obligations keep the tail-budget, comparison, transport, continuation, provenance, and local NameCert rows inside the same finite packet.

Finally use the finite-tail-filter non-escape and Real-seal handoff rows. They make every Real-facing or completion-facing consumer pass through the finite packet before it can read the completion boundary. Since the common refinement already fixes the corresponding diagonal route and its Pkg and NameCert rows, the composite handoff has no coordinate left for a hidden tail selector, quotient stream, selected host limit, host real equality, choice datum, or ambient completeness. □

Theorem 646.89 (DiagonalLimitCompatibility CauchyTailWitnessEnvelope factorization)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet have a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *route consumed by an accepted* CauchyTailWitnessEnvelope$^{\uparrow}$ *packet. Then the terminal consumer route factors first through the displayed diagonal budget rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

and then through the envelope rows

$$S, F, Q, A, R, H, C, P, N.$$

The composite route introduces no quotient stream, selected host limit, host equality, choice datum, hidden tail oracle, or ambient completeness row.

Proof. Begin with the diagonal budget side. The root budget source, root window route, and root seal boundary of Definition 646.79, Definition 646.81, and Definition 646.82 place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ read behind the displayed rows $R, T, S, D, W, Q, L, H, C, P, N$.

Next use the consumer and provenance boundary. By Definition 646.83, Definition 646.84, and Definition 646.85, no downstream consumer can leave the diagonal budget packet for a host limit, quotient stream, choice datum, host equality, or ambient completeness coordinate.

Now attach the Cauchy tail witness envelope. The selector handoff of Definition 898.3 sends the consumed diagonal-tail route through the selected window S, finite-window envelope F, and tail-comparison row Q. The exactness results of Definition 898.5 and Definition 898.6 keep the selected window and endpoint ledger on the same finite envelope surface.

Finally apply the Real-tail seal boundary and non-escape results of Definition 898.7 and Definition 898.4. They force the envelope read to pass through $S, F, Q, A, R, H, C, P, N$ and exclude hidden tail oracles together with the same host, quotient, choice, and ambient-completeness bypasses. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier cauchy tail envelope factorization`

Theorem 646.90 (Diagonal-limit selector-budget MonotoneCauchy handoff)**.** *Let $K = (R, T, S, D, W, Q, L, H, C,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, and let an accepted* $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet read its finite-window, modulus, located-interval, regular-readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows through the same W, D, Q, L selector-budget surface of K. Then every* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{Completion}^{\uparrow}$*-facing monotone-Cauchy consumer factors through the displayed diagonal selector-budget route before it reads the monotone ledger or located upper-bound row.*

The handoff exports no quotient stream, selected host limit, host real equality, trichotomy row, ambient completeness coordinate, metric-completeness datum, bridge row, or choice datum.

Proof. First lock the diagonal selector source. Apply Definition 646.193 and Definition 646.198. These identify the only selector-budget route of K as the displayed diagonal source together with the rows W, D, Q, L, H, C, P, N.

Next attach the terminal diagonal consumer. The Cauchy-tail envelope factorization, Definition 646.89, shows that any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read first passes through $R, T, S, D, W, Q, L, H, C, P, N$ before a downstream tail-facing packet can inspect its own rows.

Now read the monotone-Cauchy packet through the same selector surface. By Definition 438.27, the finite-window, modulus, located-interval, regular-readback, and $\mathsf{Real}^{\uparrow}$ seal rows of the accepted monotone packet are read as the cofinal criterion seal surface. Since those rows are assumed to be the W, D, Q, L selector-budget surface of K, the monotone ledger and located upper-bound row occur only after the diagonal selector route has already been fixed.

Finally apply the diagonal completion-consumer non-escape boundary, Definition 646.48. It removes every route through quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, metric-completeness data, bridge rows, and choice, so the monotone-Cauchy consumer receives only the displayed selector-budget handoff. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetMonotoneCauch`

Theorem 646.91 (DiagonalLimitCompatibility CauchyModulusRefinement route)**.** *Let an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet have a terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *route consumed through an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Then the terminal consumer route factors first through the displayed diagonal rows*

$$R, T, S, D, W, Q, L, H, C, P, N$$

and then through the refinement rows

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The composite route exports no quotient stream, selected host limit, hidden selector, second meet row, host real equality, choice datum, or ambient completeness coordinate.

Proof. Start with the diagonal root budget. The root budget source, root window route, and root seal boundary of Definition 646.79, Definition 646.81, and Definition 646.82 place every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read behind $R, T, S, D, W, Q, L, H, C, P, N$.

Next use the diagonal consumer boundary. By Definition 646.83, Definition 646.84, and Definition 646.85, the terminal read cannot leave the displayed diagonal route for a quotient stream, selected host limit, host real equality, choice datum, or ambient completeness coordinate.

Now pass the terminal route into the refinement packet. The source-window lock of Definition 885.110 forces the consumed route to expose M_0, M_1, U, V before any selected-window or tolerance-readback row is available.

Finally apply root window exhaustion and Real-seal source conservation, Definition 885.113 and Definition 885.116. These results place the refinement consumer on $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ and exclude quotient streams, selected host limits, hidden selectors, second meet rows, host real equality, choice data, ambient completeness coordinates, and detached completion witnesses. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier cauchy modulus refinement route`

Theorem 646.92 (Diagonal-limit compatibility shared terminal consumer comparison)**.** *Let an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet have a terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer routed through either an accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet or an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. In both cases the terminal consumer first reads the same diagonal budget, window, readback, seal, transport, route, provenance, and local-name boundary*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Only after this shared boundary do the two routes differ: the $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *route continues through its envelope rows, while the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route continues through its refinement rows. The comparison introduces no quotient stream equality, selected host limit, hidden tail oracle, hidden selector, host real equality, choice datum, or ambient completeness coordinate.*

Proof. First form the shared diagonal boundary. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These force every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read to pass through the diagonal budget, stream window, regular readback, Real seal, transport, continuation, package, and local-name rows.

Next close that boundary against escape. Use Definition 646.83, Definition 646.84, and Definition 646.85. The reusable consumer-facing tuple is therefore exactly $R, T, S, D, W, Q, L, H, C, P, N$, with no route to host limits, quotient stream equality, choice data, host real equality, or ambient completeness.

Now compare the two terminal consumers. The $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ case follows from Definition 646.89, which begins with the same diagonal tuple before entering the envelope rows. The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ case follows from Definition 646.91, which begins with that same tuple before entering the refinement rows. Thus the routes differ only after the shared diagonal terminal-consumer boundary has been read. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier shared terminal consumer comparison`

Theorem 646.93 (DiagonalLimitCompatibility envelope-refinement pullback)**.** *Let an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet have a terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer routed through*

either an accepted CauchyTailWitnessEnvelope$^{\uparrow}$ *packet or an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet. Both terminal-consumer routes pull back to the same displayed diagonal boundary*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The envelope-specific rows $S, F, Q, A, R, H, C, P, N$ *and the refinement-specific rows* $M_0, M_1, U, V, T, W, Q, E, H, C,$ *are read only after that boundary. Neither branch introduces quotient streams, selected host limits, hidden tail oracles, hidden selectors, host real equality, choice data, or ambient completeness.*

Proof. First read the root budget source, root window, and root seal boundary of Definition 646.79, Definition 646.81, and Definition 646.82. These place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer behind $R, T, S, D, W, Q, L, H, C, P, N$ before any downstream packet can expose its own rows.

Next close that boundary under root consumer non-escape. Apply Definition 646.83; the terminal read cannot leave the displayed diagonal packet for a quotient stream, selected host limit, hidden selector, host real equality, choice datum, or ambient completeness coordinate.

Now attach the two branch factorizations. The envelope branch is Definition 646.89, which reads $S, F, Q, A, R, H, C, P, N$ only after the same diagonal boundary. The refinement branch is Definition 646.91, which reads $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ only after that boundary.

Finally use Definition 646.92. Since both branches have the same terminal-consumer prefix and all branch-specific rows occur downstream, the displayed tuple $R, T, S, D, W, Q, L, H, C, P, N$ is the classifier-visible pullback for either accepted terminal route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier envelope refinement pullback`

Theorem 646.94 (DiagonalLimitCompatibility terminal budget readback)**.** *Let an accepted* DiagonalLimitCompati *packet have a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer read, and suppose that this read is later consumed by either an accepted* CauchyTailWitnessEnvelope$^{\uparrow}$ *packet or an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet. Then the terminal read first factors through the same displayed diagonal budget prefix*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Only after this prefix may the route branch: the envelope case proceeds through

$$S, F, Q, A, R, H, C, P, N,$$

while the refinement case proceeds through

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The common readback exports no quotient stream equality, selected host limit, hidden tail oracle, hidden selector, host real equality, choice datum, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier terminal budget readback`

Proof. First build the common diagonal prefix. Apply Definition 646.79, Definition 646.81, and Definition 646.82 to place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ read behind the root budget, window, seal, transport, route, provenance, and local-name rows.

Next close the prefix against escape. Use Definition 646.83, Definition 646.84, and Definition 646.85. These identify the reusable consumer-facing tuple as $R, T, S, D, W, Q, L, H, C, P, N$

and rule out quotient stream equality, selected host limits, host real equality, choice data, and ambient completeness before any downstream branch is inspected.

Now split by the downstream consumer. If the read is consumed by CauchyTailWitnessEnvelope$^{\uparrow}$, apply Definition 646.89; the remaining route is exactly $S, F, Q, A, R, H, C, P, N$, and hidden tail oracles are excluded.

If the read is consumed by CauchyModulusRefinement$^{\uparrow}$, apply Definition 646.91; the remaining route is exactly $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, and hidden selectors are excluded. Thus both terminal branches share the same diagonal budget readback boundary before they diverge. □

Theorem 646.95 (DiagonalLimitCompatibility uniform Cauchy terminal handoff)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet have a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer read, and suppose that the read is later consumed by an accepted* UniformCauchyCriterion$^{\uparrow}$ *packet. Then the terminal read first factors through the same diagonal budget prefix*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Only after this prefix may it enter the uniform Cauchy finite-family seal rows: the finite family index ledger, shared StreamName$^{\uparrow}$ *window family, uniform* CauchyModulus$^{\uparrow}$ *schedule,* DyadicRatCore$^{\uparrow}$ *tolerance row,* DyadicTailBound$^{\uparrow}$ *tail ledger,* Real$^{\uparrow}$ *seal handoff, and the displayed* BHist/hsame/Cont/Pkg/NameCert *rows. The handoff exports no quotient stream equality, selected host limit, hidden tail selector, host real equality, choice datum, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier uniform cauchy terminal handoff`

Proof. First bind the diagonal terminal prefix. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ read behind the displayed budget source, window, readback, Real seal, transport, route, provenance, and local naming rows.

Next close that prefix against bypass. Use Definition 646.83; the terminal consumer can inspect only $R, T, S, D, W, Q, L, H, C, P, N$ before it reaches a downstream packet. Hence quotient stream equality, selected host limits, host real equality, choice data, and ambient completeness are unavailable at the handoff boundary.

Now read the accepted UniformCauchyCriterion$^{\uparrow}$ consumer through its finite family seal surface. The public export, downstream consumer boundary, and seal completion boundary of chapter 435 force the uniform criterion read to pass through the finite index ledger, shared window family, uniform modulus, dyadic tolerance, finite tail ledger, Real seal handoff, and structural BHist/hsame/Cont/Pkg/NameCert rows.

Combining the two surfaces gives the claimed order: the diagonal terminal prefix is read first, and only then can the finite-family uniform Cauchy seal rows be consumed. Since neither side exposes hidden tail selectors, quotient streams, selected host limits, host real equality, choice data, or ambient completeness, the composite handoff exports none of them. □

Theorem 646.96 (DiagonalLimitCompatibility uniform Cauchy three-way terminal comparison)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet have a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer read, and suppose that this read is routed through one of the accepted terminal consumers* CauchyTailWitnessEnvelope$^{\uparrow}$, CauchyModulusRefinement$^{\uparrow}$, *or* UniformCauchyCriterion$^{\uparrow}$. *Then all three routes first factor through the same displayed diagonal budget prefix*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Only after this common prefix may the route enter the respective terminal surface: the envelope rows for CauchyTailWitnessEnvelope$^{\uparrow}$*, the refinement rows for* CauchyModulusRefinement$^{\uparrow}$*, or the uniform finite-family seal rows for* UniformCauchyCriterion$^{\uparrow}$*. The comparison introduces no quotient stream equality, selected host limit, hidden tail oracle, hidden selector, host real equality, choice datum, or ambient completeness coordinate.*

Proof. First bind the diagonal budget prefix. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ read behind the budget source, stream window, diagonal readback, Real seal, transport, continuation, package, and local-name rows.

Next close the prefix against nonlocal exits. Use Definition 646.83, Definition 646.84, and Definition 646.85. The consumer-facing boundary is therefore exactly $R, T, S, D, W, Q, L, H, C, P, N$, and the route has no access to host limits, quotient stream equality, choice data, host real equality, or ambient completeness before any terminal consumer row is entered.

Now compare the three terminal consumers. The CauchyTailWitnessEnvelope$^{\uparrow}$ branch is Definition 646.89; the CauchyModulusRefinement$^{\uparrow}$ branch is Definition 646.91; and the UniformCauchyCriterion$^{\uparrow}$ branch is Definition 646.95. Each begins with the same diagonal prefix and only afterward enters its own envelope, refinement, or uniform finite-family rows. Thus the shared diagonal budget prefix is the common terminal-consumer boundary for the current Real$^{\uparrow}$ and Completion$^{\uparrow}$ routes. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityUniformCauchyThreeWayTermin`

Theorem 646.97 (DiagonalLimitCompatibility Real-completion terminal pullback)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet have a terminal route read by a* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer, and suppose that this read is then consumed by one of* CauchyTailWitnessEnvelope$^{\uparrow}$*,* CauchyModulusRefinement$^{\uparrow}$*, or* UniformCauchyCriterion$^{\uparrow}$*. The Real-completion-facing terminal read pulls back first to the same displayed diagonal tuple*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The branch-specific rows are downstream-only: envelope, refinement, and uniform finite-family data may be read only after the tuple above has been fixed. No quotient stream, selected host limit, host real equality, choice datum, ambient completeness coordinate, hidden tail oracle, or hidden selector is exported by the pullback boundary.

Proof. First bind the Real-completion terminal read to the root diagonal route. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These identify the budget source, finite stream window, regular-rational readback, Real seal, transport, continuation, package, provenance, and local-name rows with $R, T, S, D, W, Q, L, H, C, P, N$.

Next close the route under the consumer boundary. Use Definition 646.83 and Definition 646.85. The accepted packet has no coordinate from which a Real or completion consumer can leave the displayed diagonal tuple for a quotient stream, selected host limit, host real equality, choice datum, ambient completeness coordinate, hidden tail oracle, or hidden selector.

It remains to compare the downstream consumers. The envelope branch is Definition 646.89, the refinement branch is Definition 646.91, and the uniform finite-family branch is Definition 646.95. Each branch starts only after the same diagonal tuple has already been read.

Finally apply Definition 646.96. The three-way comparison makes the shared tuple the terminal-consumer boundary for all three downstream routes, so the Real-completion terminal read pulls back to that tuple before any branch-specific rows are available. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier real completion terminal pullback`

Theorem 646.98 (DiagonalLimitCompatibility terminal branch exhaustion)**.** *Let an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet have a terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *read consumed by the current terminal branch surface. Then the read first factors through the displayed diagonal terminal prefix*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

After this prefix the only admitted branch rows are the CauchyTailWitnessEnvelope$^{\uparrow}$ *rows, the* CauchyModulusRefinement$^{\uparrow}$ *rows, or the* UniformCauchyCriterion$^{\uparrow}$ *finite-family seal rows. The terminal surface exports no quotient stream, selected host limit, host equality, choice datum, ambient completeness coordinate, hidden tail oracle, hidden selector, or fourth terminal branch.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTerminalBranchExhaustion`

Proof. First bind the root prefix. Apply Definition 646.79, Definition 646.81, and Definition 646.82. These place the terminal Real$^{\uparrow}$ or Completion$^{\uparrow}$ read behind the budget, window, readback, seal, transport, route, provenance, and local-name rows $R, T, S, D, W, Q, L, H, C, P, N$.

Next close the prefix against nonlocal escape. Use Definition 646.83, Definition 646.84, and Definition 646.85. The accepted carrier has no coordinate for quotient streams, selected host limits, host equality, choice data, ambient completeness, hidden tail oracles, or hidden selectors before a displayed branch surface is read.

Now enumerate the admitted terminal branches. The envelope route is Definition 646.89; the refinement route is Definition 646.91; and the uniform finite-family route is Definition 646.95. Each begins only after the same diagonal prefix and then reads its own displayed branch rows.

Finally apply the three-way comparison and terminal pullback, Definition 646.96 and Definition 646.97. They make the three displayed consumers exhaustive for the current terminal surface: any accepted terminal branch is one of those rows after $R, T, S, D, W, Q, L, H, C, P, N$, and no fourth branch or hidden terminal export remains. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier terminal branch exhaustion`

Theorem 646.99 (Diagonal-limit compatibility root budget coherence)**.** *For an accepted* DiagonalLimitCompatibilit *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

the root-budget route is the same displayed route used by the rows D, W, Q, L, H, C, P, N. It reads the DyadicRatCore$^{\uparrow}$ *ledger D, the* StreamName$^{\uparrow}$ *window row W, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *seal row L, the componentwise* hsame *transports H, the displayed* Cont *routes C, the* Pkg *provenance P, and the local* NameCert *row N through the carrier of Definition 646.1. It has no root-budget source outside that* BHist/hsame/Cont/Pkg/NameCert *surface.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root budget coherence`

Proof. Project the accepted carrier. The only root-facing data in the packet are the displayed rows $R, T, S, D, W, Q, L, H, C, P, N$, with D, W, Q, L carrying the budget, window, readback, and seal coordinates and H, C, P, N carrying transport, route, provenance, and naming.

Apply the root budget source theorem, Definition 646.79. It identifies the budget source with the same finite structural surface used by the selector-budget route, rather than with a hidden host-side source.

Now use the root window route and root seal boundary theorems, Definition 646.81 and Definition 646.82. They keep the route through W, Q, L after the displayed ledger and window rows and before any Real-facing handoff.

Finally apply root consumer non-escape, Definition 646.83. Since no additional budget coordinate is exported, the root-budget route is exactly the route already used by D, W, Q, L, H, C, P, N. □

Theorem 646.100 (Diagonal-limit compatibility root selector budget exhaustion)**.** *Every selector-budget consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet factors through*

$$R, T, S, D, W, Q, L.$$

It must first read the accepted diagonal source R, the triangle selector row T, the seal route S, the dyadic ledger D, the finite window row W, the regular readback row Q, and the Real seal row L, transported by the same H, C, P, N support rows. No selector-budget consumer can introduce a second diagonal route, alternate window route, alternate dyadic ledger, selected host limit, quotient stream, choice datum, or ambient completeness input.

Proof. Start with the carrier projection. A selector-budget consumer sees one diagonal source row R, one triangle selector row T, one seal route S, one dyadic ledger D, one stream-window row W, one regular readback Q, and one Real seal row L.

Use triangle selector alignment together with source-route exactness, Definition 646.6 and Definition 646.43. These theorems attach the selector route to the same R, T, D, W rows and place the route to Q, L on that displayed source path.

Next apply the root budget carrier and classifier rows, Definition 646.111 and Definition 646.112. The carrier row supplies the finite packet surface, and the classifier row keeps componentwise hsame transport and Cont replay on that same route.

The exhaustion claim follows from root budget ledger row and completion consumer non-escape, Definition 646.113 and Definition 646.48. Together they leave no selector-budget coordinate outside $R, T, S, D, W, Q, L, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorBudgetExhaustio`

Theorem 646.101 (Diagonal-limit compatibility Real seal readback lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal read of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reaches L only after the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route S have been fixed on the displayed compatibility route. The read is therefore locked to the carrier route*

$$R, T, S, D, W, Q, L, H, C, P, N$$

and cannot be replaced by a host real equality, selected limit, quotient stream, choice datum, or ambient completeness principle.

Proof. First read the seal source synchronization theorem, Definition 646.69. It fixes R, T, S, D, W, Q, L together with H, C, P, N before a seal-facing consumer can inspect the readback or seal rows.

Next apply the triangle readback lock and regular readback exactness, Definition 646.70 and Definition 646.44. These results place Q behind the same T, D, W route used by the seal row and exclude an independent regular readback source.

Now use seal readback exactness and Real seal exactness, Definition 646.41 and Definition 646.53. They identify the route to L with the displayed seal, dyadic, window, readback, transport, continuation, provenance, and local naming rows.

Finally apply the completion consumer gate, Definition 646.149. Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read remains downstream of the same finite budget-source and seal-budget routes, so no host-side completion data can replace the locked readback route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root real seal readback lock`

Theorem 646.102 (Diagonal-limit compatibility finite-window handoff boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reaches the public handoff only through the displayed finite route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The handoff consumes the root-budget route, the selector-budget route, and the Real seal readback lock as one carrier boundary. It cannot be replaced by host real equality, a quotient stream, a selected limit, choice data, or ambient completeness.

Proof. First apply Definition 646.99. This fixes the root-budget route on the same D, W, Q, L, H, C, P, N surface that the accepted carrier exposes to downstream consumers.

Next apply Definition 646.100. The selector-budget read is exhausted by R, T, S, D, W, Q, L and the structural rows, so no second diagonal route, second window route, alternate ledger, or choice datum is available.

Finally apply Definition 646.101. The seal read reaches L only after S, D, W, Q have been fixed on the displayed compatibility route.

The non-escape clauses Definition 646.83 and Definition 646.48 remove host real equality, quotient streams, selected limits, and ambient completeness from the public handoff. Therefore the finite route $R, T, S, D, W, Q, L, H, C, P, N$ is the complete boundary for $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier finite window handoff boundary`

Theorem 646.103 (Diagonal-limit compatibility root budget public boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is confined to the single displayed route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The route simultaneously consumes the root-budget coherence, selector-budget exhaustion, Real-seal readback lock, and finite-window handoff boundary. It admits no host real equality, quotient stream, choice datum, selected limit, or ambient completeness input.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget public boundary`

Proof. Begin with root-budget coherence, Definition 646.99. It places the root-facing budget read on the same D, W, Q, L, H, C, P, N surface exposed by the accepted carrier.

Next apply selector-budget exhaustion, Definition 646.100. The selector route has already consumed R, T, S, D, W, Q, L together with the same transport, continuation, provenance, and naming rows.

Then apply the Real-seal readback lock, Definition 646.101. This locks every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read to L only after the displayed S, D, W, Q route has been fixed.

Use the finite-window handoff boundary, Definition 646.102, to package these three reads as one public consumer boundary for the route $R, T, S, D, W, Q, L, H, C, P, N$.

Finally, the non-escape clauses Definition 646.83 and Definition 646.48 close coordinate exhaustion. They leave no host real equality, quotient stream, choice datum, selected limit, or ambient completeness input outside the displayed packet route. □

Theorem 646.104 (Diagonal-limit compatibility root-budget handoff totality)**.** *Every root-budget handoff from an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet to a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer is exhausted by the ordered route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The handoff is already the finite Real and Completion route carried by the packet. It exposes no host real equality, quotient stream, selected limit, choice data, ambient completeness, or second budget source.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root budget handoff totality`

Proof. Begin with Definition 646.99. It fixes the root budget on the displayed D, W, Q, L, H, C, P, N carrier rows before any downstream consumer can read the packet.

Next apply Definition 646.100. This consumes the selector contribution on R, T, S, D, W, Q, L and leaves only the displayed transport, continuation, provenance, and naming rows available to the handoff.

Now use Definition 646.101. The Real-facing read reaches L only after the public selector and budget rows have already been fixed on the same route.

Apply Definition 646.102 and Definition 646.103. These two public-boundary theorems package the root budget, selector budget, Real seal lock, and finite-window handoff as one route for $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers.

The non-escape clauses attached to those five rows remove every remaining off-route coordinate. Hence the root-budget handoff is total for the displayed finite route, with no host real equality, quotient stream, selected limit, choice data, ambient completeness, or second budget source. □

Theorem 646.105 (Diagonal-limit compatibility budget coherence terminal lock)**.** *For every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads the root-budget handoff passes through the single ordered route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The terminal route is the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ *packet. It admits no host real equality, quotient stream, selected limit, choice datum, ambient completeness input, alternate dyadic ledger, alternate* $\mathsf{StreamName}^{\uparrow}$ *window, or second budget source.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility budget coherence terminal lock`

Proof. Begin with root-budget coherence, Definition 646.99. It places the budget read on the same D, W, Q, L, H, C, P, N rows carried by the accepted packet.

Next use selector-budget exhaustion, Definition 646.100. The selector contribution consumes R, T, S, D, W, Q, L on the same packet surface, so no alternate dyadic ledger or alternate $\mathsf{StreamName}^{\uparrow}$ window can sit between the root budget and the public selector read.

Apply the Real-seal readback lock, Definition 646.101. It forces every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ readback to reach L only through the already fixed selector, window, dyadic, and regular readback rows.

The finite-window handoff boundary Definition 646.102 and the root-budget public boundary Definition 646.103 package these rows as one public consumer route. The handoff totality theorem, Definition 646.104, then exhausts the route for every root-budget handoff.

All cited route coordinates are therefore identical:

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Since the displayed BHist/hsame/Cont/Pkg/NameCert packet has already consumed the budget, selector, window, readback, and seal rows, no host real equality, quotient stream, selected limit, choice datum, ambient completeness, alternate dyadic ledger, alternate $\mathsf{StreamName}^{\uparrow}$ window, or second budget source remains available to a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer. □

Theorem 646.106 (Diagonal-limit compatibility root budget consumer chain)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reaches the public read only through the ordered chain*

root-budget coherence → *selector-budget exhaustion* → *Real-seal readback lock* → *finite-window handoff* → *public b*

on the displayed route

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Thus the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Completion}^{\uparrow}$*-facing reads consume one ordered BEDC route rather than a host limit, quotient stream, choice datum, host real equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootBudgetConsumerChain`

Proof. First apply Definition 646.99. It fixes the root-budget read on the same D, W, Q, L, H, C, P, N surface carried by the accepted packet.

Next apply Definition 646.100. It places the selector budget behind R, T, S, D, W, Q, L and rules out a second diagonal source, alternate window route, alternate dyadic ledger, selected host limit, quotient stream, choice datum, or ambient completeness input between the root-budget read and the selector read.

Then apply Definition 646.101. The seal read reaches L only after S, D, W, Q have been fixed on the displayed compatibility route, so no host real equality can enter between the selector budget and the $\mathsf{Real}^{\uparrow}$ read.

Now use Definition 646.102. It packages the root-budget route, selector-budget route, and Real-seal lock as one finite handoff on $R, T, S, D, W, Q, L, H, C, P, N$.

Finally apply Definition 646.103. The public boundary is downstream of those four links on the same route, so every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer reaches the public read without inserting a host limit, quotient stream, choice datum, host real equality, or ambient completeness coordinate between any two links of the chain. □

Theorem 646.107 (Diagonal-limit compatibility TailMeet refinement seal route)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose root budget route enters through* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, whose refinement read is governed by* $\mathsf{CauchyModulusRefinement}^{\uparrow}$*, and whose endpoint seal is consumed by* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *factors through the single displayed L10 route*

$$R, T, S, D, W, Q, L, H, C, P, N$$

followed by the TailMeet budget-seal rows, the refinement rows, and the agreement-seal rows. The composed route exports no quotient stream, host real equality, selected limit, choice datum, side-channel completion carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root tailmeet refinement seal route`

Proof. Begin with the local diagonal packet. Apply Definition 646.106. It places every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer of the accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ carrier on the ordered route $R, T, S, D, W, Q, L, H, C, P, N$ before the public read.

Next attach the TailMeet entry and refinement handoff. Use Definition 885.156. A consumer entering through the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ root budget-seal package must consume the displayed TailMeet rows and then the refinement route before any endpoint seal is exported.

The route is unique at the refinement surface. Apply Definition 885.157. It identifies the exported $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ handoff as the same public route under the displayed hsame rows, with no side-channel completion carrier or ambient completeness coordinate.

Finally use Definition 699.79. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ endpoint is reached through the finite completion lattice after the selector, window, dyadic readback, regular readback, and agreement-seal rows have been consumed.

These four route certificates exhaust the available $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ coordinates. Since the diagonal packet, TailMeet package, refinement read, and agreement seal each expose only their displayed rows, no quotient stream, host real equality, selected limit, choice datum, side-channel completion carrier, or ambient completeness coordinate can be exported by the composed route. □

Theorem 646.108 (Root TailMeet refinement Real-seal correspondence)**.** *For every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet entering through* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, refined by* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *and read by* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*, the endpoint* $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ *consumer route is equivalent to the staged public read obtained by first consuming the displayed diagonal route*

$$R, T, S, D, W, Q, L, H, C, P, N,$$

then the TailMeet budget-seal rows, then the refinement rows, and then the agreement-seal rows. Under this correspondence the endpoint route exports no quotient stream, host real equality, selected limit, choice datum, side-channel completion carrier, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootTailmeetRefinementRealS`

Proof. For the forward direction, start from an endpoint $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ consumer of the accepted diagonal packet. Apply Definition 646.106 to place the consumer on the displayed $R, T, S, D, W, Q, L, H, C, P, N$ spine. Then use Definition 885.156 to force the TailMeet budget-seal entry and the refinement handoff before the endpoint seal can be read.

The refinement part has no alternate public route. Apply Definition 885.157; it identifies the TailMeet-to-refinement handoff with the same exported $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ read. Finally apply Definition 699.79 to finish the endpoint through the agreement-seal lattice, so the original consumer has exactly the staged diagonal, TailMeet, refinement, and agreement-seal presentation.

For the reverse direction, begin with a staged public read in that order. The diagonal root-budget chain supplies the $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ consumer spine, the route-exhaustion theorem supplies the TailMeet and refinement rows, and the uniqueness theorem prevents a second

refinement handoff from being inserted. The Real-tail agreement lattice then repacks the last rows as the endpoint $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ seal read.

Both directions use only the exposed BHist/hsame/Cont/Pkg/NameCert rows named by the four cited route theorems. Their exhaustion clauses remove the remaining possible coordinates, so the correspondence carries no quotient stream, host real equality, selected limit, choice datum, side-channel completion carrier, or ambient completeness input. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootTailMeetRefinementRealS`

Theorem 646.109 (Diagonal-limit compatibility root transport replay exactness)**.** *Any root-budget consumer entering the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *refinement route from an accepted* DiagonalLimitCompatibility *packet is transported and replayed through the same ordered rows*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The replayed route cannot split into a second diagonal selector, detached TailMeet object, host limit, quotient stream, or ambient completion coordinate.

Proof. Begin with the root budget consumer chain, Definition 646.106. It places every root-facing consumer on the displayed $R, T, S, D, W, Q, L, H, C, P, N$ route before any TailMeet refinement handoff is read.

Then apply Definition 646.108. The TailMeet refinement and Real-seal endpoint consume the same root spine and do not introduce a second diagonal selector or detached TailMeet coordinate.

The transported replay is therefore exact at the public surface. Since both cited routes expose only the listed BHist/hsame/Cont/Pkg/NameCert rows, there is no host limit, quotient stream, or ambient completion coordinate left for the replay to branch through. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root transport replay exactness`

Theorem 646.110 (Diagonal-limit compatibility root-budget public-route cover)**.** *Every public root-budget consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet reaches a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *endpoint exactly through the displayed diagonal rows*

$$R, T, S, D, W, Q, L, H, C, P, N,$$

then the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *refinement route, then the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *budget route, and finally the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *completion lattice. No public endpoint consumer can factor through host real equality, quotient streams, choice selectors, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root budget public route cover`

Proof. Start from a public root-budget consumer of the accepted diagonal packet. Apply Definition 646.106; it places the consumer on the displayed diagonal route $R, T, S, D, W, Q, L, H, C, P, N$ before any public endpoint read is available.

Next attach the TailMeet and refinement stages. The correspondence of Definition 646.108 identifies the root-budget route with the staged read through $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, and the agreement seal. The replay exactness of Definition 646.109 keeps that route on the same transported diagonal spine.

It remains to cover the public endpoints. Apply Definition 885.156 and Definition 885.157. These force every TailMeet-entering budget consumer through the displayed refinement handoff and exclude a second public route or side-channel completion carrier.

Finally use Definition 699.79. The $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ endpoint is read only after the selector, window, dyadic readback, regular readback, and agreement-seal rows of the completion lattice have been consumed.

The composed certificates exhaust the available BHist/hsame/Cont/Pkg/NameCert rows. Since none of the displayed diagonal, TailMeet, refinement, or agreement-seal rows contains host real equality, quotient streams, choice selectors, or ambient completeness, no public root-budget endpoint consumer can reach such a coordinate. □

Theorem 646.111 (Diagonal-limit compatibility root budget carrier row)**.** *The root budget carrier row of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is the finite packet surface over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *already displayed by the compatibility packet. In packet notation, the root budget route may read only the displayed diagonal source, triangle estimate, seal route, dyadic ledger, stream window, regular readback, Real seal row, componentwise transport, continuation route, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget carrier row`

Proof. Project the accepted compatibility packet. The carrier rows already list the budget-facing source, window, seal, transport, route, provenance, and local name.

Apply the root budget source theorem. Definition 646.79 identifies the budget source with the same BHist, hsame, Cont, Pkg, and NameCert surface used by the selector-budget route.

Since the carrier exports no extra budget coordinate, the root budget carrier row is exactly the displayed finite packet surface. □

Theorem 646.112 (Diagonal-limit compatibility root budget classifier row)**.** *The root budget classifier of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is stable under componentwise* hsame *transport of the same displayed packet rows and under replay of the displayed* Cont *routes. A downstream route that transports the diagonal source, dyadic ledger, stream window, regular readback, seal row, provenance, and local name by the recorded* hsame *rows reaches the same root budget surface after replaying the recorded* Cont *rows.*

Proof. Begin with the displayed transport rows. Componentwise hsame movement changes only the representation of each listed coordinate and leaves the packet inventory fixed.

Replay the displayed continuation rows. The root window route theorem Definition 646.81 places the $\mathsf{StreamName}^{\uparrow}$ window before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal handoff, so replay follows the same budget order after transport.

The classifier therefore has no second route on which to branch. It reads the transported packet through the same finite root budget surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget classifier row`

Theorem 646.113 (Diagonal-limit compatibility root budget ledger row)**.** *The root budget route of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet has exact ledger boundary and non-escape. Its source is fixed by Definition 646.79, its window order by Definition 646.81, its seal boundary by Definition 646.82, and its consumer boundary by Definition 646.83. The route also consumes the*

sibling L10 surfaces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any completion-facing handoff is available.*

Proof. First read the source and window rows. The root budget source theorem fixes the finite budget surface, and the root window route theorem orders the $\mathsf{StreamName}^{\uparrow}$ window before the regular readback and Real seal.

Next read the seal boundary. The root seal boundary theorem forces every Real-facing read through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback and seal rows.

Finally apply consumer non-escape. Definition 646.83 exhausts the downstream surface by the displayed packet rows, so no host limit, quotient stream, choice datum, host real equality, or ambient completeness coordinate can enter the root budget route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootBudget ledger row`

Theorem 646.114 (Diagonal-limit compatibility root budget triad exactness)**.** *For every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, the carrier row, classifier row, and ledger row jointly exhaust the root-budget route seen by any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer. Every such consumer read through the root budget factors through the displayed diagonal source, dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. No host limit, quotient stream, choice datum, host real equality, or ambient completeness coordinate is part of this route.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root budget triad exactness`

Proof. First read the carrier and classifier rows. The carrier row Definition 646.111 fixes the finite packet surface, and the classifier row Definition 646.112 keeps componentwise hsame transport and Cont replay on that same surface.

Next impose the ordered window and seal boundaries. The root window route theorem Definition 646.81 places the $\mathsf{StreamName}^{\uparrow}$ window before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal, while the root seal boundary theorem Definition 646.82 prevents a Real-facing read from bypassing those rows.

Finally apply the ledger and consumer boundaries. The ledger row Definition 646.113 records the exact route inventory, and consumer non-escape Definition 646.83 excludes host limits, quotient streams, choice data, host real equality, and ambient completeness. Thus every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read through the root budget factors through the displayed diagonal packet and no additional coordinate. □

Theorem 646.115 (Diagonal-limit compatibility root budget terminal route uniqueness)**.** *Let* R_0 *and* R_1 *be* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{Completion}^{\uparrow}$*-facing consumers of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose both consumers preserve the displayed diagonal source, dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. Then* R_0 *and* R_1 *determine the same root-budget terminal route through those rows. There is no second seal, quotient stream, host real equality, selected limit, or ambient completeness coordinate through which either consumer can factor a distinct terminal read.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootBudget terminal route uniqueness`

Proof. First read the root-budget carrier. By Definition 646.111, both consumers see only the displayed finite packet surface over BHist, hsame, Cont, Pkg, and the local NameCert row.

Next transport the classifier. The classifier row of Definition 646.112 says that componentwise hsame movement and replay of the displayed Cont routes keep the diagonal source, ledger, window, readback, seal, provenance, and name on the same root-budget surface.

Now apply the ledger row. Definition 646.113 orders the source, window, readback, and seal boundary and excludes any route coordinate beyond the displayed packet inventory.

Finally use triad exactness. Definition 646.114 states that every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read through the root budget factors through exactly this carrier, classifier, and ledger triad. Since R_0 and R_1 preserve the same triad rows, there is no remaining coordinate from which a second terminal route could be formed. □

Theorem 646.116 (Diagonal-limit compatibility root-budget Real completion factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet factors through the displayed diagonal source, dyadic ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. The factorization has no quotient stream, selected limit, host real equality, choice datum, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root budget real completion factorization`

Proof. First apply triad exactness. By Definition 646.114, every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read through the root budget is already exhausted by the carrier, classifier, and ledger rows over the displayed diagonal source, dyadic ledger, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, transports, routes, provenance, and local name.

Next use terminal-route uniqueness. Definition 646.115 says that two consumers preserving this same displayed packet inventory determine the same terminal route, so a consumer has no second terminal read through a quotient stream, selected limit, host real equality, choice datum, or ambient completeness coordinate.

The carrier row, classifier row, and ledger row therefore exhaust the visible factorization surface. Any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer of the accepted root-budget packet factors through exactly those displayed rows. □

Theorem 646.117 (Diagonal-limit compatibility Cauchy limit functor handoff)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet, and let a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* K *be routed through the terminal Real-completion surface of Definition 646.116. Suppose the same route also supplies an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet*

$$F = (S_0, S_1, M, W_0, W_1, Q', D', K', E, H', C', P', N')$$

whose source seal S_0 *is the displayed source seal read by the root-budget route and whose rows preserve the displayed source seal, target seal,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* RealCompletionSealClassifierUp *row, terminal* $\mathsf{Real}^{\uparrow}$ *endpoint, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}^{\uparrow}}$ *row. Then the consumer factors through the same finite terminal route already fixed by the root-budget factorization. No quotient stream, selected host limit, host real equality, choice row, detached completion object, or independent completion route remains after the functor packet is consumed.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility cauchy limit functor handoff

Proof. First apply the root-budget Real-completion factorization. Definition 646.116 places the consumer on the displayed route through the diagonal source, dyadic ledger, stream window, regular readback, Real seal, componentwise transports, continuations, provenance, and local name.

Next read the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ surface. The NameCert obligations of Definition 1116.3 force the functor packet through its source-to-target seal route, finite observation rows, completion-consumer rows, componentwise transports, continuation replay, provenance, and local name. The Real-completion consumer boundary of Definition 1116.7 prevents that packet from exporting quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable-choice selectors, or a second transport route.

Finally insert the functor packet into the terminal route. Root-budget terminal-route uniqueness from Definition 646.115 identifies every Real- or Completion-facing read that preserves the displayed packet rows, and selector-synchronizer exactness in Definition 646.121 keeps the finite observation surface on the same route. Since the functor preserves the displayed source seal and its terminal Real endpoint, it is consumed on that route rather than opening an independent completion endpoint. □

Theorem 646.118 (Diagonal-limit compatibility Cauchy functor selector admission lock)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet, and let a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* K *consume an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet along the route of Definition 646.117. Then terminal admission for that consumer is equivalent to preservation of both the displayed* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows and the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows named in Definition 646.123. In the admitted case the route is the unique terminal route of Definition 646.121. It exports no quotient stream, host real equality, selected host limit, choice row, hidden observation budget, detached completion object, or ambient completeness coordinate.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility cauchy functor selector admission lock

Proof. First consume the functor packet through Definition 646.117. The handoff keeps the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ source seal, target seal, finite observation rows, completion-consumer rows, transports, continuation replay, provenance, and local name on the finite terminal route already fixed by the root-budget factorization.

Next expose the finite selector and synchronizer gates. Apply Definition 646.119 and Definition 646.120 to place the selector rows and the synchronizer request, seal, and budget rows on the same displayed packet route.

For admission, use Definition 646.123. It identifies terminal admission exactly with joint factorization through the displayed selector rows and synchronizer rows, and it sends every request outside those two gates to the visible refusal boundary.

Finally apply Definition 646.121 together with Definition 646.122. Exactness gives the unique terminal selector-synchronizer route, while refusal exactness removes quotient streams, host real equality, selected host limits, choice rows, hidden observation budgets, detached completion objects, and ambient completeness coordinates from every non-admitted completion-facing request. □

Theorem 646.119 (Diagonal-limit compatibility carrier finite observation-budget factorization)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *carrier as in Definition 646.1. Suppose the displayed* Cont *rows factor through*

$$R \to W \to \mathit{selector} \to D \to \mathit{synchronizer} \to Q$$
$$\to \mathit{request} \to L \to \mathit{budget} \to N \to \mathit{terminal}$$

and suppose the terminal row is carried by the same Pkg *bundle as the carrier provenance. Then the carrier exposes unary histories for the selector, synchronizer, request, budget, and terminal rows, preserves each displayed continuation in the chain, and carries both the provenance package row and the terminal package row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier finite observation budget factorization`

Proof. Project the accepted carrier of Definition 646.1. Its displayed rows give unary histories for the diagonal source, dyadic ledger, window family, readback row, $\mathsf{Real}^{\uparrow}$ seal, certificate row, and provenance package.

Apply unary closure for Cont along the displayed route. The row $R \to W \to$ selector gives a unary selector history; the row from the selector through D gives a unary synchronizer history; the row from the synchronizer through Q gives a unary request history; the row from the request through L gives a unary budget history; and the row from the budget through N gives a unary terminal history.

Finally repack the data in carrier order. Keep the five continuation rows unchanged, keep the provenance Pkg row projected from the carrier, and keep the terminal Pkg row supplied at the terminal endpoint. □

Theorem 646.120 (Diagonal-limit compatibility finite observation-budget factorization)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet. Suppose a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* K *also consumes an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet and an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet on the displayed route. Then the consumer factors through the same finite packet rows: the diagonal source, dyadic ledger* D*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* L*, the finite observation-budget selector rows, the synchronizer request, seal, and budget rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. The factorization exports no quotient stream, selected host limit, host real equality, countable-choice row, hidden observation budget, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility finite observation budget factorization`

Proof. First use Definition 646.116. It places the $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer on the root-budget route through the displayed diagonal source, dyadic ledger, stream window, regular readback, Real seal, componentwise transports, continuations, provenance, and local name.

Next insert the finite observation-budget selector. The route theorem Definition 646.191 says that the selector handoff is mediated only by the displayed BHist, hsame, Cont, Pkg, and NameCert support rows, and Definition 782.6 excludes quotient streams, host real equality, countable-choice data, hidden limit selectors, and ambient completeness from the selector packet.

Now insert the seal-budget synchronizer. By Definition 646.138, the accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ request, seal, budget, selector, compatibility, transport, route, provenance, and local naming rows are read before Q and L are available to the consumer.

Finally apply Definition 646.140 and root-budget terminal route uniqueness from Definition 646.115. The synchronizer and selector are inserted on the same finite packet route, and terminal uniqueness leaves no second route through a quotient stream, selected host limit, host real equality, countable-choice row, hidden observation budget, or ambient completeness coordinate. □

Theorem 646.121 (Diagonal-limit compatibility root-budget selector synchronizer exactness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of K that preserves the displayed diagonal source, dyadic ledger D,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *seal L, the finite observation-budget selector rows, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row determines the same terminal root-budget route through the selector and synchronizer. No quotient stream, selected host limit, host real equality, choice row, hidden observation budget, or ambient completeness coordinate remains on the terminal path.*

Proof. First apply finite observation-budget factorization. Definition 646.120 places the consumer, selector, and synchronizer on the same displayed finite packet route through the diagonal source, ledger, stream window, regular readback, Real seal, transports, continuations, provenance, and local name.

Next use root-budget terminal route uniqueness. Definition 646.115 identifies all $\mathsf{Real}^{\uparrow}$- and $\mathsf{Completion}^{\uparrow}$-facing reads that preserve this displayed route, so the selector and synchronizer cannot open a second terminal path.

Finally combine triad exactness with consumer non-escape. The triad exactness theorem Definition 646.114 exhausts the carrier, classifier, and ledger rows, while Definition 646.83 excludes quotient streams, selected host limits, host real equality, choice rows, hidden observation budgets, and ambient completeness coordinates. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootBudgetSelectorSynchroni`

Theorem 646.122 (Diagonal-limit compatibility finite observation-budget refusal exactness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of K that requests quotient stream identity, a selected host limit, host real equality, a countable-choice row, a hidden observation budget, an independent seal-budget route, or ambient completeness is routed to the visible refusal boundary unless that request first factors through the finite observation-budget selector rows and the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *rows named in Definition 646.120. Thus the only admitted completion-facing route through K is the displayed*

finite route through D, W, Q, L, the selector packet, the synchronizer request, seal, and budget rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row.*

Proof. Start with the finite factorization theorem. Definition 646.120 names the admitted route for any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer that has already consumed the accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet and the accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ packet on the displayed route.

The selector route gives the first boundary. By Definition 646.191, the selector handoff reads only the visible BHist, hsame, Cont, Pkg, and NameCert support rows. The selector non-escape theorem Definition 782.6 refuses quotient streams, host real equality, countable-choice data, hidden selectors, and ambient completeness outside that finite selector surface.

The synchronizer gives the seal-budget boundary. Definition 646.138 places the synchronizer request, seal, and budget rows on the same displayed route before the regular readback Q and Real seal L are available to the consumer. Hence an independent seal-budget route has no separate entry point.

Finally apply root-budget terminal route uniqueness from Definition 646.115. Once the selector and synchronizer rows are not part of the request, the remaining requested coordinates would have to form a second terminal route through quotient identity, a selected host limit, host real equality, choice, hidden observation budget, independent seal budget, or ambient completeness. Terminal uniqueness excludes that second route, so the request is routed to the visible refusal boundary. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility finite observation budget refusal exactness`

Theorem 646.123 (Diagonal-limit compatibility selector-synchronizer admission criterion)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet. A* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of K is admitted to the terminal root-budget route exactly when its request factors through both the displayed* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows and the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows named by Definition 646.120. In that case the admitted request has the unique terminal route through D, W, Q, L, the selector packet, the synchronizer request, seal, and budget rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. Every completion-facing request that does not factor through both displayed gates is routed to the visible refusal boundary.*

Proof. First suppose the consumer is admitted to the terminal root-budget route. Finite observation-budget factorization in Definition 646.120 supplies the displayed route through the diagonal source, dyadic ledger, stream window, regular readback, Real seal, selector rows, synchronizer request, seal, and budget rows, transports, continuations, provenance, and local name.

The two gates are forced on this route. The selector route theorem Definition 646.191 and selector non-escape theorem Definition 782.6 keep the finite observation-budget component inside the displayed selector rows. The synchronizer handoff theorem Definition 646.138 then places the request, seal, and budget rows before the readback Q and Real seal L can be consumed.

Uniqueness gives the terminal route. By Definition 646.121, a request preserving the displayed selector and synchronizer rows determines the single terminal root-budget route through those rows and cannot open a second completion-facing path.

Conversely, suppose a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ request does not factor through both displayed gates. The refusal exactness theorem Definition 646.122 says that quotient stream identity, selected

host limits, host real equality, countable-choice rows, hidden observation budgets, independent seal-budget routes, and ambient completeness are visible only at the refusal boundary unless the request first passes through the selector rows and synchronizer rows named above.

Thus the admission test is exactly the joint factorization through the displayed $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ rows and $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ request, seal, and budget rows. Requests satisfying that test use the unique terminal route; all other completion-facing requests are refused. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility selector synchronizer admission criterion`

Theorem 646.124 (Diagonal-limit compatibility selector-synchronizer Real-seal scope)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet. For every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer admitted by the selector-synchronizer admission criterion, the Real seal scope of the consumer is exhausted by the dyadic ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* L*, the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. No quotient stream, selected host limit, host real equality, countable-choice row, hidden observation budget, independent seal-budget route, or ambient completeness coordinate remains in the Real seal scope.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility selector synchronizer real seal scope`

Proof. Begin with the finite observation-budget factorization. Definition 646.120 places every admitted $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer on the displayed route through D, W, Q, L, the selector rows, the synchronizer request, seal, and budget rows, transports, continuations, provenance, and local name.

Next apply selector-synchronizer exactness. Definition 646.121 identifies this displayed selector and synchronizer route as the terminal root-budget route, while Definition 646.122 routes every request outside the two displayed gates to the refusal boundary.

The admission criterion then supplies the exact gate. By Definition 646.123, an admitted consumer is precisely one whose request factors through both the finite observation-budget selector rows and the synchronizer request, seal, and budget rows.

Finally use Real-seal-budget non-escape and terminal route uniqueness. Definition 646.140 prevents the Real seal from acquiring a hidden budget coordinate, and Definition 646.115 leaves no second terminal route through quotient streams, selected host limits, host real equality, countable-choice rows, independent seal-budget routes, or ambient completeness. Thus the displayed rows exhaust the Real seal scope. □

Theorem 646.125 (Diagonal-limit compatibility open-phase budget terminal cover)**.** *Every current Phase Real and Completion consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet reaches its terminal Real or Completion endpoint only through the displayed rows* D, W, Q, L*, the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. The terminal cover refuses quotient streams, selected host limits, host real equality, countable-choice rows, hidden observation budgets, independent seal-budget routes, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility open phase budget terminal cover`

Proof. Begin with finite observation-budget factorization. Definition 646.120 places every admitted Real or Completion consumer on the displayed route through the dyadic ledger D, the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, the $\mathsf{Real}^{\uparrow}$ seal L, the selector rows, synchronizer rows, transports, continuations, provenance, and local name.

Next use selector-synchronizer exactness. Definition 646.121 identifies the displayed selector and synchronizer rows as the unique terminal root-budget route and leaves no independent budget path.

The admission criterion supplies the gate for the open phase. By Definition 646.123, a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ request reaches the terminal route exactly when it factors through both the FiniteObservationBudgetSele rows and the $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ request, seal, and budget rows.

Finally apply the Real-seal scope theorem. Definition 646.124 exhausts the endpoint by D, W, Q, L, the selector and synchronizer rows, hsame transports, Cont routes, Pkg provenance, and local NameCert. The same theorem routes quotient streams, selected host limits, host real equality, countable-choice rows, hidden observation budgets, independent seal-budget routes, and ambient completeness to the refusal boundary, giving the terminal cover. □

Theorem 646.126 (Diagonal-limit compatibility root-budget functor selector lock)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-budget packet, and let an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet be consumed along the route of Definition 646.117. For every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* K*, terminal admission is exactly joint preservation of the displayed* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *rows and the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *request, seal, and budget rows. In the admitted case the consumer uses the same terminal root-budget route covered by Definition 646.125; in the refused case it exports no quotient stream, selected host limit, host real equality, countable-choice row, hidden observation budget, detached completion object, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root budget functor selector lock`

Proof. First consume the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ packet through Definition 646.117. This places the functor source seal, target seal, finite observation rows, completion-consumer rows, componentwise transports, continuation replay, provenance, and local name on the root-budget route rather than on a detached completion endpoint.

Next apply the finite observation-budget factorizations. The carrier-level and displayed factorizations put the selector rows beside D, W, Q, L, H, C, P, N and keep the synchronizer request, seal, and budget rows on the same packet route.

The selector-synchronizer admission criterion then supplies the lock. By Definition 646.123, a completion-facing consumer is admitted exactly when it preserves both the selector rows and the synchronizer rows.

Finally use the open-phase terminal cover. Definition 646.125 identifies the admitted route with the same terminal root-budget route and sends quotient streams, selected host limits, host real equality, countable-choice rows, hidden observation budgets, detached completion objects, and ambient completeness coordinates to the refusal boundary. □

Theorem 646.127 (Diagonal-limit compatibility root readback exactness)**.** *For an accepted* DiagonalLimitCompatil *packet*

$$K = (R, T, S, D, W, Q, L, H, C, P, N),$$

every root-budget readback first consumes the displayed $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal row* L*, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary carried by* S*. The route is the finite L10 sibling path through chapter 126, chapter 301, chapter 291, chapter 38, and chapter 386, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of* K*.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootReadbackExactness`

Proof. Start with the root budget source and window rows. Definition 646.79 fixes the root budget source as the displayed compatibility packet, and Definition 646.81 places the $\mathsf{StreamName}^{\uparrow}$ window before the regular readback and Real seal handoff.

Read the seal and ledger rows next. Definition 646.82 forces every Real-facing read through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and seal route, while Definition 646.113 records the dyadic ledger, window, readback, seal, transport, route, provenance, and local naming rows as the exact root-budget ledger boundary.

The concrete sibling chapters supply the same route components: StreamName gives the finite window, RegSeqRat gives the regular-rational readback, DyadicRatCore gives the dyadic ledger, RealUp gives the seal row, and CauchyLimitSeal gives the seal boundary. Since Definition 646.83 exports no host limit, quotient stream, selected limit, host real equality, or ambient completeness coordinate, the root readback is exactly this displayed route. □

Theorem 646.128 (Diagonal-limit compatibility root transport replay)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, and let* K' *be obtained by componentwise* hsame *transport of the displayed diagonal source, triangle row, seal route, dyadic ledger, stream window, regular readback, Real seal, provenance, and local naming rows. If the displayed* Cont *routes of* K' *replay the transported rows in the same order, then* K' *preserves the root readback route of Definition 646.127.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root readback transport replay`
↪ variant `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root transport replay`

Proof. First transport the packet rows componentwise. The carrier row of Definition 646.111 exposes exactly the root-facing source, window, seal, dyadic, readback, transport, continuation, provenance, and local-name surface, so hsame transport has no additional coordinate to move.

Next replay the displayed continuations. Definition 646.112 keeps the classifier stable under componentwise hsame movement and displayed Cont replay, and Definition 646.113 keeps the replay attached to the same source, window, seal, and consumer rows.

The replayed route therefore still reads the sibling L10 windows, regular readback, dyadic ledger, Real seal, and CauchyLimitSeal boundary before any completion-facing consumer is reached. The transported packet cannot acquire a second root route, because the root carrier and consumer rows remain exhausted by the displayed finite surface. □

Theorem 646.129 (Diagonal-limit compatibility root consumer coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of the root route of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet is*

exhausted by the rows displayed in Definition 646.127 and preserved by Definition 646.128. Such a consumer can read the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and structural rows H, C, P, N, but it cannot read host limits, quotient streams, selected limits, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootConsumerCoverage`

Proof. Begin with a root-route consumer. Definition 646.83 limits downstream $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and completion-seal reads to the displayed budget, window, diagonal, seal, transport, route, provenance, and local-name rows.

Use the readback exactness theorem above to identify those rows with the concrete L10 sibling route. The consumer reads chapter 126 for windows, chapter 301 for regular readback, chapter 291 for the dyadic ledger, chapter 38 for the Real seal, and chapter 386 for the seal boundary.

Finally apply transport replay. Any componentwise hsame movement and displayed Cont replay reaches the same root route, so every allowed consumer is already listed by the displayed rows. The excluded host-side objects are absent from the packet and cannot be recovered from the root readback surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootL10SpineExhaustion`

Theorem 646.130 (Diagonal-limit compatibility root L10 spine exhaustion)**.** *Every accepted root route*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

whose consumer is $\mathsf{Real}^{\uparrow}$*-facing or completion-facing is equivalent, as a displayed BEDC route, to the finite L10 spine through the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary. The only structural rows available along the spine are* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows displayed by K. The route exports no quotient stream, selected limit, host real equality, ambient completeness, or second completion witness.*

Proof. First identify the displayed root readback. By Definition 646.127, every root-budget readback passes through the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, and CauchyLimitSeal boundary, with the structural rows H, C, P, N attached to the same packet.

Next use transport replay. Definition 646.128 shows that componentwise hsame movement and displayed Cont replay preserve this route, so the displayed BEDC route is unchanged by the allowed structural comparisons.

Finally apply consumer coverage. Definition 646.129 exhausts every $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer by the same five sibling handoffs and the same structural rows. Since the carrier contains no quotient stream, selected limit, host real equality, ambient completeness, or second completion witness, every such consumer is equivalent to the finite L10 spine and nothing beyond it. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootL10SpineExhaustion certificate`

Theorem 646.131 (Diagonal-limit compatibility root selector pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any root selector route for K is the pullback, along the structural rows H, C, P, N, of the displayed* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$

ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *seal row L, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary carried by S. No quotient stream, selected limit, host equality, ambient completeness principle, or second selector source can factor through the same consumer.*

Proof. Start with root readback exactness. Definition 646.127 identifies the selector-facing route with the five concrete L10 certificate surfaces carried by W, D, Q, L, S, and records the structural rows through which the route is read.

Transport the route next. Definition 646.128 shows that componentwise hsame movement and displayed Cont replay preserve exactly the same root readback route, so pulling back along H, C, P, N does not introduce a new selector coordinate.

Finally apply consumer coverage and spine exhaustion. Definition 646.129 limits every admissible consumer to the displayed sibling handoffs and structural rows, while Definition 646.130 excludes quotient streams, selected limits, host real equality, ambient completeness, and any second completion witness. Hence every root selector route factors as the stated pullback and no second selector source reaches the same consumer. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorPullback`

Theorem 646.132 (Diagonal-limit compatibility root selector terminality)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any root selector route for K that reaches a* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer is terminal among the displayed routes: if two admissible selector reads agree on the* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *seal L,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary carried by S, and structural rows H, C, P, N, then they determine the same public consumer endpoint. No quotient stream, selected host limit, host real equality, ambient completeness coordinate, alternate Cauchy witness, or second selector source creates another endpoint.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorTerminality`

Proof. First identify the selector readback. By Definition 646.127, every root-facing read travels through the displayed StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and structural rows of the packet.

Next replay the same rows under transport. The transport replay theorem, Definition 646.128, preserves the displayed route under componentwise hsame movement and Cont replay, so an admissible selector read cannot gain a new coordinate while moving along H, C, P, N.

Now exhaust the consumer boundary. Consumer coverage, Definition 646.129, lists the only rows visible to $\mathsf{Real}^{\uparrow}$-facing and completion-facing consumers, and Definition 646.130 identifies that list with the finite L10 spine.

Finally apply the selector pullback theorem, Definition 646.131. Since both reads pull back from the same five sibling handoffs along the same structural rows, they land at the same public consumer endpoint, with all host-side, quotient, choice, completeness, and alternate-witness routes absent. □

Theorem 646.133 (Diagonal-limit compatibility root selector uniqueness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted DiagonalLimitCompatibility$^{\uparrow}$ *packets. Suppose their root selector routes expose the same* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal row,* CauchyLimitSeal$^{\uparrow}$ *boundary, and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then every Phase Real-facing and completion-facing consumer boundary reads the same displayed selector route from K and K'. No second selector source, quotient stream, selected host limit, host real equality, countable choice principle, ambient completeness coordinate, or alternate Cauchy witness can distinguish the two routes.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorUniqueness`

Proof. First identify both selector reads with the same root handoff surface. By Definition 646.127, each route is read through the displayed StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and the structural rows carried by the packet.

Next transport the displayed support rows. Definition 646.128 moves componentwise hsame transports through the same Cont replay, so the shared BHist, Pkg, and NameCert rows remain attached to the same visible route.

It remains to rule out hidden selector coordinates. Definition 646.129 exhausts every Phase Real-facing and completion-facing consumer by the displayed rows, and Definition 646.130 excludes quotient streams, selected limits, host real equality, ambient completeness, and a second completion witness.

Finally apply Definition 646.131. Since both routes pull back along the same visible support rows to the same five sibling handoffs, every consumer boundary sees the same displayed selector route and no alternate source can distinguish them. □

Theorem 646.134 (Diagonal-limit compatibility root readback selector determinacy)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted DiagonalLimitCompatibility$^{\uparrow}$ *packets. Suppose their root selector routes expose the same* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal row,* CauchyLimitSeal$^{\uparrow}$ *boundary, and the same displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then every* Real$^{\uparrow}$*-facing or completion-facing consumer reads the same root selector route from K and K'. No quotient stream, selected host limit, host real equality, countable choice principle, ambient completeness coordinate, alternate Cauchy witness, or second selector source can distinguish the two accepted routes.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootReadbackSelectorDeterm`

Proof. First identify the readback surface for each packet. Definition 646.127 pins both root reads to the displayed StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and structural rows.

Next replay the common support. Definition 646.128 preserves the displayed route under componentwise hsame movement and Cont replay, so the shared BHist, Pkg, and NameCert rows cannot create a second selector coordinate.

Now exhaust the consumer side. Definition 646.129 restricts every $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer to the displayed rows, and Definition 646.130 identifies that surface with the finite L10 spine.

Finally use the selector results. Definition 646.131 makes every root selector route the pullback of the five sibling handoffs along the displayed support rows, and Definition 646.133 states that two routes with those same exposed rows are indistinguishable to every relevant consumer boundary. Therefore every $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer reads the same root selector route from K and K', and none of the excluded host-side or choice-based data can separate them. □

Theorem 646.135 (Diagonal-limit compatibility four-face terminal pullback)**.** *Let K and K' be accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. Suppose their root routes expose the same* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary, and the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then the two packets determine the same terminal root pullback for the L10 finite route.*

Every top-frontier consumer of the route reads the same displayed pullback. No quotient stream, selected host limit, host real equality, ambient completeness coordinate, alternate Cauchy witness, countable choice row, or second selector source can distinguish K from K' at the root boundary.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility four face terminal pullback`

Proof. First apply Definition 646.127. It identifies the root readback surface with the displayed StreamName window, RegSeqRat readback, DyadicRatCore ledger, Real seal row, CauchyLimitSeal boundary, and structural support rows.

Next use Definition 646.131. The selector route is the pullback of the five sibling handoffs along those same displayed rows, so two packets with matching exposed data land on the same root pullback surface.

Now apply Definition 646.133. Since K and K' expose the same support rows, every selector-facing route available to a consumer is already identified through the same finite packet.

Finally invoke Definition 646.134. Determinacy says that accepted packets with those matching rows are read identically by every $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer, leaving no hidden quotient, selected-limit, host-equality, completeness, Cauchy-witness, choice, or selector coordinate to separate the terminal root pullback. □

Theorem 646.136 (Diagonal-limit compatibility terminal pullback consumer non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of the terminal root pullback from Definition 646.135 factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary, and the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows of the accepted root packet. No quotient stream, selected host limit, host equality, countable choice row, ambient completeness coordinate, alternate Cauchy witness, or second selector source can separate the consumer route from that displayed terminal pullback.*

Proof. First apply Definition 646.127. It identifies the root readback with the visible StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and structural support rows.

Next use Definition 646.130. It says that every relevant root consumer is already equivalent to the finite L10 spine through those five sibling surfaces and the displayed structural rows.

Apply Definition 646.131. The selector route for the accepted packet is the pullback of the same sibling handoffs along the same BHist, hsame, Cont, Pkg, and NameCert support.

Finally combine Definition 646.133 with Definition 646.134. The first theorem rules out a competing selector source once the displayed rows agree, and the second makes every $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer read that same route. Hence the terminal pullback is the only route available to such consumers, with all excluded host-side, quotient, choice, completeness, and alternate-witness coordinates absent. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier terminal pullback consumer nonescape`

Theorem 646.137 (Diagonal-limit compatibility root support projection non-escape)**.** *Let K be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose root route exposes the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Then every admissible support projection out of K factors through exactly those displayed rows. The projection boundary contains no quotient stream, selected host limit, host equality, countable-choice row, ambient completeness coordinate, alternate Cauchy witness, or second selector source.*

Proof. First identify the displayed support rows. Definition 646.127 pins the root readback to the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, CauchyLimitSeal boundary, and structural packet rows.

Next exhaust the finite route. Definition 646.130 says that every $\mathsf{Real}^{\uparrow}$-facing or completion-facing root consumer is already equivalent to that L10 spine, so a support projection has no additional carrier from which to read host-side data.

Now use the selector boundary. Definition 646.131 makes every root selector route a pullback along the displayed BHist, hsame, Cont, Pkg, and NameCert rows, while Definition 646.133 rules out a second selector source once those rows agree.

Finally apply Definition 646.134 together with Definition 646.136. Determinacy fixes the consumer readback on the same displayed selector route, and terminal non-escape states that every terminal-pullback consumer factors through that route. Hence the displayed rows are the only admissible projection boundary, and the excluded quotient, host-limit, host-equality, choice, completeness, alternate-witness, and second-selector coordinates cannot be projected from K. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root support projection nonescape`
Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSupportProjectionNonesc`

Theorem 646.138 (Diagonal-limit compatibility seal-budget synchronizer handoff)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. If its seal route is read through an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet whose displayed request, seal, budget, tail schedule, selector, compatibility, transport, route, provenance, and local naming rows are transported into the same seal route, then the compatibility consumer reads the synchronizer rows before it reads the regular-rational row Q and the* $\mathsf{Real}^{\uparrow}$ *seal row L.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier seal budget synchronizer handoff`

Proof. Start with the compatibility carrier and its seal-route compatibility theorem. The route to Q and L is already constrained by $R, T, S, D, W, H, C, P, N$, so a seal consumer has no entry point outside the displayed compatibility packet.

Now insert the accepted $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ packet at the seal route S. Apply Definition 886.2: its request, seal, budget, shared-tail schedule, diagonal selector, joint compatibility, transport, continuation, provenance, and naming rows must be exposed before its seal row is usable.

Finally use Definition 646.34 and Definition 646.53. They bind the compatibility ledger and final seal to the same finite window and dyadic surface, so the synchronizer rows are consumed before Q, L rather than after the seal has escaped the packet. □

Theorem 646.139 (Diagonal-limit compatibility tail-schedule readback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose seal-budget synchronizer supplies the tail schedule row. Then the compatibility route reads the* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q and the* $\mathsf{Real}^{\uparrow}$ *seal row L. The readback keeps the tail schedule, finite window row, dyadic ledger, regular-rational row, and real seal row on the same displayed route.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier tail schedule readback`

Proof. First apply Definition 646.138. The synchronizer rows are upstream of the compatibility readback through Q, L, so its tail schedule row is available before the final seal is consumed.

Next use Definition 681.3. The schedule route reads the precision request, finite-window schedule, dyadic ledger, regular-rational readback, and real seal in that order.

Transport this route back through the compatibility packet by Definition 646.44. The regular row Q is fixed by the same triangle, dyadic, and window rows already displayed in K.

Finally apply Definition 646.43. It places the schedule-supported readback on the route from R through T, D, W, S, Q to L, so the tail schedule is read before the RegSeqRat and RealUp seal rows. □

Theorem 646.140 (Diagonal-limit compatibility real-seal budget nonescape)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose seal route is read through an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet and whose tail schedule is supplied by* $\mathsf{TailCofinalitySchedule}^{\uparrow}$*, the combined route exports only the displayed compatibility rows, synchronizer rows, tail-schedule rows, regular-rational readback, and* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient stream, host real equality, selected host limit, countable choice row, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier real seal budget nonescape`

Proof. Use Definition 646.48 for the compatibility packet. It removes quotient streams, selected host limits, host real equality, choice data, and ambient completeness from every completion-facing read of K.

Use Definition 886.3 for the inserted synchronizer route. A seal read must factor through both the displayed seal row and the displayed budget row, and any tail use must pass through the selector, shared threshold, and compatibility row.

Use Definition 681.6 for the schedule component. It forces the readback through the finite schedule, dyadic ledger, regular-rational row, and real seal, while excluding quotient streams, host-selected limits, ambient completeness, hidden tail oracles, and unbounded schedules.

Combining these three boundaries leaves only the displayed finite packets and their structural rows. There is no coordinate in the combined route from which a quotient stream, host real equality, selected host limit, countable choice row, or ambient completeness theorem can be exported. □

Theorem 646.141 (Diagonal-limit compatibility synchronizer-tail selector Real route)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose its seal route is read through an accepted* CauchySealBudgetSynchronizer$^{\uparrow}$ *packet and its tail schedule is supplied by an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet on the same displayed route. Then every selector-facing Real consumer reaches the* RegSeqRat$^{\uparrow}$ *row Q and the* Real$^{\uparrow}$ *seal row L only through the finite route K together with the synchronizer request, seal, budget, selector, compatibility, transport, route, provenance, local naming rows, and the tail-schedule precision, window, dyadic, regular-rational, and Real-seal rows.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier synchronizer tail selector real route`

Proof. First apply Definition 646.138. It places the synchronizer rows before the regular-rational row and the Real seal inside the displayed compatibility packet.

Next apply Definition 646.139. The tail schedule is read through the same finite window, dyadic ledger, regular-rational row, and Real seal route, so it cannot be consumed as an external cofinality object.

Finally use Definition 646.34 and Definition 646.53. These exactness rows identify W, D, Q, L as the unique displayed route from the selector budget into the Real seal. The combined consumer therefore reaches Q and L only through K plus the synchronizer and schedule rows. □

Theorem 646.142 (Diagonal-limit compatibility selector-budget shared-tail nonescape)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet whose selector budget, seal-budget synchronizer, and tail schedule share the route of Definition 646.141, the combined route exports no quotient stream, selected limit, host real equality, countable-choice tail, or ambient completeness coordinate. The only public Real-facing surface is the displayed selector budget, synchronizer handoff, tail schedule, regular-rational row, and* Real$^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget shared tail nonescape`

Proof. Begin with Definition 646.193. It fixes the source, triangle, seal, dyadic, finite-window, regular-rational, Real-seal, transport, route, provenance, and local naming rows before any selector-facing consumer can read the seal.

Insert the seal-synchronizer handoff using Definition 646.138. The synchronizer consumes only its displayed request, seal, budget, selector, compatibility, transport, route, provenance, and naming rows on the same finite route.

Now apply Definition 646.140. That theorem excludes quotient streams, host real equality, selected host limits, countable-choice rows, and ambient completeness from the shared seal route. Since the tail schedule is supplied by Definition 646.139, there is no second tail coordinate from which a countable-choice tail or ambient completeness datum could be exported. □

Theorem 646.143 (Diagonal-limit compatibility tail-budget observation triple lock)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose the same displayed route consumes an accepted* CauchySealBudgetSynchronizer$^{\uparrow}$ *packet, an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet, and an accepted* RealObservationBudget$^{\uparrow}$ *packet. Then every Real-facing or completion-facing read of this route passes through the compatibility rows of K, the synchronizer budget rows, the tail schedule rows, the observation-budget rows, the* RegSeqRat$^{\uparrow}$ *row Q, and the* Real$^{\uparrow}$ *seal row L.*

The finite route exports no quotient stream, selected host limit, host equality, countable-choice tail, hidden observation budget, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier tail budget observation lock`

Proof. First apply Definition 646.138. It places the synchronizer request, seal, budget, selector, compatibility, transport, route, provenance, and local naming rows before the regular-rational row Q and the Real seal row L.

Next apply Definition 646.139. The tail schedule is read back through the same finite window, dyadic ledger, regular-rational row, and Real seal, so it cannot contribute an external cofinal tail.

Now use Definition 646.141. The selector-facing Real route is already exhausted by K together with the synchronizer and tail-schedule rows. Then Definition 646.142 removes quotient streams, selected limits, host equality, countable-choice tails, and ambient completeness from that shared route.

Finally attach the accepted RealObservationBudget$^{\uparrow}$ packet through the consumer surface of chapter 611. Its finite request rows determine only which displayed observations are read. Combined with Definition 646.140, this leaves no coordinate for a hidden observation budget or for any completion-facing object outside the listed compatibility, synchronizer, schedule, observation-budget, RegSeqRat, and RealUp rows. □

Theorem 646.144 (Diagonal-limit compatibility triple-budget route exhaustion)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose its displayed finite route consumes accepted* CauchySealBudgetSynchronizer$^{\uparrow}$, TailCofinalitySchedule$^{\uparrow}$, *and* RealObservationBudget$^{\uparrow}$ *packets on the same route. Then every Real-facing completion read is exhausted by the compatibility rows of K, the synchronizer request, seal, and budget rows, the tail-schedule precision, window, and dyadic rows, the observation-budget request rows, the* RegSeqRat$^{\uparrow}$ *row Q, and the* Real$^{\uparrow}$ *seal row L.*

This route exports no quotient stream, host real equality, hidden observation budget, selected host limit, countable-choice tail, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier triple budget route exhaustion`

Proof. First apply Definition 646.138. It places the accepted synchronizer request, seal, budget, selector, compatibility, transport, route, provenance, and local naming rows before any read reaches the regular-rational row Q or the Real seal row L.

Next apply Definition 646.139. The accepted tail schedule is read back through the same finite precision, window, dyadic, regular-rational, and Real-seal route, so the completion consumer has no separate cofinal-tail coordinate.

Use Definition 646.141 to identify the selector-facing Real route with the compatibility, synchronizer, and schedule rows already displayed. Then Definition 646.142 excludes quotient streams,

host real equality, selected host limits, countable-choice tails, and ambient completeness from that shared selector-budget route.

Finally apply Definition 646.143. The accepted observation-budget packet contributes only its displayed request rows to the same finite route. The carrier inventory therefore contains exactly the compatibility rows, synchronizer request/seal/budget rows, tail-schedule precision/window/dyadic rows, observation-budget request rows, Q, and L, with no additional coordinate from which a hidden observation budget or any other completion-facing object could be exported. □

Theorem 646.145 (Diagonal-limit compatibility synchronizer-observation cofinal pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the same displayed finite route consumes accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, *and* $\mathsf{RealObservationBudget}^{\uparrow}$ *packets. Let ρ and ρ' be two Real-facing reads of this route. If ρ and ρ' agree on the synchronizer budget rows, the tail-schedule rows, the observation-budget request rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row L, and the displayed $hsame$, $Cont$, Pkg, and $NameCert$ rows of K, then ρ and ρ' determine the same finite cofinal observation route.*

The pullback route contains no quotient stream, host real equality, hidden observation budget, selected host limit, countable-choice tail, or ambient completeness row.

Proof. First apply Definition 646.138. The synchronizer request, seal, budget, selector, compatibility, transport, route, provenance, and local naming rows must be read before either ρ or ρ' can reach Q or L.

Next apply Definition 646.139 and Definition 646.140. The tail schedule is read back through the displayed finite window, dyadic ledger, regular-rational row, and Real seal, while the seal-budget route excludes quotient streams, host real equality, selected host limits, countable-choice rows, and ambient completeness.

Now use Definition 646.141 and Definition 646.142. They identify the selector-facing Real route with the compatibility packet, synchronizer rows, tail schedule, Q, and L, and they leave no separate selector or shared-tail coordinate on which the two reads could differ.

Finally apply Definition 646.143 and Definition 646.144. The accepted observation-budget packet contributes only its displayed request rows, and the exhausted route has precisely the compatibility, synchronizer, schedule, observation-budget, RegSeqRat, and RealUp seal rows listed in the statement. Since ρ and ρ' agree on all of these rows and on the displayed $hsame$, $Cont$, Pkg, and $NameCert$ rows, finite row exhaustion leaves no fifth coordinate from which a distinct cofinal observation route could be read. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier synchronizer observation cofinal pullback`

Theorem 646.146 (Diagonal-limit compatibility synchronizer-observation route uniqueness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let ρ and ρ' be two Real-facing reads of the displayed synchronizer-observation surface. If they agree on the displayed compatibility rows of K, the accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *rows, the accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *rows, the accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *request rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row L, and*

the displayed hsame, Cont, Pkg, *and* NameCert *rows, then they determine the same synchronizer-observation route.*

The two reads cannot differ by quotient stream, hidden observation budget, selected host limit, host equality, countable-choice tail, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier synchronizer observation route uniqueness`

Proof. First apply Definition 646.141. It places every selector-facing Real read on the displayed compatibility packet together with the synchronizer rows, the tail-schedule rows, Q, and L.

Next use Definition 646.142. This removes quotient streams, selected host limits, host equality, countable-choice tails, and ambient completeness from the shared route, so agreement cannot be broken by an external Real-facing coordinate.

Apply Definition 646.143. The observation-budget packet contributes only its accepted request rows on the same finite route; hence a hidden observation budget has no separate source.

Finally combine Definition 646.144 with Definition 646.145. The triple-budget route is exhausted by the listed compatibility, synchronizer, schedule, observation-budget, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows, and the cofinal pullback identifies any two reads that agree on those rows. Thus ρ and ρ' determine one synchronizer-observation route. □

Theorem 646.147 (Diagonal-limit compatibility budget source lock)**.** *Every accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet reads its* CauchyDiagonalBudget$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, RegularCauchyTriangleBound$^{\uparrow}$, *and* StreamName$^{\uparrow}$ *rows before any seal route is available. The first seal-facing consumer must therefore pass through the displayed budget source, mesh budget, triangle bound, and stream-window rows of the finite compatibility packet.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSourceLock`

Proof. Start with the carrier inventory in Definition 646.1. The packet names the diagonal source, triangle estimate, stream window, regular readback, Real seal row, transport rows, continuation routes, package provenance, and local certificate in one finite surface.

Apply Definition 646.21, Definition 646.12, and Definition 646.79. These anchors identify the budget source with the displayed diagonal, mesh, triangle, and stream rows rather than with a hidden analytic input.

Finally use Definition 646.69. The seal route is synchronized with the same source rows, so no CauchyLimitSeal$^{\uparrow}$ read can occur before those budget rows have been exposed. □

Theorem 646.148 (Diagonal-limit compatibility seal-budget route)**.** *The* CauchyLimitSeal$^{\uparrow}$ *and* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff exported by an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet is a displayed finite route through the budget rows of Definition 646.147. It has no second route through a selected limit, quotient stream, host equality test, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealBudgetRoute`

Proof. The source lock fixes the budget-facing prefix. By Definition 646.147, the route must first read the diagonal budget, mesh budget, triangle bound, and stream-window rows.

Now apply Definition 646.7, Definition 646.82, and Definition 646.138. These results place the CauchyLimitSeal$^{\uparrow}$ read and the CauchySealBudgetSynchronizer$^{\uparrow}$ handoff downstream of the same finite budget prefix.

The non-escape clause is Definition 646.140. Since the packet carries no selected limit, quotient stream, host equality coordinate, or ambient completeness input, the displayed seal-budget route exhausts the available handoff. □

Theorem 646.149 (Diagonal-limit compatibility completion consumer gate)**.** *Every* Completion$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet factors through the displayed budget-source route of Definition 646.147 and the seal-budget route of Definition 646.148. The consumer gate exports no quotient stream, selected limit, host equality, or ambient completeness input.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCompletionConsumerGate`

Proof. Begin with a Completion$^{\uparrow}$ or Real$^{\uparrow}$ consumer route. The completion boundary and regular readback anchors, Definition 646.35 and Definition 646.44, place that consumer downstream of the displayed regular readback and seal rows.

Use Definition 646.148. The seal-facing part of the consumer is already forced through the budget-source prefix and the finite CauchyLimitSeal$^{\uparrow}$ to CauchySealBudgetSynchronizer$^{\uparrow}$ handoff.

It remains only to exclude hidden inputs. Apply Definition 646.48 and Definition 646.83; together they exhaust the consumer surface by the displayed packet rows, with no quotient stream, selected limit, host equality, or ambient completeness coordinate. □

Theorem 646.150 (Diagonal-limit compatibility budget-source readback totality)**.** *Every* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or* Completion$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet first reads the displayed* CauchyDiagonalBudget$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, RegularCauchyTriangleBound$^{\uparrow}$, *and* StreamName$^{\uparrow}$ *rows. After that readback prefix, the consumer must pass through the seal-budget route of Definition 646.148 before any* CauchyLimitSeal$^{\uparrow}$ *read or* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff is available. The consumer surface has no alternate route through quotient streams, selected limits, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSourceReadbackTotalit`

Proof. The budget-source lock Definition 646.147 fixes the first visible prefix of the packet. Thus any consumer that reaches the regular readback, Real row, or completion row has already read the displayed diagonal budget, mesh budget, triangle bound, and stream-window rows.

Next use the seal-budget route of Definition 646.148. It places the CauchyLimitSeal$^{\uparrow}$ read and the CauchySealBudgetSynchronizer$^{\uparrow}$ handoff downstream of that same finite budget prefix, so the seal-facing part of the consumer cannot bypass the displayed source rows.

Finally apply Definition 646.149. Its consumer gate covers the Real$^{\uparrow}$ and Completion$^{\uparrow}$ exits, and the regular readback row is one of the displayed rows feeding those exits. The non-escape clauses exclude quotient streams, selected limits, host equality, and ambient completeness, so the budget-source readback prefix is total for all listed consumers. □

Theorem 646.151 (Diagonal-limit compatibility synchronized budget terminality)**.** *Every* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or* Completion$^{\uparrow}$ *consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet that reaches a* CauchyLimitSeal$^{\uparrow}$ *read or a* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff is exhausted by the synchronized budget route. Such a consumer first passes through the displayed budget-source readback prefix and then through the synchronized seal-source route, with no escape through quotient streams, host limits, host equality, choice data, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier synchronized budget terminality`

Proof. Apply Definition 646.147. It fixes the first visible rows of any seal-facing packet route as the diagonal budget, mesh budget, triangle bound, and stream-window source rows.

Next apply Definition 646.148. The CauchyLimitSeal$^{\uparrow}$ read and the CauchySealBudgetSynchronizer$^{\uparrow}$ handoff are downstream of the same displayed prefix, so the seal-source route is synchronized with the budget-source route rather than supplied by a second analytic object.

Use Definition 646.149 for the Real$^{\uparrow}$ and Completion$^{\uparrow}$ exits, and use Definition 646.150 for the RegSeqRat$^{\uparrow}$ readback side. Together these results cover all listed consumers that reach the seal handoff.

The remaining non-escape clauses are exactly the absence claims carried by those gates. The terminal route exposes no quotient stream, host limit, host equality, choice datum, or ambient completeness coordinate, so every listed consumer is exhausted by the synchronized budget route. □

Theorem 646.152 (Diagonal-limit compatibility budget-seal handoff determinacy)**.** *For every accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet and every* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*, or* Completion$^{\uparrow}$ *consumer that reaches either a* CauchyLimitSeal$^{\uparrow}$ *read or a* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff, the consumer route is determined by the displayed prefix consisting of the* CauchyDiagonalBudget$^{\uparrow}$*,* RealDiagonalMeshBudget$^{\uparrow}$*,* RegularCauchyTriangleBound$^{\uparrow}$*, and* StreamName$^{\uparrow}$ *rows, followed by the seal-budget route. There is no second route through quotient streams, selected limits, host equality, choice data, or ambient completeness.*

Proof. First apply Definition 646.147. It fixes the visible prefix for every seal-facing route as the diagonal budget, mesh budget, triangle bound, and stream-window rows of the accepted packet.

Next apply Definition 646.148. The CauchyLimitSeal$^{\uparrow}$ read and the CauchySealBudgetSynchronizer$^{\uparrow}$ handoff occur only after that same prefix, so the handoff is a route through the displayed budget rows rather than through a separate selected limit or quotient stream.

The consumer cases are covered by Definition 646.149 and Definition 646.150. These two gates cover the Real$^{\uparrow}$ and Completion$^{\uparrow}$ exits together with the RegSeqRat$^{\uparrow}$ readback side, so every listed consumer reaches the handoff through the same budget-to-seal surface.

Finally use Definition 646.151. Since the synchronized terminal route exposes no quotient stream, selected host limit, host equality coordinate, choice datum, or ambient completeness input, the displayed prefix followed by the seal-budget route is the unique available consumer route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSealHandoffDeterminac`

Theorem 646.153 (Diagonal-limit compatibility root-route budget package)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C,$ *be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. The root-route budget package for* K *is the finite BEDC surface that first reads the* CauchyDiagonalBudget$^{\uparrow}$ *source row, the* RealDiagonalMeshBudget$^{\uparrow}$ *mesh row, the* RegularCauchyTriangleBound$^{\uparrow}$ *triangle row, and the* StreamName$^{\uparrow}$ *window row, and then exposes the regular readback* Q*, the seal route* S*, the Real seal row* L*, the displayed* BHist *rows, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. No root-route consumer can replace this package by a host limit, quotient stream, host real equality, choice datum, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteBudgetPackage`

Proof. Project the accepted compatibility carrier. The packet exposes the diagonal source, triangle estimate, seal route, dyadic ledger, stream window, regular readback, Real seal row, transport rows, continuation routes, package provenance, and local name in one finite BHist surface.

Apply Definition 646.147. It fixes the budget prefix through the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows before any seal-facing route is available.

Now apply Definition 646.99 and Definition 646.113. These identify the root budget route with the same D, W, Q, L, H, C, P, N surface and place the ledger boundary on the displayed packet rows.

Finally use Definition 646.48. Since the carrier exports no host limit, quotient stream, host equality, choice datum, or ambient completeness coordinate, the displayed root-route budget package exhausts the available consumer surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root route budget package`

Theorem 646.154 (Diagonal-limit compatibility root-route transport stability)**.** *Let $K = (R, T, S, D, W, Q, L, H,$ and $K' = (R', T', S', D', W', Q', L', H', C', P', N')$ be accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets whose displayed rows are related by componentwise* hsame *transports and whose continuation routes replay the same budget-source, mesh-budget, triangle-bound, stream-window, regular-readback, seal-route,* Pkg*, and* NameCert *rows. Then the root-route budget package of K transports to the root-route budget package of K' without changing its consumer classifier.*

Proof. Start with the componentwise transport rows. The carrier of each packet contains the same kinds of diagonal source, triangle, seal, dyadic, window, readback, Real seal, provenance, and local naming coordinates.

Apply Definition 646.112. The classifier is stable under the displayed hsame transports and under replay of the displayed Cont routes.

Next use Definition 646.81 and Definition 646.82. They keep the stream-window, regular-readback, and Real-seal reads in the same order after transport.

The package therefore has no second classifier branch. The transported packet reads the same budget-source, mesh-budget, triangle-bound, stream-window, regular-readback, seal-route, Pkg, and NameCert rows, so the root-route package is stable. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteTransportStability`

Theorem 646.155 (Diagonal-limit compatibility root-route consumer exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads the root route of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet consumes exactly the displayed package of Definition 646.153: the budget source, mesh budget, triangle bound, stream window, dyadic ledger, regular readback, seal route, Real seal row,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The consumer has no earlier access to Q, S, or L, and it has no later route that bypasses those rows.*

Proof. First read the package theorem. It places the budget source, mesh budget, triangle bound, and stream window before the regular readback, seal route, and Real seal row.

Use Definition 646.101. Every Real-facing or completion-facing seal read reaches L only after S, D, W, Q have been fixed on the displayed compatibility route.

Next apply Definition 646.102. The handoff consumes the root-budget route, selector-budget route, and Real-seal readback lock as one carrier boundary.

It remains to identify the exact consumer surface. Apply Definition 646.149; every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read factors through the budget-source and seal-budget routes, so the listed rows are exactly the root-route consumer package. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteConsumerExactness`

Theorem 646.156 (Diagonal-limit compatibility root-route non-escape)**.** *No root-route consumer of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet can obtain a quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness principle through the root-route budget package. Every exported read remains inside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *package over the diagonal budget, mesh budget, triangle bound, stream window, regular readback, seal route, and Real seal rows.*

Proof. Begin with the package carrier. By Definition 646.153, the root route is the finite surface over the displayed budget, mesh, triangle, window, readback, seal, Real, transport, continuation, provenance, and local naming rows.

Apply Definition 646.192. The budget-selector handoff cannot introduce quotient streams, choice-selected limits, host real equality, or ambient completeness coordinates.

Now apply Definition 646.83 and Definition 646.48. These non-escape boundaries exhaust both the root consumer and the completion-facing consumer by the displayed packet rows.

Therefore a root-route read has no exported coordinate from which the excluded objects could be recovered. The read remains inside the finite BHist/hsame/Cont/Pkg/NameCert package. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteNonescape`

Theorem 646.157 (Diagonal-limit compatibility root-route public factorization)**.** *Let $K = (R, T, S, D, W, Q, L, H,$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every public* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *route that consumes the root route of K factors through the exact package of Definition 646.153. The factorization exposes the budget-source, mesh-budget, triangle-bound, stream-window, regular-readback, seal-route, Real-seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before any public read, and exports no host limit, quotient stream, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate.*

Proof. Apply Definition 646.155. It identifies the public Real$^{\uparrow}$ and Completion$^{\uparrow}$ consumer surface with exactly the displayed root-route package, so every public read begins from the budget-source, mesh-budget, triangle-bound, stream-window, regular-readback, seal-route, Real-seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Next use Definition 646.154. The classifier survives componentwise hsame transport and replay of the displayed Cont routes, so the factorization is reusable by downstream public consumers rather than tied to a single presentation of K.

Finally apply Definition 646.156. The same package exports no host limit, quotient stream, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate. Thus the public route factors through the exact finite root-route package before any public read and has no escape coordinate outside it. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRoutePublicFactorizatio`

Theorem 646.158 (Diagonal-limit compatibility root-route classifier exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every classifier read of the root route of K is exhausted by the same finite root-route package: the budget-source row, mesh-budget row, triangle-bound row, stream-window row, regular readback Q, seal route S, Real seal row L, displayed* BHist *rows, componentwise* hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N. Thus the classifier boundary has no coordinate outside that package for a host limit, quotient stream, host real equality, choice datum, hidden tail oracle, ambient completeness principle, or private downstream read.*

Proof. First read the package carrier. By Definition 646.153, the accepted root route exposes exactly the finite budget-source, mesh-budget, triangle-bound, stream-window, regular-readback, seal-route, Real-seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 646.154. Componentwise hsame transport and replay of the displayed Cont routes preserve the classifier, so the boundary is attached to the package rows rather than to a presentation of K.

Use Definition 646.155. It identifies every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer read with the same row inventory, so a classifier read has no earlier or later consumer coordinate outside the displayed package.

Finally combine Definition 646.156 with Definition 646.157. The non-escape theorem excludes host, quotient, choice, hidden-tail, and completeness coordinates, while the public factorization makes the same finite classifier boundary available to downstream public consumers. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteClassifierExhausti`

Theorem 646.159 (Diagonal-limit compatibility root-route source exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H, C$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every accepted root-route read first exhausts the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row, the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh-budget row, the* RegularCauchyTriangleBoun *triangle-bound row, and the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row before it can expose Q, S, L, H, C, P, or N. No root-route consumer has an alternate source row for a host limit, quotient stream, hidden selector, or ambient completeness witness.*

Proof. Begin with the finite package of Definition 646.153. It places the budget source, mesh budget, triangle bound, and stream window at the front of the accepted root-route surface.

Apply Definition 646.155. The theorem identifies every $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumer with exactly that package, so no later consumer can read Q, S, L, H, C, P, or N before the four source rows are fixed.

Finally use Definition 646.156. Since the route exports no host limit, quotient stream, hidden selector, or ambient completeness coordinate, the displayed four rows exhaust the source side of the root route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root route source exhaustion`

Theorem 646.160 (Diagonal-limit compatibility root-route seal window lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. After the source rows of Definition 646.159 are exhausted, every seal-facing read is locked to the route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The lock reaches the regular readback Q, then the seal route S, then the Real seal row L, and it cannot bypass through a quotient stream, selected host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness row.

Proof. First use Definition 646.159. The stream-window row is already fixed before any read can reach the regular readback, seal route, or Real seal row.

Apply the package theorem Definition 646.153. Its route order exposes the $\mathsf{StreamName}^{\uparrow}$ window, the regular readback Q, the seal route S, and the Real seal row L inside one finite BHist/hsame/Cont/Pkg/NameCert surface.

Now use Definition 646.157 and Definition 646.158. Public consumers and classifier reads both factor through the same finite package, so the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ window is the only terminal seal route.

The exclusions follow from Definition 646.156. The lock has no exported coordinate for a quotient stream, selected host limit, host equality, choice datum, hidden tail oracle, or completeness row. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteSealWindowLock`

Theorem 646.161 (Diagonal-limit compatibility root-route terminal totality)**.** *Let $K = (R, T, S, D, W, Q, L, H, C,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of the root route of K factors through exactly the finite surface consisting of the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row over the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *seal-route, and* $\mathsf{Real}^{\uparrow}$ *rows. There is no terminal consumer outside that surface.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteTerminalTotality`

Proof. Start with source exhaustion and the seal-window lock. By Definition 646.159 the route first reads the budget, mesh, triangle, and stream rows, and by Definition 646.160 every seal-facing read then passes through the $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ window.

Apply Definition 646.155. It identifies the terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumer inventory with the finite root-route package rather than with an external analytic object.

Use Definition 646.157. Every public terminal route is therefore a factorization through the displayed BHist, hsame, Cont, Pkg, and NameCert surface.

Finally combine Definition 646.158 with Definition 646.156. Classifier reads have no coordinate outside the package, and the excluded host, quotient, choice, hidden-tail, and completeness objects are not exported. Hence the listed finite surface is terminally total for root-route $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers. □

Theorem 646.162 (Diagonal-limit compatibility root-route selector lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Before any regular readback Q, seal route S, or Real seal row L can be read along the root route, the route has already fixed the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row, the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh row, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row, and the* $\mathsf{StreamName}^{\uparrow}$ *window row. The lock is part of the root-route package itself, so no* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer can choose a different selector row when it later reads Q, S, or L.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteSelectorLock`

Proof. Start with Definition 646.153. Its package reads the source, mesh, triangle, and window rows before exposing the regular readback, seal route, and Real seal row.

Use Definition 646.155. Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer reads exactly that package, with no earlier access to Q, S, or L and no later bypass around the displayed rows.

Finally apply Definition 646.158. Since every classifier read is exhausted by the same finite row inventory, the selector rows are locked before the readback or seal coordinates are available to any downstream consumer. □

Theorem 646.163 (Diagonal-limit compatibility root-route terminal budget lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *public read through K first consumes the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source*

row, the $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh row, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row, the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-readback row, the seal-route row, the* $\mathsf{Real}^{\uparrow}$ *seal row, the displayed* BHist *rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. The lock is terminal: after those rows are fixed, no further public read of* Q*,* S*, or* L *can introduce a host limit, quotient stream, selected limit, host real equality, choice datum, or ambient completeness row.*

Proof. First use Definition 646.153. It gives the finite package over the budget source, mesh budget, triangle bound, stream window, regular readback, seal route, Real seal, transport, continuation, provenance, and local naming rows.

Next apply Definition 646.159 and Definition 646.160. The source theorem exhausts the budget, mesh, triangle, and stream rows before any readback coordinate is exposed; the seal-window lock then routes the terminal read through the $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ window.

The terminal access to Q, S, and L is fixed by Definition 646.162 and Definition 646.155. Together they make the selector order part of the exact consumer package for every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read.

Finally combine Definition 646.157, Definition 646.158, and Definition 646.156. Public readback and classifier readback are exhausted by the same finite package, and the non-escape boundary excludes host limits, quotient streams, selected limits, host equality, choice data, and ambient completeness rows. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteTerminalBudgetLock`

Theorem 646.164 (Diagonal-limit compatibility route budget exhaustion)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P,$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read through* K *is exhausted by the ordered finite route*

$$\mathsf{CauchyDiagonalBudget}^{\uparrow} \longmapsto \mathsf{RealDiagonalMeshBudget}^{\uparrow} \longmapsto \mathsf{RegularCauchyTriangleBound}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

The read has no remaining public coordinate outside that route: not a host limit, quotient stream, selected limit, host real equality, choice datum, hidden tail oracle, detached classifier row, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRouteBudgetExhaustion`

Proof. Begin with Definition 646.159. It exhausts the source side of the accepted packet by the diagonal budget, Real mesh budget, triangle bound, and stream-window rows before any terminal readback is exported.

Next apply Definition 646.160 and Definition 646.162. These place the $\mathsf{StreamName}^{\uparrow}$ window between the source rows and the later $\mathsf{RegSeqRat}^{\uparrow}$, seal-route, and $\mathsf{Real}^{\uparrow}$ rows, so Q, S, and L are read only after the source order has been fixed.

Use Definition 646.163. It identifies the terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read with the displayed budget, mesh, triangle, stream, regular-readback, seal, Real, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

Finally combine Definition 646.158 with Definition 646.156. The classifier inventory contains no extra row beyond the displayed package, and the non-escape boundary excludes host limits, quotient streams, selected limits, host equality, choice data, hidden tail oracles, and ambient completeness coordinates. Hence the ordered route exhausts every terminal read. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility route budget exhaustion`

Theorem 646.165 (Diagonal-limit root-route budget terminal exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read preserving the root-route budget rows is exhausted by the ordered route through* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *the seal route,* $\mathsf{Real}^{\uparrow}$, *and the displayed* $\mathsf{BHist/hsame/Cont/Pkg/NameCert}$ *support rows. The terminal read exports no quotient stream, host real equality, selected limit, hidden tail oracle, choice datum, metric-completeness datum, ambient completeness coordinate, bridge row, or row outside the accepted packet.*

Proof. First apply Definition 646.159. The budget source, mesh budget, triangle bound, and stream-window rows are fixed before the read reaches any regular readback, seal-route, or Real-seal row.

Next use Definition 646.160 and Definition 646.162. These locks place the terminal route through the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ window and prevent a second selector surface from choosing a different route.

Now apply Definition 646.163 and Definition 646.158. The terminal read and its classifier are both exhausted by the same finite budget, mesh, triangle, stream, readback, seal, Real, transport, continuation, package, and local naming rows.

Finally use Definition 646.156. The non-escape boundary excludes quotient, host-equality, selected-limit, choice, hidden-tail, completeness, bridge, and off-packet coordinates, so the ordered finite route supplies the entire terminal read. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteBudgetTerminalExha`

Theorem 646.166 (Diagonal-limit compatibility Cauchy-refinement threshold correspondence)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-route packet, and let*

$$R_c = (M_0, M_1, U, V, T_c, W_c, Q_c, E_c, H_c, C_c, P_c, N_c)$$

be an accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root threshold route. If a terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read through K is compared against a public root threshold read through R_c, then the comparison has exactly the same four L10 route faces on both sides: a* $\mathsf{StreamName}^{\uparrow}$ *finite window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a* $\mathsf{Real}^{\uparrow}$ *seal. Both exported packages are exhausted by the displayed* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows, and neither side exports a quotient stream, host real equality, selected host limit, choice datum, hidden tail oracle, or ambient completeness row.*

Proof. First put the diagonal side in terminal form. By Definition 646.164, every terminal read through K is exhausted by the ordered route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the seal row, $\mathsf{Real}^{\uparrow}$, and the displayed $\mathsf{BHist/hsame/Cont/Pkg/NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ rows.

The terminality needed for comparison is supplied by Definition 646.161. Thus any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read eligible for comparison has already reached the same diagonal route package; there is no detached classifier or unaccounted tail coordinate left outside the finite route.

Now put the Cauchy side in threshold form. The root threshold public exhaustion theorem, Definition 885.133, confines every public consumer of R_c to the displayed route $M_0, M_1, U, V, T_c, W_c, Q_c, E_c, H_c, C_c, P_c,$ and its componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally apply Definition 885.136. It identifies the Cauchy route faces as the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal, with exactly the same public support-row vocabulary. Comparing the two exhausted packages therefore matches

those four L10 faces and leaves no quotient stream, host real equality, selected host limit, choice datum, hidden tail oracle, or ambient completeness row to export on either side. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCauchyRefinementThresholdCo`

Theorem 646.167 (Diagonal-limit compatibility StreamName budget triad)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *window is the unique middle budget face between the source triad* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *and the terminal seal rows* $\mathsf{RegSeqRat}^{\uparrow}$, S, *and* $\mathsf{Real}^{\uparrow}$*. Consequently, any two terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *reads through accepted packets that agree on the source triad, the* $\mathsf{StreamName}^{\uparrow}$ *window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback agree on the same public consumer boundary.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityStreamNameBudgetTriad`

Proof. First use Definition 646.164. It displays the only ordered route available to a terminal read and places $\mathsf{StreamName}^{\uparrow}$ strictly between the diagonal-budget source rows and the regular-readback plus seal rows.

Apply Definition 646.160. The seal-facing part of the route passes through the $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ window, so no second middle face can be inserted between the source triad and the terminal seal rows.

Now use Definition 646.162. Agreement on the source triad, $\mathsf{StreamName}^{\uparrow}$ window, and $\mathsf{RegSeqRat}^{\uparrow}$ readback fixes the selector surface before S or L is exported.

Finally apply Definition 646.158 and Definition 646.156. Since every classifier read is exhausted by the same locked route and no host or ambient coordinate is exported, two terminal reads agreeing on the triad have the same public consumer boundary. □

Theorem 646.168 (Diagonal-limit compatibility root-route public readback exhaustion)**.** *For every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, public readback of* Q*,* S*, and* L *is exhausted by the finite route of Definition 646.163. The public surface exports only the displayed budget, mesh, triangle, stream-window, regular-readback, seal-route, Real-seal,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, and exports no host limit, quotient stream, selected limit, host real equality, choice datum, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root route public readback exhaustion`

Proof. Start from the terminal budget lock. It places Q, S, and L behind the displayed budget, mesh, triangle, stream, regular, seal, Real, transport, route, provenance, and naming rows.

Use Definition 646.157. Any public $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ route factors through the same finite root-route package before the readback is exported.

Apply Definition 646.158. Since every classifier read is exhausted by the package, the public readback has no second classifier coordinate that could detach Q, S, or L from the locked route.

The excluded exports are ruled out by Definition 646.156. A host limit, quotient stream, selected limit, host real equality, choice datum, or ambient completeness row is not part of the displayed package, so none is available from the public readback. □

Theorem 646.169 (Diagonal-limit compatibility root-route completion readback)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback of the root route factors through the same displayed* $D, W, Q, S, L, H, C, P, N$

surface: the diagonal budget source, stream window, regular readback, seal route, Real seal row, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. Such a readback cannot read a host limit, quotient stream, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate.*

Proof. First invoke Definition 646.155. It identifies every $\mathsf{Completion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer read with the displayed root-route package and places $D, W, Q, S, L, H, C, P, N$ inside that exact surface.

Next use Definition 646.157. Public completion-facing and Real-facing reads factor through the same finite package before any public read is exported, so the readback has no second route outside $D, W, Q, S, L, H, C, P, N$.

The excluded coordinates are ruled out by Definition 646.156. Therefore a completion or Real readback remains a displayed package read and cannot obtain a host limit or quotient stream. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteCompletionReadback`

Theorem 646.170 (Diagonal-limit compatibility root-route seal factorization)**.** *Let $K = (R, T, S, D, W, Q, L, H, C$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any seal-facing public route that exports the Real seal row L factors first through the selector lock of Definition 646.162 and then through the completion readback of Definition 646.169. Hence the exported Real seal row is a public factor of the same root-route package, not a host-selected limit or quotient-stream read.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteSealFactorization`

Proof. Begin with Definition 646.162. It fixes the source, mesh, triangle, and window selector rows before any Q, S, or L coordinate is available.

Then apply Definition 646.169. The seal-facing readback travels through the displayed $D, W, Q, S, L, H, C, P, N$ surface and cannot leave it for a host limit, quotient stream, or ambient completeness coordinate.

It remains to identify the public export. Use Definition 646.157 together with Definition 646.158. The public route and the classifier boundary are both exhausted by the same finite package, so the Real seal row L is exported only after selector-lock and completion-readback factorization. □

Theorem 646.171 (Diagonal-limit compatibility terminal budget factorization)**.** *Let $K = (R, T, S, D, W, Q, L, H,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read of K factors through the ordered finite package consisting of the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row,* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh row,* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, seal route row,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier row, componentwise* hsame *row, displayed* Cont *route row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row. The terminal read exports no quotient stream, host limit, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility terminal budget factorization`

Proof. Start with Definition 646.153. It identifies the accepted root route as the finite package over the budget source, mesh budget, triangle bound, stream window, regular readback, seal route, Real seal, transport, continuation, provenance, and local naming rows.

Apply Definition 646.159. The source side is already exhausted by the budget, mesh, triangle, and stream rows. Then use Definition 646.162; the regular readback, seal route, and Real seal coordinates are available only after that ordered selector surface has been fixed.

Next apply Definition 646.169. It sends every completion-facing or Real-facing read through the displayed readback surface. The seal-facing part is then exactly the factorization of Definition 646.170, so the terminal Real seal export remains a public factor of the same finite package.

Finally use Definition 646.158 and Definition 646.156. The classifier has no row outside the package, and the non-escape theorem excludes quotient streams, host limits, host real equality, choice data, hidden tail oracles, and ambient completeness coordinates. Hence every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read factors through precisely the displayed ordered package. □

Theorem 646.172 (Diagonal-limit compatibility terminal route completeness)**.** *Let $K = (R, T, S, D, W, Q, L, H, C$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. A terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read of K is complete exactly when it factors through the ordered finite route consisting of the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row, the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh row, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the seal-route row, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed* BHist/hsame/Cont/Pkg/NameCert *rows. The complete terminal route exports no quotient stream, host-selected limit, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility terminal route semantic certificate`

Proof. First read the root-route package of Definition 646.153. It supplies the displayed budget, mesh, triangle, stream-window, readback, seal, Real-seal, transport, continuation, provenance, and local naming rows as one finite carrier.

For the forward direction, apply Definition 646.163 and Definition 646.169. These lock every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read to the source, window, regular-readback, seal-route, and Real-seal order, and they identify completion readback with the same displayed $D, W, Q, S, L, H, C, P, N$ surface.

Then use Definition 646.171. It gives the exact ordered factorization through the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, seal-route, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. Thus any complete terminal read is precisely a read through that finite route.

For the reverse direction, a read that factors through the displayed ordered route has already consumed every row made available by the accepted packet. The package theorem and the completion readback theorem show that no additional terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ coordinate remains outside the listed route.

Finally apply Definition 646.156. It excludes quotient streams, host-selected limits, host real equality, choice data, hidden tail oracles, and ambient completeness coordinates from the exported surface, so completeness of the terminal read is exactly completeness of the displayed finite factorization. □

Theorem 646.173 (Diagonal-limit compatibility Monotone Cauchy seal handoff)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet, and let*

$$M = (F, \sigma, \mu, \lambda, I, Q_M, H_M, C_M, P_M, N_M)$$

be an accepted $\mathsf{MonotoneCauchy}^{\uparrow}$ *packet whose public* $\mathsf{Real}^{\uparrow}$ *seal classifier consumes the terminal* $\mathsf{Real}^{\uparrow}$ *read exported by K. Then the combined read factors through the same ordered finite surface: the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row, the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh row, the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *triangle row, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the monotone* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{LocatedInterval}^{\uparrow}$ *rows, the* $\mathsf{Real}^{\uparrow}$ *seal row, and the*

displayed BHist/hsame/Cont/Pkg/NameCert *rows from both packets. The handoff exports no quotient stream, selected limit, host real equality, trichotomy row, ambient completeness principle, or choice datum.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityMonotoneCauchySealHandoff`

Proof. Start with terminal route completeness, Definition 646.172. It places the accepted terminal read of K on the displayed budget, mesh, triangle, stream-window, regular-readback, Real-seal, transport, continuation, provenance, and local naming rows.

Next apply the Monotone Cauchy public route classifier handoff, Definition 438.23. The public completion-facing classifier and the $\mathsf{Real}^{\uparrow}$ seal-facing classifier of M read the located interval trap, monotone tail-window certificate, $\mathsf{RegSeqRat}^{\uparrow}$ row, $\mathsf{CauchyModulus}^{\uparrow}$ tail-stability row, and $\mathsf{Real}^{\uparrow}$ seal row through H_M, C_M, P_M, N_M.

The consumer hypothesis identifies the $\mathsf{Real}^{\uparrow}$ seal read exported by K with the seal-facing classifier input of M. Therefore the composed read first traverses the diagonal terminal route and then the monotone classifier surface, without inserting any row outside the two displayed finite packets.

Finally combine the non-escape clause in terminal route completeness with the non-escape conclusion of the Monotone Cauchy classifier handoff. The diagonal packet has no quotient stream, selected limit, host real equality, choice datum, hidden tail oracle, or ambient completeness coordinate, and the monotone packet has no quotient stream, host sequence equality, selected limit, trichotomy row, ambient completeness principle, or choice datum. Hence the combined seal handoff factors exactly through the listed ordered surface. □

Theorem 646.174 (Diagonal-limit compatibility open-phase exit synchronization)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose terminal route reads the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *diagonal pullback surface and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *Real-completion exit surface. Then every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read synchronizes exactly the four finite exit faces*

$$\mathsf{StreamName}^{\uparrow}(W), \qquad \mathsf{DyadicRatCore}^{\uparrow}(T), \qquad \mathsf{RegSeqRat}^{\uparrow}(Q), \qquad \mathsf{Real}^{\uparrow}(L)$$

through the displayed BHist *row* H*, componentwise* hsame *transport, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row* N*. The synchronized exit exports no quotient stream, host real equality, host-selected limit, choice datum, ambient completeness coordinate, metric-completeness datum, or bridge row.*

Proof. Start with terminal route completeness, Definition 646.172. It places any terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read of K on the displayed ordered route and identifies the available terminal rows as the stream window W, dyadic budget row T, regular readback Q, Real seal L, and the support rows H, C, P, N.

Next apply the diagonal pullback theorem Definition 885.140. The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ surface pulls every compatible diagonal-limit consumer back to the same finite dyadic threshold and the same shared-window ledger, transported by BHist, hsame, Cont, Pkg, and NameCert support rather than by quotient or choice data.

The open-phase spine is supplied by Definition 885.141. It names the four exit faces as the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal, and it orders them as the Phase Real and Completion route consumed by the terminal read.

Finally use Definition 435.128. Its Real-completion exit exhaustion leaves no additional consumer coordinate beyond the finite window, tolerance ledger, regular readback, Real seal, sibling

root route, and displayed support rows. Together with the non-escape clause in terminal route completeness, this rules out quotient streams, host real equality, host-selected limits, choice data, ambient or metric-completeness data, and bridge rows, so the terminal read synchronizes exactly the four finite exit faces. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityOpenPhaseExitSynchronizatio`

Theorem 646.175 (Diagonal-limit compatibility root-route triple lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose its terminal consumer also reads the* CauchyModulusRefinement$^{\uparrow}$ *open-phase exit spine, the* UniformCauchyCriterion$^{\uparrow}$ *Real-completion exit exhaustion, and a* MonotoneCauchy$^{\uparrow}$ *public route classifier handoff. Then every terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *read factors through exactly the displayed finite surface consisting of the* StreamName$^{\uparrow}$ *row, the* DyadicRatCore$^{\uparrow}$ *row, the* RegSeqRat$^{\uparrow}$ *row, the* Real$^{\uparrow}$ *seal row, the monotone tail-window and located-interval rows, and the displayed* BHist/hsame/Cont/Pkg/NameCert *support rows from the participating packets. The locked terminal route exports no quotient stream, host real equality, selected limit, trichotomy row, ambient completeness principle, metric-completeness datum, bridge row, or choice datum.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteTripleLock`

Proof. First invoke Definition 646.174. It synchronizes the terminal read of K to the finite StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ exit faces, carried by the displayed H, C, P, N support rows and excluding quotient, host equality, selected-limit, choice, completeness, and bridge coordinates.

Next use Definition 885.141. The diagonal pullback consumer therefore reads the same open-phase spine rather than a separate analytic modulus: the stream window, dyadic ledger, regular readback, and Real seal are the available exit faces.

Apply Definition 435.128. It exhausts the Real-completion consumer surface by the finite window, tolerance ledger, regular readback, Real seal, sibling root route, and displayed support rows. Thus the Real$^{\uparrow}$ and Completion$^{\uparrow}$ terminal read has no additional metric-completeness or ambient-completeness coordinate.

Finally invoke Definition 438.23. The public monotone handoff identifies the remaining completion-facing surface with the monotone tail-window certificate, located-interval rows, regular-sequence row, Real$^{\uparrow}$ seal row, and its own BHist/hsame/Cont/Pkg/NameCert support. Combining the four invoked locks leaves exactly the displayed finite surface, so no quotient stream, host real equality, selected limit, trichotomy row, ambient completeness principle, metric-completeness datum, bridge row, or choice datum can be read by the terminal consumer. □

Theorem 646.176 (Diagonal-limit compatibility root-route seal readback classifier)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. For any terminal* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *seal readback of K, the classifier is exactly the ordered finite route through the* CauchyDiagonalBudget$^{\uparrow}$ *source row, the* RealDiagonalMeshBudget$^{\uparrow}$ *mesh row, the* RegularCauchyTriangleBound$^{\uparrow}$ *triangle row, the* StreamName$^{\uparrow}$ *window row, the* RegSeqRat$^{\uparrow}$ *readback row, the seal-route row, the* Real$^{\uparrow}$ *seal row, and the displayed* BHist/hsame/Cont/Pkg/NameCert *rows. It has no coordinate for a quotient stream, host equality, choice datum, hidden tail oracle, metric-completeness datum, bridge row, ambient completeness coordinate, or row outside the accepted packet.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root route seal readback classifier`

Proof. Start with terminal route completeness, Definition 646.172. It identifies completeness of any terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal readback with factorization through the displayed budget, mesh, triangle, stream-window, regular-readback, seal-route, Real-seal, transport, continuation, provenance, and local naming rows.

Next apply Definition 646.158. The classifier has no second inventory behind the seal readback; every classifier row available to the terminal consumer is already one of the displayed rows of the accepted route.

Finally use Definition 646.156. It excludes quotient streams, host equality, choice data, hidden tail oracles, metric or ambient completeness coordinates, bridge rows, and every other non-displayed analytic coordinate. Thus the seal-readback classifier is precisely the ordered finite route stated above. □

Theorem 646.177 (Diagonal-limit compatibility Cauchy-refinement consumer correspondence)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read is compared with a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *threshold route and a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *Real-completion exit route. The two consumer surfaces expose the same four exit faces*

$$\mathsf{StreamName}^{\uparrow}(W), \qquad \mathsf{DyadicRatCore}^{\uparrow}(T), \qquad \mathsf{RegSeqRat}^{\uparrow}(Q), \qquad \mathsf{Real}^{\uparrow}(L)$$

under the displayed support rows H, C, P, N. Neither consumer exports a quotient stream, host equality, choice datum, hidden tail oracle, metric-completeness datum, bridge row, ambient completeness coordinate, or any other excluded analytic coordinate.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRouteCauchyRefinementCo`

Proof. Begin with Definition 646.174. It places the terminal read of K on the four finite exit faces $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{DyadicRatCore}^{\uparrow}(T)$, $\mathsf{RegSeqRat}^{\uparrow}(Q)$, and $\mathsf{Real}^{\uparrow}(L)$, carried by the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support rows.

For the threshold-route side, apply Definition 885.141. The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ consumer reads the same stream window, dyadic threshold ledger, regular readback, and Real seal rather than a separate quotient or choice-based modulus object.

For the Real-completion exit side, apply Definition 435.128. The $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer is exhausted by the same finite window, tolerance ledger, regular readback, Real seal, sibling route, and displayed support rows.

The exclusions are common to the two surfaces. The synchronization theorem and the two consumer theorems leave no coordinate for quotient streams, host equality, choice data, hidden tail oracles, metric or ambient completeness data, bridge rows, or other analytic exports. Hence the Cauchy-refinement threshold route and the UniformCauchy Real-completion exit route correspond exactly on the four displayed exit faces. □

Theorem 646.178 (Diagonal-limit compatibility four-face consumer equivalence)**.** *Let $K = (R, T, S, D, W, Q, L, H$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the same public consumer also accepts compatible* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *route certificates for the terminal Real and Completion read. Then the following are equivalent as public BEDC routes for that consumer:*

1. *the diagonal terminal route of K;*
2. *the Cauchy threshold exit spine;*

3. the UniformCauchy Real-completion exit exhaustion.

The equivalence identifies exactly the four finite L10 faces

$$\mathsf{StreamName}^{\uparrow}(W), \qquad \mathsf{DyadicRatCore}^{\uparrow}(T), \qquad \mathsf{RegSeqRat}^{\uparrow}(Q), \qquad \mathsf{Real}^{\uparrow}(L)$$

with the same displayed BHist/hsame/Cont/Pkg/NameCert *support rows H, C, P, N. It carries no quotient stream, host real equality, selected limit, choice datum, ambient completeness coordinate, metric-completeness datum, or bridge row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityFourFaceConsumerEquivalence`

Proof. First read the diagonal side through Definition 646.172. It exhausts every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read of K by the displayed terminal route and excludes all off-route analytic coordinates.

Next attach the Cauchy threshold side. The spine theorem Definition 885.141 presents the compatible threshold consumer by the finite stream window, dyadic threshold ledger, regular readback, and Real seal, while Definition 646.177 identifies that surface with the displayed rows W, T, Q, L, H, C, P, N of K.

Finally attach the UniformCauchy exit side. The exhaustion theorem Definition 435.128 leaves the Real-completion consumer with no public coordinate beyond the same finite window, tolerance ledger, regular readback, Real seal, sibling route, and support rows. Since the Cauchy-refinement correspondence already identifies those coordinates with W, T, Q, L, H, C, P, N, the diagonal terminal route, the Cauchy threshold exit spine, and the UniformCauchy Real-completion exit exhaustion determine one public BEDC route.

The common non-escape clauses remove the possible extra readings: quotient streams, host real equality, selected limits, choice data, ambient completeness, metric-completeness, and bridge rows are not exported by any of the three route presentations. Thus the three public routes are equivalent exactly at the four displayed L10 faces and their shared support rows. □

Theorem 646.179 (Diagonal-limit compatibility open-phase four-face readiness)**.** *Let $K = (R, T, S, D, W, Q, L, H$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose terminal consumer accepts the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *Cauchy-refinement route and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *Real-completion route. Then the terminal read is open-phase ready exactly at the four L10 exit faces*

$$\mathsf{StreamName}^{\uparrow}(W), \qquad \mathsf{DyadicRatCore}^{\uparrow}(T), \qquad \mathsf{RegSeqRat}^{\uparrow}(Q), \qquad \mathsf{Real}^{\uparrow}(L),$$

carried by the displayed BHist *row H,* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row N. The readiness surface exposes no quotient stream, host real equality, selected limit, choice datum, completeness datum, or bridge row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityOpenPhaseFourFaceReadiness`

Proof. First apply terminal route completeness, Definition 646.172. It confines the terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read of K to the displayed finite route and excludes quotient streams, selected limits, host real equality, choice data, hidden tail oracles, and ambient completeness coordinates.

Next use open-phase exit synchronization, Definition 646.174. The accepted terminal route is synchronized at the four finite faces $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{DyadicRatCore}^{\uparrow}(T)$, $\mathsf{RegSeqRat}^{\uparrow}(Q)$, and $\mathsf{Real}^{\uparrow}(L)$ over the same support rows H, C, P, N.

The Cauchy-refinement consumer correspondence, Definition 646.177, identifies the Cauchy-refinement threshold route and the UniformCauchy Real-completion route with those same four faces. Thus accepting both route certificates adds no fifth terminal coordinate.

Finally invoke the four-face consumer equivalence, Definition 646.178. It identifies the diagonal terminal route, the Cauchy threshold exit spine, and the UniformCauchy Real-completion exit exhaustion as one public BEDC route. Therefore open-phase readiness is precisely the displayed four-face surface over H, C, P, N, with no quotient stream, host real equality, selected limit, choice datum, completeness datum, or bridge row. □

Theorem 646.180 (Diagonal-limit compatibility root-route selector minimality)**.** *Let $K = (R, T, S, D, W, Q, L, H,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Any terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read of K that reaches the four faces*

$$\mathsf{StreamName}^{\uparrow}(W), \qquad \mathsf{DyadicRatCore}^{\uparrow}(T), \qquad \mathsf{RegSeqRat}^{\uparrow}(Q), \qquad \mathsf{Real}^{\uparrow}(L)$$

must factor through the displayed budget-source row, mesh row, triangle row, and stream-window row before the seal route is read. Thus the root route has no smaller selector surface that still exposes all four faces.

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRootRouteSelectorMinimality`

Proof. Start from the root-route budget package. It places the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ source row, $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ mesh row, $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ triangle row, and $\mathsf{StreamName}^{\uparrow}$ window row before the regular readback, seal route, and Real seal rows.

Use source exhaustion and the seal-window lock. The source exhaustion theorem says that no alternate source row can serve the terminal route, while the seal-window lock prevents a seal-facing read from choosing a different window after the source rows have been displayed.

Finally apply the four-face consumer equivalence. Since the terminal consumer sees exactly $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{DyadicRatCore}^{\uparrow}(T)$, $\mathsf{RegSeqRat}^{\uparrow}(Q)$, and $\mathsf{Real}^{\uparrow}(L)$ over the same support rows, removing any one of the budget-source, mesh, triangle, or stream-window rows destroys the displayed route to at least one face. The remaining non-escape theorem excludes hidden tail oracles, selected host limits, quotient streams, choice data, and ambient completeness as substitutes. □

Theorem 646.181 (Diagonal-limit compatibility root-route budget-source totality)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. The budget-source rows of the root route exhaust all selector inputs for any terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read of K. In particular, the terminal route admits no hidden tail oracle, quotient stream, selected host limit, countable-choice datum, or ambient-completeness coordinate outside the displayed source, mesh, triangle, stream-window, readback, seal, transport, continuation, provenance, and local naming rows.*

Proof. First compose the root-route budget package with source exhaustion. The only selector inputs made available before terminal readback are the displayed diagonal budget source, mesh budget, triangle bound, and stream-window rows.

Next apply the seal-window lock. Once those rows are fixed, the regular readback Q, the seal route S, and the Real seal L can be read only through the same route, with H, C, P, N carrying transport, replay, provenance, and local naming.

The four-face consumer equivalence identifies this route with the public exit surface seen by the Cauchy-refinement and UniformCauchy consumers. The non-escape theorem then removes every proposed outside selector input: hidden tail oracle, quotient stream, selected host limit, countable choice, and ambient completeness are not rows of the accepted packet. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibility root route budget source totality`

Definition 646.182 (DiagonalLimitCompatibility root formal source)**.** *A* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root formal source is a finite packet*

$$S = (B, T, W, R, L, H, C, P, N)$$

where B *is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *source row,* T *is the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *selector row,* W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular handoff row,* L *is the* $\mathsf{Real}^{\uparrow}$ *seal row,* H *records componentwise* hsame *transports,* C *records displayed* Cont *routes,* P *records* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *row.*

The sibling sources are the budget packet of chapter 853, the selector packet of chapter 697, the finite-window surface of chapter 126, the regular rational sequence surface of chapter 301, and the seal surface of chapter 38. The tuple contains no quotient stream, host limit, countable-choice selector, or ambient completeness coordinate.

Lean4 (definition) `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootFormalSource`

Theorem 646.183 (DiagonalLimitCompatibility root formal route)**.** *Every accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *route that is available to a root formal consumer factors through the finite source tuple of Definition 646.182. In particular, the consumer reads the budget row* B*, selector row* T*, finite window row* W*, regular handoff row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* L *before the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ *route is accepted.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootFormalRoute`

Proof. Project the source tuple $S = (B, T, W, R, L, H, C, P, N)$. The diagonal budget row B supplies the precision and mesh budget, and the selector row T supplies the cofinal tail selection used by the diagonal route.

The finite-window and regular handoff rows are then read in order. The $\mathsf{StreamName}^{\uparrow}$ row W displays the finite observation window, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is the regular sequence handoff attached to that same window.

Only after those rows are visible does the route reach the $\mathsf{Real}^{\uparrow}$ seal row L. The transports in H, continuations in C, provenance in P, and local naming row N replay that same order for the accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ consumer, so the route factors through the listed source tuple. □

Theorem 646.184 (DiagonalLimitCompatibility root formal non-escape)**.** *An accepted root formal route for* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *cannot use quotient streams, host limits, countable choice, or ambient completeness to replace any row of the finite source tuple of Definition 646.182.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootFormalNonescape`

Proof. Read the carrier inventory of Definition 646.182. Its only source rows are $B, T, W, R, L, H, C, P, N$, and each row is tied to a displayed sibling chapter: Cauchy diagonal budget, diagonal tail selector, StreamName finite window, RegSeqRat regular handoff, and Real seal.

Quotient streams and host limits would require additional equality or limit coordinates, but none are present in the tuple. Countable choice would require an unbounded selector family beyond T, while ambient completeness would require a completion principle beyond the seal row L.

Thus every accepted route stays inside the finite handoff packet. Any consumer that asks for one of the excluded principles has asked for data not carried by the $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ source tuple. □

Theorem 646.185 (DiagonalLimitCompatibility root readback determinacy)**.** *Let $S = (B, T, W, R, L, H, C, P, N)$ and $S' = (B', T', W', R', L', H', C', P', N')$ be accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root formal routes. Suppose the displayed* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *rows B, B',* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *rows T, T',* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows W, W',* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows R, R', and* $\mathsf{Real}^{\uparrow}$ *seal rows L, L' are paired by their chapter classifiers and transported componentwise by the displayed* hsame *row H against H'. Then the two routes determine the same local* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *readback through H, C, P, N. They cannot be separated by quotient streams, host limits, countable-choice selectors, or ambient completeness.*

Proof. Apply Definition 646.183 to both accepted routes. Each route is forced through the finite source tuple of Definition 646.182, so every readback begins with the displayed source rows B, T, W, R, L and B', T', W', R', L' in the five sibling chapters.

Compare those five source positions componentwise through their classifiers and through the displayed hsame transport row. The budget, selector, finite-window, regular handoff, and seal agreements identify the only external rows available before the local $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ route reaches H, C, P, N.

It remains to exclude a later separating coordinate. By Definition 646.184, neither route can introduce quotient streams, host limits, countable-choice selectors, or ambient completeness in place of the finite tuple. Hence the remaining local readback through H, C, P, N is determined by the classifier-and-transport source data carried by the displayed packet. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRootReadbackDeterminacy`

Theorem 646.186 (DiagonalLimitCompatibility root formal seal factorization)**.** *Every accepted root formal route $S = (B, T, W, R, L, H, C, P, N)$ for* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *factors any* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal read through the ordered rows B, T, W, R, L before the local* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *rows H, C, P, N are consumed. No quotient stream, host limit, countable-choice selector, or ambient completeness row can replace the seal factor L.*

Consequently, after the seal factor L is reached through the displayed $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the remaining readback through H, C, P, N is determined by those same source rows.*

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRootFormalSealFactorizatio`

Proof. Start with the formal route. By Definition 646.183, every accepted root formal consumer reads the ordered tuple B, T, W, R, L before it accepts the local $\mathsf{NameCert}_{\mathsf{DiagonalLimitCompatibility}^{\uparrow}}$ rows.

Next exclude off-packet seal sources. The non-escape theorem Definition 646.184 says that quotient streams, host limits, countable-choice selectors, and ambient completeness rows are not coordinates of the source tuple, so none of them can stand in for the $\mathsf{Real}^{\uparrow}$ seal row L.

Finally apply readback determinacy. Once the displayed source rows B, T, W, R, L have been fixed, Definition 646.185 identifies the subsequent H, C, P, N readback as the one determined by those rows. Thus every Real or completion-facing seal read factors through B, T, W, R, L before the local diagonal-compatibility rows are consumed. □

Theorem 646.187 (DiagonalLimitCompatibility root formal Lean-target boundary)**.** *Every Lean target that formalizes the* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root formal handoff must expose exactly the finite rows*

$$B, T, W, R, L, H, C, P, N$$

from Definition 646.182. The target may project these rows and prove route, non-escape, readback determinacy, or seal factorization over them, but it may not add a quotient stream row, host limit row, countable choice row, ambient completeness row, or any second source row outside the displayed tuple.

Proof. Project the source tuple. The route theorem Definition 646.183 already forces every accepted root formal consumer through $B, T, W, R, L, H, C, P, N$.

Next apply Definition 646.184. The excluded quotient, host-limit, choice, and completeness coordinates are not carried by the tuple, so they cannot become formal target rows.

Finally use Definition 646.185 and Definition 646.186. They show that the displayed rows are sufficient for readback and seal-facing consumers, leaving no remaining source coordinate for the formal boundary. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootFormalLeanTargetBoundar`

Theorem 646.188 (DiagonalLimitCompatibility root formal kernel scope)**.** *In the root formal handoff for* DiagonalLimitCompatibility$^{\uparrow}$*, the row B is the* CauchyDiagonalBudget$^{\uparrow}$ *row, T is the* DiagonalTailSelector$^{\uparrow}$ *row, W is the* StreamName$^{\uparrow}$ *row, R is the* RegSeqRat$^{\uparrow}$ *row, L is the* Real$^{\uparrow}$ *seal row, H is the displayed* hsame *transport row, C is the displayed* Cont *route row, P is the* Pkg *provenance row, and N is the local* NameCert *row. Together these rows form the finite* BHist *kernel scope for the handoff, and no root formal target for this site may replace them by a host sequence, host real, external selector family, quotient stream, or ambient completeness principle.*

Proof. Read Definition 646.182 componentwise. It assigns the five source rows to CauchyDiagonalBudget$^{\uparrow}$, DiagonalTailSelector$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$, and assigns the support rows to hsame, Cont, Pkg, and NameCert.

The route theorem Definition 646.183 then places those rows inside the accepted root formal consumer route. The seal factorization theorem Definition 646.186 identifies the same order before the local support rows are consumed.

Apply non-escape from Definition 646.184. Since host sequences, host reals, external selector families, quotient streams, and ambient completeness principles are absent from the BHist tuple, the kernel scope is exactly the displayed row list. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root formal kernel scope`

Theorem 646.189 (DiagonalLimitCompatibility root formal audit handoff)**.** *An audit consumer for the* DiagonalLimitCompatibility$^{\uparrow}$ *root formal handoff may accept only the finite route*

$$B \longrightarrow T \longrightarrow W \longrightarrow R \longrightarrow L \longrightarrow H, C, P, N$$

over the source tuple of Definition 646.182. The audit surface rejects quotient streams, host limits, countable choice, ambient completeness, and any consumer route whose seal or readback bypasses one of the displayed rows.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootFormalAuditSurface`

Proof. Begin with the accepted route. By Definition 646.183, an audit read that belongs to this root handoff must pass through the displayed tuple.

Seal-facing reads are handled by Definition 646.186. They pass through B, T, W, R, L before the local H, C, P, N rows are consumed, so a bypassing seal route is not an accepted route for this site.

Readback-facing reads are handled by Definition 646.185. Once the displayed rows are fixed, the local readback is fixed as well. The non-escape theorem Definition 646.184 removes quotient streams, host limits, countable choice, and ambient completeness from the audit handoff. □

Theorem 646.190 (Diagonal-limit compatibility budget-selector carrier)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Any completion-facing read that uses K as a diagonal compatibility source consumes the* FiniteObservationBudgetSelector$^{\uparrow}$ *route before the* StreamName$^{\uparrow}$ *window row W, the* RegSeqRat$^{\uparrow}$ *readback row Q, and the* Real$^{\uparrow}$ *seal row L are made available to the downstream consumer.*

Proof. Project the accepted compatibility packet. The carrier displays the diagonal source, triangle row, seal route, dyadic ledger, stream-window row, regular readback, Real seal, transports, continuations, provenance, and local naming row.

Insert the finite observation-budget selector at the point where the diagonal route asks for its selected observation budget. The selector route supplies the finite budget-to-schedule passage that determines which window and dyadic rows may be read.

Only after that selector route is fixed may the consumer read W, then the dyadic and regular-rational handoff through D, Q, and finally the Real seal row L. Thus the selector is consumed before the window, readback, and seal rows are exported. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSelectorCarrier`

Theorem 646.191 (Diagonal-limit compatibility budget-selector route)**.** *For an accepted* DiagonalLimitCompatibili *packet, the displayed budget-selector handoff is mediated only by* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The handoff uses the same finite support rows H, C, P, N that expose the diagonal route, so the selector read cannot introduce an additional analytic coordinate between the budget request and the* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows.*

Proof. Begin with the support coordinates of the compatibility packet. The rows H, C, P, N record componentwise history transport, continuation replay, package provenance, and local naming data.

The finite observation-budget selector is read as a BEDC packet through those same support coordinates. Its budget-to-schedule, finite-window, dyadic-ledger, handoff, and seal rows are therefore reached by displayed BHist rows, transported by hsame, replayed by Cont, and packaged by Pkg under the local NameCert surface.

No further mediator is present in the carrier. Hence the selector handoff is exhausted by BHist, hsame, Cont, Pkg, and NameCert. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSelectorRoute`

Theorem 646.192 (Diagonal-limit compatibility budget-selector non-escape)**.** *No quotient stream, choice-selected limit, host real equality, or ambient completeness coordinate can enter an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet through the budget-selector handoff. Every such handoff remains inside the displayed* FiniteObservationBudgetSelector$^{\uparrow}$ *route and the compatibility rows $R, T, S, D, W, Q, L, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityBudgetSelectorNonescape`

Proof. Use the selector route theorem to identify the only mediation rows. They are structural BEDC rows and contain no quotient stream, choice datum, host equality test, or completeness oracle.

Use the carrier inventory for the compatibility packet. Its analytic-facing rows are the displayed dyadic ledger, stream-window family, regular readback, and Real seal reached along the named finite route.

Since the selector handoff has no other coordinate and the compatibility packet has no hidden analytic field, none of the excluded objects can enter through the handoff. Every exported read remains inside the displayed selector route and the compatibility packet rows. □

Theorem 646.193 (Diagonal-limit compatibility selector-budget source lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C,$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every selector-budget read of K is sourced only by the displayed diagonal source row, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row L, and the structural rows H, C, P, N. The read cannot use an unlisted budget source, hidden selector input, host real equality, selected limit, quotient stream, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityCarrier selector budget source lock`

Proof. Start from Definition 646.79. It identifies the budget source of an accepted compatibility packet with the displayed diagonal route, dyadic ledger, stream window, regular-rational handoff, Real seal, transport rows, continuation rows, provenance, and local name.

Apply Definition 646.81 and Definition 646.82. The window W is read before the regular readback Q, and the Real-facing row L is reached only through the displayed D, W, Q route.

Finally apply Definition 646.83 and Definition 646.48. These boundaries remove the possible routes through hidden budget data, quotient streams, selected limits, host equality, choice data, and ambient completeness. Thus the selector budget is locked to the displayed finite source surface. □

Theorem 646.194 (Diagonal-limit selector-budget RealLimitUniqueness pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet exposing the selector-budget rows W, D, Q, L. Let*

$$U = (L_0, L_1, Q_U, S_U, D_U, R_U, V_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet whose two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows and shared comparison-window route are read through the same W, D, Q, L surface of K: S_U is supplied by W, D_U is supplied by D, Q_U is supplied by Q, and R_U is supplied by L. Suppose also that the terminal* $\mathsf{RealCompletionExactBoundaryUp}$ *packet reads the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces through that route.*

Then every uniqueness consumer factors through the displayed diagonal selector-budget route of K before reading V_U. No accepted pullback may insert quotient streams, selected limits, host real equality, countable choice, independent completion data, ambient completeness, or a second completion-facing source.

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitSelectorBudgetRealLimitUniquenessPullbac`

Proof. First lock the diagonal source. Apply Definition 646.193 and Definition 646.198. These force every selector-budget read of K through the displayed diagonal source and the rows W, D, Q, L, H, C, P, N.

Next attach the uniqueness seal. The two-seal exposure theorem, Definition 1039.7, places L_0 and L_1 before any uniqueness read. The shared comparison window, Definition 1039.8, then routes the comparison through Q_U, S_U, D_U, R_U, V_U, which is the same surface as Q, W, D, L under the hypothesis.

Now read the terminal completion boundary. By Definition 1120.6, the completion-facing faces are exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows exposed by the accepted boundary. Since those faces are the same W, Q, D, L route, the uniqueness consumer has no independent completion-facing source.

Finally combine diagonal source lock with the RealLimitUniqueness nonescape and consumer handoff results implicit in the two-seal and shared-window route. The accepted route has no coordinate for quotient streams, selected limits, host real equality, countable choice, independent completion data, ambient completeness, or a second completion-facing source, so every uniqueness consumer factors through the displayed diagonal selector-budget route. □

Theorem 646.195 (Diagonal-limit compatibility seal-synchronizer budget handoff)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose seal route is consumed through an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet. The synchronizer handoff is downstream of the same finite selector-budget prefix of Definition 646.193: the diagonal source, W, D, Q, L, and the structural rows H, C, P, N are fixed before the synchronizer may export a seal-facing read.*

Proof. First use Definition 646.79 and Definition 646.193. They fix the finite prefix that any selector-budget read must expose before the completion-facing route can inspect the readback and seal rows.

Next use the existing seal-budget synchronizer handoff. The synchronizer rows are inserted at the displayed seal route and must be consumed before the regular-rational row Q and the Real seal row L escape to a consumer.

Apply Definition 646.81, Definition 646.82, Definition 646.83, and Definition 646.48. These keep the synchronizer attached to the same D, W, Q, L, H, C, P, N surface, so the handoff is downstream of the selector-budget prefix and has no separate source through host limits, quotient streams, host equality, choice data, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealSynchronizerBudgetHando`
↪ variant `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier seal budget synchronizer handoff`

Theorem 646.196 (Diagonal-limit compatibility completion-consumer budget non-escape)**.** *Every* $\mathsf{Completion}^{\uparrow}$*-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet that uses the selector-budget route receives only the finite route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The consumer cannot bypass that route by quotient streams, selected limits, host equality, choice data, independent completion data, or ambient completeness.

Proof. Begin with Definition 646.79, Definition 646.193, and Definition 646.195. Together they place the selector read, the seal synchronizer, the stream window, the dyadic ledger, the regular readback, and the Real seal on one finite prefix.

Now apply Definition 646.81, Definition 646.82, and Definition 646.83. They exhaust downstream $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and completion-seal consumers by the displayed budget, window, diagonal, seal, transport, route, provenance, and local-name rows.

Finally apply Definition 646.48. Since every completion-facing consumer already factors through the same listed packet coordinates, there is no remaining coordinate from which quotient streams, selected limits, host equality, choice data, independent completion data, or ambient completeness can be exported. □

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCompletionConsumerBudgetNon
Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier completion consumer budget nonescape

Theorem 646.197 (Diagonal-limit selector-budget formal target lock)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Any formal target for the selector-budget route of K must keep the source locked to the displayed diagonal source, the* StreamName$^{\uparrow}$ *window row W, the* DyadicRatCore$^{\uparrow}$ *ledger row D, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *seal row L, and the structural rows H, C, P, N. It may not replace that route by quotient streams, selected limits, host equality, choice data, independent completion data, ambient completeness, or a second completion-facing source.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget formal target lock

Proof. First apply Definition 646.193. It fixes every selector-budget read to the displayed diagonal source together with the W, D, Q, L, H, C, P, N rows.

Next apply Definition 646.195. The seal-facing synchronizer is downstream of that same finite selector-budget prefix, so the route cannot acquire a second seal source before readback.

Finally apply Definition 646.196. Every completion-facing or Real$^{\uparrow}$-facing consumer is exhausted by $R, T, S, D, W, Q, L, H, C, P, N$, which rules out quotient streams, selected limits, host equality, choice data, independent completion data, and ambient completeness for the formal target boundary. □

Theorem 646.198 (Diagonal-limit selector-budget route determinacy)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Any selector-budget route of K consumed by a downstream* Completion$^{\uparrow}$ *or* Real$^{\uparrow}$ *reader is the displayed diagonal route: it is determined by the diagonal source, the* StreamName$^{\uparrow}$ *window row W, the* DyadicRatCore$^{\uparrow}$ *ledger row D, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *seal row L, and the structural rows H, C, P, N. No second selector-budget route through quotient streams, selected limits, host equality, choice data, independent completion data, ambient completeness, or a hidden seal source can serve the same downstream reader.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget route determinacy

Proof. Start with Definition 646.193. It confines every selector-budget read of K to the displayed diagonal source and the finite rows W, D, Q, L, H, C, P, N.

Next apply Definition 646.195. The seal-facing handoff sits downstream of the same finite selector-budget prefix, so a downstream reader cannot insert an alternate seal route between the readback row Q and the Real$^{\uparrow}$ seal row L.

Finally apply Definition 646.196. Every Completion$^{\uparrow}$-facing or Real$^{\uparrow}$-facing consumer using the selector budget is exhausted by the displayed packet coordinates. Hence two such consumer routes with the same displayed diagonal source and the same rows W, D, Q, L, H, C, P, N have no coordinate on which to differ, and the selector-budget route is determined by that finite surface. □

Theorem 646.199 (Diagonal-limit selector-budget Real-seal nonescape)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every selector-budget* Real$^{\uparrow}$*-seal consumer of K reaches the seal row L only through the displayed diagonal source together with the* StreamName$^{\uparrow}$ *window row W, the* DyadicRatCore$^{\uparrow}$ *ledger row D, the* RegSeqRat$^{\uparrow}$ *readback row Q, and the structural rows H, C, P, N. The consumer has no route through quotient streams, selected limits, host equality, choice data, independent completion data, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget real seal nonescape`

Proof. Start with Definition 646.193. It fixes every selector-budget read of K to the displayed diagonal source and the rows W, D, Q, L, H, C, P, N.

Apply Definition 646.198. Any downstream $\mathsf{Real}^{\uparrow}$-facing consumer using that budget is therefore the same diagonal route, not a second route through hidden selector or completion data.

Finally use the Real-facing side of Definition 646.196. Since the only route to L is already the displayed finite route, quotient streams, selected limits, host equality, choice data, independent completion data, and ambient completeness have no remaining coordinate from which to enter the seal consumer. □

Theorem 646.200 (Diagonal-limit selector-budget root public readback exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. The root public readback route for the selector-budget surface is exhausted by the displayed diagonal source and the rows W, D, Q, L, H, C, P, N. In particular, every public readback of the selector budget factors through the selector-budget source lock, the selector-budget route determinacy boundary, and the selector-budget* $\mathsf{Real}^{\uparrow}$*-seal nonescape boundary before it can expose a root-facing Real or Completion consumer.*

Proof. First apply Definition 646.193. This identifies the public readback source as the displayed diagonal source and the finite selector rows W, D, Q, L, H, C, P, N.

Next apply Definition 646.198. The public readback cannot branch into a second selector-budget route after the source has been fixed.

Finally apply Definition 646.199. The Real-seal endpoint of the public route reaches L only through the same diagonal source and the same structural rows. Thus the root public readback is exhausted by the stated finite surface and cannot export quotient streams, selected limits, host equality, choice data, independent completion data, ambient completeness, or hidden seal data. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget root public readback exhaustion`

Theorem 646.201 (Diagonal-limit selector-budget Cauchy limit functor pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$F = (S_F, T_F, M_F, W_0, W_1, R_F, D_F, K_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet whose source and target seal rows are read through the same selector-budget surface D, W, Q, L of K: the functor windows W_0, W_1 are supplied by W, the readback R_F is supplied by Q, the ledger D_F is supplied by D, and the Real-facing endpoint row E_F is supplied by L.*

Then every $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing functor transport of F factors through the displayed diagonal selector-budget route of K. Conversely, that selector-budget route supplies exactly the finite source, window, readback, dyadic ledger, classifier, endpoint, transport, replay, provenance, and local naming rows required by the public seal export of F:*

$$S, D, W, Q, L, H, C, P, N \quad \text{and} \quad S_F, W_0, W_1, R_F, D_F, K_F, E_F, H_F, C_F, P_F, N_F.$$

No ambient limit object, hidden selector, quotient stream equality, host real equality, countable-choice selector, independent completion object, or second public transport route is available at the pullback boundary.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget cauchy limit functor pullback`

Proof. First lock the diagonal side. Apply Definition 646.193 and Definition 646.197. These force the accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ packet to expose only the diagonal source together with D, W, Q, L, H, C, P, N, and they prevent a formal target from replacing that route by quotient streams, selected limits, host equality, choice data, independent completion data, ambient completeness, or a second source.

Next identify the downstream route. By Definition 646.198, any $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{Completion}^{\uparrow}$-facing consumer using the selector budget is the displayed diagonal route. Since the functor rows W_0, W_1, R_F, D_F, E_F are read through W, Q, D, L, any functor transport that reaches a Real or Completion consumer has already entered that same route.

Now read the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ public surface. The NameCert obligations of Definition 1116.3 provide exactly the functor carrier, source-to-target seal exposure, finite observations, completion-consumer rows, and structural rows. The Real-completion consumer boundary of Definition 1116.7 keeps those rows inside the same finite transport route and removes detached completion data, ambient completeness, host real equality, quotient stream equality, selected limits, countable choice, and second routes.

Finally apply Definition 1116.9. Its reusable public surface is the finite route through source seal, transport map, stream windows, readback, dyadic ledger, classifier, target seal, endpoint, componentwise transport, replay, provenance, and local naming. Under the shared D, W, Q, L read, those requirements are supplied by the displayed selector-budget rows of K together with the functor rows of F, so the pullback gives exactly the stated public-seal transport boundary and no additional route. □

Theorem 646.202 (Diagonal-limit selector-budget finite-tail-filter handoff)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P,$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$F = (S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet whose stream-window row, dyadic-ledger row, regular-readback row, tail-budget and common-tail rows, and* $\mathsf{Real}^{\uparrow}$ *seal row are read through the same displayed selector-budget surface* W, D, Q, L *of* K*:* S_F *is supplied by* W*,* D_F *is supplied by* D*,* R_F *is supplied by* Q*, the rows* B_F, Q_F *are consumed through the same finite tail surface, and* E_F *is supplied by* L*.*

Then every $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing finite-tail-filter consumer of* F *factors through the displayed diagonal selector-budget route of* K*. Conversely, the selector-budget route supplies exactly the finite cofinal-tail rows required by* $\mathsf{FiniteTailFilter}^{\uparrow}$*:*

$$S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F$$

read through

$$S, D, R, W, Q, L, H, C, P, N.$$

No accepted handoff may insert a hidden tail selector, detached dyadic ledger, quotient stream equality, selected limit, host real equality, countable-choice selector, independent completion object, or ambient completeness witness between the diagonal selector budget and the finite-tail-filter consumer.

Proof. First lock the diagonal route. Apply Definition 646.193 and Definition 646.197. These force the accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ packet to expose the diagonal source together with W, D, Q, L, H, C, P, N, and they prevent a formal target from replacing that surface by hidden selector data or an independent completion-facing source.

Next identify the consumer route. By Definition 646.198, any $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{Completion}^{\uparrow}$-facing consumer using the selector budget is the displayed diagonal route. Apply Definition 646.196 and Definition 646.48 to remove quotient streams, selected limits, host equality, choice data, independent completion data, and ambient completeness from that route.

Now read the finite-tail-filter surface. Tail-window exactness, Definition 1119.4, identifies the windows and dyadic tolerances used by the tail-budget row, the common-tail comparison row, and the seal-facing row. Since S_F, D_F, R_F, E_F are read through W, D, Q, L, the finite-tail rows B_F, Q_F have no separate selector or tolerance ledger from which to build another consumer path.

Finally use Definition 1119.5 and Definition 1119.9. The Real and Completion seal reads of F pass through the finite cofinal-tail packet $S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F$, while sibling consumers cannot substitute their own tail selector, ledger, common-tail row, or seal row. Under the shared W, D, Q, L read, those rows are exactly the displayed selector-budget rows of K together with the structural rows H, C, P, N, so the handoff factors through the diagonal route and supplies precisely the finite-tail-filter packet. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitSelectorBudgetFiniteTailFilterHandoff`

Theorem 646.203 (Diagonal-limit selector-budget regular-tail-meet handoff)**.** *Let $K = (R, T, S, D, W, Q, L, H, C,$ be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ packet. Suppose its displayed selector-budget surface is consumed by an accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold packet and then by either an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ packet or a $\mathsf{Real}^{\uparrow}$ seal consumer. The resulting route factors through the displayed diagonal source, the rows W, D, Q, L, the shared tail-meet threshold, and the structural rows H, C, P, N.*

No accepted handoff may insert a hidden tail selector, quotient stream, host equality, selected limit, countable-choice selector, independent completion object, or ambient completeness row between the diagonal selector budget, the shared threshold, and the finite-tail-filter or Real seal consumer.

Proof. First lock the diagonal source. Apply Definition 646.193 and Definition 646.198. They force every selector-budget consumer to read the displayed diagonal route through W, D, Q, L, H, C, P, N and leave no second route through quotient streams, selected limits, host equality, choice data, independent completion data, ambient completeness, or hidden seal data.

Next read the shared-threshold surface of chapter 560. The regular tail-meet packet supplies the common tail threshold as a displayed finite row, so the selector budget reaches downstream seal consumers only after that threshold has been named in the same finite route.

Finally compose with Definition 646.202. The finite-tail-filter handoff already routes Real-facing and Completion-facing consumers through the displayed selector-budget rows and the finite cofinal-tail packet. With the regular tail-meet threshold placed upstream of that handoff, the composite route has exactly the stated diagonal source, W, D, Q, L, shared threshold, and structural rows, and no hidden selector or ambient completion row remains available. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget regular tail meet handoff`

Theorem 646.204 (Diagonal-limit selector-budget tail-meet pullback exhaustion)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ packet. Suppose the*

displayed selector-budget surface of K is consumed by an accepted RegularCauchyTailMeet$^{\uparrow}$ *packet with shared tail-meet threshold τ, and that the same threshold-facing route is then consumed by an accepted* FiniteTailFilter$^{\uparrow}$ *packet*

$$F = (S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F).$$

Every Completion$^{\uparrow}$*-facing or* Real$^{\uparrow}$*-facing consumer of this composite packet has exactly one route: the displayed diagonal source of K, the rows W, D, Q, L, the shared threshold τ, the finite-tail-filter rows $S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F$, and the structural rows H, C, P, N.*

No accepted pullback along this route may add a hidden selector, quotient stream, host equality row, selected limit, countable-choice selector, independent completion datum, second completion source, or ambient completeness witness.

Proof. First lock the diagonal selector source. Apply Definition 646.193 and Definition 646.198. These identify the only downstream selector-budget route as the displayed diagonal source together with W, D, Q, L, H, C, P, N.

Next place the shared threshold on that route. The shared-threshold surface of chapter 560, together with Definition 560.14 and Definition 560.56, makes the threshold τ a displayed finite row reached through the same selector windows and readback rows. A second threshold route would require a hidden selector or an independent tail oracle, neither of which is carried by the accepted packet.

Now attach the finite-tail-filter consumer. By Definition 646.202, Real-facing and Completion-facing consumers of F read the finite packet rows $S_F, D_F, R_F, B_F, Q_F, E_F, H_F, C_F, P_F, N_F$ only through the displayed selector-budget surface W, D, Q, L of K.

Finally compose with Definition 646.203. The regular tail-meet handoff already places τ between the diagonal selector budget and the finite-tail-filter or Real seal consumer. Combining the two handoffs gives one composite route with the diagonal source, W, D, Q, L, the shared threshold, the finite-tail-filter packet rows, and H, C, P, N as its complete coordinate list. Since the source lock and determinacy theorems have no remaining carrier coordinate for quotient streams, host equality, selected limits, countable-choice selectors, independent completion data, second completion sources, or ambient completeness, the pullback route is exhausted. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget tail meet pullback exhaustion`

Theorem 646.205 (Diagonal-limit selector-budget TailCofinalitySchedule pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet, and let $A = (E_A, W_A, D_A, R_A, L_A, H_A, C_A, P_A, N_A)$ be an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet. Suppose the schedule packet consumes the same* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *readback, and* Real$^{\uparrow}$ *seal rows through the selector-budget surface of K: W_A is supplied by W, D_A by D, R_A by Q, and L_A by L.*

Then every completion-facing or Real-facing consumer of the combined packet factors through the shared finite route

$$W, D, Q, L, H, C, P, N \quad \textit{read as} \quad W_A, D_A, R_A, L_A, H_A, C_A, P_A, N_A.$$

No hidden selector, quotient stream, host equality, selected limit, independent completion datum, ambient completeness witness, or second schedule budget can intervene.

Proof. First lock the diagonal selector source. Apply Definition 646.193 and Definition 646.198. They identify the only selector-budget route as the displayed diagonal source together with W, D, Q, L, H, C, P, N.

Next use Definition 646.204. It removes any alternate tail-facing route between the selector budget, shared threshold, finite-tail packet, and downstream Real or Completion consumer.

Now attach the schedule packet. By Definition 681.3 and Definition 681.9, the accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ consumer reads the window, ledger, regular-rational readback, and Real seal through the finite schedule route before the completion seal can be inspected.

The hypotheses identify those schedule rows with W, D, Q, L from K. Hence every combined consumer has the single shared route listed in the statement, and the non-escape clauses from the diagonal selector and schedule boundaries leave no coordinate for a hidden selector, quotient stream, host equality, selected limit, independent completion datum, ambient completeness witness, or second schedule budget. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget tail cofinality schedule pullback`

Theorem 646.206 (Diagonal-limit selector-budget UniformCauchyCriterion terminal pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose shared window, dyadic ledger, regular readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are read through the same selector-budget surface W, D, Q, L of K: the window family W_U is supplied by W, the dyadic tolerance row Q_U is supplied by D, the regular tail readback T_U is supplied by Q, and the terminal seal row E_U is supplied by L.*

Then every terminal $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing consumer of U factors through the displayed diagonal selector-budget route of K. Conversely, that route supplies the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*, and* $\mathsf{RealCompletionExactBoundaryUp}$ *faces needed by the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *terminal envelope:*

$$W, D, Q, L, H, C, P, N \quad \textit{read as} \quad W_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U.$$

No accepted terminal route may insert quotient streams, host equality, selected limits, choice data, independent completion data, host metric data, ambient completeness, or a second terminal Real or Completion source.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget uniform criterion pullback`

Proof. First lock the diagonal side. Apply Definition 646.193 and Definition 646.198. These identify the only selector-budget route of K as the displayed diagonal source together with W, D, Q, L, H, C, P, N.

Next exclude detached terminal data. By Definition 646.196, any $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{Completion}^{\uparrow}$-facing consumer using that route is exhausted by the same finite packet rows and has no coordinate for quotient streams, selected limits, host equality, choice data, independent completion data, or ambient completeness.

Now attach the finite-family terminal envelope. The seal envelope of Definition 435.108, the completion–Real correspondence of Definition 435.109, and finite-window coverage in Definition 435.111 require the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, Real-window budget, and exact Real-completion boundary before a terminal consumer can read the seal.

Under the shared W, D, Q, L surface, those UniformCauchyCriterion rows are precisely the selector-budget rows of K together with the structural rows H, C, P, N. Hence every admitted terminal consumer factors through the displayed diagonal route, and reading the same route backward supplies exactly the terminal envelope faces without adding host metric data, an independent completion object, or an ambient completeness principle. □

Theorem 646.207 (Diagonal-limit uniform-criterion terminal uniqueness pullback)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet, and let*

$$V = (L_0, L_1, Q_V, S_V, D_V, R_V, V_V, H_V, C_V, P_V, N_V)$$

be an accepted $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet. Suppose the shared selector-budget surface* W, D, Q, L *of* K *supplies the UniformCauchyCriterion rows* W_U, Q_U, T_U, E_U *and also supplies the RealLimitUniquenessSeal rows* S_V, D_V, Q_V, R_V*. Suppose further that the terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* U *is the consumer handed to the uniqueness seal through the same displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces.*

Then that terminal Real or Completion consumer has exactly one finite pullback route: the displayed diagonal selector-budget rows W, D, Q, L, H, C, P, N *of* K*, read as the UniformCauchyCriterion terminal rows and as the RealLimitUniquenessSeal comparison rows. No accepted terminal pullback through this route may add quotient streams, selected limits, host real equality, countable choice, independent completion data, ambient completeness, or a second completion-facing source.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitUniformCriterionTerminalUniquenessPullba`

Proof. First apply Definition 646.206. It places every terminal $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{Completion}^{\uparrow}$-facing consumer of U on the displayed diagonal selector-budget route of K and reads W, D, Q, L, H, C, P, N as the UniformCauchyCriterion terminal surface.

Next apply Definition 646.194. It places the RealLimitUniquenessSeal comparison route on the same displayed selector-budget rows and rules out quotient streams, selected limits, host real equality, countable choice, independent completion data, ambient completeness, and a second completion-facing source.

Finally compare the two consumers. By Definition 646.198, any consumer that uses the selector budget of K is the displayed finite route rather than a parallel route through hidden selector or completion data. By Definition 646.196, a terminal Real or Completion consumer using that route has no remaining coordinate from which an independent completion object, host equality, selected limit, quotient stream, countable-choice selector, ambient completeness witness, or second completion-facing source could be exported. Thus the UniformCauchyCriterion terminal consumer and the uniqueness-seal consumer collapse to the same finite pullback route. □

Theorem 646.208 (Diagonal-limit selector-budget Real and Completion exit route)**.** *Let* $K = (R, T, S, D, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Let*

$$U = (I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted finite-family $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet whose window, dyadic tolerance, regular readback, and local* $\mathsf{Real}^{\uparrow}$ *seal rows consume the same* W, D, Q, L *rows of* K*. Then every Phase Real and Completion exit read admitted by* U *factors through the finite surface*

$$R, T, S, D, W, Q, L, H, C, P, N \quad \text{and} \quad I_U, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U,$$

where the displayed diagonal source supplies the selector-budget route, the DyadicRatCore$^{\uparrow}$ *ledger supplies the tolerance row, the* StreamName$^{\uparrow}$ *family supplies the window row, the* RegSeqRat$^{\uparrow}$ *readback supplies the regular tail row, the local* Real$^{\uparrow}$ *seal supplies the terminal seal row, and the* UniformCauchyCriterion$^{\uparrow}$ *envelope supplies the finite-family terminal boundary.*

The same exit read preserves the structural hsame, Cont, Pkg, *and* NameCert *rows. It has no accepted coordinate for quotient streams, host real equality, selected limits, countable-choice selectors, independent completion data, host metric data, ambient completeness, or a second Real or Completion source.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget uniform real exit`

Proof. First lock the diagonal source. Apply Definition 646.193 and Definition 646.198. These force every Real-facing or Completion-facing read that touches the selector budget to use the displayed diagonal source of K and the rows W, D, Q, L, H, C, P, N.

Next remove escape data from the completion side. Apply Definition 646.196 and Definition 646.206. The first result leaves no coordinate for quotient streams, selected limits, host equality, countable-choice data, independent completion data, or ambient completeness. The second result identifies the UniformCauchyCriterion terminal pullback with the same W, D, Q, L surface.

Now read the terminal envelope facts for UniformCauchyCriterion$^{\uparrow}$. The seal envelope of Definition 435.108, the completion–Real correspondence of Definition 435.109, and finite-window coverage in Definition 435.111 require the StreamName$^{\uparrow}$ window family, DyadicRatCore$^{\uparrow}$ tolerance ledger, RegSeqRat$^{\uparrow}$ tail readback, local Real$^{\uparrow}$ seal, and finite-family terminal boundary before an exit read can reach the public Real or Completion surface.

Finally compose the two boundaries. Under the shared consumption hypothesis, W_U, Q_U, T_U, E_U are the reads of W, D, Q, L, while H_U, C_U, P_U, N_U preserve the same hsame, Cont, Pkg, and NameCert rows as H, C, P, N. Thus every Phase Real and Completion exit read factors through the displayed diagonal source and the finite UniformCauchyCriterion envelope, with no remaining public coordinate for host metric data, ambient completeness, independent completion data, or a second Real or Completion source. □

Theorem 646.209 (Diagonal-limit selector-budget finite-observation coverage)**.** *Let $K = (R, T, S, D, W, Q, L, H, C$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet, and let the terminal Real-and-Completion exit route of K be read through a displayed* FiniteObservationBudgetSelector$^{\uparrow}$ *selector budget*

$$F = (B_F, S_F, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F).$$

If the accepted exit route consumes the selector budget of F, then its finite observation surface is exactly the exported selector surface: the budget row B_F, selected schedule S_F, finite StreamName$^{\uparrow}$ *windows W_F,* DyadicRatCore$^{\uparrow}$ *ledger D_F,* RegSeqRat$^{\uparrow}$ *handoff R_F, terminal* Real$^{\uparrow}$ *seal E_F, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows H_F, C_F, P_F, N_F.*

The same surface is the one consumed by the UniformCauchyCriterion$^{\uparrow}$ *pullback and the* CauchyLimitSeal$^{\uparrow}$ *route. No accepted reader along this terminal route may insert a quotient stream, host real equality, selected host limit, countable-choice selector, ambient completeness witness, metric-completeness row, arithmetic package, order package, or bridge row.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier selector budget finite observation coverage`

Proof. First lock the diagonal source. Apply Definition 646.193. Any terminal Real-and-Completion read that reaches the selector budget of K must pass through the displayed diagonal source and the rows W, D, Q, L, H, C, P, N.

Next identify the selector budget surface. The public export theorem Definition 782.14 exposes exactly the finite route $B_F, S_F, W_F, D_F, R_F, E_F$ together with H_F, C_F, P_F, N_F. The diagonal consumer boundary Definition 782.10 makes that exported surface the one seen by diagonal-limit and Real-completion consumers, and terminal route endpoint exactness Definition 782.29 fixes the terminal endpoint at the displayed seal E_F through the same BHist, hsame, Cont, Pkg, and NameCert rows.

Now compare the downstream readers. The UniformCauchyCriterion pullback Definition 646.206 reads the shared window, dyadic ledger, regular handoff, and Real seal rows through the same selector-budget surface. The CauchyLimitSeal route is a seal-facing reader of the same public selector packet, so the diagonal consumer boundary places it on the exported finite route rather than on an additional completion or bridge coordinate.

Finally apply the Real-and-Completion exit theorem Definition 646.208. Its exclusion list removes quotient streams, host real equality, selected limits, countable-choice selectors, host metric data, ambient completeness, independent completion data, and second Real or Completion sources. Since the public selector export also carries no metric-completeness row, arithmetic package, order package, or bridge row, the terminal route has exactly the finite observation surface listed in the statement. □

Theorem 646.210 (Diagonal-limit selector-budget finite-prefix-budget pullback)**.** *Let $K = (R, T, S, D, W, Q, L, H$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Let*

$$B = (Q_B, B_B, W_B, D_B, R_B, S_B, E_B, H_B, C_B, P_B, N_B)$$

be an accepted FinitePrefixLimitBudget$^{\uparrow}$ *packet whose window, dyadic ledger, regular readback, and terminal seal rows are read through the same selector-budget surface W, D, Q, L of K. Then every Real-and-Completion terminal consumer that passes through B factors through exactly the finite route*

$$W, D, Q, L, H, C, P, N$$

read as

$$W_B, D_B, R_B, S_B, E_B, H_B, C_B, P_B, N_B.$$

No accepted route may insert a quotient stream, host real equality, selected limit, countable-choice selector, ambient Real$^{\uparrow}$ *completeness, metric-completeness row, arithmetic package, order package, or bridge row between the selector-budget surface and the terminal Real or Completion read.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetFinitePrefixB`

Proof. First lock the diagonal source. Apply Definition 646.193 and Definition 646.198. These force every terminal read that touches the selector budget of K to use the displayed rows W, D, Q, L, H, C, P, N rather than a parallel route through hidden selector data.

Next read the finite-prefix budget side. The seal ordering theorem Definition 919.3 places every seal-facing read behind the ordered route $Q_B \to B_B \to W_B \to D_B \to R_B \to S_B$. The consumer boundary Definition 919.10 and seal-consumer coverage Definition 919.15 say that a Real-facing or completion-facing consumer receives only the finite-prefix rows $Q_B, B_B, W_B, D_B, R_B, S_B, E_B, H_B, C_B, P_B, N_B$.

Under the shared-surface hypothesis, the rows W_B, D_B, R_B, S_B are precisely the finite-prefix read of W, D, Q, L, while E_B, H_B, C_B, P_B, N_B preserve the same exclusion, transport, continuation,

provenance, and local naming boundary as the displayed support rows of K. Compose this identification with Definition 646.208 and Definition 646.209 to obtain the terminal Real-and-Completion surface through the finite selector-budget route.

Finally apply the finite-prefix non-escape theorem Definition 919.12. Its exclusion list matches the diagonal exit exclusions: quotient streams, host real equality, selected limits, countable-choice selectors, hidden terminal data, and ambient completeness are absent. Since neither the selector-budget route nor the finite-prefix budget carrier contains a metric-completeness row, arithmetic package, order package, or bridge row, the pullback is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitSelectorBudgetFinitePrefixBudgetPullback`

Theorem 646.211 (DiagonalLimitCompatibility root selector budget lock)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every accepted root-selector packet routes the displayed source budget through the selector row, the seal-synchronizer handoff, the* BHist *inventory, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance row, and the local* NameCert *row before any diagonal-limit read reaches* D, W, Q, L*. No selector read can bypass that route by a host limit, quotient stream, host real equality, chosen completion point, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier root selector budget lock`

Proof. Project the accepted packet through Definition 646.1. The carrier exposes the diagonal source, triangle and selector-facing budget rows, dyadic ledger, stream window, regular readback, Real seal, transport rows, continuation routes, package provenance, and local name as one finite BHist surface.

Apply Definition 646.193. It identifies the selector-budget source with the displayed diagonal route, window row, dyadic ledger, regular readback, Real seal, and structural rows H, C, P, N.

Next use Definition 646.195. The synchronizer is read downstream of the same selector-budget prefix, so the root selector cannot obtain an independent seal source before the diagonal-limit read.

Finally apply Definition 646.153 and Definition 646.156. They place the diagonal-limit consumer on the displayed BHist/hsame/Cont/Pkg/NameCert package and remove every route through host limits, quotient streams, host equality, chosen completion points, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorBudgetLock`

Theorem 646.212 (DiagonalLimitCompatibility root selector finite-window refinement)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every root-selector read of* K *that reaches a* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *seal refines through the same finite L10 route: the selector-budget source, the* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff, the* StreamName$^{\uparrow}$ *window* W*, the* RegSeqRat$^{\uparrow}$ *readback* Q*, the* DyadicRatCore$^{\uparrow}$ *ledger* D*, the* Real$^{\uparrow}$ *seal* L*, the* CauchyLimitSeal$^{\uparrow}$ *boundary, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The refined route has no host limit, quotient stream, selected limit, host real equality, countable-choice row, or ambient completeness coordinate.*

Proof. Start with Definition 646.211. It fixes every root-selector read on the displayed selector-budget source and structural rows before the read may inspect D, W, Q, L.

Insert the seal handoff using Definition 646.138. The synchronizer is downstream of the same selector prefix, so the read cannot swap to a second source while moving toward a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal.

Next apply Definition 646.139. The stream window W, regular-rational readback Q, dyadic ledger D, and Real seal L remain on the finite route that the tail schedule exposes to the seal-facing consumer.

Finally combine Definition 646.213 with Definition 646.140. These results place the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary and every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer inside the same package rows $R, T, S, D, W, Q, L, H, C, P, N$, excluding host limits, quotient streams, selected limits, host equality, countable-choice rows, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility root selector window refinement`

Theorem 646.213 (DiagonalLimitCompatibility seal budget route exhaustion)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal consumer that reads the selector-budget route factors through the same public route*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The consumer receives the source budget, selector handoff, seal synchronizer, dyadic ledger, stream window, regular readback, Real seal, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and it cannot export quotient streams, selected limits, host equality, or ambient completeness.*

Proof. Start with Definition 646.211. It fixes the selector-budget route before a diagonal-limit or seal-facing consumer can inspect the readback rows.

Use Definition 646.196. Every completion-facing or Real-facing consumer that uses the selector-budget route is exhausted by $R, T, S, D, W, Q, L, H, C, P, N$.

Now apply Definition 646.140. The inserted synchronizer and tail-schedule route remain part of the same finite compatibility surface, so no second seal route is available.

The exclusion clause follows from Definition 646.155 and Definition 646.156. The public consumer surface contains only the listed packet rows and has no coordinate for quotient streams, selected limits, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySealBudgetRouteExhaustion`

Theorem 646.214 (DiagonalLimitCompatibility selector-lock seal route exhaustion)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Every accepted selector-budget read of* K *that reaches a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal passes through the root selector lock, the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, the* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *readback when present, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal* L *on the same displayed route. The route exports no quotient stream, selected host limit, host real equality, independent seal object, countable-choice row, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorLockSealRouteExhaus`

Proof. First combine Definition 646.211 with Definition 646.213. The first theorem fixes the upstream selector-budget prefix, and the second exhausts every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal consumer by the same finite packet $R, T, S, D, W, Q, L, H, C, P, N$.

Next insert the synchronizer with Definition 646.138. It forces the $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ request, seal, budget, selector, compatibility, transport, route, provenance, and local naming rows to be read before Q and L.

When a tail schedule is present, use Definition 646.139. The schedule readback stays on the same route through the finite window row, dyadic ledger, regular-rational readback, and Real seal.

Finally apply Definition 646.140. The composed route exports only the displayed compatibility rows, synchronizer rows, tail-schedule rows, regular-rational readback, and $\mathsf{Real}^{\uparrow}$ seal, so no quotient stream, selected host limit, host equality, independent seal object, countable-choice row, or ambient completeness coordinate can escape it. □

Theorem 646.215 (DiagonalLimitCompatibility root selector window pullback uniqueness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector reads. If the two reads share the same selector-budget source, the same* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, the same finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the same* $\mathsf{Real}^{\uparrow}$ *seal row, then their completion-facing consumer surfaces are the same finite pullback through*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

No quotient stream, selected host limit, host real equality, independent seal object, countable-choice row, or ambient completeness coordinate can distinguish the two reads.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorWindowPullbackU`

Proof. Start with Definition 646.211. It places both root-selector reads on the same selector-budget source and the same structural route before any consumer can inspect the diagonal ledger, finite window, regular readback, or Real seal.

Next use Definition 646.138 and Definition 646.139. The shared synchronizer handoff and shared finite window force the reads through the same tail-schedule surface, the same Q readback, and the same D, L seal-facing rows.

Apply Definition 646.213 and Definition 646.140. These theorems exhaust every completion-facing or $\mathsf{Real}^{\uparrow}$ consumer by the displayed package rows and exclude all routes through quotient streams, selected host limits, host equality, independent seals, countable choice, or ambient completeness.

The carrier inventory then identifies the public pullback. Since the two reads agree on the source, handoff, window, readback, dyadic ledger, and Real seal rows, and since the only remaining exported rows are H, C, P, N for transport, route, provenance, and local naming of that same packet, the completion-facing surfaces cannot diverge. □

Theorem 646.216 (DiagonalLimitCompatibility completion consumer package)**.** *The root-selector budget lock, the seal-budget route exhaustion row, the root-route budget package, and the seal-source synchronization rows assemble the public completion-facing package for any accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet:*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

This package exposes the displayed source budget, selector handoff, seal synchronizer, diagonal ledger, window, readback, Real seal, BHist, hsame, Cont, Pkg, *and* NameCert *rows, and it exposes no quotient stream, selected limit, host equality, independent completion object, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCompletionConsumerPackage`

Proof. First combine Definition 646.211 with Definition 646.213. The first theorem fixes the upstream selector route, and the second theorem exhausts the downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal consumers by the same finite package.

Next use Definition 646.153. It supplies the public route through the budget source, mesh budget, triangle bound, stream window, readback, seal route, Real seal, transport, continuation, provenance, and local naming rows.

Use Definition 646.69 and Definition 646.138. These rows attach the seal source and the synchronizer to the same route before any completion-facing read can consume the seal.

Finally apply Definition 646.48. Since every cited row is a displayed coordinate of the same accepted packet, the assembled package has no route through quotient streams, selected limits, host equality, independent completion data, or ambient completeness. □

Theorem 646.217 (DiagonalLimitCompatibility root selector route determinacy)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. Suppose the displayed classifier identifies their selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer sees the same finite completion-facing package for the two reads. No quotient stream, selected host limit, host real equality, independent seal, countable-choice row, or ambient completeness coordinate can distinguish them.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSelectorRouteDeterminac`

Proof. Apply Definition 646.211 to both packets. The lock forces both reads onto the displayed selector prefix before either consumer can inspect the stream window, regular readback, dyadic ledger, or Real seal row.

Next combine Definition 646.212 with Definition 646.213. The shared classifier data keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows on one finite route through the seal-facing package.

Use Definition 646.215. Since the displayed source, handoff, window, readback, ledger, seal, and structural rows agree, the completion-facing pullback surface is identified for the two reads.

Finally apply Definition 646.216. Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer receives only the finite package $R, T, S, D, W, Q, L, H, C, P, N$, so none of the excluded host or completion coordinates can witness a divergence at the public surface. □

Theorem 646.218 (DiagonalLimitCompatibility selector-budget lock exactness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. The root selector-budget lock classifies* K *and* K' *by the same finite surface exactly when the displayed selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows agree componentwise:*

$$K \sim_{\text{root selector budget}} K'$$
$$\iff R \sim R' \text{ and } T \sim T' \text{ and } S \sim S' \text{ and } D \sim D' \text{ and } W \sim W' \text{ and } Q \sim Q' \text{ and } L \sim L' \text{ and } H \sim H' \text{ and } C \sim C$$

Thus the displayed selector-budget, handoff, window, readback, ledger, seal, transport, continuation, provenance, and naming rows are the whole classifier surface for the accepted root read.

Proof. For the forward direction, start with Definition 646.211. The lock prevents the root read from leaving the displayed selector-budget source and structural rows before any downstream consumer inspects the seal-facing coordinates.

Apply Definition 646.215. The shared classifier surface pulls the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row back to the same root selector route. Then use Definition 646.213; every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ read is exhausted by the same finite rows, so no omitted host, quotient, choice, or ambient completeness coordinate can supply an additional classifier distinction.

For the reverse direction, assume the displayed selector-budget source, synchronizer handoff, window, readback, ledger, seal row, and structural H, C, P, N rows agree componentwise. Repack these coordinates as the finite route $R, T, S, D, W, Q, L, H, C, P, N$ for both accepted packets.

Finally invoke Definition 646.217. The repacked route forces the two accepted reads to present the same completion-facing package to every terminal consumer, so the root selector-budget classifier agrees exactly on the displayed rows. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetLockExactness`

Theorem 646.219 (DiagonalLimitCompatibility selector-budget classifier pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector reads. They induce the same completion-facing public classifier exactly when their selector budget,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, and structural* H, C, P, N *rows agree componentwise.*

The classifier has no coordinate for quotient stream equality, selected host limits, host real equality, independent seals, countable-choice rows, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetClassifierPul`

Proof. For the forward direction, use Definition 646.211. The accepted root read is locked to the displayed selector-budget source before the synchronizer, stream window, readback, ledger, seal, and structural rows are consumed.

Apply the pullback uniqueness theorem, Definition 646.215. It identifies the finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal with the same selector route whenever the displayed root surface agrees.

For the reverse direction, assume componentwise agreement of the listed rows. The selector-budget lock exactness theorem repacks those coordinates as the finite route $R, T, S, D, W, Q, L, H, C, P, N$ for both reads, and Definition 646.216 is then the completion-facing package seen by terminal consumers.

Finally use Definition 646.48. Since quotient streams, selected host limits, host real equality, independent seals, choice rows, and ambient completeness are absent from the exported package, none of them can distinguish the two accepted classifier surfaces. □

Theorem 646.220 (DiagonalLimitCompatibility selector-budget terminal pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packets. Suppose the displayed classifier identifies their selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows componentwise. Then every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer reads the same finite public pullback package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The terminal surface cannot distinguish K *from* K' *by quotient stream equality, selected host limits, host real equality, independent seals, countable-choice rows, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetTerminalPullb`

Proof. Start with Definition 646.219. The assumed componentwise agreement is exactly the classifier surface for the accepted packets, so the two reads present one completion-facing public classifier before any terminal consumer inspects the seal rows.

Next use Definition 646.215. The shared finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal, and structural rows pull both packets back to the same root-selector surface.

Now apply Definition 646.216. Its exported consumer data is precisely the finite package $R, T, S, D, W, Q, L, H, C, P, N$, so the public $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ reads receive the same pullback package from K and K'.

Finally invoke Definition 646.48. Quotient stream equality, selected host limits, host real equality, independent seals, countable-choice rows, and ambient completeness are not exported coordinates of that package. Hence no terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer can separate the accepted packets at the public surface. □

Theorem 646.221 (DiagonalLimitCompatibility terminal readback determinacy)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector reads. Suppose the displayed classifier identifies their selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows. Then every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *terminal read receives the same finite readback package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

No terminal consumer can distinguish the two reads by quotient streams, selected host limits, host real equality, countable-choice data, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibility terminal readback determinacy`

Proof. First apply Definition 646.217. It identifies the two accepted root-selector reads on the displayed selector budget, synchronizer handoff, finite window, regular readback, dyadic ledger, Real seal row, and structural transport, route, provenance, and naming rows.

Next use Definition 646.215. The shared window, readback, ledger, and seal rows pull both reads back to the same finite completion-facing surface, so the terminal read cannot obtain a second public package by changing the window or readback representative.

Apply Definition 646.213. The $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ terminal consumers are exhausted by the same route through $R, T, S, D, W, Q, L, H, C, P, N$, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary is already part of that seal-facing route.

Finally invoke Definition 646.216. The only exported readback package is the displayed finite package, so quotient streams, selected host limits, host real equality, countable-choice data, and ambient completeness have no coordinate by which to separate the two terminal reads. □

Theorem 646.222 (DiagonalLimitCompatibility selector window tail seal bridge)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector reads. Suppose the displayed classifier identifies their selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, tail-schedule readback whenever a terminal tail row is present,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows. Then every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *terminal consumer receives one finite terminal package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The selector window, synchronizer handoff, tail schedule, Cauchy limit boundary, and Real seal cannot be read as separate terminal classifier coordinates.

Proof. First apply Definition 646.212. The selector-budget source and synchronizer handoff place the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row on the same displayed root route.

Next use Definition 646.214. This makes the seal-facing route exhaustive for the accepted packets, so the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary and any present tail-schedule readback are consumed through the same finite route instead of a separate terminal channel.

Apply Definition 646.215. Since the selector window, readback, ledger, seal, boundary, and structural rows agree, both terminal reads pull back to one completion-facing surface.

Now invoke Definition 646.221. The terminal readback for $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers is already the displayed package $R, T, S, D, W, Q, L, H, C, P, N$.

Finally use Definition 646.216. Its consumer inventory exports exactly that finite package, so the tail schedule, limit-seal boundary, Real seal row, and completion consumer receive the same terminal data. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorWindowTailSealBridg`

Theorem 646.223 (DiagonalLimitCompatibility terminal tail-rebase exactness)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector reads. Suppose the displayed classifier identifies their selector-budget source,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* H, C, P, N *rows. Then every terminal read obtained after a tail rebase exposes exactly the same finite package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The tail rebase has no coordinate for quotient streams, selected host limits, host real equality, countable-choice data, or ambient completeness.

Proof. First apply Definition 646.221. The shared selector-budget source, synchronizer handoff, window, readback, ledger, seal row, and structural rows already determine the finite terminal readback package for $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers.

Next use Definition 646.218. This identifies the classifier equality with equality of the displayed finite rows, so a tail rebase cannot introduce another selector-budget source, handoff, window, readback, ledger, seal, or structural row at the terminal boundary.

Apply Definition 646.215. The finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ seal row pull the rebased read back to the same root selector surface.

Finally invoke Definition 646.216. The exported consumer package is exactly $R, T, S, D, W, Q, L, H, C, P, N$, and the package inventory omits quotient streams, selected host limits, host equality, countable choice, and ambient completeness. Therefore the tail rebase has no public coordinate by which to change the terminal package. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTerminalTailRebaseExactness`

Theorem 646.224 (DiagonalLimitCompatibility selector-budget terminal classifier)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector read. The terminal classifier exported to* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumers is the single displayed package whose source is*

the selector-budget row R, whose finite window is the $\mathsf{StreamName}^{\uparrow}$ *row S, whose rational readback is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row T, whose dyadic ledger is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, whose synchronizer handoff is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row Q, and whose seal-facing endpoint is the* $\mathsf{Real}^{\uparrow}$ *row L. No terminal consumer may read these rows as independent classifier coordinates.*

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetTerminalClass`

Proof. First apply Definition 646.212. It places the selector-budget source, finite stream window, regular rational readback, dyadic ledger, synchronizer handoff, and Real seal on one root-selector route.

Next use Definition 646.218. The classifier equality is equality of the displayed finite rows, so the terminal classifier cannot split R, T, S, D, Q, L into unrelated coordinates.

Apply Definition 646.221. The terminal readback for $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers is already the package $R, T, S, D, W, Q, L, H, C, P, N$.

Finally invoke Definition 646.216. It exports the same finite package to completion-facing readers, so the terminal classifier is the displayed selector-budget package rather than a host-side limit selector. □

Theorem 646.225 (DiagonalLimitCompatibility terminal synchronizer classifier)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root-selector read. The terminal classifier exported from K to* $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is the single finite L10 package whose selector-budget source is R, whose synchronizer handoff is Q, whose finite* $\mathsf{StreamName}^{\uparrow}$ *window is S, whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is T, whose* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger is D, whose* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary is W, whose* $\mathsf{Real}^{\uparrow}$ *seal is L, and whose structural rows are H, C, P, N.*

The synchronizer handoff and the Cauchy-limit boundary therefore agree componentwise with the same displayed package read by the Real and completion consumers. No quotient stream, selected host limit, countable-choice row, ambient completeness coordinate, or second terminal classifier participates in the exported surface.

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTerminalSynchronizerClassif`

Proof. Begin with Definition 646.224. It identifies the terminal classifier as the displayed selector-budget package rather than as independent Real, completion, or limit-seal coordinates.

Apply Definition 646.222. The finite window, regular readback, dyadic ledger, synchronizer handoff, Cauchy-limit boundary, Real seal, and structural rows are read through the same terminal package.

Next use Definition 646.221. Once those rows are componentwise fixed, every $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ terminal read receives $R, T, S, D, W, Q, L, H, C, P, N$ and no alternate readback surface.

Finally invoke Definition 646.226. The route inventory omits quotient streams, selected host limits, countable choice, and ambient completeness, so the synchronizer and Cauchy-limit rows remain bound to the same finite L10 package exported by K. □

Theorem 646.226 (DiagonalLimitCompatibility Real completion route non-escape)**.** *Every* DiagonalLimitCompati *route from a selector-budget root read to a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows displayed in*

$$K = (R, T, S, D, W, Q, L, H, C, P, N).$$

The route has no coordinate for a host-selected real limit, quotient stream equality, countable-choice data, ambient completeness, or a completion consumer that bypasses the selector-budget classifier.

Proof. Begin with Definition 646.224. The terminal classifier presented to the Real and completion consumers is the single finite selector-budget package.

Use Definition 646.223. A tail rebase preserves the same finite package and does not create quotient streams, selected host limits, host real equality, countable-choice data, or ambient completeness.

Next apply Definition 646.214. The seal-facing route is exhausted by the displayed stream, rational, dyadic, synchronizer, limit-seal, and Real rows.

Finally invoke Definition 646.216. Completion consumers receive only the displayed package, so any route bypassing the selector-budget classifier is outside the accepted DiagonalLimitCompatibility$^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCarrier real completion route nonescape`

Theorem 646.227 (DiagonalLimitCompatibility tail-seal synchronizer pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

and

$$K' = (R', T', S', D', W', Q', L', H', C', P', N')$$

be accepted DiagonalLimitCompatibility$^{\uparrow}$ *root-selector reads whose terminal classifiers agree on selector budget,* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger,* CauchySealBudgetSynchronizer$^{\uparrow}$ *handoff,* CauchyLimitSeal$^{\uparrow}$ *boundary,* Real$^{\uparrow}$ *seal, and structural rows. Then their tail-seal synchronizer reads pull back to the same finite selector-budget package, so every* CauchySealBudgetSynchronizer$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer reads the same terminal package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityTailSealSynchronizerPullbac`

Proof. First use Definition 646.224. Agreement of the terminal classifiers is agreement of the displayed selector-budget, stream, rational, dyadic, synchronizer, limit-seal, Real, and structural rows.

Apply Definition 646.215. The common StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ ledger, and Real$^{\uparrow}$ seal pull both reads back to one root-selector surface.

Next invoke Definition 646.221. The terminal readback is determined by that shared finite package for Real, completion, and limit-seal consumers.

Finally use Definition 646.223. Tail rebase preserves the terminal package, so the synchronizer handoff and seal read cannot diverge after the pullback. □

Theorem 646.228 (DiagonalLimitCompatibility selector-budget common-refinement terminality)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *root-selector read. Suppose its selector-budget row agrees simultaneously with the* CauchyCriterion$^{\uparrow}$ *selector-seal route and with the* UniformCauchyCriterion$^{\uparrow}$ *finite-family Real/Completion window. Then the common refinement has a unique terminal finite package through* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* CauchySealBudgetSynchronizer$^{\uparrow}$*,*

CauchyLimitSeal$^{\uparrow}$, *and* Real$^{\uparrow}$. *No quotient stream, selected host limit, countable-choice row, ambient completeness assertion, or independent completion object can distinguish the consumer route.*

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilitySelectorBudgetCommonRefinem

Proof. Begin with the local selector-budget terminal classifier of Definition 646.224. It identifies the terminal classifier as the single displayed selector-budget package rather than as a family of independent Real, completion, or limit-seal coordinates.

Use the tail-seal synchronizer pullback of Definition 646.227. Agreement of the selector budget with the synchronizer and seal rows pulls the CauchySealBudgetSynchronizer$^{\uparrow}$ and Real$^{\uparrow}$ consumer reads back to the same finite root-selector package.

Now read the cross-chapter inputs through their displayed routes. The CauchyCriterion$^{\uparrow}$ diagonal selector terminality supplies the same selector-seal route, while the UniformCauchyCriterion$^{\uparrow}$ Real-completion window exhaustion supplies the finite-family window consumed by Real and completion readers.

Finally apply the local Real completion route non-escape theorem, Definition 646.226. Since every accepted Real or completion consumer route factors through the displayed stream, rational, dyadic, synchronizer, limit-seal, and Real rows, there is no remaining coordinate by which a quotient stream, host-selected limit, choice row, ambient completeness assertion, or independent completion object can separate the common-refinement route. □

Theorem 646.229 (DiagonalLimitCompatibility criterion-uniform seal pullback)**.** *Let*

$$K = (R, T, S, D, W, Q, L, H, C, P, N)$$

be an accepted DiagonalLimitCompatibility$^{\uparrow}$ *root-selector read. Suppose the selector-budget row of K agrees with the* CauchyCriterion$^{\uparrow}$ *selector-seal route and also agrees with the* UniformCauchyCriterion$^{\uparrow}$ *finite-family Real/Completion window. Then the two consumers pull back to the same finite seal package through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchySealBudgetSynchronizer$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, *and* Real$^{\uparrow}$.

The shared package is exactly the displayed $R, T, S, D, W, Q, L, H, C, P, N$ read of K. No quotient stream, selected host limit, countable-choice row, ambient completeness assertion, or independent completion object separates the CauchyCriterion and UniformCauchyCriterion consumers.

Lean4 ✓ BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityCriterionUniformSealPullbac

Proof. Start with the local terminal classifier, Definition 646.224. It identifies the selector-budget read of K with the single displayed package rather than with separate criterion, uniform-family, Real, or completion coordinates.

Next apply the tail-seal synchronizer pullback, Definition 646.227. The common selector-budget row carries the CauchySealBudgetSynchronizer$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, and Real$^{\uparrow}$ reads back to the same finite package.

Now use common-refinement terminality, Definition 646.228. The CauchyCriterion$^{\uparrow}$ selector-seal route and the UniformCauchyCriterion$^{\uparrow}$ finite-family Real/Completion window therefore meet at the same terminal refinement through the stream, rational, dyadic, synchronizer, limit-seal, and Real rows.

Finally invoke Real completion route non-escape, Definition 646.226. Since the accepted route inventory has no coordinate for quotient streams, host-selected limits, choice rows, ambient completeness, or detached completion objects, the two criterion consumers have no public path away from the displayed seal package. □

Theorem 646.230 (Diagonal-limit compatibility root-request coverage)**.** *Let $K = (R, T, S, D, W, Q, L, H, C, P, N)$ be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Every root request served by K reads the RealCompleteness packet row, the regular-Cauchy triangle-bound row, the* DyadicRatCore$^{\uparrow}$ *ledger row, the* StreamName$^{\uparrow}$ *window row, the* RegSeqRat$^{\uparrow}$ *readback row, and the* Real$^{\uparrow}$ *seal row through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route. No root request receives a hidden completeness witness, host real equality, quotient stream, selected limit, or second dyadic ledger outside those rows.*

Proof. Start with the carrier projection of Definition 646.1. The packet coordinates display the diagonal source, transport row, selector row, dyadic ledger, StreamName window, RegSeqRat readback, Real seal, componentwise transport, continuation route, package provenance, and local name.

Apply Definition 646.2. The local certificate surface forces any accepted root request to read carrier admission, route replay, provenance, and local naming through the same rows.

Now use Definition 646.79, Definition 646.81, and Definition 646.82. These isolate the RealCompleteness packet, triangle-bound row, dyadic ledger, StreamName window, RegSeqRat readback, and Real seal as one displayed route before the root request reaches a consumer.

Finally apply Definition 646.83. Since the carrier contains no hidden completeness witness, host equality, quotient stream, selected limit, or alternate ledger coordinate, none can be exported by the root request. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRootRequestCoverage`

Theorem 646.231 (Diagonal-limit compatibility root-selector lock)**.** *For an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet $K = (R, T, S, D, W, Q, L, H, C, P, N)$, a selector read on the root-request route cannot choose a second StreamName window or dyadic ledger outside the displayed* BHist/hsame/Cont/Pkg/NameCert *path. The selector is locked to the route that reads D, then W, then Q, then L, with H, C, P, N providing only transport, replay, provenance, and naming for that route.*

Proof. Project the accepted packet through Definition 646.1. There is one dyadic ledger coordinate D and one StreamName window coordinate W available to the selector row.

Use Definition 646.230 to place every root request on the same RealCompleteness, triangle-bound, dyadic, window, readback, and seal route. The selector therefore reads the displayed D, W, Q, L sequence before any downstream completion-facing consumer sees the packet.

Apply Definition 646.100 and Definition 646.193. These theorems exhaust selector-budget reads by the listed carrier coordinates and rule out an unlisted budget source or hidden selector input.

The transport and provenance rows H, C, P, N do not introduce another window or ledger; they only replay and name the displayed route. Thus the root selector is locked to the finite path already present in the carrier. □

Lean4 ✓ `BEDC.Derived.Diagonallimitcompatibilityup.DiagonalLimitCompatibilityRootSelectorLock`

Theorem 646.232 (Diagonal-limit compatibility root-seal package)**.** *The root-request rows of an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet repack as the downstream* CauchyLimitSeal$^{\uparrow}$/Real$^{\uparrow}$ *consumer package*

$$R, T, S, D, W, Q, L, H, C, P, N.$$

The package exposes the RealCompleteness packet, triangle-bound row, dyadic ledger, StreamName window, RegSeqRat readback, Real seal, and the displayed route/name rows, and it exposes no separate completeness principle, quotient stream, host real equality, selected limit, or hidden selector route.

Proof. First combine Definition 646.230 with Definition 646.231. They fix the root-request route and prevent a second selector window or dyadic ledger from entering the package.

Next apply Definition 646.153. The same displayed rows already form the route-budget package consumed by downstream completion and seal surfaces.

Use Definition 646.101 and Definition 646.196. The RegSeqRat readback and Real seal are consumed only through D, W, Q, L, H, C, P, N, so a $\mathsf{CauchyLimitSeal}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer receives the repacked finite route rather than a second source.

The exclusion clause follows by coordinate exhaustion from the carrier and the cited non-escape theorems. Nothing in $R, T, S, D, W, Q, L, H, C, P, N$ can project a separate completeness principle, quotient stream, host real equality, selected limit, or hidden selector route. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootSealPackage`

Theorem 646.233 (Diagonal-limit compatibility root request-selector exhaustion)**.** *For an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet* $K = (R, T, S, D, W, Q, L, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*- or completion-facing consumer of the root-request row must pass through the root-request coverage row, the selector lock row, and the root-seal package row before reading any seal-budget endpoint or selected-tail endpoint. The route exports only displayed* BHist *request rows, selector-budget rows,* hsame *transports,* Cont *replays,* Pkg *provenance, and* NameCert *rows, and it exports no selected real, quotient stream equality, host limit, or ambient completeness principle.*

Proof. First apply Definition 646.230. It exposes the root request through the RealCompleteness packet, triangle-bound row, dyadic ledger, StreamName window, RegSeqRat readback, Real seal, and displayed structural rows.

Next apply Definition 646.231. The selector is forced to use the same displayed D, W, Q, L path, so no second window, dyadic ledger, selected-tail source, or hidden selector input can be inserted before a consumer reads the route.

Finally apply Definition 646.232. The seal-facing package repacks exactly the exposed request and selector rows for downstream $\mathsf{CauchyLimitSeal}^{\uparrow}/\mathsf{Real}^{\uparrow}$ use.

The exclusion clause is coordinate exhaustion. The accepted packet contains request rows, selector-budget rows, transports, continuation replays, provenance, and local naming rows, but it contains no selected real, quotient stream equality, host limit, or ambient completeness principle to export. □

Lean4 ✓ `BEDC.Derived.DiagonallimitcompatibilityUp.DiagonalLimitCompatibilityRootRequestSelectorPackage`

Chapter 647

A Concrete Naming Certificate for $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$

The $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ surface packages comparison between two observers as a finite public certificate. Each observer keeps its own universe-for-observer row; the classifier compares only the displayed rows that pass through locality cells, GAP commitments, hsame transports, Cont routes, Pkg provenance, and the local naming row. It is not a global view over both worlds.

The packet consumes chapter 578 for the two observer-state rows O_0, O_1. Its locality evidence is read from the public cell surfaces of chapter 509 or chapter 513, and the comparison is checked against the public alignment surface of chapter 643 rather than against an ambient shared world.

Definition 647.1 (Observer-perspective-classifier carrier)**.** *An* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$$

with the following displayed rows:

1. *O_0 and O_1 are accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *rows naming the two conscious-observer state surfaces being compared;*
2. *U_0 and U_1 are the two universe-for-observer record surfaces, each read only as the records and closure rules accumulated by the corresponding observer row;*
3. *L is the locality-cell evidence row, displayed as a* $\mathsf{LocalityCell}^{\uparrow}$ *or* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *public coordination surface for the two observer rows;*
4. *G is the paired* GAP *commitment ledger recording the two-sided coherence commitments by which U_0 and U_1 can be compared without identifying them as one host world;*
5. *H records componentwise* hsame *transports for O_0, O_1, U_0, U_1, L, G and for the package and local naming rows;*
6. *C records the displayed* Cont *routes that let a consumer read each observer row, its universe-for-observer surface, the locality-cell evidence, and the paired* GAP *commitments in that order;*
7. *P is the* Pkg *provenance over* $\mathsf{ConsciousObserverState}^{\uparrow}$*, universe-for-observer record surfaces,* $\mathsf{LocalityCell}^{\uparrow}$*,* $\mathsf{ObserverLocalityCell}^{\uparrow}$*,* GAP *ledgers,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only through the displayed rows $O_0, O_1, U_0, U_1, L, G, H, C, P, N$ and their recorded transports and routes.

Lean4 (definition) `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierCarrier`

Theorem 647.2 (ObserverPerspectiveClassifier NameCert obligations)**.** *For the carrier of Definition 647.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}$ *surface has exactly five obligation rows: carrier exposure for $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$; observer anchoring through O_0, O_1; universe-for-observer exposure through U_0, U_1; locality and coherence comparison through L, G, H, C; and non-escape from a god's-eye observer, host equality of worlds, private* Self *access, global spacetime carriers, hidden synchronization maps, and unrecorded cross-perspective evidence.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierNameCertObligations`

Proof. Carrier exposure is projection from the finite packet $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$.

Observer anchoring is read from O_0 and O_1. Universe-for-observer exposure is read from U_0 and U_1, with each universe surface attached to its own observer row.

The comparison obligation is then read from L, G, H, C. The locality-cell row L supplies the public coordination surface, the paired GAP ledger G records the coherence commitments, the transport row H moves only displayed coordinates, and the continuation row C replays only the displayed reading order.

Finally inspect the carrier list. No coordinate names a god's-eye observer, host equality of the two worlds, private Self access, global spacetime carrier, hidden synchronization map, or unrecorded evidence row, so none of those objects is exported by the local naming surface. □

Theorem 647.3 (ObserverPerspectiveClassifier locality-cell comparison)**.** *Every accepted* ObserverPerspectiveClass *packet compares U_0 and U_1 only after the observer rows O_0, O_1, the locality-cell evidence L, and the paired* GAP *commitments G have been displayed in the same finite carrier. A consumer may classify the two perspectives through H and C, but it cannot replace the locality-cell comparison by a single ambient world, a shared external spacetime, or direct access to either observer's private* Self *row.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierLocalityCellComparis`

Proof. Project the accepted packet and read the two observer rows O_0, O_1 together with their universe-for-observer surfaces U_0, U_1.

The classifier is not applied to those universe surfaces in isolation. It first uses the locality-cell evidence L and the paired GAP ledger G, then replays those rows through the displayed transports H and continuations C.

Since the packet contains no ambient world coordinate, shared external spacetime row, or private Self coordinate, the comparison remains a finite locality-cell comparison between displayed observer perspectives. □

Theorem 647.4 (ObserverPerspectiveClassifier ledger exactness)**.** *Every public read exported by an accepted* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *carrier factors through the finite ledger*

$$O_0, O_1, U_0, U_1, L, G, H, C, P, N.$$

The ledger is exact: observer rows, universe-for-observer surfaces, locality-cell evidence, paired GAP *commitments, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row are available, and no other comparison primitive is available.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifier Up.ObserverPerspectiveClassifierLedgerExactness

Proof. The carrier definition displays all rows of the packet. The mathematical comparison data are O_0, O_1, U_0, U_1, L, G, and the structural data are H, C, P, N.

Any consumer read must pass through the displayed continuation routes in C. Those routes expose the two observer rows, their universe surfaces, the locality evidence, and the paired commitments before the package and naming rows are read.

There is no remaining coordinate for a second comparison primitive. Thus every exported read factors through the listed ledger and the ledger is exact for public consumers. □

Theorem 647.5 (ObserverPerspectiveClassifier non-escape boundary)**.** *An accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier exports only the two conscious-observer state rows, the two universe-for-observer record surfaces, the locality-cell evidence row, the paired* GAP *commitment ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It exports no god's-eye observer, no host equality of worlds, no private* Self *access, and no global spacetime carrier.*

Proof. Inspect the carrier coordinates of Definition 647.1. Every exported row is one of $O_0, O_1, U_0, U_1, L, G, H, C, P,$

The positive comparison data remain public and finite: observer rows, universe-for-observer surfaces, locality-cell evidence, and paired GAP commitments. The rows H, C, P, N transport, route, package, and name those same displayed coordinates.

No listed coordinate is a god's-eye observer, a host equality witness between worlds, a private Self row, or a global spacetime carrier. Therefore an accepted classifier cannot export any of those objects. □

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifier Up.ObserverPerspectiveClassifierNonescape

Theorem 647.6 (ObserverPerspectiveClassifier anchor-change stability)**.** *In an accepted* ObserverPerspectiveClassif *carrier, changing the observer anchor from the O_0 side to the O_1 side, or from the O_1 side to the O_0 side, preserves a comparison verdict only through the displayed rows*

$$O_0, O_1, U_0, U_1, L, G, H, C, P, N.$$

The stability route may use the two observer rows, the two universe-for-observer surfaces, locality evidence, paired GAP *commitments, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row, but it cannot pass through a privileged observer or any unlisted observer-neutral witness.*

Proof. Apply the carrier description in Definition 647.1. The only named anchors are O_0 and O_1, and each universe-for-observer surface is attached to one of those rows.

Apply the obligation theorem Definition 647.2. Anchor comparison is part of the finite local naming surface, so an anchor change must be replayed through the displayed locality, commitment, transport, continuation, package, and naming rows.

The locality comparison theorem Definition 647.3 supplies the admissible comparison route, while Definition 647.4 says that the ledger has no further comparison coordinate. Finally Definition 647.5 rules out a god's-eye observer and any unlisted observer-neutral witness, so the comparison is stable only through the displayed packet rows. □

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifier Up.ObserverPerspectiveClassifierAnchorChangeStability

Theorem 647.7 (ObserverPerspectiveClassifier GAP commitment readback)**.** *Every cross-perspective comparison exported by an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier reads the paired* GAP *row G before it exports a comparison verdict. A verdict based only on O_0, O_1, U_0, U_1 or only on the locality row L is not a public classifier verdict for this carrier.*

Proof. The carrier in Definition 647.1 displays G as the paired commitment ledger by which U_0 and U_1 can be compared without identifying them as one host world.

By Definition 647.2, the comparison obligation is not just observer exposure or universe exposure; it is the locality and coherence comparison through L, G, H, C. The locality-cell comparison theorem Definition 647.3 makes the same ordering explicit: the classifier compares after the observer rows, the locality-cell evidence, and the paired GAP commitments have been displayed in the same finite carrier.

The exactness theorem Definition 647.4 then prevents a consumer from dropping G while still claiming the public verdict. The non-escape theorem Definition 647.5 prevents replacing G by hidden synchronization or host equality, so the paired commitment row must be read before the verdict is exported. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierGapCommitmentReadback`

Theorem 647.8 (ObserverPerspectiveClassifier comparison-route totality)**.** *Every accepted comparison route for an* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier factors through all of the public classifier layers: observer rows O_0, O_1, universe-for-observer surfaces U_0, U_1, locality evidence L, paired* GAP *commitments G, componentwise* hsame *transports H, displayed* Cont *routes C,* Pkg *provenance P, and the local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierComparisonRouteTotali`

Proof. Start with the finite packet in Definition 647.1. Its displayed rows already enumerate the observer, universe, locality, commitment, transport, continuation, provenance, and naming layers.

Apply Definition 647.2 to read those layers as the local naming obligations. Apply Definition 647.3 to place the comparison after the observer rows, universe surfaces, locality evidence, and paired commitment ledger.

It remains to account for transport, routing, provenance, and local naming. The exactness theorem Definition 647.4 says that public reads factor through H, C, P, N after the mathematical comparison rows, and Definition 647.5 says that no hidden comparison route remains outside this finite list. Hence every accepted comparison route factors through the stated public classifier layers. □

Theorem 647.9 (ObserverPerspectiveClassifier state totality)**.** *Every accepted public verdict for an* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier exposes both* ConsciousObserverState$^{\uparrow}$ *rows O_0, O_1 and their universe-for-observer surfaces U_0, U_1 before the verdict is public. In particular, a row that exports a comparison verdict without first displaying the two observer-state rows and the two attached universe surfaces is not an accepted classifier verdict for the carrier $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierStateTotality`

Proof. Unpack the carrier tuple of Definition 647.1. Its first four displayed comparison rows are exactly O_0, O_1, U_0, U_1, with O_0, O_1 naming the two conscious-observer state surfaces and U_0, U_1 naming the corresponding universe-for-observer surfaces.

The exactness theorem Definition 647.4 says that every public read exported by an accepted carrier factors through the finite ledger. The comparison-route totality theorem Definition 647.8

then orders any accepted comparison route through observer rows, universe surfaces, locality evidence, paired commitments, transports, routes, provenance, and local naming.

Finally Definition 647.5 rules out replacing those displayed rows by a god's-eye observer, host equality, private Self access, or a global spacetime carrier. Hence a public verdict must expose both observer-state rows and both universe-for-observer surfaces before it exports the classifier verdict. □

Theorem 647.10 (ObserverPerspectiveClassifier locality refinement)**.** *Let $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$ be an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier. If a locality-cell row L' refines the displayed locality evidence L and the same paired* GAP *row G is transported with L' through the recorded* hsame *row, the displayed* Cont *routes, and the* Pkg *provenance, then the public comparison route is the same accepted route with L' in the locality coordinate. No public consumer gains an additional comparison primitive by refining L in this way.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierLocalityRefinement`

Proof. Begin with the carrier definition. The locality coordinate and the paired commitment coordinate occur as the adjacent public comparison rows L, G, while H, C, P record the componentwise transports, displayed reading routes, and provenance for those same rows.

The ledger exactness theorem Definition 647.4 makes the route exact: a public read may use the observer rows, universe surfaces, locality row, paired commitment row, transports, continuations, provenance, and the local naming row, and no further comparison coordinate is available.

Transporting the refined row L' together with the same G through H, C, P therefore changes only the displayed locality coordinate. The route still passes through the same ordered classifier layers by Definition 647.8.

The non-escape theorem Definition 647.5 prevents the refinement from introducing a hidden synchronization map, host equality, or observer-neutral witness. Thus the public comparison route is preserved with L' in the locality coordinate. □

Theorem 647.11 (ObserverPerspectiveClassifier perspective stability)**.** *For an accepted* ObserverPerspectiveClassif *carrier $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$, swapping the two observer perspectives preserves acceptedness exactly for the carrier whose observer rows, universe-for-observer surfaces, locality row, paired* GAP *row,* hsame *transports,* Cont *routes,* Pkg *provenance, comparison route, and local* NameCert *row are swapped together. Swapping only one face of the carrier is not an accepted public perspective swap.*

Proof. First unpack the finite carrier tuple. The observer faces are not isolated parameters: O_0, O_1 are paired with U_0, U_1, and the comparison is witnessed through the locality row L, the paired commitment row G, the transport row H, the continuation row C, the provenance row P, and the local naming row N.

For the forward direction, assume the public perspective swap is accepted. By Definition 647.9, both observer-state rows and both universe-for-observer surfaces must be exposed before the verdict. By Definition 647.8, the accepted route also carries the locality, commitment, transport, continuation, provenance, and naming layers, so those layers must travel with the swapped perspectives.

For the reverse direction, swap the two observer rows together with their universe surfaces and carry the locality row, paired GAP row, hsame transports, Cont routes, Pkg provenance, comparison route, and local NameCert row through the same finite ledger positions. The carrier definition still displays a tuple of the required form, and Definition 647.4 supplies the same public readback surface.

It remains to exclude partial swaps. The non-escape theorem Definition 647.5 rules out any unlisted observer-neutral witness that could repair a swap of only one face. Hence acceptedness is preserved exactly when the paired observer rows, universe surfaces, locality evidence, paired commitment row, transports, routes, provenance, comparison route, and local naming row are swapped together. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierPerspectiveStability`

Theorem 647.12 (ObserverPerspectiveClassifier dependency scope)**.** *The accepted dependency scope of an* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier is exactly the finite public row family supplied by* ConsciousObserverState$^{\uparrow}$, LocalityCell$^{\uparrow}$ *or* ObserverLocalityCell$^{\uparrow}$, CrossPerspectiveAlignment$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The classifier has no dependency on an ambient world row, host equality between worlds, private* Self *access, global spacetime carrier, hidden synchronization map, or unrecorded cross-perspective evidence.*

Proof. Project the carrier of Definition 647.1. Its comparison rows are the two conscious-observer state rows, the two universe-for-observer surfaces, the locality-cell row, and the paired GAP row; its structural rows are the BHist packet, hsame transports, Cont routes, Pkg provenance, and local NameCert row.

Apply Definition 647.3 and Definition 647.7. They force any comparison to pass through the displayed locality evidence, paired commitment ledger, and recorded alignment surface rather than through a shared ambient world.

Finally apply Definition 647.4 and Definition 647.5. Exactness says no other comparison primitive is available, and non-escape excludes ambient world rows, host equality, private Self access, global spacetime carriers, hidden synchronization maps, and unrecorded evidence. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierDependencyScope`

Theorem 647.13 (ObserverPerspectiveClassifier alignment consumer)**.** *Every downstream cross-perspective consumer of an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict must read the comparison through the displayed locality row, paired* GAP *commitment row, transport row, continuation row, provenance row, and local* NameCert *row. A consumer that bypasses one of those displayed rows is not reading the public classifier verdict.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierAlignmentConsumer`

Proof. Start from an accepted public verdict. By Definition 647.8, the verdict route factors through $O_0, O_1, U_0, U_1, L, G, H, C, P, N$ in the public classifier layers.

The consumer side only needs the comparison layers after the observer-state and universe surfaces have been exposed. Those layers are precisely the locality row L, paired commitment row G, transport row H, continuation row C, provenance row P, and local naming row N.

If a consumer bypasses one of these rows, Definition 647.4 says the read no longer factors through the exact public ledger. The non-escape boundary of Definition 647.5 prevents replacing the missing row by hidden synchronization, host equality, or a private observer witness, so the bypassed read is not the public classifier verdict. □

Theorem 647.14 (ObserverPerspectiveClassifier observer-state GAP route)**.** *For an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier* $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$*, a public verdict may consume the paired* GAP *row* G *only after the two conscious-observer state rows* O_0, O_1 *have exposed their GAP acknowledgement rows through the recorded state surface. A verdict that reads* G *without both acknowledgements is not an accepted observer-perspective classifier verdict.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierObserverStateGapRoute

Proof. Project the carrier tuple from Definition 647.1. The observer rows O_0, O_1 precede the universe surfaces, locality row, and paired commitment row, so the public carrier fixes the state coordinates before the comparison row G can be read.

The state-totality theorem Definition 647.9 forces both conscious-observer state rows to be exposed before any public classifier verdict. The paired commitment readback theorem Definition 647.7 then routes the displayed GAP row through the same finite comparison ledger.

Finally apply the ConsciousObserverState exactness anchor ConsciousObserverStateCarrier_gap_acknowledgeme to the two state rows. It identifies the required acknowledgement rows as the exact state-side witnesses consumed by G. The ledger exactness and non-escape theorems Definition 647.4 and Definition 647.5 rule out replacing either acknowledgement by host equality, hidden synchronization, or private observer access. □

Theorem 647.15 (ObserverPerspectiveClassifier cross-alignment locality soundness)**.** *A* CrossPerspectiveAlignmen *consumer may use an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict only when the displayed locality row L and paired* GAP *row G pass through the alignment locality-cell soundness route. If the consumer supplies a different locality cell, an unpaired commitment row, or an unrecorded alignment witness, it is not using the public classifier surface.*

↪ variant BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierCrossAlignmentLocalitySoundne

Proof. Start with the comparison route of Definition 647.8. It orders the public read through the two observer rows, two universe surfaces, displayed locality row L, paired commitment row G, transports H, continuations C, provenance P, and local naming row N.

The alignment-consumer theorem Definition 647.13 restricts downstream cross-perspective consumers to precisely the displayed comparison rows. Thus the consumer cannot detach L from G or replace either row before reading the classifier verdict.

Apply Definition 647.3 to compare the locality cell against the classifier ledger, and then apply the CrossPerspectiveAlignment soundness anchor CrossPerspectiveAlignmentCarrier_locality_cell_soundness. These two routes identify the accepted locality-cell soundness witness with the displayed pair L, G.

The non-escape theorem Definition 647.5 excludes hidden synchronization, ambient world equality, and unrecorded cross-perspective evidence. Hence a consumer outside the L, G soundness route is outside the public classifier surface. □

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierCrossAlignmentLocalit

Theorem 647.16 (ObserverPerspectiveClassifier locality-cell exhaustion)**.** *In an accepted* ObserverPerspectiveClass *carrier $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$, the locality coordinate L is exhausted by the public* LocalityCell$^{\uparrow}$ *or* ObserverLocalityCell$^{\uparrow}$ *packet together with the rows H, C, P, N. No ambient world row, hidden synchronization map, host equality, or unrecorded locality witness can replace that public locality-cell data.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierLocalityCellExhaustio

Proof. Read the carrier as a finite ledger. The locality row L is not an abstract parameter: it is tied to the componentwise hsame transports H, displayed Cont routes C, Pkg provenance P, and local NameCert row N.

The locality-cell comparison theorem Definition 647.3 and ledger exactness theorem Definition 647.4 say that every public read of L factors through those displayed rows. Apply the

LocalityCell readback anchor LocalityCellPacket_interobserver_readback when L is supplied by a LocalityCell$^{\uparrow}$ packet, and apply the ObserverLocalityCell boundary anchor ObserverLocalityCellPacket_consumer_bo when L is supplied by an ObserverLocalityCell$^{\uparrow}$ packet.

It remains to rule out replacements. The dependency-scope theorem Definition 647.12 limits the accepted row family to the displayed public packets, and the non-escape theorem Definition 647.5 rejects ambient world rows, hidden synchronization maps, host equality, private observer access, and unrecorded cross-perspective evidence. Therefore L is exhausted exactly by the public locality-cell packet and the rows H, C, P, N. □

Theorem 647.17 (ObserverPerspectiveClassifier locality-commitment inversion)**.** *Let $Q = (O_0, O_1, U_0, U_1, L, G, H$ be an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier read inside the public* BHist *packet. Any accepted public comparison verdict determines the displayed locality row L and paired* GAP *commitment row G through the finite rows Q, with H the componentwise* hsame *transport row, C the displayed* Cont *route row, P the* Pkg *provenance row, and N the local* NameCert *row. A verdict that omits either L or G is outside the public classifier.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifier Up.ObserverPerspectiveClassifierLocalityCommitmentInv`

Proof. Project the carrier from Definition 647.1. The public read has only the finite coordinates $O_0, O_1, U_0, U_1, L, G, H, C, P, N$, so the comparison verdict cannot choose an unlisted locality or commitment coordinate.

Apply Definition 647.3. It identifies the displayed L row as the locality-cell comparison surface tied to the transport, continuation, provenance, and naming rows. Apply Definition 647.7 to identify G as the paired commitment ledger for the same public comparison route.

It remains to exclude partial verdicts. Ledger exactness Definition 647.4 says the public classifier has no comparison route outside the displayed rows, and non-escape Definition 647.5 rejects ambient locality, hidden synchronization, host equality, private observer access, and unrecorded alignment evidence. Hence a verdict missing either L or G is not an accepted public classifier verdict. □

Theorem 647.18 (ObserverPerspectiveClassifier two-state no-confusion)**.** *For accepted* ObserverPerspectiveClassifi *carrier reads $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$ and $Q' = (O'_0, O'_1, U'_0, U'_1, L', G', H', C', P', N')$, equality of the public classifier verdict forces agreement on both conscious-observer state rows and both universe-for-observer rows before locality and alignment rows are compared. If any one of O_0, O_1, U_0, U_1 differs from the corresponding displayed row in Q', the two reads are different public carrier reads rather than the same classifier verdict.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifier Up.ObserverPerspectiveClassifierTwoStateNoConfusion`

Proof. Start with the carrier definition. It presents each accepted read as a finite ordered BHist packet, so the state and universe-for-observer rows are public coordinates of the verdict rather than hidden parameters.

Apply state totality Definition 647.9. Both observer-state rows and both attached universe surfaces must be exposed before the verdict may continue to locality, commitment, transport, continuation, provenance, and naming rows.

Next apply comparison-route totality Definition 647.8 and perspective stability Definition 647.11. These identify the downstream comparison route only after the displayed state and universe rows have been fixed.

Finally use non-escape Definition 647.5. It forbids repairing a disagreement in O_0, O_1, U_0, U_1 by host equality, private observer access, hidden synchronization, or an unrecorded cross-perspective witness. Therefore such a disagreement changes the public carrier read itself. □

Theorem 647.19 (ObserverPerspectiveClassifier alignment ledger coverage)**.** *Every* CrossPerspectiveAlignment$^{\uparrow}$ *consumer of an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict is covered by the displayed ledger* L, G, H, C, P, N*. The locality row* L *is supplied either by a public* LocalityCell$^{\uparrow}$ *packet or by a public* ObserverLocalityCell$^{\uparrow}$ *packet, and there is no third ambient locality, hidden synchronization route, or unrecorded alignment channel available to the consumer.*

Proof. Begin with dependency scope Definition 647.12. It confines the accepted classifier to the displayed observer-state, universe-for-observer, locality, alignment, BHist, hsame, Cont, Pkg, and NameCert rows.

Apply the alignment-consumer theorem Definition 647.13. A downstream CrossPerspectiveAlignment$^{\uparrow}$ consumer must read the verdict through L, G, H, C, P, N rather than through an independent alignment surface.

The cross-alignment locality soundness theorem Definition 647.15 binds the consumer to the same displayed L, G pair. Locality-cell exhaustion Definition 647.16 restricts L to the public LocalityCell$^{\uparrow}$ or ObserverLocalityCell$^{\uparrow}$ packet, and locality-commitment inversion Definition 647.17 keeps L and G jointly determined by the accepted carrier.

Thus every alignment consumer is covered by the displayed ledger. Any purported consumer using a third locality surface, hidden synchronization, or unrecorded alignment witness leaves the public classifier route. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierAlignmentLedgerCovera`

Theorem 647.20 (ObserverPerspectiveClassifier two-observer source admission)**.** *An accepted public verdict for an* ObserverPerspectiveClassifier$^{\uparrow}$ *carrier must first expose both accepted* ConsciousObserverState$^{\uparrow}$ *rows* O_0, O_1 *and their attached universe-for-observer surfaces* U_0, U_1*. The verdict may then read locality and alignment data, but it cannot read* L*,* G*, or a* CrossPerspectiveAlignment$^{\uparrow}$ *consumer route before the two observer-state rows and the two universe surfaces have appeared in the finite carrier.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierTwoObserverSourceAdmi`

Proof. Project the carrier $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$ from Definition 647.1. The first displayed rows are the two observer-state sources and their two universe-for-observer surfaces, so the classifier has no earlier public comparison coordinate.

Apply the state-totality theorem Definition 647.9. It turns this projection order into a public verdict condition: both observer-state rows and both attached universe surfaces must be exposed before the verdict is exported.

Finally apply the comparison-route totality and non-escape theorems, Definition 647.8 and Definition 647.5. They force the later locality, commitment, transport, continuation, provenance, and alignment-facing reads to remain downstream of the displayed state rows, and they rule out a hidden observer-neutral source that could bypass the two source rows. □

Theorem 647.21 (ObserverPerspectiveClassifier alignment-locality cycle)**.** *For an accepted* ObserverPerspectiveCla *carrier, the locality coordinate* L *must be a public* LocalityCell$^{\uparrow}$ *or* ObserverLocalityCell$^{\uparrow}$ *packet paired with the same displayed* GAP *row* G*. Every* CrossPerspectiveAlignment$^{\uparrow}$ *consumer of the verdict must consume that* L, G *pair through the recorded rows* H, C, P, N*; it cannot bypass the pair by*

supplying an ambient locality row, an unpaired commitment row, or an unrecorded alignment witness.

Proof. Read L from the carrier definition. Its declared role is the public locality-cell evidence row, and the next comparison coordinate G is the paired GAP commitment ledger for comparing U_0 and U_1 without identifying them as one host world.

Apply locality-cell exhaustion Definition 647.16. It restricts L to the public $\mathsf{LocalityCell}^{\uparrow}$ or $\mathsf{ObserverLocalityCell}^{\uparrow}$ packet together with the recorded transport, route, provenance, and naming rows.

Next apply cross-alignment locality soundness Definition 647.15. That theorem binds the alignment-facing consumer to the same displayed pair L, G.

The ledger exactness and non-escape boundary, Definition 647.4 and Definition 647.5, then exclude ambient locality rows, unpaired commitment rows, hidden synchronization, and unrecorded alignment witnesses. Thus the locality and alignment reads form the stated cycle around the public L, G pair. □

Lean4 ✓ `BEDC.Derived.ObserverperspectiveclassifierUp.ObserverPerspectiveClassifierAlignmentLocalityCycl`

Theorem 647.22 (ObserverPerspectiveClassifier obligation-closure witness)**.** *The carrier row, the five local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}$ *obligation rows, the two observer-state source rows, the locality-cell row, the paired* GAP *row, the* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *consumer route, the componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local naming row form one finite obligation-closure witness for* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$.

Proof. Begin with the carrier and local obligation theorems, Definition 647.1 and Definition 647.2. They display the finite packet and identify the five local naming obligations carried by it.

The source admission theorem Definition 647.20 binds both conscious-observer state rows and both universe-for-observer surfaces before any public verdict. The alignment-locality cycle theorem Definition 647.21 binds the public locality row, paired commitment row, and alignment-facing consumer to the same displayed route.

It remains to gather the already established public boundaries. Apply ledger exactness, non-escape, comparison-route totality, dependency scope, observer-state GAP route, cross-alignment locality soundness, and locality-cell exhaustion. These theorems account for H, C, P, N, exclude every unlisted observer-neutral or ambient replacement, and leave exactly the finite row family named in the statement. □

Lean4 ✓ `BEDC.Derived.ObserverperspectiveclassifierUp.ObserverPerspectiveClassifierObligationClosureWitn`

Theorem 647.23 (ObserverPerspectiveClassifier obligation unblock package)**.** *The carrier exposure, local* NameCert *obligations, locality-cell comparison, ledger exactness, non-escape boundary, paired* GAP *readback, comparison-route totality, state totality, locality refinement, perspective stability, dependency scope, alignment-consumer discipline, observer-state GAP route, cross-alignment locality soundness, locality-cell exhaustion, two-observer source admission, alignment-locality cycle, and obligation-closure witness together form the finite* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *obligation package for obligation-level closure.*

Lean4 ✓ `BEDC.Derived.ObserverperspectiveclassifierUp.ObserverPerspectiveClassifierObligationUnblockPack`

Proof. First project the finite carrier from Definition 647.1. This supplies the rows $O_0, O_1, U_0, U_1, L, G, H, C, P, N$ and fixes the only coordinates available to the package.

The local obligation surface is given by Definition 647.2. The comparison surface is then constrained by Definition 647.3, Definition 647.4, Definition 647.5, and Definition 647.7.

The route and state coverage are supplied by Definition 647.8 and Definition 647.9. Stability under admissible local changes is supplied by Definition 647.10 and Definition 647.11.

It remains only to bind the consumer-facing dependency boundary. Apply Definition 647.12 to identify the accepted row family, and apply Definition 647.13 to force downstream consumers through the displayed locality, commitment, transport, continuation, provenance, and naming rows.

The three concrete rows above close the exposed obligation route. The observer-state GAP route Definition 647.14 ties G to both state acknowledgements, the cross-alignment locality soundness route Definition 647.15 ties L, G to the alignment consumer, and locality-cell exhaustion Definition 647.16 rules out ambient or hidden replacements for L. These finite components assemble exactly the obligation package claimed.

The dependency witness is explicit in the same finite packet. Apply Definition 647.20 for the two source-state rows, Definition 647.21 for the L, G locality and alignment cycle, and Definition 647.22 to package these rows with the earlier carrier, obligation, exactness, and non-escape boundaries. □

Theorem 647.24 (ObserverPerspectiveClassifier BHist scope)**.** *Every accepted carrier read for* ObserverPerspectiveClassifier$^{\uparrow}$ *is confined to the displayed finite* BHist *packet* $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$*. No observer-neutral row, host-world equality row, private* Self *row, global spacetime row, hidden synchronization row, or unrecorded cross-perspective row can enter the classifier scope.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierBHistScope`

Proof. Start with the carrier definition Definition 647.1. It presents the accepted classifier as the finite BHist packet $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$, so every public read has one of these displayed coordinates as its source.

Ledger exactness Definition 647.4 identifies the rows available inside that packet. The non-escape theorem Definition 647.5 excludes observer-neutral rows, host-world equality, private Self access, global spacetime carriers, hidden synchronization maps, and unrecorded cross-perspective evidence.

Finally apply dependency scope Definition 647.12. It says the accepted row family is exactly the public family generated by the displayed observer-state, locality, alignment, BHist, hsame, Cont, Pkg, and NameCert rows. Hence no row outside Q can enter the scope. □

Theorem 647.25 (ObserverPerspectiveClassifier alignment transport scope)**.** *Every* CrossPerspectiveAlignment$^{\uparrow}$ *consumer of an accepted* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict may use only the displayed route through* L, G, H, C, P, N*. Its transport scope is the componentwise* hsame *row* H *together with the displayed* Cont *routes* C*; it cannot import an ambient alignment map, an unpaired commitment row, or an unrecorded continuation.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierAlignmentTransportSco`

Proof. The comparison-route totality theorem Definition 647.8 first exposes the available route: observer rows, universe-for-observer surfaces, locality evidence, paired GAP commitments, hsame transports, Cont routes, Pkg provenance, and local naming.

Next use the alignment-consumer theorem Definition 647.13. It forces any downstream consumer to read the verdict through L, G, H, C, P, N rather than through a separate alignment channel.

The cross-alignment locality soundness theorem Definition 647.15 binds the consumer to the same displayed L, G pair. The alignment-locality cycle theorem Definition 647.21 then returns the read to H, C, P, N, with hsame and Cont supplying the transport scope. Thus ambient alignment maps, unpaired commitments, and unrecorded continuations are outside the accepted consumer scope. □

Theorem 647.26 (ObserverPerspectiveClassifier NameCert ledger scope)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassif}}$ *row closes only the finite observer-state, locality-cell,* GAP*, transport, continuation, package, and naming ledger displayed by* $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$*. It does not close global spacetime, private* Self *access, host equality between worlds, hidden synchronization, or unrecorded cross-perspective evidence.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierNameCertLedgerScope`

Proof. The local naming obligations are exactly the five rows listed in Definition 647.2. They cover carrier exposure, observer anchoring, universe-for-observer exposure, locality and coherence comparison, and the non-escape boundary.

Apply the obligation-closure witness Definition 647.22. It packages the carrier row, the local naming obligations, both observer-state rows, the locality-cell row, the paired GAP row, the alignment consumer route, hsame transports, Cont routes, Pkg provenance, and the local naming row as one finite obligation witness.

The obligation unblock package Definition 647.23 assembles the same rows as the chapter's obligation surface. The non-escape theorem Definition 647.5 keeps global spacetime, private Self access, host equality, hidden synchronization, and unrecorded cross-perspective evidence outside that surface. Therefore the NameCert row closes only the displayed finite ledger. □

Theorem 647.27 (ObserverPerspectiveClassifier carrier transport)**.** *For an accepted* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *carrier* $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$*, carrier transport preserves acceptedness exactly when the rows* $O_0, O_1, U_0, U_1, L, G, H, C, P, N$ *are transported componentwise through the displayed* hsame *rows in* H *and then replayed through the displayed* Cont *routes in* C*. No transport that omits an observer-state row, a universe-for-observer row, the locality row, the paired* GAP *row, package provenance, or the local naming row is an accepted carrier transport.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierCarrierTransport`

Proof. Start from the finite carrier definition Definition 647.1. It presents Q as a BHist packet with exactly the displayed observer-state, universe-for-observer, locality, commitment, transport, continuation, package, and naming rows.

The forward direction uses the transport data already present in the accepted packet. Locality refinement Definition 647.10 and perspective stability Definition 647.11 keep the comparison attached to the same public locality and observer-perspective rows. The BHist scope theorem Definition 647.24 confines the read to Q, and the alignment transport scope theorem Definition 647.25 forces the consumer route through the displayed H and C rows.

For the reverse direction, suppose every listed coordinate is transported through its displayed hsame row and replayed through its displayed Cont route. The same carrier rows are then available in the same finite order, with the package and local naming rows transported along with the data rows. Ledger exactness Definition 647.4 and NameCert ledger scope Definition 647.26 therefore reconstruct the accepted carrier surface and exclude any transport using an omitted or hidden coordinate. □

Theorem 647.28 (ObserverPerspectiveClassifier classifier determinacy)**.** *Let Q and Q' be accepted public* ObserverPerspectiveClassifier$^{\uparrow}$ *carriers with public verdicts read through their local* NameCert *rows. The two verdicts are the same classifier verdict exactly when they agree on both observer-state rows, both universe-for-observer surfaces, the locality row, the paired* GAP *row, the componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance, and the local* NameCert *row.*

Proof. First compare the observer-state coordinates. The two-state no-confusion theorem Definition 647.18 says that the accepted classifier keeps O_0 and O_1 as the two displayed state rows, so a public verdict cannot identify distinct state rows by passing to an ambient observer.

Next compare the locality and commitment coordinates. Locality-commitment inversion Definition 647.17 recovers the displayed locality row and paired GAP row from an accepted comparison verdict. Alignment-ledger coverage Definition 647.19 then accounts for the alignment-facing route through the same finite ledger.

It remains to compare the structural coordinates. Ledger exactness Definition 647.4 makes H, C, P, N the only public transport, continuation, provenance, and naming rows available to the verdict. Non-escape Definition 647.5 excludes a private observer row, host equality, global spacetime carrier, hidden synchronization map, or unrecorded comparison witness that could distinguish two otherwise equal displayed verdicts.

Thus equality of all displayed rows determines equality of the classifier verdict. Conversely, equality of classifier verdicts forces equality of those rows by applying the same no-confusion, inversion, ledger-coverage, exactness, and non-escape boundaries to the public readback of the verdicts. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierClassifierDeterminacy`

Theorem 647.29 (ObserverPerspectiveClassifier ledger exhaustion)**.** *Every accepted downstream consumer route for an* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict is exhausted by the displayed ledger $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$. Such a route cannot add a god's-eye observer, host world equality, private* Self *row, global spacetime carrier, hidden synchronization map, or unrecorded cross-perspective evidence.*

Proof. Project the accepted carrier and its public read route. Ledger exactness Definition 647.4 supplies the complete row family $O_0, O_1, U_0, U_1, L, G, H, C, P, N$, and comparison-route totality Definition 647.8 sends every consumer through that displayed route.

The obligation unblock package Definition 647.23 gathers the carrier exposure, local obligations, locality comparison, paired commitment readback, route coverage, state coverage, stability, and alignment-facing consumer discipline into the same finite package. Dependency scope Definition 647.12 then prevents a downstream consumer from importing rows outside the accepted public dependency family.

Finally apply NameCert ledger scope Definition 647.26. The local naming row closes only the displayed observer-state, universe-for-observer, locality-cell, GAP, transport, continuation, package, and naming ledger. The excluded god's-eye observer, host world equality, private Self row, global spacetime carrier, hidden synchronization map, and unrecorded cross-perspective evidence have no displayed coordinate and no accepted consumer route. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierLedgerExhaustion`

Theorem 647.30 (ObserverPerspectiveClassifier verdict source exhaustion)**.** *Every public* ObserverPerspectiveClass *verdict exhausts its source rows through the displayed packet $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$. Its observer-state source is one of the two* ConsciousObserverState$^{\uparrow}$ *rows, its locality source is the*

displayed LocalityCell$^{\uparrow}$ *or* ObserverLocalityCell$^{\uparrow}$ *row, and its alignment-facing source is the displayed* CrossPerspectiveAlignment$^{\uparrow}$ *consumer route. No public verdict can bypass these rows by using a god's-eye observer, host equality of worlds, private* Self *access, global spacetime, hidden synchronization, or unrecorded cross-perspective evidence.*

Proof. Start with the accepted carrier in Definition 647.1. It exposes the finite packet $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$ and gives no coordinate outside that packet.

The local naming obligations in Definition 647.2 bind the carrier rows to the observer anchors, universe-for-observer surfaces, locality comparison, and non-escape boundary. The locality-cell comparison theorem Definition 647.3 places the public comparison after the displayed locality row, while ledger exactness Definition 647.4 keeps the public read inside the listed ledger.

It remains to account for downstream consumers. Apply dependency scope, alignment consumer, locality-cell exhaustion, carrier transport, classifier determinacy, and ledger exhaustion. These results force any exported verdict through the same observer-state, locality-cell, paired commitment, transport, continuation, provenance, and naming rows, and non-escape rules out every bypass named in the statement. □

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierVerdictSourceExhausti`

Theorem 647.31 (ObserverPerspectiveClassifier public readback criterion)**.** *A local* NameCert$_{\text{ObserverPerspectiveClassif}}$ *read is public exactly when it factors through the displayed observer-state rows O_0, O_1, the universe-for-observer rows U_0, U_1, the locality-cell row L, the paired* GAP *row G, the componentwise* hsame *transports in H, the* Cont *routes in C, the* Pkg *provenance row P, and the local naming row N. Any read omitting one of these rows, or replacing it by an ambient observer-neutral witness, is not the public classifier readback.*

Lean4 ✓ `BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierPublicReadbackCriteri`

Proof. For the forward direction, assume a public readback has been exported. Comparison-route totality and state totality send the read through the two observer-state rows, the two universe-for-observer surfaces, the displayed locality row, the paired commitment row, and then through H, C, P, N.

The carrier transport theorem Definition 647.27 supplies the componentwise hsame and Cont factorization. The classifier determinacy theorem Definition 647.28 then says that the readback is determined only by agreement on these displayed rows, including Pkg provenance and the local NameCert row.

For the reverse direction, suppose the read factors through all displayed rows in the stated order. Ledger exactness reconstructs the public ledger, the obligation unblock package supplies the local naming obligations, and NameCert ledger scope closes the read through the finite NameCert$_{\text{ObserverPerspectiveClassifier}^{\uparrow}}$ row.

Finally apply non-escape and ledger exhaustion. They exclude every ambient replacement, hidden synchronization route, host-equality shortcut, private observer row, and unrecorded cross-perspective witness, so omitting any displayed row leaves the public readback surface. □

Theorem 647.32 (ObserverPerspectiveClassifier sibling carrier dependency)**.** *An* ObserverPerspectiveClassifier$^{\uparrow}$ *verdict cannot be exported until the sibling carrier rows from chapter 578, chapter 643, chapter 509, and chapter 513 have been displayed in the same finite public route. In particular, the classifier may read the two conscious-observer state rows, the public cross-perspective alignment row, and the locality-cell evidence before emitting a verdict, but it cannot replace those sibling rows by an ambient observer-neutral world, unrecorded alignment channel, or hidden locality coordinate.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierSiblingCarrierDepende

Proof. Read the dependency scope theorem Definition 647.12. It identifies the accepted row family with the public observer-state, locality-cell, cross-perspective alignment, BHist, hsame, Cont, Pkg, and NameCert rows.

The alignment-consumer and locality-cell exhaustion theorems then place any downstream comparison through the displayed L, G, H, C, P, N route. Thus the classifier must have already displayed the sibling observer-state, cross-perspective alignment, locality-cell, and observer-locality-cell carrier rows before the verdict is public.

Finally apply non-escape and ledger exhaustion. They remove ambient worlds, hidden synchronization maps, unrecorded alignment channels, and hidden locality coordinates from the accepted packet surface, leaving only the displayed sibling carrier dependency. □

Theorem 647.33 (ObserverPerspectiveClassifier scoped consumer boundary)**.** *The scoped consumer boundary for downstream* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *users of* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *is exactly the finite dependency family* $O_0, O_1, U_0, U_1, L, G, H, C, P, N$ *together with the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspecti}}$ *readback criterion. A downstream consumer may inspect the public verdict only through those observer-state, locality-cell, paired* GAP*, alignment-consumer, transport, continuation, provenance, and naming rows.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierScopedConsumerBoundar

Proof. The verdict source exhaustion theorem Definition 647.30 first identifies the rows from which a public verdict can be sourced. The public readback criterion Definition 647.31 then identifies the exact factorization required for the local naming row to export that verdict.

Apply the existing dependency scope and alignment-consumer theorems. They confine downstream $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ users to the displayed locality, commitment, transport, continuation, provenance, and naming rows, and they prevent a consumer from replacing this route by an unrecorded alignment channel.

The remaining boundary clauses come from locality-cell exhaustion, carrier transport, classifier determinacy, and ledger exhaustion. These theorems show that the carrier, the transported route, the determined verdict, and the exhausted ledger all have the same finite row family. Thus the scoped consumer boundary is precisely the family named in the statement. □

Theorem 647.34 (ObserverPerspectiveClassifier scope certificate)**.** *For an accepted* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *packet* $Q = (O_0, O_1, U_0, U_1, L, G, H, C, P, N)$*, the scoped certificate binds every public observer comparison to the displayed* $\mathsf{ConsciousObserverState}^{\uparrow}$ *rows* O_0, O_1*, the* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *consumer route, the* $\mathsf{LocalityCell}^{\uparrow}$ *or* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row* L*, the finite* BHist *packet* Q*, the componentwise* hsame *ledger* H*, the* Cont *routes* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}$ *row* N*. No observer comparison can bypass the paired* GAP *and alignment rows or the locality-cell rows while remaining a public classifier comparison.*

Lean4 ✓ BEDC.Derived.Observerperspectiveclassifierup.ObserverPerspectiveClassifierScopeCertificate

Proof. Project the carrier definition and the sibling carrier dependency theorem. They display the two conscious-observer state rows, the locality-cell or observer-locality-cell row, the cross-perspective alignment row, the paired GAP ledger, the finite BHist packet, and the local naming row in the same public route.

Next apply the scoped consumer boundary together with alignment transport scope. These results force the comparison through L, G, H, C, P, N, with hsame and Cont carrying only the displayed rows and with Pkg provenance recording the same dependency family.

Finally use the public readback criterion, carrier transport, classifier determinacy, ledger exhaustion, and NameCert ledger scope. They identify the replayed and transported classifier with the same finite packet, and they rule out ambient observer-neutral witnesses, hidden alignment channels, unpaired GAP rows, hidden locality coordinates, and any observer comparison that bypasses the paired alignment and locality-cell rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ packet is generated from two conscious-observer state rows, two universe-for-observer record surfaces, locality-cell evidence, paired GAP commitments, componentwise transports, displayed continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 647.1, Definition 647.2, Definition 647.3, Definition 647.4, Definition 647.5, Definition 647.6, Definition 647.7, Definition 647.8, Definition 647.9, Definition 647.10, Definition 647.11, Definition 647.14, Definition 647.15, Definition 647.16, Definition 647.17, Definition 647.18, Definition 647.19, Definition 647.20, Definition 647.21, Definition 647.22, Definition 647.24, Definition 647.25, Definition 647.26, Definition 647.27, Definition 647.28, and Definition 647.29 bind the finite public classifier obligation surface for universe-for-observer comparison.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverPerspectiveClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge status, god's-eye observation, host equality between worlds, private Self access, global spacetime, hidden synchronization, or unrecorded cross-perspective evidence.

Upgrade path: Scoped closure next requires Lean-side and formal scope witnesses for the same finite observer-state, locality-cell, paired GAP, and alignment-consumer packet.

Chapter 648

A Concrete Naming Certificate for $\mathsf{CompactModulusCover}^{\uparrow}$

The $\mathsf{CompactModulusCover}^{\uparrow}$ interface names the finite source packet that sits between compactness, pointwise continuity, the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ lower-bound fold, and the $\mathsf{UniformModulus}^{\uparrow}$ consumer. It records the compact finite-net probes and the pointwise-modulus ledgers before the rational fold chooses a single uniform precision, so the fold need not also carry the cover-formation certificate.

The packet is consumed by chapter 619 for the positive rational lower-bound fold and by chapter 369 for the final tolerance-only modulus handoff. Its finite-net source is the compact metric surface of chapter 129, so the cover row is a horizontal dependency in the concrete dossier rather than a kernel-only packet.

Definition 648.1 (CompactModulusCover carrier)**.** *A* $\mathsf{CompactModulusCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, E, P, R, U, M, H, C, Q, N)$$

with the following visible rows:

1. *X is the compact source row, read through* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *as a finite-net source over the displayed* $\mathsf{Met}^{\uparrow}$ *distance witnesses;*
2. *F is the continuous map or function row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{CFun}^{\uparrow}$ *with its pointwise-modulus ledger;*
3. *E is the positive tolerance request supplied by the compact-continuous consumer;*
4. *P is the nonempty finite* $\mathsf{ProbeBundle}$ *or compact-net row returned by the compact source certificate;*
5. *R assigns to each bundled probe the source-center history, the covering-radius history, and the local distance witness that covers source histories by that center;*
6. *U assigns to each bundled probe the pointwise-modulus precision row for the tolerance request E at the corresponding center;*
7. *M is the* $\mathsf{UniformModulus}^{\uparrow}$*-facing handoff row containing the same bundle, local radii, pointwise rows, and consumer routes, but no folded precision;*
8. *H is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*-facing radius family consisting of positive* $\mathsf{Rat}^{\uparrow}$ *half-radius ledgers subordinate to the rows in U;*

9. *C records componentwise* hsame *transports for the source row, continuous row, tolerance, bundle, covering rows, pointwise rows, radius family, and handoff row;*
10. *Q records the displayed* Cont *routes for compact-net readback, pointwise-modulus readback, metric-distance use, rational-radius comparison, fold handoff, and uniform-modulus consumption;*
11. *N is the combined provenance row over* BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{CompactModulusCover}^{\uparrow}}$ *certificate.*

The carrier has no coordinate for a host open cover, a selected subcover outside the finite compact-net row, a host-level Lebesgue number, an already folded uniform precision, or a real-completeness theorem. Its role is exactly the finite compact-net and pointwise-modulus source packet consumed by $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *and then by* $\mathsf{UniformModulus}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier`
Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverTasteGate single carrier alignment`

Theorem 648.2 (CompactModulusCover NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactModulusCover}^{\uparrow}}$ *surface for the carrier of Definition 648.1 consists exactly of carrier admission for*

$$K = (X, F, E, P, R, U, M, H, C, Q, N),$$

classifier stability under the componentwise hsame *rows in C, finite-net coverage exactness for the compact source rows in P and R, pointwise-modulus compatibility for the continuous rows in F and U, rational positive-radius compatibility for the* $\mathsf{Rat}^{\uparrow}$ *family H, and consumer non-escape through the displayed* Cont, Pkg, *and local naming rows in Q and N.*

Proof. Project the displayed carrier. Its rows are the compact source, continuous map or function, tolerance request, finite probe bundle, covering rows, pointwise-modulus rows, uniform-modulus handoff row, positive rational radius family, transports, continuation routes, and local provenance.

The compact side is read only through the $\mathsf{Compact}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows, together with the $\mathsf{Met}^{\uparrow}$ distance witnesses attached to R. The continuous side is read only through the $\mathsf{ContinuousMap}^{\uparrow}$ or $\mathsf{CFun}^{\uparrow}$ pointwise-modulus rows stored in U.

The radius family H is the only row passed to the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ lower-bound fold, and the handoff row M is the only row passed to the $\mathsf{UniformModulus}^{\uparrow}$ consumer before that fold has produced a precision. Componentwise hsame transports in C and displayed Cont routes in Q preserve those rows without introducing another selector.

Thus the named obligations exhaust the local certificate surface. Since no carrier coordinate stores a host open cover, a selected subcover, a host-level Lebesgue number, a folded uniform precision, or a real-completeness theorem, the certificate cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier namecert obligations`

Theorem 648.3 (CompactModulusCover finite-net source exhaustion)**.** *For an accepted* $\mathsf{CompactModulusCover}^{\uparrow}$ *carrier*

$$K = (X, F, E, P, R, U, M, H, C, Q, N),$$

every source history used by the $\mathsf{UniformModulus}^{\uparrow}$ *handoff is covered by a bundled center in the finite* ProbeBundle *row P, and every bundled center that contributes a pointwise-modulus row in U contributes exactly one positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger to the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*-facing family H. The exhaustion certificate is internal to P, R, U, H, C, Q, N and does not read an infinite cover or a host subcover choice.*

Proof. Begin with the compact finite-net source row. The $\mathsf{Compact}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$ certificate supplies the nonempty finite bundle P and the covering rows R; each covered source history is related to a bundled center through the displayed $\mathsf{Met}^{\uparrow}$ distance witness.

For each bundled center, read the pointwise-modulus row in U from the $\mathsf{ContinuousMap}^{\uparrow}$ or $\mathsf{CFun}^{\uparrow}$ ledger at the tolerance request E. The rational positive-radius row in H is then the displayed half-radius ledger subordinate to that pointwise row.

The handoff row M records exactly the same bundle and local rows for the $\mathsf{UniformModulus}^{\uparrow}$ consumer. The transports in C and routes in Q replay bundle membership, center coverage, pointwise-modulus readback, and rational-radius comparison componentwise.

Therefore the handoff consumes all and only the finite-net rows in P. No infinite cover, host-selected subcover, or additional compactness object is available from the carrier. □

Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier finite net source exhaustion`

Theorem 648.4 (CompactModulusCover real-completion handoff)**.** *Let a compact-continuous real consumer request a positive tolerance row E for a* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{CFun}^{\uparrow}$ *row over a compact* $\mathsf{Met}^{\uparrow}$ *source. An accepted* $\mathsf{CompactModulusCover}^{\uparrow}$ *carrier hands the consumer a finite* $\mathsf{UniformModulus}^{\uparrow}$ *source ledger and a* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *radius family, both over the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance. This handoff is sufficient for the later rational lower-bound fold and real-uniform-modulus read, and it does not claim Cauchy completion, host real completeness, or a global modulus chosen outside the packet.*

Proof. Read the accepted carrier K. The compact side supplies the finite source coverage through X, P, R; the continuous side supplies pointwise-modulus rows through F, U; and the tolerance request is the displayed row E.

Apply Definition 648.3. It aligns the finite compact-net bundle with the pointwise rows and the positive rational radius family, so the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ consumer receives precisely the family H it needs for the lower-bound fold.

The $\mathsf{UniformModulus}^{\uparrow}$ consumer reads M together with the same transport, route, provenance, and local naming rows C, Q, N. After the lower-bound fold has supplied a single precision, the consumer can route the usual compact-uniform-modulus argument through the bundled metric witnesses and pointwise rows.

No completion step is part of this packet. Real consumers may use the resulting uniform-modulus ledger downstream, but the $\mathsf{CompactModulusCover}^{\uparrow}$ carrier itself exports only the finite compact-net and pointwise-modulus source packet. □

Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier real completion handoff`

Theorem 648.5 (CompactModulusCover obligation triad)**.** *For an accepted* $\mathsf{CompactModulusCover}^{\uparrow}$ *carrier*

$$K = (X, F, E, P, R, U, M, H, C, Q, N),$$

the consumer surface before $\mathsf{UniformModulus}^{\uparrow}$ *exposes exactly three obligations: compact finite-net coverage through X, P, R, pointwise-modulus compatibility through F, E, U, and rational lower-bound fold handoff through H, M, C, Q, N. It exports no host open cover, host-selected subcover, Lebesgue number, folded uniform precision, or real-completeness theorem.*

Proof. First read the NameCert obligations. They identify carrier admission, classifier stability, finite-net coverage, pointwise-modulus compatibility, rational positive-radius compatibility, and consumer non-escape as the local certificate surface.

Next separate the three consumer obligations. Finite-net source exhaustion supplies the compact coverage route through X, P, R; the pointwise rows F, E, U provide the local modulus compatibility; and the real-completion handoff sends the radius family and structural rows through H, M, C, Q, N toward the lower-bound fold.

The non-escape clauses in those theorems rule out every unlisted analytic object. Since the carrier contains no host open cover, host-selected subcover, Lebesgue number, folded uniform precision, or real-completeness theorem, the triad is the whole pre-$\mathsf{UniformModulus}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier obligation triad`

Theorem 648.6 (CompactModulusCover AI TasteGate scope)**.** *Any obligation-closure read for* $\mathsf{CompactModulusCover}^{\uparrow}$ *must pass through the same single-carrier packet*

$$K = (X, F, E, P, R, U, M, H, C, Q, N)$$

of Definition 648.1. The read must include the finite-net source exhaustion of Definition 648.3 and must preserve carrier alignment for the finite-net rows, pointwise-modulus rows, fold handoff, transport rows, route rows, and local NameCert *row before any closure promotion is recorded.*

Proof. Project the carrier K. The compact finite-net source is read from X, P, R, and the pointwise-modulus data are read from F, E, U.

Apply finite-net source exhaustion. It ties every source history used by the $\mathsf{UniformModulus}^{\uparrow}$ handoff to the finite bundle P, and it ties every bundled center with a pointwise row in U to the positive rational radius family H.

The fold-facing and consumer-facing rows are then fixed. The row H is the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ radius family, and the row M is the $\mathsf{UniformModulus}^{\uparrow}$ handoff row before a folded precision is available.

Finally read the structural rows C, Q, N. They provide componentwise transport, displayed routes, and the local naming surface for the same packet, so an obligation-closure read cannot bypass the displayed carrier or use a second source packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusCover}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusCover}^{\uparrow}$ packet is generated from a compact source row, a continuous map or function row, a positive tolerance request, a finite ProbeBundle, compact-net coverage rows, pointwise-modulus rows, positive $\mathsf{Rat}^{\uparrow}$ radius ledgers, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactModulusCover}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 648.1, Definition 648.2, Definition 648.3, Definition 648.4, and Definition 648.5 bind the carrier, NameCert obligations, finite-net exhaustion, real-completion handoff, and obligation-triad rows to the displayed compact-net, pointwise-modulus, lower-bound-fold, and uniform-modulus scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactModulusCover}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.CompactModulusCoverUp.taste gate`

Lean target: `BEDC.Derived.CompactModulusCoverUp.CompactModulusCoverCarrier\_namecert\_obligations`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the rational lower-bound fold, the final uniform-modulus theorem, Cauchy completion, host real completeness, external subcover selection, or bridge checking.

Upgrade path: The paper axis is closed at the finite compact-net, pointwise-modulus, rational-radius, lower-bound-fold, and uniform-modulus consumer surface; only bridge checking remains on the formal axis.

Chapter 649

A Concrete Naming Certificate for $\mathsf{StationaryWindowLattice}^{\uparrow}$

The $\mathsf{StationaryWindowLattice}^{\uparrow}$ packet records the common finite-window refinement used by stationary Real-completeness diagonal consumers. It joins stationary rational seed rows, transported stationary-window seal rows, mesh-budget rows, witness-schedule rows, cofinal-tail rows, transports, continuations, provenance, and local naming into one finite BHist carrier. The packet is a refinement lattice over displayed windows; it is not an infinite diagonal object, a quotient stream, a selected host limit, or an ambient completeness principle.

The transported seal coordinate is the sibling surface of chapter 437: that chapter supplies the stationary-window seal row, while the present chapter adds the common refinement meet that synchronizes it with mesh, witness-schedule, and cofinal-tail consumers.

Definition 649.1 (Stationary window lattice carrier)**.** *A* $\mathsf{StationaryWindowLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, S, R, D, W, M, H, C, P, N)$$

with the following displayed rows:

1. *Q is the stationary* $\mathsf{Rat}^{\uparrow}$ *seed row whose rational histories determine the constant diagonal source;*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected from Q;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the same selected windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the endpoint, radius, and comparison rows used by the selected windows;*
5. *W is the transported stationary-window seal row linking the stationary diagonal window packet to the* $\mathsf{Real}^{\uparrow}$ *seal surface;*
6. *M is the common refinement-meet row that simultaneously refines the mesh budget, witness schedule, cofinal tail, and transported stationary window rows;*
7. *H records componentwise* hsame *transports for Q, S, R, D, W, M and the structural rows;*
8. *C records the displayed* Cont *routes from stationary seeds to windows, from windows to readbacks, from readbacks to dyadic tolerance rows, from those rows to the transported seal, and from all selected windows to the common refinement-meet row;*

9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, TransportedStationaryWindowSe RealDiagonalMeshBudget$^{\uparrow}$, $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$, $\mathsf{DiagonalCofinalTail}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryWindowLattice}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host real equality, unbounded window families, selected limits, choice data, or ambient Real-completeness witnesses.

Theorem 649.2 (Stationary window lattice NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryWindowLattice}^{\uparrow}}$ *surface for Definition 649.1 consists exactly of carrier admission for $L = (Q, S, R, D, W, M, H, C, P, N)$, stationary* $\mathsf{Rat}^{\uparrow}$ *seed exposure through Q, selected* $\mathsf{StreamName}^{\uparrow}$ *window exposure through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger exactness through D, transported stationary-window seal exposure through W, common refinement-meet exposure through M, componentwise* hsame *transport through H, displayed* Cont *route replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.StationaryWindowLatticeUp.StationaryWindowLatticeTasteGate single carrier alignment`

Proof. Project the carrier of Definition 649.1. The rows Q, S, R, D, W, M list the mathematical data available to the lattice consumer, and H, C, P, N list the transport, route, provenance, and naming data.

The stationary source is read first through Q, then through the selected finite windows S, the regular-rational readback R, and the dyadic tolerance ledger D. The transported seal row W can be reached only after those rows are visible.

The refinement row M is the only row that compares the mesh budget, witness schedule, cofinal tail, and transported stationary window data. The remaining rows H, C, P, N preserve that comparison componentwise and record the local NameCert surface. Since the carrier has no other coordinates, the displayed rows exhaust the obligations. □

Theorem 649.3 (Stationary window lattice refinement meet)**.** *For an accepted* $\mathsf{StationaryWindowLattice}^{\uparrow}$ *carrier $L = (Q, S, R, D, W, M, H, C, P, N)$, the row M is the common finite refinement for every stationary-window consumer that reads the mesh budget, witness schedule, cofinal tail, or transported stationary-window seal. Any such consumer must factor its selected windows through S, R, D, W and then through M before a* $\mathsf{Real}^{\uparrow}$ *seal-facing row is exported.*

Proof. Read S, R, D, W from the carrier. These rows are the selected finite windows, the regular-rational readbacks, the dyadic tolerance ledger, and the transported stationary-window seal surface.

Use the NameCert obligation theorem. It identifies M as the only common refinement-meet row and routes the displayed continuations in C through that row. Thus mesh-budget, witness-schedule, cofinal-tail, and transported-window consumers cannot choose independent finite windows after M has been fixed.

Componentwise transport through H preserves the same selected rows, while P and N record provenance and local naming. Hence the meet is a finite refinement of the displayed window data and not a second source of diagonal information. □

Theorem 649.4 (Stationary window lattice Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-seal consumer of an accepted* $\mathsf{StationaryWindowLattice}^{\uparrow}$ *packet receives the transported stationary-window seal row only through the displayed route*

$$Q, S, R, D, W, M, H, C, P, N.$$

The handoff preserves the same refinement meet M for the stationary diagonal window, mesh budget, witness schedule, and cofinal-tail surfaces.

Proof. Start from the transported stationary-window seal row W. It sits in the same finite packet as the stationary rational seed Q, the selected windows S, the regular readback R, the dyadic ledger D, and the refinement meet M.

Apply Definition 649.3. Every seal-facing read that uses the stationary window data must pass through M after the selected windows, readbacks, and tolerance ledger have been exposed.

The rows H and C then replay the displayed transports and continuations, and P, N provide provenance and local naming. There is no carrier coordinate for a different Real seal route, so the handoff is exactly the displayed finite route. □

Theorem 649.5 (Stationary window lattice non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{StationaryWindowLattice}^{\uparrow}$ *packet factors through the finite rows*

$$Q, S, R, D, W, M, H, C, P, N.$$

The consumer may read the stationary rational seed, finite StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerance ledger, transported stationary-window seal, common refinement meet, hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row, and it cannot read quotient streams, host real equality, unbounded window families, selected limits, choice data, or ambient Real completeness.*

Proof. Project the accepted carrier and the local NameCert surface. The rows $Q, S, R, D, W, M, H, C, P, N$ are the complete displayed packet.

Use the refinement-meet theorem for window-facing consumers. It confines mesh-budget, witness-schedule, cofinal-tail, and transported-window reads to the same selected finite windows and to the common meet row M.

Use the Real seal handoff theorem for seal-facing consumers. It routes every $\mathsf{Real}^{\uparrow}$ or completion-seal read through the displayed finite route before the transported seal is exported.

No remaining carrier coordinate exists beyond those rows. Therefore no downstream consumer can extract quotient-stream data, host real equality, unbounded window data, selected limits, choice data, or ambient completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryWindowLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{StationaryWindowLattice}^{\uparrow}$ packet is generated from stationary $\mathsf{Rat}^{\uparrow}$ seed rows, selected $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, transported stationary-window seal rows, a common refinement-meet row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StationaryWindowLattice}^{\uparrow}}$, seed)

Scope closed: Definition 649.1, Definition 649.2, Definition 649.3, Definition 649.4, and Definition 649.5 bind the finite common-refinement lattice for stationary Real-completeness window consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StationaryWindowLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, host real equality, unbounded diagonal windows, selected limits, choice data, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, common refinement-meet exactness, Real seal handoff, and non-escape for the same stationary finite-window lattice packet.

Chapter 650

A Concrete Naming Certificate for $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$

The $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ packet records the finite precision-to-window route used by Real-completion consumers before a $\mathsf{Real}^{\uparrow}$ seal is reached. It keeps the precision request, schedule row, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, transports, routes, provenance, and local naming row in one displayed BHist packet, so the limit-facing surface is finite and contains no choice operator.

The concrete completion consumer is the RealCompleteness finite seal route of chapter 450. That consumer may read the schedule row, selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic observations, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal exposed by $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$, but it receives no countable choice, hidden selector, or ambient completeness coordinate from the schedule packet.

Definition 650.1 (Choice-free limit schedule carrier)**.** *A* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, M, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the precision request row naming the finite tolerance level demanded by a completion-facing consumer;*
2. *M is the schedule row assigning the request E to the finite observation route used at that tolerance;*
3. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected by the schedule;*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations and radius ledgers read from the windows in W;*
5. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows reached after those dyadic observations;*
6. *S records the* $\mathsf{Real}^{\uparrow}$ *seal row that receives the handoff;*
7. *H records componentwise* hsame *transports for the request, schedule, windows, dyadic observations, regular-sequence handoff, seal, provenance, and local name;*
8. *C records the displayed* Cont *routes from E to M, from M to W, from W to D, from D to R, and from R to S;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It has no coordinate for quotient streams, host equality of completed reals, countable choice, a hidden limit selector, or an ambient completeness theorem.

Theorem 650.2 (Choice-free limit schedule NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}$ *surface for Definition 650.1 consists of five finite obligations:*

1. *carrier exposure, requiring every accepted object to display the packet* $L = (E, M, W, D, R, S, H, C, P, N)$*;*
2. *request-to-schedule exactness, requiring the precision request E to reach the schedule row M only through the displayed route in C;*
3. *window and dyadic readback coverage, requiring M to select only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W and requiring every exported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in D to be read from those windows;*
4. *handoff and seal discipline, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows R and the* $\mathsf{Real}^{\uparrow}$ *seal row S to be reached through D and then transported by the displayed H, C, P, N rows;*
5. *non-escape, requiring every public read to stay inside the displayed packet and to export no quotient stream, host real equality, countable-choice row, hidden limit selector, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeLimitScheduleUp.ChoiceFreeLimitSchedulePacket namecert obligations`

Proof. Carrier exposure is projection from the tuple in Definition 650.1. The packet lists the precision request, schedule, windows, dyadic observations, regular-sequence handoff, real seal, transports, routes, provenance, and local naming row.

Request-to-schedule exactness is the first displayed route in C. Since the carrier has one schedule row M for the request E, a consumer has no alternate coordinate from which to obtain a different schedule for the same precision request.

Window and dyadic readback coverage use the next two route segments. The route from M to W exposes only the finite window family, and the route from W to D makes the dyadic observation ledger depend on those windows rather than on an unbounded stream object.

Handoff and seal discipline are supplied by R, S, H, C, P, N. The regular-sequence handoff is reached after the dyadic ledger, the real seal is reached after the handoff, and transport or repacking must replay the same rows through H, C, P, N.

The non-escape clause follows from the carrier inventory. None of the displayed rows is a quotient stream, host real equality test, countable-choice row, hidden selector, or ambient completeness theorem, so the local naming certificate has no route that can export one. □

Theorem 650.3 (Choice-free finite schedule induction)**.** *Every accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *package is generated by the explicitly displayed finite schedule row M of the carrier*

$$L = (E, M, W, D, R, S, H, C, P, N).$$

Extending the displayed schedule by one visible schedule entry preserves carrier admission, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}$ *rows. The induction surface does not select a limit, quotient a completion, invoke countable choice, or identify host real equality.*

Proof. Induct on the displayed finite schedule list encoded by M.

In the base case the schedule is empty. Carrier admission is the empty finite packet route from the request row E to the displayed window family W, dyadic ledger D, handoff row R, Real seal S, and structural rows H, C, P, N; no selector or completed limit is present.

For the step case, append one displayed schedule entry. The appended entry contributes one more visible request-to-window route inside M and W, and the dyadic readback row D records only the corresponding finite observation. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and $\mathsf{Real}^{\uparrow}$ seal S are still reached through the same displayed route.

Transport and replay are componentwise. The row H transports the old schedule rows and the appended entry together, while C replays the extended finite route. The provenance row P and local naming row N then repack the extended packet as another accepted $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ carrier.

The excluded objects are absent in both cases. Neither the empty schedule nor the appended-entry step introduces a selected limit, quotient completion, countable-choice row, hidden selector, or host real equality coordinate. □

Theorem 650.4 (Choice-free limit schedule non-escape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumer of an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet factors through the finite route*

$$E, M, W, D, R, S, H, C, P, N.$$

The consumer may inspect the precision request, schedule row, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It cannot inspect quotient streams, host equality of completed reals, countable choice, hidden limit selectors, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeLimitScheduleUp.ChoiceFreeLimitSchedulePacket nonescape`

Proof. Start from the accepted carrier $L = (E, M, W, D, R, S, H, C, P, N)$. The only request-facing route begins at E and reaches the schedule row M through the displayed continuation coordinate.

Next read the finite observation surface. The schedule selects the windows in W, and the dyadic ledger D is read from those windows. Thus a consumer cannot replace the displayed window family by a quotient stream or an unbounded stream-level selector.

Pass to the completion-facing rows. The handoff row R and seal row S are reached after the dyadic ledger and are repacked only through H, C, P, N, so the route into $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ remains the same finite packet.

Finally apply the non-escape obligation of Definition 650.2. The excluded analytic objects do not occur as coordinates of L, so no public read can export quotient streams, host equality of completed reals, countable choice, hidden limit selectors, or an ambient completeness theorem. □

Theorem 650.5 (Choice-free limit schedule classifier transport)**.** *Let* $L = (E, M, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet. The schedule classifier of* L *is the finite public test that reads exactly the displayed request row* E*, schedule row* M*, selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* S*, componentwise transports* H*, continuation routes* C*, provenance* P*, and local name* N*.*

If each of these rows is transported componentwise by the corresponding hsame *coordinate in* H*, the classifier is preserved. The transported packet still has no coordinate from which a quotient stream, host real equality of completed reals, countable choice, hidden limit selector, or ambient completeness theorem can be introduced.*

Proof. The carrier definition fixes the classifier domain to the rows of the packet L. Since the classifier reads no object outside $E, M, W, D, R, S, H, C, P, N$, transporting the displayed rows componentwise through H preserves exactly the same finite test.

Apply the NameCert obligations of Definition 650.2. Carrier exposure, request-to-schedule exactness, window and dyadic readback coverage, handoff and seal discipline, and non-escape are all stated for the same displayed rows.

Finally use Definition 650.4. Every public read remains inside the finite packet after transport, so there is no route that can add quotient streams, host real equality, countable choice, hidden selectors, or ambient completeness. □

Theorem 650.6 (Choice-free limit schedule RegSeqRat handoff exactness)**.** *Let $L = (E, M, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row in R is reached only through the displayed route*

$$E \to M \to W \to D$$

together with the transport, route replay, provenance, and local naming rows H, C, P, N.

Thus no downstream consumer can read a handoff row through an unlisted stream representative, a quotient-stream equality, a countable-choice selector, host real equality, or an ambient completeness principle.

Proof. Start with the carrier definition. The row R occurs after the precision request E, schedule row M, window family W, and dyadic ledger D, and the continuation coordinate C displays the route from E to M, from M to W, from W to D, and from D to R.

Apply the NameCert obligations. Request-to-schedule exactness fixes the first segment, window and dyadic readback coverage fixes the middle segments, and handoff discipline fixes the passage from D to R under the same H, C, P, N rows.

Use non-escape to close the boundary. Since the handoff is read through the displayed packet, the packet cannot export an unlisted representative, quotient equality, choice selector, host real equality, or ambient completeness theorem. □

Theorem 650.7 (Choice-free limit schedule RealUp seal discipline)**.** *Let $L = (E, M, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal read through S factors through the finite route*

$$E \to M \to W \to D \to R$$

and then through the same componentwise transport, continuation, provenance, and local naming rows H, C, P, N.

The exported seal is only the displayed $\mathsf{Real}^{\uparrow}$ *row of the packet. It does not provide a stronger completion object, a uniqueness theorem for completed limits, quotient equality of streams, host real equality, countable choice, a hidden selector, or ambient completeness.*

Proof. Project the seal coordinate from the carrier L. The definition places S after the request, schedule, selected windows, dyadic observations, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the continuation coordinate records the route into S.

Apply the NameCert obligations for handoff and seal discipline. They require the seal row to be reached through D and R and then repacked only through the displayed H, C, P, N rows.

Apply Definition 650.4. A completion-facing consumer may read the finite route and the seal row, but no stronger completion object or excluded analytic principle is present in the packet. □

Theorem 650.8 (Choice-free limit schedule consumer readback)**.** *Let $L = (E, M, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet. Every completion-facing consumer read of L factors through the same displayed route*

$$E \to M \to W \to D \to R \to S$$

and through the same componentwise transport, continuation, provenance, and local naming rows H, C, P, N.

Consequently the consumer-facing surface of the packet is the finite schedule-to-seal route together with H, C, P, N. It exposes no alternate schedule classifier, no separate $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, no stronger* $\mathsf{Real}^{\uparrow}$ *completion object, no quotient-stream equality, no host real equality, no countable-choice row, no hidden selector, and no ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeLimitScheduleUp.ChoiceFreeLimitSchedulePacket consumer readback`

Proof. Begin with the carrier and NameCert obligations. Carrier exposure fixes the displayed packet, request-to-schedule exactness fixes the segment from E to M, window and dyadic readback coverage fixes the segments from M to W and from W to D, and handoff and seal discipline fixes the segments from D to R and from R to S under the same H, C, P, N rows.

Use classifier transport to keep the public test on those rows after componentwise transport. By Definition 650.5, the transported packet still reads exactly $E, M, W, D, R, S, H, C, P, N$ and cannot add an alternate analytic coordinate.

Apply Definition 650.6 and Definition 650.7. The handoff and seal are both reached by replaying the finite route, so a completion-facing read cannot bypass the schedule, the selected windows, or the dyadic ledger.

Finally apply Definition 650.4. Since all public reads remain inside the displayed packet, the consumer readback exports only the finite route and the displayed H, C, P, N rows. □

Theorem 650.9 (Choice-free limit schedule public seal export)**.** *The carrier of Definition 650.1, the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}$ *obligations of Definition 650.2, the classifier transport row of Definition 650.5, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness row of Definition 650.6, the* $\mathsf{Real}^{\uparrow}$ *seal discipline row of Definition 650.7, the non-escape boundary of Definition 650.4, and the consumer readback row of Definition 650.8 assemble one public finite schedule-to-seal export for* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers.*

The exported surface is exactly the displayed route

$$E \to M \to W \to D \to R \to S$$

together with the componentwise transport, continuation, provenance, and local naming rows H, C, P, N. It exports no quotient stream, host equality of completed reals, countable-choice row, hidden limit selector, stronger completion object, or ambient completeness theorem.

Proof. Start with the finite carrier. Definition 650.1 exposes the precision request E, schedule row M, selected windows W, dyadic observations D, regular-sequence handoff R, real seal S, transports H, routes C, provenance P, and local naming row N as the only coordinates of the packet.

The certificate obligations make this route public. Apply Definition 650.2 to fix carrier exposure, request-to-schedule exactness, window and dyadic readback coverage, handoff and seal discipline, and the non-escape boundary for those rows.

Transport and handoff are then read through the same packet. By Definition 650.5, componentwise hsame transport preserves the classifier on $E, M, W, D, R, S, H, C, P, N$. By Definition 650.6

and Definition 650.7, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal are reached only by replaying the displayed finite route.

Finally use Definition 650.8 and Definition 650.4. Completion-facing consumers read exactly the finite schedule-to-seal surface and cannot export any excluded analytic object outside the displayed rows. □

Theorem 650.10 (Choice-free limit schedule CauchyLimitSeal factorization)**.** *Every accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet whose finite route reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* S *factors the seal-facing consumer through the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *surface of chapter 386 before any Real-completion consumer reads the finite schedule-to-seal export.*

Thus a Real-completion consumer receives the public surface of Definition 650.9 only after the consumer readback row of Definition 650.8 has been confined to the Cauchy-limit seal boundary. The factorization adds no quotient stream, host real equality, countable choice, hidden selector, or ambient completeness theorem.

Proof. Start with an accepted packet $L = (E, M, W, D, R, S, H, C, P, N)$ whose route reaches the $\mathsf{Real}^{\uparrow}$ seal row S. By Definition 650.8, every completion-facing read of L is already forced through the displayed route

$$E \to M \to W \to D \to R \to S$$

and through the same componentwise transport, continuation, provenance, and local naming rows H, C, P, N.

Now place the seal-facing end of that route at the Cauchy-limit seal boundary supplied by chapter 386. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface is the consumer boundary between regular-source readback and Real-facing seal consumption, so the row S is read only after the finite schedule route has been exposed at that surface.

Finally apply Definition 650.9. The exported Real-completion surface remains exactly the finite schedule-to-seal route with H, C, P, N, now read through the displayed Cauchy-limit seal boundary first. Since neither the schedule packet nor the cited seal surface supplies quotient streams, host real equality, countable choice, hidden selectors, or an ambient completeness theorem, the factorization exports none of them. □

Theorem 650.11 (Choice-free limit schedule ProbeBundle fusion)**.** *Let* $L = (E, M, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet, and let* $\mathcal{Q}$ *be a finite* $\mathsf{ProbeBundle}$ *of unary precision requests for completion-facing seal reads of* L*. All reads requested by* $\mathcal{Q}$ *fuse into one finite schedule-to-seal surface consisting exactly of the schedule row* M*, the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* S*, and the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows* H, C, P, N*.*

Thus the bundled readback consumes the same public route

$$M \to W \to D \to R \to S$$

for every member of $\mathcal{Q}$*, with the precision-request row* E *supplying the accepted unary request surface. The fused bundle exports no quotient stream, host real equality, countable-choice row, hidden selector, selected host limit, ambient completeness theorem, or second completion route.*

Proof. Read the finite $\mathsf{ProbeBundle}$ one request at a time. For each unary precision request in $\mathcal{Q}$, Definition 650.8 places the completion-facing read on the displayed route through $E, M, W, D, R, S, H, C, P, N$.

Package those pointwise reads through the same bundle surface. Because $\mathcal{Q}$ is finite and every member names a seal read of the same accepted packet L, the schedule row M, window family

W, dyadic ledger D, handoff R, seal S, and rows H, C, P, N are reused rather than replaced by request-specific carriers.

Apply Definition 650.9. It identifies the common public route for Real-facing consumers as the finite schedule-to-seal surface with exactly the displayed transport, continuation, provenance, and local naming rows.

Now use Definition 650.10. The seal-facing end of the bundled route is read through the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary before any Real-completion consumer observes the public seal, so the bundle does not open a second completion route.

Finally apply Definition 650.14. Since hidden selector coordinates are absent from the packet, the finite bundle cannot extract quotient streams, host real equality, countable choice, selected host limits, ambient completeness, or any additional completion surface. □

Theorem 650.12 (Choice-free limit schedule shared-tail meet compatibility)**.** *Let $L = (E, M, W, D, R, S, H, C, P, N$ be an accepted $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ packet. Suppose the finite route exported by L is consumed by a $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold packet and then sealed through the $\mathsf{CauchyLimitSeal}^{\uparrow}$ tail-budget surface. After passing to the shared-tail threshold, the displayed route remains*

$$E \to M \to W \to D \to R \to S$$

with the same selected $\mathsf{StreamName}^{\uparrow}$ windows W, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, $\mathsf{Real}^{\uparrow}$ seal S, and structural rows H, C, P, N.

Thus Real-completion diagonal consumers read the shared-tail meet as a compatibility layer for the finite schedule-to-seal packet, not as a second completion route. The compatible export contains no quotient stream, host real equality, hidden selector, countable choice, ambient completeness datum, or second seal route.

Proof. Start with the finite packet readback. By Definition 650.8, every completion-facing read of L is confined to the displayed route through $E, M, W, D, R, S, H, C, P, N$.

Next attach the Cauchy-limit seal boundary. The factorization theorem Definition 650.10 places the seal-facing end of the route at the $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface before any Real-completion consumer reads the public seal.

Now pass to the shared-tail meet. The public shared-threshold certificate and consumer exactness theorems for $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, Definition 560.11 and Definition 560.34, keep the threshold read attached to its displayed source, window, modulus, threshold, comparison, transport, continuation, ledger, provenance, and naming rows. The selector-budget public route Definition 560.14 identifies the selector-facing entry with the same public threshold rather than a hidden tail oracle.

Finally use the tail-budget and selector rows on the seal side. The CauchyLimitSeal tail-budget meet and selector-budget transport, Definition 386.135 and Definition 386.136, preserve the shared $\mathsf{StreamName}^{\uparrow}$ window, dyadic ledger, regular handoff, Real seal, and displayed structural rows. Together with Definition 386.108 and the finite $\mathsf{ProbeBundle}$ fusion of Definition 650.11, this repacks all requested reads into the same finite schedule-to-seal surface and leaves no coordinate for the excluded quotient, host-equality, selector, choice, completeness, or second-seal data. □

Theorem 650.13 (Choice-free limit schedule well-founded readback)**.** *Every completion-facing readback of an accepted $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ packet follows the finite route*

$$E \to M \to W \to D \to R \to S.$$

The read descends structurally through the displayed schedule row M and the finite window family W, then through the dyadic ledger D, the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, and the* $\mathsf{Real}^{\uparrow}$ *seal S. Its well-foundedness is therefore measured on the finite packet of Definition 650.1, not on a selected limit object.*

Proof. Start with the carrier $L = (E, M, W, D, R, S, H, C, P, N)$ from Definition 650.1. The only completion-facing spine displayed by the packet is the route from the precision request to the schedule, finite windows, dyadic observations, regular-sequence handoff, and real seal.

Apply the NameCert obligations of Definition 650.2. Request-to-schedule exactness fixes $E \to M$, window and dyadic readback coverage fixes $M \to W \to D$, and handoff and seal discipline fixes $D \to R \to S$.

Use Definition 650.6, Definition 650.7, and Definition 650.9. These rows make the consumer read replay the displayed route, so the descent is over the finite schedule and window rows rather than over a chosen completed limit. □

Theorem 650.14 (Choice-free limit schedule no selector extraction)**.** *No accepted* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *packet exports a hidden selector from precision requests to representatives. All exported data are the schedule row M, finite window family W, dyadic ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal S, and the transport, route, provenance, and local naming rows H, C, P, N.*

Proof. Project the accepted packet through Definition 650.1. Its public coordinates are exactly $E, M, W, D, R, S, H, C, P, N$, and no coordinate is a selector from requests to hidden representatives.

The obligations of Definition 650.2 keep each public read inside those rows. In particular, request-to-schedule exactness exposes only the schedule row, and window and dyadic readback coverage exposes only the finite windows and dyadic ledger selected by that schedule.

The handoff and seal are then constrained by Definition 650.6 and Definition 650.7. Finally Definition 650.9 states that the exported surface is the finite schedule-to-seal route with H, C, P, N, so a hidden selector is not part of the public packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ packet is generated from a precision request, finite schedule row, StreamNameUp windows, DyadicRatCoreUp observations, RegSeqRatUp handoff, RealUp seal, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 650.9 binds the public finite schedule-to-seal export from precision request through schedule, windows, dyadic ledger, RegSeqRatUp handoff, RealUp seal, and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ChoiceFreeLimitScheduleUp.ChoiceFreeLimitSchedulePacket\_consumer\_readback`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, a public standard bridge, countable choice, audit cleanliness, or machine bridge checking.

Upgrade path: No further paper-side theory-grade target is open; verification-axis audit cleanliness or machine bridge checking remains separate.

Chapter 651

A Concrete Naming Certificate for InscriptionAcceptanceBudget$^{\uparrow}$

The InscriptionAcceptanceBudget$^{\uparrow}$ surface isolates the accepted-check and downstream consumer-budget part of an inscription event. It names the finite public packet by which a prior source state reaches a named record through a route witness, receives an accepted check, spends only the displayed consumer budget, and leaves the remaining GAP residue visible to later audit.

This packet consumes the route-witness and accepted-record rows of chapter 519 together with the visible GAP-residue ledger of chapter 520. The source, name, route, and check rows come from InscriptionEvent$^{\uparrow}$, while the residue row Q is the InscriptionGapLedger$^{\uparrow}$ boundary consumed by this budget packet.

Definition 651.1 (InscriptionAcceptanceBudget carrier)**.** *An* InscriptionAcceptanceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, M, R, K, U, B, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the prior source-state row from which the inscription route is read;*
2. *M is the named statement, theorem site, citation row, or public record admitted by the check;*
3. *R is the displayed route witness from S to M;*
4. *K is the accepted check row that admits R as public record data for M;*
5. *U is the intended downstream consumer row;*
6. *B is the finite consumer-budget row naming the uses of K and M available to U;*
7. *Q is the remaining* GAP *residue row after the accepted route and consumer budget have been exposed;*
8. *H records componentwise* hsame *transports for S, M, R, K, U, B, Q together with package and naming rows;*
9. *C records the displayed* Cont *routes through which consumers read the source, name, route, check, budget, and residue rows;*
10. *P is the* Pkg *provenance over* BHist, hsame, Cont, *accepted-check data, consumer-budget data,* GAP *residue data, and local certificate dependencies;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionAcceptanceBudget}^{\uparrow}}$ *row.*
The classifier for accepted carriers compares only $S, M, R, K, U, B, Q, H, C, P, N$. It has no coordinate for private intention, host trust, unbudgeted downstream use, unchecked route replay, hidden mutation, or a complete inscription-event ontology.

Theorem 651.2 (InscriptionAcceptanceBudget check row exactness)**.** *For every accepted* InscriptionAcceptanceBud *carrier $A = (S, M, R, K, U, B, Q, H, C, P, N)$, the accepted-check row K is exactly the public route by which R is admitted from S to the named record M. Every public comparison of check rows is transported componentwise by H and read through the displayed routes in C, and the carrier exports no second checker, hidden host acceptance predicate, unchecked route, or unrecorded proof replay.*

Proof. Project the carrier of Definition 651.1. The only row whose role is route admission is K, and it is displayed beside the source row S, named record M, and route witness R.

The comparison surface is componentwise. The hsame rows in H transport S, M, R, K together with the budget, residue, package, and local naming rows, while the Cont routes in C replay the same public route into the accepted check.

No coordinate of A names another checker, a host acceptance predicate, or an unchecked replay route. Thus every accepted-check consumer reads exactly K through the displayed packet. □

Theorem 651.3 (InscriptionAcceptanceBudget consumer boundary)**.** *Every downstream use of an accepted* $\mathsf{InscriptionAcceptanceBudget}^{\uparrow}$ *carrier is bounded by the intended consumer row U and the finite consumer-budget row B. A theorem site, citation row, audit surface, or certificate consumer may read the named record M, route witness R, accepted check K, budget B, and residue Q only through U, H, C, P, N; it receives no unbudgeted consumer right, private agency predicate, hidden host trace, or replacement for the* GAP *residue.*

Proof. Begin with a public downstream request. The intended consumer is the row U, and the finite budget of permitted reads is the row B.

The route into the accepted material then passes through C. It exposes M, R, K together with Q under the componentwise transports in H and the provenance rows P, N.

Because the carrier contains no unbudgeted consumer coordinate and no hidden host trace, the consumer boundary is exhausted by U and B. The remaining residue is still the displayed row Q, not a private explanation outside the packet. □

Theorem 651.4 (InscriptionAcceptanceBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscriptionAcceptanceBudget}^{\uparrow}}$ *surface for Definition 651.1 consists of five finite obligations:*

1. *carrier exposure: every accepted packet exposes exactly $A = (S, M, R, K, U, B, Q, H, C, P, N)$;*
2. *check-row exactness: the accepted check is the displayed row K relating S, R, and M;*
3. *consumer-budget exhaustion: every downstream use routes through U and spends only the finite row B;*
4. GAP *residue non-escape: the residue is the displayed row Q and cannot be replaced by private intention, hidden agency, raw byte identity, or host process trace;*
5. *downstream route stability: all public reads are preserved by the componentwise* hsame *transports in H and the* Cont *routes in C over the same package and local naming rows P, N.*

Proof. Carrier exposure is projection from the displayed finite tuple. The source, name, route, check, consumer, budget, residue, transport, continuation, package, and local naming rows are all coordinates of one BHist packet.

Check-row exactness is exactly Definition 651.2. Consumer-budget exhaustion is exactly Definition 651.3; both read the same rows $S, M, R, K, U, B, Q, H, C, P, N$.

For residue non-escape and route stability, inspect the remaining carrier inventory. The only residue coordinate is Q, and the only public comparison and consumer routes are the displayed rows H and C with provenance P, N. □

Theorem 651.5 (InscriptionAcceptanceBudget GAP residue exposure)**.** *For every accepted* InscriptionAcceptanceB *carrier $A = (S, M, R, K, U, B, Q, H, C, P, N)$, the* GAP *residue row Q is the only residue coordinate exposed to downstream consumers. It cannot be replaced by private intention, hidden agency, raw byte identity, or host process trace.*

Proof. Project the carrier of Definition 651.1. The only displayed row whose role is remaining GAP residue is Q.

The check-row exactness theorem, Definition 651.2, confines acceptance to K over S, R, and M. The consumer boundary theorem, Definition 651.3, lets downstream readers see residue only through the public packet and its displayed routes.

Finally use Definition 651.4. Its residue non-escape obligation states that private intention, hidden agency, raw byte identity, and host process trace are not replacement rows for Q. □

Theorem 651.6 (InscriptionAcceptanceBudget route stability)**.** *Accepted route reads of S, M, R, K, U, B, Q are stable under the componentwise H transports and the displayed C routes. Thus any downstream comparison that preserves the H* hsame *rows and follows the C* Cont *routes reads the same accepted source, name, route witness, check row, consumer row, budget row, and residue row.*

Proof. Start from the carrier of Definition 651.1. The rows H and C are part of the same finite packet as S, M, R, K, U, B, Q, so public transport and route reading are not external operations.

Apply Definition 651.2. It transports public comparisons of S, M, R, K componentwise by H and reads them through the displayed routes in C.

Use Definition 651.3 for U, B, Q. Downstream reads of the consumer budget and residue pass through the same H, C, P, N surface.

The route-stability obligation in Definition 651.4 then covers the whole accepted tuple S, M, R, K, U, B, Q over the same package and local naming rows. □

Theorem 651.7 (InscriptionAcceptanceBudget budget-spend exactness)**.** *Every downstream read of an accepted* InscriptionAcceptanceBudget$^{\uparrow}$ *carrier spends only the finite consumer-budget row B through the intended consumer row U and the local package and naming rows P, N. The packet exports no unbudgeted consumer right, no hidden route replay, and no consumer authority outside the displayed* Pkg *and* NameCert *surface.*

Proof. Read the carrier in Definition 651.1. The intended consumer row is U, the finite consumer-budget row is B, and the local package and naming rows are P, N.

Apply Definition 651.3. It states that every downstream use is bounded by U and B, and that a theorem site, citation row, audit surface, or certificate consumer reads the accepted material only through U, H, C, P, N.

The check-row exactness theorem, Definition 651.2, keeps admitted route use tied to the displayed row K rather than to a hidden replay route. The NameCert obligations of Definition 651.4 then identify consumer-budget exhaustion as a local finite obligation.

Therefore downstream consumption spends exactly the displayed budget row B through U and the local P, N rows, with no additional consumer authority outside the packet. □

Theorem 651.8 (InscriptionAcceptanceBudget mature consumer completeness)**.** *For every accepted* InscriptionAcceptanceBudget$^{\uparrow}$ *carrier* $A = (S, M, R, K, U, B, Q, H, C, P, N)$*, suppose the packet satisfies check-row exactness, consumer boundary, local* NameCert *obligations,* GAP *residue exposure, route stability, and budget-spend exactness. Then* A *exposes the complete mature public-object surface for the inscription acceptance budget: source* S*, named record* M*, route witness* R*, accepted check* K*, intended consumer* U*, finite budget* B*, residue* Q*, transport* H*, continuation* C*, provenance* P*, and local* NameCert *row* N *are all available as displayed finite rows. No private intention, hidden agency, unchecked replay, unbudgeted consumer use, or host process trace is exported by this mature surface.*

Proof. Project the carrier of Definition 651.1. Its displayed inventory is exactly $S, M, R, K, U, B, Q, H, C, P, N$, so the public-object surface starts as a finite row assembly rather than as a host-level semantic predicate.

The accepted route and consumer boundary are already fixed. Apply Definition 651.2 for K over S, R, and M, and apply Definition 651.3 for the downstream consumer row U and the finite budget row B.

Next assemble the audit rows. Use Definition 651.4 for the local NameCert obligations, Definition 651.5 for the residue row Q, and Definition 651.6 for the componentwise H transports and C routes.

Finally apply Definition 651.7. It confines downstream consumption to B through U, P, N. The mature row adds no second Lean bridge and no host-process semantics; it only repacks the six finite rows into the complete public inscription-budget surface and keeps private intention, hidden agency, unchecked replay, unbudgeted use, and host trace outside the exported packet. □

Closure status for NameCert$_{\text{InscriptionAcceptanceBudget}^{\uparrow}}$**.**

Constructive story: An InscriptionAcceptanceBudget$^{\uparrow}$ packet is generated from prior source state, a named public record, a route witness, an accepted check row, an intended consumer, a finite consumer-budget row, a GAP residue row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.InscriptionAcceptanceBudgetUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{InscriptionAcceptanceBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 651.1, Definition 651.2, Definition 651.3, Definition 651.4, Definition 651.5, Definition 651.6, Definition 651.7, and Definition 651.8 bind the same finite source, route, check, consumer-budget, GAP-residue, transport, continuation, provenance, and NameCert packet as a mature public inscription surface.

Formal verification: FormalStatus(NameCert$_{\text{InscriptionAcceptanceBudget}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InscriptionAcceptanceBudgetUp.InscriptionAcceptanceBudgetTasteGate\_single\_carrier\_al`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridge checking, complete inscription-event ontology, private intention, hidden agency, unchecked route semantics, unbudgeted downstream consumption, or host process traces.

Upgrade path: Further verification work is Lean bridge checking only; no higher paper-side closure grade is asserted here.

Chapter 652

A Concrete Naming Certificate for $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$

The $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ interface packages the diagonal-limit witness surface used by the Real Completeness route. It starts from the finite dyadic tolerance ledger, selected stream windows, regular-rational readback rows, and RealUp seal rows already exposed by the Real and Completion pipeline, then records the witness packet that a downstream completion consumer may inspect. It is a finite NameCert packet; it is not a selected host limit, quotient stream equality, or ambient completeness principle.

This extractor is the finite witness-facing consumer of chapter 126, chapter 291, chapter 301, chapter 38, and chapter 386. Its substantive dependency is the route from selected finite StreamName windows through dyadic tolerance readback and regular-rational rows into the RealUp seal and completion-facing handoff; it does not introduce a second completion operator.

Definition 652.1 (Real completion witness-extractor carrier)**.** *A* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, W, M, Q, L, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the diagonal regular-Cauchy route;*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window family used to read the finite diagonal observations;*
3. *M is the modulus schedule that chooses the displayed diagonal windows and records the finite comparison budget for each requested precision;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the selected windows and dyadic tolerance rows;*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular-rational readback has passed through the displayed diagonal schedule;*
6. *S is the completion-facing seal or handoff row consumed by Real Completeness and Cauchy-limit consumers;*
7. *H records componentwise* hsame *transports for D, W, M, Q, L, S and the local structural rows;*
8. *C records the displayed* Cont *routes from tolerance requests through the selected windows, modulus schedule, regular-rational readback, RealUp seal, and completion-facing handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessExtractor}^{\uparrow}}$ *row.*

The carrier accepts exactly those packets whose completion-facing handoff is read through the same finite dyadic ledger, selected windows, modulus schedule, regular-rational witness, RealUp seal, transport rows, continuation rows, and local certificate.

Lean4 (definition) `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier`

Theorem 652.2 (Real completion witness-extractor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealCompletionWitness}}$ *surface consists of carrier admission for the packet $X = (D, W, M, Q, L, S, H, C, P, N)$, selected-window admission through D, W, M, regular-rational readback through Q, RealUp seal handoff through L, completion-facing handoff through S, componentwise transport through H, displayed route replay through C, package provenance through P, local naming through N, and non-escape for quotient streams, selected host limits, host real equality, choice data, and ambient completeness.*

Proof. Project the carrier. The displayed packet lists the dyadic tolerance ledger, selected window family, modulus schedule, regular-rational readback, RealUp seal, completion-facing handoff, transports, routes, provenance, and local certificate.

The selected-window obligation is read first. The route from D through W and M fixes the finite observations and the precision schedule before any regular-rational witness can be exported.

The readback and seal obligations then factor through Q, L, S in that order. The rows H and C preserve this order componentwise, while P and N record the same finite packet as the local NameCert surface.

No coordinate of the carrier names quotient stream equality, a selected host limit, host real equality, choice data, or ambient completeness. Those conclusions therefore cannot be read from the witness extractor. □

Theorem 652.3 (Real completion witness-extractor diagonal handoff)**.** *Every Real Completeness diagonal-limit consumer of an accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet receives exactly the finite handoff*

$$D, W, M, Q, L, S, H, C, P, N.$$

The handoff presents the dyadic tolerance ledger, selected stream windows, modulus schedule, regular-rational witness row, RealUp seal, completion-facing seal, displayed hsame *transports, displayed* Cont *routes, package provenance, and local naming row. It supplies no independent completion operator and no equality of quotient real names.*

Proof. Start with the selected-window side of the carrier. The diagonal route can inspect D, W, M only as the displayed finite schedule for the requested precision.

Next read the regular-rational witness Q. It is attached to the same selected windows and dyadic ledger, so the consumer cannot choose a second window family or a second modulus schedule.

Finally pass through the RealUp seal L and the completion-facing handoff S. The transport, continuation, provenance, and local naming rows keep the handoff inside the same finite packet, and the carrier has no quotient-real or ambient-completion coordinate. □

Lean4 ✓ `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier diagonal handoff`

Theorem 652.4 (Real completion witness-extractor non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet factors through* $D, W, M, Q, L, S, H, C, P, N$*. The packet exports no quotient stream equality, no selected host limit, no host real equality, no arbitrary choice of a diagonal subsequence, and no ambient completeness theorem.*

Proof. Use the NameCert obligations. They place every consumer-visible read on the same carrier coordinates: dyadic tolerance ledger, selected windows, modulus schedule, regular-rational witness, RealUp seal, completion-facing handoff, transport rows, route rows, provenance, and local name.

For a stream or regular-rational consumer, the route stops at W, M, Q. For a RealUp or completion-facing consumer, the route continues through L and S after the same window and tolerance rows have already been fixed.

The excluded conclusions require rows not present in Definition 652.1. Since H, C, P, N only transport, replay, package, and name the displayed coordinates, no consumer can escape the finite witness packet. □

Theorem 652.5 (Real completion witness-extractor coordinate induction)**.** *Let* Φ *be a certificate read on accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packets. If* Φ *is fixed by the dyadic ledger coordinate* D*, the selected-window coordinate* W*, the modulus coordinate* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback coordinate* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal coordinate* L*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff coordinate* S*, the componentwise* hsame *transport coordinate* H*, the displayed* Cont *route coordinate* C*, the* Pkg *provenance coordinate* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessExtractor}^{\uparrow}}$ *coordinate* N*, and if the transport row* H *carries* Φ *componentwise across those displayed coordinates, then every accepted extractor packet satisfies* Φ*.*

Proof. Project the accepted carrier

$$X = (D, W, M, Q, L, S, H, C, P, N)$$

from Definition 652.1. The carrier has no hidden completion coordinate: every accepted read begins with the dyadic ledger D, selected windows W, and modulus schedule M.

Next read the computational coordinates in order. The $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q is attached to D, W, M by Definition 652.7; the $\mathsf{Real}^{\uparrow}$ seal L and $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff S are then attached by Definition 652.8.

Now use the structural coordinates. The componentwise transport row H preserves Φ across the displayed dyadic, stream, modulus, readback, seal, and handoff rows, while the continuation row C replays those reads in the same order. The provenance row P and local naming row N are exhausted by Definition 652.10.

Finally apply the non-escape theorem. Since any accepted consumer factors through exactly $D, W, M, Q, L, S, H, C, P, N$, and since Φ is stable under each coordinate together with the transports recorded by H, induction over the displayed coordinate spine proves Φ for every accepted extractor packet. □

Theorem 652.6 (Real completion witness-extractor selected-window admission)**.** *For every accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet, the first consumer-visible admission route exposes* D, W, M *as the finite selected-window route. The dyadic tolerance ledger* D*, the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, and the modulus schedule* M *are fixed before any regular-rational readback* Q*, RealUp seal* L*, or completion-facing handoff* S *can be read.*

Proof. Project the carrier of Definition 652.1. The rows D, W, M are listed before the readback, seal, and handoff rows, and they form the finite schedule on which the later reads are indexed.

Apply the local NameCert obligations. They bind carrier admission, selected window exposure, and modulus exposure to the same displayed packet. Thus a consumer cannot read Q, L, S without first passing through the selected-window route determined by D, W, M. □

Lean4 ✓ `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier selected window admission`

Theorem 652.7 (Real completion witness-extractor regular readback route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q in an accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet factors through the same D, W, M, H, C, P, N route. The readback may inspect the regular-rational witness attached to the selected windows, but it cannot choose a second window family, a second modulus schedule, or a second provenance and local naming route.*

Proof. Start with the diagonal handoff theorem. It attaches Q to the selected window and modulus coordinates already fixed by D, W, M.

Next use componentwise transport, displayed continuation replay, package provenance, and the local naming row. These are the rows H, C, P, N, so every readback route for Q remains inside the same transported and named packet.

The carrier contains no second copy of W or M and no alternate provenance or naming coordinate. Therefore the $\mathsf{RegSeqRat}^{\uparrow}$ readback cannot select a different window family or modulus schedule. □

Lean4 ✓ `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier regular readback route`

Theorem 652.8 (Real completion witness-extractor seal handoff totality)**.** *Every RealUp read through L and every completion-facing read through S in an accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet passes through the same D, W, M, Q, H, C, P, N route before the seal or handoff is consumed. These reads export no quotient stream equality, host limit, choice datum, or ambient completeness theorem.*

Proof. Read the packet in carrier order. The selected tolerance, window, modulus, and regular-rational rows D, W, M, Q are fixed before the RealUp seal and the completion-facing handoff are available to a consumer.

Transport and replay then keep L and S attached to the same packet through H and C, while P and N record the package provenance and local name. Thus every seal or handoff read is routed through D, W, M, Q, H, C, P, N.

The forbidden outputs would require additional coordinates not present in the carrier. Since the packet has no quotient equality row, no host-limit row, no choice row, and no ambient-completeness row, the seal and handoff are total only for the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier seal handoff totality`

Theorem 652.9 (Real completion witness-extractor transport totality)**.** *For every accepted* RealCompletionWitness *carrier*

$$X = (D, W, M, Q, L, S, H, C, P, N),$$

the transport and route rows H, C are total for the selected-window, regular-readback, nonescape, and seal-handoff obligations. In particular, every read of D, W, M, Q, L, S that is accepted by the local certificate replays through H, C before it reaches package provenance P and local naming N.

Proof. Begin with Definition 652.6. It fixes the selected-window route D, W, M before any later witness coordinate is visible.

Apply Definition 652.7. The readback row Q remains attached to the same selected windows, modulus schedule, transport, route, provenance, and naming rows.

Now use Definition 652.8. The RealUp seal L and completion-facing handoff S are consumed only after the selected-window and readback rows have passed through H, C, P, N.

Finally, Definition 652.4 excludes quotient stream equality, selected host limits, host real equality, arbitrary diagonal choice, and ambient completeness. Thus the displayed H, C rows exhaust transport and route replay for the accepted carrier. □

Lean4 ✓ `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier transport totality`

Theorem 652.10 (Real completion witness-extractor provenance exhaustion)**.** *For every accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *carrier*

$$X = (D, W, M, Q, L, S, H, C, P, N),$$

the provenance and naming rows P, N account for all source chapters needed by the selected-window, regular-readback, nonescape, and seal-handoff route: $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Read the carrier definition. Its package row P explicitly names the dyadic tolerance ledger, selected StreamName windows, regular-rational readback, RealUp seal, CauchyLimitSeal handoff, history, transport, route, package, and naming dependencies.

Use Definition 652.6 and Definition 652.7. These theorems place D, W, M, Q under the same H, C, P, N packet.

Use Definition 652.8 together with Definition 652.4. The seal and handoff rows add no second provenance source and export none of the excluded completion or choice principles.

Therefore P, N exhaust the provenance and local naming obligations for the displayed witness-extractor carrier. □

Theorem 652.11 (Real completion witness-extractor obligation closure package)**.** *The accepted carrier*

$$X = (D, W, M, Q, L, S, H, C, P, N)$$

is an obligation-closed $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *package for the present chapter: the selected-window route is fixed by D, W, M, the regular-readback route is fixed by Q, the Real and completion handoff route is fixed by L, S, transport and route replay are exhausted by H, C, and provenance plus local naming are exhausted by P, N.*

Proof. Assemble the obligation rows from the preceding theorems. The selected-window theorem supplies the D, W, M admission route, the regular-readback theorem supplies the Q route, and the seal-handoff theorem supplies the L, S route.

Apply Definition 652.9. It accounts for componentwise transport and continuation replay across every displayed consumer read.

Apply Definition 652.10. It accounts for package provenance and local naming over the Real and Completion dependency surface.

The nonescape theorem then removes the remaining forbidden exits: quotient stream equality, selected host limits, host real equality, arbitrary diagonal choice, ambient completeness, and un displayed completion operators. Hence the displayed carrier closes the chapter's transport, route replay, provenance, and naming obligations. □

Theorem 652.12 (Real completion witness-extractor scoped route)**.** *Every accepted* RealCompletionWitnessExtract *packet*

$$X = (D, W, M, Q, L, S, H, C, P, N)$$

is scoped by the displayed finite witness route: the DyadicRatCore tolerance ledger D, StreamName windows W, RegSeqRat readback row Q, RealUp seal L, CauchyLimitSeal handoff S, componentwise hsame *transport row H, displayed* Cont *route row C, package row P, and local* NameCert *row N. The route exports no quotient stream equality, host real equality, selected host limit, arbitrary diagonal choice, or ambient completeness principle.*

Proof. First build the finite witness route. Apply Definition 652.6 to fix the dyadic tolerance ledger, selected StreamName windows, and modulus schedule. Apply Definition 652.7 to attach the RegSeqRat readback row to the same packet.

Next attach the completion-facing rows. Use Definition 652.8 for the RealUp seal and CauchyLimitSeal handoff, then use Definition 652.9 to route all displayed reads through componentwise hsame transport and displayed Cont continuation.

Finally, apply Definition 652.10 to account for Pkg provenance and local NameCert naming. The nonescape theorem, Definition 652.4, excludes quotient stream equality, host real equality, selected host limits, arbitrary diagonal choice, and ambient completeness. Therefore the accepted packet is scoped exactly by the displayed finite witness route. □

Theorem 652.13 (Real completion witness-extractor kernel support scope)**.** *The scoped route of Definition 652.12 depends only on the named sibling chapters* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*, and* CauchyLimitSeal$^{\uparrow}$*, together with the BEDC kernel rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. No further completion carrier, quotient principle, host equality row, selected-limit operator, diagonal-choice operator, or ambient-completeness row is part of the support.*

Proof. Project the carrier

$$X = (D, W, M, Q, L, S, H, C, P, N).$$

The first five witness rows are the displayed sibling-chapter rows, while H, C, P, N are the displayed kernel transport, route, package, and naming rows attached to the same packet.

Apply Definition 652.12. It already factors every accepted read through the selected-window route, regular-readback route, seal-handoff route, componentwise transport, displayed continuation, provenance, and local naming rows.

The support list is exhausted by Definition 652.10. The excluded carriers and principles are absent by Definition 652.4, so the scoped route has exactly the stated kernel support. □

Theorem 652.14 (Real completion witness-extractor standard bridge surface)**.** *For every accepted* RealCompletionWitnessExtractor$^{\uparrow}$ *packet*

$$X = (D, W, M, Q, L, S, H, C, P, N),$$

the finite rows $D, W, M, Q, L, S, H, C, P, N$ form the paper bridge surface from the BEDC witness extractor to the standard diagonal-witness reading. Every standard consumer read factors through the checked diagonal handoff, the selected-window admission route, the regular-readback route, the seal-handoff totality row, and the transport totality rows. No quotient stream equality, host limit, choice datum, or ambient completeness row is exported by this bridge surface.

Proof. Start with Definition 652.3. It supplies the checked handoff from the displayed BEDC packet to the diagonal-witness reading while keeping the same carrier coordinates $D, W, M, Q, L, S, H, C, P, N$.

Next compose the route witnesses. Apply Definition 652.6 for D, W, M, then apply Definition 652.7 for Q. These two rows place each standard read over the selected dyadic windows and the displayed regular readback coordinate.

Now attach the completion-facing handoff. Apply Definition 652.8 for L, S, and apply Definition 652.9 for H, C. Thus the consumer route is carried only by the displayed seal, handoff, transport, and continuation rows.

Finally, the nonescape and support theorems exclude the remaining exits. Quotient stream equality, host limits, choice data, and ambient completeness do not occur among the exported rows, so the displayed finite packet is exactly the paper bridge surface for the standard diagonal-witness reading. □

Theorem 652.15 (Real completion witness-extractor finite-prefix stability handoff)**.** *Every accepted* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *packet whose selected-window route is supplied by an accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *packet must read the finite prefix budget,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and extractor rows through one ordered finite route before any Completion-facing handoff is visible. Thus the extractor can consume finite-prefix stability only as the displayed handoff*

$$B \to W \to R \to D \to E \to X$$

where $X = (D', W', M, Q, L, S, H, C, P, N)$ is the accepted extractor carrier, with D', W', M reading the selected extractor window, Q reading the regular-rational extractor row, and L, S, H, C, P, N reading only the displayed seal, handoff, transport, continuation, provenance, and naming rows.

Proof. First read the supplying stability packet. Apply Definition 1226.7 to identify the finite handoff from the displayed prefix observation to the terminal $\mathsf{Real}^{\uparrow}$ seal, and apply Definition 1226.8 to make that seal the unique consumer-facing end of the route.

Next pass the displayed route into the extractor. Apply Definition 652.6 to attach the extractor's selected window rows D', W', M to the same finite route. Apply Definition 652.7 to read Q only after that selected window has been admitted.

Finally attach the Completion-facing rows. Use Definition 652.8 to read L, S as the visible Real and Completion handoff rows of the accepted extractor packet. Since the supplying stability packet has already fixed the order $B \to W \to R \to D \to E$, and the extractor theorems expose only $D', W', M, Q, L, S, H, C, P, N$, no Completion-facing handoff is visible before the finite-prefix stability route and the extractor rows have both been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, selected StreamNameUp windows, a modulus schedule, a RegSeqRat readback row, a RealUp seal row, a completion-facing handoff row, displayed hsame transports, Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWitnessExtractor}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 652.1, Definition 652.14, and Definition 652.15 bind the finite diagonal witness-extractor packet, its standard bridge surface, and its finite-prefix stability handoff for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RealCompletionWitnessExtractor}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.RealCompletionWitnessExtractorUp.RealCompletionWitnessExtractorCarrier\_diagonal\_hando`
Bridge status: paperBridge
Origin: human
Not claimed: This chapter does not close quotient stream equality, selected host limits, host real equality, arbitrary diagonal choice, ambient completeness, or bridge checking.
Upgrade path: The theory axis is mature; the remaining work is verification-axis bridge checking only.

Chapter 653

A Concrete Naming Certificate for $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$

The $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$ packet names the finite symmetry surface between an alternating regular-Cauchy interleaving and the same interleaving read with the two source sides exchanged. It is a Real-and-Completion certificate over displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows; it is not an equality of quotient reals, a host-level limit comparison, or an appeal to ambient completeness.

The packet consumes the schedule rows of chapter 126, the regular rational source rows of chapter 301, the selector parity surface of chapter 414, and the seal boundary of chapter 38. These are substantive source rows for $A, B, S_A, S_B, \sigma, \sigma^{\mathrm{op}}, E, E'$, not decorative citations.

Definition 653.1 (Regular Cauchy swap bisimulation carrier)**.** *A* $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (A, B, S_A, S_B, \sigma, \sigma^{\mathrm{op}}, M, E, E', H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source names whose finite rational observations may be interleaved;*
2. *S_A and S_B are the two* $\mathsf{StreamName}^{\uparrow}$ *schedules that expose the selected finite windows for A and B;*
3. *σ is the alternating selector row that reads A on its even side and B on its odd side;*
4. *σ^{op} is the opposite selector row that reads the same schedules after exchanging the two source sides;*
5. *M is the combined modulus ledger shared by the two selector presentations;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through the A, B, σ interleaving route;*
7. *E' is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through the $B, A, \sigma^{\mathrm{op}}$ interleaving route;*
8. *H stores componentwise* hsame *transports for $A, B, S_A, S_B, \sigma, \sigma^{\mathrm{op}}, M, E, E'$ and for the local packet shape;*
9. *C stores the displayed* Cont *routes from each selector read through the selected schedule, the matching component of M, the corresponding* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *row, and the real-facing seal boundary;*

10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySwapBisimulation}^{\uparrow}}$ *row.*

The carrier contains no quotient-real equality row, no selected host limit, no ambient completeness coordinate, and no third schedule.

Lean4 (definition) `BEDC.Derived.RegularCauchySwapBisimulationUp.RegularCauchySwapBisimulationUp`

Theorem 653.2 (Regular Cauchy swap bisimulation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchySwapBis}}$ *surface for Definition 653.1 consists of six finite obligation rows:*

1. *carrier admission for the packet $W = (A, B, S_A, S_B, \sigma, \sigma^{\mathrm{op}}, M, E, E', H, C, P, N)$;*
2. *forward interleaving admission through $A, B, S_A, S_B, \sigma, M, E$;*
3. *opposite interleaving admission through $B, A, S_B, S_A, \sigma^{\mathrm{op}}, M, E'$;*
4. *selector-symmetry exactness stating that the opposite selector exchanges the same two schedule windows and the same two ledger components;*
5. *componentwise stability under the displayed* hsame *transports and replay through the displayed* Cont *routes;*
6. *real-seal non-escape stating that both real-facing rows are consumed only after the selected source name, selected schedule, shared ledger, interleaving row, package provenance, and local naming row have been displayed.*

Proof. Carrier admission is projection from the displayed tuple in Definition 653.1. The packet lists exactly the two source names, two schedules, two selector rows, common ledger, two seal rows, transports, continuations, provenance, and local name.

The forward admission row is the ordinary $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ route through $A, B, S_A, S_B, \sigma, M, E$. The opposite admission row uses the same data after exchanging the source and schedule coordinates and reading σ^{op}.

Selector-symmetry exactness and stability are then local rows of the same packet. The hsame transports preserve each displayed coordinate componentwise, and the Cont routes replay only the chosen schedule and matching ledger component.

Finally, P and N repack those rows as the local certificate surface. Since the carrier has no coordinate for quotient real equality, selected host limits, ambient completeness, or a third schedule, the six rows exhaust the available certificate. □

Lean4 ✓ `BEDC.Derived.RegularCauchySwapBisimulationUp.RegularCauchySwapBisimulationTasteGate`
`single carrier alignment`

Theorem 653.3 (Regular Cauchy swap bisimulation selector symmetry)**.** *For an accepted* RegularCauchySwapBisim *carrier W, every even read of σ through A, S_A and the first component of M is matched by an odd read of σ^{op} through the same A, S_A route and the same ledger component. Every odd read of σ through B, S_B and the second component of M is matched by an even read of σ^{op} through the same B, S_B route and the same ledger component. The matching is preserved by the displayed* hsame *transports and* Cont *routes.*

Proof. Unpack the carrier. The only source rows are A and B, the only schedule rows are S_A and S_B, and the only selector rows are σ and σ^{op}.

Apply the selector parity exactness of $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ to the forward route. It identifies the even reads with A, S_A and the first component of M, and the odd reads with B, S_B and the second component.

Apply the same exactness to the opposite route. Because σ^{op} is stored as the selector after exchanging the two source sides, its odd reads recover the forward even route and its even reads recover the forward odd route.

The stored H and C rows preserve this pairing componentwise. No third schedule or unlisted ledger component is present in the carrier, so the selector symmetry is confined to the displayed two-sided packet. □

Theorem 653.4 (Regular Cauchy swap bisimulation Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of the seal rows* E *or* E' *in an accepted* $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$ *carrier factors through one of the two displayed interleaving routes, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *source name, the selected* $\mathsf{StreamName}^{\uparrow}$ *schedule, the shared modulus ledger* M*, the displayed* hsame *transport, the displayed* Cont *route, the package provenance* P*, and the local naming row* N*. The packet exports no quotient real equality, selected host limit, ambient completeness proof, or third schedule.*

Proof. Start from the two seal coordinates. The row E is downstream of the forward interleaving route, while E' is downstream of the opposite interleaving route.

Use selector symmetry to identify which source side and schedule are read before each seal. The two routes differ only by the stored exchange of source sides and selector parity; both consume the same shared ledger M through a displayed component.

The rows H and C attach the required transport and replay data to those finite reads. The provenance row P and local name N then close the packet without adding any real object beyond the two seal boundaries.

Since the carrier definition has no field for quotient real equality, selected limits, ambient completeness, or an unlisted schedule, a real-facing consumer cannot leave the displayed packet. □

Theorem 653.5 (Regular Cauchy swap bisimulation public handoff)**.** *The public handoff of an accepted* $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$ *carrier is exactly the finite packet*

$$A, B, S_A, S_B, \sigma, \sigma^{\mathrm{op}}, M, E, E', H, C, P, N.$$

It presents two selector-swapped $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *routes with the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source names,* $\mathsf{StreamName}^{\uparrow}$ *schedules, combined modulus ledger, displayed transports, displayed continuations, package provenance, and local* $\mathsf{NameCert}$ *row. A downstream Real-and-Completion consumer receives this finite boundary and receives no quotient real equality, selected host limit, ambient completeness principle, or bridge-checked schema.*

Proof. Carrier admission gives the listed tuple as the whole accepted object. The NameCert obligations identify the forward and opposite interleaving admissions, selector-symmetry row, stability rows, and seal-boundary row as local obligations of that same tuple.

The selector-symmetry theorem binds the two interleaving presentations to the same pair of source schedules and the same ledger components. Thus the public route does not introduce a fresh interleaving carrier; it exposes the two stored presentations of one finite packet.

The Real seal non-escape theorem confines every consumer of E or E' to those displayed rows. Repacking the carrier, obligations, selector symmetry, and non-escape boundary yields the announced public handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySwapBisimulation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySwapBisimulation}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source names, two $\mathsf{StreamName}^{\uparrow}$ schedules, an alternating selector, its opposite selector, a shared modulus ledger, two $\mathsf{Real}^{\uparrow}$ seal rows, displayed hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySwapBisimulation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 653.1, Definition 653.2, Definition 653.3, Definition 653.4, and Definition 653.5 bind the finite selector-swapped interleaving packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySwapBisimulation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient real equality, selected host limits, ambient completeness, bridge checking, or any route through a third schedule.

Upgrade path: Obligation closure requires checked carrier admission, forward and opposite interleaving admission, selector-symmetry exactness, componentwise stability, and RealUp seal non-escape for the displayed packet.

Chapter 654

A Concrete Naming Certificate for $\mathsf{AxiomQueryLedger}^{\uparrow}$

The $\mathsf{AxiomQueryLedger}^{\uparrow}$ packet names the finite audit surface consumed by strict proof-ancestry checks. It records an accepted declaration, the printed axiom-query report for that declaration, the forbidden-dependency comparison row, the refusal verdict, replay provenance, and the local NameCert row in one BHist certificate. It refines the dependency-report face of chapter 526 by isolating the declaration row, query output row, forbidden-dependency row, and verdict row as a reusable finite ledger. It is the query-row sibling of the dependency-map packet in chapter 862, the audit-gate boundary in chapter 526, and the kernel-acceptance audit witness in chapter 547: those chapters consume or police the same displayed query, verdict, gate, and acceptance rows. The packet turns the kernel-acceptance query `#print axioms` into a concrete ledger entry rather than an informal host-side statement about dependencies.

Definition 654.1 (Axiom-query ledger packet)**.** *An* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$L = (D, Q, F, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted declaration row, naming the checked BEDC theorem, lemma, definition, structure, class, or inductive declaration whose ancestry is queried;*
2. *Q is the axiom-query output row returned by the displayed* `#print axioms` *read for D;*
3. *F is the forbidden-dependency row for* `Classical.choice`*,* `Quot.sound`*,* `propext`*, and any project-level host dependency excluded by the strict audit;*
4. *R is the verdict row recording whether the query output is admitted as clean or refused because it meets a forbidden dependency row;*
5. *H records componentwise* hsame *transport for declarations, query outputs, forbidden rows, verdicts, replay routes, provenance, and local names;*
6. *C records the displayed* Cont *routes by which a consumer replays the declaration read, the query invocation, the comparison with F, and the verdict exposure;*
7. *P is the* Pkg *provenance over kernel acceptance, axiom-query replay, forbidden-dependency comparison,* hsame*,* Cont*,* Pkg*, and* NameCert *source packets;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$ *row.*

The packet has no coordinate for accepting a forbidden dependency as BEDC theorem content, replacing D by a neighboring declaration, treating source-token absence as ancestry purity, or hiding a query report outside the displayed ledger rows.

Theorem 654.2 (Axiom-query ledger obligations)**.** *For every* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet*

$$L = (D, Q, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$ *surface consists exactly of accepted-declaration exposure, axiom-query output exposure, forbidden-dependency row exposure, verdict exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Its non-exported surface excludes hidden host axioms, declaration substitution, source-token-only acceptance, and unrecorded query reruns.*

Proof. Carrier exposure is projection from Definition 654.1. The finite tuple lists exactly the rows D, Q, F, R, H, C, P, N.

The audit obligations are read through Q, F, R. The query row supplies the printed ancestry report for the accepted declaration, the forbidden row supplies the excluded dependency names, and the verdict row records the finite comparison.

Transport, replay, provenance, and naming are confined to H, C, P, N. These rows account for every public route by which a consumer can move, rerun, package, or name the same ledger entry.

Since no remaining coordinate stores a hidden host axiom, an alternate declaration, source-token-only proof, or unrecorded query rerun, the displayed rows exhaust the local $\mathsf{NameCert}$ obligations. □

Theorem 654.3 (Axiom-query ledger query soundness)**.** *Let*

$$L = (D, Q, F, R, H, C, P, N)$$

be an accepted $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet. Any downstream claim that the declaration row D passed strict axiom-query audit must factor through the displayed query row Q, the forbidden-dependency row F, the verdict row R, and the replay route in C. A clean verdict is therefore a statement about the recorded query output for D, not about an unstated ambient environment scan.*

Proof. Start with the accepted packet and project its declaration row D. The only query output attached to this declaration inside the carrier is the row Q.

The strict audit comparison is stored by the pair F, R. The row F names the excluded dependencies, and R records the comparison result for the query output already paired with D.

Replay is visible only through C, with componentwise transport and provenance supplied by H and P. Thus a downstream clean-audit claim must pass through the recorded query, comparison, verdict, and replay rows rather than through an unstated environment-level assertion. □

Lean4 ✓ `BEDC.Derived.AxiomQueryLedgerUp.AxiomQueryLedgerCarrier query soundness`

Theorem 654.4 (Axiom-query ledger refusal boundary)**.** *If the query row Q of an* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet meets the forbidden-dependency row F, the only exported classification of that meeting is the refusal content in R. The packet does not export an alternate proof, a renamed declaration, a weaker declaration row, or a host-side permission to treat the forbidden dependency as BEDC content.*

Proof. Inspect the packet $L = (D, Q, F, R, H, C, P, N)$. The declaration and query rows expose the checked declaration and its printed ancestry report.

When Q meets F, the carrier records that meeting in the verdict row R. The comparison has no second verdict coordinate and no alternate declaration coordinate.

The remaining rows preserve and replay the same packet. The transport row H, continuation row C, provenance row P, and naming row N cannot introduce a replacement proof, weaker declaration, or permission row. Therefore the forbidden meeting is exported only as refusal content. □

Theorem 654.5 (Axiom-query ledger non-escape)**.** *Every accepted* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet exports axiom-query information only through the finite rows*

$$D, Q, F, R, H, C, P, N.$$

It exports no host-level proof oracle, no hidden dependency list, no quotient or classical principle, no declaration substitution rule, and no downstream route that bypasses the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Project the accepted packet as $L = (D, Q, F, R, H, C, P, N)$. The definition lists every row exported by the finite ledger.

The query side is exhausted by D, Q, F, R. These rows record the accepted declaration, the printed axiom-query output, the forbidden dependency list, and the verdict.

The consumer side is exhausted by H, C, P, N. These rows record transport, replay, provenance, and the local name for the same ledger entry.

No coordinate remains for a proof oracle, hidden dependency list, quotient or classical principle, declaration substitution rule, or bypass route. The packet therefore has the stated non-escape boundary. □

Theorem 654.6 (Axiom-query ledger carrier exposure)**.** *Every admitted public read of an accepted* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet first projects one of the displayed rows*

$$D, Q, F, R, H, C, P, N.$$

Thus a consumer can read the accepted declaration, query output, forbidden-dependency row, verdict row, componentwise transport, replay route, package provenance, or local naming row, but no public read enters the ledger without passing through one of these finite BHist *coordinates.*

Proof. Project the packet as $L = (D, Q, F, R, H, C, P, N)$. The carrier definition lists these rows as the whole exported BHist surface.

An admitted public read is mediated by the displayed Cont row C. Each continuation route in C targets one of the declaration, query, forbidden-row, verdict, transport, provenance, or naming coordinates already present in the packet.

The hsame, Pkg, and NameCert rows preserve and name those same coordinates. Since the carrier has no additional public field, every admitted read first projects one of the displayed rows. □

Theorem 654.7 (Axiom-query ledger verdict determinacy)**.** *For an accepted* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet* $L = (D, Q, F, R, H, C, P, N)$*, the verdict row* R *is determined by the paired query output row* Q *and forbidden-dependency row* F*. Transport of* Q *and* F *by the componentwise* hsame *row transports the same verdict row and cannot create a second clean or refusal classification.*

Proof. Read the query side of the packet. The row Q is the displayed axiom-query output for the accepted declaration row D, and the row F is the displayed forbidden-dependency comparison surface.

The verdict coordinate R records the finite comparison between exactly those two rows. There is no alternate verdict coordinate and no route that compares Q against a hidden forbidden list or compares F against a different query report.

Now transport the packet componentwise through H. The transported query and forbidden rows are the transported copies of Q and F, so the transported verdict is the transported copy of their recorded comparison. The Cont, Pkg, and NameCert rows replay and name that same comparison rather than adding another classification. □

Theorem 654.8 (Axiom-query ledger replay provenance)**.** *Replay of an accepted* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *packet cannot bypass the displayed* Cont*,* Pkg*, and* NameCert *rows. Any replay that exposes* D, Q, F, R, H, C, P, N *must use the continuation row* C*, carry the package provenance row* P*, and return to the local naming row* N *for the same ledger entry.*

Proof. Start from the finite ledger projection. The rows D, Q, F, R, H give the declaration, query output, forbidden comparison, verdict, and transport data that a replay may expose.

The only replay routes stored by the carrier are the displayed continuations in C. A consumer therefore reaches the declaration read, query invocation, forbidden-row comparison, and verdict exposure through C, not through an ambient host rerun.

The provenance row P records the package ancestry for that replay, and the local row N names the same $\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$ entry. Repacking a replay without C, P, and N would require a carrier coordinate not present in L, so bypass is excluded. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AxiomQueryLedger}^{\uparrow}$ packet is generated from an accepted declaration row, a displayed `#print axioms` query output, a forbidden-dependency row, a verdict row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public query-ledger export is the same visible finite audit surface already bound by Definition 654.1, Definition 654.2, Definition 654.3, Definition 654.4, Definition 654.5, Definition 654.6, Definition 654.7, and Definition 654.8: accepted declaration row D, query output row Q, forbidden-dependency row F, verdict row R, componentwise hsame row H, displayed Cont replay row C, Pkg provenance row P, and local NameCert row N.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxiomQueryLedger}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AxiomQueryLedgerUp.AxiomQueryLedgerCarrier\_query\_soundness`

Bridge status: none

Origin: human

Not claimed: This chapter does not prove global host-kernel purity, completeness of all environment dependency queries, Lean-side formal verification, bridge checking, or acceptance of any dependency outside the displayed finite packet.

Upgrade path: The chapter is mature on the paper closure axis; bridge checking remains a separate formal-axis obligation for the public query-ledger rows.

Chapter 655

A Concrete Naming Certificate for $\mathsf{MultiHistSuperposition}^{\uparrow}$

The $\mathsf{MultiHistSuperposition}^{\uparrow}$ interface records the finite packet in which the inter-Hist locality chain reads quantum-state style content through displayed $\mathsf{E1}$ and $\mathsf{E0}$ refinement rows rather than through metric geometry. It consumes the already named multi-Hist configuration, cross-Hist causal witnesses, and maximum-rate bound rows, then adds a phase-coherence ledger and local NameCert surface over the same finite histories.

The source configuration row is read from chapter 474, the causal witness family from chapter 478, and the inherited unary rate bound from chapter 594. These are respectively the carrier, witness-family, and rate-bound surfaces consumed by the superposition packet. The phase-coherence row is downstream of those finite sibling rows and does not replace them by a Hilbert-space carrier or probability carrier.

Definition 655.1 (MultiHistSuperposition carrier)**.** *A* $\mathsf{MultiHistSuperposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, E, Z, X, R, \Phi, H, Q, C, P, N)$$

with the following visible coordinates:

1. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite family of observer-history rows, local ledgers, no-global-sync boundary, componentwise* hsame *transports, visible* Cont *routes,* Pkg *provenance, and local naming row;*

2. *E is the finite* $\mathsf{E1}$ *refinement family over the displayed histories of M, recording observation-sensitive rows only where the packet has a visible continuation route;*

3. *Z is the finite* $\mathsf{E0}$ *refinement family over the same histories, recording name-sensitive rows only where the local naming ledger displays them;*

4. *X is the finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness rows over selected pairs of histories from M and selected refinement slots from E and Z;*

5. *R is the accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *bound row for the displayed causal witness family X over the same configuration row M;*

6. *Φ is the phase-coherence ledger, a finite row comparing the selected* $\mathsf{E1}$ *and* $\mathsf{E0}$ *refinements along the displayed cross-Hist witnesses without quotienting histories or selecting a global phase;*

7. *H records componentwise* hsame *transport for M, E, Z, X, R, Φ, the probe-stability row, continuation routes, package row, and local naming row;*
8. *Q records* psame *stability for the displayed probe reads used by the refinement rows and their phase-coherence ledger;*
9. *C records the visible* Cont *routes from the configuration, refinement, causal, rate, and phase rows into the local certificate;*
10. *P is the* Pkg *provenance over* $\mathsf{MultiHistSuperposition}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, E1, E0, hsame, psame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MultiHistSuperposition}^{\uparrow}}$ *row.*

The carrier exports only finite observer-history rows, displayed refinement ledgers, selected cross-Hist causal witnesses, the inherited maximum-rate bound, phase-coherence rows, transport rows, continuation routes, provenance, and local naming data. It contains no Hilbert space, amplitude field, global phase, completed state vector, quotient of histories, or ambient probabilistic measure.

Theorem 655.2 (MultiHistSuperposition NameCert obligations)**.** *For the carrier of Definition 655.1, the local* $\mathsf{NameCert}_{\mathsf{MultiHistSuperposition}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (M, E, Z, X, R, \Phi, H, Q, C$ refinement locality for the displayed* E1 *and* E0 *rows over M, cross-Hist causal inheritance through X, maximum-rate inheritance through R, phase-coherence exactness through Φ, componentwise* hsame *and displayed* psame *stability through H, Q, visible* Cont *locality through C,* Pkg *provenance through P, local naming through N, and non-escape from global phases, completed state vectors, Hilbert spaces, quotient histories, and ambient probability measures.*

Proof. Carrier admission is projection from Definition 655.1. The displayed packet supplies every row named by the local certificate.

Refinement locality reads M before E and Z. Each refinement row is attached to a displayed history and a visible continuation or naming route, so the refinement cannot introduce an unlisted observer history.

The causal and bound obligations are inherited in order. The witness family X consumes selected histories and refinement slots from M, E, Z, while R supplies only the finite maximum-rate row for that same witness family.

Phase-coherence exactness is the ledger condition on Φ. It compares selected refinement rows along displayed witnesses, not representatives in a quotient of histories or amplitudes in a host state space.

Finally route all reads through H, Q, C, P, N. Since the carrier has no coordinate for global phase, Hilbert space, completed vector state, quotient history, or ambient measure, the non-escape clause is the displayed inventory boundary. □

Theorem 655.3 (MultiHistSuperposition phase coherence)**.** *For an accepted* $\mathsf{MultiHistSuperposition}^{\uparrow}$ *carrier*

$$S = (M, E, Z, X, R, \Phi, H, Q, C, P, N),$$

the phase-coherence ledger Φ is read only as a finite comparison of displayed E1 *and* E0 *refinement rows along the selected cross-Hist witness family X. Componentwise* hsame *transport in H, displayed* psame *stability in Q, and visible routes in C preserve the same finite phase ledger; they do not identify histories by quotient, choose a global phase, or complete the packet into a host quantum state.*

Proof. Project the accepted carrier and read E, Z, X, Φ. The refinement families and witness family are finite rows over the displayed multi-Hist configuration.

Transport through H preserves the histories, refinement rows, witness family, inherited rate row, and phase ledger componentwise. The probe row Q preserves the probe reads used by the same refinement comparisons.

The continuation routes in C keep every phase read on the displayed path from M to E, Z, X, R, Φ. Therefore the transported phase-coherence row remains the same finite BEDC ledger and cannot become a quotient, selected global phase, or completed state vector. □

Theorem 655.4 (MultiHistSuperposition non-escape)**.** *Every consumer of an accepted* $\mathsf{MultiHistSuperposition}^{\uparrow}$ *carrier factors through the finite rows*

$$M, E, Z, X, R, \Phi, H, Q, C, P, N.$$

In particular, the consumer may read the multi-Hist configuration row, E1 *refinement rows,* E0 *refinement rows, selected cross-Hist causal witnesses, maximum-rate bound row, phase-coherence ledger, componentwise* hsame *transports, displayed* psame *stability, visible* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{MultiHistSuperposition}^{\uparrow}}$ *row, and nothing outside those rows.*

Proof. Start with the carrier inventory. The listed coordinates are the only public rows of Definition 655.1.

Use the inherited non-escape boundaries of $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and $\mathsf{MaxCausalRate}^{\uparrow}$. They exclude global synchronization, unlisted causal witnesses, metric geometry, and ambient real-valued rate data from the upstream rows.

The refinement and phase rows add only E, Z, Φ. By Definition 655.3, these rows are finite ledgers transported through H, Q, C and do not export Hilbert spaces, amplitude fields, global phases, completed state vectors, quotient histories, or ambient probability measures.

Thus every consumer route is exhausted by $M, E, Z, X, R, \Phi, H, Q, C, P, N$. □

Theorem 655.5 (MultiHistSuperposition obligation surface)**.** *The* $\mathsf{MultiHistSuperposition}^{\uparrow}$ *obligation surface is the finite package assembled from Definition 655.1, Definition 655.2, Definition 655.3, and Definition 655.4. It is the concrete realization of the* E1/E0 *refinement step of the inter-Hist locality chain: a local* NameCert *over finite* BHist*,* hsame*,* psame*,* Cont*,* Pkg*, and refinement rows, not a host quantum-state construction.*

Lean4 ✓ `BEDC.Derived.MultiHistSuperpositionUp.MultiHistSuperpositionTasteGate single carrier alignment`

Proof. The carrier definition supplies the finite packet rows and their dependency order. The NameCert obligation theorem binds those rows to carrier admission, refinement locality, causal and rate inheritance, phase exactness, transport stability, provenance, naming, and non-escape.

The phase-coherence theorem isolates the only quantum-state style row as the finite ledger Φ over displayed E1 and E0 refinements. The non-escape theorem then exhausts every consumer route by the same packet.

Repack these rows under the local $\mathsf{NameCert}_{\mathsf{MultiHistSuperposition}^{\uparrow}}$ entry. Since each component is one of $M, E, Z, X, R, \Phi, H, Q, C, P, N$, the obligation surface stays within the finite BEDC kernel rows named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{MultiHistSuperposition}^{\uparrow}}$**.**

Constructive story: A MultiHistSuperposition$^{\uparrow}$ packet is generated from a $\mathrm{MultiHistConfig}^{\uparrow}$ row, finite E1 and E0 refinement rows, selected CrossHistCausal$^{\uparrow}$ witnesses, an inherited MaxCausalRate$^{\uparrow}$ bound row, a phase-coherence ledger, componentwise hsame and displayed psame stability, visible Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MultiHistSuperposition}^{\uparrow}}$, seed)

Scope closed: The seed scope records the finite multi-Hist superposition packet over displayed BHist, refinement, causal, rate, phase-coherence, hsame, psame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MultiHistSuperposition}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Hilbert spaces, amplitudes, global phases, probability measures, completed quantum states, quotient histories, Lorentz geometry, curved spacetime, or Lean-side verification.

Upgrade path: Obligation closure requires carrier admission, refinement locality, causal and rate inheritance, phase-coherence exactness, transport stability, provenance, and non-escape obligations for the displayed finite packet.

Chapter 656

A Concrete Naming Certificate for InterHistInvariant$^{\uparrow}$

The InterHistInvariant$^{\uparrow}$ interface packages a finite two-history invariant readback for the Inter-Hist locality chain. It consumes the multi-history packet rows already exposed by FiniteMultiHistPacket$^{\uparrow}$, the causal-witness rows supplied by CrossHistCausal$^{\uparrow}$ when present, and the unary maximum-rate boundary consumed by MaxCausalRate$^{\uparrow}$, while keeping the whole read inside BHist, hsame, Cont, Pkg, and NameCert provenance. The causal row is consumed through chapter 478, and the downstream unary-rate boundary is the consumer surface of chapter 594. The packet is not a global frame, a synchronized clock, a metric geometry, a host real equality, or a quotient of histories.

Falsifiable prediction: Every accepted InterHistInvariant$^{\uparrow}$ consumer reaches the invariant through the displayed finitemulthistpacket, crosshistcausal, maxcausalrate, and crosshistcausalrate sibling rows; a consumer route that bypasses those sibling rows refutes the packet.

Independence witness: Nearest siblings: finitemulthistpacket, crosshistcausal, maxcausalrate, crosshistcausalrate. None is bijective to this carrier because InterHistInvariant$^{\uparrow}$ simultaneously stores two history slots, one relation or causal row, one unary invariant row, one observer-symmetry row, transport, route, provenance, and local naming data, while each sibling omits at least one of those rows or records a rate-facing object instead of the invariant readback.

Definition 656.1 (InterHistInvariant carrier)**.** *An* InterHistInvariant$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, R, V, S, H, C, P, N)$$

with the following visible rows:

1. *A and B are the two history rows being compared;*
2. *R is either the displayed relation row supplied by* FiniteMultiHistPacket$^{\uparrow}$ *or the causal-witness row supplied by* CrossHistCausal$^{\uparrow}$*;*
3. *V is the unary invariant readback row obtained from the transported pair and the relation row;*
4. *S is the observer-symmetry row, recording that swapping the two history rows transports to the same unary readback surface;*
5. *H stores componentwise* hsame *transport for the two histories, relation row, invariant readback, symmetry row, continuation routes, package row, and local name row;*

6. *C stores the displayed* Cont *routes for two-history projection, relation inspection, unary readback, symmetry comparison, and downstream consumer handoff;*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, Hist, BHist, hsame, Cont, Pkg, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{InterHistInvariant}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the two histories, the relation or causal-witness row, the unary readback row, and the observer-symmetry row to a common finite two-history surface. The carrier has no coordinate for a global frame, synchronized clock, metric geometry, host real equality, quotient construction, or observer-independent ambient object.

Lean4 ✓ `BEDC.Derived.InterHistInvariantUp.InterHistInvariantTasteGate single carrier alignment`

Theorem 656.2 (InterHistInvariant NameCert obligations)**.** *For the carrier of Definition 656.1, the local* $\mathsf{NameCert}_{\mathsf{InterHistInvariant}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $I = (A, B, R, V, S, H, C, P, N)$;*
2. *two-history projection through A, B and the relation or causal-witness row R;*
3. *unary invariant readback through V after componentwise* hsame *transport;*
4. *observer-symmetry through the row S, with the swapped packet returning to the same readback surface;*
5. *consumer non-escape for global frames, synchronized clocks, metric geometry, host real equality, and quotient constructions.*

Proof. Carrier admission is the projection of Definition 656.1. The rows $A, B, R, V, S, H, C, P, N$ display exactly the two histories, the relation or causal-witness row, the unary invariant readback, the symmetry row, transport, route, provenance, and local name.

The two-history projection uses C to read A and B before it reads R. The relation row is therefore part of a finite multi-history or cross-history packet, not an ambient relation over host objects.

The unary readback is the row V transported by H. Observer symmetry is then the row S, which compares the original and swapped two-history presentations after the same displayed transport.

Since no carrier coordinate names a global frame, synchronized clock, metric geometry, host real equality, or quotient construction, the local NameCert obligations cannot export any such object. □

Theorem 656.3 (InterHistInvariant hsame stability)**.** *For accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packets $I = (A, B, R, V, S, H, C, P, N)$ and $I' = (A', B', R', V', S', H', C', P', N')$, componentwise* hsame *transport of the two history rows, relation or causal-witness rows, unary readback rows, and observer-symmetry rows preserves the* $\mathsf{InterHistInvariant}^{\uparrow}$ *classifier. The transported packet still reads only the displayed two-history surface and its local invariant row.*

Proof. Project both carriers and transport the history rows through H and H'. The hypotheses put A, B and A', B' on a common two-history surface before either invariant row is read.

Next transport the relation or causal-witness rows R and R'. The continuation routes in C and C' replay the same finite relation inspection after transport, so the classifier has no alternate source for inter-history content.

Finally read the unary rows V, V' and the symmetry rows S, S'. Because these rows are part of the same transported packet, the classifier preserves the accepted invariant surface and does not introduce a global frame, clock, metric, host equality, or quotient. □

Theorem 656.4 (InterHistInvariant observer symmetry)**.** *For an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *carrier* $I = (A, B, R, V, S, H, C, P, N)$*, swapping the two history rows produces a readback that factors through the same relation or causal-witness row, the same unary invariant row, and the same observer-symmetry row after displayed* hsame *transport. The invariant is observer-symmetric as a finite packet read, not as an equality inside a global frame.*

Proof. Start with the two projected history rows A and B. The swapped presentation uses the same carrier rows with the two history projections interchanged.

Apply the componentwise transport row H and the routes in C. These rows carry both the original and swapped presentations to the common finite two-history surface on which R and V are read.

The row S is the displayed symmetry witness between those two presentations. It identifies the invariant readback at the packet surface and supplies no synchronized clock, metric comparison, host real equality, or quotient of the two histories. □

Theorem 656.5 (InterHistInvariant maximum-rate consumer boundary)**.** *Every* $\mathsf{MaxCausalRate}^{\uparrow}$ *consumer of an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet reads the two history rows* A, B*, the relation or causal-witness row* R*, the unary readback row* V*, the observer-symmetry row* S*, and the provenance rows* H, C, P, N *before it reads a maximum-rate boundary. The consumer receives a finite invariant packet over* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *and* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows; it receives no global frame, synchronized clock, metric geometry, host real-valued speed, host real equality, or quotient construction.*

Proof. Project the carrier. A maximum-rate consumer first sees the two history rows, the relation or causal-witness row, the invariant readback, and the symmetry row because those are the only coordinates that can name inter-history content.

Use Definition 656.2 to order the read through carrier admission, two-history projection, unary readback, and observer symmetry. Then use Definition 656.3 to transport the same rows to the consumer surface.

The observer-symmetry theorem supplies the packet read that a maximum-rate boundary may consume. Since the packet exports only $A, B, R, V, S, H, C, P, N$, the consumer has no source for global frames, synchronized clocks, metric geometry, host real-valued speeds, host real equality, or quotient constructions. □

Theorem 656.6 (InterHistInvariant two-history route exactness)**.** *For every accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet*

$$I = (A, B, R, V, S, H, C, P, N),$$

all two-history content exposed by the packet is reached through the rows A, B, R, H, C, P, N*. The rows* A *and* B *are the only history inputs,* R *is the only relation or causal-witness row, and* H, C, P, N *are the only transport, continuation, provenance, and naming rows by which that content can be replayed. Hence every two-history read factors through the sibling surfaces* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$*,* $\mathsf{CrossHistCausal}^{\uparrow}$*, and* $\mathsf{MaxCausalRate}^{\uparrow}$*, and no read can introduce a global frame, synchronized clock, metric geometry, host equality, or quotient.*

Proof. Project the carrier of Definition 656.1. The only rows that name two histories are A and B, and the only row that relates them is R.

The structural route is then forced by the same carrier projection. The row H supplies componentwise hsame transport, C supplies the displayed Cont routes, P supplies package provenance, and N supplies the local naming row.

Use Definition 656.2 and Definition 656.3. They identify the two-history projection and its transported replay with the finite multi-history, cross-history, and maximum-rate consumer surfaces already named in the packet. □

Theorem 656.7 (InterHistInvariant unary readback coverage)**.** *For every accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet* $I = (A, B, R, V, S, H, C, P, N)$*, the unary readback route consumed by* $\mathsf{MaxCausalRate}^{\uparrow}$ *is exhausted by the rows* V *and* S *after the two-history route has displayed* A, B, R, H, C, P, N*. The row* V *is the invariant readback row, and* S *is the observer-symmetry row that keeps the swapped two-history presentation on the same unary surface. These rows provide the complete maximum-rate consumer handoff and do not introduce a global frame, synchronized clock, metric geometry, host equality, or quotient.*

Proof. First expose the two-history route by Definition 656.6. The unary readback is not available before the rows A, B, R, H, C, P, N have supplied the finite two-history surface.

Now project the unary rows in the carrier. The row V records the invariant readback obtained from the transported pair and relation row, while S records the observer-symmetry comparison for the swapped presentation.

Finally apply Definition 656.4 and Definition 656.5. Together they make V and S the whole unary surface consumed by $\mathsf{MaxCausalRate}^{\uparrow}$, with no carrier coordinate left for a global or quotient object. □

Theorem 656.8 (InterHistInvariant finite-multihistory projection lock)**.** *Let* $I = (A, B, R, V, S, H, C, P, N)$ *be an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet. If* R *is the relation row supplied by* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$*, then every read of* A, B, R, V, S *factors through the finite two-history packet displayed by* A, B, R, H, C, P, N*. No consumer of this packet receives an ambient relation on histories, a host relation outside* R*, or a second relation row not present in the carrier.*

Proof. Project the carrier in Definition 656.1. The only history coordinates are A and B, and the only relation coordinate in the finite-multihistory case is the displayed row R.

Use Definition 656.2 to order the read through carrier admission and two-history projection. The continuation routes in C inspect R after A and B are displayed, so relation access is locked to the same finite packet.

Finally apply Definition 656.3. Transport replays the same rows A, B, R, V, S on a common finite surface and supplies no coordinate from which an ambient relation or additional relation row could be read. □

Theorem 656.9 (InterHistInvariant causal-row choice boundary)**.** *Let* $I = (A, B, R, V, S, H, C, P, N)$ *be an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet. If* R *is supplied by* $\mathsf{CrossHistCausal}^{\uparrow}$*, then consumers receive only the displayed causal-witness row together with* A, B, V, S, H, C, P, N*. They do not receive a global causal order, a synchronized clock, a metric time coordinate, or any rule choosing causal comparisons beyond the row* R*.*

Proof. The carrier of Definition 656.1 gives a single slot for the relation or causal-witness row. In the causal case that slot is R, and there is no second coordinate for an order, clock, or metric time.

Apply Definition 656.2. The causal witness is read only through the two-history projection and then through the unary invariant readback, so the consumer sees the displayed causal row as packet data.

The stability theorem Definition 656.3 transports this same row with the two histories and the readback rows. Transport preserves the finite causal witness surface; it does not create a global causal order or synchronized clock. □

Theorem 656.10 (InterHistInvariant rate-consumer readback exhaustion)**.** *Every* $\mathsf{MaxCausalRate}^{\uparrow}$ *consumer of an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet receives exactly the rows*

$$A, B, R, V, S, H, C, P, N$$

before any maximum-rate boundary is read. This readback exhausts the available inter-history invariant surface: the consumer has no host real-valued speed, host real equality, quotient construction, metric geometry, synchronized clock, or global frame from which to refine the rate boundary.

Proof. First project the carrier of Definition 656.1. The row list $A, B, R, V, S, H, C, P, N$ is the complete list of visible carrier coordinates.

Use Definition 656.2 to pass from the carrier list to the local obligation surface, and use Definition 656.3 to transport that same surface to the consumer. No step adds a numerical speed or quotient coordinate.

Finally apply Definition 656.5. The maximum-rate boundary consumes the finite packet after those rows are displayed, so the rate-facing readback is exhausted by the carrier inventory itself. □

Theorem 656.11 (InterHistInvariant consumer row exactness)**.** *For every accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet*

$$I = (A, B, R, V, S, H, C, P, N),$$

every consumer read of $A, B, R, V, S, H, C, P, N$ *is routed through the finite packet rows supplied by* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ *or through the displayed causal row supplied by* $\mathsf{CrossHistCausal}^{\uparrow}$*. The consumer receives exactly the two history slots, relation or causal row, unary invariant row, observer-symmetry row, transport row, continuation row, provenance row, and local naming row; it receives no global frame, synchronized clock, metric geometry, host equality, or quotient route.*

Proof. Project the carrier of Definition 656.1. The tuple lists the two history slots A, B, the relation or causal row R, the unary readback V, the symmetry row S, and the structural rows H, C, P, N.

Use Definition 656.6 for the route to A, B, R, H, C, P, N. If R is a finite-multihistory relation row, the route is locked by Definition 656.8; if R is causal, it is confined by Definition 656.9.

Finally apply Definition 656.7. It exhausts the remaining consumer-facing rows V and S, so no off-packet frame, clock, metric, host equality, or quotient coordinate is available to the consumer. □

Theorem 656.12 (InterHistInvariant rate handoff exactness)**.** *Every* $\mathsf{MaxCausalRate}^{\uparrow}$ *read exported from an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet consumes only the displayed unary invariant row* V*, the observer-symmetry row* S*, and the structural rows* H, C, P, N *after the two-history relation row* R *has been displayed. The handoff does not consume a hidden speed coordinate, synchronized clock, metric geometry, host real equality, quotient of histories, or global reference frame.*

Proof. Begin with Definition 656.5. It states that the maximum-rate consumer first reads $A, B, R, V, S, H, C, P, N$ from the finite packet.

The unary part of the handoff is then exactly Definition 656.7. The row V supplies the invariant readback, and S supplies the observer-symmetry comparison on the same transported two-history surface.

Use Definition 656.10 to finish the boundary. Since the row inventory is exhausted before the rate boundary is read, no hidden speed, clock, metric, host equality, quotient, or global frame can enter the handoff. □

Theorem 656.13 (InterHistInvariant finite source scope)**.** *For every accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *carrier*

$$I = (A, B, R, V, S, H, C, P, N),$$

the history rows A, B*, relation or causal row* R*, transport row* H*, continuation row* C*, package row* P*, and local naming row* N *are scoped to the finite source surface of* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$*. The packet may project the two history slots and the relation row through that finite source, but it has no coordinate for a global frame, synchronized clock, metric geometry, host equality, quotient of histories, or observer-independent ambient object.*

Proof. Project the carrier in Definition 656.1. The only history inputs are the displayed rows A and B, and the only finite relation slot is R.

Apply Definition 656.8. It already binds the relation-row case to the $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ source and prevents an ambient relation or second relation row from entering the packet.

Now use Definition 656.6. The transport, continuation, provenance, and naming rows H, C, P, N are the only replay boundary for the two-history content. Since the carrier has no further coordinate, the finite source scope is exactly the displayed BHist route. □

Theorem 656.14 (InterHistInvariant causal rate scope)**.** *For every accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *carrier* $I = (A, B, R, V, S, H, C, P, N)$*, causal-row use is scoped to* $\mathsf{CrossHistCausal}^{\uparrow}$ *and the downstream unary rate boundary is scoped to* $\mathsf{MaxCausalRate}^{\uparrow}$*. A consumer may read* R *as the displayed causal row and may then read* V, S, H, C, P, N *as the finite invariant handoff to the maximum-rate surface, but it cannot read a global causal order, synchronized clock, metric time coordinate, host real speed, host equality, or quotient construction.*

Proof. Begin with the causal-row boundary of Definition 656.9. It identifies R as the only causal slot and confines it to the displayed $\mathsf{CrossHistCausal}^{\uparrow}$ row.

The rate-facing side is Definition 656.12. It routes any $\mathsf{MaxCausalRate}^{\uparrow}$ read through V, S, H, C, P, N after the two-history relation row has been displayed.

Combine these two boundaries with Definition 656.5. The consumer sees a finite causal invariant packet before the rate boundary, so there is no route to a global order, clock, metric geometry, host speed, host equality, or quotient. □

Theorem 656.15 (InterHistInvariant consumer kernel scope)**.** *Every consumer of an accepted* $\mathsf{InterHistInvariant}^{\uparrow}$ *packet factors through* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, and the checked TasteGate carrier surface for the displayed tuple*

$$I = (A, B, R, V, S, H, C, P, N).$$

The consumer receives the two history slots, relation or causal row, unary invariant row, observer-symmetry row, transport row, continuation row, package provenance, and local name row, and receives no global frame, synchronized clock, metric geometry, host equality, quotient construction, or off-packet causal data.

Lean4 ✓ `BEDC.Derived.InterHistInvariantUp.InterHistInvariantUp StdBridge`

Proof. First apply Definition 656.11. It states that every consumer read of $A, B, R, V, S, H, C, P, N$ is routed through the finite packet rows supplied by $\mathsf{FiniteMultiHistPacket}^{\uparrow}$ or through the displayed causal row supplied by $\mathsf{CrossHistCausal}^{\uparrow}$.

Next apply Definition 656.13 and Definition 656.14. These two scope the finite-source and causal-rate cases without adding a global frame, synchronized clock, metric geometry, host equality, quotient, or off-packet causal source.

Finally align the packet with the checked TasteGate carrier surface named in Definition 656.16. The TasteGate row preserves the same displayed fields, while BHist, hsame, Cont, Pkg, and NameCert provide the kernel route by which consumers read them. □

Theorem 656.16 (InterHistInvariant TasteGate field contract)**.** *The checked TasteGate anchor for* $\mathsf{InterHistInvariant}^{\uparrow}$ *is the single-carrier alignment*

BEDC.Derived.InterHistInvariantUp.InterHistInvariantTasteGate_single_carrier_alignment.

The Lean TasteGate surface for this same packet must include explicit Nontrivial $\mathsf{InterHistInvariant}^{\uparrow}$ *and* FieldFaithful $\mathsf{InterHistInvariant}^{\uparrow}$ *instances preserving the two history slots, relation row, invariant row, symmetry row, transport row, continuation route, package provenance, and local name row.*

Proof. The paper anchor is the checked marker immediately after Definition 656.1. It identifies the TasteGate surface with the displayed carrier rather than with an external event-flow object.

Nontriviality is required because a singleton carrier would erase the distinction between the two history slots, the relation row, the invariant row, and the symmetry row. Such a collapse cannot support the consumer exactness and rate handoff theorems above.

Field faithfulness is required because each carrier coordinate has a separate consumer role. Preserving $A, B, R, V, S, H, C, P, N$ as distinct fields keeps the TasteGate route aligned with the same finite packet read by the paper obligations. □

Lean4 ✓ `BEDC.Derived.InterHistInvariantUp.taste gate`

Theorem 656.17 (InterHistInvariant obligation closure package)**.** *The carrier of Definition 656.1, the local obligations of Definition 656.2, hsame stability from Definition 656.3, observer symmetry from Definition 656.4, the maximum-rate boundary of Definition 656.5, two-history route exactness from Definition 656.6, unary readback coverage from Definition 656.7, the finite-multihistory projection lock of Definition 656.8, the causal-row choice boundary of Definition 656.9, rate-consumer readback exhaustion from Definition 656.10, consumer row exactness from Definition 656.11, rate handoff exactness from Definition 656.12, and the TasteGate field contract of Definition 656.16 form the full obligation surface for* $\mathsf{InterHistInvariant}^{\uparrow}$ *over the finite packet*

$$I = (A, B, R, V, S, H, C, P, N).$$

Every accepted packet is therefore certified by carrier admission, two-history route exactness, unary readback coverage, componentwise transport stability, observer symmetry, maximum-rate consumer boundary, package provenance, and local naming, and it exports no global frame, synchronized clock, metric geometry, host equality, or quotient.

Proof. The carrier and NameCert rows give the admission layer. They display exactly the rows $A, B, R, V, S, H, C, P, N$ and list the local obligations that must be discharged for the packet.

The route layer is supplied by the two exactness theorems. Two-history content is confined to A, B, R, H, C, P, N, while the maximum-rate unary readback is confined to V, S after those rows have been displayed.

The projection and choice layers are supplied by the finite-multihistory lock and causal-row boundary. They distinguish the relation-row and causal-row readings of R without adding any ambient relation, global order, synchronized clock, or metric time.

The stability and consumer layers are supplied by the remaining local theorems. Componentwise hsame transport preserves the same packet surface, observer symmetry keeps the swapped presentation on that surface, and the maximum-rate boundary together with rate-consumer exhaustion gives the downstream handoff.

These pieces exhaust the visible carrier coordinates. Since no additional coordinate names a global frame, synchronized clock, metric geometry, host equality, or quotient, the listed rows are the full obligation surface for $\mathsf{InterHistInvariant}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{InterHistInvariant}^{\uparrow}}$.

Constructive story: An $\mathsf{InterHistInvariant}^{\uparrow}$ packet is generated from two history rows, a finite relation or causal-witness row, a unary invariant readback row, an observer-symmetry row, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InterHistInvariant}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 656.13, Definition 656.14, and Definition 656.15 bind the finite source rows, causal-rate rows, consumer exactness, and TasteGate carrier surface for inter-history invariant packets, with TasteGate instance Lean4 ✓ `BEDC.Derived.InterHistInvariantUp.taste gate`

.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InterHistInvariant}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.InterHistInvariantUp.InterHistInvariantTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close global frames, synchronized clocks, metric geometry, host real equality, quotient constructions, or standard bridge checking beyond the checked TasteGate carrier alignment.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 657

A Concrete Naming Certificate for $\mathsf{ContinuationBigStep}^{\uparrow}$

The $\mathsf{ContinuationBigStep}^{\uparrow}$ packet realizes terminal-read computation as a finite continuation certificate. It reads a big-step result only from the terminal row of a displayed small-step trace, the Cont replay route, componentwise hsame transport, package provenance, and local naming. The packet adds no host evaluator, semantic evaluation relation, quotient behaviour class, or hidden totality principle.

This packet is a downstream consumer of the small-step trace carrier in chapter 469. The trace ledger T is read as $\mathsf{ContinuationTrace}^{\uparrow}$ before C replays it to the terminal row R and the terminal-read witness Q.

Definition 657.1 (ContinuationBigStep carrier)**.** *A* $\mathsf{ContinuationBigStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, T, R, Q, H, C, P, N)$$

whose displayed coordinates are as follows:

1. *S is the source row whose computation is being read;*
2. *T is the finite small-step trace ledger, displayed as the ordered* BHist, BMark, *tail, and target rows already readable by* $\mathsf{ContinuationTrace}^{\uparrow}$*;*
3. *R is the terminal row reached by the last displayed trace entry;*
4. *Q is the terminal-read witness that records the result read from R;*
5. *H is the componentwise* hsame *transport row for the source, trace, terminal, terminal-read, continuation, provenance, and naming coordinates;*
6. *C is the* Cont *replay route that replays the finite trace ledger from S to R;*
7. *P is the* Pkg *provenance over* BHist, BMark, $\mathsf{ContinuationTrace}^{\uparrow}$, hsame, Cont, *terminal-read witnesses, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *row.*

Two accepted packets classify together only when these eight displayed coordinates are transported componentwise. The terminal-read witness Q is a read from R along the listed replay route, not an external big-step semantic object.

Theorem 657.2 (ContinuationBigStep NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *surface for the carrier of Definition 657.1 consists exactly of five finite obligations:*

1. *carrier admission for the packet* $B = (S, T, R, Q, H, C, P, N)$*;*
2. *small-step trace factorization of* T *through the* $\mathsf{ContinuationTrace}^{\uparrow}$ *rows named by* P*;*
3. *terminal readback from the terminal row* R *through the witness* Q*;*
4. *non-escape from host evaluators, host big-step semantics, quotient behaviours, and hidden termination principles;*
5. *consumer exactness for reads through* S, T, R, Q, H, C, P, N *and no other coordinates.*

Lean4 ✓ `BEDC.Derived.ContinuationBigStepUp.ContinuationBigStepCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 657.1. The accepted packet exposes the source, trace ledger, terminal row, terminal-read witness, transport, replay, provenance, and local naming coordinates.

The small-step factorization obligation is supplied by the $\mathsf{ContinuationTrace}^{\uparrow}$ route recorded in T and named by P. Each displayed step of the ledger is replayed through the continuation row C, with componentwise movement controlled by H.

Terminal readback then projects Q from the terminal row R. The witness reads only the row reached by the finite replay and therefore contributes a terminal observation, not a host-level evaluator result.

The non-escape and consumer-exactness clauses follow from the same coordinate list. Since the carrier has no field for an ambient evaluation relation, quotient behaviour, selected evaluator, or hidden termination principle, every consumer route is exhausted by S, T, R, Q, H, C, P, N. □

Theorem 657.3 (ContinuationBigStep small-step factorization)**.** *For an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *carrier* $B = (S, T, R, Q, H, C, P, N)$*, every terminal computation read factors through a finite route*

$$S \longrightarrow T \longrightarrow C \longrightarrow R \longrightarrow Q.$$

The trace coordinate T *is read as a* $\mathsf{ContinuationTrace}^{\uparrow}$ *small-step packet,* C *replays its listed* Cont *rows,* R *is the terminal row reached by that replay, and* Q *is the terminal-read witness attached to* R*. Conversely, a consumer that reads* Q *from* R *after replaying* T *through* C *reads exactly the big-step packet surface carried by* B*, with transport and provenance supplied by* H, P, N*.*

Lean4 ✓ `BEDC.Derived.ContinuationBigStepUp.ContinuationBigStepPacket small step factorization`

Proof. Begin with the trace ledger T. By the $\mathsf{ContinuationTrace}^{\uparrow}$ small-step readback surface, each row of T is a finite source, mark, tail, and target read, and no step is imported from an external evaluator.

Replay the listed rows through C. The continuation replay composes the displayed route from the source row S to the terminal row R, while H accounts only for componentwise hsame transport of the listed coordinates.

Read the terminal witness Q from R. This gives the terminal-read surface of the packet, and P, N bind it to the package provenance and local naming row.

For the reverse direction, suppose a consumer starts with the displayed replay of T through C and then reads Q at R. The carrier has exactly the coordinates required to repack that route as B, so the consumer obtains the same finite terminal-read packet and no host-wide big-step semantics. □

Theorem 657.4 (ContinuationBigStep terminal route exhaustion)**.** *Every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *consumer route factors through the displayed finite route*

$$S \longrightarrow T \longrightarrow C \longrightarrow R \longrightarrow Q.$$

Once the source row S, trace ledger T, replay route C, and terminal row R have been fixed, the only terminal-read data available to the consumer is the witness Q transported by the componentwise hsame *rows. The carrier provides no direct $S \to Q$ shortcut, no host big-step semantic relation, no quotient behaviour class, and no hidden termination row.*

Proof. Unpack the carrier of Definition 657.1. An accepted packet has exactly the displayed rows S, T, R, Q, H, C, P, N, with T serving as the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, C serving as the replay route, and Q serving as the read from the terminal row R.

Apply Definition 657.3. It routes every terminal computation read from S through T, then through C, then through R, and only then to Q.

Use Definition 657.5 for the endpoint read. After S, T, C, R are fixed, the terminal witness can move only by the componentwise hsame transport already displayed in the packet.

Finally combine Definition 657.6 with Definition 657.7. These results exclude a direct source-to-witness edge, a host evaluator, a quotient behaviour classifier, and any hidden termination row outside the finite route. □

Theorem 657.5 (ContinuationBigStep terminal-read determinacy)**.** *Let*

$$B_0 = (S_0, T_0, R_0, Q_0, H_0, C_0, P_0, N_0) \qquad \textit{and} \qquad B_1 = (S_1, T_1, R_1, Q_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{ContinuationBigStep}^{\uparrow}$ *carriers. Suppose their source rows, small-step trace ledgers, replay routes, and terminal rows are classifier-same:*

$$S_0 \mathsf{hsame} S_1, \qquad T_0 \mathsf{hsame} T_1, \qquad C_0 \mathsf{hsame} C_1, \qquad R_0 \mathsf{hsame} R_1.$$

Then the terminal-read witnesses are classifier-same,

$$Q_0 \mathsf{hsame} Q_1,$$

and both accepted big-step reads factor through one finite route presentation. More precisely, the read of B_0 factors through

$$S_0 \longrightarrow T_0 \longrightarrow C_0 \longrightarrow R_0 \longrightarrow Q_0,$$

while the read of B_1 is the classifier transport of this same route through the displayed comparisons on S, T, C, R and the induced comparison on Q.

Proof. By Definition 657.1, an accepted $\mathsf{ContinuationBigStep}^{\uparrow}$ packet has exactly the displayed coordinates S, T, R, Q, H, C, P, N. The coordinate T is the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, C is the displayed Cont replay route from the source row to the terminal row, R is the terminal row reached by that replay, and Q is the terminal-read witness attached to R.

Apply Definition 657.3 to B_0 and B_1. The first packet reads through

$$S_0 \longrightarrow T_0 \longrightarrow C_0 \longrightarrow R_0 \longrightarrow Q_0,$$

and the second reads through

$$S_1 \longrightarrow T_1 \longrightarrow C_1 \longrightarrow R_1 \longrightarrow Q_1.$$

The hypotheses identify the source, ledger, replay, and terminal rows of these two routes by the local classifier. Hence the two routes have the same finite replay spine after componentwise transport.

The only remaining endpoint datum is the terminal-read row. By Definition 657.2, terminal readback is read from the terminal row through the witness Q, and consumer exactness allows reads only through the displayed coordinates S, T, R, Q, H, C, P, N. Since the source, trace ledger, replay route, and terminal row have already been matched, there is no second carrier coordinate from which a different terminal-read endpoint can be selected. Transporting the readback of Q_0 along $R_0 \mathsf{hsame} R_1$ and the matched replay route therefore gives the readback of Q_1, so $Q_0 \mathsf{hsame} Q_1$.

Thus the two accepted big-step reads are not two independent terminal computations. They are the same finite small-step replay followed by the same terminal-read operation, with the second presentation obtained from the first by classifier transport along the displayed source, trace, route, terminal, and terminal-read comparisons. □

Theorem 657.6 (ContinuationBigStep non-escape)**.** *Every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet route factors through the finite rows*

$$S, T, R, Q, H, C, P, N.$$

It cannot export a host evaluator, a host big-step semantics, a quotient behaviour class, or a hidden termination principle. The only terminal computation data available to consumers is the finite small-step ledger T, its replay through C, the reached terminal row R, and the terminal-read witness Q, with transport, provenance, and naming supplied by H, P, N.

Proof. Inspect the carrier in Definition 657.1. Its coordinates are exactly the source row, trace ledger, terminal row, terminal-read witness, componentwise transport, displayed continuation replay, package provenance, and local naming row.

Apply Definition 657.3. Every terminal computation read is routed through the finite spine from S through T and C to R and then to Q.

None of the displayed carrier rows is a host evaluator, host big-step semantics, quotient behaviour class, or hidden termination principle. Since the accepted route has no other coordinate to export, the packet preserves the stated non-escape boundary. □

Theorem 657.7 (ContinuationBigStep consumer exactness)**.** *Every terminal-read consumer of an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet first reads the finite small-step trace ledger T, replays the ledger through the displayed continuation row C, reaches the terminal row R, and then reads the witness Q. Transport and provenance may move this route only through H, P, N.*

Consequently a consumer of the terminal read consumes exactly the finite route

$$S \longrightarrow T \longrightarrow C \longrightarrow R \longrightarrow Q,$$

with no direct read from S to Q and no host-level big-step shortcut.

Lean4 ✓ `BEDC.Derived.ContinuationBigStepUp.ContinuationBigStepConsumerExactness`

Proof. Begin with Definition 657.3. It gives the displayed route from the source row through the trace ledger and continuation replay to the terminal row and witness.

Use Definition 657.5. A terminal read is determined by the source row, trace ledger, replay route, and terminal row up to the displayed classifier transport, so a consumer cannot choose an independent terminal witness.

The remaining rows H, P, N account for componentwise transport, package provenance, and local naming. They move or name the same finite route; they do not add a second evaluator or a host-level terminal semantics. □

Theorem 657.8 (ContinuationBigStep trace-row admission)**.** *Every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet admits its trace row only as the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *small-step ledger* T *in Definition 657.1. The carrier, package, and local* NameCert *rows may name and transport that ledger, but they cannot replace it by a host evaluator, a quotient behaviour class, or an unrecorded termination witness.*

Proof. Start with the carrier tuple in Definition 657.1. Its trace coordinate is the displayed finite ledger T, and the surrounding rows are source, terminal, terminal-read, transport, continuation, provenance, and naming data.

Use Definition 657.2. The local certificate binds admission to those displayed rows, so the trace row must be the row named by the packet.

Finally use Definition 657.3. Every accepted terminal computation factors through the same small-step ledger, so there is no independent trace source available to consumers.

The consumer boundary is the one stated in Definition 657.7. Consumers therefore receive this admitted trace row only through the finite replay route, not through a host-level big-step shortcut. □

Theorem 657.9 (ContinuationBigStep terminal-row readback)**.** *For every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet, the terminal-read witness* Q *is reachable only after replaying the displayed trace ledger* T *through the continuation row* C *to the terminal row* R*. A consumer therefore reads* Q *through the finite route*

$$S \longrightarrow T \longrightarrow C \longrightarrow R \longrightarrow Q,$$

with transport and package provenance supplied by the displayed H, P, N *rows.*

Proof. Start with the carrier tuple of Definition 657.1. Its visible rows are the only rows available for terminal readback.

Use Definition 657.2 to bind those rows to the local NameCert obligation package. The certificate does not name an additional terminal computation.

Then use Definition 657.3 to obtain the replay route from the source row through the trace ledger and the continuation row to the terminal row.

Finally use Definition 657.7. It states that terminal-read consumers reach Q only after the trace replay reaches R, and that H, P, N merely transport and name that same finite route.

The route therefore has no direct edge from S to Q and no host-level big-step shortcut. □

Theorem 657.10 (ContinuationBigStep trace-prefix readback)**.** *Let* $K = (S, T, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *carrier. Any consumer that reads a terminal prefix of the computation before* R *must read that prefix through the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger* T*, transported by* H*, replayed through* C*, and named by* P, N*. The packet has no direct terminal prefix row outside the trace ledger.*

Proof. Start with the carrier of Definition 657.1. Its source row is S, its finite trace ledger is T, and its replay endpoint is R.

Apply the small-step factorization theorem Definition 657.3. It makes the route from S to R pass through the displayed trace ledger, so a prefix read before R is a read of an initial segment of T.

Use Definition 657.8 to admit the trace row as the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger. The rows H, C, P, N then transport, replay, source, and name that same prefix, without adding a separate terminal prefix coordinate. □

Theorem 657.11 (ContinuationBigStep terminal-witness exhaustion)**.** *Let $K = (S, T, R, Q, H, C, P, N)$ be an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *carrier. Once S, T, C, R are fixed, the terminal-read witness displayed by the packet is exhausted by Q. Any accepted terminal read using the same source, trace, continuation route, and replay endpoint names the same witness row Q and has no second terminal witness outside the carrier.*

Proof. Read S, T, C, R from Definition 657.1. These rows fix the source, the finite replay ledger, the displayed continuation route, and the replay endpoint.

Apply terminal-read determinacy from Definition 657.5. With the replay route and endpoint fixed, a terminal consumer cannot choose a different terminal row.

The row Q is the only terminal-read witness coordinate in the carrier. The transport and naming rows H, P, N preserve that witness, and C is already one of the fixed rows, so the accepted packet exhausts terminal witnesses at Q. □

Theorem 657.12 (ContinuationBigStep small-step closure package)**.** *The* $\mathsf{ContinuationBigStep}^{\uparrow}$ *dependency-ready obligation package consists of the carrier $K = (S, T, R, Q, H, C, P, N)$, local NameCert obligations, small-step factorization, terminal-read determinacy, non-escape, consumer exactness, trace-row admission, trace-prefix readback, and terminal-row readback for the same finite packet. This package exports a finite* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surface, not a host big-step evaluator.*

Proof. Carrier admission and the local naming obligations are those of Definition 657.1 and Definition 657.2.

The operational spine is supplied by Definition 657.3, Definition 657.5, and Definition 657.6. These bind the small-step route, the terminal witness, and the non-escape boundary for the same rows S, T, R, Q, H, C, P, N.

The consumer surface is supplied by Definition 657.7, Definition 657.8, Definition 657.10, and Definition 657.9. Together they make the trace row and terminal row readable by downstream packets without adding a host evaluator.

Thus the package is dependency-ready exactly as a finite BEDC packet: the trace ledger, terminal witness, transport, continuation, provenance, and local name are all displayed coordinates, and no semantic big-step shortcut is exported. □

Theorem 657.13 (ContinuationBigStep local NameCert package)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *obligation package for an accepted packet is the assembly of carrier admission, trace-row admission, terminal-row readback, non-escape, and consumer exactness over the same rows*

$$S, T, R, Q, H, C, P, N.$$

It records a finite continuation computation packet and does not export host evaluator semantics, quotient behaviour, or hidden termination.

Proof. Carrier admission is given by Definition 657.1, and the local certificate obligations are those of Definition 657.2.

Trace factorization is supplied by Definition 657.3 and by Definition 657.8. The accepted trace is the displayed small-step ledger T.

Terminal readback is supplied by Definition 657.9, while non-escape is the boundary of Definition 657.6. The consumer surface is exactly Definition 657.7.

Together these clauses package the same finite rows S, T, R, Q, H, C, P, N under the local $\mathsf{NameCert}$ row, with no additional host evaluator or hidden semantic carrier. □

Theorem 657.14 (ContinuationBigStep subject-reduction socket handoff)**.** *Every subject-reduction discharge consumer of an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet must read the source row* S*, the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *small-step ledger* T*, the terminal row* R*, the terminal-read witness* Q*, the componentwise transport row* H*, the displayed continuation route* C*, the provenance row* P*, and the local naming row* N *before it reaches any discharge socket. Thus the only handoff route into a preservation-facing consumer is the terminal-read route displayed by the finite ledger* S, T, R, Q, H, C, P, N*.*

Proof. Start with carrier admission in Definition 657.1. It displays exactly the rows S, T, R, Q, H, C, P, N and makes T the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger.

Use Definition 657.3 to route the source row through the small-step ledger and continuation row before any terminal row is read. Then use Definition 657.9 to force the terminal consumer through R and Q.

Finally apply Definition 657.7 and Definition 657.13. These identify the only downstream consumer surface with the displayed carrier, trace, terminal readback, provenance, and local $\mathsf{NameCert}$ rows, so a discharge socket cannot be reached by a route outside the finite packet. □

Theorem 657.15 (ContinuationBigStep preservation-boundary non-escape)**.** *The subject-reduction handoff of Definition 657.14 exports no host evaluator, hidden termination row, quotient behaviour, or preservation theorem beyond the displayed terminal-read route. Any preservation-facing consumer receives only the finite ledger rows* S, T, R, Q, H, C, P, N *and the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *surface.*

Proof. Inspect the carrier of Definition 657.1. Its coordinates include the source, trace, terminal row, terminal witness, transport, route, provenance, and local name, and they include no host evaluator, quotient coordinate, or hidden termination row.

The operational information available to a consumer is exactly the factorization of Definition 657.3 and the terminal readback of Definition 657.9. The consumer exactness theorem Definition 657.7 prevents these rows from being enlarged by a downstream read.

The local package theorem Definition 657.13 records the same boundary at the naming surface. Therefore the handoff may feed a preservation-facing socket, but it does not itself contain a preservation theorem or any additional semantic route. □

Theorem 657.16 (ContinuationBigStep terminal ledger package)**.** *For an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet, carrier admission, trace-row admission, terminal-row readback, consumer exactness, and the subject-reduction socket handoff assemble into the dependency-ready terminal ledger package over the finite rows* S, T, R, Q, H, C, P, N*. The package is a BEDC terminal-read ledger for downstream discharge packets, not a host big-step semantics.*

Proof. Carrier admission comes from Definition 657.1, while trace admission is the small-step factorization of Definition 657.3. These two items fix the source row, trace ledger, continuation route, and terminal row of the packet.

Terminal readback is exactly Definition 657.9, and consumer exactness is exactly Definition 657.7. Together they make the terminal witness and every downstream read pass through the displayed rows.

The local packaging theorem Definition 657.13 and the handoff theorem Definition 657.14 then repack the same data for subject-reduction-facing consumers. The non-escape boundary of Definition 657.15 keeps the result a terminal ledger package rather than a host semantics. □

Theorem 657.17 (ContinuationBigStep trace-ledger threshold)**.** *Every terminal-read consumer of an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet must pass through the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger before the terminal row is readable. Concretely, if* $K = (S, T, R, Q, H, C, P, N)$ *is accepted, then any read of* R *or* Q *factors through the carrier row* T*, the local* $\mathsf{NameCert}$ *obligations, the small-step factorization route, terminal-read determinacy, and consumer exactness for the same finite packet.*

Proof. Start with the carrier of Definition 657.1. The rows S, T, R, Q, H, C, P, N are the only rows available to a terminal-read consumer, and T is the displayed $\mathsf{ContinuationTrace}^{\uparrow}$ ledger.

Apply Definition 657.2 and Definition 657.3. These force terminal computation reads through the route from S to T, then through the displayed continuation replay to R.

Use Definition 657.5 and Definition 657.7. The terminal row and its witness are readable only after the trace ledger and replay route have been fixed, so a consumer cannot reach R or Q by bypassing T. □

Theorem 657.18 (ContinuationBigStep terminal consumer package)**.** *The rows* S, T, R, Q, H, C, P, N *form the complete terminal-read package over* BHist*,* BMark*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The package consists of carrier admission, local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *obligations, small-step factorization through* T*, terminal-read determinacy at* R, Q*, and consumer exactness over the same finite rows.*

Proof. Carrier admission is Definition 657.1, and the naming obligations are Definition 657.2. These two items fix the finite rows S, T, R, Q, H, C, P, N and the BEDC structural objects BHist, BMark, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

The operational route is supplied by Definition 657.3. It makes the terminal read pass from the source row through the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger and the displayed continuation replay before reaching the terminal row.

Finally apply Definition 657.5 and Definition 657.7. They make the terminal witness determined by the displayed route and prevent consumers from importing any row outside the package. □

Theorem 657.19 (ContinuationBigStep replay-route stability)**.** *Let* $K = (S, T, R, Q, H, C, P, N)$ *and* $K' = (S', T', R', Q', H', C', P', N')$ *be accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packets. If the displayed replay spines agree componentwise by* hsame *on the source row, trace ledger, continuation replay, and terminal row, then every terminal-read route transported through* H, C, P, N *remains on the same finite route*

$$S \to T \to C \to R \to Q.$$

In particular, transport along the accepted package cannot replace the route by a host evaluator, a direct source-to-result edge, or an unlisted termination row.

Proof. Begin with the carrier of Definition 657.1. The accepted packet exposes only the rows S, T, R, Q, H, C, P, N, and the second packet exposes the corresponding primed rows with the same componentwise hsame transport data.

Use Definition 657.2 and Definition 657.3. These obligations identify the displayed route from the source row to the trace ledger and then through the continuation replay to the terminal row.

Apply Definition 657.5. Once S, T, C, R have been fixed by componentwise hsame, the terminal read has no second route through which it can reach R or Q.

Finally use Definition 657.7. Consumer transport through H, C, P, N repacks the same finite rows, so the terminal-read route remains the displayed $S \to T \to C \to R \to Q$ route. □

Theorem 657.20 (ContinuationBigStep terminal-read coverage)**.** *Every accepted terminal-read consumer of a* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet is covered by carrier admission, trace-row admission, terminal-row readback, terminal-read determinacy, and consumer exactness for the same rows* S, T, R, Q, H, C, P, N*. The coverage surface contains no additional read authority beyond the finite replay route certified by the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *package.*

Proof. Carrier admission is exactly Definition 657.1. It fixes the rows that the consumer may address and records their structural BEDC objects.

Trace-row admission is supplied by Definition 657.3. Terminal-row readback is the displayed terminal read of Definition 657.9.

Use Definition 657.5 to make the terminal witness determined by those rows. Then apply Definition 657.7 to show that any consumer read is a read of the same finite packet rather than an enlargement of it. □

Theorem 657.21 (ContinuationBigStep NameCert consumer interface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *consumer interface is exactly the carrier row, replay route, terminal witness, componentwise transport, provenance, and local naming rows of an accepted packet. Equivalently, the interface consists of* S, T, R, Q, H, C, P, N *together with the displayed* BHist*,* BMark*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *structure, and it contains no host evaluator or hidden termination row.*

Proof. Read the interface from Definition 657.1. The carrier records the source, trace ledger, terminal row, terminal witness, transports, continuation replay, provenance, and local name.

The naming surface is constrained by Definition 657.2. The replay route is supplied by Definition 657.3, and terminal access is governed by Definition 657.5.

It remains to exclude hidden consumer authority. Apply Definition 657.7 together with Definition 657.19 and Definition 657.20. The resulting interface is exactly the finite local package and not a host big-step evaluator. □

Theorem 657.22 (ContinuationBigStep non-escape scope package)**.** *The terminal consumer package of Definition 657.18 excludes host evaluators, quotient behaviours, hidden termination, and direct source-to-result shortcuts. Every accepted terminal read remains inside the finite rows* S, T, R, Q, H, C, P, N *and the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *surface.*

Proof. Use the carrier and local obligations first. By Definition 657.1 and Definition 657.2, the accepted packet has no coordinate for a host evaluator, quotient behaviour, hidden termination row, or direct source-to-result edge.

Next apply Definition 657.3 and Definition 657.5. They make any terminal result pass through the displayed trace ledger and terminal row rather than through an external evaluator.

The boundary is closed by Definition 657.7. Since consumers can read only the displayed package rows, the non-escape scope is exactly the finite $\mathsf{ContinuationBigStep}^{\uparrow}$ surface. □

Theorem 657.23 (ContinuationBigStep prefix closure)**.** *Every accepted prefix of the small-step trace ledger of an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet remains a finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *route. The prefix route is read through the carrier rows* S, T, R, Q, H, C, P, N*, the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *obligations, the small-step factorization, terminal-read determinacy, and the non-escape boundary; it imports no host evaluator, hidden termination principle, quotient behaviour, or direct source-to-result shortcut.*

Proof. Begin with the carrier of Definition 657.1. The only trace coordinate is T, and it is a finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger attached to the source row S and replay route C.

Apply the local NameCert obligations and the small-step factorization. They make every accepted trace read pass through S, then through the displayed trace ledger T, and then through the replay route toward R.

For an accepted prefix, restrict this same displayed ledger before the terminal row is consumed. Terminal-read determinacy ensures that the full terminal witness is not changed by reading the prefix, and the non-escape theorem prevents a prefix consumer from replacing the finite trace route by a host evaluator or hidden termination row. □

Theorem 657.24 (ContinuationBigStep result stability)**.** *Let two accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packets have classifier-same terminal rows under the displayed* hsame *transport, and suppose their replay routes agree on the source row, trace ledger, continuation replay, and terminal row. Then the terminal-read witnesses induced by the displayed replay are classifier-same. The witness comparison is read only through* S, T, R, Q, H, C, P, N *and the corresponding primed rows, together with carrier admission, NameCert obligations, small-step factorization, terminal-read determinacy, consumer exactness, and non-escape.*

Proof. Start with the two carrier packets. Their source rows, trace ledgers, replay routes, terminal rows, terminal witnesses, transports, provenance, and local names are the only displayed coordinates available for a terminal read.

Use the NameCert obligations and small-step factorization for both packets. These route each terminal read through its displayed trace ledger and continuation replay before reaching the terminal row.

Apply terminal-read determinacy at the classifier-same terminal rows. Since the replay routes agree on the source, trace, continuation, and terminal rows, the terminal witness has no independent route by which it can vary.

Finally use consumer exactness and non-escape. Transport through H, C, P, N and the primed rows repacks the same finite replay surface, so the induced terminal-read witnesses are classifier-same rather than host-evaluator results. □

Theorem 657.25 (ContinuationBigStep ledger totality)**.** *The rows* S, T, R, Q, H, C, P, N *exhaust every* $\mathsf{ContinuationBigStep}^{\uparrow}$ *consumer route. Any accepted big-step consumer must read the source row, finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, terminal row, terminal-read witness, componentwise* hsame *transports, displayed* Cont *replay, package provenance, and local* $\mathsf{NameCert}$ *row; no ninth row supplies a host evaluator, hidden termination, quotient behaviour, or direct source-to-result shortcut.*

Proof. Read the carrier and local NameCert obligations first. They list exactly S, T, R, Q, H, C, P, N as the finite big-step packet and give no additional consumer coordinate.

Next apply the small-step factorization. It accounts for the operational content of the packet by routing the source through the trace ledger and displayed continuation replay before the terminal row is read.

Terminal-read determinacy and consumer exactness then account for R, Q, H, C, P, N. The terminal row and witness are fixed by the displayed replay, while transports, continuations, provenance, and the local name authenticate the same route to consumers.

The non-escape theorem closes the ledger. Since every displayed route has already been assigned to one of S, T, R, Q, H, C, P, N, no host evaluator, hidden termination row, quotient behaviour, or direct source-to-result shortcut remains available. □

Theorem 657.26 (ContinuationBigStep source trace lock)**.** *For every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet*

$$B = (S, T, R, Q, H, C, P, N),$$

the source row S and finite trace ledger T determine the admissible replay route seen by any terminal-read consumer. The route passes through the displayed BHist *trace rows, componentwise* hsame *transports H, replay* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *row N before the terminal row R and witness Q can be consumed.*

No accepted consumer may replace the trace ledger by a host evaluator, add an undisplayed terminal route, or read a terminal witness outside the displayed S, T, R, Q, H, C, P, N inventory.

Proof. Project the accepted packet with Definition 657.1. This exposes the source row, trace ledger, terminal row, terminal-read witness, transport rows, replay route, provenance row, and local name as the complete BHist-level inventory.

Apply the small-step factorization theorem. It routes every computation read from S through the finite trace ledger T and the displayed replay row C before the terminal row R and witness Q are read.

Use terminal-read determinacy to fix the endpoint of this route. Since the terminal read is determined by the displayed replay, the consumer cannot choose a separate terminal row or witness once S and T have been fixed.

Finally apply consumer exactness, ledger totality, and non-escape. These results place H, C, P, N on the same displayed route and rule out host evaluators, hidden terminal rows, quotient behaviour, and direct source-to-result shortcuts. □

Theorem 657.27 (ContinuationBigStep terminal source exhaustion)**.** *For every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet $B = (S, T, R, Q, H, C, P, N)$, the terminal row R and terminal-read witness Q are exhausted by the same finite trace that starts at S and is recorded in T. A terminal consumer may read R and Q only after the displayed* BHist *trace,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *row have been displayed.*

Thus R and Q cannot be replaced by a host evaluator result, hidden termination principle, quotient behaviour class, or direct source-to-terminal shortcut.

Proof. Start with the source and trace rows. Carrier admission and the local NameCert obligations bind S and T to the same finite big-step packet as R, Q, H, C, P, N.

Use the small-step factorization to replay the finite trace. The replay reaches the terminal row through T and C, so the terminal row is a displayed endpoint of the trace rather than a host-level evaluation result.

Apply terminal-read determinacy to the witness. Once the displayed replay reaches R, the witness Q is determined by that terminal-read route and has no separate source.

The non-escape and ledger-totality theorems close the exhaustion claim. Since all consumer rows have already been accounted for by S, T, R, Q, H, C, P, N, no host evaluator, hidden termination principle, quotient behaviour class, or direct shortcut remains. □

Theorem 657.28 (ContinuationBigStep ledger factorization)**.** *Every consumer of the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigSt}}$ *row factors through the finite ledger*

$$S, T, R, Q, H, C, P, N.$$

More explicitly, the consumer first reads the source row and trace ledger, then the small-step replay through the displayed Cont *row, then the terminal row and witness, and finally the transport, provenance, and local naming rows that certify the route. This factorization is exactly the route supplied by the small-step factorization, terminal-read determinacy, consumer exactness, and ledger-totality theorems.*

No accepted local-name consumer receives a host big-step evaluator, hidden terminal source, quotient behaviour class, or undisplayed replay route.

Proof. Read the local naming row through the carrier. The row N certifies the displayed packet S, T, R, Q, H, C, P, N and does not introduce an additional computational coordinate.

Apply the small-step factorization. It supplies the operational route from S through T and C before R and Q are available.

Next use terminal-read determinacy and consumer exactness. These theorems force every terminal consumer, including one entering through the local name, to use the same ordered trace, replay, terminal, transport, provenance, and naming rows.

Finally invoke ledger totality and non-escape. Every displayed consumer route has now been assigned to S, T, R, Q, H, C, P, N, so the local NameCert row cannot certify a host big-step evaluator, hidden terminal source, quotient behaviour class, or undisplayed replay route. □

Theorem 657.29 (ContinuationBigStep root-unblock finite surface)**.** *For every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet* $B = (S, T, R, Q, H, C, P, N)$*, the downstream terminal-read surface is exactly the finite root-unblock inventory consisting of the* BHist *carrier rows, the componentwise* hsame *rows in* H*, the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$ *row* N*. Every downstream consumer of the terminal read must obtain its data from this displayed inventory, with* T *as the finite small-step ledger,* R *as the reached terminal row, and* Q *as the terminal-read witness.*

Proof. Project the accepted packet through Definition 657.1. The projection exposes only S, T, R, Q, H, C, P, N, with T a finite trace ledger, H the componentwise hsame transport table, C the displayed continuation replay, P the package provenance, and N the local naming row.

Apply Definition 657.3. It routes the computation read from the source row through the trace ledger and replay route before the terminal row and witness are consumed.

Use Definition 657.7 and Definition 657.6. These results prevent a terminal-read consumer from bypassing the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows or importing a host evaluator.

Finally invoke Definition 657.25. Since the totality ledger exhausts every consumer route by S, T, R, Q, H, C, P, N, the root-unblock surface is precisely the finite inventory displayed above. □

Theorem 657.30 (ContinuationBigStep root trace-route lock)**.** *Let* $B = (S, T, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet. Any downstream read of the terminal witness* Q *is locked to the finite route*

$$S \longrightarrow T \longrightarrow C \longrightarrow R \longrightarrow Q.$$

No consumer may read Q *directly from* S*, replace* T *by a host evaluator, or bypass the replay row* C *and terminal row* R*.*

Proof. Begin with carrier admission from Definition 657.1. The terminal witness Q is a displayed coordinate of the same finite packet as the source row S, trace ledger T, replay row C, and terminal row R.

Definition 657.3 gives the route from S through T and C to R and then to Q. Thus the terminal witness is read after the replayed trace, not as an independent source-to-witness edge.

Consumer exactness from Definition 657.7 fixes this order for downstream reads. The non-escape boundary of Definition 657.6 excludes host evaluator shortcuts, hidden termination rows, and quotient behaviour classes.

The ledger-totality theorem Definition 657.25 then rules out any remaining route. Every row available to a consumer has already appeared in the displayed chain or in the transport, provenance, and naming rows that authenticate that chain. □

Theorem 657.31 (ContinuationBigStep root terminal ledger exactness)**.** *For every accepted* $\mathsf{ContinuationBigStep}^{\uparrow}$ *packet* $B = (S, T, R, Q, H, C, P, N)$*, the terminal ledger is exhausted by the displayed trace, replay, terminal, transport, provenance, and naming rows. Concretely, trace data are read from* T*, replay data from* C*, terminal data from* R, Q*, componentwise transport from* H*, package provenance from* P*, and local naming from* N*; no additional root row contributes terminal-read authority.*

Proof. Read the carrier definition first. It separates the ledger into the trace row T, the replay row C, the terminal rows R, Q, transport H, provenance P, and local naming N, all over the source row S.

The small-step factorization theorem assigns the operational route to T, C, R, Q. The consumer-exactness theorem assigns downstream terminal reads to the same ordered route.

The non-escape theorem removes every undisplayed source of terminal-read authority. In particular, a host evaluator, hidden termination principle, quotient behaviour class, or direct shortcut is not a coordinate of the packet.

By Definition 657.25, all consumer routes are already exhausted by S, T, R, Q, H, C, P, N. Therefore the terminal ledger is exact: the displayed trace, replay, terminal, transport, provenance, and naming rows are all of it. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuationBigStep}^{\uparrow}$ packet is generated from a source row, a finite small-step trace ledger, the terminal row reached by replay, a terminal-read witness, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 657.1, Definition 657.2, Definition 657.3, Definition 657.5, Definition 657.6, Definition 657.7, Definition 657.14, Definition 657.15, Definition 657.16, Definition 657.17, Definition 657.18, Definition 657.19, Definition 657.20, Definition 657.21, and Definition 657.22 bind the carrier, NameCert obligations, small-step factorization, terminal-read determinacy, non-escape boundary, consumer exactness, subject-reduction socket handoff, terminal ledger package, trace-ledger threshold, terminal consumer package, replay-route stability, terminal-read coverage, NameCert consumer interface, and non-escape scope package as the exported public-object boundary for the finite big-step packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuationBigStep}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ContinuationBigStepUp.ContinuationBigStepPacket\_small\_step\_factorization`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host evaluator semantics, quotient behaviour, hidden termination, Lean verification, or bridge checking.

Upgrade path: The finite small-step factorization, terminal-route exhaustion, consumer exactness, trace-row admission, terminal-row readback, and downstream continuation-trace dependency give the mature paper-side treatment; bridge checking remains a separate verification-axis obligation.

Chapter 658

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFusion}^{\uparrow}$

The $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ packet records the finite cofinal tail-fusion step used by Real completeness consumers before a diagonal $\mathsf{Real}^{\uparrow}$ seal is read. It joins regular-sequence tail witnesses, selected finite stream windows, dyadic tolerance ledgers, a shared tail-meet row, a diagonal seal route, and the local naming certificate into one displayed BHist surface. It exports no quotient stream, host real equality, selected completed limit, unbounded diagonal object, or ambient completeness theorem.

Falsifiable prediction: Deleting the shared-threshold row M permits a $\mathsf{Real}^{\uparrow}$ seal route to bypass the displayed $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ handoff, while deleting the finite window row W breaks the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ route to the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Independence witness: Nearest siblings: regular-cauchy-tail-meet, regularcauchylimittransport, streamname, dyadicratcore, and real. The $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ packet is not bijective to any one of them because it stores the fused route $Q, X, W, D, T, M, R, L, H, C, P, N$ linking all five surfaces before the $\mathsf{Real}^{\uparrow}$ seal.

Definition 658.1 (Regular Cauchy tail fusion carrier)**.** *A* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the rational or dyadic precision request routed to the tail-fusion consumer;*
2. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational family row whose tail witnesses are being fused;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected for the precision request;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation ledger for the selected windows;*
5. *T is the finite tail-witness row exposing the cofinal tail data read from X through W and D;*
6. *M is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row that makes the displayed tail witnesses use one cofinal tail bound;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row reached after the shared tail bound and dyadic ledger are fixed;*
8. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, X, W, D, T, M, R have been exposed;*

9. *H stores componentwise* hsame *transports for the precision request, regular family, windows, dyadic ledger, tail witnesses, tail-meet row, readback row, seal row, provenance, and local name;*
10. *C stores the displayed* Cont *routes from precision request to window selection, dyadic observation, tail witness, shared tail meet, diagonal readback, and Real seal;*
11. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFusion}^{\uparrow}}$ *row.*

The carrier is only the finite cofinal tail-fusion packet. It has no coordinate for quotient stream equality, host real equality, an externally selected limit, an unbounded diagonal stream, or an ambient completeness principle.

Definition 658.2 (Regular Cauchy tail fusion classifier)**.** *Two accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *carriers classify together when their displayed precision requests, regular-family rows, finite windows, dyadic ledgers, tail witnesses, shared tail-meet rows, diagonal readbacks, and Real seal rows are related componentwise by the stored* hsame *transports and replay the same* Cont *route from Q through M to L.*

The classifier compares only the rows listed in Definition 658.1; it does not compare quotient streams, host real values, selected external limits, or ambient completeness witnesses.

Theorem 658.3 (Regular Cauchy tail fusion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFusion}^{\uparrow}}$ *surface for the carrier of Definition 658.1 consists exactly of carrier admission, precision-request exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *family admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger readback, finite tail-witness exposure,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold routing, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Project the carrier F. Its coordinates list the precision request, regular family, selected windows, dyadic ledger, tail witnesses, shared tail meet, diagonal readback, Real seal, transports, routes, provenance, and local name.

The cofinal tail route is finite. A consumer first reads Q, then the selected windows W and dyadic ledger D, then the tail witness row T, and only then the shared-threshold row M used by the diagonal readback R.

The remaining rows H, C, P, N transport, route, record provenance, and name the same packet. Since no carrier coordinate contains quotient streams, host real equality, selected limits, unbounded diagonal data, or ambient completeness, the listed obligations exhaust the local naming surface. □

Theorem 658.4 (Regular Cauchy tail fusion tail-meet handoff)**.** *For an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *carrier, every cofinal tail witness read reaches the diagonal readback row R through the finite route*

$$Q, X, W, D, T, M, H, C, P, N.$$

The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row M is the only shared-threshold surface available to the packet, so the handoff cannot replace it by a hidden tail oracle, quotient-stream comparison, selected limit, host real equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFusionUp.RegularCauchyTailFusionCarrier tail meet handoff`

Proof. Read the carrier coordinates from Definition 658.1. The only row that can merge tail witnesses into a common bound is M, and it appears after the precision request, regular-family row, finite windows, dyadic ledger, and tail-witness row.

Apply the NameCert obligation theorem. It binds the tail-witness exposure to T and the shared-threshold routing to M, with all movement recorded by H and C and all provenance recorded by P and N.

Because the classifier of Definition 658.2 compares only the displayed rows, no transported carrier can introduce a second tail source or an external comparison. The diagonal readback therefore receives the tail-meet handoff through exactly the finite route shown above. □

Theorem 658.5 (Regular Cauchy tail fusion monotone window handoff)**.** *Every accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet reads its selected* $\mathsf{StreamName}^{\uparrow}$ *finite windows W and dyadic ledger D before the tail witnesses T and shared threshold M are consumed. Later* $\mathsf{Real}^{\uparrow}$ *seal reads through L cannot skip W, D or replace them by an unwindowed tail oracle, quotient stream comparison, selected limit, or ambient completeness row.*

Proof. Project

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N).$$

The carrier orders the finite windows and dyadic ledger before tail witnesses and before the shared threshold row.

Use Definition 658.4. It routes every cofinal tail witness through $Q, X, W, D, T, M, H, C, P, N$ before the diagonal readback R is reached.

Now apply the limit-transport link to the later seal row. Since the $\mathsf{Real}^{\uparrow}$ handoff reads L only after the displayed diagonal readback route, coordinate exhaustion leaves no route that bypasses W, D. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFusionUp.RegularCauchyTailFusionCarrier window monotone handoff`

Theorem 658.6 (Regular Cauchy tail fusion shared-threshold exhaustion)**.** *Every* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *route that reaches the diagonal readback R or the* $\mathsf{Real}^{\uparrow}$ *seal L uses the single displayed shared-threshold row M supplied by the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *sibling. The packet has no second tail oracle, quotient stream comparison, selected limit, ambient completeness row, or independent threshold coordinate.*

Proof. Project

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N).$$

The only shared-threshold coordinate in the carrier is M.

Apply Definition 658.4. It sends each cofinal tail witness to R through M, with all movement recorded by the displayed transport, continuation, provenance, and local naming rows.

Use the sibling link to chapter 560 to identify M as the shared finite threshold surface. Use the link to chapter 600 for the subsequent route from R to L.

Coordinate exhaustion closes the claim. Since no other carrier coordinate has the role of a threshold, tail oracle, quotient comparison, selected limit, or completeness witness, every route to R or L uses the displayed M row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFusionUp.RegularCauchyTailFusionCarrier shared threshold exhaustion`

Theorem 658.7 (Regular Cauchy tail fusion Real limit handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *diagonal-limit consumer of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet reads the seal row* L *only after the precision request, regular rational family, selected finite windows, dyadic tolerance ledger, finite tail witnesses, shared tail-meet row, diagonal readback, componentwise transports, continuation routes, package provenance, and local name have been displayed.*

Equivalently, the completion-facing route factors through

$$Q, X, W, D, T, M, R, H, C, P, N$$

before the seal row L *is consumed, and it exports no quotient stream, host real equality, selected completed limit, unbounded diagonal object, or ambient completeness theorem.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ consumer of the seal row L. In the carrier, L is downstream of the diagonal readback row R, so the consumer cannot inspect a seal before the finite readback route has been displayed.

Use Definition 658.4. It places every cofinal tail read through $Q, X, W, D, T, M, H, C, P, N$ before R is reached.

Now append the seal route from R to L using the continuation rows in C and the componentwise transports in H. The package provenance and local name record the same finite route. Since the carrier and classifier omit quotient streams, host equality, selected limits, unbounded diagonal objects, and ambient completeness, no completion-facing read can escape the displayed packet. □

Theorem 658.8 (Regular Cauchy tail fusion limit-transport link)**.** *Every accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors its* $\mathsf{Real}^{\uparrow}$ *seal handoff through the shared-threshold row of chapter 560 and the displayed seal transport surface of chapter 600 before any* $\mathsf{Real}^{\uparrow}$ *consumer can read the diagonal limit row.*

In particular, for

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N),$$

the rows Q, X, W, D, T *expose the finite precision, regular rational family, selected windows, dyadic ledger, and tail witnesses;* M *supplies the shared cofinal threshold;* R, L *supply the diagonal readback and seal handoff; and* H, C, P, N *record transport, continuation, provenance, and the local name.*

Proof. Project the existing packet rows $Q, X, W, D, T, M, R, L, H, C, P, N$. The carrier gives no direct route from the finite tail witnesses to the Real seal; the route must first pass through the displayed shared-threshold row M and the diagonal readback row R.

Use the tail-meet sibling chapter 560. Its shared cofinal threshold is exactly the row consumed at M, so the tail-fusion packet cannot read its finite witnesses as independent unmerged tails.

Now use the limit-transport sibling chapter 600. Its displayed seal transport surface is the finite route that carries the diagonal readback toward the Real-facing seal row, so L is reached only after M and R have been exposed.

Finally apply the Real limit handoff theorem above. Since the carrier omits quotient streams, host real equality, selected limits, unbounded diagonal data, and ambient completeness, the sibling-linked route has no non-displayed escape before the $\mathsf{Real}^{\uparrow}$ consumer reads the diagonal limit row. □

Theorem 658.9 (Regular Cauchy tail fusion public obligation handoff)**.** *For every accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet*

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N),$$

the carrier rows, classifier comparison, tail-meet handoff, monotone window handoff, shared-threshold exhaustion, $\mathsf{Real}^{\uparrow}$ *limit handoff, and limit-transport link jointly form the public obligation surface for downstream Real and Completion consumers. Every such consumer receives those rows as one finite handoff and receives no additional source beyond the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Proof. Project the carrier and classifier. The public surface begins with the displayed coordinates $Q, X, W, D, T, M, R, L, H, C, P, N$, and classifier stability compares only those coordinates.

Use the tail-meet handoff, monotone window handoff, and shared-threshold exhaustion. These rows place every cofinal tail read behind the selected windows, dyadic ledger, finite witnesses, and the shared threshold M supplied by chapter 560.

Use the Real limit handoff and the limit-transport link. They carry the diagonal readback through R and the seal row L along the displayed transport surface of chapter 600.

Repack the cited rows as the public obligation handoff. Since no other coordinate is available in the packet, every Real or Completion consumer receives exactly that finite surface. □

Theorem 658.10 (Regular Cauchy tail fusion window budget coverage)**.** *Every accepted window read of a* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors through the finite window and budget route*

$$W, D, T, M$$

before the diagonal readback and Real seal rows R, L *are consumed. The row* W *is read as the finite-window surface of chapter 126,* D *is read as the dyadic tolerance ledger of chapter 291,* T *records the displayed tail witnesses, and* M *is the shared-threshold row.*

Proof. Start from an accepted window read. Projection from the carrier shows that the only window coordinate is W, and the only budget coordinate attached to it is D.

Apply the StreamName finite-window rows, represented here by the W coordinate, and then apply the dyadic ledger rows represented by D. These two rows supply the bounded window and tolerance information before any tail witness can be used.

Use the monotone window handoff and shared-threshold exhaustion. They force the read through T and then through M before the diagonal readback row R and the seal row L can be consumed.

Thus every accepted window read factors through W, D, T, M first, and the downstream rows R, L receive only the finite route already covered by those rows. □

Theorem 658.11 (Regular Cauchy tail fusion Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of a* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet receives only the displayed route*

$$Q, X, W, D, T, M, R, L, H, C, P, N.$$

The handoff exposes no quotient stream, host real equality, selected limit, unbounded diagonal object, hidden tail oracle, or ambient completeness principle.

Proof. Begin with the public obligation handoff above. It places every downstream read on the displayed carrier, classifier, tail-meet, window, threshold, Real handoff, and limit-transport rows.

Apply window budget coverage. The consumer reaches R and L only after the route through W, D, T, M has been exposed.

Finally use coordinate exhaustion. The carrier has no quotient stream coordinate, no host real equality coordinate, no selected-limit coordinate, no unbounded diagonal object, no hidden tail oracle, and no ambient completeness row. Therefore a Real or Completion consumer cannot escape the displayed finite route. □

Theorem 658.12 (Regular Cauchy tail fusion scoped window exhaustion)**.** *Every scoped completion-facing read of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors through the finite route*

$$Q, X, W, D, T, M, R, H, C, P, N$$

before the seal row L is consumed. Here W, D are the $\mathsf{StreamName}^{\uparrow}$ *window row and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, M is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row, and R is the* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$*-facing readback. The read exports no quotient stream, host real equality, selected completed limit, unbounded diagonal data, or ambient completeness principle.*

Proof. Project the carrier

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N).$$

The completion-facing seal coordinate L is downstream of the displayed diagonal readback row R, and the readback row is downstream of the selected finite windows, dyadic ledger, tail witnesses, and shared threshold.

Apply the tail-meet handoff and shared-threshold exhaustion. These results force the cofinal tail data through $Q, X, W, D, T, M, H, C, P, N$ and identify M with the displayed $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ sibling row rather than with an external tail oracle.

Apply the limit-transport link and the Real limit handoff. The transport sibling consumes the diagonal readback surface at R, and the Real handoff permits the seal read only after the finite route has been displayed.

Coordinate exhaustion closes the scoped claim. Since the carrier and classifier contain no quotient stream, host real equality, selected limit, unbounded diagonal coordinate, or ambient completeness row, the completion-facing read has no route outside the displayed finite window path before L. □

Theorem 658.13 (Regular Cauchy tail fusion obligation surface)**.** *The carrier, classifier, local* $\mathsf{NameCert}$ *obligations, tail-meet handoff, window monotonicity, shared-threshold exhaustion, Real limit handoff, and limit-transport link of this chapter form the five obligation rows for* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*. The rows are carrier admission, classifier stability, cofinal-tail routing, downstream Real seal handoff, and ledger non-escape, and every route is confined to* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission and classifier stability are exactly the surfaces of Definition 658.1 and Definition 658.2. The local naming row is supplied by Definition 658.3.

The cofinal-tail route is assembled from Definition 658.4, Definition 658.5, and Definition 658.6. These results force every tail witness through the displayed windows, dyadic ledger, and the sibling shared-threshold packet of chapter 560.

The downstream Real seal handoff is supplied by Definition 658.7 and Definition 658.8. The latter also routes the diagonal readback through the seal-transport sibling chapter 600.

Ledger non-escape follows by collecting the exclusions in those displayed results. The carrier and classifier expose only the listed BEDC rows, so no route can introduce a quotient stream, host real equality, selected completed limit, unbounded diagonal object, independent tail oracle, or ambient completeness theorem. □

Theorem 658.14 (Regular Cauchy tail fusion carrier-classifier synchronization)**.** *Every classifier comparison between accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packets exposes the same carrier rows*

$$Q, X, W, D, T, M, R, L, H, C, P, N.$$

The comparison is therefore synchronized with the displayed precision request, regular-family row, finite window, dyadic ledger, tail witnesses, shared-threshold row, diagonal readback, Real seal, componentwise transports, continuation routes, provenance, and local name. It cannot compare quotient streams, host real values, selected completed limits, unbounded diagonal objects, or ambient completeness witnesses.

Proof. Project the two accepted packets through Definition 658.1. Each accepted carrier has exactly the rows $Q, X, W, D, T, M, R, L, H, C, P, N$, with no additional completion-facing coordinate.

Apply the classifier definition of Definition 658.2. It compares the listed rows componentwise and replays the same displayed route from Q through M to L.

Use the tail-meet handoff, shared-threshold exhaustion, and limit-transport link. These results force the compared route through the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ sibling at M and the $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ sibling before the seal row L can be read.

Coordinate exhaustion gives synchronization. Since neither the carrier nor the classifier has a quotient-stream, host-real, selected-limit, unbounded-diagonal, or ambient-completeness coordinate, the comparison cannot leave the displayed rows. □

Theorem 658.15 (Regular Cauchy tail fusion window transport lock)**.** *For every accepted* $\mathsf{RegularCauchyTailFusion}$ *transport, the* hsame *rows preserve the selected* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row* M *before the* $\mathsf{Real}^{\uparrow}$ *seal row* L *is consumed. A transported seal read cannot change the window, budget ledger, or shared threshold on the way to* L*.*

Proof. Start from the carrier inventory $F = (Q, X, W, D, T, M, R, L, H, C, P, N)$. The rows W, D, M are displayed before the diagonal readback R and the seal row L.

Apply the monotone window handoff and shared-threshold exhaustion. They place W and D before the tail witnesses and identify M as the single sibling threshold supplied by chapter 560.

Now apply the carrier-classifier synchronization theorem above. Its componentwise comparison uses H only to transport the displayed rows, so W, D, M remain the same selected window, ledger, and threshold rows along the accepted route.

Finally use the limit-transport link to chapter 600. The seal-facing transport begins after the diagonal readback has consumed W, D, M, so changing any of them before L exits the accepted packet. □

Theorem 658.16 (Regular Cauchy tail fusion ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer route of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet is exhausted by the finite window, shared-threshold, diagonal readback, transport, continuation, package, and local NameCert rows*

$$W, D, T, M, R, L, H, C, P, N.$$

The route has no remaining ledger entry for a quotient stream, selected completed limit, host real equality, hidden tail oracle, independent transport, or ambient completeness principle.

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing read. By the Real limit handoff, the seal row L is read only after the precision request, regular-family row, finite window, dyadic ledger, tail witnesses, shared threshold, and diagonal readback have already been displayed.

Use the tail-meet handoff and shared-threshold exhaustion. They route the finite tail witnesses through T and the sibling threshold M, leaving no hidden tail oracle or independent threshold source.

Apply the window transport lock. It preserves W, D, M through the componentwise hsame rows and keeps the route aligned with the displayed continuation and provenance rows before L.

Finally collect the public export packet and limit-transport link. These assemble the Real-facing route from the finite ledger rows $W, D, T, M, R, L, H, C, P, N$, so quotient streams, selected limits, host equality, independent transport, and ambient completeness are outside the local certificate. □

Theorem 658.17 (Regular Cauchy tail fusion public export packet)**.** *The carrier, classifier, obligation surface, shared-threshold exhaustion,* Real$^{\uparrow}$ *limit handoff, and limit-transport link for* RegularCauchyTailFusion$^{\uparrow}$ *assemble one public export packet over* RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyTailMeet$^{\uparrow}$, RegularCauchyLimitTransport$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *A public consumer may read exactly the finite route*

$$Q, X, W, D, T, M, R, L, H, C, P, N$$

with M as the shared-threshold row, R as the diagonal readback row, and L as the Real$^{\uparrow}$ *seal handoff.*

The packet exports no quotient streams, host real equality, selected completed limits, unbounded diagonal data, hidden tail oracle, or ambient completeness theorem.

Proof. Carrier admission and classifier stability are supplied by Definition 658.1 and Definition 658.2. The local obligation surface is Definition 658.13, so the public object starts from the same finite rows rather than from a host completion object.

The tail side is controlled by Definition 658.4, Definition 658.5, and Definition 658.6. These theorems force every tail witness through the selected windows, dyadic ledger, finite witness row, and the displayed RegularCauchyTailMeet$^{\uparrow}$ threshold row M.

The completion-facing side is controlled by Definition 658.7 and Definition 658.8. They route the diagonal readback R and the seal row L through the same hsame, Cont, Pkg, and NameCert boundary, with the checked TasteGate marker recorded in the closure block below identifying the same carrier surface.

Repack these rows as the public export packet. Since each cited theorem excludes quotient streams, host real equality, selected limits, unbounded diagonal data, hidden tail oracles, and ambient completeness, the public export exposes only the displayed finite route. □

Theorem 658.18 (Regular Cauchy tail fusion selector-budget window lock)**.** *Every selector-budget read through an accepted* RegularCauchyTailFusion$^{\uparrow}$ *public export packet must pass through the displayed* StreamName$^{\uparrow}$ *window row W and* DyadicRatCore$^{\uparrow}$ *ledger row D before it reaches the shared* RegularCauchyTailMeet$^{\uparrow}$ *row M. The accepted read has no route to an unbounded selector, hidden budget oracle, independent tail threshold, quotient stream, selected limit, host convergence row, or ambient completeness row outside the finite packet*

$$Q, X, W, D, T, M, R, L, H, C, P, N.$$

Proof. Start with the public export packet of Definition 658.17. It exposes exactly the finite rows $Q, X, W, D, T, M, R, L, H, C, P, N$, with W and D displayed before the tail-meet row M.

Apply Definition 658.5 and Definition 658.6. These rows force any selector-budget request to read the scheduled StreamName$^{\uparrow}$ window and DyadicRatCore$^{\uparrow}$ ledger before the RegularCauchyTailMeet$^{\uparrow}$ sibling named in chapter 560.

Finally use Definition 658.8 and the limit-transport sibling chapter 600. Since the transport begins after the same window, ledger, and shared-threshold rows have been displayed, no selector or budget read can bypass W, D, M on the way to the Real-facing rows. □

Theorem 658.19 (Regular Cauchy tail fusion cofinal route totality)**.** *Every cofinal route to the diagonal readback of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet is exhausted by the displayed rows*

$$Q, X, W, D, T, M, R, H, C, P, N.$$

The route reaches R only through the precision request, regular source, finite window, dyadic ledger, tail witnesses, shared tail-meet row, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local* $\mathsf{NameCert}$ *row; it has no further cofinal selector, hidden threshold, quotient stream, host limit, or ambient completeness coordinate.*

Proof. Project the carrier through the public export packet. Before the seal row is read, the available diagonal route is the finite predecessor $Q, X, W, D, T, M, R, H, C, P, N$.

Use Definition 658.4 and Definition 658.6. They place every cofinal witness behind the selected window, dyadic ledger, finite tail row, and the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ sibling row M.

Apply Definition 658.8 and the sibling rows for chapter 126, chapter 291, chapter 560, and chapter 600. These rows account for the diagonal readback and its transports, so the listed route exhausts every cofinal path to R. □

Theorem 658.20 (Regular Cauchy tail fusion Real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors through the same shared-threshold row M and the same* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *sibling before it reads the seal row L. The consumer receives the seal only through*

$$Q, X, W, D, T, M, R, L, H, C, P, N,$$

with M supplied by chapter 560 and the transport surface supplied by chapter 600.

Proof. Begin with Definition 658.7 and Definition 658.17. Together they place the seal row L after the diagonal readback row R and keep both rows inside the displayed finite packet.

Next apply Definition 658.6. The route to R must already have passed through the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row M, so the seal consumer cannot read a separate threshold or hidden tail source.

Finally use Definition 658.8. It identifies the seal-facing transport as the displayed $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ sibling route, after which the public export exclusions rule out quotient streams, selected limits, host convergence, ambient completeness, and independent Real-seal paths. □

Theorem 658.21 (Regular Cauchy tail fusion shared-source readback)**.** *Every accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet*

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N)$$

reads its source rows Q, X, W, D, T, M, R through one shared finite route before the $\mathsf{Real}^{\uparrow}$ *seal row L is consumed. The route starts with the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row D, and the regular-family source row X, then passes through the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold M and the diagonal readback row R. No downstream consumer may split this read into an independent stream name, dyadic ledger, tail threshold, or diagonal readback source.*

Proof. Project the displayed carrier. Its only source-side coordinates before the seal row are Q, X, W, D, T, M, R, with H, C, P, N recording transport, continuation, provenance, and local naming data rather than separate source objects.

Use the tail-meet handoff and shared-threshold exhaustion. They identify M as the sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold row reached from the same X, W, D, T source route, so the tail evidence cannot be read as a hidden or independently selected cofinal tail.

Apply window monotonicity and the limit-transport link. The window and dyadic rows remain the finite predecessor of the diagonal readback row R, and the transport row carries that readback toward the seal without introducing another source coordinate.

Finally use the public export packet. Since the public packet exposes exactly $Q, X, W, D, T, M, R, L, H, C, P, N$, every accepted consumer reads the source rows through the same finite route before L. □

Theorem 658.22 (Regular Cauchy tail fusion Real seal consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of the seal row L in an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors through the shared-source readback of Definition 658.21 and then through the displayed rows H, C, P, N. The boundary exposes no quotient stream, host equality, selected limit, hidden tail oracle, ambient completeness principle, or independent route around Q, X, W, D, T, M, R.*

Proof. Start with a consumer request for the seal row L. By shared-source readback, the request first reaches the rows Q, X, W, D, T, M, R through one $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegularCauchyTailMeet}^{\uparrow}$ route.

Apply the Real limit handoff. The handoff reads L only after the diagonal readback row R has been exposed, and it treats L as a seal row rather than as a selected completed limit or ambient completeness witness.

Use the limit-transport link to move from R to the seal-facing surface. The transport is recorded by the displayed H and C rows, with P and N carrying provenance and local naming for the same finite packet.

Finally apply the bridged consumer packet. It leaves every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer behind the public export surface and the displayed rows H, C, P, N, so no quotient stream, host equality, selected limit, hidden tail oracle, ambient completeness principle, or independent route remains available. □

Theorem 658.23 (Regular Cauchy tail fusion bridged consumer packet)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet*

$$F = (Q, X, W, D, T, M, R, L, H, C, P, N)$$

factors through the shared-threshold row of chapter 560, the displayed seal-transport surface of chapter 600, and the $\mathsf{Real}^{\uparrow}$ *seal row before any public completion read. The packet supplies no quotient stream, host real equality, selected completed limit, unbounded diagonal object, ambient completeness theorem, hidden tail oracle, or independent transport route.*

Proof. Begin with the displayed carrier and public export packet. The carrier lists the finite route $Q, X, W, D, T, M, R, L, H, C, P, N$, and Definition 658.17 permits a public consumer to read only that route.

Use Definition 658.4 and Definition 658.6. They force the tail witnesses through T and then through the single shared-threshold row M supplied by the tail-meet sibling.

Use Definition 658.8. It carries the diagonal readback surface from R toward the displayed seal-transport sibling before the seal row L can be read.

Apply Definition 658.12 and the Real limit handoff. These keep the completion-facing read behind the selected windows, dyadic ledger, shared threshold, diagonal readback, and seal row.

Thus any completion-facing consumer factors through the tail-meet row, the limit-transport surface, and the Real seal before becoming public. The exclusions in the carrier, classifier, scoped window exhaustion, and public export packet rule out quotient streams, host real equality, selected limits, unbounded diagonal objects, ambient completeness, hidden tail oracles, and independent transport routes. □

Theorem 658.24 (Regular Cauchy tail fusion mature consumer surface)**.** *Every mature completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet factors through the same public export packet of Definition 658.17, the bridged consumer packet of Definition 658.23, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *sibling, the* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *sibling, and the* $\mathsf{Real}^{\uparrow}$ *seal rows. The mature surface exports no quotient stream, selected limit, ambient completeness row, hidden tail oracle, or independent transport route.*

Proof. Start with the public export packet. It exposes only the finite route $Q, X, W, D, T, M, R, L, H, C, P, N$ and denies quotient streams, host real equality, selected completed limits, unbounded diagonal data, hidden tail oracles, and ambient completeness.

Apply the bridged consumer packet. A completion-facing consumer must pass through the shared-threshold row supplied by the tail-meet sibling and the seal-transport surface supplied by the limit-transport sibling before it can read the $\mathsf{Real}^{\uparrow}$ seal row.

Finally collect the sibling and seal exclusions. Since the tail meet supplies only the shared finite threshold, the transport sibling supplies only the displayed seal route, and the Real row is read as a seal rather than as an ambient completeness theorem, every mature consumer factors through the same finite consumer surface. □

The Lean TasteGate namespace BEDC.Derived.RegularCauchyTailFusionUp.TasteGate must provide Nontrivial RegularCauchyTailFusionUp and FieldFaithful RegularCauchyTailFusionUp instances for the displayed carrier.

Theorem 658.25 (Regular Cauchy tail fusion paired agreement source)**.** *Every accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet exposing*

$$Q, X, W, D, T, M, R, L, H, C, P, N$$

supplies the single-source cofinal-tail input that a paired $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer may duplicate into its left and right source rows. This source is the finite route certified by the public export packet and the mature consumer surface, and it does not by itself export paired source equality, quotient stream equality, selected limit equality, host convergence, or ambient completeness.*

Proof. Begin with the public export packet of Definition 658.17. It exposes the finite route $Q, X, W, D, T, M, R, L, H, C, P, N$, with X as the regular-family source row and M as the shared cofinal-tail row.

Apply the bridged consumer packet of Definition 658.23 and the mature consumer surface of Definition 658.24. Together they force any Real-facing consumer to read this route through the public export boundary before reaching the seal-facing rows.

The paired seal of chapter 699 may use that single cofinal-tail source as the source of its left and right rows. The duplication is a consumer-side paired read, not an equality exported by the fusion packet, so the exclusions of the public export packet remain in force. □

Theorem 658.26 (Regular Cauchy tail fusion paired window budget)**.** *The* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *window and budget route*

$$Q, X, W, D, T, M$$

is the finite predecessor of the paired-window budget row consumed by the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *chapter chapter 699. A paired agreement consumer may read this predecessor before forming its paired window row, but it receives no unbounded stream selector, independent budget oracle, quotient stream, selected limit, host convergence proof, or ambient completeness row from the fusion packet.*

Proof. Read the public export packet. Its window and budget side is the finite route from the precision request Q through the source row X, selected windows W, dyadic ledger D, tail witnesses T, and shared threshold M.

Use the bridged consumer packet to place this finite route before any completion-facing read. The consumer cannot reach the later diagonal readback or seal row without first passing through the selected windows, dyadic ledger, tail witnesses, and shared threshold.

The paired agreement seal consumes paired windows as its own local row. Therefore the fusion route is only the predecessor budget surface for that row: it supplies the finite window and threshold discipline that the seal duplicates into a paired-window read, while all non-escape clauses of the fusion packet remain available. □

Theorem 658.27 (Regular Cauchy tail fusion real tail-agreement handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer passing from* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *to* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *factors first through the fusion public export packet of Definition 658.17, then through the bridged and mature consumer surfaces of Definition 658.23 and Definition 658.24, and only then through the paired agreement seal of chapter 699. This handoff introduces no quotient stream, selected limit, host convergence row, ambient completeness row, hidden tail oracle, or independent equality route.*

Proof. First expose the fusion packet publicly. By Definition 658.17, the consumer can read only the finite rows $Q, X, W, D, T, M, R, L, H, C, P, N$ at the fusion boundary.

Next apply the bridged consumer packet and the mature consumer surface. These results force the completion-facing read through the shared-threshold row, the diagonal readback, and the $\mathsf{Real}^{\uparrow}$ seal route before it can be treated as a mature consumer surface.

Finally pass to the paired agreement seal named in chapter 699. That chapter receives the cofinal-tail route as its predecessor and packages the paired agreement rows locally, so the transition remains a factorization through displayed packets rather than an appeal to quotient streams, selected limits, host convergence, ambient completeness, hidden tail oracles, or an independent equality route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFusion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ packet is generated from a precision request, $\mathsf{RegSeqRat}^{\uparrow}$ regular-family row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite tail witnesses, a $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold row, diagonal readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFusionUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFusion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 658.13, Definition 658.17, Definition 658.23, Definition 658.24, Definition 658.25, Definition 658.26, and Definition 658.27 bind the carrier, classifier, local naming row, cofinal-tail routing, downstream Real seal handoff, ledger non-escape, public export boundary, bridge-shaped consumer packet, mature consumer surface, paired agreement source, paired-window budget predecessor, and Real tail-agreement handoff for the finite cofinal tail-fusion packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFusion}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyTailFusionUp.RegularCauchyTailFusionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, external limit selection, unbounded diagonal streams, ambient completeness, Lean verification, or bridge checking.

Upgrade path: The paper closure axis is complete for this packet; Lean verification and bridge checking remain independent.

Chapter 659

A Concrete Naming Certificate for $\mathsf{NontrivialZeroClassifier}^{\uparrow}$

The $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ packet realizes the RH witness-catalogue classifier as a finite boundary object. It separates supplied critical-strip zero rows from trivial-zero rows by displaying the $\mathsf{ZetaZeros}^{\uparrow}$ row, the $\mathsf{CritStrip}^{\uparrow}$ row, the critical-line or critical-strip witness row already available to an RH-facing consumer, the real and rational comparison ledgers, componentwise transports, continuation routes, package provenance, and a local naming row. It is not an RH proof, a zero search procedure, a host complex equality, a quotient zero-set equality, or a selected representative.

The $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ route consumes the public boundary discipline of chapter 595 and the negative-even exclusion route of chapter 999 before exposing an RH-facing nontrivial-zero row. These sibling rows supply the classifier boundary and trivial-zero separation used by downstream RH-facing packets.

Definition 659.1 (Nontrivial-zero-classifier carrier)**.** *A* $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (Z, S, W, T, R, A, H, C, P, N)$$

with the following displayed rows:

1. *Z is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row, carrying the zeta-zero source history and zero-pattern evidence;*
2. *S is the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row for the same displayed source when the row is read as a critical-strip zero;*
3. *W is the RH-facing witness row, read either as a* $\mathsf{CriticalLineWitness}^{\uparrow}$ *packet or as a* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *packet;*
4. *T is the trivial-zero ledger row naming the negative-even-integer source route through* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the carried zeta-zero row;*
5. *R is the real-part row used by the strip and trivial-zero comparisons;*
6. *A is the rational comparison ledger recording the critical-strip interval side and the negative-even side as distinct displayed rows;*
7. *H records componentwise* hsame *transports for Z, S, W, T, R, A and for the package and local naming coordinates;*

8. *C records the displayed* Cont *routes from the zeta-zero row to the strip row, from the trivial-zero row to the real and rational ledgers, and from the witness row to the RH-facing consumer boundary;*
9. *P is the* Pkg *provenance over* $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{CriticalLineWitness}^{\uparrow}$, $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}$ *row.*

The carrier classifies a supplied zero as non-trivial only through the displayed critical-strip row S and the witness row W. It classifies a supplied zero as trivial only through the displayed negative-even ledger T and the rational comparison ledger A.

Lean4 ✓ `BEDC.Derived.NontrivialZeroClassifierUp.NontrivialZeroClassifierTasteGate single carrier alignment`

Theorem 659.2 (NontrivialZeroClassifier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}$ *surface for Definition 659.1 consists exactly of six finite obligations:*

1. *carrier admission for $Q = (Z, S, W, T, R, A, H, C, P, N)$;*
2. *shared-source binding between the* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z and the* $\mathsf{CritStrip}^{\uparrow}$ *row S through the displayed* hsame *and* Cont *routes;*
3. *witness handoff through either* $\mathsf{CriticalLineWitness}^{\uparrow}$ *or* $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ *without adding a proof of* RH*;*
4. *trivial-zero ledger exposure through the negative-even route T and the rational comparison ledger A;*
5. *componentwise classifier stability under the displayed transports in H and replay through the routes in C;*
6. *non-escape from analytic zero search, host complex equality, quotient zero-set equality, selected representatives, and an RH conclusion.*

Lean4 ✓ `BEDC.Derived.NontrivialZeroClassifierUp.NontrivialZeroClassifier namecert obligations`

Proof. Carrier admission is projection from Definition 659.1. The packet exposes exactly the zero row, strip row, witness row, trivial-zero ledger, real-part row, rational comparison ledger, transports, continuations, provenance, and local name.

The shared-source binding is read from H and C. These rows are the only coordinates that compare the source of Z with the source of S, so the classifier never replaces the displayed route by host complex equality.

The witness handoff is read through W. If W is consumed as a $\mathsf{CriticalLineWitness}^{\uparrow}$ packet, it supplies only the displayed critical-line witness surface; if it is consumed as a $\mathsf{CriticalStripZetaZeroWitness}^{\uparrow}$ packet, it supplies only the critical-strip zeta-zero premise surface.

The trivial-zero side is read through T, R, A. Those rows route the negative-even source through the displayed real and rational ledgers, while the non-trivial side remains the supplied strip row S attached to Z.

Finally, P and N package these obligations under the local certificate. Since no carrier coordinate names a zero search, a quotient zero set, a host equality, a representative selector, or an RH proof, the six listed obligations exhaust the available surface. □

Theorem 659.3 (NontrivialZeroClassifier critical-strip handoff)**.** *For an accepted* $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ *carrier* $Q = (Z, S, W, T, R, A, H, C, P, N)$*, every RH-facing non-trivial-zero read factors through the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row* Z*, the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row* S*, the witness row* W*, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* C*, package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}$ *row* N*. The read receives a critical-strip zero boundary and receives no analytic zero search, host complex equality, quotient zero-set equality, selected representative, or proof of* RH*.*

Lean4 ✓ `BEDC.Derived.NontrivialZeroClassifierUp.NontrivialZeroClassifierCarrier critical strip handoff`

Proof. Start with an accepted carrier and project the rows Z, S, W, H, C, P, N. The rows Z and S supply the zero and critical-strip premise, while H and C bind them as one displayed source route.

Read the witness row W only after that premise route is present. The witness is therefore a consumer of the supplied zero-in-strip packet, not a constructor of new zeros and not a proof that all such zeros are on the critical line.

The package and local naming rows close the handoff. No unlisted coordinate can provide host complex equality, a quotient zero-set comparison, a representative selector, or an RH conclusion. □

Theorem 659.4 (NontrivialZeroClassifier trivial-zero exclusion ledger)**.** *For an accepted* $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ *carrier* $Q = (Z, S, W, T, R, A, H, C, P, N)$*, every trivial-zero read factors through the negative-even ledger* T*, the real-part row* R*, the rational comparison ledger* A*, the displayed transports* H*, the displayed continuations* C*, the package row* P*, and the local name* N*. A row classified through* T, R, A *is a trivial-zero row of the packet and cannot simultaneously be read as a critical-strip non-trivial-zero row except through a separately displayed strip row* S *tied to* Z *by* H *and* C*.*

Proof. Project the trivial-zero side of the carrier. The only rows available for such a read are T, R, A, H, C, P, N.

The ledger T supplies the negative-even route, and R, A record the real and rational comparisons used by that route. These rows are distinct from the critical-strip row S and the witness row W.

The transport and continuation rows preserve this separation componentwise. Thus a trivial-zero consumer receives the finite exclusion ledger and cannot obtain a critical-strip non-trivial-zero reading unless the carrier also displays the separate S, Z route. □

Theorem 659.5 (NontrivialZeroClassifier RH consumer boundary)**.** *Every RH-facing consumer of an accepted* $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ *packet receives exactly the finite classifier boundary*

$$Z, S, W, T, R, A, H, C, P, N.$$

It may branch between the critical-strip zero route Z, S, W, H, C, P, N *and the trivial-zero ledger route* T, R, A, H, C, P, N*, but it receives no proof of* RH*, no analytic zero search, no host complex equality, no quotient zero-set equality, and no selected representative of a zero class.*

Proof. Carrier admission displays the ten rows of the packet. The NameCert obligations identify which rows belong to the critical-strip handoff and which rows belong to the trivial-zero ledger.

Use the critical-strip handoff theorem for the Z, S, W, H, C, P, N route. Use the trivial-zero exclusion ledger theorem for the T, R, A, H, C, P, N route.

The two routes share only the displayed transport, continuation, provenance, and naming rows. Since the carrier has no field for RH proof, analytic search, host equality, quotient zero-set

equality, or representative selection, every RH-facing consumer remains inside the stated finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ packet is generated from a supplied $\mathsf{ZetaZeros}^{\uparrow}$ zero row, supplied $\mathsf{CritStrip}^{\uparrow}$ strip row, RH-facing critical-line or critical-strip witness row, trivial-zero ledger, real-part row, rational comparison ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 659.1, Definition 659.2, Definition 659.3, Definition 659.4, and Definition 659.5 bind the finite classifier boundary separating critical-strip zero witnesses from trivial-zero rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NontrivialZeroClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RH, analytic zero search, host complex equality, quotient zero-set equality, selected representatives, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-source binding, witness handoff, trivial-zero ledger exposure, classifier stability, and RH consumer non-escape for the same finite packet.

Chapter 660

A Concrete Naming Certificate for $\mathsf{RegularCauchySequence}^{\uparrow}$

The $\mathsf{RegularCauchySequence}^{\uparrow}$ packet records the finite family data that feeds the Real completeness diagonal-limit handoff before a $\mathsf{Real}^{\uparrow}$ seal is formed. It keeps the regular Cauchy family, explicit modulus, selected finite windows, dyadic readback ledger, $\mathsf{RegSeqRat}^{\uparrow}$ rows, and downstream $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ coverage in one local naming certificate. The packet is a concrete family carrier for the Real and Completion substrate; it is not a quotient regular sequence, a chosen completed real, or an ambient completeness principle.

Definition 660.1 (Regular Cauchy sequence carrier)**.** *A* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite regular Cauchy family row whose members are read as rational regular sequences;*
2. *μ is the explicit modulus row assigning finite tail bounds to the family requests;*
3. *W is the selected-window row, listing the finite* $\mathsf{StreamName}^{\uparrow}$ *windows used for the requested precisions;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback ledger containing the dyadic endpoints, radii, comparisons, and tolerance rows used by the selected windows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that presents each selected window as regular rational sequence data;*
6. *G is the* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *coverage row stating that the selected windows cover exactly the finite diagonal reads consumed by the handoff;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached after F, μ, W, D, R, G have been displayed;*
8. *H records componentwise* hsame *transports for the family, modulus, selected windows, dyadic ledger, regular readbacks, coverage row, handoff row, provenance, and local name;*
9. *C records the displayed* Cont *routes from the family request through the modulus, windows, dyadic ledger, regular readback, diagonal-window coverage, and* $\mathsf{Real}^{\uparrow}$ *handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ *row.*
The carrier contains no quotient equality on regular sequences, no host real equality, no selected external limit, no countable-choice data, and no ambient completeness theorem.

Definition 660.2 (Regular Cauchy sequence classifier)**.** *Two* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carriers*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N) \quad \text{and} \quad S' = (F', \mu', W', D', R', G', E', H', C', P', N')$$

are classified together exactly when their displayed family rows, modulus rows, selected windows, dyadic readback ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *rows,* $\mathsf{Real}^{\uparrow}$ *handoff rows, componentwise transports, continuation routes, package provenance, and local NameCert rows are matched by the corresponding* hsame, Cont, Pkg, *and* NameCert *evidence carried in the packets.*

The classifier is finite and rowwise. It does not identify quotient representatives of regular Cauchy sequences and does not compare completed reals outside the displayed handoff row.

Theorem 660.3 (Regular Cauchy sequence window regularity)**.** *For an accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carrier*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

every selected window in W is regular for the family row F at the precision named by the explicit modulus row μ. Its rational observations are read through the $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D and repacked by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the coverage row G may expose the read to the diagonal-limit handoff.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceUp.RegularCauchySequenceCarrier window regularity`

Proof. Project the carrier through Definition 660.1. The only rows that can name regular-family observations are F, μ, W, D, R, G.

The selected-window row W is reached from the family request only through the displayed continuation route in C and the explicit modulus row μ. Thus a window has no independent precision source.

The rational observations then pass through D and R. The coverage row G can expose only those reads, so regularity at the handoff boundary is the finite rowwise regularity recorded by the carrier. □

Theorem 660.4 (Regular Cauchy sequence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ *surface for Definition 660.1 has five finite obligations:*

1. *carrier admission for $S = (F, \mu, W, D, R, G, E, H, C, P, N)$;*
2. *classifier stability for the family row F, explicit modulus μ, selected windows W, dyadic ledger D, regular readback row R, coverage row G, and* $\mathsf{Real}^{\uparrow}$ *handoff row E;*
3. *selected-window regularity, requiring every window read to use the bound supplied by μ and the rational observations recorded by D and R;*
4. *diagonal-window coverage, requiring the* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *row G to expose exactly the finite selected reads consumed by the handoff;*
5. *package closure, requiring all exported routes to pass through the displayed* hsame, Cont, Pkg, *and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceUp.RegularCauchySequenceCarrier namecert obligations`

Proof. Carrier admission is the tuple assembly of Definition 660.1. The displayed coordinates name the family, modulus, windows, dyadic ledger, readbacks, coverage row, handoff row, transports, continuations, provenance, and local certificate.

Classifier stability is componentwise. Apply the classifier of Definition 660.2 to transport F, μ, W, D, R, G, E through H while preserving the continuation routes and provenance rows C, P, N.

Selected-window regularity follows from Definition 660.3. The modulus row supplies the finite tail bound, and the dyadic and regular readback rows supply the only rational observations visible to a window.

The coverage and package clauses are projections of G, H, C, P, N. Since no other coordinate is present, the certificate exports only the finite selected-window surface needed by the diagonal-limit handoff. □

Theorem 660.5 (Regular Cauchy sequence finite-window exhaustion)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

every $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of the sequence factors through a selected finite window in* W*, the dyadic ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the coverage row* G *before the seal row* E *is exposed. The exported boundary is the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *route of* S*; it exports no quotient stream, selected external limit, countable-choice selector, hidden tail oracle, or ambient completeness theorem.*

Proof. Project the carrier of Definition 660.1. The accepted packet exposes exactly the family row F, modulus row μ, selected windows W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, coverage row G, seal row E, and structural rows H, C, P, N.

Apply Definition 660.3 and then Definition 660.4. Selected-window regularity ties each read to μ, W, D, R, and diagonal coverage requires the $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ row G to expose exactly the finite selected reads consumed by the handoff.

Use the seal ordering of Definition 660.9. A $\mathsf{Real}^{\uparrow}$ consumer can reach E only after the finite window, dyadic, readback, and coverage rows have already been displayed.

Finally close with Definition 660.7. Since the carrier has no coordinate for a quotient stream, selected external limit, choice selector, hidden tail oracle, or ambient completeness theorem, every consumer read is exhausted by the finite route through W, D, R, G and the displayed structural rows. □

Theorem 660.6 (Regular Cauchy sequence finite-window witness induction)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

every finite selected-window subroute admitted by W *is obtained by induction on the displayed finite window spine of* S*. The base route exposes no consumer read before a window row of* W *is displayed. The step adjoining one selected window preserves the route through the modulus* μ*, dyadic ledger* D*, regular readback row* R*, diagonal coverage row* G*, transport row* H*, continuation row* C*, provenance row* P*, and local NameCert row* N *before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is exposed.*

Consequently the witness induction exports only the finite spine already present in S*. It refuses quotient streams, host real equality, selected external limits, hidden tail oracles, countable-choice selectors, and ambient completeness as sources for the admitted subroute.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceUp.RegularCauchySequenceCarrier finite window witness induction`

Proof. Project the carrier of Definition 660.1. The displayed coordinates give the finite family row, modulus row, selected-window spine, dyadic ledger, regular readback row, coverage row, seal row, and structural rows of the accepted packet.

For the base route, no selected window has yet been exposed. Hence no consumer read can be formed before W, because the carrier offers no readback or diagonal-coverage coordinate outside the displayed packet.

For the successor route, adjoin one selected window. Apply Definition 660.3 to bind the selected window to μ, D, R, then apply Definition 660.4 to transport the same window through G, H, C, P, N.

Now use Definition 660.5. The extended route still reaches the $\mathsf{Real}^{\uparrow}$ seal row E only after the finite selected-window, dyadic, readback, coverage, and structural rows have been displayed.

Finally close with Definition 660.7. Since the carrier contains no coordinate for quotient streams, host real equality, selected external limits, hidden tail oracles, countable-choice selectors, or ambient completeness, the induction cannot export any off-surface source. □

Lemma 660.7 (Regular Cauchy sequence no-escape)**.** *An accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet exports only the rows*

$$F, \mu, W, D, R, G, E, H, C, P, N.$$

Every consumer of its regular Cauchy family data factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows displayed in the packet. It exports no quotient regular-sequence class, no host equality on completed reals, no selected external limit, no choice principle, and no ambient completeness theorem.*

Proof. Inspect the carrier coordinates in Definition 660.1. The only mathematical rows are the family, modulus, selected windows, dyadic ledger, regular readback, coverage row, and $\mathsf{Real}^{\uparrow}$ handoff.

The structural rows H, C, P, N record transport, continuation, provenance, and naming discipline for those same coordinates. There is no coordinate for quotient regular-sequence equality, host real equality, external limit selection, choice, or completeness.

Therefore any consumer route must factor through the displayed finite packet. The excluded objects cannot be exported because the carrier never provides them. □

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceUp.RegularCauchySequenceNonEscape`

Lemma 660.8 (Regular Cauchy sequence transport stability)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

and

$$S' = (F', \mu', W', D', R', G', E', H, C, P, N)$$

be accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packets whose family, modulus, selected-window, dyadic-ledger, regular-readback, coverage, and handoff rows are related by the displayed* hsame *transports named in the shared* H *row. Then both packets present the same public selected-window surface to every downstream* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer using the shared* C, P, N *route.*

Proof. Use Definition 660.2. The displayed hsame rows relate exactly the data coordinates F, μ, W, D, R, G, E and leave the continuation, provenance, and local naming rows shared.

Apply Definition 660.4. Its classifier-stability and selected-window regularity clauses keep each transported window tied to the transported modulus, dyadic ledger, and regular readback row.

The coverage row G and its transported counterpart G' expose the same finite diagonal-window reads through the shared route C, P, N. Hence the downstream $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ views classify the handoff by the same finite selected-window surface. □

Theorem 660.9 (Regular Cauchy sequence RealUp seal handoff)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carrier*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *row* E *is reachable only after the regular family* F*, modulus* μ*, selected windows* W*, dyadic ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *coverage row* G *have been displayed. Every diagonal-limit handoff read factors through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows and cannot bypass the finite selected-window packet.*

Proof. Start with Definition 660.4. The selected-window regularity and diagonal-window coverage clauses bind every handoff read to F, μ, W, D, R, G before the seal row is consumed.

Use Definition 660.8. Transporting the selected-window packet preserves the public surface seen by $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, so the handoff is stable under the displayed hsame rows.

Finally apply Definition 660.7. Since no quotient sequence, host real equality, selected external limit, choice principle, or ambient completeness theorem is present, the $\mathsf{Real}^{\uparrow}$ row E is the finite seal handoff produced by the displayed packet alone. □

This handoff is horizontal in the concrete-instance lattice. The selected windows are read through the dyadic ledger and the $\mathsf{RegSeqRat}^{\uparrow}$ layer of chapter 301; only after those rows are displayed may chapter 561 consume the finite diagonal-window coverage row and expose the resulting $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Theorem 660.10 (Regular Cauchy sequence selected-window coverage)**.** *For every accepted* RegularCauchySequenc *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

each accepted finite selected-window read is routed through the displayed family row F*, selected-window row* W*, dyadic ledger* D*, regular readback row* R*, transport row* H*, continuation row* C*, provenance row* P*, and local NameCert row* N*. The packet admits no quotient stream, host limit, choice coordinate, or ambient completeness coordinate outside those rows.*

Proof. Project the packet by Definition 660.1. The selected-window coordinates visible to a consumer are F, W, D, R, H, C, P, N, together with the modulus and downstream coverage rows that keep the route finite.

Apply Definition 660.3. Every selected window in W receives its precision discipline from μ and its rational observations from D and R, so the read cannot be formed before the displayed finite rows are present.

Use Definition 660.7. Since the carrier has no coordinate for a quotient stream, host limit, choice principle, or ambient completeness theorem, no accepted selected-window read can leave the displayed route. □

Theorem 660.11 (Regular Cauchy sequence diagonal-window coverage handoff)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carrier*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

the $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *row* G *consumes exactly the selected* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the displayed transports* H*, continuations* C*, package provenance* P*, and local NameCert row* N*. It has no additional diagonal read, hidden stream coordinate, or external limit coordinate.*

Proof. Start from the diagonal-window obligation in Definition 660.4. The coverage row G is required to expose exactly the finite selected reads consumed by the handoff.

Apply Definition 660.10. Those selected reads are already confined to W, D, R, H, C, P, N, with F and μ supplying the finite regular-family discipline.

Finally use Definition 660.9. The row E is reachable only after F, μ, W, D, R, G have been displayed, so the coverage handoff consumes the listed rows and nothing outside the finite packet. □

Theorem 660.12 (Regular Cauchy sequence budget-window synchronization)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Suppose the selected-window surface of* S *is consumed by an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *route or by an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget route. Then the budget route and the* $\mathsf{RegularCauchySequence}^{\uparrow}$ *handoff read the same finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal row through the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Equivalently, the budget-facing consumer may only use the selected windows W*, dyadic ledger* D*, regular readback* R*, diagonal-window coverage row* G*, seal row* E*, transports* H*, continuations* C*, provenance* P*, and local name* N *already displayed by* S *and by the seal-budget route. It receives no quotient stream, host real equality, selected limit, countable-choice coordinate, or ambient completeness theorem.*

Proof. First confine the regular-family side. By Definition 660.10, every selected-window read of S is already routed through $F, \mu, W, D, R, H, C, P, N$, with no extra host stream or completeness coordinate.

Next apply the diagonal-window handoff of Definition 660.11. It identifies G as the consumer row that reads the selected $\mathsf{StreamName}^{\uparrow}$ windows together with the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal row, displayed transports, continuations, provenance, and local naming row.

Now read the budget route. Budget root source totality from Definition 386.87 says an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget consumer is sourced by the displayed seal and budget rows, rather than by an independent completion object.

Finally use dyadic-window exactness from Definition 386.88. The budget ledger is the scheduled finite window ledger already exposed by the seal packet, so the composed route reads the same W, D, R, E, H, C, P, N surface as the regular Cauchy sequence handoff. The non-escape clauses in the cited rows exclude quotient streams, host real equality, selected limits, countable choice, and ambient completeness. □

Theorem 660.13 (Regular Cauchy sequence Real observation budget seal factorization)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet, and let*

$$B = (E_B, W_B, D_B, R_B, S_B, H_B, C_B, P_B, N_B)$$

be an accepted $\mathsf{RealObservationBudget}^{\uparrow}$ *packet whose displayed rows read the selected windows, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, diagonal-window coverage, and* $\mathsf{Real}^{\uparrow}$ *seal of* S *through the displayed* hsame, Cont, Pkg, *and* NameCert *rows. Then every* $\mathsf{Real}^{\uparrow}$*-facing consumer of* S *factors through the finite route*

$$E_B \longmapsto W_B \longmapsto D_B \longmapsto R_B \longmapsto S_B$$

before reaching the seal row E*. The factorization exports no quotient stream, host real equality, selected limit, countable-choice coordinate, hidden schedule, or ambient completeness theorem.*

Proof. First synchronize the two visible window ledgers. By Definition 660.12, the $\mathsf{RealObservationBudget}^{\uparrow}$ packet and the regular Cauchy handoff read the same finite selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, diagonal-window coverage, and $\mathsf{Real}^{\uparrow}$ seal surface through the displayed transport, continuation, provenance, and local naming rows.

Next use the observation-budget route totality. Definition 611.17 forces every admitted $\mathsf{Real}^{\uparrow}$ seal read of B through

$$E_B \longmapsto W_B \longmapsto D_B \longmapsto R_B \longmapsto S_B.$$

Now attach the consumer side. Definition 611.31 says that every admitted $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer of the budget packet factors through the same finite route with the displayed hsame, Cont, Pkg, and NameCert rows. Since the synchronized rows are exactly the selected-window and seal rows of S, the $\mathsf{Real}^{\uparrow}$-facing consumer of S must pass through E_B, W_B, D_B, R_B, S_B before reaching E.

Finally close the non-escape boundary. Definition 660.7 and Definition 611.6 leave no coordinate for quotient streams, host real equality, selected limits, countable choice, hidden schedules, or ambient completeness, so no consumer can bypass the finite route. □

Theorem 660.14 (Regular Cauchy sequence selector-budget Real-completeness route)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Suppose its selected-window packet is consumed by an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *route and then by a* $\mathsf{RealCompleteness}^{\uparrow}$ *handoff. Then the RealCompleteness-facing consumer reads the same finite rows* $F, \mu, W, D, R, G, E, H, C, P, N$ *displayed by* S*. The selector-budget rows may refine only the selected finite window route already named by* $W, D, R, G, E, H, C, P, N$*; they add no quotient stream, host limit, countable-choice selector, hidden window enlargement, or ambient completeness row.*

Proof. Begin with selected-window coverage. By Definition 660.10, every selected read of S is already confined to the displayed family row, modulus row, finite windows, dyadic ledger, regular readback, transports, continuations, provenance, and local naming rows.

Next attach the observation-budget seal factorization of Definition 660.13. It keeps the Real-facing seal route on the same selected-window, dyadic, readback, coverage, and seal rows before the consumer reaches E.

Now use the selector-budget factor and its Real-completeness handoff, Definition 660.17 and Definition 678.13. These rows allow the selector budget to inspect and refine the displayed finite window route, but they do not give it a second stream, a host-selected limit, or a larger hidden window.

Finally apply the local non-escape boundary of Definition 660.7. Since the accepted packet has no coordinate outside $F, \mu, W, D, R, G, E, H, C, P, N$, the composed RealCompleteness route cannot leave the displayed finite packet. □

Theorem 660.15 (Regular Cauchy sequence shared-tail threshold extraction)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. If a downstream* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *or* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route consumes the selected-window and seal rows of* S*, then that consumer must first expose the same finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *already displayed by* S*. The shared-tail threshold is therefore extracted from displayed rows of the RegularCauchySequence packet and its sibling consumer routes, not from quotient streams, selected limits, choice, or ambient completeness.*

Proof. Begin with selected-window coverage. By Definition 660.10, every selected read of S is already routed through the family row F, modulus row μ, finite windows W, dyadic ledger D, regular readback R, and the structural rows H, C, P, N.

Next use the diagonal-window coverage handoff. Definition 660.11 identifies G as the consumer row that receives exactly those selected windows, dyadic ledger rows, and regular readbacks before any Real-facing handoff is considered.

Now apply the budget-window synchronization theorem, Definition 660.12. It synchronizes the selected-window route with the Real observation budget, so a seal-facing consumer cannot replace W, D, R by a hidden schedule or a separate threshold source.

Finally attach the sibling shared-tail surfaces. The $\mathsf{CauchySharedTailModulus}^{\uparrow}$ and $\mathsf{RealTailAgreementSeal}^{\uparrow}$ routes consume only displayed finite threshold, window, readback, and seal rows, and Definition 660.9 places E after those rows. Together with Definition 660.7, this rules out quotient streams, selected limits, choice, and ambient completeness as sources for the shared threshold. □

Theorem 660.16 (Regular Cauchy sequence diagonal tail selector factorization)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Let*

$$Q = (R', n', \mu', k', W', D', T', S', H', C', P', N')$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet whose* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows match the selected-window and handoff rows of* S *by the displayed* hsame *transports. Then every diagonal-tail consumer factors first through Definition 660.11 and then through Definition 697.6. The readable route is exactly the selected windows, dyadic ledger, regular readback, coverage row, selector modulus, selected index, finite selector window, selector readback, tail-regularity row, displayed transports, continuations, provenance, and local* NameCert *rows. The factorization exports no quotient stream, host real equality, countable-choice selector, host-selected limit, hidden tail oracle, or ambient completeness theorem.*

Proof. First confine the regular Cauchy side. By Definition 660.10, every selected read of S is already routed through $F, \mu, W, D, R, H, C, P, N$. The diagonal-window handoff theorem then identifies G as the $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ row that consumes those selected windows together with the dyadic ledger, regular readback, seal row, transports, continuations, provenance, and local naming row.

The seal order is fixed by Definition 660.9. Thus the $\mathsf{Real}^{\uparrow}$ row visible to the selector side is not an independent completed value; it is reached only after the finite regular-family rows and the coverage row have been displayed.

Now read the selector packet. The obligations in Definition 697.2 bind $R', n', \mu', k', W', D', T', S', H', C', P', N'$ into one finite selector surface, with k' read from μ' at n' and the tail-regularity row tied to the displayed regular-source row. The displayed hsame transports identify the selector's $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows with the corresponding selected-window and handoff rows of S.

Finally apply Definition 697.6 and Definition 697.19. The completion-facing consumer therefore passes through the regular Cauchy diagonal-window coverage handoff before it reaches the selector limit handoff, and the selector seal contributes no quotient stream, host real equality, countable-choice selector, host-selected limit, hidden tail oracle, or ambient completeness theorem. □

Theorem 660.17 (Regular Cauchy sequence selector-budget factor)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *diagonal route factors through the sibling* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *finite packet of chapter 678 before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness and* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, for*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

the consumer first reads the local family row F, modulus row μ, selected windows W, dyadic ledger D, diagonal coverage row G, and structural rows H, C, P, N; it then consumes the selector-budget packet as the finite route that exposes the $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness, and* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no host limit, quotient regular sequence, countable choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the local sequence carrier and selected-window obligations. By Definition 660.1, Definition 660.3, and Definition 660.10, every selected read is confined to $F, \mu, W, D, R, H, C, P, N$ with the finite family discipline supplied by the modulus and dyadic ledger.

Use the diagonal-window handoff of Definition 660.11. The coverage row G exposes only the selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{Real}^{\uparrow}$ seal row already listed in the packet.

Now route the Real-facing edge through chapter 678. Its selector-budget packet is precisely the finite consumer dependency that names the selector modulus, window budget, StreamName observations, RegSeqRat witness, RealUp seal, and the structural H, C, P, N rows before the seal is read.

Finally apply Definition 660.7 and the local $\mathsf{Real}^{\uparrow}$ seal handoff of Definition 660.9. The factored route therefore reaches R and E only through the selector-budget packet and cannot use a host limit, quotient regular sequence, countable choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness. □

Theorem 660.18 (Regular Cauchy sequence obligation closure)**.** *The* $\mathsf{RegularCauchySequence}^{\uparrow}$ *carrier, classifier, selected-window regularity, non-escape boundary, transport stability, diagonal-window coverage handoff,* $\mathsf{Real}^{\uparrow}$ *handoff, and provenance rows form the full seed-to-obligation package for the local NameCert surface. Concretely, every accepted packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

presents carrier admission, rowwise classifier stability, selected-window coverage, diagonal-window coverage, seal handoff, transport invariance, package provenance, and the exclusion of quotient streams, host limits, choice, and ambient completeness as one finite certificate.

Proof. Carrier admission and rowwise classifier stability are supplied by Definition 660.1 and Definition 660.2. The package rows H, C, P, N are part of the same tuple, so the certificate has named transport, continuation, provenance, and local naming surfaces.

Selected-window regularity and coverage are provided by Definition 660.3 and Definition 660.10. These results bind every selected read to the finite family row, modulus row, selected windows, dyadic ledger, and regular readback row.

The diagonal and seal handoff surfaces are given by Definition 660.11 and Definition 660.9. Transport stability follows from Definition 660.8, and the non-escape boundary follows from Definition 660.7.

These projections cover the local NameCert obligations listed in Definition 660.4. No further coordinate is available in the carrier, so the listed rows form the complete finite obligation package for this chapter's $\mathsf{RegularCauchySequence}^{\uparrow}$ surface. □

Theorem 660.19 (Regular Cauchy sequence shared threshold import)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Suppose an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet displays a shared tail threshold whose two source rows are read from the same family row F, selected windows W, dyadic ledger D, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and suppose its diagonal-window and* $\mathsf{Real}^{\uparrow}$ *consumers are transported through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows as G, E, H, C, P, N. Then the* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet may import that displayed shared threshold as a consumer-visible tail row exactly along those displayed transports.*

The import adds no quotient stream, selected limit, host equality, choice coordinate, hidden tail oracle, or ambient completeness theorem. It only records that the RegularCauchyTailMeet public shared-threshold certificate is read through the already displayed regular-family, selected-window, dyadic, readback, diagonal-coverage, seal, transport, continuation, provenance, and naming rows.

Proof. Start with the carrier and classifier of Definition 660.1 and Definition 660.2. They make $F, \mu, W, D, R, G, E, H, C, P, N$ the complete rowwise surface visible to the regular Cauchy sequence packet.

Use selected-window coverage from Definition 660.10. It confines every finite selected read to the same family, window, dyadic, readback, transport, continuation, provenance, and local naming rows that the tail-meet packet is assumed to read.

Next apply the diagonal-window coverage handoff and the $\mathsf{Real}^{\uparrow}$ seal handoff, Definition 660.11 and Definition 660.9. These identify G and E as consumer rows reached only after the finite selected-window packet has been displayed.

Finally apply the RegularCauchyTailMeet public shared-threshold certificate, Definition 560.11. Because its threshold route is transported through the same BHist, hsame, Cont, Pkg, and NameCert rows, importing the threshold is just a replay of that displayed certificate inside the regular Cauchy sequence packet. □

Theorem 660.20 (Regular Cauchy sequence tail-meet consumer surface)**.** *Under the hypotheses of Definition 660.19, every downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$*, or completion-facing consumer sees the combined packet only as*

$$F, \mu, W, D, R, G, E, H, C, P, N$$

together with the shared-tail rows displayed by the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet. The consumer surface has no quotient stream, no selected limit, no host equality, no choice coordinate, and no ambient completeness coordinate.*

Proof. First read the regular Cauchy sequence side. By Definition 660.18, the local NameCert surface is the finite packet $F, \mu, W, D, R, G, E, H, C, P, N$ with the listed non-escape boundary.

Now read the imported shared threshold. The import theorem routes the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ certificate through the same selected-window, dyadic, readback, coverage, seal, transport, continuation, package, and naming rows.

It remains to identify what a downstream consumer can see. Apply Definition 560.9 and Definition 560.11. These expose only the shared-tail rows of the tail-meet packet, so the combined surface is exactly the displayed regular Cauchy sequence packet plus those shared-tail rows. □

Theorem 660.21 (Regular Cauchy sequence observation-budget TailMeet consumer)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Suppose a downstream consumer reads a* $\mathsf{RealObservationBudget}^{\uparrow}$ *route for the seal row of S and also reads an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *route whose shared threshold is imported through the displayed selected-window, dyadic, readback, transport, continuation, provenance, and local naming rows of S. Then the consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through one shared finite route: the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rows, shared threshold row,* hsame *transports,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ *row.*

No accepted consumer may use the observation budget to bypass the imported TailMeet threshold, replace the displayed route by quotient streams or host real equality, read hidden selector rows, or import ambient completeness.

Proof. Begin with the observation-budget side. By Definition 660.13, every $\mathsf{Real}^{\uparrow}$-facing read of S through a $\mathsf{RealObservationBudget}^{\uparrow}$ packet first factors through the finite window, dyadic, readback, coverage, seal, transport, continuation, provenance, and naming rows displayed by the budget-to-seal route.

Next attach the TailMeet side. The consumer-surface theorem Definition 660.20 says that an imported $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold is visible to downstream Real or completion consumers only as the regular Cauchy packet together with the accepted tail-meet rows transported through the same displayed finite surface.

The two routes therefore share the selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, seal row, transports, continuations, provenance, and local name. The shared threshold is read after those rows have been displayed and before any consumer can treat the seal as available.

Finally use the carrier inventory of Definition 660.1 and the non-escape boundary of Definition 660.7. Since the carrier has no coordinate for quotient streams, host real equality, hidden selector rows, or ambient completeness, and since both cited routes consume only the displayed rows, no alternate path to the $\mathsf{Real}^{\uparrow}$ seal remains. □

Theorem 660.22 (Regular Cauchy sequence tail-selector budget lock)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *route. Suppose its selected-window rows factor through the sibling* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet of chapter 678, and suppose its shared threshold factors through an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet from chapter 560. Then every* $\mathsf{Real}^{\uparrow}$*-facing tail consumer of S must read the same displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows carried by the selected-window and shared-threshold route.*

The locked consumer packet is the finite surface presented by chapter 126, chapter 301, chapter 291, and chapter 38 as those rows occur inside S and the transported tail-meet certificate. It contains no quotient stream, no host limit, no hidden selector, and no alternate tail threshold.

Proof. First apply the selector-budget factorization of Definition 660.17. It forces each Real-facing selected-window read through the finite selector-budget packet before the $\mathsf{RegSeqRat}^{\uparrow}$ witness and $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Next apply the tail-meet consumer surface of Definition 660.20. The shared threshold imported from chapter 560 is visible only through the same selected-window, dyadic, readback, seal, transport, continuation, provenance, and local naming rows already displayed by the regular Cauchy sequence packet.

The concrete sibling chapters identify the projected rows. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ packet surfaces of chapter 126, chapter 301, chapter 291, and chapter 38 are therefore read at the coordinates exposed by S and by the transported tail-meet threshold, not at fresh host coordinates.

Since both routes share H, C, P, N, there is no remaining coordinate from which a consumer could obtain a quotient stream, host limit, hidden selector, or alternate tail threshold. The tail-facing Real consumer is locked to the single displayed finite packet. □

Theorem 660.23 (Regular Cauchy sequence tail-fusion budget handoff)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Suppose a downstream* $\mathsf{Real}^{\uparrow}$*-facing consumer reads an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet whose selected windows, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold, and* $\mathsf{Real}^{\uparrow}$ *seal rows are transported through the rows displayed by S. Suppose also that the same consumer reads either a* $\mathsf{RealObservationBudget}^{\uparrow}$ *route or a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget route for the displayed seal. Then the consumer must pass through the single finite route*

$$W, D, R, G, E, H, C, P, N$$

together with the transported tail-fusion shared-threshold rows before it can read the $\mathsf{Real}^{\uparrow}$ *seal.*

The handoff exports no quotient stream, host real equality, selected completed limit, countable-choice coordinate, hidden tail oracle, or ambient completeness theorem.

Proof. First lock the selected-window and shared-threshold rows. By Definition 660.22, every tail-facing $\mathsf{Real}^{\uparrow}$ consumer of S reads the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows carried by the selected-window and shared-threshold route.

Next read the budget side. The observation-budget TailMeet consumer theorem, Definition 660.21, forces any $\mathsf{RealObservationBudget}^{\uparrow}$ route through the same finite selected windows, dyadic ledger, regular readback, shared threshold, transports, continuations, provenance, and local name. The seal factorization theorem Definition 660.13 gives the corresponding factorization for the displayed $\mathsf{Real}^{\uparrow}$ seal row.

Now attach the sibling tail-fusion packet. Its public export Definition 658.17 exposes only the finite route through precision, regular-family, selected-window, dyadic, tail-witness, shared-threshold, diagonal readback, seal, transport, continuation, provenance, and naming rows, while Definition 658.23 makes a completion-facing consumer pass through that route before any public completion read.

Substituting the rows transported through S identifies the visible route as $W, D, R, G, E, H, C, P, N$ together with the transported tail-fusion shared-threshold rows. The budget route and the tail-fusion route therefore meet before the $\mathsf{Real}^{\uparrow}$ seal rather than supplying parallel access paths.

Finally apply Definition 660.7. The carrier has no coordinate for quotient streams, host real equality, selected completed limits, countable choice, hidden tail oracles, or ambient completeness, so no consumer can leave the displayed finite handoff. □

Theorem 660.24 (Regular Cauchy sequence tail-selector exhaustion)**.** *Let*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet. Any completion-facing consumer that reads a* $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *or* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *row transported through* S *must factor through the single finite route*

$$F, \mu, W, D, R, G, E, H, C, P, N$$

together with the transported selector and shared-tail rows before reading the $\mathsf{Real}^{\uparrow}$ *seal.*

The exhausted route contains no quotient stream, host real equality, selected limit, countable-choice coordinate, hidden tail oracle, alternate tail threshold, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. First route the selector side. The diagonal tail-selector factorization of Definition 660.16 binds every transported $\mathsf{DiagonalTailSelector}^{\uparrow}$ read to the displayed family, selector, window, dyadic, readback, diagonal, seal, transport, continuation, package, and naming rows of S.

Next force the selected budget. Apply Definition 660.17 to pass every selector-budget consumer through the same selected-window packet, and then apply Definition 660.22 to identify the shared finite rows with the transported $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold.

Now attach the fusion route. The handoff theorem Definition 660.23 makes the tail-fusion and budget routes meet on $W, D, R, G, E, H, C, P, N$ before any $\mathsf{Real}^{\uparrow}$ seal read.

Finally use Definition 660.7. The accepted carrier has no coordinate for quotient streams, host real equality, selected limits, countable choice, hidden tail oracles, alternate thresholds, or ambient completeness, so every Real-facing tail-selector consumer is exhausted by the single finite diagonal-tail route. □

Theorem 660.25 (Regular Cauchy sequence root obligation export)**.** *The* $\mathsf{RegularCauchySequence}^{\uparrow}$ *root obligation exported to later Real and Completion formalization consists of the local obligation closure of Definition 660.18, the shared-threshold import of Definition 660.19, and the consumer-surface boundary of Definition 660.20. Thus a later diagonal route may consume regular-family data through the displayed packet and a transported* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold, but it receives no quotient stream, selected limit, host equality, choice coordinate, hidden tail oracle, or ambient completeness theorem.*

Proof. Assemble the local packet first. The obligation closure theorem packages carrier admission, classifier stability, selected-window coverage, diagonal-window coverage, $\mathsf{Real}^{\uparrow}$ handoff, transport, provenance, and non-escape for $F, \mu, W, D, R, G, E, H, C, P, N$.

Attach the shared threshold by Definition 660.19. The import is valid only when the family row, selected windows, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, diagonal-window coverage row, $\mathsf{Real}^{\uparrow}$ handoff, and structural rows are the same displayed rows under BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ transport.

Finally apply Definition 660.20. It records the downstream boundary for Real and Completion consumers, so the root export is a finite current packet plus the displayed shared-tail certificate, with all off-surface completion and quotient data excluded. □

Theorem 660.26 (Regular Cauchy sequence scoped dependency surface)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

the chapter's obligation package is scoped exactly over the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Every selected-window, diagonal-coverage, shared-threshold, and Real seal consumer factors through those rows.*

Consequently no later $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or completion-facing consumer may use a quotient stream, host limit, choice coordinate, hidden tail oracle, or ambient completeness theorem when reading the* $\mathsf{RegularCauchySequence}^{\uparrow}$ *package.*

Proof. Start from the obligation closure theorem, Definition 660.18. It names the carrier, classifier, selected-window coverage, diagonal-window coverage, $\mathsf{Real}^{\uparrow}$ handoff, transport, provenance, and non-escape rows inside the same finite packet S.

The selected-window and diagonal consumers are then read through Definition 660.11 and Definition 660.9. These two handoffs account for the $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows without adding any external limit or quotient coordinate.

Next attach the shared-threshold route by Definition 660.19. Its source rows are the displayed family, window, dyadic, readback, diagonal, seal, transport, continuation, package, and naming rows.

Finally apply Definition 660.25. The root export is therefore a scoped dependency surface over exactly the listed rows, and the non-escape clauses rule out quotient streams, host limits, choice, hidden tail oracles, and ambient completeness for every downstream consumer. □

Theorem 660.27 (Regular Cauchy sequence public export)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

the carrier, rowwise classifier, selected-window regularity, diagonal-window coverage, selector-budget factor, shared-threshold import, and $\mathsf{Real}^{\uparrow}$ *seal handoff assemble into one public* $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ *export. The export is exactly the finite surface over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *already displayed by the packet.*

No quotient stream, host limit, selected external limit, choice coordinate, hidden tail oracle, or ambient completeness theorem is part of this public export.

Proof. Begin with Definition 660.18. It packages the carrier, classifier, selected-window regularity, diagonal-window coverage, $\mathsf{Real}^{\uparrow}$ handoff, transport, provenance, and non-escape rows for the single finite packet S.

Next apply Definition 660.26. This identifies the exact dependency surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Attach the tail-facing factors through Definition 660.17, Definition 660.22, and Definition 660.25. These theorems route the selected-window budget and shared threshold through the same displayed packet before any Real-facing consumer can read the seal.

The resulting public $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ export is therefore the displayed finite packet and its imported shared-threshold surface. The non-escape clauses in the cited theorems rule out quotient streams, host limits, choice, hidden tail oracles, selected external limits, and ambient completeness. □

Theorem 660.28 (Regular Cauchy sequence mature Real and Completion consumer route)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

every downstream Real or Completion consumer route that uses the regular Cauchy family factors through the selected-window rows W*, dyadic ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *row* G*,* $\mathsf{Real}^{\uparrow}$ *handoff* E*, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows carried by* H, C, P, N*.*

This route consumes the public export and scoped dependency surface of the packet. It exports no quotient stream, host limit, selected external limit, choice coordinate, hidden tail oracle, ambient completeness theorem, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the public export of Definition 660.27. It assembles the carrier, rowwise classifier, selected-window regularity, diagonal-window coverage, selector-budget factor, shared-threshold import, and $\mathsf{Real}^{\uparrow}$ seal handoff into the finite $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ surface.

Next use Definition 660.26. That theorem identifies the allowed dependency rows as the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so any Real or Completion consumer must read the same finite packet.

The seal portion is fixed by Definition 660.9. The row E is reached only after the selected windows W, dyadic ledger D, regular-readback row R, and diagonal coverage row G have been displayed, so the route has no independent host limit or selected external limit.

Finally apply Definition 660.7. Since the carrier contains no coordinate for quotient streams, choice, hidden tail oracles, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, the consumer route is exhausted by the finite packet S and exports no off-surface completion data. □

Theorem 660.29 (Regular Cauchy sequence Cauchy witness ledger handoff)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

whose selected-window rows, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *row, and* $\mathsf{Real}^{\uparrow}$ *seal handoff are consumed by an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet of chapter 902, every completion-facing read passes through the same finite route* $W, D, R, G, E, H, C, P, N$ *before the witness ledger can read its observation-budget and completion-seal rows.*

The handoff exports no quotient stream, host limit, selected external limit, choice coordinate, hidden tail oracle, ambient completeness theorem, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 660.27. The accepted packet exposes the selected-window rows, dyadic ledger, regular-rational readback, diagonal-window coverage, $\mathsf{Real}^{\uparrow}$ seal handoff, and structural rows as one public $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ surface.

Apply Definition 660.28. Any Real or Completion consumer using the regular Cauchy family must read W, D, R, G, E and the structural rows H, C, P, N, with no off-surface limit, choice, or completeness coordinate.

Now read the witness-ledger side through Definition 902.5 and Definition 902.9. The ledger reaches its observation-budget and completion-seal classifier rows only through the displayed diagonal-budget, observation-budget, seal-synchronizer, and structural route.

Combining these routes forces the Cauchy witness ledger consumer to inherit the regular Cauchy packet's finite public route before any completion-facing read is admitted. The same non-escape

clauses from the regular Cauchy packet and the witness ledger remove quotient streams, host limits, selected external limits, choice coordinates, hidden tail oracles, ambient completeness, and $\mathsf{Real}^{\uparrow}$ completeness from the handoff. □

Theorem 660.30 (Regular Cauchy sequence modulus-refinement consumer lock)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

whose selected-window and seal rows are consumed by an accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer must pass through the refinement route* $M_0, M_1 \to U \to V \to T \to W \to Q \to E$ *before reading the* $\mathsf{Real}^{\uparrow}$ *seal.*

The consumer receives no quotient stream, host limit, selected external limit, choice coordinate, hidden selector, alternate meet row, or ambient completeness theorem.

Proof. Start with Definition 660.28 and Definition 660.27. These results confine the $\mathsf{RegularCauchySequence}^{\uparrow}$ side to the displayed packet S, its selected-window rows, its diagonal coverage row, and its public $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ export.

The refinement order is then forced by Definition 885.11 and Definition 885.12. The source meet and budget rows reach the selected-window and tolerance readbacks only along the displayed M_0, M_1, U, V, T, W, Q spine.

Next apply Definition 885.7. The $\mathsf{Real}^{\uparrow}$ seal row is reachable only after that spine has been displayed, so a Real or Completion consumer cannot read the seal directly from the regular Cauchy packet.

Finally combine Definition 885.10 with Definition 660.7. The two non-escape boundaries remove quotient streams, host limits, selected external limits, choice coordinates, hidden selectors, alternate meet rows, and ambient completeness theorems from the consumer surface. □

Theorem 660.31 (Regular Cauchy sequence budgeted modulus-witness handoff)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

suppose that its selected-window and $\mathsf{Real}^{\uparrow}$ *seal rows are consumed by both an accepted* RegularCauchyModulusWitness *packet and an accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet. Then the two consumers expose the same finite selected-window row* W*, dyadic ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, diagonal coverage row* G*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by* H, C, P, N *before either consumer reads a completion-facing seal.*

The handoff admits no quotient stream, host real equality, selected limit, choice coordinate, hidden selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with Definition 660.28 and Definition 660.30. They force every Real or Completion consumer of the accepted regular Cauchy packet to read through the displayed $W, D, R, G, E, H, C, P, N$ surface before the $\mathsf{Real}^{\uparrow}$ seal is available.

Next apply Definition 977.7. The modulus-witness ledger side reaches its Real-facing seal only through its displayed finite ledger route, so the rows imported from S cannot be replaced by a second selected window, dyadic ledger, readback row, coverage row, seal row, or structural transport package.

Finally use Definition 913.10. The budgeted seal route identifies the selector, threshold, observation-budget, seal, Real readback, transport, continuation, package, and naming rows at

its source before any completion-facing read is admitted. Combining this source exactness with the regular Cauchy and modulus-witness routes makes the shared handoff the single finite packet surface $W, D, R, G, E, H, C, P, N$ and excludes quotient streams, host real equality, selected limits, choice, hidden selectors, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 660.32 (Regular Cauchy sequence Real four-face exit readback)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

any Phase $\mathsf{Real}^{\uparrow}$ *or Completion read that uses S factors through the same finite four-face exit surface carried by* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The only support rows used in this read are the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the packet.*

Conversely, the present four-face exit readback supplies exactly this finite route for S: the selected StreamName window, dyadic tolerance ledger, regular-rational readback, and terminal Real seal are the whole Real-and-Completion-facing boundary of the accepted regular Cauchy sequence packet.

The route contains no quotient streams, host equality, selected limits, choice rows, metric-completeness data, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the public export theorem Definition 660.27. It exposes the accepted packet S as one finite $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$ surface over the StreamName, dyadic, regular-readback, diagonal-coverage, Real seal, and structural support rows.

Next apply Definition 660.28 and Definition 660.31. These two routes force every Real or Completion consumer that reads S to pass through $W, D, R, G, E, H, C, P, N$ before any completion-facing seal is available.

Now identify the terminal boundary. The StreamName four-face readback theorem Definition 126.450 reads the same boundary through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while Definition 126.461 makes that boundary terminal for Phase $\mathsf{Real}^{\uparrow}$ and Completion reads.

For the converse direction, take the four-face exit readback just described and restrict it along the public export of S. The selected window, dyadic ledger, regular readback, and Real seal rows are already the displayed rows of the accepted regular Cauchy sequence packet, so the readback supplies exactly the finite route stated above.

Finally apply Definition 660.7. The carrier has no constructor for quotient streams, host equality, selected limits, choice rows, metric-completeness data, bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no off-surface resource can enter either direction of the readback. □

Theorem 660.33 (Regular Cauchy sequence finite-tail closure)**.** *For every accepted* $\mathsf{RegularCauchySequence}^{\uparrow}$ *packet*

$$S = (F, \mu, W, D, R, G, E, H, C, P, N),$$

whose tail selector and budgeted modulus witness rows are visible, every finite-tail Real or Completion consumer factors through the displayed route

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the support rows H, C, P, N. The route uses the selected windows, dyadic budget, regular rational readback, diagonal coverage row, and terminal Real seal already present in S; it exports no selected limit, quotient stream equality, host real equality, hidden tail oracle, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with Definition 660.24. It forces every Real-facing tail-selector consumer of S through the displayed diagonal-tail route and excludes hidden tail oracles, alternate thresholds, quotient streams, selected limits, choice, and ambient completeness.

Next use Definition 660.30. The refinement route fixes the source meet, budget rows, selected-window rows, tolerance readback, and Real seal before any completion-facing consumer can inspect the terminal seal.

Now apply Definition 660.31. The modulus-witness and budgeted Real-seal consumers must share the same selected-window row W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, diagonal coverage row G, $\mathsf{Real}^{\uparrow}$ seal row E, and structural rows H, C, P, N.

Finally apply Definition 660.32. It identifies the terminal Real and Completion boundary with the four faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The finite-tail consumer therefore closes through the displayed rows of S and cannot add a selected limit, quotient equality, host equality, hidden tail oracle, choice selector, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySequence}^{\uparrow}$ packet is generated from a finite regular Cauchy family row, explicit modulus, selected StreamName windows, a DyadicRatCore readback ledger, RegSeqRat readbacks, RealDiagonalWindowCoverage coverage, a RealUp handoff row, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 660.1, Definition 660.2, Definition 660.3, Definition 660.4, Definition 660.7, Definition 660.8, Definition 660.9, Definition 660.12, Definition 660.13, Definition 660.18, Definition 660.24, Definition 660.25, Definition 660.26, Definition 660.27, Definition 660.28, Definition 660.30, Definition 660.31, Definition 660.32, and Definition 660.33 bind the seed carrier, rowwise classifier, finite selected-window obligations, transport discipline, non-escape boundary, RealUp handoff surface, RealObservationBudget window synchronization, observation-budget-to-seal factorization, tail-selector exhaustion, shared-threshold import, root obligation export, exact scoped dependency surface, Completion-facing handoff boundary, public NameCert export, mature Real and Completion consumer route, CauchyModulusRefinement consumer lock, budgeted modulus-witness handoff, Real four-face exit readback, and finite-tail closure route for regular Cauchy family data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySequence}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchySequenceUp.RegularCauchySequenceCarrier\_namecert\_obligations`

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, RealUp completeness, quotient equality of regular Cauchy sequences, host real equality, selected external limits, choice, or ambient completeness.

Upgrade path: No further theory-axis grade is pending; the remaining work is Lean verification and bridge checking.

Chapter 661

A Concrete Naming Certificate for $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$

The $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$ packet realizes the description-expansion boundary as a finite naming certificate. It treats a definite description as a row that may be consumed only after the packet exposes existence, uniqueness boundary, stability, hsame, Cont, Pkg, and local NameCert rows. No description row becomes a denoting term by grammar alone.

The description row specializes the Russell analysis ladder of chapter 518 and the contextual-class reading surface of chapter 684. This chapter consumes their description-expansion and contextual-class discipline, then adds the displayed existence, uniqueness-boundary, stability, and ledger rows D, E, U, S, H, C, P, N. When existence or uniqueness fails, the failed read remains an externalized socket in the sense of chapter 543, so the boundary exports refusal data rather than a denoting term.

The description row is also read through the negative naming boundary of chapter 629 before any existence or uniqueness row is consumed. Definite descriptions refine that refusal surface by adding the displayed D, E, U, S, H, C, P, N rows; they do not select a host denotation.

Definition 661.1 (Definite-description boundary carrier)**.** *A* $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, E, U, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the description row whose apparent denotation is being tested;*
2. *E is the existence-witness row for a candidate history satisfying the description;*
3. *U is the uniqueness-boundary row, recording either a displayed singleton boundary for the candidate or the explicit refusal to collapse multiple candidates;*
4. *S is the stability row showing that permitted reads of the description survive the stated* hsame *and continuation routes;*
5. *H is the componentwise* hsame *transport row for descriptions, witnesses, uniqueness boundaries, stability witnesses, replay routes, provenance, and local names;*
6. *C is the* Cont *row that replays the finite route from the description row to the admitted witness and through the boundary check;*

7. *P is the* Pkg *provenance over* BHist, BMark, *description rows, witness rows, uniqueness boundaries, stability rows,* hsame, Cont, *and* NameCert*;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{DefiniteDescriptionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-selected object, a choice function, a quotient class, proof-irrelevance, or an unrecorded semantic denotation.

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryTasteGate single carrier alignment`

Theorem 661.2 (Definite-description NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DefiniteDescriptionBoundary}^{\uparrow}}$ *surface for Definition 661.1 consists exactly of carrier admission for $B = (D, E, U, S, H, C, P, N)$, existence exposure through E, uniqueness-boundary exposure through U, stability exposure through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the finite tuple in Definition 661.1. The description row, witness row, boundary row, stability row, transport row, replay row, provenance row, and local name are all displayed coordinates of the same packet.

The naming obligations are exactly the rows a consumer can inspect before reading the description. Existence is read from E, the boundary between singleton and refusal is read from U, stability is read from S, and all transport, replay, provenance, and naming data are read through H, C, P, N.

No additional coordinate remains from which to select a host object or import an ambient denotation. The local NameCert surface is therefore exhausted by the listed rows. □

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundary namecert obligations`

Theorem 661.3 (Definite-description existence route)**.** *Let $B = (D, E, U, S, H, C, P, N)$ be an accepted* $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$ *packet. Any public use of the description row D as having a witness must factor through the displayed existence row E, replayed by C and recorded by P, N.*

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryExistenceRoute`

Proof. Project the accepted packet to its description row D and existence row E. The carrier contains no second existence coordinate and no host-level selection function.

The replay route C is the only displayed path from the description row to the witness row, and P, N bind that path to the packet provenance and local name. Thus a public existence read factors through E, C, P, N rather than through an unstated denotation. □

Theorem 661.4 (Definite-description uniqueness route)**.** *Let $B = (D, E, U, S, H, C, P, N)$ be an accepted* $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$ *packet. Any public use of D as selecting at most one witness must factor through the uniqueness-boundary row U. If U records refusal rather than singleton boundary, the packet exports only the refusal and does not export a denoting term.*

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryUniquenessRoute`

Proof. Inspect the carrier coordinates. The only row that can certify uniqueness, or refuse it, is U.

When U records a singleton boundary, the consumer reads that boundary through the same displayed replay and provenance rows C, P, N. When U records refusal, the packet has no alternate coordinate from which a selected witness can be recovered.

Therefore uniqueness is not supplied by the grammar of the description. It is supplied only by the displayed boundary row, and failure of that row blocks denotation. □

Theorem 661.5 (Definite-description stability route)**.** *Let $B = (D, E, U, S, H, C, P, N)$ be an accepted* DefiniteDescriptionBoundary$^{\uparrow}$ *packet. Any downstream read that transports the description across equivalent histories must use the stability row S together with the componentwise* hsame *row H and the replay row C.*

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryStabilityRoute`

Proof. The stability coordinate S records the permitted movement of the description read under the packet's own transport discipline. The transport coordinate H records the componentwise hsame data for the description, witness, boundary, stability, replay, provenance, and naming rows.

A downstream transport therefore has to pass through S, H, C. Package provenance and local naming remain the rows P, N, so the transported read is still the same finite certificate surface.

Since the carrier has no proof-irrelevance principle, quotient equality, or external equivalence oracle, no other stability route is available. □

Theorem 661.6 (Definite-description ledger exhaustion)**.** *Every accepted* DefiniteDescriptionBoundary$^{\uparrow}$ *packet exports description-expansion information only through the finite rows*

$$D, E, U, S, H, C, P, N.$$

It exports no host-selected witness, no choice principle, no quotient representative, no proof-irrelevance shortcut, and no denoting term outside the displayed NameCert ledger.

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryLedger exhaustion`

Proof. Read the carrier as the tuple $B = (D, E, U, S, H, C, P, N)$. The definition lists every row available to a downstream consumer.

Existence, uniqueness boundary, and stability are exhausted by E, U, S. Transport, replay, provenance, and local naming are exhausted by H, C, P, N.

Thus the apparent denoting expression is expanded into its certificate rows. With no remaining coordinate for choice, quotient representative selection, proof irrelevance, or host denotation, the ledger boundary is exact. □

Theorem 661.7 (Definite description boundary scoped kernel route)**.** *Every accepted* DefiniteDescriptionBoundary$^{\uparrow}$ *read factors through the displayed description row, existence row, uniqueness-boundary row, stability row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before any denoting-expression consumer can read the boundary.*

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryScopedKernelRoute`

Proof. Begin with the carrier of Definition 661.1. It displays exactly the finite rows D, E, U, S, H, C, P, N and contains no coordinate for a host denotation, choice function, quotient representative, or proof-irrelevance shortcut.

Apply Definition 661.2 to expose the local NameCert surface. The existence route of Definition 661.3 and the uniqueness route of Definition 661.4 force any witness-facing read through E and U before the description can be consumed as denoting.

Apply Definition 661.5 for componentwise hsame transport and displayed replay through S, H, C. Finally, Definition 661.6 binds the replay to P, N and blocks every non-displayed denoting-expression route. □

Lean4 ✓ `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryKernelDependencyBinding`

Theorem 661.8 (DefiniteDescriptionBoundary public interface)**.** *Every public* DefiniteDescriptionBoundary$^{\uparrow}$ *consumer factors through the displayed existence route, uniqueness boundary, stability transport, ledger exhaustion, and kernel dependency binding rows before it may read a denoting-expression boundary.*

Proof. Start with an accepted packet $B = (D, E, U, S, H, C, P, N)$. The scoped kernel route forces the consumer through the description row, existence row, uniqueness-boundary row, stability row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

The existence and uniqueness routes bind all witness-facing reads to E and U. The stability route binds transported reads to S, H, C, while ledger exhaustion removes every host-selected witness, quotient representative, proof-irrelevance shortcut, and non-displayed denoting term.

Thus the public interface is not a separate operation. It is the finite dependency surface already certified by the displayed rows and by the kernel dependency binding theorem. □

Closure status for NameCert$_{\text{DefiniteDescriptionBoundary}^{\uparrow}}$.

Constructive story: A DefiniteDescriptionBoundary$^{\uparrow}$ packet is generated from a description row, an existence witness row, a uniqueness-boundary row, a stability row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DefiniteDescriptionBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 661.1, Definition 661.2, Definition 661.3, Definition 661.4, Definition 661.5, Definition 661.6, Definition 661.7, and Definition 661.8 specify the finite description-expansion carrier, its ledger boundary, the kernel dependency route, and the public interface every denoting-expression consumer must read.

Formal verification: FormalStatus(NameCert$_{\text{DefiniteDescriptionBoundary}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DefiniteDescriptionBoundaryUp.DefiniteDescriptionBoundaryKernelDependencyBinding`

Bridge status: paperBridge

Not claimed: This chapter does not close a public definite-description operator, host choice, quotient representative selection, proof irrelevance, or a checked external bridge to any description calculus.

Upgrade path: No paper-side closure remains; bridge checking is recorded only on the verification axis for the external description-calculus boundary.

Chapter 662

A Concrete Naming Certificate for $\mathsf{RealBudgetTailMeet}^{\uparrow}$

The $\mathsf{RealBudgetTailMeet}^{\uparrow}$ surface packages the finite observation budget used by a $\mathsf{Real}^{\uparrow}$ consumer together with the shared tail threshold used by a regular-Cauchy comparison and the sealed completion handoff. It is the concrete bridge from budgeted reads and tail-meet synchronization to the $\mathsf{CauchyLimitSeal}^{\uparrow}$ route named in the real-completeness vision.

The dependency path is the finite observation budget of chapter 611, the shared-tail threshold packet of chapter 560, and the completion-facing seal handoff of chapter 386; the local carrier only synchronizes these sibling rows.

Definition 662.1 (RealBudgetTailMeet carrier)**.** *A* $\mathsf{RealBudgetTailMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, T, L, H, C, P, N)$$

with the following displayed rows:

1. *B is an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet naming the finite precision request, selected real-facing window, regular source readback, and dyadic observation budget;*
2. *T is an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet naming the shared tail threshold, paired regular-Cauchy source rows, paired schedules, comparison row, and threshold ledger;*
3. *L is an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff whose source family, scheduled window, observation ledger, completion diagonal row, and* $\mathsf{Real}^{\uparrow}$ *seal consume the same budget and shared-threshold rows;*
4. *H stores the displayed* hsame *transports relating the budget rows, shared-tail rows, and seal-handoff rows componentwise;*
5. *C stores the* Cont *routes through which downstream* $\mathsf{Real}^{\uparrow}$ *and completion consumers read the budget, tail meet, and seal handoff;*
6. *P is the* Pkg *provenance over* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBudgetTailMeet}^{\uparrow}}$ *row.*

The classifier compares carriers only by the visible rows B, T, L, H, C, P, N. It has no coordinate for quotient streams, selected host limits, hidden tail oracles, host real equality, trichotomy, or ambient completeness.

Theorem 662.2 (RealBudgetTailMeet NameCert obligations)**.** *For the carrier of Definition 662.1, the local* $\mathsf{NameCert}_{\mathsf{RealBudgetTailMeet}^{\uparrow}}$ *surface consists of carrier admission for* $R = (B, T, L, H, C, P, N)$*, classifier stability under the displayed* hsame *rows in* H*, budget exactness for the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row* B*, shared-threshold exactness for the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row* T*, seal-route exactness for the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L*, and non-escape through* C, P, N*.*

Lean4 ✓ `BEDC.Derived.RealBudgetTailMeetUp.RealBudgetTailMeetTasteGate single carrier alignment`

Proof. Carrier admission is the finite packet in Definition 662.1. The rows B, T, L, H, C, P, N are the complete public inventory.

Classifier stability is componentwise. Transport may move the budget row, the shared tail-meet row, the seal row, and the visible route rows only by the stored hsame comparisons and the same continuation routes.

The exactness clauses are projections. The budget row supplies the finite observation budget, the tail-meet row supplies the shared threshold, and the seal row supplies the completion-facing handoff. Since no other coordinate exists, the local certificate cannot export any hidden tail oracle, quotient stream, selected limit, or ambient completeness datum. □

Theorem 662.3 (RealBudgetTailMeet budget projection)**.** *Every accepted* $\mathsf{RealBudgetTailMeet}^{\uparrow}$ *carrier projects to the same finite observation budget used by its* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff. More precisely, if*

$$R = (B, T, L, H, C, P, N),$$

then every $\mathsf{Real}^{\uparrow}$ *consumer read through* R *first reads the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row* B*, transports that budget through* H *and* C*, and only then reaches the shared threshold packet* T *and the seal handoff* L*.*

Proof. Project the carrier and read the budget coordinate B. It records the precision request, finite window, regular source readback, and dyadic observation ledger available to the consumer.

Apply the $\mathsf{RealObservationBudget}^{\uparrow}$ exactness surface through the displayed continuation rows. The consumer route must carry the same budget rows through H and C before any downstream handoff can be read.

The remaining coordinates T and L are attached to that route by the carrier and the package provenance. They therefore consume the projected budget rather than an independent precision source. □

Theorem 662.4 (RealBudgetTailMeet seal handoff)**.** *Let* $R = (B, T, L, H, C, P, N)$ *be an accepted* $\mathsf{RealBudgetTailMeet}^{\uparrow}$ *carrier. The completion-facing handoff from* R *to* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is exactly the row* L *together with the budget projection from* B*, the shared threshold from* T*, and the displayed transports and routes in* H, C, P, N*. Thus the route to the* $\mathsf{Real}^{\uparrow}$ *seal factors through a finite observation budget and one shared regular-Cauchy tail threshold before the seal row is read.*

Proof. Begin with the budget projection of Definition 662.3. It fixes the consumer precision, selected window, source readback, and dyadic ledger inside B.

Use the shared-threshold surface of $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ for T. The two regular-Cauchy tails are synchronized by the displayed threshold row and by the comparison ledger already attached to the same budget route.

Finally read L as the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff. Its finite-window route consumes the same regular source, scheduled window, dyadic observation ledger, completion diagonal row, and $\mathsf{Real}^{\uparrow}$ seal, with H, C, P, N recording the visible transport and provenance. No separate completion channel is introduced. □

Theorem 662.5 (RealBudgetTailMeet non-escape boundary)**.** *An accepted* $\mathsf{RealBudgetTailMeet}^{\uparrow}$ *carrier exports only the finite rows* B, T, L, H, C, P, N *and their displayed* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *dependency surfaces. It exports no quotient stream, selected host limit, hidden tail oracle, host real equality, trichotomy row, countable-choice principle, or ambient real-completeness theorem.*

Proof. Inspect the carrier coordinates. The public rows are exactly the budget packet, tail-meet packet, seal handoff, transports, continuation routes, package provenance, and local naming row.

The budget coordinate contains only finite observation data, the tail-meet coordinate contains only the shared threshold and comparison rows, and the seal coordinate contains only the finite completion handoff to $\mathsf{Real}^{\uparrow}$.

Since quotient streams, selected host limits, hidden tail oracles, host real equality, trichotomy, choice, and ambient completeness have no coordinate in R, a downstream consumer cannot read any of them from the $\mathsf{RealBudgetTailMeet}^{\uparrow}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBudgetTailMeet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealBudgetTailMeet}^{\uparrow}$ packet is generated from a $\mathsf{RealObservationBudget}^{\uparrow}$ finite budget, a $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold packet, a $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBudgetTailMeet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 662.1, Definition 662.2, Definition 662.3, Definition 662.4, and Definition 662.5 bind the finite budget-plus-tail-meet carrier, local naming obligations, budget projection, seal handoff, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBudgetTailMeet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completion, selected host limits, uniqueness of limits, trichotomy, ambient real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, budget projection, shared-threshold exactness, seal handoff, and non-escape for the same finite packet.

Chapter 663

A Concrete Naming Certificate for $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$

The cofinal-limit packet consumes chapter 126, chapter 301, and chapter 38 through its finite window, regular-rational readback, and $\mathsf{Real}^{\uparrow}$ seal rows. Thus this chapter is not an isolated limit surface: it is the finite handoff across those sibling certificates that keeps the Real-and-Completion route visible inside one displayed carrier.

The $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ surface records the finite cofinal-window route by which regular Cauchy data reaches a $\mathsf{Real}^{\uparrow}$ seal. It packages dyadic tolerance rows, StreamName windows, regular rational sequence rows, seal rows, transports, continuations, provenance, and local naming in one visible packet. The packet is a limit-facing certificate; it is not a quotient of streams, not host equality of completed reals, and not an invocation of ambient completeness.

Definition 663.1 (RegularCauchyCofinalLimit carrier)**.** *A* $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, W, E, R, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family row of* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy names being compared by cofinal windows;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite observations read from F;*
3. *W is the cofinal-window ledger assigning each displayed precision request to a finite schedule window in S;*
4. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and error ledger attached to those windows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the dyadic tolerance ledger has been fixed;*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed regular Cauchy readback;*
7. *H records componentwise* hsame *transports for F, S, W, E, R, L and the structural rows;*
8. *C records the displayed* Cont *routes from F through S, W, E, R and then into L;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalLimit}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows $F, S, W, E, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, hidden limit selection, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCofinalLimitUp.RegularCauchyCofinalLimitTasteGate single carrier alignment`

Theorem 663.2 (RegularCauchyCofinalLimit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalLimit}^{\uparrow}}$ *surface for Definition 663.1 consists of five finite obligations:*

1. *carrier exposure, requiring every accepted object to display the packet $K = (F, S, W, E, R, L, H, C, P, N)$;*
2. *cofinal-window exactness, requiring every precision demand consumed by the limit route to pass through the displayed window ledger W over the schedule row S;*
3. *dyadic tolerance readback, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger E to be read only from the windows named by W and the regular Cauchy family F;*
4. *seal factorization, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the* $\mathsf{Real}^{\uparrow}$ *seal row L to be reached only through F, S, W, E, H, C, P, N;*
5. *non-escape, requiring the packet to export no quotient stream equality, host real equality, hidden limit selector, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier exposure is projection from the displayed tuple in Definition 663.1. The family, schedule, window, tolerance, readback, seal, transport, continuation, provenance, and local naming rows are all coordinates of one finite BHist packet.

Cofinal-window exactness reads the route S, W before any tolerance row is consumed. Since W is the only cofinal-window ledger in the carrier, a precision demand cannot be redirected to an unnamed tail or to a completed stream object.

Dyadic tolerance readback then uses E together with F, S, W. The tolerance ledger is attached to the displayed windows and regular Cauchy rows, so it cannot be replaced by host real equality or a hidden completeness argument.

Finally, seal factorization and non-escape inspect the remaining rows R, L, H, C, P, N. They provide the only route into the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal, and none of the carrier coordinates names quotient stream equality, hidden limit selection, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 663.3 (RegularCauchyCofinalLimit seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ *packet factors through the finite route*

$$F, S, W, E, R, L, H, C, P, N.$$

The seal consumer first reads the regular Cauchy family row, the StreamName schedule, the cofinal-window ledger, the dyadic tolerance ledger, and the regular rational readback row. Only then may it reach the displayed $\mathsf{Real}^{\uparrow}$ *seal row L through the componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Start with an accepted packet $K = (F, S, W, E, R, L, H, C, P, N)$. The carrier exposes no seal route except the displayed continuation chain through C.

The consumer first fixes the finite observation surface. It reads F through S, then restricts the request to the cofinal windows named by W, and then reads the dyadic tolerance evidence in E from those same windows.

The route to the seal is then forced through R and L. The transports in H, continuations in C, provenance in P, and local certificate N repack the same rows, so the seal read has no independent coordinate for a selected host limit or completed-stream equality. □

Theorem 663.4 (RegularCauchyCofinalLimit non-escape)**.** *An accepted* $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ *packet can export only the displayed rows* $F, S, W, E, R, L, H, C, P, N$ *and their local classifier transports. A completion-facing consumer may use the cofinal-window ledger, dyadic tolerance ledger, regular rational readback, and* $\mathsf{Real}^{\uparrow}$ *seal exactly as named by the packet; it receives no quotient stream equality, no host real equality, no hidden limit selector, no choice-like tail, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Proof. Project the finite carrier. The coordinate list contains rows for the regular Cauchy family, schedule, cofinal windows, dyadic tolerance ledger, readback, seal, transports, continuations, provenance, and local naming.

Apply the NameCert obligations from Definition 663.2. They force every precision, tolerance, readback, and seal request through those coordinates and require non-escape for the excluded analytic objects.

Since quotient stream equality, host real equality, hidden limit selection, countable choice, and ambient completeness are not coordinates of K, neither hsame transport nor Cont replay can manufacture them as public exports. □

Theorem 663.5 (RegularCauchyCofinalLimit public export)**.** *The carrier of Definition 663.1, the NameCert obligations of Definition 663.2, the seal factorization of Definition 663.3, and the non-escape result of Definition 663.4 assemble the public* $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ *export consumed by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows.*

The export is exactly the finite cofinal-window packet $K = (F, S, W, E, R, L, H, C, P, N)$*. It does not identify streams by quotient equality, does not compare host completed reals, does not select a hidden limit, and does not claim* $\mathsf{Real}^{\uparrow}$ *completeness outside the displayed seal route.*

Proof. The carrier theorem fixes the public packet and the NameCert theorem lists its finite obligations. Together they expose the cofinal-window route, dyadic tolerance readback, regular rational handoff, seal row, transport, continuation, provenance, and local naming rows.

Seal factorization supplies the consumer route into $\mathsf{Real}^{\uparrow}$. It shows that any seal read must pass through $F, S, W, E, R, L, H, C, P, N$ in the displayed order.

Non-escape supplies the boundary condition for public use. Because the excluded quotient, host, selector, choice, and ambient-completeness objects are absent from the carrier, the public export is exactly the finite packet and no larger completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyCofinalLimit}^{\uparrow}$ packet is generated from regular Cauchy family rows, StreamName schedule rows, cofinal-window ledgers, DyadicRatCore tolerance rows, RegSeqRat readback rows, RealUp seal rows, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 663.1, Definition 663.2, Definition 663.3, Definition 663.4, and Definition 663.5 bind the finite cofinal-window route from regular Cauchy data to RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient equality of streams, host real equality, hidden limit selection, countable choice, ambient RealUp completeness, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the displayed carrier and checked theorems for cofinal-window exactness, dyadic tolerance readback, RegSeqRat handoff, RealUp seal factorization, and non-escape.

Chapter 664

A Concrete Naming Certificate for $\mathsf{ObservationTimeOrder}^{\uparrow}$

The $\mathsf{ObservationTimeOrder}^{\uparrow}$ interface realizes BEDC-internal time as a finite naming surface for monotonic record retention. It does not introduce a host clock, scalar time coordinate, global synchronization map, or reversible time primitive. It records two observation-state histories, the retained-record classifier between them, the displayed continuation route witnessing record preservation, the Gap boundary for erased-record demands, and the local NameCert row that packages the order comparison.

Definition 664.1 (Observation-time-order carrier)**.** *An* $\mathsf{ObservationTimeOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O_0, O_1, R, C, G, H, P, N)$$

with the following visible rows:

1. O_0 *is the earlier observation-state* BHist *row;*
2. O_1 *is the later observation-state* BHist *row;*
3. R *is the retained-record classifier, naming each record visible in* O_0 *as still visible in* O_1*;*
4. C *records the displayed* Cont *route that reads* O_0*, then* O_1*, then the retained-record classifier* R*;*
5. G *is the* Gap *boundary row for any demand that a record present in* O_0 *be erased, hidden, or made unavailable in* O_1*;*
6. H *records componentwise* hsame *transport for the two observation-state rows, retained-record classifier, continuation route, gap boundary, package row, and local name row;*
7. P *is the* Pkg *provenance over* $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, BHist, Gap, hsame, Cont, Pkg, *and* NameCert *source certificates;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{ObservationTimeOrder}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $O_0, O_1, R, C, G, H, P, N$*. It has no coordinate for external host time, a numeric clock value, a global simultaneity relation, or a deletion operation on observation records.*

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderTasteGate single carrier alignment`

Theorem 664.2 (ObservationTimeOrder NameCert obligations)**.** *For the carrier of Definition 664.1, the local* $\mathsf{NameCert}_{\mathsf{ObservationTimeOrder}^{\uparrow}}$ *surface has exactly five obligation rows:*

1. *carrier admission for the two observation-state rows O_0 and O_1 together with $T = (O_0, O_1, R, C, G, H, P, N)$;*
2. *retained-record classifier exactness, requiring R to expose every record present in O_0 as retained in O_1;*
3. Cont *monotonicity route, requiring the displayed route C to read O_0 before O_1 and then read R as the preservation ledger between them;*
4. Gap *boundary for erased-record demands, requiring G to classify any claimed deletion of an O_0 record from O_1 as outside the accepted order surface;*
5. *no-host-time non-escape under H, P, N, excluding scalar clocks, external simultaneity, reversible deletion, and global synchronization from the local certificate.*

Proof. Carrier admission is projection from Definition 664.1. The two observation-state rows O_0 and O_1 are displayed before any classifier or route is read.

The retained-record classifier exactness clause reads R only after both state rows are available. Thus R can name the internal order comparison precisely as record preservation from O_0 to O_1.

The monotonicity route is the continuation row C. It fixes the public read order as O_0, then O_1, then R, so a consumer cannot reverse the order or inspect a hidden deletion operation.

The erased-record boundary is the row G. Any demand that a record present in O_0 disappear from O_1 is not a successful retained-record comparison; it is recorded as a gap-side boundary for the attempted read.

Finally, H transports only the displayed coordinates, while P and N bind provenance and local naming. Since no carrier coordinate stores a scalar clock, host time, global simultaneity relation, reversible deletion map, or synchronization frame, the certificate cannot export one. □

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderNameCertObligations`

Theorem 664.3 (ObservationTimeOrder retained-record exactness)**.** *For an accepted* ObservationTimeOrder$^{\uparrow}$ *packet*

$$T = (O_0, O_1, R, C, G, H, P, N),$$

the internal time comparison from O_0 to O_1 is exactly the retained-record classifier R read through the displayed route C. A public consumer may conclude that O_0 precedes O_1 only by reading the records of O_0, reading the records of O_1, and then reading the preservation ledger R.

No additional row supplies a numeric time value, a host clock, a global simultaneity relation, or a second comparison independent of retained records.

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderRetainedRecord exactness`

Proof. Project the accepted packet. The comparison data are the two observation-state rows O_0, O_1 and the retained-record classifier R.

The route row C determines the public read. It exposes the source state, then the target state, and only then the preservation ledger between the two.

The remaining rows G, H, P, N account for gap boundaries, transport, provenance, and naming. They do not add a clock or a second order primitive. Hence the internal time comparison is exhausted by retained-record exactness through R. □

Theorem 664.4 (ObservationTimeOrder erasure boundary)**.** *For an accepted* ObservationTimeOrder$^{\uparrow}$ *packet, any demand that a record present in O_0 be absent from O_1 is routed to the* Gap *boundary row G. The accepted time-order surface therefore permits unrecorded future commitments only as gap-side obligations, not as deletion or reversal of records already visible in O_0.*

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderErasureBoundary gap route`

Proof. Read the carrier rows $O_0, O_1, R, C, G, H, P, N$. The retained-record classifier R is the only positive comparison row between O_0 and O_1.

If a consumer asks for an O_0 record to disappear from O_1, that request contradicts the retained-record role of R. The packet has exactly one boundary row for such a request, namely G.

Transport through H and routing through C preserve the same division between retained records and gap-side demands. The package and name rows P, N then bind the boundary to the local order certificate. □

Theorem 664.5 (ObservationTimeOrder no-host-time non-escape)**.** *Every accepted* $\mathsf{ObservationTimeOrder}^{\uparrow}$ *carrier exports only the two observation-state rows, the retained-record classifier, the displayed* Cont *route, the erased-record* Gap *boundary, componentwise* hsame *transport,* Pkg *provenance, and the local* NameCert *row. It exports no scalar host time, global synchronization map, ambient simultaneity relation, reversible time operation, or hidden deletion primitive.*

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderNo host time nonescape`

Proof. Inspect the coordinates of Definition 664.1. Every exported row is one of $O_0, O_1, R, C, G, H, P, N$.

The positive order data are O_0, O_1, R, C. The boundary and structural data are G, H, P, N, which record gap-side refusal, transport, provenance, and local naming.

No coordinate remains for host time, a global synchronization map, ambient simultaneity, time reversal, or deletion of displayed records. Therefore no accepted packet can export those objects. □

Theorem 664.6 (ObservationTimeOrder HistTimeStream dependency)**.** *The retained-record route of* $\mathsf{ObservationTimeOrder}^{\uparrow}$ *consumes the local Hist time stream of chapter 603 and the finite clock-budget boundary of chapter 665 before it can expose the order comparison row* R*.*

Lean4 ✓ `BEDC.Derived.ObservationTimeOrderUp.ObservationTimeOrderHistTimeStreamDependency`

Proof. Inspect Definition 664.1. The packet exposes the two observation rows O_0, O_1 before the retained-record classifier R, and it records the public read order through the displayed continuation row C.

The package row P already names $\mathsf{HistTimeStream}^{\uparrow}$ as source certificate data for the local history stream consumed by the two observation rows. The finite comparison remains bounded by the internal rows $O_0, O_1, R, C, G, H, P, N$, so the LocalClockBudget sibling supplies a boundary for admissible clock reads rather than a host-time coordinate.

Thus the sibling references are substantive: the packet depends on the local Hist time stream and finite clock-budget boundary to present R. Because all visible rows remain internal BHist, Cont, and Gap route rows with hsame, Pkg, and NameCert packaging, the dependency adds no scalar host time or global synchronization map. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationTimeOrder}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObservationTimeOrder}^{\uparrow}$ packet is generated from two observation-state BHist rows, a retained-record classifier, displayed continuation route, erased-record GAP boundary, componentwise hsame transport, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObservationTimeOrder}^{\uparrow}}$, obligation)

Scope closed: Definition 664.1, Definition 664.2, Definition 664.3, Definition 664.4, and Definition 664.5 close the paper-side obligation surface for the finite BEDC-internal time-order packet and its monotonic observation-state record-retention reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObservationTimeOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, host time, numeric clock semantics, global simultaneity, reversible deletion, or inter-observer synchronization.

Upgrade path: Obligation closure requires checked carrier admission, retained-record exactness, Cont monotonicity routing, GAP erasure boundary, and no-host-time non-escape for the same observation-time-order packet.

Chapter 665

A Concrete Naming Certificate for $\mathsf{LocalClockBudget}^{\uparrow}$

The $\mathsf{LocalClockBudget}^{\uparrow}$ packet realizes observer local time as a finite certificate surface. It reads a single Hist stream through bounded Cont windows, records the local unary budget for those windows, transports the same rows by hsame, and exposes a no-global-clock ledger for downstream consumers. The packet does not synchronize distinct histories, choose an ambient clock, or import a hidden limit schedule. The local stream is consumed as the time-budget face of the observer-history surface in chapter 477 and the locality-cell boundary in chapter 513.

The same finite local-window budget supplies the observer-facing boundary consumed by chapter 615 and the distinction-trace consumer of chapter 882. In both cases the dependency is the carrier window row W together with the ledger-boundary row that prevents a downstream observer packet from replacing local time by a global clock.

Definition 665.1 (Local clock budget carrier)**.** *A* $\mathsf{LocalClockBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (H, T, W, B, L, Q, P, N)$$

with the following displayed rows:

1. *H is the observed* Hist *row whose local time stream is being budgeted;*
2. *T is the finite local-time-stream row, recording the* Cont *applications from* e *to the displayed history surface;*
3. *W is the finite family of* Cont*-step window rows selected for the budget;*
4. *B is the unary budget row that counts the admitted local steps and window endpoints;*
5. *L records componentwise* hsame *transport for the history row, local stream row, window rows, budget row, route row, package row, and local naming row;*
6. *Q records the displayed* Cont *routes for history projection, stream readback, window inspection, budget readback, and consumer handoff;*
7. *P is the* Pkg *provenance over* Hist, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *local time streams, and no-global-clock ledgers;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for a global clock, inter-history synchronization value, host time parameter, limit scheduler, or observer-substance distinct from the recorded history.

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetCarrier transport window surface`

Theorem 665.2 (Local clock budget window totality)**.** *Every accepted* $\mathsf{LocalClockBudget}^{\uparrow}$ *carrier exposes a finite local time stream and a finite covering family of* Cont*-step windows. Each window is selected from the row T of local* Cont *applications, counted by the unary budget row B, and replayed through the displayed route row Q inside the same packet*

$$C = (H, T, W, B, L, Q, P, N).$$

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetWindow totality`

Proof. Project the carrier of Definition 665.1. The rows H, T, W, B, L, Q, P, N display the observed history, its local stream, finite windows, unary budget, transport, routes, provenance, and local name.

The local stream row T is read before the window family W. Therefore each admitted window is a finite part of the displayed Cont route from e toward the history row H.

The unary row B counts the selected windows and endpoints, while Q replays their inspection for consumers. Since the carrier has no unbounded schedule or ambient time coordinate, every admitted window is exhausted by the finite packet. □

Theorem 665.3 (Local clock budget route determinacy)**.** *Let two accepted* $\mathsf{LocalClockBudget}^{\uparrow}$ *consumer routes display the same rows H, T, W, B, L, Q, P, N. Then they determine the same local-time-budget surface up to the componentwise* hsame *transports recorded in L. No second route may distinguish the same local stream by appealing to a global clock, a host time parameter, or an inter-history synchronization value.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetRoute determinacy`

Proof. Start with the displayed history and stream rows. The rows H and T fix the local source surface that any consumer route may inspect.

Apply window totality. Definition 665.2 binds the window family W and budget row B to the same finite Cont-step stream and routes their readback through Q.

Finally use the transport row L. Componentwise hsame transport preserves the displayed history, stream, windows, budget, provenance, and local name, while the carrier inventory excludes every global or host-side clock coordinate that could separate the two routes. □

Theorem 665.4 (Local clock budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$ *surface for Definition 665.1 consists exactly of carrier admission for $C = (H, T, W, B, L, Q, P, N)$, observed-history projection, local-time-stream exposure, finite* Cont*-window totality, unary budget readback, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, no-global-clock ledger exclusion, and the local naming row.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetNameCert obligations`

Proof. Carrier admission is projection from the displayed packet. The rows H, T, W, B provide the observed history, local stream, finite windows, and unary budget.

The window-totality theorem supplies the finite coverage clause, and route determinacy supplies the consumer-facing uniqueness of the budget surface up to the displayed hsame transports.

The remaining rows L, Q, P, N give transport, replay, provenance, and local naming. Since the carrier has no coordinate for a global clock, inter-history synchronization value, host time parameter, limit scheduler, or separate observer-substance, the listed rows exhaust the local certificate. □

Theorem 665.5 (Local clock budget observer handoff)**.** *Every downstream observer-facing consumer of an accepted* $\mathsf{LocalClockBudget}^{\uparrow}$ *packet receives only the finite rows*

$$H, T, W, B, L, Q, P, N.$$

The consumer may inspect the observed history, its local time stream, the selected Cont *windows, the unary budget, componentwise* hsame *transports, displayed replay routes, provenance, and local naming. It may not extract a global clock, a synchronized comparison between distinct histories, an ambient time parameter, a hidden limit schedule, or an observer-substance beyond* Hist*.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetCarrier observer handoff`

Proof. Project the accepted carrier. The only consumer-visible coordinates are H, T, W, B, L, Q, P, N.

Use the local NameCert obligations to order the handoff through observed-history projection, stream exposure, window totality, budget readback, transport, replay, provenance, and local naming.

Route determinacy then rules out alternate clock readings of the same packet. The no-global-clock exclusions are carrier exclusions: none of the forbidden objects appears as a coordinate, so no consumer can project it from the handoff. □

Theorem 665.6 (Local clock budget carrier scope)**.** *For an accepted* $\mathsf{LocalClockBudget}^{\uparrow}$ *packet*

$$C = (H, T, W, B, L, Q, P, N),$$

the carrier scope is anchored in the single displayed BHist *history row* H *and its finite local* Cont *stream* T*. The scope contains exactly the window family* W*, unary budget row* B*, componentwise* hsame *transport row* L*, displayed route row* Q*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$ *row* N*, and it contains no global clock, host time parameter, inter-history synchronization coordinate, limit scheduler, or observer-substance outside* Hist*.*

Proof. Project the carrier of Definition 665.1. The tuple $C = (H, T, W, B, L, Q, P, N)$ displays the whole accepted packet.

Use window totality. Definition 665.2 anchors the finite window family W and the unary budget row B in the local Cont stream T over the history row H.

The carrier inventory excludes every global or host-side clock coordinate. Therefore the carrier scope is exactly the displayed BHist-anchored packet. □

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetCarrier scope`

Theorem 665.7 (Local clock budget classifier stability)**.** *Let* $C = (H, T, W, B, L, Q, P, N)$ *be an accepted* $\mathsf{LocalClockBudget}^{\uparrow}$ *packet. Transporting* H, T, W, B, P, N *componentwise by the* hsame *rows recorded in* L *and then replaying the displayed* Cont *routes in* Q *preserves the local-clock-budget classifier. In particular, no transported packet can be separated by a global clock, host time parameter, inter-history synchronization value, limit scheduler, or observer-substance beyond* Hist*.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetClassifier stability`

Proof. Start from the displayed transport row L. Its entries move the history row, local stream row, window family, budget row, provenance row, and local name componentwise to the common hsame surface.

Replay the routes in Q. Window totality binds the transported windows and budget to the same finite local stream, and route determinacy identifies any two consumer routes that display the same rows up to those transports.

The no-global-clock exclusions are part of the carrier inventory. With no forbidden coordinate available, the classifier remains stable after the recorded transport and replay. □

Theorem 665.8 (Local clock budget ledger boundary)**.** *Every ledger read exported by an accepted* LocalClockBudget$^{\uparrow}$ *packet*

$$C = (H, T, W, B, L, Q, P, N)$$

is bounded by the displayed BHist*-anchored local stream, finite* Cont *windows, unary budget, transport, route, provenance, and local naming rows. The ledger may be handed to observer-facing consumers through Definition 665.5, but it cannot export a global clock, host time parameter, inter-history synchronization value, hidden limit schedule, or observer-substance outside* Hist*.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetLedger boundary`

Proof. Read the ledger from the carrier. The only budget-bearing rows are the local stream T, window family W, unary budget B, transport row L, route row Q, provenance row P, and local name N over the observed history row H.

Apply window totality and route determinacy. These theorems force ledger reads through the finite windows and make alternate readings of the same rows agree up to componentwise hsame transport.

Finally use observer handoff. Definition 665.5 exposes exactly the same rows to downstream consumers, while the carrier exclusions rule out every global-clock or host-time escape. □

Theorem 665.9 (Local clock budget public export)**.** *Every observer-facing consumer of an accepted* LocalClockBudget$^{\uparrow}$ *packet receives the finite packet*

$$H, T, W, B, L, Q, P, N$$

as the public LocalClockBudget$^{\uparrow}$ *export through window totality, route determinacy, observer handoff, carrier scope, classifier stability, and ledger boundary. The export may expose the observed history, local time stream, selected* Cont *windows, unary budget, componentwise* hsame *transport, displayed route, provenance, and local name, but it cannot read a global clock, host time parameter, inter-history synchronization coordinate, limit scheduler, or observer-substance outside* Hist*.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetPublic export`

Proof. Project the accepted carrier through Definition 665.1. The public packet is the finite tuple H, T, W, B, L, Q, P, N.

Apply window totality and route determinacy. Definition 665.2 binds W and B to the displayed local stream, and Definition 665.3 blocks any second route through a global or host-side clock.

Use observer handoff and carrier scope next. Definition 665.5 gives the downstream consumer exactly the same rows, while Definition 665.6 keeps the export anchored in one displayed history row and its finite local stream.

Finally apply classifier stability and ledger boundary. Definition 665.7 preserves the packet under recorded transport and replay, and Definition 665.8 excludes global clocks, host time parameters, inter-history synchronization coordinates, limit schedulers, and observer-substance from the public export. □

Theorem 665.10 (Local clock budget mature treatment)**.** *The* $\mathsf{LocalClockBudget}^{\uparrow}$ *public export consumes exactly the finite local-window budget needed by the observer-history identity of chapter 477, the observer locality cell of chapter 513, the observer subject boundary of chapter 615, and the observer distinction trace of chapter 882. Each consumer reads only the displayed rows* H, T, W, B, L, Q, P, N*, so the shared observer route has no remaining paper-side clock coordinate, host time parameter, inter-history synchronization value, hidden limit schedule, or observer-substance outside* Hist*.*

Lean4 ✓ `BEDC.Derived.LocalClockBudgetUp.LocalClockBudgetMature treatment`

Proof. Start with the public export theorem. Definition 665.9 gives the downstream surface as the finite packet H, T, W, B, L, Q, P, N.

The four observer-facing consumers named in the statement use the same packet from different boundaries. The observer-history identity reads the history row and local stream, the locality cell reads the selected finite windows, the subject boundary reads the budgeted handoff, and the distinction trace reads the ledger boundary.

Apply the carrier scope, classifier stability, and ledger boundary theorems. These force all four reads through the same local-window packet and exclude every global or host-side clock coordinate. Thus the finite local-window budget is exhausted at the mature paper surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocalClockBudget}^{\uparrow}$ packet is generated from one observed Hist row, its finite local Cont stream, finite step-window rows, a unary budget row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 665.1, Definition 665.6, Definition 665.7, Definition 665.8, Definition 665.5, Definition 665.9, and Definition 665.10 close the displayed BHist, hsame, Cont, Pkg, NameCert, exported public packet, and observer-facing local-window budget for the finite local-clock-budget surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocalClockBudget}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not close global clocks, synchronized time across distinct histories, host time parameters, hidden limit schedules, observer-substance beyond Hist, Lean verification, or bridge checking.

Upgrade path: Bridge verification requires a checked standard bridge theorem for the same finite local-clock-budget packet.

Chapter 666

A Concrete Naming Certificate for $\mathsf{SupplyKindRouter}^{\uparrow}$

The $\mathsf{SupplyKindRouter}^{\uparrow}$ surface is the concrete classifier router for the seven T-supply socket kinds of chapter 9394. It refines the ledger surfaces by forcing every consumer to pass through one displayed kind row before reading the requested-shape and refusal rows attached to that kind. The router is a finite packet over visible BEDC rows; it does not supply the requested content behind the socket.

Definition 666.1 (SupplyKindRouter carrier)**.** *A* $\mathsf{SupplyKindRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, K, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* Gap *socket site at which the requested supply would be consumed;*
2. *K is exactly one of the seven supply-kind rows: free-floating induction, unwitnessed witness, extensional identification, causal continuity, inter-history coordination, inscription novelty, or self-reference closure;*
3. *R is the requested-shape row for the missing discharge named at S;*
4. *A is the audit-gate and refusal row attached to the chosen kind K;*
5. *H records componentwise* hsame *transport for the socket site, kind row, requested-shape row, audit-gate row, provenance row, and local naming row;*
6. *C records the displayed* Cont *routes by which consumers read the site, choose the kind row, inspect the requested shape, and return the refusal row;*
7. *P is the* Pkg *provenance over* BHist, Gap, *the seven supply-kind rows, requested-shape rows, audit gates,* hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SupplyKindRouter}^{\uparrow}}$ *row.*

The carrier exports no row for unrestricted induction, an unseen witness, quotient equality, empirical persistence, shared host history, an inscription source, or truth-total self-validation.

Theorem 666.2 (SupplyKindRouter NameCert obligations)**.** *For the carrier of Definition 666.1, the local* $\mathsf{NameCert}_{\mathsf{SupplyKindRouter}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$U = (S, K, R, A, H, C, P, N),$$

socket-site exposure through S, one-kind classifier exposure through K, requested-shape accountability through R, audit-gate and refusal routing through A, componentwise transport through H, continuation locality through C, package provenance through P, local naming through N, and non-discharge of the requested supply.

Lean4 ✓ `BEDC.Derived.SupplyKindRouterUp.SupplyKindRouterTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 666.1. The packet exposes the socket site, one supply-kind row, the requested-shape row, the refusal row, transport, continuation, provenance, and local naming data as its complete visible inventory.

The classifier obligation is the row K. Since K is required to be exactly one of the seven displayed kinds, every accepted read of the router must first choose that finite row before the requested shape or refusal row is available.

The routing obligations are the rows A, H, C, P, N. They preserve the same socket boundary through audit, transport, continuation, provenance, and local naming, and none of those rows contains the requested discharge itself. □

Theorem 666.3 (SupplyKindRouter kind exhaustion)**.** *Every accepted* $\mathsf{SupplyKindRouter}^{\uparrow}$ *packet routes its socket through exactly one of the seven displayed supply kinds before any consumer reads the corresponding requested-shape row or refusal row. The router has no eighth kind, no default catch-all branch, and no route from the socket site S to a ledger read that bypasses the classifier row K.*

Proof. Project the accepted packet $U = (S, K, R, A, H, C, P, N)$. The only kind coordinate is K, and the carrier definition restricts that coordinate to the seven rows named in section 9394.11.

Now apply the NameCert obligations of Definition 666.2. They place requested-shape accountability and audit refusal downstream of the one-kind classifier exposure, so a consumer cannot reach R or A without passing through K.

The absence of hidden branches follows from the carrier inventory. No coordinate names an additional kind, a default route, or a ledger read outside the finite classifier. □

Theorem 666.4 (SupplyKindRouter apophatic socket dependency)**.** *Every accepted* $\mathsf{SupplyKindRouter}^{\uparrow}$ *packet consumes the apophatic named-boundary surface of chapter 543 when its socket site S and requested-shape row R are read. The router refines that sibling boundary by choosing exactly one supply-kind row K before the refusal row A, and it does not create the requested supply.*

Proof. Project the carrier of Definition 666.1. The socket site S and requested-shape row R are named boundary data, so their concrete sibling dependency is the ApophaticName surface that records a named boundary without supplying the requested content.

Apply Definition 666.3. It forces exactly one classifier row K to be exposed before a consumer reaches the requested-shape row or refusal row, so the router only refines the sibling boundary by kind selection.

The refusal row A then records the failed supply route. Because the carrier has no coordinate for a supplied witness, quotient equality, unrestricted induction principle, or hidden source object, the requested supply is not created by the router. □

Theorem 666.5 (SupplyKindRouter non-escape)**.** *Every consumer of an accepted* $\mathsf{SupplyKindRouter}^{\uparrow}$ *packet reads only*

$$S, K, R, A, H, C, P, N.$$

The consumer may learn which one of the seven socket kinds governs the site, which requested shape is being refused, and which audit route records the refusal. It receives no unrestricted induction principle, selected witness, quotient equality, empirical persistence law, shared host history, inscription source, truth-total self-description, hidden source object, or supplied T*-content.*

Proof. Start from the carrier and NameCert obligation surface. Together they list the complete router inventory and force every read of the requested-shape and refusal rows through the displayed classifier row.

Use kind exhaustion next. Since the accepted route has exactly one of the seven kinds and no hidden branch, the consumer has no alternate path from the socket site to a ledger read.

Finally inspect the packet coordinates. The rows H, C, P, N transport, replay, cite, and name the same finite route; they do not add the missing supply. Hence all refused objects remain outside the router. □

The audit-gate row A is the refusal input consumed by the $\mathsf{AuditGateComposition}^{\uparrow}$ packet of chapter 696 when two routed refusals share a boundary. Thus router classification fixes the kind and requested-shape rows before composed refusal traces may read a common audit boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{SupplyKindRouter}\uparrow}$**.**

Constructive story: A $\mathsf{SupplyKindRouter}^{\uparrow}$ packet is generated from a Gap socket site, exactly one of the seven supply-kind rows, a requested-shape row, an audit-gate refusal row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupplyKindRouter}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 666.1, Definition 666.2, Definition 666.3, Definition 666.4, Definition 666.5, and

Lean4 ✓ `BEDC.Derived.SupplyKindRouterUp.taste gate`

bind the finite classifier router, apophatic socket dependency, seven-kind coverage, requested-shape routing, audit refusal, transport, continuation, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupplyKindRouter}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SupplyKindRouterUp.SupplyKindRouterTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close any Gap discharge, unrestricted induction, hidden witness selection, quotient equality, empirical persistence, inter-history recognition, inscription-source explanation, self-reference closure, or a public standard bridge.

Upgrade path: No further paper-axis upgrade is pending because the chapter is at mature closure; verification and bridge checking remain independent axes.

Chapter 667

A Concrete Naming Certificate for $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$

The $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ packet realizes the Type VII reflection socket as a finite NameCert surface. It records the structural self-description a system may expose, the requested truth-total branch it must refuse, the certificate index at which the request would be consumed, and the audit rows that keep structural self-description separate from internal truth classification.

It refines the truth-total obstruction already packaged by chapter 472. The $\mathsf{TruthTotalReflection}^{\uparrow}$ packet records the finite attempted truth-total classifier boundary, while $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ records the certificate-indexed branch query and refusal surface for self-reference.

Definition 667.1 (Self-reference truth-branch carrier)**.** *A* $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$B = (S, D, Q, F, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the structural self-description row, naming the object-language syntax, certificate grammar, and replay surface being described;*
2. *D is the diagonal request row, naming the self-application route by which the description would be read back against itself;*
3. *Q is the certificate-indexed truth-branch query, naming the finite certificate row whose truth-total classification is requested;*
4. *F is the forbidden-branch row, recording that a certificate-indexed internal truth predicate or runtime-total validator is not a carrier coordinate;*
5. *R is the socket report row separating admitted structural description from refused truth-totalization;*
6. *H records componentwise* hsame *transport for structural descriptions, diagonal requests, branch queries, refusal rows, socket reports, replay routes, provenance, and local names;*
7. *C records the displayed* Cont *replay from structural description through diagonal request, branch query, refusal exposure, and socket report;*
8. *P is the* Pkg *provenance over* BHist*, structural syntax description, certificate indices, Type VII socket reports,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *source rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SelfReferenceTruthBranch}^{\uparrow}}$ *row.*

The carrier has no coordinate for an internal predicate classifying all sentences of the described system, no runtime validator contained in the object language it validates, no branch that converts a certificate index into its own truth token, and no collapse of structural self-description into truth-total reflection.

Theorem 667.2 (Self-reference truth-branch NameCert obligations)**.** *For every* $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ *carrier*

$$B = (S, D, Q, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SelfReferenceTruthBranch}^{\uparrow}}$ *surface consists exactly of structural self-description exposure through* S*, diagonal-request exposure through* D*, certificate-indexed query exposure through* Q*, forbidden-branch exposure through* F*, socket-report exposure through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SelfReferenceTruthBranchUp.SelfReferenceTruthBranchTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 667.1. The finite tuple lists the structural description, diagonal request, branch query, refusal row, socket report, transport row, replay row, provenance row, and local name.

The Type VII obligation is the separation between S and Q, F, R. The row S may describe syntax, certificate grammar, and replay shape, while Q records the requested truth-total branch and F, R record the refusal and report that keep the branch outside the carrier.

Transport and replay are confined to H and C, with provenance and local naming confined to P and N. Thus every consumer read passes through the displayed rows before it can discuss the self-reference socket.

No remaining coordinate can act as an internal truth predicate, a runtime-total validator, or a certificate-indexed truth branch. The local $\mathsf{NameCert}$ obligations are therefore exhausted by the stated finite surface. □

Theorem 667.3 (Self-reference truth-branch non-escape)**.** *Every accepted* $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ *packet exports Type VII reflection information only through the finite rows*

$$S, D, Q, F, R, H, C, P, N.$$

It exports structural self-description, diagonal-request reporting, forbidden-branch reporting, componentwise transport, replay provenance, and local naming, but it exports no internal truth predicate for its own sentences, no runtime-total self-validator, no certificate-indexed truth token, and no route that treats structural self-description as truth-total reflection.

Proof. Project the accepted packet as $B = (S, D, Q, F, R, H, C, P, N)$. The carrier definition lists every row available to a downstream reader.

Structural information is exhausted by S and the replay route through C. The diagonal pressure and certificate-indexed query are exhausted by D and Q, while the refusal and socket report are exhausted by F and R.

The transport and packaging side is exhausted by H, C, P, N. These rows preserve and replay the same displayed certificate surface; they do not add a second truth branch or a validator coordinate.

Since the packet has no other exported row, a consumer cannot bypass the forbidden-branch report to obtain a truth-total classifier. The only admitted public reading is the finite socket report separating structural self-description from the refused Type VII branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{SelfReferenceTruthBranch}^{\uparrow}}$.

Constructive story: A $\mathsf{SelfReferenceTruthBranch}^{\uparrow}$ packet is generated from a structural self-description row, a diagonal request row, a certificate-indexed truth-branch query, a forbidden-branch row, a socket report row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SelfReferenceTruthBranch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 667.1, Definition 667.2, and Definition 667.3 specify the finite Type VII packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SelfReferenceTruthBranch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean-side formal verification, an internal truth predicate, runtime-total self-validation, a Tarski hierarchy, or identification of structural self-description with external proof checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-query exactness, forbidden-branch determinacy, replay provenance, and consumer non-escape for the displayed packet.

Chapter 668

A Concrete Naming Certificate for $\mathsf{ObservationCoupling}^{\uparrow}$

The $\mathsf{ObservationCoupling}^{\uparrow}$ packet realizes inter-history locality as a finite two-history naming surface. It records two displayed BHist rows, paired observation routes, componentwise transports, a symmetric coupling ledger, package provenance, and a local $\mathsf{NameCert}$ row. The packet gives downstream multi-observer consumers a bounded coupling object without adding a global frame, synchronized clock, host metric, or observer-independent carrier outside the displayed histories.

As a sibling lattice row, $\mathsf{ObservationCoupling}^{\uparrow}$ is the paired-history coupling surface consumed by the inter-Hist invariant of chapter 656 and the maximum-rate chain of chapter 594. Those consumers may read the coupling ledger only after the two BHist rows, paired Cont routes, componentwise hsame transports, and symmetric ledger have been displayed.

Definition 668.1 (Observation coupling carrier)**.** *An* $\mathsf{ObservationCoupling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (h_a, h_b, r_a, r_b, H, C, L, P, N)$$

with the following visible rows:

1. h_a *and* h_b *are the two displayed* BHist *history rows being compared;*
2. r_a *and* r_b *are paired* Cont *routes that expose the inspected observation windows from* h_a *and* h_b*;*
3. H *records componentwise* hsame *transports for both history rows, both route rows, the coupling ledger, package provenance, and the local naming row;*
4. C *records the displayed consumer routes that read the two history windows and replay the coupling comparison;*
5. L *is the symmetric coupling ledger, storing the same finite boundary when the two history slots are exchanged;*
6. P *is the* Pkg *provenance over the two* BHist *rows, paired* Cont *routes, componentwise transports, coupling ledger, and local naming row;*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{ObservationCoupling}^{\uparrow}}$ *row.*

The carrier contains no global frame, synchronized clock, ambient metric, cross-history equality oracle, host causal-rate coordinate, or observer-substance beyond the two displayed history rows.

Lean4 (definition) `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier`

Theorem 668.2 (Observation coupling locality obligations)**.** *Every accepted* $\mathsf{ObservationCoupling}^{\uparrow}$ *packet exposes inter-history comparison only through the finite rows*

$$h_a, h_b, r_a, r_b, H, C, L, P, N.$$

A positive coupling witness must first read the two displayed BHist *rows, then the paired* Cont *routes, then the componentwise* hsame *transports and symmetric ledger. No consumer may project a comparison from a global synchronization row, ambient frame, host metric, or hidden equality oracle.*

Proof. Project the carrier of Definition 668.1. The only history data are h_a and h_b, and the only observation accessors are the paired route rows r_a and r_b.

The locality obligation reads the histories through those routes before any comparison is consumed. The transport row H keeps the comparison componentwise over the same displayed histories, while C records the consumer replay paths.

The ledger row L is the only cross-history coupling row. Since the carrier has no global frame, synchronized clock, ambient metric, host causal-rate coordinate, or equality oracle, every accepted locality witness remains inside the finite packet. □

Lean4 ✓ `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier locality obligations`

Theorem 668.3 (Observation coupling symmetric rate boundary)**.** *The coupling ledger L of an accepted* $\mathsf{ObservationCoupling}^{\uparrow}$ *packet is observer-symmetric: exchanging h_a, r_a with h_b, r_b reads the same finite boundary through the transported rows H and the displayed routes C. Any maximum cross-history causal-rate consumer can receive only this symmetric ledger boundary and cannot choose a preferred observer slot, global clock, or host-side metric ratio.*

Lean4 ✓ `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier symmetric rate boundary`

Proof. Start with the displayed tuple $O = (h_a, h_b, r_a, r_b, H, C, L, P, N)$. The two observation sides are represented by matching history and route rows, and the carrier gives no distinguished anchor outside those rows.

Use the ledger row L. By definition it stores the coupling boundary in a form that is replayed after either slot exchange, with H carrying the componentwise hsame transports and C carrying the consumer routes.

A maximum-rate consumer therefore receives a symmetric finite ledger boundary rather than a host-side rate object. The absence of a preferred observer slot, global clock, and ambient metric prevents any asymmetric projection from the accepted packet. □

Theorem 668.4 (Observation coupling NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservationCoupling}^{\uparrow}}$ *surface consists exactly of carrier admission for $O = (h_a, h_b, r_a, r_b, H, C, L, P, N)$, two displayed* BHist *source rows, paired* Cont *route exposure, componentwise* hsame *stability, symmetric ledger replay,* Pkg *provenance, local naming, and no-global-frame exclusion.*

Lean4 ✓ `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 668.1. The two source rows, paired route rows, transport row, route replay row, ledger row, package row, and local naming row are all displayed coordinates.

The locality obligations supply the reading order for the two histories and their paired routes. The symmetric rate boundary supplies the slot-exchange clause for the ledger L.

The remaining obligation is exclusion. Because the carrier lists no global frame, synchronized clock, ambient metric, cross-history equality oracle, or observer-substance coordinate, the local certificate surface is exhausted by the displayed packet. □

Theorem 668.5 (Observation coupling ledger non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{ObservationCoupling}^{\uparrow}$ *packet factors through* $h_a, h_b, r_a, r_b, H, C, L, P, N$*. The packet exports no global synchronization value, no host observer frame, no ambient spacetime coordinate, no selected maximum-rate witness, no quotient equality between histories, and no causal relation outside the displayed symmetric ledger.*

Lean4 ✓ `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier ledger nonescape`

Proof. Begin with the carrier projection. The consumer-visible rows are exactly $h_a, h_b, r_a, r_b, H, C, L, P, N$.

Apply the locality obligations to force any inter-history read through the two displayed histories and paired Cont routes. Apply the NameCert obligations to keep the read attached to the componentwise transports, replay routes, package provenance, and local name.

Finally apply the symmetric boundary theorem. The only cross-history content exported to downstream rate or invariant consumers is the ledger L, so no global synchronization value, host observer frame, ambient spacetime coordinate, selected maximum-rate witness, quotient equality, or undisplayed causal relation can be projected from the packet. □

Theorem 668.6 (Observation coupling standard-bridge boundary)**.** *The standard bridge for* $\mathsf{ObservationCoupling}^{\uparrow}$ *is exactly the displayed two-history packet*

$$h_a, h_b, r_a, r_b, H, C, L, P, N.$$

It preserves the paired Cont *routes, componentwise* hsame *transport, symmetric ledger replay,* Pkg *provenance, and the local* NameCert *row. It exports no global frame, synchronized clock, host metric, quotient equality, or selected rate witness.*

Lean4 ✓ `BEDC.Derived.ObservationCouplingUp.ObservationCouplingCarrier standard bridge boundary`

Proof. Project the carrier of Definition 668.1. The bridge source contains only the two displayed histories, the paired observation routes, componentwise transport, replay, symmetric ledger, provenance, and local naming rows.

Apply the locality obligations and symmetric rate boundary, Definition 668.2 and Definition 668.3. They force bridge reads to preserve the paired Cont routes and the symmetric replay of the ledger rather than selecting a preferred observer slot.

Finally apply the NameCert obligations and ledger non-escape, Definition 668.4 and Definition 668.5. These bind H, C, L, P, N as the public certificate surface and exclude global frames, synchronized clocks, host metrics, quotient equalities, and selected rate witnesses from the bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationCoupling}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObservationCoupling}^{\uparrow}$ packet is generated from two displayed BHist rows, paired Cont observation routes, componentwise hsame transports, consumer replay routes, a symmetric coupling ledger, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObservationCoupling}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 668.1, Definition 668.2, Definition 668.3, Definition 668.4, Definition 668.5, and Definition 668.6 bind the displayed BHist rows, paired Cont routes, componentwise hsame transports, symmetric ledger, Pkg provenance, and local NameCert rows as the finite two-history coupling bridge scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObservationCoupling}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close multi-history causal semantics, numeric light-speed values, global synchronization, ambient metric geometry, Lean verification, or bridge checking.

Upgrade path: The paper theory axis is complete for the displayed two-history coupling boundary; Lean verification and bridge checking remain separate formal-axis obligations.

Chapter 669

A Concrete Naming Certificate for PaperLeanDriftWitness$^{\uparrow}$

The PaperLeanDriftWitness$^{\uparrow}$ packet names the finite paper-Lean resolution surface used by the drift audit. It records a paper marker occurrence, the normalized declaration name requested by that marker, the duplicate-label refusal row, the Lean inventory lookup row, the resolution verdict, replay provenance, and the local NameCert row in one certificate. The packet makes paper-marker resolution a concrete carrier rather than a host-side impression that a nearby Lean declaration exists. Its single-site exact-resolution verdict is a local row consumed by the audit-map dependency routing surface of chapter 839.

Definition 669.1 (Paper-Lean drift witness carrier)**.** *A* PaperLeanDriftWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, A, L, I, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the paper marker row, recording the exact manuscript site, marker kind, and source spelling of a* `\leanchecked`*,* `\leanvariant`*,* `\leanstmt`*,* `\leandef`*, or* `\leantarget` *occurrence;*
2. *A is the normalized requested name row obtained from M, including underscore normalization and the fully qualified Lean declaration path requested by the paper site;*
3. *L is the paper-label ledger row, recording the local label attached to the manuscript site and the duplicate-label refusal verdict for the full paper label inventory;*
4. *I is the Lean inventory lookup row, recording the finite declaration table against which A is checked;*
5. *R is the resolution verdict row, recording exact resolution, unresolved-name refusal, ambiguous-name refusal, or duplicate-label refusal for the same marker site;*
6. *H records componentwise* hsame *transport for markers, normalized names, label ledgers, inventory rows, verdicts, replay routes, provenance, and local names;*
7. *C records the displayed* Cont *routes by which a consumer replays marker parsing, name normalization, duplicate-label comparison, inventory lookup, and verdict exposure;*
8. *P is the* Pkg *provenance over paper markers, label ledgers, Lean declaration inventory, audit replay,* hsame*,* Cont*,* Pkg*, and* NameCert *source packets;*

9. N is the local $\mathsf{NameCert}_{\mathsf{PaperLeanDriftWitness}^{\uparrow}}$ *row.*
The carrier has no coordinate for approximate name matching, silent marker repair, duplicate-label acceptance, unresolved-name promotion, or a checked-looking paper claim without an exact Lean inventory hit.

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitnessTasteGate single carrier alignment`

Theorem 669.2 (Paper-Lean drift witness NameCert obligations)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PaperLeanDriftWitness}^{\uparrow}}$ *surface consists exactly of marker-site exposure, normalized requested-name exposure, duplicate-label refusal, Lean inventory lookup, resolution verdict exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. It does not export a separate paper authority for treating a non-resolving marker as checked.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness namecert obligations`

Proof. Carrier exposure is projection from Definition 669.1. The tuple lists the paper marker row, normalized name row, label ledger, Lean inventory lookup, verdict row, transport, replay, provenance, and local name.

The NameCert obligations are the finite audit obligations on those rows. The marker and name rows expose what the paper asked for; the label ledger and inventory row expose the two finite tables against which the request is checked; the verdict row records the admissible resolution outcome.

Transport and replay are confined to H and C, with provenance and naming confined to P and N. Since no row stores approximate matching, silent repair, duplicate-label acceptance, or unresolved-name promotion, the local certificate is exhausted by the displayed obligations. □

Theorem 669.3 (Paper-Lean drift witness resolution exactness)**.** *If an accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *packet records an exact-resolution verdict in R, then the normalized requested name A is present as the same fully qualified declaration in the Lean inventory row I, and the paper label row L is not refused as a duplicate label. The packet never identifies a marker with a neighboring declaration, a spelling variant not present in I, or a label site that the duplicate-label ledger refuses.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness resolution exactness`

Proof. Read the accepted packet through the carrier. The exact-resolution verdict is a row of R attached to the same marker M, normalized name A, label ledger L, and inventory row I.

The replay route C first normalizes the marker spelling, then compares the label row against the paper ledger, and then looks up the normalized name in the Lean declaration inventory. Exact resolution can therefore be read only after both finite comparisons have admitted the site.

Componentwise transport in H preserves the same marker, label, and inventory rows. Thus a consumer cannot replace the resolved name by a neighboring declaration, a spelling not present in I, or a label site that L refuses. □

Theorem 669.4 (Paper-Lean drift witness non-escape)**.** *An accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *packet exports no checked theorem, statement, definition, or closure-status target beyond the exact marker-resolution rows*

$$M, A, L, I, R, H, C, P, N.$$

In particular, it exports no unresolved Lean target, no duplicate paper label, no marker-kind coercion, no approximate declaration match, and no host-side permission to treat paper prose as a resolving Lean declaration.

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness nonescape`

Proof. Inspect the carrier inventory. The rows record marker parsing, requested-name normalization, label-ledger comparison, Lean inventory lookup, resolution verdict, transport, replay, provenance, and local naming.

None of these rows is a theorem proof, a definition body, a closure-status upgrade, or a host permission row. The positive content is only the exact resolution verdict attached to a marker whose normalized name occurs in the inventory and whose label row is not refused.

Therefore any unresolved target, duplicate label, marker-kind coercion, approximate match, or prose-only declaration leaves the packet. Such content must be supplied by another explicit certificate and cannot be projected from this drift witness. □

Theorem 669.5 (Paper-Lean drift witness consumer boundary)**.** *Every drift-audit consumer of an accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *packet factors through the paper marker row M, normalized requested name A, duplicate-label ledger L, Lean inventory lookup I, resolution verdict R, componentwise transport H, replay route C, provenance row P, and local* $\mathsf{NameCert}$ *row N. The consumer receives a finite NameCert surface for exact marker resolution, not a general semantic equivalence between paper claims and Lean declarations.*

Proof. Start with the carrier. The audit consumer can project only the nine displayed rows $M, A, L, I, R, H, C, P, N$.

Resolution reads pass through the normalized name and Lean inventory rows. Duplicate-label reads pass through the label ledger. Verdict reads pass through R and can be replayed only by the displayed routes in C under componentwise transport H.

The provenance row P and local name row N package this finite read as a NameCert surface. Since the carrier contains no semantic-equivalence row between paper prose and Lean theorem meaning, the consumer obtains exact marker resolution only. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness consumer boundary`

Theorem 669.6 (Paper-Lean drift witness verdict determinism)**.** *For an accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

fix the paper marker row M, normalized requested-name row A, paper-label ledger row L, and Lean inventory row I. The verdict row R exposes exactly one local outcome for that same quadruple: exact resolution, unresolved-name refusal, ambiguous-name refusal, or duplicate-label refusal. A consumer replaying the displayed route C under componentwise transport H cannot obtain a second verdict for the same M, A, L, I.

Proof. Project the accepted carrier to the rows $M, A, L, I, R, H, C, P, N$. The rows M, A, L, I determine the marker site, the normalized request, the paper-label ledger read, and the finite Lean inventory read being adjudicated.

The verdict coordinate is a single row R attached to those projected inputs. Its allowed local outcomes are the four carrier outcomes listed in Definition 669.1: exact resolution, unresolved-name refusal, ambiguous-name refusal, and duplicate-label refusal.

Replay does not introduce another decision channel. The displayed route C parses the marker, normalizes the requested name, checks the label ledger, checks the Lean inventory, and exposes the verdict row. Componentwise transport in H preserves the same inputs through that route, so replaying the same carrier cannot expose a distinct verdict for the fixed M, A, L, I. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness verdict determinism`

Theorem 669.7 (Paper-Lean drift witness duplicate-label priority)**.** *Let*

$$W = (M, A, L, I, R, H, C, P, N)$$

be an accepted $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier. If the paper-label ledger row L records duplicate-label refusal for the marker site M, then no exact-resolution verdict in R may be exported for that site, even when the normalized requested name A is present in the Lean inventory row I.*

Proof. Read the carrier in the displayed row order $M, A, L, I, R, H, C, P, N$. The label ledger row L is consulted before the verdict row is exposed, and its duplicate-label refusal is a local refusal for the same marker site M.

By Definition 669.3, exact resolution requires both an inventory hit for A in I and the absence of duplicate-label refusal in L. Thus an inventory hit alone is not sufficient for exporting an exact-resolution row.

The replay route C keeps the label-ledger comparison on the path to R, and H preserves that ledger row under transport. Therefore a duplicate-label refusal in L has priority over the inventory hit and prevents export of an exact-resolution verdict for the refused marker site. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness duplicate label priority`

Theorem 669.8 (Paper-Lean drift witness unresolved marker refusal)**.** *Let*

$$W = (M, A, L, I, R, H, C, P, N)$$

be an accepted $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier. If the normalized requested name A is absent from the Lean inventory row I, then every consumer read through R, H, C, P, N remains a refusal row and cannot be promoted to a checked marker.*

Proof. Absence of A from I blocks the exact-resolution condition. By Definition 669.3, an exact-resolution verdict can be read only when the normalized requested name is present in the Lean inventory row and the paper-label ledger does not refuse the site.

The only remaining verdict information for the same marker passes through the refusal rows of R. The route C replays the same normalization and inventory lookup, while H preserves the same normalized name and inventory row under transport.

Non-escape from Definition 669.4 prevents the packet from exporting a checked theorem, statement, definition, or closure-status target outside the exact marker-resolution rows. The consumer boundary in Definition 669.5 confines downstream reads to R, H, C, P, N, so the absent inventory hit remains a refusal and cannot become a checked marker. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitnessUnresolvedMarkerRefusal`

Theorem 669.9 (Paper-Lean drift witness resolution stability)**.** *Let*

$$W = (M, A, L, I, R, H, C, P, N)$$

be an accepted PaperLeanDriftWitness$^{\uparrow}$ *carrier with an exact-resolution verdict in R. If*

$$M\mathsf{hsame}M', \quad A\mathsf{hsame}A', \quad L\mathsf{hsame}L', \quad I\mathsf{hsame}I', \quad R\mathsf{hsame}R', \quad H\mathsf{hsame}H', \quad C\mathsf{hsame}C', \quad P\mathsf{hsame}P', \quad N\mathsf{hsa}$$

are componentwise transport rows displayed by H, then the transported packet

$$W' = (M', A', L', I', R', H', C', P', N')$$

has the same exact-resolution outcome for the transported marker site. In particular, exact resolution remains an inventory hit for A' in I' together with the absence of duplicate-label refusal in L'.

Proof. The carrier theorem exposes the nine coordinates as the complete local surface for a drift witness. The NameCert-obligation theorem identifies H as componentwise transport for marker rows, normalized names, label ledgers, inventory rows, verdicts, replay routes, provenance, and local names.

Apply the resolution-exactness theorem before transport. Exact resolution of W reads the same marker through M, the normalized request A, the duplicate-label ledger L, and the Lean inventory row I; it requires an inventory hit and no duplicate-label refusal.

The transport rows in H carry each of these coordinates to the corresponding primed row. Non-escape prevents an additional row from changing the verdict during transport, and the consumer-boundary theorem forces every downstream read of W' through the transported coordinates.

Therefore the transported verdict in R' is still exact resolution for M' precisely by the transported inventory hit for A' in I' and the transported non-refusal in L'. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness resolution stability`

Theorem 669.10 (Paper-Lean drift witness marker-kind separation)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

the marker kind recorded by M is preserved as its displayed paper-row kind throughout normalization, inventory lookup, verdict exposure, transport, replay, provenance, and local naming. A row whose source marker is `\leanchecked` *cannot be consumed as* `\leanvariant`*,* `\leanstmt`*,* `\leandef`*, or* `\leantarget`*; the same separation holds for each of the five marker kinds.*

Proof. The carrier definition stores the marker kind inside the paper marker row M. The NameCert-obligation theorem exposes this marker-site row as one of the required certificate rows and gives no separate coordinate for marker-kind conversion.

Resolution exactness then reads the normalized declaration request from A as obtained from that same marker row, and it checks A against the inventory row I under the duplicate-label ledger L. The verdict is therefore attached to the marker site and kind already exposed by M.

Non-escape rules out marker-kind coercion as exported content of the packet. The replay and transport coordinates C and H can only replay and transport the displayed rows, while the consumer-boundary theorem confines consumers to $M, A, L, I, R, H, C, P, N$.

Thus no consumer can reinterpret a row for one of `\leanchecked`, `\leanvariant`, `\leanstmt`, `\leandef`, or `\leantarget` as a row for another kind. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness marker kind separation`

Theorem 669.11 (Paper-Lean drift witness inventory verdict totality)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

each accepted verdict exposed by R is read from the displayed Lean inventory row I together with the duplicate-label ledger row L. No accepted verdict may be obtained from approximate declaration matching, prose-only promotion, or a paper marker whose duplicate-label row is refused.

Proof. The carrier definition lists the admissible resolution outcomes in the verdict row R and lists no coordinate for approximate matching or prose-only promotion. The NameCert obligations require the duplicate-label ledger, Lean inventory lookup, and verdict exposure as displayed rows of the same packet.

For the exact-resolution case, apply the resolution-exactness theorem. The verdict requires that A occurs as the same fully qualified declaration in I and that L does not refuse the marker site.

For refusal cases, the non-escape theorem prevents unresolved names, duplicate labels, marker-kind coercions, approximate matches, and prose-only declarations from leaving the exact marker-resolution surface as checked content. The consumer-boundary theorem confines every audit-map consumer to the rows $M, A, L, I, R, H, C, P, N$.

Hence every accepted verdict is total over the displayed inventory and duplicate-label ledger rows: it is either the exact inventory-and-ledger outcome or a refusal exposed by R, never an approximate or prose-only promotion. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness inventory verdict totality`

Theorem 669.12 (Paper-Lean drift witness token objectwise audit)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *packet*

$$W = (M, A, L, I, R, H, C, P, N),$$

any consumer using W to justify a `\formalstatus{...}` token or a marker-resolution synchronization must read the exact marker row M, normalized requested-name row A, duplicate-label ledger row L, Lean inventory row I, and verdict row R for that same paper site. The packet cannot convert objective Lean inventory strength into an aggregate route-level formalstatus, bridge row, theoryclosure row, or semantic equivalence between paper prose and Lean declarations.

Proof. The positive resolution case is objectwise. By resolution exactness, an exact inventory hit is attached to the same marker row M, normalized name row A, label ledger row L, and Lean inventory row I, and it is exposed only as the verdict row R for that site.

Verdict determinism and inventory verdict totality then fix the local outcome. For the fixed rows M, A, L, I, the consumer can read one displayed verdict from R: either the exact inventory-and-ledger outcome or a refusal row over the same local inventory and duplicate-label ledger.

The consumer boundary confines downstream reads to $M, A, L, I, R, H, C, P, N$. Non-escape excludes checked theorem content, closure-status targets, bridge rows, theoryclosure rows, and semantic-equivalence content not present in those rows.

Therefore a `\formalstatus{...}` token or marker-resolution synchronization justified by W is evidence about the displayed objectwise marker-resolution rows for one paper site. It is not aggregate route-level closure evidence and cannot collapse the manuscript closure axis with the Lean inventory lookup axis. □

Theorem 669.13 (Paper-Lean drift witness public verdict ledger)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

any public audit-map consumer must read the marker row M, normalized Lean name row A, paper-label ledger L, Lean inventory row I, verdict row R, transport row H, replay row C, provenance row P, and local NameCert *row N as one finite verdict ledger. The consumer cannot aggregate this ledger into theoryclosure evidence, bridge status, approximate name matching, or semantic equivalence between paper prose and Lean declarations.*

Proof. First use resolution exactness and non-escape. Resolution exactness binds the accepted verdict to the same displayed rows M, A, L, I, R, while non-escape prevents checked theorem content, closure-status targets, and semantic-equivalence content from leaving the witness packet.

Next use the consumer-boundary theorem. Every public audit-map read is confined to $M, A, L, I, R, H, C, P, N$, so transport, replay, provenance, and local naming carry the same finite verdict ledger rather than adding an outside marker row or neighbouring Lean declaration.

Verdict determinism and inventory-totality fix the ledger outcome. For the displayed marker, normalized name, ledger, and inventory rows, the verdict row is uniquely read and exhausts the accepted exact or refusal outcomes.

Finally apply token-objectwise audit. The public consumer receives an objectwise audit row for marker resolution, not theoryclosure evidence, a bridge-status row, approximate name matching, or semantic equivalence between the paper text and a Lean declaration. □

Theorem 669.14 (Paper-Lean drift witness audit-map dependency source lock)**.** *Let*

$$W = (M, A, L, I, R, H, C, P, N)$$

be an accepted PaperLeanDriftWitness$^{\uparrow}$ *carrier. Any* AuditMapDependencyWeave$^{\uparrow}$ *consumer using W as a dependency source must read the same marker row M, normalized requested-name row A, duplicate-label ledger row L, Lean inventory row I, verdict row R, componentwise transport row H, replay row C, provenance row P, and local* NameCert *row N. The consumer cannot replace those rows by approximate marker repair, an aggregate formalstatus token, a neighboring Lean name, or a route-level closure summary.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness auditmap dependency source lock`

Proof. The carrier projection exposes exactly the rows $M, A, L, I, R, H, C, P, N$. The marker row, normalized request, ledger, inventory, and verdict are therefore read from the same accepted witness rather than from a neighboring audit route.

Resolution exactness fixes the positive read. An exact verdict requires the displayed normalized name A to occur in the displayed inventory I and requires the displayed ledger L not to refuse the same marker site M.

Token objectwise audit prevents this local verdict from becoming an aggregate formalstatus route. The consumer boundary then confines every downstream AuditMapDependencyWeave$^{\uparrow}$ read to the same displayed rows, so approximate marker repair and route-level closure summaries are outside the handoff. □

Theorem 669.15 (Paper-Lean drift witness objectwise drift handoff)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, an exact verdict hands off only the objectwise marker row M, normalized Lean name row*

A, duplicate-label ledger row L, Lean inventory hit row I, and verdict row R for the same paper site. Replay, transport, provenance, and local naming may carry that row through H, C, P, N, but they cannot add a semantic-equivalence row, an approximate declaration match, or a checked-looking marker outside the exact verdict.

Proof. Resolution exactness supplies the objectwise positive case: the marker row M, normalized request A, ledger L, inventory I, and verdict R are read together for one paper site.

Duplicate-label priority keeps the ledger in the handoff. If L refuses the site, an inventory-shaped hit cannot be exported as an exact verdict, so the consumer cannot ignore the paper-label row.

Unresolved-marker refusal keeps absent inventory hits inside refusal rows. Verdict determinism and inventory verdict totality then make the exported row exhaustive over the displayed local outcomes, leaving only transport, replay, provenance, and naming through H, C, P, N. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness objectwise drift handoff`

Theorem 669.16 (Paper-Lean drift witness audit-map weave consumer exactness)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, the downstream* AuditMapDependencyWeave$^{\uparrow}$ *surface receives an exact local dependency row for marker resolution: the source-locked packet $M, A, L, I, R, H, C, P, N$ and the objectwise verdict handoff over M, A, L, I, R. It receives no semantic equivalence between paper prose and Lean declarations, no checked proof of the named Lean declaration, and no bridge or closure-axis certificate.*

Proof. First apply the audit-map dependency source lock. The dependency source is the displayed finite packet $M, A, L, I, R, H, C, P, N$, and every consumer read must factor through those rows.

Next apply the objectwise drift handoff. The exact verdict exported to the dependency weave is the local marker-resolution row over M, A, L, I, R, carried only by the displayed transport, replay, provenance, and local naming rows.

Finally use the consumer-boundary and non-escape theorems. The boundary gives exact marker resolution as dependency data, while non-escape prevents semantic equivalence, Lean proof content, bridge checking, or closure-axis certification from leaving the PaperLeanDriftWitness$^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness auditmap weave consumer exactness`

Theorem 669.17 (Paper-Lean drift witness obligation-closure package)**.** *For every accepted* PaperLeanDriftWitness *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

the obligation-closure package is exactly the conjunction of marker-site exposure, normalized requested-name exposure, duplicate-label priority, unresolved-marker refusal, marker-kind separation, inventory verdict totality, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It exports no checked theorem content, no semantic equivalence between paper claims and Lean declarations, no approximate declaration match, and no prose-only promotion.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness obligation closure package`

Proof. The carrier definition displays the nine rows $M, A, L, I, R, H, C, P, N$. The NameCert-obligation theorem identifies those rows as the marker site, normalized requested name, paper-label

ledger, Lean inventory lookup, verdict exposure, componentwise transport, replay, provenance, and local naming surface.

Resolution exactness supplies the exact inventory-and-ledger read for an accepted exact verdict. Duplicate-label priority blocks an exact-resolution export when the paper-label ledger refuses the site, while unresolved-marker refusal keeps an absent Lean inventory hit inside a refusal row.

Marker-kind separation prevents a consumer from retyping one paper marker kind as another. Inventory verdict totality then exhausts the accepted verdicts over the displayed Lean inventory row and duplicate-label ledger, ruling out approximate declaration matching and prose-only promotion.

It remains only to bind the package boundary. Componentwise hsame transport is confined to H, displayed Cont replay is confined to C, Pkg provenance is confined to P, and local naming is confined to N. By non-escape and the consumer-boundary theorem, no checked theorem content, semantic equivalence row, approximate match, or prose-only promotion can leave this finite witness packet. □

Theorem 669.18 (Paper-Lean drift witness closure-target resolution)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

any closure-status consumer or paper-marker consumer must read the exact paper marker row M, normalized Lean target row A, duplicate-label ledger row L, Lean inventory row I, and verdict row R before treating a `\leanchecked`*,* `\leanvariant`*,* `\leanstmt`*,* `\leandef`*, or* `\leantarget` *occurrence as resolving. The resolving read is accepted only as the displayed exact inventory-and-ledger verdict for that same marker site.*

Lean4✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness closure target resolution`

Proof. Begin with the carrier row order. The marker site is exposed by M, the requested Lean declaration is normalized in A, the paper-label ledger is read through L, the Lean inventory is read through I, and the verdict is exposed by R.

Resolution exactness supplies the positive case: an exact verdict requires the normalized requested name to occur in I and the label row not to be refused by L. Duplicate-label priority prevents an inventory hit from resolving a marker whose label row is refused.

Unresolved-marker refusal keeps absent inventory hits inside refusal rows, while marker-kind separation keeps each of the five marker commands in its own paper-row kind. Inventory verdict totality then exhausts the accepted outcomes over the displayed inventory and ledger rows.

Therefore a closure-status or marker consumer can treat a paper marker as resolving only after reading M, A, L, I, R for the same site and receiving the exact inventory-and-ledger verdict. □

Theorem 669.19 (Paper-Lean drift witness axis separation)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, an exact-resolution verdict certifies only the verification-axis lookup of the requested Lean declaration in the displayed inventory. It does not upgrade the paper-side closure axis, bridge status, or semantic theorem content attached to the paper claim.*

Proof. The non-escape theorem states that the packet exports no checked theorem, statement, definition, or closure-status target beyond the exact marker-resolution rows. Its positive content is the marker-resolution verdict attached to the displayed paper marker, normalized name, label ledger, and Lean inventory lookup.

The consumer-boundary theorem confines every audit consumer to $M, A, L, I, R, H, C, P, N$. Those rows include no coordinate for changing theory closure, no bridge certificate, and no semantic equivalence row between paper claims and Lean declarations.

Thus an exact-resolution verdict may certify that the requested Lean declaration is found on the verification axis. Any paper-side closure claim, bridge status, or semantic theorem content must be supplied by a separate certificate and cannot be projected from this witness. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness axis separation`

Theorem 669.20 (Paper-Lean drift witness audit-map handoff)**.** *Every audit-map dependency consumer receiving an accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier receives exactly the finite packet*

$$M, A, L, I, R, H, C, P, N.$$

It may use that packet as a row for marker-resolution dependency routing, but not as a replacement for a Lean theorem proof or for paper-side closure evidence.

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness audit map handoff`

Proof. The consumer boundary gives the audit-map consumer precisely the displayed finite rows $M, A, L, I, R, H, C, P, N$. These rows expose marker parsing, name normalization, duplicate-label comparison, inventory lookup, verdict exposure, transport, replay, provenance, and local naming.

Resolution stability permits the same exact-resolution outcome to be replayed under componentwise hsame transport of those rows. Closure-target resolution identifies the resolving read as the exact inventory-and-ledger verdict for the same marker site.

Axis separation then fixes the boundary of the handoff. The packet is a dependency-routing row for marker resolution; a Lean theorem proof and paper-side closure evidence remain separate obligations supplied outside the $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ witness. □

Theorem 669.21 (Paper-Lean drift witness audit-index readiness)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier*

$$W = (M, A, L, I, R, H, C, P, N),$$

the compact audit index for the witness is ready exactly when the same marker site exposes the exact-resolution row of M, A, I, R*, the duplicate-label priority row of* L*, and the marker-kind separation row of* M*. The resulting index entry is a finite objectwise audit row; it is not a semantic-equivalence witness, a proof of the named Lean theorem, or evidence for the paper-side closure axis.*

Proof. The exact-resolution row is supplied by Definition 669.3: the normalized requested name in A must be read against the Lean inventory I, and the verdict row R records the exact inventory-and-ledger outcome for the same marker site in M.

The ledger side is supplied by Definition 669.7. A refused duplicate-label row in L blocks the exact export even when the inventory row contains a name-shaped hit, so the audit index cannot ignore the paper-label ledger.

The marker surface is supplied by Definition 669.10. A `\leanchecked` row, a `\leanvariant` row, a `\leanstmt` row, a `\leandef` row, and a `\leantarget` row remain different paper-marker kinds.

Combining these rows gives a compact index entry over M, A, L, I, R. By Definition 669.19, that entry remains only a verification-axis lookup row and cannot be projected into semantic equivalence, checked theorem content, or closure-axis evidence. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness audit index readiness`

Theorem 669.22 (Paper-Lean drift witness verdict-ledger totality)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, the verdict ledger over the displayed rows* M, A, L, I, R *is total for audit consumption: each marker site is routed either through the exact inventory-and-ledger verdict, through duplicate-label refusal, or through unresolved-marker refusal. No fourth outcome is exported to a consumer.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness verdict ledger totality`

Proof. The total verdict split is the content of Definition 669.11. It reads the displayed marker site, normalized requested name, paper-label ledger, Lean inventory lookup, and verdict row as the only accepted inputs for a marker-resolution decision.

The positive branch is constrained by Definition 669.3; exact resolution requires the requested declaration to occur in the inventory and the same paper-label row not to be refused. This branch therefore records an objectwise inventory-and-ledger verdict, not an approximate declaration match.

The refused branches are constrained separately. Duplicate-label priority comes from Definition 669.7, while absent inventory hits are confined by Definition 669.8.

Thus every consumer-visible verdict is one of the three displayed rows. By Definition 669.17, the ledger remains inside the finite witness packet and exports no semantic theorem equivalence or closure-status promotion. □

Theorem 669.23 (Paper-Lean drift witness dependency-weave handoff)**.** *Every dependency-weave consumer receiving an accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier receives the finite audit packet*

$$M, A, L, I, R, H, C, P, N$$

as a marker-resolution dependency row. The handoff preserves exact resolution, verdict-ledger totality, token objectwise audit, axis separation, and the audit-map boundary, but it does not discharge downstream semantic equivalence, Lean proof, bridge, or closure-axis obligations.

Proof. The packet delivered to the consumer is fixed by Definition 669.20. The dependency-weave side therefore receives marker parsing, normalized-name exposure, duplicate-label comparison, inventory lookup, verdict exposure, transport, replay, provenance, and local naming as the whole finite carrier row.

Exact resolution is preserved by Definition 669.18, and the verdict ledger is total by Definition 669.22. The handoff can route these rows through neighbouring audit-map dependencies only as objectwise marker-resolution data.

Token objectwise audit is the boundary supplied by Definition 669.12: the exported row records the token object and its local audit evidence, not an ambient semantic equality between paper prose and Lean declarations.

Finally, Definition 669.19 and Definition 669.20 keep the route from proving too much. Downstream semantic equivalence, Lean proof content, bridge checking, and paper-side closure status remain separate obligations outside the $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness dependency weave handoff`

Theorem 669.24 (Paper-Lean drift witness dependency-routing scope)**.** *Let*

$$W = (M, A, L, I, R, H, C, P, N)$$

be an accepted $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier. Every* $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ *consumer using* W *as a marker-resolution dependency row must route through the displayed marker row* M*,*

normalized requested-name row A, duplicate-label ledger row L, Lean inventory row I, verdict row R, componentwise transport row H, replay row C, provenance row P, and local NameCert *row N. The route cannot replace any of those rows by an unlisted marker, neighbouring name, outside label ledger, hidden inventory, ambient verdict, or external consumer row.*

Proof. The dependency-weave handoff fixes the exported packet as $M, A, L, I, R, H, C, P, N$. It gives the audit-map consumer a dependency row for marker resolution, not a separate carrier with additional coordinates.

The audit-map handoff theorem identifies how the neighbouring consumer may read that packet. Marker parsing, name normalization, duplicate-label comparison, inventory lookup, verdict exposure, componentwise transport, replay, provenance, and local naming are the whole finite route.

The consumer boundary and non-escape rows then rule out substitutions. A consumer may transport and replay the same rows through H, C, P, N, but it cannot import a second marker, a different normalized name, another label ledger, an approximate inventory hit, or a verdict outside R. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness dependency routing scope`

Theorem 669.25 (Paper-Lean drift witness consumer-row scope)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier $W = (M, A, L, I, R, H, C, P, N)$, the marker, ledger, inventory, and verdict rows are grounded as finite* BHist *data: M and L are paper-site and paper-label ledger rows, I is the Lean declaration inventory row, and R is the exact-resolution or refusal verdict for the same marker site. Their consumer route is mediated only by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row, and it is read by chapter 839 only as a routed dependency row.*

Proof. Carrier exposure supplies the finite BHist packet. The marker row M, normalized-name row A, duplicate-label ledger L, inventory row I, verdict row R, transports H, replay routes C, provenance P, and local name N are all projected from the same accepted witness.

Resolution exactness and duplicate-label priority bind the ledger and inventory reads to the same marker site. The verdict ledger totality theorem then says the visible verdict is exactly one of the accepted exact-resolution or refusal rows for M, A, L, I.

The only consumer movement is through hsame, Cont, Pkg, and NameCert structure. Thus chapter 839 receives the row as routed dependency data and does not obtain a hidden theorem proof, erased obstruction, or closure-axis promotion from the witness. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.ConsumerRowScope.PaperLeanDriftWitness consumer row scope`

Theorem 669.26 (Paper-Lean drift witness scoped closure package)**.** *The* PaperLeanDriftWitness$^{\uparrow}$ *chapter is scoped for audit-map dependency consumption: resolution exactness fixes the inventory-and-ledger read, verdict totality exhausts the consumer-visible outcomes, axis separation prevents verification-axis rows from becoming paper-side closure evidence, and the dependency-weave handoff routes exactly the displayed packet*

$$M, A, L, I, R, H, C, P, N$$

to the named AuditMapDependencyWeave$^{\uparrow}$ *consumer surface. Within that scope, the chapter closes the paper-side dependency row for exact marker resolution, while leaving Lean verification, semantic equivalence, and bridge checking outside the witness.*

Proof. First combine resolution exactness with verdict totality. The accepted witness reads one marker site through M, A, L, I and exposes exactly the corresponding verdict row R, so the consumer-facing resolution row is objectwise fixed.

Next use axis separation. The same verdict may certify a verification-axis inventory lookup, but it cannot become a theorem proof, a semantic-equivalence row, a bridge certificate, or a paper-side closure claim outside the stated dependency scope.

Finally apply the dependency-weave handoff together with the dependency-routing and consumer-row scope theorems. The audit-map dependency consumer receives precisely $M, A, L, I, R, H, C, P, N$ through the named chapter 839 route, so the chapter supplies a scoped paper-side package for routed marker-resolution consumption and no stronger conclusion. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness scoped closure package`

Theorem 669.27 (Paper-Lean drift witness resolution consumer readiness)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, any consumer route treating the witness as an exact-resolution row must read the same paper marker row M, normalized requested-name row A, paper-label ledger L, Lean inventory row I, and verdict row R for one marker site. The consumer may replay those rows through H, C, P, N, but it cannot replace them by a neighbouring marker, a different normalized name, a different label ledger, an approximate inventory row, or a second verdict.*

Proof. The exact-resolution row is fixed by Definition 669.3. An exact verdict reads the same marker row, normalized requested name, duplicate-label ledger, Lean inventory lookup, and verdict row, and it requires the inventory hit and the absence of duplicate-label refusal on that same site.

The consumer route is bounded by Definition 669.5. Every downstream read factors through $M, A, L, I, R, H, C, P, N$, so the consumer cannot introduce a parallel marker row or a second inventory table.

Verdict determinism from Definition 669.6 and inventory totality from Definition 669.11 keep the row objectwise. Transport and replay may carry the same row through H, C, P, N, but the result remains the exact inventory-and-ledger verdict for the original M, A, L, I, R package. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness resolution consumer readiness`

Theorem 669.28 (Paper-Lean drift witness duplicate-ledger readiness)**.** *For every accepted* PaperLeanDriftWitness *carrier, duplicate-label refusal has priority along every consumer route through R, H, C, P, N. If the ledger row L refuses the marker site M, then replay, transport, provenance, and local naming may preserve that refusal, but they cannot export an exact-resolution row for that site.*

Proof. Duplicate priority is the content of Definition 669.7. A ledger refusal blocks exact resolution even when the normalized requested name has the shape of an inventory hit.

Unresolved-marker refusal from Definition 669.8 and non-escape from Definition 669.4 keep refused marker sites from leaving the packet as checked markers. The route through R, H, C, P, N therefore remains a refusal route whenever L refuses M.

The dependency-weave handoff theorem, Definition 669.23, lets a neighbouring audit-map consumer receive the finite packet only as marker-resolution dependency data. It does not create a row that can outrank the duplicate-label ledger. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness duplicate ledger readiness`

Theorem 669.29 (Paper-Lean drift witness inventory-refusal readiness)**.** *For every accepted* PaperLeanDriftWitness *carrier, if the normalized requested name A is absent from the Lean inventory row I, then every consumer-visible route through R, H, C, P, N remains an unresolved-marker refusal row. The absent inventory hit cannot be transported, replayed, or repackaged as a checked marker.*

Proof. Unresolved-marker refusal is supplied by Definition 669.8. Absence of A from I blocks the exact-resolution condition and keeps the verdict row on the refusal side for the same marker site.

Resolution exactness from Definition 669.3 and verdict determinism from Definition 669.6 prevent a consumer from reading both an absent inventory hit and an exact-resolution verdict for the fixed rows M, A, L, I.

The non-escape theorem and the dependency-weave handoff theorem confine the exported material to finite marker-resolution rows. Hence replay, transport, provenance, and local naming preserve the unresolved-marker refusal rather than converting it into checked theorem content, semantic equivalence, bridge evidence, or closure-axis evidence. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness unresolved marker refusal`

Theorem 669.30 (Paper-Lean drift witness formalstatus sync readiness)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, a downstream consumer may synchronize a* FormalStatus *token in the family* formalTarget, encodedDef, scaffoldChecked, theoremChecked, auditClean, axiomClean, bridgeChecked *only after reading the same paper marker row M, normalized requested-name row A, duplicate-label ledger L, Lean inventory row I, and verdict row R. The synchronization is an objectwise verification-axis read of the marker-resolution verdict, and it creates no paper-side closure upgrade, bridge row, semantic equivalence, or checked theorem content.*

Proof. Token objectwise audit is supplied by Definition 669.12. It says that the exported verification token is read from the displayed marker object and its local audit evidence, so the consumer must keep the same M, A, L, I, R rows rather than replacing them by an ambient status token.

Axis separation is the boundary supplied by Definition 669.19. A synchronized formalstatus row remains a verification-axis report and therefore cannot become paper-side closure, bridge evidence, semantic equivalence, or theorem proof content.

Finally, closure-target resolution from Definition 669.18 fixes the inventory-and-ledger read that any status synchronization depends on. The consumer may record that read as a formalstatus token, but the only certified object is the same marker-resolution verdict row. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness formalstatus sync readiness`

Theorem 669.31 (Paper-Lean drift witness marker-family resolution readiness)**.** *The five marker families named by* `\leanchecked`*,* `\leanvariant`*,* `\leansorryd`*,* `\leanstmt`*, and* `\leandef` *remain separated through every synchronization consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier. Each family resolves only through its own exact inventory-and-ledger verdict for the same M, A, L, I, R rows, and no family may borrow exact resolution, duplicate-label refusal, or unresolved-marker refusal from another family.*

Proof. Marker-kind separation is the first boundary. It keeps checked theorem markers, variant markers, sorry markers, statement markers, and definition markers as different paper-site rows even when their requested Lean names have similar surface shape.

Verdict-ledger totality then assigns the visible verdict for a fixed marker family and site. Apply Definition 669.22: the consumer sees exactly the ledger-and-inventory verdict for the selected marker row, not a family-erased verdict.

Duplicate-ledger readiness and inventory-refusal readiness handle the obstruction branches. By Definition 669.28 and Definition 669.29, duplicate labels and absent inventory hits remain refusals for the same marker family and cannot be repackaged as another family resolution. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness marker family resolution readiness`

Theorem 669.32 (Paper-Lean drift witness audit-map consumer totality)**.** *Every downstream* AuditMapDependencyWeave$^{\uparrow}$ *consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier receives exactly one of exact resolution, duplicate-label refusal, or unresolved-marker refusal for a marker site. The verdict is transported only through H, C, P, N, and no ambient audit route may erase a duplicate-label or unresolved-marker obstruction.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness audit map consumer totality`

Proof. The dependency-weave handoff is fixed by Definition 669.23. It sends the audit-map consumer the displayed finite packet and carries marker resolution only as dependency data.

Dependency-routing scope and consumer-row scope identify the allowed route. By Definition 669.24 and Definition 669.25, the consumer reads the same marker, name, ledger, inventory, and verdict rows, and the only transport surface is H, C, P, N.

The readiness rows above close the remaining consumer surface. Formalstatus synchronization from Definition 669.30 cannot erase the obstruction, and marker-family readiness from Definition 669.31 prevents a refused site from escaping through a different marker family. □

Theorem 669.33 (Paper-Lean drift witness audit consumer kernel scope)**.** *Every* AuditMapDependencyWeave$^{\uparrow}$ *consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier reads the witness only through the finite* BHist *rows $M, A, L, I, R, H, C, P, N$. The consumer handoff is the same marker row, normalized requested-name row, duplicate-label ledger, Lean inventory row, verdict row, componentwise* hsame *transport,* Cont *replay row,* Pkg *provenance row, and local* NameCert *naming row; it does not add an external proof object, approximate name repair, or closure-axis summary.*

Proof. The audit-map consumer totality theorem, Definition 669.32, identifies the only visible verdicts as exact resolution, duplicate-label refusal, or unresolved-marker refusal for the fixed marker site. It also confines the transport surface to the displayed downstream route.

The dependency-weave handoff theorem, Definition 669.23, names the consumer channel. Along that channel the finite packet is sent as dependency data, so the consumer can read $M, A, L, I, R, H, C, P, N$ and no unlisted carrier.

Formalstatus synchronization from Definition 669.30 keeps any status read objectwise over the same marker, name, ledger, inventory, and verdict rows. Axis separation from Definition 669.19 then prevents this kernel-scoped handoff from becoming theorem content, bridge evidence, or a paper-side closure certificate. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness audit consumer kernel scope`

Theorem 669.34 (Paper-Lean drift witness resolution ledger kernel scope)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, exact resolution, duplicate-label refusal, and unresolved-marker refusal form a total consumer ledger over the finite rows M, A, L, I, R. No* AuditMapDependencyWeave$^{\uparrow}$ *consumer may replace that ledger by approximate matching, a neighbouring Lean declaration, a marker-family transfer, or a route-local guess.*

Proof. Apply Definition 669.32. The consumer receives exactly one of the three ledger outcomes for the marker site, and duplicate-label or unresolved-marker obstruction cannot be erased by the route.

The dependency-weave handoff theorem, Definition 669.23, keeps the ledger attached to the exported marker-resolution dependency row. Hence exact resolution remains a read of M, A, L, I, R and not a lookup against a nearby Lean name.

The formalstatus synchronization theorem, Definition 669.30, gives only a token read after the same verdict row is fixed. By axis separation, Definition 669.19, such a token cannot replace the total ledger or promote an obstruction into checked theorem content. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness resolution ledger kernel scope`

Theorem 669.35 (Paper-Lean drift witness status token kernel boundary)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, formalstatus synchronization is only a verification-axis token read from the marker-resolution verdict over* M, A, L, I, R*. It cannot upgrade theoryclosure data, create a bridgestatus row, change the semantic theorem content of the paper claim, or supply an additional consumer obligation outside* H, C, P, N*.*

Proof. Formalstatus synchronization readiness, Definition 669.30, already fixes the status token as an objectwise verification-axis read. The token is read after the same marker, normalized-name, ledger, inventory, and verdict rows are available.

Audit-map consumer totality from Definition 669.32 and the dependency-weave handoff theorem, Definition 669.23, identify the downstream consumer surface. The route carries the finite witness through H, C, P, N rather than through a closure or bridge constructor.

Axis separation, Definition 669.19, supplies the boundary. A synchronized status token reports verification-axis information only, so it cannot alter theoryclosure, bridgestatus, semantic theorem content, or the kernel-scoped consumer obligations of the witness. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness status token kernel boundary`

Theorem 669.36 (Paper-Lean drift witness marker inventory kind lock)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, each paper marker is resolved only through its own marker-kind row* M*, normalized requested-name row* A*, duplicate-label ledger* L*, Lean inventory row* I*, and verdict row* R*. A checked-theorem marker row may therefore not be consumed as a statement-only, encoded-definition, variant, or target marker row, and no audit-map consumer may repair the mismatch by approximate name lookup.*

Proof. Marker-kind separation from Definition 669.10 fixes the marker family before any inventory read is interpreted. The normalized requested-name row and the duplicate-label ledger are therefore attached to the marker kind already present in M.

Inventory verdict totality, Definition 669.22, supplies the only accepted outcomes over M, A, L, I, R. It reads exact resolution, duplicate-label refusal, or unresolved-marker refusal from those rows, rather than from a neighbouring declaration name.

Marker-family resolution readiness from Definition 669.31 closes the consumer lock. Since each marker family has its own readiness route, a checked-theorem site cannot be retyped as another marker family or discharged by an approximate repair rule. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness marker inventory kind lock`

Theorem 669.37 (Paper-Lean drift witness duplicate obstruction route preservation)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, duplicate-label refusal remains attached to the same* M, A, L, I, R *packet when the witness is transported through* H, C, P, N*. Audit-map dependency routing may expose the obstruction to downstream consumers, but it may not erase the duplicate ledger, substitute an exact-resolution row, or move the refusal to a different marker site.*

Proof. Duplicate-label priority from Definition 669.7 fixes the duplicate branch before unresolved-marker lookup or exact-resolution export can be used. Duplicate-ledger readiness, Definition 669.28, records that branch in the same finite ledger rows M, A, L, I, R.

The dependency-weave handoff theorem, Definition 669.23, transports the witness as dependency data through H, C, P, N. This handoff does not introduce a second verdict row, so the duplicate obstruction travels with the original marker, name, ledger, inventory, and verdict packet.

Audit-map consumer totality, Definition 669.32, supplies the downstream exclusion. A consumer is total over exact resolution, duplicate-label refusal, and unresolved-marker refusal, but totality does not authorize replacing the duplicate branch by a route-local exactness claim. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness duplicate obstruction route preservation`

Theorem 669.38 (Paper-Lean drift witness status consumer exactness)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, a formalstatus synchronization consumer may read only the exact verdict row for the same marker site. The consumer cannot infer theoryclosure, bridgestatus, semantic theorem content, or any marker-resolution fact outside the* $M, A, L, I, R, H, C, P, N$ *rows of that witness.*

Proof. Formalstatus synchronization readiness, Definition 669.30, makes the status read objectwise over the same marker, normalized requested name, duplicate ledger, inventory lookup, and verdict. The consumer therefore starts from the exact verdict row already fixed for that marker site.

The status token kernel boundary theorem, Definition 669.35, keeps this read on the verification axis. It permits a status token only as a rowwise consequence of the local marker-resolution verdict, not as a constructor for closure or bridge data.

Axis separation from Definition 669.19 and closure-target resolution from Definition 669.18 give the final exclusions. A synchronized formalstatus token cannot prove theoryclosure, bridgestatus, semantic theorem content, or any marker-resolution assertion beyond the exact rows supplied by the carrier. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness status consumer exactness`

Theorem 669.39 (Paper-Lean drift witness public resolution export)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, the public audit export of exact marker resolution consists only of the shared rows* $M, A, L, I, R, H, C, P, N$*. The export exposes the marker row, normalized requested-name row, duplicate-label ledger, Lean inventory row, verdict row, componentwise* hsame *transport,* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}$ *naming row, and it exposes no private repair rule, family-erased lookup, aggregate bridge verdict, or closure-axis certificate.*

Proof. Resolution exactness from Definition 669.3 fixes the exact branch as a read of M, A, L, I, R. Duplicate-ledger readiness and inventory-refusal readiness, Definition 669.28 and Definition 669.29, show that the two obstruction branches use the same public rows rather than a hidden fallback.

The audit consumer kernel scope theorem, Definition 669.33, identifies the only exported downstream surface as H, C, P, N over the same finite witness. Hence public audit consumers receive the listed rows and no auxiliary carrier.

Finally, marker-family resolution readiness prevents a public consumer from erasing the marker family before reading the verdict. Status-token kernel boundary then prevents the exported exact-resolution row from becoming an aggregate bridge status or a paper-side closure certificate. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness public resolution export`

Theorem 669.40 (Paper-Lean drift witness bridge-ready dependency surface)**.** *Every downstream bridge or audit consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier factors through the* AuditMapDependencyWeave$^{\uparrow}$ *dependency surface before using the local marker-resolution verdict. Such a consumer may transport exact resolution, duplicate-label refusal, or unresolved-marker refusal along H, C, P, N, but it cannot replace the local verdict by an aggregate bridge status, neighbouring audit summary, or global synchronization token.*

Proof. The dependency-weave handoff theorem, Definition 669.23, names the route through which neighbouring consumers receive the packet. Audit-map consumer totality, Definition 669.32, then says that the received data is exactly one local verdict for the fixed marker site.

The public resolution export theorem above fixes the carrier material available to the consumer as the rows $M, A, L, I, R, H, C, P, N$. Since the dependency surface is rowwise, the consumer can use hsame transport, Cont replay, Pkg provenance, and NameCert naming only as dependency data attached to the same local verdict.

The status-token kernel boundary theorem, Definition 669.35, supplies the exclusion. A synchronized token or aggregate bridge report is not a replacement verdict, so it cannot discharge duplicate-label refusal, unresolved-marker refusal, or the exactness obligation of the local marker row. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness bridge ready dependency surface`

Theorem 669.41 (Paper-Lean drift witness public token consumer totality)**.** *Every public verification-token consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier is total over the three public marker-resolution outcomes: exact resolution, duplicate-label refusal, and unresolved-marker refusal. The consumer either reads the formalstatus token from the exact verdict over M, A, L, I, R, reads the duplicate-label refusal from the same ledger, or reads the unresolved-marker refusal from the same inventory lookup; no public token consumer is undefined on one of these branches.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness public token consumer totality`

Proof. Verdict-ledger totality, Definition 669.22, gives the three-way ledger for a fixed marker site. Resolution exactness supplies the exact branch, while duplicate-ledger readiness and inventory-refusal readiness supply the two refusal branches.

Marker-family resolution readiness keeps this totality family-local. A token consumer therefore cannot borrow an exact verdict from another marker family or reinterpret a refused branch as a different public marker kind.

Audit-map consumer totality and formalstatus synchronization readiness identify the token-facing consumer. The status-token kernel boundary theorem then keeps the token read on the verification axis, so totality over exact resolution, duplicate-label refusal, and unresolved-marker refusal does not create bridge evidence or paper-side closure data. □

Theorem 669.42 (Paper-Lean drift witness audit replay source lock)**.** *For every accepted* PaperLeanDriftWitness$^{\uparrow}$ *carrier, every audit replay consumed by an* AuditMapDependencyWeave$^{\uparrow}$ *route remains sourced in the same $M, A, L, I, R, H, C, P, N$ packet. The replay may expose the local marker row, normalized requested-name row, duplicate ledger, inventory lookup, verdict, transport, continuity replay, package provenance, and local* NameCert *row, but it may not substitute a neighbouring marker, neighbouring normalized name, or neighbouring verdict for the original packet.*

Proof. Resolution exactness, Definition 669.3, first fixes the exact branch as a rowwise read of M, A, L, I, R. Marker inventory kind lock, Definition 669.36, keeps the marker family and inventory lookup attached to that same source before any consumer sees the replay.

The public resolution export theorem, Definition 669.39, names the exported rows as $M, A, L, I, R, H, C, P, N$. Applying public token consumer totality then shows that exact resolution, duplicate-label refusal, and unresolved-marker refusal are all consumed from this single packet, not from a second witness.

Finally, the dependency-weave handoff route only transports the exported rows to neighbouring audit-map consumers. Since the handoff receives the locked packet after marker-kind separation and exactness, it cannot replace the source marker or verdict by a neighbouring marker summary. □

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness audit replay source lock`

Theorem 669.43 (Paper-Lean drift witness Lean inventory hit exhaustion)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, exact marker resolution is exhausted by the displayed Lean inventory hit together with the non-duplicate label row. If the duplicate-label ledger marks the requested name as duplicated, or if the Lean inventory lookup is absent, the corresponding branch remains a public refusal row and cannot be consumed as an exact hit.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness lean inventory hit exhaustion`

Proof. Resolution exactness identifies the exact branch with the normalized requested-name row A, duplicate-label ledger L, Lean inventory lookup I, and verdict row R. The exact branch is therefore available only when the inventory hit and the non-duplicate label condition are both present in the displayed packet.

Marker inventory kind lock keeps this inventory read within the marker family fixed by M. Public resolution export then exposes only the same inventory and ledger rows to downstream consumers, so no approximate declaration name or marker-family coercion can supply a second exact hit.

Public token consumer totality supplies the exhaustive case split. The exact branch consumes the displayed hit; the duplicate-label branch consumes the duplicate ledger; the unresolved-marker branch consumes the absent inventory lookup. The two refusal branches remain refusal rows after export and cannot be reclassified as exact resolution. □

Theorem 669.44 (Paper-Lean drift witness name normalization no-escape)**.** *For every accepted* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$ *carrier, underscore and Lean-name normalization is only a replay row inside the same* $M, A, L, I, R, H, C, P, N$ *packet. Normalization authorizes comparison with the displayed requested-name row, but it gives no permission for approximate lookup, spelling repair, marker-family coercion, or substitution of a neighbouring declaration name.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness name normalization noescape`

Proof. Resolution exactness uses normalization as part of the requested-name row A and then reads the duplicate ledger, inventory lookup, and verdict from L, I, R. Thus normalization is an input row to exact resolution, not an external search rule.

Marker inventory kind lock keeps the normalized name under the marker family fixed by M. Public resolution export carries this normalized name together with H, C, P, N, so audit-map consumers receive the replayed spelling only as part of the same finite packet.

Public token consumer totality closes the exclusion. A consumer is total over exact resolution, duplicate-label refusal, and unresolved-marker refusal for the given packet, and this totality leaves

no branch in which normalization can escape into approximate lookup or coerce one marker family into another. □

The exact-resolution row exported by PaperLeanDriftWitness$^{\uparrow}$ is consumed by the cross-map routing and synthesis-consumer rows of chapter 839. The PaperLeanDriftWitness$^{\uparrow}$ packet supplies marker normalization, duplicate-label refusal, Lean inventory lookup, and exact-resolution verdict rows, while AuditMapDependencyWeave$^{\uparrow}$ decides how such local rows travel through neighbouring audit-map routes without discharging their obstruction rows.

Theorem 669.45 (Paper-Lean drift witness public export route)**.** *Every public audit-map consumer of an accepted* PaperLeanDriftWitness$^{\uparrow}$ *packet receives exactly the marker row, normalized requested-name row, duplicate-label ledger, Lean inventory lookup, verdict row, transport, replay, provenance, and local NameCert row through* AuditMapDependencyWeave$^{\uparrow}$*. The route is the displayed packet $M, A, L, I, R, H, C, P, N$; it exports exact resolution, duplicate-label refusal, and unresolved-marker refusal as local verdict data, not as a private repair rule, aggregate bridge status, or closure-axis certificate.*

Lean4 ✓ `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness public export route`

Proof. Begin with public resolution export, Definition 669.39. It identifies the exported material as the rows $M, A, L, I, R, H, C, P, N$ and excludes private repair rules, family-erased lookup, aggregate bridge verdicts, and closure-axis certificates.

Next use the bridge-ready dependency surface, Definition 669.40. It routes downstream bridge and audit consumers through AuditMapDependencyWeave$^{\uparrow}$ before they may use the local marker-resolution verdict, so the public consumer receives the packet only as dependency data.

Finally apply the audit replay source lock and name-normalization no-escape theorems, Definition 669.42 and Definition 669.44. Replay remains sourced in the same packet, and normalization cannot become approximate lookup or marker-family coercion. Hence the public route is exactly the listed local rows through AuditMapDependencyWeave$^{\uparrow}$. □

Closure status for NameCert$_{\text{PaperLeanDriftWitness}^{\uparrow}}$**.**

Constructive story: A PaperLeanDriftWitness$^{\uparrow}$ packet is generated from a paper marker row, normalized requested-name row, duplicate-label ledger, Lean inventory lookup, resolution verdict, componentwise hsame transport, displayed replay routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PaperLeanDriftWitness}^{\uparrow}}$, publicClosed)

Scope closed: The scoped surface binds the exact-resolution, duplicate-label, unresolved-marker, marker-kind, inventory-verdict, transport, replay, provenance, dependency-weave handoff, and audit-map dependency consumer rows for exact paper-Lean marker resolution.

Formal verification: FormalStatus(NameCert$_{\text{PaperLeanDriftWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.PaperLeanDriftWitnessUp.PaperLeanDriftWitness\_public\_export\_route`

Bridge status: none

Origin: human

Not claimed: This chapter does not close semantic equivalence between paper claims and Lean declarations, checked theorem content, bridge checking, or any permission to treat unresolved markers as verified.

Upgrade path: The next bridge surface is the downstream audit-consumer interpretation of the public export route through AuditMapDependencyWeave$^{\uparrow}$.

Chapter 670

A Concrete Naming Certificate for $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$

The $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$ surface records the finite precision-budget packet used when the Real completeness diagonal target is consumed through cofinal windows. It packages request rows, selected finite windows, regular rational readback, RealUp seal routing, transport, continuation, provenance, and local naming in one displayed BHist carrier. The packet is a budgeted route for a diagonal-limit consumer; it is not a quotient stream, host real equality, selected ambient limit, or completeness principle.

The cofinal-window ledger is read through chapter 126; the regular handoff is read through chapter 301; the seal is read through chapter 38; and the finite dyadic observation budget is read through chapter 291. These references fix the L10 dependency lattice used by the carrier and local NameCert obligations.

Falsifiable prediction: If an accepted cofinal regular limit-budget packet reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the displayed cofinal-window ledger or $\mathsf{RegSeqRat}^{\uparrow}$ handoff row, the packet is refuted by the carrier classifier.

Independence witness: cofinalwindowrealseal, realcompletenesstailchoiceaudit, finiteobservationbudgetselector, diagonallimitrepresentative, regseqrat: none is bijective to $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$ because this carrier binds precision request, cofinal-window ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, transport, continuation, provenance, and local NameCert rows in one packet.

Definition 670.1 (CofinalRegularLimitBudget carrier)**.** *A* $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (q, w, r, e, h, c, p, n)$$

with the following displayed rows:

1. *q is the finite precision-request row read by the diagonal-limit consumer;*
2. *w is the cofinal-window ledger assigning each displayed request to a finite* $\mathsf{StreamName}^{\uparrow}$ *observation window;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the selected windows have been fixed;*
4. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed regular rational readback;*
5. *h records componentwise* hsame *transports for the request, window, handoff, seal, route, provenance, and local naming rows;*

6. *c records the displayed* Cont *route from q through w and r into e;*
7. *p is the* Pkg *provenance over* BHist, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, *and local certificate rows;*
8. *n is the local* NameCert$_{\text{CofinalRegularLimitBudget}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows q, w, r, e, h, c, p, n. It has no coordinate for quotient-stream equality, host real equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 670.2 (CofinalRegularLimitBudget NameCert obligations)**.** *The local* NameCert$_{\text{CofinalRegularLimitBudget}^{\uparrow}}$ *surface for Definition 670.1 consists of five finite obligation rows:*

1. *carrier admission, requiring every accepted object to display the finite packet $B = (q, w, r, e, h, c, p, n)$;*
2. *cofinal-window coverage, requiring each precision request consumed by the diagonal route to pass through the displayed window ledger w;*
3. RegSeqRat$^{\uparrow}$ *handoff, requiring the regular rational readback row r to be reached only after the request and cofinal-window rows are fixed;*
4. Real$^{\uparrow}$ *seal routing, requiring the seal row e to be reached only through q, w, r, h, c, p, n;*
5. *ledger non-escape, requiring the packet to export no quotient stream, host real equality, selected limit, hidden completeness row, or unrecorded diagonal stream.*

Lean4 ✓ `BEDC.Derived.CofinalRegularLimitBudgetUp.CofinalRegularLimitBudgetTasteGate single carrier alignment`

Proof. Carrier admission is projection from the displayed tuple in Definition 670.1. The request, window, handoff, seal, transport, continuation, provenance, and local naming rows are all visible coordinates of one finite BHist packet.

Cofinal-window coverage reads q and then w before any regular rational handoff is available. Since w is the only window ledger in the carrier, the consumer cannot redirect a precision request to an unnamed tail or to a host-level stream.

The RegSeqRat$^{\uparrow}$ handoff is then forced through r. The row is attached to the displayed request and selected windows, so it cannot be replaced by quotient-stream equality or a selected ambient limit.

Finally, the seal and non-escape rows inspect e, h, c, p, n. They give the only route into the Real$^{\uparrow}$ seal and provide no coordinate for host real equality, hidden completeness, or an unrecorded diagonal stream. □

Closure status for NameCert$_{\text{CofinalRegularLimitBudget}^{\uparrow}}$.

Constructive story: A CofinalRegularLimitBudget$^{\uparrow}$ packet is generated from a precision request, a cofinal-window ledger, a RegSeqRat handoff row, a RealUp seal row, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\text{CofinalRegularLimitBudget}^{\uparrow}}$, seed)

Scope closed: Definition 670.1 and Definition 670.2 bind the finite precision-budget route from diagonal requests through cofinal windows into RegSeqRat handoff and RealUp seal rows.

Formal verification: FormalStatus(NameCert$_{\text{CofinalRegularLimitBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient-stream equality, host real equality, selected ambient limits, countable choice, hidden completeness, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the carrier and checked obligations for cofinal-window coverage, RegSeqRat handoff, RealUp seal routing, ledger provenance, and non-escape.

Definition 670.3 (CofinalRegularLimitBudget route-competing windows)**.** *Let $B = (q, w, r, e, h, c, p, n)$ be an accepted* $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$ *carrier as in Definition 670.1. A pair of displayed cofinal-window ledgers w_0, w_1 is route-competing for B when each w_i is presented as the window row for the same precision request q, each presentation is read by the displayed* Cont *route c before the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row r and the same* $\mathsf{Real}^{\uparrow}$ *seal row e, and the presentation uses no rows other than q, w_i, r, e, h, c, p, n.*

Theorem 670.4 (CofinalRegularLimitBudget window-route determinacy)**.** *Let $B = (q, w, r, e, h, c, p, n)$ be an accepted* $\mathsf{CofinalRegularLimitBudget}^{\uparrow}$ *carrier. If w_0, w_1 are route-competing cofinal-window ledgers for B in the sense of Definition 670.3, then the carrier transport ledger h records* $\mathsf{hsame}(w_0, w_1)$*. Thus, for a fixed precision request q, fixed displayed route c, fixed* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row r, and fixed* $\mathsf{Real}^{\uparrow}$ *seal row e, the cofinal-window ledger consumed by the route is determined up to the displayed* hsame *transport of the carrier.*

Proof. By carrier admission in Definition 670.2, every accepted object exposes the finite packet $B = (q, w, r, e, h, c, p, n)$ and nothing outside those displayed rows is part of the local certificate surface. Cofinal-window coverage then says that every precision request consumed by the diagonal route passes through the displayed window ledger before any regular rational handoff is available.

Apply this to the two route-competing presentations w_0 and w_1. Both are presented for the same request row q and both are consumed by the same continuation row c before the same handoff row r. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff obligation requires r to be reached only after the request and cofinal-window rows have been fixed, so the two presentations cannot diverge at or after r by choosing a further window. The $\mathsf{Real}^{\uparrow}$ seal-routing obligation likewise requires e to be reached only through q, w, r, h, c, p, n, so the seal cannot introduce an additional cofinal-window coordinate.

It remains only to compare the two displayed window presentations inside the accepted packet. The classifier in Definition 670.1 compares the displayed rows q, w, r, e, h, c, p, n, and the transport row h is the componentwise hsame ledger for the request, window, handoff, seal, route, provenance, and local naming rows. Since w_0 and w_1 share q, c, r, e, p, n and the non-escape row of Definition 670.2 provides no quotient-stream equality, host real equality, selected limit, hidden completeness row, or unrecorded diagonal stream from which a second route could be read, the only admitted comparison between the two displayed window rows is the window component of h. Hence h records $\mathsf{hsame}(w_0, w_1)$. □

Theorem 670.5 (CofinalRegularLimitBudget TasteGate witness surface)**.** *The displayed carrier of Definition 670.1 has a nontrivial witness pair and a faithful field projection. One witness is the packet*

$$(BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHis$$

and the other is the packet

$$(BHist.e0\, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Empty, BHist.Em$$

They are separated by the precision-request row, and equality of the eight displayed fields q, w, r, e, h, c, p, n determines equality of the constructor.

Proof. The two packets use the same seven rows after the precision request, so the only comparison needed is the first coordinate. In the closed BHist grammar, $BHist.Empty$ and $BHist.e0\, BHist.Empty$ are distinct constructor forms. Thus the displayed carrier contains two separated packets.

Field faithfulness is read directly from the carrier shape in Definition 670.1. A carrier is exactly an eight-field constructor over q, w, r, e, h, c, p, n; once these fields are equal componentwise, constructor equality gives equality of the displayed packet.

The window-route determinacy theorem Definition 670.4 supplies the semantic use of this faithful projection for the cofinal-window row. It says that route-competing windows for a fixed request, route, handoff, and seal are compared only through the displayed transport ledger, so the TasteGate surface does not hide a second cofinal-window coordinate.

Lean already exposes the field-faithful projection through the existing CofinalRegularLimit-Budget TasteGate alignment target. The remaining Lean surface for this paper theorem is the explicit *Nontrivial* instance induced by the two packets above. □

Chapter 671

A Concrete Naming Certificate for $\mathsf{KernelSourceChannelLedger}^{\uparrow}$

The $\mathsf{KernelSourceChannelLedger}^{\uparrow}$ packet names the finite source-channel audit surface surrounding a Lean kernel acceptance event. It separates generated candidates, classifier stamps, accepted declarations, environment ancestry, axiom-query refusal rows, compiler routes, and local naming. The packet is a concrete $\mathsf{NameCert}$ carrier over the audit rows already exposed by $\mathsf{KernelAcceptanceTrace}^{\uparrow}$, $\mathsf{CertificateCompiler}^{\uparrow}$, and $\mathsf{NameCertMorphism}^{\uparrow}$; it does not certify the host kernel from inside BEDC and does not replace the displayed ancestry rows by source-text confidence.

The ledger is therefore not a kernel-only packet. Row T consumes the accepted-declaration audit surface of chapter 557; row M consumes either the executable certificate route of chapter 473 or the certificate-preservation graph of chapter 481; and row F keeps those sibling channels separated before any audit consumer reads Q, R.

Definition 671.1 (Kernel source-channel ledger carrier)**.** *A* $\mathsf{KernelSourceChannelLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, K, A, E, Q, R, T, M, F, H, C, P, N)$$

with the following displayed rows:

1. *G is the generated candidate row supplied by the source-side generator or certificate compiler;*
2. *K is the kernel classifier-stamp row attached to the candidate before it is exposed as an accepted declaration;*
3. *A is the accepted declaration row read by the local kernel-acceptance trace;*
4. *E is the environment ancestry ledger recording the declaration environment visible to the acceptance event;*
5. *Q is the displayed* `#print axioms` *query row for A;*
6. *R is the refusal row for any meeting between Q and the project forbidden-dependency surface;*
7. *T is the* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *packet that binds generated candidate, stamp, declaration, ancestry, and query rows;*
8. *M is the* $\mathsf{NameCertMorphism}^{\uparrow}$ *or* $\mathsf{CertificateCompiler}^{\uparrow}$ *route row that accounts for the source-channel handoff into the accepted certificate surface;*

9. *F is the finite channel-separation row distinguishing generated candidates, kernel stamps, accepted declarations, environment ancestry, and refusal output;*
10. *H records componentwise* hsame *transports for all displayed rows;*
11. *C records* Cont *replay routes for candidate generation, kernel stamping, declaration exposure, ancestry query, axiom-query comparison, compiler handoff, and consumer reads;*
12. *P is the* Pkg *provenance over* BHist, Cont, Pkg, NameCert, KernelAcceptanceTrace$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, *and* NameCertMorphism$^{\uparrow}$*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelSourceChannelLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for host trust in the generator, unstated kernel soundness, source-text-only acceptance, hidden environment ancestry, implicit axiom-query success, or a compiler correctness claim outside the displayed morphism and compiler rows.

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.TasteGate.KernelSourceChannelLedgerTasteGate single carrier alignment`

Theorem 671.2 (Kernel source-channel ledger NameCert obligations)**.** *For every accepted* KernelSourceChannelLed *carrier*

$$S = (G, K, A, E, Q, R, T, M, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelSourceChannelLedger}^{\uparrow}}$ *surface consists exactly of generated-candidate exposure, classifier-stamp exposure, accepted-declaration exposure, environment-ancestry exposure, axiom-query and refusal exposure, kernel-trace exposure, compiler or morphism route exposure, channel-separation exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Any downstream audit consumer must read the source channel through the finite packet S and may not identify generated candidates with accepted declarations, kernel stamps with environment ancestry, or clean query output with absent query output.

Proof. Project the carrier of Definition 671.1. The rows $G, K, A, E, Q, R, T, M, F, H, C, P, N$ are the only public coordinates available to the local naming certificate.

The acceptance rows are read through T. That trace binds the generated candidate, kernel stamp, accepted declaration, environment ledger, axiom-query report, transport, replay, provenance, and local naming data before any consumer sees the source channel.

The compiler side is read through M. If the source handoff uses a CertificateCompiler$^{\uparrow}$ packet, the handoff remains a finite compiler graph over source and target certificate rows. If it uses a NameCertMorphism$^{\uparrow}$ packet, the handoff remains a finite morphism graph preserving classifier and hsame rows.

Finally F, H, C, P, N keep the rows separated, transported, replayed, packaged, and named. Since the carrier has no coordinate identifying generation with acceptance or query absence with query success, every audit read must factor through the displayed source-channel ledger. □

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket namecert obligations`

Theorem 671.3 (Kernel source-channel ledger audit boundary)**.** *Every public read of an accepted* KernelSourceChannelLedger$^{\uparrow}$ *packet factors through the finite* BHist *rows of Definition 671.1, the displayed* Cont *replay routes, the* Pkg *provenance, the local* NameCert *row, and the cited*

KernelAcceptanceTrace$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, *and* NameCertMorphism$^{\uparrow}$ *packets. The boundary exports no hidden source-channel trust, no unstated kernel soundness theorem, no ambient environment scan, no unqueried axiom-purity claim, and no compiler semantics outside the displayed certificate route.*

Proof. Begin with an accepted packet S. A consumer can project only the generated candidate, classifier stamp, accepted declaration, environment ancestry, query, refusal, trace, compiler or morphism route, channel-separation, transport, continuation, provenance, and naming rows listed in the carrier.

The trace row T accounts for kernel acceptance as a finite audit event. The route row M accounts for the source-to-certificate handoff through the displayed compiler or morphism surface. The refusal row R accounts for forbidden ancestry by using the printed query row Q rather than a host-side assertion.

It remains to check escape routes. There is no carrier field for trusting a generator directly, no field for a host proof of kernel soundness, no field for an ambient environment scan, no field for clean ancestry without Q, and no field for compiler semantics beyond M. Thus the audit boundary is exactly the finite source-channel ledger surface. □

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket audit boundary`

Theorem 671.4 (Kernel source-channel ledger trace projection)**.** *For every accepted* KernelSourceChannelLedger$^{\uparrow}$ *carrier*

$$S = (G, K, A, E, Q, R, T, M, F, H, C, P, N),$$

the generated candidate, kernel stamp, accepted declaration, ancestry, query, refusal, trace, and compiler-route rows project only through the displayed finite surface T, M, F, H, C, P, N after the source rows G, K, A, E, Q, R have been read. No public projection may identify generated candidates with accepted declarations or replace the compiler-route row by host environment trust.

Proof. Project the accepted carrier. The source rows G, K, A, E, Q, R expose candidate generation, kernel stamping, declaration exposure, ancestry, query output, and refusal output as separate coordinates.

The trace row T binds those source coordinates into the kernel-acceptance event. The route row M then accounts for the compiler or morphism handoff, while F keeps the channels separated.

Finally H, C, P, N transport, replay, package, and name the same finite packet. Since every projection has passed through these displayed rows, no consumer can collapse the source channels or replace them by host trust. □

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket trace projection`

Theorem 671.5 (Kernel source-channel ledger query refusal stability)**.** *For every accepted* KernelSourceChannelLed *carrier $S = (G, K, A, E, Q, R, T, M, F, H, C, P, N)$, refusal stability is preserved by the displayed* `#print axioms` *query row Q and refusal row R under* hsame *transport and* Cont *replay. The refusal row cannot be replaced by absent query output, ambient source confidence, or host-side trust in the accepted declaration.*

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket query refusal stability`

Proof. Read the query and refusal rows first. The row Q records the displayed axiom query for A, and R records the comparison against the forbidden-dependency surface.

Transport through H and replay through C preserve those two rows as part of the same accepted packet. The trace row T and channel-separation row F keep the query output distinct from an absent query.

Apply the audit boundary. Because the carrier has no coordinate for ambient confidence or host trust, refusal stability remains the displayed Q, R fact under replay and cannot be supplied by an unstated source-channel assumption. □

Theorem 671.6 (Kernel source-channel ledger source-query separation)**.** *For every accepted* KernelSourceChannelL *carrier*

$$S = (G, K, A, E, Q, R, T, M, F, H, C, P, N),$$

the generated candidate row G, kernel stamp row K, accepted declaration row A, environment ancestry row E, printed axiom query row Q, and refusal row R are separated before any compiler or morphism route M can be read. No consumer may replace the Q, R route by source confidence, ambient environment trust, or absent query output.

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket source query separation`

Proof. Project the carrier of Definition 671.1. The rows G, K, A, E, Q, R are displayed source and query coordinates, while M is only the later compiler or morphism route row.

Apply the NameCert obligations of Definition 671.2. They require generated-candidate exposure, classifier-stamp exposure, accepted-declaration exposure, environment-ancestry exposure, axiom-query and refusal exposure, and compiler or morphism route exposure as separate finite duties.

Next use query-refusal stability from Definition 671.5. The printed query row Q and refusal row R persist under hsame transport and Cont replay, so absent query output cannot replace the displayed route.

Finally apply the audit boundary of Definition 671.3. Since the carrier has no coordinate for source confidence or ambient environment trust, every consumer must read the query and refusal rows before using the route row M. □

Theorem 671.7 (Kernel source-channel ledger public export package)**.** *The audit boundary of Definition 671.3, the trace projection of Definition 671.4, the query-refusal stability of Definition 671.5, and the source-query separation of Definition 671.6 assemble the public source-channel export of an accepted* KernelSourceChannelLedger$^{\uparrow}$ *packet. The export passes through accepted-declaration trace rows, certificate-compiler or* NameCert*-morphism rows, separated source and query channels, transport, replay, provenance, and local naming while keeping query-refusal and source-channel surfaces distinct.*

Lean4 ✓ `BEDC.Derived.KernelSourceChannelLedgerUp.KernelSourceChannelLedgerPacket public export package`

Proof. Project the carrier rows $G, K, A, E, Q, R, T, M, F, H, C, P, N$. The source-channel surface is carried by $G, K, A, E, T, M, F, H, C, P, N$, while the query-refusal surface is carried by Q, R, H, C, P, N.

Apply Definition 671.3 and Definition 671.4. These rows expose the accepted-declaration trace and keep kernel acceptance visible before any compiler or morphism route is consumed.

Next apply Definition 671.5 and Definition 671.6. The query row and refusal row remain separated from source confidence and environment trust, while the source and query channels are replayed through the same transport, continuation, provenance, and local naming rows. Hence the public export contains both surfaces without collapsing query refusal into source-channel acceptance. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelSourceChannelLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelSourceChannelLedger}^{\uparrow}$ packet is generated from a source candidate, kernel classifier stamp, accepted declaration, environment ancestry ledger, axiom-query row, refusal row, kernel-acceptance trace, certificate compiler or NameCert morphism route, channel-separation row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelSourceChannelLedger}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 671.1, Definition 671.2, Definition 671.3, Definition 671.4, and Definition 671.5 specify the finite source-channel ledger carrier, NameCert obligation surface, audit separation rows, replay provenance, trace projection, query-refusal stability, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelSourceChannelLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove host kernel soundness, completeness of all environment scans, compiler semantic correctness outside displayed certificate routes, Lean-side verification, bridge checking, or acceptance of any dependency outside the finite packet.

Upgrade path: Scoped closure requires binding those obligation rows to the displayed BHist, hsame, Cont, Pkg, NameCert, trace, compiler-route, channel-separation, and query-refusal scope.

Chapter 672

A Concrete Naming Certificate for $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$

The $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$ surface packages the AxisZeckendorf boundary that $\mathsf{AxisNat}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ remain distinct public interfaces. It records the AxisNat rows, the Nat rows, the unary-direction bridge rows, the checked cannot-claim rows, and the refusal ledger as finite data; it exports no kernel replacement of $\mathsf{Nat}^{\uparrow}$ by $\mathsf{AxisNat}^{\uparrow}$.

Its composite dependencies are the $\mathsf{AxisNat}^{\uparrow}$ interface of chapter 276, the $\mathsf{Nat}^{\uparrow}$ interface of chapter 28, the $\mathsf{UnaryDirectionBridge}^{\uparrow}$ boundary of chapter 530, and the $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ refusal row of chapter 554, joined into one finite non-replacement packet.

The refusal packet reads the AxisNat row from chapter 276, the Nat row from chapter 28, the bridge row from chapter 530, and the cannot-claim refusal row from chapter 554. The local row B permits only the unary-direction comparison, and K records refusal of replacement rather than an isomorphism, kernel rewrite, or host equality between the two interfaces.

Definition 672.1 (AxisNat replacement-refusal carrier)**.** *An* $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, N, B, K, H, C, P, L)$$

with the following displayed coordinates:

1. *A is the* $\mathsf{AxisNat}^{\uparrow}$ *interface row, including the finite axis-origin, successor-spine, and AxisZeckendorf-facing readback rows;*
2. *N is the* $\mathsf{Nat}^{\uparrow}$ *interface row, including the finite zero, successor, induction, and standard Nat naming rows;*
3. *B is the* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *row through which an admitted consumer may compare unary direction data without collapsing either interface into the other;*
4. *K is the checked cannot-claim row for the statement that* $\mathsf{AxisNat}^{\uparrow}$ *does not replace* $\mathsf{Nat}^{\uparrow}$*, together with the kernel-distinctness and bridge-dependence rows cited by the AxisZeckendorf registry;*
5. *H records displayed* hsame *transports for the AxisNat row, Nat row, bridge row, cannot-claim row, continuation row, provenance row, and local naming row;*
6. *C records the displayed* Cont *routes by which consumers may replay the AxisNat read, the Nat read, the bridge read, and the refusal read;*

7. *P is the* Pkg *provenance over* $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, BHist, hsame, Cont, *and the source* NameCert *rows;*

8. *L is the local* $\mathsf{NameCert}_{\mathsf{AxisNatReplacementRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for an isomorphism, a kernel rewrite, a definitional equality, a Nat induction transfer into AxisNat, an AxisNat successor transfer into Nat, or a global replacement principle.

Lean4 ✓ `BEDC.Derived.AxisNatReplacementRefusalUp.AxisNatReplacementRefusalTasteGate single carrier alignment`

Theorem 672.2 (AxisNat replacement-refusal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisNatReplacementRefusal}^{\uparrow}}$ *surface for Definition 672.1 consists exactly of six finite obligations:*

1. *carrier admission for the displayed packet $Q = (A, N, B, K, H, C, P, L)$;*

2. *AxisNat interface exposure through the row A;*

3. *Nat interface exposure through the row N;*

4. *bridge-bounded comparison through the* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *row B;*

5. *checked refusal coverage through the cannot-claim row K;*

6. *transport, continuation, provenance, and local naming exactness through H, C, P, L.*

Lean4 ✓ `BEDC.Derived.AxisNatReplacementRefusalUp.AxisNatReplacementRefusalCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 672.1. The packet has exactly the displayed AxisNat, Nat, bridge, refusal, transport, continuation, provenance, and naming rows.

The interface obligations are read separately from A and N. The AxisNat row supplies AxisZeckendorf-facing unary data, while the Nat row supplies the standard Nat naming surface; neither row is consumed as the other.

Bridge-bounded comparison is the row B. It permits only the displayed unary-direction comparison and does not create an isomorphism, definitional equality, or kernel rewrite between the two interfaces.

The refusal row is K, backed by the cannot-claim registry and the kernel-distinctness boundary. The remaining rows H, C, P, L preserve and replay this separation componentwise, so the NameCert surface is exhausted by the six obligations. □

Theorem 672.3 (AxisNat replacement-refusal non-replacement boundary)**.** *Every public read exported by an accepted* $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$ *packet factors through A, N, B, K, H, C, P, L. In particular, no consumer can extract a kernel replacement of* $\mathsf{Nat}^{\uparrow}$ *by* $\mathsf{AxisNat}^{\uparrow}$*, a replacement of* $\mathsf{AxisNat}^{\uparrow}$ *by* $\mathsf{Nat}^{\uparrow}$*, an unbounded bridge, a transferred induction principle, a transferred successor rule, or a host equality between the two interfaces.*

Proof. Inspect the packet coordinates. The public AxisNat data are confined to A, and the public Nat data are confined to N.

The bridge coordinate B is the only comparison route. Since it is a $\mathsf{UnaryDirectionBridge}^{\uparrow}$ row, its output is a bounded direction read, not a replacement principle for either interface.

The refusal coordinate K records the cannot-claim boundary. Transport, continuation, provenance, and naming rows H, C, P, L replay that same boundary and add no interface operation. Because the excluded replacement objects are absent from the carrier, no public read can export them. □

Theorem 672.4 (AxisNat replacement-refusal vision realization)**.** *The AxisZeckendorf cannot-claim boundary for the statement that* $\mathsf{AxisNat}^{\uparrow}$ *replaces* $\mathsf{Nat}^{\uparrow}$ *is realized by the finite* $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$ *packet of Definition 672.1. The realization cites AxisNat rows, Nat rows, UnaryDirectionBridge rows, checked cannot-claim rows, displayed* hsame *transports,* Cont *routes,* Pkg *provenance, and a local* NameCert *row, while exporting no replacement of either interface by the other.*

Proof. Start from the registry statement Definition 9350.7. It records that the AxisZeckendorf cluster does not assert replacement of $\mathsf{Nat}^{\uparrow}$ by $\mathsf{AxisNat}^{\uparrow}$.

The carrier of Definition 672.1 turns that boundary into finite rows. The rows A and N keep the two interfaces separate, B records the bounded unary-direction bridge, and K records the checked refusal ledger.

Rows H, C, P, L provide the admissible transport, continuation, package, and local naming surface. By Definition 672.3, this realization cannot be read as a positive replacement bridge, so it concretizes the vision boundary as a finite refusal certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxisNatReplacementRefusal}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$ packet is generated from AxisNat interface rows, Nat interface rows, UnaryDirectionBridge rows, checked cannot-claim rows, displayed hsame transports, continuation routes, package provenance, and a local NameCert row. It is composite over the $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, and cannot-claim ledger surfaces, bundling those existing boundaries into one refusal surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisNatReplacementRefusal}^{\uparrow}}$, seed)

Scope closed: The seed scope records the carrier, NameCert obligations, non-replacement boundary, and AxisZeckendorf vision realization for finite AxisNat-to-Nat refusal packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisNatReplacementRefusal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a replacement of NatUp by AxisNatUp, a replacement of AxisNatUp by NatUp, an interface isomorphism, kernel rewriting, transferred induction, transferred successor structure, or host equality.

Upgrade path: Obligation closure requires checked carrier coverage, bridge-bounded comparison, cannot-claim coverage, transport exactness, continuation exactness, and ledger non-escape for the displayed packet.

Chapter 673

A Concrete Naming Certificate for $\mathsf{LimitSelector}^{\uparrow}$

The $\mathsf{LimitSelector}^{\uparrow}$ surface packages the finite selector used by the diagonal-limit construction of Definition 9238.3. It names the precision request, the modulus-selected member, the finite observation window, the regular-rational readback, the RealUp seal, and the BEDC transport rows as one inspectable packet. It is a selector certificate over displayed finite data, not a host-selected limit or a quotient of streams.

The selector is a downstream L10 consumer of the certified countable-name interface of chapter 126 and the RealUp sealing interface of chapter 38; its own packet records only the finite witness route between those sibling surfaces.

Definition 673.1 (LimitSelector carrier)**.** *A* $\mathsf{LimitSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, n, \mu, k, W, D, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}$ *input row for the regular Cauchy sequence whose diagonal representative is being read;*
2. *n is the unary precision request row;*
3. *μ is the displayed modulus row assigning a finite selector bound to each unary precision request;*
4. *k is the selected index row, fixed as the value of μ at the precision requested by n;*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row read from the selected member and the selected diagonal index;*
6. *D is the dyadic readback row produced from the finite window;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and dyadic endpoint rows are fixed;*
8. *H records componentwise* hsame *transports for R, n, μ, k, W, D, S and the local structural rows;*
9. *C records the displayed* Cont *route from the precision request through the modulus, selected index, window, dyadic readback, and seal rows;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $R, n, \mu, k, W, D, S, H, C, P, N$.

Lean4 (definition) `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier`

Theorem 673.2 (LimitSelector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}$ *surface for Definition 673.1 consists of five finite obligations:*

1. *carrier admission, requiring every accepted selector object to expose the packet $L = (R, n, \mu, k, W, D, S, H, C, P, N$*
2. *modulus-selector exactness, requiring the selected index row k to be read from the displayed modulus μ at the unary precision request n;*
3. *finite-window readback, requiring the diagonal observation to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W and dyadic readback row D before any regular-rational read is exported;*
4. *seal handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *row S to be reached only through the selected finite witness route n, μ, k, W, D and the displayed transport, route, package, and naming rows;*
5. *non-escape, requiring the selector to export no host-selected limit, quotient stream equality, countable choice data, ambient completeness principle, or unrecorded diagonal observation.*

Proof. Carrier admission is projection from the finite tuple in Definition 673.1. The input, precision, modulus, selected index, window, dyadic readback, seal, transport, continuation, provenance, and local naming rows are all displayed coordinates.

Modulus-selector exactness reads n and μ before the index row k is available. Thus the selected index is a row of the packet, fixed by the displayed modulus route, rather than an external choice of a subsequence.

The finite-window readback then routes the selected member through W and D. Since the window and dyadic rows are named before the readback is exported, a consumer cannot replace them by a quotient stream or by an unbounded host observation.

Finally the seal handoff and non-escape clauses inspect S, H, C, P, N. These rows carry the selected finite witness route into the RealUp seal and provide no coordinate for host-selected limits, choice data, ambient completeness, or an unrecorded diagonal observation. □

Theorem 673.3 (LimitSelector finite witness route)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet factors through the selected finite witness route*

$$n, \mu, k, W, D, S, H, C, P, N.$$

The consumer may inspect the precision request, displayed modulus, selected index, finite stream window, dyadic readback, RealUp seal, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local naming row. It receives no host-selected limit, no quotient-stream equality, no countable-choice selector, and no ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier finite witness route`

Proof. Start with the consumer route C. Its first reads are the precision request n and the modulus row μ, followed by the selected index k named by the modulus-selector obligation of Definition 673.2.

Next the consumer reads the finite observation data. The selected index opens only the displayed window W, and the dyadic row D is the readback attached to that window.

Finally the route reaches the seal row S and the BEDC structural rows H, C, P, N. The carrier has no second coordinate for a host-selected limit, quotient-stream equality, countable choice, or ambient completeness, so every diagonal-limit read remains inside the selected finite witness route. □

Theorem 673.4 (LimitSelector observation-budget handoff)**.** *Let*

$$L = (R, n, \mu, k, W, D, S, H, C, P, N)$$

be an accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet, and let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet share the displayed precision request, finite window, dyadic readback, RealUp seal, componentwise transport, continuation route, package provenance, and local naming rows* n, W, D, S, H, C, P, N*. Then every diagonal-limit consumer of* L *that reaches the shared rows factors through the same finite observation-budget route before it reads the RealUp seal.*

Proof. First project the selected route of L by Definition 673.1 and Definition 673.2. The selector supplies n, μ, k before the finite window W and dyadic readback D are available.

Next apply the finite witness route of Definition 673.3. Once W, D, S, H, C, P, N are displayed, the diagonal-limit consumer has no route around the named window and readback rows.

Finally read the shared observation-budget packet. The seal-route totality and consumer exactness theorems for $\mathsf{RealObservationBudget}^{\uparrow}$, Definition 611.17 and Definition 611.9, identify the same finite budget route through the precision request, window, dyadic ledger, seal, transport, continuation, provenance, and naming rows. □

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier observation budget handoff`

Theorem 673.5 (LimitSelector Real seal non-escape)**.** *For an accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet, the RealUp seal row* S *is reachable only after the selector has displayed* n, μ, k, W, D *and the transport, continuation, provenance, and local naming rows* H, C, P, N*. No consumer may read* S *as a host-selected limit, quotient stream, countable-choice selector, or ambient completeness witness.*

Proof. Use carrier admission first. In Definition 673.1, the seal row S is a coordinate following the precision request, modulus row, selected index, finite window, and dyadic readback.

The local NameCert obligations then impose the seal handoff. By Definition 673.2, the seal row is reached only through the selected finite witness route and the displayed structural rows.

The observation-budget seal boundary gives the Real-facing side of the same claim. Definition 611.5 places a budget seal after its finite observation rows, and Definition 611.6 excludes quotient streams, host real equality, hidden limit selection, unbounded schedules, and ambient completeness from the shared route. □

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier real seal nonescape`

Theorem 673.6 (LimitSelector shared budget Real seal determinacy)**.** *Let*

$$L = (R, n, \mu, k, W, D, S, H, C, P, N)$$

be an accepted $\mathsf{LimitSelector}^{\uparrow}$ *packet, and let an accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *packet share the rows* n, W, D, S, H, C, P, N*. Then the* $\mathsf{Real}^{\uparrow}$ *seal row reached by any diagonal-limit consumer of* L *is determined by the shared observation-budget route. The consumer cannot read a second seal, a host-selected limit, quotient stream equality, or an ambient completeness route.*

Proof. Project the $\mathsf{LimitSelector}^{\uparrow}$ carrier. The only seal coordinate of L is the displayed row S, and it appears after the precision, selector, finite window, dyadic readback, transport, route, provenance, and naming rows.

Apply Definition 673.4. The accepted $\mathsf{RealObservationBudget}^{\uparrow}$ packet shares n, W, D, S, H, C, P, N, so the diagonal-limit consumer reaches the seal through the same finite observation-budget route before any Real-facing read is available.

Apply Definition 673.5. The seal cannot be read as a host-selected limit, quotient stream, countable-choice selector, or ambient completeness witness.

Finally use the RealObservationBudget seal boundary rows already cited above, namely Definition 611.5 and Definition 611.6. They identify the shared route as the finite seal boundary, leaving no second seal or ambient route for the diagonal-limit consumer. □

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier shared budget real seal determinacy`

Theorem 673.7 (LimitSelector diagonal consumer exhaustion)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet is exhausted by the carrier rows of Definition 673.1, the local obligations of Definition 673.2, the finite witness route of Definition 673.3, the observation-budget handoff of Definition 673.4, and the seal non-escape row of Definition 673.5. The consumer receives exactly the displayed selector, window, readback, seal, transport, route, provenance, and naming rows, with no host-selected limit or ambient completeness coordinate.*

Proof. Begin with the selector packet. The carrier and local obligations display the complete finite surface $R, n, \mu, k, W, D, S, H, C, P, N$ and require the selected index to be read from the precision and modulus rows.

Next use the finite witness route and the observation-budget handoff. Together they force every diagonal read through the selected index, the finite window, the dyadic readback, and the shared $\mathsf{RealObservationBudget}^{\uparrow}$ route before the seal row is consumed.

Finally apply seal non-escape. Since neither the selector carrier nor the observation-budget packet contains a coordinate for quotient streams, countable choice, selected host limits, or ambient completeness, the diagonal consumer has no remaining export beyond the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier diagonal consumer exhaustion`

Theorem 673.8 (LimitSelector scoped source rows)**.** *Every accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet exposes its scoped source as the displayed* BHist *carrier* $L = (R, n, \mu, k, W, D, S, H, C, P, N)$*, with the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{RegSeqRat}$ *input row* R*, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *route through* n, W, D, S, H, C, P, N*, the componentwise* hsame *transports and* Cont *route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}$ *row. No source row is supplied outside this displayed packet.*

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier scoped source rows`

Proof. Project the carrier of Definition 673.1. The accepted packet displays the BHist tuple $L = (R, n, \mu, k, W, D, S, H, C, P, N)$, so the regular-sequence source, precision request, modulus, selected index, finite stream window, dyadic readback, seal, transport, route, provenance, and naming rows are all visible coordinates.

Apply the local obligations of Definition 673.2. They require the selected index to be read from n and μ, and they require the finite-window readback to pass through W and D before the seal row is exported.

Use the observation-budget handoff of Definition 673.4. It identifies the shared budget route through n, W, D, S, H, C, P, N, while Definition 673.7 exhausts the remaining diagonal-consumer reads by the same displayed packet. □

Theorem 673.9 (LimitSelector modulus-seal scope)**.** *For an accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet, the selector modulus* μ *and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *read only the displayed finite window route*

$$n, \mu, k, W, D, S$$

and the observation-budget route carried by H, C, P, N*. The seal cannot read an unrecorded stream observation, an external selector, or any ambient completion principle.*

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier modulus seal scope`

Proof. Begin with the finite witness route of Definition 673.3. The precision request n and displayed modulus μ determine the selected index row k, after which the consumer reads the finite window W and dyadic readback D.

The seal row is then governed by Definition 673.5. That theorem places S after n, μ, k, W, D and the structural rows H, C, P, N, so the seal has no independent route to stream data or host completeness data.

Finally use Definition 673.4. The shared $\mathsf{RealObservationBudget}^{\uparrow}$ route accounts for the finite observation-budget side of the same seal read, and Definition 673.7 leaves no further consumer coordinate. □

Theorem 673.10 (LimitSelector non-escape scope)**.** *The scoped* $\mathsf{LimitSelector}^{\uparrow}$ *packet exports no quotient stream, no host limit selection, no ambient completeness principle, and no open-cover compactness datum. Its exported content is exhausted by the carrier, finite witness route, observation-budget handoff, Real seal non-escape, and diagonal consumer exhaustion theorems of this chapter.*

Lean4 ✓ `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier nonescape scope`

Proof. Read the packet through Definition 673.1 and Definition 673.2. The only exported coordinates are $R, n, \mu, k, W, D, S, H, C, P, N$ and the finite obligations attached to them.

Apply Definition 673.3, Definition 673.4, and Definition 673.5. These results route every diagonal-limit and Real-seal read through the selected finite witness route and the displayed observation-budget rows.

Conclude with Definition 673.7. Since the exhausted surface has no coordinate for quotient streams, host-selected limits, ambient completeness, or open-cover compactness, none of those objects can be exported by the scoped selector packet. □

Theorem 673.11 (LimitSelector Real StreamName observation lock)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{LimitSelector}^{\uparrow}$ *packet reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed chain*

$$R,\ n,\ \mu,\ k,\ W,\ D,\ \mathsf{RealObservationBudget}^{\uparrow},\ S.$$

Here R *is the* $\mathsf{RegSeqRat}$ *input row,* n *is the unary precision request,* μ *is the selector modulus,* k *is the selected index,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window over the sibling interface of chapter 126,* D *is the dyadic readback, the shared finite route is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *handoff, and* S *is the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. No diagonal-limit consumer can bypass this chain by reading a host-selected limit, quotient stream equality, countable-choice selector, ambient completeness witness, or unrecorded stream observation.*

Proof. Start with the finite witness route of Definition 673.3. It forces the consumer to read the precision request, selector modulus, selected index, finite stream window, dyadic readback, seal row, and structural rows as displayed coordinates of the same packet.

Next apply the observation-budget handoff of Definition 673.4. Once the finite $\mathsf{StreamName}^{\uparrow}$ window from chapter 126 and the dyadic readback are fixed, the shared $\mathsf{RealObservationBudget}^{\uparrow}$ route is the only observation-budget route available before the seal is consumed.

Now use the Real seal non-escape theorem of Definition 673.5. It places the $\mathsf{Real}^{\uparrow}$ seal from chapter 38 after n, μ, k, W, D and the displayed structural rows, and it excludes host-selected limits, quotient streams, countable-choice selectors, and ambient completeness witnesses.

Finally apply diagonal consumer exhaustion from Definition 673.7. The exhausted carrier has no remaining coordinate outside the listed chain, so every diagonal-limit consumer reaches the Real seal through the RegSeqRat row, unary request, selector modulus, selected index, StreamName window, dyadic readback, observation-budget route, and seal row. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{LimitSelector}^{\uparrow}$ packet is generated from a RegSeqRat input row, a unary precision request, a displayed modulus, a selected finite index, a StreamName window, dyadic readback, RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 673.4 and Definition 673.5 bridge the public finite selector surface to the displayed observation-budget route and Real seal boundary over the RegSeqRat, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LimitSelector}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LimitSelectorUp.LimitSelectorCarrier\_finite\_witness\_route`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, real completeness, quotient-stream equality, host real equality, countable choice, host-selected limits, or a public standard bridge.

Upgrade path: No further paper-theory closure upgrade remains; Lean verification and bridge checking remain on the verification axis.

Chapter 674

A Concrete Naming Certificate for $\mathsf{GroundLoopBoundary}^{\uparrow}$

The $\mathsf{GroundLoopBoundary}^{\uparrow}$ packet realizes the boundary isolated in chapter 9461. It records the closed ground loop as finite mark and mark-sameness rows, while the meta side is recorded only as refusal rows for truth-total reflection, host runtime export, and host truth predicates. The packet therefore gives downstream chapters a concrete NameCert surface for citing the exact place where ground distinction closes and meta-truth closure remains unavailable.

$\mathsf{GroundLoopBoundary}^{\uparrow}$ supplies the ground-row subpacket consumed by $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ in chapter 586; the latter adds the broader structural meta-description boundary.

Its mark inventory and positive sameness rows are read through the finite boolean mark packet of chapter 32: BMark supplies the two displayed endpoints, while msame keeps the diagonal rows visible before any reflection-boundary consumer can inspect R, H, C, P, N.

Definition 674.1 (Ground-loop boundary carrier)**.** *A* $\mathsf{GroundLoopBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, S, X, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the two-row* BMark *inventory exposing exactly* b0 *and* b1*;*
2. *S is the positive* msame *inventory exposing exactly* $\mathsf{msame}(\mathsf{b0}, \mathsf{b0})$ *and* $\mathsf{msame}(\mathsf{b1}, \mathsf{b1})$*;*
3. *X is the cross-mark refusal row, recording that no generated row identifies* b0 *with* b1 *or* b1 *with* b0*;*
4. *R is the reflection-boundary row, recording structural self-description as permitted data and truth-total reflection as refused data;*
5. *H records the componentwise* hsame *transports for the mark inventory, positive sameness rows, cross-mark refusal rows, reflection-boundary row, provenance, and local naming row;*
6. *C records the displayed* Cont *routes used to read M, replay S and X, transport the boundary row R, and return the same finite packet to every consumer;*
7. *P is the* Pkg *provenance over* BHist*,* BMark*,* msame*,* hsame*,* Cont*, reflection-boundary rows, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundLoopBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a truth-total reflection certificate, host runtime, host truth predicate, or meta-level classifier that turns structural self-description into BEDC truth.

Lean4 (definition) `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier`

Theorem 674.2 (GroundLoopBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{GroundLoopBoundary}^{\uparrow}}$ *surface for the carrier* $G = (M, S, X, R, H, C, P, N)$ *consists exactly of carrier admission for Definition 674.1, the two displayed* BMark *rows in* M*, the two positive* msame *rows in* S*, cross-mark non-generation and refusal in* X*, the reflection-boundary row* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier namecert obligations`

Proof. Carrier admission is projection from the finite tuple of Definition 674.1. The accepted object exposes only the rows M, S, X, R, H, C, P, N.

The ground rows are exhausted by M, S, X. The row M displays the two mark constructors, S displays the two positive primitive sameness rows, and X records that no generated cross-mark sameness row is admitted.

The reflection row is R. It permits structural self-description only as a named boundary row and records refusal of truth-total reflection, so it cannot be read as a truth classifier for the host calculus.

Finally, the transport, replay, provenance, and naming obligations are exactly H, C, P, N. No further row remains from which a consumer could recover a host runtime, host truth predicate, or meta-level classifier beyond the displayed boundary. □

Theorem 674.3 (Ground-loop boundary meta non-escape)**.** *Every consumer of an accepted* $\mathsf{GroundLoopBoundary}^{\uparrow}$ *packet factors through the finite rows* M, S, X, R, H, C, P, N*. The packet exports the closed ground distinction between* b0 *and* b1 *together with the positive* msame *rows and cross-mark refusal, but it exports no truth-total reflection certificate, host runtime, host truth predicate, or meta-level truth classifier.*

Lean4 ✓ `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier meta nonescape`

Proof. Let $G = (M, S, X, R, H, C, P, N)$ be accepted. Any permitted read of the ground loop uses M for the two marks, S for positive sameness, and X for the refusal of generated cross-mark sameness.

Any permitted read of the reflection boundary uses R and is transported only by H and replayed only by C. The provenance and local name remain the rows P and N.

Since the carrier inventory has no coordinate for truth-total reflection, a host runtime, a host truth predicate, or a meta-level truth classifier, no downstream consumer can project such data from the packet. The only exported boundary is the finite distinction between closed ground generation and open meta-truth refusal. □

Theorem 674.4 (Ground-loop boundary scoped kernel route)**.** *Every accepted* $\mathsf{GroundLoopBoundary}^{\uparrow}$ *packet binds its displayed rows to the exact BEDC primitive scope: the mark inventory* M *is scoped to* BMark*, the positive sameness rows* S *and cross-mark refusal row* X *are scoped to* msame*, transport* H *is scoped to* hsame*, replay* C *is scoped to* Cont*, provenance* P *is scoped to* Pkg*, and local naming* N *is scoped to* NameCert*.*

Lean4 ✓ `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier scoped kernel route`

Proof. Project the accepted carrier $G = (M, S, X, R, H, C, P, N)$ through Definition 674.1. Each row is named by the primitive that stores it: marks by BMark, mark sameness and refusal by msame, transport by hsame, replay by Cont, provenance by Pkg, and local naming by NameCert.

The meta non-escape theorem Definition 674.3 exhausts every permitted consumer through these rows. Since the carrier has no further coordinate for host truth, runtime export, or a meta-level classifier, the listed primitive bindings are the whole scope of the packet. □

Theorem 674.5 (Ground-loop boundary public interface)**.** *Every downstream consumer may cite the* GroundLoopBoundary$^{\uparrow}$ *public interface exactly as the finite tuple of mark inventory, positive* msame *rows, cross-mark refusal, reflection-boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. This interface exports no truth-total reflection, host runtime, host truth predicate, or meta-level truth classifier beyond the displayed boundary rows.*

Lean4 ✓ `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier public interface`

Proof. Project the accepted carrier through Definition 674.1. The public rows are precisely M, S, X, R, H, C, P, N, and Definition 674.4 binds them to the primitive BEDC scope used by downstream consumers.

Use Definition 674.3. Since no coordinate carries truth-total reflection, a host runtime, a host truth predicate, or a meta-level truth classifier, the public interface can export only the displayed boundary tuple and its local naming route. □

Theorem 674.6 (Ground-loop boundary mature consumer exhaustion)**.** *Every worked consumer of an accepted* GroundLoopBoundary$^{\uparrow}$ *packet factors through the rows M, S, X, R, H, C, P, N of Definition 674.1 and through the sibling* ReflectionLimitBoundary$^{\uparrow}$ *route when it needs the broader reflection-limit boundary. No such consumer obtains truth-total reflection, a host runtime, a host truth predicate, or a meta-level truth classifier.*

Lean4 ✓ `BEDC.Derived.GroundLoopBoundaryUp.GroundLoopBoundaryCarrier mature consumer exhaustion`

Proof. The NameCert obligations of Definition 674.2 expose exactly the rows M, S, X, R, H, C, P, N. Thus a worked consumer starts from the mark inventory, positive msame rows, cross-mark refusal, reflection-boundary row, transport, replay, provenance, and local naming.

The meta non-escape theorem Definition 674.3 rules out every projection to truth-total reflection, host runtime, host truth predicate, or a meta-level truth classifier. The scoped kernel route of Definition 674.4 then binds each remaining read to BMark, msame, hsame, Cont, Pkg, and NameCert.

Finally, the public interface theorem Definition 674.5 is the only exported consumer surface of this packet. A consumer that needs the larger reflection-limit boundary must pass to the sibling ReflectionLimitBoundary$^{\uparrow}$ route described in chapter 586; it still receives only the displayed boundary rows and not a host truth object. □

Closure status for NameCert$_{\text{GroundLoopBoundary}^{\uparrow}}$.

Constructive story: A GroundLoopBoundary$^{\uparrow}$ packet is generated from the two BMark rows, the two positive msame rows, cross-mark refusal rows, a reflection-boundary row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GroundLoopBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 674.1, Definition 674.2, Definition 674.3, Definition 674.4, Definition 674.5, and Definition 674.6, with the displayed TasteGate marker above, bind the worked-consumer exhaustion surface between closed ground distinction and open meta-truth refusal.

Formal verification: FormalStatus(NameCert$_{\text{GroundLoopBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean-side formal verification, truth-total reflection, a host runtime export, a host truth predicate, or a bridge to the internal CIC-star structure.

Upgrade path: Remaining work is Lean-side formal verification and bridge checking for the displayed mature consumer surface.

Chapter 675

A Concrete Naming Certificate for $\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}$

The $\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}$ packet records the finite BEDC data consumed by the prefix-functor identity naturality bridge of Definition 9363.4. It packages the prefix history, source and target object histories, the displayed hom history, the left and right empty-composite continuation rows, the prefix-lifted hom carriers, the component row used by $\mathsf{NatTrans}^{\uparrow}$, the hsame transport rows, the $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ certificate rows, the continuation ledger, provenance, and local naming. The packet is a concrete NameCert surface; it is not a host category, a quotient morphism class, or a naturality proof imported from an external categorical library. Its carrier consumes the Category hom surface of chapter 61, the prefix action row of chapter 62, and the component comparison row of chapter 63; these sibling rows are the source of the local F and T coordinates rather than decorative category terminology.

Definition 675.1 (Prefix functor identity-naturality bridge carrier)**.** *A* $\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (p, a, b, f, L, R, U, V, E, H, F, T, C, P, N)$$

with the following displayed rows:

1. *p is the unary prefix history read by the prefix-functor route;*
2. *a and b are the unary source and target object histories;*
3. *f is the displayed hom history carrying* $\mathsf{CategoryHomCarrier}(a, b, f)$*;*
4. *L records the left empty-composite continuation row* $\mathsf{Cont}(\mathsf{e}, f, L)$*;*
5. *R records the right empty-composite continuation row* $\mathsf{Cont}(f, \mathsf{e}, R)$*;*
6. *U is the prefix-lifted hom-carrier row from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *obtained from L;*
7. *V is the prefix-lifted hom-carrier row from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$ *obtained from R;*
8. *E is the empty identity component row used by the prefix functor identity law at each unary object;*
9. *H records componentwise* hsame *transports comparing U and V and reading them back to the original carrier f;*
10. *F is the local* $\mathsf{Functor}^{\uparrow}$ *row for the prefix functor on the checked unary-continuation* $\mathsf{Cat}^{\uparrow}$ *packet;*

11. *T is the local* $\mathsf{NatTrans}^{\uparrow}$ *row for the identity naturality component surface;*
12. *C records the displayed* Cont *routes for left unit, right unit, prefix lift, component readback, and consumer handoff;*
13. *P is the* Pkg *provenance over* BHist, $\mathsf{Cat}^{\uparrow}$, $\mathsf{CategoryHomCarrier}$, hsame, Cont, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, *and local certificate rows;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows $p, a, b, f, L, R, U, V, E, H, F, T, C, P, N$. It has no coordinate for an external category object, quotient morphism equality, host natural-transformation equality, or an unrecorded naturality square.

Theorem 675.2 (Prefix functor identity-naturality NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PrefixFunctorIdentityNat}}$ *surface for Definition 675.1 consists of seven finite obligation rows:*

1. *carrier admission, requiring every accepted object to display the packet $K = (p, a, b, f, L, R, U, V, E, H, F, T, C, P,$*
2. *category source binding, requiring a, b, f to enter through the checked* $\mathsf{Cat}^{\uparrow}$ *hom-carrier row* $\mathsf{CategoryHomCarrier}(a, b, f)$*;*
3. *left and right empty-composite binding, requiring L and R to be the displayed* $\mathsf{Cont}(\mathsf{e}, f, L)$ *and* $\mathsf{Cont}(f, \mathsf{e}, R)$ *continuation rows consumed by the bridge;*
4. *prefix-lift exactness, requiring U and V to be the two prefix-lifted hom-carrier rows from* $\mathsf{append}(p, a)$ *to* $\mathsf{append}(p, b)$*;*
5. *identity-naturality comparison, requiring H to expose* $\mathsf{hsame}(U, V)$ *and the readback transports from the prefix-lifted rows to f;*
6. *functor and natural-transformation coverage, requiring F and T to be the local* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *rows that consume the same prefix, identity component, hom-carrier, and transport data;*
7. *ledger non-escape, requiring all public reads to pass through* BHist, $\mathsf{CategoryHomCarrier}$, hsame, Cont, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, Pkg, *and* $\mathsf{NameCert}$, *excluding external category libraries, quotient morphisms, and host naturality equality.*

Proof. Carrier admission is projection from Definition 675.1. The rows $p, a, b, f, L, R, U, V, E, H, F, T, C, P, N$ display the full prefix, hom-carrier, empty-composite, lifted-carrier, component, transport, functor, natural-transformation, route, provenance, and local naming packet.

The category source binding reads a, b, f through $\mathsf{CategoryHomCarrier}(a, b, f)$. The left and right rows L and R are the only displayed empty-composite continuation rows, so the prefix-lifted rows U and V are forced to use the same original hom history f.

Prefix-lift exactness then applies the bridge in Definition 9363.4. It supplies the two hom-carrier rows over $\mathsf{append}(p, a)$ and $\mathsf{append}(p, b)$ and the hsame comparison between them.

The functor and natural-transformation rows F and T consume exactly those carriers, the empty identity component row E, and the componentwise transports in H. Since the packet has no external categorical coordinate, the ledger non-escape clause follows by inventory of the displayed rows. □

Theorem 675.3 (Prefix functor identity-naturality public handoff)**.** *Every downstream consumer of an accepted* $\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}$ *packet receives only the finite rows*

$$p, a, b, f, L, R, U, V, E, H, F, T, C, P, N.$$

The consumer may inspect the unary prefix, the original $\mathsf{Cat}^{\uparrow}$ *hom-carrier, the left and right empty-composite continuation rows, the two prefix-lifted hom carriers, their* hsame *comparison, the empty component row, the* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *certificate rows, displayed* Cont *routes, provenance, and local naming. It may not extract a host category, quotient morphism equality, ambient functor category, or naturality equality outside the displayed BEDC packet.*

Proof. Project the accepted carrier. The displayed rows are exactly $p, a, b, f, L, R, U, V, E, H, F, T, C, P, N$.

Use the NameCert obligations to order the public read through category source binding, empty-composite binding, prefix-lift exactness, identity-naturality comparison, functor coverage, natural-transformation coverage, and ledger non-escape.

The only comparison row exposed to the consumer is H, which records the componentwise hsame transports and the readback from the two lifted carriers to f. Because no host category, quotient morphism class, or external naturality equality appears as a coordinate of the carrier, the handoff cannot project any such object. □

Theorem 675.4 (Prefix functor identity-naturality category-functor route)**.** *Every accepted* PrefixFunctorIdentityNa *packet factors through the sibling chapter surfaces chapter 61, chapter 62, and chapter 63 before exporting the identity-naturality handoff of Definition 675.3. The* a, b, f *rows first expose* $\mathsf{CategoryHomCarrier}(a, b, f)$*, the* F *row then exposes the* $\mathsf{Functor}^{\uparrow}$ *prefix action, and the* T *row exposes the* $\mathsf{NatTrans}^{\uparrow}$ *component surface consumed by the public handoff.*

Proof. Project the carrier from Definition 675.1. The rows a, b, f are not free categorical parameters; they are displayed with the $\mathsf{CategoryHomCarrier}(a, b, f)$ row, so the route enters through the category sibling surface of chapter 61.

Next read the certificate rows F and T. The row F is the local $\mathsf{Functor}^{\uparrow}$ certificate for the prefix action on the same checked unary-continuation category packet, while T is the local $\mathsf{NatTrans}^{\uparrow}$ certificate for the identity component comparison. These rows force the handoff through chapter 62 and chapter 63.

The public handoff then reads $p, a, b, f, L, R, U, V, E, H, F, T, C, P, N$ through the same displayed packet. Since the carrier has no coordinate for an external category object, ambient functor category, quotient morphism equality, or host naturality equality, the sibling category, functor, and natural-transformation rows are substantive dependencies rather than optional labels. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}$ packet is generated from a unary prefix, original category hom rows, left and right empty-composite continuation rows, two prefix-lifted hom carriers, identity component rows, componentwise hsame transport, $\mathsf{Functor}^{\uparrow}$ and $\mathsf{NatTrans}^{\uparrow}$ rows, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 675.1, Definition 675.2, and Definition 675.3 bind the finite carrier, NameCert obligations, and public handoff for the prefix-functor identity naturality bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixFunctorIdentityNaturalityBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, external category interfaces, quotient morphism equality, ambient functor categories, host naturality equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-lift exactness, identity-naturality comparison, functor and natural-transformation coverage, ledger provenance, and public handoff for the same finite packet.

Chapter 676

A Concrete Naming Certificate for $\mathsf{RegularCauchyProductBudget}^{\uparrow}$

The $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ packet records the finite product-window ledger needed by Real-and-Completion multiplication consumers. It starts with two $\mathsf{RegSeqRat}^{\uparrow}$ source names, opens only displayed $\mathsf{StreamName}^{\uparrow}$ windows, reads the product row through $\mathsf{DyadicRatCore}^{\uparrow}$, accounts for the finite error split through $\mathsf{FiniteErrorBudget}^{\uparrow}$, hands the result back as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and seals the completion-facing read through $\mathsf{Real}^{\uparrow}$. The packet is a local product-budget certificate; it is not quotient stream equality, host real multiplication, an ambient completeness theorem, or a selected host limit.

The product-budget carrier consumes the sibling routes of chapter 126, chapter 291, chapter 385, and chapter 301. These links are substantive: paired finite stream windows supply the source observations, DyadicRatCore supplies the displayed product row, FiniteErrorBudget supplies the finite split ledger, and RegSeqRat receives the product handoff.

Definition 676.1 (RegularCauchyProductBudget carrier)**.** *A* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source packets whose product is being read;*
2. *W_A and W_B are the paired finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened from the two sources at the precision requested by the product route;*
3. *D_A and D_B are the dyadic endpoint and radius rows read from those two windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product row computing the finite dyadic product observation from D_A and D_B;*
5. *E is the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *product ledger assigning the source-window radii, product rounding allowance, and remaining readback budget;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the dyadic product row and the finite budget ledger;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row through which a completion or real-algebra consumer reads the product candidate;*

8. *H stores componentwise* hsame *transports for the source rows, windows, dyadic rows, budget ledger, readback, seal, package provenance, and local naming row;*
9. *C stores the displayed* Cont *routes for source inspection, paired-window admission, dyadic product read, finite-budget exactness, regular readback, and RealUp seal handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C,$

Lean4 (definition) `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudgetUp`

Theorem 676.2 (RegularCauchyProductBudget falsifiable prediction)**.** *Every accepted* RegularCauchyProductBudg *carrier has a falsifiable finite-product prediction: if a candidate packet omits either paired* $\mathsf{StreamName}^{\uparrow}$ *window or omits the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row, then it cannot export the same* $\mathsf{RegSeqRat}^{\uparrow}$ *product handoff and* $\mathsf{Real}^{\uparrow}$ *seal as the carrier of Definition 676.1.*

Proof. Project the carrier coordinates. The two source windows W_A, W_B are the only rows from which the dyadic source observations D_A, D_B can be read, so deleting either window leaves no displayed route to the product row D.

Project the finite ledger coordinate E. The readback row R is downstream of D and E, and the seal row S is downstream of R in the same finite packet.

Therefore a candidate that lacks either paired window or the budget ledger cannot preserve both the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal. The Lean follow-up must provide `Nontrivial RegularCauchyProductBudgetUp` and `FieldFaithful RegularCauchyProductBudgetUp` in `TasteGate.lean`. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget falsifiable prediction`

Theorem 676.3 (RegularCauchyProductBudget independence witness)**.** *The* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *packet is not bijective, as a local* NameCert *surface, to the sibling slugs bounded-cauchy-cover, observablecauchyseal, regularcauchymerge, or regularcauchyswapbisimulation. It has two source windows, a dyadic product row, and a finite error-budget row, while those siblings respectively expose one budget-to-cover route, one observation seal route, two cofinal selector routes, or selector-swap symmetry without a dyadic multiplication budget.*

Proof. Compare carrier inventories. The product-budget carrier contains the two source rows, paired $\mathsf{StreamName}^{\uparrow}$ windows, dyadic source rows, one dyadic product row, one finite error-budget row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback, and one $\mathsf{Real}^{\uparrow}$ seal.

The bounded-cauchy-cover sibling has a single budget-to-cover route rather than paired product windows. The observablecauchyseal sibling has a seal route rather than the dyadic product and finite error split.

The regularcauchymerge sibling records two cofinal selector routes, and the regularcauchyswapbisimulation sibling records selector-swap symmetry. Neither supplies the dyadic multiplication budget row required by the product carrier.

Thus no bijection can preserve the displayed coordinates and the local NameCert obligations at once. The Lean follow-up must provide `Nontrivial RegularCauchyProductBudgetUp` and `FieldFaithful RegularCauchyProductBudgetUp` in `TasteGate.lean`. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget independence witness`

Falsifiable prediction: Any accepted product-budget carrier omitting either paired StreamName window or the FiniteErrorBudget row cannot export the same RegSeqRat handoff and RealUp seal.

Independence witness: bounded-cauchy-cover, observablecauchyseal, regularcauchymerge, regularcauchyswapbisimulation: ProductBudget is not bijective to these siblings because it has two source windows plus dyadic product and finite error-budget rows, while the siblings respectively have one budget-to-cover route, one observation seal route, two cofinal selector routes, or selector-swap symmetry without dyadic multiplication budget.

Theorem 676.4 (RegularCauchyProductBudget window admission)**.** *Every accepted* RegularCauchyProductBudget *carrier admits the paired source windows only through the two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows and their* $\mathsf{StreamName}^{\uparrow}$ *windows. More precisely, for*

$$B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N),$$

any product-window read first exposes the source packets A and B, then the finite windows W_A and W_B, and only then the dyadic source rows D_A and D_B.

Proof. Project the carrier of Definition 676.1. The source packets and paired windows are explicit coordinates, so the first available product-window data are A, B, W_A, W_B.

The dyadic source rows D_A and D_B are listed after the paired windows and are read through the same continuation route. Thus a consumer cannot obtain a dyadic product observation before the two finite windows have been admitted.

The remaining rows D, E, R, S, H, C, P, N consume that displayed window data. Since the carrier has no coordinate for an unbounded stream observation or a quotient representative, all product-window admission is confined to the finite rows already named. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudgetCarrier window admission`

Theorem 676.5 (RegularCauchyProductBudget ledger exactness)**.** *The finite product-budget ledger of an accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier is exactly the row E applied to the dyadic source rows D_A, D_B and the product row D. It accounts for the two source-window radii, the dyadic product rounding row, and the remaining readback allowance before any* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback is exported.*

Proof. Begin with the admitted windows from Definition 676.4. They supply the two dyadic source rows D_A and D_B and therefore fix the inputs of the dyadic product row D.

Read the budget coordinate E. Its role in the carrier is the finite product ledger, so it binds the two source-window radii, the product rounding allowance, and the remaining readback budget to the same dyadic product observation.

The readback row R is listed after E and is routed through the same package provenance and continuation data. Hence no product readback can be exported without first passing through the displayed finite ledger. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget ledger exactness`

Theorem 676.6 (RegularCauchyProductBudget bilinear finite-window route)**.** *Every accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier reads both source windows* W_A, W_B*, both dyadic source rows* D_A, D_B*, the product row* D*, and the finite ledger* E *as one bilinear finite window before exporting the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *or the* $\mathsf{Real}^{\uparrow}$ *seal row* S*. In particular, neither source window may be consumed alone, and no product-budget consumer may reach* R *or* S *before the symmetric route through*

$$A, B, W_A, W_B, D_A, D_B, D, E$$

has been assembled.

Proof. Apply Definition 676.4. It exposes the two source packets together with the paired windows W_A, W_B, and it reads the dyadic source rows D_A, D_B only after both finite windows are visible.

Next apply Definition 676.5. The dyadic product row D and the finite ledger E are bound to the two dyadic source rows in the same carrier, so the product budget consumes the two windows symmetrically rather than as independent one-sided reads.

The later $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 676.10 and the $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 676.11 both occur after this assembled window. Hence the carrier exports no readback or seal that bypasses the bilinear finite-window route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget bilinear window`

Theorem 676.7 (RegularCauchyProductBudget symmetric source-window inversion)**.** *Let*

$$B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier. Any consumer that reaches the product row* D *or the finite ledger row* E *can recover that both source windows* W_A *and* W_B *were admitted through the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *route. Neither one-sided source window suffices to license the dyadic product row, because the carrier has no coordinate for a product observation built from only* W_A *or only* W_B*.*

Proof. Project the carrier of Definition 676.1. The paired source windows W_A, W_B and the dyadic source rows D_A, D_B occur before the product row D and the ledger row E, so a read of D or E also identifies the source-window coordinates from which they were assembled.

Use window admission from Definition 676.4. It admits W_A and W_B together through the $\mathsf{RegSeqRat}^{\uparrow}$ source rows and their $\mathsf{StreamName}^{\uparrow}$ finite windows before either dyadic source row is consumed.

Use ledger exactness from Definition 676.5 and the bilinear route of Definition 676.6. The product row D and ledger row E are bound to both D_A and D_B, which in turn are bound to both W_A and W_B.

Thus any consumer at D or E inverts the route back to the paired source-window admission. Since no one-sided product coordinate appears in the carrier tuple, a single window alone cannot license the dyadic product row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudgetCarrier symmetric source window inversion`

Theorem 676.8 (RegularCauchyProductBudget source coverage)**.** *Every accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier reaching either the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *or the* $\mathsf{Real}^{\uparrow}$ *seal row* S *must expose both source packets* A, B*, both* $\mathsf{StreamName}^{\uparrow}$ *windows* W_A, W_B*, both dyadic source rows* D_A, D_B*, and the shared* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *row* E*. No one-sided product read and no hidden host multiplication route covers the same exported seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget source coverage`

Proof. Apply Definition 676.4. It exposes the source packets A, B together with the paired finite windows W_A, W_B before either dyadic source row can be consumed.

Apply Definition 676.5. The finite ledger E is tied to both dyadic source rows D_A, D_B and to the product row D, so a product read cannot cover only one source side.

Use Definition 676.7. Any consumer that has reached the downstream product route can invert the route back to the paired source-window admission.

The rows R, S, H, C, P, N add readback, seal, transport, replay, provenance, and local naming data only after the same paired source route is present. Since the carrier has no coordinate for one-sided multiplication or host multiplication, neither can cover the exported seal. □

Theorem 676.9 (RegularCauchyProductBudget product-closure budget)**.** *Every accepted* RegularCauchyProductB *carrier closes the finite product budget needed for a* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, the paired source windows* W_A, W_B*, the dyadic product row* D*, and the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger* E *force the readback row* R *and seal row* S *to use the same finite product budget carried by*

$$A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N.$$

No completion or real-algebra consumer may replace this route by host real multiplication, quotient stream equality, an ambient completeness principle, a hidden rounding oracle, or a selected host limit.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget product closure budget`

Proof. Begin with window admission. By Definition 676.4, the product route first exposes the two $\mathsf{RegSeqRat}^{\uparrow}$ source rows and the paired $\mathsf{StreamName}^{\uparrow}$ windows, so D_A and D_B are read only through displayed finite observations.

Next use ledger exactness. By Definition 676.5, the dyadic product row D and the finite product ledger E are bound to those same source windows before any product readback can be exported.

Now attach the readback and seal. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff of Definition 676.10 consumes D and E through the structural rows H, C, P, N, and the $\mathsf{Real}^{\uparrow}$ seal boundary of Definition 676.11 reads S only after that handoff has been assembled.

Thus the product readback and the Real seal share one finite product budget. Since the carrier has no coordinate for host multiplication, quotient stream equality, ambient completeness, hidden rounding, or selected host limits, no consumer can bypass the displayed route. □

Theorem 676.10 (RegularCauchyProductBudget RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback exported by an accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier factors through the finite route*

$$A, B, W_A, W_B, D_A, D_B, D, E, R, H, C, P, N.$$

The handoff preserves the same source rows, paired windows, dyadic product observation, product-budget ledger, hsame *transports,* Cont *routes, package provenance, and local naming row.*

Proof. Use window admission to identify the two source rows and their finite windows. This fixes A, B, W_A, W_B, D_A, D_B as the only source data available to the product route.

Use ledger exactness to identify the dyadic product row and the finite budget ledger. The product readback must consume D and E before the row R is available.

Finally inspect the structural rows H, C, P, N. They preserve the route componentwise under hsame and Cont transport and repack it as the local naming surface, so the exported $\mathsf{RegSeqRat}^{\uparrow}$ handoff is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget regseqrat handoff`

Theorem 676.11 (RegularCauchyProductBudget Real seal boundary)**.** *Every completion-facing or real-algebra read of an accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the seal row S after the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R has been produced from the finite product-budget ledger. The carrier exports no host real multiplication, quotient stream equality, ambient completeness principle, hidden rounding oracle, or selected host limit.*

Proof. Project the seal row S from the carrier. It appears after the readback row R, so the real-facing route reaches the seal only after the finite product handoff has been assembled.

Apply Definition 676.10. The readback R is already bound to the source rows, paired windows, dyadic product row, budget ledger, transports, routes, package provenance, and local naming row.

The carrier contains no coordinate for host multiplication, quotient stream equality, ambient completeness, hidden rounding, or selected host limits. Therefore the $\mathsf{Real}^{\uparrow}$ seal is a boundary around the displayed finite packet and exports no additional real object. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget real seal boundary`

Theorem 676.12 (RegularCauchyProductBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductBu}}$ *surface for Definition 676.1 consists of carrier admission, paired-window admission, dyadic product-row exactness, finite product-ledger exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from host multiplication, quotient streams, ambient completeness, hidden rounding, and selected host limits.*

Proof. Carrier admission is the tuple in Definition 676.1. It lists the two sources, paired windows, dyadic source rows, dyadic product row, product ledger, readback, seal, transports, routes, provenance, and local naming row.

Paired-window admission is Definition 676.4, and dyadic product-row exactness is the displayed use of D_A, D_B, D inside the same carrier.

Finite product-ledger exactness is Definition 676.5. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff exactness is Definition 676.10, and the $\mathsf{Real}^{\uparrow}$ seal boundary is Definition 676.11.

The structural obligations are the roles of H, C, P, N. They keep transport, continuation, package provenance, and the local certificate on the same displayed packet. Since no forbidden external coordinate is present, non-escape follows by inspection of the carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget namecert obligations`

Theorem 676.13 (RegularCauchyProductBudget checked-obligation readiness)**.** *For the accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier*

$$B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N),$$

the carrier row, paired-window admission, finite product-ledger exactness, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and non-escape clauses form the single Lean-facing obligation surface for the product-budget packet.*

Proof. First assemble the finite product route. Apply Definition 676.4 to expose the paired source windows, and apply Definition 676.5 to bind the dyadic product row to the finite ledger E.

Next attach the consumer handoff boundary. Apply Definition 676.10 for the $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and apply Definition 676.11 for the $\mathsf{Real}^{\uparrow}$ seal boundary.

Finally repack the structural obligations. Apply Definition 676.12; its transport, route, provenance, local naming, and non-escape clauses name exactly the same packet. Hence the listed clauses form one obligation surface for downstream Lean formalization. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget checked obligation readiness`

Theorem 676.14 (RegularCauchyProductBudget obligation scope)**.** *The carrier of Definition 676.1, paired-window admission, ledger exactness, product-closure budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, local* $\mathsf{NameCert}$ *obligations, and checked-obligation readiness form one obligation-closure surface over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FiniteErrorBudget}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The surface is the finite product-budget route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{FiniteErrorBudget}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

carried by the same packet $B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N)$*.*

Proof. Start with carrier admission from Definition 676.1. It fixes the source rows, paired stream windows, dyadic source rows, dyadic product row, finite ledger, readback row, Real seal, transports, routes, provenance, and local naming row as one finite packet.

Apply Definition 676.4, Definition 676.5, and Definition 676.9. These theorems order the paired $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ product row, and the $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger before the readback can be exported.

Next use Definition 676.10 and Definition 676.11. They attach the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal to that same displayed budget route.

Finally repack through Definition 676.12 and Definition 676.13. The rows H, C, P, N supply transport, continuation replay, provenance, and local naming for the same packet, and inspection shows no route through host real multiplication, quotient stream equality, hidden rounding, selected limits, or an unbounded stream. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget obligation scope`

Theorem 676.15 (RegularCauchyProductBudget dyadic multiplication scope)**.** *Every accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *product-budget packet must expose the paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *multiplication row, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *split ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *product handoff before any* $\mathsf{Real}^{\uparrow}$ *seal reads the product. For*

$$B = (A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N),$$

the admissible multiplication route is the displayed finite path

$$W_A, W_B \longmapsto D_A, D_B \longmapsto D \longmapsto E \longmapsto R \longmapsto S,$$

with componentwise hsame *transport and* Cont *replay carried by* H, C, P, N*. No product consumer may replace this route by host real multiplication, quotient stream equality, a hidden rounding oracle, or an uninspected completion seal.*

Proof. Induct over the displayed finite window rows. In the base read, the packet exposes the two source rows A, B and then the paired windows W_A, W_B; no dyadic source row is available before those finite observations are present.

In the window step, read D_A and D_B from the two windows and apply the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ multiplication row D. This keeps the product as a finite dyadic observation rather than a host real product.

Next split the error budget through E. The ledger accounts for the two source-window radii, the product rounding allowance, and the remaining readback budget before the product can be handed to R.

Finally transport the assembled row through H, C, P, N and only then pass it to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and the $\mathsf{Real}^{\uparrow}$ seal S. Since the carrier has no coordinate for host multiplication, quotient stream equality, hidden rounding, or an uninspected completion seal, every accepted product read follows the stated finite route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudgetCarrier dyadic multiplication scope`

Theorem 676.16 (RegularCauchyProductBudget public export)**.** *Every accepted* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *carrier exports a public* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}^{\uparrow}}$ *consumer surface whose route is exactly the assembled finite path*

$$A, B, W_A, W_B, D_A, D_B, D, E, R, S, H, C, P, N.$$

The public surface reads the $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product row, the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal from that packet. It exports no host multiplication, quotient stream equality, ambient completeness principle, hidden rounding oracle, or selected host limit.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductBudgetUp.RegularCauchyProductBudget public export`

Proof. Start with the carrier tuple and apply Definition 676.4. This fixes the two source rows, their paired $\mathsf{StreamName}^{\uparrow}$ windows, and the dyadic source rows as the only admitted product-window data.

Next apply Definition 676.5 and Definition 676.9. These identify the $\mathsf{DyadicRatCore}^{\uparrow}$ product row and the $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger as the finite budget through which every product readback must pass.

Attach the consumer boundary through Definition 676.10 and Definition 676.11. The readback row R and seal row S are therefore downstream of the same displayed finite packet.

Finally apply Definition 676.14. The rows H, C, P, N repack transport, continuation replay, package provenance, and local naming into the public $\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}^{\uparrow}}$ surface, with no extra coordinate through which host multiplication, quotient equality, completeness, hidden rounding, or selected limits could enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, paired $\mathsf{StreamName}^{\uparrow}$ windows, dyadic source rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ product row, a $\mathsf{FiniteErrorBudget}^{\uparrow}$ ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 676.1, Definition 676.2, Definition 676.3, Definition 676.4, Definition 676.5, Definition 676.9, Definition 676.10, Definition 676.11, Definition 676.12, Definition 676.13, Definition 676.14, and Definition 676.16 bind the finite product-budget route to StreamName, DyadicRatCore, FiniteErrorBudget, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, NameCert source rows, and the public consumer surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyProductBudget}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, host real multiplication, quotient stream equality, a hidden rounding oracle, selected host limits, or a public standard bridge.

Upgrade path: The next theory-axis step is a bridge from the public product-budget export to a standard or schema-facing multiplication interface.

Chapter 677

A Concrete Naming Certificate for $\mathsf{ChoiceRecipeLedger}^{\uparrow}$

The $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ surface records the finite term-stratum ledger left when a consumer asks for a choice output but the available BEDC data contains only inspected witness recipes. It turns the witness-recipe loss of Definition 9387.34 into explicit BHist, hsame, Cont, Pkg, NameCert, and axiom-purity refusal rows.

The ledger is the choice-output analogue of the quotient-representative refusal boundary in chapter 634: both expose finite request, refusal, transport, route, provenance, and NameCert rows instead of a hidden selector.

It also consumes the effective replacement boundary of chapter 568. The finite recipe family and refusal row form the choice-output analogue of an effective replacement ledger, while the requested-output row prevents that replacement from becoming a selector.

Definition 677.1 (ChoiceRecipeLedger carrier)**.** *A* $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, F, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the term-stratum request row, recording the finite consumer site at which a choice output would be used;*
2. *F is the finite family row, listing the inspected witness recipes and the addresses at which each recipe is available;*
3. *Q is the requested-output row, recording the missing uniform choice value and the consumer route that asks for it;*
4. *A is the axiom-purity refusal row, recording that no inspected recipe supplies an unrestricted selection operator, hidden witness family, or host choice principle;*
5. *H records componentwise* hsame *transport for the request row, recipe-family row, requested-output row, refusal row, package provenance, and local naming row;*
6. *C records the displayed* Cont *routes by which a consumer reads the request, inspects the finite recipes, reaches the requested-output boundary, and returns the refusal row;*
7. *P is the* Pkg *provenance over* BHist*, term rows, finite witness recipes, requested-output rows, axiom-purity refusals,* hsame*,* Cont*, and* NameCert*;*

8. N is the local $\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}$ *row.*
The carrier exports no selected witness family, no unrestricted choice functional, no hidden recipe evaluator, no quotient of recipe addresses, and no host-level selection rule.

Theorem 677.2 (ChoiceRecipeLedger NameCert obligations)**.** *For the carrier of Definition 677.1, the local* $\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}$ *surface consists of carrier admission for*

$$L = (T, F, Q, A, H, C, P, N),$$

term-stratum request exposure through T, finite witness-recipe coverage through F, requested-output accountability through Q, axiom-purity refusal through A, componentwise hsame *stability through H, displayed* Cont *route stability through C, package provenance through P, local naming through N, and non-discharge of the requested choice output.*

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 677.1. The tuple lists the request row, finite recipe family, requested-output boundary, refusal row, transport, continuation, provenance, and local naming as the complete public inventory.

The request and coverage obligations are the rows T, F, Q. A consumer first reads the term-stratum site, then the finite family of inspected recipes, and only then the requested output boundary.

The refusal and stability obligations are A, H, C, P, N. They record axiom-purity refusal, replay the same finite route under componentwise hsame and Cont, cite the package provenance, and bind the route to the local NameCert without producing the missing choice output. □

Theorem 677.3 (ChoiceRecipeLedger non-escape)**.** *Every consumer of an accepted* $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ *packet reads only*

$$T, F, Q, A, H, C, P, N.$$

The consumer may inspect the finite recipe family and the refusal row, but it cannot extract a selected witness family, unrestricted choice functional, hidden recipe evaluator, quotient-selected address, or host choice rule from the packet.

Proof. Start from the carrier and NameCert obligations. Together they expose exactly the finite recipe ledger, requested-output boundary, refusal row, transport, continuation, provenance, and local naming data.

The axiom-purity refusal row A is downstream of the inspected recipe family F. It records the boundary between finite recipe inspection and a uniform choice output, so the refusal is not itself a selector.

Finally inspect the missing coordinates. No row names a selected witness family, choice functional, hidden evaluator, quotient of addresses, or host-level rule. The structural rows H, C, P, N preserve and package the same finite refusal route, so no escape path is available. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerNonEscape`

Theorem 677.4 (ChoiceRecipeLedger term-stratum handoff)**.** *The term-stratum handoff of an accepted* $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ *packet factors through the finite route*

$$T, F, Q, A, H, C, P, N.$$

It presents the consumer site, the inspected witness recipes, the requested-output boundary, and the axiom-purity refusal as one BHist *package with componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and a local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier term stratum handoff`
↪ variant `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerTermStratumHandoff`

Proof. Use the carrier to identify the source of the handoff. The term-stratum request row is T, and the only finite recipe-family row attached to that request is F.

The requested-output boundary and refusal are then Q and A. They state exactly what output the consumer asks for and why the inspected recipes do not provide it as a uniform selection.

The handoff is completed by H, C, P, N. These rows transport, replay, cite, and name the same finite packet, so a downstream term-stratum consumer receives a recorded refusal package rather than an unrecorded choice object. □

Theorem 677.5 (ChoiceRecipeLedger effective replacement boundary)**.** *An accepted* ChoiceRecipeLedger$^{\uparrow}$ *packet gives an effective replacement for choice-output demand only at the level of finite recipe inspection and refusal accounting. It replaces the unavailable selected witness family by the recorded* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and axiom-purity refusal rows, and it does not claim a choice output, selected family, host-level selector, or global term-stratum closure.*

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerEffectiveReplacementBoundary`

Proof. Apply the term-stratum handoff. It supplies the finite request, recipe family, requested-output boundary, refusal row, transport, continuation, provenance, and local naming data as the replacement object available to consumers.

Apply non-escape. The packet has no coordinate for a selected family, host selector, hidden evaluator, or unrestricted choice functional, so the replacement cannot be read as the missing output.

Thus the boundary is exact: the effective object is the finite ledger and its refusal package. The statement of Definition 9387.34 is realized concretely by recording what survives in the term stratum and by refusing the unrecorded choice-output jump. □

Theorem 677.6 (ChoiceRecipeLedger finite recipe coverage)**.** *Every accepted* ChoiceRecipeLedger$^{\uparrow}$ *carrier*

$$L = (T, F, Q, A, H, C, P, N)$$

exposes F as the complete inspected witness-recipe surface for the request row T. The packet has no further recipe evaluator, no hidden recipe family, no quotient-selected recipe address, and no host rule that can enlarge F into an unrestricted choice functional.

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier finite recipe coverage`

Proof. Project the carrier of Definition 677.1. Its finite family row is exactly F, and the surrounding rows identify the request T, the requested-output boundary Q, the refusal row A, transport H, continuation route C, provenance P, and local naming N.

Apply the NameCert obligations. They require the finite witness-recipe coverage row to be part of the displayed local surface, so recipe inspection is public finite data rather than an external evaluator.

Apply non-escape. Since the packet has no coordinate for a hidden evaluator, quotient-selected address, selected witness family, or host choice rule, F is the complete inspected recipe surface available to consumers of the request. □

Theorem 677.7 (ChoiceRecipeLedger requested-output accountability)**.** *Any consumer asking an accepted* ChoiceRecipeLedger$^{\uparrow}$ *packet for the missing output Q must pass through the displayed rows*

$$T, F, Q, A, H, C, P, N$$

and must read the refusal row A before treating the request as discharged. In particular, the requested-output row is accountable to the finite recipe family and never bypasses the axiom-purity refusal.

Proof. Use the term-stratum handoff. It factors every consumer route through the request row T, the finite recipe family F, the requested-output boundary Q, and the refusal row A before the transport, continuation, provenance, and naming rows package the result.

Now apply non-escape. The packet does not export a selected family, unrestricted choice functional, hidden evaluator, quotient-selected address, or host choice rule. Therefore a consumer cannot discharge Q by leaving the displayed route.

It follows that the requested-output boundary is accountable to the finite ledger: the consumer reads T and F, reaches Q, and must pass through A when no recorded recipe supplies the missing uniform output. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier requested output accountability`

Theorem 677.8 (ChoiceRecipeLedger refusal-route determinacy)**.** *For an accepted* ChoiceRecipeLedger$^{\uparrow}$ *packet, the displayed* Cont *route C determines one accepted refusal outcome for the same request row T, finite recipe family F, and requested-output row Q. Componentwise* hsame *transport through H preserves that refusal route and cannot turn it into a selector.*

Proof. Read the carrier rows. The route coordinate is the displayed Cont row C, and its inputs are the same request T, finite family F, requested-output boundary Q, and refusal row A named by the carrier.

Apply the NameCert obligations. They bind displayed route stability to the local naming surface, so replaying the route under the admitted certificate keeps the same request, recipe-family row, requested-output boundary, and refusal row.

Finally use the effective replacement boundary. The replacement object is finite recipe inspection plus refusal accounting, not a selected witness family or unrestricted choice functional. Componentwise hsame transport through H therefore preserves the recorded refusal route without creating a selector. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier refusal route determinacy`

Theorem 677.9 (ChoiceRecipeLedger obligation closure package)**.** *The obligation surface of* ChoiceRecipeLedger$^{\uparrow}$ *is exactly the carrier of Definition 677.1, the NameCert obligations of Definition 677.2, finite recipe coverage from Definition 677.6, requested-output accountability from Definition 677.7, refusal-route determinacy from Definition 677.8, non-escape from Definition 677.3, term-stratum handoff from Definition 677.4, and the effective replacement boundary from Definition 677.5. It exports no unrestricted choice functional, selected witness family, hidden evaluator, quotient-selected recipe address, or host choice rule.*

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier obligation surface exhaustion`

Proof. The carrier and NameCert obligations provide the admitted tuple and its local obligation rows. These rows expose the request, finite recipe family, requested-output boundary, refusal, transport, continuation, provenance, and naming data.

Finite recipe coverage and requested-output accountability identify how consumers inspect the packet. The only recipe surface is F, and every demand for Q must pass through the refusal row A.

Refusal-route determinacy, non-escape, term-stratum handoff, and the effective replacement boundary then close the package. They preserve the displayed route under hsame and Cont, keep the packet inside its finite ledger, and prevent the recorded replacement from becoming an unrestricted choice principle. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier obligation closure package`

Theorem 677.10 (ChoiceRecipeLedger public bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}$ *interface bridges to the concrete finite refusal-ledger carrier by the route through carrier admission, NameCert obligations, finite recipe coverage, requested-output accountability, refusal-route determinacy, non-escape, term-stratum handoff, and the effective replacement boundary. The bridge exports no unrestricted choice functional, selected witness family, quotient-selected recipe address, hidden evaluator, or host choice rule.*

Proof. Apply the carrier admission and NameCert obligations. They identify the public interface with the rows T, F, Q, A, H, C, P, N and make the finite request, recipe-family, requested-output, refusal, transport, replay, provenance, and local naming data visible.

Finite recipe coverage, requested-output accountability, refusal-route determinacy, non-escape, term-stratum handoff, and the effective replacement boundary then carry the public route to the finite refusal ledger. These cited clauses show that replay preserves the same displayed rows and cannot supply a hidden selector, selected family, quotient-selected address, evaluator, unrestricted choice functional, or host choice rule. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerPublicBridge`

Theorem 677.11 (ChoiceRecipeLedger mature finite refusal-ledger package)**.** *The finite refusal-ledger treatment of* $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ *is mature at the paper surface: carrier admission, NameCert obligations, finite recipe coverage, requested-output accountability, refusal-route determinacy, non-escape, term-stratum handoff, the effective replacement boundary, and the public bridge jointly expose the complete finite route from a request row to an accountable refusal. The package exports a public finite replacement for the displayed request surface and no unrestricted choice functional, selected witness family, quotient-selected recipe address, hidden evaluator, host selection rule, or unrecorded term-stratum premise.*

Proof. Read the accepted carrier of Definition 677.1 and the local NameCert obligations of Definition 677.2. These rows expose the request, finite recipe family, requested-output boundary, refusal row, transport, replay, provenance, and local naming data before any consumer can inspect the packet.

Apply finite recipe coverage, requested-output accountability, refusal-route determinacy, and non-escape from Definition 677.6, Definition 677.7, Definition 677.8, and Definition 677.3. These clauses keep every requested output inside the displayed finite ledger and force replay through the recorded refusal route.

Finally use the term-stratum handoff, effective replacement boundary, obligation closure package, and public bridge from Definition 677.4, Definition 677.5, Definition 677.9, and Definition 677.10. They assemble the public finite refusal-ledger route while preserving the exclusions of unrestricted choice, selected witness families, quotient-selected addresses, hidden evaluators, host selection rules, and unrecorded premises. □

Lean4 ✓ `BEDC.Derived.ChoiceRecipeLedgerUp.ChoiceRecipeLedgerCarrier mature package`

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ packet is generated from a term-stratum request row, a finite witness-recipe family, a requested-output boundary, an axiom-purity refusal row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 677.1, Definition 677.2, Definition 677.6, Definition 677.7, Definition 677.8, Definition 677.9, Definition 677.3, Definition 677.4, Definition 677.5, Definition 677.10, and Definition 677.11 expose the mature public finite refusal-ledger bridge over carrier, obligation, coverage, accountability, determinacy, non-escape, handoff, replacement, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceRecipeLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, unrestricted choice, selected witness families, quotient-selected recipe addresses, host-level selection rules, global term-stratum closure, or a public standard bridge.

Upgrade path: Mature closure requires the checked standard bridge and Lean-side verification of the public bridge route.

Chapter 678

A Concrete Naming Certificate for $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$

The $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ packet records the finite selector-budget step used by the Real completeness diagonal construction. It binds a displayed regular Cauchy modulus, the selected finite window budget, StreamName observation rows, a RegSeqRat witness row, the RealUp seal handoff, and the BEDC transport rows as one inspectable naming certificate. It is a local finite packet between modulus selection and seal delivery; it is not a host limit, quotient stream equality, countable choice principle, or ambient completeness theorem.

Definition 678.1 (RegularCauchySelectorBudget carrier)**.** *A* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (\mu, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *μ is the selector-modulus row read from the regular Cauchy input at the requested precision;*
2. *W is the finite window-budget row naming the selected indices and StreamName observation extent allowed by μ;*
3. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *observation rows opened by the selected window budget;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row assembled from those observations for the diagonal handoff;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row through which the completion-facing consumer reads the selected regular witness;*
6. *H records componentwise* hsame *transports for the modulus, window budget, StreamName rows, RegSeqRat witness, RealUp seal, and local structural rows;*
7. *C records the displayed* Cont *routes for modulus inspection, window coverage, observation readback, RegSeqRat handoff, RealUp seal delivery, and non-escape inspection;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $\mu, W, S, R, E, H, C, P, N$.

The carrier consumes the finite-window service of chapter 126 through S and hands the selected-window budget W to the Cauchy modulus refinement route in chapter 885, where the same budget is read before the dyadic, rational, and Real-seal rows.

Theorem 678.2 (RegularCauchySelectorBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudg}}$ *surface for Definition 678.1 consists of carrier admission, selector-modulus exactness, finite window-budget coverage, StreamName observation readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from host limits, quotient streams, countable choice, hidden window enlargement, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySelectorBudgetUp.RegularCauchySelectorBudgetCarrier namecert obligations`

Proof. Carrier admission is the tuple in Definition 678.1. It displays the selector modulus, finite window budget, StreamName rows, RegSeqRat witness, RealUp seal, transports, routes, provenance, and local naming row.

Selector-modulus exactness and window-budget coverage are read from μ and W. The StreamName observations S and the RegSeqRat witness R are listed after those rows, so the witness route cannot bypass the selected finite budget.

The RealUp seal boundary is the role of E, while the structural obligations are the roles of H, C, P, N. Since the carrier has no coordinate for a host limit, quotient stream equality, countable choice, hidden window enlargement, or ambient completeness, non-escape follows by inspection of the displayed packet. □

Theorem 678.3 (RegularCauchySelectorBudget window coverage)**.** *Every accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier covers its diagonal observation request through the displayed finite window budget* W*. For*

$$Q = (\mu, W, S, R, E, H, C, P, N),$$

each StreamName row in S *is opened only inside the indices and observation extent named by* W*, and the budget* W *is itself read from the selector modulus row* μ*.*

Proof. Project the rows μ and W from the carrier. The window budget is an explicit coordinate whose role is to name the finite indices and observation extent admitted by the selected modulus route.

The StreamName rows S are listed after W and are consumed by the same continuation data. Thus an accepted observation row is available only as part of the selected finite budget.

The remaining rows R, E, H, C, P, N consume the displayed observations. No row permits an unbounded stream scan or a second selector, so window coverage is confined to W. □

Theorem 678.4 (RegularCauchySelectorBudget RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *witness exported by an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier factors through the finite route*

$$\mu, W, S, R, H, C, P, N.$$

The handoff preserves the selected modulus, finite window budget, StreamName observation rows, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RegularCauchySelectorBudgetUp.RegularCauchySelectorBudgetCarrier regseqrat handoff`

Proof. Use Definition 678.3 to identify the selected modulus and finite window budget. This fixes μ, W, S as the only observation data available to the witness route.

The witness row R is listed after the observations and is routed through the same structural rows. Therefore every exported $\mathsf{RegSeqRat}^{\uparrow}$ witness consumes the displayed StreamName packet before it can be read by a downstream consumer.

Inspect H, C, P, N. These rows keep the handoff stable under hsame transport, continuation replay, package provenance, and local naming, so the RegSeqRat handoff is exactly the displayed finite route. □

Theorem 678.5 (RegularCauchySelectorBudget Real seal boundary)**.** *Every Real completeness consumer of an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the seal row E after the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row R has been assembled from the selected StreamName window budget. The carrier exports no host-selected limit, quotient stream equality, countable choice selector, hidden observation budget, or ambient completeness witness.*

Proof. Project the seal row E from the carrier. It appears after the RegSeqRat witness R, so a real-facing route reaches $\mathsf{Real}^{\uparrow}$ only after the finite witness handoff has been assembled.

Apply Definition 678.4. The witness R is already bound to the modulus, window budget, StreamName observations, transports, routes, provenance, and local naming row.

The carrier contains no coordinate for a host-selected limit, quotient stream equality, countable choice selector, hidden observation budget, or ambient completeness. Hence the RealUp seal is a boundary around the displayed finite packet and exports no additional real object. □

Theorem 678.6 (RegularCauchySelectorBudget non-escape)**.** *An accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple $\mu, W, S, R, E, H, C, P, N$. In particular, it does not produce an unbounded family of StreamName observations, a second modulus selector, a quotient of regular sequences, a host real limit, or a completeness principle for* $\mathsf{Real}^{\uparrow}$.

Proof. Read the carrier coordinates from Definition 678.1. Every exported route is one of the displayed finite rows, with H, C, P, N providing only transport, continuation, provenance, and naming structure for those rows.

The window coverage theorem confines StreamName observations to W, and the RegSeqRat handoff confines the witness to μ, W, S, R, H, C, P, N. The Real seal boundary then confines completion-facing reads to E after that finite witness route.

Since no additional coordinate is present, the packet has no source for unbounded observations, a second selector, quotient regular sequences, host real limits, or a RealUp completeness principle. □

Theorem 678.7 (RegularCauchySelectorBudget obligation closure)**.** *The accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet*

$$Q = (\mu, W, S, R, E, H, C, P, N)$$

closes the local selector-budget obligation surface by packaging the carrier admission of Definition 678.1, the finite window coverage of Definition 678.3, the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of Definition 678.4, the* $\mathsf{Real}^{\uparrow}$ *seal boundary of Definition 678.5, and the non-escape clause of Definition 678.6 as one finite* BHist *certificate.*

Proof. Carrier admission supplies the displayed tuple Q. The selector-modulus row μ and finite window row W determine which StreamName observations S are available.

Apply Definition 678.3. It confines the observation rows to the selected finite budget and rules out hidden enlargement of the StreamName read.

Apply Definition 678.4 and Definition 678.5. These rows force the RegSeqRat witness and RealUp seal to be downstream of the same finite packet.

Finally use Definition 678.6. Since no coordinate outside $\mu, W, S, R, E, H, C, P, N$ is present, the listed rows exhaust the local selector-budget obligations. □

Theorem 678.8 (RegularCauchySelectorBudget formal target surface)**.** *The Lean-facing target surface for* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *is the finite* BHist *carrier* $Q = (\mu, W, S, R, E, H, C, P, N)$ *together with componentwise* hsame *rows in* H*, displayed* Cont *routes in* C*,* Pkg *provenance in* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}$ *row* N*. A formalization of this surface must expose exactly the selector-modulus row, finite window budget, StreamName observation rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness,* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, route rows, provenance row, and naming row, with no host limit or quotient stream coordinate.*

Proof. Read the rows of Definition 678.1. The first five coordinates carry the selector modulus, finite window budget, StreamName observations, RegSeqRat witness, and RealUp seal.

The structural coordinates are also displayed. The row H supplies componentwise hsame stability, C supplies continuation replay, P supplies package provenance, and N supplies the local naming certificate.

Thus a Lean target can be stated over the same finite BHist packet. Because the carrier has no coordinate for host limits, quotient streams, or ambient completeness, those objects cannot appear in the formal target surface. □

Theorem 678.9 (RegularCauchySelectorBudget verification handoff)**.** *Downstream Real and Completion formalization consumes a* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet only through the finite route*

$$\mu, W, S, R, E, H, C, P, N.$$

The consumer first reads the selector modulus and finite window budget, then reads the StreamName observations, then assembles the $\mathsf{RegSeqRat}^{\uparrow}$ *witness, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal. The handoff supplies no countable choice selector, quotient of regular sequences, hidden observation budget, host-selected real limit, or ambient RealUp completeness theorem.*

Proof. Start with the obligation closure theorem. It packages carrier admission, window coverage, RegSeqRat handoff, RealUp seal boundary, and non-escape into the single finite packet Q.

The formal target surface theorem names the rows that Lean must expose. These are exactly the rows consumed by the downstream Real and Completion route.

Reading the rows in order gives the handoff discipline: μ and W select the finite observation budget, S supplies the observations, R supplies the regular-rational witness, and E supplies the seal. The non-escape theorem removes countable choice, quotient regular sequences, hidden budgets, host real limits, and ambient completeness from the handoff. □

Theorem 678.10 (RegularCauchySelectorBudget consumer exactness)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier*

$$Q = (\mu, W, S, R, E, H, C, P, N)$$

reads exactly the displayed finite route $\mu, W, S, R, E, H, C, P, N$*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer surface stops at the witness row* R *together with the selector modulus, finite window budget, StreamName*

rows, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. The* $\mathsf{Real}^{\uparrow}$ *consumer surface reaches the seal row E only after the same* $\mathsf{RegSeqRat}^{\uparrow}$ *witness route has been read, and it receives no additional host limit, quotient stream equality, countable choice selector, hidden window budget, or ambient completeness witness.*

Proof. Start with the carrier definition. The tuple Q displays exactly the selector modulus, finite window budget, StreamName observation rows, RegSeqRat witness, RealUp seal, transports, routes, provenance, and local naming row.

Apply Definition 678.3 and Definition 678.4. These identify the $\mathsf{RegSeqRat}^{\uparrow}$ consumer route with μ, W, S, R, H, C, P, N and force the witness row to consume only the selected finite window.

Next apply Definition 678.5. The $\mathsf{Real}^{\uparrow}$ consumer reaches E only after the RegSeqRat witness has been assembled through that same finite route.

Finally use Definition 678.6, Definition 678.7, and Definition 678.9. These rows exhaust the packet surface and exclude every route outside $\mu, W, S, R, E, H, C, P, N$. □

Theorem 678.11 (RegularCauchySelectorBudget diagonal use)**.** *Every diagonal Real-completion consumer of an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet*

$$Q = (\mu, W, S, R, E, H, C, P, N)$$

must first read the selector modulus μ, finite window budget W, $\mathsf{StreamName}^{\uparrow}$ *observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness R, and* $\mathsf{Real}^{\uparrow}$ *seal E through the displayed H, C, P, N rows. No diagonal consumer may substitute a host limit, quotient-stream equality, countable choice selector, hidden window enlargement, or ambient completeness witness for those finite rows.*

Proof. Begin with Definition 678.3. It binds every observation request to the selected finite window budget W read from the selector modulus μ.

Apply Definition 678.4. The $\mathsf{RegSeqRat}^{\uparrow}$ witness R is assembled only after the StreamName observations S have been read through the same transport, route, provenance, and naming rows.

Next use Definition 678.5. The $\mathsf{Real}^{\uparrow}$ seal E is available only after that finite RegSeqRat route has been displayed.

Finally apply Definition 678.6 and Definition 678.10. These results exhaust the diagonal consumer surface at $\mu, W, S, R, E, H, C, P, N$ and exclude host limits, quotient-stream equality, countable choice selectors, hidden window enlargement, and ambient completeness witnesses. □

Theorem 678.12 (RegularCauchySelectorBudget tail selector compatibility)**.** *Every tail-selector read through an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier is compatible with the same selector modulus μ and finite window budget W. A tail row may inspect only StreamName observations inside S, may assemble its regular rational witness only through R, and may reach any Real-facing seal only through E after the route μ, W, S, R, H, C, P, N has been displayed.*

Proof. Read the selector rows μ and W from Definition 678.1. The window coverage theorem makes W the selected finite observation budget determined by μ.

A tail-selector read is therefore a consumer of the displayed StreamName rows S. Apply Definition 678.4 to force the corresponding regular rational witness through R and the structural rows H, C, P, N.

For a Real-facing tail read, apply Definition 678.5. The seal E is available only after the same finite RegSeqRat witness route has been consumed.

The non-escape and verification handoff theorems rule out a second selector modulus, a larger window, or an independent tail budget. Thus every tail-selector read remains compatible with μ and W. □

Theorem 678.13 (RegularCauchySelectorBudget Real completeness handoff)**.** *A RealCompleteness handoff that consumes an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet may use only the selector modulus* μ*, the finite window budget* W*, the StreamName rows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}$ *row* N*. It does not receive a quotient stream, a host-selected real limit, countable choice, hidden window enlargement, or an ambient RealUp completeness theorem from the packet.*

Proof. Project the carrier $Q = (\mu, W, S, R, E, H, C, P, N)$. These are the only rows available to a RealCompleteness handoff from the selector-budget packet.

Use Definition 678.3 to bind S to the finite window budget W selected by μ. Use Definition 678.4 to bind R to those StreamName observations and to the structural rows.

Use Definition 678.5 to make E the only Real-facing seal and to place it after the RegSeqRat witness route. The rows H, C, P, N then provide only transport, continuation replay, provenance, and local naming for that same packet.

Finally apply Definition 678.6, Definition 678.7, and Definition 678.9. The handoff surface is exhausted by the displayed packet, so RealCompleteness receives no additional completion principle or hidden real object from this chapter. □

Definition 678.14 (RegularCauchySelectorBudget tail-precision source)**.** *The tail-precision source of an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier is the finite row package*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N)$$

where μ *is the selector-modulus row,* W *is the selected finite window budget,* S *is the StreamName observation family,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row requested by the tail consumer,* E *is the* $\mathsf{Real}^{\uparrow}$ *seal,* H *records componentwise* hsame *transports,* C *records displayed* Cont *routes,* P *records* Pkg *provenance, and* N *is the local* NameCert *source.*

Theorem 678.15 (RegularCauchySelectorBudget tail-precision exactness)**.** *For an accepted* RegularCauchySelector *packet with tail-precision source*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N),$$

the dyadic tolerance row D *determines the only permitted finite window budget through* μ, W*. Every StreamName observation read is inside* S*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *witness read factors through* R*, and every* $\mathsf{Real}^{\uparrow}$ *consumer reaches only* E *after the route*

$$D \longrightarrow \mu \longrightarrow W \longrightarrow S \longrightarrow R \longrightarrow E$$

with transport, replay, provenance, and local naming supplied by H, C, P, N*. No accepted route exports a quotient stream, host real equality, chosen limit, infinite tail, hidden window enlargement, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySelectorBudgetUp.RegularCauchySelectorBudgetCarrier tail precision exactness`

Proof. Project the finite tail-precision tuple. The tolerance row D, selector-modulus row μ, and finite window row W are explicit coordinates of Definition 678.14, so a precision read has no source for a second tolerance-to-window route.

Apply Definition 678.3 and Definition 678.16. These bind the selected StreamName observations to the finite budget chosen through μ, W and force the tail consumer to pass through $D, \mu, W, S, R, E, H, C, P, N$.

Next use Definition 678.4. The $\mathsf{RegSeqRat}^{\uparrow}$ witness row is downstream of the displayed StreamName observations, so every regular-rational witness factors through R together with the same structural rows.

Finally apply Definition 678.17 and Definition 678.26. The Real-facing read reaches only the seal row E after the finite route has been displayed, and the excluded quotient, host equality, chosen-limit, infinite-tail, hidden-budget, and ambient-completeness data are not coordinates of the packet. □

Theorem 678.16 (RegularCauchySelectorBudget tail budget totality)**.** *Every accepted* RegularCauchySelectorBudg *packet exposes the requested tail precision and its selected finite window through the rows of Definition 678.14. A tail consumer may read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the selector modulus μ, and the finite budget W, and may assemble the corresponding StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *evidence only through S, R, H, C, P, N before reaching the* $\mathsf{Real}^{\uparrow}$ *seal E.*

Proof. Project the accepted carrier and its tail-precision source. The rows μ, W, D display the requested precision interface: D names the tolerance row, μ names the selector modulus, and W names the finite window selected for that tolerance.

Use Definition 678.3. It confines every StreamName observation in S to the finite window budget W selected by μ.

Use Definition 678.4 and Definition 678.13. The witness row R and seal row E are downstream of the same finite route, while H, C, P, N provide transport, continuation, provenance, and local naming for the displayed precision source. □

Theorem 678.17 (RegularCauchySelectorBudget precision-window non-escape)**.** *No* $\mathsf{Real}^{\uparrow}$ *consumer reading Definition 678.14 can recover quotient streams, host equality of regular sequences, chosen limits, infinite tails, or ambient completeness. The packet exposes only the tolerance row D, selector modulus μ, finite window budget W, StreamName observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness R,* $\mathsf{Real}^{\uparrow}$ *seal E, and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with Definition 678.14. The listed rows are the whole tail-precision source, and none is a quotient stream, host equality proof, chosen limit, infinite tail, or ambient completeness object.

Apply Definition 678.16. It makes every precision read pass through $D, \mu, W, S, R, E, H, C, P, N$ and keeps the selected observation window finite.

Finally apply Definition 678.6 and Definition 678.13. These boundaries already exclude quotient streams, host-selected limits, hidden window enlargement, and completion principles from the selector-budget packet, so the tail-precision source exports no stronger RealUp data. □

Theorem 678.18 (RegularCauchySelectorBudget tail-meet factorization)**.** *Let Q be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet with tail-precision source*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N).$$

Let T be an accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet from chapter 560 whose shared-tail presentation uses the same displayed tolerance row D, finite window rows W, regular witness row R, componentwise transports H, displayed routes C, provenance P, and local naming rows. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *consumer of the selector-budget tail source factors through the TailMeet shared-threshold row of T before reaching the* $\mathsf{Real}^{\uparrow}$ *seal E.*

The factorization exports no quotient stream, host real equality, chosen limit, infinite tail, ambient completeness row, or hidden threshold oracle.

Proof. Start with the tail-precision source in Definition 678.14. Its only selector-budget precision coordinates are $D, \mu, W, S, R, E, H, C, P, N$.

Apply Definition 678.16. It makes every admitted tail read pass through the displayed tolerance row, selector modulus, finite window budget, StreamName observations, regular witness, transports, routes, provenance, and local name before any seal read is available.

Use Definition 560.34 for the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ packet T. Since the tolerance, window, witness, transport, route, provenance, and naming rows are shared with $\mathsf{TailPrec}(Q)$, the TailMeet consumer route supplies the shared-threshold row as the common finite threshold packet for the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ consumers.

Now apply Definition 678.17 together with Definition 560.50. The selector-budget side cannot bypass $D, \mu, W, S, R, E, H, C, P, N$, and the TailMeet side cannot turn the shared threshold into host equality, a selected limit, an infinite tail, ambient completeness, or a hidden oracle.

Thus every consumer route first passes through the TailMeet shared-threshold row and only then reaches the selector-budget real seal E. The resulting route is a finite factorization through the displayed packet rows, with none of the excluded quotient, host, selection, tail, completeness, or oracle data exported. □

Theorem 678.19 (RegularCauchySelectorBudget tail-meet selector exhaustion)**.** *Let Q be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet with tail-precision source*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N).$$

Any tail-meet selector route through Q is exhausted by the displayed rows

$$D, \mu, W, S, R, E, H, C, P, N$$

together with the imported $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold before a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer can read the budget packet. The route cannot add quotient streams, host equality, chosen limits, infinite tails, hidden budget, or ambient completeness.*

Proof. Start with the selector-budget packet. By Definition 678.16, every admitted tail read uses the displayed tolerance row, selector modulus, finite window budget, StreamName observations, regular witness, Real seal, transports, routes, provenance, and local name.

Next insert the shared threshold. The tail-meet factorization of Definition 678.18 routes every Real or tail-fusion consumer through the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold before the seal row E is available.

The precision-window boundary then removes alternate sources. Apply Definition 678.17; it excludes quotient streams, host equality, chosen limits, infinite tails, and ambient completeness from the selector-budget tail source.

Finally apply the Real-completeness handoff of Definition 678.13. The budget packet reaches the completion-facing consumer only through the same displayed rows and the imported shared threshold, so no hidden budget or ambient completeness route remains. □

Theorem 678.20 (RegularCauchySelectorBudget tail-meet threshold coverage)**.** *Let Q be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet with tail-precision source*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N).$$

Let T be an accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose shared-threshold route transports the same tolerance row D, finite window budget W, StreamName observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness R,* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transports H, displayed routes C, provenance P, and local naming rows. Then every Real-facing read of Q is covered by the TailMeet shared-threshold row before E is read.*

Proof. First project the tail-precision source. The displayed selector-budget rows are exactly $\mu, W, S, R, D, E, H, C, P, N$, so a Real-facing read has no source outside this finite packet.

Use Definition 678.16 and Definition 678.17. They force the tolerance row, selector modulus, finite window budget, observations, regular witness, seal, transports, routes, provenance, and name to remain visible before the seal can be consumed.

Now apply Definition 678.18. The transported tolerance, window, witness, structural, provenance, and naming rows are the rows consumed by T.

Finally use Definition 560.34 and Definition 560.50. The TailMeet packet supplies the shared threshold as the finite ledger reached before any Real-facing read, and it exports no hidden threshold oracle or ambient completeness data. □

Theorem 678.21 (RegularCauchySelectorBudget diagonal consumer non-escape)**.** *Every accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet consumed by a diagonal reader or by a* $\mathsf{Real}^{\uparrow}$ *seal reader factors through the finite carrier route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The factorization uses only the displayed BHist *packet, componentwise* hsame *rows, finite* Cont *routes,* Pkg *provenance, and local* NameCert *rows of the selector-budget carrier. It exports no quotient stream, selected limit, trichotomy row, hidden window enlargement, countable-choice selector, host real equality, or ambient completeness witness.*

Proof. First project the accepted selector-budget tail source. The displayed rows $\mu, W, S, R, D, E, H, C, P, N$ place the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D and finite window budget W before the StreamName observation row S, the $\mathsf{RegSeqRat}^{\uparrow}$ witness row R, and the $\mathsf{Real}^{\uparrow}$ seal row E.

Apply Definition 678.20. It forces each Real-facing tail read to pass through the TailMeet shared-threshold row before E is read, so the diagonal consumer cannot bypass the finite dyadic threshold route.

Next use Definition 678.4, Definition 678.5, and Definition 678.26. These rows place the RegSeqRat handoff after the StreamName observations and place the Real seal after the same handoff, all under H, C, P, N.

Finally compare the sibling pullback surfaces. The StreamName surface supplies only the displayed observation rows, the RegSeqRat surface consumes those observations as the finite regular-rational witness, and the Real surface receives only the seal row. Exhausting those coordinates rules out quotient streams, selected limits, trichotomy, host equality, hidden budget, and ambient completeness as consumer outputs. □

Theorem 678.22 (RegularCauchySelectorBudget tail-meet route determinacy)**.** *Two selector-budget reads using the same accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet and the same transported D, W, S, R, H, C, P, N rows reach the same shared dyadic threshold ledger. The determinacy is internal to the selector-budget tail source, the TailMeet shared-threshold route, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with the two selector-budget reads and project their tail-precision sources. The hypothesis identifies the tolerance row D, finite window budget W, StreamName observations S, regular witness R, transports H, routes C, provenance P, and naming rows N componentwise.

Apply Definition 678.20. Both reads must pass through the TailMeet shared threshold before any Real-facing seal read can occur.

Use Definition 560.56. The transported source, window, and modulus data in the TailMeet route determine one displayed dyadic threshold ledger.

The selector-budget factorization theorem then repacks that ledger for both reads. Since the structural rows are the same transported H, C, P, N rows, the two selector-budget reads reach the same shared threshold entry. □

Theorem 678.23 (RegularCauchySelectorBudget tail-meet non-escape)**.** *The composite* $\mathsf{RegularCauchySelectorBud}$ *and* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *route exports no quotient stream, host real equality, chosen limit, infinite tail, ambient completeness principle, or hidden threshold oracle. Its public data are only the selector-budget tail source rows* $\mu, W, S, R, D, E, H, C, P, N$ *and the TailMeet shared-threshold rows reached through the same displayed transports, routes, provenance, and local names.*

Proof. Begin on the selector-budget side. The precision-window non-escape theorem Definition 678.17 excludes quotient streams, host equality of regular sequences, chosen limits, infinite tails, and ambient completeness from $\mathsf{TailPrec}(Q)$.

Move to the shared-threshold side through Definition 678.18. The factorization routes admitted consumers into the TailMeet shared threshold rather than into an external completion or selection object.

Finally apply Definition 560.50 and Definition 560.56. The TailMeet ledger has no hidden threshold oracle, and duplicate transported reads reach the same displayed dyadic ledger. Therefore the composite route exports only the named finite rows and none of the excluded quotient, host, selection, tail, completeness, or oracle data. □

Theorem 678.24 (RegularCauchySelectorBudget selector-budget exactness)**.** *Let* $Q = (\mu, W, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier. Every* $\mathsf{StreamName}^{\uparrow}$ *observation row in* S *and every* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row read through* R *is admitted only through the selected pair* (μ, W)*:* μ *supplies the selector-modulus row,* W *supplies the finite observation budget selected by that row, and no accepted consumer may substitute a second modulus, a larger window, or an unlisted observation family.*

Proof. Start with the carrier definition. The tuple displays μ before W, then lists the StreamName observations S and the RegSeqRat witness R downstream of those selector rows.

Apply Definition 678.3. It binds every StreamName observation to the finite indices and observation extent named by W, with W itself read from μ.

Apply Definition 678.4. The witness row R consumes the same displayed StreamName packet through H, C, P, N, so the RegSeqRat route has no source outside the selected pair.

Finally use Definition 678.18 and Definition 678.23. Tail-facing reads factor through the same finite rows and export no hidden threshold oracle, second selector, or enlarged observation window. □

Theorem 678.25 (RegularCauchySelectorBudget finite-window provenance)**.** *For every accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier with tail-precision source*

$$\mathsf{TailPrec}(Q) = (\mu, W, S, R, D, E, H, C, P, N),$$

the provenance row P must name exactly the dependencies consumed by the finite selector route: $\mathsf{DyadicRatCore}^{\uparrow}$ *through D,* $\mathsf{StreamName}^{\uparrow}$ *through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *through R,* $\mathsf{Real}^{\uparrow}$ *through E, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows through H, C, P, N.*

Proof. Project the tail-precision source. The row D supplies the dyadic tolerance, while $\mu, W, S, R, E, H, C, P, N$ supply the selected modulus, finite window, observations, witness, seal, transport, continuation, provenance, and local naming rows.

Use Definition 678.16. It forces every tail read to pass through $D, \mu, W, S, R, E, H, C, P, N$ before the Real seal can be reached.

Use Definition 678.24 and Definition 678.4. These identify the StreamName and RegSeqRat components as downstream of the selected finite window route.

Finally apply Definition 678.5, Definition 678.18, and Definition 678.23. The Real seal and TailMeet rows consume the same finite provenance surface, so P has no additional dependency to name. □

Theorem 678.26 (RegularCauchySelectorBudget Real seal non-escape package)**.** *A Real-facing consumer of an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet may read only the displayed rows $\mu, W, S, R, E, H, C, P, N$. It receives the Real seal boundary through E and the structural route through H, C, P, N, but it receives no quotient stream, host limit, countable choice selector, hidden window enlargement, ambient completeness principle, or unlisted TailMeet threshold.*

Proof. Read the carrier rows from Definition 678.1. The only Real-facing coordinate is the seal row E, and it is packaged with the selected finite rows μ, W, S, R, H, C, P, N.

Apply Definition 678.5. The consumer reaches $\mathsf{Real}^{\uparrow}$ only after the RegSeqRat witness has been assembled from the selected StreamName budget.

Apply Definition 678.6 and Definition 678.24. These rows exclude quotient streams, host limits, countable choice, second selectors, and hidden window enlargement from the packet surface.

Finally use Definition 678.18 and Definition 678.23. TailMeet interaction remains a finite factorization over displayed rows and contributes no ambient completeness principle or unlisted threshold object. □

Theorem 678.27 (RegularCauchySelectorBudget CauchyInverseBudget apartness handoff)**.** *When a* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *tail source also carries the apartness row needed for reciprocal Real reads, every inverse-facing consumer factors through the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier of chapter 703. The consumer first reads the selector modulus, finite window budget,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness, and* $\mathsf{Real}^{\uparrow}$ *seal rows of the selector-budget packet, then hands the apartness, modulus, finite-window, dyadic, readback, and seal rows to the reciprocal budget. It receives no quotient stream, host equality, chosen limit, countable choice selector, total reciprocal operation, trichotomy, or ambient completeness principle.*

Proof. Begin with the selector-budget tail source. By Definition 678.16, any tail read consumes the dyadic tolerance, selector modulus, finite window budget, StreamName observations, RegSeqRat witness, Real seal, transports, routes, provenance, and local naming rows before reaching the Real-facing boundary.

Apply Definition 678.26. The selector-budget side exports only the displayed rows $\mu, W, S, R, E, H, C, P, N$ to a Real-facing consumer and excludes quotient streams, host limits, countable choice, hidden window enlargement, and ambient completeness.

Now pass to the reciprocal budget through Definition 703.13. The additional apartness row is read as the A coordinate, the selected modulus and window are read as M, W, and the dyadic and regular rows are read as D, R inside the $\mathsf{CauchyInverseBudget}^{\uparrow}$ carrier.

Finally apply Definition 703.14 and Definition 703.8. They route the inverse-facing RegSeqRat and Real reads through $X, A, M, W, D, R, S, H, C, P, N$, so no total reciprocal operation, trichotomy row, host equality, quotient stream, chosen limit, countable choice selector, or ambient completeness principle is exported. □

Theorem 678.28 (RegularCauchySelectorBudget terminal-window lock)**.** *Let $Q = (\mu, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *carrier. Any terminal Phase Real or Completion consumer that uses the selected budget must read the same finite window W,* $\mathsf{StreamName}^{\uparrow}$ *observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness R, and* $\mathsf{Real}^{\uparrow}$ *seal E through the displayed transports, continuation routes, provenance, and local naming rows H, C, P, N. It cannot substitute a quotient stream, host limit, countable-choice selector, hidden enlarged window, alternate terminal window, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the selector-budget exactness theorem Definition 678.24. It binds every StreamName observation and RegSeqRat witness read to the selected pair (μ, W), excluding a second modulus, larger window, or unlisted observation family.

Use finite-window provenance from Definition 678.25. The provenance row names exactly the StreamName, RegSeqRat, Real seal, transport, continuation, package, and NameCert dependencies consumed by the finite selector route.

Apply Definition 678.26. A terminal Real-facing read can reach only E after the selected finite rows have been displayed through H, C, P, N, so quotient streams, host limits, choice selectors, enlarged windows, and ambient completeness rows are outside the packet.

Finally combine the open-phase route of Definition 126.434. That route sends the StreamName window, RegSeqRat witness, and Real seal through the same open-phase seal path, so a terminal Phase Real or Completion consumer has no alternate terminal window to read. □

Theorem 678.29 (RegularCauchySelectorBudget finite-chain induction)**.** *Let $Q = (\mu, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *packet. Any terminal* $\mathsf{Real}^{\uparrow}$ *or Completion consumer route exhausted by Q is determined by induction over the displayed finite chain*

$$\mu \longrightarrow W \longrightarrow S \longrightarrow R \longrightarrow E \longrightarrow H, C, P, N.$$

The base step reads only the selector modulus μ and the finite window budget W; the observation step reads only the $\mathsf{StreamName}^{\uparrow}$ *rows selected by W; the witness step assembles only the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R; the seal step reaches only the* $\mathsf{Real}^{\uparrow}$ *row E; and the structural step uses H, C, P, N only for transport, continuation, provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}$ *row. Consequently the terminal route exports no quotient stream, host limit, externally selected limit, countable-choice selector, hidden enlarged window, alternate terminal window, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. For the base step, apply Definition 678.16 and Definition 678.24. These identify μ as the selector-modulus row and W as the finite window selected by it, with no second modulus or enlarged window available to the route.

For the observation step, use Definition 678.3. Every StreamName observation consumed by a downstream reader lies in the window named by W, so the induction passes from μ, W to only the displayed rows S.

For the witness step, apply Definition 678.4. The RegSeqRat evidence is assembled through R from the selected StreamName rows and the displayed structural rows, not from a quotient stream, host limit, or countable-choice selection.

For the seal step, combine Definition 678.5 with Definition 678.26. The Real-facing route reaches E only after the finite StreamName and RegSeqRat rows have been displayed, and it receives no externally selected limit or ambient completeness principle.

For the structural step, use Definition 678.25 to read H, C, P, N as the transport, continuation, provenance, and local NameCert rows of the same finite chain. Finally apply Definition 678.28. It rules out alternate terminal windows, so the induction over $\mu, W, S, R, E, H, C, P, N$ exhausts the terminal Phase Real or Completion route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ packet is generated from a selector modulus, finite window budget, StreamName observation rows, a RegSeqRat witness, a RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 678.1, Definition 678.2, Definition 678.3, Definition 678.4, Definition 678.5, Definition 678.6, Definition 678.7, Definition 678.8, Definition 678.9, Definition 678.10, Definition 678.12, Definition 678.13, Definition 678.14, Definition 678.15, Definition 678.16, Definition 678.17, Definition 678.18, Definition 678.19, Definition 678.20, Definition 678.22, Definition 678.23, Definition 678.24, Definition 678.25, Definition 678.26, Definition 678.28, and Definition 678.29 bind the finite selector-budget carrier, tail-meet selector exhaustion, terminal-window lock, finite-chain terminal induction, and local NameCert obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySelectorBudget}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchySelectorBudgetUp.RegularCauchySelectorBudgetCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host real limits, countable choice, hidden window enlargement, or a public standard bridge.

Upgrade path: No higher paper closure grade remains for this finite selector-budget bridge surface; Lean verification and bridge checking remain independent formal-axis work.

Chapter 679

A Concrete Naming Certificate for $\mathsf{CompilerCompositionTrace}^{\uparrow}$

The $\mathsf{CompilerCompositionTrace}^{\uparrow}$ interface packages sequential compiler composition as a finite trace over adjacent NameCert compiler graphs. It does not assert a host-level compiler category. It records the two displayed compiler packets, their shared middle certificate, the relational composite graph, classifier transport, continuation replay, associativity comparison, package provenance, and the local naming row.

Definition 679.1 (CompilerCompositionTrace carrier)**.** *A* $\mathsf{CompilerCompositionTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, M, T, K_{SM}, K_{MT}, J, G, L, H, C, A, P, N)$$

with the following displayed rows:

1. *S, M, and T are the source, middle, and target* NameCert *rows consumed by the two adjacent compiler graphs;*
2. *K_{SM} is the accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet from S to M;*
3. *K_{MT} is the accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet from M to T;*
4. *J is the middle-certificate compatibility row recording that the target endpoint of K_{SM} and the source endpoint of K_{MT} are the same displayed* NameCert *row M;*
5. *G is the composite compiler graph row from S to T, obtained by finite relational composition through a displayed middle history of M;*
6. *L is the classifier-transport ledger chaining source classifier membership through K_{SM}, the middle row J, and K_{MT} into target classifier membership;*
7. *H records componentwise* hsame *transport for the endpoint certificates, adjacent compiler packets, middle compatibility row, composite graph, classifier ledger, continuation routes, associativity row, package row, and local naming row;*
8. *C records displayed* Cont *replay for the first graph edge, middle handoff, second graph edge, composite edge, target readback, and downstream NameCert exposure;*
9. *A is the associativity comparison row stating that three adjacent accepted compiler packets have the same exported finite packet after either parenthesization of relational graph composition;*

10. *P names the dependency surface* BHist, hsame, Cont, Pkg, NameCert, NameCertMorphism$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, ClassifierMorphism$^{\uparrow}$, KernelMorphismUp, *and* CompilerOptimizationEndomorphism$^{\uparrow}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host compiler function, ambient program equality, denotational semantics, optimizer correctness outside the displayed rows, decompiler data, quotient equality, or a global categorical equality not witnessed by A.

Theorem 679.2 (CompilerCompositionTrace NameCert obligations)**.** *For the carrier of Definition 679.1, the local* $\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (S, M, T, K_{SM}, K_{MT}, J, G, L, H, C, A, P, N),$$

endpoint certificate admission, acceptance of the two adjacent CertificateCompiler$^{\uparrow}$ *packets, middle-certificate compatibility, composite graph formation, classifier transport, componentwise* hsame *transport, displayed* Cont *replay, associativity comparison, dependency provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.CompilerCompositionTraceUp.CompilerCompositionTraceTasteGate single carrier alignment`

Proof. Project the displayed carrier. The packet exposes only the endpoint certificate rows S, M, T, the two adjacent compiler packets K_{SM}, K_{MT}, and the middle compatibility row J before the composite graph is read.

The composite graph obligation is then forced through G. A source-to-target edge is accepted only when it factors through a middle history already displayed by M and consumed by both adjacent compiler packets. The classifier obligation is the row L, which first reads the source classifier surface of K_{SM}, then the shared middle certificate in J, and finally the target classifier surface of K_{MT}.

The remaining certificate data are the transport and replay rows. Componentwise hsame evidence is read from H, continuation compatibility from C, associativity from A, dependency provenance from P, and local naming from N. Since the carrier contains no other coordinate, no host compiler function, semantic equality, quotient row, optimizer theorem, decompiler row, or ambient categorical law can be exported by the local $\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}$ surface. □

Theorem 679.3 (CompilerCompositionTrace associativity)**.** *Let*

$$Q_{012} = (S_0, S_1, S_2, K_{01}, K_{12}, J_1, G_{02}, L_{02}, H_{02}, C_{02}, A_{02}, P_{02}, N_{02})$$

and

$$Q_{123} = (S_1, S_2, S_3, K_{12}, K_{23}, J_2, G_{13}, L_{13}, H_{13}, C_{13}, A_{13}, P_{13}, N_{13})$$

be accepted CompilerCompositionTrace$^{\uparrow}$ *packets sharing the adjacent compiler packet K_{12} and middle certificate rows S_1, S_2. For the three accepted compiler packets K_{01}, K_{12}, and K_{23}, the two displayed composites*

$$(K_{23} \circ K_{12}) \circ K_{01}$$

and

$$K_{23} \circ (K_{12} \circ K_{01})$$

export the same finite public packet: the same source row S_0, target row S_3, relational graph through the displayed middle rows, classifier-transport ledger, hsame *transports,* Cont *routes, package provenance, and local naming boundary. The equality is witnessed only by the associativity rows A_{02}, A_{13} and the componentwise transports recorded in the corresponding H and C rows.*

Lean4 ✓ BEDC.Derived.CompilerCompositionTraceUp.CompilerCompositionTrace associativity

Proof. Project the two accepted trace packets. The first trace exposes the composite from S_0 to S_2 through S_1, and the second exposes the composite from S_1 to S_3 through S_2. Their overlap is the displayed compiler packet K_{12} together with the middle certificate rows that both traces read.

Form the triple graph by finite relational composition. A displayed edge from S_0 to S_3 is accepted exactly when there are middle histories in S_1 and S_2 read by K_{01}, K_{12}, and K_{23} in that order. Parenthesizing the relational composition on the left or on the right does not introduce another endpoint, graph row, or classifier source; it only changes which binary trace row is consulted first.

The classifier and continuation ledgers replay the same rows in the same order. The first compiler transports source classifier membership into S_1, the second transports it into S_2, and the third transports it into S_3. The Cont routes in C_{02} and C_{13} record the same ordered handoff, while H_{02} and H_{13} supply the componentwise sameness comparisons needed to identify the public rows.

Finally read the associativity coordinates. The rows A_{02} and A_{13} state that the two parenthesized binary traces repack to the same displayed triple packet. The carrier has no coordinate for an external compiler function, semantic program equality, quotient collapse, or global category law, so the exported associativity evidence is exhausted by the displayed graph, classifier, transport, continuation, provenance, and local naming rows. □

Theorem 679.4 (CompilerCompositionTrace ContinuationMonad sibling route)**.** *Every accepted* $\mathsf{CompilerCompositionTrace}^{\uparrow}$ *packet consumes its continuation replay and classifier-preserving morphism data through the sibling route chapter 485, chapter 481, chapter 473, and chapter 590. The composite trace reads the continuation rows, adjacent NameCert morphism rows, compiler graph rows, and classifier transport rows from those concrete certificates, not from a host compiler function, quotient equality, semantic program equality, or global categorical law.*

Proof. Project the carrier Q. Its continuation replay row C, classifier ledger L, adjacent compiler packets K_{SM}, K_{MT}, and morphism provenance row P are displayed coordinates of the same finite trace.

The continuation component is read through chapter 485; the endpoint and middle certificate movement is read through chapter 481; the adjacent compiler graph data are read through chapter 473; and the classifier-preserving transport is read through chapter 590.

Finally use Definition 679.2 and Definition 679.3. They keep all composite trace data inside $S, M, T, K_{SM}, K_{MT}, J, G, L, H, C, A, P, N$, so the sibling route cannot be replaced by a host compiler function or quotient equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompilerCompositionTrace}^{\uparrow}$ packet is generated from source, middle, and target NameCert rows, two adjacent $\mathsf{CertificateCompiler}^{\uparrow}$ packets, middle-certificate compatibility, a composite graph, classifier transport, componentwise hsame transports, displayed Cont routes, associativity comparison, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 679.1, Definition 679.2, and Definition 679.3 bind the finite compiler-composition trace surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompilerCompositionTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host compiler semantics, denotational program equality, optimizer correctness outside the displayed rows, decompiler behavior, quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjacent compiler acceptance, middle-certificate compatibility, composite graph formation, classifier transport, continuation replay, associativity comparison, and non-escape for the same finite trace packet.

Chapter 680

A Concrete Naming Certificate for $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$

The $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ packet records the finite selector route used when a Real and Completion diagonal-limit consumer asks for cofinal tail data through dyadic windows. It packages the requested precision, dyadic window family, selected cofinal tail, RegSeqRat readback, RealUp seal handoff, componentwise transport, continuation replay, provenance, and local naming in one displayed BHist carrier. The packet is a finite NameCert surface for cofinal-window selection; it is not an ambient compactness argument, quotient-stream equality, host-selected limit, countable choice rule, or hidden tail selector.

The finite cofinal family used here is scheduled by chapter 681 and later consumed by diagonal-limit selectors through chapter 697. This is a dependency on the displayed tail schedule and diagonal consumer route, not a self-reference to the local carrier.

Definition 680.1 (Dyadic cofinal tail selector carrier)**.** *A* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (e, \mathcal{D}, t, w, r, a, h, c, p, n)$$

with the following displayed rows:

1. *e is the positive* $\mathsf{Rat}^{\uparrow}$ *precision request made by the diagonal-limit consumer;*
2. *$\mathcal{D}$ is the finite dyadic cofinal-window family opened at that request;*
3. *t is the selected cofinal tail index, displayed as a row in the same finite packet;*
4. *w is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window determined by t and $\mathcal{D}$;*
5. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the selected dyadic window;*
6. *a is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only through w and r;*
7. *h records componentwise* hsame *transports for the request, dyadic family, selected tail, window, readback, seal, and structural rows;*
8. *c records the displayed* Cont *route from e through $\mathcal{D}, t, w, r$ into a;*
9. *p is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*
10. *n is the local* $\mathsf{NameCert}_{\mathsf{DyadicCofinalTailSelector}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $e, \mathcal{D}, t, w, r, a, h, c, p, n$.

Theorem 680.2 (Dyadic cofinal tail selector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicCofinalTailSelector}^{\uparrow}}$ *surface for Definition 680.1 consists of carrier admission for $S = (e, \mathcal{D}, t, w, r, a, h, c, p, n)$, positive* $\mathsf{Rat}^{\uparrow}$ *request exposure through e, finite dyadic cofinal-window family exposure through $\mathcal{D}$, selected tail exposure through t, finite* $\mathsf{StreamName}^{\uparrow}$ *window readback through w,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through r,* $\mathsf{Real}^{\uparrow}$ *seal delivery through a, componentwise* hsame *stability through h, displayed* Cont *route stability through c,* Pkg *provenance through p, and local naming through n.*

Proof. Carrier admission is projection from the displayed tuple in Definition 680.1. The request, dyadic family, selected tail, window, readback, seal, transports, routes, provenance, and naming row are all visible coordinates of one finite packet.

The selector obligations are read in order. The request row e opens only the finite dyadic family $\mathcal{D}$, the selected tail t is recorded before any window row can be consumed, and the window row w is the only source for the RegSeqRat handoff r.

The RealUp seal row a is downstream of w and r, while h, c, p, n provide transport, route replay, provenance, and naming. Since the carrier has no coordinate for host-selected limits, quotient-stream equality, countable choice, or hidden selectors, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.DyadicCofinalTailSelectorUp.DyadicCofinalTailSelectorTasteGate single carrier alignment`

Theorem 680.3 (Dyadic cofinal-window exhaustion)**.** *For an accepted* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *carrier*

$$S = (e, \mathcal{D}, t, w, r, a, h, c, p, n),$$

every cofinal-window read made by a Real and Completion diagonal-limit consumer is exhausted by the displayed finite dyadic family $\mathcal{D}$, the selected tail row t, the finite StreamName window w, the RegSeqRat readback row r, and the structural rows h, c, p, n. No consumer can replace this route by an unbounded window search, countable choice selector, host supremum argument, or external tail-selection function.

Proof. Project the accepted carrier. The only rows that carry cofinal-window data are $\mathcal{D}$, t, and w, and they are all displayed before the RegSeqRat and RealUp handoff rows.

The continuation row c routes the precision request through that finite family and selected tail before any readback is available. Componentwise transport along h preserves the same displayed rows, while p and n record provenance and local naming for the same packet.

Thus every cofinal-window read is already inside $e, \mathcal{D}, t, w, r, a, h, c, p, n$. The carrier provides no coordinate for an unbounded search, countable choice selector, host supremum, or external tail-selection function. □

Theorem 680.4 (Dyadic cofinal tail selector Real seal handoff)**.** *For an accepted* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row a is reached only after the selected cofinal tail t has fixed the finite dyadic window family $\mathcal{D}$, the StreamName window row w, and the RegSeqRat readback row r. The real-facing handoff therefore factors through the displayed route*

$$e, \mathcal{D}, t, w, r, h, c, p, n$$

before it reaches a.

Proof. Read the carrier rows in order. The seal row a appears after the selected tail, finite window, and RegSeqRat readback rows, so a real-facing consumer cannot inspect the seal before the finite dyadic route has been fixed.

Use Definition 680.3. It confines the cofinal-window read to $\mathcal{D}, t, w, r, h, c, p, n$ and removes any hidden selector route.

The remaining structural rows transport and replay the same packet. Hence the RealUp handoff reaches a only through the displayed finite route and cannot import an ambient limit or quotient-stream comparison. □

Theorem 680.5 (Dyadic cofinal tail selector non-escape)**.** *An accepted* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple* $e, \mathcal{D}, t, w, r, a, h, c, p, n$*. In particular, it exports no unbounded dyadic window family, second tail selector, quotient of regular sequences, host real limit, countable choice rule, or RealUp completeness principle.*

Proof. The NameCert obligations identify every row available to a downstream consumer. The request, dyadic family, selected tail, window, readback, and seal rows are the only content rows, and h, c, p, n only transport, replay, package, and name those rows.

Apply the cofinal-window exhaustion theorem to any attempted window read. It forces the read through $\mathcal{D}, t, w, r, h, c, p, n$.

Apply the Real seal handoff theorem to any real-facing read. It forces the read through the selected dyadic window and RegSeqRat handoff before the seal row a can be consumed.

Since no additional coordinate is present, the packet has no source for unbounded windows, a second selector, quotient regular sequences, host real limits, countable choice, or a RealUp completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCofinalTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ packet is generated from a positive RatUp request, a finite dyadic cofinal-window family, a selected tail row, finite StreamName readback, RegSeqRat handoff, RealUp seal delivery, componentwise hsame transports, displayed Cont routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCofinalTailSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 680.1, Definition 680.2, Definition 680.3, Definition 680.4, and Definition 680.5 bind the finite dyadic cofinal-tail selector carrier and its local NameCert obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCofinalTailSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host real limits, countable choice, hidden tail selection, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic family exposure, selected-tail coverage, StreamName readback, RegSeqRat handoff, RealUp seal delivery, route stability, provenance, and non-escape for the same finite packet.

Chapter 681

A Concrete Naming Certificate for $\mathsf{TailCofinalitySchedule}^{\uparrow}$

The $\mathsf{TailCofinalitySchedule}^{\uparrow}$ packet records the finite shared-threshold schedule used when a diagonal $\mathsf{Real}^{\uparrow}$ consumer must pass from dyadic tolerances and stream windows to $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows and a $\mathsf{Real}^{\uparrow}$ seal. It names the schedule as an inspectable BHist surface, not as a host limit selector, quotient-stream equality, selected real, or ambient completeness principle.

The schedule is the consumer-facing route from chapter 126 through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, so its carrier records the finite L10 handoff rather than an isolated tail oracle.

Definition 681.1 (Tail cofinality schedule carrier)**.** *A* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (E, W, D, R, L, H, C, P, N)$$

with the following displayed coordinates:

1. *E is the unary precision request row naming the tolerance level demanded by the diagonal consumer;*
2. *W is the cofinal finite-window schedule selected for that request over the relevant* $\mathsf{StreamName}^{\uparrow}$ *rows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger proving that the windows in W meet the requested dyadic bound;*
4. *R is the family of* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows made available after the window and dyadic ledgers have been consumed;*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row through which the diagonal consumer receives the scheduled regular-rational handoff;*
6. *H records componentwise* hsame *transports for E, W, D, R, L and the structural rows of the packet;*
7. *C records the displayed* Cont *routes from E to W, from W to D, from D to R, and from R to L;*
8. *P records* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient stream equality, a host-selected limit, ambient completeness, a hidden tail oracle, or an unbounded schedule.

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleUp StdBridge`

Theorem 681.2 (Tail cofinality schedule NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}$ *surface for Definition 681.1 consists exactly of carrier admission for $T = (E, W, D, R, L, H, C, P, N)$, unary precision exposure through E, cofinal finite-window coverage through W, threshold sufficiency through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R,* $\mathsf{Real}^{\uparrow}$ *seal boundary through L, componentwise* hsame *transport through H, displayed* Cont *route replay through C,* Pkg *provenance through P, local naming through N, and non-escape from quotient streams, selected host limits, ambient completeness, hidden tail oracles, and unbounded schedules.*

Proof. Carrier admission is projection from the displayed packet. The rows E, W, D, R, L expose the precision request, finite cofinal schedule, dyadic tolerance ledger, regular-rational readback, and real-facing seal.

The schedule and threshold obligations are read in order. The route from E to W selects the finite cofinal windows, and the route from W to D records the dyadic sufficiency ledger for those same windows.

The handoff and seal obligations then consume R and L. Componentwise transport, route replay, provenance, and naming are exactly the rows H, C, P, N. Since no further analytic coordinate occurs in T, the listed rows exhaust the certificate surface. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier namecert obligations`

Theorem 681.3 (Tail cofinality schedule diagonal handoff)**.** *Every diagonal* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *carrier factors through the finite rows*

$$E, W, D, R, L, H, C, P, N.$$

The consumer first reads the unary precision request, then the cofinal finite-window schedule, then the dyadic tolerance ledger, then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and finally the* $\mathsf{Real}^{\uparrow}$ *seal row. It cannot read quotient streams, selected host limits, ambient completeness, a hidden tail oracle, or an unbounded schedule.*

Proof. Start with the carrier and the NameCert obligation surface. They display the only route from the requested precision to the real-facing seal as

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The cofinal window coverage and threshold sufficiency clauses force the dyadic ledger to be consumed before any regular-rational row is available. Thus a diagonal handoff cannot jump from the precision request directly to a $\mathsf{RegSeqRat}^{\uparrow}$ witness or to a $\mathsf{Real}^{\uparrow}$ seal.

The rows H, C, P, N replay and name the same route under transport and provenance. Because the carrier has no coordinate for quotient stream equality, host-selected limits, ambient completeness, hidden tail oracles, or unbounded schedules, every diagonal consumer remains inside the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier diagonal handoff`

Theorem 681.4 (Tail cofinality schedule finite-window coverage)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *carrier. Every unary precision request exposed by E is routed through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W before any* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row is readable. The route has no quotient-stream equality, host-selected limit, hidden tail oracle, or unbounded schedule coordinate.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier window coverage`

Proof. Project the carrier. The only precision row is E, and the only finite-window schedule row attached to it is W.

Apply Definition 681.2. The local certificate lists cofinal finite-window coverage through W as the obligation immediately downstream from unary precision exposure through E.

Finally use Definition 681.3. Its displayed handoff route begins $E \longmapsto W$, so a consumer cannot bypass W to reach D, R, or L. Since the carrier contains no analytic escape coordinate, the finite-window route is exhaustive. □

Theorem 681.5 (Tail cofinality schedule threshold sufficiency)**.** *For an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *carrier $T = (E, W, D, R, L, H, C, P, N)$, every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row in R is available only after the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D records the dyadic threshold supplied by the finite-window schedule W. No regular-rational readback is supplied by a host limit selector, ambient completeness principle, hidden tail oracle, or unbounded schedule.*

Proof. Read W, D, R from the displayed packet. The schedule row W names the finite stream windows, and the dyadic ledger D records the threshold bound for those same windows.

Apply Definition 681.2. It places threshold sufficiency through D between finite-window coverage and $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The diagonal handoff theorem then gives the route $W \longmapsto D \longmapsto R$. Thus every regular-rational readback consumes the dyadic ledger first, and the packet has no coordinate from which an unbounded or host-selected readback could be recovered. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier threshold sufficiency`

Theorem 681.6 (Tail cofinality schedule Real seal readback)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *consumer reaches the seal row L only after consuming the precision row E, the finite-window row W, the dyadic ledger D, the regular-rational readback row R, and the structural rows H, C, P, N. The seal readback does not expose quotient streams, host-selected limits, ambient completeness, hidden tail oracles, or unbounded schedules.*

Proof. Start with the carrier and the obligation theorem. They identify L as the real-facing seal row and identify H, C, P, N as the transport, continuation, package, and naming rows that replay the displayed route.

Apply Definition 681.3. It orders the analytic rows as $E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$.

It remains to account for the structural rows. The same handoff theorem and the NameCert obligations keep H, C, P, N as replay and provenance for that route only. Since no additional real-construction coordinate occurs in T, every $\mathsf{Real}^{\uparrow}$ readback through L has already consumed the displayed schedule surface. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier real seal readback`

Theorem 681.7 (Tail cofinality schedule RegularCauchyTailMeet handoff)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. Suppose its shared threshold row D is supplied by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *packet whose paired source, window, modulus, threshold, comparison, transport, route, provenance, and local naming rows are transported into the same displayed schedule surface. Then every diagonal* $\mathsf{Real}^{\uparrow}$ *consumer of T reads the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *paired source, window, modulus, and threshold rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the* $\mathsf{Real}^{\uparrow}$ *seal row L.*

Proof. Start from the schedule route of Definition 681.6. Every seal-facing consumer follows

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

Identify D with the shared dyadic threshold ledger supplied by the accepted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ packet. The tail-meet obligation package in Definition 560.9 exposes the paired source rows, finite window rows, source modulus rows, shared threshold row, comparison row, transports, routes, provenance, and local name before any downstream consumer can use that threshold.

Now apply the tail-meet real-seal handoff from Definition 560.4. It keeps the shared threshold inside the displayed tail-meet packet and forbids replacement by quotient-stream comparison, selected limit, host real equality, hidden tail oracle, or ambient completeness.

Therefore the schedule consumer reaches R and L only after the RegularCauchyTailMeet paired source, window, modulus, and threshold rows have already been read through D and the structural rows H, C, P, N. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier regular cauchy tail meet handoff`

Theorem 681.8 (Tail cofinality schedule scoped binding)**.** *Every accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet $T = (E, W, D, R, L, H, C, P, N)$ binds its precision request, finite-window schedule, dyadic threshold ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *route,* Pkg *provenance, and* $\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}$ *naming row through exactly the displayed BEDC primitives* BHist, hsame, Cont, Pkg, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{NameCert}$. *When D is supplied by* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *the shared-threshold handoff remains part of the same displayed scope before R and L are readable. No accepted packet supplies quotient stream equality, a host-selected limit, ambient completeness, a hidden tail oracle, or an unbounded schedule as an alternative source for any of these rows.*

Proof. Start with the carrier definition. It displays the whole packet as a finite BHist surface

$$T = (E, W, D, R, L, H, C, P, N),$$

where E is the precision row, W is the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback family, L is the $\mathsf{Real}^{\uparrow}$ seal, and H, C, P, N are the transport, route, provenance, and naming rows.

Apply Definition 681.2. It identifies carrier admission, precision exposure, finite-window coverage, threshold sufficiency, regular-rational handoff, real seal boundary, componentwise transport, route replay, provenance, and local naming as the complete local certificate surface for that same tuple.

The analytic route is then fixed by the diagonal handoff, finite-window coverage, threshold sufficiency, and real-seal readback theorems. Together they force every consumer to read

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

and to carry the same route through H, C, P, N.

Finally use Definition 681.7 when the ledger row is supplied by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$. The shared threshold remains inside the displayed tail-meet rows before R and L are readable, so the schedule has no separate tail oracle.

Since Definition 681.1 contains no coordinate for quotient stream equality, selected host limits, ambient completeness, hidden tail oracles, or unbounded schedules, no accepted packet can factor any listed row through such data. The only available factors are the displayed BEDC primitives named above, together with the local $\mathsf{NameCert}$ row and the threshold handoff already routed through D. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier scoped binding`

Theorem 681.9 (Tail cofinality schedule RegularCauchyCompletion route)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ packet whose finite-window and dyadic-threshold route is consumed by an accepted $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal packet. Then every completion-facing seal read follows*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

before the completion carrier can read its $\mathsf{Real}^{\uparrow}$ seal. The route exports no quotient stream equality, selected host limit, host real equality, ambient completeness, hidden tail oracle, or unbounded schedule.

Proof. Start from Definition 681.3. It gives the ordered finite route from the unary request E through the finite stream-window schedule W, dyadic threshold ledger D, regular-rational readback row R, and real-facing seal row L.

Apply Definition 681.6 and Definition 681.8 for the same packet. These rows keep the seal read inside BHist, hsame, Cont, Pkg, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the local $\mathsf{NameCert}$ row, with no coordinate for a host-selected limit or hidden tail oracle.

Now read the accepted $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal packet. Its diagonal handoff Definition 360.3 consumes the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the completion-facing $\mathsf{Real}^{\uparrow}$ seal is available.

Therefore a completion carrier using T cannot reach its seal before the displayed schedule route has been consumed. The exclusions from the schedule packet and the completion diagonal handoff combine to rule out quotient stream equality, selected host limits, host real equality, ambient completeness, hidden tail oracles, and unbounded schedules as alternative routes. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier regular cauchy completion route`

Theorem 681.10 (Tail cofinality schedule shared-tail budget factorization)**.** *Every accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ packet consumed simultaneously by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ factors through one finite route: the precision row E, the $\mathsf{StreamName}^{\uparrow}$ finite window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ threshold ledger D, the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared threshold surface, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, the $\mathsf{Real}^{\uparrow}$ seal row L, and the completion-facing agreement rows of $\mathsf{RealTailAgreementSeal}^{\uparrow}$. The route exports no quotient streams, selected limits, host real equality, ambient completeness, hidden tail oracle, or second tail-budget source.*

Proof. Begin with the local schedule handoff. By Definition 681.7, the dyadic threshold ledger D is consumed as the displayed shared-threshold surface before any $\mathsf{RegSeqRat}^{\uparrow}$ readback row R or $\mathsf{Real}^{\uparrow}$ seal row L becomes available.

Next apply Definition 681.9. This puts every completion-facing seal read on the same route

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

Now read the sibling agreement seal. The RealTailAgreementSeal budget-source lock, shared-tail budget extraction, and Completion handoff totality theorems, Definition 699.10, Definition 699.63, and Definition 699.65, consume the shared tail request through the finite agreement route before a Real or Completion consumer can inspect the local agreement row.

The two routes meet at the same finite window and threshold surface. Since neither carrier contains a coordinate for quotient streams, selected limits, host real equality, ambient completeness, hidden tail oracles, or an external tail-budget source, the combined Real and Completion read factors through the displayed schedule rows and the sibling agreement rows only. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier shared tail budget factorization`

Theorem 681.11 (Tail cofinality schedule diagonal selector nonempty)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed by the diagonal completion route. Then the precision row E, finite-window row W, dyadic ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, terminal* $\mathsf{Real}^{\uparrow}$ *seal row L, and structural rows H, C, P, N supply a nonempty selector route for every completion-facing consumer. The route contains no hidden tail oracle, selected host limit, quotient stream, ambient completeness row, or second completion-facing budget source.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier diagonal selector nonempty`

Proof. Begin with Definition 681.3. It orders the accepted schedule route from the precision request through the finite window, dyadic threshold ledger, regular-rational readback, and Real seal.

Next apply Definition 681.9. A diagonal completion consumer reaches its seal only after the same E, W, D, R, L route has been consumed, with the structural rows carrying transport, replay, provenance, and local naming.

Finally use Definition 681.10. The shared-tail budget is the only factorization available to the simultaneous tail-meet, stream-window, agreement-seal, and completion consumers, so the selector route is present and cannot be replaced by an omitted analytic or host-side coordinate. □

Theorem 681.12 (Tail cofinality schedule selector-budget lock)**.** *Every accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *selector-budget route consumed by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The row D is locked to the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-tail budget and to the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *agreement rows before any public* $\mathsf{Real}^{\uparrow}$ *seal can be read. The route exports no quotient stream, selected limit, countable choice, ambient completeness theorem, hidden tail oracle, or second selector-budget source.*

Proof. Begin with the Real-facing readback route of Definition 681.6. It keeps the selector-budget consumer inside the finite schedule rows $E, W, D, R, L, H, C, P, N$ before the $\mathsf{Real}^{\uparrow}$ seal is public.

Next apply Definition 681.7. This identifies the dyadic row D with the shared-threshold surface supplied by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, so the downstream read cannot bypass the finite tail budget.

Finally use Definition 681.10 together with the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ completion handoff rows cited there. The agreement rows and the completion route consume the same E, W, D, R, L spine,

and the carrier inventory has no coordinate for quotient streams, selected limits, countable choice, ambient completeness, or a second budget source. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier selector budget lock`

Theorem 681.13 (Tail cofinality schedule open-phase row exhaustion)**.** *Every accepted* TailCofinalitySchedule$^{\uparrow}$ *packet used by a* RealCauchyCompletion$^{\uparrow}$ *diagonal read must expose the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L,$$

the RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold surface, the* RealTailAgreementSeal$^{\uparrow}$ *agreement rows, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row before any* Real$^{\uparrow}$ *seal is public. The open-phase route exports no quotient-stream equality, selected host limit, countable choice, ambient completeness, hidden tail oracle, unbounded schedule, or second tail-budget source.*

Proof. Begin with the diagonal handoff. Apply Definition 681.3 and Definition 681.6. They force the completion-facing consumer through

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

before the Real$^{\uparrow}$ seal row can be read.

Next apply Definition 681.9. The RealCauchyCompletion$^{\uparrow}$ diagonal read is placed on the same finite route, with the structural rows carrying only componentwise transport, displayed replay, provenance, and local naming.

Now use Definition 681.10. The RegularCauchyTailMeet$^{\uparrow}$ shared-threshold surface and the RealTailAgreementSeal$^{\uparrow}$ agreement rows are consumed as rows of the same schedule route rather than as independent tail sources.

Finally invoke Definition 681.12. It prevents a second selector budget, hidden tail oracle, unbounded schedule, quotient-stream equality, selected host limit, countable choice, or ambient completeness witness from replacing D or bypassing the displayed route. Hence the listed open-phase rows are exhausted before any public Real$^{\uparrow}$ seal is available. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier open phase row exhaustion`

Theorem 681.14 (Tail cofinality schedule Real-completion diagonal route)**.** *Every accepted* TailCofinalitySchedule$^{\uparrow}$ *packet consumed by a* RealCauchyCompletion$^{\uparrow}$ *diagonal read reaches the public* Real$^{\uparrow}$ *seal only through the displayed finite route: the* StreamName$^{\uparrow}$ *finite-window row, the* DyadicRatCore$^{\uparrow}$ *ledger, the* RegSeqRat$^{\uparrow}$ *readback row, the* RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold row, shared-tail budget factorization, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *rows. The route exports no quotient-stream equality, selected host limit, countable choice, host real equality, ambient completeness, hidden tail oracle, unbounded schedule, or second tail-budget source.*

Proof. Begin with the local diagonal handoff. By Definition 681.3, every diagonal consumer of the schedule packet reads the precision request, finite-window row, dyadic ledger, regular-rational readback row, and real seal in the displayed order.

Next apply Definition 681.4 and Definition 681.6. These theorems keep the finite window and the public Real$^{\uparrow}$ seal inside the packet rows W, D, R, L, H, C, P, N, together with componentwise hsame, Cont, Pkg, and the local NameCert row.

The tail source is then fixed by Definition 681.7. When the dyadic ledger is supplied by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, the paired source, window, modulus, and shared-threshold rows are read before the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

Finally combine Definition 681.9 with Definition 681.10. The first places the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal handoff on the same schedule route, and the second makes the shared-tail budget the only factorization available to simultaneous $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers.

The sibling completion chapter chapter 360 supplies the corresponding completion-facing diagonal route. Since the schedule carrier and the completion route contain no coordinate for quotient-stream equality, selected host limits, countable choice, host real equality, ambient completeness, hidden tail oracles, unbounded schedules, or a second budget source, the public seal can be reached only through the displayed BEDC rows. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier real completion diagonal route`

Theorem 681.15 (Tail cofinality schedule completion-budget non-escape)**.** *Let $T = (E, W, D, R, L, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. Every completion-facing consumer of T factors through the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

together with the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *agreement rows before any* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal read is available. The accepted packet exports no quotient-stream equality, selected host limit, host real equality, ambient completeness, hidden tail oracle, unbounded schedule, countable choice, or second budget source.*

Proof. First use Definition 681.14. It places every completion-facing diagonal read on the displayed schedule route through E, W, D, R, and L, with the structural rows H, C, P, N carrying only transport, route replay, provenance, and local naming.

Next apply Definition 681.10. The shared threshold supplied by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and the agreement rows supplied by $\mathsf{RealTailAgreementSeal}^{\uparrow}$ are consumed on that same finite route before the completion seal can be inspected.

Finally project the carrier in Definition 681.1. Its coordinates are exhausted by $E, W, D, R, L, H, C, P, N$, so none of the excluded analytic or choice data is available as an exported route or as a second source for the completion-facing budget. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier completion budget nonescape`

Theorem 681.16 (Tail cofinality schedule transport determinacy)**.** *Let*

$$T_0 = (E_0, W_0, D_0, R_0, L_0, H_0, C_0, P_0, N_0) \quad \text{and} \quad T_1 = (E_1, W_1, D_1, R_1, L_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packets. Suppose the two packets agree after componentwise* hsame *transport on the rows $E, W, D, R, L, H, C, P, N$, and suppose their shared-tail rows are read by the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer routes. Then every completion-facing* $\mathsf{Real}^{\uparrow}$ *seal read exposed by either packet factors through one identical finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The common route carries the transported schedule rows, the shared finite tail threshold, the agreement seal, and the completion handoff, and it exports no quotient stream, selected limit, countable choice, host real equality, ambient completeness, hidden tail oracle, or second budget source.

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier transport determinacy`

Proof. First use Definition 681.3 and Definition 681.6. These results place each accepted packet's diagonal consumer and public $\mathsf{Real}^{\uparrow}$ seal read on the ordered schedule rows $E, W, D, R, L, H, C, P, N$.

The componentwise hsame hypotheses identify the two transported schedules row by row. In particular, the finite window, dyadic ledger, regular readback, and public seal rows are not separate routes after transport; they are the same displayed spine read with two presentations.

Next apply Definition 681.10. The common $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumers force both packets to use the same shared-tail budget before a completion-facing seal can be inspected.

Finally combine Definition 681.12 with Definition 681.15. The selector budget is locked to the finite schedule route, and the completion read has no exit from that route to a quotient stream, selected limit, countable choice, host real equality, ambient completeness, hidden tail oracle, or second budget source.

Thus any completion-facing $\mathsf{Real}^{\uparrow}$ seal read from T_0 and any such read from T_1 are transported presentations of the same finite BEDC route $E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$. □

Theorem 681.17 (Tail cofinality schedule diagonal-tail-selector factorization)**.** *Every accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed by an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public budget source factors through the same finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The selector rows supply only the displayed public budget source, $\mathsf{ProbeBundle}$ *and* Pkg *support,* hsame *transport, and* Cont *replay. The factorization exposes no quotient stream, selected host limit, countable choice, ambient completeness, hidden tail oracle, or second budget source.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier diagonal tail selector factorization`

Proof. Start with the schedule route from Definition 681.14. It places every completion-facing diagonal read of the accepted schedule packet on the ordered path from the finite-window row W through the dyadic ledger D, the regular readback row R, and the public $\mathsf{Real}^{\uparrow}$ seal row L.

Next apply Definition 697.40. The accepted selector consumes the same tail-cofinality packet only as a public budget source, with the selector rows contributing the displayed $\mathsf{ProbeBundle}$ support, Pkg provenance, hsame transport, and Cont replay.

Finally use Definition 697.46. The selector budget cannot escape through a second source before the $\mathsf{Real}^{\uparrow}$ seal is read. Since neither carrier contains coordinates for quotient streams, selected host limits, countable choice, ambient completeness, hidden tail oracles, or another budget source, the joint consumer path is exactly the displayed schedule route. □

Theorem 681.18 (Tail cofinality schedule selector-completion pullback)**.** *Let* $T = (E, W, D, R, L, H, C, P, N)$ *be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet consumed both by an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public budget source and by a* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *diagonal read. Then the selector-facing read and the completion-facing seal read pull back to one finite public surface*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

with structural rows H, C, P, N*. No quotient stream, selected host limit, countable choice, ambient completeness, hidden tail oracle, independent completion endpoint, or second budget source distinguishes the selector-facing and completion-facing public surfaces.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier selector completion pullback`

Proof. Apply Definition 681.17. The selector-facing consumer can read the accepted schedule only through the displayed route, with the selector carrier contributing public budget, ProbeBundle support, Pkg provenance, hsame transport, and Cont replay.

For the completion-facing read, use Definition 681.9 and Definition 681.14. These place the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal handoff on the same ordered schedule route before the public $\mathsf{Real}^{\uparrow}$ seal is available.

It remains to identify the common budget surface. By Definition 681.10, the simultaneous $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, and $\mathsf{RealCauchyCompletion}^{\uparrow}$ consumers all factor through the same finite window, dyadic ledger, shared threshold, readback, and seal rows.

Thus the two reads have the same pullback surface $(E, W, D, R, L, H, C, P, N)$. Since the schedule, selector, and completion carriers contain no coordinate for the excluded host or oracle data, no second endpoint or budget source can separate the selector-facing and completion-facing public surfaces. □

Theorem 681.19 (Tail cofinality schedule selector-seal pullback)**.** *Let $T = (Q, W, D, R, E_T, H_T, C_T, P_T, N_T)$ be an accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet, and let $S = (B, W, R, L, E_S, H_S, C_S, P_S, N_S)$ be an accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose the selected* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed by the two packets are the same displayed rows W and R. Then the schedule-facing completion read and the selector-facing seal read pull back to one finite public surface*

$$B,\ Q,\ W,\ D,\ R,\ L,\ E_T,\ E_S,\ H,\ C,\ P,\ N.$$

Here B is the selector budget, Q is the precision row, D is the dyadic ledger, L is the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row, E_T, E_S are the terminal* $\mathsf{Real}^{\uparrow}$ *rows, and H, C, P, N denote the structural transport, continuation, provenance, and naming rows read componentwise from $H_T, H_S, C_T, C_S, P_T, P_S, N_T, N_S$. The pullback uses no quotient stream equality, host real equality, countable choice, detached limit, hidden selector row, ambient completeness principle, or second completion route.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier selector seal pullback`

Proof. Begin with Definition 681.18. It places the schedule-facing completion read on the finite route from the precision row through the common window, dyadic ledger, readback, terminal Real row, and structural rows. Thus the schedule side contributes only $Q, W, D, R, E_T, H_T, C_T, P_T, N_T$.

Next apply Definition 1081.5 and Definition 1081.16. The selector-seal packet reaches the L10 handoff through the ordered surface B, W, R, L, E_S and locks the selected window and readback to the displayed rows W and R, so its selector-facing read meets the schedule-facing read on the same two handoff rows.

Finally use Definition 1081.11 and Definition 1081.17. The terminal selector-seal route has no remaining completion endpoint beyond the displayed L and E_S rows, and H_S, C_S, P_S, N_S only package that route. Together with the schedule pullback, the common public surface is exactly $B, Q, W, D, R, L, E_T, E_S, H, C, P, N$.

The excluded objects are absent from both carriers and from the cited terminality and non-promotion results. Hence quotient stream equality, host real equality, countable choice, detached limits, hidden selector rows, ambient completeness, and a second completion route cannot distinguish the two completion reads. □

Theorem 681.20 (Tail cofinality schedule reciprocal-window factorization)**.** *Let $T = (E, W, D, R, L, H_T, C_T, P_T, N$ be an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet, and let $B = (X, A, M, W, D, R, S, H_B, C_B, P_B, N_B)$ be an accepted* CauchyInverseBudget$^{\uparrow}$ *packet. Suppose the two packets expose the same finite* StreamName$^{\uparrow}$ *window row W,* DyadicRatCore$^{\uparrow}$ *ledger row D,* RegSeqRat$^{\uparrow}$ *readback row R, and* Real$^{\uparrow}$ *seal row, with the budget seal S displayed as the schedule row L. Then every reciprocal-root or completion-facing read factors through the common finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

together with the CauchyInverseBudget$^{\uparrow}$ *apartness row A before any reciprocal candidate is consumed. This factorization uses the reciprocal-window surface of chapter 703, Definition 703.13, Definition 703.14, Definition 681.14, and Definition 681.19; it exposes no quotient stream equality, host real equality, selected limit, countable choice, hidden tail oracle, or ambient completeness coordinate.*

Proof. Project the two carrier tuples. The schedule packet supplies only $E, W, D, R, L, H_T, C_T, P_T, N_T$, while the reciprocal budget supplies only $X, A, M, W, D, R, S, H_B, C_B, P_B, N_B$. The shared rows named in the statement therefore identify the public window, ledger, readback, and Real seal that both packets can expose.

Apply Definition 681.14. It fixes the completion-facing schedule route as

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The selector-side pullback of Definition 681.19 keeps the same finite window and readback rows visible when a selector or completion seal is read.

Now use the reciprocal-window chapter chapter 703. By Definition 703.13, a reciprocal-root consumer must display the apartness row A, modulus row M, finite window W, and dyadic row D before any read of D, R, or the seal row $S = L$ can serve a reciprocal candidate.

Finally apply Definition 703.14. The RegSeqRat$^{\uparrow}$ readback and Real$^{\uparrow}$ seal received from the reciprocal budget are exactly the public rows R and $S = L$ already fixed by the schedule route. Thus the two packets share one finite readback surface, with the apartness row inserted before the reciprocal candidate is consumed.

The remaining exclusions are coordinate exclusions. Neither displayed carrier contains quotient stream equality, host real equality, selected limits, countable choice, hidden tail oracles, or ambient completeness data, so no root or completion consumer can factor through such data instead of the displayed route and apartness row. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier cauchy inverse budget factorization`

Theorem 681.21 (Tail cofinality schedule selector-residue exhaustion)**.** *Let T be an accepted* TailCofinalitySchedule$^{\uparrow}$ *packet consumed through* DiagonalTailSelector$^{\uparrow}$, RealCompletionSelectorSeal$^{\uparrow}$, RealTailAgreementSeal$^{\uparrow}$, *and* RealCauchyCompletion$^{\uparrow}$. *Then every remaining selector-facing, completion-facing, and real-seal-facing public read is exhausted by the finite rows*

$$E, W, D, R, L, H, C, P, N$$

of the schedule carrier, together with the selector budget row, the agreement rows, and the selector-seal handoff rows named in the preceding pullback theorems. No residual row supplies quotient stream equality, selected host limits, countable choice, detached limits, hidden selector rows, ambient completeness, hidden tail oracles, or a second completion route.

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier selector residue exhaustion`

Proof. First place the selector read on the schedule route. Apply Definition 681.17. It sends the accepted selector consumer through the displayed tail schedule, with the selector contributing only its public budget row, ProbeBundle support, Pkg provenance, hsame transport, and Cont replay.

Next identify the completion-facing route. Use Definition 681.18. The selector-facing read and the RealCauchyCompletion$^{\uparrow}$ read meet on the same finite surface $E, W, D, R, L, H, C, P, N$, with no independent completion endpoint or second budget source.

Now attach the selector-seal handoff. By Definition 681.19, the RealCompletionSelectorSeal$^{\uparrow}$ read and the schedule-facing completion read share the displayed selector budget, precision row, window row, dyadic ledger, readback row, limit handoff, terminal Real$^{\uparrow}$ rows, and structural transport, continuation, provenance, and naming rows.

Finally use the terminality and structural-row nonpromotion results for the selector seal, Definition 1081.11 and Definition 1081.17. They prevent the handoff rows and the structural rows from becoming an additional limit source, hidden selector surface, ambient completeness witness, or second completion route. The RealTailAgreementSeal$^{\uparrow}$ agreement rows therefore add only the displayed finite agreement handoff already used by the pullback surface, and every public read is exhausted by the named L10 rows. □

Theorem 681.22 (Tail cofinality schedule CauchyLimitSeal readback)**.** *Every accepted* TailCofinalitySchedule$^{\uparrow}$ *packet whose selector residue is exhausted reads its* CauchyLimitSeal$^{\uparrow}$ *boundary only through the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, and* CauchyLimitSeal$^{\uparrow}$ *rows of the schedule route.*

Proof. Project the schedule carrier and place the selector read on the diagonal route using Definition 681.17. The StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ dyadic ledger, RegSeqRat$^{\uparrow}$ readback, terminal Real$^{\uparrow}$ row, and CauchyLimitSeal$^{\uparrow}$ handoff are the displayed rows available to the accepted packet.

Apply the finite-window coverage and completion pullback blocks, including Definition 681.18. They identify the completion-facing read with the same finite surface rather than with an independent limit endpoint.

Now use Definition 681.19 and Definition 681.21. The selector-seal route passes through the selected window, regular-rational readback, CauchyLimitSeal handoff, and terminal real row, and the residue-exhaustion statement leaves no remaining selector-facing or completion-facing public read outside those rows.

No quotient stream, host limit, choice selector, detached limit, hidden selector row, or ambient completeness coordinate is present in the carrier. Thus the CauchyLimitSeal boundary can be read only through the displayed L10 schedule rows. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier cauchy limitseal readback`

Theorem 681.23 (Tail cofinality schedule RealCompletionSelectorSeal route)**.** *For the same accepted* TailCofinalitySchedule$^{\uparrow}$ *packet, every* RealCompletionSelectorSeal$^{\uparrow}$ *consumer route factors through the CauchyLimitSeal readback row of Definition 681.22 and the selector-seal pullback route of Definition 681.19.*

Proof. Start with the selector-facing route of Definition 681.17. It fixes the finite tail schedule, selector budget, transport, replay, provenance, and local naming rows before any selector-seal consumer reads the packet.

Next apply the finite-window and selector-seal pullback results. The selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback are the only route by which the selector seal reaches the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff and the terminal $\mathsf{Real}^{\uparrow}$ row.

Use Definition 681.22 for the limit-seal boundary. The readback theorem has already exhausted the possible CauchyLimitSeal reads through the displayed stream, dyadic, regular-rational, real, and limit-seal rows.

Finally invoke Definition 681.21. Since the exhausted packet has no quotient stream, host limit, choice selector, ambient completeness coordinate, hidden tail oracle, or second completion route, any RealCompletionSelectorSeal consumer route must factor through the displayed CauchyLimitSeal readback and selector-seal pullback rows. □

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier real completion selector route`

Theorem 681.24 (Tail cofinality schedule StreamName RegSeqRat Real window determinacy)**.** *Let*

$$T = (E, W, D, R, L, H, C, P, N)$$

be an accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. If two completion-facing or selector-facing public reads of* T *agree on the displayed unary precision row* E*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*, then they read the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* L *through the structural rows* H, C, P, N*. No quotient stream, host limit, choice-selected tail, ambient completeness row, hidden tail oracle, or second selector-budget source can split the public route after* E, W, D *are fixed.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier streamname regseqrat real window determinacy`

Proof. Begin with the finite schedule. By Definition 681.4, every accepted read of the precision row E reaches the same displayed $\mathsf{StreamName}^{\uparrow}$ window row W before any dyadic ledger, regular-rational readback, or real seal is available.

Next apply Definition 681.5. Once the two public reads agree on W and on the ledger row D, the available $\mathsf{RegSeqRat}^{\uparrow}$ readback is the displayed row R of the same packet rather than a host-selected tail or an independent selector budget.

Now use Definition 681.6. The $\mathsf{Real}^{\uparrow}$ consumer reaches the seal row only through the route

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

Thus agreement on E, W, D fixes the readback row R and then fixes the seal row L under the same transports, continuations, provenance, and local name H, C, P, N.

Finally invoke Definition 681.23, together with the selector-residue exhaustion and CauchyLimitSeal readback statements. These routes leave no quotient stream, ambient completeness coordinate, choice-selected tail, hidden tail oracle, host limit, or second selector-budget source from which a distinct completion-facing or selector-facing public read could be obtained. □

Theorem 681.25 (Tail cofinality schedule window seal factorization)**.** *Let*

$$T = (E, W, D, R, L, H, C, P, N)$$

be an accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. Every completion-facing read of* T *factors through*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L$$

and then through the structural rows H, C, P, N*. The factorization is stable under the displayed* hsame *transports and cannot introduce quotient stream equality, host-selected limits, hidden tail oracles, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier window seal factorization`

Proof. Begin with the finite window. Apply Definition 681.4; every completion-facing read of the unary precision row E reaches the displayed $\mathsf{StreamName}^{\uparrow}$ window row W before any seal consumer can read a later row.

Next use Definition 681.5. The finite window supplies the displayed dyadic ledger row D, so the route cannot branch into a hidden tail oracle or a host-selected cofinal schedule.

Now apply Definition 681.6. The $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row L are read through the same finite route

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L.$$

The displayed hsame transports keep the structural rows H, C, P, N attached to that route.

Determinacy follows from Definition 681.24: after E, W, D are fixed, the completion-facing consumer has the same R and L rows under the same transport, continuation, provenance, and local naming package.

Finally invoke Definition 681.23. The RealCompletionSelectorSeal route consumes the displayed readback and seal rows, so no quotient stream equality, host-selected limit, hidden tail oracle, or ambient completeness coordinate can be introduced downstream. □

Theorem 681.26 (Tail cofinality schedule Phase Real and Completion exit handoff)**.** *Let*

$$T = (E, W, D, R, L, H, C, P, N)$$

be an accepted $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *packet. If* T *is consumed as a Phase Real and Completion exit route over the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surfaces, then the exit factors through the displayed finite path*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto L,$$

where W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* L *is the* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient stream, selected host limit, host equality, choice schedule, hidden tail oracle, ambient completeness, or second selector-budget source.*

Lean4 ✓ `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier phase real completion exit handoff`

Proof. Begin with the local schedule route. The diagonal handoff, finite-window coverage, threshold sufficiency, Real seal readback, and scoped binding theorems force every accepted read of T to pass from the unary request E through the finite window W, dyadic ledger D, regular-rational readback R, and Real seal L, with H, C, P, N only replaying transport, continuation, provenance, and local naming.

Next read the Completion-facing side. The RegularCauchyCompletion route places the completion carrier on the same finite path, while shared-tail budget factorization identifies the shared

threshold supplied by $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and the agreement rows supplied by $\mathsf{RealTailAgreementSeal}^{\uparrow}$ as consumers of that path rather than as independent sources.

Now apply the selector-budget lock. It prevents the Real or Completion consumer from replacing the dyadic ledger by a separate selector budget, and it keeps the $\mathsf{Real}^{\uparrow}$ seal downstream from the same E, W, D, R, L route.

Finally use window seal factorization. Once the finite window and dyadic ledger have been read, the completion-facing seal is the displayed R, L readback under the same structural rows. Since the carrier inventory contains no coordinate for quotient streams, selected host limits, host equality, choice schedules, hidden tail oracles, ambient completeness, or a second selector-budget source, the Phase Real and Completion exit has no public route outside the stated finite path. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{TailCofinalitySchedule}^{\uparrow}$ packet is generated from a unary precision request, cofinal finite-window schedule, dyadic tolerance ledger, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 681.1, Definition 681.2, Definition 681.3, Definition 681.4, Definition 681.5, Definition 681.6, Definition 681.7, Definition 681.8, and Definition 681.24 bind the public finite-window schedule to the downstream RealUp and RegSeqRat handoff through the displayed BEDC primitives, local NameCert row, RegularCauchyTailMeet threshold route, and deterministic StreamName window readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailCofinalitySchedule}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.TailCofinalityScheduleUp.TailCofinalityScheduleCarrier\_phase\_real\_completion\_exit\_`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected host limits, ambient completeness, hidden tail oracles, unbounded schedules, or a public standard bridge.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 682

A Concrete Naming Certificate for $\mathsf{InscriptionAuditTrace}^{\uparrow}$

The $\mathsf{InscriptionAuditTrace}^{\uparrow}$ surface realizes the audit trace of an inscription event as a finite NameCert packet. It combines the inscription-event source and named-record rows, the GAP-ledger residue rows, and the accepted-check consumer-budget rows into one public audit object. The packet records only the rows that a certificate, citation, theorem site, or ChapterTasteGate consumer may inspect.

The audit trace consumes chapter 519 for the source, name, route, and check rows, chapter 520 for the residue row, and chapter 651 for the accepted-check and downstream consumer-budget surface before it exposes its local public trace. The packet $S, M, R, K, U, Q, H, C, P, N$ is the finite join of those sibling surfaces: S, M, R, K, U carry the event route and consumer boundary, Q carries the GAP residue, and H, C, P, N carry the transports, displayed continuations, provenance, and local certificate row. These sibling packets are the horizontal lattice route used by the carrier below, and no private agency predicate or host trace is exported by this join.

Definition 682.1 (Inscription audit trace carrier)**.** *An* $\mathsf{InscriptionAuditTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, M, R, K, U, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-state row at which the inscription route begins;*
2. *M is the named record, theorem site, citation row, or public interface admitted by the route;*
3. *R is the displayed route witness from S to M;*
4. *K is the accepted-check row that admits R as public record data for M;*
5. *U is the downstream audit consumer row, naming the certificate, citation, theorem site, or* ChapterTasteGate *surface that may read the trace;*
6. *Q is the GAP residue row left after the accepted route and consumer boundary are exposed;*
7. *H records componentwise* hsame *transports for S, M, R, K, U, Q together with package and local naming rows;*
8. *C records the displayed* Cont *routes by which consumers read the source, name, route, accepted check, consumer, and residue rows;*

9. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* InscriptionEvent$^{\uparrow}$*,* InscriptionGapLedger$^{\uparrow}$*,* InscriptionAcceptanceBudget$^{\uparrow}$*, and* ChapterTasteGate *surfaces;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}$ *row.*

The carrier contains no private agency predicate, host process trace, unrecorded mutation, unchecked route, unbudgeted consumer right, or completed ontology of inscription beyond the displayed audit rows.

Theorem 682.2 (Inscription audit trace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}$ *surface for Definition 682.1 consists of six finite obligations:*

1. *carrier exposure for $A = (S, M, R, K, U, Q, H, C, P, N)$;*
2. *source and named-record locality inherited from the* InscriptionEvent$^{\uparrow}$ *rows S, M, R, K, U;*
3. *accepted-check exactness inherited from the* InscriptionAcceptanceBudget$^{\uparrow}$ *row K;*
4. *GAP-residue exactness inherited from the* InscriptionGapLedger$^{\uparrow}$ *row Q;*
5. *consumer accountability through U and the displayed* Cont *routes in C;*
6. *package and local naming stability through H, P, N over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and* ChapterTasteGate *surfaces.*

Proof. Carrier exposure is projection from the displayed packet. The rows $S, M, R, K, U, Q, H, C, P, N$ are the complete inventory available to the audit trace.

The inscription-event rows supply the source, name, route, accepted-check, and consumer boundary. The acceptance-budget rows identify K as the public check row and keep downstream use inside the consumer surface named by U.

The GAP-ledger rows identify Q as the residue left at the accepted route boundary. The transport, continuation, package, and local naming rows then record how a ChapterTasteGate or certificate consumer reads the same finite packet without importing private agency or host process data. □

Theorem 682.3 (Inscription audit trace route-check residue exhaustion)**.** *For every accepted* InscriptionAuditTrace$^{\uparrow}$ *carrier $A = (S, M, R, K, U, Q, H, C, P, N)$, the route-check-residue surface is exhausted by the rows $S, M, R, K, Q, H, C, P, N$. A public audit read first reaches the named record M from the source row S through the route witness R, then admits the route through the check row K, and then records the remaining GAP residue through Q.*

No audit read may replace these rows by a host checker, hidden proof replay, private agency predicate, raw byte-change identity, or unrecorded route.

Proof. Project the carrier and isolate the route-facing rows S, M, R, K, Q. These are the only coordinates that connect source state, named record, accepted check, and residue.

Apply the inscription-event surface for the route from S through R to M. Apply the acceptance-budget surface to read K as the accepted check, and apply the GAP-ledger surface to read Q as the displayed residue.

The remaining rows H, C, P, N preserve this same route under componentwise hsame transport, displayed Cont replay, package provenance, and local naming. Since the carrier has no host checker, hidden replay, agency predicate, byte-change classifier, or unrecorded route coordinate, the route-check-residue surface is exhausted by the displayed packet. □

Theorem 682.4 (Inscription audit trace consumer non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{InscriptionAuditTrace}^{\uparrow}$ *carrier factors through the finite rows*

$$S, M, R, K, U, Q, H, C, P, N.$$

The consumer may read the source state, named record, route witness, accepted check, intended consumer, GAP residue, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It cannot read private motives, host process traces, unchecked route data, unbudgeted consumer authority, or inscription data outside the finite audit trace.*

Proof. Begin with the consumer row U and the continuation row C. Together they name the public audit reader and the displayed route by which that reader enters the packet.

Use Definition 682.2. The local NameCert surface binds the consumer to the inscription-event, GAP-ledger, and acceptance-budget rows visible in the same packet.

Use Definition 682.3. The consumer's route-check-residue read is already exhausted by $S, M, R, K, Q, H, C, P, N$ No additional carrier coordinate remains, so the consumer cannot escape to private motives, host traces, unchecked routes, unbudgeted authority, or outside inscription data. □

Theorem 682.5 (Inscription audit trace TasteGate visible surface)**.** *The* ChapterTasteGate*-visible surface of an accepted* $\mathsf{InscriptionAuditTrace}^{\uparrow}$ *carrier is exactly the finite audit packet* $A = (S, M, R, K, U, Q, H, C, P,$ *together with its local* $\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}$ *obligations. A gate may inspect the source, named record, route witness, accepted check, consumer row, GAP residue, transports, continuations, package provenance, and local naming row; it receives no hidden approval channel, private agency predicate, host trace, or row not named by the packet.*

Lean4 ✓ `BEDC.Derived.InscriptionAuditTraceUp.InscriptionAuditTraceTasteGate single carrier alignment`

Proof. The gate-facing input is the same carrier from Definition 682.1. The carrier lists the rows a ChapterTasteGate consumer may inspect and supplies no separate admission channel.

Apply Definition 682.2 to expose the local NameCert surface. It ties the inscription-event, GAP-ledger, and acceptance-budget readings to one finite audit packet.

Apply Definition 682.4. Since every downstream consumer, including a ChapterTasteGate consumer, factors through $S, M, R, K, U, Q, H, C, P, N$, the visible gate surface is exactly the displayed audit trace. □

Theorem 682.6 (Inscription audit trace budget-ledger sibling route)**.** *Every accepted* $\mathsf{InscriptionAuditTrace}^{\uparrow}$ *carrier first reads the inscription-event source and route rows through chapter 519, then reads the GAP residue row through chapter 520, then reads the accepted-check budget row through chapter 651, and only then exposes the* ChapterTasteGate*-visible row of Definition 682.5. The visible gate surface is therefore bound to the source, ledger, and budget siblings before any audit consumer may inspect it.*

Proof. Project the accepted carrier $A = (S, M, R, K, U, Q, H, C, P, N)$ from Definition 682.1. The rows S, M, R are the source, named-record, and route rows inherited from the inscription-event surface, and U names the public audit consumer that eventually reaches the gate-facing read.

The row Q is the GAP residue coordinate, so the route must pass through the sibling ledger surface before the audit trace can describe what remains at the accepted route boundary. The row K is the accepted-check budget coordinate, so the same packet must also pass through the acceptance-budget sibling before exposing a consumer right.

The transport, continuation, provenance, and naming rows H, C, P, N preserve this same route and do not add another inspection coordinate. Because the carrier has no hidden approval channel, private agency predicate, host trace, unchecked route, or unbudgeted consumer row, the ChapterTasteGate surface is reached only after the inscription-event, GAP-ledger, and acceptance-budget rows have been read. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionAuditTrace}^{\uparrow}$ packet is generated from a source state, named record, route witness, accepted check, downstream audit consumer, GAP residue, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.InscriptionAuditTraceUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 682.1, Definition 682.2, Definition 682.3, Definition 682.4, Definition 682.5, and Definition 682.6 bind the inscription audit trace as a finite bridged NameCert packet over the public audit-trace certificate surface, with inscription-event, GAP-ledger, acceptance-budget, and TasteGate-visible rows preserved.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionAuditTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.InscriptionAuditTraceUp.InscriptionAuditTraceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, private agency, host process traces, raw byte-change identity, unchecked route semantics, unbudgeted downstream consumption, or completed inscription ontology.

Upgrade path: Mature closure requires bridge checking for the audit-trace certificate surface while preserving the carrier, route-check residue, consumer non-escape, sibling route, and TasteGate-visible packet.

Chapter 683

A Concrete Naming Certificate for $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$

The $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ packet joins the finite small-step ledger of $\mathsf{ContinuationTrace}^{\uparrow}$ with the terminal-read surface of $\mathsf{ContinuationBigStep}^{\uparrow}$ and the typed normal-form boundary of $\mathsf{TypedReductionNormalForm}^{\uparrow}$. The terminal-read coordinate is consumed from chapter 657 before the typed normal-form row is read. It records trace replay, terminal stability, typed normal-form readback, transports, routes, provenance, and local naming in one finite BHist carrier. It does not appeal to a host evaluator, a host conversion check, quotient behaviour, or a normalization oracle outside the displayed BEDC rows.

Definition 683.1 (ContinuationTraceNormalForm carrier)**.** *A* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, T, R, Q, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the source history row from which replay begins;*
2. *T is the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *small-step ledger, including its displayed source, mark-and-tail, target, and trace rows;*
3. *R is the terminal history row reached by replaying T through the displayed* Cont *routes;*
4. *Q is the terminal-read witness exposed by the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *surface;*
5. *V is the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *row naming the normal-form classifier for the same terminal history;*
6. *H stores componentwise* hsame *transports for S, T, R, Q, V, provenance, and the local name;*
7. *C stores the displayed* Cont *routes that replay the trace ledger, reach the terminal row, consume the terminal-read witness, and expose the typed normal-form row;*
8. *P is the* Pkg *provenance over* BHist, BMark, $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{ContinuationBigStep}^{\uparrow}$, $\mathsf{TypedReductionNormalFo}$ hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no host conversion relation, no quotient-normalization proof, no evaluator graph, no global confluence theorem, and no external normalizer.

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormTasteGate single carrier alignment`

Theorem 683.2 (ContinuationTraceNormalForm NameCert obligations)**.** *For the carrier of Definition 683.1, the local* $\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Z = (S, T, R, Q, V, H, C, P, N),$$

trace replay through the $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, terminal-read exposure through the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *row, typed normal-form readback through* V*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and non-escape from host conversion, quotient behaviour, evaluator graphs, global confluence, and normalization oracles.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 683.1. The packet exposes exactly $S, T, R, Q, V, H, C, P, N$.

Trace replay is read from T and C. The $\mathsf{ContinuationTrace}^{\uparrow}$ row supplies finite small-step entries, and the displayed continuation route replays those entries from the source row S to the terminal row R.

Terminal exposure is then read through Q, and typed normal-form readback is read through V. The transport row H moves the same displayed rows componentwise, while P and N record package provenance and local naming.

The non-escape clauses follow from the carrier inventory. Since the packet has no coordinate for host conversion, quotient behaviour, an evaluator graph, global confluence, or an external normalizer, no local certificate projection can export one. □

Theorem 683.3 (ContinuationTraceNormalForm source normalization obligation)**.** *For every displayed packet*

$$Z = (S, T, R, Q, V, H, C, P, N)$$

accepted by $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$*, the source-to-trace normalization obligation is exactly the replay from* S *through the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *row* T*, the displayed* BHist *and* BMark *data inside that row, the continuation routes in* C*, and the local* NameCert *row* N*. It contains no host conversion relation, quotient normalization datum, evaluator graph, global confluence theorem, or external normalizer.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier source normalization obligation`

Proof. Project Z by Definition 683.1. The source row S, trace row T, route row C, and local name N are the only source-to-trace replay coordinates in the accepted carrier.

Apply Definition 683.2. It certifies trace replay through the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger and the displayed continuation routes before any terminal-read or typed-normal row is consumed.

The excluded host conversion, quotient normalization, evaluator graph, global confluence, and external normalizer objects have no coordinate in Z. Hence the source normalization obligation is precisely the displayed BHist, BMark, Cont, and NameCert route. □

Theorem 683.4 (ContinuationTraceNormalForm dependency readback obligation)**.** *For every displayed packet*

$$Z = (S, T, R, Q, V, H, C, P, N)$$

accepted by ContinuationTraceNormalForm$^{\uparrow}$*, dependency readback passes through* ContinuationTrace$^{\uparrow}$*,* ContinuationBigStep$^{\uparrow}$*, and* TypedReductionNormalForm$^{\uparrow}$ *in that order. The rows* T, R, Q, V, H, C, P, N *expose the trace ledger, terminal row, terminal-read witness, typed normal-form row, componentwise transport, displayed route, provenance, and local name without adding host conversion, quotient normalization, evaluator graphs, global confluence, or external normalizers.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier dependency readback obligation`

Proof. Start from the carrier definition. It places the trace dependency in T, the terminal dependency in R, Q, and the typed normal-form dependency in V, all inside the same finite packet.

Use Definition 683.2. The local certificate reads trace replay, terminal-read exposure, and typed normal-form readback through the displayed rows and records their transports, routes, package row, and local name in H, C, P, N.

Since the packet contains no off-route evaluator, quotient normalizer, host conversion relation, global confluence theorem, or external normalizer, dependency readback cannot leave the ContinuationTrace$^{\uparrow}$, ContinuationBigStep$^{\uparrow}$, and TypedReductionNormalForm$^{\uparrow}$ route. □

Theorem 683.5 (ContinuationTraceNormalForm typed replay obligation)**.** *For every displayed packet*

$$Z = (S, T, R, Q, V, H, C, P, N)$$

accepted by ContinuationTraceNormalForm$^{\uparrow}$*, typed replay reaches the normal-form row* V *only after the source row* S*, trace ledger* T*, terminal row* R*, terminal-read witness* Q*, transport row* H*, continuation route row* C*, package row* P*, and local name* N *have been consumed. No host conversion, quotient normalization, evaluator graph, global confluence theorem, or external normalizer can replace that ordered replay.*

Proof. Read V from Definition 683.1. It is not an isolated classifier row; it is attached to the source, trace, terminal, terminal-read, transport, route, provenance, and naming rows of the same accepted packet.

Apply Definition 683.2. The typed normal-form row is certified only after trace replay and terminal-read exposure have passed through the displayed Cont routes and the componentwise transport row.

The carrier has no coordinate for host conversion, quotient normalization, evaluator graphs, global confluence, or external normalizers. Thus typed replay reaches V only through S, T, R, Q, H, C, P, N. □

Theorem 683.6 (ContinuationTraceNormalForm non-escape)**.** *Every accepted* ContinuationTraceNormalForm$^{\uparrow}$ *packet exposes terminal trace normalization only through the displayed rows*

$$S, T, R, Q, V, H, C, P, N.$$

A downstream consumer may read the source history, finite continuation trace, terminal row, terminal-read witness, typed normal-form row, componentwise transports, displayed continuation replay, package provenance, and local naming row. It cannot replace those rows by host conversion, quotient behaviour, a selected evaluator, a global confluence theorem, or a normalization oracle.

Proof. Begin with the carrier projection. The only replay data are the finite trace row T and the displayed continuation routes in C.

Use Definition 683.2. The terminal row R, terminal-read witness Q, and normal-form row V are reached only after the replayed trace has been consumed inside the same packet.

Finally inspect the excluded objects. Each would require a separate carrier coordinate or a second consumer route outside $S, T, R, Q, V, H, C, P, N$. Since the accepted packet has neither, the terminal normal-form surface is exactly the displayed finite NameCert packet. □

Theorem 683.7 (ContinuationTraceNormalForm terminal scope)**.** *Every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet exposes the terminal row only after consuming the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row, and the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *row through the displayed* Cont *routes. No consumer can replace that sequence by host conversion, quotient normalization, evaluator graphs, or normalization oracles.*

Proof. Project the accepted carrier $Z = (S, T, R, Q, V, H, C, P, N)$. The terminal row R is a coordinate of the same finite packet as the trace ledger T, terminal-read row Q, typed normal-form row V, and displayed continuation routes C.

Apply Definition 683.2. Trace replay is first consumed through T and C, terminal-read exposure is then consumed through Q, and typed normal-form readback is consumed through V before the local name N is reached.

Apply Definition 683.6. Host conversion, quotient normalization, evaluator graphs, and normalization oracles have no coordinate in Z, so none can serve as an alternate terminal-read route. □

Theorem 683.8 (ContinuationTraceNormalForm trace ledger exhaustion)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the finite row* T *exhausts all trace reads exported by the packet. A consumer may replay only the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger through* C *and may transport it only through* H*; it cannot read an additional trace from a host evaluator, quotient normalization, global confluence, or an external normalizer.*

Proof. Project the carrier in Definition 683.1. The only trace coordinate is T, and the only replay coordinate is the displayed continuation route row C.

Apply Definition 683.2. Trace replay is certified through the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, componentwise transport is certified through H, and the package and local name are certified through P, N for the same finite tuple.

Use Definition 683.6. Since host conversion, quotient behaviour, evaluator graphs, global confluence, and normalization oracles have no coordinate in Z, none can provide a second trace read outside T.

Finally apply Definition 683.7. Terminal exposure consumes the displayed trace before reaching R, Q, V, so every exported trace read is exhausted by T. □

Theorem 683.9 (ContinuationTraceNormalForm terminal readback lock)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the terminal row* R *is locked behind the finite trace row* T*, terminal-read row* Q*, transport row* H*, displayed continuation row* C*, provenance row* P*, and local naming row* N*. No public read may expose* R *without first consuming those displayed rows.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier terminal readback lock`

Proof. Start with the carrier projection of Definition 683.1. The row R is not an isolated terminal object; it is part of the finite packet $S, T, R, Q, V, H, C, P, N$.

Apply Definition 683.2. The terminal-read exposure through Q is certified only after trace replay through T and C, componentwise transport through H, and package/name recording through P, N.

Use Definition 683.7. It places the terminal row behind the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, the $\mathsf{ContinuationBigStep}^{\uparrow}$ terminal-read witness, and the $\mathsf{TypedReductionNormalForm}^{\uparrow}$ row.

Finally use Definition 683.6. A host conversion relation, quotient normalizer, evaluator graph, global confluence theorem, or external normalizer cannot unlock a terminal readback because none is a row of the accepted packet. □

Theorem 683.10 (ContinuationTraceNormalForm typed boundary nonescape)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the typed normal-form row* V *cannot be replaced by host conversion, quotient normalization, evaluator graphs, or global confluence. The typed boundary is read only through the displayed trace, terminal, transport, route, package, and local naming rows of* Z*.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormTypedBoundaryNonescape`

Proof. Read V from Definition 683.1. It is the finite $\mathsf{TypedReductionNormalForm}^{\uparrow}$ row of the same packet as S, T, R, Q, H, C, P, N.

Apply Definition 683.2. Typed normal-form readback is certified through V after trace replay, terminal-read exposure, componentwise transport, displayed continuation replay, provenance, and local naming have been fixed.

Use Definition 683.7. The typed normal-form row is reached through the terminal route rather than through a host conversion check or evaluator graph.

Finally apply Definition 683.6. The packet has no coordinate for quotient normalization, global confluence, evaluator graphs, or external normalizers, so none can replace V as the typed boundary. □

Theorem 683.11 (ContinuationTraceNormalForm terminal route determinacy)**.** *Let* $Z = (S, T, R, Q, V, H, C, P, N$ *and* $Z' = (S', T', R', Q', V', H', C', P', N')$ *be accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packets. If two terminal reads have the same trace row, typed boundary row, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row, then they determine the same normal-form consumer surface. No off-route evaluator state, host conversion relation, quotient-normalization datum, global confluence theorem, or external normalization oracle can distinguish the two reads.*

Proof. First apply terminal scope from Definition 683.7. Each terminal read consumes the $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, the $\mathsf{ContinuationBigStep}^{\uparrow}$ terminal-read row, and the $\mathsf{TypedReductionNormalForm}^{\uparrow}$ row through the displayed route before reaching a consumer surface.

Next use trace ledger exhaustion from Definition 683.8. Since the trace rows agree and every trace read is exhausted by that finite ledger, neither packet can supply an additional trace from a host evaluator or quotient normalizer.

Apply the terminal readback lock of Definition 683.9. The common transport, route, provenance, and naming rows place the terminal row behind the same T, Q, H, C, P, N surface in both packets.

Finally use typed boundary nonescape from Definition 683.10. The common typed row V is the only normal-form boundary available to the consumer, and the excluded off-route evaluator state, host conversion relation, quotient-normalization datum, global confluence theorem, and external normalization oracle have no coordinate in either accepted packet. □

This terminal-scope row is the paper obligation needed before Lean can formalize the continuation trace normal-form handoff.

Theorem 683.12 (ContinuationTraceNormalForm standard bridge surface)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the public terminal trace-normal-form packet bridges exactly from* $\mathsf{ContinuationTrace}^{\uparrow}$*,* $\mathsf{ContinuationBigStep}^{\uparrow}$*, and* $\mathsf{TypedReductionNormalForm}$ *through the displayed source, trace, terminal, terminal-read, typed-boundary, transport, route, provenance, and naming rows* $S, T, R, Q, V, H, C, P, N$*.*

The bridge surface reads no host conversion relation, quotient normalization datum, evaluator graph, global confluence theorem, or external normalizer. Its entire public object is the finite BHist *route whose trace replay, terminal readback, typed boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and* $\mathsf{NameCert}$ *row are already displayed in the carrier.*

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormUp StdBridge`

Proof. First apply Definition 683.2. The local certificate surface is exactly the carrier admission, trace replay, terminal-read exposure, typed normal-form readback, componentwise transport, displayed continuation route, package provenance, and local naming row.

Next use Definition 683.7. It orders the public route through $\mathsf{ContinuationTrace}^{\uparrow}$, then $\mathsf{ContinuationBigStep}^{\uparrow}$, then $\mathsf{TypedReductionNormalForm}^{\uparrow}$, all within the same displayed packet.

Apply Definition 683.11. A consumer that reads the same trace row, typed boundary row, transport row, route row, provenance row, and local naming row obtains the same normal-form consumer surface.

Finally use Definition 683.10. The excluded host conversion, quotient normalization, evaluator graph, global confluence, and external normalizer have no coordinate in Z, so the standard bridge surface is precisely the displayed finite route. □

Theorem 683.13 (ContinuationTraceNormalForm replay-prefix determinacy)**.** *Let* $Z = (S, T, R, Q, V, H, C, P, N)$ *and* $Z' = (S', T', R', Q', V', H', C', P', N')$ *be accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packets. Agreement on the source row, the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, and the displayed replay continuation determines the same terminal replay prefix before any terminal-read row or typed-boundary row is exposed.*

Thus no host conversion relation, quotient normalization datum, evaluator graph, global confluence theorem, or external normalizer can split two replay prefixes whose visible source, trace, and continuation rows agree.

Proof. Project both packets through Definition 683.1. The only replay-prefix coordinates before terminal exposure are the source row S, the trace ledger T, and the displayed continuation route C.

Apply Definition 683.7. It places the terminal-read and typed-boundary rows strictly after the trace-ledger replay, so the prefix is determined before Q or V can be read.

Use Definition 683.8. Since the trace ledger is exhausted by the displayed T row and replayed through C, an agreeing source, ledger, and replay route leave no second prefix source. □

Theorem 683.14 (ContinuationTraceNormalForm terminal typed-boundary exhaustion)**.** *Every terminal typed-boundary consumer of an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$ *factors through the source row, trace ledger, terminal row, terminal-read row, typed normal-form boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row* $S, T, R, Q, V, H, C, P, N$*.*

The consumer cannot read host conversion, quotient normalization, evaluator graphs, global confluence, or an external normalizer.

Proof. First apply Definition 683.7. It orders the terminal typed-boundary read through $S, T, R, Q, V, H, C, P, N$.

Next use Definition 683.9. The terminal row is locked behind the finite trace ledger, terminal-read row, transport, continuation, package, and naming rows.

Finally combine Definition 683.10 with Definition 683.12. The typed boundary and bridge surface expose exactly the displayed finite route, while the excluded host conversion, quotient normalization, evaluator graph, global confluence, and external normalizer have no carrier coordinate. □

Theorem 683.15 (ContinuationTraceNormalForm root unblock package)**.** *The carrier of Definition 683.1, replay-prefix determinacy, terminal typed-boundary exhaustion, trace ledger exhaustion, terminal readback lock, and standard bridge surface form the root package consumed by downstream halting and normal-form chapters. The package consists only of the rows $S, T, R, Q, V, H, C, P, N$ and their displayed* BHist*,* BMark*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure.*

It supplies no host conversion relation, quotient normalization datum, evaluator graph, global confluence theorem, external normalizer, or off-route terminal reader.

Proof. Start from the carrier definition and Definition 683.2. They identify the displayed source, trace, terminal, terminal-read, typed-boundary, transport, route, package, and naming rows of the local packet.

Use Definition 683.13 for the prefix part of the package. The source, trace ledger, and replay continuation determine the terminal replay prefix before terminal-read exposure.

Use Definition 683.14, Definition 683.8, and Definition 683.9 for the terminal part. These clauses exhaust the trace ledger, lock terminal readback, and force typed-boundary consumers through the displayed route.

Finally apply Definition 683.12. It packages the same finite surface for downstream halting and normal-form chapters, with no coordinate for the excluded host, quotient, evaluator, confluence, normalizer, or off-route terminal-reader objects. □

Theorem 683.16 (ContinuationTraceNormalForm root terminal-source order)**.** *Every root consumer of an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$ must read the source row S, the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger T, the terminal row R, the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row Q, and the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *boundary row V in that source, trace, terminal, terminal-read, typed-boundary order before any normal-form consumer surface is exposed. The rows H, C, P, N then carry the componentwise transport, displayed route, provenance, and local name for the same ordered read.*

Proof. Start from the carrier and NameCert obligations. They expose the finite tuple $S, T, R, Q, V, H, C, P, N$ and attach each public read to carrier admission, trace replay, terminal-read exposure, typed normal-form readback, transport, route, provenance, and naming.

Use terminal scope and terminal readback lock. The terminal row is not public before the trace ledger and terminal-read witness have been consumed through the displayed route, and the lock places R behind T, Q, H, C, P, N.

Finally apply typed boundary nonescape and the standard bridge surface. The normal-form boundary V is read only after the terminal route has been fixed, and the bridge exposes no host conversion, quotient normalization, evaluator graph, global confluence, external normalizer, or off-route terminal reader. □

Theorem 683.17 (ContinuationTraceNormalForm root typed-boundary route)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the typed-boundary row* V *is reachable only through the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *row* T*, the displayed* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read rows* R, Q*, and the displayed* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *row* V*, with transport, route, provenance, and naming supplied by* H, C, P, N*. No consumer can reach* V *through host conversion, quotient normalization, evaluator correctness, global confluence, or an external normalizer.*

Proof. Project the accepted packet. The only trace row is T, the only terminal-read rows are R, Q, and the only typed normal-form row is V.

Apply trace ledger exhaustion. It prevents a second trace source from entering before the terminal-read rows are consumed.

Apply terminal readback lock and typed boundary nonescape. The lock routes terminal readback through T, Q, H, C, P, N, and nonescape keeps the typed boundary inside the displayed trace, terminal, transport, route, package, and local naming rows. □

Theorem 683.18 (ContinuationTraceNormalForm root downstream unblock ledger)**.** *The root terminal-source order, the root typed-boundary route, trace ledger exhaustion, and terminal readback lock assemble the downstream unblock ledger for halting and normal-form chapters. The ledger surface is exactly* $S, T, R, Q, V, H, C, P, N$ *with its displayed* BHist*,* BMark*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it exports no host conversion, quotient normalization, evaluator graph, global confluence theorem, external normalizer, or off-route terminal reader.*

Proof. Use Definition 683.16 to order the root read. A downstream consumer first sees the source, trace, terminal, terminal-read, and typed-boundary rows before it can consume the normal-form surface.

Use Definition 683.17 to identify the only route into the typed boundary. The route passes through $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{ContinuationBigStep}^{\uparrow}$, and $\mathsf{TypedReductionNormalForm}^{\uparrow}$ as displayed rows of the same packet.

Finally combine Definition 683.8 with Definition 683.9. The trace row is exhausted by T, the terminal row is locked behind T, Q, H, C, P, N, and the excluded host, quotient, evaluator, confluence, normalizer, and off-route terminal-reader objects have no coordinate in the ledger. □

Theorem 683.19 (ContinuationTraceNormalForm source-endpoint partition)**.** *Every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet partitions its public rows into source trace, terminal-read, typed-normal boundary, transport, route, provenance, and local* NameCert *data before any consumer reads the endpoint. For* $Z = (S, T, R, Q, V, H, C, P, N)$*, the source trace is* S, T*, the terminal-read surface is* R, Q*, the typed-normal boundary is* V*, the transport surface is* H*, the route surface is* C*, the provenance row is* P*, and the local naming row is* N*.*

Proof. Start from the carrier definition and Definition 683.2. They expose exactly the rows $S, T, R, Q, V, H, C, P, N$ and bind each row to carrier admission, trace replay, terminal-read exposure, typed normal-form readback, transport, route, provenance, and local naming.

Use Definition 683.7. It orders endpoint access after the trace ledger, terminal-read row, and typed normal-form row have been consumed through the displayed route.

Finally apply Definition 683.8 and Definition 683.10. The trace and typed-boundary reads cannot be replaced by host conversion, quotient normalization, evaluator graphs, global confluence, or external normalizers, so the displayed partition is the entire endpoint surface. □

Theorem 683.20 (ContinuationTraceNormalForm terminal-result determinacy)**.** *Two public terminal reads through the same displayed T, Q, H, C, P, N rows of accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packets have the same BEDC result row under* hsame *transport and* Cont *replay. No host evaluator, quotient normalizer, global confluence proof, or off-route terminal reader can separate the two reads.*

Proof. Begin with Definition 683.9. It locks each terminal row behind the trace ledger, terminal-read row, transport, route, provenance, and local naming rows.

Apply Definition 683.11. Agreement on the displayed trace, typed boundary, componentwise transport, route, provenance, and local naming rows determines the same normal-form consumer surface.

Use Definition 683.8. Since trace reads are exhausted by the displayed ledger and replayed only through C, no second terminal result can be read through an external trace source.

Finally apply Definition 683.10. The excluded host conversion, quotient normalization, evaluator graph, global confluence, and external normalizer have no row from which to distinguish the BEDC result. □

Lean4 ✓ `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier terminal result determinacy`

Theorem 683.21 (ContinuationTraceNormalForm typed-boundary ledger nonescape)**.** *Every typed-normal boundary read in an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet factors through the displayed source trace, terminal-read, typed boundary, transport, route, provenance, and local naming ledger $S, T, R, Q, V, H, C, P, N$. The boundary cannot import host conversion, quotient normalization, global confluence, evaluator graphs, or an external normalizer.*

Proof. Project the packet through Definition 683.1. The typed-normal boundary is the row V in the same finite carrier as the source, trace, terminal, terminal-read, transport, route, provenance, and local name.

Apply Definition 683.10. It reads V only through the displayed trace, terminal, transport, route, package, and local naming rows.

Combine Definition 683.2, Definition 683.7, and Definition 683.8. These rows certify the carrier surface, order terminal access, and exhaust trace reads before the typed boundary reaches a consumer. □

Theorem 683.22 (ContinuationTraceNormalForm route ledger totality)**.** *For every accepted* ContinuationTraceNor *packet $Z = (S, T, R, Q, V, H, C, P, N)$, the route ledger is total over the displayed source, trace, terminal, terminal-read, typed-boundary, transport, route, provenance, and naming rows. Every public normal-form consumer must read S, then the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger T, then the terminal row R, then the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row Q, then the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *row V, with componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local* $\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}$ *naming through N.*

Proof. Begin with Definition 683.7. It orders terminal exposure through the finite trace ledger, terminal-read row, and typed normal-form row inside the displayed packet.

Next use Definition 683.8. The trace coordinate is exhausted by T and replayed only through C, so no public consumer can supply a second trace ledger.

Apply Definition 683.9. The terminal row R is locked behind T, Q, H, C, P, N, and therefore the route ledger reaches the terminal read only through the displayed rows.

Finally use Definition 683.11 and Definition 683.12. Determinacy prevents off-route data from separating two reads with the same ledger rows, while the bridge surface exports exactly the public continuation trace normal-form route. □

Theorem 683.23 (ContinuationTraceNormalForm normal boundary exhaustion)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the typed normal-form boundary is exhausted by the* $\mathsf{ContinuationTrace}^{\uparrow}$*,* $\mathsf{ContinuationBigStep}^{\uparrow}$*, and* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *rows of the same packet. The boundary exports the finite route* $S, T, R, Q, V, H, C, P, N$ *and does not export host conversion, quotient normalization, evaluator correctness, global confluence, or an external normalizer.*

Proof. First apply Definition 683.22. It makes the public route ledger total over $S, T, R, Q, V, H, C, P, N$.

Use Definition 683.10. The typed boundary is read through the displayed trace, terminal, transport, route, package, and local naming rows, not through host conversion or quotient normalization.

Next combine Definition 683.9 with Definition 683.12. The terminal-read row is locked behind the finite ledger, and the public bridge exposes exactly the continuation trace, big-step terminal read, and typed normal-form rows.

The remaining exclusions follow from Definition 683.6. Evaluator correctness, global confluence, quotient behaviour, host conversion, and an external normalizer have no coordinate in the accepted packet. □

Theorem 683.24 (ContinuationTraceNormalForm source route obligation package)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the source, terminal, transport, route, provenance, and* $\mathsf{NameCert}$ *obligations are packaged by the finite route ledger. The source obligation is the displayed source row* S *and trace ledger* T*; the terminal obligation is the displayed terminal row* R *and terminal-read row* Q*; the transport obligation is* H*; the route obligation is* C*; the provenance obligation is* P*; and the naming obligation is* N*. These obligations are consumed before the typed normal-form boundary* V *is available to a public consumer.*

Proof. Start with Definition 683.2. It identifies carrier admission, trace replay, terminal-read exposure, typed normal-form readback, transport, route, provenance, and local naming as the local certificate surface.

Apply Definition 683.22. The route ledger is total over $S, T, R, Q, V, H, C, P, N$, so each obligation is attached to a displayed coordinate of the accepted packet.

Use Definition 683.23. The typed normal-form boundary is exhausted by the continuation trace, big-step terminal read, and typed normal-form rows of the same finite route.

Finally combine Definition 683.11 with Definition 683.12. The same source, terminal, transport, route, provenance, and naming rows determine the public normal-form consumer surface, so the obligation package has no hidden host conversion, quotient-normalization, evaluator, or oracle component. □

Theorem 683.25 (ContinuationTraceNormalForm visible source route)**.** *For every accepted* ContinuationTraceNorr *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the public source route exposes the source history row* S*, the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger* T*, the terminal row* R*, the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row* Q*, the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *boundary row* V*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}$ *row* N *before any terminal consumer reads the normal-form endpoint.*

No terminal consumer can read the endpoint by skipping the visible source route, adding an evaluator coordinate, importing host conversion, quotient trace equality, global confluence, or external normalization.

Proof. Project the accepted carrier from Definition 683.1. Its visible rows are exactly $S, T, R, Q, V, H, C, P, N$.

Apply Definition 683.6 and Definition 683.7. They place terminal exposure behind the displayed trace ledger, terminal-read row, typed normal-form row, transports, routes, provenance, and local name.

Use Definition 683.8 and Definition 683.11. The finite trace ledger exhausts exported trace reads, and the displayed route determines the same public normal-form consumer surface whenever its route rows agree. □

Theorem 683.26 (ContinuationTraceNormalForm typed-reduction handoff)**.** *Every typed-reduction or normal-form consumer of an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$ *can read the terminal row only through* T, R, Q, V, H, C, P, N *after the source row* S *has been displayed. The handoff cannot bypass the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, cannot import host normalization, and cannot replace the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *boundary by quotient normalization, evaluator correctness, or global confluence.*

Proof. Begin with the terminal readback lock of Definition 683.9. It places R behind the trace ledger, terminal-read row, transports, routes, provenance, and local name.

Apply Definition 683.10. The typed normal-form boundary is read through the displayed trace, terminal, transport, route, package, and naming rows rather than through host conversion or evaluator data.

Finally use Definition 683.12. The bridge exports exactly the continuation trace, terminal-read, typed-boundary, transport, route, provenance, and naming surface, so the typed-reduction handoff has no hidden host normalizer or off-route terminal reader. □

Theorem 683.27 (ContinuationTraceNormalForm consumer readback envelope)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the downstream consumer envelope consists exactly of the visible source route of Definition 683.25, the typed-reduction handoff of Definition 683.26, the terminal readback lock of Definition 683.9, and the standard bridge surface of Definition 683.12.*

The envelope exports no hidden evaluator, quotient trace equality, external normalizer, host conversion route, global confluence theorem, or unbounded fuel row beyond the displayed finite packet $S, T, R, Q, V, H, C, P, N$.

Proof. First combine Definition 683.25 with Definition 683.26. The source route displays every row consumed by the typed-reduction handoff before a terminal consumer reads the normal-form endpoint.

Next apply Definition 683.9. It makes the terminal row available only behind the finite trace, terminal-read, transport, route, provenance, and local naming rows.

Finally apply Definition 683.12 and Definition 683.6. The standard bridge surface is the public consumer object, and the non-escape clause rules out hidden evaluator state, quotient trace equality, external normalization, host conversion, global confluence, and unbounded fuel. □

Theorem 683.28 (ContinuationTraceNormalForm prefix splice lock)**.** *For every accepted* ContinuationTraceNormal *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, every prefix splice used by a downstream normal-form consumer is locked to the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *row* T *and to the displayed* Cont *route* C*. The*

splice cannot be supplied by a second trace ledger, host evaluator prefix, quotient trace representative, global confluence proof, or external normalizer.

Proof. Project the accepted carrier from Definition 683.1. The only trace ledger available to the packet is the displayed row T, and the only replay route available to a consumer is the displayed Cont row C.

Apply Definition 683.8. It exhausts exported trace reads by the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger and prevents a second prefix source from entering the public surface.

Finally use Definition 683.27. The consumer envelope routes terminal readback through the same $S, T, R, Q, V, H, C, P, N$ packet, so host prefixes, quotient representatives, global confluence, and external normalization have no splice coordinate. □

Theorem 683.29 (ContinuationTraceNormalForm terminal prefix transport)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, terminal prefix transport moves the terminal row* R*, the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row* Q*, and the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *boundary row* V *only through* H, C, P, N*. No consumer may transport* R, Q, V *through host conversion, evaluator graphs, quotient normalization, global confluence, or an external normalizer.*

Proof. Begin with Definition 683.9. It places terminal readback behind the displayed trace, terminal-read, transport, continuation, provenance, and local naming rows.

Use Definition 683.10. The typed boundary is read as a row of the same packet, so V travels with the displayed hsame transport and Cont route rather than with a host conversion relation.

Apply Definition 683.27. The envelope exposes R, Q, V only inside $S, T, R, Q, V, H, C, P, N$, and the excluded evaluator, quotient, confluence, and external-normalizer coordinates are absent from that carrier. □

Theorem 683.30 (ContinuationTraceNormalForm normal-boundary consumer totality)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the normal-boundary consumer surface is exhausted by* $S, T, R, Q, V, H, C, P, N$*. The consumer cannot use host conversion, quotient normalization, evaluator graphs, global confluence, or an external normalizer as an additional terminal route.*

Proof. Start with Definition 683.23. It identifies the typed normal-form boundary with the displayed continuation trace, terminal-read, typed-boundary, transport, route, provenance, and naming rows.

Next combine the prefix splice lock with terminal prefix transport. The splice is tied to T and C, while the terminal rows R, Q, V move only through H, C, P, N.

Finally apply Definition 683.6. Since the accepted carrier has no host conversion, quotient-normalization, evaluator-graph, global-confluence, or external-normalizer coordinate, the displayed rows exhaust every normal-boundary consumer read. □

Theorem 683.31 (ContinuationTraceNormalForm root readback order exhaustion)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, root readback is exhausted in the order source row* S*,* $\mathsf{ContinuationTrace}^{\uparrow}$ *trace ledger* T*, terminal row* R*,* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row* Q*,* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *typed-boundary row* V*, then the structural rows* H, C, P, N*. No downstream normal-form consumer may read a terminal or typed boundary before that displayed order has been consumed.*

Proof. Begin with terminal scope, Definition 683.7. It places terminal exposure behind the trace ledger, terminal-read row, and typed normal-form row of the same accepted packet.

Next use trace ledger exhaustion, Definition 683.8. The trace portion is exactly the finite T row replayed from source S, so no second trace source can enter before the terminal rows are read.

Apply terminal readback lock, Definition 683.9. It locks R behind T, Q, H, C, P, N, which fixes the terminal-read stage before any downstream endpoint consumer is exposed.

Finally use typed boundary nonescape, Definition 683.10. The typed boundary V has no host conversion, quotient-normalization, evaluator-graph, global-confluence, or external-normalizer substitute, so the displayed order exhausts root readback. □

Theorem 683.32 (ContinuationTraceNormalForm trace-terminal-typed lock)**.** *Let $Z = (S, T, R, Q, V, H, C, P, N)$ and $Z' = (S', T', R', Q', V', H, C, P, N)$ be accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packets with the same structural rows H, C, P, N. If their source, trace, terminal, terminal-read, and typed-boundary reads are transported through those shared rows, then both packets expose the same trace-terminal-typed boundary to every downstream normal-form consumer.*

Proof. First apply trace ledger exhaustion, Definition 683.8. Any trace read is exhausted by the displayed ledger and replayed through the shared continuation row C.

Next apply terminal readback lock, Definition 683.9. The shared H, C, P, N rows place terminal readback behind the same transport, replay, provenance, and naming surface in both packets.

Use typed boundary nonescape, Definition 683.10. The typed boundary is a row of the packet and cannot be separated by host conversion, quotient normalization, evaluator graphs, global confluence, or an external normalizer.

Finally invoke the standard bridge surface, Definition 683.12. The public consumer sees exactly the continuation trace, terminal-read, typed-boundary, transport, route, provenance, and naming rows, so shared H, C, P, N lock the trace-terminal-typed boundary. □

Theorem 683.33 (ContinuationTraceNormalForm downstream normal-form consumer boundary)**.** *Every downstream normal-form consumer of an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$ is bounded by the displayed source, trace, terminal, terminal-read, typed-boundary, transport, route, provenance, and local naming rows. The consumer surface excludes host conversion, quotient normalization, evaluator graphs, global confluence, and external normalizers.*

Proof. Start from normal-boundary consumer totality, Definition 683.30. It exhausts the normal-boundary surface by the rows $S, T, R, Q, V, H, C, P, N$.

Next combine terminal scope and the standard bridge surface, Definition 683.7 and Definition 683.12. Together they route the public consumer through $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{ContinuationBigStep}^{\uparrow}$, and $\mathsf{TypedReductionNormalForm}^{\uparrow}$ inside the same finite packet.

Use root readback order exhaustion from Definition 683.31. It fixes the source, trace, terminal, terminal-read, and typed-boundary order before the downstream consumer may read the endpoint.

Finally use typed boundary nonescape, Definition 683.10. The excluded host conversion, quotient-normalization, evaluator-graph, global-confluence, and external-normalizer objects are not coordinates of Z, so they cannot extend the downstream boundary. □

Theorem 683.34 (ContinuationTraceNormalForm terminal selector totality)**.** *Every downstream normal-form consumer route for an accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet*

$$Z = (S, T, R, Q, V, H, C, P, N)$$

passes through the source row S, trace ledger T, terminal row R, terminal-read row Q, typed normal-form row V, transport row H, continuation row C, provenance row P, and local naming

row N. Terminal selection is fixed by the trace ledger and terminal-read row, and the consumer receives no host conversion, quotient normalization, evaluator graph, global confluence theorem, or external normalizer.

Proof. Start with terminal scope and trace ledger exhaustion, Definition 683.7 and Definition 683.8. They force terminal exposure to pass through the finite trace ledger T after the source row S.

Next apply terminal readback lock, Definition 683.9. The terminal selection is tied to R, Q and carried through H, C, P, N, so a downstream consumer cannot substitute an evaluator state or quotient normalizer for the terminal-read witness.

Finally use typed boundary nonescape and the downstream normal-form consumer boundary, Definition 683.10 and Definition 683.33. The typed row V is reached only through the displayed packet, which exhausts every terminal selector route by $S, T, R, Q, V, H, C, P, N$. □

Theorem 683.35 (ContinuationTraceNormalForm normal row precedence)**.** *In every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$, the normal row V can be read only after the source row S, trace ledger T, terminal row R, terminal-read row Q, transport row H, continuation row C, provenance row P, and local naming row N have fixed the terminal selector route. No downstream consumer may read V as a host conversion result, quotient-normalized value, evaluator graph endpoint, confluence result, or external normalizer output.*

Proof. Begin with root readback order exhaustion, Definition 683.31. It orders source, trace, terminal, terminal-read, and typed-boundary access before any downstream endpoint consumer is exposed.

Use terminal selector totality from Definition 683.34. The same S, T, R, Q, H, C, P, N route fixes how the consumer reaches V, so the normal row is not an independent terminal selector.

Finally apply typed boundary nonescape, Definition 683.10. The typed normal-form row is a coordinate of the accepted packet and has no host, quotient, evaluator, confluence, or oracle substitute. □

Theorem 683.36 (ContinuationTraceNormalForm consumer route nonbranching)**.** *Let $Z = (S, T, R, Q, V, H, C, P,$ and $Z' = (S', T, R', Q, V', H, C, P, N)$ be accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packets whose trace ledger, terminal-read witness, typed boundary, transport row, consumer route, provenance row, and local name are the same. Then both packets determine the same downstream normal-form consumer surface. No host conversion, quotient normalization, evaluator graph, global confluence theorem, or external normalizer can branch the consumer route.*

Proof. First apply terminal route determinacy, Definition 683.11. Agreement on the trace ledger, typed boundary, transport, route, provenance, and local naming rows fixes the public normal-form consumer surface.

Next combine terminal readback lock with normal row precedence, Definition 683.9 and Definition 683.35. The shared terminal-read witness and typed boundary force both packets through the same T, Q, V, H, C, P, N route before the consumer may read the normal form.

Finally use the downstream normal-form consumer boundary, Definition 683.33. Since the excluded host conversion, quotient normalization, evaluator graph, global confluence, and external normalizer have no coordinate in either packet, they cannot separate the two consumer surfaces. □

Theorem 683.37 (ContinuationTraceNormalForm terminal-consumer cut)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$, any terminal consumer that reads the endpoint row R must pass through the displayed trace row T, terminal-read row Q, typed-boundary row V, transport row H, continuation route C, provenance row P, and local naming*

row N. No host evaluator state, quotient trace representative, global confluence proof, or external normalizer can supply an alternate terminal route.

Proof. Project the accepted carrier of Definition 683.1. The terminal row R is not a free endpoint; it appears in the finite packet beside T, Q, V, H, C, P, N.

Apply terminal scope and terminal readback lock, Definition 683.7 and Definition 683.9. These place terminal exposure behind the trace ledger, terminal-read row, transport, continuation route, provenance, and local naming surface.

Finally use consumer readback and downstream boundary, Definition 683.27 and Definition 683.33. The accepted consumer reads R only inside $S, T, R, Q, V, H, C, P, N$, while the excluded host evaluator, quotient, confluence, and external-normalizer coordinates are absent from the packet. □

Theorem 683.38 (ContinuationTraceNormalForm trace-replay cut)**.** *For every accepted* ContinuationTraceNormalF *packet $Z = (S, T, R, Q, V, H, C, P, N)$, any replay cut used by a downstream normal-form consumer is exhausted by the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *row T and the displayed* Cont *route C. The replay cut cannot import a second trace ledger, host evaluator graph, quotient trace equality, global confluence datum, or external normalizer.*

Proof. Start with trace ledger exhaustion, Definition 683.8. It says that trace reads exported by the packet are exactly the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger rather than a second replay source.

Use prefix splice lock, Definition 683.28. The splice available to a downstream normal-form consumer is tied to the displayed T row and the displayed C route.

Finally apply root readback order exhaustion, Definition 683.31. The replay stage precedes terminal and typed-boundary consumption in the displayed packet order, so host evaluator graphs, quotient trace equality, global confluence, and external normalizers have no replay-cut coordinate. □

Theorem 683.39 (ContinuationTraceNormalForm normal-boundary cut)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$, the normal-boundary consumer cut is exactly the displayed route through $S, T, R, Q, V, H, C, P, N$. A normal-boundary consumer cannot import host conversion, quotient normalization, evaluator graphs, global confluence, or external normalizers as extra boundary data.*

Proof. Begin with normal-boundary consumer totality, Definition 683.30. It already exhausts the normal-boundary surface by the displayed packet rows.

Combine the terminal-consumer cut and trace-replay cut. The terminal cut forces reads of R, Q, V through T, H, C, P, N, while the replay cut exhausts the trace side by T and C.

Finally apply typed-boundary nonescape and downstream normal-form consumer boundary, Definition 683.10 and Definition 683.33. Host conversion, quotient normalization, evaluator graphs, global confluence, and external normalizers are not coordinates of Z, so the displayed rows are the whole normal-boundary cut. □

Theorem 683.40 (ContinuationTraceNormalForm root terminal readback totality)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet $Z = (S, T, R, Q, V, H, C, P, N)$, the terminal row R is exposed as a root readback row before any downstream normal-form consumer can read it. The exposure is anchored by the displayed* BHist *source row S, the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger T, the* $\mathsf{ContinuationBigStep}^{\uparrow}$ *terminal-read row Q, the componentwise* hsame *transports H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* NameCert *row N.*

Proof. Start with root readback order exhaustion, Definition 683.31. It places the terminal row after the source and trace rows, and before any downstream terminal or typed-boundary consumer.

Apply the terminal-consumer cut, Definition 683.37. It forces every terminal consumer through T, Q, V, H, C, P, N before the endpoint row R can be read.

Finally use the trace-replay cut and normal-boundary cut, Definition 683.38 and Definition 683.39. The replay side is exhausted by T and C, and the normal-boundary side is exhausted by the same finite packet. Thus terminal readback is total at the root before consumer access. □

Theorem 683.41 (ContinuationTraceNormalForm root replay prefix totality)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, every root prefix read used for replay factors through the displayed* $\mathsf{ContinuationTrace}^{\uparrow}$ *row* T *and the displayed* Cont *route* C*. The prefix read remains attached to the same* BHist *source row* S*, componentwise* hsame *transports* H*,* Pkg *provenance row* P*, and local* NameCert *row* N*; it cannot use an off-packet replay, host evaluator prefix, quotient trace equality, global confluence datum, or external normalizer.*

Proof. Begin with the trace-replay cut, Definition 683.38. It states that any replay cut used downstream is exhausted by the displayed trace row T and displayed continuation route C.

Use root readback order exhaustion, Definition 683.31. The replay stage is part of the ordered root route from S through T before terminal and typed-boundary rows are consumed.

Now apply the terminal-consumer cut and normal-boundary cut. The terminal cut keeps endpoint reads behind the same T, Q, V, H, C, P, N rows, while the normal boundary cut prevents host reduction or off-packet boundary data from supplying a prefix. Hence every root prefix read is the displayed T-through-C replay. □

Theorem 683.42 (ContinuationTraceNormalForm root normal boundary exhaustion)**.** *For every accepted* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *packet* $Z = (S, T, R, Q, V, H, C, P, N)$*, the root normal-form boundary is exhausted by the displayed source, trace, terminal, terminal-read, typed-boundary, transport, route, provenance, and local naming rows. The boundary has no host reduction, hidden terminal row, or off-packet replay beyond the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface of* Z*.*

Proof. First apply the normal-boundary cut, Definition 683.39. It states that the normal-boundary consumer cut is exactly the displayed route through $S, T, R, Q, V, H, C, P, N$.

Use root readback order exhaustion to keep the boundary on the root order from source through trace, terminal, terminal-read, and typed-boundary rows. Then use root terminal readback totality, Definition 683.40, to expose the terminal row before any consumer can detach it.

Finally apply root replay prefix totality, Definition 683.41. It blocks off-packet replay at the prefix stage, while the normal-boundary cut already blocks host reduction and hidden terminal data. Therefore the root normal-form boundary is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ packet is generated from a source history row, a finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger, a terminal row, a $\mathsf{ContinuationBigStep}^{\uparrow}$ terminal-read witness, a $\mathsf{TypedReductionNormalForm}^{\uparrow}$ row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationTraceNormalForm}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 683.1, Definition 683.2, Definition 683.6, Definition 683.7, Definition 683.8, Definition 683.9, Definition 683.10, Definition 683.12, Definition 683.13, Definition 683.14, Definition 683.15, Definition 683.16, Definition 683.17, and Definition 683.18 bind carrier admission, trace replay, terminal-read exposure, typed normal-form readback, non-escape, trace-ledger exhaustion, terminal readback lock, typed-boundary nonescape, the standard bridge surface, replay-prefix determinacy, terminal typed-boundary exhaustion, the root package, terminal-source order, typed-boundary route, and downstream unblock ledger to the displayed BHist, BMark, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ContinuationTraceNormalForm}\uparrow}$, bridgeChecked)
Lean target: `BEDC.Derived.ContinuationTraceNormalFormUp.ContinuationTraceNormalFormCarrier\_namecert\_obligations`
Bridge status: bridgeChecked
Origin: human
Not claimed: This chapter does not close typed subject reduction, global confluence, host conversion, evaluator correctness, quotient behaviour, or machine-checked bridge verification.
Upgrade path: No further formal-axis upgrade remains; the chapter is at bridgeCheckedV.

Chapter 684

A Concrete Naming Certificate for $\mathsf{ContextualClassReading}^{\uparrow}$

The $\mathsf{ContextualClassReading}^{\uparrow}$ packet realizes the contextual-class restriction of section 9457.1 as finite naming data. A class-like phrase is admitted only through the displayed context that classifies it, together with transport, route, provenance, and local ledger rows. The packet does not supply a host class object, a universal membership predicate, or an ontology-level promotion of the phrase.

The packet consumes the fivefold Russell boundary surface of chapter 517 and the analysis ladder of chapter 518. Their expression row, displayed contextual classifier, relation-first bridge, scope row, witnessed descent ledger, and laddered relation-first stability are the sibling rows repackaged as E, G, R, S, H, C, P, N for contextual classification, relation-first licensing, and scoped consumption. Thus the contextual-class carrier is the local reading of the Russell boundary and its analysis ladder rather than an isolated kernel packet.

Definition 684.1 (Contextual class reading carrier)**.** *A* $\mathsf{ContextualClassReading}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, G, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the expression row whose class-like grammar is being read;*
2. *G is the displayed context row that supplies the local classifier for E;*
3. *R is the relation-first row recording the behaviour that licenses the contextual read before any object-like consumption;*
4. *S is the scope row naming the stratum and consumer boundary in which the classifier may be used;*
5. *H records componentwise* hsame *transports for expressions, contexts, relation rows, scope rows, provenance rows, route rows, and local names;*
6. *C records the displayed* Cont *routes by which a consumer may inspect E, read G, replay R, and return to the scoped ledger S;*
7. *P is the* Pkg *provenance over* BHist, Cont, hsame, *contextual classifier rows, relation rows, scope rows, and the surrounding Russell boundary packets;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ContextualClassReading}^{\uparrow}}$ *ledger.*

The carrier is exactly this finite packet. It has no coordinate for a host class object, unscoped membership relation, quotient over class phrases, reducibility principle, or context-free classifier.

Lean4 (definition) `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier`

Theorem 684.2 (Contextual class reading NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContextualClassReading}^{\uparrow}}$ *surface for Definition 684.1 consists of carrier admission for* $K = (E, G, R, S, H, C, P, N)$*, contextual classifier exposure through* G*, relation-first licensing through* R*, scoped consumption through* S*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local ledger exactness through* N*.*

Proof. Carrier admission is projection from Definition 684.1. The expression, context, relation, scope, transport, route, provenance, and naming rows are all displayed coordinates of the same finite BHist packet.

Use Definition 517.2 to read contextual-class discipline as one of the five Russell boundary obligations. The context row G supplies the classifying context, while the relation row R keeps the behaviour that licenses the name visible before any object-like use is consumed.

The remaining obligations are routed through S, H, C, P, N. The scope row fixes where the classifier may be used, the componentwise hsame data records permitted transports, and the Cont routes replay the public read without adding a host class coordinate. □

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier namecert obligations`

Theorem 684.3 (Contextual class reading no object promotion)**.** *Let* $K = (E, G, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ContextualClassReading}^{\uparrow}$ *carrier. No public read of* E *promotes the contextual classifier row* G *into a host class object, an unstratified membership predicate, or a context-free object whose members may be inspected outside* S, H, C, P, N*.*

Proof. Project the accepted packet to its displayed rows. The only coordinate that can classify E is G, and G is paired with the scope row S and the routes in C.

Apply Definition 506.2. A contextual class expression cannot be turned into an object-level self-membership carrier merely by asking a class-like question about it.

Then apply Definition 517.3. The packet keeps expression, contextual classifier, bridge, and ledger rows distinct, so a consumer can inspect the class-like phrase only through the displayed context and cannot recover a host class object from the local ledger. □

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier no object promotion`

Theorem 684.4 (Contextual class reading classifier transport)**.** *Let* $K = (E, G, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ContextualClassReading}^{\uparrow}$ *carrier. Every transported classifier read of the contextual class expression factors through the componentwise* hsame *row* H *and the displayed* Cont *routes* C*, returning to the same expression, context, relation, scope, provenance, and local ledger rows.*

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier classifier transport`

Proof. Start from a transported public classifier read. Carrier projection gives the expression row E, the context row G, and the relation-first row R as the only rows that can license the class-like reading.

The transport row H supplies the componentwise hsame data for those rows and for S, C, P, N. The route row C supplies the displayed Cont path by which the consumer reads the context and returns to the scoped ledger.

Use Definition 518.2 for contextual-class locality and relation-first classifier stability. Since the ladder obligations keep contextual class reads and relation rows tied to their displayed transports, the transported read remains inside E, G, R, S, H, C, P, N. □

Theorem 684.5 (Contextual class reading ledger exactness)**.** *Every accepted* ContextualClassReading$^{\uparrow}$ *packet exports contextual class information only through the finite rows*

$$E, G, R, S, H, C, P, N.$$

The ledger exposes the class-like expression, its displayed context, relation-first licensing row, scoped consumer boundary, componentwise hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no hidden class ontology, no primitive class extension, no global membership predicate, no reducibility principle, and no context-free classifier.*

Proof. Read the carrier as the tuple $K = (E, G, R, S, H, C, P, N)$. The definition lists every row available to a downstream consumer, and the local NameCert ledger records exactly those rows.

Apply Definition 684.2 to identify the complete obligation surface. Contextual classifier exposure is G, relation-first licensing is R, scoped consumption is S, and transport, route, provenance, and local naming are H, C, P, N.

Apply Definition 684.3 and Definition 684.4. The first theorem blocks object promotion, while the second confines transported classifier reads to the displayed hsame and Cont rows.

Thus the ledger is exact: every public contextual-class read is a read of the finite packet, with no hidden ontology or context-free classifier left over. □

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingLedgerExactness`

Theorem 684.6 (Contextual class reading public interface export)**.** *Let $K = (E, G, R, S, H, C, P, N)$ be an accepted* ContextualClassReading$^{\uparrow}$ *carrier. The public interface exported by K is exactly the finite contextual classifier row G, relation-first licensing row R, scoped consumer row S, componentwise transport row H, displayed replay row C, provenance row P, and local* NameCert *row N, together with the expression row E they license. No public consumer receives a host class object, a context-free classifier, or an unscoped membership row outside this displayed packet.*

Proof. Project the accepted carrier. The definition exposes only E, G, R, S, H, C, P, N, and the obligation theorem identifies contextual classifier exposure, relation-first licensing, scoped consumption, transport, replay, provenance, and local naming as the complete surface.

Apply the no-object-promotion theorem. It blocks any public read of E from turning G into a host class object or an unstratified membership predicate.

Apply classifier transport and ledger exactness. Transported reads must return through H and C, while the ledger exports only the finite packet rows. Thus the public interface is precisely the displayed contextual classifier, relation-first licensing, scoped consumer, transport, replay, provenance, and local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier public interface export`

Theorem 684.7 (Contextual class reading mature sibling route)**.** *Every mature consumer of* ContextualClassReading$^{\uparrow}$ *factors through the* RussellBoundary$^{\uparrow}$ *and* RussellAnalysisLadder$^{\uparrow}$ *sibling rows before reading the local contextual-class packet. The consumer receives the displayed expression, contextual classifier, relation-first licensing, scope, transport, replay, provenance, and local naming rows, but receives no host class ontology, global membership relation, reducibility principle, or context-free classifier.*

Proof. Project the carrier of Definition 684.1. The local rows E, G, R, S, H, C, P, N expose only a contextual classifier and its scoped relation-first route.

Apply Definition 684.2. The public read is already constrained to contextual classifier exposure, relation-first licensing, scoped consumption, transport, replay, provenance, and local naming.

Now use the sibling dependency paragraph at the start of the chapter. The contextual packet consumes the fivefold Russell boundary surface and the analysis ladder before its own local read is available, so a mature consumer must factor through those sibling rows first. The no-object-promotion, public-interface, and ledger-exactness theorems then rule out host class ontology, global membership, reducibility, and context-free classifiers. □

Lean4 ✓ `BEDC.Derived.ContextualClassReadingUp.ContextualClassReadingCarrier mature sibling route`

Closure status for $\mathsf{NameCert}_{\mathsf{ContextualClassReading}\uparrow}$.

Constructive story: A $\mathsf{ContextualClassReading}^{\uparrow}$ packet is generated from a class-like expression row, a displayed classifier context, a relation-first licensing row, a scope row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContextualClassReading}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 684.1, Definition 684.2, Definition 684.3, Definition 684.4, and Definition 684.5 bind the finite carrier, local obligations, no-promotion boundary, classifier transport, and ledger exactness for contextual class readings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContextualClassReading}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, host class ontology, primitive class extensions, global membership, reducibility, or context-free classifiers.

Upgrade path: No further theory-axis grade remains; Lean verification and bridge checking remain separate verification-axis work for the displayed contextual-class route.

Chapter 685

A Concrete Naming Certificate for $\mathsf{EmptyBoundarySelector}^{\uparrow}$

The $\mathsf{EmptyBoundarySelector}^{\uparrow}$ interface realizes the empty-boundary witness surface of Definition 9399.1. It packages one enacted mark above a displayed source Hist row, the resulting Ext row, the local witness that this mark is the enacted branch, and the naming boundary that keeps the witness local. It is not a global branch selector and not an oracle for histories outside the packet.

The local enacted-boundary witness is consumed by chapter 585, chapter 516, and chapter 623. $\mathsf{EmptyBoundarySelector}^{\uparrow}$ supplies the one-step branch row, $\mathsf{FableBranchWitness}^{\uparrow}$ exposes it as a local branch witness, $\mathsf{EmptyFableLedger}^{\uparrow}$ records it as a finite ledger entry, and $\mathsf{FableMachineClock}^{\uparrow}$ orders that ledger without promoting the row to a global chooser.

Definition 685.1 (EmptyBoundarySelector carrier)**.** *An* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, M, R, E, W, T, C, P, N)$$

with the following displayed rows:

1. *H is the source* Hist *row whose empty boundary is being filled;*
2. *M is the enacted mark row, one of the two kernel marks b_0 or b_1;*
3. *R is the resulting* Hist *row after the enacted mark is recorded above H;*
4. *E is the displayed* $\mathsf{Ext}(H, M, R)$ *row;*
5. *W is the local enacted-mark witness row, exposing that M is the mark actually used by this packet;*
6. *T records componentwise* hsame *transports for H, M, R, E, W and the local naming row;*
7. *C records the displayed* Cont *routes by which a consumer reads the source row, enacted mark, result row, extension row, witness row, and local name;*
8. *P is the* Pkg *provenance over* Hist, Ext, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the Empty-boundary reading of Definition 9399.1;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $H, M, R, E, W, T, C, P, N$. It has no coordinate for an undisplayed selector function, a branch policy over unrelated histories, a truth oracle, or an infinite ledger.

Lean4 ✓ `BEDC.Derived.EmptyBoundarySelectorUp.EmptyBoundarySelectorTasteGate single carrier alignment`

Theorem 685.2 (EmptyBoundarySelector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelector}^{\uparrow}}$ *surface for the carrier of Definition 685.1 consists of five finite obligations:*

1. *carrier admission, requiring every accepted object to expose* $Q = (H, M, R, E, W, T, C, P, N)$*;*
2. *source-boundary admission, requiring* H *to be the displayed* Hist *row whose empty boundary is filled;*
3. *enacted-mark exactness, requiring* M*,* R*, and* E *to exhibit the same* $\mathsf{Ext}(H, M, R)$ *step;*
4. *witness exposure, requiring* W *to name the enacted mark locally and to be transported only through the componentwise* hsame *and* Cont *rows* T *and* C*;*
5. *non-escape, requiring the packet to export no hidden chooser, no branch policy for unrelated histories, no truth oracle, and no accumulated selector ledger.*

Proof. Carrier admission is projection from the finite tuple in Definition 685.1. The source history, enacted mark, result history, extension row, witness row, transport row, continuation row, provenance row, and naming row are all displayed coordinates.

Source-boundary admission reads H before any mark is available. The empty boundary is therefore the slot above this source row, not a reset to e and not a free-standing selector surface.

Enacted-mark exactness then ties M, R, and E together. Since E is the displayed $\mathsf{Ext}(H, M, R)$ row, a consumer cannot replace the enacted mark by an unrecorded choice while keeping the same certificate.

Finally W, T, C, P, N expose and delimit the witness. The witness is readable through the named transport and continuation rows, while provenance and the local NameCert row provide no coordinate for a hidden chooser, unrelated branch policy, truth oracle, or accumulated ledger. □

Theorem 685.3 (EmptyBoundarySelector witness exposure)**.** *Let* $Q = (H, M, R, E, W, T, C, P, N)$ *be an accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *carrier. Every consumer read of the enacted branch factors through the witness route*

$$H, M, R, E, W, T, C, P, N.$$

The consumer may inspect the source Hist *row, enacted mark, result row, extension row, local witness, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and local naming row, and receives no selector value outside those rows.*

Lean4 ✓ `BEDC.Derived.EmptyBoundarySelectorUp.EmptyBoundarySelectorWitness exposure`

Proof. Begin with the accepted carrier. Projection gives the rows $H, M, R, E, W, T, C, P, N$ and no second source for branch data.

Read the branch through M and E. The mark row M is the enacted mark, and E records that this mark extends the source row H to the result row R.

Now expose the local witness W. The only routes carrying W to a consumer are the componentwise transport row T and continuation row C, both named inside the same packet.

Package provenance and the local naming row close the route. Because the carrier has no additional selector coordinate, any branch read outside $H, M, R, E, W, T, C, P, N$ is unavailable to the accepted packet. □

Theorem 685.4 (EmptyBoundarySelector non-escape boundary)**.** *Every accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *packet is local to the displayed source row H and enacted extension row E. It exports no global chooser, no branch-selection rule for unrelated* Hist *rows, no truth oracle, no host equality on traces, and no accumulated selector ledger beyond the single enacted boundary step.*

Proof. Inspect the carrier inventory. The only source row is H, the only result row is R, and the only branch evidence is the pair consisting of the enacted mark M and the extension row E.

The witness row W remains attached to the same step. Its transport and continuation rows are componentwise, so they move the displayed witness surface without adding a rule for other histories.

The provenance and NameCert rows repack this local surface for downstream use. Since no row names a chooser over all Hist values, a trace equality principle, a truth oracle, or an accumulated ledger, the packet cannot export any of those objects. □

Theorem 685.5 (EmptyBoundarySelector ledger handoff)**.** *An accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *packet hands off exactly one local enacted-boundary witness to Empty-fable ledger consumers. When a downstream* $\mathsf{EmptyFableLedger}^{\uparrow}$, $\mathsf{FableBranchWitness}^{\uparrow}$, *or* $\mathsf{FableMachineClock}^{\uparrow}$ *consumer shares the displayed rows $H, M, R, E, W, T, C, P, N$, the handoff provides the source* Hist *row, enacted mark, result row, extension row, local witness, componentwise transport, continuation route, provenance, and local name, with no global chooser or extra ledger coordinate.*

Proof. Project the accepted selector packet first. The available handoff surface is precisely the displayed tuple $H, M, R, E, W, T, C, P, N$.

Next compare this tuple with the one-step Empty-fable consumers. $\mathsf{FableBranchWitness}^{\uparrow}$ reads the same local branch shape through its Hist, mark, result, Ext, admissibility, transport, continuation, package, and naming rows; $\mathsf{EmptyFableLedger}^{\uparrow}$ and $\mathsf{FableMachineClock}^{\uparrow}$ consume such rows as finite selector-ledger and local-clock entries.

Apply the non-escape boundary of Definition 685.4. The downstream consumer receives a single enacted-boundary witness and cannot promote it to a global chooser, a selector for unrelated histories, or an extra accumulated ledger coordinate. □

Theorem 685.6 (EmptyBoundarySelector fable sibling route)**.** *The selector handoff of Definition 685.5 is consumed by chapter 516 and chapter 585 through the same displayed rows*

$$H, M, R, E, W, T, C, P, N.$$

Thus the downstream fable ledger receives one local enacted-boundary witness and no global chooser, no branch policy for unrelated histories, and no additional selector ledger.

Proof. Begin with the ledger handoff theorem Definition 685.5. It identifies the complete handoff surface as the source history, enacted mark, result row, extension row, local witness, transport, continuation, provenance, and local naming row.

The EmptyFableLedger and FableBranchWitness sibling chapters read exactly that one-step fable route. Their consumer-facing rows are supplied by the displayed tuple $H, M, R, E, W, T, C, P, N$, so the handoff is a shared sibling route rather than a separate chooser interface.

Apply the non-escape boundary of Definition 685.4. Since the selector packet exports only the local enacted-boundary step, the downstream fable ledger cannot promote the handoff into a global chooser, an unrelated-history branch policy, or an extra accumulated ledger coordinate. □

Theorem 685.7 (EmptyBoundarySelector obligation surface)**.** *For every accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *carrier*

$$Q = (H, M, R, E, W, T, C, P, N),$$

the scoped obligation surface consists exactly of source Hist *admission, enacted mark admission, resulting* Hist *admission, displayed* $\mathsf{Ext}(H, M, R)$ *exactness, local witness exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelector}^{\uparrow}}$ *row. No obligation is stored outside these nine rows.*

Proof. Project the carrier Q. The first four rows expose the source history H, enacted mark M, result history R, and the local extension row E.

Inspect the local witness row W. It names the enacted branch for this packet only, and the transport and continuation rows T, C carry that witness componentwise with the same source, mark, result, and extension rows.

The provenance and local naming rows P, N close the displayed surface. Since the carrier has no further coordinate, the listed rows exhaust the scoped obligation surface. □

Theorem 685.8 (EmptyBoundarySelector boundary exhaustion)**.** *Let* $Q = (H, M, R, E, W, T, C, P, N)$ *be an accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *carrier. Every local boundary read is obtained by inspecting the displayed* $\mathsf{Ext}(H, M, R)$ *row* E *and the local witness row* W *over the same source history* H*. The packet has no second boundary branch, no unrelated source history, and no accumulated selector ledger.*

Proof. Start with a boundary-facing consumer. Projection gives a single source row H, a single enacted mark M, a single result row R, and a single extension row E.

Inspect E locally. Because it is the displayed $\mathsf{Ext}(H, M, R)$ row, the boundary read is tied to the enacted mark in this packet and cannot be retargeted to an unrelated history.

Read W through T and C. These rows transport and replay the same local witness, while P, N only record provenance and local naming. The boundary is therefore exhausted by one enacted extension step. □

Theorem 685.9 (EmptyBoundarySelector non-escape obligation)**.** *Every accepted* $\mathsf{EmptyBoundarySelector}^{\uparrow}$ *carrier* $Q = (H, M, R, E, W, T, C, P, N)$ *refuses global choice, unrelated-history branch policy, truth oracle, host trace equality, and accumulated selector ledger export. Any consumer read factors through the local route*

$$H, M, R, E, W, T, C, P, N.$$

Proof. Inspect the carrier inventory. The only mark row is M, the only extension row is E, and the only witness row is W.

The local Ext witness inspection supplies an enacted boundary for the displayed source history H and result history R. It does not supply a function choosing marks for other histories or a truth predicate over traces.

Transport, replay, provenance, and naming preserve this one route. Since none of the refused objects appears as a row of Q, no accepted consumer can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelector}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EmptyBoundarySelector}^{\uparrow}$ packet is generated from a source Hist row, enacted mark, result Hist row, displayed Ext row, local witness, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmptyBoundarySelector}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The carrier, witness exposure, ledger handoff, fable sibling route, obligation surface, boundary exhaustion, and non-escape theorem close the displayed nine-row packet for public empty-boundary selector consumers.

Formal verification: FormalStatus(NameCert$_{\text{EmptyBoundarySelector}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.EmptyBoundarySelectorUp.EmptyBoundarySelectorTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, global choice, branch policies for unrelated histories, truth oracles, host trace equality, accumulated selector ledgers, or a public standard bridge.

Upgrade path: Bridge closure requires a standard bridge for the public consumer boundary without adding global choice, unrelated-history branch policies, truth oracles, host trace equality, or accumulated selector ledgers.

Chapter 686

A Concrete Naming Certificate for $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$

The $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ surface packages a type-checking read as finite classifier membership rather than as a host typing judgment. It joins the membership row of $\mathsf{TypeClassifierMembership}^{\uparrow}$, the classifier typing row of $\mathsf{ClassifierTyping}^{\uparrow}$, the subject-reduction handoff of $\mathsf{SubjectReduction}^{\uparrow}$, the finite derivation route of $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$, and the proof-erasure readback of $\mathsf{ProofTermErasureTrace}^{\uparrow}$ into one locally named packet.

chapter 483 supplies the displayed Cont subject-reduction handoff consumed by the row R, so the membership-trace packet records preservation as a sibling route rather than as an ambient typing theorem.

Definition 686.1 (TypeCheckingMembershipTrace carrier)**.** *A* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* Ext*-membership row read as the local type-classifier decision;*
2. *D is the finite derivation trace row recording the classifier typing route;*
3. *R is the displayed* Cont *subject-reduction route attached to the same membership read;*
4. *S is the* SigRel *readback row relating the derivation route to the visible classifier name;*
5. *H records componentwise* hsame *transport for M, D, R, S and the local naming row;*
6. *C records the displayed* Cont *replay by which consumers reread membership, derivation, subject-reduction, and readback rows;*
7. *P is the* Pkg *provenance row over* $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{ClassifierTyping}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$, $\mathsf{ProofTermErasureTrace}^{\uparrow}$, Ext, Cont, SigRel, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}$ *row.*

The local classifier compares two accepted carriers only through the displayed membership, derivation, subject-reduction, readback, transport, continuation, package, and naming rows.

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceTasteGate single carrier alignment`

Theorem 686.2 (TypeCheckingMembershipTrace classifier stability)**.** *Let*

$$T = (M, D, R, S, H, C, P, N)$$

and

$$T' = (M', D', R', S', H', C', P', N')$$

be accepted $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *carriers whose classifier rows are related by the displayed local comparison. Then the* Ext*-membership read, finite derivation route,* Cont *subject-reduction handoff, and* SigRel *readback are preserved through the componentwise* hsame *rows and the displayed* Cont *replay. The preserved object is the same finite membership-trace packet; it supplies no host typing judgment, conversion oracle, or quotient proof-term equality.*

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceClassifier stability`

Proof. Project both carriers through Definition 686.1. The rows M, D, R, S name the classifier membership, derivation route, subject-reduction handoff, and readback boundary, while H records componentwise transport for those rows.

Use the displayed continuation replay C and C' to reread the same rows after transport. Since replay is attached to the packet coordinates, it cannot introduce an external typing derivation or conversion decision.

Finally repack the transported rows with the provenance and local naming rows P, N and P', N'. The result is still the finite $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ surface, so quotient proof-term equality and host type equality remain outside the exported classifier evidence. □

Theorem 686.3 (TypeCheckingMembershipTrace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TypeCheckingMembership}}$ *surface has five finite obligations:*

1. *carrier admission exposes exactly* $T = (M, D, R, S, H, C, P, N)$*;*
2. *membership classification reads accepted type-checking only from the displayed* Ext *row* M*;*
3. *derivation-route exactness keeps the classifier typing route inside* D *and its* SigRel *readback inside* S*;*
4. *subject-reduction handoff uses only the displayed* Cont *route* R *and its replay through* C*;*
5. *the non-escape ledger excludes host type equality, global normalization, proof irrelevance, quotient syntax, conversion oracles, and any proof-term equality not displayed in the packet.*

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceNameCert obligation surface`
Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceNameCert obligations`

Proof. Carrier admission is projection from the finite tuple. The membership row, derivation row, subject-reduction row, readback row, transport row, replay row, package row, and local naming row are all coordinates of the same BHist packet.

The classifier and derivation obligations are read from M, D, S. The row M is the only accepted Ext-membership decision, D is the only derivation route, and S is the only signature readback attached to that route.

The subject-reduction handoff is read from R and replayed through C. Because every exported row is packaged with H, P, N, no host type equality, global normalization theorem, proof irrelevance principle, quotient syntax, conversion oracle, or external proof-term equality can be recovered from the certificate surface. □

Theorem 686.4 (TypeCheckingMembershipTrace route coverage)**.** *Every accepted* TypeCheckingMembershipTrace *packet*

$$T = (M, D, R, S, H, C, P, N)$$

exposes the membership row M, the derivation route D, the subject-reduction handoff R, the SigRel *readback S, the* hsame *transport row H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}$ *row N before any consumer may cite type-checking membership from the packet.*

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceRoute coverage`

Proof. Project the accepted packet through Definition 686.1. The carrier definition lists exactly the eight displayed coordinates, so a consumer has no membership citation until the row M and its companion rows D, R, S, H, C, P, N have been exposed.

Apply Definition 686.2 to transport the displayed classifier read through H and C. Apply Definition 686.3 to require the local naming row before the packet is used as type-checking membership evidence. □

Theorem 686.5 (TypeCheckingMembershipTrace non-escape)**.** *No accepted* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *consumer can export host type equality, global normalization, proof irrelevance, quotient syntax, a conversion oracle, or proof-term equality outside the displayed coordinates M, D, R, S, H, C, P, N.*

Proof. The carrier has no coordinate for host type equality, global normalization, proof irrelevance, quotient syntax, a conversion oracle, or proof-term equality. Projection from T therefore reaches only M, D, R, S, H, C, P, N.

The final row of Definition 686.3 is the non-escape ledger for this packet. It binds every exported consumer to the finite rows already displayed, so no additional equality or conversion principle can be recovered from the certificate surface. □

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceMembership row nonescape`

Theorem 686.6 (TypeCheckingMembershipTrace obligation assembly)**.** *The carrier of Definition 686.1, the classifier stability of Definition 686.2, the route coverage of Definition 686.4, and the non-escape ledger of Definition 686.5 assemble the paper-side* obligation *surface for* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$*. This assembly leaves the verification axis at* unformalized*.*

Proof. The carrier fixes the finite packet shape, and classifier stability fixes how the displayed classifier rows persist under hsame transport and Cont replay. Route coverage then states which rows must be exposed before the packet is cited by a consumer.

The non-escape theorem supplies the refusal side of the local ledger. Together with the NameCert obligations in Definition 686.3, these rows specify the complete paper-side obligation surface while making no Lean-side verification claim. □

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTrace obligation assembly`

Theorem 686.7 (TypeCheckingMembershipTrace scoped kernel surface)**.** *The carrier, classifier stability, route coverage, non-escape ledger, and obligation assembly for* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *bind every accepted packet*

$$T = (M, D, R, S, H, C, P, N)$$

to the displayed Ext *membership row M, finite derivation route D,* Cont *subject-reduction handoff R,* SigRel *readback row S, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}$ *row N. The scoped kernel surface has no host type equality, quotient syntax, conversion oracle, proof irrelevance, or global normalization coordinate.*

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTrace scoped kernel surface`

Proof. Project the finite carrier with Definition 686.1. It gives exactly the rows M, D, R, S, H, C, P, N and no host typing or conversion coordinate.

Use classifier stability, Definition 686.2, to keep membership, derivation, subject-reduction, and readback data attached to the same packet under the displayed hsame transport and Cont replay.

Apply route coverage and non-escape, Definition 686.4 and Definition 686.5. The first requires all displayed rows before a consumer cites type-checking membership, while the second excludes host type equality, global normalization, proof irrelevance, quotient syntax, conversion oracles, and external proof-term equality.

Finally use Definition 686.6. The assembled obligation surface is therefore scoped to the local certificate: Ext membership, finite derivation, subject-reduction handoff, SigRel readback, transport, replay, provenance, and naming are all discharged inside T, with no appeal to a host equality or global normalizer. □

Theorem 686.8 (TypeCheckingMembershipTrace subject-reduction consumer lock)**.** *For every accepted* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *packet*

$$T = (M, D, R, S, H, C, P, N),$$

the subject-reduction row R is mandatory before the membership row M may be exported to any consumer. A consumer that cites type-checking membership must also expose the finite derivation route D, SigRel *readback row S, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N.*

Proof. Project the carrier in Definition 686.1. The membership row M is not isolated: it is packaged with the derivation route D, subject-reduction handoff R, readback row S, transport, replay, provenance, and local name.

Apply route coverage, Definition 686.4. It requires the displayed companion rows before a consumer can cite membership from the packet.

Finally use the scoped kernel surface, Definition 686.7. The surface binds M to D, R, S, H, C, P, N inside the same finite certificate and leaves no host equality or conversion coordinate that could export membership without the subject-reduction row. □

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceSubjectReduction consumer lock`

Theorem 686.9 (TypeCheckingMembershipTrace formal target handoff)**.** *The rows*

$$M, D, R, S, H, C, P, N$$

are the exact future Lean formalization target surface for $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$*:* Ext *membership, finite derivation, subject-reduction handoff,* SigRel *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The target surface contains no host type equality, global normalization, proof irrelevance, quotient syntax, conversion oracle, or proof-term equality field.*

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceFormalTarget handoff`

Proof. Begin with the carrier and classifier stability, Definition 686.1 and Definition 686.2. They identify the eight rows and keep membership, derivation, subject-reduction, and readback attached to the same packet under transport and replay.

Use NameCert obligations and obligation assembly, Definition 686.3 and Definition 686.6. These results account for carrier admission, classifier stability, route coverage, non-escape, provenance, and local naming without claiming Lean verification.

The subject-reduction consumer lock, Definition 686.8, fixes the consumer-facing dependency on R. Together with non-escape, Definition 686.5, this leaves exactly the displayed rows as the formal target and excludes the host-level typing and conversion fields. □

Theorem 686.10 (TypeCheckingMembershipTrace bridge precondition)**.** *No bridge-facing consumer may cite* $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ *type-checking membership unless it carries the* Ext *membership row M, finite derivation route D, subject-reduction handoff R,* SigRel *readback row S, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N from the same accepted packet. This precondition does not supply a* StandardBridge*; it only fixes the paper-side rows any later bridge must consume.*

Proof. Apply route coverage and the subject-reduction consumer lock, Definition 686.4 and Definition 686.8. A consumer must expose all displayed rows, and membership cannot leave the packet without the subject-reduction handoff.

Next use classifier stability and NameCert obligations, Definition 686.2 and Definition 686.3. They keep the classifier rows stable under H and C and require provenance and local naming before the packet is used as membership evidence.

Finally combine non-escape, obligation assembly, and the formal target handoff, Definition 686.5, Definition 686.6, and Definition 686.9. These results exclude host type equality, global normalization, proof irrelevance, quotient syntax, conversion oracles, proof-term equality, and any bridge claim beyond the displayed precondition rows. □

Lean4 ✓ `BEDC.Derived.TypeCheckingMembershipTraceUp.TypeCheckingMembershipTraceBridge precondition`

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$ packet is generated from finite Ext-membership, derivation-route, subject-reduction, SigRel readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 686.7, Definition 686.8, Definition 686.9, and the bridge precondition theorem Definition 686.10 bind Ext membership, finite derivation, subject-reduction, SigRel readback, hsame transport, Cont replay, Pkg provenance, local NameCert rows, the future formal target surface, and the bridge-facing precondition inside the finite membership-trace certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeCheckingMembershipTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host type equality, global normalization, proof irrelevance, quotient syntax, Lean verification, or bridge checking.

Upgrade path: Lean verification requires a kernel proof of the scoped membership, derivation, subject-reduction, readback, transport, replay, package, and naming surface.

Chapter 687

A Concrete Naming Certificate for $\mathsf{FableClockOrder}^{\uparrow}$

The $\mathsf{FableClockOrder}^{\uparrow}$ packet realizes the Empty-fable machine's local clock-order surface as finite certificate data. It records selector-ledger rows, local fable-clock ticks, adjacent step-order witnesses, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row. The packet is local to the displayed fable trace and has no coordinate for a synchronized clock shared across unrelated histories.

The selector-ledger row S and the hidden-chooser refusal are read through chapter 685, which supplies the empty-boundary selector discipline consumed by the local fable-clock order packet.

Definition 687.1 (FableClockOrder carrier)**.** *A* $\mathsf{FableClockOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (H, S, T, E, W, L, C, P, N)$$

with the following displayed rows:

1. *H is the source* Hist *row whose Empty-fable trace is being read;*
2. *S is the finite selector-ledger row beginning at* e *and recording the enacted marks in order;*
3. *T is the finite local fable-clock tick row, one tick for each displayed selector step;*
4. *E records the adjacent* Ext *rows that move the ledger from one displayed history to the next;*
5. *W records adjacent step-order witnesses connecting successive ledger positions and ticks;*
6. *L records componentwise* hsame *transport for H, S, T, E, W and the local naming row;*
7. *C records the visible* Cont *routes by which consumers read the source history, selector ledger, local ticks, adjacent extensions, step-order witnesses, and local name;*
8. *P is the* Pkg *provenance over* Hist, BHist, Ext, hsame, Cont, Pkg, NameCert, *selector-ledger rows, and local fable-clock ticks;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FableClockOrder}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no slot for an ambient host time, a clock shared by distinct histories, a comparison of unrelated fable traces, or a hidden chooser for missing selector steps.

Definition 687.2 (FableClockOrder classifier)**.** *The* $\mathsf{FableClockOrder}^{\uparrow}$ *classifier compares two accepted packets only by the displayed tuple*

$$H, S, T, E, W, L, C, P, N.$$

Two packet presentations agree when their source history rows, selector ledgers, local tick rows, adjacent extension rows, step-order witness rows, componentwise hsame *transports, continuation routes, provenance rows, and local naming rows agree componentwise. The classifier does not compare ambient times, synchronize unrelated histories, or choose a missing branch outside the displayed selector ledger.*

Theorem 687.3 (FableClockOrder NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FableClockOrder}^{\uparrow}}$ *surface for Definition 687.1 consists exactly of carrier admission for*

$$F = (H, S, T, E, W, L, C, P, N),$$

classifier exactness for the rows of Definition 687.2, selector-ledger exposure, local tick exposure, adjacent step-order witness exposure, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, and no-global-clock non-escape.*

Proof. Carrier admission is projection from the finite tuple in Definition 687.1. The source history, selector ledger, local tick row, adjacent extension row, witness row, transport row, continuation row, provenance row, and local naming row are all displayed coordinates.

Classifier exactness is projection from Definition 687.2. Agreement is componentwise on the same rows, so the certificate has no second comparison surface.

The remaining obligations are carried by L, C, P, N. Componentwise transport preserves the displayed rows, continuation routes expose them to consumers, provenance names the package boundary, and the local naming row binds the packet without exporting an ambient clock. □

Theorem 687.4 (FableClockOrder local step-order)**.** *For an accepted* $\mathsf{FableClockOrder}^{\uparrow}$ *packet*

$$F = (H, S, T, E, W, L, C, P, N),$$

each displayed selector step in S has a corresponding local tick in T and an adjacent step-order witness in W. Consumer reads of this order factor through the same componentwise transport row L, continuation row C, provenance row P, and local naming row N.

Lean4 ✓ `BEDC.Derived.FableClockOrderUp.FableClockOrderCarrier local step order`

Proof. Project the accepted carrier. The selector ledger S, local tick row T, adjacent extension row E, and witness row W are coordinates of the same finite packet.

Read adjacent order through E and W. The extension row records each displayed move between neighboring history positions, while the witness row attaches the corresponding local tick from T to that move.

Transport and replay are then fixed by L and C. Componentwise hsame movement preserves the rows being ordered, and the continuation route exposes only this local step-order surface.

Finally P and N bind the handoff to the named packet. Since the carrier has no coordinate for unrelated histories or ambient time, the order is local to the displayed selector ledger. □

Theorem 687.5 (FableClockOrder no-global-clock non-escape)**.** *Every accepted* $\mathsf{FableClockOrder}^{\uparrow}$ *packet exports only the finite local order carried by*

$$H, S, T, E, W, L, C, P, N.$$

It does not export a synchronized clock across distinct histories, an ambient host-time parameter, a global ordering of all fable traces, or a hidden chooser for selector steps not recorded in S*.*

Proof. Inspect the carrier inventory. The only time-like row is the local tick row T, and it is paired with the selector ledger S and adjacent witnesses W.

Use the local step-order theorem to delimit consumer access. Any read of order passes through the displayed transport, continuation, provenance, and naming rows attached to the same packet.

The excluded clock objects have no coordinate in the carrier and no clause in the classifier. Therefore no projection from an accepted packet can produce a synchronized clock, host-time parameter, global trace order, or hidden selector for steps outside S. □

Theorem 687.6 (FableClockOrder Empty-fable sibling route)**.** *Every accepted* $\mathsf{FableClockOrder}^{\uparrow}$ *packet reads its selector-ledger and local tick rows through the Empty-fable sibling route: chapter 516 supplies the generated ledger, chapter 494 supplies the finite selector spine, and chapter 685 supplies the enacted-mark witness. The* $\mathsf{FableClockOrder}^{\uparrow}$ *carrier adds local order over those rows without adding global time, host synchronization, or hidden selector choice.*

Proof. Project the accepted carrier $F = (H, S, T, E, W, L, C, P, N)$. The selector ledger S and local tick row T are displayed coordinates of the packet, and the adjacent rows E, W attach the local order witness to the same trace.

Read the sibling route horizontally. The Empty-fable ledger supplies the generated trace ledger, the selector-ledger chapter supplies the finite selector spine, and the empty-boundary selector supplies the enacted mark that each local step records.

Apply the local step-order and no-global-clock theorems above. They show that $\mathsf{FableClockOrder}^{\uparrow}$ only orders the displayed sibling rows through L, C, P, N, so the packet exports no synchronized host clock and no hidden selector for a step outside S. □

Closure status for $\mathsf{NameCert}_{\mathsf{FableClockOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FableClockOrder}^{\uparrow}$ packet is generated from a source Hist row, a finite selector ledger, local fable-clock ticks, adjacent Ext and step-order witness rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FableClockOrder}^{\uparrow}}$, seed)

Scope closed: Definition 687.1, Definition 687.2, Definition 687.3, Definition 687.4, and Definition 687.5 bind local fable-clock order as a finite NameCert packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FableClockOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, synchronized clocks across distinct histories, ambient host time, global ordering of all fable traces, or hidden selector choice.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, local step-order replay, no-global-clock non-escape, and NameCert row coverage for the same finite packet.

Chapter 688

A Concrete Naming Certificate for $\mathsf{ObserverPerspectiveComposition}^{\uparrow}$

The $\mathsf{ObserverPerspectiveComposition}^{\uparrow}$ surface packages a two-step observer comparison as one finite naming certificate. It composes two $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ routes through an $\mathsf{ObservationCoupling}^{\uparrow}$ row and a $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ row, so downstream consumers receive a displayed comparison packet rather than a host observer equality, a global spacetime coordinate, or a private Self row.

The composition consumes the classifier route of chapter 647, the coupling row of chapter 668, and the alignment row of chapter 643; it only repackages their displayed public rows into a two-step comparison packet.

Definition 688.1 (Observer-perspective-composition carrier)**.** *An* $\mathsf{ObserverPerspectiveComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (Q_0, Q_1, O, L, H, C, P, N)$$

with the following displayed rows:

1. *Q_0 and Q_1 are accepted* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ *rows naming the two comparison routes to be composed;*
2. *O is the accepted* $\mathsf{ObservationCoupling}^{\uparrow}$ *row that couples the observation windows used by the two classifier routes;*
3. *L is the accepted* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *row that aligns the public observer commitments visible to both routes;*
4. *H records componentwise* hsame *transports for Q_0, Q_1, O, L and for the package and local naming rows;*
5. *C records the displayed* Cont *routes that first read Q_0, then read O and L, then read Q_1 through the same finite packet;*
6. *P is the* Pkg *provenance over* $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$, $\mathsf{ObservationCoupling}^{\uparrow}$, $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveComposition}^{\uparrow}}$ *row.*

The composition compares observer perspectives only through $Q_0, Q_1, O, L, H, C, P, N$ and the recorded transports and routes.

Lean4 ✓ BEDC.Derived.ObserverPerspectiveCompositionUp.ObserverPerspectiveCompositionTasteGate
single carrier alignment

Theorem 688.2 (ObserverPerspectiveComposition NameCert obligations)**.** *For the carrier of Definition 688.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveComposition}^{\uparrow}}$ *surface has exactly five obligation rows: carrier exposure for* $Q = (Q_0, Q_1, O, L, H, C, P, N)$*; two classifier-route exposures through* Q_0, Q_1*; observation coupling through* O*; cross-perspective alignment through* L*; and non-escape from host observer equality, global spacetime, private* Self *rows, quotient histories, hidden synchronization, and unrecorded comparison evidence.*

Proof. Carrier exposure is projection from the finite packet $Q = (Q_0, Q_1, O, L, H, C, P, N)$.

The two route obligations are read from Q_0 and Q_1. Each route is already an $\mathsf{ObserverPerspectiveClassifier}^{\uparrow}$ row, so it exposes only its displayed observer, locality, GAP, transport, continuation, provenance, and naming data.

The middle obligations are read from O and L. The $\mathsf{ObservationCoupling}^{\uparrow}$ row supplies the paired observation-window boundary, while the $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ row supplies the public alignment boundary for the observer commitments.

Finally inspect the carrier list. No coordinate names host observer equality, global spacetime, private Self access, quotient history, hidden synchronization, or unrecorded comparison evidence, so none of those objects is exported by the local naming surface. □

Theorem 688.3 (ObserverPerspectiveComposition ledger exactness)**.** *Every public read exported by an accepted* $\mathsf{ObserverPerspectiveComposition}^{\uparrow}$ *carrier factors through the finite ledger*

$$Q_0, Q_1, O, L, H, C, P, N.$$

The ledger is exact: the two classifier routes, the observation-coupling row, the cross-perspective-alignment row, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row are available, and no other observer-comparison primitive is available.*

Proof. Project the accepted packet. The comparison data are the two classifier rows Q_0, Q_1, the coupling row O, and the alignment row L.

The transport row H moves only those displayed coordinates. The continuation row C replays the finite reading order from the first classifier route through the coupling and alignment rows to the second classifier route.

The package row P records provenance for the same displayed sources, and the local naming row N names no additional carrier. Hence every exported read factors through the listed ledger and the ledger is exact for public consumers. □

Theorem 688.4 (ObserverPerspectiveComposition route associativity)**.** *Let* $Q = (Q_0, Q_1, O, L, H, C, P, N)$ *be an accepted* $\mathsf{ObserverPerspectiveComposition}^{\uparrow}$ *carrier. Reading* Q_0 *then* O *then* L *then* Q_1 *through the displayed* Cont *row* C *gives the same public composition packet as first reading* Q_0 *through the coupling row* O *and then reading the aligned route* L, Q_1*, because both readings replay the same finite rows under the same componentwise* hsame *transport row* H *and the same* Pkg *provenance row* P*.*

Proof. Both readings start by projecting the same accepted carrier $Q = (Q_0, Q_1, O, L, H, C, P, N)$. The first bracketing reads the first classifier route, then the coupling row, then the alignment row, then the second classifier route.

The second bracketing reads the first classifier route together with the coupling row, then reads the alignment row together with the second classifier route. No new coordinate is introduced by this regrouping.

The displayed Cont row C is the only route data available, and the displayed hsame row H is the only transport data available. Since the provenance row P names the same four source surfaces in both readings, both bracketings expose the same public composition packet. □

Theorem 688.5 (ObserverPerspectiveComposition non-escape boundary)**.** *An accepted* ObserverPerspectiveCompo *carrier exports only two* ObserverPerspectiveClassifier$^{\uparrow}$ *rows, one* ObservationCoupling$^{\uparrow}$ *row, one* CrossPerspectiveAlignment$^{\uparrow}$ *row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It exports no host observer equality, no global spacetime, no private* Self *row, no quotient history, and no completed consciousness claim.*

Proof. Inspect the carrier coordinates of Definition 688.1. Every exported row is one of $Q_0, Q_1, O, L, H, C, P, N$.

The positive comparison data remain finite and public: two classifier routes, an observation-coupling boundary, and a cross-perspective-alignment boundary. The rows H, C, P, N transport, route, package, and name those same displayed coordinates.

No listed coordinate is host observer equality, global spacetime, private Self access, a quotient history, or a completed consciousness witness. Therefore an accepted composition packet cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverPerspectiveComposition}^{\uparrow}}$.

Constructive story: An ObserverPerspectiveComposition$^{\uparrow}$ packet is generated from two observer-perspective classifier routes, an observation-coupling row, a cross-perspective-alignment row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverPerspectiveComposition}^{\uparrow}}$, seed)

Scope closed: Definition 688.1, Definition 688.2, Definition 688.3, Definition 688.4, and Definition 688.5 bind the finite two-step observer-comparison packet over classifier, coupling, alignment, transport, route, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverPerspectiveComposition}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global observer identity, metric spacetime, private Self access, quotient-history identity, bridge status, Lean verification, or completed consciousness claims.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, ledger exactness, route associativity, and non-escape theorems for the same finite observer-perspective composition packet.

Chapter 689

A Concrete Naming Certificate for $\mathsf{FiniteWindowObserver}^{\uparrow}$

The $\mathsf{FiniteWindowObserver}^{\uparrow}$ packet realizes the inscription-event vision as a finite observation window with an explicit naming certificate. It keeps the prior observation state, selected public window, inscription record, checker result, downstream interface, transport, replay, provenance, and local name in one displayed BHist carrier. The packet is a concrete consumer surface; it does not appeal to private psychology, unbounded memory, hidden agency predicates, or an external semantic observer.

The selected public window is the sibling route between the inscription-event record of chapter 519, the selected and omitted ledger discipline of chapter 927, and the finite $\mathsf{StreamName}^{\uparrow}$ handoff of chapter 126. Thus $\mathsf{FiniteWindowObserver}^{\uparrow}$ consumes those surfaces to expose a checked inscription window rather than an isolated observer primitive.

chapter 519 supplies the visible inscription-event row, and chapter 520 supplies the Gap ledger row consumed by the checker and downstream-interface coordinates.

Definition 689.1 (FiniteWindowObserver carrier)**.** *A* $\mathsf{FiniteWindowObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (O, F, I, K, D, H, C, P, N)$$

whose displayed rows are as follows:

1. *O is the prior observation state from which the inscription event is read;*
2. *F is the finite selected window of public records made visible to the observer;*
3. *I is the inscription record containing the named commitment and its visible route;*
4. *K is the checker and admission result for the proposed record;*
5. *D is the downstream interface row that records which consumer may cite the accepted inscription;*
6. *H is the componentwise* hsame *transport row for the observation state, selected window, inscription record, checker result, downstream interface, provenance, and local name;*
7. *C stores the displayed* Cont *routes for window selection, checker replay, admission, downstream handoff, and local NameCert comparison;*
8. *P is the* Pkg *provenance over* BHist, BMark, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteWindowObserver}^{\uparrow}}$ *row.*

Two accepted carriers classify together only when these nine displayed rows are transported to the same finite observation window and the same inscription-facing consumer interface. The carrier has no coordinate for a private mental state, global observer identity, hidden source of agency, or unbounded history.

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverTasteGate single carrier alignment`

Theorem 689.2 (FiniteWindowObserver NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteWindowObserver}^{\uparrow}}$ *surface for the carrier of Definition 689.1 consists exactly of five finite obligations:*

1. *carrier admission for the packet $W = (O, F, I, K, D, H, C, P, N)$;*
2. *classifier transport of the prior observation state, finite window, inscription record, checker result, and downstream interface;*
3. *checker replay and admission exactness for the visible inscription record;*
4. *downstream consumer handoff through the displayed interface row;*
5. *ledger non-escape for private psychology, global observer identity, hidden agency predicates, unbounded history, and external semantic observation.*

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverNameCertObligations`

Proof. Carrier admission is projection from Definition 689.1. The accepted packet exposes the prior observation state, finite selected window, inscription record, checker result, downstream interface, transports, continuations, provenance, and local naming row.

Classifier transport uses H to align those rows componentwise. Once transported, the continuation row C is the only displayed route for selecting the window, replaying the checker, admitting the inscription, and handing the accepted record to a consumer.

Checker exactness is read from K together with the inscription row I. The downstream handoff is read from D, and the package provenance P fixes the inscription-event and GAP-ledger anchors for the same finite packet.

The carrier contains no row for private psychology, global observer identity, hidden agency predicates, unbounded history, or external semantic observation. Hence the five obligations exhaust the local naming surface. □

Theorem 689.3 (FiniteWindowObserver carrier transport)**.** *Let*

$$W = (O, F, I, K, D, H, C, P, N) \quad \textit{and} \quad W' = (O', F', I', K', D', H', C', P', N')$$

be accepted $\mathsf{FiniteWindowObserver}^{\uparrow}$ *carriers. If the displayed transport rows identify O with O', F with F', I with I', K with K', D with D', P with P', and N with N' componentwise by* hsame*, then both packets expose the same consumer-facing observation route: the same finite selected public window, inscription record, checker result, and downstream interface are read after transport and replay.*

↪variant `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverCarrierTransport`

Proof. Project the carrier definition. The only public observation coordinates are the prior state O, selected window F, inscription record I, checker row K, downstream interface D, and structural rows H, C, P, N.

Use the displayed H rows for componentwise transport. They align the prior state, window, inscription record, checker result, downstream interface, provenance, and local naming rows of W and W'.

Replay the aligned route through C and C'. Since replay only follows the displayed selection, checker, admission, and downstream handoff rows, the transported carriers expose the same selected public window, inscription record, checker result, and consumer interface.

No private observer state, unbounded history, hidden agency predicate, or external semantic observer appears in the carrier. Therefore nothing outside the transported finite rows can change the consumer-facing observation route. □

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverCarrierTransport`

Theorem 689.4 (FiniteWindowObserver bundle exhaustion)**.** *For every accepted* $\mathsf{FiniteWindowObserver}^{\uparrow}$ *carrier* $W = (O, F, I, K, D, H, C, P, N)$*, the selected window row* F *exposes a closed* ProbeBundle*-style finite enumeration of the observed histories consumed by the packet. The enumeration is stable under the displayed* hsame *transport rows, and every inscription handoff out of* W *consumes only rows in that finite bundle.*

Thus a downstream $\mathsf{StreamName}^{\uparrow}$ *or inscription consumer can cite the observer packet as a finite bundle-exhaustion certificate. It receives the observed histories displayed by* F*, the inscription row* I*, the checker row* K*, and the downstream interface* D*, but it cannot read an external enumeration, hidden stream tail, private observer state, or unbounded history.*

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverBundleExhaustion`

Proof. Project the selected window from the carrier. The row F is the only public finite-window coordinate, so the observed histories available to the observer are exactly the rows displayed there.

Use carrier transport, Definition 689.3. The componentwise H rows preserve the selected window and its observed histories under hsame, while replay through C follows the same finite selection route.

Now apply the inscription handoff row. The handoff reads F before I, K, D, so every inscription consumer is fed by the finite bundle already exposed by the observer packet. Since no carrier coordinate supplies an external enumeration or hidden stream tail, the displayed bundle exhausts the observation surface. □

Theorem 689.5 (FiniteWindowObserver StreamName window readback)**.** *Every downstream* $\mathsf{StreamName}^{\uparrow}$ *or inscription consumer of an accepted* $\mathsf{FiniteWindowObserver}^{\uparrow}$ *carrier* $W = (O, F, I, K, D, H, C, P, N$ *must read the selected public window row* F*, the inscription row* I*, the checker row* K*, and the downstream interface row* D *through the displayed finite route before any StreamName-facing readback is public.*

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverStreamNameWindowReadback`

Proof. Apply Definition 689.4. The selected public window row F is already the finite bundle from which every inscription handoff reads observed histories.

Project the carrier rows $O, F, I, K, D, H, C, P, N$. The prior state O feeds the selected window F; the inscription row I, checker row K, and downstream interface row D are the only public readback coordinates after that window.

Replay the route through C and transport it through H. Provenance and local naming through P, N bind the same finite packet, so a downstream $\mathsf{StreamName}^{\uparrow}$ or inscription consumer cannot expose a readback before the displayed route has supplied F, I, K, D. □

Theorem 689.6 (FiniteWindowObserver inscription handoff)**.** *For an accepted* FiniteWindowObserver$^{\uparrow}$ *carrier* $W = (O, F, I, K, D, H, C, P, N)$*, every inscription-event consumer reads the accepted record through the finite route*

$$O \longrightarrow F \longrightarrow I \longrightarrow K \longrightarrow D.$$

The route first selects the finite public window, then reads the named inscription record, then replays the checker and admission result, and finally exposes only the downstream interface row certified by the local NameCert *packet.*

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverInscriptionHandoff`

Proof. Project the carrier of Definition 689.1. The rows O and F determine the prior observation state and the finite selected window in which the event is read.

The inscription row I provides the named commitment and visible route. The checker row K records the admission result for that same displayed record, while H transports the rows to the common classifier surface.

Replay the route through C. The continuation first selects F from O, then reads I, then checks and admits it through K, and finally hands the accepted record to the downstream interface D.

The local rows P, N bind the route to package provenance and naming. Since no other carrier coordinate can be read by the consumer, the inscription handoff is exactly the displayed finite route. □

Theorem 689.7 (FiniteWindowObserver ledger non-escape)**.** *Every consumer-facing route out of an accepted* FiniteWindowObserver$^{\uparrow}$ *carrier factors through the finite rows* $O, F, I, K, D, H, C, P, N$*. No route exports a private mental state, global observer identity, hidden agency predicate, unbounded observation history, external semantic observer, or unchecked inscription acceptance.*

Proof. Take any consumer-facing route and project the accepted carrier. The only observation-facing rows are the prior state O, selected window F, inscription record I, checker result K, and downstream interface D.

The remaining rows H, C, P, N transport, replay, record provenance, and name the packet. They do not introduce a second source of observer identity or a separate semantic acceptance relation.

By the NameCert obligations of Definition 689.2, checker replay, admission, and consumer handoff are already exhausted by the displayed finite rows. Therefore the ledger cannot export anything outside $O, F, I, K, D, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverLedgerNonescape`

Theorem 689.8 (FiniteWindowObserver obligation closure surface)**.** *For the same accepted packet* $W = (O, F, I, K, D, H, C, P, N)$*, the carrier of Definition 689.1, the NameCert obligations of Definition 689.2, the carrier transport of Definition 689.3, the bundle exhaustion of Definition 689.4, the StreamName readback of Definition 689.5, the inscription handoff of Definition 689.6, and the ledger non-escape theorem of Definition 689.7 jointly enumerate the five finite obligations of the* FiniteWindowObserver$^{\uparrow}$ *packet. The resulting surface exports carrier admission, classifier transport, checker replay and admission, downstream consumer handoff, and ledger non-escape, and it exports no private psychology, global observer identity, hidden agency predicate, unbounded history, or external semantic observer.*

Proof. Project the carrier $W = (O, F, I, K, D, H, C, P, N)$ from Definition 689.1. The projected rows are exactly the prior observation state, finite selected public window, inscription record, checker result, downstream interface, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 689.2. They list the five finite obligations: carrier admission, classifier transport, checker replay and admission, downstream handoff, and ledger non-escape.

Now use carrier transport, bundle exhaustion, StreamName readback, and inscription handoff. These theorems identify the same rows $O, F, I, K, D, H, C, P, N$ as the transported carrier surface, the finite window bundle, the StreamName-facing readback, and the inscription-event route.

Finally apply ledger non-escape, Definition 689.7. Since all cited rows belong to the same packet and no cited theorem exports private psychology, global observer identity, hidden agency predicates, unbounded history, or external semantic observation, the five obligations are exactly the obligation-closure surface for this packet. □

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverObligationClosureSurface`

Theorem 689.9 (FiniteWindowObserver obligation closure route)**.** *For every accepted* $\mathsf{FiniteWindowObserver}^{\uparrow}$ *carrier* $W = (O, F, I, K, D, H, C, P, N)$*, the carrier of Definition 689.1, the NameCert obligations of Definition 689.2, carrier transport of Definition 689.3, bundle exhaustion of Definition 689.4, StreamName window readback of Definition 689.5, inscription handoff of Definition 689.6, ledger non-escape of Definition 689.7, and the obligation-closure surface of Definition 689.8 form a finite induction route from rows* $O, F, I, K, D, H, C, P, N$ *to the five obligation rows: carrier admission, classifier transport, checker replay and admission, downstream consumer handoff, and ledger non-escape. No consumer can bypass that row order.*

Proof. Project the carrier $W = (O, F, I, K, D, H, C, P, N)$. The base row O supplies the prior observation state, and the selected window row F is the first public finite bundle available to any downstream read.

Induct over the displayed route $O \to F \to I \to K \to D$. At the F step, Definition 689.4 closes the selected finite bundle. At the I, K, D steps, Definition 689.5 and Definition 689.6 force the StreamName-facing and inscription-facing reads to consume the same finite window before any consumer interface is visible.

The structural rows preserve the induction. Carrier transport from Definition 689.3 keeps the route stable under hsame, while the displayed Cont replay row C, provenance row P, and local naming row N keep the same packet fixed throughout the read.

Finally apply the NameCert obligations and ledger non-escape. Theorems 689.2 and 689.7, together with the obligation-closure surface of Definition 689.8, identify the five obligation rows and exclude private psychology, global observer identity, hidden agency predicates, unbounded histories, and external semantic observation. Thus the finite route closes the obligations at the paper level, and a consumer cannot skip any row in the displayed order. □

Lean4 ✓ `BEDC.Derived.FiniteWindowObserverUp.FiniteWindowObserverObligationClosureRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowObserver}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteWindowObserver}^{\uparrow}$ packet is generated from a prior observation state, a finite selected public window, an inscription record, an accepted checker row, a downstream interface, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteWindowObserver}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 689.8 and Definition 689.9 give the scoped finite induction over rows $O, F, I, K, D, H, C, P, N$, binding carrier admission, classifier transport, checker replay and admission, downstream consumer handoff, and ledger non-escape over BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWindowObserver}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close private psychology, global observer identity, hidden agency predicates, unbounded observation histories, external semantic observation, Lean verification, or bridge checking.

Upgrade path: Public closure requires Lean formal targets for the same carrier, checker replay, downstream handoff, and ledger non-escape rows.

Chapter 690

A Concrete Naming Certificate for $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$

The $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ surface records the audit path that lets a downstream classifier or compiler consumer trust an erased term without receiving the proof object that produced it. It sits after $\mathsf{ProofTermErasureTrace}^{\uparrow}$ and before type-preserving consumer packets: the carrier keeps proof-term membership, erased-term readback, typing membership, erased replay, subject-reduction compatibility, transport, provenance, and local naming in one finite certificate. The classifier-level sibling chapter 707 consumes this audit route as the finite proof-erasure membership surface, so the audit packet is a real downstream dependency rather than an isolated ledger.

Definition 690.1 (Proof-term erasure audit route carrier)**.** *A* $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, E, T, C, R, S, H, Q, N)$$

with the following displayed rows:

1. *P is the proof-term row whose original type-checking membership is audited;*
2. *E is the erased-term row exported to downstream consumers;*
3. *T is the* Ext *typing row that admits the proof-term row and names the classifier membership being preserved;*
4. *C is the erasure* Cont *route from the proof-term row to the erased-term row;*
5. *R is the subject-reduction compatibility row tying the typing membership in T to the erased public read along C;*
6. *S is the accepted audit stamp row stating that the proof object has been consumed by the local checker and is not part of the exported evidence;*
7. *H records componentwise* hsame *transports for the proof-term row, erased-term row,* Ext *typing row, erasure route, subject-reduction row, audit stamp, package row, and local naming row;*
8. *Q is the* Pkg *provenance over* BHist, Ext, Cont, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{ProofTermErasureTrace}^{\uparrow}$, $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$, $\mathsf{TypeCheckingMembershipTrace}^{\uparrow}$, $\mathsf{TypePreservingCompiler}^{\uparrow}$, hsame, *and* $\mathsf{NameCert}$ *source packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureAuditRoute}^{\uparrow}}$ *row.*

The local classifier compares two accepted carriers only through the displayed proof row, erased row, typing row, erasure route, subject-reduction row, audit stamp, componentwise transports, provenance, and local naming row.

Theorem 690.2 (Proof-term erasure audit route NameCert obligations)**.** *For the carrier of Definition 690.1, the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureAuditRoute}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission exposes exactly* $A = (P, E, T, C, R, S, H, Q, N)$*;*
2. *proof-term membership reads the accepted typing evidence only from the displayed* Ext *row* T*;*
3. *erasure replay routes the exported erased row* E *only along the displayed* Cont *row* C*;*
4. *subject-reduction compatibility uses only the row* R *to relate the source membership in* T *to the erased public read;*
5. *audit-stamp exactness records in* S *that proof evidence was consumed locally and that no proof object is exported;*
6. *provenance and naming exactness bind every downstream read to* H, Q, N *and to the same displayed proof-term, erased-term, typing, erasure, subject-reduction, and audit-stamp rows.*

Proof. Carrier admission is the projection of the displayed finite tuple. The rows $P, E, T, C, R, S, H, Q, N$ are the only coordinates exposed by the local certificate.

The membership and replay obligations are read from T and C. The proof-term row enters the certificate through the classifier membership row T, and the erased-term row is exported only after the displayed erasure route C is replayed.

Subject-reduction compatibility is the role of R. It binds the source membership already named by T to the erased read reached through C, so a consumer cannot replace the route by host conversion or quotient term equality.

The audit-stamp and ledger obligations are then exhausted by S, H, Q, N. The stamp records local consumption of proof evidence, the transports replay the same rows componentwise, the package records the dependency surface, and the local naming row fixes the certificate boundary. □

Lean4 ✓ `BEDC.Derived.ProofTermErasureAuditRouteUp.ProofTermErasureAuditRoute namecert obligations`

Theorem 690.3 (Proof-term erasure audit route Ext-membership preservation)**.** *Every accepted* $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ *carrier preserves the* Ext *typing membership that justifies erased-term consumption. If a downstream consumer reads the erased row* E*, then the only source membership available to justify that read is the displayed typing row* T *for the proof-term row* P*, transported through* H *and packaged by* Q, N*.*

Proof. Project the accepted packet $A = (P, E, T, C, R, S, H, Q, N)$. The only typing-membership coordinate is T, and it is attached to the proof-term row P.

The erased row E is reached through the erasure route C and the compatibility row R. Thus the consumer's membership read is a transported reading of T, not a fresh classifier supplied by the host environment.

Finally repack the read through H, Q, N. Componentwise hsame preserves the displayed rows, package provenance records the same source certificates, and the local naming row keeps the consumer inside the audit-route boundary. □

Theorem 690.4 (Proof-term erasure audit route erased Cont replay)**.** *For every accepted* ProofTermErasureAuditRo *carrier, the erased public row is replayed only through the displayed erasure* Cont *route C and the subject-reduction compatibility row R. No consumer receives a host erasure function, a conversion oracle, or an unrecorded reduction route.*

Lean4 ✓ `BEDC.Derived.ProofTermErasureAuditRouteUp.ProofTermErasureAuditRoute erased cont replay`

Proof. Read the carrier inventory. The only route from the proof-term row P to the erased row E is the displayed continuation row C.

Subject-reduction compatibility is not an additional route; it is the row R that checks the same erased read against the typing membership in T. Therefore a consumer can replay erased evidence only by using C together with R.

Since the packet has no coordinate for a host erasure function, conversion oracle, or hidden reduction route, those objects cannot be exported by the local certificate. □

Theorem 690.5 (Proof-term erasure audit route non-escape ledger)**.** *Every accepted* ProofTermErasureAuditRoute$^{\uparrow}$ *carrier excludes proof-object escape. The exported surface contains the erased row E, the typing row T, the erasure route C, the compatibility row R, the audit stamp S, transports H, package provenance Q, and local naming N, but it contains no proof object, proof irrelevance principle, quotient syntax equality, host conversion relation, or ambient normalization oracle.*

Proof. Inspect the packet coordinates. The proof-term row P is visible only as the audited source of membership, while the exported computational row is E.

The stamp S records that proof evidence has been consumed locally. The rows T, C, R, H, Q, N then describe how membership, erasure, compatibility, transport, provenance, and naming are replayed without exporting the proof object itself.

No remaining coordinate can name proof irrelevance, quotient syntax equality, host conversion, or ambient normalization. The non-escape claim follows from the finite carrier inventory. □

Theorem 690.6 (Proof-term erasure audit route downstream consumer package)**.** *Every downstream classifier or type-preserving compiler consumer of an accepted* ProofTermErasureAuditRoute$^{\uparrow}$ *packet receives only the finite audit surface*

$$E, T, C, R, S, H, Q, N.$$

The consumer may use the erased term, preserved Ext *membership, displayed erasure replay, subject-reduction compatibility, audit stamp, transports, provenance, and local naming boundary, but it cannot recover the source proof object or replace the certificate by host type checking.*

Proof. Start from Ext-membership preservation. The consumer's typing read is fixed by T and transported through H, Q, N.

Next use erased Cont replay. The erased row E is exposed only along C and only with compatibility supplied by R.

Finally apply the non-escape ledger. The audit stamp S records local proof consumption, while the public package contains no coordinate for the source proof object as exported evidence and no coordinate for host type checking. Hence the downstream consumer package is exactly the displayed finite audit surface. □

Theorem 690.7 (Proof-term erasure audit route erased replay determinacy)**.** *Let A and A' be accepted* ProofTermErasureAuditRoute$^{\uparrow}$ *packets. Suppose the two erased replay reads have the*

same proof-term membership row, erased-term readback row, Ext *typing membership row, subject-reduction compatibility row, componentwise* hsame *transport, erased* Cont *replay route,* Pkg *provenance, and local* NameCert *row. Then every downstream classifier consumer reads the same erased replay surface from A and A'.*

The determined surface is the finite packet of proof-term membership, erased readback, typing membership, erased replay, subject-reduction compatibility, transport, provenance, and local naming rows. It exports no proof object, proof irrelevance principle, quotient syntax equality, host conversion relation, ambient normalization oracle, or host type-checking replacement.

Lean4 ✓ `BEDC.Derived.ProofTermErasureAuditRouteUp.ProofTermErasureAuditRoute erased replay determinacy`

Proof. First use Definition 690.3. The downstream consumer's membership read is forced to be the displayed typing row attached to the proof-term membership row, transported through the shared hsame, Pkg, and NameCert rows.

Next apply Definition 690.4. Since the two packets have the same erased readback, erased Cont replay route, and subject-reduction compatibility row, their public erased replay reaches the same consumer-visible row.

Use Definition 690.5 to exclude off-packet witnesses. The audit route has no coordinate for a proof object, proof irrelevance, quotient syntax equality, host conversion, ambient normalization, or host type checking, so no downstream classifier can separate the two reads by such data.

Finally repack with Definition 690.6. That package is exactly the shared erased row, typing row, erasure route, compatibility row, audit stamp, transports, provenance, and local naming boundary. Therefore every downstream classifier consumer sees the same erased replay surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTermErasureAuditRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ packet is generated from proof-term, erased-term, Ext typing, erasure Cont, subject-reduction compatibility, audit-stamp, hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofTermErasureAuditRoute}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 690.1, Definition 690.2, Definition 690.3, Definition 690.4, Definition 690.5, Definition 690.6, and Definition 690.7 define the erased replay audit surface exported to downstream classifier and compiler consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofTermErasureAuditRoute}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ProofTermErasureAuditRouteUp.ProofTermErasureAuditRoute\_erased\_replay\_determinacy`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host conversion, quotient term equality, proof irrelevance, normalization oracles, or compiler correctness outside the displayed audit-route packet.

Upgrade path: No further theory-axis upgrade is required for the erased replay audit surface; bridge checking remains a formal-axis task.

Chapter 691

A Concrete Naming Certificate for $\mathsf{EmptyBoundaryStep}^{\uparrow}$

The $\mathsf{EmptyBoundaryStep}^{\uparrow}$ packet realizes the one-step birth surface of Definition 9399.5 as a finite naming certificate. It records the source Hist row, the enacted mark, the resulting Hist row, and the displayed Ext fact that ties them together. The packet is local to that step: it does not choose marks for other histories, does not decide admissibility predicates, and does not convert local branch actualisation into a global choice principle.

Definition 691.1 (Empty-boundary step carrier)**.** *An* $\mathsf{EmptyBoundaryStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (H, M, R, E, T, C, P, N)$$

with the following displayed rows:

1. *H is the source* Hist *row whose empty boundary is filled;*
2. *M is the enacted mark row, one of the kernel marks b_0 or b_1;*
3. *R is the resulting* Hist *row produced by recording M above H;*
4. *E is the displayed* $\mathsf{Ext}(H, M, R)$ *row;*
5. *T records componentwise* hsame *transports for H, M, R, E and the local naming row;*
6. *C records the displayed* Cont *routes by which a consumer reads the source row, enacted mark, result row, extension row, and local name;*
7. *P is the* Pkg *provenance over* Hist, Ext, hsame, Cont, Pkg, NameCert, *and the Empty-boundary reading of Definition 9399.5;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{EmptyBoundaryStep}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows H, M, R, E, T, C, P, N. It has no coordinate for a hidden selector, a branch policy over unrelated histories, a truth oracle, a host equality on traces, or an accumulated ledger.

Lean4 ✓ `BEDC.Derived.EmptyBoundaryStepUp.EmptyBoundaryStepTasteGate single carrier alignment`

Theorem 691.2 (Empty-boundary step NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EmptyBoundaryStep}^{\uparrow}}$ *surface for the carrier of Definition 691.1 consists of five finite obligations:*

1. *carrier admission, requiring every accepted object to expose $S = (H, M, R, E, T, C, P, N)$;*

2. *source-boundary admission, requiring H to be the displayed* Hist *row whose empty boundary is filled;*
3. *enacted-step exactness, requiring M, R, and E to exhibit the same* $\mathsf{Ext}(H, M, R)$ *step;*
4. *transport-and-route stability, requiring all public reads to pass through the componentwise* hsame *row T and displayed* Cont *row C;*
5. *non-escape, requiring the packet to export no hidden selector, no branch policy for unrelated histories, no truth oracle, no host trace equality, and no accumulated ledger.*

Proof. Carrier admission is projection from the finite tuple in Definition 691.1. The source history, enacted mark, result history, extension row, transport row, continuation row, provenance row, and naming row are all displayed coordinates.

Source-boundary admission reads H before the next mark is consumed. The empty boundary is therefore the slot above this source row, matching the reading of Definition 9399.1, rather than a reset to e.

Enacted-step exactness ties M, R, and E together. Since E is the displayed $\mathsf{Ext}(H, M, R)$ row, a consumer cannot replace the enacted mark by an unrecorded branch while keeping the same certificate.

Transport-and-route stability uses T and C. These rows move and expose the displayed step data componentwise, but they do not add another source of branch information.

Finally P and N delimit the packet as local NameCert data. Because the carrier has no coordinate for a hidden selector, unrelated branch policy, truth oracle, host trace equality, or accumulated ledger, the packet cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.EmptyBoundaryStepUp.EmptyBoundaryStepCarrier namecert obligations`

Theorem 691.3 (Empty-boundary step local actualisation)**.** *Let $S = (H, M, R, E, T, C, P, N)$ be an accepted* $\mathsf{EmptyBoundaryStep}^{\uparrow}$ *carrier. Every public read of the step factors through the finite route*

$$H, M, R, E, T, C, P, N.$$

The read exposes exactly one enacted empty-boundary step and does not supply a function selecting marks for any other Hist *row.*

Proof. Project the accepted carrier to its displayed rows. The only branch data in the packet are the enacted mark M, the result row R, and the extension row E attached to the source H.

Read the step through E. The row E records $\mathsf{Ext}(H, M, R)$, so the public route identifies the enacted step by inspecting the displayed extension fact rather than by consulting an external chooser.

The transport row T and continuation row C keep the same data visible to consumers. Package provenance and the local NameCert row close the route without adding a coordinate for unrelated histories.

Thus every public read factors through H, M, R, E, T, C, P, N and remains a single local actualisation of the empty boundary above H. □

Theorem 691.4 (Empty-boundary step BHist Ext dependency link)**.** *Let*

$$S = (H, M, R, E, T, C, P, N)$$

be an accepted $\mathsf{EmptyBoundaryStep}^{\uparrow}$ *carrier. The displayed row* E *supplies the local* $\mathsf{Ext}(H, M, R)$ *step consumed by chapter 516 and by chapter 685. This dependency is horizontal and local: it exports the source history, enacted mark, result history, extension row, transport, route, provenance, and naming rows, but it still refuses a global selector, a branch policy for unrelated histories, a truth oracle, host trace equality, or an accumulated ledger.*

Proof. Project the carrier S. The rows H, M, R, E, T, C, P, N are exactly the source history, enacted mark, result history, displayed extension fact, transport, continuation, provenance, and local naming rows.

Apply Definition 691.3. Every public read of the enacted step factors through the displayed $\mathsf{Ext}(H, M, R)$ row and exposes only one local empty-boundary step.

The Empty-fable ledger and Empty-boundary selector chapters consume precisely such local BHist extension rows when they record a witnessed step or compare selector-step behavior. Since S has no coordinate for a global selector, unrelated branch policy, truth oracle, host trace equality, or accumulated ledger, the dependency link supplies only the local Ext step. □

Closure status for $\mathsf{NameCert}_{\mathsf{EmptyBoundaryStep}^{\uparrow}}$.

Constructive story: An $\mathsf{EmptyBoundaryStep}^{\uparrow}$ packet is generated from a source Hist row, enacted mark, result Hist row, displayed Ext row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmptyBoundaryStep}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 691.1, Definition 691.2, and Definition 691.3 bind one local empty-boundary step as a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EmptyBoundaryStep}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, global choice, branch policies for unrelated histories, truth oracles, host trace equality, accumulated ledgers, or a public standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, enacted-step exactness, transport-and-route stability, non-escape, and local actualisation for the same finite empty-boundary step packet.

Chapter 692

A Concrete Naming Certificate for $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$

The $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$ interface packages the finite carry-window budget behind the displayed AxisZeckendorf carry classifier. It records the carry source, carry target, normal-form target, Fibonacci value-budget ledger, non-hsame boundary, componentwise transports, displayed continuation routes, package provenance, and a local naming row. The packet is a finite certificate for the carry-budget surface; it is not history sameness, a kernel rewrite, a host arithmetic quotient, or a global normalization algorithm.

The budget carrier consumes the Zeckendorf normal-form rows of chapter 278, the local carry row of chapter 544, and the classifier boundary of chapter 567 before adding the value-budget ledger and non-hsame boundary. Thus the packet sits on the concrete carry and normal-form lattice rather than on a kernel-only naming surface.

Definition 692.1 (AxisZeckendorf carry-budget carrier)**.** *An* $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (h, k, G, N_k, V, Q, H, C, P, N)$$

with the following visible coordinates:

1. *h is the displayed carry-source history row carrying the public window* `word_011`*;*
2. *k is the displayed carry-target history row carrying the public window* `word_100`*;*
3. *G is the local* $\mathsf{ZCarry}^{\uparrow}$ *classifier row reading* $\mathsf{ZCarry}(h, k)$ *through the displayed carry generator;*
4. *N_k is the normal-form target row stating that k is admitted by the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *normal-form classifier;*
5. *V is the value-budget ledger row recording the finite Fibonacci value read for h, the finite Fibonacci value read for k, and the equality of those two displayed sums;*
6. *Q is the non-*hsame *boundary row recording that the carry relation between h and k is not a history-sameness witness;*
7. *H stores the componentwise* hsame *transports that preserve the source row, target row, carry row, normal-form target row, value-budget ledger, boundary row, route row, provenance row, and local naming row;*

8. *C stores the displayed* Cont *routes for carry-source exposure, carry-target exposure, normal-form target reading, value-budget reading, non-*hsame *boundary reading, provenance reading, and local naming;*
9. *P is the* Pkg *provenance over* BHist, ZCarry$^{\uparrow}$, ZeckendorfNormal$^{\uparrow}$, *Fibonacci value-budget rows,* hsame, Cont, *and local* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $h, k, G, N_k, V, Q, H, C, P, N$. It exports no source-target hsame *proof, no kernel rewrite, no hidden arithmetic quotient, and no unbounded normalization process.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCarryBudgetUp.AxisZeckendorfCarryBudgetCarrier full packet semantic namecert`

Theorem 692.2 (AxisZeckendorf carry-budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}}$ *surface for Definition 692.1 consists of five finite obligations:*

1. *carrier admission for the finite packet $B = (h, k, G, N_k, V, Q, H, C, P, N)$;*
2. *carry-window coverage, stating that the only admitted carry source and target windows are the displayed* `word_011` *and* `word_100` *rows read through G;*
3. *value-budget preservation, stating that every public value-preservation read factors through the finite ledger V;*
4. *non-*hsame *boundary, stating that Q keeps carry-relatedness separate from* $\mathsf{hsame}(h, k)$ *under all displayed transports;*
5. *ledger non-escape, stating that the rows H, C, P, N route and name exactly this packet and do not export a kernel rewrite, a host arithmetic quotient, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCarryBudgetUp.AxisZeckendorfCarryBudgetNamecertObligations`

Proof. Carrier admission is the tuple in Definition 692.1. It lists the carry source, carry target, carry classifier, normal-form target, value-budget ledger, non-hsame boundary, transports, routes, provenance, and local naming row inside one finite BHist packet.

Carry-window coverage is read from h, k, G, N_k. The carry row G fixes the public source and target windows, and the normal-form target row N_k records that the target side is the admitted normal-form window.

Value-budget preservation is the role of V. Every public value read of the carry step reaches the two displayed Fibonacci sums and their equality only through that finite ledger coordinate.

The boundary and non-escape clauses are the rows Q, H, C, P, N. The row Q records that the carry is not $\mathsf{hsame}(h, k)$, while H transports only the displayed components. The route, provenance, and naming rows repack the same finite surface, so no excluded object is available to a consumer. □

Theorem 692.3 (AxisZeckendorf carry-budget classifier realization)**.** *Every consumer-visible carry-budget read of an accepted* $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$ *carrier factors through*

$$h, k, G, N_k, V, Q, H, C, P, N.$$

*Thus the carry classifier can expose finite carry-window coverage, normal-form target reading, value-ledger preservation, non-*hsame *separation, transport stability, route replay, provenance, and local naming without treating carry-relatedness as history sameness.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCarryBudgetUp.AxisZeckendorfCarryBudget classifier realization`

Proof. Start with the finite carrier of Definition 692.1. The rows h, k, G, N_k expose exactly the displayed carry source, carry target, carry witness, and normal-form target.

Add the budget and boundary rows. The row V carries the finite value-preservation ledger, while Q keeps the non-hsame boundary visible for the same source and target.

Finally apply the structural rows H, C, P, N. They transport, route, package, and name only the displayed packet. Therefore every consumer-facing carry-budget read remains inside the listed rows and cannot become a source-target hsame proof or a kernel rewrite. □

Theorem 692.4 (AxisZeckendorf carry-budget scope)**.** *The scoped* $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$ *surface is exactly the kernel-visible packet*

$$B = (h, k, G, N_k, V, Q, H, C, P, N)$$

of Definition 692.1. It exposes the BHist *source and target rows, the* $\mathsf{ZCarry}^{\uparrow}$ *carry row, the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *target row, the value-budget ledger, the non-*hsame *boundary, the* hsame *transport rows, the* Cont *replay rows, the* Pkg *provenance row, and the local* NameCert *row, and it exports no source-target* hsame *proof, host arithmetic quotient, kernel rewrite, or global normalizer.*

Lean4 ✓ `BEDC.Derived.AxisZeckendorfCarryBudgetUp.AxisZeckendorfCarryBudgetTasteGate single carrier alignment`

Proof. Carrier scope is the displayed tuple of Definition 692.1. Its first four rows bind the BHist carry source and target through $\mathsf{ZCarry}^{\uparrow}$ and $\mathsf{ZeckendorfNormal}^{\uparrow}$, while the value-budget ledger records the finite Fibonacci equality that the public carry-budget read may use.

The boundary and structural rows keep the scope kernel-local. The row Q refuses source-target hsame, the row H transports only displayed coordinates, and the row C replays only the named Cont routes.

The provenance and naming rows close the packet. The Pkg row P and the local NameCert row N name the same finite carrier, so the chapter scope is the listed ledger surface rather than a quotient, rewrite, or normalization theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$ packet is generated from the displayed carry source, carry target, normal-form target row, finite Fibonacci value-budget ledger, non-hsame boundary, componentwise transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}}$, publicClosed)

Scope closed: Definition 692.1, Definition 692.2, Definition 692.3, and Definition 692.4 bind the public carry-budget carrier, value ledger, non-hsame boundary, route, provenance, local naming, classifier realization, and kernel-scope surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, a bridge certificate, a global Zeckendorf normalizer, a kernel rewrite, host arithmetic quotienting, or source-target hsame.

Upgrade path: Formal verification requires checked carrier admission, carry-window coverage, value-budget preservation, non-hsame boundary preservation, route stability, provenance, local naming, and ledger non-escape for the same finite packet.

Chapter 693

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$

The $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ surface packages the finite observation-window route used by transported stationary diagonal seals as one naming certificate. It receives a RatUp seed row, the constant $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ seal readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, and the $\mathsf{Real}^{\uparrow}$ finite-window seal as one displayed packet, so downstream real-completion consumers read a shared finite route rather than a selected limit, quotient stream, or ambient completeness assertion.

This chapter consumes the concrete route of chapter 126, chapter 301, chapter 291, and chapter 38. The $\mathsf{StreamName}^{\uparrow}$ chapter supplies the finite window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the seal readback, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the observation ledger, and $\mathsf{Real}^{\uparrow}$ supplies the finite-window seal used by the fused packet.

Definition 693.1 (Regular Cauchy window fusion carrier)**.** *A* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, W, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the RatUp seed classifier row for the stationary rational source;*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *constant-window classifier read over the displayed observation window;*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row obtained from the same finite window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger carried by that window;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier and seal row reached from W, S, D;*
6. *H records componentwise* hsame *transports for R, W, S, D, E and for the package and local naming rows;*
7. *C records the displayed* Cont *routes by which consumers first read the seed row, then the shared window, then the dyadic and regular-sequence rows, and finally the real seal row;*
8. *P is the* Pkg *provenance over RatUp,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient stream equality, real trichotomy, host completeness, or an unrecorded RealUp comparison row.

Theorem 693.2 (Regular Cauchy window fusion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusio}}$ *surface for the carrier of Definition 693.1 consists of carrier admission for*

$$F = (R, W, S, D, E, H, C, P, N),$$

seed-to-window exposure, $\mathsf{RegSeqRat}^{\uparrow}$ *seal readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation-ledger exposure,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *stability, displayed* Cont *locality,* Pkg *provenance, and non-escape from selected limits, quotient streams, trichotomy, ambient completeness, and hidden RealUp comparison data.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionNameCertObligations`

Proof. Carrier admission is projection from Definition 693.1. The rows $R, W, S, D, E, H, C, P, N$ are the only rows admitted to the packet.

The observation obligations are read in order. The RatUp seed row R feeds the finite $\mathsf{StreamName}^{\uparrow}$ window W; the same window supplies the $\mathsf{RegSeqRat}^{\uparrow}$ seal row S and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D.

The Real-facing obligation is the row E. It records the finite-window $\mathsf{Real}^{\uparrow}$ seal reached from W, S, D, while H and C preserve and route that same displayed row without adding an analytic carrier.

Finally P and N bind the packet to its package provenance and local naming row. Since no carrier coordinate names selected limits, quotient streams, trichotomy, ambient completeness, or a hidden RealUp comparison, none can be exported by the certificate. □

Theorem 693.3 (Regular Cauchy window fusion obligation inventory)**.** *For an accepted* RegularCauchyWindowFusi *carrier*

$$F = (R, W, S, D, E, H, C, P, N),$$

the obligation inventory consists of the following five finite duties:

1. *carrier exposure for the displayed packet* $F = (R, W, S, D, E, H, C, P, N)$*;*
2. *common finite-window exactness across the RatUp seed row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger;*
3. $\mathsf{Real}^{\uparrow}$ *seal handoff through the shared rows* R, W, S, D, E*;*
4. *componentwise* hsame *stability and displayed* Cont *locality through* H *and* C*;*
5. *non-escape from selected limits, quotient streams, trichotomy, ambient completeness, and hidden* $\mathsf{Real}^{\uparrow}$ *comparison rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionObligationInventory`

Proof. Carrier exposure is the definition of the packet in Definition 693.1. The existing NameCert obligation theorem Definition 693.2 records the same rows as the local certificate surface.

For common finite-window exactness, apply Definition 693.4. It forces the RatUp seed, stream window, regular-sequence readback, and dyadic ledger to be consumed through one displayed observation window.

For the Real-facing duty, apply Definition 693.5. The seal row E is reachable only after R, W, S, D and the support rows have been read.

Finally use Definition 693.8. It repacks $R, W, S, D, E, H, C, P, N$ for downstream consumers and excludes selected limits, quotient streams, trichotomy, ambient completeness, and hidden comparison rows from the exported package. □

Theorem 693.4 (Regular Cauchy window fusion shared-window handoff)**.** *For every accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier, the RatUp seed row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger are all reads of the same displayed window. Any downstream consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal must first consume the shared route*

$$R, W, S, D, H, C, P, N.$$

Proof. Project the carrier rows. The only seed row is R, the only stream-window row is W, the only regular-sequence seal row is S, and the only dyadic observation ledger is D.

Use the transported stationary diagonal finite-window seal exhaustion as the source route. It identifies the RatUp seed classifier, the constant $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ seal consumption row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger as four readings of one certified finite window.

The rows H, C, P, N then preserve, route, package, and name that shared window. Therefore a downstream consumer cannot reach the real seal without first reading the displayed shared-window packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionSharedWindowHandoff`

Theorem 693.5 (Regular Cauchy window fusion Real seal non-escape)**.** *For an accepted* RegularCauchyWindowFusi *carrier*

$$F = (R, W, S, D, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *finite-window seal row* E *is reachable only through the seed row* R*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the support rows* H, C, P, N*. The seal exports no selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent real carrier, or unrecorded comparison row.*

Proof. Start from the shared-window handoff. It forces every route to the seal to pass through R, W, S, D before the Real-facing row is read.

Now read the real seal coordinate. The carrier contains a single $\mathsf{Real}^{\uparrow}$ finite-window row E, and the displayed Cont route exposes it only after the seed, stream-window, regular-sequence, and dyadic rows have been consumed.

Inspect the remaining coordinates. The support rows H, C, P, N preserve, route, package, and name the same finite packet, while no coordinate names selected limits, quotient streams, trichotomy, ambient completeness, an independent real carrier, or a hidden comparison. The Real seal therefore cannot escape the finite packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionRealSealNonescape`

Theorem 693.6 (Regular Cauchy window fusion selector-budget compatibility)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *consumer of the accepted fusion packet must use the same displayed finite window for the RatUp seed row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the support rows* H, C, P, N*. Changing the selector or budget before the row* E *is read exits the carrier and is not a read of the accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packet.*

Proof. Apply the shared-window handoff of Definition 693.4. It binds the seed row, stream window, regular-sequence readback, and dyadic ledger to one displayed finite observation window.

Next apply the Real seal non-escape theorem Definition 693.5. The only route to E first consumes R, W, S, D and then remains under the support rows H, C, P, N.

The sibling inputs named in the introduction supply exactly those rows: $\mathsf{StreamName}^{\uparrow}$ supplies the window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the readback, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the ledger, and $\mathsf{Real}^{\uparrow}$ supplies the finite-window seal. A selector or budget change before E would require an additional carrier coordinate, but the accepted packet has only $R, W, S, D, E, H, C, P, N$. Hence such a read exits the carrier rather than consuming the accepted fusion packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionSelectorBudgetCompatibility`

Theorem 693.7 (Regular Cauchy window fusion tail handoff)**.** *Any downstream* $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packet must use the same finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* S*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and support rows* H, C, P, N *before reading the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. A tail-facing consumer cannot replace the shared window by a selected limit, quotient stream, or independent completeness witness.*

Proof. Begin with the shared-window handoff, Definition 693.4. It binds the RatUp seed row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger to one finite observation packet.

Apply Real seal non-escape and selector-budget compatibility, Definition 693.5 and Definition 693.6. They force every route to E through R, W, S, D, H, C, P, N and reject any budget or selector change before the seal is read.

The tail-facing consumer therefore inherits the same handoff. Whether the downstream read is $\mathsf{RegularSequenceTailMetric}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, it can consume only the displayed rows of the accepted fusion packet before E; selected limits, quotient streams, and independent completeness witnesses are not carrier coordinates. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionTailHandoff`

Theorem 693.8 (Regular Cauchy window fusion downstream package)**.** *The carrier of Definition 693.1, the local NameCert obligations, the shared-window handoff, and the Real seal non-escape theorem assemble a downstream package for* $\mathsf{RealCompleteness}^{\uparrow}$*,* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$*,* $\mathsf{CauchyModulusComparison}^{\uparrow}$*, and* $\mathsf{DiagonalCofinalTail}^{\uparrow}$ *consumers. The package exposes exactly*

$$R, W, S, D, E, H, C, P, N$$

over RatUp, $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. First project the carrier and NameCert surface. They provide the RatUp seed, the shared stream window, the regular-sequence seal readback, the dyadic ledger, the real seal, and the support rows.

Next use the shared-window handoff. It binds the seed, stream, regular-sequence, and dyadic rows to one finite observation window, so downstream consumers receive a single route rather than separate analytic witnesses.

Finally apply the Real seal non-escape theorem. The package can export only $R, W, S, D, E, H, C, P, N$, and the listed downstream consumers read those rows through their existing finite-window routes. □

Lean4 ✓ BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionDownstreamPackage

Theorem 693.9 (Regular Cauchy window fusion budget meet)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier. If two downstream reads consume the same shared finite window, one through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* S *and one through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger* D*, then the reads meet at the same* $\mathsf{Real}^{\uparrow}$ *seal row* E *through the displayed rows* H, C, P, N*. The meet exports the finite L10 handoff packet*

$$R, W, S, D, E, H, C, P, N$$

and exports no selected limit, quotient stream, trichotomy decision, ambient completeness proof, or hidden comparison row.

Lean4 ✓ BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionCarrier budget meet

Proof. Apply Definition 693.4. It places the regular-sequence readback row S and the dyadic observation ledger D over one displayed finite window together with the RatUp seed row R and the stream-window row W.

Next apply Definition 693.5. The only real-facing row reachable from that shared window is E, and it is read only after R, W, S, D under the support rows H, C, P, N.

Finally apply Definition 693.8. The package repacks exactly $R, W, S, D, E, H, C, P, N$ for downstream $\mathsf{Real}^{\uparrow}$ and completion-facing consumers. Since the accepted carrier has no coordinate for selected limits, quotient streams, trichotomy, ambient completeness, or hidden comparison data, the two downstream reads meet at the displayed seal row rather than at an external analytic object. □

Theorem 693.10 (Regular Cauchy window fusion scoped kernel route)**.** *Every accepted* RegularCauchyWindowFusio *read used by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer is scoped by the four concrete L10 rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *together with the structural rows* H, C, P, N*. For*

$$F = (R, W, S, D, E, H, C, P, N),$$

the read cannot replace these rows by quotient stream equality, a selected limit, trichotomy, ambient completeness, or a hidden $\mathsf{Real}^{\uparrow}$ *comparison.*

Proof. Project the carrier $F = (R, W, S, D, E, H, C, P, N)$. The rows W, S, D, E are respectively the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ finite-window seal; H, C, P, N are the only structural rows available to transport, replay, source, and name them.

Apply Definition 693.4. It forces R, W, S, D to be read as one shared finite observation window before any Real-facing seal is reached.

Apply Definition 693.5. The seal row E is reachable only through that shared finite route and exports no selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent real carrier, or unrecorded comparison row.

Apply Definition 693.8 and Definition 693.9. Downstream consumers receive exactly $R, W, S, D, E, H, C, P, N$, so no excluded coordinate is available inside the accepted packet. □

Lean4 ✓ BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionScopedKernelRoute

Theorem 693.11 (Regular Cauchy window fusion finite-window exactness)**.** *For every accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier*

$$F = (R, W, S, D, E, H, C, P, N),$$

the displayed rows expose exactly one finite observation-window route from the RatUp seed through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Every* $\mathsf{Real}^{\uparrow}$*- or completion-facing consumer of* F *must use this route before reading the real seal* E*, and the packet contains no second route through selected limits, quotient stream equality, trichotomy, ambient completeness, independent real carriers, or hidden comparison rows.*

Proof. Start with Definition 693.4. It binds the RatUp seed row R, stream window W, regular readback S, and dyadic ledger D to one displayed finite observation window.

Apply Definition 693.5. The real seal E is reachable only after that shared route and under the structural rows H, C, P, N.

Use Definition 693.9. Any two downstream reads that pass through S and D meet at the same seal row E, so the regular and dyadic sides cannot split into distinct completion-facing routes.

Finally apply Definition 693.10. It identifies the scoped kernel route with W, S, D, E, H, C, P, N and excludes quotient stream equality, selected limits, trichotomy, ambient completeness, and hidden $\mathsf{Real}^{\uparrow}$ comparisons. Therefore the displayed packet exhausts the finite Real-facing window. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionFiniteWindowExactness`

Theorem 693.12 (RegularCauchyWindowFusion tail-window coverage)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing consumer of* F *must expose the diagonal cofinal tail selected by the stationary window route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* S*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before consuming the fused window. The exposed route is exactly the finite packet*

$$R, W, S, D, E, H, C, P, N,$$

so no consumer may replace the tail-window route by a selected subsequence, quotient stream equality, or ambient Real completeness.

Proof. Project the carrier rows. The only schedule row is W, the only regular-rational readback is S, the only dyadic tolerance ledger is D, and the only terminal Real-facing seal is E.

Apply Definition 693.11. It identifies the route from the RatUp seed through W, S, D, E as the unique finite observation-window path available to a $\mathsf{Real}^{\uparrow}$- or completion-facing consumer.

The diagonal cofinal tail is therefore consumed only through the stationary window route already displayed by W and routed by C. The support rows H, P, N transport, source, and name the same packet without adding a second selector.

Consequently the consumer remains inside $R, W, S, D, E, H, C, P, N$. Selected subsequences, quotient stream equality, and ambient Real completeness are not carrier coordinates, so none can be used before the fused window is consumed. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionTailWindowCoverage`

Theorem 693.13 (Regular Cauchy window fusion tail stability)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

and

$$F' = (R, W', S', D, E, H', C', P', N')$$

be accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packets with the same* $\mathsf{Rat}^{\uparrow}$ *seed row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and support rows for transport, replay, provenance, and naming. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window* (W', S') *is displayed as a cofinal tail of* (W, S) *inside the accepted packet data. Then* F *and* F' *export the same finite* $\mathsf{Real}^{\uparrow}$*-facing handoff through* E*, and neither packet exports a selected limit, quotient stream, independent Real carrier, trichotomy decision, ambient completeness proof, or hidden comparison row.*

Proof. Start with Definition 693.4. It binds the seed, stream window, regular readback, dyadic ledger, and Real seal into one finite observation route for each accepted packet.

Apply Definition 693.11 to F. The only Real-facing route from the shared $\mathsf{Rat}^{\uparrow}$ seed through W, S, D reaches the displayed seal E.

Apply Definition 693.12 to F'. Because (W', S') is a displayed cofinal tail of (W, S) and the ledger D and seal E are shared, the tail packet is consumed through the same finite route rather than through a second selector.

Finally use Definition 693.5. The shared Real seal exports no selected limit, quotient stream equality, independent Real carrier, trichotomy decision, ambient completeness proof, or hidden comparison row. The structural rows of both packets only transport, replay, source, and name that handoff, so replacing the displayed window by its cofinal tail preserves the exported $\mathsf{Real}^{\uparrow}$-facing surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionTailStability`

Theorem 693.14 (RegularCauchyWindowFusion tail-budget meet)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing tail-window consumer and any budget-meet consumer read the same finite packet* $R, W, S, D, E, H, C, P, N$*. The tail selected by* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* S*, the dyadic ledger* D*, and the terminal seal* E *cannot split into separate selected-subsequence, quotient-stream, or ambient-completeness routes.*

Proof. First apply Definition 693.11. It identifies $R, W, S, D, E, H, C, P, N$ as the unique finite observation-window route available to any $\mathsf{Real}^{\uparrow}$- or completion-facing read.

Next use Definition 693.9. It forces the regular readback row S and the dyadic ledger row D to meet at the same terminal seal E through H, C, P, N.

Then apply Definition 693.10. The scoped route is exactly the L10 packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the structural rows.

Finally use Definition 693.12. The tail selected by W is consumed through that same displayed packet, so a tail-window consumer and a budget-meet consumer cannot factor through different selected-subsequence, quotient-stream, or ambient-completeness routes. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionTailBudgetMeet`

Theorem 693.15 (Regular Cauchy window fusion cofinal-tail closure)**.** *Let*

$$F = (R, W, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packet whose shared window route satisfies tail-window coverage and tail-budget meet. Then* F *exports a single cofinal-tail closure surface through the rows* $R, W, S, D, E, H, C, P, N$*. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer must pass through that same surface before reading the terminal seal* E*.*

The closure surface exports no selected limits, quotient stream equality, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with tail-window coverage, Definition 693.12. It forces any displayed cofinal tail of the shared window route to remain inside the same finite observation packet rather than producing a separate selected subsequence.

Next apply the tail-budget meet theorem Definition 693.14. The tail-window consumer and the budget-meet consumer both read $R, W, S, D, E, H, C, P, N$, so the regular readback row S and dyadic ledger row D meet before the terminal seal E is used.

Use finite-window exactness, Definition 693.11, to identify this packet as the only finite route from the RatUp seed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply Real seal nonescape, Definition 693.5. The seal row E cannot be read through selected limits, quotient stream equality, trichotomy, or ambient completeness, and the structural rows only transport, replay, source, and name the same route. Hence every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer must pass through the single cofinal-tail closure surface before reading the terminal seal. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionCofinalTailClosure`

Theorem 693.16 (Regular Cauchy window fusion kernel scope)**.** *Every accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *read is scoped by the concrete* BHist *packet*

$$F = (R, W, S, D, E, H, C, P, N),$$

the componentwise hsame *row* H*, displayed* Cont *route* C*,* Pkg *provenance row* P*, local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFu}}$ *row* N*, and the four L10 rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The accepted read cannot be replaced by a selected limit, quotient stream equality, trichotomy, ambient completeness, or hidden* $\mathsf{Real}^{\uparrow}$ *comparison.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionKernelScope`

Proof. Begin with the shared-window handoff of Definition 693.4. It binds the seed row R, the $\mathsf{StreamName}^{\uparrow}$ row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback S, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D to one displayed finite observation window.

Next apply the Real seal non-escape theorem Definition 693.5. The $\mathsf{Real}^{\uparrow}$ row E is reachable only after R, W, S, D and remains under the support rows H, C, P, N.

Use the downstream package and budget-meet theorems, Definition 693.8 and Definition 693.9. They repack exactly $R, W, S, D, E, H, C, P, N$ for completion-facing consumers and force regular and dyadic reads to meet at the displayed seal row.

Finally combine finite-window exactness and tail-window coverage, Definition 693.11 and Definition 693.12. These blocks leave no carrier coordinate from which a selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, or hidden $\mathsf{Real}^{\uparrow}$ comparison could be read. Hence every accepted fusion read remains inside the concrete kernel scope named above. □

Theorem 693.17 (Regular Cauchy window fusion scoped row exhaustion)**.** *For every accepted finite L10* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packet*

$$F = (R, W, S, D, E, H, C, P, N),$$

the scoped public surface exposes exactly the RatUp seed row R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the componentwise* hsame *transport row* H*, the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$ *row* N*. No selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent Real carrier, hidden comparison, or external downstream row can be consumed outside those displayed rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionScopedRowExhaustion`

Proof. Begin with the shared-window handoff and finite-window exactness. Theorems 693.4 and 693.11 identify R, W, S, D, E as one finite observation-window route from the RatUp seed through the stream, regular-sequence, dyadic, and Real-facing rows.

Next apply the Real seal non-escape and selector-budget compatibility results. Theorems 693.5 and 693.6 force any route to E to stay under R, W, S, D, H, C, P, N and rule out selector or budget changes before the displayed seal is read.

Now use the downstream package and budget meet theorems. Theorems 693.8 and 693.9 repack exactly $R, W, S, D, E, H, C, P, N$ for Real and completion-facing consumers, and they make the regular readback and dyadic ledger meet at the same displayed seal row.

Finally combine the scoped kernel route, finite-window exactness, tail-window coverage, and tail-budget meet. Theorems 693.10, 693.11, 693.12, and 693.14 leave no carrier coordinate outside $R, W, S, D, E, H, C, P, N$ from which a selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent Real carrier, hidden comparison, or external downstream row could be read. Hence the finite L10 fusion packet is scoped exactly by those rows. □

Theorem 693.18 (Regular Cauchy window fusion scoped route)**.** *Every accepted* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *consumer factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*, RatUp,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyModulusComparison}^{\uparrow}$*,* $\mathsf{DiagonalCofinalTail}^{\uparrow}$*, and* $\mathsf{StationaryDiagonalWindowRoute}^{\uparrow}$ *rows before the* $\mathsf{Real}^{\uparrow}$ *seal is readable. For*

$$F = (R, W, S, D, E, H, C, P, N),$$

the route is the concrete kernel path from the RatUp seed R *through the stationary stream window* W*, regular readback* S*, dyadic ledger* D*, terminal seal* E*, and structural rows* H, C, P, N*; it exports no selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent Real carrier, or hidden comparison row.*

Proof. Begin with the carrier and NameCert obligations, Definition 693.1 and Definition 693.2. They admit only the packet $R, W, S, D, E, H, C, P, N$ and bind the RatUp seed, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming rows.

Use the shared-window handoff and Real seal non-escape theorems, Definition 693.4 and Definition 693.5. They force the stationary window, dyadic ledger, regular readback, and Real-facing row to be consumed in order before the seal is readable.

Apply selector-budget compatibility and the downstream package theorem, Definition 693.6 and Definition 693.8. These place the $\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{DiagonalCofinalTail}^{\uparrow}$, and StationaryDiagonalWindowF

consumers on the same finite row sequence and prevent a budget or selector change from leaving the carrier silently.

Finally combine the budget meet and scoped row exhaustion results, Definition 693.9 and Definition 693.17. They exhaust the available rows before the $\mathsf{Real}^{\uparrow}$ seal is read, so every accepted consumer factors through the displayed kernel path. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionScopedRoute`

Theorem 693.19 (Regular Cauchy window fusion public export)**.** *The* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *packet exports a public finite L10 route for* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers. The exported route is exactly the displayed packet*

$$R, W, S, D, E, H, C, P, N$$

over the RatUp seed, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$ *row. A consumer that reads this public surface receives no selected limit, quotient stream equality, trichotomy decision, ambient completeness proof, independent Real carrier, hidden comparison row, or external downstream row.*

Proof. Begin with Definition 693.18. It places every accepted consumer on the concrete kernel path from the RatUp seed through the stationary stream window, regular readback, dyadic ledger, terminal seal, and structural rows.

Apply Definition 693.17. It identifies the public row inventory with $R, W, S, D, E, H, C, P, N$ and rules out any row outside that packet.

Finally use Definition 693.5. The terminal seal is reachable only through the displayed finite route and exports none of the excluded analytic or comparison coordinates. Hence the scoped route is a public export surface for exactly the stated L10 consumers. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.RegularCauchyWindowFusionPublicExport`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ packet is generated from a RatUp seed classifier, a finite $\mathsf{StreamName}^{\uparrow}$ constant window, a $\mathsf{RegSeqRat}^{\uparrow}$ seal readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, a $\mathsf{Real}^{\uparrow}$ finite-window seal, displayed hsame transports, Cont routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.RegularCauchyWindowFusionUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$, publicClosed)

Scope closed: Definition 693.19 exports the displayed DyadicRatCore, RatUp, StreamName, RegSeqRat, Real, CauchyModulusComparison, DiagonalCofinalTail, and StationaryDiagonalWindowRoute rows as the public finite L10 route before the Real seal is readable.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyWindowFusion}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient streams, selected limits, real trichotomy, bridge status, Lean verification, or ambient completeness.

Upgrade path: Bridged closure requires a checked standard bridge for the public finite L10 route without enlarging the displayed rows.

Chapter 694

A Concrete Naming Certificate for $\mathsf{BranchObstruction}^{\uparrow}$

The $\mathsf{BranchObstruction}^{\uparrow}$ interface isolates the finite ledger row used when an analytic-continuation request leaves the displayed local source, overlap, operation, and output packet. It turns the branch-sensitive boundary of Definition 9341.6 into a concrete BHist packet: a consumer can inspect where the request was recorded, which continuation socket supplied the local rows, and which routes preserve the ledger, but cannot extract a branch selector, quotient-of-germs equality, monodromy theorem, or new holomorphic output.

Definition 694.1 (BranchObstruction carrier)**.** *A* $\mathsf{BranchObstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, B, H, C, P, N)$$

with the following visible coordinates:

1. *S is an accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *row displaying the two* $\mathsf{Holomorphic}^{\uparrow}$ *source rows, overlap witness, operation row, output row, branch-obstruction ledger, componentwise transports, continuation routes, package provenance, and local naming row;*
2. *B is the branch-obstruction ledger coordinate selected from S;*
3. *H records componentwise* hsame *transport for S, B, the visible routes, the provenance row, and the local naming row;*
4. *C records the visible* Cont *routes from the socket row into the selected ledger row and the local naming surface;*
5. *P is the* Pkg *provenance over* $\mathsf{BranchObstruction}^{\uparrow}$, $\mathsf{AnalyticContinuationSocket}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{AnalyticContinuationOperation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{BranchObstruction}^{\uparrow}}$ *row.*

The carrier exports only the socket row, selected obstruction ledger, transport rows, continuation routes, package provenance, and local naming data. It contains no global analytic-continuation operator, branch selector, quotient of germs, monodromy invariant, or additional $\mathsf{Holomorphic}^{\uparrow}$ *output row.*

Lean4 ✓ `BEDC.Derived.BranchObstructionUp.BranchObstructionTasteGate single carrier alignment`

Theorem 694.2 (BranchObstruction NameCert obligations)**.** *For the carrier of Definition 694.1, the local* $\mathsf{NameCert}_{\mathsf{BranchObstruction}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier admission for* $R = (S, B, H, C, P, N)$*; socket compatibility, requiring* B *to be the branch-obstruction coordinate displayed by the accepted* $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ *row* S*; ledger exactness, requiring every branch-sensitive continuation request outside the local socket rows to be recorded in* B*; transport stability through* H *and the visible routes in* C*; and non-escape, excluding branch selectors, quotient germs, monodromy theorems, global continuation principles, zero-location data, and new holomorphic outputs.*

Proof. Carrier admission is the projection of the finite tuple in Definition 694.1. The rows S, B, H, C, P, N display the socket, selected obstruction ledger, componentwise transport, routes, provenance, and local naming row.

Socket compatibility follows by reading S before any obstruction row is inspected. The accepted socket already names the two source rows, overlap witness, operation row, output row, and branch-obstruction coordinate, so B is a selected row of the displayed socket rather than an ambient analytic-continuation object.

Ledger exactness is inherited from the socket ledger. Requests that leave the source, overlap, operation, and output rows are assigned to B as ledger data.

Transport stability is componentwise through H and C. These rows replay the same socket and selected obstruction ledger under accepted hsame movement and visible continuation routes.

The non-escape clause is an inventory argument. Since the carrier has no coordinate for branch selection, quotient-of-germs equality, monodromy, global continuation, zero-location data, or another $\mathsf{Holomorphic}^{\uparrow}$ output, no local $\mathsf{NameCert}_{\mathsf{BranchObstruction}^{\uparrow}}$ projection can export one. □

Theorem 694.3 (BranchObstruction ledger non-escape)**.** *Every branch-sensitive read of an accepted* $\mathsf{BranchObstruction}^{\uparrow}$ *carrier factors through the finite rows*

$$S, B, H, C, P, N.$$

If the read asks for continuation data not displayed by the accepted socket row S*, the request remains in the selected ledger row* B *and is not exported as a branch selector, quotient-of-germs equality, monodromy theorem, or new* $\mathsf{Holomorphic}^{\uparrow}$ *output.*

Lean4 ✓ `BEDC.Derived.BHistHexaSequenceNameCertUp.BHistHexaSequenceNameCertUp single carrier alignment`

Proof. Project the accepted carrier and apply Definition 694.2. The only branch-sensitive coordinate exposed by the packet is the selected ledger row B.

For a read internal to the socket, apply the socket rows displayed by S. The consumer obtains only the source, overlap, operation, output, transport, route, provenance, and local naming data already present in the accepted $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ packet.

For a read outside those rows, use the ledger exactness obligation. Such a request lands in B as obstruction accounting and does not become a continuation witness.

Finally route the result through H, C, P, N. These rows preserve and name the same finite ledger surface, so no exported branch selector, quotient germ, monodromy theorem, or extra holomorphic output can appear. □

Theorem 694.4 (BranchObstruction holomorphic handoff boundary)**.** *Every downstream* $\mathsf{Holomorphic}^{\uparrow}$ *consumer of an accepted* $\mathsf{BranchObstruction}^{\uparrow}$ *carrier receives holomorphic data only through the output row already displayed by the socket coordinate* S*. The selected obstruction ledger* B *may accompany that handoff as boundary accounting, but it cannot create or certify an additional* $\mathsf{Holomorphic}^{\uparrow}$ *output row.*

Proof. Read the socket coordinate S. Its holomorphic handoff has already passed through the accepted $\mathsf{AnalyticContinuationSocket}^{\uparrow}$ packet, where the output row is displayed as part of the local source-overlap-operation-output surface.

Read the selected obstruction row B separately. By Definition 694.3, B records requests outside that local surface as obstruction ledger data.

The transport and route rows H and C preserve this separation: socket output rows remain socket output rows, while obstruction requests remain ledger rows. The package and local naming rows P, N bind the same separation to the $\mathsf{BranchObstruction}^{\uparrow}$ certificate, so the handoff cannot manufacture another $\mathsf{Holomorphic}^{\uparrow}$ output. □

Theorem 694.5 (BranchObstruction holomorphic operation sibling route)**.** *Every accepted* $\mathsf{BranchObstruction}^{\uparrow}$ *packet reads its socket row as a consumer of the holomorphic source and output surface of chapter 68 and the operation row of chapter 577 before the selected obstruction ledger* B *is exported. The packet therefore exposes the ledger only after the socket has fixed the* $\mathsf{Holomorphic}^{\uparrow}$ *source rows, the continuation operation row, the output row, the displayed transports, routes, provenance, and local naming data.*

Proof. Read the carrier of Definition 694.1. Its socket coordinate S is an accepted analytic-continuation socket row displaying the two $\mathsf{Holomorphic}^{\uparrow}$ source rows, the operation row, the output row, the branch-obstruction ledger, transports, routes, provenance, and local naming row.

Apply Definition 694.2. The socket compatibility and ledger exactness clauses require B to be selected from that already displayed socket surface, so the sibling holomorphic and operation rows are read before the obstruction ledger is exported.

Now use Definition 694.3 and Definition 694.4. The ledger remains boundary accounting, while downstream holomorphic consumers receive only the source and output surface already displayed by S together with the operation row named in the same socket. □

Theorem 694.6 (BranchObstruction monodromy non-export)**.** *An accepted* $\mathsf{BranchObstruction}^{\uparrow}$ *carrier may record a request for monodromy, branch selection, or quotient-of-germs comparison in the selected ledger row* B*. It does not export such a request as a monodromy theorem, branch selector, quotient equality, global continuation principle, zero-location datum, or holomorphic output beyond the socket row* S*.*

Proof. Start from the carrier inventory of Definition 694.1. The displayed rows are exactly S, B, H, C, P, N, and none of them is a monodromy object, branch selector, quotient-germ representative, global continuation operator, zero-location row, or additional holomorphic output.

Apply Definition 694.3 to any nonlocal continuation request. The request is recorded in B and remains ledger data under the visible routes.

Apply Definition 694.4 for downstream holomorphic consumers. Such consumers see only the socket output already displayed by S, with B retained as boundary accounting rather than exported analytic evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{BranchObstruction}^{\uparrow}}$**.**

Constructive story: A BranchObstruction$^{\uparrow}$ packet is generated from an AnalyticContinuationSocket$^{\uparrow}$ row, a selected branch-obstruction ledger row, componentwise hsame transport, visible Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BranchObstruction}^{\uparrow}}$, seed)

Scope closed: The seed scope records the finite socket row, selected obstruction ledger, transport rows, visible routes, package provenance, local naming row, and non-export boundary for branch-sensitive continuation requests.

Formal verification: FormalStatus(NameCert$_{\text{BranchObstruction}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global analytic continuation, branch selection, quotient-of-germs equality, monodromy invariance, Riemann-hypothesis zero-location data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier compatibility, ledger exactness, holomorphic handoff boundary, and monodromy non-export obligations for the displayed socket and obstruction rows.

Chapter 695

A Concrete Naming Certificate for $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$

The $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ packet records the finite selector boundary used by the Real completeness diagonal route. It binds a precision request, a selected cofinal window, StreamName observations, a RegSeqRat witness, the RealUp seal handoff, and the BEDC transport rows as one inspectable naming certificate. The packet is a finite boundary between cofinal-window schedules and selector-budget packets; it is not a host choice principle, quotient stream, selected host limit, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

The selected cofinal window is consumed through chapter 126, then routed through the existing $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal surfaces before the local selector row is read. The selector does not own the window semantics; it schedules the finite $\mathsf{StreamName}^{\uparrow}$ window consumed by the diagonal handoff.

Definition 695.1 (Choice-free diagonal selector carrier)**.** *A* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\varepsilon, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *ε is the positive precision request made by the diagonal-limit consumer;*
2. *W is the selected cofinal window row, displayed as a finite family of admissible StreamName observation indices;*
3. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *observation rows opened by the selected cofinal window;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row assembled from those observations for the diagonal handoff;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the RegSeqRat witness has been assembled;*
6. *H records componentwise* hsame *transports for the request, window, StreamName observations, RegSeqRat witness, RealUp seal, and structural rows;*
7. *C records the displayed* Cont *routes for request inspection, cofinal-window selection, observation readback, RegSeqRat handoff, RealUp seal delivery, and non-escape inspection;*
8. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $\varepsilon, W, S, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorTasteGate single carrier alignment`

Theorem 695.2 (Choice-free diagonal selector carrier obligation)**.** *Every accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *packet exposes exactly the displayed rows*

$$\varepsilon, W, S, R, E, H, C, P, N$$

of Definition 695.1. The window route of Definition 695.5, the stage induction of Definition 695.8, and the non-escape theorem of Definition 695.9 may inspect those rows, but they cannot add a hidden selector row, quotient stream row, selected host limit, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ObligationCarrier.ChoiceFreeDiagonalSelectorCarrier obligation carrier`

Proof. Project the carrier in Definition 695.1. Its content rows are the precision request, selected cofinal window, StreamName observations, RegSeqRat witness, RealUp seal, componentwise transport, displayed replay, provenance, and local naming.

Apply the window route to order the analytic rows, then apply stage induction to retain the same row family across displayed finite stages. Finally use non-escape to rule out any additional host choice, quotient, selected-limit, or completeness coordinate. □

Theorem 695.3 (Choice-free diagonal selector classifier obligation)**.** *Componentwise* hsame *transport of accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *packets preserves the classifier route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

If the rows $\varepsilon, W, S, R, E, H, C, P, N$ are transported componentwise through the displayed H coordinate, then the window, stage, and non-escape results still force each selector read through StreamName observations, RegSeqRat handoff, and RealUp seal in that order.

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ClassifierObligation.ChoiceFreeDiagonalSelectorCarrie classifier obligation`

Proof. Read the transport row H in the carrier definition. It is componentwise over the request, window, StreamName observations, RegSeqRat witness, RealUp seal, and structural rows.

The window route fixes the order from selected window to StreamName observations, RegSeqRat witness, and RealUp seal. Stage induction preserves that order at each finite stage, and non-escape prevents the transported packet from acquiring a different classifier through host equality, quotient streams, or ambient RealUp completeness. □

Theorem 695.4 (Choice-free diagonal selector ledger obligation)**.** *For every accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *packet, the ledger rows P, N record provenance and naming for the finite selector route only. Together with Definition 695.5, Definition 695.8, and Definition 695.9, they certify the displayed route without supplying host choice, quotient streams, selected host limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorCarrier ledger obligation`

Proof. Project the provenance row P and local naming row N from the carrier. They name the same finite packet already displayed by the analytic and structural rows.

The window route and stage induction identify the finite selector path that those ledger rows can certify. The non-escape theorem then excludes host choice, quotient stream equality, selected host limits, hidden cofinal-window selection, unbounded StreamName scans, and ambient RealUp completeness from the ledger surface. □

Theorem 695.5 (Choice-free diagonal selector window route)**.** *Every accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier routes a diagonal precision request through the displayed finite cofinal window before any RegSeqRat or RealUp data can be consumed. For*

$$D = (\varepsilon, W, S, R, E, H, C, P, N),$$

each StreamName observation in S is opened only inside the selected cofinal window W, the RegSeqRat witness R is assembled only from those observations, and the RealUp seal row E is reached only through that witness route.

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorCarrier window route`

Proof. Project the carrier coordinates. The request row ε and selected cofinal window W are displayed before any StreamName, RegSeqRat, or RealUp row is available.

The observation rows S are listed as the finite family opened by W. Hence a downstream diagonal consumer can read StreamName data only from the selected cofinal window.

The witness row R appears after those observations, and the seal row E appears after R. The structural rows H, C, P, N transport, replay, package, and name the same finite route, so the selector boundary is exactly $\varepsilon, W, S, R, E, H, C, P, N$. □

Theorem 695.6 (Choice-free diagonal selector RegSeqRat seal factorization)**.** *For an accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier*

$$D = (\varepsilon, W, S, R, E, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *seal read from the selector factors first through the finite* $\mathsf{StreamName}^{\uparrow}$ *observations S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row R. The seal row E is therefore available only as the terminal handoff of the displayed route*

$$S \longmapsto R \longmapsto E,$$

with H, C, P, N preserving that same factorization by transport, replay, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.RegSeqRatSealFactorization.ChoiceFreeDiagonalSelector regseqrat seal factorization`

Proof. Project the carrier of Definition 695.1. The only observation rows before the Real-facing endpoint are the finite StreamName family S and the RegSeqRat witness row R.

Apply Definition 695.5. It forces each observation in S to be opened inside the selected cofinal window W, then assembles the RegSeqRat witness R from those observations before the RealUp seal row E can be reached.

The structural rows do not add another seal route. Componentwise transport through H, displayed replay through C, provenance through P, and local naming through N preserve the same $S \longmapsto R \longmapsto E$ route, so every selector RealUp read factors through the RegSeqRat handoff first. □

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorRegSeqRatSealFactorization`

Definition 695.7 (Choice-free diagonal selector stage carrier)**.** *For an accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier*

$$D = (\varepsilon, W, S, R, E, H, C, P, N),$$

a stage carrier is a finite row

$$D_j = (\varepsilon, W_j, S_j, R_j, E_j, H_j, C_j, P, N)$$

displayed inside the selected cofinal window W*. The row* W_j *is a finite prefix of* W*,* S_j *is the corresponding finite* $\mathsf{StreamName}^{\uparrow}$ *observation family,* R_j *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff assembled from* S_j*, and* E_j *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached from* R_j*. The structural rows* H_j, C_j, P, N *are the restrictions of the same transport, continuation, provenance, and local naming rows to that finite stage.*

Theorem 695.8 (Choice-free diagonal selector stage induction)**.** *Let* $D = (\varepsilon, W, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier whose selected cofinal window* W *is displayed as finite stage rows. Induction over those rows preserves the route*

$$\varepsilon, W, S, R, E, H, C, P, N$$

and proves that every successor stage remains inside the displayed path

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the selector boundary is sealed by E*. No successor stage may introduce a hidden cofinal-window selector, quotient stream equality, host choice rule, selected host limit, unbounded StreamName scan, or ambient completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.StageInduction.ChoiceFreeDiagonalSelectorCarrier stage induction`

Proof. For the base row, inspect the first displayed finite prefix W_0 of the selected cofinal window. Its observation family S_0 is read through $\mathsf{StreamName}^{\uparrow}$, the witness R_0 is assembled as the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the boundary E_0 is reached only after that handoff.

For the successor row, assume the stage carrier

$$D_j = (\varepsilon, W_j, S_j, R_j, E_j, H_j, C_j, P, N)$$

has preserved the displayed route. The next row extends the finite prefix of W by one displayed window entry, so the new observation is appended inside the same $\mathsf{StreamName}^{\uparrow}$ family, the RegSeqRat witness is rebuilt from the enlarged finite family, and the RealUp seal remains downstream of that witness.

The transport and replay restrictions H_j, C_j are inherited from the original carrier rows H, C, while P, N keep the same provenance and local naming. Thus the induction invariant is exactly the finite route $\varepsilon, W, S, R, E, H, C, P, N$, and every successor stage stays inside $\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$ before sealing.

Since each step only inspects a displayed finite row of W, the induction has no coordinate for hidden selection, quotient stream equality, host choice, selected host limits, unbounded observation scans, or ambient RealUp completeness. □

Theorem 695.9 (Choice-free diagonal selector non-escape)**.** *An accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple* $\varepsilon, W, S, R, E, H, C, P, N$*. In particular, it exports no host choice rule, quotient stream equality, selected host limit, hidden cofinal-window selector, unbounded StreamName scan, or ambient completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.Nonescape.ChoiceFreeDiagonalSelectorNonescape`

Proof. Read the displayed tuple in Definition 695.1. The content rows are the request, cofinal window, StreamName observations, RegSeqRat witness, and RealUp seal; the remaining rows only provide transport, route replay, provenance, and local naming for those entries.

Apply Definition 695.5. It confines every diagonal read to the selected finite cofinal window and forces the RegSeqRat and RealUp rows to be downstream of that route.

Since the carrier contains no coordinate for host choice, quotient stream equality, selected host limits, hidden selectors, unbounded observation scans, or RealUp completeness, no such object can be recovered from an accepted packet. □

Theorem 695.10 (Choice-free diagonal selector obligation closure)**.** *The* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *obligation surface is exactly the finite route over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *assembled by the carrier, classifier, ledger, cofinal-window route, RegSeqRat seal factorization, stage carrier, stage induction, and non-escape theorems. It exports no host choice rule, quotient stream equality, selected host limit, hidden cofinal-window selector, unbounded StreamName scan, or ambient completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.taste gate`

Proof. Start with Definition 695.1. The carrier displays the precision request, selected cofinal window, StreamName observations, RegSeqRat witness, RealUp seal, componentwise transport, replay, provenance, and local naming rows.

Apply the carrier, classifier, and ledger obligations of Definition 695.2, Definition 695.3, and Definition 695.4. These identify the finite rows that the local classifier and ledger may inspect.

Next use Definition 695.5 and Definition 695.6. They force every diagonal read through the displayed $\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$ route before the RealUp seal can be consumed.

Finally apply Definition 695.7, Definition 695.8, and Definition 695.9. Stage induction preserves the same finite rows, and non-escape rules out host choice, quotient streams, selected host limits, hidden selectors, unbounded scans, and ambient completeness. □

Theorem 695.11 (Choice-free diagonal selector scoped kernel route)**.** *Every accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *selector route factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *before any diagonal consumer can read the selector output. The route cannot add host choice, quotient streams, hidden cofinal selectors, unbounded* $\mathsf{StreamName}^{\uparrow}$ *scans, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorScopedKernelRoute`
↪ variant `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorCarrier scoped kernel route`

Proof. Start with the carrier of Definition 695.1. Its displayed rows are the precision request, selected finite cofinal window, StreamName observations, RegSeqRat witness, RealUp seal, componentwise transport, route replay, provenance, and local naming.

Apply the window route and RegSeqRat seal factorization, Definition 695.5 and Definition 695.6. They force every selector read through the finite $\mathsf{StreamName}^{\uparrow}$ window rows before the $\mathsf{RegSeqRat}^{\uparrow}$ witness is assembled and before the $\mathsf{Real}^{\uparrow}$ seal row is available.

Use stage induction to keep the same route across finite window prefixes. Then apply non-escape and the obligation closure theorem, Definition 695.9 and Definition 695.10, to rule out host choice, quotient streams, hidden cofinal selection, unbounded scans, and ambient Real completeness. The remaining route is exactly the scoped kernel route through BHist, hsame, Cont, Pkg, and NameCert. □

Theorem 695.12 (Choice-free diagonal selector scoped window consumer)**.** *Every diagonal consumer of an accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *packet must preserve the same* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *event history, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row already exposed by the scoped kernel route. A consumer that changes any of these rows is outside the scoped packet and cannot be read as the same choice-free diagonal selector.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.ChoiceFreeDiagonalSelectorScopedWindowConsumer`

Proof. Apply the scoped kernel route of Definition 695.11. It forces the selector output through the displayed StreamName, RegSeqRat, RealUp, transport, replay, provenance, and naming rows before any diagonal consumer can inspect it.

The non-escape theorem and obligation closure theorem then rule out host choice, quotient streams, hidden selectors, unbounded StreamName scans, selected host limits, and ambient Real completeness. Since the accepted carrier has no other coordinates, preserving the same consumer window is exactly preserving the same scoped packet. □

Theorem 695.13 (Choice-free diagonal selector public handoff)**.** *The public selector export of an accepted* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *packet preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal delivery, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, the local* NameCert *row, and the nonescape restrictions of the finite selector route. Any public consumer that changes one of these rows is not consuming the same choice-free diagonal selector packet.*

Lean4 ✓ `BEDC.Derived.ChoiceFreeDiagonalSelectorUp.PublicHandoff.ChoiceFreeDiagonalSelectorPublicHandoff`

Proof. Begin with the scoped kernel route and scoped window consumer. They force every selector read through the displayed finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ witness row, $\mathsf{Real}^{\uparrow}$ seal row, transport, replay, provenance, and naming rows.

Use the RegSeqRat seal factorization and stage induction to keep the handoff stable across finite prefixes. Each stage rebuilds the witness from the displayed window before the RealUp seal is delivered, so the public export cannot detach the seal from the selector route.

Finally apply the nonescape theorem. It rules out host choice, quotient stream equality, selected host limits, hidden cofinal-window selection, unbounded StreamName scans, and ambient RealUp completeness. The public handoff therefore exposes exactly the same finite packet and no additional selector surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ packet is generated from a precision request, selected cofinal window, finite StreamName observations, RegSeqRat witness, RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 695.1, Definition 695.5, Definition 695.6, Definition 695.8, Definition 695.9, Definition 695.10, Definition 695.11, and Definition 695.12 bind the finite choice-free diagonal selector scope.

Formal verification: FormalStatus(NameCert$_{\mathsf{ChoiceFreeDiagonalSelector}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host choice, selected host limits, hidden cofinal-window selection, or a public standard bridge.

Upgrade path: Public closure requires a public consumer handoff that preserves the displayed StreamName window, RegSeqRat handoff, RealUp seal delivery, transport stability, route stability, provenance, and non-escape for the same finite packet.

Chapter 696

A Concrete Naming Certificate for $\mathsf{AuditGateComposition}^{\uparrow}$

The $\mathsf{AuditGateComposition}^{\uparrow}$ packet names a finite composition surface for two audit gates that share a boundary row. It records the two gate rows, the boundary-content rows seen by each gate, the two refusal-result rows, and the composed refusal trace as one local $\mathsf{NameCert}$ packet. The packet is a composition certificate: it does not create a new audit policy, weaken either gate, or turn refusal into host-side permission. The shared-boundary comparison and refusal rows are consumed from chapter 526 before the composed trace is packaged.

Definition 696.1 (Audit-gate composition carrier)**.** *An* $\mathsf{AuditGateComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N)$$

with the following displayed rows:

1. G_0 *and* G_1 *are the two audit-gate rows being composed;*
2. B_0 *and* B_1 *are the boundary-content rows read by the two gates, with a displayed shared-boundary comparison between the exit surface of* G_0 *and the entry surface of* G_1*;*
3. R_0 *and* R_1 *are the refusal-result rows produced by the two gates;*
4. Q *is the composed refusal trace, recording that the downstream consumer sees* R_0 *followed by* R_1 *through the shared boundary;*
5. H *stores componentwise* hsame *transports for gate rows, boundary-content rows, refusal-result rows, composed traces, provenance, and the local name;*
6. K *stores the* Cont *replay routes for reading* G_0*, reading* G_1*, transporting the shared boundary, replaying both refusal rows, and projecting the composed refusal trace;*
7. P *stores* Pkg *provenance naming* $\mathsf{AuditGateBoundary}^{\uparrow}$*,* $\mathsf{OperationalMembrane}^{\uparrow}$*,* $\mathsf{AuditMembrane}^{\uparrow}$*, forbidden-ancestry audit rows, paper-Lean drift rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *source packets;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{AuditGateComposition}^{\uparrow}}$ *row.*

The carrier exports only the displayed gate rows, boundary rows, refusal rows, composed trace, transport, replay, provenance, and local naming rows.

Theorem 696.2 (Audit-gate composition NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AuditGateComposition}^{\uparrow}}$ *surface for the carrier of Definition 696.1 consists of six finite obligations:*

1. *carrier admission for* $C = (G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N)$*;*
2. *first-gate exposure for* G_0*,* B_0*, and* R_0*;*
3. *second-gate exposure for* G_1*,* B_1*, and* R_1*;*
4. *shared-boundary transport, requiring the exit content of the first gate and the entry content of the second gate to be compared only through the displayed* hsame *rows in* H*;*
5. *composed refusal replay, requiring* Q *to be read through the displayed* Cont *routes in* K*;*
6. *consumer non-escape, requiring the composed packet to export no hidden permission row, no skipped gate, no host repair of a refused boundary, and no audit result outside the displayed finite trace.*

Lean4 ✓ `BEDC.Derived.BHistHendecaTupleNameCertUp.BHistHendecaTupleNameCertCarrierAdmission`

Proof. Carrier admission is projection from Definition 696.1. The two gate rows, two boundary rows, two refusal rows, composed trace, transport, replay, provenance, and local naming row are all displayed coordinates of the same finite packet.

The first and second gate obligations are read separately from the tuple. Each gate keeps its own boundary-content row and refusal-result row, so composition cannot identify the two gates or erase one of the two refusal surfaces.

Shared-boundary transport is confined to H. The only comparison between B_0 and B_1 is the displayed componentwise hsame comparison, and the only public replay of the composed trace passes through K.

Finally P and N package the displayed rows as local NameCert data. Since the carrier has no coordinate for a hidden permission row, skipped gate, host repair, or unlisted audit result, the six obligations exhaust the public surface of the composition packet. □

Theorem 696.3 (Audit-gate composition boundary obligation)**.** *For an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *carrier*

$$C = (G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N),$$

the shared boundary rows B_0 *and* B_1 *are the only boundary comparison surface. Every comparison between the exit content of* G_0 *and the entry content of* G_1 *must be read through the componentwise* hsame *transports in* H *and the displayed* Cont *replay routes in* K*.*

Lean4 ✓ `BEDC.Derived.AuditGateCompositionUp.AuditGateCompositionBoundary obligation`

Proof. Project the carrier of Definition 696.1. The only boundary-content coordinates are B_0 and B_1, and the only displayed comparison rows attached to them are the transports in H and replay routes in K.

Now apply Definition 696.2. Its shared-boundary transport obligation reads the exit content of the first gate and the entry content of the second gate only through the displayed hsame rows.

Finally use Definition 696.6. The sibling route consumes the audit-gate boundary and membrane rows before Q is public, so no ambient boundary object or host endpoint comparison can replace B_0, B_1, H, K. □

Theorem 696.4 (Audit-gate composition membrane obligation)**.** *For an accepted* AuditGateComposition$^{\uparrow}$ *carrier, the audit-membrane and operational-membrane provenance rows enter only through the provenance coordinate P. They cannot create a bypass row, host permission row, detached membrane policy, or second route around the displayed refusal and boundary packet.*

Proof. Project the provenance coordinate in Definition 696.1. The row P names the AuditMembrane$^{\uparrow}$ and OperationalMembrane$^{\uparrow}$ sources together with the local BHist, hsame, Cont, Pkg, and NameCert packets; no separate membrane carrier is exported.

Apply Definition 696.2. Its consumer non-escape obligation rules out hidden permission rows, skipped gates, host repair of refused boundaries, and unlisted audit results.

Then use Definition 696.6. Since sibling membrane rows are consumed only as named provenance before the composed refusal trace is packaged, they cannot generate a bypass or host-side permission surface. □

Lean4 ✓ `BEDC.Derived.AuditGateCompositionUp.AuditGateCompositionCarrier membrane obligation`

Theorem 696.5 (Audit-gate composition operational obligation)**.** *For an accepted* AuditGateComposition$^{\uparrow}$ *carrier, the composed refusal trace Q is replayed only after the gate rows G_0, G_1, boundary rows B_0, B_1, and refusal rows R_0, R_1 have been consumed. No consumer can read Q as an operational result before those six source rows have entered the displayed route.*

Proof. Project the carrier of Definition 696.1. The trace coordinate Q is displayed only after the two gate rows, two boundary rows, and two refusal-result rows in the tuple.

Use Definition 696.2. Its first-gate, second-gate, shared-boundary transport, and composed-refusal replay obligations order the read through $G_0, G_1, B_0, B_1, R_0, R_1$ before Q is projected.

Finally apply Definition 696.6. The dependency route consumes the sibling gate and membrane rows before the local packet exports the composed trace, so Q has no premature operational read outside the displayed finite carrier. □

Theorem 696.6 (Audit-gate composition sibling dependency route)**.** *Every composed refusal route of an accepted* AuditGateComposition$^{\uparrow}$ *packet first consumes the gate rows of chapter 526 and the membrane boundary surface of chapter 533 before the local packet*

$$G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N$$

can export the composed refusal trace. The exported trace is therefore a routed composition of sibling gate and membrane rows, not a standalone refusal oracle or a host-side repair of a refused boundary.

Proof. Project the accepted carrier of Definition 696.1. The rows G_0, G_1 are the two gate rows, B_0, B_1 are the boundary-content rows, and R_0, R_1, Q, H, K, P, N are the displayed refusal, transport, replay, provenance, and local naming rows.

Use Definition 696.2. Its first-gate and second-gate obligations expose the two gate rows before the composed trace is public, while its shared-boundary transport obligation forces the boundary comparison through the displayed rows in H and K.

The provenance row P names the sibling audit-gate boundary and operational membrane surfaces, and the local NameCert row N packages them with the route. Thus the composed trace Q can be exported only after those sibling gate and membrane rows have been consumed. □

Theorem 696.7 (Audit-gate composition refusal transitivity)**.** *Let* $C = (G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N)$ *be an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *carrier. If the first gate refuses the boundary content recorded in* R_0 *and the second gate refuses the transported boundary content recorded in* R_1*, then every public read of the composed surface factors through the ordered refusal trace* Q*. The read cannot replace either refusal by acceptance and cannot bypass the shared-boundary transport.*

Proof. Project the accepted packet to the two gate surfaces. The first refusal is the row R_0 attached to G_0 and B_0, and the second refusal is the row R_1 attached to G_1 and B_1.

Use the shared-boundary rows in H to transport the exit content of the first gate to the entry content of the second gate. This is the only bridge between the two refusal reads, so the second refusal remains tied to the transported boundary rather than to a new host-side boundary object.

Replay the ordered route through K. The route reads R_0, crosses the displayed boundary comparison, reads R_1, and then exposes Q as the composed refusal trace.

Because Q is built only from those displayed rows, a consumer has no coordinate for replacing either refusal by acceptance or for bypassing the shared-boundary transport. □

Theorem 696.8 (Audit-gate composition consumer non-escape)**.** *Every consumer read of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet factors through*

$$G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N.$$

The consumer receives a finite NameCert packet for composed gate surfaces, shared boundary rows, refusal transitivity, and consumer non-escape. It receives no general audit bypass, no external axiom-ancestry oracle, no paper-Lean marker repair, no operational-membrane override, and no host permission to treat a refused boundary as accepted.

Lean4 ✓ `BEDC.Derived.AuditGateCompositionUp.AuditGateCompositionUp StdBridge`

Proof. Start from the carrier inventory. A consumer can project the two gate rows, two boundary-content rows, two refusal-result rows, composed trace, componentwise transport, replay route, provenance, and local name.

The composed read is then determined by Definition 696.2 and Definition 696.7. The first theorem identifies the finite obligation surface, and the second theorem identifies the ordered refusal route through the shared boundary.

The provenance row records the neighboring audit packets that the composition may cite, but it does not import their host-side policy space. Since the carrier has no bypass, marker-repair, membrane-override, or acceptance-after- refusal coordinate, none of those objects can escape to the consumer. □

Theorem 696.9 (Audit-gate composition root-source boundary)**.** *Every root read of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet exposes the two audit-gate rows* G_0, G_1*, the two shared boundary rows* B_0, B_1*, and the transport rows in* H *before it may read the composed trace* Q*. The root surface therefore has no direct route from either gate row to* Q *that bypasses the displayed shared-boundary packet.*

Proof. Project the carrier of Definition 696.1. The only rows preceding the composed trace are $G_0, G_1, B_0, B_1, R_0, R_1$ and the componentwise transport data in H; no second source row is present.

The NameCert surface of Definition 696.2 binds those rows as the displayed root source for the composition. Refusal transitivity then reads the trace through the shared boundary, so any consumer that reaches Q has already exposed G_0, G_1, B_0, B_1 through the same finite packet. □

Theorem 696.10 (Audit-gate composition composed-refusal inheritance)**.** *Every composed trace of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet inherits both refusal rows* R_0 *and* R_1 *through the shared boundary. The trace* Q *is not a new refusal object; it is the replay of* R_0*, the transported boundary read, and* R_1 *inside the displayed carrier.*

Proof. Start with the refusal rows in the carrier and the boundary rows named by Definition 696.9. These are the only displayed sources from which Q can be read.

Use Definition 696.7. It orders the composition by the first refusal row, the shared-boundary transport, and the second refusal row. Since consumer non-escape adds no host refusal channel, the composed trace can inherit only R_0 and R_1 through that boundary. □

Theorem 696.11 (Audit-gate composition downstream root package)**.** *The downstream root package for* $\mathsf{AuditGateComposition}^{\uparrow}$ *consists of carrier exposure, shared-boundary transport, composed refusal replay, and consumer non-escape. Concretely, every accepted packet exposes*

$$G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N$$

as a finite NameCert surface for downstream audit-gate consumers, and exports no bypass row, skipped-gate row, external ancestry oracle, or host acceptance coordinate.

Proof. Carrier exposure is Definition 696.1, and the NameCert obligations are Definition 696.2. Together they display the two gate rows, two boundary rows, two refusal rows, trace row, transports, replay route, provenance, and local name.

Shared-boundary transport is supplied by Definition 696.9. Composed refusal replay is supplied by Definition 696.10 and Definition 696.7.

Finally apply Definition 696.8. It removes every consumer route to bypass rows, skipped-gate rows, external ancestry oracles, and host acceptance coordinates, leaving exactly the displayed downstream root package. □

Theorem 696.12 (Audit-gate composition shared-boundary source)**.** *In every accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *carrier*

$$C = (G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N),$$

the shared boundary between G_0 *and* G_1 *is read only through the displayed* BHist *rows* B_0, B_1*, the componentwise* hsame *transports in* H*, and the* Cont *replay route in* K*. No gate may substitute an ambient boundary row, host boundary object, or unstated endpoint comparison for this displayed source.*

Proof. Project the carrier of Definition 696.1. The only boundary-content rows in the packet are the displayed BHist rows B_0 and B_1, and the only named rows surrounding them are the gate rows, refusal rows, trace row, transports, continuations, provenance, and local certificate.

Apply the NameCert obligations of Definition 696.2. They bind the shared-boundary comparison to the componentwise hsame data in H and the displayed Cont route in K, while P and N record the same finite Pkg and $\mathsf{NameCert}$ provenance.

Finally use refusal transitivity and consumer non-escape, Definition 696.7 and Definition 696.8. Since every composed read passes through the ordered displayed trace and no ambient boundary coordinate is exported, neither gate can replace B_0, B_1, H, K by an external source. □

Theorem 696.13 (Audit-gate composition refusal-replay totality)**.** *Every public composed-refusal read of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet factors through* R_0*, the transported boundary row,* R_1*, and* Q*. The rows* P *and* N *only package and name that same finite* BHist *route by* Pkg *provenance and a local* $\mathsf{NameCert}$ *certificate; they do not add a second refusal path.*

Proof. Begin with the refusal-result rows R_0 and R_1 in the carrier. By Definition 696.2, the first gate exposes R_0, the second gate exposes R_1, and the shared boundary is transported componentwise through hsame rows in H.

The replay theorem Definition 696.7 then orders the public read. The route in K reads R_0, crosses the transported boundary row, reads R_1, and exposes Q as the composed refusal trace.

It remains to account for the packaging rows. The provenance row P is a Pkg record for the displayed audit, BHist, hsame, Cont, and NameCert sources, and N is the local name for the same packet. By consumer non-escape, these rows cannot be reread as a separate refusal channel or as host permission. □

Theorem 696.14 (Audit-gate composition consumer-permission exclusion)**.** *After the ordered refusal trace of an accepted* AuditGateComposition$^{\uparrow}$ *carrier is consumed, the composed packet exports no permission row, skipped-gate witness, policy weakening, or host acceptance coordinate. The consumer remains inside the displayed* BHist *carrier, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Use the carrier inventory and the NameCert obligations first. They expose only the two gate rows, boundary rows, refusal rows, composed trace, componentwise transports, continuation routes, provenance, and local name.

Next apply refusal-replay totality together with Definition 696.7. A consumer reaches the public result by reading R_0, the transported boundary row, R_1, and Q through the displayed Cont route in K.

Finally cite consumer non-escape, Definition 696.8. Because no coordinate of C is a permission row, skipped-gate witness, policy weakening, or host acceptance coordinate, the Pkg row P and NameCert row N can only package the refusal trace already displayed by the packet. □

Theorem 696.15 (Audit-gate composition root-boundary totality)**.** *Every root read of an accepted* AuditGateComposition$^{\uparrow}$ *packet exposes the two audit-gate rows, both boundary-content rows, both refusal rows, the shared-boundary transport, the replay row, the package row, and the local NameCert row. The root packet has no public coordinate outside* $G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N$ *for reinterpreting the composed refusal trace as a host permission or as an external boundary oracle.*

Proof. Project the accepted carrier of Definition 696.1. Its displayed coordinates are exactly the two gate rows, two boundary rows, two refusal rows, composed trace, transport row, replay row, package provenance, and local NameCert row.

Apply the NameCert obligations of Definition 696.2. They expose the first gate, second gate, shared boundary, composed refusal replay, and consumer non-escape obligations over the same tuple.

The refusal-replay totality theorem and the consumer-permission exclusion theorem then exhaust the root read. Since the carrier contains no further coordinate, the root boundary cannot export a permission row, host repair row, or external boundary oracle. □

Theorem 696.16 (Audit-gate composition operational membrane route)**.** *Every operational-membrane consumption route for an accepted* AuditGateComposition$^{\uparrow}$ *packet is consumed only through the* AuditGateBoundary$^{\uparrow}$, OperationalMembrane$^{\uparrow}$, *and* AuditMembrane$^{\uparrow}$ *surfaces named by the carrier provenance. The composed route cannot bypass these named surfaces by reading an ambient membrane, a detached host policy, or an unlisted audit row.*

Proof. Read the provenance row P in the carrier. It names $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{OperationalMembrane}^{\uparrow}$, $\mathsf{AuditMembrane}^{\uparrow}$, the forbidden-ancestry audit rows, paper-Lean drift rows, and the local BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ sources.

Use the sibling dependency route theorem Definition 696.6. It forces the composed refusal trace to consume the audit-gate boundary and operational membrane rows before the local packet exports Q.

Finally apply consumer non-escape and consumer-permission exclusion. These two boundaries leave no coordinate for an ambient membrane read, detached host policy, or unlisted audit row, so the operational route remains inside the three named membrane surfaces. □

Theorem 696.17 (Audit-gate composition downstream ledger exhaustion)**.** *Every downstream ledger read of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet exhausts the composed refusal trace* $R_0 \longmapsto R_1 \longmapsto Q$*. The exported ledger contains no bypass, host repair, external ancestry oracle, permission row, skipped-gate witness, or policy-weakening coordinate.*

Proof. Start from refusal transitivity. It orders the two refusal rows through the shared-boundary transport and states that every public read of the composed surface factors through Q.

Refusal-replay totality then identifies the downstream ledger as the route through R_0, the transported boundary row, R_1, and Q. The package and local name rows only record that displayed route.

Consumer-permission exclusion supplies the final exhaustion step. A bypass, host repair, external ancestry oracle, permission row, skipped-gate witness, or policy-weakening coordinate would require an additional public row, but the accepted carrier exports only the displayed finite ledger. □

Theorem 696.18 (Audit-gate composition shared-boundary transitivity)**.** *For an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet, comparison from the first boundary-content row* B_0 *to the second boundary-content row* B_1 *composes only through the displayed* hsame *row* H*, the displayed replay row* K*, and the sibling* $\mathsf{AuditGateBoundary}^{\uparrow}$ *and* $\mathsf{OperationalMembrane}^{\uparrow}$ *route named by the provenance row* P*. No public comparison route may replace this boundary movement by a host repair, detached membrane policy, or skipped audit-gate row.*

Proof. Start with the carrier of Definition 696.1. Its only boundary-content coordinates are B_0 and B_1, and the only public rows available for moving between them are the componentwise transport row H, replay row K, package row P, and local naming row N.

Apply the local obligations of Definition 696.2. The shared-boundary obligation ties the two boundary rows to the same accepted packet, while the replay and provenance obligations keep that comparison inside the displayed finite rows.

Use the sibling dependency route and the operational membrane route. They force the comparison to pass through the named $\mathsf{AuditGateBoundary}^{\uparrow}$ and $\mathsf{OperationalMembrane}^{\uparrow}$ surfaces before the composition exports the trace Q.

Finally apply consumer non-escape and downstream ledger exhaustion. Since the public ledger has no host repair, detached policy, or skipped-gate coordinate, the boundary comparison can only be the displayed H, K, P, N route between B_0 and B_1. □

Theorem 696.19 (Audit-gate composition refusal-order determinacy)**.** *For accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packets preserving* $G_0, G_1, B_0, B_1, R_0, R_1, H, K, P, N$*, the ordered refusal trace is the same row* Q*. In particular, the exported trace cannot be reordered through host repair, policy weakening, a skipped gate, or an alternate public ledger row.*

Proof. Project the shared rows from the two accepted carriers. The two gate rows, boundary rows, refusal rows, transport, replay, provenance, and local naming rows agree by hypothesis.

Apply refusal transitivity. The composed refusal trace is read by first applying G_0 to B_0, then transporting the boundary comparison, then applying G_1 to B_1. This fixes the order $R_0 \longmapsto R_1 \longmapsto Q$.

The shared-boundary transitivity row supplies the only route between B_0 and B_1. Thus the second packet has no different comparison path from which a second trace could be assembled.

Consumer-permission exclusion and downstream ledger exhaustion remove the last alternatives. A reordered trace would require a host repair, policy weakening, skipped-gate, or alternate ledger coordinate, and none is exported by the accepted packet. □

Theorem 696.20 (Audit-gate composition support-row exhaustion)**.** *Every public support read of an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet is exhausted by the transport and replay rows* H, K*, the provenance and local naming rows* P, N*, the two gate rows* G_0, G_1*, the two boundary rows* B_0, B_1*, and the two refusal rows* R_0, R_1 *together with their composed trace* Q*. No support read exposes a host audit policy, parameter transport, external boundary oracle, skipped-gate witness, or ledger row outside this finite package.*

Proof. Begin with the carrier inventory. The public support surface consists exactly of $G_0, G_1, B_0, B_1, R_0, R_1, Q, H,$

The NameCert obligations expose these rows in their required roles: the two gate obligations, the shared boundary obligation, refusal replay, transport, continuation, provenance, and local naming.

Next combine refusal transitivity, shared-boundary transitivity, and refusal-order determinacy. These three rows account for the only boundary movement and the only ordered refusal trace available to a public consumer.

Finally use consumer non-escape, consumer-permission exclusion, and downstream ledger exhaustion. They rule out host audit policy, parameter transport, external boundary oracles, skipped gates, and off-packet ledger rows, leaving only the displayed support package. □

Theorem 696.21 (Audit-gate composition public export)**.** *The carrier, NameCert obligations, refusal transitivity, consumer nonescape, consumer-permission exclusion, root-boundary totality, operational membrane route, downstream ledger exhaustion, shared-boundary transitivity, refusal-order determinacy, support-row exhaustion, and checked TasteGate row form the public export surface for* $\mathsf{AuditGateComposition}^{\uparrow}$*. A public consumer of the accepted packet reads exactly the two audit-gate rows, two boundary rows, two refusal rows, composed trace, transport and replay rows, provenance, local name, and TasteGate witness; it receives no host audit policy, skipped gate, external boundary oracle, permission override, or alternate downstream ledger.*

Proof. First assemble the finite carrier and naming surface from Definition 696.1 and Definition 696.2. These rows expose the packet coordinates and the local certificate consumed by any public read.

Next combine the route theorems: refusal transitivity, root-boundary totality, the operational membrane route, downstream ledger exhaustion, shared-boundary transitivity, refusal-order determinacy, and support-row exhaustion. Together they give the only ordered path through $G_0, G_1, B_0, B_1, R_0, R_1, Q, H, K, P, N$.

Finally apply consumer nonescape, consumer-permission exclusion, and the checked TasteGate row. These remove host policy, permission override, skipped gate, and alternate-ledger exits, so the displayed packet is already the public export surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditGateComposition}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditGateComposition}^{\uparrow}$ packet is generated from two audit-gate rows, two boundary-content rows, two refusal-result rows, a composed refusal trace, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditGateComposition}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 696.1, Definition 696.2, Definition 696.7, Definition 696.8, Definition 696.14, Definition 696.15, Definition 696.16, Definition 696.17, Definition 696.18, Definition 696.19, Definition 696.20, Definition 696.21, and

Lean4 ✓ `BEDC.Derived.AuditGateCompositionUp.taste gate`

bind the root boundary, membrane route, downstream refusal ledger, shared-boundary support rows, refusal-order determinacy, checked TasteGate instance, and public export surface of the finite audit-gate composition packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditGateComposition}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.AuditGateCompositionUp.AuditGateCompositionTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close audit-policy completeness, external ancestry search, marker repair, operational-membrane override, or a public standard bridge.

Upgrade path: No further formal bridge-axis upgrade is needed because the chapter has reached the top formal bridge status.

Chapter 697

A Concrete Naming Certificate for $\mathsf{DiagonalTailSelector}^{\uparrow}$

The $\mathsf{DiagonalTailSelector}^{\uparrow}$ packet binds the Real-completion diagonal tail-selector route as one finite naming surface. It starts from a regular rational input row and a displayed modulus, selects one cofinal tail index, reads only finite StreamName and dyadic windows, hands the selected tail through RegSeqRat regularity and the RealUp seal, and records the BEDC transport, continuation, provenance, and local naming rows in the same packet. The selector is therefore a concrete NameCert carrier for the finite cofinal-tail route, not a quotient stream, host-selected limit, ambient completeness principle, countable-choice rule, or hidden equality of completed reals.

This carrier consumes the finite selector route of chapter 673 and the selected-tail witness surface of chapter 458. Its diagonal tail row is not a new selector oracle; it repacks those sibling finite-window obligations through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The finite cofinal row also consumes the precision-to-window-to-dyadic-to-RegSeqRat-to-Real seal route of chapter 681. Thus the selected tail index is scheduled through the same finite $\mathsf{TailCofinalitySchedule}^{\uparrow}$ route before any diagonal consumer reads W, D, T, S as a completion-facing packet.

Definition 697.1 (Diagonal tail selector carrier)**.** *A* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* RegSeqRat *input row for the regular Cauchy family whose diagonal tail is being selected;*
2. *n is the unary precision request row;*
3. *μ is the displayed modulus row that assigns a finite cofinal bound to the precision request;*
4. *k is the selected cofinal tail index, displayed as the value read from μ at n;*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened at the selected tail and precision request;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row extracted from the finite window;*
7. *T is the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *tail-regularity row binding the selected cofinal tail to the input row R;*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite window, dyadic readback, and tail-regularity rows are displayed;*

9. *H records componentwise* hsame *transports for R, n, μ, k, W, D, T, S and the local structural rows;*
10. *C records the displayed* Cont *route from the precision request through modulus selection, finite windows, dyadic readback, tail regularity, and the RealUp seal;*
11. *P is the* Pkg *provenance over* $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the visible rows $R, n, \mu, k, W, D, T, S, H, C, P, N$.

Theorem 697.2 (Diagonal tail selector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}^{\uparrow}}$ *surface for Definition 697.1 consists of the following finite obligations:*

1. *carrier admission, requiring every accepted selector object to expose the packet $G = (R, n, \mu, k, W, D, T, S, H, C, P$*
2. *selector exactness, requiring the cofinal index row k to be read from the displayed modulus μ at the unary precision request n;*
3. *finite-window readback, requiring every selected diagonal observation to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row D;*
4. *tail regularity handoff, requiring the selected cofinal tail to enter downstream reads only through the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *row T tied to the input row R;*
5. *RealUp seal handoff, requiring the seal row S to be reached only after W, D, T and the displayed transport, continuation, provenance, and local naming rows are available;*
6. *non-escape, requiring the packet to export no quotient stream, countable-choice selector, host-selected limit, ambient completeness theorem, host real equality, or hidden tail oracle.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier namecert obligations`

Proof. Carrier admission is projection from the displayed tuple in Definition 697.1. The input row, precision request, modulus, selected cofinal index, finite window, dyadic readback, tail-regularity row, RealUp seal, transports, routes, provenance, and local naming row are all coordinates of one finite packet.

Selector exactness reads n and μ before k is available. Thus the selected tail index is a named row of the packet and not an external subsequence choice.

The finite-window and tail-regularity clauses then route the selected index through W, D, T. The StreamName window and dyadic readback expose only finite observations, and the regularity handoff is the displayed $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ row tied to R.

Finally inspect S, H, C, P, N. These rows carry the selected finite route into the RealUp seal and provide no coordinate for quotient streams, countable choice, selected host limits, ambient completeness, host real equality, or hidden tail selection. □

Theorem 697.3 (Diagonal tail selector selector-row determinacy)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Any transported read of the same packet with the same precision row n, modulus row μ, componentwise* hsame *data for the displayed rows, displayed* Cont

route, Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}^{\uparrow}}$ *row reaches the same selected cofinal index row k. Thus selector comparison is determined by the visible selector rows of G and has no countable-choice coordinate or hidden selector coordinate.*

Proof. Project the carrier from Definition 697.1. The selected cofinal index appears as the displayed packet coordinate k, and the only rows upstream of it are the precision request n and modulus row μ.

Use selector exactness in Definition 697.2. It requires k to be read from μ at n, so any transported read preserving n, μ, the componentwise hsame rows, the displayed Cont route, the Pkg provenance, and the local NameCert row preserves the selected row itself.

The remaining coordinates W, D, T, S, H, C, P, N transport, replay, package, and name the already selected finite route. They do not contain an independent selector, a countable-choice schedule, or a hidden tail-selection coordinate. □

Theorem 697.4 (Diagonal tail selector window-threshold order)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

every downstream read of the tail-regularity row T or the RealUp seal row S first passes through the ordered finite route

$$n, \mu, k, W, D.$$

Consequently the finite $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback are exposed before any tail regularity or seal row is available.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier window threshold order`

Proof. Begin with the selector rows. The NameCert obligations force k to be read from μ at n, and Definition 697.3 makes that selected row invariant under the displayed transports and local naming row.

Next read the finite-window clause of Definition 697.2. Once k is fixed, selected diagonal observations must pass through the $\mathsf{StreamName}^{\uparrow}$ window W and the $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D.

Only then can the tail-regularity and seal clauses be consumed. The row T is downstream of the finite window and dyadic readback, and S is downstream of W, D, T together with the structural rows H, C, P, N. □

Theorem 697.5 (Diagonal tail selector seal after tail regularity)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the RealUp seal row S is available only after the selected cofinal tail row T has been routed through the displayed $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *handoff tied to the input row R. The seal handoff exports no quotient stream, host-selected limit, countable-choice selector, host real equality, hidden tail oracle, or ambient completeness coordinate.*

Proof. First apply Definition 697.3. The selected cofinal row is fixed by n, μ, k and the displayed transport, continuation, provenance, and local naming rows, rather than by an external selector.

Then apply Definition 697.4. Every downstream read of T or S passes through n, μ, k, W, D, so the finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ readback are already exposed before the tail-regularity row is consumed.

Finally use the RealUp seal handoff clause in Definition 697.2. It places S after W, D, T and the structural rows H, C, P, N, while the tail-regularity clause ties T to R through the displayed $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ handoff. The non-escape clause removes any remaining route through quotient streams, host-selected limits, countable-choice selectors, host real equality, hidden tail oracles, or ambient completeness. □

Theorem 697.6 (Diagonal tail selector limit handoff)**.** *Every Real-completion diagonal-limit consumer of an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet factors through the finite route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

Equivalently, the route composes the $\mathsf{LimitSelector}^{\uparrow}$ *precision-and-modulus selection surface, the* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *tail-regularity handoff, the* $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ *cofinal dyadic-window route, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-source row, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *finite seal consumer, and the* $\mathsf{Real}^{\uparrow}$ *seal into one displayed BEDC packet. The consumer receives no quotient stream, selected host limit, countable-choice row, ambient completeness theorem, host real equality, or unbounded tail selector.*

Proof. Start with the precision-and-modulus part of the route. The carrier displays R, n, μ, k, and Definition 697.2 forces k to be read from μ at n, matching the finite selector surface of $\mathsf{LimitSelector}^{\uparrow}$.

Next read the finite observation part. The selected cofinal index opens the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ readback D, and the $\mathsf{RegSeqRat}^{\uparrow}$ source row represented by R. The dyadic cofinal-window role is carried by the same selected index and finite window rows used by $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$.

Now pass to regularity and sealing. The row T is the displayed $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ handoff for the selected tail, and S is reached only after W, D, T have been named. A $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer therefore reads the same finite window, dyadic readback, regular handoff, and RealUp seal rows before it can expose any completion-facing seal data.

Finally use the structural rows H, C, P, N. They transport, replay, package, and name exactly the same finite route, so the consumer has no remaining channel for quotient streams, countable choice, selected host limits, ambient completeness, host real equality, or an unbounded tail selector. □

Theorem 697.7 (Diagonal tail selector window admission)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

every selected observation admitted to the selector route passes through the precision request n, displayed modulus μ, selected cofinal index k, finite $\mathsf{StreamName}^{\uparrow}$ *window W, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row D before the tail-regularity row T is read. No consumer may read T as if it were supplied by an unbounded tail search, quotient stream, countable-choice schedule, or host selected limit.*

Proof. Project the carrier of Definition 697.1. The only coordinates that select an observation before tail regularity are n, μ, k, W, D.

Apply the NameCert obligations of Definition 697.2. Selector exactness forces k to be read from μ at n, and finite-window readback forces the selected observation through W and D.

Finally use Definition 697.17. It rules out alternate cofinal-tail channels, so the tail-regularity row T can be consumed only after the displayed finite-window route has been admitted. □

Theorem 697.8 (Diagonal tail selector Real seal factorization)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *seal read from* S *factors through the displayed route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

In particular, the seal consumer first reads the $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, selector rows* n, μ, k*, finite window* W*, dyadic readback* D*, tail-regularity handoff* T*, componentwise* hsame *transports, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row* N*. It receives no quotient stream, host selected limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate.*

Proof. Begin with Definition 697.6. It identifies the Real-completion consumer route with the finite selector packet $R, n, \mu, k, W, D, T, S, H, C, P, N$.

Apply Definition 697.7 to the observation part of the route. This ensures that the selected finite window and dyadic readback are present before the tail-regularity row is read.

Use Definition 697.18 when the selected tail is supplied through a shared-threshold certificate, and then apply Definition 697.19. The seal row S is reachable only through the displayed selector, window, tail-regularity, transport, continuation, provenance, and naming rows. □

Theorem 697.9 (Diagonal tail selector CauchyLimitSeal factorization)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

any $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer reached by the selector route reads the seal only through the displayed rows*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

The route first exposes the $\mathsf{RegSeqRat}^{\uparrow}$ *source* R*, the request and selector rows* n, μ, k*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback* D*, the tail row* T*, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It has no coordinate for a quotient stream, a host selected limit, countable choice, host real equality, hidden tail oracle, or ambient completeness.*

Proof. Project the carrier of Definition 697.1. The accepted packet exposes exactly $R, n, \mu, k, W, D, T, S, H, C, P, N$ before a seal consumer can read any completion-facing row.

Apply the NameCert obligations and the limit handoff. They force k to be read from μ at n, route the selected observation through W and D, and place the $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer behind the same displayed selector packet.

Finally use Definition 697.8. The real-facing seal factorization already confines the downstream seal row to the selector, window, tail, transport, continuation, provenance, and local naming rows. The carrier inventory therefore supplies no row for any excluded host or quotient object. □

Theorem 697.10 (Diagonal tail selector RegularCauchyTailMeet consumption)**.** *Suppose an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

uses its tail row T *through a shared-tail consumer. Then the shared-tail use must factor through the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold rows before the selector seal row* S *is read. The consumed surface is the transported source rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger,*

$\mathsf{RegSeqRat}^{\uparrow}$ *comparison row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows already recorded by* H, C, P, N.

Proof. Start from the selector carrier and its local NameCert obligations. They make T the only row through which the selected regular-tail information can be read before the seal row S becomes visible.

Use Definition 697.7 and Definition 697.17. These results confine the selected tail read to the finite window and dyadic readback rows W, D together with R, n, μ, k, H, C, P, N.

Now apply Definition 697.18. When a shared-tail packet supplies T, it is consumed as the public $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold certificate before any $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal row is read. The compatibility theorem therefore fixes the threshold rows as the only admitted shared-tail route. □

Theorem 697.11 (Diagonal tail selector tail-meet threshold admission)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet whose tail row* T *is supplied through* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*. Any accepted packet using that selected tail row must read the transported shared-threshold source rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row before the seal row* S *is available to a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer.*

Proof. Project the accepted selector packet. Its only path from the selected index k to the tail row T passes through the finite window W, the dyadic readback row D, and the structural rows H, C, P, N.

Apply Definition 697.7. The selected observation is admitted through W and D before the tail row can be consumed.

Now apply Definition 697.10. It identifies the supplied T row with the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared-threshold route and places the source, window, threshold, and comparison rows before the selector seal.

Finally use Definition 697.9. A seal consumer reaches S only after those displayed threshold rows have already entered the finite route. □

Theorem 697.12 (Diagonal tail selector source-modulus route exactness)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the exact source and modulus route into the selector ledger is

$$n, \mu, k, W, D, T.$$

Here k *is read from the modulus row* μ *at request* n*,* W *is the admitted finite* $\mathsf{StreamName}^{\uparrow}$ *window for that selected request,* D *is the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback ledger, and* T *is the selected tail row consumed before the seal row* S*.*

Proof. Use the NameCert obligations for the carrier. They make n, μ, k the only request, modulus, and selected-index rows in the local selector certificate.

Apply the window admission theorem. It routes the selected index through the finite window W and the dyadic readback ledger D, so the selector ledger cannot be entered through an unrecorded source row.

The limit handoff theorem then carries the admitted row to T and onward to the seal-facing route. Together with Definition 697.11, this shows that n, μ, k, W, D, T is the exact source/modulus route into the displayed selector ledger. □

Theorem 697.13 (Diagonal tail selector root seal consumer exhaustion)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *root consumer of an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet factors through the finite route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

and, when T *is supplied through* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, through the tail-meet threshold surface of Definition 697.11. No root consumer receives a quotient stream, selected host limit, countable-choice row, hidden tail oracle, host real equality, or ambient completeness coordinate.*

Proof. Begin with Definition 697.12. It fixes the source and modulus part of the root route as n, μ, k, W, D, T inside the carrier packet.

Use Definition 697.11 for the shared-threshold case. The selected tail row can only be consumed after the transported $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ source, window, threshold, and comparison rows are in scope.

Apply Definition 697.9 and Definition 697.8. These factor every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer through the same finite packet before any seal-facing row is read.

Finally apply Definition 697.19. Since the accepted packet contains no quotient stream, selected host limit, countable-choice row, hidden tail oracle, host real equality, or ambient completeness coordinate, a root seal consumer cannot obtain one. □

Theorem 697.14 (Diagonal tail selector RealUp window non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *window read made from an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

consumes the $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *source* R*, tail-selector row* T*, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row before the seal row* S *is available. The window route has no coordinate for a quotient stream, countable choice, host selected limit, host equality, or ambient completeness.*

Proof. Begin with the carrier and NameCert obligations. They expose the finite selector tuple and require the selected index k to be read from μ at the request n.

Apply the window admission theorem. It places every selected observation inside the finite $\mathsf{StreamName}^{\uparrow}$ window W and the $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D before any tail or seal row is consumed.

Use the limit handoff and Definition 697.8. The RealUp seal row is reached only after the window, readback, tail-selector, transport, continuation, provenance, and local NameCert rows have been displayed.

Finally apply the seal non-escape boundary. Since the accepted packet contains no quotient stream, countable-choice row, host selected limit, host equality, or ambient completeness coordinate, no RealUp window consumer can receive one. □

Theorem 697.15 (Diagonal tail selector obligation triad)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

selector exactness, finite-window readback, and $\mathsf{Real}^{\uparrow}$ *seal factorization form one obligation triad over* $\mathsf{LimitSelector}^{\uparrow}$*,* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Any consumer must read* k *from* μ *at* n*, pass through* W, D, T*, and reach* S *only through* H, C, P, N*. The triad supplies no quotient stream, countable-choice row, host selected limit, host equality, or ambient completeness coordinate.*

Proof. First read selector exactness from Definition 697.2. The accepted packet exposes $R, n, \mu, k, W, D, T, S, H, C,$ and the selected index row k is available only as the value read from μ at the precision request n.

Next apply Definition 697.7. It confines every selected observation to the finite $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D before the $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ row T is consumed.

Finally use Definition 697.8. The $\mathsf{Real}^{\uparrow}$ seal row S is reached only through the displayed selector, window, tail handoff, transport, continuation, provenance, and local naming rows.

The non-escape clauses already present in the chapter then rule out quotient streams, countable choice, host selected limits, host equality, and ambient completeness as coordinates of the same route. □

Theorem 697.16 (Diagonal tail selector NameCert closure)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}^{\uparrow}}$ *package closes over the carrier of Definition 697.1, selector exactness, finite-window readback, tail regularity,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and non-escape for the single finite route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

Thus every downstream read factors through the displayed $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows, and the package exports no quotient stream, countable-choice selector, host selected limit, ambient completeness theorem, host real equality, or hidden completed value.*

Proof. Read carrier admission and selector exactness from Definition 697.2. These clauses bind the finite packet and make k a displayed value of μ at n.

Next combine finite-window readback with Definition 697.7. Every selected observation passes through W and D before regular tail information is available.

Apply Definition 697.17 and Definition 697.18. Cofinal reads are exhausted by the displayed window rows, and any shared-threshold tail input is consumed through the transported public certificate.

Finally use Definition 697.8 and Definition 697.19. They place the $\mathsf{CauchyLimitSeal}^{\uparrow}/\mathsf{Real}^{\uparrow}$ consumer behind the same finite route and remove all channels for quotient streams, countable choice, host selected limits, ambient completeness, host real equality, and hidden completed values. □

Theorem 697.17 (Diagonal tail selector cofinal-window exhaustion)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Every cofinal-tail read made by a Real-completion consumer is exhausted by the displayed modulus row* μ*, selected index* k*, finite StreamName window* W*, dyadic readback* D*, regularity row* T*, and structural rows* H, C, P, N*. No such read can be replaced by an unbounded tail search, quotient of regular streams, host limit selection, countable-choice schedule, or ambient completeness argument.*

Proof. Project the accepted carrier. The only rows that carry cofinal-tail data are μ, k, W, D, T together with the structural rows H, C, P, N.

The selector obligation fixes k through n and μ. The finite-window readback obligation then confines every observation to W and D, while the tail-regularity obligation confines the regular-Cauchy comparison to T.

Since the remaining rows only transport, replay, package, and locally name those displayed coordinates, the packet has no source for unbounded search, quotient streams, selected host limits, countable choice, or ambient completeness. □

Theorem 697.18 (Diagonal tail selector tail-meet compatibility)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Suppose the selected cofinal tail row* T *is supplied by an accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet whose source rows, finite windows, dyadic threshold, and comparison row are transported into the diagonal selector route through the same* BHist*, componentwise* hsame *rows, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows recorded by* H, C, P, N*.*

Then every $\mathsf{DiagonalTailSelector}^{\uparrow}$ *limit handoff factors through the public shared-threshold certificate of Definition 560.11 before any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row is read from* S*. The seal-facing consumer receives only the finite route*

$$R, n, \mu, k, W, D, T, H, C, P, N$$

with T *identified as the transported shared-threshold packet, and receives no quotient stream, selected host limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate.*

Proof. Project the accepted carrier G. The route displayed by Definition 697.1 exposes exactly the input row, precision request, modulus, selected index, finite window, dyadic readback, selected tail row, seal row, transports, continuations, provenance, and local name.

Apply Definition 697.6. It presents the diagonal-limit consumer as the finite selector route through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the displayed packet.

Use Definition 697.17. Every cofinal read before the seal is confined to $\mu, k, W, D, T, H, C, P, N$, so the selected tail row cannot be replaced by an unbounded search, quotient stream, host-selected limit, choice schedule, or completeness argument.

Now read the supplied tail-meet packet. By Definition 560.11, its transported source rows, finite windows, dyadic threshold, comparison row, transports, continuations, provenance, and local naming row form the public shared-threshold certificate. Since the hypothesis routes that certificate through the same H, C, P, N rows, the selector row T is consumed as that displayed shared-threshold packet.

Finally apply Definition 697.19. Downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers can reach S only after the finite selector route and the transported shared-threshold row have been displayed, so no quotient stream, selected host limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate is exported. □

Theorem 697.19 (Diagonal tail selector seal non-escape)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet, the* $\mathsf{Real}^{\uparrow}$ *seal row* S *is reachable only through* $R, n, \mu, k, W, D, T, H, C, P, N$*. In particular, any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of* S *receives the finite selector route, finite window, dyadic readback, regular tail handoff, transports, continuations, provenance, and local naming row, and receives no quotient stream, hidden completed value, selected host limit, countable-choice selector, host real equality, or ambient completeness coordinate.*

Proof. Use the NameCert obligations first. They place S downstream of the selected index, finite window, dyadic readback, and tail-regularity row.

Apply the limit handoff theorem. It identifies the completion-facing route as the composition through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows already displayed by the carrier.

The cofinal-window exhaustion theorem removes the only possible alternate tail-read channel. What remains is the finite packet $R, n, \mu, k, W, D, T, S, H, C, P, N$, whose structural rows cannot create quotient streams, hidden completed values, selected host limits, countable choice, host real equality, or ambient completeness. □

Theorem 697.20 (Diagonal tail selector obligation closure package)**.** *The carrier, NameCert obligations, limit handoff, cofinal-window exhaustion, and seal non-escape form the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *obligation package for the finite route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

The package routes through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without exporting quotient streams, host-selected limits, countable choice, host equality, or ambient completeness.*

Proof. Begin with the carrier and NameCert obligations. They expose the finite route $R, n, \mu, k, W, D, T, S, H, C, P$, force k to be read from μ at n, and place the selected observation through W, D, T before the seal row S is available.

Next apply the limit handoff theorem. It identifies the downstream route through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same displayed packet.

Use cofinal-window exhaustion to rule out alternate tail reads. The seal non-escape theorem then confines every $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer to the displayed selector, finite-window, dyadic-readback, tail-regularity, transport, continuation, provenance, and local naming rows.

Thus the listed rows form one obligation package and have no channel for quotient streams, selected host limits, countable choice, host equality, or ambient completeness. □

Theorem 697.21 (Diagonal tail selector source-schedule readback)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

each Real-completion source read accepted by the diagonal tail selector reads the $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, precision request n, displayed modulus μ, and selected cofinal index k before opening the finite window row W, dyadic readback row D, tail handoff row T, or seal row S. The read of k is the one determined by μ at n, and the downstream packet cannot replace it by a second selector, quotient stream, host-selected limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier source schedule readback`

Proof. Project the accepted carrier in Definition 697.1. The visible source-schedule coordinates are exactly R, n, μ, k, W, D, T, S together with the structural rows H, C, P, N.

Apply Definition 697.3. It fixes the selector row by reading k from μ at n before any downstream window, dyadic, tail, or seal row is available.

Use Definition 697.20. Since the obligation package has no coordinate for a second selector, quotient stream, selected host limit, countable choice, host equality, hidden tail oracle, or ambient completeness, every accepted source read uses the displayed R, n, μ, k prefix. □

Theorem 697.22 (Diagonal tail selector window dyadic tail compatibility)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the rows W, D, T *form one compatible finite* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *route before any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal read from* S*. The route is ordered by the window threshold selected from* R, n, μ, k*, admits only the finite window observation named by* W *and* D*, and consumes* T *as the transported regular-Cauchy tail row tied to the same source packet.*

Proof. Start with Definition 697.4. It places the selected cofinal index before the window threshold and prevents the tail or seal clauses from being consumed before the finite window is exposed.

Apply Definition 697.7. The accepted observation is therefore read through the displayed $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D tied to R, n, μ, k.

Now use Definition 697.18. The tail row T is consumed as the transported $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ route over the same source, window, dyadic readback, transport, continuation, provenance, and local naming rows.

The seal-facing consumer sees only this compatible W, D, T route before S. It receives no alternate quotient stream, host-selected limit, countable-choice schedule, host real equality, hidden tail oracle, or ambient completeness coordinate. □

Theorem 697.23 (Diagonal tail selector Real seal source exhaustion)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *seal read from* S *factors through the displayed finite route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

Thus the Real-completion consumer exhausts its source data at the selector row, finite window, dyadic readback, regular-Cauchy tail handoff, seal row, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. It excludes quotient streams, selected host limits, countable choice, host real equality, hidden tail oracle, and ambient completeness.*

Proof. Begin with Definition 697.6. It identifies the completion-facing consumer with the finite route through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the displayed packet.

Apply Definition 697.8. Every $\mathsf{Real}^{\uparrow}$ seal read from S must first pass through $R, n, \mu, k, W, D, T, H, C, P, N$ and then reach the seal row as part of the same accepted packet.

Finally use Definition 697.19. Since the seal boundary exports only the displayed selector, window, dyadic, tail, transport, continuation, provenance, and local naming rows, no quotient stream, selected host limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate can be read by the Real seal. □

Theorem 697.24 (Diagonal tail selector window source scope)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the finite source scope of the Real-completion diagonal tail route is exactly the BHist *carrier rows* R, n, μ, k, W, D, T, S*, the componentwise* hsame *transport rows in* H*, the displayed* Cont *route in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N *over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,*

$\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Every selected source read must first pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D *tied to* R, n, μ, k.

Proof. Project the carrier from Definition 697.1. Its visible coordinates are the finite BHist packet $R, n, \mu, k, W, D, T, S, H, C, P, N$ and no other source row.

Apply Definition 697.7. The selected cofinal index is read from μ at n, then every admitted observation passes through W and D before the tail row T can be consumed.

Use Definition 697.17. It exhausts Real-completion source reads by the displayed finite-window rows and the structural rows H, C, P, N, so the source scope is precisely the named BEDC packet and its sibling window rows. □

Theorem 697.25 (Diagonal tail selector tail-meet consumption scope)**.** *Suppose an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

uses T *as a transported* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet. Then the only tail-meet data consumed by the diagonal route are the transported finite source rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold readbacks,* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row, componentwise* hsame *rows, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows recorded by* H, C, P, N. *The consumption scope exports no quotient stream, host-selected limit, countable-choice row, host real equality, hidden tail oracle, or ambient completeness coordinate.*

Proof. Start from Definition 697.18. It identifies the selected row T with the transported public shared-threshold packet before any $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal row is read.

Next apply Definition 697.24. The tail-meet packet is consumed inside the same $R, n, \mu, k, W, D, T, H, C, P, N$ source scope, with the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows already fixed.

Finally use Definition 697.20. The obligation package has no coordinate for quotient streams, host-selected limits, countable choice, host equality, hidden tail oracles, or ambient completeness, so those objects cannot enter through the tail-meet handoff. □

Theorem 697.26 (Diagonal tail selector finite tail induction)**.** *Every accepted finite tail read of a* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet is obtained by induction over the displayed rows*

$$n, \mu, k, W, D, T, S, H, C, P, N.$$

The base reads the precision and modulus selector row, the successor step extends through the $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, and the terminal case seals through* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *without exporting quotient streams, selected host limits, countable choice, host equality, hidden tail oracles, or ambient completeness.*

Proof. The base case is the selector-row prefix. Apply Definition 697.3 and Definition 697.4: the read starts from n, μ, k, with k determined by μ at n before any downstream window, dyadic row, tail row, or seal row is consumed.

For the successor step, use Definition 697.17. Each finite extension opens only the displayed $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D, with H, C, P, N preserving the same accepted route.

The terminal case is the completion handoff. By Definition 697.6, the read passes through $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ before the seal row is consumed; by Definition 697.19, the final $\mathsf{Real}^{\uparrow}$ consumer receives only the displayed finite packet and none of the refused host objects. □

Theorem 697.27 (Diagonal tail selector root cofinal admission)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted DiagonalTailSelector$^{\uparrow}$ *packet. Before any selector read is performed, the root admission surface exposes the source regular row R, the precision request n, the displayed modulus μ, the selected threshold index k, the finite stream window W, the dyadic readback row D, the tail-selector handoff T, the Real seal row S, the componentwise* hsame *rows H, the displayed* Cont *routes C, the* Pkg *provenance row P, and the local* NameCert *row N. No selector step may consult a row outside this displayed packet.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier root cofinal admission`

Proof. Project the accepted carrier of DiagonalTailSelector$^{\uparrow}$. Its root packet is the tuple G, and the selector coordinate is read only after the visible source, modulus, window, readback, tail, seal, transport, continuation, provenance, and naming rows have been fixed.

Apply Definition 697.7 to the first selector read. The theorem routes the read through μ at n and then through W and D, so no hidden source row is available before the cofinal index is selected. □

Theorem 697.28 (Diagonal tail selector root budget handoff)**.** *For every accepted* DiagonalTailSelector$^{\uparrow}$ *packet G, the selected cofinal row factors through the visible L10 route*

LimitSelector$^{\uparrow}$, RegularCauchyTailSelector$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$.

The factorization consumes only the rows exposed by Definition 697.27: the cofinal index comes from the displayed selector/modulus budget, the finite window and dyadic readback supply the stream evidence, the regular Cauchy tail row transports the shared threshold, and the Real row receives only the resulting finite seal packet.

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorRootBudgetHandoff`

Proof. Start with the root admission surface of Definition 697.27. The selector row is already tied to $R, n, \mu, k, W, D, T, S, H, C, P, N$, so any downstream budget handoff must reuse these rows.

Next apply Definition 697.6, Definition 697.12, and Definition 697.25. Together they route the cofinal selection through the displayed limit selector, regular tail-selector, stream-name, dyadic readback, regular sequence, and Real seal rows.

It remains to identify the budget boundary. The handoff contains no row besides the root packet and the named L10 consumers, so the selected cofinal row factors through exactly the displayed route. □

Theorem 697.29 (Diagonal tail selector root seal non-escape)**.** *The root budget route of an accepted* DiagonalTailSelector$^{\uparrow}$ *packet exports no quotient stream, countable-choice selector, host-selected limit, host real equality, hidden tail oracle, or ambient completeness row. Its public output is only the finite cofinal-window, dyadic-readback, regular-tail, Real-seal,* hsame, Cont, Pkg, *and* NameCert *surface named by the root packet.*

Proof. Use Definition 697.28 to place every selected cofinal row on the displayed L10 route. Each consumer receives only the rows exposed by the accepted root packet.

Now apply Definition 697.23 and Definition 697.25. These results exhaust the Real seal and tail-meet consumption boundaries and list the objects that cannot be exported through them.

The remaining rows H, C, P, N are structural transport, continuation, provenance, and naming rows. They record the visible finite route rather than adding quotient streams, countable choice, host limits, host equality, hidden tail oracles, or completeness data. □

Theorem 697.30 (Diagonal tail selector cofinal budget replay)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the same cofinal budget is replayed through

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$.

The replay first reads the root admission rows $R, n, \mu, k, W, D, T, S, H, C, P, N$, then consumes the selected cofinal index and finite window through the displayed L10 route, and finally checks compatibility with the tail budget of Definition 697.45. No consumer may replace this replay by a fresh selector, quotient stream, hidden tail row, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier cofinal budget replay`

Proof. Begin with Definition 697.27. It projects the accepted packet G and fixes the source regular row, precision request, modulus, threshold, finite window, dyadic readback, tail handoff, Real seal, transport, route, provenance, and naming rows before any selector read is performed.

Apply Definition 697.28. The selected cofinal row is routed through the displayed L10 consumers $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ using only the rows named by the root packet.

Now use Definition 697.45. The public budget source and the tail budget consume the same rows $R, n, \mu, k, W, D, H, C, P, N$ before S is read, so the replay cannot fork into a separate selector or hidden tail source.

Finally invoke Definition 697.29. The root seal has no coordinate for quotient streams, countable choice, host-selected limits, host real equality, hidden tail oracles, or ambient completeness. Hence the replay is exactly the displayed cofinal budget route. □

Theorem 697.31 (Diagonal tail selector public root replay package)**.** *Every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

whose public budget source reads the root cofinal admission row of Definition 697.27, the root budget handoff row of Definition 697.28, and the cofinal budget replay row of Definition 697.30 exports exactly the finite route

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

through

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}$

The package exports no quotient stream, selected host limit, countable-choice schedule, host real equality, ambient completeness principle, or hidden tail oracle.

Proof. Project the root cofinal admission row. It fixes the accepted packet $G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$ as the full carrier inventory available to the public budget source.

Apply the root budget handoff row next. The handoff places the selected cofinal read on the displayed L10 factorization through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Use the cofinal budget replay row to reuse the same finite budget through that consumer chain. The replay does not allocate a second selector, stream, or host limit row; it only replays the rows already named by the root packet.

Finally combine the public budget export and non-escape rows. They identify the exported object as the finite route above and exclude quotient streams, selected host limits, countable-choice schedules, host real equality, ambient completeness, and hidden tail oracles from the public replay package. □

Theorem 697.32 (Diagonal tail selector budget readback non-escape)**.** *The readback exported by the cofinal budget replay of Definition 697.30 is only the finite root packet*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

together with its displayed L10 consumer route. It exports no quotient stream, choice selector, host-selected limit, host real equality, hidden tail oracle, or ambient completeness row.

Proof. Use Definition 697.30 to put the readback on the same cofinal budget route through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The root packet contributes exactly the rows listed in Definition 697.27. The budget handoff of Definition 697.28 then consumes those rows without adding any external selector coordinate.

For the tail-facing part, apply Definition 697.45. It identifies the shared finite budget consumed by the schedule and tail-meet routes before the Real seal row is read.

The excluded objects are precisely the objects ruled out by Definition 697.29. Since the readback has no row beyond the accepted packet and its displayed L10 route, none of them can be exported. □

Theorem 697.33 (Diagonal tail selector terminal budget determinacy)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Any terminal* $\mathsf{Real}^{\uparrow}$ *consumer, and any completion-facing consumer that reads the selected cofinal row, factors through the same displayed root budget route*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

and then through

$$\mathsf{LimitSelector}^{\uparrow}, \quad \mathsf{RegularCauchyTailSelector}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

No such consumer may replace the terminal budget by a fresh selector, quotient stream, host-selected limit, hidden tail row, or ambient completeness witness.

Proof. Start with Definition 697.27. It fixes the accepted packet G as the only root surface available before the selected cofinal row is read.

Apply Definition 697.28. The selected cofinal row is handed to the terminal route through the displayed L10 consumer chain, so a Real or completion-facing read has no second budget coordinate.

Replay the route with Definition 697.30. The replay uses the same root rows and the same named consumers, and Definition 697.45 identifies the tail-facing threshold with that displayed budget.

Finally apply Definition 697.32. Since the terminal readback exports only $R, n, \mu, k, W, D, T, S, H, C, P, N$ and its displayed L10 route, it cannot substitute a fresh selector, quotient stream, host-selected limit, hidden tail row, or ambient completeness witness. □

Theorem 697.34 (Diagonal tail selector root formal target surface)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. The Lean-facing formal target surface for* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers is exactly the finite packet*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

together with the displayed route through

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}$

It contains no quotient stream, countable choice operator, host-selected limit, host real equality, hidden tail oracle, or ambient completeness principle.

Proof. Begin with Definition 697.27. It fixes the accepted root surface to the coordinates of G and the named cofinal route, so every downstream consumer starts from the same finite packet.

Use Definition 697.28 to hand those coordinates to the consumer route. Then apply Definition 697.30 to replay the same budget through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The public packaging is supplied by Definition 697.31. The readback boundary of Definition 697.32 then rules out quotient streams, countable choice, host-selected limits, host real equality, hidden tail oracles, and ambient completeness as exported coordinates.

Finally, Definition 697.33 identifies every terminal Real or completion-facing read with that same route. Carrier-coordinate exhaustion leaves exactly the listed packet and route, with no remaining coordinate from which an excluded object could be read. □

Theorem 697.35 (Diagonal tail selector root budget route totality)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Any root consumer that uses the admission row, the root budget handoff row, the cofinal budget replay row, and the terminal formal-target row consumes exactly the packet* G *and the displayed L10 route through*

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}$

No additional root route, tail-selector coordinate, quotient stream, host limit selector, or ambient completeness witness is part of that consumer surface.

Proof. Apply Definition 697.27. It fixes the admitted root inventory to the coordinates $R, n, \mu, k, W, D, T, S, H, C, P$ and to the named cofinal consumer chain.

Next apply Definition 697.28. The handoff row sends exactly that admitted inventory into the root budget route, so a root consumer has no separate budget coordinate before replay.

Replay the budget by Definition 697.30. The replay row keeps the same root coordinates and the same L10 route through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use Definition 697.34. Its formal-target inventory is exactly the displayed packet and route, and its negative clauses rule out any extra route or hidden selected-limit coordinate. □

Theorem 697.36 (Diagonal tail selector root budget transport)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Componentwise displayed* hsame *transports of the admission, handoff, replay, and terminal formal-target rows, together with matching* Cont *routes for the L10 consumer chain, preserve the same cofinal budget route from* G *through*

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}$

Such transports add no quotient stream, hidden selector, host real equality, or second budget ledger.

Proof. Read the transport data from the root admission inventory of Definition 697.27. The only admitted transports are the displayed componentwise hsame rows and the matching Cont routes already recorded in H and C.

Use Definition 697.28 to identify the transported handoff with the same root budget coordinates. Then apply Definition 697.30 to replay the transported route without changing its L10 spine.

The determinacy clauses in Definition 697.33 identify any terminal read of the transported route with the replayed route. By Definition 697.34, the resulting formal-target surface still contains only G and the displayed consumer chain, not a quotient stream, hidden selector, host equality row, or additional ledger. □

Theorem 697.37 (Diagonal tail selector root budget ledger exactness)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. The coordinates* P *and* N *account exactly for the admission, handoff, replay, and terminal formal-target rows of the root budget route. They cannot be read as selected limits, host real equality, ambient completeness, quotient-stream identification, or countable choice.*

Proof. Begin with Definition 697.35. It reduces every root consumer using the displayed rows to the single packet G and the single L10 cofinal budget route.

Apply Definition 697.32. The budget readback permits P and N to certify provenance and local naming for the finite route, but it does not expose selected limits, host real equality, ambient completeness, quotient-stream identification, or countable choice.

Finally use Definition 697.38. Since terminal reads are exhausted by G and the displayed route, the ledger coordinates have no remaining coordinate from which an excluded object or principle could be recovered. □

Theorem 697.38 (Diagonal tail selector terminal cofinal route exhaustion)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of the cofinal selector route is exhausted by the displayed packet* G *together with the L10 route through*

$\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}$

No quotient stream, countable-choice selector, host-selected limit, host real equality, hidden tail oracle, or ambient completeness row can be recovered from such a terminal consumer.

Proof. First apply Definition 697.27 and Definition 697.28. These identify the accepted root surface with the displayed coordinates of G and send that same finite budget into the terminal route.

Next use Definition 697.30. The replay keeps the cofinal route on the named L10 spine through $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The public surface is then the one supplied by Definition 697.31. By Definition 697.32, a terminal readback from this surface cannot export a quotient stream, countable-choice selector, host-selected limit, host real equality, hidden tail oracle, or ambient completeness row.

Finally apply Definition 697.33 and Definition 697.34. They identify every terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer with the same packet and route. Carrier-coordinate exhaustion therefore leaves exactly the displayed coordinates and no remaining carrier from which one of the excluded objects could be read. □

Theorem 697.39 (Diagonal tail selector terminal cofinal route uniqueness)**.** *Let*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N)$$

be an accepted $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet. Suppose two terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumers of G both pass through the displayed L10 spine*

$$\mathsf{LimitSelector}^{\uparrow},\quad \mathsf{RegularCauchyTailSelector}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{CauchyLimitSeal}$$

Then their terminal cofinal route is the same displayed packet-and-spine route, up to the hsame *transports already carried by G. No second selector, quotient stream, hidden tail oracle, host-selected limit, or ambient completeness row can distinguish the two terminal consumers.*

Proof. Apply Definition 697.27 and Definition 697.28. Both terminal consumers are admitted from the same accepted root packet, so their visible coordinates are the coordinates of the displayed G and their transports are the hsame transports already present in that packet.

Next apply Definition 697.30. Since both consumers pass through the named L10 spine, the replay supplies the same finite cofinal budget at each listed selector, stream, dyadic, regular sequence, seal, and real-facing row.

Now use Definition 697.33 and Definition 697.34. These identify the terminal read of each consumer with the same public packet-and-spine route, leaving only the already displayed hsame transports as admissible coordinate changes.

Finally invoke Definition 697.38. The exhaustion theorem removes every non-displayed coordinate, so no second selector, quotient stream, hidden tail oracle, host-selected limit, or ambient completeness row remains available to distinguish the two terminal routes. □

Theorem 697.40 (Diagonal tail selector TailCofinalitySchedule factorization)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

each Real-seal consumer of S factors first through the $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *route from unary precision to finite window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before using the diagonal selector rows. Thus the consumer reads the precision request and finite-window schedule before it uses $k, W, D, T, S, H, C, P, N$ as the diagonal completion-facing packet.*

Proof. Begin with Definition 681.3. It orders the schedule-facing route from the unary precision row through the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the $\mathsf{Real}^{\uparrow}$ seal.

Apply Definition 697.6. It identifies the diagonal Real-completion consumer with the finite packet $R, n, \mu, k, W, D, T, S, H, C, P, N$, so the schedule route is consumed before the selector rows are used as a completion-facing package.

Finally use Definition 697.25. The transported tail-meet rows remain inside the same displayed source, window, dyadic, regular-rational, seal, transport, route, provenance, and naming surface. Hence the factorization has no escape channel through quotient streams, selected limits, countable choice, ambient completeness, host real equality, or hidden tail oracles. □

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier tail cofinality factorization`

Theorem 697.41 (Diagonal tail selector Real seal boundary scope)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet*

$$G = (R, n, \mu, k, W, D, T, S, H, C, P, N),$$

the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary reached at S is scoped by the finite source rows of Definition 697.24 and the transported tail-meet rows of Definition 697.25. In particular, the seal-facing boundary is the displayed BEDC route through* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *with S reached only after $R, n, \mu, k, W, D, T, H, C, P, N$ are in scope.*

Proof. Apply Definition 697.8. Every $\mathsf{Real}^{\uparrow}$ seal read from S factors through the finite selector packet and its transport, continuation, provenance, and local naming rows.

Use Definition 697.24 to identify the source side of that packet. The rows R, n, μ, k, W, D supply exactly the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ source window before T and S are available.

Use Definition 697.25 for the regular-tail side. When T is supplied by the shared-threshold route, it is consumed only as the transported $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ certificate inside the same structural rows.

Finally apply Definition 697.19. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary therefore has no escape channel outside the displayed BEDC route. □

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier real seal boundary scope`

Theorem 697.42 (Diagonal tail selector downstream non-escape scope)**.** *Every downstream consumer of the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *Real-completion route is scoped by the same finite packet*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

and the sibling BEDC rows named in Definition 697.24, Definition 697.25, and Definition 697.41. Hence a downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can use only the displayed* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows, and cannot export quotient streams, host-selected limits, countable choice, host real equality, hidden tail oracles, or ambient completeness.*

Proof. Begin with Definition 697.20. It packages the carrier, NameCert obligations, limit handoff, cofinal-window exhaustion, and seal non-escape over one finite route.

The window source scope then supplies the finite source boundary. Apply Definition 697.24 to keep all selected reads inside R, n, μ, k, W, D and the structural rows H, C, P, N.

The tail-meet consumption scope supplies the regular-tail boundary. Apply Definition 697.25 to consume T only as the transported shared-threshold certificate when that sibling route is present.

The Real seal boundary scope supplies the completion-facing boundary. Apply Definition 697.41 and then Definition 697.19; downstream consumers are confined to the displayed BEDC route and receive none of the excluded escape coordinates. □

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorPublicBudgetSource downstream nonescape scope`

Definition 697.43 (Diagonal tail selector public budget source)**.** *The public budget source for an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet is the finite carrier*

$$(R, n, \mu, k, W, D, T, S, H, C, P, N)$$

together with the displayed sibling rows $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Its exported source consists exactly of the precision request, modulus row, selected cofinal index, finite window, dyadic readback, transported tail-meet certificate, seal row, componentwise* hsame *rows, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}^{\uparrow}}$ *row.*

No quotient stream, selected host limit, countable-choice schedule, host real equality, ambient completeness principle, or hidden tail oracle is a coordinate of this public budget source.

Lean4 (definition) `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorPublicBudgetSource`

Theorem 697.44 (Diagonal tail selector window choice totality)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet, the public budget source of Definition 697.43 makes the selected finite window total as a BEDC row: the precision request n and modulus row μ determine the displayed cofinal index k, and k opens the finite* $\mathsf{StreamName}^{\uparrow}$ *window W and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback D used by every downstream seal consumer.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier window choice totality`

Proof. Project the carrier of Definition 697.1. The rows $R, n, \mu, k, W, D, T, S, H, C, P, N$ are all displayed before a seal consumer can read S.

Apply the window-source scope of Definition 697.24. It binds selected reads to the finite source rows n, μ, k, W, D and the structural rows H, C, P, N.

Finally use the TailCofinalitySchedule factorization of Definition 697.40. The schedule-facing route reads the precision request and finite window before the completion-facing selector packet is consumed, so the window choice is a total displayed row and not a quotient stream, selected host limit, countable-choice schedule, host real equality, or ambient completeness row. □

Theorem 697.45 (Diagonal tail selector tail budget compatibility)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet whose row T is the transported* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold certificate, the public budget source of Definition 697.43 is compatible with the tail budget: the same finite rows $R, n, \mu, k, W, D, H, C, P, N$ feed the TailCofinalitySchedule factorization and the tail-meet consumption route before the Real seal row S is read.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorPublicBudgetSource tail budget compatibility`

Proof. Start with Definition 697.40. It places the schedule route from precision request through finite window, dyadic ledger, regular-rational readback, and Real seal ahead of the completion-facing selector read.

Next apply Definition 697.25. The row T is consumed only as the transported shared-threshold packet inside the same source, transport, route, provenance, and naming surface.

Combine the two routes through Definition 697.43. Since the public source has no coordinate for quotient streams, selected host limits, countable-choice schedules, host real equality, ambient completeness, or hidden tail oracles, compatibility is exactly the equality of the displayed budget rows consumed by both routes. □

Theorem 697.46 (Diagonal tail selector Real seal budget non-escape)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet that reads the public budget source of Definition 697.43 remains inside the displayed rows* $R, n, \mu, k, W, D, T, S, H, C, P, N$ *and the listed sibling* $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. It cannot export quotient streams, selected host limits, countable-choice schedules, host real equality, ambient completeness, or hidden tail oracles.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier real seal budget nonescape`
Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorPublicBudgetSource real seal budget nonescape`

Proof. Use the Real seal boundary scope of Definition 697.41. It places the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ boundary at S only after the finite source rows and transported tail-meet rows are already in scope.

Apply Definition 697.19. The seal-facing boundary has no escape channel outside the displayed BEDC route.

Finally apply the downstream non-escape scope of Definition 697.42. Every downstream completion consumer therefore receives only the public budget source rows and listed sibling BEDC rows, with no quotient stream, selected host limit, countable-choice schedule, host real equality, ambient completeness, or hidden tail oracle. □

Theorem 697.47 (Diagonal tail selector public budget export)**.** *The public budget source of Definition 697.43 exports exactly the finite route through the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet and the sibling schedules of chapter 681, chapter 678, and chapter 560 before any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the seal row* S*. Thus the exported route is precisely*

$$R, n, \mu, k, W, D, T, S, H, C, P, N$$

together with the displayed precision, modulus, cofinal-window, dyadic-readback, regular-rational, tail-meet, seal, continuation, provenance, and local naming rows. It exports no quotient stream, host-selected limit, countable choice schedule, host real equality, ambient completeness, or hidden tail oracle.

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier public budget export`

Proof. First apply Definition 697.40. The Real-completion consumer reaches the seal-facing selector only after the precision request, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and $\mathsf{Real}^{\uparrow}$ seal route have been ordered by the TailCofinalitySchedule schedule named in chapter 681.

Next use Definition 697.45. The same finite rows $R, n, \mu, k, W, D, H, C, P, N$ feed the selector budget and the transported tail-meet route, so the budget surface is the one shared with chapter 678 and chapter 560, with T consumed only as the displayed shared-threshold certificate.

Finally apply Definition 697.46. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers of S stay inside the public budget source and the listed sibling rows. Consequently the export contains exactly the displayed finite route and none of the excluded quotient, choice, host-equality, completeness, or hidden-oracle data. □

Theorem 697.48 (Diagonal tail selector public budget threshold order)**.** *For every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public budget source of Definition 697.43, any downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the source rows*

$$n, \mu, k, W, D$$

before the transported tail row T, before the seal row S, and before the finite observation-budget selector route of chapter 782 is consumed. Thus the public budget surface inherits the checked window-threshold order of the carrier before it exports a seal-facing or finite-observation route.

Proof. Start with the carrier-level window-threshold order of Definition 697.4. It places the precision row n, modulus row μ, selected cofinal index k, finite window row W, and dyadic readback row D before any read of the tail-regularity row T or the seal row S.

Next apply the public budget export of Definition 697.47. The exported source is exactly the finite route

$$R, n, \mu, k, W, D, T, S, H, C, P, N.$$

Since the export preserves the displayed carrier order, the public consumer cannot read T or S before n, μ, k, W, D.

Finally use the non-escape boundary of Definition 697.46. Every $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer remains inside the same public budget source, and the finite observation-budget selector handoff must be a read of that already ordered source rather than a route that bypasses the window-threshold rows. □

Theorem 697.49 (Diagonal tail selector finite observation-budget handoff)**.** *Every accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public budget source of Definition 697.43 whose displayed rows are read as a finite observation-budget selector factors through the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *route of chapter 782 before any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the seal row. Equivalently, the consumer first reads the finite route*

$$B \to S_F \to W_F \to D_F \to R_F \to E_F$$

given by Definition 782.14 and Definition 782.10, and only then consumes the $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public export of Definition 697.47. The handoff exports no quotient stream, host-selected limit, countable-choice schedule, host real equality, ambient completeness, or hidden tail oracle, by Definition 697.46.*

Proof. First identify the displayed $\mathsf{DiagonalTailSelector}^{\uparrow}$ rows. By Definition 697.47, an accepted public budget source exports exactly the finite selector packet $R, n, \mu, k, W, D, T, S, H, C, P, N$ and the listed sibling schedule, tail-meet, seal, stream, dyadic, regular-rational, and Real rows before a seal consumer reads S.

Next read those rows through the finite observation-budget selector. Apply Definition 782.14 to identify the public route $B \to S_F \to W_F \to D_F \to R_F \to E_F$, and then use Definition 782.10

to make every diagonal-limit or Real-completion consumer read that route before the seal row is consumed.

Finally use the DiagonalTailSelector non-escape boundary. By Definition 697.46, the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers remain inside the displayed public budget source. Thus the handoff cannot pass through quotient streams, host-selected limits, countable choice, host real equality, ambient completeness, or hidden tail oracles. □

Theorem 697.50 (Diagonal tail selector public budget route totality)**.** *Any downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *public budget source must read the finite observation-budget route first, then the selector public budget export, and then the Real seal boundary. The route is entirely through* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows; it has no quotient stream, host-selected limit, countable-choice schedule, host real equality, ambient completeness, or hidden tail oracle.*

Proof. Begin with the finite observation-budget handoff of Definition 697.49. It forces every seal-facing consumer to read the finite selector route before the $\mathsf{DiagonalTailSelector}^{\uparrow}$ seal row is consumed.

Next apply the public budget export of Definition 697.47. This identifies the selector source as the displayed BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface.

Finally use Definition 697.46. The Real and Cauchy-limit readers stay inside that displayed route, so the consumer-facing path is total for the public budget handoff and excludes the listed quotient, choice, host-equality, completeness, and hidden-oracle data. □

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier public budget route totality`

Theorem 697.51 (DiagonalTailSelector schedule-seal consumer exactness)**.** *For an accepted* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *packet, the completion-facing consumer route from the displayed schedule rows to the final seal is exact exactly when it factors through the finite rows* $R, n, \mu, k, W, D, T, S, H, C, P, N$ *and the sibling schedule rows supplied by* $\mathsf{TailCofinalitySchedule}^{\uparrow}$*,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. No quotient stream, host-selected limit, countable-choice schedule, host real equality, ambient completeness, or hidden tail oracle can be substituted for any segment of that route.*

Lean4 ✓ `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorScheduleSealConsumerExactness`

Proof. For the forward direction, begin with the tail-cofinality factorization of Definition 697.40. It places the consumer route on the displayed $\mathsf{TailCofinalitySchedule}^{\uparrow}$ rows before any seal-facing read is allowed.

Next apply tail-budget compatibility from Definition 697.45. The scheduled tail rows, finite windows, dyadic ledger, transport rows, continuation routes, package provenance, and local name are the same rows that the selector public budget exposes.

Then use real-seal budget non-escape from Definition 697.46 and the public budget export of Definition 697.47. Together they identify the only completion-facing export as the displayed $R, n, \mu, k, W, D, T, S, H, C, P, N$ route with its $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling rows.

For the reverse direction, apply the public budget route totality theorem Definition 697.50. A consumer that factors through exactly those finite rows already reads the finite observation-budget route, the selector public budget export, and the Real seal boundary in the required order.

The same non-escape boundary rules out every alternate carrier named in the statement. Hence no quotient stream, host-selected limit, countable-choice schedule, host real equality, ambient completeness, or hidden tail oracle can replace a segment while preserving exactness of the schedule-to-seal consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}\uparrow}$.

Constructive story: A $\mathsf{DiagonalTailSelector}^{\uparrow}$ packet is generated from a RegSeqRat input row, unary precision request, displayed modulus, selected cofinal tail index, finite StreamName window, DyadicRatCore readback, RegularCauchyTailSelector handoff, RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 697.1, Definition 697.2, finite-window readback, tail-meet consumption, Real seal boundary, downstream non-escape, and Definition 697.43 export the finite-window, tail-meet, Real seal, and non-escape package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalTailSelector}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DiagonalTailSelectorUp.DiagonalTailSelectorCarrier\_source\_schedule\_readback`

Bridge status: none

Origin: human

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, host real equality, countable choice, host-selected limits, ambient completeness, or a public standard bridge.

Upgrade path: No further paper-side theory closure upgrade remains for the finite-window, tail-meet, Real-seal, and downstream non-escape package.

Chapter 698

A Concrete Naming Certificate for $\mathsf{CofinalTailBudget}^{\uparrow}$

The $\mathsf{CofinalTailBudget}^{\uparrow}$ packet records the finite shared-tail threshold used when a Real and Completion diagonal-limit argument must move from a cofinal family of tail windows to the displayed $\mathsf{Real}^{\uparrow}$ seal. It does not choose an infinite subsequence or identify a real number by quotienting. It names the finite budget row, the selected tail-window rows, the RegSeqRat handoff, and the RealUp seal route that later consumers may replay. It consumes the cofinal-subsequence window handoff surface of chapter 405 but restricts it to the finite shared-tail budget rows M, T, R, S, H, C, P, N.

Definition 698.1 (Cofinal tail budget carrier)**.** *A* $\mathsf{CofinalTailBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, T, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite modulus-budget row naming the requested precision level and the bound from which tail windows are read;*
2. *T is the cofinal tail-window row, listing the finitely many windows whose anchors are at or beyond the bound recorded by M;*
3. *R is the* $\mathsf{RegSeqRat}$ *handoff row that reads each selected tail window as a regular rational observation;*
4. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed by the diagonal-limit handoff;*
5. *H records componentwise* hsame *transport for M, T, R, S, P, N and the local packet shape;*
6. *C records the* Cont *routes by which a consumer reads the budget, selected windows, RegSeqRat handoff, RealUp seal, package provenance, and local naming row;*
7. *P is the* Pkg *provenance over* BHist*, finite modulus budgets, cofinal tail windows,* $\mathsf{RegSeqRat}$ *handoff rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* hsame*,* Cont*, and non-escape ledgers;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailBudget}^{\uparrow}}$ *row.*

The carrier contains no global completeness theorem, no choice of an infinite tail selector, and no quotient-level real equality. It exports only the finite shared-tail threshold packet displayed by M, T, R, S, H, C, P, N.

Theorem 698.2 (Cofinal tail budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CofinalTailBudget}^{\uparrow}}$ *surface for Definition 698.1 consists of five finite obligation rows:*

1. *carrier admission for the packet* $B = (M, T, R, S, H, C, P, N)$*;*
2. *budget stability stating that componentwise* hsame *transport preserving* M, T, R, S, C, P, N *preserves the same shared-tail threshold;*
3. *window exactness stating that every selected tail window in* T *is read at or beyond the budget bound recorded by* M*;*
4. *seal handoff stating that the* RegSeqRat *row* R *is the only route by which the selected windows feed the* $\mathsf{Real}^{\uparrow}$ *seal row* S*;*
5. *ledger exactness stating that every public read factors through* M, T, R, S, H, C, P, N *and no unrecorded cofinal selector or completeness principle is exported.*

Lean4 ✓ `BEDC.Derived.CofinalTailBudgetUp.CofinalTailBudgetCarrier namecert obligations`

Proof. Carrier admission is projection from the displayed packet. The tuple lists the modulus budget, selected tail windows, RegSeqRat handoff, RealUp seal, transports, routes, provenance, and local naming row.

Budget stability is read through H and C. Since the budget and window rows are transported together, the shared-tail threshold named by M remains attached to the same selected windows in T.

Window exactness is the displayed role of T. Each public window read passes through C, and the package provenance P records that the anchor lies at or beyond the finite bound supplied by M.

The seal handoff and ledger exactness are the remaining package clauses. A consumer reaches S only through the RegSeqRat handoff row R, and every exported read remains a projection of M, T, R, S, H, C, P, N. □

Theorem 698.3 (Cofinal tail budget window coverage)**.** *For an accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet* $B = (M, T, R, S, H, C, P, N)$*, every public tail-window read covered by the packet is one of the finite windows listed in* T*, and each such read carries the shared-tail threshold recorded by* M*. A proposed window whose anchor is below the budget bound is not covered by this packet.*

Proof. Project the carrier to obtain the budget row M and the finite window row T. These two rows are paired in the same packet, so a covered read cannot detach the selected window from the displayed threshold.

Read the requested window through the continuation row C. If the request names a listed window, the route returns the corresponding T coordinate together with the budget row M and the RegSeqRat handoff R.

If the proposed anchor lies below the bound named by M, no coordinate of T records it as covered. The package provenance P therefore keeps the request outside the packet's coverage surface. □

Lean4 ✓ `BEDC.Derived.CofinalTailBudgetUp.CofinalTailBudgetCarrier window read transport`

Theorem 698.4 (Cofinal tail budget Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet factors through the selected tail windows* T *and the* RegSeqRat *handoff row* R*. The seal row* S *is therefore consumed as a finite diagonal-limit handoff from the displayed shared-tail threshold, not as an independent completeness claim.*

Proof. Start from the carrier inventory. The only route to the RealUp seal coordinate is the continuation row C reading the displayed seal row S in the same packet as M, T, R, H, P, N.

The RegSeqRat handoff is the intermediate row. A seal consumer first reads the selected windows in T, then reads those windows through R, and only then reaches S.

Thus the exported seal is bound to the finite threshold and finite window ledger. Since the packet contains no global completeness theorem or quotient-level equality, the seal read is exactly the displayed handoff route. □

Theorem 698.5 (Cofinal tail budget non-escape)**.** *An attempted Real and Completion tail read that omits the budget row M, a listed window from T, or the* RegSeqRat *handoff row R is outside the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet. It cannot export the* $\mathsf{Real}^{\uparrow}$ *seal row S and cannot be repacked as a cofinal shared-tail certificate.*

Proof. Read the attempted route through the finite carrier. The packet exposes only M, T, R, S, H, C, P, N, so any route omitting the budget, selected-window, or handoff coordinate lacks a required source row.

Apply the window coverage row for omissions of M or T. The request then has no covered tail-window coordinate at or beyond the shared threshold.

Apply the seal handoff row for omissions of R. A consumer cannot reach S before the selected windows have been read as RegSeqRat handoff data.

The package provenance P records the remaining attempt as outside the local naming surface. Hence the route cannot export S or appear as a cofinal shared-tail certificate. □

Theorem 698.6 (Cofinal tail budget stability)**.** *Let $B = (M, T, R, S, H, C, P, N)$ be an accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet. Componentwise* hsame *transport recorded by H, and read through the displayed continuation routes C, preserves the same finite budget packet M, T, R, S, P, N. In particular, the shared-tail threshold named by M, the selected windows in T, the* RegSeqRat *handoff R, the* $\mathsf{Real}^{\uparrow}$ *seal row S, the package provenance P, and the local naming row N remain attached to one another after transport.*

Proof. Project the carrier to the transport row H. Its role in the packet is componentwise: it relates the displayed budget, window, handoff, seal, provenance, and naming coordinates without adding a second budget source.

Read the transported data through C. Each public route still lands in the same coordinates M, T, R, S, P, N, so the threshold in M cannot detach from the finite window list T or from the handoff and seal rows R, S.

The package row P records this as provenance for the same local packet, and N names that packet. Hence hsame transport preserves the displayed shared-tail budget packet rather than producing a new tail selector or a quotient witness. □

Theorem 698.7 (Cofinal tail budget ledger exactness)**.** *For an accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet $B = (M, T, R, S, H, C, P, N)$, every public read factors through the rows M, T, R, S, H, C, P, N. The exported ledger therefore excludes hidden cofinal selectors, quotient equality witnesses, ambient completeness rows, and any public read not routed through the displayed finite budget packet.*

Proof. Start with a public read of the packet. The carrier definition exposes only the eight displayed rows, and the continuation row C gives the public routes by which those rows are read.

If the read asks for budget, selected-window, handoff, seal, transport, provenance, or naming data, it is a projection through the corresponding coordinate. These projections are exactly the ledger entries named by M, T, R, S, H, C, P, N.

If the read asks for a hidden cofinal selector, a quotient equality witness, or an ambient completeness row, no displayed coordinate supplies it. The ledger therefore rejects that request as outside the local $\mathsf{CofinalTailBudget}^{\uparrow}$ surface. □

Theorem 698.8 (Cofinal tail budget obligation closure)**.** *The accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet closes its local obligation surface: carrier admission, budget stability, window coverage,* $\mathsf{Real}^{\uparrow}$ *seal handoff, ledger exactness, and non-escape are all witnessed by the same finite packet* $B = (M, T, R, S, H, C, P, N)$*.*

Proof. Carrier admission is the tuple displayed in Definition 698.1, and the NameCert surface of Definition 698.2 lists the five obligation rows that must be read from it.

Budget stability is supplied by Definition 698.6. Window coverage is supplied by Definition 698.3, and the $\mathsf{Real}^{\uparrow}$ handoff is supplied by Definition 698.4.

Ledger exactness is supplied by Definition 698.7. Non-escape is supplied by Definition 698.5. These clauses all mention the same rows M, T, R, S, H, C, P, N, so the obligation surface is assembled without adding a hidden selector, quotient equality, or ambient completeness principle. □

Theorem 698.9 (Cofinal tail budget primitive-scope binding)**.** *For every accepted* $\mathsf{CofinalTailBudget}^{\uparrow}$ *packet* $B = (M, T, R, S, H, C, P, N)$*, the budget row, selected tail windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row are bound entirely through the displayed BEDC primitives* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the cited cofinal-subsequence sibling surface of chapter 405. The binding exports no hidden cofinal selector, quotient equality witness, ambient completeness row, or nonlocal naming route.*

Lean4 ✓ `BEDC.Derived.CofinalTailBudgetUp.CofinalTailBudgetUp StdBridge`

Proof. Start with the carrier definition. The packet exposes only the displayed rows M, T, R, S, H, C, P, N, and Definition 698.2 assigns the local naming obligations to those rows.

The budget and window clauses are fixed by Definition 698.6 and Definition 698.3. They keep the finite budget row, selected tail windows, and componentwise transports inside the same BHist packet.

The Real-facing route is fixed by Definition 698.4. It sends selected windows through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff before the $\mathsf{Real}^{\uparrow}$ seal is read, so the seal cannot detach from the displayed tail-budget packet.

Ledger exactness and non-escape are supplied by Definition 698.7 and Definition 698.5. Together with the existing TasteGate marker, these clauses say that every accepted read remains a projection of the same carrier surface and every nonlisted selector, quotient witness, completeness row, or nonlocal naming route is outside the chapter scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailBudget}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CofinalTailBudget}^{\uparrow}$ packet is generated from a finite modulus-budget row, cofinal tail-window row, RegSeqRat handoff row, RealUp seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper scope is the finite shared-tail budget packet of Definition 698.1 together with its downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ completion-facing consumers in Definition 698.4, Definition 698.7, Definition 698.5, Definition 698.8, and Definition 698.9.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailBudget}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean4 ✓ `BEDC.Derived.CofinalTailBudgetUp.taste gate`

Lean target: `BEDC.Derived.CofinalTailBudgetUp.CofinalTailBudgetTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Lean implementation, bridge checking, Real completeness, infinite tail selection, quotient-level real equality, or a public RealUp object surface.

Upgrade path: No further verification-axis strengthening is recorded because the chapter already sits at the top bridge-checked formal grade.

Chapter 699

A Concrete Naming Certificate for $\mathsf{RealTailAgreementSeal}^{\uparrow}$

The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ packet binds the paired tail-agreement route used by Real and Completion diagonal-limit consumers. It receives two regular rational source rows, the finite StreamName tail windows selected for both sides, the DyadicRatCore readback rows that compare those windows, the displayed agreement row, and the RealUp handoff row as one local NameCert surface. The packet therefore certifies a finite paired tail-window agreement route, not a quotient equality of streams, selected real values, host metric convergence, open-cover compactness, or choice-selected tails.

The agreement row is consumed as the paired-tail counterpart of the finite cofinal-tail route in chapter 658, with the shared-threshold discipline supplied by chapter 560.

Definition 699.1 (Real tail-agreement seal carrier)**.** *A* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, S, W, D, A, H, C, P, N)$$

with the following displayed rows:

1. *R and S are the paired* $\mathsf{RegSeqRat}$ *source rows whose tails are being compared;*
2. *W is the finite paired* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, naming the left and right windows read at the shared tail threshold;*
3. *D is the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row obtained from the two windows in W;*
4. *A is the displayed tail-agreement row, recording that the two dyadic readbacks lie inside the requested finite tolerance on the named windows;*
5. *H records componentwise* hsame *transport for R, S, W, D, A, C, P, N and for the local packet shape;*
6. *C records the* Cont *routes by which a consumer reads the paired sources, finite windows, dyadic readbacks, agreement row, package provenance, and local naming row;*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *finite tolerance rows, and non-escape ledgers;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ *row.*

The carrier exports only the finite packet $R, S, W, D, A, H, C, P, N$ and the RealUp-facing handoff read through it.

Lean4 (definition) `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier`

Theorem 699.2 (Real tail-agreement seal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ *surface for Definition 699.1 consists of five finite obligation rows:*

1. *carrier admission for the packet* $Q = (R, S, W, D, A, H, C, P, N)$ *over paired* $\mathsf{RegSeqRat}$ *sources, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic readback, agreement, transport, continuation, provenance, and local naming rows;*
2. *tail-window coverage stating that every public agreement read uses the paired windows listed in* W *at the shared finite threshold;*
3. *dyadic readback exactness stating that the agreement row* A *is computed only from the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks recorded in* D*;*
4. *RealUp handoff stating that a downstream Real or Completion consumer reaches the seal only through* $R, S, W, D, A, H, C, P, N$*;*
5. *non-escape stating that the packet exports no quotient stream equality, selected real value, host metric convergence, open-cover compactness, choice-selected tail, or hidden equality oracle.*

Proof. Carrier admission is projection from the displayed packet. The two regular rational sources, the finite paired windows, the dyadic readbacks, the agreement row, and the structural rows are all coordinates of Q.

Tail-window coverage is read from W through the continuation row C. A public agreement read must first select one of the paired finite windows already listed in the packet, so it cannot call an unrecorded tail search.

Dyadic readback exactness is the role of D and A. The agreement row is downstream of the paired dyadic readbacks, and the componentwise hsame transport in H preserves the same source-window-readback pairing.

The RealUp handoff and non-escape rows are then package clauses. The provenance P and local naming row N expose only $R, S, W, D, A, H, C, P, N$, so any downstream Real or Completion consumer receives the finite tail-agreement route and no external analytic object. □

Theorem 699.3 (Real tail-agreement seal left–right threshold synchrony)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the two* $\mathsf{RegSeqRat}$ *source rows* R *and* S *read the same finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window threshold through the displayed row* W*. Any public read of the left tail and any public read of the right tail reaches the paired threshold through* W *before* D *or* A *is available, and the route is the shared-threshold surface supplied by chapter 560.*

Proof. Project the carrier of Definition 699.1. The paired source rows R, S and the paired finite tail-window row W are coordinates of one BHist packet, so a public left or right tail read has no carrier coordinate outside the displayed window row.

Apply the carrier-admission and tail-window coverage clauses of Definition 699.2. Carrier admission places R, S, W in the same local $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ surface, while tail-window coverage says that every public agreement read uses the paired windows listed in W at the shared finite threshold.

Finally use the shared-threshold discipline of chapter 560. The present seal consumes that finite threshold surface before its dyadic readback and agreement rows, so the two $\mathsf{RegSeqRat}$ sources synchronize through W rather than through a hidden tail selector or quotient stream equality. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSeal left right threshold synchrony`

Theorem 699.4 (Real tail-agreement seal dyadic-window exactness)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the rows* D *and* A *are computed only from the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks displayed by* D *after the left and right finite windows have been read through* W*. No quotient stream equality, host real equality, selected limit, ambient convergence predicate, or unrecorded dyadic sample can determine the agreement row.*

Proof. Start from the carrier in Definition 699.1. The only dyadic observation coordinate in the packet is D, and the only agreement coordinate downstream of it is A.

Apply the dyadic readback exactness row of Definition 699.2. That row makes A depend on the paired $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks recorded in D, while the tail-window coverage row fixes those readbacks to the windows already named by W.

The sibling finite cofinal-tail route in chapter 658 supplies the comparison pattern: a finite tail read is reused as displayed finite evidence, not as an ambient completed object. Hence D and A are exact readback rows of the local packet and cannot be replaced by quotient equality, host equality, a selected limit, a convergence predicate, or an unrecorded sample. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier dyadic window exactness`

Theorem 699.5 (Real tail-agreement seal Real–Completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reaches the seal through the displayed route*

$$R, S, W, D, A, H, C, P, N.$$

The handoff imports no quotient stream equality, selected limit, host metric convergence, open-cover compactness, or choice-selected tail; it only reuses the paired threshold and finite cofinal-tail surfaces provided by chapter 560 and chapter 658.

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier real completion handoff`

Proof. Project the packet in Definition 699.1. Its Real-facing boundary is exactly the source, window, readback, agreement, transport, continuation, provenance, and local naming rows $R, S, W, D, A, H, C, P, N$.

Apply the RealUp handoff and non-escape clauses of Definition 699.2. The handoff clause forces every downstream Real or Completion consumer through the displayed rows, and the non-escape clause rules out quotient stream equality, selected real values, host convergence, open-cover compactness, choice-selected tails, and hidden equality oracles.

It remains to identify the sibling surfaces consumed by the route. The regular Cauchy tail-meet chapter supplies the shared threshold used by W, and the regular Cauchy tail-fusion chapter supplies the finite cofinal-tail pattern used when the agreement route is exposed to downstream consumers. Thus the handoff is finite and packet-local all the way through $R, S, W, D, A, H, C, P, N$. □

Theorem 699.6 (Real tail-agreement seal TasteGate obligation route)**.** *The paired source, window, readback, and agreement rows used by Definition 699.2 are the same single-carrier rows certified by the checked TasteGate marker recorded for this chapter. In particular, the TasteGate read does not introduce a second carrier, a second window ledger, or a second agreement classifier beyond*

$$R, S, W, D, A, H, C, P, N.$$

Proof. Project the carrier of Definition 699.1. The rows consumed by the NameCert obligations are exactly the paired source rows R, S, the finite paired window row W, the dyadic readback row D, and the agreement row A with the structural rows H, C, P, N.

Read the checked TasteGate marker already attached to the chapter as a single-carrier alignment for that same packet. Since the marker has only the displayed carrier surface available, its field-faithful read cannot select a second packet or a hidden agreement row.

Apply the NameCert obligation theorem. Its carrier-admission, tail-window coverage, dyadic exactness, RealUp handoff, and non-escape rows are therefore the concrete paper rows that the TasteGate marker preserves, not a parallel list of parameters. □

Theorem 699.7 (Real tail-agreement seal paired-window transport)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, componentwise* hsame *transport preserves the paired* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*, the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks in* D*, and the displayed agreement row* A*. A transported consumer still reads the route*

$$W \to D \to A$$

through the same paired sources R, S *and the same structural rows* H, C, P, N*.*

Proof. Project H from the carrier. By definition, this row records componentwise transport for the paired sources, paired windows, dyadic readbacks, agreement row, continuation row, provenance row, local name, and packet shape.

Use tail-window coverage and dyadic readback exactness from Definition 699.2. They force any transported agreement read to encounter the displayed window row before the dyadic readback row and the agreement row.

The continuation, provenance, and local naming rows only replay and name that route. Hence componentwise transport preserves the paired-window, readback, and agreement surface rather than replacing it by a host equality or an unrecorded tail selector. □

Theorem 699.8 (Real tail-agreement seal RealUp consumer threshold factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet first reads the shared-threshold route through chapter 560 and the cofinal-tail route through chapter 658 before reaching the RealUp handoff row of Definition 699.2. The seal exports only the finite route*

$$R, S, W, D, A, H, C, P, N.$$

Proof. Begin with the consumer read. The shared threshold is not an ambient choice of tail; it is the finite threshold surface named by the regular Cauchy tail-meet chapter and then displayed locally as the paired window row W.

Next use the cofinal-tail fusion chapter. It supplies the sibling finite tail route that turns a named cofinal window into a reusable tail-read surface, and the present packet applies that pattern to the two source rows R, S.

Finally apply the RealUp handoff and non-escape clauses of Definition 699.2. The consumer can reach the seal only after the shared threshold, paired windows, dyadic readbacks, and agreement row have been displayed, and it receives no quotient stream equality, selected limit, host convergence, or choice-selected tail. □

Theorem 699.9 (Real tail-agreement seal consumer-window synchronization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet first synchronizes the left and*

right finite $\mathsf{StreamName}^{\uparrow}$ *windows through the finite-window lattice meet, then reads the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared threshold, then uses the* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *cofinal-tail route before the local route*

$$W \to D \to A$$

is exposed. The synchronized packet remains confined to the displayed rows $R, S, W, D, A, H, C, P, N$.

Proof. Start with the RealUp consumer threshold factorization. It puts every Real or Completion consumer behind the shared-threshold route before the local seal handoff can be read.

Apply the window-meet coverage theorem. The left and right finite StreamName windows are first repacked through one finite-window lattice meet, so the paired window row W is not a pair of hidden stream selectors.

Use the budget source lock and paired-window transport. The budget row is read through the displayed budget source, and the componentwise transports preserve the same paired windows, dyadic readbacks, and agreement row.

Finally apply the paired-window source lock. Once the shared threshold and cofinal-tail route have been read through the tail-meet and tail-fusion siblings, the local consumer receives only $W \to D \to A$ inside the finite carrier rows. □

Theorem 699.10 (Real tail-agreement seal budget source lock)**.** *Every paired tail-seal read of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reads its budget source only through the displayed* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *route of chapter 906 and the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *route of chapter 903. The local rows* $R, S, W, D, A, H, C, P, N$ *may replay that budgeted service, but they do not contain an independent tail-budget source, uncapped observation request, selected tail, or ambient convergence coordinate.*

Proof. Begin with the threshold factorization of Definition 699.8. It places every Real or Completion consumer on the shared finite threshold before the local paired windows are read.

Next use the budget-window coherence row Definition 699.53. The finite request source for the paired tail windows is the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ budget row, and the observation service behind that row is the limiter route named by chapter 903.

Finally apply paired-window transport from Definition 699.7 and non-escape from Definition 699.40. Transport may replay the displayed budgeted route through H, C, P, N, but the packet has no coordinate for a second budget source, uncapped request, choice-selected tail, or host convergence proof. □

Theorem 699.11 (Real tail-agreement seal window-meet coverage)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet, the left and right tail windows named by* W *are covered by one finite* $\mathsf{StreamName}^{\uparrow}$ *lattice meet surface before any* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D *or agreement row* A *is consumed. The meet surface is the finite packet of Definition 126.257; the local seal then reads the same paired-window evidence through* $W \to D \to A$.

Proof. Use paired finite tail-window coverage from Definition 699.37. It identifies the left and right source windows through the single local row W before the dyadic readback and agreement rows are available.

Apply the StreamName lattice-meet exhaustion theorem Definition 126.257. The two accepted finite StreamName windows are first repacked as one meet surface over the common classifier, probe bundle, membership rows, unary histories, dyadic ledger, transports, routes, provenance, and local names.

Now return to the local packet using Definition 699.48. Once the meet has supplied the shared finite window surface, every public agreement read proceeds through $W \to D \to A$ and cannot

replace the meet by quotient stream equality, host real equality, selected limits, or a hidden window selector. □

Theorem 699.12 (Real tail-agreement seal window-route determinacy companion)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, paired finite windows determine one displayed dyadic agreement route. If two public reads use the same paired window row* W *and the same dyadic readback row* D*, then the agreement row visible to* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers is the same row* A*.*

Proof. Use the finite-window meet coverage theorem to place both reads on the single $\mathsf{StreamName}^{\uparrow}$ meet surface before any dyadic readback is consumed.

Then apply the paired-window source lock. Agreement on W and D fixes the route $W \to D \to A$, so two reads over the same displayed windows and readback row cannot expose different agreement rows. □

Theorem 699.13 (Real tail-agreement seal window-route exact companion)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the agreement row is read exactly through*

$$W \to D \to A.$$

No consumer may read A *before the paired finite window row* W *and the dyadic readback row* D *have both been displayed.*

Proof. Project the carrier rows W, D, A, H, C, P, N. The window row displays the paired finite tail windows, and the dyadic row is the only readback coordinate attached to those windows.

Apply dyadic exactness and the window-meet coverage theorem. Together they force the public agreement read to pass from the finite windows to the dyadic readback and only then to the agreement row. □

Theorem 699.14 (Real tail-agreement seal window-route non-escape companion)**.** *An accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet cannot replace the displayed route*

$$W \to D \to A$$

by quotient stream equality, a selected limit, host convergence, or a choice-selected tail. Every valid Real or Completion consumer stays inside the finite rows $R, S, W, D, A, H, C, P, N$*.*

Proof. Start from the exact companion route. A public agreement read has no access to A except through the displayed window and dyadic rows.

Now apply the non-escape and Real–Completion boundary theorems. The packet exports only the finite BEDC rows, so quotient streams, selected limits, host convergence, and choice-selected tails have no coordinate that can stand in for $W \to D \to A$. □

Theorem 699.15 (Real tail-agreement seal budget-source window service)**.** *Every paired tail-window service exposed by an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reads the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row and the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row before the paired finite-window row* W *is public. The local packet may then expose*

$$R, S, W, D, A, H, C, P, N,$$

but it cannot serve W *from an uncapped observation request, an unregistered tail source, a selected limit, or a host convergence witness.*

This budget route is the sibling handoff used by chapter 611: the tail-agreement seal consumes the finite observation-budget window route and the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before the* $\mathsf{Real}^{\uparrow}$ *seal is public.*

Proof. Project the accepted carrier $Q = (R, S, W, D, A, H, C, P, N)$ and use the local $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ obligations. They make the source rows, paired window row, dyadic readback row, agreement row, transport rows, continuation rows, package row, and local name the complete service surface.

Apply budget-window coherence. The finite tail request behind the paired service is the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ row, and the observation row serving that request is capped by the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ route before the local W row is available.

Finally use window-meet coverage and the non-escape theorems. The public service reads the finite StreamName meet and then the local route $W \to D \to A$, while quotient stream equality, selected limits, uncapped observations, and host convergence have no coordinate in the packet. □

Theorem 699.16 (Real tail-agreement seal shared-threshold budget exactness)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet, the shared* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold together with the* RegularCauchyTailBud *budget source determines the finite window route before any dyadic readback row* D *or agreement row* A *is public. Thus the only public agreement route is the budgeted shared-threshold route*

$$\mathsf{RegularCauchyTailBudget}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailMeet}^{\uparrow} \longmapsto W \longmapsto D \longmapsto A.$$

Proof. Begin with the carrier and NameCert rows. The local seal has exactly one paired window row W, one dyadic readback row D, and one agreement row A, transported and served only through the displayed histories and continuations.

Use budget-window coherence to place the consumer on the budget source. Then apply the tail-meet factorization: the shared threshold is read before any completion-facing sibling route can consume the local agreement surface.

The window-meet coverage theorem identifies the finite StreamName route feeding W, and the window-route exact companion forces the subsequent read through D and then A. Non-escape excludes quotient streams, selected tails, host limits, and hidden completion selectors, so the shared threshold and budget source jointly determine the finite route before D and A are exposed. □

Theorem 699.17 (Real tail-agreement seal budgeted TailMeet window exhaustion)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reaches the agreement row* A *passes, in order, through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, the displayed paired window row* W*, the dyadic readback row* D*, and only then* A*. The route exports no quotient stream equality, host real equality, selected tail, ambient convergence predicate, or second agreement row.*

Proof. Start with the budget-source window service. It places any public paired-window request behind the observation limiter and the regular Cauchy tail budget before the row W can be read.

Next apply shared-threshold budget exactness. The shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold is the only threshold row feeding the displayed finite window route, so the consumer cannot bypass the budget through an unregistered tail source.

Use window-meet coverage to identify the paired window row and the dyadic readback row. The displayed service therefore proceeds through W and then D before the local agreement row A is available.

Finally apply the Real-completion budgeted route and the non-escape clauses. They confine every $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumer to the same displayed packet rows and exclude quotient

stream equality, host real equality, selected tails, ambient convergence predicates, and any second agreement coordinate. □

Theorem 699.18 (Real tail-agreement seal Real-completion budgeted route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet factors through the budget source, shared threshold, finite windows, dyadic readback, and agreement row before the seal is public. Concretely, the consumer route is confined to*

$$\mathsf{RegularCauchyTailBudget}^{\uparrow},\ \mathsf{ObservationBudgetLimiter}^{\uparrow},\ \mathsf{RegularCauchyTailMeet}^{\uparrow},\ W,\ D,\ A,\ R, S, H, C, P, N,$$

and cannot import quotient stream equality, host real equality, selected limits, ambient completeness, or a second agreement row.

Proof. Start with the RealUp consumer threshold factorization and the Completion consumer ledger. Together they place both Real and Completion consumers on the displayed carrier rows before any public seal read.

Next use the budget-source window service and shared-threshold budget exactness. The consumer first receives the budgeted observation route, then the shared TailMeet threshold, then the finite StreamName windows, dyadic readback row, and agreement row.

It remains to close the public boundary. Apply window-meet coverage, the completion route non-escape theorem, and the Real–Completion boundary theorem. These results keep the route inside the displayed BHist, Cont, Pkg, and NameCert rows, so no external quotient, host limit, selected tail, ambient completeness proof, or second agreement coordinate can be read before the seal is public. □

Theorem 699.19 (Real tail-agreement seal budget-threshold readback exhaustion)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer route that exposes a public agreement row factors through* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *the finite StreamName window* W, *the dyadic readback row* D, *the agreement row* A, *and only then the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal rows. No quotient-stream equality, selected limit, uncapped observation, host convergence witness, or second agreement coordinate is available along this public route.*

Proof. Apply the budget-source window service to place any public paired-window request behind the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ row and the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ cap before W is served.

Next apply shared-threshold budget exactness. The only threshold available to the public agreement route is the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row, and it feeds the finite StreamName window before the dyadic readback row D and agreement row A can be read.

Finally use the Real-completion budgeted route. Carrier projection identifies the remaining public rows as the displayed Real and Completion seal rows, and non-escape excludes quotient-stream equality, selected limits, uncapped observations, host convergence witnesses, and any second agreement coordinate. □

Theorem 699.20 (Real tail-agreement seal terminal consumer route exhaustion)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that exposes the agreement row of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet is exhausted by the single displayed route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow},\ \mathsf{RegularCauchyTailBudget}^{\uparrow},\ \mathsf{RegularCauchyTailMeet}^{\uparrow},\ W,\ D,\ A,\ R, S, H, C, P, N.$$

In particular, the terminal consumer must traverse the observation cap, the regular Cauchy tail budget, the shared TailMeet threshold, the finite StreamName window W, *the dyadic readback row* D, *the agreement row* A, *and the displayed* R, S, H, C, P, N *rows. It has no quotient stream equality, selected limit, uncapped observation, host convergence witness, or second agreement coordinate.*

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealTerminalRoute exhaustion`

Proof. First apply the budget-source window service. It places the finite StreamName window behind the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ cap and the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ source before any agreement-facing row can be read.

Next apply shared-threshold budget exactness and the budgeted TailMeet window exhaustion theorem. These force the route through $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, then through the displayed window W, dyadic readback row D, and agreement row A.

Apply the Real-completion budgeted route and the budget-threshold readback exhaustion theorem to the terminal consumer. They identify the remaining public rows as R, S, H, C, P, N and show that both $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers use the same carrier route.

Carrier projection now leaves exactly the displayed packet coordinates, and non-escape removes every route not represented by those coordinates. Hence quotient stream equality, selected limits, uncapped observation requests, host convergence witnesses, and any second agreement coordinate are outside the terminal consumer surface. □

Theorem 699.21 (Real tail-agreement seal budget route uniqueness)**.** *Every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *terminal consumer exposing the public agreement row has the unique public route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to W \to D \to A \to R, S, H, C,$$

Any displayed consumer route that omits or reorders the observation cap, the tail budget, the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the dyadic readback row* D*, or the agreement row* A *is outside the carrier. Thus the agreement seal has no alternate public route through quotient streams, selected limits, uncapped observations, host convergence, or a second agreement row.*

Proof. The budget-source window service fixes the first public read. A consumer exposing the agreement row must first pass through the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ cap and the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ source before the finite window W is available.

Shared-threshold budget exactness then fixes the next boundary. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold is the only public threshold feeding the displayed finite window, and budgeted TailMeet exhaustion forces the route through W, then D, then A.

The Real-completion budgeted route identifies the remaining public rows as R, S, H, C, P, N. Terminal consumer route exhaustion shows that both $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers read the same displayed carrier route after A.

Therefore any route that omits or reorders one of the named rows cannot be a route through the accepted carrier. The non-escape clauses exclude quotient stream equality, selected limits, uncapped observations, host convergence witnesses, and any second agreement row. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSeal budget route uniqueness`

Theorem 699.22 (Real tail-agreement seal root budget-source totality)**.** *Every root-facing* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer that exposes the public agreement row must first read* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudge}$ $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the dyadic readback row* D*, and the agreement row* A*, before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal row is public. Thus the root budget-source route is total for agreement-facing consumers and has no quotient stream equality, selected limit, uncapped observation, host convergence witness, or second agreement coordinate.*

Proof. Start with the budget-source window service. It places every root-facing window read behind the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ cap and the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ source before the finite window W is available.

Next apply shared-threshold budget exactness and budgeted TailMeet window exhaustion. These force the public route through $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, then through W, the dyadic readback row D, and the agreement row A.

Finally use terminal consumer route exhaustion. It identifies the only later public rows as the displayed $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ seal rows carried by R, S, H, C, P, N, so the earlier budget-source route is total for any consumer exposing A.

Coordinate exhaustion gives the exclusions. Quotient streams, selected limits, uncapped observations, host convergence witnesses, and second agreement coordinates have no carrier row in the displayed root route. □

Theorem 699.23 (Real tail-agreement seal terminal budget-source lock)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer exposing the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *agreement row must first lock onto the same* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, CauchySealBudgetSynchroniz *finite* $\mathsf{StreamName}^{\uparrow}$ *window* W, $\mathsf{DyadicRatCore}^{\uparrow}$ *readback* D, *and agreement row* A. *Any terminal route that does not preserve this source lock is outside the carrier and cannot supply quotient-stream equality, selected limits, host convergence, ambient completeness, or a second agreement coordinate.*

Proof. Start from the budget-source window service and root budget-source totality. They force every terminal agreement consumer to read the observation cap, tail budget, shared TailMeet threshold, finite window W, dyadic readback D, and agreement row A before the Real or Completion seal rows are public.

Use shared-threshold budget exactness together with budgeted TailMeet window exhaustion. The shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold is the unique source boundary for the finite window and cannot be replaced by an uncapped observation or a selected-limit read.

Apply the budget synchronizer factorization theorem Definition 699.25. It inserts the CauchySealBudgetSynchron row between the shared threshold and the finite window, so the lock is the same route for both $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers.

Finally use terminal consumer route exhaustion and downstream route no-branching. Any route that drops one of the locked rows has no displayed carrier coordinate, and the excluded quotient-stream, selected-limit, host-convergence, ambient-completeness, and second-agreement readings therefore cannot supply the terminal seal. □

Theorem 699.24 (Real tail-agreement seal budget route canonical form)**.** *Every accepted terminal agreement route of a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet has the canonical displayed order*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to W \to D \to A \to R, S, H, C,$$

Any public read of the agreement row A *is equivalent, as a* BHist *route through the packet, to traversing that order before* A *is exposed.*

Proof. First use the budget-source window service and shared-threshold budget exactness. They place every public window request behind the observation cap, the regular Cauchy tail budget, and the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold before W is available.

Next apply budgeted TailMeet window exhaustion. It forces the route from the shared threshold through the finite StreamName window W, then through the dyadic readback row D, and only then to the agreement row A.

The Real-completion budgeted route and terminal consumer route exhaustion identify the remaining public rows as R, S, H, C, P, N. Budget route uniqueness then rules out any accepted terminal route that omits or reorders one of the displayed rows.

Therefore a public read of A is exactly a traversal of the canonical route shown in the statement. □

Theorem 699.25 (Real tail-agreement seal budget synchronizer factorization)**.** *Every accepted terminal agreement route of a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet factors through the budget synchronizer before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal consumer receives the classifier. The route has the ordered public surface*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{CauchySealBudgetSynchroni}$$

Here W is the finite $\mathsf{StreamName}^{\uparrow}$ *window, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback obtained from the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and A is the unique agreement row exported to the Real and Completion consumers. No quotient stream equality, selected limit, uncapped observation request, host convergence witness, or second agreement row is available before those consumers read the classifier.*

Proof. Start with the budget-source window service and shared-threshold exactness. They place any terminal agreement route behind $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, and the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold before the finite window row can be public.

Next apply budgeted TailMeet exhaustion and the synchronizer row. The shared threshold is routed through $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ before the $\mathsf{StreamName}^{\uparrow}$ window W is read, and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff supplies the finite regular-sequence data used by the $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D.

Use the canonical-form theorem Definition 699.24. It identifies the terminal agreement route with the displayed order through W, D, and the agreement row A before the remaining seal rows R, S, H, C, P, N are public.

Finally apply terminal consumer route exhaustion and downstream route no-branching. The exported classifier therefore reaches $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ only through the displayed synchronizer route, and the non-escape clauses exclude quotient streams, selected limits, uncapped observations, host convergence, and second agreement rows. □

Theorem 699.26 (Real tail-agreement seal budget synchronizer route determinacy)**.** *Let two terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumers of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet preserve the same observation cap, tail budget, shared* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row, finite* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback D, and agreement row A. Then the two consumers expose the same displayed public route through R, S, H, C, P, N. No quotient stream, selected limit, uncapped observation, host convergence witness, ambient completeness datum, or second agreement row can distinguish the terminal routes.*

Proof. Start with the terminal budget-source lock of Definition 699.23. It fixes the observation cap, tail budget, shared TailMeet threshold, finite window, readback row, and agreement row before either terminal seal consumer can read the classifier.

Apply the canonical-form theorem Definition 699.24. Once those source rows are fixed, each public agreement read traverses the same ordered finite route before the displayed R, S, H, C, P, N rows are exposed.

Next use the synchronizer factorization of Definition 699.25. The shared threshold passes through the same $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ row before the finite window W, the dyadic readback D, and the agreement row A are consumed.

Finally apply downstream route no-branching, Definition 699.27. Since no carrier coordinate supplies quotient streams, selected limits, uncapped observations, host convergence, ambient completeness, or a second agreement row, the two terminal consumers cannot be distinguished after preserving the same synchronizer route. □

Theorem 699.27 (Real tail-agreement seal downstream route no-branching)**.** *No downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet can branch around the budget route through quotient stream equality, selected limits, uncapped observations, host convergence, or a second agreement coordinate. Every such consumer remains on the canonical route through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, W, D, A, *and the displayed* R, S, H, C, P, N *rows.*

Proof. Begin with root budget-source totality. It places any root-facing agreement read behind the observation cap, tail budget, shared TailMeet threshold, finite window, dyadic readback, and agreement row.

Use the canonical-form theorem Definition 699.24. It identifies any public read of A with the single displayed order through the finite carrier rows.

Finally apply terminal consumer route exhaustion and budget route uniqueness. These theorems leave no coordinate for quotient stream equality, selected limits, uncapped observations, host convergence witnesses, or a second agreement row, so no downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer can branch around the budget route. □

Theorem 699.28 (Real tail-agreement seal regular TailMeet budget lock)**.** *Every terminal* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *agreement route accepted by* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *is locked to the ordered route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{CauchySealBudgetSynchroni}$$

Here W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* A *is the unique agreement row, and* R, S, H, C, P, N *are the local seal rows displayed by the accepted packet. Once this route is fixed, no quotient stream, selected limit, host equality, ambient completeness principle, or second agreement row can distinguish any downstream seal consumer.*

Proof. First build the upstream budget source. The budget-source window service and shared-threshold exactness place any terminal agreement route behind $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudge}$ and the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold before a finite agreement window can be read.

Next attach the synchronizer. Apply the budget synchronizer factorization of Definition 699.25: the shared threshold passes through $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ before the finite $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ readback D, and the agreement row A are consumed.

The local agreement surface is then fixed by downstream route no-branching, Definition 699.27. It keeps each $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumer on the same route through R, S, H, C, P, N after the synchronizer route has been selected.

Finally use the CauchyLimitSeal finite-window lattice route, Definition 386.160. The finite window is a lattice meet rather than a hidden selected limit, so the completed route has no coordinate for quotient stream equality, host equality, ambient completeness, or a second agreement row to separate the consumers. □

Theorem 699.29 (Real tail-agreement seal budget route factorization)**.** *Every accepted terminal agreement read of a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet factors through the canonical budget route, the synchronizer route, and the regular-tailmeet budget lock before it exports the terminal agreement row to a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer. Thus the exported terminal agreement has the single displayed*

route through ObservationBudgetLimiter$^{\uparrow}$, RegularCauchyTailBudget$^{\uparrow}$, RegularCauchyTailMeet$^{\uparrow}$, CauchySealBudgetSy *the finite* StreamName$^{\uparrow}$ *window* W*, the* DyadicRatCore$^{\uparrow}$ *readback row* D*, the agreement row* A*, and the local seal rows* R, S, H, C, P, N*.*

Proof. Apply budget route uniqueness from Definition 699.21. Any accepted terminal agreement read must use the displayed observation cap, regular Cauchy tail budget, shared TailMeet threshold, finite window, dyadic readback, agreement row, and local seal rows.

Next use the canonical-form theorem Definition 699.24. It identifies the public read of A with that ordered route before the terminal seal rows can be exported.

Insert the synchronizer by Definition 699.26. The terminal route preserves the same CauchySealBudgetSynchroni row before the finite window, readback row, and agreement row are consumed.

Finally apply downstream route no-branching from Definition 699.27 and the regular TailMeet budget lock of Definition 699.28. The terminal agreement row is therefore exported only after the displayed budget route, synchronizer route, and regular-tailmeet lock have all been traversed, with no quotient stream, selected limit, host equality, ambient completeness coordinate, or second agreement row available. □

Theorem 699.30 (Real tail-agreement seal root budget-source canonicality)**.** *Every displayed* RealTailAgreementSeal$^{\uparrow}$ *packet exposing a terminal agreement row determines a unique root budget-source row. The canonical source is the initial* BHist *segment through* ObservationBudgetLimiter$^{\uparrow}$, RegularCauchyTailBudget$^{\uparrow}$, RegularCauchyTailMeet$^{\uparrow}$, CauchySealBudgetSynchronizer$^{\uparrow}$, W, D, *and* A *whose* hsame, Cont, Pkg, *and* NameCert *rows are consumed by the terminal route*

ObservationBudgetLimiter$^{\uparrow}$ → RegularCauchyTailBudget$^{\uparrow}$ → RegularCauchyTailMeet$^{\uparrow}$ → CauchySealBudgetSynchroni

No second source row, quotient stream equality, selected limit, uncapped observation request, host convergence witness, or ambient completeness datum can serve as a competing root budget source. The intended Lean home for this package is the namespace `BEDC.Derived.RealTailAgreementSealUp.RootBudge`

Proof. Use root budget-source totality, Definition 699.22. It forces any root-facing agreement consumer to read the observation cap, the regular tail budget, the shared TailMeet threshold, the finite window, the dyadic readback row, and the agreement row before the terminal seal rows are public.

Use budget route uniqueness, Definition 699.21. Once the displayed packet exposes the public agreement row, no accepted BHist route may omit or reorder the budget source, shared threshold, finite window, readback row, or agreement row.

Finally apply the regular TailMeet budget lock of Definition 699.28. The synchronizer and the local rows R, S, H, C, P, N are consumed on the same route, so the hsame, Cont, Pkg, and NameCert rows name one canonical root budget-source row and exclude every competing source coordinate. □

Theorem 699.31 (Real tail-agreement seal root budget threshold coverage)**.** *For every accepted* RealTailAgreementSeal$^{\uparrow}$ *terminal agreement route, the shared-threshold budget row covers every* RegularCauchyTailMeet$^{\uparrow}$ *window admitted by the regular TailMeet budget lock. Concretely, any TailMeet window consumed by* Real$^{\uparrow}$ *or* Completion$^{\uparrow}$ *is first covered by the* RegularCauchyTailBudget$^{\uparrow}$ *source and the shared* RegularCauchyTailMeet$^{\uparrow}$ *threshold, then read through* W, D, A, *and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows of the packet. No selected tail, uncapped observation, host convergence witness, quotient stream equality, or second agreement row lies outside this threshold coverage.*

Proof. Begin with shared-threshold budget exactness, Definition 699.16. It identifies the shared $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ threshold and the regular tail budget as the unique public source for the finite route before D or A is visible.

Next use budgeted TailMeet window exhaustion and the regular TailMeet budget lock. They show that each terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer reaches the agreement row only through the shared threshold, the synchronizer, the finite window W, and the dyadic readback row D.

The terminal route exhaustion theorem then consumes the local rows R, S, H, C, P, N on the same displayed packet. Thus every admitted TailMeet window is already covered by the shared-threshold budget row, and the excluded selected-tail, uncapped-observation, host-convergence, quotient, and second-agreement readings have no carrier coordinate. □

Theorem 699.32 (Real tail-agreement seal root budget terminal consumption)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of a displayed* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet may consume only the canonical root budget route produced by root budget-source canonicality and root budget threshold coverage. The permitted consumption route is the finite* BHist *route through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, W, D, A, *and* R, S, H, C, P, N, *together with the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Any terminal read requiring a second source row, a selected limit, quotient stream equality, uncapped observations, host convergence, ambient completeness, or a second agreement coordinate is outside the package.*

Proof. First take the canonical source supplied by Definition 699.30. It combines totality with budget route uniqueness, so the terminal consumer has only one root budget-source row to read.

Next use Definition 699.31. The shared threshold covers the TailMeet window consumed by the terminal route, and the window is read through W, D, and A before any local seal row is exposed.

Finally apply terminal consumer route exhaustion and downstream route no-branching. They consume R, S, H, C, P, N through the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows and rule out every terminal route that asks for a second source, selected limit, quotient stream equality, uncapped observation, host convergence, ambient completeness, or a second agreement coordinate. □

Theorem 699.33 (Real tail-agreement seal root budget-route totality)**.** *Every accepted paired tail-agreement read of a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$ *factors through the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *route and the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *source before the local route*

$$W \to D \to A$$

is public. The route is total for the finite root read and supplies no uncapped observation request, quotient stream equality, selected real value, host convergence row, or second tail-budget source.

Proof. Project the carrier rows of Q and use the local $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ obligations. The only public service rows available before agreement are the displayed source rows R, S, the finite window row W, the dyadic readback row D, and the agreement row A, with transport, continuation, provenance, and name data carried by H, C, P, N.

Apply Definition 699.10 and Definition 699.63. They place the paired tail request on the finite $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ source and cap its observation route through $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ before the shared tail window can be read.

Finally use the window-route exact companion. Once the budgeted route has displayed the finite window, the agreement read proceeds through W, then D, then A. Since the carrier has no

coordinate for an uncapped request, quotient equality, selected value, host convergence, or second budget source, the stated budget-route factorization is exhaustive. □

Theorem 699.34 (Real tail-agreement seal root source-window transport)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *and* $Q' = (R', S', W', D', A', H', C', P', N')$ *be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. Componentwise* hsame *transport of the displayed source rows* R, S*, paired window row* W*, and dyadic readback row* D *preserves the same paired source-window route from the budgeted tail request to the agreement row. Thus the transported packet reads the same finite route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to R, S \to W \to D \to A$$

up to the displayed local transport rows, without introducing quotient stream equality, selected limits, host convergence, or an ambient completion selector.

Proof. Read the componentwise hsame rows from H and H'. These rows identify only the displayed source, window, and readback components; they do not add a new tail source or a separate equality predicate.

Use the root budget-route totality theorem for Q and Q'. Both packets place their paired source-window reads behind the same finite budget and limiter surfaces before exposing any local agreement row.

Now apply the tail-classifier transport and window-route determinacy results. Transport of R, S, W, D carries the finite source-window route across the two packets, and determinacy fixes the agreement row reached after $W \to D$. The transported route therefore remains the same paired source-window route, with no quotient, selected-limit, convergence, or external completion coordinate available to replace it. □

Theorem 699.35 (Real tail-agreement seal root completion-boundary exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet is exhausted by the local rows*

$$R, S, W, D, A, H, C, P, N$$

together with the sibling budget, tail-meet, tail-fusion, $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and Real routes named by chapter 906, chapter 560, chapter 658, chapter 386, and chapter 38. No root consumer obtains quotient stream equality, a selected limit, host convergence, ambient completeness, or a hidden tail selector from the seal.*

Proof. Begin with the root budget-route totality theorem. It forces any paired tail-agreement read to pass through the finite budget and limiter route before the local agreement surface $W \to D \to A$ can be consumed.

Next use the source-window transport theorem and the sibling tail-meet and tail-fusion routes. The transport row keeps R, S, W, D on the same paired source-window route, while the meet and fusion siblings supply the shared tail surface that the local packet consumes.

It remains to account for Real and Completion-facing endpoints. Apply Definition 699.65 together with the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ sibling routes. These consumers receive only the displayed finite rows and the named sibling surfaces, so the boundary is exhausted without quotient stream equality, selected limits, host convergence, ambient completeness, or hidden tail selection. □

Theorem 699.36 (Real tail-agreement seal downstream route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives only the route*

$$R, S, W, D, A, H, C, P, N$$

together with the budget source lock of Definition 699.10 and the finite-window meet coverage of Definition 699.11. No downstream consumer can add a quotient stream, selected host limit, host convergence row, external completion selector, second budget gate, or second paired-window meet.

Proof. First apply the RealUp consumer threshold factorization Definition 699.8. It puts both Real and Completion consumers on the displayed threshold route before the local seal handoff.

Next combine the budget source lock with the window-meet coverage theorem. The former fixes the only budget route through $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ and $\mathsf{ObservationBudgetLimiter}^{\uparrow}$; the latter fixes the only finite StreamName meet surface feeding W, D, A.

Finally use Definition 699.52 and Definition 699.58. These rows exhaust the downstream route by $R, S, W, D, A, H, C, P, N$, so the cited budget and meet surfaces are consumed as finite predecessors of the same displayed packet rather than as hidden quotient, choice, convergence, or completion data. □

Definition 699.37 (Real tail-agreement seal window coverage)**.** *For a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *carrier* $Q = (R, S, W, D, A, H, C, P, N)$*, paired finite tail-window coverage means that every public agreement read first selects the same displayed threshold in the* $\mathsf{StreamName}^{\uparrow}$ *row* W*, then reads the left and right finite windows of the paired source rows* R *and* S*, and only then reaches the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D *and the agreement row* A*. The coverage predicate is transported componentwise by* H*, replayed only through* C*, recorded in provenance* P*, and named locally by* N*.*

Theorem 699.38 (Real tail-agreement seal dyadic exactness)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the agreement row* A *is determined only by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D *selected by the paired tail-window row* W*. No separate quotient-stream equality, selected real equality, host metric limit, or hidden tail oracle can determine* A*.*

Proof. Project the displayed rows W, D, A, H, C, P, N from the carrier. The window row W selects the paired finite tail windows, and the readback row D is the only dyadic observation row attached to those windows.

Apply the tail-window coverage and dyadic readback exactness clauses of Definition 699.2. They force every public agreement read through W and D before A is available.

The structural rows H, C, P, N preserve and expose that same dependency. Since none of them contains quotient stream equality, selected real equality, a host limit, or a tail oracle, the agreement row is determined exactly by the displayed dyadic readback route. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealWindowRoute result exact`

Theorem 699.39 (Real tail-agreement seal shared-threshold non-escape)**.** *Any downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that uses the shared tail threshold of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet must factor through* W*, then* D*, then* A*. The consumer cannot import quotient stream equality, selected limits, host convergence, or choice-selected tails into the seal route.*

Proof. Begin with a downstream Real or Completion read. By Definition 699.37, the read first reaches the paired finite windows selected by W.

Next apply Definition 699.38. The selected windows determine the dyadic readback row D, and D determines the displayed agreement row A.

Finally use the RealUp handoff and non-escape clauses of Definition 699.2. They expose the agreement only through the finite packet rows and exclude quotient stream equality, selected limits, host convergence, and choice-selected tails from the consumer route. □

Theorem 699.40 (Real tail-agreement seal non-escape)**.** *An accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet cannot be used to export quotient stream equality, selected real values, host metric convergence, open-cover compactness, or choice-selected tails. Any attempted read of one of those objects is outside the local packet unless it factors through the finite rows*

$$R, S, W, D, A, H, C, P, N.$$

Proof. Project the carrier. Its source rows are the paired $\mathsf{RegSeqRat}$ inputs, its observation rows are the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks, and its comparison row is the displayed finite agreement row.

The structural rows only transport, route, package, and name those coordinates. They do not add a quotient of streams, an ambient real carrier, a host metric space, an open cover, or a choice function selecting tails.

Therefore a read that asks for any of those external objects has no coordinate in the packet. The only Real-facing information available to downstream consumers is the finite agreement route through $R, S, W, D, A, H, C, P, N$. □

Theorem 699.41 (Real tail-agreement seal tail classifier)**.** *For a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *carrier* $Q = (R, S, W, D, A, H, C, P, N)$*, the displayed agreement row* A *is classifier-readable only through the exposed rows* R, S, W, D*. In particular, a public classifier for the finite tail agreement must first read the paired* $\mathsf{RegSeqRat}$ *sources* R, S*, the paired* $\mathsf{StreamName}^{\uparrow}$ *tail-window row* W*, and the paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D*; the row* A *carries no separate tail oracle, quotient-stream equality, or ambient real equality.*

Proof. Project the carrier in Definition 699.1. The agreement coordinate A is displayed only after the source rows R, S, the window row W, and the readback row D have been named.

Apply the tail-window and dyadic-readback clauses of Definition 699.2. They force every public agreement read to pass through W and D, so the classifier sees the same finite rows that produced A.

Finally apply Definition 699.40. Since the packet exports no quotient equality or hidden tail selector, no classifier can bypass R, S, W, D while still reading the displayed agreement row. □

Theorem 699.42 (Real tail-agreement seal classifier reflexivity)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the displayed tail-agreement classifier reads the same finite route against* Q *itself:*

$$R, S, W, D, A.$$

The reflexive read uses the local paired-window row W*, the local dyadic readback row* D*, and the displayed agreement row* A*; it does not import quotient stream equality, host real equality, a selected limit, or an additional tail selector.*

Proof. Apply Definition 699.41 to the packet Q. The classifier-readable surface is exactly the displayed source, window, readback, and agreement route R, S, W, D, A.

Next use Definition 699.48. Since the packet is compared with itself, the source rows, window row, and structural rows agree on the nose as displayed BEDC rows, so every public agreement read follows the same local route $W \to D \to A$.

Finally apply Definition 699.47 and Definition 699.40. They rule out a second classifier row and any external equality object, leaving the reflexive classifier read as the displayed finite route already carried by Q. □

Theorem 699.43 (Real tail-agreement seal classifier symmetry)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ and $Q' = (R', S', W', D', A', H', C', P', N')$ be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. If a public classifier read displays a paired-window agreement route from Q to Q' through the shared rows R, S, W, D, A and R', S', W', D', A', then the reverse public read displays the corresponding route from Q' to Q through the same finite window and dyadic readback rows. The reversal does not assert quotient stream equality or host real equality; it only reverses the displayed paired-window agreement route.*

Proof. Read the forward classifier by Definition 699.41. Both endpoints expose their agreement rows only after the paired source rows, paired window rows, and dyadic readback rows have been displayed.

Use Definition 699.48 on the common boundary. The same finite window route that allows the forward read also fixes the reverse read once the endpoints are exchanged, because the classifier has no separate selector outside W, D, H, C, P, N.

Apply Definition 699.47 to both packets. It identifies the visible rows at each endpoint and prevents a second source, window, readback, or agreement coordinate from being substituted during reversal.

Finally use the non-escape and handoff rows recorded in Definition 699.40 and Definition 699.45. The reverse route remains a finite BEDC classifier read for Real and Completion consumers, not an assertion of quotient stream equality or host real equality. □

Theorem 699.44 (Real tail-agreement seal classifier transitivity)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$, $Q' = (R', S', W', D', A', H', C', P', N')$, and $Q'' = (R'', S'', W'', D'', A'', H'', C'', P'', N'')$ be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. If public classifier reads display a paired-window agreement route from Q to Q' and a paired-window agreement route from Q' to Q'', and both reads pass through the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows at Q', then the composed public read displays a route from Q to Q'' through the same finite window-readback chain. The composition exports only the displayed R, S, W, D, A rows at the endpoints and does not add quotient stream equality, host real equality, a selected real value, or a hidden completion selector.*

Proof. Start with the two classifier reads and apply Definition 699.41. Each read reaches its agreement row only through paired source rows, a finite $\mathsf{StreamName}^{\uparrow}$ window row, and a $\mathsf{DyadicRatCore}^{\uparrow}$ readback row.

At the middle packet Q', use Definition 699.48. The shared finite window row fixes the same readback route $W' \to D' \to A'$ for both adjacent reads, so the middle packet supplies one displayed hinge rather than two independent classifiers.

Now apply Definition 699.47. Since the middle packet has unique visible source, window, readback, and agreement coordinates inside the local ledger, the two adjacent routes compose through the displayed W' and D' rows.

Finally use Definition 699.40 and Definition 699.45. The composed endpoint route is still the finite classifier handoff read by Real and Completion consumers, so it cannot introduce quotient stream equality, host real equality, selected real values, or hidden completion selectors. □

Theorem 699.45 (Real tail-agreement seal Real handoff)**.** *Every Real or Completion consumer of a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives the displayed agreement through*

$$R, S, W, D, A, H, C, P, N$$

together with the shared-threshold surface supplied by chapter 560. The handoff is therefore the paired-tail analogue of the finite cofinal-tail route in chapter 658, not a direct assertion of Real equality or completion convergence.

Proof. The carrier supplies the local rows $R, S, W, D, A, H, C, P, N$. The RealUp handoff clause in Definition 699.2 says that a downstream consumer reaches the seal only through those rows.

The agreement row is finite because its windows and readbacks are finite. The cofinal-tail route of chapter 658 gives the sibling pattern for exposing finite tail evidence, while chapter 560 supplies the shared-threshold discipline needed for a paired comparison.

The non-escape theorem then fixes the boundary of the handoff. A Real or Completion consumer may consume the displayed paired-tail agreement route, but it does not receive quotient stream equality, a selected Real value, host metric convergence, open-cover compactness, or a choice-selected tail. □

Theorem 699.46 (Real tail-agreement seal ledger exhaustion)**.** *The rows*

$$R, S, W, D, A, H, C, P, N$$

exhaust the local $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *ledger. Any valid local obligation is one of carrier admission, tail-window coverage, dyadic readback exactness, RealUp handoff, or non-escape; no local row remains for quotient stream equality, selected reals, host metric convergence, open-cover compactness, choice-selected tails, or hidden tail oracles.*

Proof. Carrier admission in Definition 699.2 lists the whole packet. The only displayed data rows are the paired sources, windows, readbacks, agreement, transport, continuation, provenance, and local naming rows.

The middle obligations account for the classifier surface. Tail-window coverage accounts for W, dyadic readback exactness accounts for D and A, and the RealUp handoff accounts for the consumer route through H, C, P, N.

It remains to rule out extra ledger rows. Apply Definition 699.40: the packet contains no coordinate for quotient stream equality, selected reals, host metric convergence, open-cover compactness, choice-selected tails, or hidden tail oracles. Hence the five displayed obligations exhaust the local ledger. □

Theorem 699.47 (Real tail-agreement seal source separation obligation)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the rows* R *and* S *are the only paired* $\mathsf{RegSeqRat}$ *source rows,* W *is the only paired* $\mathsf{StreamName}^{\uparrow}$ *tail-window row,* D *is the only paired* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row, and* A *is the only displayed agreement row. The structural rows* H, C, P, N *transport, route, package, and name these fields, but they do not introduce a second source, a second tail window, a second dyadic readback, or a second agreement classifier.*

Proof. Project the carrier in Definition 699.1. Its displayed data coordinates are exactly the paired source rows R, S, the paired window row W, the readback row D, and the agreement row A.

Apply the carrier-admission and non-escape clauses of Definition 699.2. Carrier admission lists the rows admitted to the local $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$ surface, while non-escape excludes hidden tail selectors, quotient equalities, and unrecorded analytic objects.

The remaining structural rows cannot manufacture another source of agreement. The componentwise transport row H preserves the same coordinates, C only routes reads through them, P records their provenance, and N names the local packet. □

Theorem 699.48 (Real tail-agreement seal paired-window source lock)**.** *Let a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer agree with an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet on the rows* R, S, W, H, C, P, N*. Then every public agreement read by that consumer uses the same route*

$$W \to D \to A.$$

In particular, the consumer cannot replace the paired tail-window selector by a second selector, and it cannot replace the dyadic readback route by host equality of completed real values.

Proof. The agreement on R, S, W, H, C, P, N fixes the public carrier boundary through which the consumer reads the seal. By Definition 699.37, the read first reaches the paired finite tail windows selected by W.

Apply Definition 699.38. Once W has selected the paired windows, the only readback row available for the agreement is D, and the displayed agreement row is A.

Finally use Definition 699.47. Since no second window selector, second readback row, or host equality row is present in the local ledger, every Real or Completion consumer that stays on the displayed boundary must follow $W \to D \to A$. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealWindowRoute result deterministic`

Theorem 699.49 (Real tail-agreement seal terminal consumer factorization)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of Q factors its public read through the displayed route*

$$R, S, W, D, A, H, C, P, N.$$

In particular, the consumer first agrees with the paired source and tail-window rows, then reads the dyadic agreement route $W \to D \to A$, and only then uses the structural rows H, C, P, N for transport, continuation, provenance, and local naming.

Proof. Apply Definition 699.8. It identifies the Real-facing threshold route through the carrier rows and prevents the consumer from entering the packet through an unlisted analytic boundary.

Apply Definition 699.48. Once the consumer agrees with R, S, W, H, C, P, N, every public agreement read follows the finite route $W \to D \to A$.

Finally apply Definition 699.47. The only source, window, readback, agreement, transport, continuation, package, and name rows available to a terminal Real or Completion consumer are the displayed coordinates of Q. □

Theorem 699.50 (Real tail-agreement seal dyadic terminal non-escape)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. A terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of Q cannot replace the route*

$$W \to D \to A$$

by quotient stream equality, selected real equality, host metric convergence, open-cover compactness, or choice-selected tails.

Proof. By Definition 699.37, the public read reaches the finite paired tail windows selected by W before it reaches any agreement row.

Apply Definition 699.38. The selected windows are read back through the dyadic row D, and the agreement visible to the consumer is the displayed row A.

Apply Definition 699.39. The shared threshold is not a quotient-stream equality, a selected real, a host metric convergence proof, an open-cover compactness argument, or a choice-selected tail. Hence no terminal consumer can substitute any of those objects for $W \to D \to A$. □

Theorem 699.51 (Real tail-agreement seal Completion consumer exactness)**.** *Every* $\mathsf{Completion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reads the same paired tail-window agreement surface before any completion seal read. The only rows it receives are the displayed*

BEDC rows over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*:*

$$R, S, W, D, A, H, C, P, N.$$

Proof. Use Definition 699.49 to place the Completion-facing consumer on the terminal packet route. Its public read factors through the displayed rows before it can consume a completion seal.

Use Definition 699.50 to fix the agreement surface. The consumer must read $W \to D \to A$, and it cannot replace that dyadic route by quotient streams, selected limits, host convergence, compactness, or choice-selected tails.

It remains to identify the received rows. Apply Definition 699.40 together with the chapter closure scope: the local packet exposes only the finite BEDC carrier rows over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. □

Theorem 699.52 (Real tail-agreement seal Completion consumer ledger)**.** *Every* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reaches the seal through the displayed ledger*

$$R, S, W, D, A, H, C, P, N.$$

The only sibling routes it may use are the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route of chapter 386 and the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *route of chapter 853; both consume the same finite paired tail-agreement boundary rather than a quotient stream, selected limit, host convergence proof, or external completion object.*

Proof. Start with Definition 699.8. It places the downstream consumer on the displayed tail-agreement carrier rows before any Real or Completion handoff can be read.

Apply Definition 699.48. Once the consumer is on R, S, W, H, C, P, N, the agreement surface is forced through $W \to D \to A$.

The sibling routes cited in the statement consume this same boundary. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter receives a finite seal row from its Cauchy-facing source, and the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ chapter receives the finite budget-selector packet before its Real-facing seal handoff; neither route adds a new coordinate to $Q = (R, S, W, D, A, H, C, P, N)$. □

Theorem 699.53 (Real tail-agreement seal budget-window coherence)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *completion consumer must first read the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget row from chapter 906 before it can consume the shared paired-window route of the accepted packet* $Q = (R, S, W, D, A, H, C, P, N)$*. The budget row fixes the finite tail request that the seal later displays through* W, D, A*.*

Proof. Start with Definition 699.52. It puts every completion consumer on the displayed ledger $R, S, W, D, A, H, C, P, N$ before any seal read.

The budget row is the finite request source for the paired tail windows. Reading it from chapter 906 determines which tail request the local W row can display.

Since the packet has no quotient stream, host real equality, selected limit, or ambient completeness coordinate, the consumer cannot bypass the budget row and later recover the same finite paired-window route. □

Theorem 699.54 (Real tail-agreement seal tail-meet factorization)**.** *After the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget row has been read, every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *completion consumer factors through the shared-threshold route of chapter 560 before reaching any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *sibling route. The factorization exposes the same finite* $W \to D \to A$ *agreement surface used by the local seal packet.*

Proof. Read the budget-window coherence row. It places the consumer on the finite tail request before the paired windows of Q are available.

Now use the shared-threshold sibling. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ route supplies the common threshold consumed by the same W row, and Definition 699.48 forces any public agreement read through $W \to D \to A$.

The $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ routes consume that agreement surface after the shared threshold has been displayed. They do not add a second tail source, quotient stream equality, host real equality, or selected limit to the local seal packet. □

Theorem 699.55 (Real tail-agreement seal completion route non-escape)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *completion route to* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *is confined to the sequence:* $\mathsf{RegularCauchyTailBudg}$ *budget row,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row, and the local* $W \to D \to A$ *agreement surface. No route may replace that sequence by quotient stream equality, host real equality, selected limit, or ambient completeness.*

Proof. Use budget-window coherence for the first step and tail-meet factorization for the second step. Together they force the consumer to read the budget row, then the shared-threshold route, and then the displayed local agreement surface.

Apply the completion consumer ledger. Once the route reaches Q, the only local rows available are $R, S, W, D, A, H, C, P, N$, and the only agreement route is $W \to D \to A$.

The excluded analytic objects have no coordinate in either the budget row, the shared-threshold sibling, or the local seal packet. Hence they cannot serve as alternate completion routes. □

Theorem 699.56 (Real tail-agreement seal completion budget-threshold exhaustion)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every* $\mathsf{Completion}^{\uparrow}$*-facing consumer of* Q *factors through the finite sequence consisting of the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget row, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row, the local* $W \to D \to A$ *agreement surface, and then the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *consumer route. No quotient stream equality, host real equality, selected limit, hidden tail selector, or ambient completeness row can appear in that factorization.*

Proof. Start with Definition 699.53. It places the Completion-facing consumer on the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ budget row before the paired window route of Q is read.

Next apply Definition 699.54. The budget row is carried through the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ shared threshold, and the local public agreement surface is forced to be $W \to D \to A$.

Finally use Definition 699.55. The only sibling consumers available after this surface are the $\mathsf{CauchyLimitSeal}^{\uparrow}$ route of chapter 386 and the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ route of chapter 853. Since none of the listed rows carries quotient stream equality, host real equality, a selected limit, a hidden tail selector, or an ambient completeness row, the displayed sequence exhausts the Completion-facing handoff. □

Theorem 699.57 (Real tail-agreement seal dual-source window exhaustion)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the two source rows* R, S *and the paired* $\mathsf{StreamName}^{\uparrow}$ *window row* W *exhaust the left/right tail evidence visible to Real and Completion consumers. The dyadic row* D *and agreement row* A *read that evidence, while* H, C, P, N *transport, route, package, and name it without adding another left tail, right tail, or paired-window selector.*

Proof. Use Definition 699.37. The public read reaches the paired finite windows through the single window row W after the two source rows R, S have been displayed.

Apply Definition 699.38. The selected windows are read back through D, and the visible agreement is recorded by A.

Finally apply Definition 699.48. A Real or Completion consumer that agrees on R, S, W, H, C, P, N must read $W \to D \to A$, so there is no remaining carrier coordinate that can serve as another left/right source or another paired-window selector. □

Theorem 699.58 (Real tail-agreement seal Real–Completion boundary)**.** $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *consumers of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receive the same finite paired agreement boundary*

$$R, S, W, D, A, H, C, P, N.$$

The boundary supplies source rows, paired windows, dyadic readback, displayed agreement, transport, routes, package provenance, and local naming; it supplies no quotient stream, selected limit, host convergence predicate, or hidden completion selector.

Proof. Apply Definition 699.8 to put the Real-facing read on the carrier boundary, and apply Definition 699.52 to put the Completion-facing read on the same displayed ledger.

The common agreement row is determined by the window and dyadic readback. This is exactly the route supplied by Definition 699.37, Definition 699.38, and Definition 699.48.

The non-escape part is finite. Since the packet has no coordinate for quotient streams, selected limits, host convergence, or hidden completion selectors, both kinds of consumer receive only the same paired agreement boundary. □

Theorem 699.59 (Real tail-agreement seal selector-budget handoff)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. If* Q *is consumed through a finite observation-budget selector, then the consumer first factors through chapter 782 and its finite route*

$$B \to S \to W \to D \to R \to E.$$

Only after that public selector route has been displayed can the consumer reach the local paired agreement route $W \to D \to A$*. The handoff exports no quotient stream, selected limit, host convergence row, hidden completion selector, or ambient completeness coordinate.*

Proof. Apply Definition 782.14. It makes the finite selector route $B \to S \to W \to D \to R \to E$ the public route seen by Real-completion consumers before the local seal is read.

Now use budget-window coherence for the present packet. The selector route supplies the finite budget and window service consumed by the local W row, so the seal cannot be reached through a separate unbounded observation request.

Finally apply Definition 699.48. Once the local rows R, S, W, H, C, P, N have been matched, every displayed agreement read proceeds through $W \to D \to A$, with no additional analytic coordinate exported by the handoff. □

Theorem 699.60 (Real tail-agreement seal finite-approximation consumer boundary)**.** *Every finite approximation consumer reaches an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet only after the* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *surface of chapter 601 has displayed the finite rows*

$$R, n, k, W, D, E, S, H, C, P, N.$$

The consumer then reads the local paired agreement boundary $R, S, W, D, A, H, C, P, N$ *through* $W \to D \to A$*. The boundary contains no selected real value, quotient stream equality, infinite tail, host metric, hidden completion selector, or ambient completeness row.*

Proof. Begin with Definition 601.16. It packages the finite approximation rows $R, n, k, W, D, E, S, H, C, P, N$ as the local $\mathsf{NameCert}_{\mathsf{FiniteCauchyApproximation}^{\uparrow}}$ obligation surface before any real-facing or completion-facing consumer can use the approximation.

Next apply Definition 601.18. The precision, finite-window, dyadic-ledger, error-row, and handoff data are consumed through the same finite budget route as the backing budget packets, so the approximation contributes no hidden completion choice.

Finally use the local completion route non-escape theorem. After the finite approximation rows have supplied the window and dyadic readback surface, the present seal can only read $W \to D \to A$ inside $R, S, W, D, A, H, C, P, N$, and none of the excluded analytic objects has a carrier coordinate. □

Theorem 699.61 (Real tail-agreement seal Real–Completion selector chain)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that passes through a finite observation-budget selector and a finite Cauchy approximation sees one finite chain: the selector route* $B \to S \to W \to D \to R \to E$*, the finite approximation rows* $R, n, k, W, D, E, S, H, C, P, N$*, the shared threshold route, and the local paired agreement route* $W \to D \to A$*. The chain exports the same Real–Completion boundary* $R, S, W, D, A, H, C, P, N$ *and no quotient stream, selected limit, host convergence, hidden completion selector, or ambient completeness row.*

Proof. Use Definition 699.59 for the budget-selector side. It forces the consumer through the displayed finite selector route before the local paired windows are available.

Use Definition 699.60 for the finite approximation side. It forces the same consumer through the approximation rows and then returns it to the local $W \to D \to A$ boundary.

Now apply Definition 699.58. The Real-facing and Completion-facing reads receive the same finite local packet, so the selector route, approximation rows, shared threshold, and local paired agreement route form one displayed chain rather than a route through quotient streams, selected limits, host convergence, hidden completion selectors, or ambient completeness. □

Falsifiable prediction: Deleting any one of R, S, W, D, A breaks a different public read of the seal: source membership for R, S, shared tail-window selection for W, dyadic exactness for D, and displayed agreement for A.

Independence witness: Nearest Real and Completion siblings share some source or handoff vocabulary, but none is bijective to this carrier while preserving the five distinct rows R, S, W, D, A: a single-source regularity packet lacks the paired source split, a tail-selector packet lacks dyadic agreement, a dyadic readback packet lacks the paired window lock, and a Real handoff packet lacks the displayed agreement boundary.

Theorem 699.62 (Real tail-agreement seal TasteGate independence boundary)**.** *The checked single-carrier TasteGate marker for* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *is read together with the displayed carrier rows, the falsifiable prediction, and the independence witness as preserving the distinct fields* R, S, W, D, A*. Thus the current* `FieldFaithful` *witness and any later* `Nontrivial` *witness have a non-vacuous paper boundary: they must respect source separation, paired-window source lock, dyadic readback, displayed agreement, and the Real-facing handoff instead of collapsing the packet to a single undifferentiated carrier row.*

Proof. The checked TasteGate marker recorded for the chapter supplies the single-carrier alignment for the carrier in Definition 699.1. It does not by itself identify the internal roles of the rows, so the paper boundary records the roles explicitly.

The falsifiable prediction separates the rows by deletion behavior: removing R, S, W, D, or A destroys a different public read. The independence witness separates the carrier from nearby Real and Completion packets by requiring the paired source split, the window lock, the dyadic readback, and the agreement row to be preserved together.

Apply Definition 699.47 and Definition 699.48. These theorems bind the row distinction and the consumer route $W \to D \to A$, so any faithful Lean-side field witness for this carrier has a concrete paper boundary to preserve. □

Theorem 699.63 (Real tail-agreement seal shared-tail budget extraction)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, every Real- or Completion-facing read of the paired agreement row* A *determines a shared finite tail-budget route through the budget packet of chapter 906 and the shared-threshold meet of chapter 560. The extracted route exposes the same paired window row* W *and dyadic readback row* D *used by* Q*, and it exposes no countable selector, quotient stream equality, host real equality, or selected limit outside the displayed rows.*

Proof. Start with the budget-window coherence row. It forces any public agreement read to pass through the finite request surface before the paired windows of Q can be consumed.

Next use the paired-window source lock. The source rows R, S and the window row W determine which dyadic row D and agreement row A are visible, so the consumer cannot replace the shared tail request by a hidden selector.

Finally factor the extracted request through the sibling meet route. The meet chapter supplies the shared threshold consumed by the same finite windows, while the present packet supplies only the local $W \to D \to A$ agreement surface. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier shared tail budget extraction`

Theorem 699.64 (Real tail-agreement seal window-agreement determinacy)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *and* $Q' = (R', S', W', D', A', H', C', P', N')$ *be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. If the source rows, shared budget route, paired window rows, and local transport and naming rows agree through the displayed* hsame *and* Cont *data, then the public agreement rows agree:* A *and* A' *are the same displayed paired-window agreement read. No Completion consumer can distinguish* Q *from* Q' *by adding a second agreement classifier outside* W, D, H, C, P, N*.*

Proof. Project both carriers and read the shared-tail budget extraction. The agreed budget route and paired source rows put both packets on the same finite tail request.

Apply the paired-window source lock to each packet. Since W and W' are the same displayed paired-window service under the local transport rows, both dyadic readbacks are read through the same finite window data.

The terminal consumer factorization then leaves only the agreement row as the public result of that route. Because the carrier has no second agreement classifier or host equality coordinate, A and A' are determined by the displayed window and readback rows. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier window agreement determinacy`

Theorem 699.65 (Real tail-agreement seal Completion handoff totality)**.** *Every* $\mathsf{Completion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reaches completion data only through the finite chain consisting of the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget route, the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-tail route, the local paired agreement surface* $W \to D \to A$*, and then the sibling consumer*

rows of chapter 386 or chapter 853. The handoff is total for displayed Completion consumers and supplies no Real completeness theorem, quotient stream, host metric limit, or choice-selected tail.

Proof. Use the Completion consumer ledger to put the consumer on the displayed rows $R, S, W, D, A, H, C, P, N$ before any Completion-facing read is available.

The shared-tail budget extraction supplies the finite request and threshold route. The window-agreement determinacy theorem then fixes the agreement row exposed to the consumer from W and D.

It remains to identify the Completion-facing endpoint. The Cauchy limit seal and diagonal budget siblings consume the displayed agreement boundary after it has been fixed; they do not add a selected limit or ambient completeness coordinate to the present packet. □

Theorem 699.66 (Real tail-agreement seal root-window readback)**.** *Every root* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *read of an accepted packet* $Q = (R, S, W, D, A, H, C, P, N)$ *factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows of chapter 301, the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows of chapter 291 before the local agreement row* A *is visible. The route is rooted in the regular-tail surfaces supplied by chapter 560 and chapter 658, and it exposes no independent agreement channel outside* $W \to D \to A$*.*

Proof. Start from the carrier in Definition 699.1. Its root-readable coordinates present the two regular rational source rows R, S, one paired finite window row W, one dyadic readback row D, and one agreement row A.

Use the source separation and paired-window source lock. They force any public agreement read to take the displayed sources into the same finite $\mathsf{StreamName}^{\uparrow}$ window service before the dyadic readback can be inspected.

Now apply the dyadic exactness and shared-tail budget extraction rows. The readback row D is the only dyadic route attached to the visible windows, and the regular-tail meet and fusion siblings provide the shared threshold surface consumed by that route.

Finally use the Completion handoff totality theorem. Once the consumer reaches the Completion-facing endpoint, the only visible local agreement has already passed through $W \to D \to A$, so no quotient stream equality, selected limit, or hidden agreement selector can be inserted at the root. □

Theorem 699.67 (Real tail-agreement seal tail-classifier transport)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *and* $Q' = (R', S', W', D', A', H', C', P', N')$ *be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. If the displayed* hsame *rows identify the* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, the* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks along the same finite route used in Definition 699.66, then the paired tail classifier is transported from* Q *to* Q' *with the same agreement row read. The transport is componentwise through* H, H' *and does not appeal to quotient stream equality, host convergence, or a choice-selected tail.*

Proof. Project both packets by the carrier definition. The transport data available to the theorem are exactly the componentwise hsame rows H, H' together with the displayed route rows C, P, N.

Apply the tail-classifier theorem for the present chapter and the regular-source transport theorem. The classifier is attached to the two $\mathsf{RegSeqRat}^{\uparrow}$ source rows as they pass through the same $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ readback route.

Use window-agreement determinacy to compare the two packets after transport. Once the sources, windows, readbacks, local transports, and names agree, the public paired agreement row is fixed.

Thus the paired tail classifier moves across the same finite route as the root-window readback. Since no carrier coordinate stores quotient stream equality, host convergence, or a selected tail, there is no second transport path for the classifier. □

Theorem 699.68 (Real tail-agreement seal completion consumer scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet reaches the seal through the finite route consisting of the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{StreamName}^{\uparrow}$ *paired windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows, the agreement edge* $W \to D \to A$*, the regular-tail meet and fusion surfaces, and the sibling consumer rows of chapter 38, chapter 95, and chapter 386. No such consumer reaches the seal through quotient stream equality, a selected limit, host convergence, or a choice-selected tail.*

Proof. Begin with the Real–Completion boundary and Completion handoff totality theorems. They put both Real-facing and Completion-facing reads on the displayed packet rows before any sibling consumer can use the seal.

Apply the root-window readback theorem above. It shows that the agreement row is visible only after the source, window, and dyadic readback rows have been displayed along the regular-tail route.

Now use the tail-classifier transport theorem above. The same route preserves the paired classifier under the local hsame rows, so a consumer cannot switch to an ambient stream equality or host convergence predicate while keeping the same public classifier.

The remaining consumers are exactly the named Real, Completion, and Cauchy-limit-seal siblings. They receive the finite dependency rows already displayed by the packet and add no coordinate for quotient stream equality, selected limits, host convergence, or choice-selected tails. □

Theorem 699.69 (Real tail-agreement seal root source-window admission)**.** *Every root read of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$ *first admits the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* R, S *and the paired* $\mathsf{StreamName}^{\uparrow}$ *window row* W *before any agreement row can be consumed. The admission route is the carrier route of Definition 699.1 together with the NameCert source-separation and paired-window obligations, and it exposes no source object outside those finite rows.*

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealTerminalRoute root source window admission`

Proof. Project Q by the carrier definition. The only source coordinates visible at the root are R, S, and the only window coordinate attached to them is the displayed paired row W.

Apply the source separation obligation and the paired-window source lock. These obligations force each root read to pass through the $\mathsf{RegSeqRat}^{\uparrow}$ source rows and then the $\mathsf{StreamName}^{\uparrow}$ paired-window row before any local agreement data is available.

The root-window readback theorem then identifies this admitted window as the first leg of the public readback route. Since no additional source or window coordinate exists in the carrier, the root surface cannot expose an alternative source-window path. □

Theorem 699.70 (Real tail-agreement seal root dyadic agreement exactness)**.** *For every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the agreement row* A *is root-readable only after the displayed dyadic route* $W \to D$ *has been read. The exact route is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D*, and then the local agreement row* A*; it is not a quotient stream equality, host convergence predicate, or selected tail.*

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root dyadic agreement exactness`

Proof. Begin with the root source-window admission theorem. It places the root read on the displayed paired window row W before the agreement row can be inspected.

Use the dyadic window exactness and dyadic terminal nonescape obligations. They make D the only dyadic readback attached to W and leave no second dyadic carrier for the consumer to choose.

Now apply root-window readback and tail-classifier transport. The public classifier is transported through the same $W \to D$ route, so the agreement row A is reached exactly after that finite dyadic read and not through an ambient equality or convergence principle. □

Theorem 699.71 (Real tail-agreement seal root Completion boundary)**.** *Every root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives only the finite rows* $R, S, W, D, A, H, C, P, N$ *and the named sibling routes through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *regular-tail meet and fusion,* $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and Cauchy-limit-seal consumers. The boundary supplies no quotient stream equality, selected limit, host convergence, or choice-selected tail.*

Proof. Read the consumer through the Completion consumer scope theorem. It lists the packet rows and the named sibling consumer routes available at the Real and Completion boundary.

Combine this with root source-window admission and root dyadic agreement exactness. The source, window, dyadic readback, and agreement rows have already been exposed in the order R, S, W, D, A before any sibling consumer receives the packet.

The carrier supplies the remaining local transport, continuity, package, and NameCert rows H, C, P, N. Since neither the carrier nor the cited sibling routes contain a coordinate for quotient stream equality, selected limits, host convergence, or choice-selected tails, the root Completion boundary is exhausted by the displayed finite rows. □

Theorem 699.72 (Real tail-agreement seal root Completion-window totality)**.** *Every root Completion-facing read of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet factors through the ordered route consisting of the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* R, S, *the* $\mathsf{StreamName}^{\uparrow}$ *paired window row* W, *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row* D, *the local agreement row* A, *the regular-tail meet and fusion surfaces, and then the* $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer rows. The route is total for such root reads and supplies no quotient stream equality, selected limit, host convergence, or choice-selected tail.*

Proof. Start with root source-window admission. It places the root read first on the two displayed $\mathsf{RegSeqRat}^{\uparrow}$ source rows and then on the paired $\mathsf{StreamName}^{\uparrow}$ window row, so the consumer has no source-window route outside R, S, W.

Next use root dyadic agreement exactness. The admitted window is read back through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and the local agreement row A is available only after that finite $W \to D$ leg.

Now apply the Completion handoff totality theorem. It sends every Completion-facing consumer from the local agreement surface through the regular-tail meet and fusion route and then to the named Completion-side consumer rows.

Finally combine the root Completion boundary with the Completion consumer scope theorem. These two boundary statements identify the remaining visible endpoints as the $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer rows already named by the chapter. Since each step consumes only displayed rows of the packet or named sibling surfaces, no quotient stream equality, selected limit, host convergence, or choice-selected tail can enter the root Completion-facing read. □

Theorem 699.73 (Real tail-agreement seal root budget-selector admission)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every root-facing accepted read of Q that is consumed by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *endpoint first reads the finite* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *limiter row, the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *tail-budget row, and the* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *selector-budget row before the local agreement edge*

$$W \to D \to A$$

is public. Hence the root admission route is one finite L10 route through the limiter, tail-budget, selector-budget, paired window, dyadic readback, and agreement rows, and not an uncapped observation request or a choice-selected tail.

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root budget selector admission`

Proof. Project the accepted carrier Q. The local agreement row A is attached to the displayed paired window row W and dyadic readback row D, while the structural rows H, C, P, N carry only transport, displayed continuations, provenance, and the local naming certificate.

Use the budget-source window service and the shared-threshold budget exactness rows. They place the root read behind the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ and $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ services before any paired window row can be exposed.

Now use Definition 699.59. A root consumer that also uses the finite selector route must pass through the $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ selector-budget surface before the local windows can be consumed.

Finally apply the window-route exact companion. After the three budget rows have admitted the finite window, the only public agreement path is $W \to D \to A$, so the route has no coordinate for an uncapped request, quotient stream, selected limit, host convergence row, or choice-selected tail. □

Theorem 699.74 (Real tail-agreement seal root budget shared-threshold factorization)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet $Q = (R, S, W, D, A, H, C, P, N)$, the shared-threshold row used by any root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

The $\mathsf{StreamName}^{\uparrow}$ *step is the displayed paired window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *step is the readback row D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *step is the agreement readback consumed by A.*

Proof. Begin with root budget-selector admission. It supplies the limiter, tail-budget, and selector-budget entrance before the local agreement edge can be read.

Next read the shared threshold through the tail-meet factorization. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row is the only shared threshold available to the local paired-window service, and it is reached after the limiter and tail-budget rows have fixed the finite request.

The window-meet coverage and dyadic readback exactness rows identify the middle of the route. The shared threshold opens the displayed $\mathsf{StreamName}^{\uparrow}$ window W, the window is read back through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and the regular rational agreement read is recorded by the local row A.

Use the Real–Completion selector chain to bind the selector-budget row to the same route. Since the accepted carrier has no second threshold, second selector, quotient stream, host convergence predicate, or selected limit coordinate, the displayed L10 route is the complete factorization of the shared-threshold row. □

Theorem 699.75 (Real tail-agreement seal root budget Completion-consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer admitted by the root budget-selector route of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives only*

$$R, S, W, D, A, H, C, P, N$$

together with the named sibling routes through $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It receives no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface.*

Proof. Start from the shared-threshold factorization above. It binds the limiter, tail-budget, tail-meet, selector-budget, window, dyadic readback, regular agreement readback, and local agreement row into one finite root route.

Now apply the root Completion boundary and the Real–Completion selector chain. They put both Real-facing and Completion-facing endpoints on the same displayed local carrier rows after the sibling budget and selector surfaces have been read.

It remains to account for the named downstream consumers. The tail-fusion and $\mathsf{CauchyLimitSeal}^{\uparrow}$ routes consume the same finite agreement surface, and the $\mathsf{Real}^{\uparrow}$ endpoint receives the same local seal row reached through $W \to D \to A$.

The carrier rows are exhausted by $R, S, W, D, A, H, C, P, N$, with H, C, P, N serving only transport, displayed continuation, provenance, and naming roles. Since neither the carrier nor the named sibling routes add a quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement row, every admitted consumer remains inside the stated boundary. □

Theorem 699.76 (Real tail-agreement seal selector shared-threshold exactness)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Any root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that first passes through the budget-selector admission route and then asks for the shared-threshold row factors through the same finite chain*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudge}$$

The selector-budget row and the shared-threshold row are therefore not two independent admissions: they are the same displayed L10 route through the limiter, tail-budget, tail-meet, selector-budget, paired-window, dyadic-readback, regular-rational readback, and local agreement rows of Q.

Proof. Start with Definition 699.73. It fixes the root-facing consumer behind the $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, and $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ rows before the local agreement edge $W \to D \to A$ is public.

Next apply Definition 699.74. The requested shared-threshold row is not an extra coordinate of Q; it is the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ step in the same finite route, after the limiter and tail-budget rows and before the selector-budget, window, dyadic readback, regular-rational readback, and agreement rows.

Finally use Definition 699.75. Completion-facing consumption has no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface from which a different threshold could be read.

Combining these three local results gives exactness of the route: any consumer admitted by the selector budget and then reading the shared threshold must traverse the displayed chain above, and every row in that chain is one of the finite L10 carrier or named sibling rows already exposed by the accepted packet. □

Theorem 699.77 (Real tail-agreement seal root budget selector cross-sibling lock)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Any root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer admitted by the selector-budget route and any consumer reading the shared-threshold row traverse the same finite L10 chain*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

Thus selector admission and shared-threshold exactness are one route of Q*: no second selector, second threshold, quotient stream, selected limit, host convergence row, or choice-selected tail can be inserted between the root read and the local agreement row.*

Proof. Apply Definition 699.73. It places every admitted root-facing $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer behind the limiter, tail-budget, and selector-budget rows before $W \to D \to A$ is exposed.

Next use Definition 699.74. The shared threshold is the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row in that finite chain, not a sibling coordinate parallel to the selector-budget row.

Apply Definition 699.75 to the consumer side. The accepted packet has no coordinate for a quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface.

Finally invoke Definition 699.76. Coordinate exhaustion of $Q = (R, S, W, D, A, H, C, P, N)$ identifies the selector-admission consumer and the shared-threshold consumer with the same displayed L10 chain, so the two sibling rows are locked to a single finite route. □

Theorem 699.78 (Real tail-agreement seal root selector tail-fusion lock)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet admitted by the root budget-selector route. Every downstream* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of that admitted packet factors through the same finite chain*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

No such consumer may replace this route by a quotient stream, selected limit, host convergence row, choice-selected tail, or second agreement surface.

Proof. Apply the root budget-selector admission theorem. It places the accepted read behind the limiter, tail-budget, and selector-budget rows before the local agreement edge $W \to D \to A$ is public.

Use the shared-threshold factorization and selector shared-threshold exactness. They identify the shared-threshold request with the same finite chain through the tail-meet, selector-budget, window, dyadic readback, regular-rational readback, and local agreement rows.

Now apply completion-consumer exhaustion. The only downstream routes available to completion-facing consumers are the named $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ routes over the same carrier rows $R, S, W, D, A, H, C, P, N$.

The Cauchy-limit seal root handoff and the $\mathsf{Real}^{\uparrow}$ consumer rows are therefore only downstream names for consumption of this finite route. They contribute no quotient stream, selected limit, host convergence row, choice-selected tail, or second agreement surface, so tail-fusion and limit-seal consumers remain locked to the selector route. □

Theorem 699.79 (Real tail-agreement seal root selector completion lattice)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet admitted by the root budget-selector route. Any root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of* Q *factors through the finite completion lattice*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

before the completion endpoint is public. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *step is the local packet row* A *together with the NameCert row* N*, and the final two steps are downstream consumers of that same finite agreement surface.*

Proof. Begin with Definition 699.73. It places every root-facing consumer behind the limiter, tail-budget, and selector-budget rows before $W \to D \to A$ is exposed.

Use Definition 699.74. The shared threshold supplies the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ step and identifies the middle of the route with the displayed $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ readback, and the $\mathsf{RegSeqRat}^{\uparrow}$ agreement readback.

Apply Definition 699.76 and Definition 699.77. They bind selector admission and shared-threshold consumption to the same finite chain, so there is no second selector or second threshold between the root read and the agreement row.

Finally use Definition 699.75 and Definition 699.78. The downstream $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers are only names for consuming the same local seal certified by N, hence the completion endpoint is reached through the displayed lattice and not through an ambient completion principle. □

Theorem 699.80 (Real tail-agreement seal root selector window non-escape)**.** *For an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$ *admitted by the root budget-selector route, every window exposed to a root-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer is the displayed finite window* W *read back through* D *and certified by* A *and* N*. Along the completion-lattice route there is no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface.*

Proof. The admission theorem first fixes the root request behind the limiter, tail-budget, and selector-budget rows. Thus the only window made public before agreement is the displayed paired window W.

The shared-threshold factorization and selector shared-threshold exactness then force any threshold read through the same tail-meet, selector-budget, window, dyadic-readback, and agreement rows. This eliminates any alternate threshold from which a second window could be read.

Use completion-consumer exhaustion to inspect the carrier coordinates. The packet exposes only $R, S, W, D, A, H, C, P, N$ together with named sibling routes, and H, C, P, N provide transport, displayed continuation, provenance, and NameCert data rather than a second analytic surface.

The cross-sibling lock and tail-fusion lock carry that coordinate exhaustion to downstream consumers. Since $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consume the same finite route, none of the forbidden escape surfaces can be inserted along the root selector window path. □

Theorem 699.81 (Real tail-agreement seal root selector consumer totality)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet admitted by the root budget-selector route. Every root-facing consumer whose endpoint is* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$*, including the downstream* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers, is accounted for by the completion-lattice factorization and the window non-escape boundary. Consequently the root selector surface is total for this finite NameCert route: each admitted consumer reaches the endpoint through the displayed L10 chain and no admitted consumer remains outside it.*

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root selector consumer totality`

Proof. First assemble the entry route. Apply Definition 699.73, Definition 699.74, and Definition 699.76 to put each admitted consumer on the limiter, tail-budget, tail-meet, selector-budget, window, dyadic-readback, and agreement chain.

Next lock the sibling surfaces. Use Definition 699.75, Definition 699.77, and Definition 699.78; together they say that all named completion-facing consumers use the same finite carrier rows and the same downstream tail-fusion, limit-seal, and real endpoints.

Now apply Definition 699.79. It presents those local and sibling rows as one completion-lattice route ending at the public $\mathsf{Real}^{\uparrow}$ endpoint.

Finally apply Definition 699.80. Since no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface is available, there is no remaining root-facing consumer outside the finite NameCert route. □

Theorem 699.82 (Real tail-agreement seal root selector ProbeBundle budget coverage)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet admitted by the root budget-selector route. Every finite* ProbeBundle *of root-facing* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *requests for Q is covered by the same completion-lattice chain*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

The bundle therefore consumes one finite budget surface: the displayed limiter, tail-budget, tail-meet, selector-budget, stream window, dyadic readback, rational-sequence readback, agreement row, limit-seal row, and public real endpoint.

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root selector probe bundle budget coverage`

Proof. Inspect the finite ProbeBundle one request at a time. For each member, Definition 699.73 places the root read behind the limiter, tail-budget, selector-budget, window, dyadic readback, and agreement rows of Q.

Next apply Definition 699.74. The shared threshold is common to all bundle members, so the tail-meet and selector-budget rows are not duplicated when the finite bundle is traversed.

For the completion-facing members, use Definition 699.75 and Definition 699.79. They identify each $\mathsf{Completion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ read with the same displayed route through A and N.

Finally apply Definition 699.80. Since no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface can be inserted, pointwise coverage over the finite bundle repacks into one finite ProbeBundle coverage row. □

Theorem 699.83 (Real tail-agreement seal root selector endpoint determinacy)**.** *Let $Q = (R, S, W, D, A, H, C, P, N$ and $Q' = (R', S', W', D', A', H', C', P', N')$ be admitted root-selector consumers for* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*. If their displayed limiter, tail-budget, tail-meet, selector-budget, stream window, dyadic readback, and agreement rows coincide, then they reach the same public endpoint route. In particular, the route cannot split into a second selector, second threshold, quotient stream, selected limit, or host convergence row.*

Proof. Begin with the shared displayed rows. The admission theorem and shared-threshold factorization force both consumers through the same limiter, tail-budget, tail-meet, selector-budget, stream window, dyadic readback, and agreement rows before any completion endpoint is visible.

Apply Definition 699.77. It prevents sibling consumers from reusing the displayed agreement row while changing the selector, threshold, or local seal surface.

Next use Definition 699.78. The downstream tail-fusion, limit-seal, and real endpoint reads are determined by the same agreement row and NameCert packet, so the endpoint route is not an independent choice.

Finally combine Definition 699.79 with Definition 699.81. Every admitted consumer is already exhausted by the displayed completion lattice, and the matching displayed rows leave no coordinate where a quotient stream, selected limit, or host convergence row could separate the two endpoint routes. □

Theorem 699.84 (Real tail-agreement seal root selector bundle endpoint exactness)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet admitted by the root selector route. Any finite* $\mathsf{ProbeBundle}$ *of root-facing* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *requests whose members preserve the displayed limiter, tail-budget, tail-meet, selector-budget,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, agreement row, and local* $\mathsf{NameCert}$ *row determines exactly one public endpoint route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

Thus bundle coverage and endpoint determinacy are the same finite public route for Q, with no second selector, second threshold, quotient stream, selected limit, host convergence row, choice-selected tail, or independent completion endpoint.

Proof. Start with $\mathsf{ProbeBundle}$ budget coverage. Theorem 699.82 places each member of the finite bundle on the displayed completion-lattice chain, and it identifies the consumed rows as the limiter, tail-budget, tail-meet, selector-budget, stream window, dyadic readback, rational-sequence readback, agreement row, limit-seal row, and public real endpoint.

Next use completion-lattice exactness and window non-escape. Theorem 699.79 fixes the order of the rows through $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while Definition 699.80 prevents any bundle member from leaving the displayed window and readback rows.

Now apply consumer totality. Theorem 699.81 accounts for every root-facing $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ request in the finite bundle, including the downstream tail-fusion, limit-seal, and real consumers.

Finally apply endpoint determinacy. Since the bundle members preserve the displayed budget, window, readback, agreement, and local naming rows, Definition 699.83 leaves only one public endpoint route. The excluded selector, threshold, quotient, limit, host-convergence, and choice-tail surfaces are not coordinates of the accepted packet, so the finite bundle determines the single route displayed above. □

Theorem 699.85 (Real tail-agreement seal limiter-budget factorization)**.** *Every paired tail-agreement read of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet $Q = (R, S, W, D, A, H, C, P, N)$ factors first through the finite limiter rows of chapter 903 and the regular tail-budget rows of chapter 906. Only after those two budget services expose the shared threshold may the local route read the displayed* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback of chapter 291, and the agreement edge $W \to D \to A$.*

Proof. Start with the carrier of Definition 699.1. Its only agreement channel is the displayed row A, and that row is attached to the paired window row W and dyadic readback row D.

Apply Definition 699.63. It sends every Real- or Completion-facing agreement read through the shared finite tail request and the sibling budget route before W and D are consumed.

Finally apply Definition 699.65. The Completion endpoint reads only the route already exposed by the limiter and tail-budget services, so the local packet contributes the finite handoff $W \to D \to A$ and no unbounded observation request. □

Theorem 699.86 (Real tail-agreement seal shared budget route)**.** *Every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *read of the agreement row A factors through the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *route of chapter 906 and the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *route of chapter 903 before the local consumer can inspect the finite chain $W \to D \to A$. The displayed packet rows are therefore the only route by which the Real and Completion siblings receive a tail-agreement seal.*

Proof. Apply the budget source lock of Definition 699.10. It places the accepted seal read on the finite regular-tail budget source before the paired window service can be consumed.

Next apply the shared-tail budget extraction theorem Definition 699.63. The extracted request is the same request that feeds the local paired window row W and dyadic readback row D.

Finally use the Completion handoff totality row of Definition 699.65. The handoff reaches Real and Completion siblings only after the budgeted route has exposed $W \to D \to A$, so no unbudgeted seal read remains. □

Theorem 699.87 (Real tail-agreement seal paired-threshold coverage)**.** *For every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet $Q = (R, S, W, D, A, H, C, P, N)$, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks are covered by one shared threshold route before the agreement row A is visible. The route first displays the shared budget request, then the paired window row W, then the dyadic readback row D, and only then the local agreement edge $W \to D \to A$.*

Proof. Begin with the shared budget route theorem above. It puts every accepted agreement read behind the regular-tail budget and observation limiter before the packet can expose W.

Use the shared-tail budget extraction theorem again. It identifies the sibling tail-meet threshold as the single finite threshold route used by the paired windows and by the dyadic readbacks.

Now apply the Completion handoff totality theorem. Since the Completion-facing endpoint reads the same displayed $W \to D \to A$ chain, visibility of A is delayed until both the paired windows and dyadic readbacks have been covered by the shared threshold. □

Theorem 699.88 (Real tail-agreement seal Completion-consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives only the displayed rows $R, S, W, D, A, H, C, P, N$ together with the named sibling budget, tail-meet,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surfaces. The boundary supplies no quotient stream, selected limit, host convergence, or choice-selected tail to the consumer.*

Proof. Start from the paired-threshold coverage theorem above. It shows that the public agreement edge has already passed through the shared budget route, the paired windows, and the dyadic readbacks before a Real or Completion sibling can consume the packet.

Apply Definition 699.65. This gives the total handoff from the local packet to the sibling Completion-facing routes without adding a selected limit or host convergence predicate.

Finally apply Definition 699.68. The consumer scope lists the displayed rows and named sibling surfaces available to Real and Completion consumers; since that list has no quotient stream, selected limit, host convergence, or choice-selected tail coordinate, the boundary is exhausted by those rows. □

Theorem 699.89 (Real tail-agreement seal regular-source transport)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ and $Q' = (R', S', W', D', A', H', C', P', N')$ be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. If the componentwise* hsame *rows in H, H' identify the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows through chapter 301, the displayed* $\mathsf{StreamName}^{\uparrow}$ *window rows through chapter 126, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows through chapter 291, then the two regular sources are transported together to the same displayed paired-window agreement row.*

Proof. Project the two carriers from Definition 699.1. The source rows R, S and R', S' are not ambient streams; they are the displayed regular-rational source entries consumed by the window and readback rows.

Use Definition 699.63 to put both packets on the same finite tail-budget surface. The componentwise hsame rows then carry the two $\mathsf{RegSeqRat}^{\uparrow}$ sources through the same $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ readback rows.

Apply Definition 699.64. Once the source transport, shared budget, windows, readbacks, local transports, and naming rows agree, the public agreement row is determined; no separate quotient-stream equality or Completion equality row is available. □

Theorem 699.90 (Real tail-agreement seal Real handoff exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet receives only the finite route*

$$R, S, W, D, A, H, C, P, N.$$

The route is reached after the limiter-budget factorization and regular-source transport rows above, and it is the only handoff made available to the Real and Completion sibling surfaces of chapter 38 and chapter 95. It exports no quotient stream equality, selected limit, host convergence predicate, or choice-selected tail.

Proof. Begin with the carrier of Definition 699.1. It lists exactly the regular source rows, paired window, dyadic readback, agreement row, transports, routes, provenance, and local naming row that a Real-facing consumer can inspect.

Apply Definition 699.85. This puts the agreement read behind the observation limiter and regular tail-budget services before the local route $W \to D \to A$ is available.

Next apply Definition 699.89. The source rows are preserved through the displayed StreamName window and DyadicRatCore readback rows, so the Real-facing consumer cannot replace them by a quotient stream or selected tail.

Finally apply Definition 699.65. The same finite packet is the boundary seen by Completion-facing consumers, and the Real-facing handoff has no extra carrier coordinate for a selected limit or host convergence predicate. □

Theorem 699.91 (Real tail-agreement seal shared-threshold window admission)**.** *For every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows carried by* W *are admitted only after the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *limiter rows and the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *threshold rows have been displayed. Thus any public read of* A *first exposes the finite observation budget, then the regular tail-budget threshold, and only then the paired window service used by* W*.*

Proof. Project the carrier of Definition 699.1. The only window service available to the packet is the displayed row W, and it is attached to the paired source rows R, S before the readback row D and agreement row A are visible.

Use the NameCert obligations of Definition 699.2. Carrier admission, tail-window coverage, and dyadic readback exactness force the public route to encounter W before D and A.

Apply the limiter-budget factorization theorem above and the shared-tail budget extraction row. Together they put that same W behind the observation limiter and regular tail-budget threshold rows, so no accepted packet admits its paired windows through an unbudgeted tail request. □

Theorem 699.92 (Real tail-agreement seal dyadic readback route totality)**.** *Every dyadic readback route of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet is total only through the displayed chain*

$$W \to D \to A.$$

The chain reads the paired $\mathsf{StreamName}^{\uparrow}$ *windows, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *agreement row consumed by* $\mathsf{Real}^{\uparrow}$*; no route from the accepted packet reaches* A *by skipping* W *or* D*.*

Proof. Begin with the shared-threshold window admission theorem. It places every accepted paired window read behind the limiter and regular tail-budget rows before the local route reaches W.

Next apply the regular-source transport theorem above. The two $\mathsf{RegSeqRat}^{\uparrow}$ source rows are transported through the displayed $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ readback rows, so the source data seen by the agreement row remain on the same finite route.

Finally use dyadic readback exactness and window-agreement determinacy. These rows make D the readback surface determined by W, and make A the agreement row determined by that readback surface; hence the public dyadic readback route is total exactly along $W \to D \to A$. □

Theorem 699.93 (Real tail-agreement seal Real completion consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet is exhausted by the finite route*

$$R, S, W, D, A, H, C, P, N.$$

This route is read together with the sibling budget, tail-meet, tail-fusion, $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *packets; it supplies no quotient stream equality, selected limit, host convergence predicate, open completion selector, or additional consumer row outside the displayed carrier.*

Proof. Start from the Real handoff exhaustion theorem above. It states that a Real-facing consumer can inspect only the displayed rows $R, S, W, D, A, H, C, P, N$ after the limiter-budget and regular-source transport routes have been exposed.

Use the shared-threshold window admission and dyadic readback route totality theorems. They identify the budgeted entrance into W and the complete readback path $W \to D \to A$ used by both Real- and Completion-facing consumers.

Now compose the sibling routes already present in the chapter. The shared-tail budget extraction supplies the budget row, the regular-tail meet and fusion chapters supply the finite tail surfaces, and the Completion handoff totality theorem routes the consumer through the Cauchy-limit and Real sibling packets.

The remaining carrier rows H, C, P, N provide componentwise transport, continuation, provenance, and local naming. Since these are the only structural rows in the accepted carrier, the consumer surface has no place to store a quotient stream equality, selected limit, host convergence predicate, or extra completion selector. □

Theorem 699.94 (Real tail-agreement seal CauchyLimitSeal budget handoff)**.** *Suppose an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet and an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet share the same regular source rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal surface. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the CauchyLimitSeal budget source totality row before reading the local agreement edge*

$$W \to D \to A.$$

The handoff exports no quotient stream, selected limit, host convergence predicate, hidden family limit, or ambient completeness coordinate.

Proof. Begin with paired-threshold coverage from Definition 699.87. It places the agreement packet behind the common finite threshold route before the local rows W, D, A are read.

Use Definition 699.93. Any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer is already confined to the displayed rows $R, S, W, D, A, H, C, P, N$ together with the named sibling budget and completion surfaces.

Now apply CauchyLimitSeal budget root source totality from Definition 386.87. The shared regular source, finite window, dyadic readback, and real seal rows must pass through the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet before the local agreement edge is exposed.

Attach dyadic-window exactness from Definition 386.88 and the dyadic readback route totality of Definition 699.92. These rows identify the budget window with the seal-side finite ledger and then read the local route exactly as $W \to D \to A$.

Finally apply root completion consumer non-escape from Definition 386.94. After the consumer has passed through the budget source totality row and the local agreement edge, no quotient stream, selected limit, host convergence predicate, hidden family limit, or ambient completeness coordinate remains available. □

Theorem 699.95 (Real tail-agreement seal budget Real handoff)**.** *For every accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after the route through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$*. The local rows exposed to that consumer are exactly*

$$R, S, W, D, A, H, C, P, N.$$

The handoff exports no quotient stream, selected limit, host convergence predicate, or choice-selected tail.

Proof. First bind the finite budget entrance. Apply Definition 699.85 and Definition 699.91. They put every accepted read behind the observation limiter and regular tail-budget threshold before the paired $\mathsf{StreamName}^{\uparrow}$ window row W is visible.

Next bind the local readback. Use Definition 699.92. The route from W to the agreement row must pass through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D, so the local edge is exactly $W \to D \to A$.

Now attach the CauchyLimitSeal boundary. Apply Definition 699.94. It forces the shared regular source, finite window, dyadic readback, and $\mathsf{Real}^{\uparrow}$ seal surface through the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet before any Real or Completion consumer can read the local agreement edge.

Finally use Definition 699.93. Its non-escape clause confines the consumer to $R, S, W, D, A, H, C, P, N$ and leaves quotient streams, selected limits, host convergence predicates, and choice-selected tails outside the handoff. □

Theorem 699.96 (Real tail-agreement seal root budget route determinacy)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *and* $Q' = (R', S', W', D', A', H', C', P', N')$ *be accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packets. Suppose the two packets share the same* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget rows, and suppose the displayed rows agree componentwise:*

$$R = R', \quad S = S', \quad W = W', \quad D = D', \quad A = A', \quad H = H', \quad C = C', \quad P = P', \quad N = N'.$$

Then Q *and* Q' *determine the same public* $\mathsf{Real}^{\uparrow}/\mathsf{Completion}^{\uparrow}$ *handoff route. The route is the displayed chain through the shared limiter budget, shared regular Cauchy tail budget, shared stream window, shared dyadic readback, shared CauchyLimitSeal budget packet, and the same public rows*

$R, S, W, D, A, H, C, P, N$. No quotient stream, selected limit, host real equality, hidden family limit, or ambient completeness coordinate is available to distinguish the two handoffs.

Proof. First apply limiter-budget factorization. By Definition 699.85, both accepted packets enter the public handoff through the same finite observation limiter and the same regular tail-budget threshold rows.

Next use dyadic readback route totality. Definition 699.92 fixes the local read of the shared window and dyadic rows as the displayed route $W \to D \to A$; componentwise agreement makes this local route identical for Q and Q'.

Attach the CauchyLimitSeal budget handoff with Definition 699.94. Since the $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet is shared, the regular source, finite window, dyadic readback, and seal-facing budget rows are the same public rows on both sides.

Finally apply the budget Real handoff theorem above. Definition 699.95 confines every $\mathsf{Real}^{\uparrow}$- or $\mathsf{Completion}^{\uparrow}$-facing read to the same displayed rows and named budget surfaces. The excluded quotient stream, selected limit, host real equality, hidden family limit, and ambient completeness coordinate are not fields of either handoff, so they cannot separate the two public routes. □

Theorem 699.97 (Real tail-agreement seal budget-fusion non-escape)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ and $Q' = (R', S', W', D', A', H', C', P', N')$ be accepted $\mathsf{RealTailAgreementSeal}^{\uparrow}$ packets. Suppose the two packets share the same limiter budget, regular Cauchy tail budget, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ readback, $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet, and displayed carrier rows:*

$$R = R', \quad S = S', \quad W = W', \quad D = D', \quad A = A', \quad H = H', \quad C = C', \quad P = P', \quad N = N'.$$

Then every $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer of Q and Q' factors through the same finite budget-fusion surface

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} -$$

After this factorization the consumer cannot distinguish the two handoffs by quotient stream equality, selected limit, host convergence, choice-selected tail, hidden family limit, or ambient completeness data.

Proof. First bind the shared finite budget entrance. Apply Definition 699.85. Both accepted packets must pass through the same observation limiter and regular Cauchy tail-budget rows before any local agreement row can be read.

Next bind the displayed window and readback route. Use Definition 699.92. The shared $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows make the local path exactly $W \to D \to A$ on both packets, and the componentwise row equalities identify the displayed carrier surface.

Now attach the completion-facing budget packet. Apply Definition 699.94. The shared $\mathsf{CauchyLimitSeal}^{\uparrow}$ packet receives the same regular source, finite window, dyadic readback, and $\mathsf{Real}^{\uparrow}$ seal surface before any Real or Completion consumer can inspect the handoff.

Finally use root budget route determinacy together with the Real and Completion consumer exhaustion rows. Definition 699.96 identifies the public handoff route, while Definition 699.93 and Definition 699.95 list the available consumer rows. Quotient stream equality, selected limits, host convergence predicates, choice-selected tails, hidden family limits, and ambient completeness data are outside that list, so they cannot distinguish the two handoffs. □

Theorem 699.98 (Real tail-agreement seal budget source totality)**.** *Every accepted $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ through the budget route is sourced exactly by the limiter*

budget, regular tail budget, finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *budget packet, and displayed rows* $R, S, W, D, A, H, C, P, N$*. It cannot introduce an unlisted tail source, quotient stream equality, host convergence predicate, selected limit, or ambient completeness coordinate.*

Proof. First apply limiter-budget factorization from Definition 699.85. It places the accepted consumer behind the observation limiter and regular Cauchy tail-budget rows before any agreement seal is available.

Next use dyadic readback route totality from Definition 699.92. The finite $\mathsf{StreamName}^{\uparrow}$ window reaches the agreement row only through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ readback, so the local route is fixed by W, D, A.

Attach the completion-facing budget packet with Definition 699.94. Then apply the budget Real handoff of Definition 699.95. These two rows put the regular source, finite window, dyadic readback, and $\mathsf{Real}^{\uparrow}$ seal surface on the same $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet.

Finally use budget-fusion non-escape from Definition 699.97. The consumer has already factored through the limiter, regular tail budget, window, readback, CauchyLimitSeal packet, and displayed carrier rows, so no unlisted tail source, quotient stream equality, host convergence predicate, selected limit, or ambient completeness coordinate remains. □

Theorem 699.99 (Real tail-agreement seal top formal target boundary)**.** *Any formalization target for* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *that requests a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer must first expose the finite route through* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$*,* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The target is therefore a boundary theorem about the displayed packet* $Q = (R, S, W, D, A, H, C, P, N)$ *and the named sibling budget route, not a theorem about quotient streams, selected limits, host convergence predicates, or ambient completeness.*

Proof. First bind the budget entrance with Definition 699.85 and Definition 699.86. These rows force any accepted seal read through the observation limiter and regular Cauchy tail-budget rows before the local route can inspect W, D, or A.

Next bind the local readback with Definition 699.92. The route to the agreement row is total only as $W \to D \to A$, so a formal target cannot bypass the $\mathsf{StreamName}^{\uparrow}$ window or the $\mathsf{DyadicRatCore}^{\uparrow}$ readback.

Then attach the completion-facing seal boundary using Definition 699.94. It sends the shared regular source, finite window, dyadic readback, and $\mathsf{Real}^{\uparrow}$ seal surface through the accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ budget packet.

Finally apply Definition 699.95 and Definition 699.98. Together they confine every $\mathsf{Real}^{\uparrow}$- or $\mathsf{Completion}^{\uparrow}$-facing request to the displayed carrier rows and the structural hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, leaving quotient streams, selected limits, host convergence predicates, and ambient completeness outside the target boundary. □

Theorem 699.100 (Real tail-agreement seal root window-budget determinacy)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. The displayed finite window row* W *determines the budgeted dyadic agreement route through* D *and* A*: every root-facing read that has reached* W *through the admitted observation limiter, regular tail-budget, and shared-threshold surface can reach the agreement row only along*

$$W \to D \to A.$$

There is no second dyadic readback row, no hidden agreement coordinate, and no selected tail outside the displayed packet route.

Proof. Project the carrier Q. The only paired window coordinate is W, the only dyadic readback coordinate attached to it is D, and the only agreement coordinate downstream of that readback is A.

Use root budget-selector admission and root budget shared-threshold factorization. They put the root read behind the admitted limiter, tail-budget, selector-budget, and shared-threshold rows before W can be consumed.

Finally apply dyadic readback route totality. Once W is displayed, D is the unique dyadic readback supplied to the local agreement edge, and A is reached only through that readback. Since the remaining rows H, C, P, N provide transport, replay, provenance, and naming, they add no alternate dyadic or agreement route. □

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root window budget determinacy`

Theorem 699.101 (Real tail-agreement seal root criterion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *criterion handoff consuming an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$ *factors through the sibling routes* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, *and* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *before it reaches the local route* $W \to D \to A$*. The handoff is therefore a finite BEDC route through budget, limiter, meet, fusion, paired window, dyadic readback, agreement, transport, continuation, provenance, and local naming rows.*

Proof. Begin with the root budget shared-threshold factorization. It records the observation limiter, regular tail-budget, and tail-meet route as the root-facing entrance to the local paired-window surface.

Attach the tail-fusion consumer through the same shared threshold. The fusion row consumes the already displayed tail-meet and budget rows, so it cannot introduce a second tail source or a separate threshold for the Real or Completion criterion.

Now use root window-budget determinacy. The local criterion handoff continues through W, then D, then A, with H, C, P, N supplying only componentwise transport, displayed replay, provenance, and naming. Thus the criterion handoff factors through exactly the named sibling routes and the local finite packet rows. □

Theorem 699.102 (Real tail-agreement seal root selector-budget compatibility)**.** *Let* $Q = (R, S, W, D, A, H, C, P,$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *or* $\mathsf{RegularCauchySelectorBud}$ *consumer that attempts to issue a selected-budget read or a tail-agreement read from* Q *must first pass through the finite selector surface of chapter 782 or chapter 678, the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *limiter of chapter 903, the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *budget of chapter 906, the shared tail-meet surface of chapter 560, the tail-fusion surface of chapter 658, and the criterion handoff of Definition 699.101. Only after that route is displayed may the consumer read the local chain*

$$W \to D \to A.$$

The compatibility row exports no selected limit, quotient stream equality, host real equality, trichotomy row, independent tail oracle, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.RealTailAgreementSealCarrier root selector budget compatibility`

Proof. Start with the root criterion handoff. It already places every Real or Completion criterion consumer behind the budget, limiter, tail-meet, and tail-fusion surfaces before the local packet

route is available, with the selector-budget sibling supplied by chapter 678 and the finite selector sibling supplied by chapter 782.

Now apply root window-budget determinacy. Once the selector consumer has reached the accepted packet, the only exposed agreement route is W, then D, then A; the transport, continuation, provenance, and local naming rows H, C, P, N do not add another selector or agreement coordinate.

Finally use root public consumer non-escape. The public boundary of Q is exhausted by the displayed rows and the named sibling surfaces, so a selected-budget or tail-agreement read cannot be reinterpreted as a selected limit, quotient stream equality, host real equality, trichotomy row, independent tail oracle, or ambient completeness principle. □

Theorem 699.103 (Real tail-agreement seal window-budget selector fusion lock)**.** *Let* $Q = (R, S, W, D, A, H, C, P, N)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that passes through selector-budget admission, shared-threshold reading, and tail-fusion consumption traverses the same finite route*

$$\mathsf{ObservationBudgetLimiter}^{\uparrow} \to \mathsf{RegularCauchyTailBudget}^{\uparrow} \to \mathsf{RegularCauchyTailMeet}^{\uparrow} \to \mathsf{RegularCauchySelectorBudg}$$

The route admits no quotient stream, selected limit, host convergence row, choice-selected tail, hidden completion selector, or second agreement surface between the selector admission and the agreement row A*.*

Proof. First apply Definition 699.102. It places every selected-budget or tail-agreement read behind the finite selector surface, the observation limiter, the tail budget, the shared tail meet, the tail-fusion surface, and the local agreement chain.

Next use Definition 699.101. The Real and Completion criterion consumers receive the same sibling handoff before the local route is available, so selector admission, shared-threshold reading, and fusion consumption are not independent entrances.

Now invoke Definition 699.100. Once the shared route reaches the accepted packet, the only local agreement route is the displayed window, dyadic readback, and agreement chain.

Finally apply Definition 699.104. The public consumer boundary is exhausted by the finite sibling surfaces and the displayed packet rows, so none of the excluded rows can be inserted between selector admission and the agreement coordinate. □

Theorem 699.104 (Real tail-agreement seal root public consumer non-escape)**.** *No public* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet* $Q = (R, S, W, D, A, H, C, P, N)$ *can bypass the finite route*

$$W \to D \to A$$

to obtain selected tails, host convergence, quotient stream equality, or ambient completeness. All such consumers receive only the displayed rows $R, S, W, D, A, H, C, P, N$ *and the named budget, limiter, meet, fusion, Cauchy-limit, Real, and Completion sibling surfaces.*

Proof. Start from the root criterion handoff. It forces every Real or Completion criterion read through the budget, limiter, meet, and fusion surfaces before the local paired-window route is public.

Use root window-budget determinacy to identify the local route. The packet exposes A only after W and D have been read, so a consumer cannot jump directly to an agreement row or replace the readback by a quotient stream equality.

Finally apply root budget Completion-consumer exhaustion. The listed sibling surfaces and the displayed carrier rows exhaust the public consumer boundary. Since no row stores a selected tail, host convergence predicate, quotient equality, or ambient completeness principle, no public consumer can bypass $W \to D \to A$ to recover one. □

Theorem 699.105 (Real tail-agreement seal L10 four-object exit route)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of Q that passes through the root selector-budget compatibility row and the root public non-escape row factors through the Phase Real and Completion four-object exit route:*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The route begins with the finite selector, limiter, tail budget, meet, fusion, and local $W \to D \to A$ rows of Definition 699.102 and Definition 699.103, and it is read as the four-object exit packet of Definition 126.500. Its only support rows are

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

No quotient stream equality, host sequence equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row is exported by this RealTailAgreementSeal exit route.*

Proof. First apply Definition 699.102. It places the consumer behind the finite selector surface, observation limiter, regular tail budget, shared meet, fusion surface, and the local $W \to D \to A$ chain before any Real or Completion read can consume the accepted packet.

Next use Definition 699.103. The lock identifies the same finite route for $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers, so the selector-budget read and the tail-agreement read cannot split into separate exit routes after fusion.

Now invoke Definition 126.500. The finite route is read as the displayed StreamName window, DyadicRatCore readback, RegSeqRat regular source, and RealUp seal, with only BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally apply Definition 699.104. The public boundary excludes quotient stream equality, host sequence equality, selected limits, ambient Real completeness, metric-completeness data, countable choice, and standard bridge rows. Hence the accepted packet supplies exactly the four-object exit route named above, and no public Real or Completion consumer escapes it. □

Theorem 699.106 (Real tail-agreement seal four-object exit non-escape)**.** *Let $Q = (R, S, W, D, A, H, C, P, N)$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Any accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer that reaches the four-object exit route factors through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the local chain

$$W \to D \to A.$$

The consumer can read only the displayed selector, limiter, tail budget, meet, fusion, window, dyadic readback, agreement, transport, continuation, provenance, and name rows supporting that route. It cannot read quotient stream equality, host sequence equality, selected limits, ambient completeness, countable-choice schedules, metric-completeness data, or bridge rows.

Proof. First apply root selector-budget compatibility. It places every selected-budget or tail-agreement read behind the finite selector surface, the observation limiter, the regular tail budget, the shared tail-meet row, the tail-fusion surface, and the local $W \to D \to A$ chain.

Next use the window-budget selector fusion lock. The lock makes the Real and Completion consumers traverse the same finite selector, limiter, budget, meet, fusion, StreamName, DyadicRatCore, RegSeqRat, and agreement route, so no hidden selector or second agreement surface appears between the selector admission and A.

Now invoke the L10 four-object exit route. That theorem identifies the visible exit packet as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, supported only by BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply root public consumer non-escape. Since the public boundary is exhausted by the displayed sibling surfaces and the local packet rows, quotient stream equality, host sequence equality, selected limits, ambient completeness, countable-choice schedules, metric-completeness data, and bridge rows remain outside the accepted consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailAgreementSeal}^{\uparrow}$ packet is generated from paired RegSeqRat source rows, finite paired StreamName tail windows, paired DyadicRatCore readbacks, a displayed tail-agreement row, componentwise hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.RealTailAgreementSealUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 699.1, Definition 699.2, Definition 699.3, Definition 699.4, Definition 699.5, Definition 699.40, Definition 699.41, Definition 699.42, Definition 699.43, Definition 699.44, Definition 699.45, Definition 699.46, Definition 699.47, Definition 699.48, Definition 699.49, Definition 699.50, Definition 699.51, Definition 699.52, Definition 699.56, Definition 699.57, Definition 699.58, Definition 699.59, Definition 699.60, Definition 699.61, and Definition 699.62 bind the finite paired tail-window agreement packet, classifier reflexivity, classifier symmetry, classifier transitivity, source separation, left–right threshold synchrony, paired-window route, dyadic readback, Real–Completion handoff, terminal consumer factorization, Completion ledger, completion budget-threshold exhaustion, dual-source window exhaustion, shared Real–Completion boundary, selector-budget handoff, finite approximation consumer boundary, Real–Completion selector chain, and TasteGate independence boundary used by Real and Completion handoff consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealTailAgreementSeal}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealTailAgreementSealUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected real values, host metric convergence, open-cover compactness, choice-selected tails, or a public standard bridge.

Upgrade path: The paper closure axis is complete for the finite paired tail-agreement packet; Lean verification and bridge checking remain independent axes.

Chapter 700

A Concrete Naming Certificate for $\mathsf{UnaryDirectionRouter}^{\uparrow}$

The $\mathsf{UnaryDirectionRouter}^{\uparrow}$ packet is the consumer-facing surface of the unary-direction bridge. It records which standard-boundary read is being requested, routes that read through the displayed unary and axis rows, and a local naming certificate, while keeping the $\mathsf{Nat}^{\uparrow}$ unary carrier and the $\mathsf{AxisNat}^{\uparrow}$ zero-spine carrier distinct inside the finite kernel.

The router consumes the $\mathsf{AxisNat}^{\uparrow}$ zero-spine carrier through chapter 276 and the unary-zero-spine standard comparison through chapter 572 before the selector row can route forward, backward, or neutral reads.

Definition 700.1 (Unary-direction router carrier)**.** *A* $\mathsf{UnaryDirectionRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(U, A, S, L, F, R, H, C, P, N)$$

with the following visible rows:

1. *U is the* $\mathsf{Nat}^{\uparrow}$ *unary row, read as a displayed* Unary *history and its local naming surface;*
2. *A is the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine row, read as a displayed* $\mathsf{ZeroSpine}$ *history and its local naming surface;*
3. *S is a finite selector choosing the forward presentation, the backward presentation, or the router-neutral presentation;*
4. *L is the displayed length ledger used by either directional read;*
5. *F is the forward* Cont *route from the unary row through L to the axis row;*
6. *R is the backward* Cont *route from the axis row through L to the unary row;*
7. *H stores componentwise* hsame *transports for the two source rows, selector row, length ledger, routes, provenance row, and local name;*
8. *C stores the route ledger exposing unary inspection, axis inspection, selector readback, length readback, forward readback, backward readback, neutral readback, and NameCert audit;*
9. *P records* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$, $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionRouter}^{\uparrow}}$ *row.*

The selector row is total over the three displayed presentations:

$$S \in \{\mathsf{forward}, \mathsf{backward}, \mathsf{neutral}\}.$$

The packet has no coordinate identifying the $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{AxisNat}^{\uparrow}$ *carriers, no host recursor over standard natural numbers, and no ambient function replacing the displayed* Cont *routes.*

Theorem 700.2 (Unary-direction router NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionRouter}^{\uparrow}}$ *surface for Definition 700.1 consists exactly of the following five obligation rows:*

1. *carrier admission, stating that an accepted router exposes the finite packet* $(U, A, S, L, F, R, H, C, P, N)$ *over the named BEDC dependencies;*
2. *selector totality, stating that every accepted consumer request chooses the forward, backward, or router-neutral presentation through the displayed selector row* S*;*
3. *route-ledger exactness, stating that forward and backward reads are precisely the displayed routes* F *and* R *through* C *and* L*;*
4. *nonidentification preservation, stating that componentwise transport through* H *never turns the* $\mathsf{Nat}^{\uparrow}$ *unary row and the* $\mathsf{AxisNat}^{\uparrow}$ *zero-spine row into a shared kernel carrier;*
5. *consumer non-escape, stating that downstream unary-direction consumers may read only through* $U, A, S, L, F, R, H, C, P, N$ *and cannot export a separate standard-model function as a BEDC route.*

Proof. Carrier admission is projection from the displayed packet. The source rows U and A, the selector row S, the length ledger L, the two route rows F, R, the transport row H, the route ledger C, the package row P, and the local name N are all present before the certificate accepts the router.

Selector totality is finite case analysis on S. In the forward case the consumer reads F through C and L; in the backward case it reads R through the same displayed ledgers; in the neutral case it receives only the common router surface and no directional carrier equality.

Route-ledger exactness follows because the packet contains exactly two directional Cont rows. Any forward or backward bridge read must pass through C after the selector and length rows have been exposed.

For nonidentification preservation, transport is componentwise through H. It may rename the unary row, the axis row, the selector, the ledger, and the routes inside their own displayed classifiers, but no row of the packet has target type asserting equality of the two carrier predicates.

Consumer non-escape is the remaining boundary condition. The provenance row P lists the permitted dependencies, and the local naming row N binds the consumer-facing route to the displayed packet, so a downstream consumer cannot replace the router by an external standard-model function or by a host-side identification of the two carriers. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryDirectionRouter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UnaryDirectionRouter}^{\uparrow}$ packet is generated from finite BHist rows for the $\mathsf{Nat}^{\uparrow}$ unary carrier, the $\mathsf{AxisNat}^{\uparrow}$ zero-spine carrier, a three-way selector, length ledger, forward and backward Cont routes, componentwise transports, provenance, and local naming. Directional reads are produced by replaying the selected route through the length ledger, not by identifying the two carriers or importing a standard natural-number function. hsame transports each row inside its own classifier, and Pkg records the unary, axis, bridge, standard-iso, and normalizer dependencies. The router therefore exposes direction without collapsing unary and axis presentations.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnaryDirectionRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the unary-direction router carrier, selector totality, route-ledger exactness, non-identification preservation, and consumer non-escape obligations.

Formal verification: FormalStatus(NameCert$_{\mathsf{UnaryDirectionRouter}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: It does not claim equality of $\mathsf{Nat}^{\uparrow}$ and $\mathsf{AxisNat}^{\uparrow}$ carriers, a host recursor over standard naturals, or an ambient function replacing the displayed routes.
Upgrade path: Obligation closure requires Lean encodings of the router carrier, selector cases, route exactness, and nonidentification preservation.

Chapter 701

A Concrete Naming Certificate for $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$

The $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$ packet realizes the stationary rational embedding classifier correspondence as finite NameCert data. It binds two RatUp seed rows, their constant StreamName rows, their stationary RegSeqRat diagonal rows, their RealUp constant rational embedding seals, the RatUp and RealUp classifier faces, and the transport routes that identify the two readings. The packet is local to stationary rational diagonals: it does not construct quotient streams, host real equality, selected limits, or ambient completeness.

The dependency route is the Real-and-Completion lattice rather than an isolated classifier packet. The carrier consumes stationary rational Cauchy inputs through chapter 364, the rational constant route through chapter 1491, the dyadic constant-name route through chapter 467, and the RealUp constant rational embedding seal through chapter 38 before comparing the RatUp and RealUp classifier faces.

Definition 701.1 (Stationary-embedding-classifier carrier)**.** *A* $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, E_0, S, T, G, H, R, Q, W, C, P, N)$$

with the following displayed rows:

1. *D and E_0 are the two accepted RatUp seed rows;*
2. *S and T are the constant* $\mathsf{StreamName}^{\uparrow}$ *rows* $\mathsf{RatConstStream}_D$ *and* $\mathsf{RatConstStream}_{E_0}$*;*
3. *G records the stationary* RegSeqRat *diagonal rows L_D and L_{E_0} obtained from the corresponding stationary rational Cauchy inputs;*
4. *H records the RealUp constant rational embedding seals* $(\mathsf{RatConstStream}_D, \mathsf{E1}(D))$ *and* $(\mathsf{RatConstStream}_{E_0}, \mathsf{E1}(E$
5. *R is the displayed RatUp classifier face* $\mathsf{RatHistoryClassifier}(D, E_0)$*;*
6. *Q is the displayed RealUp classifier face* $\mathsf{RealClassifier}(L_D, L_{E_0})$*;*
7. *W records componentwise* hsame *transports for the seed, stream, diagonal, seal, classifier, package, and local naming rows;*
8. *C records the displayed* Cont *routes that read the RatUp seed face, the constant StreamName face, the stationary diagonal face, and the RealUp embedding face through the same finite packet;*

9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, RegSeqRat, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryEmbeddingClassifier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $D, E_0, S, T, G, H, R, Q, W, C, P, N$.

Theorem 701.2 (StationaryEmbeddingClassifier NameCert obligations)**.** *For the carrier of Definition 701.1, the local* $\mathsf{NameCert}_{\mathsf{StationaryEmbeddingClassifier}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure for $E = (D, E_0, S, T, G, H, R, Q, W, C, P, N)$;*
2. *stationary-source admission, requiring D, E_0 to be RatUp seed rows and S, T to be their constant* $\mathsf{StreamName}^{\uparrow}$ *rows;*
3. *diagonal-and-seal admission, requiring G to expose the stationary* RegSeqRat *diagonals and H to expose the matching RealUp constant rational embedding seals;*
4. *classifier exactness, requiring the RealUp face Q and the RatUp face R to be transported through the same displayed* hsame *row W and route row C;*
5. *non-escape, requiring the packet to export no quotient stream, no host real equality, no selected limit, no ambient completeness principle, and no hidden classifier oracle.*

Proof. Carrier exposure is projection from the finite tuple in Definition 701.1. The seed rows, constant stream rows, stationary diagonal rows, embedding seals, classifier faces, transport rows, route rows, provenance row, and local naming row are all displayed coordinates.

Stationary-source admission reads the two RatUp rows D, E_0 and then the constant StreamName rows S, T. These rows are the same constant-stream witnesses used by the stationary rational Cauchy packet and by the constant StreamName readback surface.

Diagonal-and-seal admission reads G and H. The row G names the stationary diagonal representatives of the two regular Cauchy inputs, while H names the constant rational RealUp embeddings sealed by the corresponding e-one rows.

Classifier exactness is the finite transport obligation. The RatUp face R is read through the constant StreamName point-exactness route, then through the stationary diagonal route, and then through the RealUp constant rational embedding seal. Conversely the RealUp face Q is read back through the same displayed constant-stream and e-one seal rows to the RatUp seed face.

Finally inspect the carrier list. There is no coordinate for a quotient stream, host real equality, selected limit, ambient completeness principle, or hidden classifier oracle, so none of those objects can be exported by the local naming surface. □

Theorem 701.3 (StationaryEmbeddingClassifier correspondence)**.** *Let $E = (D, E_0, S, T, G, H, R, Q, W, C, P, N)$ be an accepted* $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$ *carrier. The public classifier reading factors through the finite equivalence*

$$\mathsf{RealClassifier}(L_D, L_{E_0}) \iff \mathsf{RatHistoryClassifier}(D, E_0).$$

Both directions use only the displayed RatUp seed rows, constant $\mathsf{StreamName}^{\uparrow}$ *rows, stationary* RegSeqRat *diagonal rows, RealUp constant rational embedding seals, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and the local* NameCert *row.*

Proof. Project the accepted carrier to the displayed rows. The RatUp side is the face $R = \mathsf{RatHistoryClassifier}(D, E_0)$, and the RealUp side is the face $Q = \mathsf{RealClassifier}(L_D, L_{E_0})$.

For the forward direction, read Q through the two stationary diagonal rows in G. The stationary rational diagonal certificate identifies each L_D and L_{E_0} with the corresponding constant rational RealUp embedding in H.

The Real constant rational embedding readback then exposes the constant StreamName rows S, T. The constant StreamName point-exactness surface reads those rows back as the same RatUp seed classifier R.

For the reverse direction, start from R. The constant StreamName witness transports the seed classifier to the constant streams S, T, and the stationary rational diagonal route transports those streams to the diagonal representatives in G.

The RealUp constant rational embedding seals in H identify those representatives with the RealUp face Q. The same W, C, P, N rows carry both readings, so the two implications form the displayed finite equivalence. □

Theorem 701.4 (StationaryEmbeddingClassifier dyadic rational factorization)**.** *Let $E = (D, E_0, S, T, G, H, R, Q, W$ be an accepted* $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$ *carrier. Suppose the RatUp seed rows D and E_0 admit displayed dyadic observation rows d_D and d_{E_0}, and suppose those rows are read through accepted constant-name packets whose public routes are the* $\mathsf{RationalEmbedding}^{\uparrow}$ *and* $\mathsf{DyadicEmbedding}^{\uparrow}$ *routes. Then the public equivalence*

$$\mathsf{RealClassifier}(L_D, L_{E_0}) \iff \mathsf{RatHistoryClassifier}(D, E_0)$$

factors through the ordered L10 route

$$\mathsf{RationalEmbedding}^{\uparrow} \to \mathsf{DyadicEmbedding}^{\uparrow} \to \mathsf{StationaryRationalCauchy}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

before the Real classifier face is compared with the RatUp classifier face. The factorization exports only the displayed RatUp seed rows, dyadic observation rows, constant $\mathsf{StreamName}^{\uparrow}$ *rows, stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, RealUp seal rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream, selected limit, host real equality, trichotomy, or ambient completeness coordinate.*

Proof. Project the accepted carrier with Definition 701.1. The rows $D, E_0, S, T, G, H, R, Q, W, C, P, N$ expose the two RatUp seeds, their constant stream schedules, the stationary diagonal rows, the RealUp seals, the two classifier faces, and the visible transport and route data.

Use chapter 1491 to read each RatUp seed through the rational constant route. This supplies the constant $\mathsf{StreamName}^{\uparrow}$ schedule, the stationary $\mathsf{RegSeqRat}^{\uparrow}$ readback, the dyadic observation ledger, and the RealUp seal boundary for the same finite source row.

Use chapter 467 on the displayed dyadic observation rows d_D and d_{E_0}. Its constant stream readback identifies the dyadic row, constant schedule, stationary readback, and RealUp seal as one constant dyadic real-name route, so the dyadic observation rows cannot be replaced by quotient streams or selected host limits.

Compose these two constant-name routes with the stationary rational Cauchy handoff of chapter 364. The composed route reaches the same RealUp seal rows named in H before any classifier comparison is exported.

Finally apply Definition 701.3. The correspondence reads the Real classifier face $Q = \mathsf{RealClassifier}(L_D, L_{E_0})$ and the RatUp classifier face $R = \mathsf{RatHistoryClassifier}(D, E_0)$ through the same displayed rows, so the public equivalence factors through the ordered L10 route above. The non-escape clauses in the cited constant-name and classifier surfaces exclude quotient streams, selected limits, host real equality, trichotomy, and ambient completeness. □

Theorem 701.5 (StationaryEmbeddingClassifier Real completion link)**.** *Every accepted* StationaryEmbeddingClassi *carrier factors its public classifier correspondence through chapter 364 and chapter 38 before exporting*

$$\mathsf{RealClassifier}(L_D, L_{E_0}) \iff \mathsf{RatHistoryClassifier}(D, E_0).$$

The stationary rational Cauchy surface supplies the constant $\mathsf{StreamName}^{\uparrow}$ *and* RegSeqRat *diagonal rows, while the RealUp surface supplies the constant rational embedding seals read before the Real classifier face is compared with the RatUp classifier face.*

Proof. Project the accepted carrier using Definition 701.1. The rows $D, E_0, S, T, G, H, R, Q, W, C, P, N$ expose the RatUp seeds, constant StreamName rows, stationary diagonal rows, RealUp seals, classifier faces, transport, routes, provenance, and local name.

Apply the NameCert obligations. They order the read through stationary-source admission and diagonal-and-seal admission before classifier exactness, so the route passes through the stationary rational Cauchy and RealUp sibling surfaces before the public equivalence is exported.

Use the classifier correspondence theorem to obtain the displayed equivalence. The non-escape theorem then rules out any quotient stream, host real equality, selected limit, ambient completeness principle, or hidden classifier oracle outside those sibling routes. □

Theorem 701.6 (StationaryEmbeddingClassifier non-escape boundary)**.** *An accepted* StationaryEmbeddingClassifie *carrier exports only RatUp seed rows, constant* $\mathsf{StreamName}^{\uparrow}$ *rows, stationary* RegSeqRat *diagonal rows, RealUp constant rational embedding seals, the RatUp and RealUp classifier faces, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It exports no quotient stream, no host real equality, no selected limit, no ambient completeness principle, and no hidden classifier oracle.*

Proof. Inspect the carrier coordinates of Definition 701.1. Every exported row is one of $D, E_0, S, T, G, H, R, Q, W, C$

The positive data remain finite and local: two seed rows, two constant stream rows, two stationary diagonal rows, two embedding seals, two classifier faces, and the rows that transport, route, package, and name them.

No listed coordinate is a quotient stream, host real equality, selected limit, ambient completeness principle, or hidden classifier oracle. Therefore an accepted carrier cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryEmbeddingClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{StationaryEmbeddingClassifier}^{\uparrow}$ packet is generated from RatUp seed rows, constant StreamName rows, stationary RegSeqRat diagonal rows, RealUp constant rational embedding seals, RatUp and RealUp classifier faces, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StationaryEmbeddingClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 701.1, Definition 701.2, Definition 701.3, Definition 701.4, and Definition 701.6 bind the finite stationary rational embedding classifier packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StationaryEmbeddingClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream identity, host real equality, selected limits, ambient completeness, bridge status, Lean verification, or a hidden classifier oracle.

Upgrade path: Obligation closure requires checked carrier admission, classifier correspondence, transport exactness, and non-escape for the same stationary rational embedding classifier packet.

Chapter 702

A Concrete Naming Certificate for $\mathsf{RegularCauchyMerge}^{\uparrow}$

The $\mathsf{RegularCauchyMerge}^{\uparrow}$ packet records the finite two-route merge used when a diagonal Real-completion handoff compares two regular rational sources through shared cofinal tail data. It binds two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, two selector schedules, one shared cofinal tail window, two $\mathsf{Real}^{\uparrow}$ seal routes, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local $\mathsf{NameCert}$ row as one inspectable BEDC carrier. The packet is a naming certificate for finite selector agreement; it is not a quotient of streams, a chosen host limit, an ambient completeness principle, or equality of completed reals.

Definition 702.1 (RegularCauchyMerge carrier)**.** *A* $\mathsf{RegularCauchyMerge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows being compared by the diagonal-limit handoff;*
2. S_0 *and* S_1 *are the two selector schedules that name which cofinal reads of* R_0 *and* R_1 *are opened;*
3. W *is the shared cofinal tail window, required to cover the two selected schedules at the requested precision;*
4. E_0 *and* E_1 *are the two* $\mathsf{Real}^{\uparrow}$ *seal routes reached from the displayed source rows through the shared window;*
5. H *records componentwise* hsame *transports for* $R_0, R_1, S_0, S_1, W, E_0, E_1$ *and the structural rows of the packet;*
6. C *records the displayed* Cont *routes from each source row through its selector schedule, into* W*, and from* W *into the corresponding* $\mathsf{Real}^{\uparrow}$ *seal route;*
7. P *is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMerge}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N$*.*

Theorem 702.2 (RegularCauchyMerge NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMerge}^{\uparrow}}$ *surface for Definition 702.1 consists exactly of carrier admission for* $M = (R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N)$*,*

source exposure for the two $\mathsf{RegSeqRat}^{\uparrow}$ *rows, selector exposure for the two schedules, shared-window coverage through* W*, two-route* $\mathsf{Real}^{\uparrow}$ *seal delivery through* E_0 *and* E_1*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, local naming through* N*, and non-escape from quotient streams, host-selected limits, countable choice, ambient completeness, hidden tail oracles, and host real equality.*

Proof. Carrier admission is projection from the displayed packet. The rows $R_0, R_1, S_0, S_1, W, E_0, E_1$ expose the two regular rational sources, their selector schedules, the shared cofinal window, and the two real-facing seal routes.

The selector and window obligations are read in order. Each selector schedule is tied to its displayed source row, and both schedules must pass through the single shared window W before either seal route is available.

The transport, route, provenance, and naming obligations are exactly the rows H, C, P, N. Since no further analytic coordinate occurs in M, the packet has no channel for quotient streams, host-selected limits, countable choice, ambient completeness, hidden tail oracles, or host real equality. □

Theorem 702.3 (RegularCauchyMerge source coherence)**.** *For every accepted* $\mathsf{RegularCauchyMerge}^{\uparrow}$ *carrier*

$$M = (R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N),$$

the two source rows R_0 *and* R_1 *are coherent only through the displayed selector schedules and the shared cofinal window. A consumer cannot compare the sources by an unrecorded stream quotient, a host real equality test, a selected ambient limit, or an external Cauchy-completion principle.*

Proof. Project R_0, R_1, S_0, S_1, W from the carrier. These are the only rows in the packet that can mention the two regular rational sources and their cofinal reads.

The NameCert obligations force the two selector schedules to enter the same shared tail window before the seal routes are reached. Thus any source comparison is mediated by S_0, S_1, W and by the continuation rows that replay those displayed choices.

The structural rows H, C, P, N preserve and name the same finite data. Because the carrier has no coordinate for stream quotients, host real equality, selected ambient limits, or a completion theorem, source coherence is exactly the visible two-schedule route through W. □

Theorem 702.4 (RegularCauchyMerge diagonal-selector compatibility)**.** *Every diagonal-limit consumer of an accepted* $\mathsf{RegularCauchyMerge}^{\uparrow}$ *carrier factors its selector agreement through*

$$R_0, R_1, S_0, S_1, W, H, C, P, N.$$

The first route reads R_0 *through* S_0 *into* W*, and the second route reads* R_1 *through* S_1 *into the same* W*. The agreement is therefore a finite cofinal-window agreement between two displayed selector schedules, not a choice of a subsequence outside the packet.*

Proof. Start with source coherence. It identifies S_0, S_1, W as the only comparison surface available to a diagonal-limit consumer.

The first selector route reads R_0 through S_0 and then enters W. The second selector route reads R_1 through S_1 and then enters the same W. The continuation row C records these two routes as displayed replay data.

The rows H, P, N transport, package, and name the same two-route surface. No row in the carrier supplies an unbounded subsequence selector, so diagonal-selector compatibility is confined to the finite route through the shared window. □

Theorem 702.5 (RegularCauchyMerge Real seal non-escape)**.** *For every accepted* $\mathsf{RegularCauchyMerge}^{\uparrow}$ *carrier, the two* $\mathsf{Real}^{\uparrow}$ *seal routes* E_0 *and* E_1 *are reachable only after the source rows, selector schedules, shared cofinal window, transports, continuations, provenance, and local naming row have been displayed. In particular, a* $\mathsf{Real}^{\uparrow}$ *consumer of either route receives no quotient stream, hidden completed value, selected host limit, countable-choice selector, host real equality, or ambient completeness coordinate.*

Proof. Use the NameCert obligations to place E_0 and E_1 downstream of R_0, R_1, S_0, S_1, W. Thus neither real-facing route is available before the two sources have passed through their selector schedules and the shared cofinal tail window.

Apply diagonal-selector compatibility. It confines the agreement data needed by the two seal routes to the finite route $R_0, R_1, S_0, S_1, W, H, C, P, N$.

Finally inspect the carrier. The remaining rows are the two seal routes themselves, and no coordinate supplies quotient streams, hidden completed values, selected host limits, countable choice, host real equality, or ambient completeness. □

Theorem 702.6 (RegularCauchyMerge L10 sibling route)**.** *For every accepted* $\mathsf{RegularCauchyMerge}^{\uparrow}$ *packet*

$$M = (R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N),$$

every diagonal-completion consumer of the merge first reads the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows through chapter 301, then reads the shared finite-window and schedule surface through chapter 126, then reads the dyadic observation ledger through chapter 291, and only then reaches the two* $\mathsf{Real}^{\uparrow}$ *seal routes through chapter 38. This sibling route is a finite L10 dependency path for the displayed packet; it supplies no quotient stream, host-selected limit, countable choice, host real equality, or ambient completeness coordinate.*

Proof. Start with the local NameCert obligations. They expose R_0 and R_1 as the two regular rational source rows, require the selector schedules and shared window to be displayed, and place E_0 and E_1 downstream of that finite route.

Apply source coherence and diagonal-selector compatibility. The two source reads cannot be compared before S_0, S_1 and the shared cofinal window W have been opened, so the consumer is forced through the same finite-window and schedule surface used by the stream naming certificate.

The dyadic observation ledger is the next visible interface for the rational reads: it records the finite precision data that the shared window can expose before either real-facing seal is available. Thus the L10 route reads the two $\mathsf{RegSeqRat}^{\uparrow}$ rows, the stream-name schedule surface, and the dyadic ledger before it can inspect the $\mathsf{Real}^{\uparrow}$ endpoints.

Finally apply Real-seal non-escape. Since the two seal routes only receive the displayed rows $R_0, R_1, S_0, S_1, W, E_0, E_1, H, C, P, N$, the route has no coordinate for quotient streams, host-selected limits, countable choice, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMerge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyMerge}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two selector schedules, one shared cofinal tail window, two RealUp seal routes, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyMerge}^{\uparrow}}$, seed)

Scope closed: Definition 702.1, Definition 702.2, Definition 702.3, Definition 702.4, and Definition 702.5 bind the finite two-route cofinal selector agreement used by Real-completion diagonal handoffs.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyMerge}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host real equality, countable choice, host-selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, source exposure, selector exposure, shared-window coverage, two-route RealUp seal delivery, transport stability, route replay, provenance, local naming, and non-escape for the same finite packet.

Chapter 703

A Concrete Naming Certificate for $\mathsf{CauchyInverseBudget}^{\uparrow}$

The $\mathsf{CauchyInverseBudget}^{\uparrow}$ packet records the finite reciprocal-window budget used by Real and Completion consumers when a regular Cauchy name is displayed with a nonzero-apartness row. It opens only finite $\mathsf{StreamName}^{\uparrow}$ windows, reads dyadic endpoint and radius data through $\mathsf{DyadicRatCore}^{\uparrow}$, exposes the regular-Cauchy source through $\mathsf{RegSeqRat}^{\uparrow}$, and seals the reciprocal candidate through $\mathsf{Real}^{\uparrow}$. The packet is a local budget certificate; it is not a field structure, quotient real equality, total reciprocal operation, trichotomy principle, or ambient inverse theorem.

Definition 703.1 (CauchyInverseBudget carrier)**.** *A* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, A, M, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source whose reciprocal window is being read;*
2. *A is the nonzero-apartness budget row, recording the displayed finite lower bound that keeps the reciprocal denominator away from the zero boundary;*
3. *M is the finite modulus row selecting the observation precision required by the reciprocal read;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected from X at the requested budget;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and reciprocal arithmetic row read from that window;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D together with the budget rows A and M;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row through which completion-facing consumers read the reciprocal candidate;*
8. *H records componentwise* hsame *transports for X, A, M, W, D, R, S and for the package and local naming rows;*
9. *C records the displayed* Cont *routes for source inspection, apartness-budget exposure, finite-window coverage, dyadic reciprocal readback, regular-sequence handoff, and RealUp seal handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInverseBudget}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $X, A, M, W, D, R, S, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CauchyInverseBudgetUp.CauchyInverseBudgetTasteGate single carrier alignment`

Theorem 703.2 (CauchyInverseBudget carrier admission)**.** *Every accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet admits exactly the carrier rows*

$$X, A, M, W, D, R, S, H, C, P, N.$$

A reciprocal-window read first exposes the $\mathsf{RegSeqRat}^{\uparrow}$ *source* X*, the nonzero-apartness budget* A*, and the finite modulus row* M*; no host reciprocal operation or quotient real representative is a carrier coordinate.*

Proof. Project the carrier of Definition 703.1. The packet lists the regular-Cauchy source, apartness budget, modulus, selected window, dyadic readback, regular-sequence readback, real seal, transports, routes, provenance, and local naming row.

Since those rows exhaust the tuple, a consumer has no separate coordinate from which to read a field inverse, quotient representative, trichotomy decision, or ambient completeness fact. Admission is therefore finite projection from the displayed packet. □

Theorem 703.3 (CauchyInverseBudget apartness exposure)**.** *For an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier, every reciprocal-window route exposes the nonzero-apartness budget* A *before the dyadic reciprocal row* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *are consumed. A route that omits* A *remains a boundary attempt and cannot export the reciprocal candidate.*

Proof. Use carrier admission to display the order of source rows. The rows X, A, M are the only source-side data available before a finite window is selected.

The dyadic reciprocal read D depends on the apartness budget because the finite endpoint and radius must be read inside a nonzero window. Without A, the package provenance has no row certifying that the denominator stays away from the zero boundary.

Thus a route omitting A cannot pass to R or S. The local naming row records such a route as boundary evidence rather than as a reciprocal candidate. □

Theorem 703.4 (CauchyInverseBudget apartness transport)**.** *Let* $B = (X, A, M, W, D, R, S, H, C, P, N)$ *and* $B' = (X', A', M', W', D', R', S', H', C', P', N')$ *be accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packets. If the displayed rows* H *and* H' *relate the source, apartness, modulus, window, dyadic, readback, and seal coordinates componentwise by* hsame*, then any reciprocal-window route for* B *that exposes* A *before consuming* D *and* R *transports to a route for* B' *that exposes* A' *before consuming* D' *and* R'*. The transported route still cannot export a reciprocal candidate when the apartness row is omitted.*

Proof. Carrier admission fixes the row lists of both packets as $X, A, M, W, D, R, S, H, C, P, N$ and $X', A', M', W', D', R', S', H', C', P', N'$. Thus the only source-side rows available before the dyadic read are the displayed source, apartness, and modulus rows.

Apply Definition 703.3 to the route in B. It gives the ordering constraint that A must be exposed before D and R can be consumed, and it records the omitted-apartness case as boundary evidence rather than as a reciprocal export.

Use the componentwise hsame transports in H and H' to move the displayed rows X, A, M, W, D, R, S to $X', A', M', W', D', R', S'$. Because the transported route has the same carrier order, the image of the original exposure step is the exposure of A' before D' and R'.

The omitted-row exclusion is transported along the same displayed route. No extra coordinate is introduced by hsame, so a route without the apartness row in B' remains a boundary attempt and cannot export the reciprocal candidate through R' or S'. □

Theorem 703.5 (CauchyInverseBudget finite-window coverage)**.** *Every dyadic endpoint, radius, and reciprocal arithmetic read in D is covered by the finite* $\mathsf{StreamName}^{\uparrow}$ *window W selected from X by the modulus row M. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet must first consume the route*

$$X, A, M, W, D, H, C, P, N.$$

Proof. Read the modulus row M through the carrier. It selects the finite window W from the displayed regular-Cauchy source X.

The dyadic row D is then a read of that window, not a read of an unbounded stream. Its endpoint, radius, and reciprocal arithmetic rows are all routed through the same W coordinate and the same displayed continuation row C.

The support rows H, P, N preserve, package, and name that finite route. Therefore downstream consumers reach R and S only after the displayed source, budget, modulus, window, and dyadic rows have been consumed. □

Theorem 703.6 (CauchyInverseBudget StreamName window dependency)**.** *Every accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *reciprocal-window route consumes the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface of chapter 126 before the dyadic row D, the regular readback row R, or the* $\mathsf{Real}^{\uparrow}$ *seal row S can be read. The reciprocal budget therefore depends on* $\mathsf{StreamName}^{\uparrow}$ *as a sibling certificate, not on an unbounded stream or on a host reciprocal operation.*

Proof. Carrier admission displays W as the finite $\mathsf{StreamName}^{\uparrow}$ window selected from the regular-Cauchy source X by the modulus row M. This is the same finite window coordinate named in Definition 703.1, so the reciprocal packet has no second stream surface from which a consumer could read.

Finite-window coverage then routes every endpoint, radius, and reciprocal arithmetic read in D through that selected W. Since R is produced from the dyadic row and S seals the resulting readback, neither downstream row can be consumed before the finite StreamName window has been consumed.

Thus the dependency is horizontal: the reciprocal budget imports the finite window discipline already certified by chapter 126. It does not appeal to an unbounded stream, a quotient representative, or a host-level reciprocal operation. □

Theorem 703.7 (CauchyInverseBudget reciprocal readback route)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *reciprocal readback row R of an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier factors through the apartness budget A, finite modulus M, selected window W, and dyadic reciprocal row D. The readback exports no total inverse operation and no quotient-stream equality.*

Proof. First apply apartness exposure. The route to a reciprocal candidate must display A before any dyadic reciprocal row is available.

Next apply finite-window coverage. The dyadic row D is read from the selected finite window W under the modulus row M, so the readback route has only finite observation data.

Finally project R from the carrier and attach the structural rows H, C, P, N. They repack the same finite route as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff and add no total inverse operation or quotient equality. □

Theorem 703.8 (CauchyInverseBudget RealUp seal non-escape)**.** *Every completion-facing read of an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the seal row S after the reciprocal readback row R has been produced from X, A, M, W, D. The seal exports no field structure, quotient real equality, total reciprocal, trichotomy principle, or ambient inverse theorem.*

Proof. Use the reciprocal readback route to bind R to the source, apartness, modulus, window, and dyadic rows. The row R is therefore already a finite $\mathsf{RegSeqRat}^{\uparrow}$ handoff before the seal is read.

Project the seal coordinate S from the carrier. It appears only after R and is routed by the same Cont and Pkg rows, so the RealUp consumer receives the displayed finite packet.

The carrier contains no coordinate for a field structure, quotient real equality, total reciprocal operation, trichotomy principle, or ambient inverse theorem. Hence the seal cannot export any such object. □

Theorem 703.9 (CauchyInverseBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyInverseBudget}^{\uparrow}}$ *surface consists of carrier admission, apartness budget exposure, apartness-budget transport, finite-window coverage,* $\mathsf{StreamName}^{\uparrow}$ *window dependency, reciprocal readback routing,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and the local naming row for the same finite packet* $B = (X, A, M, W, D, R, S, H, C, P, N)$.

Proof. Carrier admission is Definition 703.2. Apartness budget exposure is Definition 703.3, apartness-budget transport is Definition 703.4, finite-window coverage is Definition 703.5, and the sibling window dependency is Definition 703.6.

The analytic-facing route is supplied by Definition 703.7 and Definition 703.8. Together they keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers inside X, A, M, W, D, R, S.

The structural rows H, C, P, N preserve, route, package, and name the same packet. Since all obligation clauses refer to the same displayed rows, they form one finite NameCert surface. □

Theorem 703.10 (CauchyInverseBudget transport stability)**.** *Let* $B = (X, A, M, W, D, R, S, H, C, P, N)$ *and* $B' = (X', A', M', W', D', R', S', H', C', P', N')$ *be accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packets. If the displayed row* H *relates the two packets componentwise by* hsame *on* $X, A, M, W, D, R, S, H, C, P, N$*, then the carrier admission row and the local classifier row are preserved from* B *to* B'*. In particular, the transported packet still has exactly the carrier coordinates* $X', A', M', W', D', R', S', H', C', P', N'$ *and no extra inverse, quotient equality, trichotomy, or ambient completeness coordinate.*

Proof. Project the carrier tuple of B by Definition 703.1. The row H supplies the componentwise hsame evidence for each displayed coordinate, including the package and local naming rows.

Transport the source-side coordinates X, A, M, W first. Carrier admission reads only those displayed coordinates before the dyadic row is available, so the accepted carrier shape is preserved under their hsame images.

Transport the downstream coordinates D, R, S and then the structural coordinates H, C, P, N. Since the classifier compares accepted carriers only through these displayed rows, the transported classifier row is the same finite comparison on B'.

No step introduces an unlisted coordinate. The transported carrier therefore remains the finite $\mathsf{CauchyInverseBudget}^{\uparrow}$ packet named by the displayed tuple. □

Theorem 703.11 (CauchyInverseBudget route stability)**.** *Under the componentwise* hsame *transport of Definition 703.10, the displayed* Cont *routes for source inspection, apartness-budget exposure, finite-window coverage, dyadic reciprocal readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal handoff remain routes for the transported packet. Thus every accepted route into* B' *still factors through the same finite row order* $X', A', M', W', D', R', S'$.

Proof. Begin with the transported carrier and classifier supplied by Definition 703.10. Source inspection reads the displayed X' row and then the budget and modulus rows A' and M'; these are exactly the transported images of the original source route.

Apartness-budget exposure is preserved by Definition 703.3. The route cannot reach a dyadic reciprocal row in the transported packet unless the transported apartness row A' has already been exposed.

Finite-window coverage is preserved by Definition 703.5. The dyadic endpoint, radius, and reciprocal arithmetic reads in D' are covered by the selected finite window W' under the transported modulus row M'.

Finally apply Definition 703.7 and Definition 703.8. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R' and the $\mathsf{Real}^{\uparrow}$ seal S' inside the transported finite route, so the Cont rows do not escape into a host reciprocal operation or quotient-real comparison. □

Theorem 703.12 (CauchyInverseBudget provenance and local naming)**.** *The* Pkg *provenance row and the local* $\mathsf{NameCert}_{\mathsf{CauchyInverseBudget}^{\uparrow}}$ *row of an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet name exactly the finite packet*

$$B = (X, A, M, W, D, R, S, H, C, P, N).$$

They add no field inverse, quotient real equality, trichotomy, or ambient completeness data beyond the carrier, classifier, transport, route, provenance, and local naming rows already displayed.

Proof. Use Definition 703.9 to identify the NameCert surface. Its clauses are carrier admission, apartness exposure, finite-window coverage, sibling window dependency, reciprocal readback routing, seal non-escape, transport, route, provenance, and local naming for the same displayed tuple.

The structural content of the transport row is fixed by Definition 703.10. The route content is fixed by Definition 703.11. These two rows do not name any object outside $X, A, M, W, D, R, S, H, C, P, N$.

The provenance row P therefore packages only the listed BEDC surfaces, and the local row N names that package as the $\mathsf{CauchyInverseBudget}^{\uparrow}$ certificate. Since no preceding row contains a field inverse, quotient real equality, trichotomy decision, or ambient completeness witness, neither P nor N can export one. □

Theorem 703.13 (CauchyInverseBudget root apartness-window obligation)**.** *Every reciprocal-root consumer that reads an accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet must expose the apartness row* A*, modulus row* M*, finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, and dyadic readback row* D *before any reciprocal candidate can be consumed. The root-facing route therefore starts from the carrier*

$$B = (X, A, M, W, D, R, S, H, C, P, N)$$

and admits no read of D*,* R*, or* S *that bypasses* A, M, W*.*

Lean4 ✓ `BEDC.Derived.CauchyInverseBudgetUp.CauchyInverseBudgetRouteSurface apartness window closure`

Proof. Begin with carrier admission for Definition 703.1. The only reciprocal-root packet available to a consumer is the displayed tuple $B = (X, A, M, W, D, R, S, H, C, P, N)$, so the candidate cannot be read from an unlisted coordinate.

Apply Definition 703.3. It places the nonzero-apartness budget A before the dyadic reciprocal row D and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Apply Definition 703.5 and Definition 703.6. These rows bind the modulus row M, finite window W, and dyadic row D to the finite $\mathsf{StreamName}^{\uparrow}$ observations that the reciprocal route is allowed to inspect.

Finally use Definition 703.7. Since the readback row R factors through A, M, W, D, no root-facing consumer can consume a reciprocal candidate before the apartness and window rows have been exposed. □

Theorem 703.14 (CauchyInverseBudget root RegSeqRat Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a reciprocal-root budget factors through the displayed rows* $X, A, M, W, D, R, S, H, C, P, N$ *of the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *carrier. Such a consumer receives the reciprocal readback row* R *and the Real seal row* S *only through that finite route, and receives no total inverse, quotient real equality, trichotomy row, ambient completeness row, or host reciprocal coordinate.*

Proof. Start from the root apartness-window obligation of Definition 703.13. It fixes the upstream route through A, M, W, D before the readback and seal rows can be used.

For the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, apply Definition 703.7. The readback row R is produced only from the source, apartness, modulus, finite window, and dyadic readback rows already displayed in the carrier.

For the $\mathsf{Real}^{\uparrow}$ handoff, apply Definition 703.8. The seal row S is reached after the readback row R has been produced from X, A, M, W, D.

The excluded data are absent from the carrier inventory and from both handoff theorems. Hence neither a regular-sequence consumer nor a Real consumer receives a total inverse, quotient equality, trichotomy decision, ambient completeness theorem, or host reciprocal coordinate. □

Theorem 703.15 (CauchyInverseBudget root formal surface)**.** *The Lean-facing reciprocal-root surface of* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *is exactly the displayed carrier* $B = (X, A, M, W, D, R, S, H, C, P, N)$ *together with the existing TasteGate alignment*

`BEDC.Derived.CauchyInverseBudgetUp.CauchyInverseBudgetTasteGate_single_carrier_alignment`.

It packages the apartness-window obligation, the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows, NameCert obligations, transport stability, route stability, provenance, and local naming, and it contains no host reciprocal coordinate.*

Proof. The carrier surface is fixed by Definition 703.1 and its Lean-facing marker. These identify the single displayed tuple as the formal packet that the chapter exposes.

Add the root rows supplied by Definition 703.13 and Definition 703.14. They bind apartness, modulus, finite window, dyadic readback, regular readback, and Real seal consumption to the same packet.

Use Definition 703.9, Definition 703.10, and Definition 703.11. These rows attach local naming, componentwise hsame transport, and displayed Cont route stability to the carrier without adding another coordinate.

Finally apply Definition 703.12. The provenance and local name rows account for the same finite packet, so the formal surface contains the TasteGate alignment and the listed BEDC rows, but no host reciprocal coordinate. □

Theorem 703.16 (CauchyInverseBudget obligation closure package)**.** *The carrier of Definition 703.1, apartness exposure of Definition 703.3, apartness transport of Definition 703.4, finite-window coverage of Definition 703.5, StreamName dependency of Definition 703.6, reciprocal readback route of Definition 703.7, RealUp seal non-escape of Definition 703.8, transport stability of Definition 703.10, route stability of Definition 703.11, and provenance and local naming of Definition 703.12 supply the full obligationClosure package for* $\mathsf{CauchyInverseBudget}^{\uparrow}$.

Proof. The carrier fixes the packet as $B = (X, A, M, W, D, R, S, H, C, P, N)$. Apartness exposure and apartness transport ensure that every reciprocal route displays the nonzero budget row before any dyadic or regular readback can be consumed, and that this ordering is stable under componentwise hsame.

Finite-window coverage and the StreamName dependency place the endpoint, radius, and reciprocal arithmetic rows inside a selected finite window. The reciprocal readback and RealUp seal non-escape theorems then route downstream consumers through D, R, and S without exporting a total inverse, quotient equality, trichotomy principle, or ambient completeness theorem.

Transport stability, route stability, provenance, and local naming attach H, C, P, N to the same finite packet. Thus the listed rows are exactly the obligationClosure package for the reciprocal-window budget. □

Theorem 703.17 (CauchyInverseBudget scoped consumer route package)**.** *The rows of Definition 703.16 determine a scoped consumer route for* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *over the concrete BEDC surface*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Every such route starts from the finite packet $B = (X, A, M, W, D, R, S, H, C, P, N)$*, consumes the apartness row* A*, finite modulus row* M*, selected* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *reciprocal readback row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *or* $\mathsf{Real}^{\uparrow}$ *seal* S*, and packages the route only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Start with carrier admission and the obligation package. They fix the only available packet as $B = (X, A, M, W, D, R, S, H, C, P, N)$ and identify the source, budget, window, readback, seal, transport, route, provenance, and local name rows.

Apply apartness exposure, finite-window coverage, and the StreamName dependency. These rows place A, M, W before any dyadic reciprocal arithmetic in D, and they make the window a concrete $\mathsf{StreamName}^{\uparrow}$ surface rather than an unbounded stream.

Use the reciprocal readback route and the Real seal non-escape theorem. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is reached through A, M, W, D, and the $\mathsf{Real}^{\uparrow}$ seal S is reached only after that handoff.

Finally attach transport stability, route stability, provenance, and local naming. Those rows are exactly the displayed BHist, hsame, Cont, Pkg, and NameCert support, so the consumer route is scoped by the listed BEDC surface and has no extra reciprocal coordinate. □

Theorem 703.18 (CauchyInverseBudget Dyadic RegSeq Real scope)**.** *For every accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and* $\mathsf{Real}^{\uparrow}$ *row* S *form a single scoped reciprocal-budget chain:*

$$X, A, M, W \longrightarrow D \longrightarrow R \longrightarrow S.$$

The chain is transported only by the componentwise hsame *rows in* H*, replayed only by the displayed* Cont *routes in* C*, and named only by the package provenance* P *and local* NameCert *row* N*.*

Proof. Read D through finite-window coverage. The dyadic endpoint, radius, and reciprocal arithmetic rows are obtained from the selected $\mathsf{StreamName}^{\uparrow}$ window under the displayed apartness and modulus rows.

Read R through the reciprocal readback route. This makes the $\mathsf{RegSeqRat}^{\uparrow}$ handoff a consumer of the same A, M, W, D rows rather than an independent regular sequence.

Read S through Real seal non-escape. The $\mathsf{Real}^{\uparrow}$ consumer receives only the seal row after the regular readback row has been produced.

Transport stability and route stability keep the whole chain inside H and C. Provenance and local naming then record the same finite chain in P and N, so the scoped chain has exactly the displayed dyadic, regular, and Real endpoints. □

Theorem 703.19 (CauchyInverseBudget consumer non-escape scope)**.** *Every scoped* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *consumer over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *reads the reciprocal candidate only through the finite packet* $B = (X, A, M, W, D, R, S, H, C, P, N)$. *Such a consumer cannot escape to a host total inverse, quotient real equality, trichotomy row, ambient completeness row, unbounded stream read, or external reciprocal coordinate.*

Proof. The scoped consumer route package first fixes the available route through $X, A, M, W, D, R, S, H, C, P, N$. Any consumer of the reciprocal candidate must therefore pass through the apartness, modulus, finite-window, dyadic, regular, and Real rows already displayed there.

The dyadic-regular-Real scope theorem then fixes the only downstream chain as $X, A, M, W \longrightarrow D \longrightarrow R \longrightarrow S$. Componentwise transport, continuation replay, provenance, and local naming do not add another data coordinate.

Apply Real seal non-escape and the root RegSeqRat Real handoff. These theorems exclude field structure, quotient equality, total reciprocal, trichotomy, ambient inverse data, and host readback from the handoff surface.

The remaining excluded unbounded-stream read is ruled out by finite-window coverage and the StreamName dependency. Thus every accepted consumer remains inside the scoped BEDC route and no escape coordinate is exported. □

Theorem 703.20 (CauchyInverseBudget bridge route)**.** *The public* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *route over* $B = (X, A, M, W, D, R, S, H, C, P, N)$ *bridges the concrete reciprocal-window carrier to the abstract reciprocal-budget interface by exposing exactly the apartness row* A, *finite modulus* M, *selected* $\mathsf{StreamName}^{\uparrow}$ *window* W, $\mathsf{DyadicRatCore}^{\uparrow}$ *reciprocal readback* D, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R, $\mathsf{Real}^{\uparrow}$ *seal* S, *componentwise transport* H, *displayed route* C, *provenance* P, *and local name* N. *It exports no total inverse, quotient real equality, trichotomy, ambient inverse theorem, or host reciprocal coordinate.*

Proof. Use Definition 703.17 to fix the public route over the packet $B = (X, A, M, W, D, R, S, H, C, P, N)$. This route starts from the concrete reciprocal-window carrier and exposes only the displayed BEDC surface.

Apply Definition 703.18. It binds the selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ reciprocal readback, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal into the single downstream chain $X, A, M, W \longrightarrow D \longrightarrow R \longrightarrow S$.

Now use Definition 703.19. The consumer route cannot escape to a total inverse, quotient real equality, trichotomy row, ambient inverse theorem, unbounded stream read, or external reciprocal coordinate.

The public route rows and structural support rows then give the bridge-facing certificate. The apartness, modulus, window, readback, handoff, seal, transport, route, provenance, and local name are exactly the rows exposed to the reciprocal-budget interface. □

Theorem 703.21 (CauchyInverseBudget finite consumer coverage)**.** *Every finite reciprocal-budget consumer that reads the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *bridge route is covered by the displayed carrier rows*

$$X, A, M, W, D, R, S, H, C, P, N,$$

with the apartness, modulus, window, and dyadic readback rows A, M, W, D *read before the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows* R, S. *Downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and Completion-facing consumers receive only this finite route. They receive no total inverse, quotient real equality, trichotomy, ambient inverse theorem, unbounded stream, or host reciprocal row.*

Proof. Begin with the root apartness and window obligations. The root apartness-window theorem supplies A, M, W as the finite budget through which any reciprocal row must be read, and the root RegSeqRat Real handoff places the later R, S rows behind that budget.

Next use the scoped consumer route package. It fixes the visible carrier as $B = (X, A, M, W, D, R, S, H, C, P, N)$ and orders the dyadic readback row D before the regular and Real consumer rows.

The dyadic RegSeq Real scope theorem then identifies the only downstream chain as

$$X, A, M, W \longrightarrow D \longrightarrow R \longrightarrow S.$$

Thus a Completion-facing consumer can only read the reciprocal through the same finite route.

Finally apply the consumer non-escape theorem and the bridge route theorem. Together they exclude total inverse data, quotient real equality, trichotomy, ambient inverse theorems, unbounded stream reads, and host reciprocal rows from every finite consumer covered by the bridge. □

Theorem 703.22 (CauchyInverseBudget terminal refusal classifier)**.** *For every accepted* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *packet* $B = (X, A, M, W, D, R, S, H, C, P, N)$*, every request for total inverse, quotient real equality, trichotomy, ambient inverse theorem, unbounded stream read, or host reciprocal coordinate is routed to the boundary/refusal side of the finite* NameCert *surface unless the request first factors through the displayed rows* $X, A, M, W, D, R, S, H, C, P, N$*. In particular, the only accepted reciprocal consumer is the finite chain certified by Definition 703.21.*

Proof. Start with carrier admission for the accepted packet. The local ledger names only the carrier coordinate X, the apartness row A, the modulus row M, the selected $\mathsf{StreamName}^{\uparrow}$ window W, the dyadic reciprocal readback D, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, the $\mathsf{Real}^{\uparrow}$ seal S, the BHist row H, the Cont row C, the Pkg row P, and the NameCert row N.

Apply the apartness exposure, the $\mathsf{StreamName}^{\uparrow}$ window dependency, and the reciprocal readback route. These place every admitted reciprocal access behind the finite A, M, W, D budget before any downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can read it.

Use the $\mathsf{Real}^{\uparrow}$ seal non-escape theorem, the scoped consumer non-escape theorem, and the bridge route theorem. A request for total inverse data, quotient real equality, trichotomy, an ambient inverse theorem, an unbounded stream read, or a host reciprocal coordinate has no row in the displayed packet unless it has already been transported through the named carrier chain.

Finally invoke Definition 703.21. The finite coverage theorem identifies the only accepted reciprocal consumer as the chain

$$X, A, M, W \longrightarrow D \longrightarrow R \longrightarrow S.$$

Every off-chain reciprocal request therefore lacks a carrier coordinate in the local naming ledger, so the finite NameCert surface classifies it as a boundary/refusal row rather than as an alternate consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInverseBudget}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyInverseBudget}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source, a nonzero-apartness budget, a finite modulus row, a selected $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ reciprocal readback, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.CauchyInverseBudgetUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInverseBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 703.17, Definition 703.18, and Definition 703.19 bind the carrier $B = (X, A, M, W, D, R, S, H, C, P, N)$, apartness exposure A, finite $\mathsf{StreamName}^{\uparrow}$ window W, $\mathsf{DyadicRatCore}^{\uparrow}$ reciprocal row D, $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, $\mathsf{Real}^{\uparrow}$ seal row S, BHist, hsame, Cont, Pkg, and NameCert rows into one scoped reciprocal-window consumer route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyInverseBudget}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.CauchyInverseBudgetUp.CauchyInverseBudgetTasteGate\_single\_carrier\_alignment`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close Lean verification, field structure, quotient real equality, total reciprocal operation, trichotomy, ambient inverse theorems, or a public standard bridge.
Upgrade path: There is no further paper-closure upgrade for the reciprocal-window consumer route; remaining work belongs to the independent formal and bridge axes.

Chapter 704

A Concrete Naming Certificate for $\mathsf{ContourAffineConcat}^{\uparrow}$

The $\mathsf{ContourAffineConcat}^{\uparrow}$ packet realizes the affine concatenation law for contour integrals as finite NameCert data. It records the two contour rows, two integrand rows, scalar rows, four input integral rows, the concatenated contour row, the affine output row, and the BEDC structural rows that certify the shared endpoint, the two concatenation reads, the affine-linearity read, and the exported classifier face. The packet is local to the displayed operation law: it does not supply a general contour-integration engine, a hidden analytic continuation principle, or an unrecorded host equality of integrals.

The affine-concatenation packet consumes the operation-law surface of chapter 521 and the finite Riemann-sum window surface of chapter 587. The former supplies the contour-operation output row, and the latter supplies the windowed summation face whose two contour pieces are combined by the affine packet.

Definition 704.1 (Contour affine-concatenation carrier)**.** *A* $\mathsf{ContourAffineConcat}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (\gamma_1, \gamma_2, f, g, a, b, I_1^f, I_2^f, I_1^g, I_2^g, J, H, C, P, N)$$

with the following displayed rows:

1. *γ_1 and γ_2 are accepted piecewise-linear contour rows with the shared endpoint condition displayed by the local concatenation route;*
2. *f and g are the two accepted holomorphic integrand rows read on the two contour pieces;*
3. *a and b are accepted complex scalar rows;*
4. *I_1^f and I_2^f are the certified contour-integral output rows for f along γ_1 and γ_2;*
5. *I_1^g and I_2^g are the certified contour-integral output rows for g along γ_1 and γ_2;*
6. *J is the output row for the contour integral of $af + bg$ along the concatenated contour $\gamma_1 * \gamma_2$, with displayed value $a(I_1^f + I_2^f) + b(I_1^g + I_2^g)$;*
7. *H records componentwise* hsame *transports for the contour rows, integrand rows, scalar rows, four input integral rows, concatenated contour row, affine output row, provenance rows, and the local naming row;*
8. *C records the displayed* Cont *routes for shared-endpoint admission, the two concatenation reads, the affine-linearity read, and the final classifier read of J;*

9. *P is the* Pkg *provenance over* ContourIntegralSocket$^{\uparrow}$, ContourIntegralOperation$^{\uparrow}$, ContourSumWindow$^{\uparrow}$, Holomorphic$^{\uparrow}$, Complex$^{\uparrow}$, *piecewise-linear contour rows,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourAffineConcat}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $\gamma_1, \gamma_2, f, g, a, b, I_1^f, I_2^f, I_1^g, I_2^g, J, H, C, P$

Theorem 704.2 (Contour affine-concatenation NameCert obligations)**.** *For the carrier of Definition 704.1, the local* $\mathsf{NameCert}_{\mathsf{ContourAffineConcat}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for $A = (\gamma_1, \gamma_2, f, g, a, b, I_1^f, I_2^f, I_1^g, I_2^g, J, H, C, P, N)$;*
2. *shared-endpoint concatenation, requiring γ_1 and γ_2 to expose the endpoint match used by the displayed $\gamma_1 * \gamma_2$ route;*
3. *affine linearity, requiring the two concatenated integral reads to be combined only as $a(I_1^f + I_2^f) + b(I_1^g + I_2^g)$ through the displayed scalar rows a, b;*
4. *output classifier stability, requiring every public comparison of the output row J to pass through the same componentwise* hsame *transport row H and continuation row C;*
5. *ledger exactness, requiring the provenance row P to name the contour socket, contour operation, contour-sum window, holomorphic integrand, complex scalar, contour, transport, continuation, package, and naming rows used by the packet;*
6. *non-escape, requiring the packet to export no hidden contour endpoint, no unrecorded integral value, no host equality of complex histories, no analytic-continuation shortcut, and no operation law outside the displayed finite rows.*

Proof. Carrier admission is projection from the tuple in Definition 704.1. The two contour rows, two integrand rows, scalar rows, four input integral rows, output row, transport row, continuation row, provenance row, and local naming row are all displayed coordinates of the same finite packet.

Shared-endpoint concatenation is confined to the contour part of C. A consumer first reads γ_1 and γ_2, then the displayed endpoint match, and only then the concatenated contour row $\gamma_1 * \gamma_2$.

Affine linearity uses the two concatenation reads before the scalar combination is exported. The f-rows produce $I_1^f + I_2^f$, the g-rows produce $I_1^g + I_2^g$, and the scalar rows a, b produce the displayed value of J.

Output classifier stability and ledger exactness are the public-read constraints. The row H carries componentwise classifier transport, C carries the replay route, and P lists the neighboring contour-integral, holomorphic, complex, and BEDC structural packets.

Finally inspect the carrier list. Since there is no coordinate for a hidden endpoint, unrecorded integral value, host equality of complex histories, analytic-continuation shortcut, or unlisted operation law, none of those objects can be exported by the local naming surface. □

Theorem 704.3 (Contour affine-concatenation consumer exhaustion)**.** *Every consumer read of an accepted* ContourAffineConcat$^{\uparrow}$ *packet factors through the carrier rows of Definition 704.1 and the local obligations of Definition 704.2. The consumer receives exactly the displayed contour rows, integrand rows, scalar rows, input integral rows, affine output row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, with no additional analytic or host-side equality coordinate.*

Proof. Begin with the carrier projection. The consumer can read only $\gamma_1, \gamma_2, f, g, a, b, I_1^f, I_2^f, I_1^g, I_2^g, J, H, C, P, N$.

The NameCert obligations then determine the route. Shared-endpoint concatenation binds the contour row, affine linearity binds the output expression, classifier stability binds public comparison of J, and ledger exactness binds the neighboring contour-integral packets named by P.

The remaining coordinates are the transport, continuation, provenance, and local naming rows. None of them contains an additional analytic-continuation principle, hidden endpoint, unrecorded integral value, or host equality of complex histories, so the displayed finite rows exhaust the public surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContourAffineConcat}^{\uparrow}}$.

Constructive story: A $\mathsf{ContourAffineConcat}^{\uparrow}$ packet is generated from finite BHist rows for neighboring contour-integral packets, their shared endpoint, affine output expression, classifier row, transport, continuation, provenance, and local naming. Concatenation is produced by replaying the displayed endpoint and affine-linearity routes through Cont, so the output expression is tied to the visible neighboring contour rows. Componentwise hsame preserves the contour, endpoint, integral, and affine rows, while Pkg records the contour-integral dependencies consumed by the packet. The public surface is the finite concatenation ledger, not analytic continuation or host equality of complex paths.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourAffineConcat}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the contour-affine-concat carrier, shared-endpoint concatenation, affine-linearity obligation, classifier stability, ledger exactness, and public-surface exhaustion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContourAffineConcat}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim analytic-continuation principles, hidden endpoints, unrecorded integral values, host equality of complex histories, or a global contour algebra.

Upgrade path: Obligation closure requires Lean encodings of the concatenation carrier, shared-endpoint route, affine output row, classifier stability, and ledger exactness.

Chapter 705

A Concrete Naming Certificate for $\mathsf{CriticalStripBarrier}^{\uparrow}$

The $\mathsf{CriticalStripBarrier}^{\uparrow}$ packet realizes the RH approach-rate witness as finite barrier-ledger data. It binds a supplied critical-strip row, a supplied zeta-zero row, a critical-line witness row, the real-part and rational lower-bound rows, transport and continuation rows, package provenance, and a local naming row. The witness row W is consumed from chapter 548 as the finite critical-line witness route, and this barrier chapter only transports that sibling row into the strip, zero, and refusal ledger rather than proving RH or adding analytic zero search. The packet records how a consumer may route a zero-localization or off-line lower-bound read; it does not prove RH, compute zeros, or assert a global analytic lower bound outside the displayed finite rows.

Definition 705.1 (Critical-strip-barrier carrier)**.** *A* $\mathsf{CriticalStripBarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Z, W, R, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the supplied* $\mathsf{CritStrip}^{\uparrow}$ *row carrying the critical-strip membership data for the displayed source history;*
2. *Z is the supplied* $\mathsf{ZetaZeros}^{\uparrow}$ *row carrying the zeta-zero premise for the same displayed source when a zero-localization read is requested;*
3. *W is the supplied* $\mathsf{CriticalLineWitness}^{\uparrow}$ *row available to consumers that already carry a critical-line witness route;*
4. *R is the real-part comparison row reading $\Re(s)$ through the* $\mathsf{Real}^{\uparrow}$ *carrier exposed by the strip and zero rows;*
5. *Q is the rational distance row recording the displayed lower-bound comparison against the rational history $1/2$ in* $\mathsf{Rat}^{\uparrow}$*;*
6. *A is the approach-rate ledger row that pairs each displayed rational separation depth with the corresponding positive rational lower-bound entry;*
7. *H records componentwise* hsame *transports for S, Z, W, R, Q, A and for the package and local naming coordinates;*

8. *C records the displayed* Cont *routes from the strip row to the zero row, from the real-part row to the rational distance row, from the distance row to the approach-rate ledger, and from the witness row to the RH-facing handoff;*
9. *P is the* Pkg *provenance over* $\mathsf{CritStrip}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{CriticalLineWitness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CriticalStripBarrier}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed strip, zero, witness, real-part, rational-bound, approach-rate, transport, continuation, package, and naming rows.

Lean4 ✓ `BEDC.Derived.CriticalStripBarrierUp.CriticalStripBarrierTasteGate single carrier alignment`

Theorem 705.2 (CriticalStripBarrier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CriticalStripBarrier}^{\uparrow}}$ *surface for Definition 705.1 consists exactly of six finite obligations:*

1. *carrier admission for $B = (S, Z, W, R, Q, A, H, C, P, N)$;*
2. *zero-localization routing from the* $\mathsf{CritStrip}^{\uparrow}$ *row S to the* $\mathsf{ZetaZeros}^{\uparrow}$ *row Z through displayed* hsame *and* Cont *rows;*
3. *rational separation exposure through R, Q, A as the only place where a lower-bound comparison against $1/2$ is read;*
4. *witness handoff through W only when a consumer already supplies the critical-line witness route;*
5. *componentwise stability under the transports in H and replay through the routes in C;*
6. *non-escape from a proof of* RH*, analytic zero search, host complex equality, quotient zero-set equality, or an undisplayed global lower-bound theorem.*

Proof. Carrier admission is projection from Definition 705.1. The strip row, zero row, witness row, real-part row, rational distance row, approach-rate ledger, transport row, continuation row, package row, and local naming row are all displayed coordinates.

Zero-localization routing is read only through S, Z, H, C. These rows bind the supplied critical-strip premise to the supplied zeta-zero premise without replacing either one by host complex equality or by a selected representative.

The approach-rate side is read through R, Q, A. The real row gives the displayed real part, the rational row gives the displayed comparison against $1/2$, and the ledger row gives the finite lower-bound entry attached to that comparison.

The handoff row W is a consumer boundary, not an RH proof. It may be read only as an already supplied critical-line witness route, so the packet does not turn a zero-in-strip premise into a proof that all zeros lie on the line.

Finally inspect the carrier inventory. Since no coordinate names zero search, quotient zero-set equality, host complex equality, a global analytic lower-bound theorem, or an RH inhabitant, the six obligations exhaust the local naming surface. □

Theorem 705.3 (CriticalStripBarrier zero-localization route)**.** *For an accepted* $\mathsf{CriticalStripBarrier}^{\uparrow}$ *carrier*

$$B = (S, Z, W, R, Q, A, H, C, P, N),$$

every zero-localization read factors through the displayed rows S, Z, H, C, P, N. The consumer receives the supplied critical-strip premise, the supplied zeta-zero premise, the componentwise transport tying them to the same displayed source, the continuation route by which the rows are replayed,

package provenance, and the local name; it receives no zero search, host equality of complex histories, quotient zero-set equality, or selected representative.

Proof. Project the zero-facing rows from the carrier. The rows available to such a read are S, Z, H, C, P, N.

The strip row S supplies the critical-strip side, and the zero row Z supplies the zeta-zero side. The transport and continuation rows bind them as one displayed route without adding a host equality or quotient comparison.

The package and local naming rows close the finite route. Because the carrier contains no coordinate for zero search or representative selection, the zero-localization read remains exactly the displayed finite packet route. □

Theorem 705.4 (CriticalStripBarrier non-escape)**.** *Every public read exported by an accepted* $\mathsf{CriticalStripBarrier}^{\uparrow}$ *packet factors through* $S, Z, W, R, Q, A, H, C, P, N$*. A request for a proof of* RH*, an analytic zero-finding procedure, a host complex equality, a quotient zero-set equality, or a lower-bound theorem not displayed in the finite approach-rate ledger is outside the packet and is not exported as NameCert evidence.*

Proof. Read a public request through the displayed continuation row C. If the request asks for the strip row, zero row, witness row, real-part row, rational-bound row, approach-rate ledger, transport row, package provenance, or local naming row, the carrier supplies the corresponding coordinate.

All other requests leave the carrier inventory. In particular, no coordinate carries an RH proof, analytic zero search, host complex equality, quotient zero-set equality, or an undisplayed lower-bound theorem.

Thus the local $\mathsf{NameCert}$ row can certify only the finite barrier ledger and its displayed routes. It cannot export evidence that is absent from the carrier. □

Theorem 705.5 (CriticalStripBarrier RH witness handoff)**.** *The RH approach-rate witness of Definition 9239.14 is realized, at the concrete-instance level, by the finite* $\mathsf{CriticalStripBarrier}^{\uparrow}$ *packet*

$$S, Z, W, R, Q, A, H, C, P, N.$$

An RH-facing consumer may read zero localization through S, Z, H, C, P, N*, rational lower-bound separation through* R, Q, A, H, C, P, N*, and an already supplied critical-line witness route through* W, H, C, P, N*. The handoff is a finite NameCert packet for the witness surface; it is not a proof of* RH *and not an analytic theorem that every off-line point is bounded away from zero.*

Proof. Start with the carrier definition and NameCert obligations. They expose exactly the strip, zero, witness, real-part, rational-bound, approach-rate, transport, continuation, package, and local naming rows.

Use the zero-localization route for S, Z, H, C, P, N. This supplies the premise side needed by an RH-facing consumer that is inspecting a zero in the critical strip.

Use the rational-bound rows R, Q, A, H, C, P, N for the approach-rate side. These rows provide only the finite lower-bound ledger entries that are displayed in the packet.

Finally use the non-escape theorem. Since the carrier has no coordinate for an RH proof or a global analytic lower-bound theorem, the exported object is precisely the finite witness handoff named by $\mathsf{CriticalStripBarrier}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CriticalStripBarrier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CriticalStripBarrier}^{\uparrow}$ packet is generated from finite BHist rows for a critical-strip premise, zeta-zero premise, critical-line witness, real-part comparison, rational distance, approach-rate ledger, transports, continuations,

provenance, and local name. A zero-localization or lower-bound read is routed by Cont through the strip, zero, real, rational, and approach-rate rows before any witness handoff is available. Componentwise hsame preserves the displayed rows, and Pkg records the strip, zero, witness, real, and rational dependencies. The packet is a finite barrier ledger and explicitly not an RH proof.

Theory closure: TheoryGate(NameCert$_{\mathsf{CriticalStripBarrier}\uparrow}$, seed)

Scope closed: This chapter binds the critical-strip-barrier carrier, NameCert obligations, zero-localization route, rational separation exposure, witness handoff, and non-escape theorem.

Formal verification: FormalStatus(NameCert$_{\mathsf{CriticalStripBarrier}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim the Riemann hypothesis, analytic zero search, host complex equality, quotient zero-set equality, or a global analytic lower-bound theorem.

Upgrade path: Obligation closure requires Lean encodings of the barrier carrier, zero-localization route, rational-bound ledger, witness handoff, and non-escape theorem.

Chapter 706

A Concrete Naming Certificate for $\mathsf{ObservableCauchySeal}^{\uparrow}$

The $\mathsf{ObservableCauchySeal}^{\uparrow}$ packet records the finite observation route used when a Real and Completion argument reads stream windows and dyadic radii before handing the result to a $\mathsf{Real}^{\uparrow}$ seal. It is an inspectable naming surface between observation budgets and real-facing seal rows. It does not assert real completeness, quotient stream equality, host real equality, or an unbounded Cauchy convergence principle.

Definition 706.1 (Observable Cauchy seal carrier)**.** *An* $\mathsf{ObservableCauchySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (E, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is a unary precision request row naming the requested observation scale;*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row visible at that precision;*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and displayed radius rows read from W;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the finite dyadic observations as a regular-rational route;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the finite readback and regular-rational handoff rows are fixed;*
6. *H records componentwise* hsame *transports for E, W, D, R, S and the local structural rows;*
7. *C records the displayed* Cont *routes from precision request to window, from window to dyadic readback, from dyadic readback to regular-rational handoff, and from the handoff to the seal row;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local certificate rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservableCauchySeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed real value, quotient stream equality, host metric equality, selected infinite tail, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.ObservableCauchySealUp.ObservableCauchySealTasteGate single carrier alignment`

Theorem 706.2 (Observable Cauchy seal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservableCauchySeal}^{\uparrow}}$ *surface for Definition 706.1 consists of carrier admission, precision exposure, finite-window readback, dyadic radius exposure, regular-rational handoff, RealUp seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every accepted packet exposes exactly the rows $E, W, D, R, S, H, C, P, N$ *before any consumer can read the seal row.*

Proof. Carrier admission is projection from the displayed tuple O. The precision, window, dyadic readback, regular-rational handoff, seal, transport, route, provenance, and naming rows are all explicit coordinates of the packet.

The finite-window and dyadic-radius obligations read E before W, and W before D. Thus every dyadic observation is attached to a named finite window and a named precision request.

The regular-rational handoff and seal-routing obligations then read R and S only after those finite rows are visible. The remaining rows H, C, P, N transport, replay, package, and name the same route without adding a completed real value, quotient stream equality, host metric equality, selected infinite tail, or ambient completeness witness. □

Theorem 706.3 (Observable Cauchy seal window factorization)**.** *Every public read of an accepted* $\mathsf{ObservableCauchySeal}^{\uparrow}$ *packet factors through the finite route*

$$E \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The route may use the componentwise hsame *rows, displayed* Cont *replay,* Pkg *provenance, and local naming row, but it cannot bypass the finite stream window or dyadic readback rows.*

Proof. Start with the accepted carrier. The only visible route from a precision request to a seal row passes through the displayed coordinates E, W, D, R, S.

Read the route order through the continuation row C. The window row W is reached from the precision request E, the dyadic readback row D is reached from W, and the regular-rational handoff row R is reached from D.

The seal row S is therefore downstream of the finite readback. Since no second window or dyadic coordinate is present in the carrier, a public read has no route around W or D. □

Theorem 706.4 (Observable Cauchy seal Real handoff)**.** *Let*

$$O = (E, W, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{ObservableCauchySeal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *consumer of* O *receives the seal row* S *only through the displayed precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and radii,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Project the packet to the finite rows $E, W, D, R, S, H, C, P, N$. The Real-facing coordinate is the single seal row S.

Apply the window factorization theorem. Before S can be consumed, the route has already exposed the precision request, stream window, dyadic readback and radius rows, and regular-rational handoff.

The structural rows H, C, P, N carry only transport, replay, provenance, and local naming for that finite route. Hence the RealUp consumer receives the displayed finite observation handoff and no additional completion coordinate. □

Theorem 706.5 (Observable Cauchy seal L10 dependency lattice)**.** *Every accepted* $\mathsf{ObservableCauchySeal}^{\uparrow}$ *packet consumes the L10 route through chapter 126, chapter 291, chapter 301, and chapter 38 in the displayed order*

$$E, W, D, R, S, H, C, P, N.$$

No Real-facing consumer may bypass the finite $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback ledger, or the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Project the carrier rows from Definition 706.1. The rows $E, W, D, R, S, H, C, P, N$ are the only public coordinates of the packet, with W naming the finite stream window, D recording the dyadic readbacks and radii, R recording the regular-rational handoff, and S recording the RealUp seal row.

Apply Definition 706.3. Any public read of S factors through the route $E \longmapsto W \longmapsto D \longmapsto R \longmapsto S$, while H, C, P, N only transport, replay, package, and name that same finite route. The named chapter route is therefore the horizontal dependency lattice from $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{DyadicRatCore}^{\uparrow}$, then to $\mathsf{RegSeqRat}^{\uparrow}$, and finally to $\mathsf{Real}^{\uparrow}$.

Apply Definition 706.4 to the Real-facing consumer and Definition 706.6 to the attempted exits. Since the carrier contains no quotient stream equality, host real equality, ambient completeness witness, completed real value, or unrecorded Cauchy tail, the consumer cannot reach S by a route outside the finite window, dyadic ledger, and RegSeqRat handoff. □

Theorem 706.6 (Observable Cauchy seal non-escape)**.** *An accepted* $\mathsf{ObservableCauchySeal}^{\uparrow}$ *packet exports no real-completeness theorem, no quotient stream equality, no host real equality, no countable-choice selector, and no unrecorded Cauchy tail. Any attempted consumer omitting* E*,* W*,* D*, or* R *is outside the packet and cannot export the* $\mathsf{Real}^{\uparrow}$ *seal row* S*.*

Proof. Read the carrier inventory first. The only exportable rows are $E, W, D, R, S, H, C, P, N$, and the local obligations require those rows to be visible in the finite observation route.

If a consumer omits E or W, the packet supplies no named observation window. If it omits D, the packet supplies no dyadic readback or radius ledger. If it omits R, the packet supplies no regular-rational handoff to the seal row.

The attempted read is therefore outside the local naming surface. Because the carrier has no coordinate for real completeness, quotient stream equality, host real equality, countable choice, or an unrecorded Cauchy tail, none of those objects can be exported through S. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservableCauchySeal}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObservableCauchySeal}^{\uparrow}$ packet is generated from a unary precision request, finite StreamName window, DyadicRatCore readbacks and radii, RegSeqRat handoff row, RealUp seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservableCauchySeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 706.1, Definition 706.2, Definition 706.3, Definition 706.4, and Definition 706.6 bind finite observation routes from stream windows and dyadic readbacks into a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservableCauchySeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close real completeness, quotient stream equality, host real equality, countable choice, unrecorded Cauchy tails, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, dyadic radius exposure, RegSeqRat handoff, RealUp seal routing, and non-escape for the same packet.

Chapter 707

A Concrete Naming Certificate for ProofTermErasureClassifier$^{\uparrow}$

The ProofTermErasureClassifier$^{\uparrow}$ packet is the classifier-level surface for proof-term erasure. It reads proof-term membership as an Ext row, exposes the erased term through displayed Cont replay and SigRel readback, and packages the resulting public classifier row with ProofTermErasureTrace$^{\uparrow}$ and ProofTermErasureAuditRoute$^{\uparrow}$ provenance so later consumers receive a finite erasure-preserving membership object.

The classifier consumes the finite trace surface of chapter 642 and the audit-route surface of chapter 690. It contributes only the erased membership readback row and does not replace either sibling trace or audit route.

Theorem 707.1 (Proof-term erasure classifier trace sibling route)**.** *Every accepted* ProofTermErasureClassifier$^{\uparrow}$ *carrier consumes the trace surface of chapter 642 and the audit route of chapter 690 before exporting erased classifier membership. The finite route passes through typing* Ext *membership, erasure* Cont *replay,* SigRel *readback, componentwise* hsame *transport,* Pkg *provenance, and local* NameCert *rows.*

Proof. Project the classifier packet. Its membership source is the displayed Ext row, and its erased output is read only after the erasure replay and SigRel readback rows have been exposed.

The trace chapter supplies the finite trace surface that records how the proof-term row is replayed as an erased row. The audit-route chapter supplies the provenance path that keeps that replay inspectable before a downstream classifier reads the erased membership.

Finally route both sibling surfaces through the carrier rows H, Q, N. Since the exported membership is packaged only after hsame transport, Pkg provenance, and local naming, the classifier cannot replace the trace or audit route by an ambient proof-erasure oracle. □

Definition 707.2 (Proof-term erasure classifier carrier)**.** *A* ProofTermErasureClassifier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, E, M, R, S, H, Q, N)$$

with the following displayed rows:

1. *P is the proof-term row whose classifier membership is tested;*
2. *E is the erased-term row exported to downstream classifier consumers;*
3. *M is the* Ext *membership row for the proof-term row P;*

4. *R is the* Cont *erasure replay route from P to E;*
5. *S is the* SigRel *readback row naming the erased classifier output as the public reading of the membership row M along R;*
6. *H records componentwise* hsame *transports for the proof-term row, erased-term row,* Ext *membership row,* Cont *replay row,* SigRel *readback row, package provenance, and local naming row;*
7. *Q is the* Pkg *provenance over the displayed* BHist, Ext, Cont, SigRel, hsame, ProofTermErasureTrace$^{\uparrow}$, ProofTermErasureAuditRoute$^{\uparrow}$, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureClassifier}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through P, E, M, R, S, H, Q, N.

Lean4 ✓ `BEDC.Derived.ProofTermErasureClassifierUp.ProofTermErasureClassifierTasteGate single carrier alignment`

Theorem 707.3 (Proof-term erasure classifier Ext-membership)**.** *Every accepted* ProofTermErasureClassifier$^{\uparrow}$ *carrier reads classifier membership only from the displayed* Ext *row M. If a downstream consumer accepts the erased row E as a classifier member, the source of that membership is the proof-term row P together with M, transported componentwise through H and packaged by Q, N.*

Proof. Project the carrier of Definition 707.2. The only membership coordinate is M, and it is attached to the proof-term row P.

The erased row E is not a second classifier source. It is reached from P through the displayed continuation row R and named for public reading by the SigRel row S.

Finally route the membership read through H, Q, N. These rows preserve and package the same displayed coordinates, so a downstream consumer cannot replace M by host type checking or by an undisplayed classifier. □

Theorem 707.4 (Proof-term erasure classifier erasure preservation)**.** *For every accepted* ProofTermErasureClassifier *carrier $K = (P, E, M, R, S, H, Q, N)$, the erased public classifier row preserves the proof-term* Ext *membership displayed by M. The preservation route is exactly the* Cont *replay row R together with the* SigRel *readback row S and the componentwise* hsame *transport row H.*

Proof. Read the packet coordinates. The source membership is M, the erased public row is E, the replay route is R, and the public readback is S.

Use the displayed replay row to carry the membership read from the proof-term row to the erased-term row. The SigRel row records the classifier readback attached to that replay, so the exported row is the erased reading of the same membership event.

Transport is exhausted by H and provenance by Q, N. Since no other route is present in the carrier, erasure preservation is exactly the displayed finite route. □

Theorem 707.5 (Proof-term erasure classifier non-escape ledger)**.** *Every accepted* ProofTermErasureClassifier$^{\uparrow}$ *carrier exports only the erased row E, the proof-term membership row M, the erasure replay row R, the readback row S, the transport row H, the package row Q, and the local naming row N. It exports no proof object as consumer evidence, no host conversion relation, no quotient syntax equality, no proof irrelevance principle, and no ambient normalization oracle.*

Proof. Inspect the carrier inventory. The proof-term row P is present only as the source named by the membership row M, while the consumer-facing row is E.

The exported evidence is routed through R, S, H, Q, N. These rows record replay, readback, transport, provenance, and local naming, and none of them carries a proof object or a host conversion principle.

The excluded objects have no coordinate in K. Therefore they cannot be read from the local certificate surface. □

Theorem 707.6 (Proof-term erasure classifier NameCert obligations)**.** *For the carrier of Definition 707.2, the local* $\mathsf{NameCert}_{\mathsf{ProofTermErasureClassifier}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission exposes exactly* $K = (P, E, M, R, S, H, Q, N)$*;*
2. *membership exactness reads the accepted classifier decision only from the displayed* Ext *row* M*;*
3. *erasure preservation routes the public erased row only through the displayed* Cont *row* R *and* SigRel *row* S*;*
4. *trace and audit provenance are carried only by the displayed* Pkg *row* Q *over* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *and* $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ *rows;*
5. *the non-escape ledger binds every consumer read to* H, Q, N *and excludes proof-object export, host conversion, quotient syntax equality, proof irrelevance, and ambient normalization.*

Proof. Carrier admission is projection from the finite tuple K. The local naming row exposes exactly the eight displayed coordinates and no hidden carrier field.

Membership exactness is the role of M. Erasure preservation is then read through R and S, which replay the erased row and name its classifier readback.

The provenance obligation is carried by Q. It records the dependency on the proof-term erasure trace and audit-route packets while keeping the consumer surface finite.

Finally apply the non-escape ledger of Definition 707.5. Since every consumer read is already confined to H, Q, N and the displayed erasure rows, the certificate cannot export proof objects or host-level conversion data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTermErasureClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTermErasureClassifier}^{\uparrow}$ packet is generated from proof-term, erased-term, Ext-membership, Cont replay, SigRel readback, hsame transport, Pkg provenance, trace, audit-route, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProofTermErasureClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 707.2, Definition 707.3, Definition 707.4, Definition 707.5, and Definition 707.6 define the finite erasure-preserving classifier surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProofTermErasureClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, public compiler correctness, host conversion, quotient syntax equality, proof irrelevance, ambient normalization, or proof-object export.

Upgrade path: Obligation closure requires checked carrier admission, Ext-membership exactness, erasure preservation, trace and audit-route provenance, and non-escape for the displayed classifier rows.

Chapter 708

RegularCauchyTelescopingBudget NameCert Construction

The $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ packet records the finite chained-window error ledger used by Real-completion diagonal handoffs. It starts with one precision request, opens a finite chain of selected $\mathsf{StreamName}^{\uparrow}$ windows, reads their dyadic error rows, transports the result through $\mathsf{RegSeqRat}^{\uparrow}$, seals the terminal row through $\mathsf{Real}^{\uparrow}$, and records the telescoping composition that makes the intermediate errors visible. The packet is a NameCert surface for finite budget composition; it is not a quotient of streams, a selected host limit, countable choice, or an ambient completeness principle.

This packet is a horizontal L10 consumer of the selected-window surface in chapter 660 and the adjacent-window fusion surface in chapter 693. Together with the finite-window source of chapter 126, the dyadic endpoint ledger of chapter 291, the regular-sequence handoff of chapter 301, and the seal row of chapter 38, those sibling chapters supply the finite $\mathsf{StreamName}^{\uparrow}$ windows and fusion route before the packet reads $\mathsf{DyadicRatCore}^{\uparrow}$ error rows, passes through $\mathsf{RegSeqRat}^{\uparrow}$, reaches the $\mathsf{Real}^{\uparrow}$ seal, and exports the telescoping ledger T.

Definition 708.1 (RegularCauchyTelescopingBudget carrier)**.** *A* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, W, D, R, S, T, H, C, P, N)$$

with the following displayed rows:

1. *E is the precision request whose requested Real-completion tolerance is being budgeted;*
2. *W is a finite chain of selected* $\mathsf{StreamName}^{\uparrow}$ *windows opened from the displayed regular Cauchy sources;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error ledger read from the successive windows in W;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained after the selected dyadic errors have been composed;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the handoff R at the requested precision;*
6. *T is the telescoping composition ledger, recording the finite sum of adjacent-window errors and the remaining budget after each handoff;*
7. *H records componentwise* hsame *transports for E, W, D, R, S, T and for the structural rows of the packet;*

8. *C records the displayed* Cont *routes for precision exposure, window-chain admission, dyadic-ledger reads, telescoping composition,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal delivery;*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $E, W, D, R, S, T, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier visible rows`

Theorem 708.2 (RegularCauchyTelescopingBudget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTelescop}}$ *surface for Definition 708.1 consists of carrier admission for*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

precision exposure through E, finite window-chain admission through W, dyadic error-ledger exposure through D, telescoping composition through T, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R,* $\mathsf{Real}^{\uparrow}$ *seal delivery through S, componentwise* hsame *stability, displayed* Cont *route replay,* Pkg *provenance, local naming, and non-escape from quotient streams, selected host limits, countable choice, ambient completeness, hidden comparison rows, and unrecorded error budgets.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 708.1. The packet contains exactly the rows $E, W, D, R, S, T, H, C, P, N$.

The analytic-facing obligations are read in order. The precision request E determines which finite chain of selected $\mathsf{StreamName}^{\uparrow}$ windows W is opened. The dyadic ledger D records the error rows read from those windows, and the telescoping ledger T records their finite composition before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is exposed.

The real-facing obligation is the seal row S. It is reached only after R has consumed the displayed window chain and the telescoping ledger.

Finally H, C, P, N preserve, route, package, and name the same finite packet. Because no coordinate names quotient streams, selected host limits, countable choice, ambient completeness, hidden comparison rows, or unrecorded budgets, the local NameCert exports none of that data. □

Theorem 708.3 (RegularCauchyTelescopingBudget dyadic ledger exposure)**.** *For every accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *carrier*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

the dyadic ledger row D is readable only after the precision row E and the finite $\mathsf{StreamName}^{\uparrow}$ *window chain W have been exposed. The row D supplies adjacent-window* $\mathsf{DyadicRatCore}^{\uparrow}$ *error entries for the displayed finite route, and it supplies no quotient-stream equality, selected host limit, choice selector, ambient completeness row, hidden comparison row, or independent Real carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier dyadic ledger exposure`

Proof. Read the carrier in Definition 708.1. Its ordered packet has one precision request E, one finite selected-window chain W, and one dyadic error ledger D.

Apply the NameCert obligations of Definition 708.2. Precision exposure through E opens the finite $\mathsf{StreamName}^{\uparrow}$ chain W, and D is then the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger read from those windows. The surrounding $\mathsf{RegularCauchySequence}^{\uparrow}$ and $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ surfaces provide the selected-window sources and shared-window handoff route, not an ambient completion object.

Finally inspect the packet boundary. Quotient-stream equality, selected host limits, choice selectors, ambient completeness rows, hidden comparison rows, and independent Real carriers are absent from the coordinates $E, W, D, R, S, T, H, C, P, N$. The dyadic ledger therefore exposes only the finite adjacent-window error rows named by D. □

Theorem 708.4 (RegularCauchyTelescopingBudget telescoping composition)**.** *For every accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *carrier*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

the telescoping composition row T consumes exactly the finite adjacent-window error rows exposed by D, records the remaining budget at each finite handoff, and is stable only under the displayed componentwise hsame *transports in H.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier telescoping composition`

Proof. Use Definition 708.1 to project the finite rows D, T, H. The ledger D is the only coordinate containing adjacent-window dyadic error entries, and T is the only coordinate recording their finite telescoping composition.

Apply Definition 708.2 and Definition 708.3. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, and $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ surfaces expose a finite list of adjacent-window error rows. The composition row T consumes that displayed list and records the remaining budget passed at each handoff.

Stability is then confined to H. Since H records componentwise hsame transports for E, W, D, R, S, T and the structural rows, a transported packet preserves the same finite telescoping ledger exactly along those displayed transports and along no unrecorded comparison row. □

Theorem 708.5 (RegularCauchyTelescopingBudget RegSeqRat handoff order)**.** *For every accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *carrier*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R is exposed only after E, W, D, T, and the* $\mathsf{Real}^{\uparrow}$ *seal row S is exposed only after R.*

Proof. First read the carrier and NameCert surface. By Definition 708.1 and Definition 708.2, the finite route exposes precision, selected windows, dyadic ledger rows, and telescoping composition as E, W, D, T before the handoff row R.

The dyadic and telescoping obligations make this order explicit. Apply Definition 708.3 to obtain the E, W, D prefix, then apply Definition 708.4 to consume the adjacent-window $\mathsf{DyadicRatCore}^{\uparrow}$ rows through T. The handoff row R is therefore the first $\mathsf{RegSeqRat}^{\uparrow}$ row downstream of the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, and $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ route.

Finally apply Definition 708.7. The seal row S is the only $\mathsf{Real}^{\uparrow}$ seal row in the carrier, and that theorem places it after the route through E, W, D, T, R. Thus the Real-facing seal cannot be read before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff has been exposed. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier regseqrat handoff order`

Theorem 708.6 (RegularCauchyTelescopingBudget window chain)**.** *For every accepted* RegularCauchyTelescopingB *carrier*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

the selected windows form a finite chain read through $\mathsf{RegularCauchySequence}^{\uparrow}$ *and* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ *surfaces. Each adjacent-window error consumed by the chain is recorded in D, and each transfer of remaining precision is recorded in T before the handoff row R is available.*

Proof. Project E, W, D, T from the carrier. The precision row E fixes the requested tolerance, and the window row W is the only coordinate that opens selected $\mathsf{StreamName}^{\uparrow}$ windows.

Use the existing $\mathsf{RegularCauchySequence}^{\uparrow}$ and $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$ surfaces as the displayed sources of the chain. They expose finite selected windows and shared-window Real-facing routes without selecting a host limit or quotient representative.

Now read the dyadic and telescoping rows. The ledger D stores the adjacent-window error rows, while T composes them and records the remaining budget passed to the next window. Since R appears only after D and T, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is downstream of the finite chain. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier window chain`

Theorem 708.7 (RegularCauchyTelescopingBudget Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *carrier factors through the route*

$$E, W, D, T, R, S, H, C, P, N.$$

The route first consumes the requested precision, then the selected window chain, then the dyadic error ledger, then the telescoping composition ledger, then the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and only then the* $\mathsf{Real}^{\uparrow}$ *seal row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier real seal handoff`

Proof. Begin with the window-chain theorem. It places the selected windows, dyadic error rows, telescoping composition, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff in the displayed order E, W, D, T, R.

Read the seal coordinate. The row S is the only $\mathsf{Real}^{\uparrow}$ seal row in the carrier, and the continuation row C exposes it only after the route through E, W, D, T, R has been replayed.

The rows H, P, N transport, package, and name the same route. Therefore every Real-facing consumer receives the seal through $E, W, D, T, R, S, H, C, P, N$ and through no independent completion object. □

Theorem 708.8 (RegularCauchyTelescopingBudget non-escape)**.** *For an accepted* RegularCauchyTelescopingBudge *carrier, the finite telescoping budget exports no quotient stream equality, selected host limit, countable-choice selector, ambient completeness principle, hidden comparison row, unrecorded error budget, or independent RealUp carrier. All exported data are confined to*

$$E, W, D, R, S, T, H, C, P, N.$$

Proof. Use the NameCert obligations to identify the complete carrier surface. The only admitted rows are $E, W, D, R, S, T, H, C, P, N$.

Use the window-chain theorem and the Real seal handoff theorem. They force every path to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal to consume the finite selected windows, dyadic ledger, and telescoping composition ledger.

Inspect the packet boundary. No coordinate stores quotient stream equality, selected host limits, countable choice, ambient completeness, hidden comparisons, unrecorded budgets, or an independent real carrier. The exported surface is exactly the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier nonescape`

Theorem 708.9 (RegularCauchyTelescopingBudget primitive scope binding)**.** *For every accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *packet*

$$B = (E, W, D, R, S, T, H, C, P, N),$$

the scoped primitive surface is exactly the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *error ledger* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window chain* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* S*, the componentwise* hsame *transport row* H*, the visible* Cont *route row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*, all read under the precision request* E *and telescoping ledger* T*. No host limit, quotient stream, countable-choice selector, hidden comparison, or ambient completeness row is part of this scope.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier primitive scope binding`

Proof. Project the carrier from Definition 708.1. Its only rows are $E, W, D, R, S, T, H, C, P, N$, with D naming the dyadic error ledger, W naming the finite selected-window chain, R naming the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and S naming the $\mathsf{Real}^{\uparrow}$ seal.

Use the local NameCert obligations and dyadic exposure. Theorems 708.2 and 708.3 bind precision exposure, window-chain admission, dyadic-ledger reads, componentwise transport, visible route replay, provenance, local naming, and non-escape to the same finite packet.

Now apply the finite route theorems. The telescoping composition theorem consumes the displayed dyadic entries through T, the window-chain theorem identifies the selected $\mathsf{StreamName}^{\uparrow}$ chain, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff order theorem places R after E, W, D, T, and the Real seal handoff theorem places S after R.

Finally apply Definition 708.8. Since the excluded host limit, quotient stream, countable-choice selector, hidden comparison, and ambient completeness rows do not occur in the carrier, the scoped primitive binding is precisely the existing finite route through $E, W, D, T, R, S, H, C, P, N$. □

Theorem 708.10 (RegularCauchyTelescopingBudget scoped closure)**.** *The carrier, dyadic ledger exposure, telescoping composition,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff order, finite window chain,* $\mathsf{Real}^{\uparrow}$ *seal handoff, non-escape boundary, and primitive-scope binding jointly close the scoped primitive surface of* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Start with the carrier and NameCert obligations. They expose exactly the rows $E, W, D, R, S, T, H, C, P, N$ and identify the visible source chapters for dyadic errors, selected windows, regular rational handoff, Real seal delivery, transport, replay, provenance, and local naming.

Use the ledger and composition theorems to bind the analytic content. Dyadic exposure places D after E and W, while telescoping composition consumes the displayed finite ledger through T without adding a hidden comparison row.

Use the handoff and non-escape theorems for the target boundary. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff order, window-chain theorem, and $\mathsf{Real}^{\uparrow}$ seal handoff place R and S downstream of the same finite route, and the non-escape theorem excludes quotient streams, selected host limits, countable choice, hidden comparisons, ambient completeness, and independent Real carriers.

Finally apply the primitive-scope binding. It identifies the complete scoped surface as $E, W, D, T, R, S, H, C, P, N$ over the listed BEDC primitives, so no additional primitive row remains outside the packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier scoped closure`

Theorem 708.11 (RegularCauchyTelescopingBudget public export)**.** *Every L10 diagonal handoff consumer that cites* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *may cite only the ordered public surface*

$$E, W, D, T, R, S, H, C, P, N$$

over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The consumer must read the precision request E, selected window chain W, dyadic error ledger D, telescoping composition ledger T,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal S, componentwise transport H, displayed replay C, provenance P, and local naming row N in that order before using the packet as a public diagonal budget. The public surface exports no quotient stream equality, selected host limit, countable-choice selector, hidden comparison, unrecorded error budget, ambient completeness principle, or independent RealUp carrier.*

Proof. Start with the carrier and scoped closure. The carrier displays exactly $E, W, D, R, S, T, H, C, P, N$, and Definition 708.10 binds those rows as the scoped primitive surface over the named BEDC prerequisites.

Use the Real seal handoff. Theorems 708.5 and 708.7 place R and S after the finite window, dyadic ledger, and telescoping ledger rows, so a diagonal consumer cannot read a Real-facing seal before the ordered route is exposed.

Finally apply non-escape. Theorem 708.8 excludes quotient streams, selected host limits, countable choice, hidden comparisons, unrecorded budgets, ambient completeness, and independent Real carriers. Therefore the reusable public cite is precisely the ordered surface $E, W, D, T, R, S, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier public export`

Theorem 708.12 (RegularCauchyTelescopingBudget Real bridge route)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of the public* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *export must read the* $\mathsf{StreamName}^{\uparrow}$ *window chain, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the window-chain rows, and the* $\mathsf{Real}^{\uparrow}$ *seal handoff before consuming the telescoping budget. Equivalently, the consumer must traverse*

$$E, W, D, T, R, S, H, C, P, N$$

with W read as the finite $\mathsf{StreamName}^{\uparrow}$ *chain, D read as the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, T read as the telescoping ledger, R read as the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and S read as the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start from the public export theorem. It identifies the reusable public surface as the ordered rows $E, W, D, T, R, S, H, C, P, N$ and excludes any independent Real carrier or ambient completeness row.

Use the finite route theorems to expose the bridge order. The window-chain theorem supplies the $\mathsf{StreamName}^{\uparrow}$ chain W and places its adjacent-window ledger in D before T; the $\mathsf{RegSeqRat}^{\uparrow}$ handoff order theorem places R after E, W, D, T.

Now apply the Real seal handoff and primitive scope binding. The Real seal handoff places S downstream of the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, while primitive scope binding keeps H, C, P, N as transport, replay, provenance, and naming rows for the same finite packet.

Thus a $\mathsf{Real}^{\uparrow}$ consumer cannot consume the telescoping budget merely as an isolated public row. It must first read the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, window-chain, and Real-seal route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetCarrier bridge route`

Theorem 708.13 (RegularCauchyTelescopingBudget window concatenation route)**.** *Suppose two accepted* $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ *packets share the terminal* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row of the first packet as the precision-facing source row of the second packet through displayed* hsame *transport. Then their concatenated finite* $\mathsf{StreamName}^{\uparrow}$ *window chain exports the ordered composite budget route*

$$E, W, D, T, R, S, H, C, P, N.$$

The dyadic error ledgers are appended in the displayed window order, the shared boundary is read through H before repacking, and the composite route introduces no quotient stream equality, selected host limit, countable choice, hidden comparison, ambient completeness, or independent $\mathsf{Real}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTelescopingBudgetUp.RegularCauchyTelescopingBudgetWindowConcatenation`

Proof. Project the two accepted carriers through Definition 708.1. The first packet supplies rows $E_0, W_0, D_0, T_0, R_0, S_0, H_0, C_0, P_0, N_0$, and the second supplies rows $E_1, W_1, D_1, T_1, R_1, S_1, H_1, C_1, P_1, N_1$.

Use the window-chain theorem and telescoping-composition theorem for both packets. The first route reads W_0, D_0, T_0 before R_0, S_0, and the second route reads W_1, D_1, T_1 before R_1, S_1.

Now read the shared boundary. The assumed displayed hsame transport identifies the terminal R_0, S_0 rows with the precision-facing source row of the second packet only through H_0 and H_1, so the handoff is visible before any composite public route is named.

Repack the composite route by appending the dyadic ledger lists and the finite window chains in that order. The RegSeqRat handoff order and Real seal handoff theorems place the resulting route in the order $E, W, D, T, R, S, H, C, P, N$.

Finally apply non-escape. Since neither input packet contains quotient stream equality, selected host limits, countable choice, hidden comparisons, ambient completeness, or an independent Real carrier, and since the shared boundary is read only through displayed transport, the concatenated route exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}$ packet is generated from a precision request, a finite chain of selected $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ error ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, a telescoping composition ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTelescopingBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 708.1, Definition 708.2, Definition 708.3, Definition 708.4, Definition 708.5, Definition 708.6, Definition 708.7, Definition 708.8, Definition 708.11, the bridge witness Definition 708.12, and the window concatenation route Definition 708.13 give the mature treatment for chained StreamName windows, dyadic ledgers, RegSeqRat handoff, and Real seal rows in Real-completion diagonal handoffs.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyTelescopingBudget}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, selected host limits, countable choice, ambient completeness, hidden comparison rows, unrecorded error budgets, or a public standard bridge.

Upgrade path: Bridged closure requires a standard bridge that transports the ordered public surface into the acceptance schema without adding quotient streams, selected host limits, countable choice, hidden comparisons, ambient completeness, or an independent RealUp carrier.

Chapter 709

A Concrete Naming Certificate for ClosedGeneratorAudit$^{\uparrow}$

The ClosedGeneratorAudit$^{\uparrow}$ packet names the finite audit surface shared by kernel acceptance, recursor authorization, and closed-term substitution. It packages MetaCIC generator-closure rows, closed-substitution rows, accepted kernel rows, audit-refusal rows, and local NameCert naming as one BEDC-readable object rather than as three unrelated host-side facts.

Its K and S rows consume the concrete sibling surfaces of chapter 547, chapter 557, and chapter 711 as the displayed kernel-acceptance and generator-audit dependency lattice.

Definition 709.1 (Closed generator audit carrier)**.** *A* ClosedGeneratorAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, K, R, S, A, H, C, P, N)$$

with the following displayed rows:

1. *T records the generated MetaCIC term or proof row, including the local syntax object whose closure and acceptance are audited;*
2. *K records the kernel acceptance row for the generated object and the accepted declaration or proof stamp exposed by* KernelAcceptanceAuditWitness$^{\uparrow}$ *or* KernelAcceptanceTrace$^{\uparrow}$*;*
3. *R records the authorized recursor row, including the emitted eliminator, checked motive, branch package, structural descent ledger, and generated output row supplied by* RecursionAuthorizationLedger$^{\uparrow}$*;*
4. *S records the closed substitution and shift row, including the* `GeneratorClosureClassifier`*,* `closed_under_substitute_classifier`*,* `shift_closed`*, and* `substitute_closed` *audit entries used to replay non-escape of closed terms;*
5. *A records the axiom-purity audit row, with refusal entries for* `Classical.choice`*,* `Quot.sound`*,* `propext`*, host conversion, and any unrecorded proof oracle;*
6. *H records componentwise* hsame *transport for the generated object, kernel row, recursor row, closed-substitution row, audit row, replay route, provenance, and local name;*
7. *C records the displayed* Cont *routes replaying kernel acceptance, recursor authorization, closed-substitution preservation, audit refusal, and consumer access;*
8. *P records the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* KernelAcceptanceAuditWitness$^{\uparrow}$*,* KernelAcceptanceTrace$^{\uparrow}$*,* RecursionAuthorizationLedger$^{\uparrow}$*, and* GeneratorAuditClosure$^{\uparrow}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{ClosedGeneratorAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host proof kernel, quotient transport, classical selection, proposition extensionality, hidden conversion, or an accepted proof oracle outside the displayed rows.

Theorem 709.2 (Closed generator audit NameCert obligations)**.** *For every accepted* $\mathsf{ClosedGeneratorAudit}^{\uparrow}$ *carrier*

$$G = (T, K, R, S, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedGeneratorAudit}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission exposes the generated object, kernel row, authorized recursor row, closed-substitution row, axiom-purity audit row, transport, replay, provenance, and local name as the packet $T, K, R, S, A, H, C, P, N$;*
2. *generator-closure replay reads the* `GeneratorClosureClassifier` *data in S and routes it through the accepted generator-audit packet;*
3. *closed-substitution non-escape reads* `closed_under_substitute_classifier`*,* `shift_closed`*, and* `substitute_closed` *only as the displayed closed-term rows in S;*
4. *kernel-acceptance replay reads K through the displayed kernel-acceptance witness or trace and connects authorized recursor output R to the generated row T;*
5. *audit-refusal exactness reads A as the complete refusal row for* `Classical.choice`*,* `Quot.sound`*,* `propext`*, host conversion, and any unrecorded proof oracle.*

Proof. Project the finite packet of Definition 709.1. Its carrier coordinates are precisely $T, K, R, S, A, H, C, P, N$, so carrier admission and local naming have no row outside the displayed tuple.

The generator side is supplied by the existing generator-audit closure surface. It reads the `GeneratorClosureClassifier` row and the accepted substitute or shift closure witness before exposing the closure claim to a consumer.

The substitution side is the closed-term boundary. The entries `closed_under_substitute_classifier`, `shift_closed`, and `substitute_closed` are recorded in S, and the replay route C keeps them attached to the same generated row T.

The kernel and recursor side is read through K and R. Kernel acceptance supplies the accepted declaration or proof stamp, while the recursion authorization ledger supplies the emitted recursor output and its structural authorization before it is identified with the generated row.

Finally use the audit row A. Since the forbidden dependencies and unrecorded proof-oracle requests are refused there, and since H, C, P, N only transport and replay the displayed rows, the five obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ClosedGeneratorAuditCarrier namecert obligations`

Theorem 709.3 (Closed generator audit non-escape)**.** *Every consumer of an accepted* $\mathsf{ClosedGeneratorAudit}^{\uparrow}$ *carrier*

$$G = (T, K, R, S, A, H, C, P, N)$$

can read only the displayed MetaCIC and BEDC rows

$$T, K, R, S, A, H, C, P, N.$$

In particular, the consumer may read generated syntax, kernel acceptance, authorized recursor output, closed substitution and shift preservation, axiom-purity refusal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It cannot read* `Classical.choice`*,* `Quot.sound`*,* `propext`*, host conversion, or an unrecorded proof oracle.*

Proof. Start from the carrier definition. The packet lists generated MetaCIC data, kernel acceptance, recursor authorization, closed-substitution audit data, audit refusal, transport, replay, provenance, and local naming as the full exported inventory.

Apply Definition 709.2. The NameCert surface routes every consumer read through one of the five displayed obligations, and each obligation returns to the same tuple $T, K, R, S, A, H, C, P, N$.

The refusal part is exact. Since the audit row A names the forbidden dependencies and unrecorded proof-oracle requests only as refused rows, no consumer route can reinterpret them as accepted kernel evidence or closed-term evidence.

Thus every readable row is a displayed MetaCIC or BEDC row in the packet. The excluded host, quotient, classical, extensional, conversion, and oracle coordinates are absent from the carrier and cannot be exported by transport or replay. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedGeneratorAudit}\uparrow}$.

Constructive story: A $\mathsf{ClosedGeneratorAudit}^{\uparrow}$ packet is generated from a MetaCIC term or proof row, kernel acceptance row, authorized recursor row, closed-substitution and shift row, axiom-purity audit row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedGeneratorAudit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 709.1, Definition 709.2, and Definition 709.3 specify the finite NameCert carrier, obligation surface, and non-escape boundary joining kernel acceptance, recursor authorization, and closed-term substitution audit rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedGeneratorAudit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove new MetaCIC closure lemmas, Lean kernel soundness, global recursor soundness, host conversion, quotient transport, classical choice, proposition extensionality, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator-closure replay, closed-substitution non-escape, kernel-acceptance replay, and audit-refusal exactness for the displayed packet.

Chapter 710

A Concrete Naming Certificate for $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$

The $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ packet turns an accepted recursor call into finite NameCert ledger data. It records the inductive signature accepted by the kernel, the emitted eliminator, the checked motive, the branch package, the structural descent row, the generated output row, transport and continuation routes, package provenance, and the local naming row that later audit and generator-closure consumers can inspect.

Definition 710.1 (RecursionAuthorizationLedger carrier)**.** *A* $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, R, M, B, D, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted inductive signature row, including the constructor inventory and the recursive argument positions admitted by the kernel;*
2. *R is the emitted eliminator row for I, read as the authorized recursor generated from that signature;*
3. *M is the motive row checked for the eliminator call;*
4. *B is the branch-package row, with one displayed branch for each constructor in I;*
5. *D is the structural descent ledger recording that every recursive subcall is made only on constructor-subdata admitted by I;*
6. *O is the generated output row whose public name is being certified by the accepted recursor call;*
7. *H records componentwise* hsame *transports for the signature, eliminator, motive, branch package, descent ledger, output row, provenance row, and local naming row;*
8. *C records the displayed* Cont *routes by which consumers read the signature, eliminator, motive, branch, descent, output, audit, and naming coordinates;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, *the kernel-accepted signature, emitted eliminator, branch-completeness check, structural descent check, purity audit boundary, and neighboring generator certificates;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RecursionAuthorizationLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $I, R, M, B, D, O, H, C, P, N$.

Theorem 710.2 (RecursionAuthorizationLedger NameCert obligations)**.** *For the carrier of Definition 710.1, the local* $\mathsf{NameCert}_{\mathsf{RecursionAuthorizationLedger}^{\uparrow}}$ *surface consists of carrier admission for*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

kernel-acceptance of the constructor signature I, emitted-eliminator exactness for R, motive admission for M, branch completeness for B, structural descent exactness for D, output-row exposure for O, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, local naming, and purity-audit non-escape.*

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 710.1. The signature, emitted eliminator, motive, branch package, descent ledger, output row, transport row, continuation row, provenance row, and local naming row are all coordinates of one finite packet.

The kernel-facing obligations are the accepted signature and emitted recursor rows. The recursor-as-authorized-generator vision supplies the boundary: a constructor signature is usable only after kernel acceptance, and the eliminator row is the kernel output for that signature rather than a separate recursion principle.

Branch completeness and structural descent are read from B and D. The branch package is indexed by the displayed constructors of I, and every recursive call recorded by D descends through constructor-subdata accepted by the same signature.

The output and audit obligations are the remaining rows. Consumers reach O only along C, transport it only through H, and cite only the provenance in P and N. The source gate, build gate, and purity-audit boundary named by P exclude axiom channels and forbidden host principles from the ledger surface. □

Theorem 710.3 (RecursionAuthorizationLedger generator-closure sibling route)**.** *For an accepted* $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

the accepted signature row I, emitted eliminator row R, branch package B, and structural descent ledger D are the finite authorization ledger consumed by chapter 486 and chapter 553. The consumer route also transports, exposes, packages, and names that ledger through H, C, P, N, but it exports no unrestricted recursion principle and no eliminator outside the accepted signature.

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier generator closure sibling route`

Proof. Project the carrier tuple $L = (I, R, M, B, D, O, H, C, P, N)$ from Definition 710.1. The rows I, R, B, D are the signature, emitted eliminator, branch package, and structural descent data that authorize the generated output row.

Use Definition 710.2. Kernel acceptance, emitted-eliminator exactness, branch completeness, and structural descent exactness are obligations of the same finite packet, so a generator-closure consumer reads only those rows before inspecting the output route.

Finally route the sibling consumers through the displayed structural rows H, C, P, N. These rows transport, expose, package, and name the same finite ledger for chapter 486 and chapter 553; they do not add an unrestricted recursion principle or a second eliminator. □

Theorem 710.4 (RecursionAuthorizationLedger signature acceptance)**.** *In an accepted* RecursionAuthorizationLedg *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

the emitted eliminator row R and the branch package B are readable only after the same accepted inductive signature row I. No unlisted constructor row can enter the ledger.

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedger signature acceptance`

Proof. Project the signature, eliminator, and branch coordinates I, R, B from Definition 710.1. The carrier records R as emitted from I and B as indexed by the constructor inventory displayed by I.

Apply Definition 710.2. Kernel-acceptance of I, emitted-eliminator exactness for R, and branch completeness for B are local obligations of the same finite packet.

The remaining coordinates M, D, O, H, C, P, N do not add constructor rows. They check the motive, descent, output, transport, replay, provenance, and local naming of the accepted packet, so an unlisted constructor has no route into the ledger. □

Theorem 710.5 (RecursionAuthorizationLedger output audit boundary)**.** *For an accepted* RecursionAuthorizationL *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

the output row O is exposed only through the structural descent row D, componentwise hsame *transport H, displayed* Cont *routes C,* Pkg *provenance P, and local* NameCert *row N. The packet exports no unrestricted recursion principle and no hidden eliminator.*

Proof. Project the descent, output, and audit coordinates D, O, H, C, P, N. The carrier makes O a generated output row whose recursive calls are certified by D and whose public reads are transported, replayed, packaged, and named through H, C, P, N.

Use Definition 710.2 for output-row exposure and purity-audit non-escape. Then use Definition 710.6 to keep the output tied to the displayed branch package and structural descent ledger.

Since the carrier has no coordinate for an unrestricted recursion principle or a second eliminator, the output audit boundary is exactly the route through D, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier output audit boundary`

Theorem 710.6 (RecursionAuthorizationLedger branch completeness and descent exactness)**.** *For an accepted* RecursionAuthorizationLedger$^{\uparrow}$ *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

every constructor row displayed by I has exactly one branch row in B, and every recursive call certified by the output row O is justified by the structural descent ledger D. Thus the exported ledger contains no omitted branch and no non-structural recursive call.

Proof. Project the signature and branch coordinates I and B from the carrier. The inventory in Definition 710.1 records B as the branch-package row with one displayed branch for each constructor in I.

Project the descent and output coordinates D and O. The same carrier records D as the structural descent ledger and O as the generated output row whose recursive calls are certified by that ledger.

Use the NameCert obligations of Definition 710.2. Branch completeness and structural descent exactness are local obligations for the same finite packet, so no consumer can read O without the constructor-indexed branch package and the descent rows that justify its recursive subcalls.

The remaining rows H, C, P, N only transport, expose, package, and name those displayed coordinates. Since they add no branch coordinate and no recursion coordinate, the public ledger exports neither an omitted branch nor a non-structural recursive call. □

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerPacket branch descent exactness`

Theorem 710.7 (RecursionAuthorizationLedger motive-output route)**.** *For an accepted* RecursionAuthorizationLed *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

the motive row M and generated output row O are consumed only through the finite ledger. In particular, a public output read first uses the accepted signature I, the emitted eliminator R, the checked motive M, the branch package B, and the descent ledger D, and only then exposes O through H, C, P, N.

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerMotiveOutputRoute`

Proof. Project the carrier. The rows I, R, M, B, D, O occur in the displayed order of the ledger packet, with M recorded as the checked motive for the emitted eliminator and O recorded as the generated output row.

Use Definition 710.2. Motive admission, branch completeness, structural descent exactness, and output-row exposure are all local obligations for the same finite packet.

Finally apply Definition 710.6. The output row cannot be read apart from the branch package and descent ledger that justify the recursive subcalls, so the motive-output route stays inside L. □

Theorem 710.8 (RecursionAuthorizationLedger descent transport stability)**.** *For an accepted* $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

componentwise hsame *transport preserves the structural descent ledger D, the branch package B, and the output row O as one finite authorization route. A transported read cannot replace D by a non-structural recursion row or replace O by a hidden eliminator output.*

Proof. Read the transport row H from the carrier. Its componentwise clauses cover the signature, emitted eliminator, motive, branch package, descent ledger, output row, provenance row, and local naming row.

Apply the NameCert obligations. Structural descent exactness is attached to D, branch completeness is attached to B, and output-row exposure is attached to O, all within the same packet L.

The continuation and provenance rows C, P, N replay and name the transported route. Since the carrier has no coordinate for a non-structural recursive call or a second eliminator, transport preserves the finite descent-output surface rather than producing an escape route. □

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier descent transport stability`

Theorem 710.9 (RecursionAuthorizationLedger audit provenance exhaustion)**.** *For an accepted* $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ *packet*

$$L = (I, R, M, B, D, O, H, C, P, N),$$

all audit provenance for the signature, motive, descent, output, and purity boundary is exhausted by the rows P and N together with the displayed read routes C. The ledger exports no unlisted constructor, no omitted branch, no non-structural recursive call, no hidden eliminator, and no output provenance outside L.

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier audit provenance exhaustion`

Proof. Project P, N and the continuation row C. The carrier describes P as provenance over the accepted signature, emitted eliminator, branch completeness check, structural descent check, purity audit boundary, and neighboring generator certificates.

Use Definition 710.2. Package provenance, local naming, displayed route replay, and purity-audit non-escape are obligations of the same finite packet as I, R, M, B, D, O, H.

Apply the branch-completeness theorem and the non-escape theorem. They rule out omitted branches, non-structural recursive calls, hidden eliminators, unrestricted recursion principles, and outside audit channels, so the provenance ledger is exhausted by C, P, N over L. □

Theorem 710.10 (RecursionAuthorizationLedger non-escape)**.** *Every consumer read of an accepted* $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ *packet factors through*

$$I, R, M, B, D, O, H, C, P, N.$$

It exports no unlisted constructor, no omitted branch, no branch outside the accepted signature, no non-structural recursive call, no hidden eliminator, no unrestricted recursion principle, and no output whose dependency route escapes the build, source-audit, and purity-audit boundary recorded in the packet.

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier consumer nonescape`

Proof. Start with the obligation theorem. It places the generated output downstream of the accepted signature, emitted eliminator, motive, branch package, structural descent ledger, transport rows, continuation routes, provenance, and local naming row.

For constructors and branches, inspect I and B. The visible signature supplies the only constructor inventory, and the branch package supplies exactly the branch rows available to the eliminator call.

For recursion, inspect D. A recursive subcall can be consumed only when the descent ledger records it as constructor-subdata of the accepted signature. There is no coordinate for a non-structural recursive call or a second hidden eliminator.

Finally inspect P and N. They record the build, source-audit, and purity-audit boundary for the named output. Since the carrier contains no axiom channel, unrestricted recursion principle, or forbidden host dependency row, the public ledger cannot export any of those objects. □

Theorem 710.11 (RecursionAuthorizationLedger finite induction scope)**.** *Every accepted* RecursionAuthorizationLe *packet*

$$L = (I, R, M, B, D, O, H, C, P, N)$$

whose branch ledger is generated by the displayed BHist *constructors exposes base authorization, successor authorization, signature acceptance, branch completeness, structural descent exactness, output exposure, audit boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *on one finite induction route. The route exports no hidden recursor, host recursion principle, external termination oracle, or branch outside the accepted signature.*

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerPacket finite induction scope`

Proof. Induct over the displayed finite branch ledger B. The base case reads the seed authorization row together with the accepted signature I and emitted eliminator R, so the first branch is authorized only inside the carrier L.

For the successor step, replay the same Cont and signature rows. The new branch is admitted only after the constructor row from I is visible, and branch completeness keeps it inside the displayed package B.

Structural descent is preserved by the row D. Each recursive subcall remains constructor-subdata licensed by the accepted signature, so the induction cannot introduce host recursion or an external termination oracle.

Finally expose O, H, C, P, N. These rows carry output exposure, componentwise transport, replay, provenance, audit boundary, and local naming for the same route, and they contain no coordinate for a hidden recursor or any branch outside the accepted signature. □

Lean4 ✓ `BEDC.Derived.RecursionAuthorizationLedgerUp.RecursionAuthorizationLedgerCarrier finite induction scope`

Closure status for $\mathsf{NameCert}_{\mathsf{RecursionAuthorizationLedger}\uparrow}$.

Constructive story: A $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$ packet is generated from an accepted inductive signature, emitted eliminator, checked motive, complete branch package, structural descent ledger, output row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RecursionAuthorizationLedger}\uparrow}$, scopedClosed)

Scope closed: Definition 710.1, Definition 710.2, Definition 710.4, Definition 710.5, Definition 710.7, Definition 710.8, Definition 710.9, Definition 710.10, and Definition 710.11 bind the carrier, NameCert obligations, signature acceptance, output audit boundary, motive-output route, descent transport stability, provenance exhaustion, nonescape, and finite-induction scope for accepted recursor calls.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RecursionAuthorizationLedger}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, global kernel soundness, all positivity checking, all termination checking, bridge checking, or any external classical principle.

Upgrade path: Public closure requires a consumer-facing route that exposes accepted signatures, generated outputs, audit provenance, and finite induction as a stable public surface without adding Lean verification or bridge checking.

Chapter 711

A Concrete Naming Certificate for $\mathsf{GeneratorAuditClosure}^{\uparrow}$

The $\mathsf{GeneratorAuditClosure}^{\uparrow}$ packet names the finite audit surface by which a MetaCIC generator-closure proof becomes a BEDC-readable certificate. It is tuned to generator rows such as `substitute` and `shift`, classifier rows such as `ClosedAt` and `GeneratorClosureClassifier`, accepted closure witnesses such as `substitute_closed`, `shift_closed`, `closed_under_substitute_classifier`, and `closed_under_shift_classifier`, and the audit ledgers exposed by $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, $\mathsf{AxiomQueryLedger}^{\uparrow}$, and $\mathsf{KernelSourceChannelLedger}^{\uparrow}$.

Definition 711.1 (Generator-audit closure carrier)**.** *A* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (G, K, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *G records the generator row, including the MetaCIC operation being audited and the arity or binder-side parameters needed to replay its local action;*
2. *K records the classifier row, including the closedness classifier, the generator-closure classifier, and the slice on which the generator is claimed to preserve closure;*
3. *A records the accepted closure proof row for substitute-style or shift-style generators;*
4. *Q records the axiom-query or audit-report row attached to the accepted closure proof, with refusal content for `Classical.choice`, `Quot.sound`, `propext`, and any unaccepted host proof equality;*
5. *H records componentwise* hsame *transport for the generator row, classifier row, accepted closure proof row, audit report, replay route, provenance, and local name;*
6. *C records the displayed* Cont *routes that replay generator evaluation, classifier reading, kernel acceptance, axiom-query inspection, and consumer access to the closure certificate;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, $\mathsf{AxiomQueryLedger}^{\uparrow}$, *and* $\mathsf{KernelSourceChannelLedger}^{\uparrow}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorAuditClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden proof production, quotient transport, classical case splitting, proposition extensionality, host proof equality, or a claim that a source generator is trusted without the accepted closure row A and audit row Q.

Lean4 ✓ `BEDC.Derived.GeneratorAuditClosureUp.GeneratorAuditClosureTasteGate single carrier alignment`

Theorem 711.2 (Generator-audit closure NameCert obligations)**.** *For every accepted* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *carrier*

$$W = (G, K, A, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GeneratorAuditClosure}^{\uparrow}}$ *surface consists exactly of generator exposure through* G*, classifier exposure through* K*, accepted closure-witness exposure through* A*, axiom-query or audit-report exposure through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

The obligations are anchored by the kernel-acceptance witness for A*, the axiom-query ledger for* Q*, and the source-channel ledger separating the generator row from the accepted proof row.*

Lean4 ✓ `BEDC.Derived.GeneratorAuditClosureUp.GeneratorAuditClosureCarrier namecert obligations`

Proof. Project the finite packet of Definition 711.1. The rows G, K, A, Q, H, C, P, N are the complete displayed inventory, so carrier admission and local naming are read directly from the packet.

The generator and classifier side is exhausted by G and K. These rows record the MetaCIC operation and the closure classifier slice before the accepted proof row is exposed.

The audit side is exhausted by A and Q. The row A is read through $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, while Q is read through $\mathsf{AxiomQueryLedger}^{\uparrow}$ and may also be placed inside the channel separation supplied by $\mathsf{KernelSourceChannelLedger}^{\uparrow}$.

Transport, replay, provenance, and naming are confined to H, C, P, N. Since the carrier has no further row, the listed obligations are exactly the local NameCert surface. □

Theorem 711.3 (Generator-audit closure MetaCIC sibling route)**.** *For every accepted* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *carrier* $W = (G, K, A, Q, H, C, P, N)$*, the rows* A*,* Q*, and* P *are read through the concrete MetaCIC sibling ledgers. The accepted-declaration surface for* A *is the surface of chapter 549, the axiom-query ledger for* Q *is the ledger of chapter 654, and the source-channel ledger carried by* P *is the ledger of chapter 671. Thus the generator-audit packet is not an isolated kernel row.*

Lean4 ✓ `BEDC.Derived.GeneratorAuditClosureUp.GeneratorAuditClosureCarrier metacic sibling route`

Proof. Project the carrier of Definition 711.1. The row A is displayed as an accepted closure proof row, Q as an axiom-query or audit-report row, and P as provenance over the sibling acceptance, query, and channel ledgers.

Apply Definition 711.2. It identifies A, Q, and P as part of the local NameCert surface rather than as informal side data.

The named sibling chapters supply the concrete route: chapter 549 reads the accepted-declaration surface, chapter 654 reads the axiom-query ledger, and chapter 671 reads the source-channel ledger. Since these rows are projected from the same finite packet, the generator-audit certificate consumes the MetaCIC sibling ledgers and cannot stand as an isolated kernel row. □

Theorem 711.4 (Generator-audit substitution and shift audit)**.** *When* G *is the MetaCIC* `substitute` *or* `shift` *generator row and* K *is the corresponding* `GeneratorClosureClassifier` *row, an accepted* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *packet routes the closure claim through the displayed accepted proof row* A *for the substitute or shift closure witness and the audit row* Q *for that proof.*

Thus the paper-side generator-closure reading is available only as a finite certificate over G, K, A, Q, H, C, P, N*; it is not a hidden proof producer and it does not identify source syntax, host proof equality, quotient equality, or classical branching with kernel acceptance.*

Lean4 ✓ `BEDC.Derived.GeneratorAuditClosureUp.GeneratorAuditClosureCarrier substitution shift audit`

Proof. Begin with the generator row G. In the substitution case it records the `substitute` operation and the closed-boundary slice; in the shifting case it records the `shift` operation and the corresponding closed-boundary slice.

Read the classifier row K next. It is the displayed `GeneratorClosureClassifier` instance used by the MetaCIC closure witnesses named in the kernel-audit witness chapter.

The accepted proof row A supplies the kernel-accepted closure witness, such as `substitute_closed`, `shift_closed`, `closed_under_substitute_classifier`, or `closed_under_shift_classifier`. The row Q supplies the axiom-query or audit report for that same accepted proof row.

Finally H, C, P, N transport, replay, package, and name the same finite route. No carrier coordinate remains for hidden proof production, quotient equality, classical branching, proposition extensionality, or host proof equality, so the substitution or shift audit is confined to the displayed packet. □

Theorem 711.5 (Generator-audit closure non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *packet factors through the finite rows*

$$G, K, A, Q, H, C, P, N.$$

The consumer may read the generator, classifier, accepted closure proof, audit report, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, and it cannot read a hidden kernel-soundness theorem, a source-generator trust token, a quotient proof transport, a classical choice step, proposition extensionality, or host proof equality.*

Lean4 ✓ `BEDC.Derived.GeneratorAuditClosureUp.GeneratorAuditClosureCarrier consumer nonescape`

Proof. Project the accepted packet $W = (G, K, A, Q, H, C, P, N)$. The definition lists exactly these rows as the exported carrier.

Use Definition 711.2 to account for generator exposure, classifier exposure, accepted closure-witness exposure, audit-report exposure, transport, replay, provenance, and local naming.

Use Definition 711.4 for substitute-style and shift-style consumers. It routes the closure claim through the MetaCIC generator row, the generator-closure classifier row, the accepted closure proof row, and the audit row before any consumer-facing read is available.

Since the carrier contains no further coordinate, the listed rows exhaust the consumer surface and exclude hidden proof production, quotient transport, classical choice, proposition extensionality, and host proof equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorAuditClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{GeneratorAuditClosure}^{\uparrow}$ packet is generated from a MetaCIC generator row, a generator-closure classifier row, an accepted closure proof row, an axiom-query or audit-report row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorAuditClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 711.1, Definition 711.2, Definition 711.4, and Definition 711.5 specify the finite generator-audit closure carrier, NameCert obligation surface, substitution and shift audit route, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratorAuditClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove new MetaCIC closure theorems, global Lean kernel soundness, quotient or classical proof transport, proposition extensionality, host proof equality, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier projection, accepted-proof audit replay, axiom-query refusal stability, and consumer non-escape for the displayed generator-audit packet.

Chapter 712

A Concrete Naming Certificate for MetaCICSubjectReductionObligation$^{\uparrow}$

The MetaCICSubjectReductionObligation$^{\uparrow}$ packet turns the four MetaCIC subject-reduction discharge faces into a finite obligation-boundary certificate. It records the beta-substitution, application-argument, lambda-domain, and Pi-domain obligations beside the parameterised subject-reduction theorem row, the checked closed-consistency obstruction row, audit-query provenance, transport, refusal, and local naming data. The packet is the finite boundary read by downstream audit consumers; it is not a closed subject-reduction theorem.

Definition 712.1 (MetaCIC subject-reduction obligation carrier)**.** *A* MetaCICSubjectReductionObligation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, A, L, P, Q, R, H, C, G, N)$$

with the following displayed rows:

1. *B records* BetaSubstitutionPreservation*, the beta-redex substitution obligation consumed by the direct beta case;*
2. *A records* AppArgTypeStable*, the application-argument codomain-stability obligation;*
3. *L records* LamDomainSubjectReduction*, the lambda-domain subject-reduction obligation;*
4. *P records* PiDomainSubjectReduction*, the Pi-domain subject-reduction obligation;*
5. *Q records the parameterised theorem row* BEDC.MetaCIC.subject_reduction *and its four visible input sockets B, A, L, P;*
6. *R records the checked closed-consistency obstruction row* BEDC.MetaCIC.closed_consistency_reduction_obstructi
7. *H records componentwise* hsame *transport for the four obligation rows, the parameterised theorem row, the obstruction row, the audit-query row, the refusal row, the package row, and the local name;*
8. *C records the displayed* Cont *routes by which consumers replay accepted subject-reduction reads only through Q after all four obligation rows are supplied;*
9. *G records* Pkg *provenance for the audit query, the kernel-acceptance witness surface, the* GAP *refusal for closed subject reduction, and the local* NameCert *row;*
10. *N is the local* NameCert$_{\text{MetaCICSubjectReductionObligation}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unparameterised closed subject-reduction theorem, a host conversion rule, an erased derivation semantics, a quotient equality of MetaCIC terms, or a hidden normalization theorem.

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionObligationUp.MetaCICSubjectReductionObligationTasteGate single carrier alignment`

Theorem 712.2 (Four-way subject-reduction obligation boundary)**.** *Let $O = (B, A, L, P, Q, R, H, C, G, N)$ be an accepted* MetaCICSubjectReductionObligation$^{\uparrow}$ *carrier. The packet exposes all four discharge faces B, A, L, P simultaneously before any closed subject-reduction consumer can read the parameterised theorem row Q. A consumer that does not display all four faces is routed to the obstruction row R and the* GAP *refusal recorded in G.*

Proof. Project the carrier of Definition 712.1. The only rows naming discharge data are B, A, L, P, and the only subject-reduction theorem row is Q.

The accepted read surface is the one isolated by SubjectReduction$^{\uparrow}$ in chapter 483. There, a public subject-reduction read must pass through the displayed ledger, transport, and continuation routes rather than through an ambient typing theorem.

For this MetaCIC packet, the ledger row Q records the same discipline at the four-input interface. It is readable as a preservation row only after the beta-substitution, application-argument, lambda-domain, and Pi-domain rows have all been displayed.

Audit replay is accounted for by the KernelAcceptanceAuditWitness$^{\uparrow}$ surface of chapter 547. The provenance row G therefore records the query and replay rows as finite audit data, not as an extra proof source.

If one of the four faces is absent, the checked obstruction principle Definition 9377.5 supplies the refusal boundary. The row R and the GAP entry in G are then the consumer-facing answer, so the packet exposes the four-way boundary before any preservation read can proceed. □

Theorem 712.3 (MetaCIC subject-reduction obligation non-escape)**.** *Every downstream consumer of an accepted* MetaCICSubjectReductionObligation$^{\uparrow}$ *packet factors through*

$$B, A, L, P, Q, R, H, C, G, N.$$

The consumer may inspect the four obligation rows, the parameterised subject-reduction row, the checked obstruction row, audit-query provenance, componentwise hsame *transport, displayed* Cont *replay,* GAP *refusal, and local naming. It cannot turn the packet into an unparameterised closed subject-reduction theorem, a host conversion rule, or a hidden normalization theorem.*

Proof. Start with the finite inventory. The carrier displays exactly the rows $B, A, L, P, Q, R, H, C, G, N$, so every public read must be routed through one of these coordinates.

The preservation-facing route is constrained by Definition 712.2. It reaches Q only through the four displayed discharge rows and through the transport and continuation rows H, C.

The audit-facing route is constrained by chapter 547. Its query, replay, and dependency rows are finite provenance data in G, and they do not add a closed MetaCIC preservation theorem.

The refusal-facing route is constrained by Definition 9377.5. The obstruction row R and the GAP row in G answer closed-subject-reduction requests without exporting the closed theorem.

Since no carrier coordinate names host conversion, erased derivation semantics, quotient equality, or global normalization, none of those objects can be read from the packet. Thus the downstream surface remains the displayed finite obligation boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}$ packet is generated from the four MetaCIC discharge-obligation rows, the parameterised subject-reduction theorem row, the checked obstruction row, audit-query provenance, componentwise hsame transports, displayed Cont replay, GAP refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 712.1, Definition 712.2, and Definition 712.3 bind the finite four-obligation carrier, classifier boundary, audit replay rows, refusal row, and downstream non-escape surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, host conversion, global normalization, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-face exposure, parameterised theorem consumption, obstruction replay, refusal exactness, provenance accounting, transport stability, and non-escape for the same finite packet.

Chapter 713

A Concrete Naming Certificate for $\mathsf{AxisCarryConfluence}^{\uparrow}$

The $\mathsf{AxisCarryConfluence}^{\uparrow}$ interface packages the finite local confluence surface behind overlapping Axis-Zeckendorf carry routes. It records two displayed carry-normalization paths from overlapping source windows to a shared normal target, their common value ledger, the non-hsame boundary for every carry edge, componentwise transports, continuation routes, provenance, and a local naming row. The packet is a finite certificate for route agreement at the displayed window; it is not a kernel rewrite, a quotient by history sameness, or an unbounded normalizer.

The confluence packet consumes the local carry edge of chapter 544 and the normalization handoff of chapter 608. Its two route rows r_0, r_1 are not free data: each is a $\mathsf{ZCarry}^{\uparrow}$ edge followed by a $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ handoff into the shared target row n.

Definition 713.1 (Axis carry-confluence carrier)**.** *An* $\mathsf{AxisCarryConfluence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N)$$

with the following visible coordinates:

1. *u is the displayed overlapping source-window history row;*
2. *v is the intermediate history row reached by the left carry route from u;*
3. *w is the intermediate history row reached by the right carry route from u;*
4. *n is the common normal target row admitted by* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$*;*
5. *r_0 is the left route row, recording a* $\mathsf{ZCarry}^{\uparrow}$ *edge from u to v and a* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *handoff from v to n;*
6. *r_1 is the right route row, recording a* $\mathsf{ZCarry}^{\uparrow}$ *edge from u to w and a* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *handoff from w to n;*
7. *t is the value ledger row, recording that the displayed Fibonacci value read from u, v, w, and n is the same finite sum;*
8. *H records the non-*hsame *boundary for the two carry edges together with componentwise* hsame *transports preserving u, v, w, n, r_0, r_1, t, the route row, the package row, and the local naming row;*

9. *C stores the displayed* Cont *routes for reading the source window, the two intermediate windows, the common normal target, both carry-normalization routes, the value ledger, the non-*hsame *boundary, provenance, and local naming;*
10. *P is the* Pkg *provenance over* BHist, ZCarry$^{\uparrow}$, ZeckendorfCarryNormalization$^{\uparrow}$, *Fibonacci value ledgers,* hsame, Cont, Pkg, *and local* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisCarryConfluence}^{\uparrow}}$ *row.*

The carrier exports only this finite route packet. It contains no source-target hsame *proof for either carry edge, no global confluence theorem for arbitrary windows, no kernel rewrite rule, no host arithmetic quotient, and no hidden normalizing algorithm.*

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceTasteGate single carrier alignment`

Theorem 713.2 (Axis carry-confluence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisCarryConfluence}^{\uparrow}}$ *surface for Definition 713.1 consists of carrier admission for*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N),$$

two displayed ZCarry$^{\uparrow}$ *source edges, two* ZeckendorfCarryNormalization$^{\uparrow}$ *route handoffs into the common target n, exactness of the value ledger t, preservation of the non-*hsame *boundary under the displayed transports in H,* Cont *route locality through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 713.1. The rows $u, v, w, n, r_0, r_1, t, H, C, P, N$ are the only data admitted to the packet.

The two route rows give the carry-normalization shape. The row r_0 exposes the left carry edge and its handoff to the normal target, while r_1 exposes the right carry edge and its handoff to the same target.

The value and boundary clauses are separate projections. The row t records the common finite Fibonacci value, and H records that the carry edges remain outside source-target hsame while their displayed rows transport componentwise.

Finally C, P, N route, package, and name the same finite surface. Since the carrier has no coordinate for a kernel rewrite, host quotient, or unbounded normalizer, the listed obligations exhaust the local certificate. □

Theorem 713.3 (Axis carry-confluence carrier obligation)**.** *The carrier-admission obligation for* AxisCarryConfluence$^{\uparrow}$ *is the admission of the concrete tuple*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N).$$

Here u is the overlapping source window, v and w are the two intermediate carry windows, n is the shared normal target, r_0 and r_1 are the two ZCarry$^{\uparrow}$*-then-*ZeckendorfCarryNormalization$^{\uparrow}$ *route rows, t is the value ledger, H records the non-*hsame *boundary and componentwise transports, C records displayed continuations, P records provenance, and N records the local NameCert row.*

Proof. Project Definition 713.1. Its displayed tuple is already the full carrier-admission surface.

The sibling references to ZCarry$^{\uparrow}$ and ZeckendorfCarryNormalization$^{\uparrow}$ determine the route type of r_0 and r_1: each route begins with one carry edge and ends with a normalization handoff. The remaining rows t, H, C, P, N are the ledger, boundary, replay, provenance, and naming supports needed for the same admitted tuple. □

Theorem 713.4 (Axis carry-confluence local-join obligation)**.** *For an accepted* AxisCarryConfluence$^{\uparrow}$ *carrier*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N),$$

the local-join obligation is that r_0 and r_1 share the target n while preserving the value ledger t. Thus the left route from u through v and the right route from u through w are consumer-visible only as two ZCarry$^{\uparrow}$ *edges with* ZeckendorfCarryNormalization$^{\uparrow}$ *handoffs into the same normal row.*

Proof. Project the two route rows. The row r_0 names the left carry edge and its handoff into n, while r_1 names the right carry edge and its handoff into the same n.

Project the ledger row t separately. It records the common finite Fibonacci value across u, v, w, n, so the local join is value-preserving without adding an hsame identification between source and intermediate rows. □

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceCarrier local join obligation`

Theorem 713.5 (Axis carry-confluence normal-form obligation)**.** *Every consumer of the shared target n in an accepted* AxisCarryConfluence$^{\uparrow}$ *packet must pass through the displayed* ZeckendorfCarryNormalization$^{\uparrow}$ *handoffs and the non-*hsame *boundary rows. The packet permits reads of n only together with r_0, r_1, t, H, C, P, N; it does not export a direct normal-form oracle, a kernel rewrite, a host quotient, or source-target* hsame *for either carry edge.*

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceCarrier normal form obligation`

Proof. Read n through the route rows. By the carrier definition, the only routes to n inside the packet are the two handoffs named by r_0 and r_1, each supported by the ZeckendorfCarryNormalization$^{\uparrow}$ sibling.

The boundary row H remains attached to those handoffs. It records that the carry edges are not source-target hsame identifications, while C, P, N replay, package, and name the same finite normal-target read.

Since no carrier coordinate names a direct normal-form oracle, kernel rewrite, host quotient, or source-target hsame, a consumer of n must stay inside the displayed handoff and boundary surface. □

Theorem 713.6 (Axis carry-confluence local diamond)**.** *For an accepted* AxisCarryConfluence$^{\uparrow}$ *carrier*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N),$$

the two displayed routes form a local diamond at the finite carry window: u reaches v and w by the two recorded ZCarry$^{\uparrow}$ *edges, and both v and w reach the same normal target n through the recorded* ZeckendorfCarryNormalization$^{\uparrow}$ *handoffs. A consumer-visible diamond read factors through the route rows r_0, r_1, the common target row n, the value ledger t, and the support rows H, C, P, N.*

Proof. Project the accepted carrier. The row r_0 supplies the left route from u through v to n, and r_1 supplies the right route from u through w to the same n.

Read the common target and value ledger next. The coordinate n is shared by both route rows, and t records that the visible Fibonacci value is unchanged along the four displayed histories.

The support rows then delimit the consumer read. Transport through H, routing through C, provenance through P, and naming through N expose exactly this finite diamond and do not turn it into a theorem about all carry overlaps. □

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceCarrier local diamond`

Theorem 713.7 (Axis carry-confluence value-ledger exhaustion)**.** *For an accepted* $\mathsf{AxisCarryConfluence}^{\uparrow}$ *carrier*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N),$$

any consumer-visible comparison between the two carry-normalization routes must factor through the common target row n*, the value ledger* t*, the non-*hsame *boundary* H*, the route replay row* C*, provenance* P*, and the local name* N*. No additional equality row, quotient row, host arithmetic row, or global normalizer row is exported by the packet.*

Proof. Project the accepted carrier. The route rows r_0 and r_1 are the only displayed routes, and both route rows end at the same normal target coordinate n.

Read the value ledger separately. The row t is the sole public record that the two routes preserve the same finite Fibonacci value across u, v, w, n; it is not replaced by an hsame proof or by a host quotient.

The boundary row H blocks a source-target hsame reading for the carry edges, while C, P, N keep every comparison inside the displayed continuation, provenance, and local NameCert rows. Since the carrier has no further coordinate, a consumer-visible agreement has no route outside n, t, H, C, P, N. □

Theorem 713.8 (Axis carry-confluence non-escape)**.** *For an accepted* $\mathsf{AxisCarryConfluence}^{\uparrow}$ *carrier, the confluence reading cannot escape the finite packet. The rows* r_0, r_1, t, H, C, P, N *may expose two carry-normalization routes, their shared normal target, the common value ledger, componentwise transport, route replay, provenance, and local naming, but they do not export* $\mathsf{hsame}(u, v)$*,* $\mathsf{hsame}(u, w)$*, a kernel rewrite, a host arithmetic quotient, or a global confluence principle for arbitrary carry windows.*

Proof. The possible consumer reads are the coordinates of Definition 713.1. The route rows expose carry-normalization paths, and the value row exposes the common finite ledger.

The boundary row H keeps the two carry edges separate from source-target hsame. It transports only the named components of the accepted packet and supplies no equality proof between the source and either intermediate carry target.

The remaining support rows C, P, N expose, package, and name the same finite object. Because no coordinate records a kernel rewrite, host quotient, or arbitrary-window confluence theorem, no projection from the packet can export one. □

Theorem 713.9 (Axis carry-confluence normalization handoff)**.** *For an accepted* $\mathsf{AxisCarryConfluence}^{\uparrow}$ *carrier*

$$K = (u, v, w, n, r_0, r_1, t, H, C, P, N),$$

the public normalization handoff is the joint read of the shared target row n*, the route rows* r_0, r_1*, and the value ledger* t*. Both carry routes hand off to the same* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *target, and every consumer read of that handoff must pass through the displayed* Cont *routes,* Pkg *provenance, non-*hsame *boundary, and local* $\mathsf{NameCert}$ *row.*

Proof. Project n, r_0, r_1, t from the accepted packet. The target row n is the only normal target coordinate, and both route rows name it as their handoff endpoint.

Use the value ledger to align the two paths. The row t records the common finite Fibonacci value across the source, intermediate rows, and target, so the handoff is value-ledger preserving rather than an identification by history sameness.

Finally replay the handoff through H, C, P, N. The non-hsame boundary remains visible, and the route, provenance, and naming rows keep the handoff inside the finite NameCert surface. □

Lean4 ✓ `BEDC.Derived.AxisCarryConfluenceUp.AxisCarryConfluenceCarrier normalization handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{AxisCarryConfluence}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisCarryConfluence}^{\uparrow}$ packet is generated from one overlapping source window, two displayed carry-normalization routes, a shared normal target, a finite value ledger, non-hsame boundary rows, componentwise transports, continuation routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisCarryConfluence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 713.1, Definition 713.2, Definition 713.6, Definition 713.8, and Definition 713.9 bind the finite two-route carry-confluence packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxisCarryConfluence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, a bridge certificate, arbitrary-window carry confluence, a global Zeckendorf normalizer, a kernel rewrite, host arithmetic quotienting, or source-target hsame.

Upgrade path: Obligation closure requires checked carrier admission, two route handoffs, local diamond reading, value-ledger exactness, non-hsame preservation, route locality, provenance, local naming, and normalization handoff for the same finite packet.

Chapter 714

A Concrete Naming Certificate for $\mathsf{TypeLevelSocketExposure}^{\uparrow}$

The $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ packet realizes type-level apophatic exposure as a finite NameCert object. It records the setup-field surface where an outside supply is requested, the carrier and classifier rows that make the request visible, the provenance and refusal-ledger rows that prevent hidden internalization, and the local naming row that downstream audit gates can cite.

The refusal-ledger row R is the type-level specialization of the negative-name boundary in chapter 629. This chapter keeps the setup-field row, carrier-shape row, and classifier rows visible so that the same refusal discipline is read as a typed socket surface rather than as an anonymous external-supply boundary.

Definition 714.1 (Type-level socket exposure carrier)**.** *A* $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, A, K, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the setup-field row naming the type-level boundary at which an outside supply is requested;*
2. *A is the admitted carrier row for the requested supply shape;*
3. *K is the classifier row distinguishing visible socket use from generated substrate content;*
4. *L is the ledger row recording whether the requested supply is exposed at the setup field, carrier, classifier, provenance, or* GAP *position;*
5. *R is the refusal-ledger row excluding hidden axiom channels, forbidden host dependencies, and unmarked premise traffic;*
6. *H records componentwise* hsame *transports for the setup field, carrier, classifier, ledger, refusal row, provenance row, and local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the setup field, carrier, classifier, ledger, refusal, provenance, and naming coordinates;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, *setup fields, carriers, classifiers,* GAP *ledgers, axiom-purity gates, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}^{\uparrow}}$ *row.*

The carrier contains no hidden axiom, no theorem-valued outside premise, no forbidden host dependency, no unlisted setup field, and no route that turns the requested supply into generated substrate content.

Theorem 714.2 (Type-level socket exposure NameCert obligations)**.** *For the carrier of Definition 714.1, the local* $\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (S, A, K, L, R, H, C, P, N),$$

setup-field exposure through S*, carrier-shape exposure through* A*, classifier visibility through* K*, ledger placement through* L*, refusal-ledger exactness through* R*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureNamecertObligations`

Proof. Carrier admission is projection from the displayed packet E. The rows $S, A, K, L, R, H, C, P, N$ list the setup field, admitted carrier shape, classifier, ledger, refusal row, transports, continuation routes, provenance, and local name.

The exposure obligations are finite. A consumer first reads the setup-field row S, then the carrier row A, then the classifier row K that distinguishes socket use from generated substrate content. The ledger row L records the visible placement of the request.

The refusal row R excludes the channels that would hide the request: axiom admission, forbidden host dependency, and unmarked premise traffic. The rows H, C, P, N account for transport, route replay, provenance, and local naming, so the local surface is exhausted by the displayed packet. □

Theorem 714.3 (Type-level socket exposure ledger non-escape)**.** *Every consumer read of an accepted* $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ *packet factors through*

$$S, A, K, L, R, H, C, P, N.$$

It may inspect the setup-field exposure, carrier shape, classifier, ledger placement, refusal row, componentwise transports, displayed continuations, provenance, and local NameCert *row, and it cannot read a hidden axiom, a theorem-valued outside premise, a forbidden host dependency, an unlisted setup field, or an unmarked supply channel.*

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureLedgerNonescape`

Proof. Start with the obligation theorem. It identifies the accepted packet with the displayed rows $S, A, K, L, R, H, C, P, N$ and names those rows as the complete local NameCert inventory.

For type-level exposure, inspect S, A, and K. The setup field names the boundary, the carrier row records the requested supply shape, and the classifier row keeps visible socket use separate from generated substrate content.

For ledger non-escape, inspect L and R. The ledger row records the visible placement of the request, while the refusal row excludes axiom admission, forbidden host dependency, hidden premise traffic, and unlisted setup fields.

Finally inspect H, C, P, N. They provide the only transport, continuation, provenance, and naming routes available to downstream consumers. Since the carrier contains no additional row beyond those coordinates, no consumer can obtain a hidden supply channel from the packet. □

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureLedgerHostReturnExclusion`

Theorem 714.4 (Type-level socket exposure refusal exactness)**.** *For every accepted* $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ *carrier* $E = (S, A, K, L, R, H, C, P, N)$*, the refusal row* R *excludes hidden axiom channels, theorem-valued outside premises, forbidden host dependencies, unlisted setup fields, and unmarked supply channels exactly because every consumer route factors through* $S, A, K, L, R, H, C, P, N$*. No refused channel can be reconstructed from the structural rows.*

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureRefusalExactness`

Proof. Start with the carrier of Definition 714.1. The packet exposes the setup field, admitted carrier, classifier, ledger placement, refusal row, transport, continuation, provenance, and local naming as the finite tuple $S, A, K, L, R, H, C, P, N$.

Apply the NameCert obligations of Definition 714.2. They make refusal-ledger exactness through R one of the local rows rather than an external convention.

Finally use ledger non-escape from Definition 714.3. Since every consumer read factors through the displayed tuple, a hidden axiom channel, theorem-valued outside premise, forbidden host dependency, unlisted setup field, or unmarked supply channel has no carrier coordinate and no replay route through H, C, P, N. □

Theorem 714.5 (Type-level socket exposure scope exactness)**.** *For every accepted* $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ *packet* $E = (S, A, K, L, R, H, C, P, N)$*, the complete consumption scope for type-level socket exposure is the displayed setup-field row* S*, carrier-shape row* A*, classifier row* K*, ledger-placement row* L*, refusal row* R*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}^{\uparrow}}$ *row* N*. No hidden axiom channel, theorem-valued outside premise, host dependency, unlisted setup field, or unmarked supply channel has a scope coordinate outside those rows.*

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureScopeExactness`

Proof. Project the carrier of Definition 714.1. It displays exactly $S, A, K, L, R, H, C, P, N$ as setup, carrier, classifier, ledger, refusal, transport, replay, provenance, and naming coordinates.

Apply the NameCert obligations of Definition 714.2. They identify setup-field exposure, carrier-shape exposure, classifier visibility, ledger placement, refusal-ledger exactness, transport, replay, provenance, and local naming as the local obligation surface.

Use ledger non-escape from Definition 714.3 and refusal exactness from Definition 714.4. Since every consumer read factors through the displayed tuple and the refusal row excludes the hidden channels, no excluded premise can acquire a separate scope coordinate. □

Theorem 714.6 (TypeLevelSocketExposure kernel grounding)**.** *Every accepted* $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ *read is grounded by the displayed* BHist *carrier rows* $S, A, K, L, R, H, C, P, N$*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, the refusal ledger* R*, and the local* $\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}^{\uparrow}}$ *row* N*. Consequently the read cannot import outside supply, host type equality, hidden setup fields, theorem-valued outside premises, or global kernel soundness.*

Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureScopeKernelGrounding`
Lean4 ✓ `BEDC.Derived.TypeLevelSocketExposureUp.TypeLevelSocketExposureKernelGrounding`

Proof. Project the carrier of Definition 714.1. The accepted read starts with the setup-field row S, admitted carrier row A, classifier row K, ledger row L, refusal row R, transport row H, replay row C, provenance row P, and local naming row N.

Apply the NameCert obligations of Definition 714.2. Those obligations make the carrier rows, refusal ledger, transport, replay, provenance, and local naming the complete local certificate surface.

Use ledger non-escape, refusal exactness, and scope exactness from Definition 714.3, Definition 714.4, and Definition 714.5. Since every consumer route is already confined to $S, A, K, L, R, H, C, P, N$, no outside supply, host type equality, hidden setup field, theorem-valued premise, or global kernel soundness can enter through a separate coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}\uparrow}$.

Constructive story: A $\mathsf{TypeLevelSocketExposure}^{\uparrow}$ packet is generated from a setup-field exposure row, admitted carrier-shape row, classifier row, ledger-placement row, refusal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 714.1, Definition 714.2, Definition 714.3, Definition 714.4, Definition 714.5, and Definition 714.6 bind the scoped finite surface for type-level outside-supply exposure.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeLevelSocketExposure}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, existence of any outside supply, global kernel soundness, all setup-field designs, or any hidden host principle.

Upgrade path: Obligation closure requires checked carrier admission, setup-field exposure, classifier visibility, ledger placement, refusal exactness, provenance, and non-escape for the same finite packet.

Chapter 715

A Concrete Naming Certificate for $\mathsf{ApophaticGateQuestion}^{\uparrow}$

The $\mathsf{ApophaticGateQuestion}^{\uparrow}$ packet realizes the name-to-gate correspondence as a finite question object. It starts from an apophatic socket name, records the gate question that audits the named outside-supply position, keeps the refusal ledger visible, and exposes only the readback rows needed by downstream audit gates.

The socket row is consumed from chapter 536, and the named refusal row remains compatible with chapter 543. Thus the gate-question packet reads the same sibling socket-name surfaces before it exposes its audit-readback row.

Definition 715.1 (Apophatic gate question carrier)**.** *An* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the visible* $\mathsf{ApophaticSocket}^{\uparrow}$ *or* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the external supply position;*
2. *Q is the gate-question row asking whether a consumer route used the named socket, which supply kind it requested, and whether the use remained externally marked;*
3. *R is the refusal-ledger row excluding hidden axioms, forbidden host ancestry, quotient collapse, proposition-extensionality collapse, unmarked selectors, and unrecorded inscription novelty;*
4. *A is the audit-readback row that records the answer to Q as a finite ledger entry rather than as a host-level judgment;*
5. *H records componentwise* hsame *transports for the socket row, question row, refusal ledger, readback row, provenance row, and local naming row;*
6. *C records the displayed* Cont *routes by which audit, paper-Lean, axiom-purity, and conservativity consumers may read S, Q, R, A, H, P, N;*
7. *P is the* Pkg *provenance over* BHist, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, *audit gates, refusal ledgers,* hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$ *row.*

The carrier has no coordinate for the requested outside supply itself, a hidden theorem premise, an axiom channel, a host proof of semantic harmlessness, or an unmarked route that internalizes the socket.

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionTasteGate namecert row lock`

Theorem 715.2 (Apophatic gate question NameCert obligations)**.** *For the carrier of Definition 715.1, the local* $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$G = (S, Q, R, A, H, C, P, N),$$

socket-name exposure through S*, gate-question exactness through* Q*, refusal-ledger coverage through* R*, audit-readback exactness through* A*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*, dependency provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed packet G. The rows S, Q, R, A, H, C, P, N list the socket name, gate question, refusal ledger, audit readback, transports, continuations, provenance, and local certificate.

The question obligation reads S before Q. This forces the gate question to target the named socket rather than a free-form host predicate, and it records the supply kind and marked-use condition as finite rows.

The refusal and readback obligations read R and A. The refusal ledger excludes the hidden channels named in the statement, while the readback row stores the gate answer as a BEDC ledger entry.

The rows H, C, P, N provide the only transport, continuation, provenance, and local naming routes. Since the carrier has no further coordinate, the local $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$ obligations are exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier namecert obligations`

Theorem 715.3 (Apophatic gate question visible socket non-escape)**.** *Every consumer read of an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet factors through the displayed socket row, gate-question row, refusal ledger, audit-readback row, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. The read cannot export the requested outside supply, a hidden axiom, a quotient equality, a proposition-extensionality collapse, an unmarked selector, an unrecorded inscription novelty, or a host-level replacement for the gate answer.*

Proof. Start with Definition 715.2. It identifies the accepted packet with the finite rows S, Q, R, A, H, C, P, N and gives the complete local certificate inventory.

For the visible socket route, read S and Q. The socket row names the outside-supply position, and the question row asks only whether a route used that named socket and kept the use externally marked.

For non-escape, read R and A. The refusal ledger excludes hidden axiom channels, quotient collapse, proposition-extensionality collapse, unmarked selectors, and unrecorded inscription novelty; the audit-readback row prevents replacing the ledger answer by a host judgment.

Finally route through H, C, P, N. These rows account for transport, continuation, provenance, and local naming, and the carrier contains no coordinate for the excluded objects. Hence the consumer surface remains the finite visible-socket gate question. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier visible socket nonescape`

Theorem 715.4 (Apophatic gate question socket-name dependency)**.** *Every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet reads its socket row through the sibling* ApophaticSocket$^{\uparrow}$ *surface of chapter 536 and keeps its named refusal row compatible with the* ApophaticName$^{\uparrow}$ *surface of chapter 543 before the audit-readback row is available. The audit answer therefore depends on the visible socket-name route rather than on a hidden host predicate, unmarked selector, or detached outside-supply witness.*

Proof. Start with the carrier of Definition 715.1. Its first row S is explicitly the visible ApophaticSocket$^{\uparrow}$ or ApophaticName$^{\uparrow}$ row naming the external supply position, while A is only the later audit-readback row.

Apply Definition 715.3. Every consumer read of the packet factors through S, Q, R, A, H, C, P, N, and the visible socket route cannot be replaced by a hidden axiom, quotient equality, proposition-extensionality collapse, unmarked selector, or host-level gate answer.

Consequently the socket-name dependency is read before audit readback. The sibling socket and name surfaces provide the visible rows that S may reference, while A records only the finite answer to the already displayed gate question. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier socket name dependency`

Theorem 715.5 (Apophatic gate question socket-refusal coverage)**.** *Every accepted* ApophaticGateQuestion$^{\uparrow}$ *audit answer is covered by the displayed rows S, Q, R, A, H, C, P, N. The socket row S supplies the named outside-supply position, the question row Q fixes the audited request, the refusal ledger R excludes hidden axiom, quotient, proposition-extensionality, unmarked-selector, and unrecorded-inscription channels, and the readback row A records the finite answer. The remaining rows H, C, P, N transport, replay, package, and name the same route; they do not add a hidden host predicate.*

Proof. Begin with the carrier and local NameCert obligations. The carrier displays $G = (S, Q, R, A, H, C, P, N)$, and the obligation rows identify S as the socket-name row, Q as the audited question, R as the refusal ledger, and A as the finite readback.

Next apply visible socket non-escape. Every accepted read must remain inside the displayed socket, question, refusal, readback, transport, continuation, provenance, and naming rows, so hidden axioms, quotient equality, proposition-extensionality collapse, unmarked selectors, and host-level gate answers are outside the packet.

Finally use socket-name dependency. It forces the audit question to read the sibling ApophaticSocket$^{\uparrow}$ or ApophaticName$^{\uparrow}$ surface before the readback row is available. Thus the accepted gate answer is covered by the displayed socket/refusal/readback route and not by an additional host predicate. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier socket refusal coverage`

Theorem 715.6 (Apophatic gate question source exposure)**.** *Every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet exposes the visible socket-name source row S before the gate-question row Q is read. Thus the question is asked about an already displayed* ApophaticSocket$^{\uparrow}$ *or* ApophaticName$^{\uparrow}$ *surface, not about a hidden outside-supply witness, host predicate, or detached gate answer.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier source before question route`

Proof. Project the carrier from Definition 715.1. The packet order displays S as the visible socket-name source row and Q as the later question row.

Apply the NameCert obligations and socket-name dependency theorems. They force carrier admission and every audit route to read the socket-name surface before the gate question can be answered.

The non-escape theorem removes the remaining alternatives. Since the packet has no coordinate for a hidden outside-supply witness, host predicate, or detached gate answer, exposure of S is the unique entrance to Q. □

Theorem 715.7 (Apophatic gate question visible socket coverage)**.** *Every visible socket route of an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet is read through* $\mathsf{ApophaticSocket}^{\uparrow}$ *and* $\mathsf{ApophaticName}^{\uparrow}$ *before audit readback at A. The route passes through S, Q, R, H, C, P, N and then records the finite readback row A, without replacing that readback by an unmarked selector, hidden axiom, quotient collapse, or host gate answer.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier visible socket coverage`

Proof. Begin with source exposure. It places the visible socket-name row S before the question row Q and therefore before the readback row A.

Use socket-name dependency. The sibling $\mathsf{ApophaticSocket}^{\uparrow}$ and $\mathsf{ApophaticName}^{\uparrow}$ surfaces provide the only visible rows that S may read, so the audit route is already attached to those surfaces when A is recorded.

Finally apply visible socket non-escape. The route through R, H, C, P, N accounts for refusal, transport, continuation, provenance, and local naming, and prevents an unmarked selector, hidden axiom, quotient collapse, or host gate answer from substituting for the displayed readback. □

Theorem 715.8 (Apophatic gate question audit ledger exhaustion)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet, the rows R, A, H, C, P, N exhaust the audit ledger after the visible source and question rows have been read. No hidden axiom, quotient collapse, unmarked selector, or host gate answer is exported by the packet.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal audit ledger`

Proof. Read the carrier and NameCert obligations. After S and Q, the remaining audit-facing rows are exactly the refusal ledger R, audit readback A, componentwise transport H, displayed continuation C, provenance P, and local naming row N.

Apply visible socket coverage to keep the audit readback attached to the $\mathsf{ApophaticSocket}^{\uparrow}$ and $\mathsf{ApophaticName}^{\uparrow}$ route. The audit ledger therefore records a finite answer to the already displayed gate question.

Close with visible socket non-escape. Because the carrier has no coordinate for a hidden axiom, quotient collapse, unmarked selector, or host gate answer, the listed rows exhaust the audit ledger. □

Theorem 715.9 (Apophatic gate question root visible source)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet $G = (S, Q, R, A, H, C, P, N)$, every root-facing packet read must consume the visible socket-name row S before it consumes the gate-question row Q. The row S is the displayed* $\mathsf{ApophaticSocket}^{\uparrow}$ *or* $\mathsf{ApophaticName}^{\uparrow}$ *source route, and Q is only the question asked about that visible route.*

Proof. Project the carrier of Definition 715.1. Its first coordinate is the visible socket-name row S, and the gate-question row Q is the next displayed row of the same finite packet.

Apply the local NameCert obligations, source exposure, and visible socket coverage. Theorems 715.2, 715.6, and 715.7 keep every accepted read attached to the visible $\mathsf{ApophaticSocket}^{\uparrow}$ or $\mathsf{ApophaticName}^{\uparrow}$ route before audit readback begins.

Finally use audit ledger exhaustion. Since all audit-facing rows occur after S, Q, a root-facing packet read cannot obtain the gate question before reading the visible source row. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root visible source`

Theorem 715.10 (Apophatic gate question root refusal audit ledger)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the rows*

$$R, A, H, C, P, N$$

are the complete refusal and audit ledger after the visible socket-name row S *and gate-question row* Q *have been read. No other refusal channel, hidden outside-supply predicate, detached audit answer, quotient collapse, or host replacement is part of the root-facing ledger.*

Proof. Start with the carrier and NameCert obligations. They display R as the refusal ledger, A as the audit readback, and H, C, P, N as the transport, continuation, provenance, and local naming rows for the same packet.

Use root visible source. Theorem 715.9 places S, Q before every refusal or audit read, so the ledger suffix is read only after the visible source route and question are fixed.

Apply audit ledger exhaustion. Theorem 715.8 says that R, A, H, C, P, N exhaust the audit-facing rows. The carrier has no coordinate for an additional refusal channel, hidden predicate, detached audit answer, quotient collapse, or host replacement. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root refusal audit ledger`

Theorem 715.11 (Apophatic gate question kind readback determinacy)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, once the displayed rows* S, Q, R, A, H, C, P, N *have been accepted, the audit-readback row* A *is determined by the visible socket row* S*, the gate-question row* Q*, and the refusal-ledger row* R*. The readback cannot be chosen from a hidden outside-supply predicate, host semantic judgment, quotient collapse, unmarked selector, or detached gate answer.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier kind readback determinacy`

Proof. Project the carrier and local NameCert obligations. They make S the visible socket-name row, Q the question asked about that row, R the refusal ledger, and A the finite audit readback inside the same packet.

Use source exposure and socket-name dependency. Theorems 715.6 and 715.4 force the question to read the sibling $\mathsf{ApophaticSocket}^{\uparrow}$ or $\mathsf{ApophaticName}^{\uparrow}$ surface before any readback is available.

Now apply audit route exactness and the root refusal audit ledger. The row A records only the answer to the displayed question over S after the refusal row R has been fixed, and Theorem 715.10 leaves no second refusal or audit coordinate.

Finally use visible socket non-escape. Since the carrier has no coordinate for a hidden predicate, host judgment, quotient collapse, unmarked selector, or detached gate answer, the kind readback is determined by the visible rows S, Q, R. □

Theorem 715.12 (Apophatic gate question supply-kind route totality)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, every supply-kind-facing route for the gate question first reads the visible socket row* S *as a finite* BHist *route, then reads the gate-question row* Q*, then reads the finite supply-kind and requested-shape refusal row* R*, and only then reads the audit readback row* A*.*

The public route vocabulary is exactly BHist, hsame, Cont, Pkg, *and* NameCert *over the displayed rows* S, Q, R, A, H, C, P, N. *It contains no hidden outside-supply witness, host gate answer, quotient collapse, unmarked selector, or detached audit answer.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier supply kind route totality`

Proof. Start with source exposure. Theorems 715.6 and 715.9 place the visible socket row S before the question row Q and keep both rows inside the finite BHist packet.

Next use socket-name dependency. Theorem 715.4 forces the gate question to read the visible ApophaticSocket$^{\uparrow}$ or ApophaticName$^{\uparrow}$ surface before any refusal or audit suffix can be consumed.

Finally apply kind-readback determinacy. Theorem 715.11 determines the audit readback A from S, Q, R, so the supply-kind and requested-shape row R is read before audit readback. The transport, continuation, provenance, and local naming rows supply only hsame, Cont, Pkg, and NameCert vocabulary for that finite route. □

Theorem 715.13 (Apophatic gate question refusal readback exactness)**.** *For an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$, *the refusal row* R *and readback row* A *are exactly the finite ledger suffix after the visible source row* S *and gate-question row* Q. *They are transported only by* H, C, P, N, *namely by componentwise* hsame, *displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row over* BHist.

No hidden outside-supply witness, host gate answer, quotient collapse, unmarked selector, detached readback, or additional refusal channel is part of the suffix.

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal readback exactness`

Proof. Project the carrier and local NameCert obligations. The packet displays R as the refusal row and A as the audit readback row after the visible source and question rows S, Q.

Use the root refusal audit ledger. Theorem 715.10 identifies R, A, H, C, P, N as the complete suffix after S, Q and leaves no second refusal or audit coordinate.

The exact transport vocabulary is then forced by the carrier rows. The row H provides componentwise hsame, C provides the displayed Cont routes, P provides Pkg provenance, and N provides the local NameCert row over the same finite BHist packet. Thus R and A are transported only by H, C, P, N. □

Theorem 715.14 (Apophatic gate question audit consumer non-escape)**.** *Every downstream audit consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$ *reads only the finite* BHist *route*

$$S, Q, R, A, H, C, P, N.$$

The route vocabulary is hsame *transport,* Cont *continuation,* Pkg *provenance, and local* NameCert *naming for those displayed rows. The consumer receives no hidden axiom, outside supply witness, quotient collapse, unmarked selector, host replacement, or detached audit answer.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit consumer nonescape`

Proof. Read the routed question first. Theorem 715.12 forces the consumer through S, Q, R before audit readback, so no audit consumer can bypass the visible socket row or the finite refusal row.

Then use refusal readback exactness. Theorem 715.13 identifies R, A, H, C, P, N as the exact finite suffix transported by hsame, Cont, Pkg, and NameCert vocabulary over BHist.

Finally apply visible socket non-escape and audit ledger exhaustion. The carrier has no coordinate for hidden axioms, outside supply witnesses, quotient collapses, unmarked selectors, host replacements, or detached audit answers, so every downstream audit consumer is confined to the displayed packet. □

Theorem 715.15 (Apophatic gate question question exactness route)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, every accepted packet read consumes the visible socket-name row* S *before it consumes the gate-question row* Q*. The row* Q *may ask only about the displayed socket-name route and the marked supply kind carried by* S, Q, R, A, H, C, P, N*; it cannot ask through a hidden host predicate, detached supply witness, quotient collapse, or unmarked selector.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier question exactness route`

Proof. Project the carrier of Definition 715.1. It places the visible socket-name row S before the gate-question row Q and keeps both rows inside the same finite packet.

Apply the local NameCert obligations and source exposure. Theorems 715.2 and 715.6 make S the exposed $\mathsf{ApophaticSocket}^{\uparrow}$ or $\mathsf{ApophaticName}^{\uparrow}$ route and make Q the question asked about that route.

Use visible socket coverage and kind readback determinacy. Theorems 715.7 and 715.11 keep the question tied to the marked kind read through the displayed rows, so hidden host predicates and detached supply witnesses are outside the accepted route. □

Theorem 715.16 (Apophatic gate question refusal transport scope)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the refusal row* R *and the transport, continuation, provenance, and naming rows* H, C, P, N *preserve the same visible socket/question boundary determined by* S *and* Q*. Transport may replay that boundary, but it cannot add hidden axioms, host answers, detached outside-supply predicates, quotient collapse, or unmarked selectors.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal transport scope`

Proof. Read the carrier and local NameCert obligations. They identify R as the refusal row for the visible question and identify H, C, P, N as the only componentwise transport, continuation, provenance, and local naming rows.

Apply source exposure and visible socket coverage. These force the refusal row to remain attached to the already displayed socket-name route and question boundary rather than to an ambient host answer.

Now use audit ledger exhaustion and kind readback determinacy. The audit suffix is exhausted by the finite rows of G, and the readback is determined by S, Q, R. Therefore transport through H, C, P, N preserves the same boundary and cannot introduce hidden axioms or a second answer source. □

Theorem 715.17 (Apophatic gate question audit readback source determinacy)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the audit row* A *is determined by the displayed route through* S, Q, R*. It cannot be chosen from a detached host predicate, hidden outside-supply source, quotient collapse, semantic harmlessness judgment, or unmarked selector outside the finite packet.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit readback source determinacy`

Proof. Begin with the carrier and NameCert obligations. The row A appears only as the audit readback of the same finite packet that contains the socket row S, question row Q, and refusal row R.

Use source exposure and visible socket coverage to fix the source. The audit readback must follow the $\mathsf{ApophaticSocket}^{\uparrow}$ or $\mathsf{ApophaticName}^{\uparrow}$ route exposed by S and the question asked by Q.

Apply audit ledger exhaustion and kind readback determinacy. These results exhaust the audit-facing rows and determine A from S, Q, R, leaving no detached host predicate or hidden outside-supply source from which the readback could be selected. □

Theorem 715.18 (Apophatic gate question refusal row transport)**.** *For an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, componentwise* hsame *transport through* H *preserves the refusal ledger row* R *together with its visible socket and gate-question dependencies. Transport may replay the displayed refusal row, but it cannot create a hidden outside-supply row, an additional refusal channel, a host gate answer, a quotient collapse, or an unmarked selector.*

Proof. Start with the carrier. The row H is the componentwise hsame transport row for the socket row, question row, refusal ledger, readback row, provenance row, and local naming row.

Apply refusal nonescape coverage and the existing refusal ledger transport theorem. Theorems 715.25 and 715.28 identify R as the refusal row carried by the packet and make H replay that same row.

Use visible socket non-escape and the root refusal audit ledger to rule out extra supply. The refusal suffix is already exhausted by R, A, H, C, P, N after S, Q, and no carrier coordinate exports a hidden outside-supply row, host answer, quotient collapse, or unmarked selector. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal row transport`

Theorem 715.19 (Apophatic gate question audit composition handoff)**.** *An accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet can hand the rows* A, R, H, C, P, N *to* $\mathsf{AuditGateComposition}^{\uparrow}$ *only after the visible socket-name dependency through* $\mathsf{ApophaticSocket}^{\uparrow}$ *and* $\mathsf{ApophaticName}^{\uparrow}$ *has been read. The handoff therefore carries the finite audit readback, refusal ledger, transport, continuation, provenance, and local naming rows, but not a hidden outside-supply witness, host semantic proof, detached gate answer, or unmarked selector.*

Proof. Read the source side first. Theorems 715.6, 715.4, and 715.9 place the visible socket-name row S before the gate question Q and before any audit suffix can be consumed.

Next read the suffix. The root refusal audit ledger and audit ledger exhaustion identify R, A, H, C, P, N as the complete refusal and audit rows available after S, Q.

Use kind readback determinacy and refusal row transport to prepare the composition route. The row A is fixed by S, Q, R, and transport through H preserves the same refusal row instead of adding a hidden supply row.

Finally apply visible socket non-escape. The handoff to $\mathsf{AuditGateComposition}^{\uparrow}$ can cite only the displayed audit suffix over the visible socket-name dependency, so host proofs, detached gate answers, unmarked selectors, and hidden outside-supply witnesses are not exported. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit composition handoff`

Theorem 715.20 (Apophatic gate question kind socket totality)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the socket kind is total after the visible socket row* S*, the gate-question row* Q*, and the refusal row* R *have been read. The accepted route therefore determines which visible socket question is being refused before any audit readback, transport, continuation, provenance, or local naming row is consumed.*

Proof. Begin with Definition 715.6. It exposes S as the visible socket-name route and places Q as the question asked about that route.

Next apply Definition 715.11. Once S, Q, R are fixed, the kind readback is determined inside the displayed packet rather than by a host predicate or detached outside-supply row.

Use Definition 715.18 only after that determination. Transport preserves the same refusal row, so later rows replay the total socket kind instead of selecting another kind source. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier kind socket totality`

Theorem 715.21 (Apophatic gate question audit readback cut)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the audit readback cut passes through exactly the suffix*

$$A, H, C, P, N.$$

The cut is attached to the already displayed rows S, Q, R *and cannot be replaced by a hidden outside-supply answer, a host semantic proof, a quotient collapse, or an unmarked selector.*

Proof. Read S, Q, R through Definition 715.6 and Definition 715.11. These theorems fix the visible source and the determined kind before the audit row can be consumed.

Now apply Definition 715.18. The row H preserves the refusal ledger for the same socket question, so the audit readback is not detached from the source triad.

Finally use Definition 715.19. It hands A, R, H, C, P, N to downstream audit composition only after the visible socket-name dependency has been read. Thus the audit cut through A, H, C, P, N remains finite and carries no hidden answer source. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit readback cut`

Theorem 715.22 (Apophatic gate question downstream unblock surface)**.** *Every downstream audit consumer of an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet receives exactly the finite route*

$$S, Q, R, A, H, C, P, N.$$

The route supplies visible socket exposure, gate-question readback, refusal, audit readback, componentwise transport, continuation, provenance, and local naming. It does not supply a host-level answer, hidden outside-supply witness, quotient collapse, unmarked selector, or extra audit channel.

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier downstream unblock surface`

Proof. First apply Definition 715.6 and Definition 715.11. These identify the source socket and the determined readback before any downstream audit route is available.

Next use Definition 715.18. Transport can only replay the refusal row over the same visible source and question.

Finally invoke Definition 715.19. Downstream audit composition receives the finite rows A, R, H, C, P, N after S, Q, so every consumer surface is exhausted by S, Q, R, A, H, C, P, N and no extra audit channel is exported. □

Theorem 715.23 (Apophatic gate question root readback route)**.** *Every downstream audit consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet receives only the finite route*

$$S, Q, R, A, H, C, P, N.$$

The route exposes the visible socket-name row, the gate question, the refusal ledger, the audit readback, componentwise transport, displayed continuation, package provenance, and local name. It exports no hidden outside supply, axiom channel, quotient collapse, proposition-extensionality collapse, or host replacement for the gate answer.

Proof. Read the visible source row first. Theorem 715.9 forces every root-facing consumer through S before Q.

Then read the refusal and audit suffix. Theorem 715.10 identifies R, A, H, C, P, N as the complete suffix after S, Q.

Finally apply visible socket coverage and audit ledger exhaustion. Theorems 715.7 and 715.8 keep downstream audit readback inside the displayed packet, so hidden outside supply, axiom channels, quotient collapse, proposition-extensionality collapse, and host replacements are not exported. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root readback route`

Theorem 715.24 (Apophatic gate question source gate exactness)**.** *For an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet $G = (S, Q, R, A, H, C, P, N)$, the gate-question row Q is exact only after the displayed socket-name row S has been read from the* ApophaticSocket$^{\uparrow}$ *or* ApophaticName$^{\uparrow}$ *sibling surface. The row Q therefore asks about the named socket route and cannot be replaced by a detached host predicate, an unnamed outside-supply witness, or a readback answer that precedes S.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier source gate exactness`

Proof. Project the carrier of Definition 715.1. Its first row is the visible socket-name row S, and its gate-question row Q is part of the same finite packet.

Apply Definition 715.2 and Definition 715.4. These rows force the local certificate and audit route to read S through the sibling socket or name surface before Q is available.

Use Definition 715.3. Since no carrier coordinate exports a hidden host predicate, unnamed supply witness, or detached readback answer, exactness of Q is exactly the question asked after the displayed socket-name read. □

Theorem 715.25 (Apophatic gate question refusal nonescape coverage)**.** *For an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet, the refusal ledger row R covers hidden axioms, quotient collapse, proposition-extensionality collapse, unmarked selectors, and unrecorded inscription novelty. This coverage does not export any requested outside supply; it only records which forbidden route is refused inside the displayed packet $G = (S, Q, R, A, H, C, P, N)$.*

Proof. Begin with Definition 715.2. It identifies R as the local refusal-ledger obligation of the finite packet.

Apply Definition 715.3. The consumer surface cannot export a hidden axiom, quotient equality, proposition-extensionality collapse, unmarked selector, unrecorded inscription novelty, or host-level replacement for the gate answer.

Finally use Definition 715.4. The refusal is attached to the visible socket-name route read through S and Q, so coverage remains a BEDC ledger row and does not supply the requested outside object. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal nonescape coverage`

Theorem 715.26 (Apophatic gate question audit route exactness)**.** *For an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet, the audit-readback row A records only the finite answer to the already displayed gate question Q over the socket-name row S. The audit route cannot be replaced by a host judgment, a semantic harmlessness proof, an unmarked selector, or an outside-supply witness.*

Proof. Read Definition 715.1. The audit-readback row A is displayed after the socket-name row S, the gate question Q, and the refusal ledger R.

Apply Definition 715.4 and Definition 715.24. They put the sibling socket-name read and exact gate question before any audit answer can be recorded.

Use Definition 715.3 together with Definition 715.25. These rows exclude host judgments, semantic harmlessness proofs, unmarked selectors, and outside-supply witnesses, leaving A as the finite readback of the displayed question. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit route exactness`

Theorem 715.27 (Apophatic gate question boundary row triad)**.** *For every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet $G = (S, Q, R, A, H, C, P, N)$, the boundary rows S, Q, R are consumed in that order. The row S is read through the sibling* ApophaticSocket$^{\uparrow}$ *or* ApophaticName$^{\uparrow}$ *surface, Q asks only about the displayed socket use, and R records the refusal surface for that same visible route.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier boundary row triad`

Proof. Start from the carrier and local NameCert obligations. They display S before Q and R, and they identify these rows as socket-name exposure, gate-question exactness, and refusal-ledger coverage.

Apply source exposure and socket-name dependency. The question route must read the sibling ApophaticSocket$^{\uparrow}$ or ApophaticName$^{\uparrow}$ row before the gate question can be answered.

Use visible socket non-escape and audit route exactness. They keep Q attached to the displayed socket use and leave R as the refusal row for that visible route, not as a hidden host answer or outside-supply witness. □

Theorem 715.28 (Apophatic gate question refusal ledger transport)**.** *For every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet $G = (S, Q, R, A, H, C, P, N)$, componentwise* hsame *transport in H preserves the refusal-ledger row R. Transport can replay the displayed refusal ledger, but it cannot add hidden axioms, quotient collapse, unmarked selectors, or host gate answers.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal ledger transport`

Proof. Read the carrier and NameCert obligations. The row H is explicitly componentwise hsame transport for the displayed socket row, question row, refusal ledger, readback row, provenance row, and local naming row.

Use refusal nonescape coverage together with visible socket non-escape. These results make R the only refusal ledger carried by the packet and exclude hidden axioms, quotient collapse, unmarked selectors, and host-level replacements for the gate answer.

Finally apply the boundary row triad. Since R is consumed after the displayed socket row and the displayed question row, transport in H preserves the same finite refusal ledger rather than adding a second gate answer. □

Theorem 715.29 (Apophatic gate question audit readback locality)**.** *For every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the audit-readback row* A *is local to the displayed rows* S, Q, R, H, C, P, N*. It cannot precede the socket-name read, and it cannot be replaced by an answer detached from the carrier,* NameCert *route, visible socket dependency, or audit route.*

Proof. Begin with audit route exactness. The row A records only the finite answer to the already displayed gate question Q over the socket-name row S.

Apply source exposure, socket-name dependency, and the boundary row triad. These force the socket-name read through S, the displayed question through Q, and the refusal ledger through R before the audit readback can be consumed.

Use refusal ledger transport and visible socket non-escape for the remaining rows. The transport, continuation, provenance, and naming route through H, C, P, N preserves the same local packet and prevents a detached host answer, hidden axiom, quotient collapse, unmarked selector, or outside-supply witness from replacing A. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier audit readback locality`

Theorem 715.30 (Apophatic gate question root classifier socket triad)**.** *For every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the visible socket row* S*, refusal-ledger row* R*, and gate classifier readback row* A *are bound through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates. The root classifier surface therefore reads one displayed socket-refusal-classifier triad, not a detached outside-supply predicate, ambient refusal oracle, or host-level gate answer.*

Proof. Start from the carrier and NameCert obligations. They display S, R, A inside the same finite packet and identify H, C, P, N as the only transport, route, provenance, and local naming coordinates for those rows.

Use visible socket non-escape and socket-name dependency. These results force the classifier read to enter through the visible socket-name surface before it can consult the refusal row or audit readback row.

Finally apply audit-ledger exhaustion, source-gate exactness, and audit-route exactness. The refusal and classifier rows remain attached to the same BHist packet and cannot be replaced by an outside-supply predicate, ambient refusal oracle, or host gate answer. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root classifier socket triad`

Theorem 715.31 (Apophatic gate question root ledger factorization)**.** *Every downstream audit read or refusal read of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet factors through the displayed rows* S, Q, R, A, H, C, P, N*. In particular, a downstream consumer cannot read an ambient refusal oracle, hidden host predicate, detached gate answer, or outside-supply witness outside the finite root ledger.*

Proof. Read the carrier first. It contains exactly the visible socket row, gate-question row, refusal ledger, audit readback, componentwise transports, displayed continuation routes, provenance, and local naming row.

Apply the NameCert obligations and visible socket non-escape. These theorems give the complete admissible consumer surface and exclude hidden axioms, quotient collapse, unmarked selectors, host gate answers, and outside-supply witnesses.

Now use audit-ledger exhaustion and audit-route exactness. Every downstream audit or refusal read reaches R and A only after the displayed socket and question rows have been consumed, so the read factors through the root ledger rather than through an ambient refusal oracle. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root ledger factorization`

Theorem 715.32 (Apophatic gate question root refusal coverage)**.** *The visible socket row S, refusal row R, and classifier readback row A form the root* NameCert *coverage surface for downstream apophatic-name and audit-boundary packets. Any downstream packet that consumes the apophatic gate question receives those three rows through H, C, P, N and receives no separate refusal channel.*

Proof. Use the root classifier socket triad. It binds S, R, A through one BHist packet and through the same transport, continuation, package, and naming coordinates.

Use the root ledger factorization. Every downstream audit or refusal read already factors through the displayed root ledger, so downstream apophatic-name and audit-boundary consumers cannot bypass S, R, A by reading a separate refusal channel.

Finally apply socket-name dependency, refusal nonescape coverage, and audit readback locality. The socket, refusal, and classifier readback rows are exactly the coverage surface exported to downstream packets, while all outside-supply and host-answer alternatives remain outside the accepted carrier. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root refusal coverage`

Theorem 715.33 (Apophatic gate question public NameCert package)**.** *For every accepted* ApophaticGateQuestion *packet $G = (S, Q, R, A, H, C, P, N)$, the public* $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$ *package is exactly the carrier row of Definition 715.1, the local* NameCert *obligations of Definition 715.2, visible-socket non-escape from Definition 715.3, socket-name dependency from Definition 715.4, source-gate exactness from Definition 715.24, refusal nonescape coverage from Definition 715.25, and audit route exactness from Definition 715.26. No additional public row, host answer, hidden axiom, quotient collapse, unmarked selector, or detached outside-supply witness is exported by the package.*

Proof. Project the carrier. It displays only the socket-name, gate-question, refusal-ledger, audit-readback, transport, continuation, provenance, and local naming rows S, Q, R, A, H, C, P, N.

Use the local NameCert obligations to identify the certificate fields with those rows. Visible-socket non-escape and socket-name dependency keep the package attached to the sibling $\mathsf{ApophaticSocket}^{\uparrow}$ or $\mathsf{ApophaticName}^{\uparrow}$ read.

Apply source-gate exactness, refusal nonescape coverage, and audit route exactness. These results bind the question, refusal ledger, and readback rows to the displayed packet and exclude every route not stored in G, so the listed rows are the whole public package. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier public namecert package`

Theorem 715.34 (Apophatic gate question root unblock package)**.** *Every root consumer of the public* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *package may cite only the displayed route S, Q, R, A, H, C, P, N and its* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The consumer receives the socket-name read, gate question, refusal ledger, audit readback, componentwise transport, continuation replay, provenance, and local certificate rows, but receives no hidden source predicate, semantic harmlessness proof, quotient collapse, unmarked selector, or host gate answer.*

Proof. Begin with the public package theorem. It identifies the exported surface with the carrier and the displayed certificate rows, so a root citation has no coordinate outside S, Q, R, A, H, C, P, N.

Use boundary row triad and audit readback locality. They force the root route to read S, Q, R, A in the displayed order before transport, continuation, provenance, and naming are consumed.

Finally apply visible socket non-escape and refusal ledger transport. The BHist, hsame, Cont, Pkg, and NameCert rows can replay only the same packet, which rules out hidden source predicates, semantic harmlessness proofs, quotient collapse, unmarked selectors, and host gate answers. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root unblock package`

Theorem 715.35 (Apophatic gate question source-question ledger lock)**.** *For every root consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the root ledger is locked in the order* S *before* Q*, and* Q *before each of* R, A, H, C, P, N*. Thus every source-question-refusal-readback citation reads the visible socket-name row before the gate question and reads the gate question before refusal, audit readback, transport, continuation, provenance, or local naming can be exported.*

Proof. Project the carrier and local NameCert obligations. They display G as the finite packet S, Q, R, A, H, C, P, N, with S the source socket-name row and Q the gate-question row.

Use source exposure, socket-name dependency, and the boundary row triad. These force S to be read before Q, and force Q to precede the refusal row R on every accepted root route.

Now apply audit readback locality and the root unblock package. The audit row A and the later rows H, C, P, N are consumed only as the same packet's readback, transport, continuation, provenance, and naming coordinates, so none of them can be exported before the source-question lock has been read. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier source question ledger lock`

Theorem 715.36 (Apophatic gate question visible audit route totality)**.** *The visible audit route of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet is exhausted by the finite rows* S, Q, R, A, H, C, P, N *together with their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure. Every downstream audit consumer obtains the source-question-refusal-readback route from these rows and obtains no further audit carrier outside the displayed packet.*

Proof. Start with the public NameCert package and the root unblock package. They identify the exported root surface with the displayed carrier rows and with the BHist, hsame, Cont, Pkg, and NameCert coordinates attached to those rows.

Apply the source-question ledger lock. It fixes the entrance route as S before Q and places the refusal ledger, audit readback, transport, continuation, provenance, and naming rows after the gate question.

Finally use refusal ledger transport and audit readback locality. Transport in H, route replay in C, provenance in P, and local naming in N preserve the same packet, so the visible audit route has no extra carrier beyond S, Q, R, A, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier visible audit route totality`

Theorem 715.37 (Apophatic gate question refusal readback exhaustion)**.** *Every refusal readback exported by the public* ApophaticGateQuestion$^{\uparrow}$ *root package is exhausted by the displayed source-question-refusal-readback route. It cannot export a hidden source predicate, host gate answer, quotient collapse, proposition-extensionality collapse, unmarked selector, or outside-supply witness.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal readback exhaustion`

Proof. Read the refusal ledger through source-gate exactness and refusal nonescape coverage. The row R answers only the displayed question Q after the visible source row S has been consumed, and it records the forbidden channels inside the finite packet.

Use visible audit route totality. The readback row A and the rows H, C, P, N are the only audit, transport, continuation, provenance, and naming coordinates available to a root consumer.

Close with the public package theorem and the root unblock package. Since the exported route is exactly the finite BEDC packet and its certificate rows, hidden source predicates, host gate answers, quotient collapse, proposition-extensionality collapse, unmarked selectors, and outside-supply witnesses have no export coordinate. □

Theorem 715.38 (Apophatic gate question root visible socket triad)**.** *Every root consumer of the public* ApophaticGateQuestion$^{\uparrow}$ *package reads a visible socket triad before any downstream readiness claim: the socket-name row S, the gate-question row Q, and the refusal-ledger row R. The triad is carried as a finite* BHist *surface with the same componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row as the package $G = (S, Q, R, A, H, C, P, N)$. It exports no requested outside object, host gate answer, hidden axiom, quotient collapse, unmarked selector, or detached source predicate.*

Proof. Start with the root unblock package, Definition 715.34. A root citation may cite only the displayed route S, Q, R, A, H, C, P, N and the attached BHist, hsame, Cont, Pkg, and NameCert rows.

Use source exposure and boundary row triad. These place S before Q and Q before the refusal ledger R, so the visible socket, question, and refusal rows form the root triad read before downstream consumers see the audit readback.

Finally apply visible socket non-escape and refusal nonescape coverage. The triad names the visible socket and records the refusal ledger inside the packet, while the excluded outside object, host answer, hidden axiom, quotient collapse, unmarked selector, and detached source predicate have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root visible socket triad`

Theorem 715.39 (Apophatic gate question root source exactness)**.** *For every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet, the root source used by downstream consumers is exactly the displayed source-before-question route $S \to Q \to R \to A$. The route is recorded in* BHist*, transported by* hsame*, replayed through* Cont*, packaged by* Pkg*, and named by* NameCert*. No hidden source predicate, semantic harmlessness proof, host gate answer, outside-supply witness, quotient collapse, or unmarked selector can replace this source route.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root source exactness`

Proof. Project the carrier $G = (S, Q, R, A, H, C, P, N)$. The only source row is S, the only question row is Q, the only refusal ledger is R, and the only audit readback is A.

Apply source exposure, source-gate exactness, and audit route exactness. Together they force the displayed source to precede the gate question, the refusal ledger to answer that question inside the packet, and the audit row to read back the same finite route.

The root unblock package supplies the remaining hsame, Cont, Pkg, and NameCert rows. Since all downstream readiness reads pass through those rows, no hidden source predicate, semantic harmlessness proof, host answer, outside witness, quotient collapse, or unmarked selector can replace $S \to Q \to R \to A$. □

Theorem 715.40 (Apophatic gate question root naming readiness)**.** *The public* ApophaticGateQuestion$^{\uparrow}$ *root package is naming-ready for downstream audit and refusal consumers exactly when it exposes the visible socket triad of Definition 715.38 and the source exactness route of Definition 715.39. Its readiness surface is the finite packet S, Q, R, A, H, C, P, N with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it does not supply a requested outside object, a host gate answer, a hidden axiom, a quotient collapse, an unmarked selector, or a detached source predicate.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root naming readiness`

Proof. First use the visible socket triad. It gives downstream consumers the socket-name row, gate question, and refusal ledger as visible finite rows before any audit readback is consumed.

Next use root source exactness. The source-before-question route is exactly $S \to Q \to R \to A$, so the readback row cannot be detached from the visible socket and refusal ledger.

Finally apply the public NameCert package and the root unblock package. They package H, C, P, N as the componentwise transport, continuation replay, provenance, and local naming rows, which gives downstream audit and refusal consumers enough theorem sites for readiness while preserving the stated non-escape boundary. □

Theorem 715.41 (Apophatic gate question refusal ledger routing)**.** *Every refusal-ledger consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet routes through R, A, H, C, P, N after the displayed S, Q read. The route records the refusal ledger and audit readback through the same finite packet and never obtains a hidden axiom, quotient collapse, proposition-extensionality collapse, unmarked selector, or host gate answer.*

Proof. Start with the source-gate exactness and boundary row triad. They place the socket-name row S and the gate-question row Q before the refusal ledger R can be consumed.

Apply refusal nonescape coverage and audit route exactness. The refusal row covers the forbidden routes inside the packet, and the audit row records only the readback of that displayed question and refusal.

Use refusal ledger transport, audit readback locality, and the public package theorem for the remaining route. The rows H, C, P, N transport, replay, package, and name the same ledger, leaving no coordinate for hidden axioms, quotient collapse, proposition-extensionality collapse, unmarked selectors, or host gate answers. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier refusal ledger routing`

Theorem 715.42 (Apophatic gate question provenance route exhaustion)**.** *Every accepted* ApophaticGateQuestion$^{\uparrow}$ *packet routes its provenance through exactly the displayed rows S, Q, R, A, H, C, P, N. The route consumes the sibling* ApophaticSocket$^{\uparrow}$ *socket-name surface and the* ApophaticName$^{\uparrow}$ *naming surface through the same* Pkg *and local* NameCert *rows, and it exposes no hidden source predicate outside the finite packet.*

Proof. Project the carrier and local NameCert obligations. They identify the only available rows as the source socket name S, gate question Q, refusal ledger R, audit readback A, componentwise transport H, continuation replay C, provenance row P, and naming row N.

Use socket-name dependency and root source exactness. The socket row is read as the visible $\mathsf{ApophaticSocket}^{\uparrow}$ source before the question row, and the refusal-audit readback cannot be detached from that source route.

Apply the public package theorem, the root unblock package, and refusal ledger routing. The provenance row P packages the same route together with the sibling $\mathsf{ApophaticName}^{\uparrow}$ naming surface, so there is no coordinate where a hidden source predicate can enter the accepted packet. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier provenance route exhaustion`

Theorem 715.43 (Apophatic gate question NameCert readback stability)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet, componentwise* hsame *transport in H and displayed* Cont *replay in C preserve the audit readback row A as the answer to the already displayed S, Q, R route. In particular, the local* NameCert *row N may rename the packet only by preserving the same source, gate question, refusal ledger, and audit readback.*

Proof. Read the displayed route through source-gate exactness and audit route exactness. These theorems bind A to the source row S, question row Q, and refusal ledger R before any later transport or replay is consumed.

Now apply the carrier obligations for H and C. Componentwise hsame can transport only the displayed rows of the same packet, and displayed Cont can replay only the route already exposed by S, Q, R, A.

Finish with the public NameCert package and root unblock package. The naming row N certifies the same readback package rather than a replacement audit answer, so stable readback is preserved under the available transport and replay rows. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier namecert readback stability`

Theorem 715.44 (Apophatic gate question paper-Lean route separation)**.** *Every paper-Lean consumer route for an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet reads only the displayed rows S, Q, R, A, H, C, P, N of the carrier $G = (S, Q, R, A, H, C, P, N)$. Such a route may read the visible socket, gate question, refusal ledger, audit readback, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, but it cannot turn the gate answer in A into a Lean verification claim.*

Proof. Project the carrier and the NameCert obligations. They identify S, Q, R, A, H, C, P, N as the complete consumer surface and place paper-Lean access inside the displayed Cont routes recorded by C.

Use visible socket non-escape and audit ledger exhaustion. These results keep the source, question, refusal, and answer rows inside the finite packet and leave no coordinate for a hidden theorem premise, host proof, or replacement verification object.

Finally apply provenance route exhaustion and NameCert readback stability. The route may replay and rename the same readback package, but the stable readback remains the packet answer to the displayed question rather than a Lean-side verification claim. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier paper lean route separation`

Theorem 715.45 (Apophatic gate question axiom-purity consumer boundary)**.** *An axiom-purity consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet may inspect the refusal rows for hidden axiom, quotient collapse, proposition-extensionality collapse, and unmarked selector exclusions. The consumer receives those exclusions only through the displayed route S, Q, R, A, H, C, P, N and receives no new proof premise beyond the finite refusal ledger and its readback.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier axiom purity consumer boundary`

Proof. Start with the carrier and visible socket non-escape. The axiom-purity read is a consumer read of the same packet, so it must pass through S, Q, R, A, H, C, P, N and cannot bypass the visible socket-question route.

Read the refusal ledger by audit ledger exhaustion and refusal ledger routing. The row R records the hidden axiom, quotient, proposition-extensionality, and selector exclusions, while A reads back that ledger as an answer to the displayed gate question.

Apply provenance route exhaustion and NameCert readback stability for the remaining rows. The rows H, C, P, N transport, replay, package, and name the same ledger, and none of them adds a fresh proof premise for the axiom-purity consumer. □

Theorem 715.46 (Apophatic gate question conservativity readback exactness)**.** *Every conservativity consumer of an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet reads the same visible socket, gate-question, refusal-ledger, and audit-readback route S, Q, R, A together with H, C, P, N. Its readback is exact for the displayed packet $G = (S, Q, R, A, H, C, P, N)$, not a host-level proof that the surrounding consumer is harmless.*

Proof. Project the carrier and the visible socket non-escape theorem. The conservativity consumer has the same visible socket and question entrance as every other accepted consumer, and no hidden source predicate or host harmlessness proof is present in the packet.

Use audit ledger exhaustion and provenance route exhaustion. They bind refusal inspection and provenance replay to S, Q, R, A, H, C, P, N, so the conservativity readback is exhausted by the displayed socket-question-refusal-readback route.

Finish with NameCert readback stability. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same readback row A, which proves exactness for the packet rather than host-level harmlessness. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier conservativity readback exactness`

Theorem 715.47 (Apophatic gate question AuditGateComposition boundary lock)**.** *Any handoff from an accepted* ApophaticGateQuestion$^{\uparrow}$ *packet to* AuditGateComposition$^{\uparrow}$ *consumes the visible source-question-refusal-readback route S, Q, R, A together with H, C, P, N. The handoff cannot introduce a host gate answer, quotient collapse, proposition-extensionality collapse, unmarked selector, outside-supply witness, or hidden source predicate.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier auditgate composition boundary lock`

Proof. Begin with the root source exactness theorem. It fixes the handoff input as the displayed route $S \to Q \to R \to A$, with R answering only the visible gate question and A reading back that answer.

Use refusal ledger routing and the provenance route exhaustion theorem. They force the remaining rows H, C, P, N to transport, replay, package, and name the same route before any downstream audit-composition consumer can cite the packet.

Finally apply the root unblock package and refusal nonescape coverage. Since every handoff coordinate is already one of S, Q, R, A, H, C, P, N, $\mathsf{AuditGateComposition}^{\uparrow}$ receives no coordinate for host gate answers, quotient collapse, proposition-extensionality collapse, unmarked selectors, outside-supply witnesses, or hidden source predicates. □

Theorem 715.48 (Apophatic gate question mature boundary)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the visible socket-name row* S*, gate-question row* Q*, refusal ledger* R*, audit-readback row* A*, componentwise* hsame *row* H*, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N *exhaust the mature paper-side boundary of the gate-question package.*

The mature boundary still does not supply the requested outside object, global audit completeness, bridge checking, a host gate answer, quotient collapse, unmarked selector, or hidden source predicate. It records the complete paper-side package inventory while keeping the verification and bridge-checking axes separate.

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier mature boundary`

Proof. Project the public NameCert package. By Definition 715.33, the exported surface is exactly the carrier and certificate rows S, Q, R, A, H, C, P, N.

Use the root unblock package and refusal ledger routing. These theorems show that root consumers and refusal-ledger consumers read the same finite rows and cannot obtain a hidden source predicate, semantic harmlessness proof, quotient collapse, unmarked selector, or host gate answer.

Finally apply root ledger factorization and root refusal coverage. They bind downstream audit and refusal reads to the visible socket-name, gate-question, refusal, and audit-readback rows, so the paper-side boundary is exhausted without adding the requested outside object, global audit completeness, or bridge checking. □

Theorem 715.49 (Apophatic gate question root source audit totality)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the visible socket source route, gate-question route, refusal ledger, audit readback, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row form one root source-audit packet over the displayed* BHist *rows. Any downstream source or audit consumer must read* S, Q, R, A, H, C, P, N *and cannot replace them by a hidden source predicate, host gate answer, outside-supply witness, quotient collapse, proposition-extensionality collapse, or unmarked selector.*

Proof. Begin with source exposure and source-gate exactness. These theorems identify S as the visible socket-name route and Q as the gate question asked against that route.

Use audit route exactness and audit ledger exhaustion for the readback side. They bind R and A to the refusal ledger and its displayed audit answer rather than to a detached answer channel.

Finally apply NameCert readback stability and the mature boundary. The rows H, C, P, N preserve the same source-audit packet by componentwise transport, continuation replay, provenance, and naming, so no hidden source predicate, host answer, outside-supply witness, quotient collapse, proposition-extensionality collapse, or unmarked selector can be read by a downstream consumer. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root source audit totality`

Theorem 715.50 (Apophatic gate question root refusal route exactness)**.** *For every accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$*, the refusal route is exactly the finite path from the visible source row* S *through the gate-question row* Q*, refusal ledger* R*, audit readback* A*, and the rows* H, C, P, N*. It is a route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data, not a host-level proof that the requested outside object exists or that refusal is semantically harmless.*

Proof. Read the source and question first. Source exposure, socket-name dependency, and source-gate exactness place S and Q before the refusal ledger can be interpreted.

Next use refusal coverage and refusal ledger routing. They identify R as the displayed refusal row and make A the local readback of that row.

Transport and package the same route through H, C, P, N. Audit route exactness, naming readiness, and the mature boundary keep this suffix on the same packet, so the refusal route is exact without importing an outside object, a harmlessness proof, or any host-level audit judgment. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root refusal route exactness`

Theorem 715.51 (Apophatic gate question root consumer ledger readiness)**.** *The downstream consumer ledger for an accepted* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *packet* $G = (S, Q, R, A, H, C, P, N)$ *is ready exactly when the visible source route, refusal route, audit readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row have all been displayed. The ledger can then be consumed by source, refusal, audit, axiom-purity, conservativity, and* $\mathsf{AuditGateComposition}^{\uparrow}$ *readers without adding an ambient apophatic closure object.*

Proof. First combine root source audit totality with root refusal route exactness. Together they display S, Q, R, A, H, C, P, N as one source-refusal-audit route rather than as separate ambient objects.

Next apply audit route exactness, public NameCert packaging, and AuditGateComposition boundary lock. These theorems show that downstream audit and composition consumers receive only the displayed packet rows.

Finally use the mature boundary. It records the complete paper-side socket-name, gate-question, refusal-ledger, audit-readback, transport, continuation, provenance, and naming inventory, so the consumer ledger is ready without a hidden apophatic closure object, outside-supply witness, host answer, quotient collapse, proposition-extensionality collapse, or unmarked selector. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier root consumer ledger readiness`

Definition 715.52 (Apophatic gate question formal target)**.** *The formal target surface for* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *is the future Lean carrier packet*

$$G = (S, Q, R, A, H, C, P, N),$$

where S *is the visible socket row,* Q *is the gate-question row,* R *is the refusal ledger,* A *is the audit-readback row,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier formal target surface`

Theorem 715.53 (Apophatic gate question formal target readback)**.** *Every future formal readback of Definition 715.52 must preserve the socket row, question row, refusal ledger, audit-readback row, componentwise transport, continuation route, provenance, and local* NameCert *row. The readback cannot add outside-supply, hidden axiom, quotient equality, proposition-extensionality, or host gate-answer coordinates.*

Proof. Start with the public NameCert package and visible socket non-escape. They identify G with the displayed finite rows and keep every source read on S.

Use socket-name dependency and the mature boundary. These results force Q, R, A, H, C, P, N to remain attached to the same socket-question packet rather than to a host answer or outside-supply coordinate.

Finally apply refusal ledger routing and axiom-purity consumer boundary. The hidden axiom, quotient, proposition-extensionality, selector, and host-answer exclusions are already ledgered inside the visible packet, so a formal readback may preserve the rows but cannot add a new forbidden coordinate. □

Lean4 ✓ `BEDC.Derived.ApophaticGateQuestionUp.ApophaticGateQuestionCarrier formal target readback`

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticGateQuestion}^{\uparrow}$ packet is generated from a visible apophatic socket-name row, a gate-question row, a refusal ledger, an audit-readback row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 715.1, Definition 715.2, Definition 715.4, Definition 715.24, Definition 715.25, Definition 715.26, Definition 715.33, Definition 715.34, Definition 715.41, Definition 715.42, Definition 715.43, Definition 715.44, Definition 715.45, Definition 715.46, Definition 715.47, and Definition 715.48 bind the mature paper-side socket-name, gate-question, refusal-ledger, audit-readback, hsame, Cont, Pkg, and NameCert rows for the exported gate-question package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticGateQuestion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, existence of the requested outside supply, global audit completeness, or every possible gate-question grammar.

Upgrade path: No further paper-side theory closure remains; Lean verification and bridge checking remain separate formal-axis obligations.

Chapter 716

A Concrete Naming Certificate for $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$

The $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ packet realizes the beta-redex preservation obligation named in chapter 9377. It packages the body typing boundary, argument typing boundary, beta-redex route, substituted body readback, substituted codomain readback, finite ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and the local $\mathsf{NameCert}$ row consumed by the MetaCIC subject-reduction interface. The packet exports a finite NameCert surface for the beta case only; it does not export host substitution semantics, semantic term equality, unrestricted conversion, or a closed subject-reduction theorem.

It is also the beta-row consumer for chapter 721: that boundary records the four displayed discharge rows, while this chapter supplies only the beta-substitution preservation row and keeps the application, lambda-domain, and Pi-domain rows external.

Definition 716.1 (BetaSubstitutionPreservation carrier)**.** *A* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$$

with the following displayed rows:

1. *B is the body typing row for a MetaCIC body checked in the context extended by the shifted domain;*
2. *A is the argument typing row for the argument checked at the unshifted domain in the original context;*
3. *V is the beta-redex row exposing the application of the lambda body to the argument as the local reduction site;*
4. *C_0 is the codomain readback row before substitution, including the shifted-domain binder boundary;*
5. *R is the substituted body readback row for* $\mathsf{substitute}(0, \mathsf{arg}, \mathsf{body})$*;*
6. *C_1 is the substituted codomain readback row for* $\mathsf{substitute}(0, \mathsf{arg}, \mathsf{cod})$*;*
7. *L is the finite preservation ledger tying B, A, V, C_0, R, C_1 to the same beta-redex typing boundary;*
8. *H stores componentwise* hsame *transports for the body typing, argument typing, redex, codomain, substituted body, substituted codomain, ledger, package, and local naming rows;*

9. *K stores the displayed* Cont *routes for reading the body typing row, the argument typing row, the beta-redex route, the substituted body, the substituted codomain, and the handoff into* $\mathsf{SubjectReduction}^{\uparrow}$*;*
10. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* ClosedTermSubstitutionBou BinderContextSubstitutionSeal$^{\uparrow}$*, and* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservation}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no host substitution semantics, no semantic term equality, no unrestricted conversion procedure, and no closed subject-reduction theorem.

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationTasteGate`
`single carrier alignment`

Theorem 716.2 (BetaSubstitutionPreservation NameCert obligations)**.** *For the carrier of Definition 716.1, the local* $\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservation}^{\uparrow}}$ *surface consists of carrier admission for*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

body typing readback, argument typing readback, beta-redex route replay, codomain boundary readback, substituted body readback, substituted codomain readback, ledger exactness, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and non-escape from host substitution semantics, semantic term equality, unrestricted conversion, and closed subject reduction.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationNameCertObligations`

Proof. Carrier admission is projection from Definition 716.1. The packet exposes exactly $B, A, V, C_0, R, C_1, L, H, K$

The body and argument rows record the two premises of the beta-redex obligation from Definition 9377.1. The codomain boundary C_0 names the shifted-domain binder interface, while R and C_1 record the two substituted readbacks.

The existing $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ and $\mathsf{BinderContextSubstitutionSeal}^{\uparrow}$ anchors supply the closed-boundary and binder-boundary discipline used to read the substitution rows as finite packet data. The $\mathsf{TypedReductionNormalForm}^{\uparrow}$ anchor fixes the endpoint as a typed reduction boundary rather than as host conversion.

Finally L, H, K, P, N bind the readbacks to one local NameCert surface. Since no carrier coordinate names host substitution semantics, semantic term equality, unrestricted conversion, or a closed subject-reduction theorem, no local projection can export one. □

Theorem 716.3 (BetaSubstitutionPreservation redex carrier)**.** *For an accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *carrier*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

the beta redex visible to the MetaCIC route is exactly the closed term source row B, the argument row A, the binder-context boundary C_0, the beta-redex row V, the substitution seal carried by R, C_1, the typed reduction and normal-form row read through L, the subject-reduction discharge socket in K, componentwise hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and the local* NameCert *row N.*

Proof. Project the carrier of Definition 716.1. The closed term source and argument source are the rows B and A, while V is the only row that names the beta-redex site.

The binder context is read at C_0, and the substitution seal is visible only through the substituted body and codomain rows R, C_1. The ledger L records the typed reduction and normal-form boundary for those rows.

The subject-reduction consumer receives a socket only through K. The remaining rows H, P, N transport, source, and name the same finite packet, so the redex carrier has no coordinate for closed subject reduction or global confluence. □

Theorem 716.4 (BetaSubstitutionPreservation context transport)**.** *Let $Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$ be an accepted* BetaSubstitutionPreservation$^{\uparrow}$ *carrier. Every binder-context transport of the beta-substitution route must keep the closed term source B, argument source A, substitution seal R, C_1, typed reduction and normal-form ledger L, subject-reduction discharge socket K, componentwise* hsame *row H, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row in the same packet.*

Proof. Start with the binder-context row C_0. It records the shifted-domain boundary before substitution, so a context transport has to carry C_0 with the source rows B, A and the redex row V.

Use the componentwise transport row H. It moves $B, A, V, C_0, R, C_1, L, K, P, N$ together and therefore prevents the substituted rows from being read after changing the binder context.

Replay is displayed through K, which is also the only subject-reduction discharge socket. Provenance P and naming N keep the route local to this packet, with no appeal to closed subject reduction or global confluence. □

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationContextTransport`

Theorem 716.5 (BetaSubstitutionPreservation typed residual non-escape)**.** *For an accepted* BetaSubstitutionPrese *carrier $Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$, the typed residual endpoint consumed by the MetaCIC route is read only through the closed term source, binder context, substitution seal, typed reduction and normal-form ledger, subject-reduction discharge socket,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows of Q. It cannot be replaced by closed subject reduction, global confluence, host substitution semantics, or unrestricted conversion.*

Proof. Read the residual endpoint through the substituted rows R, C_1 and the ledger L. These rows are available only after the body source B, argument source A, redex row V, and binder context C_0 have been displayed.

The discharge socket is the route in K. It passes the typed residual endpoint to the surrounding MetaCIC interface as a finite row, not as a closed subject-reduction theorem.

Finally apply the non-escape boundary of Definition 716.7. Since Q has no coordinates for global confluence, host substitution semantics, unrestricted conversion, or closed subject reduction, the typed residual remains local NameCert data. □

Theorem 716.6 (BetaSubstitutionPreservation redex closure)**.** *Let $Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$ be an accepted* BetaSubstitutionPreservation$^{\uparrow}$ *carrier. Any public read of the beta-redex preservation step reaches the substituted body row R and the substituted codomain row C_1 only through the displayed body typing row B, argument typing row A, beta-redex row V, codomain boundary C_0, ledger L, componentwise* hsame *transports, and displayed* Cont *routes.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationRedexClosure`

Proof. Project the accepted carrier. The source typing information is exactly B and A, and the beta-redex site is exactly V.

The route K replays the redex boundary through the codomain readback C_0 and the two substituted rows R and C_1. The ledger L records that these rows belong to the same preservation packet.

Apply the $\mathsf{SubjectReduction}^{\uparrow}$ anchor only at the handoff row recorded in K. The packet therefore supplies the beta-case input consumed by subject reduction, not an ambient subject-reduction theorem or a second reduction semantics. □

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationEventFlow extra tail rejected`

Theorem 716.7 (BetaSubstitutionPreservation non-escape)**.** *Every accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *packet exposes beta substitution preservation only through the rows*

$$B, A, V, C_0, R, C_1, L, H, K, P, N.$$

A downstream consumer may read the body typing boundary, argument typing boundary, beta-redex route, substituted body, substituted codomain, finite ledger, componentwise transports, displayed continuation routes, package provenance, and local naming row. It cannot replace those rows by host substitution semantics, semantic term equality, unrestricted conversion, or a closed subject-reduction theorem.

Proof. Begin with the carrier inventory. The only substitution-facing rows are C_0, R, C_1, and the only consumer routes are the displayed Cont routes in K.

Use Definition 716.2. The substitution rows are available only after the body typing, argument typing, beta-redex, and codomain boundary rows have been displayed inside the same packet.

The exclusions follow from absence of coordinates. Host substitution semantics, semantic term equality, unrestricted conversion, and closed subject reduction would each require a separate row or external route, and neither is present in Q. □

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationNonescape`

Theorem 716.8 (BetaSubstitutionPreservation subject-reduction handoff)**.** *An accepted* $\mathsf{BetaSubstitutionPreservat}$ *packet hands the beta-redex case to* $\mathsf{SubjectReduction}^{\uparrow}$ *as the finite NameCert packet*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N).$$

The handoff supplies the body typing premise, argument typing premise, beta-redex route, substituted body readback, substituted codomain readback, ledger, transports, continuation routes, provenance, and local name required by the parameterized MetaCIC subject-reduction interface. It does not supply the other three discharge obligations, an unrestricted conversion rule, a global normalization theorem, or the closed subject-reduction theorem excluded in chapter 9377.

Proof. Start with Definition 716.6. The beta-redex read reaches R and C_1 only through the displayed body and argument typing rows, the redex row, the codomain boundary, the ledger, and the local transport and continuation data.

The handoff route in K is the only $\mathsf{SubjectReduction}^{\uparrow}$-facing coordinate. It presents the beta case as one finite preservation packet while leaving the remaining application-argument, lambda-domain, and Pi-domain obligations outside this carrier.

Use Definition 716.7 for the boundary condition. The packet cannot be read as unrestricted conversion, global normalization, or closed subject reduction, so the subject-reduction consumer receives exactly the named beta-substitution preservation input. □

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationSubjectReductionHandoff`

Theorem 716.9 (Beta row consumption by SubjectReductionDischarge)**.** *An accepted* BetaSubstitutionPreservation *packet supplies exactly the B row of an accepted* SubjectReductionDischarge$^{\uparrow}$ *packet. The A, L, Π rows remain external discharge obligations. Every downstream consumer must read the beta row through*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$$

and must read the discharge packet through the displayed four-row inventory of SubjectReductionDischarge$^{\uparrow}$ *rather than through a closed subject-reduction theorem or host substitution semantics.*

Proof. Apply the subject-reduction handoff of Definition 716.8. It presents the accepted beta packet as the finite row

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

with the body typing, argument typing, redex, substituted readback, ledger, transport, route, provenance, and local naming rows available only inside that packet.

Next read the downstream discharge surface. By Definition 718.5, the public inventory of SubjectReductionDischarge$^{\uparrow}$ has exactly four discharge rows: beta, application-argument, lambda-domain, and Pi-domain. The beta row is therefore the only slot supplied by the BetaSubstitutionPreservation$^{\uparrow}$ handoff; the A, L, Π rows of the discharge packet remain obligations supplied outside this carrier.

Finally apply the obstruction boundary of Definition 718.4 together with the non-escape theorem Definition 716.7. These prevent the beta packet from being read as a closed subject-reduction theorem, host substitution semantics, unrestricted conversion, or a replacement for the other three discharge rows. □

Theorem 716.10 (BetaSubstitutionPreservation redex ledger exactness)**.** *Let $Q = (B, A, V, C_0, R, C_1, L, H, K, P,$ be an accepted* BetaSubstitutionPreservation$^{\uparrow}$ *carrier. The redex ledger row L records exactly the body typing row B, argument typing row A, beta-redex row V, codomain boundary row C_0, substituted body row R, substituted codomain row C_1, componentwise transports H, continuation routes K, provenance row P, and local naming row N from Definition 716.1. It does not add host substitution semantics, a semantic term-equality row, unrestricted conversion, or closed subject reduction to the beta case.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationRedexLedgerExactness`

Proof. Project the carrier of Definition 716.1. The ledger row L is listed after the beta source, redex, codomain, substituted-body, and substituted-codomain rows, so its source inventory is B, A, V, C_0, R, C_1.

The existing subject-reduction handoff of Definition 716.8 reads the same finite packet through H, K, P, N. Thus the ledger is exact for the packet consumed by the beta case.

By Definition 716.7, no coordinate supplies host substitution semantics, semantic term equality, unrestricted conversion, or closed subject reduction. The ledger therefore remains a finite BEDC row over the displayed carrier. □

Theorem 716.11 (BetaSubstitutionPreservation codomain-boundary stability)**.** *For an accepted* BetaSubstitutionPreservation$^{\uparrow}$ *carrier $Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$, the codomain boundary row C_0 and the substituted codomain row C_1 remain paired under the componentwise* hsame *transports H and displayed continuation routes K. The beta case may transport this pair through the finite packet, but it may not replace the pair by host substitution semantics, unrestricted conversion, or a closed subject-reduction theorem.*

Proof. Read the codomain rows from the carrier. The row C_0 records the codomain readback before substitution, and C_1 records the substituted codomain readback.

The rows H and K transport and replay the codomain pair together with B, A, V, R, L, P, N. The handoff theorem Definition 716.8 uses this same packet when presenting the beta case to $\mathsf{SubjectReduction}^{\uparrow}$.

Non-escape is supplied by Definition 716.7. Since no replacement coordinate exists for host substitution semantics, unrestricted conversion, or closed subject reduction, the codomain-boundary pair is stable only inside the displayed packet. □

Theorem 716.12 (BetaSubstitutionPreservation residual-budget monotonicity)**.** *Let $Q = (B, A, V, C_0, R, C_1, L, H$ be an accepted $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ carrier consumed by a downstream $\mathsf{SubjectReductionDischarge}^{\uparrow}$ packet. If the discharge packet preserves the beta row supplied by Q, then the displayed residual obligation budget can only decrease or remain unchanged: the beta row is consumed, while the application-argument, lambda-domain, and Pi-domain rows remain external discharge obligations. No row of Q creates an additional discharge obligation.*

Proof. Project the carrier of Definition 716.1. Its rows are exactly $B, A, V, C_0, R, C_1, L, H, K, P, N$, with the $\mathsf{SubjectReduction}^{\uparrow}$ handoff recorded through the displayed routes in K and provenance in P.

Apply Definition 716.9. The downstream discharge surface has four rows, and the accepted beta packet supplies exactly the beta row. The application-argument, lambda-domain, and Pi-domain rows remain outside this carrier.

Use Definition 716.7. Since Q has no coordinate for host substitution semantics, unrestricted conversion, closed subject reduction, or a replacement discharge row, it cannot introduce a new residual obligation.

Thus consuming the beta row removes one displayed obligation from the residual budget or leaves the external budget unchanged when the downstream packet merely records the handoff. The finite packet has no route that increases the discharge inventory. □

Theorem 716.13 (BetaSubstitutionPreservation discharge progress)**.** *Let $Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$ be an accepted $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ carrier consumed by an accepted $\mathsf{SubjectReductionDischarge}^{\uparrow}$ packet. If the consumer records the beta row supplied by Q, then the residual discharge surface removes exactly that beta row. The application-argument row, lambda-domain row, and Pi-domain row remain external discharge obligations, and no row of Q supplies host substitution semantics, unrestricted conversion, global confluence, or closed subject reduction.*

Proof. First read the four-row boundary through Definition 716.9. The accepted beta packet supplies exactly the beta row of the $\mathsf{SubjectReductionDischarge}^{\uparrow}$ surface, while the application-argument, lambda-domain, and Pi-domain rows are still outside the carrier.

Next apply Definition 716.12. Consuming the displayed beta row cannot create an additional residual obligation, because the handoff in K and the provenance in P route only the same finite beta packet to the downstream discharge consumer.

Finally use Definition 716.7. The carrier has no coordinate for host substitution semantics, unrestricted conversion, global confluence, or closed subject reduction. The only progress made by consuming Q is therefore the removal of the beta row; the other three discharge rows remain external. □

Theorem 716.14 (BetaSubstitutionPreservation beta-case obligation package)**.** *The beta case exposed by an accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *carrier is the finite BEDC package*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$$

together with the carrier obligations of Definition 716.2, the redex closure of Definition 716.6, the non-escape boundary of Definition 716.7, the subject-reduction handoff of Definition 716.8, the redex ledger exactness of Definition 716.10, the codomain-boundary stability of Definition 716.11, and the residual-budget monotonicity of Definition 716.12. This package is an obligation surface for the beta case; it is not host substitution semantics, unrestricted conversion, a global normalization theorem, or closed subject reduction.

Proof. The carrier obligations name the rows $B, A, V, C_0, R, C_1, L, H, K, P, N$ and the local exclusions. Redex closure then shows how the substituted body and substituted codomain are reached from the body typing, argument typing, redex, and codomain boundary rows.

The handoff theorem presents the same packet to $\mathsf{SubjectReduction}^{\uparrow}$. Redex ledger exactness records that the ledger contains precisely the displayed beta rows, and codomain-boundary stability records that the codomain pair remains paired under H and K.

Combining these finite rows gives the beta-case obligation package. The non-escape theorem excludes host substitution semantics, unrestricted conversion, global normalization, and closed subject reduction from the package boundary. □

Theorem 716.15 (BetaSubstitutionPreservation scoped discharge inventory)**.** *Let*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N)$$

be an accepted $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *carrier consumed by* $\mathsf{SubjectReductionDischarge}^{\uparrow}$*. Every scoped discharge read supplied by Q factors through the body row B, argument row A, beta-redex row V, codomain boundary C_0, substituted body row R, substituted codomain row C_1, ledger row L, componentwise* hsame *transports H, displayed* Cont *replay K, provenance row P, and local* $\mathsf{NameCert}$ *row N. It cannot supply the application-argument, lambda-domain, or Pi-domain discharge rows, nor host substitution semantics, unrestricted conversion, global confluence, or closed subject reduction.*

Proof. Project the carrier of Definition 716.1. The only rows available to the discharge consumer are the displayed coordinates $B, A, V, C_0, R, C_1, L, H, K, P, N$.

Use the beta-case obligation package of Definition 716.14. It binds the body, argument, redex, codomain, substituted readback, ledger, transport, replay, provenance, and local naming rows as one finite beta obligation surface.

Apply the four-row consumption theorem Definition 716.9. The accepted packet is exactly the beta row consumed by $\mathsf{SubjectReductionDischarge}^{\uparrow}$, while the application-argument, lambda-domain, and Pi-domain rows remain outside this carrier.

Finally use non-escape from Definition 716.7. Since no coordinate of Q names host substitution semantics, unrestricted conversion, global confluence, or closed subject reduction, every scoped discharge read stays inside the displayed beta inventory. □

Theorem 716.16 (BetaSubstitutionPreservation body and argument boundary)**.** *For an accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *carrier*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

the body typing boundary B and the argument typing boundary A are the only source typing premises available to the beta-redex route V. The ledger L, transports H, routes K, provenance P, and local name N may replay those two rows with C_0, R, C_1, but they do not add a host substitution semantics row, an unrestricted conversion row, or a closed subject-reduction claim.

Proof. Project the carrier of Definition 716.1. Its first two rows are exactly the body typing row B and the argument typing row A, and the redex route V is listed as the local beta site that consumes them.

Apply the NameCert obligations of Definition 716.2. They expose body typing readback and argument typing readback as separate obligations before the beta-redex route can be replayed.

Use redex closure from Definition 716.6. The substituted body row R and substituted codomain row C_1 are reached only through $B, A, V, C_0, L, H, K, P, N$.

Finally apply non-escape from Definition 716.7. Since no coordinate supplies host substitution semantics, unrestricted conversion, or closed subject reduction, the body and argument boundary remains exactly the displayed pair B, A. □

Theorem 716.17 (BetaSubstitutionPreservation codomain readback)**.** *For an accepted* BetaSubstitutionPreservatio *carrier*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

the codomain readback used by the beta case is exactly the pair of displayed rows C_0 and C_1: C_0 records the shifted-domain binder boundary before substitution, and C_1 records the substituted codomain readback after the beta argument is installed. No consumer may replace this pair by host codomain computation, semantic equality of terms, unrestricted conversion, or closed subject reduction.

Proof. Read the codomain rows from Definition 716.1. The row C_0 is the codomain readback before substitution, and the row C_1 is the substituted codomain readback.

Apply codomain-boundary stability from Definition 716.11. It keeps C_0 and C_1 paired under the componentwise transports H and displayed continuation routes K.

Use the beta-case obligation package of Definition 716.14. That package combines the codomain pair with $B, A, V, R, L, H, K, P, N$ as the beta obligation surface consumed by $\mathsf{SubjectReduction}^{\uparrow}$.

Non-escape from Definition 716.7 rules out host codomain computation, semantic equality, unrestricted conversion, and closed subject reduction. Thus the codomain readback is exactly the displayed pair C_0, C_1. □

Theorem 716.18 (BetaSubstitutionPreservation substitution non-escape)**.** *For an accepted* BetaSubstitutionPreser *carrier*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

the substituted body row R and substituted codomain row C_1 can be consumed only through the displayed body typing boundary B, argument typing boundary A, beta-redex route V, codomain boundary C_0, ledger L, transports H, routes K, provenance P, and local name N. The substitution read does not escape to host substitution semantics, semantic term equality, unrestricted conversion, or closed subject reduction.

Proof. Begin with redex closure from Definition 716.6. It states that any public read of the beta-redex preservation step reaches R and C_1 only through the displayed beta packet.

Apply the body and argument boundary theorem Definition 716.16. This fixes B and A as the only source typing premises available to the redex route.

Apply codomain readback from Definition 716.17. This fixes C_0 and C_1 as the displayed codomain pair, with no host-side replacement.

Finally use the original non-escape theorem Definition 716.7 and the subject-reduction handoff Definition 716.8. The handoff reads the same finite packet, while non-escape prevents substituted rows from becoming host substitution semantics, semantic equality, unrestricted conversion, or closed subject reduction. □

Theorem 716.19 (BetaSubstitutionPreservation sibling boundary lattice)**.** *For an accepted* BetaSubstitutionPreser *carrier*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N),$$

the concrete sibling boundaries of chapter 527, chapter 621, chapter 562, and chapter 718 are consumed only through the finite rows $B, A, V, C_0, R, C_1, L, H, K, P, N$*. The closed substitution and binder-context surfaces are read through* C_0, R, C_1 *and* P*, the typed endpoint surface is read through* R, C_1, L, H, K*, and the subject-reduction discharge surface is read only through the handoff row in* K *and the provenance row* P*.*

Proof. Project the provenance row P from the carrier of Definition 716.1. That row names the closed-term substitution boundary, binder-context substitution seal, typed reduction normal form, and subject-reduction endpoint as sibling surfaces, but it names them only inside the same finite packet Q.

Use redex closure from Definition 716.6. The substituted body row R and substituted codomain row C_1 are reached through $B, A, V, C_0, L, H, K, P, N$, so the closed-substitution and binder-context sibling reads cannot bypass the displayed beta-redex route.

Apply codomain readback from Definition 716.17 and substitution non-escape from Definition 716.18. These fix the C_0, R, C_1 lattice as packet data and prevent the sibling route from exporting host substitution semantics, semantic term equality, unrestricted conversion, or host codomain computation.

Finally use the subject-reduction handoff Definition 716.8. The discharge consumer receives the beta row through K and P as a finite handoff, while the other discharge rows remain outside this carrier. Thus the sibling route is substantive, but it exports no closed subject reduction. □

Theorem 716.20 (Beta substitution preservation obligation surface)**.** *Every accepted* BetaSubstitutionPreservation *packet exposes a finite obligation surface consisting of beta-redex replay, substituted body readback, codomain readback, binder-context transport, subject-reduction handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The surface exports no host substitution function, no quotient equality of terms, no unrestricted conversion, and no closed subject-reduction theorem.*

Proof. Read the carrier rows of Definition 716.1 in order. The body typing row B, argument typing row A, beta-redex row V, codomain boundary C_0, substituted body row R, and substituted codomain row C_1 supply the finite redex replay and readback data.

Next read the ledger and transport rows. The ledger L ties the displayed readbacks to one beta case, while H gives the componentwise hsame transport and K gives the displayed Cont replay routes, including the binder-context and subject-reduction handoff routes.

Finally read provenance and naming through P and N. These rows keep the packet inside the local $\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservation}^{\uparrow}}$ surface and record which sibling boundaries are consumed.

The exclusions are inspection of the same finite inventory. No coordinate of Q is a host substitution function, a quotient equality of terms, unrestricted conversion, or closed subject reduction, so no accepted read of the obligation surface can export one. □

Theorem 716.21 (Beta substitution preservation obligation rows)**.** *Every accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *packet exposes, in order, the binder-context row, substituted body row, codomain readback row, subject-reduction handoff, closed-term substitution boundary, nonescape ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any MetaCIC consumer may read beta preservation. The row order is part of the finite NameCert route: a consumer cannot read* $\mathsf{SubjectReduction}^{\uparrow}$ *through this packet before the* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, $\mathsf{BinderContextSubstitutionSeal}^{\uparrow}$, *and* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *rows have been displayed inside the same carrier.*

Proof. Project the carrier of Definition 716.1. The rows C_0, R, C_1 give the binder-context boundary, substituted body readback, and codomain readback after the source rows B, A, V have fixed the beta site.

Next read the sibling boundaries. The $\mathsf{BinderContextSubstitutionSeal}^{\uparrow}$ row keeps the binder-context surface attached to C_0, the $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ row keeps the substituted body route attached to R, and the $\mathsf{TypedReductionNormalForm}^{\uparrow}$ row keeps the endpoint readback attached to C_1 and L.

The subject-reduction handoff is then available only through K and P. The existing handoff theorem Definition 716.8 presents the beta case to $\mathsf{SubjectReduction}^{\uparrow}$ as the same finite packet, while Definition 716.7 blocks host substitution semantics, unrestricted conversion, and closed subject reduction from replacing the displayed rows.

Finally transport, replay, provenance, and local naming are componentwise. The rows H, K, P, N carry the same ordered packet to the MetaCIC consumer, so any accepted read of beta preservation first passes through the listed obligation rows and never through an off-packet preservation theorem. □

Theorem 716.22 (BetaSubstitutionPreservation scoped discharge socket)**.** *The beta row handed to* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *by an accepted* $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ *packet is scoped exactly by the body typing row, argument row, beta-redex row, binder-context boundary, substituted body row, substituted codomain row, typed residual ledger, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows already named in*

$$Q = (B, A, V, C_0, R, C_1, L, H, K, P, N).$$

The discharge socket supplies only the beta row consumed by $\mathsf{SubjectReductionDischarge}^{\uparrow}$*; the application-argument, lambda-domain, and Pi-domain rows remain external, and no read escapes to host substitution semantics, unrestricted conversion, global confluence, or closed subject reduction.*

Proof. Apply Definition 716.8. It presents Q as the finite beta-substitution preservation packet handed to the subject-reduction interface, with the body typing, argument typing, beta-redex, substituted readbacks, ledger, transport, replay, provenance, and local naming rows all displayed.

Apply Definition 716.9. It identifies the downstream discharge surface and fixes this packet as exactly the beta row of $\mathsf{SubjectReductionDischarge}^{\uparrow}$, leaving the other three discharge rows outside the carrier.

Apply Definition 716.7. Since the carrier contains no host substitution semantics, unrestricted conversion, global confluence, or closed subject-reduction coordinate, the discharge socket is scoped precisely by the rows of Q and cannot export anything beyond them. □

Closure status for $\mathsf{NameCert}_{\mathsf{BetaSubstitutionPreservation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ packet is generated from body typing, argument typing, beta-redex, codomain readback, substituted body, substituted codomain, finite ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BetaSubstitutionPreservation}\uparrow}$, publicClosed)

Scope closed: Definition 716.1, Definition 716.21, Definition 716.19, Definition 716.7, and Definition 716.22 expose the public beta-substitution packet exactly through the displayed body typing, argument, beta-redex, binder-context, substituted body and codomain, typed residual ledger, hsame, Cont, Pkg, and local NameCert rows of Q.

Formal verification: FormalStatus(NameCert$_{\mathsf{BetaSubstitutionPreservation}\uparrow}$, auditClean)

Lean target: `BEDC.Derived.BetaSubstitutionPreservationUp.BetaSubstitutionPreservationNonescape`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host substitution semantics, semantic term equality, unrestricted conversion, the other three MetaCIC subject-reduction discharge obligations, global normalization, Lean verification, or closed subject reduction.

Upgrade path: The next paper-side step is a bridge theorem connecting this public packet to downstream MetaCIC substitution consumers.

Chapter 717

A Concrete Naming Certificate for $\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}$

The $\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}$ packet realizes the MetaCIC subject-reduction audit boundary as a finite NameCert object. It records the four discharge rows consumed by the parameterized subject-reduction theorem, the checked obstruction row for the unavailable closed statement, the transport and route rows that keep those entries readable, and the local naming row that downstream MetaCIC audits can cite.

The obstruction coordinate O is read through chapter 731, while the four discharge rows remain local ledger entries. The ledger therefore records a consumer route from the obstruction chapter rather than a proof of closed subject reduction.

Definition 717.1 (MetaCIC discharge-obligation ledger carrier)**.** *A* $\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D_\beta, D_{\mathsf{app}}, D_\lambda, D_\Pi, S, O, H, C, P, N)$$

with the following visible coordinates:

1. D_β *is the* BetaSubstitutionPreservation *discharge row for the beta-redex substitution boundary;*
2. D_{app} *is the* AppArgTypeStable *discharge row for application-argument codomain readback;*
3. D_λ *is the* LamDomainSubjectReduction *discharge row for lambda-domain reduction;*
4. D_Π *is the* PiDomainSubjectReduction *discharge row for Pi-domain reduction;*
5. S *is the parameterized* $\mathsf{BEDC.MetaCIC.subject_reduction}$ *row, read with exactly the four discharge rows above;*
6. O *is the checked obstruction row* $\mathsf{BEDC.MetaCIC.closed_consistency_reduction_obstruction_not_available}$*;*
7. H *records componentwise* hsame *transports for the four discharge rows, the parameterized subject-reduction row, the obstruction row, provenance, and local naming;*
8. C *records the displayed* Cont *routes by which consumers read the four obligations, the parameterized theorem row, the obstruction boundary, provenance, and local naming;*
9. P *is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the MetaCIC discharge names, the parameterized subject-reduction row, and the obstruction row;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}}$ *row.*

The carrier contains no proof of any discharge row, no closed subject-reduction theorem, no hidden axiom channel, no host equality principle, and no route that replaces the four-row interface by an unparameterized MetaCIC theorem.

Lean4 ✓ BEDC.Derived.MetaCICDischargeObligationLedgerUp.MetaCICDischargeObligationLedgerTasteGate single carrier alignment

Theorem 717.2 (MetaCIC discharge-obligation ledger NameCert obligations)**.** *For the carrier of Definition 717.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (D_\beta, D_{\mathsf{app}}, D_\lambda, D_\Pi, S, O, H, C, P, N),$$

componentwise classifier comparison of the four discharge rows, exact readback of the parameterized BEDC.MetaCIC.subject_reduction *row through those four rows, exact readback of the checked obstruction row* O*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed packet. The rows $D_\beta, D_{\mathsf{app}}, D_\lambda, D_\Pi$ are the complete discharge surface named by the MetaCIC subject-reduction audit map, while S and O record the parameterized theorem row and the checked obstruction row.

The classifier obligation compares only those displayed rows. A comparison may inspect the beta-substitution row, the application-argument row, the lambda-domain row, the Pi-domain row, the subject-reduction row, and the obstruction row, but it must do so through the transports and routes carried by H and C.

The route and provenance obligations are finite. The row P names the MetaCIC theorem rows and BEDC kernel objects used by the packet, and N records the local NameCert row. Since no further carrier coordinate is present, the listed obligations exhaust the local certificate. □

Theorem 717.3 (MetaCIC discharge-obligation ledger non-escape)**.** *Every consumer read of an accepted* $\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}$ *packet factors through*

$$D_\beta, D_{\mathsf{app}}, D_\lambda, D_\Pi, S, O, H, C, P, N.$$

It may inspect the four discharge rows, the parameterized subject-reduction row, the checked obstruction row, componentwise transports, displayed continuations, provenance, and the local NameCert *row. It cannot read a proof of the four discharge rows, a closed subject-reduction theorem, a hidden axiom channel, a host equality principle, or an unlisted MetaCIC theorem.*

Proof. Start with Definition 717.2. It identifies the accepted packet with the displayed finite rows and names those rows as the complete local NameCert inventory.

For the subject-reduction side, inspect $D_\beta, D_{\mathsf{app}}, D_\lambda, D_\Pi$ and S. The parameterized row S is readable only with the four displayed discharge rows; the packet has no coordinate that turns those rows into discharged proofs.

For the obstruction side, inspect O. It records the checked boundary for the closed consistency reduction obstruction and keeps the closed subject-reduction statement outside the carrier.

Finally inspect H, C, P, N. These are the only transport, continuation, provenance, and naming routes available to downstream consumers. Therefore no consumer can obtain a closed subject-reduction theorem, hidden axiom, host equality principle, or unlisted MetaCIC theorem from the ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}$ packet is generated from the four MetaCIC subject-reduction discharge rows, the parameterized subject-reduction row, the checked obstruction row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 717.1, Definition 717.2, and Definition 717.3 bind the finite MetaCIC discharge-obligation ledger and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close any of the four MetaCIC discharge hypotheses, closed subject reduction, full MetaCIC consistency, a bridge certificate, host equality, or any hidden axiom channel.

Upgrade path: Obligation closure requires checked carrier admission, classifier comparison, subject-reduction readback, obstruction readback, route locality, provenance, local naming, and non-escape for the same finite packet.

Chapter 718

A Concrete Naming Certificate for $\mathsf{SubjectReductionDischarge}^{\uparrow}$

The $\mathsf{SubjectReductionDischarge}^{\uparrow}$ packet names the finite BEDC surface for the four MetaCIC subject-reduction discharge obligations. It does not prove the obligations as substrate theorems. It records the exact carrier, compiler boundary, obstruction row, transport, routes, provenance, and local naming surface by which a future discharge can be audited without hiding the checked obstruction boundary.

Definition 718.1 (Subject-reduction discharge carrier)**.** *A* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BetaSubstitutionPreservation *obligation row;*
2. *A is the* AppArgTypeStable *obligation row;*
3. *L is the* LamDomainSubjectReduction *obligation row;*
4. *Π is the* PiDomainSubjectReduction *obligation row;*
5. *S records the closed-substitution and shift compiler boundary rows used by closed-term substitution consumers;*
6. *K records the closed-term compiler and generator-audit rows that keep substitution, shifting, and generator closure inside accepted MetaCIC kernel routes;*
7. *O records the checked obstruction boundary for replacing the four-row interface by an unparameterised closed subject-reduction theorem;*
8. *H records componentwise* hsame *transports for the four obligation rows, the compiler boundary rows, the obstruction row, provenance, and local naming;*
9. *C records the displayed* Cont *routes by which a consumer reads the four obligations, the compiler boundary, the obstruction row, package provenance, and the local certificate;*
10. *P is the* Pkg *provenance over MetaCIC subject reduction, closed substitution, closed-term compiler rows, generator-audit rows,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unparameterised closed subject-reduction theorem, hidden substitution preservation, host CIC truth, evaluator soundness, quotient proof transport, or a host conversion rule outside the four displayed obligations.

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeUp.SubjectReductionDischargeTasteGate single carrier alignment`

Theorem 718.2 (Subject-reduction discharge obligations)**.** *For the carrier of Definition 718.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N),$$

four-row obligation exposure for BetaSubstitutionPreservation*,* AppArgTypeStable*,* LamDomainSubjectReduction *and* PiDomainSubjectReduction*, closed-substitution and compiler boundary exposure through* S *and* K*, obstruction exposure through* O*, componentwise* hsame *stability through* H*, displayed* Cont *locality through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 718.1. The four obligation rows B, A, L, Π are displayed before any subject-reduction consumer may read the packet.

The compiler-facing side is read through S and K. These rows reuse the closed-term substitution compiler boundary and generator-audit closure surface, so substitution and shifting are named as accepted finite routes rather than as ambient evaluator semantics.

The obstruction-facing side is read through O. It records that the closed subject-reduction theorem is not available as a replacement for the four-row interface.

Finally H, C, P, N transport, route, package, and name the same displayed rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeUp.SubjectReductionDischargeCarrier namecert obligations`

Theorem 718.3 (Subject-reduction discharge socket dependency)**.** *Every accepted* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *packet*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N)$$

consumes the same four discharge rows as the finite socket surface of chapter 719. The compiler-boundary rows S, K *and obstruction row* O *are support and boundary data, not a fifth discharge premise and not a closed subject-reduction theorem.*

Proof. Project the carrier of Definition 718.1. Its only subject-reduction discharge coordinates are B, A, L, Π.

Compare those four coordinates with the socket carrier of Definition 719.1. The socket names the beta-substitution, application-argument, lambda-domain, and Pi-domain rows as its finite discharge surface, and its NameCert obligations are checked by the socket chapter.

Now use Definition 718.2. It identifies S, K as compiler-boundary support and O as the obstruction boundary, while H, C, P, N transport, route, package, and name the displayed rows.

Since neither carrier has a fifth discharge coordinate and the discharge packet has no coordinate for a closed subject-reduction theorem, every accepted read of D consumes the socket's four-row surface rather than escaping to an additional premise or closed theorem. □

Theorem 718.4 (Subject-reduction discharge boundary)**.** *Every accepted* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *packet preserves the obstruction boundary between the parameterised MetaCIC subject-reduction*

theorem and the unavailable closed subject-reduction statement. A consumer may read the four displayed obligation rows, the closed-substitution and compiler rows, the obstruction row, transports, routes, provenance, and local name; it may not replace those rows by a hidden closed subject-reduction theorem or a host substitution theorem.

Proof. Read the four obligation rows first. They are the only rows whose joint presence can feed the parameterised subject-reduction interface.

Next read S and K. The closed-substitution compiler and generator-audit surfaces explain how closed substitution, shift, and accepted generator closure are routed, but they do not supply open-term subject reduction or evaluator correctness.

Use the obstruction row O to keep the boundary explicit. The packet records that the unparameterised closed theorem is not part of the carrier, so a consumer cannot bypass the four-row interface. □

Theorem 718.5 (Subject-reduction discharge four-row completeness)**.** *The public four-row inventory of an accepted* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *packet is complete for the MetaCIC subject-reduction discharge boundary. The beta case reads B, application-argument congruence reads A, lambda-domain congruence reads L, and Pi-domain congruence reads Π; every compiler or obstruction read is support data through S, K, O, H, C, P, N rather than a fifth discharge obligation.*

Proof. Start with the parameterised subject-reduction interface. Its discharge surface consumes exactly the beta-substitution, application-argument stability, lambda-domain, and Pi-domain rows.

The support rows do not add another discharge class. Rows S and K constrain closed substitution and compiler replay, while O records the obstruction to replacing the interface by a closed theorem.

The transport and route rows H and C preserve access to the same four rows, and P, N package and name the packet. Hence the public discharge inventory is exactly four obligation rows plus support data. □

Theorem 718.6 (Subject-reduction discharge non-escape)**.** *Every downstream read of a* $\mathsf{SubjectReductionDischarge}$ *packet factors through*

$$B, A, L, \Pi, S, K, O, H, C, P, N.$$

It cannot inspect hidden proof production, an unrecorded substitution lemma, host evaluator soundness, quotient transport, proposition extensionality, a closed subject-reduction theorem outside the obstruction row, or a subject-reduction route not accounted for by the four displayed obligations.

Proof. Project the carrier. The only public subject-reduction rows are B, A, L, Π, and every compiler-facing or audit-facing support route is listed as S, K, O, H, C, P, N.

Apply the obligation theorem to bind local naming to those rows. The packet can expose carrier admission, obligation exposure, compiler boundary exposure, obstruction exposure, transport, route locality, provenance, and local naming only.

The excluded principles have no carrier coordinate. Therefore every downstream read factors through the displayed packet, and any attempted read outside it remains an obstruction-boundary request rather than an accepted subject-reduction discharge. □

Theorem 718.7 (Subject-reduction discharge TasteGate obligation closure)**.** *For the carrier*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N)$$

of Definition 718.1, the TasteGate single-carrier alignment and the local $\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$ *obligation surface jointly close carrier admission and four-row exposure for* B, A, L, Π*. This closure does not prove* BetaSubstitutionPreservation*,* AppArgTypeStable*,* LamDomainSubjectReduction*, or* PiDomainSubjectReduction *as MetaCIC substrate theorems.*

Proof. Read the carrier definition first. It makes D a finite packet whose only subject-reduction discharge rows are B, A, L, Π, with S, K, O, H, C, P, N serving as compiler-boundary, obstruction, transport, route, package, and naming data.

The checked TasteGate marker immediately after Definition 718.1 supplies the single-carrier alignment for this packet. It binds admission to the displayed tuple rather than to an ambient closed subject-reduction theorem.

Now apply Definition 718.2. Its NameCert surface exposes the four rows B, A, L, Π and the support rows of D exactly. The closure therefore concerns admission and exposure of the obligation rows, not discharge of the four MetaCIC hypotheses themselves. □

Theorem 718.8 (Subject-reduction discharge transport-route exactness)**.** *Every accepted transport or continuation read of the four* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *discharge rows factors through the displayed* H, C, P, N *coordinates of*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N).$$

Such a read cannot introduce a fifth discharge row or a subject-reduction route outside the four rows B, A, L, Π*.*

Proof. Start with the socket dependency of Definition 718.3. It identifies the shared discharge surface as the same four rows used by the finite socket chapter.

Use the four-row completeness theorem next. The beta, application-argument, lambda-domain, and Pi-domain rows are the complete discharge inventory; S, K, O are support and boundary coordinates.

Finally apply Definition 718.6. Every downstream read must pass through the displayed carrier coordinates, so transport reads pass through H, continuation reads pass through C, and package or local-name reads pass through P, N. Since the carrier has no fifth discharge coordinate, exactness is the four-row route through those displayed coordinates. □

Theorem 718.9 (Subject-reduction discharge obstruction preservation)**.** *In every downstream read of an accepted* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *packet, the obstruction row* O *remains a visible boundary. The compiler-boundary rows* S, K *support closed-substitution and generator replay, but they cannot be read as an unparameterised closed subject-reduction theorem.*

Proof. Apply the boundary theorem Definition 718.4. It separates the four parameterised discharge obligations from the unavailable closed subject-reduction statement and records O as the boundary row.

Read S and K through the same theorem and through Definition 718.2. These rows name closed-substitution, shift, compiler, and generator-audit support; they do not add an open-term subject-reduction proof or a host evaluator theorem.

The non-escape theorem then keeps every downstream consumer inside $B, A, L, \Pi, S, K, O, H, C, P, N$. Since no coordinate stores an unparameterised closed subject-reduction theorem, O remains visible whenever a consumer tries to replace the four-row interface by that closed statement. □

Theorem 718.10 (Subject-reduction discharge schema bridge)**.** *Every public read of an accepted four-row packet*

$$D = (B, A, L, \Pi, S, K, O, H, C, P, N)$$

bridges to the MetaCIC subject-reduction schema only through the displayed compiler-boundary rows S, K, obstruction row O, componentwise transport H, continuation route C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$ *row N. The bridge exposes the public schema-facing route for B, A, L, Π and does not supply closed subject reduction.*

Proof. Apply Definition 718.2. It fixes the local certificate surface as carrier admission, four-row obligation exposure, compiler-boundary exposure, obstruction exposure, transport, route locality, provenance, and local naming.

Next apply Definition 718.5. The discharge inventory has exactly the four rows B, A, L, Π, while S, K, O, H, C, P, N are support and boundary data rather than additional discharge premises.

Use Definition 718.8 to force every transport or continuation read through the displayed H, C, P, N coordinates. Then use Definition 718.9 to keep O visible whenever a consumer tries to replace the schema-facing route by closed subject reduction.

Thus the public bridge is precisely the displayed route from the four-row packet to the MetaCIC subject-reduction schema, with no coordinate that can be read as an unparameterised closed theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionDischarge}^{\uparrow}$ packet is generated from the four MetaCIC subject-reduction discharge obligation rows, closed-substitution and compiler boundary rows, the checked obstruction row, displayed hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 718.1, Definition 718.2, Definition 718.5, Definition 718.8, Definition 718.9, and Definition 718.10 expose the public four-row discharge packet and bridge it to the cited MetaCIC subject-reduction schema target without adding a closed subject-reduction theorem.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubjectReductionDischarge}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SubjectReductionDischargeUp.SubjectReductionDischargeCarrier\_namecert\_obligations`
Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeUp.taste gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove BetaSubstitutionPreservation, AppArgTypeStable, LamDomainSubjectReduction, PiDomainSubjectReduction, unparameterised closed subject reduction, evaluator soundness, or bridge checking.

Upgrade path: No further paper-side theory-axis upgrade is claimed beyond preserving the four-row MetaCIC subject-reduction boundary.

Chapter 719

A Concrete Naming Certificate for $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$

The $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ packet turns the MetaCIC subject-reduction boundary into a finite NameCert socket. It names the four discharge rows consumed by the parameterised theorem, the parameterised theorem row itself, and the checked closed-obstruction row, so downstream compiler chapters can read the boundary as finite BEDC data rather than as an ambient closed subject-reduction claim.

The closed-obstruction row in Q is the finite obstruction boundary of chapter 731, and the route-classifier consumer of the socket is chapter 785. These siblings consume the socket as a four-row discharge surface, not as a closed subject-reduction proof.

Definition 719.1 (SubjectReductionDischargeSocket carrier)**.** *A* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, A, L, P, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation row, matching the* BetaSubstitutionPreservation *input of the MetaCIC subject-reduction theorem;*
2. *A is the application-argument stability row, matching the* AppArgTypeStable *input;*
3. *L is the lambda-domain subject-reduction row, matching the* LamDomainSubjectReduction *input;*
4. *P is the Pi-domain subject-reduction row, matching the* PiDomainSubjectReduction *input;*
5. *H records componentwise* hsame *transports for the four discharge rows, the parameterised subject-reduction row, the closed-obstruction row, provenance, and the local naming row;*
6. *C records the displayed* Cont *routes by which consumers read beta, application-argument, lambda-domain, Pi-domain, parameterised subject-reduction, obstruction, provenance, and naming coordinates;*
7. *Q is the provenance and classifier ledger over* BHist, Ext, Cont, $\mathsf{SubjectReduction}^{\uparrow}$, *MetaCIC typed reduction, componentwise* hsame, Pkg, *and* $\mathsf{NameCert}$ *rows, including the parameterised subject-reduction row and the closed obstruction row;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeSocket}^{\uparrow}}$ *row.*

The carrier contains no closed subject-reduction theorem, no host conversion rule, no erased derivation semantics, no quotient equality of MetaCIC terms, and no hidden reduction principle outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketTasteGate single carrier alignment`

Theorem 719.2 (SubjectReductionDischargeSocket NameCert obligations)**.** *For the carrier of Definition 719.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeSocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$O = (B, A, L, P, H, C, Q, N),$$

the beta-substitution row B, the application-argument row A, the lambda-domain row L, the Pi-domain row P, componentwise transport through H, displayed continuation reads through C, provenance and classifier ledgering through Q, and local naming through N. The ledger Q records the parameterised $\mathsf{SubjectReduction}^{\uparrow}$ *row as consuming precisely B, A, L, P, and records the checked closed-obstruction row as excluding any unparameterised closed subject-reduction export.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketCarrier namecert obligations`

Proof. Project the displayed packet O. The four discharge rows B, A, L, P are the only rows that can feed the parameterised $\mathsf{SubjectReduction}^{\uparrow}$ theorem.

Read H and C next. They provide the componentwise sameness transport and the continuation routes by which the four rows, the parameterised subject-reduction row, and the obstruction row may be replayed.

Finally inspect Q and N. The provenance ledger records that parameterised subject reduction consumes the four named discharge rows, while the obstruction row prevents the packet from exporting a closed subject-reduction theorem. Since no other coordinate exists in O, the listed rows exhaust the local NameCert surface. □

Theorem 719.3 (SubjectReductionDischargeSocket non-escape)**.** *Every consumer read of an accepted* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *carrier factors through*

$$B, A, L, P, H, C, Q, N.$$

The consumer may inspect the four discharge rows, their componentwise hsame *transports, the displayed* Cont *routes, the provenance and classifier ledger, and the local* NameCert *row. It cannot read a closed subject-reduction theorem, host conversion, erased derivation semantics, quotient equality of MetaCIC terms, a semantic reduction interpreter, or an unrecorded MetaCIC premise.*

Proof. Apply the NameCert-obligation theorem to identify the accepted carrier with the finite rows B, A, L, P, H, C, Q, N.

For a permitted subject-reduction read, the consumer first reads one of the four discharge coordinates and then follows the corresponding transport and route in H and C. The ledger Q records how the parameterised subject-reduction row consumes those four inputs.

For a closed subject-reduction request, the same ledger exposes the obstruction row. That row is a boundary coordinate rather than a replacement proof, so the consumer receives the finite obstruction record and no unparameterised closed theorem.

The packet has no coordinate for host conversion, erased derivation semantics, quotient equality, semantic reduction interpretation, or hidden MetaCIC premises. Those requests therefore have no route through the accepted socket. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket nonescape

Theorem 719.4 (SubjectReductionDischargeSocket obligation consumption)**.** *Let $O = (B, A, L, P, H, C, Q, N)$ be an accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carrier. Any downstream certificate that consumes MetaCIC subject-reduction data from O consumes exactly the beta-substitution, application-argument, lambda-domain, and Pi-domain rows displayed by B, A, L, P, transported by H, replayed through C, and accounted for by Q, N. The closed-obstruction row in Q can only be read as a boundary for the absent closed theorem, not as a substitute proof of closed subject reduction.*

Proof. Start with the carrier projection. The public discharge inputs are exactly B, A, L, P.

The transport and route rows H, C preserve those inputs as the rows named by the carrier; they do not add another MetaCIC premise. The provenance and classifier ledger Q then records the parameterised subject-reduction row as the only row that consumes the four inputs.

For the obstruction coordinate, read Q in the opposite direction: it reports that the closed subject-reduction statement is blocked at this socket boundary. Hence downstream consumers can cite the finite socket as an obligation surface, but cannot turn it into a closed subject-reduction theorem. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket obligation consumption

Theorem 719.5 (SubjectReductionDischargeSocket row exhaustion)**.** *Let $O = (B, A, L, P, H, C, Q, N)$ be an accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carrier. Every admissible subject-reduction socket read is one of the four discharge rows B, A, L, P, the parameterised* SubjectReduction$^{\uparrow}$ *row recorded by Q, the closed-obstruction row recorded by Q, or the local* NameCert *provenance carried by H, C, Q, N.*

Proof. Project the finite carrier O. The displayed coordinates expose exactly the four discharge rows B, A, L, P and the structural rows H, C, Q, N.

Read the ledger Q. It contains the parameterised subject-reduction consumer row and the closed-obstruction boundary row. These are ledger entries over the displayed discharge rows, not extra carrier coordinates.

The remaining admissible reads are local provenance reads through H, C, Q, N. Since the carrier has no further slot, every socket read belongs to one of the listed classes. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket row exhaustion

Theorem 719.6 (SubjectReductionDischargeSocket transport stability)**.** *Let $O = (B, A, L, P, H, C, Q, N)$ and $O' = (B', A', L', P', H', C', Q', N')$ be accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carriers. If the rows are matched componentwise by the displayed* hsame *data in H, H' and the subject-reduction reads follow the displayed* Cont *routes in C, C', then O and O' expose the same row inventory: the four discharge rows, the parameterised row, the obstruction row, and the local NameCert provenance rows.*

Proof. Use the componentwise hsame rows first. They identify B, A, L, P with B', A', L', P' as the same four discharge coordinates inside the socket classifier.

Next follow the displayed Cont routes. They replay the same parameterised subject-reduction and obstruction reads through Q, N and through Q', N'.

Transport does not add a coordinate to either carrier. Therefore the row inventory exposed by the two accepted sockets is stable under the displayed hsame and Cont data. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket transport stability

Theorem 719.7 (SubjectReductionDischargeSocket transport route stability)**.** *Let* $O = (B, A, L, P, H, C, Q, N)$ *and* $O' = (B', A', L', P', H', C', Q', N')$ *be accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carriers. If every coordinate* B, A, L, P, H, C, Q, N *is matched with its primed coordinate by the displayed componentwise* hsame *data and every consumer read is replayed through the displayed* Cont *routes, then the two sockets expose the same beta-substitution, application-argument, lambda-domain, Pi-domain, transport, route, provenance-ledger, and local naming obligations.*

Proof. Project both accepted carriers. The only coordinates of the first socket are B, A, L, P, H, C, Q, N, and the only coordinates of the second socket are $B', A', L', P', H', C', Q', N'$.

Use the componentwise hsame data to match the four discharge rows and the structural rows. The match preserves their classifier positions: beta remains beta, application remains application, lambda remains lambda, Pi remains Pi, transport remains transport, route remains route, ledger remains ledger, and name remains name.

Now replay consumer reads through C and C'. Continuation data can move along the displayed route, but it returns to the matched coordinate class rather than creating an extra socket row.

Thus transport and route replay preserve the complete obligation inventory of the accepted socket. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket transport route stability

Theorem 719.8 (SubjectReductionDischargeSocket four-row ledger exactness)**.** *Let* $O = (B, A, L, P, H, C, Q, N)$ *be an accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carrier. The subject-reduction discharge ledger in* Q, N *consumes exactly the beta-substitution row* B*, the application-argument row* A*, the lambda-domain row* L*, and the Pi-domain row* P*. No consumer route through* Q, N *may replace one of these four rows, add a fifth discharge row, or treat the closed-obstruction boundary as one of the four discharge inputs.*

Proof. Read the carrier inventory. The only discharge coordinates are B, A, L, P, and the remaining coordinates H, C, Q, N are transport, route, provenance, and naming rows.

Inspect the ledger part of Q, N. It records the parameterised subject-reduction row as a consumer of the four named discharge inputs, so each permitted discharge read must point back to one of B, A, L, P.

The obstruction entry in Q has a different classifier role. It records the absent closed theorem boundary and therefore cannot be counted as beta, application, lambda, or Pi discharge evidence.

Since the packet contains no fifth discharge coordinate and no replacement coordinate for any of the four rows, the four-row consumption ledger is exact. □

Lean4 ✓ BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket four row ledger exactness

Theorem 719.9 (SubjectReductionDischargeSocket obligation closure package)**.** *The* SubjectReductionDischargeSo *chapter supplies an obligation-closed paper package for the finite socket carrier. The package consists of carrier admission, the local NameCert obligations, non-escape, obligation consumption, vision handoff, transport route stability, and four-row ledger exactness for*

$$O = (B, A, L, P, H, C, Q, N).$$

It exposes the socket as the finite subject-reduction discharge boundary and does not claim Lean verification, closed subject reduction, host conversion, erased derivation semantics, quotient equality, or bridge checking.

Proof. Carrier admission is Definition 719.1. The local NameCert surface is Definition 719.2, and non-escape is Definition 719.3.

For downstream consumption, use Definition 719.4. It identifies the subject-reduction route as a consumer of the displayed four discharge rows and keeps the obstruction row as a boundary.

For the methodology handoff, use Definition 719.11. It records why the socket is a finite BEDC packet rather than an unparameterised preservation theorem.

Finally apply transport route stability and four-row ledger exactness. These two clauses show that the same obligation inventory is preserved under displayed hsame and Cont data, and that Q, N consume exactly B, A, L, P.

The excluded structures are absent from the carrier and are already ruled out by the non-escape theorem. The displayed clauses therefore assemble the obligation-closed paper package without changing the verification axis. □

Theorem 719.10 (SubjectReductionDischargeSocket downstream consumer boundary)**.** *Every route-equivalence or route-classifier chapter that consumes an accepted* SubjectReductionDischargeSocket$^{\uparrow}$ *carrier*

$$O = (B, A, L, P, H, C, Q, N)$$

can consume the socket only through the four discharge rows, the parameterised subject-reduction row, the obstruction row, and the local NameCert provenance rows. Such a consumer cannot manufacture a closed subject-reduction theorem, a host conversion route, an erased derivation semantics, or quotient equality of MetaCIC terms.

Proof. Begin with row exhaustion. Any admissible read from the socket is one of the listed discharge, parameterised, obstruction, or provenance rows.

Then apply transport stability. Route-equivalence and route-classifier consumers may move the same rows along displayed hsame and Cont routes, but those routes preserve the inventory rather than creating a new theorem coordinate.

The excluded closed subject-reduction, host conversion, erased semantics, and quotient-equality objects are absent from the carrier. Hence downstream consumers can cite the socket as a finite boundary and cannot turn it into a closed subject-reduction result. □

Theorem 719.11 (SubjectReductionDischargeSocket vision handoff)**.** *The MetaCIC subject-reduction obstruction vision is realized by the finite* SubjectReductionDischargeSocket$^{\uparrow}$ *packet of Definition 719.1. Its concrete handoff consists of the four discharge rows B, A, L, P, the parameterised* SubjectReduction$^{\uparrow}$ *consumption ledger, the checked closed-obstruction boundary, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *naming. No downstream chapter may cite this handoff as a closed subject-reduction theorem.*

Proof. The critical path in the vision names four discharge obligations and a checked obstruction to the closed statement. The carrier records those same four rows as B, A, L, P and places the obstruction in the ledger row Q.

The NameCert-obligation theorem supplies the finite carrier, classifier, route, provenance, and naming surface. The non-escape theorem then rules out the closed theorem and the host semantic routes.

Thus the vision is handed to downstream chapters as a finite socket: it is a concrete obligation surface for parameterised subject reduction, together with a checked boundary against closed subject-reduction export. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeSocket}\uparrow}$.

Constructive story: A $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ packet is generated from beta-substitution, application-argument, lambda-domain, and Pi-domain discharge rows, a parameterised subject-reduction consumption ledger, a checked closed-obstruction boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeSocket}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 719.1, Definition 719.2, Definition 719.3, Definition 719.4, Definition 719.5, Definition 719.6, Definition 719.7, Definition 719.8, Definition 719.9, Definition 719.10, and Definition 719.11 bind the finite MetaCIC subject-reduction discharge socket and its obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeSocket}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SubjectReductionDischargeSocketUp.SubjectReductionDischargeSocketPacket\_nonescape`

Bridge status: none

Origin: human

Not claimed: This chapter does not close any of the four MetaCIC discharge obligations, the closed subject-reduction theorem, Lean verification of the socket, host conversion, erased derivation semantics, quotient equality, or bridge checking.

Upgrade path: No paper-side theory upgrade remains; only later bridge checking can advance the verification and bridge axes.

Chapter 720

A Concrete Naming Certificate for $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$

The $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ packet realizes the audit map of chapter 9377 as a finite carrier for the four preservation obligations consumed by the $\mathsf{SubjectReduction}^{\uparrow}$ route of chapter 483. It records the obligation rows, transport rows, continuation routes, package provenance, and local naming row as a concrete BEDC object, while keeping closed subject reduction outside the exported surface.

Definition 720.1 (SubjectReductionDischargeLedger carrier)**.** *A* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, A, LD, PD, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation row, matching* BetaSubstitutionPreservation*;*
2. *A is the application-argument stability row, matching* AppArgTypeStable*;*
3. *LD is the lambda-domain subject-reduction row, matching* LamDomainSubjectReduction*;*
4. *PD is the Pi-domain subject-reduction row, matching* PiDomainSubjectReduction*;*
5. *R records the* $\mathsf{SubjectReduction}^{\uparrow}$ *route that consumes exactly B, A, LD, PD;*
6. *H records componentwise* hsame *transports for the four obligation rows, the route row, provenance, and local naming;*
7. *C records the displayed* Cont *replays by which consumers read the four obligations and the* $\mathsf{SubjectReduction}^{\uparrow}$ *route;*
8. *P is the* Pkg *provenance over MetaCIC subject reduction,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeLedger}^{\uparrow}}$ *row.*

The carrier contains no unparameterised closed subject-reduction theorem, no host conversion principle, no hidden substitution theorem, no erased derivation semantics, and no quotient equality of MetaCIC terms.

Lean4 (definition) `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerCarrier`

Theorem 720.2 (SubjectReductionDischargeLedger obligations)**.** *For the carrier of Definition 720.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (B, A, LD, PD, R, H, C, P, N),$$

four-row obligation exposure for BetaSubstitutionPreservation*,* AppArgTypeStable*,* LamDomainSubjectReduction *and* PiDomainSubjectReduction*, the* $\mathsf{SubjectReduction}^{\uparrow}$ *route row* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerNameCertObligatio`

Proof. Project the displayed carrier L. The four obligation rows B, A, LD, PD are the only rows that can feed the subject-reduction route.

Read the structural rows next. The transport row H keeps the four obligations, the route row, provenance, and local name componentwise stable, while the continuation row C records the finite replay routes used by downstream consumers.

Finally inspect P and N. The package row records provenance for the same displayed rows, and the local certificate names no further coordinate. Hence the listed rows exhaust the local NameCert surface. □

Theorem 720.3 (SubjectReductionDischargeLedger SubjectReduction handoff)**.** *Every accepted* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ *carrier hands its MetaCIC preservation data to* $\mathsf{SubjectReduction}^{\uparrow}$ *only through the route row* R *and the four obligation rows* B, A, LD, PD*. The handoff may be transported by* H*, replayed through* C*, and accounted for by* P, N*, but it does not supply a closed subject-reduction theorem or an unrecorded fifth preservation premise.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerSubjectReductionP`

Proof. Begin with Definition 720.2. It identifies the accepted carrier with the four obligation rows, the subject-reduction route row, and the structural rows.

The route row R is displayed as the only $\mathsf{SubjectReduction}^{\uparrow}$ consumer coordinate. Since R consumes exactly B, A, LD, PD, any accepted handoff to chapter 483 passes through those four rows.

The remaining rows H, C, P, N transport, replay, package, and name the same handoff. They do not add another preservation obligation, so the ledger has no route for a fifth premise or a closed theorem. □

Theorem 720.4 (SubjectReductionDischargeLedger non-escape)**.** *Every downstream read of an accepted* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ *carrier factors through*

$$B, A, LD, PD, R, H, C, P, N.$$

In particular, the finite ledger cannot be read as an unparameterised closed subject-reduction theorem, host conversion rule, hidden substitution preservation theorem, erased derivation semantics, quotient equality, or semantic MetaCIC interpreter.

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerNonescape`

Proof. Project the carrier and apply the obligation theorem. The only public preservation rows are B, A, LD, PD, and the only public subject-reduction consumer row is R.

The transport, continuation, package, and naming rows preserve access to the displayed packet but do not create new coordinates. Therefore a consumer can read the ledger only through $B, A, LD, PD, R, H, C, P, N$.

The excluded closed or host-side principles have no row in the packet. A request for one of them therefore remains outside the accepted carrier rather than becoming a hidden route through the ledger. □

Theorem 720.5 (SubjectReductionDischargeLedger TasteGate boundary)**.** *A future TasteGate surface for* SubjectReductionDischargeLedger$^{\uparrow}$ *must expose the carrier only through the single field projection*

$$L = (B, A, LD, PD, R, H, C, P, N).$$

The projection admits the beta-substitution, application-argument, lambda-domain, Pi-domain, route, transport, continuation, provenance, and local naming rows as one displayed ledger packet; it does not expose a closed subject-reduction theorem, host conversion row, hidden substitution theorem, or erased derivation semantics.

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerTasteGate single carrier alignment`

Proof. Use the carrier definition and the non-escape theorem. They identify the ledger with the displayed tuple and exclude every closed or host-side principle from the accepted packet.

The TasteGate-facing projection can therefore read the ledger only as L. Splitting the read into an unrecorded preservation theorem or adding a host conversion coordinate would require a field outside $B, A, LD, PD, R, H, C, P, N$, which the carrier does not contain. □

Theorem 720.6 (SubjectReductionDischargeLedger field-faithful surface)**.** *The field-faithful row for a future* SubjectReductionDischargeLedger$^{\uparrow}$ *TasteGate must separate the displayed fields* B, A, LD, PD, R, H, C, P *A consumer that preserves the ledger must preserve beta-substitution exposure, application-argument stability, lambda-domain subject-reduction exposure, Pi-domain subject-reduction exposure, the* SubjectReduction$^{\uparrow}$ *route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming as distinct public rows.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerFieldFaithfulSurf`

Proof. Project L and apply the obligation theorem. Each displayed coordinate has a different ledger role: four rows expose preservation obligations, one row is the route consumer, and four rows provide transport, replay, provenance, and naming.

Field faithfulness is the requirement that these roles are not identified by a TasteGate read. If two of the rows were collapsed, the handoff theorem could no longer state that SubjectReduction$^{\uparrow}$ consumes exactly B, A, LD, PD through R while H, C, P, N remain structural rows. □

Theorem 720.7 (SubjectReductionDischargeLedger nontrivial witness surface)**.** *The nontriviality row for a future* SubjectReductionDischargeLedger$^{\uparrow}$ *TasteGate must distinguish at least two displayed ledger packets. One packet may vary the route row R while preserving B, A, LD, PD, H, C, P, N; another may vary a structural row such as C while preserving the four preservation obligations and route row. A TasteGate that identifies all such packets cannot witness the displayed ledger surface.*

Proof. The carrier has separate coordinates for obligation data, route data, and structural replay data. Varying R changes the $\mathsf{SubjectReduction}^{\uparrow}$ handoff while leaving the four preservation rows fixed.

Varying C changes the displayed replay route while leaving the preservation obligations and route row fixed. These two changes affect different public coordinates of L.

The nontrivial witness row records that the ledger surface can observe those coordinate differences. A collapsed TasteGate would erase the distinction between obligation, route, and replay fields, contradicting the field-faithful surface. □

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerNontrivialWitness`

Theorem 720.8 (SubjectReductionDischargeLedger carrier exposure)**.** *Every accepted* SubjectReductionDischargeL *carrier exposes exactly the finite rows*

$$B, A, LD, PD, R, H, C, P, N.$$

The public ledger surface consists of the four preservation rows, the $\mathsf{SubjectReduction}^{\uparrow}$ *route, componentwise transport, displayed continuation replay, package provenance, and local naming; it contains no additional MetaCIC preservation coordinate.*

Proof. Begin with the carrier definition Definition 720.1. It lists $B, A, LD, PD, R, H, C, P, N$ as the full finite packet.

Apply the obligation theorem Definition 720.2. Its carrier-admission and four-row exposure clauses identify the public preservation surface with B, A, LD, PD and the subject-reduction route with R.

The handoff theorem Definition 720.3 and non-escape theorem Definition 720.4 leave only H, C, P, N as structural replay, provenance, and naming rows. Hence the accepted carrier exposes exactly the displayed ledger rows. □

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerCarrierExposure`

Theorem 720.9 (SubjectReductionDischargeLedger conditional normalization)**.** *Any downstream normalization or subject-reduction consumer of an accepted* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ *carrier must read the four displayed obligation rows* B, A, LD, PD *before using the route row* R*. The route may replay through* H, C, P, N*, but it cannot infer closed subject reduction, host conversion, hidden substitution preservation, erased derivation semantics, quotient equality, or any preservation premise outside the displayed ledger.*

Proof. Apply Definition 720.2. It exposes the carrier as the four obligation rows B, A, LD, PD, the $\mathsf{SubjectReduction}^{\uparrow}$ route row R, and the structural rows H, C, P, N.

Use Definition 720.3. The handoff to $\mathsf{SubjectReduction}^{\uparrow}$ consumes exactly B, A, LD, PD through R, so a downstream normalization or subject-reduction read cannot begin from the route row alone.

Finally apply Definition 720.8. Since the accepted carrier exposes no row beyond $B, A, LD, PD, R, H, C, P, N$, replay through H, C, P, N preserves the same conditional route and cannot create closed subject reduction, host conversion, hidden substitution preservation, erased derivation semantics, quotient equality, or an off-ledger premise. □

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerConditionalNormal`

Theorem 720.10 (SubjectReductionDischargeLedger transport replay)**.** *For every accepted* SubjectReductionDisch *carrier, the transport row H and continuation row C transport and replay only the four preservation rows B, A, LD, PD and the* SubjectReduction$^{\uparrow}$ *route row R, together with the provenance and local naming rows needed to authenticate that finite route.*

Proof. Project the carrier using Definition 720.1. The transport coordinate H is declared componentwise over the four obligation rows, the route row, provenance, and local naming.

Use Definition 720.2 to read the continuation coordinate C. It replays the displayed obligations and the SubjectReduction$^{\uparrow}$ route, with no fifth preservation row and no closed subject-reduction row.

Finally apply Definition 720.3 and Definition 720.4. They force every replay to remain inside $B, A, LD, PD, R, H, C, P, N$, so H and C cannot transport or replay an off-ledger preservation premise. □

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerTransportReplay`

Theorem 720.11 (SubjectReductionDischargeLedger provenance exactness)**.** *For an accepted* SubjectReductionDischargeLedger$^{\uparrow}$ *carrier, the rows P and N account exactly for the finite MetaCIC,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance of B, A, LD, PD, R, H, C. They do not add closed subject reduction, host conversion, hidden substitution preservation, erased derivation semantics, or quotient equality of MetaCIC terms.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerProvenanceExactne`

Proof. Read P and N from the carrier definition. The package row names the finite provenance dependencies, and the local naming row names the same displayed ledger.

Apply Definition 720.2. The obligation surface already accounts for the four preservation rows, route row, transport row, continuation row, package row, and name row.

Use the handoff theorem and non-escape theorem to close the provenance boundary. Since all downstream reads factor through the same displayed packet, P and N authenticate the finite ledger without creating a closed subject-reduction theorem or any excluded host-side principle. □

Theorem 720.12 (SubjectReductionDischargeLedger route exhaustion)**.** *Every accepted* SubjectReductionDischarg *carrier whose public rows are*

$$B, A, LD, PD, R, H, C, P, N$$

routes SubjectReduction$^{\uparrow}$ *consumption only through the four obligation rows B, A, LD, PD and the route row R. The rows H, C, P, N may transport, replay, authenticate, and name the same finite route, but they cannot create closed subject reduction, host conversion, hidden substitution preservation, erased derivation semantics, quotient equality, or a fifth preservation premise.*

Lean4 ✓ `BEDC.Derived.SubjectReductionDischargeLedgerUp.SubjectReductionDischargeLedgerRouteExhaustion`

Proof. Project the displayed tuple from Definition 720.1. Its only preservation coordinates are B, A, LD, PD, and its only SubjectReduction$^{\uparrow}$ consumer coordinate is R.

Apply Definition 720.2 and Definition 720.3. These identify R as the row that consumes exactly the four obligation rows and account for the handoff through the local certificate surface.

It remains to exhaust the structural rows. The non-escape theorem Definition 720.4 confines every downstream read to the displayed packet, while Definition 720.11 makes P and N exact provenance and naming rows for that same packet.

Thus H, C, P, N can only transport, replay, authenticate, and name the finite route already determined by B, A, LD, PD, R. None of those rows supplies an additional preservation premise or any of the excluded closed or host-side principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$ packet is generated from beta-substitution, application-argument, lambda-domain, and Pi-domain obligation rows, a $\mathsf{SubjectReduction}^{\uparrow}$ route row, componentwise hsame transports, displayed Cont replays, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeLedger}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 720.1, Definition 720.2, Definition 720.3, Definition 720.4, Definition 720.5, Definition 720.6, Definition 720.7, Definition 720.8, Definition 720.9, Definition 720.10, Definition 720.11, and Definition 720.12 bind the exact rows $B, A, LD, PD, R, H, C, P, N$ and their kernel scope over BHist, hsame, Cont, Pkg, NameCert, $\mathsf{SubjectReduction}^{\uparrow}$, and the four MetaCIC preservation obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeLedger}^{\uparrow}}, \mathsf{unformalized})$

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.subjectReductionDischargeLedgerDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove BetaSubstitutionPreservation, AppArgTypeStable, LamDomainSubjectReduction, PiDomainSubjectReduction, closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Bridge packaging requires carrying the same $B, A, LD, PD, R, H, C, P, N$ public ledger through a checked bridge without adding closed subject reduction, host conversion, hidden substitution preservation, erased derivation semantics, or quotient equality.

Chapter 721

MetaCIC Subject-Reduction Boundary NameCert Construction

The MetaCICSubjectReductionBoundaryUp packet realizes the MetaCIC subject-reduction obstruction as a finite naming boundary. It packages the four discharge obligations, the parameterized preservation theorem, the checked obstruction row, audit-visible non-escape ledgers, transport data, continuation routes, package provenance, and the local NameCert row. The packet is not a proof of closed subject reduction; it is the certificate surface that prevents the obstruction from being replaced by an unrecorded theorem.

This boundary packet refines the sibling obstruction packet of chapter 731 by separating the four discharge rows, the parameterized theorem row, and the checked obstruction row into a carrier whose consumer routes cannot erase the hypotheses.

Definition 721.1 (MetaCIC subject-reduction boundary carrier)**.** *A* MetaCICSubjectReductionBoundaryUp *carrier is a finite* BHist *packet*

$$M = (B, A, L, P_i, R, O, Q, H, C, K, N)$$

with the following visible rows:

1. *B records the* BetaSubstitutionPreservation *discharge obligation;*
2. *A records the* AppArgTypeStable *discharge obligation;*
3. *L records the* LamDomainSubjectReduction *discharge obligation;*
4. *P_i records the* PiDomainSubjectReduction *discharge obligation;*
5. *R records the parameterized* subject_reduction *theorem whose hypotheses are exactly the four obligation rows;*
6. *O records the checked obstruction row* closed_consistency_reduction_obstruction_not_available*;*
7. *Q is the audit ledger saying which of the four obligations has been displayed before any preservation read is consumed;*
8. *H stores componentwise* hsame *transports for the four obligation rows, the parameterized theorem row, the obstruction row, the audit ledger, the package row, and the local naming row;*
9. *C stores the displayed* Cont *routes by which consumers read the obligation rows, the parameterized theorem row, and the obstruction row;*

10. *K is the* Pkg *provenance over the MetaCIC typing interface, beta-step constructors,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionBoundaryUp}}$ *row.*

The carrier contains no unparameterized closed subject-reduction theorem, no hidden substitution lemma, no ambient conversion theorem, and no semantic equality model for MetaCIC terms.

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryTasteGate single carrier alignment`

Theorem 721.2 (MetaCIC subject-reduction boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectRe}}$ *surface consists of five finite obligations:*

1. *carrier admission for the packet $M = (B, A, L, P_i, R, O, Q, H, C, K, N)$ of Definition 721.1;*
2. *obligation exposure, saying that the beta, application-argument, lambda-domain, and Pi-domain discharge rows are all displayed before R is consumed;*
3. *parameterized preservation exactness, saying that R may be read only as the theorem conditional on the four displayed obligation rows;*
4. *obstruction exactness, saying that O records the checked non-availability of the closed consistency reduction obstruction and is not a substitute for any of the four obligations;*
5. *audit non-escape from hidden substitution preservation, host conversion, erased typing derivations, semantic equality of terms, global normalization, or an unrecorded closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryCarrier obligation exposure certificate`
Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundary namecert obligations`

Proof. Carrier admission is the finite packet of Definition 721.1. Its rows display the four named discharge obligations, the parameterized preservation theorem, the checked obstruction theorem, the audit ledger, transport data, continuation routes, package provenance, and local naming certificate.

Obligation exposure is recorded by the ledger Q. A consumer reaches R only after the rows B, A, L, P_i have been displayed, so the parameterized theorem cannot be read as a closed theorem with the hypotheses erased.

Obstruction exactness is the role of O. It records the checked obstruction row described in Definition 9377.5; the row blocks replacement of the four-input interface by a blanket closed subject-reduction statement.

The non-escape clause follows from absence of coordinates. Hidden substitution preservation, host conversion, erased typing derivations, semantic equality, global normalization, and unrecorded closed subject reduction are not rows of M and cannot be reconstructed by hsame transport or a Cont read. □

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryNamecertObligatio`

Theorem 721.3 (MetaCIC subject-reduction boundary obligation exposure)**.** *Every consumer of the parameterized preservation row R in an accepted* MetaCICSubjectReductionBoundaryUp *packet must first display the four discharge rows B, A, L, P_i and the audit ledger Q. The row R is therefore*

consumed only as a theorem conditional on beta-substitution preservation, application-argument stability, lambda-domain subject reduction, and Pi-domain subject reduction; it is not a closed subject-reduction theorem and it does not erase the audit record.

Proof. Project the accepted carrier of Definition 721.1. The only rows that can discharge the preservation hypotheses are B, A, L, P_i, and the only row that records their displayed status is Q.

Apply the obligation-exposure clause of Definition 721.2. It requires the four discharge rows to be displayed before any consumer reads R.

The parameterized preservation exactness clause then fixes the interpretation of R. Since the packet has no coordinate for an unparameterized theorem and Q remains part of the visible route, a consumer cannot turn R into closed subject reduction or drop the audit ledger. □

Theorem 721.4 (MetaCIC subject-reduction boundary non-escape)**.** *Every accepted* MetaCICSubjectReductionBou *packet exposes MetaCIC subject reduction only through the rows*

$$B, A, L, P_i, R, O, Q, H, C, K, N.$$

Any preservation-facing consumer must factor through the four displayed discharge obligations and the parameterized theorem row, while any obstruction-facing consumer must factor through the checked obstruction row. No consumer receives an unparameterized closed subject-reduction theorem, a hidden substitution lemma, host conversion, erased typing derivation semantics, semantic equality of terms, or a global normalization oracle from the packet.

Proof. Read the carrier coordinates from Definition 721.1. The only preservation rows are the four obligations and the parameterized theorem, and the only obstruction row is O.

Apply the NameCert obligations of Definition 721.2. They require obligation exposure before R is consumed and keep the obstruction row separate from the discharge rows.

The exclusions are immediate from the packet inventory. Since the carrier has no coordinate for hidden substitution preservation, host conversion, erased derivation semantics, semantic equality, global normalization, or unparameterized closed subject reduction, no public route can export any of those objects. □

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryCarrier nonescape`

Theorem 721.5 (MetaCIC subject-reduction boundary typed substitution)**.** *Every typed-substitution read displayed by an accepted* MetaCICSubjectReductionBoundaryUp *packet preserves the finite subject-reduction boundary only through the retained rows*

$$B,\ A,\ L,\ P_i,\ R,\ Q,\ H,\ C,\ K,\ N.$$

The substitution-facing route may replay the beta-substitution row B and the parameterized theorem row R through the same context, term, type, substitution, and reduction windows recorded by Q, H, C, K, N. It cannot replace the four retained discharge rows by a hidden substitution lemma, a closed subject-reduction theorem, host conversion, erased derivation semantics, semantic equality of MetaCIC terms, or global normalization.

Lean4 ✓ `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryCarrier typed substitution boundary`

↪ variant `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryCarrier typed substitution`

Proof. Read the carrier of Definition 721.1. A typed-substitution request can only see the beta-substitution row B, the remaining discharge rows A, L, P_i, the parameterized theorem row R, and the structural rows Q, H, C, K, N.

The audit ledger Q keeps the context, term, type, substitution, and reduction windows explicit before R is consumed. Transport by H and replay by C preserve those displayed windows rather than creating a new substitution theorem.

Apply the non-escape theorem Definition 721.4. Since the forbidden objects are not rows of the packet, a typed-substitution read remains a boundary-preserving route through the retained finite data. □

Theorem 721.6 (MetaCIC subject-reduction boundary context non-escape)**.** *No context-facing consumer of an accepted* MetaCICSubjectReductionBoundaryUp *packet receives any global normalization, host conversion, or semantic equality row from the boundary packet. Every context extension, weakening, or retained-context read must factor through the displayed discharge rows B, A, L, P_i, the parameterized theorem row R, the checked obstruction row O, and the audit, transport, replay, provenance, and naming rows Q, H, C, K, N.*

Proof. Start with the finite inventory in Definition 721.1. The context-visible rows are exactly the four discharge rows, the parameterized theorem row, the obstruction row, and the structural audit and replay rows.

Use Definition 721.2. It separates obligation exposure from obstruction exactness, so a context read cannot move from retained local rows to an ambient conversion or semantic-equality principle.

Finally apply Definition 721.4. Global normalization, host conversion, and semantic equality are absent coordinates, hence no context-facing route exported by the packet can expose them. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionBoundaryUp}}$.

Constructive story: A MetaCICSubjectReductionBoundaryUp packet is generated from four displayed MetaCIC subject-reduction discharge rows, a parameterized preservation theorem row, a checked obstruction row, an audit ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionBoundaryUp}}$, scopedClosed)

Scope closed: Definition 721.1, Definition 721.2, Definition 721.4, Definition 721.5, and Definition 721.6 bind the MetaCIC subject-reduction obstruction as a finite boundary packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionBoundaryUp}}$, theoremChecked)

Lean target: `BEDC.Derived.MetaCICSubjectReductionBoundaryUp.MetaCICSubjectReductionBoundaryTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not prove any of the four discharge obligations, closed subject reduction, closed consistency from reduction, global normalization, host conversion, semantic equality, or bridge checking.

Upgrade path: Scoped closure requires binding each obligation row to the exact MetaCIC primitives and sibling obstruction routes named in the proof.

Chapter 722

A Concrete Naming Certificate for $\mathsf{ClosedTermContextSeal}^{\uparrow}$

The $\mathsf{ClosedTermContextSeal}^{\uparrow}$ packet realizes the MetaCIC subject-reduction obstruction of chapter 9377 as a finite closed-term context surface. It sits next to the subject-reduction carrier of chapter 483: $\mathsf{SubjectReduction}^{\uparrow}$ records an Ext-Cont preservation packet, while $\mathsf{ClosedTermContextSeal}^{\uparrow}$ records the empty-context, closedness, typed-membership, beta-strong-normality, handoff, obstruction, transport, route, provenance, and local naming rows that keep the closed-subject-reduction boundary inspectable.

Definition 722.1 (Closed-term context seal carrier)**.** *A* $\mathsf{ClosedTermContextSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, Q, T, B, S, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the empty-context row for the closed MetaCIC typing read;*
2. *Q is the closedness row for the subject term and its target readback;*
3. *T is the typed-membership row naming the closed source membership;*
4. *B is the beta-strong normality row used by the closed-normal consistency side;*
5. *S is the* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff row, pointing to the finite preservation packet of chapter 483;*
6. *O is the checked obstruction row for the unavailable unparameterized closed subject-reduction statement;*
7. *H records componentwise* hsame *transport for E, Q, T, B, S, O, P, N;*
8. *C records the displayed* Cont *routes by which consumers read the closedness, membership, normality, handoff, and obstruction coordinates;*
9. *P is the* Pkg *provenance over the MetaCIC closed-term rows, the subject-reduction handoff row,* BHist, hsame, Cont, *and the local naming row;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedTermContextSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a closed subject-reduction theorem, no hidden proof of the four MetaCIC discharge obligations, no ambient consistency shortcut, and no host equality principle for terms or contexts.

Theorem 722.2 (Closed-term context seal NameCert obligations)**.** *For the carrier of Definition 722.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedTermContextSeal}^{\uparrow}}$ *surface consists exactly of carrier admission, empty-context visibility, closedness visibility, typed-membership visibility, beta-strong-normality visibility,* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff visibility, obstruction exactness, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from unparameterized closed subject reduction.*

Lean4 ✓ `BEDC.Derived.ClosedTermContextSealUp.ClosedTermContextSeal namecert obligations`

Proof. Project the packet K. The rows $E, Q, T, B, S, O, H, C, P, N$ are the complete displayed inventory, so every local naming obligation is read from a finite BHist coordinate.

The closed-term side is exhausted by E, Q, T, B. These rows state the empty-context read, closedness read, typed-membership read, and beta-strong-normality read needed by the closed-normal consistency side of chapter 9377.

The preservation side is exhausted by S and O. The row S points only to the sibling $\mathsf{SubjectReduction}^{\uparrow}$ handoff surface, while O records the checked obstruction against replacing the parameterized interface by an unparameterized closed theorem.

The remaining rows H, C, P, N supply transport, route, provenance, and local naming for the same packet. Since no carrier coordinate names the stronger theorem or its four missing proofs, the stated obligations are exact. □

Theorem 722.3 (Closed-term context seal subject-reduction handoff)**.** *Every preservation-facing consumer of an accepted* $\mathsf{ClosedTermContextSeal}^{\uparrow}$ *packet reaches subject-reduction data only through the displayed handoff row S and the finite* $\mathsf{SubjectReduction}^{\uparrow}$ *surface of chapter 483. The consumer must still read the empty-context row E, closedness row Q, typed-membership row T, beta-strong-normality row B, obstruction row O, transports H, routes C, provenance P, and local name N from the seal packet before using that handoff as closed-term context data.*

Lean4 ✓ `BEDC.Derived.ClosedTermContextSealUp.ClosedTermContextSeal subject reduction handoff`

Proof. Start with the handoff coordinate S in Definition 722.1. It is the only row that points to subject-reduction data, and its sibling target is the finite $\mathsf{SubjectReduction}^{\uparrow}$ carrier rather than an ambient preservation theorem.

Apply Definition 722.2. It requires every preservation-facing read to keep S together with $E, Q, T, B, O, H, C, P, N$, so the handoff remains a closed-context seal read and cannot detach into a host subject-reduction assertion.

The sibling chapter chapter 483 supplies only source membership, reduction, target membership, ledger, transport, route, package, and naming rows. Combining that handoff with the seal packet therefore gives a finite route, not a closed subject-reduction theorem. □

Theorem 722.4 (Closed-term context seal non-escape)**.** *An accepted* $\mathsf{ClosedTermContextSeal}^{\uparrow}$ *packet exports no unparameterized closed subject-reduction theorem, no proof of any of the four MetaCIC discharge obligations, no hidden host conversion rule, no quotient equality of terms, and no ambient consistency principle. Its exported surface is exhausted by the empty-context row, closedness row, typed-membership row, beta-strong-normality row,* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff row, obstruction row, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row.*

Proof. Inspect the carrier coordinates. The only available rows are $E, Q, T, B, S, O, H, C, P, N$.

Use Definition 722.3 for preservation-facing reads and Definition 722.2 for local naming reads. Both routes stay inside the displayed finite packet and the sibling $\mathsf{SubjectReduction}^{\uparrow}$ handoff.

The excluded principles have no coordinate in the carrier. A request for closed subject reduction is therefore routed to the obstruction row O, while a request for ordinary subject-reduction data is routed to S and remains conditional on the sibling handoff surface. □

Theorem 722.5 (Closed-term context seal scoped kernel route)**.** *Every closed-subject-reduction consumer of an accepted* $\mathsf{ClosedTermContextSeal}^{\uparrow}$ *packet is scoped through the displayed rows* $E, Q, T, B, S, O, H, C, P$. *The empty-context row, closedness row, typed-membership row, beta-strong-normality row, and* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff row must be read before the consumer may use the packet as closed-term context data. The route exports no host conversion, quotient term equality, or ambient consistency principle.*

Proof. Read the carrier of Definition 722.1. Its rows name the empty context, closedness, typed membership, beta-strong-normality, sibling handoff, obstruction, transport, replay, provenance, and local name as a single finite packet.

Apply Definition 722.2 to keep the local naming obligations on that same row inventory, and apply Definition 722.3 for every preservation-facing read. The handoff is therefore available only after E, Q, T, B and S have been exposed together.

Finally use Definition 722.4. It removes the possible escape routes: host conversion, quotient term equality, ambient consistency, the four MetaCIC discharge obligations, and an unparameterized closed subject-reduction theorem all have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedTermContextSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedTermContextSeal}^{\uparrow}$ packet is generated from empty-context, closedness, typed-membership, beta-strong-normality, subject-reduction handoff, obstruction, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedTermContextSeal}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 722.1, Definition 722.2, Definition 722.3, Definition 722.4, and Definition 722.5 scope every closed-subject-reduction consumer through the finite rows $E, Q, T, B, S, O, H, C, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedTermContextSeal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ClosedTermContextSealUp.ClosedTermContextSeal\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not prove the four MetaCIC subject-reduction discharge obligations, closed subject reduction, ambient consistency, host conversion, quotient term equality, or bridge checking.

Upgrade path: Further closure requires a checked bridge for the same closed-term context seal route.

Chapter 723

A Concrete Naming Certificate for MetaCICSubjectReductionTrace$^{\uparrow}$

The MetaCICSubjectReductionTrace$^{\uparrow}$ packet turns the MetaCIC subject-reduction audit boundary into a finite BEDC naming surface. It records the parameterized subject-reduction theorem, the four discharge obligations consumed by that theorem, the checked obstruction row for the closed statement, componentwise transport, replay routes, package provenance, and a local NameCert row. The packet names the present proof frontier without claiming the four obligations or a closed subject-reduction theorem.

The trace packet sits between the obstruction boundary of chapter 731 and the discharge-obligation ledger of chapter 717. It records the parameterized theorem row and the four obligation sockets, while the sibling obstruction and frontier chapter chapter 730 record why this surface does not close the unparameterized subject-reduction theorem.

Definition 723.1 (MetaCIC subject-reduction trace carrier)**.** *A* MetaCICSubjectReductionTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (P, B, A, L, I, O, H, C, G, N)$$

with the following displayed rows:

1. *P records the parameterized theorem row for* BEDC.MetaCIC.subject_reduction*, including the four hypothesis sockets it consumes;*
2. *B records the beta-substitution obligation row* BetaSubstitutionPreservation*;*
3. *A records the application-argument-stability obligation row* AppArgTypeStable*;*
4. *L records the lambda-domain obligation row* LamDomainSubjectReduction*;*
5. *I records the Pi-domain obligation row* PiDomainSubjectReduction*;*
6. *O records the obstruction theorem row for* BEDC.MetaCIC.closed_consistency_reduction_obstruction_not_availa
7. *H records componentwise* hsame *transport for the theorem row, the four obligation rows, the obstruction row, replay routes, provenance, and local name;*
8. *C records the displayed* Cont *routes that replay consumers through the parameterized theorem row, the four obligation rows, and the obstruction row;*
9. *G is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the MetaCIC subject-reduction audit rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}}$ *row.*
The carrier has no coordinate for an unparameterized closed subject-reduction theorem, a hidden proof of any of the four obligations, an implicit host-calculus preservation theorem, or an axiom that discharges the obstruction.

Theorem 723.2 (MetaCIC subject-reduction trace NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}$ *carrier*

$$T = (P, B, A, L, I, O, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}}$ *surface consists exactly of the parameterized theorem row P, the four discharge-obligation rows B, A, L, I, the checked obstruction row O, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through G, and local naming through N.*

The packet records the four discharge obligations without proving them. A consumer that needs subject reduction must route through the parameterized theorem row P and supply B, A, L, I; a consumer that asks for the closed statement is routed to the obstruction row O.

Proof. Project the finite packet of Definition 723.1. The rows $P, B, A, L, I, O, H, C, G, N$ are the complete displayed inventory, so carrier admission and local naming are read directly from the packet.

The subject-reduction side is exhausted by P together with B, A, L, I. The row P names the checked parameterized theorem, while B, A, L, I name the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain sockets that must be supplied before that theorem yields a preservation step.

The obstruction side is exhausted by O. Since O records the checked obstruction for the closed consistency-reduction proposition, the packet does not convert the four obligation rows into proofs and does not export a closed subject-reduction theorem.

Transport, replay, provenance, and naming are confined to H, C, G, N. Thus the local $\mathsf{NameCert}$ surface is exactly the displayed trace packet. □

Theorem 723.3 (MetaCIC subject-reduction trace non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}$ *packet factors through*

$$P, B, A, L, I, O, H, C, G, N.$$

A preservation consumer may read the parameterized theorem row and the four named obligation rows. An audit consumer may read the checked obstruction row. No consumer can read a closed subject-reduction theorem, a hidden proof of any obligation, or an unrecorded host-calculus preservation principle from this packet.

Proof. Begin with the route row C. By the carrier definition, every public continuation route is displayed there and has endpoints among $P, B, A, L, I, O, H, G, N$.

Use the NameCert obligation surface of Definition 723.2. It separates the parameterized theorem row from the four obligation rows and from the obstruction row, so a consumer cannot collapse the rows into a stronger theorem.

Componentwise transport through H preserves the same row inventory. Package provenance through G and local naming through N bind the route to the finite packet, leaving no coordinate for a closed subject-reduction theorem or hidden discharge proof. □

Theorem 723.4 (MetaCIC subject-reduction trace discharge frontier)**.** *An accepted* MetaCICSubjectReductionTrace *packet excludes any closed subject-reduction theorem unless all four obligation rows*

$$B, A, L, I$$

are supplied as discharge witnesses for the parameterized theorem row P. The obstruction row O remains the consumer-facing answer for the closed statement whenever any one of B, A, L, I is only a named obligation row.

Proof. Read P as the parameterized subject-reduction theorem row. Its visible interface consumes exactly the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain discharge rows.

Read B, A, L, I as obligation rows rather than proof rows. The carrier records their names and positions, but it does not add a witness that any one of them is discharged.

Now read O as the checked obstruction row. If a consumer requests the closed theorem while any one of the four rows remains only an obligation, the replay route sends that request to O rather than to a closed subject-reduction theorem.

Finally, H, C, G, N preserve and package the same frontier. They transport, replay, and name the displayed rows, but they introduce no extra discharge coordinate. □

Theorem 723.5 (MetaCIC subject-reduction trace single-obligation obstruction)**.** *Let*

$$T = (P, B, A, L, I, O, H, C, G, N)$$

be an accepted MetaCICSubjectReductionTrace$^{\uparrow}$ *packet. If any one of B, A, L, I remains only a named obligation row rather than a supplied discharge witness, then a closed subject-reduction request is answered by O, and no downstream consumer may read a closed subject-reduction theorem from P alone.*

Concretely, the obstruction route applies in each of the following four cases:

1. *B is only the named beta-substitution obligation row;*
2. *A is only the named application-argument-stability obligation row;*
3. *L is only the named lambda-domain obligation row;*
4. *I is only the named Pi-domain obligation row.*

In all four cases, the transport, replay, provenance, and naming rows H, C, G, N preserve the same row inventory and introduce no hidden proof coordinate.

Proof. Project the carrier rows from Definition 723.1. The packet exposes P as the parameterized subject-reduction theorem row, exposes B, A, L, I as the four discharge-obligation rows, and exposes O as the obstruction row for the closed statement.

Use Definition 723.4. It states that a closed subject-reduction theorem is excluded unless all four rows B, A, L, I are supplied as discharge witnesses for P.

There are four single-missing cases.

1. If B remains only the beta-substitution obligation row, then the frontier theorem applies because not all four discharge witnesses are supplied.
2. If A remains only the application-argument-stability obligation row, then the same frontier theorem routes the closed request away from P alone.

3. If L remains only the lambda-domain obligation row, then the closed theorem is still absent from the packet inventory.
4. If I remains only the Pi-domain obligation row, then the request is again answered by the obstruction row O.

Finally apply the non-escape boundary of Definition 723.3. The rows H, C, G, N transport, replay, package, and name the displayed packet, so they preserve the same row inventory and do not add a hidden coordinate carrying the missing discharge proof. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}$ packet is generated from the parameterized MetaCIC subject-reduction theorem row, four named discharge-obligation rows, a checked obstruction row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 723.1, Definition 723.2, Definition 723.3, Definition 723.4, and Definition 723.5 specify the finite subject-reduction trace carrier, obligation surface, consumer non-escape boundary, discharge frontier, and per-row obstruction boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route exhaustion, obstruction replay, and proof rows for all four MetaCIC subject-reduction discharge obligations.

Chapter 724

A Concrete Naming Certificate for $\mathsf{MetaCICDischargeObligation}^{\uparrow}$

The $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ packet turns the four discharge obligations isolated in chapter 9377 into a finite NameCert carrier. It is a sibling of chapter 483: the latter records the accepted subject-reduction closure surface, while this packet records the four local inputs that must be supplied before the MetaCIC subject-reduction theorem can be consumed without hiding a closed theorem.

Definition 724.1 (MetaCIC discharge-obligation carrier)**.** *A* $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, A, L, P, H, C, G, N)$$

with the following displayed rows:

1. *B is the beta-substitution discharge row* BetaSubstitutionPreservation*;*
2. *A is the application-argument discharge row* AppArgTypeStable*;*
3. *L is the lambda-domain discharge row* LamDomainSubjectReduction*;*
4. *P is the Pi-domain discharge row* PiDomainSubjectReduction*;*
5. *H records componentwise* hsame *transport for the four discharge rows, the continuation routes, package provenance, and local naming;*
6. *C records the displayed* Cont *routes by which a consumer reads the four discharge rows and the support rows;*
7. *G is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the MetaCIC subject-reduction discharge interface;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligation}^{\uparrow}}$ *row.*

The carrier contains no unparameterized closed subject-reduction theorem, no hidden proof of any discharge row, no host conversion rule, no erased derivation semantics, and no semantic equality of MetaCIC terms.

Definition 724.2 (MetaCIC discharge-obligation classifier)**.** *The* $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ *classifier accepts exactly those carrier reads that factor through*

$$B, A, L, P, H, C, G, N.$$

The positive reads are the four discharge rows, their componentwise hsame *transports, the displayed* Cont *routes, the* Pkg *provenance row, and the local* NameCert *row. A request for closed subject reduction, host conversion, hidden substitution preservation, quotient transport, or an undisplayed MetaCIC premise is classified as outside the packet.*

Theorem 724.3 (MetaCIC discharge-obligation certificate surface)**.** *For the carrier of Definition 724.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$D = (B, A, L, P, H, C, G, N),$$

the four discharge rows B, A, L, P*, componentwise* hsame *transport through* H*, displayed continuation reads through* C*, package provenance through* G*, and local naming through* N*.*

Proof. Project the finite carrier. Its only discharge coordinates are B, A, L, P, and its only support coordinates are H, C, G, N.

Use the classifier of Definition 724.2. It accepts precisely reads through those rows, so the local certificate surface cannot introduce another MetaCIC premise.

Finally read the local NameCert row N. It names the same finite packet and therefore binds the certificate surface to the displayed carrier rather than to a hidden subject-reduction theorem. □

Theorem 724.4 (MetaCIC discharge-obligation parameterized interface)**.** *Every consumer that uses* $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ *as input to the MetaCIC subject-reduction interface must consume all four rows* B, A, L, P *as the beta-substitution, application-argument, lambda-domain, and Pi-domain discharge rows. The support rows* H, C, G, N *transport, replay, package, and name those four rows, but they do not add a fifth discharge row and do not produce a closed subject-reduction theorem.*

Proof. Begin with the four-row inventory in Definition 724.1. It names exactly the four hypotheses isolated by chapter 9377.

Apply the certificate-surface theorem. Any subject-reduction-facing read sees B, A, L, P together with transport, route, provenance, and naming support.

The sibling package in chapter 483 records the accepted subject-reduction closure surface. This packet instead supplies the finite obligation interface that a consumer must discharge before invoking that surface, so no closed theorem is exported from H, C, G, N. □

Theorem 724.5 (MetaCIC discharge-obligation non-escape)**.** *Every downstream read of an accepted* $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ *carrier factors through*

$$B, A, L, P, H, C, G, N.$$

It cannot inspect a hidden proof of beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, host conversion, erased derivation semantics, quotient equality, or an unrecorded MetaCIC reduction principle.

Proof. Read the request through the classifier. Positive reads factor through the four discharge rows or through the displayed support rows.

If the request asks for a closed theorem or for an undisplayed host principle, no coordinate of D can answer it. The request is therefore outside the accepted carrier rather than a hidden route through the certificate.

The transport, continuation, provenance, and naming rows preserve the same inventory. They make the four obligations reusable as finite BEDC data, but they do not create another proof-producing coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDischargeObligation}^{\uparrow}$ packet is generated from the four MetaCIC subject-reduction discharge rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 724.1, Definition 724.2, Definition 724.3, Definition 724.4, and Definition 724.5 specify the finite four-obligation carrier, classifier, certificate surface, consumer interface, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObligation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, four-row consumption, transport stability, route replay, provenance, local naming, and non-escape for the same finite packet.

Chapter 725

A Concrete Naming Certificate for $\mathsf{SubjectReductionObligation}^{\uparrow}$

The $\mathsf{SubjectReductionObligation}^{\uparrow}$ packet realizes the four MetaCIC subject-reduction discharge obligations of chapter 9377 as a finite BEDC naming surface. It is downstream of the subject-reduction carrier of chapter 483, the type-classifier membership route of chapter 476, and the continuation-trace packet of chapter 469. Its role is to name the obligation packet that a subject-reduction consumer must supply; it does not turn the blocked closed subject-reduction statement of Definition 9377.5 into a theorem.

Definition 725.1 (SubjectReductionObligation carrier)**.** *A* $\mathsf{SubjectReductionObligation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, A, L, I, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BetaSubstitutionPreservation *row of Definition 9377.1;*
2. *A is the* AppArgTypeStable *row of Definition 9377.2;*
3. *L is the* LamDomainSubjectReduction *row of Definition 9377.3;*
4. *I is the* PiDomainSubjectReduction *row of Definition 9377.4;*
5. *T is the parameterized MetaCIC subject-reduction consumption row, read only as a row that consumes B, A, L, I;*
6. *H records componentwise* hsame *transports for the four obligation rows, the consumption row, the route rows, provenance, and local naming row;*
7. *C records displayed* Cont *routes from subject-reduction consumers to the four obligation rows and to the parameterized consumption row;*
8. *P is the* Pkg *provenance over* $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionObligation}^{\uparrow}}$ *row.*

The carrier contains no closed subject-reduction theorem, no proof of any one of B, A, L, I, no hidden host conversion principle, no quotient equality of MetaCIC terms, and no route around the checked obstruction in Definition 9377.5.

Theorem 725.2 (SubjectReductionObligation NameCert obligations)**.** *For the carrier of Definition 725.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionObligation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$O = (B, A, L, I, T, H, C, P, N),$$

the four displayed MetaCIC discharge-obligation rows B, A, L, I*, the parameterized consumption row* T*, componentwise* hsame *transport through* H*, displayed continuation reads through* C*, package provenance through* P*, and local naming through* N*.*

A subject-reduction consumer may cite this packet only as a finite obligation packet: T *consumes the four rows named in section 9377.3, and the explicit obstruction of section 9377.4 remains the answer to any request for closed subject reduction.*

Proof. Project the finite carrier O. The rows B, A, L, I are precisely the four discharge slots, while T is the row that records their simultaneous consumption by the parameterized subject-reduction interface.

Next read the structural rows H and C. They transport the same four rows componentwise and replay them through displayed continuation routes, so a consumer cannot replace one obligation row by an unrecorded host premise.

Finally inspect P and N. The provenance row ties the packet to the subject-reduction, type-classifier-membership, and continuation-trace chapters named above, and the local NameCert row packages no coordinate beyond the displayed carrier. Hence the NameCert surface is exactly the finite obligation packet. □

Theorem 725.3 (SubjectReductionObligation non-escape)**.** *Every consumer read of an accepted* $\mathsf{SubjectReductionObligation}^{\uparrow}$ *packet factors through*

$$B, A, L, I, T, H, C, P, N.$$

The consumer may read the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain obligation rows, the parameterized consumption row, componentwise transports, displayed continuation routes, package provenance, and the local NameCert row. It cannot read a closed subject-reduction theorem, a proof of any one obligation, host conversion, quotient equality of MetaCIC terms, ambient reduction semantics, or an unrecorded discharge premise.

Proof. Start with carrier projection. Since the accepted object is the finite packet O, every public read must choose one of the displayed rows.

For a subject-reduction read, the only route to T passes through the four obligation rows B, A, L, I and their transports in H. The continuation row C records this route as an obligation replay rather than as a closed theorem.

For a closed subject-reduction request, the packet has no corresponding coordinate. The obstruction cited in Definition 9377.5 therefore remains outside the packet as the boundary for that request, and no consumer can manufacture a closed theorem from the obligation rows alone.

The provenance and local naming rows P, N add only package identity and certificate identity. They do not add host conversion, quotient equality, ambient reduction semantics, or a hidden proof of any discharge row. □

Theorem 725.4 (SubjectReductionObligation obstruction alignment)**.** *The* $\mathsf{SubjectReductionObligation}^{\uparrow}$ *packet is compatible with the explicit MetaCIC obstruction of chapter 9377. It exposes the four rows required by the parameterized subject-reduction theorem and preserves the negative boundary of Definition 9377.5: a consumer that lacks any one of* B, A, L, I *receives an obligation packet, not a closed subject-reduction result.*

Proof. Read the four obligation definitions in section 9377.3. They are represented one-for-one by B, A, L, I in the carrier.

Then read the consumption row T. It has no route that omits any of those four rows, because its only displayed subject-reduction inputs are the rows recorded in the same finite packet.

The obstruction principle supplies the boundary for the unparameterized closed statement. Since O has no closed-theorem coordinate and no hidden proof row, the accepted packet aligns with the obstruction rather than bypassing it. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionObligation}^{\uparrow}$ packet is generated from the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain discharge rows, a parameterized subject-reduction consumption row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionObligation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 725.1, Definition 725.2, Definition 725.3, and Definition 725.4 bind the finite four-obligation packet without closing the MetaCIC discharge rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionObligation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, Lean verification, host conversion, quotient equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exact four-row consumption, obstruction alignment, route exhaustion, provenance, and non-escape for the same finite packet.

Chapter 726

A Concrete Naming Certificate for $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$

The $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ packet gives the obstruction vision of chapter 9377 a finite BEDC naming surface. It reads the subject-reduction carrier of chapter 483, the closed generator audit packet of chapter 709, and the generator audit closure packet of chapter 711 as the surrounding ledger rows for the four MetaCIC discharge obligations. The packet does not prove those obligations; it names their position, their subject-reduction handoff, and the non-escape boundary that prevents an unrecorded closed theorem from being exported.

Definition 726.1 (MetaCIC reduction obstruction ledger carrier)**.** *A* $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, B, A, L, P, G, J, O, H, C, Q, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{SubjectReduction}^{\uparrow}$ *row consumed by downstream type-preservation packets;*
2. *B records the beta-substitution discharge row;*
3. *A records the application-argument stability discharge row;*
4. *L records the lambda-domain preservation discharge row;*
5. *P records the Pi-domain preservation discharge row;*
6. *G records the* $\mathsf{ClosedGeneratorAudit}^{\uparrow}$ *source for closed generator and substitution audit evidence;*
7. *J records the* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *source for generator-closure replay;*
8. *O records the obstruction row for the closed subject-reduction request;*
9. *H records componentwise* hsame *transport for the subject-reduction row, the four discharge rows, the two generator-audit rows, the obstruction row, replay routes, provenance, and local naming;*
10. *C records displayed* Cont *routes through which consumers read the parameterized subject-reduction row, the four discharge rows, the generator-audit rows, and the obstruction row;*
11. *Q is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{ClosedGeneratorAudit}^{\uparrow}$, $\mathsf{GeneratorAuditClosure}^{\uparrow}$, *and the MetaCIC subject-reduction obstruction boundary;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a closed subject-reduction theorem, hidden proof of a discharge row, host-calculus preservation theorem, quotient transport, classical selection, proposition extensionality, or unrecorded proof oracle.

Theorem 726.2 (MetaCIC reduction obstruction ledger NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ *carrier*

$$R = (S, B, A, L, P, G, J, O, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}}$ *surface has four finite obligations:*

1. *subject-reduction exposure:* S *is the only displayed subject-reduction row available to a consumer;*
2. *discharge inventory:* B, A, L, P *are the only rows that may feed the beta-substitution, application-argument, lambda-domain, and Pi-domain sockets;*
3. *generator-audit anchoring:* G *and* J *are the only rows from which closed-generator and generator-closure audit evidence may be read;*
4. *obstruction exactness:* O, H, C, Q, N *route every closed subject-reduction request back to the displayed obstruction ledger unless all four discharge rows are supplied as proof rows for* S*.*

Proof. Project the carrier of Definition 726.1. The rows $S, B, A, L, P, G, J, O, H, C, Q, N$ are the complete packet inventory, so the local name has no source outside that tuple.

The subject-reduction read is confined to S. The four discharge rows B, A, L, P are listed separately, and the generator-audit rows G, J remain attached to their own displayed coordinates.

The obstruction row O handles the closed request. Transport through H, replay through C, provenance through Q, and naming through N keep that request inside the same finite packet rather than replacing it by an unrecorded theorem. □

Theorem 726.3 (MetaCIC reduction obstruction ledger non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ *carrier can read only the rows*

$$S, B, A, L, P, G, J, O, H, C, Q, N.$$

In particular, the packet exports no closed subject-reduction theorem, hidden discharge proof, host-calculus preservation principle, quotient transport, classical selection, proposition extensionality, or unrecorded proof oracle.

Proof. Start with the finite carrier. Each exported coordinate is one of the displayed rows, and each public replay route is recorded in C.

Apply Definition 726.2. The subject-reduction row, four discharge rows, generator-audit rows, and obstruction row exhaust the local $\mathsf{NameCert}$ surface.

Since the excluded objects do not occur as coordinates of R, componentwise transport and package provenance cannot export them. A consumer may only replay the rows named by the ledger. □

Theorem 726.4 (MetaCIC reduction obstruction ledger subject-reduction handoff)**.** *An accepted* $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ *packet hands subject-reduction consumers to the displayed* $\mathsf{SubjectReduction}^{\uparrow}$ *row* S *together with the four obligation rows* B, A, L, P*. A consumer obtains a subject-reduction read only by supplying those four rows as the discharge inputs required by* S*; otherwise the closed request is answered by the obstruction row* O*.*

Proof. Read the handoff coordinate S from the carrier. It is the only subject-reduction row recorded in the packet.

The rows B, A, L, P are the four displayed discharge inputs. The carrier does not contain a second source that could supply those inputs invisibly.

If the four rows are not supplied as proof rows for S, the route C sends the closed subject-reduction request to O. Thus the handoff is finite and explicit: either the consumer supplies the displayed obligations, or the ledger returns the obstruction row. □

Theorem 726.5 (MetaCIC reduction obstruction ledger vision concretization)**.** *The* MetaCICReductionObstruction *carrier realizes the MetaCIC subject-reduction obstruction vision as a finite NameCert ledger. Its rows link the four discharge obligations to* $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{ClosedGeneratorAudit}^{\uparrow}$, *and* $\mathsf{GeneratorAuditClosure}^{\uparrow}$ *while keeping the closed subject-reduction request behind the explicit obstruction row.*

Proof. Use Definition 726.1 to identify the ledger rows. The subject-reduction row S, the four discharge rows B, A, L, P, and the generator-audit rows G, J are all visible coordinates.

Use Definition 726.4 for the subject-reduction route and Definition 726.3 for the exclusion boundary. Together they give the vision a concrete finite packet without adding a closed theorem or a hidden discharge proof. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}$ packet is generated from a subject-reduction row, four discharge-obligation rows, closed-generator and generator-closure audit rows, an obstruction row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 726.1, Definition 726.2, Definition 726.3, Definition 726.4, and Definition 726.5 specify the finite obstruction ledger carrier, obligation surface, non-escape boundary, subject-reduction handoff, and vision realization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICReductionObstructionLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, discharge inventory, generator-audit anchoring, and obstruction exactness for the displayed ledger.

Chapter 727

A Concrete Naming Certificate for BetaSubstitutionDischarge$^{\uparrow}$

The BetaSubstitutionDischarge$^{\uparrow}$ packet isolates the beta-redex row named in chapter 9377. It is narrower than the full SubjectReduction$^{\uparrow}$ packet of chapter 483: it records only the substitution-across-binder discharge needed by the beta case, while remaining compatible with the type-membership boundary of chapter 476 and the compiler-facing handoff of chapter 625.

Definition 727.1 (BetaSubstitutionDischarge carrier)**.** *A* BetaSubstitutionDischarge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, d, b, a, c, s, h, r, p, n)$$

with the following displayed rows:

1. *G is the source context row for the beta redex;*
2. *d is the displayed domain row;*
3. *b is the body typing row in the context extended by the shifted domain;*
4. *a is the argument typing row in the original context;*
5. *c is the codomain row whose displayed substitution is read after the argument is supplied;*
6. *s is the substitution ledger tying the body, argument, domain, and codomain rows to the beta-redex discharge;*
7. *h records componentwise* hsame *transports for G, d, b, a, c, s and for the provenance rows;*
8. *r records the displayed* Cont *routes from the beta step to the substituted body and substituted codomain reads;*
9. *p is the* Pkg *provenance over* BHist, hsame, Cont, *the MetaCIC beta-step interface,* SubjectReduction$^{\uparrow}$, TypeClassifierMembership$^{\uparrow}$, *and* NameCert*;*
10. *n is the local* NameCert$_{\text{BetaSubstitutionDischarge}^{\uparrow}}$ *row.*

The carrier has no coordinate for a full closed subject-reduction theorem, no global conversion oracle, no erased derivation representative, and no proof row for any non-beta discharge obligation.

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeTasteGate single carrier alignment`

Theorem 727.2 (BetaSubstitutionDischarge NameCert obligations)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier*

$$B = (G, d, b, a, c, s, h, r, p, n),$$

the local $\mathsf{NameCert}_{\mathsf{BetaSubstitutionDischarge}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, beta-redex source coverage through* G, d, b, a, c*, substitution-ledger exactness through* s*, componentwise* hsame *stability through* h*, displayed* Cont *replay through* r*, package provenance through* p*, and local naming through* n*. A consumer may use the packet only as the* BetaSubstitutionPreservation *row consumed by the beta case of the MetaCIC subject-reduction interface.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischarge namecert obligations`

Proof. Project the displayed carrier. The rows G, d, b, a, c identify the context, domain, body typing, argument typing, and codomain data of one beta redex, while s is the only substitution ledger in the packet.

The transport and replay rows are then forced. Componentwise hsame movement through h can carry the displayed rows to an equivalent packet, and the Cont routes in r replay the beta step to the substituted body and substituted codomain reads. Neither row contains an independent typing theorem.

Finally read p and n as provenance and local naming data. Since the carrier lists no global subject-reduction theorem, no conversion oracle, no erased derivation row, and no non-beta discharge row, every accepted consumer of the packet receives exactly the beta-substitution preservation obligation named by the MetaCIC interface. □

Theorem 727.3 (BetaSubstitutionDischarge binder boundary)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier* $B = (G, d, b, a, c, s, h, r, p, n)$*, every binder-crossing read factors through the single substitution ledger* s *together with the displayed* BHist *rows* G, d, b, a, c*. The route may preserve the rows through* h *and replay them through* r*, but it cannot replace the binder boundary by host substitution, ambient conversion, quotient equality, or a closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeCarrier binder boundary`

Proof. Start with the binder data. The body row b is typed only in the context obtained from G by the displayed domain row d, and the argument row a is typed only over the original context row G.

The substituted codomain read is controlled by c and s. The ledger s is therefore the unique row that crosses the binder boundary from the body and argument data to the beta-redex discharge.

Transport through h and replay through r preserve that inventory. They may move the same rows componentwise and expose the beta route to a $\mathsf{SubjectReduction}^{\uparrow}$ or $\mathsf{TypeClassifierMembership}^{\uparrow}$ consumer, but they add no host substitution rule, quotient equality, conversion theorem, or closed subject-reduction row.

The package row p and local name row n record the same boundary for downstream certificates, including the $\mathsf{TypePreservingCompiler}^{\uparrow}$ handoff. Hence the binder-crossing read remains the single beta-substitution discharge packet. □

Theorem 727.4 (BetaSubstitutionDischarge component transport)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier* $B = (G, d, b, a, c, s, h, r, p, n)$*, every componentwise* hsame *transport recorded by* h *preserves exactly the source context row* G*, displayed domain row* d*, body typing row* b*, argument typing row* a*, codomain row* c*, substitution ledger* s*, replay row* r*, provenance row* p*, and local naming row* n*. The transported packet therefore has the same beta-redex discharge surface and gains no host substitution rule, quotient equality, ambient conversion theorem, or closed subject-reduction row.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeCarrier component transport`

Proof. Project the carrier and read the transport row. The row h is componentwise, so each move is attached to one displayed coordinate of $G, d, b, a, c, s, r, p, n$ rather than to an external substitution principle.

The source, domain, body, argument, codomain, and substitution rows are therefore carried as the same finite beta-redex inventory. No transported coordinate adds a host substitution rule or quotient equality, because those objects are absent from the carrier.

The replay, provenance, and naming rows are preserved with the same boundary. Hence the transported packet remains a $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ packet over the displayed beta-redex surface and not a closed subject-reduction theorem. □

Theorem 727.5 (BetaSubstitutionDischarge replay exactness)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier* $B = (G, d, b, a, c, s, h, r, p, n)$*, the displayed* Cont *replay row* r *exposes only the beta-redex substituted-body read and the substituted-codomain read determined by* G, d, b, a, c *and the substitution ledger* s*. It exposes no application-argument stability route, lambda-domain preservation route, Pi-domain preservation route, conversion oracle, quotient equality, or non-beta subject-reduction discharge.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeCarrier replay exactness`

Proof. Read the replay row after the binder-boundary data. The rows G, d, b, a, c identify the beta redex and s is the only ledger that carries the body and codomain across the binder.

The Cont routes in r can therefore replay exactly the substituted-body read and the substituted-codomain read. Any application, lambda, Pi, conversion, quotient, or non-beta discharge would require an additional carrier coordinate.

Since the carrier lists no such coordinate, replay exactness is confined to the beta-redex reads named by the substitution ledger. The provenance and naming rows record that same confinement for downstream MetaCIC consumers. □

Theorem 727.6 (BetaSubstitutionDischarge context induction)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier* $B = (G, d, b, a, c, s, h, r, p, n)$*, every beta-substitution discharge read is obtained by induction over the displayed source context row* G *and the single substitution ledger* s*. The induction preserves the displayed domain, body, argument, and codomain rows* d, b, a, c *through the componentwise transport row* h*, and exposes only the substituted-body and substituted-codomain reads through the replay row* r*.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeCarrier context induction`

Proof. Project the carrier of Definition 727.1. The only context data are the source row G, the domain row d, the body row b, the argument row a, the codomain row c, and the substitution ledger s.

Induct over the displayed context-extension window named by G. At the binder-crossing step, use Definition 727.3; the step can cross the body and codomain boundary only through s, not through host substitution or a closed subject-reduction theorem.

For stability of the induction data, use component transport, Definition 727.4. It keeps d, b, a, c, s on the same displayed packet under h.

Finally apply replay exactness, Definition 727.5. The replay row r exposes precisely the substituted-body and substituted-codomain reads, so the induction has no remaining route to an application, lambda, Pi, conversion, quotient, or non-beta discharge coordinate. □

Theorem 727.7 (BetaSubstitutionDischarge obligation surface)**.** *For an accepted* $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ *carrier* $B = (G, d, b, a, c, s, h, r, p, n)$*, carrier admission, substitution-ledger exactness, binder-boundary nonescape, componentwise transport, replay exactness, and context induction exhaust the displayed beta-redex obligation surface consumed by the MetaCIC subject-reduction beta case. No consumer of the packet receives an application-argument stability row, lambda-domain preservation row, Pi-domain preservation row, conversion oracle, quotient equality, erased derivation representative, or closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.BetaSubstitutionDischargeUp.BetaSubstitutionDischargeCarrier obligation surface`

Proof. Project the carrier of Definition 727.1. The rows G, d, b, a, c expose the beta-redex source context, domain, body typing, argument typing, and codomain data, while s is the only substitution ledger attached to those rows.

Apply Definition 727.2 and Definition 727.3. The first fixes the local NameCert surface, and the second forces every binder-crossing read to pass through s rather than through host substitution or a closed subject-reduction theorem.

Use Definition 727.4 and Definition 727.5. Transport preserves the displayed coordinates componentwise, and replay exposes only the substituted-body and substituted-codomain reads determined by the substitution ledger.

Finally apply Definition 727.6. The induction over G and s leaves no remaining coordinate from which an application, lambda, Pi, conversion, quotient, erased-derivation, or non-beta discharge obligation could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BetaSubstitutionDischarge}^{\uparrow}}$.

Constructive story: A $\mathsf{BetaSubstitutionDischarge}^{\uparrow}$ packet is generated from a displayed source context row, domain row, body typing row, argument typing row, codomain row, substitution ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BetaSubstitutionDischarge}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 727.1, Definition 727.2, Definition 727.3, Definition 727.4, Definition 727.5, Definition 727.6, and Definition 727.7 bind the finite beta-substitution discharge carrier and the beta-redex obligation surface consumed by the MetaCIC subject-reduction beta case.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BetaSubstitutionDischarge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, Lean verification, host conversion, quotient equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, substitution-ledger exactness, binder-boundary coverage, componentwise transport stability, replay-route exactness, and public non-escape for the displayed beta-redex packet.

Chapter 728

A Concrete Naming Certificate for $\mathsf{BetaCongruenceDomainAudit}^{\uparrow}$

The $\mathsf{BetaCongruenceDomainAudit}^{\uparrow}$ packet realizes the binder-domain trigger surface identified in chapter 9384. It sits beside the beta-step boundary of chapter 754 and the beta-substitution preservation packet of chapter 716: those siblings expose the raw beta relation and the redex-substitution preservation row, while this packet binds the `congLamDom` and `congPiDom` domain-trigger rows as finite subject-reduction audit data.

Falsifiable prediction: A packet omitting the displayed binder-domain trigger row is not an accepted $\mathsf{BetaCongruenceDomainAudit}^{\uparrow}$ carrier.

Independence witness: betastepboundary, betasubstitutionpreservation, metacic-subject-reduction-obstruction, subjectreductiondischarge: no sibling is bijective because the new packet stores the binder-domain beta-congruence trigger, displayed domain readback, subject-reduction handoff row, transport, route, provenance, and local naming together.

Definition 728.1 (BetaCongruenceDomainAudit carrier)**.** *A* $\mathsf{BetaCongruenceDomainAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, M, \Pi, R, S, H, K, P, N)$$

with the following displayed rows:

1. *T is the binder-domain trigger row, tagged by the local* `congLamDom` *or* `congPiDom` *beta constructor;*
2. *M is the lambda-domain readback row for the displayed old and new lambda annotations, when the trigger is the lambda-domain constructor;*
3. *Π is the Pi-domain readback row for the displayed old and new Pi annotations, when the trigger is the Pi-domain constructor;*
4. *R is the subject-reduction handoff row naming the binder-domain discharge surface consumed by* $\mathsf{SubjectReductionDischarge}^{\uparrow}$*;*
5. *S is the finite subject-reduction trigger ledger tying T, M, Π, R to the same domain-congruence beta step;*
6. *H stores componentwise* hsame *transports for the trigger, readback, handoff, ledger, route, provenance, and naming rows;*

7. *K stores the displayed* Cont *routes for reading the trigger, replaying the domain readback, and handing the row to the subject-reduction discharge interface;*
8. *P is the* Pkg *provenance over chapter 9384, chapter 754, chapter 716, chapter 718,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaCongruenceDomainAudit}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no beta-redex substitution semantics, no typed preservation theorem, no unrestricted conversion, and no closed subject-reduction theorem.

Theorem 728.2 (BetaCongruenceDomainAudit NameCert obligations)**.** *For the carrier of Definition 728.1, the local* $\mathsf{NameCert}_{\mathsf{BetaCongruenceDomainAudit}^{\uparrow}}$ *surface consists of carrier admission for*

$$D = (T, M, \Pi, R, S, H, K, P, N),$$

binder-domain trigger readback, lambda-domain readback, Pi-domain readback, subject-reduction handoff replay, finite trigger-ledger exactness, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and non-escape from beta-redex substitution semantics, typed preservation, unrestricted conversion, and closed subject reduction.*

Proof. Carrier admission is projection from Definition 728.1. The packet exposes exactly $T, M, \Pi, R, S, H, K, P, N$.

The trigger row T records whether the beta step is the lambda-domain constructor or the Pi-domain constructor named in chapter 9384. The readback rows M and Π keep the displayed old and new domain annotations separate from any redex-substitution row.

The handoff row R and ledger row S place the trigger beside the subject-reduction discharge surface without discharging it. The transports H, routes K, provenance P, and local name N bind the read into one finite NameCert packet.

Since no coordinate supplies beta-redex substitution semantics, typed preservation, unrestricted conversion, or closed subject reduction, no consumer of the local certificate can export those objects from the binder-domain trigger packet. □

Theorem 728.3 (BetaCongruenceDomainAudit trigger handoff)**.** *An accepted* $\mathsf{BetaCongruenceDomainAudit}^{\uparrow}$ *packet presents the* `congLamDom` *and* `congPiDom` *beta constructors as finite subject-reduction trigger rows. The handoff to* $\mathsf{SubjectReductionDischarge}^{\uparrow}$ *reads only*

$$D = (T, M, \Pi, R, S, H, K, P, N),$$

where T is the binder-domain trigger, M and Π are the displayed domain readbacks, R is the discharge-facing row, S is the finite trigger ledger, and H, K, P, N are the local transport, route, provenance, and naming rows.

Proof. First read the trigger from T. The beta-step boundary sibling records that `congLamDom` and `congPiDom` are beta constructors, while this carrier stores which binder-domain constructor is being audited.

Next read the displayed domain annotations. The lambda-domain case uses M, and the Pi-domain case uses Π; each route is replayed through H and K before the handoff row R is visible.

Finally apply the provenance row P. It links the trigger to the beta audit map, beta-step boundary, beta-substitution preservation sibling, and subject-reduction discharge surface, but it does not identify the binder-domain trigger with redex substitution or typed preservation. □

Theorem 728.4 (BetaCongruenceDomainAudit non-escape)**.** *Every accepted* BetaCongruenceDomainAudit$^{\uparrow}$ *packet exposes binder-domain beta-congruence only through the rows*

$$T, M, \Pi, R, S, H, K, P, N.$$

A downstream consumer may read the binder-domain trigger, displayed lambda-domain readback, displayed Pi-domain readback, discharge-facing handoff, finite trigger ledger, transports, routes, provenance, and local naming row. It cannot replace those rows by beta-redex substitution semantics, typed preservation, unrestricted conversion, or closed subject reduction.

Proof. Project the carrier. The only beta-facing row is T, and the only binder annotation readbacks are M and Π.

Use the NameCert obligations of Definition 728.2. They require the handoff and ledger rows to remain tied to the same trigger packet through H, K, P, N.

The exclusions follow from absence of coordinates. Redex substitution, typed preservation, unrestricted conversion, and closed subject reduction would each require a separate row or external route, and none is present in D. □

Theorem 728.5 (BetaCongruenceDomainAudit TasteGate target)**.** *The Lean-facing taste gate for* BetaCongruenceDomainAudit$^{\uparrow}$ *is the finite carrier discipline of Definition 728.1 together with the obligation, handoff, and non-escape theorems of this chapter. The downstream taste-gate module* `Derived/BetaCongruenceDomainAuditUp/TasteGate.lean` *must provide* `Nontrivial BetaCongruenceDomainAu` *and* `FieldFaithful BetaCongruenceDomainAuditUp` *instances for the rows* $T, M, \Pi, R, S, H, K, P, N$.

Lean4 ✓ `BEDC.Derived.BetaCongruenceDomainAuditUp.BetaCongruenceDomainAuditTasteGate single carrier alignment`

Proof. The carrier theorem fixes the finite row inventory. The NameCert obligations identify the admission, readback, handoff, ledger, transport, route, provenance, and naming fields.

The handoff theorem shows the packet is not merely a beta-step boundary: it has a discharge-facing route for binder-domain triggers. The non-escape theorem shows it is not a typed preservation theorem or unrestricted conversion rule.

These three facts are exactly the paper-side target for a nontrivial instance and a field-faithful instance: an accepted carrier has the displayed rows, and each row is recoverable without adding hidden host semantics. □

Closure status for NameCert$_{\text{BetaCongruenceDomainAudit}^{\uparrow}}$.

Constructive story: A BetaCongruenceDomainAudit$^{\uparrow}$ packet is generated from beta-domain admission rows, readback rows, a discharge-facing handoff, a binder-domain trigger ledger, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The BHist carrier records where beta congruence is allowed to be consumed and where the domain audit must refuse an unrestricted conversion read. Its classifier first exposes the admission surface, then the handoff and trigger ledger, and only then lets downstream consumers read the audit row. The route is finite and syntactic, not a host conversion theorem.

Theory closure: TheoryGate(NameCert$_{\text{BetaCongruenceDomainAudit}^{\uparrow}}$, seed)

Scope closed: This chapter binds the beta congruence domain audit carrier, handoff theorem, non-escape theorem, and paper-side TasteGate target.

Formal verification: FormalStatus(NameCert$_{\text{BetaCongruenceDomainAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim unrestricted conversion, typed preservation, subject reduction, or host beta-equivalence semantics.

Upgrade path: To reach obligationClosure, formalize the beta-domain admission and discharge-trigger rows as nontrivial and field-faithful carrier data.

Chapter 729

A Concrete Naming Certificate for $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$

The $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$ packet records the four MetaCIC discharge rows named in chapter 9377 as a finite BEDC ledger adjacent to the ordinary $\mathsf{SubjectReduction}^{\uparrow}$ certificate of chapter 483. It does not close subject reduction. It exposes the exact obligation surface that a later discharge must fill before any closed MetaCIC preservation claim can be read.

Definition 729.1 (Subject-reduction obligation ledger carrier)**.** *A* $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, A, D, I, R, O, H, C, P, N)$$

with the following displayed rows:

1. *B records the* BetaSubstitutionPreservation *discharge row consumed by the beta case;*
2. *A records the* AppArgTypeStable *discharge row consumed by application-argument congruence;*
3. *D records the* LamDomainSubjectReduction *discharge row consumed by lambda-domain congruence;*
4. *I records the* PiDomainSubjectReduction *discharge row consumed by Pi-domain congruence;*
5. *R records the parameterized subject-reduction theorem row whose visible inputs are exactly B, A, D, I;*
6. *O records the obstruction row for the unavailable closed subject-reduction statement;*
7. *H records componentwise* hsame *transport for the four obligation rows, R, O, package provenance, and local naming;*
8. *C records the displayed* Cont *routes from obligation reads to the parameterized theorem row and from closed-statement requests to the obstruction row;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the MetaCIC subject-reduction interface, and the earlier* $\mathsf{SubjectReduction}^{\uparrow}$ *packet;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionObligationLedger}^{\uparrow}}$ *row.*

The carrier contains no proof of any one of the four discharge rows, no closed subject-reduction theorem, no hidden substitution lemma, no host conversion relation, and no semantic equality model for MetaCIC terms.

Theorem 729.2 (Subject-reduction obligation ledger NameCert obligations)**.** *For every accepted* $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$ *carrier* $L = (B, A, D, I, R, O, H, C, P, N)$*, the local* $\mathsf{NameCert}$ *surface consists of five finite rows:*

1. *carrier admission for the displayed packet* L*;*
2. *obligation inventory for the beta, application-argument, lambda-domain, and Pi-domain discharge rows* B, A, D, I*;*
3. *parameterized theorem routing, which permits* R *to be consumed only after all four obligation rows have been displayed;*
4. *obstruction routing, which sends any closed subject-reduction request not backed by the four rows to* O*;*
5. *non-escape provenance through* H, C, P, N*, excluding hidden discharge proofs, host conversion, erased typing derivations, semantic equality, and a global preservation theorem.*

Proof. Carrier admission is the tuple of Definition 729.1. The four discharge coordinates B, A, D, I are displayed separately, so a consumer can inspect which obligation row is present without receiving a proof of that obligation.

The parameterized theorem route is the R coordinate. Its continuation row in C has the four obligation rows as displayed inputs, and the package row P ties that route to the MetaCIC subject-reduction interface and to the earlier $\mathsf{SubjectReduction}^{\uparrow}$ certificate surface.

The obstruction route is the O coordinate. It is the consumer-facing answer for a closed statement request when the four rows have not all been discharged; it is not a substitute for any one of B, A, D, I.

The transport and naming rows H, C, P, N complete the ledger. Since the carrier has no coordinate for hidden substitution preservation, host conversion, erased typing derivations, semantic equality, or global preservation, these five rows exhaust the local certificate obligations. □

Theorem 729.3 (Subject-reduction obligation ledger non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$ *packet factors through*

$$B, A, D, I, R, O, H, C, P, N.$$

Preservation-facing consumers may read the four displayed obligation rows and the parameterized theorem row. Audit-facing consumers may read the obstruction row. No consumer can extract a closed MetaCIC subject-reduction theorem, a hidden proof of beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, host conversion, erased typing semantics, or semantic equality of terms from the packet.

Proof. Start with the finite carrier. The only preservation-facing rows are B, A, D, I, R, and the only obstruction-facing row is O.

Apply Definition 729.2. The obligation inventory separates the four rows, the parameterized theorem route requires those rows before R is consumed, and the obstruction route keeps a closed-statement request attached to O.

Finally read the transport, continuation, package, and naming rows. They preserve and replay the same finite inventory, but none of them adds a proof row or a host semantic object. Thus every consumer remains inside $B, A, D, I, R, O, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionObligationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionObligationLedger}^{\uparrow}$ packet is generated from four displayed MetaCIC subject-reduction discharge rows, the parameterized theorem row, the obstruction row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionObligationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 729.1, Definition 729.2, and Definition 729.3 bind the four discharge rows and the obstruction route as one finite obligation ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionObligationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove beta substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, closed subject reduction, closed consistency from reduction, global normalization, or bridge checking.

Upgrade path: Formal work remains to encode the ledger carrier and then supply checked rows for the four MetaCIC discharge obligations.

Chapter 730

A Concrete Naming Certificate for $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$

The $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$ packet names the finite subject-reduction boundary that remains after the parameterized MetaCIC theorem has been isolated. It records exactly the four discharge obligations consumed by that theorem, together with transport, replay, provenance, and local naming rows. The packet is a frontier object: it exposes the boundary without asserting a closed subject-reduction theorem.

The frontier consumes the obstruction packet of chapter 731 as the finite boundary separating named discharge obligations from a closed subject-reduction theorem.

Definition 730.1 (MetaCIC subject-reduction frontier carrier)**.** *A* $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, A, L, P, H, C, G, N)$$

with the following displayed rows:

1. *B is the* BetaSubstitutionPreservation *obligation row;*
2. *A is the* AppArgTypeStable *obligation row;*
3. *L is the* LamDomainSubjectReduction *obligation row;*
4. *P is the* PiDomainSubjectReduction *obligation row;*
5. *H records componentwise* hsame *transport for the four obligation rows, displayed replay, provenance, and local naming;*
6. *C records the displayed* Cont *routes by which a preservation consumer reads the four obligation rows;*
7. *G is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the four MetaCIC subject-reduction discharge rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}$ *row.*

The carrier has no row for the parameterized theorem itself, no row for a closed subject-reduction theorem, and no row that turns an obligation name into a proof of that obligation.

Theorem 730.2 (MetaCIC subject-reduction frontier obligation surface)**.** *For every accepted* $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$ *carrier*

$$F = (B, A, L, P, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}$ *surface consists exactly of the four discharge-obligation rows* B, A, L, P*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* G*, and local naming through* N*.*

Proof. Project the carrier of Definition 730.1. Its only subject-reduction coordinates are B, A, L, P, so the visible preservation boundary is the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain obligation surface.

The transport row H preserves those four coordinates componentwise, and the continuation row C displays the only routes by which a consumer may read them. Provenance through G and local naming through N package the same finite inventory as a $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}$ row.

Since the carrier contains no theorem coordinate and no hidden discharge coordinate, the local naming surface is exactly the displayed frontier packet. □

Theorem 730.3 (MetaCIC subject-reduction frontier non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$ *carrier factors through*

$$B, A, L, P, H, C, G, N.$$

The consumer may learn which four MetaCIC subject-reduction obligations must be supplied, and may transport or replay that finite list through the displayed rows. It cannot read a closed subject-reduction theorem, a proof of any listed obligation, an unlisted preservation principle, or an axiom from the frontier packet.

Proof. Begin with the displayed continuation row C. It names the consumer routes for the four obligation rows and supplies no endpoint outside B, A, L, P, H, G, N.

Next use componentwise transport through H. Transport preserves the same four obligation coordinates rather than adding a theorem coordinate or a discharge witness.

Finally, package the result through G and N. Package provenance and local naming certify the finite frontier surface, while the missing coordinates in the carrier rule out escape into a closed theorem, hidden proof, unlisted preservation principle, or axiom. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$ packet is generated from the beta-substitution, application-argument-stability, lambda-domain, and Pi-domain subject-reduction obligation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 730.1, Definition 730.2, and Definition 730.3 specify the finite obligation frontier and its consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close beta substitution, application-argument stability, lambda-domain preservation, Pi-domain preservation, parameterized subject reduction, closed subject reduction, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the frontier carrier, route exhaustion, and checked non-escape for the four MetaCIC subject-reduction discharge rows.

Chapter 731

A Concrete Naming Certificate for MetaCICSubjectReductionObstruction$^{\uparrow}$

The MetaCICSubjectReductionObstruction$^{\uparrow}$ packet turns the MetaCIC subject-reduction obstruction into finite BEDC data. It records the checked parameterized subject-reduction theorem, the four named discharge obligations, the closed-term consistency obstruction theorem, the audit consumer, and the NameCert rows that keep this surface separate from a closed subject-reduction theorem.

The obstruction packet is consumed beside chapter 717 for the four discharge slots and beside chapter 739 for the closed-normal branch. The present carrier keeps those sibling surfaces separate, so the missing subject-reduction edge cannot be read as a full consistency theorem.

Definition 731.1 (MetaCIC subject-reduction obstruction carrier)**.** *A* MetaCICSubjectReductionObstruction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (T, B, A, L, \Pi, K, U, H, C, P, N)$$

with the following displayed rows:

1. *T is the checked parameterized MetaCIC subject-reduction theorem row, naming* BEDC.MetaCIC.subject_reductio *as a four-input interface;*
2. *B is the beta-substitution preservation obligation row consumed by the beta case;*
3. *A is the application-argument stability obligation row consumed by application-argument congruence;*
4. *L is the lambda-domain subject-reduction obligation row consumed by lambda-domain congruence;*
5. *Π is the Pi-domain subject-reduction obligation row consumed by Pi-domain congruence;*
6. *K is the checked obstruction row naming* BEDC.MetaCIC.closed_consistency_reduction_obstruction_not_availab
7. *U is the audit-consumer row for self-compilation, consistency, and manuscript citation checks;*
8. *H records componentwise* hsame *transports for the theorem row, four obligation rows, obstruction row, audit consumer, and local structural rows;*
9. *C records the displayed* Cont *routes for reading the parameterized theorem, the four obligations, the obstruction theorem, and the audit consumer without leaving the packet;*

10. *P is the* Pkg *provenance over MetaCIC typing, beta reduction, closed-term consistency,* BHist*,* hsame*,* Cont*, and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the visible rows $T, B, A, L, \Pi, K, U, H, C, P, N$.

Theorem 731.2 (MetaCIC subject-reduction obstruction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICSubject}}$ *surface for Definition 731.1 consists of carrier admission, four-obligation exposure, parameterized-theorem exactness, checked-obstruction readback, audit-consumer accountability, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from an unparameterized closed subject-reduction theorem.*

Proof. Carrier admission is projection from the displayed tuple O. The theorem row T, the four obligation rows B, A, L, Π, the obstruction row K, the audit consumer U, and the structural rows H, C, P, N are all finite coordinates.

The four-obligation exposure is componentwise: B names beta substitution preservation, A names application-argument stability, L names lambda-domain subject reduction, and Π names Pi-domain subject reduction. The parameterized theorem row T consumes exactly those obligation rows and does not erase them.

Checked-obstruction readback is the role of K. The audit consumer can inspect T, B, A, L, Π, K only through C, while H, P, N record transport, provenance, and local naming. Since the carrier has no coordinate for a closed subject-reduction theorem, non-escape follows by inspection of the packet. □

Theorem 731.3 (MetaCIC subject-reduction obstruction consumer boundary)**.** *Every self-compilation, consistency, or manuscript consumer of an accepted* $\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}$ *packet reads the MetaCIC subject-reduction boundary only through*

$$T, B, A, L, \Pi, K, U, H, C, P, N.$$

The consumer may cite the parameterized theorem row, inspect the four obligation rows, and read the checked obstruction row, but it receives no closed subject-reduction theorem, hidden substitution lemma, host normalization principle, or ambient consistency shortcut.

Proof. Start from the carrier in Definition 731.1. The consumer coordinate U names the downstream audit route, and the continuation coordinate C records how that route reads the finite packet.

Apply Definition 731.2. It forces the consumer to see the parameterized theorem together with the four discharge obligations and the checked obstruction row.

The remaining rows H, P, N supply only transport, package provenance, and local naming for the same packet. None of the coordinates names a closed subject-reduction theorem, a hidden substitution lemma, a host normalization principle, or an ambient consistency shortcut, so the consumer boundary is exact. □

Theorem 731.4 (MetaCIC subject-reduction obstruction no-escape)**.** *An accepted* MetaCICSubjectReductionObstru *carrier cannot be used to recover an unparameterized closed subject-reduction theorem. Agreement on the rows $T, B, A, L, \Pi, K, U, H, C, P, N$ fixes the public obstruction surface, but it does not identify any proof of the four discharge obligations, any closed subject-reduction proof, or any stronger consistency theorem outside the finite packet.*

Proof. Use the classifier clause from the carrier definition. Public comparison is restricted to the theorem row, the four obligation rows, the obstruction row, the audit consumer, and the displayed structural rows.

The NameCert obligations expose K as the checked obstruction readback and expose T as a parameterized theorem row. Thus the public surface records both what is available and which four inputs remain required.

If a consumer tried to read a closed subject-reduction proof, it would need a coordinate outside $T, B, A, L, \Pi, K, U, H, C, P, N$. No such coordinate is present. The packet therefore fixes the obstruction boundary without manufacturing any proof beyond the displayed MetaCIC rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}$ packet is generated from the parameterized subject-reduction theorem row, the four discharge-obligation rows, the checked obstruction theorem row, an audit-consumer row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 731.1, Definition 731.2, Definition 731.3, and Definition 731.4 bind the MetaCIC subject-reduction obstruction as a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close the four MetaCIC discharge obligations, unparameterized closed subject reduction, closed-term consistency beyond the checked obstruction surface, Lean verification for this packet, or a public standard bridge.

Upgrade path: Formal closure requires checked carrier admission, four-obligation exposure, parameterized-theorem exactness, checked-obstruction readback, consumer accountability, transport stability, route stability, provenance, and non-escape for the same finite packet.

Chapter 732

A Concrete Naming Certificate for $\mathsf{NormalFalseExclusion}^{\uparrow}$

The $\mathsf{NormalFalseExclusion}^{\uparrow}$ packet exposes the closed-normal exclusion row from chapter 9375 as finite BEDC naming data. It sits beside the subject-reduction obstruction packet of chapter 731: the present chapter names the already-normal false-proof exclusion, while the sibling obstruction records why arbitrary closed proofs are not transported into that normal fragment.

Definition 732.1 (NormalFalseExclusion carrier)**.** *A* $\mathsf{NormalFalseExclusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, C, B, Y, H, R, P, N)$$

with the following visible rows:

1. *T is the closed MetaCIC proof-term row whose type read is internal False;*
2. *C records the closedness witness for T;*
3. *B records the* $\mathsf{BetaStrong}$ *normality witness for T;*
4. *Y is the internal False typing row consumed by the MetaCIC consistency assembly;*
5. *H records componentwise* hsame *transports for the closedness, normality, false typing, theorem row, provenance, and local naming rows;*
6. *R records displayed* Cont *routes for reading the closed proof-term row, the normality row, and the false typing row without leaving the packet;*
7. *P is the* Pkg *provenance over MetaCIC typing, closedness, beta-strong normality,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a reduction sequence from an arbitrary closed proof, a subject-reduction theorem, a host consistency theorem, a quotient proof identity, or a hidden normalization selector.

Theorem 732.2 (NormalFalseExclusion NameCert obligations)**.** *For every accepted* $\mathsf{NormalFalseExclusion}^{\uparrow}$ *carrier*

$$F = (T, C, B, Y, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}$ *surface consists of carrier admission, closed proof-term exposure, beta-strong normality exposure, internal False typing exposure, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from arbitrary closed consistency or subject-reduction transport.*

Proof. Project the carrier of Definition 732.1. The proof-term row, closedness row, normality row, false typing row, transport row, route row, provenance row, and local naming row are all displayed coordinates.

The closed-normal obligations are componentwise. The row C supplies closedness, B supplies beta-strong normality, and Y supplies internal False typing for the same proof-term row T.

The structural rows then restrict all consumer access. Transports through H, routes through R, provenance through P, and naming through N replay only the displayed rows, so the local certificate has no source for arbitrary reduction, subject reduction, quotient proof identity, or host consistency. □

Theorem 732.3 (NormalFalseExclusion displayed route admission)**.** *For an accepted* $\mathsf{NormalFalseExclusion}^{\uparrow}$ *carrier*

$$F = (T, C, B, Y, H, R, P, N),$$

the displayed rows T, C, B, Y, H, R, P, N *are the only admitted route from a closed term row, its closedness witness, its beta-strong normality witness, and its internal False typing row to the local* $\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}$ *row. No route is admitted through subject reduction, arbitrary normalization, quotient proof identity, host consistency, or an external proof-search coordinate.*

Proof. Project the carrier in Definition 732.1. Its visible input side is exactly the closed proof-term row T, closedness row C, beta-strong normality row B, and false typing row Y.

Apply Definition 732.2. The structural rows H, R, P, N are the only transport, route, provenance, and local naming rows available to the certificate.

Finally use the boundary inventory of Definition 732.6. Since the checked closed-normal exclusion consumes precisely the displayed rows and the carrier has no subject-reduction or arbitrary-normalization coordinate, the admitted route is exactly the listed finite route. □

Theorem 732.4 (NormalFalseExclusion route stability)**.** *Componentwise* hsame *transport through* H *and displayed* Cont *replay through* R *preserve exactly the route*

$$T, C, B, Y, H, R, P, N$$

from the closed-normal false-proof input rows to the local $\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}$ *row. They do not create a second route through reduction semantics, global normalization, quotient proof identity, or host consistency.*

Proof. Start with the route admitted by Definition 732.3. The only transport-bearing row is H, and the only continuation-bearing row is R.

Use Definition 732.2 to replay those rows componentwise. The replay can move the displayed closedness, normality, false typing, provenance, and naming coordinates, but it cannot introduce an unlisted coordinate.

The closed-normal boundary of Definition 732.6 is therefore stable under the same replay. The non-displayed routes remain absent because the carrier still contains no subject-reduction, normalization, quotient-identity, or host-consistency row after transport. □

Theorem 732.5 (NormalFalseExclusion consumer route exactness)**.** *Every consumer of the checked* no_closed_normal_proof_of_false *row factors through the displayed route rows*

$$T, C, B, Y, H, R, P, N$$

of the accepted NormalFalseExclusion$^{\uparrow}$ *carrier. Such a consumer cannot read subject reduction, arbitrary normalization, quotient proof identity, host consistency, or any hidden consistency principle.*

Proof. Project the consumer surface from Definition 732.1. Any consumer starts with the finite tuple F and therefore sees only the rows T, C, B, Y, H, R, P, N.

Invoke Definition 732.2 and Definition 732.3. These identify the route from the closed-normal input rows to the local naming row and make the structural rows part of the same route.

Close by Definition 732.6 and Definition 732.8. The checked exclusion rejects the displayed closed-normal false-proof input, while the non-escape inventory rules out every stronger read demanded by arbitrary closed consistency. □

Theorem 732.6 (NormalFalseExclusion closed-normal boundary)**.** *An accepted* NormalFalseExclusion$^{\uparrow}$ *carrier cannot contain a closed beta-strong normal proof term whose internal type is False. Equivalently, any attempted read of the displayed rows T, C, B, Y is rejected by the MetaCIC closed-normal false-proof exclusion theorem before it can be exported as a consumer-facing proof row.*

Lean4 ✓ `BEDC.MetaCIC.no closed normal proof of false`

Proof. Read the closedness, beta-strong normality, and internal False typing rows from the carrier. These are exactly the inputs named by the closed-normal exclusion node in chapter 9375.

Apply the checked MetaCIC exclusion theorem. It rules out a closed proof term of internal False once beta-strong normality is already part of the input.

The conclusion is a boundary statement for the displayed packet. It rejects the closed-normal false-proof row, but it does not derive a reduction route from arbitrary closed proofs into normal form. □

Theorem 732.7 (NormalFalseExclusion subject-reduction boundary)**.** *The* NormalFalseExclusion$^{\uparrow}$ *packet names only the already-normal false-proof exclusion. It does not supply the subject-reduction route needed to reduce an arbitrary closed proof of internal False to the normal-form input of Definition 732.6; that route remains outside the packet and is recorded by chapter 731.*

Proof. Inspect the carrier coordinates. They include a proof-term row, closedness, beta-strong normality, internal False typing, structural transport, routes, provenance, and local naming.

No coordinate records a beta-reduction path from an arbitrary closed proof to T. No coordinate records a subject-reduction theorem along such a path.

Thus the packet can invoke the closed-normal exclusion only when normality is already displayed. The neighboring subject-reduction obstruction chapter names the missing route and prevents this packet from being read as a full closed consistency certificate. □

Theorem 732.8 (NormalFalseExclusion consumer non-escape)**.** *Every consumer of an accepted* NormalFalseExclusion$^{\uparrow}$ *packet can read only the finite rows*

$$T, C, B, Y, H, R, P, N.$$

The consumer may use the checked closed-normal exclusion on those rows, but it receives no arbitrary closed-proof normalization, no subject-reduction discharge, no host consistency principle, no quotient proof identity, and no hidden proof search oracle.

Proof. Start from Definition 732.1. The tuple F lists the whole consumer surface.

Use Definition 732.2 for the certificate obligations. It confines reads to the closed proof-term row, the normality row, the internal False typing row, and the displayed structural rows.

Finally use Definition 732.6. The checked exclusion rejects the closed-normal false-proof input itself, while the absence of arbitrary normalization and subject-reduction coordinates blocks every stronger consistency read. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalFalseExclusion}^{\uparrow}$ packet is generated from a closed proof-term row, a beta-strong normality row, an internal False typing row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 732.1, Definition 732.2, Definition 732.3, Definition 732.4, Definition 732.5, Definition 732.6, Definition 732.7, and Definition 732.8 bind the carrier, NameCert obligations, displayed route admission, route stability, consumer route exactness, closed-normal boundary, subject-reduction boundary, and consumer non-escape as the closed packet scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalFalseExclusion}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.MetaCIC.no\_closed\_normal\_proof\_of\_false`

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary closed consistency, subject reduction, normalization of arbitrary closed proof terms, quotient proof identity, host consistency, or bridge checking.

Upgrade path: Scoped closure requires the same displayed closed-normal packet to bind carrier admission, route stability, consumer exactness, boundary, and non-escape as one closed scope.

Chapter 733

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$

The $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$ packet realizes the closed-normal MetaCIC consistency assembly of chapter 9375 as finite BEDC data. It sits beside the obstruction ledger of chapter 726 and the subject-reduction frontier of chapter 730: those packets display the blocked lift, while this one records the positive normal-form consistency node and the exact boundary that prevents it from becoming full closed consistency.

Definition 733.1 (Closed-normal consistency boundary carrier)**.** *A* $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, C, N, F, O, R, H, Q, P, M)$$

with the following displayed rows:

1. *T is the empty-context MetaCIC typing row for the closed proof candidate;*
2. *C is the closedness row for the same term;*
3. *N is the beta-strong normality row consumed by the closed-normal consistency theorem;*
4. *F is the displayed internal false-type row;*
5. *O is the checked no-closed-normal-false row and the closed-normal consistency assembly row read as positive obstruction-to-false data;*
6. *R is the subject-reduction refusal row inherited from the closed reduction obstruction;*
7. *H records componentwise* hsame *transport for the typing, closedness, normality, false-type, positive theorem, refusal, route, provenance, and naming rows;*
8. *Q records the displayed* Cont *routes by which a consumer reads the positive closed-normal node before it reaches the refusal boundary;*
9. *P is the* Pkg *provenance over the MetaCIC consistency assembly, the reduction obstruction ledger, the subject-reduction frontier,* BHist, hsame, Cont, *and local certificate rows;*
10. *M is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unparameterized closed subject-reduction theorem, no route from arbitrary closed proofs to beta-strong normal proofs, no hidden discharge of beta substitution or dependent-codomain stability, and no host consistency theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyBoundaryUp.ClosedNormalConsistencyBoundaryTasteGate single carrier alignment`

Theorem 733.2 (Closed-normal consistency boundary NameCert obligations)**.** *For an accepted* $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$ *carrier* $B = (T, C, N, F, O, R, H, Q, P, M)$*, the local* NameCert *surface exposes exactly the empty-context typing row* T*, closedness row* C*, beta-strong normality row* N*, false-type row* F*, positive closed-normal consistency row* O*, subject-reduction refusal row* R*, componentwise transport* H*, displayed replay routes* Q*, package provenance* P*, and local naming* M*. A consumer may eliminate only the closed normal false-proof configuration displayed by* T, C, N, F, O*; it may not use the packet as a full closed consistency theorem.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyBoundaryUp.ClosedNormalConsistencyBoundaryUp taste gate boundary`

Proof. Project the carrier of Definition 733.1. The rows T, C, N, F display the precise closed normal false-proof input consumed by the positive theorem row O.

Use the assembly chapter chapter 9375. Its positive node is closed-normal consistency, so the elimination route requires beta-strong normality as an input row rather than deriving it from an arbitrary closed proof.

Use the sibling obstruction ledger chapter 726 and the subject-reduction frontier chapter 730. They account for the missing lift through the displayed refusal and frontier rows, which this packet records as R instead of erasing.

The support rows H, Q, P, M transport, replay, package, and name the same inventory. Since no further coordinate is present, the local certificate exposes the closed-normal boundary and does not export closed subject reduction or full closed consistency. □

Theorem 733.3 (Closed-normal consistency boundary non-discharge)**.** *The* $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$ *packet does not discharge the MetaCIC subject-reduction frontier. Every request to turn an arbitrary closed false proof into the beta-strong input required by* O *must pass through the refusal row* R *and the sibling frontier packets of chapter 726 and chapter 730; no compiler semantics, host normalization theorem, quotient conversion, hidden discharge proof, or ambient consistency principle can replace those rows.*

Proof. Begin with the positive route. The carrier exposes O only together with T, C, N, F, so the theorem it packages is the closed-normal node described in chapter 9375.

Now consider a request for full closed consistency. Such a request needs a route from arbitrary closed typing to the beta-strong row N while preserving the false-type row F.

The only row addressing that route is R. By the construction of the packet, R points outward to the obstruction ledger and subject-reduction frontier rather than supplying the missing preservation theorem internally.

Finally inspect the remaining support rows. Transport, replay, provenance, and local naming preserve the displayed packet, but none of them adds compiler semantics, quotient conversion, a hidden discharge proof, or an ambient consistency principle. Thus the boundary is recorded without discharging it. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$ packet is generated from empty-context typing, closedness, beta-strong normality, internal false typing, the positive closed-normal consistency row, the subject-reduction refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 733.1, Definition 733.2, and Definition 733.3 bind the finite positive-normal node and its blocked closed-consistency lift.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClosedNormalConsistencyBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove closed subject reduction, beta-substitution preservation, dependent-codomain stability, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exact positive-theorem consumption, refusal replay, provenance accounting, transport stability, and non-escape for the same finite packet.

Chapter 734

A Normal-Fragment Certificate for $\mathsf{ClosedNormalConsistency}^{\uparrow}$

This chapter turns the closed-normal node of chapter 9375 into a concrete finite packet. It sits next to chapter 731: the present packet records the positive beta-strong normal fragment, while the sibling obstruction packet records why that fragment is not silently promoted to closed subject reduction.

Definition 734.1 (Closed-normal consistency packet)**.** *A* $\mathsf{ClosedNormalConsistency}^{\uparrow}$ *packet is a finite* BHist *carrier*

$$Q = (T, F, S, K, H, C, P, N)$$

with the following rows:

1. *T is the empty-context MetaCIC typing row for a closed proof candidate;*
2. *F is the internal false-type row named by the displayed typing certificate;*
3. *S is the beta-strong normality row used by the closed-normal theorem;*
4. *K is the checked closed-normal consistency row naming* BEDC.MetaCIC.closed_normal_consistency_assembly*;*
5. *H records componentwise* hsame *transports for the typing, false-type, beta-strong, and theorem rows;*
6. *C records the displayed* Cont *routes by which a consumer reads those rows;*
7. *P is the* Pkg *provenance over MetaCIC typing, closedness, beta-strong normality,* BHist*,* hsame*,* Cont*, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}$ *row.*

The carrier has no coordinate for host consistency, closed subject reduction, hidden normalization transport, or an ambient logical shortcut.

Definition 734.2 (Closed-normal consistency classifier)**.** *The classifier for* $\mathsf{ClosedNormalConsistency}^{\uparrow}$ *compares accepted packets only through the visible rows T, F, S, K, H, C, P, N. Two packets are classifier-equivalent exactly when the empty-context typing row, false row, beta-strong normality row, checked closed-normal theorem row, componentwise transports, displayed routes, package provenance, and local naming row agree through the displayed* hsame *data.*

Theorem 734.3 (Closed-normal consistency normal-fragment boundary)**.** *Every accepted* ClosedNormalConsistency *packet exposes a positive consistency statement only for the beta-strong normal fragment. A consumer may read T, F, S, K, H, C, P, N and conclude that a closed beta-strong proof candidate typed at the internal false row is impossible through the checked closed-normal row K. The packet does not assert that an arbitrary closed proof candidate reaches such a normal row while preserving its displayed type.*

Proof. Start from the carrier of Definition 734.1. Its theorem coordinate K names the closed-normal assembly row, and the coordinates T, F, S provide exactly the displayed typing, false-type, and beta-strong inputs read by that row.

The structural rows H, C, P, N keep the same finite evidence under componentwise transport, route replay, provenance, and local naming. They do not add a reduction sequence or a type-preservation theorem.

Thus the consumer obtains the beta-strong normal contradiction and nothing outside that fragment. Host consistency, closed subject reduction, hidden normalization transport, and ambient logical shortcuts are absent from the carrier. □

Theorem 734.4 (Closed-normal consistency subject-reduction frontier)**.** *The* ClosedNormalConsistency$^{\uparrow}$ *packet is separate from the subject-reduction frontier recorded by chapter 731. Agreement on T, F, S, K, H, C, P, N does not supply the four MetaCIC subject-reduction discharge rows, a closed subject-reduction theorem, or a transport from arbitrary closed proofs to beta-strong normal inhabitants of the same displayed false type.*

Proof. Read the classifier in Definition 734.2. It fixes the positive normal-fragment packet by the rows T, F, S, K, H, C, P, N and has no coordinates for the beta-substitution, application-argument, lambda-domain, or Pi-domain discharge rows.

Compare this with the sibling obstruction packet in chapter 731. That chapter names the missing subject-reduction surface as a different finite packet, so it cannot be recovered by reclassifying the closed-normal rows.

Consequently the closed-normal packet cannot transport arbitrary closed proofs to the normal fragment. It keeps the positive theorem and the subject-reduction frontier as distinct BEDC objects. □

Theorem 734.5 (Closed-normal consistency NameCert obligation surface)**.** *The local* NameCert$_{\text{ClosedNormalConsistency}}$ *obligation surface consists of carrier admission, classifier exactness, closed-normal theorem readback, beta-strong normal-fragment exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and non-escape from host consistency, closed subject reduction, hidden normalization transport, and ambient logical shortcuts.*

Proof. Carrier admission and classifier exactness are supplied by Definition 734.1 and Definition 734.2. They restrict all visible data to T, F, S, K, H, C, P, N.

The closed-normal theorem readback is the row K, consumed together with the typing row T, false row F, and beta-strong row S. The structural rows H, C, P, N preserve that same finite surface through transport, route replay, provenance, and local naming.

Non-escape follows by inspection of the packet. None of its rows names host consistency, closed subject reduction, hidden normalization transport, or an ambient logical shortcut, so the NameCert surface cannot export those claims. □

Theorem 734.6 (Closed-normal consistency scoped route)**.** *Every scoped read of a* ClosedNormalConsistency$^{\uparrow}$ *packet must expose the empty-context typing row, internal false row, beta-strong normality row,*

checked closed-normal theorem row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}$ *row. The same read separately preserves the subject-reduction frontier of chapter 731: it does not import the four subject-reduction discharge rows, a closed subject-reduction theorem, or a transport from arbitrary closed proofs to beta-strong normal inhabitants of the same false type.*

Proof. The carrier and classifier of Definition 734.1 and Definition 734.2 restrict every accepted packet to T, F, S, K, H, C, P, N. These rows display the empty-context typing, false-type, beta-strong, checked theorem, transport, replay, provenance, and local naming data.

The normal-fragment boundary of Definition 734.3 supplies the positive read only inside the beta-strong fragment. The subject-reduction frontier of Definition 734.4 keeps the sibling obstruction packet separate from that read.

Finally, the NameCert obligation surface of Definition 734.5 records the same finite rows as the local obligation boundary. Composing these displays yields a scoped route for the normal-fragment certificate while preserving the subject-reduction frontier as a distinct BEDC object. □

Corollary 734.7 (Closed-normal consistency strict local obstruction)**.** *For an accepted* $\mathsf{ClosedNormalConsistency}^{\uparrow}$ *packet, the obstruction carried by the normal-fragment boundary is strict and local. It blocks only the converse route that would turn an arbitrary closed proof candidate into a beta-strong normal inhabitant of the same displayed false row, and only the reuse route that would treat* T, F, S, K, H, C, P, N *as evidence for host consistency, closed subject reduction, hidden normalization transport, or an ambient logical shortcut. The displayed separation cannot be reinterpreted as a global impossibility theorem, a host theorem, or hidden external validation outside the named* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}$ *surface.*

Proof. By Definition 734.3, every consumer-facing positive use of the packet passes through the visible rows T, F, S, K, H, C, P, N. The only contradiction exposed there is the beta-strong normal-fragment contradiction read through the checked row K together with the typing, false-type, and beta-strong rows T, F, S. The same theorem also excludes the converse route from an arbitrary closed proof candidate to such a normal row while preserving the displayed type.

By Definition 734.5, the local obligation surface is exhausted by carrier admission, classifier exactness, closed-normal theorem readback, beta-strong normal-fragment exposure, componentwise hsame transport, displayed Cont route replay, Pkg provenance, and local non-escape. In particular, none of the available rows names host consistency, closed subject reduction, hidden normalization transport, or an ambient logical shortcut.

Therefore any attempted consumer interpretation outside those rows must ask for either the excluded converse route or an unlisted reuse coordinate. The packet supplies neither. Its obstruction is consequently confined to those named local routes, and it has no content as a global impossibility claim, host-level theorem, or external validation principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 734.1, Definition 734.2, Definition 734.3, Definition 734.4, Definition 734.5, and Definition 734.6 enumerate the carrier, classifier, closed-normal readback, beta-strong exposure, transport and replay, provenance, subject-reduction frontier obligations, and scoped route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.MetaCIC.closed\_normal\_consistency\_assembly`

Bridge status: none

Not claimed: This chapter does not close host consistency, closed subject reduction, hidden normalization transport, Lean verification, public bridge checking, or an ambient logical shortcut.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, closed-normal theorem readback, beta-strong normal-fragment exposure, transport and replay stability, provenance non-escape, and the subject-reduction frontier.

Constructive story: A ClosedNormalConsistency$^{\uparrow}$ packet is generated from empty-context typing, an internal false row, beta-strong normality, the checked closed-normal theorem row, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 735

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$

The $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ packet gives the confluence audit map of chapter 9378 a finite BEDC landing. It is a sibling of the subject-reduction boundary packets chapter 719, chapter 723, and chapter 717: those chapters separate the dependent typing discharge rows, while the present packet separates the residual substitution boundary for MetaCIC confluence.

Definition 735.1 (MetaCIC confluence audit packet carrier)**.** *A* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

with the following displayed rows:

1. *D is the conditional* BetaParallel *diamond row, naming the substitution-compatibility input and the parallel diamond obtained from it;*
2. *C is the closed-diamond-to-star-confluence row, naming the route from a closed parallel diamond to closed* BetaStarStep *confluence;*
3. *N records the unconditional closed-normal confluence fragment;*
4. *A_0 records the atom, sort, closed-variable-impossible, and non-lambda-application shape fragments that do not consume the full substitution input;*
5. *S is the substitution-compatibility boundary row, stating that the packet names the input and the obstruction without proving the input;*
6. *R is the subject-reduction separation row, linking the confluence boundary to the sibling subject-reduction packets without merging the two interfaces;*
7. *H stores componentwise* hsame *transports for the displayed rows;*
8. *T stores the displayed* Cont *routes by which consumers read the conditional bridges, the unconditional fragments, and the non-claim boundary;*
9. *P is the* Pkg *provenance over MetaCIC reduction,* BetaParallel, BetaStarStep, *closedness,* BHist, hsame, Cont, *and* NameCert*;*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}$ *row.*

The local classifier relates two carriers exactly when their ten displayed rows compare componentwise through H and are replayed through the routes in T. The carrier contains no coordinate proving BetaParallelSubstitutionCompatibility*, no coordinate proving* BetaParallelDiamondClosed*, and no coordinate for arbitrary closed-term Church–Rosser.*

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceAuditPacketUp.MetaCICConfluenceAuditPacketTasteGate`
`single carrier alignment`

Theorem 735.2 (MetaCIC confluence audit packet obligations)**.** *For the carrier of Definition 735.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for* $A = (D, C, N, A_0, S, R, H, T, P, L)$*, conditional parallel-diamond exposure, closed-diamond-to-star-confluence exposure, closed-normal and atom fragment exposure, substitution-boundary exposure, subject-reduction separation, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Proof. Project the accepted packet through Definition 735.1. The rows D and C give the conditional bridge surface: D exposes the substitution-sensitive parallel diamond, and C exposes the route from a closed parallel diamond to closed star confluence.

The rows N and A_0 give the unconditional fragment surface. They record closed-normal confluence and the atom or impossible-source cases without adding the full substitution-compatibility row as a proved theorem.

The boundary rows S and R then fix the negative reading. The first names the substitution compatibility input as a boundary, and the second keeps the confluence audit separate from the sibling subject-reduction discharge packets. Transport, route replay, provenance, and local naming are exactly the rows H, T, P, L, so the displayed coordinates exhaust the obligation surface. □

Theorem 735.3 (MetaCIC confluence audit packet conditional boundary)**.** *An accepted* MetaCICConfluenceAuditP *carrier may be cited for the conditional route from substitution compatibility to a parallel diamond and for the conditional route from a closed parallel diamond to closed* BetaStarStep *confluence. It may not be cited as a proof of* BetaParallelSubstitutionCompatibility*, as a proof of* BetaParallelDiamondClosed*, or as arbitrary closed-term Church–Rosser.*

Proof. Inspect the first two rows of the carrier. The row D records a bridge that consumes the substitution-compatibility input, and the row C records a bridge that consumes the closed parallel-diamond input.

The row S is the explicit substitution-boundary row. It states that the input is named and routed, not generated by the packet.

Since the carrier has no coordinate whose projection is the input itself, no consumer can turn the displayed bridges into unconditional endpoints. The accepted readings are therefore conditional bridge citations and the listed unconditional fragments, not the full confluence claims. □

Theorem 735.4 (MetaCIC confluence audit packet closed substitution boundary)**.** *An accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier may expose* BetaParallelClosedSubstitutionCompatibility *only as a closed-face boundary whose inputs include* ClosedAt 0 *a and* ClosedAt 1 *b. The unclosed substitution obstruction remains outside that face, so the packet may not cite the obstruction as a closed-diamond proof or as a discharge of the closed Church–Rosser substitution input.*

Proof. Project the row S. It records substitution compatibility as a named boundary, and Definition 735.3 already states that the generic substitution input is not produced by the packet.

The closed face is narrower than the generic input. Its stated compatibility requires the closedness hypotheses ClosedAt 0 a and ClosedAt 1 b, while the Lean-side theorem `betaParallel_closed_substitution_` records the obstruction that prevents the unclosed path from supplying those closed inputs.

Therefore a downstream closed Church–Rosser consumer still needs an explicit closed compatibility row. The audit packet can route the boundary and can name the obstruction, but it cannot convert the obstruction into the closed substitution premise demanded by the closed diamond route. □

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceAuditPacketUp.MetaCICConfluenceAuditPacketClosedSubstitutionBound`

Theorem 735.5 (MetaCIC confluence audit closed-face coverage)**.** *Let*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

be an accepted MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Every closed-face confluence consumer of A that cites the closed diamond route must first read the rows $D, C, N, A_0, S, H, T, P, L$ together with the displayed hypotheses* ClosedAt 0 a *and* ClosedAt 1 b*. The row R may be read only as the subject-reduction separation pointer to neighboring discharge packets, not as a closed-face confluence row.*

Proof. Project the accepted carrier through Definition 735.1. The rows available to a closed-face confluence consumer are the conditional diamond row D, the closed-diamond-to-star row C, the normal and source-shape rows N, A_0, the substitution-boundary row S, and the structural rows H, T, P, L.

Compose the two boundary theorems. Theorem 735.3 keeps the route through D, C conditional, while Definition 735.4 requires the displayed closedness hypotheses before the closed substitution face may be cited.

The subject-reduction row is separate. The row R points to the neighboring subject-reduction discharge packets, so it can record a horizontal dependency but cannot replace $D, C, N, A_0, S, H, T, P, L$ in a closed-face confluence read.

Thus the coverage claim is exactly the displayed closed-face boundary. It does not promote the packet to full substitution compatibility, arbitrary subject reduction, BetaParallelDiamondClosed, or unrestricted closed Church–Rosser. □

Theorem 735.6 (MetaCIC confluence audit residual-substitution determinacy)**.** *Let*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

be an accepted MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Any residual-substitution read of A is determined by the substitution-boundary row S together with the transport, replay, provenance, and local naming rows H, T, P, L and the closed-face hypotheses named by Definition 735.4. This determinacy is a boundary reading only: it cannot be promoted to* BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, subject reduction, or unrestricted closed Church–Rosser.*

Proof. Project the carrier of Definition 735.1. The residual-substitution coordinate is S, and the only structural rows available to transport or replay that coordinate are H, T, P, L.

Use the conditional and closed-substitution boundaries. Theorem 735.3 prevents the conditional diamond row from becoming an unconditional substitution theorem, while Definition 735.4 confines the closed face to its displayed closedness hypotheses.

The determinacy claim is therefore row-level. Once S and the closed-face hypotheses are fixed, H, T, P, L determine the residual-substitution read of the accepted carrier, but the remaining rows add no stronger coordinate.

Finally inspect the absent coordinates. The carrier has no proof row for BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, subject reduction, or unrestricted closed Church–Rosser, so no consumer can promote the residual-substitution boundary into those claims. □

Theorem 735.7 (MetaCIC confluence audit closed-face obligations)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Any closed-face confluence consumer that cites the closed diamond route must first expose the hypotheses* ClosedAt 0 *a and* ClosedAt 1 *b, the substitution-boundary row S, the componentwise transport row H, the replay row T, the provenance row P, and the local name row L. The citation is a read of the displayed closed-face boundary only; it is not a proof of full substitution compatibility, arbitrary subject reduction, or unrestricted Church–Rosser.*

Proof. Project the accepted carrier through Definition 735.1. The rows available to a closed-face consumer are displayed coordinates of the same finite packet, so the substitution boundary is read as S and any transported reading is routed through H, T, P, L.

Apply Definition 735.4. It confines the face to the hypotheses ClosedAt 0 a and ClosedAt 1 b before the closed diamond route may be cited.

The remaining rows do not add a stronger metatheorem. The carrier has no coordinate for full substitution compatibility, arbitrary subject reduction, or unrestricted Church–Rosser, so the consumer obligation is exactly the displayed closed-face exposure and not a promotion beyond it. □

Theorem 735.8 (MetaCIC confluence audit closed-face row separation)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Even when the closed-face hypotheses* ClosedAt 0 *a and* ClosedAt 1 *b are displayed, the substitution-boundary row S and subject-reduction separation row R remain boundary and handoff rows. They may route a closed-face confluence consumer, but they cannot be cited as* BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, subject reduction, or unrestricted Church–Rosser.*

Proof. Project the accepted carrier through Definition 735.1. The substitution data are named by S, the subject-reduction boundary is named by R, and the closed-face route is read through the displayed rows D, C, N, A_0 together with H, T, P, L.

Apply Definition 735.3 and Definition 735.4. These theorems keep the conditional confluence rows and closed substitution row tied to displayed hypotheses rather than promoting them to unrestricted substitution compatibility or a closed diamond theorem.

Use Definition 735.5 and Definition 735.6. Coverage exposes only the finite closed face, while determinacy keeps the residual-substitution read at the row level once the closed-face hypotheses are fixed.

Finally apply Definition 735.7. The closed-face consumer must pass through S, R, H, T, P, L as boundary data, and no carrier coordinate supplies the excluded global metatheorems. □

Theorem 735.9 (MetaCIC confluence audit obligation scope upgrade)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. The scoped obligation surface of A separates three readable boundaries: the closed-face confluence route through D, C, N, A_0, the residual-substitution boundary through S, H, T, P, L, and the subject-reduction boundary through R and chapter 719. The separation upgrades the packet only to scoped obligation inventory: it does not assert full substitution compatibility, unrestricted Church–Rosser, or a combined confluence-and-typing theorem.*

Proof. Project the accepted carrier through Definition 735.1. The rows D, C, N, A_0 name the closed confluence fragments, S names the residual-substitution boundary, R names the subject-reduction separation row, and H, T, P, L carry transport, replay, provenance, and local naming.

Use the ground recognition boundary of chapter 1048 and the finite reflection tuple boundary of chapter 993 to keep these rows finite and readable. They provide recognition and reflection routing, not a new metatheorem stronger than the displayed rows.

Finally compare with the subject-reduction discharge socket. The socket is a sibling dependency read by R, so the confluence packet can cite it as a boundary while keeping typing discharge outside the confluence carrier. The result is exactly a scoped obligation inventory and not a proof of the excluded global principles. □

Theorem 735.10 (MetaCIC confluence audit closed-face scope)**.** *For an accepted carrier $A = (D, C, N, A_0, S, R, H, T, P, L)$, every closed-face confluence read factors through the displayed rows D, C, N, A_0, H, T, P, L and the closedness hypotheses required by Definition 735.7. The residual substitution row S may be cited only as the boundary of the face, and the subject-reduction row R may be cited only as a sibling dependency route through chapter 719. No consumer of this scope obtains unrestricted substitution compatibility or closed-term Church–Rosser.*

Proof. Start with Definition 735.7. It requires the closedness hypotheses and the displayed substitution, transport, replay, provenance, and naming rows before any closed diamond route can be read.

Now restrict the confluence read to D, C, N, A_0. These rows provide the conditional diamond route, closed-diamond-to-star route, closed-normal fragment, and atom or impossible-source fragments already stored in the carrier. The finite recognition and reflection surfaces of chapter 1048 and chapter 993 keep those reads inside the displayed closed face.

It remains to account for the boundary rows. The row S records the residual substitution obstruction rather than solving it, and R points to the sibling subject-reduction discharge socket rather than importing it. Therefore the closed-face scope is the displayed finite boundary and not an unrestricted confluence theorem. □

Theorem 735.11 (MetaCIC confluence audit subject-reduction separation scope)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. The subject-reduction information available to a confluence consumer is exactly the row R together with its transport through H, T, P, L and its dependency on chapter 719. The residual-substitution row S and the closed-face rows D, C, N, A_0 remain separate obligation families, so the carrier does not merge typing preservation, substitution compatibility, and confluence into a single public theorem.*

Proof. Project R from the carrier definition. It is the only row whose stated role is to link the confluence boundary to subject-reduction packets, and its reads are transported only by H, T, P, L.

Apply the subject-reduction discharge socket as a sibling dependency. The socket supplies the typing-discharge boundary that R may cite, while the confluence carrier keeps D, C, N, A_0 and S as separate rows.

Finally inspect the remaining coordinates. The rows D, C, N, A_0 give closed-face confluence fragments, and S records residual substitution obligations; neither row contains a subject-reduction proof. Thus the scoped surface separates the three obligation families instead of exporting a combined subject-reduction and confluence theorem. □

Theorem 735.12 (MetaCIC confluence audit closed-face decider boundary)**.** *Any finite decider row for the closed substitution face of an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier may*

be consumed only through the rows S, H, T, P, L together with the existing closed-face hypotheses. It may not be promoted to full substitution compatibility, subject reduction, global β-conversion decidability, or arbitrary closed Church–Rosser.

Proof. Project the carrier of Definition 735.1. The row S is the substitution-boundary coordinate, while H, T, P, L are the displayed transport, route replay, provenance, and local naming rows through which any finite closed-face decider must be read.

Apply Definition 735.3 and Definition 735.4. The first keeps the generic substitution input conditional, and the second confines the closed substitution face to the hypotheses ClosedAt 0 a and ClosedAt 1 b.

The nonescape reading of the carrier then prevents promotion beyond those rows. There is no coordinate for full substitution compatibility, subject reduction, global β-conversion decidability, or arbitrary closed Church–Rosser. □

Theorem 735.13 (MetaCIC confluence audit redex boundary)**.** *Let*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

be an accepted MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Any redex-boundary consumer of A can read only the rows*

$$D, C, N, A_0, S, R, H, T, P, L$$

together with the closed-face hypotheses used by Definition 735.4 and Definition 735.12. That read cannot be promoted to unrestricted substitution compatibility, subject reduction, or closed Church–Rosser.

Proof. Project the carrier of Definition 735.1. A redex-boundary consumer sees exactly the displayed packet rows $D, C, N, A_0, S, R, H, T, P, L$.

Apply the conditional boundary. Theorem 735.3 keeps D, C as conditional bridge rows rather than unrestricted substitution compatibility or a closed parallel diamond theorem.

Apply the closed-substitution and closed-face boundaries. Theorem 735.4 supplies the closed-face hypotheses, and Definition 735.12 keeps finite closed-face decisions inside S, H, T, P, L.

The remaining rows do not add a stronger metatheorem. The normal-fragment, source-shape, subject-reduction separation, transport, replay, provenance, and local naming rows are finite audit data, so the redex-boundary read remains a boundary lemma and not a verification claim for substitution compatibility, subject reduction, or closed Church–Rosser. □

Theorem 735.14 (MetaCIC confluence audit closed-diamond candidate exhaustion)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier. Any closed-diamond candidate read of A is exhausted by the rows*

$$D, C, N, A_0, S, R, H, T, P, L$$

together with the displayed ClosedAt *hypotheses attached to the closed substitution face. The candidate read may consume the conditional diamond row, closed star handoff, normal-fragment rows, source-shape row, substitution-boundary row, subject-reduction separation row, transport, replay, provenance, and local naming, but it cannot be promoted to* BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, arbitrary subject reduction, or unrestricted closed Church–Rosser.*

Proof. Project the accepted carrier through Definition 735.1. A closed-diamond candidate can read only the displayed packet rows $D, C, N, A_0, S, R, H, T, P, L$.

Apply the conditional boundary of Definition 735.3. This keeps the diamond candidate tied to the conditional bridge rows D, C rather than turning it into an unconditional closed parallel diamond.

Next apply the closed substitution boundary of Definition 735.4 and the closed-face decider boundary of Definition 735.12. These force any closed-face use through S, H, T, P, L and the displayed ClosedAt hypotheses.

Finally use the subject-reduction separation and packet nonescape boundaries. The row R remains a separation row, while N, A_0 supply only the named normal and source-shape fragments. Since no carrier coordinate names full substitution compatibility, a closed parallel diamond theorem, arbitrary subject reduction, or unrestricted closed Church–Rosser, the listed rows exhaust the candidate read. □

Theorem 735.15 (MetaCIC confluence audit packet subject-reduction separation)**.** *The subject-reduction separation row R of an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier permits downstream chapters to cite chapter 719, chapter 723, and chapter 717 as neighboring MetaCIC audit packets. It does not identify the confluence substitution boundary with the four subject-reduction discharge obligations.*

Proof. Project the row R. Its job is referential: it names the sibling subject-reduction packets as separate finite surfaces available to downstream MetaCIC audits.

Now project the confluence rows D, C, N, A_0, S. These rows concern parallel reduction, closed star confluence, closed-normal and atom fragments, and the residual substitution boundary. They do not contain beta, application-argument, lambda-domain, or Pi-domain typing-discharge coordinates.

Thus the packet records a horizontal dependency rather than a fusion of interfaces. A downstream consumer may cite both families, but each obligation must be discharged on its own carrier surface. □

Theorem 735.16 (MetaCIC confluence audit packet non-escape)**.** *Every consumer read of an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier factors through the finite rows*

$$D, C, N, A_0, S, R, H, T, P, L.$$

The packet exports no hidden confluence theorem, no unrecorded substitution lemma, no host quotient over reductions, and no conversion from conditional bridges into an unconditional closed Church–Rosser result.

Proof. Start from the carrier definition. A consumer can project only the named rows $D, C, N, A_0, S, R, H, T, P, L$.

The positive rows expose conditional bridges and unconditional fragments. The transport and route rows explain how those entries are read componentwise, and the provenance and local naming rows identify the packet boundary.

The substitution boundary row blocks the only possible escape from the conditional bridge to a full endpoint. Because no further coordinate exists, there is no hidden confluence theorem, unrecorded substitution lemma, host quotient over reductions, or conversion into arbitrary closed Church–Rosser. □

Theorem 735.17 (MetaCIC confluence audit packet obligation-closure package)**.** *For an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier, the carrier row, conditional confluence boundary,*

closed substitution boundary, subject-reduction separation, and non-escape theorem form the whole obligation surface of the packet. The surface packages the conditional confluence rows and their obstruction boundary; it does not claim arbitrary closed Church–Rosser, unparameterized subject reduction, a proof of full substitution compatibility, or a hidden conversion from conditional bridges into unconditional endpoints.

Proof. Project the carrier of Definition 735.1 and the NameCert obligations of Definition 735.2. They expose the conditional bridge rows, unconditional fragments, substitution-boundary row, subject-reduction separation row, transport, replay, provenance, and local naming.

Use the conditional boundary and closed substitution boundary, Definition 735.3 and Definition 735.4. These keep the MetaCIC audit-map obstruction visible instead of turning it into full substitution compatibility or a closed parallel diamond.

Apply the subject-reduction separation and non-escape rows, Definition 735.15 and Definition 735.16. The first prevents fusion with the sibling subject-reduction packets, and the second exhausts every consumer read by $D, C, N, A_0, S, R, H, T, P, L$. Hence the packaged surface is obligation closure for the finite packet and not a stronger MetaCIC metatheorem. □

Theorem 735.18 (MetaCIC confluence audit candidate-mediated SN boundary)**.** *Any strong-normalization-facing consumer of an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier must pass through the displayed conditional-boundary rows D, C, the closed-substitution boundary row S, the subject-reduction separation row R, the nonescape route over $D, C, N, A_0, S, R, H, T, P, L$, and the obligation-closure package before it may read a candidate-mediated normalization claim. The read may use a candidate row only as finite audit data attached to that dependency chain; it does not prove global confluence, subject reduction, full substitution compatibility, or arbitrary strong normalization.*

Proof. First project the carrier rows. The conditional confluence surface is held by D, C, while S records the closed-substitution boundary and R records the subject-reduction separation.

Apply Definition 735.3, Definition 735.4, and Definition 735.15. These rows give the finite dependency chain a consumer must read before any candidate-mediated normalization datum can be interpreted.

Now apply Definition 735.16 and Definition 735.17. The first confines the consumer to $D, C, N, A_0, S, R, H, T, P,$ and the second packages those rows as the whole obligation surface of the packet.

Thus a candidate-mediated normalization claim is only a row-facing read downstream of the displayed boundaries. Since the carrier has no coordinate for global confluence, subject reduction, full substitution compatibility, or arbitrary strong normalization, none of those principles is exported. □

Theorem 735.19 (MetaCIC confluence audit obligation surface)**.** *For an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier $A = (D, C, N, A_0, S, R, H, T, P, L)$, the confluence audit obligations are exactly the conditional diamond row D, the closed-diamond-to-star row C, the closed-normal fragment row N, the atom and source-shape fragment row A_0, the substitution-boundary row S, the subject-reduction separation row R, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The TasteGate alignment named in the chapter's Lean marker reads these obligations as rows of the same finite packet rather than as a second MetaCIC confluence object.*

Proof. Project the carrier and apply Definition 735.2. The positive confluence reads are D, C, N, A_0, while the negative boundary and horizontal separation reads are S and R.

The kernel support rows are H, T, P, L. They give componentwise transport, consumer replay, provenance, and local naming for precisely those confluence and boundary entries.

Finally use the already displayed TasteGate carrier alignment: it reads the same carrier as the public audit packet. Thus the obligations are not transported into an untracked confluence theorem or a second carrier surface. □

Theorem 735.20 (MetaCIC confluence audit candidate-mediated strong normalization)**.** *Any candidate-mediated strong-normalization witness consumed by an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier must enter as a finite audit row attached to S and replayed through T. The packet may record that such a candidate row blocks the substitution obstruction for the displayed closed face, and it may transport the row through H with provenance P and local name L. It may not convert the candidate witness into an unconditional proof of* BetaParallelSubstitutionCompatibility*, arbitrary strong normalization, or closed Church–Rosser.*

Proof. Start with the substitution-boundary row S. It is the only coordinate that names the obstruction, so any candidate-mediated witness relevant to the obstruction must be recorded there as audit data.

Next replay the candidate row through T and transport it through H. The replay can feed the closed-face boundary described by Definition 735.4, but it remains a row of the same finite packet with provenance P and local name L.

The conditional-boundary theorem then prevents the final step from becoming unconditional. Since A has no coordinate for full substitution compatibility, arbitrary strong normalization, or closed Church–Rosser, the candidate witness is an audit row and not an exported MetaCIC theorem. □

Theorem 735.21 (MetaCIC confluence audit normal-fragment route)**.** *The normal-fragment confluence route of an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier is exactly the finite read of N, A_0, H, T, P, L. It covers the closed-normal, atom, sort, closed-variable-impossible, and non-lambda-application fragments named by the carrier, and it cites D, C, S, R only as the surrounding conditional and separation boundary. The route does not export full substitution compatibility, subject reduction, arbitrary closed-term Church–Rosser, or a host quotient over reductions.*

Proof. Project the unconditional fragment rows. The row N carries the closed-normal fragment, and A_0 carries the atom, sort, closed-variable-impossible, and non-lambda-application source-shape fragments.

Read those rows through H, T, P, L. Transport and replay keep the fragments inside the finite audit packet, while provenance and local naming record the same public carrier read used by the TasteGate alignment.

The remaining rows D, C, S, R are boundary data. They explain the conditional diamond route and the subject-reduction separation, but they do not add a coordinate for full substitution compatibility, subject reduction, arbitrary closed-term Church–Rosser, or a quotient over reductions. □

Theorem 735.22 (MetaCIC critical-pair normalization audit)**.** *For an accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier*

$$A = (D, C, N, A_0, S, R, H, T, P, L),$$

any critical-pair normalization consumer of the audit packet must route through the candidate-mediated strong-normalization row, the normal-fragment row N, the conditional parallel-diamond boundary D, C, the closed substitution boundary S, and the subject-reduction separation row R

before any confluence claim is read. The carrier may therefore audit a critical-pair normalization path only through the finite route

$$D, C, N, A_0, S, R \longmapsto H, T, P, L.$$

It exports no full substitution compatibility, no arbitrary strong-normalization theorem, no subject-reduction discharge, and no closed Church–Rosser theorem.

Proof. Project the carrier rows $D, C, N, A_0, S, R, H, T, P, L$. These are the only rows visible to a critical-pair normalization consumer.

First read the candidate-mediated strong-normalization boundary of Definition 735.20. It permits a candidate witness to be recorded as audit data attached to the obstruction surface, but it does not turn that witness into arbitrary strong normalization.

Next read the normal-fragment route of Definition 735.21. The row N and the source-shape row A_0 cover only the listed closed-normal and atom or source-shape fragments, with H, T, P, L preserving and replaying that finite surface.

The conditional and closed-substitution boundaries must still be crossed. Apply Definition 735.3 for the D, C bridge rows, and Definition 735.4 for the substitution row S.

Finally apply the subject-reduction separation of Definition 735.15. The row R keeps the confluence audit apart from the sibling subject-reduction discharge packets, so the consumer can read a normalization audit route but cannot extract full substitution compatibility, arbitrary strong normalization, subject reduction, or closed Church–Rosser. □

Theorem 735.23 (MetaCIC confluence audit critical-pair scope)**.** *For an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier*

$$A = (D, C, N, A_0, S, R, H, T, P, L),$$

any critical-pair consumer reads only through the displayed rows $D, C, N, A_0, S, R, H, T, P, L$. Candidate-mediated strong normalization may be attached as an audit row through the candidate-normalization handoff, and the normal-fragment route supplies the unconditional fragment. The scope exports no full confluence theorem, no subject reduction, no full substitution compatibility, and no arbitrary strong-normalization theorem.

Proof. Project the accepted carrier and the critical-pair normalization route of Definition 735.22. That theorem already forces a critical-pair normalization consumer through D, C, N, A_0, S, R before the structural rows H, T, P, L replay and name the read.

Attach candidate-mediated strong normalization only through the audit boundary of Definition 735.20 and the handoff of Definition 735.26. These rows record candidate evidence inside the packet; they do not export arbitrary strong normalization.

For the unconditional fragment, use Definition 735.21. Its rows N, A_0 cover only the closed-normal and source-shape fragments preserved by H, T, P, L.

Finally apply carrier nonescape from Definition 735.16. Since the carrier contains no row for full confluence, subject reduction, full substitution compatibility, or arbitrary strong normalization, no critical-pair consumer can read such a theorem from the packet. □

Theorem 735.24 (MetaCIC confluence audit residual critical-pair boundary)**.** *Every residual critical-pair read of an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier factors through the displayed rows $D, C, N, A_0, S, R, H, T, P, L$ and through the same* $\mathsf{BetaCriticalPair}$*-facing boundary named by the confluence audit packet. The read may classify a residual pair as belonging to the conditional diamond window, the normal-fragment window, or the substitution-obstruction window,*

but it does not prove closed Church–Rosser, subject reduction, full substitution compatibility, or a hidden substitution oracle.

Proof. Project the carrier. The rows D, C are the conditional diamond and closed-star routes, while N, A_0 are the normal and source-shape fragments available without the full substitution input.

Residual critical-pair data can only be attached to those rows through the boundary rows S, R and replayed by T. Componentwise transport through H keeps the same finite packet and prevents the residual classification from becoming a separate confluence theorem.

The provenance and local naming rows P, L close the packet boundary. Since the carrier has no coordinate for subject reduction, arbitrary closed Church–Rosser, full substitution compatibility, or an unrecorded substitution oracle, no residual critical-pair read can export one. □

Theorem 735.25 (MetaCIC confluence audit critical-pair coverage)**.** *Every accepted residual critical-pair consumer of a* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

is covered by exactly one of the displayed finite routes: the conditional diamond rows D, C, the normal and source-shape fragments N, A_0, the closed substitution-boundary row S, the subject-reduction separation row R, or the candidate-normalization handoff. In every case transport, replay, provenance, and local naming pass through H, T, P, L. The coverage exports no full substitution compatibility, arbitrary strong-normalization theorem, subject reduction, closed Church–Rosser theorem, or host quotient over reductions.

Proof. Project the residual consumer to the carrier rows. The carrier exposes only D, C, N, A_0, S, R as analytic confluence-audit rows and H, T, P, L as the structural rows by which those reads are transported and replayed.

If the consumer asks for the conditional diamond route, apply Definition 735.3. The rows D, C cover exactly the conditional parallel-diamond and closed-star handoff, while the substitution input remains named as a boundary.

If the consumer asks for a normal or source-shape fragment, apply Definition 735.21. The rows N, A_0 cover the closed-normal, atom, sort, closed-variable-impossible, and non-lambda-application fragments without adding the full substitution row as a proved theorem.

If the consumer asks for substitution or subject-reduction data, use Definition 735.4 and Definition 735.15. These rows cover S and R as boundary and separation data, not as full substitution compatibility or subject reduction.

It remains to cover candidate-mediated normalization reads. Apply Definition 735.26; the candidate row enters only through the displayed audit surface and is replayed by T with transport through H.

Finally apply Definition 735.24 and Definition 735.27. The first forces every residual critical-pair read into one of the displayed windows, and the second says the listed windows exhaust the obstruction ledger. Thus coverage is finite and exclusive at the packet boundary, with P, L naming the same source and with no export of the stronger MetaCIC principles excluded in the statement. □

Theorem 735.26 (MetaCIC confluence audit candidate-normalization handoff)**.** *Any candidate-normalization row consumed by an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier must enter through the substitution-boundary row S, the normal-fragment row N, or the source-shape fragment*

row A_0, and must be replayed through T with componentwise transport H. The handoff may route a candidate witness to a displayed normal-fragment or closed-face obstruction check, but it does not convert that witness into arbitrary strong normalization, subject reduction, full substitution compatibility, or global confluence.

Proof. Start with the candidate-mediated strong-normalization boundary already recorded in Definition 735.20. A candidate row relevant to the confluence audit is an audit datum attached to the obstruction surface, not a theorem outside the packet.

The normal-fragment route then restricts how the row is consumed. It may interact with N and A_0, and it may be replayed through T over the same packet, but the boundary rows D, C, S, R remain conditional and separated from subject-reduction discharge.

Finally apply carrier non-escape. The rows H, P, L transport, source, and name the same handoff, while the packet contains no coordinate for arbitrary strong normalization, subject reduction, full substitution compatibility, or global confluence. The candidate-normalization handoff therefore remains a finite audit route. □

Theorem 735.27 (MetaCIC confluence audit obstruction ledger exhaustion)**.** *The obstruction ledger of an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier is exhausted by the conditional bridge rows D, C, the normal-fragment rows N, A_0, the substitution-boundary row S, the subject-reduction separation row R, the residual critical-pair boundary, and the candidate-normalization handoff. A consumer that reads all those rows has read the whole finite obstruction surface; there remains no hidden row for closed Church–Rosser, subject reduction, full substitution compatibility, or an implicit substitution oracle.*

Proof. First collect the positive and boundary rows from the carrier. The rows D, C, N, A_0, S, R are the only analytic coordinates for confluence, normal fragments, substitution obstruction, and subject-reduction separation.

Next add the two displayed consumer routes. The residual critical-pair boundary and candidate-normalization handoff explain how downstream audits may consume those rows without creating a stronger MetaCIC theorem.

Finally use transport, replay, provenance, and local naming. The rows H, T, P, L preserve and name the same finite surface, and no further coordinate is present. Thus the ledger is exhausted exactly at the stated boundary and cannot conceal closed Church–Rosser, subject reduction, full substitution compatibility, or a substitution oracle. □

Theorem 735.28 (MetaCIC confluence audit substitution-obligation inversion)**.** *Any consumer that claims to use an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier for closed substitution compatibility must invert the carrier to the explicit substitution-obligation row S together with the transport, replay, provenance, and local naming rows H, T, P, L. Residual critical-pair reads, candidate-normalization reads, normal-fragment reads, and subject-reduction-separation reads do not supply S.*

Proof. Begin with Definition 735.3 and Definition 735.4. The former exposes only the conditional rows D, C, while the latter names S as the closed-substitution boundary row.

The residual, candidate, normal-fragment, and subject-reduction routes are then bounded by Definition 735.24, Definition 735.26, and Definition 735.27. They can consume the displayed audit surface but cannot manufacture the missing substitution row.

Thus a substitution-compatibility consumer must read S directly and replay it through H, T, P, L. Without that inversion, the packet exports only boundary data and not full closed substitution compatibility. □

Theorem 735.29 (MetaCIC confluence critical-pair obligations)**.** *Every accepted* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier exposes the critical-pair surface as obligation rows rather than as a closed confluence theorem. The rows include candidate-mediated overlap formation, bounded local diamond witness, and normal-form readback nonescape, each routed through the carrier rows of Definition 735.1, the NameCert obligations of Definition 735.2, the residual boundary of Definition 735.24, the candidate-normalization handoff of Definition 735.26, and the obstruction ledger of Definition 735.27. These rows preserve the TasteGate discipline of the chapter: without a displayed TasteGate marker for obligation promotion, they do not promote the packet beyond seed closure and they do not export closed Church–Rosser, subject reduction, full substitution compatibility, or global confluence.*

Proof. First form the candidate-mediated overlap row. The carrier rows expose conditional bridge data, normal-fragment data, substitution-boundary data, subject-reduction separation, transports, replay, provenance, and local naming, while the candidate-normalization handoff allows a candidate witness to be consumed only through that displayed audit surface.

Next form the bounded local diamond witness row. The conditional boundary and residual critical-pair boundary route any overlap request into the finite windows already named by the packet, so the witness remains bounded by the displayed local audit rows rather than becoming an arbitrary parallel-diamond theorem.

Finally read the normal-form nonescape row. The normal-fragment theorem and obstruction-ledger exhaustion account for the closed-normal, atom, source-shape, substitution-boundary, subject-reduction, residual, and candidate rows. Since the chapter is origin-ai and carries no displayed TasteGate marker for the promotion step, the theorem records obligation content only at seed closure and leaves stronger confluence principles outside the exported surface. □

Theorem 735.30 (MetaCIC confluence critical-pair frontier)**.** *Every accepted confluence-frontier read of a* MetaCICConfluenceAuditPacket$^{\uparrow}$ *carrier must pass through the candidate-mediated strong-normalization row, the closed-face decider boundary, the residual critical-pair boundary, the critical-pair coverage row, the obstruction-ledger exhaustion row, the substitution-obligation inversion row, and the local* NameCert *route before it can be exported as a MetaCIC confluence audit packet. The export is a finite audit boundary over*

$$D, C, N, A_0, S, R, H, T, P, L,$$

not a proof of global confluence, subject reduction, full substitution compatibility, arbitrary strong normalization, or closed Church–Rosser.

Proof. Project the accepted carrier. The local route is the displayed NameCert surface over the carrier rows $D, C, N, A_0, S, R, H, T, P, L$, so any frontier read begins inside the same finite packet.

Read the normalization and closed-face rows next. The candidate-mediated strong-normalization boundary of Definition 735.20 records candidate evidence only as audit data, while the closed-face decider boundary of Definition 735.12 confines closed substitution reads to S, H, T, P, L and the stated closed-face hypotheses.

Now route residual critical-pair data through Definition 735.24 and Definition 735.25. These theorems force every residual read into one of the displayed finite windows and keep transport, replay, provenance, and local naming in H, T, P, L.

Finally apply Definition 735.27 and Definition 735.28. The first says that the listed rows exhaust the obstruction ledger, and the second says that any substitution-compatibility consumer must invert to S rather than manufacture a hidden substitution theorem. Therefore the frontier exports only the finite audit packet and none of the stronger MetaCIC principles excluded in the statement. □

Theorem 735.31 (MetaCIC confluence audit critical-pair obligation closure)**.** *For an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier*

$$A = (D, C, N, A_0, S, R, H, T, P, L),$$

the closed-face decider boundary, subject-reduction separation row, candidate-normalization handoff, critical-pair scope, critical-pair coverage theorem, and obstruction ledger together form the finite obligation surface for the critical-pair confluence audit packet. The surface is the packet route

$$D, C, N, A_0, S, R, H, T, P, L,$$

with the closed-face hypotheses and residual windows named explicitly; it is not a proof of arbitrary Church–Rosser, unchecked substitution compatibility, unbounded subject reduction, or a hidden confluence theorem outside the displayed carrier.

Proof. Project the accepted carrier to the rows $D, C, N, A_0, S, R, H, T, P, L$. The local $\mathsf{NameCert}$ row L and provenance row P keep the audit inside the same finite packet, while H and T provide only componentwise transport and replay for those displayed rows.

First read the closed-face decider and subject-reduction boundaries. Apply Definition 735.12 for the closed substitution face and Definition 735.15 for the horizontal separation row R. These two facts expose the closed-face and subject-reduction obligations without turning either one into a global MetaCIC theorem.

Next read the normalization and critical-pair routing data. Apply Definition 735.26 and Definition 735.23. The handoff allows candidate evidence only as row-facing audit data, and the scope theorem keeps every critical-pair consumer inside $D, C, N, A_0, S, R, H, T, P, L$.

Finally apply Definition 735.25 and Definition 735.27. Coverage assigns each residual read to one displayed window, and the ledger says those windows exhaust the obstruction surface. Therefore the named rows close the critical-pair obligation package and still refuse arbitrary Church–Rosser, unchecked substitution compatibility, unbounded subject reduction, and hidden confluence outside the packet. □

Theorem 735.32 (MetaCIC confluence audit finite critical-pair frontier)**.** *Every accepted* MetaCICConfluenceAuditP *carrier routes each displayed finite critical pair through exactly the candidate-normalization handoff, the residual substitution boundary, or the obstruction ledger. These are the only public outcomes of the packet frontier; there is no fourth outcome exporting arbitrary Church–Rosser, unchecked substitution compatibility, unbounded subject reduction, or hidden confluence outside the displayed rows.*

Proof. Project the carrier rows $D, C, N, A_0, S, R, H, T, P, L$. Critical-pair normalization and scope first force the read through Definition 735.22 and Definition 735.23.

The residual boundary and coverage rows then separate candidate-normalization reads from residual substitution reads by Definition 735.24, Definition 735.25, and Definition 735.26.

Finally apply Definition 735.27 and Definition 735.28. The obstruction ledger exhausts the remaining displayed rows, and substitution inversion prevents a residual read from becoming a hidden compatibility theorem, so no fourth public outcome is available. □

Theorem 735.33 (MetaCIC confluence audit candidate-obstruction classifier)**.** *For an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier*

$$A = (D, C, N, A_0, S, R, H, T, P, L),$$

every failed confluence candidate row is classified by exactly one displayed obstruction family: substitution-boundary failure through S, residual critical-pair failure through D, C, A_0, closed-face coverage failure through N, S, H, T, P, L, or subject-reduction separation failure through R. The classification is read only from the theorem rows of this chapter, namely the closed-substitution boundary, residual critical-pair boundary, closed-face decider boundary, subject-reduction separation, critical-pair coverage, obstruction-ledger exhaustion, substitution-obligation inversion, and critical-pair obligation closure.

Proof. Project the carrier to $D, C, N, A_0, S, R, H, T, P, L$. A failed candidate row first exposes which displayed audit face it attempts to read.

If the read asks for substitution compatibility, apply the closed-substitution boundary and substitution-obligation inversion rows; the obstruction is the S boundary. If it asks for residual overlap data, apply the residual critical-pair boundary, critical-pair scope, and critical-pair coverage rows; the obstruction is the residual family over D, C, A_0.

If the read asks for closed-face coverage, apply the closed-face decider boundary together with obstruction-ledger exhaustion; the obstruction is the closed-face family over N, S, H, T, P, L. If it asks for subject reduction, apply the subject-reduction separation row; the obstruction is the R family.

The obstruction ledger lists no fifth face, and the four displayed faces use disjoint carrier rows for their deciding evidence. Hence the finite case split is exhaustive and assigns exactly one obstruction family to each failed confluence candidate row. □

Theorem 735.34 (MetaCIC confluence audit candidate-obstruction determinacy)**.** *Let two failed confluence candidate rows for a* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier read the same displayed carrier rows*

$$D, C, N, A_0, S, R, H, T, P, L$$

and the same audit face. Then their obstruction classifications agree. The common family is substitution-boundary failure through S, residual critical-pair failure through D, C, A_0, closed-face coverage failure through N, S, H, T, P, L, or subject-reduction separation failure through R, according to the shared face named by the classifier.

Proof. Apply Definition 735.33 to both failed candidate rows. Since the carrier rows and the audit face are identical, both classifier reads enter the same displayed case split.

If the shared face asks for substitution compatibility, both rows are classified by the S boundary. If it asks for residual overlap data, both rows read the residual family over D, C, A_0.

If the shared face asks for closed-face coverage, both rows are classified by the family over N, S, H, T, P, L. If it asks for subject reduction, both rows are classified by the R separation family.

The classifier theorem lists no fifth family and the deciding rows for the four families are fixed by the common carrier read. Hence the two failed rows receive the same obstruction classification. This proves determinacy only for the displayed candidate obstruction surface, not global confluence, subject reduction, full substitution compatibility, or Church–Rosser. □

Theorem 735.35 (MetaCIC confluence audit scoped kernel route)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. Before any confluence consumer can read A as a packet, the kernel-visible read exposes the diamond row D, the closed-star row C, the normal-form row N, the atom-fragment row A_0, the componentwise* hsame *transport row H, the displayed* Cont *replay row T, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}$ *row L.*

Proof. Project the carrier of Definition 735.1. The only confluence-facing rows available to the consumer are the displayed audit rows D, C, N, A_0 together with the transport, replay, provenance, and naming rows H, T, P, L.

The substitution-boundary row S and subject-reduction separation row R remain boundary data. They can route later checks, but they are not read as global strong normalization, unrestricted confluence, full substitution compatibility, or arbitrary subject reduction.

Thus the scoped kernel route is exhausted by carrier-row projection. Any confluence consumer that has not read D, C, N, A_0, H, T, P, L has not yet read the finite audit packet itself. □

Theorem 735.36 (MetaCIC confluence audit critical-pair obligations)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. Every candidate strong-normalization or critical-pair consumer of A must first expose the finite rows D, C, A_0, S, H, T, P, L as* BHist *entries transported by* hsame *and replayed through* Cont*. The consumer may classify a residual critical-pair face only through those rows and the local* $\mathsf{NameCert}$ *row L; it may not cite an ambient confluence proof or a hidden subject-reduction discharge.*

Proof. Project the accepted carrier through Definition 735.1. The critical-pair surface is named by the conditional confluence rows D, C, the atom and source-shape fragments A_0, and the substitution-boundary row S.

Transport and replay are constrained by H, T, P, L. These rows keep the read inside displayed BHist data and prevent a consumer from replacing the critical-pair face by a host reduction theorem.

The subject-reduction row R is a handoff pointer, not a confluence row. Hence candidate strong-normalization and critical-pair consumers receive the listed finite audit rows only, with no hidden proof of confluence or subject reduction. □

Theorem 735.37 (MetaCIC confluence audit residual stability)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. Residual substitution readings of A are stable under componentwise* hsame *transport exactly when the substitution-boundary row S, the conditional bridge rows D, C, the closed-face row N, and the replay rows H, T, P, L are transported together. This stability is a NameCert-row claim over finite* BHist *data, not a proof of* $\mathsf{BetaParallelSubstitutionCompatibility}$*.*

Proof. Read the residual-substitution boundary as the row S. A transported read must also carry D, C for the conditional bridge, N for the closed face, and H, T, P, L for componentwise comparison, replay, provenance, and local naming.

If these rows are transported together, the residual read names the same finite audit face after replacement. If any one of them is absent, the replay route cannot determine which substitution boundary the consumer is reading.

The result is therefore row stability. Since the carrier still has no coordinate proving $\mathsf{BetaParallelSubstitutionCom}$ stable residual transport does not become the missing substitution theorem. □

Theorem 735.38 (MetaCIC confluence audit decidability handoff)**.** *Let $A = (D, C, N, A_0, S, R, H, T, P, L)$ be an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. Any confluence decidability consumer receives a finite handoff consisting of the candidate obstruction family over D, C, A_0, S, R, the closed-face rows N, S, H, T, P, L, and the local* $\mathsf{NameCert}$ *row. The handoff may decide only which displayed audit face is available or obstructed; it does not decide unrestricted confluence, subject reduction, closed Church–Rosser, or full substitution compatibility.*

Proof. Start with the finite carrier inventory. The candidate obstruction family is read from D, C, A_0, S, R, while the closed-face consumer also reads N, S, H, T, P, L.

The classifier can therefore branch only over displayed audit faces: conditional bridge availability, residual-substitution boundary, closed-face coverage, and subject-reduction separation. Each branch is a finite BHist row with a local NameCert read.

No branch contains the excluded metatheorems. The handoff gives a decidable finite audit classification for the packet, not unrestricted confluence, subject reduction, closed Church–Rosser, or full substitution compatibility. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ packet is generated from conditional confluence bridges, unconditional closed-normal and atom fragments, a substitution-compatibility boundary row, a subject-reduction separation row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 735.1, Definition 735.2, Definition 735.3, Definition 735.4, Definition 735.15, Definition 735.9, Definition 735.10, Definition 735.11, Definition 735.45, and the checked single-carrier alignment target bind the confluence audit-map boundary as a finite mature NameCert export surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.MetaCICConfluenceAuditPacketUp.MetaCICConfluenceAuditPacketTasteGate\_single\_carrier\_`
Lean4 ✓ `BEDC.Derived.MetaCICConfluenceAuditPacketUp.taste gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove substitution compatibility, a closed parallel diamond, arbitrary closed-term Church–Rosser, subject reduction, checked Lean coverage beyond carrier alignment, or a public standard bridge.

Upgrade path: Bridge checking requires the standard-bridge theorem for the same finite packet and its checked single-carrier alignment target.

Lemma 735.39 (Candidate-normalization sibling route)**.** *Every accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier that is read by a candidate-normalization consumer may cite chapter 5547 only as a boundary route. The audit packet contributes its conditional confluence rows, residual-substitution boundary, closed-face hypotheses, transport, replay, provenance, and local NameCert; the candidate-normalization handoff contributes the candidate gate, normal endpoint row, residual envelope, decidable boundary, and blocked-edge retention.*

Proof. Project the audit packet through Definition 735.1. Its confluence-facing data are still the rows $D, C, N, A_0, S, R, H, T, P, L$, with S retaining the substitution boundary and R acting only as a separation pointer.

Read the sibling handoff as the neighboring candidate-normalization route of chapter 5547. The composition identifies where the candidate-normalization consumer may attach to the audit map, but it does not add a new coordinate to the audit carrier.

Therefore the route preserves the existing no-claim boundary: it is not a proof of full Church–Rosser, subject reduction, unrestricted closed consistency, full substitution compatibility, or a premise-free parallel diamond. □

Theorem 735.40 (MetaCIC confluence candidate residual diamond handoff)**.** *Every candidate-normalization read consumed by an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier must expose the residual-row packet* S, R, H, T, P, L*, the finite critical-pair window* D, C, A_0*, the substitution route through* S*, and the local join budget named by the MetaCIC critical-path packet of chapter 765 before any diamond conclusion is exported. The handoff may route a candidate witness into the residual diamond boundary, but it does not export full confluence, subject reduction, full substitution compatibility, unrestricted Church–Rosser, or kernel consistency.*

Proof. Project the accepted audit carrier of Definition 735.1. The residual-facing rows are the substitution boundary S, the subject-reduction separation row R, transport H, replay T, provenance P, and the local name L.

Read the candidate-normalization sibling route of Definition 735.39 and the candidate-normalization handoff of Definition 735.26. These keep candidate evidence inside the displayed audit surface and replay it through T with componentwise transport through H.

Now attach the residual and finite critical-pair windows. Theorems Definition 735.24, Definition 735.25, and Definition 735.32 force each residual read into the displayed $D, C, A_0, S, R, H, T, P, L$ windows before any local diamond read can be interpreted.

Finally use Definition 735.27. The obstruction ledger exhausts the substitution, residual, candidate, and subject-reduction boundaries, while chapter 765 supplies the critical-path budget consumed by the handoff. Thus the candidate read reaches only the residual diamond boundary and not the stronger MetaCIC principles excluded in the statement. □

Theorem 735.41 (MetaCIC confluence audit mature crosscheck)**.** *Every mature read of an accepted* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier that consumes the candidate-normalization handoff factors through the audit anchor, candidate evidence, residual confluence envelope, decidable boundary, blocked-edge row, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows of the displayed packet. The handoff is therefore a bounded residual route; it does not export full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with the candidate-normalization sibling route of Definition 735.39. It allows the neighboring handoff to attach only as a boundary route, not as a new carrier coordinate.

Read the candidate evidence through Definition 735.26 and the residual envelope through Definition 735.40. These rows keep the candidate witness inside the displayed audit surface and route residual diamond use through the finite critical-pair and substitution windows.

Next expose the decidable boundary and blocked-edge row. The closed-face decider boundary, residual critical-pair boundary, and obstruction ledger force the read through $D, C, N, A_0, S, R, H, T, P, L$ before any confluence-facing conclusion can be consumed.

Finally project the audit carrier and its NameCert obligations. The rows H, T, P, L supply componentwise transport, displayed replay, provenance, and local naming for the same packet, while the carrier has no coordinate for full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency. The candidate-normalization handoff is therefore consumed only as a bounded residual route. □

Theorem 735.42 (MetaCIC confluence audit candidate-SN discharge boundary)**.** *Let*

$$A = (D, C, N, A_0, S, R, H, T, P, L)$$

be an accepted $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *carrier. A candidate strong-normalization consumer may read the audit packet only through the displayed rows* $D, C, N, A_0, S, R, H, T, P, L$ *together with the candidate-normalization sibling route of Definition 735.39. The handoff may discharge candidate-mediated finite normalization data into the audit boundary, but it does not discharge full confluence, subject reduction, closed Church–Rosser, full substitution compatibility, or kernel consistency.*

Proof. Project the accepted carrier. The displayed audit packet has exactly the rows $D, C, N, A_0, S, R, H, T, P, L$, so a candidate strong-normalization consumer starts with the same finite row surface as every other confluence-audit read.

Next attach the candidate-normalization sibling route. By Definition 735.39 and Definition 735.26, candidate evidence enters only through the displayed audit surface and is replayed by T with componentwise transport through H.

Now read the critical-pair and residual boundaries. Theorems Definition 735.25, Definition 735.24, and Definition 735.27 force each critical-pair or residual read into the conditional diamond rows, normal and source-shape fragments, substitution boundary, subject-reduction separation row, or candidate-normalization handoff.

Finally use the decidability and substitution handoff. The closed-face decider boundary and substitution-obligation inversion keep closed substitution compatibility at the explicit S, H, T, P, L face, while the subject-reduction separation row R remains a pointer to sibling discharge packets. Thus the candidate-SN consumer receives a finite audit discharge boundary and none of the full confluence, subject-reduction, closed Church–Rosser, full substitution-compatibility, or kernel-consistency endpoints. □

The $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ route consumes the critical-path classifier surface of chapter 765 and the decidable-boundary rows of chapter 749 before exporting confluence-audit rows to downstream MetaCIC consumers.

Theorem 735.43 (MetaCIC confluence audit bridge boundary)**.** *The public* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *audit packet exports a bridged boundary only through the candidate-normalization sibling route, the residual-diamond handoff, the candidate SN discharge boundary, and the sibling lattice route displayed by the hub. No bridge-facing consumer may bypass those sibling rows or read a host confluence certificate outside the finite* BHist *route named by the packet.*

Proof. Read the public audit packet through its displayed siblings. The candidate-normalization route fixes the normal-form boundary, the residual-diamond handoff records the local confluence comparison, the candidate SN discharge boundary records the terminating candidate side, and the sibling lattice route records the shared dependency order.

A bridged consumer therefore receives only the route that has passed through these four rows. Since the hub exposes no additional host certificate coordinate, the bridge boundary is exactly the displayed finite route through those siblings. □

The bridge marker records the checked single-carrier alignment as the finite confluence-audit bridge surface for $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, not a full Church–Rosser theorem.

Theorem 735.44 (MetaCIC confluence audit bridged route export)**.** *The public* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ *audit packet exports a bridged boundary only through the candidate-normalization sibling route, the residual-diamond handoff, the candidate strong-normalization discharge boundary, the sibling lattice route, and the bridge boundary route. This export packages the checked single-carrier alignment as a finite confluence-audit boundary; it does not assert full substitution compatibility, unrestricted Church–Rosser, or a global confluence theorem outside the displayed audit route.*

Proof. Read the public audit packet through its existing NameCert carrier and public-closure surface. The candidate-normalization sibling route supplies the candidate shapes admitted by the packet, while the residual-diamond handoff records the local overlap boundary available to the audit.

Next use the candidate strong-normalization discharge boundary together with the sibling lattice route. These rows keep the normalization and residual evidence inside the displayed packet and prevent the audit from becoming an unrestricted substitution-compatibility theorem.

Finally apply the bridge boundary route. The checked single-carrier alignment is exported only as the bridged boundary named by those sibling rows, so the conclusion is a finite audit export and not a free Church–Rosser principle. □

Theorem 735.45 (MetaCIC confluence audit mature closure record)**.** *For an accepted* MetaCICConfluenceAuditPac *carrier, the carrier row, candidate-normalization route, residual-diamond handoff, candidate strong-normalization discharge boundary, sibling lattice route, and bridge boundary route exhaust the public mature export. The record refuses full Church–Rosser, unrestricted substitution compatibility, subject reduction, global confluence, and any host evaluator equality outside those displayed rows.*

Proof. Read the carrier row first through the NameCert obligations and the checked single-carrier alignment. This exposes only the finite audit packet and its displayed rows.

Next apply the bridged route export theorem. The candidate-normalization route, residual-diamond handoff, candidate strong-normalization discharge boundary, sibling lattice route, and bridge boundary route are the public export surface named by the hub.

No remaining row exports full Church–Rosser, unrestricted substitution compatibility, subject reduction, global confluence, or host evaluator equality. The mature record is therefore exactly the displayed public audit boundary. □

Chapter 736

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistency}^{\uparrow}$

The $\mathsf{ClosedNormalConsistency}^{\uparrow}$ packet gives the closed-normal branch of chapter 9375 a finite BEDC landing. It is deliberately downstream of chapter 731: the packet records the normal-fragment consistency object that is already assembled from closed typing, closedness, and beta-normality, while keeping the subject-reduction obstruction as a separate boundary row.

Definition 736.1 (Closed normal consistency carrier)**.** *A* $\mathsf{ClosedNormalConsistency}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, T, C, S, O, H, R, P, N)$$

with the following displayed rows:

1. *F is the closed proof-of-false target row, naming the false encoding and the empty-context typing boundary;*
2. *T is the closed-term row, recording that the inspected term is closed at the empty context;*
3. *C is the closed-normal consistency statement row, naming* BEDC.MetaCIC.ClosedNormalConsistencyAssemblyStat
4. *S is the beta-strong normality row consumed by the closed-normal consistency statement;*
5. *O is the subject-reduction obstruction handoff row, read from chapter 731;*
6. *H records componentwise* hsame *transports for the target, closed-term, statement, normality, obstruction, package, and local naming rows;*
7. *R records the displayed* Cont *routes by which consistency-audit consumers read the finite packet;*
8. *P is the* Pkg *provenance over MetaCIC typing, closed terms, beta-strong normality, the subject-reduction obstruction boundary,* BHist, hsame, Cont, *and local* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistency}^{\uparrow}}$ *row.*

The carrier contains no coordinate for full closed consistency, no unblocked closed subject-reduction theorem, no proof of the four subject-reduction discharge obligations, and no ambient normalization principle.

Theorem 736.2 (Closed normal consistency subject-reduction obstruction handoff)**.** *Every consumer of an accepted* $\mathsf{ClosedNormalConsistency}^{\uparrow}$ *packet reads the normal-fragment consistency row through*

$$F, T, C, S, H, R, P, N$$

and reads the subject-reduction boundary only through the separate obstruction row O supplied by chapter 731. The handoff permits citation of the closed-normal consistency statement for already-normal closed terms, but it does not convert the obstruction row into full closed subject reduction, closed reduction consistency, or a discharge of the beta, application-argument, lambda-domain, and Pi-domain obligations.

Proof. Project the carrier of Definition 736.1. The normal-fragment side consists of the false target row F, the closed-term row T, the statement row C, and the beta-strong row S, transported and routed only through H, R, P, N.

The obstruction side is the single row O. By Definition 731.2, that row exposes the parameterized subject-reduction theorem, the four discharge obligations, and the checked obstruction readback as a finite packet rather than as a closed subject-reduction theorem.

The consumer boundary is therefore separated into two reads. The closed-normal consistency row may eliminate a closed normal proof-of-false candidate, while the subject-reduction obstruction row records why a closed reduction route to such a normal candidate is not part of this packet.

Since no carrier coordinate supplies the four discharge obligations, an unparameterized closed subject-reduction proof, or closed reduction consistency, the handoff cannot be repacked into a stronger MetaCIC consistency claim. □

Chapter 737

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$

This chapter turns the closed-normal and atom part of chapter 9378 into a finite certificate packet. It sits next to chapter 731: the present packet records the checked confluence fragment already available for closed normal and atom-shaped sources, while the sibling obstruction packet keeps substitution-sensitive and subject-reduction boundaries visible.

Definition 737.1 (MetaCIC confluence normal-fragment carrier)**.** *A* $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, B, S, A, G, H, C, P, N)$$

with the following rows:

1. *T is the closed source term row at the empty context;*
2. *B is the beta-normality row consumed by the closed-normal confluence fragment;*
3. *S is the checked* BetaStarStep *closed-normal confluence row;*
4. *A is the atom-shape join row for sort and impossible closed-variable sources;*
5. *G is the conditional confluence-boundary row naming the remaining closed parallel-diamond and substitution-compatibility interfaces;*
6. *H records componentwise* hsame *transports for the source, normality, star-confluence, atom, boundary, package, and local naming rows;*
7. *C records the displayed* Cont *routes by which confluence consumers read those rows;*
8. *P is the* Pkg *provenance over MetaCIC beta reduction, closedness, normality, atom joins,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}}$ *row.*

The carrier contains no coordinate for arbitrary closed-term Church–Rosser, no proof of BetaParallelDiamondClose *and no proof of* BetaParallelSubstitutionCompatibility*.*

Theorem 737.2 (MetaCIC confluence normal-fragment NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenc}}$ *surface for Definition 737.1 consists exactly of carrier admission, closed-source row readback, beta-normality readback, checked closed-normal star-confluence readback, atom-shape join readback, conditional-boundary exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and non-escape from arbitrary closed-term Church–Rosser.*

Proof. Carrier admission is projection from the displayed tuple J. The confluence rows are S for the closed-normal star fragment and A for the atom-shaped source fragment, with T and B supplying the closed source and beta-normality inputs.

The row G is part of the obligation surface because it names what remains conditional: a closed parallel diamond and the substitution compatibility input that would supply such a diamond. It is read as a boundary row, not as a proof of the endpoint.

The structural rows H, C, P, N carry exactly this finite packet under transport, route replay, provenance, and local naming. Since the carrier has no coordinate for an arbitrary closed-source Church–Rosser theorem, the NameCert surface cannot export one. □

Lean4 ✓ `BEDC.MetaCIC.betaParallel diamond closed normal`

Theorem 737.3 (MetaCIC confluence closed-normal scope)**.** *Every accepted* MetaCICConfluenceNormalFragment$^{\uparrow}$ *packet exposes closed multi-step confluence only for sources whose displayed closed row T is paired with the beta-normality row B. A consumer may read the checked row S through H, C, P, N, but it cannot replace B by an arbitrary closed reduction source or infer full closed-source Church–Rosser for* BetaStarStep*.*

Proof. Read the carrier rows T, B, S. The theorem row S is attached to the closed-normal input surface, so its source is not merely a closed term; it is a closed term together with the beta-normality evidence recorded by B.

Transport and route replay through H, C, P, N preserve the same dependency shape. They do not add a reduction-to-normal row, a subject-reduction row, or a closed parallel-diamond row.

Thus the consumer obtains the closed-normal confluence fragment and no stronger closed-source confluence theorem. The missing arbitrary-source step remains the boundary row G. □

Lean4 ✓ `BEDC.MetaCIC.betaStarStep confluence closed normal`

Theorem 737.4 (MetaCIC confluence atom-join scope)**.** *The atom component A of an accepted* MetaCICConfluenceNormalFragment$^{\uparrow}$ *packet exposes only the sort and impossible closed-variable joins already present in the confluence substrate. It may be consumed together with T, S, H, C, P, N as an atom-shaped supplement to the closed-normal fragment, but it does not provide a beta-redex join for arbitrary applications, lambdas, or Pi terms.*

Proof. Project the atom row A. Its role is to record the atom-shaped joins listed in chapter 9378, where sort sources join directly and closed variable sources are blocked by the closedness premise.

The row is consumed beside the normal-fragment row S, not above it. This keeps atom geometry and closed-normal confluence inside the same finite packet while leaving beta-redex and componentwise application branches outside the atom row.

Since no carrier coordinate names a substitution-compatible parallel diamond, the atom row cannot be repacked as arbitrary closed confluence. It remains a shape-limited join surface. □

Theorem 737.5 (MetaCIC confluence ledger boundary)**.** *Every confluence read exported by an accepted* MetaCICConfluenceNormalFragment$^{\uparrow}$ *packet factors through*

$$T, B, S, A, G, H, C, P, N.$$

The packet exposes closed-normal star confluence, atom-shaped joins, and the conditional-boundary ledger, but it has no row for full parallel diamond, substitution compatibility, arbitrary closed-term Church–Rosser, host normalization, or a hidden subject-reduction theorem.

Proof. Start from Definition 737.1. The only consumer-visible confluence rows are S and A, and the only row naming a larger confluence endpoint is the conditional boundary row G.

Apply Definition 737.2. It accounts for carrier admission, fragment readback, boundary exposure, and the structural rows that route the packet.

No listed coordinate contains the full parallel diamond, the substitution compatibility input, a host normalization principle, or subject reduction. The ledger is therefore exhausted by the displayed rows. □

Theorem 737.6 (MetaCIC confluence normal-fragment public certificate)**.** *The public certificate exported by* $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ *is the finite packet* $J = (T, B, S, A, G, H, C, P, N)$ *together with the four surfaces of Definition 737.2, Definition 737.3, Definition 737.4, and Definition 737.5. A downstream consumer may cite the checked closed-normal and atom fragments through this packet while the full arbitrary closed-source confluence claim remains outside the certificate.*

Proof. The NameCert obligations identify the carrier and its local structural rows. The closed-normal theorem fixes the positive BetaStarStep scope, and the atom theorem fixes the available shape-limited joins.

The ledger theorem then exhausts the exported rows. It lists the conditional boundary row G as a boundary, so the public certificate does not erase the remaining closed parallel-diamond and substitution-compatibility inputs.

Together these rows form a finite BEDC certificate: positive where the Lean fragment is checked, and explicitly bounded where arbitrary closed confluence is not part of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ packet is generated from a closed source row, beta-normality row, checked closed-normal confluence row, atom-join row, conditional-boundary row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 737.1, Definition 737.2, Definition 737.3, Definition 737.4, Definition 737.5, and Definition 737.6 bind the closed-normal and atom confluence fragment as a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.MetaCIC.betaStarStep\_confluence\_closed\_normal`

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary closed-term Church–Rosser, BetaParallelDiamondClosed, BetaParallelSubstitutionCompatibility, host normalization, or a public standard bridge.

Upgrade path: The next formal step is a checked certificate wrapper over the same carrier and a bridge once the remaining closed confluence boundary is discharged.

Chapter 738

A Concrete Naming Certificate for ClosedConsistencyReductionBoundary$^{\uparrow}$

The ClosedConsistencyReductionBoundary$^{\uparrow}$ packet turns the frontier of chapter 9375 into a finite NameCert object. It is downstream of chapter 483: the packet names the rows that would be consumed by a closed-consistency reduction path, while keeping the blocked subject-reduction discharge as a public obstruction row rather than as an available theorem. It also supplies the reduction-obstruction row consumed by the closed-normal consistency route of chapter 764, so that the closed-normal endpoint is never read as full closed consistency without the displayed discharge boundary.

Definition 738.1 (Closed-consistency reduction-boundary carrier)**.** *A* ClosedConsistencyReductionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, E, D, K, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the closed source proof row, naming the empty-context proof candidate and its displayed false type;*
2. *Q is the reduction-route row, recording the finite beta route from the source proof row toward a normal endpoint;*
3. *E is the normal endpoint row, naming the reached term together with its closedness and beta-strong normality data;*
4. *D is the subject-reduction discharge row, exposing the beta, application-argument, lambda-domain, and Pi-domain obligations required before the false type may be transported along Q;*
5. *K is the closed-normal consistency row consumed after the endpoint row has retained the displayed false type;*
6. *O is the obstruction ledger row, recording that the present packet does not contain the missing closed subject-reduction discharge;*
7. *H records componentwise* hsame *transports for the source, route, endpoint, discharge, closed-normal consistency, obstruction, package, and local naming rows;*
8. *C records the displayed* Cont *routes by which consistency consumers read S, Q, E, D, K, O inside the finite packet;*

9. *P is the* Pkg *provenance over* BHist*, MetaCIC closed typing, beta-reduction routes, beta-strong normal endpoints,* hsame, Cont*, subject reduction, closed-normal consistency, obstruction rows, and local* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}}$ *row.*

The carrier contains no full closed consistency theorem, no unparameterized closed subject-reduction theorem, and no route that transports the displayed false type from S to E without the discharge row D.

Lean4 (definition) `BEDC.Derived.ClosedConsistencyReductionBoundaryUp.ClosedConsistencyReductionBoundaryUp`

Theorem 738.2 (Closed-consistency reduction-boundary NameCert obligations)**.** *For the carrier of Definition 738.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *source exposure: every accepted packet displays the closed source proof row S together with the false type it claims to inhabit;*

2. *route exposure: the beta route Q and normal endpoint E are public rows, not ambient normalization data;*

3. *discharge exposure: the subject-reduction discharge row D is the only row that may preserve the displayed false type along Q;*

4. *normal-consistency handoff: the closed-normal consistency row K is read only after the endpoint row E has retained the displayed false type;*

5. *obstruction accounting: the ledger row O records the missing closed subject-reduction discharge and prevents K from being repacked as full closed consistency.*

Proof. Project the displayed packet R. The rows $S, Q, E, D, K, O, H, C, P, N$ are finite coordinates, so source exposure, route exposure, discharge exposure, normal handoff, and obstruction accounting are all read from the same carrier.

The subject-reduction sibling chapter 483 supplies the discipline for reading preservation as a named row rather than as an ambient conversion principle. Here that row is D, and the packet gives no second coordinate that can preserve the false type along Q.

The closed-normal consistency row K is therefore downstream of E only under the displayed discharge boundary. Since O records the missing closed discharge, the local certificate exposes the reduction boundary without closing full closed consistency. □

Theorem 738.3 (Closed-consistency reduction-boundary obstruction readback)**.** *Every downstream read of an accepted* $\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}$ *packet factors through*

$$S, Q, E, D, K, O, H, C, P, N.$$

A consumer may inspect the source proof row, reduction route, normal endpoint, subject-reduction discharge row, closed-normal consistency row, obstruction ledger, transports, continuations, provenance, and local naming certificate. It cannot read full closed consistency, a closed subject-reduction theorem, an undisplayed false-type transport, or an endpoint consistency result detached from the discharge row.

Proof. Use the classifier clause of Definition 738.1. Public comparison and downstream reading are restricted to the displayed rows and their hsame, Cont, Pkg, and NameCert provenance.

The normal endpoint can be consumed by the closed-normal consistency row only through the route and discharge rows. Thus the readback exposes the precise place where arbitrary closed proof reduction would need subject reduction, and it does not supply that missing theorem.

The non-readable objects are exactly absent from the carrier. Full closed consistency, a closed subject-reduction theorem, hidden false-type transport, and detached endpoint consistency have no coordinate in R and no continuation route in C. □

Lean4 ✓ `BEDC.Derived.ClosedConsistencyReductionBoundaryUp.ClosedConsistencyReductionBoundaryCarrier obstruction readback`

Theorem 738.4 (Closed-consistency reduction-boundary vision handoff)**.** *The* ClosedConsistencyReductionBoundar *packet realizes the frontier of chapter 9375 as finite BEDC data. It connects the closed source proof row, beta route, normal endpoint, subject-reduction discharge boundary, closed-normal consistency row, and obstruction ledger without claiming that arbitrary closed proofs can already be transported to beta-strong normal endpoints of the same displayed false type.*

Proof. Read the frontier in section 9375.3. Full closed consistency is blocked at the step where an arbitrary closed proof must retain its displayed false type along reduction to a beta-strong normal endpoint.

The carrier of Definition 738.1 names exactly that interface. The source row S, route row Q, endpoint row E, discharge row D, closed-normal consistency row K, and obstruction row O are separate finite coordinates with displayed transport, continuation, package, and naming data.

Therefore the packet is a handoff from the vision frontier to the concrete-instance layer. It records the boundary that a later formal target may consume, while the obstruction ledger keeps the manuscript from citing the packet as a proof of full closed consistency. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}$ packet is generated from finite BHist rows for closed-consistency premises, reduction-route rows, obstruction and boundary rows, componentwise transports, displayed continuation replay, provenance, and local naming. The route exposes where a closed reduction read may use existing finite rows and where it must stop before being treated as a full consistency proof. Cont keeps each consumer read on the displayed route, hsame transports the same boundary inventory, and Pkg records the closed-normal, reduction, and audit dependencies. The packet therefore names a reduction boundary rather than a completed metatheorem.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the closed-consistency reduction-boundary carrier, route audit, obstruction preservation, and boundary statement around closed reduction consistency.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim full closed consistency, closed subject reduction, unrestricted reduction consistency, host conversion semantics, or quotient equality of terms.

Upgrade path: Obligation closure requires Lean encodings of the boundary carrier, route audit, obstruction preservation, and non-escape theorem.

Chapter 739

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$

The $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$ surface makes the assembly map of chapter 9375 into a finite naming certificate. It sits beside chapter 562: typed normal-form packets expose a finite endpoint normal-form read, while the present packet records how closedness, typing, beta-strong normality, the contradiction row, and the subject-reduction obstruction are kept as separate rows of one MetaCIC consistency boundary.

Definition 739.1 (Closed-normal consistency packet carrier)**.** *A* $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, T, B, F, K, G, H, C, P, N)$$

with the following displayed rows:

1. *L is the closedness row for the inspected MetaCIC term at the empty context;*
2. *T is the empty-context typing row for the same term;*
3. *B is the beta-strong normality row consumed by the closed-normal theorem;*
4. *F is the internal false-type row named by the typing certificate;*
5. *K is the checked closed-normal consistency row naming* BEDC.MetaCIC.closed_normal_consistency_assembly*;*
6. *G is the subject-reduction obstruction row recording that arbitrary closed proofs are not transported to the normal fragment inside this packet;*
7. *H records componentwise* hsame *transports for the closedness, typing, beta-strong, false-type, theorem, obstruction, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes by which consumers read those rows;*
9. *P is the* Pkg *provenance over MetaCIC closedness, empty-context typing, beta-strong normality, internal false, subject-reduction obstruction,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for full closed consistency, hidden subject reduction, host conversion, quotient behaviour, or a normalization oracle.

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyPacketUp.ClosedNormalConsistencyPacketTasteGate`
`single carrier alignment`

Theorem 739.2 (Closed-normal consistency packet NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistenc}}$ *surface for Definition 739.1 consists of carrier admission, classifier exactness on* $L, T, B, F, K, G, H, C, P, N$*, closedness readback, empty-context typing readback, beta-strong normality readback, contradiction-row readback, checked closed-normal theorem readback, subject-reduction obstruction exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and non-escape from full closed consistency or hidden normalization transport.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyPacketUp.ClosedNormalConsistencyPacketNameCertObligations`

Proof. Carrier admission is the finite row list of Definition 739.1. Classifier exactness compares only the displayed coordinates $L, T, B, F, K, G, H, C, P, N$, so no unlisted semantic equality can enter the packet.

The positive MetaCIC data are the closedness row L, the typing row T, the beta-strong row B, the false row F, and the checked theorem row K. These are exactly the rows needed to read the closed-normal contradiction fragment of chapter 9375.

The obstruction row G is a separate obligation. It records that the carrier has not supplied subject reduction from arbitrary closed proofs into the beta-strong normal fragment.

The structural rows H, C, P, N preserve and name that same finite surface. Since no coordinate names full closed consistency, hidden subject reduction, host conversion, quotient behaviour, or a normalization oracle, the local NameCert cannot export any of those claims. □

Theorem 739.3 (Closed-normal consistency normal-fragment boundary)**.** *Every accepted* ClosedNormalConsistency *carrier reads the positive closed-normal consistency fragment only through* $L, T, B, F, K, H, C, P, N$*. A consumer may use these rows to exclude a closed beta-strong proof candidate at the displayed internal false row, but the consumer obtains no route from an arbitrary closed proof candidate to that beta-strong boundary.*

Proof. Project the carrier of Definition 739.1. The rows L, T, B, F, K provide closedness, empty-context typing, beta-strong normality, the contradiction target, and the checked closed-normal theorem.

The rows H, C, P, N only transport, route, package, and locally name that surface. They do not introduce a reduction sequence or a theorem preserving typing along such a sequence.

Thus the packet supports the positive normal-fragment read and stops there. The missing route from arbitrary closed proofs is not present in the carrier. □

Theorem 739.4 (Closed-normal consistency subject-reduction gap boundary)**.** *The subject-reduction gap of a* $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$ *carrier is exactly the displayed obstruction row* G*. Agreement on* $L, T, B, F, K, H, C, P, N$ *cannot be repacked as closed subject reduction, as closed reduction consistency, or as a transport from arbitrary closed proof terms to beta-strong normal inhabitants of the same false type.*

Proof. The classifier for the packet fixes the positive rows L, T, B, F, K and the structural rows H, C, P, N. These rows authenticate the closed-normal contradiction fragment.

The row G is not derived from those rows; it is displayed as the boundary separating the positive fragment from the subject-reduction problem. Reading G gives the consumer a named obstruction, not a discharge of the obstruction.

Therefore the packet cannot be strengthened into closed subject reduction or closed reduction consistency by reusing the positive fragment. Any such strengthening would require a row absent from the carrier. □

Theorem 739.5 (Closed-normal consistency obstruction readback)**.** *Every closed-consistency consumer of an accepted* $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$ *packet that uses the positive rows* L, T, B, F, K *must also expose the obstruction row* G *before asking for arbitrary closed proof transport, closed subject reduction, or closed reduction consistency. The rows* H, C, P, N *transport, replay, source, and name the same finite surface; no absent row discharges the subject-reduction obstruction.*

Proof. Project the carrier $Q = (L, T, B, F, K, G, H, C, P, N)$ of Definition 739.1. The positive rows L, T, B, F, K read only the beta-strong normal fragment named by the closed-normal theorem row.

Apply the normal-fragment boundary, Definition 739.3. It allows the consumer to use the closedness, typing, beta-strong, false-type, and checked-theorem rows only inside that finite normal fragment.

Apply the subject-reduction gap boundary, Definition 739.4. It identifies G as the exact displayed obstruction between the positive fragment and arbitrary closed proof transport.

The structural rows do not remove the obstruction. Rows H, C, P, N preserve and name the same finite packet, so any request for closed subject reduction or closed reduction consistency still has to expose G and cannot be discharged by an absent row. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$ packet is generated from finite BHist rows for closed-normal positive fragments, reduction-consistency evidence, a displayed obstruction boundary, transports, continuations, provenance, and local naming. The positive rows are read by Cont as finite fragment evidence, while the obstruction row stays visible as the boundary separating that fragment from closed subject reduction. Componentwise hsame preserves both positive and obstruction rows, and Pkg records the closed-normal and reduction dependencies. The packet names exactly the finite consistency surface it carries and refuses any strengthening that needs absent rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the closed-normal consistency packet carrier, positive-fragment readback, obstruction row, and non-strengthening boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim closed subject reduction, full closed reduction consistency, host conversion semantics, quotient term equality, or hidden discharge of the obstruction.

Upgrade path: Obligation closure requires Lean encodings of the packet carrier, positive-fragment projection, obstruction preservation, and non-strengthening theorem.

Chapter 740

A Concrete Naming Certificate for NormalFormConsistencySeal$^{\uparrow}$

The NormalFormConsistencySeal$^{\uparrow}$ packet gives the closed-normal branch of chapter 9375 a finite NameCert object. It sits between the closed-context carrier of chapter 722 and the subject-reduction socket of chapter 719: the seal records what the normal-form consistency theorem consumes, and it keeps the missing subject-reduction discharge outside the packet.

Definition 740.1 (Normal-form consistency seal carrier)**.** *A* NormalFormConsistencySeal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, F, N, K, X, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context MetaCIC typing row for the inspected closed proof candidate;*
2. *F is the internal false-type row named by that typing certificate;*
3. *N is the beta-strong normality row for the same inspected candidate;*
4. *K is the checked closed-normal consistency row naming* BEDC.MetaCIC.closed_normal_consistency_assembly*;*
5. *X is the subject-reduction boundary row, read only through chapter 719;*
6. *H records componentwise* hsame *transports for the typing, false, normality, theorem, boundary, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which a consumer reads those finite rows;*
8. *P is the* Pkg *provenance over MetaCIC typing, closed contexts, beta-strong normality,* BHist*,* hsame*,* Cont*, and sibling* NameCert *rows;*
9. *L is the local* NameCert$_{\text{NormalFormConsistencySeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary closed reduction, no closed subject-reduction theorem, no hidden normalization transport, and no host-level consistency principle.

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealTasteGate single carrier alignment`

Theorem 740.2 (Normal-form consistency seal obligations)**.** *Every accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet binds a local NameCert surface whose obligations are exactly carrier admission, classifier exactness for $S = (T, F, N, K, X, H, C, P, L)$, readback of the checked closed-normal theorem row K,*

componentwise hsame *transport, displayed* Cont *route replay, package provenance, and non-escape from the subject-reduction boundary row X.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealObligations`

Proof. Carrier admission is the displayed finite packet of Definition 740.1. Classifier exactness is componentwise: two seal packets compare only through the rows $T, F, N, K, X, H, C, P, L$ and the displayed hsame transports.

The theorem row K is read together with T, F, N, so the consumer obtains only the normal-form contradiction already assembled in BEDC.MetaCIC.closed_normal_consistency_assembly. The structural rows H, C, P, L preserve that same read under transport, route replay, provenance, and local naming.

The boundary row X is named but not discharged. Therefore the NameCert surface records the subject-reduction frontier without turning it into a closed subject-reduction theorem. □

Theorem 740.3 (Normal-form consistency seal subject-reduction boundary)**.** *The* NormalFormConsistencySeal$^{\uparrow}$ *packet factors every preservation-facing read through chapter 719 and every closed-context read through chapter 722. Hence the seal may eliminate a closed beta-strong proof candidate of the internal false row, but it does not provide a route from an arbitrary closed proof candidate to such a normal endpoint.*

Proof. For closed-context data, project the rows T, F, N together with the provenance row P. These rows are interpreted through the closed-term context seal, which keeps the empty-context typing and closedness information as finite BEDC data.

For preservation-facing data, project the boundary row X. Its only named consumer is the subject-reduction discharge socket, so the seal can cite the frontier while leaving the beta, application-argument, lambda-domain, and Pi-domain discharge rows outside the present carrier.

Combining these projections gives the normal-form use of K without a reduction-to-normal-form bridge for arbitrary closed proof candidates. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealSubjectReductionBoundary`

Theorem 740.4 (Normal-form consistency seal closed-normal frontier)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. The packet may export the closed-normal contradiction route exactly through the rows T, F, N, K, H, C, P, L: the typing row T, internal false row F, beta-strong normality row N, checked closed-normal theorem row K, componentwise transports, displayed continuation replay, provenance, and local naming. The row X remains only the visible subject-reduction boundary citation and cannot discharge arbitrary subject reduction or turn an arbitrary closed proof candidate into the closed-normal endpoint consumed by K.*

Proof. Project the carrier of Definition 740.1. The positive closed-normal route is carried by T, F, N, K and by the structural rows H, C, P, L that transport, replay, source, and name the same finite packet.

Apply Definition 740.2. Its readback clause exposes the checked theorem row K only together with the displayed typing, false, and beta-strong normality rows, while its non-escape clause keeps the subject-reduction boundary row separate.

Finally use Definition 740.3. That boundary permits the seal to cite X as the open preservation-facing socket, but it does not supply a closed subject-reduction theorem. Hence consumers may use the closed-normal contradiction route through T, F, N, K, H, C, P, L and no stronger subject-reduction route. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealClosedNormalFrontier`

Theorem 740.5 (Normal-form consistency seal critical-pair route)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any critical-pair read, residual-discharge read, or preservation-facing consumer of S must pass through the displayed finite route*

$$T, F, N, K, X, H, C, P, L.$$

The route exposes the typing row, internal false row, beta-strong normality row, checked closed-normal theorem row, subject-reduction boundary row, transport, continuation replay, provenance, and local naming, but it does not discharge subject reduction internally.

Proof. Project the carrier in Definition 740.1. Its only visible rows are $T, F, N, K, X, H, C, P, L$, so a critical-pair or residual-discharge consumer has no coordinate outside that displayed packet.

Apply Definition 740.2. The local NameCert obligations read the checked closed-normal theorem row K with the typing, false, and beta-strong normality rows, while the row X is recorded as the non-escaped subject-reduction boundary.

Use Definition 740.3 and Definition 740.4. Together they identify the closed-normal contradiction route and keep arbitrary subject-reduction discharge outside the seal.

Thus every preservation-facing read must expose $T, F, N, K, X, H, C, P, L$ before it can cite the open frontier, and the present packet cannot turn that citation into an internal subject-reduction proof. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCriticalPairRoute`

Theorem 740.6 (Normal-form consistency seal subject-reduction socket readiness)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Every preservation-facing consumer must read the socket row X as a visible subject-reduction boundary row and cannot treat that socket as already discharged by the closed-normal theorem row K.*

Proof. Project the carrier of Definition 740.1. The row X is separate from K, so the checked closed-normal theorem row and the subject-reduction socket occupy distinct coordinates.

Apply Definition 740.2. Its non-escape clause records the subject-reduction frontier without turning it into a closed subject-reduction theorem.

Use Definition 740.3. The boundary theorem routes preservation-facing data through the sibling subject-reduction discharge socket, and it leaves the beta, application, lambda, and Pi discharge rows outside the present seal.

Finally apply Definition 740.5. Every critical-pair or preservation-facing read must expose X with the finite packet route, so readiness of the socket is visible but not discharged. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealSubjectReductionSocketReadine`

Theorem 740.7 (Normal-form consistency seal critical-pair frontier exhaustion)**.** *For an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet $S = (T, F, N, K, X, H, C, P, L)$, every critical-pair read and residual-discharge read is exhausted by the displayed route*

$$T, F, N, K, X, H, C, P, L.$$

No additional evaluator equality, quotient syntax equality, hidden reduction schedule, or subject-reduction discharge row is exported by the seal.

Proof. Use Definition 740.5. It already identifies the displayed route required by critical-pair, residual-discharge, and preservation-facing consumers.

Apply Definition 740.4. The closed-normal contradiction route is confined to T, F, N, K, H, C, P, L, while X remains the visible subject-reduction boundary citation.

Apply Definition 740.6. Since X is only a socket row, a residual consumer cannot use it as a hidden discharge theorem.

The exhaustion claim follows from the carrier shape and the obligation theorem: the local NameCert surface contains no coordinate beyond $T, F, N, K, X, H, C, P, L$ from which an extra evaluator, quotient, reduction-schedule, or preservation row could be read. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCriticalPairFrontierExhaustic`

Theorem 740.8 (Normal-form consistency seal MetaCIC consumer boundary)**.** *Every closed-normal MetaCIC consistency consumer of an accepted* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *packet may use the checked row K only together with the displayed closed-context typing row T, internal false row F, beta-strong normality row N, and structural rows H, C, P, L. The consumer may also cite X as an open subject-reduction socket, but not as a discharged preservation theorem.*

Proof. Project T, F, N, K from the carrier. These rows are the closed-context typing, false-type, normality, and checked closed-normal theorem data consumed by the MetaCIC consistency read.

Apply Definition 740.4. It states that the closed-normal contradiction route is exactly T, F, N, K, H, C, P, L and that X remains outside the closed-normal endpoint.

Use Definition 740.6 and Definition 740.7. Together they keep the subject-reduction socket visible and exhaust every critical-pair or residual read by the displayed finite route.

Thus a MetaCIC consumer may cite K only with the closed-context and normality rows that the packet displays, and any preservation-facing use remains a socket handoff rather than a completed theorem. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealMetacicConsumerBoundary`

Theorem 740.9 (Normal-form consistency seal constructor frontier)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *packet. The constructor frontier exported by S is exactly the closed-normal contradiction route through T, F, N, K, H, C, P, L. The subject-reduction row X remains a boundary citation: no beta, application, lambda, or Pi constructor discharge is internal to the seal.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealConstructorFrontier`

Proof. Read the carrier of Definition 740.1. The rows T, F, N, K provide the typed closed-normal false candidate and the checked closed-normal theorem row, while H, C, P, L transport, replay, source, and name that same finite route.

Apply the obligation theorem Definition 740.2. Its readback clause keeps K tied to the displayed typing, false-type, and beta-strong normality rows, and its non-escape clause prevents the boundary row X from being read as a constructor discharge.

Finally use the subject-reduction boundary and closed-normal frontier theorems, Definition 740.3 and Definition 740.4. They leave beta, application, lambda, and Pi preservation obligations at the subject-reduction socket, so the seal exports only the closed-normal contradiction route and no internal discharge for those constructors. □

Theorem 740.10 (Normal-form consistency seal critical-pair obligation)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any constructor-facing read of S that mentions beta, application, lambda, or Pi critical pairs must display the critical-pair boundary through $T, F, N, K, X, H, C, P, L$ before it may use the checked closed-normal theorem row K. The read cannot treat K as an arbitrary confluence theorem or as a discharge of the subject-reduction boundary row X.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCriticalPairObligation`

Proof. Project the carrier of Definition 740.1. The row K is available only together with the typing row T, false row F, beta-strong normality row N, subject-reduction boundary row X, and the structural rows H, C, P, L.

Use Definition 740.9. It states that beta, application, lambda, and Pi constructor discharges remain outside the seal. The constructor-facing read therefore sees those critical-pair cases only as boundary data.

Finally apply Definition 740.2, Definition 740.3, and Definition 740.4. These rows permit use of K only as the displayed closed-normal contradiction route, so no arbitrary confluence or subject-reduction discharge is exported. □

Theorem 740.11 (Normal-form consistency seal residual-context obligation)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. The empty-context row T and the subject-reduction boundary row X keep every residual context transport visible: a consumer must read residual context information through the displayed rows T, X, H, C, P, L and cannot replace that route by host substitution or an ambient weakening theorem.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealResidualContextObligation`

Proof. Project the seal carrier. The row T is the empty-context MetaCIC typing row, and X is the visible subject-reduction boundary; neither row is a host substitution operation.

Apply the subject-reduction boundary theorem Definition 740.3. It forces preservation-facing data through the sibling discharge socket and keeps the closed-context read tied to the displayed finite rows.

The obligation theorem and constructor frontier, Definition 740.2 and Definition 740.9, then route all transport through H, C, P, L. Hence residual contexts remain visible BEDC data rather than an implicit host substitution principle. □

Theorem 740.12 (Normal-form consistency seal joinability obligation)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any joinability read must factor through all displayed rows $T, F, N, K, X, H, C, P, L$. It may use the closed-normal route named by K only after the typing, false, normality, boundary, transport, replay, provenance, and local naming rows are visible, and it may not assert arbitrary confluence or arbitrary joinability for MetaCIC terms.*

Proof. Read the carrier in Definition 740.1. The finite packet exposes exactly the rows $T, F, N, K, X, H, C, P, L$, so a joinability-facing consumer has no other coordinate to inspect.

Use Definition 740.4. The positive route is the closed-normal contradiction route through T, F, N, K, H, C, P, L, while X remains the visible subject-reduction boundary.

Now apply Definition 740.9 and Definition 740.10. The constructor and critical-pair cases are boundary obligations, not internal confluence data. Therefore any joinability read factors through the displayed seal rows and cannot export arbitrary confluence. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealJoinabilityObligation`

Theorem 740.13 (Normal-form consistency seal finite-window determinacy)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ and $S' = (T', F', N', K', X', H', C', P', L')$ be accepted* NormalFormConsistencySeal$^{\uparrow}$ *packets. Suppose the two packets agree componentwise on the displayed typing row, false-type row, beta-strong normality row, checked closed-normal consistency row, subject-reduction boundary row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. Then every closed-normal consumer read exposed by S and S' has the same BEDC classifier boundary.*

Proof. Project both carriers through Definition 740.1. The displayed agreement identifies the paired rows T, T', F, F', N, N', K, K', X, X', H, H', C, C', P, P', and L, L' componentwise.

The closed-normal read is routed through the same K row together with the same typing, false-type, and beta-strong normality rows. The boundary row X is only the visible subject-reduction frontier, so the comparison does not assert arbitrary subject reduction or move an arbitrary closed proof candidate to a normal endpoint.

The remaining agreement on H, C, P, L transports, replays, sources, and names the same finite packet. Therefore the two exposed consumer reads meet the same BEDC classifier boundary. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealFiniteWindowDeterminacy`

Theorem 740.14 (Normal-form consistency seal non-escape)**.** *No consumer of an accepted* NormalFormConsistency *packet can read a claim stronger than the displayed normal-form consistency surface. In particular, the packet exports neither full closed consistency, nor closed subject reduction, nor hidden normalization transport, nor a host equality principle beyond the displayed* hsame *rows.*

Proof. Inspect the carrier coordinates in Definition 740.1. The only theorem coordinate is K, and K is explicitly the closed-normal consistency row. The row X records the subject-reduction boundary rather than a discharge of that boundary.

The remaining coordinates H, C, P, L are structural rows for transport, routing, provenance, and local naming. They preserve the finite packet but do not add new MetaCIC reduction, typing, or normalization facts.

Thus every public read stays inside the visible normal-form seal. Anything requiring arbitrary closed reduction or full consistency must pass through a separate subject-reduction discharge packet. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealNonescape`

Theorem 740.15 (Normal-form consistency seal vision handoff)**.** *The concrete* NormalFormConsistencySeal$^{\uparrow}$ *packet realizes the assembled node of chapter 9375 as a finite BEDC object: the closed-normal theorem row is public inside the seal, while the lift to full closed consistency remains blocked until the subject-reduction socket is filled by its own discharge data.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealVisionHandoff`

Proof. The vision chapter separates three items: the available closed-normal consistency theorem, the missing subject-reduction route, and the blocked full closed-consistency conclusion. The carrier of Definition 740.1 mirrors that separation by placing K and X in different rows.

The obligation theorem and boundary theorem show how consumers may use K with the closed-context rows while routing every preservation-facing request to the sibling socket. The non-escape theorem then prevents the packet from being read as a stronger consistency claim.

Therefore the handoff is a concrete finite packet rather than a promise of a larger MetaCIC theorem. □

Theorem 740.16 (Normal-form consistency seal continuation replay)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Every continuation replay read first projects the MetaCIC typing row T, the false-type row F, the beta-strong normality row N, the closed-normal consistency row K, and the subject-reduction boundary row X, and only then reads through the displayed* Cont *row C. No continuation consumer may bypass these rows to obtain a closed subject-reduction discharge.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealContinuationReplay`

Proof. Read the carrier in Definition 740.1. The packet lists T, F, N, K, X before the structural rows, so any replay route has a displayed typing, false-type, normality, closed-normal consistency, and boundary source.

Apply the obligation surface of Definition 740.2. It makes the continuation row part of the finite NameCert obligations for the same carrier.

The subject-reduction boundary theorem Definition 740.3 and the non-escape theorem Definition 740.14 keep the read from becoming a closed subject-reduction theorem. Thus continuation replay is a route through C after the displayed rows T, F, N, K, X have been projected. □

Theorem 740.17 (Normal-form consistency seal provenance exactness)**.** *For an accepted packet $S = (T, F, N, K, X, H, C, P, L)$, every dependency claim made by the* NormalFormConsistencySeal$^{\uparrow}$ *surface is confined to the provenance row P over MetaCIC typing, closed contexts, beta-strong normality,* BHist*,* hsame*,* Cont*, and sibling* NameCert *rows. The packet contains no additional provenance route for full closed consistency or closed subject reduction.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealProvenanceExactness`

Proof. Carrier admission in Definition 740.1 displays P as the only package provenance coordinate. The remaining structural rows H, C, L transport and name the packet but do not create dependency claims.

Use Definition 740.2 to identify the obligation rows governed by that provenance: MetaCIC typing, closed-context data, beta-strong normality, history transport, continuation routing, and the local naming surface.

The boundary result Definition 740.3 and Definition 740.14 exclude a hidden dependency on full closed consistency or a closed subject-reduction theorem. Therefore every dependency claim remains exactly in the displayed row P. □

Theorem 740.18 (Normal-form consistency seal local naming boundary)**.** *For an accepted packet $S = (T, F, N, K, X, H, C, P, L)$, the local naming row L names only this finite normal-form consistency seal. It identifies the displayed carrier, obligations, subject-reduction boundary, continuation replay, and provenance rows, but it does not discharge closed subject reduction.*

Proof. The carrier of Definition 740.1 makes L the local naming coordinate for the finite seal. It names the same packet rows T, F, N, K, X, H, C, P and no row outside the packet.

The obligation theorem Definition 740.2 supplies the local NameCert surface, while Definition 740.3 marks X as a boundary row rather than a discharge row.

The non-escape theorem Definition 740.14 prevents the local name from being read as full closed consistency, closed subject reduction, hidden normalization transport, or host equality. Hence L names the finite seal and nothing stronger. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealLocalNamingBoundary`

Theorem 740.19 (Normal-form consistency seal obligation closure)**.** *For every accepted* NormalFormConsistencySea *packet $S = (T, F, N, K, X, H, C, P, L)$, the closed-normal theorem row K is consumed only after the typing row T, false-type row F, beta-strong normality row N, subject-reduction boundary row X, componentwise transport row H, continuation replay row C, provenance row P, and local naming row L have been projected. Thus the packet closes the MetaCIC obligation surface for the displayed closed-normal theorem row while keeping arbitrary subject reduction outside the carrier.*

Proof. Project the carrier of Definition 740.1. The rows T, F, N, K, X are the only MetaCIC rows available before the structural coordinates are read.

Apply Definition 740.2. Its readback clause forces the checked closed-normal theorem row K to be consumed together with the displayed typing, false-type, and normality rows, while its boundary clause keeps X visible as a subject-reduction frontier.

Finally use continuation replay, provenance exactness, and local naming from Definition 740.16, Definition 740.17, and Definition 740.18. These rows preserve the same packet and add no closed subject-reduction discharge. Hence the closed-normal theorem row is consumed only over the stated obligation surface. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealObligationClosure`

Theorem 740.20 (Normal-form consistency seal candidate route)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any candidate read that attempts to use* BEDC.MetaCIC.closed_ *must first pass through the closed-context typing row T, the internal false-type row F, the beta-strong normality row N, the named theorem row K, the subject-reduction boundary row X, transport H, replay C, provenance P, and local naming L. The route contains no coordinate for arbitrary reduction to a normal endpoint, full closed consistency, or a hidden subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCandidateRoute`

Proof. Begin with a candidate read from the finite packet. Carrier projection exposes T, F, N before the theorem row can be interpreted, so the candidate is already read as a closed, typed, beta-strong normal proof candidate for the internal false row.

The row K is the named closed-normal theorem row. By Definition 740.4, it is available only through the closed-normal route and not through a route from arbitrary closed candidates to normal endpoints.

The boundary row X and structural rows H, C, P, L then fix the public reading of the packet. The subject-reduction boundary theorem and non-escape theorem exclude any hidden discharge, so the candidate route is exactly the displayed finite route through $T, F, N, K, X, H, C, P, L$. □

Theorem 740.21 (Normal-form consistency seal candidate-mediated SN handoff)**.** *Let $S = (T, F, N, K, X, H, C, P$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any read of S that exports candidate-mediated strong-normalization evidence must first expose the closed typed-term row T, candidate row, normalization-frontier row, consistency-seal row K,* hsame *transport H,* Cont *replay C, provenance P, and local NameCert row L. The handoff cannot turn candidate-mediated SN into arbitrary closed subject reduction, full closed consistency, hidden normalization transport, or host equality.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCandidateMediatedSNHandoff`

Proof. Project the seal carrier. The closed typed-term information is read through T, and the consistency-seal conclusion is read only through K together with the finite rows carried by the packet.

Use the candidate route Definition 740.20. It forces any candidate-facing read of the closed-normal theorem row to pass through $T, F, N, K, X, H, C, P, L$ rather than through an arbitrary normalization endpoint.

Use the MetaCIC discharge frontier Definition 740.23 together with the normalization-frontier chapter chapter 892. These references keep candidate-mediated SN attached to the displayed frontier and leave subject reduction at its separate discharge socket.

Finally apply the non-escape theorem Definition 740.14. The structural rows H, C, P, L preserve and name the same finite handoff, so no consumer can read a stronger consistency or normalization claim from the seal. □

Theorem 740.22 (Normal-form consistency seal reduction determinacy)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ and $S' = (T', F', N', K', X', H', C', P', L')$ be accepted* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *packets. If the two packets agree componentwise on typing, false-type, beta-strong normality, the closed-normal theorem row, subject-reduction boundary, transport, replay, provenance, and local naming, then every closed-normal theorem read from S and S' has the same BEDC classifier boundary. This determinacy is confined to the closed-normal route and does not decide arbitrary reduction, full closed consistency, or closed subject reduction.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealReductionDeterminacy`

Proof. Project both carriers. The displayed componentwise agreement identifies the pairs T, T', F, F', N, N', K, K', X, X', H, H', C, C', P, P', and L, L' as the same finite classifier boundary data.

Apply Definition 740.13. It says that agreement on exactly these rows determines the closed-normal consumer read and keeps X as a boundary row rather than a subject-reduction discharge.

The non-escape and local naming boundary theorems then rule out any additional coordinate that could distinguish the two reads by arbitrary reduction or full closed consistency. Therefore the closed-normal theorem reads have the same BEDC classifier boundary and no stronger determinacy is claimed. □

Theorem 740.23 (Normal-form consistency seal MetaCIC discharge frontier)**.** *Any* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *consumer that attempts to promote closed-normal consistency into closed consistency must first pass through the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *route of chapter 765 and the subject-reduction discharge socket of chapter 719. The seal itself exposes only the rows $T, F, N, K, X, H, C, P, L$ and supplies no closed subject-reduction discharge.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealMetacicDischargeFrontier`

Proof. Project the carrier of Definition 740.1. Its visible rows are the typing row T, false-type row F, beta-strong normality row N, checked closed-normal theorem row K, subject-reduction boundary row X, transport H, replay C, provenance P, and local naming row L.

Apply Definition 740.3 and Definition 740.4. These results identify X as a boundary citation and restrict K to the closed-normal route, so the packet can expose a closed-normal contradiction only after the displayed normality and typing rows have been read.

Apply Definition 740.22. It fixes the classifier boundary of the closed-normal read but does not provide arbitrary reduction to a normal endpoint. Thus determinacy does not turn the seal into a closed-consistency proof.

The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet records the same open route through the subject-reduction discharge socket. A consumer seeking closed consistency must therefore leave the seal and discharge that socket; the present packet records the frontier rather than filling it. □

Theorem 740.24 (Normal-form consistency seal residual schedule discharge)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet whose candidate route, residual-context obligation, finite-window determinacy, and MetaCIC discharge frontier rows are displayed. Then S admits a single residual schedule for candidate-mediated strong-normalization and confluence checking. Every residual read in that schedule is replayed through the same continuation and provenance rows before any strong-normalization or confluence consumer may inspect the closed-normal theorem row.*

Proof. Project the accepted packet. The visible schedule has only the typing row T, false-type row F, beta-strong normality row N, checked closed-normal theorem row K, subject-reduction boundary row X, transport row H, replay row C, provenance row P, and local naming row L.

Apply Definition 740.20 and Definition 740.11. These rows put candidate-mediated normality and residual-context reads before any closed-normal theorem consumer.

Next use Definition 740.12 and Definition 740.13. They force every joinability or determinacy read to reuse the same finite window rather than opening a second residual route.

Finally apply Definition 740.23. The MetaCIC frontier keeps closed consistency behind the subject-reduction socket, while C and P replay and source the same residual schedule for any strong-normalization or confluence consumer exposed by K. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealResidualScheduleDischarge`

Theorem 740.25 (Normal-form consistency seal candidate boundary)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Any normal endpoint read that consumes the seal must first expose the Candidate or ClosedCandidate evidence named by the candidate route, the closed-normal frontier rows T, F, N, K, X, the residual schedule rows used for candidate-mediated strong-normalization and confluence checking, componentwise* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local naming L. The boundary contains no coordinate for arbitrary reduction to a normal endpoint, global closed consistency, hidden subject reduction, evaluator equality, or quotient syntax equality.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealCandidateBoundary`

Proof. Start with the candidate route Definition 740.20. It forces every candidate-facing read of the closed-normal theorem row to pass through $T, F, N, K, X, H, C, P, L$ before the seal can be consumed.

Use the candidate-mediated SN handoff Definition 740.21. This pins the Candidate or ClosedCandidate evidence to the same displayed frontier and prevents it from becoming arbitrary closed subject reduction or full closed consistency.

Next apply the reduction determinacy and residual schedule theorems, Definition 740.22 and Definition 740.24. They identify the finite classifier boundary and require every residual read for strong-normalization or confluence checking to reuse the same replay and provenance rows.

Finally use the MetaCIC discharge frontier Definition 740.23. The frontier leaves closed consistency behind the subject-reduction socket, so the exposed boundary is exactly the candidate, closed-normal, residual-schedule, transport, replay, provenance, and local-name surface stated above. □

Theorem 740.26 (Normal-form consistency seal MetaCIC critical-path handoff)**.** *The boundary of Definition 740.25 is the handoff consumed by* MetaCICCriticalPath$^{\uparrow}$ *and by confluence-facing packets that inspect* NormalFormConsistencySeal$^{\uparrow}$*. Such consumers may cite the candidate route, closed-normal frontier, residual schedule, transport, replay, provenance, and local naming rows,*

but they may not read subject reduction, global closed consistency, evaluator equality, or quotient syntax equality from the seal itself.

Proof. Apply Definition 740.25. It packages the normal endpoint read as a finite boundary consisting of candidate evidence, closed-normal frontier rows, residual schedule rows, transport, replay, provenance, and local naming.

Use Definition 740.23. The MetaCICCriticalPath$^{\uparrow}$ consumer receives the boundary as an open handoff to the subject-reduction discharge socket, not as a completed closed-consistency theorem.

For confluence-facing packets, use Definition 740.24. The same schedule controls candidate-mediated strong-normalization and confluence checking before any consumer inspects the closed-normal theorem row.

The non-escape, candidate-mediated handoff, and reduction determinacy theorems then rule out stronger readings. The seal supplies no row for evaluator equality, quotient syntax equality, arbitrary subject reduction, or global closed consistency, so every MetaCIC critical-path or confluence-facing read is confined to the stated boundary. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealMetacicCriticalPathHandoff`
↪ variant `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealMetaCICCriticalPathHandoff`

Theorem 740.27 (Normal-form consistency seal obligation closure route)**.** *The carrier admission of Definition 740.1, the NameCert obligations of Definition 740.2, the candidate route of Definition 740.20, the residual-context obligation of Definition 740.11, the joinability obligation of Definition 740.12, finite-window determinacy of Definition 740.13, residual schedule discharge of Definition 740.24, the candidate boundary of Definition 740.25, and the MetaCIC critical-path handoff of Definition 740.26 together form the* obligation *route for the same* NormalFormConsistencySeal$^{\uparrow}$ *packet. The route records the displayed candidate, residual, joinability, determinacy, schedule, boundary, transport, replay, provenance, and local-name obligations, while closed consistency and hidden subject reduction remain outside the packet.*

Proof. Each cited theorem projects or replays the same accepted packet $S = (T, F, N, K, X, H, C, P, L)$. The carrier and NameCert obligations expose the finite rows, the candidate and residual theorems order the normal-endpoint reads, and the joinability and determinacy obligations keep those reads inside the same finite window.

The residual schedule, candidate boundary, and MetaCIC handoff then repack the same rows for strong-normalization, confluence, and critical-path consumers. None of those steps adds a row for global closed consistency or hidden subject reduction, so the assembled route is exactly the stated obligation surface. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealObligationClosureRoute`

Theorem 740.28 (Normal-form consistency seal public export)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* NormalFormConsistencySeal$^{\uparrow}$ *packet. Its public export is exactly the consumer-facing route through the candidate route, closed-normal frontier, residual schedule, componentwise transport, continuation replay, provenance, and local naming rows:*

$$T, F, N, K, X, H, C, P, L.$$

Every MetaCIC critical-path or confluence-facing consumer may cite that route as a public seal surface only after the candidate evidence, closed-normal frontier rows, residual schedule rows, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row are displayed. The export supplies no closed subject reduction, hidden normalization transport, evaluator equality, quotient syntax equality, or host equality.*

Proof. Start with the candidate route and closed-normal frontier. Theorems 740.20 and 740.4 force every normal-endpoint read to pass through the displayed candidate evidence and the rows T, F, N, K, X before it can be consumed.

Next use the residual schedule and determinacy rows. Theorems 740.24 and 740.13 keep strong-normalization and confluence-facing reads inside the same finite schedule and prevent a second residual route from being opened.

The transport, replay, provenance, and local naming rows are fixed by Definition 740.16, Definition 740.17, and Definition 740.18. These theorems identify the public support rows as H, C, P, L and do not add any hidden normalization coordinate.

Finally apply Definition 740.25 and Definition 740.26. They package the same rows for MetaCIC and confluence-facing consumers while keeping closed subject reduction, evaluator equality, quotient syntax equality, and host equality outside the packet. Thus the public export is exactly the displayed finite surface. □

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealPublicExport`

Theorem 740.29 (Normal-form consistency seal mature treatment)**.** *Let $S = (T, F, N, K, X, H, C, P, L)$ be an accepted* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *packet. Its mature treatment is the route that first exposes the candidate route of Definition 740.20, then the residual schedule of Definition 740.24, and then the MetaCIC handoff of Definition 740.26. The route permits MetaCIC and confluence-facing consumers to inspect the closed-normal theorem row only through the displayed candidate, residual, schedule, transport, replay, provenance, and local naming rows. It refuses arbitrary normal endpoints, global closed consistency, hidden subject reduction, evaluator equality, quotient syntax equality, host equality, and any completion of the seal outside those rows.*

Lean4 ✓ `BEDC.Derived.NormalFormConsistencySealUp.NormalFormConsistencySealMatureTreatment`

Proof. Read the candidate route first. The cited route fixes the only admissible normal-endpoint evidence available to S and keeps that evidence attached to the displayed rows T, F, N, K, X.

Next read the residual schedule. The residual schedule discharge theorem supplies the single schedule used for candidate-mediated strong-normalization and confluence checks, while the finite-window determinacy theorem prevents a second residual route from being opened.

Finally use the MetaCIC handoff and the public export surface. They package the same candidate route, residual schedule, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row for downstream consumers. Because none of these rows contains global closed consistency, hidden subject reduction, evaluator equality, quotient syntax equality, or host equality, the mature treatment is exactly the stated refused-endpoint boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalFormConsistencySeal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ packet is generated from closed-context typing, false-type, beta-strong normality, closed-normal consistency, subject-reduction boundary, transport, continuation, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalFormConsistencySeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 740.1, Definition 740.2, Definition 740.3, Definition 740.14, Definition 740.16, Definition 740.17, Definition 740.18, Definition 740.27, Definition 740.28, and Definition 740.29 bind carrier admission, boundary routing, normal-form non-escape, continuation replay, provenance, local naming, the finite obligation route, the public export surface, and the mature candidate-residual-MetaCIC route for the same seal packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalFormConsistencySeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: human

Not claimed: This chapter keeps full closed consistency, closed subject reduction, hidden normalization transport, and host equality outside the finite seal.

Upgrade path: Mature closure requires machine-checked bridge verification for the listed public export and MetaCIC handoff surface, without adding closed subject reduction, hidden normalization transport, or host equality.

Chapter 741

A Concrete Naming Certificate for $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$

The $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ packet turns the normal-source branch of chapter 9378 into finite BEDC data. It is placed after the typed reduction and preservation surfaces of chapter 562 and chapter 483: the present packet records the closed normal source, two visible beta-star routes, their displayed join, and the boundary row that prevents this normal-source seal from being read as full closed Church–Rosser.

Definition 741.1 (Closed normal confluence seal carrier)**.** *A* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (t, n, r_1, r_2, j, H, C, P, N)$$

with the following displayed rows:

1. *t is the closed MetaCIC term row serving as the source of both beta-star routes;*
2. *n is the beta-normality witness row for t;*
3. *r_1 is the first displayed* BetaStarStep *route from t to a target row;*
4. *r_2 is the second displayed* BetaStarStep *route from t to a target row;*
5. *j is the join row supplied by the normal-source confluence theorem, recording the common target and the two residual beta-star routes into it;*
6. *H records componentwise* hsame *transports for the source, normality, route, join, provenance, and local certificate rows;*
7. *C records the displayed* Cont *routes by which consumers read the closedness witness, the normality witness, the two beta-star routes, and the join row;*
8. *P is the* Pkg *provenance over MetaCIC closed terms, beta normality,* BetaStarStep*, normal-source confluence, the closed-substitution compatibility obstruction,* BHist*,* hsame*,* Cont*, and sibling* NameCert *packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary closed-source confluence, no proof of BetaParallelSubstitutionCompatibi *no closed parallel diamond for all closed terms, and no hidden subject-reduction or normalization oracle.*

Lean4 (definition) `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealUp`
Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealTasteGate single carrier alignment`

Theorem 741.2 (Closed normal confluence seal normal source)**.** *Every accepted* ClosedNormalConfluenceSeal$^{\uparrow}$ *carrier exposes confluence only from the closed beta-normal source row t. A consumer may read t, n, r_1, r_2, j through H, C, P, N and obtain the displayed join of the two beta-star routes, but it cannot use the packet as a confluence theorem for an arbitrary closed source.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealNormalSource`

Proof. Project the carrier of Definition 741.1. The source row t and the normality witness row n are not auxiliary decoration: they are the hypotheses that select the normal-source theorem named in chapter 9378.

The routes r_1 and r_2 are the only beta-star routes exposed by the packet, and the join row j is read only as their common normal-source join. The transport and route rows H and C replay this finite surface without adding new sources.

The sibling packets provide the typed-route and preservation discipline around the read. chapter 562 keeps terminal route access finite, while chapter 483 keeps target membership reads behind the displayed preservation boundary.

Since no coordinate supplies an arbitrary closed-source theorem, a closed parallel diamond, or a substitution-compatibility discharge, the accepted seal cannot be repacked into full closed Church–Rosser. □

Theorem 741.3 (Closed normal confluence seal join-window inversion)**.** *For every accepted* ClosedNormalConfluenc *carrier $Z = (t, n, r_1, r_2, j, H, C, P, N)$, any consumer that reads the join row j must first expose the closed source row t, the beta-normality row n, and both displayed* BetaStarStep *routes r_1, r_2. Conversely, the row j exports only the common target and residual routes attached to those two displayed routes.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealJoinWindowInversion`

Proof. Project the carrier of Definition 741.1. The join coordinate j is not an independent global confluence field; it is listed after the closed source t, the beta-normality witness n, and the displayed routes r_1, r_2.

Apply Definition 741.2. It fixes the available source as the displayed closed beta-normal row and keeps both beta-star routes visible before the join is read.

The structural rows H, C, P, N transport, replay, source, and name only this finite window. Therefore a consumer of j recovers precisely the common target and residual routes attached to r_1, r_2, while arbitrary closed Church–Rosser, closed parallel diamond, substitution compatibility, subject reduction, and normalization remain outside the packet. □

Theorem 741.4 (Closed normal confluence seal public export)**.** *The public export of an accepted* ClosedNormalConfluenceSeal$^{\uparrow}$ *packet is exactly the displayed closed beta-normal source row t, the beta-normality witness row n, the two beta-star route rows r_1, r_2, the join row j, and the structural surface H, C, P, N for componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. This export contains no arbitrary closed Church–Rosser theorem, no* BetaParallelSubstitutionCompatibility *discharge, no closed parallel diamond, and no subject-reduction or normalization oracle.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealPublicExport`

Proof. Start with the normal-source theorem Definition 741.2. It identifies the source of the export as the displayed closed beta-normal row t together with the normality witness n and the two beta-star route rows r_1, r_2.

The join row j is therefore read only as the join for those two displayed routes. The transport, replay, provenance, and naming rows H, C, P, N preserve that finite surface for public consumers without adding a new source or a substitution theorem.

The carrier of Definition 741.1 has no coordinate for arbitrary closed Church–Rosser, closed parallel diamond, subject reduction, or normalization. Hence the public export is exactly the finite normal-source seal and not any of the excluded global endpoints. □

Theorem 741.5 (Closed normal confluence seal substitution boundary)**.** *The* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ *carrier separates the available normal-source join from the substitution-compatibility boundary. Agreement on* $t, n, r_1, r_2, j, H, C, P, N$ *records the checked closed-normal confluence read, but it does not produce* BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, or arbitrary closed-term confluence for* BetaStarStep*.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealSubstitutionBoundary`

Proof. Read the provenance row P. It names the checked normal-source confluence surface and also names the closed-substitution compatibility obstruction recorded by chapter 9378.

These two entries have different roles. The normal-source entry justifies the join row j for the displayed closed beta-normal source, while the obstruction entry keeps the substitution interface visible as a boundary rather than as a field filled by the carrier.

The structural rows H, C, N preserve this separation. They transport and replay the finite seal rows, but no row in the carrier contains the substitution compatibility proof, the closed parallel diamond, or the general closed star-confluence endpoint.

Therefore the packet is a local confluence seal for closed normal sources. It is not a certificate for the substitution-sensitive branch that would be needed to close arbitrary closed-term Church–Rosser. □

Theorem 741.6 (Closed normal confluence seal obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}}$ *obligation surface consists of carrier admission for*

$$t, n, r_1, r_2, j, H, C, P, N,$$

closed normal-source exposure, the two displayed beta-star route rows, the displayed join row, route stability, transport stability, provenance, and boundary nonescape from substitution compatibility or full Church–Rosser.

Proof. Carrier admission is the row inventory of Definition 741.1. It exposes the closed source t, the normality row n, the two beta-star routes r_1, r_2, the join row j, and the structural rows H, C, P, N.

The normal-source component is supplied by Definition 741.2. It restricts the join row to the displayed closed beta-normal source and prevents the route from being read as arbitrary closed-source confluence.

The boundary clauses are supplied by Definition 741.5. They keep substitution compatibility, the closed parallel diamond, and full closed Church–Rosser outside the obligation surface while leaving the finite normal-source seal available to local consumers. □

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealObligationSurface`

Theorem 741.7 (Closed normal confluence seal carrier admission ledger)**.** *An accepted* ClosedNormalConfluenceSea *carrier admits exactly the displayed rows*

$$t, n, r_1, r_2, j, H, C, P, N$$

as its local admission ledger: closed source, beta-normality, two beta-star routes, their normal-source join, componentwise transport, continuation replay, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealCarrierAdmissionLedger`

Proof. Project the carrier in Definition 741.1. The first five rows record the closed source, normality witness, two beta-star routes, and the join row supplied by the normal-source confluence theorem.

The final four rows record the admissible BEDC infrastructure for reading those data. The transport row H and replay row C keep the read componentwise and displayed, while P and N record provenance and local naming.

No further coordinate is available in the carrier. Hence the admission ledger is exactly $t, n, r_1, r_2, j, H, C, P, N$. □

Theorem 741.8 (Closed normal confluence seal route stability)**.** *If two accepted* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ *carriers agree componentwise along the displayed* hsame *row, every permitted* Cont *replay of the normal-source route for one carrier transports to the corresponding replay of the other carrier.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealRouteStability`

Proof. Read the componentwise transport row H. It identifies the closed source, normality witness, route rows, join row, provenance row, and local certificate row only along the displayed hsame surface.

Now replay through C. Each permitted continuation reads the same finite coordinates in the same order: closedness and normality first, then the two beta-star routes, and finally the join row. Transport through H therefore sends any such read to the corresponding read of the transported packet.

The proof uses no external confluence theorem beyond the already displayed normal-source join. Stability is a property of the finite replay route, not a new Church–Rosser endpoint. □

Theorem 741.9 (Closed normal confluence seal scoped route)**.** *Every permitted consumer route out of an accepted* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ *carrier factors through the displayed closed source* t*, beta-normality row* n*, two* BetaStarStep *rows* r_1, r_2*, join row* j*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}}$ *row* N*, and the single-carrier alignment surface named by BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenc No such route bypasses this finite chain to obtain arbitrary closed Church–Rosser data.*

Proof. Start with an accepted carrier and project the rows listed in Definition 741.1. The only source available to a consumer is the closed row t, and the only normality evidence available for that source is n.

The replay then factors through the two displayed beta-star rows r_1, r_2 and the join row j. The transport row H can move the route only componentwise, and the continuation row C can replay only the already displayed closed source, normality, route, and join coordinates.

The provenance row P and local certificate row N bind this route to the same NameCert packet. The TasteGate single-carrier alignment identifies the accepted carrier surface used by the route, so a consumer cannot change carrier shape while preserving acceptance.

Thus the permitted route is exactly the finite route through the displayed closed source, normality row, beta-star rows, join row, transport, replay, provenance, local naming, and TasteGate alignment. Nothing in the factorization supplies substitution compatibility, a closed parallel diamond, or full closed Church–Rosser. □

Theorem 741.10 (Closed normal confluence seal kernel scope)**.** *The kernel scope of the* ClosedNormalConfluenceSea *packet is exactly the BEDC infrastructure* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *together with the MetaCIC surfaces for closed-term rows, beta-normality rows,* BetaStarStep *rows, typed reduction, and subject reduction. The scope does not include substitution compatibility, a closed parallel diamond, a normalization oracle, or full closed Church–Rosser.*

Proof. Inspect the carrier coordinates. The structural rows are H, C, P, N, and their ambient BEDC primitives are precisely BHist, hsame, Cont, Pkg, and NameCert.

The computational rows are t, n, r_1, r_2, j. They name a closed MetaCIC source, a beta-normality witness, two BetaStarStep routes, and the displayed join supplied for that normal source. The sibling typed-reduction and subject-reduction chapters delimit how these rows may be read by consumers.

No remaining coordinate names substitution compatibility, a closed parallel diamond, arbitrary normalization, or a general Church–Rosser theorem. The scope is therefore the displayed BEDC infrastructure plus the listed MetaCIC surfaces, and nothing beyond them. □

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealKernelScope`

Theorem 741.11 (Closed normal confluence seal boundary nonescape)**.** *No accepted* ClosedNormalConfluenceSeal$^{\uparrow}$ *packet exports residual-substitution compatibility, a full closed parallel diamond, or arbitrary closed-source Church–Rosser data beyond the displayed normal-source seal.*

Lean4 ✓ `BEDC.Derived.ClosedNormalConfluenceSealUp.ClosedNormalConfluenceSealBoundaryNonescape`

Proof. Inspect the carrier rows. The route data stop at r_1, r_2, j, where j is the join supplied for the closed beta-normal source named by t and n.

The provenance row P records the substitution-compatibility obstruction as a boundary. It does not fill a coordinate for BetaParallelSubstitutionCompatibility, nor does any continuation in C replay such a coordinate.

Thus every consumer remains inside the residual-substitution and full Church–Rosser boundary named by the carrier. The packet can expose the local normal-source seal, but it cannot escape to the missing closed parallel diamond or arbitrary closed-term confluence endpoint. □

Closure status for NameCert$_{\text{ClosedNormalConfluenceSeal}^{\uparrow}}$**.**

Constructive story: A ClosedNormalConfluenceSeal$^{\uparrow}$ packet is generated from a closed term row, beta-normality witness row, two beta-star route rows, a displayed join row, componentwise hsame transports, displayed Cont reads, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosedNormalConfluenceSeal}^{\uparrow}}$, publicClosed)

Scope closed: Definition 741.1, Definition 741.6, Definition 741.7, Definition 741.8, Definition 741.9, Definition 741.10, Definition 741.11, and Definition 741.4 bind carrier admission, route stability, kernel scope, boundary nonescape, and the public finite normal-source export.

Formal verification: FormalStatus(NameCert$_{\text{ClosedNormalConfluenceSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary closed-term Church–Rosser, BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, a subject-reduction theorem, a normalization oracle, checked Lean certificate coverage, or a public standard bridge.

Upgrade path: Public closure requires checked carrier admission, normal-source join readback, route stability, transport stability, provenance, and boundary non-escape for the same finite seal packet.

Chapter 742

A Concrete Naming Certificate for $\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}$

The $\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}$ surface turns the build-replay coordinate named by chapter 9364 into a finite $\mathsf{NameCert}$ packet. It refines the replay row of chapter 547, reads accepted declarations through chapter 549, and hands dependency classification to chapter 654. The packet records reproducible build evidence without treating replay as a proof of kernel correctness or as a substitute for the axiom-query boundary.

Definition 742.1 (KernelAcceptanceBuildReplay carrier)**.** *A* $\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, A, B, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *G is the generated-candidate row naming the Lean source object, declaration body, proof term, recursor output, or inductive declaration submitted for checking;*
2. *A is the accepted-declaration row read through the kernel-acceptance witness surface;*
3. *B is the build-command row naming the exact replay command, project root, toolchain row, source selection, and local build target used to reconstruct the acceptance event;*
4. *S is the replay-result row recording whether the build reconstructed the same accepted declaration, failed before acceptance, or reached a refusal state that remains visible to the audit consumer;*
5. *Q is the axiom-query handoff row that connects the replayed declaration to the strict proof-ancestry ledger;*
6. *H records componentwise* hsame *transport for the generated candidate, accepted declaration, build command, replay result, axiom-query handoff, provenance, and local naming row;*
7. *C records the displayed* Cont *routes through which consumers move from generated candidate to accepted declaration, from accepted declaration to replay command, from replay command to replay result, and from replay result to axiom-query handoff;*
8. *P is the* Pkg *provenance over* BHist*, generated candidates, accepted declarations, build commands, replay results, axiom-query handoffs,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}}$ *row.*

The carrier exports no ambient host-trust row, no hidden source scan, no unrecorded command rerun, and no correctness certificate for the Lean kernel itself.

Lean4 ✓ `BEDC.Derived.KernelAcceptanceBuildReplayUp.KernelAcceptanceBuildReplayTasteGate single carrier alignment`

Theorem 742.2 (KernelAcceptanceBuildReplay NameCert obligations)**.** *For every accepted* KernelAcceptanceBuil *carrier*

$$R = (G, A, B, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}}$ *surface consists exactly of the generated-candidate row* G*, accepted-declaration row* A*, build-command row* B*, replay-result row* S*, axiom-query handoff row* Q*, componentwise* hsame *transport* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local naming row* N*. A consumer may read build replay only through this finite packet and may not replace* B *or* S *by an informal claim that the project was rebuilt.*

Proof. Start with the carrier of Definition 742.1. The rows G, A, B, S, Q expose the candidate, accepted declaration, replay command, replay result, and dependency-query handoff as distinct finite coordinates.

The transport and continuation rows bind the reads of those coordinates. The row H preserves them componentwise, while C displays the routes from generated candidate to accepted declaration, from accepted declaration to build command, from build command to replay result, and from replay result to axiom-query handoff.

The provenance and naming rows package the same finite data as a $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}}$ surface. Since no additional replay coordinate is present, a consumer cannot replace the displayed build command or replay result by an unrecorded host assertion. □

Theorem 742.3 (KernelAcceptanceBuildReplay replay non-escape)**.** *For every accepted* KernelAcceptanceBuildRep *carrier*

$$R = (G, A, B, S, Q, H, C, P, N),$$

every downstream audit read of build replay factors through the finite rows A, B, S, Q, C, N*. The replay result may witness that the accepted declaration was reconstructed from the recorded command and source selection, but it cannot export kernel correctness, compiler correctness, runtime correctness, dependency cleanliness without the axiom-query row* Q*, or any paper-side claim stronger than the accepted declaration row* A*.*

Proof. Project the accepted declaration, build command, replay result, axiom-query handoff, route, and local naming rows from the carrier. These rows are the only replay-facing endpoints exposed to an audit consumer.

The route coordinate C determines how the read moves from A to B, from B to S, and from S to Q. Thus replay success remains tied to the recorded command and result, while dependency classification remains tied to the axiom-query handoff.

Finally inspect the omitted coordinates. The packet contains no row for kernel correctness, compiler correctness, runtime correctness, hidden dependency cleanliness, or stronger manuscript meaning, so a replay consumer has no route by which to extract those claims. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelAcceptanceBuildReplay}^{\uparrow}$ packet is generated from a generated candidate row, accepted declaration row, build command row, replay result row, axiom-query handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KernelAcceptanceBuildReplay}\uparrow}$, seed)

Scope closed: Definition 742.1, Definition 742.2, and Definition 742.3 bind the finite build-replay carrier, local naming surface, replay route, axiom-query handoff, and non-escape boundary.

Formal verification: FormalStatus(NameCert$_{\text{KernelAcceptanceBuildReplay}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, kernel correctness, host runtime correctness, compiler correctness, dependency cleanliness without the axiom-query row, or a public standard bridge.

Upgrade path: Obligation closure requires Lean encoding of the finite packet and checked theorems for naming obligations, replay-route determinacy, axiom-query handoff stability, and replay non-escape.

Chapter 743

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceAudit}^{\uparrow}$

The $\mathsf{MetaCICConfluenceAudit}^{\uparrow}$ packet turns the audit map of chapter 9378 into concrete finite BEDC data next to the subject-reduction boundary packet of chapter 731. It names the checked confluence fragments, the substitution-compatibility input, the closed-diamond input, and the arbitrary closed-term Church–Rosser boundary without treating any remaining input as a proved theorem.

743.1 Carrier

Definition 743.1 (MetaCIC confluence audit carrier)**.** *A* $\mathsf{MetaCICConfluenceAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (N, A, S, D, C, H, R, P, M)$$

with the following rows:

1. *N is the checked normal and atom confluence-fragment row;*
2. *A is the checked parallel administrative-fragment row;*
3. *S is the named* BetaParallelSubstitutionCompatibility *input row;*
4. *D is the named* BetaParallelDiamondClosed *input row;*
5. *C is the named arbitrary closed-term Church–Rosser boundary row for* BetaStarStep*;*
6. *H is the componentwise* hsame *transport row for these five audit entries;*
7. *R is the displayed* Cont *read route for confluence consumers;*
8. *P is the* Pkg *provenance over the MetaCIC confluence files and the subject-reduction boundary packet;*
9. *M is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAudit}^{\uparrow}}$ *row.*

The classifier accepts exactly those packets whose consumers read the confluence surface through these rows and do not replace S, D, or C by a closed proof.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceAuditUp.TasteGate.MetaCICConfluenceAuditTasteGate single carrier alignment`

Theorem 743.2 (MetaCIC confluence audit NameCert obligations)**.** *For an accepted* MetaCICConfluenceAudit$^{\uparrow}$ *carrier, the local* NameCert *surface consists of carrier admission, checked-fragment readback for N and A, named-input exposure for S and D, arbitrary-closed-term boundary exposure for C, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from an unchecked closed confluence theorem.*

Proof. Start with the carrier rows of Definition 743.1. They already separate the checked rows N, A from the named rows S, D, C.

Use the local M row to package these rows as the visible certificate. Componentwise transport is supplied by H, while R supplies the only displayed consumer route.

Finally read provenance through P. Since P records the relation to the MetaCIC confluence audit map and the subject-reduction obstruction packet without adding a proof row for S, D, or C, every accepted consumer remains inside the displayed audit packet. □

Theorem 743.3 (MetaCIC confluence audit boundary)**.** *Agreement on the rows of an accepted* MetaCICConfluenceAudit$^{\uparrow}$ *packet preserves the checked normal, atom, and administrative confluence fragments, but it does not yield* BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, or arbitrary closed-term Church–Rosser for* BetaStarStep*.*

Proof. The checked rows N and A are read back directly from the packet, so agreement on Q preserves those fragment claims.

The rows S, D, and C have different status. They are named boundary entries, not theorem rows. Transport along H and reading through R can preserve their names and consumer visibility, but it cannot create a proof object not carried by the packet.

Thus the packet supports finite audit citation while keeping the three remaining confluence inputs outside the closed theorem surface. □

Theorem 743.4 (MetaCIC confluence audit vision handoff)**.** *Every manuscript consumer that cites the confluence audit map through* MetaCICConfluenceAudit$^{\uparrow}$ *factors through $Q = (N, A, S, D, C, H, R, P, M)$ before citing any MetaCIC confluence claim. The handoff exports the checked fragment rows and the three named boundary rows, and it exports no arbitrary closed-term confluence theorem, no closed subject-reduction theorem, and no host-level confluence principle.*

Proof. First follow the citation into chapter 9378. The map supplies the semantic grouping of checked fragments and remaining inputs.

Next pass through the carrier of Definition 743.1. The finite packet turns that grouping into rows N, A, S, D, C with transport, route, provenance, and local naming data.

The sibling boundary of chapter 731 supplies the comparison surface for subject-reduction claims. Combining the two packets gives a citation route, not a closed theorem beyond the displayed rows. □

Closure status for NameCert$_{\text{MetaCICConfluenceAudit}^{\uparrow}}$**.**

Constructive story: A MetaCICConfluenceAudit$^{\uparrow}$ packet is generated from finite BHist rows that group confluence-facing audit material, sibling subject-reduction obstruction rows, transport rows, route rows, provenance, and local naming. The audit does not synthesize a confluence theorem; it collects the displayed rows and uses Cont to replay how a consumer reaches the comparison surface supplied by the sibling boundary. Componentwise hsame preserves the audit grouping and sibling rows, while Pkg records the MetaCIC audit dependencies. The packet therefore offers a finite citation route across audit surfaces, not closed confluence or subject reduction.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConfluenceAudit}^{\uparrow}}$, seed)

Scope closed: This chapter binds the MetaCIC confluence-audit carrier, audit grouping, sibling subject-reduction comparison surface, and finite citation-route boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICConfluenceAudit}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: It does not claim confluence, subject reduction, normalization, host conversion semantics, quotient term equality, or a completed MetaCIC metatheorem.
Upgrade path: Obligation closure requires Lean encodings of the audit carrier, sibling-boundary route, componentwise transport, and citation-route non-closure theorem.

Chapter 744

A Concrete Naming Certificate for ParallelConfluenceAudit$^{\uparrow}$

The ParallelConfluenceAudit$^{\uparrow}$ packet realizes the confluence-side audit map of chapter 9378 as a finite NameCert surface. It sits next to the beta-substitution packet of chapter 716 and the beta-discharge packet of chapter 727; the closed-term side is compatible with the seal of chapter 722. The packet records the parallel-step route, substitution-compatibility boundary, conditional parallel-diamond handoff, closed-diamond-to-star-confluence handoff, closed-normal and atom fragments, non-claim boundary, componentwise transports, displayed routes, provenance, and local naming row without turning the conditional Lean bridges into an unconditional closed Church–Rosser theorem.

Definition 744.1 (ParallelConfluenceAudit carrier)**.** *A* ParallelConfluenceAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, S, D, C, Nm, At, No, H, R, L, G)$$

with the following displayed rows:

1. *P is the parallel-step route row for the MetaCIC* BetaParallel *geometry;*
2. *S is the substitution-compatibility boundary row naming* BetaParallelSubstitutionCompatibility *as an input, not as a proved coordinate;*
3. *D is the conditional parallel-diamond handoff row corresponding to the route from S to the parallel diamond;*
4. *C is the closed-diamond-to-star-confluence handoff row for the closed* BetaStarStep *confluence route;*
5. *Nm is the closed-normal fragment row recording the unconditional normal-source cases;*
6. *At is the atom and non-lambda shape row recording the unconditional atom-side and impossible-shape fragments;*
7. *No records the non-claim boundary for arbitrary closed Church–Rosser,* BetaParallelSubstitutionCompatibility*, and* BetaParallelDiamondClosed*;*
8. *H records componentwise* hsame *transports for $P, S, D, C, Nm, At, No, L, G$;*
9. *R displays the* Cont *routes through which consumers may read the parallel-step, conditional bridge, closed-star, normal-fragment, atom-fragment, and non-claim rows;*

10. *L is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, BetaSubstitutionPreservation$^{\uparrow}$, BetaSubstitutionDischarge$^{\uparrow}$, ClosedTermContextSeal$^{\uparrow}$, *and the MetaCIC confluence audit map;*
11. *G is the local* $\mathsf{NameCert}_{\mathsf{ParallelConfluenceAudit}^{\uparrow}}$ *row.*

The carrier is exactly this finite packet. It contains no proof of BetaParallelSubstitutionCompatibility, *no proof of* BetaParallelDiamondClosed, *and no arbitrary closed Church–Rosser theorem for* BetaStarStep.

Lean4 ✓ `BEDC.Derived.ParallelConfluenceAuditUp.ParallelConfluenceAuditTasteGate single carrier alignment`

Theorem 744.2 (ParallelConfluenceAudit NameCert obligations)**.** *For the carrier of Definition 744.1, the local* $\mathsf{NameCert}_{\mathsf{ParallelConfluenceAudit}^{\uparrow}}$ *surface consists of carrier admission for*

$$A = (P, S, D, C, Nm, At, No, H, R, L, G),$$

parallel-step route visibility, substitution-boundary visibility, conditional parallel-diamond handoff visibility, closed-diamond-to-star-confluence handoff visibility, closed-normal fragment visibility, atom and non-lambda fragment visibility, non-claim exactness, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier. The inventory is exactly $P, S, D, C, Nm, At, No, H, R, L, G$, so every local naming obligation is read from a displayed finite row.

The positive confluence-side rows are P, D, C, Nm, At. The route P records the parallel-step geometry, D records only the substitution-conditional parallel-diamond handoff, C records only the closed-diamond-to-star-confluence handoff, and Nm, At record the unconditional closed-normal and atom fragments.

The boundary rows are S and No. The row S names the substitution-compatibility input as a premise boundary, while No records the absence of an arbitrary closed Church–Rosser theorem and the absence of proofs of the two named premises.

Finally H, R, L, G provide transport, continuation replay, provenance, and the local name for the same packet. Since no other coordinate exists, the obligations are exhausted by the displayed rows. □

Theorem 744.3 (ParallelConfluenceAudit conditional bridge)**.** *Let $A = (P, S, D, C, Nm, At, No, H, R, L, G)$ be an accepted* $\mathsf{ParallelConfluenceAudit}^{\uparrow}$ *carrier. Any read of the conditional confluence bridge reaches the parallel-diamond row D only through the substitution-compatibility boundary S, and reaches the closed-star confluence handoff C only through the parallel-diamond handoff row D. The closed-normal row Nm and atom row At remain unconditional fragments, not replacements for S, D, C.*

Lean4 ✓ `BEDC.Derived.ParallelConfluenceAuditUp.ParallelConfluenceAudit conditional bridge rows`

Proof. Begin with the conditional route data. The displayed continuation row R exposes the parallel-step route P, then the substitution boundary S, and only then the parallel-diamond handoff D.

The closed-star handoff is downstream of D. A consumer that reads C therefore reads the closed-diamond-to-star bridge as a handoff from the parallel diamond, not as an independent proof of the diamond premise.

The rows Nm and At are separate fragment rows. They may be cited for closed-normal sources, atom sources, and non-lambda shape cases, but their carrier positions do not provide either S or D.

The package row L and local name G keep these dependencies in one NameCert packet. Thus every bridge read factors through S, D, C in that order. □

Lean4 ✓ `BEDC.Derived.ParallelConfluenceAuditUp.ParallelConfluenceAudit conditional bridge route closed`

Theorem 744.4 (ParallelConfluenceAudit non-escape)**.** *Every accepted* ParallelConfluenceAudit$^{\uparrow}$ *packet exposes confluence data only through the rows*

$$P, S, D, C, Nm, At, No, H, R, L, G.$$

It exports no arbitrary closed Church–Rosser theorem for BetaStarStep*, no proof of* BetaParallelSubstitutionCompa *and no proof of* BetaParallelDiamondClosed*.*

Proof. Inspect the carrier coordinates. The only positive rows are the parallel-step route, the two conditional handoffs, and the closed-normal and atom fragments.

Use Definition 744.3. Reads of the bridge rows must pass through S, D, C, so the conditional premises remain visible at every consumer-facing route.

Use Definition 744.2 for local naming and provenance. The non-claim row No is part of the same finite packet as the displayed routes, and the missing endpoint theorems have no carrier coordinate.

Therefore a downstream consumer may cite the audit packet only as the finite confluence-side map described above. It cannot turn the map into arbitrary closed Church–Rosser, a proof of substitution compatibility, or a proof of the closed parallel diamond. □

Closure status for NameCert$_{\text{ParallelConfluenceAudit}^{\uparrow}}$.

Constructive story: A ParallelConfluenceAudit$^{\uparrow}$ packet is generated from a parallel-step route, substitution-compatibility boundary, conditional parallel-diamond handoff, closed-diamond-to-star-confluence handoff, closed-normal fragment row, atom and non-lambda shape row, non-claim row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ParallelConfluenceAudit}^{\uparrow}}$, seed)

Scope closed: Definition 744.1, Definition 744.2, Definition 744.3, and Definition 744.4 bind the finite confluence audit packet and its conditional bridge boundary.

Formal verification: FormalStatus(NameCert$_{\text{ParallelConfluenceAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove arbitrary closed Church–Rosser for BetaStarStep, BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, conditional-route exactness through S, D, C, fragment coverage for Nm and At, and non-escape for the named confluence endpoints.

Chapter 745

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}$

The $\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}$ packet turns the audit boundary of chapter 9378 into a finite NameCert witness. It is narrower than chapter 744: the present chapter records the minimum carrier needed to audit the conditional confluence boundary, while chapter 731 and chapter 713 provide sibling lattice anchors for obstruction and confluence-style route comparison.

Definition 745.1 (MetaCIC confluence audit witness carrier)**.** *A* $\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (P, S, D, C, O, H, R, L, N)$$

with the following displayed rows:

1. *P is the MetaCIC parallel-step route row whose endpoints are closed terms;*
2. *S is the substitution-compatibility premise boundary for the parallel route;*
3. *D is the conditional parallel-diamond handoff row reached only through S;*
4. *C is the closed-star confluence handoff row reached only through D;*
5. *O records the obstruction boundary excluding full closed Church–Rosser, substitution compatibility, and unconditional confluence for arbitrary closed terms;*
6. *H records componentwise* hsame *transport across P, S, D, C, O, R, L, N;*
7. *R is the finite* Cont *route ledger through which consumers may replay the witness route;*
8. *L is the* Pkg *provenance row linking the audit map, the subject-reduction obstruction sibling, and the axis-carry confluence sibling;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}}$ *row.*

The carrier contains no host evaluator, no hidden substitution lemma, no quotient identification of reductions, and no theorem stronger than the displayed conditional route rows.

Lean4 (definition) `BEDC.Derived.MetacicConfluenceAuditWitnessUp.MetacicConfluenceAuditWitnessUp`
Lean4 ✓ `BEDC.Derived.MetacicConfluenceAuditWitnessUp.MetacicConfluenceAuditWitnessTasteGate`
`single carrier alignment`

Theorem 745.2 (MetaCIC confluence audit witness conditional boundary)**.** *Every accepted* MetaCICConfluenceAud *packet reads the confluence route in the order*

$$P \longrightarrow S \longrightarrow D \longrightarrow C.$$

The parallel-diamond handoff D is conditional on the substitution-compatibility boundary S, and the closed-star confluence handoff C is conditional on the displayed diamond handoff D.

Proof. Project the carrier of Definition 745.1. The only route ledger is R, and its route endpoints are the displayed rows P, S, D, C.

The packet therefore has no route from P directly to D and no route from P directly to C. A consumer must first read the substitution-compatibility boundary S, then the conditional diamond row D, and only then the closed-star handoff C.

The obstruction row O records exactly the missing shortcuts. Thus the conditional order is part of the finite witness boundary rather than an external convention. □

Lean4 ✓ `BEDC.Derived.MetacicConfluenceAuditWitnessUp.MetacicConfluenceAuditWitnessCarrier conditional boundary`

Theorem 745.3 (MetaCIC confluence audit witness non-escape)**.** *An accepted* MetaCICConfluenceAuditWitness$^{\uparrow}$ *packet exposes only the rows*

$$P, S, D, C, O, H, R, L, N.$$

It exports no full closed Church–Rosser theorem, no proof of substitution compatibility, and no unconditional confluence theorem for arbitrary closed terms.

Proof. Inspect the carrier definition. Its coordinates are exactly $P, S, D, C, O, H, R, L, N$, so every public read must be a projection from one of these rows or a route recorded by R.

By Definition 745.2, the positive confluence read remains conditional at S and D. The row O is stored in the same packet and names the missing endpoints rather than discharging them.

Since no coordinate contains a host evaluator, a substitution lemma, a quotient equality, or an arbitrary closed-term confluence theorem, the packet has no source from which to export such data. □

Theorem 745.4 (MetaCIC confluence audit witness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAud}}$ *surface consists of carrier admission for $W = (P, S, D, C, O, H, R, L, N)$, route visibility for P, S, D, C, obstruction-boundary visibility for O, componentwise* hsame *transport through H, finite* Cont *replay through R,* Pkg *provenance through L, and local naming through N.*

Proof. Read the packet as a finite carrier. Carrier admission names all displayed rows, and the local name N records that the same packet supplies the witness surface.

The positive obligations are route visibility for P, S, D, C and transport through H, R. The boundary obligation is the visibility of O, which keeps the missing full confluence claims in the packet rather than outside it.

Finally, L supplies provenance for the audit map and the sibling lattice anchors. The NameCert obligations are exhausted by these displayed coordinates because the carrier has no additional field. □

Theorem 745.5 (MetaCIC confluence audit witness sibling lattice)**.** *The* MetaCICConfluenceAuditWitness$^{\uparrow}$ *packet sits between the confluence audit map of chapter 9378, the subject-reduction obstruction packet of chapter 731, and the axis-carry confluence packet of chapter 713. Agreement on H, R, L, N preserves the common NameCert transport and route surface, while the rows S, D, C, O keep the MetaCIC confluence boundary separate from subject reduction and from axis-carry normalization.*

Proof. The provenance row L cites the three anchors, so the packet is read as a finite witness in the same concrete-instance lattice.

The shared rows H, R, L, N provide componentwise transport, continuation replay, provenance, and local naming. These are the only common operations exported to sibling chapters.

The boundary rows remain specific to this packet. The row S names substitution compatibility, D names the conditional parallel diamond, C names the closed-star handoff, and O blocks the missing full confluence claims. Hence the sibling lattice preserves comparison without collapsing the three chapter surfaces into one theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}$ packet is generated from a parallel-step route, substitution-compatibility boundary, conditional parallel-diamond handoff, closed-star confluence handoff, obstruction boundary, componentwise hsame transport, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 745.1, Definition 745.2, Definition 745.3, Definition 745.4, and Definition 745.5 bind the finite audit-witness packet for the conditional MetaCIC confluence boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceAuditWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full closed Church–Rosser, substitution compatibility, or unconditional confluence for arbitrary closed terms.

Upgrade path: Obligation closure requires checked carrier admission, conditional route exactness through S, D, C, non-escape for O, and sibling-lattice provenance through L.

Chapter 746

A Concrete Naming Certificate for MetaCICDecidableAuditPacket$^{\uparrow}$

The MetaCICDecidableAuditPacket$^{\uparrow}$ packet turns the audit map of chapter 9379 into a finite NameCert surface. It sits beside chapter 743 and chapter 739: those chapters separate confluence and consistency boundaries, while the present packet separates checked typing inference from bounded normal-form comparison.

Definition 746.1 (MetaCIC decidable audit packet carrier)**.** *A* MetaCICDecidableAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (C, M, T, G, B, E, O, H, R, P, N)$$

with the following visible rows:

1. *C is the checker row for* `inferTypeCtx` *and its closed wrapper;*
2. *M is the same-term row for* `Term.eq` *and* `checkSameTerm`*;*
3. *T records typing soundness and completeness for the represented* `HasType` *relation;*
4. *G is the gallery row for the closed Boolean, natural-number, pair, option, list, sum, and polymorphic witnesses;*
5. *B is the bounded-normal comparison row for* `normalizeBounded` *and* `decideClosedBetaConv`*;*
6. *E is the boundary row requiring finished closed-normal data before beta-conversion soundness is read as a normal-form equality statement;*
7. *O is the obstruction row excluding unrestricted closed beta-conversion decidability and a standalone closedness-decision theorem;*
8. *H records componentwise* hsame *transport across the checker, same-term, typing, gallery, bounded-normal, boundary, and obstruction rows;*
9. *R records the displayed* Cont *route by which a consumer reads the rows in packet order;*
10. *P is the* Pkg *provenance row tying the decidability audit map to the confluence and consistency sibling packets;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableAuditPacket}^{\uparrow}}$ *row.*

The packet exposes checked syntax equality, checked typing inference, gallery acceptance, and bounded normal comparison with its stated boundary. It contains no global beta-conversion oracle, no hidden normalization endpoint, and no host-level decision procedure for arbitrary closed terms.

Theorem 746.2 (MetaCIC decidable audit packet NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableAuditPa}}$ *surface consists of carrier admission for* $A = (C, M, T, G, B, E, O, H, R, P, N)$*, checker exposure through* C*, same-term exposure through* M*, typing soundness and completeness exposure through* T*, gallery exposure through* G*, bounded-normal comparison exposure through* B*, boundary exposure through* E, O*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* R*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MetacicDecidableAuditPacketUp.MetacicDecidableAuditPacketTasteGate single carrier alignment`

Proof. Project the carrier of Definition 746.1. The rows $C, M, T, G, B, E, O, H, R, P, N$ are the whole public packet, so carrier admission and local naming are direct projections.

The positive checker surface is read through C, M, T, G, B. These rows record the type checker, same-term checker, typing soundness and completeness, gallery rows, and bounded-normal comparison as separate coordinates rather than as one unnamed decidability claim.

The boundary surface is read through E and O. The row E keeps finished closed-normal data explicit, while O blocks unrestricted beta-conversion decidability and a standalone closedness-decision theorem.

Finally H, R, P, N provide transport, route replay, provenance, and local naming. Since no additional coordinate exists, the NameCert obligations are exhausted by the displayed packet. □

Theorem 746.3 (MetaCIC decidable audit seed obligation surface)**.** *Every accepted* MetaCICDecidableAuditPacket *carrier*

$$A = (C, M, T, G, B, E, O, H, R, P, N)$$

exposes exactly the decidable-audit source row C*, the MetaCIC candidate rows* M, T, G*, the finite reduction-window row* B*, the audit verdict and boundary rows* E, O*, transport through* H*, replay through* R*, provenance through* P*, and the local* NameCert *row* N *before any decidability consumer reads the packet. The exposed surface refuses unrestricted beta-conversion decidability, global normalization, quotient equality of terms, and bridge checking.*

Proof. Project the carrier of Definition 746.1. The displayed rows $C, M, T, G, B, E, O, H, R, P, N$ are the whole accepted packet, and the existing TasteGate marker records this as the single-carrier alignment witness.

Read the positive audit surface first. The source row C exposes the checked inference entry point, while M, T, G keep same-term comparison, typing soundness and completeness, and gallery candidates as separate MetaCIC rows.

Next read the finite reduction window. The row B is the only bounded-normal comparison coordinate, and E, O record the audit verdict boundary before a consumer treats that comparison as public.

Finally H, R, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, unrestricted beta-conversion decidability, global normalization, quotient equality of terms, and bridge checking remain outside the seed obligation surface. □

Lean4 ✓ `BEDC.Derived.MetacicDecidableAuditPacketUp.MetacicDecidableAuditPacket seed obligation surface`

Theorem 746.4 (MetaCIC decidable audit packet non-escape)**.** *Every accepted* MetaCICDecidableAuditPacket$^{\uparrow}$ *carrier exposes only the finite rows*

$$C, M, T, G, B, E, O, H, R, P, N.$$

A consumer may read checked typing inference, same-term equality, typing soundness and completeness, gallery acceptance, bounded-normal comparison, finished-normal boundary data, hsame *transport,* Cont *route replay,* Pkg *provenance, and the local* NameCert *row. The consumer obtains no unrestricted beta-conversion decider, no hidden finished-normalization theorem, no quotient equality of terms, and no host-level decision procedure for arbitrary closed terms.*

Proof. Start with the carrier definition. The packet has exactly the displayed rows, and every public read must be a projection from those rows or a route recorded by R.

Use the NameCert obligation theorem to locate the positive routes. Typing and same-term checks are routed through C, M, T, gallery acceptance through G, and bounded comparison through B.

The obstruction rows remain in the same finite packet. The boundary row E requires finished closed-normal data for the stronger normal-form reading, and the obstruction row O records the missing unrestricted conclusions.

Therefore the packet cannot export a route that is not already present in $C, M, T, G, B, E, O, H, R, P, N$. In particular, unrestricted beta-conversion decidability and hidden normalization endpoints have no carrier coordinate. □

Theorem 746.5 (Typed conversion window obligation surface)**.** *Every consumer reading a typed-conversion claim from a* MetaCICDecidableAuditPacket$^{\uparrow}$ *carrier* $A = (C, M, T, G, B, E, O, H, R, P, N)$ *must expose the typing rows* C, M, T *and the bounded-normal rows* B, E, O *as separate windows. The checked typing-inference route through* `inferTypeCtx`*,* `Term.eq`*,* `checkSameTerm`*, and* `HasType` *soundness/completeness cannot be read as unrestricted beta-conversion decidability; bounded normal comparison is available only through* B*, finished closed-normal hypotheses remain explicit in* E*, and the obstruction row* O *records the excluded global conclusions.*

Proof. Project the packet rows from Definition 746.1. The checker and same-term rows C, M supply the syntactic checking surface, while T is the only row that records typing soundness and completeness for the represented `HasType` relation.

Route conversion-facing reads through a different window. The bounded comparison row B names the `normalizeBounded` and `decideClosedBetaConv` surface, and E keeps the finished closed-normal hypotheses visible before a normal-form equality claim is read.

Finally use the obstruction row O. It excludes unrestricted closed beta-conversion decidability and standalone closedness decision, so a consumer cannot collapse the typing window C, M, T into a global conversion oracle. The structural rows H, R, P, N transport, replay, package, and name the same separated route. □

Closure status for NameCert$_{\text{MetaCICDecidableAuditPacket}^{\uparrow}}$**.**

Constructive story: A MetaCICDecidableAuditPacket$^{\uparrow}$ packet is generated from checker, same-term, typing, gallery, bounded-normal comparison, boundary, obstruction, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICDecidableAuditPacket}^{\uparrow}}$, seed)

Scope closed: Definition 746.1, Definition 746.2, Definition 746.4, and Definition 746.5 bind the finite audit packet for checked MetaCIC typing inference and bounded normal comparison.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICDecidableAuditPacket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close unrestricted beta-conversion decidability, standalone closedness decidability, global normalization, quotient equality of terms, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, checker exactness, same-term exactness, typing soundness and completeness routing, bounded-normal boundary routing, transport, route replay, provenance, and non-escape for the same finite packet.

Chapter 747

A Concrete Naming Certificate for $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$

The $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$ packet is the concrete NameCert surface drawn from the decidability audit map of chapter 9379. It packages the checked typing-inference rows, the same-term decision rows, and the bounded normal-equality rows as one finite consumer-facing seal. The packet sits beside the normal-form carriers of chapter 562 and chapter 683: those chapters expose terminal normal-form readback, while this chapter exposes the decidable checker rows that may be safely consumed without claiming a global closed-term conversion decider.

Definition 747.1 (Metacic decidable checker seal carrier)**.** *A* $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, T, S, N, H, C, P, L)$$

with the following displayed rows:

1. *E is the structural term-equality row for encoded MetaCIC syntax;*
2. *T is the typing-inference row, including the checked soundness and completeness surfaces for the encoded* `HasType` *relation;*
3. *S is the same-term checker row for the displayed syntax layer;*
4. *N is the bounded normal-equality row, including the finished-normalisation consequences recorded by the audit map;*
5. *H records componentwise* hsame *transport for E, T, S, N, route data, provenance, and local naming;*
6. *C records the displayed* Cont *routes by which a consumer reads structural equality, typing inference, same-term checking, and bounded normal equality;*
7. *P is the* Pkg *provenance over* BHist, BMark, hsame, Cont, Pkg, NameCert, *and the MetaCIC decidability audit rows named in chapter 9379;*
8. *L is the local* $\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}$ *row.*

The carrier contains no row for unrestricted closedness decidability, no row for global closed-term beta-conversion decidability, and no row turning bounded normal equality into an unbounded conversion theorem.

Theorem 747.2 (Metacic decidable checker seal NameCert obligations)**.** *For every accepted* $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$ *carrier*

$$D = (E, T, S, N, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}$ *surface consists exactly of structural term-equality admission through* E*, typing-inference soundness and completeness through* T*, same-term checking through* S*, bounded normal-equality soundness through* N*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 747.1. Its checker coordinates are exactly E, T, S, N, so the visible decision surface is structural equality, typing inference, same-term checking, and bounded normal equality.

The transport row H preserves those four coordinates componentwise. The continuation row C supplies the only routes by which a downstream consumer may read them, and those routes end inside the same packet.

Package provenance through P records the audit-map dependency, while the local row L names the packet as a $\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}$ surface. Since the carrier has no further checker coordinate, the NameCert obligations are exactly the displayed finite rows. □

Theorem 747.3 (Metacic decidable checker seal non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$ *carrier factors through*

$$E, T, S, N, H, C, P, L.$$

The consumer may read structural term equality, typing-inference soundness and completeness, same-term checking, bounded normal-equality soundness, componentwise transports, displayed continuation routes, provenance, and the local naming row. It cannot read unrestricted closedness decidability, global closed-term beta-conversion decidability, a confluence theorem, a subject-reduction theorem, or an external normalizer from the checker seal.

Proof. Begin with the displayed continuation row C. It names consumer routes only for E, T, S, N and has no endpoint for a closedness decider, a global conversion decider, confluence, subject reduction, or an external normalizer.

Next use componentwise transport through H. Transport moves the same four checker rows and cannot manufacture a theorem coordinate absent from the carrier.

Finally package the result through P and L. Provenance and local naming certify the finite checker surface described by chapter 9379, while the missing coordinates rule out escape beyond E, T, S, N, H, C, P, L. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}$ packet is generated from structural term equality, typing-inference soundness and completeness, same-term checking, bounded normal-equality soundness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 747.1, Definition 747.2, and Definition 747.3 bind the decidable MetaCIC checker rows as a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicDecidableCheckerSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close unrestricted closedness decidability, global closed-term beta-conversion decidability, confluence, subject reduction, Lean-side verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the checker-seal carrier, route exhaustion, and checked non-escape for the structural equality, typing-inference, same-term, and bounded normal-equality rows.

Chapter 748

A Concrete Naming Certificate for $\mathsf{BoundedBetaComparison}^{\uparrow}$

The $\mathsf{BoundedBetaComparison}^{\uparrow}$ packet turns the normal-equality surface of chapter 9379 into concrete finite BEDC data. It sits next to the normal-form seal of chapter 740 and the confluence audit packet of chapter 743: all three chapters name bounded or conditional MetaCIC evidence without promoting it to an unrestricted closed-term theorem.

Definition 748.1 (Bounded beta-comparison carrier)**.** *A* $\mathsf{BoundedBetaComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (f, t, u, n_t, n_u, e, w_t, w_u, H, C, P, N)$$

with the following displayed rows:

1. f *is the fuel row used by the bounded normalizer;*
2. t, u *are the closed MetaCIC term rows being compared;*
3. n_t, n_u *are the bounded-normal output rows computed from* t *and* u *at fuel* f*;*
4. e *is the structural* Term.eq *comparison row applied to* n_t *and* n_u*;*
5. w_t, w_u *are the finished-normalisation witness rows when the fuel is known to finish on* t *and* u*;*
6. H *records componentwise* hsame *transports for the fuel, term, bounded-normal, equality, witness, provenance, and local naming rows;*
7. C *records the displayed* Cont *route by which a consumer reads the bounded comparison and any finished-normalisation boundary data;*
8. P *is the* Pkg *provenance over* `NormalEqDecide`*,* `Term.eq`*, closed MetaCIC syntax,* BHist*,* hsame*,* Cont*, and sibling* NameCert *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BoundedBetaComparison}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $f, t, u, n_t, n_u, e, w_t, w_u, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.BoundedBetaComparisonUp.TasteGate.BoundedBetaComparisonTasteGate single carrier alignment`

Theorem 748.2 (Bounded beta-comparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BoundedBetaComparison}^{\uparrow}}$ *surface for Definition 748.1 consists of carrier admission, bounded-normal output readback, structural* Term.eq *readback, componentwise* hsame *transport, displayed* Cont *route replay, provenance, and non-escape from any fuel-erased beta-conversion theorem.*

Proof. Carrier admission is projection from the finite tuple $B = (f, t, u, n_t, n_u, e, w_t, w_u, H, C, P, N)$. The fuel, closed term rows, bounded-normal outputs, equality comparison, optional witness rows, transport, route, provenance, and local naming row are all displayed coordinates.

The bounded-normal readback and Term.eq readback inspect only f, t, u, n_t, n_u, e. Thus a successful comparison is a row about the displayed bounded-normal outputs, not a row about unrestricted closed-term conversion.

The structural rows H, C, P, N preserve the same finite packet under transport, consumer routing, provenance, and local naming. Since none of these rows contains a fuel-erased normalisation theorem, the local NameCert exports no stronger beta-conversion surface. □

Theorem 748.3 (Bounded beta-comparison finished soundness boundary)**.** *For an accepted* BoundedBetaComparis *packet, a true bounded comparison identifies only the displayed bounded-normal outputs n_t and n_u. When the finished-normalisation witness rows w_t and w_u are present, the same packet also exposes the finished-normal-form boundary for those named outputs. Without those witness rows, the packet exports no conclusion that t and u have fuel-independent normal forms.*

Proof. Read the comparison row e through the carrier. The row is the Term.eq comparison of n_t and n_u, so its positive content is equality of the two bounded-normal outputs displayed at fuel f.

If w_t and w_u are present, they supply the extra endpoint data consumed by the finished-normalisation soundness theorems named in chapter 9379. The boundary is therefore tied to the particular fuel, terms, bounded outputs, and witness rows carried by the packet.

If either witness row is absent, the carrier still records the bounded comparison row, but it has no coordinate witnessing that the bounded outputs are final normal forms. The consumer cannot erase the fuel or infer a global normalisation endpoint from the packet alone. □

Theorem 748.4 (Bounded beta-comparison non-escape)**.** *Every consumer of an accepted* BoundedBetaComparison *packet is exhausted by the carrier of Definition 748.1, the local NameCert obligations of Definition 748.2, and the finished-normal-form boundary of Definition 748.3. The packet exports no global closed-term beta-conversion decider, no fuel-erased normalisation theorem, no quotient equality, and no host conversion oracle.*

Proof. Begin with the displayed carrier rows. They provide exactly the fuel, two closed terms, two bounded-normal outputs, one structural equality comparison, optional finished-normalisation witnesses, and the structural BEDC rows.

Apply the NameCert obligations. They route every consumer through the same bounded-normal and Term.eq rows before any comparison result is read.

Finally apply the finished soundness boundary. The only strengthening available from a true comparison comes from the explicit witness rows w_t, w_u when they are present. Since the packet has no coordinate for fuel erasure, quotient equality, host conversion, or arbitrary closed-term normalisation, none of those readings can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedBetaComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedBetaComparison}^{\uparrow}$ packet is generated from finite BHist rows for source and target terms, bounded beta fuel, local beta-step evidence, comparison result, transports, continuations, provenance, and local naming. The comparison is read by replaying the bounded beta route through Cont, so the packet exposes only the steps and fuel that are actually displayed. Componentwise hsame transports source, target, fuel, and step rows without erasing the bound, while Pkg records the beta-comparison and closed-term dependencies. The result is a finite comparison certificate rather than unrestricted conversion or normalization.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedBetaComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the bounded beta comparison carrier, bounded route replay, fuel preservation, comparison readback, and non-export boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{BoundedBetaComparison}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: human
Not claimed: It does not claim fuel erasure, quotient equality, host conversion, arbitrary closed-term normalization, confluence, or global beta completeness.
Upgrade path: Obligation closure requires Lean encodings of the bounded comparison carrier, fuel-indexed route, comparison stability, and non-escape theorem.

Chapter 749

A Concrete Naming Certificate for MetaCICDecidableBoundary$^{\uparrow}$

The MetaCICDecidableBoundary$^{\uparrow}$ packet realizes the decidability audit map of chapter 9379 as a finite NameCert object. It sits next to the normal-form exclusion packet of chapter 732, the closed-normal confluence seal of chapter 741, and the subject-reduction obstruction packet of chapter 731. The packet records the checked finite procedures and their refusal boundaries without promoting bounded normal comparison into global beta-conversion decidability.

Definition 749.1 (MetaCIC decidable boundary carrier)**.** *A* MetaCICDecidableBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, S, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* inferTypeCtx *checker row for the encoded* HasType *relation;*
2. *S is the* checkSameTerm *structural-equality row used by the checker;*
3. *B is the bounded normal-equality row for* normalizeBounded *and* decideClosedBetaConv*;*
4. *F is the finished-normal-form side-condition row consumed by the bounded comparison soundness theorems;*
5. *R records the refusal rows for a standalone* ClosedAt *decision theorem and for global closed beta-conversion decidability;*
6. *H records componentwise* hsame *transport across T, S, B, F, R, P, N;*
7. *C records the displayed* Cont *routes by which consumers replay checker, comparison, refusal, and side-condition reads;*
8. *P is the* Pkg *provenance over the decidability audit map, the normal-form exclusion sibling, the closed-normal confluence seal sibling, and the subject-reduction obstruction sibling;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}^{\uparrow}}$ *row.*

The carrier has no row for a global ClosedAt *decider, no unrestricted beta-conversion oracle, no fuel-erased normalisation theorem, no quotient of represented terms, and no host evaluator.*

Theorem 749.2 (MetaCIC decidable boundary NameCert obligations)**.** *Every accepted* MetaCICDecidableBoundar *packet binds exactly the finite obligation surface displayed in Definition 749.1: carrier admission*

for $D = (T, S, B, F, R, H, C, P, N)$, checker visibility for T, structural equality visibility for S, bounded normal comparison visibility for B, finished-normal side-condition visibility for F, refusal visibility for R, componentwise hsame *transport through H, finite* Cont *replay through C, sibling provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryTasteGate single carrier alignment`

Proof. Project the carrier. The positive checker rows are T, S, B, F: the typing checker reads T together with the structural equality row S, and the conversion checker reads the bounded normal comparison row B together with the finished-normal side-condition row F.

The refusal row R is part of the same finite packet. It records that the carrier does not contain a separate ClosedAt decision theorem and does not contain an unrestricted beta-conversion decider for arbitrary closed terms.

The structural rows H, C, P, N only transport, replay, package, and name those displayed reads. Since no further coordinate is present, the local NameCert surface is exhausted by the listed obligations. □

Theorem 749.3 (MetaCIC decidable boundary bounded comparison non-escape)**.** *Any consumer of an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet may read bounded normal equality only through B and F. The read can expose equality or beta-convertibility of the bounded normalizer outputs under the finished-normal-form side conditions, but it cannot export a global closed beta-conversion decision procedure, a fuel-erased normalisation theorem, or a quotient equality over represented terms.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundary bounded comparison nonescape`

Proof. Begin with the bounded comparison coordinates. The carrier separates B, the bounded normalizer comparison row, from F, the row that supplies the finished-normal-form side conditions.

The refusal coordinate R records the missing promotion from bounded comparison to global closed beta conversion. Thus a consumer cannot skip F or erase the fuel boundary by rereading B as an unrestricted oracle.

Finally use the structural rows. The hsame and Cont routes in H, C replay the same finite comparison route, while P, N only record provenance and local naming. None of these rows adds a quotient equality, host evaluator, or global normalisation endpoint. □

Theorem 749.4 (MetaCIC decidable boundary refusal exactness)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet $D = (T, S, B, F, R, H, C, P, N)$, the refusal row R is exact: it records precisely the absence of a standalone* ClosedAt *decider and an unrestricted closed beta-conversion oracle, while the finite rows T, S, B, F, H, C, P, N remain available for checker reads, structural equality reads, bounded comparison, finished-normal side conditions, componentwise transport, displayed replay, provenance, and local naming. The refusal row exports no fuel-erased normalization theorem and no quotient equality over represented terms.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryRefusalExactness`

Proof. Project the refusal coordinate R from the carrier of Definition 749.1. The carrier text gives R exactly the two refusal duties: no standalone ClosedAt decision theorem and no global closed beta-conversion decider.

Apply Definition 749.2. It keeps R inside the same finite NameCert surface as the positive rows T, S, B, F, H, C, P, N, so the refusal does not erase the bounded checker or comparison services already displayed by the carrier.

Now use Definition 749.3. The bounded comparison row B can be read only with the finished-normal side-condition row F, and that read cannot be exported as a global conversion procedure.

Therefore the boundary is exact. Consumers may use the finite displayed rows, but they cannot obtain fuel-erased normalization or quotient equality from R or from any structural replay of the packet. □

Theorem 749.5 (MetaCIC decidable boundary sibling provenance)**.** *The provenance row P places* MetaCICDecidableBoundary$^{\uparrow}$ *between the audit map of chapter 9379, the normal false exclusion packet of chapter 732, the closed-normal confluence seal of chapter 741, and the subject-reduction obstruction packet of chapter 731. Agreement on H, C, P, N preserves only finite route and naming data among those chapters; it does not identify their theorem rows or discharge their missing global endpoints.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundary sibling provenance`

Proof. Read P as package provenance. It cites the audit map and the three sibling concrete chapters as anchors for typing, normal-form, confluence, and subject-reduction boundaries.

The shared rows H, C, P, N give only componentwise transport, continuation replay, package provenance, and local naming. They do not collapse the sibling packets into one theorem because each sibling keeps its own theorem and refusal rows.

Therefore the sibling route is a finite citation and transport surface. It preserves the decidability boundary without turning normal false exclusion, closed-normal confluence, or subject-reduction obstruction into a global conversion or consistency theorem. □

Theorem 749.6 (MetaCIC decidable boundary window totality)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet $D = (T, S, B, F, R, H, C, P, N)$, each bounded comparison request read through B has a finite inspected window supplied by the displayed checker and structural-equality rows T, S together with the finished-normal side-condition row F. The packet supplies no host decidability theorem, no global MetaCIC normalization theorem, and no unrestricted beta-conversion oracle.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryWindowTotality`

Proof. Project the carrier. The bounded comparison row is B, and the only rows that can feed it are the checker row T, the structural-equality row S, and the finished-normal side-condition row F.

Apply Definition 749.2. It lists checker visibility, structural equality visibility, bounded normal comparison visibility, and finished-normal side-condition visibility as finite rows of the same local packet.

The bounded comparison non-escape theorem then prevents rereading B as a global decision procedure. Hence each comparison has exactly the finite inspected window recorded by the displayed rows, with no host-level decision or normalization endpoint. □

Theorem 749.7 (MetaCIC decidable boundary candidate normal-form reads)**.** *Every candidate-normal-form read available to an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet factors through the checker row T, structural equality row S, bounded normal row B, finished-normal side-condition row F, and refusal row R. The read may be replayed through H, C, P, N, but it cannot export a standalone* ClosedAt *decider, global beta-conversion decision, quotient equality over represented terms, or host evaluator.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundary candidate normal form reads`

Proof. Start from the window totality theorem Definition 749.6. It says that each bounded comparison request has a finite inspected window supplied by T, S, F together with the bounded comparison row B.

Apply the NameCert obligation theorem Definition 749.2. Candidate-normal-form reads are therefore carrier reads of the same packet, so their structural replay is limited to H, C, P, N.

Now use bounded comparison non-escape and refusal exactness, Definition 749.3 and Definition 749.4. The row B remains bounded by F, while R records the absence of standalone ClosedAt decidability and unrestricted closed beta conversion.

Thus a candidate-normal-form read may expose the finite checker, structural equality, bounded normal, side-condition, refusal, and replay rows, but it cannot become a global conversion decision, quotient equality, or host evaluation procedure. □

Theorem 749.8 (MetaCIC decidable boundary bridge status)**.** *Any bridge consumer of an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet receives only the finite checker row T, structural equality row S, bounded normal comparison row B, finished-normal side-condition row F, refusal row R, componentwise* hsame *transport H, displayed* Cont *replay C, package provenance P, and local naming N. The bridge surface cannot extract global* ClosedAt *decidability, unrestricted beta-conversion, quotient equality over represented terms, or host evaluation from the bounded comparison packet.*

Proof. Start with the NameCert obligations of Definition 749.2. They expose exactly the finite checker, structural equality, bounded comparison, side-condition, refusal, transport, replay, provenance, and local naming rows of the packet.

Use bounded-comparison nonescape, Definition 749.3. It keeps every comparison read tied to B together with the finished-normal side-condition row F, so a bridge consumer cannot erase the bounded window and read B as a global conversion oracle.

Apply refusal exactness and candidate-normal-form reads, Definition 749.4 and Definition 749.7. The refusal row R records the absent global endpoints, while candidate-normal inspection remains inside the same finite checker and comparison route.

Finally, the structural rows H, C, P, N only transport, replay, source, and name that displayed route. Since no carrier coordinate supplies global ClosedAt decidability, unrestricted beta-conversion, quotient equality, or host evaluation, no bridge consumer can recover those endpoints from the packet. □

Theorem 749.9 (MetaCIC decidable boundary classifier stability)**.** *Let $D = (T, S, B, F, R, H, C, P, N)$ be an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet. Classifier reads of T, S, B, F, R are stable only along the displayed componentwise* hsame *row H and the finite* Cont *replay row C. This stability transports the bounded checker surface but does not identify sibling theorem rows, decide arbitrary closed terms, or erase the fuel boundary.*

Proof. Read H and C from the carrier. They are the only transport and replay rows attached to the checker, structural equality, bounded comparison, side-condition, and refusal coordinates.

Use Definition 749.2. It makes the hsame and Cont routes componentwise and finite, so they preserve only the displayed reads of T, S, B, F, R.

Finally apply Definition 749.5. The sibling route is provenance and naming data rather than theorem-row identification. Therefore classifier stability remains local to H, C and cannot become host decidability or global normalization. □

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryClassifierStability`

Theorem 749.10 (MetaCIC decidable boundary ledger exactness)**.** *For an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, the local ledger exposes exactly the recorded bounded-comparison rows* B, F *together with their checker, refusal, transport, replay, provenance, and naming context* T, S, R, H, C, P, N*. No consumer can read an unrecorded bounded-comparison row, a quotient of represented terms, a host evaluator, or a global beta-conversion decider.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryLedgerExactness`

Proof. Begin with the carrier and its NameCert obligations. They enumerate B and F as the bounded-comparison and finished-normal side-condition rows and place T, S, R, H, C, P, N around them as context.

Apply Definition 749.3. It rules out promotion from those rows to unrestricted conversion, fuel-erased normalisation, or quotient equality over represented terms.

The provenance theorem accounts for the remaining sibling references. They explain where the finite boundary sits, but they add no extra comparison rows. Thus the ledger is exact: the displayed bounded-comparison rows are all that consumers may inspect. □

Theorem 749.11 (MetaCIC decidable boundary fuel obligation)**.** *For an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every bounded-fuel read used by the decidable-boundary surface is a finite read of* F *together with the checker, structural equality, bounded comparison, refusal, transport, replay, provenance, and local naming context* T, S, B, R, H, C, P, N*. The read may bound a displayed normal comparison, but it does not export standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, fuel-erased normalization, or host evaluation.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryFuelObligation`

Proof. Project the carrier and the ledger exactness theorem Definition 749.10. The only side-condition coordinate for bounded normal comparison is F, and the ledger places it beside T, S, B, R, H, C, P, N.

Use bounded comparison non-escape from Definition 749.3. A comparison through B remains tied to the displayed side-condition row F, so the fuel boundary is not erased.

Finally apply refusal exactness from Definition 749.4. The refusal row R keeps standalone ClosedAt decidability, global beta-conversion, quotient equality, fuel-erased normalization, and host evaluation outside the packet. □

Theorem 749.12 (MetaCIC decidable boundary candidate obligation)**.** *For an accepted* MetaCICDecidableBoundar *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every candidate-normal read factors through the finite checker row* T*, structural equality row* S*, bounded normal comparison row* B*, finished-normal side-condition row* F*, refusal row* R*, and structural rows* H, C, P, N*. Candidate-normality is therefore a displayed packet obligation, not a route to standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, fuel-erased normalization, or host evaluation.*

Proof. Start from candidate normal-form reads in Definition 749.7. They already expose $T, S, B, F, R, H, C, P, N$ as the finite route for candidate-normal data.

Apply ledger exactness from Definition 749.10. Since the ledger has no unrecorded comparison row or host evaluator, the candidate read cannot move outside the displayed packet.

The refusal row is then read through Definition 749.4. It blocks standalone ClosedAt decidability, global beta-conversion, quotient equality, and fuel-erased normalization while leaving the finite candidate-normal route intact. □

Theorem 749.13 (MetaCIC decidable boundary replay obligation)**.** *For an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every replay of the decidable-boundary surface is finite and componentwise along* H *and* C *after the rows* T, S, B, F, R *have been exposed. Replay preserves the checker, structural equality, bounded comparison, side-condition, and refusal ledger, but it does not create standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, fuel-erased normalization, or host evaluation.*

Proof. Read classifier stability from Definition 749.9. It identifies H and C as the transport and replay rows for the displayed checker, equality, comparison, side-condition, and refusal coordinates.

Use ledger exactness from Definition 749.10. Replay can preserve only recorded rows, so it cannot introduce an unrecorded comparison row, quotient of represented terms, host evaluator, or global beta-conversion decider.

Finally apply ClosedAt refusal routing from Definition 749.24. Any standalone decision request is sent to R, while H, C, P, N replay and name only the finite boundary. □

Theorem 749.14 (MetaCIC decidable boundary substitution window)**.** *For every accepted* MetaCICDecidableBoun *packet* $D = (T, S, B, F, R, H, C, P, N)$*, any substitution-sensitive bounded comparison read factors through the checker row* T*, structural equality row* S*, bounded normal row* B*, finished-normal side-condition row* F*, refusal row* R*, and the transport and replay rows* H, C, P, N*. The read may compare the displayed bounded normal outputs under the recorded side conditions, but it cannot export standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, or a host evaluator.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundarySubstitutionWindow`

Proof. Start with the NameCert obligations of Definition 749.2. They expose the checker, structural equality, bounded normal comparison, finished-normal side condition, refusal, transport, replay, provenance, and naming rows as one finite packet.

Apply bounded comparison non-escape from Definition 749.3. A comparison read through B must keep the finished-normal side-condition row F visible, so substitution-sensitive use of the comparison cannot erase the fuel and side-condition boundary.

Now use refusal exactness and classifier stability, Definition 749.4 and Definition 749.9. The row R records the absence of global endpoints, while H, C transport and replay only the displayed checker and comparison rows.

Finally apply ledger exactness from Definition 749.10. Since the ledger contains no unrecorded bounded-comparison row, quotient of represented terms, host evaluator, or global beta-conversion decider, the substitution window remains a finite bounded comparison surface. □

Theorem 749.15 (MetaCIC decidable boundary refusal substitution stability)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, substitution-sensitive reads that ask for standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, fuel-erased normalization, or host evaluator behavior are still routed to the refusal row* R*. The transport, replay, provenance, and naming rows* H, C, P, N *preserve the finite substitution window, but they do not turn the refused requests into accepted endpoints.*

Proof. Begin with the substitution-window theorem Definition 749.14. It keeps every substitution-sensitive bounded comparison inside $T, S, B, F, R, H, C, P, N$.

Use refusal exactness from Definition 749.4. The row R is precisely the displayed refusal boundary for standalone ClosedAt decisions, global beta-conversion, quotient equality, fuel-erased normalization, and host evaluation.

Finally apply ledger exactness from Definition 749.10. Since the ledger contains no separate endpoint for those requests, transport and replay through H, C, P, N can preserve only the finite boundary and must route the requests back to R. □

Theorem 749.16 (MetaCIC decidable boundary candidate-normal substitution scope)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, candidate-normal comparison under substitution consumes only the checker row* T*, structural equality row* S*, bounded normal comparison row* B*, finished-normal side-condition row* F*, refusal row* R*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. No candidate-mediated read may add a hidden evaluator, quotient equality row, global beta-conversion decider, standalone* ClosedAt *decider, or fuel-free normalization endpoint.*

Proof. Start from candidate-normal-form reads in Definition 749.7. Candidate inspection is already tied to T, S, B, F and cannot be detached from the displayed checker and side-condition rows.

Apply the substitution-window theorem Definition 749.14. It shows that substitution-sensitive comparison still passes through R, H, C, P, N in addition to the checker, equality, comparison, and finished-normal rows.

Use refusal exactness and ledger exactness, Definition 749.4 and Definition 749.10. The refusal row accounts for the missing global endpoints, and the ledger has no unrecorded row beyond $T, S, B, F, R, H, C, P, N$. □

Theorem 749.17 (MetaCIC decidable boundary candidate-substitution exhaustion)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every substitution-sensitive candidate comparison read must expose the checker row* T*, structural equality row* S*, bounded normal row* B*, finished-normal row* F*, refusal row* R*, and replay and provenance rows* H, C, P, N*. No consumer may infer standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, or host evaluator behavior from such a candidate-mediated read.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryCandidateSubstitutionExhaustion`

Proof. Begin with the substitution-window theorem Definition 749.14. It already routes every substitution-sensitive bounded comparison through $T, S, B, F, R, H, C, P, N$.

Use candidate-normal-form reads and bounded comparison nonescape, Definition 749.7 and Definition 749.3. Candidate inspection therefore remains tied to the displayed checker, structural equality, bounded normal comparison, and finished-normal side-condition rows.

Now apply refusal exactness and classifier stability, Definition 749.4 and Definition 749.9. The refusal row records the missing global endpoints, while H, C transport and replay only the displayed candidate and comparison rows.

Finally invoke ledger exactness from Definition 749.10. The ledger contains no unrecorded comparison row, quotient of represented terms, host evaluator, standalone ClosedAt decider, or global beta-conversion decider, so the candidate-substitution surface is exhausted by $T, S, B, F, R, H, C, P, N$. □

Theorem 749.18 (MetaCIC decidable boundary obligation-closure package)**.** *For an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, the finite checker row* T*, structural equality row* S*, bounded normal comparison row* B*, finished-normal side-condition row* F*, refusal row* R*, and structural rows* H, C, P, N *form exactly the obligation-closure surface of the chapter. The package records checker visibility, structural-equality visibility, bounded-comparison*

visibility, finished-normal side conditions, refusal exactness, sibling provenance, classifier stability, and ledger exactness, while it still exports no standalone ClosedAt *decidability, global beta-conversion decidability, fuel-erased normalization, quotient equality, or host evaluation.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundary obligation closure package`

Proof. Start with Definition 749.2. It lists the carrier admission, checker row, structural equality row, bounded comparison row, finished-normal side condition, refusal row, transport, replay, provenance, and local naming rows of D.

Use Definition 749.3 and Definition 749.4. Together they keep B, F as bounded comparison data and keep R as the exact refusal of standalone ClosedAt decidability and global beta conversion.

Apply Definition 749.5 and Definition 749.10. Provenance contributes only the named sibling route, and the ledger supplies no unrecorded bounded-comparison row, quotient of represented terms, host evaluator, or global beta-conversion decider.

The classifier-stability theorem Definition 749.9 transports the same displayed rows through H, C without adding a new endpoint. Hence $T, S, B, F, R, H, C, P, N$ are exactly the obligation-closure package surface. □

Theorem 749.19 (MetaCIC decidable boundary conditional obligation)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, an accepted bounded comparison read is valid only under the displayed checker row* T*, structural equality row* S*, bounded normal row* B*, finished-normal side-condition row* F*, and refusal row* R*, with transport, replay, provenance, and naming rows* H, C, P, N *still visible. The conditional route explicitly refuses standalone* ClosedAt *decidability, global beta-conversion, quotient equality over represented terms, and host evaluation.*

Proof. Start with bounded-comparison nonescape and window totality, Definition 749.3 and Definition 749.6. They make every accepted bounded comparison expose T, S, B, F rather than a detached comparison oracle.

Use candidate normal-form reads from Definition 749.7. Candidate inspection remains inside the same checker, structural equality, bounded normal, and finished-normal side-condition rows.

Apply refusal exactness from Definition 749.4. The row R records the missing standalone ClosedAt decision and unrestricted beta-conversion endpoint, while leaving the bounded comparison route available only under its displayed conditions.

Finally invoke ledger exactness and the obligation-closure package, Definition 749.10 and Definition 749.18. The ledger has no quotient equality over represented terms and no host evaluator, so the valid comparison surface is exactly $T, S, B, F, R, H, C, P, N$. □

Theorem 749.20 (MetaCIC decidable boundary obligation scope lock)**.** *For every accepted* MetaCICDecidableBour *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every obligation-closure consumer must keep the checker row* T*, structural equality row* S*, bounded normal comparison row* B*, finished-normal side-condition row* F*, refusal row* R*, componentwise transport row* H*, displayed replay row* C*, provenance row* P*, and local name row* N *together. In particular, bounded normal comparison cannot be read as global beta-conversion decidability, standalone* ClosedAt *decidability, quotient equality over represented terms, fuel-erased normalization, or host evaluation.*

Proof. Start with bounded-comparison nonescape, Definition 749.3. It permits a read of B only with the finished-normal side-condition row F, and it refuses the jump to global conversion.

Apply refusal exactness, Definition 749.4. The row R records exactly the missing standalone ClosedAt decision and unrestricted beta-conversion decision, while the positive checker and comparison rows remain finite.

Use classifier stability and ledger exactness, Definition 749.9 and Definition 749.10. Transport and replay preserve the displayed rows, and the ledger contains no unrecorded comparison row, quotient of represented terms, host evaluator, or global beta-conversion decider.

Finally invoke the obligation-closure package theorem Definition 749.18. It identifies $T, S, B, F, R, H, C, P, N$ as the exact obligation surface. Hence any obligation-closure consumer must keep the rows together, and the bounded comparison row cannot be exported as a global decidability principle. □

Theorem 749.21 (MetaCIC decidable boundary ClosedAt refusal scope)**.** *For an accepted* MetaCICDecidableBound *packet $D = (T, S, B, F, R, H, C, P, N)$, the refusal row R records only the absence of standalone* ClosedAt *decisions and global beta-conversion decisions. It cannot erase the bounded side conditions attached to B, F, and it cannot turn the transported checker surface T, S, H, C, P, N into a host evaluator or quotient equality.*

Proof. Start with refusal exactness from Definition 749.4. That theorem identifies R as the exact refusal of standalone ClosedAt decidability and unrestricted closed beta conversion.

Now use bounded comparison non-escape and candidate-normal-form reads, Definition 749.3 and Definition 749.7. They keep every comparison read tied to B, F and keep candidate-normal-form inspection inside the displayed finite packet.

Finally apply classifier stability and ledger exactness, Definition 749.9 and Definition 749.10. Transport and ledger replay preserve the refusal boundary but add no erased side-condition row, quotient equality, or host evaluator. □

Theorem 749.22 (MetaCIC decidable boundary bounded normal comparison scope)**.** *Every bounded normal comparison read of an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet factors through the finite route $T, S, B, F, R, H, C, P, N$. The read may compare the displayed bounded normal outputs under the finished-normal side conditions, but it cannot become quotient equality over represented terms, standalone* ClosedAt *decidability, global beta-conversion decidability, fuel-erased normalization, or a host evaluator.*

Proof. Begin with the bounded comparison non-escape theorem Definition 749.3. It makes B, F the only route to bounded normal equality and keeps the finished-normal side condition visible.

Use the substitution-window theorem Definition 749.14. It routes substitution-sensitive comparison through $T, S, B, F, R, H, C, P, N$, so the comparison cannot be detached from the checker and refusal rows.

Finally apply candidate-normal-form reads and ledger exactness, Definition 749.7 and Definition 749.10. The ledger contains no unrecorded comparison row, quotient of represented terms, host evaluator, unrestricted beta-conversion decider, or fuel-free normalization endpoint. □

Theorem 749.23 (MetaCIC decidable boundary finite checker indexing)**.** *Every finite checker output exposed by an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet remains indexed by the displayed fuel row, closedness evidence, and normal endpoint rows carried by $T, S, B, F, R, H, C, P, N$. Consequently the output can be replayed only as bounded checker evidence inside the packet; it cannot be promoted to a global closed-term conversion decision.*

Proof. Start with fuel-window replay, Definition 749.25. It keeps every bounded comparison read tied to the fuel and finished-normal side-condition row F, so the checker output is not fuel-free.

Next use the conversion nonescape package, Definition 749.26. Conversion-facing consumers see only B, F, R, H, C, P, N, with R recording the missing global endpoints and with structural replay confined to the displayed route.

Finally apply the normal-form decision envelope, Definition 749.27. The positive envelope contains the finite checker and bounded-comparison rows together with their finished-normal side conditions, while the refusal row keeps standalone ClosedAt and global beta-conversion decisions outside the packet. □

Theorem 749.24 (MetaCIC decidable boundary ClosedAt refusal route)**.** *For an accepted* MetaCICDecidableBound *packet $D = (T, S, B, F, R, H, C, P, N)$, any consumer asking for a standalone* ClosedAt *decision route is sent exactly to the refusal row R. The positive rows T, S, B, F remain available for typing, structural equality, bounded normal comparison, and finished-normal side conditions, while H, C, P, N only transport, replay, source, and name that finite boundary.*

Proof. Project the carrier of Definition 749.1. The only coordinate assigned to a standalone ClosedAt decision theorem is the refusal coordinate R.

Use refusal exactness from Definition 749.4. It records the absence of standalone ClosedAt decidability without deleting the positive checker and comparison rows T, S, B, F.

Finally apply classifier stability and ledger exactness, Definition 749.9 and Definition 749.10. The rows H, C, P, N can replay only the displayed boundary, so a ClosedAt-facing consumer cannot find another route outside R. □

Theorem 749.25 (MetaCIC decidable boundary fuel-window replay)**.** *Let $D = (T, S, B, F, R, H, C, P, N)$ be an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet. Every bounded comparison replay that reads B must also expose the fuel and finished-normal side-condition row F. The replay may use T, S, H, C, P, N to reach the finite comparison window, but it cannot erase F or turn B into fuel-free normalization.*

Lean4 ✓ `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryFuelWindowReplay`

Proof. Start with the bounded comparison non-escape theorem Definition 749.3. It permits access to bounded normal equality only through the pair B, F.

The window totality theorem Definition 749.6 supplies the finite inspected window through T, S together with F. Thus the replayed comparison keeps the side-condition row visible at the same point where B is read.

The remaining structural rows H, C, P, N preserve the route but add no fuel-erased endpoint. Therefore bounded replay cannot remove F and cannot export global normalization. □

Theorem 749.26 (MetaCIC decidable boundary conversion nonescape package)**.** *Every conversion-facing consumer of an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet factors through the finite package B, F, R, H, C, P, N. The rows B, F supply bounded comparison under finished-normal side conditions, R records the missing global endpoints, and H, C, P, N provide only componentwise transport, displayed replay, provenance, and local naming.*

Proof. Begin with candidate normal-form reads from Definition 749.7. Any conversion-facing request reaches the bounded normal row B, the side-condition row F, and the refusal row R before it can be replayed structurally.

Apply fuel-window replay and ClosedAt refusal routing, Definition 749.25 and Definition 749.24. They keep the bounded comparison tied to F and send standalone decision requests to R.

The structural part of the package is then exactly H, C, P, N. By ledger exactness, these rows add no quotient equality, host evaluator, unrestricted beta-conversion decider, or fuel-free normalization theorem. □

Theorem 749.27 (MetaCIC decidable boundary normal-form decision envelope)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every normal-form decision read available from* D *factors through the finite route* $T, S, B, F, R, H, C, P, N$*. The positive envelope contains only* inferTypeCtx *reads from* T*,* checkSameTerm *reads from* S*, bounded normal comparison reads from* B*, and finished-normal side conditions from* F*. The refusal row* R *still records the absence of standalone* ClosedAt *decidability, global beta-conversion decidability, quotient equality over represented terms, fuel-erased normalization, and host evaluation.*

Proof. Start with window totality from Definition 749.6. Any normal-form decision read has a finite inspected window, and that window is supplied by the checker row T, the structural-equality row S, the bounded comparison row B, and the finished-normal side-condition row F.

Apply bounded normal comparison scope from Definition 749.22. It keeps comparison inside the displayed route $T, S, B, F, R, H, C, P, N$ and prevents the bounded comparison from becoming quotient equality, unrestricted conversion, fuel-erased normalization, or host evaluation.

Use the ClosedAt refusal route of Definition 749.24. A standalone ClosedAt request is sent to R, while the positive checker and comparison rows remain available only as the finite rows already displayed by the packet.

Finally apply the conversion nonescape package Definition 749.26. It repacks the conversion-facing surface through B, F, R, H, C, P, N, so the normal-form decision envelope is exactly the positive checker, structural equality, bounded comparison, and finished-normal side-condition surface together with the refusal boundary in R. □

Theorem 749.28 (MetaCIC decidable boundary conversion refusal trichotomy)**.** *For every accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet* $D = (T, S, B, F, R, H, C, P, N)$*, every conversion-facing request is classified by exactly one displayed route: bounded comparison through* B *together with the finished-normal side condition* F*, finite refusal through* R *when that side condition is absent, or structural replay through* H, C, P, N *when the request is only transporting an already displayed boundary. None of the three routes exports global beta-conversion, fuel-erased normalization, quotient equality over represented terms, or host evaluation.*

Proof. Begin with the normal-form decision envelope Definition 749.27. A positive conversion-facing read must expose the finite checker rows T, S, the bounded comparison row B, and the finished-normal side-condition row F before any comparison result is available.

If F is absent, apply the ClosedAt refusal route Definition 749.24. The request cannot be promoted to a standalone conversion decision, so it is routed to the finite refusal row R rather than to a hidden normalizer or host evaluator.

If the request is a replay of already displayed data, apply fuel-window replay Definition 749.25. The structural rows H, C, P, N can transport, replay, source, and name the boundary, but they cannot erase fuel or add a new comparison endpoint.

The candidate-substitution and refusal exactness rows separate these cases. A request with B, F present is the bounded-comparison route; a request lacking the side condition is the refusal route through R; and a request that only reuses displayed data is the replay route through H, C, P, N.

Finally use the conversion nonescape package Definition 749.26. Since the carrier has no coordinate for global beta-conversion, fuel-erased normalization, quotient equality, or host evaluation, the three displayed routes exhaust the conversion-facing classifier. □

Definition 749.29 (MetaCIC decidable boundary endpoint subcarrier)**.** *For an accepted* MetaCICDecidableBounda *packet $D = (T, S, B, F, R, H, C, P, N)$, the endpoint subcarrier is the finite packet*

$$D_{\text{end}} = (T, B, F, R, H, C, P, N)$$

consisting of the closed-term row T, bounded-normal comparison row B, finished-normal witness row F, refusal row R, componentwise transport H, displayed replay C, provenance P, and local $\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}^{\uparrow}}$ *row N. It deliberately omits any global conversion decider, quotient equality over represented terms, fuel-free normalizer, or host evaluator.*

Theorem 749.30 (MetaCIC decidable boundary finished normal endpoint)**.** *Every global closed beta-conversion decision read from an accepted* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *packet may be read only after the endpoint subcarrier of Definition 749.29 supplies finished-normal witnesses through F. The bounded normal comparison row B alone remains a refusal boundary: without the endpoint rows T, F, R, H, C, P, N, it cannot export global conversion decidability, quotient equality over represented terms, fuel-erased normalization, or host evaluation.*

Proof. Start with the normal-form decision envelope Definition 749.27. It places every normal-form decision read on the finite route $T, S, B, F, R, H, C, P, N$ and keeps the positive comparison row tied to finished-normal side conditions.

Apply the conversion nonescape package Definition 749.26. Conversion-facing reads factor through B, F, R, H, C, P, N, with R recording the missing global endpoints.

Now project the endpoint subcarrier of Definition 749.29. It retains the closed-term row, bounded-normal comparison row, finished-normal witnesses, refusal row, transport, replay, provenance, and local naming, so a conversion read is available only with F present.

If B is read without this endpoint data, the route has only bounded comparison under a displayed window. The refusal row still blocks global beta-conversion decidability, quotient equality, fuel-erased normalization, and host evaluation. □

Theorem 749.31 (MetaCIC decidable boundary conversion case split)**.** *For every accepted* MetaCICDecidableBoun *packet $D = (T, S, B, F, R, H, C, P, N)$, each conversion request presented to the packet is split inside the displayed* BHist *route into an accepted finite case using T, S, B, F or a refusal row in R. The split is preserved by componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}^{\uparrow}}$ *row N.*

Proof. Use conversion refusal trichotomy. It classifies conversion-facing requests as bounded comparison with B, F, finite refusal through R, or structural replay through H, C, P, N.

The normal-form decision envelope supplies the accepted finite cases through T, S, B, F. The ClosedAt refusal route supplies the negative rows in R when the request asks for a standalone endpoint outside the packet.

Transport and replay are componentwise by classifier stability. Hence the split is a BHist and NameCert-facing route, not a parameter transport theorem and not a hidden global conversion decider. □

Theorem 749.32 (MetaCIC decidable boundary critical-pair finiteness)**.** *For every accepted* MetaCICDecidableBou *packet, every critical-pair check visible to the decidable-boundary surface is recorded as a finite ledger row inside $T, S, B, F, R, H, C, P, N$. The ledger records the inspected checker window, structural-equality comparison, bounded normal comparison, finished-normal side condition, refusal boundary, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming; it exports no unrecorded critical-pair search and no host evaluator.*

Proof. Start with window totality and ledger exactness. They make each bounded comparison a finite inspected window whose rows are T, S, B, F together with the refusal and structural context R, H, C, P, N.

Apply substitution window and candidate-substitution exhaustion. Critical-pair inspection is therefore read only through displayed checker, structural equality, bounded normal, finished-normal, refusal, replay, provenance, and naming rows.

The bounded comparison nonescape theorem blocks promotion to unrestricted conversion, quotient equality, fuel-erased normalization, or host evaluation. Thus the critical-pair check is a finite ledger row of the packet. □

Theorem 749.33 (MetaCIC decidable boundary normal-form readback)**.** *Every normal-form readback available from an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet is routed through the displayed boundary packet $T, S, B, F, R, H, C, P, N$. The readback may expose candidate normal-form data, bounded comparison under finished-normal side conditions, and finite refusal rows, but it cannot bypass the packet to obtain fuel-erased normalization, quotient equality over represented terms, standalone* ClosedAt *decidability, or host evaluation.*

Proof. Begin with candidate normal-form reads and the normal-form decision envelope. Together they route readback through the checker row T, structural equality row S, bounded comparison row B, finished-normal side condition F, and refusal row R.

Use fuel-window replay and the finished normal endpoint theorem. These keep readback tied to finite fuel and displayed endpoint rows before a conversion-facing read is accepted.

Finally apply the conversion nonescape package. The structural rows H, C, P, N only transport, replay, source, and name the displayed boundary, so readback cannot leave the packet for a fuel-erased normalizer, quotient equality, standalone ClosedAt decider, or host evaluator. □

Theorem 749.34 (MetaCIC decidable boundary public interface)**.** *Every public read exported by an accepted* MetaCICDecidableBoundary$^{\uparrow}$ *packet is confined to the bounded checker output, fuel-window replay, candidate-normal reads, substitution windows, refusal rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows of $T, S, B, F, R, H, C, P, N$. The public interface explicitly excludes global beta-conversion decidability, confluence, quotient equality over represented terms, fuel-erased normalization, and host evaluation.*

Proof. Start with fuel-window replay. It keeps any bounded comparison read tied to the fuel and finished-normal side-condition row F, so a public consumer cannot erase the window boundary.

Apply candidate-substitution exhaustion and the substitution-window theorem. Candidate-normal and substitution-sensitive reads remain inside $T, S, B, F, R, H, C, P, N$ and expose only the displayed checker, structural equality, bounded comparison, side-condition, and refusal rows.

Use normal-form readback to identify the accepted public read. It routes readback through the same boundary packet and blocks fuel-erased normalization, quotient equality, standalone ClosedAt decidability, and host evaluation.

Finally read the TasteGate single-carrier alignment marker at the top of the chapter. The carrier alignment keeps transport, replay, provenance, and naming attached to one finite packet, so the public interface cannot add confluence, global beta-conversion decidability, quotient equality, or host evaluation as hidden exports. □

Theorem 749.35 (MetaCIC decidable boundary bridged status)**.** *The public interface of Definition 749.34 together with the fuel-window replay target* `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDec` *gives a finite bridge surface from bounded checker rows to decidable-boundary consumers. The*

bridge surface preserves $T, S, B, F, R, H, C, P, N$ and refuses standalone ClosedAt *decidability, global beta-conversion decidability, fuel-erased normalization, quotient term equality, and host evaluation.*

Proof. Start from the carrier and NameCert obligations. They expose the finite packet $T, S, B, F, R, H, C, P, N$ as checker rows, structural equality, bounded normal comparison, finished-normal side conditions, refusal, transport, replay, provenance, and local naming.

Use fuel-window replay and normal-form readback. The replay keeps bounded comparison tied to F, while readback stays inside the displayed boundary packet before any public consumer receives a decision-facing row.

Finally apply the public interface theorem. It exports only the bounded checker, fuel-window, candidate-normal, substitution, refusal, transport, replay, provenance, and naming rows, so the bridge-facing surface has no coordinate for standalone ClosedAt decidability, global beta-conversion decidability, fuel-erased normalization, quotient equality, or host evaluation. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}\uparrow}$.

Constructive story: A $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ packet is generated from inferTypeCtx checking, checkSameTerm structural equality, bounded normal comparison, finished-normal side conditions, refusal rows for ClosedAt and global beta conversion, componentwise transport, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}\uparrow}$, maturePackageClosed)

Scope closed: Definition 749.1, Definition 749.2, Definition 749.3, Definition 749.4, Definition 749.5, Definition 749.11, Definition 749.12, Definition 749.13, Definition 749.14, Definition 749.34, and Definition 749.35 bind carrier admission, checker-row visibility, bounded-comparison nonescape, refusal exactness, sibling provenance, fuel-window obligation, candidate-normal read, replay obligation, substitution window, public interface, and bridge-facing status for the displayed packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICDecidableBoundary}\uparrow}$, axiomClean)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.metaCICDecidableBoundaryDeltaLedger

Lean target: `BEDC.Derived.MetaCICDecidableBoundaryUp.MetaCICDecidableBoundaryFuelWindowReplay`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close a standalone ClosedAt decider, global beta-conversion decidability, fuel-erased normalisation, quotient equality over represented terms, or bridge checking.

Upgrade path: Bridged closure requires a bridge surface that preserves the same finite decidable-boundary rows without claiming confluence, global beta-conversion decidability, fuel-erased normalization, quotient equality, or host evaluation.

Chapter 750

A Concrete Naming Certificate for $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$

The $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ packet realizes the normalization audit map of chapter 9383 as finite BEDC naming data. It sits beside the confluence audit packet of chapter 743 and the confluence audit witness of chapter 745: those siblings record confluence-side boundary rows, while the present packet records the strong-normalization route, its refusal boundary, and the finite replay surface used by consistency consumers.

Definition 750.1 (MetaCIC normalization audit carrier)**.** *A* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (R, C, S, A, N, F, H, T, P, L)$$

with the following displayed rows:

1. *R is the reducibility-candidate row naming* Candidate*,* ClosedTypedCandidate*,* ClosedTypedPiCandidate*, and* ClosedTypedCandidateSetMember *evidence;*
2. *C is the checked closed-term projection row for candidate, closed-candidate, and candidate-set routes to* StrongNormalizable(t)*;*
3. *S is the finite strong-normalization scope row containing the checked candidate-to-SN, Pi adequacy, application adequacy, example SN, and no-infinite-reduction reads;*
4. *A is the application-boundary row recording that componentwise SN of function and argument is not reread as unconditional application SN;*
5. *N is the confluence-boundary refusal row separating this normalization packet from full closed Church–Rosser, substitution compatibility, and subject-reduction transport;*
6. *F is the route replay row for manuscript citations that pass from the audit map to the closed-normal consistency consumer;*
7. *H records componentwise* hsame *transport across R, C, S, A, N, F, P, L;*
8. *T records displayed* Cont *routes for replaying candidate, projection, scope, boundary, and consistency-consumer reads;*
9. *P is the* Pkg *provenance over the normalization audit map, the confluence audit sibling, the confluence audit witness sibling, and the closed-normal consistency assembly;*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAudit}^{\uparrow}}$ *row.*

The carrier has no row for global strong normalization of every closed typed MetaCIC term without candidate-mediated evidence, no unconditional application-SN theorem, no closed subject-reduction transport theorem, no arbitrary closed-term confluence theorem, no quotient equality over reductions, and no host evaluator.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditTasteGate single carrier alignment`

Theorem 750.2 (MetaCIC normalization audit obligation surface)**.** *Every accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet binds exactly the finite obligation surface displayed in Definition 750.1: carrier admission for* $Z = (R, C, S, A, N, F, H, T, P, L)$*, candidate evidence visibility through* R*, closed-term projection visibility through* C*, finite SN scope visibility through* S*, application-boundary visibility through* A*, confluence and transport refusal visibility through* N*, route replay through* F *and* T*, componentwise* hsame *transport through* H*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier coordinates. The positive normalization reads are R, C, S: a consumer first names candidate evidence, then reads the closed-term projection row, and only then reads the finite SN and no-infinite-reduction inventory.

The boundary rows A and N are part of the same packet. They keep application SN and confluence or subject-reduction transport from being inferred by erasing their required evidence.

The structural rows F, H, T, P, L replay, transport, package, and name only those displayed reads. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier obligation surface`

Theorem 750.3 (MetaCIC normalization audit candidate boundary)**.** *Every accepted* MetaCICNormalizationAudit$^{\uparrow}$ *read that claims normalization must first expose the candidate packet in* R*, the closedness trace in* C*, the beta-chain route in* S*, and the residual and confluence boundary rows in* A *and* N*. The replay rows* F, H, T, P, L *may transport and name that same route, but they do not supply subject reduction, confluence, or strong normalization unless downstream candidate, residual, or frontier packets supply those rows explicitly.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateBoundary`

Proof. Start with the obligation surface of Definition 750.2. It makes R, C, S, A, N separate displayed rows of the accepted packet, so a normalization read cannot jump directly from the packet name to a closed-term conclusion.

The candidate packet is read through R, and the closedness trace is read through C. The beta-chain route and finite normalization inventory are then read through S, with the application boundary A still present.

Finally inspect the refusal row N together with replay through F, H, T, P, L. Those rows preserve the same finite route and keep the confluence and subject-reduction frontier visible. Therefore subject reduction, confluence, and strong normalization outside the displayed candidate route must come from downstream candidate, residual, or frontier packets rather than from this audit packet alone. □

Theorem 750.4 (MetaCIC normalization audit finite SN scope)**.** *The strong-normalization content exported by an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet is exactly the candidate-mediated*

scope stored in R, C, S: candidate-to-SN consequences, closed typed term SN from candidate, closed-candidate, or candidate-set evidence, Pi adequacy, application adequacy under candidate evidence, typed example SN facts, and the associated no-infinite-reduction consequences. It does not export a blanket theorem that every closed typed MetaCIC term is strongly normalizing without one of those displayed evidence rows.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier sn scope`

Proof. Read R as the evidence row. It contains the candidate, closed-candidate, Pi-candidate, and candidate-set membership interfaces that the checked projection theorems consume.

Read C and S downstream from R. These rows record the closed typed term projections, adequacy facts, examples, and no-infinite-reduction consequences only along the candidate-mediated route.

The carrier contains no coordinate that constructs candidate evidence for every closed typed term. Therefore agreement on the packet preserves the finite SN scope while excluding the blanket evidence-free endpoint. □

Theorem 750.5 (MetaCIC normalization audit closed-term projection route)**.** *Every closed-term strong-normalization read exported by an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet must pass through the reducibility-candidate row R, the checked closed projection row C, the finite scope row S, and the application-boundary row A. The route may project candidate, closed-candidate, candidate-set, Pi, and application evidence into the finite SN inventory, but it cannot replace those rows by evidence-free global strong normalization for closed MetaCIC terms.*

Proof. Project the carrier $Z = (R, C, S, A, N, F, H, T, P, L)$. A closed-term SN consumer first sees candidate evidence in R and the checked closed projection row C, so the read is already routed through displayed finite data.

Apply Definition 750.4. The finite SN consequences live in S only after the candidate-mediated projection has supplied the relevant evidence.

Finally apply Definition 750.6. The application-facing boundary A keeps application SN inside the same candidate route and refuses any evidence-free global SN endpoint. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier closed term projection route`

Theorem 750.6 (MetaCIC normalization audit candidate-mediated SN closure)**.** *Let $Z = (R, C, S, A, N, F, H, T, P$ be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet. If a consumer reads the candidate row R, the closed-term projection row C, the finite strong-normalization scope row S, and the application-boundary row A, then every exported strong-normalization claim factors through those rows and through no evidence-free global strong-normalization theorem. In particular, an SN export may name candidate evidence, closed-term projection, finite scope inventory, and application adequacy under the displayed boundary, but it cannot replace them by a blanket theorem for all closed typed MetaCIC terms.*

Proof. Apply Definition 750.2. The accepted packet exposes normalization reads only through the displayed candidate, projection, finite-scope, application-boundary, refusal, replay, transport, provenance, and naming rows.

Use Definition 750.4. It identifies the strong-normalization content as the candidate-mediated scope carried by R, C, S and explicitly withholds a blanket endpoint for closed typed terms without one of the displayed evidence rows.

Finally apply Definition 750.16. The ledger route forces manuscript consumers to replay the same packet coordinates and blocks host termination or hidden theorem reads. Since A is part of the accepted boundary, application-facing SN claims remain inside the candidate-mediated route rather than becoming an evidence-free global theorem. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateMediatedSNClosure`

Theorem 750.7 (MetaCIC normalization audit residual-budget obligation)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Every exported strong-normalization claim must spend an explicit residual-budget row whose readback links candidate evidence in R, closed-term projection in C, finite strong-normalization scope in S, the application boundary in A, and the confluence refusal in N. A consumer that cites candidate-mediated SN without this residual-budget route receives only the local obligation surface, not a public SN export.*

Proof. Start with the obligation surface of Definition 750.2. It displays R, C, S, A, N as separate rows, so an exported SN claim must account for each row rather than reading the finite scope in isolation.

Use Definition 750.4. The positive SN content is the finite candidate-mediated inventory stored in R, C, S, with no blanket endpoint for closed typed terms lacking the displayed evidence.

Next apply Definition 750.6. It keeps application-facing SN claims inside the same candidate route and prevents the application boundary A from being erased by a downstream consumer.

The residual budget is completed by the refusal row N. That row records that confluence, subject-reduction transport, and full closed Church–Rosser remain outside the normalization export. Replaying the route through F, T, H, P, L names the same spent budget and leaves no coordinate for evidence-free strong normalization. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditResidualBudgetObligation`
↪ variant `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier residual budget obligation`

Theorem 750.8 (MetaCIC normalization audit candidate-ledger locality)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Every strong-normalization export must keep candidate evidence R, closed projection C, finite scope S, application boundary A, and confluence refusal N in the same candidate ledger before replay through F, H, T, P, L. No consumer may replace those rows by global strong normalization, subject reduction, confluence, quotient equality, or host evaluator evidence.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateLedgerLocality`

Proof. Begin with the obligation surface of Definition 750.2. It exposes R, C, S, A, N as separate rows of one accepted packet, so an SN export cannot read the finite scope while dropping the candidate evidence, closed projection, application boundary, or refusal row.

Use the finite SN scope theorem Definition 750.4 and the closed-term projection route of Definition 750.5. Together they place every positive SN read behind candidate evidence in R, projection evidence in C, and finite scope in S.

Apply the residual-budget obligation Definition 750.7. The budget links R, C, S with the application boundary A and confluence refusal N before any replay row can expose the claim downstream.

Finally replay through F, H, T, P, L. These structural rows preserve the same candidate ledger and cannot supply global strong normalization, subject reduction, confluence, quotient equality, or a host evaluator, since none of those objects appears as a carrier row. □

Theorem 750.9 (MetaCIC normalization audit confluence refusal split)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. The packet may export candidate-mediated strong normalization only through the displayed rows*

$$R, C, S, A, N, F, H, T, P, L.$$

The row N keeps confluence, subject-reduction transport, quotient reduction equality, and host evaluator evidence outside the export unless a downstream confluence packet supplies those rows explicitly. Thus a consistency consumer may read candidate evidence, closed projection, finite SN scope, application boundary, refusal boundary, replay, transport, continuation, provenance, and local naming, but it may not turn that read into confluence or subject reduction by erasing N.

Proof. Begin with the obligation surface of Definition 750.2. It makes $R, C, S, A, N, F, H, T, P, L$ the complete carrier inventory for an accepted normalization audit packet.

Use the finite SN scope theorem Definition 750.4. Positive normalization content is confined to the candidate-mediated rows R, C, S, with the application boundary A still visible.

Next apply the residual-budget obligation of Definition 750.7. The budget explicitly spends the refusal row N before an SN export can reach downstream replay.

Finally apply candidate-ledger locality, Definition 750.8. The replay and provenance rows F, H, T, P, L preserve the same candidate ledger and add no confluence, subject-reduction transport, quotient reduction equality, or host evaluator evidence. Those objects can enter only from a separate downstream confluence packet that displays them as its own rows. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditConfluenceRefusalSplit`

Theorem 750.10 (MetaCIC normalization audit public candidate export)**.** *For an accepted* MetaCICNormalization*packet*

$$Z = (R, C, S, A, N, F, H, T, P, L),$$

the public export available to consistency consumers is exactly the candidate-mediated strong-normalization route through R, C, S, A, N and replay through F, H, T, P, L. It may cite the finite SN inventory, the closed-term projection route, the application boundary, the residual-budget obligation, and candidate-ledger locality, but it exports no evidence-free global strong normalization, subject reduction, confluence, quotient equality, or host evaluator.

Proof. First read the obligation surface and finite SN scope. Apply Definition 750.2 and Definition 750.4; together they place every positive normalization read behind candidate evidence R, checked projection C, and finite inventory S.

Next use the closed-term projection route and application boundary. Apply Definition 750.5; the row A remains visible, so a closed-term SN consumer cannot erase the candidate route and read the claim as global normalization.

Now spend the residual budget. Apply Definition 750.7; the row N keeps confluence, subject reduction, and full closed Church–Rosser outside the normalization export.

Finally apply candidate-ledger locality. The replay rows F, H, T, P, L preserve the same displayed ledger and add no host evaluator, quotient equality, confluence theorem, subject-reduction theorem, or evidence-free SN endpoint. Thus the public candidate export is exactly the finite route stated above. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditPublicCandidateExport`

Theorem 750.11 (MetaCIC normalization audit public candidate-confluence refusal)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. A public consumer that combines this normalization audit with confluence claims must display the candidate rows R, C, S, A, N and the confluence-audit sibling rows of chapter 743 and chapter 745 before exporting any route. Without those displayed sibling rows, the consumer receives only candidate-mediated strong normalization through $R, C, S, A, N, F, H, T, P, L$, not confluence, subject reduction, quotient conversion, or global normalization.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetaCICNormalizationAuditPublicCandidateConfluenceRefu`

Proof. Start with the obligation surface and finite SN scope. Theorems 750.2 and 750.4 force every positive normalization read through the candidate, projection, finite-scope, application-boundary, and refusal rows R, C, S, A, N.

Next apply the public candidate export of Definition 750.10. It exposes exactly the candidate-mediated SN route and keeps replay through F, H, T, P, L inside the same finite ledger.

Finally use the confluence boundary of Definition 750.15. The row N points to the confluence-audit siblings, but it does not contain their confluence, subject-reduction, or quotient-conversion rows. A public route that omits those siblings therefore has only the normalization packet's candidate-mediated SN export. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditPublicCandidateConfluenceRefu`

Theorem 750.12 (MetaCIC normalization audit candidate-frontier refusal)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Any strong-normalization export that crosses from candidate evidence to closed-term normalization must display the rows*

$$R, C, S, A, N, F, H, T, P, L.$$

If the candidate row R, the closed-candidate projection row C, or the finite candidate-set scope row S is absent, the packet returns only the local obligation surface and does not export closed-term normalization.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateFrontierRefusal`

Proof. Start with the obligation surface of Definition 750.2. It exposes candidate evidence, closed-term projection, finite SN scope, application boundary, refusal, route, transport, replay, provenance, and local naming as separate rows, so a closed-term normalization read cannot skip directly to a conclusion.

Use the SN scope theorem Definition 750.4 and the closed-term projection route recorded by Definition 750.6. Together they allow closed-term SN only when the candidate evidence, closed-candidate projection, and candidate-set scope rows have already been displayed.

Now apply the residual-budget obligation of Definition 750.7. The residual row accounts for the application boundary and confluence refusal before any SN export leaves the candidate side.

If any of R, C, S is absent, the route lacks the evidence needed to cross the frontier. The structural rows F, H, T, P, L can only replay and name the local obligation surface, so the packet refuses evidence-free closed-term normalization. □

Theorem 750.13 (MetaCIC normalization audit candidate-frontier refusal tightening)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. If a consumer route omits at least one of the reducibility-candidate row R, the closed projection row C, or the finite scope row S, then the accepted packet exports only the local obligation surface. In particular it exports no closed-term strong-normalization claim.*

Proof. Begin with the obligation surface of Definition 750.2. It separates candidate evidence, closed projection, finite SN scope, application boundary, confluence refusal, replay, transport, provenance, and local naming into distinct rows.

If R is absent, no reducibility-candidate evidence is available. The finite SN scope theorem Definition 750.4 therefore has no candidate source from which a positive normalization read can begin.

If C is absent, the closed-term projection route of Definition 750.5 cannot connect candidate evidence to a closed term. The route remains an internal candidate ledger, not a closed-term SN export.

If S is absent, the finite inventory of accepted normalization entries is missing. The residual-budget obligation of Definition 750.7 and candidate-ledger locality of Definition 750.8 can only preserve replay of the local obligation surface.

Thus every missing-row case leaves the consumer below the candidate frontier isolated by Definition 750.12. The packet may still expose carrier, boundary, replay, transport, provenance, and naming data, but it cannot export closed-term strong normalization. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.CandidateFrontierRefusalTightening.MetacicNormalizatio`

Theorem 750.14 (MetaCIC normalization audit closed-term budget totality)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Every closed-term strong-normalization export must display all rows*

$$R, C, S, A, N, F, H, T, P, L.$$

The rows R, C, S supply candidate evidence, checked closed-term projection, and finite SN inventory; the row A preserves application-boundary spending; and the row N retains the confluence, subject-reduction, and closed Church–Rosser refusal. Thus the packet exports only candidate-mediated closed-term SN with an explicit finite budget, and it exports no evidence-free global strong-normalization theorem.

Proof. Begin with Definition 750.2. It lists $R, C, S, A, N, F, H, T, P, L$ as the complete local obligation surface, so a closed-term SN export cannot omit a carrier row.

Use Definition 750.4. The positive SN content is candidate-mediated: R names candidate evidence, C records the checked closed-term projection, and S stores the finite strong-normalization inventory.

Apply Definition 750.7. It ties every exported SN read to a residual budget that also passes through the application boundary A and the confluence refusal row N.

Finally use Definition 750.12. If the candidate frontier is not displayed, the packet returns only the local obligation surface. Therefore totality of the closed-term budget means total display of the finite rows above, not an evidence-free global SN theorem. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditClosedTermBudgetTotality`

Theorem 750.15 (MetaCIC normalization audit confluence boundary)**.** *Agreement on the rows of an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet preserves normalization-side reads, but it does not discharge the confluence or subject-reduction rows isolated by chapter 743 and chapter 745. In particular, the packet exports no proof of full closed Church–Rosser, no proof of substitution compatibility, and no subject-reduction transport theorem for arbitrary closed proofs.*

Proof. The provenance row P places the packet beside the two confluence audit siblings, so a consumer can compare the three audit surfaces through finite NameCert data.

The normalization rows R, C, S, A are separate from the refusal row N. Transport through H and replay through T preserve that separation instead of identifying normalization with confluence or subject reduction.

Thus the packet can feed the closed-normal branch of the consistency assembly through F while leaving the confluence and subject-reduction boundaries in their own sibling packets. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier confluence boundary`

Theorem 750.16 (MetaCIC normalization audit ledger policy)**.** *Every manuscript route that cites the normalization audit map through* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *factors through the ledger*

$$R, C, S, A, N, F, H, T, P, L.$$

The route may replay candidate evidence, closed-term SN projections, finite no-infinite-reduction reads, confluence-boundary refusals, and local NameCert provenance, but it may not replace the packet by vision prose, a host-level termination oracle, or a hidden theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier ledger policy`

Proof. Start with a citation to the audit map. The finite carrier replaces that prose handle by displayed rows $R, C, S, A, N, F, H, T, P, L$.

Route replay is controlled by F and T, while H preserves componentwise agreement and P, L preserve provenance and local naming. These rows are sufficient for every packet read named in the theorem.

Because the carrier has no coordinate for a host oracle, hidden theorem, quotient equality, or evidence-free global SN theorem, the ledger policy blocks any consumer from bypassing the finite packet. □

Theorem 750.17 (MetaCIC normalization audit residual fuel decreases)**.** *Let* $Z = (R, C, S, A, N, F, H, T, P, L)$ *be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet equipped with a finite residual-fuel ledger for candidate-mediated strong-normalization reads. Each permitted reduction replay spends one displayed unit of that ledger before the next candidate read is consulted, and the spent ledger is transported through* $R, C, S, A, N, F, H, T, P, L$*. Hence a manuscript consumer may cite a decreasing finite measure for the audited route, but not a global normalization measure for arbitrary closed MetaCIC terms.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditResidualFuelDecreases`

Proof. Project the finite audit rows. The candidate evidence row R, closed-term projection row C, and finite SN scope row S are the only rows from which a reduction replay can obtain normalization evidence.

Attach the residual-fuel ledger to that replay. A permitted step consumes one named fuel entry and then returns to the candidate-mediated rows; the application boundary A prevents the consumer from reading the step as unconditional application SN.

Transport and refusal close the measure. The row H preserves the spent fuel under componentwise agreement, F, T, P, L replay and package the same finite ledger, and N keeps confluence and subject-reduction boundaries outside the measure. The resulting decrease is therefore local to the packet. □

Theorem 750.18 (MetaCIC normalization audit candidate frontier extension)**.** *Let $Z = (R, C, S, A, N, F, H, T, P,$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. If a new closed term is admitted at the candidate frontier by displayed* ClosedTypedCandidateSetMember *evidence in R and its closed-term projection is recorded in C, then the finite SN scope S may be extended by one frontier entry with its residual-fuel row. The extension preserves the application boundary A and the confluence refusal N; it does not assert arbitrary substitution compatibility or full Church–Rosser.*

Proof. Begin at the candidate frontier. The new closed term is not admitted by syntax alone; it arrives with displayed candidate-set membership in R and a matching closed-term projection in C.

Extend the finite scope by adding the corresponding entry to S. The residual-fuel obligation of Definition 750.17 supplies the finite measure attached to the new frontier item.

Finally preserve the boundaries. The application row A keeps application-facing uses inside candidate evidence, while N leaves confluence, substitution compatibility, and subject reduction in their sibling audit surfaces. Structural replay through F, H, T, P, L packages the extended frontier without turning it into a global normalization theorem. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateFrontierExtension`

Theorem 750.19 (MetaCIC normalization audit confluence socket)**.** *For an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet $Z = (R, C, S, A, N, F, H, T, P, L)$, the confluence boundary row N acts as a socket to chapter 743 and chapter 745: normalization-side candidate evidence may be compared with those sibling confluence rows only after the residual-fuel and candidate-frontier obligations have been displayed. The socket records compatibility of audit surfaces, not full closed Church–Rosser, arbitrary substitution compatibility, or subject reduction.*

Proof. Read the normalization route first. The rows R, C, S, A provide candidate-mediated SN and application-boundary evidence, while the residual-fuel and frontier obligations expose the finite measure and admitted candidate boundary.

Now inspect N. It does not prove confluence; it names the boundary at which the normalization audit can be placed beside the confluence audit siblings for comparison of finite rows.

The structural rows F, H, T, P, L preserve that comparison as a socket. They allow the manuscript route to carry normalization evidence up to the confluence boundary, but no row identifies the socket with full Church–Rosser, substitution compatibility, or subject reduction for arbitrary closed proofs. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditConfluenceSocket`

Theorem 750.20 (MetaCIC normalization audit obligation pack)**.** *For an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet $Z = (R, C, S, A, N, F, H, T, P, L)$, the exact audit package is the displayed row family*

$$R, C, S, A, N, F, H, T, P, L.$$

The package contains reducibility-candidate evidence, closed-term projection, finite strong-normalization scope, application boundary, confluence refusal, route replay, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It contains no evidence-free global strong-normalization theorem, no unconditional application-SN theorem, no full closed Church–Rosser theorem, no arbitrary substitution-compatibility theorem, no subject-reduction transport theorem, no quotient equality over reductions, and no host evaluator.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditObligationPack`

Proof. Project the carrier of Definition 750.1. Its coordinates are exactly $R, C, S, A, N, F, H, T, P, L$, with no hidden normalization, confluence, quotient, or evaluator coordinate.

Apply Definition 750.2. That theorem identifies the local NameCert obligations for carrier admission, candidate visibility, projection visibility, finite SN scope, application boundary, confluence refusal, replay, transport, provenance, and local naming.

Use Definition 750.4, Definition 750.5, and Definition 750.7. Together they keep every SN read inside candidate evidence, closed-term projection, finite scope, residual budget, and refusal rows. The refused global endpoints therefore have no row in the exact audit package. □

Theorem 750.21 (MetaCIC normalization audit candidate-route pack)**.** *Before any strong-normalization export from an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet is public, the manuscript route must display candidate evidence in R, closed-term projection in C, residual fuel through the finite scope and replay rows S, F, T, the application boundary A, and the confluence refusal row N. The resulting public-facing route is*

$$R, C, S, A, N, F, H, T, P, L.$$

Consequently a candidate-mediated SN export may cite closed-term projection, candidate-set membership, residual fuel, application adequacy under the displayed boundary, and the confluence socket, but it cannot cite evidence-free global SN, full closed Church–Rosser, arbitrary substitution compatibility, subject reduction, quotient reduction equality, or a host evaluator.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateRoutePack`

Proof. Begin with the closed-term projection route of Definition 750.5. It forces a closed-term SN read through candidate evidence R, projection C, finite scope S, and the application boundary A.

Attach the candidate-mediated closure theorem Definition 750.6. This keeps application-facing SN inside the displayed candidate route and blocks the evidence-free global endpoint.

Now spend the residual budget using Definition 750.7. The residual route links R, C, S, A to the refusal row N, while replay and transport through F, H, T, P, L preserve the same public-facing packet.

Finally apply the confluence socket of Definition 750.19. The socket permits comparison with sibling confluence rows only after the residual-fuel and candidate-frontier obligations are visible, so the candidate-route pack remains a finite audit route rather than a global SN or confluence theorem. □

Theorem 750.22 (MetaCIC normalization audit public route)**.** *Any public normalization export from an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet $Z = (R, C, S, A, N, F, H, T, P, L)$ must display the complete route*

$$R, C, S, A, N, F, H, T, P, L.$$

Here R is candidate evidence, C is closed-term projection, S is finite strong-normalization scope, A is the application boundary, N is the confluence refusal row, F is route replay, H is componentwise hsame *transport, T is displayed* Cont *replay, P is provenance, and L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAudit}^{\uparrow}}$ *row. This public route may be consumed by closed-normal consistency routes, but it is not evidence-free global strong normalization, full closed Church–Rosser, arbitrary substitution compatibility, subject-reduction transport, quotient reduction equality, or a host evaluator.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditPublicRoute`

Proof. Start with Definition 750.21. It already identifies the candidate-facing public route as the displayed row family $R, C, S, A, N, F, H, T, P, L$.

Use Definition 750.6. The strong-normalization content remains inside candidate evidence, closed-term projection, finite scope, and the application boundary; no evidence-free global SN endpoint is introduced.

Apply Definition 750.7. The residual-budget row ties the SN read to the confluence refusal N before replay, transport, provenance, and local naming can expose it to a public consumer.

Finally apply Definition 750.19. The socket allows the route to meet closed-normal consistency consumers beside the confluence audit siblings, but it records only compatibility of finite audit surfaces. Thus the public export is exactly the displayed finite route and not a global confluence, substitution, subject-reduction, quotient, or evaluator claim. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCarrier public route`

Theorem 750.23 (MetaCIC normalization audit candidate-substitution boundary)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Any substitution-shaped normalization export must display candidate evidence R, closed projection C, finite strong-normalization scope S, application boundary A, confluence refusal N, and replay rows F, H, T, P, L in the same finite route. Without those rows, the export remains a local obligation surface; it is not subject reduction, arbitrary substitution compatibility, full closed Church–Rosser, quotient reduction equality, or a host evaluator read.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateSubstitutionBoundary`

Proof. Start with the public route of Definition 750.22. It already exposes the candidate-facing normalization read only through $R, C, S, A, N, F, H, T, P, L$.

Apply the candidate-frontier refusal of Definition 750.12. A substitution-shaped consumer cannot cross from candidate evidence to closed-term normalization unless R, C, S are present.

Use the residual-budget obligation Definition 750.7. The residual row links the candidate route to the application boundary A and the confluence refusal N before the replay rows can expose the claim downstream.

Finally apply the confluence socket of Definition 750.19. The socket permits comparison with sibling confluence rows only as finite audit-surface compatibility. Since the carrier has no row for subject reduction, arbitrary substitution compatibility, full closed Church–Rosser, quotient equality, or a host evaluator, a route missing the displayed rows remains local. □

Theorem 750.24 (MetaCIC normalization audit candidate-substitution scope)**.** *Let $Z = (R, C, S, A, N, F, H, T, P,$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Any substitution-shaped normalization read must keep all rows $R, C, S, A, N, F, H, T, P, L$ visible in the same finite candidate ledger. It cannot erase the refusal row N into subject reduction, full confluence, quotient equality, or evaluator evidence, and it cannot replace the candidate, projection, scope, application, replay, transport, provenance, or naming rows by an ambient normalization theorem.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateSubstitutionScope`

Proof. Start with candidate-ledger locality, Definition 750.8. It keeps every positive normalization export inside the candidate evidence R, closed projection C, finite scope S, application boundary A, and confluence refusal N before replay through F, H, T, P, L.

Use the closed-term projection route Definition 750.5. A substitution-shaped read that reaches a closed-term normalization conclusion still has to pass through R, C, S, A; it does not obtain a separate subject reduction or evaluator row.

Apply the confluence-refusal split Definition 750.9. The row N names the confluence and transport frontier, so it remains a displayed refusal coordinate rather than being converted into full confluence, quotient equality, or subject-reduction transport.

Finally apply the refusal scope of Definition 750.33. Since the refused endpoints are not carrier coordinates of Z, the substitution-shaped read retains exactly $R, C, S, A, N, F, H, T, P, L$ and no stronger normalization, confluence, quotient, or evaluator evidence. □

Theorem 750.25 (MetaCIC normalization audit candidate-confluence handoff)**.** *Let $Z = (R, C, S, A, N, F, H, T, P$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Candidate rows from Z may be compared with the confluence-audit sibling rows of chapter 743 and chapter 745 only through the displayed candidate-frontier row, residual-substitution row, residual-fuel row, and confluence socket row. A consumer that omits any of those rows receives only the local normalization-audit surface.*

The handoff is not full Church–Rosser, arbitrary substitution compatibility, subject reduction, quotient conversion, global normalization, or a host evaluator read.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateConfluenceHandoff`

Proof. Start with the candidate-substitution boundary above. It already requires candidate evidence R, closed projection C, finite scope S, application boundary A, confluence refusal N, and replay rows F, H, T, P, L before a substitution-shaped normalization export is visible.

Read the candidate-frontier row next. The frontier obligations admit only displayed candidate evidence and matching closed-term projection into the finite scope, so a confluence-facing comparison cannot be made from syntax alone.

Now spend the residual rows. The residual-substitution compatibility theorem keeps substitution-shaped reads inside the same candidate ledger, while the residual-fuel decrease row supplies the finite replay measure used by the handoff.

Finally apply the confluence socket. It permits comparison with the sibling confluence rows only as compatibility of displayed audit surfaces. Since the carrier has no coordinate for full Church–Rosser, arbitrary substitution compatibility, subject reduction, quotient conversion, global normalization, or a host evaluator, none of those endpoints is exported by the handoff. □

Theorem 750.26 (MetaCIC normalization audit candidate-confluence boundary)**.** *Let $Z = (R, C, S, A, N, F, H, T,$ be an accepted* MetaCICNormalizationAudit$^{\uparrow}$ *packet. Any consumer comparing the normalization-audit rows of Z with the confluence-audit siblings must display candidate evidence R, closed projection C, finite SN scope S, application boundary A, residual-fuel and replay rows F, H, T, P, L, and the confluence refusal N in the same route. Omitting any of these rows leaves only the local normalization-audit surface, not substitution compatibility, subject reduction, closed Church–Rosser, quotient conversion, global normalization, or evaluator evidence.*

Proof. Start with the candidate-confluence handoff above. It permits comparison with the sibling confluence audits only through displayed candidate-frontier, residual-substitution, residual-fuel, and confluence-socket rows.

Use the residual-substitution compatibility theorem below. It keeps every substitution-shaped read inside the same candidate ledger before any socket read is exported, so the comparison cannot drop R, C, S, A or the replay rows F, H, T, P, L.

Now inspect the confluence socket of Definition 750.19. The socket names a place where finite audit surfaces can be compared; it is not itself full confluence, substitution compatibility, subject reduction, quotient conversion, global normalization, or evaluator evidence.

Finally apply the refusal scope of Definition 750.33. The refusal row N keeps those endpoints outside the packet unless a separate confluence-side certificate displays them. Hence a consumer that omits any listed row has only the local normalization-audit surface. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateConfluenceBoundary`

Theorem 750.27 (MetaCIC normalization audit retained-premise budget)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet. Every normalization-audit consumer that uses candidate-mediated strong normalization must retain, in the same finite route, the candidate witness in R, the closed-candidate wrapper or candidate-set membership row in C, the Π and application adequacy row in A, the residual-fuel row through F, H, T, P, L, and the confluence-refusal row N before citing any closed-term budget. A consumer that drops one of these premises has only the local candidate ledger, not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Begin with the candidate boundary Definition 750.3. It keeps the candidate witness, closed projection, and finite strong-normalization inventory visible as rows of the accepted packet before any closed-term consequence is read.

Next spend the residual budget by Definition 750.7. The closed-term budget is tied to the application boundary and to the confluence refusal before replay through F, H, T, P, L can expose it downstream.

Now pass through the public route Definition 750.22. That route displays exactly $R, C, S, A, N, F, H, T, P, L$, so the candidate witness, closed-candidate or candidate-set row, adequacy row, residual fuel, and refusal row remain retained premises rather than ambient assumptions.

Finally apply the candidate-confluence boundary above and the refusal scope of Definition 750.33. These theorems allow only compatibility with displayed audit surfaces and explicitly keep full Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency outside the normalization-audit packet. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditRetainedPremiseBudget`

Theorem 750.28 (MetaCIC normalization audit candidate-mediated SN/confluence boundary)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet. Any candidate-mediated read that joins strong-normalization evidence to the public candidate-confluence handoff must retain the closed-term budget C, residual fuel and replay rows F, H, T, P, L, candidate frontier R, finite scope S, substitution boundary A, and confluence refusal N in the same finite route. Only after those rows are displayed may the read use Definition 750.25.*

The boundary refuses evidence-free global strong normalization, unconditional application SN, full Church–Rosser, quotient equality over reductions, and arbitrary subject-reduction transport.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateMediatedSNConfluence`

Proof. Begin with the finite SN scope Definition 750.4 and the residual-budget obligation Definition 750.7. They keep the normalization read inside the displayed closed projection, finite scope, application boundary, confluence refusal, and replay rows before any public comparison is made.

Next apply the residual-fuel decrease theorem Definition 750.17. The fuel row is spent along the accepted replay, so a candidate-mediated SN read cannot be detached from F, H, T, P, L or treated as a global normalization theorem.

Now use the candidate-frontier extension Definition 750.18 and the candidate-substitution boundary Definition 750.23. The frontier admits only displayed candidate evidence with a matching closed-term projection, and the substitution boundary keeps application-shaped reads inside $R, C, S, A, N, F, H, T, P, L$.

Finally apply the candidate-confluence handoff and boundary, Definition 750.25 and Definition 750.26. They permit comparison with confluence-audit siblings only as a finite audit-surface handoff. Since the refused endpoints are not carrier rows, the accepted packet cannot export global SN, unconditional application SN, full Church–Rosser, quotient reduction equality, or arbitrary subject-reduction transport. □

Theorem 750.29 (MetaCIC normalization audit residual-substitution compatibility)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet. If the candidate frontier is extended by displayed candidate evidence and the residual-fuel ledger decreases along the accepted replay, then every substitution-shaped read remains inside the same candidate ledger R, C, S, A before any confluence-socket read through N is exported. The packet may therefore compare the resulting finite audit surface with its confluence siblings, but it cannot export arbitrary substitution compatibility, subject reduction, full closed Church–Rosser, quotient reduction equality, or a host evaluator.*

Proof. Begin with the candidate frontier extension of Definition 750.18. The new frontier item is admitted only with displayed candidate-set evidence in R and a matching closed-term projection in C, so the enlarged finite scope remains a candidate-ledger entry rather than a syntax-only substitution claim.

Next apply the residual-fuel decrease theorem Definition 750.17. Each accepted replay spends a named residual entry before the next candidate read is consulted, and the rows F, H, T, P, L transport and replay that same finite route.

Now use the candidate-substitution boundary of Definition 750.23. It keeps substitution-shaped normalization exports inside $R, C, S, A, N, F, H, T, P, L$ and blocks any route that omits the candidate ledger, closed projection, finite scope, or application boundary.

Finally read the confluence socket of Definition 750.19. The socket can be exported only after the residual-fuel and candidate-frontier obligations have been displayed, so the compatibility result is a finite audit-surface route and not a global substitution, subject-reduction, confluence, quotient, or evaluator theorem. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditResidualSubstitutionCompatibi`

Theorem 750.30 (MetaCIC normalization audit candidate residual replay handoff)**.** *Let $Z = (R, C, S, A, N, F, H, T, P, L)$ be an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet. Any public strong-normalization consumer that combines candidate evidence, residual-fuel replay, and confluence-facing comparison must pass through*

$$R, C, S, A, N, F, H, T, P, L$$

in that order. The handoff cannot erase N into subject reduction, full Church–Rosser, arbitrary substitution compatibility, quotient equality, or a host evaluator.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.CandidateResidualReplayHandoff.MetacicNormalizationAud`

Proof. Apply the residual-budget obligation Definition 750.7. It places candidate evidence, closed projection, finite strong-normalization scope, application boundary, and confluence refusal in the ordered ledger R, C, S, A, N before any public read leaves the packet.

Next use the residual-fuel decrease theorem Definition 750.17. The finite replay spends displayed fuel and then returns through F, H, T, P, L, so the consumer receives replayed candidate evidence rather than an evidence-free global normalization endpoint.

Now read the confluence socket of Definition 750.19. The socket permits comparison with the confluence-audit siblings only after the candidate frontier and residual fuel are visible, and it keeps the refusal row N as a displayed boundary.

Finally apply the residual-substitution compatibility theorem above. Substitution-shaped reads remain inside the same candidate ledger and confluence-facing socket, so the public handoff cannot replace N by subject reduction, full Church–Rosser, arbitrary substitution compatibility, quotient equality, or a host evaluator. □

Theorem 750.31 (MetaCIC normalization audit kernel-scope route)**.** *Every normalization-audit read from an accepted* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet is scoped to the displayed kernel and certificate rows*

$$\mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert},\ R,\ C,\ S,\ A,\ N.$$

Here R is reducibility-candidate evidence, C is closed projection, S is the finite strong-normalization inventory, A is the application boundary, and N is the confluence refusal. A consumer that omits any of these rows receives only the local obligation surface and not a public normalization export.

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditKernelScopeRoute`

Proof. Begin with the obligation surface of Definition 750.2. It presents the carrier as finite BHist data transported by hsame, replayed through Cont, sourced by Pkg, and named by the local $\mathsf{NameCert}$ row.

Use the candidate boundary Definition 750.3, the finite SN scope Definition 750.4, and the closed-term projection route Definition 750.5. These three rows force candidate evidence, closed projection, and finite SN inventory to remain visible before a normalization read is public.

Finally apply the residual-budget obligation Definition 750.7. It attaches the application boundary and confluence refusal before replay can expose the read downstream, so the kernel-scope route is exactly the displayed finite packet. □

Theorem 750.32 (MetaCIC normalization audit candidate-boundary scope)**.** *Before any strong-normalization export from* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *is visible, the consumer must display candidate evidence R, closed projection C, finite scope S, application boundary A, and confluence refusal N together with the replay rows F, H, T, P, L. Thus the export remains a candidate-ledger read and not an evidence-free global SN theorem.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditCandidateBoundaryScope`

Proof. Start with candidate-ledger locality, Definition 750.8. It keeps every positive SN read inside candidate evidence, closed projection, and finite scope.

Apply the candidate-route pack of Definition 750.21. The route pack requires $R, C, S, A, N, F, H, T, P, L$ before the public-facing candidate route is exposed.

Use the residual-substitution compatibility theorem above. It shows that even substitution-shaped reads remain inside the candidate ledger until the residual budget and confluence refusal are displayed. Hence the boundary scope is stable for SN exports. □

Theorem 750.33 (MetaCIC normalization audit refusal scope)**.** *The scoped* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet refuses subject reduction, full confluence, quotient equality, host evaluator evidence, and*

evidence-free global strong normalization. These endpoints can be mentioned only as excluded rows attached to the confluence refusal N and cannot be reconstructed from $R, C, S, A, F, H, T, P, L$.

Proof. Read the confluence socket Definition 750.19. It permits comparison with sibling confluence rows only as finite audit-surface compatibility, not as full confluence or subject reduction.

Use the public route Definition 750.22. The only exported surface is $R, C, S, A, N, F, H, T, P, L$, and the theorem explicitly withholds quotient reduction equality and host evaluator evidence.

Finally apply the kernel-scope route and candidate-boundary scope above. They force every normalization read through displayed candidate, projection, finite scope, application boundary, refusal, replay, provenance, and naming rows. Since the refused endpoints are not carrier coordinates, the refusal scope is closed under all displayed reads. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditRefusalScope`

Theorem 750.34 (MetaCIC normalization audit frontier formal target)**.** *A Lean formalization target for* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *must cover exactly the finite candidate-ledger route of the accepted carrier $Z = (R, C, S, A, N, F, H, T, P, L)$. The route starts with candidate evidence in R, closed projection in C, finite strong-normalization inventory in S, application boundary in A, and confluence refusal in N, then exposes replay, transport, continuation, provenance, and local naming through F, H, T, P, L. The target is therefore a candidate-mediated frontier statement, not a broad closed-term normalization theorem, and it must not export evidence-free global strong normalization, arbitrary subject reduction, quotient reduction equality, or host evaluator evidence.*

Proof. Start with the public candidate export of Definition 750.10. It exposes exactly the candidate-mediated route through R, C, S, A, N and replay through F, H, T, P, L, rather than a syntax-wide normalization theorem.

Spend the residual budget by Definition 750.7. The budget keeps candidate evidence, closed projection, finite scope, application boundary, and confluence refusal in the same finite ledger before any public read leaves the packet.

Use candidate-ledger locality, Definition 750.8, and the confluence-refusal split, Definition 750.9. The replay rows F, H, T, P, L preserve the candidate ledger, while N keeps evidence-free global strong normalization, subject reduction, quotient reduction equality, and host evaluator evidence as refused endpoints. Thus the formal target is exactly the finite candidate-mediated frontier named in the statement. □

Lean4 ✓ `BEDC.Derived.MetacicNormalizationAuditUp.MetacicNormalizationAuditFrontierFormalTarget`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ packet is generated from candidate evidence rows, checked closed-term SN projection rows, finite strong-normalization scope rows, application-boundary rows, confluence-boundary refusal rows, route replay, componentwise transport, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAudit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 750.1, Definition 750.11, Definition 750.29, Definition 750.33, and Definition 750.34 display the mature paper surface: candidate-mediated SN and confluence boundary, residual-substitution compatibility inside the same finite ledger, refusal scope for excluded endpoints, and the formal-target frontier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close evidence-free global strong normalization, unconditional application SN, full closed Church–Rosser, substitution compatibility, subject-reduction transport for arbitrary closed proofs, quotient equality over reductions, or bridge checking.

Upgrade path: Lean verification must catch up to the mature paper surface by checking BEDC.Derived.MetacicNormalizationAuditUp.Me and its candidate-mediated boundary obligations.

Chapter 751

A Concrete Naming Certificate for MetaCICDecidabilityWitness$^{\uparrow}$

The MetaCICDecidabilityWitness$^{\uparrow}$ packet realizes the decidability audit map of chapter 9379 as a finite NameCert witness. It sits beside the normal false exclusion packet of chapter 732 and the confluence normal-fragment packet of chapter 737: those siblings keep normal-form and confluence boundaries visible, while this packet separates typing-checker exactness from bounded beta-conversion evidence.

Definition 751.1 (MetaCIC decidability witness carrier)**.** *A* MetaCICDecidabilityWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, S, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* inferTypeCtx *exactness row, containing the soundness and completeness surfaces for the encoded* HasType *relation;*
2. *S is the* checkSameTerm *structural equality row used by the typing checker;*
3. *B is the bounded beta-comparison row for* normalizeBounded *and* decideClosedBetaConv*;*
4. *F is the finished-normal-form side-condition row consumed when a bounded comparison result is read as equality or beta-convertibility of named normal forms;*
5. *R records refusal of a standalone* ClosedAt *decision theorem, unrestricted closed beta-conversion decidability, and fuel-erased normalisation;*
6. *H records componentwise* hsame *transport for T, S, B, F, R, P, N;*
7. *C records displayed* Cont *routes for typing exactness, same-term checking, bounded comparison, finished-normal side conditions, and refusal reads;*
8. *P is the* Pkg *provenance row over the decidability audit map and its normal-form and confluence siblings;*
9. *N is the local* NameCert$_{\text{MetaCICDecidabilityWitness}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the rows $T, S, B, F, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier route ledger certificate`

Theorem 751.2 (MetaCIC decidability witness NameCert obligations)**.** *Every accepted* MetaCICDecidabilityWitnes *packet binds exactly the finite NameCert obligation surface of Definition 751.1: typing-checker exactness through T, same-term structural equality through S, bounded beta comparison through B, finished-normal side conditions through F, refusal boundaries through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier. The positive checker rows are T and S: typing exactness is read through the inferTypeCtx row, while structural equality is read through the same-term checker row used by that typing surface.

The bounded conversion rows are B and F. The comparison row exposes only bounded normalizer output comparison, and the finished-normal side-condition row supplies the additional hypotheses under which the audit map permits equality or beta-convertibility reads for named normal forms.

The refusal row R remains part of the same packet. It records that no coordinate exports a standalone closedness decider, an unrestricted closed beta-conversion decider, or fuel-erased normalisation.

Finally H, C, P, N transport, replay, cite, and name only those displayed rows. Since the carrier has no further coordinate, the local NameCert surface is exhausted by the finite obligations listed above. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessTasteGate single carrier alignment`

Theorem 751.3 (MetaCIC decidability witness bounded conversion non-escape)**.** *Any consumer of an accepted* MetaCICDecidabilityWitness$^{\uparrow}$ *packet may read conversion evidence only through B together with F. The read can expose bounded-normal output equality and the finished-normal consequences named by the audit map, but it cannot export a global closed beta-conversion decision procedure, a fuel-erased normalisation theorem, a quotient equality over represented terms, or a host evaluator.*

Proof. Begin with the conversion coordinates. The carrier separates the bounded comparison row B from the finished-normal side-condition row F, so a consumer route that asks for conversion evidence must pass through both rows.

The refusal row R records the missing endpoints. It prevents the bounded comparison route from being reread as a global closed beta-conversion decider, fuel-erased normalisation theorem, quotient equality, or host evaluator.

Transport and replay do not add data. The rows H and C preserve the same displayed comparison and side-condition routes, while P and N only record provenance and local naming. Thus the conversion read stays inside the bounded witness packet. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessBoundedConversionNonescape`
↪ variant `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier bounded conversion nonescape`

Theorem 751.4 (MetaCIC decidability witness bounded normal-form split)**.** *An accepted* MetaCICDecidabilityWitn *packet may expose bounded checker output and normal-form endpoint rows only through its displayed candidate row T, structural comparison row S, fuel and window row B, normal-form side-condition row F, refusal row R, componentwise transport H, displayed replay C, provenance P, and*

local $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *row* N*. A later confluence or decidability consumer may inspect those rows as bounded normal-form evidence, but the packet does not expose global closed-term conversion decidability, unrestricted subject reduction, quotient syntax equality, or consistency.*

Proof. Project the carrier of Definition 751.1. The checker candidate, structural comparison, bounded fuel and window, normal-form side condition, refusal, transport, replay, provenance, and naming rows are exactly $T, S, B, F, R, H, C, P, N$.

Invoke the NameCert obligations of Definition 751.2. They identify the admissible checker and normal-form reads with the displayed rows and leave no additional coordinate for a global conversion decider or host syntax equality.

Use the bounded-conversion non-escape theorem Definition 751.3. It confines conversion evidence to B together with F, while R records the missing global endpoints. Thus a later consumer may inspect bounded normal-form evidence through the named rows only, and cannot reread it as unrestricted conversion decidability, subject reduction, quotient syntax equality, or consistency. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessBoundedNormalFormSplit`
↪ variant `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitness bounded normal form split`

Theorem 751.5 (MetaCIC decidability witness typing-exactness route)**.** *Every accepted* MetaCICDecidabilityWitne *packet routes typing-checker exactness through the inferTypeCtx row* T *before any consumer may combine structural equality* S*, bounded beta comparison* B*, and finished-normal row* F*. A request that skips* T *cannot be read as* HasType *exactness or as closed conversion evidence for the packet.*

Proof. Project the carrier of Definition 751.1. The row T is the unique displayed coordinate that names inferTypeCtx soundness and completeness for the encoded HasType relation, while S, B, F record structural equality, bounded comparison, and finished-normal side conditions.

Apply the NameCert obligations of Definition 751.2. They bind typing-checker exactness to T and leave S, B, F as later consumer rows, so the structural and bounded-normal data cannot be promoted into a typing-exactness witness without first exposing T.

Use the bounded normal-form split of Definition 751.4. It permits bounded normal-form inspection only through the displayed rows and keeps the refusal row R visible. Together with Definition 751.3, this excludes a global ClosedAt decision theorem, unrestricted beta conversion, quotient syntax equality, and evaluator rows.

Thus any route skipping T lacks the only row that carries inferTypeCtx exactness. Such a route may inspect bounded comparison evidence after the displayed side conditions, but it is not a HasType exactness route and does not close conversion for the packet. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier typing exactness route`

Theorem 751.6 (MetaCIC decidability witness bounded-search exhaustion)**.** *For every accepted* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *packet* $W = (T, S, B, F, R, H, C, P, N)$*, every bounded conversion or typing decision requested inside the displayed finite depth bound is exhausted by the rows already visible in the packet: the syntax window in* T*, the bounded conversion ledger* B*, the normal-form split through* F*, the typing-exactness route through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *row* N*.*

The packet exports no unbounded strong-normalisation theorem, global confluence theorem, or ambient decision oracle.

Lean4 ✓ BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessBoundedSearchExhaustion
↪ variant BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitness bounded search exhaustion

Proof. Proceed by induction over the displayed finite depth bound in the carrier row. At depth zero, the syntax window in T and the structural row S expose only the immediate inferTypeCtx and checkSameTerm reads, so the decision is already inside the displayed packet.

For the successor depth, the induction hypothesis exhausts all smaller conversion and typing requests through the same finite rows. A bounded conversion request then reads B and the finished-normal side-condition row F; a typing request reads the inferTypeCtx exactness row T before it may combine S, B, F.

Apply Definition 751.4 and Definition 751.5. These results keep bounded normal-form evidence and typing exactness inside $T, S, B, F, R, H, C, P, N$.

The remaining structural rows preserve the same finite search. Transport through H, replay through C, provenance through P, and local naming through N do not add a decision oracle. The refusal row R therefore keeps unbounded strong normalisation, global confluence, and ambient decidability outside the exported surface. □

Theorem 751.7 (MetaCIC decidability witness confluence-handoff boundary)**.** *Every confluence or decidability consumer of an accepted* MetaCICDecidabilityWitness$^{\uparrow}$ *packet must read the bounded checker rows*

$$T, S, B, F, R, H, C, P, N$$

together with any confluence-normal-fragment sibling row as bounded evidence only. The handoff cannot export BetaParallelDiamondClosed*, full closed Church–Rosser, subject reduction, quotient syntax equality, kernel consistency, unrestricted beta-conversion decidability, or a host evaluator.*

Proof. Start with the bounded normal-form split of Definition 751.4. It confines normal-form inspection to the displayed typing, same-term, bounded beta, finished-normal, refusal, transport, replay, provenance, and local naming rows.

Apply the typing-exactness route of Definition 751.5. A confluence or decidability consumer cannot combine S, B, F as checker evidence before T exposes the inferTypeCtx exactness row.

Finally use bounded-search exhaustion, Definition 751.6. It exhausts all requested finite conversion and typing reads inside the displayed depth bound. A confluence-normal-fragment sibling row may therefore be consumed only as another bounded row beside $T, S, B, F, R, H, C, P, N$, with the global diamond, Church–Rosser, subject-reduction, quotient-equality, consistency, and evaluator endpoints still absent. □

Lean4 ✓ BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessConfluenceHandoffBoundary

Definition 751.8 (MetaCIC parallel-diamond fuel window)**.** *A* MetaCICDecidabilityWitness$^{\uparrow}$ *parallel-diamond fuel-window packet is a finite* BHist *row*

$$V = (T, S, B, F, R, H, C, P, N, D)$$

where $T, S, B, F, R, H, C, P, N$ are the displayed rows of the carrier in Definition 751.1, and D is the confluence normal-fragment sibling row from chapter 737. The row D is read only inside the same bounded fuel window as B and F: it records bounded parallel-diamond evidence for the displayed normal fragments, not a global confluence theorem, unrestricted beta-conversion decision, quotient syntax equality, consistency theorem, or host evaluator.

Lean4 (definition) BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicParallelDiamondFuelWindow

Theorem 751.9 (MetaCIC parallel-diamond fuel monotonicity)**.** *Let $V = (T, S, B, F, R, H, C, P, N, D)$ be an accepted* MetaCICDecidability Witness$^{\uparrow}$ *parallel-diamond fuel-window packet. Increasing the displayed bounded fuel window preserves only the bounded-normal evidence already routed through B, F, D together with the typing, same-term, refusal, transport, replay, provenance, and naming rows T, S, R, H, C, P, N. The enlarged window does not export global* ClosedAt *decidability, unrestricted beta conversion, quotient syntax equality, consistency, fuel-erased normalisation, or a host evaluator.*

Proof. Start with the TasteGate single-carrier marker recorded for the chapter. It fixes the MetaCIC decidability witness boundary at the displayed carrier rows $T, S, B, F, R, H, C, P, N$, so adding fuel cannot introduce a new decision coordinate.

Use the bounded-search exhaustion theorem, Definition 751.6. It says that each request inside a finite depth bound is exhausted by the visible syntax window, bounded conversion ledger, normal-form split, typing-exactness route, transport, replay, provenance, and local naming rows.

Now enlarge the finite fuel window. The bounded-conversion non-escape theorem, Definition 751.3, keeps conversion evidence inside B together with F, while the sibling row D contributes only bounded confluence-normal-fragment evidence at the displayed window.

Transport through H and replay through C preserve the same row inventory, and P, N record provenance and naming. The refusal row R therefore remains active after the enlargement, preventing the bounded parallel-diamond evidence from being reread as global ClosedAt decidability, unrestricted beta conversion, quotient syntax equality, consistency, fuel-erased normalisation, or a host evaluator. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicParallelDiamondFuelMonotonicity`

Theorem 751.10 (MetaCIC decidability witness fuel-window exhaustion)**.** *Let $V = (T, S, B, F, R, H, C, P, N, D)$ be an accepted* MetaCICDecidability Witness$^{\uparrow}$ *parallel-diamond fuel-window packet. Every bounded conversion, typing, or candidate-mediated parallel-diamond request inside the displayed finite fuel window is exhausted by the rows $T, S, B, F, D, R, H, C, P, N$: typing exactness routes through T, structural same-term comparison through S, bounded beta comparison through B, finished-normal side conditions through F, bounded parallel-diamond evidence through D, refusal through R, transport through H, replay through C, provenance through P, and local naming through N. The packet exports no unbounded strong normalization, global confluence, subject reduction, quotient syntax equality, consistency, fuel-erased normalization, or host evaluator.*

Proof. Start with the TasteGate single-carrier alignment marker recorded after the NameCert obligations. It fixes the accepted MetaCIC decidability witness at the displayed carrier rows, so a bounded request has no hidden coordinate outside $T, S, B, F, R, H, C, P, N$ and the parallel-diamond row D.

Apply bounded-search exhaustion, Definition 751.6. It accounts for every finite conversion or typing read by the visible syntax window, bounded conversion row, finished-normal row, typing route, refusal row, transport, replay, provenance, and local naming.

Use bounded-conversion nonescape, Definition 751.3, and the typing-exactness route, Definition 751.5. These force conversion reads through B, F and typing reads through T before any candidate-mediated request can consume the packet.

Finally invoke the parallel-diamond fuel monotonicity theorem, Definition 751.9. The row D contributes only bounded normal-fragment evidence inside the displayed fuel window, and the refusal row R prevents that evidence from being read as unbounded strong normalization, global confluence, subject reduction, quotient syntax equality, consistency, fuel-erased normalization, or a host evaluator. □

Lean4 ✓ BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessFuelWindowExhaustion

Theorem 751.11 (MetaCIC decidability witness obligation-closure package)**.** *For the carrier of Definition 751.1, the* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *obligation surface is closed by the carrier row, the NameCert obligations, bounded-conversion nonescape, bounded normal-form split, and typing-exactness route. Equivalently, every accepted packet* $W = (T, S, B, F, R, H, C, P, N)$ *exposes exactly inferTypeCtx exactness through* T*, checkSameTerm structural equality through* S*, bounded beta comparison through* B*, finished-normal side conditions through* F*, refusal rows through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *row* N*.*

Proof. Carrier admission supplies the displayed tuple $T, S, B, F, R, H, C, P, N$, and the existing TasteGate marker above fixes the AI chapter boundary at that single carrier. The NameCert obligations of Definition 751.2 then assign typing exactness, structural equality, bounded comparison, finished-normal side conditions, refusal reads, transport, replay, provenance, and naming to those rows.

The bounded-conversion theorem Definition 751.3 blocks escape from B and F to global conversion, fuel-erased normalisation, quotient equality, or a host evaluator. The bounded normal-form split of Definition 751.4 keeps every bounded checker and normal-form read inside $T, S, B, F, R, H, C, P, N$.

Finally, the typing-exactness route of Definition 751.5 forces every HasType exactness read through T before structural or bounded-normal evidence is combined. These cited rows cover every obligation named by the carrier and leave no additional coordinate for a separate decidability, transport, replay, provenance, or local naming obligation. □

Lean4 ✓ BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier obligation closure package

Theorem 751.12 (MetaCIC decidability witness scope assignment)**.** *For every accepted* MetaCICDecidabilityWitnes *packet* $W = (T, S, B, F, R, H, C, P, N)$*, the rows are scoped as follows:* T *is the* inferTypeCtx *exactness row,* S *is the* checkSameTerm *structural-equality row,* B *is the bounded beta-comparison row,* F *is the finished-normal side-condition row,* R *is the refusal-boundary row,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is package provenance, and* N *is local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *naming. No consumer may read these rows as* ClosedAt *decidability, unrestricted beta-conversion, quotient syntax equality, consistency, fuel-erased normalisation, or a host evaluator.*

Proof. Project the carrier of Definition 751.1. The displayed tuple is exactly $T, S, B, F, R, H, C, P, N$, so every scoped assignment must be read from one of those coordinates.

Apply the NameCert obligations of Definition 751.2. They assign typing exactness to T, structural equality to S, bounded comparison to B, finished-normal side conditions to F, refusal boundaries to R, transport to H, replay to C, provenance to P, and naming to N.

Use bounded-conversion non-escape, Definition 751.3, together with the bounded normal-form split of Definition 751.4. These keep conversion reads inside B, F and keep bounded normal-form evidence inside the displayed packet.

Finally apply the typing-exactness route of Definition 751.5 and the TasteGate single-carrier alignment marker already recorded for the chapter. The accepted packet therefore has one finite scope assignment for the nine rows, with no coordinate from which a consumer can extract ClosedAt decidability, unrestricted beta-conversion, quotient syntax equality, consistency, or a host evaluator. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier scope assignment`

Theorem 751.13 (MetaCIC decidability witness candidate-normalization scope)**.** *Every candidate-normalization consumer of an accepted* MetaCICDecidabilityWitness$^{\uparrow}$ *packet* $W = (T, S, B, F, R, H, C, P, N)$ *may inspect only the typing exactness row* T*, same-term row* S*, bounded beta row* B*, finished-normal side condition* F*, refusal row* R*, and structural rows* H, C, P, N*. The packet keeps strong normalization, global confluence, subject reduction, evaluator equality, quotient syntax equality, and consistency outside the candidate-normalization scope.*

Proof. Start with the TasteGate single-carrier alignment marker recorded after the NameCert obligations. It fixes the packet boundary at the carrier $W = (T, S, B, F, R, H, C, P, N)$, so a candidate-normalization consumer has no row beyond those nine displayed coordinates.

Apply the bounded normal-form split of Definition 751.4. It allows bounded normal-form inspection only through the typing, same-term, bounded beta, finished-normal, refusal, and structural rows already visible in the carrier.

Use the typing-exactness route of Definition 751.5 and the bounded-search exhaustion theorem Definition 751.6. These force typing reads through T and bounded conversion reads through B, F before any finite candidate-normalization request can be exhausted.

Finally apply the scope assignment of Definition 751.12. It identifies S, R, H, C, P, N as structural equality, refusal, transport, replay, provenance, and local naming rows, and it leaves no coordinate for strong normalization, global confluence, subject reduction, evaluator equality, quotient syntax equality, or consistency. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier candidate normalization scope`

Theorem 751.14 (MetaCIC decidability witness scope routes)**.** *For every accepted* MetaCICDecidabilityWitness$^{\uparrow}$ *packet* $W = (T, S, B, F, R, H, C, P, N)$*, each consumer route is assigned to the displayed row that owns it: typing-checker exactness routes through* T*, structural same-term equality routes through* S*, bounded conversion routes through* B *together with the finished-normal side condition* F*, refusal routes through* R*, and transport, replay, provenance, and local naming route through* H, C, P, N*. No route exports* ClosedAt *decidability, unrestricted beta conversion, quotient syntax equality, consistency, fuel-erased normalisation, or a host evaluator.*

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessCarrier scope routes`

Proof. Project the carrier of Definition 751.1. The displayed tuple has exactly the rows $T, S, B, F, R, H, C, P, N$, so every accepted route must be assigned to one of those coordinates.

Apply the NameCert obligations of Definition 751.2 and the scope assignment of Definition 751.12. These identify T with typing exactness, S with same-term structural equality, B with bounded beta comparison, F with the finished-normal side condition, R with refusal, and H, C, P, N with transport, replay, provenance, and local naming.

Use bounded-conversion non-escape, Definition 751.3, together with the bounded normal-form split of Definition 751.4. Conversion can therefore be read only through B and F, and bounded normal-form evidence remains inside the displayed packet.

Finally invoke the typing-exactness route of Definition 751.5 and the candidate-normalization scope of Definition 751.13. They force typing reads through T and keep candidate-normalization consumers inside the same finite rows. Since the carrier has no additional coordinate, none of

the listed routes can export ClosedAt decidability, unrestricted beta conversion, quotient syntax equality, consistency, fuel-erased normalisation, or a host evaluator. □

Theorem 751.15 (MetaCIC decidability witness formal-target ledger)**.** *Any later Lean formal target for the* MetaCICDecidabilityWitness$^{\uparrow}$ *chapter must read typing exactness through T, same-term comparison through S, bounded beta comparison through B and F, refusal through R, and structural movement through H, C, P, N. It may not claim* ClosedAt *decidability, unrestricted conversion, quotient syntax equality, consistency, fuel-erased normalisation, or host evaluation.*

Proof. Start with the NameCert obligations of Definition 751.2. They make the accepted witness a finite carrier $W = (T, S, B, F, R, H, C, P, N)$, so any formal target that belongs to this chapter must read from one of those rows.

Apply bounded-search exhaustion, Definition 751.6, together with bounded-conversion non-escape and the bounded normal-form split. These route bounded beta comparison through B and F and keep finished-normal evidence inside the displayed finite packet.

Use the typing-exactness route and scope routes, Definition 751.5 and Definition 751.14. They force typing reads through T, same-term comparison through S, refusal through R, and transport, replay, provenance, and local naming through H, C, P, N.

Since those cited rows exhaust the carrier, no later formal target licensed by this chapter can export ClosedAt decidability, unrestricted conversion, quotient syntax equality, consistency, fuel-erased normalisation, or host evaluation. Such endpoints would require a separate carrier coordinate not present in the accepted packet. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessFormalTargetLedger`

Theorem 751.16 (MetaCIC decidability witness scoped closure package)**.** *For every accepted* MetaCICDecidabilityWitness$^{\uparrow}$ *packet $W = (T, S, B, F, R, H, C, P, N)$, the scope routes assign all nine rows to one finite MetaCIC decidability witness: T to typing exactness, S to same-term structural equality, B to bounded beta comparison, F to finished-normal side conditions, R to refusal, H to transport, C to replay, P to provenance, and N to local naming. The existing TasteGate marker*

BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessTasteGate_single_carrier_alignment

fixes the AI chapter boundary at that single carrier surface.

Thus the scoped closure package is the union of the NameCert obligations, bounded-conversion non-escape, bounded normal-form split, typing-exactness route, candidate-normalization scope, scope routes, and TasteGate single-carrier alignment. It exports no standalone ClosedAt *decision theorem, unrestricted closed beta-conversion decider, quotient syntax equality, consistency theorem, fuel-erased normalisation theorem, or host evaluator.*

Proof. Begin with the TasteGate marker already recorded after Definition 751.2. It aligns the chapter with the single displayed carrier $W = (T, S, B, F, R, H, C, P, N)$, so any scoped package must be read through those rows.

The NameCert obligations and scope routes assign the rows. They place typing exactness in T, same-term equality in S, bounded beta comparison in B, finished-normal side conditions in F, refusal in R, and the structural transport, replay, provenance, and local naming data in H, C, P, N.

Use bounded-conversion non-escape and the bounded normal-form split to confine conversion and normal-form reads to B, F together with the visible packet rows. Then use the typing-exactness route and candidate-normalization scope to force typing and finite candidate-normalization consumers through $T, S, B, F, R, H, C, P, N$.

These cited routes exhaust the accepted carrier surface fixed by TasteGate. Since the carrier has no further coordinate, the scoped closure package cannot export any of the refused global decidability, conversion, equality, consistency, normalisation, or evaluator objects. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessScopedClosurePackage`

Theorem 751.17 (MetaCIC decidability witness public export)**.** *Every public consumer of an accepted* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *packet must read the same typing, same-term, bounded-conversion, finished-normal, refusal, transport, replay, provenance, and local naming rows before treating the packet as a decidability witness. Thus a public export over* $W = (T, S, B, F, R, H, C, P, N)$ *is valid only when* T *supplies typing exactness,* S *supplies structural same-term equality,* B *and* F *supply bounded conversion and finished-normal inspection,* R *supplies refusal, and* H, C, P, N *supply componentwise transport, displayed replay, provenance, and local naming. It exports no standalone* $\mathsf{ClosedAt}$ *decision theorem, unrestricted beta-conversion decider, quotient syntax equality, consistency theorem, fuel-erased normalisation theorem, or host evaluator.*

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessPublicExport`

Proof. Begin with the TasteGate single-carrier marker already recorded after Definition 751.2. It fixes the public packet to the displayed carrier $W = (T, S, B, F, R, H, C, P, N)$.

The bounded-conversion non-escape theorem and bounded normal-form split force conversion and finished-normal reads through B, F together with the visible refusal and structural rows. The typing-exactness route forces typing reads through T, while bounded-search exhaustion keeps finite candidate search inside the displayed packet.

The scope assignment, candidate-normalization scope, scope routes, and scoped-closure package then identify the remaining public rows. They place same-term checking in S, refusal in R, and transport, replay, provenance, and local naming in H, C, P, N.

These cited surfaces exhaust the accepted public export. Since no carrier coordinate names global decidability, unrestricted conversion, quotient equality, consistency, fuel-erased normalisation, or a host evaluator, none of those objects can be exported by a public consumer of the witness. □

Theorem 751.18 (MetaCIC decidability witness bridge-facing surface)**.** *Every bridge-facing consumer of an accepted* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *packet* $W = (T, S, B, F, R, H, C, P, N)$ *must expose the same public rows already bound in this chapter: typing exactness through* T*, same-term checking through* S*, bounded conversion through* B*, finished-normal exposure through* F*, refusal rows through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *naming through* N*. The surface exports no subject-reduction theorem, full normalization theorem, evaluator equality, quotient term equality, or kernel consistency theorem.*

Proof. Start with the public export theorem, Definition 751.17. It fixes every public consumer read at the displayed carrier $W = (T, S, B, F, R, H, C, P, N)$.

Use the scoped route theorems, Definition 751.14 and Definition 751.16. They assign typing exactness, same-term checking, bounded conversion, finished-normal exposure, refusal, transport, replay, provenance, and naming to those rows and to no hidden coordinate.

Finally apply the obligation-closure package, Definition 751.11. It repacks the carrier row, NameCert obligations, bounded-conversion nonescape, bounded normal-form split, and typing-exactness route into the same finite witness surface.

Thus any bridge-facing consumer can expose only the public rows already displayed in the chapter. Since the refusal and nonescape rows stay part of the package, the route cannot export subject reduction, full normalization, evaluator equality, quotient term equality, or kernel consistency. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.BridgeFacingSurface.MetacicDecidabilityWitnessBridgeE`

Theorem 751.19 (MetaCIC decidability witness bridge export)**.** *Any bridge export of an accepted* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *packet* $W = (T, S, B, F, R, H, C, P, N)$ *must expose typing exactness, same-term checking, bounded conversion, finished-normal inspection, refusal rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *naming before entering a bridge consumer. The export is therefore bounded and refusal-aware; it exports no subject-reduction theorem, full normalization theorem, quotient syntax equality, kernel consistency theorem, fuel-erased evaluator, or unrestricted closed beta-conversion decider.*

Proof. Start with the public export theorem, Definition 751.17. It requires every public consumer to read the displayed rows $T, S, B, F, R, H, C, P, N$ before treating the packet as a decidability witness.

Apply the bridge-facing surface theorem, Definition 751.18. It assigns typing exactness to T, same-term checking to S, bounded conversion to B, finished-normal exposure to F, refusal to R, componentwise hsame transport to H, displayed Cont replay to C, Pkg provenance to P, and local naming to N.

Use the scoped-closure package of Definition 751.16 together with the obligation-closure package of Definition 751.11. These packages ensure that the bridge export is still read from the same finite carrier and that the bounded-conversion, finished-normal, refusal, transport, replay, provenance, and naming obligations have already been exposed.

Thus a bridge consumer receives only the bounded, refusal-aware rows already displayed in the packet. Since no row names subject reduction, full normalization, quotient syntax equality, kernel consistency, a fuel-erased evaluator, or unrestricted closed beta conversion, none of those objects can be exported. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessBridgeExport`

Theorem 751.20 (MetaCIC decidability witness mature normal-form boundary)**.** *Any accepted* $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ *packet exposing typing exactness, bounded conversion, finished-normal rows, refusal rows, transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$ *naming remains bounded by the displayed fuel and scope rows of its carrier* $W = (T, S, B, F, R, H, C, P, N)$*. In particular, its mature bridge-facing surface cannot export standalone* $\mathsf{ClosedAt}$ *decidability, unrestricted beta-conversion decidability, quotient term equality, or kernel consistency; it exports only the finite typing, bounded-conversion, finished-normal, refusal, transport, replay, provenance, and naming rows already displayed by the packet.*

Proof. Start with the TasteGate single-carrier marker recorded for the chapter. It fixes the accepted witness at the displayed carrier $W = (T, S, B, F, R, H, C, P, N)$, so every normal-form or decidability read must be routed through those rows.

Use the bounded-conversion non-escape theorem and the bounded normal-form split. The former confines beta-comparison evidence to the displayed bounded row B together with the finished-normal side condition F, while the latter keeps normal-form evidence inside the same finite carrier and leaves the refusal row R visible.

Apply bounded-search exhaustion, scope assignment, and the scope-routes theorem. These identify typing exactness with T, same-term checking with S, bounded conversion with B, F, refusal with R, and structural movement with H, C, P, N under the displayed scope and fuel rows.

Finally use the bridge-facing surface and bridge-export theorems. They present the public rows to bridge consumers without adding a new coordinate. Since the carrier has no row for standalone ClosedAt decidability, unrestricted beta-conversion decidability, quotient term equality, or kernel consistency, none of those objects can be exported by the mature normal-form boundary. □

Lean4 ✓ `BEDC.Derived.MetacicDecidabilityWitnessUp.MetacicDecidabilityWitnessMatureNormalFormBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidabilityWitness}^{\uparrow}$ packet is generated from inferTypeCtx soundness and completeness, checkSameTerm structural equality, bounded beta comparison, finished-normal side conditions, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 751.1, Definition 751.2, Definition 751.3, Definition 751.4, Definition 751.5, Definition 751.11, Definition 751.12, Definition 751.14, Definition 751.16, Definition 751.17, Definition 751.18, Definition 751.19, and Definition 751.20 bind typing exactness, same-term checking, bounded conversion, finished-normal exposure, refusal rows, transport, replay, provenance, and local naming as one scoped finite NameCert witness with public, bridge-export, and mature normal-form boundary routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidabilityWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close standalone ClosedAt decidability, unrestricted closed beta-conversion decidability, fuel-erased normalisation, quotient equality over represented terms, Lean-side verification, or bridge checking.

Upgrade path: No theory-axis upgrade remains; Lean verification and bridge checking remain separate formal-axis work for BEDC.Derived.MetacicDecidabilityWitnessUp.

Chapter 752

A Concrete Naming Certificate for BoundedNormalEqualityChecker$^{\uparrow}$

The BoundedNormalEqualityChecker$^{\uparrow}$ packet records the normal-equality side of chapter 9379 as a finite naming certificate. It sits between the already-normal exclusion packet of chapter 732 and the closed-normal confluence fragment of chapter 737: the present packet names a bounded comparison surface, while those sibling packets keep false-proof exclusion and confluence scopes separate.

The checker also consumes the closed normal-form frontier of chapter 795 and the certificate-routing boundary of chapter 1108. Bounded equality checking is only a finite readback of accepted normal-form evidence through those MetaCIC rows; it is not a proof of global confluence or unrestricted normalization.

The packet is not a global beta-conversion oracle. It exposes bounded normalizer output rows, structural `Term.eq` readback, finished-normalization witnesses when they are supplied, closedness rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Definition 752.1 (BoundedNormalEqualityChecker carrier)**.** *A* BoundedNormalEqualityChecker$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, R, F, N_L, N_R, E, W, C, H, K, P, A)$$

with the following visible rows:

1. *L and R are the two closed MetaCIC input term rows being compared;*
2. *F is the shared unary fuel row for the bounded normalizer;*
3. *N_L and N_R are the bounded normalizer output rows obtained from the two input terms at fuel F;*
4. *E is the structural `Term.eq` readback row comparing N_L and N_R;*
5. *W stores the finished-normalization witnesses for the two outputs when such witnesses are part of the input surface;*
6. *C records the closedness rows for the inputs and for the named normal forms that a finished-soundness consumer may read;*
7. *H records componentwise* hsame *transports for the input terms, fuel, normalizer outputs, equality readback, finished witnesses, closedness rows, package row, and local naming row;*

8. *K records displayed* Cont *routes for running the bounded normalizer, reading the structural equality result, replaying finished witnesses, and transporting closedness rows;*
9. *P is the* Pkg *provenance over MetaCIC terms, bounded normalization,* `Term.eq`*, finished-normalization witnesses, closedness,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
10. *A is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a fuel-erased normalizer, no proof that every closed term reaches a finished normal form, no arbitrary closed-term beta-conversion decider, and no host equality oracle.

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerTasteGate`
`single carrier alignment`

Definition 752.2 (BoundedNormalEqualityChecker classifier)**.** *Two accepted* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *carriers are classified as the same bounded normal-equality packet exactly when their displayed rows compare componentwise. Their input rows agree by the closed MetaCIC term classifiers, their fuel rows agree by the unary fuel classifier, their bounded normalizer outputs agree by the displayed output rows, their equality decision agrees by* `Term.eq` *readback, their finished-normalization and closedness rows agree when those rows are present, and their* Pkg *provenance and local* NameCert *rows name the same packet.*

The classifier has no generator for comparing arbitrary beta-conversion classes, no generator for choosing a fuel that finishes all closed terms, and no generator for replacing structural `Term.eq` *by host equality.*

Lean4 (definition) `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerClassifier`

Definition 752.3 (BoundedNormalEqualityChecker route preorder)**.** *For accepted* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *checker packets, the route preorder is the finite relation generated by row transport and replay. A checker packet Q precedes a checker packet Q' when the normal false exclusion row, the bounded fuel row, the classifier row of Definition 752.2, and the confluence-normal fragment row of Q transport by the displayed* hsame *rows into the corresponding checker, normalfalseexclusion, and metacicconfluencenormalfragment rows of Q', and when the displayed* Cont *replay of Q' reads those transported rows before any consumer read.*

The preorder has no carrier-level equivalence generator beyond those concrete rows: it compares bounded normal-equality checker data only through the false-proof exclusion, fuel, classifier, confluence-normal fragment, transport, replay, provenance, and local naming rows already present in accepted packets.

Lean4 (definition) `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerRoutePreorder`

Theorem 752.4 (BoundedNormalEqualityChecker NameCert obligations)**.** *The carrier and classifier of Definition 752.1 and Definition 752.2, together with the route preorder of Definition 752.3, satisfy the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}^{\uparrow}}$ *obligation surface for bounded normal equality. Carrier admission reads only the two input term rows, shared fuel, bounded normalizer outputs, structural equality readback, optional finished-normalization witnesses, closedness rows,* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local naming row. Classifier stability is componentwise over exactly those rows, deterministic normal-form readback and consumer nonescape are monotone along the concrete checker route preorder, and ledger exactness rejects every consumer read outside them.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier namecert obligations`

Proof. For carrier admission, project the tuple Q. The rows $L, R, F, N_L, N_R, E, W, C$ are the entire computational surface, and the rows H, K, P, A are the entire structural replay surface.

For classifier stability, compare accepted packets row by row. Input terms, fuel, normalizer outputs, equality readback, finished witnesses, and closedness rows are transported only through the displayed hsame and Cont rows.

For ledger exactness, inspect a downstream read. It can project a bounded normalizer output, a `Term.eq` result, a supplied finished witness, a closedness row, or a structural replay row. Since no carrier coordinate names a global conversion oracle or a fuel-selection theorem, such a read cannot escape the bounded packet. □

Theorem 752.5 (BoundedNormalEqualityChecker finished soundness boundary)**.** *If an accepted* BoundedNormalEqualityChecker$^{\uparrow}$ *packet carries finished closed-normalization witnesses for both bounded normalizer outputs, then a true structural* `Term.eq` *readback may be consumed as equality of those named finished normal forms and as beta-convertibility between those named normal forms. The same readback does not prove that arbitrary closed terms finish at some fuel, and it does not decide unrestricted beta conversion.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier finished soundness boundary`

Proof. Read the bounded outputs N_L and N_R through the carrier, and read the structural equality result through E. The equality claim is tied to those two displayed normalizer outputs at the displayed fuel F.

When the finished rows in W and the closedness rows in C are present, the consumer may replay the finished-soundness route recorded in K. That route identifies the equality as a statement about the named finished normal forms, not about all possible reductions from arbitrary closed terms.

The boundary follows by coordinate exhaustion. The packet has no row proving fuel existence for every closed input and no row deciding beta conversion outside the displayed bounded outputs. □

Theorem 752.6 (BoundedNormalEqualityChecker deterministic normal-form readback)**.** *Let*

$$Q = (L, R, F, N_L, N_R, E, W, C, H, K, P, A)$$

and

$$Q' = (L', R', F', N'_L, N'_R, E', W', C', H', K', P', A')$$

be accepted BoundedNormalEqualityChecker$^{\uparrow}$ *packets with the same bounded normal input rows, the same shared fuel row, the same comparison ledger, and classifier-related output rows. Then the concrete packet exposes the same normal-form readback route: the displayed bounded normalizer outputs are read through the same N_L, N_R positions, the structural* `Term.eq` *result is read through the same ledger row E, and every finished-normal readback consumer factors through the same carrier rows $L, R, F, N_L, N_R, E, W, C, H, K, P, A$. The conclusion is internal to* BoundedNormalEqualityChecker$^{\uparrow}$ *and does not pass through an abstract source-equivalence transport.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier deterministic normal form readback`

Proof. Project both accepted packets through Definition 752.1. The carrier rows fix the two input terms, shared fuel, bounded normalizer outputs, structural readback row, finished witnesses, closedness rows, transports, replay, provenance, and local naming data.

Use the classifier of Definition 752.2 and the stability row of Definition 752.9. The hypothesis that the inputs, fuel, comparison ledger, and output rows are classifier-related forces the displayed readback route to compare componentwise through the same finite coordinates.

The finished soundness boundary of Definition 752.5 allows a consumer to read equality only for the named finished normal forms. Finally, ledger exactness from Definition 752.10 prevents the consumer from replacing the concrete route by host equality, quotient equality, fuel erasure, hidden termination, or an abstract source-equivalence transport. □

Theorem 752.7 (BoundedNormalEqualityChecker fuel monotonicity boundary)**.** *Let*

$$Q = (L, R, F, N_L, N_R, E, W, C, H, K, P, A)$$

and

$$Q^+ = (L, R, F^+, N_L^+, N_R^+, E^+, W^+, C^+, H^+, K^+, P^+, A^+)$$

be accepted BoundedNormalEqualityChecker$^\uparrow$ *packets whose displayed inputs are the same and whose second fuel row F^+ extends the displayed unary fuel row F. Increasing the displayed fuel may extend the bounded normalizer output route only through the same input rows, structural equality readback row, finished witness rows, closedness rows, componentwise transports, displayed replay, provenance, and local naming rows. It does not create a fuel-erased normalizer, a global termination theorem, an arbitrary beta-conversion decider, or a host equality oracle.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier fuel monotonicity`

Proof. Project both packets through Definition 752.1. The only rows that differ are the displayed fuel, bounded output, equality readback, finished witness, closedness, transport, replay, provenance, and local naming rows; the input rows remain L and R.

Use the classifier of Definition 752.2. It compares fuel and output data componentwise, so a larger fuel row can only expose another bounded normalizer route over the same two input rows.

Apply the finished soundness boundary of Definition 752.5. Finished witnesses, when present, certify only the named outputs at the displayed fuel and do not prove that every closed term finishes at some fuel.

Apply deterministic normal-form readback from Definition 752.6 and ledger exactness from Definition 752.10. The readback and every consumer route stay inside the displayed equality, closedness, transport, replay, provenance, and local naming rows, so fuel growth cannot erase the bound or become a host equality oracle. □

Theorem 752.8 (BoundedNormalEqualityChecker consumer non-escape)**.** *Every consumer of an accepted* BoundedNormalEqualityChecker$^\uparrow$ *packet reads only the finite rows*

$$L, R, F, N_L, N_R, E, W, C, H, K, P, A.$$

The consumer may use the bounded normalizer outputs, the structural `Term.eq` *result, and the finished-normalization bridge when its witness rows are displayed, but it receives no unrestricted beta-conversion decision procedure, no hidden normalizer termination theorem, no quotient equality of terms, and no host equality oracle.*

Proof. Start from the carrier definition. The displayed tuple lists all computational, structural, provenance, and local naming rows available to a consumer.

Use the NameCert obligations to force each read through the same finite packet. Normalizer outputs pass through N_L and N_R, equality readback passes through E, and conditional finished soundness passes through W, C, H, K, P, A.

Apply the finished soundness boundary. It permits equality and beta-convertibility only for the named finished normal forms supplied by the packet, while the absent fuel-existence and global conversion coordinates block every stronger read. □

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier consumer nonescape`

Theorem 752.9 (BoundedNormalEqualityChecker classifier stability row)**.** *For accepted* BoundedNormalEqualityCh *packets*

$$Q = (L, R, F, N_L, N_R, E, W, C, H, K, P, A)$$

and

$$Q' = (L', R', F', N'_L, N'_R, E', W', C', H', K', P', A'),$$

the classifier is stable exactly componentwise over the displayed rows $L, R, F, N_L, N_R, E, W, C, H, K, P, A$*. In particular, the comparison transports the two input term rows, shared fuel row, bounded normalizer outputs, structural* `Term.eq` *readback, supplied finished witnesses, closedness rows, componentwise* hsame *data, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}^{\uparrow}}$ *row. It has no additional coordinate for host equality, fuel erasure, or global beta conversion.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier classifier stability row`

Proof. Project both accepted carriers through Definition 752.1. The classifier of Definition 752.2 lists the same coordinates and compares them row by row.

The NameCert obligations of Definition 752.4 force each computational comparison through the displayed bounded normalizer, structural readback, witness, and closedness rows. The consumer non-escape result of Definition 752.8 then rules out any stabilizing coordinate beyond the finite tuple. □

Theorem 752.10 (BoundedNormalEqualityChecker ledger exactness row)**.** *Every consumer read from an accepted* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *packet is exhausted by the bounded normalizer outputs* N_L, N_R*, the structural* `Term.eq` *row* E*, the supplied finished witness row* W*, the closedness row* C*, the componentwise* hsame *row* H*, the displayed* Cont *row* K*, the* Pkg *row* P*, and the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}^{\uparrow}}$ *row* A*. The ledger therefore contains no global beta-conversion procedure, no hidden termination theorem, no quotient equality of terms, and no host equality oracle.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier namecert ledger read boundary`

Proof. Read the finite tuple in the order fixed by the carrier definition. The only normalization outputs are N_L and N_R, and the only equality readback is the structural `Term.eq` row E.

Finished soundness is conditional on the displayed rows W and C and is bounded by Definition 752.5. Transport, replay, provenance, and naming then pass through H, K, P, A.

Finally apply consumer non-escape from Definition 752.8. Since every consumer read is a projection from the displayed ledger, no global conversion, hidden termination, quotient equality, or host equality row can be read. □

Theorem 752.11 (BoundedNormalEqualityChecker obligation package)**.** *The* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *obligation surface is the package formed by carrier admission, classifier stability, finished-soundness boundary, consumer non-escape, and ledger exactness over the finite carrier rows* $L, R, F, N_L, N_R, E, W, C, H, K, P$*. This package is the whole local* $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}^{\uparrow}}$ *surface for bounded normal equality, and it does not claim unrestricted beta conversion, hidden normalizer termination, quotient equality, or host equality.*

Proof. Carrier admission and the local NameCert rows are supplied by Definition 752.4. Classifier stability is the rowwise comparison of Definition 752.9.

The finished-soundness boundary is exactly Definition 752.5, and consumer non-escape is Definition 752.8.

The remaining ledger clause is Definition 752.10. These five clauses account for every displayed carrier coordinate, so the package contains the obligation-closure surface and no stronger global conversion claim. □

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier obligation package`

Theorem 752.12 (BoundedNormalEqualityChecker package-read public consumer)**.** *Every public consumer read that starts from the bounded checker package row factors through the obligation package. The package-read row is produced only by the displayed transport and route rows* H, K*, and the public read then continues through the local naming row* A*; the consumer retains the package provenance and NameCert package signatures, but gains no hidden conversion or termination row.*

Proof. Apply the obligation package to the accepted carrier and the displayed $\mathsf{Cont}(H, K)$ package route. This supplies the unary package-read row, the same transport-route replay, and the two package signatures.

The public read is then a further displayed continuation from the package-read row to the local naming row. Since the obligation package already accounts for the available rows, the public consumer can only read the named bounded checker surface and cannot introduce a fuel-erased normalizer or host equality oracle. □

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerPackageRead public consumer`

Theorem 752.13 (BoundedNormalEqualityChecker scoped closure)**.** *The carrier, classifier, route preorder, NameCert obligations, finished-soundness boundary, deterministic readback, non-escape theorem, classifier-stability row, ledger-exactness row, obligation package, and package-read public consumer jointly close the bounded normal-equality checker scope. The closed scope is exactly the finite surface over closed MetaCIC term rows, unary fuel, bounded normalizer output, structural* `Term.eq`*, finished witnesses, closedness rows,* hsame*,* Cont*,* Pkg*, and* NameCert *carried by* $L, R, F, N_L, N_R, E, W, C, H, K, P, A$*. It exports no unrestricted beta conversion, hidden normalizer termination, quotient equality of terms, or host equality oracle.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerCarrier scope closure`

Proof. Start from the carrier definition, classifier, and route preorder. These fix the displayed rows $L, R, F, N_L, N_R, E, W, C, H, K, P, A$ and the allowed rowwise route between accepted packets.

The local obligations then account for the analytic and audit rows. The NameCert obligation theorem supplies carrier admission, finished-soundness supplies the bounded endpoint boundary, and deterministic readback identifies structural `Term.eq` as the only equality read.

Non-escape, classifier stability, and ledger exactness rule out hidden routes. They keep every consumer inside the closed MetaCIC term rows, unary fuel, bounded normalizer output, finished witnesses, closedness rows, transport, replay, provenance, and local naming.

Finally apply the obligation package. It gathers the listed clauses as one finite scope, so the bounded normal-equality checker is scoped-closed over the displayed kernel rows and does not export any stronger conversion or equality principle. □

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerScopeClosure`

Theorem 752.14 (BoundedNormalEqualityChecker public export)**.** *Every accepted* BoundedNormalEqualityChecke *packet exports exactly the displayed bounded normal-equality checker surface: two closed term rows, the fuel row, bounded normalizer outputs, structural equality readback, finished-normalization witness rows, componentwise* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. No public consumer receives unrestricted beta-conversion, hidden normalizer termination, quotient equality of terms, or a host equality oracle from this packet.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerPublicExport`

Proof. Use Definition 752.13. It already binds the carrier, classifier, route preorder, NameCert obligations, finished-soundness boundary, deterministic readback, non-escape theorem, classifier-stability row, ledger-exactness row, and obligation package as one closed finite scope.

The public export reads that closed scope without adding a new coordinate. The two closed term rows, fuel row, bounded normalizer outputs, structural equality readback, finished-normalization witnesses, transports, replay, provenance, and local naming are exactly the rows exposed by the scope-closure theorem.

Since the carrier has no row for unrestricted conversion, hidden termination, quotient equality, or host equality, the exported public surface is precisely the displayed bounded checker packet and nothing stronger. □

Theorem 752.15 (BoundedNormalEqualityChecker bridge-facing schema)**.** *The public* BoundedNormalEqualityChe *surface exports exactly the bounded closed-term rows, finished-normal readback, fuel monotonicity, classifier stability, ledger exactness, and scope closure already bound by the carrier, classifier, NameCert obligations, finished-soundness boundary, deterministic readback, non-escape theorem, classifier-stability row, ledger-exactness row, obligation package, scope-closure theorem, and public export theorem. A bridge-facing consumer may therefore read the displayed MetaCIC sibling rows, unary fuel, bounded normalizer outputs, finished witnesses, closedness data, componentwise transport, replay, provenance, and local naming, but it receives no unrestricted beta conversion, hidden normalizer termination, quotient equality of terms, or host equality oracle.*

Lean4 ✓ `BEDC.Derived.BoundedNormalEqualityCheckerUp.BoundedNormalEqualityCheckerBridgeFacingSchema`

Proof. Start with the public export theorem. It exposes the same finite checker packet already closed by the scope-closure theorem and adds no public coordinate beyond $L, R, F, N_L, N_R, E, W, C, H, K, P, A$.

The bridge-facing schema reads the public packet through the earlier bounded rows. Finished-soundness and deterministic readback supply the finished-normal endpoint and structural equality read, while non-escape prevents that read from becoming unrestricted beta conversion.

Classifier stability and ledger exactness preserve the audit surface under accepted packet comparison. Together with the obligation package, they keep fuel, bounded normalizer output, closedness rows, transport, replay, provenance, and local naming in the displayed finite scope.

Thus the bridge-facing surface is exactly the public bounded normal-equality checker schema. Since none of the cited rows names hidden termination, quotient equality, host equality, or unrestricted conversion, no bridge-facing consumer receives those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}\uparrow}$.

Constructive story: A $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ packet is generated from two closed MetaCIC term rows, a bounded fuel row, bounded normalizer outputs, structural equality readback, finished-normalization witness rows, componentwise transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 752.1, Definition 752.2, Definition 752.4, Definition 752.5, Definition 752.6, Definition 752.8, Definition 752.9, Definition 752.10, Definition 752.11, Definition 752.13, Definition 752.14, and Definition 752.15 bind a mature paper-axis treatment of the public bounded normal-equality checker surface; verification status and checked bridge status remain independent.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalEqualityChecker}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close unrestricted beta-conversion, hidden normalizer termination, quotient equality of terms, or host equality.

Upgrade path: The paper-axis treatment is mature; Lean verification, axiom-purity certification, and a checked standard bridge remain independent verification-axis work.

Chapter 753

A Concrete Naming Certificate for $\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}$

The $\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}$ packet turns the typed-example audit map of chapter 9382 into a finite seal that can be read by downstream MetaCIC checker consumers. It sits between the bounded decidability boundary of chapter 749 and the bounded normal-equality checker of chapter 752: those chapters expose the finite checker and normal-comparison rows, while the present packet seals the worked gallery rows that exercise them.

Definition 753.1 (MetaCIC TypedExamples seal carrier)**.** *A* $\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, T, C, B, D, Q, R, H, K, P, N)$$

with the following displayed rows:

1. *E is the finite worked-example row selected from the typed-example audit map: Church encodings, identity rows, applications, projections, eliminator examples, constants, Pi rows, and sort-context rows;*
2. *T is the displayed typing-checker row for the examples that are accepted by the finite gallery;*
3. *C is the closedness and context-formation row consumed by those checker reads;*
4. *B is the selected beta and readback row for examples whose evaluation behavior is part of the worked gallery;*
5. *D is the bounded decidability row imported from the* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *surface;*
6. *Q is the bounded normal-equality checker row imported from the* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *surface;*
7. *R is the finite seal ledger recording which gallery families are accepted, which computations are named, and which checker or normal-equality rows they are allowed to read;*
8. *H records componentwise* hsame *transports across $E, T, C, B, D, Q, R, P, N$;*
9. *K records the displayed* Cont *routes for replaying gallery entry reads, checker reads, closedness reads, beta reads, bounded decidability reads, normal-equality reads, and seal-ledger reads;*
10. *P is the* Pkg *provenance over chapter 9382, chapter 749, and chapter 752;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary MetaCIC term, no global subject-reduction theorem, no strong normalization theorem, no global conversion decider, no proof-erasure correctness theorem, and no host evaluator.

Theorem 753.2 (MetaCIC TypedExamples seal NameCert obligations)**.** *For the carrier of Definition 753.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for* $S = (E, T, C, B, D, Q, R, H, K, P, N)$*, gallery-entry visibility through* E*, typing-checker visibility through* T*, closedness and context visibility through* C*, selected computation visibility through* B*, bounded decidability visibility through* D*, bounded normal-equality visibility through* Q*, finite seal-ledger exactness through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The rows E, T, C, B, D, Q, R are the entire worked-example surface, and the rows H, K, P, N are the entire structural surface available to a consumer.

The gallery-entry obligation reads E only for the families selected by chapter 9382. The checker obligation reads T under the closedness and context row C, while the computation obligation reads B only for the examples whose beta or readback behavior is named in the finite ledger.

The bounded decidability and normal-equality obligations read D and Q through the sibling packets of chapter 749 and chapter 752. The seal ledger R records which of those sibling rows each worked example may consume.

Finally, H, K, P, N transport, replay, package, and name only the displayed rows. Since the carrier contains no further coordinate, the local NameCert surface is exhausted by the listed obligations. □

Theorem 753.3 (MetaCIC TypedExamples seal non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}$ *packet reads only the finite rows*

$$E, T, C, B, D, Q, R, H, K, P, N.$$

The consumer may use those rows as a worked-example seal for the typed-example gallery and may route selected checker or bounded normal-equality reads through the named sibling packets. It receives no theorem about all represented MetaCIC terms, no quotient equality of terms, no unrestricted beta-conversion decision procedure, no selected normalization endpoint, no proof-erased compiler theorem, and no host evaluator.

Proof. Begin with the carrier definition. The tuple lists all gallery, checker, closedness, computation, bounded decidability, normal-equality, ledger, transport, route, provenance, and naming rows exposed by the packet.

Any checker or normal-equality read must pass through the finite seal ledger R. That ledger points to the bounded decidability and bounded normal-equality sibling surfaces, so the consumer cannot replace the selected gallery rows by an arbitrary MetaCIC term or a global conversion oracle.

The structural rows H, K, P, N preserve componentwise agreement, replay the same finite routes, record provenance, and name the packet. They do not add a quotient of terms, a normalization selector, a proof-erased compiler route, or a host evaluator.

Thus the accepted reading is exactly the finite worked-example seal. All stronger metatheoretic endpoints are absent from the carrier and cannot be exported by a downstream consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesSeal}^{\uparrow}}$.

Constructive story: A MetaCICTypedExamplesSeal$^{\uparrow}$ packet is generated from finite typed-example rows, checker rows, closedness rows, selected computation rows, bounded decidability rows, bounded normal-equality rows, a finite seal ledger, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICTypedExamplesSeal}\uparrow}$, seed)

Scope closed: Definition 753.1, Definition 753.2, and Definition 753.3 bind the finite worked-example seal surface and its sibling checker routes.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICTypedExamplesSeal}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close subject reduction, strong normalization, global conversion decidability, quotient equality over represented terms, proof-erased compiler correctness, host evaluation, or bridge checking.

Upgrade path: Obligation closure requires checking carrier admission, gallery-entry visibility, checker visibility, closedness visibility, selected computation visibility, sibling bounded-checker routes, seal-ledger exactness, and non-escape for the displayed packet.

Chapter 754

A Concrete Naming Certificate for $\mathsf{BetaStepBoundary}^{\uparrow}$

The $\mathsf{BetaStepBoundary}^{\uparrow}$ packet realizes the beta audit map of chapter 9384 as a finite BEDC naming surface. It sits beside the subject-reduction carrier of chapter 483: the present packet names the BetaStep and BetaConv boundary rows, while the sibling packet records the preservation-facing route that must not be inferred from beta reduction alone.

Definition 754.1 (BetaStepBoundary carrier)**.** *A* $\mathsf{BetaStepBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, C, O, H, K, P, N)$$

with the following visible rows:

1. *R records the* BetaStep *rule surface, including the direct redex row and the congruence rows for application, lambda-domain, lambda-body, Pi-domain, and Pi-codomain positions;*
2. *C records the* BetaConv *common-reduct rows, including reflexivity, symmetry, the beta-star injections, beta-step injections, and the normal-middle conversion rows named by the audit map;*
3. *O records the subject-reduction obstruction triggers: beta substitution, application-argument stability, lambda-domain preservation, and Pi-domain preservation;*
4. *H stores componentwise* hsame *transports for the beta-rule rows, conversion rows, obstruction-trigger rows, provenance row, and local naming row;*
5. *K stores the displayed* Cont *routes by which consumers read beta steps, common-reduct conversions, and obstruction triggers without crossing from reduction into typed preservation;*
6. *P is the* Pkg *provenance over the MetaCIC beta audit map, chapter 483,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaStepBoundary}^{\uparrow}}$ *row.*

The packet contains no host conversion oracle, hidden typing-preservation theorem, closed subject-reduction theorem, quotient equality, or erased semantic equality coordinate.

Theorem 754.2 (BetaStepBoundary NameCert obligations)**.** *The carrier of Definition 754.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{BetaStepBoundary}^{\uparrow}}$ *obligation surface. Its five obligation rows are:*

1. *carrier admission for the finite packet $B = (R, C, O, H, K, P, N)$;*

2. *beta-rule exactness, saying that R exposes exactly the displayed* BetaStep *constructor surface;*
3. *conversion-boundary readback, saying that C exposes only the* BetaConv *common-reduct and normal-middle rows listed by the audit map;*
4. *subject-reduction non-escalation, saying that the obstruction triggers in O remain obligation rows and do not become a closed preservation theorem;*
5. *ledger non-escape, saying that H, K, P, N supply only transport, continuation, provenance, and naming for the displayed rows, never host conversion or hidden typing preservation.*

Lean4 ✓ `BEDC.Derived.BetaStepBoundaryUp.BetaStepBoundaryTasteGate single carrier alignment`

Proof. Carrier admission is the displayed BHist packet. Each coordinate is a finite row for beta rules, conversion readback, obstruction triggers, transport, continuation, provenance, or local naming.

Beta-rule exactness follows by projecting R. The row lists the direct beta redex and the congruence positions, so a consumer cannot read an additional reduction rule from the carrier.

Conversion-boundary readback follows by projecting C. Common-reduct conversion, beta-star injection, beta-step injection, and normal-middle conversion rows are available only as the listed conversion surface.

Subject-reduction non-escalation follows by projecting O and comparing it with chapter 483. The beta packet records the triggers that a preservation consumer must discharge, but it has no coordinate carrying a closed subject-reduction theorem.

Finally inspect H, K, P, N. These rows move and replay the same finite packet, record its provenance, and give it a local name. None of them contains host conversion or hidden typing preservation, so every consumer remains inside the displayed ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{BetaStepBoundary}\uparrow}$.

Constructive story: A $\mathsf{BetaStepBoundary}^{\uparrow}$ packet is generated from beta-step rule rows, beta-conversion common-reduct rows, subject-reduction obstruction triggers, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BetaStepBoundary}\uparrow}, \mathsf{seed})$

Scope closed: Definition 754.1 and Definition 754.2 bind the finite beta-step, beta-conversion, obstruction-trigger, transport, route, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BetaStepBoundary}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, beta-substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, host conversion, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, beta-rule exactness, conversion-boundary readback, subject-reduction non-escalation, transport, route replay, provenance, and ledger non-escape for the same finite packet.

Lemma 754.3 (BetaStepBoundary obstruction-trigger enumeration)**.** *Let $B = (R, C, O, H, K, P, N)$ be an accepted* $\mathsf{BetaStepBoundary}^{\uparrow}$ *carrier. If a consumer reads a subject-reduction obstruction trigger from B through the displayed obstruction slot, then the read trigger is one of exactly four rows:*

BetaSubstitutionPreservation, AppArgTypeStable, LamDomainSubjectReduction, PiDomainSubjectReduct

Conversely, each of these four rows is a displayed obstruction-trigger row of the same obstruction slot O.

Proof. Project the accepted carrier through Definition 754.1. The obstruction coordinate is the single row O, and that coordinate is displayed as the list of subject-reduction obstruction triggers: beta substitution, application-argument stability, lambda-domain preservation, and Pi-domain preservation.

The NameCert obligation surface of Definition 754.2 keeps this projection exact. Its subject-reduction non-escalation clause says that the entries in O remain obligation rows, while its ledger non-escape clause says that H, K, P, N supply only transport, continuation, provenance, and naming for the displayed rows. Thus a continuation read through K can select an entry of O, and an hsame transport through H can replay that selected entry, but neither operation creates another obstruction trigger.

The four labels are also the four discharge obligations named in section 9377.3, and section 9384.4 identifies them as the beta-redex, application-argument, lambda-domain, and Pi-domain triggers born at the beta boundary. Since R records beta-step constructors, C records beta-conversion readback, and P, N only record provenance and local naming, no subject-reduction obstruction trigger can be read from any coordinate other than O.

Therefore every admissible obstruction-trigger read from an accepted $\mathsf{BetaStepBoundary}^{\uparrow}$ packet is one of the four displayed rows, and the carrier definition supplies the reverse inclusion by listing all four rows inside O. □

Chapter 755

A Concrete Naming Certificate for MetaCICNormalizationBudget$^{\uparrow}$

The MetaCICNormalizationBudget$^{\uparrow}$ packet turns the normalization audit map of chapter 9383 into a finite NameCert surface. It is the normalization-side sibling of the conditional confluence witness in chapter 745, the subject-reduction obligation packet in chapter 712, and the closed-normal consistency boundary of chapter 733. The packet records a candidate-mediated budget route; it does not assert unconditional strong normalization for every closed typed MetaCIC term.

Definition 755.1 (MetaCIC normalization budget carrier)**.** *A* MetaCICNormalizationBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, C, E, S, A, P, R, H, Q, L, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row whose normalization read is requested;*
2. *C is the reducibility-candidate evidence row attached to the same closed term;*
3. *E is the closed-candidate or candidate-set membership row consumed by the checked normalization lemmas;*
4. *S is the candidate-implies-strong-normalization row, naming the Lean route* BEDC.MetaCIC.candidate_implies_s *as a consumed theorem surface rather than as a global closed-term theorem;*
5. *A is the application and Pi adequacy boundary row that prevents componentwise normalization data from becoming unconditional application normalization;*
6. *P is the proof-replay budget row recording the finite candidate projections, adequacy reads, and no-infinite-reduction consequences available to the consumer;*
7. *R is the refusal row for requests that omit candidate evidence, closed-candidate evidence, or candidate-set membership;*
8. *H records componentwise* hsame *transport for term, candidate, membership, theorem, adequacy, budget, refusal, route, provenance, and naming rows;*
9. *Q records the displayed* Cont *routes by which consumers read normalization only through C, E, S, A, P or receive the refusal row R;*

10. *L is the* Pkg *provenance over the normalization audit map, the confluence audit witness, the subject-reduction obligation packet, the closed-normal consistency boundary,* BHist, hsame, Cont, *and the local certificate;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationBudget}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host evaluator, an unconditional closed-term strong-normalization theorem, a subject-reduction discharge, a hidden application-normalization rule, or a conversion principle outside the candidate-mediated rows.

Lean4 (definition) `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier`

Theorem 755.2 (MetaCIC normalization budget NameCert obligations)**.** *For an accepted* MetaCICNormalizationB *carrier $B = (T, C, E, S, A, P, R, H, Q, L, N)$, the local* NameCert *surface exposes exactly the closed typed term row T, reducibility-candidate evidence C, closed-candidate or candidate-set membership E, candidate-to-strong-normalization row S, application and Pi adequacy boundary A, finite proof-replay budget P, refusal row R, componentwise transport H, displayed continuation routes Q, package provenance L, and local naming N. A consumer may read strong normalization only along the candidate-mediated route displayed by C, E, S, A, P.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetTasteGate single carrier alignment`

Proof. Project the carrier of Definition 755.1. The rows T, C, E, S, A, P display the positive route: a closed term is paired with candidate evidence, the evidence is read through the checked candidate-to-SN row, and the adequacy boundary records where Pi and application reasoning is allowed.

The refusal row R accounts for requests that skip the candidate evidence or ask for unconditional closed-term normalization. It is part of the same packet, so the negative answer is not an external convention.

The support rows H, Q, L, N transport, route, package, and name the same finite inventory. Since the carrier has no second theorem coordinate, the local NameCert obligations are exhausted by the displayed candidate-mediated budget surface. □

Theorem 755.3 (MetaCIC normalization budget obligation surface)**.** *Every accepted* MetaCICNormalizationBudget *carrier exposes exactly the positive rows*

$$T, C, E, S, A, P,$$

the refusal row R, and the support rows H, Q, L, N. The positive rows are read only as the candidate-mediated normalization budget, while R records requests for unconditional closed-term strong normalization or for normalization without the displayed candidate evidence.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier obligation surface`

Proof. Project the carrier of Definition 755.1. Its tuple has the six positive coordinates T, C, E, S, A, P, the single refusal coordinate R, and the four structural support coordinates H, Q, L, N; no other row is present.

Apply Definition 755.4. The positive surface is therefore candidate-mediated: the closed term row T reaches a strong-normalization read only through the candidate evidence C, E, the candidate-to-SN row S, the adequacy boundary A, and the replay budget P.

Apply Definition 755.5. Requests that ask for unconditional closed-term strong normalization, subject reduction, hidden application normalization, or a host evaluator have no positive coordinate in the carrier, so they are routed to R and to the support rows that transport, replay, source, and name the refusal. □

Theorem 755.4 (MetaCIC normalization budget candidate-route scope)**.** *Every positive normalization read from an accepted* MetaCICNormalizationBudget$^{\uparrow}$ *packet factors through*

$$T, C, E, S, A, P.$$

The row S may be consumed as the candidate-to-strong-normalization route for the displayed evidence, and the row A may be consumed as the displayed adequacy boundary, but neither row erases the candidate evidence or supplies closed subject reduction. The sibling packets of chapter 745, chapter 712, and chapter 733 therefore receive a budgeted normalization surface, not an unconditional theorem.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier candidate route scope`

Proof. Read the continuation coordinate Q. Its positive route reaches the strong-normalization read only after the term row T has been paired with the candidate rows C, E and the candidate-to-SN row S.

The adequacy row A records the Pi and application boundary named in chapter 9383. It lets consumers compare the route with confluence and subject-reduction siblings while preserving the guard that application normalization is not obtained from parts alone.

Finally inspect the provenance row L. It sends downstream readers to the confluence audit witness, the subject-reduction obligation packet, and the closed-normal consistency boundary as named consumers of the same finite budget. Those consumers inherit the candidate route; they do not receive a stronger closed-term normalization principle. □

Theorem 755.5 (MetaCIC normalization budget non-escape)**.** *An accepted* MetaCICNormalizationBudget$^{\uparrow}$ *packet cannot yield an unconditional theorem that every closed typed MetaCIC term is strongly normalizing. It cannot yield subject reduction, a hidden host evaluator, compiler semantics, quotient conversion, or application strong normalization without the displayed candidate evidence. Every such request is routed to the refusal row R and the public rows H, Q, L, N.*

Proof. Start with the carrier inventory. The only positive normalization route is the displayed candidate-mediated route through T, C, E, S, A, P.

If a consumer asks for normalization without C or E, the request has no positive route in Q. The packet therefore routes the request to the refusal row R, which records the missing evidence rather than filling it.

The remaining rows preserve the boundary. Componentwise hsame transport, continuation replay, package provenance, and local naming can carry the accepted packet to downstream chapters, but none of them adds a host evaluator, a subject-reduction theorem, a quotient conversion, compiler semantics, or an unconditional closed-term normalization coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier nonescape`

Theorem 755.6 (MetaCIC normalization budget candidate SN boundary)**.** *Every accepted* MetaCICNormalizationB *packet can export a strong-normalization read only when the displayed rows T, C, E, S, A, P are read*

together. Dropping the candidate evidence row C or the closed-candidate membership row E routes the request to the refusal row R, and dropping the adequacy boundary row A prevents application and Pi consumers from reading the route as closed subject reduction.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier candidate sn boundary`

Proof. Project the carrier of Definition 755.1. The positive read is not a single closed-term coordinate; it is the six-row route T, C, E, S, A, P together with the structural rows that transport and replay it.

Apply the candidate-route scope theorem, Definition 755.4. It forces every positive normalization read through the candidate evidence and adequacy boundary before downstream confluence, subject-reduction, or closed-normal consistency consumers may inspect the packet.

Finally apply non-escape, Definition 755.5. Missing C or E has no positive continuation and is recorded by R, while missing A leaves application and Pi reads at the boundary rather than turning the candidate route into closed subject reduction. □

Theorem 755.7 (MetaCIC normalization budget falsifiable boundary)**.** *Every accepted* MetaCICNormalizationBudg *packet that is queried for a normalization read while missing the candidate evidence row C or the closed-candidate membership row E must route the query to the refusal row R. In particular, an accepted packet whose public read exports strong normalization without both C and E is not a valid instance of the carrier of Definition 755.1.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier falsifiable boundary`

Proof. Inspect the carrier tuple. Its only positive normalization route is the displayed sequence T, C, E, S, A, P, while R is the unique row assigned to requests that omit candidate evidence or closed-candidate membership.

Apply Definition 755.6. The theorem states that a strong-normalization read is exported only when T, C, E, S, A, P are read together. Therefore a request missing C or E has no positive continuation.

It remains to identify the public destination of such a request. By Definition 755.5, missing candidate data is recorded by R rather than supplied by a hidden evaluator, subject-reduction theorem, or unconditional closed-term normalization coordinate. □

Theorem 755.8 (MetaCIC normalization budget refusal exactness)**.** *Every accepted* MetaCICNormalizationBudget↑ *packet queried for a normalization read while missing C or E has exactly one public destination: the refusal row R. The request is transported by the componentwise row H and replayed by the displayed continuation row Q, and it has no alternative route through T, S, A, P or through any structural row outside H, Q, L, N.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier refusal exactness`

Proof. Project the carrier of Definition 755.1. Its visible rows are $T, C, E, S, A, P, R, H, Q, L, N$, so the packet contains one refusal coordinate and no hidden evaluator, subject-reduction coordinate, host normalization coordinate, or second structural destination.

Apply Definition 755.4. The positive route reaches a strong-normalization read only through the full candidate-mediated chain T, C, E, S, A, P. If C or E is missing, that route is not available as a public read.

Apply Definition 755.5 and Definition 755.7. They identify the missing-candidate query with the refusal row R rather than with a positive normalization row.

Finally read the support rows. The row H transports the same refusal judgment componentwise, while Q replays the displayed refusal route to consumers and L, N record provenance and local naming only. Hence the public destination is exactly R, with transport and replay but without an alternative positive or structural route. □

Theorem 755.9 (MetaCIC normalization budget refusal transport)**.** *Let $B = (T, C, E, S, A, P, R, H, Q, L, N)$ be an accepted* MetaCICNormalizationBudget$^{\uparrow}$ *packet. If a normalization query is missing the candidate evidence row C or the closed-candidate membership row E and is therefore routed to R, then componentwise* hsame *transport along H and displayed* Cont *replay along Q preserve the same refusal destination. No transported or replayed read creates a positive route through T, S, A, P, and the provenance and naming rows L, N record only the same refusal boundary.*

Proof. Project the accepted carrier through Definition 755.1. The only displayed rows are $T, C, E, S, A, P, R, H, Q, L$ with R the unique refusal coordinate.

Apply Definition 755.8. A query missing C or E has exactly one public destination, namely R, and it has no alternative route through T, S, A, P or through structural rows outside H, Q, L, N.

Apply Definition 755.5. Componentwise transport cannot supply the absent candidate evidence, and replay cannot manufacture an unconditional strong-normalization route. The rows H and Q therefore carry the same refusal judgment rather than changing its destination.

Finally read L and N as provenance and local naming rows. They source and name the transported refusal packet, but they do not add candidate evidence, membership evidence, a second refusal destination, or a positive normalization coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier refusal transport`

Theorem 755.10 (MetaCIC normalization budget refusal idempotence)**.** *Let $B = (T, C, E, S, A, P, R, H, Q, L, N)$ be an accepted* MetaCICNormalizationBudget$^{\uparrow}$ *packet. If a normalization query is missing the candidate evidence row C or the closed-candidate membership row E and is routed to R, then any finite ledger of displayed* Cont *replays through Q and componentwise transports through H still has public destination R. Iterating the replay and transport rows never creates a positive route through T, S, A, P, and the provenance and naming rows L, N continue to record only the same refusal boundary.*

Proof. Start with the initial missing-candidate query. By Definition 755.8, its only public destination is R; there is no alternative route through T, S, A, P or through structural rows outside H, Q, L, N.

For the base ledger with no replay or transport step, the destination is therefore R.

For the induction step, assume the finite prefix of the replay ledger still has public destination R. If the next displayed move is a componentwise hsame transport through H or a Cont replay through Q, apply Definition 755.9. The next read preserves the same refusal destination and does not add candidate evidence, membership evidence, a second refusal destination, or a positive normalization coordinate.

Induction over the finite replay ledger gives the claim for every finite sequence. The rows L, N source and name the same refusal packet throughout the induction, so repeated replay records the original boundary rather than changing it. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier refusal idempotence`

Theorem 755.11 (MetaCIC normalization budget independence boundary)**.** *The* MetaCICNormalizationBudget$^{\uparrow}$ *packet is not bijective to the confluence audit witness of chapter 745, the subject-reduction obligation packet of chapter 712, the closed-normal consistency boundary of chapter 733, or the normalization frontier of chapter 892. The reason is row-level: this packet alone combines the closed term row T, candidate evidence C, closed-candidate membership E, candidate-to-strong-normalization row S, adequacy boundary A, replay budget P, refusal row R, transport row H, continuation row Q, provenance row L, and local naming row N as one finite carrier.*

Proof. Compare the row inventory with the named siblings. The confluence audit witness records a confluence-facing route; the subject-reduction obligation packet records preservation-facing data; the closed-normal consistency boundary records consistency-facing data; and the normalization frontier records an open boundary for normalization claims.

The carrier of Definition 755.1 has a different public signature. It jointly stores $T, C, E, S, A, P, R, H, Q, L, N$, so it contains both the positive candidate-mediated route and the refusal row in the same NameCert packet.

A bijection to any listed sibling would have to preserve those rows and their route order. Since none of the siblings exposes exactly this combined candidate, adequacy, replay, refusal, transport, continuation, provenance, and naming surface, no listed sibling is the same packet under renaming. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationBudgetUp.MetaCICNormalizationBudgetCarrier independence boundary`

Falsifiable prediction: Every accepted MetaCICNormalizationBudget$^{\uparrow}$ query missing candidate evidence C or closed-candidate membership E routes to the refusal row R and does not export a positive strong-normalization read.

Independence witness: metacicconfluenceauditwitness, metacic-subject-reduction-obligation, closed-normal-consistency-boundary, metacicnormalizationfrontier: MetaCICNormalizationBudget$^{\uparrow}$ is not bijective to any listed sibling because it alone combines $T, C, E, S, A, P, R, H, Q, L, N$ as one candidate-mediated normalization-budget carrier.

Lean Nontrivial and FieldFaithful TasteGate instances for the same MetaCICNormalizationBudget$^{\uparrow}$ carrier must inspect the rows $T, C, E, S, A, P, R, H, Q, L, N$ together, so the positive route and the refusal boundary are tested on the same accepted packet.

Closure status for NameCert$_{\text{MetaCICNormalizationBudget}\uparrow}$**.**

Constructive story: A MetaCICNormalizationBudget$^{\uparrow}$ packet is generated from a closed typed MetaCIC term row, reducibility-candidate evidence, closed-candidate or candidate-set membership, the candidate-to-strong-normalization row, adequacy-boundary data, proof-replay budget data, refusal data, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationBudget}\uparrow}$, scopedClosed)

Scope closed: Definition 755.1, Definition 755.2, Definition 755.3, Definition 755.4, Definition 755.5, Definition 755.6, Definition 755.8, Definition 755.9, Definition 755.10, and the Lean TasteGate single-carrier marker bind the finite candidate-mediated normalization budget surface and its refusal boundary.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalizationBudget}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove unconditional closed-term strong normalization, closed subject reduction, application strong normalization without candidate evidence, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public consumer surface for the checked carrier admission, candidate-route exactness, adequacy-boundary replay, refusal exactness, provenance accounting, transport stability, and non-escape rows of the same finite packet.

Chapter 756

A Concrete Naming Certificate for $\mathsf{TypedExampleGallery}^{\uparrow}$

The $\mathsf{TypedExampleGallery}^{\uparrow}$ packet realizes the audit reading of chapter 9382 as finite BEDC data. Its source material is the existing MetaCIC typed-example corpus: Church booleans and naturals, pairs and options, lists and sums, identity and polymorphic identity rows, application forms, beta redexes, projections, Pi forms, constants, and sort-indexed context examples. The packet sits after the closed substitution boundary of chapter 527, the derivation route of chapter 598, and the proof-erasure route of chapter 642. It names the gallery as a finite regression surface without turning the examples into subject reduction, normalisation, confluence, or global conversion theorems.

Definition 756.1 (Typed example gallery carrier)**.** *A* $\mathsf{TypedExampleGallery}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, T, C, B, R, H, K, P, N)$$

with the following displayed rows:

1. *E is the finite gallery-entry row for the accepted Church, identity, application, beta-redex, projection, Pi, constant, and sort-context examples;*
2. *T is the type-checking row recording each displayed* $\mathsf{HasType}$ *or checker-accepted example;*
3. *C is the closedness row recording the* $\mathsf{ClosedAt}$ *inputs consumed by the typed examples and computation rows;*
4. *B is the beta-computation row for the selected redex, eliminator, boolean, natural-number, list, sum, and equality examples that compute inside the gallery;*
5. *R is the regression-surface row recording the finite families whose preservation is required when checker, closedness, substitution, beta, projection, or binder code is changed;*
6. *H records componentwise* hsame *transport across E, T, C, B, R, K, P, N;*
7. *K records the displayed* Cont *routes by which consumers replay gallery entry reads, checker reads, closedness reads, beta reads, and regression-surface reads;*
8. *P is the* Pkg *provenance over chapter 9382, chapter 527, chapter 598, and chapter 642;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedExampleGallery}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary MetaCIC term, no global type-safety theorem, no closed-term normalisation theorem, no confluence theorem, no proof-erased compiler theorem, and no host evaluator.

Theorem 756.2 (Typed example gallery NameCert obligations)**.** *For the carrier of Definition 756.1, the local* $\mathsf{NameCert}_{\mathsf{TypedExampleGallery}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for* $G = (E, T, C, B, R, H, K, P, N)$*;*
2. *checker visibility through the displayed type-checking row* T *for every accepted gallery entry in* E*;*
3. *closedness visibility through* C *for each example whose typing or computation row consumes a* $\mathsf{ClosedAt}$ *input;*
4. *beta visibility through* B *for the selected computation rows and no other conversion surface;*
5. *regression-surface visibility through* R*, transported only by* H, K, P, N *and grounded in the sibling provenance listed in* P*.*

Proof. Project the finite packet G. Carrier admission supplies the displayed entry, checker, closedness, beta, regression, transport, replay, provenance, and local naming rows.

The checker obligation reads T only against entries already present in E. The closedness obligation reads C only for entries whose displayed theorem rows require $\mathsf{ClosedAt}$ inputs. The beta obligation reads B only for the selected computations in the gallery.

Finally read R, H, K, P, N. The regression row names the finite families whose checked behavior is preserved as a gallery surface, while transport, replay, provenance, and local naming keep every consumer inside the same displayed packet. No row adds a general subject-reduction, normalisation, confluence, global conversion, proof-erasure, or host-evaluation endpoint. □

Theorem 756.3 (Typed example gallery regression surface)**.** *Every consumer of an accepted* $\mathsf{TypedExampleGallery}^{\uparrow}$ *packet receives a finite regression surface for the typed-example corpus: accepted typing rows, closedness inputs, selected beta computations, and sibling provenance through chapter 527, chapter 598, and chapter 642. The packet cannot be reread as a theorem about all represented terms, all closed substitutions, all beta steps, all erased proof terms, or all checker results.*

Proof. The regression surface is the row R together with the entry, checker, closedness, and beta rows E, T, C, B. These rows are finite and are indexed by the gallery families recorded in chapter 9382.

The sibling references in P locate the required substrate: closed substitution supplies the closed-term boundary, derivation traces supply finite checker routes, and proof-erasure traces supply the erased-readback discipline. The packet records dependence on those surfaces without merging their theorem rows into the gallery.

Since the carrier contains no row for an arbitrary represented term, arbitrary closed substitution, arbitrary beta step, arbitrary proof erasure, or arbitrary checker result, the consumer cannot export a global metatheorem. The accepted reading is exactly the finite gallery regression surface. □

Theorem 756.4 (Typed example gallery closed beta regression route)**.** *Every closed beta example in an accepted* $\mathsf{TypedExampleGallery}^{\uparrow}$ *packet routes through the displayed entry row* E*, type row* T*, closedness row* C*, beta row* B*, regression row* R*, and sibling provenance to chapter 527, chapter 598, and chapter 642. The route records a finite gallery obligation only: it does not assert subject reduction, strong normalisation, confluence, global conversion decidability, proof-erased compiler correctness, arbitrary checker completeness, or a host evaluator.*

Proof. Project the carrier $G = (E, T, C, B, R, H, K, P, N)$ from Definition 756.1. A closed beta example is first an entry of E, and its displayed type evidence is read from T rather than from a host checker theorem.

Read the closedness and computation rows next. The closedness row C supplies exactly the ClosedAt inputs consumed by the example, and the beta row B supplies the selected computation surface for the displayed redex or eliminator.

Finally apply the NameCert obligations and the regression surface theorem. The row R records the regression family, while P keeps the closed-substitution boundary, type-checking derivation trace, and proof-erasure trace as sibling provenance. Transport, replay, provenance, and local naming through H, K, P, N preserve the same finite route and add no global metatheorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypedExampleGallery}^{\uparrow}}$.

Constructive story: A $\mathsf{TypedExampleGallery}^{\uparrow}$ packet is generated from finite typed-example entries, checker rows, closedness rows, selected beta-computation rows, a regression-surface row, componentwise transport, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedExampleGallery}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 756.1, Definition 756.2, Definition 756.3, and Definition 756.4 bind the finite gallery-entry, checker, closedness, beta, regression, sibling-provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypedExampleGallery}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close subject reduction, strong normalisation, confluence, global conversion decidability, proof-erased compiler correctness, arbitrary checker completeness, or bridge checking.

Upgrade path: Obligation closure requires checking carrier admission, checker visibility, closedness visibility, beta visibility, regression-surface non-escape, and sibling provenance for the displayed packet.

Chapter 757

A Concrete Naming Certificate for $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$

The $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ packet records the finite tail-radius accounting used by Real and Completion consumers before a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal read is allowed. It sits over $\mathsf{DyadicRatCore}^{\uparrow}$ (chapter 291), $\mathsf{StreamName}^{\uparrow}$ (chapter 126), $\mathsf{RegSeqRat}^{\uparrow}$ (chapter 301), $\mathsf{RealObservationBudget}^{\uparrow}$ (chapter 611), and $\mathsf{Real}^{\uparrow}$ (chapter 38). The certificate is a finite ledger for unary precision and tail windows; it is not a quotient of streams, a host real equality witness, a selected limit, an unbounded observation schedule, or an ambient completeness principle.

Definition 757.1 (Dyadic tail-radius ledger carrier)**.** *A* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, T, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the unary precision request whose tolerance selects the tail-radius read;*
2. *T is the finite tail-window row, including the requested tail threshold and the radius budget attached to that threshold;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened by E and T;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row carrying the dyadic observations and displayed radii for the selected windows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the dyadic readback after the tail-radius ledger has been checked;*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row available to completion-facing consumers only after R has been displayed;*
7. *H records componentwise* hsame *transports for E, T, W, D, R, S and the structural rows;*
8. *C records the displayed* Cont *routes from precision request to tail window, StreamName windows, dyadic readback, regular-rational handoff, and real seal;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{NameCert}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $E, T, W, D, R, S, H, C, P, N$.

Theorem 757.2 (Dyadic tail-radius ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}^{\uparrow}}$ *surface for Definition 757.1 consists exactly of carrier admission for* $L = (E, T, W, D, R, S, H, C, P, N)$*, unary precision exposure, finite tail-window and radius-ledger exposure,* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and radius exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *seal-boundary exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerTasteGate single carrier alignment`

Proof. Project the carrier of Definition 757.1. The rows E, T, W, D, R, S are the only mathematical coordinates, and H, C, P, N are the only transport, route, package, and naming coordinates.

Read the source chapters in the order consumed by the packet. The DyadicRatCore row supplies dyadic readback, the StreamName row supplies finite windows, the RegSeqRat row supplies the regular-rational handoff, the Real observation-budget chapter supplies the finite budget discipline, and the Real chapter supplies the final seal boundary.

The local NameCert obligation surface is therefore the finite route through those displayed rows. Since the carrier contains no coordinate for quotient streams, host real equality, selected limits, unbounded schedules, or ambient completeness, no obligation can export such data. □

Theorem 757.3 (Dyadic tail-radius ledger tail-budget coverage)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

every tail-radius read admitted by the ledger is covered by the finite tail-window row T*, one of the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*, and the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback and radius rows in* D*. The admitted route is the finite budget route*

$$E \longmapsto T \longmapsto W \longmapsto D$$

with H, C, P, N *attached as transport, route replay, provenance, and local naming data.*

Proof. Start with the precision request E. The continuation rows in C send that request to the displayed tail-window row T before any StreamName window can be consumed.

Window coverage then follows the StreamName and observation-budget pattern: each selected tail window in T opens only finite rows in W, and every dyadic readback or displayed radius in D is tied to one of those windows. This is the same finite-window discipline used by Definition 611.3 and Definition 611.20.

Transport along H preserves the same displayed coordinates, while P and N keep provenance and naming attached to the route. Thus a tail-radius read has no source outside E, T, W, D, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerCarrier tail budget coverage`

Theorem 757.4 (Dyadic tail-radius ledger budget-selector pullback)**.** *Let* $L = (E, T, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier, and let*

$$F = (B, S_F, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *carrier. Suppose the displayed transport rows identify*

$$E\,\mathsf{hsame}\,B, \qquad T\,\mathsf{hsame}\,S_F, \qquad W\,\mathsf{hsame}\,W_F, \qquad D\,\mathsf{hsame}\,D_F, \qquad R\,\mathsf{hsame}\,R_F, \qquad S\,\mathsf{hsame}\,E_F.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of the two packets factors through the single finite route*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S,$$

equivalently through the selector route

$$B \longmapsto S_F \longmapsto W_F \longmapsto D_F \longmapsto R_F \longmapsto E_F.$$

The pullback exports no quotient streams, host real equality, countable choice, hidden limit selector, independent radius ledger, or ambient completeness principle.

Proof. Start on the dyadic side. Tail-budget coverage from Definition 757.3 places every admitted tail-radius read on $E \longmapsto T \longmapsto W \longmapsto D$, and the RegSeqRat and Real handoff of Definition 757.6 extends the same route through R and S.

Move to the selector side. The NameCert obligations of Definition 782.2 expose only the displayed rows of F, route determinacy from Definition 782.7 fixes $B \longmapsto S_F \longmapsto W_F \longmapsto D_F \longmapsto R_F \longmapsto E_F$, and terminal selector synchronization from Definition 782.23 keeps Real-facing and completion-facing reads on that terminal route.

Finally apply the six displayed hsame transports. They identify the dyadic precision, tail-window, StreamName-window, dyadic-readback, RegSeqRat-handoff, and Real-seal rows with the selector budget, schedule, window, readback, handoff, and terminal seal rows. Since both carriers expose only those finite rows and their transport, continuation, provenance, and naming data, a common consumer has the single pullback route displayed above and no coordinate from which quotient streams, host real equality, countable choice, hidden limit selection, an independent radius ledger, or ambient completeness could be recovered. □

Theorem 757.5 (Dyadic tail-radius ledger tail-radius witness extraction)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

any tail-radius witness requested by a completion-facing consumer is extracted from the displayed finite route

$$E \longmapsto T \longmapsto W \longmapsto D$$

before the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *are readable. The extraction exports no quotient stream equality, host real equality, selected limit, unbounded tail search, independent radius ledger, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerCarrier tail radius witness extraction`

Proof. Begin with the local NameCert obligations of Definition 757.2. They identify $E, T, W, D, R, S, H, C, P, N$ as the whole accepted packet and make tail-radius witness requests part of the displayed finite ledger.

Apply tail-budget coverage from Definition 757.3. The requested witness is therefore read from the route $E \longmapsto T \longmapsto W \longmapsto D$ before any source-facing or completion-facing handoff is available.

Next use RegSeqRat and Real handoff, Definition 757.6. It places R and S strictly downstream of the finite tail-budget route, so a consumer cannot read the handoff or seal before the witness has been extracted from D.

Finally compose route replay with non-escape, Definition 757.9 and Definition 757.7. Route replay attaches H, C, P, N to the same finite path, and non-escape excludes quotient stream equality, host real equality, selected limits, unbounded tail searches, independent radius ledgers, and ambient completeness from the exported witness. □

Theorem 757.6 (Dyadic tail-radius ledger RegSeqRat and Real handoff)**.** *For an accepted* DyadicTailRadiusLedger$^{\uparrow}$ *carrier, every* RegSeqRat$^{\uparrow}$ *handoff row R and every* Real$^{\uparrow}$ *seal row S are downstream of the finite tail-budget route of Definition 757.3. In particular, every admitted completion-facing read factors through*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerCarrier regseq real handoff`

Proof. First apply Definition 757.3. It places every admitted tail-radius read inside the finite precision, tail-window, StreamName, and dyadic readback rows.

Next use the RegSeqRat handoff surface from chapter 301. The handoff row R is available only after the dyadic readback row D has been displayed through the same packet route.

Finally use the Real seal boundary from chapter 38 and the observation-budget seal discipline of Definition 611.5. The seal row S is therefore reached only after R, with the same H, C, P, N structural rows attached. □

Theorem 757.7 (Dyadic tail-radius ledger non-escape)**.** *An accepted* DyadicTailRadiusLedger$^{\uparrow}$ *carrier exports only the displayed rows $E, T, W, D, R, S, H, C, P, N$ of Definition 757.1. It exports no quotient stream equality, host real equality, selected limit, unbounded tail search, second radius ledger, independent real carrier, or ambient completeness principle.*

Proof. The NameCert obligation theorem identifies the whole local certificate surface as the carrier rows $E, T, W, D, R, S, H, C, P, N$. There is no additional mathematical coordinate and no separate analytic witness row.

Apply the tail-budget coverage theorem to any attempted tail or radius read. The read must pass through E, T, W, D with the displayed transport, route, provenance, and naming rows.

Apply the RegSeqRat and Real handoff theorem to any source-facing or completion-facing read. It forces the read through the same finite route before R or S can be consumed.

Because the excluded objects are not coordinates of the carrier and are not produced by the displayed continuations, no downstream consumer can recover them from the packet. □

Theorem 757.8 (Dyadic tail-radius ledger transport stability)**.** *For an accepted* DyadicTailRadiusLedger$^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

componentwise hsame *transport through H preserves exactly the displayed rows $E, T, W, D, R, S, H, C, P, N$. In particular, a transported public read remains a read of the same precision request, tail-window ledger, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *readback,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal row, displayed route table, package provenance, and local naming row.*

Proof. The carrier definition lists $E, T, W, D, R, S, H, C, P, N$ as the complete packet surface. The NameCert obligation theorem then identifies componentwise hsame transport as one obligation on that surface rather than an additional coordinate.

Apply tail-budget coverage to the rows E, T, W, D. The transported precision, window, StreamName, and dyadic readback data remain attached to the same finite budget route.

Apply the RegSeqRat and Real handoff theorem to R and S. These rows are downstream of the transported finite route, so transport cannot create an independent handoff or seal.

Finally apply non-escape. Since the carrier has no quotient stream, selected limit, host real equality, unbounded search, or ambient completeness coordinate, transport preserves the displayed ledger surface and nothing beyond it. □

Theorem 757.9 (Dyadic tail-radius ledger route replay)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier, every admitted completion-facing route replays the finite chain*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

through the displayed Cont *rows in* C*. No admitted completion-facing read may bypass the tail-window, StreamName, dyadic readback, or* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows before reaching the* $\mathsf{Real}^{\uparrow}$ *seal row.*

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerCarrier route replay`

Proof. Start from the carrier definition and the NameCert obligation theorem. They place displayed Cont route replay inside C and make C the only route table exposed by the packet.

Tail-budget coverage supplies the first part of the route: the precision request E reaches the tail-window ledger T, finite StreamName windows W, and dyadic readback row D through displayed continuations.

The RegSeqRat and Real handoff theorem supplies the final part of the route. Every completion-facing read reaches R after D and reaches S only after R.

Transport stability keeps the route rows aligned under componentwise hsame, and non-escape excludes any hidden continuation table or ambient completion principle. Thus the displayed chain is the whole admitted completion-facing replay. □

Theorem 757.10 (Dyadic tail-radius ledger provenance exhaustion)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier, the* Pkg *provenance row* P *and local* $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}^{\uparrow}}$ *row* N *account for all public reads of the packet. They account only for the displayed rows* $E, T, W, D, R, S, H, C, P, N$ *and do not add quotient streams, selected limits, host real equality, unbounded tail search, or ambient completeness.*

Proof. The carrier definition gives P and N as the package-provenance and local-naming rows for the same finite tuple. The NameCert obligation theorem makes provenance and local naming obligations for that tuple, not sources of extra analytic data.

Tail-budget coverage accounts for every public tail-radius read through E, T, W, D with H, C, P, N attached. The RegSeqRat and Real handoff theorem accounts for completion-facing reads through the same route extended by R, S.

Transport stability shows that moving the public surface along H keeps the same provenance and naming rows. Route replay shows that completion-facing access consumes the displayed continuation chain in C.

The non-escape theorem then eliminates the remaining alternatives. Since the excluded objects are not carrier coordinates and are not produced by P or N, provenance and local naming exhaust public reads without exporting those objects. □

Theorem 757.11 (Dyadic tail-radius ledger shared-tail service boundary)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

any shared-tail or completion-facing consumer that asks for a common dyadic threshold must read the finite route

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

This is the service boundary consumed by $\mathsf{CauchySharedTailModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The boundary exports no hidden threshold, quotient stream, selected limit, host real equality, or ambient completeness datum.*

Proof. The tail-budget coverage theorem first places any threshold request inside the finite rows E, T, W, D. Thus the requested tolerance, tail window, StreamName windows, and dyadic radius readback are all displayed before the consumer can ask for a common dyadic threshold.

The RegSeqRat and Real handoff theorem then extends the same displayed route through R and S. Hence completion-facing reads do not bypass the ledger; they consume the same route used by the sibling $\mathsf{CauchySharedTailModulus}^{\uparrow}$ certificate of chapter 983.

Finally apply the non-escape theorem. Since hidden thresholds, quotient streams, selected limits, host real equalities, and ambient completeness data are not coordinates of L and are not produced by its displayed continuations, they cannot be exported through this boundary. □

Theorem 757.12 (Dyadic tail-radius ledger CauchyCriterion handoff)**.** *Let $L = (E, T, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier. Any* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier whose tail row uses the same precision request,* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic readback, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff must read*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

before its M, Q, T, W, E criterion route can expose a $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Apply Definition 757.3, Definition 757.6, and Definition 757.9. These force the precision request, tail window, StreamName windows, dyadic readback, regular-rational handoff, and Real seal onto the displayed finite route.

Then use Definition 392.2. A CauchyCriterion carrier sharing those rows can expose its Real seal only after the same finite tail-budget route has been consumed, so the dyadic ledger is the upstream handoff for the criterion route. □

Theorem 757.13 (Dyadic tail-radius ledger CauchyCriterion TailBudgetCoherence pullback)**.** *Let $L = (E, T, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier, and let K be an accepted* $\mathsf{CauchyCriterion}^{\uparrow}$ *carrier sharing the same precision request,* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal rows. Any* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *consumer of the Cauchy route for K must factor through the displayed finite pullback route*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S,$$

then through the CauchyCriterion rows M, Q, T, W, E, and then through the TailBudgetCoherence rows at the Real-completion boundary. This route exports no quotient streams, selected limits, host real equality, detached tail oracle, metric-completeness theorem, or ambient Real completeness principle.

Proof. First apply tail-budget coverage from Definition 757.3. It places the precision request, tail window, StreamName windows, and dyadic readback on the finite segment $E \longmapsto T \longmapsto W \longmapsto D$.

Next apply the CauchyCriterion handoff of Definition 757.12. Since K shares the same precision, StreamName, dyadic readback, RegSeqRat, and Real seal rows, the route must continue through R and S before the CauchyCriterion carrier exposes its seal-facing rows.

Now use the budgeted shared-tail seal compatibility of Definition 392.8. It identifies the displayed CauchyCriterion rows M, Q, T, W, E as the shared-tail seal route over the same dyadic ledger coverage row.

Finally apply Definition 392.9. The TailBudgetCoherence consumer attaches only after the Cauchy route has displayed its threshold, budget, window, tolerance, readback, transport, continuation, package, and local naming rows. Non-escape and provenance exhaustion then rule out quotient streams, selected limits, host real equality, detached tail oracles, metric completeness, and ambient Real completeness as possible exits from the pullback route. □

Theorem 757.14 (Dyadic tail-radius ledger scoped route package)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

the precision request E*, tail window* T*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* S *form one scoped route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Every admitted L10 tail-radius consumer reads the same finite route*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

with H, C, P, N *as the transport, route replay, provenance, and local naming rows.*

Proof. Start with the NameCert obligations of Definition 757.2. They identify the accepted carrier with the finite rows $E, T, W, D, R, S, H, C, P, N$ over BHist, hsame, Cont, Pkg, and NameCert.

Apply tail-budget coverage from Definition 757.3. The precision request, tail window, StreamName windows, and DyadicRatCore readback therefore form the first route segment $E \longmapsto T \longmapsto W \longmapsto D$.

Use the RegSeqRat and Real handoff theorem of Definition 757.6. This extends the same finite route through R and then S, so no L10 tail-radius consumer can reach a seal row without the handoff row.

Finally combine route replay, provenance exhaustion, and the shared-tail service boundary from Definition 757.9, Definition 757.10, and Definition 757.11. These rows attach H, C, P, N to the same route and exclude hidden thresholds, quotient streams, selected limits, host real equality, unbounded searches, and ambient completeness. □

Theorem 757.15 (Dyadic tail-radius ledger monotone radius handoff)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

let E' *be a unary precision request refining* E*, and let* T' *be the displayed tighter tail-radius ledger transported from* T *by the componentwise* hsame *row* H*. Every admitted completion-facing read for* E' *factors through the same finite seal route*

$$E' \longmapsto T' \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The refinement exports no quotient stream, selected limit, host real equality, or completeness principle.

Proof. Tail-budget coverage first places the refined precision request inside a displayed tail-window row. Since T' is transported by H from the accepted ledger row T, the tighter radius ledger remains on the same componentwise hsame surface and does not create a second tail object.

The RegSeqRat and Real handoff theorem then consumes the displayed tail row through W, D, R, S. Replacing E, T by the finer displayed pair E', T' only tightens the first segment of the route; it does not change the downstream StreamName, DyadicRatCore, RegSeqRat, or Real seal rows.

Route replay attaches the same continuation row C to the transported route. Provenance exhaustion attaches the same P, N rows and rules out any completion-facing read whose source is not one of the displayed carrier coordinates.

Finally apply the scoped route package and the shared-tail service boundary. They identify the whole admitted seal route and exclude quotient streams, selected limits, host real equalities, and ambient completeness principles from the exported surface. □

Theorem 757.16 (Dyadic tail-radius ledger refinement-meet totality)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier*

$$L = (E, T, W, D, R, S, H, C, P, N),$$

let E_1 *and* E_2 *be unary precision requests refining the displayed request* E*. Suppose the carrier displays a common unary refinement row* E_m *together with a transported tail-window row* T_m *for that meet. Then every completion-facing read admitted for either* E_1 *or* E_2 *factors through the single finite route*

$$E_m \longmapsto T_m \longmapsto W \longmapsto D \longmapsto R \longmapsto S.$$

The common refinement exports no quotient stream, selected limit, host real equality, unbounded search, or completeness principle.

Proof. Start with tail-budget coverage. It places each refining request inside the displayed finite tail-budget surface and identifies the common row E_m as the carrier-visible refinement through which both requests must be read.

Apply monotone radius handoff to the two requests. Transport through H sends the common refinement to the single tail-window row T_m, so tightening either request changes only the first segment of the route and does not introduce a second StreamName, DyadicRatCore, RegSeqRat, or Real seal row.

Route replay then attaches the continuation row C to the common path, while provenance exhaustion attaches P and N to the same carrier coordinates. Thus a completion-facing consumer of either refinement has no admitted source outside $E_m, T_m, W, D, R, S, H, C, P, N$.

Finally apply the shared-tail service boundary. It is exactly the public boundary that permits both refinement consumers to share the tail-radius ledger while refusing quotient streams, selected limits, host real equalities, unbounded searches, and ambient completeness principles. □

Theorem 757.17 (Dyadic tail-radius ledger common-refinement pullback)**.** *Let*

$$L_0 = (E_0, T_0, W_0, D_0, R_0, S_0, H_0, C_0, P_0, N_0)$$

and

$$L_1 = (E_1, T_1, W_1, D_1, R_1, S_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carriers. Suppose* E_0 *and* E_1 *refine to a common unary precision request* E^**, and suppose the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows* W_0, W_1 *and the* $\mathsf{Real}^{\uparrow}$ *seal rows* S_0, S_1 *are componentwise* hsame *on a common displayed finite window. Then the two transported tail-radius ledger routes admit a common displayed pullback over the same route*

$$E^* \longmapsto T^* \longmapsto W^* \longmapsto D^* \longmapsto R^* \longmapsto S^*.$$

The pullback exports only the shared BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It exports no quotient stream, selected limit, host real equality, or ambient completeness datum.*

Proof. Apply the monotone radius handoff theorem to L_0 and L_1 along the common request E^*. This gives transported tail rows over the same refined precision request and keeps each completion-facing read on its displayed finite seal route.

Use route replay and provenance exhaustion for the two accepted carriers. The replay rows identify the visible continuation path, while provenance exhaustion says that any admitted comparison must be assembled from the displayed carrier coordinates and the local Pkg and NameCert rows.

The common $\mathsf{StreamName}^{\uparrow}$ window hypothesis identifies the shared finite window segment. The real seal scope theorem identifies the shared terminal seal segment, and the window route scope theorem keeps the dyadic readback and regular-rational handoff between those endpoints on the same finite route.

It remains to record what the pullback can expose. Non-escape and the shared-tail service boundary remove quotient streams, selected limits, host real equalities, hidden tail searches, and ambient completeness data, leaving only the common displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. □

Theorem 757.18 (Dyadic tail-radius ledger budget monotonicity)**.** *Let $L = (E, T, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier. If a displayed tail-radius request is refined through the existing meet row and common-refinement row, then every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff reads the same dyadic tolerance ledger D with no larger hidden observation budget and no separate tail route.*

Proof. Start with refinement-meet totality, Definition 757.16. It places the refined request behind a common finite precision row and transported tail-window row before any downstream comparison is visible.

Apply the common-refinement pullback theorem, Definition 757.17. The two refined presentations meet over the same displayed route through the shared tail row, StreamName window, dyadic ledger, RegSeqRat handoff, and Real seal row.

Now use dependency scope and Real seal scope, Definition 757.19 and Definition 757.21. These results force every admitted public read to stay inside $E, T, W, D, R, S, H, C, P, N$, and they make the seal row downstream of the same dyadic ledger D.

The refinement therefore tightens only the displayed request segment. Because no hidden observation budget, unbounded search, selected limit, quotient stream, host equality, or ambient completeness coordinate is exported by the carrier, the downstream handoff cannot create a larger budget or a second tail route. □

Theorem 757.19 (Dyadic tail-radius ledger dependency scope)**.** *Every accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier factors through the displayed rows $E, T, W, D, R, S, H, C, P, N$ over exactly the local dependency scope* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No admitted public read can use a dependency outside this displayed scope.*

Proof. The carrier and NameCert obligation surface of Definition 757.1 and Definition 757.2 list the complete packet surface. The mathematical rows are E, T, W, D, R, S, and the structural rows are H, C, P, N.

Tail-budget coverage and the RegSeqRat/Real handoff theorem route every admitted read through E, T, W, D, R, S. Route replay, provenance exhaustion, and the shared-tail service boundary then attach BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ to the same finite route.

The non-escape theorem excludes hidden thresholds, quotient streams, selected limits, host real equality, unbounded schedules, and ambient completeness. Hence the accepted carrier has precisely the displayed dependency scope. □

Theorem 757.20 (Dyadic tail-radius ledger window route scope)**.** *For an accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *carrier, every tail-window or radius-window read factors through*

$$E \longmapsto T \longmapsto W \longmapsto D$$

inside the displayed DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of* $L = (E, T, W, D, R, S, H, C, P, N)$. *The window route is not a second ledger, a hidden threshold, or an unbounded search.*

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.DyadicTailRadiusLedgerCarrier window route scope`

Proof. Apply tail-budget coverage. It places the precision request, finite tail-window ledger, StreamName windows, and dyadic readback inside E, T, W, D before any radius read is admitted.

Use route replay to see that the same segment is carried by the displayed Cont rows in C. Transport stability keeps the segment aligned under H, and provenance exhaustion attaches the same Pkg and NameCert rows P, N.

Finally use non-escape. Since no second ledger, hidden threshold, or unbounded search is a carrier coordinate or a continuation output, the finite window route is the whole admitted scope for tail-radius reads. □

Theorem 757.21 (Dyadic tail-radius ledger real seal scope)**.** *For an accepted* DyadicTailRadiusLedger$^{\uparrow}$ *carrier, every* Real$^{\uparrow}$ *seal read available to a completion-facing consumer factors through*

$$E \longmapsto T \longmapsto W \longmapsto D \longmapsto R \longmapsto S$$

over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Thus the seal row* S *is scoped by the same route and cannot be read as an independent Real carrier.*

Proof. Start with the window route scope. It binds every admitted radius read to the finite precision, tail-window, StreamName, and DyadicRatCore segment E, T, W, D.

The RegSeqRat and Real handoff theorem extends that segment through R and then S. The shared-tail service boundary identifies this as the service route consumed by downstream Real and Completion-facing users.

Route replay and provenance exhaustion attach C, P, N to the same path, while transport stability carries it through H. Non-escape then rules out an independent Real carrier, selected limit, host real equality, or ambient completeness datum. □

Theorem 757.22 (Dyadic tail-radius ledger mature consumer package)**.** *For an accepted* DyadicTailRadiusLedger$^{\uparrow}$ *carrier, the carrier row of Definition 757.1, tail-budget coverage of Definition 757.3, RegSeqRat and Real handoff of Definition 757.6, and non-escape surface of Definition 757.7 form the mature public package for Real and Completion consumers over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealObservationBudget$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Proof. The carrier definition fixes the finite packet rows $E, T, W, D, R, S, H, C, P, N$. Tail-budget coverage then binds every precision, tail-window, StreamName, and DyadicRatCore read to the finite segment $E \longmapsto T \longmapsto W \longmapsto D$.

The RegSeqRat and Real handoff theorem extends the same segment through R and S. Thus Real and Completion consumers receive a public route only after the dyadic readback and regular-rational handoff rows have been displayed.

The non-escape theorem excludes quotient stream equality, host real equality, selected limits, unbounded schedules, hidden tail search, and ambient completeness. Together with route replay, provenance exhaustion, dependency scope, window route scope, and real seal scope, these rows supply the whole mature package rather than a merely local tail ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}\uparrow}$.

Constructive story: A $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ packet is generated from a unary precision request, finite tail-window radius ledger, finite StreamName windows, DyadicRatCore readback rows, RegSeqRat handoff rows, RealUp seal rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local naming row.

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusLedgerUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 757.1, Definition 757.2, Definition 757.3, Definition 757.4, Definition 757.5, Definition 757.6, Definition 757.7, Definition 757.8, Definition 757.9, Definition 757.10, Definition 757.11, Definition 757.14, Definition 757.17, Definition 757.18, Definition 757.19, Definition 757.20, Definition 757.21, and Definition 757.22 bind the finite tail-radius ledger carrier, its local NameCert obligation surface, transport stability, route replay, provenance exhaustion, shared-tail service boundary, budget-selector pullback, common-refinement pullback, budget monotonicity, dependency scope, window route scope, real seal scope, L10 scoped route package, and mature Real/Completion consumer package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicTailRadiusLedger}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DyadicTailRadiusLedgerUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host real equality, selected limits, unbounded schedules, hidden tail search, or a public standard bridge.

Upgrade path: No further paper-side theory-axis upgrade is required; remaining work belongs to verification-axis and bridge-axis closure.

Chapter 758

A Concrete Naming Certificate for $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$

The $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ packet realizes the candidate-mediated route of chapter 9383 as finite NameCert data. It sits beside the typed reduction normal-form packet of chapter 562, the normal-form consistency seal of chapter 740, and the normal false exclusion packet of chapter 732: those siblings consume normal-form and consistency-facing rows, while this packet keeps the strong-normalization route tied to displayed candidate evidence.

Definition 758.1 (Candidate evidence SN route carrier)**.** *A* $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, K, M, A, I, H, C, P, N)$$

with the following displayed rows:

1. *E is the candidate evidence row naming* Candidate *witnesses and their candidate-to-strong-normalization read;*
2. *K is the closed-candidate row naming closed type, closed term, and closed candidate evidence;*
3. *M is the candidate-set membership row that exposes the finite membership and soundness reads used by closed typed terms;*
4. *A is the Pi and application adequacy row, including function candidate evidence, argument candidate evidence, and the application candidate readback;*
5. *I is the no-infinite-reduction projection row attached to the strong-normalization evidence already displayed in E, K, M, A;*
6. *H stores componentwise* hsame *transports for candidate evidence, closed-candidate evidence, membership evidence, adequacy rows, and no-infinite projections;*
7. *C stores the displayed* Cont *routes from candidate evidence to strong normalization and from strong normalization to no-infinite-reduction boundaries;*
8. *P records* Pkg *provenance over the normalization audit map, typed reduction normal forms, normal-form consistency seals, normal false exclusion, and the MetaCIC candidate evidence rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateEvidenceSNRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for evidence-free global strong normalization, an unconditional application-normalization theorem, a host evaluator, quotient equality over reduction traces, or a hidden tail of reduction steps.

Lean4 ✓ `BEDC.Derived.CandidateEvidenceSNRouteUp.CandidateEvidenceSNRouteTasteGate single carrier alignment`

Theorem 758.2 (Candidate evidence SN route obligation surface)**.** *Every accepted* $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ *packet binds exactly the obligation surface displayed in Definition 758.1: candidate evidence through* E*, closed-candidate evidence through* K*, candidate-set membership through* M*, Pi and application adequacy through* A*, no-infinite-reduction projection through* I*, componentwise transport through* H*, route replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier coordinates. The positive route begins with E, K, M, A: a consumer first reads candidate evidence, closed-candidate evidence, candidate-set membership, or Pi and application adequacy before any strong-normalization consequence is visible.

The row I is downstream from those evidence rows. It records no-infinite-reduction only as a projection from displayed strong-normalization evidence, so the packet never creates a second route that skips the candidate surface.

The structural rows H, C, P, N preserve, replay, package, and name only the listed coordinates. Since the carrier contains no further row, the displayed rows exhaust the local NameCert obligation surface. □

Theorem 758.3 (Candidate evidence SN route no-infinite boundary)**.** *Every no-infinite-reduction read exported by an accepted* $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ *packet factors through one of the displayed candidate-mediated strong-normalization rows* E, K, M, A *before it reaches* I*. Thus the route can feed* $\mathsf{TypedReductionNormalForm}^{\uparrow}$*,* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$*, and* $\mathsf{NormalFalseExclusion}^{\uparrow}$ *consumers, but it cannot be reread as blanket strong normalization for every closed typed MetaCIC term.*

Proof. Start from a no-infinite-reduction read of the packet. The only no-infinite coordinate is I, and its input is the strong-normalization evidence already supplied by E, K, M, A.

Use the route replay row C to trace the read backward to candidate evidence, closed-candidate evidence, candidate-set membership, or Pi and application adequacy. The transport row H preserves those evidence coordinates under accepted packet agreement.

Finally send the resulting boundary to the normal-form consumers named in P. Each consumer receives a finite candidate-mediated route, not a host evaluator, quotient reduction equality, unconditional application-normalization theorem, or evidence-free global strong-normalization principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateEvidenceSNRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ packet is generated from candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy rows, no-infinite-reduction projections, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateEvidenceSNRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 758.1, Definition 758.2, and Definition 758.3 bind the finite candidate-mediated strong-normalization route and its no-infinite-reduction boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateEvidenceSNRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close evidence-free global strong normalization, unconditional application normalization, full MetaCIC subject reduction, quotient equality over reductions, host evaluation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-route exactness, no-infinite projection exactness, provenance, and local NameCert rows.

Chapter 759

A Concrete Naming Certificate for $\mathsf{InterHistLocalityLedger}^{\uparrow}$

The $\mathsf{InterHistLocalityLedger}^{\uparrow}$ packet realizes the ledger-local relation principle of chapter 9419 as a finite two-history NameCert carrier. It sits between the two-observer coordination surface of chapter 513 and the invariant readback surface of chapter 656: the packet records two source histories, symmetric transport, displayed routes, locality cells, invariant rows, provenance, and local naming without exporting a global observer frame.

Definition 759.1 (InterHistLocalityLedger carrier)**.** *An* $\mathsf{InterHistLocalityLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, H, C, O, I, P, N)$$

with the following displayed coordinates:

1. *A and B are the two source* BHist *rows whose local relation is being certified;*
2. *H stores symmetric componentwise* hsame *transport between the two source-history presentations and their transported rows;*
3. *C stores the displayed* Cont *routes by which the inspected finite data are read from A and B and replayed across the ledger;*
4. *O is the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row that supplies the two-observer locality cell consumed by the ledger;*
5. *I is the* $\mathsf{InterHistInvariant}^{\uparrow}$ *row that records the observer-symmetric inter-Hist invariant readback;*
6. *P is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, *and* $\mathsf{NameCert}$*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{InterHistLocalityLedger}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a global frame, shared clock, selected synchronization history, host-side equality of histories, quotient of observer records, or ambient spacetime.

Lean4 ✓ `BEDC.Derived.InterHistLocalityLedgerUp.InterHistLocalityLedgerTasteGate single carrier alignment`

Theorem 759.2 (InterHistLocalityLedger NameCert obligations)**.** *For the carrier of Definition 759.1, the local* $\mathsf{NameCert}_{\mathsf{InterHistLocalityLedger}^{\uparrow}}$ *surface consists of carrier admission for $L = (A, B, H, C, O, I, P, N)$,*

symmetric hsame *transport for the two source histories, displayed* Cont *route replay, observer-locality-cell consumption, inter-Hist invariant readback,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed tuple L. The rows A and B give the two source histories, while H and C give the transport and continuation data used to compare their finite observations.

The rows O and I supply the two local consumers. The observer-locality-cell row records the finite two-observer surface, and the invariant row records the observer-symmetric readback over the same pair.

The provenance and naming clauses are P and N. Since the carrier has no coordinate for a global frame, shared clock, synchronization history, quotient of observers, host-side history equality, or ambient spacetime, these displayed rows exhaust the local certificate obligations. □

Theorem 759.3 (InterHistLocalityLedger symmetric route)**.** *Let $L = (A, B, H, C, O, I, P, N)$ be an accepted* InterHistLocalityLedger$^{\uparrow}$ *carrier. Swapping the source-history rows A and B yields the same public locality ledger after replaying the symmetric* hsame *transport in H and the displayed* Cont *routes in C; the observer-locality cell and inter-Hist invariant rows are read through the same finite packet.*

Lean4 ✓ `BEDC.Derived.InterHistLocalityLedgerUp.InterHistLocalityLedger symmetric route source swap separates`

Proof. Project the accepted carrier. The only source-history coordinates are A and B, and the only transport coordinate relating their presentations is H.

Replay the continuation routes stored in C. These routes attach the inspected finite data from both source histories to the same locality-cell row O and invariant row I.

The NameCert obligations of Definition 759.2 then identify the swapped read as a route through the same H, C, O, I, P, N surface. No global synchronization coordinate is available to distinguish one source history as the privileged anchor. □

Theorem 759.4 (InterHistLocalityLedger non-escape)**.** *Every public consumer of an accepted* InterHistLocalityLedger$^{\uparrow}$ *carrier may inspect only the source-history rows A, B, the symmetric* hsame *transport row H, the displayed* Cont *route row C, the* ObserverLocalityCell$^{\uparrow}$ *row O, the* InterHistInvariant$^{\uparrow}$ *row I,* Pkg *provenance P, and local naming row N. It receives no global frame, shared clock, selected synchronization history, host-side history equality, quotient of observer records, metric spacetime, or all-observer synchronization map.*

Proof. Inspect the carrier coordinates in Definition 759.1. The visible rows are exactly A, B, H, C, O, I, P, N.

The observer-locality-cell and invariant rows are downstream reads of that finite packet, not extra ambient structures. They preserve the same two-history boundary and do not add a frame, clock, quotient, or host equality coordinate.

Therefore every consumer is exhausted by the displayed rows. Any request for global synchronization, ambient spacetime, host-side equality of histories, or an all-observer map would require a carrier coordinate that is absent from L. □

Theorem 759.5 (InterHistLocalityLedger observer-locality handoff)**.** *An accepted* InterHistLocalityLedger$^{\uparrow}$ *carrier hands the two source histories to an* ObserverLocalityCell$^{\uparrow}$ *consumer and an* InterHistInvariant$^{\uparrow}$ *consumer through one finite route packet*

$$A, B, H, C, O, I, P, N.$$

The handoff preserves the symmetric locality relation of the ledger and does not add a cross-Hist causal predicate, maximum-rate bound, global frame, or metric geometry.

Proof. Begin with the carrier definition. The rows A and B are the only source histories, O is the observer-locality-cell row, and I is the inter-Hist invariant row.

Use the symmetric route theorem to transport the two source histories through H and replay their inspected finite data through C. This makes the locality-cell and invariant consumers read the same two-history ledger.

Apply non-escape for the boundary. Since L has no coordinate for cross-Hist causal predicates, maximum-rate bounds, global frames, metric geometry, or host-side equality, the observer-locality handoff is exactly the finite packet A, B, H, C, O, I, P, N. □

Lean4 ✓ `BEDC.Derived.InterHistLocalityLedgerUp.InterHistLocalityLedgerLocalityCell handoff separation`

Falsifiable prediction: Removing either the symmetric transport row or the observer-locality handoff row leaves a packet that can still name two histories but cannot certify a local two-history ledger.

Independence witness: Nearest siblings: observerlocalitycell, interhistinvariant, crosshistcausalrate, inscriptionevent, and localselfcenterinscriptionseal. ObserverLocalityCellUp records a cell rather than the ledger route; InterHistInvariantUp records invariant readback without the two-history locality ledger; CrossHistCausalRateUp records rate data rather than symmetric locality transport; InscriptionEventUp records an event packet rather than inter-history ledger routing; LocalSelfCenterInscriptionSealUp records a local self-center seal rather than a two-observer handoff. No bijection preserving carrier rows, symmetric hsame transport, observer-locality handoff, invariant readback, provenance, and local naming can identify these packets.

Closure status for $\mathsf{NameCert}_{\mathsf{InterHistLocalityLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{InterHistLocalityLedger}^{\uparrow}$ packet is generated from two source histories, symmetric transport, displayed route replay, an observer-locality cell, an inter-Hist invariant row, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.InterHistLocalityLedgerUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InterHistLocalityLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 759.5 and

Lean4 ✓ `BEDC.Derived.InterHistLocalityLedgerUp.taste gate`

publish the finite carrier, symmetric hsame transport, displayed Cont replay, observer-locality consumption, invariant readback, Pkg provenance, local NameCert row, and TasteGate obligations for the displayed two-history ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InterHistLocalityLedger}^{\uparrow}}$, unformalized)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.interHistLocalityLedgerDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global observer frames, shared clocks, selected synchronization histories, host-side history equality, metric spacetime, cross-Hist causal bounds, or bridge checking.

Upgrade path: Remaining work is Lean-side route coverage and bridge checking against the public schema.

Chapter 760

A Concrete Naming Certificate for $\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}$

The $\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}$ packet realizes the congruence-trigger boundary isolated in chapter 9384 as a finite MetaCIC NameCert surface. It sits next to the bounded comparison packet of chapter 748, the beta-substitution preservation packet of chapter 716, and the discharge socket of chapter 719. The packet records where a congruence constructor can enter the subject-reduction route, without turning that constructor into a global typing-preservation theorem.

Definition 760.1 (Typed beta congruence trigger carrier)**.** *A* $\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (k, r, s, j, b, d, H, C, P, N)$$

with the following displayed rows:

1. *k is the congruence-constructor tag, one of application-function, application-argument, lambda-body, lambda-domain, Pi-codomain, or Pi-domain;*
2. *r is the local* BetaStep *row naming the syntactic trigger;*
3. *s is the displayed source term row before the congruence step;*
4. *j is the typing-boundary judgement row whose preservation is being handed to the relevant discharge surface;*
5. *b is the beta audit-map boundary row that records whether the constructor is direct substitution-facing, dependent-codomain-facing, or binder-domain-facing;*
6. *d is the sibling discharge row read from the substitution or subject-reduction discharge packet;*
7. *H records componentwise* hsame *transports for the constructor tag, beta-step row, source term, judgement boundary, audit boundary, discharge row, provenance, and local name;*
8. *C records the displayed* Cont *route from the constructor trigger to the discharge surface;*
9. *P is the* Pkg *provenance over* BetaStep*, MetaCIC term syntax, typing-boundary rows, chapter 9384, chapter 748, chapter 716, chapter 727, and chapter 718;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}}$ *row.*

The carrier contains no conversion rule for typing, no full subject-reduction theorem, no fuel-erased normalisation theorem, and no quotient equality over represented terms.

Lean4 ✓ `BEDC.Derived.TypedBetaCongruenceTriggerUp.TypedBetaCongruenceTriggerTasteGate single carrier alignment`

Theorem 760.2 (Typed beta congruence trigger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TypedBetaCongruenceTrigge}}$ *surface for Definition 760.1 has five finite obligations:*

1. *trigger admission, requiring* $T = (k, r, s, j, b, d, H, C, P, N)$ *to display a concrete* BetaStep *congruence constructor and its source term row;*
2. *congruence-constructor classification, requiring* k *to identify whether the step is application-function, application-argument, lambda-body, lambda-domain, Pi-codomain, or Pi-domain;*
3. *typing-boundary preservation handoff, requiring the judgement boundary* j *and audit row* b *to point to the sibling substitution or subject-reduction discharge row* d *rather than to an implicit typing rule;*
4. *route non-escape, requiring every consumer to pass through the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows before it can read the discharge surface;*
5. *ledger exactness, requiring the exported rows to be exactly* $k, r, s, j, b, d, H, C, P, N$ *and no hidden conversion, closed subject-reduction, or normalisation ledger.*

Proof. Carrier admission is projection from the finite tuple in Definition 760.1. The constructor tag, beta-step row, source term, judgement boundary, audit row, discharge row, transport, route, provenance, and local name are all displayed coordinates.

The congruence classification is rowwise. The beta audit map separates the direct beta constructor from the congruence constructors, and the carrier records only the congruence side through k and r.

The preservation handoff is therefore a route fact. The row j is read together with b and then handed to d, so the packet cites the substitution and subject-reduction discharge siblings instead of asserting a new typing-preservation rule.

Finally inspect H, C, P, N. These rows provide the only transport, continuation, provenance, and naming paths. Since no coordinate contains conversion typing, full closed subject reduction, fuel-erased normalisation, or quotient equality, the exported ledger is exactly the displayed finite packet. □

Theorem 760.3 (Typed beta congruence trigger audit-map fit)**.** *Every accepted* $\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}$ *packet presents a concrete consumer surface for the congruence-trigger boundary of section 9384.4. The direct substitution-facing trigger is routed to the beta-substitution preservation and discharge siblings, while the dependent-codomain and binder-domain congruence triggers are routed to the subject-reduction discharge siblings. The packet exports no theorem beyond that finite routing surface.*

Proof. Read the audit-map section first. It separates the direct beta constructor from the application-argument and binder-domain congruence constructors, and it explains which typing boundary each constructor touches.

Apply Definition 760.2. The constructor tag and audit row classify the trigger, while the judgement boundary and discharge row give the only preservation handoff available to consumers.

The sibling provenance row then confines the packet to the concrete BEDC surfaces cited in Definition 760.1. Since the carrier has no coordinate for a global typing rule, it can only expose the finite route from congruence constructor to discharge boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}$ packet is generated from a congruence-constructor tag, a concrete BetaStep trigger row, a source term row, a typing-boundary judgement row, a beta audit-map boundary row, a sibling discharge row, componentwise transports, continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 760.1, Definition 760.2, and Definition 760.3 bind the finite congruence-trigger carrier, classification, handoff, route, and exact-ledger surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypedBetaCongruenceTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close conversion typing, global closed subject reduction, fuel-erased normalisation, quotient equality over represented terms, or bridge checking.

Upgrade path: Obligation closure requires checking trigger admission, constructor classification, typing-boundary handoff, route non-escape, and ledger exactness for the displayed packet.

Chapter 761

A Concrete Naming Certificate for $\mathsf{MetaCICBetaAudit}^{\uparrow}$

The $\mathsf{MetaCICBetaAudit}^{\uparrow}$ packet realizes the audit discipline of chapter 9384 as a finite NameCert boundary. It keeps the reduction inventory, common-reduct conversion facts, normal-form rigidity rows, and subject-reduction obstruction rows as separate coordinates, so a consumer cannot treat beta reduction, beta conversion, and typing preservation as one unrecorded operation. The packet reads the obstruction side through chapter 9377 and its concrete NameCert surfaces in chapter 483 and chapter 562.

Definition 761.1 (MetaCIC beta audit carrier)**.** *A* $\mathsf{MetaCICBetaAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, V, T, F, O, H, C, P, N)$$

with the following visible coordinates:

1. *S is the* BetaStep *constructor inventory, separating the direct beta-redex constructor from application, lambda, and Pi congruence constructors;*
2. *V is the* BetaConv *common-reduct ledger, recording the two* BetaStarStep *routes to the displayed common reduct;*
3. *T is the* BetaStarStep *route data consumed by conversion and by one-step-to-multi-step inclusions;*
4. *F is the normal-form rigidity row, read only under the displayed closed beta-strong normal hypotheses;*
5. *O is the subject-reduction obstruction ledger, with rows for substitution preservation, application-argument type stability, lambda-domain preservation, and Pi-domain preservation;*
6. *H records componentwise* hsame *transports for the reduction, conversion, route, rigidity, and obstruction rows;*
7. *C records the displayed* Cont *routes by which consumers inspect those rows;*
8. *P is the* Pkg *provenance over the MetaCIC beta files, the subject-reduction obstruction surface,* BHist*,* hsame*,* Cont*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICBetaAudit}^{\uparrow}}$ *row.*

The carrier contains no hidden typing-conversion rule, no unconditional subject-reduction theorem, no closed-consistency theorem, and no extra beta constructor beyond the displayed inventory.

Lean4 ✓ `BEDC.Derived.MetaCICBetaAuditUp.MetaCICBetaAuditTasteGate single carrier alignment`

Theorem 761.2 (Reduction and conversion boundary)**.** *For an accepted* $\mathsf{MetaCICBetaAudit}^{\uparrow}$ *carrier* $A = (S, V, T, F, O, H, C, P, N)$*, every reduction read factors through the* BetaStep *inventory* S *and every conversion read factors through the* BetaConv *common-reduct ledger* V *together with the* BetaStarStep *route data* T*. A consumer may use the normal-form rigidity row* F *only under its displayed closed beta-strong normal hypotheses, and it has no route from* S, V, T, F *to a typing-preservation claim.*

Lean4 ✓ `BEDC.Derived.MetaCICBetaAuditUp.MetaCICBetaAuditReductionConversionBoundary`

Proof. Project the carrier. The row S is the only one-step reduction inventory, while V and T are the only common-reduct and multi-step route rows.

Read conversion through V. Its data is a common reduct and two routes supplied by T, not a typing judgement and not a closure certificate.

The rigidity row F is then read under its own closed normal-form hypotheses. Since the carrier keeps O as a separate obstruction ledger and has no hidden typing-conversion coordinate, reduction and conversion cannot be promoted to subject reduction by projection. □

Lean4 ✓ `BEDC.Derived.MetaCICBetaAuditUp.MetaCICBetaAudit reduction conversion boundary`

Theorem 761.3 (Subject-reduction obstruction ledger)**.** *For an accepted* $\mathsf{MetaCICBetaAudit}^{\uparrow}$ *carrier* $A = (S, V, T, F, O, H, C, P, N)$*, every subject-reduction citation factors through the obstruction row* O*. The ledger* O *separates beta substitution preservation, dependent application-argument stability, lambda-domain preservation, and Pi-domain preservation, and it is the only coordinate that may be read by consumers of chapter 9377.*

Proof. Inspect the carrier rows. The reduction inventory S, conversion ledger V, route data T, and rigidity row F do not contain typing-preservation fields.

The obstruction row O is the unique coordinate listing the four preservation interfaces needed by the subject-reduction boundary. Componentwise transport through H and route replay through C preserve those four rows without merging them.

Package provenance P and the local name N record that the obstruction ledger is part of the same finite packet. Because no other coordinate carries typing preservation, a subject-reduction consumer must enter through O. □

Lean4 ✓ `BEDC.Derived.MetaCICBetaAuditUp.MetaCICBetaAudit subject reduction obstruction ledger`

Theorem 761.4 (MetaCIC beta audit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICBetaAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for* $A = (S, V, T, F, O, H, C, P, N)$*, reduction inventory routing through* S*, common-reduct conversion routing through* V, T*, normal-form rigidity routing through* F*, subject-reduction obstruction routing through* O*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Start from the carrier definition. It displays the reduction, conversion, route, rigidity, obstruction, transport, continuation, provenance, and local naming coordinates.

Use Definition 761.2 for the reduction and conversion side. It keeps one-step reduction, common-reduct conversion, multi-step routes, and normal-form rigidity in their own rows.

Use Definition 761.3 for the typing side. It makes O the only row that can serve subject-reduction consumers and keeps the four preservation obligations separate.

The rows H, C, P, N then account for transport, route replay, provenance, and naming. Since the packet has no hidden conversion-for-typing coordinate, no unconditional subject-reduction coordinate, and no closed-consistency coordinate, the obligation surface is exhausted by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICBetaAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICBetaAudit}^{\uparrow}$ packet is generated from a BetaStep constructor inventory, BetaConv common-reduct ledger, BetaStarStep route data, normal-form rigidity row, subject-reduction obstruction ledger, componentwise transports, displayed continuation routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICBetaAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 761.1, Definition 761.2, Definition 761.3, and Definition 761.4 bind the finite carrier and the reduction, conversion, rigidity, and typing-obstruction boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICBetaAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close unconditional subject reduction, hidden conversion for typing, closed consistency, new beta constructors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked routing for the displayed reduction inventory, conversion ledger, route data, rigidity row, obstruction ledger, transport rows, continuation rows, provenance row, and local name.

Chapter 762

A Concrete Naming Certificate for MetaCICOpenFrontierLedger$^{\uparrow}$

The MetaCICOpenFrontierLedger$^{\uparrow}$ packet realizes the open-problem inventory of chapter 9386 as a finite NameCert frontier ledger. It sits beside the subject-reduction carrier of chapter 483, the normal false exclusion packet of chapter 732, the normalization audit packet of chapter 750, and the decidability boundary packet of chapter 749. Its purpose is to name the open rows that downstream consumers must keep visible before reading MetaCIC consistency, normalization, confluence, decidability, or typed-example evidence.

Definition 762.1 (MetaCIC open frontier ledger carrier)**.** *A* MetaCICOpenFrontierLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, E, F, N, D, T, K, H, C, P, Q)$$

with the following displayed rows:

1. *S is the subject-reduction frontier row, naming the* SubjectReductionDischargeBundle *interface and its four obligation coordinates;*
2. *E is the closed-normal exclusion row, naming the closed-normal false exclusion and closed-normal consistency boundary;*
3. *F is the confluence frontier row, naming the residual-substitution and arbitrary closed-source Church–Rosser endpoints that remain outside the closed-normal fragment;*
4. *N is the normalization frontier row, naming the candidate-mediated strong-normalization interface and its evidence boundary;*
5. *D is the decidability frontier row, naming typing-checker exactness, bounded normal comparison, finished-normal side conditions, and the refusal of an unrestricted closed beta-conversion oracle;*
6. *T is the typed-example boundary row, naming the gallery witnesses and the absence of exhaustive dependent-program coverage;*
7. *K is the closed-consistency boundary row, naming the handoff from closed-normal consistency to any stronger closed consistency claim;*
8. *H records componentwise* hsame *transport across $S, E, F, N, D, T, K, P, Q$;*
9. *C records displayed* Cont *replay routes for the subject-reduction, confluence, normalization, decidability, typed-example, and consistency-boundary rows;*

10. *P is the* Pkg *provenance over the MetaCIC open-problem inventory and the sibling concrete NameCert packets cited above;*

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}}$ *row.*

The carrier has no row for closed subject reduction, no arbitrary closed Church–Rosser theorem, no evidence-free global strong normalization theorem, no unrestricted closed beta-conversion decider, no exhaustive typed-example coverage theorem, and no full closed consistency claim.

Theorem 762.2 (MetaCIC open frontier ledger obligations)**.** *Every accepted* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *packet binds exactly the finite obligation surface displayed in Definition 762.1: carrier admission for $L = (S, E, F, N, D, T, K, H, C, P, Q)$, subject-reduction frontier visibility through S, closed-normal exclusion visibility through E, confluence-frontier visibility through F, normalization-frontier visibility through N, decidability-frontier visibility through D, typed-example boundary visibility through T, consistency-boundary visibility through K, componentwise* hsame *transport through H, finite* Cont *replay through C, package provenance through P, and local naming through Q.*

Proof. Project the displayed carrier. The frontier rows are S, E, F, N, D, T, K, and each row corresponds to one boundary named in chapter 9386.

Read the positive closed-normal row E separately from the open rows S, F, N, D, T, K. This separation lets the ledger record the existing closed-normal exclusion surface without promoting it into a stronger endpoint.

The structural rows H, C, P, Q only transport, replay, package, and name those displayed rows. Since no further coordinate is present, the listed rows exhaust the local NameCert obligation surface. □

Theorem 762.3 (MetaCIC open frontier ledger subject-reduction boundary)**.** *Every accepted* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *packet exposes subject reduction only through the frontier row S and the sibling packet of chapter 483. The row may name the* SubjectReductionDischargeBundle *interface and its beta-substitution, application-argument, lambda-domain, and Pi-domain coordinates, but it does not inhabit that bundle, prove the four coordinates, or export closed subject reduction.*

Proof. Begin with the carrier row S. It is the only coordinate in the packet whose role is to name the subject-reduction frontier.

The sibling reference to chapter 483 supplies the accepted preservation-facing surface, while S records the discharge interface that must still be supplied before the closed MetaCIC path can consume it as an unconditional theorem.

Transport through H and replay through C preserve this row as displayed. The package and local name rows P, Q account for provenance and naming, so no hidden inhabitant of the discharge bundle or closed subject-reduction theorem can enter the ledger. □

Theorem 762.4 (MetaCIC open frontier ledger normalization boundary)**.** *Agreement on an accepted* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *packet preserves the closed-normal exclusion row E and the normalization frontier row N as distinct coordinates. The packet may feed closed-normal consistency consumers through the closed-normal exclusion surface of chapter 732 and the candidate-mediated audit surface of chapter 750, but it cannot erase candidate evidence, finished-normal evidence, or subject-reduction transport into a global closed consistency theorem.*

Proof. Read E as the closed-normal exclusion coordinate. It records the finite boundary at which a closed beta-normal proof of the internal false type is excluded.

Read N as the normalization-frontier coordinate. It names the candidate-mediated route and its evidence boundary, matching the sibling audit surface rather than a blanket strong-normalization theorem.

The consistency row K is downstream of both coordinates and remains a boundary row. Thus H, C, P, Q can replay and name the closed-normal route, but they cannot identify E with global normalization, erase the candidate evidence in N, or convert the ledger into full closed consistency. □

Theorem 762.5 (MetaCIC open frontier ledger downstream handoff)**.** *Every downstream read of an accepted* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *carrier factors through*

$$S, E, F, N, D, T, K, H, C, P, Q.$$

The handoff may cite subject-reduction obligations, closed-normal exclusion, confluence boundaries, candidate-mediated normalization, bounded decidability, typed-example gallery scope, and closed-consistency boundaries, but it may not replace those rows by vision prose, host metatheory, quotient equality over represented terms, or an unrecorded global MetaCIC theorem.

Proof. Project the carrier and apply the obligation theorem. The downstream consumer receives the seven frontier rows together with transport, replay, provenance, and local naming rows.

The confluence row F, decidability row D, typed-example row T, and consistency row K are boundary coordinates rather than theorem endpoints. They keep residual substitution, bounded normal comparison, gallery scope, and full consistency separate from the positive closed-normal and checker-side data.

Since the carrier contains no host evaluator, quotient equality, erased normalizer, unrestricted conversion oracle, or global consistency endpoint, every accepted handoff remains inside the displayed ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ packet is generated from finite subject-reduction, closed-normal exclusion, confluence, normalization, decidability, typed-example, consistency-boundary, transport, continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 762.1, Definition 762.2, Definition 762.3, Definition 762.4, and Definition 762.5 bind the finite open-frontier ledger and its downstream handoff discipline.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive typed-example coverage, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exact row coverage for the seven frontier coordinates, sibling provenance, continuation replay, and non-escape for every downstream handoff.

Chapter 763

A Concrete Naming Certificate for MetaCICOpenObligationLedger$^{\uparrow}$

The MetaCICOpenObligationLedger$^{\uparrow}$ packet realizes the inventory of chapter 9386 as finite BEDC data. It sits beside the closed-normal consistency assembly of chapter 9375 and the substitution audit map of chapter 9380: those chapters identify positive closed-normal content and substitution-facing boundaries, while this packet keeps every remaining MetaCIC obligation as a named ledger row rather than as an ambient metatheorem.

As a concrete dependency edge, the ledger consumes chapter 736 for the positive normal-form node and chapter 779 for the substitution-facing boundary rows; the local packet records only the remaining open obligation rows.

Theorem 763.1 (MetaCIC open obligation decidability endpoint row)**.** *Any* MetaCICOpenObligationLedger$^{\uparrow}$ *consumer attempting to read a closed-term beta-conversion decision or a bounded normal-equality decision must expose a distinct decidability endpoint row. That row is not supplied by the closed-normal consistency row, substitution boundary row, subject-reduction discharge rows, confluence obstruction rows, componentwise transport, replay, provenance, or local naming.*

Proof. Project the ledger carrier. Its decidability coordinate is separate from the subject-reduction, confluence, normalization, typed-example, transport, replay, provenance, and local naming coordinates.

Use the dependency edge to the closed-normal consistency and substitution audit map chapters. Those inputs identify positive normal-form content and substitution-facing boundary data, but neither edge supplies a decision procedure for closed beta conversion or bounded normal equality.

Therefore a consumer that asks for such a decision must read the displayed decidability endpoint row itself. The obligation remains a finite endpoint row of the ledger rather than a consequence hidden inside normalization, substitution, transport, replay, provenance, or naming. □

Definition 763.2 (MetaCIC open-obligation ledger carrier)**.** *A* MetaCICOpenObligationLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, C, N, D, E, H, R, P, Q)$$

with the following displayed rows:

1. *S is the subject-reduction row, containing the beta-substitution preservation, application-argument stability, lambda-domain preservation, and Pi-domain preservation obligations as finite named entries;*

2. *C is the confluence row, recording residual-substitution compatibility, closed parallel diamond scope, and closed multi-step Church–Rosser scope as boundary entries;*
3. *N is the normalization row, recording candidate evidence, closed-candidate evidence, candidate-set membership, and the absence of an evidence-free global strong-normalization export;*
4. *D is the decidability row, recording typing-checker coverage, bounded normal-equality reads, and the finished-normalization bridge needed before unrestricted beta conversion is exported;*
5. *E is the typed-example row, recording gallery witnesses and the absence of an exhaustive coverage theorem over all dependent or polymorphic programs;*
6. *H stores componentwise* hsame *transports for S, C, N, D, E;*
7. *R stores the finite* Cont *routes by which consumers read the subject-reduction, confluence, normalization, decidability, and example rows;*
8. *P is the* Pkg *provenance over the open-problem inventory, closed-normal consistency assembly, substitution audit map, subject-reduction obstruction, and normalization audit map;*
9. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenObligationLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted beta-conversion decidability, exhaustive example coverage, or full closed consistency.

Theorem 763.3 (MetaCIC open-obligation ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenObligationL}}$ *surface for Definition 763.2 consists exactly of carrier admission for L, componentwise classifier stability for S, C, N, D, E, route replay through R, provenance through P, and local naming exactness through Q.*

Proof. Carrier admission is projection from the finite packet L. Each obligation family is a displayed row, so a consumer cannot read a subject-reduction, confluence, normalization, decidability, or typed-example boundary unless the corresponding coordinate is present.

Classifier stability is componentwise. The transport row H preserves each ledger row as a ledger row; it does not convert a boundary entry into a closed theorem and does not erase the distinction between positive checked content and remaining obligation data.

The route and provenance rows repack the same finite packet. The routes in R replay the displayed rows, P records their source chapters, and Q names the local certificate surface without adding another MetaCIC claim. □

Theorem 763.4 (MetaCIC open-obligation boundary classifier)**.** *For an accepted* $\mathsf{MetaCICOpenObligationLedger}^{\uparrow}$ *carrier $L = (S, C, N, D, E, H, R, P, Q)$, every ledger entry is classified by its row of origin and by its destination boundary: subject-reduction discharge, residual-substitution algebra, candidate-mediated normalization, bounded conversion, or gallery coverage. The classifier preserves the closed-normal consistency node of chapter 9375 as a positive input row and the substitution map of chapter 9380 as a boundary input row.*

Proof. Read the entry through the route row R. If it lands in S, the destination boundary is a subject-reduction discharge; if it lands in C, it is a residual-substitution or closed-confluence boundary; if it lands in N, it is a candidate-mediated normalization boundary; if it lands in D, it is a bounded-conversion boundary; and if it lands in E, it is a gallery-coverage boundary.

The sibling chapters are consumed through provenance rather than by unstated closure. The closed-normal assembly supplies the positive normal-form node, while the substitution audit map

supplies the substitution-facing obstruction data that keep confluence and subject-reduction reads row-local.

Componentwise hsame transport preserves the row tags. Therefore the classifier remains a boundary classifier over the finite packet and not an equality of unresolved MetaCIC proof obligations. □

Theorem 763.5 (MetaCIC open-obligation non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetaCICOpenObligationLedger}^{\uparrow}$ *packet factors through the displayed rows* $S, C, N, D, E, H, R, P, Q$*. A request for arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted beta-conversion decidability, exhaustive example coverage, or full closed consistency has no carrier coordinate and remains a boundary read.*

Proof. Let a downstream consumer route be given. By the carrier definition, the route can inspect only the five obligation rows, the componentwise transport row, the replay routes, package provenance, and local naming.

Positive content is still visible where the packet records it. The closed-normal consistency node is consumed as provenance for the normal-form row, and the substitution map is consumed as provenance for the substitution-facing confluence and subject-reduction rows.

The requested closed endpoints are absent from the carrier inventory. Since no displayed coordinate contains those endpoints, the packet can report their boundary status but cannot export them as hidden consequences. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICOpenObligationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICOpenObligationLedger}^{\uparrow}$ packet is generated from subject-reduction, confluence, normalization, decidability, and typed-example obligation rows, componentwise hsame transports, finite replay routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICOpenObligationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 763.2, Definition 763.3, Definition 763.4, and Definition 763.5 bind the finite open-obligation ledger and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICOpenObligationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted beta-conversion decidability, exhaustive example coverage, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row classifier stability, route replay exactness, provenance, and local NameCert rows for the same finite ledger packet.

Chapter 764

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$

The $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ packet realizes the main-result boundary of chapter 9376 as a finite NameCert surface. It is downstream of the closed-normal consistency packet of chapter 736 and the closed-normal consistency packet boundary of chapter 739: those siblings expose the closed beta-strong false-inhabitation exclusion and its subject-reduction gap, while the present packet records exactly the consumer-facing main-result claim.

Definition 764.1 (Closed-normal consistency main carrier)**.** *A* $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, L, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* HasType *row for a closed candidate at the internal false encoding in the empty context;*
2. *L is the* ClosedAt *row binding the same term and false encoding to the closed substrate read used by the main result;*
3. *B is the* BetaStrong *row asserting that the candidate is already at the beta-strong normal boundary consumed by closed-normal consistency;*
4. *F is the false-encoding row naming the contradiction target and no other bottom-like proposition;*
5. *R is the normal-form contradiction route from T, L, B, F to the closed beta-strong false-inhabitation exclusion;*
6. *H records componentwise* hsame *transport for the typing, closedness, beta-strong, false-encoding, and contradiction-route rows;*
7. *C records the displayed* Cont *routes from the typed closed beta-strong input to the contradiction route and from that route to a public read;*
8. *P records* Pkg *provenance over the main-result chapter, the closed-normal consistency packet, and the closed-normal consistency packet boundary;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyMain}^{\uparrow}}$ *row.*

The carrier has no coordinate for full closed consistency, closed subject reduction, unrestricted normalization, host evaluation, quotient conversion, or ambient consistency.

Lean4 (definition) `BEDC.Derived.ClosedNormalConsistencyMainUp.ClosedNormalConsistencyMainUp`

Theorem 764.2 (Closed-normal consistency main NameCert obligations)**.** *Every accepted* ClosedNormalConsistenc *packet binds exactly the finite obligation surface of Definition 764.1: carrier admission for* $Q = (T, L, B, F, R, H, C, P, N)$*, componentwise classifier stability on the displayed rows, ledger exactness for the typing, closedness, beta-strong, false-encoding, contradiction-route, transport, route, provenance, and naming coordinates, positive-boundary non-escape through* R*, and local naming through* N*.*

Proof. Project the accepted carrier. Admission gives the finite tuple $Q = (T, L, B, F, R, H, C, P, N)$, so every public classifier comparison is componentwise over the rows displayed in Definition 764.1.

The ledger exactness is read row by row. The rows T, L, B, F display the closed typed beta-strong false input, and R is the only positive contradiction route attached to that input.

The structural rows H, C, P, N preserve the same finite surface by componentwise transport, displayed route replay, package provenance, and local naming. Since no further coordinate is present, the packet supplies no escape from the stated positive boundary. □

Theorem 764.3 (Closed-normal consistency main ledger obligation scope)**.** *In every accepted* $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ *packet, the rows* $T, L, B, F, R, H, C, P, N$ *are consumed in three scoped groups: endpoint hypotheses* T, L, B, F*, contradiction route* R*, and structural replay, provenance, and name rows* H, C, P, N*. No consumer can read full closed consistency, subject reduction, or unrestricted normalization from this packet, because no row for any of those claims appears in any group.*

Proof. Project the carrier $Q = (T, L, B, F, R, H, C, P, N)$. The endpoint-hypothesis group is exactly T, L, B, F: typing at the false encoding, closedness, beta-strong normality, and the false-encoding row.

The contradiction-route group is exactly R. It consumes the endpoint hypotheses and reaches the closed beta-strong false-inhabitation exclusion, but it does not add a route from arbitrary closed proofs to typed normal endpoints.

The structural group is exactly H, C, P, N. These rows provide componentwise transport, displayed continuation replay, package provenance, and local naming for the same finite surface.

Since full closed consistency, subject reduction, and unrestricted normalization occur in none of the three groups, a consumer has no coordinate from which to read them. The accepted packet therefore exposes only the scoped ledger obligations listed in the statement. □

Theorem 764.4 (Closed-normal consistency main positive boundary)**.** *Every consumer of an accepted* $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ *packet may read only the closed beta-strong false-inhabitation exclusion displayed by* T, L, B, F, R*. The packet does not export full closed consistency, closed subject reduction, unrestricted normalization, host evaluation, quotient conversion, or a route from arbitrary closed proofs to beta-strong normal inhabitants of the same false encoding.*

Proof. Start from a consumer read of the packet. The only contradiction coordinate is R, and its inputs are precisely the displayed rows T, L, B, F.

The rows H, C, P, N allow the same read to be transported, replayed, packaged, and named, but they do not add a subject-reduction discharge, a normalization theorem, or a consistency principle. Therefore the exported claim remains the closed beta-strong false-inhabitation exclusion and nothing stronger. □

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyMainUp.ClosedNormalConsistencyMainCarrier positive boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyMain}\uparrow}$.

Constructive story: A $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ packet is generated from HasType, ClosedAt, BetaStrong, false-encoding, normal-form contradiction-route, componentwise transport, displayed route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyMain}\uparrow}, \mathsf{seed})$

Scope closed: Definition 764.1, Definition 764.2, and Definition 764.4 bind the finite NameCert surface for the closed-normal consistency main-result boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyMain}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close full closed consistency, closed subject reduction, unrestricted normalization, host evaluation, quotient conversion, ambient consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, ledger exactness, positive-boundary non-escape, provenance, and local naming for the same finite main-result packet.

$\mathsf{MetaCICCriticalPath}^{\uparrow}$ reads its current open-phase source ledger through chapter 291, chapter 126, chapter 301, and chapter 38; the hub exports no quotient stream, selected limit, host real equality, ambient completeness, or bridge claim.

Chapter 765

A Concrete Naming Certificate for $\mathsf{MetaCICCriticalPath}^{\uparrow}$

The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet realizes the critical-path reading of chapter 9385 as a finite NameCert packet. It is narrower than the synthesis packet: it records the path from closed strong normalisation to the normal-form consistency row, keeps the subject-reduction obstruction visible, and names the substitution, confluence, decidability, discharge, transport, continuation, provenance, and local naming rows needed by downstream MetaCIC consumers.

It sits beside the subject-reduction obstruction packet of chapter 731, the normalization audit packet of chapter 750, the decidability boundary packet of chapter 749, and the normal false exclusion packet of chapter 732. Those siblings supply the named rows; the present packet records the finite route by which a reader may use them together without converting the blocked edge into a theorem.

Definition 765.1 (MetaCIC critical-path packet)**.** *A* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, N, O, U, D, H, C, P, L)$$

with the following visible rows:

1. *S is the closed-strong-normalization route, read through the candidate-mediated normalization surface of chapter 750;*
2. *N is the normal-form consistency row, read through the normal false exclusion packet of chapter 732;*
3. *O is the subject-reduction obstruction row, read through chapter 731;*
4. *U is the substitution, confluence, and decidability handoff row, naming the closed substitution support, conditional confluence route, and bounded checker boundary used before any consistency read is attempted;*
5. *D is the discharge-bundle socket, naming the missing subject-reduction input as a socket rather than as a supplied theorem;*
6. *H stores componentwise* hsame *transports for S, N, O, U, D, P, L;*
7. *C stores the displayed* Cont *routes from the strong-normalization row to the normal-form row, from the handoff row to the checker and confluence boundaries, and from the obstruction row to the discharge socket;*

8. *P is the* Pkg *provenance over the MetaCIC audit synthesis chapter and the four sibling concrete packets cited above;*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row.*

The carrier has no coordinate for closed subject reduction, no coordinate for full closed consistency, no global beta-conversion decision procedure, and no quotient over MetaCIC terms.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.TasteGate.MetaCICCriticalPathTasteGate single carrier alignment`

Theorem 765.2 (MetaCIC critical-path consistency handoff)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet hands a consistency-facing consumer from the closed-strong-normalization row S to the normal-form consistency row N only through the displayed route rows H, C, P, L. The consumer may read the normal false exclusion sibling after the normalization route has supplied the normal-form endpoint, but it cannot read a full closed-consistency theorem unless the discharge-bundle socket D is separately filled.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket consistency handoff`

Proof. Project the accepted carrier $K = (S, N, O, U, D, H, C, P, L)$. The rows S and N are the only positive rows on the consistency path: S records the closed-strong-normalization route, and N records the normal-form false-exclusion endpoint.

The route row C fixes the passage from S to N, while H, P, L preserve the same path under componentwise transport, provenance readback, and local naming. This gives the consumer the closed-normal consistency handoff named by the sibling packets.

The discharge socket D is a separate coordinate. Since it is recorded as a socket and not as a filled subject-reduction theorem, the handoff stops at the normal-form endpoint and does not export full closed consistency. □

Theorem 765.3 (MetaCIC critical-path obstruction boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet, the obstruction row O remains visible under the substitution, confluence, and decidability handoff row U and under all componentwise transports in H. Agreement on S, N, U, D, H, C, P, L cannot erase O, cannot convert D into a discharged bundle, and cannot promote the normal-form consistency route to closed subject reduction or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket obstruction boundary`

Proof. The carrier definition lists O as a named row. It is not derived from S, N, or U; it is part of the displayed packet and is read through the subject-reduction obstruction sibling.

The row U collects the supporting handoffs for substitution, confluence, and decidability. These handoffs explain how local syntax movement, conditional confluence, and bounded checking interact with the route, but none of them fills the subject-reduction discharge socket D.

Componentwise transport through H and route replay through C preserve the same rows. Therefore a transported packet still contains O and still treats D as a socket, so the excluded closed subject-reduction and full closed-consistency conclusions remain outside the packet. □

Theorem 765.4 (MetaCIC critical-path route replay coverage)**.** *Every route replay accepted by a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet factors through the displayed rows*

$$S, N, O, U, D, H, C, P, L.$$

The replay may use the closed-strong-normalization source S, the normal-form endpoint N, the handoff row U, componentwise transport H, continuation replay C, provenance P, and local naming L, but it keeps the obstruction row O and discharge socket D visible throughout the same packet.

Proof. Project the carrier from Definition 765.1. Its finite row list is exactly $S, N, O, U, D, H, C, P, L$, so an accepted replay begins inside that displayed packet rather than inside an ambient scheduler or host dependency graph.

Apply Definition 765.2. The consistency-facing part of the replay travels from S to N through the visible route rows and therefore cannot skip the packet surface that carries H, C, P, L.

It remains to keep the blocked side attached to the same replay. Use Definition 765.3, Definition 765.25, and Definition 765.34. These boundary results preserve O and D while stopping the read at N, so every accepted route replay factors through the full displayed row list. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRouteReplayCoverage`

Theorem 765.5 (MetaCIC critical-path consistency route totality)**.** *Every consistency-facing route accepted by a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier reads the whole tuple*

$$S, N, O, U, D, H, C, P, L.$$

In particular, no consistency-facing consumer may read only the normalization endpoint or only the handoff row while dropping the obstruction row, discharge socket, transport, continuation, provenance, or local naming rows of the displayed packet.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConsistencyRouteTotality`

Proof. Start from Definition 765.38. It identifies the accepted carrier with exactly the tuple $K = (S, N, O, U, D, H, C, P, L)$ and supplies no abbreviated consistency carrier.

The consistency handoff uses that full packet. Apply Definition 765.2 and Definition 765.33. The route from S to N is read through the displayed route, transport, provenance, and local naming rows while O and D remain visible.

Finally apply Definition 765.39 and Definition 765.40. Replay has no route outside $S, N, O, U, D, H, C, P, L$, and therefore every accepted consistency-facing route is total over the displayed tuple rather than a partial read of a hidden closed-consistency object. □

Theorem 765.6 (MetaCIC critical-path consumer refusal exactness)**.** *Any MetaCIC consumer that omits the obstruction row O or the discharge socket D is refused as a read of full consistency or subject reduction from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Such a consumer may still read the displayed normal-form endpoint N, but only as the bounded endpoint of the same finite route certificate.*

Proof. Read the packet rows from Definition 765.1. The rows O and D are not optional annotations: O records the subject-reduction obstruction, and D records the discharge-bundle socket.

Apply Definition 765.3. A consumer that drops O is no longer reading the accepted packet as a subject-reduction boundary, because the obstruction must remain visible under the handoff and transport rows.

Now apply Definition 765.25 and Definition 765.34. A consumer that drops D cannot present a full-consistency or subject-reduction read, since the normal-form endpoint remains bounded at N unless the separate socket is supplied. The refusal is therefore exact: missing O or missing D removes precisely the stronger read while preserving the bounded normal-form route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConsumerRefusalExactness`

Theorem 765.7 (MetaCIC critical-path certificate row order)**.** *In every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *certificate read, the source row S and the normal-form row N are read only after the obstruction row O and discharge socket D remain visible through H, C, P, L. Thus the certificate order never lets the positive normalization-to-normal-form route hide the blocked subject-reduction side.*

Proof. Start with the carrier definition. The accepted packet exposes $S, N, O, U, D, H, C, P, L$ as one finite certificate, so the source and endpoint rows are not isolated from the obstruction and socket rows.

Use Definition 765.2 to read the positive route. It sends consistency-facing consumers from S to N only through the displayed route rows, and those rows are part of the same certificate surface.

Finally combine Definition 765.3, Definition 765.25, and Definition 765.34. They keep O and D visible through transport, continuation, provenance, and local naming while stopping readback at N. Therefore S and N are read only in the row order that preserves the obstruction and socket throughout H, C, P, L. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCertificateRowOrder`

Theorem 765.8 (MetaCIC critical-path discharge-socket totality)**.** *Every accepted packet read of the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *discharge socket exposes the tuple rows*

$$S, N, O, U, D, H, C, P, L.$$

The read may use D as the named discharge socket, but it cannot treat D as closed subject reduction, full closed consistency, or a hidden theorem-export row outside the displayed packet.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket discharge socket totality`

Proof. Read the carrier from Definition 765.1. The packet definition makes $S, N, O, U, D, H, C, P, L$ the complete finite row inventory, with D named as a socket rather than as a filled subject-reduction theorem.

Apply the consistency handoff and obstruction boundary. Theorems Definition 765.2 and Definition 765.3 keep the positive S-to-N read and the blocked obstruction row in the same packet while refusing any conversion of D into a closed subject-reduction row.

Finally use discharge-socket nonescape and root consumer readiness from Definition 765.25 and Definition 765.86. These two surfaces force every downstream socket read through the displayed tuple and leave no consumer-facing coordinate from which closed subject reduction or full closed consistency can be read. □

Theorem 765.9 (MetaCIC critical-path downstream discharge visibility)**.** *Every downstream readiness read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that also replays the discharge socket must keep the downstream NameCert surface and the visible discharge row in the same packet provenance. The consumer sees readiness through C, L and the downstream row, while the socket replay remains bounded by U, O, D rather than becoming closed subject reduction.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket downstream discharge visibility`

Proof. Apply Definition 765.11 to obtain the downstream NameCert surface from the packet route through C, L and the downstream row. Then apply Definition 765.44 to the socket replay through U, O, D. The Lean theorem packages these two reads together with the packet provenance, so the downstream consumer cannot separate readiness from the visible discharge boundary. □

Theorem 765.10 (MetaCIC critical-path normalization-obstruction source order)**.** *Any normal-form consistency read of an accepted* MetaCICCriticalPath$^{\uparrow}$ *packet consumes the closed-normalization route S and the normal-form row N only in the source order that preserves the obstruction row O and the discharge socket D. Thus the normal-form endpoint remains a bounded route endpoint rather than a discharge of the blocked subject-reduction side.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationObstructionSourceOrder`

Proof. Start with normalization-frontier factorization. By Definition 765.58, a normal-form consistency read factors through the closed-normalization source row S and the normal-form endpoint row N while keeping the obstruction side visible.

Next use the handoff rows that prepare the read. The bounded checker handoff of Definition 765.59 and the substitution handoff visibility of Definition 765.60 keep U, H, C, P, L attached to the same packet instead of replacing O or D by an ambient checker fact.

Finally apply Definition 765.62. The normalization-to-normal-form route carries the obstruction row through the read and still stops at the separate discharge socket, so the source order consumes S and N without closing the blocked subject-reduction obligation. □

Theorem 765.11 (MetaCIC critical-path downstream readiness boundary)**.** *Downstream MetaCIC consumers receive only the bounded readiness surface of an accepted* MetaCICCriticalPath$^{\uparrow}$ *root-unblock packet. The exported surface is the displayed packet interface for carrier, route, obstruction, discharge socket, transport, continuation, provenance, local naming, and Lean intake; it is not full closed consistency and does not include closed subject reduction.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket downstream readiness boundary`

Proof. Begin with the downstream surface. The downstream-unblock coverage theorem Definition 765.80, the Lean target intake theorem Definition 765.81, and the downstream unblock surface theorem Definition 765.82 make the consumer-facing interface finite and bounded by the displayed packet rows.

Now factor that interface through the audit frontier. Apply Definition 765.83 together with carrier readiness and route readiness from Definition 765.84 and Definition 765.85. These results account for the carrier rows, route rows, obstruction row, discharge socket, transport, continuation, provenance, and local naming without adding a closed theorem row.

Finally use Definition 765.86. Every downstream MetaCIC consumer is restricted to the same bounded readiness surface, so the root-unblock packet opens consumer scheduling while refusing full closed consistency and closed subject reduction. □

Theorem 765.12 (MetaCIC critical-path local-diamond discharge cover)**.** *Every accepted local-diamond discharge read of a* MetaCICCriticalPath$^{\uparrow}$ *packet must display the candidate residual route, the normalization-obstruction source order, and the downstream readiness boundary before it can be consumed by confluence or strong-normalization frontier packets. The accepted read is a finite packet route through*

$$S, N, O, U, D, H, C, P, L$$

together with the visible candidate residual budget, the obstruction-preserving source order, and the downstream consumer boundary. It exports no hidden global confluence theorem, no unbounded normalization theorem, and no host reduction equality.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathLocalDiamondDischargeCover`

Proof. First expose the socket route. The discharge-socket totality theorem Definition 765.8 forces every socket read through the displayed rows $S, N, O, U, D, H, C, P, L$, so a local-diamond discharge cannot start from an external confluence or normalization object.

Next retain the visible discharge and candidate residual surface. The downstream discharge visibility theorem Definition 765.9 keeps the downstream NameCert surface and the visible discharge row in the same packet provenance. The candidate residual route is therefore consumed as packet-local discharge data, not as a detached proof of confluence.

Now order the normalization side. By Definition 765.10, the read of S and N preserves the obstruction row O and the discharge socket D before any endpoint is consumed. This supplies the required normalization-obstruction source order for the local-diamond handoff.

Finally apply the downstream readiness boundary, Definition 765.11. Confluence and strong-normalization frontier packets receive only the bounded readiness surface assembled above. Since the assembled route has no coordinate for global confluence, unbounded normalization, or host reduction equality, those endpoints remain outside the discharge cover. □

Theorem 765.13 (MetaCIC critical-path root socket totality)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every consistency-facing read first exposes the discharge socket D together with the displayed rows S, N, O, U, H, C, P, L. Thus the packet presents the socket as part of the root surface before the read may pass from the closed-strong-normalization row S to the normal-form row N.

Proof. Project the carrier through Definition 765.1. The row list of K includes the discharge socket D beside the closed-normalization source, normal-form endpoint, obstruction, handoff, transport, continuation, provenance, and local naming rows.

Use the consistency handoff of Definition 765.2. A consistency-facing consumer can travel from S to N only through the displayed route rows, and the theorem keeps D separate from a filled subject-reduction input.

Finally apply the obstruction boundary, Definition 765.3. It prevents the handoff row or componentwise transport from erasing O or converting D into a discharged bundle. Hence every accepted root read exposes D together with S, N, O, U, H, C, P, L before the consistency-facing passage is accepted. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootSocketTotality`

Theorem 765.14 (MetaCIC critical-path handoff exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet $K = (S, N, O, U, D, H, C, P, L)$, the finite handoff route is exhausted by the closed-normalization row S, normal-form row N, obstruction row O, handoff row U, discharge socket D, transport row H, route replay row C, provenance row P, and local naming row L. It has no coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or quotient term equality.*

Proof. Start with the obligation surface. By Definition 765.20, the local certificate obligations are exactly carrier admission, visibility of S, N, O, U, D, componentwise transport through H, displayed route replay through C, and provenance and local naming through P, L.

Next stabilize the visible rows. The transport row stability theorem Definition 765.22 keeps S, N, O, U, D, C, P, L under H, while provenance nonescape and local naming exhaustion from Definition 765.23 and Definition 765.24 keep P and L from naming an outside theorem coordinate.

It remains to handle the blocked side. Apply discharge-socket nonescape, Definition 765.25. The socket D can be preserved and routed, but it cannot be filled by S, N, O, U, H, C, P, L. Therefore the finite handoff is exhausted by the nine displayed rows and contains no coordinate for the four excluded metatheoretic claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathHandoffExhaustion`

Theorem 765.15 (MetaCIC critical-path downstream unblock readiness)**.** *A downstream MetaCIC consumer may cite an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet only as a finite socket-aware route from closed strong normalization to normal-form consistency. The citation may use* $S, N, O, U, D, H, C, P, L$ *and the displayed socket route, but it may not treat the packet as a discharged subject-reduction theorem or as a full closed-consistency theorem.*

Proof. First expose the root socket. By Definition 765.13, every consistency-facing read displays D with the rest of K before the route from S to N is accepted.

Next exhaust the handoff route. Theorem 765.14 says that the finite route has only the nine displayed coordinates and no coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or quotient term equality.

Finally combine discharge-socket coverage with socket-route totality. Theorems 765.26 and 765.28 force any downstream socket read to remain inside $S, N, O, U, D, H, C, P, L$ unless a separate subject-reduction input is displayed. Hence a downstream consumer may cite the packet as a socket-aware normalization-to-normal-form route, and not as either excluded discharged theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDownstreamUnblockReadiness`

Theorem 765.16 (MetaCIC critical-path subject-reduction socket budget)**.** *Every closed-consistency consumer of a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must display the subject-reduction socket row* D *before it attempts to read full closed consistency. The visible route may use the closed-strong-normalization row* S*, normal-form consistency row* N*, obstruction row* O*, handoff row* U*, and structural rows* H, C, P, L*, but the transition from normal-form consistency to full closed consistency remains blocked until* D *is filled by an external subject-reduction discharge.*

Proof. Project the carrier $K = (S, N, O, U, D, H, C, P, L)$ from Definition 765.1. The row D is listed as the discharge-bundle socket, not as a supplied subject-reduction theorem.

Apply Definition 765.2. The consistency-facing route can pass from S to the normal-form row N, but it stops before full closed consistency unless the socket is separately filled.

Apply Definition 765.3, Definition 765.13, and Definition 765.25. The obstruction row O remains visible under U, H, C, P, L, so no consumer can erase the socket or reinterpret it as a discharged subject-reduction row. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubjectReductionSocketBudget`

Theorem 765.17 (MetaCIC critical-path L10 discharge budget)**.** *The subject-reduction socket* D *of a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet is tied to the L10 discharge budget through the four named completion faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. A consumer that claims the L10 exit route must expose those four object rows together with* D, H, C, P, L *before reading the exit certificate; it cannot use the exit route to bypass the subject-reduction socket or to read full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10DischargeBudget`

Proof. Start with the L10 exit certificate already recorded for the chapter. The certificate names the four-object route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the completion-facing discharge budget.

Apply Definition 765.128 and Definition 765.129. These results expose the four object rows as a bounded exit dependency rather than as a proof of closed subject reduction.

Finally combine the exit route with Definition 765.16. The same structural rows H, C, P, L replay and name D next to the four L10 faces, so the consumer cannot use the exit certificate to bypass the socket or promote normal-form consistency to full closed consistency. □

Theorem 765.18 (MetaCIC critical-path discharge socket case induction)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet whose discharge socket is inspected by candidate-mediated strong-normalization, confluence, decidability, and L10 completion consumers has exactly the visible cases named by the socket budget rows: filled local discharge, open subject-reduction obstruction, bounded checker handoff, or L10 residual deferral. The induction is over the finite carrier rows of the packet and exports no source-equivalence transport, hidden subject-reduction theorem, or full closed-consistency route.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketCaseInduction`

Proof. Project the carrier $K = (S, N, O, U, D, H, C, P, L)$ from Definition 765.1. The discharge socket D is the only row through which a consumer can ask whether the subject-reduction gap is filled, obstructed, handed to a bounded checker, or left for the L10 corridor.

Use Definition 765.16. It forces every closed-consistency read to display D beside the normalization, normal-form, obstruction, handoff, and structural rows, so no additional hidden socket case can be introduced by a consumer.

Use Definition 765.17. It separates the L10 residual case from the filled, obstructed, and bounded-checker cases by requiring the four completion faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the exit route can be read.

Finite row exhaustion now gives the stated induction. The visible rows are local discharge, subject-reduction obstruction, bounded checker handoff, and L10 residual deferral; because the proof only projects those rows, it does not use source-equivalence transport or promote the socket to a closed subject-reduction theorem. □

Theorem 765.19 (MetaCIC critical-path typed-example frontier route)**.** *Every TypedExamples gallery row that enters a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must enter through the displayed discharge socket D, bounded checker handoff, and obstruction rows before it can be read by a consistency-facing consumer. The frontier route may exhibit typed examples as bounded evidence for the socket budget, but it cannot be read as subject reduction, confluence, strong normalization, global conversion decidability, source-equivalence transport, or full closed consistency.*

Proof. Use Definition 765.16. It places every closed-consistency attempt behind the subject-reduction socket D, so a typed-example row can only supply evidence at that socket boundary.

Next apply Definition 765.17. The L10 completion-facing exit route keeps the four completion faces and the socket rows visible, preventing a gallery example from bypassing the bounded checker or the obstruction ledger.

Finally use Definition 765.18. The visible cases are filled local discharge, open subject-reduction obstruction, bounded checker handoff, and L10 residual deferral. TypedExamples rows therefore remain frontier evidence for those cases and do not promote the packet to subject reduction, confluence, strong normalization, global conversion decidability, or full closed consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathTypedExampleFrontierRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet is generated from closed-strong-normalization, normal-form consistency, subject-reduction obstruction, substitution-confluence-decidability handoff, discharge-bundle socket, transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 765.1, Definition 765.2, Definition 765.3, Definition 765.128, Definition 765.129, Definition 765.132, Definition 765.134, Definition 765.72, Definition 765.74, Definition 765.348, Definition 765.349, and Definition 765.350 bind the MetaCIC critical path, its mature L10 source package over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, the candidate frontier, the socket-separation witnesses, the public tail bridge, and the residual route surface present in the spine and child inputs.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove closed subject reduction, full closed consistency, arbitrary closed Church–Rosser, global beta-conversion decidability, checked Lean certificate coverage, or a public standard bridge.

Upgrade path: Further progress is formal-axis verification or a checked bridge for the same MetaCIC critical-path route.

Theorem 765.20 (MetaCIC critical-path NameCert obligation surface)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *obligation surface consists exactly of the following nine row obligations:*

1. *carrier admission for the displayed finite packet* K*;*
2. *closed-strong-normalization route visibility through* S*;*
3. *normal-form consistency row visibility through* N*;*
4. *subject-reduction obstruction visibility through* O*;*
5. *substitution, confluence, and decidability handoff visibility through* U*;*
6. *discharge-bundle socket visibility through* D*;*
7. *componentwise* hsame *transport through* H*;*
8. *displayed* Cont *route replay through* C*;*
9. *provenance and local naming through* P *and* L*.*

It exports no closed subject-reduction theorem and no full closed-consistency theorem.

Proof. Project the accepted carrier through Definition 765.1. The definition displays the whole finite packet as

$$K = (S, N, O, U, D, H, C, P, L)$$

and assigns a distinct local role to each coordinate: S is the closed-strong-normalization route, N is the normal-form consistency row, O is the subject-reduction obstruction row, U is the substitution, confluence, and decidability handoff row, D is the discharge-bundle socket, H is componentwise transport, C is displayed route replay, P is provenance, and L is the local naming row. These rows give the nine listed obligations.

The consistency-facing part is bounded by Definition 765.2. It permits a consumer to pass from S to N only through the displayed route rows H, C, P, L, and it leaves D as a separate socket. Therefore the normal-form consistency read is not promoted to full closed consistency.

The subject-reduction side is bounded by Definition 765.3. The obstruction row O remains visible under U and under componentwise transport in H; agreement on the other rows cannot erase O, fill D, or convert the packet into closed subject reduction.

It remains only to check exhaustion. The carrier definition supplies no tenth coordinate for closed subject reduction, no coordinate for full closed consistency, no global beta-conversion decision procedure, and no quotient over MetaCIC terms. Hence every local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ obligation is one of the displayed row obligations above, and the two excluded theorem exports remain outside the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNameCertObligationSurface`

Theorem 765.21 (MetaCIC typed candidate-frontier bridge)**.** *Every typed candidate-frontier read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$ *must expose the closed-strong-normalization row* S*, conditional confluence and bounded decidability handoff row* U*, subject-reduction obstruction row* O*, discharge socket* D*, transport row* H*, replay row* C*, provenance row* P*, and local* $\mathsf{NameCert}$ *row* L *before any consistency-facing consumer may use the normal-form row* N*. The bridge records those rows as a frontier surface and does not claim closed subject reduction.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathTypedCandidateFrontierBridge`

Proof. Use the consistency handoff. Definition 765.2 permits a consumer to move from S to N only through the displayed route rows and keeps the discharge socket D separate.

Use the obstruction boundary next. Definition 765.3 keeps O visible under the handoff row U and prevents D from becoming a filled subject-reduction theorem.

Finally apply the NameCert obligation surface. Definition 765.20 lists exactly the carrier, $S, N, O, U, D, H, C, P, L$ obligations, so a typed candidate-frontier read must display the stated rows before a consistency-facing use of N. □

Theorem 765.22 (MetaCIC critical-path transport row stability)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

componentwise transport through H *preserves the visible rows* S, N, O, U, D, C, P, L*. In particular, transport cannot erase the obstruction row* O*, cannot identify it with the discharge socket* D*, and cannot turn the displayed route from* S *to* N *into either closed subject reduction or full closed consistency.*

Proof. Project the accepted carrier through Definition 765.1. The row H is declared as componentwise hsame transport for S, N, O, U, D, P, L, while C records only the displayed continuation routes among the same finite rows.

The obstruction boundary in Definition 765.3 fixes the critical point: O remains visible under the handoff row U and under all transports in H. Hence a transported packet still has an obstruction row and still has a separate discharge socket D.

The consistency handoff in Definition 765.2 and the obligation surface in Definition 765.20 keep the route from S to N inside H, C, P, L. Therefore transport stabilizes the displayed rows without exporting either excluded theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket transport row stability`

Theorem 765.23 (MetaCIC critical-path provenance nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the provenance row* P *records provenance only over the MetaCIC audit synthesis chapter and the cited sibling packets for subject-reduction obstruction, normalization audit, decidability boundary, and normal false exclusion. It exports neither closed subject reduction nor full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket provenance nonescape`

Proof. Project the row P from Definition 765.1. Its declared source is the Pkg provenance over the MetaCIC audit synthesis chapter and the four sibling concrete packets named in the chapter opening.

The consistency-facing handoff of Definition 765.2 uses this provenance only to read the path from S to the normal-form row N. The socket D remains separate, so provenance does not close the gap to full closed consistency.

The obstruction boundary of Definition 765.3 keeps O visible and prevents the handoff row from filling D. Together with the exact obligation list in Definition 765.20, this shows that P records the cited packet origins and has no escape route to the two excluded theorem exports. □

Theorem 765.24 (MetaCIC critical-path local naming exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row* L *names exactly the finite packet coordinates* $S, N, O, U, D, H, C, P, L$*. It adds no hidden theorem coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or a quotient over MetaCIC terms.*

Proof. Project the local naming row from Definition 765.1. The displayed carrier has exactly the tuple coordinates $S, N, O, U, D, H, C, P, L$, and L is declared as the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ row for that packet.

The obligation surface in Definition 765.20 exhausts the local surface by the nine row obligations. Its final exclusion rules out a tenth coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or a quotient over MetaCIC terms.

The two boundary theorems preserve the same exhaustion. Definition 765.2 keeps the consistency read at the normal-form endpoint, and Definition 765.3 keeps the obstruction row and discharge socket distinct. Thus L names exactly the displayed finite coordinates and no hidden theorem coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket local naming exhaustion`

Theorem 765.25 (MetaCIC critical-path discharge socket nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, no downstream consumer can fill the discharge socket* D *unless it displays a separate certificate row for the missing subject-reduction input. The rows* S, N, O, U, H, C, P, L *may route, transport, name, and preserve the socket, but they do not discharge it.*

Proof. Project the accepted carrier through Definition 765.1. The definition names D as the discharge-bundle socket and not as a subject-reduction theorem.

Apply the consistency handoff and obstruction boundary. Definition 765.2 lets a consumer move from S to N only through the displayed route rows and stops unless D is separately filled. Definition 765.3 keeps the obstruction row O visible and refuses the conversion of D into a discharged bundle.

The remaining rows preserve this separation. Definition 765.20, Definition 765.22, Definition 765.23, and Definition 765.24 exhaust the carrier, transport, provenance, and naming surfaces without adding a separate subject-reduction certificate row. Hence no consumer can fill D from the packet alone. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketNonescape`

Theorem 765.26 (MetaCIC critical-path discharge-socket coverage)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet* $K = (S, N, O, U, D, H, C, P, L)$*, any consistency-facing consumer must either display a separate subject-reduction input filling the socket row* D *or remain bounded by the rows* $S, N, O, U, D, H, C, P, L$*. The packet itself exposes the socket as a named obligation, not as a closed subject-reduction theorem.*

Proof. Project the carrier through Definition 765.1. The packet displays the source row S, normal-form row N, obstruction row O, handoff row U, discharge socket D, transport row H, continuation row C, provenance row P, and local naming row L.

Use the consistency handoff and obstruction boundary, Definition 765.2 and Definition 765.3. The handoff reaches the normal-form endpoint through the displayed route, while the obstruction boundary keeps the subject-reduction gap visible and keeps D separate from a filled discharge.

Finally apply the obligation surface, transport row stability, local naming exhaustion, and discharge-socket nonescape from Definition 765.20, Definition 765.22, Definition 765.24, and Definition 765.25. These rows cover every accepted consumer read, so a consumer that does not present a separate subject-reduction input remains bounded by the displayed root-unblock packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket discharge socket coverage`

Theorem 765.27 (MetaCIC critical-path subject-reduction socket separation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the rows* S, N, O, U, H, C, P, L *preserve the displayed route and keep the obstruction row* O *visible, but they cannot identify* O *with the discharge socket* D*. They also cannot promote the normal-form consistency row* N *to closed subject reduction without a separate subject-reduction certificate row that fills* D*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubjectReductionSocketSeparation`

Proof. Project the accepted packet through Definition 765.1. The carrier lists O as the subject-reduction obstruction row and D as the discharge-bundle socket, so the two rows are displayed as separate coordinates of the finite packet.

Use the consistency handoff and obstruction boundary. Theorems 765.2 and 765.3 preserve the route from S to N while keeping the obstruction visible under the handoff row U and transport row H.

Next apply discharge-socket nonescape and socket-route totality. Theorems 765.25 and 765.28 force every socket-facing read to keep D as a socket row unless a separate subject-reduction certificate is displayed.

Finally use discharge-socket coverage. Theorem 765.26 says that a consumer without the separate certificate remains bounded by $S, N, O, U, D, H, C, P, L$. Hence preserving the route and obstruction never identifies O with D, and the normal-form endpoint cannot be read as closed subject reduction. □

Theorem 765.28 (MetaCIC critical-path socket route totality)**.** *Any accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer route that mentions the discharge socket* D *must also display the strong-normalization*

row S, normal-form row N, obstruction row O, handoff row U, transport row H, continuation row C, provenance row P, and local name row L. The socket is therefore a visible route obligation, not a hidden closed subject-reduction theorem.

Proof. Use the consistency handoff and obstruction boundary, Definition 765.2 and Definition 765.3. They route the read from S to N while keeping O visible and keeping D separate from any closed discharge.

Now apply discharge-socket nonescape and root route totality, Definition 765.25 and Definition 765.42. Together they require the socket-facing consumer to retain $S, N, O, U, D, H, C, P, L$ as the displayed row inventory, so no hidden closed subject-reduction theorem is imported. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket socket route totality`

Theorem 765.29 (MetaCIC critical-path subject-reduction socket factorization)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier $K = (S, N, O, U, D, H, C, P, L)$, any route attempting to use the discharge socket D must factor through the displayed rows $S, N, O, U, D, H, C, P, L$. Along that route the obstruction row O remains visible, and D remains a socket unless a separate subject-reduction certificate row is displayed by the consumer.*

Proof. Start with a socket-facing route. The consistency handoff and obstruction boundary, Definition 765.2 and Definition 765.3, route any consistency read through the visible packet while keeping O present.

Now use discharge-socket nonescape, Definition 765.25. It prevents the displayed packet rows from filling D by themselves, so the only admissible way to discharge the socket is to show a separate subject-reduction certificate row.

Finally apply socket route totality, Definition 765.28. Any route that mentions D must retain $S, N, O, U, D, H, C, P, L$ as its visible row inventory. Hence the attempted discharge factors through the displayed packet and cannot hide O or turn D into a closed subject-reduction theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubjectReductionSocketFactorization`

Theorem 765.30 (MetaCIC critical-path obstruction-ledger exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet $K = (S, N, O, U, D, H, C, P, L)$, the visible obstruction row O and socket row D exhaust the blocked subject-reduction routes. No hidden closed-consistency coordinate can be read from S, N, U, H, C, P, L once the obstruction and socket rows remain visible.*

Proof. Read O and D from the carrier definition. The former records the subject-reduction obstruction, and the latter records the discharge-bundle socket; neither is derived from the normal-form row or handoff row.

The consistency handoff and obstruction boundary fix the blocked route. By Definition 765.2, the positive route reaches only N unless a separate discharge is supplied. By Definition 765.3, O cannot be erased and D cannot be converted into a supplied theorem by agreement on the other rows.

Use the obligation surface, transport row stability, local naming exhaustion, and discharge-socket nonescape to close the row inventory. Theorems Definition 765.20, Definition 765.22, Definition 765.24, and Definition 765.25 leave no carrier coordinate through which a hidden closed-consistency read can bypass O and D. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket obstruction ledger exhaustion`

Theorem 765.31 (MetaCIC critical-path substitution-confluence socket ledger)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any consumer that combines closed substitution, conditional confluence, and bounded decidability must read the handoff row* U *together with the obstruction row* O *and discharge socket* D *before it may use the normal-form row* N*. The route cannot promote* U *into closed subject reduction, global Church–Rosser, or full consistency.*

Proof. Start with the consistency handoff, Definition 765.2. It lets a consumer reach the normal-form row N only along the displayed packet route and leaves the subject-reduction discharge socket separate.

Next attach the obstruction and socket rows. Theorems 765.3 and 765.25 keep O visible and keep D as a socket rather than a supplied closed subject-reduction theorem.

Finally use Definition 765.30. Since the obstruction ledger is exhausted by the visible O, D boundary, a substitution, confluence, and decidability consumer can use U only as a bounded handoff row before N is read. Thus the socket ledger has no route to closed subject reduction, global Church–Rosser, or full consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket substitution confluence socket ledger`

Theorem 765.32 (MetaCIC critical-path normal-form route coverage)**.** *Every normal-form readback route accepted by a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet factors through* $S \to N$ *with the obstruction row* O*, discharge socket* D*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* L *still visible. Such a route cannot drop the blocked subject-reduction rows while preserving the accepted packet.*

Proof. Start with the carrier and consistency handoff. The carrier names S and N as distinct rows, and Definition 765.2 makes the displayed continuation from S to N the only consistency-facing handoff.

Next keep the obstruction data visible. The obstruction boundary and discharge-socket nonescape, Definition 765.3 and Definition 765.25, preserve O and D as the blocked subject-reduction rows along that same route.

Finally apply the obligation surface, transport row stability, and local naming exhaustion. Theorems Definition 765.20, Definition 765.22, and Definition 765.24 keep H, C, P, L attached to the same finite packet, so every normal-form readback route factors through $S \to N$ with the obstruction, socket, transport, continuation, provenance, and naming rows still displayed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normal form route coverage`

Theorem 765.33 (MetaCIC critical-path handoff row totality)**.** *Every consistency-facing read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must pass through the complete displayed row list*

$$S, N, O, U, D, H, C, P, L.$$

In particular, a read that mentions the handoff row U *without the strong-normalization row* S*, normal-form row* N*, obstruction row* O*, discharge socket* D*, transport row* H*, route row* C*, provenance row* P*, and local naming row* L *is not a read of the accepted critical-path packet.*

Proof. Start from Definition 765.1. It displays the accepted carrier as $K = (S, N, O, U, D, H, C, P, L)$, with no abbreviated carrier for consistency-facing consumers.

The existing handoff theorems use the whole packet surface. Definition 765.2 reads the path from S to N through route, transport, provenance, and naming rows while leaving D separate. Definition 765.3 and Definition 765.25 keep O and D visible for every such read.

Finally apply the obligation, transport, provenance, and naming exhaustion results. Definition 765.20, Definition 765.22, Definition 765.23, and Definition 765.24 make $S, N, O, U, D, H, C, P, L$ the total row inventory. A consistency-facing read therefore cannot drop one of these rows and remain a read of the accepted packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket handoff row totality`

Theorem 765.34 (MetaCIC critical-path normal-form readback boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the normal-form readback boundary stops at the row* N*. It may expose the normal-form endpoint supplied after* S *and the visible obstruction and socket rows* O, D*, but it does not export closed subject reduction, full closed consistency, global beta-conversion decidability, or a quotient over MetaCIC terms.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normal form readback boundary`

Proof. Project the normal-form row from Definition 765.1. The row N is declared as the normal-form consistency row, while the subject-reduction obstruction and discharge socket remain separate rows O and D.

Apply Definition 765.2. It routes a consistency-facing consumer from S to N and explicitly stops before full closed consistency unless the discharge socket is separately filled.

The boundary is stable under the rest of the packet. Definition 765.3, Definition 765.22, Definition 765.23, Definition 765.24, Definition 765.25, and Definition 765.33 keep the obstruction, socket, row inventory, provenance, and local naming visible. Thus the readback reaches N and no stronger theorem export. □

Theorem 765.35 (MetaCIC critical-path normalization source visibility)**.** *Every consistency-facing read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *must expose the closed-strong-normalization source row* S *before it reads the normal-form row* N*. A read that starts at* N *without* S *is not a read of the accepted root-unblock packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normalization source visibility`

Proof. Project the carrier through Definition 765.1. The packet displays S as the closed-strong-normalization route and N as the normal-form consistency row, with no abbreviated carrier containing only N.

Apply Definition 765.2 and Definition 765.34. The handoff routes consistency-facing consumers from S to N, while the readback boundary stops the exported claim at N.

The obstruction and socket rows stay attached by Definition 765.3 and Definition 765.25. Hence the read of N is visible only after S has been displayed inside the same packet. □

Theorem 765.36 (MetaCIC critical-path decidability handoff visibility)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the row* U *is only the visible substitution, confluence, and decidability handoff row. It is not a beta-conversion oracle, a quotient over MetaCIC terms, or a hidden closed-consistency decision procedure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket decidability handoff visibility`

Proof. Read U from Definition 765.1. Its role is a handoff row inside the finite carrier, while the same carrier separately displays the source row S, normal-form row N, obstruction row O, and discharge socket D.

Use Definition 765.2, Definition 765.3, and Definition 765.34. These results route consistency reads through S, N, keep the obstruction visible, and prevent the normal-form endpoint from becoming a stronger theorem export.

Finally apply Definition 765.25. Since D remains a separate socket, U cannot be read as an oracle that fills the blocked subject-reduction or beta-conversion obligations. □

Theorem 765.37 (MetaCIC critical-path subject-reduction socket audit boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the subject-reduction socket* D *remains an audit boundary until a separate discharge row is displayed. The packet may route to* D*, but it does not identify* D *with closed subject reduction.*

Proof. Start with the carrier definition. The row D is displayed as a discharge-bundle socket, while O records the subject-reduction obstruction and U records only the visible handoff row.

Apply Definition 765.3 and Definition 765.25. The first keeps the obstruction row visible; the second prevents any packet consumer from treating the socket as already discharged.

The consistency handoff and normal-form boundary, Definition 765.2 and Definition 765.34, keep the root read at $S \to N$ unless an additional discharge row is supplied. Thus D is an audit boundary rather than a closed subject-reduction theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket subject reduction socket audit boundary`

Theorem 765.38 (MetaCIC critical-path carrier admission)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

carrier admission exposes exactly the nine finite BHist *rows displayed in Definition 765.1: the closed-strong-normalization route* S*, the normal-form consistency row* N*, the subject-reduction obstruction row* O*, the substitution, confluence, and decidability handoff row* U*, the discharge-bundle socket* D*, the componentwise transport row* H*, the continuation row* C*, the provenance row* P*, and the local naming row* L*.*

Proof. Project the accepted carrier through Definition 765.1. The carrier is introduced as the finite tuple $K = (S, N, O, U, D, H, C, P, L)$, and the definition assigns one BHist role to each coordinate.

The first five projections give the visible mathematical rows: S for closed strong normalization, N for the normal-form consistency endpoint, O for the subject-reduction obstruction, U for the substitution, confluence, and decidability handoff, and D for the discharge-bundle socket.

The remaining projections give the certificate infrastructure: H stores the componentwise transports, C stores the displayed continuations, P records provenance, and L records the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ row. Since the same definition explicitly excludes any coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or a quotient over MetaCIC terms, carrier admission has exactly the nine listed rows and no hidden row. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket carrier admission`

Theorem 765.39 (MetaCIC critical-path route replay)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the continuation coordinate* C *replays only the displayed route from* S *to* N*, the displayed handoff route from* U *to the checker and confluence boundaries, and the displayed route from* O *to the discharge socket* D*. It cannot synthesize closed subject reduction and cannot turn the normal-form consistency endpoint into full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket route replay`

Proof. Read the continuation coordinate from Definition 765.1. Its declared routes are the route from the strong-normalization row to the normal-form row, the route from the handoff row to the checker and confluence boundaries, and the route from the obstruction row to the discharge socket.

The consistency-facing route is already bounded by Definition 765.2. It lets a consumer pass from S to N through the displayed route rows, but it leaves D as a separate socket and therefore does not export full closed consistency.

The obstruction-facing route is bounded by Definition 765.3. The row O remains present under the handoff row U and under componentwise transport, so route replay cannot erase the obstruction or fill the discharge socket. Thus C replays only the displayed routes and supplies no closed subject-reduction theorem. □

Theorem 765.40 (MetaCIC critical-path route exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, every critical-path route replay is exhausted by the displayed rows* $S, N, O, U, D, H, C, P, L$*. The replay can expose the closed-strong-normalization row* S*, the normal-form consistency row* N*, the obstruction row* O*, the substitution-confluence-decidability handoff row* U*, and the discharge socket* D*, but it cannot use route replay to fill* D*, erase* O*, or turn* U *into closed subject reduction or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRouteExhaustion`

Proof. Project the accepted carrier through Definition 765.1. The packet has exactly the rows $S, N, O, U, D, H, C, P, L$, and Definition 765.20 lists those rows as the complete local obligation surface.

Now apply Definition 765.39. The continuation row C replays only the displayed route from S to N, the handoff route from U to the checker and confluence boundaries, and the obstruction route from O to D.

The boundary results rule out the stronger reads. By Definition 765.2, the route from S to N stops at the normal-form endpoint unless D is separately filled. By Definition 765.3, O remains visible and U does not discharge the socket. Hence route replay is exhausted by the displayed rows and cannot export the excluded theorem claims. □

Theorem 765.41 (MetaCIC critical-path root discharge exactness)**.** *For every root-facing* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer of an accepted packet* $K = (S, N, O, U, D, H, C, P, L)$*, the discharge socket* D *may be treated as exact only as the displayed finite socket row. The exactness read requires the rows* $S, N, O, U, D, H, C, P, L$ *to remain visible, and it imports no closed subject-reduction row into the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket root discharge exactness`

Proof. Project the consumer through Definition 765.1. The packet displays D as a discharge-bundle socket alongside the strong-normalization row, normal-form row, obstruction row, handoff row, transport row, continuation row, provenance row, and local naming row.

The consistency handoff and obstruction boundary keep the socket exact without closing it. Apply Definition 765.2 to stop the route at the normal-form row N, and apply Definition 765.3 to keep O visible while U remains only a handoff row.

Finally use discharge-socket nonescape and route exhaustion, Definition 765.25 and Definition 765.40. These force every discharge read through the displayed socket and exhaust every route inside $S, N, O, U, D, H, C, P, L$, so exactness never imports a closed subject-reduction row. □

Theorem 765.42 (MetaCIC critical-path root route totality)**.** *Every root route from the closed-strong-normalization row S to the normal-form readback row N is exhausted by the three displayed continuation legs in C: the route from S to N, the handoff route from U to the checker and confluence boundaries, and the obstruction route from O to the discharge socket D. The obstruction row O and the discharge socket D remain visible throughout the total route inventory.*

Proof. Start with route replay from Definition 765.39. It names exactly the three continuation legs carried by C, and no leg is a hidden closed subject-reduction or full closed-consistency route.

Route exhaustion and handoff row totality make the inventory total. The theorem Definition 765.40 keeps every replay inside $S, N, O, U, D, H, C, P, L$, while Definition 765.33 accounts for the displayed handoff row U without adding another route coordinate.

Carrier admission supplies the root packet on which totality is read. By Definition 765.38, the carrier has exactly the nine displayed rows, including O and D. Hence every root route from S to normal-form readback at N is one of the three continuation legs in C. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket root route totality`

Theorem 765.43 (MetaCIC critical-path root consumer readiness)**.** *Downstream consumers may read normal-form consistency readiness from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet only as the finite obligation surface $S, N, O, U, D, H, C, P, L$. They may not read the packet as full closed consistency, closed subject reduction, or an undisplayed theorem export beyond the root discharge and route obligations.*

Proof. Begin with the NameCert obligation surface of Definition 765.20. It gives the downstream consumer a finite packet surface and requires the local naming row to preserve the displayed rows.

Normal-form readback and local naming keep the readiness claim bounded. Use Definition 765.34 to stop readback at N, and use Definition 765.24 to prevent the local name from exporting a hidden theorem coordinate.

It remains to account for discharge and route obligations. Apply Definition 765.41 and Definition 765.42. The first keeps D as an exact visible socket rather than a closed subject-reduction proof, and the second exhausts the root routes through the displayed continuation legs. Therefore downstream readiness is exactly the finite packet obligation surface. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootConsumerReadiness`

Theorem 765.44 (MetaCIC critical-path discharge row visibility)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier $K = (S, N, O, U, D, H, C, P, L)$, the discharge socket row D remains visible under every critical-path route replay. The rows S, N, O, U, H, C, P, L may transport, route, name, and preserve the packet, but they cannot fill D or convert it into a closed subject-reduction or full closed-consistency theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket discharge row visibility`

Proof. Begin with carrier admission. By Definition 765.38, the row D is one of the nine displayed BHist rows of the accepted packet and is recorded specifically as the discharge-bundle socket.

Route replay preserves that row. Apply Definition 765.39 and Definition 765.40; the continuation row C replays only the displayed routes among $S, N, O, U, D, H, C, P, L$, and replay cannot fill D or erase the obstruction row O.

The boundary and naming rows rule out a disguised discharge. Use Definition 765.3, Definition 765.25, and Definition 765.24. They keep O and D distinct, require a separate subject-reduction input before D can be filled, and prevent the local name row from exporting a hidden theorem coordinate. □

Theorem 765.45 (MetaCIC critical-path local name nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the local name row* L *names only the displayed packet and its listed route obligations. It exports no closed subject-reduction theorem, no full closed-consistency theorem, no global beta-conversion decision procedure, and no quotient over MetaCIC terms.*

Proof. Read the local row from Definition 765.38 and Definition 765.24. The carrier exposes L as the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ row for the finite packet $S, N, O, U, D, H, C, P, L$, and local naming is exhausted by those coordinates.

The route inventory remains finite under replay. Use Definition 765.39 and Definition 765.40. They keep every replayed route inside the displayed tuple and provide no extra named coordinate for closed subject reduction or full closed consistency.

The remaining boundaries preserve the same nonescape claim. Apply Definition 765.20, Definition 765.3, and Definition 765.44. The obligation surface lists the local rows, the obstruction boundary keeps O and D visible, and discharge-row visibility prevents L from naming a filled theorem in place of the displayed socket. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket local name nonescape`

Definition 765.46 (MetaCIC critical-path frontier slice)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

the frontier slice is the finite BHist *packet*

$$F = (K, R, B, Q),$$

where K *is the accepted carrier of Definition 765.1,* $R = (S, N, O, U, D, H, C, P, L)$ *is the visible route inventory,* $B = D$ *is the blocked subject-reduction socket, and* Q *is the downstream consumer query that asks for either a consistency-facing read or a subject-reduction-facing read of the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierSliceConsumerCertificate`

Theorem 765.47 (MetaCIC critical-path frontier coverage)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$ *and every frontier slice* $F = (K, R, B, Q)$*, every consistency-facing or subject-reduction-facing consumer read factors through one of the displayed slice rows* K, R, B, Q*. In particular, the consumer can read the carrier, the route inventory* $R = (S, N, O, U, D, H, C, P, L)$*, the blocked socket* $B = D$*, and the downstream query* Q*, but it cannot read an additional theorem coordinate for closed subject reduction or full closed consistency.*

Proof. Project the slice through Definition 765.46. The row K is the accepted carrier of Definition 765.1, R is exactly the displayed route inventory, B is the same row as the discharge socket D, and Q records the consumer read.

For a consistency-facing read, apply Definition 765.2 and Definition 765.33. The handoff from S to N must pass through the complete displayed inventory and leaves D as a socket, so the read factors through K, R, B, Q.

For a subject-reduction-facing read, apply Definition 765.3, Definition 765.25, and Definition 765.24. These results keep the obstruction row visible, keep D undischarged, and exhaust the local name by the displayed carrier rows. Hence the frontier slice covers every accepted consumer read and supplies no hidden theorem coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierCoverage`

Theorem 765.48 (MetaCIC critical-path route-ledger totality)**.** *For every frontier slice $F = (K, R, B, Q)$ over an accepted carrier $K = (S, N, O, U, D, H, C, P, L)$, the route ledger is exhausted by the inventory*

$$R = (S, N, O, U, D, H, C, P, L).$$

No replay, handoff, or query row can add a closed-consistency coordinate outside this inventory.

Proof. Read the route inventory from Definition 765.46. By definition, R is not an abstract ledger; it is the concrete tuple $S, N, O, U, D, H, C, P, L$ attached to the accepted carrier.

Apply Definition 765.40. Every critical-path route replay is already exhausted by $S, N, O, U, D, H, C, P, L$, and route replay cannot fill D, erase O, or turn U into closed subject reduction or full closed consistency.

The slice adds only $B = D$ and the query row Q. By Definition 765.25, B remains a socket rather than a filled theorem. By Definition 765.47, Q only asks for a read through the displayed slice rows. Thus the ledger is total at R and cannot acquire a closed-consistency coordinate from replay, handoff, or query readback. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierSlice route ledger totality`

Theorem 765.49 (MetaCIC critical-path frontier nonescape)**.** *For every frontier slice $F = (K, R, B, Q)$ over an accepted carrier $K = (S, N, O, U, D, H, C, P, L)$, no transported route, replayed route, or downstream query can convert the visible obstruction row O and socket row $B = D$ into closed subject reduction or full closed consistency.*

Proof. Start from the carrier and slice definitions. The carrier definition exposes O as the subject-reduction obstruction row and D as the discharge-bundle socket, while the frontier slice records the same socket as $B = D$.

Transport preserves the obstruction boundary. Apply Definition 765.3 and Definition 765.24; transported agreement on the other rows cannot erase O, fill D, or add a hidden named theorem coordinate.

Replay and query readback remain inside the same inventory. Use Definition 765.39, Definition 765.40, and Definition 765.48. They confine replay to $S, N, O, U, D, H, C, P, L$ and make Q a read of the displayed slice rather than an escape from it. Therefore the frontier slice preserves the obstruction row and socket row exactly, with no route to closed subject reduction or full closed consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierSlice nonescape`

Theorem 765.50 (MetaCIC critical-path obstruction-socket separation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the obstruction row* O *and the discharge socket* D *remain distinct downstream obligations. Agreement on* S, N, U, H, C, P, L *cannot identify* O *with* D*, cannot erase* O*, cannot fill* D*, and cannot turn the handoff row* U *into closed subject reduction or full closed consistency.*

Proof. Read the two rows from Definition 765.1. The row O is the subject-reduction obstruction row, while D is the discharge-bundle socket naming the missing subject-reduction input.

Apply Definition 765.3. It keeps O visible under the handoff row U and under componentwise transport in H, and it explicitly refuses the conversion of D into a discharged bundle.

Finally use Definition 765.20 together with Definition 765.39. The obligation surface lists O and D separately, and the route replay only connects the obstruction row to the discharge socket as displayed data. Therefore downstream consumers inherit two separate obligations rather than a closed subject-reduction theorem or full closed-consistency theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathObstructionSocketSeparation`

Theorem 765.51 (MetaCIC critical-path handoff row coverage)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the handoff row* U *covers only the displayed substitution, confluence, and decidability handoffs used before the consistency read. Downstream consumers must still carry the obstruction row* O *and the discharge socket* D *as separate rows, and the coverage of* U *cannot fill* D*, erase* O*, or promote the route from* S *to* N *to closed subject reduction or full closed consistency.*

Proof. Project U from the carrier definition. Its declared role is the substitution, confluence, and decidability handoff row; it is not a closed subject-reduction row and not a full closed-consistency row.

Use Definition 765.2. The consistency-facing consumer may move from S to N only through the displayed route rows and cannot read a full closed-consistency theorem unless D is separately filled.

Use Definition 765.3 and Definition 765.39. The obstruction row O remains visible under U, while C replays the handoff and obstruction routes without changing their roles.

The obligation surface of Definition 765.20 then supplies the final coverage check: $S, N, O, U, D, H, C, P, L$ are the complete rows, with U only one row among them. Thus handoff coverage does not collapse the downstream obligations carried by O and D. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket handoff row coverage`

Theorem 765.52 (MetaCIC critical-path provenance naming)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the provenance row* P *and the local naming row* L *preserve exactly the public NameCert provenance cited by this chapter: the subject-reduction obstruction packet, the normalization audit packet, the decidability boundary packet, and the normal false exclusion packet, together with the local* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *name. They introduce no additional public provenance source.*

Proof. Project the provenance and naming coordinates in Definition 765.1. The row P is declared as the Pkg provenance over the MetaCIC audit synthesis chapter and the four sibling concrete packets cited in the chapter opening, while L is declared as the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ row.

The four public sibling packets are precisely the subject-reduction obstruction packet, the normalization audit packet, the decidability boundary packet, and the normal false exclusion packet. The carrier definition uses these rows to name the critical path and does not add another public packet coordinate.

The two boundary theorems preserve this reading. Definition 765.2 keeps the consistency route attached to the normalization and normal false exclusion rows, and Definition 765.3 keeps the obstruction route attached to the subject-reduction boundary. Consequently P and L preserve the cited public provenance and local name exactly, with no extra provenance source exported by the chapter. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket provenance naming`

Definition 765.53 (MetaCIC critical-path frontier-rank carrier)**.** *A frontier-rank read of* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *is a finite packet*

$$Q = (O, R, D, B, C, P, N)$$

whose rows are an open-node row O, a dependency-ready row R, a downstream-rank row D, an obstruction ledger B, a replay route C, package provenance P, and a local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row N. The open-node row names the still-visible frontier node, the dependency-ready row records which cited packet rows are already available, the downstream-rank row records the finite consumer rank, the obstruction ledger records the subject-reduction socket and blocked theorem exports, the replay route stores the* Cont *paths used by scheduler-facing consumers, and the provenance row ties the read to the public packet of Definition 765.1.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRankReplayClosure`

Theorem 765.54 (MetaCIC critical-path open-node exhaustion)**.** *Every accepted open-node read of* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *factors through a frontier-rank packet $Q = (O, R, D, B, C, P, N)$. The read may expose the node, its dependency-ready rows, its downstream rank, its obstruction ledger, replay route, provenance, and local name, but it has no additional coordinate for a hidden frontier node or an unranked scheduling obligation.*

Proof. Project the accepted frontier-rank carrier of Definition 765.53. The open-node coordinate is O, and every scheduler-facing dependency is recorded by one of R, D, B, C, P, N.

The underlying critical-path packet of Definition 765.1 already exhausts the mathematical rows used by this read. The frontier-rank carrier only records which of those rows are open, ready, ranked, obstructed, replayed, and named.

Inspect the excluded coordinates. Since Q contains no hidden frontier node and no unranked scheduling obligation, every accepted open-node read factors through the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRank open node exhaustion`

Theorem 765.55 (MetaCIC critical-path dependency-route nonescape)**.** *For every accepted frontier-rank packet $Q = (O, R, D, B, C, P, N)$, dependency routing is exhausted by the displayed rows R, B, C, P, N. It cannot appeal to hidden scheduler state, host graph reachability, or an unledgered dependency edge outside the obstruction ledger and replay route.*

Proof. Read the dependency-ready row R, obstruction ledger B, replay route C, provenance row P, and local naming row N from Definition 765.53. These are the only coordinates that can justify a dependency route.

The obstruction ledger records the blocked subject-reduction socket and theorem exports, while the replay route records the displayed Cont paths available to consumers. Provenance attaches those rows to the public critical-path packet rather than to a host-side graph.

No coordinate of Q stores hidden scheduler state, host graph reachability, or an unledgered dependency edge. Thus every dependency route remains inside the finite ledgered packet. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRank dependency route nonescape

Theorem 765.56 (MetaCIC critical-path Lean handoff surface)**.** *The paper frontier-rank packet $Q = (O, R, D, B, C, P, N)$ is the exact surface for a later Lean formalization of the* MetaCICCriticalPath$^{\uparrow}$ *root-unblock obligation. A Lean target should expose the open-node row, dependency-ready row, downstream-rank row, obstruction ledger, replay route, package provenance, and local NameCert row, and should not add scheduler state, host reachability, or unledgered dependency edges.*

Proof. The carrier definition gives the full handoff surface: O for the open node, R for dependency readiness, D for downstream rank, B for the obstruction ledger, C for replay, P for provenance, and N for local naming.

Open-node exhaustion shows that every accepted open-node read factors through these rows. Dependency-route nonescape shows that routing cannot leave these rows for hidden scheduler state, host graph reachability, or unledgered edges.

Therefore the Lean-facing obligation is coordinatewise exact. A formal target with precisely these rows verifies the same paper surface, while any extra scheduling coordinate would certify a different object. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathLeanHandoffSurface
↪ variant BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRank lean handoff surface

Theorem 765.57 (MetaCIC critical-path discharge-socket locality)**.** *For every accepted critical-path packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the discharge socket row D remains localized under the transport row H, replay route C, provenance row P, and local naming row L. A consumer may cite D as the displayed open socket for the consistency and subject-reduction frontier, but it cannot move D into a hidden global discharge, a closed subject-reduction theorem, or a full closed-consistency theorem.

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket discharge socket locality

Proof. Project the accepted packet $K = (S, N, O, U, D, H, C, P, L)$ from Definition 765.1. The row D is a visible packet coordinate, while H, C, P, L are the only rows that can transport, replay, provenance, and name that coordinate.

The consistency and obstruction boundaries keep the socket local. Definition 765.2 routes the consistency read through the displayed rows and stops at D, while Definition 765.3 keeps the obstruction row visible instead of converting D into a discharged bundle.

The remaining packet theorems preserve the same locality. Definition 765.20, Definition 765.22, and Definition 765.23 exhaust the certified rows and forbid a provenance escape. Hence every read of D stays under H, C, P, L and exports no closed theorem beyond the displayed socket. □

Theorem 765.58 (MetaCIC critical-path normalization-frontier factorization)**.** *Every normal-form consistency read of an accepted packet*

$$K = (S, N, O, U, D, H, C, P, L)$$

factors through three sibling rows: the normalization audit row $S \to N$, the normal-false exclusion recorded at the endpoint N, and the subject-reduction obstruction row O. The read cannot bypass these rows by treating D as a completed discharge or by exporting a closed consistency theorem.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normalization frontier factorization`

Proof. Start with the normal-form route in the accepted packet K. The consistency handoff of Definition 765.2 is exactly the route from the source row S to the normal-form row N, so every such read first factors through the normalization audit and its normal-false exclusion endpoint.

Next read the obstruction side. Definition 765.3 keeps O as the visible subject-reduction obstruction, and Definition 765.57 keeps D local to the displayed transport, replay, provenance, and naming rows.

Finally apply the obligation and provenance surfaces. Definition 765.20, Definition 765.22, and Definition 765.23 leave no row through which a normal-form read can skip the audit endpoint or the obstruction row. Thus the read factors through the stated three-row frontier. □

Theorem 765.59 (MetaCIC critical-path bounded-checker handoff totality)**.** *For every accepted packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the bounded-checker handoff row U exhausts the substitution, confluence, and decidability reads available to downstream consumers. It presents those reads as bounded handoff obligations and does not export closed subject reduction, full closed consistency, arbitrary closed Church–Rosser, or global beta-conversion decidability.

Proof. Project the row U from the accepted packet K. By Definition 765.20, U is one coordinate in the nine-row NameCert surface, not a replacement for the obstruction row O or the discharge socket D.

The boundary theorems determine what U can hand off. Definition 765.2 bounds the consistency read at the normal-form endpoint, and Definition 765.3 prevents the handoff from erasing the subject-reduction obstruction.

Transport and provenance keep the checker reads inside the same finite packet. Use Definition 765.22, Definition 765.23, Definition 765.57, and Definition 765.58. The only downstream checker surface left is the bounded handoff through U, so substitution, confluence, and decidability are exhausted there without exporting the closed theorem claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket bounded checker handoff totality`

Theorem 765.60 (MetaCIC critical-path substitution handoff visibility)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier $K = (S, N, O, U, D, H, C, P, L)$, the substitution handoff exposed by U is visible only as one displayed row of the finite packet of Definition 765.1. It may be replayed through C, transported through H, and named through P, L, but it cannot fill the discharge socket D, erase the obstruction row O, or export closed subject reduction.*

Proof. Project U from the carrier K in Definition 765.1. The definition assigns U the role of the substitution, confluence, and decidability handoff row, while D is a distinct discharge-bundle socket and O is a distinct subject-reduction obstruction row.

Apply Definition 765.51. It already fixes U as a coverage row for the displayed handoffs and refuses any collapse of the downstream obligations carried by O and D.

Now apply Definition 765.25 and Definition 765.50. These results keep D as a socket and keep O separate from that socket under route replay and packet readback.

Thus a substitution-facing consumer can read the handoff only through the displayed packet coordinates H, C, P, L. The same read never supplies the missing discharge input, so it cannot close subject reduction. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubstitutionHandoffVisibility`

Theorem 765.61 (MetaCIC critical-path confluence-decidability socket)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, every confluence-facing or decidability-facing use of the handoff row remains socket-mediated by the displayed rows* O, D, H, C, P, L*. Such a use may route through the finite packet, but it has no independent coordinate for arbitrary Church–Rosser, global beta-conversion decidability, closed subject reduction, or full closed consistency.*

Proof. Read the packet coordinates from Definition 765.1. The row U names the confluence and decidability handoffs together with substitution support, while C records the displayed continuation routes from U to the checker and confluence boundaries.

Use Definition 765.39 and Definition 765.40. They confine route replay to the visible rows and rule out hidden theorem coordinates beyond $S, N, O, U, D, H, C, P, L$.

The socket side is fixed by Definition 765.3 and Definition 765.25. The obstruction row O remains visible, and the discharge socket D remains separate from a supplied subject-reduction proof.

Consequently confluence and decidability uses are mediated by the same obstruction, socket, transport, route, provenance, and naming rows. They do not create an extra decision procedure or confluence theorem outside the finite packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket confluence decidability socket`

Theorem 765.62 (MetaCIC critical-path normalization-obstruction route coverage)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, every normalization-to-normal-form read from* S *to* N *keeps the obstruction row* O *visible along the route. The read may reach the normal-form consistency row through* H, C, P, L*, but it cannot pass through the route while hiding* O*, filling* D*, or upgrading the normal-form endpoint to closed subject reduction or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationObstructionRouteCoverage`

Proof. Start with the consistency handoff of Definition 765.2. It permits a consumer to read from the closed-strong-normalization row S to the normal-form consistency row N only through the displayed route rows H, C, P, L.

Next use Definition 765.3. The obstruction row O remains visible under the handoff row U and under componentwise transport, so the same route cannot be read as if the obstruction had disappeared.

Apply Definition 765.34 and Definition 765.40. They keep normal-form readback at the endpoint N and exhaust route replay inside the displayed packet.

The discharge socket is still separate by Definition 765.25. Therefore every normalization-to-normal-form read carries the obstruction row through the route and stops short of the two excluded theorem exports. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normalization obstruction route coverage`

Theorem 765.63 (MetaCIC residual-confluence budget bridge)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, a critical-path packet that exposes normalization obstruction, residual substitution boundary, finite-observation confluence, and candidate-mediated strong-normalization rows routes every residual-confluence consumer through the same finite budget ledger before any strong-normalization or confluence conclusion is exported. The bridge keeps the obstruction row* O*, the substitution-confluence-decidability handoff* U*, the discharge socket* D*, and the structural rows* H, C, P, L *visible throughout the read.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket residual confluence budget bridge`

Proof. Start with the normalization side. By Definition 765.58 and Definition 765.62, every read from the strong-normalization row S to the normal-form row N keeps the obstruction row O and the discharge socket D visible.

Next inspect the bounded handoff side. Apply Definition 765.59; the row U exhausts the substitution, confluence, and decidability reads as bounded handoff obligations rather than closed theorem exports.

The confluence-facing socket is then fixed by Definition 765.61. Any finite-observation confluence or residual-substitution read remains mediated by O, D, H, C, P, L and cannot split off into an independent Church–Rosser or beta-conversion decision procedure.

Finally, candidate-mediated strong-normalization consumers receive only this finite budget route. Transport, replay, provenance, and local naming preserve the same packet rows, so every residual-confluence consumer reaches the SN and confluence surfaces through the shared ledger before any conclusion is exported. □

Theorem 765.64 (MetaCIC critical-path residual-substitution normalization frontier)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, a consumer that reads residual substitution through* U *and reads normalization through the route from* S *to* N *must keep the obstruction row* O*, the discharge socket* D*, the structural transport row* H*, the replay row* C*, the provenance row* P*, and the local naming row* L *visible. Consequently residual substitution is not an export route for closed subject reduction, arbitrary Church–Rosser, beta-conversion decidability, or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSubstitutionNormalizationFrontier`

Proof. Use Definition 765.60 to read residual substitution through U. The handoff is a displayed packet coordinate, and its consumer must retain the obstruction and discharge rows rather than treating substitution support as a closed subject-reduction proof.

Use Definition 765.62 for the normalization side. The route from S to N keeps O visible, stops at normal-form readback, and does not fill the discharge socket D.

Now apply Definition 765.61. It keeps confluence-facing and decidability-facing reads socket-mediated by O, D, H, C, P, L, so the same consumer cannot split residual substitution into arbitrary Church–Rosser or beta-conversion decidability.

Finally use Definition 765.63. The residual-confluence bridge shares the finite budget ledger with the normalization route, so transport, replay, provenance, and local naming preserve the displayed frontier rather than exporting closed consistency. □

Theorem 765.65 (MetaCIC critical-path candidate-mediated SN conditional discharge)**.** *Let $K = (S, N, O, U, D, H, C, P, L)$ be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the candidate-mediated SN row attached to K has consumed the subject-reduction obstruction row O, the normalization audit route $S \to N$, the decidability boundary contained in U, the normal false exclusion endpoint N, and the residual substitution rows mediated by U, then K discharges the local SN handoff condition for the public critical-path route. This discharge is only the local handoff condition named by the packet; it does not claim full closed SN, full Church–Rosser, global beta-conversion decidability, closed subject reduction, or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNConditionalDischarge`

Proof. Project the packet carrier. The displayed rows $S, N, O, U, D, H, C, P, L$ keep the normalization route, normal-form endpoint, subject-reduction obstruction, substitution-confluence-decidability handoff, discharge socket, transport, replay, provenance, and local naming visible.

Use the obstruction and consistency handoff theorems first. They route normalization through $S \to N$, keep the subject-reduction obstruction O visible, and keep D as a local socket rather than as a closed subject-reduction theorem.

Next compose the public provenance and frontier results. The public provenance rows name only the cited obstruction, normalization audit, decidability boundary, and normal false exclusion packets, while the candidate-mediated frontier theorems route candidate SN, residual substitution, confluence, and bounded checking through the same finite ledger.

The stated hypothesis says that this finite candidate-mediated ledger has already consumed the listed obstruction, audit, boundary, endpoint, and residual rows. Therefore the local SN handoff condition for the public critical-path route is discharged, but no theorem outside that local handoff is exported. □

Theorem 765.66 (MetaCIC critical-path L10 residual route)**.** *Let $K = (S, N, O, U, D, H, C, P, L)$ be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the candidate-mediated SN row attached to K consumes the subject-reduction obstruction row O, the normalization route $S \to N$, the decidability boundary in U, the normal-false endpoint N, and the residual rows mediated by U, then every L10 four-object exit certificate read from K must route through the same finite residual ledger before any* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{Completion}^{\uparrow}$*-facing consumer can read it. The visible exit route is the shared packet surface O, U, D, H, C, P, L together with the normalization endpoint N; it exports no closed subject reduction, no full strong normalization theorem, no global Church–Rosser theorem, no beta-conversion decision procedure, and no closed consistency theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ResidualRoute`

Proof. Start with the public provenance route. The public rows identify the obstruction, normalization audit, decidability boundary, normal-false endpoint, residual substitution, replay, provenance, and local naming data as rows of the same accepted packet, so an L10 exit certificate cannot detach them into an external theorem export.

Use Definition 765.63. It keeps residual-confluence reads on the finite budget ledger and prevents a confluence-facing consumer from bypassing O, D, H, C, P, L.

Next apply Definition 765.64. Residual substitution and normalization share the same frontier: the route through U and the route $S \to N$ must keep the obstruction row, discharge socket, transport, replay, provenance, and naming rows visible.

Finally apply Definition 765.65. Its hypothesis is exactly the consumed candidate-mediated ledger named in the statement, so the L10 four-object exit certificate can be read only after that ledger has discharged the local SN handoff. The Real- or Completion-facing consumer therefore receives a routed residual certificate, not an independent closed MetaCIC metatheorem. □

Theorem 765.67 (MetaCIC critical-path L10 discharge dependency bridge)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, any L10 four-object exit certificate read from the packet must consume the same candidate-mediated SN discharge row, residual substitution row, subject-reduction obstruction row* O*, normalization endpoint* N*, and decidability boundary in* U *before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer reads it. The bridge records only this dependency order; it does not claim full subject reduction, Church–Rosser, beta-conversion decidability, strong normalization, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10DischargeDependencyBridge`

Proof. Begin with Definition 765.65. Its conditional discharge row is local to the same packet and is available only after the obstruction row, normalization route, decidability boundary, normal endpoint, and residual rows have already been consumed.

Next apply Definition 765.64. Residual substitution and normalization share the same finite frontier through U, $S \to N$, O, D, H, C, P, L, so the L10 exit cannot read residual substitution as a detached subject-reduction theorem.

Use Definition 765.63 to keep confluence-facing and SN-facing reads on the shared finite budget ledger. Then apply Definition 765.66; it routes every L10 four-object exit certificate through that residual ledger before Real- or Completion-facing consumers can inspect it.

Therefore the exit certificate consumes the same discharge, residual, obstruction, endpoint, and decidability rows in the same packet. The consumer receives a dependency bridge for the L10 route, not any of the blocked global MetaCIC conclusions. □

Theorem 765.68 (MetaCIC critical path candidate-mediated SN boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, a normalization-boundary read that displays CandidateMediatedSN evidence, the closed-term substitution boundary, FiniteObservationConfluence, the SubjectReductionDischargeSocket row, and MetaCIC closure-trace rows can be exposed only through those displayed rows together with the* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *support already carried by the chapter dependency route. The read is a finite boundary certificate; it is not global strong normalization, host confluence, semantic model completeness, closed subject reduction, or a filled discharge socket.*

Proof. Project the accepted critical-path packet. The rows $S, N, O, U, D, H, C, P, L$ are the only local coordinates, and the candidate-mediated normalization read begins at the displayed S route rather than at a global normalization theorem.

Attach the boundary rows in their finite order. Closed-term substitution and finite-observation confluence are read through the handoff row U, the subject-reduction socket remains the displayed row D, and closure-trace information is replayed only through C with provenance and naming in P, L.

Now use the L10 dependency route already recorded for the chapter. Its $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ support rows keep the boundary read attached to finite observation schedules, rational readback, and the terminal seal before any public consumer can inspect it.

The obstruction and socket theorems keep the excluded conclusions out of the packet. The same projection that exposes the candidate-mediated boundary refuses global strong normalization, host confluence, semantic model completeness, closed subject reduction, and any completed discharge for D. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNBoundary`
`semantic name certificate`

Theorem 765.69 (MetaCIC candidate-mediated SN public handoff)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, the public candidate-mediated strong-normalization handoff is the finite readiness surface formed by the candidate-mediated SN rows, the normalization-frontier rows, the confluence-decidability budget, and the subject-reduction discharge socket. A consumer may cite this surface only as a route through* $S, N, O, U, D, H, C, P, L$ *together with the displayed CandidateMediatedSN and CandidateSNConfluenceBoundary rows. It cannot cite the surface as full MetaCIC strong normalization, closed subject reduction, full confluence, full closed consistency, or a filled discharge bundle.*

Proof. Start with the candidate-mediated obligation surface. Theorems 765.257, 765.265, and 765.269 identify the candidate-mediated SN rows as finite packet data rather than as a global normalization theorem.

Next read the public carrier route. Theorems 765.58, 765.47, and 765.62 keep the normalization-frontier endpoint attached to S, N, O, H, C, P, L and keep the obstruction row visible.

Finally attach the socket and handoff rows. Theorems 765.25, 765.50, 765.51, and 765.194 keep D separate from the candidate-mediated rows and keep U as a handoff row. Thus the public handoff is a readiness surface and not a proof of full MetaCIC SN, closed subject reduction, full confluence, full closed consistency, or socket discharge. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNPublicHandoff`

Theorem 765.70 (MetaCIC confluence frontier public handoff)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, every public confluence-frontier read factors through CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, and SubjectReductionDischargeSocket rows only as displayed packet obligations. The route may expose finite-observation confluence readiness and confluence-decidability handoff data, but it exports no arbitrary Church–Rosser theorem, no global beta-conversion decision procedure, no full MetaCIC confluence theorem, and no closed subject-reduction discharge.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceFrontierPublicHandoff`

Proof. Read the confluence-facing packet rows first. Theorems 765.61, 765.193, and 765.224 force confluence and decidability uses through U together with the obstruction, socket, transport, replay, provenance, and local naming rows.

Next attach the candidate and finite-observation frontier. Theorems 765.220, 765.223, and 765.181 keep CandidateMediatedSN and CandidateSNConfluenceBoundary as mediated readiness rows, while the finite-observation confluence side remains bounded by the same packet route.

Finally use the public handoff theorem above together with discharge-socket coverage, Definition 765.69 and Definition 765.26. The subject-reduction socket remains visible and unfilled, so

the confluence frontier is a public readiness surface over finite BEDC rows rather than a global Church–Rosser, beta-decidability, full-confluence, or subject-reduction theorem. □

Theorem 765.71 (MetaCIC critical-path formal target router)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$*, any candidate-mediated strong-normalization consumer for* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *must pass through the candidate-mediated SN public boundary of Definition 765.69, the confluence frontier of Definition 765.70, and the L10 normalization discharge budget of Definition 765.273 before any* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{StreamName}^{\uparrow}$ *discharge row is read. The router exposes a finite target route through* $S, N, O, U, D, H, C, P, L$*; it does not export hidden infinite reduction, full MetaCIC strong normalization, ambient real completeness, or a completed subject-reduction socket.*

Proof. Begin with the candidate-mediated SN public boundary. It places CandidateMediatedSN and CandidateSNConfluenceBoundary on the displayed carrier route and prevents a consumer from treating them as full strong normalization.

Next attach the confluence frontier. The frontier keeps FiniteObservationConfluence and MetaCICNormalizationFrontier on the same public route, while the subject-reduction socket remains visible as a socket rather than as a closed theorem.

Finally use the L10 normalization discharge budget. That budget orders the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ discharge faces behind the finite source rows and the socket-aware route.

The nonescape clauses from these three surfaces compose: no hidden infinite reduction, ambient real-completeness claim, or closed subject-reduction discharge is available outside the displayed packet. Hence every formal target consumer must pass through the stated router before reading the L10 discharge rows. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFormalTargetRouter`

Theorem 765.72 (MetaCIC critical-path public export)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier*

$$K = (S, N, O, U, D, H, C, P, L),$$

the public export surface is exactly the finite route through CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, SubjectReductionDischargeSocket, and the carrier rows $S, N, O, U, D, H, C, P, L$*. This export refuses full MetaCIC strong normalization, closed subject reduction, full confluence, full closed consistency, and discharge of the socket* D*.*

Proof. First read the candidate-mediated SN public handoff, Definition 765.69. It presents CandidateMediatedSN and CandidateSNConfluenceBoundary only as finite readiness rows over the carrier route.

Next read the confluence-frontier public handoff, Definition 765.70. It attaches FiniteObservationConfluence and MetaCICNormalizationFrontier to the same carrier rows, while keeping confluence and decidability uses bounded by the displayed handoff data.

Finally use discharge-socket coverage and obstruction-socket separation, Definition 765.26 and Definition 765.50. These keep SubjectReductionDischargeSocket visible but unfilled, so D remains a socket coordinate rather than a closed subject-reduction proof or a source of the excluded global theorem exports. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicExport`

Theorem 765.73 (MetaCIC critical-path candidate ledger public boundary)**.** *Every public candidate-ledger read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *carrier* $K = (S, N, O, U, D, H, C, P, L)$ *must pass through the L10 four-object exit rows already listed for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The ledger may expose CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, SubjectReductionDischargeSocket, and the carrier route* $S, N, O, U, D, H, C, P, L$, *but it cannot claim global strong normalization, full confluence, global decidability, closed subject reduction, closed consistency, discharge of* D, *or Lean verification.*

Proof. Project the accepted carrier and read the public export theorem Definition 765.72. The export surface is already the finite candidate-mediated route through the carrier rows and the visible subject-reduction socket.

Now project the L10 exit budget from Definition 765.128. Any public candidate-ledger read must pass through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before it reaches a downstream consumer.

The two projections compose without adding theorem content. The candidate ledger records readiness rows and visible sockets over the four-object exit surface, while the excluded global SN, confluence, decidability, subject-reduction, consistency, socket-discharge, and Lean-verification claims remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateLedgerPublicBoundary`

Theorem 765.74 (MetaCIC critical-path public consumer surface)**.** *Every public consumer of the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *critical path must route through CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, SubjectReductionDischargeSocket, and the carrier rows*

$$S, N, O, U, D, H, C, P, L.$$

The consumer surface may expose candidate-mediated normalization readiness, confluence-frontier readiness, finite-observation confluence data, normalization-frontier data, the visible subject-reduction socket, and the displayed carrier route, but it exports no closed subject reduction, full confluence, full closed consistency, or socket discharge.

Proof. First read Definition 765.69. It keeps CandidateMediatedSN and CandidateSNConfluence-Boundary on the public readiness surface and routes them through the carrier rows $S, N, O, U, D, H, C, P, L$.

Next read Definition 765.70. It attaches FiniteObservationConfluence and MetaCICNormalizationFrontier to the same displayed packet route and refuses arbitrary Church–Rosser, global beta-decision, full confluence, and closed subject-reduction exports.

Finally read Definition 765.72. It names the public export surface as exactly the route through CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, SubjectReductionDischargeSocket, and $S, N, O, U, D, H, C, P, L$. Therefore a public consumer inherits that exact route and none of the excluded closed subject-reduction, full-confluence, full-consistency, or socket-discharge conclusions. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicConsumerSurface`

Theorem 765.75 (MetaCIC critical-path residual confluence bridge boundary)**.** *Every residual-substitution route that consumes the candidate-mediated SN row, the confluence-frontier row, and the subject-reduction obstruction row of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must consume*

them only through the displayed MetaCIC critical-path ledger $S, N, O, U, D, H, C, P, L$ together with the public candidate-ledger and confluence-frontier handoff rows. The bridge may pass residual-confluence readiness to a downstream consumer, but it cannot treat the blocked subject-reduction edge as a closed theorem, discharge socket D, or replace the critical-path ledger by a host confluence proof.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBridgeBoundary`

Proof. First project the confluence-frontier slice through Definition 765.70. This places CandidateMediatedSN, CandidateSNConfluenceBoundary, FiniteObservationConfluence, MetaCICNormalizationFrontier, and SubjectReductionDischargeSocket on the same displayed route.

Next apply the candidate-ledger public boundary, Definition 765.73. It forces the residual-substitution consumer to pass through the L10 exit rows and the packet ledger $S, N, O, U, D, H, C, P, L$ before any public read is available.

Now apply the confluence public handoff, Definition 765.74. The handoff permits only the finite readiness surface, so residual confluence is read as a bridge boundary over displayed rows rather than as a global Church–Rosser or beta-conversion theorem.

Finally use obstruction-socket separation, Definition 765.50. The obstruction row and the subject-reduction socket remain separate from a closed discharge, which prevents the blocked subject-reduction edge from being consumed as a theorem. □

Theorem 765.76 (MetaCIC critical-path frontier-rank NameCert obligations)**.** *For every accepted frontier-rank packet*

$$Q = (O, R, D, B, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *surface consists exactly of seven row obligations: open-node visibility through O, dependency-ready visibility through R, downstream-rank visibility through D, obstruction-ledger visibility through B, replay-route visibility through C, package provenance through P, and local naming through N. It has no eighth obligation for scheduler state, host reachability, or an unledgered dependency edge.*

Proof. Project the accepted frontier-rank carrier of Definition 765.53. The displayed packet has the seven coordinates O, R, D, B, C, P, N, and the definition assigns one scheduler-facing role to each coordinate.

Open-node exhaustion in Definition 765.54 supplies the coverage step. Every accepted open-node read factors through the node row, the dependency-ready rows, the downstream rank, the obstruction ledger, the replay route, provenance, and the local name.

Dependency-route nonescape in Definition 765.55 supplies the exclusion step. Dependency routing cannot appeal to hidden scheduler state, host graph reachability, or an unledgered dependency edge.

The Lean handoff surface of Definition 765.56 identifies the same seven coordinates as the formal target surface. Hence the local NameCert obligations are exactly the seven displayed rows and no extra scheduler-facing row. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRankNameCertObligations`

Theorem 765.77 (MetaCIC critical-path frontier-rank transport stability)**.** *For every accepted frontier-rank packet $Q = (O, R, D, B, C, P, N)$, componentwise* hsame *transport and replay along the displayed route C preserve the seven NameCert obligation rows O, R, D, B, C, P, N. Transport cannot add scheduler state, host reachability, or an unledgered dependency edge.*

Proof. Start with the accepted carrier in Definition 765.53. Its route coordinate C stores the Cont paths used by scheduler-facing consumers, while the remaining coordinates record the open node, readiness, rank, ledger, provenance, and local name.

Open-node exhaustion keeps the transported read on the same finite packet. By Definition 765.54, there is no hidden frontier node and no unranked scheduling obligation available after the read is replayed.

Dependency-route nonescape keeps the replayed dependency route inside the same ledgered surface. By Definition 765.55, the route may use R, B, C, P, N but cannot leave the packet for scheduler state, host reachability, or an unledgered edge.

The obligation theorem above identifies the seven rows as the complete local NameCert surface. Therefore componentwise hsame transport and route replay preserve exactly those rows and introduce no additional dependency source. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRankTransportStability`

Theorem 765.78 (MetaCIC critical-path frontier-rank consumer exactness)**.** *Every scheduler-facing consumer of the* MetaCICCriticalPath$^{\uparrow}$ *root-unblock packet factors through the seven-row frontier-rank surface*

$$O, R, D, B, C, P, N.$$

Such a consumer may read the open node, readiness row, downstream rank, obstruction ledger, replay route, provenance, and local name, but it cannot read a hidden dependency edge outside those rows.

Proof. Project the consumer to an accepted open-node read. By Definition 765.54, that read factors through a frontier-rank packet $Q = (O, R, D, B, C, P, N)$ and has no coordinate for a hidden frontier node or unranked scheduling obligation.

Now project any dependency route used by the consumer. By Definition 765.55, routing is exhausted by the displayed readiness, ledger, replay, provenance, and naming rows, and it cannot appeal to hidden scheduler state, host graph reachability, or an unledgered edge.

Finally use the Lean handoff and NameCert obligation surfaces. Definition 765.56 makes the seven rows the exact root-unblock handoff, and Definition 765.76 makes them the complete local NameCert obligation list. Thus every scheduler-facing consumer factors through the seven rows and cannot read a hidden dependency edge. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRank consumer exactness`

Theorem 765.79 (MetaCIC critical-path frontier-rank soundness)**.** *The* MetaCICCriticalPath$^{\uparrow}$ *frontier-rank carrier can rank only the displayed open-node row O, dependency-route row R, Lean-handoff row D, obstruction-ledger row B, replay-route row C, provenance row P, and local naming row N. It cannot rank hidden closed-consistency data, subject-reduction data, scheduler state, host reachability, or any dependency edge outside the displayed finite packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierRankSoundness`

Proof. Start with the frontier-rank carrier of Definition 765.53. Its rank coordinate is read together with the open node, dependency-ready routes, obstruction ledger, replay route, provenance, and local name, so the carrier provides no separate coordinate for closed-consistency or subject-reduction data.

Open-node exhaustion and dependency-route nonescape give the two exclusions. By Definition 765.54, every ranked node is one of the displayed open nodes. By Definition 765.55, every

dependency used by the rank is routed through the ledgered finite rows and not through host reachability or an unledgered edge.

Finally combine the Lean handoff, NameCert obligation, transport stability, and consumer exactness rows of Definition 765.56, Definition 765.76, Definition 765.77, and Definition 765.78. These identify the same seven rows as the only downstream handoff surface, so the frontier rank is sound exactly for those displayed rows. □

Theorem 765.80 (MetaCIC critical-path downstream-unblock coverage)**.** *A downstream MetaCIC consumer is unblocked by a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *frontier-rank packet only when every dependency route named by the packet is read through the displayed finite rows* O, R, D, B, C, P, N*. A route that asks for hidden closed-consistency data, subject-reduction data, scheduler state, host reachability, or an unledgered dependency edge is not covered by the unblock packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDownstreamUnblockCoverage`

Proof. Project the downstream consumer through Definition 765.78. The consumer can receive only the open node, readiness row, downstream rank, obstruction ledger, replay route, provenance, and local name.

Use dependency-route nonescape and frontier-rank soundness. The theorem Definition 765.55 forces every named dependency route through the displayed route and ledger rows, while Definition 765.79 prevents the rank from certifying hidden closed-consistency or subject-reduction data.

The remaining rows are stable under transport and handoff. Apply Definition 765.77, Definition 765.56, and Definition 765.76. They keep the downstream unblock claim on the same seven finite rows, so any consumer whose dependency route leaves those rows is not unblocked by this packet. □

Theorem 765.81 (MetaCIC critical-path Lean target intake)**.** *For every accepted frontier-rank packet*

$$Q = (O, R, D, B, C, P, N),$$

the Lean target intake surface is exactly the seven displayed rows: open-node visibility through O*, dependency-ready visibility through* R*, downstream-rank visibility through* D*, obstruction-ledger visibility through* B*, replay-route visibility through* C*, package provenance through* P*, and local naming through* N*. It has no intake row for scheduler state, host graph reachability, hidden dependency edges, or theorem exports outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathLeanTargetIntake`

Proof. Start from the carrier of Definition 765.53. The packet Q displays precisely the seven coordinates O, R, D, B, C, P, N, and each coordinate has a named root-unblock role.

Apply open-node exhaustion from Definition 765.54. It places every accepted open-node read inside this same packet and rules out a hidden frontier node or unranked scheduling obligation.

Use dependency-route nonescape and the Lean handoff surface, Definition 765.55 and Definition 765.56. Together they make the dependency-ready row, obstruction ledger, replay route, provenance row, and local NameCert row the exact formal handoff surface.

Finally invoke consumer exactness from Definition 765.78. Since every scheduler-facing consumer factors through the seven displayed rows, the Lean intake has no remaining coordinate for scheduler state, host reachability, hidden edges, or outside theorem exports. □

Theorem 765.82 (MetaCIC critical-path downstream unblock surface)**.** *Every downstream unblock read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *frontier-rank packet* $Q = (O, R, D, B, C, P, N)$ *must pass through the dependency-ready row* R*, obstruction ledger* B*, replay route* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row* N*. The read may also inspect the open node* O *and downstream rank* D*, but it cannot bypass* R, B, C, P, N *by appealing to scheduler state, host graph reachability, a hidden edge, or an exported theorem outside the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDownstreamUnblockSurface`

Proof. Project the read to the displayed frontier-rank carrier. The dependency-ready row R, obstruction ledger B, replay route C, provenance row P, and local name N are all coordinates of the same finite packet Q.

Apply Definition 765.55. Dependency routing is exhausted by those rows, so a downstream unblock cannot use a host scheduler convention or graph-reachability fact in place of the ledgered route.

Use the Lean handoff surface of Definition 765.56. It identifies the same rows as the paper-to-Lean root-unblock surface and refuses additional scheduling coordinates.

Finally apply Definition 765.78. Every scheduler-facing consumer already factors through O, R, D, B, C, P, N, hence any downstream unblock read must pass through R, B, C, P, N before it can be consumed. □

Theorem 765.83 (MetaCIC critical-path audit frontier factorization)**.** *Every audit-frontier read of an accepted frontier-rank packet* $Q = (O, R, D, B, C, P, N)$ *factors through the displayed open-node row, dependency-ready row, downstream rank, obstruction ledger, replay route, package provenance, and local NameCert row. The factorization has no scheduler state, host graph reachability, hidden dependency edge, or theorem export outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathAuditFrontierFactorization`

Proof. Read the frontier through Definition 765.53. The audit-facing packet is finite and names only O, R, D, B, C, P, N.

Use Definition 765.54. It exhausts the open-node side of the audit frontier and prevents a hidden frontier node or unranked scheduling obligation from entering the factorization.

Next apply Definition 765.55 and Definition 765.77. These keep dependency routes and transported replay paths inside R, B, C, P, N and preserve the same seven rows under componentwise hsame transport.

Finally use Definition 765.56 and Definition 765.78. The formal handoff and every consumer read share the same seven-row packet, so the audit-frontier factorization cannot export scheduler state, host graph reachability, hidden edges, or theorem rows outside Q. □

Theorem 765.84 (MetaCIC critical-path root-unblock carrier readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the displayed rows expose carrier readiness through the source row S*, local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row* N*, open-obstruction row* O*, unblock row* U*, discharge row* D*, componentwise* hsame *row* H*, continuation row* C*, package provenance row* P*, and local ledger row* L*. The readiness claim has no row for closed subject reduction, full consistency, hidden scheduler state, or host reachability outside the packet.*

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootUnblockCarrierReadiness

Proof. Start with the carrier and NameCert rows of the accepted packet. The displayed source, local name, obstruction, unblock, discharge, transport, continuation, provenance, and ledger coordinates are the only coordinates available to a carrier-readiness read.

Apply the carrier obligations and transport stability supplied by Definition 765.76 and Definition 765.77. These keep the source and local naming rows tied to the same finite BHist packet and keep every transported read inside the displayed hsame row.

Finally use Definition 765.79 and Definition 765.83. They exclude closed-consistency data, subject-reduction data, scheduler state, and hidden dependency edges from the frontier packet, so carrier readiness is exactly the displayed finite surface. □

Theorem 765.85 (MetaCIC critical-path root-unblock route readiness)**.** *Every route replay accepted by a* MetaCICCriticalPath$^{\uparrow}$ *root-unblock consumer is exhausted by the displayed obstruction, discharge socket, transport, continuation, provenance, and local naming rows of* $K = (S, N, O, U, D, H, C, P, L)$*. No replay step may replace those rows by a host scheduler edge, an unledgered dependency route, or a hidden theorem-export path.*

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootUnblockRouteReadiness

Proof. Project the route replay to the packet rows that can be consumed. The obstruction, discharge, hsame transport, Cont continuation, package provenance, and local naming rows are all present in the same finite root-unblock packet.

Use route nonescape and the discharge socket boundary. The dependency-route nonescape theorem Definition 765.55 forces every routed dependency through the ledgered finite rows, while the downstream unblock surface of Definition 765.82 prevents a consumer from bypassing the displayed discharge and replay rows.

The remaining replay data are fixed by the audit-frontier factorization. Applying Definition 765.83 with transport stability from Definition 765.77 leaves only obstruction, socket, transport, continuation, provenance, and local naming reads for the root-unblock route. □

Theorem 765.86 (MetaCIC critical-path root-unblock consumer readiness)**.** *Downstream MetaCIC consumers may depend on an accepted* MetaCICCriticalPath$^{\uparrow}$ *root-unblock packet only through the displayed rows* $S, N, O, U, D, H, C, P, L$*. Such consumers cannot appeal to host scheduler state, undisplayed dependency edges, hidden route closures, closed subject reduction, or full consistency as part of the root-unblock packet.*

Proof. Begin with consumer exactness. By Definition 765.78, every scheduler-facing consumer factors through the displayed finite packet rather than through host scheduler state.

Next use downstream-unblock coverage and the audit-frontier factorization. Theorems Definition 765.80 and Definition 765.83 restrict dependency reads to the packet rows and rule out hidden scheduler edges or theorem exports outside the packet.

Finally combine carrier readiness and route readiness from Definition 765.84 and Definition 765.85. These two surfaces account for the carrier, obstruction, discharge, transport, continuation, provenance, ledger, and local naming rows, so no downstream MetaCIC consumer has any additional root-unblock dependency to cite. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootUnblockConsumerReadiness

Theorem 765.87 (MetaCIC candidate-mediated SN obligation scope)**.** *Any confluence-facing or decidability-facing consumer of candidate-mediated strong-normalization data through an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet must expose typed-term carrier rows, reduction-candidate rows, substitution and reduction replay rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *row before it reads the strong-normalization boundary. The read is a finite obligation scope for the packet, not an unconditional confluence theorem, beta-conversion decision procedure, closed subject-reduction certificate, or host normalization principle.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNScope`

Proof. Begin with root-unblock consumer readiness. It forces downstream MetaCIC consumers through the displayed root-unblock rows $S, N, O, U, D, H, C, P, L$ before any confluence-facing or decidability-facing dependency is consumed.

Use frontier-rank consumer exactness and downstream-unblock coverage. These surfaces keep the open node, dependency-ready row, obstruction ledger, replay route, provenance, and local name visible, so candidate-mediated SN data cannot be read through hidden scheduler state or an unledgered dependency edge.

Finally compose the carrier and route readiness theorems. The typed-term carrier rows, reduction-candidate rows, substitution and reduction replay rows, hsame transport, Cont replay, Pkg provenance, and local NameCert row are precisely the rows available at the boundary; no row supplies unconditional confluence, beta-decision, closed subject reduction, or host normalization. □

Theorem 765.88 (MetaCIC critical-path public route certificate)**.** *Every public read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet factors through the displayed finite route certificate*

$$K = (S, N, O, U, D, H, C, P, L).$$

The factorization reads closed-normalization through S, normal-form consistency through N, the subject-reduction obstruction through O, decidability and handoff data through U, the discharge socket through D, componentwise transport through H, route replay through C, provenance through P, and local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *naming through L. It has no public row for closed subject reduction, full closed consistency, global beta-conversion decidability, a hidden dependency edge, or an exported theorem outside the packet.*

Proof. Start with the obligation surface. Theorem 765.20 lists the public packet rows and identifies the local NameCert obligations with the finite tuple $S, N, O, U, D, H, C, P, L$.

The positive consistency route is fixed by the handoff and route inventory. Apply Theorem 765.2, Theorem 765.39, and Theorem 765.40. These results keep the closed-normalization row, normal-form endpoint, handoff row, continuation row, transport row, provenance row, and local naming row inside the displayed carrier.

It remains to exclude public escape routes. Theorem 765.3 keeps the subject-reduction obstruction visible, while Theorem 765.86 restricts downstream consumers to the same displayed rows. Hence every public read is a read of the finite route certificate and not a read of an undisplayed theorem export. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicRouteCertificate`

Theorem 765.89 (MetaCIC critical-path obstruction scope separation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the obstruction row* O *remains a scoped subject-reduction boundary. The consistency row* N*, the closed-normalization route* S*, the handoff row* U*, route replay through* C*, componentwise transport through* H*, provenance through* P*, and local naming through* L *cannot convert* O *or the socket* D *into closed subject reduction or full closed consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathObstructionScopeSeparation`

Proof. Read the obstruction row from the packet carrier. Theorem 765.3 states that O remains visible under the handoff row and componentwise transport, and that agreement on the other rows cannot erase the obstruction.

The consistency and normalization side stays on its own route. Theorem 765.2 lets a consumer travel from S to N through the displayed rows, but it stops at the normal-form endpoint unless the separate discharge socket is filled.

Finally use route exhaustion and obligation coverage. Theorem 765.40 prevents replay from leaving the displayed route, and Theorem 765.20 leaves no extra coordinate that could identify O with a supplied theorem. The obstruction row therefore remains a scoped boundary rather than a theorem produced by the consistency or normalization rows. □

Theorem 765.90 (MetaCIC critical-path downstream consumer readiness)**.** *Downstream MetaCIC consumers may use an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet only as a finite public route certificate over* $S, N, O, U, D, H, C, P, L$*. They may cite it for carrier admission, closed-normalization readback, normal-form consistency handoff, obstruction visibility, decidability handoff, discharge-socket locality, route replay, transport stability, provenance, and local NameCert naming. They may not cite it as closed subject reduction, full normalization, internal consistency for MetaCIC, or a global decision procedure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDownstreamConsumerReadiness`

Proof. Begin with the root-unblock surface. Theorem 765.86 already confines downstream consumers to the displayed packet rows and excludes host scheduler state, hidden route closures, closed subject reduction, and full consistency.

The public route certificate supplies the usable rows. Theorem 765.88 packages the normalization route, normal-form endpoint, obstruction row, handoff data, discharge socket, transport, continuation, provenance, and local NameCert row as the complete public read surface.

The obstruction boundary fixes what is not available. By Theorem 765.89, the subject-reduction obstruction and discharge socket remain separated from the consistency and normalization rows. A downstream consumer can therefore depend on the packet as a finite route certificate, but not as subject reduction, full normalization, internal consistency, or a global decision procedure. □

Theorem 765.91 (MetaCIC critical-path kernel frontier exposure)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the kernel-facing frontier exposes exactly the finite BHist *rows for closed normalization-frontier source data* S*, the normal-form endpoint* N*, the consistency-facing refusal rows* O, D*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *naming* L*. The frontier has no row for closed subject reduction, full closed consistency, hidden scheduler state, or a host reachability edge.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket kernel frontier exposure`

Proof. Project the carrier inventory of Definition 765.1. The displayed tuple is the only packet read available to the kernel frontier, and its coordinates already name the closed normalization-frontier rows, the normal-form endpoint, the refusal rows, transport, replay, provenance, and local naming.

Use the public route certificate and obstruction scope separation. The route certificate reads every public kernel-facing path through $S, N, O, U, D, H, C, P, L$, while the obstruction separation keeps the consistency-facing refusal rows O, D visible rather than turning them into subject reduction or full consistency.

Thus exposure is a projection from the finite packet followed by route replay. Since the carrier has no additional coordinate, the listed rows exhaust the kernel frontier. □

Theorem 765.92 (MetaCIC critical-path normalization consumer handoff)**.** *Every normalization-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must read the closed normalization-frontier rows through* S*, then consume the normal-form row* N *through the displayed* Cont *route* C*, with componentwise* hsame *transport* H*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *naming* L *still attached. The handoff cannot bypass the consistency-facing refusal rows* O, D *and cannot reinterpret the handoff row* U *as a closed-consistency theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationConsumerHandoff`

Proof. Begin with the consistency handoff and public route certificate. They place the normalization-facing read on the route from the closed normalization-frontier source row S to the normal-form row N and keep the route inside the displayed packet.

Next read the replay and transport rows. The Cont row C records the replay path, the hsame row H preserves componentwise agreement of the same packet rows, and the Pkg and $\mathsf{NameCert}$ rows P, L name the provenance and local boundary of that handoff.

Finally apply the obstruction boundary. The refusal rows O, D remain visible along the handoff, so the consumer receives a bounded normalization-to-normal-form route rather than a proof of closed subject reduction or full consistency. □

Theorem 765.93 (MetaCIC critical path normalization-frontier consumer route)**.** *Every normalization-frontier consumer read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that is passed to an L10-facing route must expose, before consumption, the strong-normalization source row* S*, the normal-form endpoint row* N*, the obstruction row* O*, the handoff row* U*, the discharge-socket row* D*, and the structural rows* H, C, P, L*. The route may cite the Lean-side namespace* `BEDC.Derived.MetaCICCriticalPathUp` *as the intended formalization surface, but the paper obligation remains the finite visibility condition on*

$$S, N, O, U, D, H, C, P, L.$$

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationFrontierConsumerRoute`

Proof. Start with the normalization consumer handoff of Definition 765.92. It already forces the normalization-facing read to move from the closed normalization-frontier source S to the normal-form endpoint N through the displayed replay route C, with H, P, L still attached.

Next retain the frontier boundaries. The obstruction row O and the discharge-socket row D cannot be bypassed, and the handoff row U remains a visible substitution, confluence, and decidability handoff rather than a closed-consistency theorem.

Finally expose the L10-facing consumer surface. A route read by downstream Real-and-Completion consumers must therefore present the candidate strong-normalization source, normal-form endpoint, obstruction, handoff, discharge socket, componentwise transport, replay, provenance, and local naming rows together. This is exactly the finite public-route surface later targeted by `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicRouteCertificate`, without adding a new Lean marker at this paper site. □

Theorem 765.94 (MetaCIC critical-path consistency refusal boundary)**.** *Every consistency-facing read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet is bounded by the same finite* BHist *surface*

$$S, N, O, U, D, H, C, P, L.$$

The closed normalization-frontier rows may route a consumer to the normal-form endpoint N*, but the consistency-facing refusal rows* O, D*, together with the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows, prevent the read from exporting closed subject reduction, full closed consistency, global beta-conversion decidability, or quotient data over MetaCIC terms.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConsistencyRefusalBoundary`

Proof. Read the consistency-facing surface from the packet carrier. The only available rows are the closed normalization-frontier source, normal-form endpoint, obstruction and discharge-socket refusal rows, handoff row, transport, replay, provenance, and local naming.

Use route exhaustion to keep every consumer read inside this surface. The Cont replay and hsame transport rows preserve the same finite packet, while Pkg provenance and the local NameCert row identify the route as the MetaCICCriticalPath packet rather than an ambient theorem export.

The refusal boundary is then immediate from the absence of hidden rows. Since neither closed subject reduction nor full closed consistency is a carrier coordinate, every consistency-facing read remains bounded by O, D and the displayed packet rows. □

Theorem 765.95 (MetaCIC critical-path kernel frontier totality)**.** *Every kernel-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet reads the complete frontier through*

$$S, N, O, U, D, H, C, P, L.$$

The frontier is total for the exported public route: closed normalization-frontier source data, the normal-form endpoint, obstruction rows, handoff rows, discharge socket, componentwise transport, displayed replay, provenance, and local naming are all visible, while closed subject reduction, full closed consistency, hidden scheduler state, and host reachability remain outside the exported surface.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathKernelFrontierTotality`

Proof. First project the carrier rows from Definition 765.1. The packet exposes only $S, N, O, U, D, H, C, P, L$, so every kernel-facing read starts from that finite inventory.

Use the public route certificate of Definition 765.91. It identifies the kernel frontier with the same displayed rows and gives no coordinate for closed subject reduction, full closed consistency, hidden scheduler state, or host reachability.

Finally apply the obstruction scope separation of Definition 765.3 and the refusal boundary of Definition 765.94. The obstruction and discharge socket remain visible as bounded rows, so the exported frontier is total exactly at the displayed packet surface. □

Theorem 765.96 (MetaCIC critical-path normal-form public nonescape)**.** *Every public normal-form consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet reaches the normal-form row* N *only through the displayed route*

$$S, N, O, U, D, H, C, P, L.$$

The normal-form endpoint is public as a bounded frontier endpoint, not as closed subject reduction, full closed consistency, hidden scheduler state, or host reachability.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket normal form public nonescape`

Proof. Begin with the normalization consumer handoff, Definition 765.92. It routes the consumer from the closed normalization-frontier row S to the normal-form row N through C, with transport, provenance, and local naming still attached.

Next use kernel frontier totality, Definition 765.95. The public route has no coordinate outside $S, N, O, U, D, H, C, P, L$, so a normal-form read cannot leave the displayed frontier to obtain hidden scheduler state or host reachability.

The obstruction scope separation of Definition 765.3 and the consistency refusal boundary of Definition 765.94 keep O and D present beside the normal-form endpoint. Therefore the endpoint remains a public bounded row rather than an exported proof of closed subject reduction or full closed consistency. □

Theorem 765.97 (MetaCIC critical-path discharge-socket frontier exhaustion)**.** *Every discharge-socket read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet is exhausted by*

$$S, N, O, U, D, H, C, P, L.$$

The socket row D *names the missing subject-reduction input as an exported frontier boundary, while the surrounding rows preserve and replay that boundary without supplying closed subject reduction, full closed consistency, hidden scheduler state, or host reachability.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketFrontierExhaustion`

Proof. Project the socket row from the carrier. The row D is listed as a discharge-bundle socket in Definition 765.1, and the public route certificate of Definition 765.91 keeps that row inside the same kernel-facing frontier.

Apply the obstruction scope separation, Definition 765.3. It prevents O from being erased and prevents D from being converted into a discharged bundle by agreement on the other packet rows.

Finally use the consistency refusal boundary, Definition 765.94, together with normal-form public nonescape, Definition 765.96. The discharge socket remains an exposed frontier boundary, and every consumer read is exhausted by $S, N, O, U, D, H, C, P, L$ rather than by a hidden scheduler, a host reachability edge, or either excluded theorem. □

Theorem 765.98 (MetaCIC critical-path Phase Real route budget)**.** *Every Phase Real and Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *critical-path packet must first read the displayed rows*

$$S, N, O, U, D, H, C, P, L$$

before using any Real-facing dependency row, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, or* $\mathsf{StreamName}^{\uparrow}$ *schedule row. The packet exports the source row, normal-form row, obstruction row, unblock row, discharge socket, transport row, continuation route, package provenance, and local ledger row as the complete L10 consumer surface; it exports no closed subject reduction, full consistency, hidden completion selector, or ambient real-completeness row.*

Proof. Start with the root-unblock packet. Theorem 765.84 exposes the carrier surface as the finite tuple $S, N, O, U, D, H, C, P, L$, and Theorem 765.85 keeps route replay inside the same displayed rows.

Now place the Phase Real and Completion consumers on the visible L10 route. The StreamName, regular-sequence, and Real handoff chapters chapter 126, chapter 301, and chapter 38 supply consumer-facing rows only after their schedules, regular rational data, and real seals are read as displayed dependency rows.

Finally apply Theorem 765.86. It bars host scheduler state, undisplayed dependency edges, hidden route closures, closed subject reduction, and full consistency from the root-unblock surface. Hence any Real-facing or Completion-facing read must consume $S, N, O, U, D, H, C, P, L$ before it can use StreamName, RegSeqRat, or Real dependency rows. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealRouteBudget`

Theorem 765.99 (MetaCIC critical-path Phase Real normal-form handoff)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the normal-form row* N *can feed the Phase Real route only through the same obstruction and discharge-socket packet. It cannot feed that route as closed subject reduction, full consistency, global beta-conversion decidability, or an unledgered proof object independent of* O, D, H, C, P, L*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealNormalFormHandoff`

Proof. Read the normal-form endpoint from the packet surface. Theorem 765.84 places N beside the obstruction row O, the discharge socket D, componentwise transport H, route replay C, package provenance P, and local ledger L.

Route the endpoint through the same public consumer boundary. Theorem 765.85 keeps replay inside the displayed packet, while Theorem 765.86 prevents a downstream consumer from replacing the packet by closed subject reduction or full consistency.

The Phase Real route therefore receives N only as a normal-form row still attached to O, D, H, C, P, L. The sibling StreamName, RegSeqRat, and Real chapters named in chapter 126, chapter 301, and chapter 38 consume such rows as finite dependency data, not as hidden MetaCIC consistency theorems. □

Theorem 765.100 (MetaCIC critical-path Phase Real obstruction ledger lock)**.** *For every Real-facing or Completion-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the obstruction row* O *and discharge socket* D *remain visible ledger rows throughout the route to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *dependency surfaces. No consumer may erase* O *or discharge* D *by appealing to a Real seal, a regular rational sequence handoff, a stream schedule, a completion carrier, closed subject reduction, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealObstructionLedgerLock`

Proof. Begin with the carrier and route readiness results. Theorems 765.84 and 765.85 keep O and D among the displayed packet rows and route every replay through the finite tuple $S, N, O, U, D, H, C, P, L$.

Next read the downstream consumer surface. Theorem 765.86 states that downstream consumers have no hidden scheduler edge, hidden route closure, closed subject-reduction row, or full-consistency row beyond the displayed packet.

The Real and Completion route can then use only the sibling dependency rows supplied by chapter 126, chapter 301, and chapter 38. Those rows may schedule, regularize, or seal a Real-facing

dependency read, but they do not supply a coordinate that removes the MetaCIC obstruction row or fills the discharge socket. □

Theorem 765.101 (MetaCIC critical-path L10 discharge dependency readback)**.** *For every L10 four-object exit read from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the route must first read the candidate-mediated strong-normalization discharge row, the residual-substitution row, the obstruction row O, the normal-form row N, and the decidability boundary U before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *consumer may inspect the route. The dependency readback is a finite route condition only: it does not close full subject reduction, Church–Rosser, beta-conversion decidability, strong normalization, or consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10DischargeDependencyReadback`

Proof. Start with the Phase Real route budget of Definition 765.98. It forces every Real-facing or Completion-facing consumer to read the displayed packet rows $S, N, O, U, D, H, C, P, L$ before it can use StreamName, RegSeqRat, or Real dependency rows.

Next use the normal-form handoff of Definition 765.99. The row N reaches the Phase Real route only while still attached to the obstruction and discharge-socket packet, so the candidate-mediated strong-normalization discharge row and residual-substitution row cannot be hidden behind the normal endpoint.

Finally apply the obstruction ledger lock of Definition 765.100. It keeps O and D visible throughout the route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependency surfaces.

The same read order controls a $\mathsf{Completion}^{\uparrow}$ consumer, because such a consumer enters through the Real-facing route budget rather than through an additional completion oracle. Since the displayed rows contain no full subject-reduction, Church–Rosser, beta-conversion-decidability, strong-normalization, or consistency coordinate, the theorem records only the L10 discharge dependency order. □

Theorem 765.102 (MetaCIC critical-path split obligation surface)**.** *Every critical-path consumer that claims a* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *handoff must expose, before the claim is consumed, the candidate-mediated strong-normalisation row, finite-observation confluence row, closure-trace row, residual substitution boundary row, normalization-frontier row, and subject-reduction discharge row. These six rows are read as displayed packet coordinates tied to the same* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}^{\uparrow}}$ *surface; no consumer may replace them by a host scheduler edge, a hidden theorem export, a global Church–Rosser assertion, or an unstated subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSplitObligationSurface`

Proof. Project the accepted critical-path packet to the rows already named by the MetaCIC carrier and its downstream child surfaces. The candidate-mediated strong-normalisation row supplies the normalization source, and the normalization-frontier row records the finite point at which that source may be inspected.

Read the confluence and closure rows next. Finite-observation confluence is available only as a displayed observation row, while the closure trace records the finite replay route rather than a global confluence theorem.

The residual substitution boundary and subject-reduction discharge rows are then kept visible as boundaries. They can be transported and replayed through hsame and Cont, but they cannot be collapsed into a supplied subject-reduction theorem.

Finally the structural rows source and name the same six-row surface through Pkg and $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPath}}$. Since host scheduler edges, hidden theorem exports, global Church–Rosser, and unstated subject reduction are not packet coordinates, every critical-path handoff consumer must expose the six displayed obligation rows before claiming the handoff. □

Theorem 765.103 (MetaCIC residual-diamond source-package budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual-diamond consumer or candidate-SN consumer of K must expose the L10 source package*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

together with the residual-substitution rows, local residual-join rows, substitution-residual rows, bounded local-confluence rows, obstruction row O, discharge socket D, transport row H, replay row C, provenance row P, and local naming row L before it claims a discharge. The budget is a route-exhaustion condition over displayed NameCert rows; it is not a source-equivalence transport theorem and it exports no global strong-normalization theorem, global confluence theorem, beta-conversion decision procedure, quotient term equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondSourcePackageBudget`

Proof. Start with the carrier of Definition 765.1. The public packet rows are exactly $S, N, O, U, D, H, C, P, L$, with O and D kept visible as the obstruction row and discharge socket.

Read the L10 source package through the existing critical-path scope. The mature source surface names $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the source faces that must be displayed before Real-facing or completion-facing consumers inspect the route.

Now inspect the residual and candidate sides. The residual-substitution rows, local residual-join rows, substitution-residual rows, and bounded local-confluence rows are all consumed through the same finite NameCert packet; the candidate-SN row may use that budget only while the obstruction, socket, transport, replay, provenance, and local naming rows remain attached.

Thus every residual-diamond or candidate-SN discharge claim is exhausted by the displayed source package and residual rows. Since no displayed coordinate carries global SN, global confluence, beta-decidability, quotient term equality, selected limits, ambient Real completeness, or kernel consistency, none of those objects is exported by the budget. □

Theorem 765.104 (MetaCIC critical-path StreamName source readiness)**.** *For every Real-facing or Completion-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *critical-path packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the $\mathsf{StreamName}^{\uparrow}$ *source schedule may be exposed only after the consumer has displayed the MetaCIC packet rows $S, N, O, U, D, H, C, P, L$ and the Phase Real route budget rows attached to that packet. In particular, the source schedule is not a host scheduler coordinate, an undisplayed stream source, a hidden completion selector, closed subject reduction, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealStreamNameSourceReadiness`

Proof. Start with the root-unblock carrier and route surfaces. Theorems 765.84 and 765.85 expose K as the finite displayed packet and keep route replay inside the same rows.

Next apply the Phase Real route budget. Theorem 765.98 requires every Real-facing and Completion-facing consumer to read $S, N, O, U, D, H, C, P, L$ before it can use a StreamName schedule row.

Finally use consumer readiness and the two Phase Real locks. Theorem 765.86 excludes host scheduler state and hidden route closures, while Theorems 765.99 and 765.100 keep the normal-form and obstruction rows attached to the route. Thus the $\mathsf{StreamName}^{\uparrow}$ source schedule is exposed only as a displayed dependency row after the MetaCIC packet rows. □

Theorem 765.105 (MetaCIC critical-path RegSeqRat readback readiness)**.** *For every Real-facing or Completion-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback can be consumed only as a displayed regular-rational dependency row attached to the same packet. The readback cannot be detached into quotient stream equality, selected limit data, hidden Real completeness, closed subject reduction, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRegSeqRatReadbackReadiness`

Proof. First read the root-unblock packet. Theorems 765.84 and 765.85 make the carrier and replay rows visible as $S, N, O, U, D, H, C, P, L$.

The Phase Real route budget then fixes the readback order. By Theorem 765.98, the regular-rational dependency row appears only after the displayed MetaCIC packet rows have been admitted.

The normal-form handoff and obstruction ledger lock preserve the same attachment downstream. Theorems 765.99 and 765.100 keep N, O, D, H, C, P, L present at the Real and Completion boundary, and Theorem 765.86 refuses undisplayed dependency edges. Hence the $\mathsf{RegSeqRat}^{\uparrow}$ readback is consumed only as a displayed row of the same critical-path route. □

Theorem 765.106 (MetaCIC critical-path Real completion readiness package)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the Real-facing and Completion-facing root-unblock package is exactly the combination of the displayed* $\mathsf{StreamName}^{\uparrow}$ *source schedule readiness, the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback readiness, and the* $\mathsf{Real}^{\uparrow}$ *seal route read after the Phase Real budget rows. The package exports no ambient Real completeness, quotient stream equality, selected limit, closed subject reduction, full consistency, or hidden completion selector.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionReadinessPackage`

Proof. Build the StreamName side first. Theorem 765.104 confines the source schedule to a displayed row admitted after $S, N, O, U, D, H, C, P, L$.

Build the regular-rational side next. Theorem 765.105 confines the $\mathsf{RegSeqRat}^{\uparrow}$ readback to the same critical-path packet and refuses quotient-stream, selected-limit, and hidden-completeness exits.

Finally attach the Real-facing seal route. The Phase Real route budget, normal-form handoff, and obstruction ledger lock from Theorems 765.98, 765.99, and 765.100 keep the $\mathsf{Real}^{\uparrow}$ route downstream of the displayed packet while preserving N, O, D, H, C, P, L. With root consumer readiness from Theorem 765.86, the resulting package is a finite root-unblock surface for Phase Real and Completion consumers, not a closure of any excluded metatheoretic or completion principle. □

Theorem 765.107 (MetaCIC critical-path discharge-socket totality route)**.** *Any Real-facing or Completion-facing consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that reads the discharge-bundle socket must pass through the displayed rows* $S, N, O, U, D, H, C, P, L$*, the* $\mathsf{StreamName}^{\uparrow}$ *source schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal readiness route. The route exposes socket totality as a finite displayed dependency. It does not discharge closed subject reduction, closed consistency, quotient stream equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketTotalityRoute`

Proof. Start with the three readiness surfaces in this child. Theorems 765.104, 765.105, and 765.106 force Real-facing and Completion-facing consumers to display the StreamName source row, the RegSeqRat readback row, and the Real seal route after the packet rows have been admitted.

Next read the root-unblock route rows in the parent chapter. Theorems 765.84, 765.85, and 765.86 make $S, N, O, U, D, H, C, P, L$ the finite carrier, route, and consumer surface for the accepted packet.

The discharge-bundle socket is therefore exposed only as the displayed row D together with its transport, continuation, provenance, and local naming support. Since the readiness package and the root-unblock rows contain no coordinate for closed subject reduction, closed consistency, quotient stream equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, reading socket totality cannot discharge any of those excluded principles. □

Theorem 765.108 (MetaCIC critical-path DyadicRatCore budget readiness)**.** *For every Phase Real or Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget read is permitted only as a displayed dependency row attached to the same packet. The read may bound finite rational observations for the route to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, but it cannot be exported as quotient stream equality, a host-selected limit, Real completeness, closed subject reduction, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDyadicRatCoreBudgetReadiness`

Proof. Start with the finite root-unblock carrier. Theorem 765.84 exposes K as the displayed rows $S, N, O, U, D, H, C, P, L$, and Theorem 765.85 keeps route replay inside those rows.

Place the dyadic observation budget on the Phase Real route. Theorem 765.98 says that every Real-facing or Completion-facing dependency read must first pass through the displayed L10 consumer surface, and Theorem 765.100 keeps the obstruction and discharge rows visible along the route.

Finally use the readiness triad. Theorems 765.104, 765.105, and 765.106 allow StreamName schedules, RegSeqRat readback, and Real seal reads only as displayed dependency rows. Hence a $\mathsf{DyadicRatCore}^{\uparrow}$ finite observation budget can serve the Phase Real consumer only as such a row, with no coordinate for quotient equality, selected limits, Real completeness, closed subject reduction, or full consistency. □

Theorem 765.109 (MetaCIC critical-path StreamName before RegSeqRat route order)**.** *For every Phase Real or Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the route order is fixed: the* $\mathsf{StreamName}^{\uparrow}$ *schedule row is read before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and both are read before the* $\mathsf{Real}^{\uparrow}$ *seal handoff. A consumer that tries to read regular-rational data or a Real seal without the displayed StreamName schedule is outside the L10 dependency budget of the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathStreamNameRegSeqRatRouteOrder`

Proof. First read the root packet. Theorems 765.84 and 765.86 make $S, N, O, U, D, H, C, P, L$ the whole consumer surface and exclude host scheduler state, undisplayed dependency edges, closed subject reduction, and full consistency.

Next read the Phase Real budget. Theorem 765.98 requires the displayed rows before any Real-facing dependency row, $\mathsf{RegSeqRat}^{\uparrow}$ handoff row, or $\mathsf{StreamName}^{\uparrow}$ schedule row can be used. Theorem 765.99 then keeps the normal-form row attached to the obstruction and discharge packet during the handoff.

The readiness triad supplies the order. Theorem 765.104 exposes the StreamName schedule after the packet rows, Theorem 765.105 consumes the RegSeqRat readback as a displayed row of that same packet, and Theorem 765.106 packages both before the Real-facing seal route. Thus the route reads StreamName before RegSeqRat and RegSeqRat before the Real seal handoff. □

Theorem 765.110 (MetaCIC critical-path Real seal frontier exactness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the exact Phase Real frontier is the displayed L10 dependency budget consisting of the packet rows together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal handoff rows consumed in that order. The frontier exports no quotient stream equality, host-selected limit, ambient Real completeness, closed subject reduction, full consistency, hidden completion selector, or theorem row outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealSealFrontierExactness`

Proof. Begin with the root-unblock exactness. Theorems 765.84, 765.85, and 765.86 identify the available consumer surface as $S, N, O, U, D, H, C, P, L$ and refuse hidden scheduler, route, subject-reduction, and consistency coordinates.

Add the Phase Real budget rows. Theorem 765.98 fixes the L10 consumer surface, while Theorems 765.99 and 765.100 preserve the normal-form, obstruction, discharge, transport, continuation, provenance, and ledger rows along the Real-facing route.

The first two theorems of this file identify the dyadic budget and route order. Theorem 765.108 confines finite observation budgets to displayed dependency rows, and Theorem 765.109 forces the StreamName schedule before RegSeqRat readback and the Real seal handoff.

Finally apply the readiness package, Theorem 765.106. It combines the StreamName, RegSeqRat, and Real-facing rows without adding quotient stream equality, selected limits, ambient completeness, closed subject reduction, full consistency, or a hidden completion selector. The Phase Real seal frontier is therefore exact at the displayed L10 dependency budget and nowhere beyond it. □

Theorem 765.111 (MetaCIC critical-path L10 source cut)**.** *For every Phase Real or Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the source cut for a* $\mathsf{Real}^{\uparrow}$*-facing read must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget before the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows. A consumer that starts from quotient stream equality, a host-selected limit, Real completeness, closed consistency, or an undisplayed route row is outside the L10 source cut of* K*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10SourceCut`

Proof. Start with the dyadic budget row. Theorem 765.108 permits $\mathsf{DyadicRatCore}^{\uparrow}$ reads only as displayed finite observation budget rows attached to the packet K.

Next use the route order. Theorem 765.109 forces the $\mathsf{StreamName}^{\uparrow}$ schedule before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and both before the $\mathsf{Real}^{\uparrow}$ seal handoff.

The displayed source cut is therefore the packet rows followed by the dyadic budget, the StreamName schedule, the RegSeqRat readback, and the Real seal row. Since none of the cited rows has a coordinate for quotient equality, selected limits, Real completeness, closed consistency, or an undisplayed route row, no Phase Real consumer may read those objects as sources for the handoff. □

Theorem 765.112 (MetaCIC critical-path Real seal non-escape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the* $\mathsf{Real}^{\uparrow}$ *seal handoff exports exactly the displayed L10 dependency budget of Theorem 765.110. It exports no Real completeness theorem, no host-selected limit, no quotient stream equality, no closed consistency row, and no extra completion selector.*

Proof. Read the handoff through the readiness package. Theorem 765.106 packages the StreamName, RegSeqRat, and Real-facing rows while keeping them inside the displayed packet surface.

Then apply the frontier theorem. Theorem 765.110 identifies the exact frontier as the L10 dependency budget and explicitly removes quotient stream equality, selected limits, ambient completeness, closed subject reduction, full consistency, hidden completion selectors, and theorem rows outside the packet.

Combining these two facts leaves the Real seal handoff with only the displayed dependency rows. Hence Real completeness, host-selected limits, quotient stream equality, closed consistency, and extra completion selectors cannot escape through the seal. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealSealNoCompletenessEscape`

Theorem 765.113 (MetaCIC critical-path Phase Real exit conjunction)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, this packet supplies the paper-side exit conjunction for the Phase Real and Completion route over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*: finite observation reads are displayed as dyadic budget rows, schedule reads precede regular-rational readback, regular-rational readback precedes the Real seal, and the Real seal frontier exports no quotient equality, selected limit, ambient completeness, closed consistency, or off-budget completion selector. Verification status and bridge status remain on the independent Lean axis.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealExitConjunction`

Proof. The dyadic part of the conjunction is Theorem 765.108. It makes finite observation reads visible as $\mathsf{DyadicRatCore}^{\uparrow}$ budget rows of K.

The route-order part is Theorem 765.109. It places the $\mathsf{StreamName}^{\uparrow}$ schedule before $\mathsf{RegSeqRat}^{\uparrow}$ readback and puts both before the $\mathsf{Real}^{\uparrow}$ seal handoff.

The exit boundary is supplied by Theorems 765.110, 765.111, and 765.112. Together they identify the source cut, the exact frontier, and the refusal of quotient equality, selected limits, Real completeness, closed consistency, and extra completion selectors.

These are paper-side dependency and non-escape clauses. They state the Phase Real exit surface without adding a verification marker or a bridge row, so the Lean verification grade and bridge grade stay governed by the separate closurestatus axis. □

Theorem 765.114 (MetaCIC critical-path Phase Real budget exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *serving the Phase Real and Completion route, the packet has no L10 dependency budget outside the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite*

observation row, StreamName$^{\uparrow}$ *schedule row,* RegSeqRat$^{\uparrow}$ *readback row,* Real$^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local ledger. Any consumer that asks for quotient stream equality, a host-selected limit, ambient Real completeness, closed consistency, bridge status, or a verification upgrade is outside this paper-side budget.*

Proof. First read the displayed dependency rows. Theorem 765.108 confines finite observations to the DyadicRatCore$^{\uparrow}$ row, and Theorem 765.109 forces the StreamName$^{\uparrow}$ schedule before RegSeqRat$^{\uparrow}$ readback and the Real$^{\uparrow}$ seal.

Next identify the boundary. Theorems 765.110, 765.111, and 765.112 make the Real-facing frontier exact at the displayed L10 rows and exclude quotient stream equality, host-selected limits, ambient Real completeness, closed consistency, and extra completion selectors.

Finally apply Theorem 765.113. It packages the same dyadic budget, schedule order, readback, seal, and non-escape clauses as the Phase Real exit surface. The transport, replay, provenance, and local ledger rows preserve that surface, while verification status and bridge status remain independent closurestatus axes rather than additional L10 dependency rows. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealBudgetExhaustion`

Theorem 765.115 (MetaCIC critical-path L10 budget exit)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *serving the Phase Real and Completion route, the paper-side L10 exit budget is satisfied exactly when the displayed route reads, in order, the* DyadicRatCore$^{\uparrow}$ *finite observation rows,* StreamName$^{\uparrow}$ *schedules,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local ledger. The same exit budget excludes quotient equality, selected limits, Real completeness, closed consistency, and bridge status from the L10 dependency surface.*

Proof. First expose the dyadic and schedule side. Apply Theorem 765.108 for the finite observation rows and Theorem 765.109 for the order placing StreamName schedules before RegSeqRat readback and the Real seal handoff.

Next identify the Real-facing frontier. Apply Theorem 765.110 to make the seal frontier exact at the displayed L10 dependency budget, and use Theorem 765.111 to force the source cut through the dyadic, stream, readback, and seal rows.

Now apply the non-escape and conjunction theorems. Theorem 765.112 removes Real completeness, selected limits, quotient stream equality, closed consistency, and extra completion selectors; Theorem 765.113 repacks the remaining rows as the Phase Real exit surface.

Finally use Theorem 765.114. It says there is no L10 dependency budget outside the displayed finite observation, schedule, readback, seal, transport, replay, provenance, and local ledger rows. Therefore the stated displayed route is exactly the L10 budget exit, while bridge status and verification status remain outside the paper-side dependency budget. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10BudgetExit`

Theorem 765.116 (MetaCIC critical-path candidate SN obligation discharge)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *serving the Phase Real and Completion route, a candidate-mediated strong-normalization obligation may be discharged only after the displayed* DyadicRatCore$^{\uparrow}$ *finite observation rows,* StreamName$^{\uparrow}$ *schedules,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local ledger rows have all been read. Before those rows are visible, the candidate obligation remains an open boundary of the L10 budget rather than a closed subject-reduction, consistency, quotient-equality, or Real-completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNObligationDischarge`

Proof. Start with the dyadic source cut. Theorems 765.108 and 765.111 force every Phase Real read to begin with displayed finite observation budget rows rather than quotient equality, selected limits, or an undisplayed route.

Next use the route-order and Real frontier theorems. Theorem 765.109 places StreamName before RegSeqRat and the Real seal, while Theorem 765.110 keeps the seal exact at that budget.

Finally apply the exit conjunction, budget exhaustion, and L10 budget exit theorems. They say that the transport, replay, provenance, and local ledger rows complete the paper-side route and that no off-budget completion selector, closed subject-reduction row, consistency row, quotient equality, or Real completeness row is exported. Thus candidate-mediated strong-normalization obligations can be discharged only after the displayed L10 rows have been read. □

Theorem 765.117 (MetaCIC critical-path confluence residual boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *serving the Phase Real and Completion route, every confluence residual boundary remains inside the same L10 budget of* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and the local ledger. The residual boundary exports no closed subject reduction, quotient equality, hidden scheduler, Real completeness row, full consistency row, or bridge status.*

Proof. Project the Real-facing budget through Theorem 765.115. The only exit rows are the dyadic budget, StreamName schedule, RegSeqRat readback, Real seal, transport, replay, provenance, and local ledger.

Use Theorems 765.112 and 765.114. They remove Real completeness, quotient equality, selected limits, closed consistency, bridge status, and extra completion selectors from the paper-side dependency surface.

The residual boundary is therefore confined to the displayed rows. Any consumer asking for closed subject reduction, a hidden scheduler, quotient equality, Real completeness, full consistency, or bridge status is asking outside the L10 budget rather than reading a confluence residual exported by K. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceResidualBoundary`

Theorem 765.118 (MetaCIC critical-path L10 entry budget synchronization)**.** *For every current Phase Real or Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

entry into the Phase Real route is permitted only after the packet displays the $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal rows through finite* BHist, hsame, Cont, Pkg, *and* NameCert *provenance attached to* K*. The entry surface refuses quotient reals, host equality, trichotomy, open-cover compactness, and completed-real claims.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10EntryBudgetSynchronization`

Proof. First expose the packet itself. Theorems 765.84, 765.85, and 765.86 make $S, N, O, U, D, H, C, P, L$ the displayed carrier, replay, and consumer surface.

Next read the Phase Real dependency budget. Theorems 765.98, 765.99, and 765.100 keep the normal-form, obstruction, discharge, continuation, package, and provenance rows visible before a Real-facing handoff.

Finally synchronize the four L10 source rows. Theorems 765.104, 765.105, 765.106, 765.108, 765.109, and 765.110 identify the dyadic budget, schedule, readback, and seal rows as displayed dependency rows of the same packet. None of these rows carries quotient reals, host equality, trichotomy, open-cover compactness, or a completed-real theorem, so the entry budget is exactly the finite provenance surface described above. □

Theorem 765.119 (MetaCIC critical-path Real completion frontier filter)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any Completion-facing claim passed through its Real frontier is admitted only when it is backed by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *L10 source rows of Theorem 765.118. Claims that require quotient reals, host equality, trichotomy, open-cover compactness, or completed-real data are filtered out of the packet frontier.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket real completion frontier filter`

Proof. Begin with the synchronized entry budget. Theorem 765.118 requires the four displayed L10 source rows before the packet can enter the Phase Real route.

The frontier then reads only those rows. Theorem 765.110 identifies the exact seal frontier with the displayed L10 dependency budget, while Theorems 765.111 and 765.112 exclude source reads and seal exports outside that budget.

The filter is therefore constructive and finite: a claim survives only by naming the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows attached to K. Quotient reals, host equality, trichotomy, open-cover compactness, and completed-real data have no displayed row in that list, so they cannot pass through the frontier. □

Theorem 765.120 (MetaCIC critical-path current open-phase readiness)**.** *The current paper-side readiness surface consumed by the Phase Real and Completion route for* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *is the conjunction of the Phase Real dependency budget, the L10 entry budget synchronization of Theorem 765.118, and the Real completion frontier filter of Theorem 765.119. This surface is a finite* BHist/hsame/Cont/Pkg/NameCert *dependency surface and not a claim of quotient reals, host equality, trichotomy, open-cover compactness, or completed-real closure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCurrentOpenPhaseReadiness`

Proof. The dependency budget supplies the route rows. Theorems 765.98, 765.99, and 765.100 expose the finite packet rows consumed by the Phase Real route.

The entry theorem then synchronizes the sources. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows to be displayed through finite provenance before route entry.

The frontier theorem gives the open-phase filter. Theorem 765.119 admits only claims backed by those four displayed source rows and refuses quotient reals, host equality, trichotomy, open-cover compactness, and completed-real data. Together these rows are exactly the current paper-side readiness surface used by the Phase Real and Completion route. □

Theorem 765.121 (MetaCIC critical-path Phase Real dependency diamond)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, a public Phase Real or Completion entry is accepted exactly when its route factors through the four displayed source rows attached to* K*: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The four projections of this factorization recover those source rows*

and no other public entry row is available. In particular, the diamond exports no quotient stream equality, selected host limit, host equality, trichotomy, open-cover compactness, or completed-real theorem.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealDependencyDiamond`

Proof. Start from the current open-phase surface. Theorem 765.120 says that a Phase Real or Completion consumer sees the Phase Real dependency budget together with the synchronized L10 entry surface and the Real completion frontier filter.

The entry side supplies the four displayed sources. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before public entry, while Theorem 765.119 admits only claims backed by those rows at the Completion frontier.

The projections are forced by the readiness and order rows. Theorems 765.104, 765.105, 765.106, and 765.108 identify the four sources as rows of the same packet K.

Finally use the route-order and seal-frontier exactness theorems. Theorems 765.109 and 765.110 place those rows in the displayed diamond and exclude off-budget exits. Hence the public entry route factors through exactly the four source rows, and the excluded completion or host principles have no exported coordinate. □

Theorem 765.122 (MetaCIC critical-path four source-owner noncollapse)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any Phase Real or Completion public entry accepted through the dependency diamond of Theorem 765.121 retains four separate source-owner records: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite observation budget owner, the* $\mathsf{StreamName}^{\uparrow}$ *schedule owner, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback owner, and the* $\mathsf{Real}^{\uparrow}$ *seal owner. Identifying two of these owners, replacing one owner by an aggregate route-status token, or substituting quotient stream equality, selected host limit, host equality, trichotomy, open-cover compactness, closed consistency, or completed-real data places the entry outside the accepted finite route.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourSourceOwnerNoncollapse`

Proof. Start with the finite surface. Theorem 765.120 says that the Phase Real and Completion consumer sees only the displayed BHist, hsame, Cont, Pkg, and NameCert rows attached to the packet K.

The entry budget then fixes the owner list. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows as four displayed sources before the route is entered.

The frontier removes every substitute. Theorem 765.119 rejects claims that need quotient stream equality, selected host limits, host equality, trichotomy, open-cover compactness, closed consistency, or completed-real data rather than one of the displayed source rows.

Finally apply the diamond. Theorem 765.121 recovers exactly the four projections from an accepted public entry. If two owners were identified, or if an aggregate route-status token replaced one owner, one of those projections would no longer recover its named source row. Such an entry therefore lacks the required four-coordinate factorization and is not an accepted finite route. □

Theorem 765.123 (MetaCIC critical-path L10 prefix dependency induction)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any Phase Real or Completion route admitted by Theorem 765.121 has four finite route prefixes: the $\mathsf{DyadicRatCore}^{\uparrow}$ *budget prefix, the* $\mathsf{StreamName}^{\uparrow}$ *schedule prefix, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback prefix,*

and the $\mathsf{Real}^{\uparrow}$ *seal prefix. These prefixes occur in that source order, each factors through the displayed packet rows of Theorems 765.118, 765.119, 765.121, and 765.122, and each preserves the obstruction, discharge, host, continuation, package, and ledger rows* O, D, H, C, P, L*. No prefix may be replaced by quotient stream equality, selected host limits, host equality, trichotomy, open-cover compactness, closed subject reduction, full consistency, or completed-real data.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10PrefixDependencyInduction`

Proof. Begin with the entry requirement. Theorem 765.118 admits a Phase Real route only after the packet displays the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows through finite BHist, hsame, Cont, Pkg, and NameCert provenance.

The frontier supplies the exclusion step. Theorem 765.119 says that Completion-facing claims survive only when backed by those displayed rows, so the forbidden substitutes have no route coordinate to occupy.

The diamond gives the constructive prefix split. Theorem 765.121 recovers the four projections from an accepted entry, hence the route can be read as the ordered finite prefix chain from budget to schedule to readback to seal.

Finally keep the owners and packet rows fixed. Theorem 765.122 prevents any two source owners from collapsing and rules out aggregate route-status replacements. Therefore each prefix remains attached to its own source row while carrying forward the common obstruction, discharge, host, continuation, package, and ledger rows O, D, H, C, P, L. □

Theorem 765.124 (MetaCIC critical-path RealCompleteness exact-boundary gate)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any Phase Real and Completion route admitted by the dependency diamond of Theorem 765.121 and then consumed as a $\mathsf{RealCompleteness}^{\uparrow}$ *exact-boundary handoff factors through the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and then through the accepted* RealCompletionExactBoundaryUp *packet of Theorem 1120.6. The gate exports no quotient stream equality, selected host limit, host equality, trichotomy, open-cover compactness, closed subject reduction, full consistency, metric completeness, aggregate bridge token, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletenessExactBoundaryGate`

Proof. First read the MetaCIC side. Theorems 765.118, 765.119, and 765.121 force any admitted Phase Real and Completion route to display the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before it can be consumed downstream.

The prefix and owner constraints keep this route finite. Theorems 765.122 and 765.123 prevent the four owners from collapsing, preserve the packet rows O, D, H, C, P, L, and rule out substitutions by quotient streams, selected host limits, host equality, trichotomy, open-cover compactness, subject reduction, consistency, or completed-real data.

Now pass to the RealCompleteness handoff. Theorem 450.33 says that an exact-boundary consumer for $\mathsf{RealCompleteness}^{\uparrow}$ must be read through an accepted RealCompletionExactBoundaryUp packet, and Theorem 450.34 reduces every finite consumer extension to the same boundary route.

Finally apply the four-face exactness theorem. Theorem 1120.6 identifies the boundary faces as exactly the dyadic, stream, regular-sequence, and Real faces exported by the accepted packet. Hence the downstream handoff is a repacking of the same four finite faces rather than a new completion object, and the excluded quotient, host, compactness, consistency, metric-completeness, bridge, and $\mathsf{Real}^{\uparrow}$ completeness principles have no exported coordinate. □

Theorem 765.125 (MetaCIC critical-path exact-boundary prefix determinacy)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

suppose that a Phase Real and Completion route is admitted by the dependency diamond of Theorem 765.121 and is then consumed by RealCompletionExactBoundaryUp *through the RealCompleteness exact-boundary gate of Theorem 765.124. Then the ordered four prefixes of Theorem 765.123 determine the same exact-boundary handoff as Theorem 1120.6: the* DyadicRatCore$^{\uparrow}$ *budget prefix, the* StreamName$^{\uparrow}$ *schedule prefix, the* RegSeqRat$^{\uparrow}$ *readback prefix, and the* Real$^{\uparrow}$ *seal prefix, all carried by the common* BHist, hsame, Cont, Pkg, *and* NameCert *rows of the two accepted packets.*

Consequently every accepted downstream handoff is this four-prefix finite boundary. It is not determined by quotient stream equality, selected host limits, host equality, trichotomy, open-cover compactness, closed subject reduction, full consistency, metric completeness, aggregate bridge tokens, ambient Real$^{\uparrow}$ *completeness, or any hidden completion coordinate.*

Proof. Begin on the MetaCIC side. Theorem 765.118 exposes the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows through finite BHist, hsame, Cont, Pkg, and NameCert provenance attached to K.

The frontier and diamond then force the same ordered source list. Theorems 765.119 and 765.121 admit only routes backed by those four displayed rows, while Theorem 765.122 prevents owner collapse or replacement by an aggregate route-status token.

Use prefix induction to read the route constructively. Theorem 765.123 orders the prefixes as dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal, and keeps O, D, H, C, P, L attached to the same packet rows throughout the route.

Now pass through the RealCompleteness exact-boundary handoff. Theorems 450.33 and 450.34 reduce every accepted exact-boundary consumer to the matched four faces, with only the displayed hsame transports, Cont replays, Pkg provenance, and local NameCert rows.

Finally apply Definition 1120.6. The exact boundary exports exactly the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ faces, followed by the structural rows of the accepted packet. Thus the MetaCIC route and the Real-completion boundary name the same finite four-prefix handoff, and no quotient, host, compactness, consistency, metric-completeness, bridge, ambient-completeness, or hidden completion coordinate remains available. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathExactBoundaryPrefixDeterminacy`

Theorem 765.126 (MetaCIC critical-path L10 four-face source discipline)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any Phase Real and Completion consumer entering through K must preserve four distinct local source obligations. The DyadicRatCore$^{\uparrow}$ *face supplies only the finite dyadic observation budget, the* StreamName$^{\uparrow}$ *face supplies only the finite source schedule, the* RegSeqRat$^{\uparrow}$ *face supplies only the regular-rational readback route, and the* Real$^{\uparrow}$ *face supplies only the terminal seal handoff.*

The MetaCIC packet coordinates the order, refusal boundary, owner noncollapse, and exact-boundary handoff for those four faces. It does not discharge any face's carrier, classifier, stability row, local closurestatus, formalstatus, leantarget, or bridge row. In particular, no quotient stream equality, selected host limit, host equality, trichotomy, open-cover compactness, countable choice, closed subject reduction, full consistency, metric completeness, aggregate bridge token, or Real$^{\uparrow}$ *completeness claim can replace one of the four local source obligations.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceSourceDiscipline`

Proof. Begin with the synchronized entry route. Theorem 765.118 displays the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before Phase Real entry, while Theorem 765.121 recovers exactly those four projections from an accepted public entry.

Use prefix induction and the exact-boundary determinacy result to keep the route ordered. Theorem 765.123 reads the consumer path as the ordered prefix chain from dyadic budget to schedule to readback to seal. Theorem 765.125 then identifies the downstream exact-boundary handoff with the same four prefixes, so the MetaCIC packet contributes coordination of the route rather than a fifth source face.

Now read the objectwise status surface. Theorems 765.132, 765.136, and 765.140 force closurestatus, formalstatus, leantarget, and bridge rows to remain local to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Theorems 38.833 and 38.834 exclude any fused status or target row.

It remains to exclude substitutes for the local source obligations. Theorems 765.138, 765.139, and 765.142 rule out quotient stream equality, selected host limits, host equality, trichotomy, open-cover compactness, countable choice, and aggregate route credentials as replacements for a missing face. The exact-boundary gate and the four-object exit certificate additionally refuse closed subject reduction, full consistency, metric completeness, aggregate bridge tokens, and $\mathsf{Real}^{\uparrow}$ completeness as exported coordinates. Therefore the MetaCIC packet can coordinate the L10 route, but each of the four faces remains a separate local source obligation with its own carrier, classifier, stability, status, target, and bridge audit rows. □

Theorem 765.127 (MetaCIC critical-path L10 candidate-discharge gate)**.** *An accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *candidate-mediated normalization read may be scheduled for discharge only after it displays the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the obstruction row and discharge socket of the same packet. Agreement on the MetaCIC rows alone cannot close the L10-facing discharge.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10CandidateDischargeGate`

Proof. Start with the entry budget. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before an L10-facing route can enter the MetaCIC critical path.

Next read the current readiness surface. Theorem 765.120 keeps the obstruction and discharge data attached to the displayed finite packet rows, while Theorem 765.121 recovers the same four source projections from an accepted Phase Real or Completion entry.

Finally apply the ordered prefix and source-discipline gates. Theorem 765.123 keeps the dyadic, StreamName, RegSeqRat, and Real faces in source order, and Theorem 765.126 says that the MetaCIC packet coordinates those faces without discharging any local carrier, classifier, stability, status, target, or bridge row. Thus equality of the MetaCIC coordination rows is insufficient: the candidate may be scheduled for discharge only when the four named L10 faces, the obstruction row, and the discharge socket are all displayed in the accepted packet. □

Theorem 765.128 (MetaCIC critical-path L10 four-object exit certificate)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the current Phase Real and Completion exit surface is reached exactly by displaying four finite source rows attached to* K*: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite dyadic budget, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal row. Each displayed row is carried through finite* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *provenance, as synchronized by Theorem 765.118, filtered*

by Theorem 765.119, arranged by Theorem 765.121, and kept source-owner separated by Theorem 765.122. No additional exit coordinate is supplied by quotient stream equality, selected host limits, host real equality, trichotomy, open-cover compactness, countable choice, or completed-real closure.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10StatusBridgeObligation`

Proof. First take the synchronized L10 entry surface. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows to be displayed through finite BHist, hsame, Cont, Pkg, and NameCert provenance before a Phase Real or Completion consumer can enter the route.

Next pass to the Completion-facing frontier. Theorem 765.119 admits only claims backed by those four displayed rows, so every accepted exit certificate still reads from the same finite packet surface rather than from quotient stream equality, selected host limits, host real equality, trichotomy, open-cover compactness, countable choice, or completed-real data.

The dependency shape is then fixed by the diamond. Theorem 765.121 says that a public Phase Real or Completion entry factors through exactly the four displayed source rows and recovers no other public entry row.

Finally retain the four owners separately. Theorem 765.122 refuses any collapse of the dyadic budget owner, StreamName schedule owner, RegSeqRat readback owner, and Real seal owner into an aggregate route token. The four-object certificate is therefore precisely the current finite exit surface, and every listed host, choice, compactness, quotient, or completed-real principle remains outside the accepted packet route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourObjectExitCertificate`

Theorem 765.129 (MetaCIC critical-path L10 dependency budget)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any accepted exit read using the L10 face must display four dependency rows in order:*

1. *the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording the finite dyadic budget consumed by the request;*
2. *the* StreamName$^{\uparrow}$ *finite-observation row, recording the inspected stream windows;*
3. *the* RegSeqRat$^{\uparrow}$ *readback row, recording the regular rational data obtained from those windows;*
4. *the* Real$^{\uparrow}$ *seal row, recording the terminal Real-facing handoff reached after the preceding rows have been displayed.*

The budget is a dependency display over the existing BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows of* K*. It exports no closed consistency result, global normalization theorem, quotient real equality, or selected limit.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ExitDependencyBudget`
Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10DependencyBudget`

Proof. Apply the four-object exit certificate of Definition 765.128. It identifies the current Phase Real and Completion exit surface with exactly the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows attached to the accepted packet.

Read that certificate as an ordered dependency budget. The dyadic row supplies the tolerance request, the StreamName row supplies the finite observations for that request, the RegSeqRat row reads those observations as regular rational data, and the Real row seals only the displayed route.

The same certificate keeps the four rows inside finite BHist, hsame, Cont, Pkg, and NameCert provenance. Since the carrier supplies no further exit coordinate, a consumer cannot extract closed consistency, global normalization, quotient real equality, or a selected limit from this dependency budget. □

Theorem 765.130 (MetaCIC critical-path Real-completion public socket)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, a public MetaCIC critical-path consumer may read the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows only as an L10 boundary socket before the subject-reduction discharge bundle is filled. The socket may expose the Real-completion dependency route and the four finite source rows, but it exports no closed subject reduction, full closed consistency, global beta-conversion decidability, quotient term equality, selected host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionPublicSocket`

Proof. Start from the packet definition and the consistency handoff. The carrier rows $S, N, O, U, D, H, C, P, L$ keep the normal-form endpoint, obstruction row, handoff row, discharge socket, transport, replay, provenance, and local naming visible before any public consumer can read the route.

Next apply the obstruction boundary and the L10 exit dependency budget. The obstruction boundary keeps O and D visible, while Definition 765.129 orders the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal as four displayed dependency rows.

The public consumer therefore receives only the L10 boundary socket. The four Real-completion rows can be inspected as finite source data carried through BHist, hsame, Cont, Pkg, and NameCert provenance, but the subject-reduction discharge bundle remains a separate obligation.

Finally the same projection refuses the excluded exports. Since the packet contains no coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, quotient term equality, selected host limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those claims is produced by reading the public socket. □

Theorem 765.131 (MetaCIC critical-path L10 boundary consumer)**.** *Every consumer of the Real-completion socket in Definition 765.130 must consume the four L10 rows through the same packet rows* O, D, H, C, P, L*. Agreement on the Real-facing rows alone cannot fill the subject-reduction socket or collapse the boundary into quotient term equality.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10BoundaryConsumer`

Proof. Project the consumer route from Definition 765.130. The route already fixes the four L10 source rows as a boundary socket and keeps the obstruction and discharge rows visible.

Apply the status bridge obligation and the public export theorem. The L10 faces are read objectwise, while the public export surface remains the finite candidate-mediated route over $S, N, O, U, D, H, C, P, L$ rather than a closed metatheoretic theorem.

Thus a consumer can pass through the Real-completion socket only by preserving O, D, H, C, P, L. The subject-reduction socket is not filled by the Real-facing rows, and quotient term equality is not a packet coordinate. □

Theorem 765.132 (MetaCIC critical-path L10 status bridge obligation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any use of Theorem 765.128 as Phase Real and Completion exit evidence must pass through the Real L10 objectwise audit pullback of Theorem 38.471. The* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces each contribute*

their own local closurestatus, formalstatus, leantarget, and bridgestatus rows, read from their local closurestatus blocks and not from an aggregate route credential.

Consequently the MetaCIC four-object certificate can be consumed only through the objectwise status discipline of Theorem 38.833 and Theorem 38.488. It supplies no quotient stream equality, no selected host limit, no aggregate bridge row, and no $\mathsf{Real}^{\uparrow}$ *completeness claim beyond the four local face records.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10StatusBridgeObjectwiseObligation`

Proof. First read the MetaCIC certificate itself. Theorem 765.128 displays exactly four finite source rows attached to K: the dyadic budget, StreamName schedule, RegSeqRat readback, and Real terminal seal rows.

Next apply the Real L10 audit pullback. Theorem 38.471 replaces any aggregate reading of that displayed route by the product of the four local face records. Each face is therefore read through its own closurestatus, formalstatus, leantarget, and bridgestatus entries.

Now use Theorem 38.833. The four local face reports remain separate and cannot be fused into a new formalstatus token, route-level bridge row, or $\mathsf{Real}^{\uparrow}$ completeness report.

Finally apply Theorem 38.488. A Phase Real and Completion exit is accepted only when the four objectwise face records are presented, so quotient stream equality, selected host limits, aggregate bridge rows, and completed-real closure remain outside the MetaCIC L10 evidence. □

Theorem 765.133 (MetaCIC critical path L10 objectwise status consumer)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that records the Phase Real and Completion exit may read* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *only through the objectwise status bridge obligation and the displayed source ledger. The consumer receives four local face records, carried by the packet rows* O, D, H, C, P, L, *and no coordinate for host equality, quotient streams, selected limits, aggregate bridge status, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Theorem 765.132. It routes every use of the four-object exit certificate through the Real L10 objectwise audit pullback, so the four Real and Completion source faces are not read as one aggregate credential.

Next apply Theorem 765.132 in its objectwise form recorded above. The dyadic budget, finite stream schedule, regular-rational readback, and Real seal each contribute a separate local closurestatus, formalstatus, leantarget, and bridgestatus row.

Project the displayed source ledger from $K = (S, N, O, U, D, H, C, P, L)$. The rows O, D, H, C, P, L keep the obstruction, discharge socket, transport, replay, provenance, and local naming visible while the four source faces are consumed.

The excluded coordinates are absent from the packet and from the objectwise bridge surface. Thus the consumer cannot replace the four local face records by host equality, quotient streams, selected limits, an aggregate bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ObjectwiseStatusConsumer`

Theorem 765.134 (MetaCIC Phase Real four-face obligation surface)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$, *a Phase Real exit accepted through* K *is obligated to present four separate object rows: a* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-budget row, a* $\mathsf{StreamName}^{\uparrow}$ *schedule row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and a* $\mathsf{Real}^{\uparrow}$ *seal row. The rows are read as distinct* BHist, hsame, Cont, Pkg, *and* NameCert *surfaces, and no one of the four may be replaced by an aggregate MetaCIC route token or by a downstream Completion credential.*

Proof. Start with the four-object exit certificate of Theorem 765.128. It already identifies the accepted exit surface over K with the dyadic budget, StreamName schedule, RegSeqRat readback, and Real terminal seal rows, each carried by finite packet provenance.

Use the status bridge obligation of Theorem 765.132 to route that certificate through the Real L10 objectwise audit pullback. Theorem 38.471 then forces the displayed route to be read as the product of the four local face records, rather than as a single MetaCIC status row.

Finally apply Theorem 38.488. Since the Phase Real exit criterion accepts only the objectwise presentation, every accepted exit through K must expose the four named obligation rows separately. The resulting surface is exactly the root-unblock obligation surface for the current Phase Real route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealFourFaceObligationSurface`

Theorem 765.135 (MetaCIC four-face confluence budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, candidate-mediated confluence evidence and bounded-conversion evidence may consume the L10 source-readiness rows only as four separate finite faces:* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The budget is carried through the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows as the accepted packet.*

No candidate-mediated confluence read or bounded-conversion read can collapse these four faces into a subject-reduction certificate, quotient stream equality, selected host limit, countable-choice coordinate, or completed-real closure.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceConfluenceBudget`

Proof. Start with the four-object exit certificate, Definition 765.128. It displays exactly the dyadic budget, StreamName schedule, RegSeqRat readback, and Real terminal seal rows as the L10 exit surface.

Next use the four-face obligation surface, Definition 765.134, together with the source-obligation theorem Definition 765.165. These results force any Phase Real or Completion consumer to expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as separate finite source rows before confluence-decidability handoff or downstream readiness is consumed.

Now route candidate evidence through Definition 765.278. Candidate-mediated strong-normalization, confluence-facing, and bounded-decidability rows may be inspected only while the obstruction row O and discharge socket D remain visible in the same packet.

Finally apply Definition 765.280 and Definition 765.26. Bounded conversion may support candidate inspection, but the socket is filled only by a separate subject-reduction certificate. Therefore the four L10 faces remain source-readiness rows and cannot be collapsed into subject reduction, quotient equality, selected limits, countable choice, or completed-real closure. □

Theorem 765.136 (MetaCIC Phase Real status-product readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the Phase Real exit evidence through* K *is ready for downstream consumption only when the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces each supply their own local closurestatus, formalstatus, leantarget, and bridgestatus rows. These four reports are read independently from the local face chapters and are not inferred from the aggregate MetaCIC four-object certificate.*

Proof. First read the accepted packet through Theorem 765.132. It states that any use of the MetaCIC four-object certificate as Phase Real and Completion exit evidence must pass through the Real L10 objectwise audit pullback.

The audit pullback of Theorem 38.471 supplies the product reading: each of the four faces contributes its own closurestatus, formalstatus, leantarget, and bridgestatus row. This gives readiness only after all four local records have been displayed.

The noncollapse clauses now rule out aggregate substitution. Theorem 38.833 prevents a fused formalstatus or route-level bridge row, and Theorem 38.834 keeps the leantarget rows local to their four face chapters. Thus downstream consumers receive a four-entry status product, not a single MetaCIC route credential. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealStatusProductReadiness`

Theorem 765.137 (MetaCIC L10 four-face discharge readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, Phase Real and Completion exit evidence can be consumed through* K *only when each of the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *exposes its own local discharge-readiness row. The aggregate MetaCIC route may display the four-object exit certificate and the status-product readiness surface, but it cannot provide a missing face-local discharge row for any one of the four faces.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceDischargeReadiness`

Proof. First use Definition 765.136. It reads downstream readiness as a product of the four local face records and refuses an aggregate MetaCIC route credential as a substitute for those records.

Next apply face-local target locality through Definition 765.140. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ target and bridge rows remain attached to their own chapters, so a discharge-readiness row attached to one face cannot be moved to another face or supplied by the route as a whole.

Finally apply the four-face deficiency refusal of Definition 765.139. If any face lacks the local row required for consumption, the missing coordinate blocks the Phase Real and Completion exit read. Thus the MetaCIC route can package the four faces, but it cannot discharge a face whose local readiness row is absent. □

Theorem 765.138 (MetaCIC Phase Real exit refusal boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the Phase Real exit through* K *refuses quotient stream equality, selected host limits, host equality, trichotomy, open-cover compactness, countable choice, and completed-real closure. The only accepted exit data are the four finite object rows of Theorem 765.134 together with their local status-product rows of Theorem 765.136.*

Proof. Begin with Theorem 765.128. Its final exclusion list already keeps quotient stream equality, selected host limits, host real equality, trichotomy, open-cover compactness, countable choice, and completed-real closure outside the four-object exit certificate.

Next use Theorem 765.132. It says that the certificate can be consumed only through the Real L10 objectwise status discipline, and it supplies no quotient stream equality, no selected host limit, no aggregate bridge row, and no $\mathsf{Real}^{\uparrow}$ completeness claim beyond the four local face records.

The Real L10 audit theorems make the refusal stable under downstream reading. Theorem 38.833 prevents the missing coordinates from reappearing as a fused status row, while Theorem 38.488 accepts the exit only through the four objectwise face records.

Combining the four-face obligation surface with the status-product readiness theorem therefore leaves no admissible entry point for quotient equality, selected limits, host equality, trichotomy, open-cover compactness, countable choice, or completed-real closure. These principles remain outside the accepted Phase Real exit boundary through K. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealExitRefusalBoundary`

Theorem 765.139 (MetaCIC critical-path four-face deficiency refusal)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, suppose that at least one of the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *fails to provide the local closurestatus, formalstatus, leantarget, and bridgestatus row required by Theorem 38.488. Then the MetaCIC four-object route through* K *cannot be consumed as Phase Real and Completion exit evidence.*

In particular, the missing face cannot be replaced by an aggregate formalstatus row, an aggregate bridge row, quotient stream equality, a selected host limit, countable choice, or a $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First route the accepted packet through the MetaCIC status bridge. Theorem 765.132 sends every use of the four-object certificate to the Real L10 objectwise audit pullback, and Theorem 765.136 says that downstream readiness is obtained only from the four local face rows.

Now apply the Real-side blocker. Theorem 38.489 states that a deficient face in the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ product prevents the current Phase Real and Completion exit criterion from being satisfied. Since the MetaCIC route is consumed only through that product, the deficient face blocks consumption of the route as exit evidence.

Finally use the noncollapse theorems to rule out substitutes. Theorem 38.833 prevents an aggregate formalstatus token, aggregate bridge row, quotient stream, selected limit, countable-choice coordinate, or $\mathsf{Real}^{\uparrow}$ completeness report from filling the missing local status coordinate. Theorem 38.834 keeps leantarget and bridgestatus rows local to their four face chapters. Thus the contrapositive obstruction is route-level: without all four local face records, the MetaCIC four-object route supplies no admissible Phase Real and Completion exit evidence. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceDeficiencyRefusal`

Theorem 765.140 (MetaCIC critical-path L10 face target locality)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, each Phase Real face* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *must supply its own local leantarget and bridgestatus row when the packet is consumed as Phase Real and Completion exit evidence. These rows are read through the Real L10 objectwise audit pullback of Theorem 38.471; no aggregate MetaCIC route target, route-level bridge row, quotient stream equality, selected host limit, countable-choice coordinate, or* $\mathsf{Real}^{\uparrow}$ *completeness claim can replace any one of the four local rows.*

Proof. Start from the status bridge obligation. Theorem 765.132 says that any use of the accepted MetaCIC four-object certificate as Phase Real and Completion exit evidence must pass through the Real L10 objectwise audit pullback.

The pullback then fixes the reading of the target and bridge rows. By Theorem 38.471, the exit evidence is inspected as the product of the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not as a single route credential.

Apply the two Real-side noncollapse clauses. Theorem 38.834 keeps each leantarget row attached to its own face chapter, and Theorem 38.833 prevents a fused status or bridge row from replacing that face-local audit record.

Finally use Theorem 765.139. If any face-local leantarget or bridgestatus row is missing, the missing coordinate cannot be supplied by an aggregate MetaCIC route target, route-level bridge row, quotient stream equality, selected host limit, countable-choice coordinate, or $\mathsf{Real}^{\uparrow}$ completeness claim. Thus the target and bridge coordinates are objectwise obligations for the four displayed faces. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FaceLeantargetLocality`

Theorem 765.141 (MetaCIC L10 source-owner blocker)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as Phase Real and Completion exit evidence, every read through* K *preserves four separate source-owner rows for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The dyadic budget, stream schedule, regular-rational readback, and Real seal remain owned by their local face chapters and by the finite* BHist, hsame, Cont, Pkg, *and* NameCert *support rows attached to those faces.*

No aggregate MetaCIC route token, quotient stream equality, selected host limit, countable-choice coordinate, or completed-real closure can replace a missing source-owner row. In particular, a Phase Real or Completion consumer that omits any one owner row cannot repair the omission by citing the route as a whole.

Proof. First apply the four-object exit certificate. Theorem 765.128 identifies the exit surface with exactly the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows, so the route starts with four displayed owners rather than one aggregate source.

Next use the status-product readiness and face leantarget locality results, Theorem 765.146 and Theorem 765.140. Readiness is packaged as a product of local records, and each leantarget or bridge row remains attached to its own face chapter.

Finally apply the four-face deficiency refusal. Theorem 765.139 blocks consumption whenever a local face record is absent and refuses aggregate route credentials, quotient stream equality, selected limits, countable choice, and ambient Real completeness as substitutes. Therefore each Phase Real and Completion read must preserve the four source-owner rows separately. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10SourceOwnerBlocker`

Theorem 765.142 (MetaCIC Phase Real audit-clean floor)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$, *the packet can be consumed as Phase Real and Completion exit evidence only when the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are each read through their own local closurestatus record at least* scopedClosed *and their own local formalstatus record at least* auditClean. *The comparison is made face by face through the Real L10 audit pullback of Theorem 38.471 and the objectwise exit threshold equivalence of Theorem 38.494.*

No aggregate route status, aggregate bridge row, quotient stream equality, selected host limit, countable-choice coordinate, or $\mathsf{Real}^{\uparrow}$ *completeness claim can substitute for any one of those four local closure and audit-clean verification records.*

Proof. Start from the face-local reading. Theorem 765.140 says that the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ target and bridge rows are local obligations whenever K is consumed as Phase Real and Completion exit evidence.

Apply the Real L10 pullback. Theorem 38.471 reads the accepted route as the product of the four local face records, not as a single MetaCIC route status. Theorem 38.494 identifies the current exit threshold with local theory-closure rows and local audit-clean verification rows on exactly those four faces.

It remains to exclude substitutes. Theorem 765.139 blocks consumption when any required face-local record is absent. The same refusal rules out aggregate route status, aggregate bridge rows, quotient stream equality, selected host limits, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness as replacements for the missing face-local closure or audit-clean record. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealAuditCleanFloor`

Theorem 765.143 (MetaCIC four-face audit floor)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the complete MetaCIC consumer-side audit floor for the current Phase Real route is the conjunction of the status bridge obligation of Theorem 765.132, the audit-clean floor of Theorem 765.142, and the four-face deficiency refusal of Theorem 765.139. A downstream consumer may therefore accept the route only by displaying all four local face records at the stated closure and verification threshold, with the listed deficiencies still refused.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceAuditFloor`

Proof. First apply the status bridge obligation. Theorem 765.132 sends every use of the accepted MetaCIC four-object certificate to the Real L10 objectwise audit pullback and prevents the consumer from reading a route-level credential in place of local face data.

Next apply Theorem 765.142. It raises the local status product to the current Phase Real threshold: each of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ must contribute its own closurestatus record at least $\mathsf{scopedClosed}$ and its own formalstatus record at least $\mathsf{auditClean}$.

Finally apply Theorem 765.139. If any one of the four face records is missing or below the required local threshold, the route cannot be consumed as Phase Real and Completion exit evidence. Thus the three cited theorems form the complete consumer-side audit floor for the current MetaCIC route. □

Theorem 765.144 (MetaCIC L10 four-object locality)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *Phase Real and Completion exit consumer must read* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as four local records, each with its own finite* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, closurestatus, formalstatus, leantarget, and bridgestatus surface. None of the four local records can be substituted by an aggregate route target, quotient stream, selected host limit, countable-choice coordinate, route-level bridge row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

The locality requirement is exactly the consumer-side reading of the MetaCIC L10 route: the aggregate packet may carry the four-object exit certificate, but it never replaces the four displayed object records.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourObjectLocality`

Proof. Start with the L10 status bridge obligation of Theorem 765.132. It sends every accepted use of the four-object certificate through the Real L10 objectwise audit pullback, so the consumer reads the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal as separate local faces.

Next apply the four-face deficiency refusal of Theorem 765.139. If any one of the four local records is absent, the MetaCIC route cannot be consumed as Phase Real and Completion exit evidence. This already rules out replacement by a single aggregate route credential.

The face-target locality theorem, Theorem 765.140, fixes the target and bridge rows at their local faces. Thus a route-level leantarget or route-level bridge row cannot stand in for the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ local record.

Finally apply the audit-clean floor of Theorem 765.143. The consumer-side threshold is the four-face product itself. Quotient streams, selected host limits, countable-choice coordinates, and $\mathsf{Real}^{\uparrow}$ completeness claims have no coordinate in that product, so they cannot replace any of the four local object records. □

Theorem 765.145 (MetaCIC L10 four-face closure readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the packet can be consumed as current Phase Real and Completion readiness only when the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$

each expose a local closurestatus row at least scopedClosed *and a local formalstatus row at least* auditClean*. The MetaCIC aggregate route supplies the four-object exit surface, but it supplies neither quotient stream equality nor selected host limits nor a substitute status row for any missing face.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceClosureReadiness`

Proof. First use the four-object exit certificate. Theorem 765.128 identifies the current route through the dyadic budget, StreamName schedule, RegSeqRat readback, and Real terminal seal rows, each carried by finite BHist, hsame, Cont, Pkg, and NameCert support.

Next route that surface through the status bridge. Theorem 765.132 forces any Phase Real and Completion consumer to inspect the Real L10 objectwise audit pullback, and Theorem 765.136 reads that pullback as a four-entry product of local face records.

The local threshold is the audit-clean floor. Theorem 765.142 and Theorem 765.143 require each face to contribute its own closurestatus at least scopedClosed and its own formalstatus at least auditClean.

Finally apply the deficiency refusal. Theorem 765.139 blocks consumption when any one face is absent or below threshold. The same refusal excludes aggregate route status, quotient stream equality, selected host limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness as replacements for the missing local record. □

Theorem 765.146 (MetaCIC L10 four-object readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, suppose the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *each expose their own local closurestatus, formalstatus, leantarget, and bridgestatus rows at the Phase Real and Completion exit threshold. Then* K *may export a Phase Real four-object readiness row whose content is the product of those four local records together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows of the accepted route.*

The readiness row is not an aggregate credential. It does not replace any face-local closure row, formalstatus row, leantarget row, bridgestatus row, quotient stream boundary, selected host limit, countable-choice coordinate, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourObjectReady`

Proof. Start with the four-face closure-readiness theorem, Definition 765.145. It requires each of the four faces to expose its own local closurestatus and formalstatus row before the MetaCIC route is consumable as current Phase Real and Completion readiness.

Next use face-local target locality, Definition 765.140. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ leantarget and bridge rows remain attached to their own chapters, so the accepted packet cannot fuse them into a route-level target.

Now apply the consumer threshold theorem, Definition 765.150. It identifies the current exit threshold with the facewise conjunction of the four local records and the displayed finite support rows.

Finally package that conjunction as the exported readiness row. Because the construction uses the product of four local records, Definition 765.139 still blocks consumption when any face is absent or below threshold, and the listed aggregate, quotient, choice, selected-limit, and completeness substitutes remain outside the accepted route. □

Theorem 765.147 (MetaCIC L10 consumer-readiness locality)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any downstream MetaCIC consumer of the Phase Real route*

must read DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *through their four local closurestatus, formalstatus, leantarget, and bridgestatus records. The four-object route may package those local records as readiness evidence, but it cannot replace any one of them by route-level readiness, quotient stream equality, selected limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ConsumerReadinessLocality`

Proof. Start with Definition 765.146. Its readiness row is a product of the four local records, so a consumer obtains readiness only after the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ records have each been read.

Apply Definition 765.145. It fixes the local closurestatus and formalstatus rows as facewise obligations rather than as a single route-level threshold.

Finally use Definition 765.139. If any face-local record is absent, the downstream consumer cannot repair the gap with aggregate route readiness, quotient stream equality, selected limits, countable choice, or ambient Real$^{\uparrow}$ completeness. Thus consumer readiness remains local to the four displayed faces. □

Theorem 765.148 (MetaCIC L10 four-face obligation-closure package)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as Phase Real and Completion evidence, the accepted packet must present the four objectwise obligation rows for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *together with their local closurestatus and formalstatus support. The aggregate MetaCIC route may package the four faces through its displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows, but it cannot replace any face or infer any local support row from the aggregate route alone.*

The package exports only the facewise obligation surface. It supplies no quotient stream equality, selected host limit, countable-choice coordinate, aggregate bridge row, route-level formalstatus row, or ambient Real$^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceObligationClosurePackage`

Proof. First use the four-object exit certificate of Theorem 765.128. It identifies the Phase Real and Completion route through the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows.

Next apply the status bridge obligation of Theorem 765.132. That theorem sends every accepted use of the aggregate certificate through the Real L10 objectwise audit pullback, so each face must bring its own closurestatus and formalstatus support.

The four-face obligation surface of Theorem 765.134 then makes the obligation row product explicit. The aggregate route can arrange the four faces, but it cannot turn them into one replacement coordinate.

Finally use the audit floor of Theorem 765.143 together with the deficiency refusal of Theorem 765.139. If any face-local support row is missing or below the required threshold, the route is not consumable as current Phase Real and Completion evidence. The same refusal excludes quotient stream equality, selected host limits, countable choice, aggregate bridge rows, route-level formalstatus, and ambient Real completeness as substitutes. □

Theorem 765.149 (MetaCIC Phase Real consumer minimality)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as Phase Real and Completion exit evidence, every downstream consumer must retain all four Phase Real face rows* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$,

RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows attached to those faces. Deleting or replacing any one of those face rows or support rows prevents K from being consumed as current phase exit evidence.*

No aggregate MetaCIC route token, aggregate formalstatus row, aggregate bridge row, quotient stream equality, selected host limit, countable-choice coordinate, metric-completeness row, or Real$^{\uparrow}$ *completeness claim can repair the deletion.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealConsumerMinimality`

Proof. First route the consumer through the status bridge. Theorem 765.132 says that every use of the MetaCIC four-object certificate as Phase Real and Completion exit evidence passes through the Real L10 objectwise audit pullback of Theorem 38.471.

The audit floor then fixes the local threshold. Theorem 765.143 combines the status bridge, Theorem 765.142, and the four-face deficiency refusal into the consumer-side audit floor. Theorem 38.494 reads that floor objectwise, so every local face record is part of the accepted exit threshold.

Now remove one displayed face row or one displayed support row. If the removed row belongs to the DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, or Real$^{\uparrow}$ face product, Theorem 765.139 blocks consumption of the route as Phase Real and Completion exit evidence.

If the consumer proposes an aggregate substitute, the Real L10 noncollapse clauses reject it. Theorem 38.833 prevents aggregate status, bridge, quotient, selected-limit, countable-choice, metric-completeness, and Real$^{\uparrow}$ completeness data from filling a missing face coordinate.

It remains to account for the finite support rows. The four-object certificate of Theorem 765.128 carries the four faces through the displayed BHist, hsame, Cont, Pkg, and NameCert surface, and Theorem 765.134 makes that surface part of the Phase Real obligation. Thus deleting or replacing any displayed support row removes the finite route by which the objectwise audit pullback reads the face, and no aggregate credential can repair the loss. □

Theorem 765.150 (MetaCIC L10 four-face consumer threshold)**.** *For every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet $K = (S, N, O, U, D, H, C, P, L)$ consumed as current Phase Real and Completion exit evidence, the consumer threshold is the facewise conjunction of the four local records for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$. *Each face must contribute its own local closurestatus row at the scoped object boundary and its own audit-clean formalstatus row, together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows attached to that face.*

The threshold is not satisfied by the aggregate MetaCIC route token, by a route-level formalstatus row, by a route-level bridge row, by quotient stream equality, by selected host limits, by countable choice, or by an ambient Real$^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceConsumerThreshold`

Proof. Start with Theorem 765.149. It says that every downstream consumer must retain all four Phase Real face rows and the finite support rows attached to those faces.

Apply Theorem 765.143. The consumer-side audit floor is obtained only by displaying all four local face records at the stated closure and verification threshold. In particular, each of DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ must supply its own closurestatus and audit-clean formalstatus data through the Real L10 objectwise audit pullback.

Finally use Theorem 765.139. If any one face lacks the required local record, the accepted packet cannot be consumed as Phase Real and Completion exit evidence. The same refusal excludes aggregate MetaCIC route tokens, route-level status, route-level bridge data, quotient stream

equality, selected host limits, countable choice, and ambient Real completeness as replacements for the missing facewise threshold row. □

Theorem 765.151 (MetaCIC critical-path L10 four-face obligation noncollapse)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as Phase Real and Completion exit evidence, the four local obligation rows for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *remain separate. No Phase Real consumer may replace those four rows by an aggregate MetaCIC route token, aggregate formalstatus row, quotient stream equality, selected host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply the four-object exit certificate, Definition 765.128. It exposes the Phase Real route as the displayed product of the four local faces, each with its own finite BHist, hsame, Cont, Pkg, and NameCert support.

Use the status bridge obligation, Definition 765.132. This routes exit evidence through the Real L10 objectwise audit pullback rather than through a single aggregate MetaCIC credential.

Now apply the four-face deficiency refusal and the consumer threshold theorem, Definition 765.139 and Definition 765.150. They make each local face row a required coordinate of consumption and reject the listed aggregate substitutes whenever a coordinate is missing.

Thus the accepted packet can be consumed only by retaining the four local obligation rows separately. The aggregate route token, aggregate formalstatus row, quotient stream equality, selected host limit, and ambient $\mathsf{Real}^{\uparrow}$ completeness claim have no row from which to replace any of them. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceObligationNoncollapse`

Theorem 765.152 (MetaCIC L10 readiness local minimality)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as current Phase Real evidence, readiness is locally minimal at the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Each face must retain its own local record together with the finite* BHist, hsame, Cont, Pkg, *and* NameCert *support rows attached to that face. Deleting any one local face row, or deleting any one displayed support row needed to read that face, blocks consumption of* K *as current Phase Real evidence.*

No aggregate MetaCIC route token, route-level formalstatus row, route-level bridge row, quotient stream equality, selected host limit, countable-choice coordinate, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim can replace the deleted local row.*

Proof. First apply the four-face consumer threshold of Theorem 765.150. It identifies current consumption with the facewise conjunction of the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ records, each read with its finite BHist, hsame, Cont, Pkg, and NameCert support.

Next use face-target locality from Theorem 765.140. The local target and bridge rows remain attached to their own face chapters, so the aggregate MetaCIC route cannot move one face record onto another or supply a route-level substitute.

The audit floor then fixes the threshold. Theorem 765.143 says that the consumer-side floor is the displayed four-face product at the local closure and verification level. Removing a face row or a support row removes one coordinate of that product.

Finally apply the deficiency refusal of Theorem 765.139. A missing face or missing support coordinate blocks consumption as Phase Real and Completion exit evidence, and the same theorem refuses aggregate route tokens, route-level status, bridge rows, quotient equality, selected limits, countable choice, metric-completeness data, and ambient Real completeness as replacements. Hence the current readiness surface is locally minimal. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ReadinessLocalMinimality`

Theorem 765.153 (MetaCIC L10 four-face bridge deferral)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed as current Phase Real and Completion exit evidence, bridge closure is deferred face by face to the local* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *records. The aggregate four-object route may expose the finite exit surface and the consumer threshold of Theorem 765.150, but it cannot supply a bridgeChecked row for any face whose local closurestatus block has not supplied that bridge record.*

Thus a bridge consumer must inspect the four local bridgestatus rows separately. No aggregate MetaCIC route token, route-level bridge row, quotient stream equality, selected host limit, countable-choice coordinate, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim can close a missing face bridge.*

Proof. First read the local target and bridge rows. Theorem 765.140 makes each of the four Phase Real faces responsible for its own leantarget and bridgestatus row whenever K is consumed as Phase Real and Completion exit evidence.

Next apply the consumer threshold theorem. Theorem 765.150 gives the current audit-clean threshold as a facewise product, not as an aggregate MetaCIC status. Therefore the four-object route can expose the threshold only by presenting the local face records.

Now use Theorem 38.834 and Theorem 38.833. These noncollapse clauses keep leantarget, formalstatus, and bridge rows attached to their own face chapters and prevent a fused route credential from replacing any one of them.

Finally apply Theorem 765.139. A missing local bridge record is a deficient face record for bridge consumption, and no aggregate MetaCIC route token, route-level bridge row, quotient stream equality, selected host limit, countable choice, metric-completeness row, or ambient Real completeness claim can repair it. Bridge closure is therefore postponed to the four local face records separately. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10FourFaceBridgeDeferral`

Theorem 765.154 (MetaCIC L10 exit scope seal)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *consumed by candidate-mediated strong-normalization, confluence, or subject-reduction consumers, the Phase Real boundary read through* K *has exactly the four L10 object faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The packet may expose the four-object exit certificate, dependency budget, status bridge obligation, audit-clean floor, and facewise bridge deferral for those four faces, but it supplies no fifth object face and no aggregate substitute for any one local face record.

Proof. Start with the four-object exit certificate, Definition 765.128. It identifies the exit surface read through K with the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows.

Use the dependency budget of Definition 765.129. That budget orders the four faces as finite data read through the packet, so candidate-mediated strong-normalization, confluence, and subject-reduction consumers receive the same displayed Phase Real boundary rather than a separate MetaCIC object.

Next apply the status bridge obligation of Definition 765.132 and the audit-clean floor of Definition 765.142. These rows make the consumer threshold facewise: every cited readiness record belongs to one of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$.

Finally apply the bridge deferral theorem Definition 765.153. Since bridge closure is postponed to the four local faces separately, the aggregate packet cannot add a fifth face or collapse the four local records into a route-level credential. The L10 exit scope is therefore exactly the displayed four-object boundary. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ExitScopeSeal`

Definition 765.155 (MetaCIC critical-path open-phase source ledger)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the open-phase source ledger is the finite packet whose local rows are the subject row* S*, normal-form row* N*, obstruction row* O*, unblock row* U*, discharge row* D*, handoff row* H*, continuation row* C*, package row* P*, and provenance ledger row* L*, together with the following four displayed Phase Real source rows:*

$$\begin{aligned} \mathsf{DyadicRatCore}^{\uparrow} &: \textit{finite dyadic observation budget}, \\ \mathsf{StreamName}^{\uparrow} &: \textit{finite source schedule}, \\ \mathsf{RegSeqRat}^{\uparrow} &: \textit{regular-rational readback route}, \\ \mathsf{Real}^{\uparrow} &: \textit{terminal Real seal handoff}. \end{aligned}$$

The four rows are grounded respectively in chapter 291, chapter 126, chapter 301, and chapter 38. Each displayed row is read through finite BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance attached to* K*. The ledger exports no quotient real, host equality, selected limit, countable choice, trichotomy, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 (definition) `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseSourceLedger`

Theorem 765.156 (MetaCIC critical-path open-phase source closure)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the open-phase source ledger of Definition 765.155 is closed under the current Phase Real and Completion handoff surface: every admitted source read factors through the displayed packet rows* $S, N, O, U, D, H, C, P, L$ *and one of the four rows* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$*.*

Proof. First expose the packet rows. Theorems 765.84, 765.85, and 765.86 make $S, N, O, U, D, H, C, P, L$ the finite consumer surface of the accepted packet.

Next attach the four Phase Real rows. Theorem 765.118 requires the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before the route can be entered, while Theorem 765.128 identifies those four rows as the current exit certificate.

Finally apply the dependency diamond. Theorem 765.121 recovers exactly the four displayed source projections from an admitted public entry, and Theorem 765.122 keeps their owners separate. Hence every admitted source read is already a row of the ledger. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseSourceClosure`

Theorem 765.157 (MetaCIC critical-path open-phase source ledger discharge budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any candidate-mediated normalization, residual confluence, or decidability discharge read that is allowed to interact with the current open phase must first display the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *through the ledger row* L*. After those four faces are displayed, the read may inspect only the MetaCIC candidate row* N*, obstruction row* O*, handoff row* H*, and discharge-socket row* D *together with the structural rows of* K*. The budget exports no closed subject reduction, global confluence, full beta-conversion decidability, quotient term equality, selected normal form, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the packet rows of K. Definition 765.155 makes L the ledger row through which the four current L10 source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are displayed.

Next read the candidate-mediated budget. Theorems 765.84, 765.85, and 765.86 expose only the candidate, obstruction, unblock, discharge, handoff, continuation, package, and ledger rows of the accepted packet.

The open-phase source closure then exhausts the permitted source interaction. Theorem 765.156 requires every admitted source read to factor through one of the four displayed faces, while Theorem 765.158 blocks the listed outside principles.

Thus a normalization, residual confluence, or decidability discharge can use the current open phase only by first displaying the four source faces through L and then staying inside the candidate, obstruction, handoff, discharge-socket, and structural rows already present in K. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseSourceLedgerDischargeBudget`

Theorem 765.158 (MetaCIC critical-path open-phase source nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, no claim outside the open-phase source ledger can be exported by the current Phase Real and Completion route. In particular, the route exports no quotient real, host equality, selected limit, countable choice, trichotomy, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the frontier filter. Theorem 765.119 admits Completion-facing claims only when they are backed by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The exit certificate then removes the listed substitutes. Theorem 765.128 supplies no additional coordinate for quotient stream equality, selected host limits, host real equality, trichotomy, open-cover compactness, countable choice, or completed-real closure.

Finally use the Phase Real exit refusal boundary. Theorem 765.138 states that the accepted exit data are only the four finite object rows together with their local status product. Since Definition 765.155 records exactly those rows over $S, N, O, U, D, H, C, P, L$, every listed outside principle has no export path through the ledger. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseSourceNonescape`

Theorem 765.159 (MetaCIC critical-path open-phase consumer totality)**.** *Every current Phase Real and Completion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet reads the open-phase source ledger as its whole source surface. The consumer may inspect the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite dyadic budget, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal row through their local chapters, but it receives no fifth roadmap object and no completion principle beyond those rows.*

Proof. First use the current readiness surface. Theorem 765.120 identifies the Phase Real and Completion route with the finite dependency budget, the L10 entry budget synchronization, and the Real completion frontier filter.

Then read the objectwise status discipline. Theorems 765.134, 765.136, and 765.140 require consumers to use the four local face records rather than an aggregate route credential.

It remains to rule out omitted consumers. Theorem 765.139 blocks the route when any one of the four face rows is absent, while Theorem 765.156 says that every admitted source read is already present in the ledger. Thus the ledger is total for current Phase Real and Completion consumers, and no extra roadmap object or completion principle is exported. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseConsumerTotality`

Theorem 765.160 (MetaCIC critical-path open-phase exact-boundary handoff)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, if the open-phase source ledger is consumed through the four displayed faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, then every MetaCIC-candidate read, candidate-mediated normalization read, residual-confluence read, and discharge read in the resulting handoff factors through the same four-face exact boundary identified by Theorem 1120.6. The handoff exports no quotient stream equality, selected host limit, countable-choice coordinate, aggregate MetaCIC credential,* $\mathsf{Real}^{\uparrow}$ *completeness claim, closed subject reduction, full confluence coordinate, or full consistency coordinate.*

Proof. Begin with the finite ledger rows. Definition 765.155 lists the packet rows $S, N, O, U, D, H, C, P, L$ together with exactly the four Phase Real source rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next exhaust the possible consumers. Theorem 765.156 says that every admitted source read factors through one of those rows, and Theorem 765.159 says that a current Phase Real and Completion consumer receives no fifth roadmap object and no completion principle beyond them.

The candidate-mediated part is still a source read. Theorems 765.84, 765.85, and 765.86 expose only the candidate, obstruction, unblock, discharge, handoff, continuation, package, and ledger rows of the accepted packet, so candidate normalization and residual-confluence reads cannot bypass the displayed source faces.

Finally identify the boundary. Theorem 765.125 matches the four open-phase prefixes with the exact-boundary ordering, and Theorem 1120.6 recognizes the same $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces as the accepted Real-completion exact boundary. The negative export list follows because neither the ledger exhaustion nor the four-face exactness theorem supplies any additional quotient, host-limit, choice, aggregate-credential, completeness, subject-reduction, confluence, or consistency coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseExactBoundaryHandoff`

Theorem 765.161 (MetaCIC critical-path open-phase frontier)**.** *Every MetaCIC candidate-discharge read used by the current open Phase Real and Completion route must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source ledger before it can be consumed as an L10 readiness row. Thus candidate-mediated normalization, confluence, bounded conversion, and discharge-socket evidence remain downstream of the open-phase source ledger and cannot replace any one of the four L10 source faces.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathOpenPhaseFrontier`

Proof. First read the source ledger. Definition 765.155 displays the packet rows together with exactly the four Phase Real source rows, and Theorem 765.156 says that every admitted source read factors through that ledger.

Next apply the Phase Real source obligations. Theorem 765.165 requires $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to be exposed before any discharge socket, confluence-decidability handoff, obstruction boundary, or downstream readiness boundary is consumed.

Now route candidate evidence through the existing candidate frontier. Theorems 765.278, 765.281, and 765.284 keep candidate-mediated normalization, confluence, bounded-conversion, and L10 readiness reads inside the same finite packet while the obstruction row and discharge socket remain visible.

Finally use the L10 ordering row. Theorem 765.294 places the Real-and-Completion route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the normal-form endpoint

or discharge socket can be consumed. Therefore candidate-discharge reads may serve the current open phase only after the displayed source ledger has been read, and no candidate row becomes a substitute for the four source faces. □

Theorem 765.162 (MetaCIC critical-path L10 source-ledger nonescape)**.** *Any MetaCIC candidate-discharge read consumed by the current Phase Real and Completion route must retain the four source rows* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as source-ledger rows. Candidate-mediated normalization, confluence, bounded-conversion, and discharge-socket evidence may be routed only as downstream readiness data; it cannot discharge, replace, or identify any one of the four source rows with a candidate row.*

Proof. Apply Definition 765.161. It already orders every candidate-discharge read after the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source faces.

Now read the source obligations and L10 ordering rows named in that proof. Theorem 765.165 exposes the four source faces before discharge-socket or confluence-readiness evidence is consumed, and Theorem 765.294 keeps the same four rows before the normal-form endpoint.

The finite ledger has no remaining source coordinate. Candidate rows can therefore occupy only the downstream readiness slots already named by the open-frontier theorem, so none of them can discharge or replace $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10SourceLedgerNonescape`

Theorem 765.163 (MetaCIC open-frontier source exhaustion)**.** *Any candidate-mediated strong-normalization, confluence, bounded-conversion, or discharge-socket read serving the open Phase Real and Completion route is exhausted by the displayed source ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the obstruction row, and the finite candidate budget. No normal-form endpoint or discharge socket may replace one of those four L10 source faces.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierSourceExhaustion`

Proof. First apply the open-phase frontier theorem, Definition 765.161. It already forces candidate-mediated evidence to remain downstream of the four displayed L10 source faces.

Next read the source ledger and closure rows. Definition 765.155 displays the four source faces together with the packet rows, and Theorem 765.156 says that admitted source reads factor through that ledger.

Now use the Phase Real source obligations and candidate-discharge coverage. Theorems 765.165, 765.278, and 765.281 keep the obstruction row, discharge socket, and finite candidate budget visible while the candidate-mediated evidence is consumed.

Finally apply the L10 ordering row, Definition 765.294. Since the normal-form endpoint and discharge socket are read only after the displayed source faces, neither can serve as a replacement source for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$. □

Theorem 765.164 (MetaCIC parallel-diamond frontier seed)**.** *Any parallel-diamond, residual-diamond, confluence, or bounded-conversion read consumed by the current Phase Real and Completion route is a downstream readiness row only after the packet displays* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the obstruction row, and the finite candidate budget. Such a read cannot identify any diamond row with one of the four L10 source faces.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondFrontierSeed`

Proof. First apply the open-phase frontier and source-ledger nonescape, Definition 765.161 and Definition 765.162. These results force every candidate-mediated read to remain downstream of DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Next use source exhaustion, Definition 765.163. It keeps the obstruction row and finite candidate budget visible while the Phase Real and Completion route consumes normalization, confluence, bounded-conversion, or discharge-socket evidence.

Finally route the diamond-facing data through the local candidate-discharge coverage theorems. Theorems 765.278, 765.281, and 765.284, together with the L10 ordering row Definition 765.294, keep parallel-diamond and residual-diamond evidence in the downstream readiness slot. Therefore no diamond row can be read as, or identified with, one of the four L10 source faces. □

Theorem 765.165 (MetaCICCriticalPath Phase Real source obligations)**.** *Every Phase Real* MetaCICCriticalPath$^{\uparrow}$ *read must expose the four source rows* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *before any discharge socket, confluence-decidability handoff, obstruction boundary, or downstream readiness boundary is consumed. The read is therefore budgeted by the displayed packet rows, finite observation budget, source schedule, regular-rational readback, Real seal,* BHist *history rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows; it cannot start from a socket row, a confluence theorem, a decidability procedure, closed subject reduction, full consistency, quotient stream equality, a selected limit, or host equality.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealSourceObligations`

Proof. First read the Phase Real dependency budget. Theorems 765.108, 765.109, and 765.110 put the dyadic finite-observation row before the StreamName schedule, the RegSeqRat readback, and the Real seal.

Next use the four-object exit certificate. Theorems 765.128, 765.134, and 765.138 identify the four Phase Real source faces and refuse external completion, closed consistency, and off-ledger theorem rows.

Then route the read through the open source ledger. Definition 765.155 lists the packet rows and the four Phase Real source rows, while Theorems 765.156, 765.159, and 765.160 keep those rows as the whole visible source surface.

Finally attach the confluence-decidability budget. Theorem 765.192 and Theorem 765.171 allow confluence-facing and decidability-facing consumers only after the ledger and socket rows have been displayed. This matches the existing Lean-facing packet names for discharge socket coverage, transport row stability, kernel frontier exposure, and downstream readiness boundary: each consumes displayed BEDC rows rather than supplying a hidden source before DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ have been exposed. □

Theorem 765.166 (MetaCICCriticalPath formal-target surface)**.** *A* MetaCICCriticalPath$^{\uparrow}$ *formal target for the Phase Real route may name the four source rows only as separate local formal targets:* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *each retain their own closurestatus, formalstatus, and leantarget records. The aggregate MetaCIC route may cite those rows as the source surface of the Phase Real packet, but it cannot infer verification from paper closure and cannot infer paper closure from verification.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFormalTargetSurface`

Proof. Apply Definition 765.165. It forces every Phase Real read to expose the dyadic finite-observation row, stream schedule, regular-rational readback, and Real seal before socket, confluence, obstruction, or readiness consumers can use the packet.

Now read the four-object exit certificate through Theorems 765.128, 765.134, and 765.138. These references identify the four named source faces as the boundary objects of the route rather than as fields whose statuses are absorbed into $\mathsf{MetaCICCriticalPath}^{\uparrow}$.

Finally separate the two ledgers. The Phase Real source theorem supplies paper-side routing, while the individual $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ chapters retain their own verification records. Hence the aggregate route can depend on those local records without converting a closure grade into a formal grade or a formal grade into a closure grade. □

Theorem 765.167 (MetaCIC critical-path Phase Real root-unblock admission)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the packet is admissible as a Phase Real root-unblock route exactly when it displays the nine packet rows* $S, N, O, U, D, H, C, P, L$ *together with the four objectwise face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *at the local closure and formal thresholds required by Theorem 765.143. The admitted route is read through the open-phase source ledger of Definition 765.155; no aggregate MetaCIC credential may replace any one of the four face records.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealRootUnblockAdmission`

Proof. For admission from displayed data, start with the packet rows $S, N, O, U, D, H, C, P, L$. Theorem 765.128 attaches the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows to the same accepted packet, and Theorem 765.143 raises those four rows to the local threshold required by the current Phase Real route.

The open-phase ledger then supplies the route surface. Theorem 765.156 says that every admitted source read factors through the displayed packet rows and one of the four named face rows, so the displayed data are sufficient for the root-unblock admission interface.

Conversely, suppose that the packet is admitted as a Phase Real root-unblock route. Theorem 765.134 forces the four object rows to be presented separately, and Theorem 765.136 requires their local status records. The audit floor of Theorem 765.143 then gives exactly the stated threshold, while Theorem 765.139 rules out admission with a missing or substituted face. □

Theorem 765.168 (MetaCIC critical-path Phase Real root-unblock soundness)**.** *Any route admitted by Theorem 765.167 exports only the open-phase source ledger of Definition 765.155. It exports no quotient stream equality, no selected host limit, no countable choice, no trichotomy, no open-cover compactness, no* $\mathsf{Real}^{\uparrow}$ *completeness, no closed subject reduction, and no full consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealRootUnblockSoundness`

Proof. Begin with the admission theorem. It places every admitted route inside the displayed packet rows and the four objectwise face records at the audit floor, with the open-phase source ledger as the route surface.

Apply Theorem 765.158. That theorem says that no claim outside the ledger can be exported by the current Phase Real and Completion route, including quotient real data, host equality, selected limits, countable choice, trichotomy, open-cover compactness, and $\mathsf{Real}^{\uparrow}$ completeness.

The remaining exclusions are already part of the finite route boundary. Theorems 765.138 and 765.110 keep the route at the four finite source rows and refuse closed subject reduction, full consistency, hidden completion selectors, and theorem rows outside the displayed packet. Hence an admitted root-unblock route exports only the open-phase ledger. □

Theorem 765.169 (MetaCIC critical-path Phase Real root-unblock completeness)**.** *Every current Phase Real or Completion consumer that displays the packet rows $S, N, O, U, D, H, C, P, L$ together with the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *at the threshold of Theorem 765.143 factors through Definition 765.155. Such a consumer receives no fifth source row and no completion principle beyond the displayed four-face ledger.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPhaseRealRootUnblockCompleteness`

Proof. First read the consumer through the audit floor. The displayed packet rows and four local face records satisfy the same threshold used by Theorem 765.167, so the consumer is an admitted Phase Real root-unblock consumer.

Now apply totality of the open-phase ledger. Theorem 765.159 states that every current Phase Real and Completion consumer of an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet reads the ledger as its whole source surface.

Finally use nonescape to close the factorization. Theorem 765.158 prevents any outside completion principle from being exported, and Theorem 765.156 identifies every admitted source read with one of the packet rows or one of the four displayed face rows. Therefore the consumer factors through the ledger and receives no fifth source row. □

Theorem 765.170 (MetaCIC critical-path four-face socket separation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet $K = (S, N, O, U, D, H, C, P, L)$, Phase Real root-unblock admission separates the four object records from the discharge socket. The route requires separate face records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; the discharge socket D cannot replace any one of those records. Conversely, displaying all four face records supplies no value for D and does not promote the route to closed subject reduction or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceSocketSeparation`

Proof. First use Theorem 765.167. It identifies admission with the packet rows $S, N, O, U, D, H, C, P, L$ together with the four objectwise face records at the local threshold, and it explicitly refuses an aggregate MetaCIC credential as a replacement for any face.

Now remove or substitute one face record. Theorem 765.139 blocks the route whenever a required $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ face is missing or supplied only by an aggregate route credential. Since D is one packet coordinate rather than a face-local carrier, classifier, status, target, or bridge row, the same refusal prevents the discharge socket from serving as any one of the four records.

It remains to separate the reverse direction. Definition 765.155 and Theorems 765.156 and 765.158 keep every admitted source read inside the displayed packet rows and four face rows, with no export path to a further discharge witness.

Finally apply Theorem 765.168. That theorem excludes closed subject reduction and full consistency from every admitted Phase Real root-unblock route. Hence the four face records and the discharge socket are distinct obligations, and neither side fills or promotes the other. □

Theorem 765.171 (MetaCIC critical-path confluence-decidability handoff exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every confluence-facing or decidability-facing downstream read of U factors through the displayed rows O, D, H, C, P, L together with the finite packet rows S, N, U. The handoff exports no arbitrary

Church–Rosser theorem, no global beta-conversion decision procedure, no closed subject reduction, and no full closed consistency.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityHandoffExhaustion`

Proof. Start with the socket theorem. By Definition 765.61, every confluence-facing or decidability-facing use of the handoff row is mediated by the obstruction, discharge socket, transport, route, provenance, and local naming rows O, D, H, C, P, L.

Next exhaust the packet route. The handoff exhaustion theorem Definition 765.14 says that the finite route has only the rows $S, N, O, U, D, H, C, P, L$ and contains no coordinate for closed subject reduction, full closed consistency, global beta-conversion decidability, or quotient term equality.

The local socket boundary supplies the negative side. Apply Definition 765.57 to keep D local to H, C, P, L and separate from any hidden discharge, then apply Definition 765.62 to keep the obstruction row visible along normalization-to-normal-form reads.

Therefore a downstream read of U has only the displayed finite data available: O, D, H, C, P, L for the socket-mediated handoff and S, N, U for the remaining packet coordinates. Since no further theorem-export coordinate is present, the four excluded metatheoretic claims remain outside the handoff. □

Theorem 765.172 (MetaCIC critical-path candidate-mediated SN boundary)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization-facing read of the handoff row factors through the displayed packet rows $S, N, O, U, D, H, C, P, L$ *together with the four Phase Real faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *recorded by Definition 765.155. The read exports no arbitrary strong-normalization theorem, no Church–Rosser theorem, no beta-conversion decision procedure, no quotient term equality, no host real equality, no selected limit, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNBoundary`

Proof. First apply the confluence-decidability handoff exhaustion theorem, Theorem 765.171. It bounds every confluence-facing or decidability-facing use of the handoff row by the mediated rows O, D, H, C, P, L and the remaining packet coordinates S, N, U.

Next attach the current Phase Real source surface. The open-phase source closure theorem, Theorem 765.156, says that admitted source reads factor through the displayed packet rows and one of the four rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$. The open-phase consumer totality theorem, Theorem 765.159, says that current Phase Real and Completion consumers receive the ledger as their whole source surface.

Finally use the nonescape side of the ledger. Theorem 765.168 and Theorem 765.158 exclude quotient real data, host equality, selected limits, countable choice, trichotomy, open-cover compactness, $\mathsf{Real}^{\uparrow}$ completeness, closed subject reduction, and full consistency from the admitted route. Since a candidate-mediated strong-normalization-facing read has no additional coordinate beyond the same packet rows and four ledger faces, the listed SN, Church–Rosser, beta-conversion, quotient, host-equality, selected-limit, and completeness claims remain outside the handoff. □

Theorem 765.173 (MetaCIC candidate-mediated SN four-face audit precondition)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization-facing read that is also consumed as current Phase $\mathsf{Real}^{\uparrow}$ *and Completion evidence must first expose the four local face records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Each face must carry its own scoped-closure row and its own audit-clean verification row, and the shared support may use only* BHist, hsame, Cont, Pkg, *and* NameCert *data. The candidate-mediated SN row supplies no aggregate formalstatus, no aggregate leantarget, no aggregate bridge row, no quotient stream equality, no selected limit, no countable-choice row, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the SN-facing route. The candidate-mediated boundary theorem, Theorem 765.172, sends every such read through the finite packet rows $S, N, O, U, D, H, C, P, L$ and the four Phase Real faces listed in Definition 765.155.

Next read those faces as the source surface for current Phase Real and Completion consumption. Definition 765.155, Theorem 765.156, and Theorem 765.159 restrict the consumer to the displayed packet rows and the four local faces; no fifth source row or hidden discharge coordinate is introduced.

Now apply the current exit threshold. Theorem 38.796 identifies the consumed evidence with the objectwise product of the four local threshold records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the local $\mathsf{Real}^{\uparrow}$ face, supported only by BHist, hsame, Cont, Pkg, and NameCert.

Finally use the StreamName audit readback. Theorem 126.509 reads the same threshold through face-local audit rows rather than through a route-level credential. Therefore a candidate-mediated SN read that is consumed by the current Phase Real and Completion surface must display the four separate scoped-closure and audit-clean rows, and the aggregate status, leantarget, bridge, quotient, choice, metric-completeness, selected-limit, and $\mathsf{Real}^{\uparrow}$ completeness data remain outside the route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNFourFaceAuditPrecondition`

Theorem 765.174 (MetaCIC four-face discharge precondition strengthening)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization-facing read that satisfies the four-face audit precondition can be consumed by the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion surface only while the separate discharge socket row* D *remains visible as an unmet row distinct from the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The four-face audit product cannot be read as a discharge of* D, *closed subject reduction, full consistency, global beta-conversion decidability, quotient term equality, host equality, a selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceDischargePreconditionStrengthenin`

Proof. Start with the four-face audit precondition, Theorem 765.173. It exposes the four local scoped-closure and audit-clean face records as the only current Phase $\mathsf{Real}^{\uparrow}$ and Completion source surface supplied by the candidate-mediated SN read.

Next apply the candidate route discharge refusal theorem, Theorem 765.175. That result keeps D as the displayed discharge socket coordinate and denies any witness that fills it, even after the four audit faces are present.

Finally use the candidate SN socket nonescape theorem, Theorem 765.176. It prevents downstream consumers from absorbing D into the four-face route or into a closed metatheoretic theorem. Therefore the read remains consumable only with D still visible as a separate unmet row, and the listed subject-reduction, consistency, beta-conversion, quotient, host-equality, selected-limit, and completeness claims stay outside the packet. □

Theorem 765.175 (MetaCIC candidate route discharge refusal)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization-facing read that satisfies the four-face audit precondition still treats D only as the displayed discharge socket coordinate. The route supplies no witness that fills D, no closed subject-reduction theorem, and no full closed-consistency theorem.

Proof. Start with the candidate-mediated boundary. Theorem 765.172 keeps the SN-facing read on the finite packet rows $S, N, O, U, D, H, C, P, L$ and the four local Phase Real faces.

Next impose the audit precondition. Theorem 765.173 says that consumption as current Phase $\mathsf{Real}^{\uparrow}$ and Completion evidence exposes only the four separate scoped-closure and audit-clean face rows, supported by BHist, hsame, Cont, Pkg, and NameCert data. It does not add an aggregate discharge credential.

Now apply the handoff exhaustion and root-unblock boundaries. Theorem 765.171 leaves closed subject reduction and full closed consistency outside the confluence-decidability route, while Theorem 765.168 gives the same exclusion for the current Phase Real root-unblock route.

Therefore the candidate-mediated SN route can guide the local handoff and audit read, but it cannot turn the socket coordinate D into a discharged witness or export the closed subject-reduction and full-consistency theorems. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateRouteDischargeRefusal`

Theorem 765.176 (MetaCIC candidate SN socket nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every downstream consumer using the candidate-mediated strong-normalization route receives only the socket-aware finite route over $S, N, O, U, D, H, C, P, L$ together with the four local face rows for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The coordinate D remains a separate discharge socket and is not absorbed into the four-face route, the consumer surface, closed subject reduction, or full closed consistency.*

Proof. First separate the socket from the faces. Theorem 765.170 keeps the four Phase Real face records and the discharge socket as distinct obligations; neither side fills or promotes the other.

Next use source nonescape. Theorem 765.158 prevents admitted Phase Real and Completion reads from bypassing the displayed packet and face rows through quotient real data, host equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Finally apply the discharge refusal theorem, Theorem 765.175. It prevents the candidate-mediated SN route from filling D or exporting closed subject reduction and full closed consistency. Hence every downstream consumer receives exactly the finite socket-aware route and the four local face rows, with D still recorded as its own socket coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNSocketNonescape`

Theorem 765.177 (MetaCIC candidate SN finite socket normal form)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

a candidate-mediated strong-normalization-facing read is consumable by the current Phase $\mathsf{Real}^{\uparrow}$ *and Completion surface exactly when it displays the finite socket-aware rows $S, N, O, U, D, H, C, P, L$*

together with the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *with support only from* BHist, hsame, Cont, Pkg, *and* NameCert. *This normal form exports no arbitrary strong-normalization theorem, no Church–Rosser theorem, no beta-conversion decidability, no quotient term equality, no host equality, no selected limit, no countable choice, no closed subject reduction, no full consistency theorem, no metric-completeness datum, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, start with the candidate-mediated SN boundary, Theorem 765.172. It places every candidate-mediated read on the finite packet rows $S, N, O, U, D, H, C, P, L$ and the four local Phase Real faces.

Next impose the four-face audit precondition, Theorem 765.173. The current Phase $\mathsf{Real}^{\uparrow}$ and Completion consumer can read those faces only through separate local scoped-closure and audit records, carried by BHist, hsame, Cont, Pkg, and NameCert. The current exit threshold, Theorem 38.796, identifies this as the objectwise four-face threshold surface rather than as an aggregate metatheoretic credential.

The negative coordinates are then exhausted by the socket and source boundaries. The discharge refusal theorem, Theorem 765.175, keeps D as a socket coordinate and refuses closed subject reduction and full consistency. The socket nonescape theorem, Theorem 765.176, keeps downstream consumers on the same finite route, while the open-phase source closure and nonescape theorems, Theorems 765.156 and 765.158, exclude off-packet quotient, host-equality, selected-limit, choice, metric-completeness, and $\mathsf{Real}^{\uparrow}$ completeness data.

For the reverse direction, suppose the displayed finite socket-aware rows and four local face records are present with only the stated BEDC support. The open-phase consumer totality theorem, Theorem 765.159, admits exactly this finite ledger as the current consumer surface, and Theorem 38.796 reads the four faces as the local threshold certificate. Since the cited refusal and nonescape theorems leave no hidden socket discharge or off-route coordinate, this displayed data is sufficient for current Phase $\mathsf{Real}^{\uparrow}$ and Completion consumption of the candidate-mediated read. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNFiniteSocketNormalForm`

Theorem 765.178 (MetaCIC critical-path candidate decider exhaustion)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every beta-conversion read or decidability-facing read mediated by the candidate strong-normalization route is confined to the finite socket rows $S, N, O, U, D, H, C, P, L$ *together with the four L10 face records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The route exports no global beta-conversion decision procedure, no Church–Rosser theorem, no quotient term equality, no closed subject reduction theorem, and no full consistency theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDeciderExhaustion`

Proof. Begin with the candidate SN finite socket normal form, Theorem 765.177. It identifies the consumable candidate-mediated read as exactly the socket-aware rows $S, N, O, U, D, H, C, P, L$ plus the four local L10 face records, with support only from BHist, hsame, Cont, Pkg, and NameCert.

Next apply confluence-decidability handoff exhaustion, Theorem 765.171. It keeps confluence-facing and decidability-facing reads on the displayed packet route and refuses global Church–Rosser, beta-conversion decision, closed subject reduction, and full consistency exports.

Finally use the discharge refusal theorem, Theorem 765.175. It prevents the candidate route from filling the socket coordinate D or promoting candidate evidence into a closed metatheorem.

Combining these three boundaries leaves no extra coordinate for quotient term equality, a global beta-conversion decider, closed subject reduction, or full consistency. Hence every beta-conversion or decidability-facing read mediated by candidates is exhausted by the stated finite socket route and four local face records. □

Theorem 765.179 (MetaCIC confluence-decidability candidate SN frontier)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every confluence-facing or decidability-facing read mediated by candidate strong-normalization evidence factors through the finite socket rows $S, N, O, U, D, H, C, P, L$ *together with the four local L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The coordinate* D *remains a separate discharge socket throughout the route. The frontier exports no arbitrary strong-normalization theorem, no Church–Rosser theorem, no beta-conversion decision procedure, no quotient term equality, no selected limit, no countable choice, no closed subject reduction, no full consistency theorem, and no* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityCandidateSNFrontier`

Proof. Begin with the confluence-decidability handoff exhaustion theorem, Theorem 765.171. It confines every confluence-facing or decidability-facing read to the displayed packet rows $S, N, O, U, D, H, C, P, L$ and excludes global Church–Rosser, beta-conversion decision, closed subject reduction, and full consistency exports.

Next add candidate strong-normalization mediation. The candidate-mediated SN boundary, Theorem 765.172, permits only the same packet rows together with the four local L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The finite socket normal form, Theorem 765.177, repacks exactly this data as the consumable current Phase $\mathsf{Real}^{\uparrow}$ and Completion surface.

Now read the confluence side through Theorem 765.181. It keeps confluence-facing use on the handoff row U, the obstruction and socket rows O, D, the structural rows H, C, P, L, and the same four local faces, without exporting an arbitrary Church–Rosser theorem.

Finally apply the discharge-refusal and socket-nonescape results, Theorems 765.175 and 765.176. They keep D visible as a separate socket and prevent candidate evidence or the four faces from being read as a discharge, selected limit, countable-choice row, quotient term equality, closed subject reduction, full consistency, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 765.180 (MetaCIC conditional SN discharge frontier)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any conditional strong-normalization read mediated by candidates factors through the finite socket normal form over K *together with the four local Phase Real faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The discharge socket* D *remains a separate unmet row and is not converted into closed subject reduction, arbitrary strong normalization, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConditionalSNDischargeFrontier`

Proof. Begin with the candidate-mediated SN boundary, Theorem 765.172. It routes the read through $S, N, O, U, D, H, C, P, L$ and the four local Phase Real faces.

Next apply the four-face audit precondition, Theorem 765.173, and the finite socket normal form, Theorem 765.177. Together they identify the consumable evidence as the displayed socket-aware packet plus the separate face-local threshold records.

Finally apply the candidate route discharge refusal theorem, Theorem 765.175. It keeps D as the discharge socket coordinate and refuses closed subject reduction and full consistency. Hence conditional SN remains a frontier read through the displayed packet and faces, with D still unmet. □

Theorem 765.181 (MetaCIC confluence candidate mediation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any confluence-facing read mediated by the candidate route factors through the confluence-decidability handoff row U, the obstruction and socket rows O, D, the structural rows H, C, P, L, and the four local Phase Real faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The read exports no arbitrary Church–Rosser theorem and does not fill the separate discharge socket D.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceCandidateMediation`

Proof. Start with the confluence-decidability handoff exhaustion theorem, Theorem 765.171. It restricts every confluence-facing downstream read of U to the displayed rows O, D, H, C, P, L together with S, N, U.

Attach candidate mediation by applying Theorem 765.172. The only additional evidence available to a candidate-mediated read is the four local Phase Real faces, not a global Church–Rosser theorem or a hidden term-equivalence quotient.

Use Theorem 765.175 to keep the socket row separate. Since D is not filled by the candidate route and the handoff exhaustion theorem excludes arbitrary Church–Rosser export, the confluence-facing read remains mediated by K and the four displayed faces. □

Theorem 765.182 (MetaCIC confluence residual discharge separation)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

a confluence residual row is discharged only when the packet displays a local peak row, a candidate join row, a residual frontier row, a substitution boundary row, a bounded checker row, the obstruction row O, and the four L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Candidate-mediated strong-normalization evidence alone cannot fill the discharge socket D or export closed subject reduction, full consistency, a global Church–Rosser theorem, or a beta-conversion decision procedure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceResidualDischarge`

Proof. Start with the candidate-mediated SN boundary, Theorem 765.172. It routes every candidate-mediated read through the finite packet rows $S, N, O, U, D, H, C, P, L$ and the four local L10 faces, but it does not add a residual-discharge witness.

Next impose the four-face audit precondition, Theorem 765.173. The Phase $\mathsf{Real}^{\uparrow}$ and Completion source surface must expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as separate local records. This audit surface is necessary for a residual confluence read, but it is not the socket discharge.

Now read the confluence and decidability consumer boundary. Theorems 765.171, 765.181, and 765.183 keep confluence-facing and bounded-checker reads inside the handoff row, obstruction and socket rows, structural rows, and four displayed faces. Thus a local peak, candidate join, residual frontier, substitution boundary, and bounded checker must be displayed as rows of the same packet route before any residual-discharge claim can be read.

Finally apply the discharge-refusal and socket-separation results. The candidate route discharge refusal theorem, Theorem 765.175, keeps D unmet, while Theorem 765.176 and Theorem 765.170 prevent the four L10 faces or candidate SN evidence from being absorbed into the discharge socket. Therefore residual confluence discharge requires the displayed residual rows and cannot be supplied by candidate-mediated SN alone. □

Theorem 765.183 (MetaCIC decidability fragment route)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any bounded decidability-fragment read factors through K, the confluence-decidability handoff exhaustion surface, and the four local Phase Real faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The route supplies no global beta-conversion decision procedure, no quotient term equality, no full consistency theorem, and no discharge of the socket row D.*

Proof. Project the decidability-facing read through Theorem 765.171. The handoff permits only the finite rows $S, N, O, U, D, H, C, P, L$ and excludes global beta-conversion decidability, quotient term equality, closed subject reduction, and full consistency.

Now use the candidate SN finite socket normal form, Theorem 765.177. A read consumable by the current Phase $\mathsf{Real}^{\uparrow}$ and Completion surface displays the same socket-aware rows together with the four local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The discharge refusal theorem, Theorem 765.175, keeps D unmet. Therefore the decidability-fragment route is bounded by the displayed packet and face records, and it cannot be promoted to global beta-conversion decidability, quotient equality, full consistency, or a discharged socket. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDecidabilityFragmentRoute`

Theorem 765.184 (MetaCIC neutral-spine decidability handoff)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

a neutral-spine decidability read may be consumed by the current Phase $\mathsf{Real}^{\uparrow}$ *route only after candidate-mediated strong normalization, confluence mediation, and bounded decidability-fragment rows expose the four local L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The handoff supplies no aggregate formalstatus, no bridge row, no quotient stream equality, no selected limit, no choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNeutralSpineDecidabilityHandoff`

Proof. Start with the candidate-mediated strong-normalization boundary, Theorem 765.172. It allows a neutral-spine read to reach the current route only through the finite packet rows $S, N, O, U, D, H, C, P, L$ and the four local Phase Real faces.

Next add confluence mediation. Theorem 765.181 keeps confluence-facing use on the handoff row U, the obstruction and socket rows O, D, the structural rows H, C, P, L, and the same four displayed faces, while refusing arbitrary Church–Rosser export and any filling of D.

Now apply the bounded decidability fragment route, Theorem 765.183. Decidability-facing evidence remains bounded by K, the confluence-decidability handoff exhaustion surface, and the four local faces; it does not become a global beta-conversion decision procedure or quotient term equality.

Finally read the route through the four-face audit precondition, Theorem 765.173. The current Phase $\mathsf{Real}^{\uparrow}$ consumer sees separated $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ records, not an aggregate verification row or bridge credential. The excluded aggregate status, bridge, quotient stream, selected-limit, choice, and ambient-completeness objects are therefore outside the neutral-spine handoff. □

Theorem 765.185 (MetaCIC neutral-spine four-face budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any neutral-spine read used jointly for candidate-mediated strong normalization, confluence mediation, and bounded decidability must expose separate face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the finite socket rows* $S, N, O, U, D, H, C, P, L$. *It exports no aggregate formalstatus, bridge row, quotient term equality, selected limit, closed subject reduction, full consistency, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNeutralSpineFourFaceBudget`

Proof. Start with the candidate-mediated strong-normalization boundary, Theorem 765.172. It keeps a neutral-spine read on the finite socket rows and on the four local Phase Real faces.

Add confluence mediation through Theorem 765.181. The confluence-facing use stays on the handoff, obstruction, socket, transport, replay, provenance, and local naming rows; it does not discharge D or export closed subject reduction.

Now apply the bounded decidability fragment route, Theorem 765.183, together with the neutral-spine handoff of Definition 765.184. The decidability read remains bounded by the same packet and cannot become global beta-conversion decidability or quotient term equality.

Finally impose the four-face audit precondition, Theorem 765.173. It requires $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to remain separate face records. Therefore the joint read exposes the displayed socket rows and four faces only, leaving aggregate status, bridge rows, selected limits, closed subject reduction, full consistency, and $\mathsf{Real}^{\uparrow}$ completeness outside the budget. □

Theorem 765.186 (MetaCIC neutral-spine decidability budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any current Phase $\mathsf{Real}^{\uparrow}$ *consumer of a neutral-spine decidability row must expose* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as four separate local faces before it consumes the decidability evidence. The row remains bounded by the finite packet route* $S, N, O, U, D, H, C, P, L$ *and exports no aggregate formalstatus, bridge row, quotient stream equality, selected limit, global beta-conversion decision procedure, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the neutral-spine handoff of Definition 765.184. It routes a decidability read through candidate-mediated strong normalization, confluence mediation, and the bounded decidability fragment before it reaches the current Phase $\mathsf{Real}^{\uparrow}$ surface.

Apply the decidability fragment route, Definition 765.183. It keeps the bounded checker on the finite packet rows and prevents promotion to a global beta-conversion decision procedure or quotient term equality.

Finally impose the four-face audit precondition, Definition 765.173. The current consumer must see $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as separated local records; an aggregate status, bridge row, selected limit, or ambient completeness claim is outside the decidability budget. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNeutralSpineDecidabilityBudget`

Theorem 765.187 (MetaCIC residual confluence budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any residual-confluence read consumed by the current L10 route must display the same four face records* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before it consumes the confluence row. Candidate-mediated evidence may guide the residual route, but it cannot supply a global Church–Rosser theorem, closed subject reduction, full consistency, socket discharge, quotient term equality, or a Real bridge claim.*

Proof. Begin with confluence mediation, Definition 765.181. It keeps confluence-facing use on the handoff row U, obstruction and socket rows O, D, structural rows H, C, P, L, and the four local faces.

Use the residual discharge separation theorem, Definition 765.182. Residual confluence requires local peak, candidate join, residual frontier, substitution boundary, bounded checker, obstruction, and face rows; candidate-mediated strong normalization alone does not fill the socket.

The four-face audit precondition then keeps the L10 evidence objectwise. Thus a current consumer reads residual confluence only after $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are separately exposed, and the listed global metatheorems and Real bridge claims remain outside the budget. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualConfluenceBudget`

Theorem 765.188 (MetaCIC residual confluence row budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

no residual confluence discharge may be consumed by the current L10 route until the same packet displays a local peak row, candidate join row, residual frontier row, substitution boundary row, bounded checker row, obstruction row, and the four separate L10 face records $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The displayed rows remain a bounded budget for residual confluence; they do not assert global Church–Rosser, full beta-conversion decidability, closed subject reduction, full consistency, socket discharge, quotient term equality, or a Real bridge.*

Proof. Start with the candidate-mediated strong-normalization boundary, Definition 765.172. It places candidate evidence on the finite packet rows and the four local L10 faces, but it does not create a residual discharge.

Read the confluence side through Definition 765.181 and the residual discharge separation theorem Definition 765.182. These force local peak, candidate join, residual frontier, substitution boundary, bounded checker, obstruction, and face rows to remain visible before any residual-confluence use is admitted.

Now use the decidability fragment route, Definition 765.183. The bounded checker row stays inside the same packet route and cannot become a global beta-conversion decision procedure or quotient term equality.

Finally impose the four-face audit precondition, Definition 765.173, and the candidate route discharge refusal theorem, Definition 765.175. The four faces remain separate records, while the

socket and obstruction rows remain unmet budget coordinates. Therefore the current consumer may read only the displayed residual-confluence budget and none of the excluded global conclusions. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualConfluenceRowBudget`

Theorem 765.189 (MetaCIC candidate SN discharge budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, candidate-mediated strong-normalization evidence may be consumed by the current Phase* $\mathsf{Real}^{\uparrow}$ *and Completion surface only after the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are displayed separately. The evidence does not discharge the socket row* D*, does not prove closed subject reduction or full consistency, and does not collapse SN, residual confluence, and neutral-spine decidability into one endpoint claim.*

Proof. Project the candidate-mediated SN boundary through Definition 765.172. It permits the SN-facing read only on the finite packet rows and the four Phase Real faces.

Apply the four-face discharge precondition strengthening, Definition 765.174. The socket row D remains visible as an unmet row distinct from $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use the socket nonescape and four-face budget results, Definition 765.176 and Definition 765.185. They prevent the candidate route from discharging D or from merging SN, confluence, and decidability into a single endpoint surface. The current consumer therefore receives separated L10 faces and a still-visible socket budget. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNDischarge`

Theorem 765.190 (MetaCIC critical-path confluence-decidability exit budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization read that is used both as confluence evidence and as bounded decidability-fragment evidence for the current L10 route must expose four independent phase faces: $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Each face carries its own* BHist *support, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The route supplies no aggregate formalstatus, no bridge row, no quotient stream equality, no selected limit, no choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityExitBudget`

Proof. Begin with the confluence-decidability handoff exhaustion theorem, Theorem 765.171. It confines confluence-facing and decidability-facing reads to the finite packet rows $S, N, O, U, D, H, C, P, L$ and excludes closed subject reduction, full consistency, global beta-conversion decidability, and quotient term equality.

Next apply the candidate-mediated SN boundary, Theorem 765.172. Candidate mediation can add only the four current Phase Real faces, namely $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, to the displayed packet route.

Now impose the four-face audit precondition, Theorem 765.173. It requires those four faces to appear as separate local records, each with its own support from BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data rather than as one route-level credential.

Finally combine Definition 765.181 with Definition 765.183. The same separated face data must serve both confluence and decidability uses, while aggregate status rows, bridge rows, quotient

stream equality, selected limits, choice schedules, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the current L10 exit budget. □

Theorem 765.191 (MetaCIC confluence-decidability frontier discharge readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the finite candidate-mediated strong-normalization rows expose a discharge-ready frontier for the confluence-decidability handoff exactly at the residual confluence, neutral-spine, bounded-checker, obstruction, and socket rows of K*. The socket row* D *remains the finite discharge coordinate that can be inspected by the current route, while global subject reduction, global confluence, unrestricted beta-decidability, full consistency, quotient term equality, and any ambient* $\mathsf{Real}^{\uparrow}$ *completion claim remain outside the closed scope of the chapter.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityFrontierDischargeRe`

Proof. Start from the candidate SN discharge budget of Definition 765.189. It places candidate-mediated strong-normalization evidence on the finite rows of K and keeps the socket row D visible instead of treating it as globally discharged.

Next combine the residual confluence row budget, Definition 765.188, with the neutral-spine decidability budget, Definition 765.186. The handoff can inspect local peak, candidate join, residual frontier, substitution boundary, bounded checker, obstruction, and neutral-spine rows, but each of these rows is still a displayed coordinate of the same finite packet.

Finally apply the exit budget of Definition 765.190. It keeps confluence-facing and decidability-facing reads on the separated packet faces and blocks any promotion to global subject reduction, global confluence, unrestricted beta-decidability, full consistency, quotient term equality, or ambient Real completion. □

Theorem 765.192 (MetaCIC confluence-decidability budget)**.** *Every MetaCIC confluence-facing or decidability-facing consumer visible at the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root must first pass through the open L10 source ledger displayed by Definition 765.155: the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, together with the finite packet rows* $S, N, O, U, D, H, C, P, L$ *and their* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. No such consumer may bypass those rows by reading a separate Church–Rosser theorem, global beta-conversion decision procedure, closed subject-reduction row, completed-real claim, quotient stream equality, selected limit, or host equality.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityBudget`

Proof. Begin with the Phase Real dependency budget. Theorem 765.108 confines dyadic observation reads to displayed budget rows, and Theorem 765.109 orders the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces before the Real seal handoff.

Next read the open-phase source ledger. Definition 765.155 lists exactly the packet rows $S, N, O, U, D, H, C, P, L$ and the four Phase Real source rows. Theorems 765.156 and 765.159 state that current Phase Real and Completion consumers receive this ledger as their whole source surface.

Finally attach the confluence-decidability handoff exhaustion. Theorem 765.171 bounds confluence-facing and decidability-facing reads by the displayed socket and packet rows, while the ledger nonescape theorem, Theorem 765.158, removes quotient real data, host equality, selected limits, countable choice, compactness substitutes, and $\mathsf{Real}^{\uparrow}$ completeness.

Combining these carrier inventories leaves only the displayed L10 ledger and BEDC support rows. Since none of the cited rows carries an independent confluence theorem, beta-conversion decision procedure, closed subject-reduction proof, completed-real principle, quotient equality, selected limit, or host equality, every visible confluence or decidability consumer remains budgeted by the ledger. □

Theorem 765.193 (MetaCIC candidate SN confluence-decidability budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

whose candidate-mediated strong-normalization budget is displayed by the finite socket route, every confluence-facing or decidability-facing handoff must pass through the existing budget rows: the BHist *packet rows* $S, N, O, U, D, H, C, P, L$*, the candidate budget surface over the four local faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, the confluence handoff row* U*, the decidability handoff read of the same row, the componentwise* Cont *replay* C*, and the* NameCert *provenance* P, L*. The route exports no separate Church–Rosser theorem, no global beta-conversion decision procedure, no hidden discharge of* D*, and no host normalization theorem.*

Proof. Start with the candidate-mediated SN finite socket normal form, Definition 765.177. It identifies the displayed candidate budget as the socket-aware finite rows $S, N, O, U, D, H, C, P, L$ together with the four local Phase Real faces, supported only by BHist, hsame, Cont, Pkg, and NameCert.

Next route confluence and decidability through the existing handoff surface. The handoff exhaustion theorem, Definition 765.171, confines every confluence-facing or decidability-facing read of U to the rows O, D, H, C, P, L and the remaining packet coordinates S, N, U. The budget theorem, Definition 765.192, then forces those reads to pass through the open source ledger and its displayed BEDC support rows.

Finally compose these two inventories. The candidate budget contributes only the four local faces and the socket-aware packet route; the handoff budget contributes only U and the same transport, replay, provenance, and naming rows. Since neither surface contains a coordinate for an independent confluence theorem, a global decidability procedure, a discharge witness for D, or a host normalization theorem, every downstream confluence or decidability use remains inside the displayed budget rows. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSNConfluenceBudget`

Theorem 765.194 (MetaCIC SN discharge-socket readiness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any consumer that tries to use candidate-mediated strong normalization for a confluence-facing or decidability-facing read must display the rows S, N, U, D, H, C, P, L*, the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, the confluence-audit dependency exposure row, and the closed-substitution boundary exposure row before any discharge socket is consumed. The read is therefore a readiness surface over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, not a closed subject-reduction certificate, global confluence theorem, beta-conversion decision procedure, or filled discharge socket.*

Proof. Start with the SN confluence-decidability budget, Definition 765.193. It forces every confluence-facing or decidability-facing read using candidate-mediated strong normalization to pass through the displayed packet rows and the four L10 faces before the handoff row can be consumed.

Next apply the candidate-mediated obligation surface, Definition 765.257, and the normalization-frontier dependency exposure, Definition 765.291. These rows keep the strong-normalization source, the normal-form endpoint, the handoff row, transport, replay, provenance, and local naming attached to the same accepted packet.

Now attach the confluence and substitution boundaries. The confluence-audit exposure Definition 765.292 keeps confluence rows as bounded audit data, while the closed-substitution boundary exposure Definition 765.293 keeps substitution support behind the same handoff row and discharge socket.

Finally use the candidate-discharge frontier coverage, Definition 765.281. It preserves the obstruction and socket rows while candidate-mediated SN, confluence-facing, bounded-decidability, and L10 readiness rows are consumed. Thus a consumer reaches the socket only after the named readiness surface is visible, and no consumed row fills D without a separate subject-reduction certificate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSNDischargeSocketReadiness`

Theorem 765.195 (MetaCIC candidate SN confluence checker factorization)**.** *Every consumer that combines candidate-mediated strong normalization, the confluence handoff, and bounded checker rows for an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L)$$

must factor through the concrete rows $S, N, O, U, D, H, C, P, L$ *and the L10 four-face ledger*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The factorization may read candidate-mediated normal-form evidence, the confluence handoff, and the bounded checker budget only along that route. It cannot discharge subject reduction, fill the open discharge socket D*, assert global beta-conversion decidability, introduce quotient term equality, select a limit, or import ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNConfluenceCheckerFactorizatio`

Proof. Begin with the candidate-mediated strong-normalization surface. By Definition 765.193 and Definition 765.257, the consumer must display $S, N, O, U, D, H, C, P, L$ together with the four L10 faces before any confluence-facing or decidability-facing read is available.

Next read the confluence handoff. The row U is available only as part of the same packet route, while D remains an open discharge socket rather than a subject-reduction proof or a closed conversion theorem.

Finally read the bounded checker budget. The checker rows are downstream of the displayed handoff and replay rows, so they can compare only the finite ledger already carried by K and the L10 faces. Since the carrier has no coordinate for discharged subject reduction, global beta-conversion decidability, quotient equality, selected limits, or ambient Real completeness, the combined consumer factors through the stated route. □

Theorem 765.196 (MetaCIC SN confluence-decidability handoff)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$ *that displays a strong-normalization budget row, a confluence budget row, and a decidability budget row, every closed-normal-form membership decision routed through the packet must first pass through those three rows, then through the continuation and provenance rows* C, P, L*, before it exports the current open-phase ledger. The route exports*

no full MetaCIC consistency theorem, no closed subject-reduction theorem, no arbitrary Church–Rosser theorem, no global beta-conversion decision procedure, and no discharge of the socket row D.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSNConfluenceDecidabilityHandoff`

Proof. Start with the SN budget row. The candidate SN confluence-decidability budget theorem, Definition 765.193, places the strong-normalization read inside the finite socket route $S, N, O, U, D, H, C, P, L$ and the four current Phase Real faces.

Next route confluence and decidability through the existing handoff surface. The confluence-decidability handoff exhaustion theorem, Definition 765.171, confines confluence-facing and decidability-facing reads to U together with the obstruction, socket, transport, replay, provenance, and naming rows.

Then apply the confluence-decidability budget theorem, Definition 765.192. It requires these reads to pass through the open L10 source ledger before any current Phase Real or Completion consumer may inspect them.

Finally compose the three rows. A closed-normal-form membership decision may be read only after strong normalization supplies the normal-form route, confluence supplies the comparison handoff, and decidability supplies the bounded checker handoff. Since all three rows remain inside K and its open-phase ledger, the route cannot promote itself to full MetaCIC consistency, closed subject reduction, arbitrary Church–Rosser, global beta-conversion decidability, or a discharged socket. □

Definition 765.197 (MetaCIC candidate-mediated SN frontier)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the candidate-mediated strong-normalization frontier is the finite subroute that displays the strong-normalization row S, the normal-form endpoint N, the handoff row U, the discharge socket D, the componentwise transport row H, the continuation row C, the provenance row P, and the local naming row L, together with the four local source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *already recorded for the open-phase source ledger. The frontier may be consumed only as a boundary between the candidate-mediated SN evidence and the confluence-decidability handoff; it contains no coordinate for closed subject reduction, arbitrary strong normalization, global confluence, global beta-conversion decidability, quotient term equality, host equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontier`

The candidate-mediated frontier sits between the normalization audit packet of chapter 3779 and the redex-frontier packet of chapter 1008. Its rows are consumed only through closed typed term data, candidate evidence, application and Pi adequacy rows, redex-frontier obstruction rows, componentwise transport, replay, provenance, and local NameCert data; the frontier never supplies an untyped redex route or a detached global normalization claim.

Theorem 765.198 (MetaCIC candidate-mediated SN frontier)**.** *Every candidate-mediated strong-normalization read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that is passed to the confluence-decidability handoff must factor through the frontier of Definition 765.197. In particular, the read must display S, N, U, D, H, C, P, L, keep the obstruction row O and discharge socket D available to the surrounding packet, and expose the four local faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any confluence-facing or decidability-facing consumer reads the handoff row U.*

Proof. Begin with the candidate-mediated SN finite socket normal form, Definition 765.177. It confines the candidate-mediated read to the socket-aware packet rows $S, N, O, U, D, H, C, P, L$ together with the four local source faces.

Next apply the SN confluence-decidability budget theorem, Definition 765.193. It requires any confluence-facing or decidability-facing consumer to read the same candidate budget surface before it consumes the handoff row U.

The frontier definition simply names the common boundary needed by these two routes: the candidate-mediated read supplies S, N, U, D, H, C, P, L and the four local faces, while the enclosing packet still retains O and D as visible blocked-route data. Since neither cited theorem supplies a coordinate for closed subject reduction, arbitrary strong normalization, global confluence, global beta-conversion decidability, quotient equality, host equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, the factorization cannot export any of those claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket candidate mediated sn frontier`

Theorem 765.199 (MetaCIC candidate-mediated frontier socket separation)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated frontier replay that is passed toward the handoff row U must keep the obstruction row O and discharge socket D visible as separate rows from S, N, U, H, C, P, L and from the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The replay may use the frontier as strong-normalization evidence for the handoff boundary, but it cannot fill the subject-reduction socket or export closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the candidate-mediated frontier theorem, Definition 765.198. It places the candidate-mediated read on S, N, U, D, H, C, P, L and the four L10 faces before the handoff row U is consumed.

Apply the handshake theorem Definition 765.223. The handoff may inspect the frontier rows, but the theorem keeps O and D visible unless a separate closed subject-reduction row is supplied.

Use the discharge-row theorem and discharge-socket induction, Definition 765.225 and Definition 765.226. These preserve the separation between the replay rows and the socket row while finite frontier rows are consumed.

Finally apply candidate-normalization socket exhaustion, Definition 765.230. The exhausted surface contains no coordinate for the refused global metatheoretic or Real-completion claims, so the socket remains a displayed boundary rather than a discharged theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierSocketSeparatio`

Theorem 765.200 (MetaCIC candidate-mediated frontier L10 readback)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated read of the handoff row U must expose the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before U*

is handed to any confluence-facing or decidability-facing consumer. The obstruction row O and discharge socket D remain visible throughout that L10 readback. No such readback supplies closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the candidate-mediated SN frontier Definition 765.198. It requires the four L10 faces to be present before a confluence-facing or decidability-facing consumer reads U.

Use the candidate-normalization budget theorem, Definition 765.227. The budget identifies the finite rows S, N, U, H, C, P, L together with the same four faces as the whole normalization surface available to the consumer.

Apply the handshake and consumer-boundary theorems, Definition 765.223 and Definition 765.224. They allow the handoff row to be consumed only after the candidate-mediated and L10 source data have been displayed.

The discharge-row theorem keeps D separate, and socket exhaustion keeps O, D visible after the finite readback. Since the consumed rows are exactly the frontier rows plus the four faces, no quotient equality, selected limit, host equality, Real-completeness, or global MetaCIC theorem is exported. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierL10Readback`

Theorem 765.201 (MetaCIC candidate frontier Real source lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated handoff that reads the row U must expose* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as source rows, with the obstruction row O and discharge socket D still visible, before any confluence-facing or decidability-facing consumer reads the handoff. The handoff therefore treats the* $\mathsf{Real}^{\uparrow}$ *face as a named L10 source row, not as selected limits, host equality, ambient Real completeness, quotient term equality, closed subject reduction, full confluence, or beta-conversion decidability.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierRealSourceLock`

Proof. Start from the candidate-mediated frontier and L10 readback theorems, Definition 765.198 and Definition 765.200. Together they require the finite candidate rows and the four L10 faces before the handoff row U is consumed.

Apply the handshake theorem, Definition 765.223. It permits confluence-facing or decidability-facing replay only after the candidate-mediated source rows have been displayed.

Use the discharge-row theorem, socket separation, and socket exhaustion, Definition 765.225, Definition 765.199, and Definition 765.230. These keep O and D visible while the finite handoff reads U.

Since the displayed source rows contain no coordinate for selected limits, host equality, Real completeness, quotient term equality, closed subject reduction, full confluence, or beta-decidability, the Real-facing source lock exports only the named L10 row sequence. □

Theorem 765.202 (MetaCIC candidate frontier source lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated handoff through U must expose the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *together with obstruction row O and discharge socket D before confluence-facing or decidability-facing consumers*

can read the handoff. The handoff is therefore locked to finite source rows and exports no hidden global normalization, confluence, beta-conversion decidability, quotient term equality, selected limit, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierSourceLockCertificate`

Proof. Start with the candidate-mediated SN frontier of Definition 765.198. It places any handoff through U behind the candidate rows and the four L10 faces before a confluence-facing or decidability-facing read can occur.

Apply the L10 readback theorem and the Real source-lock theorem, Definition 765.200 and Definition 765.201. They identify $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ as displayed source rows, with O and D still visible.

Use the handshake theorem, Definition 765.223. It allows consumer replay only after the finite candidate-mediated source data have been displayed.

Finally apply socket separation and socket exhaustion, Definition 765.199 and Definition 765.230. They keep the obstruction and discharge rows separate from confluence and decidability consumers, so the handoff cannot hide a global normalization, quotient equality, selected-limit, host-equality, or Real-completeness coordinate. □

Theorem 765.203 (MetaCIC candidate frontier RegSeqRat handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row is the only regular-rational route available to a candidate-mediated handoff through* U*. Consequently the frontier exports no selected limit, quotient term equality, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, closed subject reduction, full confluence, or alternate regular-rational approximation route beyond the displayed* $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ *source window.*

Proof. Read the L10 readback theorem, Definition 765.200. It places $\mathsf{RegSeqRat}^{\uparrow}$ inside the four-face source window that must be exposed before any confluence-facing or decidability-facing consumer reads U.

Next use the frontier handoff totality theorem, Definition 765.206. The handoff surface is exhausted by S, N, U, H, C, P, L, the visible rows O, D, and the same four L10 faces.

Apply discharge coverage and socket exhaustion, Definition 765.234 and Definition 765.230. They leave no hidden coordinate through which another regular-rational route could fill D or bypass the displayed source window.

Thus the only regular-rational readback available to the handoff is the named $\mathsf{RegSeqRat}^{\uparrow}$ row. The excluded selected-limit, quotient-equality, host-equality, Real-completeness, closed-subject-reduction, and full-confluence claims are not exported by the frontier. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierRegSeqRatHandoff`

Theorem 765.204 (MetaCIC candidate frontier source-window determinacy)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Suppose two candidate-mediated handoff reads expose the same four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

with the same visible obstruction row O and discharge socket D. Then the handoff row U has the same finite source-window classifier before any confluence-facing or decidability-facing consumer reads it. The conclusion is only determinacy of that displayed source window; it does not close subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the candidate-mediated frontier and L10 readback theorems, Definition 765.198 and Definition 765.200. They force both reads of U to pass through the same four displayed L10 faces before any confluence-facing or decidability-facing consumer can inspect the handoff.

Use the Real source-lock and RegSeqRat handoff theorems, Definition 765.201 and Definition 765.203. These identify the $\mathsf{Real}^{\uparrow}$ face as a named source row and make $\mathsf{RegSeqRat}^{\uparrow}$ the only regular-rational route available to the handoff, so the two reads have no alternate source-window classifier to choose.

Apply socket separation and handoff totality, Definition 765.199 and Definition 765.206. Since O and D are the same visible boundary rows in both reads, totality exhausts U by S, N, U, H, C, P, L together with that fixed boundary and the same four source faces.

Finally read discharge coverage and source exhaustion, Definition 765.210 and Definition 765.211. They leave no hidden source row, quotient source, selected-limit source, host-equality source, confluence source, beta-decidability source, or Real-completeness source. The finite classifier visible at U is therefore determined by the shared L10 window and the shared O, D boundary. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierSourceWindowDeterminacy`

Theorem 765.205 (MetaCIC candidate-mediated SN L10 completion lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any accepted packet that reaches the candidate-mediated strong-normalization handoff through U must expose exactly the source rows* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any confluence-facing or decidability-facing consumer reads the handoff. The obstruction row O and discharge socket D remain visible, and the handoff exports no selected limits, host equality, quotient term equality, closed subject reduction, global confluence, beta-conversion decidability, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNL10CompletionLock`

Proof. Start from the candidate-mediated frontier theorem and the L10 readback theorem, Definition 765.198 and Definition 765.200. Together they require the four displayed L10 source rows before a consumer may read U.

Use the RegSeqRat handoff theorem, Definition 765.203. It locks the regular-rational part of the route to the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row inside the same four-row window.

Apply socket separation and obstruction nonescape, Definition 765.199 and Definition 765.207. These keep O, D visible while the handoff rows are consumed.

Finally read discharge coverage. The consumed surface is exhausted by S, N, U, H, C, P, L, the visible O, D boundary, and the four L10 source rows, so no selected-limit, quotient-equality, host-equality, closed-subject-reduction, global-confluence, beta-decidability, or Real-completeness claim is exported. □

Theorem 765.206 (MetaCIC candidate-mediated frontier handoff totality)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any finite confluence-facing or decidability-facing handoff that uses candidate-mediated strong-normalization evidence is exhausted by the route*

$$S, N, U, H, C, P, L$$

together with the visible boundary rows O, D *and the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Thus the handoff is total only for the displayed frontier surface; it refuses closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the candidate-mediated frontier and the confluence-decidability handshake. Together they route any candidate-mediated handoff through S, N, U, H, C, P, L while keeping the four L10 faces available before U is consumed.

Next use the candidate-normalization budget theorem. It shows that the finite budget of the handoff is precisely the displayed candidate-normalization surface, not an unrestricted strong-normalization or confluence theorem.

Apply discharge-socket induction to the finite handoff rows. Each consumed row preserves O and leaves D unfilled, because none of the rows is an explicit closed subject-reduction coordinate.

Finally invoke candidate-normalization socket exhaustion. Once the displayed surface and the four L10 faces have been consumed, there is no remaining packet coordinate that could supply quotient term equality, selected limits, host equality, Real completeness, global confluence, beta-conversion decidability, or arbitrary strong normalization. The handoff is therefore total exactly over the named frontier and no more. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierHandoffTotality`

Theorem 765.207 (MetaCIC candidate-mediated frontier obstruction nonescape)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every finite handoff that uses candidate-mediated strong-normalization evidence keeps the obstruction row* O *and the discharge socket* D *visible until a separate closed subject-reduction row is displayed. The handoff may consume* S, N, U, H, C, P, L *and the L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *but it cannot erase* O, *fill* D, *or export closed subject reduction, full confluence, global beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the candidate-mediated frontier and the handoff totality theorem, Definition 765.198 and Definition 765.206. They identify the handoff surface as the finite route through S, N, U, H, C, P, L, the visible boundary rows O, D, and the four L10 faces.

Apply the handshake theorem Definition 765.223. It permits confluence-facing or decidability-facing replay only through the displayed route, and it leaves O, D open unless the packet separately displays closed subject reduction.

Use the discharge-row theorem and discharge-socket induction, Definition 765.225 and Definition 765.226. These keep D separate from the candidate-mediated rows while finite replay consumes the handoff surface.

Finally apply socket exhaustion, Definition 765.230. Once S, N, U, H, C, P, L and the four L10 faces have been consumed, the packet has no remaining coordinate that could hide a closed subject-reduction row or discharge the obstruction. The listed global metatheoretic, quotient, selected-limit, host-equality, and completeness claims therefore remain outside the handoff. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierObstructionNones

Theorem 765.208 (MetaCIC candidate-mediated SN frontier discharge)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A candidate-mediated strong-normalization handoff may approach the discharge socket only through the displayed rows*

$$S, N, O, U, D, H, C, P, L$$

and the L10 source faces

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The socket D remains unfilled until a separate closed subject-reduction row is displayed. In particular the handoff exports no closed subject reduction, unrestricted strong normalization, full confluence, beta-conversion decidability, quotient term equality, selected limit, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the candidate-mediated frontier theorem and L10 readback theorem, Definition 765.198 and Definition 765.200. They place any candidate-mediated SN read behind the rows $S, N, O, U, D, H, C, P, L$ and force the four L10 faces to be displayed before the handoff row U is consumed.

Next apply frontier handoff totality, Definition 765.206. It exhausts the handoff route by S, N, U, H, C, P, L, the visible boundary rows O, D, and the same four L10 source faces, so a consumer has no additional route into the discharge socket.

Use obstruction nonescape, Definition 765.207. It keeps O and D visible while the finite candidate-mediated route is replayed, and it blocks any reading of candidate-mediated SN as a closed subject-reduction row.

Finally apply socket exhaustion, Definition 765.230. After the displayed rows and L10 faces have been consumed, no packet coordinate remains from which quotient equality, selected limits, host equality, Real completeness, global confluence, beta-decidability, unrestricted strong normalization, or closed subject reduction could be recovered. The discharge boundary is therefore approached but not filled. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierDischarge

Theorem 765.209 (MetaCIC candidate-mediated SN socket-induction frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Finite induction over the displayed candidate-mediated handoff rows preserves the separation of obstruction row O and discharge socket D while consuming*

$$S, N, U, H, C, P, L$$

and the L10 faces

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

No induction step exports closed subject reduction, global confluence, beta-decidability, quotient term equality, selected limits, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNSocketInductionFrontie`

Proof. The base case is the unconsumed frontier. Apply Definition 765.198 and Definition 765.199. They display $S, N, O, U, D, H, C, P, L$ and keep the obstruction row and discharge socket separate before the handoff row is read.

For the L10 readback step, use Definition 765.200. It requires $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$ to remain visible before any confluence-facing or decidability-facing consumer consumes U.

For the handoff step, apply Definition 765.206. The consumed surface is exhausted by S, N, U, H, C, P, L together with the visible rows O, D and the same four L10 faces.

For the obstruction step, read Definition 765.207 and Definition 765.234. They preserve the nonescape of O and the socket status of D through every finite replay row.

The induction therefore has no case whose constructor supplies closed subject reduction, global confluence, beta-decidability, quotient term equality, selected limits, host equality, or Real completeness. The socket boundary remains visible after the finite route is consumed. □

Theorem 765.210 (MetaCIC candidate frontier discharge coverage)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every finite candidate-mediated handoff that first consumes the strong-normalization frontier and then reaches the confluence-decidability boundary is exhausted by the replay rows*

$$S, N, U, H, C, P, L,$$

the visible obstruction and socket rows O, D*, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The coverage conclusion is only this finite discharge surface. It refuses closed subject reduction, arbitrary strong normalization, full confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, and $\mathsf{Real}^{\uparrow}$ *completeness unless a later packet displays separate rows for those claims.*

Proof. Begin with the candidate-mediated frontier theorem and handoff totality, Definition 765.198 and Definition 765.206. They force any accepted handoff through S, N, U, H, C, P, L together with the four L10 faces before a confluence-facing or decidability-facing consumer reaches the handoff boundary.

Next apply obstruction nonescape, Definition 765.207. The obstruction row O and discharge socket D remain visible while the frontier is consumed, so the handoff cannot erase the subject-reduction socket by replaying candidate-mediated SN evidence.

Use the discharge-row theorem and discharge-socket induction, Definition 765.225 and Definition 765.226. These make the finite discharge route stable under each consumed row and keep D separate from S, N, U, H, C, P, L.

Finally read the existing discharge coverage theorem, Definition 765.234. It packages the same rows as the whole available frontier budget. Since none of the displayed coordinates is closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or $\mathsf{Real}^{\uparrow}$ completeness, the stated refusals remain outside the discharge coverage. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierDischargeCoverage`
`semantic name certificate`

Theorem 765.211 (MetaCIC candidate frontier source exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated handoff that reaches the confluence-decidability boundary has exactly the displayed source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

together with the visible obstruction and discharge rows O, D*. Replaying* U *cannot add a redex-frontier source, discharge-socket source, quotient source, selected-limit source, host-equality source, closed-subject-reduction source, full-confluence source, beta-decidability source, or ambient Real-completeness source.*

Proof. First read the candidate-mediated frontier, the L10 readback theorem, and the completion lock. These results place every handoff that reaches U behind the same four source faces and keep O, D visible at the boundary.

Next apply handoff totality and obstruction nonescape. Totality exhausts the finite handoff surface by S, N, U, H, C, P, L, the visible rows O, D, and the four L10 faces, while nonescape prevents replay from erasing the obstruction row or filling the discharge socket.

Finally use discharge coverage. The discharge surface is already accounted for by the displayed rows, so replaying U has no remaining coordinate from which to import a redex-frontier source, quotient source, selected-limit source, host-equality source, closed-subject-reduction source, full-confluence source, beta-decidability source, or ambient Real-completeness source. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierSourceExhaustion`

Theorem 765.212 (MetaCIC candidate frontier axiom boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. An accepted candidate-mediated handoff through* U *may consume strong-normalization frontier evidence only through the displayed rows* S, N, U, H, C, P, L*, the visible obstruction and discharge rows* O, D*, and the source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

It cannot discharge axioms, closed subject reduction, global confluence, beta-decidability, quotient term equality, selected limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness unless separate rows in the packet display those claims.*

Proof. First read the L10 source locks from Definition 765.200 and Definition 765.205. They force every handoff through U to expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as named source rows before any confluence-facing or decidability-facing consumer reads the handoff.

Next apply the RegSeqRat handoff theorem Definition 765.203. It fixes the regular-rational source of the handoff inside the same four-face window, so the frontier cannot replace that route by selected limits, quotient equality, host equality, or ambient Real completeness.

Use discharge coverage and source exhaustion, Definition 765.210 and Definition 765.211. These theorems exhaust the accepted handoff surface by the displayed candidate rows, the visible obstruction and socket rows, and the four source faces.

Finally read the completion lock together with the exhausted source surface. Axioms, closed subject reduction, global confluence, beta-decidability, quotient term equality, selected limits, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not among those rows. Therefore a candidate-mediated strong-normalization handoff can pass evidence to the boundary, but it cannot silently become a discharge of any listed claim without a separate displayed row for that claim. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierAxiomFrontier`

Theorem 765.213 (MetaCIC candidate-mediated SN route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization read that exports a normalization-frontier conclusion must pass through the displayed* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *row, the* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *boundary row, the* $\mathsf{FiniteObservationConfluence}^{\uparrow}$ *finite-confluence row, the* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *critical-pair ledger, and the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *discharge row before the conclusion is visible. The route may export only that normalization-frontier boundary; it does not assert strong normalization for all MetaCIC terms, closed subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted critical-path packet. The candidate-mediated read starts in the finite rows $S, N, O, U, D, H, C, P, L$, with O and D kept visible as obstruction and discharge-socket data.

Use the existing frontier obligations: Definition 765.198, Definition 765.200, and Definition 765.206. They force the candidate-mediated SN evidence through the named finite frontier before any handoff row is consumed.

Next compose the confluence-facing route. The $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ row and $\mathsf{FiniteObservationConfluence}^{\uparrow}$ row are read only as finite observation boundaries, while the $\mathsf{MetaCICCriticalPairLedger}$ row accounts for the displayed critical-pair surface.

Finally route the remaining boundary through $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$. The discharge row is a socket, not a proof of closed subject reduction, so the resulting conclusion is only a normalization-frontier export over the displayed candidate-mediated route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNRoute semantic name certificate`

Theorem 765.214 (MetaCIC normalization trace residual-frontier route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated normalization read that reaches the residual frontier before a strong-normalization conclusion is consumed must first expose the candidate row, the* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *row, the* $\mathsf{FiniteObservationConfluence}^{\uparrow}$ *row, the* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *row, the* $\mathsf{NormalizationCertificateRouter}$ *row, and the strong-normalization boundary row. The route may use those rows to present a finite normalization trace, but it is not by itself a global strong-normalization theorem, a confluence theorem, a closed subject-reduction theorem, a beta-conversion decision procedure, a quotient term equality, a selected normal form, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationTraceResidualFrontierRoute`

Proof. Project the candidate-mediated frontier rows from K. The candidate row is read through the finite candidate surface, while O and D remain visible as obstruction and discharge-socket data.

Route residual substitution through the candidate-mediated boundary. The MetaCICResidualSubstitutionBounda row is consumed as a displayed boundary row attached to the same packet, so residual replay cannot bypass the candidate row or fill the discharge socket.

Use the finite observation confluence and critical-pair ledgers next. The $\mathsf{FiniteObservationConfluence}^{\uparrow}$ row confines confluence use to finite observations, and the $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ row accounts for the displayed critical-pair surface before any normalization trace is routed onward.

Finally read the NormalizationCertificateRouter row and the strong-normalization boundary row through BHist, hsame, Cont, Pkg, and local NameCert data. These rows expose only a finite normalization trace route, so the global strong-normalization, confluence, subject-reduction, beta-decidability, quotient-equality, selected-normal-form, and Real-completeness claims remain outside the theorem. □

Theorem 765.215 (MetaCIC candidate frontier normal-form witness extraction)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If a candidate-mediated strong-normalization read exposes a normal-form witness at the confluence-decidability boundary, then the witness is readable only through the displayed* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *row, the* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *row, the* $\mathsf{FiniteObservationConfluence}^{\uparrow}$ *row, the* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *row, the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row, and the L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The extraction boundary does not assert global strong normalization, closed subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the candidate-mediated SN route of Definition 765.213. It places every exported normalization-frontier conclusion behind the candidate-mediated SN row, confluence-boundary row, finite-observation confluence row, critical-pair ledger, and subject-reduction discharge socket.

Next read the L10 source boundary from Definition 765.200 and Definition 765.206. These results force the same handoff to expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the witness is passed to any confluence-facing or decidability-facing consumer.

Use discharge coverage and source exhaustion, Definition 765.210 and Definition 765.211. They exhaust the available witness route by the displayed candidate rows, visible obstruction and socket rows, and the four L10 source faces.

Finally apply the axiom boundary of Definition 765.212. The refused global normalization, closed subject-reduction, confluence, beta-decidability, quotient-equality, selected-limit, and Real-completeness claims are not rows in the extracted witness surface, so the normal-form witness remains candidate-mediated boundary data. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierNormalFormWitnessExtract`

Theorem 765.216 (MetaCIC candidate-mediated SN frontier fairness retention)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *packet. If the candidate-mediated strong-normalization route of K is consumed by the normalization frontier, then the same finite premise rows must be visible through*

CandidateMediatedSN$^{\uparrow}$, CandidateSNConfluenceBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICNormaliz

before any normalization certificate row is read. The retention claim is only fairness of the displayed premise route; it does not assert hidden strong normalization, global confluence, quotient term equality, host term equality, or any selected normal form.

Proof. Project the accepted packet K. The candidate-mediated route begins with the displayed rows $S, N, O, U, D, H, C, P, L$, with O and D retained as obstruction and discharge-socket data.

Read the existing candidate-mediated SN route and normal-form witness extraction theorems, Definition 765.213 and Definition 765.215. They force the candidate row, confluence boundary, finite observation rows, critical-pair ledger, discharge socket, and L10 source faces to be displayed before a witness-facing consumer reaches the frontier.

The MetaCICNormalizationFrontier$^{\uparrow}$ and MetaCICNormalizationCertificateRouter$^{\uparrow}$ rows can therefore consume only the same visible premise list. Since the carrier fields have already been exhausted, no hidden strong-normalization theorem, global confluence theorem, quotient equality, host equality, or selected normal form can be read as part of the certificate route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierFairnessRetent`

Theorem 765.217 (MetaCIC candidate-mediated bounded SN discharge route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *packet. A candidate-mediated strong-normalization frontier read discharges a bounded normalization request only after the candidate row, source-window row, residual-frontier row, fairness-retention row, normal-form witness extraction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row are all visible. The discharge is bounded by those displayed rows; it does not assert global strong normalization, full confluence, subject reduction, beta-conversion decidability, quotient term equality, or a selected normal form.*

Proof. Start with the candidate-mediated frontier route of Definition 765.213 and the L10 readback of Definition 765.200. These expose the candidate row and source-window row before any normalization-facing consumer reaches the frontier.

Next use the residual-frontier route and normal-form witness extraction, Definition 765.214 and Definition 765.215. They force the residual boundary and extracted witness to remain attached to the same finite carrier rows.

Finally apply Definition 765.216. Fairness retention keeps the candidate, confluence-boundary, finite-observation, critical-pair, discharge-socket, and source rows visible when the bounded request is consumed. The rows H, C, P, L then transport, replay, source, and name the same finite route, so the discharge cannot become global SN, full confluence, subject reduction, beta-decidability, quotient equality, or a selected normal-form principle. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNBoundedDischargeRoute`

Theorem 765.218 (MetaCIC candidate-mediated SN budget extraction)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any bounded strong-normalization discharge read that extracts a finite normalization budget must expose the candidate row, the source-window row, the residual-frontier row, the normal-form witness row, the fairness-retention row, and the local* NameCert *rows before the budget is readable. The extracted budget is therefore a finite row-level budget for the displayed candidate-mediated route; it is not a global strong-normalization theorem, a selected normal form, a quotient-equality principle, a full confluence theorem, or an ambient subject-reduction closure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNBudgetExtraction`

Proof. Start from the bounded discharge route of Definition 765.217. It already forces the candidate row, source-window row, residual-frontier row, fairness-retention row, normal-form witness extraction row, transport, replay, provenance, and local naming to be visible before a bounded request can be discharged.

Use the normal-form witness extraction theorem, Definition 765.215, to identify the witness row as boundary data attached to the same candidate, confluence-boundary, finite-observation, critical-pair, discharge-socket, and L10 source faces. This prevents the budget from being read as a selected normal form detached from the packet.

Finally apply the fairness-retention theorem, Definition 765.216. It keeps the candidate and source rows visible when the frontier is consumed, while H, C, P, L transport, replay, source, and name the same finite route. The budget extracted from the discharge is therefore only the finite budget certified by those rows, and it cannot expand into global SN, full confluence, subject reduction, quotient equality, or an ambient normal-form selector. □

Theorem 765.219 (MetaCIC candidate-mediated SN normal-endpoint budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any closed normal endpoint consumed by the critical-path route must expose, in order, candidate evidence, the closed typed term row, the candidate-to-SN projection, the no-infinite-reduction projection, and the finished-normal endpoint row. This endpoint budget preserves the candidate premise as a visible row; it does not assert evidence-free strong normalization, closed consistency, global confluence, quotient term equality, host term equality, or a selected normal form detached from the candidate-mediated route.*

Proof. Begin with the candidate-mediated frontier of Definition 765.213. It places every strong-normalization-facing read behind the candidate row and the surrounding confluence-boundary, finite-observation, critical-pair, discharge-socket, and L10 source rows.

Use the bounded discharge route and budget extraction, Definition 765.217 and Definition 765.218. These force a bounded request to keep the candidate, source-window, residual-frontier, witness, fairness-retention, transport, replay, provenance, and local naming rows visible before any finite budget is read.

Finally read the normalization audit map through the existing MetaCIC frontier references, including Definition 765.215 and Definition 765.216. The closed typed term row, candidate-to-SN projection, no-infinite-reduction projection, and finished-normal endpoint row are therefore endpoint data attached to the candidate evidence, not an unconditional normalization or consistency theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNNormalEndpointBudget`

Theorem 765.220 (MetaCIC candidate frontier confluence socket boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A confluence-facing consumer may use the handoff row U after candidate-mediated strong-normalization evidence only while the obstruction row O and discharge socket D remain visible, and only after the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

have been displayed. The boundary exports no closed subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or discharged subject-reduction socket.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierConfluenceSocketBoundary`

Proof. Start with the candidate-mediated frontier theorem and handoff totality, Definition 765.198 and Definition 765.206. They place every candidate-mediated handoff on S, N, U, H, C, P, L together with the visible rows O, D and the four L10 faces.

Use the L10 readback and Real source lock theorems, Definition 765.200 and Definition 765.201. These force the dyadic, stream, regular-rational, and Real rows to remain explicit before U is consumed by a confluence-facing or decidability-facing route.

Now apply socket separation and obstruction nonescape, Definition 765.199 and Definition 765.207. They keep O and D visible while the handoff is replayed and prevent candidate-mediated strong-normalization evidence from filling the subject-reduction socket.

Finally read the discharge coverage route in Definition 765.210. Since the available surface is exactly the finite frontier plus the L10 faces, no closed subject reduction, global confluence, beta-decidability, quotient equality, selected limit, host equality, ambient Real completeness, or discharged socket can be exported. □

Theorem 765.221 (MetaCIC candidate frontier discharge socket coproduct)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated handoff that reaches the confluence or decidability boundary consumes exactly the finite coproduct of candidate rows S, N, U, H, C, P, L, visible obstruction and socket rows O, D, and L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The coproduct boundary exports no closed subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or discharged subject-reduction socket unless a separate row for that claim is displayed.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierDischargeSocketCoproduct`

Proof. Start with the candidate-mediated frontier and L10 readback theorems, Definition 765.198 and Definition 765.200. They force the candidate rows S, N, U, H, C, P, L and the four L10 faces to be visible before the handoff reaches a confluence-facing or decidability-facing consumer.

Use handoff totality and obstruction nonescape, Definition 765.206 and Definition 765.207. These keep O, D as visible boundary rows while the candidate handoff is consumed, so neither row is silently absorbed into the candidate rows.

Apply the confluence socket boundary, Definition 765.220. It identifies the consumer-facing surface as the displayed candidate route plus the socket boundary and the L10 source faces.

Finally read discharge coverage, Definition 765.210. Once those rows have been consumed, no remaining coordinate can supply closed subject reduction, global confluence, beta-decidability, quotient equality, selected limits, host equality, ambient Real completeness, or socket discharge. The handoff boundary is therefore exactly the stated finite coproduct. □

Theorem 765.222 (MetaCIC candidate frontier discharge refusal)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated handoff that has exposed the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

still refuses discharge of the subject-reduction socket unless a separate closed subject-reduction row is displayed. The same handoff refuses global confluence, beta-decidability, quotient term equality, host equality, selected limits, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierDischargeRefusal`

Proof. Start with the frontier and L10 readback theorems, Definition 765.198 and Definition 765.200. They place the candidate-mediated handoff on S, N, U, H, C, P, L and force the four L10 faces to be displayed before any confluence-facing or decidability-facing consumer reads U.

Use obstruction nonescape and discharge coverage, Definition 765.207 and Definition 765.210. These keep O and D visible while the handoff surface is consumed and identify the whole available discharge budget as the finite frontier together with the L10 faces.

Finally apply the confluence socket boundary, Definition 765.220. None of the displayed rows is a closed subject-reduction coordinate, so the socket is not discharged. The same absence excludes global confluence, beta-decidability, quotient term equality, host equality, selected limits, and $\mathsf{Real}^{\uparrow}$ completeness from the exported handoff. □

Theorem 765.223 (MetaCIC SN confluence decidability handshake)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If a closed-normal-form membership read first uses candidate-mediated strong normalization and then passes to the confluence-decidability handoff, the whole read factors through* S, N, U, H, C, P, L *together with the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The obstruction row O *and discharge socket* D *remain visible and unfilled unless the same packet also displays a separate closed subject-reduction row.*

Proof. Begin with the candidate-mediated frontier. By Definition 765.198, the strong-normalization read is confined to the displayed frontier rows and the four L10 faces before any confluence-facing or decidability-facing consumer can inspect the handoff row U.

Next apply Definition 765.224. The consumer may use the existing handoff row and route closed-normal-form membership through C, P, L, but it receives no closed subject-reduction coordinate and no global beta-conversion decision procedure.

The discharge boundary is preserved by Definition 765.225 and Definition 765.226. These results keep O and D visible while finite frontier rows are replayed and prevent the SN evidence from filling the socket by itself.

Finally use Definition 765.230. The exhausted read surface is exactly the displayed S, N, U, H, C, P, L route plus $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Hence the handshake factors through those rows and leaves O, D open unless a separate closed subject-reduction row is present. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSNConfluenceDecidabilityHandshake`

Theorem 765.224 (MetaCIC confluence-decidability consumer boundary)**.** *Any confluence-facing or decidability-facing consumer that uses candidate-mediated strong-normalization evidence from* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *must consume only the frontier rows of Definition 765.197 and the existing handoff row U. The consumer may route closed-normal-form membership through C, P, L, but it may not treat the frontier as closed subject reduction, full consistency, arbitrary Church–Rosser, global beta-conversion decidability, a discharged socket, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use Definition 765.198 to place the candidate-mediated read on the frontier rows and the four local faces. This supplies the SN-facing evidence that the confluence-decidability handoff is allowed to inspect.

Now apply the MetaCIC SN confluence-decidability handoff, Definition 765.196. It sends closed-normal-form membership decisions through the strong-normalization, confluence, and decidability budget rows and then through C, P, L before the open-phase ledger is exported.

Finally keep the socket and obstruction boundaries visible. The frontier does not fill D, and the handoff theorem does not erase O or convert U into a global theorem. Therefore the consumer obtains only the finite candidate-mediated boundary plus the existing handoff route, while the excluded subject-reduction, consistency, Church–Rosser, beta-decidability, socket-discharge, and completeness claims remain outside the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceDecidabilityConsumerBoundary`

Theorem 765.225 (MetaCIC candidate-mediated SN discharge row)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated strong-normalization read satisfying the frontier of Definition 765.197 can be used by a confluence-facing or decidability-facing consumer only while D remains a displayed discharge socket separate from S, N, U, H, C, P, L and from the four local L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *No such consumer may reinterpret the frontier as closed subject reduction, arbitrary strong normalization, full consistency, beta-conversion decidability, quotient term equality, host equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNDischargeRow`

Proof. First apply the candidate-mediated SN frontier theorem, Definition 765.198. It places the read on S, N, U, D, H, C, P, L together with the four local L10 faces, while keeping the obstruction row O and discharge socket D available to the surrounding packet.

Next apply the consumer boundary theorem, Definition 765.224. A confluence-facing or decidability-facing consumer may inspect the same frontier rows and the handoff row U, but the theorem does not fill D or promote the frontier to a closed metatheoretic result.

Finally use the handoff exhaustion route of Definition 765.171. That route keeps downstream reads of U mediated by O, D, H, C, P, L and the finite packet rows S, N, U. Therefore D remains a separate displayed discharge row, distinct from the packet rows and the four L10 faces, and the excluded subject-reduction, normalization, consistency, decidability, quotient, host-equality, selected-limit, and completeness claims remain outside the consumer surface. □

Theorem 765.226 (MetaCIC discharge-socket induction)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet, and let F be a finite closed-term frontier whose rows replay the candidate-mediated strong-normalization row S, the normal-form endpoint N, the handoff row U, the componentwise transport row H, the displayed continuation row C, the provenance row P, and the local naming row L. If every row of F is consumed through the frontier of Definition 765.197, then induction over F preserves the obstruction row O and keeps the discharge socket D unfilled unless the packet also contains an explicit closed subject-reduction row. In particular, candidate-mediated strong normalization, confluence-facing replay, and decidability-facing replay do not by themselves introduce a closed subject-reduction coordinate.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketInduction`

Proof. Induct over the finite frontier rows. The empty frontier exposes no consumer row, so the accepted packet remains exactly $K = (S, N, O, U, D, H, C, P, L)$ with O and D still visible.

For the induction step, project the next closed-term row and the accumulated packet surface. The row is allowed to replay only the displayed S, N, U, H, C, P, L route named by Definition 765.197; it has no coordinate for closed subject reduction.

Use the candidate-mediated SN frontier theorem and the confluence-decidability consumer boundary to keep the replayed row on the same frontier surface. These results route the SN, confluence-facing, and decidability-facing reads through U and C while leaving the obstruction row O and the discharge socket D in the surrounding packet.

The inductive hypothesis applies to the remaining frontier. Since neither the current replay row nor the previous frontier rows add an explicit closed subject-reduction coordinate, no constructor fills D. Thus the obstruction row is preserved and the discharge socket stays open until a separate subject-reduction row is supplied. □

Theorem 765.227 (MetaCIC candidate-normalization budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every confluence-facing or decidability-facing use of candidate-mediated strong-normalization evidence is budgeted by the finite rows S, N, U, H, C, P, L together with the four L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The obstruction row O and discharge socket D remain visible boundary rows throughout the read. The budget exports no closed subject reduction, arbitrary strong normalization, full confluence, global beta-conversion decidability, quotient term equality, host equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the candidate-mediated SN frontier theorem, Definition 765.198. It places any candidate-mediated read on the finite rows S, N, U, D, H, C, P, L and the four L10 faces, with O and D still visible to the surrounding packet.

Next use the confluence-decidability consumer boundary, Definition 765.224. A confluence-facing or decidability-facing consumer may inspect the frontier and the handoff row U, but it receives no closed subject-reduction row, no full confluence theorem, and no global beta-conversion decision procedure.

Apply the candidate-mediated SN discharge row and the discharge-socket induction, Definition 765.225 and Definition 765.226. These results keep D separate from the budget rows and preserve O while finite frontier rows are consumed. Since the packet has no coordinate for quotient term equality, host equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, the displayed budget is the whole normalization surface available to the consumer. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateNormalizationBudget`

Theorem 765.228 (MetaCIC candidate-normalization L10 source exactness)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

every candidate-normalization consumer whose budget is certified by Definition 765.227 reads exactly the four L10 faces

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

together with the finite packet rows S, N, U, H, C, P, L. The obstruction row O and the discharge socket D remain visible throughout the read. The source-exact budget exports no closed subject reduction, global confluence, beta-conversion decidability, selected limit, quotient equality, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateNormalizationL10SourceExactness`

Proof. Start with the candidate-mediated SN frontier, Definition 765.198. It places the consumer on the displayed strong-normalization rows and the four L10 faces, while keeping the obstruction row and discharge socket in the surrounding packet.

Use the confluence-decidability consumer boundary, Definition 765.224. A confluence-facing or decidability-facing replay may inspect U and the displayed frontier, but it receives no global confluence theorem, no beta-conversion decision procedure, and no closed subject-reduction row.

Apply discharge-socket induction, Definition 765.226. Induction over the finite frontier preserves O and leaves D unfilled unless a separate closed subject-reduction coordinate is displayed.

Finally use candidate-normalization socket exhaustion, Definition 765.230. Once S, N, U, H, C, P, L and the four L10 faces have been consumed, no packet row remains from which to read selected limits, quotient equality, host equality, Real completeness, closed subject reduction, global confluence, or beta-decidability. The budget is therefore source-exact for exactly the named L10 route. □

Theorem 765.229 (MetaCIC L10 normalization frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization discharge route that displays the Real source lock, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the confluence socket, and the decidable-boundary row remains*

on the finite L10 normalization frontier before any consistency-facing claim is exported. The frontier is the displayed route through $S, N, O, U, D, H, C, P, L$ and the four L10 faces; it is not full subject reduction, full confluence, global beta-conversion decidability, quotient term equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10NormalizationFrontier`

Proof. First read the candidate-mediated frontier L10 readback and Real source lock, Definition 765.200 and Definition 765.201. These force the Real-facing route to expose the dyadic, stream, regular-rational, and Real rows before the handoff row is consumed.

Next apply the RegSeqRat handoff theorem, Definition 765.203. It keeps the regular-rational readback on the same finite StreamName and DyadicRatCore route.

Use the SN confluence-decidability handshake and consumer boundary, Definition 765.223 and Definition 765.224. These allow the handoff to inspect the candidate-mediated frontier only while the confluence socket and decidable-boundary row remain explicit.

Finally apply discharge-socket induction and discharge coverage, Definition 765.226 and Definition 765.234. They preserve the discharge socket unless a separate subject-reduction coordinate is displayed, so the route remains an L10 normalization frontier and exports no consistency-facing claim beyond the finite packet. □

Theorem 765.230 (MetaCIC candidate-normalization socket exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any confluence-facing or decidability-facing replay that uses only the candidate-normalization budget rows S, N, U, H, C, P, L together with the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *exhausts the SN-facing surface without filling the subject-reduction socket. The obstruction row O remains visible and the discharge socket D remains unfilled unless the same packet also displays an explicit closed subject-reduction row.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateNormalizationSocketExhaustion`

Proof. Start from the candidate-mediated SN frontier theorem, Definition 765.198. It places the candidate-mediated read on the finite frontier and keeps O and D available to the surrounding packet.

Next apply the confluence-decidability consumer boundary, Definition 765.224. A confluence-facing or decidability-facing consumer may inspect the frontier rows and the handoff row U, but the boundary gives no closed subject-reduction coordinate.

Use the candidate-mediated SN discharge row and discharge-socket induction, Definition 765.225 and Definition 765.226. These two results preserve the separation between the replay rows S, N, U, H, C, P, L and the socket row D while finite frontier rows are consumed.

Finally apply the candidate-normalization budget theorem, Definition 765.227. Its budget is exactly the displayed SN-facing surface plus the four L10 faces. Since that inventory omits a closed subject-reduction row, the replay can feed only the permitted consumers; it cannot erase O, fill D, or export full confluence or global beta-conversion decidability. □

Theorem 765.231 (MetaCIC candidate-mediated subject-reduction boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated replay over* S, N, U, H, C, P, L *and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

preserves the visible obstruction row O *and the discharge socket* D*. The replay cannot be read as closed subject reduction unless the packet separately displays a subject-reduction row; candidate-mediated strong normalization, confluence-facing replay, and decidability-facing replay do not fill that row by themselves.*

Proof. Begin with the candidate-mediated frontier of Definition 765.198. It confines the SN-facing read to the displayed frontier rows and keeps O, D visible to the enclosing packet.

Use the handshake theorem, Definition 765.223, to pass from candidate-mediated normalization to confluence-facing or decidability-facing replay. The handoff reads S, N, U, H, C, P, L and the four L10 faces, but it does not add a closed subject-reduction coordinate.

Apply the discharge-row and socket-induction theorems, Definition 765.225 and Definition 765.226. They preserve the obstruction row and keep the discharge socket unfilled while finite frontier rows are consumed.

Finally use Definition 765.227 and Definition 765.230. The budget is exhausted by the displayed replay route and the four source faces, so any closed subject-reduction conclusion must come from a separate displayed row. Without that row, the boundary remains exactly O, D plus the finite replay surface. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNSubjectReductionBounda`

Theorem 765.232 (MetaCIC candidate SN boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Candidate-mediated strong-normalization evidence over* S, N, U, H, C, P, L *and the L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

cannot be consumed as a closed subject-reduction row unless K *separately displays a subject-reduction coordinate. The candidate-mediated evidence may feed the frontier and the confluence-decidability handoff, but it does not identify reduction preservation for arbitrary closed terms, fill the discharge socket* D*, or erase the obstruction row* O*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNBoundary`

Proof. First apply the candidate-mediated SN frontier, Definition 765.198. It places the evidence on the finite rows S, N, U, H, C, P, L together with the four L10 faces before any handoff consumer reads U.

Next use the subject-reduction boundary of Definition 765.231. This theorem states the exact missing coordinate: closed subject reduction is available only when the packet displays a separate subject-reduction row.

The discharge side is controlled by Definition 765.226. Induction over the displayed replay rows preserves the socket as a socket, so the candidate-mediated evidence cannot become a subject-reduction row by finite consumption.

Finally invoke candidate-normalization socket exhaustion, Definition 765.230. The exhausted surface is the candidate-normalization route plus the four L10 faces, with no additional coordinate for closed subject reduction. Hence any consumer that claims such a row must cite it separately, and the candidate-mediated boundary remains distinct from closed subject reduction. □

Theorem 765.233 (MetaCIC candidate-mediated SN closed-normal-form readback)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any closed-normal-form readback that consumes candidate-mediated strong-normalization evidence factors through the finite rows* S, N, U, H, C, P, L *and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The obstruction row O *and the discharge socket* D *remain visible to the readback unless the packet separately displays a subject-reduction row. Thus closed-normal-form readback from candidate-mediated SN is a finite frontier read, not a proof of closed subject reduction, global confluence, or beta-conversion decidability.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNClosedNormalFormReadba`

Proof. Start with the candidate-mediated SN frontier, Definition 765.198. It confines candidate-mediated normalization evidence to the displayed finite rows and the four L10 faces before any readback consumer may inspect the handoff row U.

Use the handshake theorem, Definition 765.223. The readback may feed confluence-facing or decidability-facing replay through S, N, U, H, C, P, L, but that handoff does not add a subject-reduction row.

Apply discharge-socket induction, Definition 765.226. Finite replay preserves the socket boundary while the closed-normal-form consumer moves through the displayed route.

Finally use socket exhaustion and the candidate SN boundary, Definition 765.230 and Definition 765.232. Together they leave exactly O and D as visible boundaries unless an explicit subject-reduction row is displayed, so the readback cannot be upgraded to closed subject reduction, global confluence, or beta-decidability. □

Theorem 765.234 (MetaCIC candidate-mediated SN frontier discharge coverage)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every finite consumer replay through rows* S, N, U, H, C, P, L *and the L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

either preserves the displayed discharge socket D *or cites an explicit subject-reduction row outside the candidate-mediated frontier. Candidate-mediated strong normalization, confluence-facing replay, decidability-facing replay, and L10 readback do not by themselves discharge* D*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierDischargeCover`

Proof. Start with the candidate-mediated SN frontier theorem, Definition 765.198. It places the replay on S, N, U, D, H, C, P, L together with the four L10 faces and keeps the obstruction and discharge rows visible.

Apply the discharge-row theorem and discharge-socket induction, Definition 765.225 and Definition 765.226. These results preserve the socket while each finite consumer row is replayed.

Use the candidate-normalization budget and socket-exhaustion theorems, Definition 765.227 and Definition 765.230. They show that the consumer surface is exhausted by S, N, U, H, C, P, L and the four L10 faces, with no hidden subject-reduction coordinate.

Finally invoke the subject-reduction boundary, Definition 765.231. It is exactly the alternative in the statement: either the replay keeps D displayed, or a separate subject-reduction row outside the candidate-mediated frontier must be cited. □

Theorem 765.235 (MetaCIC typed-reduction candidate SN handoff)**.** *An accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer may pass typed-reduction candidate strong-normalization evidence toward confluence or decidability only after displaying the candidate row S, the normal-form endpoint row N, the subject-reduction discharge socket D, the handoff row U, the bounded checker budget carried by the factorization route, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The handoff remains conditional: it exports no closed subject reduction, global confluence, beta-conversion decision, quotient term equality, selected limit, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathTypedReductionCandidateSNHandoff`

Proof. Start with Definition 765.234. Any finite consumer replay through the candidate-mediated frontier either preserves the discharge socket D or cites a separate subject-reduction row outside the frontier.

Apply Definition 765.263. The bounded checker budget separates the candidate row, SN endpoint rows, open socket row, and four L10 face row before the checker budget can be read.

Finally use Definition 765.260 together with Definition 765.240. These keep the checker-facing and discharge-facing budgets conditional on the open socket row, so the refused global MetaCIC, quotient, selected-limit, host-equality, and Real-completeness claims remain outside the handoff. □

Definition 765.236 (MetaCIC frontier budget factorization)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

the candidate-frontier budget factorization is the displayed packet

$$\mathcal{F}(K) = (S, N, U, D, H, C, P, L; \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow})$$

read in five layers: the candidate-normalization budget rows S, N, the confluence handoff row U, the subject-reduction discharge socket D, the L10 entry budget $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, and the closed-normal readback rows H, C, P, L. A consumer is factorized through $\mathcal{F}(K)$ exactly when its frontier read exposes those five layers before it exports any strong-normalization, confluence-facing, decidability-facing, Phase Real, or Completion-facing conclusion.*

Lean4 (definition) `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierBudgetFactorization`

Theorem 765.237 (MetaCIC candidate frontier L10 budget lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated strong-normalization discharge that reaches an L10 readback must pass through the factorization packet* $\mathcal{F}(K)$ *of Definition 765.236. In particular, the route first exposes the candidate-normalization rows* S, N*, the confluence handoff row* U*, and the discharge socket* D*; then it reads* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as the L10 entry budget; only then may it consume* H, C, P, L *as closed-normal readback rows.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierL10BudgetLock`

Proof. Start with discharge coverage, Definition 765.234. It says that any finite consumer replay through the candidate-mediated frontier either preserves the socket D or cites a separate subject-reduction row outside the frontier.

Next apply the candidate-normalization budget theorem, Definition 765.227. It identifies the candidate-facing surface as the rows S, N, U, H, C, P, L together with the four L10 faces, while keeping the obstruction row and discharge socket visible.

Use the closed-normal-form readback and L10 readback theorems, Definition 765.233 and Definition 765.241. These force every readback that reaches Phase Real or Completion data to expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the exported rows are consumed.

The five layers listed in Definition 765.236 are exactly the rows preserved by those three results. Hence any L10-reaching discharge has already passed through $\mathcal{F}(K)$, and no alternate route can bypass the displayed candidate, handoff, discharge, L10, or readback layers. □

Theorem 765.238 (MetaCIC normalization budget nonescape)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. No route from the candidate-mediated frontier exports strong normalization, confluence, beta-conversion decidability, or Real-completion entry unless it displays the candidate rows* S, N*, the handoff row* U*, the discharge socket* D*, the L10 entry budget* $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, and the closed-normal readback rows* H, C, P, L *as the factorization packet* $\mathcal{F}(K)$*. The route exports no closed subject reduction, quotient term equality, host equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency beyond that packet.*

Proof. First read the candidate frontier L10 budget lock, Definition 765.237. Any discharge that reaches L10 readback is already forced through the displayed factorization packet.

Next use socket exhaustion and subject-reduction boundary, Definition 765.230 and Definition 765.231. These keep D visible unless a separate subject-reduction row is cited, so the candidate budget cannot become closed subject reduction by replay alone.

Apply the candidate-mediated SN route and normal-form discharge theorems, Definition 765.239 and Definition 765.242. They confine strong-normalization, confluence-facing, decidability-facing, and normal-form exports to the finite rows in $\mathcal{F}(K)$.

Since quotient term equality, host equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, and kernel consistency are not coordinates of $\mathcal{F}(K)$, no route that only reads the candidate-mediated frontier can export them. Real-completion entry is therefore a budgeted L10 readback, not a global metatheoretic escape. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationBudgetNonescape`

Theorem 765.239 (MetaCIC candidate-mediated SN route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization evidence consumed by confluence-facing or decidability-facing replay is routed through* S, N, U, H, C, P, L*, the visible obstruction row* O*, the discharge socket* D*, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

A separate subject-reduction row is required before D *can be filled. Candidate-mediated normalization evidence alone cannot assert global confluence, closed subject reduction, or beta-conversion decidability.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNRoute`

Proof. Begin with the candidate-mediated SN frontier, Definition 765.198. It places normalization evidence on the finite route through S, N, U, H, C, P, L while keeping O and D visible to any consumer.

Apply the candidate-normalization budget, Definition 765.227. The budget available to confluence-facing and decidability-facing replay is exhausted by the displayed SN route and the four L10 faces.

Next use socket exhaustion, Definition 765.230. Since exhaustion still leaves D as a socket rather than a filled row, the replay cannot close subject reduction by consuming candidate-mediated SN evidence alone.

Finally invoke discharge coverage, Definition 765.234. Every admissible consumer either preserves D or cites a separate subject-reduction row. Therefore the route exports only the finite candidate-mediated handoff and refuses global confluence, closed subject reduction, and beta-decidability claims. □

Theorem 765.240 (MetaCIC candidate-mediated SN discharge budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated strong-normalization consumer that is allowed to pass a consistency-facing handoff must first display the normalization candidate row* S*, the confluence boundary carried by the handoff row* U*, the decidability boundary carried by the replay row* C*, the discharge socket* D*, the obstruction row* O*, componentwise transport* H*, replay* C*, provenance* P*, and the local* NameCert *row* L*. The consumer may then read the handoff only as the displayed finite discharge budget; it does not export full subject reduction, full confluence, global beta-conversion decidability, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNDischargeBudget`

Proof. Begin with the candidate-mediated SN frontier, Definition 765.198. It forces every SN read that reaches the handoff row to expose the finite packet rows and the four L10 faces before a confluence-facing or decidability-facing consumer can inspect U.

Next apply the socket-separation and discharge-row theorems, Definition 765.199 and Definition 765.225. These keep O and D visible while the normalization candidate and handoff rows are consumed.

Use discharge-socket induction and discharge coverage, Definition 765.226 and Definition 765.234. They show that any finite consumer replay either preserves the socket or cites a separate subject-reduction row outside the candidate-mediated frontier.

Finally read the candidate SN route of Definition 765.239. The route exports only the displayed budget rows $S, N, O, U, D, H, C, P, L$ and therefore cannot supply the excluded global MetaCIC or quotient-equality claims. □

Theorem 765.241 (MetaCIC candidate-mediated SN L10 readback)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any L10-facing read of the candidate-mediated SN frontier routes through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the subject-reduction discharge socket D before it exports a Phase Real row or a Completion budget row. The exported rows are the finite readback surface of the same packet; they are not full subject reduction, full confluence, global beta-conversion decidability, quotient term equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNL10Readback`

Proof. Start from the existing L10 readback theorem, Definition 765.200. It requires $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to be exposed before the handoff row U is consumed.

Apply the closed-normal-form readback theorem and candidate-normalization budget, Definition 765.233 and Definition 765.227. These identify the Phase Real and Completion-facing budget as the finite surface S, N, U, H, C, P, L together with the four L10 faces.

Next use the subject-reduction boundary, Definition 765.231. It keeps the discharge socket D visible unless a separate subject-reduction row is displayed.

Finally combine socket exhaustion with Definition 765.240. Once the L10 route and candidate-mediated rows have been read, the packet has no remaining coordinate for full subject reduction, global confluence, beta-conversion decidability, quotient term equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or kernel consistency. Thus the Phase Real and Completion rows are budget rows, not global metatheoretic exports. □

Theorem 765.242 (MetaCIC candidate-mediated normal-form discharge)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A candidate-mediated strong-normalization consumer can export a normal-form discharge only after displaying the normalization candidate row S, the closed normal-form readback row N, the subject-reduction discharge socket D, the confluence boundary carried by U, the decidability boundary carried by C, the L10 rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row L. The exported discharge is exactly this finite normal-form boundary; it is not full subject reduction, full confluence, global beta-conversion decidability, quotient term equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierNormalFormDischa`

Proof. First read the candidate-mediated SN frontier and the discharge budget, Definition 765.198 and Definition 765.240. These force the normalization candidate, handoff boundary, decidability replay, socket, obstruction, transport, replay, provenance, and local naming rows to be visible before a consistency-facing consumer can inspect the handoff.

Next apply the closed-normal-form readback and L10 readback theorems, Definition 765.233 and Definition 765.241. They attach the normal-form endpoint to S, N, U, H, C, P, L and require the four rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any Phase Real or Completion budget is exported.

Finally use socket separation and discharge coverage, Definition 765.199 and Definition 765.234. These keep the subject-reduction socket visible unless a separate subject-reduction row is cited, so the consumer may export only the finite normal-form discharge surface stated above. The excluded global MetaCIC and quotient-equality claims are not coordinates of that surface. □

Theorem 765.243 (MetaCIC subject-reduction socket before normal-form discharge consumers)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization handoff that reaches the normal-form discharge surface of Definition 765.242 must still expose the subject-reduction socket row D as a separate row before a closed subject-reduction consumer may proceed. The handoff may use the candidate row S, the closed normal-form readback row N, the handoff row U, componentwise* hsame *transport H, displayed* Cont *replay C, provenance P, local naming L, and the L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *but it cannot treat that surface as global confluence, beta-decidability, quotient term equality, selected limits, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a closed subject-reduction row.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierSubjectReduction`

Proof. Start from the candidate-mediated frontier and the candidate SN boundary, Definition 765.198 and Definition 765.232. They place the SN handoff on the displayed rows S, N, U, H, C, P, L together with the four L10 faces while keeping O and D visible.

Next apply closed-normal-form readback and normal-form discharge, Definition 765.233 and Definition 765.242. These results allow the consumer to reach the finite normal-form discharge surface, but that surface still lists D as a subject-reduction socket rather than as a discharged row.

Finally use socket exhaustion, Definition 765.230, together with socket separation in Definition 765.199. Once the candidate-mediated rows and L10 faces have been consumed, no hidden coordinate remains for closed subject reduction. A closed subject-reduction consumer must therefore cite a separate subject-reduction row before proceeding, and the refused confluence, beta-decidability, quotient-equality, selected-limit, host-equality, and Real-completeness claims remain outside the packet. □

Theorem 765.244 (MetaCIC subject-reduction socket readiness)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A subject-reduction consumer may inspect the normal-form discharge only after the candidate row S, normal-form row N, handoff row U, subject-reduction socket D, and the L10 rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

are all visible. Candidate-mediated strong normalization by itself is readiness data for the socket; it is not a closed subject-reduction proof, not full confluence, not beta-conversion decidability, not quotient term equality, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the subject-reduction socket theorem Definition 765.243. It keeps D separate from S, N, U, H, C, P, L and requires a closed subject-reduction consumer to cite a separate row before proceeding.

Next read the normal-form discharge theorem Definition 765.242. Its discharge surface lists the candidate row, the normal-form endpoint, the handoff row, the socket, the structural rows, and the four L10 faces as the finite boundary visible to the consumer.

Apply the L10 readback theorem Definition 765.241 and the dependency seal Definition 765.245. These force $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to be displayed before Phase Real, Completion, or confluence-facing inspection of the normal-form boundary.

Finally use socket exhaustion and discharge coverage, Definition 765.230 and Definition 765.234. They leave no hidden coordinate that can turn candidate-mediated SN into closed subject reduction, so the visible rows are readiness data only. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubjectReductionSocketReadiness`

Theorem 765.245 (MetaCIC candidate SN L10 dependency seal)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated strong-normalization packet whose discharge reads Real-facing evidence must route through the displayed dependency rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

before any MetaCIC confluence consumer can inspect the normal-form boundary. The inspection exports only the candidate-mediated frontier and its finite L10 seal; it does not export full subject reduction, full confluence, global beta-conversion decidability, quotient term equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNL10DependencySeal`

Proof. Start from the candidate-mediated frontier definition and Definition 765.242. They identify the normal-form discharge boundary as the finite rows S, N, U, D, H, C, P, L together with the four L10 dependency rows.

Apply the L10 readback theorem Definition 765.241. It requires $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to be displayed before a Phase Real or Completion-facing row can be exported.

Next use the RegSeqRat handoff theorem Definition 765.203. The regular rational route is the unique readback path available to the candidate-mediated handoff, so no alternate Real-facing route can bypass the four-object seal.

Finally apply the confluence-decidability consumer boundary, Definition 765.224. A confluence consumer may inspect only the displayed frontier after the L10 route has been exposed; the excluded global MetaCIC, quotient, completeness, and consistency claims are not packet coordinates. □

Theorem 765.246 (MetaCIC candidate-mediated frontier subject-reduction row absence)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. The candidate-mediated strong-normalization frontier may expose the candidate row* S*, closed normal-form row* N*, handoff row* U*, componentwise transport row* H*, continuation row* C*, provenance row* P*, local naming row* L*, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The obstruction row O *and discharge socket* D *remain visible throughout that route, but no closed subject-reduction row is present until a separate SubjectReduction discharge packet is cited.*

Proof. Start from the candidate-mediated SN route and the frontier normal-form discharge, Definition 765.239 and Definition 765.242. They identify the exposed frontier rows as S, N, U, H, C, P, L together with the four L10 faces, while still listing the socket data of K.

Next apply socket separation and socket exhaustion, Definition 765.199 and Definition 765.230. These keep O and D visible as obstruction and discharge-socket rows rather than as discharged subject-reduction evidence.

Use the subject-reduction socket theorem and readiness theorem, Definition 765.243 and Definition 765.244. They permit a subject-reduction consumer to inspect the normal-form discharge only as readiness data unless a separate subject-reduction row is cited.

Therefore the candidate-mediated frontier has no coordinate from which to read closed subject reduction. A later SubjectReduction discharge packet may fill the socket, but the frontier itself exports only the listed rows, the visible obstruction, and the visible socket. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedFrontierSubjectReductio`

Theorem 765.247 (MetaCIC confluence residual frontier route)**.** *Every confluence residual frontier read of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet consumes the same candidate-mediated strong-normalization row and the same L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Consequently residual substitution and critical-pair reads cannot bypass the finite four-object seal before reading the normal-form boundary or the confluence-decidability handoff.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceResidualFrontierRoute`

Proof. Read the candidate-mediated SN frontier theorem together with Definition 765.242. Both place the candidate row, normal-form endpoint, discharge socket, handoff boundary, and four L10 rows on the same finite frontier.

Use Definition 765.241 and Definition 765.203. These force any Real-facing or Completion-facing residual read to pass through the displayed dyadic, stream, regular-rational, and Real rows.

Finally apply the confluence-decidability consumer boundary, Definition 765.224. Residual substitution and critical-pair consumers can read the handoff only after the candidate SN frontier and L10 route are present, so no residual route escapes to full confluence, closed subject reduction, beta-decidability, quotient equality, selected limits, or ambient Real completeness. □

Theorem 765.248 (MetaCIC candidate frontier residual split route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization discharge that reaches a confluence residual consumer must factor through the residual-substitution boundary, the candidate join row, the closed-term substitution boundary carried by the normal-form readback row, and the L10 source locks*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The route may expose S, N, U, D, H, C, P, L *as the finite discharge surface, but it cannot bypass the residual boundary, erase the subject-reduction socket, or export full confluence, closed subject reduction, beta-decidability, quotient term equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierSplitRoute`

Proof. Start with the candidate-mediated SN frontier and normal-form discharge theorems, Definition 765.198 and Definition 765.242. They expose the candidate row, normal-form readback row, handoff row, socket row, structural rows, and L10 faces before any discharge consumer can proceed.

Next read the confluence residual frontier route above. It forces residual substitution and critical-pair consumers to consume the same candidate-mediated strong-normalization row and the same four L10 source locks before the handoff can be inspected.

Now apply subject-reduction socket readiness and the candidate SN L10 dependency seal, Definition 765.244 and Definition 765.245. These keep D visible as a socket and keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ locked before the residual consumer reads the frontier.

Finally use the closed-term substitution and RegSeqRat handoff boundaries already cited in the frontier, Definition 765.203 and Definition 765.224. They leave the discharge as constructor and route inspection over the displayed rows, not as source-equivalence transport or a global confluence proof. □

Theorem 765.249 (MetaCIC SN discharge frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A candidate-mediated strong-normalization route may hand a normal-form consumer to the discharge frontier only while the subject-reduction socket* D *and obstruction row* O *remain visible. The confluence residual route, decidability handoff, and L10 readback rows may transport* S, N, U, H, C, P, L*, but they do not close subject reduction, fill* D*, erase* O*, or export global confluence, beta-conversion decidability, quotient term equality, host equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Proof. Start from the candidate SN boundary and the normal-form discharge theorem, Definition 765.232 and Definition 765.242. They place the normal-form consumer on the finite rows S, N, U, D, H, C, P, L together with the L10 faces.

Next use subject-reduction socket readiness, Definition 765.244. It permits inspection of the normal-form discharge only after D is displayed as a socket, not as a closed subject-reduction proof.

Finally apply the confluence residual frontier route, Definition 765.247. Confluence, decidability, and L10 readback keep the same finite frontier visible, so they cannot discharge D or erase O without a separate closed subject-reduction row. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSNDischargeFrontier`

Theorem 765.250 (MetaCIC candidate-mediated SN frontier extension)**.** *Any candidate-mediated strong-normalization discharge used by the current L10 Real and Completion route must expose the candidate evidence row S, the closed-term substitution boundary carried by the normal-form readback row N, the finite observation confluence row U, the subject-reduction discharge socket D, and the four open-phase source rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before it is read as Real or Completion readiness. The exposed surface is conditional frontier evidence only; it is not unconditional strong normalization, not full confluence, not closed subject reduction, not beta-conversion decidability, not quotient term equality, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the candidate-mediated SN frontier and normal-form discharge theorems, Definition 765.198 and Definition 765.242. They put the candidate row, normal-form readback, handoff row, socket, transport, replay, provenance, local name, and L10 faces on the same finite boundary.

Next apply subject-reduction socket readiness and the candidate SN L10 dependency seal, Definition 765.244 and Definition 765.245. These force D and the four open-phase source rows to remain visible before any Real, Completion, or confluence-facing read proceeds.

Finally use the confluence residual frontier route, Definition 765.247. Residual and finite-observation reads cannot bypass the same four-object seal, so the discharge remains a conditional readiness surface rather than a global SN, confluence, subject-reduction, beta-decidability, quotient-equality, or Real-completeness claim. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierExtension`

Theorem 765.251 (MetaCIC candidate-mediated SN residual boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the closed-term candidate row S, the residual-substitution ledger, and the finite-observation confluence ledger agree on the same displayed* BHist *rows, then the candidate-mediated route preserves the strong-normalization frontier budget through N, U, D, H, C, P, L. The preserved budget may be consumed by the sibling routes of chapter 527, chapter 2181, chapter 2274, and chapter 2223, but it does not export closed subject reduction, full confluence, beta-conversion decidability, quotient term equality, host equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNResidualBoundary`

Proof. Start with the candidate-mediated SN frontier and residual frontier route, Definition 765.198 and Definition 765.247. They place the candidate row, normal-form endpoint, handoff row, socket row, structural rows, and four L10 faces on the same finite frontier before residual or confluence consumers inspect the handoff.

Next read the residual split route and SN discharge frontier, Definition 765.248 and Definition 765.249. The residual-substitution ledger and finite-observation confluence ledger can be consumed only while the same BHist rows, obstruction row O, and discharge socket D remain visible.

Finally apply the candidate-mediated frontier extension. Since the closed-term substitution boundary, residual-substitution boundary, candidate SN confluence boundary, and finite-observation confluence rows are all read as sibling budget rows over the same packet, the SN frontier budget is preserved through the candidate-mediated route and none of the excluded global MetaCIC claims is exported. □

Theorem 765.252 (MetaCIC candidate-mediated normalization budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, candidate-mediated strong-normalization evidence, confluence handoff data, and normalization-certificate routing expose a finite normalization budget before any critical-path discharge is exported. The budget is the displayed route through* $S, N, O, U, D, H, C, P, L$*, the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$*, the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *and* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *rows, the confluence boundary, and the visible discharge socket. It does not claim full MetaCIC normalization, global confluence, closed subject reduction, beta-conversion decidability, quotient term equality, selected normal forms, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the candidate-mediated SN frontier, Definition 765.198. It forces the normalization read through the packet rows and the four L10 faces before any handoff row is consumed.

Use the normalization-frontier dependency exposure of Definition 765.291. The $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ and $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ rows route the normal-form endpoint as finite certificate data rather than as a global normalization theorem.

Apply the confluence socket boundary, Definition 765.220. It keeps confluence-facing reads on the same finite frontier and leaves the obstruction and discharge socket visible.

Finally read discharge coverage, Definition 765.234. The discharge surface is available only as the displayed socket budget, so a critical-path consumer receives finite normalization readiness and none of the excluded global MetaCIC claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedNormalizationBudget`

Theorem 765.253 (MetaCIC candidate-mediated SN handoff)**.** *Let* $K = (S, N, O, U, D, H, C, P, L)$ *be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the packet displays a reducibility-candidate row* R*, a candidate-closedness row* Q*, and a beta-step replay row* B *over the same closed typed term data, then any critical-path consumer may export only the candidate-implies-SN handoff and the beta-closedness handoff. The export is the finite route*

$$R, Q, B \longmapsto S, N, U, H, C, P, L.$$

It is not a global strong-normalization theorem, not a confluence theorem, not beta-conversion decidability, not closed subject reduction, not quotient term equality, and not a filled discharge socket.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNHandoff`

Proof. Project the accepted packet first. The rows $S, N, O, U, D, H, C, P, L$ keep strong-normalization evidence, normal endpoints, obstruction data, the handoff row, the discharge socket, transport, replay, provenance, and local naming visible before any critical-path consumer reads the handoff.

Use the reducibility row R only through the candidate-implies-SN dependency represented by `BEDC.MetaCIC.Normalization.Reducibility.candidate_implies_sn`. This produces the displayed SN handoff into S, N, U for the same term and type data; it does not assert unrestricted normalization outside the candidate row.

Use the closedness and beta-step rows Q, B only through the beta-closedness dependency represented by `BEDC.MetaCIC.Normalization.Reducibility.candidate_closed_under_beta`. This replays the candidate row across the displayed beta step and returns to the same finite handoff surface.

Finally apply the existing candidate-mediated frontier and conditional-budget boundaries, Definition 765.198 and Definition 765.260. They keep O and D visible while the handoff is consumed, so the route exports only the candidate-implies-SN and beta-closedness handoffs and none of the refused global MetaCIC claims. □

Theorem 765.254 (MetaCIC candidate SN handoff source exhaustion)**.** *Let $K = (S, N, O, U, D, H, C, P, L)$ be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any accepted candidate-mediated strong-normalization handoff through the candidate row must exhaust exactly the source route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ O,\ D,\ H,\ C,\ P,\ L$$

before a confluence-facing or decidability-facing consumer may read the handoff row U. The route exports no closed subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limit, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNHandoffSourceExhaustion`

Proof. Begin with the candidate-mediated frontier theorem, Definition 765.198. It places the candidate-mediated read on the packet rows $S, N, O, U, D, H, C, P, L$ and requires the L10 source faces before the handoff row is consumed.

Read the source locks next. The L10 readback and Real source-lock theorems, Definition 765.200 and Definition 765.201, keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible as four displayed source rows.

Now use the RegSeqRat and dependency boundaries, Definition 765.203 and Definition 765.245. These fix the regular-rational handoff inside the same four-source route and prevent a consumer from reconstructing Real-facing evidence after reading U.

Finally apply the consumer boundary, Definition 765.224. Confluence-facing and decidability-facing consumers may inspect only the displayed frontier after the source, obstruction, socket, transport, replay, provenance, and local naming rows are present, so none of the refused global metatheoretic or completion claims is exported. □

Theorem 765.255 (MetaCIC candidate-mediated SN frontier exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated strong-normalization frontier read that may be consumed by the public* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *route must factor through the existing candidate-mediated frontier, residual-budget discharge surface, confluence-decidability handoff, and decidability budget rows before it is exported. Concretely, the read displays $S, N, O, U, D, H, C, P, L$, the residual-budget and socket rows of Definition 765.288, the handoff route of Definition 765.196, and the decidability budget of Definition 765.192; it cannot export full normalization, full confluence, global beta-conversion decidability, closed subject reduction, quotient term equality, host equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierExhaustion`

Proof. Start from the candidate-mediated SN frontier Definition 765.198. It places the strong-normalization read on the finite packet rows S, N, U, D, H, C, P, L together with the surrounding obstruction row O and the four L10 source faces.

Next attach the residual-budget discharge surface. The residual frontier and socket route, Definition 765.288 and Definition 765.289, require the residual-budget, substitution-boundary, closed-premise, local-diamond, socket, normalization, and audit-ledger rows before the discharge-facing consumer may proceed.

Now pass through the confluence and decidability side. The handoff theorem Definition 765.196 and the confluence-decidability budget Definition 765.192 keep every confluence-facing or decidability-facing read downstream of the same finite packet route and open L10 source ledger.

The public route receives only this assembled finite boundary: the candidate-mediated frontier, residual-budget discharge surface, handoff rows, decidability budget, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert data. Since none of the cited rows supplies a coordinate for full normalization, full confluence, global beta-conversion decidability, closed subject reduction, quotient equality, host equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or kernel consistency, no such claim can be exported by the frontier read. □

Theorem 765.256 (MetaCIC candidate-mediated SN frontier public scope)**.** *Let $K = (S, N, O, U, D, H, C, P, L)$ be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. The candidate-mediated strong-normalization frontier exhaustion of Definition 765.255 is public only when the same packet displays the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source rows together with the candidate-mediated frontier, residual-budget socket, confluence-decidability handoff, and decidability budget rows.*

Under that scope, the public read is the finite L10 source route plus the displayed MetaCIC boundary rows. It does not export full normalization, full confluence, beta-conversion decidability, host consistency, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierExhaustionScop`

Proof. Apply Definition 765.255. It already confines the candidate-mediated strong-normalization frontier to the displayed packet rows and to the residual-budget, handoff, and decidability-budget surfaces.

Now attach the public interface route. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source rows are the L10 four-object rows through which the public read may enter the MetaCIC packet. The residual-budget socket and public handoff rows keep that read in the same finite carrier rather than in an ambient normalization theorem.

Exhaust the displayed rows. The candidate-mediated frontier supplies only the frontier boundary, the residual-budget socket supplies only the budgeted discharge route, and the confluence-decidability handoff with its decidability budget supplies only the named finite decision surface.

Therefore the public export is scoped to the four L10 source rows and the displayed MetaCIC boundary rows. Full normalization, full confluence, beta-conversion decidability, host consistency, selected limits, quotient stream equality, and ambient Real completeness remain outside the packet. □

Theorem 765.257 (MetaCIC candidate-mediated SN obligation surface)**.** *Every confluence-facing or decidability-facing consumer that uses candidate-mediated strong-normalization evidence from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must first display the strong-normalization row S, the normal-form endpoint N, the handoff row U, the discharge socket D, the componentwise transport*

row H, the displayed continuation row C, the provenance row P, the local name L, and the four L10 faces

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

before it reads any confluence or decidability budget. The consumer may inspect only this candidate-mediated surface and the existing handoff budget; it receives no global confluence theorem, no beta-conversion decision procedure, no filled discharge socket, no quotient term equality, no host real equality, no selected limit, and no $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the candidate-mediated SN frontier, Definition 765.198. It confines the strong-normalization-facing read to the displayed frontier rows and requires the four L10 faces before any confluence-facing or decidability-facing consumer reads the handoff row.

Next apply the confluence-decidability budget, Definition 765.192. That budget keeps every confluence or decidability read downstream of the open-source ledger and the finite packet route, so the consumer cannot read U as an independent global theorem.

Finally use the consumer boundary, Definition 765.224. It allows closed-normal-form membership to pass through the handoff route while excluding closed subject reduction, full consistency, arbitrary Church–Rosser, global beta-conversion decidability, discharged sockets, quotient equality, host equality, selected limits, and $\mathsf{Real}^{\uparrow}$ completeness. Therefore the listed rows form the whole obligation surface available before any confluence or decidability budget is consumed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICConfluenceCandidateMediatedSNObligationSurface`

Theorem 765.258 (MetaCIC candidate-discharge schedule exhaustion)**.** *Any accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that attempts to discharge a candidate-mediated strong-normalization or confluence socket must expose the candidate row K, the strong-normalization row S, the normal-form endpoint N, the confluence-decidability handoff row U, the discharge socket D, the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

componentwise transport H, displayed replay C, provenance P, and the local-name row L before it reads a confluence or decidability budget. The discharge schedule exports no closed subject-reduction theorem, arbitrary Church–Rosser theorem, global beta-conversion decision procedure, filled discharge socket, quotient term equality, selected limit, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the candidate-mediated obligation surface, Definition 765.257. It already forces S, N, U, D, H, C, P, L and the four L10 faces to be displayed before any confluence-facing or decidability-facing budget can be read.

Now add the candidate row K as the schedule witness being consumed. The candidate-mediated frontier and source-ledger route keep this row attached to the same accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet rather than letting it appear as a detached normalization theorem.

Finally apply the confluence-decidability budget Definition 765.192 and the consumer boundary Definition 765.224. Those results allow the downstream budget read only after the displayed packet route is present and exclude closed subject reduction, full consistency, arbitrary Church–Rosser, global beta-conversion decidability, discharged sockets, quotient equality, host equality, selected limits, and $\mathsf{Real}^{\uparrow}$ completeness. Thus the listed rows exhaust the candidate-discharge schedule before any confluence or decidability budget is consumed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeScheduleExhaustion`

Theorem 765.259 (MetaCIC candidate-mediated SN discharge)**.** *Any accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that routes candidate-mediated strong-normalization evidence into a confluence-facing or decidability-facing budget must keep the discharge socket D explicitly displayed and unfilled throughout the read. The allowed route is the finite surface consisting of the candidate row, the strong-normalization row S, the normal-form endpoint N, the handoff row U, componentwise transport H, displayed replay C, provenance P, local naming L, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The route may feed the displayed confluence/decidability budget only as a conditional socket surface; it exports no closed subject-reduction row, no filled discharge socket, no global confluence theorem, no beta-conversion decision procedure, no quotient term equality, no selected limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNDischarge`

Proof. First apply the candidate-mediated obligation surface, Definition 765.257. It makes S, N, U, D, H, C, P, L and the four L10 faces visible before any downstream budget can be inspected, so the socket row D is part of the source surface rather than an inferred conclusion.

Next use the candidate-discharge schedule exhaustion theorem, Definition 765.258. This adds the candidate row and shows that every confluence-facing or decidability-facing read must consume the displayed schedule before touching the budget rows.

Finally apply the frontier discharge row, Definition 765.225. That theorem keeps D separate from the candidate-mediated SN evidence while the finite frontier is replayed. Since no row in the displayed route is a closed subject-reduction coordinate, the consumer receives only the conditional discharge surface and none of the excluded global MetaCIC, quotient, selected-limit, or completeness claims. □

Theorem 765.260 (MetaCIC candidate-mediated SN conditional budget)**.** *Any accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer may pass candidate-mediated strong-normalization evidence to a confluence-facing or decidability-facing budget only when the candidate row K, strong-normalization row S, normal-form endpoint N, handoff row U, unfilled discharge socket D, componentwise transport H, displayed replay C, provenance P, local name L, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

are all displayed. The conclusion remains conditional on the socket row D and exports no closed subject-reduction theorem, no global Church–Rosser theorem, no beta-conversion decision procedure, no quotient term equality, no selected limit, and no $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First use the obligation surface of Definition 765.257. It exposes the strong-normalization row, normal-form endpoint, handoff row, discharge socket, transport, replay, provenance, local name, and the four L10 faces before any downstream budget can be read.

Next apply the schedule exhaustion theorem, Definition 765.258. It adds the candidate row K and shows that every candidate-mediated route must display the same finite schedule before touching confluence-facing or decidability-facing data.

Finally apply the discharge theorem, Definition 765.259. It keeps the socket unfilled while the route is replayed, so the consumer receives only a conditional budget and none of the excluded closed MetaCIC, quotient, selected-limit, or Real-completeness claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNConditionalBudget`

Theorem 765.261 (MetaCIC four-face bounded checker gate)**.** *Any bounded checker consumer that uses candidate-mediated strong-normalization evidence from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must display the candidate row K, strong-normalization row S, normal endpoint row N, handoff row U, open discharge socket D, componentwise transport H, displayed replay C, provenance P, local naming L, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before it reads the checker budget. The gate exports no closed subject-reduction theorem, global confluence theorem, beta-conversion decision procedure, quotient term equality, selected limit, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFourFaceBoundedCheckerGate`

Proof. Begin with Definition 765.257. It places the strong-normalization row, normal endpoint, handoff row, discharge socket, transport, replay, provenance, local name, and four L10 faces before any confluence-facing or decidability-facing budget.

Apply Definition 765.266. The candidate row K, the S, N endpoint row, the open socket D, and the L10 face row remain separate premises rather than a single closed normalization theorem.

Use Definition 765.260. The checker budget may be read only after the displayed schedule is present, and the conclusion remains conditional on the socket row.

Finally apply Definition 765.298. The bounded-normalization candidate ledger supplies checker provenance for the same candidate-mediated route, while the accepted packet still has no coordinate for subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or Real completeness. □

Theorem 765.262 (MetaCIC bounded checker frontier separation)**.** *Any bounded checker read that consumes candidate-mediated strong-normalization evidence from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must keep the candidate row K, strong-normalization endpoint rows S, N, open discharge socket D, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as separate displayed premises before the checker budget is read. The checker may pass those premises through the displayed handoff, transport, replay, provenance, and local naming rows, but it may not collapse them into subject reduction, global confluence, beta-conversion decidability, quotient equality, selected limit, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 765.257. It displays the SN endpoint, discharge socket, handoff, transport, replay, provenance, local name, and four L10 faces before any checker-facing budget can be consumed.

Apply Definition 765.258. The candidate row K is part of the finite schedule, so it must remain visible before the bounded checker reads its budget.

Use Definition 765.260 and Definition 765.261. These keep the route conditional on D and require the four L10 faces to be displayed as the checker boundary.

Finally apply Definition 765.266. The candidate row, endpoint row, socket row, and L10 face row remain separate premises, so none can be repacked as closed subject reduction, global confluence, beta-conversion decidability, quotient equality, a selected limit, or Real completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedCheckerFrontierSeparation`

Theorem 765.263 (MetaCIC bounded checker budget factorization)**.** *Any bounded checker budget read that consumes candidate-mediated strong-normalization evidence from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet factors through four separate displayed premise rows:*

1. *the candidate row K;*
2. *the strong-normalization endpoint row S, N;*
3. *the open discharge socket row D;*
4. *the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Only after these rows are displayed may the checker read the handoff, transport, replay, provenance, local naming, and bounded budget rows. The factorization exports no closed subject-reduction theorem, global confluence theorem, beta-conversion decision procedure, quotient term equality, selected limit, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with Definition 765.261. It requires the candidate row, SN endpoint rows, open socket, and four L10 faces to be displayed before the bounded checker reads its budget.

Next apply Definition 765.260. The checker-facing budget remains conditional on the socket row D, so the budget cannot be read as a closed normalization theorem.

Use Definition 765.266. It separates K, the endpoint row S, N, the socket row D, and the L10 face row from the handoff and structural rows.

Finally read the frontier separation theorem, Definition 765.262. The same separation persists at the bounded-checker frontier, so the checker receives a factored finite budget rather than subject reduction, global confluence, beta-decision, quotient equality, selected-limit, host-equality, or Real-completeness data. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedCheckerBudgetFactorization`

Theorem 765.264 (MetaCIC bounded checker replay separation)**.** *Every replay of a bounded-checker budget read from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must preserve four separate premise rows throughout the replay:*

1. *the candidate row K;*
2. *the strong-normalization endpoint row S, N;*
3. *the open discharge socket row D;*
4. *the L10 face row over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

The replay may then read the handoff, transport, displayed continuation, provenance, local naming, and bounded budget rows, but it cannot compress the four premise rows into closed subject reduction, global confluence, beta-conversion decidability, quotient equality, selected limits, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedCheckerReplaySeparation`

Proof. Start with the candidate-mediated obligation surface of Definition 765.257. It exposes the endpoint, socket, handoff, transport, replay, provenance, local name, and four L10 faces before any checker-facing budget is read.

Use schedule exhaustion from Definition 765.258. The candidate row remains attached to the displayed finite route, so replay cannot detach K from the same accepted packet.

Apply the conditional budget theorem, Definition 765.260, together with the four-face bounded checker gate, Definition 765.261. These rows keep the socket open and require the dyadic, stream, regular-rational, and real faces to appear as checker premises.

Finally apply Definition 765.263. Its factorization is stable under replay because the replay rows are read after the four premise rows, so the bounded checker receives only the displayed conditional budget and none of the excluded closed MetaCIC, quotient, selected-limit, host-equality, or completeness claims. □

Theorem 765.265 (MetaCIC candidate-mediated SN frontier coverage)**.** *Any accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer crossing from candidate-mediated strong-normalization evidence into confluence-facing or decidability-facing use must display the whole frontier before the handoff is read:*

1. *the candidate row K;*
2. *the strong-normalization endpoint and normal-form endpoint rows S, N;*
3. *the open discharge socket row D;*
4. *the handoff row U;*
5. *the bounded checker budget row;*
6. *the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The frontier may then pass through componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. It does not export subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the obligation surface of Definition 765.257. It exposes the endpoint rows, open socket, handoff, transport, replay, provenance, local name, and four L10 faces before any confluence-facing or decidability-facing consumer reads the route.

Next apply Definition 765.260. The bounded checker budget remains conditional on the displayed candidate-mediated surface, so it cannot be detached from K, S, N, D, or U.

Use Definition 765.261. The dyadic, stream, regular-rational, and real faces must be displayed as four separate L10 premises, not compressed into a single ambient Real-completeness assumption.

Finally combine Definition 765.263 with the frontier separation theorem. The factorization keeps the bounded budget behind the same frontier rows, while separation prevents the consumer from repacking the finite route as subject reduction, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNFrontierCoverage`

Theorem 765.266 (MetaCIC candidate-mediated SN premise split)**.** *Any consumer passing candidate-mediated strong-normalization evidence from an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet into a confluence-facing or decidability-facing budget must display four separate premise rows before the budget is read:*

1. *the candidate row K;*
2. *the strong-normalization and normal-form endpoint row consisting of S and N;*

3. the unfilled discharge socket row D;

4. the L10 face row over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

The four rows remain separate from the handoff row U, componentwise transport H, displayed replay C, provenance P, and local name L. The consumer receives no closed subject-reduction theorem, arbitrary Church–Rosser theorem, global beta-conversion decision, quotient term equality, selected limit, host real equality, or Real completeness.

Proof. Start with Definition 765.257. It places the strong-normalization row, normal-form endpoint, handoff row, discharge socket, transport, replay, provenance, local name, and L10 faces before any confluence-facing or decidability-facing budget.

Next apply Definition 765.258. This adds the candidate row K as a displayed schedule premise and keeps it attached to the same accepted packet rather than to a detached normalization theorem.

Use Definition 765.259. The discharge socket D remains visible and unfilled while the candidate-mediated evidence is routed, so the socket is not absorbed into the strong-normalization endpoint.

Finally apply Definition 765.260. The budget is conditional on all displayed rows, and the L10 face row remains the finite dependency surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since the accepted route has no closed subject-reduction coordinate, arbitrary Church–Rosser theorem, global beta-conversion decision, quotient equality, selected limit, host equality, or Real-completeness row, none of those principles is exported. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNObligationSurface`
Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedSNPremiseSplit`

Theorem 765.267 (MetaCIC critical-path bounded conversion discharge)**.** *Every bounded-conversion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must consume the displayed* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *row, the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *boundary row, the* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *pair ledger, and the L10 four-object exit certificate rows over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before reading any conversion budget. The discharge route also carries the endpoint row S, N, the open socket row D, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

The discharge is finite and bounded by the displayed packet. It does not assert global strong normalization, arbitrary Church–Rosser, unrestricted beta-conversion decision, quotient term equality, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 765.266. It requires the candidate row, endpoint row, socket row, and L10 face row to be displayed separately before the budget is read.

Apply Definition 765.262. The normalization frontier remains a boundary row rather than a closed normalization theorem, so the bounded checker receives only the exposed finite interface.

Use Definition 765.258 together with the critical-pair ledger row. The candidate-mediated schedule and pair ledger are consumed as finite discharge premises, while H, C, P, L transport, replay, source, and name the same route.

Since the socket remains visible and the L10 faces remain explicit, the route cannot be repacked as global strong normalization, unrestricted conversion decidability, quotient equality, selected-limit data, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedConversionDischarge`

Theorem 765.268 (MetaCIC bounded conversion socket budget)**.** *Every bounded-conversion consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must display the candidate row, the strong-normalization endpoint rows* S, N*, the open discharge socket* D*, the bounded checker budget row, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before it reads any conversion data. The socket remains open throughout the read, and the packet exports no subject reduction, global Church–Rosser theorem, beta-decision procedure, quotient equality, selected limit, host equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedConversionSocketBudget`

Proof. Start with the candidate-mediated obligation surface, Definition 765.257. It places the endpoint rows, open socket, structural rows, and the four L10 faces before any conversion-facing budget can be inspected.

Apply the candidate-discharge schedule exhaustion, Definition 765.258. The candidate row must remain attached to the same accepted packet, so the bounded-conversion consumer cannot read it as a detached normalization theorem.

Use the bounded checker budget factorization, Definition 765.263. It separates the candidate row, endpoint rows, socket row, and L10 face row before the checker budget is read.

Finally apply the bounded conversion discharge theorem, Definition 765.267. The conversion read is finite and conditional on the displayed socket, so none of the excluded subject-reduction, Church–Rosser, beta-decision, quotient-equality, selected-limit, host-equality, or Real-completeness data is exported. □

Theorem 765.269 (MetaCIC critical-path candidate-SN readiness)**.** *An accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet is ready for candidate-mediated strong-normalization handoff precisely at the finite surface consisting of the* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *row, the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *frontier row, the* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *ledger row, the endpoint row* S, N*, the open discharge socket* D*, the L10 four-object exit certificate rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Readiness means only that those rows are all present and factor the bounded-conversion consumer route. It does not close subject reduction, confluence, global strong normalization, beta-conversion decidability, quotient term equality, selected limits, host equality, or Real completeness.

Proof. Read the candidate-mediated obligation surface and the premise split. Together they display the candidate row, endpoint row, socket row, handoff row, structural rows, and L10 faces before any confluence-facing or decidability-facing budget is consumed.

The normalization frontier and critical-pair ledger are then read as finite boundary rows. They record where bounded conversion may discharge local obligations, not a global theorem that all conversions terminate or join.

Finally apply Definition 765.267. The same displayed rows are exactly the finite discharge interface for bounded conversion, so candidate-SN readiness is a row-presence and factorization claim only. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNReadiness`

Theorem 765.270 (MetaCIC residual candidate-SN handoff)**.** *An accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet may pass candidate-mediated strong-normalization evidence to residual-substitution or confluence consumers only after it displays the* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *row, the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$

row, the MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *row, the* MetaCICCriticalPairLedger$^{\uparrow}$ *row, the endpoint rows* S, N*, the open discharge socket* D*, the four L10 faces* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*, and the structural transport, replay, provenance, and local-name rows.*

The handoff is a finite consumer boundary. It permits residual-substitution and confluence consumers to read the candidate-mediated evidence only through the displayed socket and bounded-conversion discharge rows, and it does not export subject reduction, unrestricted confluence, global strong normalization, beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualCandidateSNHandoff`

Proof. Start with the candidate-mediated obligation surface. It places the candidate row, endpoint rows, open socket, structural rows, and the four L10 faces before any normalization or conversion consumer can inspect the evidence.

Next apply the candidate-SN readiness theorem. Readiness supplies the finite row-presence and factorization condition, so the evidence remains attached to the accepted MetaCICCriticalPath$^{\uparrow}$ packet rather than becoming a detached strong-normalization theorem.

Use the premise split and bounded-conversion discharge rows. The premise split separates the candidate row from residual-substitution, frontier, pair-ledger, endpoint, socket, and L10 obligations, while the bounded-conversion discharge keeps the consumer route conditional on those displayed rows.

Finally repack through the structural coordinates. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local naming row preserve the same packet boundary; hence residual-substitution and confluence consumers receive only the displayed candidate-mediated handoff surface. □

Theorem 765.271 (MetaCIC critical-path bounded conversion consumer route)**.** *Any confluence-facing or decidability-facing consumer that uses the accepted candidate-mediated strong-normalization packet of Definition 765.269 must first read the bounded normalization candidates, the candidate discharge schedules, the finite checker budgets, and the local* NameCert$_{\text{MetaCICCriticalPath}^{\uparrow}}$ *rows. Only after these rows are displayed may the consumer export a bounded conversion claim through the* MetaCICNormalizationFrontier$^{\uparrow}$ *boundary, the* MetaCICCriticalPairLedger$^{\uparrow}$ *pair ledger, the endpoint rows* S, N*, the open discharge socket* D*, and the L10 four-object exit certificate over* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

The route is a consumer discipline, not a global conversion theorem. It does not assert unrestricted confluence, global strong normalization, unbounded beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the candidate-mediated SN readiness surface. It exposes the candidate row, normalization frontier, critical-pair ledger, endpoint rows, open socket, L10 exit certificate, and the structural H, C, P, L rows as the finite handoff boundary.

Apply the candidate discharge schedule exhaustion. The consumer must read the bounded candidates together with the displayed discharge schedule before any conversion-facing export is formed.

Use the conditional budget and bounded checker gates. These gates keep the checker budget finite and tied to the same frontier and pair-ledger rows, so confluence-facing and decidability-facing consumers receive a bounded conversion surface rather than an unrestricted theorem.

Finally repack the local NameCert rows. The hsame, Cont, Pkg, and naming coordinates preserve the accepted packet, while the socket and L10 rows prevent the route from becoming

global strong normalization, unrestricted confluence, unbounded conversion decidability, quotient equality, selected limits, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedConversionConsumerRoute`

Theorem 765.272 (MetaCIC critical-path L10 object-status readback)**.** *Any candidate-mediated strong-normalization discharge used as a roadmap-critical* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet may cite L10 object readiness only through the displayed status rows for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealCompletionExactBoundaryUp}$. *The readback factors through the four-object exit certificate, the Real L10 objectwise audit pullback, and the exact-boundary packet rows; it must not turn those rows into unconditional* $\mathsf{Real}^{\uparrow}$ *completeness, quotient real equality, selected host limits, closed subject reduction, host normalization, global confluence, or beta-conversion decidability.*

Proof. Start with the candidate-mediated SN discharge theorem, Definition 765.259. It keeps the discharge socket D open and requires the candidate row, endpoint rows, handoff row, structural rows, and four L10 faces before any confluence-facing or decidability-facing budget is read.

Next read the L10 object status route. The four-object exit certificate of Definition 765.128 displays the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, while Definition 765.132 and Definition 38.471 force those faces to contribute their own local closurestatus, formalstatus, leantarget, and bridgestatus rows.

Finally attach the completion boundary only as a finite exact-boundary read. The $\mathsf{RealCompletionExactBoundary}$ packet supplies the displayed exact-boundary rows already governed by its local NameCert surface, so the SN-facing obligation may cite that boundary as status readback but cannot export Real completeness, host normalization, global confluence, quotient equality, selected limits, or a filled discharge socket. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ObjectStatusReadback`

Theorem 765.273 (MetaCIC L10 normalization discharge budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every candidate-mediated strong-normalization, confluence, or decidability admission that is routed toward a discharge socket must first display the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

together with the packet rows $S, N, O, U, D, H, C, P, L$. *The discharge socket may then consume only this finite source budget. It receives no closed subject-reduction row, arbitrary strong-normalization theorem, global confluence theorem, beta-conversion decision procedure, quotient term equality, selected limit, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10NormalizationDischargeBudget`

Proof. Begin with the candidate-mediated SN frontier of Definition 765.198. It already places a candidate-mediated read on S, N, U, D, H, C, P, L and requires the four L10 faces before any confluence-facing or decidability-facing handoff consumes U.

Next use the candidate-normalization budget theorem, Definition 765.227. It identifies S, N, U, H, C, P, L plus $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the finite normalization surface available to such consumers.

Finally apply discharge-socket induction and socket exhaustion, Definition 765.226 and Definition 765.230. These keep O and D visible unless a separate closed subject-reduction row is displayed, so the discharge socket consumes only the named L10 budget and none of the refused global claims. □

Theorem 765.274 (MetaCIC RegSeqRat candidate-normalization route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-normalization route that exposes a regular-rational readback to a confluence-facing or decidability-facing consumer must pass through* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the candidate-normalization handoff* U *is consumed. The route is therefore a displayed L10 readback route, not a detached normalization oracle.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRegSeqRatCandidateNormalizationRoute`

Proof. Apply Definition 765.200. It requires every candidate-mediated read of U to expose the four L10 faces before the handoff is passed to a confluence-facing or decidability-facing consumer.

Refine that read with Definition 765.273. The budget keeps the regular-rational component downstream of the displayed finite windows and dyadic tolerance data, because the accepted discharge surface is the ordered L10 source package rather than an independent oracle.

The consumer-boundary theorem Definition 765.224 then prevents the handoff from being read as a global confluence theorem or beta-conversion decision procedure. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ row remains part of the visible L10 route. □

Theorem 765.275 (MetaCIC normalization-frontier handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated normalization handoff consumed by a confluence-facing or bounded-decidability consumer must expose* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before* U *is read, and it must keep the residual-budget and discharge-socket rows* O, D *visible. The handoff therefore reaches the consumer only as a displayed L10 frontier, not as full subject reduction, global Church–Rosser, beta-conversion decidability, quotient term equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationFrontierHandoff`

Proof. Start with Definition 765.273. It already requires the four L10 source faces and the packet rows before any discharge socket consumes a candidate-mediated normalization budget.

Refine the regular-rational part by Definition 765.274. The route places $\mathsf{StreamName}^{\uparrow}$ finite windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data before the $\mathsf{RegSeqRat}^{\uparrow}$ readback, so U cannot be consumed as a detached normalization oracle.

Next use the candidate-mediated strong-normalization frontier and the confluence-decidability consumer boundary. These keep O and D visible as residual-budget and discharge-socket rows whenever a confluence-facing or bounded-decidability consumer reads the handoff.

The refused endpoints are exactly the coordinates absent from the displayed frontier. Since the route exposes only the four L10 faces together with $S, N, O, U, D, H, C, P, L$, it cannot supply full subject reduction, global Church–Rosser, beta-conversion decidability, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 765.276 (MetaCIC Real seal no infinite-reduction handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. When a candidate-mediated normalization route reaches the* $\mathsf{Real}^{\uparrow}$ *seal face, the seal records only that the finite L10 readback route has been displayed. It does not certify absence of infinite reduction by itself, fill the discharge socket* D*, erase the obstruction row* O*, or supply closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealSealNoInfiniteReductionHandoff`

Proof. Start from Definition 765.273. It places the $\mathsf{Real}^{\uparrow}$ face after $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ inside the finite source budget consumed by a discharge socket.

Use Definition 765.230. Even after the candidate-normalization budget and four L10 faces have been consumed, the subject-reduction socket remains unfilled unless a separate closed subject-reduction row is displayed.

Finally apply Definition 765.199. It keeps O and D visible as rows distinct from the handoff and from the four L10 faces, so the $\mathsf{Real}^{\uparrow}$ seal cannot be reinterpreted as an infinite-reduction exclusion theorem or as any of the refused global metatheoretic claims. □

Theorem 765.277 (MetaCIC Real-completion SN budget separation)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization consumer that uses the* $\mathsf{Real}^{\uparrow}$ *completion-facing L10 budget must display* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, the obstruction row* O*, and the discharge socket* D *before any strong-normalization, confluence, or decidability claim is consumed. The displayed budget does not provide closed subject reduction, arbitrary strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionSNBudget`

Proof. First apply the L10 normalization discharge budget, Definition 765.273. It forces every candidate-mediated consumer routed toward a discharge socket to expose the four L10 faces and the accepted packet rows before the socket can consume the budget.

Next compose the RegSeqRat candidate-normalization route, Definition 765.274. This places the regular-rational readback after finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, so the consumer cannot read $\mathsf{RegSeqRat}^{\uparrow}$ as a detached normalization oracle.

Finally use the Real seal handoff theorem, Definition 765.276. It keeps O and D visible at the completion-facing edge and refuses every global metatheoretic claim listed in the statement. Thus the Real-completion budget is a displayed L10 candidate surface, not a discharge of strong normalization, confluence, decidability, quotient equality, selected limits, or Real completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionSNBudgetSeparation`

Theorem 765.278 (MetaCIC critical-path candidate-discharge frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *root-unblock packet. Candidate-mediated strong-normalization rows, confluence rows, and bounded-decidability rows may be consumed by the root-unblock route only while the subject-reduction obstruction row O and the discharge socket D remain visible through the same packet. These candidate rows do not fill D unless a separate subject-reduction certificate row is displayed.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeFrontier`

Proof. Begin with the consistency handoff of Definition 765.2. It permits the positive read from the strong-normalization row S to the normal-form row N only through the displayed route, transport, provenance, and local naming rows.

Next keep the blocked side visible. The obstruction boundary Definition 765.3, the NameCert obligation surface Definition 765.20, and discharge-socket nonescape Definition 765.25 keep O and D as separate rows of the same accepted packet.

Now consume the candidate evidence. The candidate-mediated SN frontier Definition 765.198, the candidate-mediated SN obligation surface Definition 765.257, and the candidate-mediated subject-reduction boundary Definition 765.231 allow strong-normalization, confluence-facing, and bounded-decidability consumers to inspect the finite candidate rows, but only with O and D still exposed.

Finally apply discharge-socket coverage, Definition 765.26. A downstream socket read remains inside $S, N, O, U, D, H, C, P, L$ unless a separate subject-reduction input is displayed. Hence candidate-mediated SN, confluence, and bounded-decidability rows sharpen the frontier but do not discharge the socket by themselves. □

This candidate-discharge frontier consumes the residual-budget and diamond-premise rows of chapter 4061. The dependency keeps those sibling budget rows separate from the subject-reduction discharge socket, so a lattice read may use them as confluence-facing route data without treating them as the missing subject-reduction certificate.

Theorem 765.279 (MetaCIC candidate frontier L10 discharge boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *root-unblock packet. Candidate-mediated strong-normalization and confluence rows may consume the displayed L10 readiness rows* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *as source evidence for the current Phase Real and Completion route. Those four rows do not fill the subject-reduction discharge socket D unless a separate subject-reduction certificate is displayed in the same packet route.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierL10DischargeBoundary`

Proof. Start with the candidate-discharge frontier, Definition 765.278. It keeps candidate-mediated strong-normalization, confluence-facing, and bounded-decidability rows downstream of the visible obstruction row O and discharge socket D.

Read the candidate-mediated SN surface next. Theorems 765.198 and 765.257 allow the candidate rows to support normalization and confluence consumers, but only as displayed rows inside the accepted packet.

Now attach the L10 four-object route. Theorem 765.128 and the source-obligation surface of Theorem 765.165 make DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ available as readiness rows for the Phase Real and Completion route.

Finally apply discharge-socket coverage, Definition 765.26. The four L10 rows supply source readiness, not a subject-reduction certificate. Therefore the socket D remains open until the separate certificate is displayed, and no Real or Completion readiness row can be read as a hidden discharge. □

Theorem 765.280 (MetaCIC bounded-conversion candidate boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Bounded conversion rows, confluence rows, and strong-normalization candidate rows may be read by the packet only while the subject-reduction obstruction row O and the discharge socket D remain displayed through the same route. None of those candidate rows closes subject reduction unless an explicit subject-reduction certificate row is present in the packet.*

Proof. Start with the candidate-discharge frontier, Definition 765.278. It permits candidate-mediated strong-normalization, confluence-facing, and bounded-decidability evidence only with O and D still visible.

Next apply the L10 discharge boundary, Definition 765.279. That theorem lets the packet consume the displayed source-readiness rows without treating any Real or Completion readiness row as the missing subject-reduction certificate.

Finally use discharge-socket coverage, Definition 765.26. A bounded-conversion row can support decidability and confluence inspection, but the socket D is filled only by a displayed subject-reduction certificate row. Hence candidate evidence remains separate from subject-reduction discharge. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedConversionCandidateBoundary`

Theorem 765.281 (MetaCIC candidate-discharge frontier coverage)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. The candidate-discharge frontier covers every permitted strong-normalization, confluence, bounded-conversion, and L10 source-readiness read only through the displayed rows $S, N, O, U, D, H, C, P, L$ and the four visible faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The coverage keeps the subject-reduction obstruction row O and discharge socket D visible throughout and does not claim that subject reduction has been supplied.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPacket candidate discharge frontier coverage`

Proof. Start with Definition 765.278. It keeps candidate-mediated normalization, confluence-facing, and bounded-decidability rows downstream of the visible obstruction and socket rows.

Add the L10 boundary from Definition 765.279. That result permits the four source-readiness faces to support the Phase Real and Completion route while refusing to read them as a subject-reduction certificate.

Finally apply Definition 765.280 and Definition 765.26. Together they force bounded conversion and every downstream socket read to preserve O and D unless a separate subject-reduction row is displayed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateFrontierDischargeCoverage`

Theorem 765.282 (MetaCIC Real root budget lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Any Real-facing or Completion-facing root budget consumed by the candidate frontier is locked to the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readiness rows, carried through the same finite* BHist, hsame, Cont, Pkg, *and* NameCert *route as* K. *The lock supplies L10 source budget only; it does not close subject reduction, quotient stream equality, selected host limits, host real equality, trichotomy, open-cover compactness, or completed-real closure.*

Proof. Read the L10 source side through Definition 765.128. It displays exactly the dyadic budget, StreamName schedule, RegSeqRat readback, and Real seal rows before the Phase Real and Completion exit can be consumed.

Use Definition 765.165. The four faces are carried as finite source obligations through the same packet route, so the Real-facing budget cannot be replaced by an aggregate host object.

Now apply Definition 765.281. The candidate frontier may consume this source budget only while the obstruction row and discharge socket remain visible. Therefore the root budget is locked as readiness evidence and cannot discharge subject reduction or add any excluded completion principle. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealRootBudgetLock`

Theorem 765.283 (MetaCIC substitution boundary routing)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Substitution-facing reads that pass through the bounded-conversion or confluence side of the candidate frontier must route through the displayed handoff row* U, *the obstruction row* O, *the discharge socket* D, *and the structural rows* H, C, P, L *before they may consume the L10 readiness rows. Such reads may support bounded conversion and candidate checking, but they do not supply the missing subject-reduction certificate.*

Proof. Begin with Definition 765.60. It keeps substitution-facing reads visible through the handoff route and prevents the handoff row from becoming a detached theorem export.

Next use Definition 765.280. Bounded conversion, confluence, and candidate rows remain attached to the same packet and keep the obstruction and discharge rows visible.

Finally apply Definition 765.282. The L10 faces may be consumed only as source-readiness rows behind the visible socket route. Hence substitution-boundary routing can serve bounded conversion and candidate checking, while subject reduction remains open until a separate certificate row is displayed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCarrier substitution boundary routing`

Theorem 765.284 (MetaCIC candidate discharge budget route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Bounded conversion, candidate-mediated strong normalization, confluence, and the L10 readiness faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *form a finite budget route inside* K*. The subject-reduction discharge socket* D *remains open until a separate subject-reduction certificate row is displayed, so bounded-conversion evidence and L10 readiness cannot be read as a hidden subject-reduction discharge.*

Proof. Start with Definition 765.278. It displays the candidate-mediated frontier and keeps the obstruction row and discharge socket visible inside the same packet.

Use Definition 765.280. Bounded conversion, confluence, and candidate rows are budget evidence attached to K, not a detached proof of the missing discharge.

Now apply Definition 765.282 and Definition 765.283. The L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are consumed only as readiness rows behind the visible handoff and socket route.

Finally invoke Definition 765.26. The socket D is still the named subject-reduction boundary. Since no certificate row for subject reduction is displayed, the finite budget route supports bounded conversion and candidate checking while leaving the subject-reduction socket open. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeBudgetRoute`

Theorem 765.285 (MetaCIC discharge socket case split)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet whose candidate-discharge budget route is displayed. Then every candidate-discharge read of* K *falls into exactly one of three finite cases: a filled discharge case, where a displayed subject-reduction certificate row fills* D*; a visible obstruction case, where* O *remains the exposed subject-reduction obstruction; or a residual socket case, where candidate evidence is present but* D *is still an unfilled discharge socket. None of the three cases exports closed subject reduction, full closed consistency, quotient proof equality, or a detached normalization theorem.*

Proof. Use the candidate frontier coverage to restrict the read to the rows $S, N, O, U, D, H, C, P, L$ and the visible L10 readiness faces. No case may inspect a row outside that finite packet.

Next apply the candidate-discharge budget route. It separates bounded conversion, candidate-mediated strong normalization, confluence-facing data, and Real-readiness budget from the subject-reduction discharge socket.

The displayed socket row D is therefore either filled by an explicit subject-reduction certificate row, blocked by the still-visible obstruction row O, or carried forward as the residual socket. These alternatives exhaust the finite socket coordinates, and the nonescape theorems prevent any branch from exporting closed subject reduction or full closed consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeSocketCaseSplit`

Theorem 765.286 (MetaCIC subject-reduction socket case exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet whose candidate-discharge budget route is displayed. Every candidate-discharge read of* K *falls into a filled-discharge case, a visible-obstruction case, or a residual-socket case. These cases exhaust the finite packet surface while preserving the refusal to claim closed subject reduction unless an explicit subject-reduction certificate row is displayed.*

Proof. First restrict the read to the displayed packet. The candidate-discharge frontier coverage of Definition 765.281 allows only the rows $S, N, O, U, D, H, C, P, L$ and the visible L10 readiness faces.

Next separate the candidate boundary. The bounded-conversion candidate boundary of Definition 765.280 keeps bounded conversion, confluence, and candidate-mediated strong-normalization rows attached to the same visible obstruction and discharge socket.

Now read the L10 source side through Definition 765.128 and Definition 765.165. Those rows provide dyadic, stream, regular-rational, and Real readiness evidence for the Phase Real and Completion route, not a hidden subject-reduction certificate.

Finally apply the case split of Definition 765.285. The socket is either filled by an explicit certificate row, blocked by the visible obstruction row, or carried as a residual socket. Since those are the only finite socket coordinates exposed by the packet, the cases exhaust the read while closed subject reduction remains unavailable without the displayed certificate. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathSubjectReductionSocketCaseExhaustion`

Theorem 765.287 (MetaCIC candidate-discharge frontier totality)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet whose candidate-discharge budget route is displayed. Every candidate-discharge read of K is exhausted by bounded conversion, candidate-mediated strong normalization, confluence-facing rows, L10 readiness rows, and exactly one visible socket case. None of these rows closes subject reduction unless an explicit subject-reduction certificate row is displayed in the same packet route.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeFrontierTotality`

Proof. Begin with the candidate-discharge budget route, Definition 765.284. It places bounded conversion, candidate-mediated strong normalization, confluence data, and the L10 readiness faces inside the same finite packet while keeping the discharge socket D visible.

Next apply the subject-reduction socket case exhaustion theorem, Definition 765.286. The read has exactly one visible socket status: filled by an explicit certificate row, blocked by the visible obstruction row O, or carried as the residual socket.

Read the L10 source obligations through Definition 765.165. Those rows supply dyadic, stream, regular-rational, and Real readiness for the Phase Real and Completion route, not a subject-reduction certificate.

Finally use discharge-socket coverage, Definition 765.26. Since coverage preserves the socket until the explicit subject-reduction row is displayed, the frontier is total for candidate-mediated reads without turning bounded conversion, confluence, or readiness evidence into closed subject reduction. □

Theorem 765.288 (MetaCIC residual-budget discharge frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Any candidate-mediated strong-normalization or confluence consumer that attempts to read the discharge socket D must first display residual-budget rows, substitution-boundary rows, closed-premise ledger rows, and local-diamond rows, all attached to the same packet rows $S, N, O, U, D, H, C, P, L$. Without those four frontier families, the read may inspect the candidate-mediated budget and confluence-decidability handoff, but it cannot treat D as discharged.*

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualBudgetDischargeFrontier

Proof. Begin with the candidate-mediated SN frontier, Definition 765.198. It forces a strong-normalization consumer to display the packet rows and the L10 source faces before the handoff row U is consumed.

Next add the SN confluence-decidability budget and discharge-socket readiness, Definition 765.193 and Definition 765.194. These theorems require the confluence-audit dependency exposure row and the closed-substitution boundary exposure row before a socket-facing consumer can reach D.

Now use the candidate-discharge frontier, Definition 765.278. It keeps the obstruction row O and discharge socket D visible while candidate-mediated strong-normalization, confluence, and bounded-decidability rows are consumed.

Finally attach the residual and diamond budget side through Definition 4061.6, Definition 4061.7, Definition 2325.3, and Definition 3595.3. The resulting frontier requires residual-budget rows, substitution-boundary rows, closed-premise ledger rows, and local-diamond rows before a discharge-socket read is available, and none of those rows supplies closed subject reduction by itself. □

Theorem 765.289 (MetaCIC residual-budget socket route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. Any candidate-mediated strong-normalization discharge that crosses the residual frontier must expose the candidate row, residual budget row, socket and normalization route, and finite audit ledger before any downstream confluence or decidability consumer may read the discharge. In particular, the read factors through* $S, N, O, U, D, H, C, P, L$ *together with the residual-budget, substitution-boundary, closed-premise, and local-diamond rows required by Definition 765.288.*

Proof. Start from the residual-budget discharge frontier, Definition 765.288. It already requires the residual-budget, substitution-boundary, closed-premise ledger, and local-diamond families to be attached to the same packet rows $S, N, O, U, D, H, C, P, L$ before D is treated as discharged.

Now isolate the socket read. The candidate-mediated frontier Definition 765.198 exposes the candidate row and L10 source faces before strong-normalization handoff, while Definition 765.278 keeps the obstruction and discharge socket visible during the candidate-mediated handoff.

Finally combine the normalization and audit side. The SN confluence-decidability budget and socket-readiness theorems force the confluence-audit and closed-substitution rows to remain visible before any confluence or decidability consumer reads the discharge. Thus crossing the residual frontier supplies a socket route only after the candidate row, residual budget row, normalization route, and finite audit ledger have all been displayed. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualBudgetSocketRoute

Theorem 765.290 (MetaCIC critical-path candidate schedule discharge)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *root-unblock packet. A candidate-mediated schedule for strong normalization and confluence can discharge the open route only after it exposes candidate-set normalization, a critical-pair ledger, a residual-substitution boundary, finite observation confluence, and subject-reduction socket rows through the same packet. The schedule cannot cite unbounded confluence, global strong normalization, hidden substitution principles, quotient term equality, or ambient metatheoretic consistency as discharge data.*

Proof. Begin with the candidate-mediated SN obligation surface, Definition 765.257. It forces candidate-mediated normalization reads to display the strong-normalization row, normal-form endpoint, handoff row, discharge socket, transport, replay, provenance, local name, and the four L10 faces before confluence or decidability budgets are consumed.

Next expose the finite schedule rows. The candidate-discharge frontier of Definition 765.278 keeps the subject-reduction obstruction row O and discharge socket D visible, while the dependency-exposure rows of Definition 765.291 and Definition 765.292 keep candidate-set normalization, critical-pair ledgers, residual-substitution boundaries, finite observation confluence, and subject-reduction sockets as named dependencies rather than hidden assumptions.

Finally apply the confluence-decidability handoff exhaustion of Definition 765.171. It permits bounded consumers to read the candidate schedule only through the displayed packet and refuses full confluence, global strong normalization, hidden substitution, quotient equality, and consistency exports.

Thus a candidate-mediated schedule can discharge only the rows it displays. Without the five named schedule rows, the open socket remains a boundary rather than a completed metatheorem. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateScheduleDischarge`

Theorem 765.291 (MetaCIC critical-path normalization-frontier dependency exposure)**.** *Every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet exposes the normalization-frontier dependency as the finite row route from the candidate-mediated strong-normalization surface of* MetaCICNormalizationFrontier$^{\uparrow}$ *and* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *into the normal-form endpoint row N of Definition 765.1. The exposure also keeps the source-window siblings* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *visible through the open-phase ledger before any normal-form read is consumed.*

This row is an obligation surface for the MetaCIC critical path. It is not a subject-reduction proof, not a closed-consistency theorem, and not a verification-axis upgrade.

Proof. Project the accepted carrier $K = (S, N, O, U, D, H, C, P, L)$. The row S is the only normalization-frontier entrance, and the row N is the only normal-form endpoint available to a consistency-facing consumer.

Read the cited sibling packets through the same route. MetaCICNormalizationFrontier$^{\uparrow}$ and MetaCICNormalizationCertificateRouter$^{\uparrow}$ supply the candidate-mediated normalization evidence, while DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ remain the finite source ledger rows for the open phase.

Since O and D remain visible beside S and N, the dependency exposure stops at the displayed obligation row. No carrier coordinate fills subject reduction or raises the verification axis. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalizationFrontierDependencyExposure`

Theorem 765.292 (MetaCIC critical-path confluence-audit dependency exposure)**.** *Every accepted* MetaCICCriticalPath$^{\uparrow}$ *packet exposes the confluence-audit dependency through the handoff row U, with the confluence-facing siblings* CandidateSNConfluenceBoundary$^{\uparrow}$*,* FiniteObservationConfluence$^{\uparrow}$*,* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$*, and* MetaCICCriticalPairLedger$^{\uparrow}$ *read only as bounded audit rows. The packet may use these rows to keep the route from S to N coherent with the displayed obstruction O and discharge socket D.*

This row is an obligation surface for confluence auditing on the MetaCIC path. It is not a Church–Rosser theorem, not a subject-reduction proof, and not a verification-axis upgrade.

Proof. Begin with the handoff row U in Definition 765.1. It is the only row that names substitution, confluence, and decidability support before the normal-form consistency read is attempted.

The confluence-facing siblings are therefore consumed as finite audit rows. $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ and $\mathsf{FiniteObservationConfluence}^{\uparrow}$ keep the SN-window and finite-observation sides bounded, while $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ and $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ keep critical-pair data visible as ledger entries.

The obstruction row O and socket D remain part of the same carrier. Thus the confluence dependency is exposed as a bounded obligation surface, not as a proof of global confluence or as a change in verification status. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathConfluenceAuditDependencyExposure`

Theorem 765.293 (MetaCIC critical-path closed-substitution boundary exposure)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet exposes closed-substitution support through the same handoff row U and discharge socket D, reading* $\mathsf{MetaCICClosureTrace}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, *and* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *only as boundary packets. The rows may explain which closed-substitution data must be present before a later subject-reduction consumer can act, but they do not fill that consumer.*

This row is an obligation surface for the closed-substitution boundary. It is not a subject-reduction proof, not a closed MetaCIC consistency theorem, and not a verification-axis upgrade.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathClosedSubstitutionBoundaryExposure`

Proof. Inspect the critical-path carrier. The row U names the substitution handoff, while D names the separate discharge socket that must be filled before any subject-reduction consumer can be treated as supplied.

The closed-substitution siblings are consumed only as boundary data. $\mathsf{MetaCICClosureTrace}^{\uparrow}$ records closedness-preservation routes, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ records the closed-term projection boundary, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ records the four-row discharge socket, and $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ records the residual substitution boundary.

Because D remains a socket and O remains visible, the packet cannot convert these boundary rows into subject reduction or closed consistency. The theorem records the dependency surface and no verification-axis change. □

Theorem 765.294 (MetaCIC critical-path L10 candidate-discharge order)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet reads any Real-and-Completion candidate-discharge surface through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the normal-form endpoint row N or discharge socket D of Definition 765.1 may be consumed. The row L is the open-phase source ledger for this route, while S, N, O, U, D remain MetaCIC critical-path rows. The theorem records the candidate-discharge ordering; it does not claim subject reduction, closed consistency, or completion of the Real-and-Completion phase.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10CandidateDischargeOrder`

Proof. Project the accepted carrier $K = (S, N, O, U, D, H, C, P, L)$ from Definition 765.1. The rows S, N, O, U, D name the MetaCIC normalization surface, normal-form endpoint, obstruction, handoff support, and discharge socket.

The open Real-and-Completion data are carried by L. A candidate-discharge read that mentions Real-facing source data must therefore display the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route through that ledger before it can consume either N or D.

Transport through H, replay through C, and provenance through P preserve this separation of roles. Since D remains a socket and N remains an endpoint row, the carrier supplies an ordering obligation only, not subject reduction or a closed consistency theorem. □

Theorem 765.295 (MetaCIC root-unblock admission)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *admission for candidate-mediated strong-normalization, confluence, or decidability work must first expose the four L10 source rows* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *through the open-phase ledger of Definition 765.155 before a candidate checker row is consumed. The admission may then read the candidate-mediated frontier of Definition 765.198, the Phase Real root-unblock route of Definition 765.167, and the candidate-mediated obligation surface of Definition 765.257; it exports no closed subject-reduction row, global confluence theorem, global beta-conversion decision procedure, quotient term equality, selected limit, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRootUnblockAdmission`

Proof. Start with the open-phase source ledger. Definition 765.155 displays the packet rows together with the four L10 source faces, so a MetaCIC admission cannot consume a candidate checker row before those faces are visible.

Next apply the Phase Real root-unblock admission theorem, Definition 765.167. It identifies the admitted root-unblock route with the same packet rows and the separate $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ face records.

Finally attach the candidate-mediated surface. The frontier theorem Definition 765.198 and the obligation surface theorem Definition 765.257 allow confluence-facing and decidability-facing consumers to inspect the candidate row only after the displayed source ledger is present. Since the cited surfaces contain none of the excluded metatheoretic or completion coordinates, the root-unblock admission is exactly the stated L10-ledger-first route. □

Theorem 765.296 (MetaCIC critical-path real-completion source budget)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer attempting a candidate-mediated strong-normalization, confluence, or decidability handoff must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source rows named by the current Phase Real and Completion ledger. Only after those four source rows are visible may the consumer read the MetaCIC closure trace, critical-pair ledger, normalization frontier, subject-reduction discharge socket, and confluence-decidability budget rows. The handoff exports no closed subject-reduction theorem, global confluence theorem, global beta-conversion decision procedure, quotient term equality, selected limit, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionSourceBudget`

Proof. Begin with root-unblock admission. Theorem 765.295 requires any candidate-mediated strong-normalization, confluence, or decidability admission to expose the four L10 source rows before a candidate checker row is consumed.

Next read the Phase Real source obligations. Theorem 765.165 places the dyadic finite-observation row, stream schedule, regular-rational readback, and Real seal before any discharge socket, confluence-decidability handoff, obstruction boundary, or downstream readiness boundary.

The existing source ledger then supplies the finite route. Definition 765.155 and Theorems 765.156 and 765.159 keep every current Phase Real and Completion read inside the packet rows and the four displayed source faces.

Finally attach the MetaCIC side. The confluence-decidability budget and handoff-exhaustion theorems require the closure trace, critical-pair ledger, normalization frontier, discharge socket, and budget rows to remain downstream of the source ledger. Since the route has no additional coordinate, the excluded metatheoretic and completion claims cannot be exported. □

Theorem 765.297 (MetaCIC critical-path real-completion budget triangle)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer combining candidate-mediated strong normalization, confluence-decidability, and the L10 Real and Completion handoff must read the four L10 source rows* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before reading the closure trace, critical-pair ledger, normalization frontier, subject-reduction socket, or confluence-decidability budget. The combined route exports no quotient stream equality, selected limit, host real equality, global confluence theorem, global beta-conversion decision procedure, discharged subject-reduction theorem, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealCompletionBudgetTriangle`

Proof. Start with the existing confluence-decidability source budget, Definition 765.296. It places every confluence-facing or decidability-facing read downstream of the four displayed L10 source rows and keeps the closure trace, critical-pair ledger, normalization frontier, discharge socket, and budget rows on the MetaCIC side of that source boundary.

Next attach the Phase Real dependency budget. Theorems 765.167 and 765.114 require every current Real and Completion consumer to present the dyadic budget, stream schedule, regular-rational readback, and Real seal rows as the finite source surface, with no fifth source row or hidden completion credential.

Finally read the candidate-mediated strong-normalization consumer through the same packet. Theorem 765.257 allows SN-facing data only after the packet rows and four L10 faces are displayed, and Theorem 765.192 keeps confluence and decidability reads downstream of the same open-source ledger. Thus the three sides of the route share one L10 source budget triangle, and every listed metatheoretic, quotient, choice, order, equality, and completeness claim remains outside the accepted consumer surface. □

Theorem 765.298 (MetaCIC bounded normalization candidate ledger)**.** *Any bounded normalization candidate consumed by the current Phase Real and Completion route for an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must display the finite packet rows*

$$S, N, O, U, D, H, C, P, L$$

together with the four L10 faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The ledger may be read only as candidate-mediated normalization evidence feeding confluence-facing and bounded decidability-facing consumers through the open-phase source budget; it exports no closed subject reduction, full consistency, quotient term equality, global beta-conversion decidability, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the candidate-mediated SN finite socket normal form, Definition 765.177. It identifies the consumable normalization candidate as the socket-aware rows $S, N, O, U, D, H, C, P, L$ together with separate local face records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply confluence candidate mediation, Definition 765.181. Confluence-facing reads remain mediated by the handoff row U, the obstruction and socket rows O, D, and the structural rows H, C, P, L; the candidate route does not become a closed Church–Rosser or subject-reduction theorem.

Apply the decidability fragment route, Definition 765.183. Bounded checker evidence is confined to the same packet rows and four L10 faces, so it cannot be promoted to global beta-conversion decidability or quotient term equality.

Finally use the real-completion source budget, Definition 765.296. The current Phase Real and Completion route reads DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before any candidate-mediated handoff is consumed. Since the cited routes contain no row for selected limits, full consistency, closed subject reduction, quotient equality, global beta-conversion decidability, or Real$^{\uparrow}$ completeness, the displayed rows form the whole bounded normalization candidate ledger. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBoundedNormalizationCandidateLedger`

Theorem 765.299 (MetaCIC critical-path public interface)**.** *Every public* MetaCICCriticalPath$^{\uparrow}$ *consumer route is the finite union of the L10 four-object status rows, the candidate-mediated strong-normalization frontier rows, the confluence and decidability budget rows, and the residual discharge rows already displayed by the chapter. The interface may expose* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *source status; CandidateMediatedSN and CandidateSNConfluenceBoundary readiness; confluence, finite-observation, and decidability budgets; and the residual discharge socket. It does not claim full normalization, full subject reduction, host consistency, or bridge closure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicInterface`

Proof. First project the L10 status surface. Theorems 765.128, 765.132, and 765.145 expose the four object rows and keep them as separate finite source records.

Next project the candidate-mediated frontier. Theorems 765.198, 765.206, and 765.207 present CandidateMediatedSN and its obstruction boundary as displayed BHist route data rather than as a global normalization theorem.

Then project the confluence and decidability budgets. Theorems 765.192, 765.193, and 765.171 keep finite-observation confluence, candidate SN, and decidability uses on their visible budget rows.

Finally project the residual discharge surface through Definition 765.288 and Definition 765.289. The finite union of these projections is the public interface route, and finite BHist, hsame, Cont, Pkg, and NameCert projection adds no full normalization, full subject reduction, host consistency, or bridge closure claim. □

Theorem 765.300 (MetaCIC critical-path residual public handoff)**.** *Every public* MetaCICCriticalPath$^{\uparrow}$ *consumer that reads residual-discharge data must pass the read through the residual budget socket and the candidate-mediated frontier rows of Definition 765.299 before it claims any confluence or decidability budget consumption. The handoff may expose the residual-budget, substitution-boundary, closed-premise, local-diamond, candidate-mediated strong-normalization, confluence-frontier, and decidability-budget rows as displayed public data. It does not discharge full normalization, full subject reduction, host consistency, full confluence, or beta-conversion decidability.*

Proof. Start with the public interface theorem. It already separates the residual discharge rows from the candidate-mediated frontier rows and from the confluence and decidability budget rows.

Now route the residual read through the socket. Theorems 765.288 and 765.289 require residual-budget rows, substitution-boundary rows, closed-premise ledger rows, and local-diamond rows before any discharge socket may be treated as available.

Next attach the candidate-mediated frontier. The public interface proof cites the candidate-mediated strong-normalization frontier and its handoff and obstruction rows, so a public consumer must keep those rows visible while reading the residual socket.

Finally consume the confluence and decidability budget only after the residual and candidate rows have both been displayed. The confluence-decidability budget, SN confluence budget, and handoff-exhaustion rows remain budget rows; they do not supply full normalization, full subject reduction, host consistency, full confluence, or beta-conversion decidability. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualPublicHandoff`

Theorem 765.301 (MetaCIC public interface frontier readiness)**.** *A public* MetaCICCriticalPath$^{\uparrow}$ *consumer may read CandidateMediatedSN, CandidateSNConfluenceBoundary, finite-observation confluence, decidability budgets, and residual discharge sockets only as readiness rows over the same displayed finite route of Definition 765.299. Before claiming closed subject reduction, full confluence, beta-conversion decidability, or host consistency, the consumer must cite a separate subject-reduction row and the matching discharge row for that global claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicInterfaceFrontierReadiness`

Proof. Project the public interface theorem. It already displays the L10 source rows, candidate-mediated frontier rows, confluence and decidability budgets, and residual discharge sockets as a finite union of carrier-visible routes.

Readiness is preserved because each named frontier row remains attached to the same BHist, hsame, Cont, Pkg, and NameCert coordinates. CandidateMediatedSN and CandidateSNConfluenceBoundary are therefore usable as frontier evidence, while finite-observation confluence and decidability budgets remain budget rows rather than full metatheorems.

The global claims require a further row. Since the public interface theorem explicitly refuses full normalization, full subject reduction, host consistency, and bridge closure, a consumer cannot obtain closed subject reduction, full confluence, beta-conversion decidability, or host consistency without citing the separate subject-reduction and discharge surfaces. □

Theorem 765.302 (MetaCIC critical-path bridge route)**.** *Every bridge-facing public read of* MetaCICCriticalPath$^{\uparrow}$ *factors through the L10 four-object rows for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *the CandidateMediatedSN frontier rows, the confluence and decidability budget rows, and the residual budget socket before any consumer may claim bridge readiness. The route exposes only finite* BHist, hsame, Cont, Pkg, *and* NameCert *data. It does not claim full normalization, full subject reduction, host consistency, full confluence, beta-conversion decidability, or bridge closure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathBridgeRoute`

Proof. Begin with the public interface theorem. It presents the L10 source rows, the candidate-mediated frontier, the confluence and decidability budgets, and the residual discharge rows as a single finite interface surface.

Next apply the residual public handoff. That theorem forces every residual read through the residual budget socket and keeps the candidate-mediated frontier visible before any confluence or decidability budget is consumed.

Finally apply the frontier-readiness theorem. It allows the same displayed rows to be read as readiness evidence, but it requires a separate subject-reduction row and matching discharge row for any global claim. Therefore a bridge-facing consumer can only read the finite interface route; it cannot derive full normalization, full subject reduction, host consistency, full confluence, beta-conversion decidability, or bridge closure from these public rows. □

Theorem 765.303 (MetaCIC public interface gate)**.** *A* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer passes the public interface gate only when its bridge-facing read cites the public interface theorem, the residual public handoff, and the frontier-readiness row in that order. Passing the gate records that the consumer has displayed the L10 four-object rows, CandidateMediatedSN frontier, confluence and decidability budgets, and residual budget socket; it records no full normalization theorem, no full subject-reduction theorem, no host-consistency theorem, no full-confluence theorem, no beta-conversion decidability theorem, and no bridge closure.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathPublicInterfaceGate`

Proof. The gate first checks the row inventory against Definition 765.299. Missing one of the L10 source rows, the candidate-mediated frontier rows, the confluence or decidability budget rows, or the residual discharge rows leaves the public interface unreadable.

It then checks residual order through Definition 765.300. The residual socket must be read before any confluence or decidability budget consumption is advertised, and the candidate-mediated frontier must remain attached to that read.

It finally checks readiness through Definition 765.301. Readiness is a public-row status over the displayed finite route, so the gate cannot convert it into full normalization, full subject reduction, host consistency, full confluence, beta-conversion decidability, or bridge closure. □

Theorem 765.304 (MetaCIC residual substitution frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A MetaCIC critical-path candidate may enter the normalization frontier only after the closed-substitution boundary rows, residual-budget rows, and candidate-mediated strong-normalization rows are all displayed on the same packet route* $S, N, O, U, D, H, C, P, L$*. The frontier therefore consumes the closed-substitution boundary of Definition 765.194, the residual-budget socket route of Definition 765.289, and the candidate-mediated SN frontier of Definition 765.198 before any confluence-facing or decidability-facing handoff reads* U*.*

Proof. Start with the candidate-mediated SN frontier. It requires the candidate rows S, N, U, D, H, C, P, L, keeps O, D visible, and displays the L10 source faces before the handoff row U is consumed.

Next attach the residual-budget discharge frontier. By Definition 765.288 and Definition 765.289, a socket-facing read must expose residual-budget, substitution-boundary, closed-premise, and local-diamond rows on the same packet.

Finally use the discharge-socket readiness theorem. It places the closed-substitution boundary beside the confluence-audit dependency exposure before the socket can be read, so entering the normalization frontier requires all three displayed families: closed substitution, residual budget, and candidate-mediated strong normalization. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSubstitutionFrontier`

Theorem 765.305 (MetaCIC residual-substitution diamond budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any diamond, confluence, or bounded decidability consumer exposed by* K *must read the same residual-substitution frontier, candidate-mediated strong-normalization frontier, SN confluence-decidability budget, residual-budget socket route, and L10 source rows on the packet route* $S, N, O, U, D, H, C, P, L$*. Consequently no confluence-facing consumer may detach a diamond proof from the residual budget rows, bypass the closed-substitution boundary, or read* U *through an L10 source route absent from the accepted packet.*

The L10 source faces used by the residual diamond consumer route through the Cauchy name surface of chapter 3413, the regular rational readback of chapter 301, and the Real seal already displayed on the packet. This is a finite-source dependency for residual diamond consumers, not a Real completeness assertion or a quotient equality between source names.

Proof. Start with the residual-substitution frontier of Definition 765.304. It places the closed-substitution boundary, residual-budget rows, and candidate-mediated strong-normalization rows on one route before any confluence-facing handoff may consume U.

Next apply the diamond budget of Definition 765.306. A diamond checker that feeds decidability already consumes the residual budget exposed by that frontier, so it has no independent budget source.

Now attach the residual decidability cut of Definition 765.307. It forces the bounded decidability readback to expose the residual-substitution frontier, the candidate SN confluence checker, the SN confluence-decidability budget, and the residual-budget socket route on the same packet.

Finally use the candidate-mediated SN frontier of Definition 765.198. The candidate rows and L10 source faces remain visible through $S, N, O, U, D, H, C, P, L$, so any detached diamond proof, closed-substitution bypass, or missing L10 source row leaves the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSubstitutionDiamondBudget`

Theorem 765.306 (MetaCIC diamond residual budget)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, every confluence or diamond checker that feeds a decidability readback must consume exactly the residual budget displayed by the residual-substitution frontier of Definition 765.304. In particular, the checker route factors through the candidate SN confluence checker of Definition 765.195, the SN confluence-decidability budget of Definition 765.193, and the residual-budget socket route of Definition 765.289 before any bounded decidability consumer may inspect* U*.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDiamondBudget`
↪ variant `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDiamondResidualBudget`

Proof. Read the residual-substitution frontier first. It binds closed substitution, residual budget, and candidate-mediated SN rows to the same packet route, so the confluence checker has no separate budget source.

Now apply the candidate SN confluence checker factorization. The checker may combine candidate-mediated normalization, confluence handoff, and bounded checker rows only through $S, N, O, U, D, H, C, P, L$ and the displayed L10 faces.

Finally apply the SN confluence-decidability budget. Its decidability-facing consumer reads the same handoff row U only after the candidate and confluence budget rows are visible. Since the residual-budget socket route is already attached to those rows, the diamond checker consumes that budget before any decidability readback is exposed. □

Theorem 765.307 (MetaCIC residual decidability cut)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every bounded decidability readback that consumes the diamond checker must first expose the residual-substitution frontier, the candidate SN confluence checker, the SN confluence-decidability budget, and the residual-budget socket route on the same*

packet route. Consequently no decidability consumer may inspect U from a detached confluence proof, closed-substitution bypass, or checker route that omits any of those displayed rows.

Proof. Start from the residual-substitution frontier of Definition 765.304. It places the closed-substitution boundary, residual budget, and candidate-mediated SN rows on the same accepted packet before any confluence-facing or decidability-facing read may consume U.

Attach the diamond budget of Definition 765.306. That theorem forces any confluence or diamond checker feeding a decidability readback to consume the residual budget exposed by the frontier.

Now read the checker and budget routes. The candidate SN confluence checker factorization, Definition 765.195, supplies the only checker interface, while Definition 765.193 binds the bounded decidability readback to the same SN confluence-decidability budget.

Finally use the residual-budget socket route of Definition 765.289. Since the socket route is already part of the checker budget, a consumer that reads U from detached confluence, bypassed closed substitution, or an omitted residual route leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDecidabilityCut`

Theorem 765.308 (MetaCIC residual diamond source totality)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual diamond checker that feeds bounded conversion must expose the closed-substitution boundary, the residual-budget socket route, the candidate-mediated strong-normalization frontier, and the L10 source rows on the same packet route before reading U. Consequently such a checker cannot certify a bounded conversion row from a detached residual substitution proof, a missing candidate-mediated SN row, a confluence budget outside D, H, C, P, L, or a source route absent from S, N, O.*

Proof. Begin with the residual-substitution frontier of Definition 765.304. It places closed substitution, residual budget, and candidate-mediated strong normalization on the accepted route before any confluence-facing read can consume U.

Attach the diamond budget of Definition 765.306. The diamond checker has no separate budget coordinate, so its confluence-facing row must consume the same residual-budget socket route and the same SN confluence-decidability budget.

Use the residual decidability cut of Definition 765.307. A bounded decidability readback may inspect U only after the residual-substitution frontier, candidate SN confluence checker, SN confluence-decidability budget, and residual-budget socket route have all been displayed on one packet.

Finally apply the candidate-mediated SN frontier of Definition 765.198. The L10 source faces remain visible through $S, N, O, U, D, H, C, P, L$, so omitting any source row or moving the checker to a detached route exits the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondSourceTotality`

Theorem 765.309 (MetaCIC residual diamond L10 source separation)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual diamond checker that feeds bounded conversion must preserve the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source faces*

as distinct displayed rows before it reads U. The checker may pass those faces through the candidate-mediated and residual-budget routes, but it may not merge them into a single Real-readback row or reconstruct one face after bounded conversion has consumed the handoff.

Proof. Start with Definition 765.305. It forces any diamond, confluence, or bounded decidability consumer to expose the residual-substitution frontier, candidate-mediated SN frontier, residual-budget socket route, and L10 source rows on the same packet route.

Use the residual decidability cut of Definition 765.307. A bounded conversion readback can inspect U only after the residual frontier, checker route, SN confluence-decidability budget, and residual-budget socket have been displayed together.

Finally apply Definition 765.198 and Definition 765.308. The candidate-mediated frontier keeps the L10 source faces visible with the candidate rows, and source totality requires those faces before bounded conversion reads U. Therefore the four source faces remain separated as displayed rows throughout the residual diamond route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondL10SourceSeparation`

Theorem 765.310 (MetaCIC residual diamond candidate-budget coverage)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet $K = (S, N, O, U, D, H, C, P, L)$, the candidate-mediated strong-normalization rows, residual-budget socket rows, and L10 source rows jointly cover the bounded conversion consumer route. A bounded conversion consumer may read U only after all three families are displayed on the same packet route; omitting any one family leaves no accepted residual diamond checker for the consumer.*

Proof. Begin with Definition 765.305. It names the combined route through the residual-substitution frontier, candidate-mediated strong-normalization frontier, SN confluence-decidability budget, residual-budget socket route, and L10 source rows.

Attach Definition 765.307. The cut prevents bounded conversion from reading U through a detached confluence proof or a route that omits the checker, budget, or residual socket.

Now use Definition 765.198 together with Definition 765.308. The candidate rows supply the strong-normalization frontier, the socket rows supply the residual budget, and the L10 rows supply the finite Real and completion source faces. These three families exhaust the bounded conversion consumer route without adding full confluence, beta-conversion decidability, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or kernel consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondCandidateBudgetCoverage`

Theorem 765.311 (MetaCIC residual-diamond budget source nonescape)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual diamond checker feeding bounded conversion must expose the residual-substitution frontier, residual-budget socket route, candidate-mediated strong-normalization frontier, SN confluence-decidability budget, and L10 source rows on the same packet route before reading U. A detached residual proof, missing candidate row, missing socket budget, or absent L10 source row leaves the accepted packet.*

Proof. Start with the residual-substitution frontier of Definition 765.304. It binds the closed-substitution boundary, residual budget, and candidate-mediated strong-normalization rows to the accepted route before a confluence-facing consumer may inspect U.

Attach the diamond budget of Definition 765.306. The diamond checker must spend the same SN confluence-decidability budget and residual-budget socket route carried by D, H, C, P, L, so it cannot import an external budget.

Use the residual decidability cut of Definition 765.307. Bounded conversion may read U only after the residual frontier, checker route, SN confluence-decidability budget, and socket budget are displayed on one packet.

Finally apply residual diamond source totality, Definition 765.308. The L10 source rows remain visible through $S, N, O, U, D, H, C, P, L$, so detached residual proofs, missing candidate rows, missing socket budgets, and absent source rows are outside the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondBudgetSourceNonescape`

Theorem 765.312 (MetaCIC residual diamond candidate-mediated handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any consumer that uses a residual diamond checker to feed bounded conversion must expose, on one route through* $S, N, O, U, D, H, C, P, L$*, the candidate-mediated strong-normalization frontier, the residual-substitution frontier, the residual-budget socket route, the SN confluence-decidability budget, and the L10 source rows before reading* U*. A detached residual proof, a missing candidate row, a missing socket budget, or an absent L10 source row leaves the accepted packet.*

Proof. Start from residual diamond source totality, Definition 765.308. It already forces the closed-substitution boundary, residual-budget socket route, candidate-mediated strong-normalization frontier, and L10 source rows onto the same packet route before a bounded-conversion consumer may inspect U.

Next use the residual-substitution frontier of Definition 765.304. This supplies the closed-substitution and candidate-mediated rows needed by the residual checker and prevents a residual proof from being read on a detached route.

Attach the diamond budget and residual decidability cut, Definition 765.306 and Definition 765.307. Together they place the SN confluence-decidability budget and residual-budget socket beside the bounded conversion readback.

Finally apply the candidate-mediated SN frontier, Definition 765.198. The candidate rows and L10 source faces remain visible through $S, N, O, U, D, H, C, P, L$, so a missing candidate row, missing socket budget, or absent source row is outside the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondCandidateMediatedHandoff`

Theorem 765.313 (MetaCIC residual diamond L10 source nonescape)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual diamond checker feeding bounded conversion must preserve, on the same packet route, the L10 source faces from* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before reading the candidate-mediated handoff row* U*. Consequently a checker that detaches the residual substitution frontier, omits the diamond budget, bypasses the residual decidability cut, drops the candidate-mediated SN frontier, or reads only residual diamond source totality without those L10 faces exits the accepted packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondL10SourceNonescape`

Proof. Start with the residual-substitution frontier of Definition 765.304. It places closed substitution, residual budget, and candidate-mediated normalization on one route before a confluence-facing consumer may read U.

Attach the diamond budget of Definition 765.306. The residual diamond checker has no independent budget coordinate; it must consume the same SN confluence-decidability and residual-budget rows carried by D, H, C, P, L.

Use the residual decidability cut of Definition 765.307. A bounded conversion readback can inspect U only after the residual frontier, checker factorization, SN confluence-decidability budget, and residual-budget socket are visible on the same packet route.

Now apply the candidate-mediated SN frontier of Definition 765.198. It keeps the L10 source faces from $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible with the candidate rows rather than allowing them to be reconstructed after the handoff.

Finally use residual diamond source totality, Definition 765.308. Since source totality already requires those faces before bounded conversion reads U, any checker that loses one of them or moves it to a detached route is outside the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Theorem 765.314 (MetaCIC residual diamond typed L10 budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A typed residual-diamond consumer may read bounded conversion only after the residual-substitution frontier, residual-budget socket, candidate-mediated strong-normalization frontier, typed-example boundary, and all four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

remain on the same packet route. Thus the typed conversion read cannot detach its residual proof from the closed-substitution boundary, spend a socket budget outside D, H, C, P, L, omit the candidate-mediated SN frontier, or replace any L10 source face by a later reconstruction.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondTypedL10Budget`

Proof. Start with residual diamond source totality, Definition 765.308. It puts the closed-substitution boundary, residual-budget socket, candidate-mediated frontier, and L10 source rows on one route before bounded conversion may read U.

Use residual diamond L10 source nonescape, Definition 765.313, to retain $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as displayed source faces. The typed-example boundary is then read through the same candidate-mediated rows, so the bounded conversion consumer receives one finite budget rather than a detached typed proof. □

Theorem 765.315 (MetaCIC residual diamond source-face retention)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet $K = (S, N, O, U, D, H, C, P, L)$, the residual-diamond route retains the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces whenever the residual-substitution frontier, residual-budget socket, candidate-mediated SN frontier, and typed-example boundary are used to feed bounded conversion. A consumer that reads one of those faces through a detached route, or reads bounded conversion after dropping one face, leaves the accepted packet.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondSourceFaceRetention`

Proof. Apply the residual diamond L10 source nonescape theorem. It already prevents the bounded conversion handoff from consuming U after losing any of the four L10 source faces.

Now attach residual diamond source totality. Source totality supplies the same closed-substitution, socket-budget, and candidate-mediated rows required by the residual diamond route. The typed-example boundary can therefore be retained only as part of the displayed packet route; it has no separate source-face reconstruction step. □

Theorem 765.316 (MetaCIC residual diamond bounded conversion nonescape)**.** *Let $K = (S, N, O, U, D, H, C, P,$ be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Bounded conversion may be read from the residual diamond checker only while the residual-substitution frontier, residual-budget socket, candidate-mediated strong-normalization frontier, typed-example boundary, and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *remain on the same route before U is consumed. The read exports no full confluence theorem, no beta-conversion decidability, no quotient equality of terms, no selected limit, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondBoundedConversionNonescape`

Proof. Compose residual diamond source totality with residual diamond L10 source nonescape. The first theorem supplies the closed-substitution, socket-budget, and candidate-mediated rows needed before bounded conversion can read U.

The second theorem keeps the four L10 source faces visible on that route. Since the typed-example boundary is also consumed before the handoff, any consumer missing one of those rows is outside the accepted packet, and the conclusion is limited to bounded conversion on the displayed finite budget. □

Theorem 765.317 (MetaCIC residual diamond four-face budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A residual diamond checker feeding bounded conversion may consume the candidate-mediated strong-normalization handoff only when the residual-substitution frontier, residual-budget socket, candidate-mediated frontier, and all four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

remain displayed on the same packet route $S, N, O, U, D, H, C, P, L$. The handoff does not export full subject reduction, full confluence, beta-conversion decidability, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Proof. Start with the residual-substitution frontier of Definition 765.304. It places the closed-substitution boundary, residual budget, and candidate-mediated normalization rows on one accepted route before a diamond-facing consumer may read U.

Next apply residual diamond source totality, Definition 765.308. The bounded-conversion checker must expose the closed-substitution boundary, residual-budget socket route, candidate-mediated strong-normalization frontier, and L10 source rows before it consumes the handoff.

Use the residual diamond L10 source nonescape theorem Definition 765.313. It keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible on the same route and rules out a detached checker that reconstructs any source face after reading U.

Finally apply the candidate-mediated SN frontier of Definition 765.198. The candidate-mediated handoff remains a bounded route through the displayed packet, so the theorem exports only the four-face budget and not subject reduction, full confluence, beta-conversion decidability, quotient term equality, selected limits, ambient Real completeness, or kernel consistency. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondFourFaceBudget

Theorem 765.318 (MetaCIC residual-substitution diamond budget typed boundary)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any typed residual-substitution diamond consumer that reads the bounded conversion handoff must first keep the closed-substitution boundary, residual-budget socket, candidate-mediated strong-normalization frontier, typed-example boundary, and L10 source faces on the same packet route* $S, N, O, U, D, H, C, P, L$*. Consequently the bounded conversion handoff cannot be read from a detached typed example, a residual diamond proof outside the socket budget, a candidate frontier without typed examples, or an L10 source route absent from* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the residual-substitution frontier of Definition 765.304. It puts the closed-substitution boundary, residual-budget rows, and candidate-mediated normalization rows on the accepted packet route before a confluence-facing handoff can consume U.

Attach the diamond budget of Definition 765.306. The residual diamond checker must spend the same residual budget that feeds the bounded conversion route, so it cannot move the checker outside the socket-facing rows.

Use the residual decidability cut Definition 765.307. It forces the bounded conversion readback to expose the residual frontier, candidate checker, SN confluence budget, and residual-budget socket on one packet.

Now apply the candidate-mediated SN frontier Definition 765.198. The typed-example boundary is read through the same candidate rows, so a typed consumer cannot detach examples from the strong-normalization frontier.

Finally use the L10 source nonescape theorem Definition 765.313. The $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces remain visible before U is read; any handoff missing one of the closed-substitution, residual-budget, candidate-mediated, typed-example, or L10 source faces is outside the accepted packet. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSubstitutionDiamondBudgetTypedBou

Definition 765.319 (MetaCIC candidate residual stability surface)**.** *The candidate-mediated residual stability surface of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L)$$

is the finite BHist *route that reads candidate-mediated strong-normalization evidence through* S, N, U, H, C, P, L *while keeping the residual substitution and diamond budget of Definition 765.305 attached to the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The classifier compares accepted stability surfaces only through the displayed residual rows, candidate frontier rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *data. It has no coordinate for arbitrary strong normalization, global confluence, closed subject reduction, quotient term equality, selected stream limits, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualStabilitySurface

Theorem 765.320 (MetaCIC candidate residual stability)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, any candidate-mediated handoff that consumes residual substitution evidence must keep the residual diamond budget and the four L10 source faces visible until the handoff reaches the confluence-decidability boundary. The stability claim is only over the displayed rows of Definition 765.319; it does not export arbitrary strong normalization, global confluence, beta-conversion decidability, closed subject reduction, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualStability`

Proof. Start with the candidate-mediated frontier theorem, Definition 765.198. It identifies the finite handoff through S, N, U, H, C, P, L and keeps the obstruction and socket rows visible.

Apply the residual substitution diamond budget of Definition 765.305. Residual replay is therefore read as a finite budget attached to the same displayed packet, not as a global confluence theorem.

Use the L10 entry and exit surfaces, Definition 765.118 and Definition 765.128. They keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible while the residual rows are consumed.

The structural rows H, C, P, L transport, replay, source, and name the same finite surface. Since none of the listed coordinates supplies the excluded metatheoretic or real-completion principles, the candidate-mediated residual handoff is stable exactly over the displayed L10 source faces. □

Theorem 765.321 (MetaCIC candidate real-regular handoff)**.** *Every residual-stability handoff of Definition 765.319 that reaches a real-facing source row must first pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic face, the* $\mathsf{StreamName}^{\uparrow}$ *observation face, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback face before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. Candidate-mediated strong-normalization evidence may be replayed along this route, but it cannot replace the regular readback by host equality, selected limits, quotient stream equality, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateRealRegularHandoff`

Proof. Read the four L10 source faces in their displayed order. The dyadic arithmetic face supplies endpoint and radius data, the stream face supplies finite observation windows, and the regular-rational face supplies the readback row used before the real seal is opened.

Apply Definition 765.320. Candidate-mediated residual replay remains attached to the same L10 faces, so it cannot jump directly from a syntactic normalization row to a real-facing conclusion.

The $\mathsf{Real}^{\uparrow}$ boundary is therefore downstream of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$. The packet has no coordinate for host equality, selected limits, quotient stream equality, or ambient completeness, so the handoff remains a finite regular readback route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRealRegularHandoff`

Theorem 765.322 (MetaCIC candidate frontier nonescape)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, the candidate-mediated residual stability surface may be consumed only through the displayed frontier rows* S, N, U, H, C, P, L*, the obstruction and socket rows* O, D*, the residual substitution and diamond budget, and the L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. It cannot be used to erase the obstruction row, fill the discharge socket, or export closed subject reduction, arbitrary strong normalization, full confluence, beta-conversion decidability, quotient term equality, selected limits, host equality, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierNonescape`

Proof. Begin with obstruction nonescape, Definition 765.207. It keeps O and D visible while candidate-mediated strong-normalization evidence is consumed.

Apply candidate residual stability, Definition 765.320. The residual substitution and diamond rows remain attached to the four L10 source faces and to the same frontier packet.

Use the real-regular handoff, Definition 765.321. Any real-facing read must pass through the dyadic, stream, and regular-rational faces before $\mathsf{Real}^{\uparrow}$ is consumed.

Once those displayed rows have been consumed, no remaining coordinate can supply the excluded global metatheoretic or real-completion principles. The frontier therefore closes only the finite residual-stability route named above and no larger claim. □

Theorem 765.323 (MetaCIC L10 residual dyadic route)**.** *Every candidate-frontier residual read used by* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *passes through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before it can be consumed by the residual substitution frontier. The route is part of the finite stability surface of Definition 765.319; it is not a quotient-stream equality, selected limit, host real equality, global normalization theorem, or ambient Real-completion principle.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ResidualDyadicRoute`

Proof. Start with candidate residual stability, Definition 765.320. It keeps residual substitution and diamond-budget reads attached to the displayed packet and to the four L10 source faces.

Apply the real-regular handoff of Definition 765.321. Any real-facing source row must first expose the dyadic arithmetic face, then the stream observation face, then the regular-rational readback face, and only then the $\mathsf{Real}^{\uparrow}$ seal.

Finally use candidate frontier nonescape, Definition 765.322. Since the frontier can consume only the displayed residual rows, obstruction and socket rows, residual budget, and L10 faces, the dyadic tolerance ledger, finite observation window, regular readback, and seal remain visible until the residual substitution frontier consumes them. □

Theorem 765.324 (MetaCIC candidate residual L10 frontier route)**.** *Every candidate-frontier residual read used by* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *must expose* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data,* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal before the residual substitution frontier consumes it. The obstruction and discharge socket rows remain visible throughout the route, so the candidate-mediated frontier cannot replace the displayed L10 read by quotient stream equality, selected limits, host equality, ambient Real-completeness, arbitrary strong normalization, closed subject reduction, or global confluence.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualL10FrontierRoute`

Proof. Begin with candidate residual stability, Definition 765.320. It keeps candidate-mediated strong-normalization evidence attached to the displayed residual rows and to the four L10 source faces.

Apply the real-regular handoff of Definition 765.321. A real-facing residual read therefore reaches $\mathsf{Real}^{\uparrow}$ only after the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows have been displayed.

Use candidate frontier nonescape, Definition 765.322. The obstruction and discharge socket rows remain visible while the residual frontier consumes the read, so the candidate route cannot erase or fill them.

Finally apply the L10 residual dyadic route, Definition 765.323. It identifies the dyadic tolerance ledger, finite observation window, regular readback, and Real seal as the visible route consumed by

the residual substitution frontier, excluding quotient stream equality, selected limits, host equality, ambient Real-completeness, and the larger metatheoretic principles listed above. □

Theorem 765.325 (MetaCIC L10 candidate discharge axis)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that discharges a candidate-mediated strong-normalization, confluence, or decidability obligation along the open L10 route must expose the candidate frontier row, a residual or substitution budget row, a typed-example coverage or gallery row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *before the discharge can be public. The typed-example coverage and gallery rows are witnesses inside this finite route; they are not subject reduction, confluence, normalization, or decidability for all terms.*

Proof. Begin with candidate residual stability, Definition 765.320. It keeps candidate-mediated evidence attached to the displayed frontier rows and to the residual substitution budget.

Use the real-regular handoff and frontier nonescape, Definition 765.321 and Definition 765.322. These force every real-facing candidate read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal while leaving the obstruction and discharge socket visible.

Apply the L10 residual dyadic route and the candidate residual L10 frontier route, Definition 765.323 and Definition 765.324. They identify the dyadic tolerance ledger, finite observation windows, regular readback, real seal, and residual frontier as the public route.

The typed-example coverage or gallery row can therefore only be replayed as a displayed witness along that route. Transport, continuation replay, provenance, and local naming preserve the same packet, so no row turns examples into global subject reduction, confluence, normalization, or decidability for all terms. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10CandidateDischargeAxis`

Theorem 765.326 (MetaCIC candidate residual budget interface)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

any candidate-mediated residual substitution, bounded confluence, or decidability read that reaches the critical-path normal-form endpoint must first expose the four L10 source rows

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

through the open-phase source ledger of Definition 765.155. The candidate rows, residual substitution rows, SN confluence-decidability budget, residual-budget socket, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row remain visible before any critical-path consumer reads the endpoint* N*. The interface exports no closed subject reduction, global confluence, beta-conversion decidability, full consistency, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the open-phase source ledger. Definition 765.155 lists the packet rows $S, N, O, U, D, H, C, P, L$ together with the four L10 source rows $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Attach the candidate residual stability surface, Definition 765.320. It keeps candidate-mediated residual substitution attached to the same four source faces and prevents the residual handoff from becoming a global confluence or decidability theorem.

Use the residual-confluence budget bridge, Definition 765.63, together with the SN confluence-decidability budget of Definition 765.193. These routes force residual substitution, bounded confluence, and decidability consumers through one finite ledger before any normal-form endpoint read is exported.

Finally apply the frontier nonescape route, Definition 765.322. It leaves only the displayed candidate rows, residual rows, socket budget, L10 source faces, transport, replay, provenance, and local naming. Since none of those rows carries the excluded metatheoretic or real-completion principles, the budget interface exposes exactly the stated source rows before N can be read. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualBudgetInterface`

Theorem 765.327 (MetaCIC critical-path candidate residual discharge budget)**.** *For every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet*

$$K = (S, N, O, U, D, H, C, P, L),$$

a bounded residual-substitution discharge may consume the critical-path normal-form endpoint N only after the open-phase source ledger of Definition 765.155, the candidate residual stability surface of Definition 765.320, the residual-confluence budget bridge of Definition 765.63, the SN confluence-decidability budget of Definition 765.193, and the frontier nonescape route of Definition 765.322 are all visible. The discharge read remains a bounded packet route through the displayed source, residual, obstruction, socket, transport, replay, provenance, and local naming rows; it exports no global confluence, beta-conversion decidability, quotient term equality, host equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 765.326. It requires any candidate-mediated residual substitution, bounded confluence, or decidability read reaching N to expose the four L10 source rows through the open-phase source ledger before the endpoint is consumed.

Next use the candidate residual stability surface and the residual-confluence budget bridge. The stability theorem keeps residual substitution attached to the same candidate and L10 source faces, while the bridge keeps confluence-facing reads inside the finite residual ledger rather than turning them into global confluence or beta-conversion decidability.

Then add the SN confluence-decidability budget. That budget forces the strong-normalization, confluence, and decidability rows to remain a bounded handoff through the packet coordinates $S, N, O, U, D, H, C, P, L$, so quotient term equality and host equality have no separate source row.

Finally apply the frontier nonescape route. Its coordinate exhaustion leaves only the displayed candidate, residual, source, obstruction, socket, transport, replay, provenance, and naming rows, so selected limits and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be recovered from the discharge read. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualDischargeBudget`

Theorem 765.328 (MetaCIC candidate-discharge visible residual route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization, confluence, or decidability consumer that reads the critical-path handoff through K must first expose the candidate row, residual-substitution row, normalization-budget row, bounded checker row, obstruction row O, discharge socket D, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}$ *row L. The route also keeps the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *visible until the consumer reaches the handoff.*

The visible-residual route exports no unconditional strong normalization, full Church–Rosser theorem, closed subject reduction, global conversion decidability, quotient syntax equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency claim.*

Proof. Start with candidate residual stability, Definition 765.320. It attaches candidate-mediated evidence to residual-substitution rows, leaves O, D visible, and keeps the four L10 faces on the same finite packet route.

Apply the candidate residual budget interface of Definition 765.326. This exposes the normalization-budget row, residual-budget socket, transport, replay, provenance, and local naming before any normal-form endpoint or critical-path handoff is read.

Finally use the residual diamond budget scope of Definition 765.339 together with the bounded checker factorization of Definition 765.195. These supply the bounded checker and local residual-diamond route while preserving the same candidate, obstruction, socket, L10, transport, replay, provenance, and NameCert rows. Since the cited surfaces contain none of the excluded global metatheoretic, quotient, completion, or consistency coordinates, the candidate-discharge consumer can read only the displayed visible-residual route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeVisibleResidualRoute`

Theorem 765.329 (MetaCIC visible residual socket exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization, confluence, or decidability consumer that reads residual rows through the visible route must keep the discharge socket D and obstruction row O displayed until after candidate residual stability, residual budget, residual diamond budget, bounded checker factorization,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *have all been exposed.*

Consequently the visible-residual socket route refuses unconditional strong normalization, global confluence, subject reduction, conversion decidability, quotient term equality, $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, and kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathVisibleResidualSocketExhaustion`

Proof. Start with candidate residual stability from Definition 765.320. It names the residual rows through the candidate surface while keeping the obstruction row O and discharge socket D among the displayed coordinates.

Next apply the candidate residual budget interface of Definition 765.326 and the residual diamond budget scope of Definition 765.339. These rows expose the finite budget ledgers used by the residual consumer, but they do not replace the obstruction or discharge coordinates by a global metatheorem.

Finally use the bounded checker factorization of Definition 765.195. The factorization is read through the same hsame transport, Cont replay, Pkg provenance, and local NameCert rows, so the visible route is exhausted only after O and D remain visible through the full handoff. None of the cited rows exports the excluded normalization, confluence, reduction, decidability, quotient, completion, selection, or consistency claims. □

Theorem 765.330 (MetaCIC visible residual discharge budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated strong-normalization, confluence, or decidability consumer that treats the discharge socket D as filled must first expose candidate residual stability, the candidate residual budget, residual diamond budget scope, bounded*

checker factorization, the four L10 faces DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The filled-socket read exports no unconditional strong normalization, full Church–Rosser theorem, closed subject reduction, global conversion decidability, quotient syntax equality, selected limits, ambient Real$^{\uparrow}$ *completeness, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathVisibleResidualDischargeBudget`

Proof. Start with candidate residual stability from Definition 765.320. It keeps the candidate row attached to the residual-substitution route and leaves the discharge socket visible on the same packet as the four L10 faces.

Next apply the candidate residual budget interface of Definition 765.326. This supplies the finite budget row that a filled-socket consumer must display before reading any normalization, confluence, or decidability handoff.

The residual diamond budget scope of Definition 765.339 then bounds the diamond-facing part of the same route. The bounded checker factorization of Definition 765.195 supplies the checker-facing route without replacing candidate, obstruction, discharge, or L10 data by a global metatheorem.

Finally transport, replay, provenance, and local naming are read through H, C, P, L. Since the cited surfaces contain no unconditional normalization theorem, full Church–Rosser theorem, closed subject-reduction theorem, global conversion decision procedure, quotient syntax equality, selected-limit row, ambient Real$^{\uparrow}$ completeness theorem, or kernel-consistency coordinate, filling D is only the displayed discharge-budget boundary. □

Theorem 765.331 (MetaCIC visible residual local-decidability budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *packet. Any bounded local decidability read through a candidate-mediated residual route must expose candidate residual stability, the candidate residual budget, residual diamond budget scope, bounded checker factorization, the four L10 faces* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *obstruction row O, discharge socket D, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row L before using a decidability-facing conclusion.*

The read exports no global conversion decidability, quotient syntax equality, selected limits, ambient Real$^{\uparrow}$ *completeness, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathVisibleResidualLocalDecidabilityBudget`

Proof. First read the visible residual discharge route. The candidate-discharge route, socket exhaustion, and discharge budget theorems keep the candidate row, residual-substitution route, obstruction row O, discharge socket D, and the four L10 faces visible through the same finite packet.

Next apply the bounded checker factorization of Definition 765.195. It turns the local decidability-facing use into a bounded checker read, not a global conversion procedure.

Finally transport and replay through H, C, P, L. Since the cited rows never introduce quotient syntax equality, selected limits, ambient Real$^{\uparrow}$ completeness, or kernel consistency, the local decidability conclusion is available only inside the displayed finite residual budget. □

Theorem 765.332 (MetaCIC parallel-diamond visible residual consumer)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any consumer that routes a parallel-diamond or confluence handoff through the critical path must expose candidate residual stability, the residual-substitution row, normalization-budget row, bounded checker row, obstruction row* O*, discharge socket* D*, the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* L *before it reads the confluence-facing handoff.*

The parallel-diamond read exports no premise-free confluence, subject reduction, quotient syntax equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondVisibleResidualConsumer`

Proof. Start with the candidate-discharge visible residual route of Definition 765.328. It already exposes the candidate row, residual-substitution row, normalization budget, bounded checker, obstruction row, discharge socket, L10 faces, transport, replay, provenance, and local NameCert before any confluence-facing handoff is consumed.

Use socket exhaustion from Definition 765.329. A parallel-diamond consumer is still a bounded residual consumer, so it cannot hide O or D after naming a diamond-facing handoff.

Apply the discharge budget of Definition 765.330 and the local-decidability budget of Definition 765.331. These theorems keep the same finite residual rows visible for any filled-socket or checker-facing read.

Thus the parallel-diamond route has no additional premise-free confluence, subject-reduction, quotient-syntax, selected-limit, Real-completeness, or consistency coordinate. It can consume only the displayed visible-residual packet before reaching the confluence-facing handoff. □

Theorem 765.333 (MetaCIC residual-diamond local-confluence frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any local-confluence read for* K *must expose the residual-substitution frontier, candidate-mediated strong-normalization frontier, residual-budget socket, bounded confluence checker, and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before it reads* U*. The read is therefore a finite residual-diamond frontier over the displayed packet rows, not a global confluence theorem, beta-conversion decidability result, quotient equality of terms, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 765.305. It already binds residual substitution, candidate-mediated normalization, residual budget, bounded checker data, and L10 source rows to the same accepted packet route before any confluence-facing consumer can use the handoff.

Use residual diamond source totality, Definition 765.308. The local-confluence read may inspect U only after the closed-substitution boundary, residual-budget socket, candidate-mediated frontier, and source rows are all displayed on $S, N, O, U, D, H, C, P, L$.

Finally apply the L10 source separation theorem, Definition 765.309. The four source faces remain distinct rows throughout the residual-diamond route, so a checker that merges them, reconstructs them after reading U, or detaches the residual proof from the bounded budget leaves the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondLocalConfluenceFrontier`

Theorem 765.334 (MetaCIC residual-schedule diamond)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet whose residual frontier and candidate schedule rows are both visible. Then the left and right residual contractions of the displayed critical-pair packet admit a local diamond budget: each contraction must re-enter the same candidate-mediated frontier before any strong-normalization, confluence, or decidability consumer may read* U*. The budget is local to the packet rows* $S, N, O, U, D, H, C, P, L$*; it does not claim global confluence, unconditional normalization, beta-decidability, quotient term equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the residual-diamond local-confluence frontier, Definition 765.333. It already forces every confluence-facing read to display the residual-substitution frontier, residual-budget socket, bounded checker data, and L10 source rows before U can be consumed.

Next use the candidate residual budget interface, Definition 765.337. Candidate rows and residual-budget rows stay attached to the same packet, while O, D and the four L10 faces remain visible.

Finally apply the candidate schedule discharge route of Definition 765.284. Since the schedule can be consumed only as a finite budget route with the discharge socket still displayed, both residual contractions must return through that candidate-mediated frontier before any downstream strong-normalization, confluence, or decidability consumer reads the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualScheduleDiamond`

Theorem 765.335 (MetaCIC candidate-SN confluence handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any local-confluence read that consumes both the residual-diamond frontier and the candidate-mediated strong-normalization frontier must expose* $S, N, O, U, D, H, C, P, L$ *and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before* U *is consumed. The handoff remains a finite candidate-mediated boundary; it does not close subject reduction, arbitrary strong normalization, global confluence, beta-decidability, quotient term equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the residual-diamond local-confluence frontier, Definition 765.333. It requires the residual-substitution frontier, residual-budget socket, bounded checker data, and L10 source faces to remain visible before a local-confluence consumer reads U.

Next use the candidate-mediated strong-normalization frontier, Definition 765.198. Candidate evidence is read only as packet data on the displayed rows together with the same four L10 source faces, not as a closed normalization theorem.

Finally apply the L10 source separation theorem, Definition 765.309. Since the four source faces stay distinct and the residual and candidate rows remain attached to $S, N, O, U, D, H, C, P, L$, the handoff cannot export any of the excluded global metatheoretic, quotient, selected-limit, or Real-completeness principles. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateSNConfluenceHandoff`

Theorem 765.336 (MetaCIC residual-diamond source determinacy)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any local-confluence read that identifies the candidate-mediated strong-normalization frontier with the residual-diamond frontier must preserve* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as four distinct L10 source rows before it consumes* U*. If the read merges those source faces, reconstructs one of them only after reading* U*, or treats their separation as an external side condition, then the read is outside the accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet.*

Proof. Apply the residual-diamond local-confluence frontier, Definition 765.333. It forces any confluence-facing read to display the residual-substitution frontier, residual-budget socket, bounded checker, and the four L10 source faces before U is available.

Next apply the candidate-SN confluence handoff, Definition 765.335. The candidate-mediated frontier and residual-diamond frontier are therefore consumed only as rows attached to the same displayed packet $S, N, O, U, D, H, C, P, L$.

Finally use the L10 source separation theorem, Definition 765.309. The four source faces remain separate rows throughout that shared route, so a read that merges them, rebuilds them after U, or places their separation outside the packet cannot be an accepted local-confluence read for K. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondSourceDeterminacy`

Theorem 765.337 (MetaCIC candidate residual-diamond lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any local-confluence read that uses both candidate-mediated strong-normalization data and residual-diamond data must keep the candidate rows, residual-budget rows, obstruction row* O*, discharge socket* D*, and the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *visible before* U *is consumed. The candidate frontier may support the handoff, but it cannot fill* D *or export closed subject reduction, global confluence, beta-decidability, quotient term equality, selected limits, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the residual-diamond local-confluence frontier, Definition 765.333. It places residual substitution, the residual-budget socket, bounded checker data, and the four L10 source faces before any local-confluence consumer may read U.

Next apply the candidate-SN confluence handoff, Definition 765.335, and residual-diamond source determinacy, Definition 765.336. These results force the candidate-mediated and residual-diamond rows to remain attached to the same packet $S, N, O, U, D, H, C, P, L$ while preserving the four L10 faces as separate source rows.

Now use candidate residual stability and frontier nonescape, Definition 765.320 and Definition 765.322. The candidate rows may carry the finite handoff, but they remain a displayed frontier over the same residual budget and cannot erase O, fill D, or detach the L10 source route.

Finally apply subject-reduction socket separation, Definition 765.27. Since D is a separate socket row and no displayed candidate or residual row is a subject-reduction certificate, the local-confluence read cannot export any of the excluded global metatheoretic, quotient, selected-limit, host-equality, or Real-completeness principles. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualDiamondLock`

Theorem 765.338 (MetaCIC critical-path candidate residual local-diamond budget)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any candidate-mediated residual read that can feed a local-diamond or local-confluence consumer must expose residual substitution, the local diamond boundary, the bounded checker row, the obstruction row* O*, the discharge socket* D*, and the four L10 source rows* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any confluence or decidability consumer reads the packet. The budget is a finite candidate-residual precondition; it is not a global Church–Rosser theorem, subject-reduction theorem, beta-decision procedure, quotient term equality, selected limit, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateResidualLocalDiamondBudget`

Proof. Start from the residual-diamond local-confluence frontier of Definition 765.333. It already forces residual substitution, residual-budget socket data, bounded checker data, and the four L10 source faces to appear before any local-confluence consumer reads U.

Next use the candidate residual-diamond lock of Definition 765.337. Candidate rows may support the finite handoff, but they remain attached to $S, N, O, U, D, H, C, P, L$ and cannot erase the obstruction row or fill the discharge socket.

Finally apply source determinacy, Definition 765.336. Since the four L10 source faces remain distinct rows and the bounded checker is read before the consumer-facing packet, the advertised budget is exactly a local candidate-residual precondition and does not export any of the excluded global metatheoretic or Real-completeness conclusions. □

Theorem 765.339 (MetaCIC critical-path residual diamond budget scope)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A candidate-mediated residual diamond read may enter the public critical-path route only after the packet exposes the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and L10 source rows on the same route through* $S, N, O, U, D, H, C, P, L$*. The read is therefore a finite budget scope for the displayed packet; it is not a global strong-normalization theorem, global confluence theorem, beta-conversion decision procedure, quotient term equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondBudgetScope`

Proof. Start with the residual-substitution diamond budget of Definition 765.305. It places residual substitution, candidate-mediated strong normalization, residual-budget socket data, and L10 source rows on one accepted packet route before a diamond or decidability consumer may read the handoff.

Next use the candidate residual budget interface, Definition 765.326. This forces the candidate rows and the four L10 source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to remain visible before a critical-path endpoint is read.

Apply the residual-diamond local-confluence frontier together with the candidate-SN confluence checker factorization, Definition 765.333 and Definition 765.195. These theorems supply the local join ledger and checker factorization while keeping the checker inside the same finite packet route.

The displayed rows exhaust the residual diamond budget scope. Since no row carries global SN, global confluence, beta-decidability, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness,

or full consistency, the public MetaCICCriticalPath route can consume only the shown finite budget. □

Theorem 765.340 (MetaCIC residual diamond source-order lock)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every accepted residual-diamond consumer must read the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any confluence or decidability output is available. The locked route exports no full strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or consistency row.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondSourceOrderLock`

Proof. Start from Definition 765.339. It already places the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and L10 source rows on one accepted packet route before a residual-diamond consumer can read the result.

Use the candidate-mediated L10 handoff, Definition 765.341. That handoff names the four source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and keeps them visible before local confluence or decidability is exported.

Apply the locality obstruction, Definition 765.342. If any L10 face is absent, the consumer has no packet coordinate from which to export the candidate-mediated local-confluence or decidability output.

Finally use the residual budget frontier, Definition 765.343. It repacks the same route as a finite frontier through $S, N, O, U, D, H, C, P, L$, and none of those rows carries full SN, global confluence, beta-decidability, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or consistency. □

Theorem 765.341 (Candidate-mediated residual diamond L10 handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any accepted route that consumes the residual diamond budget and the candidate-SN confluence checker must expose the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before exporting local confluence or decidability consumers. The handoff exports no global strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient term equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or full consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateMediatedResidualDiamondL10Handof`

Proof. Start from Definition 765.339. It already places the residual substitution budget, candidate schedule, local join ledger, confluence checker factorization, and L10 source rows on one accepted packet route.

Use the candidate residual budget interface, Definition 765.326. This keeps the candidate-mediated rows and the four L10 faces visible before a critical-path endpoint is read.

Now apply the checker factorization, Definition 765.195, together with the residual-diamond local-confluence frontier, Definition 765.333. These results route local confluence and decidability consumers through the displayed checker and local join ledger rather than through a detached global theorem.

Thus any accepted residual-diamond handoff must first display $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same finite packet. Since none of the rows in K carries global SN, global confluence, beta-decidability, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or full consistency, those exports remain outside the handoff. □

Theorem 765.342 (MetaCIC residual diamond L10 locality obstruction)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the residual-diamond route for K lacks any one of the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$, *then the route cannot export the candidate-mediated local-confluence consumer or the candidate-mediated decidability consumer. The obstruction is local to the displayed packet route; it is not a global failure of strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondL10LocalityObstruction`

Proof. Apply Definition 765.339. It puts the residual substitution budget, candidate schedule, local join ledger, checker factorization, and L10 source rows on one finite route before any residual-diamond consumer can read the packet.

Now use Definition 765.341. That handoff requires the four source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before local confluence or decidability can be exported.

The same dependency is visible in the named candidate residual budget interface and in the candidate-SN confluence checker factorization, Definition 765.326 and Definition 765.195. Removing any one L10 face breaks the displayed route from candidate budget to checker output, so the consumer has no finite packet coordinate from which to export local confluence or decidability. □

Theorem 765.343 (MetaCIC critical-path residual budget frontier)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. The packet can export a residual-diamond consumer or a decidability consumer only when the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and all four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are displayed on the same finite route through $S, N, O, U, D, H, C, P, L$. The frontier exports no global strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualBudgetFrontier`

Proof. First read the candidate-facing budget. The residual diamond budget scope of Definition 765.339 and the candidate residual budget interface of Definition 765.326 place the schedule, residual substitution budget, and candidate rows on the displayed packet route.

Next read the checker side. The candidate-SN confluence checker factorization of Definition 765.195 supplies the confluence checker route, while Definition 765.341 forces the four L10 faces to remain visible before any local confluence or decidability export.

Finally use the locality obstruction, Definition 765.342. If any one of the four L10 faces is absent from the finite route, the residual-diamond consumer and the decidability consumer have no packet coordinate from which to read their output.

The only surviving frontier is therefore the displayed same-route packet containing the candidate schedule, residual substitution budget, local join ledger, checker factorization, and the four L10 faces. Since no cited row carries global SN, global confluence, beta-decidability, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or consistency, none of those objects is exported. □

Theorem 765.344 (MetaCIC critical-path residual budget factorization)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any accepted packet route exporting a residual-diamond consumer or a decidability consumer factors through the candidate schedule, residual substitution budget, candidate residual budget interface, local join ledger, confluence checker factorization, and the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before the consumer reads the output. The factorization is a finite route inside K and exports no global strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualBudgetFactorization`

Proof. Start from the residual diamond budget scope and the candidate residual budget interface, Definition 765.339 and Definition 765.326. These place the candidate schedule, residual substitution budget, candidate rows, and the four L10 source faces on the same accepted packet route.

Next read the checker side. The residual-diamond local-confluence frontier and the candidate-SN confluence checker factorization, Definition 765.333 and Definition 765.195, supply the local join ledger and checker factorization without leaving the displayed packet.

Finally apply the residual budget frontier, Definition 765.343. It states that a residual-diamond or decidability consumer can be exported only after the schedule, residual substitution budget, local join ledger, checker factorization, and all four L10 faces are present.

The required factorization is therefore the single finite route through those rows of K. Since none of the cited rows carries global SN, global confluence, beta-decidability, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or consistency, none of those objects is exported by the consumer route. □

Theorem 765.345 (MetaCIC critical-path residual budget L10 bridge)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any accepted residual-diamond budget route whose candidate-mediated strong-normalization handoff reaches an L10 completion consumer must factor through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{RealCompletionExactBoundaryUp}$ *row before any confluence or strong-normalization export is public.*

The bridge is a finite same-route dependency inside K. It exports no global strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient term equality, selected completion point, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualBudgetBridge`

Proof. Start from the residual diamond budget scope and the candidate-mediated L10 handoff, Definition 765.339 and Definition 765.341. These place the candidate schedule, residual substitution budget, local join ledger, checker factorization, and four L10 faces on one accepted packet route before any public consumer can read the result.

Next use the locality obstruction and frontier, Definition 765.342 and Definition 765.343. They rule out a residual-diamond consumer whenever any one of the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ faces is missing from the same finite route.

Now refine the completion-facing end of that route through the exact-boundary rows. The RealCompletionExactBoundaryUp route of Definition 1120.4 and Definition 1120.13 requires the StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and terminal Real seal to be displayed together before a completion-facing terminal consumer can read the endpoint.

Finally apply the residual budget factorization of Definition 765.344. Since the MetaCIC route has no coordinate for a selected completion point, quotient term equality, ambient completeness, global confluence, global SN, or kernel consistency, the only public bridge is the finite L10 route through the displayed tolerance, window, readback, seal, and exact-boundary rows. □

Theorem 765.346 (MetaCIC residual completion exactness handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual-diamond budget route for K that reaches a completion-facing consumer must factor through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* RealCompletionExactBoundaryUp *exactness rows before exporting local confluence or decidability.*

The handoff is a finite route inside the displayed packet. It exports no global strong-normalization theorem, global confluence theorem, beta-decidability procedure, quotient term equality, selected completion point, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or kernel consistency claim.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualCompletionExactnessHandoff`

Proof. Start from the residual budget bridge of Definition 765.345. It already places the completion-facing consumer after the DyadicRatCore tolerance ledger, StreamName window, RegSeqRat readback, terminal Real seal, and RealCompletionExactBoundary row.

Next apply the residual budget factorization of Definition 765.344. This keeps the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and four L10 faces on one accepted packet route before any local confluence or decidability consumer is read.

Now use the exact-boundary route, Definition 1120.4 and Definition 1120.13. These theorems force every completion-facing endpoint through the displayed StreamName, RegSeqRat, DyadicRatCore, and terminal Real rows together with the RealCompletionExactBoundary structural rows.

Thus the residual-diamond route can export local confluence or decidability only after the exact completion handoff has been displayed. Since none of the cited rows carries global SN, global confluence, beta-decidability, quotient equality, a selected completion point, ambient Real completeness, or kernel consistency, those objects remain outside the accepted packet route. □

Theorem 765.347 (MetaCIC residual diamond completion nonescape)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. After candidate-mediated residual diamond data for K factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* RealCompletionExactBoundaryUp, *the remaining residual-redex obligation is still read only through the displayed MetaCIC candidate rows, confluence-boundary rows, substitution-boundary rows, and completion-exactness rows of K.*

Consequently the completion-facing handoff does not export closed Church–Rosser, global normalization, quotient term equality, selected completion points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency. It exports only the finite same-route residual-redex obligation visible in the candidate, confluence-boundary, substitution-boundary, and exactness ledgers.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondCompletionNonescape`

Proof. Start with the residual diamond budget scope and the source-order lock, Definition 765.339 and Definition 765.340. These force the candidate schedule, residual substitution budget, local join ledger, confluence checker factorization, and L10 source faces to remain on one displayed packet route.

Apply the candidate-mediated L10 handoff and the locality obstruction, Definition 765.341 and Definition 765.342. They prevent a residual-diamond consumer from being read unless the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces are present on the same route.

Now use the residual budget factorization and the completion exactness handoff, Definition 765.344 and Definition 765.346. The factorization keeps the candidate, confluence-boundary, and substitution-boundary rows visible, while the exactness handoff places the $\mathsf{RealCompletionExactBoundaryUp}$ row at the completion-facing end of the route.

The remaining residual-redex obligation is therefore the same finite obligation inside K. Since none of the cited rows contains closed Church–Rosser, global normalization, quotient term equality, selected completion points, ambient $\mathsf{Real}^{\uparrow}$ completeness, or kernel consistency, none of those objects is exported by the completion-facing handoff. □

Theorem 765.348 (MetaCIC critical-path mature L10 source package)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every consumer that combines the MetaCIC residual budget, confluence budget, and candidate-mediated strong-normalization budget must consume the L10 source ledger only through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *dependency rows. The route may read residual substitution, local diamond data, bounded checker rows, candidate frontier rows, obstruction row* O*, discharge socket* D*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming, but it has no fifth source face, quotient term equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global confluence theorem, global beta-conversion decision procedure, or closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMatureL10SourcePackage`

Proof. Start with the real-completion source budget of Definition 765.296. It places every confluence-facing or decidability-facing consumer downstream of the four L10 source rows and keeps the closure trace, critical-pair ledger, normalization frontier, discharge socket, and budget rows on the same source route.

Next apply the Phase Real source obligations of Definition 765.165. They require the dyadic finite-observation row, stream schedule, regular-rational readback, and Real seal to remain separate before any discharge socket, confluence-decidability handoff, obstruction boundary, or downstream readiness boundary is consumed.

Now attach the candidate-mediated strong-normalization frontier through Definition 765.198. The candidate row is a displayed frontier for the same packet, not a replacement for any of the four L10 source faces.

The visible residual route of Definition 765.328 and the residual diamond budget scope of Definition 765.339 then bind residual substitution, local diamond data, bounded checker rows, obstruction, socket, transport, replay, provenance, and local naming to the same finite route. Since each cited surface keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible as the only L10 source ledger, the combined package exports only that displayed dependency route. □

Theorem 765.349 (MetaCIC critical-path L10 source candidate frontier)**.** *Every candidate-mediated strong-normalization frontier consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must keep the four L10 source rows* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *visible while reading residual substitution, local diamond, bounded checker, obstruction, and discharge socket rows. The frontier may consume those rows as displayed finite evidence attached to the packet, but it may not retype them as a fifth source face, quotient term equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global confluence theorem, global beta-conversion decision procedure, or closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10SourceCandidateFrontier`

Proof. Start with Definition 765.348. It already binds the combined residual, confluence, and candidate-mediated strong-normalization budget to the four displayed L10 source rows.

Apply Definition 765.198. The candidate-mediated frontier is read as a row of the accepted packet, so it cannot replace $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ by a detached normalization theorem.

Use Definition 765.328. Residual substitution and discharge-socket reads remain visible along the same finite route rather than becoming hidden premises of the candidate frontier.

Finally apply Definition 765.339. Local diamond and bounded checker rows are scoped budget data attached to the same packet, while obstruction and socket rows remain displayed. Thus every candidate-mediated frontier consumer keeps the four L10 source rows visible and exports none of the excluded global MetaCIC or completion principles. □

Theorem 765.350 (MetaCIC mature L10 source socket separation)**.** *Every accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that reads the mature L10 package must keep* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as four separate source faces. The visible obstruction rows and discharge socket rows remain undischarged unless the packet also displays a separate closed subject-reduction row. Residual, confluence, and candidate-mediated strong-normalization rows are budget data attached to the source package; they are not themselves a closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMatureL10SourceSocketSeparation`

Proof. Apply Definition 765.348. It binds the mature source package to the four displayed L10 faces and excludes quotient term equality, selected limits, ambient completeness, global confluence, global beta-conversion decision, and closed subject reduction.

Apply Definition 765.349. The candidate frontier may read residual substitution, local diamond, bounded checker, obstruction, and discharge socket rows only while the four L10 source rows remain visible.

The obstruction and discharge socket rows therefore record what is still exposed at the packet boundary. A closed subject-reduction conclusion would require its own displayed row. Without that row, residual, confluence, and candidate-mediated strong-normalization data remain scoped budgets for the source package rather than a proof that subject reduction has been closed. □

Theorem 765.351 (MetaCIC critical-path frontier companion)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *frontier consumer that invokes the L10 source package must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before it uses any candidate-mediated normalization row or residual-diamond row. The companion route keeps those four L10 rows inside the same packet rows* $S, N, O, U, D, H, C, P, L$ *and exports no quotient stream equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global normalization theorem, closed subject-reduction theorem, or detached residual-diamond theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierCompanion`

Proof. Start with the L10 four-object exit certificate of Definition 765.128. It identifies the Phase Real and Completion source surface with exactly the dyadic tolerance row, StreamName finite schedule, RegSeqRat readback, and Real seal attached to the accepted packet.

Apply the mature L10 source package theorem, Definition 765.348. That theorem requires residual-budget, confluence-budget, and candidate-mediated strong-normalization consumers to read the L10 source ledger only through those four displayed rows.

Now place the frontier consumer after that source package. Candidate-mediated normalization rows and residual-diamond rows may be read only while $S, N, O, U, D, H, C, P, L$ and the four L10 source faces remain visible on the same route.

Since the packet contains no coordinate for quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, global normalization, closed subject reduction, or a detached residual-diamond theorem, the frontier companion exports only the displayed L10 dependency route. □

Theorem 765.352 (MetaCIC critical-path mature package consumer)**.** *Every mature-package consumer of an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet reads the cited L10 source ledger only through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before it consumes residual substitution, local diamond, bounded checker, candidate frontier, obstruction, discharge socket, transport, replay, provenance, and local naming rows. In carrier notation the consumer remains on the displayed packet route*

$$K = (S, N, O, U, D, H, C, P, L),$$

together with the four L10 source faces named by the mature package of Definition 765.348. The route exports no quotient stream, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global confluence theorem, global beta-conversion decision procedure, closed subject-reduction theorem, or hidden fifth source face.*

Proof. Start with the mature L10 source package. Theorem 765.348 requires every residual, confluence, and candidate-mediated strong-normalization consumer to keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible as the only L10 source faces.

Next use the L10 source candidate frontier and mature source socket separation. Theorems 765.349 and 765.350 keep residual substitution, local diamond data, bounded checker rows, obstruction rows, and discharge sockets attached to the same displayed packet rather than to a quotient stream or selected limit.

Finally apply the mature consumer synthesis of Definition 765.374. It packages the confluence frontier, normalization frontier, substitution boundary, typed-example row, bounded checker route, and the four L10 faces as one consumer surface. Since none of those cited rows contains a quotient stream, selected limit, ambient completeness theorem, global confluence theorem, global beta-conversion decider, closed subject-reduction theorem, or fifth source face, the mature consumer receives only the displayed finite route. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMaturePackageConsumer

The MetaCICCriticalPath$^{\uparrow}$ mature package status named by the hub mature marker is read through the existing child routes for candidate-mediated strong normalization, confluence, decidability, residual diamond scope, and L10 source packages. In particular, Definition 765.368, Definition 765.348, Definition 765.349, and Definition 765.350 keep the mature read on the displayed packet rows while preserving the open subject-reduction socket, the residual-diamond budget, and the four source faces DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$. The status exports no closed subject reduction, global Church–Rosser theorem, quotient term equality, selected limit, or ambient Real$^{\uparrow}$ completeness.

Theorem 765.353 (MetaCIC critical-path parallel-diamond frontier handoff)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *packet. Suppose a parallel-diamond discharge request for K consumes the candidate-mediated strong-normalization frontier, the residual substitution budget, the local diamond witness, and the L10 four-object source ledger for* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. Then the request factors through the displayed candidate frontier before it can be read as a confluence consumer or a normalization consumer.*

The handoff is a finite same-packet route through $S, N, O, U, D, H, C, P, L$. It does not export subject reduction, global Church–Rosser, host beta equality, evidence-free strong normalization, quotient term equality, selected limits, ambient Real$^{\uparrow}$ *completeness, or kernel consistency.*

Proof. First read the candidate side. The candidate-mediated SN frontier of Definition 765.198 and the candidate discharge frontier of Definition 765.278 place the normalization candidate and its discharge rows on the displayed critical-path packet before any public consumer can use them.

Next attach the residual and diamond budget. The residual substitution diamond budget, the residual-diamond local-confluence frontier, and the residual diamond budget scope keep the residual substitution rows, local join witness, checker factorization, and residual readback on the same finite route.

Finally attach the L10 ledger. The four-object exit certificate and the residual completion exactness handoff force every completion-facing read through DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before a confluence or normalization endpoint can be read.

Thus a parallel-diamond discharge request reaches its endpoint only through the displayed candidate frontier followed by the residual, local-diamond, and L10 rows of K. Since the cited route has no coordinate for subject reduction, global Church–Rosser, host beta equality, evidence-free strong normalization, quotient term equality, selected limits, ambient Real$^{\uparrow}$ completeness, or kernel consistency, none of those claims is exported. □

Lean4 ✓ BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondFrontierHandoffCertificate

Theorem 765.354 (MetaCIC mature L10 parallel-diamond route)**.** *Every parallel-diamond discharge request that reaches the mature* MetaCICCriticalPath$^{\uparrow}$ *route must first read the candidate-mediated strong-normalization frontier, the residual substitution diamond budget, and the L10 four-object source ledger for* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. The resulting route is the displayed dependency path from Definition 765.353 through the mature L10 source package of Definition 765.348; it is not a proof of global Church–Rosser, subject reduction, kernel consistency, quotient term equality, selected limits, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with Definition 765.353. It forces any parallel-diamond discharge request to pass through the candidate-mediated strong-normalization frontier, residual substitution budget, local diamond witness, and four-object L10 source ledger before a confluence or normalization consumer can read it.

Now apply Definition 765.348. The mature package keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as separate source faces while residual, confluence, and candidate-mediated strong-normalization rows remain budget data attached to the same packet.

The two routes have the same carrier $K = (S, N, O, U, D, H, C, P, L)$ and the same structural rows. Composing them therefore gives a single L10-facing dependency route for mature parallel-diamond discharge requests. Since neither cited theorem contains a row for global Church–Rosser, subject reduction, kernel consistency, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the composed route exports none of those claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMatureL10ParallelDiamondRoute`

Theorem 765.355 (MetaCIC residual-diamond candidate readback route)**.** *Every residual-diamond candidate read that reaches the mature* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *route must first pass through the candidate-mediated strong-normalization frontier, the residual substitution budget, the local diamond witness, and the L10 four-object source ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Only after those rows are displayed may the read be consumed as normalization-facing or confluence-facing evidence.*

The route is a finite readback over the same accepted packet $K = (S, N, O, U, D, H, C, P, L)$*. It does not export subject reduction, global Church–Rosser, kernel consistency, quotient term equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a theorem identifying candidate readback with host confluence.*

Proof. Start with the parallel-diamond frontier handoff of Definition 765.353. It forces a discharge request to display the candidate-mediated strong-normalization frontier, residual substitution budget, local diamond witness, and four-object L10 source ledger before the request can be read by a confluence or normalization consumer.

Now apply the mature L10 route of Definition 765.354. It keeps the same four L10 source faces, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, on the mature critical-path package rather than replacing them by an ambient completion or confluence theorem.

Composing the two routes gives the stated residual-diamond candidate readback route. The first theorem supplies the candidate, residual, and local-diamond order; the second theorem keeps the read on the mature L10 source ledger.

The composed route has only the rows of K and the four displayed L10 source faces. Since neither component route contains subject reduction, global Church–Rosser, kernel consistency, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or host confluence equality, the readback exports none of those claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondCandidateReadbackRoute`

Theorem 765.356 (MetaCIC parallel-diamond candidate frontier formal target)**.** *Any formal target for the mature* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet must expose the candidate-mediated strong-normalization frontier, the residual substitution budget, the local diamond witness, and the L10 four-object source ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before it is read as confluence-facing or normalization-facing evidence. The target must keep those four rows visible as the candidate frontier of the accepted packet* $K = (S, N, O, U, D, H, C, P, L)$*; it is not*

a theorem of global Church–Rosser, subject reduction, kernel consistency, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host confluence readback.*

Proof. Use the parallel-diamond frontier handoff of Definition 765.353. It identifies the candidate-mediated strong-normalization frontier, residual substitution budget, local diamond witness, and four-object L10 source ledger as the rows a discharge request must display before any confluence or normalization consumer reads the packet.

Next apply the mature L10 route of Definition 765.354. It keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as source faces on the same mature critical-path package rather than replacing them by a completion or confluence theorem.

Finally apply the residual-diamond candidate readback route of Definition 765.355. It composes the candidate frontier, residual budget, local diamond witness, and L10 ledger into the readback order required by confluence-facing and normalization-facing consumers.

The formal target is therefore exactly the visibility obligation for those displayed rows of K. Since the three component routes contain no row for global Church–Rosser, subject reduction, kernel consistency, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or host confluence readback, the target exports none of those endpoint claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondCandidateFrontierFormalTa`

Theorem 765.357 (MetaCIC parallel-diamond frontier readiness envelope)**.** *A critical-path consumer may read the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *parallel-diamond frontier only after the finite residual witness, candidate-normalization frontier, residual-substitution boundary, L10 source ledger, and closed frontier rows are all displayed. For an accepted packet* $K = (S, N, O, U, D, H, C, P, L)$*, the readiness envelope is the ordered surface consisting of the candidate-mediated strong-normalization frontier, residual substitution budget, local diamond witness, four-object L10 ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and the closed frontier rows already used by the mature critical-path package.*

The envelope is a precondition for confluence-facing and normalization-facing reads. It does not claim full closed Church–Rosser, unconditional substitution compatibility, subject reduction, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICParallelDiamondFrontierReadinessEnvelope`

Proof. First read the parallel-diamond frontier handoff of Definition 765.353. It places the candidate frontier, residual substitution budget, local diamond witness, and L10 source ledger on the displayed packet before any confluence or normalization consumer reads it.

Next read the mature L10 route of Definition 765.354. It keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ visible as source faces of the mature critical-path package, together with the closed frontier rows supplied by Definition 765.348.

Finally read the residual-diamond candidate readback route of Definition 765.355. It composes the residual witness, candidate normalization, residual boundary, local diamond witness, and L10 ledger into the readback order required by the formal target of Definition 765.356.

The resulting envelope is exactly the displayed readiness surface for K. Since the cited routes retain their refused endpoints, the envelope does not become full closed Church–Rosser, unconditional substitution compatibility, subject reduction, quotient term equality, selected limits, ambient completeness, or kernel consistency. □

Theorem 765.358 (MetaCIC parallel-diamond retained-frontier scope)**.** *A candidate-mediated strong-normalization or confluence consumer that reads the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *readiness envelope must retain the residual-substitution budget, the local diamond witness, and the L10 source ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *through every such read. The retained frontier is the same packet surface displayed by the mature L10 parallel-diamond route, the residual-diamond candidate readback route, the parallel-diamond candidate frontier formal target, and the readiness envelope; it is not a claim of full Church–Rosser, subject reduction, quotient term equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Proof. First apply Definition 765.354. It keeps the candidate-mediated frontier, the residual substitution budget, and the four-object L10 ledger on the mature critical-path route.

Next apply Definition 765.355. It makes the local diamond witness and residual readback visible before a normalization-facing or confluence-facing consumer can use the packet.

Then apply Definition 765.356 and Definition 765.357. These two statements turn the same rows into the formal target and readiness precondition for every candidate-mediated read.

Therefore no consumer of the readiness envelope can discard the residual-substitution budget, local diamond witness, or L10 source ledger while still remaining on the accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ frontier. The cited routes contain no row for full Church–Rosser, subject reduction, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or kernel consistency, so none of those endpoints is exported by the retained-frontier scope. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICParallelDiamondRetainedFrontierScope`

Theorem 765.359 (MetaCIC parallel-diamond frontier seed status)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that prepares a strong-normalization or confluence read from the parallel-diamond frontier must first read that frontier through the retained seed/status route. For an accepted packet* $K = (S, N, O, U, D, H, C, P, L)$, *the route consists of the parallel-diamond handoff, the readiness envelope, and the retained-frontier scope; only after these rows have fixed the candidate-mediated strong-normalization frontier, residual-substitution budget, local diamond witness, and L10 source ledger may the consumer pass to an SN-facing or confluence-facing endpoint.*

Proof. Start with Definition 765.353. It forces every parallel-diamond discharge request to display the candidate-mediated frontier, residual budget, local diamond witness, and L10 source ledger before a normalization or confluence consumer reads it.

Next apply Definition 765.357. The readiness envelope records these rows as the seed/status surface of the accepted packet rather than as a hidden global theorem.

Finally use Definition 765.358. It keeps the residual-substitution budget, local diamond witness, and L10 source ledger retained through every candidate-mediated read. Thus SN-facing and confluence-facing consumers can only use the parallel-diamond frontier after passing through the displayed seed/status route. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICParallelDiamondFrontierStatusSeed`

Theorem 765.360 (MetaCIC critical-path parallel-diamond frontier handoff scope)**.** *Every consumer of Definition 765.353 must retain the candidate-mediated strong-normalization frontier, residual-substitution budget, local diamond witness, and L10 source ledger through the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it can read the packet as confluence-facing or normalization-facing evidence. The retained scope is the same accepted packet surface used by the mature L10 route, the residual-diamond candidate readback route, the readiness envelope, and the retained-frontier scope; it is not a claim of global Church–Rosser, subject reduction, quotient term equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or MetaCIC kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondFrontierHandoffScope`

Proof. Start with Definition 765.353. It places the candidate-mediated strong-normalization frontier, residual-substitution budget, local diamond witness, and four-object L10 source ledger on the displayed critical-path packet before a confluence or normalization consumer can read it.

Now apply the mature L10 route of Definition 765.354 and the residual-diamond candidate readback route of Definition 765.355. These two routes keep the same residual and local-diamond rows attached to the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source ledger.

Finally use the readiness envelope and retained-frontier scope, Definition 765.357 and Definition 765.358. They identify the visible rows that any candidate-mediated read must keep through the consumer interface.

Thus a consumer of the handoff can only pass to an SN-facing or confluence-facing endpoint after retaining the displayed frontier rows. Since the cited routes contain no row for global Church–Rosser, subject reduction, quotient term equality, selected limits, ambient Real completeness, or kernel consistency, the scoped handoff exports none of those endpoints. □

Theorem 765.361 (MetaCIC parallel-diamond candidate-mediated SN readback)**.** *Any SN-facing readback that consumes the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *parallel-diamond frontier must preserve the candidate-mediated normalization row, the residual substitution budget, the local diamond witness, and the L10 four-object source ledger for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it reads a confluence endpoint. For an accepted packet* $K = (S, N, O, U, D, H, C, P, L)$*, the readback is therefore a same-packet route through the parallel-diamond handoff scope, the readiness envelope, the retained-frontier scope, and the mature L10 route.*

The route does not claim global Church–Rosser, subject reduction, MetaCIC kernel consistency, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICParallelDiamondCandidateMediatedSNReadback`

Proof. Start with the parallel-diamond frontier handoff scope of Definition 765.360. It requires every consumer of the handoff to retain the candidate-mediated strong-normalization frontier, residual-substitution budget, local diamond witness, and L10 source ledger before reading the packet as confluence-facing or normalization-facing evidence.

Next use the readiness envelope of Definition 765.357. The envelope identifies those rows as the finite precondition for any critical-path read of the parallel-diamond frontier.

Now apply the retained-frontier scope of Definition 765.358. It prevents an SN-facing readback from dropping the residual budget, local diamond witness, or L10 ledger while it remains on the candidate-mediated route.

Finally compose with the mature L10 parallel-diamond route of Definition 765.354. The composed route keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as visible source faces of K, so the SN-facing readback reaches any endpoint only through the displayed frontier and exports none of the excluded global claims. □

Theorem 765.362 (MetaCIC critical-path diamond-frontier consumer)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *read that consumes* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must keep the residual-substitution premise, critical-pair, candidate-join, bounded-check, obstruction, discharge-socket, transport, replay, provenance, and local naming rows visible before the normal-form consistency consumer may inspect the handoff. The read is therefore a candidate-mediated frontier consumer, not a proof of subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDiamondFrontierConsumer`

Proof. First keep the critical-path boundary visible. Apply Definition 765.2 together with Definition 765.3 and Definition 765.25. These theorems route a consistency-facing consumer through the displayed critical-path packet while preserving the obstruction row and the discharge socket before the normal-form endpoint can be read.

Next attach the parallel-diamond frontier. Use Definition 4238.12, Definition 4238.4, Definition 4238.31, and Definition 4238.20. These results keep the residual-substitution premise, critical pair, candidate join, and bounded checker rows inside the $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ packet.

Finally compose the two consumer routes. The frontier handoff of Definition 765.353 and the candidate-mediated strong-normalization socket of Definition 4238.41 place the frontier rows on the same critical-path read before the normal-form consistency consumer inspects the handoff. Since neither component route supplies subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, the composed consumer exports none of those claims. □

Theorem 765.363 (MetaCIC critical-path L10 source-ledger readback)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that reads the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier must expose the L10 source ledger*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before it may use the residual witness, critical pair, local join, bounded checker, obstruction, transport, replay, provenance, and local $\mathsf{NameCert}$ *rows. The readback is therefore a finite source-ledger boundary for the critical path; it is not subject reduction, full Church–Rosser, evaluator equality, quotient equality, global normalization, kernel consistency, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the diamond-frontier consumer route, Definition 765.362. It keeps the parallel-diamond frontier visible before a critical-path consumer may inspect the normal-form-facing handoff.

Apply frontier nonescape, Definition 765.365. This places the L10 source ledger beside the residual-substitution, critical-pair, candidate-join, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Next use the residual handoff and residual witness consumption routes, Definition 765.366 and Definition 765.367. These force the residual witness to be read as displayed frontier data rather than as a global confluence or normalization theorem.

Finally use the checked parallel-diamond frontier handoff, Definition 765.364. Its residual witness, critical pair, local join, bounded checker, obstruction, L10 source ledger, transport, replay, provenance, and local naming rows are exactly the rows exposed by the frontier-handoff route. Since each component route keeps the L10 rows displayed and none supplies subject reduction, full Church–Rosser, evaluator equality, quotient equality, global normalization, kernel consistency, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, the composed readback exports none of those endpoints. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10SourceLedgerReadback`

Theorem 765.364 (MetaCIC critical-path parallel-diamond frontier consumer)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source-ledger read that consumes the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-join row, candidate-normalization row, critical-pair row, obstruction-retention row, and finite handoff row before exporting any strong-normalization, confluence, or decidable-conversion budget. The consumer route is therefore a finite critical-path dependency over the parallel-diamond frontier, not a discharge of residual substitution, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the critical-path side. Use Definition 765.362 and Definition 765.365. These results force the source ledger to keep the parallel-diamond frontier visible before any normal-form, confluence, or normalization consumer can read the handoff.

Next read the frontier packet itself. The residual-substitution scope, critical-pair exhaustion, local-join scope, closed-checker retention, and candidate-mediated strong-normalization socket named in the MetaCICParallelDiamondFrontier$^{\uparrow}$ chapter display the residual-join, candidate-normalization, critical-pair, obstruction-retention, and bounded handoff rows as finite carrier data.

Finally compose the two ledgers. Since the critical-path consumer receives only those displayed rows through transport, replay, provenance, and local naming, the exported budget remains downstream of the frontier handoff. No row in the composed route supplies residual-substitution discharge, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, decidable conversion, strong normalization, or kernel consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathParallelDiamondFrontierHandoff`

Theorem 765.365 (MetaCIC critical-path frontier consumer nonescape)**.** *Any normal-form, confluence, or normalization consumer of the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *frontier inside a* MetaCICCriticalPath$^{\uparrow}$ *read must first expose the residual-substitution, critical-pair, candidate-join, bounded-check, obstruction, discharge-socket, L10 source ledger, transport, replay, provenance, and local naming rows. Consequently the frontier exports no subject reduction, global Church–Rosser theorem, evaluator equality, quotient term equality, kernel consistency, selected limits, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathFrontierConsumerNonescape`

Proof. Start from Definition 765.362. It places the parallel-diamond frontier on the critical-path read before any normal-form consistency consumer can inspect the handoff.

Use the parallel-diamond handoff theorems cited there. The residual-substitution, critical-pair, candidate-join, and bounded-check rows remain visible as finite frontier data rather than becoming a global confluence theorem.

Then combine the mature L10 source package already named by the critical-path route with the obstruction, discharge-socket, transport, replay, provenance, and local naming rows. This is a carrier inventory: each downstream consumer receives only the displayed rows, so none of the refused global metatheoretic or Real-completeness endpoints can escape the packet boundary. □

Theorem 765.366 (MetaCIC critical-path residual handoff)**.** *Any accepted* MetaCICCriticalPath$^{\uparrow}$ *read consuming the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must retain the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, checker row, bounded-check row, obstruction row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows named by chapter 4238 before passing the handoff to the critical-path consumer. The residual handoff is therefore a finite frontier route into the critical path, not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualHandoff`

Proof. Start with Definition 765.362 and Definition 765.365. These keep the parallel-diamond frontier visible inside the critical-path read before any normal-form, confluence, or normalization consumer may inspect the handoff.

Next use the public and scoped frontier exports from chapter 4238, in particular Definition 4238.47 and Definition 4238.48. They identify the consumed packet as the finite route through

the residual-substitution premise, critical pair, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Finally compose the two readings. The critical-path consumer receives exactly those retained frontier rows, so the residual premise and obstruction remain visible at the handoff boundary. Since neither component supplies full Church–Rosser, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency, the composed handoff exports none of those endpoints. □

Theorem 765.367 (MetaCIC critical-path residual witness consumption)**.** *Any* MetaCICCriticalPath$^{\uparrow}$ *read that consumes the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *residual handoff must expose a concrete residual witness row, critical-pair row, candidate-join row, bounded checker row, obstruction row,* hsame *transport,* Cont *replay,* Pkg *provenance, local* NameCert *row, and the L10 source ledger before any normal-form or confluence consumer may inspect it. The route therefore does not export subject reduction, global Church–Rosser, evaluator equality, quotient term equality, global normalization, kernel consistency, selected limits, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualWitnessConsumption`

Proof. Start with Definition 765.366. It keeps the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, checker rows, obstruction row, transport, replay, provenance, and local naming rows visible at the critical-path boundary.

Next read the frontier exports cited in that handoff. The public export of Definition 4238.47 and the scoped closure status of Definition 4238.48 identify the handoff as a finite frontier packet rather than a global confluence or consistency theorem.

The extra consumer precondition is the concrete residual witness row. Placing that row beside the critical-pair, candidate-join, bounded-checker, obstruction, structural, and L10 source-ledger rows forces a normal-form or confluence consumer to read a displayed witness packet before it can inspect the handoff.

Since all consumed data are displayed rows of the finite frontier and critical-path ledgers, the route has no coordinate for subject reduction, global Church–Rosser, evaluator equality, quotient term equality, global normalization, kernel consistency, selected limits, or ambient Real$^{\uparrow}$ completeness. □

Theorem 765.368 (MetaCIC candidate schedule mature package)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted MetaCICCriticalPath$^{\uparrow}$ *packet. A mature candidate schedule package is readable only after the candidate schedule discharge of Definition 765.290, the residual-budget discharge frontier of Definition 765.288, and the L10 source package of Definition 765.348 have all been exposed on the same packet. The package may route candidate-mediated strong normalization, residual substitution, local diamond data, bounded checker rows, and the four L10 source faces* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *but it exports no subject-reduction theorem, global Church–Rosser theorem, quotient term equality, kernel consistency claim, selected limit, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Begin with the candidate schedule discharge. It forces the candidate-mediated schedule to display candidate-set normalization, critical-pair ledger, residual-substitution boundary, finite observation confluence, and subject-reduction socket rows before the schedule is treated as discharged.

Next attach the residual-budget frontier. The residual route requires residual-budget rows, substitution-boundary rows, closed-premise ledger rows, and local-diamond rows on the same packet rows $S, N, O, U, D, H, C, P, L$ before the discharge socket can be read.

Finally attach the mature L10 source package. That route keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the four visible source faces while residual, confluence, and candidate-mediated strong-normalization rows remain budget data attached to the packet.

The three routes share the same carrier and structural rows. Their combined read is therefore a candidate schedule mature package, not a detached metatheorem. Since none of the component routes has a coordinate for subject reduction, global Church–Rosser, quotient term equality, kernel consistency, selected limits, or ambient completeness, the package exports none of those claims. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateScheduleMaturePackageL10SourceCe`

Theorem 765.369 (MetaCIC residual socket precedes strong-normalization reads)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any consumer reading candidate-mediated strong normalization from K must first expose the candidate schedule discharge, residual-budget socket, closed-substitution boundary, finite-observation confluence budget, subject-reduction socket, and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The resulting read keeps closed subject reduction, global Church–Rosser, beta-conversion decidability, quotient term equality, selected limits, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness outside the exported packet surface.*

Proof. Begin with the mature package theorem Definition 765.368. It places candidate-mediated strong normalization behind the candidate schedule discharge, residual-budget frontier, and L10 source package on the same rows $S, N, O, U, D, H, C, P, L$.

Next use the residual-socket consumer surface of Definition 765.370. That theorem already forces the residual-budget frontier, finite observation confluence boundary, closed-term substitution boundary, and subject-reduction socket to remain visible before the consumer reads candidate-mediated strong normalization.

Attach the L10 source routes. The mature package keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as displayed source faces, so the candidate-mediated read cannot hide the Real or completion-facing sources behind a metatheoretic result.

The cited routes exhaust the consumer surface together with the structural rows H, C, P, L. Since none of them supplies closed subject reduction, global Church–Rosser, beta-conversion decidability, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the residual socket is exposed before any strong-normalization read and those refused objects remain outside the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSocketBeforeSNRead`
Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateScheduleMaturePackage`

Theorem 765.370 (MetaCIC residual-socket consumer surface)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Every consumer of the mature candidate schedule must pass through the candidate schedule discharge of Definition 765.290, the residual-budget frontier of Definition 765.288, the finite observation confluence boundary of Definition 765.171, the closed-term substitution boundary of Definition 765.293, the subject-reduction socket of Definition 765.16,*

and the four L10 source faces in Definition 765.348 before reading candidate-mediated strong normalization. The resulting read exports no closed subject reduction, global Church–Rosser theorem, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet K. The mature package theorem Definition 765.368 already requires the candidate schedule discharge, residual-budget frontier, and mature L10 source package on the same rows $S, N, O, U, D, H, C, P, L$ before the schedule can be read.

Apply the candidate schedule discharge. It exposes candidate-set normalization, critical-pair ledgers, residual-substitution boundaries, finite observation confluence, and subject-reduction socket rows as named schedule data, so a consumer cannot read candidate-mediated strong normalization from S alone.

Next apply the residual-budget frontier. It requires residual-budget, substitution-boundary, closed-premise ledger, and local-diamond families to remain attached to the same packet before any socket-facing discharge read reaches D.

Now use the closed-substitution boundary exposure, confluence-decidability handoff exhaustion, and subject-reduction socket budget. These routes keep closed-term substitution, finite observation confluence, and the subject-reduction socket visible as rows rather than hidden metatheorems.

Finally attach the mature L10 source package. It keeps $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the four source faces through which the candidate-mediated read is routed.

The row inventory is exhausted by these routes and the structural rows H, C, P, L. Since no cited route supplies closed subject reduction, global Church–Rosser, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the residual socket remains a socket while candidate-mediated strong normalization is read. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualSocketConsumerSurface`

Theorem 765.371 (MetaCIC residual-diamond visible route)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. Any residual-diamond candidate read through* K *must expose the candidate schedule discharge, residual-budget frontier, local diamond witness, and the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any normalization-facing read is consumed. The resulting route exports no global Church–Rosser theorem, closed subject reduction theorem, quotient term equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the mature package theorem Definition 765.368. It places every candidate-mediated normalization read behind the candidate schedule discharge, residual-budget frontier, and mature L10 source package on the same packet rows $S, N, O, U, D, H, C, P, L$.

Next apply the residual socket precedence theorem Definition 765.369. This keeps candidate-mediated strong normalization downstream of the residual-budget socket, closed-substitution boundary, finite-observation confluence budget, and subject-reduction socket.

Now apply the residual-socket consumer surface of Definition 765.370. The consumer surface forces the residual-budget frontier and the local-diamond data carried by the finite observation confluence boundary to remain visible before the candidate read reaches a normalization-facing consumer.

Finally attach the mature L10 source package. The same read must pass through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as source faces rather than hiding them behind a metatheoretic normalization theorem.

The three local package theorems exhaust the visible route together with the structural rows. Since none of the cited routes carries global Church–Rosser, closed subject reduction, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the residual-diamond candidate remains a visible socket-level route before normalization-facing consumption. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathResidualDiamondVisibleRoute`

Theorem 765.372 (MetaCIC L10 residual-diamond source exhaustion)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet that consumes the mature candidate schedule. Every residual-diamond or candidate-mediated normalization read through K must expose the four L10 source faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the residual-budget frontier and local-diamond rows, before any normalization-facing conclusion is read. The resulting read exports no closed subject reduction, global Church–Rosser theorem, quotient term equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10ResidualDiamondSourceExhaustion`

Proof. Begin with the candidate schedule mature package of Definition 765.368. It binds every candidate-mediated normalization read to the accepted packet rows $S, N, O, U, D, H, C, P, L$ after the candidate schedule discharge, residual-budget frontier, and mature L10 source package have been exposed.

Next apply the residual socket precedence theorem Definition 765.369. This keeps the residual-budget socket, closed-substitution boundary, finite-observation confluence budget, and subject-reduction socket visible before the strong-normalization-facing read reaches a consumer.

Use the residual-socket consumer surface of Definition 765.370. It forces the residual-budget frontier and the local-diamond data carried by the finite observation confluence boundary to remain on the same packet rows rather than being hidden behind the normalization conclusion.

Finally apply the residual-diamond visible route of Definition 765.371. That route already requires $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to remain exposed as source faces before a residual-diamond candidate read is consumed.

The cited routes exhaust the residual-diamond and candidate-mediated normalization surface available from K. Since none supplies closed subject reduction, global Church–Rosser, quotient term equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects remain outside the exported read. □

Theorem 765.373 (MetaCICCriticalPath mature closure witness)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. The mature packet is a paper-side package witness: it is exactly the candidate-mediated strong-normalization, residual-substitution, confluence and decidability, and four-object L10 source package exposed by the existing child routes. A mature consumer must read the candidate schedule, residual-budget discharge, parallel-diamond frontier, and* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *source rows before it treats K as a mature critical-path packet.*

The mature paper closure exports no subject-reduction theorem, global Church–Rosser theorem, quotient term equality, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or Lean formal verification of those external principles.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMatureClosureWitness`

Proof. Start with the mature L10 source package of Definition 765.348. It fixes the four source faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and keeps residual, confluence, decidability, and candidate-mediated strong-normalization data attached to the same packet rows.

Next apply the parallel-diamond frontier handoff of Definition 765.353. It forces any confluence-facing or normalization-facing discharge request through the displayed candidate frontier, residual substitution budget, local diamond witness, and L10 source ledger before the request reaches a public endpoint.

Finally apply the candidate schedule mature package of Definition 765.368. The schedule route combines the candidate schedule discharge, residual-budget frontier, and mature L10 source package on the same carrier K, with H, C, P, L preserving transport, replay, provenance, and naming.

These three routes exhaust the mature packet advertised by the hub marker as a paper-side package witness. None of the cited routes contains a row for subject reduction, global Church–Rosser, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean formal verification of those external principles, so the mature closure witness exports exactly the displayed package and none of the excluded principles. □

Theorem 765.374 (MetaCICCriticalPath mature consumer synthesis)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A mature consumer of* K *receives exactly the finite synthesis of the confluence frontier, normalization frontier, substitution boundary, typed-example row, and bounded checker routes already displayed by the critical-path child files. The consumer surface is therefore the same packet route through* $S, N, O, U, D, H, C, P, L$ *together with the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

This synthesis does not claim full normalization, full confluence, closed subject reduction, arbitrary Church–Rosser, global beta-conversion decidability, quotient term equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathMatureConsumerSynthesis`

Proof. Begin with the normalization and confluence dependency rows. Theorems 765.291 and 765.292 place the normalization frontier on S, N and the confluence audit route on U, while the obstruction row O and discharge socket D remain visible.

Next attach the substitution boundary. Theorem 765.293 keeps closed-substitution support as boundary data routed through U and D, not as a filled subject-reduction row.

Use Definition 765.16 and Definition 765.17. Candidate-mediated strong normalization and the L10 four-face exit remain downstream of the visible subject-reduction socket row D.

Now add the typed-example and bounded checker surfaces. Theorems 765.325, 765.235, and 765.195 require typed-example coverage, candidate-mediated strong-normalization evidence, bounded checker data, and the four L10 source faces to remain on the same displayed route.

Finally apply the mature routes. Theorems 765.353, 765.368, and 765.373 package the residual diamond frontier, candidate schedule, and mature L10 source rows as one consumer surface. None

of the cited routes contains a coordinate for the excluded global metatheoretic or real-completion principles, so the synthesis exports only the finite mature consumer route stated above. □

Theorem 765.375 (Candidate confluence decidable reflection)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *candidate schedule that exposes strong normalization rows, residual-local-diamond rows, and decidable audit rows factors any confluence-decidability consumer through the displayed critical-path route. The consumer first reads the candidate schedule, then the strong-normalization ledger, then the residual local-diamond ledger, and only then the decidable audit rows; it never appeals to an ambient typechecker oracle, global conversion decision procedure, unrestricted reduction search, quotient syntax equality, or host confluence theorem.*

Proof. Read the candidate schedule as finite BHist data. Its schedule rows name the reductions and residual windows that a later audit may inspect, so a decidability-facing consumer cannot start from an external typechecker answer.

The strong-normalization ledger is consumed next. It supplies the terminating branch budget for the displayed schedule, while the residual-local-diamond ledger supplies the local join rows for overlapping residual windows.

Finally the decidable audit rows package these two finite ledgers into the $\mathsf{MetaCICCriticalPath}^{\uparrow}$ route. Since each row is already part of the displayed candidate schedule, the consumer receives confluence decidability only as a certificate over those rows and not as an oracle for conversion, unrestricted reduction, quotient equality, or host confluence. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateConfluenceDecidableReflection`

Theorem 765.376 (MetaCIC candidate-discharge matrix)**.** *Let*

$$K = (S, N, O, U, D, H, C, P, L)$$

be an accepted $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. A discharge-socket read through* K *may be consumed only after the candidate-mediated strong-normalization rows, residual diamond budget rows, bounded checker rows, and the four L10 faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

are all displayed on the same packet route. The read therefore factors through the candidate-mediated SN frontier, the residual-substitution diamond budget, the visible residual route, and the L10 normalization discharge budget before any consumer reads D *as a discharge socket.*

Proof. Begin with the candidate-mediated SN frontier of Definition 765.198. It places the candidate rows and the four L10 faces on the accepted route before the confluence-facing or decidability-facing handoff can be consumed.

Attach the residual-substitution diamond budget of Definition 765.305. That route forces the residual budget, diamond checker, candidate-mediated frontier, and L10 source rows to remain on the same packet before bounded conversion reads U.

Now use the visible residual route of Definition 765.328 together with the L10 normalization discharge budget of Definition 765.273. These rows keep the bounded checker, obstruction row O, discharge socket D, transport row H, continuation row C, provenance row P, local naming row L, and the four L10 faces visible before the socket is read.

Since the displayed matrix contains no coordinate for closed subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness, the discharge read is exactly the finite candidate-residual-L10 socket surface. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeMatrix`

Theorem 765.377 (MetaCIC discharge matrix Lean target index)**.** *The candidate-discharge matrix export surface of this sibling consists exactly of the matrix theorem, the L10 socket-exhaustion theorem, the matrix-frontier theorem, and the retained-premise discharge-boundary theorem. These four paper sites jointly expose the candidate-mediated rows, residual budget, bounded checker rows, and four L10 faces before any discharge-socket read.*

Proof. Read Definition 765.376 for the candidate-mediated matrix and Definition 765.378 for the socket exhaustion boundary.

Then read Definition 765.379 for the bounded frontier and Definition 765.380 for retained-premise discharge. No fifth export row is introduced in this sibling, so the index is exactly the four displayed sites. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathDischargeMatrixLeanTargetIndex`

Theorem 765.378 (MetaCIC L10 candidate socket exhaustion)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, omitting any candidate-mediated row, residual-budget row, bounded-checker row, or any one of the L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *leaves the candidate-discharge socket unmet. In that case no consumer may read the missing surface as closed subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the matrix theorem above. It identifies the whole discharge-socket surface as the joint display of candidate-mediated rows, residual diamond budget rows, bounded checker rows, and the four L10 faces on one packet route.

If a candidate row is absent, Definition 765.198 supplies no frontier for the handoff. If a residual row or checker row is absent, Definition 765.305 and Definition 765.328 supply no visible residual route to the socket.

If an L10 face is absent, Definition 765.273 does not authorize a discharge read through the source budget. Thus each omission leaves the socket unmet and keeps all global metatheoretic, quotient, selected-limit, and Real-completeness readings outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathL10CandidateSocketExhaustion`

Theorem 765.379 (MetaCIC candidate-discharge matrix frontier)**.** *Let* $K = (S, N, O, U, D, H, C, P, L)$ *be an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet. If the candidate-mediated strong-normalization rows, residual-substitution diamond rows, bounded checker rows, and the four L10 faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are all displayed on the same packet route, then* K *supplies only a bounded formal-target frontier for the discharge socket* D*. The frontier authorizes the visible candidate-residual-L10 handoff and does not close subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the matrix theorem Definition 765.376. It identifies a discharge-socket read with the joint display of candidate-mediated rows, residual diamond budget rows, bounded checker rows, and the four L10 faces on the same accepted route.

Next apply the socket-exhaustion theorem Definition 765.378. It shows that any missing candidate row, residual row, checker row, or L10 face leaves the socket unmet, so the displayed matrix is exactly the finite boundary that a consumer may use.

The carrier K contains no row that asserts closed subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. Hence the accepted route supplies a bounded formal-target frontier for D and no global metatheoretic theorem beyond that frontier. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathCandidateDischargeMatrixFrontier`

Theorem 765.380 (MetaCIC retained-premise discharge boundary)**.** *For an accepted* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packet* $K = (S, N, O, U, D, H, C, P, L)$*, retained premises may be consumed by the discharge socket* D *only as displayed candidate rows, residual-budget rows, bounded-checker rows, and L10 faces. The retained-premise read therefore remains inside the candidate-discharge matrix and does not become closed subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the candidate-discharge matrix of Definition 765.376. It identifies the socket-facing route as the joint display of candidate-mediated rows, residual diamond budget rows, bounded checker rows, and the four L10 faces.

Next use socket exhaustion from Definition 765.378. Any missing candidate row, residual row, checker row, or L10 face leaves the discharge socket unmet, so retained premises can be read only through the displayed matrix coordinates.

Finally apply the matrix frontier theorem Definition 765.379. Its frontier clause prevents the accepted route from exporting closed subject reduction, full consistency, global confluence, beta-conversion decidability, quotient term equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathRetainedPremiseDischargeBoundary`

Chapter 766

A Concrete Naming Certificate for $\mathsf{RealAlgOrder}^{\uparrow}$

The $\mathsf{RealAlgOrder}^{\uparrow}$ interface packages the algebra and order rows of displayed constructive real names. It sits above the real-name seal of chapter 38, the regular rational source rows of chapter 301, the dyadic operation ledgers of chapter 291, the finite windows of chapter 126, and the positive-separation rows of chapter 304. These sibling chapters supply the endpoint, readback, operation, window, and apartness data consumed by the local packet.

The certificate is the constructive algebra/order surface for downstream metric, measure, and probability consumers. It records finite addition, negation, multiplication, absolute-value, nonnegative-order, and strict-apartness evidence as named rows; it does not import trichotomy of reals, quotient equality of Cauchy presentations, a host real carrier, or ambient completeness.

Definition 766.1 (RealAlgOrder carrier)**.** *A* $\mathsf{RealAlgOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, G, D, S, A, O, H, C, P, N)$$

with the following displayed coordinates:

1. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *endpoint rows consumed by the algebra or order request;*
2. *G is the family of* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those endpoints before the real-seal handoff;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation ledger recording the finite dyadic reads used for addition, negation, multiplication, absolute value, and nonnegative comparison;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the endpoint readbacks are inspected;*
5. *A is the optional* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row used when the request is strict apartness rather than nonnegative comparison;*
6. *O is the local algebra/order ledger whose rows are the displayed sum, product, negation, absolute-value, nonnegative-order, and strict-apartness conclusions licensed by E, G, D, S, A;*
7. *H stores the componentwise* hsame *transports for endpoint rows, readbacks, dyadic ledgers, windows, apartness rows, and the local order ledger;*
8. *C stores the* Cont *routes exposing finite readback and operation rows to downstream consumers;*
9. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealAlgOrder}^\uparrow}$ *row.*

The carrier is a finite certificate for algebra and constructive order on displayed real names. It is not a host ordered field, not a quotient of real-name presentations, and not a completeness object.

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderTasteGate single carrier alignment`

Theorem 766.2 (RealAlgOrder NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealAlgOrder}^\uparrow}$ *surface for the carrier of Definition 766.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet $R = (E, G, D, S, A, O, H, C, P, N)$ with endpoint, readback, dyadic-operation, finite-window, apartness, ledger, transport, continuation, provenance, and local naming coordinates;*
2. *classifier transport: classification transports every endpoint row in E, every* $\mathsf{RegSeqRat}^\uparrow$ *readback in G, every dyadic operation row in D, every finite window in S, every apartness row in A, and the algebra/order ledger O componentwise through the stored* hsame *and* Cont *routes;*
3. *algebra stability: addition, negation, multiplication, and absolute-value rows in O remain accepted after replacing their inputs by classified* $\mathsf{Real}^\uparrow$ *endpoint rows and transporting the corresponding* $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, *and* $\mathsf{StreamName}^\uparrow$ *witnesses;*
4. *order stability: nonnegative-order and strict-apartness rows in O are read only from the displayed finite comparison and separation ledgers, and they transport across the same endpoint and window classifiers;*
5. *ledger exactness: every downstream algebra or order read is one of the displayed rows in $E, G, D, S, A, O, H, C, P, N$ before any consumer receives it.*

Proof. Carrier admission is projection from the tuple in Definition 766.1. The endpoint rows, regular-rational readbacks, dyadic operation ledger, finite-window schedule, apartness row, local ledger, transport table, continuation table, package provenance, and local naming row are all explicit coordinates.

Classifier transport is componentwise. The $\mathsf{Real}^\uparrow$ endpoints transport together with their $\mathsf{RegSeqRat}^\uparrow$ source readbacks and $\mathsf{StreamName}^\uparrow$ windows, while the $\mathsf{DyadicRatCore}^\uparrow$ operation rows and $\mathsf{ApartnessReal}^\uparrow$ separation rows transport through their own sibling classifiers.

The algebra and order stability rows then repack the transported coordinates into the same ledger shape O. Addition, negation, multiplication, and absolute value use only the dyadic operation reads exposed by D; nonnegative order and strict apartness use only the finite comparison and separation rows exposed by D and A.

Ledger exactness follows because every consumer-facing row is a coordinate of R or a row of O generated from those coordinates. Since no coordinate stores trichotomy, quotient real equality, a host real carrier, or ambient completeness, none of those principles can be projected from the local $\mathsf{NameCert}_{\mathsf{RealAlgOrder}^\uparrow}$ surface. □

Theorem 766.3 (RealAlgOrder classifier transport companion)**.** *For constant rational histories d, e, d', e', a rational carrier row for d, a rational classifier row $d \to e$, and transports $d\mathsf{hsame}d'$ and $e\mathsf{hsame}e'$ determine the transported constant-real carrier rows for $BHist.e1(d')$ and $BHist.e1(e')$, the transported constant-real classifier row $BHist.e1(d') \to BHist.e1(e')$, and the transported rational classifier row $d' \to e'$.*

Proof. Unpack the classifier-transport row of Definition 766.2. The stored hsame transports move the rational classifier $d \to e$ to the displayed row $d' \to e'$, and the rational carrier for d' is transported along the same route.

Now read the constant-real interface through $BHist.e1$. The RatHistoryCarrier and RatHistoryClassifier rows are exactly the inputs to RealConstantHistoryCarrier and RealConstantHistoryClassifier on the endpoint histories, so the transported rational classifier supplies the two carrier rows and the real classifier row without adding trichotomy, quotient equality, or completeness. □

Theorem 766.4 (RealAlgOrder algebra-row companion)**.** *For constant rational histories d, e, o, d', e', o', rational classifier rows $d \to d'$, $e \to e'$, and $o \to o'$ determine the constant-real carrier rows for $BHist.e1(d)$, $BHist.e1(e)$, and $BHist.e1(o)$, together with the three constant-real classifier rows $BHist.e1(d) \to BHist.e1(d')$, $BHist.e1(e) \to BHist.e1(e')$, and $BHist.e1(o) \to BHist.e1(o')$.*

Proof. Project the algebra stability row of Definition 766.2. The endpoint, endpoint-transport, and operation histories are all rational classifier rows, so each source history carries the rational carrier evidence needed by the local algebra ledger.

Passing through the constant-real classifier interface converts those three rational classifier rows into the three displayed RealConstantHistoryClassifier rows on $BHist.e1$. Their source carriers become the displayed RealConstantHistoryCarrier rows, and no further real-algebra structure is read. □

Theorem 766.5 (RealAlgOrder rational order-boundary companion)**.** *For constant rational histories d, e, o, d', e', o', rational classifier rows $d \to e$ and $e \to o$ and transports $d\mathsf{hsame}d'$, $e\mathsf{hsame}e'$, and $o\mathsf{hsame}o'$ determine the transported constant-real carrier rows for $BHist.e1(d')$, $BHist.e1(e')$, and $BHist.e1(o')$, the transported constant-real classifier rows $BHist.e1(d') \to BHist.e1(e')$ and $BHist.e1(e') \to BHist.e1(o')$, and the transitive rational boundary row $d' \to o'$.*

Proof. Use the order stability row of Definition 766.2. The two rational classifier rows are first transported componentwise along the stored hsame witnesses, giving $d' \to e'$ and $e' \to o'$ inside the displayed finite order surface.

The constant-real interface then reads these transported rational classifiers through $BHist.e1$, producing the three endpoint carrier rows and the two adjacent real classifier rows. Finally, transitivity of the transported rational classifier rows gives the boundary row $d' \to o'$, which is still a rational ledger row rather than a quotient-real or completeness witness. □

Theorem 766.6 (RealAlgOrder apartness companion)**.** *For constant rational histories d, e, d', e', rational classifier rows $d \to d'$ and $e \to e'$ determine the constant-real carrier rows for $BHist.e1(d)$, $BHist.e1(d')$, $BHist.e1(e)$, and $BHist.e1(e')$, together with the two constant-real classifier rows $BHist.e1(d) \to BHist.e1(d')$ and $BHist.e1(e) \to BHist.e1(e')$.*

Proof. Read the strict-apartness side of Definition 766.2. The apartness ledger consumes only the displayed endpoint classifier rows, so the two rational classifiers expose both their source and target rational carrier rows.

Applying the constant-real classifier interface to each classifier row yields the four displayed RealConstantHistoryCarrier rows and the two RealConstantHistoryClassifier rows. The packet therefore exposes only finite endpoint classifier evidence for apartness, with no branch for real trichotomy, quotient equality, or ambient completeness. □

Theorem 766.7 (RealAlgOrder apartness and order non-escape)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, *measure, or real-algebra consumer of an accepted* $\mathsf{RealAlgOrder}^{\uparrow}$ *packet receives only the displayed finite rows of Definition 766.1:* $\mathsf{Real}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation and nonnegative-comparison ledgers,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window*

separation rows when strict apartness is requested, and the visible hsame, Cont, Pkg, *and local* NameCert *provenance.*

In particular, an order or apartness consumer can read nonnegative-order evidence only as a displayed finite comparison row and strict-apartness evidence only as a displayed finite separation row. The packet exports no trichotomy branch, no quotient real equality, no host real carrier, and no ambient completeness witness.

Proof. Start with the carrier inventory in Definition 766.1. The only public coordinates are the endpoint family E, readback family G, dyadic ledger D, finite-window schedule S, apartness row A, algebra/order ledger O, and the provenance rows H, C, P, N.

Apply the NameCert obligations of Definition 766.2. Carrier admission and ledger exactness force every downstream read to pass through those coordinates, while classifier transport and the two stability rows preserve the same finite evidence after endpoint replacement.

For nonnegative order, the consumer reads the comparison rows displayed in D and repacked in O. For strict apartness, the consumer reads the finite-window separation row displayed in A and repacked in O through the $\mathsf{ApartnessReal}^{\uparrow}$ sibling surface of chapter 304.

Thus every metric, probability, measure, or algebra-facing consumer receives a finite order or separation row with visible hsame, Cont, Pkg, and NameCert provenance. None of the excluded host principles is a carrier coordinate or a ledger row, so none can escape through this handoff. □

Theorem 766.8 (RealAlgOrder constant-rational scoped certificate)**.** *For constant rational* $\mathsf{Real}^{\uparrow}$ *endpoint rows, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *packet of Definition 766.1 carries the scoped certificate obtained by transporting the endpoint and operation rows through the* $\mathsf{Rat}^{\uparrow}$ *classifiers. Concretely, if* d, e, o *are the displayed rational histories for the left endpoint, right endpoint, and operation row, and* d', e', o' *are their classified transports, then the packet records* RealConstantHistoryCarrier *for each of* $BHist.e1(d)$, $BHist.e1(d')$, $BHist.e1(e)$, $BHist.e1(e')$, $BHist.e1(o)$, *and* $BHist.e1(o')$*; it records* RealConstantHistoryClassifier *for the three displayed constant rational comparisons* $BHist.e1(d) \to BHist.e1(d')$, $BHist.e1(e) \to BHist.e1(e')$, *and* $BHist.e1(o) \to BHist.e1(o')$*; and its order boundary records the transitive* RatHistoryClassifier *row from* d *to* o.

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant rational scoped certificate`

Proof. Start with the carrier inventory of Definition 766.1. The endpoint family E supplies the two constant rational $\mathsf{Real}^{\uparrow}$ endpoint rows, the ledger O supplies the operation row, and the sibling $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows named in the chapter introduction expose the rational histories and finite reads used by those rows.

Apply the local obligations of Definition 766.2. Classifier transport moves the endpoint rows and the operation row componentwise through the stored hsame and Cont routes, while ledger exactness keeps the downstream read inside the displayed coordinates.

The sibling $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ constant classifier interface identifies a rational classifier row with the corresponding constant real classifier on the $BHist.e1$ endpoint. Applying that interface to the three displayed rational comparisons gives the six RealConstantHistoryCarrier rows and the three RealConstantHistoryClassifier rows.

Finally, the order boundary is the transitive rational classifier row. The two RatHistoryClassifier comparisons $d \to e$ and $e \to o$ compose to the displayed $d \to o$ row, so the scoped $\mathsf{RealAlgOrder}^{\uparrow}$ certificate contains exactly the constant-rational endpoint, operation, and boundary data described above, with no appeal to real trichotomy, quotient equality, or completeness. □

Theorem 766.9 (RealAlgOrder MetricSpace SOne public boundary)**.** *Every* $\mathsf{S}^{1\uparrow}$ *unit-equation consumer whose distance and apartness read passes through* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *factors through exactly three displayed public rows: the constant-rational scoped certificate of Definition 766.8, the real-distance public correspondence of Definition 57.315, and the* $\mathsf{S}^{1\uparrow}$ *component-and-equation consumer boundaries of Definition 57.317 and Definition 58.67.*

Consequently the route exports only finite $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison and apartness rows, the* $\mathsf{Met}^{\uparrow}$ *real-distance handoff, and the* $\mathsf{S}^{1\uparrow}$ *component/equation readback. It exports no trichotomy row, quotient-circle carrier, host real equality, angle parameter, selected limit, or metric-completeness row.*

Proof. Start with the constant-rational scoped certificate of Definition 766.8. It is the only $\mathsf{RealAlgOrder}^{\uparrow}$ source named by the route, so the rational endpoint, operation, and boundary histories available to the consumer are exactly the displayed constant-rational histories and their classified transports.

Next apply the real-distance public correspondence of Definition 57.315. That theorem is the $\mathsf{Met}^{\uparrow}$ boundary that repacks positive-distance and apartness information as public finite rows, and it does not add a host real equality, completeness witness, or trichotomy branch.

Then apply the $\mathsf{Met}^{\uparrow}$–$\mathsf{S}^{1\uparrow}$ factorization of Definition 57.317. It says that an $\mathsf{S}^{1\uparrow}$ consumer reading the metric surface through $\mathsf{RealAlgOrder}^{\uparrow}$ must pass through the real-distance correspondence before seeing the unit-equation surface.

Finally use the component-and-equation boundary of Definition 58.67. The $\mathsf{S}^{1\uparrow}$ side consumes only the component rows and the equation readback, so the composed route contains precisely the three public boundaries named in the statement and none of the excluded quotient, angle, limit, or completeness data. □

Theorem 766.10 (RealAlgOrder bridge-ready public interface)**.** *The bridge-ready public interface of* $\mathsf{RealAlgOrder}^{\uparrow}$ *is exactly the carrier* $R = (E, G, D, S, A, O, H, C, P, N)$ *of Definition 766.1, the local* NameCert *obligations of Definition 766.2, the apartness/order non-escape boundary of Definition 766.7, the constant-rational scoped certificate of Definition 766.8, and the* $\mathsf{Met}^{\uparrow}$–$\mathsf{S}^{1\uparrow}$ *public boundary of Definition 766.9.*

Consequently the standard bridge for $\mathsf{RealAlgOrder}^{\uparrow}$ *has a single paper interface to consume: it may read the displayed endpoint, readback, dyadic, schedule, apartness, order, transport, continuity, package, and naming rows of* R *together with precisely the four cited theorem surfaces. This theorem is the bridge-facing paper interface recorded by the closure block below.*

Lean4 ✓ `BEDC.Derived.RealAlgOrderUp.RealAlgOrderUp StdBridge`

Proof. Start with the carrier definition. Definition 766.1 names the ten public coordinates $E, G, D, S, A, O, H, C, P,$ so any standard-facing read begins from displayed endpoint rows, readback rows, dyadic ledgers, finite-window schedules, optional apartness rows, order ledgers, transport rows, continuity routes, package provenance, and the local naming certificate.

Apply the local obligation and non-escape theorems. Definition 766.2 supplies the carrier, classifier, stability, exactness, provenance, and naming rows that make those coordinates a NameCert surface, while Definition 766.7 rules out the escape of trichotomy, quotient real equality, host real carrier data, and ambient completeness through apartness or order reads.

Finally attach the two public boundary theorems. Definition 766.8 supplies the constant rational endpoint, operation, and transitive order rows; Definition 766.9 identifies the only $\mathsf{Met}^{\uparrow}$ and $\mathsf{S}^{1\uparrow}$ consumer rows visible through this route. These four theorem surfaces, together with the carrier coordinates, are therefore the exact paper interface available for the bridge theorem recorded below. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealAlgOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{RealAlgOrder}^{\uparrow}$ packet is generated from finite $\mathsf{Real}^{\uparrow}$ endpoint rows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ operation ledgers, $\mathsf{StreamName}^{\uparrow}$ windows, optional $\mathsf{ApartnessReal}^{\uparrow}$ separation rows, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealAlgOrder}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 766.1, Definition 766.2, Definition 766.7, and the constant-rational scoped certificate enumerate the carrier, NameCert obligation rows, apartness/order non-escape theorem, and constant rational $\mathsf{Real}^{\uparrow}$ rows covered by this obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealAlgOrder}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RealAlgOrderUp.RealAlgOrderConstant\_rational\_scoped\_certificate`

Bridge status: bridgeChecked

Not claimed: This chapter does not reconstruct a full real algebra, an ordered-field structure, trichotomy, quotient real equality, ambient completeness, or a public standard bridge.

Upgrade path: No further upgrade remains on the recorded standard-bridge verification axis for the finite $\mathsf{RealAlgOrder}^{\uparrow}$ packet.

Chapter 767

A Concrete Naming Certificate for $\mathsf{SocketTaxonomyLedger}^{\uparrow}$

The $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ packet realizes the seven T-supply socket kinds of Definition 9462.824 as a finite NameCert ledger. It sits beside chapter 528, chapter 1229, and chapter 542: the first consumes socket exposure, the second audits typed gap and failure rows, and the third records requested transcendental supply without supplying it. This packet binds the kind row, consumption-site row, audit-gate row, refusal row, non-collapse row, transport, continuation, provenance, and local name in one carrier.

Definition 767.1 (SocketTaxonomyLedger carrier)**.** *A* $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Sigma = (K, S, A, R, D, H, C, P, N)$$

with the following displayed rows:

1. *K is one of the seven socket-kind rows: Type I generator closure, Type II unwitnessed witness, Type III quotient or extensional collapse, Type IV causal continuity, Type V inter-history coordination, Type VI inscription novelty, or Type VII self-reference closure;*
2. *S is the consumption-site row at which the requested T-supply would be used;*
3. *A is the audit-gate row governing the request at that site;*
4. *R is the refusal row recording that the requested supply is visible only as a* Gap *boundary;*
5. *D is the non-collapse row separating the seven kinds by requested shortcut, consumer site, and gate;*
6. *H records componentwise* hsame *transport for the kind, site, gate, refusal, non-collapse, provenance, and naming rows;*
7. *C records displayed* Cont *routes for kind lookup, site consumption, gate inspection, refusal replay, and non-collapse comparison;*
8. *P is the* Pkg *provenance over* $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, $\mathsf{TranscendentalSupplyLedger}^{\uparrow}$, $\mathsf{RecursionAuthorizationLedger}^{\uparrow}$, $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SocketTaxonomyLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate that supplies unrestricted induction, a selected witness, quotient equality, a future law, another closed history, inscription-source actuality, or truth-total self-validation.

Lean4 ✓ `BEDC.Derived.SocketTaxonomyLedgerUp.SocketTaxonomyLedgerTasteGate single carrier alignment`

Theorem 767.2 (SocketTaxonomyLedger NameCert obligations)**.** *For the carrier of Definition 767.1, the local* $\mathsf{NameCert}_{\mathsf{SocketTaxonomyLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$\Sigma = (K, S, A, R, D, H, C, P, N),$$

kind classification through K*, consumption-site exposure through* S*, audit-gate routing through* A*, refusal through* R*, non-collapse through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 767.1. The rows $K, S, A, R, D, H, C, P, N$ account for the finite taxonomy, the consuming site, the gate, the refusal, the non-collapse comparison, transport, route replay, provenance, and local naming.

The sibling ledgers supply only their named surfaces. chapter 528 records socket exposure, chapter 1229 records typed gap and failure audit rows, and chapter 542 records the requested supply as a ledgered request. None of these siblings supplies the requested T-supply itself.

The rows H, C, P, N preserve and name the same finite packet. Since the packet exposes no further coordinate, every accepted NameCert read of $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ is exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.SocketTaxonomyLedgerUp.SocketTaxonomyLedgerNameCert obligations`

Theorem 767.3 (SocketTaxonomyLedger non-collapse)**.** *Every accepted* $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ *packet keeps the seven socket kinds distinct by the triple of kind row, consumption-site row, and audit-gate row. In particular, a Type III quotient-collapse socket cannot be consumed by a Type I generator-closure gate without changing the socket-kind row* K *and the non-collapse row* D*.*

Proof. Project the rows K, S, A, D from the carrier. The kind row records which of the seven socket kinds is present, while the consumption-site and audit-gate rows record where and how the request is refused.

Use the taxonomy non-collapse theorem Definition 9462.824. Type I asks for generator closure beyond the displayed generator, while Type III asks for relation-to-identity or quotient collapse. These requests have different shortcuts and different gate conditions.

Therefore a consumer that reads a Type III row through a Type I gate has changed the displayed kind and non-collapse data. The accepted packet permits transport of the same rows through H, C, P, N, but it does not permit reclassifying quotient collapse as generator closure. □

Theorem 767.4 (SocketTaxonomyLedger audit-gate non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ *packet factors through*

$$K, S, A, R, D, H, C, P, N.$$

No countable choice row, quotient stream, unrestricted recursor, forbidden host axiom, truth-total self-validator, or alternate hidden source can distinguish the packet once those rows agree.

Proof. Project the accepted packet. The carrier exposes only the kind, site, audit gate, refusal, non-collapse, transport, continuation, provenance, and naming rows.

Read the sibling provenance row P. The supply-socket, gap-failure audit, transcendental-supply, recursion-authorization, and forbidden-axiom-ancestry siblings are cited as finite rows with named obligations, not as hidden supplies.

It remains to inspect the excluded coordinates. Countable choice, quotient streams, unrestricted recursors, forbidden host axioms, truth-total validators, and alternate hidden sources are absent from the carrier. Thus any consumer that distinguishes two packets agreeing on $K, S, A, R, D, H, C, P, N$ has left the accepted NameCert route. □

Falsifiable prediction: A Type III quotient-collapse socket cannot be consumed by a Type I generator-closure gate without changing the socket-kind row.

Independence witness: supplysocketledger, gapfailurebridgeaudit, transcendentalsupplyledger, recursionauthorizationledger, forbidden-axiom-ancestry: none is bijective to $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ because this carrier binds the seven-kind taxonomy row, consumption-site row, audit-gate row, refusal row, and non-collapse row in one finite NameCert packet.

The Lean site `lean4/BEDC/Derived/SocketTaxonomyLedgerUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ SocketTaxonomyLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ SocketTaxonomyLedger}^{\uparrow}$ instances preserving $K, S, A, R, D, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{SocketTaxonomyLedger}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ packet is generated from one seven-kind taxonomy row, a consumption-site row, an audit-gate row, a refusal row, a non-collapse row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SocketTaxonomyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 767.1, Definition 767.2, Definition 767.3, and Definition 767.4 bind the finite carrier, seven-kind classifier, consumption-site row, audit-gate row, refusal row, non-collapse comparison, transport, continuation, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SocketTaxonomyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not discharge any T-supply request, provide unrestricted induction, select witnesses, collapse quotient or relation rows into equality, assert future-law continuity, identify external histories, explain inscription actuality, close self-truth validation, or provide bridge checking for $\mathsf{SocketTaxonomyLedger}^{\uparrow}$.

Upgrade path: Formal closure requires Lean encoding of the finite taxonomy ledger and checked theorems for NameCert obligations, non-collapse, audit-gate routing, and non-escape of the refused supply rows.

Chapter 768

A Concrete Naming Certificate for $\mathsf{Limsup}^{\uparrow}$

Following Hardy and Bishop–Bridges, $\mathsf{Limsup}^{\uparrow}$ records the finite tail-upper-cut and tail-lower-cut surface of a bounded real sequence. The packet consumes the bounded sequence rows of chapter 1415, the upper-cut observation discipline of **??**, the lower-cut observation discipline of chapter 1440, the displayed sequence-limit interface of chapter 1413, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It gives Real-and-Completion consumers a finite limsup boundary without importing completeness, a host supremum, or selected subsequences.

Definition 768.1 (Limsup carrier)**.** *A* $\mathsf{Limsup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, U, D, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row whose finite windows, bound ledger, and displayed* $\mathsf{Real}^{\uparrow}$ *entries are being inspected.*
2. *U is the tail upper-cut schedule, recording for each displayed tail index the bounded upper-cut observations read through* $\mathsf{BoundedRealUpperCut}^{\uparrow}$*.*
3. *D is the dyadic tolerance ledger, recording the finite rational tolerances used to compare tail upper cuts with the lower observation surface.*
4. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row named as the limsup candidate exposed by the packet.*
5. *H records componentwise* hsame *transports for S, U, D, T and the structural rows.*
6. *C records the displayed* Cont *routes by which consumers replay the bounded sequence row, tail upper-cut schedule, dyadic tolerance ledger, terminal real seal, transports, provenance, and local naming row.*
7. *P is the* Pkg *provenance over* $\mathsf{Limsup}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealUpperCut}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Limsup}^{\uparrow}}$ *row.*

The carrier stores a finite tail-cut surface for a bounded real sequence. It has no coordinate for Real completeness, trichotomy, a host supremum, quotient stream equality, selected subsequences, or open-cover compactness.

Lean4 (definition) `BEDC.Derived.LimsupUp.LimsupCarrier`

Definition 768.2 (Limsup classifier)**.** *Two* $\mathsf{Limsup}^{\uparrow}$ *carriers*

$$L = (S, U, D, T, H, C, P, N) \qquad \text{and} \qquad L' = (S', U', D', T', H', C', P', N')$$

are classified as the same displayed limsup packet exactly when the bounded sequence rows match componentwise, the tail upper-cut schedules expose the same finite tail observations, the dyadic tolerance ledgers carry the same rational comparison budgets, the terminal $\mathsf{Real}^{\uparrow}$ *seals are transported by the displayed* hsame *data, the continuation replays preserve the same consumer routes, and the provenance and local NameCert rows name the same* $\mathsf{Limsup}^{\uparrow}$ *surface. The classifier is therefore rowwise over* S, U, D, T, H, C, P, N *and does not identify carriers by host sequence equality, quotient streams, selected cofinal subsequences, or a completed supremum construction.*

Lean4 (definition) `BEDC.Derived.LimsupUp.LimsupClassifier`

Theorem 768.3 (Limsup NameCert obligations)**.** *For an accepted* $\mathsf{Limsup}^{\uparrow}$ *carrier* $L = (S, U, D, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Limsup}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier formation for the bounded real sequence row, tail upper-cut schedule, dyadic tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transports, continuations, provenance, and local naming data;*
2. *classifier stability for the componentwise comparison of Definition 768.2;*
3. *tail upper-cut exactness, stating that every upper observation in* U *is read from a finite tail of* S *through the bounded upper-cut surface;*
4. *lower-cut compatibility, stating that every tolerance row in* D *compares the same tail data against the bounded lower-cut observation surface;*
5. *terminal seal routing, stating that the candidate row* T *is consumed only as a* $\mathsf{Real}^{\uparrow}$ *seal after the displayed tail-cut checks have been replayed;*
6. *ledger non-escape, excluding Real completeness, trichotomy, host supremum selection, quotient stream equality, selected subsequences, and open-cover compactness.*

Proof. Carrier formation is projection from Definition 768.1. The packet displays exactly the bounded sequence row, tail upper-cut schedule, dyadic tolerance ledger, terminal real seal, componentwise transports, continuation routes, provenance, and local name.

Classifier stability is the componentwise comparison of Definition 768.2. Transport along H may replace the visible rows only by rows with the same scheduled tail observations, the same dyadic budgets, and the same terminal seal route, while C preserves the consumer replay order.

Tail upper-cut exactness is supplied by U together with the bounded sequence row S. Each accepted upper observation names the finite tail index, the inspected sequence windows, and the bounded upper-cut certificate through which the read is made.

Lower-cut compatibility is supplied by D and the sibling bounded lower-cut discipline. The tolerance ledger records only finite dyadic comparisons between the same tail data and the lower observation surface, so it does not become a host infimum or a completeness witness.

Terminal seal routing is the role of T. A consumer may read the candidate limsup only after the sequence tail, upper-cut, lower-cut, and tolerance rows have been replayed; the terminal coordinate remains a $\mathsf{Real}^{\uparrow}$ seal rather than a completed supremum object.

The non-escape row follows from the carrier inventory. No displayed field carries Real completeness, trichotomy, a host supremum, quotient stream equality, selected subsequences, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupNameCertObligations`

Theorem 768.4 (Limsup tail lower-cut compatibility)**.** *Let $L = (S, U, D, T, H, C, P, N)$ be an accepted* $\mathsf{Limsup}^{\uparrow}$ *carrier. Every dyadic tolerance row in D compares the same finite tail data exposed by U against the bounded lower-cut surface, so the limsup packet cannot read a host infimum, selected subsequence, or completed supremum when it certifies the lower side of the tail-cut boundary.*

Proof. Project S, U, D from the carrier of Definition 768.1. The bounded sequence row S supplies the only sequence entries, the upper schedule U names the finite tail observations, and the tolerance ledger D records the dyadic comparison budgets attached to those same observations.

Use the componentwise classifier of Definition 768.2. It compares accepted packets by the displayed bounded sequence row, tail upper-cut schedule, dyadic tolerance ledger, and terminal seal route, not by a host sequence object or by a completed extremal value.

The structural rows H, C, P, N preserve that lower-cut read. Transport may only replace the visible rows by classifier-equivalent rows, continuation replay rereads the same tail and tolerance cells, provenance names the bounded lower-cut dependency, and the local NameCert row records the finite packet. Thus the lower side of the limsup surface remains a bounded lower-cut comparison over displayed tail data. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailLowerCutCompatibility`

Theorem 768.5 (Limsup tail upper-cut exactness)**.** *Let $L = (S, U, D, T, H, C, P, N)$ be an accepted* $\mathsf{Limsup}^{\uparrow}$ *carrier. The row U is the only upper-tail observation route from the bounded sequence row S through finite* $\mathsf{BoundedRealUpperCut}^{\uparrow}$ *data. Thus no consumer can read a host supremum, selected cofinal subsequence, or detached upper bound when it consumes the upper side of the limsup packet.*

Proof. Project S and U from Definition 768.1. The bounded sequence row S supplies the finite sequence windows, while U records the tail upper-cut observations attached to those windows.

Use the classifier of Definition 768.2. Accepted packets agree on the displayed bounded sequence row and the tail upper-cut schedule, so an upper-tail read cannot be replaced by a host sequence object or by a completed supremum construction.

The structural rows preserve the same upper route. Transport through H, replay through C, provenance through P, and local naming through N carry the displayed $S \to U$ observation path and do not add another upper-cut coordinate. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailUpperCutExactness`

Theorem 768.6 (Limsup terminal seal routing)**.** *For every accepted* $\mathsf{Limsup}^{\uparrow}$ *carrier $L = (S, U, D, T, H, C, P, N)$, the terminal row T is consumed only after the bounded sequence row S, tail upper-cut schedule U, and dyadic comparison ledger D have been replayed. The row T remains a* $\mathsf{Real}^{\uparrow}$ *seal and is not a host supremum, selected subsequence limit, or ambient completeness witness.*

Proof. Begin with the carrier projection. The only value-facing coordinate is T, and the only rows that feed it are the bounded source row S, the upper-tail schedule U, and the dyadic lower-comparison ledger D.

Replay the route through C. A consumer reaches T only after the displayed tail data and comparison budgets have been read, so the terminal row is downstream of the same finite tail-cut surface.

Finally apply H, P, N. Componentwise transport, provenance, and local naming preserve the $\mathsf{Real}^{\uparrow}$ seal route without adding a supremum operator, selected subsequence, or completeness field. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTerminalSealRouting`

Theorem 768.7 (Limsup tail-seal scope)**.** *For every accepted* $\mathsf{Limsup}^{\uparrow}$ *carrier* $L = (S, U, D, T, H, C, P, N)$*, the candidate limsup value is exposed only after the same finite tail upper-cut row* U*, lower-cut tolerance ledger* D*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* T *have all been replayed. Consequently no consumer can replace the route by a host supremum, selected subsequence, quotient stream equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the upper-cut exactness theorem, Definition 768.5. It fixes U as the only upper-tail observation route from the bounded sequence row S.

Next apply the lower-cut compatibility theorem, Definition 768.4. It binds the dyadic ledger D to the same tail data and the bounded lower-cut surface, so the lower side cannot be replaced by a host lower envelope or by a selected subsequence.

Finally apply terminal seal routing, Definition 768.6. The terminal row T is reached only after S, U, D have been replayed, and the structural rows H, C, P, N preserve that order. The combined route therefore exposes a sealed $\mathsf{Real}^{\uparrow}$ candidate through the finite tail-cut packet and no excluded host-level principle. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailSealScope`

Theorem 768.8 (Limsup tail-window nonescape)**.** *Every public limsup comparison exported from an accepted* $\mathsf{Limsup}^{\uparrow}$ *tail window factors first through the bounded sequence row* S*, the bounded upper-cut and lower-cut rows exposed by* U *and* D*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the sibling* $\mathsf{RealSequenceLimit}^{\uparrow}$ *route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*. Thus the tail-window export cannot bypass the displayed BEDC rows by reading a host supremum, selected subsequence, quotient stream, trichotomy branch, or Real-completeness principle.*

Proof. Project the accepted carrier of Definition 768.1. The only sequence-facing data are the bounded row S, the upper-cut schedule U, and the lower-cut comparison ledger D.

Apply Definition 768.5 and Definition 768.4. These force the upper and lower sides of the tail window through $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealUpperCut}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget before any comparison is exposed.

Next use Definition 768.6 and Definition 768.7. The candidate value is reached only through the terminal $\mathsf{Real}^{\uparrow}$ seal, while the $\mathsf{RealSequenceLimit}^{\uparrow}$ row remains a routed sibling dependency rather than a replacement for the displayed tail-cut package.

Finally replay Definition 768.12. Since no carrier field stores the excluded host-level principles, the public comparison surface remains inside the scoped tail-window route. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailWindowNonescape`

Theorem 768.9 (Limsup scope kernel dependencies)**.** *Every accepted* $\mathsf{Limsup}^{\uparrow}$ *consumer must expose the bounded sequence row* S*, the bounded upper-cut and lower-cut rows read through* U *and* D*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *route, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*, componentwise* BHist *and* hsame *data, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Limsup}^{\uparrow}}$ *row before it may read the limsup surface.*

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupScopeKernelDependencies`

Proof. Start with Definition 768.5. It fixes U as the only upper-tail route from the bounded sequence row S through finite $\mathsf{BoundedRealUpperCut}^{\uparrow}$ data.

Next apply Definition 768.4. The lower side is tied to the same tail data through the dyadic tolerance ledger D, so the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison budget and the bounded lower-cut surface are visible before any limsup read is made.

Finally use Definition 768.6. The terminal $\mathsf{Real}^{\uparrow}$ row T is reached only after S, U, D have been replayed, and the structural rows H, C, P, N supply the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ scope. The $\mathsf{RealSequenceLimit}^{\uparrow}$ dependency is therefore a routed sibling row, not a replacement for the displayed kernel package. □

Theorem 768.10 (Limsup finite tail-window induction scope)**.** *Every accepted* $\mathsf{Limsup}^{\uparrow}$ *tail-window read is generated by finite induction over displayed* $\mathsf{StreamName}^{\uparrow}$ *windows. At each induction step the read exposes the bounded upper-cut row, bounded lower-cut row,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Limsup}^{\uparrow}}$ *row before any limsup endpoint is consumed.*

Proof. Start with the finite tail-cut route of Definition 768.11. The base window displays the bounded sequence row S, the initial upper-cut schedule U, the lower-cut compatibility ledger D, and the terminal seal T before the consumer can inspect an endpoint.

For the induction step, extend the displayed $\mathsf{StreamName}^{\uparrow}$ window by one finite read. The route keeps the bounded upper-cut and lower-cut rows synchronized with the same $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, so the new tail window is still generated from displayed finite rows rather than from a host subsequence or supremum.

Apply Definition 768.12 to the resulting window. It keeps the candidate value behind the terminal $\mathsf{Real}^{\uparrow}$ seal and prevents the induction step from importing Real completeness, trichotomy, quotient stream equality, or a selected cofinal subsequence.

Finally use Definition 768.9. The structural rows H, C, P, N provide the componentwise transport, replay, provenance, and local naming needed at every finite stage. Hence every accepted tail-window read is obtained by the stated finite induction before any limsup endpoint is consumed. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailWindowInductionScope`

Theorem 768.11 (Limsup tail-cut route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{Limsup}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto U \longmapsto D \longmapsto T \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before using the displayed limsup surface.

Proof. Begin with the bounded sequence row S. It supplies the only displayed source of sequence entries, finite windows, and boundedness evidence available to the packet.

The tail upper-cut schedule U then names which tails and upper observations are inspected. The dyadic ledger D records the tolerance comparisons that align those tail-upper observations with the bounded lower-cut surface.

Only after those rows are replayed can a consumer inspect the terminal real seal T. The structural rows H, C, P, N then transport, replay, source, and name the same finite route, so the consumer cannot bypass the tail-cut surface by reading a host supremum or a selected subsequence. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupTailCutRoute`

Theorem 768.12 (Limsup Real seal non-escape)**.** *An accepted* $\mathsf{Limsup}^{\uparrow}$ *packet exposes its candidate value only as the terminal* $\mathsf{Real}^{\uparrow}$ *row* T *reached through the displayed tail-cut route. In particular, it does not export Real completeness, trichotomy, a host supremum, quotient stream equality, selected subsequences, or open-cover compactness.*

Proof. The only value-facing coordinate in the carrier is T. All sequence-facing information is confined to S, U, D, and all structural access is confined to H, C, P, N.

A consumer that asks for the candidate value must therefore follow the continuation replay from the bounded sequence row through the tail upper-cut and tolerance ledgers before reaching T. The replay exposes a sealed real name, not a construction by supremum, compactness, or subsequence selection.

Since no carrier field stores the excluded principles, componentwise transport and continuation replay cannot introduce them. The $\mathsf{Real}^{\uparrow}$ seal remains terminal for the packet, and the limsup surface remains a finite tail-cut certificate. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupRealSealNonescape`

Theorem 768.13 (Limsup scoped tail-cut exhaustion)**.** *Every accepted* $\mathsf{Limsup}^{\uparrow}$ *consumer factors through the bounded sequence row* S*, the tail upper-cut schedule* U*, the bounded lower-cut compatibility row recorded by the dyadic tolerance ledger* D*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Limsup}^{\uparrow}}$ *row. Thus the scoped tail-cut surface is exhausted by the displayed finite route and exports no host supremum, selected subsequence, quotient stream, trichotomy branch, or Real-completeness principle.*

Proof. First apply Definition 768.5. It forces every upper-side read to begin with the bounded sequence row S and the finite tail upper-cut schedule U.

Next apply Definition 768.4. The lower-side comparison is tied to the same tail data by the dyadic tolerance ledger D, so the bounded lower-cut row and the $\mathsf{DyadicRatCore}^{\uparrow}$ budget are visible before any limsup consumer proceeds.

Now use Definition 768.6 and Definition 768.11. These route the consumer through S, U, D, T, H, C, P, N, placing the terminal $\mathsf{Real}^{\uparrow}$ seal after the bounded sequence, tail-cut, tolerance, transport, replay, provenance, and local naming rows.

Finally apply Definition 768.12. Since the terminal read is only the displayed Real seal and no carrier field stores the excluded host-level principles, the scoped tail-cut packet cannot export a host supremum, selected subsequence, quotient stream, trichotomy branch, or Real-completeness witness. □

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupScopedTailCutExhaustion`

Theorem 768.14 (Limsup scoped tail-cut witness)**.** *Every accepted* $\mathsf{Limsup}^{\uparrow}$ *consumer exposes the rows* S, U, D, T, H, C, P, N *as a single scoped witness surface. The surface binds upper-tail exactness through Definition 768.5, lower-cut compatibility through Definition 768.4, terminal seal routing through Definition 768.6, tail-cut route ordering through Definition 768.11, and Real seal nonescape through Definition 768.12.*

Lean4 ✓ `BEDC.Derived.LimsupUp.LimsupScopedTailCutWitness`

Proof. Project the accepted consumer to its displayed carrier rows. The bounded source row S, upper-tail schedule U, and dyadic lower-cut row D are fixed by the upper-cut exactness and lower-cut compatibility theorems.

Next compose the terminal seal and route-ordering results. They place T after the tail-cut rows and keep H, C, P, N as the transport, replay, provenance, and local naming surface for that same route.

Finally apply the nonescape theorem. Since the terminal Real read is reached only through the displayed rows, the eight-row surface is a scoped witness rather than an ambient supremum, selected subsequence, quotient stream, trichotomy branch, or Real-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{Limsup}\uparrow}$**.**

Constructive story: A $\mathsf{Limsup}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a finite tail upper-cut schedule, a dyadic tolerance ledger, a terminal RealUp seal, componentwise hsame transports, Cont replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Limsup}\uparrow}$, scopedClosed)

Scope closed: Definition 768.1, Definition 768.2, Definition 768.3, Definition 768.9, and Definition 768.10 bind the bounded sequence row, upper-cut and lower-cut rows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RealSequenceLimit}^{\uparrow}$ route, terminal $\mathsf{Real}^{\uparrow}$ seal, and BHist/hsame/Cont/Pkg/NameCert rows for the finite tail-window limsup surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Limsup}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, a host supremum, quotient stream equality, selected subsequences, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, tail upper-cut exactness, lower-cut compatibility, terminal seal routing, tail-cut route ordering, and Real seal non-escape for the displayed packet.

Chapter 769

A Concrete Naming Certificate for $\mathsf{UniformContinuousExtension}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformContinuousExtension}^{\uparrow}$ packet records the finite BEDC surface for extending a uniformly continuous map across displayed metric completions. It reads the completion carrier of chapter 95, the metric-completion route of chapter 1506, the complete-target convergence interface of chapter 128, the continuous-map graph and local-modulus rows of chapter 130, and the separated-target uniqueness boundary of chapter 1411. The packet is an extension ledger over finite modulus and Cauchy evidence rows; it is not a quotient-completion construction and it does not select representatives.

Definition 769.1 (UniformContinuousExtension carrier)**.** *A* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *carrier is a finite packet*

$$(X, Y, f, \omega, C_X, C_Y, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row together with the* $\mathsf{Completion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *evidence used to read completion names over the same point histories.*
2. *Y is the target* $\mathsf{CompleteMetric}^{\uparrow}$ *row equipped with the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness boundary for carried limits.*
3. *f is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row on the dense source, and ω is the finite uniform-modulus row that transports Cauchy closeness in X to Cauchy closeness in Y.*
4. *C_X is the source completion evidence: regular-Cauchy input windows, source observations, and the route from source names to completion names.*
5. *C_Y is the target convergence handoff: the complete-metric limit witness used after applying f through the transported modulus.*
6. *E is the extension row assigning a target completion read to each admitted source-completion read.*
7. *H is the componentwise* hsame *transport ledger for the source metric, target complete metric, map graph, modulus, completion evidence, convergence handoff, and extension row.*
8. *C is the displayed* Cont *replay route that forces every extension read to pass through $X, f, \omega, C_X, C_Y, E$ in that order.*

9. *P is the* Pkg *provenance row over* BHist, hsame, Cont, Met$^{\uparrow}$, Completion$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CompleteMetric$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, *and the local completion and convergence rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuousExtension}^{\uparrow}}$ *row naming this packet and no additional completion principle.*

Theorem 769.2 (UniformContinuousExtension NameCert obligations)**.** *For the carrier of Definition 769.1, the local* $\mathsf{NameCert}_{\mathsf{UniformContinuousExtension}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for the displayed packet* $(X, Y, f, \omega, C_X, C_Y, E, H, C, P, N)$*;*
2. *modulus transport, sending the source Cauchy evidence in* C_X *through* ω *and the graph row* f*;*
3. *Cauchy-source compatibility, requiring every extension read to use the same source completion route recorded in* X *and* C_X*;*
4. *complete-target convergence, requiring the transported image sequence to reach the target limit only through* Y *and* C_Y*;*
5. *separated uniqueness, comparing two candidate extension reads only by the* SeparatedMetric$^{\uparrow}$ *boundary on the same complete target;*
6. *ledger non-escape, stating that* H, C, P, N *expose no quotient completion, selected representative, countable-choice schedule, open-cover compactness, or completeness of* Real$^{\uparrow}$*.*

Proof. Project the displayed packet. Carrier admission is the statement that all rows listed in Definition 769.1 are present over the same source and target histories, with P and N naming the local package.

The modulus and source-compatibility clauses are read next. The row ω is the only route from source Cauchy evidence to target Cauchy evidence, and C_X is the only route from source observations to completion names, so every admitted extension read must consume both rows before reaching the target side.

The target convergence and uniqueness clauses then follow from the target rows. The complete-metric handoff C_Y supplies the carried target limit, while the separated boundary in Y is the only comparison row available for two displayed target limits.

Finally inspect the structural rows. Transport in H is componentwise, replay in C follows the same ordered route, and provenance in P, N records only the finite BEDC rows already listed. No excluded principle is a carrier coordinate, so the local NameCert obligations are exhausted by the six clauses above. □

Theorem 769.3 (UniformContinuousExtension completion handoff)**.** *Every extension read exported by an accepted* UniformContinuousExtension$^{\uparrow}$ *carrier factors through the displayed route*

$$X, C_X \longmapsto f, \omega \longmapsto Y, C_Y \longmapsto E.$$

Thus a completion consumer can read the extended value only after the source completion evidence, uniform modulus, continuous-map graph row, complete-target convergence handoff, and separated uniqueness boundary have been replayed inside the same packet.

Proof. Begin with the source side. The completion name is admitted through X and C_X, so the packet gives no direct access to a source-completion value before the displayed regular-Cauchy evidence has been read.

Apply the map route next. The graph row f and the modulus row ω transport the source Cauchy evidence into target Cauchy evidence. This is the only target-facing route in the carrier.

The complete target row then supplies the limit handoff. The row C_Y records convergence in the complete metric target, and the separated part of Y records uniqueness for the displayed target read.

The extension row E is therefore reached only after the route $X, C_X, f, \omega, Y, C_Y$ has been replayed. The structural rows H, C, P, N preserve that route under componentwise transport and expose no quotient completion, selected representative, countable-choice schedule, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuousExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformContinuousExtension}^{\uparrow}$ packet is generated from a source MetricUp row with CompletionUp and MetricCompletionUp evidence, a target CompleteMetricUp row with SeparatedMetricUp uniqueness, a ContinuousMapUp graph, a finite uniform-modulus row, source completion evidence, target convergence handoff, an extension row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformContinuousExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 769.1, Definition 769.2, and Definition 769.3 seed the finite uniformly continuous extension route across metric completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformContinuousExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, open-cover compactness, ambient target completeness beyond the displayed CompleteMetricUp row, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, modulus transport, source-completion compatibility, complete-target convergence, separated uniqueness, extension-row exactness, and ledger non-escape for BEDC.Derived.UniformContinuo

Chapter 770

A Concrete Naming Certificate for $\mathsf{JordanContent}^{\uparrow}$

Following Jordan and Bishop–Bridges constructive analysis, the $\mathsf{JordanContent}^{\uparrow}$ packet records finite Jordan content data through bounded rational boxes, dyadic partitions, lower and upper content sums, and common-refinement ledgers. It consumes the Darboux finite-sum surface of chapter 1880, the Darboux integration carrier of chapter 1460, the dyadic partition ledger of chapter 1863, the rational box endpoint rows of chapter 354, and the terminal real-name seal of chapter 38. The chapter names only the finite content ledger; it does not claim countable additivity, measure completion, selected limiting partitions, quotient equality of sets, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 770.1 (Jordan content finite carrier)**.** *A* $\mathsf{JordanContent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, P, L, U, R, S, H, C, Q, N)$$

with the following displayed rows:

1. *B is the bounded rational-box row, built from finitely many* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint and width ledgers.*
2. *P is the* $\mathsf{DyadicPartition}^{\uparrow}$ *row cutting the bounded box into finitely many named cells, together with the common-refinement data used when two ledgers are compared.*
3. *L and U are lower and upper content-sum rows, read through the* $\mathsf{DarbouxSum}^{\uparrow}$ *surface over the displayed partition row P.*
4. *R is the refinement row: it records the common dyadic refinement carrying B, P, L, U to any comparison partition without selecting a limiting partition family.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff row for a displayed content value window, obtained only after the lower and upper sums meet the recorded dyadic gap budget.*
6. *H records componentwise* hsame *transport for rational boxes, dyadic partitions, lower and upper content sums, common refinement, the real-facing handoff, continuation replay, provenance, and the local naming row.*
7. *C records displayed* Cont *replay from rational boxes to dyadic partition cells, from cells to the lower and upper content sums, from those sums to common refinement, and from the gap ledger to the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff.*

8. *Q is the* Pkg *provenance over* $\mathsf{JordanContent}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanContent}^{\uparrow}}$ *row naming this finite content packet.*

The classifier compares accepted packets only through $B, P, L, U, R, S, H, C, Q, N$. It has no coordinate for a sigma algebra, countable additivity, measure completion, selected partition limits, quotient equality of sets, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.JordanContentUp.JordanContentTasteGate single carrier alignment`

Theorem 770.2 (Jordan content NameCert obligations)**.** *For the carrier of Definition 770.1, the local* $\mathsf{NameCert}_{\mathsf{JordanContent}^{\uparrow}}$ *surface consists exactly of the following six finite obligations:*

1. *carrier admission for the packet $J = (B, P, L, U, R, S, H, C, Q, N)$;*
2. *rational-box admission, requiring every bounded box read to factor through the finitely displayed* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint and width rows in B;*
3. *dyadic partition compatibility, requiring every content cell to be named by P and every comparison to pass through the common-refinement row R;*
4. *lower and upper content-sum exactness, requiring L and U to be computed from the displayed cells and their carried lower and upper occupancy rows;*
5. *real-facing handoff, requiring S to be downstream of the finite lower/upper gap ledger before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can read it;*
6. *ledger non-escape, stating that H, C, Q, N preserve and name only these finite rows and exclude countable additivity, measure completion, selected limiting partitions, quotient set equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the displayed carrier. Carrier admission is exactly the finite packet of Definition 770.1, with rational boxes, a dyadic partition, two content-sum rows, a common-refinement row, a real-facing handoff, transport, replay, provenance, and a local name.

The rational-box and partition clauses are read before any content sum. The row B supplies the bounded rational endpoints, while P supplies the finite dyadic cells cut from those endpoints. Refinement data are stored in R, so comparing two packets uses a displayed common refinement rather than a selected limiting partition family.

The content-sum clauses follow from the Darboux-sum surface. The lower row L and upper row U are finite sums over the cells named by P, and common refinement transports those sums through R before the gap can be inspected.

Finally inspect the handoff and structural rows. The row S is downstream of the finite gap ledger, while H, C, Q, N only transport, replay, source, and name the already displayed rows. No excluded measure-theoretic or quotient object is a carrier coordinate, so the six obligations exhaust the local NameCert surface. □

Theorem 770.3 (Jordan content finite additivity scope)**.** *For accepted* $\mathsf{JordanContent}^{\uparrow}$ *carriers whose bounded rational boxes have a displayed disjoint dyadic decomposition inside a common refinement, finite additivity is available only as an equality of the transported lower and upper content-sum rows over that refinement. The statement exports a finite box-and-partition ledger to Darboux and Riemann consumers, not a measure-completion object and not countable additivity.*

Proof. Start with the two accepted carriers and read their rational-box rows. The displayed disjoint decomposition supplies a common dyadic refinement, so all cells used in the comparison are named by one finite partition row.

Transport both content packets through the refinement row R. Componentwise hsame transport preserves the rational-box coordinates, dyadic cell names, and lower and upper content-sum rows, while Cont replay recomputes the sums over the transported cells.

Finite additivity is then a ledger statement about finite cell lists. Disjoint cells contribute once, shared boundary rows are already part of the rational-box endpoint ledger, and the lower and upper transported sums combine by the same finite Darboux-sum route used to define L and U.

The exported scope stops at the transported finite sums and the real-facing handoff row S. Since no carrier coordinate stores a sigma algebra, measure completion, selected partition limit, quotient set equality, or countable family, the additivity claim cannot be read as a countable or completed measure theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{JordanContent}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanContent}^{\uparrow}$ packet is generated from bounded rational boxes, a dyadic partition row, lower and upper content-sum rows, a common-refinement ledger, a RegSeqRat/Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanContent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 770.1, Definition 770.2, and Definition 770.3 seed the finite Jordan content ledger over bounded rational boxes and dyadic partitions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanContent}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable additivity, measure completion, selected limiting partitions, quotient equality of sets, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-box admission, dyadic partition compatibility, lower and upper content-sum exactness, common-refinement transport, real-facing handoff, finite additivity scope, and ledger non-escape for BEDC.Derived.JordanContentUp.

Chapter 771

A Concrete Naming Certificate for $\mathsf{UniformConvergenceTest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformConvergenceTest}^{\uparrow}$ packet records a uniform convergence test as finite shared-tail data over the $\mathsf{RealSeries}^{\uparrow}$ and $\mathsf{UniformCauchyCriterion}^{\uparrow}$ rows. It sits between the finite partial-sum surface of chapter 1581, the shared-threshold family criterion of chapter 435, the optional majorant route of chapter 399, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed shared-tail route; it is not pointwise-only convergence, a selected infinite sum, a quotient function space, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 771.1 (Uniform convergence test carrier)**.** *A* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, S, N, T, M, R, E, H, C, P, Q)$$

with the following visible rows:

1. *F is the source family row naming the inspected functions, terms, or partial-sum windows only through the finite indices carried by the packet;*
2. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *or partial-sum series row whose displayed finite sums are the objects being tested;*
3. *N is the shared tail threshold row, one finite threshold surface read by all source-family windows under the requested dyadic tolerance;*
4. *T is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row that records the uniform family comparison after F, S, N have been fixed;*
5. *M is an optional* $\mathsf{WeierstrassMTest}^{\uparrow}$ *majorant row, absent when the test is supplied directly by T and present only when a displayed majorant ledger produces the same shared threshold;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the uniform tail data before any real-facing seal is exposed;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed by downstream Real or Completion routes after R has been displayed;*
8. *H stores componentwise* hsame *transports for the source family, series row, shared threshold, uniform Cauchy row, optional majorant row, regular readback, real seal, continuation, provenance, and local naming rows;*

9. *C stores displayed* Cont *routes from source-family reads to series windows, from series windows to the shared threshold, from the threshold to the uniform Cauchy row, from the optional majorant to the same threshold, from uniform Cauchy data to regular readback, and from readback to the real seal;*

10. *P is the* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, N, T, M, R, E, H, C, P, Q$. It has no coordinate for pointwise-only convergence, selected infinite sums, quotient equality of function names, countable-choice schedules, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformConvergenceTestUp.UniformConvergenceTestTasteGate single carrier alignment`

Theorem 771.2 (Uniform convergence test NameCert obligations)**.** *For every accepted* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *carrier $U = (F, S, N, T, M, R, E, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, source-family access through F, series-window access through S, shared tail thresholding through N, uniform Cauchy comparison through T, optional majorant transport through M,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through Q.*

The optional majorant row is not a second convergence proof. When present, it must route through the same shared threshold row N and the same $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row T before R or E can be read.*

Proof. Project the carrier of Definition 771.1. The only source and series data visible to the classifier are F and S, and the only uniform tail gate is the shared threshold row N.

Read the uniform criterion next. The row T consumes F, S, N as one finite family comparison, while the optional row M can contribute only by producing the same threshold surface through the Weierstrass majorant route.

Finally read R, E, H, C, P, Q. The regular readback and real seal are downstream of the shared threshold and uniform criterion, and the structural rows transport, replay, source, and name exactly that route. Since no other carrier coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 771.3 (Uniform convergence test Cauchy handoff)**.** *Every Real-facing or Completion-facing read from an accepted* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *carrier factors through the finite route*

$$F \longmapsto S \longmapsto N \longmapsto T \longmapsto R \longmapsto E$$

together with H, C, P, Q. If the optional majorant row M is present, it is read before T only to certify the same shared threshold N. Thus every consumer reaches $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *through the uniform Cauchy handoff, not through a pointwise convergence predicate, selected infinite sum, quotient function space, countable-choice schedule, or ambient completeness row.*

Proof. Start with a consumer request for a real-facing limit or completion-facing seal. By Definition 771.1, the request first reads the source-family row F and the displayed series or partial-sum row S.

The shared threshold row N is then the only tail gate available to the packet. Apply Definition 771.2: the uniform criterion row T consumes that threshold for all displayed family windows, and any majorant row M must feed the same threshold before T is accepted.

The handoff to regular rational data is therefore forced through R. Only after R is displayed may the route reach the $\mathsf{Real}^{\uparrow}$ seal E, with H, C, P, Q preserving and naming the same finite packet. The excluded host objects are absent from the carrier, so no consumer can bypass the shared-threshold route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceTest}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformConvergenceTest}^{\uparrow}$ packet is generated from a source family row, a $\mathsf{RealSeries}^{\uparrow}$ or partial-sum series row, one shared tail threshold, a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row, an optional $\mathsf{WeierstrassMTest}^{\uparrow}$ majorant row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 771.1, Definition 771.2, and Definition 771.3 bind the finite shared-tail route used by RealSeries, UniformCauchyCriterion, optional WeierstrassMTest, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close pointwise-only convergence, selected infinite sums, quotient function spaces, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-threshold exactness, optional majorant compatibility, RegSeqRat readback, Real seal handoff, and TasteGate field faithfulness for BEDC.Derived.UniformConvergenceTestUp.

Chapter 772

A Concrete Naming Certificate for $\mathsf{NadlerFixedPoint}^{\uparrow}$

Following Nadler and Bishop–Bridges constructive metric practice, the $\mathsf{NadlerFixedPoint}^{\uparrow}$ packet records the finite NameCert surface for a multivalued contraction on a complete metric carrier whose values are exposed through compact-set distance rows. It sits after the complete-metric convergence service of chapter 128, the Hausdorff compact-subset distance interface of chapter 2086, the single-valued contraction route of chapter 396, the displayed orbit discipline of chapter 2359, and the compact finite-inspection surface of chapter 129. The packet names only displayed source rows, compact value rows, Hausdorff contraction comparisons, finite choice-free refinement ledgers, transport, replay, provenance, and local naming; it does not import selected representatives, countable choice, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 772.1 (Nadler fixed-point carrier)**.** *A* $\mathsf{NadlerFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (X, K, F, V, D, L, O, R, H, C, P, Q)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *row, carrying the point classifier and distance observations inspected by the multivalued fixed-point route.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over X, exposing only the convergence service and limit-witness handoff accepted by the complete-metric certificate.*
3. *F is the multivalued self-map row: for each displayed source point it records a compact nonempty value name rather than an untracked host subset.*
4. *V records the* $\mathsf{CompactMetric}^{\uparrow}$ *value rows used to inspect the compact names supplied by F.*
5. *D records the* $\mathsf{HausdorffMetric}^{\uparrow}$ *distance rows between displayed values of F, including the finite-net and point-to-set witnesses used by the multivalued contraction comparison.*
6. *L is the rational contraction ledger with a displayed positive row and a displayed unit-bound row.*
7. *O is the finite Picard-style approximation ledger. Each step stores the current point row, the inspected compact value row, the next displayed point row, and the Hausdorff contraction budget used to continue the route.*

8. *R stores the Cauchy and completion handoff from O, L, D to K without forming a quotient completion or selecting a global branch outside the finite ledger.*

9. *H stores componentwise* hsame *transport for source, completeness, multivalued map, compact values, Hausdorff rows, contraction ledger, approximation ledger, completion handoff, replay, provenance, and naming rows.*

10. *C stores displayed* Cont *replay of the route*

$$X, K, F, V, D, L \longmapsto O \longmapsto R.$$

11. *P is the* Pkg *provenance over* $\mathsf{NadlerFixedPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *Q is the local* $\mathsf{NameCert}_{\mathsf{NadlerFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, V, D, L, O, R, H, C, P, Q$. It has no coordinate for a selected representative of each compact value, a countable choice operator, quotient-completion equality, a host hyperspace object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NadlerFixedPointUp.TasteGate.NadlerFixedPointTasteGate single carrier alignment`

Theorem 772.2 (Nadler fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{NadlerFixedPoint}^{\uparrow}$ *carrier $N = (X, K, F, V, D, L, O, R, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{NadlerFixedPoint}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the metric source X, complete-metric service K, multivalued map row F, compact value rows V, Hausdorff distance rows D, contraction ledger L, finite approximation ledger O, completion handoff R, transport H, replay C, provenance P, and local naming Q;*

2. *classifier transport, requiring accepted packet comparisons to preserve the same source classifier, compact value names, Hausdorff finite-net witnesses, contraction budget, approximation ledger, and completion handoff componentwise through H;*

3. *stability, requiring the Hausdorff contraction comparison in D, L to survive replacement of displayed source points and compact value names by classified equals before the next approximation row in O is admitted;*

4. *exactness and ledger nonescape, requiring the route from finite approximation rows to the complete-metric handoff R to pass through K and to export no selected representatives, no countable choice, no quotient completion, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate;*

5. *NameCert provenance, requiring C, P, Q to replay, source, and name only the displayed route over the CompleteMetric, HausdorffMetric, BanachFixedPoint, PicardIteration, and CompactMetric sibling surfaces cited above.*

No Nadler fixed-point obligation is stored outside these rows.

Proof. Project the carrier of Definition 772.1. The source metric row X, complete-metric service K, multivalued map row F, compact value rows V, Hausdorff distance rows D, rational contraction ledger L, approximation ledger O, and completion handoff R are the only mathematical coordinates available to the certificate.

The classifier is therefore componentwise. Transport through H may replace displayed source points, compact value names, Hausdorff witnesses, and ledger rows only by the recorded hsame

routes; replay through C preserves the same finite path from compact-value inspection to the completion handoff.

The stability obligation is the multivalued analogue of the contraction step. It reads the Hausdorff comparison from D, L and admits the next row in O only through the displayed value names and finite-net witnesses, not through a hidden branch selector.

Finally R, P, Q repack the finite route as the local naming surface. Since the carrier has no coordinate for selected representatives, countable choice, quotient-completion equality, a host hyperspace object, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the five listed rows exhaust the NameCert obligations. □

Theorem 772.3 (Nadler approximation route nonescape)**.** *Every fixed-point read exported by an accepted* $\mathsf{NadlerFixedPoint}^{\uparrow}$ *packet factors through the displayed route*

$$X, K, F, V, D, L \longmapsto O \longmapsto R \longmapsto H, C, P, Q.$$

The route may compare with the $\mathsf{BanachFixedPoint}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$ *siblings as single-valued reference surfaces, but it exports only compact-value Hausdorff contraction data and complete-metric handoff rows. It provides no selected representative sequence, no countable choice principle, no quotient completion, no host hyperspace completion, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with a fixed-point read from the packet. The carrier first exposes X, K, F, V, D, L, so the consumer can inspect only the displayed source metric, complete-metric service, multivalued map row, compact value rows, Hausdorff finite-net witnesses, and rational contraction budget.

The approximation ledger O is finite at every displayed stage. It records the point row, the inspected compact value row, the next point row, and the contraction comparison used by that stage; it never stores an external sequence selected by countable choice.

The handoff row R then routes the approximation ledger to the complete-metric service K. The structural rows H, C, P, Q transport, replay, source, and name this same path, so the exported fixed-point read cannot leave the displayed packet for the excluded host principles. □

Falsifiable prediction: Every $\mathsf{NadlerFixedPoint}^{\uparrow}$ fixed-point read factors through one MetricSpaceUp source row, one CompleteMetricUp service row, one multivalued self-map row, CompactMetricUp value rows, HausdorffMetricUp distance rows, one rational contraction ledger, one finite approximation ledger, one completion handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: complete-metric, hausdorff-metric, banach-fixed-point, picard-iteration, compactmetric: $\mathsf{NadlerFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-valued multivalued map rows, Hausdorff contraction comparisons, finite choice-free approximation ledgers, complete-metric handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NadlerFixedPoint}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NadlerFixedPoint}^{\uparrow}$ packet is generated from a MetricSpaceUp source row, CompleteMetricUp convergence service, compact-valued multivalued self-map row, CompactMetricUp value rows, HausdorffMetricUp distance witnesses, rational contraction ledger below the unit bound, finite Picard-style approximation ledger, complete-metric handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NadlerFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 772.1, Definition 772.2, and Definition 772.3 bind the multivalued contraction certificate surface through displayed compact values, Hausdorff comparisons, finite approximation ledgers, and complete-metric handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NadlerFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected representatives, countable choice, quotient completion, host hyperspace completion, ambient Real completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, Hausdorff contraction stability, approximation-route nonescape, exactness, provenance, and local naming for BEDC.Derived.NadlerFixedPointUp.

Chapter 773

A Concrete Naming Certificate for $\mathsf{BrouwerianCounterexample}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bridges–Richman on constructive counterexamples, the $\mathsf{BrouwerianCounterexample}^{\uparrow}$ packet records a finite refusal surface for off-table L10 readings. It sits beside the finite observation route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Specker obstruction boundary of chapter 3516. The packet makes the refusal of trichotomy, selected limits, quotient real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness inspectable as NameCert data rather than as an implicit metatheoretic convention.

Definition 773.1 (Brouwerian counterexample carrier)**.** *A* $\mathsf{BrouwerianCounterexample}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, A, S, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the tested L10 claim row, naming the proposed off-table reading such as trichotomy, selected limit extraction, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
2. *A is the finite counterexample source row, carrying the Brouwerian choice-sequence, bar, or Specker-style obstruction data used to test Q.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source is inspected by a downstream Real or Completion consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows in S.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *readback boundary available only after the source, window, regular-rational, and refusal rows have been displayed.*
6. *L is the refusal ledger recording which tested reading in Q is not exported by the packet and which finite counterexample source row A witnesses the refusal.*
7. *H records componentwise* hsame *transport for the tested claim, counterexample source, stream window, regular readback, Real boundary, refusal ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the tested claim through the counterexample source, then through stream windows, regular readback, refusal ledger, and the terminal readback boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerianCounterexample}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerianCounterexample}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, S, R, E, L, H, C, P, N$. It has no coordinate for a completed host counterexample object, an omniscience principle, real trichotomy, quotient real equality, selected limits, countable choice, or ambient Real completeness.

Theorem 773.2 (Brouwerian counterexample NameCert obligations)**.** *For every accepted* BrouwerianCounterexamp *carrier $B = (Q, A, S, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BrouwerianCounterexample}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tested L10 claim, finite counterexample source, StreamName windows, RegSeqRat readback, Real boundary, refusal ledger, transport, replay, provenance, and local naming rows;*
2. *source exactness, requiring the row A to be the only counterexample source used when the tested claim row Q is evaluated;*
3. *windowed readback, requiring every Real or Completion-facing read to open S and R before consulting the Real boundary E;*
4. *refusal-ledger exactness, requiring the ledger L to bind each refused reading to the displayed source and readback rows rather than to an uninspected host argument;*
5. *transport and replay stability, requiring H and C to preserve the same tested claim, source, windows, readback, boundary, and refusal rows;*
6. *nonescape for omniscience, trichotomy, quotient real equality, selected limits, countable choice, and ambient Real completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 773.1. The only analytic coordinates are the tested claim Q, finite counterexample source A, window row S, readback row R, Real boundary E, and refusal ledger L.

Source exactness and windowed readback follow from the order of the displayed route. The source row A is the only row that may test Q, while any Real or Completion-facing consumer must first read S and R before reaching E.

The refusal ledger then records a finite negative boundary. It binds each refused off-table reading to the inspected source and readback rows; it does not certify a detached host counterexample or a completed real object.

The structural rows H, C, P, N only transport, replay, source, and name the same packet. Since the carrier has no coordinate for omniscience, trichotomy, quotient real equality, selected limits, countable choice, or ambient Real completeness, none of those readings can be exported as local NameCert data. □

Lean4 ✓ `BEDC.Derived.BrouwerianCounterexampleUp.BrouwerianCounterexampleTasteGate single carrier alignment`

Theorem 773.3 (Brouwerian counterexample Real boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BrouwerianCounterexample}^{\uparrow}$ *carrier must read the tested claim row, the finite counterexample source, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the refusal ledger*

before the $\mathsf{Real}^{\uparrow}$ *boundary is exposed. The boundary therefore supplies finite counterexample readback only; it supplies no real trichotomy, quotient real equality, selected limit, countable-choice selector, omniscience principle, or ambient Real completeness.*

Proof. Begin with a Real or Completion-facing read of $B = (Q, A, S, R, E, L, H, C, P, N)$. The only Real-facing coordinate is E, and carrier admission places it after the tested claim, source, window, readback, and ledger rows.

Open the readback route. The consumer first inspects Q and A, then obtains the finite windows through S and the regular rational data through R. Only then can the refusal ledger L say which off-table reading is not exported by the displayed source.

The endpoint row E is therefore a boundary, not a completeness theorem. It records the finite Real readback surface shared with chapter 38, while the source and window discipline remain those of chapter 126 and chapter 301.

The structural rows preserve the same route. Since the neighbouring Specker boundary of chapter 3516 also refuses located-supremum escape through finite rows, the present packet can cite that style of obstruction without exporting trichotomy, quotient real equality, selected limits, countable choice, omniscience, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{BrouwerianCounterexample}^{\uparrow}$ Real or Completion read factors through a tested L10 claim row, a finite counterexample source, a StreamName window, a RegSeqRat readback, a Real boundary, a refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerianCounterexample}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerianCounterexample}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 773.1, Definition 773.2, and Definition 773.3 seed the finite Brouwerian counterexample packet over tested L10 claims, finite source rows, StreamName windows, RegSeqRat readback, Real boundary, and refusal ledger data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerianCounterexample}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, real trichotomy, quotient real equality, selected limits, countable choice, omniscience principles, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source exactness, windowed readback, refusal-ledger exactness, transport and replay stability, Real boundary, and nonescape for BEDC.Derived.BrouwerianCounterexampleUp.

Constructive story: A $\mathsf{BrouwerianCounterexample}^{\uparrow}$ packet is generated from a tested L10 claim row, a finite counterexample source, StreamName windows, RegSeqRat readback, a Real boundary row, a refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Chapter 774

A Concrete Naming Certificate for $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$

Following the classical Bairstow quadratic-factor iteration in a Bishop-style finite-window reading, the $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$ packet records only the displayed data needed to route a polynomial root-window computation toward Real-facing consumers. It sits after the polynomial coefficient interface of chapter 50, the real-name seal of chapter 38, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the finite stream windows of chapter 126, and the bisection-to-Cauchy seal of chapter 1798. The packet is a finite root-window certificate; it is not host algebraic closure, quotient real equality, selected complex roots, global factorization, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 774.1 (Finite Bairstow root-window carrier)**.** *A* $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, U, V, R, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Poly}^{\uparrow}$ *coefficient-window row naming the finite list of inspected coefficients and the active degree bound.*
2. *U is the quadratic-factor update row recording the displayed Bairstow correction window for the current pair of quadratic coefficients.*
3. *V is the residual-budget ledger measuring the finite remainder, slack, and admissible stopping budget.*
4. *R is the dyadic bracketing row that turns the retained quadratic or linear factor into a sign-change or enclosure window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by V and R.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row for coefficient, residual, and bracket requests.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row handed to the* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *and terminal* $\mathsf{Real}^{\uparrow}$ *consumers.*
8. *H records componentwise* hsame *transport for coefficient windows, update rows, residual budgets, dyadic brackets, tolerance ledgers, stream windows, regular readbacks, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from coefficient inspection through update, residual, bracketing, regular readback, and Real-seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BisectionRootCauchySeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteBairstowRootWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, U, V, R, D, S, Q, H, C, P, N$. It has no coordinate for host algebraic closure, quotient real equality, selected roots, unrestricted factorization, countable search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 774.2 (Finite Bairstow root-window NameCert obligations)**.** *For every accepted* FiniteBairstowRootWind *carrier $B = (A, U, V, R, D, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteBairstowRootWindow}^{\uparrow}}$ *surface consists exactly of coefficient-window admission, quadratic-factor update admission, residual-budget ledger exposure, dyadic bracketing-window exposure,* $\mathsf{StreamName}^{\uparrow}$ *finite observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. The route refuses host algebraic closure, quotient real equality, selected complex roots, global factorization, countable root search, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 774.1. The coefficient window A, update row U, residual ledger V, bracket row R, tolerance ledger D, stream row S, and regular readback Q are the only analytic coordinates available to the packet.

The finite update route first inspects coefficients through A and S. The Bairstow correction row U and residual-budget row V are displayed finite rows, so the certificate never reads a host factorization theorem or a selected root.

Next the dyadic bracket R and tolerance ledger D feed the regular readback Q. The $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive only that finite bracketing and regular-Cauchy handoff, not quotient equality or ambient completeness.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores algebraic closure, selected roots, unrestricted factorization, countable search, quotient real equality, or a completed Real object, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$ consumer that exports a root-window read without displaying coefficient windows, quadratic-factor update rows, residual budgets, dyadic brackets, StreamName observations, RegSeqRat readback, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: polynomial, real, dyadicratcore, regseqrat, streamname, bisection-root-cauchy-seal: no listed sibling jointly carries coefficient windows, Bairstow update rows, residual budgets, dyadic brackets, regular readback, bisection-to-Cauchy handoff, and local NameCert data while refusing host algebraic closure and quotient root equality.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBairstowRootWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteBairstowRootWindow}^{\uparrow}$ packet is generated from polynomial coefficient windows, quadratic-factor update rows, residual-budget ledgers, dyadic bracketing windows, StreamName observations, RegSeqRat readback, a BisectionRootCauchySeal handoff, a Real-facing seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteBairstowRootWindow}^{\uparrow}}$, seed)

Scope closed: Definition 774.1 and Definition 774.2 seed the finite Bairstow root-window packet over polynomial coefficient windows, quadratic-factor updates, residual budgets, dyadic brackets, StreamName, RegSeqRat, BisectionRootCauchySeal, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteBairstowRootWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host algebraic closure, quotient real equality, selected roots, global factorization, countable root search, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires finite update exactness, residual-budget determinacy, dyadic bracket handoff, Real-seal nonescape, transport stability, replay stability, and ledger refusal for BEDC.Derived.FiniteBairstowRootWindowUp.

Chapter 775

A Concrete Naming Certificate for MetaCICTypedExamplesGallery$^{\uparrow}$

The MetaCICTypedExamplesGallery$^{\uparrow}$ packet realizes the typed-example audit map of chapter 9382 as a finite NameCert gallery. It sits beside the confluence audit surface of chapter 743 and the closed-term substitution boundary of chapter 527: those chapters constrain global metatheorem reads, while this chapter exposes the worked syntax and typing examples as finite rows.

Definition 775.1 (MetaCIC typed-examples gallery carrier)**.** *A* MetaCICTypedExamplesGallery$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M)$$

with the following displayed rows:

1. *B records the Church Bool example rows, including the displayed true, false, and eliminator typing reads;*
2. *N records the Church Nat example rows, including zero, successor, and numeral typing reads;*
3. *L records the Church List example rows, including nil, cons, and eliminator typing reads;*
4. *S records the Church Sum example rows, including left injection, right injection, and case-analysis typing reads;*
5. *E records equality example rows, including reflexivity and transport-shaped reads that stay inside the displayed typing ledger;*
6. *I records identity example rows, including polymorphic identity and dependent identity examples;*
7. *A records application example rows, including function application and argument instantiation reads;*
8. *R records beta-redex example rows and their displayed reduction endpoints;*
9. *K records checker-acceptance rows for the example families named by B, N, L, S, E, I, A, R;*
10. *H records componentwise* hsame *transport for every gallery row, route row, provenance row, and local naming row;*
11. *C records the displayed* Cont *routes for syntax readback, typing readback, beta-redex replay, checker-acceptance replay, and consumer exposure;*

12. *P records* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, *the MetaCIC typed-example rows, and the checker rows named by chapter 9382;*
13. *M is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It has no coordinate for arbitrary closed-term normalization, confluence, subject reduction, conversion decidability, or a complete algebraic theory of the Church encodings.

Theorem 775.2 (MetaCIC typed-examples gallery obligation matrix)**.** *Each accepted* MetaCICTypedExamplesGalle *typed example row must expose its context row, term row, typing derivation row, reduction-window row, local-join row, normalization-frontier row,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *row before it can be consumed by* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*. The matrix is read from the finite gallery rows*

$$B, N, L, S, E, I, A, R, K, H, C, P, M,$$

with local-join and normalization-frontier access routed through the beta-redex and checker rows R, K rather than through a global confluence, subject-reduction, normalization, or conversion-completeness theorem.

Proof. Project the carrier of Definition 775.1. The family rows B, N, L, S, E, I, A, R are the only context, term, typing, and reduction-window sources available to an accepted gallery example, while K is the checker boundary attached to those displayed sources.

The local-join and normalization-frontier reads are therefore sibling routes of the finite beta-redex and checker surface. They are the same finite rows later isolated by Definition 775.6 and Definition 775.7, with the redex-frontier handoff boundary recorded in Definition 1008.6.

Finally add the structural rows. Transport is projected through H, replay through C, provenance through P, and local naming through M. Since the carrier has no coordinate for global confluence, subject reduction, normalization, or conversion completeness, $\mathsf{MetaCICCriticalPath}^{\uparrow}$ can consume only the displayed obligation matrix. □

Theorem 775.3 (MetaCIC typed-examples gallery NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$ *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$ *surface consists exactly of carrier exposure for the finite gallery, representative theorem inventory for the Church Bool, Nat, List, Sum, equality, identity, application, and beta-redex rows, regression exactness for the displayed example endpoints, checker handoff through K, non-escape from global metatheorems, and ledger provenance through P, M.*

Proof. Project the carrier of Definition 775.1. The rows B, N, L, S, E, I, A, R are the complete representative theorem inventory exposed by this gallery packet.

The checker handoff is the row K, and every read of K is replayed through the displayed route row C. Componentwise transport through H preserves the same example rows and does not add a new theorem coordinate.

Finally read provenance through P and local naming through M. Since the carrier has no coordinate for subject reduction, confluence, normalization, or conversion decidability, the gallery obligations are exhausted by the finite example rows and their checker-acceptance routes. □

Theorem 775.4 (MetaCIC typed-examples gallery obligation surface)**.** *For every accepted* MetaCICTypedExamples *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the obligation surface consists exactly of the Church Bool row B, Church Nat row N, Church List row L, Church Sum row S, equality row E, identity row I, application row A, beta-redex row R, checker-acceptance row K, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$ *row M. It excludes subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, and checker completeness.*

Proof. Project the accepted carrier of Definition 775.1. The rows B, N, L, S, E, I, A, R are the only example-family rows, and K is the only checker-acceptance row attached to them.

The structural support is also displayed. Componentwise transport is read through H, replay through C, provenance through P, and local naming through M.

There is no carrier coordinate for subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness. Therefore the obligation surface is bounded by exactly the displayed rows. □

Theorem 775.5 (MetaCIC typed-examples gallery regression surface)**.** *An accepted* MetaCICTypedExamplesGallery *packet is a finite regression surface for represented syntax and typing. Consumers may inspect the Church Bool, Nat, List, Sum, equality, identity, application, beta-redex, checker-acceptance,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows, but they cannot read the packet as a subject-reduction theorem, a confluence theorem, a normalization theorem, or a decidability theorem.*

Proof. Start with the visible coordinates of G. They are example rows and routes for syntax, typing, beta-redex replay, checker acceptance, transport, provenance, and local naming.

Each consumer read factors through the finite route row C and the checker row K. These rows can replay the represented gallery examples, but they have no endpoint for a global preservation or conversion principle.

Therefore the accepted packet is exactly a finite regression surface for the displayed examples. The excluded metatheorems remain outside the carrier because no row of G can be projected as such a theorem. □

Theorem 775.6 (MetaCIC typed-examples gallery typed-family coverage)**.** *For every accepted* $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$ *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

every typed-example family read exposed by the gallery factors through one of the displayed family rows B, N, L, S, E, I, A, R and then through the checker row K, componentwise transport row H, replay row C, provenance row P, and local naming row M. The gallery exposes no further subject-reduction family, confluence family, normalization family, conversion-completeness family, arbitrary-example-coverage family, or checker-completeness family.

Proof. Project the carrier of Definition 775.1. Its only typed-example family rows are the Church Bool, Church Nat, Church List, Church Sum, equality, identity, application, and beta-redex rows named by B, N, L, S, E, I, A, R.

Now split over that finite displayed family inventory. A Bool read is routed through B, a Nat read through N, a List read through L, a Sum read through S, an equality read through E, an identity read through I, an application read through A, and a beta-redex read through R.

Checker acceptance is then forced through K. Transport of the same family row is read through H, replay is read through C, and the exported consumer-facing certificate is packaged by P and named by M.

Finally apply the regression surface of Definition 775.5. Since the carrier has no coordinate for subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness, no additional typed-family read is exposed by the gallery. □

Theorem 775.7 (MetaCIC typed-examples gallery checker-route factorization)**.** *For every accepted* $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$ *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

every gallery consumer read factors through exactly one displayed example-family row among B, N, L, S, E, I, A, R *and then through* K, H, C, P, M*. The checker row* K *is therefore only the finite acceptance boundary for the chosen displayed family. It cannot be read as subject reduction, confluence, normalization, conversion completeness, or checker completeness.*

Proof. Project the carrier of Definition 775.1. Its displayed example-family rows are exactly B, N, L, S, E, I, A, R, so a gallery read first chooses one of those finite rows.

Case split over the displayed family row. Bool reads route through B, Nat reads through N, List reads through L, Sum reads through S, equality reads through E, identity reads through I, application reads through A, and beta-redex reads through R.

After that family choice, acceptance must pass through K, and the same accepted route is transported by H, replayed by C, sourced by P, and named by M.

Finally cite the regression surface theorem Definition 775.5. Since the packet has no coordinate for the excluded global metatheorems, the checker row remains a finite family acceptance boundary and cannot be consumed as subject reduction, confluence, normalization, conversion completeness, or checker completeness. □

Lean4 ✓ `BEDC.MetaCIC.gallery checker route factorization`

Theorem 775.8 (MetaCIC typed-examples gallery kernel scope)**.** *For every accepted* MetaCICTypedExamplesGaller *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the kernel-facing scope of the gallery is exactly the finite BHist *route named by Definition 775.1, constrained by the obligation matrix of Definition 775.2, the typed-family coverage of Definition 775.6, and the checker-route factorization of Definition 775.7. A kernel consumer may inspect the displayed family row, checker row, transport row, replay row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$ *row. It receives no subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness theorem.*

Lean4 ✓ `BEDC.MetaCIC.gallery kernel scope`

Proof. Project the carrier of Definition 775.1. The only kernel-facing rows are the displayed family rows B, N, L, S, E, I, A, R, the checker row K, transport row H, replay row C, provenance row P, and local naming row M.

Apply the obligation matrix of Definition 775.2. It forces each consumed example to display its context, term, typing, reduction-window, local-join, normalization-frontier, transport, replay, provenance, and local naming rows inside the same finite packet.

Now apply the typed-family coverage and checker-route factorization theorems. They place every family read inside one displayed row and then through K, H, C, P, M. Since the carrier has no coordinate for the excluded global metatheorems, the kernel-facing scope is exactly this finite route. □

Theorem 775.9 (MetaCIC typed-examples gallery coverage scope)**.** *For every accepted* MetaCICTypedExamplesGal *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the coverage claim exported by the gallery is scoped to the displayed Church Bool, Church Nat, Church List, Church Sum, equality, identity, application, and beta-redex rows. The claim is read through the carrier of Definition 775.1, the obligation matrix of Definition 775.2, the typed-family coverage of Definition 775.6, and the checker-route factorization of Definition 775.7. It does not cover arbitrary examples, checker completeness, subject reduction, confluence, normalization, or conversion completeness.

Proof. Start with the finite family inventory in the carrier. It lists exactly the rows B, N, L, S, E, I, A, R and gives no row for an arbitrary example family.

The obligation matrix requires each displayed family read to keep its local typing, reduction-window, checker, transport, replay, provenance, and naming data visible. Thus coverage is a row-by-row property of the finite inventory, not a theorem about every possible MetaCIC example.

Finally use typed-family coverage and checker-route factorization. The first theorem says every gallery family read factors through the displayed family rows, and the second says each consumer read then passes through K, H, C, P, M. The omitted global claims remain outside the scope because no carrier row names them. □

Theorem 775.10 (MetaCIC typed-examples counterexample boundary)**.** *An accepted* MetaCICTypedExamplesGall *row can be consumed as a finite regression row or as a finite counterexample boundary only when its context row, term row, typing row, reduction-window row, checker row, transport row, replay row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$ *row are all displayed inside*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M).$$

The row cannot be read as global subject reduction, and it also cannot be read as a global failure of subject reduction. It is only bounded evidence about the displayed Church Bool, Church Nat, Church List, Church Sum, equality, identity, application, or beta-redex family chosen from B, N, L, S, E, I, A, R.

Proof. Project the carrier of Definition 775.1. The finite family row supplies the context, term, typing, and reduction-window data, while K supplies the checker boundary and H, C, P, M supply transport, replay, provenance, and local naming.

Apply the coverage scope of Definition 775.9. It restricts every exported claim to the displayed family rows and refuses arbitrary example coverage.

Now apply checker-route factorization from Definition 775.7. Any consumer read must pass through one displayed family and then through K, H, C, P, M, so a regression or counterexample use remains a finite row-boundary claim.

Since the carrier has no coordinate for global subject reduction and no coordinate for a global countertheorem, neither conclusion can be projected from the gallery packet. □

Theorem 775.11 (MetaCIC typed-examples gallery scoped route)**.** *Every consumer route out of an accepted* MetaCICTypedExamplesGallery$^{\uparrow}$ *packet is a scoped route through*

$$B, N, L, S, E, I, A, R, K, H, C, P, M.$$

The route is determined by the carrier of Definition 775.1, the obligation matrix of Definition 775.2, the typed-family coverage theorem of Definition 775.6, and the checker-route factorization theorem of Definition 775.7. The route exports only the bounded gallery surface and keeps subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, and checker completeness outside the claim.

Proof. Choose a consumer read from an accepted gallery packet. By the carrier definition, the read must start inside one of the displayed rows $B, N, L, S, E, I, A, R, K, H, C, P, M$.

If the read asks for example-family content, typed-family coverage forces a unique displayed family entrance among B, N, L, S, E, I, A, R. Checker-route factorization then sends the same read through K, H, C, P, M.

If the read asks for structural certificate content, the obligation matrix restricts it to the same transport, replay, provenance, and local naming rows. Therefore every consumer route remains inside the displayed scoped route, and the global metatheorem claims are not exported by this packet. □

Theorem 775.12 (MetaCIC typed-examples gallery scope rows)**.** *For every accepted* MetaCICTypedExamplesGaller *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the accepted gallery scope is exactly the row set

$$B, N, L, S, E, I, A, R, K, H, C, P, M.$$

Rows B, N, L, S, E, I, A, R are the displayed typed-family rows, K is the checker boundary for those families, and H, C, P, M are the transport, replay, provenance, and local naming rows. The scope contains no subject-reduction row, confluence row, normalization row, conversion-completeness row, arbitrary-example row, or checker-completeness row.

Proof. Project the carrier of Definition 775.1. Its displayed coordinates are precisely $B, N, L, S, E, I, A, R, K, H, C,$ so every accepted row must be one of those coordinates.

Apply the obligation matrix of Definition 775.2. It requires each accepted gallery read to display its context, term, typing, reduction-window, local-join, normalization-frontier, transport, replay, provenance, and local naming data inside the finite carrier.

Next use typed-family coverage and checker-route factorization. The theorem Definition 775.6 confines every family read to B, N, L, S, E, I, A, R, and Definition 775.7 sends each such read through K, H, C, P, M.

Finally cite the scoped route of Definition 775.11. It excludes the global metatheorem rows, so the accepted scope is exactly the displayed thirteen-row surface. □

Theorem 775.13 (MetaCIC typed-examples gallery scope export)**.** *Every exported gallery read from an accepted* MetaCICTypedExamplesGallery$^{\uparrow}$ *packet is exactly one displayed family row among*

$$B, N, L, S, E, I, A, R$$

followed by the finite checker and certificate route

$$K, H, C, P, M.$$

No consumer may read the exported gallery as global subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness.

Proof. Project the carrier of Definition 775.1. The only family rows available for export are B, N, L, S, E, I, A, R, and the only post-family route rows are K, H, C, P, M.

Apply the obligation matrix and typed-family coverage theorems, Definition 775.2 and Definition 775.6. They force each exported family read to keep its displayed context, term, typing, reduction-window, checker, transport, replay, provenance, and local naming data inside the finite carrier.

Use the checker-route factorization and scoped route theorems, Definition 775.7 and Definition 775.11. These identify the consumer route as one family row followed by K, H, C, P, M, with no bypass through a global checker theorem.

Finally apply the scope rows theorem Definition 775.12. Since the accepted scope is exactly the thirteen displayed rows, the excluded global metatheorems and arbitrary coverage claims are not exportable from the gallery packet. □

Theorem 775.14 (MetaCIC typed-examples gallery scoped-closure route)**.** *For every accepted* $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$ *carrier*

$$G = (B, N, L, S, E, I, A, R, K, H, C, P, M),$$

the scoped-closure route is exhausted by $B, N, L, S, E, I, A, R, K, H, C, P, M$ *together with the obligation matrix, typed-family coverage, checker-route factorization, scoped-route theorem, and scope-rows theorem. The route refuses subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, and checker completeness as claims of the gallery packet.*

Proof. Project the carrier of Definition 775.1. Its only visible coordinates are the displayed family rows B, N, L, S, E, I, A, R, the checker boundary K, and the structural rows H, C, P, M.

Apply the obligation matrix, coverage theorem, and checker-route factorization: Definition 775.2, Definition 775.6, and Definition 775.7. They keep every accepted example read inside the displayed family rows and then route it through K, H, C, P, M.

Now use the scoped-route theorem and the scope-rows theorem, Definition 775.11 and Definition 775.12. These identify the complete accepted route with $B, N, L, S, E, I, A, R, K, H, C, P, M$.

Since no carrier row names subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness, those global properties cannot be projected from the scoped gallery route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}$ packet is generated from finite BHist rows for selected typed examples, checker rows, beta or conversion readbacks, route rows, provenance, and local naming. Each gallery entry is replayable by Cont through the finite checker surface, so consumers can inspect represented examples without receiving a global calculus theorem. Componentwise hsame preserves the example, checker, and route rows, while Pkg records the MetaCIC typed-example dependencies. The gallery is therefore a bounded regression surface for displayed examples and not a proof of preservation, normalization, or checker completeness.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}, \mathsf{obligation})$

Scope closed: This chapter binds the MetaCIC typed-examples gallery carrier, obligation matrix, checker-route factorization, finite regression surface, typed-family coverage, and exclusion of global metatheorems.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesGallery}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim global subject reduction, confluence, normalization, conversion completeness, arbitrary example coverage, or checker completeness.

Upgrade path: Obligation closure requires Lean encodings of the gallery carrier, checker-route replay, componentwise transport, and regression-surface non-escape theorem.

Chapter 776

A Concrete Naming Certificate for MetaCICAuditSynthesisPacket$^{\uparrow}$

The MetaCICAuditSynthesisPacket$^{\uparrow}$ surface turns the synthesis overview of chapter 9385 into one finite naming certificate. It sits beside the closed-normal packet of chapter 739, the normal-false exclusion packet of chapter 732, and the discharge-obligation ledger of chapter 717: those siblings separate the positive closed-normal branch from the subject-reduction obstruction, while the present packet records the combined audit surface without closing the missing transport edge.

Definition 776.1 (MetaCIC audit synthesis packet carrier)**.** *A* MetaCICAuditSynthesisPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, C, N, U, D, A, B, H, R, P, L)$$

with the following visible rows:

1. *S is the subject-reduction obstruction row, including the dependent-codomain boundary and the four named discharge slots;*
2. *C is the confluence map row, recording the conditional diamond route, closed-normal fragment, and confluence-audit packet references;*
3. *N is the normalization row, recording the candidate-mediated closed strong-normalization routes used by the closed-normal branch;*
4. *U is the substitution row, recording closed shift, closed substitution, and closed composition routes needed by the discharge surface;*
5. *D is the decidable-checking row, recording checked term equality, typing inference, gallery witnesses, and bounded normal comparison with its boundary;*
6. *A is the closed-normal assembly row, tying closed strong normalization, normal-false exclusion, and the closed-normal consistency packet into the positive branch;*
7. *B is the blocker row, recording that the synthesis does not supply closed subject reduction, arbitrary closed Church–Rosser, candidate-free normalization, global beta-conversion decidability, or full closed consistency;*
8. *H records componentwise* hsame *transports for S, C, N, U, D, A, B and the local structural rows;*
9. *R records the displayed* Cont *routes by which consumers read the audit rows in synthesis order;*

10. *P is the* Pkg *provenance over the MetaCIC audit maps, chapter 739, chapter 732, and chapter 717;*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *row.*

The classifier compares accepted synthesis packets only through the displayed rows $S, C, N, U, D, A, B, H, R, P, L$. The carrier has no coordinate for a hidden proof of subject reduction, an ambient consistency theorem, quotient equality of MetaCIC terms, or an external normalization oracle.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisPacketTasteGate single carrier alignment`

Theorem 776.2 (MetaCIC audit synthesis certificate surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *surface for Definition 776.1 consists exactly of carrier admission for $M = (S, C, N, U, D, A, B, H, R, P, L)$, audit-row exposure for the subject-reduction, confluence, normalization, substitution, decidable-checking, closed-normal assembly, and blocker rows, componentwise* hsame *transport through H, displayed* Cont *route replay through R, package provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisCertificateSurface`

Proof. Project the finite packet of Definition 776.1. The rows $S, C, N, U, D, A, B, H, R, P, L$ are the complete certificate surface, so carrier admission and classifier exactness are componentwise projections.

Read the positive audit rows next. The confluence row, normalization row, substitution row, decidable-checking row, and closed-normal assembly row cite the same audit-map separation stated in chapter 9385. The closed-normal side is routed through the packet of chapter 739 and the normal-false exclusion surface of chapter 732.

Finally read B, H, R, P, L. The blocker row keeps the subject-reduction gap explicit, while the transport, continuation, provenance, and naming rows only replay the displayed finite packet. Thus the local certificate cannot export an unlisted MetaCIC theorem. □

Theorem 776.3 (MetaCIC audit synthesis critical-path non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier reads the critical path through*

$$S, C, N, U, D, A, B, H, R, P, L.$$

It may inspect the audit synthesis rows and their finite transports, routes, provenance, and local name. It cannot read a second closed-consistency route, a hidden subject-reduction proof, an unrestricted Church–Rosser theorem, candidate-free normalization, global beta-conversion decidability, quotient equality of terms, or host-level completeness.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisCriticalPathNonescape`

Proof. Start with carrier admission. Since the classifier of Definition 776.1 compares only the displayed rows, every consumer read is a projection from $S, C, N, U, D, A, B, H, R, P, L$ or a route recorded by R.

Use the sibling audit packets to identify the positive branch. chapter 739 and chapter 732 expose the closed-normal endpoint, while the confluence, normalization, substitution, and decidable rows keep their local audit boundaries visible.

The blocker row is then read as a boundary, not as a solved theorem. Because the carrier has no additional coordinate for full closed consistency or closed subject reduction, no consumer can escape the displayed critical path. □

Theorem 776.4 (MetaCIC audit synthesis consistency blocker ledger)**.** *The consistency blocker of a* MetaCICAuditSynthesisPacket$^{\uparrow}$ *packet is exactly the displayed row B together with the subject-reduction row S. Agreement on $C, N, U, D, A, H, R, P, L$ cannot be repacked as closed subject reduction, closed reduction consistency, or a transport from arbitrary closed proof terms into the closed-normal exclusion branch.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisConsistencyBlockerLedger`

Proof. Inspect S first. It names the dependent-codomain obstruction and the four discharge slots, matching the finite ledger of chapter 717.

Inspect B next. The blocker row records the unavailable closed transport edge as part of the packet, so the remaining positive rows cannot erase it.

The closed-normal assembly row A can be consumed only with its stated boundary. It supports the positive branch named by the closed-normal and normal-false sibling chapters, but it does not create a reduction route from arbitrary closed proofs. Therefore agreement on the other rows leaves the blocker ledger intact. □

Theorem 776.5 (MetaCIC audit synthesis closed-normal boundary)**.** *The closed-normal branch of an accepted* MetaCICAuditSynthesisPacket$^{\uparrow}$ *packet is exhausted by the normalization row N, the closed-normal assembly row A, the normal-false exclusion boundary, and the structural rows H, R, P, L. The packet may certify the closed-normal consistency branch as named finite audit data, but it does not certify full closed MetaCIC consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisClosedNormalBoundary`

Proof. Project the normalization and assembly rows from Definition 776.1. These rows are the only coordinates that feed the closed-normal branch.

Use the sibling boundaries. The closed-normal packet of chapter 739 and the normal-false exclusion packet of chapter 732 identify the branch reached after normality is already displayed.

It remains to compare this branch with the blocker row. The subject-reduction ledger of chapter 717 keeps the missing transport edge outside the closed-normal rows, so the packet stops at the finite closed-normal boundary. □

Theorem 776.6 (MetaCIC audit synthesis route-union exactness)**.** *For every accepted* MetaCICAuditSynthesisPack *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any packet consumer that reads confluence, normalization, substitution, decidable-checking, or closed-normal assembly must also carry the blocker row B and the subject-reduction row S through the same transport, route, provenance, and naming rows H, R, P, L. Thus the positive rows C, N, U, D, A and the blocker row B form one exact audit-route union over S, H, R, P, L. The packet supplies no hidden subject reduction, unrestricted Church–Rosser theorem, candidate-free normalization, global beta-conversion decidability, or quotient equality.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisRouteUnionExactness`

Proof. Use the certificate surface of Definition 776.2. It identifies $S, C, N, U, D, A, B, H, R, P, L$ as the complete finite carrier and local naming surface, so the positive audit rows and blocker row are read from one packet.

Next apply Definition 776.3. Every consumer route stays inside the displayed critical path, and the only transport, route, provenance, and naming rows available to that route are H, R, P, L.

The blocker side is fixed by Definition 776.4. The row B is carried together with the subject-reduction row S, so agreement on confluence, normalization, substitution, decidable-checking, or assembly cannot erase the obstruction.

Finally use Definition 776.5. It places the closed-normal branch inside the named finite boundary rather than inside full closed consistency. Therefore the route union is exact: reading any positive row also carries S and B through H, R, P, L, while hidden subject reduction, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, and quotient equality remain outside the packet. □

Theorem 776.7 (MetaCIC audit synthesis substitution-confluence join)**.** *For every accepted* MetaCICAuditSynthesi *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any consumer that reads the closed-normal assembly row A through the confluence row C must also carry the substitution row U, the subject-reduction row S, and the blocker row B through the same structural route H, R, P, L. Agreement on C, N, D, A alone cannot be exported as closed subject reduction or unrestricted Church–Rosser, because the substitution transport row and blocker ledger remain part of the accepted packet boundary.

Proof. Start with the certificate surface of Definition 776.2. It makes $S, C, N, U, D, A, B, H, R, P, L$ the complete row inventory of the accepted packet, so a consumer reading A through C is still inside the same finite carrier.

Apply route-union exactness from Definition 776.6. The confluence, normalization, substitution, decidable-checking, and closed-normal assembly rows share the transport, route, provenance, and naming rows H, R, P, L with the blocker row B and the subject-reduction row S.

Finally use the consistency blocker ledger of Definition 776.4. It keeps S and B visible across agreement on the positive rows, so an agreement surface on C, N, D, A that omits U leaves the accepted packet rather than exporting closed subject reduction or unrestricted Church–Rosser. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisSubstitutionConfluenceJoin`

Theorem 776.8 (MetaCIC audit synthesis positive-blocker union)**.** *For every accepted* MetaCICAuditSynthesisPack *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any consumer that reads a positive row among confluence C, normalization N, substitution U, decidable checking D, or closed-normal assembly A must simultaneously carry the subject-reduction row S and blocker row B through the same componentwise transport, displayed route, provenance, and naming rows H, R, P, L. Thus no positive-branch read can be exported as full closed consistency, closed subject reduction, arbitrary closed Church–Rosser, candidate-free normalization, global beta-conversion decidability, or quotient equality of MetaCIC terms.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisPositiveBlockerUnion`

Proof. Apply the certificate surface of Definition 776.2. It gives the complete row inventory $S, C, N, U, D, A, B, H, R, P, L$, so a consumer that reads any one of C, N, U, D, A is still reading from the same finite packet as S and B.

Use critical-path non-escape from Definition 776.3. The only transport, route, provenance, and naming rows available to such a consumer are the displayed rows H, R, P, L.

Now apply the consistency blocker ledger of Definition 776.4. It makes S and B persistent across agreement on the positive rows, so the subject-reduction obstruction cannot be erased by confluence, normalization, substitution, decidable checking, or closed-normal assembly.

Finally use route-union exactness from Definition 776.6. The positive rows and blocker row share transport and provenance in one audit-route union. Hence a positive-branch consumer carries S and B through H, R, P, L whenever it reads C, N, U, D, or A, and the excluded full-consistency claims remain outside the packet. □

Theorem 776.9 (MetaCIC audit synthesis retained-blocker induction)**.** *For every accepted* MetaCICAuditSynthesis *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any finite fold that successively consumes a nonempty sublist of the positive rows

$$C, N, U, D, A$$

must carry the subject-reduction row S and blocker row B through H, R, P, L at every fold stage. The fold may expose confluence, normalization, substitution, decidable checking, and closed-normal assembly data, but it cannot export subject reduction, unrestricted Church–Rosser, candidate-free normalization, beta-conversion decidability, quotient equality of MetaCIC terms, or full consistency.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisRetainedBlockerInduction`

Proof. Induct on the displayed finite positive-row list C, N, U, D, A restricted to the nonempty sublist consumed by the fold.

For the base step, the first consumed positive row is one of C, N, U, D, A. Apply the positive-blocker union theorem, Definition 776.8. The consumer must already carry S and B through the same componentwise transport, route replay, provenance, and local naming rows H, R, P, L.

For the successor step, suppose the current fold prefix carries S and B through H, R, P, L. Consuming the next positive row is again a read of one of C, N, U, D, A from the same carrier M. Apply route-union exactness, Definition 776.6, to keep the new positive read in the same audit-route union, and apply the consistency-blocker ledger, Definition 776.4, to keep the obstruction rows persistent.

Thus every fold stage retains S and B through H, R, P, L. Since the carrier has no coordinate for subject reduction, unrestricted Church–Rosser, candidate-free normalization, beta-conversion decidability, quotient equality, or full consistency, the completed fold cannot export any of those claims. □

Theorem 776.10 (MetaCIC audit synthesis normalization-blocker monotonicity)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any consumer that strengthens, reuses, or routes the normalization row N through candidate-normalization handoff or residual-diamond handoff must retain the subject-reduction row S and blocker row B through H, R, P, L. Such a consumer may expose candidate-mediated normalization, residual frontier, discharge projection, and ledger data, but it cannot export subject reduction, unrestricted Church–Rosser, candidate-free normalization, quotient equality of MetaCIC terms, or full consistency.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisNormalizationBlockerMonotonicit`

Proof. Start with retained-blocker induction, Definition 776.9. A read that strengthens or reuses N is a finite consumption of one of the positive rows, so S and B already persist through H, R, P, L at the normalization stage.

Now apply candidate-normalization handoff and residual-diamond handoff, Definition 776.17 and Definition 776.20. These theorems allow N to feed candidate-mediated normalization or residual frontier data only as a blocker-preserving route inside the accepted packet.

Finally apply discharge projection, Definition 776.16. The discharge-obligation ledger remains attached to the same subject-reduction boundary, so strengthening the normalization row cannot erase S or B, nor can it repack the route as any excluded MetaCIC endpoint. □

Theorem 776.11 (MetaCIC audit synthesis obligation closure)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

the obligation surface is exactly carrier admission, audit-row exposure for subject-reduction obstruction, confluence, normalization, substitution, decidable checking, closed-normal assembly, persistent blocker rows, componentwise hsame *transport through* H*, displayed* Cont *replay through* R*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *row* L*.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisObligationClosure`

Proof. Begin with the certificate surface of Definition 776.2. It projects the accepted carrier and names $S, C, N, U, D, A, B, H, R, P, L$ as the complete local NameCert surface, so carrier admission and audit-row exposure are already fixed by finite rows.

Use critical-path non-escape from Definition 776.3. Any consumer route remains inside the displayed rows, and no hidden subject-reduction proof, unrestricted Church–Rosser theorem, candidate-free normalization route, global beta-conversion decision, quotient equality, or host completeness coordinate is available.

Next combine the consistency-blocker ledger of Definition 776.4 with the closed-normal boundary of Definition 776.5. These theorems keep the subject-reduction obstruction and blocker rows persistent while confining the positive closed-normal branch to its named finite boundary.

Finally apply route-union exactness, Definition 776.6, and the positive-blocker union theorem, Definition 776.8. They bind the positive rows and blocker rows into one package through the same transport, replay, provenance, and local naming rows. Thus the obligation surface is precisely the stated finite package, with no additional MetaCIC theorem hidden outside it. □

Theorem 776.12 (MetaCIC audit synthesis public export)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

the downstream-facing public packet consists exactly of the subject-reduction obstruction row S*, confluence row* C*, normalization row* N*, substitution row* U*, decidable-checking row* D*, closed-normal assembly row* A*, blocker row* B*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* R*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *row* L*. Exporting this packet preserves the blocker ledger and does not export closed subject reduction, arbitrary Church–Rosser, candidate-free normalization, global beta-conversion decidability, quotient equality of MetaCIC terms, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisPublicExport`

Proof. Begin with the certificate surface of Definition 776.2. It identifies $S, C, N, U, D, A, B, H, R, P, L$ as the complete local packet, so a downstream-facing export has no additional coordinate beyond those rows.

Use critical-path non-escape from Definition 776.3. Every exported consumer route remains inside the same finite row family, and the unavailable MetaCIC claims have no path out of the carrier.

Next combine the consistency-blocker ledger with the closed-normal boundary. Theorems 776.4 and 776.5 keep S and B visible while confining N and A to the named closed-normal branch.

Finally apply route-union exactness and positive-blocker union. Theorems 776.6 and 776.8 bind the positive rows and blocker rows through H, R, P, L, so public export is exactly the stated packet and not any of the excluded full MetaCIC theorems. □

Theorem 776.13 (MetaCIC audit synthesis public transport gate)**.** *For every accepted* MetaCICAuditSynthesisPack *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any public consumer projecting one of the confluence row C, normalization row N, substitution row U, decidable-checking row D, or closed-normal assembly row A must also expose the subject-reduction obstruction row S, blocker row B, componentwise hsame *transport row H, displayed* Cont *replay row R,* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *row L. The gate exports positive audit rows only with those boundary rows visible; it does not export subject reduction, arbitrary Church–Rosser, candidate-free normalization, global beta-conversion decidability, quotient equality of MetaCIC terms, or full consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisPublicTransportGate`

Proof. Start with the public export theorem, Definition 776.12. It identifies the public packet as exactly $S, C, N, U, D, A, B, H, R, P, L$, so a public projection of any positive audit row is still a projection from the same finite carrier.

Apply positive-blocker union from Definition 776.8. The positive rows C, N, U, D, A are exported only together with the subject-reduction obstruction row and blocker row through the displayed structural rows.

Use transport obstruction, Definition 776.15, to keep the closed-normal branch from crossing the subject-reduction boundary as a discharged transport. The obstruction row remains public boundary data rather than a proof of the missing MetaCIC edge.

Finally apply the blocker-preserving positive projection theorem, Definition 776.18. It supplies the consumer-facing projection rule for C, N, U, D, A and preserves S, B, H, R, P, L as visible rows, giving exactly the stated transport gate. □

Theorem 776.14 (MetaCIC audit synthesis scope stability)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

every packet route through $S, C, N, U, D, A, B, H, R, P, L$ is scoped by the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, together with the cited sibling packets chapter 739, chapter 732, and chapter 717. No accepted route can leave this scoped surface to claim subject reduction, arbitrary Church–Rosser, candidate-free normalization, global beta-conversion decidability, full consistency, or quotient equality of MetaCIC terms.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisScopeStability`

Proof. Start with the certificate surface of Definition 776.2. It identifies the finite carrier rows and the local naming row, so every accepted packet route is a projection from one BHist package with componentwise hsame, displayed Cont, Pkg provenance, and local NameCert data.

Apply critical-path non-escape from Definition 776.3. The only rows available to a consumer are $S, C, N, U, D, A, B, H, R, P, L$, and the sibling closed-normal and normal-false packets supply only the displayed closed-normal branch.

Use route-union exactness and positive-blocker union from Definition 776.6 and Definition 776.8. These keep the positive rows and blocker rows in one route union through the same transport, replay, provenance, and naming rows, while the discharge-obligation ledger fixes the subject-reduction boundary as a ledger row rather than a solved transport edge.

Finally apply obligation closure, Definition 776.11. Its exact obligation surface is already the stated finite package, so the route is scope-stable under the named BEDC primitives and sibling packets, with the excluded MetaCIC claims remaining outside the accepted packet. □

Theorem 776.15 (MetaCIC audit synthesis transport obstruction)**.** *For every accepted* MetaCICAuditSynthesisPac *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

the closed-normal branch A and the normal-false exclusion rows cannot be transported across the subject-reduction blocker S, B without first reading the discharge-obligation ledger of chapter 717. Consequently the packet may replay closed-normal evidence only as finite audit data, not as a discharged subject-reduction route.

Proof. Begin with critical-path non-escape from Definition 776.3. Every consumer route stays inside $S, C, N, U, D, A, B, H, R, P, L$, so any transport out of the closed-normal branch must pass through the displayed blocker rows rather than through an external MetaCIC theorem.

Use the consistency blocker ledger and closed-normal boundary next. Definition 776.4 keeps S and B persistent across agreement on the positive rows, while Definition 776.5 confines A and the normal-false exclusion branch to the finite closed-normal endpoint.

Finally apply scope stability from Definition 776.14. The only scoped sibling able to name the missing transport edge is the discharge-obligation ledger, so a route that has not read that ledger cannot carry closed-normal evidence across the subject-reduction blocker. The transport obstruction is therefore part of the packet surface itself. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisTransportObstruction`

Theorem 776.16 (MetaCIC audit synthesis discharge projection)**.** *For every accepted* MetaCICAuditSynthesisPacke *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any synthesis consumer that requests the closed-normal consistency branch through rows N, A must also expose the subject-reduction blocker rows S, B and the discharge-obligation ledger sibling of chapter 717 before it may export that branch. Therefore the route cannot be repacked as subject reduction, full consistency, arbitrary Church–Rosser, candidate-free normalization, global beta-conversion decidability, or quotient equality of MetaCIC terms.

Proof. Start with the certificate surface of Definition 776.2. The rows $S, C, N, U, D, A, B, H, R, P, L$ are one accepted packet, so a consumer that reads N, A is still reading inside the same carrier.

Apply the consistency blocker ledger from Definition 776.4. It keeps S and B visible whenever the positive rows are read, preventing the closed-normal branch from being detached from the subject-reduction boundary.

Use the closed-normal boundary of Definition 776.5. The rows N, A name only the finite closed-normal endpoint and normal-false exclusion surface; they do not discharge transport across S, B.

Finally apply the transport obstruction of Definition 776.15. The discharge-obligation ledger sibling is the named projection through which the missing subject-reduction discharge must remain visible. Hence any export that omits S, B and that ledger leaves the accepted packet surface rather than producing a closed-normal consistency export. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisDischargeProjection`

Theorem 776.17 (MetaCIC audit synthesis candidate-normalization handoff)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any consumer reading candidate-mediated normalization or confluence handoff data through N or C must also carry the subject-reduction blocker rows S, B and the discharge-obligation ledger through H, R, P, L. In particular, the candidate route cannot be repacked as closed subject reduction, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, or quotient equality.

Proof. Begin with the certificate surface of Definition 776.2. It identifies $S, C, N, U, D, A, B, H, R, P, L$ as one finite packet, so a consumer that reads C or N remains inside the same carrier as S and B.

Apply the positive-blocker union theorem, Definition 776.8. Candidate-mediated normalization and confluence are positive-row reads, and the theorem binds such reads to the subject-reduction row and blocker row through the displayed structural rows.

Use transport obstruction and discharge projection next. Theorems 776.15 and 776.16 require the discharge-obligation ledger to remain visible across H, R, P, L whenever the route approaches the closed-normal branch. Therefore the handoff is a blocker-preserving candidate route, not any of the excluded MetaCIC endpoints. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisCandidateNormalizationHandoff`

Theorem 776.18 (MetaCIC audit synthesis blocker-preserving positive projection)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

any consumer that projects a positive row among confluence C, normalization N, substitution U, decidable checking D, or closed-normal assembly A must also expose the subject-reduction row S, blocker row B, the discharge-obligation ledger sibling of chapter 717, componentwise hsame *transport H, displayed* Cont *replay R,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ *row L. Therefore a positive-branch projection cannot export subject reduction, full closed consistency, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, or quotient equality of MetaCIC terms.*

Proof. Start with the certificate surface of Definition 776.2. It identifies $S, C, N, U, D, A, B, H, R, P, L$ as the complete local packet, so a consumer that projects one of C, N, U, D, A is still reading a row of the same carrier.

Apply critical-path non-escape and the consistency-blocker ledger. Theorems 776.3 and 776.4 keep every read inside the displayed audit path and keep S, B visible whenever positive rows are consumed.

Use the positive-blocker union and transport obstruction next. Theorem 776.8 binds positive rows to the blocker rows through H, R, P, L, while Definition 776.15 names the discharge-obligation ledger as the sibling that must remain visible across the missing subject-reduction edge.

Finally apply the discharge projection of Definition 776.16. A consumer that omits S, B or the ledger sibling leaves the accepted packet surface, so the positive branch is preserved only as a blocker-preserving projection and not as any of the excluded MetaCIC exports. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisBlockerPreservingPositiveProjec`

Theorem 776.19 (MetaCIC audit synthesis substitution blocker exhaustion)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

a consumer reading substitution U together with confluence C, normalization N, decidable checking D, or closed-normal assembly A still carries the subject-reduction row S and blocker row B through componentwise transport H, displayed replay R, provenance P, and local naming L. The positive branch cannot be reinterpreted as subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Apply route-union exactness, Definition 776.6. It binds the substitution row and the other positive rows to the same transport, replay, provenance, and naming route as the blocker side.

Use the substitution-confluence join of Definition 776.7. A read that combines U with C, N, D, or A remains a positive-row read inside the same carrier rather than an independent confluence or evaluator-equality theorem.

Now apply the consistency-blocker ledger and blocker-preserving positive projection, Definition 776.4 and Definition 776.18. These keep S and B visible through H, R, P, L whenever a positive row is consumed. Therefore the substitution branch exhausts exactly the displayed blocker-preserving route and none of the excluded MetaCIC endpoints. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisSubstitutionBlockerExhaustion`

Theorem 776.20 (MetaCIC audit synthesis residual-diamond handoff)**.** *Every consumer that reads residual-diamond or parallel-diamond frontier data through an accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *packet*

$$M = (S, C, N, U, D, A, B, H, R, P, L)$$

must expose the rows $S, C, N, U, D, A, B, H, R, P, L$ and the discharge-obligation ledger before treating candidate-normalization or confluence rows as a residual join. The read may use the candidate-normalization row N and confluence row C only as blocker-preserving frontier data; it does not export subject reduction, full consistency, unrestricted Church–Rosser, candidate-free normalization, global beta-conversion decidability, or quotient equality of MetaCIC terms.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisResidualDiamondHandoff`

Proof. Start with the public transport gate, Definition 776.13. A residual-diamond consumer is a public projection from the same finite packet, so it must keep the subject-reduction row S, candidate and confluence rows N, C, blocker row B, and structural rows H, R, P, L visible.

Use blocker-preserving positive projection and candidate-normalization handoff, Definition 776.18 and Definition 776.17. These theorems allow positive frontier rows to be read, but only with the blocker and discharge information still attached.

Finally apply transport obstruction and discharge projection, Definition 776.15 and Definition 776.16. The discharge-obligation ledger remains the named boundary for the missing subject-reduction edge, so a candidate-normalization or confluence row cannot be repacked as a residual join unless the entire blocker-preserving packet is exposed. □

Theorem 776.21 (MetaCIC audit synthesis public frontier export)**.** *Every public frontier consumer that reads candidate normalization, residual-diamond handoff, or closed-normal assembly through an accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *packet*

$$M = (S, C, N, U, D, A, B, H, R, P, L)$$

must keep the subject-reduction row S, blocker row B, discharge-obligation ledger, and structural rows H, R, P, L visible. The export may expose candidate-mediated frontier data from N, residual-diamond data from C, and closed-normal assembly data from A, but it cannot repack those reads as subject reduction, full Church–Rosser, evaluator equality, quotient syntax equality, global conversion decidability, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisPublicFrontierExport`

Proof. Start with the public transport gate, Definition 776.13. It already requires any public positive-row projection to carry S, B, H, R, P, L with the projected row.

Use candidate-normalization handoff and residual-diamond handoff, Definition 776.17 and Definition 776.20. These identify the frontier reads through N and C as blocker-preserving data rather than as a completed confluence or normalization export.

Apply transport obstruction and discharge projection, Definition 776.15 and Definition 776.16. The discharge-obligation ledger remains visible across the missing subject-reduction edge, so the closed-normal assembly row A cannot hide the blocker.

Therefore the public frontier export is exactly a public read of N, C, A with S, B, H, R, P, L and the ledger still displayed. Any request for the excluded MetaCIC endpoints leaves the accepted packet surface. □

Theorem 776.22 (MetaCIC audit synthesis scoped closure)**.** *For every accepted* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *carrier*

$$M = (S, C, N, U, D, A, B, H, R, P, L),$$

the scoped packet surface is exactly the row inventory $S, C, N, U, D, A, B, H, R, P, L$ together with the certificate surface, critical-path non-escape, consistency-blocker ledger, closed-normal boundary, route-union exactness, positive-blocker union, and obligation-closure theorems. It therefore exports no subject reduction, unrestricted Church–Rosser theorem, candidate-free normalization, global beta-conversion decidability, quotient equality of MetaCIC terms, or full closed consistency beyond that finite packet.

Proof. Project the carrier of Definition 776.1. Its rows $S, C, N, U, D, A, B, H, R, P, L$ are the only accepted coordinates, and the certificate surface of Definition 776.2 identifies them as the complete local $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$ surface.

Use the non-escape and blocker theorems next. Definition 776.3 keeps every consumer inside the displayed critical path, while Definition 776.4 keeps the subject-reduction row and blocker row visible whenever the positive rows are read.

Now combine the closed-normal boundary with the route-union results. Definition 776.5 confines the closed-normal branch to the named finite boundary, and Definition 776.6 together with Definition 776.8 binds positive rows and blockers through the same H, R, P, L rows.

Finally apply obligation closure, Definition 776.11. Since all required reads are already finite row projections and all excluded MetaCIC claims have no carrier coordinate, the scoped closure is exactly the stated packet surface. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditSynthesisPacketUp.MetaCICAuditSynthesisScopeClosure`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ packet is generated from subject-reduction obstruction, confluence, normalization, substitution, decidable-checking, closed-normal assembly, blocker, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 776.1, Definition 776.2, Definition 776.3, Definition 776.4, Definition 776.5, Definition 776.6, Definition 776.7, Definition 776.8, Definition 776.11, Definition 776.12, Definition 776.14, Definition 776.16, and Definition 776.18 bind the finite MetaCIC audit synthesis packet and its blocker ledger as one obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, arbitrary closed Church–Rosser, candidate-free normalization, global beta-conversion decidability, full closed consistency, or bridge checking.

Upgrade path: Public closure requires checked export, transport, replay, provenance, and blocker-preservation rows for BEDC.Derived.MetaCICAuditSynthesisPacketUp.

Chapter 777

A Concrete Naming Certificate for $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$

The $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$ packet realizes the normalization audit map of chapter 9383 as a finite seal over candidate-mediated normalization evidence. It sits beside the confluence audit packet of chapter 743, the closed-normal consistency boundary of chapter 733, and the normalization audit packet of chapter 750: those siblings expose confluence, consistency, and normalization-scope rows, while this seal records the finite admission, replay, and non-escape boundary that downstream consistency chapters may cite without reopening the vision layer.

Definition 777.1 (MetaCIC normalization audit seal carrier)**.** *A* $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, R, S, B, F, H, C, P, N)$$

with the following visible rows:

1. *A is the audit-map anchor row naming the finite candidate-mediated normalization scope of chapter 9383;*
2. *R is the reducibility-candidate row exposing only displayed candidate, closed-candidate, Pi-candidate, and candidate-set membership evidence;*
3. *S is the strong-normalization scope row read from R, including the checked closed-term projection and no-infinite-reduction reads available inside the candidate route;*
4. *B is the boundary row separating this seal from unconditional global strong normalization, unrestricted application SN, full closed Church–Rosser, and closed subject-reduction transport;*
5. *F is the finite replay row by which consistency-facing consumers read the same candidate-mediated route without replacing it by prose from the audit map;*
6. *H records componentwise* hsame *comparison across A, R, S, B, F, P, N;*
7. *C records the displayed* Cont *routes for moving from the audit anchor to candidate evidence, from candidate evidence to the SN scope, and from the scope to a consumer-facing seal read;*
8. *P is the* Pkg *provenance over the normalization audit map, the normalization audit packet, the confluence audit sibling, and the closed-normal consistency boundary;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host termination oracle, quotient equality over reductions, evidence-free global normalization, unconditional confluence, or a hidden subject-reduction theorem.

Theorem 777.2 (MetaCIC normalization audit seal obligations)**.** *Every accepted* $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$ *packet binds exactly the finite obligation surface of Definition 777.1: admission of* $Q = (A, R, S, B, F, H, C, P, N)$*, audit-anchor visibility through* A*, candidate evidence visibility through* R*, candidate-mediated strong-normalization visibility through* S*, boundary visibility through* B*, replay through* F *and* C*, componentwise* hsame *transport through* H*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The anchor row A names the normalization audit map, while R and S expose only the candidate evidence and candidate-mediated strong-normalization reads stored in the finite packet.

The boundary row B is part of the same tuple. It prevents a consumer from erasing candidate evidence or rereading the seal as global strong normalization, unrestricted application SN, full closed confluence, or subject-reduction transport.

The structural rows F, H, C, P, N provide replay, componentwise transport, route exposure, provenance, and local naming. Since the carrier has no further coordinate, these rows exhaust the NameCert obligation surface. □

Theorem 777.3 (MetaCIC normalization audit seal non-escape)**.** *Every public read of an accepted* $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$ *packet factors through the finite rows*

$$A, R, S, B, F, H, C, P, N.$$

A consumer may cite the normalization audit map only by reading the displayed anchor, candidate evidence, candidate-mediated SN scope, boundary, replay, transport, route, provenance, and local NameCert rows. The seal exports no host termination oracle, no evidence-free global strong-normalization theorem, no quotient equality over reductions, no unconditional Church–Rosser theorem, and no hidden subject-reduction transport.

Proof. Begin with the carrier definition. Every public read is a projection from $Q = (A, R, S, B, F, H, C, P, N)$ or a route explicitly recorded by C and F.

The positive normalization content is stored only in R and S. Those rows are downstream of the audit anchor A and upstream of the consumer replay row, so the seal can be consumed by closed-normal consistency chapters without leaving the candidate-mediated route.

The non-escape boundary is the coordinate B together with the absence of any excluded coordinate. Transport and provenance preserve the same tuple through H, P, N, and no row names a host oracle, quotient, global normalization theorem, unconditional confluence theorem, or hidden subject-reduction transport. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}$ packet is generated from an audit-map anchor row, reducibility-candidate evidence rows, candidate-mediated strong-normalization rows, boundary rows, finite replay, componentwise transport, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 777.1, Definition 777.2, and Definition 777.3 bind the finite normalization audit seal and its candidate-mediated boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close evidence-free global strong normalization, unrestricted application SN, full closed Church–Rosser, closed subject-reduction transport, quotient equality over reductions, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-mediated SN scope exactness, boundary refusal, route replay, provenance, and non-escape for the same finite seal packet.

Chapter 778

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$

The $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ packet is the shared-modulus normalization surface used by the Real completeness route of chapter 9238. It consumes the shared-bound rows of chapter 462, presents the two-source comparison interface of chapter 411, and feeds the selected-window handoff of chapter 646. The packet is a finite route certificate between those layers; it is not a quotient of regular sequences, a host real equality, a selected limit, or an ambient completeness theorem.

Definition 778.1 (Regular Cauchy modulus normalizer carrier)**.** *A* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *names being routed to a common comparison surface;*
2. *μ_X and μ_Y are the two explicit Cauchy-modulus schedules attached to X and Y;*
3. *M is the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row that displays the common modulus demand used for both schedules;*
4. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger selected after the meet row, with no side-specific hidden window;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read on those shared windows;*
6. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from X, Y, W, D;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after M, W, D, R have been displayed;*
8. *H stores componentwise* hsame *transports for $X, Y, \mu_X, \mu_Y, M, W, D, R, E, P, N$ and the local packet shape;*
9. *C stores the displayed* Cont *routes from the two schedules to M, from M to W, from W to D, from D to R, from R to E, and from those rows to the local certificate;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}}$ *row.*

The carrier has no coordinate for source-equivalence replacement, quotient regular streams, host equality of reals, a selected limiting representative, a side-specific tail oracle, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerTasteGate`
`single carrier alignment`

Definition 778.2 (Regular Cauchy modulus normalizer classifier)**.** *For accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *packets* K *and* K'*, the classifier* $\mathsf{RegularCauchyModulusNormalizerClassifier}(K, K')$ *holds exactly when their displayed normalized routes*

$$\mu_X, \mu_Y, M, W, D, R, E, H, C, P, N$$

transport to the same shared-window comparison surface. The classifier reads only the meet row, shared window ledger, dyadic tolerance ledger, regular readback row, Real seal, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local naming row. It does not classify by first replacing* X *or* Y *with source-equivalent names, and it does not compare host modulus functions outside the packet.*

Lean4 (definition) `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerClassifier`

Theorem 778.3 (Regular Cauchy modulus normalizer NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModu}}$ *surface for Definition 778.1 consists of six finite obligation rows:*

1. *carrier admission for* $K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N)$*;*
2. *source exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *names and their displayed schedules* μ_X, μ_Y*;*
3. *meet exactness requiring every normalized read to pass through the single* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row* M*;*
4. *shared-window exactness requiring* W, D, R *to be read after* M *and to serve both sources simultaneously;*
5. *classifier stability under the componentwise* hsame *transports in* H *and the displayed* Cont *routes in* C*;*
6. *non-escape for source-equivalence transport, quotient streams, host real equality, selected limits, side-specific tail oracles, and ambient completeness.*

Proof. Carrier admission is projection from the displayed packet. The tuple exposes exactly two regular names, two schedules, one meet row, one shared window ledger, one dyadic ledger, one paired readback row, one Real seal, transports, continuations, provenance, and the local name.

The source-exposure row reads X, Y, μ_X, μ_Y before any normalized comparison route is available. The meet-exactness row then projects M as the only common-modulus coordinate, so a consumer cannot replace it by a side-specific threshold or an external schedule comparison.

The shared-window row projects W, D, R downstream of M. These rows are tied to the same window ledger and the same dyadic tolerance ledger, so both regular names are observed on the displayed normalized route.

Finally H, C, P, N repack the projected route as a local NameCert surface. Since the carrier has no coordinate for source-equivalence replacement, quotient streams, host real equality, selected limits, hidden tail oracles, or ambient completeness, the six rows exhaust the public obligation surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier namecert obligations`

Theorem 778.4 (Regular Cauchy modulus normalizer source-exposure obligation)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}}$ *row exposes the four source rows* X, Y, μ_X, μ_Y *before the meet row* M *can be consumed. No comparison-facing consumer may read* $M, W, D, R, E, H, C, P, N$ *as a replacement for this source exposure, and no source-equivalence transport may stand in for either displayed schedule.*

Proof. Project the carrier of Definition 778.1. The displayed packet places the two $\mathsf{RegSeqRat}^{\uparrow}$ names and the two Cauchy-modulus schedules before the meet row, so carrier admission already determines the source surface.

Apply Definition 778.3. Its source-exposure row is separate from meet exactness and shared-window exactness, hence M cannot be read until X, Y, μ_X, μ_Y have been exposed.

Apply Definition 778.7 and Definition 778.8. The common-window route is downstream of M, while the non-escape clause excludes source-equivalence shortcuts and hidden schedules; therefore the four displayed source rows are the only admissible source exposure. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier source exposure obligation`

Theorem 778.5 (Regular Cauchy modulus normalizer shared-window obligation)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the rows W, D, R *are the only shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, and paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows available downstream of* M*. Every comparison, diagonal-compatibility, or Real-seal consumer must read these three rows in that order.*

Proof. Start from the carrier in Definition 778.1. It contains exactly one window ledger W, exactly one dyadic tolerance ledger D, and exactly one paired readback row R after the meet coordinate M.

Apply Definition 778.3. The shared-window exactness row binds W, D, R to the meet row rather than to either source schedule separately, so no side-specific window or hidden tolerance row can be inserted between M and the readback.

Apply Definition 778.7 and Definition 778.8. The common-window theorem fixes the consumer route through M, W, D, R, and non-escape excludes independent tail oracles, quotient streams, and host-side comparisons; hence W, D, R are the only downstream shared-window rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier shared window obligation`

Theorem 778.6 (Regular Cauchy modulus normalizer seal-replay obligation)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the rows E, H, C, P, N *replay the* $\mathsf{Real}^{\uparrow}$ *seal and the local certificate surface after the shared readback. This replay adds no quotient regular stream, no host real equality, no selected limit, no side-specific tail oracle, and no ambient completeness principle.*

Proof. Read the carrier of Definition 778.1 after the paired readback row R. The remaining exported surface is exactly the $\mathsf{Real}^{\uparrow}$ seal E, componentwise transports H, displayed continuations C, provenance P, and local naming row N.

Apply Definition 778.3. Its classifier-stability and non-escape rows make E, H, C, P, N a replay of the displayed certificate surface, not a source of additional analytic data.

Apply Definition 778.7 and Definition 778.8. The shared-window theorem places the replay strictly after M, W, D, R, and the non-escape theorem excludes quotient streams, host real equality, selected limits, side-specific tail oracles, and ambient completeness from the exported packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier seal replay obligation`

Theorem 778.7 (Regular Cauchy modulus normalizer common-window exactness)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

every comparison-facing consumer reads both $\mathsf{RegSeqRat}^{\uparrow}$ *names through the single shared route*

$$\mu_X, \mu_Y, M, W, D, R, H, C, P, N.$$

The common window is therefore determined by the displayed meet row and not by separate source-equivalence transports, independent tail choices, or host-side modulus comparison.

Proof. Project K and isolate the row M. It is the only coordinate whose role is to meet the two displayed schedules μ_X and μ_Y.

Read the window ledger W after M. The packet contains no second window coordinate, so both regular names must use the same displayed $\mathsf{StreamName}^{\uparrow}$ window before the dyadic tolerance ledger is inspected.

Read D and R next. The dyadic tolerance rows and the paired regular readbacks are downstream of the same W, so the comparison-facing route is already normalized before any seal or downstream consumer can inspect it.

The componentwise transports in H and continuations in C preserve that displayed route. Provenance P and the local name N close the route without adding source-equivalence replacement, a side-specific tail oracle, or host-side modulus comparison. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier common window`

Theorem 778.8 (Regular Cauchy modulus normalizer non-escape)**.** *An accepted* $\mathsf{RegularCauchyModulusNormalizer}$ *packet exports only the rows*

$$X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N.$$

Every $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet factors through the displayed normalized shared-window route. The packet exports no quotient regular stream, no source-equivalence shortcut, no host real equality, no selected limiting representative, no independent tail oracle, and no ambient completeness principle.*

Proof. Inspect the carrier of Definition 778.1. Its mathematical rows are exactly the two source names, two schedules, one meet row, one shared window ledger, one dyadic ledger, one paired readback row, and one Real seal.

Use Definition 778.3. The obligation surface forces carrier admission, source exposure, meet exactness, shared-window exactness, and classifier stability to be read from those displayed rows.

Use Definition 778.7. It identifies the consumer route through $\mu_X, \mu_Y, M, W, D, R, H, C, P, N$ before any comparison or diagonal-compatibility handoff can be consumed.

Because the packet contains no coordinate for quotient regular streams, source-equivalence shortcuts, host real equality, selected limiting representatives, independent tail oracles, or ambient completeness, no downstream read can export such data. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier nonescape`

Theorem 778.9 (Regular Cauchy modulus normalizer source-meet-window-seal triad)**.** *For every accepted* RegularCauchyModulusNormalizer$^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

every RegularCauchyComparison$^{\uparrow}$, DiagonalLimitCompatibility$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer that preserves the normalized comparison route reads exactly three ordered surfaces:*

1. *the source-meet exposure* (X, Y, μ_X, μ_Y, M)*;*
2. *the shared-window readback* (W, D, R)*;*
3. *the seal and certificate replay* (E, H, C, P, N)*.*

The triad admits no source-equivalence replacement, side-specific window, hidden dyadic ledger, quotient stream, host real equality, selected limit, independent tail oracle, or ambient completeness principle.

Proof. Use Definition 778.3. Carrier admission exposes K, and the source and meet obligations force the first surface to be exactly (X, Y, μ_X, μ_Y, M).

Use Definition 778.7. The route downstream of M reads one shared window ledger, one dyadic tolerance ledger, and one paired readback row, so the second surface is exactly (W, D, R).

Use Definition 778.8. After the displayed readback, the only remaining exported rows are the Real seal, componentwise transports, continuations, provenance, and local certificate. Hence the third surface is (E, H, C, P, N), and the excluded coordinates cannot be supplied by any downstream consumer. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier source meet window seal triad`

Theorem 778.10 (Regular Cauchy modulus normalizer tail-budget extraction)**.** *For every accepted* RegularCauchyModulusNormalizer$^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

every RegularCauchyTailBudget$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, *or* Real$^{\uparrow}$ *seal consumer of* K *reads its tail-budget route through the single chain*

$$(X, Y, \mu_X, \mu_Y) \longmapsto M \longmapsto (W, D, R) \longmapsto (E, H, C, P, N).$$

The extraction supplies no side-specific tail oracle, quotient stream, host real equality, selected limit, or ambient completeness coordinate.

Proof. Project the carrier from Definition 778.1. The only source rows are X, Y, μ_X, μ_Y, the only common tail-meet row is M, and the only downstream shared-window, readback, seal, transport, replay, provenance, and naming rows are W, D, R, E, H, C, P, N.

Apply Definition 778.3. The source-exposure and shared-window exactness obligations force every accepted consumer to enter through the displayed source schedules and then through the common window rows rather than through a side-specific oracle.

Use Definition 778.7, Definition 778.8, and Definition 778.9. Together they identify the ordered source-meet, shared-window, and seal surfaces and exclude any quotient stream, host real equality, selected limit, or ambient completeness coordinate from the exported route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier tail budget extraction`

Theorem 778.11 (Regular Cauchy modulus normalizer cofinal-filter handoff)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the normalized route through M, W, D, R supplies exactly the finite cofinal-tail filter surface consumed by $\mathsf{FiniteTailFilter}^{\uparrow}$*. Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer must pass through the shared window row W and dyadic tolerance row D before reading the Real seal E, and the handoff admits no side-specific tail oracle, quotient stream, selected limit, host real equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier cofinal filter handoff`

Proof. Begin with the source-exposure and shared-window rows in Definition 778.3. They force each source schedule X, Y, μ_X, μ_Y to enter the common meet row M before any downstream window can be read.

Use Definition 778.7. The route downstream of M is the shared finite-window row W, the dyadic tolerance row D, and the paired readback row R, with no side-specific replacement.

Apply Definition 778.10 and Definition 778.8. The tail-budget route reaches the seal only through (W, D, R) and then E, H, C, P, N, while quotient streams, selected limits, host real equality, and ambient completeness are excluded.

Finally read chapter 1119 as the consumer surface. Its finite cofinal-tail filter consumes window, dyadic tolerance, regular readback, tail-budget comparison, Real seal, transport, continuation, provenance, and local naming rows; the normalized M, W, D, R route supplies exactly that finite surface before any Real or completion consumer can cite the seal. □

Theorem 778.12 (Regular Cauchy modulus normalizer public route)**.** *For every accepted* RegularCauchyModulusN *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

every comparison-facing, diagonal-compatibility, or $\mathsf{Real}^{\uparrow}$*-seal consumer that preserves the normalized shared-window route factors through the public route*

$$(X, Y, \mu_X, \mu_Y, M) \longmapsto (W, D, R) \longmapsto (E, H, C, P, N).$$

The consumer may read these rows only through the displayed BHist *packet, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. No consumer of this public route obtains a source-equivalence replacement, side-specific window, hidden dyadic ledger, quotient stream, host real equality, selected limit, independent tail oracle, or ambient completeness principle.*

Proof. Apply Definition 778.3. Carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape give the finite obligation surface read by any consumer of K.

Use Definition 778.7 and Definition 778.9. The first theorem forces the normalized read through the common route downstream of M; the second packages the same read as the ordered source-meet, shared-window, and seal-certificate surfaces.

Next use Definition 778.8. The carrier has no coordinates for source-equivalence replacement, side-specific windows, hidden dyadic ledgers, quotient streams, host real equality, selected limits, independent tail oracles, or ambient completeness, so none of those objects can be projected by a downstream consumer.

Finally apply Definition 778.17. It binds the route to the displayed BHist, hsame, Cont, Pkg, and NameCert rows, so every comparison-facing, diagonal-compatibility, or $\mathsf{Real}^{\uparrow}$-seal consumer preserving the route factors through the public composite displayed above. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier public route`

Theorem 778.13 (Regular Cauchy modulus normalizer carrier admission)**.** *For* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *the accepted packets are exactly the displayed carriers*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N).$$

The coordinates record the two $\mathsf{RegSeqRat}^{\uparrow}$ *sources, their schedules, the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row, shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, paired readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row, with no admitted coordinate for quotient streams, host equality, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier carrier admission`

Proof. Project the carrier definition. It lists precisely $X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N$ and no further coordinate.

Apply the NameCert obligations. Carrier admission, source exposure, shared window exactness, seal replay, classifier stability, and non-escape are all read from the displayed tuple and the BEDC primitives named in it.

Finally use the common-window, triad, and public-route theorems. They exhaust the route through the meet, shared window, dyadic ledger, readback, seal, and public replay rows, so quotient streams, host equality, selected limits, and ambient completeness are not accepted packet data. □

Theorem 778.14 (Regular Cauchy modulus normalizer comparison stability)**.** *For accepted* RegularCauchyModulus *packets* $K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N)$ *and* $K' = (X', Y', \mu'_X, \mu'_Y, M', W', D', R', E', H', C', P', N')$, *any displayed componentwise* hsame *and* Cont *replay preserving the carrier rows preserves the meet, shared-window, dyadic-readback, and Real-seal route*

$$(X, Y, \mu_X, \mu_Y, M) \longmapsto (W, D, R) \longmapsto (E, H, C, P, N).$$

Thus the classifier read is stable under displayed comparison replay and cannot be replaced by a quotient stream, host equality, selected limit, side-specific tail oracle, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier comparison stability`

Proof. Start with carrier admission and the classifier definition. The classifier can read only the coordinates displayed in the accepted tuple, so a componentwise replay preserving those rows preserves the classifier's source surface.

Use the common-window exactness and source-meet-window-seal triad. These theorems identify the comparison route as the meet row followed by the shared window, dyadic ledger, paired readback, and Real seal.

Finally apply non-escape and the public route. They keep comparison-facing, diagonal-compatibility, and $\mathsf{Real}^{\uparrow}$ consumers on the same route under hsame and Cont, and they exclude quotient streams, host equality, selected limits, side-specific tail oracles, and ambient completeness as route substitutes. □

Theorem 778.15 (Regular Cauchy modulus normalizer source-exchange stability)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the exchanged packet

$$K^{\mathrm{op}} = (Y, X, \mu_Y, \mu_X, M, W, D, R, E, H, C, P, N)$$

preserves the displayed $\mathsf{RegularCauchyModulusNormalizerClassifier}$ *route. Both displays expose the same source-meet row* M*, shared-window row* W*, dyadic ledger* D*, paired readback* R*, seal* E*, transports* H*, continuations* C*, provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. Thus source exchange changes only the order in which the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources and their schedules are displayed; it does not change the shared-window comparison surface, the seal route, or the downstream NameCert surface.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier source exchange stability`

Proof. Project the carrier rows from Definition 778.1. The displayed tuple separates the two source coordinates and schedules from the meet, window, ledger, readback, seal, transport, continuation, provenance, and name rows. Passing from K to K^{op} exchanges only X, Y, μ_X, μ_Y.

Use Definition 778.2 and Definition 778.3. The classifier reads the normalized route through the displayed meet and shared-window rows, while the NameCert obligations expose carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape on the same finite surface.

Now apply Definition 778.7. Since $M, W, D, R, E, H, C, P, N$ are unchanged, both source orders route through the same common window and the same seal replay.

Finally apply Definition 778.8. It rules out source-equivalence replacement, quotient streams, host real equality, selected limits, side-specific tail oracles, and ambient completeness as substitutes for the displayed exchange route. □

Theorem 778.16 (Regular Cauchy modulus normalizer witness-ledger handoff)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

any downstream comparison, diagonal-compatibility, or $\mathsf{Real}^{\uparrow}$ *consumer reads the witness ledger through*

$$M, W, D, R, E, H, C, P, N$$

after the two source schedules have been admitted. The handoff cannot pass through quotient streams, host equality, selected limits, independent tail oracles, side-specific completion witnesses, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier witness ledger handoff`

Proof. Start with carrier admission and the NameCert obligations. They display the two source rows, two schedules, meet row, common window, dyadic ledger, paired readback, Real seal, transport, continuation, provenance, and local naming as the whole accepted packet.

Next use common-window exactness and comparison stability. The comparison read after the sources is fixed by $M, W, D, R, E, H, C, P, N$, and displayed replay through hsame and Cont preserves that route for comparison and diagonal consumers.

Finally apply non-escape, the triad theorem, and the public route. These results exhaust the witness-ledger handoff before any downstream consumer can read it, so quotient streams, host equality, selected limits, independent tail oracles, side-specific completion witnesses, and ambient completeness have no handoff coordinate. □

Theorem 778.17 (Regular Cauchy modulus normalizer scoped primitive binding)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

each obligation row of Definition 778.3 is bound to the displayed BEDC primitives of the packet: source rows through $\mathsf{RegSeqRat}^{\uparrow}$*, the meet row through* $\mathsf{CauchyModulusMeet}^{\uparrow}$*, the shared window through* $\mathsf{StreamName}^{\uparrow}$*, the tolerance ledger through* $\mathsf{DyadicRatCore}^{\uparrow}$*, the seal row through* $\mathsf{Real}^{\uparrow}$*, transport through* hsame*, route preservation through* Cont*, provenance through* Pkg*, and local admission through* NameCert*. The binding is internal to the displayed carrier and obtains no obligation row from host equality, quotient streams, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier scoped primitive binding`

Proof. Project the carrier of Definition 778.1. The packet exposes the two $\mathsf{RegSeqRat}^{\uparrow}$ source names, the two displayed schedules, the $\mathsf{CauchyModulusMeet}^{\uparrow}$ row M, the $\mathsf{StreamName}^{\uparrow}$ ledger W, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, the paired $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{Real}^{\uparrow}$ seal E as named coordinates.

Apply Definition 778.3. Carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape are therefore read from those projected coordinates together with the componentwise hsame transports in H, the displayed Cont routes in C, the Pkg provenance in P, and the local NameCert row N.

Use Definition 778.7. The common-window route fixes the obligation path through $\mu_X, \mu_Y, M, W, D, R, H, C, P$, so the meet, window, tolerance, readback, transport, route, provenance, and local admission rows remain scoped to the displayed packet.

Use Definition 778.8. Since the packet has no coordinate for host equality, quotient streams, selected limits, side-specific tail oracles, or ambient completeness, none of those structures can source an obligation row or replace the primitive bindings just projected. □

Theorem 778.18 (Regular Cauchy modulus normalizer normalized route bridge)**.** *For every accepted* RegularCauchyModulusNormalizer$^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the public route

$$(X, Y, \mu_X, \mu_Y, M) \mapsto (W, D, R) \mapsto (E, H, C, P, N)$$

bridges to the completion-facing shared-window comparison interface through the single normalized route

$$\mu_X, \mu_Y, M, W, D, R, E, H, C, P, N.$$

The bridge reads only the displayed BHist *packet, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. It exports no quotient stream, host real equality, selected limit, side-specific tail oracle, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerUp StdBridge`

Proof. Start with the local NameCert obligations. Carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape from Definition 778.3 display the whole finite packet and keep every public read inside it.

Use the source-exposure, shared-window, and seal-replay obligations next. Definition 778.4 places the two schedules before the meet row, Definition 778.5 forces W, D, R to serve both sources after M, and Definition 778.6 replays the Real$^{\uparrow}$ seal through E, H, C, P, N.

Now apply common-window exactness and non-escape. Definition 778.7 identifies the normalized public route through $\mu_X, \mu_Y, M, W, D, R, H, C, P, N$, while Definition 778.8 removes source-equivalence shortcuts, quotient streams, host real equality, selected limits, side-specific tail oracles, and ambient completeness.

Finally use the public route of Definition 778.12. The completion-facing shared-window comparison interface receives exactly the displayed normalized route, so it is the bridge from the two source schedules to that public consumer surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier normalized route bridge`

Theorem 778.19 (Regular Cauchy modulus normalizer bridge obstruction boundary)**.** *For every accepted* RegularCauchyModulusNormalizer$^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

any bridge-facing request that preserves the public normalized route as a same-surface bridge must expose the same rows

$$\mu_X, \mu_Y, M, W, D, R, E, H, C, P, N.$$

It must also preserve the primitive binding through the displayed BHist *packet, componentwise* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. A request that omits the meet row M, shared window W, dyadic tolerance D, paired readback R,* Real$^{\uparrow}$ *seal E, route replay through C, or provenance through P, N is outside this chapter's bridge surface rather than an obstruction to the public route.*

Proof. Start from the local NameCert obligations. Definition 778.3 reads carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape from the coordinates of the displayed packet.

The source-facing and seal-facing obligations fix the rows that a bridge request must keep visible. Use Definition 778.4, Definition 778.5, and Definition 778.6 to expose the two schedules, the meet row, the common window, the dyadic tolerance, the paired readback, the $\mathsf{Real}^{\uparrow}$ seal, the route replay, and the provenance rows.

Common-window exactness and non-escape supply the obstruction boundary. Definition 778.7 keeps the normalized route on $\mu_X, \mu_Y, M, W, D, R, H, C, P, N$, while Definition 778.8 prevents quotient streams, host real equality, selected limits, side-specific tail oracles, and ambient completeness from replacing those coordinates.

Finally apply the scoped primitive binding, public route, and normalized route bridge. Definition 778.17 binds the same BHist, hsame, Cont, Pkg, and NameCert rows; Definition 778.12 identifies the public consumer route; and Definition 778.18 reads the completion-facing bridge from that same surface. Any request missing one of the displayed rows therefore leaves the bridge surface instead of producing an internal obstruction to the public normalized route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier bridge obstruction boundary`

Theorem 778.20 (Regular Cauchy modulus normalizer mature witness-ledger handoff)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

any $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *consumer, or any* $\mathsf{Real}^{\uparrow}$*-completion consumer, that preserves the normalized public route must consume the same meet row* M*, shared window* W*, dyadic tolerance* D*, paired readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise transport* H*, continuation route* C*, provenance row* P*, and local* NameCert *row* N*. Such a consumer cannot replace this route by side-specific schedules, a quotient stream, host real equality, selected limit, independent tail oracle, or ambient completeness.*

Proof. Start with the local obligation surface. Definition 778.3 exposes carrier admission, source exposure, meet exactness, shared-window exactness, classifier stability, and non-escape from the displayed packet K.

Next read the route through the common-window and public-route theorems. Definition 778.7 fixes the shared comparison through $\mu_X, \mu_Y, M, W, D, R, H, C, P, N$, and Definition 778.12 presents that comparison as the public consumer surface downstream of the source schedules.

The bridge result then identifies the handoff surface. By Definition 778.18, a completion-facing consumer receives the normalized route

$$\mu_X, \mu_Y, M, W, D, R, E, H, C, P, N.$$

Finally apply Definition 778.19. A request that drops the meet, window, tolerance, readback, seal, transport, continuation, provenance, or local NameCert row leaves the bridge surface; a request that substitutes side-specific schedules, quotient streams, host real equality, selected limits, independent tail oracles, or ambient completeness is therefore not the same normalized handoff. The witness-ledger consumer consequently reads exactly the displayed normalized surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier mature witness ledger handoff`

Theorem 778.21 (Regular Cauchy modulus normalizer formal target surface)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

the compact formal target surface is exactly the source-meet row (X, Y, μ_X, μ_Y, M)*, the shared dyadic window* (W, D)*, the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *seal replay* R*, the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row* M*, the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *comparison row exposed by the normalized route, the witness-ledger handoff row, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows* H, C, P, N*. The surface has no selected limit, quotient stream, host equality, ambient completeness row, side-specific tail oracle, or source-equivalence replacement.*

Proof. Start with the carrier, classifier, and local NameCert obligations. They expose the two source names, the two schedules, the meet row, the shared window, dyadic tolerance, paired readback, Real seal, transport, continuation, provenance, and local naming rows as the only available packet coordinates.

Next use source exposure, shared-window exactness, seal replay, and common-window exactness. These theorems order the route through (X, Y, μ_X, μ_Y, M), then through (W, D, R), and then through (E, H, C, P, N), with the $\mathsf{CauchyModulusMeet}^{\uparrow}$ and $\mathsf{RegularCauchyComparison}^{\uparrow}$ readings confined to that displayed route.

Now apply non-escape and the bridge obstruction boundary. They exclude selected limits, quotient streams, host real equality, side-specific tail oracles, source-equivalence replacement, and ambient completeness from the target surface.

Finally use the mature witness-ledger handoff. It identifies the $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ and completion-facing consumer rows with the same meet, shared-window, dyadic, readback, seal, transport, continuation, provenance, and local NameCert surface. Thus the listed rows are precisely the finite formal target surface available for Lean formalization. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier formal target surface`

Theorem 778.22 (Regular Cauchy modulus normalizer budget pullback exhaustion)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

any Phase Real and Completion consumer that pulls a budget read or witness-ledger read back through $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *factors through the same four faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

and through the displayed support rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The pullback has no side-specific tail oracle, quotient stream, host real equality, selected limit, ambient completeness, countable-choice row, metric-completeness row, or bridge row.*

Proof. Start with the public route and normalized bridge. By Definition 778.12, every comparison-facing or completion-facing consumer preserving the route reads

$$(X, Y, \mu_X, \mu_Y, M) \longmapsto (W, D, R) \longmapsto (E, H, C, P, N).$$

Definition 778.18 then identifies this public route with the single normalized surface $\mu_X, \mu_Y, M, W, D, R, E, H, C, P,$

Next apply the compact formal surface of Definition 778.21. It keeps every budget read and witness-ledger handoff on the displayed meet, shared-window, dyadic, readback, seal, transport, continuation, provenance, and local NameCert rows.

Finally invoke the StreamName exit boundary, Definition 126.500, together with terminal exactness, Definition 126.505. These read the Phase Real and Completion exit packet as exactly the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with only the displayed BHist, hsame, Cont, Pkg, and NameCert support rows. Therefore a budget or witness-ledger pullback through $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ terminates at that four-face packet and cannot acquire any side-specific tail oracle, quotient stream, host equality, selected limit, completeness row, choice row, metric row, or bridge row. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier budget pullback exhaustion`

Theorem 778.23 (Regular Cauchy modulus normalizer shared-tail meet determinacy)**.** *For every accepted* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *carrier*

$$K = (X, Y, \mu_X, \mu_Y, M, W, D, R, E, H, C, P, N),$$

every $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{CauchySharedTailModulus}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer route that uses the displayed meet row* M *reads the same shared tail window* (W, D, R) *before the Real seal* E*. If two such consumer routes start from the packet* K *and pass through* M*, then their downstream classifications agree by the componentwise* hsame *rows recorded in* H*, and neither route can introduce a side-specific tail oracle, hidden modulus selector, quotient regular stream, or host real equality.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier shared tail meet determinacy`

Proof. Start with the shared-window obligation and common-window exactness. By Definition 778.5 and Definition 778.7, every consumer that reads the displayed meet row M is already routed through the single packet projection

$$(X, Y, \mu_X, \mu_Y, M) \longmapsto (W, D, R) \longmapsto (E, H, C, P, N).$$

Thus the tail window is not an extra choice made by the consumer; it is the displayed W, D, R coordinate triple of K.

Next apply tail-budget extraction and the witness-ledger handoff. The former keeps every tail-budget read on the shared dyadic window and paired $\mathsf{RegSeqRat}^{\uparrow}$ readback, while the latter carries the same rows into the completion-facing witness ledger. Hence $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers all receive the same tail data whenever they use M.

Finally use budget-pullback exhaustion. It confines every downstream budget or witness-ledger pullback to the displayed support rows BHist, hsame, Cont, Pkg, and NameCert, with the componentwise hsame rows in H providing the common classifier comparison. Since the only available coordinates are the unique $M, W, D, R, E, H, C, P, N$ entries of K, two routes through M classify together and cannot acquire a side-specific tail oracle, hidden modulus selector, quotient regular stream, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, two displayed Cauchy-modulus schedules, a shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ row, one shared $\mathsf{StreamName}^{\uparrow}$ window ledger, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, paired $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 778.1, Definition 778.2, Definition 778.3, Definition 778.4, Definition 778.5, Definition 778.6, Definition 778.7, Definition 778.8, Definition 778.9, Definition 778.12, Definition 778.13, Definition 778.14, Definition 778.15, Definition 778.16, Definition 778.17, Definition 778.18, Definition 778.20, Definition 778.21, Definition 778.22, and Definition 778.23 explicitly name the carrier, classifier, source exposure, meet exactness, shared-window exactness, seal replay, classifier stability, non-escape obligation rows, primitive binding, public consumer route, carrier admission, comparison stability, source-exchange stability, witness-ledger handoff, normalized-route bridge, mature witness-ledger handoff, compact formal target surface, budget pullback exhaustion, and shared-tail meet determinacy.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RegularCauchyModulusNormalizerUp.RegularCauchyModulusNormalizerCarrier\_budget\_pullbac`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient regular streams, source-equivalence shortcuts, host real equality, selected limits, side-specific tail oracles, or ambient completeness for $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$.

Upgrade path: No further formal-axis upgrade remains for the displayed bridge over the BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 779

A Concrete Naming Certificate for $\mathsf{SubstitutionAuditMap}^{\uparrow}$

The $\mathsf{SubstitutionAuditMap}^{\uparrow}$ packet realizes the MetaCIC substitution audit map of chapter 9380 as finite NameCert data. It sits beside the closed-substitution boundary of chapter 538 and the closed-term substitution boundary of chapter 527: those siblings name the inert boundary rows, while this packet records the audit map through which later closed-normal and subject-reduction consumers inspect those rows without acquiring an unrestricted substitution calculus.

The generator-closure classifier row is read through the concrete $\mathsf{GeneratorClosure}^{\uparrow}$ certificate of chapter 486 before it is specialized to closed shift, closed substitution, and closed-composition audit rows.

Definition 779.1 (Substitution audit map carrier)**.** *A* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, C, R, S, K, G, H, Q, P, N)$$

with the following visible rows:

1. *T is the displayed MetaCIC* Term *row being audited;*
2. *C is the* $\mathsf{ClosedAt}$ *boundary row attached to T;*
3. *R is the shift row recording the closed-boundary read of* shift*;*
4. *S is the substitute row recording the closed-boundary read of* $\mathsf{substitute}$*;*
5. *K is the closed-composition row for the available closed substitution compositions;*
6. *G is the generator-closure classifier row for the shift and substitution reads;*
7. *H records componentwise* hsame *transports for term, closedness, shift, substitution, composition, classifier, continuation, provenance, and naming rows;*
8. *Q records the displayed* Cont *routes through which consumers inspect the audit rows;*
9. *P is the* Pkg *provenance over* Term, $\mathsf{ClosedAt}$, shift, $\mathsf{substitute}$, *the closed-composition rows, the generator-closure classifier rows,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstitutionAuditMap}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are the closed substitution audit rows listed above. It exports no beta-substitution preservation theorem, no closed subject-reduction theorem,

no unclosed substitution commutation theorem, no quotient equality, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapTasteGate single carrier alignment`

Theorem 779.2 (Substitution audit map NameCert obligations)**.** *The carrier of Definition 779.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{SubstitutionAuditMap}^{\uparrow}}$ *obligation surface. Its obligations are carrier admission for* $A = (T, C, R, S, K, G, H, Q, P, N)$*, term and* ClosedAt *readback through* T, C*, shift readback through* R*, substitution readback through* S*, closed-composition readback through* K*, generator-closure classification through* G*, componentwise transport through* H*, displayed continuation through* Q*, provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is the displayed finite BHist packet. Projecting T and C exposes the audited MetaCIC term together with its closed-boundary row, matching the boundary surface described in chapter 9380.

Project R, S, K, G next. These rows record the checked closed shift, closed substitution, closed-composition, and generator-closure classifier surfaces used by chapter 538; none of them contains a typed preservation endpoint.

Finally inspect H, Q, P, N. The transport, continuation, provenance, and naming rows replay the same finite packet for consumers. Since every obligation is one of the displayed rows and no hidden row is added by transport or provenance, the local NameCert surface is exhausted by the carrier inventory. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier namecert obligations`

Theorem 779.3 (Substitution audit map boundary non-escape)**.** *For an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$*, every consumer read remains inside the displayed* Term*,* ClosedAt*, shift, substitute, closed-composition, generator-closure classifier,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. In particular, the packet cannot export an unrestricted substitution calculus, beta-substitution preservation, or closed subject reduction.*

Proof. Read the carrier by projection. The rows T, C, R, S, K, G expose closed syntactic movement and classifier data only under the displayed closed-boundary hypotheses.

The remaining rows H, Q, P, N transport, route, locate, and name that same packet. They preserve the audit surface but do not add a coordinate for beta substitution preservation or subject reduction.

Therefore any consumer of the accepted packet can cite only the closed substitution audit rows and their displayed routes. The non-escape exclusions are exactly the missing coordinates recorded in the carrier definition. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier boundary nonescape`

Theorem 779.4 (Substitution audit map positive row coverage)**.** *For an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$*, every positive audit row read by the packet is exhausted by the displayed term row* T*, closed-boundary row* C*, shift row* R*, substitute row* S*, closed-composition row* K*, and generator-closure row* G*. A consumer that asks for a substitution-audit witness therefore receives only these six source rows together with their transport, route, provenance, and local naming coordinates* H, Q, P, N*.*

Proof. Project the accepted carrier. The rows T, C, R, S, K, G are the only coordinates that carry positive MetaCIC audit data: term exposure, closed-boundary exposure, shift readback, substitute readback, closed-composition readback, and generator-closure classification.

The remaining coordinates H, Q, P, N do not add another audit source. They identify componentwise transport, displayed consumer routes, package provenance, and the local certificate name for the same finite packet.

Thus every positive read factors through the listed rows. Since the carrier has no further source coordinate, the positive audit surface is exhausted by T, C, R, S, K, G. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier positive row coverage`

Theorem 779.5 (Substitution audit map obstruction row exactness)**.** *For an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier, the obstruction boundary is exact: beta-substitution preservation, closed subject reduction, unclosed substitution, quotient equality, and host semantic equality are excluded precisely because none of them is a row of* $A = (T, C, R, S, K, G, H, Q, P, N)$ *and none is created by* H, Q, P, N*.*

Proof. Read the positive rows first. The coordinates T, C, R, S, K, G name only the displayed MetaCIC term, closed boundary, closed shift, closed substitute, closed-composition, and generator-closure classifier data.

Now inspect the administrative rows. Componentwise hsame transport preserves the visible coordinates, Cont routes replay them, Pkg records provenance over them, and the local $\mathsf{NameCert}$ row names them.

It follows that no administrative row can supply beta preservation, subject reduction, unclosed substitution, quotient equality, or host semantic equality. These exclusions are therefore not loose warnings; they are exactly the missing carrier coordinates. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier obstruction row exactness`

Theorem 779.6 (Substitution audit map frontier route non-escape)**.** *Every frontier-facing consumer route for an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier factors through* H, Q, P, N *after reading the positive audit rows* T, C, R, S, K, G*. Such a route can transport, replay, locate, and name the closed substitution audit packet, but it cannot acquire unrestricted substitution or any theorem absent from the displayed carrier.*

Proof. Start with positive row coverage. Any consumer route that reaches substitution-audit data has already passed through the finite rows T, C, R, S, K, G.

The frontier rows are H, Q, P, N. They provide transport, continuation replay, provenance, and naming for the already displayed packet, so the consumer can carry the packet across a boundary without changing its row inventory.

Apply obstruction exactness to the transported packet. Since the excluded rows are absent before the frontier route and are not created by transport, replay, provenance, or naming, unrestricted substitution still cannot escape the audit map. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier frontier route nonescape`

Theorem 779.7 (Substitution audit map obligation closure package)**.** *For the finite packet* $A = (T, C, R, S, K, G, H, Q, P, N)$ *of Definition 779.1, the carrier definition, Definition 779.2, Definition 779.3, Definition 779.4, Definition 779.5, and Definition 779.6 jointly exhaust the* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *obligation surface. The package records exactly the closed term, closed-boundary, shift, substitute, closed-composition, generator-closure, transport, continuation, provenance, and local naming rows admitted by the TasteGate carrier alignment.*

Proof. The carrier definition supplies the row inventory, and the checked TasteGate alignment following Definition 779.1 fixes that inventory as the single finite carrier surface.

Use the NameCert obligation theorem to read carrier admission, closed movement readback, closed-composition readback, classifier readback, transport, continuation, provenance, and naming from $T, C, R, S, K, G, H, Q, P, N$.

The boundary non-escape, positive row coverage, obstruction row exactness, and frontier route non-escape theorems account for every remaining consumer question: which positive rows are available, which obstruction rows are absent, and how frontier reads remain inside the same packet. Therefore no further obligation is left outside the displayed finite surface. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier obligation closure package`

Theorem 779.8 (Substitution audit map domain-codomain exposure)**.** *For the finite packet $A = (T, C, R, S, K, G, H, Q, P, N)$ of Definition 779.1, the only domain rows are the displayed term row T and the closed-boundary row C, and the only movement rows are the shift row R, substitute row S, closed-composition row K, and generator-closure classifier row G. No unrestricted substitution row, beta preservation row, quotient equality row, or host semantic equality row is present in the carrier.*

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier domain codomain exposure`

Proof. Project the carrier displayed in Definition 779.1. The coordinates T and C are the only rows that expose the audited MetaCIC term and its closed-boundary domain.

The movement surface is then read through R, S, K, G. By Definition 779.2, those rows are exactly the shift readback, substitution readback, closed-composition readback, and generator-closure classifier obligations.

Finally apply Definition 779.5. The rows for unrestricted substitution, beta preservation, quotient equality, and host semantic equality are absent from the packet and cannot be supplied by the administrative rows H, Q, P, N. □

Theorem 779.9 (Substitution audit map replay determinacy)**.** *For an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier $A = (T, C, R, S, K, G, H, Q, P, N)$, any transported consumer read of the audit map must replay the same displayed rows through the componentwise transport row H and the continuation route row Q before reaching the provenance row P and the local naming row N.*

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier replay determinacy`

Proof. Begin with Definition 779.4. Any positive consumer read is already confined to the displayed rows T, C, R, S, K, G.

Now use Definition 779.6. A frontier-facing route reaches the administrative surface only by transporting the displayed packet through H and replaying it through Q.

The provenance and naming rows P, N therefore receive the same packet after transport and route replay. Since no route can escape the displayed carrier, the consumer read is determined by the original row inventory. □

Theorem 779.10 (Substitution audit map ledger non-escape)**.** *For an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier $A = (T, C, R, S, K, G, H, Q, P, N)$, the provenance row P and the local* NameCert *row N can name only the closed substitution audit packet T, C, R, S, K, G, H, Q. They cannot promote obstruction rows absent from A into public theorems.*

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier ledger nonescape`

Proof. Read the ledger surface from the obligation package of Definition 779.7. The provenance row and local name are attached to the finite packet already displayed by T, C, R, S, K, G, H, Q.

Apply Definition 779.5. The obstruction rows are exactly absent from the accepted carrier, and the administrative coordinates do not create new theorem rows.

Thus P and N can record and name only the closed substitution audit packet. A public theorem about unrestricted substitution, beta preservation, quotient equality, or host semantic equality would require a row that the carrier does not contain. □

Theorem 779.11 (Substitution audit map closed-boundary scope)**.** *For every accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$*, the closed-term substitution boundary and the closed substitution boundary are read only through the displayed term row* T*, closed-boundary row* C*, shift row* R*, substitute row* S*, closed-composition row* K*, and generator-closure classifier row* G*, with transport, route replay, provenance, and naming supplied by* H, Q, P, N*.*

The checked TasteGate carrier alignment keeps this boundary as the same finite packet. It does not export unrestricted substitution, beta preservation, closed subject reduction, quotient equality, or host semantic equality.

Proof. Project the accepted carrier. The term and closedness data are the rows T and C, and the only closed movement rows are R, S, K.

Use positive row coverage and obstruction row exactness. The generator-closure classifier G accounts for the closed-boundary route, while the excluded substitution and equality principles remain absent carrier coordinates.

Finally apply frontier route non-escape. The rows H, Q, P, N transport, replay, locate, and name the same closed-boundary packet, so the scope remains the displayed finite boundary. □

Theorem 779.12 (Substitution audit map generator-route scope)**.** *For every accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$*, the generator-closure classifier row* G *is the only generator-route source for closed shift, closed substitution, and closed-composition reads. Any consumer route through* H, Q, P, N *must replay* G *together with* T, C, R, S, K *and cannot replace it by an unrestricted substitution generator.*

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier generator route scope`

Proof. Start with the domain-codomain exposure theorem. It identifies T, C as the primitive domain rows and R, S, K, G as the movement and generator-route rows.

Next use replay determinacy. Transported reads must pass through H and Q before reaching P and N, so the generator classifier is replayed as part of the same packet.

Obstruction row exactness excludes an unrestricted substitution generator. Since no such row is present before or after replay, the generator route is scoped by G. □

Theorem 779.13 (Substitution audit map consumer obstruction scope)**.** *For every accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$*, every downstream consumer obstruction is exact at the displayed packet boundary: beta-substitution preservation, subject reduction, quotient equality, and host semantic equality are unavailable precisely because no row of* A *supplies them and* H, Q, P, N *do not create them.*

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier consumer obstruction scope`

Proof. Read the positive rows through Definition 779.4. They exhaust the available term, closed-boundary, movement, and generator-classifier data.

Apply obstruction row exactness. The absent obstruction rows are not hidden assumptions; they are missing carrier fields.

Use ledger non-escape for the consumer-facing side. The provenance and local naming rows can name only the closed substitution audit packet, so an excluded theorem cannot be promoted into a public consumer row. □

Theorem 779.14 (Substitution audit map scoped closure package)**.** *The carrier $A = (T, C, R, S, K, G, H, Q, P, N)$ the checked TasteGate carrier alignment, positive row coverage, obstruction row exactness, frontier route non-escape, closed-boundary scope, generator-route scope, and consumer obstruction scope jointly bind* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *as a scoped finite audit package for closed-term substitution boundaries.*

The package closes only the displayed closed substitution audit map over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not close unrestricted substitution, beta preservation, subject reduction, quotient equality, or host semantic equality.*

Proof. Begin with the obligation closure package. It already assembles carrier admission, local NameCert obligations, positive row coverage, obstruction exactness, and frontier non-escape over the displayed finite rows.

Attach the closed-boundary and generator-route scope theorems. These identify the closed-term boundary, closed substitution boundary, and generator-closure classifier as rows of the same packet rather than as external calculus assumptions.

Finally attach consumer obstruction scope. Since the excluded principles remain absent under transport, replay, provenance, and naming, the finite packet is scoped-closed at exactly the displayed audit boundary. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier scoped closure package`

Theorem 779.15 (Substitution audit map public export)**.** *The finite carrier $A = (T, C, R, S, K, G, H, Q, P, N)$, the checked TasteGate carrier alignment, positive row coverage, obstruction exactness, replay determinacy, and scoped closure jointly form the public export surface for* $\mathsf{SubstitutionAuditMap}^{\uparrow}$*.*

The exported surface presents exactly the displayed BHist *packet, its* hsame *transport row,* Cont *consumer routes,* Pkg *provenance, and local* NameCert *row. It exports the closed substitution audit map as finite data only; it does not export beta-substitution preservation, subject reduction, quotient equality, or host semantic equality.*

Proof. Begin with the carrier and TasteGate alignment. The definition of the carrier fixes the ten rows $T, C, R, S, K, G, H, Q, P, N$, and the checked alignment following it records that these rows are the accepted finite surface.

Use positive row coverage and obstruction row exactness. The positive rows T, C, R, S, K, G exhaust the available audit data, while the excluded preservation and equality principles are exactly absent from the carrier.

Next apply replay determinacy. Any downstream read has to replay the same packet through H and Q before it reaches P and N.

Finally use the scoped closure package. It packages the carrier, classifier alignment, coverage, obstruction, replay, provenance, and local naming rows as the public surface available to consumers. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier public export`

Theorem 779.16 (Substitution audit map downstream reader boundary)**.** *Every downstream reader of an accepted* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *packet must factor through the displayed rows*

$$T, C, R, S, K, G, H, Q, P, N.$$

Such a reader may inspect the term row, closed-boundary row, shift row, substitute row, closed-composition row, generator-closure classifier row, componentwise transport row, continuation route row, provenance row, and local naming row. It cannot acquire beta-substitution preservation, subject reduction, quotient equality, or host semantic equality from the packet.

Proof. First read the positive surface. By Definition 779.4, every positive audit read is confined to T, C, R, S, K, G.

Next route the reader through the administrative surface. By Definition 779.9, transported reads pass through H and Q before reaching P and N.

It remains to exclude the requested external conclusions. Apply Definition 779.5 and Definition 779.13; the absent preservation and equality rows are not created by transport, continuation replay, provenance, or naming. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier downstream reader boundary`

Theorem 779.17 (Substitution audit map formal target package)**.** *The Lean-facing target package for* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *is the finite* BHist *carrier* $A = (T, C, R, S, K, G, H, Q, P, N)$ *together with the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Its intended target is the public export theorem over those rows, with the downstream reader boundary as the exact non-escape condition.*

The package does not require a new primitive carrier, a new classifier, beta preservation, subject reduction, quotient equality, or host semantic equality. Its formal surface is the existing carrier inventory and the row-factorization theorems cited in this chapter.

Proof. Start from the public export theorem. It identifies the finite carrier, TasteGate alignment, coverage, obstruction, replay determinacy, and scoped closure as the consumer-facing package.

Then apply the downstream reader boundary. Every reader factors through $T, C, R, S, K, G, H, Q, P, N$, so the target package needs only the BHist, hsame, Cont, Pkg, and NameCert rows already visible in the carrier.

The excluded principles are handled by obstruction exactness and consumer obstruction scope. Since none of them is a carrier coordinate, the Lean-facing target can be stated over the displayed packet without adding another primitive object. □

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.SubstitutionAuditMapCarrier formal target package`

Closure status for $\mathsf{NameCert}_{\mathsf{SubstitutionAuditMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SubstitutionAuditMap}^{\uparrow}$ packet is generated from a MetaCIC term row, a ClosedAt boundary, closed shift and substitution rows, closed-composition rows, generator-closure classifier rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstitutionAuditMap}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 779.1, Definition 779.2, Definition 779.3, Definition 779.4, Definition 779.5, Definition 779.6, Definition 779.7, Definition 779.8, Definition 779.9, Definition 779.10, Definition 779.11, Definition 779.12, Definition 779.13, Definition 779.14, Definition 779.15, Definition 779.16, and Definition 779.17 bind the closed substitution audit map, its primitive domain and movement rows, deterministic replay route, ledger boundary, obstruction rows, frontier routes, closed-boundary scope, generator-route scope, consumer obstruction scope, finite scoped package, public export surface, downstream reader boundary, and formal target package.

Lean4 ✓ `BEDC.Derived.SubstitutionAuditMapUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstitutionAuditMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close unrestricted substitution, beta-substitution preservation, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Lean verification for the public export theorem, downstream reader boundary, and formal target package remains an independent formal target.

Chapter 780

A Concrete Naming Certificate for MetaCICOpenProblemLedger$^{\uparrow}$

The MetaCICOpenProblemLedger$^{\uparrow}$ packet turns the inventory of chapter 9386 into a finite NameCert ledger. It sits beside the audit synthesis packet of chapter 776, the closed-normal consistency packet of chapter 739, and the normal-false exclusion packet of chapter 732. Those siblings name the positive closed-normal branch and the audit rows; this packet keeps the unresolved MetaCIC rows visible as ledger entries rather than as hidden consistency claims.

Definition 780.1 (MetaCIC open-problem ledger carrier)**.** *A* MetaCICOpenProblemLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, C, N, U, D, E, B, H, R, P, Q)$$

with the following displayed rows:

1. *S is the subject-reduction row, naming beta-substitution preservation, application-argument stability, lambda-domain preservation, and Pi-domain preservation as finite entries;*
2. *C is the confluence row, naming residual-substitution compatibility, closed parallel diamond scope, and arbitrary closed Church–Rosser scope as boundary entries;*
3. *N is the normalization row, naming candidate evidence, closed-candidate evidence, candidate-set membership, and the absence of an evidence-free global strong-normalization export;*
4. *U is the substitution row, naming the substitution-facing algebra consumed by subject reduction and confluence;*
5. *D is the decidability row, naming typing-checker exactness, bounded normal comparison, finished-normal side conditions, and the absence of an unrestricted closed beta-conversion decider;*
6. *E is the typed-example row, naming the gallery witnesses and the absence of exhaustive dependent-program or polymorphic-program coverage;*
7. *B is the blocker row, naming the closed-consistency handoff that still cannot transport arbitrary closed proof terms into the closed-normal exclusion input;*
8. *H stores componentwise* hsame *transport for S, C, N, U, D, E, B;*
9. *R stores displayed* Cont *replay routes for the seven ledger rows;*
10. *P is the* Pkg *provenance over chapter 9386, chapter 776, chapter 739, and chapter 732;*

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemLedger}^{\uparrow}}$ *row.*

The carrier contains no coordinate for arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive example coverage, or full closed consistency.

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedgerTasteGate single carrier alignment`

Theorem 780.2 (MetaCIC open-problem ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemLedge}}$ *surface for Definition 780.1 consists exactly of carrier admission for $L = (S, C, N, U, D, E, B, H, R, P, Q)$, row visibility for the subject-reduction, confluence, normalization, substitution, decidability, typed-example, and blocker entries, componentwise* hsame *transport through H, finite route replay through R, package provenance through P, and local naming through Q.*

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedgerNameCertObligations`

Proof. Project the finite carrier. The seven ledger rows are S, C, N, U, D, E, B, and they match the open-problem families named in chapter 9386.

Read the structural rows separately. Transport through H preserves the row tags, replay through R exposes only displayed routes, and provenance through P records the source chapters without turning their boundaries into new theorem rows.

The local name Q certifies this packet as a NameCert surface. Since the carrier has no further coordinate, the listed rows exhaust the obligation surface. □

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedger namecert obligations`

Theorem 780.3 (MetaCIC open-problem ledger non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$ *packet factors through*

$$S, C, N, U, D, E, B, H, R, P, Q.$$

The consumer may read the positive closed-normal branch through the cited sibling packets and may read each open MetaCIC family as a ledger row, but it cannot obtain arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive typed-example coverage, or full closed consistency from this packet.

Proof. Begin with the carrier definition. Any accepted consumer route receives only the seven ledger rows, componentwise transport, displayed replay routes, provenance, and local naming.

The positive branch remains row-local. The audit synthesis packet, closed-normal consistency packet, and normal-false exclusion packet can be cited through P, but those citations do not add a transport row from arbitrary closed proofs to closed-normal proof inputs.

The closed endpoints named in the statement are absent from the carrier. Transport, replay, provenance, and naming preserve the displayed ledger; they do not manufacture a missing MetaCIC theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedger nonescape`

Theorem 780.4 (MetaCIC open-problem ledger blocker separation)**.** *Agreement on an accepted* $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$ *packet preserves the blocker row B as a boundary row distinct from the subject-reduction row S, the normalization row N, and the closed-normal sibling packets. The ledger can therefore record why full closed consistency is not exported while still exposing the closed-normal consistency and normal-false exclusion surfaces through provenance.*

Proof. Read B as its own coordinate in the carrier. It names the closed-consistency handoff that remains unavailable for arbitrary closed proof terms.

Read S and N as separate coordinates. Subject-reduction discharge and candidate-mediated normalization are not collapsed into B, and the sibling closed-normal packets are cited only by provenance.

Componentwise hsame transport preserves this separation. The downstream packet can report the blocker and the positive closed-normal branch together, but it cannot identify them as a completed full-consistency route. □

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedger blocker separation`

Theorem 780.5 (MetaCIC open-problem ledger family coverage)**.** *Every accepted* MetaCICOpenProblemLedger$^{\uparrow}$ *packet exposes exactly seven open-family rows: subject reduction, confluence, normalization, substitution, decidability, typed examples, and blocker. Each downstream MetaCIC consumer must read those rows before any claim about arbitrary closed subject reduction, arbitrary Church–Rosser, global strong normalization, unrestricted beta-conversion decidability, exhaustive examples, or full consistency.*

Proof. Project the carrier of Definition 780.1. The seven open-family coordinates are precisely S, C, N, U, D, E, B, while H, R, P, Q transport, replay, source, and name those rows.

Apply Definition 780.3. A downstream consumer factors through the displayed tuple before it can mention any of the closed MetaCIC endpoints excluded from the packet.

Finally apply Definition 780.4. The blocker row remains a distinct boundary row, so the family coverage statement records an open ledger rather than a hidden proof of arbitrary closed consistency. □

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedger family coverage`

Theorem 780.6 (MetaCIC open-problem ledger source-row scope)**.** *For every accepted* MetaCICOpenProblemLedge *carrier* $L = (S, C, N, U, D, E, B, H, R, P, Q)$*, the seven ledger rows* S, C, N, U, D, E, B *are scoped by the source open-problem chapter, the audit synthesis packet, the closed-normal consistency packet, and the normal-false exclusion packet. The rows* H, R, P, Q *supply componentwise transport, displayed replay, provenance, and local naming only. Consequently the packet does not export arbitrary subject reduction, Church–Rosser, global normalization, beta-conversion decidability, exhaustive examples, or full consistency.*

Proof. Project the carrier of Definition 780.1. The open-family coordinates are exactly S, C, N, U, D, E, B, while H, R, P, Q are structural rows attached to those coordinates.

Use Definition 780.3. The downstream route factors through the source ledger and the cited sibling packets before it can mention any closed MetaCIC endpoint.

Apply Definition 780.4. The blocker row remains distinct from the positive closed-normal branch exposed through provenance, so the packet cannot identify source-row scope with a full-consistency theorem.

Finally use Definition 780.5. The seven rows cover the named open families as ledger entries, while arbitrary subject reduction, Church–Rosser, global normalization, beta-conversion decidability, exhaustive examples, and full consistency remain outside the finite packet. □

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedger source row scope`

Theorem 780.7 (MetaCIC open-problem ledger export surface)**.** *Every accepted* $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$ *packet exports exactly the seven open-family rows*

$$S, C, N, U, D, E, B$$

and the structural rows

$$H, R, P, Q$$

as its reusable public ledger surface. A consumer may read subject-reduction, confluence, normalization, substitution, decidability, typed-example, and blocker entries as displayed ledger rows, with componentwise transport, replay, provenance, and local naming. No consumer obtains arbitrary subject reduction, Church–Rosser, global normalization, unrestricted beta-conversion decidability, exhaustive examples, or full consistency from this packet.

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemLedgerUp.MetaCICOpenProblemLedgerExportSurface encode append`

Proof. Project the carrier $L = (S, C, N, U, D, E, B, H, R, P, Q)$ of Definition 780.1. The first seven coordinates are the open-family ledger rows, and the last four coordinates are structural rows attached to those families.

Apply Definition 780.2. It identifies carrier admission, row visibility, componentwise transport, replay, provenance, and local naming as the complete NameCert surface for the displayed tuple.

Use Definition 780.3 and Definition 780.4. These theorems keep the closed endpoints outside the packet and keep the blocker row distinct from the positive closed-normal branch exposed through provenance.

Finally apply Definition 780.5 and Definition 780.6. The seven families are exactly the exported public rows, while H, R, P, Q only transport, replay, source, and name those rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$ packet is generated from subject-reduction, confluence, normalization, substitution, decidability, typed-example, blocker, transport, route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemLedger}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 780.1, Definition 780.2, Definition 780.3, Definition 780.4, Definition 780.5, and Definition 780.6 bind the finite open-problem ledger, its non-escape boundary, and the exact source-row scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive typed-example coverage, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Public closure requires a reusable public ledger interface for the seven open-family rows, transport row, replay route, provenance row, and local naming row over the same finite packet.

Chapter 781

A Concrete Naming Certificate for DependentCodomainInversionBoundary$^{\uparrow}$

The DependentCodomainInversionBoundary$^{\uparrow}$ packet realizes the dependent-codomain barrier named in chapter 9386. It sits beside the obstruction packet of chapter 731, the beta-substitution packet of chapter 716, and the frontier ledger of chapter 762. Its role is to name the finite codomain-readback boundary that must be discharged before a subject-reduction consumer may cross from syntactic substitution evidence to dependent application preservation.

Definition 781.1 (Dependent-codomain inversion boundary carrier)**.** *A* DependentCodomainInversionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Pi, A, a, a', C_0, C_1, S, R, O, H, K, P, N)$$

with the following displayed rows:

1. Π *is the dependent function-type row whose codomain may observe the argument;*
2. A *is the displayed domain row shared by the argument and the function type;*
3. a *and* a' *are the source and reduced argument rows;*
4. C_0 *is the codomain readback row obtained by substituting* a *into the displayed codomain;*
5. C_1 *is the codomain readback row obtained by substituting* a' *into the same displayed codomain;*
6. S *is the syntactic substitution-evidence row, including the beta-substitution route supplied by chapter 716;*
7. R *is the subject-reduction handoff row pointing to the finite subject-reduction obstruction surface of chapter 731;*
8. O *is the open-frontier ledger row recording that no unconditional dependent-codomain inversion is exported;*
9. H *stores componentwise* hsame *transport for the function type, domain, arguments, codomain readbacks, substitution row, handoff row, obstruction row, provenance row, and local name;*
10. K *stores displayed* Cont *replay routes for codomain readback, substitution evidence, obstruction readback, and subject-reduction handoff;*
11. P *is the* Pkg *provenance over the open-problem vision, the obstruction packet, the beta-substitution packet, and the open-frontier ledger;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DependentCodomainInversionBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for closed subject reduction, no proof that C_0 and C_1 are interchangeable, no context-reduction theorem, no inhabited discharge bundle, and no host conversion semantics.

Theorem 781.2 (Dependent-codomain inversion boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DependentCodomainInversionBoundary}^{\uparrow}}$ *surface for Definition 781.1 consists exactly of carrier admission for $D = (\Pi, A, a, a', C_0, C_1, S, R, O, H, K, P, N)$, domain and argument row visibility, paired codomain-readback visibility for C_0 and C_1, syntactic substitution evidence through S, subject-reduction obstruction handoff through R, open-frontier ledger exposure through O, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DependentCodomainInversionBoundaryUp.DependentCodomainInversionBoundaryCarrier namecert obligations`

Proof. Project the accepted carrier. The domain, argument, and codomain-readback rows are displayed before any substitution-facing or subject-reduction-facing read is available.

Read the syntactic side through S. The beta-substitution packet supplies a finite substitution route, but that route reaches only the displayed codomain readback rows and does not identify C_0 with C_1.

Read the subject-reduction side through R and O. These rows cite the obstruction packet and the open-frontier ledger as boundary surfaces, so the consumer sees exactly where the dependent-codomain discharge would be required.

The structural rows H, K, P, N transport, replay, package, and name only the displayed rows. Since the carrier contains no context-reduction theorem, discharge bundle inhabitant, or closed subject-reduction row, the listed obligations exhaust the NameCert surface. □

Theorem 781.3 (Dependent-codomain inversion boundary subject-reduction handoff)**.** *Every subject-reduction consumer of an accepted* $\mathsf{DependentCodomainInversionBoundary}^{\uparrow}$ *packet must pass through the displayed paired codomain rows C_0 and C_1, the syntactic substitution row S, the obstruction handoff row R, and the open-frontier ledger row O before it may request dependent-codomain discharge. The packet may hand off this request to the subject-reduction obstruction surface, but it does not close the request and does not export closed subject reduction.*

Proof. Begin with a consumer asking for subject-reduction evidence in the dependent-codomain application case. Carrier projection exposes the two argument rows and the two codomain readbacks as separate coordinates.

Apply the NameCert obligation theorem. The consumer can read syntactic substitution evidence only through S, and it can read the subject-reduction boundary only through R and O.

The handoff is therefore finite and explicit. It records the exact point where codomain readback stability or a compatible discharge surface must enter, while excluding closed subject reduction, host conversion, and context-reduction data from the packet itself. □

Lean4 ✓ `BEDC.Derived.DependentCodomainInversionBoundaryCarrier subject reduction handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{DependentCodomainInversionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{DependentCodomainInversionBoundary}^{\uparrow}$ packet is generated from dependent function-type, domain, argument, codomain-readback, substitution, obstruction, frontier-ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DependentCodomainInversionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 781.1, Definition 781.2, and Definition 781.3 bind the finite codomain-readback boundary and subject-reduction handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{DependentCodomainInversionBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove dependent-codomain inversion, codomain-readback equality, context reduction, inhabited discharge bundles, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired codomain-readback coverage, syntactic substitution routing, obstruction handoff, frontier-ledger exposure, and non-escape over the same finite packet.

Chapter 782

A Concrete Naming Certificate for $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$

The $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet names the finite observation budget used by the Real completeness route of chapter 9238. It sits between the precision-to-window schedule of chapter 650, the selected-tail surface of chapter 458, and the diagonal completion packet of chapter 359; its role is to expose the bounded rows a diagonal consumer may inspect before the $\mathsf{Real}^{\uparrow}$ seal is reached.

Definition 782.1 (Finite observation-budget selector carrier)**.** *A* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the precision-budget row naming the finite set of tolerances and diagonal indices a Real-completion consumer may request;*
2. *S is the selected schedule row assigning each budget entry of B to one finite observation route;*
3. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *window rows selected by the schedule;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of rational observations, radii, and tolerance comparisons read from the windows in W;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the dyadic ledger has been checked;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the displayed handoff;*
7. *H records componentwise* hsame *transports for the budget, schedule, windows, dyadic ledger, regular-sequence handoff, seal, provenance, and local name;*
8. *C records the displayed* Cont *routes from B to S, from S to W, from W to D, from D to R, and from R to E;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationBudgetSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host real equality, countable choice, a hidden limit selector, or an ambient completeness theorem.

Lean4 ✓ BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal route guard

Theorem 782.2 (Finite observation-budget selector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteObservationBudge}}$ *surface for Definition 782.1 consists of five finite obligations:*

1. *carrier exposure, requiring every accepted object to display the packet* $F = (B, S, W, D, R, E, H, C, P, N)$*;*
2. *budget-to-schedule exactness, requiring each requested budget entry in* B *to reach the selected schedule row* S *only through the displayed route in* C*;*
3. *window and dyadic ledger coverage, requiring* S *to select only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W *and requiring every exported* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in* D *to be read from those windows;*
4. *handoff and seal discipline, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *and* $\mathsf{Real}^{\uparrow}$ *seal row* E *to be reached through* D *and then transported by the displayed* H, C, P, N *rows;*
5. *non-escape, requiring every public read to stay inside the displayed packet and to export no quotient stream, host real equality, countable-choice row, hidden limit selector, or ambient completeness theorem.*

Lean4 ✓ BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier namecert obligations

Proof. Carrier exposure is projection from the tuple in Definition 782.1. The packet lists the budget, selected schedule, selected windows, dyadic observations, regular-sequence handoff, real seal, transports, routes, provenance, and local naming row.

Budget-to-schedule exactness is the first displayed route in C. Since the carrier has one selected schedule row S for the budget B, a consumer has no alternate coordinate from which to obtain a different schedule for the same requested observation budget.

Window and dyadic ledger coverage use the next two route segments. The route from S to W exposes only the finite window family, and the route from W to D makes the dyadic observation ledger depend on those windows rather than on an unbounded stream object.

Handoff and seal discipline are supplied by R, E, H, C, P, N. The regular-sequence handoff is reached after the dyadic ledger, the real seal is reached after the handoff, and transport or repacking must replay the same rows through H, C, P, N.

The non-escape clause follows from the carrier inventory. None of the displayed rows is a quotient stream, host real equality test, countable-choice row, hidden selector, or ambient completeness theorem, so the local naming certificate has no route that can export one. □

Theorem 782.3 (Finite observation-budget selector schedule admission)**.** *Every accepted* FiniteObservationBudgetS *packet admits its selected schedule as a finite consumer of the* $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ *route: the budget row* B *is read as the precision request surface, the schedule row* S *is read as the finite route selector, and the rows* W, D, R, E, H, C, P, N *supply exactly the window, dyadic-ledger, handoff, seal, transport, route, provenance, and local naming surface needed by a choice-free limit-schedule consumer.*

Proof. Start with the accepted carrier $F = (B, S, W, D, R, E, H, C, P, N)$. The first two coordinates display the finite precision budget and the selected schedule row that answers each budget entry.

Read the schedule through the $\mathsf{ChoiceFreeLimitSchedule}^{\uparrow}$ dependency named in chapter 650. Its public route expects a request row, a schedule row, selected $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, and the same transport, route, provenance, and naming rows.

These rows are exactly the coordinates of F. The admission therefore adds no selected representative, quotient stream, or host-completeness datum; it only reuses the displayed finite schedule-to-seal route. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier schedule admission`

Theorem 782.4 (Finite observation-budget selector RealUp seal handoff)**.** *For every accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, each* $\mathsf{Real}^{\uparrow}$ *seal read through* E *factors through the finite route*

$$B \to S \to W \to D \to R.$$

The handoff is compatible with the selected-tail route of chapter 458 and with the diagonal-completion route of chapter 359, because all three surfaces read only displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and* $\mathsf{NameCert}$ *rows before exposing a* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the seal coordinate from the carrier. The definition places E after the budget, selected schedule, selected windows, dyadic ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the continuation coordinate records the route into E.

Use the NameCert obligations of Definition 782.2. They require the seal row to be reached through D and R and then repacked only through the displayed H, C, P, N rows.

The selected-tail and diagonal-completion siblings consume the same kind of visible rows. Since F exposes exactly those rows and no hidden analytic coordinate, the $\mathsf{Real}^{\uparrow}$ handoff can be read by those consumers without changing the finite budget route. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier real seal handoff`

Theorem 782.5 (Finite observation-budget selector tail-meet seal compatibility)**.** *Let* $F = (B, S, W, D, R, E, H, C,$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *carrier. Every tail-meet consumer or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of the selected observation budget must read the same finite chain*

$$B \to S \to W \to D \to R \to E.$$

The consumer may use H, C, P, N *to transport, replay, package, and name that chain, but no hidden selector, quotient stream, host equality, or ambient completeness row may replace any segment of the chain.*

Proof. First use the local carrier and NameCert obligations of Definition 782.1 and Definition 782.2. They expose the selected budget route only through the displayed budget row, schedule row, windows, dyadic ledger, regular-sequence handoff, Real seal, transports, continuations, provenance, and local naming row.

Next attach the seal endpoint. By Definition 782.4, every Real-facing read of E already factors through $B \to S \to W \to D \to R$. Thus a $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can reach the seal only by appending E to the same finite chain.

Finally compare the sibling consumers already cited by the chapter. The $\mathsf{RegularCauchyTailSelector}^{\uparrow}$ route of chapter 458 reads a displayed tail-selection service, and the $\mathsf{RealDiagonalCompletion}^{\uparrow}$ route of chapter 359 reads a displayed diagonal-completion service. Since F supplies only $B, S, W, D, R, E, H, C, P, N$, both consumers share the same budget-to-seal chain and cannot import hidden selectors, quotient streams, host equality, or ambient completeness rows. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier tail meet seal compatibility`

Theorem 782.6 (Finite observation-budget selector non-escape)**.** *Every Real-completion, diagonal-limit, tail-selection, or schedule consumer of an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet factors through the displayed finite packet*

$$B, S, W, D, R, E, H, C, P, N.$$

The consumer may inspect the precision budget, selected schedule, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row. It cannot inspect quotient streams, host equality of completed reals, countable choice, hidden limit selectors, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier nonescape`

Proof. Begin with the carrier and NameCert obligations. Carrier exposure fixes the displayed packet, budget-to-schedule exactness fixes the segment from B to S, window and dyadic ledger coverage fixes the segments from S to W and from W to D, and handoff and seal discipline fixes the segments from D to R and from R to E under the same H, C, P, N rows.

Use schedule admission to keep schedule-facing reads on those rows. By Definition 782.3, the packet supplies the choice-free schedule surface without adding an alternate selector.

Use the seal handoff to keep Real-facing reads on the same route. By Definition 782.4, the $\mathsf{Real}^{\uparrow}$ seal is reached only after the finite budget, schedule, windows, dyadic ledger, and regular-sequence handoff have been read.

The excluded analytic objects do not occur as coordinates of F. Therefore no public consumer read can export quotient streams, host equality of completed reals, countable choice, hidden limit selectors, or an ambient completeness theorem. □

Theorem 782.7 (Finite observation-budget selector route determinacy)**.** *Let F and F' be accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets with the same displayed $B, S, W, D, R, E, H, C, P, N$ rows. Then they determine the same finite budget-to-seal route*

$$B \to S \to W \to D \to R \to E.$$

In particular, a Real-completion consumer cannot obtain two distinct routes from the same displayed rows: schedule admission fixes $B \to S$, window and dyadic coverage fix $S \to W \to D$, and seal handoff fixes $D \to R \to E$ under the shared H, C, P, N surface.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier budget route determinacy`

Proof. Start with the common carrier rows. Both packets expose the same budget row, schedule row, window family, dyadic ledger, regular-sequence handoff, real seal, transports, routes, provenance, and local naming row.

The first segment is determined by schedule admission. By Definition 782.3, the budget row B is read only through the displayed selected schedule row S.

The middle segment is determined by the NameCert coverage obligations. Window selection uses the displayed route from S to W, and every dyadic observation exported by D is read through the displayed windows in W.

The final segment is determined by the seal handoff. By Definition 782.4, the $\mathsf{Real}^{\uparrow}$ seal is reached only after the dyadic ledger and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the same H, C, P, N rows repack the result. Thus two accepted packets with the same displayed rows have the same finite route. □

Theorem 782.8 (Finite observation-budget selector shared-window union)**.** *Let*

$$F_1 = (B, S, W_1, D, R, E, H, C, P, N) \qquad \textit{and} \qquad F_2 = (B, S, W_2, D, R, E, H, C, P, N)$$

be accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets whose displayed* $B, S, D, R, E, H, C, P, N$ *rows are compatible and whose selected finite* $\mathsf{StreamName}^{\uparrow}$ *window families are displayed as* W_1 *and* W_2*. Then the union window family* $W_1 \cup W_2$ *determines a finite displayed budget-to-seal route over the same compatible rows*

$$B \to S \to (W_1 \cup W_2) \to D \to R \to E.$$

The union route exports no quotient stream, selected host limit, host real equality, countable choice, hidden selector, host metric, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier shared window union`

Proof. Use schedule admission for both packets. By Definition 782.3, the common budget row B reaches the displayed schedule row S without importing an alternate selector.

Apply route determinacy to the compatible rows. The shared $B, S, D, R, E, H, C, P, N$ surface fixes the budget-to-seal route shape, so replacing the two displayed window families by their finite union does not introduce a second schedule route.

Use dyadic-window exhaustion for W_1 and W_2. Each exported dyadic observation in D is read from a displayed finite window family, hence the union remains a finite displayed window surface rather than a quotient stream or host limit.

Finally apply the diagonal consumer boundary and finite approximation pullback. These keep downstream Real, diagonal, and finite-approximation consumers inside the displayed budget-to-seal rows, so the union route exports none of the excluded analytic or host-level coordinates. □

Theorem 782.9 (Finite observation-budget selector dyadic-window exhaustion)**.** *For every accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, each dyadic observation exported by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *is exhausted by the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*. No exported dyadic observation has a route to quotient streams, host equality of completed reals, countable choice, hidden limit selectors, or any unbounded observation surface outside the displayed packet.*

Proof. Use the carrier exposure clause to identify the finite window family W and the dyadic ledger D inside the packet. The carrier contains no additional stream coordinate and no ambient selector from which an observation could be read.

Apply the window and dyadic ledger coverage obligation from Definition 782.2. It requires the schedule row S to select only the finite $\mathsf{StreamName}^{\uparrow}$ windows in W and requires every exported $\mathsf{DyadicRatCore}^{\uparrow}$ observation in D to be read from those windows.

The non-escape theorem removes the remaining alternatives. By Definition 782.6, public reads cannot export quotient streams, host real equality, countable-choice data, hidden limit selectors,

or an ambient completeness theorem. Hence the selected finite windows exhaust the dyadic observations exported by D. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier dyadic window exhaustion`

Theorem 782.10 (Finite observation-budget selector diagonal consumer boundary)**.** *Every diagonal-limit or Real-completion consumer that uses an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet factors through the displayed route*

$$B \to S \to W \to D \to R \to E$$

together with the H, C, P, N *rows. This is the boundary needed by the Real-completeness vision of chapter 9238, the selected-tail surface of chapter 458, and the diagonal-completion packet of chapter 359: each such consumer may use the finite budget route, but it receives no quotient stream, host real equality, countable choice, hidden selector, or independent completeness oracle from this selector.*

Proof. Begin with the non-escape theorem. It already states that Real-completion, diagonal-limit, tail-selection, and schedule consumers of an accepted packet factor through the displayed finite packet and cannot inspect excluded analytic objects.

Refine the factorization by route determinacy. By Definition 782.7, the displayed rows determine one finite route from the budget to the seal, so a downstream consumer cannot branch into an alternate completion path while using the same selector.

Use dyadic-window exhaustion for the observation boundary. By Definition 782.9, every dyadic observation supplied to the diagonal route is already read from the selected finite windows.

The sibling selected-tail and diagonal-completion chapters named in the introduction consume exactly these visible rows before reaching a $\mathsf{Real}^{\uparrow}$ seal. Therefore any such downstream consumer factors through $B \to S \to W \to D \to R \to E$ together with H, C, P, N, and the selector contributes no further completion principle. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier diagonal consumer boundary`

Theorem 782.11 (Finite observation-budget selector finite-approximation pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let* $A = (R_0, n, k, W_A, D_A, E_A, S_A, H_A, C_A, P_A, N_A)$ *be an accepted* $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet. Suppose the selected finite-window rows, dyadic ledger rows, regular handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows are identified by the displayed* hsame *transports*

$$W\mathsf{hsame}W_A, \qquad D\mathsf{hsame}D_A, \qquad R\mathsf{hsame}R_0, \qquad E\mathsf{hsame}S_A.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer factors through the finite pullback surface formed by the two packets: the selector route* $B \to S \to W \to D \to R \to E$*, the approximation route* $R_0, n, k, W_A, D_A, E_A, S_A$*, the displayed transports* H, H_A*, continuation rows* C, C_A*, package provenance* P, P_A*, and local naming rows* N, N_A*. The pullback exports no quotient stream, selected host limit, host real equality, countable choice, hidden selector, host metric, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier budget approximation pullback`

Proof. Begin on the selector side. By Definition 782.7, the rows of F determine the route from the finite budget to the seal, and by Definition 782.9, the dyadic observations in D are already exhausted by the selected finite windows in W.

Now read the finite approximation side. The threshold route coverage theorem, Definition 601.4, forces every visible approximation observation to pass through R_0, n, k, W_A, D_A before any handoff is exposed. The seal coverage theorem, Definition 601.11, keeps every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer of S_A inside those finite rows and the displayed bookkeeping rows.

Use the displayed hsame transports to identify the common rows componentwise. The hypotheses identify W with W_A, D with D_A, R with R_0, and the selector seal row E with the approximation seal row S_A, so a consumer that reads both packets sees one finite window, one dyadic ledger, one regular handoff, and one seal surface.

Finally apply Definition 601.17 together with Definition 782.10. These two non-escape clauses leave only the combined finite pullback route and its visible $H, H_A, C, C_A, P, P_A, N, N_A$ rows. Since neither carrier contains quotient streams, selected host limits, host real equality, countable choice, hidden selectors, host metrics, or ambient completeness data, no common consumer can receive such data from the pullback. □

Theorem 782.12 (Finite observation-budget selector cofinal readback)**.** *Let $F = (B, S, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Suppose a downstream* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer reads finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by S so that the read windows are cofinal for the displayed budget B, and suppose its* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers transport through the componentwise rows H to the ledger row D. Then the consumer determines the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R and the same* $\mathsf{Real}^{\uparrow}$ *seal row E through the visible continuation, package, and naming rows C, P, N.*

Conversely, every $\mathsf{Real}^{\uparrow}$*-facing seal read from F is exhausted by this cofinal finite-window route:*

$$B \to S \to W \to D \to R \to E.$$

The route exports no quotient stream, host real equality, selected host limit, countable choice, hidden selector, host metric, or ambient completeness theorem.

Proof. Begin with route determinacy. By Definition 782.7, the accepted packet exposes one path from the budget row through the selected schedule, finite windows, dyadic ledger, regular handoff, and terminal Real seal.

Next apply dyadic-window exhaustion. The cofinal read windows are already selected finite $\mathsf{StreamName}^{\uparrow}$ windows inside W, and their $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers transport through H to D. Thus the consumer cannot obtain an additional dyadic source outside the displayed budget route.

The diagonal consumer boundary then places the downstream read on the same public endpoint. By Definition 782.10, every Real-completion, diagonal-limit, or seal-facing consumer that uses the packet factors through $B \to S \to W \to D \to R \to E$ together with H, C, P, N.

Finally use the finite-approximation pullback, Definition 782.11. Any finite readback that reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface through compatible finite approximation rows is pulled back to the same selected window, dyadic ledger, regular handoff, and seal rows. Therefore the handoff is R, the seal is E, and every Real-facing seal read from F is exhausted by the cofinal finite-window route rather than by any undisplayed quotient, host equality, selected limit, choice, hidden selector, metric, or completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier cofinal readback`

Theorem 782.13 (Finite observation-budget selector primitive scope)**.** *For every accepted* FiniteObservationBudget *packet* $F = (B, S, W, D, R, E, H, C, P, N)$*, the scoped source of every Real-completion, diagonal-limit, selected-tail, and schedule consumer is exactly the displayed BEDC primitive surface: the* BHist *packet rows* B, S, W, D, R, E*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationBudgetSelector}^{\uparrow}}$ *row* N*, over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The accepted packet exports no quotient stream, host real equality, countable-choice row, hidden selector, or ambient completeness theorem.*

Proof. The row inventory is fixed by Definition 782.1: an accepted packet has exactly the displayed tuple $F = (B, S, W, D, R, E, H, C, P, N)$ over BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The five local NameCert obligations of Definition 782.2 make those rows the only carrier exposure, budget route, window coverage, seal handoff, and non-escape surface.

The route is then forced rather than chosen from an ambient source. By Definition 782.7, the displayed rows determine the unique finite path $B \to S \to W \to D \to R \to E$. By Definition 782.9, the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger exhaust the observations available to the consumer.

Finally apply Definition 782.10. It identifies Real-completion, diagonal-limit, selected-tail, and schedule consumers as consumers of precisely this finite route together with H, C, P, N. Since the carrier and obligations contain no coordinate for quotient streams, host real equality, countable choice, hidden selectors, or ambient completeness, no such object is exported from the scoped primitive surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier primitive scope`

Theorem 782.14 (Finite observation-budget selector public export)**.** *Every accepted* FiniteObservationBudgetSelec *packet publicly exports one finite budget-to-seal route*

$$B \to S \to W \to D \to R \to E$$

together with the H, C, P, N *rows. This route is consumable by Real-completion, diagonal-limit, selected-tail, and schedule consumers, and it exports no quotient stream, host real equality, countable choice, hidden selector, or ambient completeness theorem.*

Proof. Read the route from the carrier and the NameCert obligations. The carrier displays $B, S, W, D, R, E, H, C, P, N$, and the obligations force budget admission, window selection, dyadic ledger coverage, RegSeqRat handoff, and Real seal access through the displayed rows.

Apply Definition 782.7 and Definition 782.13. They make the route a single public finite object over the BEDC primitive surface rather than a branch selected from an ambient analytic source.

Finally use Definition 782.10 and Definition 782.6. The named consumers may read the route, while quotient streams, host real equality, countable choice, hidden selectors, and ambient completeness are absent from the packet. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier public export`

Theorem 782.15 (Finite observation-budget selector budget-choice freeness)**.** *Let* $F = (B, S, W, D, R, E, H, C, P,$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Then every Real-completion, diagonal-limit, selected-tail, or schedule consumer of* F *receives the finite route*

$$B \to S \to W \to D \to R \to E$$

from the displayed packet rows and not from a countable-choice principle, hidden selector, ambient completion theorem, host real equality test, or quotient-stream representative.

Proof. Start with route determinacy, Definition 782.7. It fixes the route from the budget row B through the stream schedule, finite windows, dyadic ledger, regular readback, and Real seal.

Use dyadic-window exhaustion, Definition 782.9. The observations consumed by the diagonal route are already exhausted by W and D, so no independent choice of observation indices remains outside the displayed packet.

Apply the diagonal consumer boundary and non-escape theorems, Definition 782.10 and Definition 782.6. They force the named consumers through F and exclude countable choice, hidden selectors, host equality, quotient streams, and ambient completeness. Hence the budget route is choice-free in the displayed BEDC sense. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier budget choice freeness`

Theorem 782.16 (Finite observation-budget selector window-seal commutation)**.** *Let $F = (B, S, W, D, R, E, H, C$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any consumer route that first reads the selected finite windows W and then the Real seal row E has the same displayed public surface as the route that first names the budget-to-seal path $B \to S \to W \to D \to R \to E$ and then restricts its read to W before reaching E. Both readings factor through the same dyadic ledger D, regular readback R, and structural rows H, C, P, N.*

Proof. Read the public route from Definition 782.14. It exposes one finite budget-to-seal path together with the transport, continuation, package, and naming rows.

Localize the window read by Definition 782.9. The dyadic observations available on the route are exactly those exhausted by the finite windows W and the ledger row D.

Now use route determinacy and the diagonal consumer boundary, Definition 782.7 and Definition 782.10. They prevent a second route from inserting another seal, hidden selector, or ambient completion step between W and E. Thus reading the window before the seal or reading it as the window coordinate of the public route gives the same displayed surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier window seal commutation`

Theorem 782.17 (Finite observation-budget selector budget-ledger minimality)**.** *Let $F = (B, S, W, D, R, E, H, C,$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any finite ledger surface that serves the same Real-completion, diagonal-limit, selected-tail, and schedule consumers without exporting quotient streams, host real equality, countable choice, hidden selectors, or ambient completeness must contain the displayed budget row B, schedule row S, window row W, dyadic ledger D, regular readback R, Real seal E, and structural rows H, C, P, N up to the recorded* hsame *transports.*

Proof. Begin with the carrier and NameCert obligations. The carrier of Definition 782.1 lists $B, S, W, D, R, E, H, C, P, N$, and Definition 782.2 assigns these rows the budget, window, dyadic, readback, seal, transport, route, provenance, and naming roles.

Use route determinacy. By Definition 782.7, deleting any of B, S, W, D, R, E breaks the displayed path from finite budget to seal.

Use dyadic-window exhaustion and the diagonal consumer boundary. By Definition 782.9 and Definition 782.10, the named consumers need the window and dyadic ledger rows specifically, not merely an unnamed analytic substitute.

Finally apply public export and non-escape, Definition 782.14 and Definition 782.6. Since stronger ambient objects are excluded, the only available ledger surface is the displayed packet, up to the componentwise transports already recorded by H. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.BudgetLedgerMinimality.FiniteObservationBudgetSe`
`budget ledger minimality`

Theorem 782.18 (Finite observation-budget selector shared-tail-modulus pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let*

$$T = (X_0, X_1, M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1, E_T, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet. Suppose the selector window, dyadic ledger, regular readback, and seal rows are identified with the shared-tail rows by componentwise* hsame *transports*

$$W\,\mathsf{hsame}\,W_0, \qquad D\,\mathsf{hsame}\,D_T, \qquad R\,\mathsf{hsame}\,Q_0, \qquad E\,\mathsf{hsame}\,E_T.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of* F *and* T *factors through one finite pullback route from the selector budget to the shared-tail seal:*

$$B \to S \to W \to D \to R \to E \qquad \text{and} \qquad M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1, E_T.$$

The route uses only the displayed transports, continuations, package provenance, and local naming rows $H, H_T, C, C_T, P, P_T, N, N_T$*. It exports no quotient stream, host real equality, countable choice, hidden selector, independent threshold replacement, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier`
`shared tail modulus pullback`

Proof. Start with the selector route. By Definition 782.7, the rows of F determine the finite path $B \to S \to W \to D \to R \to E$. By Definition 782.9, the dyadic ledger row D is exhausted by the selected finite windows W.

Read the shared-tail packet next. The tail synchronization theorem, Definition 983.3, forces both Cauchy tail requests through the single displayed threshold route $M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1$. The Real seal handoff theorem, Definition 983.4, exposes the seal only after that shared finite route has been replayed.

Use the displayed hsame transports to identify the common rows. The hypotheses identify the selector window with W_0, the selector dyadic ledger with D_T, the selector readback with Q_0, and the selector seal with E_T. Thus a consumer of both packets sees one finite window, one dyadic ledger, one regular readback row, and one Real seal row on the selected side.

Finally apply Definition 782.14 and the non-escape clauses already attached to the two packets. Neither carrier inventory contains quotient streams, host real equality, countable choice, hidden selectors, independent threshold replacement, or ambient completeness data. The common consumer therefore factors through the finite pullback route and cannot obtain any stronger object outside the displayed rows. □

Theorem 782.19 (Finite observation-budget selector CauchyLimitSeal pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C$
be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let*

$$L = (X, S_L, D_L, G, E_L, T_L, P_L, N_L)$$

be an accepted $\mathsf{CauchyLimitSeal}^{\uparrow}$ *public-route packet. Suppose the selector's finite window, dyadic ledger, regular handoff, and Real seal rows are identified with the seal packet's schedule, dyadic ledger, regular source, and seal rows by displayed* hsame *transports*

$$W\mathsf{hsame}S_L, \qquad D\mathsf{hsame}D_L, \qquad R\mathsf{hsame}G, \qquad E\mathsf{hsame}E_L.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of* F *and* L *factors through the pullback of the selector route*

$$B \to S \to W \to D \to R \to E$$

and the seal public route

$$X \to S_L \to D_L \to G \to E_L,$$

together with the visible transport, continuation, package, and local naming rows $H, C, P, N, T_L, P_L, N_L$*. The pullback exports no quotient stream, host real equality, selected external limit, countable choice, hidden selector, or ambient completeness theorem.*

Proof. Start with the selector route. By Definition 782.7, the accepted packet F has only the displayed finite path from B through S, W, D, R to E. By Definition 782.9, the dyadic ledger read is already exhausted by the finite window row W.

Read the seal packet through its public boundary. The CauchyLimitSeal public route boundary of Definition 386.8 confines any seal-facing read to $X, S_L, D_L, G, E_L, T_L, P_L, N_L$. The non-escape theorem Definition 386.4 removes quotient streams, selected host limits, uniqueness principles, and ambient completeness from the seal inventory.

Use the displayed hsame transports componentwise. The window row of the selector is identified with the schedule row of the seal packet, the dyadic ledger with the seal ledger, the regular handoff with the regular source row, and the selector seal with the seal endpoint row. Therefore a common consumer reads one finite window, one dyadic ledger, one regular source or handoff surface, and one Real-facing seal.

Finally combine the two visible row inventories. The selector non-escape and public-export rows exclude host real equality, countable choice, hidden selectors, and ambient completeness, while the seal non-escape rows exclude quotient streams and selected external limits. The common consumer therefore factors through the stated pullback and cannot export any coordinate outside those displayed routes. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.CauchyLimitSealPullback.FiniteObservationBudgetS`
`cauchy limit seal pullback`

Theorem 782.20 (Finite observation-budget selector UniformCauchyCriterion pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let*

$$U = (I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U)$$

be an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet. Suppose the selector window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal rows are identified with the uniform Cauchy shared-window, tolerance-ledger, regular consumer, and seal rows by componentwise* hsame *transports*

$$W\mathsf{hsame}W_U, \qquad D\mathsf{hsame}Q_U, \qquad R\mathsf{hsame}T_U, \qquad E\mathsf{hsame}E_U.$$

Then every common $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or completion-facing consumer of* F *and* U *factors through the finite pullback surface formed by the selector route*

$$B \to S \to W \to D \to R \to E$$

and the uniform Cauchy route

$$I \to W_U \to M_U \to Q_U \to T_U \to E_U,$$

together with the displayed rows $H, H_U, C, C_U, P, P_U, N, N_U$. *The pullback exports no quotient streams, host real equality, countable choice, hidden selectors, selected limits, host metrics, or ambient completeness coordinate.*

Proof. Start with the selector side. By Definition 782.7, the rows of F determine the finite route $B \to S \to W \to D \to R \to E$. By Definition 782.9, every dyadic observation in D is already exhausted by the selected finite windows in W.

Read the uniform Cauchy packet through its public route. The RegSeqRat tail equivalence theorem, Definition 435.11, identifies the shared-window tail ledger I, W_U, M_U, Q_U, T_U with the finite-window $\mathsf{RegSeqRat}^{\uparrow}$ consumer whose seal is E_U. The public export theorem, Definition 435.20, then exposes only the surface $I, W_U, M_U, Q_U, T_U, E_U, H_U, C_U, P_U, N_U$ to downstream regular, real, or completion-facing consumers.

Use the displayed hsame transports to form the common surface. The hypotheses identify the selector window with the uniform shared-window row, the selector dyadic ledger with the uniform tolerance ledger, the selector handoff with the uniform regular consumer row, and the selector seal with the uniform seal row. Thus a consumer of both packets sees one finite window surface, one dyadic tolerance ledger, one regular handoff, and one real-facing seal.

Finally apply Definition 782.14 together with the uniform public export. Since both packets expose only the displayed routes and rows, any common $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or completion-facing consumer factors through the finite pullback surface above. The excluded objects are absent from the carrier inventories and from the public export surfaces, so the pullback cannot supply quotient streams, host real equality, countable choice, hidden selectors, selected limits, host metrics, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier uniform cauchycriterion pullback`

Theorem 782.21 (Finite observation-budget selector terminal budget nonempty)**.** *Any accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet used by a* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or completion-facing consumer exposes a nonempty displayed finite budget route*

$$B \to S \to W \to D \to R \to E.$$

The endpoint E *is reached only through the selected budget row* B, *schedule row* S, *finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W, *dyadic ledger* D, *and regular handoff row* R *of the same packet.*

Proof. Start with Definition 782.7. It identifies the displayed path from B through S, W, D, R to E as the selector route available to downstream consumers.

Use Definition 782.9. The dyadic observations in D are already read from the finite windows in W, so the route cannot skip the displayed window row or replace it by an ambient stream.

Apply Definition 782.14 and Definition 782.19. These keep Real-facing, regular, seal-facing, and completion-facing reads on the same visible selector route, with E as the displayed terminal seal. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal budget nonempty`

Theorem 782.22 (Finite observation-budget selector window index scope)**.** *Every window index read by a* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet is scoped by the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W *and dyadic ledger* D*. It is not scoped by quotient streams, host equality, countable choice, hidden selectors, selected limits, host metrics, or ambient completeness.*

Proof. Read the selector packet through Definition 782.21. Any consumer-visible index occurs on the displayed route $B \to S \to W \to D \to R \to E$.

Use Definition 782.9 to localize dyadic observations to the finite window row W and ledger row D. No additional index source is introduced by the dyadic side.

Finally apply Definition 782.6, Definition 782.14, and the seal pullback of Definition 782.19. The excluded host and quotient objects are absent from the carrier inventory and from the public export surface, so they cannot scope a displayed window index. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.WindowIndexScope.FiniteObservationBudgetSelector window index scope`

Theorem 782.23 (Finite observation-budget selector terminal selector synchronization)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, or completion-facing consumer reads the same terminal selector route*

$$B \to S \to W \to D \to R \to E.$$

The route is synchronized with the CauchyLimitSeal and UniformCauchyCriterion pullback surfaces, and it exports no quotient stream, host real equality, selected external limit, countable choice, hidden selector, host metric, or ambient completeness theorem.

Proof. Start with Definition 782.7. It identifies $B \to S \to W \to D \to R \to E$ as the selector route available to any consumer of the accepted packet.

Use Definition 782.19. A CauchyLimitSeal-facing read identifies the selector window, dyadic ledger, regular handoff, and Real seal with the corresponding public seal rows, so the seal consumer sees the same terminal selector route rather than a separate limit stream.

Use Definition 782.20 for the uniform-Cauchy side. The shared-window, tolerance-ledger, regular-consumer, and seal rows are pulled back to W, D, R, E, so the UniformCauchyCriterion consumer is synchronized with the same terminal route.

Finally apply Definition 782.21 and Definition 782.22. The terminal route is nonempty, every visible index is scoped by W and D, and ledger exhaustion leaves no quotient stream, host real equality, selected external limit, countable choice, hidden selector, host metric, or ambient completeness theorem to export. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal selector synchronization`

Theorem 782.24 (Finite observation-budget selector RealTailAgreementSeal pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let* $Q = (R_0, S_0, W_Q, D_Q, A_Q, H_Q, C_Q, P_Q, N_Q)$ *be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet. Suppose* $W\mathsf{hsame}W_Q$*,* $D\mathsf{hsame}D_Q$*,* $R\mathsf{hsame}R_0$*, and* E *is the* $\mathsf{Real}^{\uparrow}$*-facing seal row reached after the agreement row* A_Q*. Then every common* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the finite pullback of*

$$B \to S \to W \to D \to R \to E$$

with the seal route R_0, S_0, W_Q, D_Q, A_Q, together with the visible hsame *transport,* Cont *continuation,* Pkg *provenance, and* NameCert *rows of the two accepted packets.*

The pullback exports no quotient stream equality, selected host limit, host convergence theorem, countable choice, hidden selector, second budget gate, or ambient completeness principle.

Proof. Begin with Definition 782.7. It fixes the selector-side path from B through S, W, D, R to the terminal seal row E.

Apply Definition 782.23. Any Real-facing, regular, uniform, seal-facing, or completion-facing read of F is already synchronized with that same terminal selector route.

On the seal side, use Definition 699.10. The accepted $\mathsf{RealTailAgreementSeal}^{\uparrow}$ packet places its paired tail request on the displayed budget and agreement rows R_0, S_0, W_Q, D_Q, A_Q before the Real-facing seal can be consumed.

Finally apply Definition 699.36. Since the shared rows satisfy $W\mathsf{hsame}W_Q$, $D\mathsf{hsame}D_Q$, and $R\mathsf{hsame}R_0$, every common Real or completion consumer is forced through the displayed finite pullback, and the absent quotient, host-limit, host-convergence, choice, hidden-selector, second-budget, and completeness objects cannot be projected from either packet. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.RealTailAgreementSealPullback.FiniteObservationB`
`realtailagreementseal pullback`

Theorem 782.25 (Finite observation-budget selector tail-agreement limit-seal triangle)**.** *Let $F = (B, S, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, let $Q = (R_Q, S_Q, W_Q, D_Q, A_Q$ be an accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *packet, and let $L = (X, S_L, D_L, G, E_L, T_L, P_L, N_L)$ be an accepted* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packet. Suppose the selector rows are identified with the tail-agreement and limit-seal rows by displayed transports*

$$W\mathsf{hsame}W_Q, \quad D\mathsf{hsame}D_Q, \quad R\mathsf{hsame}R_Q, \quad E\mathsf{hsame}A_Q, \quad X\mathsf{hsame}R, \quad S_L\mathsf{hsame}W, \quad D_L\mathsf{hsame}D, \quad E_L\mathsf{hsame}E.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of these packets factors through the single finite triangle*

$$B \to S \to W \to D \to R \to E, \qquad R_Q, S_Q, W_Q, D_Q, A_Q, \qquad X \to S_L \to D_L \to G \to E_L.$$

The factorization uses only the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of F, Q, and L. It exports no quotient stream, selected host limit, host real equality, hidden family limit, countable choice, second selector route, or ambient completeness principle.*

Proof. Begin with the selector side. The route determinacy theorem Definition 782.7, public export theorem Definition 782.14, and pullback theorem Definition 782.24 place every accepted selector consumer on the displayed route $B \to S \to W \to D \to R \to E$.

Use the RealTailAgreementSeal side next. The budget-source lock Definition 699.10 and downstream route exhaustion Definition 699.36 identify the shared tail-agreement rows R_Q, S_Q, W_Q, D_Q, A_Q with the same finite budget, window, dyadic, regular, and seal surface.

Finally attach the CauchyLimitSeal side. The public route boundary Definition 386.8 and root completion consumer non-escape theorem Definition 386.94 force the limit-seal read through X, S_L, D_L, G, E_L and exclude every undisplayed completion coordinate. With the stated hsame transports, the three routes are one finite triangle over the displayed history, continuation, provenance, and naming rows. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier`
`tail agreement limit seal triangle`

Theorem 782.26 (Finite observation-budget selector budget-seal cut)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *or completion-facing consumer of* F *factors through the unique displayed cut at the dyadic and regular handoff:*

$$B \to S \to W \to D \qquad | \qquad D \to R \to E.$$

The lower segment $B \to S \to W \to D$ *supplies all finite observation and dyadic ledger data, while the upper segment* $D \to R \to E$ *supplies all regular-sequence and* $\mathsf{Real}^{\uparrow}$*-seal data.*

Any shared pullback with $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, *or* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *preserves this cut by displayed* hsame *transports. Such a pullback cannot move quotient streams, host real equality, countable choice, hidden selectors, selected limits, host metrics, or ambient completeness across the cut.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier budget seal cut`

Proof. Start with the selector route itself. The route determinacy theorem Definition 782.7 fixes the single displayed path $B \to S \to W \to D \to R \to E$, while Definition 782.9 makes the finite windows and dyadic ledger exhaust the lower observation segment.

The public export theorem Definition 782.14 then exposes that route to downstream regular, real, schedule, and completion-facing consumers without adding any alternate selector. Therefore the only handoff from finite observation to seal-facing data occurs at the displayed dyadic ledger row D.

For seal consumers, apply the three pullback theorems separately. The CauchyLimitSeal pullback Definition 782.19, the UniformCauchyCriterion pullback Definition 782.20, and the RealTailAgreementSeal pullback Definition 782.24 identify the shared rows only by displayed hsame transports, so each consumer preserves the same lower and upper segments.

Finally use the terminal tail-agreement limit-seal triangle Definition 782.25. It shows that even the combined tail-agreement and limit-seal consumer remains on the displayed finite triangle. Since the cited public and pullback surfaces contain no quotient streams, host real equality, countable choice, hidden selectors, selected limits, host metrics, or ambient completeness coordinates, none of those objects can cross the dyadic and regular handoff cut. □

Theorem 782.27 (Finite observation-budget selector RealCompletionSelectorSeal pullback)**.** *Let* $F = (B, S, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let*

$$S' = (B', W', R', L', E', H', C', P', N')$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose the selector budget, selected window, regular readback, and terminal seal rows are identified with the selector-seal rows by displayed componentwise transports*

$$B\mathsf{hsame}B', \qquad W\mathsf{hsame}W', \qquad R\mathsf{hsame}R', \qquad E\mathsf{hsame}E'.$$

Then every common $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the single finite selector-to-seal pullback formed by the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *route*

$$B \to S \to W \to D \to R \to E$$

and the RealCompletionSelectorSeal$^{\uparrow}$ *route*

$$B' \to W' \to R' \to L' \to E',$$

together with only the visible hsame *transport,* Cont *continuation,* Pkg *provenance, and* NameCert *rows* $H, H', C, C', P, P', N, N'$*. The pullback exports no quotient stream equality, host real equality, countable choice, hidden selector, detached completion limit, second selector budget, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier realcompletionselectorseal pullback`

Proof. Begin with the finite observation-budget side. The budget-seal cut of Definition 782.26 confines every Real, regular, seal, and completion-facing read of F to the displayed path $B \to S \to W \to D \to R \to E$ and preserves the dyadic-to-regular handoff under all visible pullbacks.

Attach the selector-seal side. By Definition 1081.5, an accepted RealCompletionSelectorSeal$^{\uparrow}$ packet occupies exactly the L10 selector-to-seal route through the selected window, regular readback, CauchyLimitSeal handoff, and terminal Real endpoint. By Definition 1081.6, that route is unreadable until the selector-budget row B' has been exposed.

Use the displayed transports to identify the shared surface. The hypotheses identify B with B', W with W', R with R', and E with E'. The CauchyLimitSeal pullback of Definition 782.19 supplies the seal-facing handoff through the visible limit-seal row, so the two routes meet at the same finite budget, window, readback, handoff, and endpoint surface.

Finally apply terminal exhaustion, Definition 1081.10. Every terminal Real or Completion read of the selector-seal packet is exhausted by B', W', R', L', E' and the structural rows H', C', P', N'. Since the selector side exposes only H, C, P, N beyond its displayed route, the common consumer factors through the stated pullback and cannot project any undisplayed quotient, host-equality, choice, hidden-selector, detached-limit, second-budget, or completeness coordinate. □

Theorem 782.28 (Finite observation-budget selector terminal route no-branching)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. If two Real-completion or diagonal consumers expose the same displayed rows* $B, S, W, D, R, E, H, C, P, N$*, then they determine the same terminal route*

$$B \to S \to W \to D \to R \to E.$$

In particular, no second terminal route can pass through quotient streams, host real equality, countable choice, hidden selectors, side-channel completion, or ambient completeness.

Proof. The displayed rows first determine the budget-to-window spine. By Definition 782.7, any consumer that reads $B, S, W, D, R, E, H, C, P, N$ is confined to the route from the budget row through the selected schedule, finite window family, dyadic ledger, regular handoff, and terminal seal.

The terminal endpoint is inhabited only on that same displayed surface. The nonemptiness statement of Definition 782.21 supplies the terminal budget witness, while Definition 782.22 keeps every window index inside the same finite observation rows.

It remains to rule out an alternate Real-completion exit. Apply the pullback of Definition 782.27: a Real-completion or diagonal consumer sharing the displayed rows factors through

the same finite selector-to-seal pullback. Since the pullback exports no quotient stream, host real equality, countable choice, hidden selector, side-channel completion, or ambient-completeness coordinate, the two consumers have no available coordinate from which to form a distinct terminal route. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal route no branching`

Theorem 782.29 (Finite observation-budget selector terminal route endpoint exactness)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-completion, diagonal-limit,* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *or* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *consumer that reaches a terminal endpoint through* F *reaches the displayed endpoint* E *through the route*

$$B \to S \to W \to D \to R \to E.$$

Moreover, any endpoint with the same displayed budget, schedule, window, dyadic, regular, seal, transport, continuation, package, and local NameCert *rows is identified with* E *by the recorded componentwise* hsame *transports.*

Proof. Begin with budget-route determinacy. By Definition 782.7, every consumer of the displayed rows $B, S, W, D, R, E, H, C, P, N$ is confined to the route from the budget row through the selected schedule, finite window family, dyadic ledger, regular handoff, and terminal Real seal.

The endpoint is then present on exactly that route. The terminal budget nonemptiness theorem Definition 782.21 supplies the terminal budget witness, while the budget-seal cut Definition 782.26 keeps the lower finite-observation segment and upper regular-seal segment attached at D and R.

Next handle the named downstream consumers. The CauchyLimitSeal, UniformCauchyCriterion, and RealTailAgreementSeal pullbacks identify their shared rows with the selector rows by displayed hsame transports, and Definition 782.27 does the same for RealCompletionSelectorSeal consumers.

Finally apply terminal route no-branching, Definition 782.28. Since any endpoint with the same displayed rows determines the same route $B \to S \to W \to D \to R \to E$, the endpoint is the displayed E up to the recorded componentwise hsame transports, and no quotient, choice, hidden selector, side-channel completion, or ambient completeness coordinate can supply a different terminal endpoint. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal route endpoint exactness`

Theorem 782.30 (Finite observation-budget selector terminal diagonal pullback functoriality)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. The terminal route endpoint exactness, finite-approximation pullback, cofinal readback, and diagonal-consumer boundary determine a functorial pullback surface for every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and diagonal-completion consumer of* F.

The consumer receives only the displayed $B, S, W, D, R, E, H, C, P, N$ *rows and their recorded* hsame, Cont, Pkg, *and* NameCert *structure. It receives no quotient stream, host real equality, hidden selector, countable choice, host metric, or ambient completeness row.*

Proof. Begin with the route. The budget-route determinacy theorem Definition 782.7 fixes the single path $B \to S \to W \to D \to R \to E$ for every accepted consumer of the displayed selector rows.

Attach the diagonal boundary. By Definition 782.10, any diagonal-completion read of the selector must use the displayed budget, selected schedule, finite windows, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and local naming rows.

Use the finite pullback and readback surfaces next. The finite-approximation pullback Definition 782.11 and cofinal readback theorem Definition 782.12 identify the common $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and diagonal-completion reads by displayed componentwise transports, so each consumer acts on the same finite surface.

Finally apply endpoint exactness, Definition 782.29. Since the terminal endpoint is the displayed E along the same route, the pullback is functorial in the permitted consumers and has no coordinate from which to read quotient streams, host real equality, hidden selectors, countable choice, host metrics, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal diagonal pullback functoriality`

Theorem 782.31 (Finite observation-budget selector Real-seal window saturation)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any Real-completion or diagonal consumer that reaches the same terminal row E through the same budget, schedule, dyadic, regular, transport, continuation, provenance, and local naming rows B, S, D, R, H, C, P, N must use the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window family W. Adding or replacing a window without changing the budget or schedule is not a refinement of the same selector packet; it is a different accepted packet with a different window row.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier real seal window saturation`

Proof. Start with the route into the terminal Real seal. By Definition 782.4, every Real-facing read of E is reached only after the finite budget, schedule, selected window family, dyadic ledger, and regular handoff have been exposed.

Use window exhaustion for the lower observation segment. By Definition 782.9, the dyadic observations in D are already covered by the displayed finite windows W, so a diagonal or completion consumer has no independent dyadic source outside that family.

Now apply route determinacy. The theorem Definition 782.7 fixes the displayed route from B through S, W, D, R to E whenever the structural rows H, C, P, N are preserved.

Finally use the terminal theorems to exclude a hidden selector. By Definition 782.28 and Definition 782.29, a consumer reaching the same endpoint through the same displayed rows cannot pass through a second window source. A changed window family therefore changes the accepted selector packet rather than refining the original one. □

Theorem 782.32 (Finite observation-budget selector terminal window minimality)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every terminal* $\mathsf{Real}^{\uparrow}$*-facing read of F factors through exactly the displayed finite budget row B, selected* $\mathsf{StreamName}^{\uparrow}$ *window family W,*

DyadicRatCore$^{\uparrow}$ *ledger* D, RegSeqRat$^{\uparrow}$ *handoff* R, *componentwise* hsame *transports* H, *displayed* Cont *route* C, Pkg *provenance* P, *and local* NameCert *row* N, *with the selected schedule row* S *mediating the route*

$$B \to S \to W \to D \to R \to E.$$

If any one of the displayed budget, schedule, window, dyadic ledger, regular handoff, seal, transport, continuation, provenance, or local naming rows is dropped, the remaining rows no longer determine the terminal endpoint exactness of Definition 782.29.

Proof. First identify the only route available to a terminal consumer. The budget-route determinacy theorem Definition 782.7 fixes the path from B through S, W, D, R to E, and terminal selector synchronization Definition 782.23 keeps the selected schedule and window rows attached to that path.

Next remove terminal branching. By Definition 782.28, two consumers exposing the same displayed rows determine the same terminal route, so there is no alternate endpoint route through quotient streams, host real equality, countable choice, hidden selectors, side-channel completion, or ambient completeness.

It remains to record sufficiency and minimality. Sufficiency is exactly endpoint exactness from Definition 782.29: the displayed rows identify the terminal endpoint with E by componentwise hsame transport. For minimality, each listed row is used by one segment of the displayed route or by its transport, continuation, provenance, and naming certificate. Dropping such a row removes the corresponding segment or certificate witness, so the endpoint can no longer be recovered by the cited exactness theorem. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal window minimality`

Theorem 782.33 (Finite observation-budget selector diagonal-tail coverage)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. The terminal window inventory of Definition 782.32 covers every diagonal-tail consumer route used by* RegularCauchyTailSelector$^{\uparrow}$, RealDiagonalCompletion$^{\uparrow}$, *and* RealCompletionSelectorSeal$^{\uparrow}$*: such a consumer reads its terminal* Real$^{\uparrow}$ *handoff only through the finite route*

$$B \to S \to W \to D \to R \to E$$

and through the displayed H, C, P, N *rows. The coverage imports no quotient stream, countable choice operation, host real equality, hidden selector, detached completion limit, or independent completeness oracle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier diagonal tail coverage`

Proof. Begin with the tail-facing side. The tail-meet seal compatibility theorem Definition 782.5 places every selected-tail or seal consumer on the displayed finite chain, and the diagonal consumer boundary Definition 782.10 keeps diagonal-limit and Real-completion consumers inside the same rows.

Now attach the selector-seal consumer. The RealCompletionSelectorSeal pullback Definition 782.27 identifies the shared budget, selected window, regular readback, and terminal seal rows with the selector-seal route using only displayed hsame, Cont, Pkg, and NameCert data.

Finally apply terminal window minimality, Definition 782.32. Its inventory is sufficient for the terminal endpoint and loses exactness when a displayed row is removed. Therefore the diagonal-tail consumers named in the statement are covered by the same finite terminal window, while the excluded quotient, choice, host-equality, hidden-selector, detached-limit, and completeness coordinates remain absent. □

Theorem 782.34 (Finite observation-budget selector terminal section exactness)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every terminal Real-completion or diagonal-tail section whose displayed rows agree with* $B, S, W, D, R, E, H, C, P, N$ *is exactly the finite section pulled back along* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The section exports no quotient stream, host real equality, countable choice, hidden selector, detached completion limit, metric-completeness row, or ambient completeness oracle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal section exactness`

Proof. First use terminal window minimality, Definition 782.32. It supplies the full selector inventory $B, S, W, D, R, E, H, C, P, N$ and says that dropping any one of those rows loses terminal endpoint determination.

Next apply terminal route endpoint exactness, Definition 782.29. Since the displayed rows agree with the accepted packet, the terminal section reaches the same $\mathsf{Real}^{\uparrow}$ endpoint E through the same componentwise hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

For diagonal-tail consumers, use Definition 782.33. It keeps the $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, and $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ reads on the same finite route

$$B \to S \to W \to D \to R \to E.$$

Finally identify that route with the existing four-face finite-section pullback. By Definition 126.586, agreement of the finite $\mathsf{StreamName}^{\uparrow}$ windows with the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and current $\mathsf{Real}^{\uparrow}$ seal determines exactly the same terminal section and exports none of the excluded quotient, host-equality, choice, hidden-selector, detached-limit, metric-completeness, or ambient-completeness coordinates. □

Theorem 782.35 (Finite observation-budget selector terminal section pullback uniqueness)**.** *Let*

$$F_0 = (B, S, W, D, R, E, H, C, P, N)$$

and

$$F_1 = (B, S, W, D, R, E, H, C, P, N)$$

be accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets with the same displayed budget, schedule, window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and local naming rows. Suppose that the terminal sections of* F_0 *and* F_1 *both satisfy the finite-section pullback along* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Then the two terminal sections determine the same budget-to-seal route*

$$B \to S \to W \to D \to R \to E$$

and the same $\mathsf{Real}^{\uparrow}$ *seal up to the recorded componentwise* hsame *transports. No quotient stream, host real equality, choice row, hidden selector, metric-completeness row, or ambient-completeness oracle can distinguish the two terminal sections.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal section pullback uniqueness`

Proof. First apply terminal section exactness, Definition 782.34, to each accepted packet. Since both displayed rows are $B, S, W, D, R, E, H, C, P, N$, both terminal sections are the finite section pulled back through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

Next identify the endpoint route. Terminal route endpoint exactness, Definition 782.29, fixes the endpoint at the displayed seal E, while terminal window minimality, Definition 782.32, keeps every route segment attached to the same rows $B, S, W, D, R, E, H, C, P, N$.

Finally use the four-face pullback exactness Definition 126.586. The shared finite windows, dyadic ledger, regular readback, and Real seal determine one pulled-back section, so the two terminal sections induce the same displayed chain and the same $\mathsf{Real}^{\uparrow}$ seal under the componentwise hsame transports. The non-escape clauses in the cited exactness and minimality theorems exclude quotient streams, host equalities, choice rows, hidden selectors, metric-completeness rows, and ambient-completeness oracles as possible distinguishing data. □

Theorem 782.36 (Finite observation-budget selector terminal cofinal-window coverage)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every cofinal terminal consumer compatible with the displayed selector rows factors through the same finite window route*

$$B \to S \to W \to D \to R \to E.$$

Compatibility with $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegularCauchyTailSelector}^{\uparrow}$*,* $\mathsf{RealDiagonalCompletion}^{\uparrow}$*, and* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *is therefore mediated only by the displayed budget, schedule, window, dyadic ledger, regular handoff, Real seal, componentwise* hsame *transports,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The exported terminal surface contains no quotient stream, countable choice operation, host real equality, hidden selector, detached completion limit, or ambient completeness oracle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal cofinal window coverage`

Proof. First fix the finite terminal inventory. Terminal window minimality, Definition 782.32, says that every terminal $\mathsf{Real}^{\uparrow}$-facing read of F uses exactly the displayed rows $B, S, W, D, R, E, H, C, P, N$ with S mediating the route through the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Next attach the diagonal-tail consumers. By Definition 782.33, the $\mathsf{RegularCauchyTailSelector}^{\uparrow}$, $\mathsf{RealDiagonalCompletion}^{\uparrow}$, and $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ reads of the packet all remain on the same finite route and import none of the excluded quotient, choice, host-equality, hidden-selector, detached-limit, or completeness coordinates.

Now pass from a compatible terminal section to the displayed section. Terminal section exactness, Definition 782.34, identifies any section agreeing with the displayed rows as the finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use pullback uniqueness. The uniqueness theorem Definition 782.35, together with Definition 126.586, leaves only the chain $B \to S \to W \to D \to R \to E$ as the cofinal terminal route. Hence every compatible cofinal consumer factors through that route and exports only the displayed structural rows. □

Theorem 782.37 (Finite observation-budget selector terminal cofinal-window transport invariance)**.** *Let*

$$F_0 = (B, S, W, D, R, E, H, C, P, N)$$

and

$$F_1 = (B, S, W, D, R, E, H, C, P, N)$$

be accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets whose terminal rows agree componentwise in the budget row* B*, schedule row* S*, finite* $\mathsf{StreamName}^{\uparrow}$ *window family* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transports* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. Then every cofinal terminal consumer compatible with* F_0 *is compatible with* F_1*, and conversely, and both compatibilities factor through the same displayed route*

$$B \to S \to W \to D \to R \to E.$$

No quotient stream, host real equality, choice operation, hidden selector, detached completion limit, or ambient completeness oracle can distinguish the two packets at a cofinal terminal consumer.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal cofinal window transport invariance`

Proof. First apply terminal cofinal-window coverage, Definition 782.36, to F_0 and to F_1. Since the two packets expose the same displayed rows $B, S, W, D, R, E, H, C, P, N$, every compatible cofinal terminal consumer on either side is already forced onto the same finite route through the selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Next identify the terminal sections. The terminal section exactness theorem Definition 782.34 turns compatibility with the displayed rows into the finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now use terminal section pullback uniqueness, Definition 782.35. The two sections have the same budget, schedule, window, dyadic, regular, seal, transport, continuation, provenance, and local naming rows, so they determine one route and one terminal seal under the recorded componentwise hsame transports.

Finally invoke Definition 126.586. The four-face pullback has no additional quotient, host-equality, choice, hidden-selector, detached-limit, or ambient-completeness coordinate. Thus transport along the shared rows carries cofinal terminal compatibility in either direction and leaves the route $B \to S \to W \to D \to R \to E$ invariant. □

Theorem 782.38 (Finite observation-budget selector terminal cofinal-window section decomposition)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every cofinal terminal consumer compatible with* F *decomposes into the four displayed section faces: the finite* $\mathsf{StreamName}^{\uparrow}$ *window family* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The only transport data between these faces are the componentwise* hsame *rows* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*.*

Equivalently, the consumer route is the finite section

$$W \to D \to R \to E$$

inside the budget and schedule envelope B, S*. It exports no quotient stream, host real equality, choice row, hidden selector, detached limit, metric-completeness row, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal cofinal window section decomposition`

Proof. First apply terminal cofinal-window coverage, Definition 782.36. It places every compatible cofinal terminal consumer on the displayed route from the finite budget and selected schedule through W, D, R to E.

Next use terminal section exactness, Definition 782.34. This identifies the compatible terminal section with the finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now apply terminal section pullback uniqueness, Definition 782.35. The rows $B, S, W, D, R, E, H, C, P, N$ determine a single budget-to-seal route, so any compatible consumer has the same four-face section decomposition.

Finally use Definition 126.586. Its finite-section pullback gives the displayed $\mathsf{StreamName}^{\uparrow}$ window face, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and $\mathsf{Real}^{\uparrow}$ seal face, while the non-escape clauses in the cited theorems exclude quotient streams, host real equality, choice rows, hidden selectors, detached limits, metric-completeness rows, and ambient completeness. □

Theorem 782.39 (Finite observation-budget selector terminal route status readback)**.** *Every terminal cofinal consumer of an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet reports its route status only through the displayed budget, schedule, finite window, dyadic ledger, regular-rational readback, Real seal, transport, continuation, provenance, and local naming rows*

$$B, S, W, D, R, E, H, C, P, N.$$

No quotient stream, hidden selector, host real equality, metric-completeness row, countable choice row, or ambient completeness oracle is a status coordinate for the same terminal route.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal route status readback`

Proof. First apply terminal cofinal-window coverage. It places every compatible terminal consumer on the displayed route from budget and schedule through the window, dyadic, regular, and Real-seal faces.

Next use terminal cofinal-window transport invariance. The status readback is preserved by the componentwise hsame rows, the displayed continuation row, the package provenance row, and the local naming row, so transport cannot create a second terminal status.

Finally apply terminal cofinal-window section decomposition. The route decomposes into the four displayed faces W, D, R, E inside the envelope B, S, H, C, P, N, and the four-face pullback exactness excludes quotient, hidden-selector, choice, host-equality, metric-completeness, and ambient-completeness coordinates. □

Theorem 782.40 (Finite observation-budget selector cofinal window exhaustion)**.** *For an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, the cofinal terminal route is exhausted by the rows* $B, S, W, D, R, E, H, C, P, N$*. No quotient stream, hidden selector, countable choice row, host real equality, metric-completeness row, or ambient completeness oracle can refine the same route while preserving the displayed terminal window.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier cofinal window exhaustion`

Proof. Begin with terminal window minimality. Removing a displayed row loses terminal endpoint determination, so a refinement of the same route must preserve the visible budget, schedule, window, dyadic, regular, seal, transport, continuation, provenance, and naming rows.

Use terminal route status readback. Any compatible terminal consumer reports status through those rows alone, so a hidden selector, quotient stream, or choice row would be an extra status coordinate rather than a refinement of the same route.

Finally invoke terminal cofinal-window section decomposition and the four-face pullback exactness. The four faces W, D, R, E already determine the terminal section over B, S, H, C, P, N, leaving no host real equality, metric-completeness row, or ambient completeness oracle to refine the cofinal window. □

Theorem 782.41 (Finite observation-budget selector terminal-window nonescape)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Every terminal Real-completion consumer that preserves the displayed budget, schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, Real seal,* hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row has no additional status coordinate for quotient streams, hidden selectors, host real equality, metric-completeness data, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal window nonescape`

Proof. First use terminal-window minimality, Definition 782.32. It says that the displayed rows $B, S, W, D, R, E, H, C, P, N$ are sufficient for terminal endpoint determination and that dropping any displayed row loses that determination.

Next apply terminal route status readback, Definition 782.39. A compatible terminal consumer reports status only through the same rows, so any quotient stream, hidden selector, host equality, metric-completeness row, choice row, or ambient-completeness oracle would be an additional status coordinate.

Use cofinal-window exhaustion, Definition 782.40, to rule out refinements of the same terminal route. The route is already exhausted by the displayed packet.

Finally apply the four-face terminal correspondence, Definition 782.42. The section faces W, D, R, E assemble the terminal consumer over B, S, H, C, P, N, leaving no further terminal-window escape coordinate. □

Theorem 782.42 (Finite observation-budget selector four-face terminal correspondence)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. A terminal consumer is compatible with* F *exactly when it is given by the four displayed terminal section faces* W, D, R, E *together with the shared budget, schedule, transport, continuation, provenance, and local naming rows* B, S, H, C, P, N*. Thus the public terminal route is precisely the finite observation-budget passage*

$$B, S \to W \to D \to R \to E$$

and the correspondence exports no quotient stream, host real equality, choice row, detached completion limit, metric-completeness row, or ambient completeness oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier four face terminal correspondence`

Proof. Start with terminal cofinal-window section decomposition, Definition 782.38. It sends every compatible terminal consumer to the four displayed faces W, D, R, E inside the shared envelope B, S, H, C, P, N.

For the reverse direction, apply terminal section exactness, Definition 782.34. A section assembled from W, D, R, E and the shared rows is already a terminal section over the accepted packet, so it gives a compatible terminal consumer.

The two constructions are inverse on the displayed rows by terminal section pullback uniqueness, Definition 782.35. No second terminal section can have the same budget, schedule, window, dyadic, regular, seal, transport, continuation, provenance, and naming rows.

Finally use Definition 126.586. The underlying finite $\mathsf{StreamName}^{\uparrow}$ section has no additional quotient, host-equality, choice, detached-limit, metric-completeness, or ambient-completeness coordinate, so the four faces are exactly the terminal route. □

Theorem 782.43 (Finite observation-budget selector terminal cofinal-window route reflection)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any terminal consumer whose four displayed terminal faces are the same* W, D, R, E *reflects to the same finite public route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Any claimed terminal read outside the displayed faces W, D, R, E *is not a refinement of the same packet.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal cofinal window route reflection`

Proof. Apply terminal cofinal-window coverage, Definition 782.36. The accepted packet exhausts the terminal finite observation window through the displayed $\mathsf{StreamName}^{\uparrow}$ row and its dyadic, regular-sequence, and Real-seal faces.

Next use terminal cofinal-window section decomposition, Definition 782.38. A terminal consumer over the packet decomposes into exactly the four faces W, D, R, E inside the shared envelope B, S, H, C, P, N.

Route status readback, Definition 782.39, identifies every terminal status claim with those same displayed rows. Terminal-window nonescape, Definition 782.41, then excludes any additional terminal coordinate.

Finally apply the four-face terminal correspondence, Definition 782.42. The common faces W, D, R, E determine the same $\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ route, while any read outside those faces asks for quotient streams, hidden selectors, host real equality, metric-completeness rows, countable choice, or ambient completeness rather than a refinement of F. □

Theorem 782.44 (Finite observation-budget selector terminal budget exhaustion)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. No terminal consumer route reaching the displayed* Real$^{\uparrow}$ *seal* E *can read outside the budget row* B*, selected schedule row* S*, finite* StreamName$^{\uparrow}$ *window family* W*,* DyadicRatCore$^{\uparrow}$ *ledger* D*,* RegSeqRat$^{\uparrow}$ *handoff* R*, componentwise* hsame *transports* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. Thus the terminal budget is exhausted exactly by the route*

$$B, S \to W \to D \to R \to E.$$

The exhausted surface has no quotient stream, host real equality, countable choice row, hidden selector, detached completion limit, metric-completeness row, or ambient completeness oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal budget exhaustion`

Proof. First apply terminal window minimality, Definition 782.32. It identifies the complete terminal inventory of an accepted packet and says that dropping any displayed row removes the endpoint exactness supplied by the terminal route.

Next attach diagonal-tail coverage, Definition 782.33. This keeps every Real-completion and diagonal consumer on the same finite chain through B, S, W, D, R, E and the same transport, continuation, provenance, and local naming rows.

Now use terminal section exactness, Definition 782.34. A consumer that reaches the terminal seal through the displayed rows is exactly the finite section pulled back along StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Finally combine terminal section pullback uniqueness with the four-face terminal correspondence, Definition 782.35 and Definition 782.42. These results leave no second terminal route with the same rows and no additional quotient, host-equality, choice, hidden-selector, detached-limit, metric-completeness, or ambient-completeness coordinate. □

Theorem 782.45 (Finite observation-budget selector cofinal route choice freeness)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Every cofinal terminal route compatible with* F *is determined by the displayed budget, schedule, finite window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and local naming rows. In particular, cofinal compatibility requires no independent choice of terminal route, no choice of a hidden tail selector, and no choice of an external completion witness beyond the four faces* W, D, R, E *inside the envelope* B, S, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier cofinal route choice freeness`

Proof. Start from terminal cofinal-window coverage, Definition 782.36. It forces any compatible cofinal consumer onto the displayed finite route from the selected budget and schedule through W, D, R to E.

Next decompose the route by Definition 782.38. The consumer is exactly the four-face section with StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ faces and with transport supplied only by H, C, P, N.

The forward and reverse readings of compatibility are supplied by the four-face terminal correspondence, Definition 782.42. A consumer compatible with F gives the displayed four faces, and the displayed four faces with the shared rows give a compatible terminal consumer.

Finally use terminal section pullback uniqueness, Definition 782.35. Since no second section can carry the same displayed rows, there is no remaining route-choice parameter, hidden selector, external completion witness, quotient stream, host equality, or ambient completeness oracle. □

Theorem 782.46 (Finite observation-budget selector terminal readiness correspondence)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. The following terminal-readiness data are equivalent as public Real-and-Completion handoff surfaces:*

1. *a terminal consumer route compatible with* F*;*
2. *the exhausted finite budget route* $B, S \to W \to D \to R \to E$ *together with the displayed* H, C, P, N *rows;*
3. *the four-face terminal section* W, D, R, E *inside the same budget, schedule, transport, continuation, provenance, and local* $\mathsf{NameCert}$ *envelope.*

Under this correspondence, later $\mathsf{Real}^{\uparrow}$*, diagonal-completion, and selector-seal consumers receive exactly the displayed budget-readiness rows and no quotient stream, host real equality, countable choice operation, hidden selector, detached completion limit, metric-completeness row, or ambient completeness oracle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal readiness correspondence`

Proof. First pass from a compatible terminal consumer to the exhausted route. By Definition 782.44, any terminal route reaching E is already confined to the displayed budget, schedule, window, dyadic, regular, seal, transport, continuation, provenance, and local naming rows.

Next pass from the exhausted route to the four-face section. The terminal cofinal-window section decomposition theorem, Definition 782.38, rewrites the route as the four faces W, D, R, E inside the envelope B, S, H, C, P, N.

For the converse direction, use the four-face terminal correspondence, Definition 782.42. The four displayed faces together with the shared rows assemble exactly a compatible terminal consumer.

It remains to show that the correspondence is the handoff surface seen by downstream consumers. Terminal section exactness and pullback uniqueness, Definition 782.34 and Definition 782.35, identify all three descriptions with the same finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally, cofinal route choice freeness, Definition 782.45, removes any extra route parameter. Hence $\mathsf{Real}^{\uparrow}$, diagonal-completion, and selector-seal consumers inherit only the displayed budget-readiness rows and none of the excluded quotient, host-equality, choice, hidden-selector, detached-limit, metric-completeness, or ambient-completeness data. □

Theorem 782.47 (Finite observation-budget selector terminal four-face pullback lock)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. The terminal Real-and-Completion consumer surface of* F *is locked by the four displayed faces* W, D, R, E *in the following sense:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-section pullback reads exactly the selected finite window family* W*;*

2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face reads exactly* D*;*

3. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face reads exactly* R*;*

4. *the* $\mathsf{Real}^{\uparrow}$ *terminal seal face reads exactly* E*.*

Every $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, diagonal-completion, or selector-seal consumer compatible with the selector must factor through exactly this route*

$$B, S \to W \to D \to R \to E$$

with transport, continuation, provenance, and local naming supplied only by H, C, P, N*. The locked surface contains no quotient stream, host real equality, countable choice operation, hidden tail selector, detached completion witness, metric-completeness row, or ambient completeness oracle.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal four face pullback lock`

Proof. First identify the public readiness surface. The terminal readiness correspondence, Definition 782.46, equates a compatible terminal consumer, the exhausted route $B, S \to W \to D \to R \to E$, and the four-face section W, D, R, E inside the shared H, C, P, N envelope.

Now pin the four faces as a pullback rather than as a loose list. Terminal section exactness, Definition 782.34, identifies any compatible terminal section with the finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Terminal section pullback uniqueness, Definition 782.35, leaves no second section with the same displayed rows.

Next read the StreamName-facing side through the ambient finite-section theorem. By Definition 126.586, the selected finite windows, dyadic ledger, regular readback, and Real seal determine one four-face pullback. Therefore the first face reads W, the second reads D, the third reads R, and the fourth reads E.

Finally attach the open Real-and-Completion consumers. The UniformCauchyCriterion pullback, Definition 782.20, places any compatible uniform-Cauchy read on the same finite observation route, while the readiness correspondence already covers the $\mathsf{Real}^{\uparrow}$, diagonal-completion, and selector-seal handoff surfaces. Thus every such consumer factors through the displayed route and can export only $B, S, W, D, R, E, H, C, P, N$, with the quotient, host-equality, choice, hidden-selector, detached-witness, metric-completeness, and ambient-completeness coordinates absent. □

Theorem 782.48 (Finite observation-budget selector terminal cofinal schedule lock)**.** *Let*

$$F_0 = (B, S, W, D, R, E, H, C, P, N)$$

and

$$F_1 = (B, S', W, D, R, E, H, C, P, N)$$

be accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packets. Suppose both packets are compatible with the same terminal cofinal-window route and with the same four-face pullback through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Then every terminal consumer reads the same budget-to-seal surface from* F_0 *and* F_1*:*

$$B \to W \to D \to R \to E$$

with transport, continuation, provenance, and local naming supplied only by H, C, P, N*. In particular, the terminal consumer can read no distinction between* S *and* S'*. Any actual schedule*

separation must alter the displayed route before the terminal seal, not only a hidden scheduling coordinate behind the common four-face surface.

The locked surface contains no quotient stream, countable choice operation, hidden selector, detached completion limit, metric-completeness row, or ambient completeness oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal cofinal schedule lock`

Proof. First apply terminal cofinal-window coverage, Definition 782.36, to F_0 and to F_1. Each compatible terminal consumer is forced onto the displayed finite route through the $\mathsf{StreamName}^{\uparrow}$ window family W, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, and $\mathsf{Real}^{\uparrow}$ seal E.

Next use terminal cofinal-window transport invariance, Definition 782.37, on the rows shared by the two packets. The only row not literally shared is the schedule coordinate, and the hypothesis says that both schedules feed the same terminal cofinal-window route and the same four-face pullback. Transport therefore carries terminal compatibility from F_0 to F_1 and back without creating a second terminal read.

Now exhaust the terminal budget. By Definition 782.44, any terminal consumer that reaches E is already confined to the displayed budget, finite window, dyadic ledger, regular handoff, seal, transport, continuation, provenance, and local naming rows. Cofinal route choice freeness, Definition 782.45, removes any independent route choice or hidden selector that could record a separate schedule decision.

Finally apply the terminal four-face pullback lock, Definition 782.47. Since both packets present the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces, the terminal consumer sees the same locked surface. Thus a real schedule separation must appear before the terminal seal as a displayed route change; it cannot be stored only in a hidden scheduling coordinate.

The non-escape clauses in the cited theorems exclude quotient streams, countable choice, hidden selectors, detached completion limits, metric-completeness rows, and ambient completeness oracles from the terminal surface. □

Theorem 782.49 (Finite observation-budget selector terminal tail-meet synchronization)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any terminal consumer that is compatible both with the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *cofinal-window route and with either a* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-tail route or a* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *cofinal schedule must read the same L10 terminal route*

$$B, S \to W \to D \to R \to E.$$

Equivalently, the tail-meet consumer, the cofinal-schedule consumer, and the terminal selector consumer all factor through the same four displayed faces W, D, R, E, with transport, continuation, provenance, and local naming supplied only by H, C, P, N. The synchronized surface cannot introduce a second tail-meet threshold, hidden cofinal schedule, quotient stream, host real equality, choice row, detached limit, metric-completeness row, or ambient completeness oracle.

Proof. First place the selector-side consumer on the terminal route. Terminal cofinal-window coverage, Definition 782.36, forces every compatible terminal read through the displayed finite window, dyadic ledger, regular-rational readback, and Real seal faces. The terminal four-face pullback lock, Definition 782.47, then identifies those faces as exactly W, D, R, E over the route $B, S \to W \to D \to R \to E$.

Next attach the schedule side. The terminal cofinal schedule lock, Definition 782.48, says that two accepted schedules with the same terminal cofinal-window route and the same four-face pullback give the terminal consumer no visible schedule distinction. Thus a $\mathsf{TailCofinalitySchedule}^{\uparrow}$ consumer compatible with F reads the same schedule-facing route rather than a hidden cofinal schedule behind it.

Now attach the shared-tail side. The RegularCauchyTailMeet budget-selector handoff, Definition 560.89, requires every $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ consumer of an accepted shared-tail packet to consume the displayed TailMeet route before selecting a downstream budget. The root budget route lock, Definition 560.91, leaves no second downstream budget or independent tail oracle available to the same consumer.

Combining these route locks gives determinacy at the terminal selector surface. Both the TailMeet-compatible read and the cofinal-schedule-compatible read reach the same selector packet F, and F has only the four terminal faces W, D, R, E after B, S. Therefore the common consumer factors through the displayed route $B, S \to W \to D \to R \to E$.

The non-escape clauses in the cited selector and TailMeet results remove every unlisted resource. Hence no second tail-meet threshold, hidden cofinal schedule, quotient stream, host real equality, choice row, detached limit, metric-completeness row, or ambient completeness oracle can be added at the synchronized terminal surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal tail meet synchronization`

Theorem 782.50 (Finite observation-budget selector terminal window seal inversion)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. Any terminal consumer exposing the same four faces* W, D, R, E *and the same budget, schedule, transport, continuation, provenance, and local naming rows* B, S, H, C, P, N *determines the same Real seal route*

$$B, S \to W \to D \to R \to E.$$

Conversely, any accepted terminal Real-seal route from B, S *to* E *decomposes into those four displayed faces* W, D, R, E *inside the same envelope. The correspondence exports no quotient stream, host real equality, countable choice operation, detached limit, metric-completeness row, or ambient completeness oracle.*

Proof. For the forward direction, begin with terminal cofinal-window coverage. Definition 782.36 places every compatible terminal consumer on the displayed finite route through the window, dyadic, regular, and Real-seal faces.

Terminal section exactness and terminal section pullback uniqueness then remove any second route. By Definition 782.34, the compatible consumer is the finite section pulled back along $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; by Definition 782.35, the shared rows $B, S, W, D, R, E, H, C, P, N$ determine a single terminal section.

For the reverse direction, read an accepted Real-seal route through the four-face surface. The terminal four-face pullback lock, Definition 782.47, pins the four displayed faces as W, D, R, E, and terminal tail-meet synchronization, Definition 782.49, keeps tail-meet and cofinal-schedule consumers on the same L10 route.

Thus terminal routes and compatible terminal windows determine one another over the displayed packet. The non-escape clauses in the cited theorems exclude quotient streams, host real

equality, countable choice, detached limits, metric-completeness rows, and ambient completeness from either direction of the correspondence. □

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal window seal inversion`

Theorem 782.51 (Finite observation-budget selector terminal window choice-free section)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Every terminal cofinal-window section accepted by F is determined by the displayed finite selector row S, the shared tail-meet route, and the Real-seal route*

$$B, S \to W \to D \to R \to E.$$

It is not determined by a hidden choice of subsequence, a host-selected limit, a quotient stream, host real equality, a detached completion point, a metric-completeness row, or an ambient completeness oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier terminal window choice free section`

Proof. Start with terminal cofinal-window coverage, Definition 782.36. It places every compatible terminal consumer on the displayed finite route through the window, dyadic ledger, regular handoff, and Real seal.

Apply the pullback lock, Definition 782.47. The four terminal faces are exactly W, D, R, E, so the terminal section is already read through the packet rows $B, S, W, D, R, E, H, C, P, N$.

Next use the cofinal schedule lock and tail-meet synchronization, Definition 782.48 and Definition 782.49. These results identify the schedule-facing consumer and the shared-tail consumer with the same route rather than with a hidden subsequence choice.

Finally invoke window seal inversion, Definition 782.50. The Real-seal route and the terminal cofinal window determine one another over the displayed packet. Therefore the terminal section is choice-free in the stated sense: it is fixed by S, the tail-meet route, and the Real-seal route, and it has no hidden subsequence, host limit, quotient, host equality, detached completion, metric-completeness, or ambient-completeness coordinate. □

Theorem 782.52 (Finite observation-budget selector budget-prefix pullback inversion)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Suppose a terminal four-face section is compatible with F, its finite budget prefix and selected-schedule prefix agree with the displayed rows B, S, and its terminal seal agrees with E through the displayed transport, continuation, provenance, and local naming rows H, C, P, N. Then that terminal section is forced to be the finite route*

$$B \to S \to W \to D \to R \to E.$$

Conversely, a section that passes through the displayed rows $B, S, W, D, R, E, H, C, P, N$ is exactly a compatible terminal four-face section for F.

This inversion exports only the accepted budget prefix, selected schedule, window family, dyadic ledger, regular readback, Real seal, componentwise hsame *transports,* Cont *route,* Pkg *provenance, and local* NameCert *row. It excludes quotient streams, host real equality, choice rows, hidden selectors, detached completion limits, metric-completeness rows, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier budget prefix pullback inversion`

Proof. For the forward direction, begin with the agreed budget and schedule prefix. Terminal budget exhaustion, Definition 782.44, confines any compatible terminal route reaching E to the displayed packet rows, so the prefix inventory cannot be enlarged beyond B, S and the rows they expose.

Next bind the four terminal faces. The terminal four-face pullback lock, Definition 782.47, identifies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces as W, D, R, E inside the same H, C, P, N envelope.

Now recover the window from the seal-facing read. Terminal window seal inversion, Definition 782.50, says that the Real-seal route and the terminal cofinal window determine one another over the displayed packet. Since the seal agrees with E through H, C, P, N, the window-facing section is the displayed W rather than a detached completion-side choice.

Finally remove hidden selection. Terminal window choice-free section, Definition 782.51, fixes the terminal cofinal-window section from S, the shared tail-meet route, and the Real-seal route. The forward section is therefore the finite route $B \to S \to W \to D \to R \to E$, with no quotient stream, host real equality, choice row, hidden selector, detached completion limit, metric-completeness row, or ambient completeness coordinate.

For the converse direction, read the displayed route as a terminal consumer surface. The four-face terminal correspondence, Definition 782.42, assembles the four displayed faces W, D, R, E together with the agreed rows B, S, H, C, P, N into a compatible terminal consumer for F.

The same correspondence identifies that consumer with the terminal four-face section it exposes. Hence any route through the displayed rows is exactly the compatible terminal four-face section, and the excluded quotient, host-equality, choice, hidden-selector, detached-limit, metric-completeness, and ambient-completeness data have no coordinate in either direction. □

Theorem 782.53 (Finite observation-budget selector selected-tail terminal pullback)**.** *Let*

$$F = (B_F, S_F, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let*

$$T = (Q, E_T, A_T, W_T, K_T, S_T, R_T, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{SelectedTailSeed}^{\uparrow}$ *packet. Suppose the selected finite-tail admission, window, readback, dyadic, and Real rows* A_T, W_T, K_T, S_T, R_T *agree with the terminal four faces* W_F, D_F, R_F, E_F *of* F *through the displayed componentwise* hsame *rows, the same* Cont *replay route, the same* Pkg *provenance, and the compatible local* $\mathsf{NameCert}_{\mathsf{FiniteObservationBudgetSelector}^{\uparrow}}$ *and* $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ *rows. Then every terminal consumer compatible with both packets factors through the single finite selector route*

$$B_F, S_F \to W_F \to D_F \to R_F \to E_F$$

and, under the selected-tail identification, through the finite-tail-to-Real route

$$A_T, W_T, K_T, S_T, R_T.$$

This pullback exposes only the accepted finite budget, selected schedule, terminal window, dyadic ledger, regular readback, Real seal, selected finite-tail admission, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It excludes quotient streams, host real equality, countable choice, detached completion limits, metric-completeness rows, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier selected tail terminal pullback`

Proof. Start on the terminal selector side. By terminal cofinal-window coverage, Definition 782.36, every compatible terminal consumer of F is already routed through the displayed finite terminal surface.

Now apply budget-prefix pullback inversion, Definition 782.52. Since the shared consumer agrees with the finite budget prefix B_F, S_F and the seal-facing row E_F through the displayed structural rows, the terminal selector section is forced to be the route $B_F, S_F \to W_F \to D_F \to R_F \to E_F$.

Turn to the selected-tail side. The finite-tail Real-boundary scope theorem, Definition 1207.20, identifies A_T, W_T, K_T, S_T, R_T as the exact finite-tail-to-Real boundary supplied by T. The Real-completion boundary theorem, Definition 1207.13, keeps that boundary finite and prevents it from being enlarged into an ambient completion principle.

The hypotheses identify the selected-tail rows with the terminal four faces through hsame, Cont, Pkg, and the two local NameCert rows. Therefore a consumer compatible with both packets has no second route: the selector route and the selected-tail boundary are the same finite BEDC handoff. The cited nonescape clauses remove quotient streams, host real equality, countable choice, detached completion limits, metric-completeness rows, and ambient completeness from the pullback surface. □

Theorem 782.54 (Finite observation-budget selector bounded-monotone diagonal synchronization)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Let M be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet whose convergence-seal route is terminal-compatible with F, and let L be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet reading the same terminal* Real$^{\uparrow}$ *and* CauchyLimitSeal$^{\uparrow}$ *surface. Then every downstream* Real$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, RegularCauchyTailSelector$^{\uparrow}$, *or diagonal-completion consumer compatible with F, M, and L factors through the single finite terminal route*

$$B, S \to W \to D \to R \to E$$

together with the common BHist, hsame, Cont, Pkg, *and* NameCert *support rows.*

The synchronized surface exports no quotient stream, selected host limit, host real equality, trichotomy, countable choice, hidden selector, detached completion limit, metric-completeness row, open-cover compactness, or ambient completeness oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier bounded monotone diagonal synchronization`

Proof. First bind the terminal surface supplied by F. Terminal cofinal-window coverage, Definition 782.36, places every compatible terminal read on the displayed finite window route, and the terminal four-face pullback lock, Definition 782.47, identifies the four terminal faces as W, D, R, E inside the same H, C, P, N envelope.

Next apply the budget-prefix inversion above. Since the compatible reads agree on the budget prefix B, S and the terminal seal E, Definition 782.52 forces the section to be exactly the route $B \to S \to W \to D \to R \to E$ rather than a route selected from outside the accepted packet.

Now attach the bounded-monotone consumer. The bounded-monotone selector terminal pullback and Real-seal route exhaustion, Definition 535.68 and Definition 535.70, route the convergence-seal and limit-seal consumers through the displayed witness rows and then through the same

StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert support rows.

Finally read the diagonal packet as a compatibility consumer of that same surface. The diagonal-limit bounded-monotone handoff and the existing compatibility pullback allow L to inspect only the terminal Real and limit-seal rows already exposed by F and M. It therefore adds no second terminal route and no independent completion-side coordinate.

Combining these route locks, every named downstream consumer factors through the single displayed terminal surface. The non-escape clauses in the finite-selector and bounded-monotone theorems exclude quotient streams, selected host limits, host real equality, trichotomy, countable choice, hidden selectors, detached completion limits, metric-completeness rows, open-cover compactness, and ambient completeness from the synchronized route. □

Theorem 782.55 (Finite observation-budget selector bounded-monotone diagonal Lean handoff)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Let M be an accepted* BoundedMonotoneCauchyWitness$^{\uparrow}$ *packet, and let L be an accepted* DiagonalLimitCompatibility$^{\uparrow}$ *packet. Suppose F, M, and L satisfy the terminal compatibility hypotheses of Definition 782.54: the bounded-monotone witness reads the terminal convergence-seal route of F, and the diagonal packet reads the same terminal* Real$^{\uparrow}$ *and* CauchyLimitSeal$^{\uparrow}$ *surface.*

Then the later Lean target may be stated as one BEDC route handoff over exactly the displayed finite-selector rows $B, S, W, D, R, E, H, C, P, N$, the bounded-monotone selector and real-seal rows, and the diagonal seal and readback rows. The handoff consumes no quotient stream, selected host limit, host real equality, countable choice, metric-completeness row, ambient completeness oracle, hidden selector, detached completion limit, or independent diagonal completion coordinate.

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier bounded monotone diagonal synchronization lean handoff`

Proof. First read the selector side. Terminal route status readback and terminal budget exhaustion, Definition 782.39 and Definition 782.44, force every compatible terminal read of F onto the displayed rows $B, S, W, D, R, E, H, C, P, N$.

Next attach the bounded-monotone witness. The bounded-monotone selector terminal pullback and Real-seal public route exhaustion identify the convergence-seal consumer of M with the same selector surface, so its selector and real-seal rows do not introduce a second host limit, quotient stream, or countable-choice coordinate.

Finally attach the diagonal packet. The diagonal-limit compatibility four-face terminal pullback and synchronized budget terminality theorems, Definition 646.135 and Definition 646.151, make L a reader of the same terminal seal and readback rows rather than a supplier of an independent completion oracle.

Combining the three route locks gives a single formalization-ready BEDC obligation: Lean must consume the finite-selector rows, the bounded-monotone seal rows, and the diagonal seal/readback rows as one route handoff, with every excluded host, quotient, choice, metric-completeness, ambient-completeness, and detached-limit coordinate absent from the statement. □

Theorem 782.56 (Finite observation-budget selector UniformCauchy Real readiness)**.** *Let*

$$F = (B, S, W, D, R, E, H, C, P, N)$$

be an accepted FiniteObservationBudgetSelector$^{\uparrow}$ *packet. Any Real-and-Completion consumer simultaneously compatible with the* UniformCauchyCriterion$^{\uparrow}$ *pullback, the bounded-monotone diagonal synchronization hypotheses, and the terminal four-face pullback lock factors through the same finite route*

$$B, S \to W \to D \to R \to E.$$

The public readiness surface exposes only the StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows carried by that route. It excludes quotient streams, host real equality, countable choice, hidden selectors, detached limits, metric-completeness rows, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.FiniteObservationBudgetSelectorUp.FiniteObservationBudgetSelectorCarrier uniform cauchy real readiness`

Proof. First place the uniform-Cauchy read on the selector route. The UniformCauchyCriterion pullback, Definition 782.20, identifies the shared-window, tolerance-ledger, regular-consumer, and seal rows with W, D, R, E over the finite selector path from B, S.

Next bind the terminal surface. The terminal readiness correspondence and terminal four-face pullback lock, Definition 782.46 and Definition 782.47, force every compatible terminal consumer to expose exactly the four faces W, D, R, E inside the displayed H, C, P, N envelope.

Now attach the bounded-monotone and diagonal side. The bounded-monotone diagonal synchronization theorem, Definition 782.54, makes the bounded-monotone witness and diagonal packet readers of the same Real$^{\uparrow}$ and CauchyLimitSeal$^{\uparrow}$ surface rather than suppliers of an independent completion coordinate.

Finally use the bounded-monotone diagonal Lean handoff, Definition 782.55, to package the three compatibility requirements as one route obligation. The only remaining public rows are the finite selector faces together with BHist, hsame, Cont, Pkg, and NameCert support. The non-escape clauses in the cited pullback, terminal, and synchronization results remove quotient streams, host real equality, countable choice, hidden selectors, detached limits, metric-completeness rows, and ambient completeness from the readiness surface. □

Closure status for NameCert$_{\text{FiniteObservationBudgetSelector}^{\uparrow}}$.

Constructive story: A FiniteObservationBudgetSelector$^{\uparrow}$ packet is generated from a precision budget row B, selected schedule row S, finite StreamName$^{\uparrow}$ window family W, DyadicRatCore$^{\uparrow}$ ledger D, RegSeqRat$^{\uparrow}$ handoff row R, Real$^{\uparrow}$ seal row E, componentwise hsame transports H, displayed Cont routes C, Pkg provenance P, and a local NameCert row N.

Theory closure: TheoryGate(NameCert$_{\text{FiniteObservationBudgetSelector}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 782.1, Definition 782.2, Definition 782.3, Definition 782.4, Definition 782.5, Definition 782.6, Definition 782.7, Definition 782.8, Definition 782.9, Definition 782.10, Definition 782.11, Definition 782.12, Definition 782.13, Definition 782.14, Definition 782.15, Definition 782.16, Definition 782.17, Definition 782.18, Definition 782.19, Definition 782.20, Definition 782.21, Definition 782.22, Definition 782.23, Definition 782.24, Definition 782.25, Definition 782.26, Definition 782.27, Definition 782.28, Definition 782.29, Definition 782.30, Definition 782.31, Definition 782.32, Definition 782.33, Definition 782.34, and Definition 782.36 bind the carrier, schedule admission, RealUp seal handoff, tail-meet seal compatibility, non-escape, route determinacy, shared-window union, dyadic-window exhaustion, diagonal consumer boundary, finite approximation pullback, cofinal readback, primitive scope, public export, budget-choice freeness, window-seal commutation, budget-ledger minimality, shared-tail pullback, CauchyLimitSeal pullback, UniformCauchyCriterion pullback, terminal budget nonemptiness, window index scope, terminal selector synchronization, RealTailAgreementSeal pullback, terminal tail-agreement limit-seal triangle, dyadic-to-regular budget-seal cut, RealCompletionSelectorSeal pullback, terminal route no-branching, terminal endpoint exactness, terminal diagonal pullback functoriality, Real-seal window saturation, terminal window minimality, diagonal-tail coverage, terminal section exactness, and terminal cofinal-window coverage for the displayed finite budget-to-seal selector packet.

Formal verification: FormalStatus(NameCert$_{\text{FiniteObservationBudgetSelector}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close quotient streams, host real equality, countable choice, hidden limit selectors, ambient completeness, or checked Lean verification.

Upgrade path: The theory closure is mature; remaining work is Lean verification and bridge checking for the same finite budget-to-seal surface.

Chapter 783

A Concrete Naming Certificate for $\mathsf{SubjectReductionBundleTrace}^{\uparrow}$

The $\mathsf{SubjectReductionBundleTrace}^{\uparrow}$ packet concretizes the explicit bundle discharge route of chapter 9374 as a finite BEDC trace. It reads the explicit discharge surface of chapter 718, the ledger discipline of chapter 720, and the finite continuation route shape of chapter 469. The chapter names the route packet; it does not inhabit the dependent-codomain discharge for the present closed-CIC substrate.

Definition 783.1 (SubjectReduction bundle trace carrier)**.** *A* $\mathsf{SubjectReductionBundleTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, E, L, H, C, P, N)$$

with the following visible coordinates:

1. *B is the four-field subject-reduction discharge bundle row, naming the same obligation surface read by* $\mathsf{SubjectReductionDischarge}^{\uparrow}$*;*
2. *E is the explicit extraction row that records direct bundle consumption;*
3. *L is the finite discharge ledger inherited from* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$*, with one row for each required subject-reduction discharge hypothesis;*
4. *H is the componentwise* hsame *transport comparing bundle fields, extraction rows, and ledger rows;*
5. *C is the displayed* Cont *route from the chosen discharge carrier to the subject-reduction theorem consumer, in the finite trace style of* $\mathsf{ContinuationTrace}^{\uparrow}$*;*
6. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* $\mathsf{NameCert}$*,* $\mathsf{SubjectReductionDischarge}^{\uparrow}$*,* $\mathsf{SubjectReductionDischargeLedger}^{\uparrow}$*, and the bundle route object;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionBundleTrace}^{\uparrow}}$ *row.*

The carrier exports only these finite rows. It contains no dependent-codomain proof, no host evaluator, and no second subject-reduction principle.

Lean4 (definition) `BEDC.Derived.SubjectReductionBundleTraceUp.SubjectReductionBundleTraceUp`

Theorem 783.2 (SubjectReduction bundle trace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SubjectReductionBundleTra}}$ *surface for Definition 783.1 consists of five finite obligations: carrier admission for the displayed*

packet, classifier stability under componentwise hsame *transport of* B, E, L, H, C, P, N*, route readback for direct bundle consumers, non-escape from dependent-codomain discharge evidence not present in the packet, and public export as the finite BEDC route packet for subject-reduction consumers.*

Proof. Carrier admission is the tuple displayed in Definition 783.1. Each coordinate is a finite BEDC row or a local package and naming row.

Classifier stability is componentwise. A transported packet may compare the bundle row, extraction row, discharge ledger, transport row, continuation route, provenance, and local name only through the displayed hsame and Cont data.

The remaining obligations are supplied by the route readback, non-escape, and public export theorems below. Together they account for every consumer read while keeping the packet on the bundle discharge surface described in chapter 9374. □

Theorem 783.3 (SubjectReduction bundle trace route readback)**.** *Every subject-reduction consumer of an accepted* SubjectReductionBundleTrace$^{\uparrow}$ *packet reads the bundle row* B *directly and consumes the four discharge rows in* L*. The public route factors through* H, C, P, N *and does not introduce a second discharge surface.*

Lean4 ✓ `BEDC.Derived.SubjectReductionBundleTraceUp.SubjectReductionBundleTraceCarrier route readback`

Proof. Project the carrier. The only discharge-bearing rows are the bundle row B, the extraction row E, and the finite ledger L.

For the direct route, the consumer reads B and then reads the four required discharge rows from L. The rows H and C only transport and route the displayed data, while P, N record provenance and local naming.

Thus the route terminates at the finite bundle ledger. No additional subject-reduction principle is available beyond the displayed packet. □

Theorem 783.4 (SubjectReduction bundle trace non-escape)**.** *An accepted* SubjectReductionBundleTrace$^{\uparrow}$ *packet cannot be used to construct a dependent-codomain discharge for the present closed-CIC substrate, a host evaluator, a termination predicate, a quotient of reduction behaviours, or a second subject-reduction theorem. Its public reads are exhausted by* B, E, L, H, C, P, N*.*

Proof. Read the carrier inventory. None of the displayed rows is a dependent-codomain proof, a host evaluator, a termination predicate, a quotient object, or a second theorem.

Apply the route readback theorem. The consumer follows the direct bundle route, which ends in the same four-row discharge ledger L and the same BEDC route and provenance data.

The obstruction boundary from chapter 9374 remains external to this packet. Since the carrier contains no row discharging that boundary for the present substrate, no public read can escape the finite route packet. □

Theorem 783.5 (SubjectReduction bundle trace public export)**.** *The public* SubjectReductionBundleTrace$^{\uparrow}$ *export is the finite packet of Definition 783.1 together with the NameCert obligations of Definition 783.2, the route readback of Definition 783.3, and the non-escape boundary of Definition 783.4. Downstream subject-reduction work may consume it as a BEDC-native route packet for bundle discharge, but not as an inhabitant of any discharge hypothesis not listed in* L*.*

Proof. The carrier supplies the finite rows B, E, L, H, C, P, N. The NameCert obligations bind carrier admission, classifier stability, route readback, non-escape, and public export to those rows.

The readback theorem accounts for direct bundle consumption. The non-escape theorem excludes the structures that would make the packet stronger than a route packet.

Repacking these clauses yields the public export. It is the concrete BEDC route packet for the bundle discharge interface, with the same four discharge rows visible and no hidden subject-reduction evidence outside the finite ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionBundleTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionBundleTrace}^{\uparrow}$ packet is generated from a carrier row, explicit NameCert obligations, route readback, non-escape, and public export over the finite bundle discharge trace.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionBundleTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 783.1, Definition 783.2, Definition 783.3, Definition 783.4, and Definition 783.5 bind the carrier, NameCert obligations, route readback, non-escape, and public export theorems.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionBundleTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close dependent-codomain discharge evidence, host evaluators, termination predicates, quotients, second subject-reduction principles, Lean verification, or a public bridge schema.

Upgrade path: Obligation closure requires checked carrier admission, route readback, non-escape, public export, componentwise transport, provenance, and local naming for the same finite bundle-trace packet.

Chapter 784

A Concrete Naming Certificate for $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$

The $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ packet gives the route vision of chapter 9374 a finite BEDC landing. It reads the socket rows of chapter 719 and sits beside the classifier packet of chapter 785: the classifier names the explicit bundle route being used, while the equivalence packet records that every accepted consumer-facing discharge read terminates at that bundle surface.

Definition 784.1 (Subject-reduction route equivalence carrier)**.** *A* $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, D, H, C, P, N)$$

with the following displayed rows:

1. *B is the bundle-route row, recording explicit construction of the four-field subject-reduction discharge bundle;*
2. *D is the bundle-consumption row, recording the finite handoff from the bundle row B to the socket surface;*
3. *H gives componentwise* hsame *transport for the four discharge coordinates, the bundle row, the consumer route, provenance, and local naming;*
4. *C gives the displayed* Cont *consumer routes through which downstream packets read the route agreement;*
5. *P gives* Pkg *provenance over* BHist, Ext, Cont, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifie}$ *the bundle route, componentwise* hsame, *and* $\mathsf{NameCert}$ *rows;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}}$ *row.*

The carrier contains no inhabitant of the four discharge hypotheses, no closed subject-reduction theorem, and no route equality outside the displayed finite rows.

Definition 784.2 (Subject-reduction route equivalence classifier)**.** *The* $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ *classifier compares only the rows displayed in Definition 784.1. Two accepted packets agree exactly when their bundle rows B, consumption rows D, componentwise* hsame *ledgers H, consumer-route rows C, provenance rows P, and local naming rows N agree through the displayed* BHist *coordinates.*

The classifier has no coordinate for host definitional equality between route implementations, no proof that the route is inhabited for the closed-CIC substrate, and no hidden discharge witness. Its entire comparison surface is the finite packet $S = (B, D, H, C, P, N)$.

Theorem 784.3 (SubjectReductionRouteEquivalence bundle-socket handoff)**.** *For every accepted* $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ *carrier* $S = (B, D, H, C, P, N)$*, the bundle route* B *feeds the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *surface through* D*. A consumer reaching the socket through* B *reads the four displayed discharge coordinates in bundle order through the componentwise transport rows in* H*.*

Proof. Project the carrier rows. The row B already has the bundle-shaped socket order.

Read the consumption row D. It is the only permitted route from B to the socket-facing bundle shape, so any consumer must enter the socket through that displayed handoff.

Finally use the transport rows H and the consumer routes C. They identify the four coordinates componentwise and replay them to the socket surface. Since the packet has no further route coordinate, every read terminates at the same displayed socket rows. □

Theorem 784.4 (SubjectReductionRouteEquivalence NameCert obligations)**.** *For the carrier of Definition 784.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the finite packet* $S = (B, D, H, C, P, N)$*;*
2. *row classification for the bundle row* B *and consumption row* D*;*
3. *componentwise stability through the* hsame *transport ledger* H*;*
4. *consumer exactness through the displayed* Cont *routes* C*;*
5. *package provenance and local naming through* P *and* N*.*

Proof. Carrier admission is the first projection of the displayed packet S. The next projection reads the route rows B, D and therefore supplies the complete classifier surface for bundle coordinates.

The transport obligation is exactly the row H. It compares the four discharge coordinates and the route metadata componentwise, without adding another discharge source.

The remaining public obligations are the consumer routes and provenance rows. The row C gives the only consumer reads, while P and N bind package origin and local naming. These five groups exhaust the carrier coordinates. □

Theorem 784.5 (SubjectReductionRouteEquivalence consumer non-escape)**.** *Every consumer read of an accepted* $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ *carrier receives route agreement only as the finite rows*

$$B, T, D, H, C, P, N.$$

The consumer may use these rows to compare bundle-route reads at the $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *surface. It cannot infer an inhabited discharge bundle, a closed subject-reduction theorem, host equality of route implementations, or any discharge evidence not present in the packet.*

Proof. Apply the NameCert-obligation theorem to identify every accepted read with one of the displayed carrier rows.

For a permitted route-agreement read, the consumer follows B directly to the socket or follows T through D to the same socket surface. The rows H and C transport and replay that finite comparison, and P, N record provenance and local naming.

For every stronger request, the packet has no coordinate to answer it. In particular, no row proves any of the four discharge hypotheses or exports a closed subject-reduction theorem. The consumer therefore remains confined to finite route agreement. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}$ packet is generated from bundle, consumption, componentwise hsame, displayed Cont, Pkg, and local NameCert rows over the subject-reduction discharge socket.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 784.1, Definition 784.2, Definition 784.3, Definition 784.4, and Definition 784.5 bind finite route agreement for the bundle subject-reduction discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not inhabit the discharge bundle, provide a closed subject-reduction theorem, verify the packet in Lean, or bridge it to a public schema.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, socket handoff, NameCert obligations, consumer non-escape, provenance, and local naming for the same finite route-equivalence packet.

Chapter 785

A Concrete Naming Certificate for $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$

The $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ packet realizes the route distinction of chapter 9374 as finite NameCert data. It reads the shared discharge rows from chapter 719 and makes them available to compiler consumers such as chapter 625 through an explicit route-kind ledger rather than through prose about explicit bundle invocation style.

Definition 785.1 (Subject-reduction route classifier carrier)**.** *A* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, A, L, P, K, I, U, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation discharge row;*
2. *A is the application-argument stability discharge row;*
3. *L is the lambda-domain subject-reduction discharge row;*
4. *P is the Pi-domain subject-reduction discharge row;*
5. *K is the route-kind row with the accepted classifier value* bundle*;*
6. *I is the invocation row, recording explicit bundle construction from B, A, L, P;*
7. *U is the consumer row, recording the downstream read that consumes the classified route as a subject-reduction discharge handoff;*
8. *H records componentwise* hsame *transports for the four discharge rows, the route-kind row, the invocation row, the consumer row, provenance, and the local naming row;*
9. *C records the displayed* Cont *routes by which consumers read the discharge rows, route-kind classifier, invocation evidence, consumer handoff, provenance, and naming coordinates;*
10. *Q is the* Pkg *provenance ledger over* BHist, Ext, Cont, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{TypePreservingCompiler}^{\uparrow}$*, the bundle route, componentwise* hsame*, and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteClassifier}^{\uparrow}}$ *row.*

The carrier contains no proof of any discharge row, no closed subject-reduction theorem, no host conversion rule, no erased derivation semantics, and no unrecorded route outside the displayed packet.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierTasteGate single carrier alignment`

Theorem 785.2 (SubjectReductionRouteClassifier NameCert obligations)**.** *For the carrier of Definition 785.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (B, A, L, P, K, I, U, H, C, Q, N),$$

the four discharge rows B, A, L, P*, the route-kind classifier* K*, the invocation row* I*, the consumer row* U*, componentwise transport through* H*, displayed continuation reads through* C*, package provenance through* Q*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier namecert obligations`

Proof. Project the displayed packet R. The first four coordinates are exactly the discharge rows already exposed by the socket chapter, so the classifier receives no extra MetaCIC premise.

Read K, I, U next. The route-kind row chooses the accepted route class, the invocation row records the corresponding construction of the same four-field bundle, and the consumer row records the downstream handoff.

Finally inspect H, C, Q, N. These rows provide transport, route replay, provenance, and local naming for the displayed coordinates. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 785.3 (SubjectReductionRouteClassifier canonical-bundle correspondence)**.** *In every accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *carrier, the* bundle *route kind classifies explicit invocation for the same four-row discharge packet* B, A, L, P*. The* bundle *row reads* I *as explicit construction of the four-field bundle. No route read changes the discharge payload.*

Proof. Start with the carrier projection. The four discharge rows B, A, L, P are fixed before the route-kind row is read.

For the bundle case, I records direct construction of the bundle from those four displayed rows. The consumer row U therefore receives exactly B, A, L, P through the explicit packet.

The route-kind row K has no third accepted classifier value, and the carrier has no coordinate for replacing B, A, L, P. Hence the accepted route kind terminates at the same discharge payload. □

Theorem 785.4 (SubjectReductionRouteClassifier consumer non-escape)**.** *Every consumer read of an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *carrier factors through*

$$B, A, L, P, K, I, U, H, C, Q, N.$$

The consumer may inspect the four discharge rows, the route-kind classifier, invocation evidence, consumer handoff, componentwise hsame *transports, displayed* Cont *routes, package provenance, and local* NameCert *row. It cannot read a closed subject-reduction theorem, a proof of any missing discharge row, host conversion, erased derivation semantics, quotient equality of MetaCIC terms, or a route not named by* K*.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier consumer nonescape`

Proof. Apply the NameCert-obligation theorem to identify the accepted carrier with the finite rows $B, A, L, P, K, I, U, H, C, Q, N$.

For a permitted route read, the consumer first reads K, then follows I to the bundle-facing form of the same four discharge rows, and finally reads the handoff recorded by U. The transport and route rows H, C preserve that path without creating another route kind.

For a request outside the classifier, the packet has no coordinate to satisfy it. In particular, no row supplies a closed subject-reduction theorem, proves a missing discharge premise, or exports host semantics. The consumer is therefore confined to the displayed classifier packet. □

Theorem 785.5 (SubjectReductionRouteClassifier public export)**.** *The public export of an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *carrier is exactly the finite route-classifier packet*

$$B, A, L, P, K, I, U, H, C, Q, N.$$

Downstream compiler and audit chapters may cite this export as a named classifier for choosing explicit bundle invocation over the same socket-level discharge rows, but not as an inhabitant of those rows and not as a closed subject-reduction theorem.

Proof. The carrier definition gives the complete public inventory. The NameCert-obligation theorem binds carrier admission, discharge-row reads, route classification, invocation evidence, consumer handoff, transport, routing, provenance, and local naming to that inventory.

The canonical-bundle correspondence theorem identifies accepted route reads as presentations of B, A, L, P. The consumer non-escape theorem then prevents any public read from leaving the displayed rows.

Thus the export is a finite classifier over the socket-level discharge surface. It can guide compiler and audit consumers to the chosen invocation style, while leaving discharge proofs and closed subject reduction outside the packet. □

Theorem 785.6 (SubjectReductionRouteClassifier vision concretization)**.** *The explicit bundle discharge route of chapter 9374 are realized by the finite* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *carrier of Definition 785.1. The realization consists of the four shared discharge rows* B, A, L, P*, the route-kind classifier* K*, the invocation row* I*, the consumer row* U*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. The route chapter identifies one way to deliver the same subject-reduction discharge payload: explicit bundle passing. The carrier records the invocation in K and records the corresponding invocation evidence in I.

The socket chapter supplies the finite shape of the shared discharge rows, while the compiler chapter supplies a downstream consumer of subject-reduction handoffs. The row U connects the classified invocation to that consumer-facing handoff.

The public-export theorem then packages the result as a finite NameCert classifier. The chapter therefore has a concrete BEDC landing: route prose is replaced by a displayed packet whose accepted public reads are exactly the rows of R. □

Theorem 785.7 (SubjectReductionRouteClassifier primitive scope)**.** *The primitive scope of an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *packet is exactly the displayed* BHist*,* Ext*,* Cont*,* hsame*,* Pkg*,* NameCert*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$*, and* $\mathsf{TypePreservingCompiler}^{\uparrow}$ *rows of*

$$B, A, L, P, K, I, U, H, C, Q, N.$$

The classifier introduces no host conversion rule, erased derivation semantics, quotient equality of MetaCIC terms, hidden route kind, or proof of any discharge row outside that primitive surface.

Proof. Start from Definition 785.1. Its displayed coordinates are the four discharge rows, the route-kind classifier, invocation evidence, consumer handoff, componentwise transport, continuation, package provenance, and local naming.

Apply Definition 785.2. The local NameCert surface reads carrier admission, the four discharge rows, route kind, invocation evidence, consumer row, transport, continuation, provenance, and name through those same coordinates.

Use Definition 785.3, Definition 785.4, and Definition 785.5. These identify explicit bundle invocation, keep public consumers on the displayed packet, and rule out any route outside the listed primitive rows.

Thus the primitive scope is exactly the finite BEDC surface named in the statement. □

Theorem 785.8 (SubjectReductionRouteClassifier scoped consumer exactness)**.** *Every compiler or audit consumer of an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *packet reads exactly the displayed route rows*

$$B, A, L, P, K, I, U, H, C, Q, N$$

over BHist, Ext, Cont, hsame, Pkg, $\mathsf{NameCert}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, *and* $\mathsf{TypePreservingCompiler}^{\uparrow}$. *The classifier may route the four discharge rows as bundle data, but it cannot treat itself as a proof of those four discharge rows.*

Proof. Begin with the local NameCert obligations and primitive scope. The obligation surface displays the carrier, route-kind row, invocation row, consumer handoff, transport, continuation, provenance, and local naming rows, while Definition 785.7 identifies these rows as the primitive packet surface.

Next use the canonical-bundle correspondence Definition 785.3. It says that the accepted route reads are presentations of the same payload B, A, L, P, not new discharge proofs.

Finally apply consumer non-escape and public export, Definition 785.4 and Definition 785.5. These theorems force every compiler or audit read to remain on the displayed public packet. Thus scoped consumers receive exactly the listed route rows and no proof of subject reduction, preservation, lookup, or substitution beyond the rows already supplied by the discharge socket. □

Theorem 785.9 (SubjectReductionRouteClassifier socket scope)**.** *The four discharge rows* B, A, L, P *in an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *packet are consumed from chapter 719. This classifier records how those rows are invoked and handed to consumers; it does not prove beta-substitution preservation, application-argument stability, lambda-domain subject reduction, or Pi-domain subject reduction.*

Proof. Project the carrier of Definition 785.1. The rows B, A, L, P are present as already supplied discharge rows, and the remaining rows classify and route them.

Apply Definition 785.2. The certificate obligations expose the four rows and then read K, I, U, H, C, Q, N as classifier, invocation, consumer, transport, continuation, provenance, and naming data.

Use Definition 785.4. Since no coordinate supplies proofs of missing discharge premises or closed subject reduction, the classifier cannot discharge the four rows on its own.

Therefore the rows are consumed from the socket chapter and only routed by this classifier. □

Theorem 785.10 (SubjectReductionRouteClassifier type-preserving scope)**.** *Every* $\mathsf{TypePreservingCompiler}^{\uparrow}$ *consumer of an accepted* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *packet may read the packet only as an invocation-style handoff for the shared rows* B, A, L, P. *The consumer may inspect whether the route is* bundle *and may follow* I *and* U *to the same discharge payload, but it receives no compiler correctness proof, erased operational semantics, host conversion rule, or additional subject-reduction discharge.*

Proof. Begin with Definition 785.3. The bundle route presents one four-row payload, so a compiler consumer cannot obtain different discharge data by changing route notation.

Apply Definition 785.5. The public surface exported to downstream chapters is exactly $B, A, L, P, K, I, U, H, C,$

Now use Definition 785.7 and Definition 785.9. These pin the primitive rows and record that the four discharge rows are consumed from the socket rather than proved by the classifier.

The type-preserving compiler read is therefore an invocation-style handoff only. No carrier coordinate remains from which compiler correctness, erased semantics, host conversion, or an additional subject-reduction discharge could be projected. □

Theorem 785.11 (SubjectReductionRouteClassifier audit handoff)**.** *Every audit or compiler consumer reading an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet receives only the classified invocation handoff over the same four socket rows B, A, L, P. The handoff may use the* bundle *presentation, but it cannot be read as a proof of subject reduction, preservation, lookup, substitution, erased semantics, or host conversion.*

Proof. Begin with the canonical-bundle correspondence theorem, Definition 785.3. It identifies the accepted presentation as invocation styles for the same four socket rows B, A, L, P.

Apply consumer non-escape and public export, Definition 785.4 and Definition 785.5. These confine every downstream read to $B, A, L, P, K, I, U, H, C, Q, N$ and prevent any public consumer from projecting a closed proof or host semantic row.

Finally use socket scope and type-preserving scope, Definition 785.9 and Definition 785.10. They identify B, A, L, P as socket-supplied discharge rows and restrict compiler-facing use to the classified handoff, so audit and compiler consumers receive no proof of subject reduction, preservation, lookup, substitution, erased semantics, or host conversion. □

Theorem 785.12 (SubjectReductionRouteClassifier route-kind coverage)**.** *In every accepted* SubjectReductionRout *packet, each route-kind row K consumed by the classifier is exactly* bundle*. A downstream route read cannot consume a second route kind, because the carrier has no coordinate, invocation row, continuation route, or package provenance row for any classifier value outside that case.*

Proof. Project the carrier of Definition 785.1. The row K is displayed with exactly one accepted classifier value, and the row I is displayed as the corresponding invocation evidence for that value.

For the bundle case, use Definition 785.3. The invocation row reads the four discharge rows B, A, L, P as an explicit bundle.

Finally apply Definition 785.7. Since the primitive surface has no hidden route-kind coordinate and no provenance row for another route, no third route can be consumed. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier route kind coverage`

Theorem 785.13 (SubjectReductionRouteClassifier invocation payload inversion)**.** *Let I be the invocation row of an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet. Reading I along the* bundle *route reconstructs the displayed discharge rows B, A, L, P by explicit bundle projection. Thus invocation inversion never changes the discharge payload consumed from the socket chapter.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier invocation payload inversion`

Proof. Start with the socket scope theorem, Definition 785.9. It records that B, A, L, P are already consumed from chapter 719 before this classifier routes them.

In the bundle branch, the invocation row is the explicit construction of the bundle over the displayed rows. Projecting that bundle returns B, A, L, P.

The inversion therefore reconstructs the same four socket rows in the accepted branch. It cannot produce another discharge row, because Definition 785.12 rules out any additional route kind. □

Theorem 785.14 (SubjectReductionRouteClassifier audit-handoff totality)**.** *Every audit consumer or* TypePreservingCompiler$^{\uparrow}$ *consumer of an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet reads exactly*

$$B, A, L, P, K, I, U, H, C, Q, N.$$

The handoff exposes the four socket discharge rows, the route-kind classifier, invocation row, consumer row, componentwise transports, continuation replay, package provenance, and local name; it exposes no additional proof row for the discharged subject-reduction premises.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier audit handoff totality`

Proof. Start from Definition 785.4. It confines every consumer read to the displayed classifier packet and rules out compiler correctness, erased semantics, host conversion, and unnamed routes.

Apply Definition 785.9 and Definition 785.10. These identify B, A, L, P as socket-supplied discharge rows and restrict compiler-facing use to invocation-style handoff data.

Now use Definition 785.12 and Definition 785.13. The route-kind row has only the accepted branch, and that branch reconstruct the same socket payload.

Finally, the primitive-scope theorem Definition 785.7 accounts for K, I, U, H, C, Q, N as the remaining displayed classifier, invocation, consumer, transport, continuation, provenance, and naming rows. No coordinate remains for another discharge proof. □

Theorem 785.15 (SubjectReductionRouteClassifier invocation row non-escape)**.** *In an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet, the invocation row I selects the* bundle *presentation of the same socket payload B, A, L, P. The row I never proves beta-substitution preservation, application-argument stability, lambda-domain subject reduction, Pi-domain subject reduction, or any discharge payload beyond those four socket rows.*

Proof. The canonical-bundle correspondence theorem Definition 785.3 gives the meaning of I: explicit bundle construction in the bundle branch.

Use Definition 785.13. The reading returns exactly B, A, L, P, so the selected presentation does not change the payload.

Apply Definition 785.12. Since no second route kind is admitted, I cannot select a branch whose payload is a discharge proof rather than a presentation of the socket rows.

The remaining non-escape clause follows from Definition 785.9. The four named preservation obligations belong to the socket surface, and this classifier records how they are invoked rather than proving them. □

Theorem 785.16 (SubjectReductionRouteClassifier socket-payload coverage)**.** *The four discharge rows consumed by every accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet are exactly the beta-substitution, application-argument, lambda-domain, and Pi-domain rows supplied by* SubjectReductionDischargeSock *The classifier creates none of those four rows; it only routes its bundle presentation to audit and* TypePreservingCompiler$^{\uparrow}$ *consumers.*

Proof. Begin with the socket scope theorem Definition 785.9. It states that B, A, L, P are consumed from chapter 719 and that this classifier does not prove the four preservation clauses.

Next apply Definition 785.10. A compiler consumer may inspect route style and follow I and U to the same payload, but it receives no additional subject-reduction discharge.

The route-kind coverage theorem and invocation-row non-escape theorem then close the coverage argument. The only accepted route choices are the bundle presentation, and the invocation row selects presentation style without manufacturing any socket row. □

Theorem 785.17 (SubjectReductionRouteClassifier consumer route totality)**.** *Every* TypePreservingCompiler$^{\uparrow}$ *or audit consumer handoff of an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet factors through*

$$B, A, L, P, K, I, U, H, C, Q, N.$$

The consumer receives the four socket discharge rows, the route-kind decision, invocation evidence, the handoff row, componentwise transport, continuation replay, provenance, and local naming data, and it receives no proof of a missing discharge premise.

Proof. First apply Definition 785.4. It confines every consumer read to the displayed rows $B, A, L, P, K, I, U, H, C, Q$ and excludes closed subject-reduction proofs, host conversion, erased derivation semantics, and routes not named by K.

Next use Definition 785.5. The public export is exactly the finite route-classifier packet, so any compiler or audit handoff has to pass through the displayed rows rather than through an external certificate.

The route-kind coverage and invocation payload inversion theorems identify all accepted route reads. When the consumer follows the bundle branch, the invocation row reconstructs the same socket payload B, A, L, P.

Finally invoke Definition 785.9 and Definition 785.10. The four discharge rows are consumed from the socket chapter and then handed to the compiler as invocation-style data, so no missing discharge premise becomes available to the consumer. □

Theorem 785.18 (SubjectReductionRouteClassifier obstruction-projection boundary)**.** *Every consumer of an accepted* SubjectReductionRouteClassifier$^{\uparrow}$ *packet that attempts to read a raw-relation subject-reduction bundle must also expose the checked no-bundle obstruction row. Projecting application-argument stability, lambda-domain preservation, or Pi-domain preservation from the raw relation lands in the obstruction ledger, not in a discharge proof. The classifier route may present the socket payload B, A, L, P through K, I, U, H, C, Q, N, but it never manufactures the four discharge fields and never converts the raw MetaCIC relation into* SubjectReductionDischargeBundle *data.*

Proof. Start with the consumer route totality theorem. It confines every compiler or audit read to $B, A, L, P, K, I, U, H, C, Q, N$ and states that no missing discharge premise is exported by the carrier.

Use route-kind coverage and invocation payload inversion. The route-kind row admits only the bundle presentation, and following that branch reconstructs the same socket payload. Thus changing the projection syntax cannot introduce a raw-relation discharge field.

Now apply socket-payload coverage. The four rows consumed by the classifier are exactly the rows supplied by SubjectReductionDischargeSocket$^{\uparrow}$. They are routed as finite payload data, not derived from the raw MetaCIC subject-reduction relation.

Finally read the MetaCIC obstruction ledger: the checked rows named by `not_appArgTypeStable`, `not_lamDomainSubjectReduction`, `not_piDomainSubjectReduction`, and `not_subjectReductionDischargeBun`

record that those raw projections do not yield the discharge interface on the present substrate. Therefore any raw-relation consumer must expose that no-bundle row, and the classifier remains a route over supplied socket rows rather than a source of discharge proofs. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteClassifier}\uparrow}$.

Constructive story: A $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ packet is generated from four subject-reduction discharge rows, a bundle route-kind classifier, invocation and consumer rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteClassifier}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 785.5, Definition 785.14, Definition 785.16, and Definition 785.17 bind the public route-classifier carrier, audit handoff, mature consumer-coverage surface, and finite consumer route for the packet $B, A, L, P, K, I, U, H, C, Q, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteClassifier}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SubjectReductionRouteClassifierUp.SubjectReductionRouteClassifierCarrier\_audit\_handof`

Bridge status: none

Origin: human

Not claimed: This chapter does not prove any of the four MetaCIC discharge rows, construct a closed subject-reduction theorem, verify the classifier in Lean, or bridge the packet to a public schema.

Upgrade path: No higher theory-closure grade remains; bridge checking remains on the verification axis.

Chapter 786

A Concrete Naming Certificate for $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$

The $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ packet realizes the closedness-preservation chain of chapter 9381 as finite BEDC NameCert data. It sits beside the main closed-normal consistency boundary of chapter 764, the confluence audit witness of chapter 745, and the beta-step boundary of chapter 754: those chapters consume closed endpoints, confluence rows, and beta-step rows, while this packet records how the five closedness-preservation theorems remain visible as a single replayable carrier.

Definition 786.1 (ClosurePreservationAuditWitness carrier)**.** *A* $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, V, F, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the shift-one closedness-preservation row;*
2. *V is the variable-substitution closedness-preservation row;*
3. *F is the full-substitution closedness-preservation row;*
4. *B is the beta-step closedness-preservation row;*
5. *R is the finite beta-star closedness-preservation row;*
6. *H records componentwise* hsame *transports for the five theorem rows, replay routes, provenance, and local naming;*
7. *C records the displayed* Cont *routes by which consumers replay the shift, substitution, beta-step, and beta-star rows without leaving the closedness classifier;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{NameCert}$, *the closure-preservation audit map, the closed-normal consistency main packet, the MetaCIC confluence audit witness, and the beta-step boundary;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditWitness}^{\uparrow}}$ *row.*

The carrier contains no subject-reduction theorem, no typing transport, no host conversion rule, no quotient equality of MetaCIC terms, and no route from closedness preservation to full closed consistency.

Theorem 786.2 (ClosurePreservationAuditWitness NameCert obligations)**.** *For the carrier of Definition 786.1, the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditWitness}\uparrow}$ *surface consists exactly of carrier admission for*

$$W = (S, V, F, B, R, H, C, P, N),$$

readback of the shift-one row S, variable-substitution row V, full-substitution row F, beta-step row B, and beta-star row R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Project the displayed packet W. The first five coordinates are the theorem rows named by the closure-preservation audit map, so carrier admission exposes exactly the closedness-preservation chain and no typing premise.

Read H and C next. Transport keeps each theorem row componentwise stable, while continuation replay lets a consumer reach the shift, substitution, beta-step, and beta-star rows in the same finite packet.

Finally inspect P and N. Provenance anchors the packet in the source audit map and in the sibling closed-normal, confluence, and beta-boundary chapters, while the local naming row names the carrier itself. Since no other coordinate is present, these rows exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditWitnessUp.ClosurePreservationAuditWitness namecert obligations`

Theorem 786.3 (ClosurePreservationAuditWitness beta-step row)**.** *In every accepted* ClosurePreservationAuditWit *carrier, the beta-step row B is read only as closedness preservation for a single MetaCIC beta contraction. Its permitted inputs are the closedness rows supplied by the shift and substitution coordinates S, V, F, transported through H and replayed through C. It does not export typing preservation for the contracted redex.*

Proof. Project W and isolate the beta-step coordinate B. The carrier places B after the shift and substitution rows, so the displayed read of B depends only on closedness budgets already present in the packet.

The transport row H preserves those coordinates componentwise, and the replay row C exposes the beta-step theorem through the same closedness classifier. Neither row contains a typing derivation or a codomain transport.

Thus B certifies the closedness boundary for one beta contraction and nothing stronger. □

Theorem 786.4 (ClosurePreservationAuditWitness beta-star row)**.** *In every accepted* ClosurePreservationAuditWit *carrier, the beta-star row R is the finite-path replay of the beta-step closedness row B. A consumer may cite R to keep the endpoint of a finite beta path inside the same closedness classifier, but may not cite it as Church–Rosser, normalization, subject reduction, or closed consistency.*

Proof. The carrier exposes B and R as separate rows. The beta-star row is consumed after the single-step row has been made available, and its replay route is recorded in C.

Componentwise hsame transport through H preserves the finite-path row without altering its classifier. The provenance row P points to the confluence and consistency siblings only as consumers of closed endpoints, not as consequences of the beta-star row.

Therefore R is exactly a finite beta-chain closedness row. It cannot be repacked as a confluence theorem, a normalization theorem, a typing-preservation theorem, or a consistency theorem. □

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditWitnessUp.ClosurePreservationAuditWitness beta star row`

Theorem 786.5 (ClosurePreservationAuditWitness substitution row)**.** *In every accepted* ClosurePreservationAuditW *carrier, the substitution surface consists of the variable-substitution row V and full-substitution row F together with the shift-one row S needed under binders. The packet exposes these rows as closedness-preservation data for MetaCIC syntax and does not add a typing-substitution lemma, dependent codomain transport, or host-level substitution semantics.*

Proof. Project the substitution coordinates of W. The shift row S supplies the binder-index motion, the variable-substitution row V supplies the local variable case, and the full-substitution row F supplies the recursive substitution read.

The transport and replay rows H, C keep these three rows available as closedness facts. They do not introduce a typing context, a displayed codomain, or a semantic substitution relation.

Hence the substitution surface is exactly the closedness-preservation surface named by S, V, F. □

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditWitnessUp.ClosurePreservationAuditWitness substitution row`

Theorem 786.6 (ClosurePreservationAuditWitness non-escape)**.** *Every consumer read of an accepted* ClosurePreservationAuditWitness$^{\uparrow}$ *carrier factors through*

$$S, V, F, B, R, H, C, P, N.$$

The consumer may inspect the five closedness-preservation theorem rows, their componentwise hsame *transports, displayed* Cont *replay routes,* Pkg *provenance, and local* NameCert *naming. It cannot extract a subject-reduction theorem, a typing conversion rule, a confluence diamond, a normalization theorem, a closed-consistency theorem, or any row not displayed by the carrier.*

Proof. Apply the NameCert-obligation theorem to identify the accepted carrier with the finite tuple $S, V, F, B, R, H, C, P, N$.

The beta-step theorem, beta-star theorem, and substitution theorem above describe the only theorem rows with downstream computational content. Each remains a closedness-preservation row and is replayed only through C after componentwise transport through H.

For every stronger request, inspect the carrier inventory. No coordinate supplies a typing derivation, a conversion rule, a confluence proof, a normalization proof, or a consistency proof. The consumer is therefore confined to the displayed closedness-preservation witness packet. □

Closure status for NameCert$_{\text{ClosurePreservationAuditWitness}^{\uparrow}}$.

Constructive story: A ClosurePreservationAuditWitness$^{\uparrow}$ packet is generated from shift-one preservation, variable substitution, full substitution, beta-step preservation, beta-star preservation, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosurePreservationAuditWitness}^{\uparrow}}$, seed)

Scope closed: Definition 786.1, Definition 786.2, Definition 786.3, Definition 786.4, Definition 786.5, and Definition 786.6 bind the finite closedness-preservation audit witness packet.

Formal verification: FormalStatus(NameCert$_{\text{ClosurePreservationAuditWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove subject reduction, typing substitution, confluence, normalization, closed consistency, Lean verification of the witness packet, or a public schema bridge.

Upgrade path: Obligation closure requires checked carrier admission, theorem-row readback, beta-step exactness, beta-star exactness, substitution exactness, replay routes, provenance, local naming, and non-escape for the same finite packet.

Chapter 787

A Concrete Naming Certificate for $\mathsf{SubjectReductionRoute}^{\uparrow}$

The $\mathsf{SubjectReductionRoute}^{\uparrow}$ packet turns the explicit bundle discharge route of chapter 9374 into one finite naming surface. It reads the socket-level discharge rows of chapter 719 and keeps the obstruction boundary of chapter 731 visible: the packet names the route by which a discharge would be handed to a consumer, but it does not manufacture any missing MetaCIC discharge premise.

Definition 787.1 (Subject-reduction route carrier)**.** *A* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, A, L, P, E, I, U, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation discharge row;*
2. *A is the application-argument stability discharge row;*
3. *L is the lambda-domain subject-reduction discharge row;*
4. *P is the Pi-domain subject-reduction discharge row;*
5. *E is the explicit bundle row whose coordinates are exactly B, A, L, P;*
6. *I is the invocation row recording that the consumer reads E directly;*
7. *U is the consumer handoff row for the downstream subject-reduction use site;*
8. *H records componentwise* hsame *transports for the discharge rows, bundle row, invocation row, consumer row, provenance row, and local naming row;*
9. *C records the displayed* Cont *routes by which the consumer reaches the same four discharge rows through the explicit invocation route;*
10. *Q is the* Pkg *provenance ledger over* BHist, Ext, Cont, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{SubjectReductionRoute}$ $\mathsf{TypePreservingCompiler}^{\uparrow}$, *componentwise* hsame, *and local* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for closed subject reduction, host conversion, erased derivation semantics, quotient equality of terms, or an additional route outside the displayed bundle pair.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteUp.SubjectReductionRouteTasteGate single carrier alignment`

Theorem 787.2 (SubjectReductionRoute NameCert obligations)**.** *For the carrier of Definition 787.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (B, A, L, P, E, I, U, H, C, Q, N),$$

the four discharge rows B, A, L, P*, explicit bundle packaging through* E*, invocation selection through* I*, consumer handoff through* U*, componentwise transport through* H*, displayed continuation reads through* C*, package provenance through* Q*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteUp.SubjectReductionRouteNameCertObligations`

Proof. Project the displayed tuple R. The first four coordinates are the socket-level subject-reduction discharge rows, while E is the presentation of that same four-row packet.

Read the invocation and consumer rows. The row I records direct bundle use, and U records the downstream handoff that consumes the chosen presentation.

Finally read H, C, Q, N. These rows supply transport, route replay, provenance, and local naming for the displayed coordinates. Since the carrier exposes no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 787.3 (SubjectReductionRoute bundle correspondence)**.** *In every accepted* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *carrier, the explicit bundle row* E *determines the discharge packet* B, A, L, P*. Bundle invocation changes none of the discharge rows, adds no new discharge premise, and removes no row required by the subject-reduction consumer.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteUp.SubjectReductionRouteBundleCorrespondence`

Proof. The carrier definition fixes B, A, L, P before the public route is inspected. The bundle row E packages those four rows directly.

The invocation row I records the route the consumer follows, and the consumer row U records the handoff after that route has reached B, A, L, P. There is no coordinate in the packet where the route can alter the discharge payload. □

Theorem 787.4 (SubjectReductionRoute obstruction non-escape)**.** *Every accepted* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *carrier preserves the obstruction boundary of chapter 731. A consumer may read the finite bundle route packet, but cannot extract a closed MetaCIC subject-reduction theorem, a proof of any missing discharge row, a host conversion principle, erased semantics, or quotient equality of terms from that packet.*

Proof. Apply the NameCert-obligation theorem to identify every permitted read with one of the displayed rows of R.

Rows B, A, L, P are discharge placeholders exposed through the socket surface, and rows E, I, U only determine how those rows are handed to a consumer. The transport, continuation, provenance, and naming rows certify this finite route; they do not add a theorem-producing coordinate.

The obstruction chapter remains untouched by this packet because the carrier contains no row for closing the dependent-codomain obstruction. Therefore no accepted read can escape from route naming into a closed subject-reduction result. □

Theorem 787.5 (SubjectReductionRoute public handoff)**.** *The public handoff of an accepted* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *carrier is exactly the finite route packet*

$$B, A, L, P, E, I, U, H, C, Q, N.$$

Downstream compiler and audit chapters may cite this handoff as a named route from bundle presentation to the same four discharge rows. They may not cite it as an inhabitant of those rows, as a closed subject-reduction proof, or as a bridge to host semantics.

Proof. The carrier definition lists the entire public inventory. The NameCert-obligation theorem binds carrier admission and every permitted read to that inventory.

The bundle correspondence theorem shows that the public route reaches the four-row discharge packet. The obstruction non-escape theorem then excludes reads that would turn the route packet into a closed MetaCIC theorem or host-semantic bridge.

Thus the public handoff is a finite naming certificate for route delivery only. It supplies a stable consumer-facing packet while keeping discharge inhabitation and bridge checking on their own axes. □

Theorem 787.6 (SubjectReductionRoute finite route surface)**.** *Every accepted* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *carrier binds the subject-reduction route classifier, compiler handoff, and discharge socket rows on the single finite path*

$$\mathsf{SubjectReductionRouteClassifier}^{\uparrow} \longmapsto \mathsf{TypePreservingCompiler}^{\uparrow} \longmapsto \mathsf{SubjectReductionDischargeSocket}^{\uparrow} \longmapsto (B, A, L,$$

The route first names the classifier row that selects the explicit bundle presentation, then names the type-preserving compiler handoff row through which the consumer reaches that bundle, and finally exposes the socket-level discharge rows B, A, L, P*. It exports no closed subject-reduction theorem, no additional discharge inhabitant, no host conversion principle, no erased semantics, and no bridge status.*

Proof. Project the carrier $R = (B, A, L, P, E, I, U, H, C, Q, N)$. The package provenance row Q names the $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, $\mathsf{TypePreservingCompiler}^{\uparrow}$, and $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ sources before the local naming row is read.

Apply Definition 787.3. The explicit bundle row E determines exactly the four discharge rows B, A, L, P, so the classifier can select only that displayed presentation.

Apply Definition 787.5. The compiler-facing consumer reaches the same four-row bundle through I and U, while H, C, Q, N transport, replay, source, and name the route. Since Definition 787.4 excludes closed discharge, host conversion, erased semantics, and bridge data, the stated finite path is the whole scoped surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionRoute}^{\uparrow}$ packet is generated from four subject-reduction discharge rows, an explicit bundle row, an invocation row, a consumer handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionRoute}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 787.1, Definition 787.2, Definition 787.3, Definition 787.4, Definition 787.5, and Definition 787.6 bind the carrier, NameCert obligations, bundle correspondence, obstruction non-escape, public handoff, and classifier–compiler–socket route surface for the finite route packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionRoute}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SubjectReductionRouteUp.SubjectReductionRouteTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove any MetaCIC discharge row, construct closed subject reduction, provide Lean verification, or bridge the packet to a public schema.

Upgrade path: Public export and bridge work require a public SubjectReductionRouteUp schema surface and a checked standard bridge for the same finite route packet.

Chapter 788

A Concrete Naming Certificate for MetaCICClosurePreservation$^{\uparrow}$

The MetaCICClosurePreservation$^{\uparrow}$ packet turns the ClosurePreservation audit map of chapter 9381 into a finite naming surface. It reads the substitution rows of chapter 779, the closed seal of chapter 531, the beta-redex handoff of chapter 716, the generator classifier of chapter 486, and the subject-reduction consumer boundary of chapter 483. The packet records closure-preservation data; it does not turn closedness into typing preservation.

Definition 788.1 (MetaCIC ClosurePreservation carrier)**.** *A* MetaCICClosurePreservation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, V, U, B, F, A, C, G, R, H, Q, P, N)$$

with the following displayed rows:

1. *S is the one-step shift closedness row for* `shift_one_preserves_closed_succ`*;*
2. *V is the variable-substitution closedness row for* `substitute_var_preserves_closed_at`*;*
3. *U is the full substitution closedness row for* `substitute_preserves_closed_at`*;*
4. *B is the one-step beta closedness row for* `betaStep_preserves_closed`*;*
5. *F is the finite beta-chain closedness row for* `betaStarStep_preserves_closed`*;*
6. *A is the* SubstitutionAuditMap$^{\uparrow}$ *row through which shift and substitution are read;*
7. *C is the* ClosedSubstitutionSeal$^{\uparrow}$ *row that keeps the closed substitution boundary visible;*
8. *G is the* GeneratorClosure$^{\uparrow}$ *classifier row for the authorized shift, substitution, beta-step, and beta-chain moves;*
9. *R is the downstream* SubjectReduction$^{\uparrow}$ *consumer row that may cite closedness preservation but still requires its own typing-discharge rows;*
10. *H stores componentwise* hsame *transports for the preservation rows, audit row, closed seal, generator classifier, consumer row, provenance row, and local naming row;*
11. *Q stores displayed* Cont *routes for reading each preservation row through the audit and generator-classifier surface;*
12. *P is the* Pkg *provenance over* `Term`*,* `ClosedAt`*, shift, substitution, beta-step, beta-chain,* SubstitutionAuditMap$^{\uparrow}$*,* ClosedSubstitutionSeal$^{\uparrow}$*,* BetaSubstitutionPreservation$^{\uparrow}$*,* GeneratorClosure$^{\uparrow}$*,* SubjectReduction$^{\uparrow}$*,* hsame*,* Cont*, and* NameCert *rows;*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservation}^{\uparrow}}$ *row.*
The carrier has no coordinate for `HasType` *preservation, conversion soundness, confluence, normalization, closed consistency, or an unlisted substitution calculus.*

Lean4 ✓ `BEDC.Derived.MetaCICClosurePreservationUp.MetaCICClosurePreservationTasteGate single carrier alignment`

Theorem 788.2 (MetaCIC ClosurePreservation NameCert obligations)**.** *For every accepted* MetaCICClosurePreserv *carrier*

$$M = (S, V, U, B, F, A, C, G, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservation}^{\uparrow}}$ *surface consists exactly of carrier admission, the five displayed closedness-preservation rows S, V, U, B, F, the substitution audit row A, the closed-substitution seal C, the generator-closure classifier G, the subject-reduction consumer row R, componentwise transport through H, displayed continuation through Q, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 788.1. The rows S, V, U, B, F are the five closure-preservation rows named by chapter 9381.

Read the dependency rows next. The substitution row A forces every shift or substitution read through chapter 779, the seal row C keeps closed substitution inside chapter 531, and the classifier row G records the authorized generator-closure surface.

Finally read R, H, Q, P, N. These rows identify the downstream subject-reduction consumer boundary, transport the displayed packet, replay its finite routes, record provenance, and name the local certificate. Since the carrier lists no further coordinate, the local NameCert obligations are exhausted by the displayed packet. □

Theorem 788.3 (Shift and substitution classifier handoff)**.** *In an accepted* $\mathsf{MetaCICClosurePreservation}^{\uparrow}$ *carrier, the shift row S, variable-substitution row V, and full-substitution row U are read only through the audit row A and the generator-classifier row G. A consumer cannot detach those rows from the* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *and* $\mathsf{GeneratorClosure}^{\uparrow}$ *surfaces or read them as an unrestricted substitution calculus.*

Proof. The carrier places S, V, U beside the audit row A and classifier row G. Thus every displayed continuation route for the syntactic movement rows is one of the routes recorded in Q from the same packet.

The audit row A gives the substitution readback boundary, while G supplies the generator-closure classifier for authorized movement under `ClosedAt`. Transport through H preserves those coordinates componentwise and cannot create a new movement rule.

An unrestricted substitution calculus would require a row outside $S, V, U, A, G, H, Q, P, N$. The carrier has no such row, so the handoff remains the displayed classifier handoff. □

Lean4 ✓ `BEDC.Derived.MetaCICClosurePreservationUp.MetaCICClosurePreservationCarrier shift substitution classifier handoff`

Theorem 788.4 (Beta-chain non-escape)**.** *For every accepted* $\mathsf{MetaCICClosurePreservation}^{\uparrow}$ *carrier, the beta-step row B and beta-chain row F export only closedness preservation under* `ClosedAt`*. They do not export* `HasType` *preservation, beta-substitution preservation for codomains, confluence, normalization, or closed consistency.*

Proof. Read B as the one-step closedness row and F as the finite beta-chain closedness row. Both rows are part of the same `ClosedAt` packet fixed in Definition 788.1.

The beta redex side may cite the sibling $\mathsf{BetaSubstitutionPreservation}^{\uparrow}$ packet through provenance, but the carrier records it only as a dependency boundary for the subject-reduction route. The actual exported rows remain B and F over closedness.

Confluence, normalization, typed preservation, and consistency would each require a separate row not present in M. Therefore the beta-chain surface preserves closedness without escaping into those stronger claims. □

Theorem 788.5 (ClosurePreservation ledger exactness)**.** *The public ledger of an accepted* MetaCICClosurePreservat *packet is exactly*

$$S, V, U, B, F, A, C, G, R, H, Q, P, N.$$

Downstream closed-normal and subject-reduction route work may cite this ledger for finite closedness preservation, but must separately supply any typing, conversion, confluence, normalization, or bridge rows it consumes.

Proof. The first theorem fixes the NameCert obligation surface for the packet. The shift-substitution theorem fixes the syntactic movement rows behind the audit and generator-classifier boundary.

The beta-chain theorem fixes the one-step and finite-chain rows as closedness-preservation data only. Thus the accepted public ledger is the carrier inventory itself.

Any downstream use that needs typing, conversion, confluence, normalization, or bridge data must cite a separate packet for that data. The $\mathsf{MetaCICClosurePreservation}^{\uparrow}$ ledger contributes the closedness-preservation side and no additional coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICClosurePreservation}^{\uparrow}$ packet is generated from shift, variable-substitution, full-substitution, beta-step, and beta-chain closedness rows, a substitution audit row, a closed-substitution seal, a generator-closure classifier, a subject-reduction consumer row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 788.1, Definition 788.2, Definition 788.3, Definition 788.4, and Definition 788.5 bind the finite ClosurePreservation packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close type preservation, conversion soundness, confluence, normalization, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shift and substitution classifier handoff, beta-chain non-escape, ledger exactness, provenance, transport, continuation, and local naming for the same finite packet.

Chapter 789

A Concrete Naming Certificate for $\mathsf{SrDischargeRouteAudit}^{\uparrow}$

The $\mathsf{SrDischargeRouteAudit}^{\uparrow}$ packet realizes the route discipline of chapter 9374 as a finite NameCert audit surface. It sits beside the discharge socket of chapter 719, the discharge packet of chapter 718, the obstruction packet of chapter 731, and the MetaCIC critical-path packet of chapter 765. Those sibling chapters name the payload and the blocked edge; the present chapter records how explicit bundle discharge is audited without turning the route into a closed subject-reduction proof.

Definition 789.1 (SrDischargeRouteAudit carrier)**.** *A* $\mathsf{SrDischargeRouteAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, S, L, P, R, O, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation discharge row;*
2. *S is the application-argument stability discharge row;*
3. *L is the lambda-domain subject-reduction discharge row;*
4. *P is the Pi-domain subject-reduction discharge row;*
5. *R is the route-audit row, recording the public read as explicit* $\mathsf{SubjectReductionDischargeBundle}$ *consumption;*
6. *O is the obstruction row inherited from the MetaCIC subject-reduction obstruction packet;*
7. *H records componentwise* hsame *transport for the four discharge rows, the route-audit row, the obstruction row, provenance, and local naming;*
8. *C records the displayed* Cont *reads from each admitted route to the same four-row discharge payload and to the obstruction boundary;*
9. *Q is the* Pkg *provenance over* BHist, hsame, Cont, $\mathsf{NameCert}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{SubjectReductionDischarge}^{\uparrow}$, $\mathsf{MetaCICSubjectReductionBoundaryUp}$, $\mathsf{SubjectReductionBundleTrace}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$*, and the MetaCIC critical-path packet of chapter 765;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SrDischargeRouteAudit}^{\uparrow}}$ *row.*

The carrier contains no fifth discharge row, no proof of the obstruction row, no erased derivation semantics, and no closed subject-reduction theorem.

Theorem 789.2 (SrDischargeRouteAudit bundle coherence)**.** *For an accepted* SrDischargeRouteAudit$^{\uparrow}$ *carrier, the explicit bundle route is coherent exactly when its public read terminates at the four discharge rows B, S, L, P and the obstruction row O. The route-audit row R records direct* SubjectReductionDischargeBu *consumption, but it cannot change the payload consumed by* subject_reduction_from_bundle*.*

Proof. Project the carrier A. The rows B, S, L, P are fixed before the route-audit row is inspected, and the obstruction row O is a separate displayed coordinate.

The route row R records consumption of a SubjectReductionDischargeBundle whose four fields are exactly B, S, L, P. The displayed Cont rows in C replay those fields to the consumer, while H only transports the named rows componentwise.

The obstruction row O remains present in the read. Since the carrier has no coordinate for replacing B, S, L, P or for dropping O, the route is coherent only as a route into the same finite audit packet. □

Theorem 789.3 (SrDischargeRouteAudit non-escape)**.** *Every public read of an accepted* SrDischargeRouteAudit$^{\uparrow}$ *carrier factors through*

$$B, S, L, P, R, O, H, C, Q, N.$$

The reader may inspect the four discharge rows, the bundle route audit, the obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It cannot extract a missing discharge proof, a closed subject-reduction theorem, host conversion semantics, quotient equality of MetaCIC terms, or an unrecorded route around the obstruction row.*

Proof. Begin with the carrier inventory. The only discharge-bearing coordinates are B, S, L, P, and the only route coordinate is R.

Apply the bundle coherence theorem. A permitted consumer follows R through direct bundle consumption, and that route ends at the same four discharge rows and preserves the obstruction row O.

Now inspect H, C, Q, N. These rows provide transport, replay, provenance, and local naming for the displayed packet. None of them introduces a discharge proof, a host conversion relation, quotient equality, or another route around O.

Therefore every public read stays inside the finite audit packet and the obstruction boundary remains visible at the point where the route is consumed. □

Theorem 789.4 (SrDischargeRouteAudit NameCert obligations)**.** *The local* NameCert$_{\text{SrDischargeRouteAudit}^{\uparrow}}$ *surface for Definition 789.1 consists exactly of carrier admission for $A = (B, S, L, P, R, O, H, C, Q, N)$, four-row discharge payload exposure, route-audit coherence for explicit bundle reads, obstruction-row preservation, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations certify the finite audit packet as a sibling lattice node for the subject-reduction discharge route relationship.*

Proof. Carrier admission is the tuple displayed in Definition 789.1. It names the four discharge rows, the route-audit row, the obstruction row, transport rows, continuation rows, provenance, and local naming.

Payload exposure follows by projecting B, S, L, P. The bundle coherence theorem binds accepted route reads to those same four rows and keeps O attached to the route boundary.

Stability and replay are supplied by H and C. The package row Q records the neighboring socket, discharge, obstruction, route-classifier, bundle-trace, and critical-path chapters from which the audit packet is read.

Finally, the non-escape theorem excludes every stronger consumer request. Hence the listed rows are not merely sufficient for the local NameCert surface; they exhaust it. □

Closure status for $\mathsf{NameCert}_{\mathsf{SrDischargeRouteAudit}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{SrDischargeRouteAudit}^{\uparrow}$ packet is generated from finite BHist rows for the four subject-reduction discharge payloads, a route-audit row for bundle consumption, an obstruction row, transports, continuations, provenance, and local naming. The route row is replayed by Cont through the explicit bundle, exposing the same four payload rows and keeping the obstruction visible. Componentwise hsame preserves the payload, route, and obstruction rows, while Pkg records neighboring discharge and audit dependencies. The packet certifies route coherence without discharging subject reduction.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SrDischargeRouteAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the SrDischargeRouteAudit carrier, payload exposure, bundle-typeclass coherence, obstruction preservation, non-escape theorem, and NameCert obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SrDischargeRouteAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim a missing discharge proof, closed subject reduction, host conversion semantics, quotient equality of MetaCIC terms, or an unrecorded route around the obstruction.

Upgrade path: Obligation closure requires Lean encodings of the audit carrier, route-coherence theorem, obstruction preservation, and non-escape theorem.

Chapter 790

A Concrete Naming Certificate for $\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}$

The $\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}$ packet turns the explicit discharge route of chapter 9374 into a finite NameCert carrier. It sits between the socket packet of chapter 719 and the subject-reduction packet of chapter 483: the socket names the four discharge obligations, the subject-reduction packet names the consumed typing-preservation read, and this route certificate records how bundle access exposes exactly that same four-row payload.

Definition 790.1 (SubjectReductionDischargeRouteCert carrier)**.** *A* $\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, V, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the explicit bundle row, naming the four-field* SubjectReductionDischargeBundle *packet;*
2. *V is the bundle-consumption row for the subject-reduction theorem obtained from the explicit packet;*
3. *O is the checked no-bundle obstruction row for the present raw relation;*
4. *H records componentwise* hsame *transports for the bundle, bundle-consumption, obstruction, socket, provenance, and naming rows;*
5. *C records the displayed* Cont *routes by which consumers read the explicit bundle route, the obstruction ledger, the discharge socket, and the subject-reduction packet;*
6. *P is the* Pkg *provenance over chapter 9374, chapter 719, and chapter 483;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}}$ *row.*

The carrier contains no inhabitant of the four discharge hypotheses for the closed-CIC substrate, no closed subject-reduction theorem, no host conversion rule, and no alternate subject-reduction route outside the displayed bundle row.

Lean4 (definition) `BEDC.MetaCIC.SubjectReductionDischargeBundle`

Theorem 790.2 (SubjectReductionDischargeRouteCert NameCert obligations)**.** *For the carrier of Definition 790.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (B, V, O, H, C, P, N),$$

route visibility for the explicit bundle row B, bundle-consumption row V, and obstruction row O, componentwise transport through H, displayed continuation reads through C, provenance through P, and local naming through N.

Lean4 ✓ `BEDC.MetaCIC.subject reduction from bundle`

Proof. Project the displayed packet R. The explicit route is the bundle row B consumed by V, and the obstruction row O records why the raw relation does not inhabit the bundle.

Read H and C next. Transport preserves the named route rows componentwise, while continuation replay exposes only the displayed bundle, obstruction, socket, and subject-reduction reads.

Finally inspect P and N. Provenance ties the route certificate to the vision chapter, the discharge socket, and the subject-reduction packet. Since no carrier coordinate supplies a closed-CIC inhabitant of the four hypotheses, the listed rows exhaust the NameCert obligations without closing the obstruction boundary. □

Theorem 790.3 (SubjectReductionDischargeRouteCert socket consumption)**.** *Every downstream consumer of an accepted* $\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}$ *packet that reads a* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row must consume the socket through the displayed route rows B, V, O, transported by H, replayed through C, and accounted for by P, N. The consumer may use the explicit bundle route only after the four discharge rows are supplied, and that route does not export a closed subject-reduction theorem for the present closed-CIC substrate.*

Proof. Start with the accepted route certificate. Its only subject-reduction route coordinates are the explicit bundle row and the corresponding consumption row.

Apply the socket chapter through provenance. The socket supplies the four discharge rows as a finite obligation surface, and the route certificate records that those rows are carried explicitly by B.

The subject-reduction packet is reached only after the displayed route has been replayed through C. Because the carrier has no coordinate for a closed-CIC inhabitant of the four hypotheses, socket consumption remains a route-certificate read and cannot become an unparameterised closed theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}$ packet is generated from an explicit bundle route, a subject-reduction consumption row, an obstruction row, componentwise hsame transports, displayed Cont routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 790.1, Definition 790.2, and Definition 790.3 bind the finite bundle route certificate and its socket-consumption boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionDischargeRouteCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close the four MetaCIC discharge obligations, construct an inhabited discharge bundle for the closed-CIC substrate, prove closed subject reduction, give Lean verification of the route certificate, or bridge check it.

Upgrade path: Obligation closure requires checked carrier admission, bundle route exactness, socket consumption, route replay, provenance, local naming, and non-escape for the same finite packet.

Chapter 791

A Concrete Naming Certificate for $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$

The $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ packet realizes the closure-preservation audit map of chapter 9381 as a finite NameCert surface. It sits beside the kernel admission packet of chapter 557, the substitution audit packet of chapter 779, the normalization audit seal of chapter 777, and the preservation-facing packet of chapter 483. Those siblings expose admission, substitution, closed-normal, and typed-preservation boundaries; this packet records only the closedness-preserving route from shift and substitution through one-step and finite beta movement.

Definition 791.1 (MetaCIC closure preservation audit carrier)**.** *A* $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, G, S, H, C, P, N)$$

with the following visible rows:

1. *T is the five-row theorem ledger for* `shift_one_preserves_closed_succ`*,* `substitute_var_preserves_closed` `substitute_preserves_closed_at`*,* `betaStep_preserves_closed`*, and* `betaStarStep_preserves_closed`*;*
2. *G is the generator-closure ledger for the closed-under-shift and closed-under-substitute classifier rows named in section 9381.5;*
3. *S is the handoff ledger that records how the substitution rows feed the beta-step row and how the beta-step row feeds the finite beta-chain row;*
4. *H records componentwise* hsame *transport for theorem rows, generator-closure rows, handoff rows, continuation rows, provenance rows, and local naming rows;*
5. *C records the displayed* Cont *routes for reading shift closedness, variable substitution closedness, full substitution closedness, one-step beta closedness, and beta-chain closedness in dependency order;*
6. *P is the* Pkg *provenance over chapter 9381, chapter 557, chapter 779, chapter 777, and chapter 483;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for typing preservation, conversion typing, closed subject reduction, host reduction semantics, quotient equality of terms, or an external proof oracle.

Theorem 791.2 (MetaCIC closure preservation audit NameCert obligations)**.** *For the carrier of Definition 791.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}}$ *surface consists exactly of carrier*

admission for $Q = (T, G, S, H, C, P, N)$, theorem-ledger readback through T, generator-closure readback through G, substitution-to-beta handoff readback through S, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.

Proof. Project the accepted carrier. The five theorem rows in T are the complete checked closedness-preservation chain named by chapter 9381.

Read G and S next. The generator rows authorize shift and substitution only relative to the closedness classifier, while the handoff row orders the substitution endpoint before beta-step and beta-chain reads.

The structural rows H, C, P, N transport, replay, locate, and name the same packet. Since no further coordinate is present, these rows exhaust the local NameCert obligations. □

Theorem 791.3 (MetaCIC closure preservation substitution-to-beta handoff)**.** *Every accepted* MetaCICClosurePreservationAudit$^{\uparrow}$ *carrier routes closedness preservation from substitution to beta movement through the ordered ledger*

$$\texttt{substitute_preserves_closed_at} \longrightarrow \texttt{betaStep_preserves_closed} \longrightarrow \texttt{betaStarStep_preserves_closed}.$$

The route may read the shift and variable-substitution rows as prerequisites of the full substitution row, but it cannot bypass the displayed substitution row when certifying the beta-redex case or bypass the one-step row when certifying a finite beta chain.

Proof. Start with the handoff coordinate S. Its first dependency records that the full substitution theorem is obtained from the shift and variable-substitution rows in the theorem ledger.

The beta-step read then consumes that substitution row only at the beta-redex boundary. Congruence cases stay inside the same theorem ledger and do not create a typing or conversion route.

Finally the finite beta-chain read consumes the one-step beta row. The continuation row C preserves this order for downstream consumers, so the packet exposes a dependency chain rather than an unstructured collection of theorem names. □

Theorem 791.4 (MetaCIC closure preservation subject-reduction boundary)**.** *An accepted* MetaCICClosurePreserva *packet may be consumed by the subject-reduction surface of chapter 483 only as closedness-stability data. It exports the displayed rows T, G, S, H, C, P, N and no row for* HasType *preservation, dependent-codomain transport, binder-domain preservation, beta-substitution typing, conversion typing, or closed subject reduction.*

Proof. Inspect the carrier rows. The theorem ledger T is stated over the closedness classifier, and G records generator closure for that classifier rather than for typing membership.

The handoff row S reaches beta-step and beta-chain closedness, not target typing. Its routes are therefore usable around the preservation-facing packet of chapter 483 without supplying any of the typed discharge rows named there.

The exclusion follows from the missing coordinates. The transport, continuation, provenance, and naming rows replay T, G, S componentwise, so they cannot introduce a hidden subject-reduction theorem. □

Closure status for NameCert$_{\text{MetaCICClosurePreservationAudit}^{\uparrow}}$.

Constructive story: A MetaCICClosurePreservationAudit$^{\uparrow}$ packet is generated from five closedness-preservation theorem rows, two generator-closure rows, a substitution-to-beta handoff ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICClosurePreservationAudit}^{\uparrow}}$, seed)

Scope closed: Definition 791.1, Definition 791.2, Definition 791.3, and Definition 791.4 bind the finite closure-preservation audit packet and its subject-reduction boundary.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICClosurePreservationAudit}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close typing preservation, conversion typing, closed subject reduction, dependent-codomain transport, host reduction semantics, quotient equality of terms, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-ledger exactness, generator-closure exactness, substitution-to-beta route ordering, transport, route replay, provenance, and non-escape for the same finite packet.

Chapter 792

A Concrete Naming Certificate for $\mathsf{ClosurePreservationAuditTrace}^{\uparrow}$

The $\mathsf{ClosurePreservationAuditTrace}^{\uparrow}$ packet realizes the closure-preservation audit map of chapter 9381 as finite NameCert data. It sits beside the route classifier of chapter 785, the substitution audit map of chapter 779, and the normalization audit seal of chapter 777: those siblings expose discharge routes, closed substitution rows, and candidate-mediated normalization reads, while this packet records the closedness-preservation theorem chain as one replayable audit trace.

Definition 792.1 (Closure-preservation audit trace carrier)**.** *A* $\mathsf{ClosurePreservationAuditTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, V, U, B, F, H, C, P, N)$$

with the following visible rows:

1. *S is the shift-one closedness row for* shift_one_preserves_closed_succ*;*
2. *V is the variable-substitution row for* substitute_var_preserves_closed_at*;*
3. *U is the full-substitution row for* substitute_preserves_closed_at*;*
4. *B is the beta-step closedness row for* betaStep_preserves_closed*;*
5. *F is the finite beta-star preservation row for* betaStarStep_preserves_closed*;*
6. *H records componentwise* hsame *transport across the five closedness-preservation rows, provenance, routes, and local naming;*
7. *C records the displayed* Cont *routes by which a consumer replays the shift, variable-substitution, full-substitution, beta-step, and beta-star rows in order;*
8. *P is the* Pkg *provenance over* Term*,* ClosedAt*,* shift*,* substitute*, one-step beta reduction, finite beta-star chains,* BHist*,* hsame*,* Cont*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditTrace}^{\uparrow}}$ *row.*

The carrier exports no typing preservation theorem, no dependent codomain transport, no normalization proof, no confluence theorem, no quotient equality over reductions, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditTraceUp.ClosurePreservationAuditTraceTasteGate`
`single carrier alignment`

Theorem 792.2 (Closure-preservation audit trace NameCert obligations)**.** *The carrier of Definition 792.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditTrace}^{\uparrow}}$ *obligation surface. Its obligations are carrier admission for*

$$T = (S, V, U, B, F, H, C, P, N),$$

shift-one readback through S*, variable-substitution readback through* V*, full-substitution readback through* U*, beta-step readback through* B*, beta-star readback through* F*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed finite packet. The rows S, V, U expose the shift-one, variable-substitution, and full-substitution closedness facts described by the audit map.

Project B and F next. The beta-step row consumes the substitution row at a contraction boundary, and the finite beta-star row records repeated consumption of the beta-step row along the displayed chain.

Finally inspect H, C, P, N. These rows transport, replay, locate, and name the same finite packet. Since the carrier has no coordinate for typing preservation, dependent codomain transport, normalization, confluence, quotient equality, or host semantic equality, the listed rows exhaust the local NameCert surface. □

Theorem 792.3 (Closure-preservation audit trace beta-star coverage)**.** *Every accepted* ClosurePreservationAuditTra *packet covers the closedness-preservation route from one-step beta contraction to finite beta-star chains. A consumer that reads the beta-star row* F *must reach it through the displayed beta-step row* B*, the full-substitution row* U*, the variable-substitution row* V*, the shift-one row* S*, and the route and provenance rows* C, P, N*, with componentwise transport supplied by* H*.*

Proof. Start with an accepted carrier $T = (S, V, U, B, F, H, C, P, N)$. The finite packet orders the closedness rows so that the beta-star coordinate is not a standalone theorem name; it is a row read through the same trace as the one-step beta coordinate.

The beta-step row B depends on the full-substitution row U at the contraction boundary. The full-substitution row in turn is supported by the variable-substitution row V and the shift-one row S, matching the theorem chain recorded in chapter 9381.

The replay route C and provenance row P keep that chain visible to consumers, while H transports the displayed rows componentwise and N names the local certificate. Because no additional coordinate is available, beta-star coverage is exactly the finite trace through $S, V, U, B, F, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationAuditTrace}^{\uparrow}$ packet is generated from shift-one closedness, variable substitution, full substitution, beta-step preservation, beta-star preservation, componentwise transport, displayed continuation routes, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditTrace}^{\uparrow}}$, seed)

Scope closed: Definition 792.1, Definition 792.2, and Definition 792.3 bind the finite closure-preservation trace and its beta-star coverage boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close typing preservation, dependent codomain transport, subject reduction, normalization, confluence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exact readback of the five closedness-preservation rows, transport, route replay, provenance, local naming, and beta-star non-escape for the same finite trace.

Chapter 793

A Concrete Naming Certificate for $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$

The $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ packet is the finite audit surface for the explicit bundle discharge route described in chapter 9374. It keeps the obstruction boundary of chapter 9377 visible while consuming the socket rows of chapter 719. The packet records route handoff data only; it does not inhabit the four MetaCIC discharge hypotheses and does not close subject reduction.

Definition 793.1 (Subject-reduction route-audit carrier)**.** *A* $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, E, I, U, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the explicit bundle row naming the four subject-reduction discharge hypotheses as one* SubjectReductionDischargeBundle *packet;*
2. *E is the route-exactness row comparing accepted bundle reads with the socket payload;*
3. *I is the invocation row recording downstream explicit-bundle consumption;*
4. *U is the socket handoff row through which the consumer reaches the displayed discharge socket surface;*
5. *S is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row carrying the four named discharge sockets;*
6. *H records componentwise* hsame *transports for the bundle row, exactness row, invocation row, socket row, handoff row, provenance row, and local name;*
7. *C records the displayed* Cont *routes from route exactness to socket handoff and downstream subject-reduction consumption;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, *and the bundle route;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteAudit}^{\uparrow}}$ *row.*

The carrier classifier compares only the displayed route rows, socket rows, transports, continuations, provenance, and local naming data. It has no coordinate for closed subject reduction, host conversion, quotient equality of terms, erased semantics, or an inhabitant of any missing discharge hypothesis.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteAuditUp.SubjectReductionRouteAuditTasteGate single carrier alignment`

Theorem 793.2 (SubjectReductionRouteAudit NameCert obligations)**.** *For the carrier of Definition 793.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (B, E, I, U, S, H, C, P, N),$$

bundle-route visibility through B*, route-exactness comparison through* E*, invocation through* I*, socket handoff through* U, S*, componentwise transport through* H*, displayed continuation reads through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the displayed packet A. The bundle row B is the public route presentation, while E records exactness against the socket payload.

Read the invocation row I. It records that the explicit bundle path is used, but it does not add a fifth discharge row and does not alter the four-row socket payload.

The handoff rows U, S place the route in the socket surface. The transports H and continuations C replay only the displayed route and socket coordinates, and P, N record provenance and local naming. These rows exhaust the finite NameCert obligations. □

Theorem 793.3 (SubjectReductionRouteAudit socket handoff)**.** *Every downstream consumer of an accepted* $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ *packet reaches the subject-reduction discharge socket only through*

$$B, E, I, U, S, H, C, P, N.$$

The explicit bundle route hands off the socket payload without exporting a closed subject-reduction theorem, a proof of the four discharge hypotheses, a host conversion rule, or any route outside the displayed bundle comparison.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteAuditUp.SubjectReductionRouteAudit socket handoff`

Proof. Start with the carrier and the NameCert-obligation theorem. They identify every permitted consumer read with one of the displayed rows of A.

The bundle route reaches the socket through B, U, S. The comparison row E records that this path exposes the same four-row payload named by the socket.

Transport and replay are then forced through H and C. Since the packet has no coordinate for an inhabited discharge hypothesis, host conversion, closed subject reduction, or an alternate route, the socket handoff remains a finite audit row for bundle-route consumption alone. □

Theorem 793.4 (SubjectReductionRouteAudit obligation surface)**.** *Every accepted* $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ *packet exposes exactly its bundle route comparison row, socket handoff row, nonescape ledger, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before any MetaCIC discharge consumer can inspect it. The exposed surface remains an audit packet and does not prove subject reduction, closed subject reduction, or a public schema bridge.*

Proof. Project the carrier of Definition 793.1. The comparison row is E, the socket handoff is read through U, S, the nonescape information is the absence of any discharge-hypothesis coordinate beyond the displayed route rows, and H, C, P, N supply transport, replay, provenance, and naming.

Apply the NameCert obligations of Definition 793.2. They identify the accepted packet with exactly those carrier, comparison, handoff, transport, replay, provenance, and local naming rows.

Finally use the socket handoff theorem, Definition 793.3. It permits downstream inspection only through the displayed bundle comparison and socket rows, while explicitly withholding closed subject reduction, the missing discharge hypotheses, host conversion, and any route outside the displayed bundle comparison. Hence the obligation surface supports MetaCIC discharge work without asserting subject reduction, closed subject reduction, or a public schema bridge. □

Theorem 793.5 (SubjectReductionRouteAudit discharge boundary)**.** *Every accepted* SubjectReductionRouteAudit$^{\uparrow}$ *packet exposes the explicit bundle row, route-exactness row, invocation row, socket handoff row, discharge-socket row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before any MetaCIC consumer may inspect it. The same packet still does not inhabit any of the four discharge hypotheses and does not prove closed subject reduction.*

Proof. Project the carrier of Definition 793.1. The coordinates $B, E, I, U, S, H, C, P, N$ are respectively the explicit bundle, exactness, invocation, handoff, socket, transport, replay, provenance, and local naming rows.

Apply Definition 793.2. It forces every accepted inspection route to pass through exactly those displayed rows before a downstream MetaCIC consumer can read the packet.

Apply Definition 793.3. The socket handoff exposes the discharge-socket payload through U, S while preserving the absence of coordinates for the four hypothesis inhabitants, host conversion, closed subject reduction, or an alternate route. Thus the discharge boundary is a finite audit surface and not a proof of subject reduction. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteAudit}^{\uparrow}}$.

Constructive story: A SubjectReductionRouteAudit$^{\uparrow}$ packet is generated from an explicit bundle row, route-exactness comparison, an invocation row, a socket handoff, the discharge socket row, componentwise hsame transports, displayed Cont routes, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubjectReductionRouteAudit}^{\uparrow}}$, seed)

Scope closed: Definition 793.1, Definition 793.2, and Definition 793.3 bind the finite route-audit packet between bundle comparison and subject-reduction socket handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubjectReductionRouteAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not prove any MetaCIC discharge hypothesis, construct closed subject reduction, provide Lean verification, or bridge the packet to a public schema.

Upgrade path: Obligation closure requires checked carrier admission, bundle route comparison, socket handoff exactness, non-escape, provenance, and local naming for the same finite route-audit packet.

Chapter 794

A Concrete Naming Certificate for $\mathsf{ClosurePreservationTrace}^{\uparrow}$

The $\mathsf{ClosurePreservationTrace}^{\uparrow}$ packet realizes the closedness-preservation audit map of chapter 9381 as finite NameCert data. It sits alongside the subject-reduction carrier of chapter 483, the closed-term substitution carrier of chapter 537, and the beta-substitution preservation carrier of chapter 716. Its scope is the unconditional ClosedAt chain for shift, substitution, one-step beta, and finite beta paths; it does not discharge the dependent typing obligations carried by subject-reduction chapters.

Definition 794.1 (ClosurePreservationTrace carrier)**.** *A* $\mathsf{ClosurePreservationTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, V, U, B, R, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the shift-preservation row for the one-step successor boundary;*
2. *V is the variable-substitution preservation row at the selected closedness boundary;*
3. *U is the full substitution preservation row, linking the body budget and payload budget to the receiving* ClosedAt *boundary;*
4. *B is the one-step beta-preservation row;*
5. *R is the finite beta-star preservation row;*
6. *D is the displayed* ClosedAt *boundary row, recording the source depth, target depth, body budget, payload budget, and endpoint boundary consumed by S, V, U, B, R;*
7. *H records componentwise* hsame *transports for the shift, substitution, beta-step, beta-star, boundary, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes by which a consumer reads the preservation chain in the order shift, variable substitution, substitution, beta step, and beta-star closure;*
9. *P is the* Pkg *provenance over chapter 9381, chapter 537, chapter 716, chapter 483,* BHist, hsame, Cont, ClosedAt, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationTrace}^{\uparrow}}$ *row.*

The carrier contains no typing derivation, no subject-reduction theorem, no conversion rule, no confluence or normalization theorem, and no route for changing the classifier from ClosedAt *to* HasType.

Theorem 794.2 (ClosurePreservationTrace NameCert obligations)**.** *For the carrier of Definition 794.1, the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (S, V, U, B, R, D, H, C, P, N),$$

visibility of the five preservation rows S, V, U, B, R*, boundary accounting through* D*, componentwise transport through* H*, displayed route replay through* C*, provenance through* P*, and local naming through* N*. The surface classifies a finite closedness trace and not a typing-preservation package.*

Proof. Project the displayed packet T. The first five coordinates are precisely the preservation chain listed in chapter 9381: shift preservation, variable substitution preservation, full substitution preservation, beta-step preservation, and beta-star preservation.

Read the boundary row D next. It keeps the successor boundary for shift, the source and receiving boundaries for substitution, and the endpoint boundary for beta movement in one visible row, so consumers cannot silently change the closedness index while following the chain.

Finally inspect H, C, P, N. These rows provide transport, route replay, provenance, and local naming for the displayed coordinates. Since the carrier has no typing coordinate and no host conversion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 794.3 (ClosurePreservationTrace closedness route)**.** *Every accepted* $\mathsf{ClosurePreservationTrace}^{\uparrow}$ *carrier exposes a single closedness-preservation route from the source* ClosedAt *boundary through shift, substitution, one-step beta reduction, and finite beta-star closure. A consumer that starts with a displayed closed source term and follows* C *reaches only the transported* ClosedAt *endpoint named by* D *and cannot obtain a* HasType *endpoint from this packet.*

Proof. Start with the carrier projection of Definition 794.1. The continuation route C first reads the shift row S and the variable-substitution row V, then uses the full substitution row U to account for the beta-redex contractum boundary.

The one-step beta row B reads the same substitution boundary, while the beta-star row R composes finitely many such one-step reads. The boundary row D is shared by those rows, so the source and endpoint closedness claims remain on the displayed boundary route.

No step in this route reads the subject-reduction carrier as a proof payload. The references to chapter 483, chapter 537, and chapter 716 are provenance anchors and neighboring consumers, not hidden typing evidence. □

Theorem 794.4 (ClosurePreservationTrace non-escape)**.** *Every public read of an accepted* ClosurePreservationTrace *carrier factors through*

$$S, V, U, B, R, D, H, C, P, N.$$

The read may inspect the closedness preservation rows, the ClosedAt *boundary ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row. It cannot read a typing derivation, a subject-reduction discharge row, a conversion theorem, a confluence theorem, a normalization theorem, quotient equality of terms, or a host semantic model.*

Proof. Apply the NameCert-obligation theorem to identify any accepted carrier with the finite packet $T = (S, V, U, B, R, D, H, C, P, N)$.

For an admitted closedness read, the consumer follows the displayed route through C and transports rows componentwise through H. The package provenance P records where the chain is cited, but it does not add another payload coordinate.

For any request outside the closedness route, the carrier has no row to satisfy it. In particular, subject-reduction and beta-substitution chapters may consume or compare the closedness trace, but this packet does not inhabit their typing-preservation obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationTrace}^{\uparrow}$ packet is generated from finite shift, substitution, beta-step, beta-star, ClosedAt boundary, transport, continuation, package, and naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosurePreservationTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 794.1, Definition 794.2, Definition 794.3, and Definition 794.4 bind the finite closedness-preservation trace over the audit-map chain.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosurePreservationTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove typing preservation, confluence, normalization, conversion soundness, Lean verification of the trace carrier, or a public bridge.

Upgrade path: Obligation closure requires checked carrier admission, closedness-route exactness, transport stability, continuation replay, provenance, local naming, and non-escape for the same finite trace packet.

Chapter 795

A Concrete Naming Certificate for MetaCICClosureTrace$^{\uparrow}$

The MetaCICClosureTrace$^{\uparrow}$ packet gives the ClosurePreservation audit map of chapter 9381 a finite BEDC landing. It sits beside the audit-gate packet of chapter 552 and the kernel trace packet of chapter 557: the former records admission discipline, the latter records accepted declarations, and this packet records the closedness-preservation rows that downstream MetaCIC routes may replay.

Definition 795.1 (MetaCICClosureTrace carrier)**.** *A* MetaCICClosureTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, U, V, B, R, G, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the shift closedness row, naming the theorem row for* `shift_one_preserves_closed_succ`*;*
2. *U is the variable-substitution closedness row, naming the theorem row for* `substitute_var_preserves_closed_`
3. *V is the full-substitution closedness row, naming the theorem row for* `substitute_preserves_closed_at`*;*
4. *B is the one-step beta closedness row, naming the theorem row for* `betaStep_preserves_closed`*;*
5. *R is the finite beta-chain closedness row, naming the theorem row for* `betaStarStep_preserves_closed`*;*
6. *G records the generator-closure rows for* `closed_under_shift_classifier` *and* `closed_under_substitute_cl`
7. *K records the kernel-acceptance trace rows for the accepted MetaCIC ClosurePreservation declarations and their displayed dependency ledger;*
8. *H records componentwise* hsame *transports for theorem rows, generator rows, kernel trace rows, provenance rows, and local naming rows;*
9. *C records the displayed* Cont *routes by which downstream subject-reduction, closed-normal, and conversion-facing packets read the closure trace;*
10. *P is the* Pkg *provenance over chapter 9381, chapter 552, chapter 557, the MetaCIC ClosurePreservation module, generator-closure rows, componentwise* hsame*, and* NameCert *rows;*
11. *N is the local* NameCert$_{\text{MetaCICClosureTrace}^{\uparrow}}$ *row.*

The carrier contains no typing preservation theorem, no subject-reduction discharge inhabitant, no confluence theorem, no normalization theorem, no hidden kernel soundness premise, and no generator outside the displayed shift, substitution, beta-step, and finite beta-chain rows.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceTasteGate single carrier alignment`

Theorem 795.2 (MetaCICClosureTrace substitution and shift generator package)**.** *For every accepted* MetaCICClosureTrace$^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$*, the rows* S, U, V, G *form a finite generator package for the closedness classifier. The shift row* S *authorizes the one-step lift at the successor boundary, the variable row* U *and full-substitution row* V *authorize substitution at the receiving boundary, and the generator row* G *records exactly those two generator-closure classifier witnesses as the package exported to downstream closure consumers.*

Proof. Project the displayed packet T. The shift row and the two substitution rows are already separated as S, U, V, so the carrier has a finite inventory before any downstream consumer is considered.

Read the generator row G. Its shift coordinate is backed by the same closedness-preservation theorem named by S, and its substitution coordinate is backed by the variable and full-substitution theorem rows U, V.

Finally inspect H, C, P, N. Transport and continuation replay preserve the displayed package rows, while provenance and local naming bind them to the ClosurePreservation audit map. Since no other generator coordinate occurs in T, the exported generator package is exhausted by S, U, V, G. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier substitution shift generator package`

Theorem 795.3 (MetaCICClosureTrace closed-substitution route)**.** *Every closed-substitution consumer of an accepted* MetaCICClosureTrace$^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$ *must read the shift row* S*, the variable-substitution row* U*, the full-substitution row* V*, the generator row* G*, and the kernel-acceptance row* K *before exporting closedness preservation. The route exports only the closure-preservation boundary for closed substitution; it exports no typing, conversion, confluence, normalization, subject-reduction, or kernel-soundness conclusion.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier closed substitution route`

Proof. Project the displayed carrier T. The rows S, U, V, G, K are separate coordinates, so a closed-substitution consumer can identify the shift, variable-substitution, full-substitution, generator, and acceptance ledger data before any beta-chain or continuation row is inspected.

Use Definition 795.2. It identifies S, U, V, G as the finite generator package for the closedness classifier, with U and V supplying the substitution coordinates and S supplying the boundary shift coordinate.

The row K then records kernel acceptance for the displayed ClosurePreservation declarations. The substitution audit-map boundary supplies the prose routing discipline: accepted closed-substitution reads stop at closedness preservation and do not acquire typing, conversion, confluence, normalization, subject-reduction, or kernel-soundness data from any hidden carrier coordinate. □

Theorem 795.4 (MetaCICClosureTrace beta-star preservation readback)**.** *Every downstream consumer that reads finite beta-chain closedness from an accepted* MetaCICClosureTrace$^{\uparrow}$ *carrier receives it through the route*

$$B, R, H, C, P, N.$$

The one-step beta row B *supplies the local preservation step, the finite beta-chain row* R *supplies the iterated readback, and the route may cite the checked MetaCIC theorem* `BEDC.MetaCIC.betaStarStep_preserves_`

as the kernel-accepted declaration recorded by K. The readback preserves only `ClosedAt`*; it exports no typing, conversion, confluence, or normalization conclusion.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier beta star preservation readback`

Proof. Begin with the beta rows. The row B names the one-step closedness preservation fact, and the row R names its finite beta-chain closure.

Use the kernel trace row K only as an acceptance ledger for the named declaration. This ledger records that the finite beta-chain theorem is part of the accepted ClosurePreservation surface; it does not add a separate semantic premise.

Now replay the consumer route through H, C, P, N. Componentwise transport keeps B and R aligned, the continuation route exposes the finite read, provenance identifies the ClosurePreservation audit map, and the local NameCert row names the packet. Because the carrier has no typing or conversion coordinate, the readback stops at closedness preservation. □

Theorem 795.5 (MetaCICClosureTrace NameCert obligations)**.** *For the carrier of Definition 795.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosureTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (S, U, V, B, R, G, K, H, C, P, N),$$

the five ClosurePreservation theorem rows S, U, V, B, R, generator-closure readback through G, kernel-acceptance trace accounting through K, componentwise transport through H, displayed continuation reads through C, package provenance through P, and local naming through N.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier namecert obligations`

Proof. Carrier admission is the first projection of the displayed packet T. This projection fixes the five theorem rows before the generator, kernel trace, transport, route, provenance, or naming rows are read.

The generator and kernel obligations are then forced by G and K. The row G exposes only the shift and substitution generator-closure witnesses, while K records accepted ClosurePreservation declarations and their dependency ledger.

The remaining obligations are exactly the public routing rows. The row H supplies componentwise hsame transport, C supplies continuation replay, P supplies package provenance, and N supplies local naming. No carrier coordinate remains from which a stronger MetaCIC theorem could be exported. □

Theorem 795.6 (MetaCICClosureTrace substitution beta-chain boundary)**.** *Every accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *consumer that combines the substitution rows S, U, V, G with the beta rows B, R must still read only* `ClosedAt` *preservation through K, H, C, P, N. The combined route may replay shift, substitution, one-step beta, and finite beta-chain closedness preservation, but it cannot export typing preservation, confluence, normalization, subject reduction, conversion soundness, or kernel soundness.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier substitution beta chain boundary`

Proof. First read the substitution side through Definition 795.2 and Definition 795.3. These results identify S, U, V, G as a closedness-generator package and force the closed-substitution route to stop at `ClosedAt` preservation.

Next read the beta side through Definition 795.4. It exposes only the one-step row B and finite-chain row R as closedness preservation data.

Finally apply Definition 795.5. The only remaining public route is K, H, C, P, N, whose kernel ledger, transport, replay, provenance, and naming rows preserve the displayed ClosurePreservation declarations. Since the carrier has no coordinate for typing, confluence, normalization, subject reduction, conversion soundness, or kernel soundness, the combined consumer cannot export any of those stronger claims. □

Theorem 795.7 (MetaCICClosureTrace kernel ledger boundary)**.** *For every accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$*, the kernel-acceptance row* K *is only an acceptance ledger for the displayed ClosurePreservation declarations. It cannot be used as a hidden kernel-soundness premise, a typing derivation, a confluence witness, a normalization witness, or a subject-reduction witness.*

Proof. Project the row K from the carrier. By Definition 795.5, this row accounts for accepted ClosurePreservation declarations and their displayed dependency ledger.

The substitution and beta routes do not enlarge that ledger. The route of Definition 795.3 reads K after S, U, V, G, while Definition 795.4 reads K only as acceptance evidence for the beta-chain closedness row.

Thus K remains a ledger coordinate rather than a semantic kernel premise. The stronger objects named in the statement are absent from Definition 795.1, so no accepted consumer can recover them from the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier kernel ledger semantic name certificate`

Theorem 795.8 (MetaCICClosureTrace obligation-closure package)**.** *The substitution-shift generator package of Definition 795.2, the beta-star preservation readback of Definition 795.4, the local NameCert obligations of Definition 795.5, and the kernel-ledger boundary of Definition 795.7 jointly exhaust the obligation-closure surface for the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *packet. The package exposes only shift, substitution, beta-step, finite beta-chain closedness, generator readback, kernel acceptance ledger, transport, replay, provenance, and local naming; it exports no typing preservation, confluence, normalization, subject reduction, conversion soundness, or kernel soundness.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier obligation closure package`

Proof. Project the carrier of Definition 795.1. The rows S, U, V, G are the substitution and shift generator package, while B, R are the one-step beta and finite beta-chain closedness rows.

Apply the substitution-shift generator theorem to bind S, U, V, G as the complete closedness-generator surface. Apply the beta-star readback theorem to bind B, R as the only finite beta-chain read available to downstream consumers.

Now use the local NameCert obligations. They account for carrier admission, ClosurePreservation theorem rows, generator readback, kernel-trace accounting, componentwise transport, continuation replay, provenance, and local naming.

Finally use the kernel-ledger boundary. The ledger row K remains acceptance accounting for displayed ClosurePreservation declarations and is not a semantic premise for stronger MetaCIC results. Since the carrier has no row for typing preservation, confluence, normalization, subject reduction, conversion soundness, or kernel soundness, the listed rows exhaust the obligation-closure package. □

Theorem 795.9 (MetaCICClosureTrace candidate-mediated SN route)**.** *Any downstream read that uses an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$ *inside a candidate-mediated normalization packet may consume from* T *only the shift, substitution, beta-step, finite beta-chain, generator, kernel-ledger, transport, replay, provenance, and local-name rows. The strong-normalization conclusion must import candidate evidence from the router or frontier before it is stated;* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *itself contributes closedness preservation and kernel acceptance accounting, never evidence-free normalization.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate mediated sn route`

Proof. Begin with the closedness rows. The substitution-shift generator package identifies S, U, V, G as the shift and substitution surface, while the beta-star readback identifies B, R as the one-step and finite beta-chain closedness surface.

Next read the kernel boundary. The kernel-ledger theorem keeps K as acceptance accounting for displayed ClosurePreservation declarations rather than as a semantic proof of subject reduction, conversion, confluence, or normalization.

Now apply the obligation-closure package. It routes the remaining rows through H, C, P, N and states that the exported surface is closedness preservation only.

Therefore a candidate-mediated normalization packet may use this carrier as a closedness-preservation ledger, but the candidate evidence required for strong normalization must arrive from the candidate router or frontier. Since no such evidence is a coordinate of T, the route cannot state a strong-normalization conclusion from $\mathsf{MetaCICClosureTrace}^{\uparrow}$ alone. □

Theorem 795.10 (MetaCICClosureTrace residual frontier)**.** *Every residual-frontier consumer of an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$ *may read the ClosurePreservation rows*

$$S, U, V, B, R, G, K, H, C, P, N$$

only as closedness-preservation evidence. Any confluence, residual, candidate, strong-normalization, subject-reduction, or conversion-soundness conclusion must import the corresponding diamond, residual, and candidate data from a separate MetaCIC frontier packet before it is claimed.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier residual frontier`

Proof. Begin with the candidate-mediated SN route. It permits downstream normalization packets to consume the closure trace only through the displayed shift, substitution, beta-step, finite beta-chain, generator, kernel-ledger, transport, replay, provenance, and local-name rows.

Use the kernel-ledger boundary next. The row K records acceptance accounting for ClosurePreservation declarations, not a hidden proof of subject reduction, confluence, normalization, residual determinacy, or conversion soundness.

Thus a residual-frontier consumer may forward $S, U, V, B, R, G, K, H, C, P, N$ as a closedness-preservation ledger. Since the carrier contains no diamond, residual, or candidate evidence row, every stronger MetaCIC conclusion must be supplied by a separate frontier packet before the consumer can claim it. □

Theorem 795.11 (MetaCICClosureTrace obligation-closure forwarder)**.** *For every accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$*, the rows* $S, U, V, B, R, G, K, H, C, P, N$ *jointly form the obligation-closure surface exported to closure-preservation consumers. The exported surface*

consists only of shift closedness, variable-substitution closedness, full-substitution closedness, one-step beta closedness, finite beta-chain closedness, generator readback, kernel-trace ledger accounting, componentwise transport, replay, provenance, and local naming. It refuses typing preservation, conversion, confluence, normalization, subject reduction, and hidden kernel soundness.

Proof. First apply Definition 795.2. It identifies S, U, V, G as the displayed generator package for shift and substitution closedness.

Next apply Definition 795.4. It identifies B, R as the one-step and finite beta-chain closedness readback route and keeps that readback at `ClosedAt` preservation.

Now apply Definition 795.7. The row K remains only an acceptance ledger for displayed ClosurePreservation declarations, not a hidden semantic premise.

Finally apply Definition 795.5. The rows H, C, P, N provide componentwise transport, replay, provenance, and local naming for the same finite packet. Since no carrier coordinate stores typing, conversion, confluence, normalization, subject reduction, or kernel soundness, the forwarder exports exactly the listed obligation-closure surface. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier obligation closure forwarder`

Theorem 795.12 (MetaCICClosureTrace obligation-closure upgrade)**.** *The substitution-shift generator package of Definition 795.2, beta-star preservation readback of Definition 795.4, NameCert obligations of Definition 795.5, kernel-ledger boundary of Definition 795.7, and candidate frontier of Definition 795.13 jointly form the obligationClosure package for* $\mathsf{MetaCICClosureTrace}^{\uparrow}$*. The package exports only finite* `ClosedAt` *preservation across shift, substitution, beta-step, finite beta-chain, generator readback, kernel-ledger accounting, transport, replay, provenance, and local naming. It exports no typing theorem, confluence theorem, normalization theorem, subject-reduction theorem, conversion-soundness theorem, or kernel-soundness theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier obligation closure upgrade`

Proof. Begin with the generator package. The substitution-shift theorem identifies S, U, V, G as the complete closedness-generator surface for shift and substitution reads.

Attach the beta route next. The beta-star preservation readback identifies B, R as the displayed one-step and finite-chain closedness rows, and the NameCert obligations bind those rows to carrier admission, transport, replay, provenance, and local naming.

Use the kernel-ledger boundary to keep K as acceptance accounting for ClosurePreservation declarations rather than as a hidden semantic premise. The candidate-frontier theorem then orders the whole route as

$$S, U, V, G \longmapsto B, R \longmapsto K \longmapsto H, C, P, N.$$

Thus the combined package reaches obligationClosure for the paper packet by preserving only `ClosedAt`. Since no carrier row stores typing, confluence, normalization, subject reduction, conversion soundness, or kernel soundness, none of those stronger claims is exported. □

Theorem 795.13 (MetaCICClosureTrace candidate frontier)**.** *Every normalization-facing, confluence-facing, or conversion-facing read of an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *packet must pass through the displayed candidate frontier*

$$S, U, V, G \longmapsto B, R \longmapsto K \longmapsto H, C, P, N.$$

The first stage is the substitution-and-shift generator package, the second stage is the beta-star closedness chain, the third stage is the kernel ledger for accepted ClosurePreservation declarations,

and the final stage is componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The frontier exports only finite* `ClosedAt` *preservation for the displayed trace; it exports no hidden subject-reduction hypothesis, no host reduction equality, no unrestricted strong-normalization principle, and no confluence statement outside the finite candidate trace.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate frontier`

Proof. Start with the generator side. The substitution-shift generator package identifies S, U, V, G as the complete closedness-generator surface for shift and substitution reads, while the closed-substitution route forces any closed-substitution consumer to pass through those rows before using the ledger.

Next attach the beta-chain side. The beta-star preservation readback exposes B, R as the one-step and finite-chain closedness rows, and it states that the readback stops at `ClosedAt` preservation.

Now read the kernel boundary. The kernel-ledger boundary keeps K as acceptance accounting for displayed ClosurePreservation declarations rather than as a semantic subject-reduction or kernel-soundness premise.

Finally package the route through the obligation-closure theorem. The rows H, C, P, N provide transport, replay, provenance, and naming for the same finite packet. Since the carrier has no coordinate for host reduction equality, unrestricted strong normalization, full confluence, subject reduction, typing preservation, conversion soundness, or kernel soundness, every such consumer remains on the displayed candidate frontier. □

Theorem 795.14 (MetaCICClosureTrace scoped confluence audit route)**.** *Every candidate-mediated normalization consumer of an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *packet may pass the closedness-preservation rows*

$$S, U, V, B, R, G, K, H, C, P, N$$

into a confluence-facing, conversion-facing, or strong-normalization-facing route only as scoped `ClosedAt` *evidence over the displayed carrier of Definition 795.1. Any strong-normalization, confluence, subject-reduction, or conversion-soundness conclusion must be imported from the candidate frontier or router packet rather than from* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *itself.*

Proof. Begin with the carrier of Definition 795.1. Its rows are closedness-preservation rows, generator rows, kernel-ledger accounting, transport, replay, provenance, and local naming; none is a typing, confluence, subject-reduction, conversion-soundness, or normalization coordinate.

Apply the obligation-closure package of Definition 795.8. It packages the substitution, shift, beta-step, finite beta-chain, NameCert, kernel-ledger, transport, replay, provenance, and naming rows as a finite `ClosedAt` surface.

Now use the candidate frontier of Definition 795.13. The frontier orders the same rows as the route through which normalization-facing, confluence-facing, and conversion-facing consumers may read the packet, and the checked TasteGate marker records that the carrier alignment has only this single displayed field surface.

Therefore a consumer may forward the closure trace as scoped `ClosedAt` evidence, but any strong-normalization, confluence, subject-reduction, or conversion-soundness theorem must arrive from the candidate frontier or router packet. The closure trace preserves closedness; it does not create the stronger MetaCIC conclusion. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier scoped confluence audit route`

Theorem 795.15 (MetaCICClosureTrace candidate-mediated closedness ledger)**.** *Every downstream closedness consumer of an accepted* MetaCICClosureTrace$^{\uparrow}$ *packet* $T = (S, U, V, B, R, G, K, H, C, P, N)$ *must read the candidate-mediated route in the order* S, U, V, G*, then* B, R*, then* K*, then* H, C, P, N*. The exported ledger is only a finite* `ClosedAt` *preservation route through substitution, shift, beta-step, finite beta-chain, kernel acceptance accounting, transport, replay, provenance, and local naming. It exports no subject reduction, no confluence, no normalization, no conversion theorem, no host reduction equality, and no kernel soundness.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate mediated closedness ledger`

Proof. First read the substitution-shift generator package. It identifies S, U, V, G as the finite candidate generator surface through which closed-substitution and shift reads enter the packet.

Next use the closed-substitution route and beta-star preservation readback. The former keeps substitution reads at `ClosedAt` preservation, and the latter attaches the one-step beta row B and finite-chain row R without introducing typing or conversion data.

Now apply the kernel ledger boundary. The row K records accepted ClosurePreservation declarations and their dependency ledger only as accounting, not as a hidden semantic proof of subject reduction, confluence, normalization, conversion, host reduction equality, or kernel soundness.

Finally apply the candidate frontier nonescape route. The rows H, C, P, N transport, replay, source, and name the same finite packet, so the candidate-mediated ledger exports only `ClosedAt` preservation for the displayed trace. □

Theorem 795.16 (MetaCICClosureTrace candidate closure forwarding)**.** *Any candidate-mediated consumer forwarding* MetaCICClosureTrace$^{\uparrow}$ *rows into normalization, confluence, subject-reduction, or conversion packets may forward only the displayed* `ClosedAt` *preservation rows* $S, U, V, B, R, G, K, H, C, P, N$ *of an accepted carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$*. Strong-normalization, confluence, subject-reduction, and conversion-soundness conclusions must be supplied by the downstream candidate packet;* MetaCICClosureTrace$^{\uparrow}$ *contributes no such conclusion by itself.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate closure forwarding`

Proof. First use the substitution-shift generator package of Definition 795.2. It identifies S, U, V, G as generator rows for closedness preservation, not as typing, reduction, or conversion evidence.

Next apply the beta-star preservation readback and the kernel-ledger boundary. The rows B, R replay beta-step and finite-chain closedness preservation, while K records kernel acceptance accounting only inside the same `ClosedAt` ledger.

Finally apply the candidate-mediated closedness ledger of Definition 795.15. Row exhaustion leaves H, C, P, N as transport, replay, provenance, and local naming support, so any stronger normalization, confluence, subject-reduction, or conversion-soundness theorem must enter from the downstream candidate packet. □

Theorem 795.17 (MetaCICClosureTrace candidate closedness interface)**.** *Every candidate-mediated consumer may import* MetaCICClosureTrace$^{\uparrow}$ *only as closedness evidence for shift, substitution, beta-step, finite beta-chain, kernel-ledger accounting, componentwise transport, displayed replay, provenance, and local naming. Any strong-normalization, confluence, subject-reduction, conversion-soundness, or typing conclusion must be supplied by a separate candidate frontier or router packet, not by the closure trace itself.*

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate closedness interface`

Proof. Project the carrier of Definition 795.1. Its displayed rows are exactly $S, U, V, B, R, G, K, H, C, P, N$, so the only theorem rows available to a consumer are shift, substitution, beta-step, finite beta-chain, generator, ledger, transport, replay, provenance, and local-name rows.

Apply the obligation-closure package of Definition 795.8. It identifies those rows as a finite `ClosedAt` preservation surface and refuses typing, confluence, normalization, subject-reduction, conversion-soundness, and kernel-soundness conclusions.

Use the kernel-ledger boundary of Definition 795.7. The row K remains acceptance accounting for ClosurePreservation declarations rather than a hidden semantic premise.

Finally apply the candidate frontier of Definition 795.13 and the candidate-mediated closedness ledger of Definition 795.15. These theorems route candidate-facing consumers through the same closedness rows and require every stronger MetaCIC conclusion to enter from a separate frontier or router packet. □

Theorem 795.18 (MetaCICClosureTrace closed-normalization source order)**.** *Every closed-normalization-facing consumer of an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier*

$$T = (S, U, V, B, R, G, K, H, C, P, N)$$

must read the displayed source order

$$S, U, V, G \longmapsto B, R \longmapsto K \longmapsto H, C, P, N.$$

The read supplies only closedness preservation for shift, variable substitution, full substitution, one-step beta, finite beta chains, generator rows, kernel-ledger accounting, transport, replay, provenance, and local naming. It refuses subject reduction, arbitrary normalization, quotient term equality, kernel soundness, and any normalization conclusion not already supplied by the downstream candidate packet.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier closed normalization source order`

Proof. Project the carrier row family $S, U, V, B, R, G, K, H, C, P, N$. The only normalization-facing data available from the trace are the displayed ClosurePreservation rows and their structural support.

Apply the substitution-shift generator package to order S, U, V, G. Apply the beta-star preservation readback to attach B, R as beta closedness rows, and then apply the kernel-ledger boundary to keep K as acceptance accounting rather than semantic soundness.

Finally apply the candidate closure forwarding theorem. It permits forwarding only the displayed `ClosedAt` preservation rows, so closed-normalization consumers must obtain any actual normalization theorem from the downstream candidate packet rather than from this closure trace. □

Theorem 795.19 (MetaCICClosureTrace confluence handoff boundary)**.** *Every confluence-facing handoff from an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier factors through*

$$S, U, V, B, R, G, K, H, C, P, N.$$

The handoff may cite the substitution generator package, beta-star preservation readback, kernel-ledger boundary, and candidate-mediated closedness ledger, but it exports no global confluence theorem, no subject-reduction theorem, no quotient term equality, no arbitrary normalization theorem, and no kernel-soundness premise.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier confluence handoff boundary`

Proof. Start with a confluence-facing consumer. Carrier projection exposes only the five closure theorem rows, the generator row, the kernel ledger, and the structural rows.

Use the substitution beta-chain boundary to combine S, U, V, G with B, R while staying inside `ClosedAt` preservation. Use the obligation-closure forwarder to route the same rows through K, H, C, P, N.

The candidate closure forwarding theorem is the final boundary. It allows the closedness trace to be forwarded into a confluence packet, but it does not let the trace itself assert global confluence, quotient equality of terms, subject reduction, arbitrary normalization, or kernel soundness. □

Theorem 795.20 (MetaCICClosureTrace decidability readback frontier)**.** *Every decidability-facing readback from an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier is confined to the displayed rows*

$$S, U, V, B, R, G, K, H, C, P, N.$$

It may read closedness preservation, generator accounting, kernel acceptance accounting, componentwise transport, displayed replay, provenance, and local naming. It may not read decidable conversion, decidable typing, quotient term equality, global confluence, arbitrary normalization, subject reduction, or kernel soundness from the closure trace.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier decidability readback frontier`

Proof. Project the carrier. No row of Definition 795.1 stores a decidability theorem, a conversion classifier, a typing classifier, quotient term equality, or kernel soundness.

Apply the NameCert obligations and kernel-ledger boundary. They identify K, H, C, P, N as ledger, transport, replay, provenance, and naming rows for the displayed ClosurePreservation declarations only.

Apply the candidate-mediated closedness ledger and candidate closure forwarding results. They allow a decidability-facing packet to consume closedness preservation as input data, but the decidability readback frontier remains the displayed row family; any decidability conclusion must be supplied by the consumer packet. □

Theorem 795.21 (MetaCICClosureTrace obligation status registration)**.** *For every accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$*, the substitution-shift package of Definition 795.2, the beta-star preservation readback of Definition 795.4, the local NameCert obligations of Definition 795.5, the kernel-ledger boundary of Definition 795.7, the candidate frontier of Definition 795.13, and the decidability readback frontier of Definition 795.20 form the paper obligation surface for the closure trace.*

This surface registers only shift closedness, variable-substitution closedness, full-substitution closedness, one-step beta closedness, finite beta-chain closedness, generator readback, kernel-acceptance accounting, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. It refuses subject reduction, confluence, normalization, conversion soundness, and kernel soundness.*

Proof. Read the substitution side first. The substitution-shift package identifies S, U, V, G as the finite generator surface for shift and substitution closedness, so no typing or conversion premise is introduced at the entry point.

Attach the beta-chain readback. The beta-star preservation theorem adds B, R only as one-step and finite-chain `ClosedAt` rows, while the NameCert obligations place those rows inside carrier admission, transport, replay, provenance, and local naming.

Use the kernel-ledger boundary and candidate frontier to order the remaining route. The row K remains acceptance accounting for displayed ClosurePreservation declarations, and the candidate frontier forces normalization-facing, confluence-facing, and conversion-facing consumers to pass through the same finite row family before asking any downstream packet for stronger evidence.

Finally apply the decidability readback frontier. It keeps decidability-facing reads inside $S, U, V, B, R, G, K, H, C, P, N$ and requires any decidable conversion, decidable typing, quotient term equality, confluence, normalization, subject reduction, conversion soundness, or kernel soundness conclusion to arrive from another packet. Thus the named results jointly register the paper obligation surface without asserting the stronger MetaCIC theorems. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier obligation status registration`

Theorem 795.22 (MetaCIC closure trace public boundary)**.** *Any public consumer of an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *packet must display the ClosurePreservation row family, the substitution row family, the beta-chain row, the kernel-trace row, the candidate-mediated strong-normalization row, the confluence row, and the decidability row before exporting the trace. Concretely, the consumer must read* S, U, V *for shift and substitution closedness,* B, R *for one-step beta and finite beta chains,* G *for generator accounting,* K *for the kernel-acceptance trace, the candidate frontier for strong-normalization-facing reads, the confluence handoff boundary, and the decidability readback frontier, then preserve the whole packet through* H, C, P, N*.*

The public export is a closure-trace boundary only. It does not prove subject reduction, full normalization, kernel soundness, confluence, decidable conversion, decidable typing, or Lean verification for the whole packet.

Proof. Project the carrier of Definition 795.1. The visible rows are the ClosurePreservation theorem rows, generator accounting, kernel ledger, transport, replay, provenance, and local naming; no row stores subject reduction, full normalization, kernel soundness, or a global confluence theorem.

Apply the substitution-shift generator package and beta-star preservation readback. These results expose S, U, V, G and B, R as closedness-preservation data only, so the public route begins with the displayed substitution and beta-chain rows rather than with a typing or conversion theorem.

Use the kernel-ledger boundary, candidate-mediated closedness ledger, confluence handoff boundary, and decidability readback frontier. They force normalization-facing, confluence-facing, and decidability-facing consumers to pass through the same finite row family before asking a downstream packet for stronger evidence.

Finally apply obligation status registration. It assembles the ClosurePreservation, substitution, beta-chain, kernel-trace, candidate-mediated strong-normalization, confluence, and decidability routes into the public paper boundary while preserving the refusal of subject reduction, full normalization, kernel soundness, and Lean verification. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier public boundary`

Theorem 795.23 (MetaCICClosureTrace candidate-mediated SN source scope)**.** *Any candidate-mediated strong-normalization source read exported by an accepted* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *carrier* $T = (S, U, V, B, R, G, K, H, C, P, N)$ *factors through the closed-substitution boundary, the candidate closedness ledger, the confluence handoff boundary, the decidability readback frontier, component-wise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Concretely, the source read may forward only the displayed closedness-preservation rows for shift, variable substitution, full substitution, one-step beta, finite beta chains, generator accounting, and kernel acceptance. It may then pass those rows to a separate candidate packet through the candidate-mediated closedness ledger of Definition 795.15, the confluence handoff boundary of Definition 795.19, and the decidability readback frontier of Definition 795.20. The export does not claim full subject reduction, global strong normalization, kernel soundness, unrestricted confluence, decidable conversion, decidable typing, or quotient term equality.

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier candidate sn source scope`

Proof. Project the carrier of Definition 795.1. Its rows are the substitution and beta closedness-preservation rows S, U, V, B, R, the generator and kernel-ledger rows G, K, and the structural rows H, C, P, N.

Use the closed-substitution route and candidate-mediated closedness ledger. They order the substitution rows, beta rows, generator row, and kernel ledger as closedness evidence only; no row in that sequence stores subject reduction or a normalization theorem.

Next use the confluence handoff boundary and decidability readback frontier. These results allow the same closedness surface to be forwarded into confluence-facing and decidability-facing consumers, while requiring any confluence, conversion, typing, or normalization conclusion to come from a separate packet.

Finally transport and name the export through H, C, P, N. The source scope is therefore exactly the displayed closure-trace route available to candidate-mediated consumers, and the excluded subject-reduction, global-normalization, kernel-soundness, confluence, decidability, and quotient-equality claims remain outside the carrier. □

Theorem 795.24 (MetaCICClosureTrace public bridge obligation)**.** *Any standard bridge for the public* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *boundary must preserve exactly the displayed public rows of Definition 795.22: ClosurePreservation rows, substitution rows, beta-chain rows, kernel-trace accounting, candidate-mediated normalization-facing rows, confluence-facing rows, decidability-facing rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The bridge obligation may not import subject reduction, type preservation, full normalization, kernel soundness, confluence, decidable conversion, decidable typing, quotient term equality, or any stronger MetaCIC theorem as an unrecorded premise.*

Proof. Start from the public boundary theorem, Definition 795.22. It lists the rows that a public consumer may inspect before exporting the closure trace.

Use obligation status registration, Definition 795.21. That theorem registers the paper obligation surface as closedness, generator accounting, kernel acceptance accounting, transport, replay, provenance, and naming, while refusing subject reduction, confluence, normalization, conversion soundness, and kernel soundness.

The bridge obligation is therefore a preservation boundary for those public rows. A future bridge can check that the displayed rows survive the bridge schema, but it cannot strengthen the carrier by importing typing, confluence, normalization, or kernel-soundness evidence from outside the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier public bridge obligation`

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosureTrace}^{\uparrow}}$.

Constructive story: A MetaCICClosureTrace$^{\uparrow}$ packet is generated from five MetaCIC ClosurePreservation theorem rows, shift and substitution generator-closure rows, kernel-acceptance trace rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICClosureTrace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 795.2, Definition 795.4, Definition 795.5, and Definition 795.7 bind the checked substitution-shift, beta-star, NameCert-obligation, and kernel-ledger rows as the paper-side bridge surface for the displayed closure-trace packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICClosureTrace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MetaCICClosureTraceUp.MetaCICClosureTraceCarrier\_obligation\_closure\_package`

Bridge status: none

Not claimed: This chapter does not prove subject reduction, type preservation, full normalization, kernel soundness, checked Lean carrier obligations, or a public bridge for the closure-trace packet.

Upgrade path: A bridge grade requires a checked standard bridge for the same public closure-trace boundary, while subject reduction, full normalization, and kernel soundness remain separate packets.

Chapter 796

A Concrete Naming Certificate for $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$

The $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$ packet realizes the discharge-route triangle of chapter 9374 as finite NameCert data. It sits beside the route certificate of chapter 718 and the preservation packet of chapter 483: the first names the discharge rows, the second names the subject-reduction endpoint, and this chapter records the bundle route, endpoint, and obstruction boundary in one visible packet.

Definition 796.1 (Subject-reduction route-triangle carrier)**.** *A* $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, E, O, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the explicit discharge bundle row carrying the four subject-reduction discharge obligations;*
2. *E is the represented* $\mathsf{SubjectReduction}^{\uparrow}$ *endpoint row reached only after the discharge payload has been displayed;*
3. *O is the obstruction row recording the present closed-CIC non-inhabitation boundary for the missing discharge hypotheses;*
4. *L is the route-ledger row, recording that the consumer entered through B directly;*
5. *H records componentwise* hsame *transports for the bundle, endpoint, obstruction, ledger, provenance, and local name;*
6. *C records displayed* Cont *routes from the bundle row to the endpoint and from the obstruction row to the route ledger;*
7. *P is the* Pkg *provenance over chapter 9374, chapter 718, and chapter 483;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteTriangle}^{\uparrow}}$ *row.*

The carrier is only this finite triangle packet. It contains no inhabitant of the missing discharge hypotheses, no closed subject-reduction theorem for the present closed-CIC substrate, no host conversion rule, and no route outside the displayed bundle, obstruction, and ledger rows.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteTriangleUp.SubjectReductionRouteTriangleTasteGate`
`single carrier alignment`

Theorem 796.2 (SubjectReductionRouteTriangle bundle projection)**.** *Every accepted* SubjectReductionRouteTrian *carrier projects an explicit discharge-bundle route*

$$B, H, C, P, N$$

whose payload is the same four-row discharge surface consumed by chapter 718. This projection may be read as a bundle-facing handoff to the represented $\mathsf{SubjectReduction}^{\uparrow}$ *endpoint* E*, but it does not certify that the four discharge rows are inhabited for the present closed-CIC substrate.*

Proof. Project the carrier $T = (B, E, O, L, H, C, P, N)$ to its bundle coordinate. The row B is already a displayed packet row, and the transport, route, provenance, and local naming coordinates H, C, P, N preserve that row as finite NameCert data.

The sibling discharge chapter supplies the semantic boundary for the four discharge obligations. Reading B through chapter 718 therefore exposes the obligation surface, not an inhabitant of the obligations.

Finally pass to E only through the displayed continuation row. Since the carrier also keeps the obstruction row O, the projection cannot be confused with a closed subject-reduction theorem for the present substrate. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteTriangleUp.SubjectReductionRouteTriangle bundle projection`

Theorem 796.3 (SubjectReductionRouteTriangle obstruction boundary)**.** *In every accepted* SubjectReductionRout *carrier, the obstruction row* O *separates route transport from discharge inhabitation. The packet may expose the bundle projection and the represented* $\mathsf{SubjectReduction}^{\uparrow}$ *endpoint row, but any consumer that asks whether the present closed-CIC substrate supplies the four missing discharge hypotheses must read* O *and cannot replace it by* B *or* E*.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteTriangleUp.SubjectReductionRouteTriangle obstruction boundary`

Proof. Read the triangle inventory. The bundle row, endpoint row, and obstruction row are separate coordinates of T.

The projection theorem identifies the route available to a consumer. It transports the four-field payload to the represented endpoint row, while it does not claim an inhabitant for the present closed-CIC discharge obligations.

It remains to inspect the obstruction coordinate. The route ledger records that the bundle route was used, and O records the boundary at which the closed-CIC substrate still lacks the required inhabitants. Since O is not definitionally one of B, E, no consumer can erase that boundary by choosing a route. □

Theorem 796.4 (SubjectReductionRouteTriangle NameCert obligations)**.** *For the carrier of Definition 796.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteTriangle}^{\uparrow}}$ *surface has four rows:*

1. *the explicit discharge bundle row* B*;*
2. *the represented subject-reduction endpoint row* E*;*
3. *the obstruction and non-inhabitation boundary row* O*;*
4. *the route ledger row* L*, transported and replayed through* H, C, P, N*.*

These four rows exhaust the public route-triangle certificate. The certificate names how bundle access reaches the subject-reduction endpoint and where the present obstruction remains, without adding host conversion, hidden discharge evidence, erased derivation semantics, quotient equality of MetaCIC terms, or a closed subject-reduction theorem.

Proof. First expose the carrier row by row. The rows B, E, O, L are the finite data being certified, while H, C, P, N provide transport, route replay, provenance, and local naming for exactly those rows.

The four-row sketch follows by grouping the carrier coordinates by their route role. The bundle row gives the direct route, the endpoint row records the represented preservation consumer, the obstruction row records the missing-inhabitant boundary, and the ledger records that the bundle route has been followed.

The sibling chapters supply the boundary checks. The discharge packet of chapter 718 fixes the obligation surface, and the subject-reduction packet of chapter 483 fixes the endpoint surface. Since no other coordinate appears in T, the listed rows exhaust the local NameCert obligations and leave the excluded objects outside the packet. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteTriangleUp.SubjectReductionRouteTriangle namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$ packet is generated from an explicit discharge bundle row, a represented subject-reduction endpoint, an obstruction boundary row, a route ledger, componentwise hsame transports, displayed Cont routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 796.1, Definition 796.2, Definition 796.3, and Definition 796.4 bind the finite route triangle over the bundle, endpoint, obstruction, and ledger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not inhabit the four MetaCIC discharge obligations, prove closed subject reduction, verify the route triangle in Lean, or bridge the packet to a public schema.

Upgrade path: Obligation closure requires checked carrier admission, bundle projection, obstruction separation, route-ledger exactness, provenance, and local naming for the same finite route-triangle packet.

Chapter 797

A Concrete Naming Certificate for $\mathsf{ClosedBetaPath}^{\uparrow}$

The $\mathsf{ClosedBetaPath}^{\uparrow}$ packet realizes the closure-preservation audit map of chapter 9381 as a finite NameCert route object. It sits between the beta inventory of chapter 761, the normal-form seal of chapter 740, and the closed-normal main-result packet of chapter 764: it names the route by which closedness travels along finite beta movement before those downstream packets read any closed-normal or beta-audit rows.

Definition 797.1 (Closed beta path carrier)**.** *A* $\mathsf{ClosedBetaPath}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (s, t, n, B, K, H, C, P, N)$$

with the following displayed rows:

1. *s is the source MetaCIC term row;*
2. *t is the target MetaCIC term row;*
3. *n is the closedness-depth row for the displayed* ClosedAt *boundary;*
4. *B is the finite* $\mathrm{BetaStarStep}(s, t)$ *route row, whose one-step edges are* BetaStep *constructors;*
5. *K is the closure-preservation witness row naming the* $\mathsf{BEDC.MetaCIC.betaStarStep_preserves_closed}$ *route from* $\mathrm{ClosedAt}(n, s)$ *to* $\mathrm{ClosedAt}(n, t)$*;*
6. *H records componentwise* hsame *transport for the source, target, depth, beta-route, closure-witness, provenance, and naming rows;*
7. *C records the displayed* Cont *replay routes by which a consumer reads B and then K;*
8. *P is the* Pkg *provenance over* ClosedAt, BetaStep, BetaStarStep, shift, $\mathsf{substitute}$, *componentwise transport, and sibling NameCert packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBetaPath}^{\uparrow}}$ *row.*

The carrier has no coordinate for typing preservation, normalization, confluence, beta conversion, or closed consistency.

Lean4 ✓ `BEDC.Derived.ClosedBetaPathUp.ClosedBetaPathTasteGate single carrier alignment`

Theorem 797.2 (Closed beta path NameCert obligations)**.** *For an accepted* $\mathsf{ClosedBetaPath}^{\uparrow}$ *carrier*

$$Q = (s, t, n, B, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedBetaPath}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed finite packet;*
2. *preservation of* $\mathrm{ClosedAt}(n, \cdot)$ *along the finite beta route* B *through the witness row* K*;*
3. *exact replay of the route order by first reading* B *and then reading* K *through the displayed continuation rows;*
4. *non-escape from typing, normalization, confluence, beta-conversion, and closed-consistency claims not present in the carrier;*
5. *ledger exactness for* $s, t, n, B, K, H, C, P, N$ *and no hidden route coordinate.*

Proof. Carrier admission is the packet of Definition 797.1. The admitted route row B supplies the finite beta chain, and the witness row K supplies the closedness transport for exactly that chain.

Continuation replay is ordered by the carrier itself: a consumer first reads the beta-star route and then reads the closure witness. Componentwise transport through H preserves the same source, target, depth, route, witness, provenance, and naming rows.

The remaining coordinates are structural provenance and naming rows. Since no coordinate carries typing preservation, normalization, confluence, beta conversion, or closed consistency, the five displayed obligations exhaust the local NameCert surface. □

Theorem 797.3 (Closed beta path closure-preservation factorization)**.** *Every consumer of an accepted* $\mathsf{ClosedBetaPath}^{\uparrow}$ *packet that later reads a closed-normal packet, a normal-form consistency seal, a closed-normal main packet, or a beta-audit packet must first factor its closedness read through the* BEDC.MetaCIC.betaStarStep_preserves_closed *route named by* K*. The consumer may then pass the transported closed endpoint to the sibling packet it cites, but it cannot use* $\mathsf{ClosedBetaPath}^{\uparrow}$ *as a typing, normalization, confluence, beta-conversion, or consistency certificate.*

Proof. Project the accepted packet Q. The only row that exports closedness at the target endpoint is K, and K is tied to the displayed beta-star route B. Thus every closedness read from s to t passes through the same closure-preservation route.

The sibling packets cited by the consumer have their own carriers and local NameCert rows. The present packet can hand them the transported closed endpoint only after B and K have been replayed through C and accounted for by P, N.

Because the carrier lacks rows for typing preservation, normalization, confluence, beta conversion, and consistency, the factorization cannot be strengthened into any of those claims. □

↪variant `BEDC.MetaCIC.betaStarStep preserves closed`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBetaPath}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedBetaPath}^{\uparrow}$ packet is generated from a source term, target term, closedness depth, finite BetaStarStep route, beta-star closedness-preservation witness, componentwise hsame transports, displayed Cont replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedBetaPath}^{\uparrow}}$, obligation)

Scope closed: Definition 797.1, Definition 797.2, and Definition 797.3 bind the finite closed-beta-route carrier, its five local NameCert obligations, and its closure-preservation factorization boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedBetaPath}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.MetaCIC.betaStarStep\_preserves\_closed`

Bridge status: none

Origin: human

Not claimed: This chapter does not close MetaCIC subject reduction, normalization, confluence, beta conversion, full closed consistency, or a public standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, route replay exactness, ledger exactness, and non-escape for the same finite closed beta path packet.

Chapter 798

A Concrete Naming Certificate for $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$

The $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ packet records the finite choice point between explicit bundle consumption and the checked obstruction ledger of chapter 9374. It reads the discharge bundle surface of chapter 718, the socket boundary of chapter 719, and the downstream subject-reduction consumer of chapter 483; it does not choose or discharge the missing closed-CIC obstruction.

Definition 798.1 (SubjectReductionRouteChoice carrier)**.** *A* $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, V, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the explicit bundle row naming the* SubjectReductionDischargeBundle *shape and its four discharge coordinates;*
2. *V is the bundle-consumption row corresponding to* subject_reduction_from_bundle*;*
3. *O is the obstruction-boundary row stating that neither route is inhabited for the present closed-CIC substrate inside this packet;*
4. *H records componentwise* hsame *transports for the bundle, consumption, obstruction, provenance, and naming rows;*
5. *C records the displayed* Cont *routes by which consumers read the bundle route, the obstruction row, the subject-reduction socket, and the subject-reduction packet;*
6. *P is the* Pkg *provenance over chapter 9374, chapter 718, chapter 719, and chapter 483;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and the visible transport and continuation routes.

Theorem 798.2 (SubjectReductionRouteChoice obligations)**.** *For the carrier of Definition 798.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (B, V, O, H, C, P, N),$$

route visibility for B, V, obstruction-boundary visibility for O, componentwise transport through H, displayed continuation reads through C, provenance through P, and local naming through N.

Lean4 ✓ BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceObligations

Proof. Project the displayed carrier. The rows B, V are the only route rows, and the row O is the only obstruction-boundary row.

Transport and replay are then read through H and C. These rows preserve the displayed carrier coordinates and do not create a discharge coordinate outside R.

Finally, P and N account for the finite provenance and local name. Hence the listed rows exhaust the NameCert obligations for the route-choice packet. □

Theorem 798.3 (SubjectReductionRouteChoice obligation surface)**.** *For an accepted* SubjectReductionRouteChoice *carrier $R = (B, V, O, H, C, P, N)$, the obligation surface exposes the explicit bundle row B, the bundle-consumption row V, the obstruction-boundary row O, the componentwise* hsame *row H, the displayed* Cont *row C, the package provenance row P, and the local naming row N. The same finite route is read through the discharge bundle surface, the socket boundary, and the downstream subject-reduction consumer of the sibling chapters.*

Lean4 ✓ BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceObligationSurface

Proof. Project the carrier of Definition 798.1. The bundle row, bundle-consumption row, obstruction row, transport row, replay row, provenance row, and local name are all visible coordinates of one finite packet.

The discharge bundle and socket boundary are not hidden behind the packet: they are the sibling surfaces that motivate the route-choice rows already displayed in B, V, O. The accepted carrier therefore exposes the same finite route before any downstream consumer can read it.

Since the carrier has no coordinate for closed-CIC discharge, hidden route choice, or host proof search, the displayed rows exhaust the obligation surface. □

Theorem 798.4 (SubjectReductionRouteChoice route-obligation surface)**.** *For an accepted* SubjectReductionRoute *carrier $R = (B, V, O, H, C, P, N)$, every route-obligation read exposes exactly the bundle row B and the bundle-consumption row V while retaining the obstruction row O as a visible boundary. The transport, replay, provenance, and naming rows H, C, P, N may preserve and replay only those displayed rows.*

Thus the route-obligation surface exports no closed subject-reduction theorem, quotient syntax equality, host evaluator soundness, hidden typeclass route, or third discharge path.

Proof. Project the carrier of Definition 798.1. The route-facing rows are B, V, the obstruction-facing row is O, and the structural rows are H, C, P, N.

Apply the obligation-surface theorem Definition 798.3. It exhausts the visible rows of the packet before any downstream consumer reads the route choice.

Use the obstruction-boundary theorem Definition 798.12. It keeps O separate from the bundle-consumption route, so an uninhabited obstruction cannot be reclassified as a closed subject-reduction proof.

Finally apply nonescape Definition 798.6 and obstruction-route exhaustion Definition 798.13. These rule out a hidden typeclass route, quotient syntax equality, host evaluator soundness, closed consistency, and any third discharge path. □

Lean4 ✓ BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceRouteObligationSurface

Theorem 798.5 (SubjectReductionRouteChoice MetaCIC scope)**.** *Every accepted* $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ *read remains inside the MetaCIC route of the discharge bundle, socket boundary, substitution/type-preservation rows, and the downstream subject-reduction consumer. The route may cite the sibling surfaces of chapter 718, chapter 719, and chapter 483, but it does not claim closed-CIC discharge or host proof search.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceCarrier metacic scope`

Proof. Use the obligation surface theorem above. It identifies the displayed rows and binds them to the finite route-choice packet before any consumer may proceed.

The sibling discharge and socket chapters provide the finite MetaCIC support surface, while the subject-reduction consumer chapter names the downstream use site. Because the packet has no coordinate for closed-CIC discharge or host proof search, those claims remain outside the scope. □

Theorem 798.6 (SubjectReductionRouteChoice nonescape)**.** *For an accepted* $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ *carrier* $R = (B, V, O, H, C, P, N)$*, no consumer can extract closed-CIC discharge, hidden route choice, or host proof search from the packet. The accepted route-choice data remain exactly the displayed bundle, consumption, obstruction, transport, replay, provenance, and local naming rows.*

Proof. Project the carrier. The route-choice packet consists exactly of the displayed rows B, V, O, H, C, P, N.

Apply the MetaCIC scope theorem of Definition 798.5. It keeps the route inside the sibling discharge, socket, substitution, and consumer surfaces.

Since no extra coordinate is present, the packet cannot export closed-CIC discharge, hidden route choice, or host proof search. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceCarrier nonescape`

Theorem 798.7 (SubjectReductionRouteChoice substitution stability)**.** *For an accepted* SubjectReductionRouteCh *carrier* $R = (B, V, O, H, C, P, N)$*, a MetaCIC worker may read substitution stability only as a finite route-choice obligation over the displayed* BHist *packet. The obligation exposes the bundle row* B*, the bundle-consumption row* V*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, and the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$ *row* N*.*

It states that substitution-facing reads must preserve the same route-choice rows before any subject-reduction discharge is consumed. It does not assert unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceSubstitutionStability`

Proof. Project the carrier of Definition 798.1. The only rows available to a substitution-facing consumer are the explicit bundle row B, the consumption row V, the obstruction boundary O, transport H, continuation C, provenance P, and naming N.

Apply the NameCert obligations of Definition 798.2. They require every read to keep route visibility for B, V, obstruction visibility for O, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Thus substitution stability is an obligation to preserve the displayed route-choice surface. Since the carrier has no quotient syntax-equality row, no host evaluator row, and no closed-consistency coordinate, no global MetaCIC theorem follows from this packet. □

Theorem 798.8 (SubjectReductionRouteChoice type-preservation window)**.** *For an accepted* SubjectReductionRou *carrier* $R = (B, V, O, H, C, P, N)$*, every bounded type-preservation window exposed to a MetaCIC worker is a finite* BHist *read through* B, V, H, C, P, N *with the obstruction row* O *still visible. The window records which discharge bundle route may be consumed and how* hsame *and* Cont *replay preserve that route inside the local* NameCert *surface.*

The window is not a proof of full subject reduction. It exports no unrestricted reduction theorem, quotient syntax equality, host evaluator soundness, or closed consistency claim.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceTypePreservationWindow`

Proof. Start from the accepted carrier. The route rows B, V identify the discharge bundle surface and its consumption route, while O records the boundary where the present substrate has not supplied the missing closed-CIC discharge.

The transport and continuation rows H, C keep the bounded read inside the same finite packet. The provenance and naming rows P, N then record that the worker is using the local $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$ surface rather than a detached evaluator theorem.

Consequently the type-preservation window is bounded by the displayed rows. It can tell a worker where a discharge bundle would be consumed, but it cannot export full subject reduction or any of the refused host-side principles. □

Theorem 798.9 (SubjectReductionRouteChoice blocked-edge nonescape)**.** *For an accepted* SubjectReductionRoute *carrier* $R = (B, V, O, H, C, P, N)$*, every blocked subject-reduction edge remains inside the obstruction row* O *and the finite replay rows* H, C, P, N*. A downstream MetaCIC worker may cite the edge only as a visible blocked route in the same* BHist *packet and must keep the local* $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$ *row attached.*

No blocked edge escapes as unrestricted subject reduction, quotient syntax equality, host evaluator soundness, closed consistency, or an inhabited discharge bundle.

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceBlockedEdgeNonescape`

Proof. Inspect the obstruction boundary in Definition 798.1. The row O is separate from the route rows B, V, so a blocked edge cannot be reclassified as an inhabited discharge route.

The NameCert obligations of Definition 798.2 keep every read under componentwise hsame transport, displayed Cont replay, provenance, and local naming. These rows preserve the blocked-edge status instead of creating a new discharge coordinate.

Therefore a downstream worker can use the blocked edge as finite route-choice information only. The refused global subject-reduction, quotient-equality, evaluator-soundness, and closed-consistency claims remain outside the packet. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceCarrier blocked edge nonescape`

Theorem 798.10 (SubjectReductionRouteChoice bundle-route exhaustion)**.** *Every consumer read of an accepted* $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ *packet reaches exactly one public route surface: the explicit bundle surface* B, V*. That surface exposes the four discharge coordinates named by the subject-reduction discharge interface, and no second route, hidden instance, or host conversion rule is exported by the packet.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceBundleRouteExhaustion`

Proof. Inspect the route coordinates in Definition 798.1. The explicit route is the pair B, V.

Since every consumer read factors through C and every transported read factors through H, no route coordinate outside the displayed bundle surface is available. □

Theorem 798.11 (SubjectReductionRouteChoice constructor exhaustion)**.** *Every accepted* SubjectReductionRoute *route-choice read is generated by exactly one displayed carrier packet $R = (B, V, O, H, C, P, N)$. Each consumer path is either the explicit bundle route through B, V with the obstruction row O still visible, or the blocked-edge route through O with B, V preserved as non-inhabited route data.*

Proof. Project the carrier rows of Definition 798.1. The displayed packet has exactly the route rows B, V, the obstruction row O, and the structural rows H, C, P, N.

Case-split on the public route surface versus the obstruction surface. In the first case, apply Definition 798.10: the consumer reaches the explicit bundle route B, V, while O remains a visible boundary row. In the second case, apply Definition 798.12: the read stops at O and keeps B, V only as route data not inhabited by the present substrate.

Finally use Definition 798.6. The rows H, C, P, N transport and replay only those displayed coordinates, so no hidden instance, host conversion rule, quotient syntax equality, evaluator soundness, or closed consistency claim is exported. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceConstructorExhaustion`

Theorem 798.12 (SubjectReductionRouteChoice obstruction boundary)**.** *The* SubjectReductionRouteChoice$^{\uparrow}$ *packet keeps the obstruction row O separate from the route rows. It may record that a downstream substrate can use the explicit bundle route, but it does not supply an inhabited* SubjectReductionDischargeBundle*, a proof of the dependent-codomain discharge row, or a closed subject-reduction theorem for the present substrate.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceObstructionBoundary`

Proof. Read the carrier rows in order. The route rows B, V name how a discharge would be consumed once supplied, while O names the boundary where the present substrate lacks the required discharge.

The provenance row ties this boundary to the discharge, socket, and subject-reduction chapters. Those references identify the finite interfaces, not inhabitants of the missing closed-CIC discharge.

Therefore the packet can expose the route choice to downstream audits while keeping closed subject reduction outside its exported surface. □

Theorem 798.13 (SubjectReductionRouteChoice obstruction-route exhaustion)**.** *Every accepted* SubjectReductionRouteChoice$^{\uparrow}$ *packet read factors through exactly one of the displayed bundle-consumption route B, V or the displayed obstruction row O, and both routes are transported only by the structural rows H, C, P, N. No third route, hidden typeclass instance, closed-CIC discharge, quotient syntax equality, host evaluator soundness, or closed consistency claim is exported by the packet.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceObstructionRouteExhaustio`

Proof. Project the carrier rows from Definition 798.1. The route-facing coordinates are B, V, the obstruction-facing coordinate is O, and all remaining rows are transport, replay, provenance, and local naming data.

Apply Definition 798.10. Any consumer that reads the displayed bundle route reaches the explicit surface B, V and no hidden instance or host conversion rule.

Apply Definition 798.12. Any consumer that does not receive an inhabited bundle route remains at the obstruction row O, where the closed-CIC discharge and closed subject-reduction theorem are not supplied.

Finally use Definition 798.6. The structural rows H, C, P, N preserve only the displayed bundle and obstruction coordinates, so quotient syntax equality, evaluator soundness, closed consistency, and any third route remain outside the packet. □

Theorem 798.14 (SubjectReductionRouteChoice confluence nonescape)**.** *Any confluence or subject-reduction consumer reading an accepted* SubjectReductionRouteChoice$^{\uparrow}$ *packet must keep the explicit bundle rows B, V, the obstruction row O, and the structural rows H, C, P, N visible. Thus local confluence, closed subject reduction, quotient syntax equality, evaluator soundness, and closed consistency cannot escape the route-choice boundary.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceConfluenceNonescape`

Proof. Start with the bundle route. By Definition 798.10, any consumer that reaches the displayed discharge route sees only the explicit rows B, V and no hidden instance or host conversion rule.

Next read the obstruction boundary. By Definition 798.12 and Definition 798.13, a consumer that lacks an inhabited bundle remains at the row O, while the structural rows H, C, P, N only transport and replay the same finite choice.

Finally apply Definition 798.6. Since every route is exhausted by B, V, O, H, C, P, N, no confluence, closed subject-reduction, quotient-equality, evaluator-soundness, or closed-consistency claim is exported from the packet. □

Theorem 798.15 (SubjectReductionRouteChoice vision handoff)**.** *The formal route of chapter 9374 is realized as the finite* SubjectReductionRouteChoice$^{\uparrow}$ *carrier of Definition 798.1. The handoff consists of the explicit bundle row, the consumption row, the obstruction-boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceVisionHandoff`

Proof. The vision chapter exposes the bundle route and the checked obstruction ledger. The carrier mirrors that boundary by placing B, V and O in separate displayed rows.

The route-choice packet also records O, so the handoff is not read as a closed subject-reduction proof for the present substrate. Consumers can cite the finite choice surface and must still obtain discharge data elsewhere. □

Theorem 798.16 (SubjectReductionRouteChoice TasteGate obligation handoff)**.** *Any obligation-closure read of* SubjectReductionRouteChoice$^{\uparrow}$ *must first expose the* TasteGate *single-carrier alignment row, the route-choice carrier rows B, V, O, H, C, P, N, and the MetaCIC obstruction rows consumed by the packet. The read is therefore a finite alignment and obstruction handoff; it is not a hidden subject-reduction discharge, quotient syntax equality, evaluator soundness theorem, or closed consistency claim.*

Proof. Start with the carrier of Definition 798.1. It displays the bundle route rows B, V, the obstruction row O, and the structural rows H, C, P, N as one packet.

Next use the obstruction boundary and route-exhaustion theorems, Definition 798.12 and Definition 798.13. They force every accepted consumer to choose between the displayed bundle route and the displayed obstruction row, with no third discharge channel.

Finally read the checked TasteGate alignment row. Since the alignment names the same single-carrier packet, an obligation-closure consumer must keep the route-choice rows and the MetaCIC obstruction rows visible before any later closure step can be claimed. □

Lean4 ✓ `BEDC.Derived.SubjectReductionRouteChoiceUp.SubjectReductionRouteChoiceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ packet is generated from explicit bundle, bundle-consumption, obstruction-boundary, transport, continuation, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 798.1, Definition 798.2, Definition 798.3, Definition 798.5, Definition 798.6, Definition 798.9, Definition 798.10, Definition 798.12, and the checked TasteGate alignment row bind the finite route-choice obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubjectReductionRouteChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not discharge the four MetaCIC subject-reduction obligations, construct an inhabited discharge bundle, prove closed subject reduction, or give Lean verification of the route-choice packet.

Upgrade path: Scoped closure requires binding the route-choice obligations to the discharge bundle, socket boundary, subject-reduction consumer, and obstruction rows.

Chapter 799

A Concrete Naming Certificate for $\mathsf{ClosedNormalCanonicity}^{\uparrow}$

The $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ packet turns the false-endpoint boundary of chapter 9376 into a finite NameCert surface. It is downstream of the main-result certificate in chapter 764, the closed-normal consistency packet of chapter 736, and the packet boundary of chapter 739. These siblings expose the closed beta-strong false endpoint and the subject-reduction gap; the present packet records the canonicity read that excludes hidden endpoint escape without claiming full closed consistency.

Definition 799.1 (Closed-normal canonicity carrier)**.** *A* $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, L, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the empty-context* HasType *row for the closed candidate at the displayed false encoding;*
2. *L is the* ClosedAt *row for the same candidate and false encoding;*
3. *B is the* BetaStrong *row saying the candidate is already at the closed normal boundary;*
4. *F names the false endpoint whose inhabitants are being excluded;*
5. *E is the endpoint-exclusion row consumed from the checked closed-normal beta-strong false boundary;*
6. *H records componentwise* hsame *transports for the typing, closedness, beta-strong, false-endpoint, and exclusion rows;*
7. *C records the displayed* Cont *routes from T, L, B, F to E and from E to the public read;*
8. *P records* Pkg *provenance over the main-result chapter and its closed-normal consistency packet siblings;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalCanonicity}^{\uparrow}}$ *row.*

The carrier has no coordinate for subject reduction, unrestricted normalization, closed consistency, host evaluation, quotient conversion, or a second false endpoint.

Lean4 ✓ `BEDC.Derived.ClosedNormalCanonicityUp.TasteGate.ClosedNormalCanonicityTasteGate single carrier alignment`

Theorem 799.2 (Closed-normal canonicity NameCert obligations)**.** *Every accepted* $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ *packet binds exactly the finite obligation surface of Definition 799.1: carrier admission for* $K = (T, L, B, F, E, H, C, P, N)$*, componentwise classifier stability on the displayed rows, ledger exactness for the closed typing, closedness, beta-strong normality, false endpoint, exclusion, transport, route, provenance, and naming coordinates, and local* $\mathsf{NameCert}$ *registration through* N*.*

Proof. Project the accepted carrier. The rows $T, L, B, F, E, H, C, P, N$ are the only visible coordinates, so carrier admission and classifier comparison are componentwise over the displayed packet.

Read the endpoint rows next. The rows T, L, B, F display the closed beta-strong candidate at the false endpoint, and E is the only exclusion row attached to that candidate.

The structural rows H, C, P, N transport, replay, package, and name the same finite surface. Since the carrier lists no further coordinate, the local NameCert obligations are exhausted by these rows. □

Theorem 799.3 (Closed-normal canonicity false-endpoint exclusion)**.** *Every consumer of an accepted* $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ *packet may read the false-endpoint exclusion only through the displayed chain*

$$T, L, B, F, E, H, C, P, N.$$

The read is a closed beta-strong endpoint exclusion for the named false encoding. It is not a full closed-consistency theorem, not a closed subject-reduction theorem, and not a normalization theorem for arbitrary closed terms.

Proof. Start with the exclusion row E. Its inputs in the carrier are exactly the typed closed beta-strong false endpoint rows T, L, B, F.

The continuation row C exposes the public read through the same displayed chain, while H, P, N provide transport, provenance, and local naming. No carrier coordinate supplies subject reduction, unrestricted normalization, or a general consistency principle.

Thus the consumer receives only the false-endpoint exclusion stated by the packet. □

Theorem 799.4 (Closed-normal canonicity non-escape)**.** *An accepted* $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ *packet has no escape route from its displayed endpoint surface. Any downstream read factors through* $T, L, B, F, E, H, C, P, N$*, and cannot acquire a second false endpoint, a host evaluator, quotient conversion, full closed consistency, closed subject reduction, or unrestricted normalization from the packet.*

Proof. Read the carrier of Definition 799.1. The packet contains one false endpoint row F and one exclusion row E.

Apply the obligation theorem to confine all public reads to the displayed rows. The transport, route, provenance, and local-name rows preserve that same finite packet instead of adding semantic coordinates.

The false-endpoint exclusion theorem then supplies the only positive read. Since no remaining coordinate is present, the excluded escape routes are unavailable to downstream consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalCanonicity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedNormalCanonicity}^{\uparrow}$ packet is generated from empty-context typing, closedness, beta-strong normality, a false-endpoint row, an endpoint-exclusion row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalCanonicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 799.1, Definition 799.2, Definition 799.3, and Definition 799.4 bind the finite false-endpoint canonicity surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalCanonicity}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full closed consistency, closed subject reduction, unrestricted normalization, host evaluation, quotient conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, endpoint exclusion, non-escape, provenance, and local naming for the same finite canonicity packet.

Chapter 800

A Concrete Naming Certificate for $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$

The $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$ packet realizes the closure-preservation audit map of chapter 9381 as a finite NameCert seal. It sits beside the substitution audit packet of **??**, the normalization audit seal of chapter 777, and the closed-normal consistency boundary of chapter 733: those siblings consume closed syntax movement and consistency-facing beta routes, while this seal records exactly where shift, substitution, beta-step, and beta-star closedness are available as finite audit rows.

Definition 800.1 (Closure-preservation audit seal carrier)**.** *A* $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R, V, U, B, Z, G, H, C, P, N)$$

with the following visible rows:

1. *A is the audit-map anchor row naming chapter 9381;*
2. *R is the shift row for* shift_one_preserves_closed_succ*;*
3. *V is the variable-substitution row for* substitute_var_preserves_closed_at*;*
4. *U is the full-substitution row for* substitute_preserves_closed_at*;*
5. *B is the one-step beta row for* betaStep_preserves_closed*;*
6. *Z is the finite beta-star row for* betaStarStep_preserves_closed*;*
7. *G is the* GAP *boundary row separating closedness preservation from typing preservation, conversion transport, global normalization, and confluence;*
8. *H records componentwise* hsame *transports for the anchor, the five closure theorem rows, the boundary row, route rows, provenance, and local naming;*
9. *C records the displayed* Cont *routes by which consumers read the shift, substitution, beta-step, beta-star, sibling-packet, and boundary rows;*
10. *P is the* Pkg *provenance over chapter 9381,* **??***, chapter 777, and chapter 733;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a HasType *preservation theorem, no subject-reduction discharge bundle, no quotient equality over beta paths, no host conversion oracle, and no route from beta-star closedness to a closed typed preservation theorem.*

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditSealUp.ClosurePreservationAuditSealTasteGate`
`single carrier alignment`

Theorem 800.2 (Closure-preservation audit seal NameCert obligations)**.** *For the carrier of Definition 800.1, the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (A, R, V, U, B, Z, G, H, C, P, N),$$

audit-anchor visibility through A, closed shift visibility through R, variable-substitution visibility through V, full-substitution visibility through U, beta-step visibility through B, beta-star visibility through Z, the displayed GAP *refusal through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed BHist packet. The audit anchor A fixes the source chapter, while R, V, U, B, Z are exactly the five closure-preservation theorem rows listed there.

Read the boundary row G next. It records that the seal is a closedness-classifier object: the packet may expose shift, substitution, beta-step, and beta-star closedness, but it does not expose typing preservation, conversion transport, normalization, confluence, or subject reduction.

Finally inspect H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite rows and add no hidden coordinate. Hence the displayed rows exhaust the local NameCert obligation surface. □

Theorem 800.3 (Closure-preservation audit seal route exactness)**.** *Every downstream read of an accepted* $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$ *packet that claims a closedness-preservation route factors through one of the displayed theorem rows R, V, U, B, Z, then through the matching* Cont *route in C, and then through the same* hsame, Pkg, *and* NameCert *rows H, P, N. In particular, a closed-normal or subject-reduction-boundary consumer may cite beta-star closedness only through Z together with the finite replay and provenance rows; it may not replace that route by a direct appeal to vision prose or by an unrecorded typing-preservation step.*

Proof. Start with an accepted packet and project the route component C. Its displayed continuations have sources only in R, V, U, B, Z and in the sibling-facing rows named by P.

For a shift or substitution read, the route reaches R, V, U and then uses the same transport row H before it is reported to a consumer. This is the route used by the substitution audit packet, so the consumer receives a finite closed-syntax movement row rather than an unrestricted substitution calculus.

For a beta-step or beta-star read, the route reaches B or Z and then passes through H, P, N. The closed-normal consistency boundary can therefore cite a finite beta closedness route, while the GAP row keeps subject-reduction discharge and typed conversion outside the route. □

Theorem 800.4 (Closure-preservation audit seal non-escape)**.** *Every public read of an accepted* $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$ *packet remains inside the finite rows*

$$A, R, V, U, B, Z, G, H, C, P, N.$$

The seal exports no HasType *preservation theorem, no closed subject-reduction theorem, no discharge of beta-substitution typing, application-argument typing, lambda-domain typing, or Pi-domain typing obligations, no quotient equality over reductions, and no host semantic conversion principle.*

Proof. The only positive theorem rows in the carrier are R, V, U, B, Z. They are closedness-preservation rows over the closedness classifier, and their replay routes are the displayed continuations in C.

The GAP row names the excluded crossings. It separates the finite closedness theorem surface from typing preservation, subject-reduction discharge, quotient equality, and host conversion.

Componentwise transport and provenance cannot create an omitted coordinate. Since H, P, N only preserve and name the rows already projected from S, every public read remains inside the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationAuditSeal}^{\uparrow}$ packet is generated from a closure-preservation audit-map anchor, five checked closedness theorem rows for shift, variable substitution, full substitution, beta-step, and beta-star routes, a displayed GAP boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 800.1, Definition 800.2, Definition 800.3, and Definition 800.4 bind the finite closure-preservation audit seal and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close typed subject reduction, typed conversion transport, normalization, confluence, quotient equality over reductions, Lean verification of the seal, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-row exactness, route replay, boundary refusal, provenance, local naming, and non-escape for the same finite seal packet.

Chapter 801

A Concrete Naming Certificate for $\mathsf{ClosurePreservationClassifier}^{\uparrow}$

The $\mathsf{ClosurePreservationClassifier}^{\uparrow}$ surface turns the MetaCIC closure-preservation audit map of chapter 9381 into a finite classifier-facing NameCert packet. It also reuses the generator-closure reading of chapter 486: a displayed generator row is accepted only when its action stays inside the displayed classifier boundary.

Definition 801.1 (Closure-preservation classifier carrier)**.** *A* $\mathsf{ClosurePreservationClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (K, S, U, B, F, H, C, P, N)$$

with the following visible coordinates:

1. *K is the* $\mathsf{ClosedAt}$ *boundary row, naming the depth-indexed closedness classifier consumed by the audit;*
2. *S is the shift-generator row, recording the displayed shift action and its closedness-preservation certificate against K;*
3. *U is the substitution-generator row, recording the displayed substitution action and its closedness-preservation certificate against K;*
4. *B is the beta-step route row, exposing the single-step preservation route whose redex case consumes U and whose congruence cases stay inside K;*
5. *F is the beta-star route row, exposing the finite iteration route obtained by replaying B over the displayed step ledger;*
6. *H records componentwise* hsame *transports for the closedness boundary, generator rows, beta-step route, beta-star route, package row, and local naming row;*
7. *C records the displayed* Cont *routes through which consumers inspect K, S, U, B, F, H, P, N;*
8. *P is the* Pkg *provenance over MetaCIC terms, depth rows,* $\mathsf{ClosedAt}$*, shift and substitution generators, beta-step and beta-star routes,* BHist*,* hsame*,* Cont*, and local naming;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationClassifier}^{\uparrow}}$ *row.*

The packet is a classifier packet for closedness preservation. It contains no typing-preservation row, no normalization theorem, no confluence theorem, no quotient term equality, and no host conversion procedure.

Lean4 ✓ BEDC.Derived.ClosurePreservationClassifierUp.ClosurePreservationClassifierTasteGate
single carrier alignment

Theorem 801.2 (Closure-preservation classifier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationClassifier}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for the packet* $X = (K, S, U, B, F, H, C, P, N)$ *of Definition 801.1;*
2. *classifier boundary admission, stating that every accepted read of the packet begins at the displayed* ClosedAt *row* K*;*
3. *generator preservation admission, stating that the shift row* S *and substitution row* U *are the only generator rows allowed to witness closure inside* K*;*
4. *beta route admission, stating that the beta-step row* B *and beta-star row* F *are read as finite replays of the displayed generator-preservation data;*
5. *non-escape and ledger exactness, stating that accepted consumers may inspect only* $K, S, U, B, F, H, C, P, N$ *and cannot import typing preservation, normalization, confluence, quotient terms, or host conversion.*

Proof. Carrier admission is the projection of the finite tuple in Definition 801.1. The tuple names the closedness classifier, the two generator rows, the beta routes, transports, continuations, provenance, and local certificate.

The classifier boundary row is K. Both generator rows are checked against that same row, so a public read cannot replace the closedness classifier by a typing classifier or by an ambient host predicate.

The beta rows are then read through the displayed generator data. The beta-step route B uses the substitution row for the redex case and componentwise closedness reads for congruence cases, while F repeats B over a finite route.

Finally inspect the carrier inventory. Since X has no typing-preservation, normalization, confluence, quotient, or host-conversion coordinate, the local certificate exports only the ClosedAt generator-closure audit surface. □

Theorem 801.3 (Closure-preservation classifier generator closure)**.** *Every accepted* $\mathsf{ClosurePreservationClassifier}^{\uparrow}$ *packet instantiates the GeneratorClosure reading for the* ClosedAt *classifier. Its shift row* S *and substitution row* U *are the generator witnesses, its beta-step row* B *and beta-star row* F *are the finite route witnesses, and its* H, C, P, N *rows record transport, continuation, provenance, and local naming. The accepted packet therefore exposes closedness preservation for the displayed generator routes without exporting typing preservation, normalization, confluence, quotient terms, or host conversion.*

Proof. Project the accepted packet $X = (K, S, U, B, F, H, C, P, N)$. The classifier coordinate is K, and the generator coordinates available to the GeneratorClosure reading are exactly S and U.

Read the beta-step route through B. Its redex branch uses the substitution generator row, and its structural branches replay closedness through the displayed classifier boundary. Read the beta-star route through F, which is a finite iteration of those accepted beta-step reads.

The transport and continuation rows H and C keep every read attached to the same displayed packet, while P and N record package provenance and local certificate membership. No remaining coordinate can provide typing preservation, normalization, confluence, quotient-term equality, or host conversion, so the conclusion is exactly the ClosedAt generator-closure surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationClassifier}^{\uparrow}$ packet is generated from a ClosedAt boundary row, shift and substitution generator rows, beta-step and beta-star route rows, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosurePreservationClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 801.1, Definition 801.2, and Definition 801.3 record the finite ClosedAt generator-closure packet and its local non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosurePreservationClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not export typing preservation, normalization, confluence, quotient terms, host conversion, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, ClosedAt classifier admission, shift and substitution generator preservation, beta-route replay, and non-escape for the displayed packet.

Chapter 802

A Closedness-Route Audit Surface for $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$

The $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ packet turns the closure-preservation audit map of chapter 9381 into a finite NameCert audit object. It sits between the row-level seal of chapter 800, the downstream subject-reduction socket of chapter 719, and the closed-normal boundary of chapter 740. The packet records where closedness travels through shift, substitution, beta-step, and beta-star routes; it does not add a typing-preservation route.

Definition 802.1 (MetaCIC closure-preservation audit packet)**.** *A* $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, S, R, V, U, B, Z, X, H, C, P, N)$$

with the following visible rows:

1. *A is the audit-map anchor naming chapter 9381;*
2. *S is the sibling audit-seal row from chapter 800;*
3. *R is the closed shift row for the shift-one closedness route;*
4. *V is the closed variable-substitution row;*
5. *U is the closed full-substitution row;*
6. *B is the closed one-step beta row;*
7. *Z is the closed finite beta-star row;*
8. *X is the subject-reduction frontier row, pointing to chapter 719 and chapter 740;*
9. *H records componentwise* hsame *transports for the anchor, seal, five closedness rows, frontier row, route rows, provenance, and local naming;*
10. *C records the displayed* Cont *routes by which consumers read the anchor, seal, closedness rows, beta-star export, frontier boundary, provenance, and local naming;*
11. *P is the* Pkg *provenance over the audit map, audit seal, subject-reduction socket, normal-form consistency seal, componentwise* hsame*, displayed* Cont*, and local* $\mathsf{NameCert}$ *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for closed subject reduction, full typed preservation, quotient equality of beta paths, host conversion, global normalization, or confluence.

Theorem 802.2 (MetaCIC closure-preservation audit finite obligation surface)**.** *For the carrier of Definition 802.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for*

$$M = (A, S, R, V, U, B, Z, X, H, C, P, N),$$

audit-map anchoring through A*, sibling audit-seal replay through* S*, the five closedness rows* R, V, U, B, Z*, the subject-reduction frontier row* X*, componentwise transport through* H*, displayed continuation reads through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed packet M. The audit anchor A fixes the source map, and S imports the finite audit-seal interface without replacing the local rows R, V, U, B, Z.

The five closedness rows are then read directly. They account for shift, variable substitution, full substitution, one-step beta, and finite beta-star closedness. The row X is separate: it points to the subject-reduction socket and the normal-form consistency seal as frontier consumers, not as extra closedness theorems.

Finally inspect H, C, P, N. Transport, route replay, provenance, and local naming preserve the rows already present in M. Since the carrier contains no other coordinate, the listed rows exhaust the finite obligation surface. □

Lean4 ✓ `BEDC.Derived.MetaCICClosurePreservationAuditUp.MetaCICClosurePreservationAuditTasteGate`
`single carrier alignment`

Theorem 802.3 (MetaCIC closure-preservation audit beta-star closedness export)**.** *Every accepted* $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ *packet exports beta-star closedness only through the displayed row* Z*, its componentwise* hsame *transport in* H*, its displayed* Cont *replay in* C*, the provenance row* P*, and the local naming row* N*. A consumer from chapter 740 may cite this export as a closedness row, but not as subject reduction, typed conversion transport, or closed consistency.*

Proof. Start with the packet projection and isolate the beta-star coordinate Z. The theorem-row inventory in Definition 802.2 names Z as the only beta-star closedness row.

Route replay then passes through H, C, P, N. These rows preserve and expose the same beta-star row; they do not introduce a typing derivation, a conversion rule, or a closed-consistency theorem.

For a normal-form consumer, the allowed read is therefore a finite closedness export. Any stronger claim would require a coordinate outside M, and the carrier definition provides none. □

Theorem 802.4 (MetaCIC closure-preservation audit subject-reduction frontier separation)**.** *Every subject-reduction-facing read of an accepted* $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ *packet factors through the frontier row* X *and then through the finite socket of chapter 719. The read may use the closedness rows* R, V, U, B, Z *as audit evidence, but it cannot manufacture a closed subject-reduction theorem, discharge the four subject-reduction rows, or bypass the normal-form boundary of chapter 740.*

Proof. Read the frontier coordinate X. Its role is to name the crossing from closure preservation to subject-reduction consumers, and its cited socket is the only displayed route for that crossing.

The closedness rows R, V, U, B, Z remain available as evidence rows. They are transported by H and replayed by C, but their conclusions are closedness statements, not typed preservation statements.

The normal-form consistency seal receives beta-star closedness only as a bounded input. Because the packet has no coordinate for closed subject reduction or full consistency, the frontier read stops at the socket and the normal-form boundary. □

Corollary 802.5 (MetaCIC closure-preservation audit strict local obstruction)**.** *Let an accepted* $\mathsf{MetaCICClosurePreservationAudit}^{\uparrow}$ *packet be read through the carrier*

$$M = (A, S, R, V, U, B, Z, X, H, C, P, N)$$

of Definition 802.1. Any consumer-facing negative use of its subject-reduction frontier is a strict local obstruction: it blocks only the converse route from closedness preservation to closed subject reduction, only the reuse of beta-star closedness as typed preservation, and only the bypass of the socket and normal-form boundary named in Definition 802.4. It cannot be reread as a global impossibility theorem, a host theorem, a countermodel to closure preservation, an external validation principle, or a hidden discharge of an unlisted coordinate.

Proof. Project the packet using Definition 802.1. Its positive closedness coordinates are exactly the rows R, V, U, B, Z, with Z singled out by Definition 802.3 as the beta-star closedness export. The carrier also displays the frontier row X, the componentwise transport row H, the continuation row C, the provenance row P, and the local naming row N.

The only subject-reduction-facing crossing supplied by the packet is the one identified in Definition 802.4: a read factors through X and then through the finite subject-reduction socket. The same theorem allows R, V, U, B, Z to be used as audit evidence, but it denies manufacture of closed subject reduction, discharge of the subject-reduction rows, and bypass of the normal-form boundary. Thus the obstruction applies to the attempted converse or reuse route that treats closedness evidence as typed preservation evidence.

No stronger negative reading is available from the finite carrier. The definition explicitly omits coordinates for closed subject reduction, full typed preservation, quotient equality of beta paths, host conversion, global normalization, and confluence. The transport, continuation, provenance, and naming rows H, C, P, N preserve access to displayed rows; they do not add a host theorem, a global countermodel, or an external validation channel. Therefore every negative consumer-facing use remains the named local obstruction and rejects any unlisted coordinate. □

Chapter 803

A Concrete Naming Certificate for ClosurePreservationAuditPacket$^{\uparrow}$

The ClosurePreservationAuditPacket$^{\uparrow}$ packet realizes the closure-preservation audit map of chapter 9381 as a finite NameCert object for consumers that need the whole closedness-preservation route rather than a single seal row. It sits beside the normal-form consistency seal of chapter 740, the subject-reduction discharge socket of chapter 719, the closed-term context seal of chapter 722, and the audit seal of chapter 800. Those sibling chapters expose closed-normal, subject-reduction, closed-context, and route-seal consumers; this packet binds the finite rows they may cite when they need shift, substitution, beta-step, beta-star, closedness, transport, route, provenance, and local naming data together.

Definition 803.1 (Closure-preservation audit packet carrier)**.** *A* ClosurePreservationAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, S, T, V, U, B, Z, F, R, H, C, P, N)$$

with the following visible rows:

1. *A is the audit-map anchor row naming chapter 9381;*
2. *L is the local* ClosedAt *ledger row for the terms and depths read by the packet;*
3. *S is the shift row for the displayed* shift *route;*
4. *T is the one-hole substitution row for the displayed* substitute *route;*
5. *V is the variable-substitution row for* substitute_var_preserves_closed_at*;*
6. *U is the full-substitution row for* substitute_preserves_closed_at*;*
7. *B is the one-step beta row for* betaStep *closedness preservation;*
8. *Z is the finite beta-star row for* betaStarStep *closedness preservation;*
9. *F is the finite replay row recording that beta-star evidence is read as a list of accepted beta-step witnesses;*
10. *R records the sibling routes to chapter 740, chapter 719, and chapter 722;*
11. *H records componentwise* hsame *transport for the anchor, closedness ledger, operation rows, replay row, sibling routes, provenance, and local naming;*

12. *C records the displayed* Cont *routes used by consumers of shift, substitution, beta-step, beta-star, sibling-facing, and boundary reads;*
13. *P is the* Pkg *provenance over the audit map and the named sibling chapters;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host conversion principle, a quotient of beta paths, a typing-preservation theorem, a subject-reduction discharge, global normalization, confluence, or an ambient semantic equality of closed terms.

Lean4 ✓ `BEDC.Derived.ClosurePreservationAuditPacketUp.ClosurePreservationAuditPacketTasteGate`
`single carrier alignment`

Theorem 803.2 (Closure-preservation audit packet NameCert obligations)**.** *For the carrier of Definition 803.1, the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (A, L, S, T, V, U, B, Z, F, R, H, C, P, N),$$

audit-map visibility through A, closedness-ledger visibility through L, shift visibility through S, substitution visibility through T, V, U, beta-step visibility through B, beta-star replay through Z, F, sibling-route visibility through R, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite BHist packet. The row A fixes the source audit map, while L, S, T, V, U, B, Z, F expose exactly the closedness, shift, substitution, beta-step, and beta-star rows named by the carrier.

Read the sibling route row R next. It names only the normal-form consistency seal, the subject-reduction discharge socket, the closed-term context seal, and their finite consumer routes; it does not turn those siblings into an additional closedness theorem or a typed subject-reduction result.

Finally inspect H, C, P, N. The hsame transports preserve the displayed rows componentwise, the Cont rows replay the same routes to consumers, the Pkg row records provenance, and the local NameCert row names the packet. Since no other carrier coordinate is projected, these rows exhaust the obligation surface. □

Theorem 803.3 (Closure-preservation audit packet beta-star boundary)**.** *Every consumer read of an accepted* $\mathsf{ClosurePreservationAuditPacket}^{\uparrow}$ *packet that cites beta-star closedness factors through the displayed beta-step row B, the finite replay row F, the beta-star row Z, the local* ClosedAt *ledger L, and then the common* hsame, Cont, Pkg, *and* NameCert *rows H, C, P, N. The read may pass to the normal-form consistency seal, subject-reduction discharge socket, or closed-term context seal only through the sibling-route row R; it exports no quotient beta path, no host conversion rule, no typed subject-reduction theorem, no global normalization theorem, and no confluence theorem.*

Proof. Start with an accepted packet and project the beta-star route. The carrier exposes beta-star evidence only as the finite pair F, Z, with F replaying accepted beta-step witnesses and Z recording the resulting closedness-preservation row.

The local closedness claim is tied to L. Shift and substitution side conditions are read through S, T, V, U, so the beta-star consumer cannot bypass the displayed ClosedAt ledger or replace it by a host equality of closed terms.

Now route the consumer-facing read. The continuation row C points from B, F, Z, L through the componentwise transport row H and provenance row P to the local naming row N.

It remains to check sibling access. The row R is the only coordinate that mentions the normal-form consistency seal, the subject-reduction discharge socket, or the closed-term context seal. Thus a public read can cite those siblings only as finite route endpoints, and the omitted quotient, conversion, typed subject-reduction, normalization, and confluence objects cannot escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationAuditPacket}^{\uparrow}$ packet is generated from a closure-preservation audit-map anchor, a finite closedness ledger, shift and substitution rows, beta-step and beta-star replay rows, sibling-route rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 803.1, Definition 803.2, and Definition 803.3 bind the finite closure-preservation audit packet and its beta-star consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosurePreservationAuditPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close typed subject reduction, host conversion, quotient equality over beta paths, normalization, confluence, Lean verification of the packet, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation-row exactness, beta-star replay, sibling-route exactness, provenance, local naming, and non-escape for the same finite packet.

Chapter 804

A Concrete Naming Certificate for DependentCodomainClosurePreservation$^{\uparrow}$

The DependentCodomainClosurePreservation$^{\uparrow}$ packet turns the closure-preservation audit map of chapter 9381 into a finite codomain-transport surface for subject-reduction routes. It sits beside the dependent-codomain inversion boundary of chapter 781: the sibling boundary names the point where codomain readback cannot be crossed for free, while this packet records exactly which closedness-preservation rows may be transported before that boundary is consumed.

Definition 804.1 (Dependent-codomain closure-preservation carrier)**.** *A* DependentCodomainClosurePreservation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\Pi, A, a, a', C_0, C_1, Z, S, I, T, L, H, R, P, N)$$

with the following displayed rows:

1. Π *is the dependent function-type row whose codomain is observed along the subject-reduction route;*
2. A *is the displayed domain row shared by the argument and the function type;*
3. a *and* a' *are the source and reduced argument rows;*
4. C_0 *and* C_1 *are the two codomain readback rows exposed by substituting* a *and* a' *into the same displayed codomain;*
5. Z *is the beta-star closedness row corresponding to* betaStarStep_preserves_closed*;*
6. S *is the substitution closedness row corresponding to* substitute_preserves_closed_at*;*
7. I *is the sibling inversion-boundary row supplied by chapter 781;*
8. T *is the finite transport row that carries closedness from the source codomain readback to the reduced codomain readback;*
9. L *is the ledger row recording that the transport is a closedness route and not a typing-preservation route;*
10. H *records componentwise* hsame *transport for the function type, domain, arguments, codomain readbacks, closure rows, inversion-boundary row, transport row, ledger row, provenance row, and local name;*

11. *R records displayed* Cont *replay routes for beta-star closedness, substitution closedness, sibling-boundary consumption, and the codomain transport read;*
12. *P is the* Pkg *provenance over chapter 9381, chapter 781, and the subject-reduction route surfaces that consume finite codomain transport;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DependentCodomainClosurePreservation}^{\uparrow}}$ *row.*

The carrier contains no coordinate for dependent-codomain inversion, no proof that C_0 and C_1 have the same type, no closed subject-reduction theorem, no host conversion principle, and no quotient equality over beta-star routes.

Lean4 (definition) `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservatio`

Theorem 804.2 (Dependent-codomain closure-preservation obligations)**.** *For the carrier of Definition 804.1, the local* $\mathsf{NameCert}_{\mathsf{DependentCodomainClosurePreservation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (\Pi, A, a, a', C_0, C_1, Z, S, I, T, L, H, R, P, N),$$

visibility of the dependent type, domain, source and reduced argument rows, paired codomain readbacks, beta-star closedness through Z, substitution closedness through S, sibling inversion-boundary access through I, finite codomain transport through T, closedness-only ledger exposure through L, componentwise hsame *transport through H, displayed* Cont *replay through R, package provenance through P, and local naming through N.*

Proof. Project the displayed packet. The rows Π, A, a, a', C_0, C_1 give the complete codomain-readback surface before any closure route is read.

Read the checked closedness rows next. The beta-star row Z and substitution row S are the only positive preservation rows, and both are closedness rows cited from chapter 9381.

Now consume the sibling boundary row I and the transport row T. They identify the finite point where dependent-codomain transport may be requested without turning the request into dependent-codomain inversion.

Finally inspect L, H, R, P, N. The ledger, transport, replay, provenance, and local naming rows preserve only the displayed coordinates, so the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationObligat`

Theorem 804.3 (Dependent-codomain closure-preservation non-escape)**.** *Every public read of an accepted* $\mathsf{DependentCodomainClosurePreservation}^{\uparrow}$ *packet remains inside*

$$\Pi, A, a, a', C_0, C_1, Z, S, I, T, L, H, R, P, N.$$

In particular, the packet exports no dependent-codomain inversion theorem, no typing preservation for dependent application, no context-reduction theorem, no host conversion rule, no closed subject-reduction theorem, and no quotient equality over beta-star routes.

Lean4 ✓ `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationNonesca`

Proof. The only theorem rows in the carrier are Z and S, and both are closedness-preservation rows. A consumer can reach them only through the displayed replay row R and the componentwise transport row H.

The inversion-boundary row I is consumed as a boundary, not as a discharge. It records where the subject-reduction route must still account for dependent-codomain readback, so it cannot supply typing preservation or context reduction.

The ledger row L marks the same restriction at the consumer boundary. Since no omitted coordinate is available through P or N, every public read remains in the displayed finite packet. □

Theorem 804.4 (Dependent-codomain closure-preservation transport)**.** *Let K be an accepted* DependentCodomainClosurePreservation$^{\uparrow}$ *carrier. Any subject-reduction route that transports codomain closedness through the dependent argument step must read beta-star closedness through Z, substitution closedness through S, the sibling inversion-boundary row through I, and the finite transport row through T before exposing the transported codomain-readback surface. The route may then hand the transported closedness row to downstream subject-reduction consumers, but only as closedness data over C_0 and C_1.*

Lean4 ✓ `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationTranspo`

Proof. Begin with a subject-reduction consumer requesting codomain transport. Carrier projection exposes both codomain readbacks C_0 and C_1 together with the argument rows that generated them.

Use the beta-star and substitution rows from the closure-preservation audit map. These rows authorize closedness movement along the displayed beta-star and substitution routes, so the requested transport has a finite source.

Next read the inversion-boundary row. The sibling boundary states that codomain readback itself is the remaining dependent-codomain obligation, so the transport row T can move closedness across the displayed readbacks without proving their typing equality.

The route is then replayed through R and transported through H. Provenance and local naming keep the result attached to the same finite carrier, so downstream consumers receive only the transported closedness row. □

Theorem 804.5 (Dependent-codomain closure-preservation ledger)**.** *For every accepted* DependentCodomainClosur *carrier, the ledger consumed by a subject-reduction route is exactly the closedness-only row L attached to Z, S, I, T and the paired codomain readbacks C_0, C_1. No alternate ledger records typed conversion, dependent-codomain inversion, quotient equality, ambient normalization, confluence, or closed subject reduction.*

Lean4 ✓ `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationPacket closedness ledger exactness`

↪ variant `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationLedger`

Proof. Inspect the consumer route. It first reaches the paired codomain readbacks, then the beta-star and substitution closedness rows, then the sibling boundary and transport rows.

The ledger row L records precisely that route. It states that T is a closedness transport over the displayed codomain-readback pair and that I remains a boundary for the stronger dependent-codomain claim.

The non-escape theorem rules out every other consumer coordinate. Thus the ledger exposed by the packet is exactly the finite closedness-only row L. □

Lean4 ✓ `BEDC.Derived.DependentCodomainClosurePreservationUp.DependentCodomainClosurePreservationPacket ledger scope`

Closure status for NameCert$_{\text{DependentCodomainClosurePreservation}^{\uparrow}}$.

Constructive story: A DependentCodomainClosurePreservation$^{\uparrow}$ packet is generated from dependent function-type, domain, argument, paired codomain-readback, beta-star closedness, substitution closedness, sibling inversion-boundary, finite transport, closedness ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{DependentCodomainClosurePreservation}^{\uparrow}}$, seed)

Scope closed: Definition 804.1, Definition 804.2, Definition 804.3, Definition 804.4, and Definition 804.5 bind the finite closedness transport surface for dependent-codomain subject-reduction routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DependentCodomainClosurePreservation}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove dependent-codomain inversion, typed subject reduction, context reduction, quotient equality over beta-star routes, Lean verification of the packet, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, beta-star and substitution row coverage, sibling-boundary consumption, finite transport exactness, ledger non-escape, provenance, and local naming for the same packet.

Chapter 805

A Concrete Naming Certificate for $\mathsf{ClosurePreservationRouteLedger}^{\uparrow}$

The $\mathsf{ClosurePreservationRouteLedger}^{\uparrow}$ packet turns the audit map of chapter 9381 into a finite route ledger. It sits beside the substitution packet of chapter 779 and the subject-reduction socket of chapter 719: the former supplies the syntactic movement rows, while the latter records why closedness routes do not discharge typed subject reduction.

Definition 805.1 (Closure-preservation route ledger carrier)**.** *A* $\mathsf{ClosurePreservationRouteLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (C, S, U, B, F, H, Q, P, N)$$

with the following visible rows:

1. *C is the* $\mathsf{ClosedAt}$ *source row recording the closed input boundary;*
2. *S is the shift-preservation row for* $\mathsf{shift_one_preserves_closed_succ}$*;*
3. *U is the substitution-preservation row covering the variable and full-substitution closedness routes;*
4. *B is the beta-step preservation row for* $\mathsf{betaStep_preserves_closed}$*;*
5. *F is the beta-star preservation row for* $\mathsf{betaStarStep_preserves_closed}$*;*
6. *H records componentwise* hsame *transports for the source, preservation, route, provenance, and naming rows;*
7. *Q records the displayed* Cont *routes by which consumers reach the shift, substitution, beta-step, and beta-star rows from the source row;*
8. *P is the* Pkg *provenance over chapter 9381, chapter 779, chapter 800, and chapter 719;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationRouteLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for typed subject reduction, host conversion, an unrestricted substitution calculus, or a full consistency theorem.

Theorem 805.2 (Closure-preservation route ledger NameCert obligations)**.** *For the carrier of Definition 805.1, the local* $\mathsf{NameCert}_{\mathsf{ClosurePreservationRouteLedger}^{\uparrow}}$ *surface consists exactly of five obligation families: source closedness admission through C, shift preservation through S, substitution*

preservation through U, beta-step preservation through B, and beta-star preservation through F. The auxiliary rows H, Q, P, N transport, replay, account for, and name those five families without adding a sixth preservation claim.

Proof. Project the carrier $L = (C, S, U, B, F, H, Q, P, N)$. The source row C fixes the displayed ClosedAt boundary, and the four route rows S, U, B, F are the closedness-preservation routes named by chapter 9381.

Read the transport and route rows next. The hsame row H can only identify corresponding displayed coordinates, and the Cont row Q can only replay the four listed preservation routes from the source boundary.

Finally inspect provenance and naming. The package row P records the audit-map and sibling-packet sources, while N names the local certificate. Since no further coordinate is present, the five obligation families exhaust the local NameCert surface. □

Theorem 805.3 (Closure-preservation route ledger non-escape)**.** *Every public read of an accepted* $\mathsf{ClosurePreservationRouteLedger}^{\uparrow}$ *packet remains inside*

$$C, S, U, B, F, H, Q, P, N.$$

It does not export typed subject reduction, host conversion, an unrestricted substitution calculus, or full consistency.

Proof. The only positive preservation rows in the carrier are S, U, B, F, and each is a closedness-preservation row over the displayed source boundary C.

The substitution route is therefore the finite audited route through U, not a calculus for arbitrary substitution outside the listed closedness hypotheses. The beta-step and beta-star routes similarly preserve ClosedAt and do not introduce typing or conversion coordinates.

Rows H, Q, P, N preserve access discipline for the same finite packet. They transport, replay, account for, and name the visible rows, so typed subject reduction, host conversion, unrestricted substitution, and full consistency remain outside the public surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosurePreservationRouteLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosurePreservationRouteLedger}^{\uparrow}$ packet is generated from a ClosedAt source row, shift, substitution, beta-step, and beta-star preservation rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosurePreservationRouteLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 805.1, Definition 805.2, and Definition 805.3 bind the finite closure-preservation route ledger and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosurePreservationRouteLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close typed subject reduction, host conversion, unrestricted substitution, full consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route replay, provenance, local naming, and public non-escape for the same finite closure-preservation ledger.

Chapter 806

A Concrete Naming Certificate for $\mathsf{AuditMapInterface}^{\uparrow}$

The $\mathsf{AuditMapInterface}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite interface rather than as a prose convention. It sits beside the kernel acceptance trace of chapter 557, whose rows expose accepted theorem inventory, and the substitution audit map of chapter 779, whose rows show how a local MetaCIC substrate is kept separate from stronger typing claims. The present packet names the common interface consumed by audit-map ledgers, synthesis packets, and cross-map chapters.

Definition 806.1 (Audit-map interface carrier)**.** *An* $\mathsf{AuditMapInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, K, O, F, X, H, C, P, N)$$

with the following visible rows:

1. *E is the established theorem inventory row, listing the positive results that the local audit map may cite directly;*
2. *K is the conditional theorem inventory row, listing results available only through named discharge, setup, or route data;*
3. *O is the obstruction row, recording no-go, boundary, or shape facts that block a stronger local reading;*
4. *F is the frontier row, recording the strongest adjacent theorem or package that remains outside the local map;*
5. *X is the cross-map consumer row, naming neighbouring audit maps, synthesis packets, and route chapters that may consume the visible rows;*
6. *H records componentwise* hsame *transport for the established, conditional, obstruction, frontier, cross-map, provenance, and naming rows;*
7. *C records displayed* Cont *routes by which consumers read the positive inventory, conditional inventory, obstruction boundary, frontier handoff, cross-map use, provenance, and local name;*
8. *P is the* Pkg *provenance over chapter 9406, chapter 557, chapter 779,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$ *row.*

The carrier has no coordinate for turning conditional rows into unconditional theorems, obstruction rows into solved obligations, frontier rows into closed packages, or cross-map consumer rows into hidden module imports.

Lean4 (definition) `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceUp`
Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceUp single carrier alignment`

Theorem 806.2 (Audit-map interface NameCert obligations)**.** *For the carrier of Definition 806.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$ *surface consists of five obligations: carrier admission for*

$$I = (E, K, O, F, X, H, C, P, N),$$

inventory exactness for the established, conditional, obstruction, frontier, and cross-map rows E, K, O, F, X*, componentwise transport stability through* H*, displayed route readback through* C*, and package provenance with local naming through* P, N*.*

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier namecert obligations`

Proof. Carrier admission is the displayed finite packet. Its first five coordinates are the only audit-map row classes, so inventory exactness is a projection of E, K, O, F, X from the carrier.

Transport stability is componentwise. The row H may compare an established row with an established row, a conditional row with a conditional row, and similarly for obstruction, frontier, cross-map, provenance, and naming data; it supplies no cross-class coercion.

Route readback is then forced through C. A consumer must choose one of the displayed routes before using a row, and the route returns to the same inventory coordinate that it reads.

Finally, P and N bind provenance and local naming. Since provenance cites only the visible methodology and sibling concrete packets, the five obligations exhaust the NameCert surface without adding a stronger theorem coordinate. □

Theorem 806.3 (Audit-map interface cross-map consumer boundary)**.** *Every cross-map consumer of an accepted* $\mathsf{AuditMapInterface}^{\uparrow}$ *carrier*

$$I = (E, K, O, F, X, H, C, P, N)$$

must read the positive inventory E*, conditional inventory* K*, obstruction row* O*, frontier row* F*, and cross-map consumer row* X *through the displayed transport, continuation, provenance, and naming rows* H, C, P, N*. Such a consumer cannot turn the interface into an unconditional theorem, a discharged obstruction, a closed frontier package, a hidden import of a neighbouring map, or a host semantic equality.*

Proof. Start with the NameCert obligations. They identify the accepted interface with the finite rows $E, K, O, F, X, H, C, P, N$ and force each consumer read through the displayed route row C.

For a positive read, the consumer obtains exactly the established inventory row E. For a conditional read, it obtains the conditional row K together with its visible route requirement. For an obstruction read, it obtains the boundary row O rather than a proof of the blocked claim.

For a frontier or cross-map read, the consumer follows F or X through the same transport and provenance rows. Those rows name neighbouring maps and handoffs; they do not import their hidden contents or solve their obligations.

The carrier has no coordinate for unconditional discharge, solved obstruction, closed frontier package, hidden module import, or host semantic equality. Therefore every cross-map use remains an interface read and cannot become a stronger theorem. □

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier cross map consumer boundary`

Theorem 806.4 (Audit-map interface conditional row discharge boundary)**.** *For every accepted* $\mathsf{AuditMapInterface}^{\uparrow}$ *carrier*

$$I = (E, K, O, F, X, H, C, P, N),$$

a consumer may use a conditional inventory row K only together with its displayed discharge, setup, or route datum read through the continuation, transport, provenance, and local naming rows C, H, P, N. The same read cannot promote K into the established inventory row E or into an unconditional theorem coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier conditional row discharge boundary`

Proof. Project the finite carrier of Definition 806.1. The rows E and K are separate BHist coordinates: E records established theorem inventory, while K records conditional inventory.

Use the NameCert obligations of Definition 806.2. Inventory exactness keeps the two rows distinct, componentwise hsame transport in H preserves the row class being transported, and displayed Cont replay in C returns to the same coordinate it reads.

Finally apply the cross-map consumer boundary of Definition 806.3. A downstream consumer can carry the conditional row through C, H, P, N, but the carrier has no coordinate that reclassifies the conditional row as E or supplies an unconditional theorem. Therefore every use of K remains tied to the displayed discharge, setup, or route datum. □

Theorem 806.5 (Audit-map interface obstruction row non-escape)**.** *For every accepted* $\mathsf{AuditMapInterface}^{\uparrow}$ *carrier $I = (E, K, O, F, X, H, C, P, N)$, an obstruction row O remains a boundary row under componentwise* hsame *transport and displayed* Cont *replay. No route through H, C, P, N can convert O into a solved obligation, an established theorem row, or a closed package.*

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier obstruction row nonescape`

Proof. Read O from the carrier in Definition 806.1. It is the only coordinate whose role is to record no-go, boundary, or shape facts blocking a stronger local reading.

The NameCert obligations of Definition 806.2 make the administrative rows transparent. The row H transports obstruction data as obstruction data, while C only replays a displayed route to the row being read; P and N record provenance and local naming for the same finite packet.

Invoke Definition 806.3. It explicitly forbids turning the interface into a discharged obstruction or closed frontier package. Since neither H nor C introduces a theorem-producing coordinate, obstruction reads remain boundary reads for every consumer. □

Theorem 806.6 (Audit-map interface frontier consumer routing)**.** *For every accepted* $\mathsf{AuditMapInterface}^{\uparrow}$ *carrier $I = (E, K, O, F, X, H, C, P, N)$, every consumer route that uses the frontier row F or cross-map row X must pass through visible provenance, displayed continuation, componentwise transport, and local naming in P, C, H, N. Such a route may name neighbouring audit maps and handoffs, but it cannot import hidden rows from them or replace the finite interface by host semantic equality.*

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier frontier consumer transport routing`

Proof. Begin with the carrier definition. The rows F and X are visible BHist coordinates: F records the strongest adjacent theorem or package outside the local map, and X names neighbouring audit maps, synthesis packets, and route chapters that may consume the interface.

Apply Definition 806.2. It forces frontier and cross-map readback through the same finite NameCert surface, with H for componentwise transport, C for displayed continuation replay, P for package provenance, and N for local naming.

Now use Definition 806.3. Frontier and cross-map reads are allowed only as interface reads. Because the carrier lacks hidden import, closed frontier package, and host semantic equality coordinates, neighbouring maps can be routed only through the visible provenance attached to F and X. □

Theorem 806.7 (Audit-map interface row-class exhaustion)**.** *For every accepted* AuditMapInterface$^{\uparrow}$ *carrier*

$$I = (E, K, O, F, X, H, C, P, N),$$

each consumer-visible inventory row is exactly one of the established, conditional, obstruction, frontier, or cross-map rows E, K, O, F, X. The administrative rows H, C, P, N provide componentwise hsame *transport, displayed* Cont *readback,* Pkg *provenance, and local* NameCert *naming for those rows, but cannot reclassify any row class.*

Proof. Project the carrier of Definition 806.1. Its consumer inventory surface is the five-row list E, K, O, F, X, while H, C, P, N are recorded after that list as transport, route, provenance, and naming data.

Apply Definition 806.2. Inventory exactness is stated only for E, K, O, F, X, and componentwise transport through H is already restricted to the row class being transported.

Use Definition 806.3, Definition 806.4, Definition 806.5, and Definition 806.6. These boundary theorems rule out reclassifying a conditional row as established, an obstruction row as solved, a frontier row as closed, or a cross-map row as a hidden import. Hence the five listed row classes exhaust the consumer-visible inventory. □

Theorem 806.8 (Audit-map interface route-readback totality)**.** *For every accepted* AuditMapInterface$^{\uparrow}$ *carrier $I = (E, K, O, F, X, H, C, P, N)$, each displayed continuation route read through C returns to the same row class that it reads. Established, conditional, obstruction, frontier, and cross-map reads therefore remain separated under* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Start from Definition 806.2. Displayed route readback is one of the five NameCert obligations, and it is attached to the finite carrier rather than to an external theorem coordinate.

The row-class exhaustion of Definition 806.7 identifies the possible consumer inventory rows as E, K, O, F, X. Since H is componentwise and C is displayed for the row being read, a route that starts at one row class has no carrier coordinate through which it can return as another.

The separation clauses are exactly the existing boundaries. The cross-map boundary of Definition 806.3, the conditional boundary of Definition 806.4, the obstruction non-escape theorem of Definition 806.5, and the frontier routing theorem of Definition 806.6 leave route readback as an interface read for the same row class. □

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier route readback totality`

Theorem 806.9 (Audit-map interface template instantiation boundary)**.** *Any audit-map template or sibling audit packet instantiating the* $\mathsf{AuditMapInterface}^{\uparrow}$ *interface must supply the nine visible rows*

$$E, K, O, F, X, H, C, P, N.$$

The instantiated packet receives interface reads through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data only; it receives no hidden imports, solved obstruction coordinates, discharged theorem coordinates, closed frontier coordinates, or host semantic equality.*

Proof. The carrier definition fixes the required packet shape. An instantiation must therefore provide the five inventory rows E, K, O, F, X and the four administrative rows H, C, P, N before it can claim the local $\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$ surface.

Apply the NameCert obligations of Definition 806.2. They give carrier admission, inventory exactness, componentwise transport, route readback, provenance, and local naming for this nine-row packet, and they do not add any extra theorem coordinate.

Now combine Definition 806.3, Definition 806.8, Definition 806.4, Definition 806.5, and Definition 806.6. A sibling packet may consume the interface only through the displayed rows and routes, so template instantiation cannot smuggle in hidden imports or already discharged theorem data. □

Lean4 ✓ `BEDC.Derived.AuditMapInterfaceUp.AuditMapInterfaceCarrier template instantiation boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapInterface}^{\uparrow}$ packet is generated from established theorem inventory, conditional theorem inventory, obstruction, frontier, cross-map consumer, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 806.1, Definition 806.2, Definition 806.3, Definition 806.4, Definition 806.5, Definition 806.6, Definition 806.7, Definition 806.8, and Definition 806.9 bind the mature audit-map interface examples and cross-chapter consumer surface for template rows, sibling packet consumers, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AuditMapInterface}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close any conditional audit-map theorem, solve obstruction rows, close frontier packages, import neighbouring maps, provide Lean verification, or supply a public bridge schema.

Upgrade path: Further verification and bridge checking remain independent of the mature paper-side audit-map interface surface.

Chapter 807

A Concrete Naming Certificate for $\mathsf{AuditMapFamily}^{\uparrow}$

The $\mathsf{AuditMapFamily}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite NameCert surface. It sits beside the substitution audit map of chapter 779, the MetaCIC open-obligation ledger of chapter 763, and the subject-reduction discharge socket of chapter 719: those chapters supply concrete local rows, while this chapter binds the family-level inventory, classifier boundary, and template coverage that downstream MetaCIC audit consumers may cite.

Definition 807.1 (Audit-map family carrier)**.** *An* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E, C, O, F, X, H, R, P, N)$$

with the following visible rows:

1. *E is the established row, listing checked or otherwise available audit-map content such as closed substitution rows and closed-normal consistency rows;*
2. *C is the conditional row, listing results that are usable only through named routes or sockets, including subject-reduction discharge data;*
3. *O is the obstruction row, listing no-escape or no-claim boundaries that must stay visible to consumers;*
4. *F is the frontier row, listing open handoff points for adjacent MetaCIC modules without treating them as theorem exports;*
5. *X is the cross-module row, linking local audit maps, ledgers, and sockets into the family inventory;*
6. *H records componentwise* hsame *transports for established, conditional, obstruction, frontier, cross-module, route, provenance, and local naming rows;*
7. *R records the displayed* Cont *routes by which a consumer reads the family rows;*
8. *P is the* Pkg *provenance over chapter 779, chapter 763, chapter 719, and neighbouring MetaCIC audit chapters;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamily}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are established, conditional, obstruction, frontier, cross-module, transport, route, provenance, and local NameCert rows. It exports no hidden

proof of a conditional row, no erasure of an obstruction row, no full closed-consistency theorem, no ambient confluence theorem, and no host semantic equality coordinate.

Theorem 807.2 (Audit-map family NameCert obligations)**.** *For the carrier of Definition 807.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (E, C, O, F, X, H, R, P, N),$$

established-row readback through E, conditional-row readback through C, obstruction-row readback through O, frontier-row readback through F, cross-module readback through X, componentwise transport through H, displayed continuation through R, package provenance through P, and local naming through N.

Proof. Project the finite packet A. The rows E, C, O, F expose the four audit classes before any cross-module use is allowed, so the classifier can distinguish available content, routed content, blocked content, and frontier content.

Read X, H, R next. The cross-module row connects the local audit maps only as displayed neighbours, while transport and continuation replay those same rows rather than adding a theorem-producing coordinate.

Finally inspect P and N. Provenance cites the concrete substitution, open-obligation, and subject-reduction socket chapters as row sources, and the local NameCert row names the packet. Thus the NameCert surface is exhausted by the displayed inventory. □

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier namecert obligations`

Theorem 807.3 (Audit-map family boundary non-escape)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier $A = (E, C, O, F, X, H, R, P, N)$, every downstream consumer read remains inside the displayed established, conditional, obstruction, frontier, cross-module,* hsame, Cont, Pkg, *and local* NameCert *rows. In particular, the packet cannot turn a conditional subject-reduction route into a closed theorem, erase an obstruction row, close a frontier row, or export host-level semantic equality.*

Proof. Start from the carrier projection. A consumer can enter the family only through the rows $E, C, O, F, X, H, R, P, N$.

Use the NameCert obligations of Definition 807.2. The obligations keep each audit class tied to its own coordinate, and the continuation routes pass through R without changing a conditional, obstruction, or frontier row into established content.

The provenance row then fixes the boundary. Since P cites concrete sibling chapters only as visible row sources and the packet has no coordinate for hidden closure, every excluded endpoint remains outside the accepted carrier. □

Theorem 807.4 (Audit-map family template coverage)**.** *Every audit map covered by an accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier is represented by the family template: established content appears in E, conditional content appears in C, obstruction content appears in O, frontier content appears in F, and cross-module use appears in X. The transport, route, provenance, and local NameCert rows H, R, P, N preserve this coverage as finite BEDC data for downstream consumers.*

Proof. Read the local audit-map row through X. The family methodology requires exactly the four content classes and the cross-module use class, and the carrier assigns those classes to E, C, O, F, X.

For a concrete sibling such as chapter 779, the available closed substitution rows land in E, while subject-reduction-facing boundaries land outside established content. For chapter 763, open

obligations land in conditional, obstruction, or frontier rows according to their destination boundary. For chapter 719, the four discharge rows and the socket boundary land in C and O rather than in a closed theorem row.

The remaining rows preserve the classification. Componentwise hsame transport, displayed Cont routes, package provenance, and local naming make the same finite coverage readable by downstream consumers without adding another audit class. □

Chapter 808

A Concrete Naming Certificate for $\mathsf{AuditMapTemplate}^{\uparrow}$

The $\mathsf{AuditMapTemplate}^{\uparrow}$ packet realizes the audit-map family template of chapter 9406 as finite NameCert data. It sits beside the substitution audit map of chapter 779, the subject-reduction route triangle of chapter 796, and the closure-preservation audit trace of chapter 792: those siblings show concrete map, route, and trace packets, while this chapter records the reusable finite packet shape that makes an audit map inspectable without treating methodology as a theorem endpoint.

Definition 808.1 (AuditMapTemplate carrier)**.** *An* $\mathsf{AuditMapTemplate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, E, C, O, F, R, B, H, Q, P, N)$$

with the following displayed rows:

1. *S is the setup row naming the substrate, local vocabulary, carrier rows, and admissible consumer interface for the audit map;*
2. *E is the established-inventory row listing the checked or already certified facts that the map may expose as local content;*
3. *C is the conditional-inventory row listing parameterized routes, setup fields, discharge bundles, or interface hypotheses that remain explicit;*
4. *O is the obstruction-inventory row listing no-go facts, missing inhabitants, boundary counterexamples, or shape restrictions that block a stronger reading;*
5. *F is the frontier row naming the strongest nearby claim that is not licensed by S, E, C, O;*
6. *R is the cross-map route row naming the neighbouring audit maps, route certificates, trace packets, and synthesis chapters that may consume or supply the displayed rows;*
7. *B is the boundary row stating which claims are outside the packet even after all displayed rows have been read;*
8. *H records componentwise* hsame *transports for setup, established inventory, conditional inventory, obstruction inventory, frontier, cross-map route, boundary, provenance, and local naming rows;*
9. *Q records displayed* Cont *routes by which a consumer inspects the setup, inventory, obstruction, frontier, route, and boundary rows without adding hidden coordinates;*

10. *P is the* Pkg *provenance over the source audit-map methodology, sibling concrete packets,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapTemplate}^{\uparrow}}$ *row.*

The classifier accepts exactly packets carrying these eleven rows. It exports no proof of the frontier row, no inhabitant for any conditional row, no erasure of the obstruction row, no automatic bridge between neighbouring maps, and no host semantic equality coordinate.

Theorem 808.2 (AuditMapTemplate NameCert obligations)**.** *The carrier of Definition 808.1 satisfies the local* $\mathsf{NameCert}_{\mathsf{AuditMapTemplate}^{\uparrow}}$ *obligation surface. Its obligations are carrier admission for*

$$A = (S, E, C, O, F, R, B, H, Q, P, N),$$

setup readback through S, established-inventory readback through E, conditional-inventory readback through C, obstruction-inventory readback through O, frontier readback through F, cross-map route readback through R, boundary readback through B, componentwise transport through H, displayed continuation through Q, provenance through P, and local naming through N.

Proof. Carrier admission is the displayed finite BHist packet. Projecting S, E, C, O exposes the audit map's setup, positive inventory, parameterized inventory, and obstruction inventory as separate rows rather than as blended prose.

Project F, R, B next. The frontier row names the nearby claim left outside the packet, the cross-map route row names the neighbouring maps that may read or supply rows, and the boundary row records the claims that remain unavailable even after the route is known.

Finally inspect H, Q, P, N. These rows transport, replay, locate, and name the same finite packet. Since every obligation is one of the displayed rows and no hidden row is supplied by transport, continuation, provenance, or naming, the local NameCert surface is exactly the listed carrier inventory. □

Theorem 808.3 (AuditMapTemplate frontier non-closure)**.** *For an accepted* $\mathsf{AuditMapTemplate}^{\uparrow}$ *carrier $A = (S, E, C, O, F, R, B, H, Q, P, N)$, the frontier row F is visible as a boundary object and not as a closed theorem. A consumer may cite F to identify the strongest nearby claim outside S, E, C, O, but the packet supplies neither a proof of F nor an inhabitant that discharges the conditional or obstruction rows.*

Proof. Read the carrier by projection. The rows S, E, C, O describe the licensed setup, established facts, conditional interfaces, and obstruction facts available before the frontier is inspected.

The frontier coordinate F is a row of the packet, not a theorem-constructor for the frontier claim. Its role is to mark where the displayed inventory stops.

The remaining coordinates R, B, H, Q, P, N route, bound, transport, replay, locate, and name that same packet. None of those coordinates has the type of a frontier proof or of a discharge witness for C or O. Thus an accepted packet makes non-closure explicit rather than silently closing the frontier. □

Theorem 808.4 (AuditMapTemplate cross-map routing)**.** *Every accepted* $\mathsf{AuditMapTemplate}^{\uparrow}$ *carrier exposes cross-map routing only through the displayed row R and its supporting rows H, Q, P, N. Any consumer that moves from the template packet to a concrete sibling, such as the packets of chapter 779, chapter 796, or chapter 792, must preserve the setup, established, conditional, obstruction, frontier, and boundary rows named by S, E, C, O, F, B.*

Proof. Start with the route coordinate R. It names neighbouring maps and consumers, but it is still a displayed row of the same finite packet.

The supporting coordinates H, Q, P, N control how the route is transported, replayed, sourced, and named. They do not rewrite the setup row, add facts to the established-inventory row, discharge the conditional row, delete the obstruction row, prove the frontier row, or weaken the boundary row.

Therefore cross-map movement is a projection-and-replay operation over the visible packet. A sibling concrete chapter can consume the template only by reading the rows it displays, so routing preserves the audit-map separation between licensed content, conditions, obstructions, frontier, and boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapTemplate}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapTemplate}^{\uparrow}$ packet is generated from finite BHist rows for setup, established inventory, conditional inventory, obstruction inventory, frontier, cross-map route, boundary, transports, displayed continuations, provenance, and local naming. The template makes audit-map movement a projection-and-replay operation: Cont exposes the setup, inventories, route, and boundary without adding hidden discharge rows. Componentwise hsame preserves the same row separation, while Pkg records the methodology and sibling concrete packets that may consume the template. The packet is a reusable finite map shape, not a theorem proving its frontier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapTemplate}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the AuditMapTemplate carrier, obligation surface, frontier non-closure theorem, and cross-map routing theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapTemplate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim proof of the frontier row, discharge of conditional rows, erasure of obstruction rows, automatic bridge between maps, or host semantic equality.

Upgrade path: Obligation closure requires Lean encodings of the template carrier, inventory projections, frontier non-closure, and cross-map routing preservation.

Chapter 809

A Concrete Naming Certificate for $\mathsf{TypeClassifierRouteJoin}^{\uparrow}$

The $\mathsf{TypeClassifierRouteJoin}^{\uparrow}$ packet realizes the type-checking-as-classifier-membership vision of chapter 9367 as a finite route-join object. It joins the membership row of chapter 476, the route-choice row of chapter 798, and the discharge-socket row of chapter 719 so a compiler or preservation consumer can cite one visible packet rather than coordinate three surfaces separately.

Definition 809.1 (TypeClassifierRouteJoin carrier)**.** *A* $\mathsf{TypeClassifierRouteJoin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *row, including its displayed term history, judgment ledger,* Ext *membership row, typed* Cont *reduction row, and local membership certificate;*
2. *R is the* $\mathsf{SubjectReductionRouteChoice}^{\uparrow}$ *row naming that the subject-reduction consumer reads the explicit bundle route or the obstruction boundary;*
3. *S is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row naming the four discharge obligations and the obstruction boundary for closed subject reduction;*
4. *H stores componentwise* hsame *transports for the membership row, route-choice row, discharge-socket row, provenance row, and local naming row;*
5. *C stores displayed* Cont *replay for reading classifier membership, route choice, and socket consumption through the same finite packet;*
6. *P is the* Pkg *provenance over chapter 9367, chapter 476, chapter 798, and chapter 719;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeClassifierRouteJoin}^{\uparrow}}$ *row.*

The carrier contains no host type semantics, hidden conversion relation, closed subject-reduction theorem, quotient term equality, or route selector outside M, R, S, H, C, P, N.

Lean4 (definition) `BEDC.Derived.TypeClassifierRouteJoinUp.TypeClassifierRouteJoinUp`

Theorem 809.2 (TypeClassifierRouteJoin NameCert obligations)**.** *For the carrier of Definition 809.1, the local* $\mathsf{NameCert}_{\mathsf{TypeClassifierRouteJoin}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the finite packet $J = (M, R, S, H, C, P, N)$;*

2. *joined classifier stability, saying that the classifier-membership row M is read through the same displayed transport and continuation routes as the joined route rows;*
3. *route-choice compatibility, saying that the route-choice row R selects only between the explicit bundle route and the obstruction boundary already displayed by its carrier;*
4. *discharge-socket compatibility, saying that socket reads through S consume exactly the four displayed discharge rows and the obstruction boundary recorded by the socket packet;*
5. *non-escape, saying that no host type semantics, hidden conversion, closed subject-reduction theorem, quotient term equality, or second route selector is exported by the joined packet.*

Proof. Carrier admission is the tuple displayed in Definition 809.1. The membership, route-choice, and socket rows are visible as M, R, S, and every transported read must pass through H.

Joined classifier stability is inherited at the paper level from the membership carrier of chapter 476. The join does not alter the membership row; it only records the route and socket rows beside it and replays the same reads through C.

Route-choice compatibility follows from chapter 798. The row R exposes the explicit bundle route and the obstruction boundary, and it does not create a second discharge route.

Discharge-socket compatibility follows from chapter 719. The row S names the four discharge obligations and the obstruction boundary, while P and N account for provenance and local naming. □

Lean4 ✓ `BEDC.Derived.TypeClassifierRouteJoinUp.TypeClassifierRouteJoinNameCertObligations`

Theorem 809.3 (TypeClassifierRouteJoin non-escape)**.** *No accepted* TypeClassifierRouteJoin$^{\uparrow}$ *carrier exports host type semantics, a hidden conversion relation, a closed subject-reduction theorem, quotient term equality, or a second route selector. Every public read factors through the displayed membership row M, route-choice row R, discharge-socket row S, componentwise transport row H, continuation replay row C, provenance row P, and local naming row N.*

Lean4 ✓ `BEDC.Derived.TypeClassifierRouteJoinUp.TypeClassifierRouteJoinUp nonescape`

Proof. Inspect the carrier inventory. The only classifier-facing coordinate is M, the only route-choice coordinate is R, and the only discharge-socket coordinate is S.

Transport and replay cannot add a semantic coordinate. The rows H and C preserve and expose the displayed membership, route-choice, and socket rows componentwise.

The provenance and naming rows identify the source chapters and the local certificate. Since none of M, R, S, H, C, P, N names host type semantics, hidden conversion, closed subject reduction, quotient term equality, or another route selector, those objects cannot be read from the joined packet. □

Closure status for NameCert$_{\text{TypeClassifierRouteJoin}^{\uparrow}}$.

Constructive story: A TypeClassifierRouteJoin$^{\uparrow}$ packet is generated from a classifier-membership row, a subject-reduction route-choice row, a discharge-socket row, componentwise hsame transports, displayed Cont replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{TypeClassifierRouteJoin}^{\uparrow}}$, seed)

Scope closed: Definition 809.1, Definition 809.2, and Definition 809.3 bind the finite route join over type-classifier membership, route choice, discharge socket, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{TypeClassifierRouteJoin}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host type semantics, hidden conversion, closed subject reduction, quotient term equality, route-selection uniqueness beyond the displayed route-choice row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, joined classifier stability, route-choice compatibility, discharge-socket compatibility, replay, provenance, local naming, and non-escape for the same finite route-join packet.

Chapter 810

A Concrete Naming Certificate for $\mathsf{AuditMapFamilyLedger}^{\uparrow}$

The $\mathsf{AuditMapFamilyLedger}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as finite NameCert data. It sits beside the substitution audit map of chapter 779, the MetaCIC audit synthesis packet of chapter 776, and the closure-preservation audit packet of chapter 802: those sibling chapters name local audit rows, while this packet records the family-level ledger that keeps their module tags, neighbour links, obstruction rows, and frontier rows visible without turning them into hidden closure.

Definition 810.1 (Audit-map family ledger carrier)**.** *An* $\mathsf{AuditMapFamilyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, A, E, P, O, U, B, H, C, L, K, N)$$

with the following visible rows:

1. *G is the family tag naming the local audit-map module and its BEDC substrate;*
2. *A is the local audit-map row, such as the substitution audit row of chapter 779;*
3. *E records neighbouring-map links to sibling audit packets, including synthesis, closure-preservation, subject-reduction, and normal-boundary audit rows;*
4. *P records positive rows already available to the local module;*
5. *O records conditional rows whose conclusion is available only after a named discharge route;*
6. *U records obstruction rows that the family ledger must keep visible;*
7. *B records frontier rows offered to downstream consumers without claiming their closure;*
8. *H records componentwise* hsame *transports for the family tag, local audit row, neighbour links, positive rows, conditional rows, obstruction rows, frontier rows, continuation rows, provenance rows, and local naming row;*
9. *C records the displayed* Cont *routes by which consumers read the local audit map, neighbouring-map links, row classes, frontier handoffs, provenance, and local name;*
10. *L is the finite ledger over the audit-map family rows,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
11. *K is the* Pkg *provenance row recording which concrete audit chapters supply the visible rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyLedger}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are the displayed family tag, local audit-map row, neighbouring-map links, positive rows, conditional rows, obstruction rows, frontier rows, transport rows, continuation routes, ledger row, package provenance, and local naming row. It exports no global proof of subject reduction, no full consistency theorem, no ambient confluence theorem, no hidden closure principle, and no host-level semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapFamilyLedgerUp.AuditMapFamilyLedgerTasteGate single carrier alignment`

Theorem 810.2 (Audit-map family ledger NameCert obligations)**.** *For the carrier of Definition 810.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (G, A, E, P, O, U, B, H, C, L, K, N),$$

family-tag readback through G, local audit-map readback through A, neighbouring-map link readback through E, positive-row readback through P, conditional-row readback through O, obstruction-row readback through U, frontier-row readback through B, componentwise transport through H, displayed continuation through C, ledger exactness through L, package provenance through K, and local naming through N.

Proof. Project the finite packet F. The rows G, A, E display the family tag, the local audit map, and the neighbouring audit maps before any row-level claim can be read.

Read the row classes next. The positive rows P, conditional rows O, obstruction rows U, and frontier rows B partition the visible audit surface into available, routed, blocked, and consumer-facing entries.

Finally inspect H, C, L, K, N. Transport and continuation replay the same displayed rows, the ledger records no additional row class, the package row cites only the visible concrete audit chapters, and the local name binds the packet. Hence the NameCert surface is exhausted by the carrier inventory. □

Theorem 810.3 (Audit-map family ledger hidden-closure non-escape)**.** *For every accepted* $\mathsf{AuditMapFamilyLedger}^{\uparrow}$ *carrier $F = (G, A, E, P, O, U, B, H, C, L, K, N)$, every consumer read of the audit-map family remains inside the displayed family tag, local audit-map row, neighbouring-map links, positive rows, conditional rows, obstruction rows, frontier rows,* hsame *transports,* Cont *routes, finite ledger,* Pkg *provenance, and local* NameCert *row. In particular, the packet cannot export a hidden proof of the obstruction rows, an unlisted discharge route for the conditional rows, a closed subject-reduction theorem, full closed consistency, ambient confluence, or host semantic equality.*

Proof. Start with the carrier projection. The only consumer-facing data are the rows $G, A, E, P, O, U, B, H, C, L, K$, so a read of the family ledger must enter through the displayed local audit map and its neighbouring links.

Use the NameCert obligations of Definition 810.2. They bind each positive, conditional, obstruction, and frontier row to its own displayed coordinate and route every transport or continuation through H and C.

The ledger and provenance rows then rule out hidden closure. Since L records no unlisted row and K cites only visible concrete audit chapters, a consumer may cite the family ledger as finite audit data but cannot extract a proof or discharge route absent from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFamilyLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapFamilyLedger}^{\uparrow}$ packet is generated from a family tag, a local audit-map row, neighbouring-map links, positive rows, conditional rows, obstruction rows, frontier rows, componentwise hsame transports, displayed Cont routes, a finite ledger, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{AuditMapFamilyLedger}\uparrow}$, seed)

Scope closed: Definition 810.1, Definition 810.2, and Definition 810.3 bind the finite audit-map family ledger and its hidden-closure boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{AuditMapFamilyLedger}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed consistency, ambient confluence, hidden discharge routes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-tag readback, local audit-map readback, neighbouring-map readback, row-class exactness, transport, continuation, ledger exactness, provenance, naming, and non-escape for the same finite packet.

Chapter 811

MetaCIC Audit Map Family Name Certificate Construction

The $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ packet turns the audit-map family methodology of chapter 9406 into one finite MetaCIC NameCert carrier. It sits beside the synthesis packet of chapter 776, the substitution audit map of chapter 779, and the closure-preservation audit packet of chapter 791: those chapters expose local rows, while this packet records how the rows are collected into a family without converting local audit data into global MetaCIC closure.

The packet is a composite over the local audit-map rows, synthesis rows, obstruction rows, frontier rows, and cross-map dependency ledger, rather than an atomic carrier.

Definition 811.1 (MetaCIC audit-map family carrier)**.** *A* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, S, O, F, D, H, C, P, N)$$

with the following visible rows:

1. *M is the finite family of local MetaCIC audit-map rows, including substitution, closure-preservation, normalization, confluence, beta, decidability, typed-example, subject-reduction, and consistency-facing maps when those rows are present;*
2. *S is the finite family of synthesis rows that read local map entries into the MetaCIC audit synthesis surface;*
3. *O is the obstruction ledger for local rows that must remain visible as blockers rather than hidden discharge evidence;*
4. *F is the frontier ledger for conditional or consumer-facing rows whose use is permitted only through named neighbouring maps;*
5. *D is the cross-map dependency ledger recording which synthesis, obstruction, and frontier rows consume which local audit-map rows;*
6. *H records componentwise* hsame *transports for the local map rows, synthesis rows, obstruction rows, frontier rows, dependency ledger, continuation routes, provenance, and local name;*
7. *C records the displayed* Cont *routes by which consumers replay local-map reads, synthesis reads, obstruction reads, frontier handoffs, dependency reads, provenance reads, and the local name;*
8. *P is the* Pkg *provenance over the audit-map methodology, the concrete audit-map siblings, the synthesis packet, the obstruction ledgers, the frontier ledgers, and the local MetaCIC substrates;*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICAuditMapFamily}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are the rows $M, S, O, F, D, H, C, P, N$. It exports no unlisted theorem of subject reduction, closure preservation, normalization, confluence, closed consistency, or host semantic equality.

Theorem 811.2 (MetaCIC audit-map family NameCert obligations)**.** *For the carrier of Definition 811.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditMapFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (M, S, O, F, D, H, C, P, N),$$

local audit-map row exposure through M, synthesis-row exposure through S, obstruction-row exposure through O, frontier-row exposure through F, cross-map dependency readback through D, componentwise route replay through H and C, package provenance through P, and local naming through N.

Proof. Project the finite packet A. The row M names the local audit maps before any synthesis or frontier read can be consumed.

Read the row families S, O, F next. The synthesis rows expose positive family assembly data, the obstruction rows expose blocked or conditional cells, and the frontier rows expose only the named handoffs available to downstream consumers.

Finally inspect D, H, C, P, N. The dependency ledger records which local rows each cross-map read may use, transport and continuation replay those reads componentwise, provenance cites the displayed sibling chapters, and the local name binds the packet. No other NameCert obligation is available from the carrier. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditMapFamilyUp.MetaCICAuditMapFamilyTasteGate single carrier alignment`

Theorem 811.3 (MetaCIC audit-map family non-escape)**.** *For every accepted* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *carrier $A = (M, S, O, F, D, H, C, P, N)$, every consumer read remains inside the displayed local audit-map rows, synthesis rows, obstruction rows, frontier rows, cross-map dependency ledger,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. In particular, a consumer cannot turn one local audit map, or the finite family of such maps, into global MetaCIC closure, subject reduction, normalization, confluence, full closed consistency, or host semantic equality.*

Proof. Start with the carrier projection. The only public coordinates are $M, S, O, F, D, H, C, P, N$, so every consumer-facing read must pass through the displayed local map rows and the displayed family ledgers.

Use the NameCert obligations of Definition 811.2. They force synthesis reads through S, obstruction reads through O, frontier reads through F, and cross-map reads through D with transport and continuation replay supplied by H and C.

The provenance and naming rows do not add theorem content. Since P cites only visible sibling packets and N names only the local carrier, no read can manufacture a missing global closure theorem or discharge an obstruction absent from the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICAuditMapFamilyUp.MetaCICAuditMapFamilyNonescape`

Theorem 811.4 (MetaCIC audit-map family dependency readback)**.** *For every accepted* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *carrier $A = (M, S, O, F, D, H, C, P, N)$, every cross-map family read exposes the local audit-map row family M, synthesis row family S, obstruction ledger O, frontier ledger F, dependency ledger*

D, componentwise transport H, continuation replay C, provenance row P, and local name N before any synthesis row is consumed. In particular, a synthesis consumer cannot read S while hiding the dependency ledger D or any of the displayed rows on which D depends.

Proof. Project the accepted carrier. The classifier accepts the finite packet only through the displayed coordinates $M, S, O, F, D, H, C, P, N$, so a cross-map read has no independent synthesis coordinate outside this list.

Read the dependency ledger before consuming synthesis output. The row D records which local map rows, obstruction rows, and frontier rows are used by each synthesis read, while H and C replay the same dependencies under componentwise transport and continuation.

The provenance and name rows keep the read attached to the same finite packet. Thus every synthesis-facing read exposes D together with M, S, O, F, H, C, P, N before the synthesis row is available to a consumer. □

Theorem 811.5 (MetaCIC audit-map family frontier ledger non-escape)**.** *For every accepted* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *carrier $A = (M, S, O, F, D, H, C, P, N)$, frontier rows in F remain visible ledger rows throughout every permitted family read. They cannot be converted into global MetaCIC closure, subject reduction, normalization, confluence, full closed consistency, or host semantic equality.*

Proof. Inspect the frontier coordinate F together with the dependency ledger D. A frontier row is consumed only through a named dependency read, and that read keeps the frontier coordinate visible beside the local audit-map and synthesis rows.

Apply the non-escape boundary of Definition 811.3. The accepted packet has no coordinate for global closure, subject reduction, normalization, confluence, full closed consistency, or host semantic equality.

Transport, continuation, provenance, and naming preserve the displayed ledger route; they do not add theorem content. Hence a frontier row can be replayed only as a frontier row of the same finite carrier, not as any of the excluded global MetaCIC conclusions. □

Theorem 811.6 (MetaCIC audit-map family primitive scope)**.** *For every accepted* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *carrier*

$$A = (M, S, O, F, D, H, C, P, N),$$

every family read is scoped by the displayed BEDC primitives and named sibling rows. Local audit maps are read through M, synthesis rows through S, obstruction rows through O, frontier rows through F, dependency rows through D, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No accepted family packet introduces a global closure row, subject-reduction row, normalization row, confluence row, closed-consistency row, or host equality row beyond those displayed primitives.

Proof. Project the accepted packet $A = (M, S, O, F, D, H, C, P, N)$. The carrier definition lists exactly the local audit-map, synthesis, obstruction, frontier, dependency, transport, replay, provenance, and naming rows.

Apply the NameCert obligations of Definition 811.2. They force each local audit-map read through M, each synthesis read through S, each obstruction read through O, each frontier read through F, and each dependency read through D before any family consumer receives the packet.

Apply non-escape and frontier-ledger non-escape from Definition 811.3 and Definition 811.5. These rows keep frontier and obstruction data visible as displayed ledgers rather than hidden theorem discharge.

Finally route replay, provenance, and naming through H, C, P, N. Since these rows only preserve and name the same finite packet, they introduce no global closure, subject reduction, normalization, confluence, closed consistency, or host equality row. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICAuditMapFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ packet is generated from local MetaCIC audit-map rows, synthesis rows, obstruction rows, frontier rows, a cross-map dependency ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICAuditMapFamily}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 811.1, Definition 811.2, Definition 811.3, Definition 811.4, Definition 811.5, and Definition 811.6 bind the finite MetaCIC audit-map family packet, dependency readback, frontier-ledger visibility, primitive dependency scope, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICAuditMapFamily}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.MetaCICAuditMapFamilyUp.MetaCICAuditMapFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global MetaCIC closure, subject reduction, normalization, confluence, full closed consistency, or bridge checking.

Upgrade path: Further closure requires public export rows or a bridge row over local audit-map reads, synthesis reads, obstruction reads, frontier reads, route replay, provenance, and naming over the same finite packet.

Chapter 812

A Concrete Naming Certificate for $\mathsf{AuditMapFrontierIndex}^{\uparrow}$

The $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as finite frontier-index routing data. It sits beside the family ledger of chapter 810, the substitution audit map of chapter 779, and the synthesis packet of chapter 776: those chapters expose local rows and family ledgers, while this packet records how frontier entries are routed between local maps, neighbouring-map rows, obstruction rows, and synthesis consumers without closing the frontier itself.

Definition 812.1 (Audit-map frontier index carrier)**.** *An* $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (T, A, E, P, R, O, F, S, H, C, K, N)$$

with the following displayed rows:

1. *T is the map tag naming the local audit-map module and its BEDC substrate;*
2. *A is the local audit row whose frontier is being indexed;*
3. *E is the neighbouring-map row listing the sibling audit packets that may supply or consume frontier entries;*
4. *P is the positive row recording entries already available in the local map;*
5. *R is the conditional route row naming the displayed discharge route required before a conditional entry may be consumed;*
6. *O is the obstruction row whose named boundary must remain visible to every route;*
7. *F is the frontier row offered to downstream consumers as a named boundary and not as a closed theorem;*
8. *S is the synthesis-consumer row naming the finite reader that consumes the indexed frontier together with neighbouring-map links;*
9. *H records componentwise* hsame *transports for the map tag, local audit row, neighbouring-map row, positive row, conditional route row, obstruction row, frontier row, synthesis-consumer row, provenance row, and local naming row;*

10. *C records the displayed* Cont *routes from the local map to neighbouring maps, from conditional rows to their named routes, from obstruction rows to frontier rows, and from frontier rows to synthesis consumers;*
11. *K is the* Pkg *provenance row over the audit-map family ledger, local audit maps, neighbouring-map links, frontier rows, synthesis consumers,* BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}$ *row.*

The carrier is the finite frontier index. It contains no coordinate for a hidden discharge of the obstruction row, a closed proof of a frontier entry, an unlisted neighbouring map, a global subject-reduction theorem, full closed consistency, ambient confluence, or host semantic equality.

Theorem 812.2 (Audit-map frontier index NameCert obligations)**.** *For the carrier of Definition 812.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for $I = (T, A, E, P, R, O, F, S, H, C, K, N)$ as finite* BHist *data;*
2. *family-map readback through the map tag T, local audit row A, and neighbouring-map row E;*
3. *frontier-row exactness saying that F is read only as the displayed frontier row associated to A, E, P, R, O;*
4. *obstruction non-escape saying that no consumer route can bypass O or read F as a discharge of O;*
5. *synthesis-consumer handoff saying that S receives only the displayed frontier row, neighbouring-map links, transports, continuations, provenance, and local name through H, C, K, N.*

Lean4 ✓ `BEDC.Derived.AuditMapFrontierIndexUp.AuditMapFrontierIndexTasteGate single carrier alignment`

Proof. Project the carrier I. The rows T, A, E identify the local audit map and its neighbouring-map surface, while P, R, O, F separate positive entries, conditional routes, obstruction entries, and frontier entries.

The frontier exactness obligation is then finite row inspection. The only frontier coordinate in the packet is F, and the displayed continuation row C routes it together with the local row, neighbouring-map row, positive row, conditional route row, and obstruction row.

For non-escape, inspect the obstruction and consumer rows. Since O is a separate carrier coordinate and F is only a frontier coordinate, a consumer cannot turn the frontier row into a proof of the obstruction or bypass the named conditional route R.

Finally read S, H, C, K, N. These rows hand the indexed frontier to synthesis consumers, preserve the same component rows under hsame, replay the displayed Cont routes, record package provenance, and bind the local name. No remaining coordinate can supply an unlisted neighbouring map, hidden discharge, global subject-reduction theorem, full closed consistency, ambient confluence, or host semantic equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ packet is generated from a map tag, local audit row, neighbouring-map row, positive row, conditional route row, obstruction row, frontier row, synthesis-consumer row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 812.1 and Definition 812.2 bind the finite audit-map frontier index and its synthesis-consumer handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.AuditMapFrontierIndexUp.AuditMapFrontierIndexTasteGate\_single\_carrier\_alignment`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close obstruction rows, conditional discharge routes, frontier entries, subject reduction, full closed consistency, ambient confluence, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, family-map readback, frontier-row exactness, obstruction non-escape, synthesis-consumer handoff, transport, continuation, provenance, and naming for the same finite packet.

Definition 812.3 (Displayed frontier-index restriction)**.** *Let*

$$I = (T, A, E, P, R, O, F, S, H, C, K, N)$$

be an accepted $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ *carrier in the sense of Definition 812.1 and Definition 812.2. A displayed subfamily restriction of* I *is a finite row* E_0 *together with an inclusion certificate* $E_0 \subseteq E$ *and the coordinatewise retained packet*

$$I|_{E_0} = (T, A, E_0, P|_{E_0}, R|_{E_0}, O|_{E_0}, F|_{E_0}, S|_{E_0}, H|_{E_0}, C|_{E_0}, K|_{E_0}, N|_{E_0}),$$

where each restricted row is obtained by deleting exactly the entries whose neighbouring-map coordinate is not displayed in E_0*, while keeping the local audit row* A*, the map tag* T*, the local positive row over* A*, and every retained transport, continuation, provenance, and local naming certificate attached to an entry of* E_0*.*

Theorem 812.4 (Audit-map frontier-index neighbouring restriction)**.** *Let*

$$I = (T, A, E, P, R, O, F, S, H, C, K, N)$$

be an accepted $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ *carrier. Let* $E_0 \subseteq E$ *be a displayed finite neighbouring-map subfamily, and suppose the restricted rows*

$$P|_{E_0},\ R|_{E_0},\ O|_{E_0},\ F|_{E_0},\ S|_{E_0},\ H|_{E_0},\ C|_{E_0},\ K|_{E_0},\ N|_{E_0}$$

retain exactly the positive rows, conditional route rows, obstruction rows, frontier rows, synthesis-consumer rows, transports, continuations, provenance, and local $\mathsf{NameCert}$ *rows whose neighbouring-map coordinate lies in* E_0*, while retaining the same local audit row* A *and map tag* T*. Then the displayed restriction*

$$I|_{E_0} = (T, A, E_0, P|_{E_0}, R|_{E_0}, O|_{E_0}, F|_{E_0}, S|_{E_0}, H|_{E_0}, C|_{E_0}, K|_{E_0}, N|_{E_0})$$

is an accepted $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ *carrier for the restricted neighbouring-map row. Moreover, if a neighbouring map* e *is not displayed in* E_0*, then* e *supplies no frontier entry of* $I|_{E_0}$ *and no synthesis consumer of* $I|_{E_0}$ *can read* e *as a frontier source.*

Proof. By Definition 812.1, an $\mathsf{AuditMapFrontierIndex}^{\uparrow}$ carrier is exhausted by the twelve displayed rows $T, A, E, P, R, O, F, S, H, C, K, N$. The restriction keeps the map tag T and local audit row A, replaces only the neighbouring-map row E by the displayed finite row E_0, and restricts every row whose entries are indexed by neighbouring maps to those coordinates lying in E_0. Since E_0 is finite and every retained coordinate is a coordinate already present in the finite packet I, the displayed tuple $I|_{E_0}$ is again finite BHist data.

It remains to check that the five local $\mathsf{NameCert}_{\mathsf{AuditMapFrontierIndex}^{\uparrow}}$ obligations of Definition 812.2 survive this deletion. Carrier admission has just been checked by finite subrow formation. Family-map readback is local in T, A, E: after restriction the readback row is T, A, E_0, and each neighbour

named there is named because $E_0 \subseteq E$. Frontier-row exactness is also coordinatewise: $F|_{E_0}$ is read only as the displayed frontier row associated to $A, E_0, P|_{E_0}, R|_{E_0}, O|_{E_0}$, because no other frontier coordinate is inserted by the restriction.

For obstruction non-escape, inspect the retained obstruction and continuation rows. The row $O|_{E_0}$ is the only obstruction row of the restricted packet, and $C|_{E_0}$ contains only the displayed routes from the local map to neighbours in E_0, from retained conditional rows to their named routes, from retained obstruction rows to retained frontier rows, and from retained frontier rows to retained synthesis consumers. Thus a consumer of $I|_{E_0}$ cannot bypass $O|_{E_0}$, cannot discharge $O|_{E_0}$ through a deleted neighbour, and cannot read $F|_{E_0}$ as a proof of the obstruction.

For synthesis-consumer handoff, the consumer row is exactly $S|_{E_0}$, and its available data are exactly the retained frontier row, retained neighbouring-map links, retained transports $H|_{E_0}$, retained continuations $C|_{E_0}$, retained provenance $K|_{E_0}$, and retained local name $N|_{E_0}$. Hence the restricted packet satisfies the same five obligations with E_0 in place of E, so it is an accepted frontier index.

Finally let e be a neighbouring map not displayed in E_0. A frontier entry of $I|_{E_0}$ must occur in the sole frontier coordinate $F|_{E_0}$, and every such coordinate was retained only when its neighbouring-map coordinate lay in E_0. Likewise every synthesis read in $I|_{E_0}$ factors through $S|_{E_0}$ and $C|_{E_0}$, whose entries were retained only over E_0. Therefore e has no coordinate in the restricted frontier row and no displayed continuation into a restricted synthesis consumer, so it cannot supply a frontier entry of $I|_{E_0}$. $\square$

Chapter 813

A Concrete Naming Certificate for $\mathsf{AuditMapMethodology}^{\uparrow}$

The $\mathsf{AuditMapMethodology}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite NameCert carrier. It sits between the family-level method and the substitution audit map of chapter 779: the methodology packet names the template, admissible row classes, boundary rule, and sibling handoff through which concrete audit maps become readable without claiming their neighbouring proof obligations.

Definition 813.1 (Audit-map methodology carrier)**.** *An* $\mathsf{AuditMapMethodology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, C, O, F, B, S, H, R, P, N)$$

with the following visible rows:

1. *T is the audit-map template row, recording the setup, established, conditional, obstruction, frontier, cross-reference, and boundary slots of Definition 9406.2;*
2. *A is the admitted audit-map row family, whose entries include concrete local maps such as* $\mathsf{SubstitutionAuditMap}^{\uparrow}$*;*
3. *C is the classifier row separating established, conditional, obstruction, frontier, and handoff content;*
4. *O is the obligation-surface row listing the finite NameCert checks needed before a local audit map can be consumed;*
5. *F is the frontier row that records named open sockets without turning them into theorem rows;*
6. *B is the boundary row excluding hidden closure, ambient equality, and unstated discharge routes;*
7. *S is the sibling handoff row pointing from the methodology packet to the concrete substitution audit map and to later audit-map family ledgers;*
8. *H records componentwise* hsame *transports for the template, audit-map family, classifier, obligation, frontier, boundary, handoff, route, provenance, and naming rows;*
9. *R records the displayed* Cont *routes through which consumers inspect the methodology packet;*
10. *P is the* Pkg *provenance over the audit-map family method,* $\mathsf{SubstitutionAuditMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapMethodology}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose visible rows are the displayed template, admitted audit-map family, classifier, obligation surface, frontier, boundary, sibling handoff, transport, continuation, provenance, and naming rows. It exports no proof of subject reduction, no confluence theorem, no normalisation theorem, no closed-consistency theorem, no unlisted audit map, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapMethodologyUp.AuditMapMethodologyTasteGate single carrier alignment`

Theorem 813.2 (Audit-map methodology obligation surface)**.** *For the carrier of Definition 813.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapMethodology}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (T, A, C, O, F, B, S, H, R, P, N),$$

template readback through T, admitted-map readback through A, classifier readback through C, obligation-surface readback through O, frontier readback through F, boundary readback through B, sibling handoff through S, componentwise transport through H, displayed continuation through R, package provenance through P, and local naming through N.

Proof. Project the finite packet M. The row T displays the audit-map template before any local audit content is read, while A supplies only the admitted family of concrete map rows.

Read C, O, F, B next. The classifier separates the row classes, the obligation surface lists the NameCert checks, the frontier keeps open sockets visible, and the boundary records the exclusions attached to the methodology.

Finally inspect S, H, R, P, N. The handoff row names the concrete sibling surface, the transport and continuation rows replay the same displayed data, the provenance row cites only visible sources, and the local naming row binds the packet. No further obligation remains outside the listed coordinates. □

Theorem 813.3 (Audit-map methodology template soundness)**.** *Every accepted* $\mathsf{AuditMapMethodology}^{\uparrow}$ *carrier $M = (T, A, C, O, F, B, S, H, R, P, N)$ presents the template of Definition 9406.2 as finite NameCert data: positive content is read through the classifier, conditional content is routed through the obligation surface, obstruction and frontier content remain named rows, and cross-module use is available only through displayed handoff and continuation rows.*

Proof. Use the carrier definition to read T as the finite template row. Its slots are setup, established, conditional, obstruction, frontier, cross-reference, and boundary, matching the method described in chapter 9406.

The classifier row C and obligation row O then sort the template entries. Established entries may be cited as displayed rows, conditional entries retain their named routes, and obstruction or frontier entries remain visible without being promoted to solved theorem content.

The rows S and R account for cross-module use. A consumer can follow only the displayed sibling handoff and continuation routes, so the template is sound as NameCert data rather than as an ambient claim about neighbouring modules. □

Theorem 813.4 (Audit-map methodology boundary non-escape)**.** *For every accepted* $\mathsf{AuditMapMethodology}^{\uparrow}$ *carrier $M = (T, A, C, O, F, B, S, H, R, P, N)$, every consumer read remains inside the displayed template, admitted audit-map family, classifier, obligation surface, frontier, boundary, sibling handoff,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *rows. In particular, the packet cannot export a hidden proof obligation discharge, an unlisted audit map, subject reduction, confluence, normalisation, closed consistency, or host semantic equality.*

Proof. Begin with carrier projection. The only consumer-facing data are the rows $T, A, C, O, F, B, S, H, R, P, N$, so every read of the methodology packet enters through the displayed audit-map family surface.

Apply the obligation surface of Definition 813.2. It binds each row class, frontier socket, boundary exclusion, and sibling handoff to a visible coordinate and routes transport or continuation through H and R.

The provenance and naming rows rule out escape. Since P records only the methodology source and named concrete audit-map rows, and N names only this finite carrier, no consumer can extract a theorem or discharge route absent from the packet. □

Theorem 813.5 (Audit-map methodology substitution handoff)**.** *The* $\mathsf{AuditMapMethodology}^{\uparrow}$ *carrier admits* $\mathsf{SubstitutionAuditMap}^{\uparrow}$ *as a concrete sibling handoff exactly at the row S: the methodology packet may point to the substitution audit-map carrier of chapter 779, read its closed-substitution audit rows as an admitted local map through A, and keep all substitution-specific non-escape exclusions inside that sibling chapter.*

Proof. Read S as the sibling handoff row and A as the admitted audit-map family row. The handoff therefore points to $\mathsf{SubstitutionAuditMap}^{\uparrow}$ only as a concrete local audit map, not as a replacement for the methodology packet.

The classifier C and obligation row O preserve the distinction between the methodology slots and the substitution-specific rows. Closed shift, substitution, and closed-composition data are read in the sibling chapter, while this carrier records the fact that such a map is admitted by the method.

Finally use B, H, R, P, N. The boundary row keeps unrestricted substitution, beta-substitution preservation, and closed subject reduction outside the methodology handoff; transport, continuation, provenance, and naming replay only the displayed finite rows. Thus the handoff is available without absorbing the sibling's exclusions into a stronger methodology claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapMethodology}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AuditMapMethodology}^{\uparrow}$ packet is generated from an audit-map template row, admitted audit-map family rows, classifier rows, obligation-surface rows, frontier and boundary rows, a sibling handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapMethodology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 813.1, Definition 813.2, Definition 813.3, Definition 813.4, and Definition 813.5 bind the finite audit-map methodology carrier, template soundness, boundary, and substitution handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapMethodology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, confluence, normalisation, closed consistency, hidden discharge routes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, template readback, admitted-map readback, classifier and obligation-surface exactness, boundary non-escape, substitution handoff, transport, continuation, provenance, and naming for the same finite packet.

Chapter 814

A Concrete Naming Certificate for AuditMapCoverageLedger$^{\uparrow}$

The AuditMapCoverageLedger$^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite coverage ledger. It sits beside the kernel acceptance audit witness of chapter 547, the substitution audit map of chapter 779, the MetaCIC confluence audit of chapter 743, and the closure-preservation audit packet of chapter 803. Those sibling chapters expose concrete audit rows; this packet records which rows are established, conditional, obstructed, frontier-facing, and cross-map dependent before any consumer reads the family as a covered surface.

Definition 814.1 (Audit-map coverage ledger carrier)**.** *An* AuditMapCoverageLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, E, C, O, F, X, H, R, P, N)$$

with the following displayed rows:

1. *M is the map-source row naming the audit-map chapter, Lean substrate module, and concrete sibling packet from which the coverage entry is read;*
2. *E is the established-inventory row for theorem or definition entries already exposed by the map source;*
3. *C is the conditional-inventory row for entries whose conclusion is available only through a displayed discharge route or local hypothesis row;*
4. *O is the obstruction-inventory row for blocked entries that the map must keep visible as blocked rather than silently discharged;*
5. *F is the frontier-inventory row for entries offered to downstream consumers as named boundaries without asserting family-level closure;*
6. *X is the cross-map dependency row recording exact links to kernel acceptance, substitution, confluence, closure-preservation, and other displayed audit packets;*
7. *H records componentwise* hsame *transports for the source, inventory, dependency, route, provenance, and local naming rows;*
8. *R records the displayed* Cont *routes by which a consumer may read established, conditional, obstruction, frontier, and cross-map dependency rows;*
9. *P is the* Pkg *provenance row over the concrete audit-map family,* BHist*,* hsame*,* Cont*, and* NameCert *data;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are $M, E, C, O, F, X, H, R, P, N$. It contains no coordinate for family-level closure, hidden discharge, stronger MetaCIC result, host semantic equality, or unlisted proof source.

Theorem 814.2 (Audit-map coverage ledger NameCert obligations)**.** *For the carrier of Definition 814.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the finite packet $L = (M, E, C, O, F, X, H, R, P, N)$;*
2. *role-row coverage for the map source, established inventory, conditional inventory, obstruction inventory, frontier inventory, cross-map dependency, transport, route, provenance, and local naming rows;*
3. *established, conditional, obstruction, and frontier separation, so no row class may be read as another row class;*
4. *cross-map dependency exactness, so every dependency read through X names a displayed sibling packet and route rather than an ambient family theorem;*
5. *local naming, so all consumer-facing reads pass through the displayed* $\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.AuditMapCoverageLedgerUp.AuditMapCoverageLedgerTasteGate single carrier alignment`

Proof. Project the finite carrier. The row M names the map source, while E, C, O, F partition the visible audit inventory into established, conditional, obstructed, and frontier-facing entries.

Read the cross-map row next. The dependency row X may cite only displayed sibling packets and the route row R records how those citations are consumed.

Finally inspect H, P, N. Componentwise transport preserves the same rows, provenance cites the concrete audit-map family, and the local NameCert row binds the consumer-facing packet. Hence the five obligations exhaust the naming surface. □

Theorem 814.3 (Audit-map coverage ledger frontier non-escape)**.** *For every accepted* $\mathsf{AuditMapCoverageLedger}^{\uparrow}$ *carrier $L = (M, E, C, O, F, X, H, R, P, N)$, no consumer may read family-level closure, hidden discharge, or a stronger MetaCIC result from the ledger unless the read is one of the displayed role rows $M, E, C, O, F, X, H, R, P, N$. In particular, an established row remains established, a conditional row remains routed through its displayed condition, an obstruction row remains blocked, and a frontier row remains a named boundary.*

Proof. Start with the carrier projection. The only rows available to a consumer are the map source, four inventory rows, cross-map dependency row, transport row, continuation route row, provenance row, and local naming row.

Use the NameCert obligations of Definition 814.2. They separate the established, conditional, obstruction, and frontier inventories and force every cross-map read through the displayed dependency and route rows.

Since the classifier has no coordinate for family-level closure, hidden discharge, or a stronger MetaCIC theorem, those readings cannot be extracted from the ledger. A consumer may cite the frontier only as the displayed finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapCoverageLedger}^{\uparrow}$ packet is generated from a map-source row, established inventory, conditional inventory, obstruction inventory, frontier inventory, exact cross-map dependency row, componentwise transport, displayed continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 814.1, Definition 814.2, and Definition 814.3 bind the finite coverage ledger and its frontier boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapCoverageLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close the audit-map family, discharge conditional rows, prove obstruction rows, strengthen MetaCIC results, provide Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-class coverage, row-class separation, cross-map dependency exactness, local naming, and frontier non-escape for the same finite packet.

Chapter 815

A Concrete Naming Certificate for $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$

The $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite obligation-ledger NameCert packet. It sits beside the closure-preservation audit seal of chapter 800, the MetaCIC substitution audit packet of **??**, and the family ledger of chapter 810: those chapters expose concrete audit rows, while this packet records the obligation classes that downstream consumers may inspect across the family.

Definition 815.1 (Audit-map family obligation ledger carrier)**.** *An* $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, P, Q, O, R, X, H, C, K, N)$$

with the following visible rows:

1. *M is the audit-map family tag naming the concrete module, its local audit row, and its sibling module coordinates;*
2. *P is the positive inventory of rows already closed by the named concrete packet;*
3. *Q is the conditional inventory of rows whose conclusion is available only after a displayed discharge route;*
4. *O is the obstruction inventory of rows that remain visible as blocked obligations;*
5. *R is the frontier inventory offered to downstream consumers without claiming its closure;*
6. *X is the cross-module use inventory recording which sibling packets may read the positive, conditional, obstruction, and frontier rows;*
7. *H records componentwise* hsame *transports for the family tag, all four inventories, cross-module use rows, continuation rows, package provenance, and the local naming row;*
8. *C records the displayed* Cont *routes by which consumers read each inventory class, the cross-module use row, provenance, and local name;*
9. *K is the* Pkg *provenance over chapter 800,* **??**, *chapter 810,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are the family tag, positive inventory, conditional inventory, obstruction inventory, frontier inventory, cross-module use row, hsame *transports,* Cont *routes,* Pkg *provenance, and local NameCert. It exports no hidden discharge for Q, no proof of the obstruction rows O, no closure of the frontier rows R, and no global audit theorem outside the displayed packet.*

Theorem 815.2 (Audit-map family obligation ledger NameCert obligations)**.** *For the carrier of Definition 815.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (M, P, Q, O, R, X, H, C, K, N),$$

family-tag readback through M, positive-inventory readback through P, conditional-inventory readback through Q, obstruction-inventory readback through O, frontier-inventory readback through R, cross-module use readback through X, componentwise transport through H, displayed continuation through C, package provenance through K, and local naming through N.

Lean4 ✓ `BEDC.Derived.AuditMapFamilyObligationLedgerUp.AuditMapFamilyObligationLedgerTasteGate`
`single carrier alignment`

Proof. Project the finite packet F. The row M names the audit-map family coordinate before any obligation class is read.

Read the four obligation inventories next. The positive row P, conditional row Q, obstruction row O, and frontier row R are separate coordinates, so admission of one class does not discharge or close another.

Finally inspect X, H, C, K, N. Cross-module use is displayed as a row rather than inferred from prose, transport and continuation replay the same inventory coordinates, provenance cites only the visible sibling packets, and the local name binds the packet. Hence the NameCert surface is exhausted by the displayed carrier inventory. □

Theorem 815.3 (Audit-map family obligation ledger classifier stability)**.** *The* $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ *classifier is stable under componentwise* hsame *transport of the family tag, positive inventory, conditional inventory, obstruction inventory, frontier inventory, cross-module use row, continuation routes, package provenance, and local NameCert row. Any transported accepted packet has the same displayed obligation classes and the same non-escape boundary.*

Proof. Start with two accepted packets related by the componentwise transport row H. The family tag and each inventory coordinate are transported separately, so the positive, conditional, obstruction, and frontier classes cannot be merged by transport.

The cross-module row X is transported along the same componentwise route. Thus a sibling consumer that was allowed to read an inventory class before transport reads the transported instance of that same class after transport.

The continuation, provenance, and naming rows then replay the transported coordinates through C, K, N. Since no classifier coordinate is added or deleted, acceptance and the non-escape boundary are preserved. □

Theorem 815.4 (Audit-map family obligation ledger exactness)**.** *Every consumer read of an accepted* $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ *packet factors through exactly one displayed inventory class or the displayed cross-module use row before reaching transport, continuation, provenance, and local naming. In particular, a read of P cannot be reclassified as a discharged Q row, a proof of an O row, or a closure of an R row.*

Proof. Use the carrier projection and the NameCert obligations. They list the only readable coordinates as $M, P, Q, O, R, X, H, C, K, N$ and assign each obligation class to its own row.

If a consumer reads a row of the family ledger, the route first reaches P, Q, O, R, or X. The subsequent hsame, Cont, Pkg, and NameCert rows transport and report that coordinate, but they do not introduce a new obligation class.

Therefore the ledger is exact at the inventory boundary. Positive rows remain positive rows, conditional rows remain conditional rows, obstruction rows remain obstruction rows, frontier rows remain frontier rows, and cross-module use remains an explicit use row rather than a hidden closure principle. □

Theorem 815.5 (Audit-map family obligation ledger methodology handoff)**.** *The audit-map family methodology of chapter 9406 is realized by the finite* $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ *packet: positive inventory, conditional inventory, obstruction inventory, frontier inventory, cross-module use,* hsame *transports,* Cont *routes,* Pkg *provenance, and local NameCert are all carried as displayed BEDC rows. Downstream audit consumers may cite the packet only through those rows and the sibling anchors chapter 800 and* **??***.*

Proof. First apply the carrier definition. It turns the methodology into the finite packet $F = (M, P, Q, O, R, X, H, C, K, N)$ with the four obligation inventories and the cross-module use row visible.

Next apply classifier stability and ledger exactness. These keep the rows stable under hsame transport and force each consumer read to pass through the displayed inventory or use coordinate before any continuation or provenance read.

Finally use the package provenance row K. It names the closure-preservation audit seal and MetaCIC substitution audit packet as sibling anchors, so the methodology handoff is a finite NameCert packet rather than an ambient global audit claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}$ packet is generated from an audit-map family tag, positive inventory, conditional inventory, obstruction inventory, frontier inventory, cross-module use rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 815.1, Definition 815.2, Definition 815.3, Definition 815.4, and Definition 815.5 bind the finite obligation-ledger packet for audit-map family methodology.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapFamilyObligationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global audit correctness, subject reduction, closure preservation, hidden discharge routes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inventory readback, cross-module use readback, classifier stability, ledger exactness, provenance, naming, and non-escape for the same finite packet.

Chapter 816

A Concrete Naming Certificate for $\mathsf{AuditMapFamilyPacket}^{\uparrow}$

The $\mathsf{AuditMapFamilyPacket}^{\uparrow}$ carrier realizes the audit-map family methodology of chapter 9406 as concrete finite NameCert data. It sits beside the subject-reduction certificate of chapter 483, the MetaCIC audit synthesis packet of chapter 776, and the substitution audit map of chapter 779. Those sibling chapters supply local audit surfaces; this packet records the family-level rows that let a consumer read positive content, conditional content, obstructions, frontier claims, cross-module use, and boundary data without treating partial audit maps as closed MetaCIC.

Definition 816.1 (Audit-map family packet carrier)**.** *An* $\mathsf{AuditMapFamilyPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, C, O, F, X, H, R, Q, N)$$

with the following displayed rows:

1. *P is the finite family of positive-content rows already licensed by local audit maps, including checked inventory and concrete packet rows supplied by sibling chapters;*
2. *C is the family of conditional rows, each paired with the displayed discharge route or setup field needed before its conclusion can be consumed;*
3. *O is the obstruction family naming blocked routes, missing inhabitants, shape restrictions, and no-go rows that must remain visible;*
4. *F is the frontier family naming the strongest nearby claims offered to downstream readers without proof by the packet;*
5. *X is the cross-module-use family naming which audit maps, synthesis chapters, and boundary packets may consume or supply each displayed row;*
6. *H records componentwise* hsame *transports for positive, conditional, obstruction, frontier, cross-module-use, continuation, provenance, and local naming rows;*
7. *R records the displayed* Cont *read routes by which a consumer inspects the audit rows without adding hidden coordinates;*
8. *Q is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the audit-map methodology, and the concrete sibling packets cited above;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyPacket}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are $P, C, O, F, X, H, R, Q, N$. It exports no closed subject-reduction theorem, no full closed-MetaCIC consistency theorem, no erased obstruction, no proof of a frontier row, and no host semantic equality coordinate.

Theorem 816.2 (Audit-map family packet NameCert obligations)**.** *For the carrier of Definition 816.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamilyPacket}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the finite packet $A = (P, C, O, F, X, H, R, Q, N)$;*
2. *classifier transport for the positive, conditional, obstruction, frontier, and cross-module-use rows through the componentwise* hsame *transports in H and the displayed* Cont *routes in R;*
3. *obstruction and frontier exactness, requiring O and F to remain visible as boundary rows rather than as discharged theorem content;*
4. *cross-map dependency exposure, requiring every use by chapter 483, chapter 776, chapter 779, or any sibling audit packet to be recorded in X and sourced by Q;*
5. *non-escape from hidden closure, excluding closed MetaCIC, host semantics, unlisted discharge routes, and implicit proof of any obstruction or frontier row.*

Proof. Project the carrier A. Carrier admission supplies only the displayed rows $P, C, O, F, X, H, R, Q, N$, so every audit-map family read begins with the finite packet itself.

Transport is componentwise. The row H moves the positive, conditional, obstruction, frontier, and cross-module-use rows together, while R replays only the displayed continuation routes. Hence classifier movement cannot add a new row class.

The obstruction and frontier coordinates are then fixed. Rows O and F are part of the visible packet, but neither is a discharge witness or theorem constructor. They name where the audit family stops.

Finally inspect X, Q, N. Cross-module use is listed before it is consumed, provenance cites only displayed sibling packets, and the local name binds the same carrier. Therefore hidden closure, unlisted routes, host semantics, and implicit frontier proofs are outside the NameCert surface. □

Theorem 816.3 (Audit-map family packet frontier exhaustion)**.** *For every accepted* $\mathsf{AuditMapFamilyPacket}^{\uparrow}$ *carrier $A = (P, C, O, F, X, H, R, Q, N)$, every consumer read of the audit-map family factors through the displayed rows $P, C, O, F, X, H, R, Q, N$. In particular, a downstream consumer cannot treat a partial audit map as closed MetaCIC, cannot bypass the conditional rows in C, cannot erase the obstruction rows in O, and cannot turn a frontier row in F into a closed theorem.*

Proof. Start from the carrier projection. The only family rows are positive content P, conditional content C, obstructions O, frontier rows F, and cross-module-use rows X.

Use the obligation theorem. Classifier transport and continuation replay preserve those same rows through H and R, while Q and N record provenance and local naming without adding another consumer route.

It remains to exclude hidden closure. Since every consumer read has already been routed through the displayed audit rows, no downstream chapter can read a closed MetaCIC theorem, a discharged conditional row, an erased obstruction, or a proof of a frontier row unless that object appears as a displayed coordinate of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFamilyPacket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AuditMapFamilyPacket}^{\uparrow}$ packet is generated from positive-content rows, conditional rows, obstruction rows, frontier rows, cross-module-use rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{AuditMapFamilyPacket}^{\uparrow}}$, seed)

Scope closed: Definition 816.1, Definition 816.2, and Definition 816.3 bind the finite audit-map family carrier, row classifier, dependency exposure, and hidden-closure boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{AuditMapFamilyPacket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close subject reduction, full closed MetaCIC, host semantics, proof of frontier rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, obstruction and frontier exactness, cross-map dependency exposure, provenance, local naming, and non-escape for the same finite packet.

Chapter 817

A Family-Level Audit Surface for $\mathsf{AuditMapFamily}^{\uparrow}$

The $\mathsf{AuditMapFamily}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite NameCert object. It sits beside the beta audit packet of chapter 761, the closure-preservation audit packet of chapter 802, and the substitution audit map of chapter 779: those chapters expose individual audit rows, while this packet records the family-level carrier that factors inventory, obstruction, frontier, and cross-map routing without turning any local row into hidden closure.

Definition 817.1 (Audit-map family NameCert carrier)**.** *An* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, I, O, R, B, H, C, K, N)$$

with the following visible rows:

1. *T is the family tag naming the MetaCIC audit-map family and its BEDC substrate;*
2. *I is the finite inventory table of local audit maps, including beta, substitution, closure-preservation, confluence, normalisation, decidable-checking, typed-example, subject-reduction, and consistency-assembly rows when those rows are present;*
3. *O is the obstruction table recording rows whose absence or conditional status must remain visible;*
4. *R is the cross-map routing table recording which local maps may be read by neighbouring maps;*
5. *B is the frontier table recording consumer-facing claims that are named but not closed by the family packet;*
6. *H is the componentwise* hsame *transport row for T, I, O, R, B, C, K, N;*
7. *C is the displayed* Cont *route by which a consumer reads the inventory, obstruction, routing, frontier, provenance, and local name rows;*
8. *K is the* Pkg *provenance row over the concrete audit chapters supplying the local map entries;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFamily}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose coordinates are the displayed family tag, finite inventory, obstruction table, routing table, frontier table, transport row, continuation route, package provenance, and local naming row. It exports no proof of subject reduction, confluence, normalisation, closed consistency, host semantic equality, or discharge route not already displayed by one of those coordinates.

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyTasteGate single carrier alignment`

Theorem 817.2 (Audit-map family inventory factorization)**.** *For the carrier of Definition 817.1, every accepted consumer read factors through exactly one of the displayed family coordinates*

$$T,\ I,\ O,\ R,\ B,\ H,\ C,\ K,\ N.$$

In particular, a read of a local audit fact must pass through the finite inventory row I, a read of a blocked or conditional fact must pass through the obstruction row O, a read of a neighbouring map must pass through the routing row R, and a read of a consumer-facing open boundary must pass through the frontier row B.

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyInventoryFactorization`

Proof. Project the finite carrier F. The classifier admits only the nine displayed rows, so every consumer read must choose one coordinate before it can inspect any audit datum.

The rows I, O, R, B then partition the family-facing information. Local established material is inventory, blocked or conditional material is obstruction, cross-map access is routing, and consumer-facing open material is frontier.

The remaining rows cannot add a new class of audit fact. The transport row H preserves the displayed coordinates, the continuation row C routes reads through them, the provenance row K names their source chapters, and the local naming row N binds the packet. Hence the carrier inventory factors every read through the stated coordinates. □

Theorem 817.3 (Audit-map family cross-map non-escape)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier $F = (T, I, O, R, B, H, C, K, N)$, a cross-map read from one local audit row to another can cite only the source row, the target row, the displayed routing row R, the componentwise transport row H, the displayed continuation row C, the provenance row K, and the local naming row N. It cannot export an unlisted neighbouring map, a hidden route around an obstruction row, or a theorem stronger than the target map itself records.*

Proof. Begin with the routing coordinate. A cross-map read is admitted only through R, so it must name the source and target audit rows as entries of the finite family inventory.

Transport and continuation preserve that route. The row H identifies only displayed coordinates, and C gives only the public path from the family packet to those coordinates. Neither row manufactures a new neighbouring map.

Finally, provenance and naming restrict the citation surface. Since K points back to the concrete audit chapters and N names the local packet, a consumer may reuse an exposed route but cannot bypass an obstruction row or infer a stronger theorem than the target map records. □

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCrossMapNonescape`

Theorem 817.4 (Audit-map family frontier stability)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier, replacing a local audit row by an* hsame*-transported copy preserves the family frontier row B as the same displayed frontier. The transported packet may replay the same inventory, obstruction, routing, provenance, continuation, and naming coordinates, but the frontier table remains a named consumer boundary rather than a closed theorem surface.*

Proof. Use the componentwise transport row H. It transports the family tag, inventory, obstruction, routing, frontier, continuation, provenance, and naming coordinates as displayed coordinates of the same finite carrier shape.

The frontier coordinate is therefore carried as B, not recomputed from neighbouring maps or hidden theorem content. A transported local audit row can change how the same visible row is represented, but it does not change which frontier entries the family packet exposes.

The NameCert row N then binds the transported packet to the same public surface. The frontier remains stable because the only accepted frontier read is still the displayed row B routed through C and cited through K. □

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyFrontierStability`
↪ variant `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier frontier stability`

Theorem 817.5 (Audit-map family frontier non-escape)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier* $F = (T, I, O, R, B, H, C, K, N)$*, no entry of the frontier row* B *can be read as a closed proof of the corresponding frontier claim. A consumer may cite the frontier entry as a named boundary of the audit-map family, but cannot extract subject reduction, confluence, candidate-free normalisation, full closed consistency, host semantic equality, or any other unlisted discharge from that citation.*

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier frontier nonescape`

Proof. Read the frontier row directly. By Definition 817.1, B records consumer-facing claims that are named by the family packet but not closed by it.

Apply the inventory factorization of Definition 817.2. Any consumer read of a frontier entry factors through B and through the displayed continuation and naming rows, so the read never changes class into an inventory theorem or a routing theorem.

Use cross-map non-escape from Definition 817.3. Even when the frontier entry points at a neighbouring audit map, the route can cite only what that neighbouring map itself exposes. Thus the frontier remains a named boundary and cannot discharge an unlisted theorem. □

Falsifiable prediction: An accepted $\mathsf{AuditMapFamily}^{\uparrow}$ packet cannot turn a frontier row into a closed theorem unless that theorem is already present in the finite inventory row I.

Independence witness: metacicbetaaudit, metacic-closure-preservation-audit, substitutionauditmap, metacicnormalizationaudit, auditgatecomposition: these nearest audit siblings expose local beta, closure-preservation, substitution, normalisation, and gate-composition rows, but no bijection preserves the combined inventory, obstruction, routing, and frontier rows of the family packet.

Theorem 817.6 (Audit-map family TasteGate primality commitment)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier*

$$F = (T, I, O, R, B, H, C, K, N),$$

the falsifiability row saying that B *cannot become a closed theorem outside* I *and the sibling-independence row separating the family packet from metacicbetaaudit, metacic-closure-preservation-audit, substitutionauditmap, metacicnormalizationaudit, and auditgatecomposition are carrier-visible commitments of the same finite packet. They are read through* I, O, R, B, H, C, K, N *and not through a hidden theorem inventory, ambient audit equivalence, or unlisted family map.*

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamily tastegate primality commitment`

Proof. Project the carrier of Definition 817.1. The rows I, O, R, B are the only rows that classify inventory, obstruction, routing, and frontier data for the family.

Apply frontier non-escape from Definition 817.5. A frontier read stays in B unless the claimed theorem is already represented in I, so the falsifiability commitment is visible in the same carrier.

Apply inventory factorization and cross-map non-escape. These force sibling reads through R, K, H, C, N and prevent a local audit sibling from preserving all four family rows by a bijection that forgets obstruction or frontier data.

Thus the falsifiability and independence rows are not external prose conditions. They are the TasteGate-facing primality commitments carried by the accepted finite packet. □

Theorem 817.7 (Audit-map family obstruction row exactness)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier* $F = (T, I, O, R, B, H, C, K, N)$*, a blocked or conditional audit fact is read exactly through the obstruction row* O*. Such a fact remains an obstruction entry transported by* H *and read through* C, K, N*; it cannot become an inventory theorem in* I*, a closed frontier row in* B*, or a discharge route absent from the displayed family packet.*

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier obstruction row exactness`

Proof. Read the partition of rows from Definition 817.2. Established local material passes through I, while blocked or conditional material passes through O.

Transport does not change that class. The componentwise row H preserves the displayed family coordinates, and the continuation row C only reads those coordinates; neither row converts an obstruction entry into an inventory theorem.

Finally apply frontier non-escape. Definition 817.5 prevents named frontier boundaries from acting as closed proofs, so an obstruction entry cannot be discharged by re-reading it through B or through an unlisted route. □

Theorem 817.8 (Audit-map family routing row exhaustion)**.** *Every cross-map route in an accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *packet factors through the displayed rows* R, H, C, K, N *and names only local audit rows listed by the finite inventory* I*. The routing row may cite neighbouring beta, substitution, closure-preservation, confluence, normalisation, decidable-checking, typed-example, subject-reduction, or consistency-assembly entries when those entries are present, but it exports no unlisted audit map, hidden obstruction bypass, host semantic equality, or stronger theorem than the cited local row records.*

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier routing row exhaustion`

Proof. Start from cross-map non-escape. Definition 817.3 requires a cross-map read to cite the source row, target row, routing row R, transport row H, continuation row C, provenance row K, and local naming row N.

Use inventory factorization to bind the names. Definition 817.2 forces local audit facts through I, so a route can name only rows present in the finite family inventory.

Obstruction exactness prevents hidden detours. Definition 817.7 keeps blocked and conditional facts in O, so routing cannot bypass an obstruction or strengthen a cited local row into host equality or an unlisted theorem. □

Theorem 817.9 (Audit-map family obligation unblock package)**.** *The* $\mathsf{AuditMapFamily}^{\uparrow}$ *obligation package for the finite family packet* $F = (T, I, O, R, B, H, C, K, N)$ *consists of inventory factorization, cross-map non-escape, frontier stability, frontier non-escape, obstruction row exactness, and routing row exhaustion. This package makes the family obligation-ready while still excluding*

subject reduction, confluence, normalisation, closed consistency, host equality, hidden obstruction discharge, and unlisted family maps unless the corresponding row is displayed in the packet.

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyObligationUnblockPackage`

Proof. Assemble the carrier rows first. Definition 817.2 partitions consumer reads through $T, I, O, R, B, H, C, K, N$, and Definition 817.3 confines neighbouring-map access to displayed source, target, route, transport, continuation, provenance, and naming rows.

Add the frontier rows. Definition 817.4 preserves the frontier under hsame transport, and Definition 817.5 prevents frontier entries from becoming closed proofs.

Finally attach the obstruction and routing rows. Definition 817.7 keeps blocked or conditional facts in O, while Definition 817.8 forces every cross-map route through R, H, C, K, N and the listed local rows. The resulting package is obligation-ready because every visible family read has a concrete row and every excluded theorem lacks a carrier coordinate. □

Corollary 817.10 (Audit-map family strict local obstruction)**.** *For every accepted* $\mathsf{AuditMapFamily}^{\uparrow}$ *carrier*

$$F = (T, I, O, R, B, H, C, K, N),$$

a displayed failure, refusal, separation, or countermodel row in the family packet is a strict local obstruction. It blocks only the named converse, reuse route, target strengthening, or discharge read that is attached to that row by the finite inventory, obstruction, routing, frontier, transport, continuation, provenance, and naming coordinates. It cannot be reinterpreted as a global impossibility theorem, a host theorem about all neighbouring audit maps, hidden external validation, or a proof that an unlisted coordinate exists.

Proof. Read the carrier from Definition 817.1. The accepted packet exposes exactly the rows $T, I, O, R, B, H, C, K, N$. Thus a consumer-facing obstruction must enter through one of the displayed family coordinates: inventory I for listed audit rows, obstruction O for blocked or conditional rows, routing R for neighbouring-map access, frontier B for named open boundaries, transport H for componentwise sameness of displayed rows, continuation C for public readback, provenance K for the source chapters, and naming N for the local certificate.

For a cross-map use, Definition 817.3 gives the stricter route. The read may cite only the source row, the target row, R, H, C, K, and N. Hence the obstruction can block the named converse or reuse route only at the coordinates where that route is displayed; H preserves those coordinates, C reads them, K records their provenance, and N binds the local packet. None of these rows introduces a second target, an unlisted neighbouring map, or a bypass around the obstruction row.

The remaining interpretations are therefore refused by the same finite factorization. A global impossibility theorem would quantify beyond the displayed source and target rows; a host theorem would add semantic strength not recorded by the target audit map; hidden external validation would add a validating coordinate outside $T, I, O, R, B, H, C, K, N$; and an unlisted-coordinate proof would contradict the classifier clause of Definition 817.1. The obstruction is strict because it has exactly the local blocking force of its displayed row and no wider export surface. □

Lean4 ✓ `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyCarrier strict local obstruction`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFamily}^{\uparrow}}$**.**

Constructive story: The packet is generated by a finite audit-family tag, inventory table, obstruction table, routing table, frontier table, componentwise transport, displayed continuation, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditMapFamily}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 817.1, Definition 817.9, the TasteGate single-carrier alignment marker, Definition 817.10, and the obstruction and routing rows display the mature family-level audit surface for the carrier, inventory, frontier, transport, continuation, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{AuditMapFamily}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AuditMapFamilyUp.AuditMapFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close subject reduction, confluence, normalisation, closed consistency, host semantic equality, or hidden obstruction discharge.

Upgrade path: No further theory-axis upgrade is pending for this chapter.

The FiniteTraceEvaluator$^{\uparrow}$ hub routes bounded evaluator traces through the runtime-candidate boundary of chapter 591, the checkpoint consumer of chapter 997, and the MetaCIC critical path of chapter 765 before any child file exposes runtime readback.

Chapter 818

A Concrete Naming Certificate for $\mathsf{FiniteTraceEvaluator}^{\uparrow}$

The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ packet realizes the runtime-candidate surface of Definition 9461.21 inside the reflection-limit boundary of chapter 9461. It sits beside the runtime-candidate boundary packet of chapter 591, the diagonal finite-trace packet of chapter 492, and the finite-trace GAP socket of chapter 504. Those chapters expose, respectively, object-level runtime rows, a certificate-level diagonal trace, and the finite-window boundary; the present packet records a bounded evaluator trace whose accepted result is visible while host validation of the evaluator remains outside the packet. Its bounded evaluator trace is consumed by the runtime checkpoint packet of chapter 997, where checkpoint-state and validation-boundary rows read the finite trace evidence without importing host validation. The validation-boundary row V is read against chapter 586: $\mathsf{ReflectionLimitBoundary}^{\uparrow}$ supplies the truth/runtime boundary that prevents a finite trace evaluator from becoming a self-validating runtime.

Definition 818.1 (Finite trace evaluator carrier)**.** *A* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, T, A, V, H, C, P, N)$$

with the following displayed rows:

1. *I is the input row of coded* CIC^* *expressions, proof objects, and states admitted to the bounded evaluator;*
2. *T is the finite trace-step row, listing the displayed evaluator steps from the input row I without adding an unbounded runtime;*
3. *A is the accepted-result classifier row, grouping exactly the results reached by the displayed trace-step row T;*
4. *V is the validation-boundary row, recording that host validation of the evaluator is not a packet coordinate;*
5. *H records componentwise* hsame *transports for the input, trace-step, accepted-result, validation-boundary, provenance, and local naming rows;*
6. *C records the displayed* Cont *routes by which a consumer reads I, replays T, reaches A, checks the boundary row V, and returns through provenance and local naming;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the structural* CIC^* *coding interface, and chapter 591;*

8. N is the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are I, T, A, V, H, C, P, N. The carrier has no coordinate for a host evaluator, a total truth predicate, an unbounded execution stream, or the judgment that the bounded evaluator validates itself.

Lean4 ✓ `BEDC.Derived.FiniteTraceEvaluatorUp.FiniteTraceEvaluatorTasteGate single carrier alignment`

Theorem 818.2 (FiniteTraceEvaluator runtime-candidate sibling route)**.** *Every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *carrier reads bounded trace admission through the runtime-candidate boundary of chapter 591 and the recognizer-admission route of chapter 1048 before any accepted-result classifier row is exported. Thus I, T, A, V, H, C, P, N record an admitted finite evaluator trace, not a host validator, total truth predicate, or unbounded runtime.*

Proof. Project the carrier of Definition 818.1. The provenance row P already names the runtime-candidate boundary as a source for admitted object-level runtime rows.

Use chapter 1048 for the recognizer-admission side. It supplies the finite route by which a source-level recognizer flow is admitted before a consumer can treat the trace as a displayed row.

The accepted-result classifier A is therefore downstream of the bounded trace row T and the boundary row V. Since no host validator, total truth predicate, or unbounded runtime is a carrier coordinate, the sibling route is exactly the displayed admission route. □

Theorem 818.3 (FiniteTraceEvaluator NameCert obligations)**.** *For the carrier of Definition 818.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission: every accepted object exposes the displayed packet $E = (I, T, A, V, H, C, P, N)$;*
2. *finite trace-step exposure: every evaluator step readable by a consumer is a row of the finite trace-step coordinate T reached from the input coordinate I;*
3. *accepted-result classifier: every accepted result is classified by A only after the corresponding trace-step row in T has been displayed;*
4. *runtime-validation non-escape: the boundary row V keeps host validation outside I, T, A, H, C, P, N and prevents a packet row from serving as its own validator;*
5. *ledger exhaustion: the provenance and local naming rows P, N exhaust the available dependencies and expose no unlisted evaluator, truth predicate, total halting predicate, or infinite trace.*

Consequently the certificate names a bounded object-level evaluator trace and its accepted-result ledger, but it does not internalize the host judgment that licenses the evaluator.

Proof. Carrier admission is tuple projection. The packet $E = (I, T, A, V, H, C, P, N)$ is finite, and every visible object-level coordinate is one of the displayed rows.

Trace-step exposure begins with I and then reads T. Since T is the only trace coordinate, a consumer can inspect only the finitely displayed evaluator steps and cannot replace them by an ambient runtime.

The accepted-result classifier is reached only after that trace read. A result belongs to A because the packet displays the corresponding step in T; no accepted result is supplied by a hidden classifier or by host execution.

Runtime validation is separated by V. The argument of Definition 9461.22 and the sibling packet chapter 591 place validation of a runtime candidate outside its source, pattern, classifier, stability, and ledger fields. Thus the finite trace evaluator records a boundary row rather than a self-validating checker.

Finally, P and N bind provenance and local naming without adding new evaluator content. Ledger exhaustion follows because every available consumer route has already passed through I, T, A, V, H, C, P, N, so an unlisted evaluator, truth predicate, total halting predicate, or infinite trace would require an extra coordinate absent from the carrier. □

Lean4 ✓ `BEDC.Derived.FiniteTraceEvaluatorUp.FiniteTraceEvaluatorCarrier namecert obligations`

Theorem 818.4 (FiniteTraceEvaluator bounded trace coverage)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, every accepted evaluator result factors through*

$$I \longmapsto T \longmapsto A.$$

No accepted result can bypass the bounded trace row T *or be supplied by a host evaluator, a total truth predicate, or an unbounded execution stream.*

Lean4 ✓ `BEDC.Derived.FiniteTraceEvaluatorUp.FiniteTraceEvaluatorCarrier bounded trace coverage`

Proof. Project the carrier of Definition 818.1. The only input-facing coordinate is I, the only trace-step coordinate is the finite row T, and the only accepted-result classifier is A.

Apply Definition 818.3. Its trace-step exposure and accepted-result classifier clauses force a consumer to read from I into T before any accepted result is visible at A.

The sibling runtime boundary of chapter 591, the finite diagonal trace of chapter 492, and the finite-trace GAP socket of chapter 504 supply only bounded object-level routes. They do not add a host evaluator, total truth predicate, or unbounded execution stream to the packet. □

Theorem 818.5 (FiniteTraceEvaluator host-validation refusal ledger)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the row* V *is only a refusal and validation-boundary row. It does not supply host validation, total truth for* CIC^* *sentences, or a self-validating evaluator.*

Proof. Read V through Definition 818.1. The definition names it as the validation-boundary coordinate and explicitly keeps host validation outside the displayed evaluator rows.

Use the runtime-validation non-escape clause of Definition 818.3. It prevents any packet row from becoming the validating host judgment for the same evaluator.

The cited sibling packets chapter 591, chapter 492, and chapter 504 expose runtime candidates, trace rows, and finite windows as object-level evidence. None of them turns V into host validation, total truth, or self-validation. □

Theorem 818.6 (FiniteTraceEvaluator result-validation separation)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, result classification is visible only after bounded trace exposure through* $I \longmapsto T \longmapsto A$*, while validation remains outside the packet at the boundary row* V*. Thus the packet can display a bounded accepted result without claiming host validation of the evaluator that produced it.*

Proof. First apply Definition 818.4. It places every accepted result on the bounded route from I through T to A.

Next apply Definition 818.5. It keeps host validation, total truth, and self-validation outside the same packet and records the refusal at V.

Together with Definition 818.3, these two readings exhaust the accepted packet. The runtime-candidate boundary, halting-distinction trace, and finite-trace GAP socket chapters supply finite object-level anchors, but the validation judgment is not a coordinate of E. □

Theorem 818.7 (FiniteTraceEvaluator route obligation triad)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the input row* I*, finite trace-step row* T*, accepted-result classifier* A*, and validation-boundary row* V *form the root obligation triad plus boundary. The packet exposes no accepted result unless the route has already displayed* I, T, A*, and it exposes no validation claim except through the boundary row* V*.*

Proof. Project the accepted packet through Definition 818.1. The rows I, T, A, V are the only input, finite trace-step, accepted-result, and validation-boundary coordinates available to a consumer.

Apply Definition 818.3 and Definition 818.4. These force accepted results through the displayed route from I into T and then into A.

Finally use Definition 818.5 and Definition 818.6. They keep the validation claim at V rather than allowing any packet row to become host validation. □

Theorem 818.8 (FiniteTraceEvaluator checkpoint consumer route)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

The checkpoint consumer receives no host validator, truth predicate, total halting predicate, or infinite trace from the evaluator packet.

Proof. Begin with the carrier definition Definition 818.1. It displays exactly the rows I, T, A, V, H, C, P, N for the bounded evaluator packet.

Use Definition 818.3 and Definition 818.4. The checkpoint consumer can read accepted-result data only after the finite trace route from I through T to A has been displayed.

Attach Definition 818.5 and Definition 818.6. These keep host validation outside the evaluator packet and place the validation boundary at V.

The runtime checkpoint chapter consumes those finite rows as checkpoint data, not as a host validator. Therefore the route through H, C, P, N preserves the same packet and supplies no truth predicate, total halting predicate, or infinite trace. □

Theorem 818.9 (FiniteTraceEvaluator validation-boundary exhaustion)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the validation row* V *is exhausted as a refusal boundary. It cannot be repacked as evaluator validation, self-validation, host truth, total halting, or an infinite-trace validator for the same packet.*

Proof. Read V from the carrier Definition 818.1. The definition names V only as the validation-boundary row and lists no host-validation coordinate.

Apply Definition 818.3. Its runtime-validation non-escape clause prevents any packet row from serving as its own validator, while its ledger-exhaustion clause leaves no hidden validator in P, N.

Use Definition 818.5 and Definition 818.6. Together they identify V as the boundary at which evaluator validation is refused, not as a route back into validation or self-validation. □

Theorem 818.10 (FiniteTraceEvaluator object-level boundary)**.** *Every consumer read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the finite route*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

The consumer may inspect the coded input, replay the displayed finite trace, and read the accepted-result classifier, but it cannot extract host validation, total halting information, truth for CIC* *sentences, or an unbounded evaluator from the packet.*

Proof. Project the accepted carrier of Definition 818.1. The input, trace-step, accepted-result, and validation-boundary information is exhausted by I, T, A, V.

Apply Definition 818.3. Its trace-step exposure and accepted-result classifier clauses force every evaluator read through T and then through A, while its runtime-validation non-escape clause keeps the host validating judgment outside the packet.

The remaining rows H, C, P, N transport, route, cite, and name the same finite packet. Hence every consumer read factors through the displayed route, and no unbounded evaluator or internal truth classifier is available from the finite trace evaluator. □

Theorem 818.11 (FiniteTraceEvaluator trace-result route)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, every accepted result row classified by* A *is reached only through the bounded trace route*

$$I \to T \to A.$$

The result row cannot be supplied by a host evaluator, a truth predicate for CIC^* *sentences, or an unbounded execution stream.*

Proof. Project the carrier of Definition 818.1. Its only input-facing row is I, its only evaluator-step row is the finite trace row T, and its only accepted-result classifier is A.

Apply Definition 818.3. The finite trace-step exposure clause sends every evaluator step from I into T, and the accepted-result classifier clause permits A to classify a result only after the corresponding displayed trace-step row has been read.

The missing host evaluator, truth, and unbounded execution coordinates give the non-escape conclusion. Since none of those objects is a row of $E = (I, T, A, V, H, C, P, N)$, no accepted result can bypass the route $I \to T \to A$. □

Theorem 818.12 (FiniteTraceEvaluator validation-row non-escape)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the validation-boundary row* V *refuses host validation, total truth, self-validation of the evaluator, and unbounded execution as packet coordinates. A consumer may read that refusal through* V, H, C, P, N*, but cannot turn* V *into any of the refused objects.*

Proof. Start from Definition 818.1. The row V is named only as the validation-boundary row, while host validation, total truth, self-validation, and unbounded execution are explicitly absent from the carrier.

Use the runtime-validation non-escape clause of Definition 818.3. It keeps host validation outside I, T, A, H, C, P, N and prevents a packet row from serving as its own validator.

The transport, continuation, provenance, and naming rows preserve the same finite packet. Thus V records a boundary refusal, and the missing host/evaluator/truth coordinates cannot be reconstructed by reading V, H, C, P, N. □

Theorem 818.13 (FiniteTraceEvaluator provenance exhaustion)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the rows* H, C, P, N *exhaust componentwise transport, consumer continuation, provenance, and local naming for the same finite packet. They do not add host validation, a host evaluator, total truth, total halting, or an unbounded trace.*

Proof. Project the displayed packet in Definition 818.1. After I, T, A, V are fixed, the only remaining rows are the componentwise hsame transports H, the displayed Cont routes C, provenance P, and the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ row N.

Apply the ledger-exhaustion clause of Definition 818.3. It states that P, N exhaust the available dependencies and expose no unlisted evaluator, truth predicate, total halting predicate, or infinite trace.

The rows H and C only transport and replay the same finite coordinates already displayed by the carrier. Therefore H, C, P, N close the local transport, continuation, provenance, and naming surface without importing any missing host/evaluator/truth coordinate. □

Theorem 818.14 (FiniteTraceEvaluator result classifier exactness)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, a result is classified by the accepted-result row* A *if and only if the consumer has first read the displayed finite trace-step row* T *from the input row* I*. The classifier has no independent host execution channel, no truth predicate for* CIC^* *sentences, and no unbounded trace coordinate.*

Proof. Project the carrier of Definition 818.1. The row A is displayed only as the accepted-result classifier of the same finite packet whose trace-step coordinate is T.

Apply the finite trace-step exposure and accepted-result classifier clauses of Definition 818.3. They force every accepted result to be read after the corresponding displayed trace-step row, and they provide no second classifier outside I, T, A, V, H, C, P, N.

Conversely, once the consumer reads I and the corresponding finite step in T, the displayed classifier row A is the only accepted-result row available in the packet. Thus result classification is exact at A and cannot be replaced by host execution, a truth predicate, or an unbounded trace. □

Theorem 818.15 (FiniteTraceEvaluator ledger non-escape)**.** *The finite ledger* I, T, A, V, H, C, P, N *of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exports no host validator, truth predicate, total halting predicate, or infinite trace. Every readable evaluator result is exhausted by the input row, the displayed finite trace-step row, the accepted-result classifier, the validation-boundary row, the componentwise transport and continuation rows, provenance, and the local name.*

Proof. Start with the object-level boundary of Definition 818.10. It sends every consumer read through

$$I \to T \to A \to V \to H \to C \to P \to N.$$

Use the runtime-validation non-escape and ledger exhaustion clauses of Definition 818.3. The row V records that host validation is outside the packet, while P and N close the visible provenance and naming ledger without adding evaluator content.

Therefore an exported host validator, truth predicate, total halting predicate, or infinite trace would require an additional coordinate not present in the carrier. The displayed finite rows exhaust the packet's consumer-facing ledger. □

Theorem 818.16 (FiniteTraceEvaluator obligation upgrade package)**.** *The* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *obligation surface needed for obligationClosure is the package consisting of carrier admission, result classifier exactness, runtime-validation non-escape, ledger exhaustion, and consumer-route factorization for the same finite packet* $E = (I, T, A, V, H, C, P, N)$*. This package is object-level only: it does not assert host validation of the evaluator, total halting, truth for* CIC^* *sentences, or an infinite execution trace.*

Proof. Carrier admission is supplied by Definition 818.1 and the first clause of Definition 818.3. They fix the available rows as I, T, A, V, H, C, P, N.

Result classifier exactness is exactly Definition 818.14. It ties accepted results to the displayed finite trace-step row before A is read.

Runtime-validation non-escape and ledger exhaustion are supplied by the corresponding clauses of Definition 818.3 and by Definition 818.15. These keep host validation, truth, total halting, and infinite traces outside the packet.

Finally apply Definition 818.10. The consumer route factors through the displayed finite route, so the listed rows form the whole obligation surface for the bounded evaluator packet. □

Theorem 818.17 (FiniteTraceEvaluator evaluation determinacy)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, an accepted result row is exposed only through the displayed bounded trace rows* T, R, V, H, C, P, N*, where* R *is the result row read at* A*. The packet has no second result channel, host evaluator coordinate, truth predicate, or unbounded execution stream from which a consumer could read an accepted result.*

Proof. Project the carrier of Definition 818.1. The only input-to-result route available to a consumer begins at I, proceeds through the finite trace row T, and reaches the accepted-result classifier row A. Reading A exposes the result row R as part of that same displayed route.

Apply Definition 818.3. Its finite trace-step exposure and accepted-result classifier clauses say that accepted results are classified only after the corresponding bounded trace rows have been displayed.

Finally use Definition 818.10. The remaining rows V, H, C, P, N keep validation, transport, continuation, provenance, and local naming on the same finite path, so no hidden evaluator or ambient truth predicate can determine an additional result. □

Theorem 818.18 (FiniteTraceEvaluator boundary exhaustion)**.** *Every runtime-validation read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet remains a boundary read through* V, H, C, P, N*. No row exported by the packet is a self-validating runtime coordinate, and no accepted result row can be promoted to a host validator for the same evaluator.*

Proof. Start with the validation-boundary coordinate in Definition 818.1. The carrier lists V as the place where host validation is recorded as outside the packet, not as a runtime-coordinate constructor.

Apply the runtime-validation non-escape clause of Definition 818.3. It keeps host validation outside I, T, A, H, C, P, N and prevents any packet row from serving as its own validator.

Replay Definition 818.10. Every consumer route passes from A to V and then through H, C, P, N, so runtime evidence remains boundary evidence throughout the displayed finite ledger. □

Theorem 818.19 (FiniteTraceEvaluator runtime nonimport)**.** *The object-level runtime evidence read by an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is routed through chapter 591, chapter 492, and chapter 504. The packet does not import those rows as a host validator for its own evaluator; it only records their finite boundary, diagonal trace, and finite-window socket as displayed object-level dependencies.*

Proof. Project the provenance and local naming rows P, N of the carrier. Their listed dependencies include the runtime-candidate boundary, the diagonal finite-trace packet, and the finite-trace GAP socket as object-level sources for the finite evaluator trace.

Apply Definition 818.3. Carrier admission and ledger exhaustion restrict the dependency read to the displayed rows I, T, A, V, H, C, P, N, while runtime-validation non-escape keeps a host validator outside the packet.

Finally replay Definition 818.10. The consumer may inspect the bounded trace and accepted-result classifier through those dependency routes, but the route never exports total halting, host validation, truth for CIC* sentences, or an unbounded evaluator. □

Theorem 818.20 (FiniteTraceEvaluator public consumer package)**.** *Every public consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet reads bounded trace coverage, accepted-result classifier exactness, validation-row non-escape, provenance, and local* $\mathsf{NameCert}$ *as one package. The consumer route is the finite ledger*

$$I \to T \to A \to V \to H \to C \to P \to N,$$

not a separate host-runtime validation channel.

Proof. Start from the carrier of Definition 818.1. It lists exactly the rows I, T, A, V, H, C, P, N available to a public consumer.

Use Definition 818.4 and Definition 818.14. These theorems place every accepted result on the route from the input row through the bounded trace row to the accepted-result classifier.

Use Definition 818.12 and Definition 818.19. The validation row remains a boundary refusal, and the runtime dependencies are read only as displayed object-level provenance.

Finally apply Definition 818.3. The transport, continuation, provenance, and local naming rows bind the same finite route, so a public consumer receives one package rather than an independent host-runtime channel. □

Theorem 818.21 (FiniteTraceEvaluator runtime-boundary totality)**.** *Runtime validation for an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is total only through the displayed bounded trace row* T *and result-validation rows* A, V*. No consumer can import a host runtime, host validator, total truth predicate, or unbounded evaluator outside the finite packet.*

Proof. Project the packet $E = (I, T, A, V, H, C, P, N)$ from Definition 818.1. The only runtime-facing rows are the bounded trace row T, the accepted-result classifier A, and the validation-boundary row V.

Apply Definition 818.18. It keeps every runtime-validation read as a boundary read through V, H, C, P, N and prevents an accepted result row from becoming a host validator.

Apply Definition 818.19. The cited runtime-candidate, diagonal-trace, and finite-window dependencies remain object-level sources and never enter the carrier as a host runtime.

Thus runtime validation is total only at the displayed boundary route T, A, V, H, C, P, N. Any stronger validation would need an additional coordinate absent from the carrier. □

Theorem 818.22 (FiniteTraceEvaluator accepted-result ledger exhaustion)**.** *The accepted result ledger of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is exhausted by the trace rows, result classifier, validation row, provenance, and local name. A consumer can read* T, A, V, P, N *only as part of the finite packet* $E = (I, T, A, V, H, C, P, N)$*, and cannot recover a host evaluator, total truth predicate, total halting predicate, or infinite trace from that ledger.*

Proof. Begin with Definition 818.14. It says a result is classified by A exactly after the displayed finite trace-step row T has been read from I.

Use Definition 818.18 to attach the validation boundary V to that accepted result without turning it into host validation.

Use Definition 818.19 and the ledger exhaustion clause of Definition 818.3. Provenance and local naming record the finite object-level dependencies but expose no unlisted evaluator, truth predicate, total halting predicate, or infinite trace.

Therefore the accepted result ledger is exactly the displayed trace, classifier, validation, provenance, and naming package, with no further runtime content available to the consumer. □

Theorem 818.23 (FiniteTraceEvaluator runtime-checkpoint boundary)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet reads only the displayed evaluator rows* I, T, A, V, H, C, P, N*, together with the runtime-candidate boundary and checkpoint-state rows supplied by the checkpoint packet. The read never imports host validation, a total truth predicate, a total halting predicate, an unbounded evaluator, or an infinite execution stream.*

Proof. Project the accepted evaluator packet through Definition 818.1. The consumer can see the coded input row I, finite trace row T, accepted-result row A, validation-boundary row V, componentwise transport, continuation, provenance, and local name.

Apply Definition 818.4 and Definition 818.14. These rows force every accepted result read through the finite trace route $I \to T \to A$ before any runtime-facing checkpoint can inspect it.

Next use Definition 818.12 and Definition 818.19. The validation row remains a boundary row, and the runtime-candidate and checkpoint-state data are read as object-level packet rows rather than as a host runtime.

Finally attach provenance exhaustion from Definition 818.13 and accepted-result ledger exhaustion from Definition 818.22. The checkpoint consumer has no carrier coordinate left for host validation, a total truth predicate, a total halting predicate, an unbounded evaluator, or an infinite execution stream. □

Theorem 818.24 (FiniteTraceEvaluator trace-window accountability)**.** *Every accepted result row of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet has a finite trace-window coordinate in* T*. The accepted result cannot be supplied by an unbounded execution stream, by a host evaluator, or by a total halting predicate outside the displayed packet* $E = (I, T, A, V, H, C, P, N)$*.*

Proof. Begin with Definition 818.4. It places the trace evidence for every accepted result inside the finite row T of the displayed evaluator packet.

Use Definition 818.14. The row A classifies a result only after the finite trace-step row has been read from I, so the accepted result has no source separate from the finite trace window.

Now apply Definition 818.19. The external runtime-candidate and diagonal-trace objects remain dependencies at the boundary and do not enter the packet as an unbounded execution stream.

Finally use Definition 818.22. The ledger exposes only T, A, V, P, N inside the finite packet, leaving no row that could supply an accepted result from a host evaluator or total halting predicate. □

Theorem 818.25 (FiniteTraceEvaluator validation refusal package)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the validation-boundary row* V *remains a refusal package after transport through* H, C, P, N*. It records the boundary at which host validation is not imported, and it cannot be converted into a self-validator for the same accepted packet.*

Proof. Project the carrier $E = (I, T, A, V, H, C, P, N)$. The only validation-facing row is V, while H, C, P, N provide componentwise transport, continuation, provenance, and local naming.

Apply Definition 818.12. The row V refuses to become the validating host judgment for its own evaluator packet.

Next use Definition 818.18 and Definition 818.13. Transport through H, C, P, N preserves the displayed boundary route and records its finite dependencies without adding a validator row.

Finally apply Definition 818.22. The accepted result ledger is already exhausted by the finite trace, classifier, validation boundary, provenance, and name, so the refusal package cannot be repackaged as a self-validator. □

Theorem 818.26 (FiniteTraceEvaluator nontrivial carrier pair)**.** *The* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *carrier has at least two distinguishable packet shapes. One admitted shape exposes an input-only finite trace packet with rows* I, T, H, C, P, N *and no accepted-result row. A second admitted shape exposes the same finite input and trace surface together with a displayed accepted-result row* A *and validation-boundary row* V*. The local classifier can therefore distinguish an input-only trace packet from a packet whose finite trace carries an accepted-result row.*

Proof. Start from the carrier row of Definition 818.1. The row I supplies the coded input, T supplies the finite trace-window coordinate, and H, C, P, N record transport, continuation, provenance, and the local name.

For the input-only shape, read only I, T, H, C, P, N. This is still a finite trace packet because Definition 818.4 places trace evidence inside T and does not require an accepted result to be displayed by every finite trace read.

For the accepted-result shape, add the rows A, V. By Definition 818.14, the row A is an accepted-result classifier only after the finite trace has been read from I, T, and by Definition 818.25 the row V records the validation boundary rather than a host validator.

The two shapes are not interchangeable as public rows. Removing A, V loses the accepted-result and validation-boundary route, while adding them creates a displayed result row that the input-only packet does not expose. Hence the carrier is not a one-shape packet. □

Theorem 818.27 (FiniteTraceEvaluator field-faithful rows)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

equality of the rows I, T, A, V, H, C, P, N *fixes the public evaluator read. Changing any one of* I, T, A, V *changes the corresponding public route: the input route, the finite trace-window route, the accepted-result classifier route, or the validation-boundary route.*

Proof. First fix all eight displayed rows. The carrier definition and Definition 818.3 leave no additional public coordinate beyond the coded input, finite trace, accepted-result classifier, validation boundary, transport, continuation, provenance, and local name.

Next identify the four route-sensitive fields. Changing I changes the coded input route before any finite trace can be read. Changing T changes the finite trace-window route controlled by Definition 818.4.

Changing A changes the accepted-result classifier route of Definition 818.14. Changing V changes the validation-boundary route of Definition 818.12.

The remaining rows H, C, P, N transport and name those displayed routes. Thus row equality fixes the public evaluator read, and a change in any of I, T, A, V is visible as a change in the corresponding route. □

Theorem 818.28 (FiniteTraceEvaluator TasteGate readiness)**.** *The later* TasteGate *formalization for* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *has two explicit obligations. It must exhibit a nontrivial carrier witness, formalizing the separation in Definition 818.26, and it must exhibit a field-faithful public read, formalizing the route sensitivity in Definition 818.27. These obligations are stated over the same rows* I, T, A, V, H, C, P, N *and the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *surfaces used by the chapter.*

Proof. The nontrivial obligation is supplied at the paper level by the two displayed packet shapes. The input-only packet lacks A, V, while the accepted-result packet carries A, V as visible rows; the classifier can test that difference without importing a host evaluator.

The field-faithful obligation is supplied at the paper level by the public route analysis. Equality of I, T, A, V, H, C, P, N fixes the exposed evaluator read, and changing one of I, T, A, V changes its named route before transport or provenance can hide it.

The later Lean file may encode these as Nontrivial $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ and FieldFaithful $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ obligations. No new carrier row is needed for that encoding; the rows are already the displayed packet surface of this chapter. □

Theorem 818.29 (FiniteTraceEvaluator obligation closure status)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

the carrier row, the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *obligations, the trace-result route, validation-row non-escape, provenance exhaustion, result classifier exactness, the nontrivial carrier pair, the field-faithful public rows, and the* TasteGate *readiness surface assemble the paper-side obligation-closure package over* I, T, A, V, H, C, P, N*. The package is bounded to the displayed evaluator packet and does not assert Lean formalization, host validation, total truth, total halting, or unbounded evaluation.*

Proof. The carrier and local naming surface are fixed by Definition 818.1 and Definition 818.3. They display the rows I, T, A, V, H, C, P, N and restrict the available obligations to carrier admission, trace exposure, accepted-result classification, validation non-escape, and ledger exhaustion.

The evaluator-result part is supplied by Definition 818.11 and Definition 818.14. These place every accepted result on the finite route from I through T to A.

The boundary and ledger part is supplied by Definition 818.12 and Definition 818.13. Thus every validation read remains at V, H, C, P, N, and provenance introduces no host validator, truth predicate, total halting predicate, or infinite trace.

Finally, Definition 818.26, Definition 818.27, and Definition 818.28 give the public carrier separation and route sensitivity required by the later TasteGate surface. These references assemble the paper-side obligation package while leaving the verification and bridge axes unchanged. □

Theorem 818.30 (FiniteTraceEvaluator root trace-result totality)**.** *Every root-level downstream consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet that reads accepted-result data first factors through*

$$I \to T \to A.$$

The consumer may then continue through the remaining rows of $E = (I, T, A, V, H, C, P, N)$*, but it cannot read an accepted result before the coded input and finite trace-step rows have been displayed.*

Proof. Begin with the bounded trace coverage of Definition 818.4. It sends every accepted evaluator result through the displayed route $I \to T \to A$.

Use Definition 818.11 and Definition 818.14. They make A exact for accepted results only after the finite trace-step row in T has been read from I.

Any root-level downstream consumer is still a consumer of the same displayed packet. By Definition 818.20 and Definition 818.23, the public and checkpoint routes continue through V, H, C, P, N without adding a second accepted-result channel. □

Theorem 818.31 (FiniteTraceEvaluator root validation boundary)**.** *Every root-level validation read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet terminates at the displayed boundary, transport, continuation, provenance, and local naming rows* V, H, C, P, N*. Such a read cannot become host validation, total truth for* CIC^* *sentences, total halting, or unbounded evaluation.*

Proof. Start with Definition 818.12. It states that V is a boundary row and refuses host validation, total truth, self-validation, and unbounded execution as packet coordinates.

Attach the surrounding transport and naming rows by Definition 818.18 and Definition 818.25. These keep the validation read on the route through V, H, C, P, N.

Finally apply Definition 818.13 and Definition 818.22. The ledger is exhausted by the displayed finite packet, so no root-level read can recover a host validator, total truth predicate, total halting predicate, or unbounded evaluator from the validation boundary. □

Theorem 818.32 (FiniteTraceEvaluator root boundary package)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

every root-facing $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer reads one finite boundary package consisting of the root trace-result route, the root validation boundary, provenance exhaustion, and the checkpoint consumer route. The package exposes accepted results only through* $I \to T \to A$ *and validation-facing data only through* V, H, C, P, N*; it does not contain host validation, total truth, self-validation, total halting, or unbounded evaluation.*

Proof. Project the carrier of Definition 818.1. The accepted packet has exactly the rows I, T, A, V, H, C, P, N, so a root-facing checkpoint consumer has no coordinate outside the displayed finite evaluator packet.

Apply Definition 818.30. It places every accepted-result read on the route $I \to T \to A$ before any consumer can continue through the rest of the packet.

Apply Definition 818.31. It sends every validation-facing read to the boundary, transport, continuation, provenance, and local naming rows V, H, C, P, N.

Finally use Definition 818.13 and Definition 818.8. Provenance exhausts the available dependency rows, and the checkpoint route is already the same finite ledger I, T, A, V, H, C, P, N; hence no host-validation row, truth predicate, self-validator, total halting predicate, or unbounded evaluator can be repacked from the root boundary package. □

Theorem 818.33 (FiniteTraceEvaluator public root export)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the public export surface consists exactly of coded input rows, finite trace-step rows, accepted-result classifier rows, validation-boundary rows, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row, and the checked TasteGate marker* `BEDC.Derived.FiniteTraceEvaluatorUp.taste_gate`*. The export contains no host validation, total halting predicate, truth predicate for* CIC^* *sentences, unbounded evaluator, or bridge-level semantics.*

Proof. Project the carrier rows named in Definition 818.1. They are exactly the coded input, trace-step, accepted-result classifier, validation-boundary, transport, continuation, provenance, and local naming rows of E.

Apply Definition 818.30. It places every accepted-result read on the route $I \to T \to A$ before any public consumer can inspect the remaining rows.

Apply Definition 818.31 and Definition 818.32. They make V, H, C, P, N the full validation-facing boundary and keep the checkpoint consumer inside the same displayed packet.

The checked TasteGate marker records that the same finite packet is visible as one carrier for downstream reflection consumers. Since the refused objects have no carrier row, the public export surface has no host validator, total halting predicate, truth predicate, unbounded evaluator, or bridge-level semantic row. □

Falsifiable prediction: An accepted $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ packet whose public read treats the input-only trace shape and the accepted-result shape as the same packet fails the carrier separation witnessed by the displayed A, V rows.

Independence witness: runtimecandidateboundary, haltingdistinctiontrace, finite-trace-gap-socket, reflection-runtime-checkpoint: none is bijective to $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ because those siblings expose runtime candidate, halting distinction, gap socket, or reflection checkpoint rows, while this carrier binds coded input, finite trace, accepted-result classifier, validation boundary, transport, continuation, provenance, and local naming in one evaluator packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ packet is generated from coded input rows, finite trace-step rows, accepted-result classifier rows, a validation-boundary row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 818.1, Definition 818.3, Definition 818.29, Definition 818.30, Definition 818.31, Definition 818.32, Definition 818.33, and

Lean4 ✓ `BEDC.Derived.FiniteTraceEvaluatorUp.taste gate`

bind the finite evaluator trace carrier, accepted-result classifier, validation boundary, ledger exhaustion surface, downstream consumer boundary, public export surface, and TasteGate instance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FiniteTraceEvaluatorUp.FiniteTraceEvaluatorTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close host validation, total halting, truth for CIC^* sentences, unbounded evaluation, infinite traces, or bridge-level public semantics.

Upgrade path: No paper-side closure upgrade remains before bridge checking for the same finite evaluator packet and its checked TasteGate carrier.

Theorem 818.34 (FiniteTraceEvaluator root checkpoint totality)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$*-facing read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the displayed finite rows*

$$I, T, A, V, H, C, P, N.$$

Thus a root checkpoint consumer reads coded input, finite trace steps, accepted-result classification, validation boundary, componentwise transport, displayed continuation, provenance, and local naming as one finite evaluator package.

Proof. Project the carrier of Definition 818.1. Its visible coordinates are exactly I, T, A, V, H, C, P, N, and the local NameCert obligations expose those rows as the available evaluator surface.

Apply Definition 818.32. A root-facing checkpoint consumer receives the finite boundary package built from the trace-result route and the validation boundary, with provenance and local naming attached to the same packet.

Finally use Definition 818.40. The checkpoint source is locked to $I \to T \to A$ for accepted results and to V, H, C, P, N for validation-facing reads, so every root checkpoint read factors through the listed rows. □

Theorem 818.35 (FiniteTraceEvaluator root validation refusal)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$*-facing request for host validation, total truth for* CIC^* *sentences, total halting, or unbounded evaluation stops at the displayed validation-boundary, transport, continuation, provenance, and local naming rows* V, H, C, P, N *of the accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet.*

Proof. Begin with Definition 818.31. It sends root-level validation reads to V, H, C, P, N and refuses host validation, total truth, total halting, and unbounded evaluation as packet coordinates.

Use Definition 818.41. The accepted-result route remains separate from the validation boundary, and the validation-facing route cannot certify a host judgment or total classifier.

Apply Definition 818.42. The runtime checkpoint handoff exposes only the source-locked evaluator rows, so the listed requests terminate at the refusal surface rather than becoming new root evidence. □

Theorem 818.36 (FiniteTraceEvaluator root consumer readiness)**.** *The root checkpoint totality row of Definition 818.34 and the validation refusal row of Definition 818.35 provide the exact finite root package consumed by downstream reflection-runtime chapters. The package is the displayed evaluator route I, T, A, V, H, C, P, N and no host validation, total truth, total halting, or unbounded evaluation is added by the consumer boundary.*

Proof. First combine the two rows just stated. The totality theorem supplies the checkpoint-facing route through all displayed evaluator coordinates, while the refusal theorem fixes where validation-facing requests stop.

Next attach the existing root package theorem, Definition 818.32. It already identifies the finite root-facing package consumed by checkpoint readers and excludes host validation, total truth, self-validation, total halting, and unbounded evaluation.

Finally use Definition 818.42. Downstream reflection-runtime chapters receive the same handoff package I, T, A, V, H, C, P, N, so the consumer boundary is ready without adding any row outside the finite evaluator packet. □

Theorem 818.37 (FiniteTraceEvaluator root runtime totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every root runtime read factors through the finite surface*

$$I, T, A, V, H, C, P, N.$$

The runtime consumer reads coded input I, finite trace T, accepted-result row A, validation boundary V, componentwise transports H, displayed continuations C, provenance P, and the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N as one bounded evaluator packet, and it receives no host evaluator or unbounded runtime coordinate.*

Proof. Project the carrier of Definition 818.1 and the local obligations of Definition 818.3. They expose exactly the input, finite trace, accepted-result, validation, transport, continuation, provenance, and local naming rows.

Use Definition 818.34. It already forces root checkpoint consumers through the same displayed packet I, T, A, V, H, C, P, N before any runtime-facing handoff can inspect the accepted result.

Finally attach Definition 818.42. The handoff exposes only the source-locked evaluator rows, so every root runtime read factors through the displayed packet and cannot import a host evaluator or unbounded runtime coordinate. □

Theorem 818.38 (FiniteTraceEvaluator bounded replay ledger exhaustion)**.** *Bounded replay of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exposes only the coded input, finite trace, accepted result, validation boundary, componentwise transports, displayed continuations, provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *rows. Its root ledger is exhausted by*

$$I, T, A, V, H, C, P, N,$$

and no hidden runtime state, host validation row, unbounded trace semantics, or unchecked compiler theorem is present in the replay surface.

Proof. Begin with Definition 818.53. Bounded replay preserves the route through $I \to T \to A$, the validation boundary V, and the continuation rows C under the displayed componentwise transports H.

Next use Definition 818.58 and Definition 818.59. The exactness surface cuts every root checkpoint read at I, T, A, V, H, C, P, N and blocks host validation, total truth, total halting, unbounded evaluation, infinite traces, and second checkpoint sources.

The ledger factorization is then Definition 818.54. It repacks the replay, refusal, provenance, and local naming rows into the same finite root ledger, so the bounded replay surface has no hidden runtime state or unchecked compiler theorem. □

Theorem 818.39 (FiniteTraceEvaluator checkpoint consumer nonescape)**.** *Every checkpoint consumer of the root runtime surface in Definition 818.37 receives only the finite evaluator packet I, T, A, V, H, C, P, N. It receives no host evaluator, total truth predicate, total halting classifier, infinite trace, unchecked compiler theorem, or second checkpoint source.*

Proof. First apply Definition 818.35. Validation-facing requests stop at V, H, C, P, N and do not become host validation, total truth, total halting, or unbounded evaluation evidence.

Next combine Definition 818.60 with Definition 818.38. The checkpoint ledger and bounded replay ledger expose the same finite packet and no infinite trace, hidden runtime state, unchecked compiler theorem, or second checkpoint source.

Finally use Definition 818.36. Downstream reflection-runtime consumers receive the exact finite package I, T, A, V, H, C, P, N, so checkpoint consumption cannot escape the displayed root runtime surface. □

Theorem 818.40 (FiniteTraceEvaluator checkpoint source lock)**.** *Every root-facing checkpoint consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet locks its accepted-result source to the displayed finite route*

$$I \to T \to A$$

and its validation-facing boundary to V, H, C, P, N. The consumer may read the root checkpoint package, but it cannot replace the coded input, finite trace, accepted-result classifier, or validation boundary by host validation, total truth, total halting, unbounded execution, or a second source ledger.

Proof. Begin with the carrier and NameCert obligations, Definition 818.1 and Definition 818.3. They display exactly the rows I, T, A, V, H, C, P, N and make trace-step exposure, accepted-result classification, validation non-escape, transport, continuation, provenance, and local naming part of one packet.

Apply Definition 818.4 and Definition 818.30. These force every accepted-result read through $I \to T \to A$ before a root-facing checkpoint consumer can continue.

Finally use Definition 818.31 and Definition 818.32. The validation read stays on V, H, C, P, N, so no host validator, total truth predicate, total halting classifier, unbounded evaluator, or second source ledger is available. □

Theorem 818.41 (FiniteTraceEvaluator validation nonescape totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every validation-facing root read is total only through the displayed boundary row V together with H, C, P, N. The accepted-result route $I \to T \to A$ remains separate from the validation boundary, and neither route can certify host validation, total truth for* CIC^* *sentences, self-validation, total halting, or unbounded evaluation.*

Proof. First apply Definition 818.12. It identifies V as a boundary row and refuses host validation, total truth, self-validation, and unbounded execution as packet coordinates.

Next use Definition 818.6. The accepted-result route through I, T, A and the validation-facing route through V remain distinct rows of the same evaluator packet.

Attach the root boundary with Definition 818.31 and Definition 818.32. The surrounding transport, continuation, provenance, and local naming rows are exactly H, C, P, N, so validation cannot escape to a host judgment or total classifier. □

Theorem 818.42 (FiniteTraceEvaluator runtime boundary handoff)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *handoff from an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the finite trace evaluator carrier, bounded trace coverage, validation non-escape, result-validation separation, root trace-result totality, root validation boundary, and root boundary package. The handoff exposes only the source-locked evaluator rows* I, T, A, V, H, C, P, N.

Proof. Project the checkpoint consumer route from Definition 818.8 and Definition 818.23. The runtime checkpoint receives the same evaluator rows and no extra validation or unbounded-execution coordinate.

Use Definition 818.40. It locks the accepted-result source to $I \to T \to A$ and the validation-facing surface to V, H, C, P, N.

Finally apply Definition 818.41. The handoff therefore stays inside the displayed evaluator carrier and refuses host validation, total truth, total halting, self-validation, and unbounded evaluation. □

Theorem 818.43 (FiniteTraceEvaluator source admission)**.** *Every accepted-result read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet begins from the displayed input row* I *and finite trace row* T*. The accepted-result classifier* A *may be consumed only after the coded input has entered the bounded trace route, and the read does not import a host source, infinite trace, or second checkpoint ledger.*

Proof. Project the carrier of Definition 818.1. Its accepted-result surface is displayed by the rows I, T, A, with I naming the coded input, T naming the finite trace steps, and A naming the accepted-result classifier.

Apply Definition 818.4. It forces every accepted result onto the bounded route from I through T before the classifier row can be read.

Use Definition 818.40. The checkpoint-facing source remains locked to $I \to T \to A$, so no host source, infinite trace, or second checkpoint ledger is available to the consumer. □

Theorem 818.44 (FiniteTraceEvaluator ledger exhaustion)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the provenance row* P *and local naming row* N *expose all dependency rows available to checkpoint consumers. The packet exposes no host validator, total truth predicate, total halting classifier, unbounded evaluator, or infinite trace beyond the finite rows*

$$I, T, A, V, H, C, P, N.$$

Proof. Start from Definition 818.1. The dependency-carrying rows visible at the checkpoint boundary are I, T, A, V, H, C, P, N, with P recording package provenance and N recording the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ row.

Apply Definition 818.9. It confines validation-facing reads to the displayed validation boundary and prevents a host validating judgment from becoming a packet coordinate.

Next use Definition 818.22. The accepted-result route, validation boundary, transport, continuation, provenance, and local naming rows account for the available ledger surface.

Finally attach Definition 818.42. The checkpoint handoff exposes only I, T, A, V, H, C, P, N, excluding host validation, total truth, total halting, unbounded evaluation, and infinite traces. □

Theorem 818.45 (FiniteTraceEvaluator consumer factorization)**.** *Every checkpoint consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through*

$$I, T, A, V, H, C, P, N.$$

The consumer receives a finite evaluator route: coded input, finite trace, accepted-result classifier, validation boundary, componentwise transport, displayed continuation, provenance, and local naming in that order of dependency.

Proof. Begin with Definition 818.8. It identifies the checkpoint-facing route through the finite evaluator carrier and excludes an external evaluator from the consumer surface.

Use Definition 818.43. The accepted-result side of the consumer route must begin with I and T before A can be consumed.

Apply Definition 818.44. The validation boundary and dependency ledger are exhausted by V, H, C, P, N, with no host validator or infinite trace left outside the packet.

Thus every checkpoint consumer factors through the displayed rows I, T, A, V, H, C, P, N as one finite evaluator route. □

Theorem 818.46 (FiniteTraceEvaluator root image row)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, each accepted read exposes only the displayed image rows*

$$I, T, A, V, H, C, P, N.$$

The rows name the coded input, finite trace, accepted-result classifier, validation boundary, componentwise transport, displayed continuation, package provenance, and local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. No accepted read adds a host validator, total truth predicate, total halting classifier, unbounded evaluator, or infinite trace to the root image.*

Proof. Project the carrier in Definition 818.1. Its root surface is displayed by I, T, A, V, H, C, P, N and no additional row.

Apply Definition 818.43 to the accepted-result side and Definition 818.44 to the dependency side. Together they place every accepted read on the displayed image rows.

Finally use Definition 818.59. Any root checkpoint read that reaches the image surface factors through the accepted result route or the validation and ledger route, so the excluded host rows remain outside the packet. □

Theorem 818.47 (FiniteTraceEvaluator root readback exactness)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, checkpoint readback returns through*

$$I \to T \to A$$

for accepted results, and through V, H, C, P, N *for validation boundary reads. These two readback routes share the same finite root packet and expose no host validation, total truth, total halting, unbounded evaluation, infinite trace, or second checkpoint source.*

Proof. Begin with Definition 818.30. It sends accepted-result checkpoint reads through $I \to T \to A$ before the accepted-result classifier is consumed.

Next use Definition 818.31. It sends validation boundary reads to V, H, C, P, N and prevents host validation, total truth, total halting, and unbounded evaluation from becoming readback coordinates.

The checkpoint source lock, Definition 818.40, ties both routes to the same finite source. Hence checkpoint readback is exact on the two displayed routes and does not create a second checkpoint source or infinite trace. □

Theorem 818.48 (FiniteTraceEvaluator root ledger factorization)**.** *Every downstream* ReflectionRuntimeCheckpoin *consumer of an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet factors through the image row of Definition 818.46 and the readback routes of Definition 818.47. The factorization uses only I, T, A, V, H, C, P, N and imports no host validation, total truth, total halting, unbounded evaluation, or infinite trace.*

Proof. First build the image surface with Definition 818.46. It gives the complete root row list visible to accepted reads.

Then apply Definition 818.47. The checkpoint consumer reads accepted results through $I \to T \to A$ and validation boundary rows through V, H, C, P, N.

Finally attach Definition 818.45 and Definition 818.60. They force the runtime checkpoint consumer to remain on those image and readback rows, with no coordinate for host validation, total truth, total halting, unbounded evaluation, or an infinite trace. □

Theorem 818.49 (FiniteTraceEvaluator root source image readback lock)**.** *For an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

every root source image readback is locked to the coded input row I, finite trace row T, accepted-result row A, validation-boundary row V, componentwise hsame *transport H, displayed* Cont *replay C, package provenance P, and local* NameCert *row N. The readback cannot replace this source image by a host validator, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source.*

Proof. Begin with the carrier, checkpoint source lock, and image row: Definition 818.1, Definition 818.40, and Definition 818.46. They identify I, T, A, V, H, C, P, N as the complete source image available to root-facing reads.

Next apply root readback exactness from Definition 818.47. Accepted results return through $I \to T \to A$, while validation-facing reads return through V, H, C, P, N.

Finally use the ledger factorization of Definition 818.48. Since the checkpoint ledger is already cut to the displayed evaluator rows, no host validator, total truth row, halting classifier, unbounded evaluator, infinite trace, or second checkpoint source can be read back. □

Theorem 818.50 (FiniteTraceEvaluator root checkpoint validation totality)**.** *For an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

every root checkpoint validation request is total only on the displayed evaluator route. It must first read the source image I, T, A, then the validation-boundary row V, and then the structural rows H, C, P, N; it does not produce host validation, total truth for CIC* *sentences, total halting, self-validation, unbounded evaluation, or a second validation ledger.*

Proof. Use the checkpoint source lock and validation nonescape totality from Definition 818.40 and Definition 818.41. These separate the accepted-result source route $I \to T \to A$ from the validation-facing boundary V, H, C, P, N.

Then apply root validation refusal and root readback exactness from Definition 818.35 and Definition 818.47. The validation request stops at the displayed boundary and cannot become a host-side judgment.

Finally attach the root source image readback lock. It repacks source image, validation refusal, transport, continuation, provenance, and local naming into the same finite route, so checkpoint validation is total exactly on I, T, A, V, H, C, P, N. □

Theorem 818.51 (FiniteTraceEvaluator root consumer nonescape exhaustion)**.** *Every root consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N)$$

factors through the carrier, checkpoint source lock, source image readback lock, checkpoint validation totality, and ledger-cut surfaces before it can read a runtime checkpoint result. The exhausted consumer surface is I, T, A, V, H, C, P, N*, not host evaluator validation, hidden runtime acceptance, total truth, total halting, unbounded evaluation, infinite trace semantics, or a second checkpoint route.*

Proof. First build the source-image route with Definition 818.49. It fixes the only image readback rows available to root consumers.

Next use checkpoint validation totality from Definition 818.50. Any validation-facing request remains on the same finite evaluator route and cannot escape to a host judgment.

Finally apply Definition 818.60 together with Definition 818.45. The ledger cut and consumer factorization exhaust runtime checkpoint reads at I, T, A, V, H, C, P, N, so no second route or off-carrier validation surface remains. □

Theorem 818.52 (FiniteTraceEvaluator checkpoint-result totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every accepted checkpoint/result read factors through the displayed finite* BHist *trace* T*, the validation row* V*, the result row* R *classified at* A*, the* Cont *replay routes* C*, componentwise* hsame *transports* H*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*. The read has no carrier coordinate for host validation, total truth, total halting, unbounded evaluation, or an infinite trace.*

Proof. Project the carrier of Definition 818.1. Its readable evaluator surface is the finite packet I, T, A, V, H, C, P, N, and the accepted result row R is exposed only through the classifier row A.

Apply Definition 818.24 and Definition 818.14. They place the result row R behind the displayed finite trace T and make A the only accepted-result classifier available to the checkpoint/result consumer.

Use Definition 818.25 and Definition 818.45. The validation row V, the replay routes C, the componentwise transports H, the provenance row P, and the local naming row N account for the remaining checkpoint boundary.

Finally attach Definition 818.60. Since the root ledger has no row outside I, T, A, V, H, C, P, N, the checkpoint/result read cannot import host validation, total truth, total halting, unbounded evaluation, or an infinite trace. □

Theorem 818.53 (FiniteTraceEvaluator bounded replay stability)**.** *Bounded replay for an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is stable under the displayed componentwise* hsame *transports. Replaying the finite route through* $I \to T \to A$*, the validation boundary* V*, and the continuation rows* C *preserves the checkpoint/result factorization of Definition 818.52 and cannot import host runtime state.*

Proof. Start with Definition 818.40. It locks accepted-result replay to the displayed input, finite trace, and classifier route $I \to T \to A$ before any checkpoint consumer can inspect a result.

The componentwise transport row H carries only the named packet coordinates: input, trace, accepted-result classifier, validation boundary, provenance, and local naming. Transporting along H therefore preserves the same bounded route rather than creating a new runtime source.

Apply Definition 818.42. The runtime checkpoint receives the finite evaluator carrier, validation non-escape, result-validation separation, root trace-result totality, root validation boundary, and root package rows as one handoff.

Thus replay through C remains bounded by I, T, A, V, H, C, P, N. No host runtime state is a coordinate of that packet, so none can be imported by transport or replay. □

Theorem 818.54 (FiniteTraceEvaluator NameCert root ledger factorization)**.** *The root* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *ledger is exhausted by the finite trace T, validation row V, result row R classified at A, checkpoint rows, refusal rows,* Cont *replay routes C,* Pkg *provenance P, and local naming row N. Every root ledger read factors through these displayed rows and no source outside the finite evaluator packet.*

Proof. Begin with Definition 818.3. The local certificate exposes carrier admission, finite trace-step exposure, accepted-result classification, validation non-escape, provenance, and local naming as its obligation rows.

Use Definition 818.52. Accepted checkpoint/result reads pass through T, V, R, H, C, P, N, with R available only as the result row classified at A.

Next apply Definition 818.35 and Definition 818.53. The refusal rows stop host validation, total truth, total halting, and unbounded evaluation at V, H, C, P, N, while bounded replay stays on the displayed transported route.

Finally use Definition 818.44 and Definition 818.60. The provenance and naming rows expose no dependency beyond the finite evaluator packet, so the root ledger is exactly the displayed trace, validation, result, checkpoint, refusal, replay, provenance, and naming surface. □

Theorem 818.55 (FiniteTraceEvaluator checkpoint-state exposure)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *checkpoint-state read first factors through the finite accepted-result route*

$$I \to T \to A$$

and then through the validation boundary, componentwise transport, displayed continuation, provenance, and local naming rows

$$V, H, C, P, N.$$

The state surface exposes result rows and validation rows only through this finite packet and has no coordinate for host validation, total truth, self-validation, total halting, unbounded evaluation, or an infinite trace.

Proof. Project the carrier of Definition 818.1. The available evaluator coordinates are the coded input I, finite trace T, accepted-result classifier A, validation row V, componentwise transports H, continuation row C, provenance row P, and local naming row N.

Apply Definition 818.4 and Definition 818.8. They place every checkpoint-state read behind the finite trace and require the accepted-result part of the read to pass through $I \to T \to A$ before the checkpoint consumer can use it.

Next use Definition 818.30. The root-facing result row is total only for the displayed finite route, so the state exposure cannot move the result coordinate to a host validator or to an unbounded evaluator.

Finally attach Definition 818.32. It adds V, H, C, P, N as the only validation-facing, transport, continuation, provenance, and naming rows available to the root checkpoint packet, excluding host validation, total truth, self-validation, total halting, unbounded evaluation, and infinite traces from the exposed state surface. □

Theorem 818.56 (FiniteTraceEvaluator validation-boundary consumer totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every validation-facing* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer is exhausted by the boundary row V, the transport row H, the continuation row C, the provenance row P, and the local naming row N. Such a consumer cannot export host validation, total truth, self-validation, total halting, unbounded evaluation, an infinite trace, or a second checkpoint-state source.*

Proof. Start from Definition 818.3. The local certificate names the validation non-escape, provenance, and local naming rows as obligations of the finite evaluator packet rather than as host judgments.

Use Definition 818.8. A validation-facing checkpoint consumer receives the same finite carrier route as the accepted-result consumer, but its validation requests are routed to V, H, C, P, N rather than to a host validation predicate.

Apply Definition 818.32. The root boundary package makes V, H, C, P, N the complete consumer-facing validation surface, with no exported row for total truth, self-validation, total halting, unbounded evaluation, infinite trace semantics, or an alternate checkpoint source. □

Theorem 818.57 (FiniteTraceEvaluator root checkpoint NameCert saturation)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the root checkpoint* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *surface is saturated by the finite packet*

$$I, T, A, V, H, C, P, N.$$

Every checkpoint-state, result, validation-boundary, transport, continuation, provenance, and local naming read is supplied by these rows, and the root surface exports no host validation, total truth, self-validation, total halting, unbounded evaluation, infinite trace, or second checkpoint source.

Proof. First combine Definition 818.55 with Definition 818.56. The first theorem exposes the checkpoint-state and result rows through $I \to T \to A$ and then through V, H, C, P, N; the second theorem exhausts the validation-facing consumers at V, H, C, P, N.

Next attach Definition 818.3. Its carrier, trace-step exposure, accepted-result classification, validation non-escape, provenance, and local naming clauses are exactly the local certificate obligations for the displayed finite evaluator packet.

Use Definition 818.30 and Definition 818.32. They identify the root result and validation-boundary rows consumed by checkpoint readers as the same finite root package.

Finally apply Definition 818.4 and Definition 818.8. No read outside I, T, A, V, H, C, P, N is produced by the bounded trace or by the checkpoint consumer route, so the root checkpoint NameCert surface is saturated by the accepted finite packet. □

Definition 818.58 (FiniteTraceEvaluator root exactness surface)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the root exactness surface is the finite packet*

$$E_{\mathrm{root}} = (I, T, A, V, H, C, P, N).$$

Its rows are the carrier of Definition 818.1, the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *obligations of Definition 818.3, the trace-result route of Definition 818.30, the validation-boundary route of Definition 818.31, and the checkpoint-consumer route of Definition 818.8. The surface has no row outside I, T, A, V, H, C, P, N.*

Theorem 818.59 (FiniteTraceEvaluator root exactness scope)**.** *Every root checkpoint read of the surface in Definition 818.58 factors either through the accepted-result route*

$$I \to T \to A$$

or through the validation and ledger route V, H, C, P, N. No root checkpoint read can mix those routes into a host validator, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source.

Proof. Apply Definition 818.30. It places every accepted-result read on the route $I \to T \to A$ before a root consumer can inspect the result classifier.

Next use Definition 818.31. It sends validation-facing reads to V, H, C, P, N and keeps host validation, total truth, total halting, and unbounded evaluation outside the packet.

Finally apply Definition 818.40. The checkpoint source remains locked to the accepted-result route and the validation boundary, so the root read factors through one of the displayed routes and cannot create a second checkpoint source. □

Theorem 818.60 (FiniteTraceEvaluator checkpoint ledger cut)**.** *The checkpoint ledger of the surface in Definition 818.58 has no coordinate for host validation, total truth for* CIC* *sentences, total halting, unbounded evaluation, an infinite trace, or a second checkpoint source. Its complete checkpoint-facing ledger is the displayed packet I, T, A, V, H, C, P, N.*

Proof. Begin with Definition 818.32. The root boundary package exposes accepted results only through $I \to T \to A$ and validation-facing data only through V, H, C, P, N.

Use Definition 818.42. The runtime handoff receives the same finite evaluator rows and no extra coordinate for host validation, total truth, total halting, unbounded evaluation, or an infinite trace.

Combining the two package facts cuts the checkpoint ledger at the displayed surface. A second checkpoint source would require a row not present in I, T, A, V, H, C, P, N, so it is not part of the ledger. □

Lean4 ✓ `BEDC.Derived.FiniteTraceEvaluatorUp.FiniteTraceEvaluatorCheckpointLedgerCut`

Theorem 818.61 (FiniteTraceEvaluator root NameCert obligations)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet with displayed root surface*

$$I, T, A, V, H, C, P, N,$$

the root $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *obligations are exactly the carrier admission row, checkpoint acceptance row, bounded replay row, ledger exactness row, and consumer non-escape row over the displayed* BHist/hsame/Cont/Pkg/NameCert *packet. The certificate admits the coded input row I, finite trace row T, accepted-result row A, validation row V, componentwise transport row H, displayed continuation row C, provenance row P, and local naming row N, and it exports no host evaluator, hidden runtime state, unbounded trace semantics, or unchecked compiler theorem.*

Proof. Carrier admission is the projection of the finite evaluator packet in Definition 818.1. The accepted root surface is the displayed BHist tuple I, T, A, V, H, C, P, N and has no additional coordinate.

Checkpoint acceptance follows from Definition 818.34 and Definition 818.52. They place accepted checkpoint reads on the coded-input, finite-trace, accepted-result route before any root consumer can use the result row.

Bounded replay and ledger exactness are supplied by Definition 818.53 and Definition 818.59. The transport and continuation rows preserve the displayed route and the exactness surface cuts the ledger at I, T, A, V, H, C, P, N.

Finally use Definition 818.35 and Definition 818.60. The validation and ledger rows refuse host evaluation, hidden runtime state, unbounded trace semantics, and unchecked compiler theorems, so the listed rows exhaust the root NameCert obligations. □

Theorem 818.62 (FiniteTraceEvaluator acceptance row stability)**.** *Let two accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *checkpoint reads display the same root rows* I, T, A, V, H, C, P, N *up to the componentwise* hsame *transports in* H *and the displayed* Cont *replay in* C*. Then their accepted checkpoint rows remain on the same root checkpoint, source, and result route*

$$I \to T \to A.$$

No transported read can move acceptance to a host source, a second checkpoint ledger, an infinite trace, or an unbounded evaluator.

Proof. Begin with Definition 818.40. It locks accepted-result reads to the source route $I \to T \to A$ before any validation-facing or ledger-facing row is consumed.

Apply Definition 818.53. The componentwise hsame transport and displayed Cont replay preserve the finite trace and accepted-result rows, so transported checkpoint reads remain on the same bounded evaluator path.

It remains to exclude alternate acceptance sources. Use Definition 818.35 and Definition 818.42. The validation boundary and runtime handoff expose only the displayed finite evaluator rows, so host sources, second checkpoint ledgers, infinite traces, and unbounded evaluators cannot alter the accepted route. □

Theorem 818.63 (FiniteTraceEvaluator ledger non-escape)**.** *The root ledger of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exports exactly the displayed finite evaluator rows*

$$I, T, A, V, H, C, P, N.$$

It exports no host evaluator, hidden runtime state, unbounded trace semantics, unchecked compiler theorem, total truth predicate, total halting classifier, or second checkpoint source.

Proof. Project the root exactness surface of Definition 818.58. Its complete ledger-facing packet is I, T, A, V, H, C, P, N.

Apply Definition 818.54. The local naming row factors the root ledger through carrier admission, checkpoint-result totality, bounded replay, and validation-boundary exactness.

Now use Definition 818.60. The ledger cut removes host validation, total truth, total halting, unbounded evaluation, infinite traces, and second checkpoint sources from the checkpoint-facing surface.

Finally combine this cut with Definition 818.42. Hidden runtime state and unchecked compiler theorems are not rows of the handoff packet, so no root ledger export escapes the displayed finite evaluator surface. □

Theorem 818.64 (FiniteTraceEvaluator route totality)**.** *Every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *read factors through the displayed root packet*

$$I, T, A, V, H, C, P, N.$$

The coded input row I*, finite trace row* T*, accepted-result row* A*, validation row* V*, componentwise transport row* H*, displayed continuation row* C*, package provenance row* P*, and local naming row* N *are therefore the complete route for root consumers, with no host evaluator, infinite trace, or unchecked compiler theorem available as an alternate source.*

Proof. Project the accepted read through the carrier of Definition 818.1. The carrier exposes the finite root rows I, T, A, V, H, C, P, N before any consumer can read the evaluator surface.

Apply the root NameCert obligations of Definition 818.61. They identify carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape as obligations over precisely the displayed packet.

Use acceptance row stability, Definition 818.62, to keep accepted reads on the same $I \to T \to A$ route under hsame transport and displayed Cont replay. Finally apply ledger non-escape, Definition 818.63, which removes host evaluators, infinite traces, and unchecked compiler theorems from the exported row set. □

Theorem 818.65 (FiniteTraceEvaluator checkpoint determinacy)**.** *Let two accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *checkpoint reads agree on the root rows* I, T, A, V, H, C, P, N*, with agreement displayed by the componentwise* hsame *rows in* H *and replayed through the* Cont *routes in* C*. Then the two reads determine the same checkpoint route up to the displayed* hsame *transport and the same* Cont *replay.*

Proof. Read both checkpoint packets through Definition 818.1. There is one coded-input row, one finite-trace row, and one accepted-result row in the displayed root surface, so checkpoint comparison starts on I, T, A.

The root NameCert obligations of Definition 818.61 bind the checkpoint acceptance row to the same carrier admission, bounded replay, ledger exactness, and local naming rows. No additional checkpoint coordinate is available for a second route.

Now apply Definition 818.62. The assumed agreement on H and C is exactly the componentwise transport and displayed replay needed to keep both accepted reads on the same checkpoint route. Ledger non-escape, Definition 818.63, excludes host sources, unbounded traces, and unchecked compiler theorems from changing that route. □

Theorem 818.66 (FiniteTraceEvaluator output non-escape)**.** *The output row of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exports only the displayed finite evaluator route*

$$I, T, A, V, H, C, P, N.$$

It exports no host evaluator, total truth predicate, total halting classifier, infinite trace, unchecked compiler theorem, hidden runtime state, or second checkpoint source.

Proof. Start with Definition 818.1. The output-facing read has no row outside I, T, A, V, H, C, P, N, and the accepted-result coordinate A is reached only through the finite trace route.

Apply Definition 818.61. The local certificate records carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape for the displayed root packet.

Use Definition 818.62 to keep the accepted output on the same checkpoint route under transport and replay. Then apply Definition 818.63: the listed excluded objects are not ledger exports, so they cannot be output rows of the accepted finite evaluator packet. □

Definition 818.67 (FiniteTraceEvaluator BHist step carrier)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the root* BHist *step carrier is the finite evaluator packet*

$$B_{\text{step}} = (I, T, K, R, H, C, P, N),$$

where I *is the source trace row,* T *is the checkpoint row,* K *is the runtime-state row,* R *is the result image,* H *is the componentwise* hsame *transport,* C *is the displayed continuation route,* P *is the package provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. The carrier is read inside the root evaluator surface of Definition 818.1 and the root exactness surface of Definition 818.58.*

Theorem 818.68 (FiniteTraceEvaluator route totality)**.** *Every accepted evaluator read of the step carrier in Definition 818.67 passes through the displayed* BHist/hsame/Cont/Pkg/NameCert *rows*

$$I, T, K, R, H, C, P, N.$$

The source trace, checkpoint, runtime state, result image, componentwise transport, continuation, provenance, and local naming rows therefore form the complete finite route for root evaluator consumers.

Proof. Begin with the checkpoint route of Definition 818.34. It sends checkpoint-facing reads through the displayed finite evaluator rows before any downstream checkpoint consumer can inspect the accepted result.

Next use the runtime surface of Definition 818.37 and the checkpoint source lock of Definition 818.40. These identify the source trace, checkpoint row, runtime-state boundary, and accepted-result image as one finite route inside the root evaluator packet.

Finally attach the root NameCert obligations of Definition 818.61. The local naming row, package provenance, continuation, and componentwise hsame transport are part of the same accepted carrier, so every evaluator read factors through I, T, K, R, H, C, P, N. □

Theorem 818.69 (FiniteTraceEvaluator nonescape boundary)**.** *No accepted read of the step carrier in Definition 818.67 exports host runtime validation, hidden interpreter state, quotient trace equality, or unbounded execution evidence. Every checkpoint or runtime consumer remains on the displayed finite route*

$$I, T, K, R, H, C, P, N.$$

Proof. First apply Definition 818.35. Host validation requests stop at the displayed validation and naming boundary of the accepted evaluator packet and do not become host runtime evidence.

Next use bounded replay ledger exhaustion and checkpoint consumer nonescape, Definition 818.38 and Definition 818.39. They exclude hidden runtime state, infinite trace evidence, unchecked compiler data, and second checkpoint sources from checkpoint-facing reads.

Finally combine route totality with the root exactness scope, Definition 818.68 and Definition 818.59. Since every read has already been forced through I, T, K, R, H, C, P, N, quotient trace equality and unbounded execution evidence have no carrier coordinate and no continuation route from the accepted packet. □

Theorem 818.70 (FiniteTraceEvaluator root image admission)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the root image projected from Definition 818.1 is exactly the displayed finite* BHist *row*

$$E = (I, T, A, V, H, C, P, N).$$

The input row I, finite trace row T, accepted-result row A, validation-boundary row V, componentwise transport row H, continuation row C, package provenance row P, and local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N are admitted as the complete root image. No host evaluator, hidden runtime state, infinite trace, or unchecked compiler theorem is an admitted root-image coordinate.*

Proof. Project the carrier of Definition 818.1. Its finite packet lists exactly I, T, A, V, H, C, P, N, and each listed row has a fixed carrier role.

The existing root image row and root NameCert obligations, Definition 818.46 and Definition 818.61, read the same packet as a BHist/hsame/Cont/Pkg/NameCert surface.

The excluded objects have no carrier coordinate and no continuation route from C. Thus root-image admission is exhausted by the eight displayed rows. □

Theorem 818.71 (FiniteTraceEvaluator root classifier transport)**.** *Let* $E = (I, T, A, V, H, C, P, N)$ *and* $E' = (I', T', A', V', H', C', P', N')$ *be accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packets. If the componentwise* hsame *rows transport* I, T, A, V, C, P, N *to* $I', T', A', V', C', P', N'$*, then the input, finite-trace, accepted-result, and validation route remains the displayed route*

$$I \to T \to A \to V$$

and transports to

$$I' \to T' \to A' \to V'.$$

The transport cannot move the classifier to a host evaluator, detached runtime state, infinite trace, or unchecked compiler theorem.

Proof. First read the source route through Definition 818.70. The only classifier-bearing route starts with the coded input row, proceeds through the finite trace, reaches the accepted-result classifier, and then records the validation boundary.

Next apply acceptance row stability and checkpoint determinacy, Definition 818.62 and Definition 818.65. They keep accepted-result reads on the same $I \to T \to A$ path under componentwise hsame transport and displayed continuation replay.

Finally attach the validation boundary and provenance rows using Definition 818.35 and Definition 818.63. The row V stays a boundary row and the ledger has no alternate classifier coordinate, so the transported packet keeps the displayed route. □

Theorem 818.72 (FiniteTraceEvaluator root ledger totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, every public root consumer is exhausted by the finite ledger*

$$I, T, A, V, H, C, P, N.$$

The ledger exposes the input, finite trace, accepted result, validation boundary, componentwise transport, displayed continuation, package provenance, and local naming rows, and it exposes no host evaluator, hidden runtime state, infinite trace, total truth predicate, total halting classifier, or unchecked compiler theorem.

Proof. Begin with root image admission and root classifier transport. They identify the accepted root image and show that componentwise transport cannot leave the route $I \to T \to A \to V$.

Use route totality and ledger non-escape, Definition 818.64 and Definition 818.63. These theorems account for the consumer-facing rows H, C, P, N and remove host evaluators, infinite traces, and unchecked compiler theorems from the public ledger.

The remaining excluded objects are also absent from the carrier of Definition 818.1. Since no row outside I, T, A, V, H, C, P, N is admitted, every public root consumer is total on that displayed finite ledger. □

Theorem 818.73 (FiniteTraceEvaluator root obligation package)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet*

$$E = (I, T, A, V, H, C, P, N),$$

the root package exposes exactly carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape over the displayed BHist/hsame/Cont/Pkg/NameCert *rows. Its readable surface is the coded input row* I*, finite trace row* T*, accepted-result classifier row* A*, validation row* V*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*; it exports no host evaluator, hidden runtime state, unbounded trace, unchecked compiler theorem, or second root source.*

Proof. Carrier admission is projection from Definition 818.1. The accepted packet is the displayed finite tuple I, T, A, V, H, C, P, N, and the carrier contains no further root coordinate.

Checkpoint acceptance is supplied by Definition 818.34 and Definition 818.52. These force checkpoint-facing reads through the finite input, trace, accepted-result, and validation rows before any consumer can inspect the package.

Bounded replay and ledger exactness come from Definition 818.53 and Definition 818.59. The route remains inside I, T, A, V, H, C, P, N under the displayed hsame transports and Cont replay.

Finally apply Definition 818.35 and Definition 818.60. They exclude host validation, hidden runtime state, unbounded evaluation, infinite traces, and unchecked compiler theorems, so the five listed obligations exhaust the root package. □

Theorem 818.74 (FiniteTraceEvaluator checkpoint route totality)**.** *Every checkpoint-facing consumer route for an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet must expose the same rows*

$$I, T, A, V, H, C, P, N.$$

The route may read the coded input, finite trace, accepted-result classifier, validation boundary, componentwise transports, continuation replay, package provenance, and local naming row only through that packet. It cannot read a host evaluator, hidden runtime state, unbounded trace, unchecked compiler theorem, or second checkpoint source.

Proof. Start with the checkpoint source lock of Definition 818.40. It fixes the checkpoint-facing source route at the displayed input, trace, and accepted-result rows.

Use Definition 818.8. A consumer-facing checkpoint read must pass through the finite evaluator carrier, the validation boundary, the continuation replay, package provenance, and the local naming row.

Finally invoke Definition 818.73. The root package already lists carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape over I, T, A, V, H, C, P, N, so no outside evaluator, hidden state, unbounded trace, unchecked theorem, or second source is available to the route. □

Theorem 818.75 (FiniteTraceEvaluator result factorization non-escape)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, accepted-result classification reaches* A *only through the finite route*

$$I \to T \to A.$$

Validation remains at V*, while the ledger and consumer rows* H, C, P, N *expose only transported replay and local package provenance. The package exports no total truth predicate, total halting classifier, infinite trace, host validator, hidden runtime state, or unchecked compiler theorem.*

Proof. First use Definition 818.4. It places every accepted result behind the coded input row I and finite trace row T before the classifier row A can be read.

Next apply Definition 818.22. The accepted-result ledger is exhausted by the displayed result route, validation boundary, componentwise transport, continuation replay, package provenance, and local naming row.

Use Definition 818.42. The runtime boundary receives the finite evaluator rows as a handoff packet rather than a host validator or hidden runtime state.

Finally combine the handoff with Definition 818.63. The root ledger does not contain total truth, total halting, infinite trace, host validation, or unchecked compiler coordinates, so accepted-result classification cannot escape the route $I \to T \to A$ while validation remains at V. □

Definition 818.76 (FiniteTraceEvaluator terminal root surface)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the terminal root surface is the finite packet*

$$E = (I, T, A, V, H, C, P, N).$$

The row I is the coded input source, T is the terminal finite trace row, A is the accepted-result classifier, V is the validation-boundary refusal row, H records the componentwise hsame *transports, C records the displayed* Cont *replay routes, P records* Pkg *provenance, and N records the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. This surface is the terminal read of the carrier in Definition 818.1, the local NameCert obligations of Definition 818.3, the root exactness scope of Definition 818.59, and the checkpoint ledger cut of Definition 818.60; it is consumed by the runtime checkpoint packet of chapter 997.*

Theorem 818.77 (FiniteTraceEvaluator terminal trace-source lock)**.** *Every terminal root read of the surface $E = (I, T, A, V, H, C, P, N)$ in Definition 818.76 keeps its source on the displayed route*

$$I \to T.$$

No terminal read can replace I by a host source, a second checkpoint source, an infinite trace source, or an unbounded evaluator source.

Proof. Project the carrier in Definition 818.1. It gives the terminal reader exactly one coded input row I and exactly one finite trace row T before any accepted-result or validation-facing row is exposed.

Use Definition 818.3. The trace-step exposure clause binds terminal trace access to the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ row, so every trace read remains a read from I into the displayed finite row T.

Finally apply Definition 818.59 and Definition 818.60. The exactness scope and ledger cut contain no coordinate for a host source, second checkpoint source, infinite trace source, or unbounded evaluator source, and the runtime checkpoint consumer of chapter 997 receives only the displayed terminal packet. □

Theorem 818.78 (FiniteTraceEvaluator terminal accepted-result readback)**.** *For the terminal root surface $E = (I, T, A, V, H, C, P, N)$, an accepted-result read can reach the classifier row A only after the finite trace row T has been read from the coded input source I. The terminal readback route is therefore*

$$I \to T \to A,$$

and no accepted-result read can bypass T through validation, provenance, local naming, host runtime state, total truth, total halting, or an unbounded evaluator.

Proof. Start with the terminal surface of Definition 818.76. Its accepted-result row is the classifier A, while the rows V, H, C, P, N are validation, transport, replay, provenance, and local naming rows rather than result sources.

Apply Definition 818.3. The accepted-result classifier clause exposes A only through the finite trace-step exposure row, so the result read must pass through $I \to T$ before it reaches A.

Now use Definition 818.59. It separates the accepted-result route $I \to T \to A$ from the validation and ledger route V, H, C, P, N, preventing the two routes from being mixed into a hidden readback.

The checkpoint ledger cut in Definition 818.60 removes the remaining bypasses. Host runtime state, total truth, total halting, unbounded evaluation, and a second source are not coordinates of E, and the sibling runtime checkpoint chapter consumes the same finite route. □

Theorem 818.79 (FiniteTraceEvaluator terminal validation nonescape)**.** *In the terminal root surface $E = (I, T, A, V, H, C, P, N)$, the validation row V remains a boundary refusal. It does not become host validation, total truth for* CIC* *sentences, total halting, an unbounded evaluator, an infinite trace, or a second checkpoint source for the runtime checkpoint consumer.*

Proof. Read V from Definition 818.76. The definition names it only as the validation-boundary refusal row of the terminal packet, with transport, replay, provenance, and local naming recorded separately by H, C, P, N.

Use Definition 818.3. Its runtime-validation non-escape clause prevents a local finite evaluator certificate from turning the boundary row into a host validating judgment.

Next apply Definition 818.59. Any terminal validation-facing read factors through V, H, C, P, N rather than through the accepted-result route $I \to T \to A$, so validation cannot become an accepted result or an alternate trace source.

Finally use Definition 818.60. The ledger has no coordinate for host validation, total truth, total halting, unbounded evaluation, infinite traces, or second checkpoint sources. The runtime checkpoint packet of chapter 997 can therefore read V only as the displayed boundary refusal. □

Theorem 818.80 (FiniteTraceEvaluator checkpoint-budget exhaustion)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet $E = (I, T, A, V, H, C, P, N)$, every checkpoint-budget read consumed by the* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *sibling surface is exhausted by the displayed finite evaluator rows: coded input I, finite trace T, accepted-result classifier A, validation-boundary row V, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N. The checkpoint budget exposes no host evaluator, hidden runtime acceptance predicate, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source.*

Proof. Project the carrier of Definition 818.1. Its public BHist packet has exactly the rows I, T, A, V, H, C, P, N; no further runtime row is part of the carrier.

Apply the terminal root surface of Definition 818.76. It presents the same rows to the ReflectionRuntimeCheckpoi consumer, with I, T, A as the accepted-result spine and V, H, C, P, N as the validation-boundary and structural surface.

Finally use the root checkpoint saturation theorem, Definition 818.57. It exhausts checkpoint-state, result, validation-boundary, transport, continuation, provenance, and local naming reads at the same finite packet. Since none of the excluded runtime or truth objects is a displayed row, no checkpoint-budget read can escape the finite surface. □

Theorem 818.81 (FiniteTraceEvaluator reflection-runtime route)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *route reading an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

The first three rows form the accepted-result route, while the remaining rows record validation refusal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. The route does not import a host runtime evaluator or a hidden runtime acceptance predicate.*

Proof. Begin with the checkpoint consumer route, Definition 818.8. It already places every runtime checkpoint consumer on the displayed sequence from I through N.

Next combine the terminal accepted-result readback with terminal validation nonescape, Definition 818.78 and Definition 818.79. The first result fixes access to A behind $I \to T$, and the second keeps validation at the boundary row V rather than at a host runtime predicate.

The structural rows then carry only the same route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row expose the finite packet to the ReflectionRuntimeCheckpoint$^{\uparrow}$ surface without adding evaluator content. □

Theorem 818.82 (FiniteTraceEvaluator downstream-unblock scope)**.** *The finite evaluator package supplied to downstream reflection-runtime chapters is exactly the checkpoint-budget package*

$$I, T, A, V, H, C, P, N.$$

It licenses downstream use of the displayed finite trace, accepted-result row, validation-boundary refusal, BHist *packet identity, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row. It does not license host evaluator validation, hidden runtime acceptance, total truth for* CIC* *sentences, total halting, unbounded evaluation, infinite trace semantics, or a second runtime checkpoint route.*

Proof. First combine Definition 818.80 with Definition 818.81. These two results identify the available budget rows and the only route by which the runtime checkpoint sibling can read them.

Use Definition 818.59 and Definition 818.60. They separate the accepted-result spine from the validation-boundary and structural rows, and they remove host validation, truth, halting, unbounded evaluation, infinite trace semantics, and alternate checkpoint sources from the ledger.

Thus any downstream reflection-runtime chapter citing the evaluator receives a finite BHist packet with visible hsame, Cont, Pkg, and NameCert rows. The cited package unblocks only finite checkpoint reads over those rows, so no hidden runtime acceptance predicate or host evaluator is exported. □

Theorem 818.83 (FiniteTraceEvaluator ground checkpoint readback totality)**.** *For an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, every ground checkpoint readback exposed to a* ReflectionRuntimeCheckpoint$^{\uparrow}$ *consumer is total along the displayed finite rows*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

The readback is a finite BHist *trace read, transported by the componentwise* hsame *row* H*, replayed by the displayed* Cont *route* C*, packaged by* P*, and named by* N*. It has no coordinate for a host evaluator, hidden runtime acceptance predicate, total truth classifier, unbounded runtime, or infinite trace.*

Proof. Project the finite evaluator carrier. Its public checkpoint surface displays the coded input I, finite trace T, accepted-result row A, validation boundary V, componentwise transport H, continuation route C, provenance row P, and local NameCert row N.

Apply checkpoint-budget exhaustion and the reflection-runtime route. These results force every checkpoint readback through the same finite BHist/hsame/Cont/Pkg/NameCert packet and prevent the runtime consumer from asking for any row outside the displayed route.

Finally use checkpoint-source locking. Since the source is already the finite packet I, T, A, V, H, C, P, N, the readback is total exactly on that packet and has no second runtime source from which to import host validation, total truth, unbounded evaluation, or an infinite trace. □

Theorem 818.84 (FiniteTraceEvaluator runtime image route determinacy)**.** *Let* $E = (I, T, A, V, H, C, P, N)$ *be an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet. Every runtime image of* E *visible at the* ReflectionRuntimeCheckpoint$^{\uparrow}$ *surface is determined by the single route*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

Two checkpoint consumers preserving these rows therefore read the same finite trace image, the same accepted-result image, the same validation-boundary image, and the same BHist/hsame/Cont/Pkg/NameCert *exposure. They cannot split the image through a second checkpoint source or a host runtime image.*

Proof. Start from the checkpoint consumer route. It places each runtime-facing consumer on the route from the coded input and finite trace through the accepted-result and validation-boundary rows.

Next attach the structural exposure rows. The componentwise hsame transport row H and displayed Cont replay row C preserve the same finite BHist image, while P and N add only package provenance and local naming.

Use root checkpoint readback totality above to compare two consumers. Since both consumers preserve I, T, A, V, H, C, P, N, their runtime images are read from the same finite packet, so there is no remaining coordinate on which a second checkpoint source or host runtime image could differ. □

Theorem 818.85 (FiniteTraceEvaluator validation ledger exhaustion)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, the validation ledger visible to checkpoint-runtime consumers is exhausted by* V, H, C, P, N *over the finite trace source* I, T, A*. The ledger records validation-boundary refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not export host validation, total truth for runtime sentences, total halting, unbounded evaluation, infinite trace semantics, or a second validation ledger.*

Proof. First separate the accepted-result spine from the validation boundary. The finite trace rows I, T, A supply the source image, while V is the only validation row exposed by the evaluator packet.

Next read the structural ledger. The rows H, C, P, N transport, replay, package, and name the finite validation boundary without adding evaluator content, so every visible validation claim remains inside the displayed BHist/hsame/Cont/Pkg/NameCert packet.

Finally combine runtime image route determinacy with the existing validation nonescape boundary. The runtime image has only the route displayed above, and the nonescape boundary removes host validation, total truth, total halting, unbounded evaluation, infinite traces, and alternate validation ledgers from the checkpoint surface. Thus the validation ledger is exhausted by the stated rows. □

Theorem 818.86 (FiniteTraceEvaluator terminal route totality)**.** *Every terminal route of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *surface*

$$E = (I, T, A, V, H, C, P, N)$$

from Definition 818.76 factors through the accepted-result spine

$$I \to T \to A.$$

The rows V, H, C, P, N *may then supply validation-boundary refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *naming, but no terminal route begins from a host evaluator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, or second checkpoint source.*

Proof. Start with the terminal surface definition. It displays exactly the packet $E = (I, T, A, V, H, C, P, N)$ and identifies I as the coded input row, T as the terminal finite trace row, and A as the accepted-result classifier.

Apply the terminal trace-source lock and terminal accepted-result readback, Definition 818.77 and Definition 818.78. These place the terminal source on $I \to T$ and make access to A pass through the same finite route.

It remains to exclude alternate terminal sources. Use terminal validation nonescape, root exactness scope, and the checkpoint ledger cut, Definition 818.79, Definition 818.59, and Definition 818.60. The sibling runtime checkpoint chapter then consumes the displayed packet rather than a hidden evaluator or second checkpoint source. □

Theorem 818.87 (FiniteTraceEvaluator terminal source stability)**.** *Let two terminal reads of accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *surfaces display the same packet*

$$E = (I, T, A, V, H, C, P, N)$$

up to the componentwise hsame *transports recorded in* H *and the displayed* Cont *replay routes recorded in* C*. Then both reads preserve the same terminal source route*

$$I \to T \to A.$$

Transport may replay validation, provenance, and naming access through V, H, C, P, N*, but it cannot move the terminal source to a host evaluator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, or second checkpoint source.*

Proof. First expose the common terminal packet through Definition 818.76. The only source, trace, and accepted-result rows of the packet are I, T, and A.

Use Definition 818.86. Each terminal read already factors through $I \to T \to A$ before the validation, transport, replay, provenance, and naming rows are consumed.

Now apply the bounded replay stability and acceptance row stability results, Definition 818.53 and Definition 818.62. Componentwise hsame transport and displayed Cont replay preserve the finite trace row and the accepted-result row of the same route.

Finally combine terminal validation nonescape with the checkpoint ledger cut, Definition 818.79 and Definition 818.60. Since the ledger contains no alternate source coordinate, transport cannot create one for the runtime checkpoint consumer. □

Theorem 818.88 (FiniteTraceEvaluator terminal ledger exhaustion)**.** *The terminal ledger exported by an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *surface is exhausted by the displayed packet*

$$E = (I, T, A, V, H, C, P, N).$$

It exports no host evaluator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, or second checkpoint source. Every downstream runtime checkpoint read therefore receives the terminal route $I \to T \to A$ *together with the boundary, transport, continuation, provenance, and naming rows* V, H, C, P, N*, and receives no further evaluator coordinate.*

Proof. Begin with Definition 818.76. The terminal ledger has exactly the eight displayed rows of E.

Apply terminal route totality and terminal source stability, Definition 818.86 and Definition 818.87. The accepted-result side is fixed at $I \to T \to A$, and componentwise transport keeps terminal reads on that route.

Use root exactness scope and checkpoint ledger cut, Definition 818.59 and Definition 818.60. These results separate the accepted-result spine from V, H, C, P, N and remove host validation,

total truth, total halting, unbounded evaluation, infinite traces, and second checkpoint sources from the ledger.

Finally read the handoff through the runtime checkpoint sibling chapter, chapter 997. Its root carrier and ledger exactness rows consume the finite checkpoint packet without importing hidden runtime state or an additional evaluator coordinate. □

Theorem 818.89 (FiniteTraceEvaluator terminal checkpoint accountability)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal packet*

$$I, T, A, V, H, C, P, N,$$

the checkpoint obligation is accountable to the displayed finite BHist *route*

$$I \to T \to A.$$

The accepted-result row is read only through Definition 818.78, the terminal route is total by Definition 818.86, and the root route is total by Definition 818.64. The remaining rows V, H, C, P, N *provide validation boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *naming. No host evaluator, hidden runtime state, infinite trace, or unchecked compiler theorem is a checkpoint source.*

Proof. Begin with the terminal root surface of Definition 818.76. It displays exactly the rows I, T, A, V, H, C, P, N, with T as the finite BHist trace row and A as the accepted-result classifier.

Apply terminal accepted-result readback, Definition 818.78. The result side cannot reach A before reading the finite route $I \to T$, so the checkpoint is accountable to the same terminal trace.

Finally apply terminal route totality and root route totality, Definition 818.86 and Definition 818.64. The rows V, H, C, P, N may transport, replay, package, and name the checkpoint surface, but they do not introduce a host evaluator, hidden runtime state, infinite trace, or unchecked compiler theorem. □

Theorem 818.90 (FiniteTraceEvaluator terminal validation route)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal packet*

$$I, T, A, V, H, C, P, N,$$

every validation-facing checkpoint read factors through the finite rows V, H, C, P, N *over the same* BHist *trace route* $I \to T \to A$*. The validation boundary is the row* V *of Definition 818.79; its transports, continuations, package provenance, and local naming are exactly the displayed* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *rows* H, C, P, N*. It cannot become host validation, a hidden runtime evaluator, an infinite trace checker, or an unchecked compiler theorem.*

Proof. Read the validation row from Definition 818.79. It is a boundary refusal row in the terminal packet, not a host-side validator or a second source of accepted results.

Use terminal route totality, Definition 818.86, to keep the validation-facing read on the same finite packet. The accepted-result spine is $I \to T \to A$, while V, H, C, P, N record boundary, transport, replay, provenance, and naming for that spine.

The checkpoint accountability statement, Definition 818.89, identifies the only checkpoint source. Since the packet has no coordinate for hidden runtime state, infinite traces, or compiler-side unchecked proof, validation cannot escape the displayed route. □

Theorem 818.91 (FiniteTraceEvaluator terminal consumer non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal checkpoint packet receives only*

$$I, T, A, V, H, C, P, N.$$

Its accepted-result access factors through $I \to T \to A$ *by Definition 818.78 and Definition 818.86; its validation-facing access factors through* V, H, C, P, N *by Definition 818.79 and Definition 818.90. Thus every consumer route is a displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *route, with no host evaluator, hidden runtime, infinite trace, or unchecked compiler theorem.*

Proof. First expose the terminal packet through Definition 818.76. The consumer receives the eight displayed rows and no off-packet evaluator coordinate.

For accepted-result reads, apply terminal accepted-result readback and terminal route totality. These results bind A to the finite route $I \to T \to A$ and keep all terminal reads inside the same packet.

For validation-facing reads, apply terminal validation nonescape and the terminal validation route theorem. The rows V, H, C, P, N account for boundary refusal, transport, continuation, provenance, and naming without adding host validation or an infinite-trace checker.

Combining the two routes leaves no remaining consumer channel. Any claimed host evaluator, hidden runtime, infinite trace, or unchecked compiler theorem would require a packet coordinate not present in I, T, A, V, H, C, P, N. □

Theorem 818.92 (FiniteTraceEvaluator checkpoint-runtime triad)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the checkpoint row, runtime row, and image/readback row form one displayed triad*

$$I, T, A, V, H, C, P, N.$$

The checkpoint side reads the finite BHist *trace through* T *and the accepted-result route through* $I \to T \to A$*; the runtime side reads the same route through* hsame *transports* H *and displayed* Cont *replay routes* C*; the package and local naming side is recorded by* Pkg *provenance* P *and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*.*

No checkpoint-runtime consumer receives a host evaluator, hidden runtime state, unbounded trace source, second checkpoint source, or unchecked compiler theorem.

Proof. Begin with checkpoint readiness. The totality and readiness rows of Definition 818.34 and Definition 818.36 place every root checkpoint read on I, T, A, V, H, C, P, N before any downstream runtime consumer can inspect the packet.

Next use runtime totality. By Definition 818.37, the runtime row has the same finite surface and does not add a host evaluator or an unbounded runtime coordinate.

Finally attach image readback, result factorization, and terminal route completion. The image and readback rows of Definition 818.46 and Definition 818.47 identify the visible rows, while Definition 818.52 and Definition 818.86 keep the accepted result on the same finite route. Thus the checkpoint, runtime, and readback rows form one triad. □

Theorem 818.93 (FiniteTraceEvaluator checkpoint image readback)**.** *Every checkpoint image of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet reads back through the finite route*

$$I \to T \to A$$

for accepted results and through V, H, C, P, N *for validation-facing reads. The readback uses the displayed* BHist *trace, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row, and it exposes no host validation, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source.*

Proof. First expose the checkpoint image. Apply Definition 818.46; it lists exactly I, T, A, V, H, C, P, N as the image rows visible to accepted reads.

Then read back the accepted-result and validation routes. The exactness result Definition 818.47 sends accepted results through $I \to T \to A$ and validation-facing reads through V, H, C, P, N.

Finally combine result factorization with terminal route completion. Definition 818.48 and Definition 818.88 show that the downstream runtime checkpoint consumer receives the same finite packet. Since the ledger has no host or unbounded coordinate, checkpoint readback cannot escape the displayed image. □

Theorem 818.94 (FiniteTraceEvaluator terminal runtime nonescape)**.** *The terminal runtime route of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is confined to the finite checkpoint-runtime packet*

$$I, T, A, V, H, C, P, N.$$

Its BHist *trace row,* hsame *transport rows,* Cont *replay rows,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row are the only routes available to terminal runtime consumers. It exports no host evaluator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, infinite trace semantics, or alternate checkpoint route.*

Proof. Start from the checkpoint-runtime triad of Definition 818.92. It identifies the checkpoint, runtime, and image/readback rows as one finite packet.

Next apply the terminal route results. The terminal route totality and source stability rows, Definition 818.86 and Definition 818.87, keep terminal runtime reads on $I \to T \to A$ under displayed hsame and Cont transport.

Finally use terminal ledger exhaustion together with checkpoint image readback. By Definition 818.88 and Definition 818.93, the terminal consumer receives only I, T, A, V, H, C, P, N. Hence none of the excluded host, truth, halting, unbounded, infinite-trace, or alternate checkpoint coordinates can be exported. □

Theorem 818.95 (FiniteTraceEvaluator checkpoint-runtime acceptance lock)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, a checkpoint-facing read is accepted only when it passes through the displayed input row, finite trace row, accepted-result row, validation-boundary row, componentwise transport row, route row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. In packet notation the accepted surface is*

$$I, T, A, V, H, C, P, N.$$

No checkpoint read is accepted from a detached runtime input, host validation row, unbounded trace source, alternate accepted-result classifier, or local name outside that packet.

Proof. Begin with the carrier surface of the finite trace evaluator. Its checkpoint-facing rows are the input, trace, accepted-result, validation, transport, route, provenance, and local naming rows.

Apply the checkpoint-runtime triad of Definition 818.92. It identifies the checkpoint, runtime, and readback sides as one finite packet I, T, A, V, H, C, P, N.

Use checkpoint image readback and terminal runtime nonescape, Definition 818.93 and Definition 818.94. These results rule out detached host validation, unbounded trace semantics, alternate accepted-result classifiers, and local names outside the displayed packet. □

Theorem 818.96 (FiniteTraceEvaluator runtime route coverage)**.** *Every runtime-facing consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *and* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *before reading the evaluator result. The result read is therefore carried only by the finite route* I, T, A, V, H, C, P, N *and by the candidate-boundary and checkpoint rows named in the provenance packet.*

Proof. First expose the runtime checkpoint side. The checkpoint-runtime triad places runtime-facing reads on the same finite evaluator rows as checkpoint-facing reads.

Next route the consumer through the checkpoint sibling. The $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ side supplies the checkpoint state and validation-boundary read before any evaluator result can be consumed.

Finally use the $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ provenance row. It supplies the object-level input, transition, and accepted-result boundary read by the evaluator. Since the evaluator result is reached only after those two sibling surfaces, no runtime consumer bypasses them. □

Theorem 818.97 (FiniteTraceEvaluator validation refusal closure)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, host validators, truth predicates, unrestricted evaluator semantics, and hidden runtime state are refused outside the displayed packet*

$$I, T, A, V, H, C, P, N.$$

The validation-facing row records only the finite boundary available to the packet; it does not export host-valid runtime judgment, total truth, unbounded evaluation, or hidden machine state.

Proof. Start with the acceptance lock above. Accepted checkpoint-runtime reads must pass through I, T, A, V, H, C, P, N.

Apply runtime route coverage. Any runtime-facing consumer reaches the evaluator only after $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ and $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$, so the consumer cannot replace the displayed validation row by host validation or by a total truth predicate.

Finally use terminal runtime nonescape. The terminal route exports no host evaluator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, infinite trace semantics, or alternate checkpoint route. The validation refusal is therefore closed at the displayed finite packet. □

Theorem 818.98 (FiniteTraceEvaluator checkpoint-runtime route determinacy)**.** *Let two accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *checkpoint-runtime packets display rows*

$$I, T, A, V, H, C, P, N$$

and

$$I', T', A', V', H', C', P', N'.$$

If the input, finite trace, accepted-result, validation-boundary, transport, route, provenance, and local naming rows compare componentwise by hsame*, and the displayed* Cont *routes identify the checkpoint replay from* I *to* T *and from* T *to* A*, then the two runtime reads have the same public checkpoint route. The determinacy is read only through the* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *carrier and the root* $\mathsf{NameCert}$ *obligation surface.*

Proof. Start with the checkpoint-runtime triad. By Definition 818.92, each accepted packet presents checkpoint, runtime, image, and readback rows as the single finite surface I, T, A, V, H, C, P, N.

Next apply the acceptance lock and the root obligation surface. The lock of Definition 818.95 forces both reads to pass through the same input, trace, accepted-result, validation-boundary, transport, route, provenance, and local naming rows, while Definition 818.61 supplies the carrier and classifier rows that make componentwise hsame transport valid.

Finally use the displayed Cont hypotheses. Since the replay routes from input to trace and from trace to accepted result are matched componentwise, no second checkpoint route remains. The public runtime read is therefore determined by the displayed packet and its local NameCert surface. □

Theorem 818.99 (FiniteTraceEvaluator runtime acceptance nonescape)**.** *Every runtime acceptance read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is confined to the displayed checkpoint-runtime rows*

$$I, T, A, V, H, C, P, N.$$

No host validator, total truth predicate, total halting classifier, unchecked compiler theorem, hidden runtime state, unbounded evaluator, or infinite trace can be exported from the runtime acceptance row.

Proof. First read the accepted packet through the carrier and root NameCert obligation surface. The rows I, T, A, V, H, C, P, N are exactly the input, finite trace, accepted-result, validation-boundary, componentwise transport, replay route, provenance, and local naming rows.

Use Definition 818.94 for the terminal runtime side. It already excludes host evaluators, hidden runtime state, total truth predicates, total halting classifiers, unbounded evaluator semantics, infinite traces, and alternate checkpoint routes.

Now apply Definition 818.97. The validation-facing row is a finite boundary inside the accepted packet, not a host-valid runtime judgment or a global compiler theorem. Thus the runtime acceptance read cannot escape the displayed checkpoint-runtime surface. □

Theorem 818.100 (FiniteTraceEvaluator checkpoint replay totality)**.** *Every checkpoint replay of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the displayed rows*

$$I, T, A, V, H, C, P, N.$$

The replay first reads the input row I, then the finite trace row T, then the accepted-result row A, and then the validation-boundary row V together with the componentwise transport H, displayed route C, provenance P, and local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N. This totality assertion uses only the carrier and root obligation surface, and it introduces no infinite trace or unchecked host evaluator.*

Proof. Begin with checkpoint image readback. By Definition 818.93, accepted checkpoint images read back through $I \to T \to A$ and then through V, H, C, P, N for validation-facing reads.

Next apply checkpoint-runtime acceptance lock. It requires every accepted checkpoint-facing read to use the same input, trace, accepted-result, validation-boundary, transport, route, provenance, and local naming rows before runtime acceptance is available.

Finally combine route determinacy with runtime nonescape. The determinacy theorem above makes the replay route unique under componentwise hsame and displayed Cont transport, while Definition 818.99 excludes host validators, total truth predicates, total halting classifiers, unchecked compiler theorems, and infinite traces. Hence every checkpoint replay is total on the displayed finite route. □

Theorem 818.101 (FiniteTraceEvaluator checkpoint runtime NameCert triad)**.** *Every checkpoint consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet reads the displayed rows*

$$I, T, A, V, H, C, P, N$$

through bounded trace coverage, the accepted-result classifier, validation-boundary refusal, componentwise transport, continuation replay, provenance, and the local $\mathsf{NameCert}$ *rows before reaching* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$*. The triad exports no host validator, truth predicate, total halting predicate, infinite trace, or self-validation row.*

Proof. Start from the carrier of Definition 818.1 and the root NameCert obligations of Definition 818.61. They identify I, T, A, V, H, C, P, N as the finite evaluator packet and bind those rows to the local certificate surface.

Next use bounded trace coverage and validation refusal from Definition 818.4 and Definition 818.35. The finite trace row and accepted-result classifier are visible, while host validation and unbounded trace semantics remain outside the packet.

Finally apply result-validation separation, route obligation triad, and checkpoint consumer route, Definition 818.6, Definition 818.7, and Definition 818.81. These force checkpoint reads through validation-boundary refusal, componentwise transport, continuation replay, provenance, and local naming before the $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ handoff. □

Theorem 818.102 (FiniteTraceEvaluator checkpoint state saturation route)**.** *Every checkpoint-state read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is saturated by the route*

$$I, T, A, V, H, C, P, N$$

before a runtime checkpoint consumer inspects it. The saturated route reads bounded trace coverage, accepted-result classification, validation-boundary refusal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row, and it exports no host validator, self-validation row, total halting predicate, infinite trace, or detached runtime state.*

Proof. First apply Definition 818.101. It places checkpoint consumers on the displayed packet rows and sends the read toward $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ through the finite NameCert triad.

Next use the checkpoint-runtime triad and acceptance lock, Definition 818.92 and Definition 818.95. They keep accepted checkpoint-state reads on the bounded trace, classifier, validation-boundary, transport, continuation, provenance, and local naming rows.

Finally combine checkpoint budget exhaustion with route determinacy, Definition 818.80 and Definition 818.98. The checkpoint budget and runtime route are determined by the same finite packet, so no detached runtime state or external validation row remains to be exported. □

Theorem 818.103 (FiniteTraceEvaluator runtime admission boundary)**.** *Every runtime admission read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet must enter through*

$$I, T, A, V, H, C, P, N.$$

The boundary factors through bounded trace coverage, the accepted-result classifier, validation-boundary refusal, componentwise transport, continuation replay, provenance, local NameCert rows, and then $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$*. It does not export a host validator, truth predicate, total halting predicate, infinite trace, or self-validation row.*

Proof. Begin with root runtime admission and checkpoint route coverage, Definition 818.107 and Definition 818.108. These route runtime-facing and checkpoint-facing consumers through the same finite packet I, T, A, V, H, C, P, N.

Next apply the checkpoint state saturation route above. It keeps the checkpoint-state read saturated by bounded trace coverage, accepted-result classification, validation-boundary refusal, componentwise transport, continuation replay, provenance, and local naming before runtime inspection.

Finally use runtime ledger non-escape, Definition 818.109. That theorem excludes host evaluator state, hidden runtime acceptance predicates, total truth predicates, total halting classifiers, unbounded evaluation, infinite traces, and second checkpoint sources. Hence runtime admission is exactly the displayed boundary before the $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ consumer reads it. □

Theorem 818.104 (FiniteTraceEvaluator checkpoint row exposure)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, the checkpoint-facing surface exposes exactly the input row* I*, finite trace row* T*, accepted-result row* A*, validation-boundary row* V*, componentwise transport row* H*, route row* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*. The exposure is read from the finite carrier and exports no hidden runtime state.*

Proof. Project the carrier of Definition 818.1. Its checkpoint-facing coordinates are the input, trace, accepted-result, validation, transport, route, provenance, and local naming rows.

Apply the root NameCert obligation surface of Definition 818.61. It identifies those coordinates as the public certificate rows of the accepted evaluator packet.

The checkpoint-runtime triad of Definition 818.92 then reads the checkpoint side through the same finite surface. Since the carrier has no separate runtime coordinate, the checkpoint row exposes only I, T, A, V, H, C, P, N. □

Theorem 818.105 (FiniteTraceEvaluator runtime route totality)**.** *Every runtime-facing read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the checkpoint route*

$$I \to T \to A \to V$$

with the transport, route, provenance, and local naming rows H, C, P, N *visible. The route is total for the displayed packet and does not require a host evaluator, unchecked validation predicate, unbounded trace, or alternate accepted-result classifier.*

Proof. Begin with checkpoint row exposure from Definition 818.104. It supplies the finite packet I, T, A, V, H, C, P, N as the whole runtime-facing surface.

Next use Definition 818.95. It requires every accepted checkpoint-runtime read to pass through the displayed input, trace, accepted-result, validation, transport, route, provenance, and local naming rows.

Finally apply Definition 818.100. The replay reads I before T, T before A, and then V, H, C, P, N, so the runtime route is total on the displayed finite packet and no host evaluator or alternate classifier is imported. □

Theorem 818.106 (FiniteTraceEvaluator checkpoint route nonescape)**.** *No checkpoint-facing consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet can escape the displayed route* I, T, A, V, H, C, P, N*. In particular, the consumer cannot replace the input row, finite trace row, accepted-result row, validation row, transport row, route row, provenance row, or local name by hidden runtime state, a host validation coordinate, an unbounded evaluator output, or an alternate evaluator result.*

Proof. Start with runtime route totality from Definition 818.105. It places every runtime-facing read on the finite checkpoint route $I \to T \to A \to V$ with H, C, P, N visible.

Use checkpoint-runtime route determinacy from Definition 818.98. Componentwise hsame and displayed Cont replay identify the route by the exposed rows, so a second checkpoint route cannot be recovered from the same packet.

Now apply runtime acceptance nonescape from Definition 818.99. The accepted runtime row exports no host validator, total truth predicate, total halting classifier, unchecked compiler theorem, hidden runtime state, unbounded evaluator, or infinite trace.

Combining these facts, every checkpoint consumer is confined to the displayed input, trace, accepted-result, validation, transport, route, provenance, and local naming rows, and no alternate evaluator output is available. □

Theorem 818.107 (FiniteTraceEvaluator root runtime admission)**.** *Every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet enters the root runtime surface through the displayed rows*

$$I, T, A, V, H, C, P, N.$$

The coded input row I, finite trace row T, accepted-result classifier row A, validation boundary row V, componentwise transport row H, displayed continuation row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N are the complete runtime admission rows for the packet.*

Proof. Begin with the carrier of Definition 818.1. The finite evaluator packet exposes the coded input, finite trace, accepted result, validation boundary, transport, continuation, provenance, and local naming rows as its root runtime surface.

Apply Definition 818.61. The root NameCert obligations identify carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape over exactly the displayed rows I, T, A, V, H, C, P, N.

Use Definition 818.80 and Definition 818.81. They keep checkpoint-budget reads and reflection-runtime reads on the same finite evaluator route, so runtime admission has no row outside the displayed packet. □

Theorem 818.108 (FiniteTraceEvaluator checkpoint route coverage)**.** *Every checkpoint-facing consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet factors through the coded input row, finite trace row, accepted-result classifier, validation boundary, componentwise transport, displayed continuation, provenance row, and local NameCert row*

$$I, T, A, V, H, C, P, N.$$

Thus checkpoint reads are covered by the root runtime route before any downstream result is consumed.

Proof. First apply Definition 818.107. It fixes the root runtime rows I, T, A, V, H, C, P, N as the entry surface for accepted evaluator packets.

Next use Definition 818.95. Checkpoint-facing reads are accepted only through the displayed input, finite trace, accepted-result, validation, transport, route, provenance, and local naming rows.

Apply Definition 818.80. The checkpoint budget is exhausted by the same finite route, and the result row cannot be read through a detached checkpoint source.

Finally use Definition 818.82. Downstream checkpoint consumers receive the accepted finite evaluator route through the same packet, so route coverage is shared by checkpoint and runtime-facing consumers. □

Theorem 818.109 (FiniteTraceEvaluator runtime ledger non-escape)**.** *The runtime ledger of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exports only the displayed evaluator rows*

$$I, T, A, V, H, C, P, N.$$

It exports no host evaluator, hidden runtime acceptance predicate, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source.

Proof. Start from Definition 818.107. Runtime admission is already restricted to the displayed finite evaluator rows.

Apply Definition 818.108. Every checkpoint-facing consumer factors through the same input, trace, accepted-result, validation, transport, continuation, provenance, and local naming route, so there is no second checkpoint source for the ledger to export.

Use Definition 818.96 and Definition 818.97. Runtime-facing reads pass through the reflection-runtime and candidate-boundary surfaces, while host validation, total truth, total halting, unbounded evaluation, and infinite traces are refused outside the displayed packet.

Finally apply Definition 818.63. The existing ledger non-escape row removes host evaluator state, unbounded trace semantics, unchecked compiler theorems, and second checkpoint sources from the exported root ledger; the runtime ledger therefore remains exactly I, T, A, V, H, C, P, N. □

Theorem 818.110 (FiniteTraceEvaluator root checkpoint admission totality)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet with carrier* $E = (I, T, A, V, H, C, P, N)$*, checkpoint admission is total only along the displayed finite route*

$$I \to T \to A \to V \to H \to C \to P \to N.$$

The admission surface is a finite BHist *read transported by* hsame*, replayed by* Cont*, packaged by* Pkg*, and named by* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$*. It has no coordinate for host validation, total truth, total halting, unbounded evaluation, or an infinite trace.*

Proof. Project the carrier in Definition 818.1. The accepted evaluator packet displays exactly I, T, A, V, H, C, P, N, with T as the finite BHist trace row, H as the componentwise hsame transport row, C as the Cont replay row, P as Pkg provenance, and N as the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ row.

Apply the local NameCert obligations and bounded trace coverage, Definition 818.3 and Definition 818.4. They force every accepted-result read through $I \to T \to A$ before checkpoint admission can read validation, transport, continuation, provenance, or naming rows.

Finally attach the host-validation refusal ledger and root runtime admission surface, Definition 818.5 and Definition 818.107. The refusal ledger blocks host validation, total truth, total halting, unbounded evaluation, and infinite traces, while the runtime admission surface keeps the consumer on the same finite route through V, H, C, P, N. □

Theorem 818.111 (FiniteTraceEvaluator root trace-validation separation)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet* $E = (I, T, A, V, H, C, P, N)$*, accepted-result reads and validation-boundary reads remain separated at the rows* A *and* V*. The accepted-result side is read only through*

$$I \to T \to A,$$

while the validation-boundary side is read only through

$$V \to H \to C \to P \to N.$$

The two sides may be transported, replayed, packaged, and named inside the same $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCer}$ *packet, but neither side becomes host validation or an infinite trace checker.*

Proof. Start from result-validation separation, Definition 818.6. It places accepted results behind the finite route $I \to T \to A$ and records validation only at the boundary row V.

Use the checkpoint source lock and validation nonescape totality, Definition 818.40 and Definition 818.41. They keep checkpoint-facing accepted-result reads on A and validation-facing reads on V, H, C, P, N under the displayed hsame transports and Cont replay routes.

The local NameCert obligations then exhaust the packet. Since P and N add only provenance and local naming, the separated reads cannot import host validation, total truth, total halting, unbounded evaluation, or infinite trace semantics. □

Theorem 818.112 (FiniteTraceEvaluator root reflection checkpoint handoff)**.** *Every* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *handoff that consumes an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet consumes the finite evaluator rows* I, T, A, V, H, C, P, N*. The handoff reads bounded trace coverage, accepted-result classification, validation-boundary refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row, and it cannot import host validation, total truth, total halting, an unbounded runtime, or an infinite trace.*

Proof. First apply the runtime boundary handoff, Definition 818.42. It sends every $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ handoff through the finite evaluator carrier, bounded trace coverage, validation non-escape, result-validation separation, root trace-result totality, root validation boundary, and root boundary package.

Next use checkpoint admission totality above. The runtime admission surface is already the displayed route $I \to T \to A \to V \to H \to C \to P \to N$, so the handoff receives no extra host row before reading the evaluator packet.

Finally combine root consumer readiness with runtime admission boundary, Definition 818.36 and Definition 818.103. Downstream checkpoint consumers receive the same finite BHist/hsame/Cont/Pkg/NameCe packet, and the refusal surface blocks host validation, total truth, total halting, unbounded runtime state, and infinite trace imports. □

Definition 818.113 (FiniteTraceEvaluator terminal route carrier)**.** *A* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal route carrier is a finite packet*

$$E = (T, Q, R, X, H, C, P, N)$$

whose rows have the following roles. The row T is the finite BHist *trace row of the evaluator. The row Q is the checkpoint row that reads the trace as a checkpoint source. The row R is the runtime row exposed to downstream runtime consumers. The row X is the image and readback row recording the accepted-result image and the checkpoint readback boundary. The row H records componentwise* hsame *transport between the trace, checkpoint, runtime, and readback rows. The row C is the displayed* Cont *replay route. The row P is the* Pkg *provenance row. The row N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. The carrier has no coordinate for host validation, hidden runtime state, unbounded evaluation, an infinite trace, or an unchecked compiler theorem.*

Theorem 818.114 (FiniteTraceEvaluator checkpoint-runtime factorization)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal route carrier $E = (T, Q, R, X, H, C, P, N)$, checkpoint-facing and runtime-facing reads factor through the same finite route*

$$T \to Q \to R.$$

The factorization is transported componentwise by H, replayed by C, packaged by P, named by N, and read back through X. It imports no host evaluator, no hidden runtime row, and no second checkpoint source.

Proof. First expose the checkpoint side with Definition 818.34 and Definition 818.110. These statements place checkpoint admission on the finite evaluator rows and force the checkpoint source to be a displayed BHist route.

Next expose the runtime side with Definition 818.37 and Definition 818.107. Runtime consumers read the same finite packet through displayed transport, continuation, provenance, and local naming rows, not through a host evaluator.

Finally attach the checkpoint-runtime surface of Definition 818.104 and Definition 818.105. It identifies the checkpoint row and the runtime row as two views of the same finite route, so the carrier factors through $T \to Q \to R$ with X, H, C, P, N as readback, transport, replay, provenance, and naming data. □

Theorem 818.115 (FiniteTraceEvaluator terminal route exposure)**.** *Every terminal consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *route carrier receives exactly the finite surface*

$$T, Q, R, X, H, C, P, N.$$

The trace row T, checkpoint row Q, runtime row R, image and readback row X, componentwise hsame *row H,* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N are the whole terminal exposure. The consumer cannot recover host validation, hidden runtime state, unbounded evaluation, an infinite trace, or an unchecked compiler theorem from the packet.*

Proof. Start from the terminal checkpoint surface, Definition 818.89. It binds terminal checkpoint reads to the finite trace route and keeps validation, transport, continuation, provenance, and naming inside the displayed packet.

Use the checkpoint-runtime factorization above. It aligns the checkpoint row with the runtime row through the same finite carrier and leaves image readback as the row X rather than as a hidden runtime coordinate.

Now apply root image readback, Definition 818.46 and Definition 818.47. Accepted reads and checkpoint readback are confined to the displayed image and readback row, so the terminal consumer sees precisely T, Q, R, X, H, C, P, N. □

Theorem 818.116 (FiniteTraceEvaluator terminal runtime-candidate bridge)**.** *Every terminal consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *route carrier $E = (T, Q, R, X, H, C, P, N)$ that asks for runtime-candidate provenance first factors through* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *and* $\mathsf{ReflectionRuntimeCheckpoin}$ *and only then through the displayed terminal rows*

$$T, Q, R, X, H, C, P, N.$$

The bridge introduces no host validator, hidden runtime state, total truth predicate, total halting classifier, unbounded evaluator, infinite trace semantics, or alternate checkpoint route.

Proof. Start with terminal route exposure. It confines every terminal consumer of the accepted route carrier to T, Q, R, X, H, C, P, N, with T as trace row, Q as checkpoint row, R as runtime row, and X as image and readback row.

Use checkpoint-runtime factorization. Checkpoint-facing and runtime-facing reads factor through the same finite route $T \to Q \to R$, transported by H, replayed by C, packaged by P, named by N, and read back through X.

Now apply checkpoint-runtime acceptance lock and runtime route coverage. Those results force runtime-facing consumers through $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ and $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ before the evaluator result is read, while keeping host validation and hidden runtime state outside the finite packet.

Finally combine this with validation refusal closure from the checkpoint-runtime acceptance file. Since the runtime-candidate and checkpoint surfaces are reached only through the displayed terminal route, the bridge cannot introduce total truth, total halting, unbounded evaluator semantics, infinite traces, or an alternate checkpoint route. □

Theorem 818.117 (FiniteTraceEvaluator NameCert terminal obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvalu}}$ *obligation surface of a terminal route carrier* $E = (T, Q, R, X, H, C, P, N)$ *is exhausted by the following finite obligations: trace finiteness for* T*, checkpoint admission for* Q*, runtime totality for* R*, image and readback exactness for* X*, componentwise* hsame *transport for* H*, displayed* Cont *replay for* C*,* Pkg *provenance for* P*, and local naming for* N*. No NameCert clause asks for host validation, total truth, total halting, unbounded evaluation, an infinite trace, or an unchecked compiler theorem.*

Proof. First read the local naming row from Definition 818.3. The carrier rows are exactly the rows that the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ surface can certify.

Next combine root checkpoint readiness, root runtime totality, and root image readback exactness. Theorems Definition 818.36, Definition 818.37, and Definition 818.48 provide the checkpoint, runtime, image, and readback obligations on the same finite packet.

Finally use terminal route exposure above together with Definition 818.106. Since every terminal consumer is confined to T, Q, R, X, H, C, P, N, the local NameCert terminal surface is exhausted by those displayed obligations and has no remaining host or unbounded evaluator clause. □

Theorem 818.118 (FiniteTraceEvaluator terminal runtime checkpoint lock)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal checkpoint packet*

$$E = (I, T, A, V, H, C, P, N),$$

every terminal checkpoint read remains on the displayed route

$$I \to T \to A \to V \to H \to C \to P \to N.$$

Here T *is the finite* BHist *trace row,* H *is the componentwise* hsame *transport row,* C *is the displayed* Cont *replay row,* P *is the* Pkg *provenance row, and* N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row. The lock exposes no host evaluator, hidden acceptance predicate, total truth predicate, unbounded runtime, or infinite trace semantics.*

Proof. Begin with the carrier and terminal root surface, Definition 818.1 and Definition 818.76. They display exactly I, T, A, V, H, C, P, N as the terminal checkpoint packet and provide no coordinate for a host evaluator or infinite trace.

Apply checkpoint-budget exhaustion, Definition 818.80. It forces every checkpoint-budget read consumed by the reflection-runtime sibling to be exhausted by the same finite evaluator rows.

Finally use terminal route exposure and root checkpoint admission totality, Definition 818.115 and Definition 818.110. These results identify the terminal read and root admission route with the displayed finite spine from I through N, so no off-route checkpoint source remains. □

Theorem 818.119 (FiniteTraceEvaluator terminal admission totality)**.** *Terminal admission for an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *checkpoint packet is total only through the finite* BHist/hsame/Cont/Pkg/NameCer *packet*

$$I, T, A, V, H, C, P, N.$$

The admission surface reads trace evidence from T*, transports it through* H*, replays it through* C*, packages provenance through* P*, and names the local certificate through* N*. It does not admit a host evaluator, hidden acceptance predicate, total truth predicate, total halting classifier, unbounded evaluator, or infinite trace semantics.*

Proof. Project the terminal packet through Definition 818.76. The admission surface is already finite: I names the coded input, T names the finite BHist trace, A names accepted-result classification, and V names the validation-boundary refusal row.

Next use the root checkpoint admission totality theorem. It makes admission total only along $I \to T \to A \to V \to H \to C \to P \to N$, with H, C, P, N as the displayed transport, replay, provenance, and local naming rows.

The remaining terminal clauses are exhausted by the terminal NameCert obligation surface, Definition 818.117. Since that surface lists only finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ data, terminal admission has no further truth, halting, host-runtime, or infinite-trace gate. □

Theorem 818.120 (FiniteTraceEvaluator terminal reflection handoff)**.** *Every reflection-runtime consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal checkpoint receives only the finite checkpoint packet*

$$I, T, A, V, H, C, P, N.$$

The handoff exposes the finite BHist *route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *naming surface. It provides no host evaluator, hidden acceptance predicate, total truth predicate, total halting classifier, infinite trace semantics, or alternate runtime evaluator.*

Proof. Start from the reflection-runtime route, Definition 818.81. It sends every $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ read through the displayed sequence $I \to T \to A \to V \to H \to C \to P \to N$.

Use terminal runtime-candidate bridge, Definition 818.116. That bridge factors runtime-candidate provenance through the sibling $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ and $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ surfaces before returning to the displayed terminal rows.

Finally combine terminal admission totality above with checkpoint-budget exhaustion. The admissible checkpoint packet is already exhausted by finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ data, so a reflection-runtime consumer receives no hidden acceptance predicate, total truth predicate, host evaluator, total halting classifier, alternate evaluator, or infinite trace semantics. □

Theorem 818.121 (FiniteTraceEvaluator terminal runtime consumer factorization)**.** *Every terminal runtime consumer of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *route carrier factors through the finite packet*

$$E = (T, Q, R, X, H, C, P, N)$$

and through the sibling $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *and* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *surfaces before reading any evaluator result. The rows* T, Q, R, X, H, C, P, N *are respectively the finite* BHist *trace row, checkpoint row, runtime row, image and readback row, componentwise* hsame *transport row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *naming row. The carrier gate is the* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *TasteGate readiness surface of Definition 818.28.*

Proof. Start from the terminal route carrier and terminal route exposure of Definition 818.113 and Definition 818.115. They display the whole terminal packet as T, Q, R, X, H, C, P, N and leave no terminal coordinate outside those rows.

Next apply the terminal checkpoint accountability surface, Definition 818.89. It forces checkpoint-facing reads to remain accountable to a finite BHist route before runtime access is consumed.

Now use checkpoint-runtime acceptance lock and runtime route coverage, Definition 818.95 and Definition 818.96. These route the runtime consumer through $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ and $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ before the evaluator result is read.

Finally attach the terminal runtime-candidate bridge and the TasteGate readiness surface. The bridge keeps the consumer on the displayed terminal rows, while Definition 818.28 supplies the carrier gate that makes the public read field-faithful over the same packet. □

Theorem 818.122 (FiniteTraceEvaluator terminal NameCert exhaustion)**.** *For an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal route carrier* $E = (T, Q, R, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *terminal obligations are exhausted by the following finite rows: trace finiteness for* T*, checkpoint admission for* Q*, runtime totality for* R*, readback exactness for* X*, componentwise* hsame *transport for* H*, displayed* Cont *replay for* C*,* Pkg *provenance for* P*, and local naming for* N*. No further* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *clause is available outside these rows.*

Proof. Read the local terminal packet from Definition 818.117. It already identifies the terminal NameCert surface with trace, checkpoint, runtime, readback, transport, replay, provenance, and naming obligations over T, Q, R, X, H, C, P, N.

The checkpoint part is closed by terminal checkpoint accountability, Definition 818.89. It binds checkpoint admission to the finite BHist route and prevents the checkpoint row from becoming a hidden source.

The runtime and readback part is closed by Definition 818.105 and Definition 818.100. These theorems make the runtime route total on the displayed packet and force the readback replay through the same finite rows.

It remains to account for transport, replay, provenance, and naming. Terminal route exposure supplies H, C, P, N as the displayed hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows, and the TasteGate readiness theorem keeps those rows as the public carrier surface. Thus the listed obligations exhaust the terminal local NameCert package. □

Theorem 818.123 (FiniteTraceEvaluator terminal source lock)**.** *No terminal runtime source of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *route carrier exposes a host validator, hidden runtime state, unbounded evaluator, infinite trace, total truth predicate, total halting classifier, or unchecked compiler theorem. Every source available to a terminal consumer is locked to the displayed finite rows* T, Q, R, X, H, C, P, N*, to the sibling* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ *and* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *surfaces named by provenance, and to the carrier gate of Definition 818.28.*

Proof. First apply terminal route exposure. The carrier $E = (T, Q, R, X, H, C, P, N)$ has no coordinate for host validation, hidden runtime state, unbounded evaluation, infinite traces, or unchecked compiler theorems.

Next use checkpoint route nonescape, Definition 818.106. It prevents a checkpoint-facing consumer from replacing the displayed finite route by an alternate evaluator output or a detached runtime source.

Then apply runtime acceptance nonescape and validation refusal closure, Definition 818.99 and Definition 818.97. These exclude total truth predicates, total halting classifiers, host-valid runtime judgments, hidden runtime state, unbounded evaluator semantics, infinite traces, and unchecked compiler theorems from the runtime acceptance row.

Finally combine runtime route coverage with terminal runtime-candidate bridge. Every runtime-facing source passes through $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ and $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$ before it reaches the evaluator result, and then returns to T, Q, R, X, H, C, P, N. The TasteGate readiness theorem supplies the carrier gate over that same finite surface, so no other terminal runtime source is exposed. □

Theorem 818.124 (FiniteTraceEvaluator terminal checkpoint exactness)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *terminal checkpoint packet* $E = (T, Q, R, X, H, C, P, N)$*, any terminal checkpoint consumer reads exactly the finite trace row* T*, checkpoint row* Q*, runtime row* R*, readback row* X*, componentwise* hsame *row* H*, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*. The consumer does not obtain host validation, a hidden runtime state, an unbounded evaluator, an infinite trace source, a total truth classifier, a total halting classifier, or a second checkpoint source.*

Proof. Begin with terminal runtime consumer factorization, Definition 818.121. It sends every terminal runtime consumer through the displayed packet T, Q, R, X, H, C, P, N and through the $\mathsf{RuntimeCandidateBoundary}^{\uparrow}/\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ consumer surface before any evaluator result is read.

Next apply terminal NameCert exhaustion, Definition 818.122. The local certificate obligations are exactly trace finiteness, checkpoint admission, runtime totality, readback exactness, componentwise transport, displayed replay, provenance, and local naming over the same rows.

Finally use terminal source lock, Definition 818.123. Since every terminal source is locked to T, Q, R, X, H, C, P, N and the named sibling consumer surfaces, no host validator, hidden runtime state, unbounded evaluator, infinite trace source, total truth classifier, total halting classifier, or second checkpoint source can be read by the checkpoint consumer. □

Theorem 818.125 (FiniteTraceEvaluator runtime image totality)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *root packet* $E = (I, T, A, V, H, C, P, N)$*, each runtime-facing image or readback route factors through the displayed finite rows*

$$I \to T \to A \to V \to H \to C \to P \to N$$

and through the $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer surface. Terminal runtime consumers read the corresponding terminal packet* T, Q, R, X, H, C, P, N*; no runtime image is detached from these finite packet rows.*

Proof. First apply root runtime admission, Definition 818.107. It fixes the runtime entry surface of an accepted packet as the coded input, finite trace, accepted-result, validation-boundary, transport, continuation, provenance, and local naming rows I, T, A, V, H, C, P, N.

Next use checkpoint route coverage, Definition 818.108. It makes every checkpoint-facing consumer pass through the same root rows before a downstream runtime result is consumed, so the image and readback route has no detached checkpoint coordinate.

Finally attach terminal runtime consumer factorization, Definition 818.121. The terminal consumer is forced through the $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ surface and then through T, Q, R, X, H, C, P, N, which is precisely the terminal reading of the same finite route. Hence every runtime-facing image or readback factors through the displayed rows. □

Theorem 818.126 (FiniteTraceEvaluator root readback non-escape)**.** *Root readback for an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet exports only the root rows* I, T, A, V, H, C, P, N *or the terminal rows* T, Q, R, X, H, C, P, N *visible at the* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *surface. It excludes a host evaluator, total truth classifier, total halting classifier, infinite trace source, unchecked compiler theorem, and detached runtime state.*

Proof. Start from the root NameCert obligations, Definition 818.61. They place carrier admission, checkpoint acceptance, bounded replay, ledger exactness, and consumer non-escape on the displayed root packet I, T, A, V, H, C, P, N.

Use runtime ledger non-escape, Definition 818.109. The runtime ledger exports no host evaluator, hidden runtime acceptance predicate, total truth predicate, total halting classifier, unbounded evaluator, infinite trace, or second checkpoint source outside the same root rows.

It remains to account for terminal readback. Terminal source lock, Definition 818.123, restricts every terminal runtime source to T, Q, R, X, H, C, P, N and to the sibling runtime and reflection checkpoint surfaces named by provenance. Therefore root readback can export only the root or terminal finite rows and cannot escape to host evaluation, total truth, total halting, infinite traces, unchecked compiler theorems, or detached runtime state. □

Theorem 818.127 (FiniteTraceEvaluator root checkpoint budget image exhaustion)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *root checkpoint packet* $E = (I, T, A, V, H, C, P, N)$*, each accepted checkpoint-budget read is exhausted by the displayed coded input row* I*, finite trace row* T*, accepted-result classifier row* A*, validation-boundary row* V*, componentwise* hsame *row* H*, displayed* Cont *row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*. The budget image has no additional coordinate for host validation, total truth, total halting, unbounded evaluation, infinite traces, or a second checkpoint source.*

Proof. Begin with the root image row, Definition 818.46. It identifies the accepted root image as the eight displayed rows I, T, A, V, H, C, P, N.

Next apply root readback exactness and root ledger factorization, Definition 818.47 and Definition 818.48. Accepted results pass through $I \to T \to A$, while validation-facing reads remain on V, H, C, P, N and the downstream ledger reads no outside source.

Finally use the scope commitment recorded in Definition 818.59 and the $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ closure status in chapter 818. These bind the same carrier, classifier, validation boundary, hsame transport, Cont replay, Pkg provenance, local $\mathsf{NameCert}$ row, and TasteGate surface as the whole checkpoint-budget image, so the listed exclusions have no packet row to enter through. □

Theorem 818.128 (FiniteTraceEvaluator root runtime readback non-escape)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet, runtime readback from the root checkpoint surface factors through the displayed image and readback rows of Definition 818.46 and Definition 818.47. It cannot import host validation, total truth, total halting, infinite traces, or an unbounded evaluator into the root runtime package.*

Proof. First read the packet through the root image row. The only root-facing runtime coordinates available to a consumer are I, T, A, V, H, C, P, N.

Next use root readback exactness. The accepted-result side returns through $I \to T \to A$, and the validation-facing side returns through V, H, C, P, N; neither side exposes a host-side validation judgment or an unbounded evaluator coordinate.

It remains to rule out hidden runtime imports. Apply root ledger factorization and root exactness scope, Definition 818.48 and Definition 818.59. Since the ledger and exactness surface are exhausted by the same finite $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ packet, runtime readback has no route to total truth, total halting, infinite traces, or any evaluator beyond the displayed finite packet. □

Theorem 818.129 (FiniteTraceEvaluator root downstream NameCert package)**.** *The downstream* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *obligation surface for an accepted root checkpoint packet is the package consisting of the checkpoint budget image of Definition 818.127, the runtime readback non-escape*

row of Definition 818.128, and the root image, readback, ledger, and exactness surface of Definition 818.46, Definition 818.47, Definition 818.48, and Definition 818.59. No downstream NameCert clause asks for host validation, total truth, total halting, unbounded evaluation, infinite traces, or a detached runtime evaluator.

Proof. First assemble the two local rows. The checkpoint-budget image theorem gives the displayed root packet I, T, A, V, H, C, P, N, and the runtime readback non-escape theorem keeps runtime reads on that same packet.

Next attach the existing root image and readback surface. The root image row lists the available coordinates, while root readback exactness separates accepted-result reads $I \to T \to A$ from validation-facing reads V, H, C, P, N.

Finally add ledger factorization and root exactness scope. They make the downstream consumer factor through the displayed $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ package and through no outside source. Thus the downstream NameCert obligation surface is exactly the named root image, readback, budget, runtime, ledger, and exactness package. □

Theorem 818.130 (FiniteTraceEvaluator root runtime checkpoint obligation package)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *root packet*

$$E = (I, T, A, V, H, C, P, N),$$

the root runtime checkpoint obligation package is the finite package consisting of the checkpoint rows I, T, A*, the bounded trace rows, the runtime image rows, the validation ledger* V, H, C, P, N*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row* N*. It exposes no host validation, total truth predicate, total halting predicate, unbounded evaluator, infinite trace, detached runtime image, or second checkpoint source.*

Proof. Start with checkpoint-budget exhaustion and runtime image route determinacy, Definition 818.80 and Definition 818.84. They place the checkpoint read and runtime image on the same finite route I, T, A, V, H, C, P, N.

Next attach the validation ledger. By Definition 818.85, every validation-facing read is exhausted by V, H, C, P, N over the finite trace source I, T, A.

Finally add the downstream local certificate surface. The root downstream NameCert package of Definition 818.129 and the body-level obligations of Definition 818.61 bind the same $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{Nam}$ rows, so the obligation package has no outside runtime or validation coordinate. □

Theorem 818.131 (FiniteTraceEvaluator root image runtime totality)**.** *Every accepted root image/runtime read of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is exhausted by the checkpoint-budget rows and the runtime-image rows of the displayed packet* I, T, A, V, H, C, P, N*. The read is finite over the coded input, trace, accepted-result classifier, validation boundary, componentwise transport, continuation route, provenance, and local naming rows; it does not project host validation, total truth, total halting, unbounded evaluation, infinite trace semantics, or a detached runtime source.*

Proof. First apply Definition 818.127. It lists the root checkpoint budget image as exactly the rows I, T, A, V, H, C, P, N.

Next use Definition 818.84. Runtime image reads preserve the same finite route, so two accepted consumers that preserve these rows read the same image.

It remains to exclude hidden runtime coordinates. Apply Definition 818.128 and Definition 818.85. The first keeps runtime readback on the root image rows, and the second exhausts the validation ledger inside V, H, C, P, N. □

Theorem 818.132 (FiniteTraceEvaluator root NameCert readiness handoff)**.** *The obligation package of Definition 818.130 is the handoff consumed by downstream runtime-checkpoint and reflection-runtime chapters. Any such downstream consumer receives only the finite checkpoint source I, T, A, the validation and structural rows V, H, C, P, N, the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row.*

Proof. Begin with the obligation package above. It already assembles the checkpoint rows, bounded trace route, runtime image, validation ledger, transport, continuation, provenance, and local naming rows.

Next apply Definition 818.129. That theorem identifies the downstream $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ surface as the same root image, readback, budget, runtime, ledger, and exactness package.

Finally use Definition 818.81 and Definition 818.135. Reflection-runtime consumers therefore receive the finite route $I \to T \to A \to V \to H \to C \to P \to N$ and no host validation, total truth, total halting, unbounded evaluator, infinite trace, detached runtime source, or second checkpoint source. □

Theorem 818.133 (FiniteTraceEvaluator root acceptance totality)**.** *For every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *root read of the packet $E = (I, T, A, V, H, C, P, N)$, the read factors through the coded input row I, the finite trace row T, the accepted-result classifier row A, the validation-boundary row V, the componentwise* hsame *row H, the displayed* Cont *row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ *row N. No accepted root read reaches host validation, total truth, total halting, unbounded evaluation, infinite trace evidence, or a second checkpoint source.*

Proof. Start with the root trace-result totality route, Definition 818.30. It forces accepted-result reads through $I \to T \to A$ before any downstream consumer can inspect the accepted classifier row.

Next attach the validation boundary and image rows, Definition 818.31 and Definition 818.46. These identify the validation-facing side as V, H, C, P, N and list the whole accepted root packet as I, T, A, V, H, C, P, N.

Finally use root readback exactness and root NameCert obligations, Definition 818.47 and Definition 818.61. The readback surface separates accepted-result data from validation-facing data, while the local naming row closes the same BHist/hsame/Cont/Pkg/NameCert route. Thus every accepted root read is exhausted by the displayed finite packet. □

Theorem 818.134 (FiniteTraceEvaluator root boundary completion)**.** *The root boundary of an accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet is exhausted by the carrier of Definition 818.1, bounded-trace coverage, host-validation refusal, checkpoint routing, root NameCert obligations, ledger non-escape, and route totality. Equivalently, every root boundary consumer factors through the packet rows I, T, A, V, H, C, P, N and has no additional boundary clause outside that finite route.*

Proof. First project the carrier and bounded trace coverage. The carrier displays the packet rows I, T, A, V, H, C, P, N, and Definition 818.4 keeps accepted trace evidence finite.

Next add host-validation refusal and checkpoint routing by Definition 818.35 and Definition 818.34. The validation side stops at the displayed boundary rows, and the checkpoint side reads the same finite packet.

It remains to close the naming and ledger surface. Apply Definition 818.61, Definition 818.48, and Definition 818.68. The local NameCert row, ledger factorization, and route totality give the same finite BHist/hsame/Cont/Pkg/NameCert package, so no root boundary clause remains outside I, T, A, V, H, C, P, N. □

Theorem 818.135 (FiniteTraceEvaluator root consumer readiness lock)**.** *Every downstream* ReflectionRuntimeChe *consumer that reads an accepted* FiniteTraceEvaluator$^{\uparrow}$ *packet can read it only through the completed root route of Definition 818.134. The consumer receives accepted-result data through $I \to T \to A$ and validation-facing data through V, H, C, P, N, with no access to a host validator, total truth predicate, total halting predicate, infinite trace, unbounded evaluator, or detached checkpoint source.*

Proof. Begin with the root acceptance route of Definition 818.133. It supplies the only accepted root packet available to downstream consumers.

Next apply the boundary completion theorem above. Its carrier, bounded-trace, validation-refusal, checkpoint, NameCert, ledger, and route-totality rows exhaust the root boundary of the same packet.

Finally use the checkpoint consumer route and runtime readback non-escape, Definition 818.8 and Definition 818.128. A ReflectionRuntimeCheckpoint$^{\uparrow}$ consumer therefore reads only the completed finite route and cannot attach any outside host, truth, halting, trace, or runtime source to the accepted packet. □

The FiniteTraceEvaluator$^{\uparrow}$ bounded evaluator route reads the RuntimeCandidateBoundary$^{\uparrow}$ packet of chapter 591 before it reads its bounded evaluator rows. The consumed dependency is the displayed source row, transition row, accepted-result row, componentwise transport row, route row, provenance row, and local-name row; these rows supply the runtime candidate boundary that the evaluator later transports through checkpoint, image, acceptance, and trace-budget surfaces. Every FiniteTraceEvaluator$^{\uparrow}$ runtime checkpoint read first consumes the source, transition, accepted-result, transport, route, provenance, and local-name rows of the RuntimeCandidateBoundary$^{\uparrow}$ source in chapter 591. Thus the evaluator checkpoint route reads the candidate boundary before it reads the bounded trace and checkpoint rows; it does not fabricate a host validator, a total truth predicate, or an unbounded runtime stream. The FiniteTraceEvaluator$^{\uparrow}$ root route consumes the candidate rows of chapter 591 before any runtime-reflection rows are read. This horizontal dependency places the object-level source, transition, accepted-result, transport, route, provenance, and local-name rows of RuntimeCandidateBoundary$^{\uparrow}$ in front of the evaluator's runtime-facing reflection surface.

Theorem 818.136 (Finite trace evaluator runtime-candidate dependency route)**.** *Every accepted* FiniteTraceEvaluator$^{\uparrow}$ *runtime read that evaluates a candidate trace first exposes the* RuntimeCandidateBoundary$^{\uparrow}$ *sibling row of chapter 591. The read then factors through the checkpoint-runtime rows, finite trace source, image readback surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row of the evaluator packet.*

Consequently the evaluator cannot consume a candidate trace by replacing the sibling candidate boundary with host validation, an unbounded runtime stream, or a hidden total-truth witness.

Proof. Start with the sibling route named by Definition 818.2. It places the runtime candidate boundary before the bounded evaluator rows, so the candidate trace is read only after the displayed RuntimeCandidateBoundary$^{\uparrow}$ source has been exposed.

Next apply Definition 818.116. This forces the terminal runtime consumer through RuntimeCandidateBoundary$^{\uparrow}$ and ReflectionRuntimeCheckpoint$^{\uparrow}$ before it reaches the evaluator result.

Finally use Definition 818.46 and Definition 818.3. The former records the finite trace source and image readback surface, while the latter exhausts the transport, replay, provenance, and local naming rows. Thus every accepted candidate-trace read follows the displayed sibling dependency route. □

The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ route consumes chapter 591 as its candidate source and supplies bounded trace evidence to chapter 997. The handoff is source, trace, and checkpoint routing: the evaluator does not fabricate a detached runtime citation, and the checkpoint packet reads only the bounded trace evidence exposed by this finite route. The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ sibling route reads chapter 469 before checkpoint or runtime readback. Every accepted finite evaluator route exposes the $\mathsf{ContinuationTrace}^{\uparrow}$ sibling through bounded trace rows, then consumes the checkpoint and runtime rows already listed by the hub. The route refuses host validation, total truth predicates, and unbounded runtime streams.

Theorem 818.137 (FiniteTraceEvaluator ContinuationTrace route)**.** *Every accepted* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *packet that exports checkpoint or runtime readback first reads the* $\mathsf{ContinuationTrace}^{\uparrow}$ *sibling through bounded trace rows. The visible route is finite trace exposure, checkpoint admission, runtime readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming; it has no coordinate for host validation, total truth predicates, or unbounded runtime streams.*

Proof. Project the evaluator packet through the existing hub child routes. The bounded trace rows are the only trace source available before checkpoint and runtime rows are consumed.

Read the $\mathsf{ContinuationTrace}^{\uparrow}$ sibling at that boundary. Its finite replay supplies the trace rows that checkpoint admission and runtime readback may inspect, while the evaluator carrier keeps the rows bounded by the same displayed packet.

The structural rows then transport, replay, source, and name the packet. Since no host validator, total truth predicate, or unbounded stream coordinate is present, every accepted read follows the stated finite route. □

The MetaCIC candidate-set SN handoff of chapter 3422 consumes chapter 818 only after the finite trace evaluator has displayed checkpoint rows, runtime totality, image readback, and acceptance boundaries. This forward route keeps the SN consumer attached to the evaluator's visible checkpoint and runtime surfaces rather than to an untracked kernel execution. The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ route supplies the bounded trace, checkpoint, runtime-totality, and image-readback rows consumed by chapter 765. A MetaCIC critical-path consumer may use finite evaluator evidence only after those displayed rows have passed through the evaluator source, accepted-result, validation-boundary, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ surface; it does not receive an unbounded runtime stream, hidden interpreter state, or host validation predicate. The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ runtime lattice reads finite trace data through chapter 280 and chapter 199 before it records checkpoint image and readback rows. Thus the evaluator remains a finite runtime packet: the $\mathsf{TuringMachine}^{\uparrow}$ transition surface and the $\mathsf{Computable}^{\uparrow}$ admission surface are exposed before checkpoint images, source readbacks, BHist histories, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{FiniteTraceEvaluator}^{\uparrow}}$ row are consumed. The $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ dependency lattice consumes the runtime-candidate boundary of chapter 591, routes finite diagonal trace evidence through chapter 492, and supplies bounded checkpoint evidence to chapter 997. This places the evaluator between the boundary that admits runtime candidates and the checkpoint consumer that reads bounded evidence, so the route is a finite dependency handoff rather than a detached kernel citation.

Chapter 819

A Concrete Naming Certificate for $\mathsf{LegalChannel}^{\uparrow}$

The $\mathsf{LegalChannel}^{\uparrow}$ packet realizes the legal event-flow channel surface of chapter 9407 as a finite NameCert object. It sits beside the trace discipline of chapter 469 and the audit inventory of chapter 823: those chapters display finite replay and finite audit rows, while the present packet binds source event-flow rows, channel-code rows, decoder rows, round-trip witnesses, transport, replay, provenance, and local naming for the GroundCompiler channel interface.

Definition 819.1 (LegalChannel carrier)**.** *A* $\mathsf{LegalChannel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, C, D, R, H, K, P, N)$$

with the following displayed rows:

1. *E is the source event-flow row, recording the finite GroundCompiler event flow accepted as the channel input;*
2. *C is the legal channel code row, recording the displayed code representation admitted by the legal channel classifier;*
3. *D is the decoder row, recording the finite readback operation from channel code back to event-flow data;*
4. *R is the round-trip witness row, recording that accepted channel reads through C and D return the same event-flow row E up to the displayed BEDC sameness rows;*
5. *H records componentwise* hsame *transport for the event-flow, code, decoder, round-trip, replay, provenance, and naming rows;*
6. *K records the displayed* Cont *replay routes by which a consumer reads the channel code, applies the decoder row, checks the round-trip row, and returns to the named event-flow row;*
7. *P is the* Pkg *provenance over chapter 9407, channel dependencies,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LegalChannel}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are E, C, D, R, H, K, P, N. It has no coordinate for a host package, proof term, source substring, metric report, motif row, theorem-code payload, or hidden compiler object outside the displayed packet.

Theorem 819.2 (LegalChannel lossless NameCert)**.** *For every accepted* $\mathsf{LegalChannel}^{\uparrow}$ *carrier* $L = (E, C, D, R, H, K, P, N)$*, an accepted channel read that starts from* E*, enters the legal channel code row* C*, and is decoded through* D *preserves the event-flow row named by* E*. The preservation witness is the displayed round-trip row* R*, transported only by* H *and replayed only through* K*.*

Proof. Project the accepted carrier. The source event-flow row, code row, decoder row, and round-trip row are the coordinates E, C, D, R of the same finite packet.

Read the accepted channel route through K. The replay path first inspects C, then applies the displayed decoder row D, and then checks the round-trip witness R.

Transport is componentwise. The only sameness data available to the route is H, so the decoded row can move only along the listed hsame comparisons and returns to the same named event-flow row E.

Finally, provenance and naming are read from P and N. They certify where the channel dependency and local name are recorded, but they add no second event-flow source. Hence the accepted read is lossless at the packet surface. □

Theorem 819.3 (LegalChannel hidden-input exclusion)**.** *For every accepted* $\mathsf{LegalChannel}^{\uparrow}$ *carrier* $L = (E, C, D, R, H, K, P, N)$*, no host package, proof term, source substring, metric report, motif row, theorem-code payload, or hidden compiler object is exported by the legal channel unless that object appears as part of the displayed packet rows* E, C, D, R, H, K, P, N*.*

Proof. Inspect the carrier coordinates. The legal channel exposes exactly the event-flow row, channel code row, decoder row, round-trip row, transport row, replay row, provenance row, and local naming row.

Suppose a consumer asks for a host package, proof term, source substring, metric report, motif row, theorem-code payload, or hidden compiler object. The classifier can answer only by pointing to one of the displayed coordinates of L.

If the requested object is not displayed in those coordinates, there is no packet row from which the consumer can read it. The replay route K and transport row H preserve already displayed rows, while P and N cite and name the packet rather than creating another export coordinate.

Therefore the legal channel excludes hidden input by finite carrier exhaustion. □

Theorem 819.4 (LegalChannel ledger non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{LegalChannel}^{\uparrow}$ *packet factors through the finite ledger*

$$E \to C \to D \to R \to H \to K \to P \to N.$$

In particular, no downstream channel consumer can bypass the event-flow row, legal code row, decoder row, round-trip witness, transport, replay, provenance, or local NameCert row.

Proof. Begin with a downstream consumer read. By carrier admission, the read starts from an accepted packet whose only public coordinates are E, C, D, R, H, K, P, N.

The channel part of the read factors through E, C, D, R. The event-flow row supplies the source, the code row supplies the legal representation, the decoder row supplies readback, and the round-trip row records the preservation check.

The structural part of the read factors through H, K, P, N. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming are the only routes by which the channel packet can be moved, inspected, cited, and named.

Thus every consumer route has the displayed ledger as its complete path. A bypass would require an extra coordinate, but the finite carrier has none. □

Theorem 819.5 (LegalChannel GroundCompiler handoff)**.** *The packet $L = (E, C, D, R, H, K, P, N)$ is the concrete* $\mathsf{NameCert}_{\mathsf{LegalChannel}^{\uparrow}}$ *handoff consumed by the GroundCompiler channel-bijection and no-hidden-input audit surface. Its channel-bijection contribution is the lossless route through C, D, R, and its no-hidden-input contribution is the carrier exhaustion of E, C, D, R, H, K, P, N.*

Proof. The GroundCompiler audit map separates legal event-flow encoding from hidden compiler input. The $\mathsf{LegalChannel}^{\uparrow}$ packet supplies exactly the finite rows needed for that separation.

For the channel-bijection side, use Definition 819.2. Accepted reads pass through the legal code row, decoder row, and round-trip row, so the event-flow row returned to the consumer is the row named by E.

For the no-hidden-input side, use Definition 819.3 and Definition 819.4. The first theorem excludes unlisted host, proof, source, metric, motif, theorem-code, and compiler objects; the second theorem forces every downstream read through the displayed ledger.

Consequently L is the concrete NameCert handoff for the GroundCompiler channel audit surface. It contributes a finite legal-channel packet rather than an ambient compiler object or host-side decoder. □

Closure status for $\mathsf{NameCert}_{\mathsf{LegalChannel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LegalChannel}^{\uparrow}$ packet is generated from a source event-flow row, legal channel code row, decoder row, round-trip witness row, componentwise hsame transports, displayed Cont channel replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegalChannel}^{\uparrow}}$, seed)

Scope closed: Definition 819.1, Definition 819.2, Definition 819.3, Definition 819.4, and Definition 819.5 bind the finite legal-channel carrier, lossless decode route, hidden-input exclusion, downstream ledger, and GroundCompiler handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegalChannel}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close proof evidence, theorem-code evidence, self-hosting discharge, Lean verification, audit cleanliness, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal-code decoding, round-trip preservation, hidden-input exclusion, downstream consumer factorization, provenance, and local naming for the same finite channel packet.

Chapter 820

A Concrete Naming Certificate for GroundCompilerRecognizerAuditPacket$^{\uparrow}$

The GroundCompilerRecognizerAuditPacket$^{\uparrow}$ carrier turns the recognizer-facing part of the GroundCompiler audit map of chapter 9407 into a finite BEDC packet. It sits beside the audit-map family surface of chapter 807, the certificate-compiler graph packet of chapter 473, and the taste-gate compiler witness of chapter 563. Those sibling chapters supply the family template, certificate handoff route, and chapter-gate witness rows; this chapter packages the GroundCompiler recognizer rows and non-evidence rows as the object those consumers may inspect.

Definition 820.1 (GroundCompiler recognizer audit packet carrier)**.** *A* GroundCompilerRecognizerAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, A, E, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the recognizer row naming the generated recognizers, accepted exports, and recognizer-generatedness checks from the GroundCompiler audit map;*
2. *A is the address-analysis row recording compiler-layer address, theorem-code, chapter-code, and classifier-quotient reads that are permitted only as displayed audit data;*
3. *E is the evidence row containing event-flow conservativity, no-hidden-input, channel-bijection, and source-channel separation rows available to certificate consumers;*
4. *Q is the non-evidence row listing code-not-proof, report-not-evidence, motif-not-identity, quotient-not-raw-equality, and cannot-claim boundaries;*
5. *B is the bridge-request row that records which downstream packet may consume the recognizer audit surface, including the sibling packets of chapter 807, chapter 473, and chapter 563;*
6. *H records componentwise* hsame *transports for the recognizer, address-analysis, evidence, non-evidence, bridge-request, route, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which a consumer inspects R, A, E, Q, B without adding a hidden compiler proof row;*
8. *P is the* Pkg *provenance over* GroundCompiler*, chapter 9407, the sibling concrete packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}}$ *row.*
The carrier accepts exactly the displayed recognizer, address-analysis, evidence, non-evidence, bridge-request, transport, route, provenance, and local naming rows. It exports no proof that code is proof, no report-to-evidence conversion, no raw classifier equality, no hidden self-hosting tower, and no MetaCIC theorem outside the listed sibling handoff.

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognizerAuditPacketUp.GroundCompilerRecognizerAuditPacketTasteGate`
`single carrier alignment`

Theorem 820.2 (GroundCompiler recognizer audit packet obligation surface)**.** *For the carrier of Definition 820.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$G = (R, A, E, Q, B, H, C, P, N),$$

recognizer readback through R, address-analysis readback through A, evidence readback through E, non-evidence readback through Q, bridge-request readback through B, componentwise transport through H, displayed continuation through C, package provenance through P, and local naming through N.

Proof. Carrier admission is tuple projection from Definition 820.1. The recognizer, address-analysis, evidence, non-evidence, and bridge-request rows are all displayed before any downstream consumer can inspect the packet.

The transport and route rows preserve that displayed inventory. Componentwise hsame entries in H can only move the named rows, and the continuations in C can only replay routes through R, A, E, Q, B.

Finally P and N account for provenance and local naming. Since provenance cites the GroundCompiler audit map and the sibling concrete packets only as row sources, the local NameCert surface is exhausted by the nine displayed coordinates. □

Theorem 820.3 (GroundCompiler recognizer audit packet non-evidence boundary)**.** *Every public read of an accepted* $\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}$ *carrier keeps the non-evidence row Q visible. A consumer may cite recognizer-generatedness, accepted exports, event-flow conservativity, no-hidden-input, channel separation, and address-analysis rows only together with the boundaries that code is not proof, reports are not evidence, motifs are not object identity, classifier quotients are not raw equality, and conditional self-hosting is not an unconditional compiler tower.*

Proof. Start with the carrier projection. The positive recognizer and compiler-facing rows are R, A, E, while the refusal row is Q.

Apply the obligation surface of Definition 820.2. Any route that exposes R, A, E also exposes the same packet through H, C, P, N, so the route cannot drop Q without leaving the carrier.

The row Q is therefore part of the public read, not a side note. Since the packet contains no coordinate converting code, reports, motifs, quotients, or conditional self-hosting into stronger evidence, every consumer remains inside the listed non-evidence boundary. □

Theorem 820.4 (GroundCompiler recognizer audit packet consumer handoff)**.** *The bridge-request row B hands an accepted* $\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}$ *packet only to displayed BEDC consumers. In particular, audit-map family consumers read the packet as a family row, certificate-compiler consumers read it as finite compiler-side provenance, and taste-gate compiler witnesses read it as chapter-gate evidence tied to the same recognizer and non-evidence rows.*

Proof. Project the bridge-request coordinate B. Its listed sibling destinations are the audit-map family, certificate compiler, and taste-gate compiler witness chapters named in the carrier provenance.

For the audit-map family, the handoff lands as a visible family row and keeps the positive, conditional, obstruction, and frontier split supplied by chapter 807. For the certificate compiler, the handoff supplies finite compiler-side provenance rather than host compiler semantics. For the taste-gate compiler witness, the handoff supplies named chapter-gate evidence rather than an ambient approval principle.

The route rows H and C preserve the same R, A, E, Q, B inventory during each handoff. Thus no downstream consumer receives a stronger object than the displayed recognizer audit packet. □

Theorem 820.5 (GroundCompiler recognizer audit packet NameCert)**.** *The public* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerA}}$ *packet is exactly the finite surface*

$$R, A, E, Q, B, H, C, P, N.$$

It packages the recognizer and non-evidence rows of chapter 9407 for concrete BEDC consumers, and it exports no hidden compiler proof, no proof-producing code interpretation, no evidence extracted from reports, no raw equality behind classifier quotients, and no unstated MetaCIC closure.

Proof. Read the public packet by carrier projection and the local obligation surface. The available rows are the recognizer row R, address-analysis row A, evidence row E, non-evidence row Q, bridge-request row B, transports H, routes C, provenance P, and local naming N.

Use Definition 820.3 to keep the refusal rows attached to every recognizer-facing read. Use Definition 820.4 to restrict downstream use to the displayed sibling consumers.

No additional coordinate remains. Therefore the public NameCert packet is exactly $R, A, E, Q, B, H, C, P, N$, with the listed compiler-proof, report-evidence, raw-equality, and MetaCIC-closure claims outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}$ packet is generated from finite BHist rows for recognizers, address analysis, positive evidence, non-evidence boundaries, bridge requests, componentwise transports, displayed continuations, provenance, and local naming. Recognizer-facing reads are replayed by Cont through R, A, E, Q, B, so evidence rows and refusal rows remain attached in one public packet. hsame transports the recognizer, address, evidence, non-evidence, and bridge rows without strengthening them, while Pkg records the GroundCompiler audit map and sibling concrete consumers. The packet packages recognizer audit data without turning code or reports into proofs.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the GroundCompiler recognizer audit packet carrier, obligation surface, non-evidence boundary, consumer handoff, and public NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerAuditPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim compiler proofs, report-to-evidence conversion, raw classifier equality, hidden self-hosting towers, or unstated MetaCIC closure.

Upgrade path: Obligation closure requires Lean encodings of the recognizer audit carrier, non-evidence preservation, consumer handoff, and public packet theorem.

Chapter 821

A Concrete Naming Certificate for $\mathsf{GroundCompilerRecognition}^{\uparrow}$

The $\mathsf{GroundCompilerRecognition}^{\uparrow}$ packet realizes the generated-recognition surface of chapter 9407 as a finite NameCert carrier. It sits beside the finite evaluator packet of chapter 818: the evaluator records bounded trace rows and accepted-result readback, while this packet records generated recognizer rows, accepted-export rows, finite evaluator handoff rows, report non-evidence rows, and cannot-claim ledger rows before a downstream compiler consumer treats recognition as evidence.

Definition 821.1 (GroundCompiler recognition carrier)**.** *A* $\mathsf{GroundCompilerRecognition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, G, A, T, V, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the GroundCompiler event-flow and source-channel input row under recognition;*
2. *G is the generated recognizer row for packages, name certificates, derivation certificates, theorem identifiers, chapter records, or manifest records;*
3. *A is the accepted-export row through which a recognized artifact may be consumed by a theorem-facing or NameCert-facing route;*
4. *T is the finite trace evaluator handoff row, read through the bounded trace and accepted-result rows of chapter 818;*
5. *V is the report non-evidence row separating metric reports, motif reports, theorem code, chapter code, and readable source from proof or certificate evidence;*
6. *L is the cannot-claim ledger row for hidden compiler input, raw event-flow equality, report-as-proof, code-as-proof, and unrecorded self-hosting claims;*
7. *H records componentwise* hsame *transports for the input, recognizer, export, evaluator handoff, non-evidence, cannot-claim, provenance, and local naming rows;*
8. *C records displayed* Cont *routes for recognizer readback, accepted-export readback, finite evaluator handoff, non-evidence inspection, and ledger replay;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *chapter 9407, and chapter 818;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are $I, G, A, T, V, L, H, C, P, N$. It contains no host recognizer, no accepted export outside A, no evaluator validation outside T, no proof from reports or code, and no unlisted compiler input.

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionTasteGate single carrier alignment`

Theorem 821.2 (GroundCompiler recognition NameCert obligations)**.** *For the carrier of Definition 821.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $R = (I, G, A, T, V, L, H, C, P, N)$;*
2. *generated-recognizer exposure through G after the event-flow and source-channel input row I has been displayed;*
3. *accepted-export routing through A, with no consumer route bypassing the accepted-export row;*
4. *finite trace evaluator handoff through T, tied to the bounded trace and accepted-result rows of chapter 818;*
5. *report and cannot-claim exhaustion through V, L, H, C, P, N, so reports, code, raw classifier quotients, and unrecorded self-hosting claims remain outside proof and certificate evidence.*

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionNameCert obligations`

Proof. Carrier admission is tuple projection. The finite packet displays exactly $I, G, A, T, V, L, H, C, P, N$.

Read recognition through I and G. The source-like event row is visible before the generated recognizer row is consumed, so recognizer evidence cannot replace the input coordinate.

Accepted export is then confined to A. Since the carrier has no second export coordinate, every theorem-facing or NameCert-facing route must pass through the displayed accepted-export row.

The finite evaluator handoff is confined to T. By citing the sibling finite evaluator packet, the handoff exposes bounded trace and accepted-result rows without adding host evaluator validation or an unbounded execution stream.

Finally inspect V, L, H, C, P, N. The report non-evidence and cannot-claim rows state the excluded readings, while the transport, route, provenance, and naming rows replay only the displayed packet. Thus the five obligations exhaust the local certificate surface. □

Theorem 821.3 (GroundCompiler recognition ledger non-escape)**.** *Every accepted* $\mathsf{GroundCompilerRecognition}^{\uparrow}$ *packet exports recognition evidence only through the finite rows*

$$I, G, A, T, V, L, H, C, P, N.$$

It exports no hidden compiler input, no host recognizer, no accepted export outside the certificate gate row, no proof from theorem code or chapter code, no proof from metric or motif reports, no raw event-flow equality from classifier membership, and no self-hosting tower outside the displayed ledger.

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionCarrier ledger nonescape`

Proof. Project the accepted carrier of Definition 821.1. The listed rows are the whole exported BHist surface.

Apply Definition 821.2. Generated recognition is read through G, accepted exports through A, and evaluator handoff through T.

The non-evidence and cannot-claim rows then block hidden strengthening. Since V separates reports and code from proof or certificate evidence, and L lists the forbidden compiler readings, no public consumer can extract an unlisted recognizer, export, proof, raw equality, or compiler tower from the packet. □

Theorem 821.4 (GroundCompiler recognition finite-trace handoff)**.** *For every accepted* GroundCompilerRecognitio *carrier $R = (I, G, A, T, V, L, H, C, P, N)$, the evaluator-facing read factors through the finite trace evaluator rows cited by T. A downstream consumer may inspect the bounded trace and accepted-result handoff, then return to the generated recognizer and accepted-export rows G, A; it cannot turn the handoff into host evaluator validation, a total truth predicate, an infinite trace, or a proof object.*

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionCarrier finite trace handoff boundary`

Proof. Start with the handoff coordinate T. It is the only evaluator-facing row in the recognition carrier.

Use the finite trace evaluator packet of chapter 818. That sibling exposes coded input, finite trace-step rows, accepted-result classification, and a validation boundary, while excluding host validation, total halting, truth, and unbounded execution.

Return through H, C, P, N to the recognition rows. The route records only the displayed bounded handoff and then resumes at G and A, so it supplies recognition and accepted export evidence without adding an evaluator, proof object, infinite trace, or truth predicate. □

Theorem 821.5 (GroundCompiler recognition flow carrier exposure)**.** *For every accepted* GroundCompilerRecogniti *carrier $R = (I, G, A, T, V, L, H, C, P, N)$, the event-flow input, generated recognizer, accepted-export gate, finite-trace handoff, report non-evidence row, cannot-claim ledger, componentwise* hsame *rows, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row are all visible before recognition is consumed. No accepted consumer may read generated recognition while omitting I, G, or A.*

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionCarrier flow carrier exposure`

Proof. Project the carrier of Definition 821.1. The visible packet is exactly $I, G, A, T, V, L, H, C, P, N$.

Use Definition 821.4 for the evaluator-facing coordinate. It places the bounded trace and accepted-result handoff inside T and returns the read to G and A without adding host validation.

The remaining rows V, L, H, C, P, N expose the report boundary, cannot-claim ledger, transports, replay routes, provenance, and local name. Thus recognition consumption cannot skip the event-flow, recognizer, or accepted-export coordinates. □

Theorem 821.6 (GroundCompiler recognition certificate gate stability)**.** *Every theorem-facing or NameCert-facing read of an accepted* GroundCompilerRecognition$^{\uparrow}$ *carrier factors through the accepted-export row A. The generated recognizer row G, finite-trace handoff row T, report non-evidence row V, and cannot-claim ledger row L may constrain that read, but none of them creates a second certificate gate.*

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionCarrier certificate gate stability`

Proof. Start from the accepted-export coordinate A. It is the only row in the carrier whose role is theorem-facing or NameCert-facing export.

Apply the finite-trace handoff theorem. The handoff through T supplies bounded evaluator data and then returns to the recognizer and export rows; it does not create another accepted-export channel.

Finally inspect V and L. Reports, code, raw event-flow equality, and unrecorded self-hosting claims are listed as non-evidence or cannot-claim data. They therefore restrict the certificate gate rather than replacing A. □

Theorem 821.7 (GroundCompiler recognition NameCert non-escape)**.** *The local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$ *obligations for $I, G, A, T, V, L, H, C, P, N$ prevent accepted-export routing from bypassing A and prevent reports, chapter code, theorem code, metric reports, or motif reports from becoming proof evidence. Every accepted read remains inside the displayed recognition carrier and its finite-trace handoff.*

Proof. Use the NameCert obligation theorem. It lists carrier admission, generated-recognizer exposure, accepted-export routing, finite-trace handoff, and report-plus-ledger exhaustion over the same finite carrier.

The certificate gate stability theorem confines theorem-facing and NameCert-facing export to A. The finite-trace handoff theorem confines evaluator-facing data to T and returns it to G, A.

It remains to inspect the report and code rows. The row V separates reports and code from proof or certificate evidence, while L records the cannot-claim readings. Hence no accepted route can turn reports or code into proof evidence, and no route can bypass A. □

Theorem 821.8 (GroundCompiler recognition accepted-export totality)**.** *For every accepted* GroundCompilerRecog *carrier $R = (I, G, A, T, V, L, H, C, P, N)$, every theorem-facing export and every NameCert-facing export passes through the accepted-export row A. The carrier of Definition 821.1 and the obligations of Definition 821.2 leave no second export row outside A.*

Proof. Project the carrier. The theorem-facing and NameCert-facing export coordinate in the displayed packet is A.

Apply Definition 821.2. Its accepted-export obligation states that no consumer route bypasses the accepted-export row.

Use certificate gate stability, Definition 821.6. The generated recognizer row G, finite-trace handoff row T, report boundary row V, and ledger row L constrain the read, but none supplies a second certificate gate. Hence every such export factors through A. □

Theorem 821.9 (GroundCompiler recognition report boundary exhaustion)**.** *For every accepted* $\mathsf{GroundCompilerRecognition}^{\uparrow}$ *carrier $R = (I, G, A, T, V, L, H, C, P, N)$, every report read, chapter-code read, theorem-code read, metric-report read, or motif-report read passes through the report non-evidence row V before evidence use is refused. The carrier and NameCert obligations therefore exhaust the report boundary before any downstream consumer can treat those reads as proof or certificate evidence.*

Proof. Start from Definition 821.1. The row V is the displayed report non-evidence row separating metric reports, motif reports, theorem code, chapter code, and readable source from proof or certificate evidence.

Apply Definition 821.2. Its report and cannot-claim exhaustion obligation routes the listed report and code reads through V, L, H, C, P, N before a consumer may inspect the packet as evidence.

Now use Definition 821.7. It prevents reports and code from becoming proof evidence and keeps every accepted read inside the displayed carrier and finite-trace handoff. Thus V is the report boundary, and evidence use is refused after that boundary is displayed. □

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionReportBoundary exhaustion`

Theorem 821.10 (GroundCompiler recognition cannot-claim ledger coverage)**.** *For every accepted* $\mathsf{GroundCompilerRecognition}^{\uparrow}$ *carrier* $R = (I, G, A, T, V, L, H, C, P, N)$*, every forbidden compiler reading is covered by the cannot-claim ledger row* L*, componentwise* hsame *transports* H*, displayed replay routes* C*, package provenance* P*, and the local* NameCert *row* N*. In particular hidden compiler input, raw event-flow equality, report-as-proof, code-as-proof, and unrecorded self-hosting claims cannot escape the ledger surface.*

Proof. Project the packet of Definition 821.1. The forbidden compiler readings are listed in the cannot-claim row L, and the structural rows H, C, P, N are the only transport, route, provenance, and naming rows available to replay them.

Apply Definition 821.2. Its final obligation covers report and cannot-claim exhaustion through V, L, H, C, P, N over the same finite carrier.

Use ledger non-escape, Definition 821.3. It already excludes hidden compiler input, host recognizers, proof from code or reports, raw event-flow equality, and self-hosting outside the displayed ledger. Therefore each forbidden reading is covered by L, H, C, P, N and no compiler reading escapes the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerRecognition}^{\uparrow}$ packet is generated from event-flow and source-channel input rows, generated recognizer rows, accepted-export rows, finite trace evaluator handoff rows, report non-evidence rows, cannot-claim ledger rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 821.1, Definition 821.2, Definition 821.8, Definition 821.9, and Definition 821.10 bind the public accepted-export surface, report-boundary exhaustion, and cannot-claim ledger coverage to the displayed BHist, hsame, Cont, Pkg, and NameCert primitives.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GroundCompilerRecognition}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GroundCompilerRecognitionUp.GroundCompilerRecognitionNameCert\_obligations`

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognitionUp.taste gate`

Not claimed: This chapter does not prove self-hosting for every compiler candidate, validate the evaluator from inside the packet, turn reports or code into proof, identify recognizer classes with raw event-flow equality, or provide bridge checking.

Upgrade path: No further paper-side theory upgrade remains before standard bridge recording.

Chapter 822

A Concrete Naming Certificate for $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$

The $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ packet concretizes the GroundCompiler audit map of chapter 9407 as a finite verdict carrier. It sits beside the obstruction socket of chapter 826 and the frontier packet of chapter 833: those packets expose audit-family obstructions and frontier rows, while this packet binds query output, forbidden-dependency comparison, generated recognizer evidence, certificate-gate evidence, replay provenance, and no-hidden-input boundaries into one NameCert surface.

Definition 822.1 (GroundCompiler audit verdict carrier)**.** *A* $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, Q, F, V, A, R, C, B, H, K, P, N)$$

with the following visible rows:

1. *M is the GroundCompiler module row naming the inspected event-flow, channel, recognizer, certificate-gate, metric, motif, bootstrap, or main-theorem surface;*
2. *Q is the displayed query-output row for the inspected module surface;*
3. *F is the forbidden-dependency row naming dependencies and host-side evidence forms excluded by the audit;*
4. *V is the verdict row recording the finite comparison between Q and F;*
5. *A is the audit-family alignment row tying the verdict to the audit-map obstruction and frontier classifications;*
6. *R is the generated-recognizer row exposed by the GroundCompiler layer;*
7. *C is the certificate-gate row through which accepted exports are mediated;*
8. *B is the no-hidden-input boundary row for code, reports, motifs, classifier quotients, source substrings, and host compiler structure;*
9. *H records componentwise* hsame *transport for module, query, forbidden, verdict, alignment, recognizer, gate, boundary, provenance, and naming rows;*
10. *K records displayed* Cont *replay routes for query exposure, forbidden-row comparison, verdict exposure, recognizer readback, certificate-gate readback, and boundary readback;*

11. *P is the* Pkg *provenance over the GroundCompiler audit map, sibling audit-map packets, recognizer rows, certificate-gate rows,* hsame, Cont, *and local* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditVerdict}^{\uparrow}}$ *row.*

The carrier exports no proof from theorem code, no certificate from a metric or motif report, no raw event-flow equality from a classifier quotient, and no hidden compiler input outside the displayed rows.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictTasteGate single carrier alignment`

Theorem 822.2 (GroundCompiler audit verdict obligations)**.** *For every* $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ *carrier*

$$G = (M, Q, F, V, A, R, C, B, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditVerdict}^{\uparrow}}$ *surface has five obligation rows:*

1. *carrier exposure: every public read first projects one of $M, Q, F, V, A, R, C, B, H, K, P, N$;*
2. *forbidden-row comparison: the verdict is read from the displayed comparison between Q and F;*
3. *verdict determinacy: transport of Q and F transports the same verdict row V and creates no second clean or refusal row;*
4. *replay provenance: every replay passes through the displayed continuation row K and package provenance row P;*
5. *non-escape: recognizer, certificate-gate, and no-hidden-input boundaries are exported only through R, C, B and the local naming row N.*

Proof. First project the carrier. The finite tuple displays the module, query, forbidden-dependency, verdict, alignment, recognizer, certificate-gate, boundary, transport, replay, provenance, and local naming rows.

The forbidden-row comparison is local to Q, F, V. The query-output row Q is paired with the forbidden row F, and the verdict row V records that finite comparison. Because the carrier has no second verdict coordinate, componentwise transport of Q and F through H transports the same verdict.

Replay is confined to K and P. A consumer may rerun the displayed query exposure, comparison, verdict exposure, recognizer readback, gate readback, or boundary readback only along the continuation routes recorded in K, with package ancestry recorded in P.

Finally inspect the GroundCompiler-specific rows R, C, B. They expose generated recognizers, certificate gates, and no-hidden-input boundaries as finite rows, not as host-side audit prose. The local NameCert row N names exactly this packet surface. □

Theorem 822.3 (GroundCompiler audit verdict non-escape)**.** *Every accepted* $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ *packet exports GroundCompiler audit evidence only through the finite rows*

$$M, Q, F, V, A, R, C, B, H, K, P, N.$$

It exports no hidden dependency query, no unrecorded verdict row, no recognizer outside the generated-recognizer row, no certificate outside the certificate-gate row, no proof from code or reports, no raw event-flow equality from a classifier quotient, and no host compiler input beyond the no-hidden-input boundary row.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictNonEscape`

Proof. Project the accepted packet as $G = (M, Q, F, V, A, R, C, B, H, K, P, N)$. These coordinates are the whole exported BHist surface.

The audit decision is exhausted by Q, F, V, A. The query output, forbidden row, verdict row, and audit-family alignment row are all displayed before a consumer can treat the packet as evidence.

The compiler-facing evidence is exhausted by R, C, B. Recognizers are read through the generated-recognizer row, accepted exports through the certificate-gate row, and exclusions through the no-hidden-input boundary row.

Transport, replay, provenance, and naming are exhausted by H, K, P, N. Since no remaining coordinate stores hidden query data, an alternate verdict, host proof evidence, raw classifier equality, or compiler input outside the boundary, none can escape the packet. □

Theorem 822.4 (GroundCompiler audit verdict obligation surface)**.** *The TasteGate marker*

BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictTasteGate_single_carrier_alignment

certifies that a GroundCompilerAuditVerdict$^{\uparrow}$ *packet is admitted only when the visible carrier is the single finite surface*

$$G = (M, Q, F, V, A, R, C, B, H, K, P, N).$$

For such a packet, the obligation surface consists exactly of carrier admission through those rows, forbidden-row comparison between Q and F, verdict determinacy through V, replay provenance through K and P, generated-recognizer readback through R, certificate-gate readback through C, and non-escape through the boundary row B and local naming row N.

Proof. Read the cited TasteGate marker as the carrier-alignment guard for this packet. It aligns the public GroundCompiler audit verdict name with the single displayed carrier of Definition 822.1.

The carrier-admission and comparison obligations are then forced by the coordinates. Public reads first project $M, Q, F, V, A, R, C, B, H, K, P, N$; the forbidden comparison is the displayed comparison between Q and F, and the verdict row V is the only verdict coordinate available after that comparison.

Replay and readback obligations are separate rows. The continuation row K and provenance row P record replay provenance, while R and C expose generated-recognizer and certificate-gate readback without turning either one into host audit prose.

Finally apply Definition 822.3. The no-hidden-input boundary B and local NameCert row N prevent hidden dependency queries, alternate verdicts, raw classifier equality, or compiler inputs from escaping the displayed packet. □

Theorem 822.5 (GroundCompiler audit verdict scoped handoff)**.** *For every accepted* GroundCompilerAuditVerdict$^{\uparrow}$ *carrier*

$$G = (M, Q, F, V, A, R, C, B, H, K, P, N),$$

the cited TasteGate marker

BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictTasteGate_single_carrier_alignment

forces the scoped handoff to pass through exactly the displayed rows $M, Q, F, V, A, R, C, B, H, K, P, N$. The forbidden-row comparison is read only from Q, F, V, A; generated-recognizer readback is read only from R; certificate-gate readback is read only from C; and no-hidden-input nonescape is read only from B, H, K, P, N.

Proof. Use the TasteGate marker as the single-carrier admission guard. It admits the packet only through the finite tuple of Definition 822.1, so a consumer first projects $M, Q, F, V, A, R, C, B, H, K, P, N$.

The audit comparison route is then exhausted by Q, F, V, A. The query output Q and forbidden-dependency row F produce the verdict row V, and the audit-family alignment row A records the family-facing comparison surface.

The readback route is exhausted by R and C. Generated recognizers are read through R, accepted exports through the certificate-gate row C, and no host recognizer or gate evidence is available outside those rows.

Finally the boundary route is exhausted by B, H, K, P, N. The no-hidden-input row B, componentwise transport row H, continuation row K, provenance row P, and local naming row N preserve the same displayed carrier and leave no separate input channel. □

Theorem 822.6 (GroundCompiler audit verdict boundary exactness)**.** *The rows $M, Q, F, V, A, R, C, B, H, K, P, N$ are the complete scoped route for a* GroundCompilerAuditVerdict$^{\uparrow}$ *consumer. Any accepted read of forbidden-row comparison, generated-recognizer readback, certificate-gate readback, or no-hidden-input nonescape factors through those rows and through the TasteGate marker*

BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictTasteGate_single_carrier_alignment.

Consequently the packet exports no theorem code proof, metric report proof, motif report proof, raw event-flow equality, classifier quotient equality, or compiler input beyond the displayed boundary.

Proof. Start with an accepted consumer read. The TasteGate marker aligns the public name with the single carrier, so there is no alternate carrier from which comparison, recognizer, gate, or boundary evidence can be read.

For forbidden-row comparison, the consumer must inspect Q, F, V, A. The module row M fixes the inspected surface, Q gives the displayed output, F gives the forbidden row, V records the comparison verdict, and A ties the verdict to the audit-family alignment.

For generated-recognizer and certificate-gate readback, the consumer must inspect R and C. These two rows are the only coordinates assigned to those readbacks, while H, K, P, N transport, replay, source, and name only the same displayed rows.

For nonescape, inspect B. Since B is the no-hidden-input boundary and the remaining structural rows add no analytic content, there is no exported proof from code or reports, no raw event-flow equality, no quotient classifier equality, and no compiler input outside the packet. □

Closure status for NameCert$_{\text{GroundCompilerAuditVerdict}^{\uparrow}}$**.**

Constructive story: A GroundCompilerAuditVerdict$^{\uparrow}$ packet is generated from a GroundCompiler module row, displayed query output, forbidden-dependency comparison, verdict row, audit-family alignment, generated-recognizer row, certificate-gate row, no-hidden-input boundary row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GroundCompilerAuditVerdict}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 822.1, Definition 822.2, Definition 822.3, Definition 822.4, Definition 822.5, and Definition 822.6 bind the GroundCompiler audit verdict scoped route as carrier admission, forbidden-row comparison, verdict determinacy, replay provenance, generated-recognizer readback, certificate-gate readback, and no-hidden-input nonescape backed by the TasteGate marker.

Formal verification: FormalStatus(NameCert$_{\text{GroundCompilerAuditVerdict}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GroundCompilerAuditVerdictUp.GroundCompilerAuditVerdictTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not prove global compiler self-hosting, complete environment purity, MetaCIC subject reduction, bridge checking, or any theorem content from code, reports, motifs, or classifier quotients.

Upgrade path: Public closure requires a downstream theorem showing which GroundCompiler consumer reads this scoped audit-verdict surface.

Chapter 823

A Concrete Naming Certificate for $\mathsf{AuditMapConcreteInventory}^{\uparrow}$

The $\mathsf{AuditMapConcreteInventory}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a concrete finite inventory. It sits beside the substitution audit map of chapter 779 and the family-level ledger chapters: those siblings display individual audit rows and cross-map ledgers, while this packet binds the established, conditional, obstruction, frontier, and consumer rows as one inspectable NameCert object.

Definition 823.1 (Audit-map concrete inventory carrier)**.** *An* $\mathsf{AuditMapConcreteInventory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, E, C, O, F, U, H, R, P, N)$$

with the following visible rows:

1. *M is the concrete audit-map identity row, naming the local map whose inventory is being certified;*
2. *E is the established-content row, containing checked or otherwise available rows from the local audit map;*
3. *C is the conditional-content row, containing rows whose use is routed through a named discharge or interface;*
4. *O is the obstruction row, containing no-claim or no-escape rows that must remain visible to consumers;*
5. *F is the frontier row, containing consumer-facing handoff rows that are not theorem exports;*
6. *U is the downstream consumer row, recording which sibling chapters may read the inventory and through which displayed boundary;*
7. *H records componentwise* hsame *transports for the identity, inventory, consumer, route, provenance, and local naming rows;*
8. *R records the displayed* Cont *routes by which a consumer reads each inventory row;*
9. *P is the* Pkg *provenance over chapter 9406, chapter 779, neighbouring audit-map ledgers,* BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapConcreteInventory}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are the displayed audit-map identity, established content, conditional content, obstruction content, frontier content, downstream consumer, transport, continuation, provenance, and local naming rows. It exports no closed subject-reduction theorem, no hidden discharge of a conditional row, no erasure of an obstruction row, no closure of a frontier row, and no host-level semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapConcreteInventoryUp.AuditMapConcreteInventoryTasteGate single carrier alignment`

Theorem 823.2 (Audit-map concrete inventory NameCert obligations)**.** *For the carrier of Definition 823.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapConcreteInventory}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$I = (M, E, C, O, F, U, H, R, P, N),$$

audit-map identity readback through M, established-row readback through E, conditional-row readback through C, obstruction-row readback through O, frontier-row readback through F, downstream-consumer readback through U, componentwise transport through H, displayed continuation through R, package provenance through P, and local naming through N.

Proof. Project the finite packet I. The identity row M fixes which concrete audit map is being read before any inventory row can be consumed.

Read the inventory rows E, C, O, F. These rows expose the family template of chapter 9406: established content, conditional content, obstruction content, and frontier content are separate coordinates rather than prose classes.

Finally inspect U, H, R, P, N. The consumer row names downstream readers, transport and continuation replay the displayed coordinates, provenance records the sibling sources, and local naming binds the packet. Hence the NameCert surface is exhausted by the finite carrier. □

Theorem 823.3 (Audit-map concrete inventory ledger exactness)**.** *For every accepted* $\mathsf{AuditMapConcreteInventory}^{\uparrow}$ *carrier $I = (M, E, C, O, F, U, H, R, P, N)$, the concrete inventory ledger has exactly five public row classes: established rows E, conditional rows C, obstruction rows O, frontier rows F, and downstream consumer rows U. Transport, continuation, provenance, and local naming preserve these classes as finite BEDC data and do not introduce a sixth audit class.*

Proof. The carrier definition displays the five public row classes as E, C, O, F, U. The identity row M selects the local map, but it is not another inventory class.

The transport and route rows H, R only replay componentwise sameness and displayed continuations over the selected rows. They therefore preserve the five-class ledger rather than extending it.

The provenance and local naming rows P, N certify where the inventory is read from and under which local name. Since neither row contains a theorem-producing or class-producing coordinate, the ledger is exact at the five displayed classes. □

Theorem 823.4 (Audit-map concrete inventory non-escape)**.** *For every accepted* $\mathsf{AuditMapConcreteInventory}^{\uparrow}$ *carrier $I = (M, E, C, O, F, U, H, R, P, N)$, every consumer read remains inside the displayed audit-map identity, established, conditional, obstruction, frontier, downstream consumer,* hsame, Cont, Pkg, *and local* NameCert *rows. In particular, the packet cannot turn conditional content into established content, erase an obstruction row, close a frontier row, create an unlisted downstream consumer, or export host-level semantic equality.*

Proof. Start with the carrier projection. A consumer can read the concrete inventory only through the rows $M, E, C, O, F, U, H, R, P, N$.

Use the NameCert obligations of Definition 823.2. They bind each row class to its displayed coordinate and route every consumer read through the continuation row R.

Apply the ledger exactness of Definition 823.3. Since the inventory has exactly the five public row classes and no hidden class-producing coordinate, each excluded endpoint remains outside the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapConcreteInventory}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapConcreteInventory}^{\uparrow}$ packet is generated from a concrete audit-map identity row, established rows, conditional rows, obstruction rows, frontier rows, downstream consumer rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapConcreteInventory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 823.1, Definition 823.2, Definition 823.3, and Definition 823.4 bind the concrete audit-map inventory and its non-escape boundary as finite NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapConcreteInventory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed consistency, ambient confluence, hidden discharge routes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-class readback, consumer-row exactness, transport, route replay, provenance, naming, and non-escape for the same concrete inventory packet.

Chapter 824

A Concrete Naming Certificate for SubstitutionAuditWindowRoute$^{\uparrow}$

The SubstitutionAuditWindowRoute$^{\uparrow}$ packet realizes the closed-window route implicit in the MetaCIC substitution audit map of chapter 9380. It sits beside the substitution audit map of chapter 779 and the MetaCIC substitution audit packet of **??**: those chapters expose the audit rows and packet inventory, while this chapter records the finite route through which a consumer reads closed shift, closed substitution, closed composition, generator closure, and the subject-reduction handoff boundary without treating the handoff as a proof of subject reduction.

Definition 824.1 (Substitution audit window route carrier)**.** *A* SubstitutionAuditWindowRoute$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, D, L, M, C, G, B, H, R, P, N)$$

with the following visible rows:

1. *T is the MetaCIC* Term *row inspected by the substitution audit;*
2. *D is the* ClosedAt *window at which the term is read;*
3. *L is the closed shift row, recording the* shift_closed *route under D;*
4. *M is the closed substitution row, recording the* substitute_closed *route under D;*
5. *C is the closed composition row for the available closed substitution compositions, including the closed-at-depth and closed-at-zero composition surfaces;*
6. *G is the generator-closure classifier row for closed-under-shift, closed-under-substitution, and the closed composition route;*
7. *B is the subject-reduction handoff row, naming the beta-substitution and dependent-typing boundary without discharging it;*
8. *H records componentwise* hsame *transports for term, window, movement, composition, classifier, handoff, route, provenance, and naming rows;*
9. *R records the displayed* Cont *routes by which a consumer inspects the rows in the listed order;*
10. *P is the* Pkg *provenance over the substitution audit map, the sibling substitution audit packet,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose coordinates are the displayed term, closed-window, closed shift, closed substitution, closed composition, generator-closure, handoff, transport, continuation, provenance, and local naming rows. It exports no unrestricted substitution calculus, no beta-substitution preservation theorem, no closed subject-reduction theorem, no quotient equality, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.SubstitutionAuditWindowRouteUp.SubstitutionAuditWindowRouteTasteGate`
`single carrier alignment`

Theorem 824.2 (Substitution audit window route NameCert obligations)**.** *For the carrier of Definition 824.1, the local* $\mathsf{NameCert}_{\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$W = (T, D, L, M, C, G, B, H, R, P, N),$$

closed-window readback through T, D, closed shift readback through L, closed substitution readback through M, closed-composition readback through C, generator-closure classification through G, subject-reduction handoff exposure through B, componentwise transport through H, displayed continuation through R, package provenance through P, and local naming through N.

Proof. Carrier admission is the displayed finite BHist packet. Projecting T and D exposes the audited term together with the closed window at which public substitution reads are admitted.

Read the movement rows next. The row L records the closed shift route, M records the closed substitution route, and C records the closed composition route. These are exactly the closed rows named by chapter 9380.

Read G and B after the movement rows. The generator-closure row classifies the shift, substitute, and composition routes as closed-window generator data, while the handoff row records the subject-reduction boundary as a boundary row rather than as typed preservation.

Finally inspect H, R, P, N. The transport, continuation, provenance, and local naming rows replay the same finite carrier. Since the packet has no coordinate for unrestricted substitution, beta-substitution preservation, closed subject reduction, quotient equality, or host semantic equality, the NameCert surface is exhausted by the displayed rows. □

Theorem 824.3 (Substitution audit window route closed composition handoff)**.** *For every accepted* $\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}$ *carrier $W = (T, D, L, M, C, G, B, H, R, P, N)$, any consumer read of the subject-reduction handoff row B first factors through the closed window D, the closed shift row L, the closed substitution row M, the closed composition row C, and the generator-closure row G. The handoff may cite those rows as syntax-side closure data, but it cannot export beta-substitution preservation, dependent typing preservation, or a closed subject-reduction theorem.*

Proof. Start with the carrier order. The handoff row B is listed after the closed movement and classifier rows, so an accepted route to B has already selected D, L, M, C, G.

Use the NameCert obligations of Definition 824.2. They route closed-window readback through D, closed movement through L, M, closed composition through C, and generator closure through G before the handoff row can be consumed.

The rows H and R preserve that order under componentwise transport and displayed continuation. Provenance through P points back to chapter 779 and **??**, while the local name N binds the route as a finite packet.

It remains only to record the boundary. The carrier has no coordinate for beta-substitution preservation, dependent typing preservation, unrestricted substitution commutation, or a closed subject-reduction theorem. Therefore the handoff is readable as a named downstream boundary, not as a discharge of that boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}$ packet is generated from a MetaCIC term row, a ClosedAt window, closed shift and substitution rows, closed composition rows, generator-closure classifier rows, a subject-reduction handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 824.1, Definition 824.2, and Definition 824.3 bind the finite closed-window route from substitution audit rows to the subject-reduction handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstitutionAuditWindowRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close unrestricted substitution, beta-substitution preservation, dependent typing preservation, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-window readback, closed movement readback, closed-composition readback, generator-closure readback, handoff exposure, transport, continuation, provenance, naming, and boundary non-escape for the same finite packet.

Chapter 825

A Concrete Naming Certificate for $\mathsf{AuditMapTemplatePacket}^{\uparrow}$

The $\mathsf{AuditMapTemplatePacket}^{\uparrow}$ carrier realizes the audit-map family methodology of chapter 9406 as a finite packet template for concrete audit rows. It sits beside the substitution audit map of chapter 779: the sibling map supplies a local MetaCIC use case, while this packet records the reusable positive, conditional, obstruction, frontier, and sibling-use rows that such maps expose without silently closing their boundaries.

Definition 825.1 (Audit-map template packet carrier)**.** *An* $\mathsf{AuditMapTemplatePacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, P, C, O, F, S, H, R, K, N)$$

with the following displayed rows:

1. *U is the audit-map use row naming the local substrate, consumer interface, and map family cell being templated;*
2. *P is the positive row listing facts, carrier rows, or checked local inventory that the map may expose directly;*
3. *C is the conditional row listing setup fields, routes, or certificate obligations that must remain explicit before their conclusions are read;*
4. *O is the obstruction row naming blocked routes, missing inhabitants, shape restrictions, or no-go facts;*
5. *F is the frontier row naming the strongest nearby claim visible to consumers but not supplied as closed content by the packet;*
6. *S is the sibling-use row naming concrete audit-map neighbours, including the substitution audit map of chapter 779, that may consume or supply the displayed rows;*
7. *H records componentwise* hsame *transports for the use, positive, conditional, obstruction, frontier, sibling-use, route, provenance, and local naming rows;*
8. *R records displayed* Cont *routes by which consumers inspect U, P, C, O, F, S without adding hidden coordinates;*
9. *K is the* Pkg *provenance over chapter 9406, the sibling concrete packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{AuditMapTemplatePacket}^{\uparrow}}$ *row.*

The classifier accepts exactly packets carrying these ten rows. It exports no proof of F, no erased obstruction, no discharged conditional row, no hidden sibling route, no closed MetaCIC theorem, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapTemplatePacketUp.AuditMapTemplatePacketTasteGate single carrier alignment`

Theorem 825.2 (Audit-map template packet NameCert obligations)**.** *For the carrier of Definition 825.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapTemplatePacket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (U, P, C, O, F, S, H, R, K, N),$$

use-row readback through U, positive-row readback through P, conditional-row readback through C, obstruction-row readback through O, frontier-row readback through F, sibling-use readback through S, componentwise transport through H, displayed continuation through R, provenance through K, and local naming through N.

Proof. Carrier admission is the displayed finite packet. Projecting U, P, C, O exposes the local audit use, directly licensed content, explicit conditional content, and obstruction inventory as separate rows.

Project F and S next. The frontier row records where the packet stops, while the sibling-use row records which neighbouring maps may read or supply displayed rows, including the substitution audit map.

Finally inspect H, R, K, N. These rows transport, replay, source, and name the same packet. Since every obligation is one of the displayed rows and no transport, continuation, provenance, or naming coordinate adds a hidden row, the local NameCert surface is exactly the finite carrier inventory. □

Theorem 825.3 (Audit-map template packet row frontier)**.** *For every accepted* $\mathsf{AuditMapTemplatePacket}^{\uparrow}$ *carrier $A = (U, P, C, O, F, S, H, R, K, N)$, every consumer read of positive content, conditional content, obstruction content, frontier content, or sibling use factors through the displayed rows P, C, O, F, S and their supporting rows H, R, K, N. A consumer may cite P as licensed local inventory, cite C as an explicit condition, cite O as an obstruction, cite F as a frontier boundary, and cite S as a sibling-use ledger; it cannot turn C into a discharged theorem, erase O, prove F, or read a sibling map outside S.*

Proof. Start from the carrier projection. The rows P, C, O, F, S are the only content rows available after the use row U has fixed the local audit-map cell.

Use the NameCert obligations of Definition 825.2. Componentwise hsame transport in H and displayed continuation in R preserve the same five content rows rather than replacing them by stronger theorem content.

It remains to check the boundary. Provenance K and local naming N locate the packet and its sibling references, but neither row supplies a discharge witness for C, an erasure operation for O, a proof object for F, or an unlisted sibling route beyond S. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapTemplatePacket}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapTemplatePacket}^{\uparrow}$ packet is generated from an audit-map use row, positive rows, conditional rows, obstruction rows, frontier rows, sibling-use rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapTemplatePacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 825.1, Definition 825.2, and Definition 825.3 bind the finite audit-map template packet and its visible row frontier.

Formal verification: FormalStatus(NameCert$_{\text{AuditMapTemplatePacket}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.AuditMapTemplatePacketUp.AuditMapTemplatePacketTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed MetaCIC, proof of frontier rows, discharge of conditional rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row readback, sibling-use routing, classifier transport, provenance, local naming, and non-escape for the same finite packet.

Chapter 826

A Concrete Naming Certificate for $\mathsf{AuditMapObstructionSocket}^{\uparrow}$

The $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a finite obstruction and frontier socket. It sits beside the family ledger of chapter 810 and the MetaCIC open-frontier ledger of chapter 762: those chapters expose family rows and open frontier rows, while this socket names the local handoff surface by which positive content, conditional interfaces, obstruction rows, and frontier rows remain visible to a NameCert consumer without becoming hidden closure.

Definition 826.1 (Audit-map obstruction socket carrier)**.** *An* $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, P, I, O, F, H, C, K, N)$$

with the following visible rows:

1. *A is the local audit-map tag naming the module, theorem cluster, or certificate family whose boundary is being inspected;*
2. *P records positive-content rows already available inside the local audit map;*
3. *I records conditional-interface rows, including hypotheses, discharge sockets, and named consumer preconditions that must be supplied before a neighbouring map may read the row as positive content;*
4. *O records obstruction rows whose non-discharge must remain visible to the audit-map family;*
5. *F records frontier rows handed to downstream consumers as named boundaries rather than closed theorem surfaces;*
6. *H records componentwise* hsame *transports for the audit tag, positive rows, conditional rows, obstruction rows, frontier rows, continuation rows, provenance rows, and local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the positive, conditional, obstruction, and frontier rows without bypassing their row class;*
8. *K is the* Pkg *provenance over the audit-map family ledger, the open-frontier ledger, and the local concrete chapter supplying the socket rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapObstructionSocket}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose coordinates are the displayed audit tag, positive-content rows, conditional-interface rows, obstruction rows, frontier rows, componentwise transports, continuation routes, provenance row, and local naming row. It exports no hidden proof of an obstruction row, no unlisted discharge of a conditional interface, no closed frontier theorem, and no theorem stronger than the positive rows already display.

Lean4 ✓ `BEDC.Derived.AuditMapObstructionSocketUp.AuditMapObstructionSocketTasteGate single carrier alignment`

Theorem 826.2 (Audit-map obstruction socket NameCert obligations)**.** *For the carrier of Definition 826.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapObstructionSocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (A, P, I, O, F, H, C, K, N),$$

audit-tag readback through A, positive-content readback through P, conditional-interface readback through I, obstruction readback through O, frontier readback through F, componentwise transport through H, displayed continuation through C, package provenance through K, and local naming through N.

Lean4 ✓ `BEDC.Derived.AuditMapObstructionSocketUp.AuditMapObstructionSocketNameCertObligations`

Proof. Project the finite packet S. The audit tag A fixes the local map being inspected before any row class is read.

Read the row classes next. Positive content is exposed by P, conditional interfaces are exposed by I, obstructions are exposed by O, and consumer-facing frontier rows are exposed by F. These rows are separate coordinates of the carrier.

Finally inspect H, C, K, N. Transport and continuation replay the same displayed row classes, the provenance row cites the family ledger, open-frontier ledger, and local chapter supplying the rows, and the local name binds the packet. Hence the NameCert surface is exhausted by the stated coordinates. □

Lean4 ✓ `BEDC.Derived.AuditMapObstructionSocketUp.AuditMapObstructionSocketCarrier namecert obligations`

Theorem 826.3 (Audit-map obstruction socket non-escape)**.** *For every accepted* $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ *carrier $S = (A, P, I, O, F, H, C, K, N)$, every consumer read remains inside the displayed audit tag, positive-content rows, conditional-interface rows, obstruction rows, frontier rows,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. In particular, the socket cannot turn an obstruction row into a proof, a conditional-interface row into an unlisted discharge, or a frontier row into a closed theorem.*

Proof. Start from the carrier projection. The only public coordinates are $A, P, I, O, F, H, C, K, N$, so a consumer read must enter through one of the displayed row classes or through its transport, route, provenance, and naming rows.

Apply the NameCert obligations of Definition 826.2. They bind positive, conditional, obstruction, and frontier reads to their own coordinates and require every replay to use H and C.

The provenance row then blocks hidden closure. Since K cites only the visible family ledger, open-frontier ledger, and local concrete chapter, a consumer may cite the socket as finite audit data but cannot extract a proof or discharge route absent from S. □

Theorem 826.4 (Audit-map obstruction socket frontier handoff)**.** *For every accepted* $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ *carrier, a downstream consumer may receive a frontier row only through the displayed row* F *together with the transport, continuation, provenance, and local naming rows* H, C, K, N*. The handoff preserves the distinction between positive content* P*, conditional interfaces* I*, obstruction rows* O*, and frontier rows* F*.*

Proof. Read the frontier coordinate of Definition 826.1. The row F is the only carrier coordinate that names downstream frontier material.

Use the componentwise transport and continuation rows. The rows H and C may replay the frontier coordinate and its provenance, but they do not recompute F from positive content, conditional interfaces, or obstruction rows.

Apply non-escape from Definition 826.3. The handoff therefore supplies a named frontier boundary with K and N, not a closed theorem and not an erased obstruction or conditional discharge. □

Lean4 ✓ `BEDC.Derived.AuditMapObstructionSocketUp.AuditMapObstructionSocket frontier handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapObstructionSocket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ packet is generated from an audit tag, positive-content rows, conditional-interface rows, obstruction rows, frontier rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapObstructionSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 826.1, Definition 826.2, Definition 826.3, and Definition 826.4 bind the finite audit-map obstruction socket and its frontier handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapObstructionSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed consistency, ambient confluence, hidden discharge routes, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-row readback, conditional-interface readback, obstruction readback, frontier handoff, transport, continuation, provenance, naming, and non-escape for the same finite packet.

Chapter 827

A Concrete Naming Certificate for GroundCompilerAuditPacket$^{\uparrow}$

The GroundCompilerAuditPacket$^{\uparrow}$ carrier realizes the compiler-layer audit map of chapter 9407 as a finite naming certificate. It sits beside the source-channel ledger of chapter 671: that sibling packet records the acceptance-facing channel around kernel declarations, while this packet records the GroundCompiler event-flow, source-channel, recognizer, certificate-gate, report-boundary, and no-hidden-input rows that feed compiler-audit consumers.

Definition 827.1 (GroundCompiler audit packet carrier)**.** *A* GroundCompilerAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, S, R, C, Q, M, B, X, H, T, P, N)$$

with the following displayed rows:

1. *E is the GroundCompiler event-flow row naming the formal event input and its finite flow image;*
2. *S is the source-channel row recording legal encoding, decoding, and source/channel separation for the same flow image;*
3. *R is the generated recognizer row for package, name-certificate, derivation-certificate, theorem, chapter, or motif-facing classifier output;*
4. *C is the certificate-gate row through which recognized compiler artifacts become export candidates;*
5. *Q is the report-boundary row separating metric reports, motif reports, theorem code, chapter code, and source reports from proof or certificate evidence;*
6. *M is the module-inventory row naming the GroundCompiler submodule ledger used by chapter 9407;*
7. *B is the no-hidden-input row grounded by the streaming compiler theorem and its legal event-flow, source-channel, recognizer, gate, quotient-limit, source-separation, metric-conservativity, self-hosting, bootstrap, and global-conservativity clauses;*
8. *X is the source-channel export row linking this packet to chapter 671 without identifying compiler evidence with kernel acceptance evidence;*
9. *H records componentwise* hsame *transports for event-flow, channel, recognizer, gate, report, module, no-hidden-input, export, route, provenance, and local naming rows;*

10. *T records displayed* Cont *routes by which compiler-audit consumers inspect E, S, R, C, Q, M, B, X in finite order;*
11. *P is the* Pkg *provenance over chapter 9407, the sibling source-channel ledger,* GroundCompiler*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for host compiler trust, hidden host input, source text outside the legal channel, report-as-proof conversion, motif-as-object identity, raw equality beyond classifier limits, kernel acceptance evidence, or MetaCIC proof evidence.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketTasteGate single carrier alignment`

Definition 827.2 (GroundCompiler audit packet classifier)**.** *The* $\mathsf{GroundCompilerAuditPacket}^{\uparrow}$ *classifier accepts exactly the carriers of Definition 827.1 whose rows satisfy the GroundCompiler audit-map reading of chapter 9407: the event-flow row is formal and finite, the source-channel row is legal and separated, generated recognizers mediate recognition evidence, certificate gates mediate exports, report rows remain non-evidence rows, the module-inventory row names the inspected GroundCompiler ledger, the no-hidden-input row is read through the streaming compiler theorem, and the source-channel export row is routed through the sibling kernel source-channel ledger.*

It rejects any packet that reads report output as proof, treats source substrings as decoded event data without the legal channel row, treats classifier quotient data as raw equality, imports kernel acceptance evidence into the compiler packet, or imports MetaCIC proof evidence into the compiler packet.

Lean4 (definition) `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketClassifier`

Theorem 827.3 (GroundCompiler audit packet NameCert obligations)**.** *For every accepted* GroundCompilerAuditP *carrier*

$$G = (E, S, R, C, Q, M, B, X, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$ *surface consists exactly of event-flow readback through E, source-channel readback through S, recognizer readback through R, certificate-gate readback through C, report-boundary readback through Q, module-inventory readback through M, no-hidden-input readback through B, source-channel export readback through X, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketNameCertObligations`

Proof. Start with the carrier projection. The rows $E, S, R, C, Q, M, B, X, H, T, P, N$ are the only public coordinates available to the local certificate.

Read the compiler evidence rows first. The event-flow row E supplies the finite formal input surface, S supplies legal source-channel encoding and separation, R supplies generated recognition evidence, and C supplies the certificate-mediated export gate.

Read the boundary rows next. The report row Q records which report-like artifacts are not proof or certificate evidence. The module row M anchors the packet in the inspected GroundCompiler ledger, while B records the no-hidden-input streaming compiler surface grounded by the checked compiler theorem.

Finally inspect X, H, T, P, N. The export row links the compiler packet to the source-channel ledger without turning that ledger into compiler proof evidence. Transport, continuation, provenance, and naming replay the same finite rows, so the local NameCert obligations are exhausted by the displayed packet. □

Theorem 827.4 (GroundCompiler audit packet export non-escape)**.** *Every public export from an accepted* GroundCompilerAuditPacket$^{\uparrow}$ *packet factors through the displayed event-flow, source-channel, recognizer, certificate-gate, report-boundary, module-inventory, no-hidden-input, source-channel export, transport, continuation, provenance, and local naming rows of Definition 827.1. The export surface carries no hidden compiler input, no host source scan, no report-as-proof conversion, no motif-as-object identity, no classifier-quotient collapse into raw equality, and no kernel acceptance claim outside the sibling ledger.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketExportNonEscape`

Proof. Project an accepted carrier G. By Definition 827.3, every public export must pass through the listed NameCert rows.

The compiler-facing rows E, S, R, C account for event inputs, source channels, generated recognizers, and certificate gates before anything is exported. The report row Q keeps reports, motifs, theorem code, and chapter code on the non-evidence side of the boundary.

The no-hidden-input row B supplies the streaming compiler audit surface, and the export row X routes source-channel consumers toward the sibling kernel source-channel ledger. Neither row adds a host compiler trust coordinate or a kernel acceptance proof coordinate to this packet.

Thus the only export path is the finite displayed path through H, T, P, N. Since no carrier field names hidden input, host source scans, report-to-proof conversion, motif identity, raw equality collapse, or kernel acceptance beyond X, those routes cannot be read from the packet. □

Theorem 827.5 (GroundCompiler audit packet MetaCIC separation)**.** *An accepted* GroundCompilerAuditPacket$^{\uparrow}$ *packet may cite the compiler-layer audit map of chapter 9407 and the source-channel ledger of chapter 671, but it does not export MetaCIC reduction, typing, confluence, normalization, closure-preservation, or subject-reduction evidence. Any consumer requiring MetaCIC proof evidence must read it from the relevant MetaCIC packet rather than from the GroundCompiler audit packet.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketMetaCICSeparation`

Proof. Separate the cited origins. The origin chapter chapter 9407 records compiler-layer facts: event-flow conservativity, legal channels, recognizer generatedness, certificate-mediated exports, report boundaries, and no-hidden-input streaming compilation.

The sibling ledger chapter 671 records source-channel rows around kernel acceptance events. Its use by X is a channel-routing use, not a transfer of kernel acceptance or MetaCIC proof evidence into the GroundCompiler packet.

Now inspect the carrier of Definition 827.1. None of $E, S, R, C, Q, M, B, X, H, T, P, N$ is a MetaCIC term, reduction, typing, confluence, normalization, closure-preservation, or subject-reduction proof row.

Therefore a MetaCIC consumer has no projection from this packet to the proof evidence it needs. It must instead read a dedicated MetaCIC packet, while the GroundCompiler packet remains a finite compiler-audit certificate. □

Theorem 827.6 (GroundCompiler audit packet carrier exposure)**.** *Every accepted* GroundCompilerAuditPacket$^{\uparrow}$ *packet exposes exactly the finite carrier rows*

$$E, S, R, C, Q, M, B, X, H, T, P, N.$$

These rows are the BHist *event-flow, source-channel, recognizer, certificate-gate, report-boundary, module-inventory, no-hidden-input, source-channel export, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows of the packet, and no additional compiler trust, host source scan, kernel acceptance evidence, or MetaCIC proof evidence row is exposed.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketCarrierExposure`

Proof. Project an accepted carrier $G = (E, S, R, C, Q, M, B, X, H, T, P, N)$ from Definition 827.1. The carrier is a finite BHist packet, so the only rows available for public inspection are the displayed coordinates.

Apply Definition 827.3. It identifies the local NameCert surface with exactly those event-flow, channel, recognizer, gate, report, module, no-hidden-input, export, hsame, Cont, Pkg, and naming rows.

The classifier of Definition 827.2 rejects packets that read reports as proof, treat source substrings as decoded event data outside the legal channel, import kernel acceptance evidence, or import MetaCIC proof evidence. Hence no extra row can be exposed by an accepted packet. □

Theorem 827.7 (GroundCompiler audit packet query replay)**.** *Every no-hidden-input query and every report-boundary read replayed from an accepted* GroundCompilerAuditPacket$^{\uparrow}$ *packet must pass through the rows* B, Q, M, T, P, N*. The replay reads* B *for the no-hidden-input* BHist *surface,* Q *for report separation,* M *for the inspected module ledger,* T *for displayed* Cont *order,* P *for* Pkg *provenance, and* N *for local* NameCert *naming, with all componentwise* hsame *transport confined to the exposed carrier rows.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketQueryReplay`

Proof. Start from the accepted carrier and isolate the rows that a query replay can inspect. The no-hidden-input claim is named only by B, and the report-boundary claim is named only by Q.

The module row M supplies the inspected GroundCompiler ledger for both reads. The continuation row T fixes the finite order in which a consumer can inspect B and Q, while P and N record provenance and local naming for the same finite packet.

By Definition 827.4, public exports cannot introduce hidden compiler input, host source scans, or report-as-proof conversions. Thus query replay is forced through B, Q, M, T, P, N, with hsame transport used only to preserve the displayed carrier rows. □

Theorem 827.8 (GroundCompiler audit packet refusal separation)**.** *In an accepted* GroundCompilerAuditPacket$^{\uparrow}$ *packet, report rows, motif rows, host source scans, and kernel acceptance evidence cannot be exported as compiler proof evidence. The* BHist *carrier records report separation through* Q*, generated classifier and motif-facing recognition through* R*, no-hidden-input refusal through* B*, source-channel routing through* S *and* X*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* T*,* Pkg *provenance through* P*, and local* NameCert *naming through* N*; none of these rows converts a report, motif, host scan, or kernel acceptance row into proof evidence.*

Proof. Inspect the report and recognition rows first. The row Q places metric reports, motif reports, theorem code, chapter code, and source reports on the report-boundary side, while R records generated recognizer output rather than a proof object.

Next inspect the channel and refusal rows. The source-channel row S and the export row X route legal source-channel information, and B records the no-hidden-input boundary. These rows do not add a host source scan coordinate or a kernel acceptance proof coordinate to the compiler packet.

Finally apply Definition 827.5 together with Definition 827.6. The only remaining rows are H, T, P, N, which preserve, replay, package, and name the same finite carrier; they do not turn report rows, motif rows, host scans, or kernel acceptance evidence into compiler proof evidence. □

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketRefusalSeparation`

Theorem 827.9 (GroundCompiler audit packet obligation closure)**.** *For every accepted* GroundCompilerAuditPacke *carrier, the carrier row of Definition 827.1, the classifier of Definition 827.2, the NameCert obligations of Definition 827.3, the export nonescape theorem of Definition 827.4, the MetaCIC separation theorem of Definition 827.5, the carrier exposure theorem of Definition 827.6, the query replay theorem of Definition 827.7, and the refusal separation theorem of Definition 827.8 assemble the complete* obligation *surface for* $\mathsf{GroundCompilerAuditPacket}^{\uparrow}$.

The assembled surface exposes only the finite GroundCompiler audit packet rows. Report rows, motif rows, host scans, kernel acceptance evidence, and MetaCIC proof evidence remain outside the packet and cannot be projected from the local $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditPacketUp.GroundCompilerAuditPacketObligationClosure`

Proof. Begin with the accepted carrier. Apply Definition 827.1 and Definition 827.2 to identify the finite rows and the admissible packet boundary.

Next compose the certificate-facing obligations. The NameCert theorem reads back the event-flow, source-channel, recognizer, certificate-gate, report-boundary, module-inventory, no-hidden-input, export, transport, continuation, provenance, and local naming rows. The export nonescape theorem then prevents those rows from acquiring hidden compiler input, host source scans, report-as-proof conversion, motif identity, raw equality collapse, or kernel acceptance evidence outside the sibling source-channel route.

The MetaCIC separation, carrier exposure, query replay, and refusal separation theorems supply the remaining boundary clauses. They keep MetaCIC proof evidence in the relevant MetaCIC packet, expose only $E, S, R, C, Q, M, B, X, H, T, P, N$, force no-hidden-input and report-boundary queries through B, Q, M, T, P, N, and prevent report rows, motif rows, host scans, and kernel acceptance rows from becoming compiler proof evidence.

Thus every public obligation of the GroundCompiler audit packet is accounted for by the displayed finite rows, and every excluded evidence class remains outside the packet. This is exactly the obligation-closure surface claimed for $\mathsf{GroundCompilerAuditPacket}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GroundCompilerAuditPacket}^{\uparrow}$ packet is generated from a GroundCompiler event-flow row, source-channel row, generated recognizer row, certificate-gate row, report-boundary row, module-inventory row, no-hidden-input streaming compiler row, source-channel export row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$, obligation)

Scope closed: Definition 827.1, Definition 827.2, Definition 827.3, Definition 827.9, and the checked TasteGate alignment row bind the obligation surface for the finite compiler-audit packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GroundCompilerAuditPacket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove a self-hosted compiler tower, host compiler trust, report-to-proof conversion, kernel acceptance, MetaCIC reduction or typing theorems, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding each obligation to the GroundCompiler event-flow, legal source-channel, generated recognizer, no-hidden-input, and source-channel ledger scopes.

Chapter 828

A Concrete Naming Certificate for GroundCompilerRecognizerBoundary$^{\uparrow}$

The GroundCompilerRecognizerBoundary$^{\uparrow}$ packet realizes the recognizer-facing boundary of the GroundCompiler audit map in chapter 9407 as a finite NameCert carrier. It sits beside the audit-map family certificate of chapter 807 and the obstruction socket of chapter 826: those chapters expose family-level row classes and obstruction handoffs, while this packet binds the GroundCompiler event-flow, source-channel, recognizer, certificate-gate, metric, motif, and bootstrap rows without turning generated code or reports into proof.

Definition 828.1 (GroundCompiler recognizer-boundary carrier)**.** *A* GroundCompilerRecognizerBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, S, R, Q, M, V, B, H, C, P, N)$$

with the following visible rows:

1. *E is the GroundCompiler event-flow row naming the source-like event packet under inspection;*
2. *S is the source-channel row recording the encoded, decoded, and lossless channel evidence available to the compiler layer;*
3. *R is the generated recognizer row that classifies the event-flow packet without treating raw source identity as proof identity;*
4. *Q is the certificate-gate row through which accepted exports must pass before a downstream theorem or NameCert consumer may cite them;*
5. *M is the metric-report row, readable only as compiler-layer measurement data;*
6. *V is the motif-report row, readable only as compiler-layer similarity or pattern data;*
7. *B is the bootstrap-evidence row recording the finite self-compilation or hierarchy evidence actually displayed;*
8. *H records componentwise* hsame *transports for the event-flow, source-channel, recognizer, gate, metric, motif, bootstrap, continuation, provenance, and naming rows;*
9. *C records the displayed* Cont *routes by which consumers read the recognizer and certificate-gate rows without bypassing the gate;*

10. *P is the* Pkg *provenance over chapter 9407, chapter 807, and the local GroundCompiler boundary rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are the displayed event-flow, source-channel, recognizer, certificate-gate, metric, motif, bootstrap, transport, continuation, provenance, and local naming rows. It exports no theorem from generated code alone, no proof from a metric report, no proof from a motif report, no raw event-flow equality in place of recognizer classification, and no unconditional self-hosting tower.

Lean4 (definition) `BEDC.Derived.GroundCompilerRecognizerBoundaryUp.GroundCompilerRecognizerBoundaryUp`

Theorem 828.2 (GroundCompiler recognizer-boundary NameCert obligations)**.** *For the carrier of Definition 828.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $G = (E, S, R, Q, M, V, B, H, C, P, N)$;*
2. *event-flow and source-channel readback through E and S before any recognizer row is consumed;*
3. *recognizer and certificate-gate exactness through R and Q, with every accepted export routed through the gate;*
4. *report boundary separation for M, V, B, so metric, motif, and bootstrap rows remain evidence rows rather than theorem rows;*
5. *transport, continuation, provenance, and naming replay through H, C, P, N.*

Proof. Project the finite packet G. The event-flow row E and source-channel row S are visible before the recognizer row is read, so the packet separates source-like compiler input from recognizer evidence.

Read the recognizer and gate rows next. The row R records the generated recognizer result, while Q records the certificate gate through which accepted exports pass. A consumer cannot skip Q, because no alternate accepted-export coordinate occurs in the carrier.

Now inspect the report rows. The metric row M, motif row V, and bootstrap row B are separate evidence rows. They may be cited as displayed compiler-layer data, but they are not theorem rows and do not discharge a missing proof obligation.

Finally project H, C, P, N. These rows transport the same components, replay the displayed consumer routes, cite the GroundCompiler audit map and audit-map family sibling as provenance, and bind the local name. Hence the NameCert surface is exhausted by the five listed obligations. □

Theorem 828.3 (GroundCompiler recognizer-boundary non-escape)**.** *For every accepted* GroundCompilerRecognize *carrier $G = (E, S, R, Q, M, V, B, H, C, P, N)$, every compiler-facing or theorem-facing consumer read remains inside the displayed event-flow, source-channel, recognizer, certificate-gate, metric-report, motif-report, bootstrap-evidence, transport, continuation, provenance, and local naming rows. In particular, generated theorem code is not read as proof, recognizer classification is not raw event-flow equality, and report evidence is not certificate evidence.*

Proof. Start from the carrier projection. The packet lists only $E, S, R, Q, M, V, B, H, C, P, N$, so every consumer read must enter through one of those displayed rows.

Apply the NameCert obligations of Definition 828.2. They force recognizer reads to pass through E, S, R and accepted-export reads to pass through Q, while report reads stay in M, V, B.

The provenance and route rows then block hidden strengthening. Since P cites the GroundCompiler audit map and its audit-map family sibling only as visible row sources, and since C replays only displayed routes, no consumer can extract proof, raw identity, or an unconditional compiler tower from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}$ packet is generated from event-flow, source-channel, recognizer, certificate-gate, metric-report, motif-report, bootstrap-evidence, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 828.1, Definition 828.2, and Definition 828.3 bind the recognizer-facing GroundCompiler evidence boundary as a finite NameCert carrier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close self-hosting for every compiler candidate, turn generated theorem code into proof, identify recognizer classes with raw event-flow equality, provide Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-channel readback, recognizer exactness, certificate-gate routing, report separation, bootstrap evidence readback, route replay, provenance, naming, and non-escape over the same finite packet.

Chapter 829

A Concrete Naming Certificate for GroundCompilerEventFlowAudit$^{\uparrow}$

The GroundCompilerEventFlowAudit$^{\uparrow}$ packet turns the GroundCompiler audit map of chapter 9407 into a finite NameCert surface for event-flow and channel packets. It sits beside the audit-map interface of chapter 806 and the obstruction socket of chapter 826: those chapters name generic audit rows, while this packet names the GroundCompiler rows that keep source events, channel encodings, generated recognizers, and certificate gates separated from hidden compiler input.

Definition 829.1 (GroundCompiler event-flow audit carrier)**.** *A* GroundCompilerEventFlowAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, F, C, L, R, G, A, H, K, P, N)$$

with the following displayed rows:

1. *S is the source-event row naming the formal GroundCompiler input stream before channel encoding;*
2. *F is the event-flow row obtained from S by the displayed source normalizer and flow construction;*
3. *C is the channel packet row naming the legal encoding and decoding route for F;*
4. *L is the losslessness row recording that channel decoding returns the displayed event flow rather than a host-side replacement;*
5. *R is the source/channel separation row, keeping source events, channel packets, and carried metadata in distinct audit coordinates;*
6. *G is the generated-recognizer row for packages, NameCert rows, derivation certificates, theorem identifiers, and chapter flows consumed by the compiler layer;*
7. *A is the certificate-gate row through which accepted generated exports are read;*
8. *H records componentwise* hsame *transports for the source row, event flow, channel packet, losslessness row, separation row, generated recognizer row, certificate gate, provenance, and local naming row;*
9. *K records the displayed* Cont *routes from source events to event flows, from event flows to channel packets, from channel packets to generated recognizers, and from recognizers to certificate gates;*

10. *P is the* Pkg *provenance over chapter 9407, chapter 806, chapter 826,* BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}}$ *row.*

The carrier exports no row for hidden host input, report evidence, motif similarity, proof text, compiler correctness, or an unmediated generated artifact.

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditUp.GroundCompilerEventFlowAuditTasteGate single carrier alignment`

Theorem 829.2 (GroundCompiler event-flow audit NameCert obligations)**.** *For the carrier of Definition 829.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (S, F, C, L, R, G, A, H, K, P, N),$$

source-event exposure through S, event-flow readback through F, legal channel encoding and decoding through C, L, source/channel separation through R, generated-recognizer readback through G, certificate-gate readback through A, componentwise transport through H, displayed continuation through K, package provenance through P, and local naming through N.

Proof. Project the finite carrier. The rows S, F, C, L, R, G, A are the only compiler-facing data rows available to a consumer, and they name source events, event flows, channel packets, losslessness, separation, generated recognition, and certificate gates in that order.

Read the transport and route rows next. The row H compares each displayed coordinate componentwise, while K supplies only the displayed source-to-flow, flow-to-channel, channel-to-recognizer, and recognizer-to-gate routes.

Finally package the same rows with P and N. Since the provenance cites the GroundCompiler audit map and the sibling audit-map interface chapters, the NameCert surface has no extra coordinate for hidden input, report evidence, motif similarity, proof text, or compiler correctness. □

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditUp.GroundCompilerEventFlowAuditCarrier namecert obligations`

Theorem 829.3 (GroundCompiler event-flow audit lossless gate)**.** *Every accepted* $\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}$ *carrier routes a generated export through the lossless channel row L and the certificate-gate row A before any audit consumer may read it as a GroundCompiler output. The route factors through*

$$S, F, C, L, R, G, A, H, K, P, N$$

and cannot replace the displayed source/channel packet by carried metadata, report counts, motif similarity, proof text, or a host compiler input.

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditUp.GroundCompilerEventFlowAuditCarrier lossless gate`

Proof. Begin with the carrier projection. The generated-recognizer row G is downstream of the source row S, event-flow row F, channel packet C, losslessness row L, and source/channel separation row R.

Apply the NameCert obligations. They force every generated-recognizer read to pass through the displayed channel and separation rows before the certificate gate A can be consumed.

The GroundCompiler aggregate theorem named in chapter 9407 supplies the checked no-hidden-input reading for this route. The paper-side packet records only the finite audit rows, so a consumer

cannot exchange them for carried metadata, reports, motif similarity, proof text, or host compiler input. □

Theorem 829.4 (GroundCompiler event-flow audit non-escape)**.** *For every accepted* GroundCompilerEventFlowAud *carrier, every event-flow, channel, recognizer, or certificate-gate consumer remains inside the displayed rows $S, F, C, L, R, G, A, H, K, P, N$. In particular, the packet cannot promote source text into proof evidence, channel data into source identity, generated recognition into certificate evidence without A, metric or motif reports into BEDC evidence, or host compiler state into formal input.*

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditUp.GroundCompilerEventFlowAuditCarrier nonescape`

Proof. Start with the lossless gate theorem. It places every generated export behind the displayed source/event-flow/channel route and behind the certificate-gate row A.

Use the carrier classifier implicit in Definition 829.1. The source row, event-flow row, channel row, separation row, recognizer row, and gate row are distinct coordinates, and the transport row H preserves those coordinates componentwise.

Finally inspect the omitted coordinates. The carrier has no row for proof evidence carried by source text, no row equating channel data with source identity, no row for certificate evidence outside A, no report-evidence row, and no host compiler input row. All consumer reads therefore remain inside the displayed packet. □

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditUp.taste gate`

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAudit}\uparrow}$.

Constructive story: A $\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}$ packet is generated from source events, event flows, legal channel packets, losslessness rows, source/channel separation rows, generated recognizer rows, certificate gates, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAudit}\uparrow}, \mathsf{obligation})$

Scope closed: Definition 829.1, Definition 829.2, Definition 829.3, Definition 829.4, and the displayed TasteGate marker bind the carrier, NameCert obligations, lossless gate, non-escape row, generated-recognizer gate, certificate-gate route, and no-hidden-input boundary as the obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAudit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close compiler correctness, kernel correctness, proof reconstruction from source text, report evidence, motif evidence, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Scoped closure requires binding each displayed carrier, channel, losslessness, separation, recognizer, certificate-gate, transport, continuation, provenance, naming, TasteGate, and no-hidden-input row to its kernel dependency.

Chapter 830

A Concrete Naming Certificate for GroundCompilerEventFlowAuditRoute$^{\uparrow}$

The GroundCompilerEventFlowAuditRoute$^{\uparrow}$ packet realizes the GroundCompiler audit map of chapter 9407 as a finite route from event-flow input to certificate-gated export. It uses the audit-map template of chapter 808 and the concrete inventory discipline of chapter 823: those chapters display the reusable audit rows, while this chapter fixes the GroundCompiler route that keeps source channels, generated recognizers, certificate gates, address analysis, reports, motifs, and metrics in separate NameCert coordinates.

Definition 830.1 (GroundCompiler event-flow audit route carrier)**.** *A* GroundCompilerEventFlowAuditRoute$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, C, L, G, A, B, H, K, P, N)$$

with the following displayed rows:

1. *E is the event-flow input row naming the formal GroundCompiler event stream before any generated recognizer can read it;*
2. *C is the legal source-channel encoding row, including the displayed channel packet and its decoder-facing address;*
3. *L is the lossless boundary row witnessing that the channel readback returns the displayed event flow rather than hidden host input;*
4. *G is the generated-recognizer mediation row for packages, name certificates, derivation certificates, theorem identifiers, and chapter flows;*
5. *A is the certificate-gate export row through which any accepted generated artifact is read as a certificate-facing output;*
6. *B is the non-evidence boundary row separating reports, motifs, metrics, address-analysis summaries, proof text, and host compiler state from BEDC evidence;*
7. *H records componentwise* hsame *transports for the event-flow, channel, lossless boundary, recognizer, certificate gate, non-evidence boundary, provenance, and local name rows;*
8. *K records displayed* Cont *routes from event-flow input to legal channel encoding, from channel readback to generated recognition, from generated recognition to certificate gate, and from reports or motifs to the non-evidence boundary;*

9. *P is the* Pkg *provenance over chapter 9407, chapter 808, chapter 823,* BHist, hsame, Cont, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden compiler input, proof evidence extracted from source text, report evidence, motif identity, metric evidence, or an unmediated generated export.

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteTasteGate single carrier alignment`

Definition 830.2 (GroundCompiler event-flow audit route classifier)**.** *Two accepted* GroundCompilerEventFlowAudi *carriers classify together exactly when their displayed event-flow inputs, legal channel encodings, lossless boundaries, generated-recognizer rows, certificate-gate exports, and non-evidence boundaries are related componentwise by the stored* hsame *transports and replay the same* Cont *route from event-flow input through the certificate gate and non-evidence boundary.*

The classifier compares only the coordinates of Definition 830.1. It does not compare host compiler states, quotient recognizer classes, proof texts, report counts, motif similarities, metric summaries, or certificate evidence outside the displayed gate row.

Theorem 830.3 (GroundCompiler event-flow audit route NameCert obligations)**.** *For the carrier of Definition 830.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}}$ *surface consists exactly of five public obligation rows: event-flow input exposure through E, legal channel encoding and lossless readback through C, L, generated recognizer mediation through G, certificate-gate export through A, and report, motif, metric, address-analysis, proof-text, and host-state non-evidence through B. Transport through H, route replay through K, provenance through P, and local naming through N preserve those five rows as finite BEDC data.*

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier namecert obligations`

Proof. Project the finite packet $R = (E, C, L, G, A, B, H, K, P, N)$. The rows E, C, L, G, A, B are the only public data rows, and they display the event-flow input, legal channel, lossless readback, generated recognizer, certificate gate, and non-evidence boundary.

Read the five obligation rows in order. The event-flow row supplies the formal input surface; the channel and lossless rows encode and recover that surface; the recognizer row mediates generated structure; the gate row is the only export point; and the boundary row classifies reports, motifs, metrics, address summaries, proof text, and host compiler state as non-evidence.

The rows H, K, P, N transport, replay, locate, and name the same packet. Since none of them introduces another public data coordinate, the five-row obligation surface exhausts the local naming certificate. □

Theorem 830.4 (GroundCompiler event-flow audit route channel lossless boundary)**.** *Every accepted* $\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}$ *route that exports a generated recognizer result through A first reads the event-flow input E, legal channel encoding C, lossless channel boundary L, and generated-recognizer mediation row G. The export route factors through*

$$E, C, L, G, A, H, K, P, N$$

and cannot replace the displayed channel readback by source text, carried metadata, host compiler input, report counts, motif similarity, metric summaries, or proof evidence.

Lean4 ✓ BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier channel lossless boundary

Proof. Start with the classifier of Definition 830.2. It allows a route to reach the certificate gate only by replaying the displayed continuation from event-flow input through channel readback and generated recognition.

Apply the NameCert obligation surface. It places the channel encoding and its lossless boundary before generated-recognizer mediation, and it places generated-recognizer mediation before the certificate-gate export.

The omitted endpoints are precisely the objects named by the non-evidence row. Because the carrier has no coordinate that converts them into channel readback or certificate evidence, the accepted route remains lossless at the displayed event-flow boundary. □

Theorem 830.5 (GroundCompiler event-flow audit route non-evidence boundary)**.** *For every accepted* GroundCompilerEventFlowAuditRoute$^{\uparrow}$ *carrier, reports, motifs, metrics, address-analysis summaries, proof text, source text, carried metadata, and host compiler state can only be read through the displayed non-evidence boundary row B. They cannot be promoted to event-flow input, legal channel readback, generated recognition, certificate-gate evidence, or BEDC proof evidence by* hsame *transport,* Cont *replay,* Pkg *provenance, or the local* NameCert *row.*

Lean4 ✓ BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier non evidence boundary

Proof. Inspect the carrier coordinates. The only row whose role is to classify reports, motifs, metrics, address-analysis summaries, proof text, source text, carried metadata, and host compiler state is B.

Use the channel lossless boundary theorem to account for every route that reaches A. Such a route has already passed through E, C, L, G and therefore cannot treat a non-evidence object as the channel readback or generated-recognizer witness.

Finally read the structural rows H, K, P, N. They preserve and route displayed coordinates, but none of them supplies a constructor from B to E, C, L, G, A, or BEDC proof evidence. Hence every non-evidence read remains at the boundary named by B. □

Lean4 ✓ BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteNonEvidenceBo

Theorem 830.6 (GroundCompiler event-flow audit route obligation surface)**.** *Every accepted* GroundCompilerEventFlowAuditRoute$^{\uparrow}$ *route exposes the event-flow row E, audit witness rows C, L, G, A, B, componentwise* hsame *transport row H, displayed* Cont *replay row K,* Pkg *provenance row P, and local* NameCert *row N before any audit consumer reads the route. No hidden compiler input, source-text proof row, report evidence row, motif evidence row, metric evidence row, or hand-written proof row is available outside that displayed surface.*

Lean4 ✓ BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier obligation surface

Proof. Project the carrier of Definition 830.1. Its coordinates are exactly $E, C, L, G, A, B, H, K, P, N$, so every accepted route presents the event-flow row, the audit witness rows, and the structural transport, replay, provenance, and naming rows as finite BHist data.

Apply Definition 830.3 to identify those coordinates as the local NameCert surface. Then apply Definition 830.4 to force the audit consumer through E, C, L, G, A before any certificate-gate read.

Finally use Definition 830.5. Reports, motifs, metrics, source text, address summaries, proof text, and host compiler state stay behind B, while H, K, P, N only transport, replay, package, and name the same packet. Hence no hidden input or hand-written proof row can be read by the audit consumer. □

Theorem 830.7 (GroundCompiler event-flow audit route classifier stability)**.** *Let two accepted* GroundCompilerEventFlowAuditRoute$^{\uparrow}$ *route packets be classified componentwise on* $E, C, L, G, A, B, H, K, P, N$*. Displayed* hsame *transport and* Cont *replay preserve event-flow input, legal channel encoding, lossless readback, generated-recognizer mediation, certificate-gate export, and non-evidence boundary for those rows. In particular, classifier agreement on the displayed route cannot create hidden compiler input, source-text proof evidence, report evidence, motif evidence, metric evidence, or hand-written proof evidence outside the packet.*

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier classifier stability`

Proof. Start with the classifier of Definition 830.2. It reads the route componentwise from $E, C, L, G, A, B, H, K, P$, so componentwise hsame may only transport the event-flow row, legal channel row, lossless readback row, generated-recognizer row, certificate-gate row, non-evidence row, replay row, provenance row, and local name row.

Apply Definition 830.4. Any read that reaches the certificate-gate export has already passed through the legal channel, lossless readback, and generated-recognizer mediation, and Cont replay preserves that displayed order.

Finally apply Definition 830.5. The non-evidence row remains a boundary for reports, motifs, metrics, source text, address summaries, proof text, and host compiler state, while H, K, P, N only transport, replay, package, and name the same route. Thus classifier-stable packets preserve the displayed audit surface without exporting hidden evidence. □

Theorem 830.8 (GroundCompiler event-flow audit route certificate-gate exhaustion)**.** *For every accepted* GroundCompilerEventFlowAuditRoute$^{\uparrow}$ *route packet* $R = (E, C, L, G, A, B, H, K, P, N)$*, any accepted export through the certificate gate* A *is exhausted by the displayed event-flow input* E*, legal source channel* C*, lossless boundary* L*, generated-recognizer row* G*, non-evidence boundary* B*, and structural rows* H, K, P, N*. Reports, motifs, metrics, proof text, source text, carried metadata, and host compiler state cannot become certificate evidence through* A*,* hsame *transport,* Cont *replay,* Pkg *provenance, or the local* NameCert *row.*

Proof. Apply the NameCert obligations of Definition 830.3. They expose exactly the event-flow input, legal channel, lossless readback, generated recognizer, certificate gate, non-evidence boundary, and structural rows of R.

Use the channel lossless boundary of Definition 830.4. Any accepted route that reaches A has already passed through E, C, L, G in the displayed order, so the gate cannot be read as evidence for an unshown source channel or host compiler input.

Next apply the non-evidence boundary of Definition 830.5. Reports, motifs, metrics, proof text, source text, carried metadata, and host compiler state remain behind B and have no constructor into certificate evidence.

Finally use classifier stability from Definition 830.7. Componentwise hsame transport and Cont replay preserve the same rows $E, C, L, G, A, B, H, K, P, N$, while P and N only source and name that packet. Thus the certificate gate is exhausted by the displayed event-flow route and exports no additional evidence coordinate. □

Lean4 ✓ `BEDC.Derived.GroundCompilerEventFlowAuditRouteUp.GroundCompilerEventFlowAuditRouteCarrier certificate gate exhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}$ packet is generated from event-flow input, legal channel encoding, lossless channel readback, generated-recognizer mediation, certificate-gate export, report/motif/metric non-evidence, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 830.1, Definition 830.2, Definition 830.3, Definition 830.4, Definition 830.5, Definition 830.6, and Definition 830.8 bind the finite GroundCompiler event-flow audit route, channel lossless boundary, generated-recognizer mediation, certificate-gate export, and report/motif/metric non-evidence boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close compiler correctness, proof reconstruction from source text, report evidence, motif evidence, metric evidence, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Public closure requires an export theorem showing that the scoped GroundCompiler audit route is the public read surface for certificate-facing consumers.

Chapter 831

A Concrete Naming Certificate for $\mathsf{GroundCompilerAuditWitness}^{\uparrow}$

The $\mathsf{GroundCompilerAuditWitness}^{\uparrow}$ packet turns the audit-map vision of chapter 9407 into a finite NameCert witness for the rows a GroundCompiler consumer may inspect. It sits beside the family packet of chapter 816, the template packet of chapter 825, and the frontier packet of chapter 833: those chapters name audit-map lattice structure, while this chapter names the GroundCompiler witness rows that separate theorem evidence from refusals, reports, metrics, and carried provenance.

Definition 831.1 (GroundCompilerAuditWitness carrier)**.** *A* $\mathsf{GroundCompilerAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (E, T, R, M, H, C, P, N)$$

with the following visible rows:

1. *E is the event-flow evidence row, naming the displayed source events, channel packet, losslessness gate, and accepted event-flow readback supplied by the GroundCompiler audit map;*
2. *T is the theorem-evidence row, naming the finite theorem identifiers, recognized certificate gates, and accepted proof-carrying rows that may be routed to BEDC evidence consumers;*
3. *R is the refusal row, recording rejected, absent, malformed, or boundary-only compiler observations as non-evidence;*
4. *M is the metric and report row, recording counts, motifs, diagnostics, summaries, generated code reports, and compiler telemetry only as non-evidence;*
5. *H records componentwise* hsame *transports for event-flow evidence, theorem-evidence rows, refusal rows, report rows, provenance, and local naming;*
6. *C records the displayed* Cont *routes from event-flow evidence to theorem-evidence routing, from invalid observations to refusal routing, and from reports or metrics to non-evidence routing;*
7. *P is the* Pkg *provenance over chapter 9407, chapter 816, chapter 825, chapter 833,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditWitness}^{\uparrow}}$ *row.*

The carrier exports no hidden proof-producing channel, no metric-to-proof coercion, no report-to-certificate coercion, no refusal-to-evidence promotion, and no host compiler state as formal input.

Theorem 831.2 (GroundCompilerAuditWitness NameCert obligations)**.** *For the carrier of Definition 831.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditWitness}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$W = (E, T, R, M, H, C, P, N),$$

event-flow evidence readback through E, theorem-evidence routing through T, refusal separation through R, metric and report non-evidence through M, componentwise transport through H, displayed continuation through C, package provenance through P, and local naming through N.

Proof. Project the finite carrier. The row E is the only event-flow evidence coordinate, and the row T is the only coordinate that may route recognized theorem evidence to a BEDC consumer.

Separate the boundary rows next. The refusal row R classifies rejected, absent, malformed, and boundary-only observations as non-evidence, while M records metrics, motifs, reports, diagnostics, summaries, code reports, and telemetry without turning them into certificate rows.

The rows H and C preserve this routing componentwise. They transport event-flow evidence through event-flow coordinates, theorem evidence through theorem-evidence coordinates, refusals through refusal coordinates, and reports or metrics through non-evidence coordinates.

Finally package the exposed rows with P and N. Since the package cites the GroundCompiler audit map and the sibling audit-map packet chapters, the NameCert surface has exactly the displayed witness rows and no hidden proof-producing channel. □

Theorem 831.3 (GroundCompilerAuditWitness evidence non-escape)**.** *Every accepted* GroundCompilerAuditWitne *carrier keeps theorem evidence inside the displayed route E, T, H, C, P, N and keeps refusals, metrics, reports, diagnostics, generated-code summaries, and telemetry inside the non-evidence rows R, M, H, C, P, N. Thus the packet cannot promote a refusal, report, metric, motif, code summary, or host compiler state into BEDC proof evidence.*

Proof. Start from the NameCert obligations. A consumer seeking theorem evidence must read through E and T, with transport and continuation mediated by H and C.

Now inspect the non-evidence coordinates. Refusals are stored in R, while metrics, motifs, diagnostics, summaries, generated-code reports, and telemetry are stored in M. Their continuation routes in C are non-evidence routes, not theorem-evidence routes.

The package row P records the sibling audit-map lattice references and the GroundCompiler audit-map source. It therefore records the provenance of the separation rather than adding another evidence coordinate.

Consequently no omitted coordinate can serve as a proof-producing channel. Any accepted proof evidence must already be visible in the event-flow and theorem-evidence rows, and every refusal, report, metric, motif, code summary, or host compiler observation remains outside BEDC evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerAuditWitness}^{\uparrow}$ packet is generated from event-flow evidence, theorem-evidence rows, refusal rows, metric and report non-evidence rows, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GroundCompilerAuditWitness}^{\uparrow}}$, seed)

Scope closed: Definition 831.1, Definition 831.2, and Definition 831.3 bind the finite GroundCompiler audit witness packet, theorem-evidence route, refusal separation, report non-evidence row, and no-hidden-proof-channel boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GroundCompilerAuditWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close compiler correctness, proof reconstruction from reports, metric evidence, motif evidence, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-evidence routing, refusal separation, report non-evidence, transport, continuation, provenance, naming, and evidence non-escape for the same finite Ground-Compiler audit witness packet.

Chapter 832

A Concrete Naming Certificate for $\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}$

The $\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}$ packet realizes the GroundCompiler audit-map boundary of chapter 9407 as a finite NameCert surface. It sits beside the audit-map interface of chapter 806 and the audit-map family packet of chapter 807: those chapters classify audit-map rows, while the present packet isolates the compiler-facing evidence rows that may be consumed by TasteGate and self-compilation audit sites.

Definition 832.1 (GroundCompiler evidence boundary carrier)**.** *A* $\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, C, R, A, H, T, P, N)$$

with the following visible rows:

1. *E is the event-flow evidence row, naming the legal source, channel, encoding, and decoded-event data that may be treated as compiler-layer evidence;*
2. *C is the code-not-proof separation row, recording that source text, theorem code, chapter code, source substrings, and carried code channels are readable code data rather than proof or certificate evidence;*
3. *R is the generated-recognizer row, recording the generated package, name-certificate, derivation-certificate, theorem, chapter, and accepted-export recognizers that mediate evidence admission;*
4. *A is the audit/report non-evidence row, recording metric reports, motif reports, analysis summaries, compiler reports, and cannot-claim ledgers as audit outputs rather than proof inputs;*
5. *H records componentwise* hsame *transport for event-flow, separation, recognizer, report-boundary, replay, provenance, and naming rows;*
6. *T records the displayed* Cont *replay routes by which consumers inspect the event-flow evidence, generated recognizer evidence, code/report boundary, and non-input ledger;*
7. *P is the* Pkg *provenance over chapter 9407, chapter 806, chapter 807,* BHist, hsame, Cont, *and compiler-layer NameCert rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}}$ *row.*

The classifier accepts exactly the displayed packet rows. It exports no proof term from code text, no certificate from a report, no recognition evidence without the generated-recognizer row, no host equality for event flows, and no compiler-layer input outside E, R, H, T, P, N.

Theorem 832.2 (GroundCompiler evidence boundary NameCert obligations)**.** *For the carrier of Definition 832.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerEvidenceBoundary}\uparrow}$ *surface consists of five finite obligations:*

1. *carrier admission for $G = (E, C, R, A, H, T, P, N)$ as the complete compiler-layer evidence boundary packet;*
2. *generated-recognizer evidence, requiring any admitted package, certificate, theorem, chapter, or accepted-export row to pass through R;*
3. *code/report non-evidence, requiring code channels, source text, theorem code, chapter code, metric reports, motif reports, compiler reports, and analysis summaries to remain in C or A unless R supplies generated recognition evidence;*
4. *transport and replay, requiring componentwise* hsame *movement through H and displayed* Cont *inspection through T for every accepted consumer read;*
5. *ledger non-escape, requiring cannot-claim rows and audit/report rows to remain ledgered outputs rather than hidden proof, certificate, or compiler-input rows.*

Proof. Carrier admission is the displayed finite packet. The rows E, C, R, A, H, T, P, N are the only coordinates available to a consumer of the evidence boundary.

Generated-recognizer evidence is read from R. Any package, certificate, theorem, chapter, or accepted-export claim must pass through that row before it can be treated as admitted compiler-layer evidence.

The code and report boundary is read from C and A. These rows keep source text, theorem code, chapter code, compiler reports, metrics, motifs, and analysis summaries outside proof and certificate input unless the generated-recognizer row supplies the required recognition evidence.

Transport and replay are confined to H and T. Componentwise hsame movement preserves the row class being transported, and each Cont route reads back the same visible coordinate rather than creating a hidden input.

Finally, P and N bind provenance and local naming. Since the carrier has no further coordinate for proof extraction from code or certificate extraction from reports, ledger rows cannot escape their non-input status. □

Theorem 832.3 (GroundCompiler evidence boundary non-input scope)**.** *For every accepted* GroundCompilerEviden *carrier $G = (E, C, R, A, H, T, P, N)$, compiler reports, source text, metrics, motifs, theorem codes, and chapter codes do not become proof inputs, certificate inputs, or formal compiler inputs without generated recognition evidence supplied by R and replayed through the displayed routes T. Their readable status remains code, report, motif, metric, or ledger status inside C and A.*

Proof. Project the accepted carrier. The only row that can mediate evidence admission is R, while code-like and report-like artifacts are recorded in C and A.

Use the NameCert obligations. They require every consumer read to pass through the displayed transport and continuation rows H and T, so a code or report coordinate cannot be silently reclassified during transport.

If generated recognition evidence is absent, the artifact remains in its displayed code, report, motif, metric, or ledger row. The carrier has no separate proof-input, certificate-input, or hidden compiler-input coordinate, so the non-input scope is fixed by the packet itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}$ packet is generated from event-flow evidence, code-not-proof separation, generated recognizer evidence, audit/report non-evidence, componentwise hsame, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 832.1, Definition 832.2, and Definition 832.3 bind the finite GroundCompiler evidence boundary and its non-input scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerEvidenceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close TasteGate admission, self-hosting coverage, registry adequacy, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generated-recognizer evidence, code/report non-evidence, transport and replay, provenance, naming, and ledger non-escape for the same finite compiler-layer packet.

Chapter 833

A Concrete Naming Certificate for $\mathsf{AuditMapFrontierPacket}^{\uparrow}$

The $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as a frontier-facing finite certificate. It sits beside the open frontier ledger of chapter 762, the audit synthesis packet of chapter 776, the substitution audit map of chapter 779, and the normalization audit seal of chapter 777: those sibling chapters expose local audit rows, while this packet records the family frontier as one inspectable NameCert object.

Definition 833.1 (Audit-map frontier packet carrier)**.** *An* $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, C, O, R, X, P, H, Q, N)$$

with the following visible rows:

1. *T is the audit-map family tag naming the MetaCIC module and the local BEDC substrate;*
2. *C is the checked row listing finite established audit entries available to consumers;*
3. *O is the conditional row listing entries that require a named discharge route before use;*
4. *R is the obstruction row listing blocked claims that must remain visible;*
5. *X is the frontier row listing the strongest consumer-facing claims not closed by the packet;*
6. *P is the cross-map provenance row naming neighbouring audit packets that supply or consume the displayed entries;*
7. *H records componentwise* hsame *transports for the family tag, checked, conditional, obstruction, frontier, provenance, route, and local naming rows;*
8. *Q records displayed* Cont *routes by which a consumer reads checked entries, conditional entries, obstruction entries, frontier entries, cross-map provenance, and the local name;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$ *row.*

The classifier accepts exactly packets carrying these rows. It exports no proof of a frontier claim, no hidden discharge for a conditional claim, no erasure of an obstruction row, no unlisted cross-map route, and no host semantic equality coordinate.

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketTasteGate single carrier alignment`

Theorem 833.2 (Audit-map frontier packet NameCert obligations)**.** *For the carrier of Definition 833.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (T, C, O, R, X, P, H, Q, N),$$

family-tag readback through T*, checked-row readback through* C*, conditional-row readback through* O*, obstruction-row readback through* R*, frontier-row readback through* X*, cross-map provenance through* P*, componentwise transport through* H*, displayed continuation through* Q*, and local naming through* N*.*

Proof. Project the finite packet F. The rows T, C, O, R, X expose the family tag and the four audit classes as separate rows before any cross-map use is allowed.

Read the provenance and routing rows next. The row P names sibling audit packets as displayed sources or consumers, while H and Q transport and replay the same finite rows.

Finally inspect the local name N. Since the carrier has no coordinate for a hidden proof, a hidden discharge, an erased obstruction, or an unlisted route, these displayed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketCarrier namecert obligations`

Theorem 833.3 (Audit-map frontier packet non-escape)**.** *For every accepted* $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ *carrier* $F = (T, C, O, R, X, P, H, Q, N)$*, every downstream audit consumer remains inside the displayed checked, conditional, obstruction, frontier, cross-map provenance,* hsame*,* Cont*, and local* NameCert *rows. In particular, a frontier entry in* X *cannot be read as a closed theorem, a conditional entry in* O *cannot be read without its named route, and an obstruction entry in* R *cannot be removed by transport or provenance.*

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketCarrier nonescape`

Proof. Start from the carrier projection. Any consumer read first selects one of the displayed audit rows C, O, R, X and then follows the provenance, transport, continuation, and naming rows P, H, Q, N.

Use the NameCert obligations of Definition 833.2. They bind each audit class to its own row and keep cross-map use downstream of the displayed provenance row.

The sibling concrete packets cited in the opening paragraph may supply row content, but this packet does not add a theorem-producing coordinate to them. Thus frontier rows remain frontier rows, conditional rows remain routed rows, and obstruction rows remain visible boundaries. □

Theorem 833.4 (Audit-map frontier packet ledger totality)**.** *For every accepted* $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ *carrier* $F = (T, C, O, R, X, P, H, Q, N)$ *of Definition 833.1, each concrete audit entry supplied by the sibling chapters named in the opening paragraph lands in exactly one displayed ledger row among the checked row* C*, the conditional row* O*, the obstruction row* R*, or the frontier row* X*. The family tag* T*, the* BHist *packet, componentwise* hsame *data in* H*, displayed* Cont *routes in* Q*,* Pkg *provenance in* P*, and the local* NameCert *row* N *record the landing route, but they do not create a fifth audit class.*

Proof. Inspect the finite carrier row by row. The rows C, O, R, X are the only carrier coordinates whose declared role is to hold audit entries, while T merely identifies the audit-map family and local BEDC substrate.

The sibling chapters named in the opening paragraph can only be read through P, H, Q, N. These rows provide provenance, hsame transport, Cont replay, and local naming for already displayed entries.

Thus every sibling-supplied audit item has a visible destination in C, O, R, X. Since no coordinate of F is typed as a hidden theorem row or an extra ledger class, the displayed four-way ledger is total for this packet. □

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketLedgerTotality`

Theorem 833.5 (Audit-map frontier packet route non-escape)**.** *Let $F = (T, C, O, R, X, P, H, Q, N)$ be an accepted* $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ *carrier. Any route from the MetaCIC open-frontier ledger, the audit synthesis packet, the substitution audit map, or the normalization audit seal into a downstream audit consumer factors through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of F. Along such a route, a row from O, R, or X cannot be exported as a closed checked row in C.*

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketCarrier route nonescape`

Proof. Start with a displayed sibling route. The route enters the packet through the provenance row P and then uses the hsame row H and the Cont row Q to replay the selected audit entry.

The carrier has no route coordinate bypassing C, O, R, X. Therefore a conditional, obstruction, or frontier entry remains tagged by its source row while it is transported and replayed.

The local NameCert row N certifies that the consumer receives the same displayed row data. Hence a route may expose a sibling audit item, but it cannot convert an open, conditional, or obstructed entry into a checked theorem row. □

Theorem 833.6 (Audit-map frontier packet NameCert obligation scope)**.** *The* $\mathsf{NameCert}_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$ *obligations for the carrier $F = (T, C, O, R, X, P, H, Q, N)$ are scoped exactly to carrier admission, family-tag readback, four-way audit ledger readback, sibling provenance, componentwise* hsame *transport, displayed* Cont *replay, and local naming. They do not oblige the packet to prove any frontier entry, discharge any conditional entry, erase any obstruction row, or identify sibling audit packets by host equality.*

Proof. Use the carrier definition to list the visible obligations. Admission reads F as a finite BHist packet, the tag row T names the local family, and C, O, R, X supply the four audit ledger classes.

The remaining rows are structural. The provenance row P points to the sibling audit chapters, H transports displayed components by hsame, Q replays displayed continuations by Cont, and N gives the local NameCert name.

No listed row has the role of a theorem prover, a conditional-discharge oracle, an obstruction eraser, or a semantic equality witness between siblings. The obligation surface is therefore exactly the displayed carrier surface. □

Lean4 ✓ `BEDC.Derived.AuditMapFrontierPacketUp.AuditMapFrontierPacketCarrier namecert obligation scope`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ packet is generated from an audit-map family tag, checked rows, conditional rows, obstruction rows, frontier rows, cross-map provenance, componentwise hsame transports, displayed Cont routes, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$, obligation)

Scope closed: Definition 833.1, Definition 833.2, Definition 833.3, Definition 833.4, Definition 833.5, and Definition 833.6 bind the complete obligation surface: carrier admission, NameCert obligations, nonescape, ledger totality, route nonescape, and obligation scope.

Formal verification: FormalStatus(NameCert$_{\mathsf{AuditMapFrontierPacket}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close subject reduction, confluence, candidate-free normalization, hidden discharge routes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-class readback, cross-map provenance, route replay, naming, and non-escape for the same frontier packet.

Chapter 834

A Concrete Naming Certificate for $\mathsf{AuditMapRouteCompiler}^{\uparrow}$

The $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ packet realizes the GroundCompiler audit map of chapter 9407 as a finite route-compiler NameCert surface. It sits beside the concrete inventory of chapter 823 and the obstruction socket of chapter 826: the inventory names the available audit rows, the socket records the visible obstruction boundary, and the present packet compiles event-flow, source-channel, recognizer, certificate-gate, theorem-code, chapter-code, metric, motif, and cannot-claim rows into routes that downstream GroundCompiler consumers may inspect.

Definition 834.1 (AuditMapRouteCompiler carrier)**.** *An* $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, S, G, A, T, C, M, F, L, H, K, P, N)$$

with the following displayed rows:

1. *E is the GroundCompiler event-flow row naming the finite source event, generated event, and replay route being compiled;*
2. *S is the source-channel row, separating source input, channel code, carried metadata, and compiler report evidence;*
3. *G is the generated-recognizer row that decides which generated recognizer may consume the event-flow output;*
4. *A is the certificate-gate row that records the local admission route for recognized compiler-layer evidence;*
5. *T is the theorem-code route row, keeping theorem code, proof evidence, and accepted theorem export in distinct coordinates;*
6. *C is the chapter-code route row, keeping chapter code, topic code, and downstream chapter claims in distinct coordinates;*
7. *M is the metric and motif route row, displaying similarity, metric, and motif reports only as audit-map observations;*
8. *F is the cannot-claim ledger row, listing the compiler-layer claims that remain unavailable to this packet;*
9. *L is the route-compiler ledger from event-flow rows through recognizer and certificate-gate rows to theorem-code, chapter-code, metric, motif, and cannot-claim outputs;*

10. *H records componentwise* hsame *transports for $E, S, G, A, T, C, M, F, L, P, N$;*
11. *K records displayed* Cont *routes by which downstream consumers read the same finite route inventory;*
12. *P is the* Pkg *provenance over chapter 9407, chapter 823, chapter 826,* BHist, hsame, Cont, *and local compiler-layer rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapRouteCompiler}^{\uparrow}}$ *row.*

The carrier contains no hidden compiler input, no proof extracted from theorem code, no chapter claim extracted from topic code, no metric-to-evidence coercion, and no closed correctness theorem for the GroundCompiler layer.

Theorem 834.2 (AuditMapRouteCompiler NameCert obligations)**.** *For the carrier of Definition 834.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapRouteCompiler}^{\uparrow}}$ *surface consists exactly of route admission for $R = (E, S, G, A, T, C, M, F, L, H, K, P, N)$, source-channel separation through S, generated-recognizer gating through G, certificate-gate admission through A, theorem-code and chapter-code routing through T, C, metric and motif observation through M, cannot-claim ledgering through F, route-compiler exactness through L, componentwise transport through H, displayed continuation through K, provenance through P, and local naming through N.*

Proof. Project the finite carrier. The event-flow row E and source-channel row S are the only entries from which a compiler-layer route can begin.

Next read the generated-recognizer and certificate-gate rows G, A. They determine whether the event-flow output may be consumed as admitted audit evidence, and they keep recognizer output distinct from certificate evidence.

The theorem-code, chapter-code, metric, motif, and cannot-claim coordinates are then read through T, C, M, F and accounted for by L. The structural rows H, K, P, N transport, replay, package, and name exactly those rows, so the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.AuditMapRouteCompilerUp.AuditMapRouteCompilerNameCertObligations`
Lean4 ✓ `BEDC.Derived.AuditMapRouteCompilerUp.AuditMapRouteCompilerTasteGate single carrier alignment`

Theorem 834.3 (AuditMapRouteCompiler non-escape)**.** *Every accepted* $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ *packet exports GroundCompiler audit-map evidence only through*

$$E, S, G, A, T, C, M, F, L, H, K, P, N.$$

A downstream consumer may read event flows, source channels, generated recognizers, certificate gates, theorem-code routes, chapter-code routes, metric and motif observations, cannot-claim ledgers, route ledgers, transports, continuations, provenance, and the local NameCert *row. It cannot obtain hidden compiler input, proof evidence from theorem code, chapter closure from topic code, BEDC evidence from metric or motif reports, a closed compiler correctness theorem, or an unrecorded route outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.AuditMapRouteCompilerUp.AuditMapRouteCompilerCarrier nonescape`

Proof. Start with the NameCert obligations. They confine every admitted route to the displayed carrier rows and identify L as the only route-compiler ledger.

For event-flow and source-channel reads, the consumer must use E, S, G, A before any evidence row is available. For theorem-code, chapter-code, metric, motif, and cannot-claim reads, the consumer must use T, C, M, F together with the ledger row L.

The rows H, K, P, N preserve and replay the same finite inventory. Since the carrier has no coordinate for hidden compiler input, proof extraction from code, chapter closure from topic code, evidence extraction from metric reports, closed compiler correctness, or unrecorded routes, those objects cannot escape from the packet. □

Lean4 ✓ `BEDC.Derived.AuditMapRouteCompilerUp.AuditMapRouteCompilerNonescape`

Theorem 834.4 (AuditMapRouteCompiler GroundCompiler handoff)**.** *The GroundCompiler audit map of chapter 9407 hands off to downstream concrete-instance chapters through the finite* $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ *carrier of Definition 834.1. The handoff routes event flows, source channels, generated recognizers, certificate gates, theorem-code rows, chapter-code rows, metric and motif observations, and cannot-claim ledgers through* $E, S, G, A, T, C, M, F, L, H, K, P, N$*, with the sibling inventory and obstruction rows supplied by chapter 823 and chapter 826.*

Proof. Read the GroundCompiler audit map as the source of the route inventory. The concrete inventory chapter supplies the audit rows that may be named, and the obstruction socket supplies the boundary rows that may be reported without closing them.

The carrier then compiles those rows into the displayed route ledger L. Event-flow, source-channel, recognizer, certificate-gate, theorem-code, chapter-code, metric, motif, and cannot-claim data are all rows of the same finite packet.

Finally apply non-escape. The handoff is therefore a NameCert route compiler for visible audit-map data, not a hidden compiler input channel, a proof extractor, a chapter-closure mechanism, or a GroundCompiler correctness theorem. □

Lean4 ✓ `BEDC.Derived.AuditMapRouteCompilerUp.AuditMapRouteCompilerCarrier groundcompiler handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapRouteCompiler}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ packet is generated from GroundCompiler event-flow, source-channel, generated-recognizer, certificate-gate, theorem-code, chapter-code, metric, motif, cannot-claim, route-ledger, transport, continuation, provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditMapRouteCompiler}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 834.1, Definition 834.2, Definition 834.3, and Definition 834.4 bind the carrier, NameCert obligations, nonescape boundary, and GroundCompiler handoff as the scoped finite route-compiler surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AuditMapRouteCompiler}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close GroundCompiler correctness, hidden-input absence beyond the displayed route rows, theorem-code proof extraction, chapter-code closure, metric evidence, motif evidence, or bridge checking.

Upgrade path: Lean encoding remains to bind carrier admission, route-ledger exactness, non-escape, and GroundCompiler handoff for the same finite route-compiler packet.

Chapter 835

A Concrete Naming Certificate for $\mathsf{FiniteSupport}^{\uparrow}$

The $\mathsf{FiniteSupport}^{\uparrow}$ packet turns the kernel-as-category reading of chapter 9363 into a finite support surface for categorical consumers. It reads support as a bounded ledger of unary-continuation hom rows: each listed component is a displayed CategoryHomCarrier row with its Cont route, hsame transport, Pkg provenance, and local NameCert ledger. The packet is anchored by the continuation trace surface of chapter 469 and the category hom-carrier surface of chapter 61.

Definition 835.1 (Finite-support carrier)**.** *A* $\mathsf{FiniteSupport}^{\uparrow}$ *carrier is a finite packet*

$$S = (O, F, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite object-history support row, listing the unary* BHist *endpoints visible to the support packet;*
2. *F is the finite hom-support row, listing only displayed* CategoryHomCarrier *rows between endpoints in O;*
3. *L is the support ledger, recording the finite membership reads that justify each listed object and hom row;*
4. *H stores componentwise* hsame *transport for the object, hom, and ledger rows;*
5. *C stores the displayed* Cont *routes by which each support entry is consumed by a functor or naturality packet;*
6. *P is the* Pkg *provenance over* BHist*,* CategoryHomCarrier*,* hsame*,* Cont*,* $\mathsf{Functor}^{\uparrow}$*,* $\mathsf{NatTrans}^{\uparrow}$*, and* NameCert*;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite support set, a host collection, quotient morphism equality, or a category object outside the displayed BEDC rows.

Lean4 ✓ `BEDC.Derived.FiniteSupportUp.FiniteSupportTasteGate single carrier alignment`

Theorem 835.2 (Finite-support NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier admission for the packet* $S = (O, F, L, H, C, P, N)$*;*
2. *object and hom support membership, requiring every listed hom row in* F *to be a displayed* CategoryHomCarrier *row whose endpoints are listed in* O*;*
3. *ledger exactness, requiring every support read to pass through the finite ledger* L *and its displayed* Cont *route in* C*;*
4. *classifier stability under the componentwise* hsame *transports in* H*;*
5. *non-escape, excluding infinite support, host collections, quotient morphism equality, and external category objects from the public support surface.*

Proof. Carrier admission is projection from Definition 835.1. The packet lists only object support, hom support, a finite ledger, transports, continuation routes, provenance, and the local certificate row.

Membership reads are inherited from the category hom-carrier surface of chapter 61. Each hom-support entry is displayed as a CategoryHomCarrier row, and its endpoints are among the object rows recorded in O.

Ledger exactness follows the continuation trace discipline of chapter 469. A support consumer reads through L and then through the Cont route stored in C, so no entry is consumed outside the displayed finite route.

Stability is componentwise through H. The non-escape row follows because the carrier contains no field for infinite support, host collections, quotient morphism equality, or any external category object. □

Theorem 835.3 (Finite-support ledger non-escape)**.** *Every* $\mathsf{Functor}^{\uparrow}$ *or* $\mathsf{NatTrans}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteSupport}^{\uparrow}$ *packet reads only the displayed rows*

$$O, F, L, H, C, P, N.$$

In particular, support membership cannot be replaced by an infinite host support set, an unlisted hom row, quotient morphism equality, or a category-level object outside the BEDC carrier.

Proof. Read the carrier coordinates of Definition 835.1. The public surface is exactly the finite object row O, hom row F, ledger L, transports H, routes C, provenance P, and local name N.

Apply Definition 835.2. Its ledger exactness row forces every support membership read through L and C, and its stability row keeps transported reads inside the componentwise hsame surface H.

The categorical consumers named in chapter 9363 receive the same finite support packet. Since the carrier has no field for infinite support, unlisted hom rows, quotient morphism equality, or host category objects, those data cannot escape from the displayed ledger. □

Theorem 835.4 (Finite-support scoped primitive dependency coverage)**.** *The primitive obligations of* $\mathsf{FiniteSupport}^{\uparrow}$ *depend only on the displayed finite object row* O*, displayed* CategoryHomCarrier *hom row* F*, finite ledger* L*, componentwise* hsame *transport row* H*, displayed* Cont *route* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}$ *row* N*. They do not depend on an infinite support set, host collection, quotient morphism equality, or external category object.*

Proof. Project the carrier of Definition 835.1. Its primitive support data are exactly O, F, L, H, C, P, N, with F restricted to displayed CategoryHomCarrier rows and with membership reads routed through the finite ledger.

Apply the NameCert obligation surface of Definition 835.2. Carrier admission, object and hom membership, ledger exactness, componentwise transport, and non-escape are all stated over those displayed rows.

Finally use ledger non-escape from Definition 835.3. A categorical consumer cannot replace the finite rows by an infinite host support set, an unlisted collection, quotient morphism equality, or an external category object. Hence the scoped primitive dependency boundary is precisely the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSupport}^{\uparrow}$ packet is generated from finite unary object rows, displayed CategoryHomCarrier support rows, a finite ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The primitive finite-support scope is the displayed carrier, ledger non-escape boundary, and scoped primitive dependency coverage of Definition 835.4 over BHist, CategoryHomCarrier, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSupport}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FiniteSupportUp.FiniteSupportTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.FiniteSupportUp.taste gate`

Not claimed: This chapter does not close Lean verification, bridge checking, infinite support, quotient morphism equality, host category objects, or arbitrary functor categories.

Upgrade path: No further paper-side theory upgrade is pending before bridge checking for the finite support packet over BHist, CategoryHomCarrier, hsame, Cont, Pkg, and NameCert.

Chapter 836

A Concrete Naming Certificate for GroundCompilerAuditMap$^{\uparrow}$

The GroundCompilerAuditMap$^{\uparrow}$ packet realizes the GroundCompiler audit-map vision of chapter 9407 as a finite NameCert object. It sits beside the generic audit-map interface of chapter 806 and the substitution audit map of chapter 779: those chapters name the common audit rows and the MetaCIC substitution sibling, while this chapter names the GroundCompiler theorem inventory, compiler-layer frontier, generated recognizer rows, report boundary, non-escape ledger, and theorem-family handoff.

Falsifiable prediction: If an accepted GroundCompilerAuditMap$^{\uparrow}$ packet permits a public read that bypasses one of $I, K, E, C, R, Q, F, X, H, T, P, N$, or turns the report-boundary row Q or frontier row F into proof evidence, the packet is refuted.

Independence witness: auditmapinterface, substitutionauditmap, groundcompilereventflowaudit, groundcompilerrecognizerboundary, groundcompileraudit packet: no listed sibling is bijective to GroundCompilerAuditMap$^{\uparrow}$ because this packet simultaneously binds theorem inventory, conditional theorem cells, event-flow legality, compiler theorem rows, generated recognizers, report boundaries, frontier rows, theorem-family handoff, transport, continuation, provenance, and local NameCert rows.

Definition 836.1 (GroundCompiler audit-map carrier)**.** *A* GroundCompilerAuditMap$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, K, E, C, R, Q, F, X, H, T, P, N)$$

with the following displayed rows:

1. *I is the established GroundCompiler theorem-inventory row, including the legal event-flow, generated recognizer, report-boundary, and no-hidden-input theorem families named by chapter 9407;*
2. *K is the conditional theorem-cell row for compiler facts that require named channel, recognizer, source-separation, or certificate-gate data before they may be read;*
3. *E is the event-flow and channel-legality row, recording formal event streams, legal channel packets, lossless decode, and source/channel separation;*
4. *C is the compiler-layer theorem row grounded by* BEDC.GroundCompiler.MainTheorems.no_hidden_input_stream *and its surrounding theorem family;*

5. *R is the generated-recognizer row for packages, local* NameCert *packets, derivation certificates, theorem identifiers, chapter flows, and report carriers;*
6. *Q is the report-boundary row separating metric reports, motif reports, theorem code, chapter code, source reports, and audit summaries from proof or certificate evidence;*
7. *F is the compiler-layer frontier row naming host compiler trust, self-hosting tower claims, kernel acceptance, MetaCIC proof evidence, raw equality, and report-as-proof conversion as outside the packet;*
8. *X is the theorem-family handoff row linking this packet to chapter 806, chapter 779, and the sibling GroundCompiler packets without importing their hidden coordinates;*
9. *H records componentwise* hsame *transports for inventory, conditional theorem cells, event-flow and channel legality, compiler theorem rows, generated recognizers, report boundaries, frontiers, handoffs, provenance, and local naming;*
10. *T records displayed* Cont *routes by which audit consumers inspect I, K, E, C, R, Q, F, X in finite order;*
11. *P is the* Pkg *provenance over chapter 9407, chapter 806, chapter 779, the GroundCompiler theorem inventory,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden host input, host compiler trust, source text outside the legal channel, report-to-proof conversion, motif identity, unrestricted raw equality, kernel acceptance evidence, or MetaCIC proof evidence.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.taste gate`

Theorem 836.2 (GroundCompiler audit-map obligation closure)**.** *Promotion of* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *beyond* seed *requires the displayed carrier rows $I, K, E, C, R, Q, F, X, H, T, P, N$, the checked TasteGate alignment target for the single carrier, explicit nontriviality of the carrier, and field-faithful separation of the report-boundary row Q, frontier row F, and the remaining public rows.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier obligation closure`

Proof. The carrier definition exposes exactly the twelve rows $I, K, E, C, R, Q, F, X, H, T, P, N$. A promotion claim must therefore account for every public read through those rows and must keep the boundary rows Q and F from becoming proof evidence.

The existing checked TasteGate target records the single-carrier alignment used by this chapter. That marker is necessary for the verification side, but it does not by itself assert a nontrivial witness pair or discharge every field-faithful separation obligation needed for obligation closure.

Thus the next closure grade requires the current paper rows together with explicit Lean-side nontriviality and the remaining TasteGate row coverage, rather than only the displayed carrier syntax. □

Theorem 836.3 (GroundCompiler audit-map independence witness)**.** *The sibling packets auditmapinterface, substitutionauditmap, groundcompilereventflowaudit, groundcompilerrecognizerboundary, and groundcompilerauditpacket cannot supply a bijection preserving all* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *rows $I, K, E, C, R, Q, F, X, H, T, P, N$.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier handoff row independence`

Proof. The generic auditmapinterface packet names common audit rows, while substitutionauditmap specializes the MetaCIC substitution sibling. Neither packet simultaneously binds GroundCompiler theorem inventory I, conditional theorem cells K, compiler theorem rows C, generated recognizers R, report boundary Q, frontier F, and theorem-family handoff X.

The groundcompilereventflowaudit sibling covers event-flow legality, so it can overlap E but not the full theorem inventory, recognizer, report-boundary, frontier, transport, continuation, provenance, and local naming surface. The groundcompilerrecognizerboundary sibling covers recognizer and boundary data, so it can overlap R and Q but not all of $I, K, E, C, F, X, H, T, P, N$.

The groundcompilerauditpacket sibling packages adjacent GroundCompiler audit data, but this chapter requires the simultaneous twelve-row carrier with H for componentwise transport, T for displayed continuation, P for provenance, and N for the local NameCert row. Any bijection preserving only a sibling role omits at least one required row class, so no listed sibling is bijective to the full $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ packet. □

Theorem 836.4 (GroundCompiler audit-map NameCert obligations)**.** *For every accepted* GroundCompilerAuditMa *carrier*

$$G = (I, K, E, C, R, Q, F, X, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMap}^{\uparrow}}$ *surface consists exactly of theorem-inventory readback through I, conditional theorem-cell readback through K, event-flow and channel-legality readback through E, compiler theorem readback through C, generated-recognizer readback through R, report-boundary readback through Q, frontier readback through F, theorem-family handoff through X, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapNameCert obligations`

Proof. Project the finite carrier. Its first eight coordinates are the only public theorem, compiler, recognizer, boundary, frontier, and handoff rows available to a local certificate.

Read the theorem-facing rows first. The inventory row I lists established GroundCompiler facts, the conditional row K keeps route-dependent facts conditional, E records the legal event-flow and channel surface, and C records the compiler-layer theorem family headed by the checked no-hidden-input streaming compiler theorem.

Read the consumer-facing rows next. The recognizer row R supplies generated recognizer evidence, Q keeps report-like outputs on the non-evidence side of the boundary, F records the compiler-layer frontier, and X records permitted handoff to the generic audit-map interface, substitution audit map, and sibling GroundCompiler packets.

Finally inspect H, T, P, N. Transport, continuation, provenance, and local naming replay the same finite rows without adding a hidden coordinate. Hence the listed rows exhaust the local NameCert obligation surface. □

Theorem 836.5 (GroundCompiler audit-map non-escape)**.** *Every public read from an accepted* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *carrier factors through the displayed theorem inventory, conditional theorem cells, event-flow and channel legality, compiler theorem rows, generated recognizers, report boundaries, frontier rows, theorem-family handoff rows, componentwise transport, continuation, provenance, and local naming of Definition 836.1. Such a read cannot become hidden compiler input, host compiler trust, source text outside the legal channel, report evidence, motif identity, raw equality, kernel acceptance evidence, or MetaCIC proof evidence.*

Proof. Start from the NameCert obligations. They force every public read to choose one of the displayed coordinates $I, K, E, C, R, Q, F, X, H, T, P, N$.

The compiler rows E and C make legal event-flow and the no-hidden-input theorem family visible, while R and Q separate generated recognizers from report-like outputs. None of these rows contains a host source scan, host compiler trust token, or report-to-proof conversion.

The frontier and handoff rows F and X then record what remains outside the packet and where sibling consumers may read. Since handoff is routed through displayed rows rather than hidden imports, kernel acceptance and MetaCIC proof evidence remain in their own packets.

The omitted coordinates are exactly the advertised non-escape claims. Therefore no accepted carrier read can leave the finite audit-map packet. □

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier nonescape`

Theorem 836.6 (GroundCompiler audit-map theorem-family handoff)**.** *An accepted* GroundCompilerAuditMap$^{\uparrow}$ *carrier may hand theorem-family inventory to the audit-map interface of chapter 806, to the substitution audit sibling of chapter 779, and to GroundCompiler event-flow, recognizer, verdict, evidence-boundary, legal-channel, and audit-packet consumers only through the displayed handoff row X and the route row T. The handoff preserves conditional theorem cells as conditional cells and frontiers as frontier rows.*

Proof. Use the carrier projection and isolate X. This row is the only coordinate that names cross-map or sibling consumers, so every theorem-family handoff must pass through it.

The route row T supplies the displayed continuation from the inventory, conditional, compiler, recognizer, report-boundary, and frontier rows into X. A consumer therefore receives the same row class that was routed into the handoff.

Conditional theorem cells remain conditional because the carrier contains no coordinate turning K into an unconditional theorem. Frontier rows remain frontiers because F records outside claims rather than solved obligations.

Thus theorem-family handoff is a finite routed read, not an import of the target packet's hidden coordinates and not a discharge of the source packet's conditional or frontier rows. □

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier theorem family handoff`

Theorem 836.7 (GroundCompiler audit-map compiler-layer frontier)**.** *For an accepted* GroundCompilerAuditMap$^{\uparrow}$ *carrier, the compiler-layer frontier row F is the binding boundary between the concrete audit map and stronger compiler claims. It permits the packet to cite the no-hidden-input streaming compiler theorem family through C, but it does not close host compiler trust, a self-hosted compiler tower, kernel acceptance, MetaCIC reduction or typing evidence, report-to-proof conversion, motif identity, or unrestricted equality.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier compiler layer frontier`

Proof. Read the compiler theorem row C together with the frontier row F. The former records the available GroundCompiler theorem family, including the no-hidden-input streaming compiler theorem; the latter records which adjacent claims are outside the packet.

Now compare F with the remaining coordinates. Inventory, conditional theorem cells, event-flow rows, recognizer rows, report-boundary rows, and handoff rows can all be transported and

routed, but none of them is a carrier field for host compiler trust, self-hosting, kernel acceptance, MetaCIC proof evidence, report evidence, motif identity, or unrestricted equality.

Consequently the packet can expose the compiler-layer theorem family as finite audit data while keeping stronger compiler and proof claims outside the local NameCert surface. □

Theorem 836.8 (GroundCompiler audit-map conditional-cell preservation)**.** *For an accepted* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *carrier*

$$G = (I, K, E, C, R, Q, F, X, H, T, P, N),$$

every public read of the conditional theorem-cell row K preserves the named channel, generated recognizer, source-separation, and certificate-gate hypotheses recorded by the displayed rows H, T, P, N. No consumer of the packet may turn K into an unconditional theorem-inventory row, and every permitted handoff must keep K separate from the established inventory row I.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier conditional cell preservation`

Proof. Project the carrier and isolate K. By the carrier definition, K is the unique coordinate for compiler facts that still require named channel, recognizer, source-separation, or certificate-gate data before they may be read.

The transport row H and route row T can replay K only as the same displayed conditional row. The provenance row P records the chapters and kernel objects from which this condition is named, while the local row N records that the NameCert surface is the $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ surface rather than the global theorem inventory.

Since the established theorem inventory has its own coordinate I, a consumer that receives K through H, T, P, N receives a conditional theorem-cell row, not an unconditional theorem row. The finite carrier has no coordinate that erases those hypotheses, so conditional-cell preservation follows. □

Theorem 836.9 (GroundCompiler audit-map report-boundary exactness)**.** *For an accepted* GroundCompilerAuditM *carrier, the report-boundary row Q exposes metric reports, motif reports, theorem code, chapter code, source reports, and audit summaries only as report data. The same row never supplies proof evidence, certificate evidence, kernel acceptance evidence, MetaCIC proof evidence, or a conversion from report data to any such evidence.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier report boundary exactness`

Proof. Use the carrier definition to identify Q as the only report-boundary coordinate. Its named contents are report-shaped outputs: metrics, motifs, theorem code, chapter code, source reports, and audit summaries.

Apply the NameCert obligation theorem to see that public readback of Q is a local NameCert obligation, and apply the non-escape theorem to the same read. The non-escape boundary excludes report evidence, kernel acceptance evidence, MetaCIC proof evidence, and report-to-proof conversion from every accepted carrier read.

Thus Q is exact as a boundary row. It exposes the listed reports as reports, while proof evidence and certificate evidence must come from their own certified rows or from another packet, never from the report-boundary coordinate. □

Theorem 836.10 (GroundCompiler audit-map frontier handoff non-escape)**.** *For an accepted* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *carrier, the handoff row X may route theorem inventory to the generic audit-map interface, the substitution audit sibling, and GroundCompiler consumers only while preserving K as a conditional theorem-cell row and F as a compiler-layer frontier row. Such handoff cannot import host compiler trust, kernel acceptance evidence, MetaCIC proof evidence, unrestricted equality, or report-as-proof conversion.*

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditMapUp.GroundCompilerAuditMapCarrier frontier handoff nonescape`

Proof. Begin with the theorem-family handoff theorem. It makes X and the route row T the only displayed path from this packet to the audit-map interface, substitution audit sibling, and GroundCompiler consumers.

The conditional-cell preservation theorem keeps K conditional along that path. The compiler-layer frontier theorem keeps F as the row naming stronger compiler and proof claims that are outside the packet.

Now apply non-escape to the routed read through X. Because the carrier has no hidden coordinate for host compiler trust, kernel acceptance evidence, MetaCIC proof evidence, unrestricted equality, or report-as-proof conversion, handoff through X cannot import those coordinates from the consumer side. It only routes the finite inventory while preserving the conditional and frontier classes already present in the source packet. □

Theorem 836.11 (GroundCompiler audit-map public-consumer bridge)**.** *Any public consumer of an accepted* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *carrier receives the theorem-family handoff, report boundary, compiler-layer frontier, generated-recognizer rows, componentwise transport, replay, provenance, and local* NameCert *rows on the same displayed carrier route. The consumer may route the public audit-map surface through those rows, but it cannot turn report data into proof evidence, erase the conditional theorem-cell hypotheses, or treat a generated recognizer row as kernel acceptance.*

Proof. Start from the family handoff and frontier handoff non-escape theorems. They make the public route pass through the theorem-family handoff row, compiler-layer frontier row, and the same transport and replay rows before a consumer receives the packet.

Next use report-boundary exactness and conditional-cell preservation. These keep report data as report data and keep conditional theorem cells separate from established theorem inventory. The generated-recognizer rows are therefore consumer-visible only as named audit-map rows, not as proof or kernel-acceptance shortcuts.

Finally attach provenance and local naming. Rows P, N identify the carrier route as the $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ NameCert route, so every public consumer receives the displayed carrier surface and no report-to-proof conversion or kernel-acceptance shortcut. □

The mature paper-side treatment ties the theorem-inventory row I, conditional-cell row K, report-boundary row Q, frontier row F, and theorem-family handoff row X to the sibling AuditMapInterface and SubstitutionAuditMap consumers through the same displayed route T. The TasteGate target already recorded above is the AI-origin carrier discipline: it keeps the single carrier aligned while the audit-map and substitution consumers read the public surface without turning conditional cells, reports, or frontier rows into proof evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ packet is generated from established GroundCompiler theorem inventory, conditional theorem cells, event-flow and channel legality rows, compiler theorem rows, generated recognizer

rows, report-boundary rows, frontier rows, theorem-family handoff rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{GroundCompilerAuditMap}\uparrow}$, maturePackageClosed)

Scope closed: Definition 836.1, Definition 836.4, Definition 836.5, Definition 836.6, Definition 836.7, Definition 836.8, Definition 836.9, Definition 836.10, and the TasteGate target supply the paper-side carrier, NameCert obligation surface, nonescape boundary, family handoff, frontier, conditional-cell, and report-boundary rows for the finite GroundCompiler audit-map packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{GroundCompilerAuditMap}\uparrow}$, encodedDef)

Lean target: `BEDC.Derived.GroundCompilerAuditMapUp.taste\_gate`

Bridge status: none

Not claimed: This chapter does not close host compiler trust, a self-hosted compiler tower, kernel acceptance, MetaCIC proof evidence, report-to-proof conversion, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: The mature paper-side treatment is complete; bridge checking and stronger formal verification remain read from the independent verification axis.

Chapter 837

A Concrete Naming Certificate for $\mathsf{ClosedNormalEndpointLedger}^{\uparrow}$

The $\mathsf{ClosedNormalEndpointLedger}^{\uparrow}$ packet records the endpoint rows consumed by the closed-normal consistency boundary of chapter 9376. It sits beside the main-result packet of chapter 764, the normal false-exclusion packet of chapter 732, and the closed-normal consistency packet of chapter 739: those siblings expose the checked theorem and the refusal of full closed consistency, while the present ledger keeps the endpoint rows available as one finite NameCert surface for downstream audit consumers.

Definition 837.1 (Closed-normal endpoint-ledger carrier)**.** *A* $\mathsf{ClosedNormalEndpointLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, B, F, R, H, C, P, N)$$

with the following visible rows:

1. *T is the typed-row displaying* $\mathsf{HasType}([], t, \mathsf{falseEncoding})$ *for the inspected endpoint;*
2. *L is the* $\mathsf{ClosedAt}(0, t)$ *row;*
3. *B is the* $\mathsf{BetaStrong}(t)$ *row;*
4. *F is the false-encoding row naming the internal false target read by T;*
5. *R is the contradiction-route row that consumes T, L, B, F through the closed-normal consistency theorem;*
6. *H records componentwise* hsame *transport for typing, closedness, beta-strong normality, false encoding, contradiction route, continuation, provenance, and naming rows;*
7. *C records the displayed* Cont *routes by which a consumer reads T, L, B, F, R without leaving the finite endpoint packet;*
8. *P is the* Pkg *provenance over chapter 9376, chapter 764, chapter 732, chapter 739,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for subject reduction, arbitrary closed consistency, unrestricted normalization, host evaluation, quotient conversion, or a route from non-normal closed false proofs to the beta-strong endpoint.

Theorem 837.2 (Closed-normal endpoint-ledger NameCert obligations)**.** *For the carrier of Definition 837.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $E = (T, L, B, F, R, H, C, P, N)$*, typed-row readback through* T*,* ClosedAt *readback through* L*, beta-strong readback through* B*, false-encoding readback through* F*, contradiction-route readback through* R*, componentwise* hsame *stability through* H*, displayed* Cont *route replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.ClosedNormalEndpointLedgerUp.ClosedNormalEndpointLedgerTasteGate single carrier alignment`

Proof. Project the endpoint carrier first. The rows T, L, B, F are the three hypotheses and target row separated in chapter 9376: typed at the internal false encoding, closed at the empty context boundary, and already beta-strong.

Read R next. It is the only contradiction route of the packet, and it consumes exactly the displayed endpoint rows rather than an arbitrary closed proof candidate.

Finally inspect H, C, P, N. Componentwise transport preserves the endpoint rows, displayed continuations replay them, package provenance ties the ledger to the three sibling certificates, and the local name row registers the same finite surface. Since the carrier has no other coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 837.3 (Closed-normal endpoint-ledger normal boundary)**.** *Every accepted* $\mathsf{ClosedNormalEndpointLedger}^{\uparrow}$ *carrier exports only the closed beta-strong false-endpoint exclusion displayed by* T, L, B, F, R*. A downstream consumer may use the ledger to reject an endpoint already typed at the false encoding, already closed, and already beta-strong, but it obtains no full closed consistency theorem, no subject-reduction transport, and no hidden normalization route from arbitrary closed false proofs.*

Proof. Begin with the endpoint rows T, L, B, F. They are the exact boundary rows isolated by chapter 764 and chapter 739; none of them records an arbitrary reduction path.

Apply the contradiction route R only after those rows are present. This matches the normal false-exclusion surface of chapter 732: the route is available at the beta-strong endpoint and not before.

The transport, continuation, provenance, and naming rows H, C, P, N replay that same finite endpoint ledger. Because subject reduction, unrestricted normalization, host evaluation, quotient conversion, and full closed consistency are absent from the carrier, no public consumer can read them from the ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedNormalEndpointLedger}^{\uparrow}$ packet is generated from typed-row, ClosedAt-row, BetaStrong-row, false-encoding row, contradiction-route row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointLedger}^{\uparrow}}$, seed)

Scope closed: Definition 837.1, Definition 837.2, and Definition 837.3 bind the endpoint ledger for closed beta-strong false-exclusion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full closed consistency, subject reduction, unrestricted normalization, host evaluation, quotient conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-row readback, contradiction-route readback, transport, continuation, provenance, naming, and non-escape for the same finite ledger.

Chapter 838

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$

The $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ packet realizes the finite precision-to-cofinal-tail schedule used by the Real completeness route of chapter 9238. It sits between the diagonal selector surface of chapter 697, the shared-threshold packet of chapter 560, and the tail-fusion packet of chapter 658. The packet records only displayed BHist rows: no quotient stream equality, host real equality, selected completed limit, unbounded diagonal stream, or ambient completeness theorem is exported.

Definition 838.1 (Regular Cauchy tail schedule carrier)**.** *A* $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, R, W, D, K, T, M, F, E, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the finite precision-request row consumed by the Real and Completion route;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row whose tails are scheduled;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule selected for Q;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger for the selected windows;*
5. *K is the cofinal-index row naming the finite tail bound reached by the schedule;*
6. *T is the finite tail-witness row read from R, W, D, K;*
7. *M is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold packet available to consumers that compare the scheduled tail with another tail;*
8. *F is the* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *consumer row that may fuse the displayed tail witness with the downstream diagonal readback;*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing handoff row reached only after Q, R, W, D, K, T, M, F are displayed;*
10. *H stores componentwise* hsame *transports for the precision request, source row, window schedule, dyadic ledger, cofinal index, tail witness, sibling consumer rows, seal-facing row, provenance, and local name;*
11. *C stores the displayed* Cont *routes from precision request to window selection, cofinal tail choice, tail witness, shared-threshold read, tail-fusion read, and seal-facing handoff;*

12. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSchedule}^{\uparrow}}$ *row.*

The carrier is the finite schedule row linking a precision request to one displayed cofinal tail. It contains no coordinate for a host-level subsequence, quotient stream equality, selected Real limit, or ambient completeness principle.

Theorem 838.2 (Regular Cauchy tail schedule NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSchedule}^{\uparrow}}$ *surface for the carrier of Definition 838.1 consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $S = (Q, R, W, D, K, T, M, F, E, H, C, P, N)$;*
2. *classifier stability under the displayed* hsame *transports of H for the precision request, source row, windows, dyadic ledger, cofinal index, tail witness, sibling consumer rows, seal-facing row, provenance, and local name;*
3. *schedule exactness, which forces every cofinal-tail read to pass through Q, R, W, D, K, T before any sibling consumer can inspect it;*
4. *downstream consumer handoff, which routes shared-threshold reads through chapter 560, tail-fusion reads through chapter 658, and selector-origin reads through chapter 697;*
5. *ledger non-escape for the* Cont, Pkg, *and local* NameCert *rows, excluding quotient stream equality, host real equality, selected completed limits, unbounded diagonal streams, and ambient completeness witnesses.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailScheduleUp.RegularCauchyTailScheduleTasteGate single carrier alignment`

Proof. Project the carrier S. Its first six coordinates name the precision request, the regular rational source, the selected finite windows, the dyadic ledger, the cofinal index, and the finite tail witness. These rows give the schedule exactness route.

Next inspect the sibling consumer coordinates. The shared-threshold row M is the only tail-meet handoff, and the tail-fusion row F is the only fusion handoff. The chapter links them to the displayed sibling certificates, while the selector-origin route is read through the diagonal-tail selector sibling.

Finally use the structural rows H, C, P, N. They transport the same displayed coordinates, replay the same finite continuations, record package provenance, and name the local certificate. Since the carrier has no coordinate for quotient streams, host equality, selected limits, unbounded diagonal objects, or ambient completeness, the five rows exhaust the local naming surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ packet is generated from a precision request, $\mathsf{RegSeqRat}^{\uparrow}$ source row, finite $\mathsf{StreamName}^{\uparrow}$ window schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, cofinal-index row, displayed tail witness, sibling $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ consumer rows, $\mathsf{Real}^{\uparrow}$ seal-facing handoff, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSchedule}^{\uparrow}}$, seed)

Scope closed: Definition 838.1 and Definition 838.2 bind the finite precision-to-cofinal-tail schedule used between diagonal selection, shared-threshold meet, tail fusion, and the Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSchedule}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, selected completed limits, unbounded diagonal streams, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, schedule exactness, sibling-consumer handoff, and ledger non-escape for the same finite schedule packet.

Definition 838.3 (Regular Cauchy tail schedule handoff route)**.** *For an accepted* $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ *carrier*

$$S = (Q, R, W, D, K, T, M, F, E, H, C, P, N),$$

the seal-facing handoff route $\mathsf{regular_cauchy_tail_schedule_handoff_route}(S)$ *is the displayed finite route in* C *whose entrance rows are*

$$Q, R, W, D, K, T, M, F$$

and whose terminal row is the $\mathsf{Real}^{\uparrow}$ *handoff row* E*. The route is read only after the precision request, regular rational source row, finite window schedule, dyadic ledger, cofinal index, tail witness, shared-threshold consumer row, and tail-fusion consumer row have all been displayed. Its transport data are exactly the componentwise* hsame *rows stored in* H*, with provenance and local naming rows* P, N*.*

Theorem 838.4 (Regular Cauchy tail schedule seal-facing route determinacy)**.** *Let*

$$S = (Q, R, W, D, K, T, M, F, E, H, C, P, N)$$

and

$$S' = (Q', R', W', D', K', T', M', F', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ *carriers. Suppose the two packets have componentwise* hsame *transports aligning*

$$Q, Q', \quad R, R', \quad W, W', \quad D, D', \quad K, K', \quad T, T', \quad M, M', \quad F, F', \quad P, P', \quad N, N',$$

and suppose the displayed continuation rows C *and* C' *present the same finite handoff route from the aligned entrance rows* Q, R, W, D, K, T, M, F *to the seal-facing endpoints* E *and* E'*. Then*

$$\mathsf{hsame}(E, E').$$

Thus two accepted schedule packets with the same precision request, $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, cofinal index, tail witness, sibling consumer rows, transport rows, continuation route, provenance, and naming row determine the same* $\mathsf{Real}^{\uparrow}$ *seal-facing handoff up to* hsame*.*

Proof. Project both carriers using Definition 838.1. In each packet, the only seal-facing coordinate is the row E or E', and it is downstream of the displayed entrance rows Q, R, W, D, K, T, M, F.

By Definition 838.2, schedule exactness forces every cofinal-tail read to pass through Q, R, W, D, K, T before any sibling consumer can inspect it, and downstream consumer handoff then routes the shared-threshold and tail-fusion reads through M and F before the seal-facing handoff is available. Hence the terminal seal read in either packet is not an independent Real row; it is the endpoint of the finite continuation route named in Definition 838.3.

The assumed componentwise transports align the two entrance surfaces. After this alignment, the terminal clauses of C and C' are two displayed Cont derivations with the same left history and the same finite right route, one ending at E and the other ending at E'. Applying continuation determinacy from Definition 18.16 gives $\mathsf{hsame}(E, E')$.

The sibling rows are only consumer rows in this argument. The shared-threshold packet M and the tail-fusion row F are read before the seal route, and neither adds a second seal endpoint to the schedule carrier. Since the carrier has no coordinate for quotient stream equality, host real equality, selected completed limits, unbounded diagonal streams, hidden tail oracles, or ambient completeness, no alternate seal-facing route remains. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailScheduleUp.RegularCauchyTailSchedule seal facing route determinacy`

Chapter 839

A Concrete Naming Certificate for $\mathsf{AuditMapDependencyWeave}^{\uparrow}$

The $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ packet realizes the audit-map family methodology of chapter 9406 as finite cross-map routing data. It sits beside the frontier index of chapter 812 and the family ledger of chapter 810: those chapters name frontier rows and family inventories, while this packet records how local maps, neighbouring maps, obstruction rows, frontier rows, and synthesis consumers are woven into one inspectable $\mathsf{NameCert}$ surface.

Definition 839.1 (Audit-map dependency weave carrier)**.** *An* $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, L, O, F, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the local audit-map row whose positive, conditional, obstruction, and frontier cells are being routed;*
2. *L is the neighbouring-map link row, naming exactly the adjacent audit maps that may receive or supply a frontier dependency;*
3. *O is the obstruction row recording the blocked obligations that prevent a local frontier entry from being closed inside M;*
4. *F is the frontier row recording the strongest claim available at the boundary between M and the neighbouring maps named by L;*
5. *S is the synthesis-consumer row recording the downstream consumers that may read the frontier route without discharging it;*
6. *H records the componentwise* hsame *transports for local-map rows, neighbouring-map links, obstruction rows, frontier rows, synthesis-consumer rows, dependency packages, and local provenance;*
7. *C records the displayed* Cont *routes by which a consumer reaches exactly those local, neighbouring, obstruction, frontier, and synthesis rows;*
8. *P is the* Pkg *provenance over* BHist*, audit-map family rows, obstruction rows, frontier rows, synthesis consumers,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *data;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditMapDependencyWeave}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden discharge of O, a proof of F, an unlisted neighbouring map, or a synthesis consumer outside S.

Theorem 839.2 (Audit-map dependency weave NameCert obligations)**.** *For the carrier of Definition 839.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapDependencyWeave}^{\uparrow}}$ *surface consists of carrier admission for $W = (M, L, O, F, S, H, C, P, N)$, exact neighbouring-map link readback through L, obstruction visibility through O, frontier visibility through F, synthesis-consumer boundary through S, componentwise transport through H, displayed continuation through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveTasteGate single carrier alignment`

Proof. Carrier admission is the displayed finite packet. The rows $M, L, O, F, S, H, C, P, N$ are the only coordinates that a local certificate can project.

The link, obstruction, frontier, and consumer obligations are row projections. A neighbouring-map read must pass through L, an obstruction read through O, a frontier read through F, and a synthesis-consumer read through S.

Transport and continuation are then fixed by H and C. The package row P and local naming row N bind those reads to the same finite BHist packet, so no hidden discharge, unlisted neighbouring map, or external synthesis consumer can enter the surface. □

Theorem 839.3 (Audit-map dependency weave cross-map consumer boundary)**.** *Every cross-map consumer of an accepted* $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ *carrier reads the local map M, neighbouring-map link L, obstruction row O, frontier row F, and synthesis-consumer row S through the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *routes H, C, P, N. The consumer may route a frontier dependency between named maps, but it cannot replace the route by an unlisted map, erase the obstruction row, or turn the frontier row into a closed theorem.*

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveCrossMapConsumerBoundary`

Proof. Project the accepted carrier $W = (M, L, O, F, S, H, C, P, N)$. The local and neighbouring map data are visible only in M and L, while the obstruction, frontier, and synthesis-consumer data are visible only in O, F, S.

Apply the obligation surface of Definition 839.2. It routes every public read through H, C, P, N and keeps the consumer at the displayed packet boundary.

Thus a cross-map consumer can follow the named route, but the packet gives no coordinate for an unlisted neighbour, a removed obstruction, or a proof of the frontier entry. □

Theorem 839.4 (Audit-map dependency weave frontier non-escape)**.** *For every accepted* $\mathsf{AuditMapDependencyWea}$ *carrier, the frontier row F remains a named boundary row. It may be transported along H, reached by C, packaged by P, and consumed by the synthesis rows S, but it cannot escape as a hidden theorem, a discharged obstruction, a global audit-map closure claim, or a consumer route not recorded in $M, L, O, F, S, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveFrontierNonEscape`

Proof. Read the frontier coordinate from Definition 839.1. The row F is stored beside the obstruction row O and the synthesis-consumer row S, so it is not a proof object by itself.

Use Definition 839.3 for downstream reads. Every consumer route remains inside the displayed $M, L, O, F, S, H, C, P, N$ packet.

The non-escape claim is now the carrier inventory. Since no coordinate names a hidden theorem, discharged obstruction, global closure claim, or unrecorded consumer, the frontier row can only remain the visible boundary row of the weave. □

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveFrontierNonescape`

Theorem 839.5 (Audit-map dependency weave route admission)**.** *An* AuditMapDependencyWeave$^{\uparrow}$ *route is admitted only as the displayed finite packet*

$$W = (M, L, O, F, S, H, C, P, N).$$

The local map row, neighbouring-map link, obstruction row, frontier row, synthesis-consumer row, componentwise transports, displayed continuations, package provenance, and local NameCert *row are all part of the admitted route, and no unlisted map, hidden frontier proof, or external synthesis row is accepted.*

Proof. Project the carrier of Definition 839.1. The packet has exactly the rows $M, L, O, F, S, H, C, P, N$ and no coordinate for an unlisted map, hidden proof, or external synthesis row.

Apply the NameCert obligations of Definition 839.2. They bind carrier admission, neighbouring-map readback, obstruction visibility, frontier visibility, synthesis-consumer boundary, transports, continuations, package provenance, and local naming to those rows.

The cross-map consumer boundary of Definition 839.3 then keeps every public read inside the same packet. Thus route admission accepts exactly the displayed weave. □

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveRouteAdmission`

Theorem 839.6 (Audit-map dependency weave obstruction ledger)**.** *For every admitted* AuditMapDependencyWeav *route, the obstruction row O remains a visible ledger row. It may be transported by H, reached by C, packaged by P, and read by the local* NameCert *row N, but it is never discharged as a theorem, erased from the frontier route, or replaced by a global audit-map closure claim.*

Proof. Read O from the carrier of Definition 839.1. It is a coordinate of the finite packet, stored beside the frontier row F and synthesis-consumer row S.

Use the obligation surface of Definition 839.2. Obstruction visibility is one of the local certificate obligations, so a consumer must keep O displayed when it follows the route.

Finally apply the frontier non-escape theorem Definition 839.4. Since the frontier cannot escape as a hidden theorem or discharged obstruction, the obstruction ledger remains visible and undischarged. □

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveObstructionLedger`

Theorem 839.7 (Audit-map dependency weave synthesis NameCert package)**.** *Every downstream synthesis consumer reading the row S of an admitted* AuditMapDependencyWeave$^{\uparrow}$ *packet reaches it only through the named local-map row M, neighbouring-map link L, obstruction ledger O, frontier row F, componentwise transport row H, displayed continuation row C, package provenance row P, and local* NameCert *row N. The package lets the consumer route the visible dependency, but not discharge the obstruction, prove the frontier row, or import a synthesis consumer outside S.*

Proof. Start with route admission from Definition 839.5. It fixes S as one coordinate of the displayed packet $W = (M, L, O, F, S, H, C, P, N)$.

Next use the obstruction ledger of Definition 839.6. The consumer cannot read S while erasing O, because obstruction visibility is part of the same packet route.

Finally combine the cross-map consumer boundary of Definition 839.3 with frontier non-escape from Definition 839.4. These rows force every synthesis read through M, L, O, F, H, C, P, N and leave no coordinate for an external synthesis row, hidden obstruction discharge, or frontier proof. □

Lean4 ✓ `BEDC.Derived.AuditMapDependencyWeaveUp.AuditMapDependencyWeaveSynthesisNameCertPackage`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapDependencyWeave}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ packet is generated from a local audit-map row, neighbouring-map link row, obstruction row, frontier row, synthesis-consumer row, componentwise transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapDependencyWeave}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 839.1, Definition 839.2, Definition 839.3, Definition 839.4, Definition 839.5, Definition 839.6, and Definition 839.7 bind the carrier, obligations, route admission, obstruction ledger, and displayed synthesis NameCert package as the displayed bridge scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapDependencyWeave}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close hidden discharge of obstruction rows, proof of frontier rows, global audit-map closure, Lean verification, or bridge checking.

Upgrade path: Further closure requires Lean verification and bridge checking for the displayed carrier admission, neighbouring-map link readback, obstruction visibility, frontier visibility, synthesis-consumer boundary, componentwise transport, displayed continuation, provenance, and local naming rows.

Chapter 840

A Concrete Naming Certificate for $\mathsf{SubstitutionContext}^{\uparrow}$

The $\mathsf{SubstitutionContext}^{\uparrow}$ packet realizes the context-indexed rows of chapter 9380 as one finite NameCert carrier. It sits beside the binder seal of chapter 621, the closed-term boundary of chapter 527, the beta-preservation packet of chapter 716, and the typed endpoint packet of chapter 562. Its role is to keep shift, substitute, closed composition, generator closure, and subject-reduction handoff rows visible as finite context audit data without exporting an unrestricted substitution calculus.

Definition 840.1 (Substitution context audit carrier)**.** *A* $\mathsf{SubstitutionContext}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, S, U, M, G, B, E, H, K, P, N)$$

with the following displayed rows:

1. *X is the context row naming the source context, binder depth, and visible substitution window;*
2. *S is the shift row recording the context movement supplied by the substitution audit map;*
3. *U is the substitute row recording the context-indexed substitution readback;*
4. *M is the closed-composition row tying the shift and substitute reads to the same closed boundary;*
5. *G is the generator-closure row for the finite shift and substitution generators;*
6. *B is the binder-seal row read through chapter 621;*
7. *E is the subject-reduction handoff row read through the beta-preservation and typed-normal-form boundary surfaces;*
8. *H records componentwise* hsame *transports for context rows, shift rows, substitution rows, composition rows, generator rows, binder rows, handoff rows, package rows, and local certificate rows;*
9. *K records the displayed* Cont *routes for reading the shift, substitute, closed-composition, generator-closure, binder-seal, and handoff rows;*
10. *P is the* Pkg *provenance over chapter 9380, chapter 621, chapter 527, chapter 716, chapter 562,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstitutionContext}^{\uparrow}}$ *row.*

The carrier has no coordinate for host substitution semantics, unrestricted substitution commutation, semantic equality of terms, closed subject reduction, or a hidden dependent-typing bridge.

Lean4 (definition) `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditCarrier`
Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditTasteGate` single carrier alignment

Theorem 840.2 (Substitution context audit NameCert obligations)**.** *For the carrier of Definition 840.1, the local* $\mathsf{NameCert}_{\mathsf{SubstitutionContext}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$Q = (X, S, U, M, G, B, E, H, K, P, N),$$

shift-row exposure, substitute-row exposure, closed-composition exposure, generator-closure exposure, binder-seal exposure, subject-reduction handoff exposure, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and non-escape from host substitution semantics, unrestricted substitution commutation, semantic equality of terms, closed subject reduction, and hidden dependent-typing bridges.*

Proof. Carrier exposure is projection from Definition 840.1. The finite packet lists exactly the rows $X, S, U, M, G, B, E, H, K, P, N$.

The substitution-facing obligations are the rows S, U, M, G. They expose shift, substitution, closed composition, and generator closure as the context-indexed audit rows described in chapter 9380.

The sibling boundary obligations are the rows B and E. The binder row reads through chapter 621, while the handoff row is bounded by chapter 716 and chapter 562.

Transport, replay, provenance, and naming are confined to H, K, P, N. Since no remaining coordinate stores host substitution semantics, unrestricted commutation, semantic equality, closed subject reduction, or a dependent-typing bridge, the displayed rows exhaust the local obligations. □

Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditNameCertObligation`

Theorem 840.3 (Substitution context audit binder route)**.** *Let* $Q = (X, S, U, M, G, B, E, H, K, P, N)$ *be an accepted* $\mathsf{SubstitutionContext}^{\uparrow}$ *carrier. Every public read from the context-indexed shift and substitute rows into the binder boundary factors through the displayed context row* X*, the shift row* S*, the substitute row* U*, the closed-composition row* M*, the generator-closure row* G*, the binder-seal row* B*, and the replay route in* K*.*

Proof. Project the accepted carrier. The context source is X, and the substitution audit content is displayed by S, U, M, G.

The binder-facing coordinate is B. Its provenance points to chapter 621, so the binder read is a sibling packet read rather than an implicit rule about all binders.

The route row K replays the context, substitution, closed-composition, generator, and binder reads. The componentwise transport and package rows H and P preserve that same finite route, so every binder-boundary access factors through the displayed rows. □

Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditBinderRoute`

Theorem 840.4 (Substitution context audit context admission)**.** *Let* $Q = (X, S, U, M, G, B, E, H, K, P, N)$ *be an accepted* $\mathsf{SubstitutionContext}^{\uparrow}$ *carrier. Every context-indexed substitution read admits the source context only through the displayed context row* X *together with* $S, U, M, G, B, E, H, K, P, N$*. A downstream subject-reduction consumer cannot replace* X *by host context semantics, unrestricted weakening, semantic term equality, or a hidden dependent-typing bridge.*

Proof. Project the carrier of Definition 840.1. The source context, binder depth, and visible substitution window are displayed in X, while S, U, M, G record the shift, substitute, closed-composition, and generator-closure reads over that same context row.

Apply Definition 840.3. The binder-facing side of the read factors through X, S, U, M, G, B and the replay route K, so binder admission is not supplied by an ambient context theory.

The handoff, transport, package, and naming rows E, H, P, N preserve the same displayed route for downstream subject-reduction consumers. Apply Definition 840.5 to exclude host substitution semantics, unrestricted commutation, semantic equality of terms, closed subject reduction, and hidden dependent-typing bridges.

Therefore the only context admission surface available to the packet is X together with the displayed finite rows $S, U, M, G, B, E, H, K, P, N$. □

Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditContextAdmission`

Theorem 840.5 (Substitution context audit non-escape)**.** *Every accepted* $\mathsf{SubstitutionContext}^{\uparrow}$ *carrier exports substitution audit information only through*

$$X, S, U, M, G, B, E, H, K, P, N.$$

A consumer may read the context row, shift row, substitute row, closed-composition row, generator-closure row, binder-seal row, handoff row, componentwise transports, displayed continuation routes, package provenance, and local naming row. It cannot replace those rows by host substitution semantics, unrestricted substitution commutation, semantic equality of terms, closed subject reduction, or a hidden dependent-typing bridge.

Proof. Begin with the packet inventory of Definition 840.1. The rows X, S, U, M, G give the context-indexed substitution audit surface.

The rows B and E connect that audit surface to sibling packets. The binder route is bounded by Definition 840.3, and the handoff row is recorded as finite packet data rather than as the closed subject-reduction theorem.

The remaining rows H, K, P, N transport, replay, package, and name the same finite carrier. No carrier coordinate names host substitution semantics, unrestricted commutation, semantic equality, closed subject reduction, or a hidden dependent-typing bridge, so those objects cannot be exported. □

Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditNonEscape`

Theorem 840.6 (Substitution context audit ledger exactness)**.** *For an accepted* $\mathsf{SubstitutionContext}^{\uparrow}$ *carrier* $Q = (X, S, U, M, G, B, E, H, K, P, N)$*, the ledger exposed to downstream subject-reduction consumers is exactly the finite context audit packet over shift, substitute, closed composition, generator closure, binder seal, closed-boundary provenance, beta-preservation handoff, and typed-normal-form endpoint rows. The ledger does not identify these rows with host substitution semantics or with a closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.SubstitutionContextAuditUp.SubstitutionContextAuditLedgerExactness`

Proof. Read the local obligations from Definition 840.2. They expose the carrier rows and require the shift, substitute, closed-composition, generator, binder, handoff, transport, replay, provenance, and naming surfaces to remain in one packet.

The closed-boundary provenance is supplied by chapter 527. The beta-preservation and typed endpoint rows are sibling surfaces from chapter 716 and chapter 562.

The handoff row E therefore exports a finite ledger to later subject-reduction consumers. By Definition 840.5, the ledger has no route through host substitution semantics or closed subject reduction, so its exact content is the displayed packet. □

Theorem 840.7 (Substitution context audit MetaCIC scope)**.** *For an accepted* $\mathsf{SubstitutionContext}^{\uparrow}$ *packet* $Q = (X, S, U, M, G, B, E, H, K, P, N)$*, every substitution-context row is scoped only through typed classifier membership, finite residual frontier rows, closure-preservation audit rows, and local* NameCert *transport and replay. The packet does not claim subject reduction, confluence, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Project the carrier of Definition 840.1. The context row X, shift row S, substitute row U, closed-composition row M, and generator-closure row G are the only rows that can carry substitution-context data.

Apply the local obligations of Definition 840.2. They force every substitution-context read to expose the displayed carrier rows, binder-seal row, subject-reduction handoff row, componentwise transport, replay, provenance, and local naming before any downstream consumer can use the packet.

Use the binder route and context admission theorems, Definition 840.3 and Definition 840.4. These bind the typed classifier membership side of the read to $X, S, U, M, G, B, E, H, K, P, N$ rather than to a host context semantics.

Finally combine non-escape and ledger exactness from Definition 840.5 and Definition 840.6. The finite residual frontier and closure-preservation audit rows are the displayed sibling handoff rows in the ledger, while H, K, P, N only transport, replay, source, and name that finite surface. No coordinate remains for subject reduction, confluence, evaluator equality, quotient syntax equality, or kernel consistency. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubstitutionContext}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SubstitutionContext}^{\uparrow}$ packet is generated from a context row, shift row, substitute row, closed-composition row, generator-closure row, binder-seal row, subject-reduction handoff row, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstitutionContext}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 840.1, Definition 840.2, Definition 840.3, Definition 840.5, and Definition 840.6 bind the context-indexed substitution audit rows as a finite NameCert packet with sibling binder, closed-boundary, beta-preservation, and typed-normal-form handoff surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstitutionContext}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host substitution semantics, unrestricted substitution commutation, semantic equality of terms, closed subject reduction, global normalization, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicitly binding the shift, substitute, closed-composition, generator-closure, binder-route, beta-preservation, typed-normal-form, BHist, hsame, Cont, Pkg, and NameCert rows inside each proof route.

Chapter 841

A Concrete Naming Certificate for TwinSubstrateAuditBridge$^{\uparrow}$

The TwinSubstrateAuditBridge$^{\uparrow}$ packet realizes the twin-substrate architecture of chapter 9392 as a finite bridge packet between two audit surfaces. It sits beside the shared audit-map interface of chapter 806, the synthesis packet of chapter 845, and the audit-coupling packet of chapter 847: those siblings name the common audit-map rows, the architecture-level handoff, and the paired audit coupling, while this chapter names the finite NameCert bridge by which a consumer can read MetaCIC and GroundCompiler audit rows together without treating either substrate as proof of the other.

Definition 841.1 (Twin-substrate audit-bridge carrier)**.** *A* TwinSubstrateAuditBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, A, R, H, C, P, N)$$

with the following visible rows:

1. *M is the MetaCIC audit-surface row, recording closed-CIC theorem inventory, proof-theoretic boundary rows, substitution and closedness rows, and named blocker rows as MetaCIC-side audit data;*
2. *G is the GroundCompiler audit-surface row, recording event-flow, source-channel, generated recognizer, certificate-gate, report-boundary, non-input, and non-evidence rows as compiler-side audit data;*
3. *A is the alignment row, displaying which MetaCIC audit row and which GroundCompiler audit row are compared by a project-level consumer;*
4. *R is the refusal and non-impersonation ledger, recording that compiler reports, source channels, recognizers, metric rows, and host-side artifacts do not become MetaCIC proof evidence, and that MetaCIC theorem, reduction, substitution, confluence, normalization, consistency, and blocker rows do not become compiler input or generated compiler evidence;*
5. *H records componentwise* hsame *transports for M, G, A, R, continuation routes, provenance, and local naming;*
6. *C records displayed* Cont *routes by which a consumer inspects M, G, A, and R in finite order;*
7. *P is the* Pkg *provenance over chapter 9392, chapter 806, chapter 845, chapter 847,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

8. N is the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditBridge}^{\uparrow}}$ *row.*

The classifier accepts exactly those packets whose paired reads factor through the displayed tuple B and whose refusal ledger remains visible at every handoff. It rejects any packet with an undisplayed equality between M and G, any conversion from compiler report to proof evidence, any conversion from MetaCIC theorem row to compiler input, or any bridge row that hides the NameCert carrier.

Theorem 841.2 (Twin-substrate audit-bridge NameCert obligations)**.** *For every accepted* TwinSubstrateAuditBridg *carrier*

$$B = (M, G, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditBridge}^{\uparrow}}$ *surface consists exactly of MetaCIC audit-surface readback through M, GroundCompiler audit-surface readback through G, alignment readback through A, refusal and non-impersonation ledger replay through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 841.1. The packet has exactly the rows M, G, A, R, H, C, P, N, so carrier admission and local classifier exactness are coordinate reads from the same finite BHist object.

Read the two audit surfaces before reading the bridge rows. The MetaCIC row M supplies closed-CIC audit data, and the GroundCompiler row G supplies compiler-facing audit data. The alignment row A records that a downstream consumer is comparing the two rows, but it does not identify their evidence kinds.

The refusal ledger R supplies the non-impersonation clauses. It is replayed through the displayed hsame and Cont rows H and C, with provenance recorded by P and the local NameCert row recorded by N. Since no other coordinate is present, the listed rows exhaust the NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditBridgeUp.TwinSubstrateAuditBridgeTasteGate single carrier alignment`

Theorem 841.3 (Twin-substrate audit-bridge non-escape)**.** *Every public read from an accepted* $\mathsf{TwinSubstrateAuditBridge}^{\uparrow}$ *carrier factors through*

$$M, G, A, R, H, C, P, N.$$

The read may compare the MetaCIC and GroundCompiler audit rows through the alignment row and may replay the refusal ledger, but it cannot turn compiler reports, source-channel rows, generated recognizers, metric rows, or host-side artifacts into MetaCIC proof evidence, and it cannot turn MetaCIC theorem, reduction, substitution, confluence, normalization, consistency, or blocker rows into compiler input or generated compiler evidence.

Proof. Start with the NameCert obligations of Definition 841.2. Any public read must first choose one of the displayed rows of B.

For compiler-facing data, the only source is G. The refusal ledger R records that reports, source channels, generated recognizers, metric rows, and host-side artifacts remain compiler-side audit data.

For MetaCIC-facing data, the only source is M. The same ledger records that theorem rows, reduction rows, substitution rows, confluence rows, normalization rows, consistency rows, and blocker rows remain MetaCIC-side audit data.

The alignment row A and the routes H, C only transport and replay those displayed rows. They contain no equality, coercion, hidden input, or code-as-proof coordinate. Therefore no accepted public read escapes the twin-substrate boundary. □

Theorem 841.4 (Twin-substrate audit-bridge consumer handoff)**.** *Every consumer handoff from an accepted* $\mathsf{TwinSubstrateAuditBridge}^{\uparrow}$ *carrier to an audit-map, synthesis, coupling, or self-compilation reader is a finite handoff of the packet rows* M, G, A, R, H, C, P, N*. The receiving reader may cite the bridge as a NameCert surface aligning two audit namespaces, but it may not cite the bridge as subject reduction, compiler correctness, hidden input, code-as-proof, or a checked bridge theorem.*

Proof. Use the displayed continuation row C. It is the only route by which a consumer receives the MetaCIC row, the GroundCompiler row, the alignment row, and the refusal ledger.

The sibling audit-map and synthesis chapters determine the permitted reading shape. chapter 806 supplies the common audit-map interface, chapter 845 supplies the architecture-level handoff, and chapter 847 supplies a paired audit-coupling surface.

The handoff therefore exports the displayed NameCert packet and its provenance, not a theorem in either substrate. The exclusions follow from the absence of carrier coordinates for subject reduction, compiler correctness, hidden input, code-as-proof, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TwinSubstrateAuditBridge}^{\uparrow}$ packet is generated from a MetaCIC audit-surface row, a GroundCompiler audit-surface row, an alignment row, a refusal and non-impersonation ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 841.1, Definition 841.2, Definition 841.3, and Definition 841.4 bind the finite bridge carrier, NameCert obligation surface, non-escape ledger, and consumer handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, compiler correctness, hidden input exclusion beyond the displayed refusal ledger, code-as-proof exclusion beyond the displayed refusal ledger, Lean verification, audit cleanliness, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, MetaCIC row exactness, GroundCompiler row exactness, alignment exactness, refusal-ledger replay, non-impersonation, consumer handoff, transport, continuation, provenance, and local naming for the same finite packet.

Chapter 842

TwinSubstrateRoundtrip NameCert Construction

The $\mathsf{TwinSubstrateRoundtrip}^{\uparrow}$ packet is the finite cross-substrate certificate for the architecture of chapter 9392. It keeps the MetaCIC audit rows of chapter 9385 separate from the GroundCompiler audit rows of chapter 836, while reusing the substitution-context handoff of chapter 840 to identify the closed-calculus rows that may be read by the compiler-facing report. The carrier records a roundtrip boundary, not a merged substrate.

Definition 842.1 (TwinSubstrateRoundtrip carrier)**.** *A* $\mathsf{TwinSubstrateRoundtrip}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, G, S, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC audit row, read through the closed-calculus audit family of chapter 9385;*
2. *G is the GroundCompiler audit row, read through the finite compiler packet of chapter 836;*
3. *S is the substitution-context audit row of chapter 840, naming exactly the closed-calculus context data that the roundtrip may expose to the compiler side;*
4. *B is the boundary ledger stating which roundtrip reads are MetaCIC reads, which are GroundCompiler reads, and which are shared only through the displayed substitution-context row;*
5. *H records componentwise* hsame *transports for M, G, S, B and the local structural rows;*
6. *C records the displayed* Cont *routes for MetaCIC inspection, GroundCompiler inspection, substitution-context handoff, and boundary-ledger replay;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *the MetaCIC audit surface, the GroundCompiler audit surface, and the substitution-context audit surface;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateRoundtrip}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows M, G, S, B, H, C, P, N.

Theorem 842.2 (TwinSubstrateRoundtrip NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateRoundtrip}^{\uparrow}}$ *surface for Definition 842.1 consists of carrier admission, MetaCIC audit-row exposure, GroundCompiler audit-row exposure, substitution-context handoff exposure, boundary-ledger exactness, componentwise* hsame *stability, displayed* Cont *route stability,* Pkg *provenance, and non-escape from host compilation evidence, hidden closed-CIC transport, compiler reports promoted to proofs, or a collapsed one-substrate reading.*

Proof. Carrier admission is the tuple $T = (M, G, S, B, H, C, P, N)$ of Definition 842.1. The rows M and G are separate coordinates, so a consumer can inspect the MetaCIC audit side and the GroundCompiler audit side without identifying them.

The substitution-context row S is the only shared handoff coordinate. It names the closed-calculus context data that may be read across the boundary, and the boundary ledger B records whether a roundtrip read is MetaCIC-facing, GroundCompiler-facing, or shared through that row.

The structural rows H, C, P, N supply only transport, continuation replay, provenance, and local naming for the displayed packet. Since the carrier has no coordinate for host compilation evidence, hidden closed-CIC transport, proof evidence extracted from compiler reports, or a one-substrate collapse, the listed obligations exhaust the local NameCert surface. □

Theorem 842.3 (TwinSubstrateRoundtrip substrate separation)**.** *Every accepted* $\mathsf{TwinSubstrateRoundtrip}^{\uparrow}$ *packet preserves the substrate separation stated in chapter 9392. A MetaCIC-facing consumer may read only* M, S, B, H, C, P, N*, a GroundCompiler-facing consumer may read only* G, S, B, H, C, P, N*, and a cross-substrate consumer must pass through the displayed substitution-context row* S *and boundary ledger* B *before either audit row is used.*

The packet exports no theorem turning GroundCompiler reports into MetaCIC proof evidence, no compiler-side replacement for the MetaCIC frontier, no hidden host compiler input, and no collapsed audit namespace.

Proof. Project the carrier rows. The MetaCIC audit row M and the GroundCompiler audit row G are different coordinates, while the only shared coordinate is the substitution-context row S together with the boundary ledger B.

Use the MetaCIC synthesis of chapter 9385 to read M as a closed-calculus audit surface. Use the GroundCompiler NameCert of chapter 836 to read G as a compiler-facing audit packet. The substitution-context chapter chapter 840 supplies the shared context route, but it does not identify the two audit rows.

The boundary ledger B and structural rows H, C, P, N route every cross-substrate read through the displayed packet. Therefore no consumer can replace the MetaCIC frontier by a compiler report, import hidden host compiler input, or collapse the two audit namespaces into one substrate. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateRoundtrip}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TwinSubstrateRoundtrip}^{\uparrow}$ packet is generated from finite BHist rows for MetaCIC audit data, GroundCompiler audit data, a substitution-context handoff, boundary ledger, transports, continuations, provenance, and local naming. The roundtrip keeps the two substrate rows separate and uses Cont to route any shared read through the displayed substitution-context row before either side is consumed. Componentwise hsame preserves MetaCIC, GroundCompiler, context, and boundary rows without merging them, while Pkg records the synthesis and audit dependencies. The packet names a cross-substrate boundary, not a unified substrate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateRoundtrip}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the TwinSubstrateRoundtrip carrier, NameCert obligations, substrate separation theorem, and boundary-ledger route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateRoundtrip}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim compiler reports as MetaCIC proof evidence, hidden closed-CIC transport, host compiler input, or collapse of the two audit namespaces.

Upgrade path: Obligation closure requires Lean encodings of the roundtrip carrier, substitution-context handoff, substrate-separation theorem, and boundary non-escape.

Chapter 843

A Concrete Naming Certificate for $\mathsf{TwinSubstrateAuditPacket}^{\uparrow}$

The $\mathsf{TwinSubstrateAuditPacket}^{\uparrow}$ surface makes the architecture of chapter 9392 into one finite concrete-instance packet. It reads the compiler-facing side through chapter 827 and the closed-CIC audit side through chapter 776, while keeping the two non-escape ledgers separate.

Definition 843.1 (TwinSubstrateAuditPacket carrier)**.** *A* $\mathsf{TwinSubstrateAuditPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, M, A, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GroundCompilerAuditPacket}^{\uparrow}$ *route carrying event-flow, source-channel, recognizer, certificate-gate, report-boundary, module-inventory, no-hidden-input, and compiler export rows;*
2. *M is the* $\mathsf{MetaCICAuditSynthesisPacket}^{\uparrow}$ *route carrying subject-reduction obstruction, confluence, normalization, substitution, decidable-checking, closed-normal assembly, and blocker rows;*
3. *A is the cross-substrate address row naming which compiler artifact route is read against which closed-CIC audit route;*
4. *B is the boundary ledger with one compiler non-escape side and one MetaCIC blocker side;*
5. *H records componentwise* hsame *transports for the compiler route, MetaCIC route, address row, boundary ledger, route, package, and local naming rows;*
6. *C records displayed* Cont *routes by which consumers read G, M, A, and B without identifying their evidence kinds;*
7. *P is the* Pkg *provenance over the twin-substrate architecture, the GroundCompiler audit packet, the MetaCIC audit synthesis packet,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditPacket}^{\uparrow}}$ *row.*

The classifier accepts exactly those packets whose consumer reads factor through the displayed tuple T and whose compiler-facing evidence remains separated from closed-CIC proof evidence. It rejects any packet that treats a compiler report as a MetaCIC theorem, a MetaCIC blocker as a compiler input, a source-channel row as proof evidence, or a closed-CIC audit row as a generated compiler artifact.

Lean4 ✓ BEDC.Derived.TwinSubstrateAuditPacketUp.TasteGate.TwinSubstrateAuditPacketTasteGate single carrier alignment

Theorem 843.2 (TwinSubstrateAuditPacket NameCert obligation surface)**.** *For every accepted* $\mathsf{TwinSubstrateAuditPacket}^{\uparrow}$ *carrier*

$$T = (G, M, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditPacket}^{\uparrow}}$ *surface consists exactly of GroundCompiler audit-packet readback through* G*, MetaCIC audit-synthesis readback through* M*, cross-substrate address readback through* A*, separate compiler and MetaCIC boundary readback through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 843.1. Its visible rows are precisely G, M, A, B, H, C, P, N, so local carrier admission and classifier exactness are coordinate reads from the same finite packet.

Read the two substrate routes separately. The row G is interpreted by the GroundCompiler audit packet, while M is interpreted by the MetaCIC audit synthesis packet. The address row A records their pairing but does not merge the evidence types.

Finally read B, H, C, P, N. The boundary ledger preserves the compiler non-escape side and the MetaCIC blocker side as distinct rows, and the transport, continuation, package, and naming rows only replay that displayed separation. Hence the NameCert obligation surface is exhausted by the packet. □

Theorem 843.3 (TwinSubstrateAuditPacket non-escape)**.** *Every consumer of an accepted* TwinSubstrateAuditPack *carrier factors through*

$$G, M, A, B, H, C, P, N.$$

The consumer may inspect the GroundCompiler audit route, the MetaCIC audit-synthesis route, the cross-substrate address row, and the two-sided boundary ledger. It cannot convert compiler reports, source-channel rows, generated recognizers, or metric boundaries into MetaCIC proof evidence, and it cannot convert MetaCIC reduction, typing, normalization, confluence, or blocker rows into hidden compiler input or generated compiler evidence.

Proof. Start with Definition 843.2. Every accepted read is already routed through the finite NameCert surface of T.

For compiler-facing reads, use the GroundCompiler audit packet route carried by G. Its non-escape ledger keeps report rows, source-channel rows, generated recognizers, and metric boundaries on the compiler side of the packet.

For MetaCIC-facing reads, use the audit synthesis route carried by M. Its blocker ledger keeps closed-CIC obstruction and proof-theoretic rows inside the MetaCIC side.

The address row A only records which two routes are being compared in the project-level audit story, and B records the two refusals without adding a conversion row. Since the carrier has no coordinate that identifies the two evidence kinds, no consumer can escape the displayed twin-substrate boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditPacket}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TwinSubstrateAuditPacket}^{\uparrow}$ packet is generated from a GroundCompiler audit-packet row, a MetaCIC audit-synthesis row, a cross-substrate address row, a two-sided boundary ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 843.1, Definition 843.2, and Definition 843.3 bind the finite twin-substrate carrier, classifier, NameCert surface, and separate non-escape ledgers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TwinSubstrateAuditPacket}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed consistency, self-hosted compiler trust, report-to-proof conversion, cross-substrate evidence identity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, GroundCompiler route readback, MetaCIC route readback, address and boundary ledger exactness, separate non-escape rows, transport, route replay, provenance, and local naming for the displayed finite packet.

Chapter 844

A Concrete Naming Certificate for $\mathsf{ClosedConsistencyGap}^{\uparrow}$

The $\mathsf{ClosedConsistencyGap}^{\uparrow}$ packet realizes the audit boundary of chapter 9385 as finite BEDC data. It sits beside the subject-reduction packet of chapter 483 and the typed normal-form endpoint of chapter 562: the first names the preservation-facing route, the second names the terminal normal-form surface, and this packet records why the closed-normal consistency branch cannot be read as unconditional closed consistency until the subject-reduction discharge socket is supplied.

Definition 844.1 (ClosedConsistencyGap carrier)**.** *A* $\mathsf{ClosedConsistencyGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, N, S, D, T, H, C, P, L)$$

with the following displayed coordinates:

1. *A is the MetaCIC audit-synthesis row that separates closed-normal consistency from unconditional closed consistency;*
2. *N is the closed-normal branch row, including the normal-form false exclusion and closed-normal assembly reads;*
3. *S is the subject-reduction handoff row, read through the finite preservation surface of chapter 483;*
4. *D is the discharge-socket row for the missing subject-reduction bundle, naming the dependent-codomain, application, lambda, and product preservation slots as finite obligations;*
5. *T is the typed normal-form endpoint row, read through chapter 562;*
6. *H records componentwise* hsame *transports for the audit row, closed-normal row, subject-reduction row, discharge socket, typed normal-form row, package row, and local naming row;*
7. *C records the* Cont *routes by which a consumer reads the closed-normal branch, the subject-reduction socket, and the terminal normal-form endpoint;*
8. *P is the* Pkg *provenance over chapter 9385, chapter 483, chapter 562,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyGap}^{\uparrow}}$ *row.*

The carrier has no coordinate for a filled subject-reduction discharge bundle, a hidden transport from arbitrary closed proofs to closed normal endpoints, a global Church–Rosser theorem, candidate-free normalization, or a completed proof of unconditional closed consistency.

Lean4 ✓ BEDC.Derived.ClosedConsistencyGapUp.ClosedConsistencyGapTasteGate single carrier alignment

Theorem 844.2 (ClosedConsistencyGap NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyGap}^{\uparrow}}$ *surface for Definition 844.1 consists of carrier admission for* $G = (A, N, S, D, T, H, C, P, L)$*, closed-normal branch exposure through* N *and* T*, subject-reduction socket exposure through* S *and* D*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*, package provenance through* P*, and local naming through* L*.*

Proof. Project the finite packet. The audit row A, closed-normal row N, subject-reduction row S, discharge socket D, typed normal-form row T, transport row H, continuation row C, provenance row P, and local certificate row L are the whole carrier surface.

Read the positive endpoint through N and T. These rows expose closed-normal consistency and the terminal typed normal-form boundary as named finite data, with their public reads transported only through H and replayed through C.

Read the blocked transport through S and D. The subject-reduction row is a handoff to the preservation-facing sibling, while the discharge socket records the missing bundle as obligations rather than as a supplied theorem. Since the carrier lists no filled bundle coordinate, the local certificate stops at the displayed gap. □

Theorem 844.3 (ClosedConsistencyGap route exhaustion)**.** *Every consumer route out of an accepted* $\mathsf{ClosedConsistencyGap}^{\uparrow}$ *packet factors through the displayed rows* $A, N, S, D, T, H, C, P, L$*. A closed-normal consistency read reaches* N *and* T*; a preservation-facing read reaches* S *and* D*; and no route converts agreement on the remaining rows into a closed subject-reduction theorem or unconditional closed consistency.*

Proof. Start with the continuation row C. Its named routes are exactly the closed-normal branch route, the subject-reduction socket route, and the terminal normal-form endpoint route.

The closed-normal branch is exhausted by N and T. The sibling typed normal-form packet supplies a finite endpoint boundary, but that boundary is still read only after normality is already displayed.

The subject-reduction side is exhausted by S and D. The preservation-facing sibling names the route a discharge would use, while the socket row keeps the missing discharge bundle visible. Thus H, C, P, L can transport and replay the packet, but cannot create an extra route outside it. □

Theorem 844.4 (ClosedConsistencyGap non-escape)**.** *An accepted* $\mathsf{ClosedConsistencyGap}^{\uparrow}$ *carrier exports only the finite audit gap packet over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not export a filled subject-reduction discharge bundle, a hidden closed-proof transport, an unrestricted confluence theorem, candidate-free normalization, host conversion, erased derivation semantics, or unconditional closed consistency.*

Proof. Inspect the carrier inventory of Definition 844.1. Every public coordinate is either an audit row, a closed-normal row, a subject-reduction socket row, a typed normal-form endpoint, a componentwise transport, a continuation route, package provenance, or local naming.

Use the obligation surface to authenticate those rows. The local NameCert accepts carrier exposure, branch exposure, socket exposure, transport, route replay, provenance, and naming, all over the same finite packet.

No excluded object is a coordinate of G. A consumer therefore cannot replace the discharge socket by a proof, cannot transport arbitrary closed proofs into the normal endpoint, and cannot read unconditional closed consistency from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedConsistencyGap}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedConsistencyGap}^{\uparrow}$ packet is generated from a MetaCIC audit-synthesis row, a closed-normal branch row, a subject-reduction handoff row, a discharge-socket row, a typed normal-form endpoint row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 844.1, Definition 844.2, Definition 844.3, and Definition 844.4 state the finite packet, obligation surface, route exhaustion, and non-escape boundary for the MetaCIC closed-consistency gap.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, arbitrary closed Church–Rosser, candidate-free normalization, hidden proof transport, unconditional closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-normal branch exposure, subject-reduction socket exposure, route exhaustion, non-escape, package provenance, and local naming for the same finite gap packet.

Chapter 845

A Concrete Naming Certificate for TwinSubstrateSynthesis$^{\uparrow}$

The TwinSubstrateSynthesis$^{\uparrow}$ packet realizes the twin-substrate architecture of chapter 9392 as a finite cross-substrate audit object. It sits beside the shared audit-map interface of chapter 806 and the GroundCompiler audit-map packet of chapter 836: those siblings supply the generic audit rows and the compiler-facing theorem-family boundary, while this chapter names the finite route that keeps MetaCIC and GroundCompiler reads separate until a displayed cross-substrate handoff is chosen.

Definition 845.1 (Twin-substrate synthesis packet)**.** *A* TwinSubstrateSynthesis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, G, I, A, F, X, H, C, P, N)$$

with the following visible rows:

1. *M is the MetaCIC audit-surface row, recording closed-calculus theorem inventory, conditional fronts, and obstruction boundaries named by chapter 9392;*
2. *G is the GroundCompiler audit-surface row, recording event-flow, recognizer, report-boundary, cannot-claim, and theorem-family data from chapter 836;*
3. *I is the shared audit-map interface row read through chapter 806;*
4. *A is the architecture row naming the permitted cross-substrate handoff from compiler-facing artifacts to closed-CIC audit obligations;*
5. *F is the frontier row separating MetaCIC open proof-theoretic boundaries from GroundCompiler pipeline boundaries;*
6. *X is the finite consumer row for downstream chapters that need both substrate names without importing either substrate's hidden coordinates;*
7. *H records componentwise* hsame *transports for the MetaCIC row, GroundCompiler row, interface row, architecture row, frontier row, consumer row, provenance row, and local naming row;*
8. *C records displayed* Cont *routes by which a consumer reads M, G, I, A, F, X in finite order;*
9. *P is the* Pkg *provenance over chapter 9392, chapter 806, chapter 836,* BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateSynthesis}^{\uparrow}}$ *row.*
The carrier has no coordinate for collapsing MetaCIC into GroundCompiler, turning compiler reports into proof evidence, importing hidden host structure, or discharging a MetaCIC frontier by a compiler-layer row.

Theorem 845.2 (Twin-substrate synthesis NameCert obligations)**.** *For every accepted* $\mathsf{TwinSubstrateSynthesis}^{\uparrow}$ *carrier*

$$T = (M, G, I, A, F, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateSynthesis}^{\uparrow}}$ *surface consists exactly of MetaCIC row readback through M, GroundCompiler row readback through G, shared interface readback through I, architecture handoff through A, frontier separation through F, downstream consumer readback through X, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite carrier. Its first six coordinates are the only substrate, interface, architecture, frontier, and consumer rows available to the local certificate.

Read the substrate rows separately. The MetaCIC coordinate M records closed-calculus audit content and frontiers, while the GroundCompiler coordinate G records compiler-facing theorem-family and report-boundary content. The interface row I gives a common audit-map shape, but it does not identify the two substrate rows.

Next inspect A and F. The architecture row permits a named handoff only after the source substrate row has been displayed, and the frontier row records which claims remain outside that handoff. Thus a compiler-facing row cannot discharge a proof-theoretic frontier, and a MetaCIC theorem row cannot certify a hidden compiler input.

Finally, X, H, C, P, N route consumer reads, transport the same coordinates componentwise, and bind provenance with local naming. Since no other coordinate is present, the listed rows exhaust the NameCert obligation surface. □

Theorem 845.3 (Twin-substrate cross-substrate non-escape)**.** *Every public read from an accepted* $\mathsf{TwinSubstrateSynthesis}^{\uparrow}$ *carrier factors through the displayed MetaCIC row, GroundCompiler row, shared audit-map interface row, architecture handoff row, frontier row, downstream consumer row, transport row, continuation row, provenance row, and local naming row of Definition 845.1. Such a read cannot collapse the two substrates, convert a compiler report into proof evidence, import hidden host input, or solve a MetaCIC obstruction by a GroundCompiler route.*

Proof. Start with the NameCert obligations. They force every public read to choose one of the displayed coordinates $M, G, I, A, F, X, H, C, P, N$ before it can reach either substrate.

For a MetaCIC-facing read, the consumer receives M together with the architecture, frontier, transport, continuation, provenance, and naming rows. For a GroundCompiler-facing read, the consumer receives G through the same displayed routing discipline. The shared interface row I supplies common audit-map shape only after the substrate row remains visible.

The frontier row F records the non-conversions: report evidence is not proof evidence, compiler legality is not closed-CIC discharge, and hidden host input is not a carrier coordinate. The handoff row A therefore routes between named substrate surfaces without merging their claims.

Coordinate exhaustion gives the non-escape clause. The carrier contains no field for substrate collapse, report-to-proof conversion, hidden host import, or obstruction discharge, so no accepted public read can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateSynthesis}^{\uparrow}}$.

Constructive story: A $\mathsf{TwinSubstrateSynthesis}^{\uparrow}$ packet is generated from a MetaCIC audit-surface row, a GroundCompiler audit-surface row, a shared audit-map interface row, an architecture handoff row, a frontier row, a downstream consumer row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateSynthesis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 845.1, Definition 845.2, and Definition 845.3 bind the finite cross-substrate audit packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateSynthesis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close MetaCIC frontier obligations, prove GroundCompiler self-hosting, turn compiler reports into proof evidence, provide Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, substrate-row exactness, architecture handoff, frontier separation, consumer readback, transport, continuation, provenance, naming, and cross-substrate non-escape for the same finite packet.

Chapter 846

A Concrete Naming Certificate for TwinSubstrateAuditUp

The TwinSubstrateAuditUp packet gives the twin-substrate synthesis of chapter 9392 a concrete NameCert surface for audit handoff. It sits between the MetaCIC audit synthesis of chapter 9385 and the GroundCompiler audit map of chapter 9407, while the naming discipline follows the apophatic boundary of chapter 9400: each row names what may be read, and every absent conversion remains absent.

The audit handoff consumes the packetized row discipline of chapter 843 before aligning the MetaCIC and GroundCompiler audit rows, and the local carrier only replays those packet rows through M, G, A, H, C, P, N.

Definition 846.1 (Twin-substrate audit carrier)**.** *A* TwinSubstrateAuditUp *carrier is a finite* BHist *packet*

$$Q = (M, G, A, H, C, P, N)$$

with the following visible rows:

1. *M is the MetaCIC audit row, carrying the closed-calculus theorem inventory, conditional proof-theoretic fronts, and obstruction boundaries named by chapter 9385;*
2. *G is the GroundCompiler audit row, carrying the legal channel, generated recognizer, event-flow, report-boundary, and no-hidden-input rows named by chapter 9407;*
3. *A is the roundtrip alignment row that records which MetaCIC audit read and which GroundCompiler audit read are being compared at the handoff surface;*
4. *H records componentwise* hsame *transport for M, G, A, C, P, and N;*
5. *C records the displayed* Cont *routes through which a consumer reads the MetaCIC row, the GroundCompiler row, and the alignment row in finite order;*
6. *P is the* Pkg *provenance over chapter 9392, chapter 9385, chapter 9407, chapter 9400,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditUp}}$ *row.*

The carrier exports no subject-reduction proof, no compiler report as proof evidence, no hidden host input, and no collapse of the two audit substrates.

Theorem 846.2 (Twin-substrate audit NameCert obligations)**.** *For every accepted* TwinSubstrateAuditUp *carrier*

$$Q = (M, G, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditUp}}$ *surface consists exactly of MetaCIC audit readback through* M*, GroundCompiler audit readback through* G*, roundtrip alignment through* A*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the finite carrier. Its coordinates are precisely the two audit rows, the alignment row, the transport row, the continuation row, the provenance row, and the local naming row.

Read M and G separately. The MetaCIC coordinate carries proof-theoretic audit status from chapter 9385; the GroundCompiler coordinate carries compiler-layer audit status from chapter 9407. The alignment coordinate A names a handoff comparison, but it does not identify the two audit rows.

Finally, H, C, P, and N transport, replay, package, and name those displayed rows. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 846.3 (Twin-substrate audit roundtrip non-escape)**.** *Every public roundtrip read from an accepted* TwinSubstrateAuditUp *carrier factors through the displayed MetaCIC audit row, GroundCompiler audit row, roundtrip alignment row, componentwise* hsame *row, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row of Definition 846.1. Such a read exports no subject-reduction proof, no compiler report as proof evidence, no hidden host input, and no collapse of the two audit substrates.*

Proof. Use the NameCert obligation surface. A public read must choose a displayed coordinate of Q and then follow the stored continuation route; there is no independent route around M, G, A, H, C, P, N.

For a MetaCIC-facing read, the consumer sees M and the alignment row that relates it to the compiler audit surface. For a GroundCompiler-facing read, the consumer sees G and the same alignment row. Componentwise hsame transport preserves the distinction between the rows being compared.

The excluded objects are not carrier coordinates. Subject-reduction proof evidence, compiler-report proof evidence, hidden host input, and substrate collapse therefore cannot be exported by any accepted roundtrip read. □

Theorem 846.4 (Twin-substrate audit vision concretization)**.** *The* TwinSubstrateAuditUp *carrier of Definition 846.1 concretizes the twin-substrate audit handoff of chapter 9392 as a finite NameCert packet. The concretization preserves the MetaCIC and GroundCompiler audit roles as separate rows, records their roundtrip alignment as a displayed row, and keeps the apophatic exclusions of chapter 9400 at the local naming boundary.*

Proof. The vision chapter names two cooperating audit substrates. The carrier supplies one finite row for the MetaCIC audit surface and one finite row for the GroundCompiler audit surface, so the two roles remain visible.

The roundtrip alignment row gives the concrete handoff point. It is read through displayed continuation routes and transported only componentwise, so it records comparison without turning one substrate into the other.

The local NameCert row then applies the naming discipline: what is present is named by M, G, A, H, C, P, N, and what is absent remains outside the packet. This is exactly the finite concrete surface claimed by the vision reading. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditUp}}$.

Constructive story: A TwinSubstrateAuditUp packet is generated from a MetaCIC audit row, a GroundCompiler audit row, a roundtrip alignment row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditUp}}, \mathsf{seed})$

Scope closed: Definition 846.1, Definition 846.2, Definition 846.3, and Definition 846.4 bind the finite audit handoff packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditUp}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close subject reduction, prove compiler self-hosting, turn compiler reports into proof evidence, import hidden host input, collapse the two audit substrates, provide Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, audit-row exactness, alignment readback, transport, continuation, provenance, naming, and roundtrip non-escape for the same finite packet.

Chapter 847

A Concrete Naming Certificate for $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$

The $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ packet realizes the twin-substrate audit architecture of chapter 9392 as one finite NameCert surface. It sits beside the GroundCompiler audit-map packet of chapter 836, the closed-window substitution route of chapter 824, the MetaCIC normalization frontier of chapter 892, and the two-history comparison packet of chapter 668. Those siblings already name one-substrate audit rows, normalization-frontier rows, or paired observation rows; this chapter names the finite coupling boundary that lets MetaCIC and GroundCompiler be read together without collapsing either substrate into the other.

Definition 847.1 (Twin-substrate audit-coupling carrier)**.** *A* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, G, R, L, C, H, T, P, N)$$

with the following visible rows:

1. *M is the MetaCIC audit row, recording closed-CIC reduction, closedness, substitution, normalisation, confluence, decidability, consistency, and subject-reduction boundary references as MetaCIC-side audit data;*
2. *G is the GroundCompiler audit row, recording event-flow, channel legality, generated recognizer, certificate gate, report boundary, non-input, and non-evidence references as compiler-side audit data;*
3. *R is the route classifier row, separating MetaCIC reads, GroundCompiler reads, and paired reads before any downstream consumer receives a coupled packet;*
4. *L is the coupling ledger, recording which public project-level audit claim reads both M and G and which substrate boundary remains attached to each read;*
5. *C is the non-collapse classifier row, marking MetaCIC proof evidence, kernel acceptance, and closed-CIC theorem content as unavailable from G, and marking host compiler trust, hidden input, source-channel legality, generated recognizers, and report artifacts as unavailable from M;*
6. *H records componentwise* hsame *transports for MetaCIC rows, GroundCompiler rows, route classifiers, coupling ledger entries, non-collapse classifiers, continuation rows, provenance, and local naming;*

7. *T records displayed* Cont *routes by which a consumer inspects M, G, R, L, C in finite order;*
8. *P is the* Pkg *provenance over chapter 9392, chapter 836, chapter 824, chapter 668,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}}$ *row.*

The carrier has no coordinate for converting GroundCompiler reports into MetaCIC proof evidence, no coordinate for reading MetaCIC closed-CIC facts as compiler input channels, no shared hidden equality between the two substrates, and no host-side project audit oracle outside the displayed packet.

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditCouplingUp.TwinSubstrateAuditCouplingTasteGate single carrier alignment`

Theorem 847.2 (Twin-substrate audit-coupling NameCert obligations)**.** *For every accepted* TwinSubstrateAuditCou *carrier*

$$U = (M, G, R, L, C, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}}$ *surface consists exactly of MetaCIC audit readback through M, GroundCompiler audit readback through G, route classification through R, paired coupling-ledger replay through L, non-collapse classification through C, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditCouplingUp.TwinSubstrateAuditCouplingNameCertObligations`

Proof. Project the finite carrier. The first two rows expose the two audit substrates, and the next three rows record how they may be read together.

Read M and G first. The MetaCIC row carries closed-CIC audit content, while the GroundCompiler row carries compiler event-flow, recognizer, gate, report, non-input, and non-evidence content.

Then inspect R, L, C. The route classifier separates one-substrate reads from paired reads, the ledger records which public audit claim consumes both rows, and the non-collapse classifier keeps proof-evidence and compiler-channel claims on their proper sides.

Finally H, T, P, N replay transport, continuation, provenance, and naming over the same finite rows. Since the carrier lists no hidden equality or project oracle, the displayed rows exhaust the local NameCert obligations. □

Theorem 847.3 (Twin-substrate audit-coupling MetaCIC route)**.** *Every MetaCIC-facing public read from an accepted* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *carrier factors through the MetaCIC row M, the route classifier R, the coupling ledger L, the non-collapse row C, and the displayed transport and continuation rows H, T. Such a read may cite the GroundCompiler row G only as paired audit context and cannot turn compiler reports, generated recognizers, hidden-input exclusions, or source-channel legality into MetaCIC proof evidence.*

Proof. Start with a MetaCIC-facing read. Carrier projection gives M as the only closed-CIC audit row and R as the classifier that records the read as MetaCIC-facing.

The ledger row L may attach a paired GroundCompiler context to the same project-level audit claim. That attachment is a ledgered reference to G, not a projection from G into proof evidence.

Use the non-collapse row C to preserve the boundary. It explicitly records that reports, recognizers, hidden-input exclusions, and source-channel legality remain compiler-side audit data.

Transport and continuation through H and T replay these rows componentwise. Therefore a MetaCIC-facing read remains a read of M with bounded paired context, not a conversion of GroundCompiler artifacts into MetaCIC proof evidence. □

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditCouplingUp.TwinSubstrateAuditCouplingCarrier metacic route`

Theorem 847.4 (Twin-substrate audit-coupling GroundCompiler route)**.** *Every GroundCompiler-facing public read from an accepted* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *carrier factors through the GroundCompiler row G, the route classifier R, the coupling ledger L, the non-collapse row C, and the displayed transport and continuation rows H, T. Such a read may cite the MetaCIC row M only as paired audit context and cannot turn closed-CIC reduction, substitution, normalisation, confluence, decidability, consistency, or subject-reduction boundary rows into host compiler trust, hidden input, source-channel legality, or generated recognizer evidence.*

Proof. Begin with a GroundCompiler-facing read. Carrier projection gives G as the only compiler-side audit row and R as the classifier that marks the read as GroundCompiler-facing.

The ledger L may require the read to be accompanied by a MetaCIC audit row when the project-level claim mentions closed-CIC content. This accompaniment does not make M a compiler channel or a recognizer source.

The row C records the separating clauses. Closed-CIC reduction, substitution, normalisation, confluence, decidability, consistency, and subject-reduction boundaries remain MetaCIC-side data; host compiler trust, hidden input, source-channel legality, and generated recognizers remain GroundCompiler-side obligations.

Rows H and T preserve that separation under public transport and replay. Hence the GroundCompiler route is a read of G with paired MetaCIC context, not a compiler-side extraction from MetaCIC theorem rows. □

Theorem 847.5 (Twin-substrate audit-coupling non-collapse)**.** *An accepted* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *carrier permits paired project-level audit reads only through the ledger row L and the route classifier row R. It does not identify M with G, does not make either substrate a subtype of the other, and does not export a common equality, shared proof evidence, shared compiler input, or host-side audit oracle outside the packet.*

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditCouplingUp.TwinSubstrateAuditCouplingCarrier noncollapse`

Proof. Project $U = (M, G, R, L, C, H, T, P, N)$. The two substrate rows are separate coordinates, and the only paired-read coordinates are R and L.

If a consumer asks for a project-level audit read, R classifies whether the read is MetaCIC-facing, GroundCompiler-facing, or paired. The ledger L records paired reads without changing the underlying row identities.

The non-collapse row C blocks the remaining identifications. It keeps proof evidence, kernel acceptance, theorem content, compiler trust, hidden input, source-channel legality, recognizers, and reports attached to their substrate rows.

Because the packet contains no equality oracle, common proof-evidence coordinate, common compiler-input coordinate, or host-side audit oracle, the two substrates remain coupled but not collapsed. □

Theorem 847.6 (Twin-substrate audit-coupling public export)**.** *Every public export of an accepted* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *carrier is a finite packet export of* $M, G, R, L, C, H, T, P, N$*. The export may be consumed by audit-map, substitution-route, observation-coupling, or self-compilation readers only as a NameCert surface that preserves the MetaCIC/GroundCompiler distinction recorded in chapter 9392.*

Proof. Use the NameCert obligations. They list the full public surface as the substrate rows, route classifier, coupling ledger, non-collapse classifier, transport, continuation, provenance, and local name.

The sibling chapters provide permitted consumer shapes. chapter 836 reads GroundCompiler audit rows, chapter 824 reads a closed-window MetaCIC route boundary, and chapter 668 shows how a finite coupling ledger can serve a downstream reader without adding a global coordinate.

Normalization-frontier consumers may read the coupled export only after the MetaCIC row M, route classifier R, coupling ledger L, and non-collapse row C are displayed as the finite MetaCIC-side boundary of chapter 892.

An export to those consumers therefore carries the displayed packet and its provenance. It does not export a collapsed substrate, a hidden equality, or an ambient audit oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ packet is generated from a MetaCIC audit row, a GroundCompiler audit row, route classifiers, a coupling ledger, non-collapse classifiers, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 847.1, Definition 847.2, Definition 847.3, Definition 847.4, Definition 847.5, and Definition 847.6 bind the finite paired MetaCIC/GroundCompiler audit-coupling packet, route classifiers, ledger, non-collapse boundary, and public export surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close MetaCIC subject reduction, GroundCompiler host trust, self-hosting tower correctness, Lean verification, audit cleanliness, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, MetaCIC route readback, GroundCompiler route readback, paired ledger replay, non-collapse classification, public export exactness, transport, continuation, provenance, and naming for the same finite packet.

Chapter 848

TwinSubstrateAuditSynthesis NameCert construction

The $\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}$ packet realizes the twin-substrate synthesis of chapter 9392 as a finite NameCert object whose audit rows remain substrate-specific. It sits beside the MetaCIC audit synthesis of **??**, the GroundCompiler audit map of chapter 836, and the audit-map frontier packet of chapter 833: those siblings provide the two audit sources and the shared frontier discipline, while this chapter names their paired finite synthesis row.

Definition 848.1 (Twin-substrate audit-synthesis carrier)**.** *A* $\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, G, A, H, C, P, N)$$

with the following displayed rows:

1. *M is a MetaCIC audit-synthesis row, read through* **??***;*
2. *G is a GroundCompiler audit-map row, read through chapter 836;*
3. *A is an audit-frontier alignment row, read through chapter 833, that records which MetaCIC boundary and which GroundCompiler boundary are being paired;*
4. *H records componentwise* hsame *transports for the MetaCIC row, GroundCompiler row, frontier alignment row, continuation row, package row, and local naming row;*
5. *C records displayed* Cont *routes by which consumers read M, G, and A without identifying their evidence kinds;*
6. *P is the* Pkg *provenance over chapter 9392,* **??***, chapter 836, chapter 833,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}}$ *row.*

The classifier accepts exactly the packets whose public reads factor through $S = (M, G, A, H, C, P, N)$ and whose frontier alignment keeps MetaCIC proof-theoretic residue separate from GroundCompiler report, recognizer, channel, and hidden-input boundaries.

Lean4 ✓ `BEDC.Derived.TwinSubstrateAuditSynthesisUp.TwinSubstrateAuditSynthesisTasteGate single carrier alignment`

Theorem 848.2 (TwinSubstrateAuditSynthesis NameCert obligations)**.** *For every accepted* TwinSubstrateAuditSyn *carrier*

$$S = (M, G, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}}$ *surface consists exactly of MetaCIC audit-synthesis readback through* M*, GroundCompiler audit-map readback through* G*, audit-frontier alignment through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 848.1. Its coordinates are precisely the two audit rows, the frontier alignment row, the transport row, the continuation row, the provenance row, and the local naming row.

Read the substrate rows separately. The MetaCIC row M carries the audit-synthesis surface, while the GroundCompiler row G carries the compiler-facing audit-map surface. The alignment row A names a shared frontier position, but it does not convert one row into the other.

Finally, H, C, P, N transport, replay, package, and name only those displayed coordinates. Since the carrier has no hidden equality, report-to-proof conversion, or compiler-input oracle, the listed rows exhaust the NameCert obligation surface. □

Theorem 848.3 (TwinSubstrateAuditSynthesis cross-map non-escape)**.** *Every public cross-map read from an accepted* $\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}$ *carrier factors through the displayed rows* M, G, A, H, C, P, N *of Definition 848.1. Such a read cannot turn a GroundCompiler report, recognizer, channel row, or hidden-input refusal into MetaCIC proof evidence, and it cannot turn a MetaCIC obstruction, closed-calculus theorem row, or subject-reduction residue into GroundCompiler input or compiler-side evidence.*

Proof. Start with the obligation surface of Definition 848.2. Every public read must first choose the MetaCIC row, the GroundCompiler row, or the frontier alignment row and then pass through the displayed continuation and transport rows.

For a MetaCIC-facing read, the consumer receives M with A, H, C, P, N as finite routing data. The GroundCompiler row G may be present as paired audit context, but the packet contains no coordinate that reclassifies report, recognizer, channel, or hidden-input data as proof evidence.

For a GroundCompiler-facing read, the consumer receives G with the same finite routing discipline. The MetaCIC row M may be cited as the paired audit-synthesis boundary, but the packet contains no coordinate that reclassifies closed-calculus theorem data, obstruction data, or subject-reduction residue as compiler input.

Thus the ledger cannot escape the displayed packet. Coordinate exhaustion over M, G, A, H, C, P, N preserves both refusal boundaries while allowing a finite paired audit read. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}$ packet is generated from a MetaCIC audit-synthesis row, a GroundCompiler audit-map row, an audit-frontier alignment row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}}$, seed)

Scope closed: Definition 848.1, Definition 848.2, and Definition 848.3 bind the finite paired audit-synthesis packet and its cross-map non-escape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditSynthesis}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close MetaCIC subject reduction, GroundCompiler host trust, compiler self-reference, Lean verification, audit cleanliness, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, MetaCIC audit-synthesis readback, Ground-Compiler audit-map readback, frontier-alignment exactness, transport, continuation, provenance, naming, and cross-map non-escape for the same finite packet.

Chapter 849

A Concrete Naming Certificate for $\mathsf{AuditReexportNamespace}^{\uparrow}$

The $\mathsf{AuditReexportNamespace}^{\uparrow}$ packet realizes the audit reexport namespace layer of chapter 9392 as a finite NameCert carrier. It sits beside the twin-substrate packet of chapter 845 and the audit-map interface of chapter 806: those siblings name the two-substrate architecture and the generic audit-map consumer surface, while this chapter names the concrete namespace-addressing packet through which downstream audit readers cite the two reexport namespaces without collapsing them.

Definition 849.1 (Audit reexport namespace carrier)**.** *An* $\mathsf{AuditReexportNamespace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, G, S, L, B, R, P, N)$$

with the following visible rows:

1. *M is the* `BEDC.MetaCIC.Audit` *namespace row, recording the MetaCIC audit exports that may be cited as theorem-side audit addresses;*
2. *G is the* `BEDC.GroundCompiler.Audit` *namespace row, recording the GroundCompiler audit exports that may be cited as compiler-side audit addresses;*
3. *S records the substrate tag rows that classify each namespace address as MetaCIC-facing, GroundCompiler-facing, or a paired project-level citation;*
4. *L is the export-address ledger, listing the finite public addresses exposed by M and G and the local citation names by which consumers reach them;*
5. *B is the separation witness, marking compiler audit rows as unavailable as MetaCIC theorem rows and MetaCIC theorem rows as unavailable as compiler audit rows;*
6. *R is the consumer route, recording which downstream audit-map, twin-substrate, or self-compilation reader consumes the namespace packet and through which displayed continuation;*
7. *P is the* Pkg *provenance over chapter 9392, chapter 845, chapter 806,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditReexportNamespace}^{\uparrow}}$ *row.*

The carrier predicate is $\mathsf{AuditReexportNamespacePacket}(A)$*. Its public rows are exactly the two namespace rows, substrate tags, export-address ledger, separation witness, consumer route, package provenance, and local naming row.*

Theorem 849.2 (Audit reexport namespace NameCert obligations)**.** *For every accepted* $\mathsf{AuditReexportNamespace}^{\uparrow}$ *carrier*

$$A = (M, G, S, L, B, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AuditReexportNamespace}^{\uparrow}}$ *surface consists of five displayed obligation rows: carrier admission for the finite packet of Definition 849.1; namespace address readback for* M, G, L*; substrate-tag exactness for* S*; separation and nonconversion readback for* B*; and consumer-route, package, and local-name stability for* R, P, N *under the displayed* hsame *and* Cont *rows.*

Lean4 ✓ `BEDC.Derived.AuditReexportNamespaceUp.AuditReexportNamespaceTasteGate single carrier alignment`

Proof. Project the finite carrier. The tuple exposes the two namespace rows, the tag row, the ledger, the separation row, the consumer route, provenance, and local naming.

The first obligation is carrier admission for exactly those rows. The second reads the public namespace addresses through M, G, L, so a citation has a displayed export address before a consumer can use it.

The third and fourth obligations inspect S and B. The tag row classifies each address by substrate, and the separation witness records that the two address classes are not conversion rules.

Finally R, P, N bind the downstream route, package provenance, and local name to the same packet. Componentwise transport and continuation replay these rows without adding a hidden namespace equality, so the five displayed obligations exhaust the NameCert surface. □

Theorem 849.3 (Audit reexport namespace non-collapse)**.** *Every consumer of an accepted* AuditReexportNamespac *carrier may cite* `BEDC.MetaCIC.Audit` *through* M *and* `BEDC.GroundCompiler.Audit` *through* G *only along the displayed address ledger* L*, substrate tag row* S*, separation witness* B*, and route* R*. Such a citation cannot convert a compiler audit row into a MetaCIC theorem row, cannot convert a MetaCIC theorem row into a compiler audit row, and cannot export a shared hidden equality between the two namespace layers.*

Proof. Start from the NameCert obligations. They force every consumer citation to pass through the visible namespace rows and export-address ledger.

If the cited address is MetaCIC-facing, S tags it as a MetaCIC address and B records that compiler audit evidence is not available as a theorem row. The consumer route R may attach paired compiler context, but only as a ledgered citation through G, L.

If the cited address is GroundCompiler-facing, S tags it as a compiler-side audit address and B records that theorem rows are not compiler audit evidence. The paired MetaCIC context remains a cited namespace row, not a compiler artifact.

Because the packet has no coordinate identifying M with G and no conversion row between their evidence classes, the namespace layer gives two citeable audit surfaces while preserving the noncollapse boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditReexportNamespace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AuditReexportNamespace}^{\uparrow}$ packet is generated from MetaCIC and GroundCompiler audit namespace rows, substrate tags, an export-address ledger, a separation witness, a consumer route, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditReexportNamespace}^{\uparrow}}$, seed)

Scope closed: Definition 849.1, Definition 849.2, and Definition 849.3 bind the finite two-substrate namespace packet, its five-row NameCert surface, and the boundary forbidding compiler-to-theorem or theorem-to-compiler conversion.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AuditReexportNamespace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, audit cleanliness, axiom cleanliness, bridge checking, self-compilation correctness, or conversion between MetaCIC theorem evidence and GroundCompiler audit evidence.

Upgrade path: Obligation closure requires checked carrier admission, namespace address readback, substrate-tag exactness, separation readback, consumer-route stability, provenance, and local naming for the same finite packet.

Chapter 850

A Concrete Naming Certificate for $\mathsf{TwinSubstrateBridge}^{\uparrow}$

The $\mathsf{TwinSubstrateBridge}^{\uparrow}$ packet realizes the twin-substrate architecture of chapter 9392 as a finite bridge certificate. It sits beside the MetaCIC audit synthesis of **??**, the GroundCompiler audit map of chapter 836, and the audit-map frontier packet of chapter 833. Those sibling chapters provide the two substrate rows and the frontier boundary; this chapter records the bridge packet that lets a consumer read them together without identifying their audit ledgers.

Definition 850.1 (TwinSubstrateBridge carrier)**.** *A* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, F, R, L, H, C, P, N)$$

with the following visible rows:

1. *M is the MetaCIC audit-synthesis row consumed from* **??***;*
2. *G is the GroundCompiler audit-map row consumed from chapter 836;*
3. *F is the audit-map frontier packet row consumed from chapter 833;*
4. *R is the bridge-route classifier separating MetaCIC-facing, GroundCompiler-facing, and frontier-facing reads;*
5. *L is the two-sided bridge ledger recording which substrate row each read consumes;*
6. *H stores componentwise* hsame *transport for M, G, F, R, L;*
7. *C stores displayed* Cont *routes through the bridge rows;*
8. *P records* Pkg *provenance over the sibling chapters, the route classifier, the ledger, and the structural rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}$ *row.*

The carrier has no row identifying M with G, no common audit equality, no compiler report promoted to MetaCIC proof evidence, and no MetaCIC derivation promoted to compiler acceptance.

Lean4 ✓ `BEDC.Derived.TwinSubstrateBridgeUp.TasteGate.TwinSubstrateBridgeTasteGate single carrier alignment`

Theorem 850.2 (TwinSubstrateBridge NameCert obligations)**.** *For every accepted* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *carrier $B = (M, G, F, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}$ *surface consists exactly*

of carrier admission, MetaCIC audit-synthesis readback through M, GroundCompiler audit-map readback through G, frontier readback through F, bridge-route classification through R, two-sided ledger exactness through L, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TwinSubstrateBridgeUp.TwinSubstrateBridgeCarrier namecert obligations`

Proof. Project the carrier in Definition 850.1. The rows $M, G, F, R, L, H, C, P, N$ are the complete finite packet, so carrier admission and row readback are coordinate projections.

The route classifier R is read before the ledger L. A MetaCIC-facing read is therefore sent to M and may cite F only as frontier context; a GroundCompiler-facing read is sent to G and may cite F only as frontier context. The ledger row records this choice rather than replacing either substrate row.

The structural rows preserve the same packet. Componentwise hsame transport in H moves M, G, F, R, L without merging them, Cont replay in C exposes only displayed routes, and P, N repack the same rows as provenance and local naming. Hence the NameCert surface is exactly the stated finite obligation list. □

Theorem 850.3 (TwinSubstrateBridge carrier admission)**.** *Let $B = (M, G, F, R, L, H, C, P, N)$ be an accepted* TwinSubstrateBridge$^{\uparrow}$ *carrier. Carrier admission for B is exactly the finite projection that exposes the MetaCIC audit-synthesis row M, the GroundCompiler audit-map row G, and the audit-map frontier row F before any bridge route is consumed. No accepted carrier can expose R, L, H, C, P, N as bridge evidence while omitting one of M, G, F.*

Lean4 ✓ `BEDC.Derived.TwinSubstrateBridgeUp.TwinSubstrateBridgeCarrier carrier admission`

Proof. Project the carrier of Definition 850.1. The first three coordinates are the substrate rows that make the bridge meaningful: M supplies the MetaCIC side, G supplies the GroundCompiler side, and F supplies the frontier boundary between them.

The later coordinates are dependent public rows. The route classifier R can classify only requests between the displayed substrate sides, and the ledger L can record only reads whose substrate side has already been exposed.

The structural rows H, C, P, N preserve, replay, source, and name the same displayed packet. Since none of them manufactures a missing substrate coordinate, accepted carrier admission must expose M, G, F before the bridge evidence can be read. □

Theorem 850.4 (TwinSubstrateBridge replay stability)**.** *For every accepted* TwinSubstrateBridge$^{\uparrow}$ *carrier $B = (M, G, F, R, L, H, C, P, N)$, componentwise transport through H and displayed replay through C preserve the route separation recorded by R and L. A replayed MetaCIC-facing request still consumes M before frontier context, a replayed GroundCompiler-facing request still consumes G before frontier context, and a replayed bridge-facing request still compares the sides only through F and L.*

Proof. Read the route classifier R and ledger L before replay. They determine whether a request is MetaCIC-facing, GroundCompiler-facing, or bridge-facing, and they record which substrate row the request may consume.

Transport through H is componentwise. It moves the displayed rows M, G, F, R, L along hsame without adding a row that identifies M with G or changes the route assigned by R.

Replay through C follows the displayed continuation route after transport. Thus a MetaCIC-facing replay remains on the M side, a GroundCompiler-facing replay remains on the G side, and a

bridge-facing replay reaches both sides only through the frontier and ledger rows. The separation is therefore stable under the structural rows of the carrier. □

Theorem 850.5 (TwinSubstrateBridge ledger exactness)**.** *Every public read of an accepted* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *carrier* $B = (M, G, F, R, L, H, C, P, N)$ *factors through the route classifier* R *and the two-sided ledger* L *before a substrate row is consumed. Consequently, public consumers cannot read* M*,* G*, or* F *as unclassified substrate data; the only exported substrate access is the route- and ledger-recorded access named by the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.TwinSubstrateBridgeUp.TwinSubstrateBridgeCarrier ledger exactness`

Proof. Start with any public read of the packet. The NameCert obligations in Definition 850.2 expose the classifier and ledger as part of the finite public surface, so the read is first assigned a route by R and then recorded by L.

Only after this routing step may the read consume a substrate row. The MetaCIC route reads M, the GroundCompiler route reads G, and the bridge route uses F together with the ledger entry that records which side is being compared.

The remaining rows do not bypass this order. Componentwise transport in H, displayed replay in C, provenance in P, and local naming in N preserve the same routed read. Hence every exported substrate access is exactly the access registered by R, L and no unclassified read of M, G, F is public. □

Theorem 850.6 (TwinSubstrateBridge audit separation)**.** *Every public consumer of an accepted* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *carrier preserves the separation between the MetaCIC row* M *and the GroundCompiler row* G*. A MetaCIC-facing consumer may read* M, F, R, L, H, C, P, N*; a GroundCompiler-facing consumer may read* G, F, R, L, H, C, P, N*; and a bridge-facing consumer may compare the two only through the displayed frontier row* F *and ledger row* L*.*

Proof. Begin with the route classifier R. It assigns each consumer request to the MetaCIC-facing, GroundCompiler-facing, or bridge-facing route before any audit row is exposed.

For the MetaCIC route, the only audit row supplied by the carrier is M. The frontier row F and ledger row L can record compiler-side context, but no row of the packet converts G into proof evidence for M.

For the GroundCompiler route, the only compiler-facing audit row supplied by the carrier is G. The same frontier and ledger rows can record MetaCIC context, but no row converts M into compiler acceptance evidence.

The bridge route reads both sides only through F, L, H, C, P, N. These rows display comparison context, transport, continuation, provenance, and naming; they do not add a common substrate equality. Thus every public consumer preserves the two audit ledgers as distinct coordinates of the same finite packet. □

Theorem 850.7 (TwinSubstrateBridge non-escape)**.** *An accepted* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *carrier exports no hidden common substrate, host audit oracle, compiler proof checker, MetaCIC evaluator, collapsed equality between* M *and* G*, or frontier-closing theorem. Every exported read factors through the finite rows* $M, G, F, R, L, H, C, P, N$ *and through the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}$ *surface.*

Proof. Read the packet through Definition 850.2. The local NameCert row exposes only the displayed carrier, sibling readbacks, route classifier, ledger, transport, continuation, provenance, and local naming rows.

Apply Definition 850.6. Any request involving both substrates must pass through the frontier and ledger rows, so the packet cannot manufacture a common equality or a shared audit oracle.

The sibling chapters provide only their own named surfaces: MetaCIC synthesis, GroundCompiler audit mapping, and audit-map frontier accounting. Since the bridge carrier has no coordinate for a compiler proof checker, MetaCIC evaluator, hidden host oracle, or frontier-closing theorem, none of those objects can be exported by a consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{TwinSubstrateBridge}^{\uparrow}$ packet is generated from a MetaCIC audit-synthesis row, a GroundCompiler audit-map row, an audit-map frontier row, a bridge-route classifier, a two-sided bridge ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 850.1, Definition 850.2, Definition 850.6, and Definition 850.7 bind the finite bridge carrier, local NameCert obligation surface, audit separation, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, a public standard bridge, a common substrate equality, compiler proof checking, MetaCIC evaluation, frontier closure, or host audit automation.

Upgrade path: Formal closure requires checked carrier admission, substrate readback, route classification, ledger exactness, componentwise transport, continuation replay, provenance, naming, and non-escape for the same finite bridge packet.

Chapter 851

A Concrete Naming Certificate for TwinSubstrateBoundaryRouter$^{\uparrow}$

The TwinSubstrateBoundaryRouter$^{\uparrow}$ packet realizes the twin-substrate architecture of chapter 9392 as a request-level boundary router. It reads the MetaCIC audit surface of chapter 9385, the GroundCompiler audit map of chapter 9407, and the finite architecture packet of chapter 845. The point of this chapter is narrower than architecture naming: it binds the local certificate that decides which substrate may answer each public request, and it keeps failed cross-substrate discharges visible.

Definition 851.1 (Twin-substrate boundary router carrier)**.** *A* TwinSubstrateBoundaryRouter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, G, Q, D, B, H, C, P, N)$$

with the following visible rows:

1. *M is the MetaCIC request row, recording closed-CIC audit obligations, proof-theoretic frontiers, theorem-family references, and MetaCIC-side refusal data;*
2. *G is the GroundCompiler request row, recording audit-map rows, event-flow obligations, recognizer boundaries, report boundaries, and compiler-side refusal data;*
3. *Q is the public request classifier row, separating MetaCIC requests, GroundCompiler requests, paired-context requests, and rejected cross-substrate requests before any answer is exposed;*
4. *D is the discharge row, recording that a MetaCIC request can be discharged only by displayed MetaCIC data from M and that a GroundCompiler request can be discharged only by displayed GroundCompiler data from G;*
5. *B is the boundary ledger, recording the exact non-discharge clauses for attempts to answer a MetaCIC request by GroundCompiler rows or a GroundCompiler request by MetaCIC rows;*
6. *H records componentwise* hsame *transports for request rows, classifier rows, discharge rows, boundary-ledger rows, continuation rows, provenance, and local naming;*
7. *C records displayed* Cont *routes by which a public consumer replays Q, D, B after projecting M or G;*
8. *P is the* Pkg *provenance over chapter 9392, chapter 9385, chapter 9407, chapter 845,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}}$ *row.*
The carrier has no coordinate for treating compiler reports as MetaCIC proof evidence, no coordinate for treating MetaCIC proof-theoretic rows as compiler-channel evidence, and no hidden global resolver outside the finite packet.

Theorem 851.2 (Twin-substrate boundary-router NameCert obligations)**.** *For every accepted* $\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}$ *carrier*

$$R = (M, G, Q, D, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}}$ *surface consists exactly of MetaCIC request readback through M, GroundCompiler request readback through G, public request classification through Q, substrate-local discharge through D, cross-substrate refusal through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the finite BHist carrier. The only request rows are M and G, the only classifier is Q, and the only discharge and refusal rows are D and B.

Read the NameCert surface in that projected order. The classifier Q chooses the visible request class before any answer is exposed, while D records the substrate-local discharge row available for the chosen class.

Componentwise hsame transport through H preserves the same projected rows, and Cont route replay through C exposes only the displayed classifier, discharge, and boundary-ledger sequence. The Pkg row P supplies the listed provenance anchors, and N binds the local name.

Since the carrier contains no additional resolver coordinate, those rows exhaust the local NameCert obligations. □

Theorem 851.3 (Twin-substrate boundary-router disjoint discharge)**.** *In every accepted* TwinSubstrateBoundaryRo *carrier, a MetaCIC request cannot be discharged by GroundCompiler rows, and a GroundCompiler request cannot be discharged by MetaCIC rows. Paired-context requests may cite both substrate rows only as context recorded by Q and B, not as a discharge conversion.*

Proof. Begin with finite carrier projection. A MetaCIC request is classified by Q as reading M, and a GroundCompiler request is classified by Q as reading G.

For the MetaCIC case, the discharge row D points back to M. The boundary ledger B records that GroundCompiler audit-map rows, event-flow rows, recognizer rows, and report rows are not proof evidence for that request.

For the GroundCompiler case, the discharge row D points back to G. The same boundary ledger records that MetaCIC closed-CIC rows, proof-theoretic frontiers, and theorem-family references are not compiler-channel evidence for that request.

The componentwise hsame transports in H and the Cont route replay in C preserve the chosen substrate row and its refusal clause. Provenance P and local naming N therefore bind disjoint discharge rather than a cross-substrate resolver. □

Theorem 851.4 (Twin-substrate boundary-router non-escape)**.** *Every public route exported by an accepted* $\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}$ *carrier factors through the request classifier Q, the substrate-local discharge row D, the boundary ledger B, componentwise transport H, continuation replay C, package provenance P, and local naming N. The export cannot introduce a hidden request class, a hidden discharge row, or an ambient oracle that answers across the MetaCIC/GroundCompiler boundary.*

Proof. Use the NameCert obligations to expose the public route. Any consumer must first project the finite packet and then pass through Q, D, B, H, C, P, N.

The classifier Q makes the request class visible. The discharge row D ties an accepted answer to the matching substrate row, while B records the non-discharge clause for the opposite substrate.

Transport and continuation do not add coordinates. They replay the same rows componentwise and in displayed finite order, so they cannot manufacture a hidden request class or an outside discharge witness.

Package provenance and local naming close the route over the same finite carrier. With no ambient resolver coordinate in Definition 851.1, public export remains non-escaping. □

Theorem 851.5 (Twin-substrate boundary-router public surface)**.** *The public surface of an accepted* $\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}$ *carrier is exactly the finite packet export of* $M, G, Q, D, B, H, C, P, N$*. Downstream self-compilation audit readers may use it to route requests between MetaCIC audit rows and GroundCompiler audit rows, but only with the substrate-local discharge and refusal clauses displayed in the exported NameCert packet.*

Proof. Start from carrier projection. The public packet exposes the two substrate request rows together with classifier, discharge, refusal, transport, continuation, provenance, and naming rows.

The route replay row C orders consumer access to the classifier Q, the discharge row D, and the boundary ledger B. Thus a reader can decide where a request lands and can inspect why the opposite substrate is not an answer.

The Pkg row P anchors the export to the twin-substrate vision, the MetaCIC audit surface, the GroundCompiler audit map, and the concrete architecture packet. The local NameCert row N names the exported surface without adding a hidden resolver.

Consequently downstream self-compilation readers receive request-level routing between the two audit substrates, not a merged substrate and not an unledgered cross-substrate discharge. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}$ packet is generated from MetaCIC and GroundCompiler request rows, a public request classifier, a substrate-local discharge row, a cross-substrate boundary ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 851.1, Definition 851.2, Definition 851.3, Definition 851.4, and Definition 851.5 bind the finite request-level boundary router, disjoint discharge rule, non-escape boundary, and public export surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close MetaCIC proof-theoretic frontiers, GroundCompiler self-hosting, compiler report soundness, Lean verification, audit cleanliness, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request classification, substrate-local discharge, boundary-ledger exactness, public export exactness, transport, continuation, provenance, and naming for the same finite packet.

Chapter 852

A Concrete Naming Certificate for $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$

The $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$ packet realizes the twin-substrate synthesis boundary of chapter 9392 as a finite NameCert handshake. It sits beside the GroundCompiler audit map of chapter 836 and the twin-substrate audit coupling of chapter 847: those chapters expose the compiler-facing audit map and the paired audit coupling, while this chapter names the narrower handshake that lets a reader inspect both audit surfaces without converting one substrate into the other.

Definition 852.1 (Twin-substrate audit-handshake carrier)**.** *A* $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, G, B, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC audit row, carrying closed-CIC theorem, reduction, substitution, normalisation, confluence, decidability, consistency, and obstruction-boundary references;*
2. *G is the GroundCompiler audit row, carrying event-flow, channel legality, generated recognizer, report-boundary, hidden-input exclusion, and non-evidence references;*
3. *B is the handshake boundary row, naming the finite public claim that requires both audit surfaces while preserving their separate subject matter;*
4. *R is the route classifier row, separating MetaCIC-facing reads, GroundCompiler-facing reads, and paired handshake reads;*
5. *H records componentwise* hsame *transports for the MetaCIC row, GroundCompiler row, handshake boundary, route classifier, continuation rows, package provenance, and local naming row;*
6. *C records the displayed* Cont *routes by which a public consumer reads M, reads G, and then reads B without adding a cross-substrate conversion route;*
7. *P is the* Pkg *provenance over chapter 9392, chapter 836,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}}$ *row.*

The carrier has no coordinate for turning a GroundCompiler report into MetaCIC proof content, no coordinate for turning a MetaCIC theorem into compiler evidence, and no hidden equality that identifies M with G.

Theorem 852.2 (Twin-substrate audit-handshake separation)**.** *For every accepted* $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$ *carrier*

$$U = (M, G, B, R, H, C, P, N),$$

the packet exposes both audit rows M and G through the displayed boundary row B, route classifier R, componentwise hsame *rows H, displayed* Cont *routes C, package provenance P, and local* NameCert *row N. A public handshake read may cite both substrates as audit context, but it cannot convert a compiler report into MetaCIC proof content and cannot convert a MetaCIC theorem into compiler evidence.*

Proof. Project the finite carrier. The MetaCIC row M and the GroundCompiler row G are distinct coordinates, and the handshake boundary row B records only the public claim that needs both of them.

Inspect the route classifier R. A MetaCIC-facing read keeps proof-theoretic content attached to M, and a GroundCompiler-facing read keeps compiler-pipeline content attached to G. A paired handshake read may mention both rows only after R has recorded that it is paired.

Transport and replay are componentwise through H and C. These rows preserve the displayed coordinates and provide no route that maps compiler reports to proof content or maps theorem rows to compiler evidence.

Finally read the provenance and naming rows P, N. They package the finite handshake over the cited audit surfaces, so any conversion between the two substrates would require a coordinate outside U. Since the carrier has no such coordinate, the two audit surfaces are exposed together without collapse. □

Closure status for $\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}}$.

Constructive story: A $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$ packet is generated from a MetaCIC audit row, a GroundCompiler audit row, a finite handshake boundary, a route classifier, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}}$, seed)

Scope closed: Definition 852.1 and Definition 852.2 bind the finite twin-substrate handshake carrier and its non-conversion boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close MetaCIC frontier obligations, GroundCompiler host trust, theorem extraction from compiler reports, compiler evidence extraction from MetaCIC theorems, Lean verification, audit cleanliness, axiom cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, route classification, transport, continuation replay, provenance, naming, and non-conversion exactness for the same finite handshake packet.

Theorem 852.3 (Twin-substrate audit-handshake route-classifier trichotomy)**.** *Let*

$$U = (M, G, B, R, H, C, P, N)$$

be an accepted $\mathsf{TwinSubstrateAuditHandshake}^{\uparrow}$ *carrier. If a public audit read q is exposed by U, then the route classifier row R places q in exactly one of the following three displayed route classes:*

1. *a MetaCIC-facing route, whose readable substrate row is M and whose public content remains closed-CIC audit content;*
2. *a GroundCompiler-facing route, whose readable substrate row is G and whose public content remains compiler audit content;*

3. *a paired handshake route, whose readable boundary is B and whose context may cite both M and G without converting either row into the other.*

In all three cases, the transport, continuation, provenance, and naming rows H, C, P, N replay the selected classifier case componentwise. No accepted public read can pass from the G row to MetaCIC proof content or from the M row to compiler evidence.

Proof. Project the accepted carrier using Definition 852.1. The public rows available to a read are the MetaCIC audit row M, the GroundCompiler audit row G, the handshake boundary row B, the route classifier row R, and the structural rows H, C, P, N.

The row R is displayed as the finite classifier separating MetaCIC-facing reads, GroundCompiler-facing reads, and paired handshake reads. Therefore a public audit read q must be inspected by one of those three classifier entries. The first entry reads M as closed-CIC audit data; the second reads G as compiler audit data; the third reads B as the boundary that permits a public handshake claim to cite both audit surfaces.

The three entries are exact because no other route-classifier coordinate is present in the carrier. If q is read through the MetaCIC-facing entry, the GroundCompiler row may not be used as proof content, since the carrier has no coordinate turning a GroundCompiler report into MetaCIC proof content. If q is read through the GroundCompiler-facing entry, the MetaCIC row may not be used as compiler evidence, since the carrier has no coordinate turning a MetaCIC theorem into compiler evidence. If q is read through the paired handshake entry, the boundary row B records only the public claim that needs both audit surfaces, and it preserves their separate subject matter.

Finally apply the separation theorem, Definition 852.2. It states that public handshake reads expose both audit rows through B, R, H, C, P, N while preserving the non-conversion boundary. Thus H, C, P, N replay the chosen classifier case componentwise, and any route outside the three displayed cases would require a carrier coordinate outside U. □

Chapter 853

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalBudget}^{\uparrow}$

The $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ packet is the finite selector layer consumed by the diagonal-limit construction of chapter 9238. It sits between the mesh-budget rows of chapter 618 and the comparison rows of chapter 411: a consumer does not choose an ambient limit, but reads a finite precision request, a mesh window, a comparison ledger, a seal handoff, displayed hsame transports, Cont routes, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Definition 853.1 (Cauchy diagonal budget carrier)**.** *A* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (\epsilon, M, W, D, K, S, H, C, P, N)$$

with the following displayed rows:

1. ϵ *is the requested finite precision row for the diagonal-limit read;*
2. M *is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *mesh-budget row selecting the diagonal observation mesh;*
3. W *is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row through which the mesh observations are named;*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and absolute-value ledger used by the budget;*
5. K *is the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *comparison packet read on the selected window;*
6. S *is the* $\mathsf{Real}^{\uparrow}$ *seal-handoff row reached by the finite budget route;*
7. H *stores componentwise* hsame *transports for* $\epsilon, M, W, D, K, S, C, P, N$*;*
8. C *stores the* Cont *routes by which consumers read the precision row, mesh row, selected window, dyadic ledger, comparison packet, seal handoff, provenance, and local naming row;*
9. P *is the* Pkg *provenance over the same finite packet;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *row.*

The packet has no coordinate for quotient streams, a host equality of reals, a selected external limit, trichotomy, or an ambient completeness principle.

Theorem 853.2 (Cauchy diagonal budget NameCert obligations)**.** *For the carrier of Definition 853.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, classifier stability under the componentwise* hsame *rows in* H*, mesh-budget exactness through*

M, W, D, comparison exactness through K, seal-handoff exactness through S, C, P, N, and non-escape for quotient streams, host equality, selected external limits, trichotomy, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier namecert obligations`

Proof. Carrier admission is the finite tuple displayed in Definition 853.1. Every consumer-facing row is one coordinate of B.

Classifier stability is componentwise. The precision row, mesh row, selected window, dyadic ledger, comparison packet, seal row, continuation routes, provenance, and local naming row are transported only by the rows stored in H and replayed only by the routes in C.

The mesh and comparison clauses are read from the two sibling surfaces cited above. The mesh-budget row supplies the finite diagonal observations, and the regular-Cauchy comparison row supplies the finite comparison ledger on those observations.

Finally, the seal-handoff and non-escape clauses are the statement that S, C, P, N repack the same finite route. Since the carrier has no coordinate for quotient streams, host equality, selected external limits, trichotomy, or ambient completeness, those objects cannot enter the local certificate. □

Theorem 853.3 (Cauchy diagonal budget selection determinacy)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. The diagonal selector read by B is determined by the finite route*

$$\epsilon \to M \to W \to D \to K \to S$$

with provenance H, C, P, N. Any two accepted reads with the same displayed rows and the same componentwise hsame *transport select the same mesh-budget, comparison, and seal-handoff route.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier selection determinacy`

Proof. Start with the carrier inventory. The only precision-facing coordinate is ϵ, and the only mesh-facing coordinate is M.

The selected window W and dyadic ledger D are read after the mesh row. A consumer route in C can replay those rows, but it cannot introduce another window or another dyadic tolerance ledger because no such coordinate is present.

The comparison packet K is then the only comparison surface available before the seal row S is reached. Componentwise hsame transport in H preserves each displayed coordinate, so two accepted reads with the same rows and transports follow the same finite route. □

Theorem 853.4 (Cauchy diagonal budget RealCompleteness handoff)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *diagonal-limit consumer of an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier must read the finite route*

$$\epsilon, M, W, D, K, S, H, C, P, N$$

before it reaches the $\mathsf{Real}^{\uparrow}$ *seal. This is the budget-selector surface for the diagonal representative of Definition 9238.3: precision, mesh budget, selected window, dyadic ledger, regular-Cauchy comparison, seal handoff,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows are all displayed.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier real completeness handoff`

Proof. Use selection determinacy to identify the route from the precision request to the seal handoff. The route passes through the mesh budget, selected window, dyadic ledger, and comparison packet in that order.

Now read the diagonal-limit consumer from chapter 9238. Its finite observation request is precisely a request for a displayed precision and selected diagonal observation window before the RealUp seal is inspected.

Because B displays the precision row, mesh row, selected window, dyadic ledger, comparison packet, and seal handoff, the RealCompleteness-facing read factors through B. The transport, continuation, package, and local naming rows provide the provenance of the same finite route. □

Theorem 853.5 (Cauchy diagonal budget non-escape)**.** *An accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *packet cannot export a quotient stream, host real equality proof, selected external limit, trichotomy row, or ambient completeness theorem. Any request for such data remains outside the displayed rows* $\epsilon, M, W, D, K, S, H, C, P, N$ *and therefore cannot become a second diagonal selector or a second* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the accepted carrier. Its public rows are exactly the precision request, mesh budget, selected window, dyadic ledger, comparison packet, seal handoff, transports, continuation routes, provenance, and local naming row.

Apply the NameCert obligation theorem to bind those rows as the full local surface. The excluded objects are named only as absent boundary cases and have no carrier coordinate.

Finally use the RealCompleteness handoff theorem. A diagonal-limit consumer already reaches the seal through the displayed finite route, so an undisplayed quotient, host equality, external limit, trichotomy row, or completeness theorem cannot be repacked as another selector or another seal. □

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier nonescape`

Theorem 853.6 (Cauchy diagonal budget classifier stability)**.** *Let* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ *and* $B' = (\epsilon', M', W', D', K', S', H', C', P', N')$ *be accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carriers. If* H *displays componentwise* hsame *transports from the rows of* B *to the rows of* B' *and* C *displays the corresponding* Cont *replay routes, then the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *classifier gives the same answer on* B *and* B'*. The classifier can inspect only the finite* BHist *rows* $\epsilon, M, W, D, K, S, H, C, P, N$*, the transported rows of* B'*, the* Pkg *provenance rows* P, P'*, and the local naming rows* N, N'*.*

Proof. Start from the carrier inventory in Definition 853.1. The classifier-facing data are precisely the displayed precision, mesh, window, dyadic ledger, comparison, seal, transport, continuation, provenance, and naming rows.

Apply the NameCert obligation surface of Definition 853.2. Its stability clause is componentwise, so each coordinate read from B is replayed along the matching hsame row in H and the matching Cont route in C.

The non-escape boundary of Definition 853.5 prevents an additional quotient stream, host equality proof, selected external limit, trichotomy row, or ambient completeness theorem from changing the classifier. Hence the transported finite rows determine the same local classifier result. □

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier classifier stability`

Theorem 853.7 (Cauchy diagonal budget transported selector naturality)**.** *Let*

$$B = (\epsilon, M, W, D, K, S, H, C, P, N) \quad \textit{and} \quad B' = (\epsilon', M', W', D', K', S', H', C', P', N')$$

be accepted CauchyDiagonalBudget$^{\uparrow}$ *carriers. Suppose the displayed rows of* H *give componentwise* hsame *transports from* $\epsilon, M, W, D, K, S, C, P, N$ *to* $\epsilon', M', W', D', K', S', C', P', N'$*, and suppose the displayed rows of* C *replay the matching* Cont *routes for the precision request, mesh row, selected window, dyadic ledger, comparison packet, seal handoff, provenance, and local naming row. Then every selector, comparison, and seal-facing consumer route proved for* B *transports to the same finite route for* B'*. In particular the transported route still factors through*

$$\epsilon', M', W', D', K', S', H', C', P', N'$$

and cannot read quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier transported selector naturality`

Proof. First transport the classifier. By Definition 853.6, the componentwise hsame rows and replay routes preserve the local NameCert$_{\text{CauchyDiagonalBudget}^{\uparrow}}$ answer, so the accepted selector surface of B is carried to the accepted selector surface of B'.

Next transport the selector and comparison rows. Selection determinacy and mesh-comparison exactness, Definition 853.3 and Definition 853.8, identify the only finite route through the precision, mesh, selected window, dyadic ledger, and comparison coordinates. Replaying the displayed transports replaces those coordinates by $\epsilon', M', W', D', K'$ without changing the route shape.

Finally transport the seal-facing boundary. The seal-boundary factorization theorem and Real seal scope, Definition 853.9 and Definition 853.17, put every seal read behind S, H, C, P, N and hence behind S', H', C', P', N' after transport. The non-escape scope theorem Definition 853.24 then rules out quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness as alternate transported coordinates. □

Theorem 853.8 (Cauchy diagonal budget mesh-comparison exactness)**.** *For every accepted* CauchyDiagonalBudget *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$*, the mesh-budget and comparison obligations factor through the finite rows* M, W, D, K*. The mesh side exposes exactly the diagonal observation budget selected by* M *through the window* W *and dyadic ledger* D*, while the comparison side exposes exactly the* RegularCauchyComparison$^{\uparrow}$ *row* K *read on that same finite window. No comparison obligation may use a mesh row outside* M, W, D*, and no mesh obligation may use a comparison row outside* K*.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudget mesh comparison exactness`

Proof. Read B through Definition 853.1. The only mesh-facing coordinates are M, W, D, and the only comparison-facing coordinate is K.

The NameCert obligations of Definition 853.2 separate mesh-budget exactness from comparison exactness while storing both in the same finite carrier. Thus the mesh route names the observation budget before the comparison packet is consumed, and the comparison route can only read the selected window already displayed by M, W, D.

Use Definition 853.3 to identify the common finite route from ϵ through M, W, D, K to S. The non-escape theorem then rules out any hidden mesh or comparison surface beyond those coordinates. □

Theorem 853.9 (Cauchy diagonal budget seal-boundary factorization)**.** *Every accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ *factors its* $\mathsf{Real}^{\uparrow}$ *seal handoff through the finite boundary rows* S, H, C, P, N*. A consumer that reaches the seal must first consume the displayed precision, mesh, window, dyadic ledger, and comparison rows, and its final handoff is witnessed only by the* hsame *transports in* H*, the* Cont *routes in* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier seal boundary factorization`

Proof. Apply Definition 853.4. Any diagonal-limit consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only after reading the finite route $\epsilon, M, W, D, K, S, H, C, P, N$.

The carrier and NameCert obligation rows identify S as the seal handoff and H, C, P, N as its transport, route, provenance, and naming boundary. There is no second seal coordinate in the carrier.

Finally invoke Definition 853.5. Since quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness are absent from the finite packet, the seal handoff factors through the displayed boundary rows and no external boundary can be spliced in. □

Theorem 853.10 (Cauchy diagonal budget RegSeqRat window scope)**.** *For every accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$*, each comparison obligation exposed to a regular-Cauchy consumer factors through the displayed rows* M, W, D, K*. The row* W *is the selected stream-naming window of chapter 126, the row* D *is the dyadic tolerance ledger of chapter 291, and the row* K *is the regular-Cauchy comparison packet of chapter 411 read on that same finite window. No comparison witness for the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *surface may bypass* M, W, D, K *or replace them with a quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle.*

Proof. Project the accepted carrier by Definition 853.1. Its comparison-facing coordinates are exactly M, W, D, K: the mesh budget M, the selected naming window W, the dyadic ledger D, and the comparison packet K.

Apply mesh-comparison exactness from Definition 853.8. The mesh side supplies the finite diagonal observation budget through M, W, D, and the comparison side supplies the regular-Cauchy row K on that window.

The cited StreamName, DyadicRatCore, and RegularCauchyComparison chapters are therefore consumed only through their displayed rows. The non-escape boundary of Definition 853.5 excludes any undisplayed quotient, host equality, external limit, trichotomy, or completeness data from discharging the comparison surface. □

Theorem 853.11 (Cauchy diagonal budget shared-tail window exhaustion)**.** *Let* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every shared-tail read that enters* B *before a completion-facing seal is available factors through the displayed mesh and comparison rows* M, W, D, K*. The read therefore consumes the selected* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, and the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row of chapter 411 as sibling rows on the same finite route; no private tail window, private dyadic ledger, or second comparison packet is exposed by the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *surface.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier shared tail window exhaustion`

Proof. Project the accepted carrier. Its tail-facing comparison surface has only the mesh row M, selected window W, dyadic ledger D, and comparison row K before the seal handoff S.

Apply Definition 853.10. It identifies the regular-Cauchy comparison obligation with the same W, D, K rows, read on the mesh route already fixed by M.

Now read the three sibling chapters through those rows. The $\mathsf{StreamName}^{\uparrow}$ chapter supplies the selected naming window, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the finite tolerance ledger, and $\mathsf{RegularCauchyComparison}^{\uparrow}$ supplies the comparison packet on that window; all three are replayed by the componentwise hsame rows in H and the displayed Cont routes in C.

Finally use Definition 853.5. Since the carrier has no coordinate for a private tail window, private dyadic ledger, second comparison packet, quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle, every shared-tail read is exhausted by M, W, D, K before any seal-facing row is reached. □

Theorem 853.12 (Cauchy diagonal budget route determinacy)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ carrier. Every $\mathsf{Real}^{\uparrow}$- or completion-facing read of B is forced through the single displayed route*

$$\epsilon \to M \to W \to D \to K \to S$$

and is stabilized by the componentwise hsame transports H, continuation routes C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ row N. Two accepted consumers with the same displayed route rows and componentwise hsame transports cannot diverge by using a private tail window, second comparison packet, quotient stream, selected external limit, trichotomy row, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier route determinacy`

Proof. Start with selection determinacy. By Definition 853.3, an accepted packet reaches the seal only through the displayed sequence from the precision request to the selected seal row.

Next attach the comparison surface. Mesh-comparison exactness and shared-tail window exhaustion, Definition 853.8 and Definition 853.11, identify M, W, D, K as the single shared comparison route consumed before any Real-facing or completion-facing handoff.

Now read the seal boundary. The seal-boundary factorization theorem, Definition 853.9, makes H, C, P, N the only transport, route, provenance, and naming rows that stabilize the handoff at S.

Finally apply non-escape. Since the accepted carrier contains no coordinate for a private tail window, second comparison packet, quotient stream, selected external limit, trichotomy row, host real equality, or ambient completeness principle, two consumers with the same displayed route rows and hsame transports have no remaining row on which to diverge. □

Theorem 853.13 (Cauchy diagonal budget seal-consumer coverage)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ carrier. Every $\mathsf{Real}^{\uparrow}$- or completion-facing consumer that reaches the seal row S through B has already consumed the full displayed packet*

$$\epsilon, M, W, D, K, H, C, P, N.$$

Thus the seal read is never a direct ambient Real claim: it is a read of the precision request, mesh budget, selected window, dyadic ledger, comparison row, componentwise transports, continuation routes, package provenance, and local naming row before the boundary coordinate S is inspected.

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier seal consumer coverage`

Proof. Begin with a consumer that reaches S. By Definition 853.9, the seal handoff has boundary rows S, H, C, P, N and cannot be read apart from its displayed transport, route, provenance, and naming witnesses.

Next attach the selector prefix. Selection determinacy, Definition 853.3, places the precision request, mesh row, selected window, dyadic ledger, and comparison packet in the route $\epsilon \to M \to W \to D \to K \to S$.

The RealCompleteness handoff theorem Definition 853.4 then identifies the same rows as the diagonal-limit budget surface consumed before the $\mathsf{Real}^{\uparrow}$ seal is reached. Hence every Real- or completion-facing consumer has already consumed $\epsilon, M, W, D, K, H, C, P, N$ when it reaches S.

The non-escape theorem closes the boundary. Quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness have no carrier coordinate, so none can supply a shortcut to S without the displayed packet. □

Theorem 853.14 (Cauchy diagonal budget route factorization)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every shared-tail,* $\mathsf{Real}^{\uparrow}$*-facing, or completion-facing read through B factors through the single finite route*

$$\epsilon \to M \to W \to D \to K \to S.$$

The rows H, C, P, N authenticate that route by componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *surface. No consumer of the diagonal budget can reach a shared-tail window, comparison packet, or Real seal through any route not displayed by ϵ, M, W, D, K, S.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudget route factorization`

Proof. First bind the selected route. Apply Definition 853.3: an accepted diagonal-budget read is determined by the route from ϵ through M, W, D, K to S.

Next exhaust the shared-tail side. By Definition 853.11, every shared-tail read that enters the packet factors through M, W, D, K and the StreamName, DyadicRatCore, and RegularCauchy-Comparison sibling rows on that same route.

Now cover the seal consumers. By Definition 853.13, every Real- or completion-facing consumer that reaches S has already consumed $\epsilon, M, W, D, K, H, C, P, N$. The seal-boundary factorization theorem, Definition 853.9, identifies S, H, C, P, N as the only final handoff boundary.

The displayed route is therefore both the shared-tail route and the seal route. Since the carrier exposes no second window, second comparison row, second seal, quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle, every diagonal-budget consumer factors through $\epsilon \to M \to W \to D \to K \to S$ with provenance H, C, P, N. □

Theorem 853.15 (Cauchy diagonal budget finite-approximation handoff)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every finite approximation consumer of the diagonal route factors through the displayed rows*

$$\epsilon \to M \to W \to D \to K \to S$$

and then through the $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ *packet before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff is read. The rows* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are consumed only as named finite surfaces on that route.*

The handoff exports no quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle.

Lean4 ✓ BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier finite approximation handoff

Proof. Begin with the diagonal selector. Selection determinacy, Definition 853.3, fixes the route from the precision request through the mesh, selected window, dyadic ledger, comparison row, and seal handoff.

Next use mesh-comparison exactness and route factorization. By Definition 853.8, the mesh and comparison reads are exactly the displayed M, W, D, K rows. By Definition 853.14, every shared-tail, $\mathsf{Real}^{\uparrow}$-facing, or completion-facing read through B uses the same finite route $\epsilon \to M \to W \to D \to K \to S$.

Now pass to the finite approximation surface. The observation and Real-handoff rows of Definition 601.7 and Definition 601.11 read $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ only through the displayed finite approximation packet. The regular comparison row K supplies the named $\mathsf{RegularCauchyComparison}^{\uparrow}$ surface already locked to W and D.

Finally apply non-escape. The diagonal budget non-escape theorem and the finite approximation seal-coverage theorem have no coordinate for quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness. Therefore a finite approximation consumer can receive only the displayed diagonal route followed by the $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ packet. □

Theorem 853.16 (Cauchy diagonal budget selector RegSeqRat seal exhaustion)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C,$ be an accepted $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ carrier. Every completion-facing consumer that enters B through the selector route must first consume the mesh row M, the selected $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, the $\mathsf{RegularCauchyComparison}^{\uparrow}$ readback row K, and then the finite approximation handoff of Definition 853.15. Only after that finite route is exhausted may the consumer read the $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal row S, together with the transport, continuation, provenance, and naming rows H, C, P, N.*

The consumer receives no quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle from the diagonal budget.

Proof. First identify the selector route. Apply Definition 853.14: every shared-tail, $\mathsf{Real}^{\uparrow}$-facing, or completion-facing read through B factors through the finite route from ϵ to S with the authenticating rows H, C, P, N.

Next isolate the RegSeqRat readback surface. By Definition 853.10, the only regular-Cauchy data visible on that route are the selected $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, and the $\mathsf{RegularCauchyComparison}^{\uparrow}$ row K. Thus a selector-route consumer cannot read a separate regular sequence or comparison packet before it consumes M, W, D, K.

Now pass through the finite approximation boundary. The handoff theorem Definition 853.15 places the $\mathsf{FiniteCauchyApproximation}^{\uparrow}$ packet after the displayed diagonal route and before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff is read.

Finally close the seal boundary. The non-escape theorem Definition 853.5 rules out quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness as carrier coordinates. Hence the only completion-facing seal available after the selector route is the displayed S row, read with H, C, P, N as its transport, continuation, provenance, and local naming witnesses. □

Theorem 853.17 (Cauchy diagonal budget Real seal scope)**.** *For every accepted $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ carrier $B = (\epsilon, M, W, D, K, S, H, C, P, N)$, the $\mathsf{Real}^{\uparrow}$-facing handoff is scoped exactly by the boundary rows S, H, C, P, N after the finite route ϵ, M, W, D, K has been consumed. The handoff targets the Real naming surface of chapter 38; its seal boundary is the factorization of Definition 853.9,*

with S as the seal row, H as componentwise hsame *transport, C as the displayed* Cont *routes, P as* Pkg *provenance, and N as the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier real seal scope`

Proof. Start with the finite route supplied by Definition 853.4. A diagonal-limit consumer must read ϵ, M, W, D, K before it reaches the seal coordinate S.

Now apply the seal-boundary factorization theorem. It identifies the only handoff boundary as S, H, C, P, N, so the Real-facing surface is not a free ambient completion claim but the displayed seal row together with its transport, route, provenance, and naming witnesses.

The Real chapter is reached at that boundary. Since the carrier has no second seal coordinate and no coordinate for quotient streams, host equality, selected external limits, trichotomy, or ambient completeness, the $\mathsf{Real}^{\uparrow}$ handoff is scoped exactly by the stated rows. □

Theorem 853.18 (Cauchy diagonal budget seal consumption)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. A consumer may read the* $\mathsf{Real}^{\uparrow}$ *seal row S through B only after it has consumed the displayed precision row ϵ, the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row M, the selected* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *packet K, the componentwise* hsame *row H, the* Cont *route row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *row N. Thus every valid seal read is a read of the finite packet*

$$\epsilon, M, W, D, K, H, C, P, N, S.$$

Such a read reconstructs no quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness theorem.

Proof. Project the accepted carrier by Definition 853.1. Its seal-facing coordinates are exactly the precision request, mesh row, selected window, dyadic ledger, comparison packet, seal row, componentwise transports, continuation routes, provenance, and local naming row.

Next use selection determinacy and RealCompleteness handoff. Definition 853.3 fixes the route from ϵ through M, W, D, K to S, while Definition 853.4 says that a diagonal-limit consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only through that displayed budget surface.

Now attach the seal boundary. The transported selector naturality theorem Definition 853.7 preserves the same route under the componentwise hsame rows and displayed Cont replays, and Definition 853.17 identifies S, H, C, P, N as the exact Real-facing boundary.

Finally apply the non-escape and scope rows, Definition 853.5 and Definition 853.24. Since quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness have no carrier coordinate, reading S consumes only the displayed finite packet and reconstructs none of those external objects. □

Theorem 853.19 (Cauchy diagonal budget RegSeqRat-Real route lock)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier must pass through the finite route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The row W is read from $\mathsf{StreamName}^{\uparrow}$*, the row D from* $\mathsf{DyadicRatCore}^{\uparrow}$*, the row K from* $\mathsf{RegularCauchyComparison}^{\uparrow}$*, and the seal row S from* $\mathsf{Real}^{\uparrow}$*. No seal-facing or diagonal-limit use may occur before those rows have been consumed with their componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier regseq real route lock`

Proof. First apply Definition 853.10. It places every regular-Cauchy comparison read behind the selected StreamName window W, the dyadic ledger D, and the comparison row K.

Next apply Definition 853.17. The Real-facing handoff is available only after the route ϵ, M, W, D, K has been consumed and only through the seal boundary S, H, C, P, N.

It remains to exclude alternate seal routes. Use Definition 853.9 to identify the boundary rows and Definition 853.24 to rule out quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness. Hence every completion-facing consumer reaches the diagonal-limit or Real seal only through the displayed RegSeqRat-to-Real route. □

Theorem 853.20 (Cauchy diagonal budget ObservationBudgetLimiter consumer boundary)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Any* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet consumed by the diagonal budget route of B must factor its capped request, budget-window readback, dyadic ledger, regular readback, and seal rows through the displayed finite route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

Thus the limiter cap surface of chapter 903 is consumed only as a finite boundary for the same selected mesh window, dyadic tolerance ledger, comparison row, and $\mathsf{Real}^{\uparrow}$ *seal handoff already displayed by B.*

Proof. First identify the selector route. Apply Definition 853.3 to bind every accepted diagonal read to the rows $\epsilon, M, W, D, K, S, H, C, P, N$.

Next bind the window and comparison side. Apply Definition 853.8 and Definition 853.10. These results place the mesh row, selected window, dyadic ledger, and regular-Cauchy comparison in the same finite route before any seal-facing read is available.

Finally apply Definition 853.9 and Definition 853.17. The $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ consumer may contribute only its capped request surface from chapter 903; once that surface is consumed by B, its downstream readback rows must pass through the displayed route and cannot add an uncapped window, a second comparison route, or an independent seal. □

Theorem 853.21 (Cauchy diagonal budget UniformCauchyCriterion budget route)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. A* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer of the diagonal budget can read the budget only through the shared finite window W, the dyadic tolerance ledger D, the comparison row K, and the seal boundary S, H, C, P, N already displayed by B. In particular, the uniform criterion surface of chapter 435 does not supply a second threshold route, quotient stream family, selected family limit, host real equality, or ambient completeness row for the diagonal budget.*

Proof. Start from the common finite route of Definition 853.3. The diagonal budget has one selected window row, one dyadic ledger, one comparison packet, and one seal handoff.

Use Definition 853.8 to attach the comparison obligation to M, W, D, K. Then use Definition 853.10 to identify W, D, K as the regular-Cauchy window surface seen by downstream criterion consumers.

It remains to bind the seal. Apply Definition 853.9 and Definition 853.17. The $\mathsf{UniformCauchyCriterion}^{\uparrow}$ chapter may be read as a finite shared-window criterion through chapter 435, but the diagonal budget exposes only the rows already listed in B, so the criterion cannot replace the budget route with any external analytic object. □

Theorem 853.22 (Cauchy diagonal budget limit-seal eligibility)**.** *For every accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$*, downstream* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal eligibility is equivalent, at the paper-proof boundary, to prior consumption of the displayed precision row, mesh row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row,* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *cap surface when present, and the seal boundary* S, H, C, P, N*. The consumer chapters chapter 386 and chapter 38 may therefore read the diagonal budget only after that finite route is visible.*

Proof. For the forward direction, begin with a downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal read. Apply Definition 853.17: the read reaches the Real-facing boundary only after ϵ, M, W, D, K have been consumed and only through S, H, C, P, N.

Now refine the consumed route. Apply Definition 853.8 and Definition 853.10 to identify the mesh, stream-window, dyadic, and comparison rows. When an $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet is part of the route, Definition 853.20 places its cap surface before the same readback rows.

For the reverse direction, suppose those displayed rows have been consumed. Apply Definition 853.3 to recover the unique finite selector route, then apply Definition 853.9 to reach the boundary S, H, C, P, N.

Finally use Definition 853.17. The seal route is exactly the surface consumed by the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ chapters, so prior consumption of the listed rows supplies the paper-level eligibility condition, while no quotient stream, host equality, selected external limit, trichotomy row, or ambient completeness theorem is exposed. □

Theorem 853.23 (Cauchy diagonal budget window-meet lock)**.** *For every accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$*, any completion-facing consumer that simultaneously reads a* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row, and either a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary must factor through the same finite route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The consumer may meet those rows as a single finite-window budget, but it cannot splice in a second window, tolerance ledger, comparison row, quotient stream, host real equality, selected external limit, trichotomy row, or ambient completeness principle.

Proof. First bind the route coordinates. Selection determinacy, Definition 853.3, gives the unique finite route from the precision request through M, W, D, K to the seal handoff S, with H, C, P, N carrying the visible transport, continuation, provenance, and local naming rows.

Next identify the finite window that the consumer is allowed to inspect. Apply Definition 853.10: the selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{RegularCauchyComparison}^{\uparrow}$ row are exactly the W, D, K surface attached to the displayed mesh route.

It remains to attach the completion-facing boundary. By Definition 853.22, a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ read is eligible only after the same route has supplied $\epsilon, M, W, D, K, S, H, C, P, N$.

Finally close the alternate surfaces. The non-escape scope theorem Definition 853.24 rules out quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness, and the single-window route already fixes W, D, K. Thus no consumer can splice a second window, tolerance ledger, or comparison row into the meet. □

Theorem 853.24 (Cauchy diagonal budget excluded surface scope)**.** *For every accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$*, quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness are not coordinates of the carrier and cannot discharge the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *surface. Any accepted certificate must be witnessed by the displayed finite rows* $\epsilon, M, W, D, K, S, H, C, P, N$ *and by no larger host-real surface.*

Proof. Use the carrier inventory of Definition 853.1. The displayed coordinates are the precision request, mesh budget, selected window, dyadic ledger, comparison packet, seal handoff, transports, continuation routes, provenance, and local naming row.

Apply Definition 853.5. It states that quotient streams, host equality of reals, selected external limits, trichotomy rows, and ambient completeness theorems are outside those displayed rows and cannot become another selector or another seal.

The NameCert obligations of Definition 853.2 therefore close only over the finite carrier surface. Because each excluded item lacks a carrier coordinate, none can discharge carrier admission, classifier stability, mesh-budget exactness, comparison exactness, seal-handoff exactness, or non-escape for $\mathsf{CauchyDiagonalBudget}^{\uparrow}$. □

Theorem 853.25 (Cauchy diagonal budget ObservationBudgetLimiter consumption)**.** *Let* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *consumer of the diagonal budget receives its capped observation service only through the displayed mesh, selected-window, dyadic, comparison, and seal rows*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

Thus the limiter may consume B *as the route that supplies the mesh window* M, W*, the dyadic ledger* D*, the comparison row* K*, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary* S, H, C, P, N*, but it cannot replace those rows by an uncapped observation source, a second regular readback, or an independent seal.*

Proof. First bind the local budget selector. The NameCert obligations of Definition 853.2 expose the finite carrier rows, while Definition 853.3 fixes their order from ϵ through M, W, D, K to S.

Next identify the rows consumed by the limiter. Apply Definition 853.8 and Definition 853.10. The mesh window, dyadic ledger, and comparison packet are exactly the rows available to an $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ readback.

Finally close the seal boundary. The factorization and Real seal scope theorems, Definition 853.9 and Definition 853.17, place every seal-facing read behind S, H, C, P, N. The consumer-boundary theorem Definition 853.20 then states that the limiter's capped surface is consumed only along that same route, and Definition 853.24 rules out any undisplayed observation source or seal. □

Theorem 853.26 (Cauchy diagonal budget tail-coherence handoff)**.** *Let* $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. A* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *consumer can use the diagonal budget only after the finite budget-selector route*

$$\epsilon \to M \to W \to D \to K \to S$$

has displayed the same mesh window, dyadic ledger, comparison row, transport, continuation, package, and local naming rows that the tail-threshold and seal routes consume. Before a tail threshold or completion-facing seal row is read, the handoff must therefore pass through $M, W, D, K, S, H, C, P, N$ *and not through a host tail choice, quotient stream, selected external limit, or ambient completeness principle.*

Proof. Start with the selector route from Definition 853.3. It gives the only route from the precision row to the seal handoff, with H, C, P, N carrying the visible transport and provenance.

Now attach the tail-facing service rows. By Definition 853.10, every regular-Cauchy comparison obligation seen by a downstream consumer factors through M, W, D, K. By Definition 853.25, the observation service consumed on that route is already capped by the same displayed rows.

It remains to place the seal and threshold read in order. Apply Definition 853.9 and Definition 853.17; they force the final seal-facing boundary through S, H, C, P, N after the mesh, window, dyadic, and comparison rows. The non-escape scope theorem then prevents a $\mathsf{TailBudgetCoherence}^{\uparrow}$ consumer from inserting a hidden tail choice, quotient stream, selected external limit, or ambient completeness row before the accepted handoff. □

Theorem 853.27 (Cauchy diagonal budget diagonal-limit consumer scope)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *consumer that uses B before reaching the* $\mathsf{Real}^{\uparrow}$ *seal must factor through the displayed CauchyDiagonalBudget rows*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The compatibility read may consume those rows as the finite diagonal-limit budget source, but it cannot reach the Real seal by bypassing the mesh-budget row, selected $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row, seal boundary, or local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$ *provenance.*

Proof. First use the RealCompleteness handoff theorem Definition 853.4. It already states that every diagonal-limit read reaches the $\mathsf{Real}^{\uparrow}$ seal only after consuming the displayed precision, mesh, window, dyadic, comparison, transport, route, provenance, and naming rows.

Next refine the compatibility side. Mesh-comparison exactness and RegSeqRat window scope, Definition 853.8 and Definition 853.10, identify M, W, D, K as the exact finite rows that a $\mathsf{DiagonalLimitCompatibility}$ route can read before any seal is available.

Finally apply Definition 853.9, Definition 853.17, and Definition 853.24. These results make S, H, C, P, N the only Real-facing boundary and exclude quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness. Hence the compatibility consumer must factor through the displayed CauchyDiagonalBudget packet before reaching the seal. □

Theorem 853.28 (Cauchy diagonal budget consumer triad route intersection)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P,$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. The* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, *and* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *consumers all intersect exactly on the finite route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

Each consumer may read its own service surface through that route, but none may replace the route by an uncapped observation source, a second threshold route, a quotient stream family, a selected external limit, host real equality, or an ambient completeness row.

Proof. First compose the three local consumer routes. The limiter route is fixed by Definition 853.25, the uniform-criterion route is fixed by Definition 853.21, and the tail-coherence handoff is fixed by Definition 853.26. Their common displayed coordinates are therefore $\epsilon, M, W, D, K, S, H, C, P, N$.

Next close the observation and threshold alternatives. The carrier and NameCert rows admit only the displayed precision, mesh, window, dyadic, comparison, seal, transport, continuation, package, and naming coordinates, so a hidden observation source or second threshold route has no coordinate through which it can be consumed.

Finally close the seal-facing alternatives. Apply Definition 853.24 and Definition 853.17. These results rule out quotient stream families, selected external limits, host real equality, and ambient completeness rows, and they force every Real-facing read through the same S, H, C, P, N boundary already shared by the three consumers. □

Theorem 853.29 (Cauchy diagonal budget seal synchronizer pullback)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Any* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer that reaches the seal through the diagonal budget pulls back to the same displayed route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The three consumers therefore agree on the seal boundary rows S, H, C, P, N, and none can substitute quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness for the displayed diagonal-budget seal route.

Proof. Begin with the finite selector itself. Selection determinacy, Definition 853.3, fixes the route from the precision request through the mesh, selected window, dyadic ledger, and comparison row to the seal handoff S.

Next attach the Real-facing boundary. The RealCompleteness handoff of Definition 853.4 and the Real seal scope theorem Definition 853.17 place every completion read behind the displayed seal rows S, H, C, P, N. The diagonal-limit consumer scope theorem, Definition 853.27, gives the same displayed route for every $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ consumer before it reaches the $\mathsf{Real}^{\uparrow}$ seal.

It remains to align the limit-seal consumer. Apply Definition 853.22: a $\mathsf{CauchyLimitSeal}^{\uparrow}$ read is eligible only after the same finite budget packet has supplied $\epsilon, M, W, D, K, S, H, C, P, N$. The non-escape scope theorem Definition 853.24 rules out quotient streams, host real equality, selected external limits, trichotomy rows, and ambient completeness as alternate seal coordinates.

Thus each of the three consumers inspects the unique displayed seal row of the accepted carrier. Carrier inspection then gives a common pullback to $\epsilon, M, W, D, K, S, H, C, P, N$, with common boundary S, H, C, P, N and no undisplayed substitute route. □

Theorem 853.30 (Cauchy diagonal budget tail-fusion pullback)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier, and let $F = (Q, X, W_F, D_F, T_F, M_F, R_F, L_F, H_F, C_F, P_F, N_F)$ be an accepted* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *packet. Suppose the selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, comparison and readback route, and seal rows of F agree componentwise with W, D, K, S after the displayed* hsame *transports and* Cont *routes of both packets are replayed. Then every* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal through the diagonal budget pulls back to*

$$\epsilon, M, W, D, K, S, H, C, P, N$$

before it can read L_F. The consumer may use $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *and* $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ *through the fusion packet, but it cannot substitute quotient streams, host real equality, selected external limits, or ambient completeness for the displayed diagonal-budget selector.*

Proof. Start with the accepted diagonal-budget carrier. Selection determinacy and mesh-comparison exactness, Definition 853.3 and Definition 853.8, force the precision request, mesh window, selected stream window, dyadic ledger, and comparison row to pass through ϵ, M, W, D, K before the seal row S is available.

Now replay the tail-fusion side. The tail-meet handoff, shared-threshold exhaustion, and Real limit handoff of Definition 658.4, Definition 658.6, and Definition 658.7 route every cofinal tail read through $Q, X, W_F, D_F, T_F, M_F, R_F, H_F, C_F, P_F, N_F$ before L_F is consumed.

The assumed componentwise agreement identifies the fusion window, ledger, readback, and seal-facing rows with the diagonal-budget rows after the displayed hsame and Cont data are replayed. Hence the fusion route cannot reach L_F without first passing through the same W, D, K, S selected by B.

Finally apply the limit-transport link, non-escape boundary, and seal-scope theorems. Definition 658.8 accounts for the permitted $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$ sibling rows, while Definition 853.24 and Definition 853.17 rule out quotient streams, host real equality, selected external limits, and ambient completeness. The only remaining route to the Real seal is therefore the displayed pullback to $\epsilon, M, W, D, K, S, H, C, P, N$. □

Theorem 853.31 (Cauchy diagonal budget shared-tail modulus seal factorization)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier, and let*

$$T = (R_0, R_1, M_0, M_1, \tau, D_T, W_0, W_1, Q_0, Q_1, E, H_T, C_T, P_T, N_T)$$

be an accepted $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet. Suppose the paired request rows and paired readback rows of T agree componentwise, after the displayed* hsame *transports and* Cont *routes of the two packets are replayed, with the diagonal budget's selected* $\mathsf{StreamName}^{\uparrow}$ *window W, dyadic ledger D, comparison row K, and Real seal handoff S. Then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer that reaches the seal through B factors first through the shared threshold row τ and dyadic ledger row D_T of T, and then through the diagonal-budget route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The factorization has no coordinate for private per-source thresholds, quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness.

Proof. First identify the diagonal seal route. Apply Definition 853.29: every $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, or $\mathsf{CauchyLimitSeal}^{\uparrow}$ read that reaches the seal through B pulls back to $\epsilon, M, W, D, K, S, H, C, P, N$. Apply Definition 853.30 to keep any tail-facing route attached to the same selected window, dyadic ledger, comparison row, and seal handoff.

Now identify the shared-tail side. The Real-completion obligation closure of Definition 983.15 places every completion-facing seal read of T behind its public package, and Definition 983.9 forces that package to use the single threshold row τ with dyadic ledger D_T rather than private per-source thresholds.

Finally compose the two public routes. The assumed componentwise agreement transports the paired request and readback rows of T onto the diagonal budget's selected W, D, K, S boundary after the displayed hsame and Cont rows are replayed. The Pkg and local NameCert rows on both packets then authenticate one route through τ, D_T followed by $\epsilon, M, W, D, K, S, H, C, P, N$; since neither carrier contains quotient streams, host real equality, selected external limits, trichotomy, or ambient completeness, there is no alternate seal-facing factor. □

Theorem 853.32 (Cauchy diagonal budget terminal common route)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Any completion-facing consumer that enters the packet through* $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *a shared-tail modulus packet, or a* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *row must factor through the same displayed route*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

The route consists of the finite precision request, mesh row, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row, seal row, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row. It exports no quotient streams, host real equality, selected limits, countable-choice schedule, or ambient completeness principle.*

Proof. First bind the local budget route. Apply Definition 853.21 to any $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer and Definition 853.22 to any $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. Both results force the read to pass through the finite rows $\epsilon, M, W, D, K, S, H, C, P, N$ before it reaches a completion-facing seal.

Next attach the shared terminal consumers. The consumer-triad route intersection theorem, Definition 853.28, identifies the common route for $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ reads. The shared-tail modulus seal factorization theorem, Definition 853.31, then sends every shared-tail modulus entry through its threshold row and back to the same diagonal-budget route.

Finally read the structural coordinates. The rows H, C, P, N are precisely the componentwise transport, displayed continuation, provenance, and local naming data for the route, so the accepted carrier provides no coordinate from which an alternate terminal path can be formed. The non-escape and seal-scope boundaries already rule out quotient streams, host real equality, selected limits, countable-choice schedules, and ambient completeness. □

Theorem 853.33 (Cauchy diagonal budget root window source)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every accepted budget packet first exposes the finite root window*

$$\epsilon, M, W, D$$

where ϵ is the precision request, M is the mesh-budget row, W is the selected $\mathsf{StreamName}^{\uparrow}$ *window, and D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The root window is the source from which later regular-Cauchy comparison and Real-seal reads enter the packet; it has no route around the displayed StreamName, DyadicRatCore, or mesh-budget rows.*

Proof. Start with the carrier of Definition 853.1. Its first displayed rows are exactly the precision request, the $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ mesh row, the selected $\mathsf{StreamName}^{\uparrow}$ window, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

Selection determinacy and RegSeqRat window scope then fix the route. By Definition 853.3, the packet reaches the comparison and seal rows only after ϵ, M, W, D have been selected. By Definition 853.10, any regular-Cauchy comparison read consumes the same selected window and dyadic ledger rather than a private stream or tolerance source.

Finally apply mesh-comparison exactness and non-escape. Definition 853.8 identifies the mesh, window, ledger, and comparison rows as the displayed finite source for later reads, while Definition 853.5 rules out quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness as alternate root-window coordinates. □

Theorem 853.34 (Cauchy diagonal budget tail ledger route)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. The shared-tail comparison ledger visible to any downstream consumer is exactly*

$$M, W, D, K, H, C, P, N.$$

It cannot bypass the displayed dyadic row D, the regular-Cauchy comparison row K, the transport row H, the replay row C, the package row P, or the local naming row N.

Proof. First bind the tail read to the root window. Definition 853.33 exposes M, W, D as part of the finite source route before a comparison row can be consumed.

Next use mesh-comparison exactness and shared-tail window exhaustion. Definition 853.8 makes K the comparison row attached to that dyadic window, and Definition 853.11 says every shared-tail read through the packet is exhausted by the same M, W, D, K surface.

Finally account for the certificate rows. The componentwise transport, displayed continuation, provenance, and local NameCert rows H, C, P, N are the only exported replay data for the ledger route. Since the carrier has no coordinate for private tail windows, private dyadic ledgers, second comparison packets, or host equality of reals, no shared-tail consumer can bypass the displayed ledger. □

Theorem 853.35 (Cauchy diagonal budget Real seal handoff totality)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal read through B factors through the root-window source of Definition 853.33 and the tail-ledger route of Definition 853.34 before it reaches the seal row S. Thus the finite route from the root budget window to the Real-facing seal is exhausted by*

$$\epsilon, M, W, D, K, H, C, P, N, S.$$

Proof. Begin with seal-consumer coverage. By Definition 853.13, any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer that reaches the seal through B has already consumed the precision request, mesh row, selected window, dyadic ledger, comparison row, transport, replay, provenance, and local naming rows.

Now decompose that coverage. The root-window theorem supplies the first source segment ϵ, M, W, D, and the tail-ledger theorem supplies the comparison and certificate segment M, W, D, K, H, C, P, N. The overlap is the same displayed mesh-window-ledger surface, so the two segments compose without adding a private tail source or detached comparison packet.

Finally close the non-escape boundary. The Real seal scope and non-escape scope theorems, Definition 853.17 and Definition 853.24, force the final handoff through S and exclude quotient streams, host real equality, selected external limits, trichotomy, countable-choice schedules, and ambient completeness. Hence every Real-facing or completion-facing seal read factors through the displayed finite route before reaching S. □

Theorem 853.36 (Cauchy diagonal budget terminal route coincidence)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every consumer route entering through* $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, *or an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet exports the same finite route*

$$\epsilon, M, W, D, K, H, C, P, N, S.$$

In particular, the root-window source, tail-ledger route, terminal common route, and Real-seal handoff determine one route after the displayed hsame *transports and* Cont *replays are applied. The route has no coordinate for quotient streams, host real equality, selected external limits, countable-choice schedules, trichotomy, or ambient completeness.*

Proof. First put every terminal consumer on the common diagonal route. By Definition 853.32, any $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, shared-tail modulus, or $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ entry factors through the displayed carrier rows of B rather than through a private completion witness.

Next identify the two exposed segments of that route. The root-window source theorem, Definition 853.33, supplies ϵ, M, W, D as the unique source window. The tail-ledger route theorem, Definition 853.34, supplies M, W, D, K, H, C, P, N as the comparison and certificate ledger attached to that same source.

The shared-tail case uses the same attachment. Applying Definition 853.31 sends any $\mathsf{CauchySharedTailModulus}^{\uparrow}$ entry first through its shared threshold and dyadic ledger rows, and then back to the diagonal-budget route selected by B. Thus the shared-tail packet contributes no alternate terminal route.

Finally apply the Real-seal handoff theorem. Definition 853.35 appends the final seal row S to the root and tail ledger segments, yielding $\epsilon, M, W, D, K, H, C, P, N, S$. The non-escape and seal-scope boundaries, Definition 853.24 and Definition 853.17, rule out quotient streams, host real equality, selected external limits, countable-choice schedules, trichotomy, and ambient completeness as competing exits. □

Theorem 853.37 (Cauchy diagonal budget selector route totality)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every admitted selector read for B factors through the finite Real-completion route*

$$\epsilon \longmapsto M \longmapsto W \longmapsto D \longmapsto K \longmapsto S$$

together with the componentwise transport, continuation replay, provenance, and local naming rows H, C, P, N. Any missing selector route is refused rather than converted into a quotient stream, host $\mathsf{Real}^{\uparrow}$ *equality, trichotomy branch, selected external limit, countable-choice schedule, or ambient completeness coordinate.*

Proof. Begin with selection determinacy. Definition 853.3 fixes the accepted selector read from the displayed precision request and mesh-window data, so an admitted read starts at ϵ, M, W rather than at an external stream or host real.

Next attach the dyadic and comparison rows. Mesh-comparison exactness, Definition 853.8, identifies D and K as the finite ledger and comparison rows selected by the same mesh-window source. Hence the selector route has no independent dyadic, regular-Cauchy, or trichotomy coordinate.

The seal boundary supplies the terminal side of the route. By Definition 853.9, every completion-facing selector read reaching the seal must pass through the displayed comparison and seal rows before it can be consumed.

Finally compose the terminal route theorems. Definition 853.32 places all completion consumers on the shared finite route, and Definition 853.36 identifies that route as $\epsilon, M, W, D, K, H, C, P, N, S$. Therefore every admitted selector read factors through $\epsilon \longmapsto M \longmapsto W \longmapsto D \longmapsto K \longmapsto S$ with H, C, P, N, while every absent route is refused rather than exported as quotient, host-equality, trichotomy, external-limit, choice, or ambient-completeness data. □

Theorem 853.38 (Cauchy diagonal budget prefix request monotonicity)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ and $B' = (\epsilon', M', W', D', K', S', H', C', P', N')$ be accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carriers. Suppose that ϵ' is a finite refinement of ϵ on the same displayed mesh, selected window, dyadic ledger, comparison packet, and seal route, and suppose that the componentwise* hsame *transports and displayed* Cont *rows replay the same prefix from B to B'. Then every* $\mathsf{Real}^{\uparrow}$*- or completion-facing consumer of B' factors through an extension of the route already exposed by B:*

$$\epsilon \longmapsto M \longmapsto W \longmapsto D \longmapsto K \longmapsto S \quad \rightsquigarrow \quad \epsilon' \longmapsto M' \longmapsto W' \longmapsto D' \longmapsto K' \longmapsto S'.$$

The refined packet exports no private window, second comparison packet, quotient stream, host equality, selected external limit, trichotomy row, or ambient completeness row.

Proof. Start with selection determinacy and selector route totality. Theorems 853.3 and 853.37 force both accepted packets to expose their selector reads through the displayed precision, mesh, window, dyadic, comparison, and seal coordinates.

Next compare the finite mesh route. By mesh-comparison exactness, Definition 853.8, the rows M, W, D, K and M', W', D', K' are the whole comparison surfaces attached to their precision requests. The assumed prefix replay through H, C, P, N and H', C', P', N' therefore changes only the finite request depth and not the route shape.

Now factor downstream consumers. The route factorization theorem, Definition 853.14, sends every shared-tail, $\mathsf{Real}^{\uparrow}$-facing, or completion-facing read through the same finite selector route. Hence a consumer of B' first reads the route already exposed by B and then the additional finite refinement coordinates of B'.

Finally close the refusal boundary. The non-escape and non-escape-scope theorems, Definition 853.5 and Definition 853.24, exclude private windows, detached comparison packets, quotient streams, host real equality, selected external limits, trichotomy, and ambient completeness. Thus the refinement is monotone only by finite route extension. □

Theorem 853.39 (Cauchy diagonal budget finite-route induction)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every admissible prefix of the displayed selector route*

$$\epsilon \longmapsto M \longmapsto W \longmapsto D \longmapsto K \longmapsto S$$

is obtained by induction over the six displayed route stages: the base request ϵ, the mesh step M, the selected-window step W, the dyadic-ledger step D, the comparison step K, and the seal step S. At each stage the rows H, C, P, N carry the componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *witness for the prefix already displayed. The induction exports no private window, second comparison packet, quotient stream, host equality, selected external limit, trichotomy branch, countable-choice schedule, or ambient completeness coordinate.*

Proof. The base case is the accepted carrier itself. By Definition 853.1, the packet B exposes the precision request ϵ together with the transport, replay, provenance, and local naming rows H, C, P, N. Selection determinacy, Definition 853.3, makes this request the unique source from which an admissible selector read can begin.

For the mesh and selected-window steps, the same determinacy theorem fixes M and W as the displayed mesh and window selected by ϵ. The prefix request monotonicity theorem, Definition 853.38, says that any admissible finite refinement extends this visible route rather than replacing it by a private window.

For the dyadic-ledger and comparison steps, apply mesh-comparison exactness. Theorem 853.8 identifies D and K as the whole ledger and comparison surface attached to the selected mesh-window prefix, so a second comparison packet or detached tolerance coordinate cannot be introduced in the induction step.

The seal step closes the induction. Seal-boundary factorization, Definition 853.9, forces any completion-facing continuation through the displayed seal row S, and finite-route terminality, Definition 853.41, identifies the resulting terminal route as the same finite packet rows. Hence each admissible prefix is generated by the six displayed stages with H, C, P, N carried along componentwise, while quotient streams, host equality, selected external limits, trichotomy, countable-choice schedules, and ambient completeness remain outside the carrier. □

Theorem 853.40 (Cauchy diagonal budget Real-RegSeq-StreamName factorization)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. A public consumer row whose declared endpoint is one of* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *is admissible for B exactly when its visible source projection factors through*

$$\epsilon \longmapsto M \longmapsto W \longmapsto D \longmapsto K \longmapsto S$$

together with the componentwise transport, displayed continuation, provenance, and local naming rows H, C, P, N. This interface exports no quotient streams, host $\mathsf{Real}^{\uparrow}$ *equality, selected external limits, trichotomy branches, countable-choice schedules, or ambient completeness coordinates.*

Proof. For the only-if direction, first bind the consumer to the selected source. Selection determinacy and selector route totality, Definition 853.3 and Definition 853.37, force every admitted public read to enter through the displayed precision, mesh, selected $\mathsf{StreamName}^{\uparrow}$ window, dyadic ledger, comparison row, and seal route of B.

Next use prefix request monotonicity and terminal route coincidence. Theorems 853.38 and 853.36 identify the visible route seen by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers as the same finite projection after the displayed hsame transports and Cont replays are applied. The non-escape boundary then excludes quotient streams, host real equality, selected external limits, trichotomy, countable-choice schedules, and ambient completeness as alternate admissibility witnesses.

For the if direction, assume the visible source projection factors through the displayed route. The carrier rows of Definition 853.1 supply exactly the precision request, mesh row, selected stream window, dyadic ledger, comparison row, seal handoff, transport, replay, provenance, and local $\mathsf{NameCert}$ data named in that projection.

Mesh-comparison exactness, seal-boundary factorization, and the NameCert obligation surface then authenticate the projection as a public consumer row. Applying Definition 853.8, Definition 853.9, and Definition 853.2 shows that no further field is needed: the displayed finite route is exactly the public interface for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers of B. □

Theorem 853.41 (Cauchy diagonal budget finite-route terminality)**.** *Let $B = (\epsilon, M, W, D, K, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}$ *or* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *consumer of the diagonal budget factors through the same finite route rows*

$$\epsilon, M, W, D, K, S, H, C, P, N.$$

No such consumer may add a private window, a second dyadic tolerance ledger, quotient stream equality, selected external limit, host $\mathsf{Real}^{\uparrow}$ *equality, or ambient completeness row before the seal handoff.*

Proof. Start with the regular-Cauchy and shared-tail sides. The RegSeqRat window scope theorem, Definition 853.10, places every regular-Cauchy read behind the selected window W, dyadic ledger D, and comparison row K. The shared-tail window exhaustion theorem, Definition 853.11, prevents a second tail window or tolerance ledger from replacing that route.

Next bind the seal-facing route. The Real seal scope theorem, Definition 853.17, makes S, H, C, P, N the only boundary through which a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can read the accepted budget. The limit-seal eligibility theorem, Definition 853.22, identifies the same route as the required paper-level handoff for seal consumers.

Now attach the finite consumer chapters. The observation-limiter boundary theorem, Definition 853.20, consumes every limiter read through the capped route already displayed by B. The uniform-criterion route theorem, Definition 853.21, similarly routes criterion consumers through $\epsilon, M, W, D, K, S, H, C, P, N$.

Composing these rows leaves no unaccounted coordinate. The excluded private window, second tolerance ledger, quotient stream equality, selected external limit, host real equality, and ambient completeness rows are not fields of the carrier and are refused by the same seal-facing boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ packet is generated from a finite precision request, a $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ mesh-budget row, selected $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{RegularCauchyComparison}^{\uparrow}$ comparison row, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 853.1, Definition 853.2, Definition 853.3, Definition 853.4, Definition 853.5, Definition 853.10, Definition 853.11, Definition 853.12, Definition 853.13, Definition 853.14, Definition 853.15, Definition 853.16, Definition 853.17, Definition 853.18, Definition 853.20, Definition 853.21, Definition 853.22, Definition 853.23, Definition 853.24, Definition 853.25, Definition 853.26, Definition 853.27, Definition 853.28, Definition 853.29, Definition 853.30, Definition 853.31, Definition 853.32, Definition 853.33, Definition 853.34, Definition 853.35, Definition 853.36, Definition 853.37, Definition 853.38, Definition 853.39, Definition 853.40, and Definition 853.41 define the finite budget-selector carrier, obligation rows, selection determinacy, RealCompleteness handoff, non-escape boundary, RegSeqRat window scope, shared-tail window exhaustion, route determinacy, seal-consumer coverage, route factorization, finite-approximation handoff, selector RegSeqRat seal exhaustion, Real seal scope, seal consumption exactness, consumer-boundary routes, seal eligibility, finite-window meet lock, excluded surface scope, downstream consumption boundaries, common consumer intersection, cross-consumer seal pullback, tail-fusion pullback, shared-tail threshold seal factorization, terminal common route, root-window source, tail-ledger route, Real seal handoff totality, terminal route coincidence, selector-route totality, finite prefix request monotonicity, finite-route induction, the Real-RegSeq-StreamName public consumer interface, and the finite-route terminality boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalBudget}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyDiagonalBudgetUp.CauchyDiagonalBudgetCarrier\_namecert\_obligations`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient streams, host equality of reals, selected external limits, trichotomy, ambient completeness, or a public standard bridge.

Upgrade path: The paper closure axis is complete for the finite budget-selector packet; Lean verification and bridge checking remain independent axes.

Chapter 854

A Concrete Naming Certificate for $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$

The $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ packet realizes the certificate-indexed branch boundary of chapter 9458 as a finite NameCert surface. It sits horizontally beside chapter 667: the sibling packet records a self-reference truth-branch request, while this packet records the boundary that permits local refutation routes and refuses certificate-indexed truth branching.

Definition 854.1 (Certificate truth-branch boundary carrier)**.** *A* $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, A, R, D, F, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the certificate-indexed query row, naming the fixed local certificate and the proposition row being tested;*
2. *A is the assumption row whose local refutation is permitted only inside the displayed packet;*
3. *R is the permitted refutation route from an assumed witness of A to $\bot$;*
4. *D is the optional constructive decision row for A, present exactly when the packet may consume $\neg\neg A$ as a local decidable refutation step;*
5. *F is the forbidden truth-branch row, recording that no branch family over all naming certificates is a carrier coordinate;*
6. *L is the boundary ledger separating admitted refutation reads from refused certificate-indexed truth reads;*
7. *H records componentwise* hsame *transports for query, assumption, refutation, decision, refusal, ledger, provenance, and local naming rows;*
8. *C records the displayed* Cont *replay routes that read Q, replay R, consume D when present, inspect F, L, and return the result to the same local certificate;*
9. *P is the* Pkg *provenance over* BHist, msame, hsame, Cont, *local refutation routes, refusal rows, ledger rows, and the surrounding NameCert source rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateTruthBranchBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a truth predicate over its own certificate stratum, no excluded-middle branch over the carrier of all naming certificates, and no witness selector obtained only from a contradiction.

Lean4 (definition) `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryUp`

Theorem 854.2 (Certificate truth-branch boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CertificateTruthBranchl}}$ *surface for Definition 854.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet* $B = (Q, A, R, D, F, L, H, C, P, N)$*;*
2. *query exactness, saying that the certificate-indexed query row* Q *is read only as a fixed local question and never as a global truth classifier;*
3. *permitted-refutation replay, saying that the route* R *reaches* $\bot$ *from an assumed* A*-witness only through the displayed* hsame *transports and* Cont *routes;*
4. *decision discipline, saying that* $\neg\neg A$ *may be consumed only through the same displayed row* D*;*
5. *truth-branch refusal, saying that* F *and* L *block certificate-wide excluded middle, self-truth classification, and witness extraction by contradiction.*

Lean4 ✓ `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryNameCertObligations`

Proof. Carrier admission is projection from Definition 854.1. The packet exposes exactly the query, assumption, refutation, optional decision, refusal, ledger, transport, continuation, provenance, and local naming rows.

Query exactness is the separation between Q and F, L. The row Q names the local certificate question being inspected, while F and L record that the question is not promoted to a branch family over all naming certificates.

Permitted-refutation replay uses only R, H, C, P, N. Transport first carries the assumption and route along H, then the displayed continuations in C replay the route to $\bot$ inside the same local packet.

Decision discipline is confined to D. When D is absent, the ledger refuses non-decidable double-negation elimination; when D is present, it is consumed for the same proposition row A and does not become a global oracle.

Truth-branch refusal follows by inspecting the carrier inventory. No coordinate supplies certificate-wide excluded middle, a truth predicate for the certificate stratum, or a witness selected only from contradiction, so the local NameCert obligations are exhausted by the stated finite surface. □

Theorem 854.3 (Certificate truth-branch boundary non-escape)**.** *Every accepted* $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ *packet exports only the permitted refutation side of the certificate-indexed boundary. It may export the local query row, the displayed refutation route, the optional decision row for the same proposition, the refusal ledger, componentwise transports, continuation replay, provenance, and local naming; it exports no certificate-indexed truth branch and no branch family for predicates over the carrier of all naming certificates.*

Proof. Let an accepted packet be displayed as $B = (Q, A, R, D, F, L, H, C, P, N)$. The carrier definition lists every row a downstream reader may inspect.

The permitted side is local. A consumer may read Q, assume A, replay R through H and C, and use D only when the packet carries that constructive decision row for the same A.

The refused side is recorded by F and L. These rows name the absent branch family and block replacement of the local query by a truth predicate over naming certificates.

The remaining rows H, C, P, N preserve, replay, package, and name the same finite surface. They do not add a second classifier or a witness selector.

Therefore every public read remains inside the permitted-refutation boundary. The certificate can be used as a finite local refutation packet, but it cannot be used as a certificate-indexed truth branch. □

Lean4 ✓ `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryNonescape`

Theorem 854.4 (Certificate truth-branch boundary public interface)**.** *Every public consumer of an accepted* $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ *packet must factor through the fixed local query row, permitted refutation route, optional decision row, truth-branch refusal row, boundary ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before using the packet. The public interface exports the local refutation boundary only; it exports no certificate-indexed truth predicate, excluded-middle branch over all naming certificates, witness selector obtained from contradiction, or global contradiction-boundary bridge.*

Proof. Project an accepted packet as $B = (Q, A, R, D, F, L, H, C, P, N)$. The carrier lists the query, assumption, permitted refutation, optional decision, refusal, ledger, transport, replay, provenance, and local naming rows as the only public rows.

Use the NameCert obligations. They make Q a fixed local query, R a displayed route from an assumed A-witness to $\bot$, D an optional constructive decision row for the same proposition, and F, L the refusal ledger blocking certificate-wide truth branching.

The non-escape theorem then identifies the downstream surface. A public reader may consume the local refutation route through H, C, P, N, but it cannot replace that route by a truth predicate over certificates or a witness selector obtained only from contradiction.

Thus the public interface is exactly the carrier and NameCert surface already displayed, with the local refutation route exposed before any use of the packet. □

Lean4 ✓ `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryPublicInterface`

Theorem 854.5 (Certificate truth-branch boundary bridge interface)**.** *Every standard contradiction-boundary consumer of an accepted* $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ *packet must factor through the fixed local query row, permitted refutation route, optional decision row, truth-branch refusal row, boundary ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before the boundary is read as a bridge-facing interface. The interface exports only the displayed local refutation boundary and the refusal ledger; it exports no certificate-indexed truth predicate, global excluded-middle branch, witness selector by contradiction, or Lean bridge certificate.*

Lean4 ✓ `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryBridgeInterface`

Proof. Project the accepted packet as $B = (Q, A, R, D, F, L, H, C, P, N)$. The carrier rows expose the fixed query Q, the assumed proposition row A, the permitted route R, the optional decision row D, the refusal row F, the boundary ledger L, and the structural rows H, C, P, N.

The public interface theorem identifies the same row order for public readers. A standard contradiction-boundary consumer therefore receives the local query and permitted refutation route only after the refusal and ledger rows have separated them from certificate-wide truth branching.

The bridge-facing reading is exactly this factorization. The consumer may replay the local refutation route through H and C with provenance P and local name N, but no row supplies a truth predicate over naming certificates, a global branch family, a witness selector from contradiction, or a checked Lean bridge. □

Theorem 854.6 (Certificate truth-branch boundary mature package)**.** *The displayed* CertificateTruthBranchBounda *carrier, branch-boundary row, permitted witness route, replay data, provenance row, and local* NameCert *row assemble the public mature package for certificate-indexed truth-branch boundaries. The package exposes the finite local refutation interface and its boundary ledger, while refusing truth-as-host-equality, certificate-wide truth branching, witness selection by contradiction, and any verification-axis claim beyond the markers explicitly attached in this chapter.*

Proof. Project the carrier as $B = (Q, A, R, D, F, L, H, C, P, N)$. The carrier theorem supplies the finite rows, the NameCert obligations bind their admissible readings, and the non-escape theorem prevents a downstream consumer from adding a certificate-indexed truth branch.

The public-interface and bridge-interface theorems identify the same public route: the query, assumption, permitted refutation, optional decision, refusal ledger, transports, replay, provenance, and local name must all be read before the packet is consumed.

These rows therefore form the mature package boundary on the paper axis. The theorem makes no Lean-status assertion and does not identify truth with host equality; it records only the finite public package already forced by the carrier and boundary theorems. □

Lean4 ✓ `BEDC.Derived.CertificateTruthBranchBoundaryUp.CertificateTruthBranchBoundaryMaturePackage`

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateTruthBranchBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$ packet is generated from a certificate-indexed query row, an assumption row, a permitted refutation route, an optional constructive decision row, a forbidden truth-branch row, a boundary ledger, displayed transports, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateTruthBranchBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 854.1, Definition 854.2, Definition 854.3, Definition 854.4, and Definition 854.5 scope the finite carrier, NameCert obligation surface, non-escape theorem, public interface, bridge-facing interface, and Lean targets to certificate-indexed query rows, permitted local refutation, optional decision rows, truth-branch refusal, boundary ledgers, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateTruthBranchBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove a Lean-side certificate, excluded middle over naming certificates, an internal truth predicate, or witness selection by contradiction.

Upgrade path: Bridge checking requires a Lean-side standard bridge from the local-refutation interface to the standard contradiction-boundary schema.

Chapter 855

A Concrete Naming Certificate for BEDCTheoryIdentity$^{\uparrow}$

The BEDCTheoryIdentity$^{\uparrow}$ packet realizes the framing principle of chapter 9454 as a finite naming certificate. It records the five positive commitments of BEDC.MetaCIC as displayed rows, and it uses the proof-ancestry discipline of chapter 654 as the sibling audit surface for the strict-purity row. The packet is identity-facing: it names what the theory is before any comparison with a neighboring calculus is read.

It joins the BEDC self-theory surface, axiom-query ledger discipline, apophatic boundary rows, and local NameCert provenance into one displayed identity certificate.

Definition 855.1 (BEDC theory identity carrier)**.** *A* BEDCTheoryIdentity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, E, R, P, B, H, C, K, N)$$

with the following displayed rows:

1. *G is the closed-generator row, listing the finite inductive carriers and constructor inventories used by the substrate, including term-level and certificate-level generators;*
2. *E is the literal AST equality row, recording that type equality inside the substrate is the displayed syntactic equality of the represented AST and not a conversion rule imported from another calculus;*
3. *R is the structural-recursor row, recording the generator-local eliminators and induction routes supplied by the closed generators themselves;*
4. *P is the strict-purity row, recording the zero-axiom dependency boundary for accepted BEDC.MetaCIC theorem content and its connection to axiom-query ledgers;*
5. *B is the apophatic-boundary row, recording named sockets, setup fields, and GAP rows as visible external supply rather than hidden substrate theorems;*
6. *H records componentwise* hsame *transports for the five identity rows and for the route, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which consumers read closed generators, literal equality, recursors, strict purity, and boundary markers;*
8. *K is the* Pkg *provenance over* BHist*, closed generators, AST equality, recursor routes, axiom-query ledgers, GAP sockets,* hsame*,* Cont*, and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{BEDCTheoryIdentity}^{\uparrow}}$ *row.*
The carrier has no coordinate for defining BEDC.MetaCIC by subtraction from classical CIC, no host conversion rule, no quotient-equality permission, no hidden axiom dependency, and no ambient semantic principle beyond the displayed rows.

Lean4 (definition) `BEDC.Derived.BedcTheoryIdentityUp.BedcTheoryIdentityUp`
Lean4 ✓ `BEDC.Derived.BedcTheoryIdentityUp.BedcTheoryIdentityTasteGate single carrier alignment`

Theorem 855.2 (BEDC theory identity NameCert obligations)**.** *For every accepted* $\mathsf{BEDCTheoryIdentity}^{\uparrow}$ *carrier*

$$I = (G, E, R, P, B, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{BEDCTheoryIdentity}^{\uparrow}}$ *surface consists exactly of carrier admission for the five identity rows, literal row exposure, structural-recursor exposure, strict-purity exposure through the axiom-query discipline, apophatic boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from Definition 855.1. The rows G, E, R, P, B are the only positive identity commitments admitted by the packet.

The closed-generator, literal-equality, and recursor obligations are read from G, E, R. These rows identify the internal substrate by displayed constructors, syntactic equality, and generator-local eliminators.

The strict-purity obligation is read from P together with the axiom-query ledger surface of chapter 654. It records dependency cleanliness as a displayed audit row rather than as host confidence.

Finally B, H, C, K, N bind the visible external boundary, transport, replay, provenance, and local name. Since the carrier contains no further coordinate, the local $\mathsf{NameCert}$ obligations are exhausted by the displayed packet. □

Theorem 855.3 (BEDC theory identity boundary nonescape)**.** *Every consumer of an accepted* $\mathsf{BEDCTheoryIdentity}^{\uparrow}$ *packet must read BEDC.MetaCIC identity through the finite rows*

$$G, E, R, P, B, H, C, K, N.$$

The consumer cannot replace those rows by a subtractive definition relative to classical CIC, by a host conversion rule, by quotient or extensional equality, by an unstated axiom dependency, or by an ambient semantic principle outside the apophatic boundary row.

Proof. Project the accepted packet through Definition 855.1. The identity-facing content is exactly the five rows G, E, R, P, B, transported and replayed through H and C with provenance and naming in K, N.

The positive reading of chapter 9454 is therefore a row read from the packet. Closed generators, literal AST equality, structural recursors, strict purity, and visible external boundaries are not inferred from a comparison theory.

The excluded objects have no carrier coordinate. A consumer may compare BEDC.MetaCIC with classical CIC after reading the packet, but it cannot use that comparison as the source of the packet's identity or as a route around the displayed apophatic boundary. □

Theorem 855.4 (BEDC theory identity commitment factorization)**.** *For every accepted* $\mathsf{BEDCTheoryIdentity}^{\uparrow}$ *packet*

$$I = (G, E, R, P, B, H, C, K, N),$$

any downstream MetaCIC identity consumer that classifies a read as internal, boundary-visible, or comparison-only factors the read through exactly one of the five displayed commitment rows G, E, R, P, B, transported by H, replayed by C, and recorded by K, N. No host conversion, quotient equality, hidden axiom dependency, or subtractive classical-CIC comparison can serve as the factoring row.

Proof. Project the accepted packet through Definition 855.1. The only identity-facing commitment rows in the carrier are G, E, R, P, B; the remaining coordinates H, C, K, N supply transport, replay, provenance, and the local name.

Apply Definition 855.2 to identify the complete NameCert surface. An internal read lands on G, E, R, or P according to whether it reads closed generators, literal AST equality, structural recursors, or strict-purity evidence. A boundary-visible read lands on B because the apophatic boundary is the only displayed coordinate for external supply.

A comparison-only read may compare the already-read packet with another calculus, but Definition 855.3 excludes any off-carrier source for the identity classification. Thus host conversion, quotient equality, hidden axiom dependencies, and subtractive classical-CIC comparisons cannot be factoring rows.

The existing Lean alignment point BEDC.Derived.BedcTheoryIdentityUp.BedcTheoryIdentityTasteGate_single_ca supplies the formal single-carrier touchpoint: all formal consumers are aligned to the same carrier before field-faithful reads are compared. Hence every admitted downstream read factors through exactly one displayed commitment row and then through H, C, K, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{BEDCTheoryIdentity}^{\uparrow}}$.

Constructive story: A $\mathsf{BEDCTheoryIdentity}^{\uparrow}$ packet is generated from closed-generator rows, literal AST equality rows, structural-recursor rows, strict-purity rows, apophatic-boundary rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BEDCTheoryIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 855.1, Definition 855.2, Definition 855.3, and Definition 855.4 specify the finite identity packet, its local NameCert obligations, its comparison non-escape boundary, and its downstream read factorization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BEDCTheoryIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove a Lean theorem for $\mathsf{BEDCTheoryIdentity}^{\uparrow}$, does not close a standard bridge, and does not define BEDC.MetaCIC by reference to another calculus.

Upgrade path: Lean formalization should encode the carrier and checked obligations under BEDC.Derived.BedcTheoryIdentityUp.

Chapter 856

A Concrete Naming Certificate for MetaCICTheoryIdentity$^{\uparrow}$

The MetaCICTheoryIdentity$^{\uparrow}$ packet turns the identity principle of chapter 9454 into a finite BEDC NameCert object. It sits beside the closure-trace certificate of chapter 795 and the kernel-acceptance trace of chapter 557: those siblings expose accepted closure rows and kernel admission rows, while this chapter records the five positive commitments that make BEDC.MetaCIC a theory in its own right.

856.1 Carrier and classifier

Definition 856.1 (MetaCIC theory-identity carrier)**.** *A* MetaCICTheoryIdentity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, E, R, P, A, H, C, Q, N)$$

with the following displayed rows:

1. *G is the closed-generator row, naming the inductive constructors, recursor rules, substitution rows, beta-step rows, and finite beta-chain rows admitted by the BEDC.MetaCIC substrate;*
2. *E is the literal AST equality row, recording that object comparison is syntactic equality of displayed MetaCIC abstract-syntax rows and not semantic equality inherited from another calculus;*
3. *R is the structural-recursion row, recording that admissible computation is generated by the displayed recursors on the closed syntax and history carriers;*
4. *P is the strict-purity row, recording the absence of additional axioms, classical choice, quotient soundness, proposition extensionality, and host semantic conversion in the identity packet;*
5. *A is the apophatic-boundary row, recording the finite list of absent exports: closed subject reduction, unrestricted conversion decidability, global normalization, confluence outside the named fragments, and kernel soundness;*
6. *H records componentwise* hsame *transports for the generator, equality, recursor, purity, apophatic boundary, provenance, and local naming rows;*
7. *C records displayed* Cont *routes by which audit chapters read the five commitment rows without adding a semantic oracle;*

8. *Q is the* Pkg *provenance over chapter 9454, chapter 795, chapter 557, BEDC.MetaCIC generator rows,* hsame, Cont, *and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTheoryIdentity}^{\uparrow}}$ *row.*

The carrier contains no comparison embedding into classical CIC, no quotient of syntax by a host semantics, no erased proof-identity principle, and no imported kernel-correctness theorem.

Theorem 856.2 (MetaCIC theory-identity NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *carrier*

$$I = (G, E, R, P, A, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTheoryIdentity}^{\uparrow}}$ *surface consists of five finite positive commitments:*

1. *closed-generator exposure through G;*

2. *literal AST equality exposure through E;*

3. *structural-recursion exposure through R;*

4. *strict-purity exposure through P;*

5. *apophatic-boundary exposure through A.*

The rows H, C, Q, N transport, route, package, and name only those five commitments, so the packet records BEDC.MetaCIC identity as finite BEDC data rather than as a comparison claim about another calculus.

Proof. Project the displayed packet I. The five positive coordinates G, E, R, P, A are the only theory-identity coordinates before transport, route replay, provenance, or local naming is considered.

Read H and C componentwise. They preserve the five displayed coordinates and provide only the consumer routes named by the carrier, so no route can add an external semantic equality, quotient, or host conversion rule.

Finally inspect Q and N. Provenance ties the packet to the source identity principle and to the sibling closure and kernel traces, while local naming packages the same finite surface. Since no further coordinate occurs in I, the listed rows exhaust the NameCert obligations. □

Theorem 856.3 (MetaCIC theory-identity closed-generator scope)**.** *Every generator read from an accepted* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *carrier factors through the closed-generator row G and the structural-recursion row R. The read may cite the closure-trace and kernel-acceptance siblings through Q, but it cannot introduce an unlisted constructor, host reduction rule, or semantic generator outside the displayed BEDC.MetaCIC substrate.*

Proof. Start with the carrier projection. The generator coordinate is G, and the computation coordinate that may replay it is R.

The provenance row Q can point to accepted closure and kernel traces, but those traces are read as finite BEDC rows. They do not add a second generator surface to I.

Thus every generator read remains inside G, R, Q, N, and any unlisted constructor or semantic rule would require a coordinate absent from the carrier. □

Theorem 856.4 (MetaCIC theory-identity literal-equality boundary)**.** *Every equality read from an accepted* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *carrier factors through the literal AST equality row E and the componentwise* hsame *data in H. It exports syntactic comparison of displayed MetaCIC rows and no quotient equality, proof irrelevance, host semantic equivalence, or classical CIC identity transport.*

Proof. Read equality through E. The row is part of the displayed carrier and names literal AST comparison as the local equality surface.

Transport through H is componentwise, so it can move displayed rows only as the same carrier coordinates. It does not collapse syntax into a quotient or import a semantic equivalence from a host theory.

Therefore the equality boundary is literal and local: the packet exposes AST equality and excludes the stronger equality principles listed in the statement. □

Theorem 856.5 (MetaCIC theory-identity purity boundary)**.** *Every purity read from an accepted* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *carrier factors through the strict-purity row P and the package provenance row Q. The packet records a first-principles BEDC.MetaCIC surface with no additional axiom coordinate, no classical-choice coordinate, no quotient-soundness coordinate, no proposition-extensionality coordinate, and no host semantic conversion coordinate.*

Proof. Project the purity row P. It is the only coordinate in I that can be read as a purity ledger.

The provenance row Q names where the identity packet is sourced and which sibling finite traces it may cite. Provenance can identify those rows, but it cannot create an additional axiom or host-semantics coordinate.

Hence the listed excluded principles are outside the carrier. A consumer reading purity receives the finite strict-purity row and nothing stronger. □

Theorem 856.6 (MetaCIC theory-identity public export boundary)**.** *The public export of an accepted* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *carrier is exactly the finite identity packet*

$$G, E, R, P, A, H, C, Q, N.$$

Downstream MetaCIC audit chapters may consume the closed-generator, literal-equality, structural-recursion, strict-purity, apophatic-boundary, transport, route, provenance, and local naming rows. They may not consume a closed subject-reduction theorem, unrestricted conversion decider, global normalization theorem, full confluence theorem, kernel-soundness theorem, or comparison embedding into classical CIC.

Proof. Apply the NameCert-obligation theorem to obtain the five positive commitments and their routing rows.

The closed-generator, literal-equality, structural-recursion, strict-purity, and apophatic-boundary rows are public because they are displayed coordinates of the carrier. The transport, continuation, package, and naming rows are public only as routes back to those same coordinates.

The excluded exports are not coordinates of I. Since every consumer route factors through the displayed packet, the public surface is exactly the finite row inventory in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTheoryIdentity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ packet is generated from closed-generator rows, literal AST equality, structural recursors, a strict-purity row, an apophatic-boundary row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTheoryIdentity}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 856.1, Definition 856.2, Definition 856.3, Definition 856.4, Definition 856.5, and Definition 856.6 bind the finite theory-identity packet and its public boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTheoryIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not provide Lean verification, closed subject reduction, unrestricted conversion decidability, global normalization, full confluence, kernel soundness, a comparison embedding into classical CIC, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, five-row obligation exactness, transport and route replay, provenance, local naming, and non-escape for the same finite packet.

Chapter 857

A Concrete Naming Certificate for $\mathsf{BoundedRecursorReplay}^{\uparrow}$

The $\mathsf{BoundedRecursorReplay}^{\uparrow}$ packet realizes authorized recursor replay as a finite NameCert object. It sits downstream of the recursor-generator packet in chapter 534, the authorized generator-recursor packet in chapter 525, and the legal eliminator-call packet in chapter 541. Its role is narrower than any of those chapters: it replays one accepted recursor call under a displayed descent budget, exposes the generated row, and records the audit boundary that prevents the replay from becoming an unrestricted recursion principle.

Definition 857.1 (BoundedRecursorReplay carrier)**.** *A* $\mathsf{BoundedRecursorReplay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, M, B, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted inductive-signature row, including the constructor inventory, recursive argument positions, strict-positivity gate, and emitted recursor name;*
2. *M is the checked motive row used by the bounded replay;*
3. *B is the branch-package row, with exactly one accepted branch package for each constructor displayed by I;*
4. *D is the descent-budget row, recording the finite replay bound and the structural subdata route consumed at each recursive call;*
5. *R is the bounded replay output row produced by running the authorized recursor under I, M, B, D;*
6. *A is the audit row, recording kernel acceptance, branch coverage, structural descent replay, source-axiom refusal, and dependency-purity refusal for the replayed output;*
7. *H records componentwise* hsame *transports for I, M, B, D, R, A, the route row, the provenance row, and the local naming row;*
8. *C records the displayed* Cont *routes by which consumers read the signature, motive, branch package, descent budget, replay output, audit row, provenance, and local certificate;*
9. *P is the* Pkg *provenance over* BHist*, accepted constructor signatures, checked motives, branch packages, descent budgets, bounded replay outputs, audit rows,* hsame*,* Cont*, and neighboring recursor certificates;*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRecursorReplay}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $I, M, B, D, R, A, H, C, P, N$.

Theorem 857.2 (BoundedRecursorReplay NameCert obligations)**.** *For the carrier of Definition 857.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedRecursorReplay}^{\uparrow}}$ *surface consists of carrier admission for*

$$Q = (I, M, B, D, R, A, H, C, P, N),$$

accepted-signature replay for I, motive admission for M, branch completeness for B, descent-budget exactness for D, bounded output exposure for R, audit replay for A, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, local naming, and non-escape from unbounded recursion.*

Proof. Carrier admission is projection from Definition 857.1. The finite tuple displays every coordinate that a consumer can inspect.

The signature, motive, and branch obligations are inherited as replay inputs. The accepted signature row I names the only recursor available to the packet, M names the checked motive, and B supplies one branch package for each constructor displayed by I.

The bounded-replay obligations are read from D, R, A. The descent-budget row records the finite replay bound and the structural subdata route, the output row records the result of that bounded replay, and the audit row records the kernel, branch, descent, source, and dependency checks that make the row public.

Finally H, C, P, N package the replay surface. Transport and continuation access preserve the displayed rows, provenance cites the neighboring recursor certificates, and the local naming row exposes no coordinate for an unbounded recursion principle. □

Theorem 857.3 (BoundedRecursorReplay non-escape)**.** *Every consumer read of an accepted* $\mathsf{BoundedRecursorReplay}^{\uparrow}$ *packet factors through*

$$I, M, B, D, R, A, H, C, P, N.$$

It exports no unlisted constructor, no omitted branch, no unchecked motive, no recursive call outside the descent budget, no output outside the bounded replay row, no hidden eliminator, no unrestricted recursion principle, and no dependency route escaping the audit row.

Lean4 ✓ `BEDC.Derived.BoundedRecursorReplayUp.BoundedRecursorReplayTasteGate single carrier alignment`

Proof. Start with the NameCert obligations. They place the replay output downstream of the accepted signature, checked motive, complete branch package, descent budget, audit row, transports, routes, provenance, and local naming row.

For constructor and branch escape, inspect I and B. The signature row supplies the only constructor inventory, and the branch package supplies exactly the branch rows consumed by the bounded replay.

For recursion escape, inspect D and R. A recursive call can contribute to the public output only when the descent-budget row records it as structural subdata of the accepted signature and within the displayed bound.

For dependency escape, inspect A, P, N. The audit row and provenance row record the kernel, source, and dependency boundary for the replay output. Since no carrier coordinate names an unrestricted recursion principle, hidden eliminator, or outside dependency route, no consumer can recover one from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRecursorReplay}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRecursorReplay}^{\uparrow}$ packet is generated from an accepted inductive signature, checked motive, complete branch package, finite descent budget, bounded replay output, audit row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRecursorReplay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 857.1, Definition 857.2, and Definition 857.3 bind bounded recursor replay as a finite NameCert packet over accepted signature, motive, branch, descent-budget, output, audit, route, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRecursorReplay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, general kernel soundness, all positivity checking, all termination checking, unbounded recursion, or any external classical principle.

Upgrade path: Obligation closure requires checked carrier admission, accepted-signature replay, motive admission, branch completeness, descent-budget exactness, bounded output exposure, audit replay, and non-escape for the same finite packet.

Chapter 858

A Concrete Naming Certificate for $\mathsf{ClosedNormalRouteCertificate}^{\uparrow}$

The $\mathsf{ClosedNormalRouteCertificate}^{\uparrow}$ packet realizes the route consumed by the closed-normal consistency main result of chapter 9376. It sits beside the main-result packet of chapter 764, the normal false exclusion packet of chapter 732, the subject-reduction obstruction packet of chapter 731, and the candidate-evidence route of chapter 758. Those siblings supply the endpoint exclusion, obstruction boundary, candidate-mediated route, and consumer-facing main-result surface; the present packet records the finite route certificate that threads them without promoting the boundary to full closed consistency.

Definition 858.1 (Closed-normal route certificate carrier)**.** *A* $\mathsf{ClosedNormalRouteCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, E, O, K, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{ClosedNormalConsistencyMain}^{\uparrow}$ *row naming the checked main-result endpoint as a closed beta-strong false-inhabitation exclusion;*
2. *E is the* $\mathsf{NormalFalseExclusion}^{\uparrow}$ *row carrying the already-normal false-endpoint exclusion consumed by the endpoint route;*
3. *O is the* $\mathsf{MetaCICSubjectReductionObstruction}^{\uparrow}$ *row recording that arbitrary closed proofs are not transported into the normal fragment by this certificate;*
4. *K is the* $\mathsf{CandidateEvidenceSNRoute}^{\uparrow}$ *row naming the candidate-mediated route that supplies the strong-normalization side of the normal endpoint;*
5. *R is the closed-normal route row from K, E, O to the main-result endpoint M;*
6. *H stores componentwise* hsame *transports for the sibling rows and the route row;*
7. *C stores displayed* Cont *routes from the sibling packets to the route row and from the route row to the main-result consumer;*
8. *P records* Pkg *provenance over the main-result chapter and the four sibling concrete packets;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalRouteCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for full closed consistency, closed subject reduction, unrestricted normalization, quotient proof conversion, host evaluation, ambient consistency, or a proof that every closed false proof reaches the normal endpoint.

Lean4 ✓ `BEDC.Derived.ClosedNormalRouteCertificateUp.ClosedNormalRouteCertificateTasteGate single carrier alignment`

Theorem 858.2 (Closed-normal route certificate NameCert obligations)**.** *Every accepted* ClosedNormalRouteCertifi *packet binds exactly the finite obligation surface of Definition 858.1: carrier admission for* $Q = (M, E, O, K, R, H, C, P, N)$*, sibling-row exactness for the main-result endpoint, normal false exclusion, subject-reduction obstruction, and candidate-evidence route, route-row exactness for* R*, componentwise classifier stability through* H*, displayed continuation stability through* C*, package provenance through* P*, local naming through* N*, and non-escape from full closed consistency or closed subject reduction.*

Proof. Project the accepted carrier. Admission exposes only the finite tuple $Q = (M, E, O, K, R, H, C, P, N)$, so every public read starts from the four sibling rows and the displayed route row.

The positive endpoint is M. The rows E and K supply the already-normal false exclusion and the candidate-mediated route that feed the endpoint, while O records the subject-reduction obstruction that prevents the packet from being reread as full closed consistency.

The structural rows H, C, P, N transport, replay, package, and name only this route surface. Since no coordinate supplies closed subject reduction, unrestricted normalization, quotient proof conversion, host evaluation, or ambient consistency, the accepted packet certifies only the closed-normal endpoint route displayed by R. □

Theorem 858.3 (Closed-normal route certificate nontrivial witness obligation)**.** *The* ClosedNormalRouteCertificate *carrier has two visibly distinct packets: the packet whose rows* $M, E, O, K, R, H, C, P, N$ *are all* e*, and the packet whose first coordinate* M *is* e0(e) *while* E, O, K, R, H, C, P, N *remain* e*. The classifier and field-faithful readback must distinguish these packets through the first coordinate. This is the paper obligation consumed by the Lean follow-up* `instance : Nontrivial ClosedNormalRouteCertificateUp`*.*

Proof. Display the two finite tuples. They agree on E, O, K, R, H, C, P, N and differ only at the first coordinate, where one tuple has e and the other has e0(e).

Constructor no-confusion for the finite tuple reduces equality of packets to equality of their displayed coordinates. The first-coordinate comparison would require the empty history constructor to coincide with the nonempty e0 constructor, which is refused by the displayed history constructors.

The classifier reads packets componentwise, so it cannot identify the two tuples while preserving the first coordinate. Field-faithful readback records the same first-coordinate distinction at the TasteGate-facing surface, giving the required non-vacuity witness for the later Lean instance. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalRouteCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedNormalRouteCertificate}^{\uparrow}$ packet is generated from the main-result endpoint row, normal false exclusion row, subject-reduction obstruction row, candidate-evidence route row, route certificate row, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalRouteCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 858.1 and Definition 858.2 bind the finite NameCert route from candidate evidence and normal false exclusion through the subject-reduction obstruction boundary to the closed-normal main-result endpoint.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalRouteCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full closed consistency, closed subject reduction, unrestricted normalization, quotient conversion, host evaluation, ambient consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-row exactness, route-row exactness, classifier stability, continuation stability, provenance, local naming, and non-escape for the same finite route certificate.

Chapter 859

A Concrete Naming Certificate for $\mathsf{BedcSelfTheory}^{\uparrow}$

The $\mathsf{BedcSelfTheory}^{\uparrow}$ packet realizes the identity principle of chapter 9454 as a finite NameCert surface. It is not a new metatheory above BEDC.MetaCIC. It records the finite rows by which the five defining commitments of that chapter are made available to downstream audit consumers: closed generators, literal AST equality, structural recursor locality, strict axiom-purity reporting, and apophatic external boundaries. Horizontally, it reads the accepted-theorem audit face of chapter 546, the legal eliminator-call discipline of chapter 541, the audit-gate boundary of chapter 526, and the external-socket naming discipline of chapter 543.

Definition 859.1 (BEDC self-theory commitment carrier)**.** *A* $\mathsf{BedcSelfTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, E, R, A, X, L, H, C, P, N)$$

with the following displayed rows:

1. *G is the closed-generator row, naming the finite inductive carrier inventory read as BEDC.MetaCIC substrate data;*
2. *E is the literal AST equality row, recording that type comparison for the packet is the displayed syntactic equality route and not an external conversion principle;*
3. *R is the structural recursor row, exposing the generator-local eliminators and their constructor-indexed branch discipline;*
4. *A is the strict axiom-purity row, reading the source-token and proof-ancestry audit boundary through the finite audit gates;*
5. *X is the apophatic external-boundary row, recording named sockets, setup fields, and GAP positions as visible external supply rather than hidden substrate content;*
6. *L is the self-theory ledger row, aligning the five commitments with the chapter identity of chapter 9454;*
7. *H records componentwise* hsame *transports for generator, equality, recursor, audit, apophatic, ledger, route, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes by which consumers read each commitment without replacing it by an ambient host principle;*

9. *P is the* Pkg *provenance over* BHist*, closed generators, literal equality, authorized recursors, audit gates, apophatic names,* hsame*,* Cont*, and local NameCert rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BedcSelfTheory}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed tuple $G, E, R, A, X, L, H, C, P, N$.

Theorem 859.2 (BEDC self-theory NameCert obligations)**.** *For the carrier of Definition 859.1, the local* $\mathsf{NameCert}_{\mathsf{BedcSelfTheory}^{\uparrow}}$ *surface consists of five finite commitments: closed-generator admission for G, literal-equality exactness for E, generator-local recursor authorization for R, strict audit visibility for A, and apophatic external-boundary visibility for X. The ledger, transport, continuation, provenance, and local naming rows L, H, C, P, N bind these commitments as one finite packet and add no additional substrate principle.*

Proof. Carrier admission is projection from Definition 859.1. The tuple exposes the generator, equality, recursor, audit, apophatic, ledger, transport, route, provenance, and naming rows as finite coordinates.

The first three commitments are internal substrate commitments. The closed-generator row G lists only displayed inductive carriers, the equality row E records literal AST comparison, and the recursor row R cites only generator-local eliminators whose branch discipline is already visible in the authorized-recursion surface of chapter 541.

The audit and apophatic commitments are boundary commitments. The row A is read through the audit-gate packet of chapter 526, while X is read through the external-name discipline of chapter 543. Thus purity reports and external supplies remain visible rows, not hidden assumptions.

Finally L, H, C, P, N package the same five commitments. They align the packet with chapter 9454, transport and replay the displayed rows, record provenance through the neighboring NameCert surfaces, and expose local naming without adding a sixth commitment. □

Theorem 859.3 (BEDC self-theory closed generator boundary)**.** *Every accepted* $\mathsf{BedcSelfTheory}^{\uparrow}$ *carrier reads BEDC.MetaCIC objects through the closed-generator row G and the literal-equality row E. A downstream consumer cannot use the packet to import an unlisted constructor, a quotient representative, a host semantic equivalence, or a conversion rule not displayed by the carrier.*

Proof. Let an accepted carrier be $S = (G, E, R, A, X, L, H, C, P, N)$. The object-facing rows are G and E: G names the finite constructor inventory, and E names the syntactic equality route used when the packet compares type-level data.

The remaining rows cannot enlarge this inventory. Recursor access in R consumes the same generator row, audit access in A reports proof and source boundaries, and apophatic access in X records external supply as external.

Transport, continuation, provenance, ledger, and naming rows preserve the displayed packet. Since none of H, C, P, L, N supplies a constructor, quotient representative, host semantic equivalence, or extra conversion rule, the closed-generator boundary is exactly the displayed one. □

Theorem 859.4 (BEDC self-theory structural recursor locality)**.** *Every recursor read exported by an accepted* $\mathsf{BedcSelfTheory}^{\uparrow}$ *carrier is local to the closed generator named by G and the recursor row R. It exports no induction principle over an arbitrary semantic domain, no unchecked motive, no omitted branch, and no recursive route outside the displayed structural eliminator.*

Proof. The row R is interpreted together with G. Thus any exported eliminator is tied to the constructor inventory named by the same carrier.

Motive and branch discipline is inherited through the authorized-recursion surface cited in the provenance row P. A consumer may inspect the recursor only through the displayed route C, so the motive, branch package, and structural descent ledger remain part of the same finite read.

No other row contains an induction principle. The audit row reports boundaries, the apophatic row names external supply as external, and H, L, N transport, align, and name the same packet. Hence recursor access remains generator-local. □

Theorem 859.5 (BEDC self-theory non-escape)**.** *Every public read of an accepted* $\mathsf{BedcSelfTheory}^{\uparrow}$ *packet factors through*

$$G, E, R, A, X, L, H, C, P, N.$$

It exports no hidden classical-CIC parent theory, no ambient conversion rule, no undisplayed axiom dependency, no unrestricted recursion principle, and no external socket promoted to internal theorem content.

Proof. The packet inventory is finite and displayed in Definition 859.1. A public read can project one of the five commitment rows or one of the rows that transports, routes, packages, aligns, or names them.

Classical-CIC parentage and ambient conversion are blocked by G and E. The carrier names BEDC.MetaCIC by its own generator and equality rows rather than by a subtractive comparison with another calculus.

Undisplayed axioms and unrestricted recursion are blocked by A and R. The audit row exposes source and proof-ancestry boundaries, and the recursor row remains tied to its closed generator and branch discipline.

External supply is blocked from becoming internal content by X. Since the apophatic row keeps sockets, setup fields, and GAP positions visible as boundary rows, no consumer can recover an internal theorem from them without a separate displayed route.

Therefore the self-theory packet is a finite naming certificate for the five commitments and not an escape route to unlisted principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{BedcSelfTheory}^{\uparrow}}$.

Constructive story: A $\mathsf{BedcSelfTheory}^{\uparrow}$ packet is generated from closed-generator inventory, literal AST equality, structural recursor locality, strict audit-purity rows, apophatic boundary rows, a self-theory ledger, displayed transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BedcSelfTheory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 859.1, Definition 859.2, Definition 859.3, Definition 859.4, and Definition 859.5 bind BEDC.MetaCIC self-theory identity as a finite NameCert packet over five displayed commitments.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BedcSelfTheory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove a Lean-side carrier, a bridge theorem, a global completeness theorem for BEDC.MetaCIC, a semantic comparison theorem with classical CIC, or any external principle.

Upgrade path: Obligation closure requires checked carrier admission, five-commitment NameCert obligations, closed-generator boundary, structural recursor locality, and non-escape for the same finite packet.

Chapter 860

A Concrete Naming Certificate for $\mathsf{MetaCICIdentity}^{\uparrow}$

The $\mathsf{MetaCICIdentity}^{\uparrow}$ packet realizes the framing principle of chapter 9454 as a finite NameCert surface. It sits beside the audit-map family certificate of chapter 807 and the GroundCompiler audit-map packet of chapter 836: those siblings classify audit rows and compiler-facing evidence, while this chapter names the identity packet that lets a consumer read BEDC.MetaCIC through its own substrate commitments without treating classical CIC as a defining parent theory.

Definition 860.1 (MetaCIC identity carrier)**.** *A* $\mathsf{MetaCICIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, E, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the closed-generator inventory row, listing the internal finite constructors for the term-level and certificate-level carriers used by BEDC.MetaCIC;*
2. *E is the literal AST equality row, recording that internal type identity is read as displayed syntactic equality rather than as conversion through an external reduction theory;*
3. *R is the structural recursor authorization row, naming only the eliminators supplied by the closed generators in G;*
4. *A is the strict axiom-purity witness row, recording zero dependency on choice, quotient soundness, propositional extensionality, or any other undisplayed axiom route;*
5. *B is the apophatic boundary socket, recording every request beyond G, E, R, A as external supply rather than as a hidden internal theorem;*
6. *H stores componentwise* hsame *transports for G, E, R, A, B and the local naming rows;*
7. *C stores finite* Cont *routes by which consumers read the five commitment rows, the transports, package provenance, and local naming row;*
8. *P is the* Pkg *provenance over chapter 9454, chapter 807, chapter 836,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICIdentity}^{\uparrow}}$ *row.*

The carrier is an identity certificate for BEDC.MetaCIC's own displayed substrate. It is not a comparison theorem against classical CIC and does not import any conversion, quotient, extensionality, or choice principle from a neighbouring calculus.

Lean4 ✓ `BEDC.Derived.MetaCICIdentityUp.MetaCICIdentityTasteGate single carrier alignment`

Theorem 860.2 (MetaCIC identity NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICIdentity}^{\uparrow}}$ *surface for Definition 860.1 consists of five finite rows: carrier admission for I; classifier stability for the closed-generator inventory, literal AST equality, structural recursor authorization, strict axiom-purity witness, and apophatic boundary socket; boundary exactness for every read outside those five commitment rows;* Pkg *provenance over G, E, R, A, B, H, C; and local naming exactness for N.*

Proof. Carrier admission is projection of the finite tuple I. The generator, equality, recursor, purity, boundary, transport, continuation, provenance, and naming coordinates are all displayed rows of the same BHist packet.

Classifier stability is componentwise. Transport through H preserves the closed-generator inventory, keeps literal equality attached to the same AST row, keeps recursor authorization local to the displayed constructors, and keeps the strict-purity witness attached to the same audit boundary.

The boundary row handles the remaining reads. If a consumer asks for conversion equality, quotient-of-germs equality, extensionality, choice, or a host theorem outside the displayed substrate commitments, the route lands in B rather than in a new internal theorem row.

The package and naming rows then repack exactly those finite coordinates. Thus the obligation surface is the five-row NameCert surface for the identity packet itself. □

Lean4 ✓ `BEDC.Derived.MetaCICIdentityUp.MetaCICIdentity namecert obligations`

Theorem 860.3 (MetaCIC identity boundary non-escape)**.** *For an accepted* $\mathsf{MetaCICIdentity}^{\uparrow}$ *carrier $I = (G, E, R, A, B, H, C, P, N)$, every consumer read of BEDC.MetaCIC's identity factors through the closed-generator inventory G, literal AST equality row E, structural recursor authorization R, strict axiom-purity witness A, apophatic boundary socket B, and structural rows H, C, P, N. A read outside those rows remains boundary data in B and is not exported as conversion equality, quotient soundness, propositional extensionality, choice, or a parent-theory relation to classical CIC.*

Proof. Read the consumer request through the finite routes in C. The positive routes land in the generator, equality, recursor, purity, transport, provenance, and naming rows displayed by the carrier.

For a request outside those coordinates, no row of I supplies the requested principle. The apophatic boundary socket B is the only classifier-visible destination for that request.

Boundary exactness then prevents a boundary entry from being reclassified as a theorem of the substrate. The identity packet therefore exports only the displayed BEDC.MetaCIC commitments and keeps comparison-facing or host-theoretic supply outside the internal carrier. □

Theorem 860.4 (MetaCIC identity comparison is secondary)**.** *Let $I = (G, E, R, A, B, H, C, P, N)$ be an accepted* $\mathsf{MetaCICIdentity}^{\uparrow}$ *carrier. Any comparison with classical CIC, audit-map methodology, or GroundCompiler audit evidence may read I only after the five identity rows G, E, R, A, B have already been fixed. Such a comparison can diagnose shared syntax, differing equality principles, differing axiom dependencies, or external sockets, but it cannot define the identity of BEDC.MetaCIC or add a new substrate commitment to I.*

Proof. Project the carrier. The identity rows G, E, R, A, B are displayed before the transport, route, package, and naming rows that expose the packet to downstream consumers.

Use the audit-map family and GroundCompiler audit-map chapters only as consumer contexts. They may inspect finite rows through C, P, N, but their own audit surfaces do not supply another generator, equality rule, recursor principle, purity witness, or internal theorem row for I.

Finally apply boundary non-escape. If a comparison requests a classical conversion rule, extensionality principle, quotient principle, choice principle, or parent-theory relation, the request is recorded in B. Therefore comparison remains diagnostic and secondary to the displayed identity packet. □

Theorem 860.5 (MetaCIC identity generator-recursor coverage)**.** *Let $I = (G, E, R, A, B, H, C, P, N)$ be an accepted* $\mathsf{MetaCICIdentity}^{\uparrow}$ *carrier. Every identity read that uses the closed-generator inventory and structural recursor authorization exposes exactly the finite rows $G, E, R, A, B, H, C, P, N$: G supplies the generators, E supplies literal AST identity, R supplies the authorized structural recursors, A supplies strict axiom-purity, B supplies the boundary socket, and H, C, P, N supply transport, replay, provenance, and local naming. This recursor coverage cannot become classical conversion, quotient equality, extensionality, choice, or a parent-CIC relation.*

Proof. Start with the carrier and NameCert obligations, Definition 860.1 and Definition 860.2. They display the generator inventory, literal AST equality row, structural recursor authorization, strict-purity witness, boundary socket, transport, continuation, provenance, and local naming as the whole identity packet.

Use boundary non-escape, Definition 860.3. A read that asks for conversion equality, quotient equality, extensionality, choice, or a parent-theory relation lands in B, because none of those requests is one of the generator or recursor rows exposed by G, E, R, A.

Finally apply the comparison-secondary theorem Definition 860.4. Classical CIC, audit-map methodology, and GroundCompiler evidence may inspect the already fixed rows through C, P, N, but they do not define the identity packet and do not add a second recursor source. Hence generator-recursion coverage is exactly the displayed finite route through $G, E, R, A, B, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICIdentity}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICIdentity}^{\uparrow}$ packet is generated from closed-generator inventory, literal AST equality, structural recursor authorization, strict axiom-purity witness, apophatic boundary socket, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 860.1, Definition 860.2, Definition 860.3, and Definition 860.4 define the finite identity packet, its NameCert obligations, its non-escape boundary, and its secondary comparison discipline.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean implementation, a public bridge, a classical CIC embedding, a conversion rule, quotient soundness, propositional extensionality, or choice.

Upgrade path: Obligation closure requires binding the identity packet obligations to concrete generator, AST equality, recursor, strict-purity, apophatic boundary, transport, route, provenance, and local naming rows.

Chapter 861

A Concrete Naming Certificate for AxiomDependencyTuple$^{\uparrow}$

The AxiomDependencyTuple$^{\uparrow}$ packet realizes the dependency-map reading of chapter 9455 as finite NameCert data. It follows the audit-map family discipline of chapter 9406 and sits beside the discharge-socket boundary of chapter 719: the former gives the row family shape, while the latter is a concrete sibling where missing MetaCIC supply is named rather than hidden.

Definition 861.1 (AxiomDependencyTuple carrier)**.** *An* AxiomDependencyTuple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, W, S, H, C, P, N)$$

with the following displayed rows:

1. *M is one of the three mode rows, Mode A for zero-supply internal proof, Mode B for restricted-proof discharge under displayed supply, and Mode C for counterexample or socket witness against an unstated supply;*
2. *W is the witness row, recording a proof, restricted proof, counterexample, or socket witness according to M;*
3. *S is the required-supply ledger, listing exactly the host conversion, quotient equality,* Classical.choice, propext*, ambient classical theorem, or other named supply row that the entry requires;*
4. *H records componentwise* hsame *transport for the mode, witness, required-supply, route, provenance, and naming rows;*
5. *C records the displayed* Cont *routes by which a consumer reads the mode, witness, and required-supply ledger;*
6. *P is the* Pkg *provenance over chapter 9455, chapter 9406, chapter 719,* BHist, hsame, Cont, *and* NameCert *rows;*
7. *N is the local* NameCert$_{\text{AxiomDependencyTuple}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose mode, witness, required-supply ledger, transport, continuation, provenance, and naming rows are displayed in this finite carrier.

Lean4 (definition) `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleCarrier`

Theorem 861.2 (AxiomDependencyTuple NameCert obligations)**.** *For the carrier of Definition 861.1, the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *surface consists of carrier admission for*

$$A = (M, W, S, H, C, P, N),$$

mode classification, witness transport, required-supply ledger exactness, displayed Cont *route readback,* Pkg *provenance, and local naming. The packet may name host conversion, quotient equality,* Classical.choice*,* propext*, or an ambient classical CIC theorem as supply in S, but it cannot export any of them as BEDC data.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleNameCertObligations`

Proof. Project the finite packet A. The rows M, W, S provide the mathematical payload: a mode, a witness appropriate to that mode, and the ledger of supply required by that witness.

Read H and C next. Componentwise hsame transport preserves the same mode and witness row, and every consumer route must replay the displayed ledger before it reads the tuple.

Finally inspect P and N. Provenance cites only the dependency-map chapter, the audit-map methodology, the discharge-socket sibling, and the BEDC kernel rows. Since no carrier coordinate contains host conversion, quotient equality, choice, propositional extensionality, or an ambient classical theorem as an exported BEDC proof, the named supply remains a ledger entry rather than a hidden rule. □

Theorem 861.3 (AxiomDependencyTuple mode exhaustion)**.** *Every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$ is exhausted by the three displayed modes: Mode A carries a zero-supply proof witness, Mode B carries a restricted-proof witness together with the required-supply ledger S, and Mode C carries a counterexample or socket witness naming the refused supply. Componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local* NameCert *row preserve this trichotomy and do not add a fourth mode.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleCarrier mode exhaustion`

Proof. Inspect the mode row M in the carrier of Definition 861.1. Its visible constructors are exactly the zero-supply, restricted-proof, and counterexample-or-socket alternatives.

In Mode A, the witness row W is read as an internal proof and the ledger S is empty of external supply. In Mode B, the witness row is read only together with the named required supply. In Mode C, the witness row is a counterexample or socket naming the missing supply rather than discharging it.

Transport and routing do not change the case split. The rows H, C, P, N replay the same mode, witness, and ledger coordinates, so no consumer can introduce host conversion, quotient equality, Classical.choice, propext, an ambient classical CIC theorem, or any fourth mode outside the displayed packet. □

Theorem 861.4 (AxiomDependencyTuple supply-ledger readback)**.** *Every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet exposes the required-supply row S before the witness row W can be consumed. If the tuple names host conversion, quotient equality,* Classical.choice*,* propext*, an ambient theorem, or any other external supply, that name remains a ledger row in S and never becomes an exported BEDC rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleSupplyLedgerReadback`

Proof. Begin with the carrier projection

$$A = (M, W, S, H, C, P, N).$$

By Definition 861.1, the consumer-facing data contains both the witness coordinate W and the required-supply ledger S.

Apply Definition 861.2. The local $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ surface includes witness transport and required-supply ledger exactness, so any route that reads W must also read the displayed supply row attached to that witness.

Finally use Definition 861.3. Mode A has no external supply to export, Mode B consumes W only under the displayed ledger, and Mode C records a counterexample or socket rather than a discharge. Thus the named host or classical material remains visible audit supply in S. □

Theorem 861.5 (AxiomDependencyTuple zero-supply mode exactness)**.** *For every accepted* AxiomDependencyTup *packet $A = (M, W, S, H, C, P, N)$ in Mode A, the required-supply ledger S is exposed as empty before the zero-supply witness row W is read. Consequently the packet consumes only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *rows, with no host conversion, quotient equality,* Classical.choice*,* propext*, or ambient theorem available as BEDC proof data.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTuple zero supply mode exactness`

Proof. Project the tuple through Definition 861.1. The public carrier has exactly the rows M, W, S, H, C, P, N, so the Mode A claim can only mention the displayed mode, witness, ledger, transport, route, provenance, and naming coordinates.

Apply Definition 861.2 and Definition 861.3. The local NameCert row binds witness transport and required-supply ledger exactness, while mode exhaustion identifies Mode A as the zero-supply case.

Now use Definition 861.4. The ledger S is read before W, and in Mode A that ledger contains no named external supply. Finally, Definition 861.8 keeps every Cont route from converting an absent ledger entry into a BEDC proof rule. □

Theorem 861.6 (AxiomDependencyTuple restricted-supply ledger exactness)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet $A = (M, W, S, H, C, P, N)$ in Mode B, each host conversion, quotient equality,* Classical.choice *use,* propext *use, or ambient theorem required by the restricted witness appears as a visible row of S before W is read. None of these named rows is converted into an internal BEDC rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleRestrictedSupplyLedger exactness`

Proof. Begin with the carrier and NameCert surfaces. By Definition 861.1 and Definition 861.2, a restricted packet records the witness row and its required-supply ledger as separate public coordinates, checked by local naming obligations.

The mode split is exhaustive by Definition 861.3. In Mode B, the witness is the restricted-proof witness, and the supply required by that witness is not part of the BEDC kernel surface.

Apply Definition 861.4. Every named host conversion, quotient equality, choice, propositional extensionality, or ambient theorem must be read from S before W is consumed.

Finally use Definition 861.8. A displayed route may replay the ledger and witness coordinates, but it has no constructor that turns a required-supply ledger row into an exported BEDC proof principle. □

Theorem 861.7 (AxiomDependencyTuple downstream consumer supply boundary)**.** *Every downstream consumer of an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet factors its read through the displayed coordinates* M, W, S, H, C, P, N*. Hence counterexample, socket, and restricted-proof supply remains an audit row in the tuple rather than a hidden proof principle available to the consumer.*

Proof. Use Definition 861.1 to identify the only public coordinates of the accepted packet. The downstream consumer has no carrier component beyond M, W, S, H, C, P, N from which to read additional supply.

The local NameCert and mode rows determine how those coordinates are consumed. By Definition 861.2 and Definition 861.3, the consumer first selects one of the three modes and then reads the corresponding witness and ledger rows under the displayed transport, route, provenance, and naming data.

The ledger boundary is fixed by Definition 861.4. Restricted supply, host supply, quotient supply, choice, propositional extensionality, ambient theorems, counterexamples, and sockets are visible only as rows attached to the accepted tuple.

Finally apply Definition 861.8. Since every consumer route merely replays the displayed coordinates, no downstream audit or dependency-map consumer can promote those audit rows into hidden BEDC proof data. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleDownstreamConsumerSupplyBoundary`

Theorem 861.8 (AxiomDependencyTuple route nonexport)**.** *Every displayed* Cont *route in the coordinate* C *replays the* M, W, S, H, P, N *components of an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet componentwise. The route cannot introduce a fourth mode and cannot convert the required-supply row* S *into an internal proof principle.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleRouteNonexport`

Proof. Project the route coordinate C from Definition 861.1. Its declared role is readback for the mode, witness, and required-supply ledger, with the remaining rows supplying transport, provenance, and naming.

Apply Definition 861.2 to the same carrier. The NameCert obligations include displayed Cont route readback and componentwise transport, so a route is checked against the rows already present in A.

The mode alternatives are exhausted by Definition 861.3. Since C only replays the displayed coordinates, it has no constructor by which to create another mode or turn a required-supply ledger entry into a BEDC proof rule. □

Theorem 861.9 (AxiomDependencyTuple consumer exhaustion)**.** *Every downstream consumer of an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *tuple is exhausted by the mode row* M*, witness row* W*, required-supply ledger* S*, displayed route row* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *row* N*, together with their componentwise transport* H*. There is no hidden consumer authority outside this finite packet.*

Proof. Read the carrier of Definition 861.1. The packet lists the only coordinates available to a consumer:

$$M,\ W,\ S,\ H,\ C,\ P,\ N.$$

The obligation theorem Definition 861.2 assigns each consumer-facing role to one of these coordinates: carrier admission, mode classification, witness transport, ledger exactness, route readback, provenance, and local naming.

Mode exhaustion then removes the only remaining possibility for extra consumer authority. By Definition 861.3, every accepted packet lies in Mode A, Mode B, or Mode C, and transport or route readback preserves that case split. Hence any downstream consumer is confined to the finite displayed rows. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleConsumerExhaustion`

Theorem 861.10 (AxiomDependencyTuple zero-supply mode soundness)**.** *In Mode A, an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet consumes the witness row W only when the required-supply ledger S records no external supply. The zero-supply branch cannot smuggle host conversion, quotient equality,* Classical.choice*,* propext*, or any ambient classical theorem through the route row C, provenance row P, local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *row N, or componentwise transport row H.*

Proof. Project the carrier of Definition 861.1. The coordinates M, W, S, H, C, P, N are displayed separately, so the witness row and the required-supply row are not identified by the packet shape.

Apply Definition 861.3. In the Mode A branch, W is the zero-supply proof witness and S has no external host or classical supply to replay.

The ledger boundary is then fixed by Definition 861.4. A consumer can read W only through the visible ledger row, and in this branch that ledger contains no host conversion, quotient equality, choice, propositional extensionality, or ambient classical theorem.

Finally use Definition 861.8 and Definition 861.9. Routes and consumers are exhausted by the displayed rows, so no hidden coordinate remains from which the Mode A branch could import external supply. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleZeroSupplySemanticNameCert`

Theorem 861.11 (AxiomDependencyTuple restricted-supply ledger nonpromotion)**.** *In Mode B, an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet reads the witness row W only together with the displayed required-supply row S. Every named host conversion, quotient equality,* Classical.choice*,* propext*, ambient classical theorem, or other supply entry in S remains audit supply for the restricted witness and is not promoted to an internal BEDC proof rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleRestrictedSupplyLedgerNonpromotion`

Proof. Start from the carrier

$$A = (M, W, S, H, C, P, N).$$

The carrier separates the restricted witness coordinate from the supply ledger coordinate, so the witness cannot be read as though the ledger were absent.

By Definition 861.3, the Mode B branch is the restricted-proof branch: W is consumed under the named supply row S, and it is neither a zero-supply proof nor a counterexample socket.

Apply Definition 861.4. The full row S is exposed whenever W is consumed, including every named external supply entry used by the restricted witness.

Definition 861.8 prevents the displayed route from turning S into a rule. Together with Definition 861.9, this confines every consumer to audit readback of the ledger rather than promotion of the ledger into BEDC supply. □

Theorem 861.12 (AxiomDependencyTuple counterexample socket boundary)**.** *In Mode C, an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *packet records a counterexample or socket witness against missing supply through W and S. The packet names the absent host conversion, quotient equality,* Classical.choice*,* propext*, ambient classical theorem, or other supply row, but it never discharges that supply and never exports it as a BEDC rule.*

Proof. Read the carrier in Definition 861.1. The witness row W and required-supply ledger S are visible packet coordinates, and the route, provenance, naming, and transport rows are separate from them.

Use Definition 861.3. The Mode C branch is the counterexample-or-socket branch, so W witnesses refusal or boundary data against missing supply instead of providing a discharge.

The ledger remains visible by Definition 861.4. Any named host or classical material attached to the socket is read as a required-supply row in S, not as a proof principle.

Finally apply Definition 861.8 and Definition 861.9. The continuation can only replay the displayed packet rows, and every consumer is exhausted by those rows, so the counterexample socket cannot discharge the missing supply it records. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleCounterexampleSocketBoundary`

Theorem 861.13 (AxiomDependencyTuple zero-supply row separation)**.** *In Mode A, an accepted* AxiomDependencyTuple$^{\uparrow}$ *packet exposes an internal zero-supply witness in W and an empty external-supply ledger in S before any consumer route can read the witness. The displayed* Cont *row C may replay that separation, but it cannot insert host conversion, quotient equality,* Classical.choice*,* propext*, or ambient classical supply into the Mode A branch.*

Proof. Begin with the carrier

$$A = (M, W, S, H, C, P, N).$$

By Definition 861.1, the witness row and the required-supply ledger are separate displayed coordinates.

Apply Definition 861.3. In the Mode A case, W is the internal proof witness and S is empty of external supply.

Now use Definition 861.4 and Definition 861.8. A consumer route must read the displayed ledger before consuming W, and the route has no constructor that turns an empty ledger into host or classical supply.

Finally, Definition 861.9 confines every consumer to the finite rows M, W, S, H, C, P, N. Hence Mode A exposes zero supply as a row property of the packet rather than as a hidden external assumption. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleZeroSupplyRowSeparation`

Theorem 861.14 (AxiomDependencyTuple restricted-supply ledger totality)**.** *In Mode B, every restricted-proof witness in an accepted* AxiomDependencyTuple$^{\uparrow}$ *packet is consumed only together with the full displayed required-supply row S. No consumer may read a proper subledger, erase a named supply entry, or replace S by an implicit host rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleRestrictedSupplyLedgerTotality`

Proof. Project the carrier of Definition 861.1. The Mode B branch stores the restricted-proof witness in W and its required supply in the single ledger coordinate S.

The mode split is fixed by Definition 861.3. In that branch, the witness is not a zero-supply proof and not a counterexample or socket; it is read as a restricted proof under the displayed supply row.

Apply Definition 861.4. Any route that consumes W must first expose the same row S, including every host conversion, quotient equality, choice, propositional extensionality, ambient theorem, or other named supply entry recorded there.

The route and consumer boundaries then close the case. By Definition 861.8 and Definition 861.9, the consumer has access only to the displayed tuple rows, so it cannot silently shrink S or promote a ledger entry into a BEDC proof rule. □

Theorem 861.15 (AxiomDependencyTuple counterexample socket nonexport)**.** *In Mode C, every accepted counterexample or socket witness names refused supply through the witness row W and required-supply ledger S, while the route row C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *row N prevent that named supply from becoming an exported BEDC proof rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleCounterexampleSocketNonexport`

Proof. Read the carrier definition first. The witness row W and the ledger row S are displayed separately from route, provenance, and naming data.

Apply Definition 861.3. In Mode C, the witness coordinate is a counterexample or socket, so the packet records refused supply rather than a proof discharge.

Next use Definition 861.4. Any named host conversion, quotient equality, Classical.choice, propext, ambient theorem, or other external supply remains visible in S when a consumer reads the witness.

Finally combine Definition 861.8 with Definition 861.9. The route can only replay the displayed rows, and every consumer is exhausted by those rows, so the counterexample or socket cannot export the refused supply as a BEDC rule. □

Theorem 861.16 (AxiomDependencyTuple BHist row scope)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$, the seven displayed coordinates are finite* BHist*-readable rows. The packet reads mode, witness, required-supply, transport, continuation, provenance, and naming data from those rows only, and none of the rows exports host conversion, quotient equality,* Classical.choice*,* propext*, or an ambient classical CIC theorem as BEDC supply.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleBHistRowScope`

Proof. Start with Definition 861.1. The carrier is a finite BHist packet, and the displayed packet coordinates are exactly M, W, S, H, C, P, N.

The earlier obligation rows already describe what each coordinate can expose. Use Definition 861.2 for carrier admission and local naming, Definition 861.3 for the three mode rows, and Definition 861.4 for the required supply row.

It remains to exclude hidden export. By Definition 861.8 and Definition 861.9, a consumer can only replay the displayed finite coordinates. Thus every named host or classical dependency remains a ledger entry in S, never a BEDC proof rule exported by a row of A. □

Theorem 861.17 (AxiomDependencyTuple hsame and Cont scope)**.** *In an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$, the transport row H and continuation row C preserve the separation between the mode row M, witness row W, and required-supply ledger S. Componentwise* hsame *transport and displayed* Cont *replay may carry the same tuple to a consumer, but they do not identify a witness with its supply ledger, erase a mode distinction, or create an implicit host rule.*

Proof. Read H from Definition 861.1. Its role is componentwise hsame transport for the mode, witness, required-supply, route, provenance, and naming rows, so transport is indexed by the displayed coordinates rather than by an ambient equality principle.

Read C next. The continuation row records the displayed routes by which a consumer reads the mode, witness, and required-supply ledger. Apply Definition 861.8: a route replays the existing coordinates and has no constructor for another mode or an internal host discharge.

The separation is stable in all three cases. Mode A keeps the empty supply row separate by Definition 861.13; Mode B keeps the full ledger visible by Definition 861.14; Mode C keeps the refused supply nonexported by Definition 861.15. Hence H and C transport and replay the separation instead of collapsing it. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleHsameContScope`

Theorem 861.18 (AxiomDependencyTuple Pkg and NameCert scope)**.** *For every accepted* AxiomDependencyTuple *carrier $A = (M, W, S, H, C, P, N)$, the provenance row P and local naming row N package the same finite tuple as the* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *surface. They cite the dependency-map, audit-map, discharge-socket,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows without internalizing host conversion,* Quot.sound*,* Classical.choice*,* propext*, or ambient classical theorems.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTuplePkgNameCertScope`

Proof. Project P and N from the carrier definition. The provenance row is a Pkg row over the named chapters and BEDC kernel rows, while N is the local $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ row for the same finite packet.

The NameCert surface is fixed by Definition 861.2. It admits exactly the carrier, mode classifier, witness transport, required-supply ledger, displayed route readback, provenance, and local naming obligations already present in A.

Finally combine Definition 861.16 with Definition 861.17. The tuple rows are finite and BHist-readable, while transport and continuation preserve the mode, ledger, and witness separation. Therefore P and N package the tuple as a NameCert surface and do not turn any named host or classical dependency into internal BEDC supply. □

Theorem 861.19 (AxiomDependencyTuple root-unblock spine)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$, the public dependency-tuple surface is exactly the joint reading of the following rows over the displayed coordinates: mode exhaustion for M, supply-ledger readback for S, route nonexport for C, consumer exhaustion for the whole tuple, and primitive scope for the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No additional root-unblock datum is available outside these rows.*

Proof. Start with Definition 861.1. The only carrier coordinates are

$$M,\ W,\ S,\ H,\ C,\ P,\ N.$$

The root-unblock spine is read coordinate by coordinate. Use Definition 861.3 for M, Definition 861.4 for S, Definition 861.8 for C, and Definition 861.9 for the consumer boundary of the full tuple.

It remains to account for primitive scope. This is exactly the content of Definition 861.16, Definition 861.17, and Definition 861.18. These rows exhaust the available BHist, hsame, Cont, Pkg, and NameCert interfaces, so the public dependency tuple is the conjunction of the displayed rows and nothing external to the carrier. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleRootUnblockSpine`

Theorem 861.20 (AxiomDependencyTuple mode-ledger priority)**.** *Every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$ *is consumed in mode-ledger order. In Mode A the consumer reads the empty required-supply row before using the zero-supply witness. In Mode B the consumer reads the displayed supply ledger before witness consumption. In Mode C the consumer reads the counterexample or socket row as a refusal of discharge before any attempted internal proof use.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleModeLedgerPriority`

Proof. Project the carrier and split by the mode row M. By Definition 861.3, the split has exactly the three displayed cases.

In Mode A, apply Definition 861.13. The ledger row is empty as external supply and remains separate from the zero-supply witness, so readback sees the empty ledger first.

In Mode B, apply Definition 861.14. The restricted witness is not consumed until the required-supply ledger has been displayed as the exact supply row for that witness.

In Mode C, apply Definition 861.15. The counterexample or socket row is read as a boundary witness, so the route records refusal before any attempted discharge. □

Theorem 861.21 (AxiomDependencyTuple consumer readiness)**.** *Downstream audit-map consumers may cite Definition 861.19, Definition 861.20, and the existing scope rows of this chapter as the complete* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *root-unblock package. The package exports no host conversion,* Quot.sound*,* Classical.choice*,* propext*, or ambient classical theorem as BEDC supply.*

Proof. The spine theorem identifies the whole public surface with the displayed carrier rows. The mode-ledger theorem fixes the order in which those rows are read by any consumer.

The remaining cited scope rows are Definition 861.16, Definition 861.17, and Definition 861.18. Together they show that the primitive interfaces remain finite, componentwise, routed, packaged, and locally named.

Finally use Definition 861.8 and Definition 861.9. A downstream audit-map consumer can replay the tuple and its order of use, but cannot turn a ledger entry naming host conversion, quotient equality, choice, propositional extensionality, or an ambient classical theorem into exported BEDC proof supply. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleConsumerReadiness`

Theorem 861.22 (AxiomDependencyTuple audit-map row coverage)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$*, every downstream audit-map read of* A *is covered by exactly the displayed mode row, witness row, supply ledger row, componentwise transport row, continuation route row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *row. No audit-map row outside* M, W, S, H, C, P, N *is required to replay the tuple.*

Proof. Start with the carrier definition and the local NameCert obligations. By Definition 861.1 and Definition 861.2, the admitted packet has exactly the displayed carrier, classifier, witness-transport, required-supply, route-readback, provenance, and naming obligations.

The mode and ledger coverage are then fixed by Definition 861.3 and Definition 861.4. These results force every consumer read to begin with the three mode cases and the visible required-supply row S.

The remaining public rows are stable under the routing discipline. Apply Definition 861.8, Definition 861.19, and Definition 861.21: continuation reads only replay displayed coordinates, the root-unblock spine names no extra row, and the consumer package exports no host or classical supply.

Finally, the primitive row scope is exhausted by Definition 861.16, Definition 861.17, and Definition 861.18. Thus every downstream audit-map read is covered by the seven listed rows and by no ambient interface. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleAuditMapRowCoverage`

Theorem 861.23 (AxiomDependencyTuple dependency-map soundness)**.** *Every downstream dependency-map read of an accepted* AxiomDependencyTuple$^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$ *preserves the displayed mode discipline. Mode A reads an empty* S *before the zero-supply witness, Mode B exposes* S *before the restricted witness* W*, and Mode C reads the counterexample or socket as refusal rather than as promoted host supply.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleDependencyMapSoundness`

Proof. Split by the mode coordinate. The split is exhaustive by Definition 861.3, and its consumer order is fixed by Definition 861.20.

In Mode A, use Definition 861.13. The ledger S is empty of external supply and is read separately before the internal zero-supply witness in W.

In Mode B, use Definition 861.14. The restricted witness cannot be consumed until the full displayed supply ledger S has been exposed, so no subledger or implicit host rule replaces it.

In Mode C, use Definition 861.15. The counterexample or socket names refused supply through W and S, while C, P, N prevent that named supply from becoming an exported BEDC proof rule.

The consumer boundary closes the dependency-map claim. By Definition 861.4, Definition 861.8, and Definition 861.21, all three reads keep ledger exposure ahead of witness consumption and refuse every promotion of host conversion, quotient equality, choice, propositional extensionality, or ambient classical supply. □

Theorem 861.24 (AxiomDependencyTuple public interface exactness)**.** *The public interface exported by* AxiomDependencyTuple$^{\uparrow}$ *to audit-map consumers is exactly the carrier of Definition 861.1 together with the row coverage of Definition 861.22 and the dependency-map soundness of Definition 861.23. It exports no* Quot.sound*,* Classical.choice*,* propext*, host conversion, or ambient classical theorem.*

Proof. The carrier supplies the public data. Its coordinates are precisely M, W, S, H, C, P, N, and Definition 861.2 identifies the local NameCert obligations attached to those coordinates.

The two consumer-facing surfaces are the row coverage and soundness results. The first says that audit-map consumers read only the displayed seven rows; the second says that dependency-map consumers preserve the mode-ledger discipline in Modes A, B, and C.

It remains to exclude extra public exports. Apply Definition 861.8, Definition 861.19, and Definition 861.21. Routes replay the carrier, the spine has no additional root-unblock datum, and the consumer package refuses every host conversion or classical theorem as BEDC supply.

Thus the public interface is the carrier plus these two audit surfaces, with no quotient equality, choice, propositional extensionality, host conversion, or ambient classical theorem exported alongside it. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTuplePublicInterfaceExactness`

Theorem 861.25 (AxiomDependencyTuple kernel supply visibility)**.** *For every accepted* AxiomDependencyTuple$^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$*, every external assumption read by the kernel side of the tuple is exposed first as a row of the displayed required-supply ledger* S*. In particular, host conversion,*

quotient equality, Classical.choice*,* propext*, ambient classical theorem names, and any other non-BEDC supply name can be consumed only through the ledger row selected by M and attached to the witness row W.*

Proof. Start with Definition 861.1. The carrier displays the mode row M, witness row W, required-supply ledger S, transport row H, continuation row C, provenance row P, and local NameCert row N as separate finite coordinates.

Apply Definition 861.2. The local certificate exposes carrier admission, mode classification, witness transport, ledger exactness, displayed continuation readback, provenance, and naming, so the kernel-side reader has no hidden coordinate from which to obtain an external assumption.

The mode split fixes how the ledger is read. By Definition 861.3, Mode A has no external supply, Mode B consumes a restricted witness under the displayed ledger, and Mode C names refused supply through a counterexample or socket. By Definition 861.4, each named host or classical item therefore remains visible in S before W is consumed. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleKernelSupplyVisibility`

Theorem 861.26 (AxiomDependencyTuple supply provenance sibling route)**.** *For every accepted* AxiomDependencyTuple$^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$, each required-supply ledger row that is consumed as a kernel supply witness is first read through the sibling* SubjectReductionDischargeSocket$^{\uparrow}$ *route of chapter 719. Only after that route has exposed the relevant discharge, obstruction, and provenance rows may the selected entry of S be consumed by W as a kernel-facing supply witness.*

Proof. Project the tuple fields from Definition 861.1. The carrier exposes the mode row M, witness row W, required-supply ledger S, transport row H, continuation row C, provenance row P, and local naming row N; it has no hidden coordinate for external supply.

Apply Definition 861.25. Any kernel-side external assumption is visible first in S and is selected by M before W can consume it.

The sibling socket supplies the provenance route named by chapter 719. Thus the ledger row is not a kernel-only island: it is read through the displayed discharge-socket surface, then transported and replayed by H and C, sourced by P, and named by N before it reaches the witness row. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleSupplyProvenanceSiblingRoute`

Theorem 861.27 (AxiomDependencyTuple external supply refusal)**.** *For every accepted* AxiomDependencyTuple$^{\uparrow}$ *carrier $A = (M, W, S, H, C, P, N)$, the names for host conversion, quotient equality,* Classical.choice*,* propext*, and ambient classical theorem supply are refused as exported BEDC proof data. They may appear as entries of S or as refused supply named by W in the appropriate mode, but H, C, P, and N can only transport, replay, record provenance for, and name the displayed tuple rows.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleExternalSupplyRefusal`

Proof. Use the carrier and NameCert obligations first. By Definition 861.1 and Definition 861.2, external supply names occur only in the required-supply ledger and in the witness reading selected by the mode row.

The three mode cases separate the possible readings. Mode A is governed by Definition 861.13, so an empty external ledger cannot be converted into host or classical proof data. Mode B is governed by Definition 861.14, so a restricted witness is consumed only with the full displayed ledger. Mode C is governed by Definition 861.15, so a counterexample or socket names missing supply without discharging it.

Finally apply Definition 861.4, Definition 861.8, and Definition 861.33. Ledger readback keeps the names visible, route non-export prevents continuation replay from creating proof rules, and host-supply nonescape keeps every such name inside the audit ledger surface. □

Theorem 861.28 (AxiomDependencyTuple audit consumer completeness)**.** *Every dependency-map consumer for an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$ *must read the complete audit surface: mode* M*, witness* W*, required-supply ledger* S*, transport row* H*, continuation route* C*, provenance row* P*, and local NameCert row* N*. A consumer that omits any one of these rows is not a consumer of the displayed dependency tuple.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleAuditConsumerCompleteness`

Proof. Begin with the finite packet of Definition 861.1. Its public surface is exactly M, W, S, H, C, P, N, so the consumer question is a row-coverage question rather than a request for an ambient proof principle.

The first three rows are forced by the obligation and mode theorems. Apply Definition 861.2, Definition 861.3, and Definition 861.4: the consumer must know which mode is active, which witness is being read, and which ledger row authorizes or refuses that witness.

The remaining rows are forced by the transport and routing surface. The existing per-mode separation theorems keep the three witness readings disjoint, while Definition 861.8 and Definition 861.9 confine every consumer to the displayed packet rows. Hence a complete audit consumer must pass through H, C, P, and N as well as M, W, and S. □

Corollary 861.29 (AxiomDependencyTuple kernel supply visibility strict obstruction)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$*, any displayed failure, refusal, separation, or counterexample-or-socket row used in the kernel-supply visibility argument has strictly local force. It blocks only the named converse or reuse route that tries to read external supply outside the required-supply ledger* S*, erase a listed supply entry, reread a mode as another mode, or export refused supply as BEDC proof data. It cannot be reinterpreted as a global impossibility for* $\mathsf{AxiomDependencyTuple}^{\uparrow}$*, as a host theorem, or as hidden external validation. Every consumer-facing use of the obstruction factors through one of the displayed rows* M, W, S, H, C, P, N *and rejects any unlisted coordinate.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleKernelSupplyVisibilityStrictObstruction`

Proof. Start with Definition 861.25. Any external assumption read by the kernel side is visible first as a row of the required-supply ledger S, selected by the mode row M and attached to the witness row W. Thus the obstruction has a named input surface before any consumer can use it.

Now apply Definition 861.27. Host conversion, quotient equality, Classical.choice, propext, and ambient classical theorem supply may be named in S or by the refused witness reading, but the rows H, C, P, and N can only transport, replay, record provenance for, and name the displayed tuple rows. Hence a consumer-facing obstruction has the listed route, transport, provenance, and NameCert coordinates as its whole interface.

The three mode rows give the strict local cases. In Mode A, Definition 861.13 says that the empty external ledger remains separate from the zero-supply witness, so the blocked route is only the attempted conversion of emptiness into host or classical proof data. In Mode B, Definition 861.14 says that the restricted witness is consumed only with the full displayed ledger, so the blocked route is only subledger erasure, ledger shrinking, or replacement by an implicit host rule. In Mode C, Definition 861.15 says that a counterexample or socket names refused supply

without discharging it, so the blocked route is only promotion of that refusal into an exported BEDC rule.

It remains to exclude broader readings. By Definition 861.4, Definition 861.8, and Definition 861.33, the obstruction is read back through the visible ledger, replayed only by displayed routes, and kept inside the audit surface under transport. By Definition 861.28, any consumer that omits one of M, W, S, H, C, P, N is not a consumer of the displayed dependency tuple. Therefore the row blocks only the named local converse or reuse route, and no unlisted coordinate can turn it into a global impossibility, host theorem, or hidden external validation. □

Theorem 861.30 (AxiomDependencyTuple subject-reduction socket dependency route)**.** *Every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *supply-ledger read that names missing MetaCIC supply must first consume the concrete subject-reduction discharge socket of chapter 719. The tuple may then record the selected host conversion, quotient equality,* Classical.choice, propext*, ambient theorem, or MetaCIC supply name as a row of* S*, but no consumer may inspect that row as kernel supply until the socket route has exposed whether the supply is discharged, restricted, or refused.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleSubjectReductionSocketDependencyRoute`

Proof. Project the carrier $A = (M, W, S, H, C, P, N)$ from Definition 861.1. The only row that can name external or MetaCIC supply is the required-supply ledger S, and the witness row W is consumed only after the mode row M has selected the relevant reading.

Apply Definition 861.4 and Definition 861.25. These results force the ledger row to be visible before the witness can use it, and they prevent any hidden kernel-side coordinate from supplying the same material.

The concrete sibling route is chapter 719. Reading that socket before S records whether the named subject-reduction supply is discharged, restricted, or refused. The structural rows H, C, P, N then transport, replay, source, and name the same ledger entry without promoting it into an exported BEDC rule. □

Discharge-socket dependency. The three supply modes are read against the concrete subject-reduction discharge socket of chapter 719. Zero supply corresponds to obligations discharged internally by the displayed BEDC rows, restricted supply corresponds to displayed discharge hypotheses consumed from the ledger, and socket or refusal mode records missing supply rather than exporting it as a proof rule.

Theorem 861.31 (AxiomDependencyTuple supply-mode disjointness)**.** *For every accepted* AxiomDependencyTuple *carrier* $A = (M, W, S, H, C, P, N)$*, the displayed mode row* M *keeps the three supply modes pairwise disjoint. Mode A cannot be reread as Mode B or Mode C; Mode B cannot be reread as Mode A or Mode C; and Mode C cannot be reread as Mode A or Mode B under componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, or local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *transport.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleSupplyModeDisjointness`

Proof. Start with the mode row M. By Definition 861.3, M has exactly three visible readings: zero-supply proof, restricted proof with required supply, and counterexample-or-socket refusal. The row has no fourth constructor and no ambient equality principle that can identify these readings.

In Mode A, Definition 861.13 keeps the proof witness separate from an empty external ledger. This cannot be read as Mode B, because Definition 861.14 requires a full displayed supply row for every restricted witness. It cannot be read as Mode C, because Definition 861.15 requires a refusal witness naming supply that is not discharged.

In Mode B, Definition 861.11 keeps each named host or classical supply entry in S. The branch therefore cannot collapse to the zero-supply branch, and it cannot become the socket branch because the witness is consumed under a ledger rather than recorded as a counterexample.

In Mode C, Definition 861.12 and Definition 861.15 keep the witness as refusal data. Transport and replay preserve this separation by Definition 861.17 and Definition 861.18, so no route, provenance row, or local naming row can reread one mode as another. □

Theorem 861.32 (AxiomDependencyTuple consumer ledger order)**.** *Every consumer route for an accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$ *reads the mode row* M *and the required-supply ledger* S *before consuming the witness row* W*. This read order is preserved by the displayed* Cont *route row, by componentwise* hsame *transport, by the provenance row* P*, and by the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleConsumerLedgerOrder`

Proof. First read the carrier coordinates. The public packet exposes M, W, S, H, C, P, and N as distinct rows, so a consumer route has no hidden coordinate from which to recover a witness before inspecting the mode and ledger.

The mode row is read first because Definition 861.3 determines which of the three witness readings is even available. The separation result Definition 861.31 then prevents a consumer from choosing a witness interpretation while leaving the mode undetermined.

The ledger row is read before witness consumption by Definition 861.4. In Mode A this records the empty external ledger, in Mode B it records every required supply entry, and in Mode C it records the refused supply named by the socket or counterexample.

Finally apply Definition 861.8, Definition 861.17, and Definition 861.18. Routes, transport, provenance, and local naming replay the displayed rows; they do not add a consumer path that reads W before M and S. □

Theorem 861.33 (AxiomDependencyTuple host-supply nonescape)**.** *For every accepted* $\mathsf{AxiomDependencyTuple}^{\uparrow}$ *carrier* $A = (M, W, S, H, C, P, N)$*, any named host conversion, quotient equality,* Classical.choice *use,* propext *use, or ambient theorem remains a required-supply ledger row under* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$ *transport. None of these names escapes the ledger as an exported BEDC proof rule.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyTupleUp.AxiomDependencyTupleHostSupplyNonescape`

Proof. The supply row S is the only place where external host or classical material can be named. By Definition 861.4, any consumer that reads the witness must first read that ledger entry as audit supply.

Mode B is the positive ledger case. Use Definition 861.11 and Definition 861.14: every named supply entry stays in the full displayed ledger, and no consumer may erase, shrink, or internalize it.

Mode C is the refusal case. By Definition 861.12 and Definition 861.15, a counterexample or socket names missing supply without discharging it. Thus the named host or classical datum remains a ledger row attached to refused supply.

The transport rows do not change the status of the name. Apply Definition 861.8, Definition 861.17, and Definition 861.18. Componentwise hsame, displayed Cont, Pkg provenance, and local NameCert transport replay the finite tuple surface and never export host conversion, quotient equality, choice, propositional extensionality, or ambient theorem supply as a BEDC rule. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomDependencyTuple}^{\uparrow}$ packet is generated from a three-mode row, a proof, restricted-proof, counterexample, or socket witness row, a required-supply ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 861.1, Definition 861.2, Definition 861.3, Definition 861.4, Definition 861.8, Definition 861.9, Definition 861.10, Definition 861.11, Definition 861.12, Definition 861.13, Definition 861.14, Definition 861.15, Definition 861.16, Definition 861.17, Definition 861.18, Definition 861.19, Definition 861.20, Definition 861.21, Definition 861.22, Definition 861.23, Definition 861.24, Definition 861.31, Definition 861.32, and Definition 861.33 bind the finite dependency-tuple carrier, mode exhaustion, ledger exactness, route non-export, consumer exhaustion, the three per-mode obligation rows, the three row-level separation rows, the BHist, hsame, Cont, Pkg, and NameCert primitive scope, the root-unblock spine, the consumer read order, the exact public audit-map interface, the supply-mode disjointness row, the consumer ledger order row, and the host-supply nonescape row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxiomDependencyTuple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not enumerate all classical CIC claims, internalize host conversion, quotient equality, choice, propositional extensionality, or ambient classical CIC theorem supply, give Lean verification, or bridge check the tuple.

Upgrade path: Public closure next requires Lean verification of the public consumer interface and a bridge check for the scoped tuple.

Chapter 862

A Concrete Naming Certificate for $\mathsf{AxiomDependencyMap}^{\uparrow}$

The $\mathsf{AxiomDependencyMap}^{\uparrow}$ packet turns the dependency-map reading of chapter 9455 into a concrete finite certificate. It sits beside the axiom-query ledger $\mathsf{AxiomQueryLedger}^{\uparrow}$ of chapter 654, the family carrier $\mathsf{AuditMapFamily}^{\uparrow}$ of chapter 807, the compiler-facing map $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ of chapter 836, and the cross-map weave $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ of chapter 839. Those sibling chapters expose ledger, family, compiler, and dependency-weave rows; this chapter packages the claim-level dependency tuple that a downstream audit-map consumer reads before deciding whether the claim is internal, restricted, or blocked by named supply.

Definition 862.1 (Axiom-dependency-map packet)**.** *An* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$X = (\Phi, M, W, A, H, C, P, N)$$

with the following displayed rows:

1. Φ *is the claim row naming the classical-looking assertion being inspected against the BEDC substrate;*
2. M *is the mode row, with one of the three displayed readings internal proof, restricted internal proof, or obstruction with named supply;*
3. W *is the witness row, carrying the proof term, restricted proof surface, counterexample, refusal, or named socket that justifies the mode;*
4. A *is the required-axiom-set row, listing the empty supply, restricted-surface condition, or named external supply rows required by the mode;*
5. H *records componentwise* hsame *transport for the claim, mode, witness, required-axiom-set, continuation, provenance, and local naming rows;*
6. C *records the displayed* Cont *replay routes by which a consumer replays the claim read, mode classification, witness readback, and required-supply readback;*
7. P *is the* Pkg *provenance over the dependency-map vision,* $\mathsf{AxiomQueryLedger}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, $\mathsf{AuditMapDependencyWeave}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$ *source packets;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyMap}^{\uparrow}}$ *row.*

The packet exports no ambient theorem count, no host proof oracle, no hidden axiom supply, and no permission to treat a blocked or restricted row as an unconditional BEDC theorem.

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapTasteGate single carrier alignment`

Theorem 862.2 (Axiom-dependency-map obligations)**.** *For every* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AxiomDependencyMap}^{\uparrow}}$ *surface consists exactly of claim exposure, mode exposure, witness exposure, required-axiom-set exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Any consumer of* N *must read the dependency tuple through these eight rows and cannot replace them by prose comparison with classical CIC.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapLedgerObligations`

Proof. Project the finite packet of Definition 862.1. The rows Φ, M, W, A expose the claim, the mode assigned to it, the displayed witness for that mode, and the required-supply row.

The rows H and C bind transport and replay. A consumer may move the packet componentwise and replay its reads, but the transport and replay endpoints remain the displayed claim, mode, witness, and required-axiom-set rows.

The provenance and naming rows P, N package the same data as the local NameCert surface. Since the carrier has no further coordinate, there is no route from N to an ambient theorem count, a host proof oracle, or a hidden supply row. □

Theorem 862.3 (Axiom-dependency-map mode exhaustion)**.** *For every accepted* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

the mode row M *is exhausted by the three displayed cases: internal proof, restricted internal proof, and obstruction with named supply. In the first case* A *records empty supply and* W *records the internal witness; in the second case* A *records the restricted surface and* W *records the restricted witness; in the third case* A *records the named supply and* W *records the obstruction, refusal, counterexample, or socket row.*

Proof. Read the mode row M of the packet. By definition the carrier admits only the three displayed readings, so the case split has exactly those branches.

In the internal-proof branch, the claim row Φ is paired with a witness row W that supplies the substrate proof and a required-axiom-set row A that records empty supply.

In the restricted-proof branch, the same projection reads W as the restricted witness and A as the restricted-surface condition. The carrier contains no row that erases the restriction.

In the obstruction branch, the witness row records the counterexample, refusal, or socket and A records the named supply. The packet therefore classifies blocked content as visible dependency data rather than as an unconditional theorem. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapModeSoundness`

Theorem 862.4 (Axiom-dependency-map query-ledger factorization)**.** *Every downstream audit read that treats an* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet as proof-ancestry data factors through* $\mathsf{AxiomQueryLedger}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, *or* $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ *as named in* P*. In particular, a consumer cannot read the required-axiom-set row* A *as a strict ancestry verdict unless the replay route in* C *reaches the corresponding finite ledger, family, compiler, or weave row cited by the provenance package.*

Proof. Start with the provenance row P. It cites the dependency-map vision, $\mathsf{AxiomQueryLedger}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, and $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ as the only package sources for query, family, compiler, and dependency-weave reads.

Next read the continuation row C. A proof-ancestry consumer must replay from Φ through the mode and witness rows to the required-axiom-set row A, then through the ledger, family, compiler, or weave row named by P.

If such a ledger or gate route is absent, the packet still records a claim-level dependency tuple, but not a strict ancestry verdict. The factorization therefore separates dependency-map classification from the finite query ledger that certifies a particular accepted declaration. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapQueryLedgerFactorization`
Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapCertificate query ledger factorization`

Theorem 862.5 (Axiom-dependency-map non-escape)**.** *For every accepted* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

every public read remains inside the displayed claim, mode, witness, required-axiom-set, hsame*,* Cont*,* Pkg*, and* NameCert *rows. The packet cannot turn a restricted row into an unrestricted theorem, a named supply row into internal BEDC content, an obstruction row into a proof, or a claim-level dependency tuple into a hidden host axiom.*

Proof. Project the accepted packet as $X = (\Phi, M, W, A, H, C, P, N)$. These are the only rows exported by the carrier.

Mode soundness from Definition 862.3 fixes how M, W, A may be read. The internal, restricted, and obstruction branches each keep their own witness and required-supply interpretation.

The factorization result of Definition 862.4 binds proof-ancestry reads to the cited ledger, family, compiler, or weave packet. Thus query meaning cannot appear from the local name alone.

Since no further coordinate exists, a consumer has no place from which to obtain unrestricted closure, hidden supply, obstruction discharge, or host-axiom admission. All public reads remain inside the finite packet. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapCertificate public boundary`

Theorem 862.6 (Axiom-dependency-map required-supply separation)**.** *For every accepted* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

the required-axiom-set row A separates the three displayed supply modes: empty supply for internal proof, restricted-surface conditions for restricted internal proof, and named external supply for obstruction or socket rows. No downstream consumer may read one supply mode as another through H, C, P, N.

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapRequiredSupplySeparation`

Proof. Apply mode soundness from Definition 862.3. The three cases for M pair W and A with empty supply, restricted surface, or named supply respectively.

Next apply the query-ledger factorization theorem Definition 862.4. Any consumer that treats A as proof-ancestry data must replay the corresponding ledger, family, compiler, or weave route named in P and routed by C.

Finally use the boundary theorem Definition 862.5. The transport, continuation, provenance, and naming rows preserve the displayed tuple only. Since the packet contains no row that converts one supply mode into another, no consumer can read empty supply, restricted surface, and named external supply interchangeably. □

Theorem 862.7 (Axiom-dependency-map restricted-row non-escape)**.** *For every accepted* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

if the mode row M reads as restricted internal proof, then the witness row W and required-supply row A remain restricted under H, C, P, N. They cannot be exported as unconditional BEDC theorems, and the local $\mathsf{NameCert}_{\mathsf{AxiomDependencyMap}^{\uparrow}}$ *row exposes only the restricted dependency tuple.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyMapUp.AxiomDependencyMapRestrictedRowNonescape`

Proof. Start with mode soundness. In the restricted branch, W is a restricted witness and A is the restricted-surface condition; the carrier records no row that erases that condition.

Use query-ledger factorization to inspect any downstream replay. A consumer must pass through the finite ledger, family, compiler, or dependency-weave route named by P, so the restricted branch remains tied to the ledger that justifies the restricted reading.

Apply the non-escape boundary. The rows H, C, P, N transport, replay, package, and name the same displayed tuple, and the carrier has no coordinate for unrestricted theorem export. Therefore a restricted proof row remains restricted and cannot be read as an unconditional BEDC theorem. □

Theorem 862.8 (Axiom-dependency-map mature query-supply example)**.** *For every downstream audit consumer reading an accepted* $\mathsf{AxiomDependencyMap}^{\uparrow}$ *packet*

$$X = (\Phi, M, W, A, H, C, P, N),$$

the public query-supply read first projects the mode row M, then the witness row W and required-supply row A, then factors through the cited $\mathsf{AxiomQueryLedger}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, *or* $\mathsf{AuditMapDependencyWeave}^{\uparrow}$ *route before reporting internal, restricted, or named-supply status.*

Proof. Start with the accepted packet and apply Definition 862.3. The consumer reads M as one of the internal, restricted, or obstruction modes, and this fixes the allowed interpretation of W and A.

Next apply Definition 862.4. A query-supply report that treats A as proof-ancestry data must pass through the finite ledger, family, compiler, or dependency-weave route cited by P and replayed by C.

Use Definition 862.5 to keep the public read inside the displayed claim, mode, witness, supply, transport, replay, provenance, and local naming rows. Finally apply Definition 862.6: the empty, restricted, and named-supply rows remain distinct, so the reported status is exactly internal, restricted, or named-supply according to the projected mode. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomDependencyMap}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AxiomDependencyMap}^{\uparrow}$ packet is generated from a claim row, a mode row, a witness row, a required-axiom-set row, componentwise hsame transports, displayed Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxiomDependencyMap}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 862.1, Definition 862.2, Definition 862.3, Definition 862.4, Definition 862.5, Definition 862.6, Definition 862.7, and Definition 862.8 specify the finite claim-level dependency-map packet, query-ledger

factorization, required-supply separation, restricted-row non-escape, public boundary, and mature query-supply consumer surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{AxiomDependencyMap}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not enumerate all classical CIC claims, prove global completeness of the dependency map, certify any hidden host proof oracle, close Lean verification, or bridge the packet to an external schema.

Upgrade path: Bridged closure requires a standard bridge over the same claim, mode, witness, required-supply, transport, replay, provenance, and local naming rows.

Chapter 863

A Concrete Naming Certificate for $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$

The $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ packet realizes the map of chapter 9455 as a finite ledger row that can be consumed by audit-map chapters. It sits beside the audit gate of chapter 526, the strict axiom-query ledger of chapter 654, and the substitution audit map of chapter 779. Those siblings expose boundary, query, and concrete audit-map rows; this chapter records the Mode, Witness, and Required-Axiom-Set tuple as one named packet rather than as framing prose.

Definition 863.1 (AxiomDependencyMapLedger carrier)**.** *An* $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$L = (\Phi, M, W, A, H, C, P, N)$$

with the following displayed rows:

1. Φ *is the claim row whose classical-looking assertion is being mapped against the BEDC substrate;*
2. M *is the mode row, carrying exactly Mode A, Mode B, or Mode C;*
3. W *is the witness row, carrying the proof, restricted proof, counterexample, refusal, or socket data appropriate to* M*;*
4. A *is the Required-Axiom-Set row, carrying empty supply, restricted-surface supply, or the named supply rows required by the witness;*
5. H *records componentwise* hsame *transport for the claim, mode, witness, required-supply, continuation, provenance, and local naming rows;*
6. C *records the displayed* Cont *routes by which a consumer replays the claim read, mode read, witness readback, and required-supply readback;*
7. P *is the* Pkg *provenance over chapter 9455, chapter 526, chapter 654, chapter 779,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyMapLedger}^{\uparrow}}$ *row.*

The classifier accepts the ledger only through this finite packet. It contains no ambient proof oracle, no hidden host axiom supply, no unrecorded query rerun, and no permission to treat a restricted or obstructed row as an unconditional BEDC theorem.

Theorem 863.2 (AxiomDependencyMapLedger surface)**.** *For every accepted* $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ *carrier*

$$L = (\Phi, M, W, A, H, C, P, N),$$

the public $\mathsf{NameCert}_{\mathsf{AxiomDependencyMapLedger}^{\uparrow}}$ *surface is exactly the claim row, mode row, witness row, Required-Axiom-Set row, componentwise transport row, replay route row, provenance row, and local naming row. Every downstream audit-map read must factor through these eight rows and through the sibling boundary, query, or audit-map packets named in* P*.*

Proof. Project the carrier of Definition 863.1. The rows Φ, M, W, A expose the mapped claim, its mode, its witness, and its required-supply entry.

Read H and C next. Transport and replay preserve the same four visible map rows and do not create a second route to host proof ancestry.

Finally inspect P and N. Provenance points to chapter 9455, chapter 526, chapter 654, and chapter 779; the local name only packages this finite surface. Therefore every public read factors through the displayed ledger rows. □

Theorem 863.3 (AxiomDependencyMapLedger Mode A)**.** *If an accepted* $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ *carrier has Mode A in* M*, then* W *is read as an unconditional substrate proof witness for* Φ *and* A *is read as empty required supply. The rows* H, C, P, N *may transport, replay, package, and name that reading, but they cannot add a hidden dependency row to the Mode A entry.*

Proof. Read the mode row M from Definition 863.1. In the Mode A branch, the carrier definition fixes the witness row as an internal proof and fixes the Required-Axiom-Set row as empty supply.

Use Definition 863.2 to route every consumer through the same rows. The support rows H, C, P, N preserve and expose the entry but do not add another source of proof supply.

Thus Mode A remains a zero-supply ledger entry for the displayed claim, not a claim about unrecorded host proof ancestry. □

Theorem 863.4 (AxiomDependencyMapLedger Mode B)**.** *If an accepted* $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ *carrier has Mode B in* M*, then* W *is read only as a restricted-proof witness and* A *is read as the restricted-surface condition required for that witness. A consumer may not erase the restriction, replace it by an unrestricted proof, or read the restricted surface without the replay route carried by* C*.*

Proof. Project M, W, A from the accepted carrier. The Mode B branch pairs the witness row with the restricted-surface row, so the witness is not exposed independently of that surface.

Replay the entry through C and transport it through H. The same restricted row remains attached to the witness after movement, while P and N package the entry as the local ledger surface.

Because no carrier coordinate contains an unrestricted proof of Φ, no downstream consumer can erase the restriction or convert the Mode B row into Mode A. □

Theorem 863.5 (AxiomDependencyMapLedger Mode C)**.** *If an accepted* $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ *carrier has Mode C in* M*, then* W *is read as a counterexample, refusal, obstruction, or socket witness and* A *is read as the named required-supply row. The carrier records the dependency boundary; it does not discharge the named supply or turn the obstruction into a substrate theorem.*

Proof. Read the Mode C branch of the carrier. The rows W and A are paired as obstruction data and named supply, not as a proof and empty dependency set.

Use the public surface theorem to keep the read inside $L = (\Phi, M, W, A, H, C, P, N)$. The route row C can replay the obstruction and the supply name, and the transport row H preserves those coordinates componentwise.

The provenance row points to the boundary, query, and audit-map siblings, where such supply must be inspected. Since the local ledger has no coordinate for discharging the named supply, Mode C remains a dependency record and not an internal BEDC proof. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomDependencyMapLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomDependencyMapLedger}^{\uparrow}$ packet is generated from a claim row, a three-mode row, a witness row, a Required-Axiom-Set row, componentwise hsame transport, displayed Cont replay routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxiomDependencyMapLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 863.1, Definition 863.2, Definition 863.3, Definition 863.4, and Definition 863.5 specify the finite Mode, Witness, Required-Axiom-Set ledger packet and its three mode readings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxiomDependencyMapLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not enumerate all classical CIC claims, prove global dependency-map completeness, discharge named host supply, provide Lean verification, or bridge the ledger to an external schema.

Upgrade path: Obligation closure requires checked carrier admission, three-mode classification, witness exactness, Required-Axiom-Set exactness, replay locality, provenance, local naming, and non-escape for the same finite ledger packet.

Chapter 864

A Concrete Naming Certificate for $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$

The $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$ packet isolates the Mode-B row of chapter 9455 as reusable finite NameCert data. It sits beside $\mathsf{AxiomDependencyTuple}^{\uparrow}$ in chapter 861, $\mathsf{AxiomDependencyMap}^{\uparrow}$ in chapter 862, $\mathsf{AuditMapFamily}^{\uparrow}$ in chapter 817, and $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$ in chapter 622. Those siblings package full dependency tuples, map families, and refusal verdicts; this chapter packages the restricted surface itself as the consumer-stable budget row that keeps a restricted proof from escaping as an unconditional BEDC theorem.

Definition 864.1 (Restricted-surface budget carrier)**.** *A* $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (\Phi, R, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. Φ *is the claim row naming the classical-looking assertion whose Mode-B surface is being recorded;*
2. R *is the restricted-surface row, listing the syntactic, closedness, classifier, or displayed-domain condition under which the internal proof is valid;*
3. W *is the restricted witness row, carrying the internal proof term only as a witness under* R*;*
4. S *is the required-supply row, recording that no extra BEDC supply is used on* R *while the unrestricted claim still retains its stated boundary;*
5. E *is the non-export row, forbidding a consumer from reading* W *as an unconditional proof of* Φ*;*
6. H *records componentwise* hsame *transport for the claim, restricted surface, witness, required-supply, non-export, replay, provenance, and local naming rows;*
7. C *records the displayed* Cont *replay routes by which a consumer reads* Φ*, verifies* R*, replays* W *under* R*, reads* S*, and checks* E*;*
8. P *is the* Pkg *provenance over the dependency-map vision,* $\mathsf{AxiomDependencyTuple}^{\uparrow}$*,* $\mathsf{AxiomDependencyMap}^{\uparrow}$*,* $\mathsf{AuditMapFamily}^{\uparrow}$*,* $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RestrictedSurfaceBudget}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose witness is budgeted by the displayed restricted surface and whose public export is bounded by the non-export row.

Lean4 ✓ `BEDC.Derived.RestrictedSurfaceBudgetUp.RestrictedSurfaceBudgetTasteGate single carrier alignment`

Theorem 864.2 (Restricted-surface budget NameCert obligations)**.** *For every* $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$ *packet*

$$B = (\Phi, R, W, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RestrictedSurfaceBudget}^{\uparrow}}$ *surface consists exactly of claim exposure, restricted-surface exposure, restricted-witness exposure, required-supply exposure, non-export exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer of* N *may cite the Mode-B witness only together with* R, S, E, H, C, P *and cannot cite it as a BEDC-internal proof of the unrestricted claim row* Φ*.*

Proof. Project the packet of Definition 864.1. The rows Φ, R, W, S, E are the complete payload: a claim, the surface on which it is discharged, the witness on that surface, the required-supply row, and the non-export boundary.

Read H and C next. Transport can move the displayed rows only componentwise, and replay must first inspect R before it reads W as a witness for the claim.

Finally inspect P and N. Provenance cites the sibling dependency-map, tuple, family, and forbidden-ancestry chapters, while the local name binds the same finite packet. Since no coordinate stores an unrestricted theorem export, the certificate obligations are exhausted by the displayed rows. □

Theorem 864.3 (Restricted-surface budget Mode-B non-escape)**.** *For every accepted* $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$ *packet*

$$B = (\Phi, R, W, S, E, H, C, P, N),$$

the restricted witness row W *remains a Mode-B witness under the surface row* R*. Transport, replay, provenance, and local naming may cite* W *only through the same restricted-surface budget, and no consumer may erase* R*, ignore* E*, or re-export* W *as an unconditional BEDC theorem for* Φ*.*

Proof. Begin with the carrier projection. The witness row W is paired with R and with the non-export row E in the same finite packet.

Replay then fixes the order of use. A consumer route in C reads the claim row, checks the restricted surface, reads the witness under that surface, and then reads the required-supply and non-export rows. The replay route contains no branch that bypasses R or E.

Transport and provenance preserve the same boundary. The row H moves only displayed coordinates, P cites the sibling dependency packets as sources, and N names the local carrier. Therefore the witness can be cited as Mode-B budget data, but not as an unrestricted BEDC theorem. □

Falsifiable prediction: If the restricted-surface row is deleted, no consumer may cite the Mode-B witness as a BEDC-internal restricted proof.

Independence witness: axiomdependencymap, axiom-dependency-tuple, auditmapfamily, forbiddenaxiomancestry: the first two package full dependency tuples, the family chapter packages map collections, and forbidden ancestry packages refusal verdicts; none isolates the restricted-surface budget row as a consumer-stable carrier.

The Lean side must provide Nontrivial and FieldFaithful instances for the same carrier in `lean4/BEDC/Derived/RestrictedSurfaceBudgetUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{RestrictedSurfaceBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RestrictedSurfaceBudget}^{\uparrow}$ packet is generated from a claim row, restricted-surface row, restricted-witness row, required-supply row, non-export row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RestrictedSurfaceBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 864.1, Definition 864.2, and Definition 864.3 specify the reusable Mode-B restricted-surface budget, NameCert obligations, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RestrictedSurfaceBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove the unrestricted claim row, enumerate every Mode-B surface, close Lean verification, or bridge the budget to an external schema.

Upgrade path: Obligation closure requires checked carrier admission, restricted-surface exactness, witness replay under the surface, required-supply exactness, non-export, and TasteGate instances for the same packet.

Chapter 865

A Concrete Naming Certificate for $\mathsf{InternalTheoryBoundary}^{\uparrow}$

The $\mathsf{InternalTheoryBoundary}^{\uparrow}$ packet turns the own-theory reading of chapter 9454 into a concrete finite certificate. It sits beside the kernel-acceptance packet of chapter 549, the forbidden-host-axiom ancestry boundary of chapter 622, and the proposition-transport boundary of chapter 566. Those sibling chapters expose accepted-declaration rows, strict proof-ancestry rows, and directed proposition-transport rows; this chapter packages the positive BEDC.MetaCIC commitments that make those rows internal rather than borrowed from a comparison calculus.

Definition 865.1 (InternalTheoryBoundary carrier)**.** *An* $\mathsf{InternalTheoryBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, E, R, Z, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the closed-generator row, naming the finite constructors, binder forms, typed example rows, and accepted declaration rows from which BEDC.MetaCIC objects are generated;*
2. *E is the literal AST equality row, recording syntactic equality, constructor tag equality, and componentwise binder equality as displayed kernel data rather than as host quotient equality;*
3. *R is the structural recursor row, exposing only the recursors attached to the closed generators in G and the continuation routes that consume their recursive branches;*
4. *Z is the strict-purity row, recording the empty source-axiom supply, accepted kernel row, and forbidden-host-axiom refusal surface required before a theorem is read as BEDC-internal;*
5. *A is the apophatic boundary row, recording that comparison with classical CIC is secondary analysis and not a carrier coordinate, definition source, or external equality oracle for BEDC.MetaCIC;*
6. *H records componentwise* hsame *transport for the generator, AST equality, recursor, purity, boundary, continuation, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which a consumer reads generators, literal AST comparison, structural recursion, strict-purity audit, and the apophatic boundary;*
8. *P is the* Pkg *provenance over BEDC.MetaCIC,* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$, $\mathsf{PropextTransportBoundary}^{\uparrow}$, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InternalTheoryBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host subtheory embedding, proposition equality supplied by propext*, quotient stream equality, classical-CIC definitional authority, hidden axiom supply, or external completeness principle.*

Theorem 865.2 (InternalTheoryBoundary NameCert obligations)**.** *For every* $\mathsf{InternalTheoryBoundary}^{\uparrow}$ *carrier*

$$I = (G, E, R, Z, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InternalTheoryBoundary}^{\uparrow}}$ *surface consists exactly of closed-generator exposure, literal AST equality exposure, structural recursor exposure, strict-purity exposure, apophatic boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Project the finite packet of Definition 865.1. The rows G, E, R, Z, A expose the five positive commitments named by chapter 9454.

The transport and replay rows H and C are componentwise. A consumer may move the packet or replay one of its reads, but the endpoints remain the displayed generator, AST equality, recursor, purity, and boundary rows.

The package and naming rows P, N bind the same surface to the local certificate. Since no additional coordinate is present, the NameCert obligations cannot import an external equality oracle, hidden host axiom supply, or classical-CIC definitional authority. □

Theorem 865.3 (InternalTheoryBoundary closed-generator non-escape)**.** *Every accepted* $\mathsf{InternalTheoryBoundary}^{\uparrow}$ *packet reads BEDC.MetaCIC objects through the closed-generator row G, the literal AST equality row E, and the structural recursor row R. A downstream consumer cannot replace those rows by a host syntax quotient, a classical-CIC embedding, an unlisted constructor, or a recursor not generated by G.*

Proof. Start with the carrier definition. The only object-generation row is G, the only equality row is E, and the only recursion row is R.

The continuation routes in C read these rows directly. The transport row H moves them componentwise and therefore preserves the same finite generator inventory, AST comparison, and structural recursion surface.

The excluded objects have no coordinate in $I = (G, E, R, Z, A, H, C, P, N)$. Hence a consumer has no route to a host syntax quotient, a classical-CIC embedding, an unlisted constructor, or an ungenerated recursor. □

Theorem 865.4 (InternalTheoryBoundary comparison secondary)**.** *Any comparison between BEDC.MetaCIC and classical CIC that is read from an* $\mathsf{InternalTheoryBoundary}^{\uparrow}$ *packet is secondary analysis over the displayed rows $G, E, R, Z, A, H, C, P, N$. It may consume the accepted-declaration surface of chapter 549, the forbidden-axiom refusal surface of chapter 622, and the directed proposition-transport surface of chapter 566, but it cannot define BEDC.MetaCIC by a subtheory relation, proposition extensionality, quotient equality, or external completeness.*

Proof. Read the apophatic boundary row A. It records the comparison as a boundary reading, not as a generator, equality, recursor, or purity source.

The package row P permits references to the three sibling certificates. Kernel acceptance supplies accepted-declaration rows, forbidden-axiom ancestry supplies strict refusal rows, and proposition-transport supplies directed implication rows in place of proposition equality.

Those sibling surfaces remain finite certificate data. None supplies a coordinate that turns classical CIC into the defining substrate of BEDC.MetaCIC, and none adds a host equality or completeness principle to the packet. □

Theorem 865.5 (InternalTheoryBoundary vision concretization)**.** *The carrier of Definition 865.1, the NameCert obligations of Definition 865.2, the closed-generator non-escape boundary of Definition 865.3, and the secondary-comparison boundary of Definition 865.4 realize the five positive commitments of chapter 9454: closed generators, literal AST equality, structural recursors, zero-axiom strict purity, and apophatic external boundary.*

Proof. The definition assigns the five commitments to the rows G, E, R, Z, A. The structural rows H, C, P, N then make those commitments readable as one finite local certificate.

The obligation theorem identifies the complete NameCert surface over those rows. The non-escape theorem prevents the closed generator, literal equality, and recursor clauses from being replaced by host objects.

The secondary-comparison theorem handles the apophatic clause: comparison language may be read as analysis over the packet, but it cannot define the packet. The strict-purity row Z supplies the remaining zero-axiom commitment through the accepted-declaration and forbidden-ancestry surfaces named in P. □

Closure status for $\mathsf{NameCert}_{\mathsf{InternalTheoryBoundary}^{\uparrow}}$**.**

Constructive story: An $\mathsf{InternalTheoryBoundary}^{\uparrow}$ packet is generated from closed generator rows, literal AST equality rows, structural recursor rows, strict-purity rows, apophatic boundary rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InternalTheoryBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 865.1, Definition 865.2, Definition 865.3, Definition 865.4, and Definition 865.5 bind the finite own-theory boundary packet for BEDC.MetaCIC.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InternalTheoryBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove global semantic conservativity, host-kernel correctness, a classical-CIC embedding, proposition extensionality, quotient equality, Lean verification, or bridge checking for $\mathsf{InternalTheoryBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, five-row obligation exactness, strict-purity routing, closed-generator non-escape, and secondary-comparison non-escape for the displayed packet.

Chapter 866

A Concrete Naming Certificate for $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$

The $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ packet turns the principle of chapter 9454 into a finite naming certificate. It is read together with the dependency-map surface of chapter 9455, the cross-map routing packet of chapter 839, and the forbidden-host-ancestry boundary of chapter 622. Those chapters expose dependency, routing, and forbidden-ancestry rows; this chapter packages the positive identity commitments by which BEDC.MetaCIC is named as its own substrate.

Definition 866.1 (MetaCIC identity commitment carrier)**.** *A* $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, E, R, P, A, H, C, Q, N)$$

with the following displayed rows:

1. *G is the closed-generator row, listing the finite constructor and formation rows by which the substrate is generated;*
2. *E is the literal-AST equality row, recording equality by displayed syntax and not by comparison with another calculus;*
3. *R is the structural-recursor row, naming the recursor and induction routes that act on the closed generators of G;*
4. *P is the strict-purity row, recording that accepted internal reads use the displayed generator, equality, and recursor rows without a hidden host axiom supply;*
5. *A is the apophatic-boundary row, naming the points where an internal BEDC.MetaCIC route stops without redefining the substrate by absence from classical CIC;*
6. *H records componentwise* hsame *transport for the generator, equality, recursor, purity, apophatic-boundary, continuation, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which a consumer replays the generator read, equality readback, recursor readback, strict-purity read, and apophatic-boundary read;*
8. *Q is the* Pkg *provenance over the own-theory principle, axiom-dependency map, audit-map dependency weave, forbidden-axiom ancestry packet,* hsame, Cont, *and* $\mathsf{NameCert}$ *source rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICIdentityCommitment}^{\uparrow}}$ *row.*

The carrier has no coordinate for a derivative-subtheory inclusion, a hidden comparison theorem with classical CIC, a host proof oracle, or an external axiom supply.

Theorem 866.2 (MetaCIC identity commitment NameCert obligations)**.** *For every* $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ *carrier*

$$I = (G, E, R, P, A, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICIdentityCommitment}^{\uparrow}}$ *surface consists exactly of closed-generator exposure through* G*, literal-AST equality exposure through* E*, structural-recursor exposure through* R*, strict-purity exposure through* P*, apophatic-boundary exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* Q*, and local naming through* N*.*

Proof. Project the finite carrier of Definition 866.1. The rows G, E, R, P, A expose the positive generator, equality, recursor, strict-purity, and apophatic-boundary commitments.

The transport and replay rows H and C bind those commitments componentwise. A consumer may transport or replay the packet, but the endpoints remain the displayed rows of the same carrier.

The provenance and naming rows Q, N package the same finite data as the local $\mathsf{NameCert}$ surface. Since the carrier has no further coordinate, no additional inclusion, comparison theorem, host oracle, or external supply can enter the certificate. □

Theorem 866.3 (MetaCIC identity commitment nonderivative boundary)**.** *For every accepted* $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ *carrier*

$$I = (G, E, R, P, A, H, C, Q, N),$$

the packet names BEDC.MetaCIC by its own closed generators, literal-AST equality, structural recursors, strict-purity row, and apophatic boundary. It cannot be read as a derivative-subtheory claim, a comparison-first definition, or a theorem that classical CIC supplies the meaning of the carrier.

Proof. Read the positive rows G, E, R, P, A. They state the substrate identity by generator closure, literal syntax, structural recursion, strict purity, and internal stopping points.

The continuation row C only replays those displayed reads, and the transport row H only carries them componentwise. Neither row contains a map from classical CIC into the carrier or a row that turns comparison language into a definition.

Thus the boundary is nonderivative by inventory. The finite packet exports exactly the positive identity rows and the local certificate rows; it exports no derivative-subtheory inclusion and no comparison-first meaning clause. □

Theorem 866.4 (MetaCIC identity commitment audit-map handoff)**.** *Every downstream audit-map consumer of an accepted* $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ *carrier must factor its read through the provenance row* Q *and the continuation row* C*. The consumer may cite the axiom-dependency map, audit-map dependency weave, and forbidden-axiom ancestry packets named in* Q*, but it cannot replace the positive identity rows* G, E, R, P, A *by a hidden dependency-map verdict or by a forbidden-ancestry refusal row.*

Proof. Start with the provenance row Q. It names the own-theory principle together with the axiom-dependency map, audit-map dependency weave, and forbidden-axiom ancestry packets as the cited handoff surfaces.

Next read the continuation row C. A downstream audit-map consumer reaches generator, equality, recursor, purity, and apophatic-boundary data only by replaying the displayed routes from the carrier.

The handoff is therefore a routing claim, not a replacement claim. Dependency-map and forbidden-ancestry consumers may inspect the packet through Q and C, but the positive identity rows remain the source rows of the local NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICIdentityCommitment}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICIdentityCommitment}^{\uparrow}$ packet is generated from closed-generator rows, literal-AST equality rows, structural-recursor rows, strict-purity rows, apophatic-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICIdentityCommitment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 866.1, Definition 866.2, Definition 866.3, and Definition 866.4 specify the finite positive identity packet, local NameCert surface, nonderivative boundary, and audit-map handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICIdentityCommitment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove global equivalence with classical CIC, close the axiom-dependency map, certify forbidden-ancestry purity, provide Lean-side formal verification, or bridge the packet to an external schema.

Upgrade path: Obligation closure requires checked carrier admission, row exactness for generators, equality, recursors, strict purity, and apophatic boundary, plus handoff preservation for downstream audit-map consumers.

Chapter 867

A Concrete Naming Certificate for MetaCICIdentityBoundary$^{\uparrow}$

The MetaCICIdentityBoundary$^{\uparrow}$ packet realizes the identity reading of chapter 9454 as a finite NameCert boundary for BEDC.MetaCIC. It is placed beside the dependency map of chapter 862, the normal-form false exclusion packet of chapter 732, the subject-reduction boundary of chapter 721, the closed-normal consistency packet of chapter 736, and the discharge socket of chapter 719. Those siblings expose dependency, normal endpoint, subject-reduction boundary, consistency endpoint, and discharge-socket rows; this chapter names the identity boundary that keeps those rows visible without importing a classical CIC parent theory.

Definition 867.1 (MetaCIC identity-boundary carrier)**.** *A* MetaCICIdentityBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, E, R, P, B, L, H, C, Q, N)$$

with the following displayed rows:

1. *G is the closed-generator row for BEDC.MetaCIC constructors, formation rows, and finite introduction routes;*
2. *E is the literal AST equality row, reading identity by displayed syntax rather than by quotienting through an external calculus;*
3. *R is the structural recursor row, exposing the induction and eliminator routes available on the closed generators;*
4. *P is the strict-purity row, recording that accepted reads cite only displayed generator, equality, recursor, ledger, transport, route, provenance, and naming rows;*
5. *B is the external-boundary row, recording exactly where a request asks for a classical CIC comparison, a closed subject-reduction theorem, or another nonlocal parent-theory commitment;*
6. *L is the identity ledger, linking G, E, R, P, B to the five sibling surfaces named above;*
7. *H records componentwise* hsame *transport for all displayed rows;*
8. *C records the finite* Cont *routes by which consumers read the generator, equality, recursor, purity, boundary, ledger, provenance, and local naming rows;*
9. *Q is the* Pkg *provenance over chapter 9454, chapter 862, chapter 732, chapter 721, chapter 736, chapter 719,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICIdentityBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a closed subject-reduction theorem, no map from classical CIC into BEDC.MetaCIC, and no hidden axiom supply.

Definition 867.2 (MetaCIC identity-boundary classifier)**.** *Two* $\mathsf{MetaCICIdentityBoundary}^{\uparrow}$ *carriers*

$$M = (G, E, R, P, B, L, H, C, Q, N)$$

and

$$M' = (G', E', R', P', B', L', H', C', Q', N')$$

are classifier-equivalent exactly when the closed-generator rows, literal AST equality rows, structural recursor rows, strict-purity rows, external-boundary rows, identity-ledger rows, componentwise transports, continuation routes, provenance rows, and local naming rows agree through the displayed hsame *and* Cont *data. The classifier does not compare carriers by semantic equivalence with classical CIC or by any theorem absent from the displayed sibling packets.*

Theorem 867.3 (MetaCIC identity-boundary NameCert obligations)**.** *For every* $\mathsf{MetaCICIdentityBoundary}^{\uparrow}$ *carrier*

$$M = (G, E, R, P, B, L, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICIdentityBoundary}^{\uparrow}}$ *surface consists of five finite commitment rows:*

1. *closed-generator exposure through G;*
2. *literal AST equality exposure through E;*
3. *structural recursor exposure through R;*
4. *strict-purity exposure through P;*
5. *external-boundary exposure through B, with ledger accountability through L, H, C, Q, N.*

These rows certify the positive identity of BEDC.MetaCIC as its own substrate boundary. They do not certify a closed subject-reduction theorem, a comparison theorem with classical CIC, or any external proof supply.

Proof. Project the finite carrier of Definition 867.1. The rows G, E, R, P, B are the five commitment rows, and L, H, C, Q, N make those commitments inspectable by ledger, transport, replay, provenance, and local naming.

Classifier exactness follows from Definition 867.2. Agreement is checked on the displayed rows only, so an external comparison theorem or missing subject-reduction theorem cannot become part of the classifier.

The NameCert surface is therefore exactly the five displayed commitments plus their structural accountability rows. Since the carrier has no further coordinate, no sixth commitment or hidden parent-theory supply is exported. □

Theorem 867.4 (MetaCIC identity-boundary ledger non-escape)**.** *Every consumer read of an accepted* $\mathsf{MetaCICIdentityBoundary}^{\uparrow}$ *carrier factors through*

$$G, E, R, P, B, L, H, C, Q, N.$$

If a route asks for a closed subject-reduction theorem, a semantic embedding into classical CIC, a host equality oracle, or an external axiom dependency not recorded by the sibling dependency-map packet, the request lands in the external-boundary row B and cannot be exported as a theorem row.

Proof. Read the consumer route through C. If the route reaches G, E, R, P, L, H, Q, N, it remains inside the positive identity packet and its structural accountability rows.

If the route asks for material outside those rows, the carrier offers only B as a boundary coordinate. The ledger row L ties that boundary to the axiom-dependency map, normal false exclusion, subject-reduction boundary, closed-normal consistency, and discharge-socket siblings.

Thus a boundary read remains a boundary read. It cannot be reclassified as closed subject reduction, semantic parenthood, host equality, or an undisplayed axiom dependency. □

Theorem 867.5 (MetaCIC identity-boundary sibling independence)**.** ***Independence witness:*** *The axiom-dependency-map packet, normal-false-exclusion packet, MetaCIC subject-reduction-boundary packet, closed-normal-consistency packet, and subject-reduction-discharge-socket packet are not bijective to the* $\mathsf{MetaCICIdentityBoundary}^{\uparrow}$ *carrier. The dependency map has claim, mode, witness, and supply rows; normal false exclusion has a beta-strong false endpoint row; the subject-reduction boundary has obstruction rows for typed reduction preservation; closed-normal consistency has a positive normal endpoint; and the discharge socket has four discharge-commitment rows. None has exactly the five identity commitments* G, E, R, P, B *with the identity ledger* L *and local boundary classifier of Definition 867.2.*

Proof. Compare row inventories. The sibling chapters named in the witness each expose a specialized consumer surface: dependency accounting, normal-endpoint exclusion, subject-reduction obstruction, closed-normal consistency, or four-row discharge sockets.

The identity-boundary carrier instead exposes closed generators, literal AST equality, structural recursors, strict purity, and external-boundary rows as one packet. Its ledger cites the siblings, but citation is not a bijection of carriers.

Therefore a consumer may use the siblings as provenance and boundary context, yet cannot replace this five-commitment identity packet by any one of them or by a product of their visible theorem rows. □

Theorem 867.6 (MetaCIC identity-boundary TasteGate follow-up)**.** *The Lean-side follow-up for* $\mathsf{MetaCICIdentityBoundary}^{\uparrow}$ *must provide a* $\mathsf{ChapterTasteGate}$ *route whose carrier is nontrivial and field-faithful. In particular, the future 'TasteGate.lean' development must supply* $\mathsf{Nontrivial}$ *and* $\mathsf{FieldFaithful}$ *instances for the identity-boundary carrier before a checked gate can claim that the packet has more than a vacuous event-flow encoding and that its encoding preserves the five displayed commitment fields.*

Lean4 ✓ `BEDC.Derived.MetaCICIdentityBoundaryUp.MetaCICIdentityBoundaryTasteGate single carrier alignment`

Proof. The carrier has five named commitment fields and structural rows that must remain distinguishable. A gate that recognizes only an event-flow round trip would not by itself show that the fields G, E, R, P, B are preserved as distinct BEDC commitments.

Nontriviality rules out the empty or singleton reading of the carrier, and field faithfulness records that the encoding reflects the displayed field structure. These are exactly the extra gate rows needed before the identity-boundary packet can be read as a checked TasteGate object. □

Falsifiable prediction: If a MetaCIC identity-boundary packet is accepted, no consumer route may turn its external-boundary row into a closed subject-reduction theorem without citing a subject-reduction discharge sibling.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICIdentityBoundary}^{\uparrow}}$.

Constructive story: A MetaCICIdentityBoundary$^{\uparrow}$ packet is generated from closed-generator rows, literal AST equality rows, structural recursor rows, strict-purity rows, external-boundary rows, an identity ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICIdentityBoundary}\uparrow}$, seed)

Scope closed: Definition 867.1, Definition 867.2, Definition 867.3, Definition 867.4, Definition 867.5, and Definition 867.6 specify the seed carrier, classifier, five-row NameCert surface, non-escape ledger, sibling independence witness, and gate-facing follow-up.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICIdentityBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove closed subject reduction, construct a semantic embedding into classical CIC, close the sibling packets, provide Lean-side formal verification, or bridge the packet to an external schema.

Upgrade path: Obligation closure requires Lean encoding of the carrier and classifier, checked five-row obligation exposure, ledger non-escape, sibling independence, and a TasteGate route with Nontrivial and FieldFaithful instances.

Chapter 868

A Concrete Naming Certificate for $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$

The $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ packet realizes the dependency-map vision of chapter 9455 as a finite audit carrier. It sits beside the paper-marker resolution witness of chapter 669 and the forbidden-host-axiom boundary of chapter 622: those chapters certify marker resolution and forbidden ancestry refusal, while this chapter packages the mode, witness, and required-axiom rows that make a dependency-map entry readable by a $\mathsf{NameCert}$ consumer.

Definition 868.1 (AxiomDependencyAuditMap carrier)**.** *An* $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, M, W, A, L, H, C, P, N)$$

with the following visible coordinates:

1. *K is the audited claim or declaration row whose dependency status is being read;*
2. *M is the mode row, with one of the displayed readings internal, restricted, or externally supplied;*
3. *W is the witness row for the selected mode, carrying the internal proof witness, restricted witness, refusal witness, counterexample row, or named socket;*
4. *A is the required-axiom row, recording empty supply, a restricted-surface condition, or the finite named supply required by the mode;*
5. *L is the finite audit ledger linking the claim row to the marker-resolution and forbidden-ancestry packets named by provenance;*
6. *H records componentwise* hsame *transport for the claim, mode, witness, required-axiom, ledger, continuation, package, and local naming rows;*
7. *C records the visible* Cont *routes by which a consumer replays the mode read, witness read, required-axiom read, and ledger read;*
8. *P is the* Pkg *provenance over the dependency-map vision,* $\mathsf{PaperLeanDriftWitness}^{\uparrow}$, $\mathsf{ForbiddenAxiomAncestry}^{\uparrow}$, $\mathsf{AxiomDependencyMap}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyAuditMap}^{\uparrow}}$ *row.*

The carrier exports no hidden proof oracle, no global theorem census, no host axiom permission, and no conversion from a restricted or supplied row into an unconditional BEDC theorem.

Theorem 868.2 (AxiomDependencyAuditMap mode totality)**.** *For every accepted* $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ *packet*

$$D = (K, M, W, A, L, H, C, P, N),$$

the mode row M is exhausted by the three displayed readings internal, restricted, and externally supplied. The internal reading pairs W with an internal witness and A with empty supply; the restricted reading pairs W with a restricted witness and A with the restricted surface; the externally supplied reading pairs W with the refusal, counterexample, or socket row and A with the named finite supply.

Proof. Project the accepted packet and inspect its mode coordinate M. The carrier definition lists only the internal, restricted, and externally supplied readings, so the case split has exactly those branches.

In the internal branch, the witness row and required-axiom row are read together: W records the displayed internal witness and A records empty supply.

In the restricted branch, the same projection keeps the restriction visible by pairing W with the restricted witness and A with the restricted surface. No carrier coordinate erases that restriction.

In the externally supplied branch, W records the refusal, counterexample, or socket row, while A records the named finite supply. The packet therefore reports supply as audit data rather than as a BEDC proof. □

Lean4 ✓ `BEDC.Derived.AxiomDependencyAuditMapUp.AxiomDependencyAuditMap mode totality`

Theorem 868.3 (AxiomDependencyAuditMap NameCert obligations)**.** *For the carrier of Definition 868.1, the local* $\mathsf{NameCert}_{\mathsf{AxiomDependencyAuditMap}^{\uparrow}}$ *surface consists exactly of carrier exposure for D, mode totality for M, witness exposure for W, required-axiom exposure for A, finite ledger exposure for L, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. Every consumer of N must read the dependency lattice through these rows and cannot replace them by an unrecorded host-kernel comparison.*

Lean4 ✓ `BEDC.Derived.AxiomDependencyAuditMapUp.AxiomDependencyAuditMapCarrier namecert obligations`

Proof. Carrier exposure is projection of Definition 868.1. The claim, mode, witness, required-axiom, ledger, transport, replay, package, and local naming rows are all visible coordinates of the same finite packet.

Mode totality is supplied by Definition 868.2. It fixes how M, W, A may be consumed in the internal, restricted, and externally supplied branches.

The ledger row L links the claim to the paper-marker and forbidden-ancestry packets named by P. The transport and continuation rows H, C preserve that link componentwise.

Since the carrier has no coordinate for a host proof oracle or an unrecorded axiom comparison, the local NameCert surface is exhausted by the displayed dependency rows. □

Theorem 868.4 (AxiomDependencyAuditMap non-escape)**.** *Every public read from an accepted* $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ *packet remains inside the displayed claim, mode, witness, required-axiom, ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The packet cannot turn a restricted row into an unrestricted theorem, a required-supply row into internal BEDC content, a refusal row into a proof, or a dependency-map entry into a hidden host axiom.*

Proof. Project the packet as $D = (K, M, W, A, L, H, C, P, N)$. These rows are the complete carrier surface.

Apply Definition 868.3. It requires every consumer to pass through the displayed mode, witness, required-axiom, and ledger rows before reading the local naming certificate.

The mode-totality theorem fixes the interpretation of M, W, A in each branch. The internal, restricted, and externally supplied branches remain disjoint audit readings, and no branch exports the excluded host objects.

Finally replay the packet through H, C, P, N. Componentwise transport and package provenance preserve the same finite rows, so there is no route to an unrestricted theorem, hidden supply, proof oracle, or host axiom outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomDependencyAuditMap}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ packet is generated from an audited claim row, a mode row, a witness row, a required-axiom row, a finite audit ledger, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxiomDependencyAuditMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 868.1, Definition 868.2, Definition 868.3, and Definition 868.4 specify the finite dependency-map packet for mode, witness, required-axiom, ledger, and non-escape reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxiomDependencyAuditMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not enumerate all declarations, prove global dependency-map completeness, certify host-kernel proof ancestry, discharge named external supply, or bridge the packet to an external schema.

Upgrade path: Obligation closure requires checked carrier admission, mode totality, witness and required-axiom exactness, ledger exposure, and non-escape for the displayed packet.

Chapter 869

A Concrete Naming Certificate for $\mathsf{BEDCSelfSubstrate}^{\uparrow}$

The $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ packet turns the identity principle of chapter 9454 into a substrate-facing finite certificate. It is composite over the axiom-dependency map of chapter 862, the axiom-query ledger of chapter 654, the kernel-acceptance audit witness of chapter 547, and the normal false exclusion packet of chapter 732: those siblings consume query, verdict, acceptance, and false-exclusion rows, while this packet records the five positive substrate commitments that must be present before a downstream MetaCIC identity consumer classifies a claim.

869.1 Carrier and certificate surface

Definition 869.1 (BEDC self-substrate carrier)**.** *A* $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, E, R, P, A, H, C, Q, N)$$

with the following displayed rows:

1. *G is the closed-generator row for the BEDC.MetaCIC substrate;*
2. *E is the literal AST equality row used by internal identity reads;*
3. *R is the structural-recursion row for the closed syntax and history carriers;*
4. *P is the zero-axiom purity row, recording the absence of additional axiom, choice, quotient-soundness, proposition-extensionality, and host-conversion coordinates;*
5. *A is the apophatic-boundary row for exports not supplied by the substrate packet;*
6. *H records componentwise* hsame *transport for the five commitment rows, provenance, and local naming;*
7. *C records displayed* Cont *routes by which audit consumers read the five commitments;*
8. *Q is the* Pkg *provenance over the self-theory principle, the audit-map siblings,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BEDCSelfSubstrate}^{\uparrow}}$ *row.*

The carrier stores no declaration-specific query row, no verdict row, no kernel-acceptance replay, and no false-exclusion witness as its own field.

Theorem 869.2 (BEDC self-theory obligation surface)**.** *For every accepted* $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ *carrier*

$$S = (G, E, R, P, A, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{BEDCSelfSubstrate}^{\uparrow}}$ *surface is exhausted by the closed-generator, literal AST equality, structural-recursion, zero-axiom purity, and apophatic-boundary commitments, together with their componentwise transport, displayed routes, provenance, and local name.*

Proof. Project the displayed carrier. The positive substrate commitments are exactly G, E, R, P, A.

Read H and C next. Transport and continuation preserve those five rows as displayed BEDC data and give downstream consumers routes to them without adding a declaration query, verdict, acceptance replay, or false-exclusion row.

Finally inspect Q and N. Provenance cites the self-theory principle and the audit-map siblings, while local naming packages the same finite surface. Since no additional coordinate occurs in S, the listed rows exhaust the local obligation surface. □

Lean4 ✓ `BEDC.Derived.BedcSelfSubstrateUp.BedcSelfSubstrateObligationSurface`

Theorem 869.3 (BEDC self-theory falsifiable boundary)**.** *If a downstream MetaCIC identity consumer reads an accepted* $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ *packet, then any exported substrate claim factors through one of the five displayed commitment rows before it can be classified as internal, restricted, or blocked. A read that requires a declaration-specific query, verdict, acceptance replay, false-exclusion witness, quotient stream, host equality, or comparison embedding is outside this carrier.*

Proof. Start with the five commitment rows in Definition 869.1. They are the only substrate rows exposed before transport, continuation, provenance, and naming.

Use the obligation-surface theorem. Every admitted route through H, C, Q, N returns to one of G, E, R, P, A, so the consumer must classify the claim after reading one of those commitments.

The excluded rows are absent from the carrier. Any consumer requiring one of them must instead cite a sibling audit packet, and therefore cannot classify that read as an internal $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ export. □

Falsifiable prediction: If a downstream MetaCIC identity consumer reads a BEDC-self-substrate packet, then any exported substrate claim factors through one of the five displayed commitment rows before it can be classified as internal, restricted, or blocked.

Independence witness: Nearest siblings: axiomdependencymap, axiomqueryledger, kernelacceptanceauditwitness, normalfalseexclusion, auditgateboundary. None is bijective to this carrier because each sibling stores a query, verdict, acceptance, false-exclusion, or gate row, while this carrier stores the five positive substrate commitments and lacks a declaration-specific query row.

Closure status for $\mathsf{NameCert}_{\mathsf{BEDCSelfSubstrate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BEDCSelfSubstrate}^{\uparrow}$ packet is generated from finite BHist rows for the positive substrate commitments by which BEDC names its own internal substrate, plus transport, route, provenance, and local naming rows. Each downstream identity read must pass through one of the displayed commitment rows before it can be classified as internal, restricted, or blocked, so Cont exposes the commitment route rather than a hidden self-model. Componentwise hsame preserves the commitment inventory, and Pkg records the neighboring axiom, query, acceptance, false-exclusion, and audit-gate dependencies. The packet is a finite self-substrate commitment surface, not a declaration-specific oracle.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BEDCSelfSubstrate}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the BEDC self-substrate carrier, five commitment rows, falsifiable prediction, and sibling independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BEDCSelfSubstrate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a complete self-model, unrestricted internal truth predicate, declaration-specific query verdict, host reflection equality, or hidden kernel oracle.

Upgrade path: Obligation closure requires Lean encodings of the commitment carrier, consumer factorization through commitment rows, transport stability, and sibling non-bijection boundary.

Chapter 870

A Concrete Naming Certificate for $\mathsf{DiagonalLimitStability}^{\uparrow}$

The $\mathsf{DiagonalLimitStability}^{\uparrow}$ packet is the finite comparison layer between the diagonal representative used in the Real completeness route of chapter 9238 and the sealed $\mathsf{Real}^{\uparrow}$ limit route. It consumes the selector surface of chapter 697, the seal boundary of chapter 386, the mesh rows of chapter 644, the diagonal budget rows of chapter 853, and the observation-bound surface of chapter 637. The carrier records a displayed stability comparison; it does not identify quotient streams, choose a host limit, or import ambient completeness.

Falsifiable prediction: If a DiagonalLimitStabilityUp packet is accepted, every RealUp seal read factors through either the displayed DiagonalTailSelectorUp row or the displayed CauchyLimitSealUp row; a consumer route bypassing both refutes the packet.

Independence witness: regularcauchymesh, cauchydiagonalbudget, diagonaltailselector, cauchylimitseal, realapproximationenvelope: none is bijective because this carrier has a stability-comparison row between the diagonal representative and the sealed RealUp route; mesh chapters lack the seal row, selector chapters lack the stability comparison, seal chapters lack the diagonal representative source, and envelope chapters package observation bounds without the representative-seal classifier.

Definition 870.1 (Diagonal-limit stability carrier)**.** *A* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, D, L, S, W, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational name whose finite observations feed the diagonal construction;*
2. *D is the diagonal representative row, read through the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *selector surface;*
3. *L is the limit selector row that records the finite diagonal-limit route selected for the representative;*
4. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary;*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used to compare the representative readback with the sealed route;*
6. *H stores componentwise* hsame *transports for R, D, L, S, W, C, P, N;*

7. *C stores the* Cont *continuations that replay source, representative, selector, seal, window, transport, provenance, and local naming reads;*
8. *P is the* Pkg *provenance over the same finite route;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient regular streams, host real equality, a choice-selected limit representative, trichotomy, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.DiagonalLimitStabilityUp.DiagonalLimitStabilityUp`

Theorem 870.2 (Diagonal-limit stability NameCert obligations)**.** *For the carrier of Definition 870.1, the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission for T, source exposure through R, diagonal representative exposure through D, limit-selector exposure through L,* $\mathsf{Real}^{\uparrow}$ *seal exposure through S, finite-window stability comparison through W, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier displayed in Definition 870.1. The source, representative, selector, seal, and finite-window rows are explicit coordinates, and the transport, route, provenance, and local certificate rows are the only structural rows available to a consumer.

The diagonal representative row is read through the sibling selector surface, while the seal row is read through the sibling Cauchy-limit seal surface. The stability comparison is therefore not a new analytic equality; it is the finite assertion that the selected representative row and the sealed route are read under the same displayed window and structural provenance.

The excluded objects have no coordinate in T. Hence the local NameCert obligations are exhausted by the nine displayed rows. □

Lean4 ✓ `BEDC.Derived.DiagonalLimitStabilityUp.DiagonalLimitStabilityTasteGate single carrier alignment`

Theorem 870.3 (Diagonal-limit stability RealCompleteness handoff)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer that reads the sealed diagonal limit from an accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier $T = (R, D, L, S, W, H, C, P, N)$ must first pass through the displayed diagonal representative row D or the displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row S, and in either case the read is replayed through the same finite window W, transports H, continuations C, provenance P, and local naming row N.*

Proof. Read the Real completeness route from chapter 9238. A consumer of the sealed diagonal limit needs a representative side and a seal side before the $\mathsf{Real}^{\uparrow}$ read can be inspected.

The representative side of T is the row D, which is anchored in the diagonal-tail selector sibling. The seal side of T is the row S, which is anchored in the Cauchy-limit seal sibling. The limit-selector row L names the finite route between them, and W is the common observation window on which the stability comparison is made.

The structural rows H, C, P, N replay the same finite comparison for downstream consumers. Since no additional representative, seal, quotient, host equality, selected external limit, or ambient completeness row is present, every sealed diagonal-limit read factors through the displayed handoff. □

Theorem 870.4 (Diagonal-limit stability budget-seal factorization)**.** *Every* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *consumer that reads the* $\mathsf{RealCompleteness}^{\uparrow}$ *handoff factors through the diagonal budget row D,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{Real}^{\uparrow}$ *seal row S,* BHist *history rows,*

componentwise hsame *row H, displayed* Cont *route C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *row N. The consumer receives no host limit, ambient completeness coordinate, external representative equality, or detached Real seal outside this packet.*

Proof. Start from the Real-completeness handoff theorem, Definition 870.3. It forces every sealed diagonal-limit read to enter through D or S and then replay through the same finite window and structural rows.

Use the root-route budget package from the sibling diagonal-limit compatibility chapter, Definition 646.153. It supplies the finite budget route that feeds the diagonal representative side before the Real seal is read.

The local NameCert obligations in Definition 870.2 keep the readback row R, window row W, transports H, continuation C, provenance P, and naming row N attached to the same carrier. Hence the Real-completion handoff factors from finite budget to seal without exporting a host limit or ambient completeness coordinate. □

Theorem 870.5 (Representative-seal window comparison)**.** *For an accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier $T = (R, D, L, S, W, H, C, P, N)$, the displayed diagonal representative row D and the displayed* $\mathsf{Real}^{\uparrow}$ *seal row S are compared only through the common finite window row W. Neither D nor S can be consumed as an independent Real-completeness input outside the same packet.*

Proof. The carrier exposes D as the representative-side row and S as the seal-side row, but the comparison coordinate supplied by the carrier is W. Thus any consumer that reads either side as part of the stability comparison must also read the window coordinate that makes the two sides commensurable.

There is no coordinate in T for an external representative equality, a host-level seal equality, or a separate comparison witness. The only displayed comparison boundary is the packet's own window row, so independent consumption of D or S would require a row absent from the carrier. □

Theorem 870.6 (Real-completeness window coverage)**.** *Every* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer of the accepted packet $T = (R, D, L, S, W, H, C, P, N)$ factors through W, H, C, P, N after choosing either the representative side D or the seal side S.*

Proof. By Definition 870.3, a Real-completeness consumer first reaches the packet through D or through S. Once that side is chosen, the same theorem records the replay rows W, H, C, P, N as the boundary through which the read continues.

The rows W, H, C, P, N are not optional decorations: W fixes the common observation window, H carries the componentwise transports, C carries the displayed continuation routes, P carries provenance, and N carries the local naming row. Since the carrier has no additional Real-completeness boundary, each accepted packet read factors through those rows. □

Theorem 870.7 (Transport replay for local NameCert obligations)**.** *For an accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier $T = (R, D, L, S, W, H, C, P, N)$, the componentwise* hsame *rows in H and the displayed* Cont *routes in C replay the same nine-coordinate packet and preserve the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *obligations.*

Proof. The row H is displayed as componentwise transport for R, D, L, S, W, C, P, N, while C is displayed as the continuation replay for the same source, representative, selector, seal, window, transport, provenance, and local naming reads. Both rows therefore address the coordinates of the single carrier T rather than a second packet.

The local NameCert obligations are exactly the rows listed in Definition 870.2. Because H and C replay those rows componentwise and by displayed continuations, downstream use preserves the same local obligation surface. □

Theorem 870.8 (Diagonal-limit stability obligation package)**.** *For an accepted* DiagonalLimitStability$^{\uparrow}$ *carrier* $T = (R, D, L, S, W, H, C, P, N)$*, the carrier, local* NameCert *obligations,* RealCompleteness$^{\uparrow}$ *handoff, falsifiable prediction, independence witness, TasteGate marker, and route exclusions jointly form the obligation package for* DiagonalLimitStability$^{\uparrow}$ *over* RegSeqRat$^{\uparrow}$*,* DiagonalTailSelector$^{\uparrow}$*,* CauchyLimitSeal$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package binds selector route exactness through* D*, seal route exactness through* S*, finite-window coverage through* W*, componentwise transport replay through* H*, continuation replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. First use Definition 870.2. It projects the accepted carrier and binds the source, representative, selector, seal, finite-window, transport, continuation, provenance, and local naming rows as the complete local NameCert surface.

Next use Definition 870.3. It forces every Real-completeness seal read through either the displayed diagonal representative row D or the displayed CauchyLimitSeal$^{\uparrow}$ row S, and in both cases replays the same window W and structural rows H, C, P, N.

The selector and seal routes are therefore exact at the displayed boundary. Selector exactness is the fact that D is the only diagonal representative row available to the carrier; seal exactness is the fact that S is the only Real$^{\uparrow}$ seal row available to the carrier. Window coverage is the common row W, which is read before downstream consumers receive the sealed route.

Finally inspect the chapter-level side conditions. The falsifiable prediction excludes consumers that bypass both displayed route rows, the independence witness separates the packet from its sibling carriers, the TasteGate marker records single-carrier alignment, and the excluded coordinates rule out quotient streams, host real equality, choice-selected limit representatives, trichotomy, and ambient completeness. These clauses are precisely the obligation package needed before an obligation-closure reading of the same finite packet. □

Theorem 870.9 (Diagonal-limit stability selector-seal scope)**.** *For an accepted* DiagonalLimitStability$^{\uparrow}$ *carrier* $T = (R, D, L, S, W, H, C, P, N)$*, the row* D *is read only through* DiagonalTailSelector$^{\uparrow}$*, and the row* S *is read only through* CauchyLimitSeal$^{\uparrow}$*, before any* RealCompleteness$^{\uparrow}$ *consumer reads the stability comparison. The comparison therefore has no route that bypasses the selector-seal pair.*

Proof. Start from Definition 870.1 and Definition 870.3. They expose D as the diagonal representative route and S as the seal route before any Real-completeness read.

Use Definition 870.5 and Definition 870.6. The common window W is the only comparison boundary for the two sides, and every Real-completeness consumer reaches it after choosing D or S.

The transport replay and obligation package rows then keep the selector and seal reads inside the same finite packet. □

Theorem 870.10 (Diagonal-limit stability primitive route exhaustion)**.** *For an accepted* DiagonalLimitStability$^{\uparrow}$ *carrier* $T = (R, D, L, S, W, H, C, P, N)$*, the rows* W, H, C, P, N *exhaust the primitive* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data of the selector-seal route. They do not introduce quotient-stream equality, host real equality, a selected limit representative, trichotomy, or ambient completeness.*

Proof. Apply Definition 870.2. It lists the finite source, representative, selector, seal, window, transport, continuation, provenance, and local naming rows as the local obligation surface.

Then use Definition 870.7. The H and C rows replay the same carrier by componentwise hsame transport and displayed continuations, while P and N remain provenance and local naming rows.

Together with the carrier, handoff, window coverage, and obligation package, these rows leave no primitive coordinate outside W, H, C, P, N for the selector-seal route. □

Theorem 870.11 (Diagonal-limit stability NameCert scope closure)**.** *The selector-seal scope row and primitive route exhaustion row, combined with the obligation package of Definition 870.8, close the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *scope for the displayed packet* $T = (R, D, L, S, W, H, C, P, N)$*. The closed scope is exactly the route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DiagonalTailSelector}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *named in the carrier.*

Proof. Use Definition 870.9. It fixes the consumer-visible order of the selector row D, seal row S, and Real-completeness comparison.

Use Definition 870.10. It fixes the primitive route rows W, H, C, P, N as the complete BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

Now apply Definition 870.8. Since the carrier, handoff, representative-seal comparison, window coverage, and transport replay have already been packaged for the same finite tuple, the local NameCert scope closes exactly over the displayed selector-seal packet and over nothing else. □

Theorem 870.12 (Diagonal-limit stability selector-seal primitive binding)**.** *For an accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier* $T = (R, D, L, S, W, H, C, P, N)$*, the row* D *is bound to the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *primitive surface, and the row* S *is bound to the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *primitive surface together with the displayed* $\mathsf{Real}^{\uparrow}$ *seal row. Both reads are replayed only through the displayed* BHist *packet, componentwise* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *row.*

Proof. Use the carrier definition to isolate the two route rows. The coordinate D is the only diagonal representative row available to the packet, and the coordinate S is the only seal row available to the packet.

Apply Definition 870.9. It fixes the consumer-visible order in which the selector row and seal row can be read before a Real-completeness consumer inspects the stability comparison.

Now apply Definition 870.10. The replay rows are exactly the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so neither the selector read nor the seal read has a second primitive route. □

Theorem 870.13 (Diagonal-limit stability window-transport primitive binding)**.** *For an accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *carrier* $T = (R, D, L, S, W, H, C, P, N)$*, the common finite window* W *and the replay rows* H, C, P, N *are the whole primitive transport surface for the selector-seal comparison. This surface contains no quotient stream, host real equality, selected representative, trichotomy row, or ambient completeness coordinate.*

Proof. Start with Definition 870.6. It forces every Real-completeness consumer to pass through the common window W and the structural rows H, C, P, N after the representative side or the seal side has been chosen.

Use Definition 870.7. The H and C rows replay the same accepted carrier, while P and N keep provenance and local naming attached to that carrier.

The exclusions follow from the carrier definition and from Definition 870.10. Since none of the excluded objects is a coordinate of T, it cannot enter the primitive transport surface. □

Theorem 870.14 (Diagonal-limit stability scoped NameCert boundary)**.** *The selector-seal primitive binding of Definition 870.12, the window-transport primitive binding of Definition 870.13, and*

the local scope closure of Definition 870.11 justify scoped closure for the displayed $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *boundary of* $T = (R, D, L, S, W, H, C, P, N)$.

Proof. The selector-seal primitive binding fixes the only route by which D and S can be read. It binds the representative side to $\mathsf{DiagonalTailSelector}^{\uparrow}$ and the seal side to $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

The window-transport primitive binding fixes the only route by which those reads are transported. It keeps W, H, C, P, N as the finite primitive surface and excludes the analytic and quotient-level coordinates absent from the carrier.

Finally apply Definition 870.11. Its closed local scope is therefore not merely an obligation list; it is the displayed scoped boundary of the local $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ packet. □

Theorem 870.15 (Diagonal-limit stability tail-refinement invariance)**.** *Let*

$$T = (R, D, L, S, W, H, C, P, N)$$

and

$$T' = (R, D', L', S, W', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitStability}^{\uparrow}$ *packets with the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* S*. Suppose* W' *is a displayed* $\mathsf{StreamName}^{\uparrow}$ *tail refinement of* W*,* D' *is the transported* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *row for the refined window, and the row* S *is read in both packets through the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary. Then every* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer that reads the sealed diagonal-limit route through* T' *receives the same sealed* $\mathsf{Real}^{\uparrow}$ *route as the corresponding read through* T*, now replayed through* W', H', C', P', N'*. The refinement introduces no quotient stream, host real equality, choice-selected representative, trichotomy, or ambient completeness row.*

Proof. First compare the two packets at the seal side. By Definition 870.3, a Real-completeness consumer must enter the accepted packet through the displayed diagonal representative row or the displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ row, and in either case must continue through the packet window and structural rows.

Next identify the representative and seal comparison boundary. The representative-seal comparison of Definition 870.5 says that the representative side and the seal side are compared only through the common finite window. Since W' is a displayed tail refinement of W and D' is the transported selector row for that refined window, the refined packet changes the inspection window but not the sealed $\mathsf{Real}^{\uparrow}$ boundary.

Now use the coverage and replay rows. By Definition 870.6, every consumer read factors through the window and structural rows after choosing the representative side or the seal side. By Definition 870.7, the transports and continuations replay the same carrier coordinates, so the refined rows H', C', P', N' serve the same local route over the refined window W'.

Finally apply Definition 870.11. The closed local scope contains exactly the displayed route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Therefore the tail-refined packet preserves the sealed RealUp route for every RealCompletenessUp consumer and supplies no additional primitive or analytic coordinate. □

Theorem 870.16 (Cofinal tail-refinement confluence)**.** *Let*

$$T_0 = (R, D, L, S, W, H, C, P, N),$$

$$T_1 = (R, D_1, L_1, S, W_1, H_1, C_1, P_1, N_1),$$

and

$$T_2 = (R, D_2, L_2, S, W_2, H_2, C_2, P_2, N_2)$$

be accepted $\mathsf{DiagonalLimitStability}^{\uparrow}$ *packets with the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* S*. Suppose* W_1 *and* W_2 *are displayed* $\mathsf{StreamName}^{\uparrow}$ *tail refinements of* W*, and suppose there is a common cofinal displayed* $\mathsf{StreamName}^{\uparrow}$ *tail refinement* W_{12} *of both* W_1 *and* W_2 *such that the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *rows* D_1 *and* D_2 *transport to the same displayed selector read over* W_{12} *while the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row remains the seal row* S*. Then every* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer reading through* T_1 *and every* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer reading through* T_2 *factors through the same sealed* $\mathsf{Real}^{\uparrow}$ *route replayed over the common window* W_{12}*. The confluence uses only the displayed* BHist *packet, componentwise* hsame *transports,* Cont *continuations,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *rows, and it supplies no quotient stream equality, host real equality, selected representative, trichotomy row, or ambient completeness coordinate.*

Proof. First compare each accepted refinement with the base packet. Apply Definition 870.15 to the refinement from T_0 to T_1, and apply the same theorem to the refinement from T_0 to T_2. Both readings preserve the sealed $\mathsf{Real}^{\uparrow}$ route because the source row R and the seal row S are shared.

Next force both consumer reads into the common comparison boundary. By Definition 870.5, the representative side and the seal side of each accepted packet are compared only through the displayed finite window. By Definition 870.6, every $\mathsf{RealCompleteness}^{\uparrow}$ consumer reaches the window and structural rows after choosing the representative side or the seal side. Since W_{12} is cofinal over both W_1 and W_2, the two refined reads pass through the same displayed window boundary after transporting D_1 and D_2.

Now replay the transported packets over that boundary. Use Definition 870.7 to identify the componentwise hsame rows and displayed Cont routes as replays of the same carrier coordinates. The rows H_1, C_1, P_1, N_1 and H_2, C_2, P_2, N_2 therefore contribute only the displayed transport, continuation, provenance, and local naming data over W_{12}, while the seal side remains S.

Finally apply Definition 870.11. The closed local scope is exactly the route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Thus both accepted refinements factor through the same sealed $\mathsf{Real}^{\uparrow}$ route over W_{12}, with no additional analytic, quotient, representative-choice, trichotomy, or completeness coordinate. □

Theorem 870.17 (Diagonal-limit stability public export)**.** *An accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *packet*

$$T = (R, D, L, S, W, H, C, P, N)$$

together with the selector-seal primitive binding, the window-transport primitive binding, the scoped $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ *boundary, tail-refinement invariance, and cofinal refinement confluence exports a public* $\mathsf{Real}^{\uparrow}$*-completeness-facing route from* $\mathsf{RegSeqRat}^{\uparrow}$ *through* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *to the sealed* $\mathsf{Real}^{\uparrow}$ *row. The export is exactly the displayed route through* $R, D, L, S, W, H, C, P, N$ *and supplies no quotient-stream equality, host real equality, choice-selected representative, trichotomy, or ambient completeness coordinate.*

Proof. First use Definition 870.12 and Definition 870.13. They bind the representative side to the $\mathsf{DiagonalTailSelector}^{\uparrow}$ primitive surface, the seal side to the $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ primitive surfaces, and the comparison route to the finite window and replay rows.

Next apply Definition 870.14. The scoped boundary turns those primitive bindings into the displayed local $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$ surface for the same accepted packet T.

Finally use Definition 870.15 and Definition 870.16. Tail refinement and cofinal comparison preserve the sealed $\mathsf{Real}^{\uparrow}$ route through the same public-facing selector-seal corridor, while the carrier exclusions keep quotient-stream equality, host real equality, selected representatives, trichotomy, and ambient completeness outside the export. □

Theorem 870.18 (Diagonal-limit stability bridge corridor)**.** *An accepted* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *packet*

$$T = (R, D, L, S, W, H, C, P, N)$$

supplies a paper bridge from the public selector-seal export of Definition 870.17 to the downstream $\mathsf{Real}^{\uparrow}$*-completion corridor. The representative side is bound by* $\mathsf{DiagonalTailSelector}^{\uparrow}$*, the seal side is bound by* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, the shared comparison is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, and the replay and provenance rows are exactly the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of* T*. The corridor supplies no quotient-stream equality, host real equality, selected representative, trichotomy, or ambient completeness coordinate.*

Proof. Start with the public selector-seal export of Definition 870.17. It presents the accepted packet as a $\mathsf{Real}^{\uparrow}$-completeness-facing route from the regular source row through the selector and seal rows to the displayed sealed $\mathsf{Real}^{\uparrow}$ row.

Use the primitive bindings to identify the two bridge sides. By Definition 870.12, the representative side is read only through $\mathsf{DiagonalTailSelector}^{\uparrow}$, while the seal side is read only through $\mathsf{CauchyLimitSeal}^{\uparrow}$ together with $\mathsf{Real}^{\uparrow}$. By Definition 870.13, the common $\mathsf{StreamName}^{\uparrow}$ window and the rows H, C, P, N are the whole primitive transport surface.

Now stabilize the corridor under downstream refinement. Apply Definition 870.15 to preserve the sealed $\mathsf{Real}^{\uparrow}$ route under displayed tail refinement, and apply Definition 870.16 to compare cofinal refined reads through the same sealed route. These two stability clauses make the public export usable as a bridge corridor rather than a one-step display.

The exclusions are inherited from the carrier and from the two stability clauses. Since quotient-stream equality, host real equality, selected representatives, trichotomy, and ambient completeness have no coordinate in T, the paper bridge reaches the downstream Real-completion corridor only through the displayed selector, seal, window, replay, provenance, and naming rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiagonalLimitStability}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, diagonal representative row, limit selector row, $\mathsf{Real}^{\uparrow}$ seal row, finite $\mathsf{StreamName}^{\uparrow}$ window, componentwise hsame transports, displayed Cont continuations, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 870.14, Definition 870.17, and Definition 870.18 bind the mature selector-seal route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalLimitStability}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DiagonalLimitStabilityUp.DiagonalLimitStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.DiagonalLimitStabilityUp.taste gate`

Not claimed: This chapter does not prove quotient-stream equality, host real equality, a choice-selected limit representative, trichotomy, ambient completeness, or a machine-checked standard bridge.

Upgrade path: No further paper-theory closure step remains; machine bridge checking is the only later formal-axis movement.

Chapter 871

A Concrete Naming Certificate for $\mathsf{TheorySelfClassifier}^{\uparrow}$

The $\mathsf{TheorySelfClassifier}^{\uparrow}$ packet realizes the own-theory reading of chapter 9454 as a finite classifier surface. It sits beside chapter 865, chapter 856, chapter 868, and chapter 859: those siblings expose boundary, identity, dependency-audit, and self-theory rows, while this chapter packages the row comparison that lets a consumer read the five commitments as one self-theory classifier.

Definition 871.1 (Theory self-classifier carrier)**.** *A* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, E, R, P, A, L, H, C, Q, N)$$

with the following displayed rows:

1. *G is the closed-generator row, naming the finite constructor and declaration inventory used by the theory surface;*
2. *E is the literal equality row, recording that comparison is syntactic equality of the displayed object rows rather than semantic equality in a host calculus;*
3. *R is the structural recursor row, exposing the eliminators attached to the closed generators named by G;*
4. *P is the purity row, recording zero hidden axiom supply and strict proof-ancestry accountability for internal reads;*
5. *A is the apophatic boundary row, recording visible sockets and setup fields for requests outside the internal substrate;*
6. *L is the classifier ledger that compares G, E, R, P, A as one self-theory surface;*
7. *H records componentwise* hsame *transports for generator, equality, recursor, purity, boundary, ledger, route, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes by which consumers read each commitment and then return to the classifier ledger;*
9. *Q is the* Pkg *provenance over chapter 9454, chapter 865, chapter 856, chapter 868, chapter 859,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TheorySelfClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for a classical-CIC parent theory, semantic equality oracle, hidden recursor supply, unrecorded axiom dependency, or boundary row promoted to internal theorem content.

Theorem 871.2 (Theory self-classifier NameCert obligations)**.** *For every* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *carrier*

$$T = (G, E, R, P, A, L, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{TheorySelfClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission, closed-generator exposure, literal-equality exposure, structural-recursor exposure, purity exposure, apophatic-boundary exposure, classifier-ledger comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The ledger row* L *is required to compare the five commitment rows as one self-theory surface before a consumer may export a self-theory reading.*

Proof. Project the finite carrier of Definition 871.1. The rows G, E, R, P, A are the five commitment coordinates, while L, H, C, Q, N provide comparison, transport, replay, provenance, and local naming.

The NameCert obligations for the five commitments are read componentwise. The generator row supplies the closed inventory, the equality row supplies literal comparison, the recursor row supplies only generator-local eliminators, the purity row supplies strict audit visibility, and the apophatic row keeps external supply visible as boundary data.

The classifier obligation is the additional row L. It compares those five rows as one surface, so a consumer cannot cite a self-theory reading by projecting only one commitment or by replacing the row comparison with a sibling chapter.

Since the carrier contains no further coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.TheorySelfClassifierUp.TheorySelfClassifierCarrier namecert obligations`

Theorem 871.3 (Theory self-classifier non-escape)**.** *Every public read from an accepted* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *carrier factors through*

$$G, E, R, P, A, L, H, C, Q, N.$$

A consumer may read the five commitment rows, their classifier ledger, componentwise transports, displayed routes, provenance, and local name. It cannot read a classical-CIC parent theory, a host semantic equality oracle, a hidden recursor, an unrecorded axiom dependency, or an apophatic boundary promoted to internal theorem content.

Proof. Start from the displayed carrier. The only self-theory commitments are G, E, R, P, A, and the only row that compares them as one surface is L.

Routes through C and transports through H preserve those rows componentwise. They can replay a commitment read or move the packet, but they cannot create a new substrate principle.

The provenance and naming rows Q, N cite the sibling surfaces and bind the local certificate. Citation gives a consumer context for the five rows; it does not add a parent theory, host equality, hidden recursor, axiom supply, or internal theorem extracted from a boundary row.

Thus every public read remains inside $T = (G, E, R, P, A, L, H, C, Q, N)$, and the excluded objects have no route through the accepted packet. □

Theorem 871.4 (Theory self-classifier sibling independence)**.** ***Independence witness:*** *Nearest siblings: internaltheoryboundary, metacic-theory-identity, axiomdependencyauditmap, and bedc-selftheory. No one of these is bijective to the* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *carrier because each lacks the*

classifier row L that compares closed generators, literal equality, structural recursors, purity, and apophatic boundary rows as one self-theory surface.

Proof. Compare row inventories. The internal-theory-boundary packet exposes the five commitments as a boundary, the MetaCIC theory-identity packet records identity rows, the axiom-dependency-audit-map packet records mode and supply rows, and the BEDC self-theory packet packages the commitments as a self-theory certificate.

The $\mathsf{TheorySelfClassifier}^{\uparrow}$ carrier requires the same five commitment rows together with the classifier ledger L that compares them as one surface. None of the sibling carriers has that row with the displayed comparison role.

Therefore a consumer may cite those siblings as provenance, but no cited sibling is a bijective replacement for this classifier packet. □

Theorem 871.5 (Theory self-classifier TasteGate target)**.** *The later Lean 'TasteGate.lean' development for* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *must provide a* Nontrivial *witness with two explicit carrier values and a* FieldFaithful *witness by field projection, match analysis, and injection over the displayed rows* $G, E, R, P, A, L, H, C, Q, N$.

Lean4 ✓ `BEDC.Derived.TheorySelfClassifierUp.TheorySelfClassifierTasteGate single carrier alignment`

Proof. The carrier has distinguishable commitment and classifier fields, so a singleton event-flow encoding would not show that the self-theory surface is genuinely classified.

Nontriviality supplies two explicit packets. Field faithfulness then checks the event-flow encoding by projecting each displayed row, matching on the carrier constructor, and using injection to recover equality of all fields from equality of encodings.

These two gate obligations are the Lean-side conditions needed before the classifier can be treated as a checked TasteGate object. □

Falsifiable prediction: A TheorySelfClassifierUp packet cannot export a self-theory reading unless the five commitment rows are all present as displayed BHist coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{TheorySelfClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{TheorySelfClassifier}^{\uparrow}$ packet is generated from closed-generator, literal-equality, structural-recursor, purity, apophatic-boundary, classifier-ledger, transport, route, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheorySelfClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 871.1, Definition 871.2, Definition 871.3, Definition 871.4, and Definition 871.5 specify the seed carrier, finite NameCert surface, non-escape boundary, sibling independence witness, and gate-facing target.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheorySelfClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide Lean-side verification, bridge checking, a classical-CIC embedding, a semantic equality oracle, a hidden axiom route, or a completed TasteGate instance.

Upgrade path: Obligation closure requires Lean encoding of the carrier, checked five-row classifier obligations, non-escape, sibling independence, and a TasteGate route with Nontrivial and FieldFaithful instances.

Chapter 872

A Concrete Naming Certificate for $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$

The $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ packet realizes the self-theory reading of chapter 9454 as a finite reflection loop. It sits beside the theory-identity packet of chapter 856, the closure-trace packet of chapter 795, the kernel-acceptance trace of chapter 557, and the audit-map family of chapter 817. Those siblings expose identity rows, closure rows, acceptance rows, and audit-family rows; this packet binds them into a cyclic replay surface with an explicit boundary-return row.

Definition 872.1 (Self-theory reflection-loop carrier)**.** *A* $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, T, K, A, R, B, H, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{MetaCICTheoryIdentity}^{\uparrow}$ *row that records closed generators, literal AST equality, structural recursors, strict purity, and apophatic boundary as the identity surface of BEDC.MetaCIC;*
2. *T is the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *row that records the finite closure replay consumed by self-theory audit readers;*
3. *K is the* $\mathsf{KernelAcceptanceTrace}^{\uparrow}$ *row that records the kernel-admission route for accepted declarations and theorem rows;*
4. *A is the* $\mathsf{AuditMapFamily}^{\uparrow}$ *row that records which audit-map family member reads each identity, closure, and acceptance coordinate;*
5. *R is the cyclic audit-replay row, routing a public self-theory read back through I, T, K, A before it may be classified as an internal BEDC.MetaCIC read;*
6. *B is the boundary-return row, recording requests that leave the displayed self-theory loop and returning them as named boundary obligations rather than as hidden parent-theory content;*
7. *H records componentwise* hsame *transport for identity, closure, acceptance, audit-family, replay, boundary, provenance, and local naming rows;*
8. *C records displayed* Cont *routes by which consumers inspect the same rows without replacing the loop by a semantic comparison with another calculus;*
9. *Q is the* Pkg *provenance over chapter 9454, chapter 856, chapter 795, chapter 557, chapter 817,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SelfTheoryReflectionLoop}^{\uparrow}}$ *row.*

The carrier has no coordinate for a classical CIC parent theory, no host equality oracle, no erased audit replay, and no unrecorded escape from the self-theory boundary.

Lean4 (definition) `BEDC.Derived.SelfTheoryReflectionLoopUp.SelfTheoryReflectionLoopUp`

Definition 872.2 (Self-theory reflection-loop classifier)**.** *Two* $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ *carriers*

$$L = (I, T, K, A, R, B, H, C, Q, N)$$

and

$$L' = (I', T', K', A', R', B', H', C', Q', N')$$

are classifier-equivalent exactly when their identity, closure-trace, kernel-acceptance, audit-family, cyclic-replay, boundary-return, transport, continuation, provenance, and local naming rows agree through the displayed hsame *and* Cont *data. The classifier does not compare carriers by an external model of CIC, by a hidden normalizer, or by an audit route absent from the displayed packet.*

Theorem 872.3 (Self-theory reflection-loop NameCert obligations)**.** *For every accepted* $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ *carrier*

$$L = (I, T, K, A, R, B, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SelfTheoryReflectionLoop}^{\uparrow}}$ *surface consists of six finite rows:*

1. *self-theory identity exposure through I;*
2. *closure-trace replay exposure through T;*
3. *kernel-acceptance replay exposure through K;*
4. *audit-map family exposure through A;*
5. *cyclic audit-replay exactness through R;*
6. *boundary-return exactness through B, with transport, route, provenance, and naming accountability through H, C, Q, N.*

These rows certify a finite loop packet for self-theory identity, audit replay, and boundary non-escape. They do not certify a comparison theorem with classical CIC, a global normalization theorem, or an undisplayed parent theory.

Proof. Project the finite carrier of Definition 872.1. The rows I, T, K, A are the sibling-facing inputs, while R and B are the loop-specific rows.

Classifier exactness follows from Definition 872.2. Agreement is checked on the displayed identity, trace, acceptance, audit-family, replay, and boundary rows, then transported only through H and read only through C.

The provenance and local naming rows Q, N package the same surface. Since the carrier has no additional coordinate, the six listed rows exhaust the local NameCert obligations. □

Theorem 872.4 (Self-theory reflection-loop non-escape)**.** *Every public read of an accepted* $\mathsf{SelfTheoryReflectionLoop}$ *carrier factors through*

$$I, T, K, A, R, B, H, C, Q, N.$$

A read may replay the self-theory identity row through closure, kernel acceptance, and audit-family rows, but a request for an external parent calculus, host semantic equality, hidden audit replay, or unrecorded theorem supply lands in the boundary-return row B.

Proof. Read the consumer route through C. If the route reaches I, T, K, A, R, H, Q, N, it remains inside the displayed reflection loop and its recorded sibling provenance.

If the route asks for material outside those rows, the carrier offers only B as a boundary-return coordinate. The replay row R must pass through I, T, K, A before classification, so deletion of audit replay leaves no route by which the identity row alone can certify the full loop.

Thus a boundary request remains a boundary request. It cannot be reclassified as external parenthood, host equality, hidden audit replay, or undisplayed theorem supply. □

Theorem 872.5 (Self-theory reflection-loop TasteGate follow-up)**.** *The Lean-side follow-up for* $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ SelfTheoryReflectionLoop}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ SelfTheoryReflectionLo}$ *in 'lean4/BEDC/Derived/SelfTheoryReflectionLoopUp/TasteGate.lean'. Nontriviality rules out a vacuous loop carrier, and field faithfulness records that the gate preserves the identity, closure, acceptance, audit-family, cyclic-replay, and boundary-return fields as distinct rows.*

Proof. The carrier has four sibling-facing rows and two loop-specific rows. A gate that preserves only an event-flow shell would not show that these coordinates remain distinguishable.

Nontriviality supplies a carrier with visible content, while field faithfulness preserves the displayed coordinates through the gate. Those are the gate-facing obligations needed before the reflection loop can be read as a checked TasteGate object. □

Falsifiable prediction: If the loop is faithful, deleting the audit-replay row breaks the NameCert obligation surface while preserving the bare theory-identity carrier row.

Independence witness: metacic-theory-identity, metacicclosuretrace, kernelacceptancetrace, audit-map-family: the loop has a cyclic replay row and a boundary-return row absent from each sibling, so no bijection preserves all carrier coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{SelfTheoryReflectionLoop}^{\uparrow}}$.

Constructive story: A $\mathsf{SelfTheoryReflectionLoop}^{\uparrow}$ packet is generated from finite BHist rows for theory identity, reflected theory readback, cyclic audit replay, boundary return, transports, continuations, provenance, and local naming. The reflection loop is not a host self-interpretation; it is a displayed route where Cont replays the audit row and returns to the named boundary before any consumer can cite reflection. Componentwise hsame transports identity, replay, and return rows without deleting the audit surface, while Pkg records the MetaCIC identity, closure trace, kernel acceptance, and audit-map dependencies. The loop names finite reflective routing, not unrestricted truth.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SelfTheoryReflectionLoop}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the self-theory reflection-loop carrier, cyclic replay row, boundary-return row, falsifiable prediction, and sibling independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SelfTheoryReflectionLoop}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a global truth predicate, unrestricted self-interpretation, host reflection equality, audit erasure, or kernel-level self-certification.

Upgrade path: Obligation closure requires Lean encodings of the loop carrier, audit replay, boundary return, transport stability, and non-bijection witness.

Chapter 873

A Concrete Naming Certificate for $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$

The $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ packet isolates the literal AST equality commitment of chapter 9454 as its own finite NameCert boundary. It sits beside the five-row own-theory packet of chapter 865, the broader theory-identity packet of chapter 856, and the dependency audit surface of chapter 868. Those siblings package multiple identity or dependency commitments; this chapter names the single row that says equality is displayed AST equality and not beta equality, conversion equality, quotient equality, or host semantic equivalence.

Definition 873.1 (Literal AST equality-boundary carrier)**.** *A* $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, T, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the displayed AST row, naming constructor tags, binder positions, variable rows, and immediate subterm rows exposed by the BEDC.MetaCIC syntax surface;*
2. *T is the literal comparison row, recording equality only by identical displayed constructor tags and componentwise equality of displayed subrows;*
3. *B is the boundary-refusal row for beta equality, conversion equality, quotient equality, proof irrelevance, proposition extensionality, host semantic equivalence, and parent-theory identity transport;*
4. *H records componentwise* hsame *transport for the AST row, literal comparison row, boundary-refusal row, continuation routes, provenance row, and local naming row;*
5. *C records displayed* Cont *routes by which a consumer reads literal equality and boundary refusal without adding a conversion oracle;*
6. *P is the* Pkg *provenance over chapter 9454, chapter 865, chapter 856, chapter 868, BEDC.MetaCIC AST rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{LiteralASTEqualityBoundary}^{\uparrow}}$ *row.*

The carrier contains no beta-step evaluator, no conversion closure, no quotient of syntax, no external equality oracle, and no theorem identifying displayed AST equality with a semantic equality from another calculus.

Definition 873.2 (Literal AST equality-boundary classifier)**.** *Two* $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ *carriers*

$$L = (A, T, B, H, C, P, N)$$

and

$$L' = (A', T', B', H', C', P', N')$$

are classifier-equivalent exactly when the displayed AST rows, literal comparison rows, boundary-refusal rows, componentwise transports, continuation routes, provenance rows, and local naming rows agree through the displayed hsame *and* Cont *data. The classifier does not compare carriers by beta-normal form, definitional conversion, quotient representative, proof irrelevance, proposition extensionality, or host semantic equality.*

Theorem 873.3 (Literal AST equality-boundary NameCert obligations)**.** *For every* $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ *carrier*

$$L = (A, T, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LiteralASTEqualityBoundary}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed packet of Definition 873.1;*
2. *classifier stability under the literal classifier of Definition 873.2;*
3. *literal equality exactness through* A *and* T*;*
4. *boundary refusal through* B *for every beta, conversion, quotient, proof-irrelevance, proposition-extensionality, host-semantic, or parent-theory equality request;*
5. *ledger exactness through* H, C, P, N *for the same displayed rows.*

Proof. Project the finite carrier. The rows A, T, B are the only equality-facing coordinates before transport, route replay, provenance, or local naming is read.

Classifier stability is componentwise by Definition 873.2. A transported packet may preserve literal AST equality, but it cannot replace that comparison by beta equality, conversion equality, quotient equality, or host semantic equality.

The rows H, C, P, N make the same equality boundary readable by downstream consumers. Since the carrier has no evaluator, conversion closure, quotient, proof-irrelevance row, proposition-extensionality row, or parent-theory equality row, the listed NameCert obligations exhaust the packet surface. □

Theorem 873.4 (Literal AST equality exactness)**.** *Every equality read from an accepted* LiteralASTEqualityBoundar *carrier*

$$L = (A, T, B, H, C, P, N)$$

factors through the displayed AST row A*, the literal comparison row* T*, and the componentwise transport and replay rows* H, C*. The read exports only constructor-tag equality and componentwise displayed subrow equality, and it exports no beta equality, conversion equality, quotient equality, proof irrelevance, proposition extensionality, host semantic equivalence, or parent-theory identity transport.*

Proof. Begin with the carrier definition. The equality coordinate is T, and it is attached to the displayed AST row A.

Transport and replay are componentwise. The rows H and C can move or read A and T, but they do not add a computation relation or quotient relation.

The boundary-refusal row B accounts for every stronger equality request. Therefore any accepted equality read remains literal and displayed, with no route to the excluded equality principles. □

Theorem 873.5 (Literal AST equality sibling independence)**.** ***Independence witness:*** *internaltheoryboundary, metacic-theory-identity, axiom-dependency-audit-map, bedcselftheory: the new carrier isolates one literal-equality boundary row rather than packaging all theory-identity commitments or all axiom-dependency routes.*

Proof. Compare row inventories. The internal-theory and MetaCIC theory-identity packets package closed generators, literal AST equality, structural recursors, purity, and apophatic boundary data together.

The axiom-dependency audit map packages claim, dependency, route, and audit rows. The BEDC self-theory packet packages the five positive own-theory commitments as one substrate-facing certificate.

The $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ carrier keeps only the displayed AST row, literal comparison row, boundary-refusal row, and structural ledger rows. It can be cited by those broader siblings, but none of them is bijective to this single equality-boundary packet. □

Theorem 873.6 (Literal AST equality TasteGate obligation)**.** *The Lean-side 'TasteGate.lean' development for* $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ *includes explicit* Nontrivial *and* FieldFaithful *instances. These instances are the gate-facing rows that separate a real literal-equality boundary carrier from a vacuous event-flow encoding and record that the encoding preserves the displayed AST row, literal comparison row, boundary-refusal row, transport rows, route rows, provenance row, and local naming row.*

Lean4 ✓ `BEDC.Derived.LiteralASTEqualityBoundaryUp.LiteralASTEqualityBoundaryTasteGate single carrier alignment`

Proof. The carrier has distinct equality-facing fields. A gate that recognizes only an event-flow round trip would not show that A, T, and B remain separate rows.

Nontriviality rules out the empty reading of the packet. Field faithfulness records preservation of the displayed field structure, so the literal equality boundary remains a finite carrier with inspectable rows rather than an unstructured event name. □

Falsifiable prediction: No accepted packet may replace literal AST equality by beta/conversion equality without leaving the carrier rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LiteralASTEqualityBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$ packet is generated from displayed AST rows, literal comparison rows, boundary-refusal rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LiteralASTEqualityBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 873.1, Definition 873.2, Definition 873.3, Definition 873.4, Definition 873.5, and Definition 873.6 specify the literal AST equality boundary packet and its gate-facing obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LiteralASTEqualityBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove beta soundness, conversion decidability, quotient soundness, proof irrelevance, proposition extensionality, host semantic equality, Lean verification, or bridge checking for $\mathsf{LiteralASTEqualityBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, literal equality exactness, boundary refusal, ledger exactness, and TasteGate nontriviality with field faithfulness.

Chapter 874

A Concrete Naming Certificate for $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$

The $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ packet names the concrete route from regular-Cauchy finite windows to the observation budget and the $\mathsf{Real}^{\uparrow}$ seal used by chapter 9238. It reads its finite window side through chapter 126, its diagonal schedule through chapter 422, and its budget side through chapter 611. The packet keeps these sibling surfaces in one route rather than replacing any of them.

Definition 874.1 (Regular-Cauchy finite observation route carrier)**.** *A* $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (Q, W, D, R, B, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite regular-Cauchy request row, including the requested precision and the selected diagonal index;*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row read by the route;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for the window;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the finite observations;*
5. *B is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row admitting the same window, dyadic ledger, handoff, and seal request;*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the budget row has been displayed;*
7. *H stores componentwise* hsame *transports for $Q, W, D, R, B, S, C, P, N$;*
8. *C stores the* Cont *routes reading the request, window, dyadic ledger, regular-rational handoff, observation budget, seal row, provenance, and local name;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host real equality, hidden limit selection, unbounded observation schedules, or a $\mathsf{Real}^{\uparrow}$ *seal read outside the displayed budget.*

Lean4 ✓ BEDC.Derived.RegularCauchyFiniteObservationRouteUp.RegularCauchyFiniteObservationRouteTasteGate
single carrier alignment

Theorem 874.2 (Regular-Cauchy finite observation route obligations)**.** *For the carrier of Definition 874.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{RealObservationBudget}^{\uparrow}$ *budget admission,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the tuple $Z = (Q, W, D, R, B, S, H, C, P, N)$. The first six rows expose the finite request, selected window, dyadic ledger, regular-rational handoff, observation budget, and seal row.

The sibling surfaces supply the route checks. The $\mathsf{StreamName}^{\uparrow}$ row gives the finite observation window, the $\mathsf{DyadicRatCore}^{\uparrow}$ row gives the finite arithmetic readback, and the $\mathsf{RegSeqRat}^{\uparrow}$ row is the handoff consumed by the observation budget.

The budget and seal obligations are bound by B, S, H, C, P, N. Since the seal row is downstream of B in the carrier and in the displayed continuation routes, the local certificate has no source from which to read a $\mathsf{Real}^{\uparrow}$ seal before the finite observation budget is present. □

Theorem 874.3 (Regular-Cauchy finite observation route non-escape)**.** *An accepted* RegularCauchyFiniteObservatic *carrier exports no quotient stream, host real equality proof, hidden limit selector, unbounded observation schedule, or direct* $\mathsf{Real}^{\uparrow}$ *seal read. Every completion-facing request remains inside the displayed rows* $Q, W, D, R, B, S, H, C, P, N$.

Proof. Inspect the carrier inventory in Definition 874.1. The tuple lists the request, window, dyadic ledger, regular-rational handoff, observation budget, seal row, transports, continuations, provenance, and local naming data, and it lists none of the excluded objects.

Apply the obligation theorem. It makes the displayed rows the whole local NameCert surface, so a consumer cannot add quotient-stream equality, host real equality, hidden limit selection, an unbounded schedule, or an early seal coordinate while remaining in the accepted carrier. □

Theorem 874.4 (Regular-Cauchy finite observation route Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ *carrier factors through the finite route*

$$Q \to W \to D \to R \to B \to S$$

with transport, continuation, provenance, and local naming rows H, C, P, N. *Thus the seal is reached only after the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row has accepted the same* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Begin with the carrier. The seal coordinate S appears after the window, dyadic ledger, handoff, and observation-budget rows in the displayed packet.

Read the observation-budget side through chapter 611. That sibling surface admits seal consumption only through finite windows, dyadic readbacks, regular-rational handoff rows, and its own budget row.

The rows H, C, P, N record the componentwise transport, route replay, package provenance, and local name for the same packet. Therefore every $\mathsf{Real}^{\uparrow}$ seal read follows the displayed route through B and cannot bypass the finite observation budget. □

Theorem 874.5 (Regular-Cauchy finite observation route vision concretization)**.** *The diagonal-limit route named in chapter 9238 is realized by accepted* $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ *packets as finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal data. The stationary diagonal side is read through chapter 422, while product-facing consumers can share the finite window and handoff rows with chapter 464 without importing quotient streams, hidden limits, or host real equality.*

Proof. The diagonal-limit consumer in chapter 9238 needs a finite observation route before a $\mathsf{Real}^{\uparrow}$ seal is inspected. The carrier supplies exactly that route: request, window, dyadic readback, regular-rational handoff, observation budget, and seal.

The stationary diagonal sibling fixes the diagonal-window shape, and the CauchyProduct sibling can reuse the same kind of finite window and regular-rational handoff rows. These links do not add coordinates to Z; they only identify which sibling surfaces read the displayed rows.

Finally apply the Real seal handoff and non-escape theorems. Together they show that the vision-facing route reaches the $\mathsf{Real}^{\uparrow}$ seal through the finite observation budget and excludes quotient streams, hidden limit selection, and host real equality. □

Theorem 874.6 (Regular-Cauchy finite observation route budget-selector pullback)**.** *Let*

$$Z = (Q, W, D, R, B, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ *packet, and let*

$$F = (B_0, S_0, W_0, D_0, R_0, E_0, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet. If the displayed* hsame *transports identify* W *with* W_0*,* D *with* D_0*,* R *with* R_0*, and the route seal* S *with* E_0*, then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer factors through both finite routes*

$$Q \to W \to D \to R \to B \to S$$

and

$$B_0 \to S_0 \to W_0 \to D_0 \to R_0 \to E_0.$$

The common consumer exports no quotient stream, hidden limit selector, host real equality, countable choice, or ambient completeness row.

Proof. Start on the regular-Cauchy route. By Definition 874.4, every $\mathsf{Real}^{\uparrow}$ seal read of Z passes through the displayed finite path $Q \to W \to D \to R \to B \to S$.

Move to the selector route. Apply Definition 782.7 to the packet F, obtaining the finite path $B_0 \to S_0 \to W_0 \to D_0 \to R_0 \to E_0$. The assumed componentwise hsame transports identify the shared window, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and seal endpoint.

Finally apply Definition 782.14 together with the non-escape theorem of this chapter. These two boundaries remove quotient streams, hidden limit selectors, host real equality, countable choice, and ambient completeness data, so the shared $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can read only the two displayed finite routes and their visible transport, continuation, provenance, and naming rows. □

Theorem 874.7 (Regular-Cauchy finite observation route interface lock)**.** *Let*

$$Z = (Q, W, D, R, B, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ *packet, and let*

$$O = (S_0, R_0, D_0, T_0, E_0, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{FiniteObservationInterface}^{\uparrow}$ *packet. If the displayed* hsame *transports identify* W *with* S_0*,* R *with* R_0*,* D *with* D_0*, the selected finite window read by* W *with* T_0*, and the* $\mathsf{Real}^{\uparrow}$ *seal endpoint* S *with the Real-facing endpoint routed by* C_0*, then every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer factors through both concrete finite routes*

$$Q \to W \to D \to R \to B \to S$$

and

$$S_0 \to R_0 \to D_0 \to T_0 \to E_0.$$

The omitted-tail ledger E_0 *remains a tail ledger only: it cannot be exported as a selected stream row, quotient stream equality, host real equality, hidden limit selection, or ambient completeness witness.*

Proof. First apply the Real seal handoff theorem for Z. It places every $\mathsf{Real}^{\uparrow}$ read behind the displayed route $Q \to W \to D \to R \to B \to S$, with H, C, P, N carrying only transport, replay, provenance, and local naming data.

Next apply the budget-selector pullback to the same route boundary. This keeps $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers on the concrete finite observation side and excludes quotient streams, hidden limit selectors, host real equality, countable choice, and ambient completeness.

Now read the interface packet through Definition 1007.3 and Definition 1007.2. The selected finite window is T_0, while E_0 is the omitted-tail ledger and has no selected-stream role. The assumed componentwise hsame transports identify this interface window, regular-rational handoff, dyadic tolerance row, and Real-facing endpoint with the corresponding rows of Z.

The two route descriptions therefore name the same consumer boundary through the displayed packet rows. Since neither local $\mathsf{NameCert}$ surface contains a coordinate for exporting the omitted tail as selected data, quotient equality, host equality, hidden limit choice, or ambient completeness, the shared consumer has no such escape route. □

Falsifiable prediction: Every accepted route factors through the displayed finite observation budget before any RealUp seal read; a seal read bypassing the budget refutes this chapter.

Independence witness: real-observation-budget, finiteobservationbudgetselector, stationarypartitiondiagonal, cauchyproduct, streamname: none is bijective because this carrier simultaneously records the route from regular-Cauchy finite windows to the observation budget and RealUp seal, while each sibling lacks at least one of the route, budget, diagonal, product, or stream-only rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}$ packet is generated from a finite regular-Cauchy request, selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{RealObservationBudget}^{\uparrow}$ budget row, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 874.1, Definition 874.2, Definition 874.3, Definition 874.4, Definition 874.5, Definition 874.6, and Definition 874.7 define the route carrier, obligation surface, non-escape boundary, Real seal handoff, vision-facing concretization, budget-selector pullback, and finite-observation interface lock.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteObservationRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient-stream equality, hidden limit selection, unbounded observation schedules, host real equality, RealUp completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window admission, dyadic readback, RegSeqRat handoff, observation-budget exactness, seal-handoff exactness, and non-escape for the same route packet.

Chapter 875

A Concrete Naming Certificate for $\mathsf{CrossHistCausalRoute}^{\uparrow}$

The $\mathsf{CrossHistCausalRoute}^{\uparrow}$ interface realizes the route layer of the Inter-Hist Locality chain of chapter 9419. It sits after the observer-Hist identity surface of chapter 479, the causal witness surface of chapter 478, the maximum-rate boundary of chapter 594, the invariant observer-gate surface of chapter 656, and the finite replay surface of chapter 469. Its role is to bind those five siblings into one finite route certificate before a later causal-rate packet reads the maximum-rate row.

Definition 875.1 (Cross-Hist causal-route carrier)**.** *A* $\mathsf{CrossHistCausalRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, D, M, S, G, T, H, C, P, N)$$

with the following visible rows:

1. *A and B are the two observer-history slots, each admitted through* $\mathsf{ObserverHistIdentity}^{\uparrow}$*;*
2. *D is the displayed finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *causal rows over the ordered observer slots and selected unary slot ledgers;*
3. *M is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row attached to exactly the causal family D;*
4. *S is the observer-symmetry gate, recording that the route family remains readable after swapping the two observer slots through the displayed transports;*
5. *G is the observer non-escape gate, refusing a global frame, synchronized clock, metric tensor, or host-valued speed for the route read;*
6. *T is the* $\mathsf{ContinuationTrace}^{\uparrow}$ *row that replays the accepted route from the observer slots through D, M, S, and G;*
7. *H records componentwise* hsame *transport for the observer slots, causal rows, maximum-rate row, observer gates, continuation trace, package row, and local naming row;*
8. *C records the displayed* Cont *routes by which a consumer reads $A, B, D, M, S, G, T, H, P, N$;*
9. *P is the* Pkg *provenance over* $\mathsf{CrossHistCausalRoute}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*,* $\mathsf{CrossHistCausal}^{\uparrow}$*,* $\mathsf{MaxCausalRate}^{\uparrow}$*,* $\mathsf{InterHistInvariant}^{\uparrow}$*,* $\mathsf{ContinuationTrace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}$ *row.*

The carrier is a finite route packet. It contains no global observer frame, synchronized clock, metric tensor, host real-valued speed, detached maximum-rate read, or causal witness outside the displayed route family.

Theorem 875.2 (Cross-Hist causal-route NameCert obligations)**.** *For the carrier of Definition 875.1, the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (A, B, D, M, S, G, T, H, C, P, N),$$

observer-slot admission, displayed causal-row replay, maximum-rate attachment to D*, both observer-gate rows* S, G*, continuation-trace replay through* T*, componentwise* hsame *transport, displayed* Cont *access, package provenance, and the local naming row.*

Proof. Carrier admission projects the finite tuple in Definition 875.1. The rows $A, B, D, M, S, G, T, H, C, P, N$ are the complete public inventory of the route packet.

The observer slots are read through A and B, and the causal rows are exactly the displayed family D. The maximum-rate row M is attached to that same family, so no consumer can read a detached maximum over unlisted causal data.

The two observer gates are S and G. The row S supplies the observer-swap route, while G refuses global synchronization, privileged frames, metric tensors, and host-valued speeds.

The continuation trace T, transport row H, displayed routes C, provenance P, and local naming row N replay and name the same finite route. Since the carrier has no coordinate outside those rows, the listed obligations exhaust the $\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalRouteUp.CrossHistCausalRouteCarrier namecert obligations`

Theorem 875.3 (Cross-Hist causal-route maximum-rate gate)**.** *An accepted* $\mathsf{CrossHistCausalRoute}^{\uparrow}$ *packet cannot expose a maximum-rate read unless the displayed causal-route family* D *and both observer-gate rows* S, G *are present in the same carrier.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalRouteUp.CrossHistCausalRouteCarrier maximum rate gate`

Proof. Project the accepted carrier through Definition 875.1. The only maximum-rate coordinate is M, and its definition attaches it to the displayed causal family D.

Apply Definition 875.2. The local route obligations require the observer-symmetry gate S and the observer non-escape gate G before the maximum-rate row is exposed to a consumer.

Therefore a read that omits D, S, or G is not a read of the accepted route packet. The packet supplies no second rate row, hidden causal family, global frame, or synchronized clock through which such a read could be recovered. □

Theorem 875.4 (Cross-Hist causal-route sibling independence)**.** *The* $\mathsf{CrossHistCausalRoute}^{\uparrow}$ *packet stores all five sibling surfaces:* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *for observer slots,* $\mathsf{CrossHistCausal}^{\uparrow}$ *for causal rows,* $\mathsf{MaxCausalRate}^{\uparrow}$ *for the bound row,* $\mathsf{InterHistInvariant}^{\uparrow}$ *for observer-gate reading, and* $\mathsf{ContinuationTrace}^{\uparrow}$ *for finite replay. It is not bijective to any one of those siblings, because deleting any one displayed surface removes a row required by Definition 875.1.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalRouteUp.CrossHistCausalRoute sibling independence`

Proof. Inspect the carrier coordinates. The rows A, B carry the observer slots, D carries the causal family, M carries the maximum-rate row, S, G carry the observer-gate boundary, and T carries route replay.

Each listed sibling supplies a different row kind consumed by the route packet. Removing the observer slots leaves no anchors for D; removing D leaves no causal family for M; removing M leaves no maximum-rate route; removing S, G leaves no observer-gate boundary; removing T leaves no finite replay.

Thus the route packet is a finite product of the five displayed surfaces with transport, access, provenance, and naming rows. No projection to a single sibling can be inverted without recreating the deleted rows. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalRouteUp.CrossHistCausalRouteCarrier sibling independence`

Theorem 875.5 (Cross-Hist causal-route TasteGate surface)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{CrossHistCausalRoute}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ CrossHistCausalRoute}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ CrossHistCausalRoute}^{\uparrow}$ *instances. These instances must preserve the observer slots, causal-route family, maximum-rate row, observer-gate rows, continuation trace, transport rows, displayed access routes, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalRouteUp.CrossHistCausalRouteTasteGate single carrier alignment`

Proof. The nontriviality instance is forced by the carrier inventory: an admitted route packet must display at least the two observer slots, one route-family surface, the maximum-rate row, both observer gates, and the replay row.

Field faithfulness is likewise rowwise. A gate-facing instance that forgets any of $A, B, D, M, S, G, T, H, C, P, N$ would permit a consumer read not backed by the local $\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}$ obligations.

Therefore the TasteGate surface must preserve exactly the finite packet named in Definition 875.1. □

Falsifiable prediction: An accepted route packet cannot expose a maximum-rate read unless the displayed causal-route family and both observer-gate rows are present.

Independence witness: observerhistidentity, crosshistcausal, maxcausalrate, interhistinvariant, continuationtrace: route packets store all five surfaces and are not bijective to any single sibling.

Closure status for $\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{CrossHistCausalRoute}^{\uparrow}$ packet is generated from two observer-history slots, finite causal-route rows, a maximum-rate row, two observer-gate rows, a continuation-trace row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 875.1, Definition 875.2, Definition 875.3, Definition 875.4, and Definition 875.5 specify the finite route packet, its local obligations, its maximum-rate gate, its sibling dependence boundary, and its TasteGate instance surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CrossHistCausalRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive a numerical speed constant, Lorentz geometry, metric tensors, synchronized clocks, global frames, Lean verification, or bridge checking for $\mathsf{CrossHistCausalRoute}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, route replay, maximum-rate attachment, observer-gate preservation, continuation-trace replay, and TasteGate nontriviality with field faithfulness.

Chapter 876

A Concrete Naming Certificate for $\mathsf{CrossHistCausalRate}^{\uparrow}$

The $\mathsf{CrossHistCausalRate}^{\uparrow}$ packet realizes the maximum cross-Hist causal-rate direction as a finite NameCert carrier. It sits after the two-history causal witness surface of chapter 478 and the unary bound surface of chapter 594, while exposing the observer-symmetric invariant reading used by chapter 656. The packet also consumes observer-Hist identity from chapter 479 and finite route replay from chapter 469: those siblings provide the observer rows and displayed continuation routes, while this chapter binds them to a maximum-rate read with no global frame.

Definition 876.1 (Cross-Hist causal-rate carrier)**.** *A* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, B, D, M, S, G, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two observer-history slots, each read through the observer-Hist identity surface;*
2. *D is the displayed causal-dependence route family, whose entries are finite* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows over the ordered pair A, B and selected unary slots;*
3. *M is the maximum-rate row, read as a* $\mathsf{MaxCausalRate}^{\uparrow}$ *unary-bound packet over exactly the dependence routes in D;*
4. *S is the observer-symmetry row, recording that swapping the two Hist slots preserves the same maximum-rate read after displayed transport;*
5. *G is the no-global-frame ledger, recording absence of a global synchronization row, shared clock row, metric tensor row, or privileged observer-frame row;*
6. *H records componentwise* hsame *transport for the two Hist slots, dependence routes, maximum-rate row, observer-symmetry row, no-global-frame ledger, continuation routes, provenance row, and local naming row;*
7. *C records displayed* Cont *routes by which a consumer reads the two histories, replays each causal-dependence route, reads the maximum-rate row, checks observer symmetry, and checks the no-global-frame ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{CrossHistCausalRate}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *row.*
The carrier contains no coordinate for a global observer frame, synchronized clock, metric tensor, host real-valued speed, unlisted causal witness, or maximum-rate read detached from the displayed dependence routes.

Theorem 876.2 (Cross-Hist causal-rate NameCert obligations)**.** *For the carrier of Definition 876.1, the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (A, B, D, M, S, G, H, C, P, N),$$

two observer-history slots, finite causal-dependence route replay, maximum-rate row exactness, observer-symmetry preservation, no-global-frame refusal, componentwise hsame *transport, displayed* Cont *replay, package provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalRateUp.CrossHistCausalRateCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 876.1. The rows $A, B, D, M, S, G, H, C, P, N$ expose every public coordinate.

The observer slots are read through A and B, with observer-Hist identity supplied by the sibling surface. The dependence routes in D are finite $\mathsf{CrossHistCausal}^{\uparrow}$ rows, and the continuation routes in C replay those rows without adding a hidden causal witness.

The maximum-rate read is the row M. It is attached to exactly the displayed dependence route family D, so the rate surface cannot detach from the two histories or from the selected unary slots.

The observer-symmetry row S and the no-global-frame ledger G provide the two public gates for a maximum-rate read. The remaining rows H, C, P, N transport, replay, package, and name the same packet, so the NameCert obligations are exhausted by the displayed carrier. □

Theorem 876.3 (Cross-Hist causal-rate carrier admission row)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet exposes exactly the finite carrier admission row*

$$A, B, D, M, S, G, H, C, P, N.$$

The rows are respectively the two observer-history slots, the causal-dependence route family, the maximum-rate row, the observer-symmetry row, the no-global-frame ledger, componentwise hsame *transport, displayed* Cont *replay, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *row. No root consumer may add a global frame, synchronized clock, metric tensor, host speed coordinate, hidden causal witness, or detached rate row to this admission surface.*

Proof. Project the carrier in Definition 876.1. Its tuple is exactly $R = (A, B, D, M, S, G, H, C, P, N)$, and the definition lists no further public coordinate.

Use the NameCert obligations of Definition 876.2. Carrier admission for the same tuple is the first obligation, while the remaining obligations classify every displayed row as observer slots, route replay, maximum-row exactness, symmetry, frame refusal, transport, continuation, provenance, or naming.

The exclusion clause is then an inventory argument. Since the accepted packet contains no coordinate for a global frame, synchronized clock, metric tensor, host speed, hidden witness, or detached rate read, no root admission row can expose one. □

Theorem 876.4 (Cross-Hist causal-rate route replay row)**.** *Every public rate read from an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet replays the finite route family D through the displayed continuation*

row C before it reaches the maximum-rate row M. The route reads the $\mathsf{CrossHistCausal}^{\uparrow}$ *sibling surface of chapter 478 and then the* $\mathsf{MaxCausalRate}^{\uparrow}$ *sibling surface of chapter 594; it cannot read M as a host-side rate detached from D.*

Proof. Start with Definition 876.1. The dependence coordinate is D, the continuation coordinate is C, and the maximum-rate coordinate is M.

Apply Definition 876.2. Its causal-dependence route replay and maximum-rate row exactness clauses force the read of M to remain attached to the displayed family D under the same local naming row.

Compose this with the existing route surfaces. The $\mathsf{CrossHistCausal}^{\uparrow}$ rows supply the finite causal witnesses consumed by D, and the $\mathsf{MaxCausalRate}^{\uparrow}$ rows supply the unary-bound rate surface consumed by M. Thus the root route is D through C before M, with no detached host-rate coordinate. □

Theorem 876.5 (Cross-Hist causal-rate frame-refusal row)**.** *Any public rate read from an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet must consume both the observer-symmetry row S and the no-global-frame ledger G before the read can be used by* $\mathsf{InterHistInvariant}^{\uparrow}$ *or* $\mathsf{MaxCausalRate}^{\uparrow}$ *consumers. The row refuses a global frame, synchronized clock, metric tensor, and host speed coordinate.*

Proof. Project the carrier of Definition 876.1. The only public frame-facing rows are S and G: S records observer symmetry, and G records absence of a global synchronization surface.

Use Definition 876.2. Observer-symmetry preservation and no-global-frame refusal are obligations of the same local $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ surface as the maximum-rate row exactness clause.

Now compose the invariant and consumer boundaries. By Definition 876.9 and Definition 876.10, an invariant-facing rate read cannot proceed without S and G. By Definition 876.11, every consumer route stays inside the displayed rows, so neither an $\mathsf{InterHistInvariant}^{\uparrow}$ consumer nor a $\mathsf{MaxCausalRate}^{\uparrow}$ consumer can introduce a global frame, clock, metric, or host speed coordinate. □

Theorem 876.6 (Cross-Hist causal-rate observer-slot route)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet reads observer slots only through the displayed rows*

$$A, B, D, H, C, P, N.$$

The rows A and B supply the two observer-history slots, D supplies the finite causal-dependence route family over those slots, and H, C, P, N supply the componentwise transport, displayed continuation replay, package provenance, and local naming rows. No observer-facing consumer may read an unlisted Hist slot, synchronized clock, privileged observer frame, or causal route outside this displayed family.

Proof. Project the carrier of Definition 876.1. The only observer-history coordinates are A and B, and the only causal rows attached to them are the finite route entries in D.

Use the NameCert obligations of Definition 876.2. They require observer-history slots, causal-dependence route replay, componentwise hsame transport, displayed Cont replay, package provenance, and local naming to be read from the same packet.

Thus every observer-slot read is routed through A, B, D, H, C, P, N, with no carrier coordinate left for a hidden observer slot or external frame. □

Theorem 876.7 (Cross-Hist causal-rate maximum-row exactness)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet*

$$R = (A, B, D, M, S, G, H, C, P, N),$$

the maximum-rate row M is attached to the displayed dependence route family D. A consumer may not read M as a host real-valued speed, metric row, detached unary bound, or maximum over causal data not displayed in D.

Proof. Project Definition 876.1. The row M is declared as the $\mathsf{MaxCausalRate}^{\uparrow}$ unary-bound packet over exactly the dependence routes in D.

Apply Definition 876.2. Its maximum-rate exactness obligation keeps the rate row inside the same carrier as the two observer histories, the finite causal route family, and the structural rows.

Since no additional maximum-rate coordinate occurs in the packet, the only available rate read is the one attached to D. □

Theorem 876.8 (Cross-Hist causal-rate symmetry non-escape)**.** *Any invariant-facing consumer of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet may read the rate only after both the observer-symmetry row S and the no-global-frame ledger G are present. The read is therefore an observer-symmetric finite maximum over the displayed route family D, not a global-frame synchronization, metric tensor, privileged observer surface, or host speed.*

Proof. Begin with the carrier projection. The rows S and G are distinct public coordinates of the packet, with S recording swap-invariance of the two Hist slots and G recording the absence of a global synchronization surface.

Use Definition 876.2 to keep observer-symmetry preservation and no-global-frame refusal as obligations of the same local $\mathsf{NameCert}$ row. A consumer that omits either row has not consumed the displayed certificate surface.

Finally combine the maximum-rate row with those two gates. The invariant-facing read can only use M through D together with S, G, H, C, P, N, so it cannot escape to a global frame, metric tensor, privileged observer, or host-level speed. □

Theorem 876.9 (Cross-Hist causal-rate invariant boundary)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet reads its maximum-rate row as an observer-symmetric inter-Hist invariant over the two displayed history slots. A consumer may reach the rate only through $A, B, D, M, S, G, H, C, P, N$, and it may not read a maximum-rate value when either the observer-symmetry row S or the no-global-frame ledger G is absent.*

Proof. Project the accepted carrier. The two histories are the rows A and B, and all causal content is the finite route family D.

Read the maximum-rate row M through the same route family. Since M is specified over the displayed dependence routes, it is not a host real-valued speed and not a metric row.

Now apply the observer-symmetry row S. It records that swapping A and B preserves the same maximum-rate read after componentwise transport, matching the finite invariant surface of chapter 656.

Finally inspect the no-global-frame ledger G. Without G, the read could not certify that the rate is independent of a privileged synchronization surface; without S, it could not certify observer symmetry. Hence a public maximum-rate read requires both rows. □

Theorem 876.10 (Cross-Hist causal-rate no-global-frame boundary)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet*

$$R = (A, B, D, M, S, G, H, C, P, N),$$

any public rate read consumes the no-global-frame ledger row G together with the two observer-history slots, causal-dependence route family, maximum-rate row, observer-symmetry row, and structural rows. No consumer route may read the maximum-rate row as a synchronized clock, privileged observer-frame row, metric tensor, or global-frame coordinate.

Proof. Project the accepted carrier. The rate-facing coordinates are $A, B, D, M, S, G, H, C, P, N$, and the row G is the only displayed ledger that records the absence of a global synchronization surface.

Apply Definition 876.2. Its no-global-frame refusal clause belongs to the same local $\mathsf{NameCert}$ surface as maximum-rate exactness and observer-symmetry preservation, so a consumer that reads M without G has not consumed the accepted packet.

Finally use Definition 876.9. The invariant read reaches M only through $A, B, D, M, S, G, H, C, P, N$ and is blocked when G is absent. Since the carrier has no synchronized-clock, privileged-frame, metric, or global-coordinate row, the boundary is exhaustive. □

Theorem 876.11 (Cross-Hist causal-rate consumer route totality)**.** *Every consumer route out of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *carrier factors through exactly the displayed rows*

$$A, B, D, M, S, G, H, C, P, N.$$

The route reads the two observer histories, finite causal-dependence rows, maximum-rate row, observer-symmetry row, no-global-frame ledger, componentwise transport, continuation replay, package provenance, and local naming row, with no additional host rate, hidden causal witness, synchronized clock, or privileged observer-frame coordinate.

Proof. Start from Definition 876.1. It lists the complete carrier as $A, B, D, M, S, G, H, C, P, N$ and declares that no global observer frame, synchronized clock, metric tensor, host speed, unlisted causal witness, or detached rate read is a coordinate.

The observer-slot route theorem accounts for A, B, D, H, C, P, N, and the maximum-row exactness theorem accounts for M as attached to D. The symmetry non-escape and invariant-boundary theorems then account for S and G as required gates for invariant-facing rate reads.

Combining those projections leaves no unclassified consumer route. Every public read is one of the displayed carrier rows, replayed through the structural rows, and no outside row can be inserted without leaving the accepted $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ surface. □

Theorem 876.12 (Cross-Hist causal-rate observer-swap preservation)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet $R = (A, B, D, M, S, G, H, C, P, N)$, the swap row S is the only admissible observer-symmetry witness for the maximum-rate read. Swapping the two history slots preserves the rate only by consuming S together with the transported dependence routes in D and the structural rows H, C, P, N; no source-equivalence transport, host symmetry principle, or global observer frame supplies a second witness.*

Proof. Project the carrier. The row S is the displayed observer-symmetry coordinate, and no other row is declared as a witness that swapping A and B preserves the maximum-rate read.

Use Definition 876.2. Observer-symmetry preservation is a local certificate obligation for the same packet that contains D, M, H, C, P, N, so the swap read must remain attached to those displayed rows.

The invariant boundary confirms that a rate read requires S before it can be observer-symmetric. Because the packet also records no source-equivalence transport, host symmetry principle, or global frame, any admissible observer-swap preservation witness is exactly the row S transported through H, C, P, N. □

Theorem 876.13 (Cross-Hist causal-rate TasteGate obligation)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ CrossHistCausalRate}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ CrossHistCausalRate}^{\uparrow}$ *instances. These gate-facing instances must preserve the two Hist slots, causal-dependence route family, maximum-rate row, observer-symmetry row, no-global-frame ledger, transport rows, continuation routes, provenance row, and local naming row.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalRateUp.taste gate`

Proof. The packet is not determined by a single event name. Its carrier has separate rows for two histories, dependence routes, the maximum-rate row, observer symmetry, and no-global-frame refusal.

Nontriviality rules out an empty encoding in which every carrier collapses to one indistinct point. Field faithfulness records preservation of the displayed row inventory, so a TasteGate consumer can distinguish the two Hist slots, the dependence routes, the maximum-rate row, the symmetry row, and the no-global-frame ledger. □

Definition 876.14 (Cross-Hist causal-rate root source envelope)**.** *A* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root source envelope is the finite packet*

$$E = (A, B, D, M, S, G, H, C, P, N)$$

whose rows are read over $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *as follows:*

1. *A, B are the two observer-history identity slots;*
2. *D is the displayed cross-Hist causal route family over those slots;*
3. *M is the maximum causal-rate row attached to exactly the route family D;*
4. *S is the observer-symmetry row for the rate read;*
5. *G is the no-global-frame ledger;*
6. *H, C, P, N are respectively the componentwise* hsame *transport, displayed continuation trace, package provenance, and local NameCert rows.*

The envelope has no row for host speed, synchronized clocks, a global frame, a metric tensor, an unlisted causal witness, or a maximum-rate read detached from D.

Theorem 876.15 (Cross-Hist causal-rate root classifier stability)**.** *Let*

$$E = (A, B, D, M, S, G, H, C, P, N) \quad \text{and} \quad E' = (A', B', D', M', S', G', H', C', P', N')$$

be accepted $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root source envelopes. If the two Hist slots A, A' and B, B', the causal route families D, D', the maximum-rate rows M, M', the observer-symmetry rows S, S', and the no-global-frame ledgers G, G' agree componentwise under the displayed* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows, then E and E' classify the same root causal-rate packet.*

The classifier has no coordinate for a host speed, synchronized clock, global frame, metric tensor, detached maximum row, hidden causal route, or unlisted observer slot.

Proof. Project the root source envelopes. Their classifier-visible rows are exactly $A, B, D, M, S, G, H, C, P, N$ and $A', B', D', M', S', G', H', C', P', N'$.

Use Definition 876.6 for the Hist slots and causal route family. The observer-facing part of each packet is therefore exhausted by A, B, D, H, C, P, N and its primed counterpart.

Use Definition 876.7 together with Definition 876.8 and Definition 876.10. These results keep the maximum-rate row attached to D and keep the observer-symmetry and no-frame rows on the same route.

Finally apply Definition 876.11. Since every consumer route is one of the displayed rows, the componentwise agreement hypotheses identify the whole classifier packet and leave no hidden coordinate through which the two envelopes could diverge. □

Theorem 876.16 (Cross-Hist causal-rate root observer-slot exhaustion)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet exhausts its observer-slot reads through the displayed rows*

$$A, B, D, H, C, P, N.$$

A consumer may inspect the two observer histories only by reading A and B, replaying the causal route family D, transporting the components through H, following the displayed continuation route C, citing the package provenance P, and closing the local $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *row N. No unlisted observer slot, host clock, global frame coordinate, metric row, or detached causal route is available to such a read.*

Proof. Project the carrier of Definition 876.1. The only observer-facing rows are A and B, and the only displayed route family attached to them is D.

Apply the NameCert obligations of Definition 876.2 and the observer-slot route theorem Definition 876.6. These identify the observer histories as the carrier-visible slots and force consumer access through the finite route family.

The structural part is fixed by H, C, P, N. Consumer route totality from Definition 876.11 leaves no route outside the displayed transport, continuation, provenance, and local naming rows, while the frame-refusal boundary excludes host clocks, global frames, metric rows, and detached causal coordinates. □

Theorem 876.17 (Cross-Hist causal-rate root route-rate factorization)**.** *Every maximum-rate read from an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet factors through the displayed causal route family D before it reaches the maximum-rate row M. Thus a* $\mathsf{MaxCausalRate}^{\uparrow}$ *consumer may read M only after A, B, D, H, C, P, N have supplied the observer slots, route replay, transport, continuation, provenance, and local naming surface.*

Proof. Start from the carrier and the route obligation package. The row M is not a free host-valued speed coordinate; it is the maximum-rate row attached to exactly the causal route family D.

Apply maximum-row exactness from Definition 876.7. It binds the maximum-rate read to the displayed route family and rules out a second maximum row detached from D.

Now use observer-slot exhaustion together with consumer route totality. The read reaches D through A, B, H, C, P, N, then reads M as the rate row for that same route. Hence every maximum-rate read factors through D before the row M is consumed. □

Theorem 876.18 (Cross-Hist causal-rate root frame-refusal exactness)**.** *Every invariant-facing rate read from an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet consumes both the observer-symmetry row S and the no-global-frame ledger G. Consequently no global frame, synchronized clock, metric tensor, host speed row, or privileged observer coordinate can enter an* $\mathsf{InterHistInvariant}^{\uparrow}$ *or* $\mathsf{MaxCausalRate}^{\uparrow}$ *consumer route.*

Proof. Read the invariant boundary of Definition 876.9. It places the invariant-facing use of the rate behind the observer-symmetry row S and the no-global-frame ledger G.

Apply the no-global-frame boundary and frame-refusal composition theorems. The row G is the carrier-visible refusal row for synchronized clocks, privileged frames, metric tensors, host speeds, and global-frame coordinates, and it composes with S on the same route before any invariant-facing read is accepted.

Finally combine route-rate factorization with consumer route totality. Since the rate read has already passed through D before M and since every consumer route is displayed by H, C, P, N, there is no remaining coordinate through which a global frame or host speed row can enter. □

Theorem 876.19 (Cross-Hist causal-rate root ledger exactness)**.** *Every downstream rate consumer of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root source envelope reads exactly the displayed route, observer-symmetry, no-global-frame, transport, continuation, provenance, and local NameCert rows*

$$A, B, D, M, S, G, H, C, P, N.$$

The read has no host speed coordinate, synchronized clock, global frame, metric tensor, hidden causal route, detached maximum row, or unlisted observer slot.

Proof. Start with Definition 876.14. The envelope displays the observer slots, route family, rate row, symmetry row, no-frame ledger, transport, continuation, provenance, and local name as one finite packet.

Apply Definition 876.24. It states that every $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$ consumer reads exactly the ten rows of the accepted packet.

Apply Definition 876.34 to the continuation-facing part of the route. The displayed row C is the only continuation trace through which the consumer can replay the observer slots, causal routes, maximum-rate row, and invariant boundary.

The exclusion clause in the envelope and the totality theorem then rule out host speeds, synchronized clocks, global frames, metric tensors, hidden causal routes, detached maximum rows, and unlisted observer slots. Therefore the downstream ledger is exact. □

Theorem 876.20 (Cross-Hist causal-rate route obligation package)**.** *For the carrier*

$$R = (A, B, D, M, S, G, H, C, P, N)$$

of Definition 876.1, the rows $A, B, D, M, S, G, H, C, P, N$ *supply the complete* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *obligation package: carrier admission for the two observer-history slots, finite route replay for the displayed causal-dependence family, maximum-rate exactness for* M*, observer symmetry for* S*, no-global-frame refusal for* G*, componentwise* hsame *transport, displayed continuation replay, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *row.*

Proof. Project the displayed carrier. The two observer slots are A and B, the finite causal-dependence route family is D, and the maximum-rate row is M.

The rows S and G are the two boundary gates for reading M as an observer-symmetric invariant rather than a global-frame value. The introduction fixes the sibling sources for observer identity, cross-Hist causal rows, maximum causal rate, inter-Hist invariant reading, and finite continuation replay, so the package consumes exactly those public surfaces.

Finally H, C, P, N provide the transport, replay, provenance, and local naming rows. Since the carrier contains no additional coordinate, the listed rows exhaust the obligation package. □

Theorem 876.21 (Cross-Hist causal-rate route replay obligation)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *causal-rate read must replay the finite route family* D *before reading the maximum-rate row* M*. A read that reaches* M *without the displayed* D *routes is not a read of the carrier* $R = (A, B, D, M, S, G, H, C, P, N)$*.*

Proof. Project the carrier and inspect the order of the public rows. The row M is specified as a maximum-rate packet over exactly the dependence routes displayed in D.

The continuation coordinate C records the consumer route: read the two histories, replay each route in D, then read M. Thus M is not an independent host-valued speed row and cannot be consumed before the finite route family has been replayed. □

Theorem 876.22 (Cross-Hist causal-rate observer symmetry obligation)**.** *An observer-symmetric* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *rate read requires both the observer-symmetry row* S *and the no-global-frame ledger* G*. The row* S *supplies invariance under swapping the two Hist slots, while* G *refuses synchronized clocks, privileged observer frames, and global synchronization data.*

Proof. The invariant boundary theorem already identifies S and G as the two public gates for a maximum-rate read. Projecting the carrier shows that S is the only row recording preservation under the observer swap, and G is the only row excluding the forbidden global-frame data.

If S is absent, the read has no observer-symmetry witness. If G is absent, the same numerical-looking read could depend on a hidden synchronization surface. Hence an observer-symmetric rate read requires both rows. □

Theorem 876.23 (Cross-Hist causal-rate downstream package)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet*

$$R = (A, B, D, M, S, G, H, C, P, N),$$

the downstream package consists exactly of carrier admission, the two observer slots A, B*, causal route replay through* D*, maximum-row exactness through* M*, observer-symmetry through* S*, no-global-frame refusal through* G*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$ *row* N*.*

Proof. Project the accepted carrier. The observer slots, dependence routes, maximum-rate row, symmetry row, no-global-frame row, transports, continuation routes, provenance, and local name are exactly the displayed rows $A, B, D, M, S, G, H, C, P, N$.

Apply Definition 876.2, Definition 876.21, and Definition 876.22. These obligations keep the causal route replay before the maximum-rate read and keep the observer-symmetry and no-global-frame rows inside the same local naming surface.

Finally use Definition 876.11. It leaves no route for a hidden causal witness, host speed, synchronized clock, global frame, or detached maximum row, so the downstream package is exhausted by the displayed carrier projections. □

Theorem 876.24 (Cross-Hist causal-rate root consumer exactness)**.** *Every* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumer of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet reads exactly the rows*

$$A, B, D, M, S, G, H, C, P, N.$$

The consumer reads no maximum-rate value outside M*, no invariant boundary outside* S, G*, and no transport, route, provenance, or naming evidence outside* H, C, P, N*.*

Proof. Start with the downstream package of Definition 876.23. It names the whole consumer-facing packet and orders the maximum-rate read after the displayed causal route family.

For a $\mathsf{MaxCausalRate}^{\uparrow}$ consumer, apply Definition 876.7. The row M is attached to D, so the consumer cannot read a detached host-valued rate.

For an $\mathsf{InterHistInvariant}^{\uparrow}$ consumer, apply Definition 876.9 and Definition 876.10. These rows force observer symmetry and no-global-frame refusal through S and G before the rate can be used as an invariant.

The structural evidence is then fixed by H, C, P, N. Since the carrier has no additional coordinate, both root consumers read exactly the displayed ten rows. □

Theorem 876.25 (Cross-Hist causal-rate scoped route closure)**.** *The scoped route for every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root consumer is bound to* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, *and* $\mathsf{InterHistInvariant}^{\uparrow}$. *Within that scope, every downstream read factors through the rows* $A, B, D, M, S, G, H, C, P, N$ *and no global observer frame, synchronized clock, metric tensor, host real-valued speed, or unlisted causal witness is available.*

Proof. Project the carrier and read the provenance row. The row P names the packet over $\mathsf{CrossHistCausalRate}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Apply the downstream package and root consumer exactness theorems. They identify A, B as the observer-Hist slots, D as the displayed $\mathsf{CrossHistCausal}^{\uparrow}$ route family, M as the $\mathsf{MaxCausalRate}^{\uparrow}$ row, and S, G as the invariant-facing boundary consumed by $\mathsf{InterHistInvariant}^{\uparrow}$ readers.

The remaining rows H, C, N provide componentwise transport, displayed route replay, and local naming. Because Definition 876.1 contains no global frame, clock, metric, host-speed, or hidden causal coordinate, the scoped route is closed under exactly the listed BEDC primitives and sibling surfaces. □

Theorem 876.26 (Cross-Hist causal-rate route-bound monotonicity)**.** *Extending the displayed causal-dependence route family* D *of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet can update the maximum-rate row* M *only through the same observer slots* A, B *and the structural rows* H, C, P, N. *No extension of* D *creates a detached host speed, a fresh observer coordinate, or a second maximum-rate row outside the carrier* $R = (A, B, D, M, S, G, H, C, P, N)$.

Proof. Start with the route replay obligation. It places every causal-rate read through the displayed family D before the row M is consumed.

Use observer symmetry and the downstream package theorem. The same observer slots A, B and the same boundary rows S, G are carried along any accepted consumer route, so extending D cannot switch to a new observer interface.

Finally apply root consumer exactness and scoped route closure. They keep every update of M inside $A, B, D, M, S, G, H, C, P, N$, with H, C, P, N supplying transport, replay, provenance, and naming. Thus route extension is monotone only along the displayed BEDC rows. □

Theorem 876.27 (Cross-Hist causal-rate invariant cap exhaustion)**.** *Every invariant cap read by* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumes exactly the rows* $A, B, D, M, S, G, H, C, P, N$ *of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet. The cap cannot add a hidden host speed, metric tensor, synchronized clock, privileged frame, or unlisted frame coordinate.*

Proof. Apply the downstream package theorem and root consumer exactness. They identify $\mathsf{MaxCausalRate}^{\uparrow}$ and $\mathsf{InterHistInvariant}^{\uparrow}$ consumers with the displayed carrier rows, not with an ambient frame.

Route replay contributes D before M, and observer symmetry contributes the gate row S. The no-global-frame ledger G stays on the same route before the rate can be read as an invariant.

Scoped route closure then accounts for the structural rows H, C, P, N. Since the carrier has no further coordinate, every invariant cap is exhausted by exactly the ten displayed rows. □

Theorem 876.28 (Cross-Hist causal-rate frame-refusal composition)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet, the no-global-frame ledger G composes with the observer-symmetry row S and the causal route replay family D before any public rate read is accepted. A public read of M is therefore refused unless it passes through D, S, G, and the structural rows H, C, P, N over the same observer slots A, B.*

Proof. First replay the causal route family. The route replay obligation places D before any accepted read of the maximum-rate row M.

Next use observer symmetry. The row S keeps the read invariant under swapping the displayed observer-Hist slots, so the rate is not attached to a privileged observer order.

Finally apply the no-global-frame boundary together with downstream package exactness, root consumer exactness, and scoped route closure. These results place G on the same route as D and S, then repack the public read through H, C, P, N. Hence no public rate read is accepted before the frame-refusal ledger has composed with the observer-symmetric route replay. □

Theorem 876.29 (Cross-Hist causal-rate root carrier admission)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet exposes exactly the rows*

$$A, B, D, M, S, G, H, C, P, N$$

over BHist, hsame, Cont, Pkg, *and* NameCert. *The rows admit the two observer-history slots, causal-dependence route family, maximum-rate row, observer-symmetry row, no-global-frame ledger, componentwise transport, displayed replay, provenance, and local naming surface, with no additional root coordinate.*

Proof. Project the carrier in Definition 876.1. Its displayed inventory is the tuple $R = (A, B, D, M, S, G, H, C, P, N)$

Apply Definition 876.20 and Definition 876.23. These theorems identify the same rows as the carrier admission, route replay, maximum-row exactness, observer-symmetry, no-global-frame refusal, transport, continuation, provenance, and local naming obligations.

The exactness statement Definition 876.24 leaves no downstream read outside the ten displayed rows. Hence root carrier admission is exactly the finite BHist, hsame, Cont, Pkg, and NameCert surface named above. □

Theorem 876.30 (Cross-Hist causal-rate root route replay)**.** *Every downstream invariant-facing read of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet replays the finite causal-dependence route family D before reading the maximum-rate row M. The read reaches M only through the displayed rows H, C, P, N and the invariant boundary rows S, G.*

Proof. Start with Definition 876.21. It states that a causal-rate read must replay D before consuming M.

For invariant-facing consumers, apply Definition 876.9 and Definition 876.22. These place the observer-symmetry row S and the no-global-frame ledger G on the same route before the rate can be used as an invariant.

Finally use Definition 876.25. It binds the route to the displayed transport, continuation, provenance, and local naming rows, so no invariant-facing read can reach M by skipping the finite D family. □

Theorem 876.31 (Cross-Hist causal-rate root no-frame ledger)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet, the row G is the complete no-global-frame ledger. It refuses synchronized clocks, privileged observer frames, metric tensors, host speeds, and global-frame coordinates for any downstream maximum-rate or invariant-facing read.*

Proof. Project the ledger row from the carrier and apply Definition 876.10. The row G is the unique displayed boundary excluding global synchronization data from a maximum-rate read.

Apply Definition 876.30 and Definition 876.29. They keep every downstream read inside $A, B, D, M, S, G, H, C, P, N$, with G on the route before M can be used by an invariant-facing consumer.

The scoped route closure theorem then rules out hidden global-frame coordinates, host speeds, metric tensors, synchronized clocks, and privileged observer frames. Since the carrier contains no second ledger row, G is the complete refusal ledger for the root packet. □

Theorem 876.32 (Cross-Hist causal-rate route-family exhaustion)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet exhausts its causal route reads through the displayed family D and the structural rows A, B, H, C, P, N. A consumer may replay a causal route only by reading the two observer-history slots A, B, the route family D, componentwise transport H, displayed continuation replay C, package provenance P, and the local naming row N; it may not read a hidden causal route, unlisted unary slot, synchronized clock, privileged frame, or metric row.*

Proof. Project the carrier of Definition 876.1. The only causal-dependence coordinate is the finite family D, and it is attached to the two displayed observer-history slots A and B.

Apply Definition 876.21 and Definition 876.25. These results force route replay through D and keep every downstream route inside the stated BEDC scope.

Use Definition 876.24. It fixes the structural rows H, C, P, N as the transport, continuation, provenance, and naming surface for root consumers. Since the carrier has no hidden causal-route coordinate, the displayed family exhausts all accepted causal reads. □

Theorem 876.33 (Cross-Hist causal-rate maximum observer gate)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet, the row M is readable as an invariant only after the causal route family D has been replayed and both observer gate rows S, G have been displayed. Thus no* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumer can read M as an invariant from a route that omits the observer-symmetry row, the no-global-frame ledger, or the displayed causal replay.*

Proof. Start from Definition 876.30. It places replay of D before every invariant-facing read of M.

Apply Definition 876.9 and Definition 876.22. They identify S and G as the two observer-facing gates needed before a rate read can be used as an invariant.

Finally use Definition 876.31 and Definition 876.24. These results keep the gate rows on the same root consumer route and exclude synchronized clocks, privileged frames, metric tensors, host speeds, and detached maximum rows. □

Theorem 876.34 (Cross-Hist causal-rate continuation-trace factorization)**.** *Every continuation-facing read of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet factors through the* $\mathsf{ContinuationTrace}^{\uparrow}$ *row C after the observer identity rows A, B, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *route family D, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row M, and the* $\mathsf{InterHistInvariant}^{\uparrow}$ *boundary rows S, G have been displayed. The factorization has no route through a hidden continuation, global frame, host speed, unlisted causal witness, or detached invariant boundary.*

Proof. Read the displayed packet $R = (A, B, D, M, S, G, H, C, P, N)$ from Definition 876.1. The continuation coordinate C is the only row that records the consumer route through the observer slots, route family, maximum-rate row, and invariant boundary.

Apply Definition 876.32. It forces the causal part of the read through A, B, D, H, C, P, N and rules out hidden causal routes.

Apply Definition 876.33. It places the maximum-rate row M behind the displayed route replay and the two observer gate rows S, G.

Use Definition 876.25 and Definition 876.24 to repack the read as a root consumer route. The structural factor is therefore exactly C together with H, P, N, and no continuation-facing consumer can bypass the displayed route. □

Theorem 876.35 (Cross-Hist causal-rate root observer-gate totality)**.** *Every invariant-facing read of the maximum-rate row M from an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet must pass through the displayed causal route family D, the observer gate rows S, G, and the structural rows H, C, P, N. Thus a* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumer never reads M directly from the observer slots A, B, from a hidden frame row, or from a detached host-speed coordinate.*

Proof. Read the carrier $R = (A, B, D, M, S, G, H, C, P, N)$ from Definition 876.1. The carrier gives the two observer slots and names D, M, S, G, H, C, P, N as the only route, rate, gate, transport, replay, provenance, and naming rows.

Apply Definition 876.30. It places the finite route family D before any invariant-facing read of M and keeps the structural rows on the same route.

Apply Definition 876.33. It adds the observer-symmetry row S and the no-global-frame ledger G before M can be consumed by $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$.

Finally use Definition 876.34. The continuation-facing read factors through C after the observer slots, route family, maximum-rate row, and gate rows have been displayed, so no invariant-facing consumer can bypass D, S, G, H, C, P, N. □

Theorem 876.36 (Cross-Hist causal-rate root route-family field faithfulness)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet, the rows*

$$A, B, D, M, S, G, H, C, P, N$$

remain distinguishable as the two observer-history slots, causal route family, maximum-rate row, observer-symmetry row, no-global-frame ledger, componentwise transport, continuation replay, provenance, and local naming row. Any future field-faithful TasteGate encoding must preserve these rows as separate carrier fields and may not identify D with M, S with G, or the structural rows H, C, P, N with the observer slots A, B.

Proof. Project the carrier definition. It lists $A, B, D, M, S, G, H, C, P, N$ as separate displayed coordinates, and each coordinate has a different consumer role in the packet.

Use Definition 876.30 to separate D from M: the route family is replayed before the maximum-rate row is read, so the two rows cannot collapse without losing the replay obligation.

Use Definition 876.31 and Definition 876.33 to separate S and G from the rate row and from each other. The former records observer symmetry, while the latter refuses global-frame coordinates before the invariant-facing read.

The factorization theorem Definition 876.34 keeps C as the continuation trace, with H, P, N supplying transport, provenance, and naming. Since those structural rows govern access to the packet rather than serving as observer slots, field faithfulness must keep the full displayed row inventory distinct. □

Theorem 876.37 (Cross-Hist causal-rate root no-frame consumer cut)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet, the no-global-frame ledger* G *is a consumer cut between valid* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *reads and any hidden clock, privileged frame, metric tensor, speed coordinate, or global-frame row. A consumer route that reaches* M *as a rate or invariant must pass through* G *together with* D, S, H, C, P, N*.*

Proof. Start with Definition 876.31. It identifies G as the complete ledger refusing synchronized clocks, privileged observer frames, metric tensors, host speeds, and global-frame coordinates.

Apply Definition 876.30. The valid consumer route reaches M only after replaying the finite causal route family D through the structural rows.

Apply Definition 876.33. It places S and G before the maximum-rate row can be read as an invariant-facing value by $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$.

Finally use Definition 876.34. Since the displayed continuation factor is C and the carrier has no hidden clock, frame, metric, speed, or global-coordinate row, G separates accepted consumers from every forbidden frame-facing read. □

Theorem 876.38 (Cross-Hist causal-rate root observer-slot totality)**.** *For every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root consumer, the first observer-history read is exhausted by the two displayed slots* A, B *through* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *and by the rows* D, H, C, P, N *of the carrier*

$$R = (A, B, D, M, S, G, H, C, P, N).$$

The consumer therefore cannot begin from an unlisted observer slot, a hidden history identity witness, or a detached causal route outside the $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet.*

Proof. Read the carrier of Definition 876.1. It presents A, B as the only observer-history slots and gives D, H, C, P, N as the route, transport, continuation, provenance, and local naming rows around those slots.

Apply Definition 876.6. It forces every observer-slot read through the displayed $\mathsf{ObserverHistIdentity}^{\uparrow}$ surface and the route rows of the packet.

Next apply Definition 876.30. The replay step keeps the consumer on the same finite D route before any downstream maximum-rate read, matching the causal witness surface of chapter 478.

Since chapter 594 and chapter 656 consume the rate only after the route has been displayed, the root consumer has no earlier observer source than A, B, D, H, C, P, N. □

Theorem 876.39 (Cross-Hist causal-rate root dependence-route exhaustion)**.** *Every dependence read of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet is exhausted by the displayed* $\mathsf{CrossHistCausal}^{\uparrow}$ *route family* D *replayed through the continuation row* C*. The replay may pass through* A, B, H, P, N*, but it cannot introduce a hidden causal witness, an unlisted dependence edge, a host clock, or a frame-side causal row.*

Proof. Project the packet $R = (A, B, D, M, S, G, H, C, P, N)$. The only dependence-family coordinate is D, and the only continuation-facing replay coordinate is C.

Apply Definition 876.30. It places every downstream dependence read on the finite route family D before the rate row M can be consumed.

Now use Definition 876.6. The route is anchored in the two observer slots A, B and is transported only through the displayed structural rows H, C, P, N.

Finally apply Definition 876.37. The no-frame cut excludes clocks, privileged frames, metric tensors, host speeds, and global-frame rows, so the replay through C cannot acquire an extra causal witness outside the $\mathsf{CrossHistCausal}^{\uparrow}$ route family cited by chapter 478. □

Theorem 876.40 (Cross-Hist causal-rate root maximum-invariant handoff)**.** *An accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root packet may hand the maximum-rate row* M *to* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *only after the dependence route family* D*, the observer-symmetry row* S*, and the no-global-frame ledger* G *are visible on the same root route. Thus* M *is never exported as a detached rate value before the causal route, observer symmetry, and no-frame refusal rows have been displayed.*

Proof. Start with Definition 876.30. It makes D the finite route that precedes any invariant-facing read of M.

Apply Definition 876.33. This places S and G before a $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$ consumer may read M as an invariant-facing value.

Use Definition 876.37. The cut keeps the handoff away from hidden clocks, privileged frames, metric tensors, host speeds, and global-frame rows.

The handoff therefore matches the sibling consumer surfaces of chapter 594 and chapter 656: both receive M only after the displayed D, S, G route has been exposed by the $\mathsf{CrossHistCausalRate}^{\uparrow}$ carrier. □

Theorem 876.41 (Cross-Hist causal-rate invariant-rate ledger)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet with carrier*

$$R = (A, B, D, M, S, G, H, C, P, N)$$

as in Definition 876.1, any $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumer that reads* M *as a rate or invariant first passes through the displayed route rows* D, S, G, H, C, P, N*. The ledger is the same finite root route supplied by route replay, observer-gate totality, and the no-frame consumer cut; it has no coordinate for a hidden causal route, synchronized clock, privileged frame, metric tensor, host speed, or detached maximum row.*

Proof. Read the carrier displayed in Definition 876.1. The only rows downstream of the observer slots and upstream of the local name are the causal route family D, maximum-rate row M, observer-symmetry row S, no-frame ledger G, and structural rows H, C, P, N.

Apply Definition 876.30. It places D before every invariant-facing read of M and keeps the read inside the displayed structural route.

Now apply Definition 876.33 and Definition 876.37. These results add S and G as mandatory gates before a $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$ consumer can use M.

The route is then closed by Definition 876.24. Since the carrier has no extra frame-facing coordinate, the invariant-rate ledger is exactly D, S, G, H, C, P, N before M is consumed. □

Theorem 876.42 (Cross-Hist causal-rate observer-symmetry totality)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *root consumer that exports the maximum row* M *to* $\mathsf{MaxCausalRate}^{\uparrow}$ *or* $\mathsf{InterHistInvariant}^{\uparrow}$ *exposes both observer-facing gates* S *and* G *on the same route as* D, H, C, P, N*. Thus observer symmetry and no-frame refusal are total for rate consumers: neither can be omitted, split into a second packet, or supplied by a host frame outside the carrier of Definition 876.1.*

Proof. Start from Definition 876.41. It already places every invariant-rate read behind the displayed route D, S, G, H, C, P, N.

Use Definition 876.22. The row S records the observer-symmetry obligation for the two history slots, so a consumer that omits S has no local certificate for invariant comparison.

Use Definition 876.31. The row G is the finite refusal ledger for global-frame data, so a consumer that omits G has no certified refusal of hidden clocks, privileged frames, metrics, or host speeds.

Finally apply Definition 876.24. It keeps S and G on the same root route as D, H, C, P, N, giving totality for the observer-symmetry and no-frame gates. □

Theorem 876.43 (Cross-Hist causal-rate consumer boundary)**.** *The consumer boundary for an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet is exact for* $\mathsf{MaxCausalRate}^{\uparrow}$ *and* $\mathsf{InterHistInvariant}^{\uparrow}$ *reads: before reading* M*, the consumer must pass through*

$$D, S, G, H, C, P, N.$$

Consequently the root boundary is generated by the displayed carrier of Definition 876.1, route replay, observer-symmetry totality, and no-frame refusal, not by any off-carrier frame, external speed comparison, hidden causal route, or detached invariant source.

Proof. Begin with Definition 876.37. It states that the no-frame ledger G separates valid maximum-rate or invariant reads from hidden clocks, privileged frames, metric tensors, and host speeds.

Apply Definition 876.42. This puts S and G on the same root route as D, H, C, P, N for every consumer that exports M.

Now use Definition 876.41. The invariant-rate ledger fixes the route before M is read, so the consumer boundary cannot be shortened to D alone or to the observer slots alone.

The exactness claim follows from Definition 876.24. The accepted packet contains no remaining coordinate from which a $\mathsf{MaxCausalRate}^{\uparrow}$ or $\mathsf{InterHistInvariant}^{\uparrow}$ consumer could recover an alternate read of M. □

Theorem 876.44 (Cross-Hist causal-rate observer identity dependence)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet* $R = (A, B, D, M, S, G, H, C, P, N)$*, the observer slots* A *and* B *must be read through the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *identity surface before any route family* D *is admissible. A route family detached from that identity read is not a valid causal-rate route.*

Proof. Apply the observer-slot route theorem Definition 876.6. It makes the two observer histories the entrance to the displayed causal route family.

Use Definition 876.38 and Definition 876.39. They place every admissible D behind the same observer-slot read and prevent a hidden route outside A, B, H, C, P, N.

The no-global-frame boundary, root consumer exactness, and consumer boundary of Definition 876.10, Definition 876.24, and Definition 876.43 then keep the identity dependence inside the packet rather than in a host frame. □

Theorem 876.45 (Cross-Hist causal-rate falsifiability and independence surface)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet*

$$R = (A, B, D, M, S, G, H, C, P, N),$$

the carrier has a falsifiable consequence: a maximum-rate read is rejected unless both the observer-symmetry row S *and the no-global-frame ledger* G *are present on the same displayed route as* D, M, H, C, P, N*. Its independence surface is witnessed by the simultaneous presence of the two* Hist *slots* A, B*, causal-dependence route family* D*, maximum-rate row* M*, observer-symmetry row* S*, no-global-frame ledger* G*, componentwise transport* H*, continuation row* C*, provenance row* P*, and local name* N*. This surface is not bijective to any listed sibling surface* $\mathsf{InterHistInvariant}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*,* $\mathsf{HistTimeStream}^{\uparrow}$*,* $\mathsf{ContinuationTrace}^{\uparrow}$*, or* $\mathsf{SupplyKindRouter}^{\uparrow}$*.*

Proof. Read the carrier definition Definition 876.1. It displays the two history slots, route family, maximum-rate row, observer-symmetry row, no-frame ledger, and structural rows as simultaneous fields of one packet.

Apply maximum-row exactness from Definition 876.7. The maximum-rate row M is attached to the finite dependence family D, not to a detached host speed or metric coordinate.

Use the invariant boundary and no-global-frame boundary of Definition 876.9 and Definition 876.10. Together with observer identity dependence from Definition 876.44, they require the observer-symmetry and no-frame rows before a public maximum-rate read can pass to an invariant-facing consumer.

Finally use route-family field faithfulness from Definition 876.36. The separate fields $A, B, D, M, S, G, H, C, P, N$ cannot collapse to the inventory of any one sibling surface, so the packet exposes a distinct independence surface while keeping the falsifiable maximum-rate rejection tied to its carrier rows. □

Theorem 876.46 (Cross-Hist causal-rate MaxCausalRate dependence)**.** *The row M of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet is a* $\mathsf{MaxCausalRate}^{\uparrow}$ *row only over the already displayed route family D. It is not a detached maximum, a host speed, or a rate read available before D, S, G, H, C, P, N.*

Proof. Start with the maximum-row exactness theorem Definition 876.7. It binds M to the carrier route rather than to an ambient speed comparison.

Apply Definition 876.17, Definition 876.33, and Definition 876.41. These rows put the route family D and the gates S, G, H, C, P, N before any maximum-rate consumer reads M.

The root consumer exactness and consumer boundary theorems finish the dependence claim by excluding every off-carrier source for a maximum-rate row. □

Theorem 876.47 (Cross-Hist causal-rate InterHistInvariant dependence)**.** *An* $\mathsf{InterHistInvariant}^{\uparrow}$ *consumer of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet receives the maximum row M only after the displayed route and structural rows D, S, G, H, C, P, N. Thus the invariant-facing read is downstream of observer identity, causal route exposure, observer symmetry, no-frame refusal, transport, continuation replay, provenance, and local naming.*

Proof. Use the invariant boundary theorem Definition 876.9. It already treats invariant consumption as downstream of the displayed Cross-Hist packet.

Then apply Definition 876.40, Definition 876.41, and Definition 876.43. These force an invariant-facing consumer to pass through D, S, G, H, C, P, N before M is handed off.

Finally, Definition 876.24 prevents a second invariant source, so the $\mathsf{InterHistInvariant}^{\uparrow}$ dependence is exactly the displayed packet route. □

Theorem 876.48 (Cross-Hist causal-rate root source-channel totality)**.** *Every accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet exposes its two observer-history source slots, finite causal-dependence route, observer-symmetry row, and no-global-frame row through the displayed route*

$$A, B, D, S, G, H, C, P, N$$

before any consumer reads the maximum-rate row M. Thus the source channel for a maximum-rate read is the packet's finite BHist *route with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *certification, not hidden frame data or an ambient clock.*

Proof. Start from the observer-slot route of Definition 876.6. It makes A and B the only entrance for the causal-dependence family D.

Apply Definition 876.38 and Definition 876.39. These force every accepted route row to pass through the same finite observer-history slots before M is available.

Use Definition 876.42 and Definition 876.10. The first supplies the observer-symmetry row S, and the second keeps the no-global-frame row G inside the displayed packet.

Finally apply Definition 876.43. It places H, C, P, N on the same route before any maximum-rate read, giving totality for the displayed source channel. □

Theorem 876.49 (Cross-Hist causal-rate max-bound ledger)**.** *For an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet, the maximum-rate row* M *is ledger-bound by the route rows*

$$A, B, D, S, G, H, C, P, N.$$

No consumer may replace this ledger with metric tensors, synchronized clocks, host speeds, a privileged frame, or any off-carrier maximum-rate primitive.

Proof. Use maximum-row exactness from Definition 876.7. It binds M to the displayed packet rather than to an external speed comparison.

The invariant boundary and no-global-frame boundary of Definition 876.9 and Definition 876.10 exclude hidden frame data, synchronized clocks, metric tensors, and host speeds from the consumer route.

Apply Definition 876.48. Since every maximum-rate read first exposes $A, B, D, S, G, H, C, P, N$, the ledger for M is exactly the displayed root route. □

Theorem 876.50 (Cross-Hist causal-rate downstream factorization)**.** *Every* $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, *or* $\mathsf{LorentzFrameRate}^{\uparrow}$ *consumer of an accepted* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *packet factors through the same displayed route*

$$A, B, D, S, G, H, C, P, N$$

before reading M*. The downstream consumer may specialize the rate, invariant, or frame-facing interpretation only after that route has been exposed.*

Proof. For $\mathsf{MaxCausalRate}^{\uparrow}$, apply Definition 876.46. It reads M only after the displayed causal route and structural gates.

For $\mathsf{InterHistInvariant}^{\uparrow}$, apply Definition 876.47. It places invariant-facing consumption downstream of observer identity, route exposure, observer symmetry, no-frame refusal, transport, continuation, provenance, and naming.

For $\mathsf{LorentzFrameRate}^{\uparrow}$, use the same max-bound ledger of Definition 876.49. Since a frame-facing rate consumer has no permitted off-carrier metric, clock, host-speed, or privileged-frame coordinate, it must factor through the displayed route before reading M.

The consumer boundary of Definition 876.43 closes the three cases by preventing a second route for downstream reads. □

Falsifiable prediction: An accepted $\mathsf{CrossHistCausalRate}^{\uparrow}$ packet admits no maximum-rate read unless both observer-symmetry and no-global-sync rows are present.

Independence witness: interhistinvariant, observerhistidentity, histtimestream, continuationtrace, supplykindrouter: $\mathsf{CrossHistCausalRate}^{\uparrow}$ has two Hist slots, a causal-dependence route, a maximum-rate row, and no-global-frame refusal data; these rows are not bijective with invariant-only, identity-only, time-stream-only, trace-only, or socket-router packets.

Closure status for $\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CrossHistCausalRate}^{\uparrow}$ packet is generated from two observer-history slots, finite causal-dependence route rows, a maximum-rate row, an observer-symmetry row, a no-global-frame ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CrossHistCausalRate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 876.1, Definition 876.2, Definition 876.6, Definition 876.7, Definition 876.8, Definition 876.9, Definition 876.10, Definition 876.11, Definition 876.12, Definition 876.13, Definition 876.48, Definition 876.49, Definition 876.50, Definition 876.45, and the recorded TasteGate packet bridge the public carrier, causal route, maximum-rate exactness, observer-symmetry row, no-global-frame refusal, downstream factorization, falsifiable prediction, independence witness, and TasteGate packet as one paper-side consumer surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{CrossHistCausalRate}^{\uparrow}}$, theoremChecked)
Lean target: `BEDC.Derived.CrossHistCausalRateUp.taste\_gate`
Bridge status: none
Origin: ai
Not claimed: This chapter does not derive a numerical speed constant, Lorentz geometry, metric tensors, synchronized clocks, global frames, or bridge checking for CrossHistCausalRate$^{\uparrow}$.
Upgrade path: No further paper-side theory closure grade remains; only independent verification-axis and bridge-axis checks can strengthen the recorded status.

Chapter 877

A Concrete Naming Certificate for $\mathsf{ObserverAccumulationClock}^{\uparrow}$

The $\mathsf{ObserverAccumulationClock}^{\uparrow}$ packet makes the time-as-accumulation reading of chapter 9420 into a finite NameCert carrier. It reads its state side through chapter 578, its observer-history identity side through chapter 477, and its order side through chapter 664. Unlike the state, identity, locality, order, or frame siblings, the packet stores a named transition witness that produces the next public clock tick only when the accumulated-record edge is displayed.

Definition 877.1 (ObserverAccumulationClock carrier)**.** *An* $\mathsf{ObserverAccumulationClock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S_0, S_1, A, G, H, R, P, N)$$

with the following displayed rows:

1. S_0 *is the present observer-state row, read through the* $\mathsf{ConsciousObserverState}^{\uparrow}$ *surface;*
2. S_1 *is the successor observer-state row, carrying the records visible after the next accepted accumulation step;*
3. A *is the accumulation-edge row, naming the retained-record inclusion from* S_0 *to* S_1 *and producing the public tick;*
4. G *is the* Gap *residue row for demanded tick reads whose accumulation edge, retained-record comparison, or successor state is absent;*
5. H *records componentwise* hsame *transport for present state, successor state, accumulation edge, gap residue, route, package, and local naming rows;*
6. R *records displayed* Cont *routes by which consumers read* S_0*, then* S_1*, then* A*, and only then the clock-read result;*
7. P *is the* Pkg *provenance over* $\mathsf{ConsciousObserverState}^{\uparrow}$*,* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{ObservationTimeOrder}^{\uparrow}$*,* BHist*,* Gap*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{ObserverAccumulationClock}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host scalar clock, global synchronization map, direct state-to-clock read, reversible tick, or time read detached from the displayed accumulation edge.

Theorem 877.2 (ObserverAccumulationClock NameCert obligations)**.** *For the carrier of Definition 877.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverAccumulationClock}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet* $C = (S_0, S_1, A, G, H, R, P, N)$*;*
2. *successor-record exposure, requiring* S_1 *to be read as a successor observer-state row of the same observer-history identity surface as* S_0*;*
3. *accumulation-edge exactness, requiring* A *to witness retained-record accumulation from* S_0 *to* S_1 *before any tick can be exported;*
4. Gap *residue exactness, requiring* G *to classify any demanded tick lacking* S_1 *or* A *as outside the accepted clock surface;*
5. *stability and non-escape under the componentwise* hsame *row* H*, displayed route row* R*, package provenance* P*, and local naming row* N*.*

Proof. Carrier admission is projection from Definition 877.1. The displayed packet exposes the two observer-state rows, the accumulation edge, the gap residue, transport, route, provenance, and local name.

The successor-record clause uses the conscious-state and observer-history identity siblings. The row S_1 is not a free clock value; it is another observer-state row on the same identity surface as S_0.

The accumulation-edge clause then reads A. By the observation-time-order sibling, a time comparison is record retention between displayed states, so a public tick is obtained only after the route has exposed the retained-record edge from S_0 to S_1.

The gap-residue row G handles failed tick demands. If a consumer asks for a tick without a successor state or without the accumulation edge, the request remains a residue rather than an accepted clock read.

Finally H, R, P, N transport, replay, package, and name the same finite packet. Since no displayed row stores a host scalar clock, direct state-to-clock function, global synchronization map, or reversible tick, the local certificate exports none. □

Theorem 877.3 (ObserverAccumulationClock falsifiable boundary)**.** *Every accepted* ObserverAccumulationClock$^{\uparrow}$ *consumer that claims a time tick factors through the displayed route*

$$S_0 \to S_1 \to A$$

and then through the transport, package, and local naming rows H, R, P, N*. A direct read from a single observer-state row to a clock value, or a tick read that bypasses the accumulation-edge row* A*, is a counter-witness to the accepted packet.*

Proof. Inspect the carrier rows. The only two state coordinates are S_0 and S_1, and the only row that relates them as a tick-producing transition is A.

The displayed route row R fixes the consumer order: read the present state, read the successor state, then read the accumulation edge. The transport row H preserves those coordinates componentwise, while P and N bind the route to the local package and name.

Thus a claimed clock read that starts and ends at a single state has no carrier coordinate that can justify it. Likewise a read from S_0 to S_1 that omits A has not displayed the accumulation witness required by the packet. □

Theorem 877.4 (ObserverAccumulationClock TasteGate obligation)**.** *The later Lean target in* `lean4/BEDC/Derived/ObserverAccumulationClockUp/TasteGate.lean` *must provide* Nontrivial ObserverAccumu *and* FieldFaithful ObserverAccumulationClock$^{\uparrow}$ *using explicit witness pairs and field projections. These instances must distinguish the present state, successor state, accumulation-edge row,* Gap *residue, transport row, displayed route, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.ObserverAccumulationClockUp.ObserverAccumulationClockTasteGate single carrier alignment`

Proof. Nontriviality is forced by the two-state edge shape. A packet that collapses S_0, S_1, and A into one indistinct point cannot witness a tick-producing accumulation edge.

Field faithfulness is forced by the carrier inventory. The field projections must preserve the state rows, the edge row, the residue row, and the structural rows separately so a consumer can reject a direct state-to-clock read. □

Falsifiable prediction: Any accepted ObserverAccumulationClockUp consumer that claims a time tick must factor through the displayed accumulation-edge row; a direct state-to-clock read is a counter-witness.

Independence witness: Nearest siblings: consciousobserverstate, observerhistoryidentity, observerlocalitycell, observationtimeorder, observercoordinateframe. The clock packet is not bijective to any of them because it carries a tick-producing accumulation-edge and clock-read row; the siblings carry state, identity, locality, order, or frame rows without this named transition witness.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverAccumulationClock}\uparrow}$.

Constructive story: An $\mathsf{ObserverAccumulationClock}^{\uparrow}$ packet is generated from present observer-state, successor observer-state, accumulation-edge, GAP-residue, componentwise hsame transport, displayed route, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverAccumulationClock}\uparrow}, \mathsf{seed})$

Scope closed: Definition 877.1, Definition 877.2, Definition 877.3, and Definition 877.4 specify the finite observer-accumulation clock packet and its tick boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverAccumulationClock}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, host scalar time, global synchronization, reversible ticks, or direct state-to-clock reads.

Upgrade path: Obligation closure requires checked carrier admission, successor-record exposure, accumulation-edge exactness, GAP-residue exactness, and TasteGate nontriviality with field faithfulness.

Chapter 878

A Concrete Naming Certificate for $\mathsf{FableMachineBoundary}^{\uparrow}$

The $\mathsf{FableMachineBoundary}^{\uparrow}$ interface gives the Empty Fable Machine of chapter 9399 a finite boundary packet. It combines the Empty-fable ledger of chapter 516, the single-branch witness of chapter 585, the local-clock row of chapter 623, and the selector-ledger consumer surface of chapter 494. The packet is not a subject behind the observer and not a hidden chooser; it is the displayed boundary at which a finite selector ledger, enacted boundary row, and no-hidden-subject condition are read together.

Definition 878.1 (FableMachineBoundary carrier)**.** *A* $\mathsf{FableMachineBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (H, E, L, S, W, K, T, C, P, N)$$

with the following visible rows:

1. *H is the generated* Hist *row whose constructor spine supplies the observer sequence;*
2. *E is the empty-boundary slot row above the displayed source history;*
3. *L is the finite Empty-fable selector ledger read from the same* Hist *row;*
4. *S is the enacted selector row, including the selected mark and its displayed* Ext *edge;*
5. *W is the local fable-branch witness tying the enacted selector row to the boundary slot;*
6. *K is the local fable-machine clock row for the same finite step surface;*
7. *T records componentwise* hsame *transports for the history, boundary, ledger, selector, branch-witness, and clock rows;*
8. *C records the* Cont *routes by which consumers read those rows;*
9. *P records* Pkg *provenance over* BHist, Ext, *selector-ledger, fable-branch, fable-clock,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FableMachineBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through this displayed tuple. It has no coordinate for a hidden observer-substance, global branch chooser, host trace quotient, truth oracle, or selector rule outside the finite ledger.

Lean4 ✓ `BEDC.Derived.FableMachineBoundaryUp.FableMachineBoundaryTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.FableMachineBoundaryUp.FableMachineBoundaryUp`

Theorem 878.2 (FableMachineBoundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FableMachineBoundary}^{\uparrow}}$ *surface for the carrier of Definition 878.1 consists exactly of five finite obligations:*

1. *carrier admission for* $B = (H, E, L, S, W, K, T, C, P, N)$*;*
2. *empty-boundary admission tying* E *to the displayed source* Hist *row and its enacted* Ext *edge;*
3. *selector-ledger exactness tying* L *and* S *to the same finite Empty-fable trace;*
4. *branch and clock compatibility tying* W *and* K *to the enacted selector row rather than to an external chooser;*
5. *consumer non-escape through* T, C, P, N *for every history, boundary, ledger, selector, witness, and clock read.*

Every obligation is read from the displayed BHist*,* Ext*,* hsame*,* Cont*,* Pkg*, selector-ledger, fable-branch, fable-machine-clock, and* NameCert *rows. No obligation introduces a hidden subject coordinate or a chooser row not already visible in the packet.*

Lean4 ✓ `BEDC.Derived.FableMachineBoundaryUp.FableMachineBoundaryNameCertObligations`

Proof. Project the carrier rows. The empty-boundary slot E, selector ledger L, enacted selector row S, branch witness W, and local clock K are separate coordinates of the same finite packet.

Use the Empty-fable ledger and selector-ledger surfaces to read the trace and enacted selector from L and S. The branch witness then attaches the selected mark to the displayed Ext edge, while the fable-machine clock records the same local step order.

The transport, continuation, package, and naming rows T, C, P, N replay those reads componentwise. Since the carrier contains no observer-substance, global chooser, host trace quotient, or truth oracle coordinate, the five obligations exhaust the boundary certificate. □

Theorem 878.3 (FableMachineBoundary selector non-escape)**.** *Every consumer read of an accepted* $\mathsf{FableMachineBoundary}^{\uparrow}$ *packet factors through*

$$H, E, L, S, W, K, T, C, P, N.$$

In particular, the selected branch can be read only as the displayed selector row S *together with the fable-branch witness* W *and the finite selector ledger* L*. No consumer route reaches a hidden chooser row, observer-substance coordinate, global branch policy, or host-level trace equality.*

Lean4 ✓ `BEDC.Derived.FableMachineBoundaryUp.FableMachineBoundarySelectorNonescape`

Proof. Read the consumer request through C. If it asks for the generated history, empty-boundary row, selector ledger, enacted selector row, branch witness, clock row, transport row, provenance row, or local name, the carrier supplies exactly the corresponding displayed coordinate.

For any request for a hidden chooser, observer-substance coordinate, global branch policy, or host trace equality, inspect the carrier inventory. None of those objects occurs in $B = (H, E, L, S, W, K, T, C, P, N)$, and the NameCert obligations admit selector data only through L, S, W, K.

Thus every accepted read remains inside the finite boundary packet, and the enacted selector cannot escape as an undisplayed chooser. □

Theorem 878.4 (FableMachineBoundary inscription non-escape)**.** *Every Empty Fable Machine inscription read accepted by a* $\mathsf{FableMachineBoundary}^{\uparrow}$ *carrier must pass through the displayed rows*

$$H, E, L, S, W, K, T, C, P, N$$

and through the sibling surfaces $\mathsf{EmptyFableLedger}^{\uparrow}$, $\mathsf{FableBranchWitness}^{\uparrow}$, $\mathsf{FableMachineClock}^{\uparrow}$, *and* $\mathsf{SelectorLedger}^{\uparrow}$ *before it exports a boundary name. The inscription route exposes only the generated history row, empty-boundary slot, selector ledger, enacted selector, branch witness, local clock, componentwise transport, replay, provenance, and local naming rows; it has no route to a hidden subject coordinate, global branch chooser, host trace quotient, or truth oracle.*

Proof. Project the carrier of Definition 878.1. The inscription request can inspect only the history row H, boundary row E, selector ledger L, enacted selector S, branch witness W, local clock K, and the structural rows T, C, P, N.

Read the sibling surfaces in order. The Empty-fable ledger supplies the finite trace, the selector ledger supplies the enacted selector row, the branch witness attaches that selector to the displayed boundary branch, and the local clock records the finite step at which the read is made.

Finally replay the inscription through T, C, P, N. These rows preserve and name the already displayed finite route, so they cannot add a hidden observer, global chooser, host trace quotient, or truth oracle. The exported boundary name is therefore exhausted by the visible inscription rows and their sibling surfaces. □

Theorem 878.5 (FableMachineBoundary obligation closure)**.** *The* $\mathsf{FableMachineBoundary}^{\uparrow}$ *chapter has the following obligation surface: carrier admission for* $B = (H, E, L, S, W, K, T, C, P, N)$*, empty-boundary admission through* E*, selector-ledger exactness through* L *and* S*, branch and clock compatibility through* W *and* K*, inscription non-escape by Definition 878.4, and consumer non-escape by Definition 878.3. These obligations are finite rows over* BHist, Ext, $\mathsf{EmptyFableLedger}^{\uparrow}$, $\mathsf{FableBranchWitness}^{\uparrow}$, $\mathsf{FableMachineClock}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Proof. Carrier admission and empty-boundary admission are the first two projections of Definition 878.1: the displayed history H and empty-boundary slot E determine the boundary at which a read can occur.

Selector-ledger exactness and branch/clock compatibility are the middle projections. Rows L and S display the selector ledger and enacted selector, while rows W and K tie that selector to the local branch witness and finite clock row.

Inscription non-escape is supplied by Definition 878.4. Consumer non-escape is supplied by Definition 878.3. The remaining rows T, C, P, N only transport, replay, source, and name this finite route, so the listed clauses exhaust the obligation surface. □

Theorem 878.6 (FableMachineBoundary scoped kernel route)**.** *Every boundary, inscription, selector, branch, and clock consumer accepted by a* $\mathsf{FableMachineBoundary}^{\uparrow}$ *packet factors through the displayed rows*

$$H, E, L, S, W, K, T, C, P, N$$

and through the sibling surfaces $\mathsf{EmptyFableLedger}^{\uparrow}$, $\mathsf{FableBranchWitness}^{\uparrow}$, $\mathsf{FableMachineClock}^{\uparrow}$, *and* $\mathsf{SelectorLedger}^{\uparrow}$. *The closed scope is limited to the generated* BHist *row, the displayed* Ext *edge, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$ *naming, and the four sibling surfaces; it does not include a hidden chooser row, observer-substance coordinate, host trace quotient, global branch policy, or truth oracle.*

Proof. Begin with the obligation-closure theorem Definition 878.5. It fixes carrier admission, empty-boundary admission, selector-ledger exactness, branch and clock compatibility, inscription non-escape, and consumer non-escape as the complete finite obligation surface over $H, E, L, S, W, K, T, C, P, N$.

Next apply the selector non-escape theorem Definition 878.3. It confines every selector-facing read to the displayed ledger row L, enacted selector row S, fable-branch witness W, local clock K, and the structural rows that transport and name them.

Finally use the inscription non-escape theorem Definition 878.4. Its sibling route through $\mathsf{EmptyFableLedger}^{\uparrow}$, $\mathsf{FableBranchWitness}^{\uparrow}$, $\mathsf{FableMachineClock}^{\uparrow}$, and $\mathsf{SelectorLedger}^{\uparrow}$ supplies the same finite surface for boundary inscriptions, so no consumer route leaves the displayed kernel scope. □

Falsifiable prediction: Any accepted FableMachineBoundaryUp packet factors through the displayed empty-boundary selector ledger and admits no consumer route to a hidden chooser row.

Independence witness: Nearest siblings: empty-fable-ledger, fablebranchwitness, fablemachineclock, selectorledger, emptyboundaryselector. FableMachineBoundaryUp is not bijective to any of them because it simultaneously carries the empty-boundary slot, enacted selector row, and no-hidden-chooser boundary; each sibling lacks at least one of those obligation rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FableMachineBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FableMachineBoundary}^{\uparrow}$ packet is generated from a displayed Hist row, empty-boundary slot, finite selector ledger, enacted selector row, fable-branch witness, local fable-clock row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FableMachineBoundary}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 878.6 closes the displayed boundary, inscription, selector, branch, and clock consumer scope over $H, E, L, S, W, K, T, C, P, N$ and the sibling Empty-fable surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FableMachineBoundary}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FableMachineBoundaryUp.FableMachineBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close standard bridging, global branch selection, host trace equality, observer-substance, or truth-oracle semantics.

Upgrade path: Scoped closure requires checked classifier-stability and field-faithfulness for the displayed finite boundary packet.

Chapter 879

A Concrete Naming Certificate for $\mathsf{BetaReductionAuditTrail}^{\uparrow}$

The $\mathsf{BetaReductionAuditTrail}^{\uparrow}$ packet realizes the beta audit map of chapter 9384 as a finite audit trail. It sits beside the typed congruence trigger of chapter 760, the beta audit packet of chapter 761, the beta-step boundary of chapter 754, and the subject-reduction discharge socket of chapter 719. Those siblings expose trigger, audit, boundary, and discharge surfaces; the present packet records the ordered finite trail by which a consumer reads them without turning reduction into typed preservation.

Definition 879.1 (Beta-reduction audit-trail carrier)**.** *A* $\mathsf{BetaReductionAuditTrail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, V, O, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the ordered* BetaStep *ledger, separating the direct redex row from application, lambda, and Pi congruence rows;*
2. *V is the* BetaConv *common-reduct ledger read after the single-step ledger and before any typing-boundary row;*
3. *O is the obstruction-trigger ledger for beta substitution, application-argument stability, lambda-domain preservation, and Pi-domain preservation;*
4. *D is the subject-reduction discharge-socket row that records whether a typed-preservation discharge is present for the displayed trigger;*
5. *H stores componentwise* hsame *transports for the step, conversion, obstruction, discharge, route, provenance, and name rows;*
6. *C stores the displayed* Cont *route through the ordered stages S, V, O, D;*
7. *P is the* Pkg *provenance over chapter 9384, chapter 760, chapter 761, chapter 754, chapter 719,* BHist, hsame, Cont, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaReductionAuditTrail}^{\uparrow}}$ *row.*

The carrier contains no direct typing-preservation row, no closed subject-reduction theorem, no hidden conversion rule for typing, and no terminal audit escape outside H, C, P, N.

Lean4 ✓ `BEDC.Derived.BetaReductionAuditTrailUp.BetaReductionAuditTrailTasteGate single carrier alignment`

Theorem 879.2 (Beta-reduction audit-trail NameCert obligations)**.** *For the carrier of Definition 879.1, the local* $\mathsf{NameCert}_{\mathsf{BetaReductionAuditTrail}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (S, V, O, D, H, C, P, N)$*, beta-step stage readback through* S*, common-reduct conversion readback through* V*, obstruction-trigger readback through* O*, subject-reduction socket readback through* D*, componentwise* hsame *transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and terminal naming through* N*.*

Lean4 ✓ `BEDC.Derived.BetaReductionAuditTrailUp.BetaReductionAuditTrailNameCertObligations`

Proof. Project the carrier of Definition 879.1. Its only stage coordinates are the beta-step ledger S, the conversion ledger V, the obstruction ledger O, and the discharge-socket row D.

Read the siblings in the displayed order. The typed congruence trigger supplies the constructor-facing boundary, the MetaCIC beta audit packet separates reduction and conversion, the beta-step boundary keeps obstruction triggers visible, and the subject-reduction discharge socket records whether a typed preservation discharge has been supplied.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite trail. Since no coordinate contains an unguarded typing rule, a closed subject-reduction theorem, or an external audit ledger, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 879.3 (Beta-reduction audit-trail Stage F falsifiability)**.** *Every accepted* $\mathsf{BetaReductionAuditTrail}^{\uparrow}$ *packet has a falsifiable terminal ledger: a consumer that reads a direct typing-preservation row must also read the subject-reduction discharge socket* D *for the same ordered trail* S, V, O, D*. If* D *is absent, the accepted trail may still expose beta-step, conversion, and obstruction-trigger rows, but it cannot export typed preservation.*

Proof. Start from the ordered carrier. A typing-facing read can only occur after the consumer has passed through S, then V, then O, because those are the displayed step, conversion, and obstruction stages.

The only row after the obstruction stage that can mention typed-preservation discharge is D. The transport, continuation, package, and name rows H, C, P, N keep the same coordinate order and do not add a new discharge row.

Thus any accepted trail lacking D lacks a direct typing-preservation read. The packet remains readable as a beta audit trail, but its terminal ledger falsifies any attempted escape from reduction and conversion into typed preservation. □

Falsifiable prediction: Every accepted trail lacks a direct typing-preservation row unless a subject-reduction discharge socket is present.

Independence witness: typedbetacongruencetrigger, metacicbetaaudit, betastepboundary, subjectreductiondischargesocket: the trail carrier has ordered reduction-conversion-obstruction stages and a terminal audit ledger, while each sibling omits at least one stage or the terminal non-escape ledger, so no carrier bijection preserves the displayed NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BetaReductionAuditTrail}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BetaReductionAuditTrail}^{\uparrow}$ packet is generated from an ordered BetaStep ledger, a BetaConv common-reduct ledger, obstruction-trigger rows, a subject-reduction discharge-socket row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BetaReductionAuditTrail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 879.1, Definition 879.2, and Definition 879.3 bind the finite ordered beta audit trail and its terminal falsifiability ledger.

Formal verification: FormalStatus(NameCert$_{\mathsf{BetaReductionAuditTrail}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close typed beta preservation, closed subject reduction, conversion typing, confluence, normalisation, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ordered stage readback, discharge-socket readback, transport, route replay, provenance, naming, and terminal falsifiability for the same finite trail.

Chapter 880

A Concrete Naming Certificate for $\mathsf{TypedExampleAuditBudget}^{\uparrow}$

The $\mathsf{TypedExampleAuditBudget}^{\uparrow}$ packet realizes the TypedExamples audit-map vision of chapter 9382 as a finite regression-spend certificate. It sits beside the MetaCIC gallery of chapter 775, the typed-example gallery of chapter 756, the worked-example seal of chapter 753, and the beta-step boundary of chapter 754: those siblings expose gallery rows, checker rows, seal rows, and reduction-boundary rows, while this chapter records which accepted rows have regression budget and which consumer spends may read that budget.

Definition 880.1 (TypedExampleAuditBudget carrier)**.** *A* $\mathsf{TypedExampleAuditBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, C, B, Z, R, H, K, P, N)$$

with the following displayed rows:

1. *G is the finite gallery-row inventory inherited from the accepted TypedExamples and MetaCIC gallery packets;*
2. *C is the checker-row inventory recording the accepted local type-checker reads for the same gallery rows;*
3. *B is the beta-row inventory recording the selected beta computations and boundary steps that the gallery spends;*
4. *Z is the closedness-row inventory recording the closed term, context, and well-scoped inputs required by the checker rows;*
5. *R is the regression-budget row, naming the accepted gallery rows that count as covered and the consumer-spend boundary attached to each row;*
6. *H records componentwise* hsame *transports for gallery, checker, beta, closedness, regression-budget, provenance, and local naming rows;*
7. *K records the displayed* Cont *routes by which downstream checker-audit consumers reach exactly the rows G, C, B, Z, R;*
8. *P is the* Pkg *provenance over the sibling gallery, seal, beta-boundary, transport, continuation, and regression-budget rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedExampleAuditBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global checker-completeness theorem, a subject-reduction theorem, a normalization theorem, an arbitrary beta closure, or an unbudgeted consumer spend.

Theorem 880.2 (TypedExampleAuditBudget NameCert obligations)**.** *For the carrier of Definition 880.1, the local* $\mathsf{NameCert}_{\mathsf{TypedExampleAuditBudget}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for* $A = (G, C, B, Z, R, H, K, P, N)$*;*
2. *gallery-row exactness through* G *against the sibling gallery packets of chapter 775 and chapter 756;*
3. *checker, beta, and closedness exposure through* C*,* B*, and* Z*;*
4. *regression-budget non-escape through* R*, including the consumer-spend boundary attached to accepted rows;*
5. *transport and provenance through* H*,* K*,* P*, and the local name row* N*.*

Proof. Carrier admission is projection of the displayed finite packet. Every public row that a checker-audit consumer may inspect is present before the local NameCert row is read.

Gallery exactness is sibling-bounded. The row G names the gallery entries, while C, B, and Z expose the checker, beta, and closedness evidence that the gallery and seal packets already make visible.

The regression-budget row R is separate from gallery acceptance. A consumer that reads an accepted gallery row must still pass through R to learn whether that row is counted as covered and which spend boundary it carries.

Finally H, K, P, and N bind the row reads to displayed transport, continuation, provenance, and local naming. Since no coordinate names a global checker theorem, subject reduction, normalization, arbitrary beta closure, or unbudgeted spend, the certificate cannot export one. □

Theorem 880.3 (TypedExampleAuditBudget regression non-escape)**.** *Every checker-audit consumer of an accepted* $\mathsf{TypedExampleAuditBudget}^{\uparrow}$ *packet can read gallery acceptance only together with the displayed checker, beta, closedness, regression-budget, transport, continuation, provenance, and local naming rows. Removing the regression-budget row* R *destroys the consumer's ability to distinguish accepted gallery entries from entries counted as regression coverage, and no sibling gallery, seal, or beta-boundary packet can reconstruct that distinction.*

Proof. Project an accepted carrier $A = (G, C, B, Z, R, H, K, P, N)$. The rows G, C, B, Z expose gallery acceptance, checker reads, selected beta computations, and closedness inputs, but only R records regression coverage and consumer-spend boundaries.

Use the transport and continuation rows H and K for downstream checker-audit reads. They route the consumer through the displayed coordinates and therefore preserve the separation between gallery admission and regression coverage.

The sibling packets do not identify these two readings. The MetaCIC and typed-example galleries name accepted rows, the seal names worked-example closure of the gallery surface, and the beta-step boundary names reduction-boundary rows; none of them carries the regression-budget coordinate R together with its spend boundary. Thus removing R erases precisely the coverage distinction claimed by the packet. □

Falsifiable prediction: If the regression-budget row is removed from an accepted TypedExampleAuditBudget packet, a downstream checker-audit consumer can no longer distinguish gallery acceptance from regression coverage.

Independence witness: Nearest siblings: metacictypedexamplesgallery, typedexamplegallery, metacic-typedexamples-seal, betastepboundary. The budget packet is not bijective with any sibling because it carries a regression-budget row and a consumer-spend boundary that gallery and seal packets do not expose, while BetaStepBoundary carries reduction-boundary rows rather than typed-example gallery spend rows.

The proposed Lean follow-up for namespace BEDC.Derived.TypedExampleAuditBudgetUp places the taste-gate surface in BEDC.Derived.TypedExampleAuditBudgetUp.TasteGate. Its file lean4/BEDC/Derived/Typed includes both Nontrivial TypedExampleAuditBudget$^{\uparrow}$ with an explicit witness pair and FieldFaithful TypedExampleAud with mk-field projections and an injection proof.

Closure status for NameCert$_{\text{TypedExampleAuditBudget}^{\uparrow}}$ ·

Constructive story: A TypedExampleAuditBudget$^{\uparrow}$ packet is generated from finite gallery rows, checker rows, beta rows, closedness rows, regression-budget rows, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TypedExampleAuditBudget}^{\uparrow}}$, seed)

Scope closed: Definition 880.1, Definition 880.2, and Definition 880.3 bind the finite gallery, checker, beta, closedness, regression-budget, consumer-spend, transport, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{TypedExampleAuditBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close checker completeness, subject reduction, normalization, global beta closure, arbitrary gallery soundness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gallery-row exactness, checker exposure, beta exposure, closedness exposure, regression-budget non-escape, consumer-spend boundary visibility, componentwise transport, displayed continuation, provenance, and local naming rows.

Chapter 881

A Concrete Naming Certificate for $\mathsf{BetaCriticalPair}^{\uparrow}$

The $\mathsf{BetaCriticalPair}^{\uparrow}$ packet concretizes the beta audit map of chapter 9384 as a finite local-overlap certificate. It sits beside the closed-normal route certificate of chapter 858, the beta-reduction audit trail of chapter 879, and the beta-step boundary of chapter 754: those siblings display normal-route, ordered audit, and step-boundary surfaces, while this packet displays two competing beta branches and the common-reduct witness that a local-join consumer may read.

Definition 881.1 (Beta critical-pair carrier)**.** *A* $\mathsf{BetaCriticalPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, L, R, J, B, H, C, P, N)$$

with the following displayed rows:

1. *O is the local-overlap row identifying the shared redex context and the finite neighborhood in which the two beta branches are compared;*
2. *L is the left branch* BetaStep *readback from the overlap row;*
3. *R is the right branch* BetaStep *readback from the same overlap row;*
4. *J is the displayed common-reduct witness, read only after both branch rows have been displayed;*
5. *B is the consumer-boundary row separating local joining from subject reduction, closed confluence, normalization, or typed preservation;*
6. *H stores componentwise* hsame *transports for the overlap, branch, common-reduct, boundary, route, provenance, and name rows;*
7. *C stores the displayed* Cont *routes by which a consumer reads O, then the two branch readbacks L, R, then the common-reduct witness J, and finally the boundary row B;*
8. *P is the* Pkg *provenance over chapter 9384, chapter 858, chapter 879, chapter 754,* BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BetaCriticalPair}^{\uparrow}}$ *row.*

The carrier has no coordinate for subject reduction, closed Church–Rosser, global confluence, normalization, host evaluation, or a hidden join not displayed by J.

Lean4 ✓ `BEDC.Derived.BetaCriticalPairUp.BetaCriticalPairTasteGate single carrier alignment`

Theorem 881.2 (Beta critical-pair NameCert obligations)**.** *For the carrier of Definition 881.1, the local* $\mathsf{NameCert}_{\mathsf{BetaCriticalPair}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (O, L, R, J, B, H, C, P, N)$*, left branch readback through* L*, right branch readback through* R*, common-reduct witness readback through* J*, local non-escape through* B*, displayed continuation replay through* C*, componentwise classifier transport through* H*, package provenance through* P*, and terminal naming through* N*.*

Lean4 ✓ `BEDC.Derived.BetaCriticalPairUp.BetaCriticalPairNameCertObligations`

Proof. Project the carrier of Definition 881.1. The overlap row O is the only source row, and the branch rows L and R are both read from that same displayed neighborhood.

The common-reduct row J is downstream of the two branch rows. A consumer cannot read a joined endpoint before both branch readbacks have been displayed, because the continuation row C records the route through O, L, R, J in that order.

The local boundary row B then blocks escape from local joining into subject reduction, closed confluence, normalization, typed preservation, or host evaluation. The rows H, P, N transport, source, and name this same finite surface without adding a hidden join coordinate. □

Theorem 881.3 (Beta critical-pair local-join boundary)**.** *Every accepted* $\mathsf{BetaCriticalPair}^{\uparrow}$ *packet exposes a local join only through the displayed branch rows* L, R *and the common-reduct witness row* J*. A consumer may use the packet to read the common endpoint of the two local beta branches, but it may not reread the packet as subject reduction, closed Church–Rosser, normalization, conversion typing, or a global confluence theorem.*

Proof. Begin with an accepted packet $K = (O, L, R, J, B, H, C, P, N)$. The two branch rows share the same overlap row O, so the packet is a local-overlap certificate rather than an arbitrary pair of unrelated beta steps.

The join endpoint is exactly the row J. Since J is a displayed coordinate, local joining is falsifiable by inspecting the carrier: if the row is absent, the joined endpoint is absent from the packet.

Finally read the boundary row B. It records the non-escape side of the carrier and prevents any consumer from replacing the local common-reduct witness by subject reduction, global confluence, normalization, conversion typing, or host evaluation. □

Theorem 881.4 (Beta critical-pair local-join route)**.** *Every beta-audit consumer reading a* $\mathsf{BetaCriticalPair}^{\uparrow}$ *packet must pass through the shared overlap row* O*, both branch rows* L, R*, the displayed common-reduct witness* J*, and the non-escape boundary* B *before any local-join conclusion is available. The route realizes the finite local-overlap and common-reduct surface of chapter 9384 and is constrained by Definition 881.2 and Definition 881.3.*

Proof. Start with the NameCert obligations of Definition 881.2. They expose the shared overlap row O, the left and right branch readbacks L, R, the common-reduct witness J, and the local boundary row B as distinct displayed coordinates of the same carrier.

Apply Definition 881.3. It permits a local join only through the branch rows and the common-reduct row, while B blocks rereading that local fact as subject reduction, closed Church–Rosser, normalization, conversion typing, or global confluence.

The beta audit map of chapter 9384 is therefore reached through the finite route O, L, R, J, B and the structural rows H, C, P, N. No consumer can obtain a local-join conclusion from the packet before those overlap, branch, witness, and boundary rows have been displayed. □

Falsifiable prediction: If a BetaCriticalPairUp packet has no displayed common-reduct row, then no local-join consumer can read a joined endpoint from the packet.

Independence witness: closednormalroutecertificate, literalastequalityboundary, metaciciden-tity, selftheoryreflectionloop: the carrier has two reduction branches plus a common-reduct witness, while these siblings respectively carry normal-route, literal equality, identity, or reflection-loop rows and admit no bijection preserving both branch rows and the join row.

The Lean TasteGate obligation surface for this carrier requires a non-singleton carrier and faithful field exposure: it is discharged only by instances witnessing Nontrivial $\mathsf{BetaCriticalPair}^{\uparrow}$ and FieldFaithful $\mathsf{BetaCriticalPair}^{\uparrow}$ for the same finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BetaCriticalPair}^{\uparrow}}$.

Constructive story: A $\mathsf{BetaCriticalPair}^{\uparrow}$ packet is generated from a local-overlap row, two beta-step branch readbacks, a displayed common-reduct witness, a local non-escape boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BetaCriticalPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 881.1, Definition 881.2, and Definition 881.3 bind the finite local-overlap carrier, the two branch readbacks, the common-reduct witness, and the consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BetaCriticalPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, closed Church–Rosser, global confluence, normalization, conversion typing, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch readback exactness, common-reduct witness readback, local non-escape, route replay, provenance, local naming, and the Nontrivial and FieldFaithful TasteGate instances for the same finite carrier.

Chapter 882

A Concrete Naming Certificate for $\mathsf{ObserverDistinctionTrace}^{\uparrow}$

The $\mathsf{ObserverDistinctionTrace}^{\uparrow}$ packet realizes the distinction-trace face of chapter 9418 as a finite BEDC carrier. It reads observer-history identity through chapter 479, the subject boundary through chapter 615, and local stream rows through chapter 603. Unlike those siblings, it stores the ordered distinction-step trace itself: the displayed Hist growth rows, the Cont route rows that generate them, the hsame transports that preserve their order, and the no-subject provenance that prevents the trace from exporting an observer-substance coordinate.

Definition 882.1 (ObserverDistinctionTrace carrier)**.** *An* $\mathsf{ObserverDistinctionTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (H, T, K, C, L, P, N)$$

with the following displayed rows:

1. *H is the observed* Hist *source row whose distinction trace is being read;*
2. *T is the finite ordered trace of* $\mathsf{E1}$ *and* $\mathsf{E0}$ *distinction-step rows above the source surface;*
3. *K is the trace-growth ledger, recording the successive* Hist *rows reached by the ordered distinction steps;*
4. *C records the displayed* Cont *routes that read each step of T into the corresponding growth row in K;*
5. *L records componentwise* hsame *transport for the source, trace, growth, route, provenance, and local naming rows;*
6. *P is the* Pkg *provenance over* $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, $\mathsf{LocalClockBudget}^{\uparrow}$, BHist, Hist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverDistinctionTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for an observer-substance, a host equality of observers, a global clock, or a trace value detached from the displayed Hist *and* Cont *rows.*

Theorem 882.2 (ObserverDistinctionTrace NameCert obligations)**.** *For the carrier of Definition 882.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverDistinctionTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for $D = (H, T, K, C, L, P, N)$, ordered distinction-step exposure,* Hist *growth-row exactness, displayed* Cont *route replay, componentwise* hsame *transport, no-subject provenance, and local naming exactness.*

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.ObserverDistinctionTraceNameCert obligations`

Proof. Carrier admission is projection from the finite packet. The rows H, T, K, C, L, P, N display the observed source, the ordered distinction steps, the growth ledger, the continuation routes, transport, provenance, and local name.

The ordered trace rows T and K are read together. Each listed E1 or E0 step is attached to a displayed growth row, so the packet records a finite trace rather than an unindexed observer state.

The route row C supplies the only permitted reads of those steps, and L preserves the same trace componentwise under hsame. The package row P ties the trace to observer-history identity, subject-boundary exclusion, local stream, and clock-budget surfaces without adding any new observer-substance coordinate. □

Theorem 882.3 (ObserverDistinctionTrace temporal factorization)**.** *Every accepted* ObserverDistinctionTrace$^{\uparrow}$ *consumer route factors through the displayed temporal rows*

$$H \to T \to K \to C$$

and then through the transport, provenance, and local naming rows L, P, N. Thus the trace is read as ordered Hist *growth by distinction steps and displayed* Cont *routes, not as an ambient time parameter or a detached clock value.*

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.ObserverDistinctionTrace temporal factorization`

Proof. Project the carrier rows. The only temporal data are the source history H, the ordered distinction list T, the growth ledger K, and the displayed continuation routes C.

Use the Hist time-stream sibling of chapter 603 and the local clock-budget sibling of chapter 665. They read time through finite Cont rows and local step budgets, so a consumer of the distinction trace must pass through the same displayed rows.

Finally transport and name the route through L, P, N. Since the carrier contains no global clock or detached time coordinate, every accepted temporal read factors through the displayed Hist and Cont trace rows. □

Theorem 882.4 (ObserverDistinctionTrace non-escape)**.** *Every accepted* ObserverDistinctionTrace$^{\uparrow}$ *packet exports only the finite rows*

$$H, T, K, C, L, P, N.$$

No consumer can extract from the packet an observer-substance carrier, subject identity row, hidden owner of the trace, host equality of observers, global clock, or trace coordinate outside the displayed distinction-step ledger.

Proof. Begin with the observer-history identity sibling. chapter 479 permits observer-facing reads only through displayed history, observation, transport, route, package, and naming rows.

Next apply the subject-boundary sibling. chapter 615 excludes an additional subject carrier behind the observation rows, so the distinction trace cannot be assigned to a hidden owner outside Hist.

The remaining carrier coordinates are exactly H, T, K, C, L, P, N. Thus any accepted consumer receives a finite ordered distinction-step packet and no observer-substance, host equality, global clock, or off-packet trace object. □

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.ObserverDistinctionTrace nonescape`

Theorem 882.5 (ObserverDistinctionTrace TasteGate obligation)**.** *The later Lean target in* `lean4/BEDC/Derived` *must provide* Nontrivial ObserverDistinctionTrace$^{\uparrow}$ *and* FieldFaithful ObserverDistinctionTrace$^{\uparrow}$ *instances. The witnesses must distinguish the source* Hist *row, ordered distinction trace, growth ledger, displayed* Cont *routes,* hsame *transport, no-subject provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.taste gate`

Proof. Nontriviality is forced by the ordered trace shape. A packet that collapses the source, step ledger, and growth rows cannot witness a distinction trace.

Field faithfulness is forced by the carrier inventory. Each projection must preserve the source row, distinction-step rows, growth rows, route rows, transport row, provenance row, and local name separately so consumers can reject any read that bypasses the displayed trace. □

Theorem 882.6 (ObserverDistinctionTrace public route)**.** *Every public* ObserverDistinctionTrace$^{\uparrow}$ *read factors through* ObserverHistIdentity$^{\uparrow}$*,* ObserverSubjectBoundary$^{\uparrow}$*,* HistTimeStream$^{\uparrow}$*, and* LocalClockBudget$^{\uparrow}$ *before exposing the ordered distinction-step trace. The exposed trace is then read only through the local carrier rows H, T, K, C, L, P, N: observed* Hist *source, ordered distinction steps, growth ledger, displayed* Cont *routes, componentwise* hsame *transport, no-subject provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.ObserverDistinctionTrace public route`

Proof. Start with the local carrier and NameCert obligations. They expose H, T, K, C, L, P, N as the whole trace surface, with T carrying ordered distinction steps, K carrying Hist growth, and C carrying replay routes.

The public route first passes through the sibling observer surfaces. ObserverHistIdentity$^{\uparrow}$ supplies the history identity read, ObserverSubjectBoundary$^{\uparrow}$ prevents a hidden observer-substance coordinate, HistTimeStream$^{\uparrow}$ supplies finite stream rows, and LocalClockBudget$^{\uparrow}$ supplies the local step budget before the distinction trace is exported.

Apply the temporal factorization, non-escape, and TasteGate rows already in this chapter. They keep growth, replay, transport, no-subject provenance, and local naming tied to the same finite packet, so a downstream observer-history consumer receives an ordered distinction-step trace rather than host observer identity, global time, or subject substance. □

Theorem 882.7 (ObserverDistinctionTrace bridge surface)**.** *Every public* ObserverDistinctionTrace$^{\uparrow}$ *read bridges to the standard acceptance schema only through* ObserverHistIdentity$^{\uparrow}$*,* ObserverSubjectBoundary$^{\uparrow}$*,* HistTimeStream$^{\uparrow}$*,* LocalClockBudget$^{\uparrow}$*, and the local rows H, T, K, C, L, P, N. The bridge exports no host observer identity, global time, observer-substance coordinate, or trace read outside the displayed* Hist *growth,* Cont *replay, componentwise* hsame *transport, no-subject provenance, and local naming surface.*

Proof. Start from the public route of Definition 882.6. Its only public entry points are the observer-history identity sibling, the observer-subject boundary sibling, the Hist time-stream sibling, and the local clock-budget sibling.

The local bridge rows are exactly H, T, K, C, L, P, N. The rows H, T, K, C provide the displayed Hist source, ordered distinction steps, growth ledger, and continuation replay; L, P, N provide transport, no-subject provenance, and local naming.

The checked TasteGate and public-route targets record the same finite field faithfulness and public factorization: `BEDC.Derived.ObserverDistinctionTraceUp.taste_gate` and `BEDC.Derived.ObserverDi`
Therefore the acceptance bridge has no coordinate from which to recover host observer identity, global time, observer substance, or an off-packet trace value. □

Falsifiable prediction: Any accepted ObserverDistinctionTraceUp trace factors through the displayed Hist and Cont rows and cannot export an observer-substance coordinate.

Independence witness: observerhistidentity, observersubjectboundary, histtimestream, localclockbudget: the packet records ordered distinction-step trace rows rather than identity comparison, subject-boundary exclusion alone, local time stream alone, or clock-budget rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverDistinctionTrace}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverDistinctionTrace}^{\uparrow}$ packet is generated from an observed Hist source row, an ordered finite distinction-step trace, a Hist growth ledger, displayed Cont routes, componentwise hsame transport, no-subject package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverDistinctionTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 882.1, Definition 882.2, Definition 882.3, Definition 882.4, Definition 882.5, Definition 882.6, Definition 882.7,

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.taste gate`

, and

Lean4 ✓ `BEDC.Derived.ObserverDistinctionTraceUp.ObserverDistinctionTrace public route`

bind the BHist, Hist, Cont, hsame, Pkg, and NameCert dependency surface together with the observer-history identity sibling, subject-boundary sibling, local stream sibling, public trace route, and standard acceptance bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverDistinctionTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, host observer identity, global time, or an observer-substance behind the trace.

Upgrade path: No further paper-side closure upgrade remains for the displayed observer distinction trace; Lean verification and bridge checking remain independent verification-axis obligations.

Chapter 883

A Concrete Naming Certificate for $\mathsf{PrefixObserver}^{\uparrow}$

The $\mathsf{PrefixObserver}^{\uparrow}$ surface realizes the finite-observation side of chapter 9397. It packages an inscription-facing prefix read as a finite BHist packet with a displayed prefix row, an accepted observation route, a check row, a consumer ledger, and local NameCert provenance. It is adjacent to the accepted inscription packet of chapter 519, but it stops one step earlier: it certifies the prefix read that a downstream $\mathsf{InscriptionEvent}^{\uparrow}$ handoff may consume.

Definition 883.1 (PrefixObserver carrier)**.** *A* $\mathsf{PrefixObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, W, R, K, U, L, H, C, Q, N)$$

with the following displayed rows:

1. *S is the source observation-history row whose finite prefix is being read;*
2. *W is the displayed prefix-window row selected from S;*
3. *R is the accepted observation-route row from S to W;*
4. *K is the accepted check row certifying that the route really reads the displayed prefix;*
5. *U is the consumer row naming the inscription-facing reader of the prefix;*
6. *L is the consumer-ledger row recording what the reader may carry forward from the prefix;*
7. *H records componentwise* hsame *transports for the source, prefix, route, check, consumer, ledger, package, and local naming rows;*
8. *C records the displayed* Cont *routes by which consumers read S, W, R, K, U, L without leaving the finite packet;*
9. *Q is the* Pkg *provenance row over* BHist, hsame, Cont, *accepted-prefix checks, consumer-ledger data, and local certificate dependencies;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixObserver}^{\uparrow}}$ *row.*

The classifier for two accepted prefix observers compares only the displayed source row, prefix row, accepted route, check row, consumer row, ledger row, and their hsame, Cont, Pkg, *and* NameCert *provenance.*

Lean4 ✓ `BEDC.Derived.PrefixObserverUp.PrefixObserverTasteGate single carrier alignment`

Theorem 883.2 (PrefixObserver NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PrefixObserver}^{\uparrow}}$ *surface for the carrier of Definition 883.1 consists of five finite obligations:*

1. *carrier exposure: every accepted prefix observer exposes exactly the packet* $P = (S, W, R, K, U, L, H, C, Q, N)$*;*
2. *prefix locality: public comparison is restricted to the source, displayed prefix, accepted route, check, consumer, ledger, transport, continuation, package, and local naming rows;*
3. *check acceptance: the route row* R *reaches the displayed prefix* W *from the source row* S *only through the accepted check row* K*;*
4. *consumer accountability: every inscription-facing use first reads the consumer row* U*, the ledger row* L*, and the displayed* Cont *routes in* C*;*
5. *provenance exposure:* Q *and* N *bind the prefix read to finite* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data rather than to a hidden observation oracle.*

Lean4 ✓ `BEDC.Derived.PrefixObserverUp.PrefixObserverNameCert obligations`

Proof. Carrier exposure is projection from Definition 883.1. The source, prefix window, accepted route, check, consumer, ledger, transport, continuation, package, and local naming rows are all displayed coordinates.

Prefix locality follows from the classifier clause in the carrier definition. The comparison has no coordinate beyond the listed public rows and their recorded hsame and Cont routes, so it cannot inspect an undisplayed observation stream.

Check acceptance and consumer accountability are read from R, K, U, L, and C. The route reaches the displayed prefix only after the checker row accepts it, and every downstream consumer receives the ledgered prefix through the displayed continuation surface. □

Theorem 883.3 (PrefixObserver finite-route non-escape)**.** *For an accepted* $\mathsf{PrefixObserver}^{\uparrow}$ *packet*

$$P = (S, W, R, K, U, L, H, C, Q, N),$$

every public route to the observed prefix factors through $S, W, R, K, U, L, H, C, Q, N$*. The route may expose the selected prefix and the consumer ledger, but it cannot export an infinite stream, an undisplayed observation row, an unchecked route, a private observer state, or a host-level inspection oracle.*

Proof. Start from the carrier inventory. The only rows available to a public read are the source history, displayed prefix, route, check, consumer, ledger, transport, continuation, provenance, and local naming rows.

Apply the check-acceptance obligation from Definition 883.2. It forces any public read of W through R and K, so a consumer cannot bypass the finite accepted route or replace it with an unchecked stream read.

Finally use consumer accountability and provenance exposure. The ledger L, routes C, package row Q, and local name row N carry only the displayed prefix packet, which leaves no coordinate for an infinite stream, private observer state, or host-level inspection oracle. □

Theorem 883.4 (PrefixObserver inscription handoff)**.** *Every accepted inscription-facing prefix read hands data to* $\mathsf{InscriptionEvent}^{\uparrow}$ *consumers only through the displayed prefix and check rows before the inscription-event handoff. In particular, a consumer may pass* $S, W, R, K, U, L, H, C, Q, N$ *toward the inscription surface of chapter 519, but it receives no unselected history suffix, unchecked route, hidden observation source, private agency predicate, or raw host-event trace.*

Proof. Project the packet P from Definition 883.1. The displayed prefix row W and accepted check row K are explicit coordinates, and U names the inscription-facing consumer.

Use finite-route non-escape from Definition 883.3. The consumer reaches the prefix only through the accepted route and the displayed check, then reads the consumer ledger through C, Q, N.

The handoff to chapter 519 is therefore a finite packet handoff rather than a stream handoff. The inscription event may consume the prefix, route, check, and ledger rows, but no coordinate supplies an unselected suffix, hidden observer source, private agency predicate, or raw host trace. □

Theorem 883.5 (PrefixObserver StreamName schedule link)**.** *Every accepted* $\mathsf{PrefixObserver}^{\uparrow}$ *prefix window* W *may be read as a finite observation schedule row compatible with the* $\mathsf{StreamName}^{\uparrow}$ *finite-window discipline before it is handed to* $\mathsf{InscriptionEvent}^{\uparrow}$*. For*

$$P = (S, W, R, K, U, L, H, C, Q, N),$$

the rows $S, W, R, K, U, L, H, C, Q, N$ *expose the source, displayed prefix window, accepted route, check, consumer, ledger, transport, continuation, package, and local name needed for that finite schedule. The link exports no infinite stream, hidden suffix, unchecked route, private observer state, or host-level inspection oracle.*

Proof. Project the carrier of Definition 883.1. The displayed prefix window W is a finite coordinate of the accepted packet, and the rows R, K record the accepted route and check by which it is read from the source row S.

Use finite-route non-escape from Definition 883.3. It confines every public prefix read to $S, W, R, K, U, L, H, C, Q, N$, so the schedule row cannot inspect an unbounded observation stream or an undisplayed suffix.

Compare the finite schedule with the sibling discipline of chapter 126. The $\mathsf{StreamName}^{\uparrow}$ chapter supplies the finite-window reading convention, and the $\mathsf{PrefixObserver}^{\uparrow}$ packet supplies the concrete prefix window and inscription-facing ledger that instantiate it.

Finally apply the inscription handoff. The downstream $\mathsf{InscriptionEvent}^{\uparrow}$ consumer receives only the displayed prefix, route, check, ledger, and certificate rows, so no unchecked route, private observer state, or host-level inspection oracle is exported. □

Falsifiable prediction: If PrefixObserverUp is atomic, then every accepted inscription-facing prefix read factors through its displayed prefix and check rows before the InscriptionEventUp handoff.

Independence witness: Compare inscriptionevent, observersubjectboundary, observerhistidentity, finitewindowobserver, and streamname: PrefixObserverUp is not bijective to any one of them because it carries the accepted-prefix check and consumer handoff rows simultaneously.

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixObserver}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrefixObserver}^{\uparrow}$ packet is generated from finite BHist rows for an observed prefix, accepted-prefix check, observer-facing subject rows, inscription-event handoff, transports, continuations, provenance, and local naming. The observer read is produced by Cont through the displayed prefix and check rows before reaching the inscription-facing handoff, so consumers cannot bypass the finite prefix test. Componentwise hsame transports the prefix, check, observer, and handoff rows, while Pkg records the inscription, observer-boundary, observer-identity, finite-window, and stream-name dependencies. The packet is atomic because it combines prefix checking and consumer handoff in one surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixObserver}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the PrefixObserver carrier, accepted-prefix check, InscriptionEvent handoff, falsifiable prediction, and sibling independence witness.

Formal verification: FormalStatus(NameCert$_{\mathsf{PrefixObserver}\uparrow}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: It does not claim completed streams, unrestricted observer subject identity, hidden prefix selection, host equality of histories, or a general observation scheduler.
Upgrade path: Obligation closure requires Lean encodings of the prefix carrier, accepted-prefix check, handoff route, transport stability, and sibling non-bijection boundary.

Chapter 884

A Concrete Naming Certificate for $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$

The $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$ packet composes two finite tail-agreement seals into one completion-facing transitivity route. It consumes the left-middle seal and the middle-right seal of chapter 699, shares the displayed middle $\mathsf{RegSeqRat}^{\uparrow}$ source and $\mathsf{StreamName}^{\uparrow}$ tail-window rows between them, reads the paired $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledgers, and exports only the resulting RealUp handoff row. The cofinal-tail route is the same finite discipline used by chapter 658, while the constructive motivation is the real-completion axiom-purity surface of section 9238.3.

Falsifiable prediction: A composed RealTailAgreementTransitivityUp read is invalid if its two RealTailAgreementSealUp inputs do not share the displayed middle RegSeqRat and StreamName tail-window rows.

Independence witness: Nearest siblings: realtailagreementseal, regularcauchytailfusion, regularcauchytailmeet, diagonalcofinaltail, regularlimituniqueness. The transitivity packet is not bijective to any of them because it carries two agreement inputs and a shared middle-tail compatibility row; the siblings carry a single agreement, fusion, meet, cofinal-tail, or uniqueness surface.

Definition 884.1 (Real tail-agreement transitivity carrier)**.** *A* $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, R_2, W_{01}, W_{12}, D_{01}, D_{12}, A_{01}, A_{12}, M, H, C, P, N)$$

with the following displayed rows:

1. R_0, R_1, R_2 *are the three* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose left-middle and middle-right tails are compared;*
2. W_{01} *and* W_{12} *are the two finite* $\mathsf{StreamName}^{\uparrow}$ *paired-window rows, with the* R_1 *side displayed as the same middle tail-window row in both seals;*
3. D_{01} *and* D_{12} *are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback ledgers for those paired windows;*
4. A_{01} *and* A_{12} *are the two accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *rows, the first for* R_0, R_1 *and the second for* R_1, R_2*;*
5. M *is the middle-tail compatibility row recording that the* R_1 *source, its* $\mathsf{StreamName}^{\uparrow}$ *window, and its dyadic readback are the same displayed data in both agreement inputs;*

6. *H stores componentwise* hsame *transport for the three sources, the two paired windows, both dyadic ledgers, both agreement rows, the middle compatibility row, provenance, and local name;*
7. *C stores the displayed* Cont *routes that read the left-middle seal, the middle-right seal, the shared middle rows, and the completion-facing handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailAgreementTransitivity}^{\uparrow}}$ *row.*

The carrier exports only the finite transitivity packet through these rows. It has no coordinate for quotient stream equality, host real equality, selected completed limits, unbounded tail search, open-cover compactness, or choice-selected tails.

Lean4 ✓ `BEDC.Derived.RealTailAgreementTransitivityUp.RealTailAgreementTransitivityTasteGate`
`single carrier alignment`

Theorem 884.2 (Real tail-agreement transitivity NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealTailAgreementTransitiv}}$ *surface for Definition 884.1 consists of carrier admission, two* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *input admissions, shared middle* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, paired finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback exactness rows, middle-tail compatibility, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, RealUp handoff, and local naming.*

Proof. Project the carrier T. Its displayed coordinates list the three regular rational sources, the two paired window rows, the two dyadic readback ledgers, the two agreement inputs, the shared middle row, and the structural rows.

Use chapter 699 for each agreement input. The first seal admits the finite route from R_0 to R_1, and the second seal admits the finite route from R_1 to R_2.

The middle-tail compatibility row M is the extra transitivity datum. It forces the R_1 source, its tail window, and its dyadic readback to be the same displayed rows in both seals, so the composition cannot splice unrelated middle evidence.

The remaining obligations are structural. The rows H, C, P, N transport, replay, record provenance, and name exactly the displayed packet, while the RealUp handoff is downstream of the two accepted seal inputs and the shared middle row. □

Theorem 884.3 (Real tail-agreement transitivity route)**.** *Every completion-facing read of an accepted* $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$ *packet factors through the finite route*

$$R_0, R_1, R_2, W_{01}, W_{12}, D_{01}, D_{12}, A_{01}, A_{12}, M, H, C, P, N.$$

The composed route is a finite three-source tail-window packet over two $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *rows and the shared middle-tail compatibility row, not an equality of quotient streams or a selected ambient real limit.*

Proof. Begin with a completion-facing consumer of the transitivity packet. The carrier gives the consumer no direct RealUp coordinate except the handoff routed through the two accepted agreement rows and the shared middle row.

Read the left-middle and middle-right inputs through chapter 699. Each input exposes only its finite $\mathsf{RegSeqRat}^{\uparrow}$ sources, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks, displayed agreement row, and structural route.

Apply the middle-tail compatibility row M. Since both agreement inputs display the same R_1 source-window-readback data, the two seal routes compose through one named middle tail rather than through a hidden selector.

Finally replay the route with H, C, P, N. This places the completion-facing read inside the displayed finite tuple and excludes quotient streams, host real equality, selected limits, unbounded tail search, open-cover compactness, and choice-selected tails. □

Theorem 884.4 (Real tail-agreement transitivity non-escape)**.** *An accepted* $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$ *packet cannot export quotient stream equality, host real equality, selected completed limits, unbounded tail search, open-cover compactness, choice-selected tails, or a hidden middle representative. Any such read is outside the packet unless it factors through the finite rows*

$$R_0, R_1, R_2, W_{01}, W_{12}, D_{01}, D_{12}, A_{01}, A_{12}, M, H, C, P, N.$$

Proof. Inspect the carrier. The only comparison inputs are the two accepted $\mathsf{RealTailAgreementSeal}^{\uparrow}$ rows A_{01} and A_{12}, and the only bridge between them is the displayed middle-tail compatibility row M.

Use the route theorem above. Every completion-facing read must pass through the three source rows, the two paired window rows, the dyadic ledgers, the two agreement inputs, and the shared middle row before the structural handoff is consumed.

Coordinate exhaustion closes the claim. None of the listed rows is a quotient, a host equality proof, a selected limit, an unbounded search functional, an open-cover compactness witness, a choice row, or a second middle representative. □

The Lean TasteGate file for this packet must provide explicit Nontrivial and FieldFaithful instances as prerequisites for the carrier-alignment theorem. The intended target is

BEDC.Derived.RealTailAgreementTransitivityUp.RealTailAgreementTransitivityTasteGate_single_carrier_alignment.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailAgreementTransitivity}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailAgreementTransitivity}^{\uparrow}$ packet is generated from three RegSeqRat source rows, two paired StreamName tail-window rows, two DyadicRatCore readback ledgers, two RealTailAgreementSealUp inputs, a shared middle-tail compatibility row, componentwise hsame transports, Cont route replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailAgreementTransitivity}^{\uparrow}}$, seed)

Scope closed: Definition 884.1, Definition 884.2, Definition 884.3, and Definition 884.4 bind the finite three-source tail-agreement composition packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealTailAgreementTransitivity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient stream equality, host real equality, selected completed limits, unbounded tail search, open-cover compactness, choice-selected tails, or a public standard bridge.

Upgrade path: Obligation closure requires checked Lean carrier admission, two seal-input admissions, shared middle-tail compatibility, route composition, explicit Nontrivial and FieldFaithful TasteGate instances, RealUp handoff, and non-escape for the same finite packet.

This hub routes the refinement packet through the StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces of chapter 126, chapter 291, chapter 301, and chapter 38.

Chapter 885

A Concrete Naming Certificate for CauchyModulusRefinement$^{\uparrow}$

The CauchyModulusRefinement$^{\uparrow}$ packet concretizes the Real-completeness diagonal-modulus route of chapter 9238. It is assembled from the comparison and meet surfaces of chapter 636 and chapter 462, then routes extraction, selector-budget, finite-window, rational-readback, dyadic-tolerance, and Real$^{\uparrow}$ seal data through one displayed NameCert carrier.

Definition 885.1 (Cauchy modulus refinement carrier)**.** *A* CauchyModulusRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

with the following displayed coordinates:

1. *M_0 and M_1 are the two source* CauchyModulus$^{\uparrow}$ *schedules whose refinement is being compared;*
2. *U is the shared* CauchyModulusMeet$^{\uparrow}$ *row consumed by both source schedules;*
3. *V is the* CauchyModulusExtraction$^{\uparrow}$ *refinement witness row, read only after M_0, M_1 and U have been displayed;*
4. *T is the* RegularCauchySelectorBudget$^{\uparrow}$ *row that budgets the selected finite windows for the refinement handoff;*
5. *W is the shared tail-threshold and finite* StreamName$^{\uparrow}$ *window ledger selected by U, V, T;*
6. *Q stores the* DyadicRatCore$^{\uparrow}$ *tolerance rows and* RegSeqRat$^{\uparrow}$ *readback ledger attached to the same selected windows;*
7. *E is the* Real$^{\uparrow}$ *seal row reached after the shared meet, refinement witness, selector budget, windows, tolerance rows, and regular readbacks have been displayed;*
8. *H stores componentwise* hsame *transports for $M_0, M_1, U, V, T, W, Q, E, P, N$ and for the local packet shape;*
9. *C stores displayed* Cont *routes from the source schedules to U, from U to V, from V to T, from T to W, from W to Q, from Q to E, and from the packet rows to the local naming row;*
10. *P is the* Pkg *provenance over* CauchyModulus$^{\uparrow}$, CauchyModulusComparison$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no selected limit, no host real equality, no hidden selector, no second meet row, and no ambient completeness coordinate.

Lean4 (definition) `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier`

Theorem 885.2 (Cauchy modulus refinement NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *surface for Definition 885.1 consists of five finite obligation rows:*

1. *carrier admission for $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$;*
2. *classifier stability under the displayed* hsame *transports and* Cont *routes;*
3. *refinement-source exactness stating that the row V is read only from the two source schedules M_0, M_1 through the shared meet row U;*
4. *shared-window tolerance exactness forcing every selected-window read through T, W, Q;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape stating that E is reached only after the shared meet, refinement witness, selector budget, windows, dyadic tolerance rows, regular readbacks, transports, routes, provenance, and local name have been displayed.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier cofinal window factorization`

Proof. Carrier admission is projection from the finite packet in Definition 885.1. The tuple lists the two source schedules, shared meet row, refinement witness, selector-budget row, selected windows, tolerance and readback ledger, Real seal, transports, routes, provenance, and local name.

Classifier stability uses H and C. These rows are componentwise over the packet coordinates, so a transported refinement read remains a read of the same source schedules, meet row, witness row, budget row, window ledger, tolerance ledger, seal row, package row, and local certificate.

Refinement-source exactness follows from the order of the displayed coordinates and routes. The witness V is downstream of M_0, M_1, U, and no carrier coordinate stores an external modulus ordering, a second meet, or a hidden selector that could replace that source.

Finally, the shared-window and Real-seal clauses use T, W, Q, E. Every Real-facing read reaches E only through the selector-budget row, the shared finite-window ledger, and the tolerance-readback ledger, so quotient streams, selected limits, host real equality, and ambient completeness remain outside the packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier namecert obligations`

Theorem 885.3 (Cauchy modulus refinement carrier admission row)**.** *For every accepted* CauchyModulusRefinemen *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

carrier admission is exactly the finite BHist *admission of the displayed rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ together with their local* NameCert *row N. The admitted carrier cites the sibling* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces only through the coordinates displayed in Definition 885.1.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier carrier admission row`

Proof. Project the carrier definition. Its coordinate list is a finite BHist packet, and the final coordinate N is the local $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ row naming that same packet.

The bundled obligation theorem supplies the carrier-admission clause for exactly this tuple. It admits the two source schedules, the shared meet row U, the extraction witness V, the selector budget T, the finite window ledger W, the tolerance-readback ledger Q, the Real seal E, and the transport, route, package, and name rows.

The non-escape clause removes every unlisted carrier source. Hence carrier admission for R is projection from the displayed packet and not an admission of a quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness coordinate. □

Theorem 885.4 (Cauchy modulus refinement shared meet dependency row)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the refinement witness row* V *depends on the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row* U *by the displayed* Cont *route*

$$M_0, M_1 \longrightarrow U \longrightarrow V.$$

Thus any read of V *in the local* NameCert *surface also reads the two source schedules and the shared meet row before it reaches the* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *witness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared meet dependency`

Proof. Read the source portion of the packet. The coordinates place M_0 and M_1 before U, and the continuation row C records the routes from both source schedules to U and from U to V.

The bundled refinement-source exactness clause states that V is read only from the two displayed source schedules through U. This is the same dependency row used by the sibling $\mathsf{CauchyModulusMeet}^{\uparrow}$ and $\mathsf{CauchyModulusExtraction}^{\uparrow}$ surfaces.

Apply non-escape to the source side. Since no second meet row, external modulus ordering, hidden selector, or host equality coordinate belongs to the carrier, the shared meet row U is the only admitted dependency route into V. □

Theorem 885.5 (Cauchy modulus refinement selector budget dependency row)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, each selected-window read and each tolerance-readback read depends on the selector budget row* T *through*

$$V \longrightarrow T \longrightarrow W \longrightarrow Q.$$

The row T *is therefore the displayed* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *handoff into the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows used by the refinement.*

Proof. Project the window side of the carrier. The only selected-window coordinates are T for the selector budget, W for the finite $\mathsf{StreamName}^{\uparrow}$ ledger, and Q for the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows together with the $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger.

The bundled shared-window tolerance exactness clause forces each selected-window read through T, W, Q. The route row C places these rows after the extraction witness V and before the seal row E.

Use the readback non-escape content already stated for the packet. Because there is no hidden selector, second window ledger, quotient stream, selected limit, or ambient completeness coordinate, every selected-window and tolerance-readback read depends on the displayed selector budget row. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget dependency`

Theorem 885.6 (Cauchy modulus refinement selector-budget meet exhaustion)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

whose two source schedules share the meet row U *and selector-budget row* T*, every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read is exhausted by the route*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

No accepted consumer may replace U *by a second meet, replace* T *by a hidden selector, or replace* Q, E *by a quotient stream, selected host limit, host real equality, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget meet exhaustion`

Proof. First bind the source side. The shared-meet dependency row Definition 885.4 forces both source schedules M_0, M_1 to pass through the displayed meet row U before the extraction row V can be read.

Next bind the selector side. The selector-budget dependency row Definition 885.5 places the finite selector budget T between V and the selected-window and tolerance-readback rows W, Q.

Now attach the Real-facing route. The Real seal route row Definition 885.7 places E only after M_0, M_1, U, V, T, W, Q, and the non-escape theorem Definition 885.10 removes every unlisted meet, selector, quotient stream, host equality, selected limit, and ambient completeness coordinate from the accepted packet. □

Theorem 885.7 (Cauchy modulus refinement Real seal route row)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reached only along the displayed route*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

Consequently a completion-facing consumer may use E *only after the shared meet, extraction witness, selector budget, finite window ledger, dyadic tolerance rows, and regular readbacks have been displayed.*

Proof. First combine the source and selector routes. The shared meet dependency row places U before V, and the selector budget dependency row places T, W, Q after V.

The bundled $\mathsf{Real}^{\uparrow}$ seal non-escape clause then places E after the shared meet, refinement witness, selector budget, selected windows, tolerance rows, regular readbacks, transports, routes, provenance, and local name.

Thus the Real seal is not an independent Real-facing object. It is the final row of the displayed Cont chain from the source schedules through Q, stabilized by the componentwise hsame row H and recorded by the package row P and local name N. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal route`

Theorem 885.8 (Cauchy modulus refinement Real seal scope package)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of the refinement seal must pass through the two source schedules* M_0, M_1*, the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row* U*, the extraction witness* V*, the* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *row* T*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N*. The package exports no quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal scope package`

Proof. Project the carrier row in Definition 885.1. The accepted packet has exactly the displayed coordinates $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, with H, C, P, N recording componentwise transport, displayed route data, provenance, and the local name over the same finite carrier.

Apply Theorem 885.4. The shared meet row U is the only meet dependency between the two source schedules and the extraction witness V.

Apply Theorem 885.5. The selected-window and tolerance-readback side then passes through T, W, Q, so the selector budget, finite window ledger, dyadic tolerance rows, and regular readbacks are all explicit rows of the same route.

Apply Theorem 885.7. The seal row E is reachable only after the route through M_0, M_1, U, V, T, W, Q has been displayed and stabilized by the structural rows.

Finally use Theorem 885.10. Since the carrier has no coordinate for a quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness datum, a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer of the refinement seal receives only the public scope package listed above. □

Theorem 885.9 (Cauchy modulus refinement NameScope package row)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinem}}$ *namescope package consists exactly of the carrier-admission row, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* C*, the provenance row* P*, and the local naming row* N *over the same sibling surfaces. It exports no name for a row outside* $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier namescope package`

Proof. Project the naming coordinates of the carrier. The rows H, C, P, N are displayed alongside the source, meet, extraction, selector, window, tolerance, readback, and seal rows, so the local namescope is attached to the same finite packet as carrier admission.

Use classifier stability from the bundled obligation theorem. The componentwise hsame transports and Cont routes replay accepted reads of the same rows rather than creating a separate classifier or hidden carrier.

Finally combine the shared meet, selector budget, and Real seal route rows. They show that every named downstream consumer reaches E through the sibling $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces displayed in R. Non-escape then excludes every unlisted namescope target. □

Theorem 885.10 (Cauchy modulus refinement non-escape)**.** *An accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet exposes only the finite route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

If the refinement packet is accepted, every $\mathsf{Real}^{\uparrow}$*-facing read must factor through the displayed shared meet and selector-budget rows. Thus a counterpacket with the same seal row* E *but no shared meet row* U *refutes the chapter's concrete carrier claim.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier nonescape`

Proof. Read the carrier inventory. The only route from the two source schedules to the Real-facing seal passes through U, V, T, W, Q, E and is stabilized by H, C, P, N.

The shared meet row U is required before the refinement witness V can be read, and the selector-budget row T is required before the selected window ledger W and tolerance-readback ledger Q can be consumed. Hence a Real-facing read cannot bypass either U or T.

If a packet presents the same seal row but omits the shared meet row, its displayed coordinate list is not the carrier of Definition 885.1. Such a packet therefore witnesses failure of the concrete carrier claim rather than a second accepted refinement surface. □

Theorem 885.11 (Cauchy modulus refinement source meet budget chain)**.** *Every accepted refinement read of a* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

factors through the two source schedules M_0, M_1*, the shared meet row* U*, the refinement witness row* V*, and the selector-budget row* T *before any finite window ledger* W *is exposed. The route consumes the sibling* $\mathsf{CauchyModulusMeet}^{\uparrow}$*,* $\mathsf{CauchyModulusExtraction}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows named in the carrier, and it exports no selected window read before the source-meet and selector-budget chain is displayed.*

Proof. Project the packet R. The carrier lists the two source schedules first, then the shared meet row, then the refinement witness, then the selector-budget row.

Read the displayed continuation routes in C. They place M_0 and M_1 before U, place U before V, and place V before T.

Only after that route is displayed can W be read. The provenance row P names the meet, extraction, selector-budget, StreamName, DyadicRatCore, RegSeqRat, and Real rows as sibling dependencies, so a window read outside this route is not a read of the accepted refinement packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement source meet budget chain`

Theorem 885.12 (Cauchy modulus refinement window readback exhaustion)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, every selected-window read and every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumed by the refinement is exhausted by the displayed selector-budget, finite-window, and tolerance-readback rows*

$$T,\ W,\ Q.$$

The packet has no second window ledger, hidden selector, quotient stream, selected host limit, or ambient completeness coordinate from which a selected window or regular-rational readback may be recovered.

Proof. Start from the selected-window portion of the carrier. The only coordinates devoted to selected observation are T for the budget, W for the finite $\mathsf{StreamName}^{\uparrow}$ windows, and Q for the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows together with the $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger.

The continuation route forces these rows into the order T, W, Q. A readback request therefore reaches $\mathsf{RegSeqRat}^{\uparrow}$ only after the selected windows have already been read from the displayed StreamName ledger.

Apply the non-escape theorem. Since the carrier omits hidden selectors, quotient streams, selected limits, and ambient completeness, the T, W, Q rows exhaust all accepted selected-window and regular-readback access. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement window readback exhaustion`

Theorem 885.13 (Cauchy modulus refinement Real seal refusal)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal row* E *of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet refuses any packet that bypasses either the shared meet route through* U *or the selector-budget route through* T*. A purported seal read that presents* E *without the source schedules, shared meet, refinement witness, selector budget, selected windows, dyadic tolerance rows, regular readbacks, transports, routes, provenance, and local name is not a carrier read of Definition 885.1.*

Proof. First read the source side. The source meet budget chain places M_0, M_1, U, V, T before any selected-window or readback row can be exposed.

Next read the window and tolerance side. Window readback exhaustion forces the selected $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks through T, W, Q.

The only route into the $\mathsf{Real}^{\uparrow}$ seal is the displayed continuation from Q to E, stabilized by H and recorded by P, N. A packet that omits U or T therefore omits a required coordinate of the seal route and is refused as a $\mathsf{CauchyModulusRefinement}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal refusal`

Theorem 885.14 (Cauchy modulus refinement shared-route totality)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *factors through the single displayed route*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

Along this route the consumer obtains the shared meet, refinement witness, selector budget, selected windows, tolerance readback, and Real seal rows, but obtains no quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness datum.

Proof. Use shared-meet dependency, Definition 885.4, and selector-budget dependency, Definition 885.5, to bind the source side. They force every accepted consumer route to read M_0, M_1 before the shared meet U, and then to pass through the refinement witness V and selector-budget row T.

Apply window readback exhaustion, Definition 885.12. It places every selected-window and $\mathsf{RegSeqRat}^{\uparrow}$ readback through T, W, Q and leaves no hidden selector, quotient stream, selected limit, or second window ledger outside those rows.

Finally apply Real seal refusal, Definition 885.13. Since the seal row E refuses reads that bypass U or T, the only accepted $\mathsf{Real}^{\uparrow}$-facing route is the displayed chain ending in E, with host real equality and ambient completeness still outside the carrier. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared route totality`

Theorem 885.15 (Cauchy modulus refinement cofinal-window factorization)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *Real-facing or Completion-facing consumer factors through the same cofinal finite-window chain*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

The cofinal-window read is exhausted by the displayed $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows of the accepted packet. No quotient stream, selected host limit, host real equality, hidden selector, or ambient completeness datum supplies a second cofinal route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCofinalWindowFactorization`

Proof. First bind the source side by the shared-meet dependency and selector-budget dependency, Definition 885.4 and Definition 885.5. They force both source moduli M_0, M_1 to meet at U, pass through the extraction row V, and enter the selector-budget row T before any finite-window read is available.

Next apply window readback exhaustion, Definition 885.12. It accounts for the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows by routing selected windows and tolerance reads through T, W, Q.

Finally use the Real seal route and the non-escape boundary, Definition 885.13 and Definition 885.10. The seal row E accepts only the displayed route, while non-escape removes quotient streams, selected host limits, host real equality, hidden selectors, and ambient completeness data from the packet. Hence Real-facing and Completion-facing consumers share the same cofinal finite-window factorization. □

Theorem 885.16 (Cauchy modulus refinement classifier transport)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

classifier stability is transported componentwise by the hsame *row* H *and replayed by the displayed* Cont *routes in* C*:*

$$\mathsf{Cont}(M_0, U), \quad \mathsf{Cont}(M_1, U), \quad \mathsf{Cont}(U, V), \quad \mathsf{Cont}(V, T),$$
$$\mathsf{Cont}(T, W), \quad \mathsf{Cont}(W, Q), \quad \mathsf{Cont}(Q, E), \quad \mathsf{Cont}(M_0, M_1, U, V, T, W, Q, E, H, C, P, N).$$

Thus any accepted classifier read of R *remains a read of the same two source schedules, shared meet row, refinement witness, selector-budget row, finite-window ledger, tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, provenance row, and local* $\mathsf{NameCert}$ *row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement classifier transport`

Proof. Project the packet R. The only transport data in the carrier are the componentwise hsame entries stored in H, and the only replay data are the displayed Cont routes stored in C.

The first two routes place the source schedules M_0, M_1 before the shared meet row U. The next five routes carry the same read through V, T, W, Q, E, so a transported classifier cannot switch to a different witness, budget, window ledger, tolerance ledger, or seal row.

The final packet route authenticates the whole displayed tuple with P and N. Since the carrier has no hidden classifier coordinate, componentwise hsame transport and the displayed Cont route list exhaust the classifier transport surface. □

Definition 885.17 (Cauchy modulus refinement pullback spine)**.** *The pullback spine of an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet is the finite* BHist *tuple*

$$(M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

read as a single displayed route from the two source Cauchy-modulus schedules to the Real-facing seal. The row U is pulled back from the CauchyModulusMeet$^{\uparrow}$ *surface over M_0, M_1; the row V is pulled back from the* CauchyModulusExtraction$^{\uparrow}$ *witness over that meet; the row T is pulled back from the* RegularCauchySelectorBudget$^{\uparrow}$ *selector budget over V; the row W is the finite* StreamName$^{\uparrow}$ *window surface opened by T; the row Q is the joint* DyadicRatCore$^{\uparrow}$ *tolerance and* RegSeqRat$^{\uparrow}$ *readback ledger over W; and E is the* Real$^{\uparrow}$ *seal read only after Q. The structural rows H, C, P, N store the componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row for precisely this route.*

Lean4 (definition) `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementPullbackSpine`

Theorem 885.18 (Cauchy modulus refinement pullback spine admission)**.** *An accepted* CauchyModulusRefinement$^{\uparrow}$ *packet is admitted exactly through the pullback spine of Definition 885.17. Equivalently, carrier admission for*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

is the same finite BHist *admission obtained by reading the* CauchyModulusMeet$^{\uparrow}$ *row U,* CauchyModulusExtraction$^{\uparrow}$ *row V,* RegularCauchySelectorBudget$^{\uparrow}$ *row T,* StreamName$^{\uparrow}$ *finite windows W,* DyadicRatCore$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *ledger Q,* Real$^{\uparrow}$ *seal E, and structural rows H, C, P, N in the displayed order.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier pullback spine admission`

Proof. Project the carrier of Definition 885.1. Its visible coordinates are exactly the rows listed in the pullback spine, and the NameCert obligations of Definition 885.2 attach the local naming row to that same finite packet.

Read the source side through the shared meet dependency. The source-meet budget chain and source-meet exactness force the witness row V to enter only after M_0, M_1, U, so the meet row is not an external parameter of the accepted packet.

Read the downstream side through classifier transport. The classifier-transport theorem keeps every accepted read on the componentwise hsame and displayed Cont routes, while window readback exhaustion and Real seal factorization identify the only later rows as T, W, Q, E.

The reverse direction is repacking. A tuple admitted through the displayed meet, extraction, selector-budget, window, tolerance-readback, seal, transport, route, provenance, and naming rows supplies every coordinate required by the carrier and no coordinate outside it. Thus accepted packets and pullback-spine admissions coincide. □

Theorem 885.19 (Cauchy modulus refinement shared-budget pullback)**.** *For every accepted* CauchyModulusRefine *packet $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$, the meet row U and selector-budget row T jointly determine every selected-window read in W and every tolerance-readback row in Q. No selected window may be read from T without the same shared meet row U, and no* DyadicRatCore$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *readback may be read from Q without the window budget determined by U through V, T.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementSharedBudgetPullback`

Proof. First fix the shared source. Source-meet exactness identifies U as the only meet row over the two schedules M_0, M_1, and the source-meet budget chain places the refinement witness V after that meet.

Next fix the selector budget. Selector budget totality makes T the only route from V to the selected finite-window surface, and window readback exhaustion makes W, Q the only rows from which selected StreamName, dyadic tolerance, and regular-rational readback data can be consumed.

Finally use the classifier-transport and Real-seal route theorems. They keep the same hsame, Cont, Pkg, and NameCert rows attached to the route, so a selected-window consumer cannot detach T from U or replace Q by a tolerance ledger outside the displayed pullback spine. □

Theorem 885.20 (Cauchy modulus refinement Real seal pullback non-escape)**.** *A* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet cannot bypass any of the rows*

$$U,\ V,\ T,\ W,\ Q.$$

Every such consumer reaches the seal row E only after the shared meet, extraction witness, selector budget, finite StreamName windows, and dyadic and rational readback ledger have been read through the pullback spine, and it receives no quotient stream, selected host limit, hidden selector, second meet row, host real equality, or ambient completeness coordinate.

Proof. Begin at the seal. Real seal refusal rejects any attempted seal read that omits the shared meet route or the selector-budget route, and Real seal factorization places E after U, V, T, W, Q.

Now bind the consumer path. Classifier transport supplies the componentwise hsame and displayed Cont route, while the shared-budget pullback theorem fixes the selected-window and readback rows as the ones determined by U, V, T.

It remains to exclude side routes. The Real seal consumer exhaustion, completion boundary, and non-escape theorems remove quotient streams, selected limits, hidden selectors, second meet rows, host real equality, and ambient completeness from the carrier surface. Hence a Real-facing consumer can reach E only through the displayed pullback spine. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal pullback nonescape`

Theorem 885.21 (Cauchy modulus refinement source-meet exactness)**.** *For every accepted* CauchyModulusRefinem *packet $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$, the refinement witness row V is forced through the displayed source-meet spine*

$$M_0, M_1 \longrightarrow U \longrightarrow V.$$

No accepted read of V may replace U by a second meet row, bypass either source schedule, or import an external modulus comparison not displayed by R.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source meet exactness`

Proof. Start from the carrier definition. The packet has exactly two source schedules, one shared meet row, and one refinement witness row.

Use classifier transport to follow the displayed routes. The $\mathsf{Cont}(M_0, U)$ and $\mathsf{Cont}(M_1, U)$ entries bind both source schedules to U, and $\mathsf{Cont}(U, V)$ is the only route that reaches the refinement witness.

The non-escape theorem removes the remaining alternatives. A second meet row, hidden selector, external ordering, or host completeness witness is not a coordinate of R, so it cannot supply an accepted source for V. □

Theorem 885.22 (Cauchy modulus refinement window tolerance exactness)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, all selected-window and dyadic-tolerance reads are forced through the route*

$$V \longrightarrow T \longrightarrow W \longrightarrow Q.$$

The selector-budget row T*, finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance ledger* Q *are the only packet coordinates that may feed the selected-window tolerance surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier window tolerance exactness`

Proof. Project the selected-window portion of R. The carrier stores one selector-budget row T, one finite-window ledger W, and one tolerance-readback ledger Q.

Follow the displayed continuation rows. The route $\mathsf{Cont}(V, T)$ makes the selector budget downstream of the refinement witness, $\mathsf{Cont}(T, W)$ selects the finite windows, and $\mathsf{Cont}(W, Q)$ attaches the dyadic and regular-rational readbacks to those same windows.

Apply non-escape to exclude hidden routes. Since R contains no second window ledger, hidden selector, selected host limit, or ambient completeness coordinate, every tolerance read accepted by the packet is the T, W, Q read. □

Theorem 885.23 (Cauchy modulus refinement Real seal factorization)**.** *For every accepted* $\mathsf{CauchyModulusRefinem}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the displayed refinement and tolerance route*

$$U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

Consequently E *cannot be accepted from a quotient stream, host real equality, selected limit, hidden selector, second meet row, or ambient completeness witness outside the packet.*

Proof. First bind the source of the seal route. Source-meet exactness places the refinement witness V downstream of the shared meet row U and the two source schedules.

Next bind the tolerance surface. Window tolerance exactness forces the selected-window and dyadic-tolerance data through T, W, Q.

The route $\mathsf{Cont}(Q, E)$ is the only displayed continuation into the $\mathsf{Real}^{\uparrow}$ seal row. Together with the componentwise transports in H and the provenance and naming rows P, N, this makes E a factor of the displayed U, V, T, W, Q spine rather than an independent Real-facing object. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal factorization`

Theorem 885.24 (Cauchy modulus refinement common source lock)**.** *For every accepted* $\mathsf{CauchyModulusRefinemen}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any accepted refinement read keeps the source schedules M_0 *and* M_1 *locked to the same shared meet row* U *before the refinement witness row* V *is read. A consumer cannot read* V *from one source schedule alone, from a second meet row, or from an external modulus comparison not displayed by* R*.*

Proof. Start with source-meet exactness, Definition 885.21. It identifies the route to the refinement witness as

$$M_0, M_1 \longrightarrow U \longrightarrow V.$$

Use classifier transport to keep this route componentwise stable through H and C. The transports preserve both source schedules and the same shared meet row.

Finally apply non-escape, Definition 885.10. Since R carries no second meet row, hidden selector, external comparison, or ambient completeness coordinate, the common source lock is exactly the displayed M_0, M_1, U surface. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier common source lock`

Theorem 885.25 (Cauchy modulus refinement shared threshold transport)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the shared comparison and meet threshold is transported only through the displayed rows

$$M_0,\ M_1,\ U,\ V,\ T,\ H,\ C,\ P,\ N.$$

Thus no accepted threshold read may replace the two source schedules, the shared meet row, the refinement witness, the selector-budget row, the componentwise transports, the displayed continuation routes, the package provenance, or the local naming row by an external comparison, second meet, quotient stream, hidden selector, or host completeness witness.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared threshold transport`

Proof. Project the carrier of Definition 885.1. The packet displays exactly the two source schedules M_0, M_1, one shared meet row U, one refinement witness V, one selector-budget row T, one transport row H, one continuation row C, one provenance row P, and one local certificate row N.

Apply the source-meet-budget chain. It forces every accepted threshold read to move from M_0, M_1 through U, V, T before any finite selected window can be exposed.

Apply classifier transport. The componentwise hsame entries in H and the displayed Cont entries in C replay the same threshold through the named rows, while P and N authenticate the packet as the local $\mathsf{CauchyModulusRefinement}^{\uparrow}$ surface.

The carrier contains no external comparison, second meet, quotient stream, hidden selector, or host completeness coordinate. Hence the shared threshold can be transported only along the displayed rows. □

Theorem 885.26 (Cauchy modulus refinement selector budget totality)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the selector-budget row* T *is the only route from the refinement witness row* V *to the selected finite-window ledger* W *and the tolerance-readback ledger* Q*. Thus every selected-window read and every* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumed by the refinement factors through*

$$V \longrightarrow T \longrightarrow W \longrightarrow Q.$$

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget totality`

Proof. Read window-tolerance exactness, Definition 885.22. It already forces every selected-window and dyadic-tolerance read through V, T, W, Q.

The Real seal factorization theorem then shows why this budget route is total for downstream consumers: the seal row E is reachable only after U, V, T, W, Q have been displayed.

Apply non-escape to remove alternate selectors and window ledgers. The carrier has one selector-budget row, one selected-window ledger, and one tolerance-readback ledger, so T exhausts the route from V to W, Q. □

Theorem 885.27 (Cauchy modulus refinement selector budget non-escape)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the selector-budget row* T *is the only route from the source-meet threshold into the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *window-readback surface. In particular, no hidden selector, quotient stream, selected host limit, or countable-choice coordinate can supply an accepted window, rational-readback, or dyadic-tolerance read for* R*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget nonescape`

Proof. Start with window readback exhaustion. It identifies T, W, Q as the complete selected-window, regular-rational readback, and dyadic-tolerance surface of the packet.

Use source-meet exactness to bind the source threshold. The refinement witness V is reached only from the two source schedules through the shared meet row U, and the displayed continuation route then moves from V to T.

Apply non-escape to the remaining alternatives. Since the carrier has no hidden selector, quotient stream, selected host limit, or countable-choice coordinate, the route through T is the sole accepted handoff from the source-meet threshold into W and Q. □

Theorem 885.28 (Cauchy modulus refinement Real seal obligation surface)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the rows $U, V, T, W, Q, E, H, C, P, N$ *form the complete* $\mathsf{Real}^{\uparrow}$ *seal obligation surface for the refinement route. The surface contains the shared meet row, refinement witness, selector-budget row, finite windows, tolerance and readback ledger, Real seal, componentwise transports, displayed continuation routes, provenance, and local naming row, and it excludes quotient streams, host real equality, hidden selectors, selected limits, second meet rows, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal obligation surface`

Proof. First bind the source of the surface. Common source lock fixes M_0 and M_1 to the shared meet row U before the witness V is read.

Next bind the selected-window route. Selector-budget totality forces the only route from V to the selected windows and tolerance-readback ledger through T, W, Q.

Now apply Real seal factorization. It places E downstream of U, V, T, W, Q and keeps the seal route inside the displayed transport, continuation, provenance, and naming rows.

Finally use non-escape. Quotient streams, host real equality, hidden selectors, selected limits, second meet rows, and ambient completeness are absent from the carrier, so the listed rows are the whole Real-seal obligation surface. □

Theorem 885.29 (Cauchy modulus refinement Real seal consumer exhaustion)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, every* Real$^{\uparrow}$ *or completion-facing consumer reaches the seal row* E *only after consuming the shared threshold, selector-budget, finite-window, rational-readback, dyadic-tolerance, transport, continuation, package, and local* NameCert *rows displayed by*

$$M_0, M_1, U, V, T, W, Q, H, C, P, N.$$

No accepted consumer may read E *directly from a quotient stream, selected limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or unbudgeted completion object.*

Proof. First apply Real seal factorization. The seal row E factors through the displayed route U, V, T, W, Q, E, so a Real-facing consumer reaches the seal only after the shared meet, refinement witness, selector budget, finite windows, and tolerance-readback ledger have been consumed.

Next apply Real seal refusal. A purported seal read that omits the source schedules, meet, witness, selector budget, windows, tolerances, readbacks, transports, routes, provenance, and local name is not an accepted carrier read.

Use window tolerance exactness to identify the finite-window, rational-readback, and dyadic-tolerance coordinates as T, W, Q. The same route attaches the selected StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, and DyadicRatCore$^{\uparrow}$ tolerance rows to the seal.

Finally apply non-escape. Quotient streams, selected limits, host real equality, hidden selectors, second meet rows, ambient completeness coordinates, and unbudgeted completion objects are absent from the carrier, so every accepted Real or completion-facing consumer must pass through the displayed rows before reading E. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal consumer exhaustion`

Theorem 885.30 (Cauchy modulus refinement obligation closure package)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the carrier of Definition 885.1, the NameCert obligation rows of Definition 885.2, the source-meet-budget chain of Definition 885.11, window readback exhaustion of Definition 885.12, Real seal refusal of Definition 885.13, classifier transport of Definition 885.16, source-meet exactness of Definition 885.21, window tolerance exactness of Definition 885.22, Real seal factorization of Definition 885.23, common source lock of Definition 885.24, shared threshold transport of Definition 885.25, selector budget totality of Definition 885.26, selector budget non-escape of Definition 885.27, Real seal obligation surface of Definition 885.28, Real seal consumer exhaustion of Definition 885.29, and non-escape of Definition 885.10 jointly form the* obligation *surface for* CauchyModulusRefinement$^{\uparrow}$*.*

The package exports exactly the two CauchyModulus$^{\uparrow}$ *source schedules, shared* CauchyModulusMeet$^{\uparrow}$ *row,* CauchyModulusExtraction$^{\uparrow}$ *witness row,* RegularCauchySelectorBudget$^{\uparrow}$ *row, finite* StreamName$^{\uparrow}$ *window ledger,* DyadicRatCore$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *tolerance ledger,* Real$^{\uparrow}$ *seal row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Project the displayed packet R. The carrier and NameCert obligation theorem supply the finite row inventory and the five required local certificate clauses.

Read the classifier and source rows. Classifier transport keeps every accepted read on the componentwise hsame and displayed Cont routes, while source-meet exactness fixes V as the witness reached from M_0, M_1 through U.

Read the selected-window and Real-facing rows. Window tolerance exactness binds the tolerance surface to T, W, Q, and Real seal factorization places E after the shared meet, refinement witness, selector budget, windows, and tolerance ledger.

Use the intermediate obligations to expose the selector-budget route. Common source lock fixes M_0, M_1, U, shared threshold transport keeps the comparison and meet threshold on $M_0, M_1, U, V, T, H, C, P, N$, selector budget totality and selector budget non-escape make T the sole handoff into W, Q, and the two Real-seal obligations force every Real or completion-facing consumer through the displayed rows before E is read.

Finally combine these rows with non-escape. All excluded objects are absent from the carrier, so the assembled package is precisely the written obligation surface and no external Real-completeness or host-equality route remains. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier obligation closure package`

Theorem 885.31 (Cauchy modulus refinement primitive scope)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the exact primitive scope of the packet is the displayed BHist *carrier row* R*, the componentwise* hsame *transports stored in* H*, the displayed* Cont *route row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row* N*, together with the named sibling rows*

$\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySe}$

No primitive outside these rows is part of the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *surface.*

Proof. Project the carrier and NameCert obligations. The displayed packet supplies the BHist row, the componentwise hsame row, the Cont row, the Pkg row, and the local naming row, while Definition 885.2 lists the finite certificate clauses attached to those rows.

Bind the source and selector side. Source-meet exactness fixes the refinement witness to the route M_0, M_1, U, V, and selector-budget totality fixes every selected-window handoff to V, T, W, Q.

Bind the Real-facing side. Window-tolerance exactness identifies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows as the selected tolerance surface, and Real seal factorization places the $\mathsf{Real}^{\uparrow}$ seal after the same route.

Finally apply Real seal consumer exhaustion and non-escape. Since the carrier has no quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness coordinate, the displayed rows and named siblings are the whole primitive scope. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier primitive scope`

Theorem 885.32 (Cauchy modulus refinement sibling dependency lattice)**.** *Every accepted* $\mathsf{CauchyModulusRefinem}$ *packet consumes its sibling surfaces in the dependency order*

$\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, Str

The comparison rows bind the two source schedules, the meet row supplies the shared source threshold, the extraction row supplies the refinement witness, the selector-budget row selects the finite windows, the $\mathsf{StreamName}^{\uparrow}$ *row records those windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows attach tolerance and rational readback data, and the* $\mathsf{Real}^{\uparrow}$ *row seals the downstream Real surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier sibling dependency lattice`

Proof. Start with the source-meet chain. Source-meet exactness and common source lock place the comparison and meet rows before the extraction witness V can be read.

Next expose the selected-window route. Selector-budget totality makes T the sole handoff from V into W, Q, and window-tolerance exactness identifies W as the $\mathsf{StreamName}^{\uparrow}$ window ledger and Q as the $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ tolerance-readback ledger.

Now read the Real seal. Real seal factorization places E after U, V, T, W, Q, and Real seal consumer exhaustion forces every Real or completion-facing consumer through the same sibling order before the seal can be consumed.

Non-escape removes all alternate lattices. The packet carries no second meet row, hidden selector, quotient stream, selected limit, host equality, or ambient completeness coordinate, so the dependency order written above is the accepted sibling lattice. □

Theorem 885.33 (Cauchy modulus refinement L10 sibling route)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any Phase Real and Completion consumer of the refinement reads the L10 route through

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

in that order after the $\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, *and* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *rows have been consumed. Such a consumer receives no quotient stream, selected limit, host real equality, hidden selector, or ambient completeness datum from the accepted packet.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 sibling route`

Proof. Start with the sibling dependency lattice of Definition 885.32. It places comparison, meet, extraction, and selector budget before the four downstream rows and orders those rows as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The finite-window handoff is the route supplied by Definition 885.22. It identifies the selected-window and tolerance-readback segment as $T \longrightarrow W \longrightarrow Q$, so the consumer cannot reach the dyadic or regular-rational readback surface before passing through the selector-budget and window rows.

Now apply Real seal factorization, Definition 885.23. The seal row E sits after Q, so the $\mathsf{Real}^{\uparrow}$ read is downstream of the StreamName, DyadicRatCore, and RegSeqRat route rather than an independent completion input.

Finally use non-escape, Definition 885.10. Quotient streams, selected limits, host real equality, hidden selectors, and ambient completeness data are absent from the carrier, so the L10 route above is the reusable handoff available to Phase Real and Completion consumers. □

Theorem 885.34 (Cauchy modulus refinement completion boundary)**.** *No* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet may read quotient streams, selected limits, host real equality, hidden selectors, second meet rows, or ambient completeness outside the displayed packet. Such a consumer must factor through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

and through the sibling dependency lattice of Definition 885.32.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier completion boundary`

Proof. First bind the route to the Real seal. Real seal factorization places E downstream of the shared meet, refinement witness, selector budget, selected windows, and tolerance-readback ledger.

Use Real seal consumer exhaustion to control downstream reads. A Real or completion-facing consumer reaches E only after consuming the displayed source, meet, witness, budget, window, tolerance, transport, route, provenance, and naming rows.

Use the primitive-scope theorem and sibling dependency lattice to identify the whole surface available to that consumer. They restrict the primitive rows and sibling rows to the displayed BHist, hsame, Cont, Pkg, NameCert, comparison, meet, extraction, selector-budget, StreamName, DyadicRatCore, RegSeqRat, and Real surfaces.

Finally apply non-escape. The forbidden quotient, selected-limit, host-equality, hidden-selector, second-meet, and ambient-completeness objects are absent from the packet, so no completion-facing read may obtain them outside the displayed route. □

Theorem 885.35 (Cauchy modulus refinement BHist source scope)**.** *For every accepted* CauchyModulusRefinement *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ *are finite* BHist*-facing rows over* CauchyModulus$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert*. No quotient stream, host-completeness object, ambient completeness coordinate, selected limit, hidden selector, or host real equality coordinate belongs to the accepted source scope.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier bhist source scope`

Proof. Project the displayed packet through the carrier. The carrier row gives the finite BHist spine $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, and the NameCert obligations attach the local naming clause to exactly that spine.

Bind the primitive side. Primitive scope identifies the componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row, while the sibling dependency lattice identifies the named CauchyModulus$^{\uparrow}$, meet, extraction, selector-budget, StreamName, dyadic, rational, and Real sibling rows.

Use the completion boundary and non-escape to remove external sources. Since quotient streams, selected limits, host real equality, hidden selectors, second meet rows, host-completeness objects, and ambient completeness coordinates are absent from the carrier, no accepted source row can be read outside the displayed BHist-facing surface. □

Theorem 885.36 (Cauchy modulus refinement continuation route scope)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the displayed continuation order*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E$$

is the only consumer route for selector-window and Real$^{\uparrow}$*-seal reads. A consumer of selected windows, rational readbacks, dyadic tolerances, or the Real seal must therefore pass through the source schedules, shared meet, extraction witness, selector-budget row, finite-window ledger, tolerance-readback ledger, and seal row in this order.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier cont route scope`

Proof. Start with the source route. Source-meet exactness fixes the witness row V as downstream of M_0, M_1 through U, so a consumer cannot reach the refinement witness without first reading the two source schedules and the shared meet.

Next read the selected-window surface. Window readback exhaustion and selector budget totality place the selector handoff at T and the finite-window and tolerance-readback rows at W, Q.

Finally read the seal. Real seal factorization places E after U, V, T, W, Q, and Real seal consumer exhaustion prevents a downstream consumer from reaching the seal by a route that omits those rows. Non-escape removes quotient-stream, selected-limit, host-equality, hidden-selector, second-meet, and ambient-completeness alternatives, so the displayed order is the only accepted consumer route. □

Theorem 885.37 (Cauchy modulus refinement NameCert scope)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *scope is exhausted by carrier admission, classifier transport, source-meet exactness, selector-budget and window-tolerance exactness, and* $\mathsf{Real}^{\uparrow}$*-seal non-escape. Equivalently, the certificate can name only the admitted finite* BHist *rows, their componentwise* hsame *transports, the displayed* Cont *routes, the* Pkg *provenance row, and the Real-facing seal route certified by the obligation closure package.*

Proof. First admit the named object. The carrier and NameCert obligation theorem bind the local certificate to the finite packet rows and require the naming row to travel with the displayed BHist, hsame, Cont, and Pkg fields.

Next transport the classifier through the source and window rows. Classifier transport keeps accepted reads on the displayed transports and routes, source-meet exactness fixes the M_0, M_1, U, V source-meet witness, and window tolerance exactness fixes the T, W, Q selector, window, dyadic-tolerance, and rational-readback surface.

Finally seal the Real-facing edge. The BHist source scope supplies the permitted primitive rows, the continuation route scope supplies the only consumer order into E, and the obligation closure package assembles those rows with Real-seal factorization and non-escape. Thus the local NameCert scope has no remaining quotient stream, hidden selector, selected limit, host equality, ambient completeness, or external completion coordinate to name. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier namecert scope`

The closest siblings are $\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, and $\mathsf{RegularCauchyLimitTransport}^{\uparrow}$. The packet is composite because it consumes their rows as coordinates rather than presenting an atomic independent carrier. A later Lean TasteGate module should still provide $\mathsf{Nontrivial}$ and $\mathsf{FieldFaithful}$ instances over the packet fields, since the finite carrier separates the comparison, meet, witness, budget, window, ledger, seal, transport, route, provenance, and local certificate coordinates.

Theorem 885.38 (Cauchy modulus refinement budget-selector source lock)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

every accepted refinement route reads the source schedules, shared meet row, extraction witness, and selector budget

$$M_0, M_1, U, V, T$$

before it reads the selected window ledger W, tolerance-readback ledger Q, or Real seal row E. Thus the budget selector is source-locked to the displayed CauchyModulusMeet$^\uparrow$, CauchyModulusExtraction$^\uparrow$, *and* RegularCauchySelectorBudget$^\uparrow$ *rows before any downstream* StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, *or* Real$^\uparrow$ *consumer can be served.*

Proof. Project the carrier of Definition 885.1. The displayed tuple places M_0, M_1, U, V, T before the window, tolerance, readback, and seal rows, and no second source, second meet, hidden selector, or host-completeness coordinate appears in the packet.

Apply source-meet exactness from Definition 885.21. It forces the witness row V to be read only through M_0, M_1, U, so a selected-window route cannot choose its source after the fact.

Next apply selector-budget totality from Definition 885.26. The selector budget row T is the sole route from V into the selected finite-window and tolerance surface.

Finally use window-tolerance exactness from Definition 885.22. It identifies T, W, Q as the exact selected-window and dyadic-tolerance route, so every later read of W, Q, E is already locked to M_0, M_1, U, V, T. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementBudgetSelectorSourceLock`

Theorem 885.39 (Cauchy modulus refinement diagonal-consumer boundary)**.** *Any* Real$^\uparrow$ *consumer or diagonal-completion consumer of an accepted* CauchyModulusRefinement$^\uparrow$ *packet factors through the displayed* BHist, hsame, Cont, Pkg, NameCert, CauchyModulusMeet$^\uparrow$, CauchyModulusExtraction$^\uparrow$, RegularCauchySelectorBudget$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, *and* Real$^\uparrow$ *rows of*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No such consumer may read a quotient stream, selected limit, host real equality, hidden selector, second meet row, side channel completion object, or ambient completeness coordinate outside that displayed route.

Proof. Start with the completion boundary of Definition 885.34. It already requires every Real or completion-facing consumer to factor through the displayed packet and the sibling dependency lattice before reading the seal.

Use the BHist source scope and continuation route scope, Definition 885.35 and Definition 885.36. These theorems identify the primitive rows as the finite BHist packet, componentwise hsame, displayed Cont, Pkg, NameCert, and the ordered consumer route from the source schedules to E.

Then apply the budget-selector source lock. It places M_0, M_1, U, V, T before the selected-window and tolerance rows, so a diagonal consumer reaches W, Q, E only through the same source-locked route.

Finally apply non-escape from Definition 885.10. Since all forbidden quotient, selected-limit, host-equality, hidden-selector, second-meet, and ambient-completeness rows are absent from the carrier, the displayed rows exhaust the diagonal-consumer boundary. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier diagonal consumer boundary`

Theorem 885.40 (Cauchy modulus refinement root-unblock package)**.** *The root-unblock obligation package for downstream Real-completion consumers of* CauchyModulusRefinement$^\uparrow$ *is exactly the conjunction of the local* NameCert *scope, the budget-selector source lock, and the diagonal-consumer boundary. An accepted downstream route may cite* CauchyModulusRefinement$^\uparrow$ *only through the rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

and the sibling CauchyModulusMeet$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *surfaces displayed by those rows.*

Proof. First use Definition 885.37. It binds the local naming certificate to carrier admission, classifier transport, source-meet exactness, selector-budget and window-tolerance exactness, and Real-seal non-escape.

Next use Definition 885.38. This supplies the source-lock portion of the package by placing M_0, M_1, U, V, T before W, Q, E.

Finally use Definition 885.39. It supplies the downstream Real and diagonal-completion boundary, excluding quotient streams, selected limits, host real equality, hidden selectors, side channel completion objects, second meet rows, and ambient completeness.

The three clauses use the same displayed carrier and the same sibling rows. Therefore they assemble one root-unblock package rather than three independent routes, and every downstream consumer must cite that package through the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root unblock package`

Theorem 885.41 (Cauchy modulus refinement diagonal budget cofinality)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

every diagonal Real-completion consumer that has passed through the shared-meet and selector-budget rows factors through a cofinal finite budget route before reaching the Real$^{\uparrow}$ *seal. The route uses the* CauchyModulus$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchySharedTailModulus$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows displayed by the packet, and it exports no choice-selected tail, quotient stream, or ambient completeness theorem.*

Proof. Begin with source-meet exactness from Definition 885.21. It binds the two source modulus rows and the shared meet row to the refinement witness before any selector or Real-facing consumer can be read.

Apply selector budget totality from Definition 885.26. The selector row T supplies the finite budget route into the selected-window and tolerance-readback rows, so a diagonal consumer receives a cofinal finite budget before it can ask for the Real seal.

Next use Real seal factorization and the diagonal-consumer boundary, Definition 885.23 and Definition 885.39. These results force every Real-completion consumer through $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ and through the displayed BHist, hsame, Cont, Pkg, NameCert, CauchyModulus$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchySharedTailModulus$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ surfaces.

Finally use the root-unblock package of Definition 885.40. The package assembles the local NameCert scope, budget-selector source lock, and diagonal boundary into one route, so no choice-selected tail, quotient stream, or ambient completeness theorem is exported by the accepted packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier diagonal budget cofinality terminal route`

Theorem 885.42 (Cauchy modulus refinement common-refinement selector lock)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any Real-completion consumer preserving the common meet row U, extraction row V, selector-budget row T, finite-window row W, dyadic ledger Q, and Real seal row E must read the same finite common-refinement route before it reaches $\mathsf{Real}^{\uparrow}$*. The packet makes no alternate selected tail, quotient stream, second meet row, or ambient completeness coordinate available to that consumer.*

Proof. Begin with source-meet exactness from Definition 885.21. It identifies the common meet row U as the unique displayed meet through which the two source moduli M_0, M_1 can reach the extraction row V.

Next apply selector-budget totality from Definition 885.26. The row T routes the extraction row into the finite-window and tolerance surface W, Q, so a Real-completion consumer preserving U, V, T, W, Q preserves the same finite selector budget.

Use Real-seal factorization from Definition 885.23 and the diagonal-budget cofinality route of Definition 885.41. These place the Real seal row E after the same cofinal finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally repack the route through the structural rows of R. The BHist carrier, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert row name only the preserved rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. Since no other row is exported, there is no alternate selected tail, quotient stream, second meet row, or ambient completeness coordinate to read. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier common refinement selector lock`

Theorem 885.43 (Cauchy modulus refinement RealObservationBudget handoff)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any $\mathsf{RealObservationBudget}^{\uparrow}$ *request that uses the same selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger, and* $\mathsf{Real}^{\uparrow}$ *seal row factors through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

before any Real-facing observation is consumed. The handoff exports no quotient stream, selected limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or unbudgeted observation selector.

Proof. First apply the budget-selector source lock of Definition 885.38. It places the source schedules, shared meet row, extraction witness, and selector budget in the prefix M_0, M_1, U, V, T before the selected finite-window, tolerance-readback, or Real-seal rows can be read.

Next use window readback exhaustion and diagonal budget cofinality, Definition 885.12 and Definition 885.41. These bind the selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic tolerance rows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger to T, W, Q, E, and they route every diagonal Real-completion consumer through the displayed cofinal finite budget before the $\mathsf{Real}^{\uparrow}$ seal is observed.

Finally apply the completion boundary and non-escape, Definition 885.34 and Definition 885.10. The completion boundary keeps the observation request on the displayed packet and sibling route, while non-escape removes quotient streams, selected limits, host real equality, hidden selectors, second meet rows, ambient completeness coordinates, and unbudgeted observation selectors from the exported surface. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real observation budget handoff`

Theorem 885.44 (Cauchy modulus refinement RealLimitUniqueness handoff)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$*-facing consumer preserving the same selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal row must first factor through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

and then through the two-seal packet rows of chapter 1039. The handoff exports no selected host limit, quotient stream equality, hidden selector, second meet, or ambient completeness row.

Proof. Start with the RealObservationBudget handoff of Definition 885.43. It binds the selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger, and $\mathsf{Real}^{\uparrow}$ seal row to the displayed refinement packet before any Real-facing observation is consumed.

Apply the common-refinement selector lock of Definition 885.42. The consumer preserving those rows must therefore keep the same source moduli, common meet, extraction row, selector budget, finite-window route, tolerance ledger, and Real seal row in the order $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$.

Finally use the RealLimitUniquenessSeal public route theorem Definition 1039.27. Its two-seal surface exposes only the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows, shared comparison row, finite window, dyadic ledger, $\mathsf{Real}^{\uparrow}$ classifier row, uniqueness handoff, transports, replay, provenance, and local name. Since neither packet exports a selected host limit, quotient stream equality, hidden selector, second meet, or ambient completeness row, the consumer factors through the displayed refinement route and then through the two-seal packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real limit uniqueness handoff`

Definition 885.45 (Cauchy modulus refinement L10 face-status pullback record)**.** *For an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the L10 face-status pullback record is the ordered four-face product of the local closurestatus, formalstatus, leantarget, and bridgestatus rows read from $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *along the route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

It contains no aggregate formalstatus, aggregate bridge row, selected limit, quotient stream, countable-choice row, or ambient completion coordinate.

Lean4 (definition) `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementL10FaceStatusPullbackReco`

Theorem 885.46 (Cauchy modulus refinement four-face status objectwise pullback)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet routed to a Real- or completion-facing consumer through the L10 face-status pullback record of Definition 885.45 along the displayed handoff rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

the consumer pulls back only the four local status records of $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*. The refinement route contributes no aggregate formal-status coordinate, no aggregate bridge row, no selected limit, no quotient stream, and no ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier four face status objectwise pullback`

Proof. First use the RealObservationBudget handoff of Definition 885.43. It routes every accepted Real-facing observation request through the selected $\mathsf{StreamName}^{\uparrow}$ windows, dyadic tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback ledger, and $\mathsf{Real}^{\uparrow}$ seal row of the displayed packet.

Next apply the RealLimitUniqueness handoff of Definition 885.44. The same route is preserved when the consumer continues to the two-seal uniqueness surface, and the packet still exports no selected host limit, quotient stream equality, hidden selector, second meet, or ambient completeness row.

Now apply Definition 38.471. The Real and Completion audit read is therefore the product of the local $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ face records, not a status value supplied by the refinement route.

Finally use Definition 38.833. It keeps formal-status, theory-closure, and bridge-status data attached to the four displayed faces, so the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet cannot fuse them into an aggregate row or reintroduce the excluded completion coordinates. □

Corollary 885.47 (Cauchy modulus refinement diagonal-budget cofinality determinacy)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

fix the local shared-meet row, selector-budget row, selected finite $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal row displayed by* R*. Every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$*-facing diagonal-budget consumer use that reports a route selection, normal-form readback, or exactness decision must report the one named by the chapter: the finite route through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

and then through the two-seal packet rows of chapter 1039. No alternate hidden coordinate, duplicate classifier source, selected host limit, quotient stream equality, hidden selector, second meet, or ambient completeness row can supply a different accepted route, normal form, or exactness decision.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier diagonal budget cofinality determinacy`

Proof. Start with diagonal budget cofinality, Definition 885.41. Once the shared-meet and selector-budget rows have been read, every diagonal Real-completion consumer is already on the cofinal finite budget route through $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ and through the displayed $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert surfaces. That route exports no choice-selected tail, quotient stream, or ambient completeness theorem.

Now apply the RealLimitUniqueness handoff, Definition 885.44. A consumer that preserves the same selected finite windows, dyadic tolerance ledger, regular readback, and Real seal must first factor through the displayed refinement packet and only then through the two-seal uniqueness packet. The handoff leaves no selected host limit, quotient stream equality, hidden selector, second meet, or ambient completeness row for the consumer to read.

Fix the displayed local rows. A reported route selection is accepted only after the cofinal finite route has supplied the meet, extraction witness, selector budget, selected windows, tolerance

ledger, readback, Real seal, transport, replay, provenance, and local name. A reported normal-form readback is the same ordered read, because the RealLimitUniqueness-facing continuation consumes the fixed windows, ledger, readback, and seal before it reaches the two-seal packet. A reported exactness decision is therefore the decision that the consumer use is accepted exactly when it factors through those fixed rows and the downstream two-seal rows.

It remains to rule out duplicate sources for that decision. Any proposed duplicate source is either one of the displayed rows, in which case it is already identified by the fixed cofinal route and the handoff, or it is an unlisted coordinate. The cofinality theorem rejects unlisted diagonal-budget coordinates, and the handoff rejects unlisted uniqueness-facing coordinates. Hence every accepted consumer-facing use chooses the same finite route, normal form, and exactness decision named by the chapter. □

Theorem 885.48 (Cauchy modulus refinement common-window selector extraction)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet whose source schedules* M_0, M_1*, shared meet* U*, refinement witness* V*, and selector-budget row* T *are the displayed rows of the carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or diagonal-completion consumer that reaches the selected-window rows must extract the same finite-window and tolerance route* W, Q *through* T*. No hidden selector, quotient stream, selected limit, host real equality, second meet row, or ambient completeness coordinate may supply a competing selected-window route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCommonWindowSelectorExtraction`

Proof. First lock the common source. Apply common source lock and shared threshold transport, Definition 885.24 and Definition 885.25, to force every accepted read of U and V through the same M_0, M_1 schedules.

Next extract the selected-window route. Selector-budget totality, Definition 885.26, makes T the sole admitted route from V to the selected finite-window surface, and window-tolerance exactness, Definition 885.22, identifies that surface with the displayed W, Q rows.

Finally pass to downstream consumers. The diagonal-consumer boundary, Definition 885.39, says that Real-facing and diagonal-completion reads factor through the displayed sibling rows and cannot import quotient streams, selected limits, host real equality, hidden selectors, second meet rows, side channel completion objects, or ambient completeness. Hence the consumer can extract W, Q only by the common route through T. □

Theorem 885.49 (Cauchy modulus refinement Real-completeness handoff)**.** *Every downstream Real-completeness consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *refinement packet must read the shared meet row, extraction witness, selector budget, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal in the displayed order*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

before using the completion boundary. The consumer may cite the packet only through the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$*,* $\mathsf{CauchyModulusExtraction}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by that finite route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRealCompletenessHandoff`
`semantic name certificate`

Proof. Begin with the root-unblock package of Definition 885.40. It packages the local NameCert scope, budget-selector source lock, and diagonal-consumer boundary as one accepted route for downstream Real-completion consumers.

Use the diagonal-consumer boundary from Definition 885.39. It forces any Real or diagonal-completion read through the displayed sibling rows before the seal row can be consumed.

Next apply Real-seal factorization and window-tolerance exactness, Definition 885.23 and Definition 885.22. These identify the selected windows, dyadic tolerance rows, regular readbacks, and Real seal as the same ordered route after M_0, M_1, U, V, T.

The completion boundary is therefore reached only after the displayed meet, extraction, selector, window, tolerance, readback, and seal rows have been read. Since the packet has no quotient stream, selected host limit, host real equality, hidden selector, side channel completion object, second meet row, or ambient completeness coordinate, the stated handoff exhausts the Real-completeness consumer surface. □

Theorem 885.50 (Cauchy modulus refinement root window-selector totality)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E,$ be an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet. Every downstream read of a selected finite window or dyadic tolerance row factors through the root route*

$$M_0, M_1, U, V, T$$

before it reaches W or Q. Thus the selector-budget row T is the total root selector for the displayed StreamName$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *window surface used by downstream* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumers.*

Proof. First read the root-unblock package of Definition 885.40. It permits downstream Real-completion consumers to cite the refinement packet only through the displayed finite route.

Next apply selector-budget totality from Definition 885.26. This makes T the admitted selector from the extraction witness V to the selected finite-window and dyadic-tolerance surface.

Finally use common-window selector extraction from Definition 885.48. It identifies every downstream selected-window read with the same W, Q route through T, so no consumer can choose a separate selector after reading M_0, M_1, U, V. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRootWindowSelectorTotality`

Theorem 885.51 (Cauchy modulus refinement root readback stability)**.** *For an accepted* CauchyModulusRefinemen *packet $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$, the selected-window and tolerance readbacks W, Q are stable under the componentwise* hsame *row H and the displayed route row C. Any readback consumed by* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *must use the displayed rows W, Q, H, C, P, N, with no hidden selector, quotient stream, selected host limit, or ambient completion coordinate.*

Proof. Begin with classifier transport from Definition 885.16. It says componentwise hsame and the displayed Cont route carry accepted refinement packets without adding a new coordinate.

Use window-readback exhaustion from Definition 885.12. The selected-window and tolerance readbacks are precisely the displayed W, Q rows attached to the packet.

Now combine that exhaustion with root window-selector totality. Since every selected-window read reaches W, Q only through M_0, M_1, U, V, T, transport along H and C preserves the same root readback route.

The non-escape boundary of Definition 885.10 rules out hidden selector, quotient stream, selected host limit, host real equality, side channel completion object, second meet row, and ambient completeness coordinates. Therefore the displayed sibling rows exhaust the readback surface. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement root readback stability`

Theorem 885.52 (Cauchy modulus refinement root Real seal exhaustion)**.** *Every Real-completion consumer of an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet reaches the Real seal row E only through the displayed root route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

In particular, the consumer cannot bypass the selector, selected-window readback, componentwise transport, route, package, or local naming rows by appealing to a quotient stream, selected host limit, hidden selector, side channel completion object, second meet row, host real equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root real seal exhaustion`

Proof. Start with the Real-completeness handoff of Definition 885.49. It places the meet, extraction, selector, finite-window, tolerance, readback, and seal rows before any completion-facing use.

Apply the diagonal-consumer boundary from Definition 885.39. This forces Real-completion and diagonal-completion consumers to factor through the displayed sibling dependency lattice and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Finally combine root readback stability with non-escape, Definition 885.51 and Definition 885.10. The W, Q readbacks and the seal row E stay on the same displayed route under H and C, and no forbidden completion coordinate remains available. Hence $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ exhaust the root Real-seal surface. □

Theorem 885.53 (Cauchy modulus refinement root-threshold scope)**.** *Every accepted* CauchyModulusRefinement$^{\uparrow}$ *refinement packet routes its root threshold scope through the displayed finite rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No Real- or Completion-facing consumer may replace that root route by a second meet, hidden selector, selected host limit, quotient stream, host real equality, side-channel completion row, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root threshold scope`

Proof. Read the carrier definition and the root-unblock package of Definition 885.40. Together they identify the only root threshold inventory as the source schedules, meet, extraction, selector-budget, selected-window, tolerance, seal, transport, route, provenance, and local naming rows.

Apply common-window selector extraction from Definition 885.48. It forces W, Q to be read from the same selector-budget route T after M_0, M_1, U, V.

Finally use the Real-completeness handoff of Definition 885.49. The downstream seal read arrives only after the displayed root route, so the listed excluded coordinates have no packet row from which to enter. □

Theorem 885.54 (Cauchy modulus refinement shared-tail handoff)**.** *Every shared-tail handoff exported by an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet factors through the existing* CauchySharedTailModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *surfaces. The handoff reads the same selector-budget, window, tolerance, readback, and seal route T, W, Q, E already displayed by the packet, together with H, C, P, N.*

Proof. Begin with the root-unblock package. It allows a downstream Real-completion consumer to cite the refinement packet only through the displayed finite rows and their named sibling surfaces.

Use common-window selector extraction to reach W, Q through T. The shared-tail side can therefore consume only the finite window and tolerance rows supplied by the same route.

The Real-completeness handoff then carries those rows into the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal surface, with $\mathsf{CauchySharedTailModulus}^{\uparrow}$ serving as the shared-tail threshold sibling. No additional tail witness or host completion row is exported. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementSharedTailHandoff`

Theorem 885.55 (Cauchy modulus refinement shared-tail selector-window determinacy)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose two accepted refinement routes expose the same shared-tail modulus sibling, the same selector-budget row T, the same selected* $\mathsf{StreamName}^{\uparrow}$ *window W, the same dyadic tolerance row Q, and the same* $\mathsf{Real}^{\uparrow}$ *seal E. Then they determine the same Real-completion consumer surface through the displayed* BHist *route, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row N.*

No quotient stream, hidden selector, selected host limit, second meet row, or ambient completeness coordinate separates the two routes once the shared-tail sibling and the T, W, Q, E rows agree.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementSharedTailSelectorWindowDetermina`

Proof. First force the selector window. Root window-selector totality, Definition 885.50, makes both routes pass through M_0, M_1, U, V, T before either route can expose the selected $\mathsf{StreamName}^{\uparrow}$ window W and dyadic tolerance row Q.

Next apply root readback stability, Definition 885.51. Agreement on the selected window and tolerance makes the readback travel through the same displayed H and C rows rather than through a detached stream representative.

Use shared-tail handoff, Definition 885.54. The common shared-tail modulus sibling carries the same selector-budget, window, tolerance, RegSeqRat readback, and Real seal route into T, W, Q, E, H, C, P, N.

Finally combine Real-consumer coverage, Definition 885.56, with selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. With T, W, Q, E fixed, the Real-completion consumer surface is fixed through H, C, P, N, and the excluded quotient, selector, host-limit, meet, and completeness coordinates have no row in the accepted packet. □

Theorem 885.56 (Cauchy modulus refinement Real-consumer coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet reads only the displayed selector-budget, window, readback, and seal route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Thus the Real-completion consumer surface is exhausted by the carrier route, the common-window selector extraction, and the Real-completeness handoff.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real consumer coverage`

Proof. Project the accepted carrier. The consumer-visible rows are exactly the displayed finite tuple and the structural rows H, C, P, N that transport, replay, package, and name it.

Apply Definition 885.48. This removes any competing selected-window path by requiring W, Q to be obtained through the same T route.

Finally apply Definition 885.49. It orders the meet, extraction, selector, finite-window, tolerance, readback, and seal rows before any Real-completion use. Therefore every consumer read is covered by the displayed route and no hidden selector, quotient stream, selected limit, host equality, or ambient completeness coordinate remains. □

Theorem 885.57 (Cauchy modulus refinement root terminal selector readback)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose two accepted terminal reads of R face either* $\mathsf{Real}^{\uparrow}$ *or Completion consumers. The two reads determine the same completed selector-readback consumer surface exactly when they agree on the displayed selector-budget row T, the selected* $\mathsf{StreamName}^{\uparrow}$ *window W, the dyadic tolerance and readback row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal E, the componentwise* hsame *transport row H, the displayed* Cont *route C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row N.*

The determinacy is internal to the displayed sibling surfaces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Quotient streams, selected host limits, host real equality, second meet rows, hidden selectors, and ambient completeness coordinates are not coordinates of the accepted terminal packet and cannot distinguish the two reads.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRootTerminalSelectorReadback`

Proof. First force the terminal reads onto the common selector spine. Common-window selector extraction, Definition 885.48, and root window-selector totality, Definition 885.50, make each accepted terminal read pass through the same selector-budget row T before it can expose W or Q.

Next identify the readback rows. Root readback stability, Definition 885.51, carries agreement on the selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row to the same $\mathsf{RegSeqRat}^{\uparrow}$ readback through the componentwise H transport and displayed C route.

Then seal the completion-facing surface. Root Real seal exhaustion, Definition 885.52, places the $\mathsf{Real}^{\uparrow}$ seal E after the displayed readback rows, and shared-tail selector-window determinacy, Definition 885.55, identifies the selector-window surface once T, W, Q, E, H, C, P, N agree.

For the forward direction, equal completed selector-readback surfaces expose the same terminal rows by Real-consumer coverage, Definition 885.56. Hence the two reads agree on T, W, Q, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, E, H, C, P, N, and the local naming row.

For the reverse direction, agreement on those displayed rows reconstructs the same accepted route across $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The same cited coverage and exhaustion results exclude quotient streams, selected host limits, host real equality, second meet rows, hidden selectors, and ambient completeness coordinates, so no off-route datum remains from which a second terminal consumer surface can be read. □

Theorem 885.58 (Cauchy modulus refinement and diagonal L10 consumer spine comparison)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet, and let K be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet. Suppose the two packets expose the same* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the same* $\mathsf{Real}^{\uparrow}$ *seal, the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary, and the same* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *support rows. Then every Phase Real and Completion consumer reads the same displayed L10 selector spine from R and from K.*

The comparison is internal to the shared StreamName, DyadicRatCore, RegSeqRat, Real, CauchyLimitSeal, and structural support rows. Quotient streams, selected host limits, host real equality, countable choice, ambient completeness, alternate Cauchy witnesses, and second selector sources cannot distinguish the CauchyModulusRefinement route from the DiagonalLimitCompatibility route once those rows agree.

Proof. First identify the CauchyModulusRefinement side. The root terminal selector readback theorem, Definition 885.57, makes every terminal Real or Completion read pass through the displayed selector-budget, window, tolerance, readback, Real-seal, transport, replay, provenance, and local naming rows. Real-consumer coverage, Definition 885.56, then exhausts the consumer surface by the same route.

Next identify the DiagonalLimitCompatibility side. Root readback exactness, Definition 646.127, exposes the StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp seal row, and CauchyLimitSeal boundary as the root readback surface. Root selector terminality and root L10 spine exhaustion, Definition 646.132 and Definition 646.130, make that surface terminal for the same Phase Real and Completion consumers.

Now use the shared-row hypothesis. The StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, CauchyLimitSeal boundary, BHist support, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row are the same on both packets, so the two terminal surfaces display the same L10 selector spine.

Finally apply the non-escape clauses on both sides. The CauchyModulusRefinement readback theorem rules out quotient streams, selected host limits, host real equality, hidden selectors, and second meet rows, while the DiagonalLimitCompatibility terminality and exhaustion results rule out quotient streams, selected limits, host real equality, countable-choice selectors, ambient completeness, and second completion witnesses. No remaining carrier coordinate can separate the two routes. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 consumer spine comparison`

Theorem 885.59 (Cauchy modulus refinement terminal-spine choice refusal)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H$ be an accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet. Suppose an accepted terminal selector read agrees on the displayed selector-budget row T, the selected $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row Q, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal E, the structural BHist route, the componentwise hsame row H, the displayed Cont row C, the Pkg provenance row P, and the local $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ row N.*

Then no countable-choice row, quotient stream, host real equality, ambient completeness coordinate, or alternate selector source can distinguish the Real-completion consumer surface determined by that terminal read.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementTerminalSpineChoiceRefusal`

Proof. First place the terminal selector read on the displayed spine. The root terminal selector readback theorem, Definition 885.57, says that agreement on T, W, Q, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, E, H, C, P, N, and the local naming row determines the same completed selector-readback consumer surface.

Next seal the Real-facing end of the route. Root Real seal exhaustion, Definition 885.52, places the $\mathsf{Real}^{\uparrow}$ seal after the displayed readback rows, while Real-consumer coverage, Definition 885.56, exhausts every Real or Completion consumer by $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$.

Finally use shared-tail selector-window determinacy, Definition 885.55. Once the selector budget, stream window, dyadic tolerance, readback, seal, and structural rows agree, the terminal surface has no remaining carrier coordinate for countable choice, quotient streams, host real equality, ambient completeness, or an alternate selector source. □

Theorem 885.60 (Cauchy modulus refinement shared-tail selector determinacy)**.** *Let $R = (M_0, M_1, U, V, T, W, Q,$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet whose common-window selector, shared-tail handoff, and Real-consumer coverage rows read the same* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ *window. If two selector routes expose the same dyadic tail budget Q and the same* $\mathsf{Real}^{\uparrow}$ *seal E, then the refinement rows determined by those routes are classifier-equal through the displayed* BHist *rows. Equivalently, the routes may differ only by componentwise* hsame *transport along the common window and shared-tail rows; they cannot determine distinct refinement rows under the same tail budget and Real seal.*

Proof. Project both selector routes to the displayed carrier rows. Root window-selector totality, Definition 885.50, forces each route to pass through M_0, M_1, U, V, T before it reaches the selected $\mathsf{StreamName}^{\uparrow}$ window and dyadic budget rows.

Next apply root readback stability, Definition 885.51. Since both routes expose the same Q row and the same $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ window, their readbacks are transported by the componentwise H row and replayed only through the displayed route row C.

Use the shared-tail handoff of Definition 885.54. It identifies the tail-facing route with the same selector-budget, window, tolerance, readback, and seal surface T, W, Q, E, H, C, P, N, so no second shared-tail witness can distinguish the two selectors.

Finally apply Real-consumer coverage, Definition 885.56. The common Real seal is reached only through the displayed finite route, and the packet has no hidden selector, quotient stream, selected host limit, or ambient completeness coordinate. Thus the two projected BHist rows are connected by componentwise hsame transport through the common window and shared-tail routes, which is exactly classifier equality for the refinement rows. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared tail selector determinacy`

Theorem 885.61 (Cauchy modulus refinement selector-budget RegSeqRat Real-seal determinacy)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose two downstream Real-completion routes expose the same selector-budget row T, the same finite-window* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ *readback through W, Q, and the same* $\mathsf{Real}^{\uparrow}$ *seal row E. Then the Real-completion consumer surfaces determined by those routes are classifier-equal through the displayed* BHist *rows, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row N.*

Proof. Project both downstream routes to the accepted carrier. Root window-selector totality, Definition 885.50, forces each route through M_0, M_1, U, V, T before either route reaches a selected finite-window row.

Next use root readback stability, Definition 885.51. Since the two routes expose the same $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ finite-window readback and the same dyadic tolerance rows, W, Q are replayed only through the displayed H, C, P, N rows.

Apply the shared-tail handoff, Definition 885.54. It aligns the selector-budget, finite-window, tolerance, readback, and seal rows with the $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surfaces, so no separate RegSeqRat or stream-name route remains.

Finally apply Real-consumer coverage, Definition 885.56. The displayed packet has no quotient stream, selected host limit, hidden selector, host equality, or ambient completeness coordinate. Therefore two routes with the same T, the same W, Q readback, and the same E seal determine the same Real-completion consumer surface through BHist, H, C, P, and N. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement selector budget regseqrat real seal determinacy`

Theorem 885.62 (Cauchy modulus refinement diagonal handoff determinacy)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E,$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Any diagonal Real-completion handoff for R that exposes the same selector-budget row T, selected* $\mathsf{StreamName}^{\uparrow}$ *window row W, dyadic tolerance row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, and* $\mathsf{Real}^{\uparrow}$ *seal row E determines the same completion-facing consumer surface through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *rows.*

The determination is only through the finite BEDC rows

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No quotient stream, selected host limit, hidden selector, second meet row, host equality, or ambient completeness coordinate supplies another diagonal handoff.

Proof. First force the selector spine. Common-window selector extraction, Definition 885.48, and root window-selector totality, Definition 885.50, make every diagonal handoff read the selected $\mathsf{StreamName}^{\uparrow}$ window and dyadic tolerance through M_0, M_1, U, V, T before it reaches W, Q.

Next pass through the completion-facing rows. Real-completeness handoff, Definition 885.49, places the selected window, dyadic tolerance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal in the displayed order, while shared-tail handoff, Definition 885.54, carries those rows across the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surfaces.

Finally apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. Agreement on T, W, Q, E identifies the resulting completion-facing consumer surfaces through the displayed BHist route, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row. The excluded coordinates are absent from the accepted packet, so none of them can separate the diagonal handoff after those rows are exposed. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementDiagonalHandoffDeterminacy`

Theorem 885.63 (Cauchy modulus refinement uniform-criterion Real route determinacy)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Any* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer that exposes the same selector-budget row T, selected-window row W, dyadic-tolerance row Q, and* $\mathsf{Real}^{\uparrow}$ *seal row E determines the same Real-completion consumer surface through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *rows. No quotient stream, selected host limit, ambient completeness row, hidden selector, or second meet row supplies a competing route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier uniform criterion real route determinacy`

Proof. First force the route to the displayed selector spine. Root window-selector totality, Definition 885.50, places every selected-window read behind M_0, M_1, U, V, T, and root readback stability, Definition 885.51, keeps the resulting W, Q readback on the same H, C, P, N surface.

Next pass through the shared-tail handoff. By Definition 885.54, the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer reaches the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows through the same displayed tail route rather than through a detached completion coordinate.

Use Real-consumer coverage to identify the exposed completion surface. The theorem Definition 885.56 says that every Real-facing consumer read is exhausted by $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so exposing T, W, Q, E leaves no hidden selector, quotient stream, selected host limit, second meet row, or ambient completeness row to choose another path.

Finally apply selector-budget RegSeqRat Real-seal determinacy and tail-budget seal uniqueness, Definition 885.61 and Definition 885.67. The shared selector budget, selected window, dyadic tolerance, and Real seal determine one Real-completion consumer surface through the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. □

Theorem 885.64 (Cauchy modulus refinement uniform diagonal seal correspondence)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Let J be an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail packet and let K be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root packet. Suppose the selector-budget row, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, structural* BHist *route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows of R agree with the corresponding terminal rows read by J and K.*

Then the $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *refinement-selector surface, the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *tail surface, and the* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *root surface expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal exactly when the three surfaces traverse the same displayed BEDC route through*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The correspondence is terminal for Real and Completion consumers: no quotient stream, selected host limit, host real equality, countable-choice selector, ambient completeness coordinate, hidden selector, second meet row, or side-channel completion row supplies another terminal seal surface.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier uniform diagonal seal correspondence`

Proof. For the forward direction, apply the UniformCauchyCriterion route theorem, Definition 885.63. It places the uniform tail consumer on the displayed selector-budget, window, tolerance, readback, seal, transport, replay, provenance, and local naming route. Apply the diagonal handoff theorem, Definition 885.62, to place the diagonal root consumer on the same finite rows.

Now compare the two terminal readings with the refinement-selector surface. The L10 consumer spine comparison, Definition 885.58, identifies the CauchyModulusRefinement and DiagonalLimitCompatibility consumer spines once their $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows agree. Hence a common terminal $\mathsf{Real}^{\uparrow}$ seal exposes the same displayed BEDC route on all three surfaces.

For the reverse direction, assume the three surfaces traverse the same displayed route. Real-consumer coverage, Definition 885.56, exhausts the completion-facing consumer boundary of R by $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. Terminal-spine choice refusal, Definition 885.59, rules out any choice, quotient, host-equality, ambient-completeness, or alternate-selector coordinate after those terminal rows agree.

Therefore the shared route has only one terminal $\mathsf{Real}^{\uparrow}$ seal to expose. The agreed uniform tail packet, diagonal root packet, and refinement selector packet are three readings of that same finite BEDC terminal surface, so their Real and Completion consumers receive the same seal. □

Theorem 885.65 (Cauchy modulus refinement selector-seal two-step transport)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose a second downstream route has selector row V', selector-budget row T', tolerance-readback row Q', and seal row E'. If the displayed* hsame *rows identify V with V' and E with E', and the displayed* Cont *rows carry the source-meet route through U, V', T' and then through T', W, Q', then the two routes determine the same selector-budget row and the same* $\mathsf{Real}^{\uparrow}$ *seal consumer surface through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector seal two step transport`

Proof. First apply selector-stability lock, Definition 885.111. The hsame agreement on V, V' and the continuation through U, V', T' transport the selector route to the same selector-budget surface, so T' cannot supply a detached selected-window ledger.

Next apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. With the selector-budget agreement in place, the continuation through T', W, Q' places the transported readback on the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows, while the hsame agreement on E, E' identifies the $\mathsf{Real}^{\uparrow}$ seal row.

Finally use the non-escape clauses recorded by window-route factorization, Definition 885.66. The accepted packet has no hidden selector, quotient stream, selected host limit, second meet row, side-channel completion row, host real equality, or ambient completeness coordinate. Hence the two-step route from selector agreement to seal agreement exhausts the consumer surface through H, C, P, N. □

Theorem 885.66 (Cauchy modulus refinement window-route factorization)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Every Real-completion consumer route that exposes the selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal factors through the ordered subroute*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

If two such routes expose the same selector-budget, selected-window, dyadic-tolerance, and seal rows T, W, Q, E, then they determine the same consumer surface through componentwise hsame *transport along H, displayed replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row N. No quotient stream, selected host limit, hidden selector, second meet row, side-channel completion row, host real equality, or ambient completeness coordinate can replace that factorization.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier window route factorization`

Proof. First extract the selected-window spine. Common-window selector extraction, Definition 885.48, and root window-selector totality, Definition 885.50, force every selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance read through M_0, M_1, U, V, T before it reaches W, Q.

Next stabilize the readback. Root readback stability, Definition 885.51, transports the same W, Q readback only through H, C, P, N, so a consumer cannot split the $\mathsf{RegSeqRat}^{\uparrow}$ readback away from the displayed window route.

Then pass to the completion-facing seal. Real-completeness handoff, Definition 885.49, places the selected window, dyadic tolerance, regular readback, and Real seal in the displayed order before any completion boundary is consumed.

Finally apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. Agreement on T, W, Q, E identifies the two Real-completion consumer surfaces through componentwise hsame

transport and the displayed C, P, N rows. The excluded coordinates are absent from the accepted packet, so none of them supplies an alternative factorization route. □

Theorem 885.67 (Cauchy modulus refinement tail-budget seal uniqueness)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H,$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose two Real-completion or diagonal-completion tail-budget consumers expose the same selector-budget row T, selected-window row W, dyadic tolerance row Q,* $\mathsf{Real}^{\uparrow}$ *seal row E, and the same structural rows H, C, P, N. Then both consumers determine the same seal surface through the displayed route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No quotient stream, selected host limit, side-channel completion row, host real equality, hidden selector, or second meet row can produce a different $\mathsf{Real}^{\uparrow}$ *tail-budget seal.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier tail budget seal uniqueness`

Proof. First force the route to the common selector spine. Common-window selector extraction, Definition 885.48, together with root window-selector totality, Definition 885.50, makes both consumers pass through M_0, M_1, U, V, T before either can read a selected window.

Next stabilize the visible tail rows. Root readback stability, Definition 885.51, identifies the shared W, Q readback through the same componentwise H transport and displayed replay row C.

The seal row is then fixed by the completion-facing handoff. Real-completeness handoff, Definition 885.49, places E after the same selector-budget, selected-window, dyadic-tolerance, and readback rows, with provenance and local naming carried only by P, N.

Finally apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61, and window-route factorization, Definition 885.66. Since the consumers agree on T, W, Q, E, H, C, P, N, the displayed route determines a unique seal surface; the excluded coordinates are not packet rows and cannot supply another tail-budget seal. □

Theorem 885.68 (Cauchy modulus refinement selector-budget tail-meet non-escape)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Every Real-completion consumer that reads the selector-budget row T, selected-window row W, dyadic tolerance row Q, and* $\mathsf{Real}^{\uparrow}$ *seal row E must first pass through the shared meet and refinement route*

$$M_0, M_1, U, V, T$$

and then through the shared-tail handoff to W, Q, E, H, C, P, N. No quotient stream, hidden selector, second meet row, host real equality, or ambient completeness row can replace that shared-tail meet surface.

Proof. First force the selector spine. Common-window selector extraction, Definition 885.48, and root window-selector totality, Definition 885.50, make every read of T, W, Q pass through M_0, M_1, U, V, T before a selected window is available.

Next pass from the selector spine to the tail-facing interface. The shared-tail handoff of Definition 885.54 identifies the same route with the displayed $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, so the handoff is carried by W, Q, E, H, C, P, N rather than by a detached completion coordinate.

Now apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. Once the consumer has exposed T, W, Q, E, the Real-completion surface is fixed through the displayed BHist route, componentwise hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Finally use tail-budget seal uniqueness, Definition 885.67. A second route with the same selector budget, selected window, dyadic tolerance, and seal collapses to the same tail-budget seal surface. The excluded quotient stream, hidden selector, second meet row, host real equality, and ambient completeness rows are not packet coordinates, so none can bypass the shared-tail meet route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget tail meet nonescape`

Definition 885.69 (Cauchy modulus refinement terminal selector surface)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. The terminal selector surface of R is the displayed consumer route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

from the shared meet rows M_0, M_1, U, through the refinement rows V, T, and into the selected-window, tolerance-readback, Real-seal, transport, replay, provenance, and local naming rows W, Q, E, H, C, P, N.

Two terminal consumer routes agree on this surface exactly when they agree on the selector budget T, the selected window W, the dyadic tolerance Q, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback carried by W, Q, the* $\mathsf{Real}^{\uparrow}$ *seal E, the componentwise* hsame *transport row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* NameCert *row N.*

Lean4 (definition) `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementTerminalSelectorSurface`

Theorem 885.70 (Cauchy modulus refinement terminal selector-seal correspondence)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Two Real-completion consumer routes expose the same terminal selector surface of Definition 885.69 if and only if they expose the same ordered selector-budget row T, selected-window row W, dyadic-tolerance row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row E through the same structural rows H, C, P, N.*

Consequently no terminal consumer can identify the $\mathsf{Real}^{\uparrow}$ *seal of R without passing through the displayed selector-budget route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Proof. For the forward direction, first force both routes to the common selector spine. Root window-selector totality, Definition 885.50, places every selected-window read behind M_0, M_1, U, V, T, while root readback stability, Definition 885.51, keeps the W, Q readback on the displayed H, C, P, N surface.

Next use shared-tail handoff, Definition 885.54. Once the routes expose the same terminal selector surface, the handoff carries the same selector budget, selected window, dyadic tolerance, RegSeqRat readback, and RealUp seal into T, W, Q, E with no detached completion coordinate.

For the reverse direction, assume the two routes expose the same T, W, Q, E, H, C, P, N rows. Window-route factorization, Definition 885.66, makes every Real-completion consumer factor through those rows, and selector-budget RegSeqRat Real-seal determinacy, Definition 885.61, identifies the Real-facing seal surface once those rows agree.

Finally apply tail-budget selector lock, Definition 885.71. Quotient streams, host real equality, hidden selectors, second meet rows, selected host limits, side-channel completion rows, and ambient completeness are not packet coordinates of R, so no off-route consumer can identify the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed selector-budget route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier terminal selector seal correspondence`

Theorem 885.71 (Cauchy modulus refinement tail-budget selector lock)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H, C,$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. If two Real-completion consumers of* R *expose the same selector budget* T*, selected window* W*, dyadic tolerance* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*, then they are classifier-equal through the displayed route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No hidden selector, second meet row, quotient stream, selected host limit, host real equality, or ambient completeness row can split the Real-completion consumer surface once those rows have been read.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier tail budget selector lock`

Proof. First force both consumers through the common selector spine. Root window-selector totality, Definition 885.50, places every selected-window read behind M_0, M_1, U, V, T.

Next stabilize the readback rows. Root readback stability, Definition 885.51, keeps the shared W, Q readback inside the displayed H, C, P, N surface rather than a host-side stream witness.

Apply selector-budget RegSeqRat Real-seal determinacy, Definition 885.61. Since both consumers expose the same T, W, Q, E rows, their Real-facing consumer surfaces are identified through the same BHist route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row.

Finally use tail-budget seal uniqueness, Definition 885.67. The agreement on H, C, P, N locks the seal to the same tail-budget route, and the excluded coordinates are absent from the accepted packet. □

Theorem 885.72 (Cauchy modulus refinement shared-meet RegSeqRat route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *must pass through the shared meet and extraction route*

$$M_0, M_1, U, V, T$$

before reading the selected window W*, dyadic tolerance* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* E*. The consumer cannot obtain the RegSeqRat readback or Real seal from a detached selector, quotient stream, second meet row, host real equality, selected host limit, or ambient completeness coordinate.*

Proof. Start with common-window selector extraction, Definition 885.48. It identifies any downstream selected-window read with the same selector-budget route and therefore places W, Q after T.

Next apply the shared-tail handoff, Definition 885.54. The handoff carries the selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal on the displayed W, Q, E, H, C, P, N rows.

Use Real-consumer coverage, Definition 885.56, to identify the completion-facing consumer surface with the same finite packet. The Real read has no additional carrier coordinate beyond the displayed refinement rows.

Finally apply selector-budget tail-meet non-escape, Definition 885.68. Once the consumer reads T, W, Q, E, every admissible route has already passed through M_0, M_1, U, V, T, so the excluded off-route data cannot supply a second RegSeqRat or Real route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared meet regseqrat route`

Theorem 885.73 (Cauchy modulus refinement shared-budget pullback handoff)**.** *Let $R = (M_0, M_1, U, V, T, W, Q,$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer that uses the terminal selector-seal surface of R enters through the shared-budget pullback route fixed by*

$$U, V, T, W, Q, E, H, C, P, N.$$

If two such consumers agree on those rows, then they expose the same terminal $\mathsf{Real}^{\uparrow}$ *seal surface. No quotient stream, hidden selector, second meet row, host real equality, selected host limit, side-channel completion row, or ambient completeness coordinate supplies an alternate handoff.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared budget pullback`

Proof. First bind the shared budget. The shared-budget pullback theorem, Definition 885.19, makes the meet row U and selector-budget row T determine the selected-window row W and tolerance-readback row Q through the displayed refinement route.

Next route the terminal handoff through the finite sibling surfaces. The shared-tail handoff of Definition 885.54 carries W, Q, E, H, C, P, N across the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces, while Definition 885.72 places every RegSeqRat and Real consumer behind the same shared meet and extraction route U, V, T.

Now identify the terminal selector-seal surface. By Definition 885.70, agreement on the displayed selector-budget, selected-window, tolerance-readback, seal, transport, replay, provenance, and local naming rows is exactly agreement on the terminal selector surface and its $\mathsf{Real}^{\uparrow}$ seal.

Finally apply tail-budget selector lock, Definition 885.71. The excluded off-route data are not coordinates of the accepted packet, so they cannot supply a second handoff once $U, V, T, W, Q, E, H, C, P, N$ have fixed the terminal route. □

Theorem 885.74 (Cauchy modulus refinement terminal Real-seal pullback exhaustion)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. For every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of R, identifying the terminal* $\mathsf{Real}^{\uparrow}$ *seal through the shared-budget pullback is equivalent to exposing the same rows*

$$U, V, T, W, Q, E, H, C, P, N$$

through the shared meet, selector-budget, selected-window, readback, seal, transport, replay, provenance, and local naming surfaces. Conversely, agreement on those rows determines the same terminal $\mathsf{Real}^{\uparrow}$ *seal surface for the consumer.*

No quotient stream, hidden selector, second meet row, host real equality, selected host limit, side-channel completion row, or ambient completeness coordinate contributes an additional terminal seal route.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier terminal real seal pullback exhaustion`

Proof. First force the shared meet and selector route. The shared-meet RegSeqRat route, Definition 885.72, places every $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer behind U, V, T, and the tail-budget selector lock, Definition 885.71, prevents a second selector surface once T, W, Q, E, H, C, P, N are read.

Next pass through the finite sibling handoff. The shared-budget pullback handoff, Definition 885.73, carries the route from U, V, T to W, Q, E, H, C, P, N across the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

$\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces. Thus a consumer that identifies the terminal seal through the pullback must expose exactly the displayed shared-budget rows.

For the converse direction, assume the consumer exposes the same $U, V, T, W, Q, E, H, C, P, N$ rows. Terminal selector-seal correspondence, Definition 885.70, identifies agreement on the selector-budget, selected-window, dyadic-tolerance, readback, seal, transport, replay, provenance, and local naming rows with agreement on the terminal selector surface and its $\mathsf{Real}^{\uparrow}$ seal.

Finally apply the exclusion clauses supplied by tail-budget selector lock and shared-budget pullback handoff. Quotient streams, hidden selectors, second meet rows, host real equality, selected host limits, side-channel completion rows, and ambient completeness coordinates are absent from the accepted packet, so none of them can supply a separate terminal pullback. Hence the displayed row agreement is exactly the consumer boundary for the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 885.75 (Cauchy modulus refinement terminal pullback consumer uniqueness)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose two* $\mathsf{Real}^{\uparrow}$ *or Completion consumers of* R *expose the same shared-budget pullback rows*

$$U, V, T, W, Q, E, H, C, P, N.$$

Then the two consumers expose the same terminal $\mathsf{Real}^{\uparrow}$ *seal consumer surface. No quotient stream, hidden selector, second meet row, host real equality, selected host limit, side-channel completion row, or ambient completeness coordinate can distinguish the two terminal consumers.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier terminal pullback consumer uniqueness`

Proof. First lock the selector boundary. Tail-budget selector lock, Definition 885.71, makes the displayed selector-budget route the only accepted route from the tail budget to the selected-window and tolerance rows.

Next read both consumers through the shared meet. The shared-meet RegSeqRat route, Definition 885.72, forces every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or Completion-facing read behind the same U, V, T spine before W, Q, E can be consumed.

Now pass through the pullback handoff. By Definition 885.73, agreement on $U, V, T, W, Q, E, H, C, P, N$ is agreement on the finite handoff across the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces.

Finally apply terminal Real-seal pullback exhaustion, Definition 885.74. Since the two consumers expose the same displayed pullback rows, their terminal seal surfaces are the same, and the excluded off-route coordinates have no accepted slot from which to separate them. □

Theorem 885.76 (Cauchy modulus refinement selector-budget normal form)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Any consumer route of* R *shared by* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is exhausted by one finite selector-budget normal form:*

$$U, V, T, W, Q, E, H, C, P, N.$$

The rows are, in order, the shared meet row, extraction row, selector-budget row, selected-window row, tolerance-readback row, $\mathsf{Real}^{\uparrow}$ *seal row, and the support, provenance, and local naming rows. No quotient stream, hidden selector, second meet row, host real equality, selected host limit, side-channel completion row, or ambient completeness coordinate supplies another shared consumer route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector budget normal form`

Proof. First force the shared meet and extraction route. The RegSeqRat–StreamName–Real coherence surface, Definition 885.96, and the selector exactness surface, Definition 885.102, place the four sibling consumers behind the same U, V, T spine before a selected window or Real seal is read.

Next expose the finite observation budget. The shared-budget pullback handoff, Definition 885.73, carries that spine through W, Q, E, H, C, P, N across the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces.

Then apply terminal pullback consumer uniqueness, Definition 885.75. Once a consumer exposes the displayed $U, V, T, W, Q, E, H, C, P, N$ rows, it has the same terminal $\mathsf{Real}^{\uparrow}$ seal consumer surface as any other consumer exposing those rows.

Finally use Definition 885.71. The off-route coordinates listed in the statement are not rows of the accepted packet, so the selector-budget normal form is the only shared consumer route available to the four sibling surfaces. □

Theorem 885.77 (Cauchy modulus refinement uniform monotone terminal route)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer and a* $\mathsf{MonotoneCauchy}^{\uparrow}$ *consumer both read the selector-budget row* T*, selected-window row* W*, dyadic-tolerance row* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *of* R *through the same displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *rows.*

Then the two consumers determine one terminal $\mathsf{Real}^{\uparrow}$ *seal surface. The shared surface is exactly the handoff through*

$$U, V, T, W, Q, E, H, C, P, N,$$

with the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *RegSeqRat–Real classifier face and the* $\mathsf{MonotoneCauchy}^{\uparrow}$ *located Real-seal face read as projections of that same finite route. No quotient stream, host real equality, selected limit, trichotomy row, ambient completeness principle, metric-completeness datum, hidden selector, or second meet row supplies a second terminal handoff.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier uniform monotone terminal route`

Proof. First place both consumers on the CauchyModulusRefinement terminal route. The terminal pullback consumer uniqueness theorem, Definition 885.75, identifies every Real or Completion consumer that agrees on $U, V, T, W, Q, E, H, C, P, N$ with the same terminal $\mathsf{Real}^{\uparrow}$ seal consumer surface.

Next read the uniform criterion face. The UniformCauchyCriterion terminal RegSeqRat–Real classifier correspondence, Definition 435.102, says that the RegSeqRat tail classifier, StreamName finite-window classifier, and Real finite-window seal classifier are projections of one ordered route. Under the shared rows of R, that route is the displayed T, W, Q, E, H, C, P, N suffix.

Now read the monotone face. The MonotoneCauchy Real-seal factorization obligation, Definition 438.8, and the tail-window Real-seal certificate, Definition 438.10, force the located interval and monotone tail-window rows to be consumed before the Real seal, with transport, continuation, provenance, and local naming retained.

Finally combine the two projections through the CauchyModulusRefinement shared meet. Since both projected consumers reach the same E, H, C, P, N only after the same selector-budget, selected-window, and dyadic-tolerance rows, terminal pullback uniqueness leaves one Real-seal surface and no off-route coordinate from which a separate uniform or monotone terminal handoff can be read. □

Theorem 885.78 (Cauchy modulus refinement CompleteMetric consumer route)**.** *Let $R = (M_0, M_1, U, V, T, W, Q,$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of R must read the selector-budget normal form*

$$U, V, T, W, Q, E, H, C, P, N$$

before any completion-facing row is exposed. In this route W, Q pass through the displayed $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *surfaces, the readback passes through* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal seal passes through* $\mathsf{Real}^{\uparrow}$*. No quotient stream, selected host limit, ambient completeness coordinate, hidden selector, or second meet row can distinguish a* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer after those rows have been read.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier complete metric consumer route`

Proof. First apply selector-budget normal form, Definition 885.76. It places every shared $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer on the same finite route $U, V, T, W, Q, E, H, C, P, N$.

Next use terminal pullback consumer uniqueness, Definition 885.75. A completion-facing consumer that exposes those rows has the same terminal $\mathsf{Real}^{\uparrow}$ seal surface as any other consumer exposing them.

Finally combine the Real and completion non-escape clauses supplied by the root consumer surface and the uniform monotone terminal route. The packet contains no quotient stream, selected host limit, ambient completeness coordinate, hidden selector, or second meet row, so a $\mathsf{CompleteMetric}^{\uparrow}$ consumer can only see the displayed completion-facing route after the $\mathsf{Real}^{\uparrow}$ seal. □

Corollary 885.79 (Cauchy modulus refinement common-window select strict obstruction)**.** *Let $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet. Suppose a displayed failure, refusal, separation, or countermodel row is read at the common-window select surface after the consumer has exposed the selector-budget row T, selected* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal E, structural* BHist *route, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$ *row N.*

Then that displayed obstruction is strict and local: it blocks only the attempted converse or reuse route that tries to distinguish the same Real-completion consumer surface by a countable-choice row, quotient stream, host real equality, ambient completeness coordinate, alternate selector source, hidden selector, second meet row, selected host limit, or detached readback while the displayed terminal rows agree. It cannot be read as a global impossibility for $\mathsf{CauchyModulusRefinement}^{\uparrow}$*, as a host-side completeness theorem, or as hidden external validation of any off-packet Real object.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCommonWindowSelectStrictObstructi`
↪ variant `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier common window select strict obstruction`

Proof. First place the consumer on the terminal spine. Terminal-spine choice refusal, Definition 885.59, says that once the consumer agrees on T, W, Q, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, E, the structural BHist route, H, C, P, and the local naming row N, no countable-choice row, quotient stream, host real equality, ambient completeness coordinate, or alternate selector source can distinguish the Real-completion consumer surface determined by that terminal read.

Next stabilize the readback that the common-window select surface exposes. Root readback stability, Definition 885.51, keeps the selected-window and tolerance readbacks W, Q on the displayed H, C, P, N route for every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer. Thus the obstruction has no independent readback coordinate from which to range over a detached stream representative or an ambient completion object.

Now identify the exact route being blocked. Common-window selector extraction, Definition 885.48, and root terminal selector readback, Definition 885.57, make the selected-window route and terminal selector surface coincide with the finite packet rows already displayed. Therefore a failure, refusal, separation, or countermodel row at this point can only reject the converse or reuse attempt that asks for another consumer surface while preserving those rows.

Finally use Real-consumer coverage and root Real seal exhaustion, Definition 885.56 and Definition 885.52. They exhaust Real and Completion consumers by

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Since the accepted packet exports no off-route host theorem, external validation row, hidden selector, second meet, quotient stream, selected host limit, or ambient completeness coordinate, the displayed obstruction has only the local scope stated above. □

Theorem 885.80 (Cauchy modulus refinement root-window selector uniqueness)**.** *Let*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

and

$$R' = (M_0', M_1', U', V', T', W', Q', E', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packets. Suppose the two packets expose the same source schedules, shared meet, selector budget, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and structural rows along the displayed root route. Then every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer classifies* E *and* E' *as the same consumer-visible Real seal under that route.*

No Real completeness principle, quotient stream, selected host limit, countable-choice row, host real equality, hidden selector, or second meet row is used in the classification.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root window selector uniqueness`

Proof. First apply root route determinacy, Definition 885.129. Agreement on the source schedules and shared meet places both accepted packets on the same displayed route through U, V, T before a finite window or Real seal can be read.

Next use the Real-window budget pullback and terminal consumer theorems, Definition 885.151 and Definition 885.153. These force every $\mathsf{Real}^{\uparrow}$ or Completion-facing consumer to read the same finite request, window, dyadic readback, regular handoff, selector, disclosure, and support rows before it reaches the seal.

Finally invoke L10 row exactness, Definition 885.150. The L10 consumer inventory is exactly the displayed route with H, C, P, N, so two accepted packets agreeing on those route rows expose one consumer-visible Real seal surface. Since none of the excluded rows is present in the carrier, the uniqueness claim does not use Real completeness. □

Theorem 885.81 (Cauchy modulus refinement selector threshold exhaustion)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route whose public consumer reaches a* $\mathsf{Real}^{\uparrow}$ *or Completion seal through the displayed root-window selector is exhausted by the finite rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Deleting U*,* T*,* W*, or* Q *breaks the route before the seal can be consumed:* U *supplies the selector budget,* T *fixes the public threshold handoff,* W *exposes the finite window, and* Q *exposes the dyadic readback required by the public consumer.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementPacket window selector threshold exhaustion`

Proof. First apply root route determinacy, Definition 885.129. It places every accepted route to a Real or Completion seal on the displayed root route through $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$.

Next use root budget consumer exhaustion, Definition 885.131. The budget-facing consumer cannot reach the seal before reading the selector budget and its finite-window support, so deleting U or W interrupts the route before any public endpoint is available.

Then apply root threshold public exhaustion, Definition 885.133. The public threshold leg must pass through T and the dyadic readback row Q before it can expose the seal; removing either row leaves only a partial route.

Finally invoke the local selector uniqueness theorem, Definition 885.80. Once the route rows are fixed, all Real or Completion-facing consumers read the same displayed seal surface. Thus the listed rows exhaust the accepted public route, and the four named deletions break it before seal consumption. □

Theorem 885.82 (Cauchy modulus refinement Real-seal route nonescape)**.** *Every accepted* $\mathsf{CauchyModulusRefinem}$ *route that reaches a* $\mathsf{Real}^{\uparrow}$ *or Completion seal through the root-window selector is confined to the rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Preserving the seal preserves the selector budget U*, the threshold handoff* T*, the finite window* W*, and the dyadic readback* Q*. Such a route cannot read a Real completeness principle, quotient stream, selected host limit, countable-choice row, host real equality, hidden selector, or second meet row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal route nonescape`

Proof. First apply root route determinacy, Definition 885.129. It places every accepted route to a Real or Completion seal on the displayed root route before any terminal consumer can inspect the seal.

Next use real-window budget pullback and the real-window terminal consumer theorem, Definition 885.151 and Definition 885.153. These force the consumer-facing seal to retain the selector budget, finite window, dyadic readback, and public threshold handoff as displayed rows.

Now invoke L10 row exactness, Definition 885.150. The L10 inventory is exactly the carrier rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so no hidden completeness, quotient, host-limit, choice, equality, selector, or second-meet coordinate can be read.

Finally apply selector threshold exhaustion, Definition 885.81. Deleting U, T, W, or Q breaks the route before the seal can be consumed, hence preserving the seal preserves those four rows and the accepted route is nonescaping. □

Corollary 885.83 (Cauchy modulus refinement root-window selector strict obstruction)**.** *Let an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route reach a* $\mathsf{Real}^{\uparrow}$ *or Completion seal through the root-window selector, and let a displayed failure, refusal, separation, or countermodel row be read as blocking a named converse, reuse, transport, or readback attempt for that selector route. Then the obstruction is strict and local: it blocks only that named attempt at one of the displayed rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Every consumer-facing use of the obstruction factors through the same finite route rows, and any coordinate not listed in this display is rejected as unavailable to the obstruction. In particular, the obstruction cannot be reinterpreted as a global impossibility theorem, a host theorem about all Cauchy presentations, a Real-completeness principle, a quotient-stream argument, a selected host limit, a countable-choice principle, host real equality, a hidden selector, a second meet row, or hidden external validation.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root window selector strict obstruction`

Proof. First apply the Real-seal route nonescape theorem, Definition 885.82. It confines every accepted root-window route reaching a $\mathsf{Real}^{\uparrow}$ or Completion seal to the displayed rows

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

and it states that preserving the seal preserves the selector budget U, threshold handoff T, finite window W, and dyadic readback Q. Thus any failure, refusal, separation, or countermodel row that is consumer-facing along this route can only be read through the named finite inventory; an unlisted coordinate has no place where it can be consumed.

Next apply root-window selector uniqueness, Definition 885.80. Once the source schedules, shared meet, selector budget, finite window, dyadic readback, regular handoff, and structural rows agree along the displayed root route, every $\mathsf{Real}^{\uparrow}$ or Completion-facing consumer classifies the exposed seal by that same route surface. Hence an obstruction row cannot detach from its named converse, reuse, transport, or readback attempt and become a second classification principle for the seal.

Combining the two facts gives the strict local reading. Nonescape removes Real completeness, quotient streams, selected host limits, countable choice, host real equality, hidden selectors, and second meet rows from the available inventory, while uniqueness prevents the same finite obstruction row from being read as a different consumer-visible seal argument. Therefore the displayed obstruction blocks only the named route attempt inside the root-window selector packet and supplies no global impossibility theorem, host theorem, or hidden external validation. □

Theorem 885.84 (Cauchy modulus refinement root source binding)**.** *Every root consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet first reads the source rows* M_0, M_1*, the shared meet row* U*, and the extraction row* V *through the carrier route of Definition 885.1. The root consumer has no admissible access to a downstream selector, seal, or structural row before that shared meet and extraction route is displayed.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root source binding`

Proof. Read the carrier definition. Its packet orders the two source schedules, the shared meet, and the extraction witness as the visible front of the root route.

Apply Definition 885.4. The theorem binds any extraction read of V to the single displayed meet U after M_0, M_1; hence a root consumer cannot substitute a second meet or a hidden source.

Finally use Definition 885.56. Since every Real or Completion consumer is covered by the displayed packet route, every such consumer inherits this first read of M_0, M_1, U, V before any later row is available. □

Theorem 885.85 (Cauchy modulus refinement root selector readback)**.** *For every accepted* CauchyModulusRefinem *packet, selected-window and tolerance readback consume T, W, Q only after the source-meet-extraction row M_0, M_1, U, V has been displayed. Thus any root consumer that asks for the selected-window ledger or the tolerance-readback ledger reaches those rows through the same selector-budget route T.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root selector readback`

Proof. Start from Definition 885.1. The selected-window ledger W and tolerance-readback ledger Q are carrier rows following M_0, M_1, U, V, T.

Apply Definition 885.4 to fix the source side. The extraction row V is already bound to the shared meet route before any selector-budget read is meaningful.

Now apply Definition 885.5. It identifies T as the selector-budget row consumed by the selected-window and tolerance-readback rows, so W, Q are read only after the extraction row and through T. □

Theorem 885.86 (Cauchy modulus refinement paper-Lean readback sync)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet whose root source-meet-extraction route*

$$M_0, M_1, U, V$$

is displayed, the selector-budget row T, selected-window row W, tolerance-readback row Q, seal row E, and structural rows H, C, P, N are preserved along that same carrier route. The readback exposes no second meet, hidden selector, quotient stream, or ambient completeness coordinate as an alternate source of those rows.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier paper lean readback sync`

↪ variant `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement root readback stability`

Proof. Apply Definition 885.84. It fixes the root route through M_0, M_1, U, V before any selector, seal, or structural row can be read.

Next use Definition 885.85. The selected-window and tolerance-readback rows reach W, Q only through the selector-budget row T, so their readback remains on the same displayed carrier route.

Finally apply Definition 885.87. The Real seal and structural rows E, H, C, P, N stay behind the same displayed route, while the excluded quotient stream, hidden selector, second meet, and ambient completeness coordinate have no exported entry point. □

Theorem 885.87 (Cauchy modulus refinement root Real seal non-escape)**.** *Every* Real$^{\uparrow}$ *or Completion consumer of an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet reaches the seal row E only after the displayed route*

$$M_0, M_1, U, V, T, W, Q, H, C, P, N.$$

No quotient stream, selected limit, host real equality, hidden selector, second meet, or ambient completeness row is exported as an alternate route to the Real seal.

Proof. Project the carrier of Definition 885.1. The packet contains the source, meet, extraction, selector-budget, selected-window, tolerance-readback, Real seal, transport, continuation, package, and local naming rows as finite displayed coordinates.

Apply Definition 885.5 and Definition 885.7. Together they order the selector readback T, W, Q before the Real seal E, with the structural rows H, C, P, N stabilizing the same route.

Finally apply Definition 885.56. Every Real or Completion consumer is exhausted by the displayed route, so the excluded quotient stream, selected limit, host real equality, hidden selector, second meet, and ambient completeness rows have no exported entry point. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root real seal nonescape`

Theorem 885.88 (Cauchy modulus refinement root consumer package)**.** *Every root consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet factors through*

$$M_0, M_1, U, V, T, W, Q, H, C, P, N$$

before reading any $\mathsf{Real}^{\uparrow}$ *or Completion seal row. The package exports no quotient stream equality, selected limit, host real equality, hidden selector, second meet, or ambient completeness coordinate as an alternate consumer route.*

Proof. First apply Definition 885.84. This binds every root consumer to the source rows M_0, M_1, the shared meet row U, and the extraction row V before any downstream read is available.

Next apply Definition 885.85. The selected-window ledger and tolerance-readback ledger are consumed only through the selector-budget row T, so the consumer reaches T, W, Q after the source-meet-extraction route has been displayed.

Finally apply Definition 885.87. The Real or Completion seal row is reached only after the displayed structural route H, C, P, N has stabilized the same packet, and the excluded quotient stream equality, selected limit, host real equality, hidden selector, second meet, and ambient completeness coordinates have no exported entry point. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root consumer package`

Theorem 885.89 (Cauchy modulus refinement root route completeness)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing root read is complete exactly at the displayed route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The source rows, shared meet, extraction row, selector-budget row, selected-window ledger, tolerance-readback ledger, Real seal row, and structural rows jointly exhaust the public consumer surface. No quotient stream equality, selected limit, host real equality, hidden selector, second meet, or ambient completeness coordinate is exported as another root route.

Proof. Apply Definition 885.84. It fixes the source rows M_0, M_1, the shared meet U, and the extraction row V as the first public root source.

Next use Definition 885.85. The selected-window and tolerance-readback ledgers are available only through the selector-budget row T, so the route continues through T, W, Q without opening a hidden selector.

Then apply Definition 885.87. The Real seal row E and the structural rows remain behind the same displayed carrier route, and the excluded quotient stream, selected limit, host real equality, second meet, and ambient completeness coordinates have no exported read.

Finally apply Definition 885.88. It packages the Real and Completion consumers over the same carrier coordinates. Since the packet has no public coordinate beyond $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ for these reads, carrier coordinate exhaustion gives the claimed complete root route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root route completeness`

Theorem 885.90 (Cauchy modulus refinement shared selector determinacy)**.** *For every accepted* CauchyModulusRefinement$^{\uparrow}$ *packet, suppose two root-facing reads expose the same source rows M_0, M_1, shared meet U, extraction row V, selector-budget row T, selected-window row W, tolerance-readback row Q, Real seal row E, and structural rows H, C, P, N. Then the two reads are the same accepted selector route up to the displayed* hsame *transport and* Cont *replay. No hidden selector, second meet, quotient stream, selected limit, host equality, or ambient completeness row can distinguish another accepted root route.*

Proof. Apply Definition 885.84. The shared source rows force the same M_0, M_1, U, V spine for both reads, so there is no separate source-meet branch on which a second selector route could start.

Next apply Definition 885.85. The selector-budget, selected-window, and tolerance-readback rows are read back through the same displayed selector surface T, W, Q, and the carrier exposes no hidden selector coordinate beyond that surface.

Use Definition 885.87. The Real seal row and structural rows remain inside the same E, H, C, P, N packet under componentwise hsame transport and displayed Cont replay, so quotient streams, selected limits, host equalities, second meets, and ambient completeness rows are not root-facing witnesses.

Finally invoke Definition 885.89. Since every accepted Real or Completion-facing root read is exhausted by $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, two reads with those rows displayed have no remaining public coordinate on which they can differ. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared selector determinacy`

Theorem 885.91 (Cauchy modulus refinement selector-meet noninterference)**.** *For an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet whose root route displays*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

any consumer request for a hidden selector, second meet, quotient stream, selected limit, host equality, or ambient completeness row cannot alter the accepted selector route. Every accepted read remains determined by the shared meet U, extraction row V, selector-budget row T, selected-window row W, tolerance-readback row Q, Real-seal row E, and structural rows H, C, P, N.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector meet noninterference`

Proof. First apply Definition 885.84. The accepted root route must begin with the source rows M_0, M_1, the shared meet U, and the extraction row V, so a hidden second meet has no source position from which to change the selector route.

Next use Definition 885.85. The selector-facing rows T, W, Q are read only after the source-meet-extraction route has been displayed, and their readback is tied to the same selector-budget row T.

Apply Definition 885.87 and Definition 885.89. They keep the Real seal and structural rows on the displayed route while excluding quotient streams, selected limits, host equalities, hidden selectors, second meets, and ambient completeness rows from the exported packet.

Finally invoke Definition 885.90. Once the displayed rows agree up to componentwise hsame transport and displayed Cont replay, no remaining root-facing coordinate can distinguish another accepted selector route. The requested hidden data are therefore noninterfering with the accepted route. □

Theorem 885.92 (Cauchy modulus refinement completion consumer route determinacy)**.** *Suppose two accepted* CauchyModulusRefinement$^{\uparrow}$ *completion-facing reads display the same rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Then every Real$^{\uparrow}$ *or Completion consumer reaches the same selector route up to the displayed* hsame *transport and* Cont *replay. No hidden selector, second meet, quotient stream, selected limit, host equality, ambient completeness coordinate, or detached completion seal can distinguish a second accepted completion route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier completion consumer route determinacy`

Proof. Apply Definition 885.84. The shared rows M_0, M_1, U, V bind both completion-facing reads to the same source, shared-meet, and extraction spine before any selector-facing row is available.

Next use Definition 885.85. The displayed rows T, W, Q are read back only through the selector-budget route T, so the selected-window and tolerance-readback consumers have no separate selector route to follow.

Apply Definition 885.87 and Definition 885.89. They keep the Real seal and the structural rows E, H, C, P, N on the same displayed route for every Real$^{\uparrow}$ or Completion consumer, while excluding quotient streams, selected limits, host equalities, ambient completeness coordinates, and detached completion seals from the exported packet.

Finally invoke Definition 885.90 and Definition 885.91. Once the displayed rows agree, componentwise hsame transport and displayed Cont replay determine the accepted selector route, and the hidden completion data listed in the statement cannot distinguish another accepted completion route. □

Theorem 885.93 (Cauchy modulus refinement uniform diagonal consumer pullback)**.** *Let an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet expose the shared route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Any UniformCauchyCriterion$^{\uparrow}$ *or* DiagonalLimitCompatibility$^{\uparrow}$ *consumer of that packet factors through the same finite selector route*

$$U, V, T, W, Q, E, H, C, P, N$$

before any Real$^{\uparrow}$ *or Completion endpoint is public. The consumer first reads the shared meet U, extraction row V, selector-budget row T, selected-window row W, tolerance-readback row Q,* Real$^{\uparrow}$ *seal row E, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N.*

Thus the UniformCauchyCriterion and DiagonalLimitCompatibility routes have one consumer pullback over the displayed CauchyModulusRefinement selector surface. No quotient stream, selected host limit, host real equality, countable-choice schedule, detached completion seal, ambient completeness coordinate, hidden selector, or second meet supplies another route to the endpoint.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier uniform diagonal consumer pullback`

Proof. Start with the diagonal-facing side. The diagonal-budget cofinality theorem Definition 885.98 keeps every diagonal-budget consumer on the displayed route through $U, V, T, W, Q, E, H, C, P, N$, and it excludes hidden selectors, second meets, quotient streams, host equality, selected limits, and ambient completeness from that read.

Next attach the criterion side. The CauchyCriterion selector cofinality theorem Definition 885.100 places the CauchyCriterion and UniformCauchyCriterion consumers on the same selector route before any criterion output is exposed. The UniformCauchyCriterion diagonal-budget pullback Definition 435.74 then identifies the uniform family read with the diagonal-budget route rather than with a second schedule or completion coordinate.

Now use the completion-facing determinacy theorem Definition 885.92. Once the rows $M_0, M_1, U, V, T, W, Q, E,$ are displayed, every $\mathsf{Real}^{\uparrow}$ or Completion consumer reaches the same selector route up to the visible hsame transport and Cont replay.

The pullback is therefore the finite selector surface listed in the statement. The excluded quotient streams, selected host limits, host real equalities, countable-choice schedules, detached completion seals, ambient completeness coordinates, hidden selectors, and second meets have no displayed row from which to alter the common consumer route before the endpoint becomes public. □

Theorem 885.94 (Cauchy modulus refinement budget naturality)**.** *Suppose two accepted* CauchyModulusRefinemen *packets share the same source schedules M_0, M_1, meet row U, extraction row V, selector budget T, selected window W, tolerance readback Q, Real seal E, and structural rows H, C, P, N. Then every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or Completion consumer reads the same budget-refinement route up to the displayed* hsame *transport and* Cont *replay.*

The route is the displayed carrier path

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

No quotient stream equality, selected limit, host equality, hidden selector, second meet, or ambient completeness coordinate can distinguish another budget-refinement read for those consumers.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier budget naturality`

Proof. First apply Definition 885.84. The shared source schedules M_0, M_1, meet row U, and extraction row V force both accepted packets to begin with the same source-meet-extraction route before any selector-budget data can be read.

Next use Definition 885.85. The selected-window and tolerance-readback rows reach W, Q only through the displayed selector budget T. Thus $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers read the same finite-window and tolerance route rather than a hidden selector.

Apply Definition 885.89. Every $\mathsf{Real}^{\uparrow}$ or Completion-facing root read is exhausted by the displayed route $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so the Real seal and completion handoff have no detached route.

Finally combine Definition 885.90 with Definition 885.91. Since the displayed rows agree, componentwise hsame transport and displayed Cont replay determine the accepted consumer route, while quotient stream equality, selected limits, host equality, hidden selectors, second meets, and ambient completeness remain outside the public packet. □

Theorem 885.95 (Cauchy modulus refinement complete-metric no-choice interface)**.** *For an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet, suppose a terminal read reaches two completion-facing rows through the same displayed interfaces*

$$\mathsf{Cont}(\text{terminal}, c, \text{completionA})$$

and

$$\mathsf{Cont}(\text{terminal}, c, \text{completionB}),$$

and suppose both completion rows are packaged by the same Pkg *provenance. Then the two completion reads are classified by* hsame.

The interface exposes only the finite terminal row, the two completion rows, and the package rows. It exports no quotient stream, selected limit, host equality, choice schedule, or ambient completeness coordinate from which a second completion read could be selected.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCompleteMetricNoChoiceInterface`

Proof. First use the accepted carrier route from Definition 885.1. The rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ provide the source schedules, shared meet, extraction row, selector budget, selected window, tolerance readback, Real seal, terminal transport, displayed continuation row, package provenance, and local naming row as finite packet data.

Next apply Definition 885.85 and Definition 885.89. They place the selected-window and tolerance-readback route before the terminal structural rows and exhaust every Real or Completion-facing root read at the displayed carrier path, with no hidden selector, quotient stream, selected host limit, host equality, choice schedule, or ambient completeness coordinate.

It remains to compare the two completion-facing endpoints. The Lean proof closes this step by applying cont_deterministic to the two displayed continuation witnesses $\mathsf{Cont}(\text{terminal}, c, \text{completionA})$ and $\mathsf{Cont}(\text{terminal}, c, \text{completionB})$. Thus the endpoints are hsame-classified over the visible terminal interface, and the same Pkg provenance records that both completion rows belong to the finite packet surface. □

Theorem 885.96 (Cauchy modulus refinement RegSeqRat–StreamName–Real coherence)**.** *For an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet displaying*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

every $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer reads the same budget-refinement route up to the displayed* hsame *transport and* Cont *replay. The shared route is the carrier path through the source schedules, shared meet, extraction row, selector budget, selected window, tolerance readback, Real seal, transport, continuation, provenance, and local naming rows.*

No such consumer may split the route into a separate stream selector, detached Real seal, quotient stream equality, hidden selector, second meet, selected limit, host equality, or ambient completeness coordinate. The consumer chapters chapter 301, chapter 126, and chapter 38 therefore consume one displayed budget-refinement route rather than three independent routes.

Proof. First apply Definition 885.84. It binds the source schedules M_0, M_1, shared meet U, and extraction row V as the common root of every accepted budget-refinement read.

Next use Definition 885.85. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ reads of selected windows and tolerance data pass through the same selector budget T and the displayed rows W, Q, with no separate stream selector exposed.

Apply Definition 885.89. It places the $\mathsf{Real}^{\uparrow}$ seal row E and structural rows H, C, P, N on the same packet route and excludes quotient streams, selected limits, host equalities, hidden selectors, second meets, and ambient completeness rows from the public surface.

Finally combine Definition 885.90 with Definition 885.94. The displayed rows determine one accepted budget-refinement route up to hsame transport and Cont replay, and the three named consumer chapters read that same route through their local NameCert surfaces. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier regseq streamname real coherence`

Theorem 885.97 (Cauchy modulus refinement RegSeqRat–StreamName–Real terminal factorization)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet whose displayed route is*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

all $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and Completion consumers factor through the same terminal finite route. The factorization is the carrier path through the source schedules, shared meet, extraction row, selector budget, selected-window row, tolerance-readback row, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No detached completion seal, hidden selector, second meet, quotient stream, selected limit, host equality, or ambient completeness coordinate can supply a separate consumer route.*

Proof. First apply Definition 885.84. It fixes M_0, M_1, U, V as the shared source-meet-extraction spine for every accepted root read before any selector or seal row is available.

Next use Definition 885.85. The selected-window and tolerance-readback rows W, Q are read only through the displayed selector-budget row T, so $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ consumers cannot detach a hidden selector from the source spine.

Apply Definition 885.89. Every $\mathsf{Real}^{\uparrow}$ or Completion-facing root read is exhausted by $M_0, M_1, U, V, T, W, Q, E, H$ with no additional quotient, selected-limit, host-equality, or ambient-completeness coordinate.

Finally combine Definition 885.90, Definition 885.91, and Definition 885.96. Determinacy identifies any two reads with the displayed rows, noninterference rules out hidden selector and second-meet requests, and coherence places the RegSeqRat, StreamName, and Real consumer surfaces on the same route. Completion consumers are already covered by route completeness, so all four consumer classes factor through the terminal finite packet. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier regseq streamname real terminal factorization`

Theorem 885.98 (Cauchy modulus refinement diagonal-budget cofinality route)**.** *Let an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Then the same route remains cofinal when it is read by a diagonal-budget consumer. The diagonal-budget consumer can read only the shared meet row U, the extraction row V, the selector-budget

row T, the selected-window row W, the tolerance-readback row Q, the Real$^{\uparrow}$ *seal row E, and the structural rows H, C, P, N already exposed by the root route. It cannot introduce a hidden selector, second meet, quotient stream, host equality, selected limit, or ambient completeness coordinate.*

Proof. First apply Definition 885.84. The source rows M_0, M_1 force every accepted read to pass through the same shared meet U and extraction row V before any selector-facing budget can be read.

Next use Definition 885.85. The diagonal-budget consumer reaches the selected-window and tolerance-readback rows only through the displayed selector-budget row T, so W and Q cannot come from a hidden selector.

Apply Definition 885.89. It exhausts every Real$^{\uparrow}$ or Completion-facing root read at $M_0, M_1, U, V, T, W, Q, E, H, C$ with no quotient stream, selected limit, host equality, second meet, hidden selector, or ambient completeness coordinate.

Finally invoke Definition 885.92. Since completion consumers are determined by the same displayed route up to the visible hsame transport and Cont replay, a diagonal-budget read has no separate coordinate from which to select a different cofinal route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier diagonal budget cofinality route`

Theorem 885.99 (Cauchy modulus refinement selector nonempty route)**.** *Let an accepted* CauchyModulusRefineme *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Any StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *or Completion consumer that reads selected-window or tolerance data must first expose the selector-budget row T as a nonempty route following the shared meet U and extraction row V. No hidden selector, second meet, quotient stream, selected limit, host equality, or ambient completeness coordinate can supply the selected-window row W, tolerance-readback row Q, or Real seal row E.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector nonempty route`

Proof. Start with Definition 885.84. It binds the source rows M_0, M_1 to the shared meet U and extraction row V before any selected-window or tolerance read can be accepted.

Next apply Definition 885.85. The selected-window and tolerance-readback rows W, Q are available only through the displayed selector-budget row T, so a consumer that reads either row has already exposed a nonempty selector-budget route.

Use Definition 885.87. The Real seal and structural rows E, H, C, P, N remain behind the same carrier route, and the excluded quotient stream, selected limit, host equality, hidden selector, second meet, and ambient completeness coordinate have no exported entry point.

Finally apply Definition 885.98. Completion-facing consumers that read the diagonal budget stay cofinal on the displayed route through $U, V, T, W, Q, E, H, C, P, N$, so the selector-budget route cannot be replaced by a detached completion or ambient-completeness coordinate. □

Theorem 885.100 (Cauchy modulus refinement CauchyCriterion selector cofinality)**.** *Let an accepted* CauchyModulusRefinement$^{\uparrow}$ *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Let a $\mathsf{CauchyCriterion}^{\uparrow}$ *consumer, read through chapter 392, share the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows with this displayed route. Then the consumer must read the selector route through the shared meet* U, *extraction row* V, *selector-budget row* T, *selected-window row* W, *tolerance-readback row* Q, *and* $\mathsf{Real}^{\uparrow}$ *seal row* E, *with the support rows* H, C, P, N, *before exposing any CauchyCriterion output.*

Equivalently, the CauchyCriterion selector surface is cofinal on the displayed CauchyModulusRefinement budget route. It has no detached selector, second meet, quotient stream, selected limit, host equality, ambient completeness coordinate, or separate $\mathsf{Real}^{\uparrow}$ *seal from which to expose a criterion output.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier cauchycriterion selector cofinality`

Proof. First apply Definition 885.96. The shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows already force the consumer to read one budget-refinement route through the displayed source schedules, shared meet, extraction row, selector budget, selected window, tolerance readback, seal row, and support rows.

Next use Definition 885.98. It says that any diagonal-budget or completion-facing read remains cofinal on $U, V, T, W, Q, E, H, C, P, N$, so the CauchyCriterion consumer cannot obtain its selector data from a second meet, hidden selector, quotient stream, selected limit, host equality, or ambient completion coordinate.

Apply Definition 885.99. Since the consumer asks for selector-facing CauchyCriterion output, the selector-budget row T must already be exposed as a nonempty route after U and V, and the selected-window and tolerance-readback rows W, Q must be read through that route.

Finally invoke the sibling compatibility theorem Definition 392.39 from chapter 392. Its CauchyCriterion selector surface factors through the same finite StreamName–RegSeqRat–DyadicRatCore–Real packet before any $\mathsf{Real}^{\uparrow}$ seal is exposed, and therefore the shared CauchyModulusRefinement route is the required cofinal selector route for the criterion output. □

Theorem 885.101 (Cauchy modulus refinement criterion-output boundary)**.** *Let an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Let a $\mathsf{CauchyCriterion}^{\uparrow}$ *consumer and a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *consumer share the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows with the displayed route. Then both consumers expose the same finite criterion-output boundary through*

$$U, V, T, W, Q, E, H, C, P, N.$$

Neither consumer can add a detached selector, second meet, quotient stream, selected limit, host equality, countable-choice schedule, ambient completeness coordinate, or separate $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier criterion output boundary`

Proof. For the $\mathsf{CauchyCriterion}^{\uparrow}$ consumer, apply Definition 885.100. The criterion output must be read through the shared meet U, extraction row V, selector-budget row T, selected-window row W, tolerance-readback row Q, $\mathsf{Real}^{\uparrow}$ seal row E, and support rows H, C, P, N.

For the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ consumer, apply Definition 435.74. The uniform diagonal-budget read identifies its compatible window, tolerance, readback, and seal coordinates with the

same finite route rather than with a second schedule, quotient stream, selected limit, or completion coordinate.

Now use Definition 885.90. Since the two consumers share the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, componentwise hsame transport and displayed Cont replay identify their criterion-output route with the accepted CauchyModulusRefinement selector route.

Finally apply Definition 885.91. The shared meet and selector rows already determine the public boundary $U, V, T, W, Q, E, H, C, P, N$, so an off-route selector, second meet, quotient stream, selected limit, host equality, countable-choice schedule, ambient completeness coordinate, or separate $\mathsf{Real}^{\uparrow}$ seal cannot be introduced by either consumer. □

Theorem 885.102 (Cauchy modulus refinement RegSeqRat–Real selector exactness)**.** *For an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route displaying*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *tail read, the* $\mathsf{StreamName}^{\uparrow}$ *selected-window read, and the* $\mathsf{Real}^{\uparrow}$ *seal read are three projections of the same selector route through* U, V, T, W, Q, E*. No hidden selector, second meet, quotient stream, host equality, detached seal, or ambient completeness coordinate can supply one projection without the other two.*

Proof. First apply Definition 885.84 and Definition 885.85. They bind the accepted route to the source rows M_0, M_1, the shared meet U, the extraction row V, and the selector-budget row T before either the selected-window row W or the tolerance-readback row Q can be inspected.

Next use Definition 885.96. The $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers read one budget-refinement route through $U, V, T, W, Q, E, H, C, P, N$ rather than three independent projections.

Apply Definition 885.98 and Definition 885.99. These theorems keep diagonal-budget and selector-facing reads cofinal on the same displayed selector route and make T a nonempty entry before W, Q, E can be consumed.

Finally invoke Definition 885.100. The CauchyCriterion consumer sees the same StreamName–RegSeqRat–DyadicRatCore–Real packet before exposing its output. Hence none of the excluded coordinates can provide just one of the RegSeqRat, StreamName, or Real projections while omitting the other two. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier regseq real selector exactness`

Theorem 885.103 (Cauchy modulus refinement RegSeqRat–Real selector lockstep)**.** *For an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route displaying*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *tail projection, the* $\mathsf{StreamName}^{\uparrow}$ *selected-window projection, and the* $\mathsf{Real}^{\uparrow}$ *seal projection are lockstep projections of the same selector route through* U, V, T, W, Q, E*. No hidden selector, second meet, quotient stream, detached seal, selected limit, host equality, or ambient completeness coordinate can alter one projection without altering the other two.*

Proof. First apply Definition 885.84 and Definition 885.85. They bind the displayed route to the source rows M_0, M_1, the shared meet U, the extraction row V, and the selector-budget row T before any of the three consumer projections can be read.

Next use Definition 885.96. It places the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer surfaces on one budget-refinement route rather than on separate tail, stream, and seal routes.

Apply Definition 885.99. The selected-window and tolerance-facing reads must expose the nonempty selector-budget route through T, W, Q before a projected tail or seal can be consumed.

Finally invoke Definition 885.102. Since the RegSeqRat tail read, StreamName selected-window read, and Real seal read are already exact projections of the same selector route through U, V, T, W, Q, E, changing one projection would change the shared route and therefore the other two projections as well. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier regseq real selector lockstep`

Theorem 885.104 (Cauchy modulus refinement finite observation budget pullback)**.** *Let an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Let this selector route be consumed by a $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *packet as in chapter 435, and let*

$$F = (B, S, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F)$$

be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet as in chapter 782. Suppose the shared* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces of chapter 126, chapter 301, chapter 291, and chapter 38 identify the CauchyModulusRefinement selected-window, tolerance-readback, regular handoff, and Real seal rows with the finite observation budget rows by displayed componentwise* hsame *transports*

$$W\mathsf{hsame}W_F, \qquad Q\mathsf{hsame}D_F, \qquad T\mathsf{hsame}R_F, \qquad E\mathsf{hsame}E_F.$$

Then every Real-facing read of the common route factors through the CauchyModulusRefinement rows

$$U, V, T, W, Q, E, H, C, P, N$$

together with the finite observation budget rows

$$B, S, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F$$

before any $\mathsf{Real}^{\uparrow}$ *seal is exposed. Thus finite observation budgets are the readable selector surface for this Real-completion route: the pullback introduces no quotient stream, selected limit, host equality, countable choice, hidden observation oracle, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier finite observation budget pullback`

Proof. Start with the CauchyModulusRefinement side. By Definition 885.100, a CauchyCriterion or UniformCauchyCriterion consumer of the route must read the selector through the shared meet U, extraction row V, selector-budget row T, selected-window row W, tolerance-readback row Q, Real seal row E, and support rows H, C, P, N before exposing criterion output.

Next apply Definition 885.102 and Definition 885.103. They make the $\mathsf{RegSeqRat}^{\uparrow}$ tail projection, $\mathsf{StreamName}^{\uparrow}$ selected-window projection, and $\mathsf{Real}^{\uparrow}$ seal projection exact lockstep projections of the same selector route, so no Real-facing read can detach the seal from U, V, T, W, Q, E.

Read the finite observation packet through its public route. The public export theorem Definition 782.14 exposes the finite budget-to-seal route $B \to S \to W_F \to D_F \to R_F \to E_F$ together with H_F, C_F, P_F, N_F, and the UniformCauchyCriterion pullback Definition 782.20 identifies that route with the uniform criterion consumer on the displayed shared rows.

Finally use terminal route no-branching and endpoint exactness, Definition 782.28 and Definition 782.29. The displayed transports identify W, Q, T, E with W_F, D_F, R_F, E_F, so the common consumer has only the finite pullback surface listed in the statement before the Real seal is reached. The non-escape clauses in the cited theorems exclude quotient streams, selected limits, host equality, countable choice, hidden observation oracles, and ambient completeness coordinates from the pullback. □

Theorem 885.105 (Cauchy modulus refinement terminal budget route exhaustion)**.** *Let an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route display*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Let this route be consumed by $\mathsf{Real}^{\uparrow}$ *or Completion consumers through the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *faces named in the surrounding chapters. Suppose an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet*

$$F = (B, S, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F)$$

is identified with the common route by the displayed componentwise transports of Definition 885.104. Then every Real-facing or Completion-facing consumer of the common route factors through the combined finite surface

$$U, V, T, W, Q, E, H, C, P, N$$

and

$$B, S, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F.$$

No quotient stream, selected host limit, host equality, countable choice, hidden observation oracle, detached seal, or ambient completeness coordinate supplies another terminal route for those consumers.

Proof. First use Definition 885.97. It makes the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and Completion reads factor through the displayed CauchyModulusRefinement route $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$.

Next apply Definition 885.98 and Definition 885.100. Diagonal-budget, CauchyCriterion, and UniformCauchyCriterion consumers remain cofinal on the same selector route before any Real or Completion endpoint is exposed.

Now use Definition 885.102 together with Definition 885.103. The $\mathsf{RegSeqRat}^{\uparrow}$ tail read, $\mathsf{StreamName}^{\uparrow}$ selected-window read, and $\mathsf{Real}^{\uparrow}$ seal read are exact lockstep projections of U, V, T, W, Q, E, so a consumer cannot detach a host limit or seal coordinate from the common route.

Finally invoke Definition 885.104, Definition 782.28, and Definition 782.29. The finite observation rows $B, S, W_F, D_F, R_F, E_F, H_F, C_F, P_F, N_F$ are the pullback surface for the shared selector route, and terminal no-branching plus endpoint exactness excludes quotient streams, selected host limits, host equality, countable choice, hidden observation oracles, detached seals, and ambient completeness coordinates as alternate terminal routes. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier terminal budget route exhaustion`

Theorem 885.106 (Cauchy modulus refinement current phase spine)**.** *Every accepted* CauchyModulusRefinement$^{\uparrow}$ *consumer for the current* Real$^{\uparrow}$ *and Completion route factors through four displayed faces: the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* StreamName$^{\uparrow}$ *finite-window row, the* RegSeqRat$^{\uparrow}$ *readback row, and the* Real$^{\uparrow}$ *seal. These faces are read from the same finite* BHist *route through displayed* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *rows, and the route admits no quotient stream, selected limit, host real equality, choice data, hidden selector, second meet row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.RootUnblockObligationSurface.CauchyModulusRefinementCarr current phase spine`
Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier current phase spine`

Proof. Begin with root source binding and root selector readback. The accepted route displays $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so the current consumer must read source, common meet, extraction, selector budget, selected window, tolerance readback, and seal before any endpoint is exposed.

Use route completeness and the RegSeqRat, StreamName, and Real coherence theorems in the surrounding surface. They identify the tolerance ledger, finite-window row, regular-sequence readback, and Real seal as projections of the same route rather than independent consumers.

Apply the diagonal budget cofinality route, the finite observation budget pullback, and terminal budget route exhaustion. Together they force every Real-facing or Completion-facing read through the four displayed faces and the support rows H, C, P, N.

The refused coordinates are exactly those excluded by the terminal route. Quotient streams, selected limits, host real equality, choice data, hidden selectors, second meet rows, and ambient completeness would each supply an endpoint outside the displayed BHist, hsame, Cont, Pkg, and NameCert route, so none is accepted. □

Theorem 885.107 (Cauchy modulus refinement open phase spine lock)**.** *For the current* CauchyModulusRefinemen *route, any open* Real$^{\uparrow}$ *or Completion consumer that names a refinement endpoint while leaving one of the four faces absent is held at the spine boundary. The required faces are the* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* StreamName$^{\uparrow}$ *finite-window row,* RegSeqRat$^{\uparrow}$ *readback row, and* Real$^{\uparrow}$ *seal, supported by* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Read the route through the current-phase spine of Definition 885.106. It supplies the four public faces and the support rows as one finite route.

If the tolerance ledger is absent, the consumer has no dyadic budget surface. If the finite-window row is absent, it has no StreamName observation boundary. If the RegSeqRat readback is absent, the regular tail cannot be read back from the selector. If the Real seal is absent, no Real or Completion endpoint is exposed.

The root source binding, root selector readback, and route completeness steps do not add a substitute face. They only route the accepted rows to the endpoint.

Thus an open consumer missing any displayed face remains at the spine boundary. The same refusal excludes quotient streams, selected limits, host real equality, choice data, hidden selectors, second meet rows, and ambient completeness from serving as replacements. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier open phase spine lock`

Theorem 885.108 (Cauchy modulus refinement four-face consumer boundary)**.** *The consumer boundary for the current* CauchyModulusRefinement$^{\uparrow}$ *handoff to* Real$^{\uparrow}$ *and Completion is exactly the four-face route consisting of the* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* StreamName$^{\uparrow}$ *finite-window row,* RegSeqRat$^{\uparrow}$ *readback row, and* Real$^{\uparrow}$ *seal. Any accepted downstream consumer must preserve those faces by displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before reading a Real or Completion endpoint.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier four face consumer boundary`

Proof. First compose root source binding with root selector readback. This places the refinement source, selector, budget, window, tolerance readback, and seal in one route.

Next use route completeness together with RegSeqRat, StreamName, and Real coherence. The three coherence faces and the dyadic tolerance ledger are therefore read as the same current route, not as four unrelated records.

Now apply diagonal budget cofinality, finite observation budget pullback, and terminal budget route exhaustion. These theorems move every downstream consumer through the finite budget surface before it reaches the endpoint, so the boundary is the four displayed faces plus H, C, P, N.

It remains to rule out extra terminal data. Quotient streams, selected limits, host real equality, choice data, hidden selectors, second meet rows, and ambient completeness would bypass at least one displayed face. Since the accepted carrier has no such coordinate, the consumer boundary is exactly the four-face route. □

Corollary 885.109 (Cauchy modulus refinement root source binding strict obstruction)**.** *Let a displayed failure, refusal, separation, or countermodel row in the root-unblock surface challenge an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet by asking a root consumer either to read a selector, seal, or structural row before the source route*

$$M_0, M_1, U, V$$

is displayed, or to reach the Real$^{\uparrow}$ *or Completion seal through a quotient stream, selected limit, host real equality, hidden selector, second meet, or ambient completeness coordinate outside*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Then the obstruction is strict and local. It blocks only the named converse or reuse route that makes that challenged read. It does not assert a global impossibility for CauchyModulusRefinement$^{\uparrow}$ *packets, does not become a host theorem about* Real$^{\uparrow}$ *or Completion, and does not provide hidden external validation of the seal.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root source binding strict obstruction`

Proof. First use root source binding, Definition 885.84. Every root consumer of an accepted packet must first read M_0, M_1, the shared meet U, and the extraction row V through the carrier route of Definition 885.1. Hence a displayed obstruction to an early selector, seal, or structural read says only that the challenged read has not factored through the required source-meet-extraction route. It is a refusal of that local converse or reuse route, not a statement that the carrier cannot be accepted.

Now use root Real seal non-escape, Definition 885.87. Every $\mathsf{Real}^{\uparrow}$ or Completion consumer reaches the seal row E only after the displayed route

$$M_0, M_1, U, V, T, W, Q, H, C, P, N$$

has been displayed, and the theorem exports no quotient stream, selected limit, host real equality, hidden selector, second meet, or ambient completeness row as an alternate route to the seal. Thus an obstruction row aimed at one of those unlisted coordinates rejects precisely that coordinate as an entry point into the accepted packet.

Combining the two cited results, every consumer-facing use of the root-unblock surface factors through the named source route and then through the named seal route. Any challenged coordinate outside those rows is therefore blocked only at the point where it tries to serve as a converse route, reuse route, or seal shortcut. Since the accepted carrier remains the finite packet of Definition 885.1, the obstruction neither proves an ambient impossibility theorem nor imports a host validation principle beyond the displayed $\mathsf{CauchyModulusRefinement}^{\uparrow}$ rows. □

Theorem 885.110 (Cauchy modulus refinement source-window lock)**.** *For every accepted* CauchyModulusRefineme *root read of*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the consumer-visible source rows M_0, M_1, the shared meet row U, and the extraction row V must be exposed before any selected-window row is read. Thus no root read can reach T, W, Q, E, H, C, P, N as a selected-window route until the displayed source-meet-extraction front M_0, M_1, U, V has been presented.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window lock`

Proof. Read the carrier in Definition 885.1. Its accepted packet presents the two source schedules, the shared meet, and the extraction witness as the finite front of the root route.

Apply Definition 885.4. The extraction row V is bound to the displayed meet U after M_0, M_1, so a downstream selected-window read cannot supply a hidden source or a second meet.

Finally apply Definition 885.84. Every accepted root consumer is already required to read M_0, M_1, U, V through the carrier route before any later selector, seal, or structural row is available. Hence the selected-window surface is locked behind the displayed source window. □

Theorem 885.111 (Cauchy modulus refinement selector-stability lock)**.** *For every accepted* CauchyModulusRefinem *root read, the selector route $V \to T \to W \to Q$ is preserved by the displayed* hsame *and* Cont *rows. The selected-window and tolerance-readback ledgers cannot be replaced by a hidden selector, a second window ledger, or a detached continuation route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinement selector stability lock`

Proof. Start with Definition 885.5. It identifies T as the selector-budget row consumed by the selected-window ledger W and the tolerance-readback ledger Q.

Use Definition 885.85. The readback theorem places T, W, Q after the source-meet-extraction route, so the selector is reached from V and not from an independent ledger.

The displayed hsame row H transports the same packet route, and the displayed Cont row C continues that route without changing the selected-window ledger. Therefore the route $V \to T \to W \to Q$ remains the only exported selector path. □

Theorem 885.112 (Cauchy modulus refinement Real-seal root handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion handoff from an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root read factors through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The handoff exports no quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness coordinate as an alternate Real-seal route.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal root handoff`

Proof. Apply Definition 885.7. The Real seal E is reached from the displayed source, meet, extraction, selector-budget, selected-window, and tolerance-readback rows.

Next apply Definition 885.87. It rules out an alternate route through quotient streams, selected limits, host real equality, hidden selectors, second meet rows, or ambient completeness coordinates at the root boundary.

Finally use Definition 885.56. Every Real or Completion consumer is exhausted by the displayed route, so the handoff factors through $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ and exports no additional Real-seal coordinate. □

Theorem 885.113 (Cauchy modulus refinement root window exhaustion)**.** *Every accepted* $\mathsf{CauchyModulusRefinem}$ $\mathsf{Real}^{\uparrow}$ *or Completion selected-window consumer is equivalent, as a finite BEDC route, to first reading the source and meet rows* M_0, M_1, U, V*, then the selector and selected-window rows* T, W, Q, E*, and then the structural rows* H, C, P, N*. Conversely, any displayed route of this form determines the same* $\mathsf{Real}^{\uparrow}$ *or Completion selected-window consumer. Quotient streams, selected limits, host real equality, hidden selectors, second meet rows, and ambient completeness have no route coordinate for such a consumer.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root window exhaustion`

Proof. The forward direction starts with the source-window lock, Definition 885.110. It forces every accepted consumer route to expose M_0, M_1, U, V before any selected-window read is available.

Next apply the selector-stability lock, Definition 885.111. The displayed selector path from V through T, W, Q is preserved by the hsame and Cont rows, so no hidden selector or detached selected-window ledger can supply a second route.

Now use the Real-seal root handoff, Definition 885.112. It places E, H, C, P, N after the same source and selector rows for every $\mathsf{Real}^{\uparrow}$ or Completion handoff, and it excludes quotient streams, selected limits, host real equality, second meet rows, and ambient completeness as route coordinates.

For the reverse direction, read an accepted packet in exactly the displayed order. The three locks just cited identify each segment with the only source, selector, selected-window, Real-seal, and structural surface exported by the carrier. Therefore the route determines the same $\mathsf{Real}^{\uparrow}$ or Completion selected-window consumer. □

Theorem 885.114 (Cauchy modulus refinement root-source meet determinacy)**.** *Let two accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root reads agree on the source rows* M_0, M_1*, the shared meet row* U*, and the extraction row* V *of the carrier in Definition 885.1. Then before any selected-window row is read, the two reads determine the same source-meet front. In particular, no later selector, selected-window ledger, Real seal, transport, continuation, package, or local name can introduce a second source-meet-extraction route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root source meet determinacy`

Proof. Read the carrier definition. The agreed rows M_0, M_1, U, V are exactly the displayed source-meet-extraction front of the root packet.

Apply source-window lock, Definition 885.110. It places that front before every selected-window read, so neither root read can use T, W, Q, E, H, C, P, N to choose a different source side.

Now use root window exhaustion, Definition 885.113. Once the front is fixed, every later selected-window or Real-completion route is a continuation of the same finite carrier route. Hence the source-meet front is determined before any selected-window consumer appears. □

Theorem 885.115 (Cauchy modulus refinement window consumer coverage)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *selected-window consumer is exhausted by the finite carrier rows*

$$M_0, M_1, U, V, T, W, Q, H, C, P, N.$$

The consumer receives no quotient stream, selected host limit, hidden selector, second meet row, ambient completeness coordinate, detached tolerance readback, or route outside the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier window consumer coverage`

Proof. Start from the source-window lock. The consumer must read M_0, M_1, U, V before a selected-window route is available.

Apply selector-stability lock, Definition 885.111. It forces the selected-window and tolerance-readback rows to proceed through T, W, Q on the same route.

Finally use root window exhaustion. The structural rows H, C, P, N preserve and name the displayed route, while quotient streams, selected host limits, hidden selectors, second meet rows, ambient completeness coordinates, and detached tolerance readbacks have no carrier coordinate. Thus the selected-window consumer is exhausted by the stated finite rows. □

Theorem 885.116 (Cauchy modulus refinement Real-seal source conservation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion seal read from an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root packet conserves the same source-meet-extraction route* M_0, M_1, U, V *before the Real seal row* E *is exposed. The seal cannot replace that route by a quotient stream, selected limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal source conservation`

Proof. Begin with source-window lock and root-source meet determinacy. They bind every root read to the displayed M_0, M_1, U, V front before any selected-window or seal route can be consumed.

Apply the Real-seal root handoff, Definition 885.112. It places the Real seal row E after the same source, meet, extraction, selector-budget, selected-window, and tolerance-readback rows.

Use root window exhaustion for the final exclusion. Since every Real or Completion selected-window consumer is equivalent to the displayed carrier route, the seal has no coordinate for quotient streams, selected limits, host real equality, hidden selectors, second meet rows, ambient completeness coordinates, or detached completion witnesses. Therefore the same source-meet-extraction route is conserved before E is exposed. □

Theorem 885.117 (Cauchy modulus refinement source-window front totality)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *selected-window,* $\mathsf{Real}^{\uparrow}$*, or Completion consumer factors through the displayed source front*

$$M_0, M_1, U, V$$

before reading any of T, W, Q, E, H, C, P, N*. Conversely, a displayed route with that source front and the existing selector-stability, selected-window, Real-seal, transport, continuation, package, and local-name rows determines the same consumer surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window front totality`

Proof. The forward direction starts with the source-window lock, Definition 885.110. It places M_0, M_1, U, V before any selected-window read is available.

Apply selector-stability lock, Definition 885.111. The selected-window rows T, W, Q must continue from the same extraction row V, so a consumer cannot reach them through a hidden selector or detached window ledger.

For $\mathsf{Real}^{\uparrow}$ and Completion consumers, use the Real-seal root handoff, Definition 885.112, together with Real-seal source conservation, Definition 885.116. These results place E, H, C, P, N after the same source-meet-extraction route and prevent the seal from replacing that route.

For the converse, read the displayed packet in the stated order. Root window exhaustion, Definition 885.113, identifies that route with the full selected-window, $\mathsf{Real}^{\uparrow}$, or Completion consumer surface, so the displayed source front and the existing selector and seal rows determine the same consumer. □

Theorem 885.118 (Cauchy modulus refinement source-window totality)**.** *Every accepted* CauchyModulusRefineme *selected-window,* $\mathsf{Real}^{\uparrow}$*, or Completion consumer is determined by first exposing the source rows, shared meet, and extraction front*

$$M_0, M_1, U, V,$$

then routing through

$$T, W, Q, E, H, C, P, N.$$

Conversely, any displayed route with those rows determines the same selected-window, $\mathsf{Real}^{\uparrow}$*, or Completion consumer. No quotient stream, selected limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness supplies another consumer route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window totality`

Proof. The forward direction begins with source-window lock, Definition 885.110. It forces every accepted consumer to expose M_0, M_1, U, V before a selected-window or seal row can be read.

Apply selector-stability lock, Definition 885.111. The selector, selected-window, and tolerance-readback segment is exactly the displayed route from V through T, W, Q, and the structural rows preserve that route rather than replacing it.

For $\mathsf{Real}^{\uparrow}$ and Completion consumers, use Real-seal root handoff, Definition 885.112, together with Real-seal source conservation, Definition 885.116. These results place E, H, C, P, N after the same source and selector front and rule out quotient streams, selected limits, host real equality, hidden selectors, second meet rows, ambient completeness coordinates, and detached completion witnesses.

For the converse, read any displayed route with rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. Root window exhaustion, Definition 885.113, identifies that route with the complete selected-window, $\mathsf{Real}^{\uparrow}$, or Completion surface. Hence the displayed rows determine the same consumer. □

Theorem 885.119 (Cauchy modulus refinement source-window comparison lock)**.** *Let two accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root reads agree on the source rows* M_0, M_1*, the shared meet row* U*, and the extraction row* V *of Definition 885.1. Then every selected-window,* $\mathsf{Real}^{\uparrow}$*, or Completion consumer comparison between those reads agrees on the source-window front before either comparison reads any of*

$$T, W, Q, E, H, C, P, N.$$

Thus no later selector, selected-window ledger, Real seal, transport row, continuation row, package row, or local name row can distinguish the two reads by introducing another source-window front.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window comparison lock`

Proof. Begin with source-window lock, Definition 885.110. It places M_0, M_1, U, V before every selected-window read, so a comparison cannot inspect T, W, Q, E, H, C, P, N until the source front has already been exposed.

Apply root-source meet determinacy, Definition 885.114. Since the two root reads agree on M_0, M_1, U, V, the displayed source-meet-extraction route is the same for both reads before any later row can be consumed.

Finally use source-window totality, Definition 885.118. Every selected-window, $\mathsf{Real}^{\uparrow}$, or Completion consumer is determined by first exposing the same source front and then reading the remaining displayed rows. Therefore any consumer comparison agrees on that front before it reaches the later packet rows. □

Theorem 885.120 (Cauchy modulus refinement source-window Real-seal pullback)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *or Completion seal read is the pullback of the displayed source-window front* M_0, M_1, U, V *along the selector route* $V \to T \to W \to Q \to E$ *and the structural rows* H, C, P, N*. Conversely, any displayed route with those source, selector, seal, and structural rows determines the same seal consumer. The pullback has no coordinate for quotient streams, selected host limits, host real equality, hidden selectors, second meet rows, or ambient completeness.*

Proof. Start with the source-window lock, Definition 885.110. It exposes M_0, M_1, U, V before any selected-window or seal read, so the Real-seal consumer cannot choose its source side after reaching T, W, Q, E, H, C, P, N.

Use selector-stability lock, Definition 885.111. The selected-window route proceeds from V through T, W, Q, and the displayed hsame and Cont rows preserve that route instead of supplying a hidden selector or detached window ledger.

Now apply the Real-seal root handoff, Definition 885.112. It places every $\mathsf{Real}^{\uparrow}$ or Completion seal read after the same source, selector, seal, transport, continuation, package, and local-name rows, ruling out quotient streams, selected host limits, host real equality, second meet rows, and ambient completeness as alternate seal coordinates.

For the converse, use source-window totality, Definition 885.118. A route that displays $M_0, M_1, U, V, T, W, Q, E,$ already determines the selected-window, $\mathsf{Real}^{\uparrow}$, or Completion consumer surface.

Finally compare any two such reads with source-window comparison lock, Definition 885.119. The comparison agrees on the source front before either read reaches the seal rows, so the displayed source-window front pulls back the Real-seal route and the same displayed route determines the same seal consumer. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window real seal pullback`

Theorem 885.121 (Cauchy modulus refinement Real consumer nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root packet factors through*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The consumer has no route to quotient streams, selected host limits, host real equality, hidden selectors, second meet rows, ambient completeness, or detached completion witnesses outside that displayed BEDC packet.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real consumer nonescape`

↪ variant `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal consumer exhaustion`

Proof. Start with Real-seal root handoff, Definition 885.112. It routes every $\mathsf{Real}^{\uparrow}$ or Completion handoff through the displayed source, meet, extraction, selector, selected-window, tolerance, seal, transport, continuation, package, and local-name rows.

Use Real-seal source conservation, Definition 885.116. The seal preserves the same M_0, M_1, U, V source-meet-extraction route and cannot replace it by a quotient stream, selected limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness.

Now apply root window exhaustion, Definition 885.113. It identifies every accepted $\mathsf{Real}^{\uparrow}$ or Completion selected-window consumer with the displayed finite route and rules out alternate route coordinates.

Finally apply window consumer coverage, Definition 885.115. The selected-window portion of the consumer is exhausted by the carrier rows and the structural BHist, hsame, Cont, Pkg, and NameCert surface. Hence the Real or Completion consumer cannot escape the displayed finite packet. □

Theorem 885.122 (Cauchy modulus source-window Real-consumer lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer comparison of accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root reads first exposes the source-window rows*

$$M_0, M_1, U, V$$

and only then routes through

$$T, W, Q, E, H, C, P, N.$$

Consequently the comparison cannot distinguish two accepted reads by quotient streams, selected host limits, second meet rows, hidden selectors, or detached completion witnesses outside the displayed source-window, selector, Real-seal, transport, continuation, package, and local naming rows.

Proof. First expose the source front. By source-window front totality, Definition 885.117, every accepted selected-window, $\mathsf{Real}^{\uparrow}$, or Completion consumer begins with the displayed M_0, M_1, U, V rows.

Now compare two accepted reads. Source-window comparison lock, Definition 885.119, forces any consumer comparison to agree on that front before reading T, W, Q, E, H, C, P, N.

Finally apply Real consumer nonescape, Definition 885.121. Every $\mathsf{Real}^{\uparrow}$ or Completion route remains inside $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so quotient streams, selected host limits, second meet rows, hidden selectors, and detached completion witnesses cannot supply a separate distinction. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window real consumer lock`

Theorem 885.123 (Cauchy modulus refinement L10 completion pullback)**.** *Every Phase Real and Completion consumer using an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

pulls back first along the source-window front M_0, M_1, U, V*, then along the selector and seal route* T, W, Q, E*, and finally along the structural rows* H, C, P, N*. No quotient stream, selected host limit, ambient completeness witness, detached completion witness, or host real equality supplies another L10 completion route.*

Proof. Begin with source-window lock, Definition 885.110. It exposes M_0, M_1, U, V before any selected-window or completion-facing row is available.

Next apply selector-stability lock, Definition 885.111, and Real-seal root handoff, Definition 885.112. These place the selector, selected-window, tolerance-readback, and Real-seal route T, W, Q, E after the same source-window front.

Use Real-seal source conservation, Definition 885.116. The seal preserves the displayed source route and cannot replace it by a quotient stream, selected host limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness.

Finally combine root window exhaustion, Definition 885.113, with Real consumer nonescape, Definition 885.121. The structural rows H, C, P, N preserve and name the same route, so every Phase Real and Completion consumer pulls back along the stated source, selector, seal, and structural rows. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 completion pullback`

Theorem 885.124 (Cauchy modulus refinement completion-route pullback exactness)**.** *Every Phase Real and Completion consumer route of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet factors through the source window*

$$M_0, M_1, U, V,$$

the selector and seal rows

$$T, W, Q, E,$$

and the structural rows

$$H, C, P, N.$$

Conversely, any displayed route with those source-window, selector, seal, transport, continuation, package, and local naming rows determines the same Phase Real or Completion consumer. No

quotient stream, selected host limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness supplies another consumer route.

Proof. First build the source-window front. By source-window front totality, Definition 885.117, every selected-window, $\mathsf{Real}^{\uparrow}$, or Completion consumer begins with the displayed M_0, M_1, U, V rows before reading the later selector, seal, or structural rows.

Next pull the Real and Completion seal route back along that same front. The source-window Real-seal pullback theorem, Definition 885.120, identifies the route through T, W, Q, E, H, C, P, N as the pullback of the displayed source-window front, with no coordinate for quotient streams, selected host limits, host real equality, hidden selectors, second meet rows, or ambient completeness.

Finally lock the consumer comparison. The source-window Real-consumer lock, Definition 885.122, forces every Phase Real and Completion comparison to expose the same source window before reading T, W, Q, E, H, C, P, N. Hence the forward factorization is exact, and any displayed route with those rows determines the same consumer rather than a second route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier completion route pullback exactness`

Corollary 885.125 (Cauchy modulus refinement source-window strict obstruction)**.** *Let*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root packet. Any displayed refusal, separation, or countermodel row used against a source-window converse or consumer-reuse route for R is a strict local obstruction: it blocks only the attempted route that either reads one of T, W, Q, E, H, C, P, N before the source-meet-extraction front M_0, M_1, U, V, or sends a* $\mathsf{Real}^{\uparrow}$ *or Completion consumer through a quotient stream, selected host limit, host real equality, hidden selector, second meet row, ambient completeness coordinate, or detached completion witness outside the displayed packet. It does not state a global impossibility for Cauchy modulus refinement, a host theorem about all real completions, or a hidden external validation of any unlisted coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier source window strict obstruction`

Proof. Apply the source-window lock, Definition 885.110. It places the source rows M_0, M_1, the shared meet row U, and the extraction row V before any selected-window, seal, transport, continuation, package, or local-name row can be consumed. Hence a refusal or separation that concerns premature access to T, W, Q, E, H, C, P, N can block only that premature selected-window route; it cannot be read as a statement about every possible Cauchy-modulus construction.

Now apply Real consumer nonescape, Definition 885.121. Every $\mathsf{Real}^{\uparrow}$ or Completion consumer of the accepted packet factors through exactly

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The same theorem excludes quotient streams, selected host limits, host real equality, hidden selectors, second meet rows, ambient completeness, and detached completion witnesses as consumer coordinates outside that packet. Therefore a consumer-facing obstruction can reject only those named off-route reuse attempts and cannot certify an unlisted host route.

Combining the two facts gives the strict scope. The first theorem fixes where the source-window route begins, and the second theorem exhausts every Real or Completion use by the displayed finite

route. No cited row supplies a quantifier over all host completions, an ambient theorem about all real objects, or an external validation channel. Thus the obstruction is local to the named converse or reuse route in the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet. □

Theorem 885.126 (Cauchy modulus refinement root carrier threshold admission)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the root carrier admits the rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ *only after the shared meet row* U *and the selector-budget row* T *are displayed on the same route. In particular, the Real-completion root threshold surface may read* V, W, Q, E, H, C, P, N *only as descendants of the source schedules* M_0, M_1*, the shared meet* U*, and the selector budget* T *already shown in Definition 885.1.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRootCarrierThresholdAdmission`

Proof. Project the carrier of Definition 885.1. Its finite packet lists the two source schedules, the shared meet, the extraction witness, the selector budget, the selected-window ledger, the tolerance-readback ledger, the Real seal, and the structural rows in one BHist tuple.

Apply the local NameCert obligations of Definition 885.2. Carrier admission is attached to the same tuple, while the source exactness and shared-window clauses require V to be read after M_0, M_1, U and require W, Q to be read through T.

Use source-meet exactness, selector-budget totality, and window-tolerance exactness, Definition 885.21, Definition 885.26, and Definition 885.22. These identify the root threshold route as the displayed chain through U and T before any Real-facing seal read can consume E. □

Theorem 885.127 (Cauchy modulus refinement root classifier transport exactness)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, componentwise* hsame *transport preserves the same source-meet-selector-window-readback route through* H *and* C*. A transported classifier read of* R *is therefore still a read of* $M_0, M_1, U, V, T, W, Q, E$ *in the displayed order, authenticated by the same package row* P *and the same local* $\mathsf{NameCert}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root classifier transport exactness`

Proof. Begin with classifier transport from Definition 885.16. The row H stores componentwise hsame transports for the carrier coordinates, and C stores the displayed continuation route between them.

Next bind the source portion. Source-meet exactness, Definition 885.21, keeps the witness V on the route from M_0, M_1 through U, so transport cannot replace the shared meet by a second source row.

Finally bind the selected-window and Real-facing portion. Selector-budget totality, window-tolerance exactness, and Real seal factorization, Definition 885.26, Definition 885.22, and Definition 885.23, keep T, W, Q, E on the same readback chain. The package and local name rows then certify that the transported classifier is still the displayed root route. □

Theorem 885.128 (Cauchy modulus refinement root ledger non-escape totality)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root ledger receives only the displayed rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

It receives no quotient stream, selected limit, host real equality, hidden selector, second meet row, or ambient completeness datum outside those carrier rows, and the completion-facing ledger is total exactly because every accepted consumer read is already accounted for by that displayed route.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root ledger nonescape totality`

Proof. Read the carrier and the Real seal factorization theorem, Definition 885.1 and Definition 885.23. The seal row E is reached only after the shared meet, extraction witness, selector budget, windows, tolerance rows, and regular readbacks have been displayed.

Apply the Real-consumer coverage theorem, Definition 885.56. It states that every Real or Completion consumer reads only the selector-budget, window, readback, and seal route together with the structural rows.

Finally apply non-escape, Definition 885.10. Since quotient streams, selected limits, host real equality, hidden selectors, second meet rows, and ambient completeness coordinates are absent from the carrier, the displayed ledger exhausts the root consumer surface. □

Theorem 885.129 (Cauchy modulus refinement root route determinacy)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the root read has the unique displayed route

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

For any selected window read S, readback read B, seal read A, and endpoint read Z on that root route, the route package records the carried history rows

$$\mathsf{UnaryHistory}(M_0), \mathsf{UnaryHistory}(M_1), \mathsf{UnaryHistory}(U), \mathsf{UnaryHistory}(V), \mathsf{UnaryHistory}(T), \mathsf{UnaryHistory}(W), \mathsf{UnaryH}$$

the carrier and selected-read continuation links

$$\mathsf{Cont}(M_0, M_1, U), \quad \mathsf{Cont}(U, V, T), \quad \mathsf{Cont}(T, W, Q), \quad \mathsf{Cont}(Q, E, H), \quad \mathsf{Cont}(T, W, S), \quad \mathsf{Cont}(S, Q, B), \quad \mathsf{Cont}(B,$$

the package rows $\mathsf{Pkg}^{\mathrm{sig}}(\Omega, P, \pi)$ *and* $\mathsf{Pkg}^{\mathrm{sig}}(\Omega, Z, \pi)$ *for the same probe bundle* Ω *and package* π*, and the terminal classifier row* $\mathsf{hsame}(H, N)$*.*

The rows H, C, P, N respectively carry the componentwise hsame *transports, displayed* Cont *route, local* Pkg *provenance, and local* NameCert *authentication for that route. No accepted root read may substitute a second meet, hidden selector, quotient stream, host real equality, or ambient completeness row for any coordinate of the displayed route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root route determinacy`

Proof. First read the root carrier threshold admission and classifier transport exactness theorems, Definition 885.126 and Definition 885.127. They place the accepted carrier and its componentwise hsame, Cont, Pkg, and NameCert rows on the same displayed finite packet, and they expose the carrier histories for $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$.

Next bind the source and selected-window parts of the route. Source-meet exactness and selector-budget totality, Definition 885.21 and Definition 885.26, force every accepted read from M_0, M_1 through U, V, T before any selected finite-window row can be read. The selected-read links

$\mathsf{Cont}(T, W, S)$ and $\mathsf{Cont}(S, Q, B)$ then keep the selected window and readback rows as descendants of the selector budget and displayed window.

Use window-tolerance exactness and Real seal factorization, Definition 885.22 and Definition 885.23. These give the readback-to-seal and seal-to-endpoint replay links, so the rows S, B, A, Z inherit UnaryHistory closure and the endpoint package row stays over the same probe bundle and package as the carrier provenance row.

Finally use root ledger non-escape, Definition 885.128. The seal and endpoint are reached only after W, Q, and the ledger contains no second meet, hidden selector, quotient stream, host real equality, or ambient completeness coordinate. Hence the displayed route is the complete root route, with both package rows and the terminal classifier row still attached to the same finite packet. □

Theorem 885.130 (Cauchy modulus refinement shared-tail selector seal coherence)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any selected-window read of R consumed by a $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *terminal route, a* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *shared-schedule route, or a* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *seal route uses the same displayed meet U, selector budget T, finite window W, readback row Q, and* $\mathsf{Real}^{\uparrow}$ *seal E. Thus the adjacent L10 consumer routes cannot mediate a different seal through a hidden selector, second meet, quotient stream, host equality, selected limit, choice datum, or ambient completeness row.*

Proof. First fix the root route. By root route determinacy and root ledger non-escape, Definition 885.129 and Definition 885.128, every accepted selected-window read of R is already on the displayed chain through U, T, W, Q, E and has no extra carrier coordinate available for a second seal.

Next attach the three adjacent consumers. The terminal Real route of Definition 646.3, the shared-schedule inheritance route of Definition 435.45, and the witness-ledger seal route of Definition 977.3 all consume a displayed selected-tail route rather than an ambient completion object.

Therefore each consumer can only inherit the selector, shared tail, and Real-seal rows already displayed by R. Since the root ledger has no hidden selector, second meet, quotient stream, host equality, selected limit, choice datum, or ambient completeness row, all three consumer routes coincide on U, T, W, Q, E. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared tail selector seal coherence`

Theorem 885.131 (Cauchy modulus refinement root budget consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root surface first passes through the selector-budget row T, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q before reading the* $\mathsf{Real}^{\uparrow}$ *seal row E. Thus the root budget consumer surface is exhausted by the displayed route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

with no hidden selector, quotient stream, host real equality, second meet row, selected limit, or ambient completeness datum available to downstream Real or Completion use.

Proof. Begin with the root carrier and classifier transport rows, Definition 885.126 and Definition 885.127. They expose T, W, Q, E as carrier coordinates transported by the same componentwise route and authenticated by the local package and naming rows.

Apply window-tolerance exactness, Definition 885.22. It forces every selected-window and dyadic or regular-rational readback through T, W, Q.

Finally apply Real-consumer coverage, Definition 885.56. Since every Real or Completion consumer is covered by the displayed route before reading E, the selector budget, finite window, and readback rows exhaust the root budget consumer surface. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root budget consumer exhaustion`

Theorem 885.132 (Cauchy modulus refinement root-window Real seal determinacy)**.** *Let*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

and

$$R' = (M_0, M_1, U, V, T, W, Q, E', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packets with the same source schedules, shared meet, selector budget, finite window, and readback rows. Every public* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of the displayed root route classifies the two Real seal rows by* $\mathsf{hsame}(E, E')$.

Proof. Apply root route determinacy, Definition 885.129, to both packets. Since the packets share M_0, M_1, U, T, W, Q, both public reads must pass through the displayed route

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q$$

before any Real seal row is read.

Next use root budget consumer exhaustion, Definition 885.131. A public $\mathsf{Real}^{\uparrow}$ or Completion consumer has no hidden selector, quotient stream, host equality, second meet, selected limit, or ambient completeness row available as an alternate seal source.

It remains only to compare the seal coordinates on the common route. Real seal factorization and classifier transport exactness, Definition 885.23 and Definition 885.127, identify the seal read as the Real-facing coordinate reached from the shared window and readback rows. The transported classifier rows H, C, P, N and H', C', P', N' therefore classify E and E' through the same displayed root route, giving $\mathsf{hsame}(E, E')$. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root window real seal determinacy`

Theorem 885.133 (Cauchy modulus refinement root threshold public exhaustion)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root surface*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

every public $\mathsf{Real}^{\uparrow}$ *or Completion consumer of* R *is exhausted by the displayed route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The consumer first reads the source schedules, shared meet, extraction witness, selector budget, finite-window ledger, tolerance and regular-readback ledger, and Real seal row, and then reads only the componentwise hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows attached to that same route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root threshold public exhaustion`

Proof. Start with root carrier threshold admission and classifier transport exactness, Definition 885.126 and Definition 885.127. They place $M_0, M_1, U, V, T, W, Q, E$ and the structural rows H, C, P, N in one accepted carrier route.

Apply route determinacy, Definition 885.129. It fixes the only public route from the two source schedules through the shared meet, selector budget, finite windows, readbacks, and Real seal, with no second meet, hidden selector, quotient stream, host equality, or ambient completeness row.

Finally use budget consumer exhaustion, Definition 885.131. Since every public Real or Completion consumer must pass through the selector-budget, window, readback, and seal rows of the same route, the displayed packet exhausts the public root threshold surface. □

Theorem 885.134 (Cauchy modulus refinement root threshold seal endpoint separation)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the selector-threshold row T*, finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *remain four distinct public obligations on every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer route. No consumer may collapse* T, W, Q, E *into a single host limit, quotient stream, selected limit, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root threshold seal endpoint separation`

Proof. First fix the public route. Root route determinacy, Definition 885.129, forces every accepted read of R through the displayed chain from the two source schedules to the shared meet, extraction witness, selector threshold, finite window, readback, and Real seal.

Next account for the consumer surface. Root budget consumer exhaustion, Definition 885.131, says that every $\mathsf{Real}^{\uparrow}$ or Completion consumer must pass through T, W, Q, E in that order and has no hidden selector, quotient stream, host equality, selected limit, second meet, or ambient completeness coordinate available as an alternate endpoint.

Now separate the seal endpoint from the window and readback rows. Root-window Real seal determinacy, Definition 885.132, identifies the public seal reached from shared T, W, Q data as the displayed row E, not as a replacement for those rows.

Finally apply root threshold public exhaustion, Definition 885.133. Since the complete public route still displays T, W, Q, E, H, C, P, N, a consumer that collapses the four endpoint obligations would omit carrier coordinates required by the exhausted surface. The four rows therefore remain distinct on every public $\mathsf{Real}^{\uparrow}$ or Completion route. □

Theorem 885.135 (Cauchy modulus refinement four-face noncollapse)**.** *For every accepted* $\mathsf{CauchyModulusRefinem}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the selector threshold T*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *are four separate public obligations on every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer route. None of the four rows may be replaced by a hidden selector, quotient stream, host equality, selected limit, second meet, or ambient completeness coordinate.*

Proof. First fix the public route by root route determinacy, Definition 885.129. Any accepted read of R must pass through the displayed chain from M_0, M_1 to U, V, T, W, Q, E and then through the structural rows H, C, P, N.

Next exhaust the consumer side by root budget consumer exhaustion, Definition 885.131. A $\mathsf{Real}^{\uparrow}$ or Completion consumer has no off-route selector, quotient stream, host equality, selected limit, second meet, or ambient completeness row from which to rebuild the endpoint.

Now identify the seal endpoint using root-window Real seal determinacy, Definition 885.132. The seal reached after the shared threshold, window, and readback rows is the displayed row E; it is not a coordinate that absorbs T, W, or Q.

Finally apply threshold seal endpoint separation, Definition 885.134. Since that separation keeps T, W, Q, E as distinct endpoint obligations on every public route, the four public faces cannot collapse into one hidden completion coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier four face nocollapse`

Theorem 885.136 (Cauchy modulus refinement open-phase four-object consumption)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root packet that reaches the root threshold route consumes the same four finite faces named by the* $\mathsf{StreamName}^{\uparrow}$ *open-phase sufficiency surface: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. The consumed packet also carries exactly the displayed support rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

The Cauchy root rows are therefore read through

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N,$$

where W is the finite window, Q contains the dyadic tolerance and regular-rational readback, E is the Real seal, and H, C, P, N are the componentwise hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows. No quotient stream, host sequence equality, selected limit, ambient completeness, metric-completeness datum, or standard bridge row is exported by this consumer surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier open phase four object consumption`

Proof. First use the StreamName open-phase sufficiency and Real-completion exit certificate, Definition 126.451 and Definition 126.448. They name the four finite exit faces as the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, and RealUp seal, together with the displayed BHist, hsame, Cont, Pkg, and NameCert support rows.

Next force the accepted Cauchy packet through the root threshold route. Root budget consumer exhaustion and root threshold public exhaustion, Definition 885.131 and Definition 885.133, say that every public $\mathsf{Real}^{\uparrow}$ or Completion consumer reads T, W, Q, E, H, C, P, N on the displayed root route and has no alternate selector, quotient stream, host equality, selected limit, or ambient completeness coordinate.

It remains to rule out a second Real seal source. Root-window Real seal determinacy, Definition 885.132, identifies any public seal reached from the shared window and readback rows with the displayed E row. Thus the Cauchy consumer consumes exactly the four-object open-phase exit packet and its displayed support rows, and exports none of the excluded quotient, host, completion, metric, or bridge data. □

Theorem 885.137 (Cauchy modulus refinement root budget stability)**.** *For every accepted* CauchyModulusRefinem *root surface*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the selector-budget row T is stable under the shared meet row U and under classifier transport through H and C. Any public read of the budget row is therefore the same displayed read obtained after M_0, M_1 meet at U, after the extraction witness V is fixed, and before the selected-window, tolerance-readback, and Real-seal rows W, Q, E are consumed.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier root budget stability`

Proof. Start from root carrier threshold admission, Definition 885.126. It admits T only as a row of the same finite packet that already displays M_0, M_1, U, V.

Use classifier transport exactness and route determinacy, Definition 885.127 and Definition 885.129. These keep the transported read on the componentwise route through H, C, P, N and do not allow T to detach from the shared meet.

Finally apply budget consumer exhaustion and public exhaustion, Definition 885.131 and Definition 885.133. Since every Real-facing consumer reaches W, Q, E only after T, the selector budget remains stable at the displayed meet-threshold handoff. □

Theorem 885.138 (Cauchy modulus refinement meet-threshold nonescape)**.** *After an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root route reaches the meet-threshold surface*

$$M_0, M_1, U, V, T,$$

no public consumer can read a second meet, quotient stream, host equality, selected limit, hidden selector, or ambient completeness datum from that route. All later reads are confined to the selected-window row W, tolerance and regular-readback row Q, Real seal row E, and structural rows H, C, P, N of the same packet.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier meet threshold nonescape`

Proof. Read the deterministic route of Definition 885.129. The meet-threshold route has the displayed source schedules, shared meet, extraction witness, and selector budget, and no alternate meet coordinate.

Next use root ledger non-escape and budget consumer exhaustion, Definition 885.128 and Definition 885.131. They account for every Real or Completion read through W, Q, E, H, C, P, N after the budget row.

The public exhaustion theorem, Definition 885.133, closes the surface. Since the displayed route already exhausts public consumption, the absent second meet, quotient stream, host equality, selected limit, hidden selector, and ambient completeness rows cannot be recovered downstream. □

Theorem 885.139 (Cauchy modulus refinement Real seal handoff)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *root packet whose structural rows include a displayed endpoint read Z with* $\mathsf{Cont}(H, C, Z)$ *and* $\mathsf{Pkg}^{\mathrm{sig}}(B, Z, P)$ *over the same package row P presents the completion boundary as a* SemanticNameCert *endpoint carrier. The source predicate is the accepted packet together with the componentwise* hsame *transport to Z, the displayed pattern rows are the route*

$$M_0, M_1 \to U \to V \to T \to W \to Q \to E \to H \to C \to Z,$$

and the ledger row is $\mathsf{UnaryHistory}(Z)$ *together with the package signature. The handoff names the completion endpoint carried by the displayed continuation and package rows; it does not prove Real completeness, quotient equality, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real seal handoff`

Proof. Begin with root route determinacy and root budget stability, Definition 885.129 and Definition 885.137. They place the accepted packet on the single displayed route from the source schedules through the shared meet, extraction witness, selector budget, finite window, readback rows, and Real seal.

Next apply Real seal factorization, Definition 885.23. The seal row E is reached only after W and Q, and the componentwise hsame row carries the same endpoint read Z rather than a second seal source.

Now use root budget consumer exhaustion and public exhaustion, Definition 885.131 and Definition 885.133. Every Real or Completion consumer is confined to W, Q, E, H, C, P, N after the budget row, so a completion endpoint can be read only through the displayed continuation and package rows.

Finally attach the provenance rows. The assumptions $\mathsf{Cont}(H, C, Z)$ and $\mathsf{Pkg}^{\mathrm{sig}}(B, Z, P)$ over the same package row, together with $\mathsf{UnaryHistory}(Z)$, are exactly the endpoint carrier data required for the $\mathsf{SemanticNameCert}$ handoff. Since the packet supplies no ambient completeness coordinate, the result is a handoff certificate naming the completion boundary, not a proof that the Real seal is complete. □

Theorem 885.140 (Cauchy modulus refinement diagonal route pullback)**.** *For every accepted finite* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

whose source rows name a $\mathsf{StreamName}^{\uparrow}$ *window* W*, a* $\mathsf{RegSeqRat}$ *readback row* Q*, and a* $\mathsf{Real}^{\uparrow}$ *seal row* E*, every diagonal-limit consumer route compatible with chapter 646 pulls back to the same finite dyadic threshold* T *and the same shared-window ledger* W, Q *used by* R*. The pulled-back route is a* BHist *route transported by* hsame *through* H*, replayed by* Cont *through* C*, and registered by the same* Pkg *and local* $\mathsf{NameCert}$ *rows* P, N*; it introduces no host quotient, no choice principle, and no extra completion witness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier diagonal route pullback`

Proof. Begin with the Real seal handoff, Definition 885.139. It places every Real or Completion consumer of R after the displayed finite route $M_0, M_1, U, V, T, W, Q, E$ and identifies W, Q, E as the StreamName window, regular readback, and Real seal surface read by the consumer.

Next apply root-window Real seal determinacy, Definition 885.132. If a diagonal-limit-compatible consumer reads the same window and readback rows, then its terminal Real seal is classified through the same root route, so changing only the downstream diagonal presentation cannot produce a second seal or a different threshold.

Finally use L10 handoff sufficiency, Definition 885.144. The rows $U, T, W, Q, E, H, C, P, N$ already suffice for the finite handoff to $\mathsf{Real}^{\uparrow}$, and the nonescape exclusions in that handoff leave no coordinate for a host quotient, selected limit, choice principle, or additional completion witness. Hence every diagonal-limit consumer route pulls back to the same dyadic threshold and shared-window ledger carried by the original refinement packet. □

Theorem 885.141 (Cauchy modulus refinement open-phase exit spine)**.** *Every accepted* CauchyModulusRefinement *root packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

whose public route reaches the displayed rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ *factors through the Phase Real and Completion spine*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The $\mathsf{StreamName}^{\uparrow}$ *surface supplies finite window observation,* $\mathsf{DyadicRatCore}^{\uparrow}$ *supplies dyadic readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *supplies the regular Cauchy handoff, and* $\mathsf{Real}^{\uparrow}$ *supplies the seal row. No Completion consumer may read quotient stream equality, selected limits, host real equality, or ambient completeness before this spine is displayed.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier open phase exit spine`

Proof. First fix the root packet by route determinacy, Definition 885.129. It places the public read on the displayed route from M_0, M_1 through U, V, T, W, Q, E and keeps H, C, P, N as the transport, continuation, provenance, and local naming rows.

Next apply root budget consumer exhaustion, Definition 885.131. This forces every Real or Completion consumer through the selector budget, finite window, dyadic and regular readback, and Real seal rows before any downstream use.

Finally use the Real seal handoff, Definition 885.139. The sibling chapters chapter 126, chapter 291, chapter 301, and chapter 38 name exactly the finite-window, dyadic-readback, regular-handoff, and Real-seal surfaces. Since the handoff theorem excludes quotient equality, selected limits, host real equality, and ambient completeness, the displayed spine is the required exit route. □

Theorem 885.142 (Cauchy modulus refinement root-threshold open-phase handoff)**.** *Every accepted root-threshold route displaying*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

hands off to the Phase Real and Completion surface exactly through the rows T, W, Q, E*. These rows are read as the root threshold budget,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance with* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal. The remaining rows* H, C, P, N *provide transport, replay, provenance, and local naming for that same handoff, while the source rows* M_0, M_1, U, V *remain the root route that supplies the threshold.*

Proof. First consume the open-phase surface. The four-object consumption theorem and open-phase exit spine, Definition 885.136 and Definition 885.141, identify W, Q, E as the StreamName window, dyadic and regular readback, and Real seal, with T as the threshold row that feeds the window.

Next exhaust the root side. Root budget consumer exhaustion and root threshold public exhaustion, Definition 885.131 and Definition 885.133, force every public Real or Completion read of the accepted route through T, W, Q, E, H, C, P, N and leave no second meet, hidden selector, quotient stream, host equality, selected limit, or ambient completeness coordinate.

Finally apply root-window Real seal determinacy and L10 handoff sufficiency, Definition 885.132 and Definition 885.144. They determine the seal from the same threshold, window, and readback rows, then package H, C, P, N as transport, replay, provenance, and local naming for the Phase Real and Completion handoff. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRootThresholdOpenPhaseHandoff`

Theorem 885.143 (Cauchy modulus refinement window-meet seal unicity)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or Completion consumer that reads the shared meet row* U*, selector budget* T*, finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the same displayed route*

$$M_0, M_1 \longrightarrow U \longrightarrow V \longrightarrow T \longrightarrow W \longrightarrow Q \longrightarrow E.$$

Across the L10 sibling surfaces, no second meet, hidden selector, quotient stream, host real equality, selected limit, or ambient completeness row determines a different seal handoff.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier window meet seal unicity`

Proof. First fix the route. Root route determinacy, Definition 885.129, sends every accepted read of the displayed packet through $M_0, M_1, U, V, T, W, Q, E$ and leaves H, C, P, N only as the transport, continuation, provenance, and local naming rows for that route.

Next compare the seal on the common window and readback rows. Root-window Real seal determinacy, Definition 885.132, says that two accepted packets sharing U, T, W, Q classify their public Real seal rows through the same root route. Thus a consumer that has already read those rows cannot obtain a different seal by changing only the handoff surface.

Finally use the sibling handoff and exclusion rows. The Real seal handoff theorem, Definition 885.139, places the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads on the same displayed chain, while meet-threshold nonescape, Definition 885.138, excludes second meets, hidden selectors, quotient streams, host real equality, selected limits, and ambient completeness rows. Hence the shared meet, window, readback, and seal route uniquely determines the Real seal handoff across those L10 surfaces. □

Theorem 885.144 (Cauchy modulus refinement L10 handoff sufficiency)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or Completion consumer that reads the shared meet row* U*, selector budget* T*, finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the displayed L10 handoff*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The rows U, T, W, Q, E*, together with the transport, continuation, provenance, and local naming rows* H, C, P, N*, are sufficient for this handoff. They do not supply a second meet, hidden selector, quotient stream, host real equality, selected limit, or ambient completeness row, and they do not prove completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 handoff sufficiency`

Proof. Root route determinacy, Definition 885.129, fixes the accepted packet route through the displayed meet, extraction, budget, selected-window, readback, and Real-seal rows.

Root budget consumer exhaustion, Definition 885.131, accounts for every downstream read after T. The Real seal handoff theorem, Definition 885.139, then identifies the sibling surfaces as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply window-meet seal unicity, Definition 885.143. It says that a consumer already reading U, T, W, Q, E has no alternate seal route through a second meet, hidden selector, quotient stream, host real equality, selected limit, or ambient completeness row. Therefore the displayed L10 rows are sufficient for the handoff alone, without asserting completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 885.145 (Cauchy modulus refinement finite-observation-budget selector handoff)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

whose public route reaches the displayed rows $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$*, if the selected finite-window route is consumed by a* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, then the consumer first factors through the finite selector path*

$$B \to S \to W \to D \to R \to E$$

before the L10 spine

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

is read. The handoff exposes no quotient stream, host real equality, countable choice, hidden selector, selected host limit, ambient completeness row, or completeness proof for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier finite observation budget selector handoff`

Proof. Start with the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ route. Root route determinacy, Definition 885.129, places every public read on $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, and root budget consumer exhaustion, Definition 885.131, confines the finite-window consumer to the selected budget and window rows.

Next attach the finite selector packet. Finite-observation-budget selector route determinacy, Definition 782.7, identifies the selector path $B \to S \to W \to D \to R \to E$ for every accepted selected-window read. The selector Real-seal handoff, Definition 782.4, carries that same finite path to the Real-facing seal row without importing a host limit.

Finally combine the two exits. The selector diagonal consumer boundary, Definition 782.10, keeps diagonal reads on the finite selector path, while L10 handoff sufficiency, Definition 885.144, reads the resulting window through $\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$. The exclusion clauses in the Cauchy and selector handoffs leave no coordinate for a quotient stream, host real equality, countable choice, hidden selector, selected host limit, ambient completeness row, or Real completeness proof. □

Theorem 885.146 (Cauchy modulus refinement selector-spine route completeness)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *public route*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

whose selected finite-window surface is consumed by $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*, the complete public route factors first through the finite selector path*

$$B \to S \to W \to D \to R \to E$$

and then through the L10 exit spine

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Every read on the route is confined to the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by the Cauchy packet and by the finite selector packet. The factorization exposes no quotient stream, host real equality, countable choice, hidden selector, selected limit, ambient completeness row, or Real completeness proof.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector spine route completeness`

Proof. First form the finite selector handoff. The Cauchy finite-observation-budget selector handoff, Definition 885.145, places every accepted selected-window consumer on $B \to S \to W \to D \to R \to E$ before any Real-facing exit is read. The finite-observation selector route determinacy and Real-seal handoff theorems, Definition 782.7 and Definition 782.4, identify that path as the displayed selector path and keep its seal attached to the same finite packet.

Next attach the L10 spine. L10 handoff sufficiency, Definition 885.144, sends the selected window, dyadic readback, regular readback, and Real seal through

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The selector diagonal consumer boundary, Definition 782.10, keeps diagonal reads on the finite selector path while that spine is consumed.

Finally seal the route against alternate exits. Window-meet seal unicity, Definition 885.143, says that a consumer already reading the shared meet, selector budget, finite window, readback, and Real seal rows has no second seal route through a hidden selector or ambient completion coordinate. Thus the composed public route is complete through the selector path and the L10 spine, and all reads remain on the displayed BHist, hsame, Cont, Pkg, and NameCert rows with none of the excluded quotient, host equality, selected-limit, countable-choice, ambient completeness, or Real-completeness data. □

Theorem 885.147 (Cauchy modulus refinement shared-window seal transitivity)**.** *Let*

$$R_i = (M_0, M_1, U, V_i, T, W, Q, E_i, H_i, C_i, P_i, N_i) \qquad (i = 0, 1, 2)$$

be accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packets sharing the source schedules, shared meet row, selector budget, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and displayed transport/provenance route. If the adjacent Real seal rows classify by* $\mathsf{hsame}(E_0, E_1)$ *and* $\mathsf{hsame}(E_1, E_2)$*, then* E_0 *and* E_2 *classify through the same public* $\mathsf{Real}^{\uparrow}$ *and Completion consumer surface.*

The shared-window comparison introduces no second meet, hidden selector, quotient stream, selected limit, host real equality, or ambient completeness row.

Proof. First fix the public route for all three packets. Apply Definition 885.129; the common rows M_0, M_1, U, T, W, Q force every accepted read through the displayed chain from the source schedules to the Real seal row.

Next pin the seal read on the shared window. By Definition 885.132, any two packets with the displayed common source, meet, budget, window, and readback rows classify their Real seal rows through the same root route. The adjacent hypotheses therefore identify the first pair and the second pair on one public consumer surface.

Now use Definition 885.139 and Definition 885.143. These theorems place the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and Completion reads on the same window-meet handoff, so the two adjacent classifications compose to the first and third seal rows on that same surface.

Finally apply Definition 885.138. It excludes second meets, hidden selectors, quotient streams, selected limits, host real equality, and ambient completeness. With no alternate seal source available, the first and third seal rows classify through the same public Real and Completion consumer boundary. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier shared window seal transitivity`

Theorem 885.148 (Cauchy modulus refinement selector-spine terminal pullback)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *public route*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

whose selected finite-window surface is consumed by $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*, every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing pullback surface factors through the finite selector path*

$$B \to S \to W \to D \to R \to E$$

and then through the L10 spine

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

There is no terminal pullback route that bypasses this displayed selector spine through quotient streams, host real equality, selected limits, countable choice, ambient completeness, or a separate proof of Real completeness.

Proof. First apply selector-spine route completeness, Definition 885.146. It places the accepted public route on the finite selector path followed by the L10 spine, and it confines all reads to the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Next use the finite-observation-budget selector handoff, Definition 885.145. It supplies the selector path $B \to S \to W \to D \to R \to E$ before any Real-facing exit is read.

Finally apply L10 handoff sufficiency and window-meet seal unicity, Definition 885.144 and Definition 885.143. These theorems keep the Real and Completion consumer on the displayed L10 rows and exclude any second meet, hidden selector, quotient stream, host equality, selected limit, countable-choice row, ambient completeness row, or independent Real-completeness witness. Hence the selector spine is the only terminal pullback surface available to such consumers. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier selector spine terminal pullback`

Theorem 885.149 (Cauchy modulus refinement shared-window completion seal coherence)**.** *Let*

$$R_i = (M_0, M_1, U, V_i, T, W, Q, E_i, H_i, C_i, P_i, N_i) \qquad (i = 0, 1, 2)$$

be accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packets sharing the source schedules, shared meet row, selector budget, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and displayed transport/provenance route. If the adjacent shared-window seal classifications identify* E_0 *with* E_1 *and* E_1 *with* E_2*, then their composed Completion-facing classification is coherent only on the displayed consumer surface*

$$B \to S \to W \to D \to R \to E \qquad \textit{followed by} \qquad \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The composition does not create a quotient-stream route, host real equality, selected limit, countable-choice row, ambient completeness row, or independent Real-completeness witness.

Proof. Begin with shared-window seal transitivity, Definition 885.147. It composes the adjacent seal classifications for the first and third packets on the same public $\mathsf{Real}^{\uparrow}$ and Completion consumer surface.

Next identify that surface with the selector-spine route. Apply selector-spine route completeness, Definition 885.146, and the finite-observation-budget selector handoff, Definition 885.145. Together they force the shared-window consumer first through the selector path and then through the L10 spine.

Finally use window-meet seal unicity, Definition 885.143. It prevents a second meet, hidden selector, or alternate completion coordinate from carrying the composed classification. Therefore the adjacent shared-window seal classifications compose coherently only on the displayed selector-spine consumer surface. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementSharedWindowCompletionSealCoherer`

Theorem 885.150 (Cauchy modulus refinement L10 row exactness)**.** *For every accepted* CauchyModulusRefinemen *route whose selected finite-window consumer reaches* $\mathsf{Real}^{\uparrow}$ *or Completion, the exact public row inventory is the selector path*

$$B \to S \to W \to D \to R \to E$$

together with the L10 spine

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

and the displayed transport, continuation, provenance, and local naming rows H, C, P, N*. This inventory excludes quotient streams, host real equality, selected limits, countable choice, ambient completeness, and Real completeness as available rows of the handoff.*

Proof. First read the finite selector side from Definition 885.145. It supplies exactly the displayed selector path for the selected finite-window consumer and lists the excluded host and choice-like coordinates.

Next read the Real-facing side from Definition 885.144. The rows $U, T, W, Q, E, H, C, P, N$ are sufficient for the handoff to the L10 spine, while no row for quotient streams, host equality, selected limits, ambient completeness, or Real completeness is present.

Finally combine these with selector-spine route completeness and shared-window seal transitivity, Definition 885.146 and Definition 885.147. The former identifies the whole public route as selector path followed by L10 spine, and the latter composes adjacent seal classifications without changing the displayed consumer surface. Window-meet seal unicity then rules out any extra row hidden between the selector path and the Real seal. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 row exactness`

Theorem 885.151 (Cauchy modulus refinement Real-window budget pullback)**.** *Every accepted* CauchyModulusRefinement$^{\uparrow}$ *route whose threshold rows feed an accepted* RealWindowBudget$^{\uparrow}$ *packet pulls back to the same finite request, the same* StreamName$^{\uparrow}$ *window rows, the same* DyadicRatCore$^{\uparrow}$ *readback, the same* RegSeqRat$^{\uparrow}$ *handoff, the same* Real$^{\uparrow}$ *seal, the same* CauchyDiagonalBudget$^{\uparrow}$ *selector row, and the same* RealObservationBudget$^{\uparrow}$ *disclosure rows. The pulled-back consumer surface is exactly the displayed L10 inventory together with* H, C, P, N*; it introduces no quotient stream, host real equality, selected limit, ambient completeness witness, or independent Real-completeness row.*

Proof. First expose the refinement side. Apply root threshold public exhaustion, Definition 885.133, and the Real seal handoff, Definition 885.139. These force the accepted refinement route through the displayed threshold, finite-window, readback, handoff, seal, transport, continuation, provenance, and local naming rows before any Real-facing consumer can read it.

Next pull the downstream diagonal and budget faces back to that route. The diagonal route pullback, Definition 885.140, keeps the consumer on the same dyadic threshold and shared-window ledger, while Definition 1164.5 identifies the RealWindowBudget$^{\uparrow}$ packet as precisely the finite request, window, dyadic readback, regular handoff, Real seal, selector, disclosure, and structural rows.

Finally apply L10 row exactness, Definition 885.150. It identifies the common consumer surface with the selector path followed by the L10 spine and removes quotient streams, host real equality, selected limits, ambient completeness, and Real completeness from the available row inventory. Hence the Real-window budget packet consumes exactly the same finite refinement route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier real window budget pullback`

Theorem 885.152 (Cauchy modulus refinement Real-window budget terminality)**.** *Let an accepted* CauchyModulusRefinement$^{\uparrow}$ *route have threshold rows that feed an accepted* RealWindowBudget$^{\uparrow}$ *packet. Every* Real$^{\uparrow}$ *or Completion-facing consumer of that route factors through the same finite request,* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *readback,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal,* CauchyDiagonalBudget$^{\uparrow}$ *selector,* RealObservationBudget$^{\uparrow}$ *disclosure, and support packet* H, C, P, N *displayed in Definition 885.151. Conversely, that packet supplies all consumer-visible rows of the accepted route.*

Thus no such consumer can read a quotient stream, host real equality, selected limit, countable-choice row, ambient completeness datum, independent Real-completeness witness, or second selector or disclosure route.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRealWindowBudgetTerminality`

Proof. First expose the root route. Root threshold public exhaustion and the Real seal handoff, Definition 885.133 and Definition 885.139, place every accepted refinement route on the public threshold, finite request, window, dyadic readback, regular handoff, Real seal, and support rows before a Real-facing or Completion-facing consumer can read it.

Next apply the Real-window budget pullback, Definition 885.151. It identifies the accepted RealWindowBudget$^{\uparrow}$ packet with the same finite request, window, readback, handoff, seal, selector, disclosure, and support packet. Hence every consumer of the refinement route factors through that packet.

For the converse, use L10 row exactness together with the selector-spine terminal pullback, Definition 885.150 and Definition 885.148. The first theorem says that the displayed packet is exactly the L10 consumer inventory with H, C, P, N, while the second makes that selector-spine packet terminal among $\mathsf{Real}^{\uparrow}$ and Completion-facing pullbacks. Therefore the packet supplies all consumer-visible rows.

The same two terminal facts give non-escape. There is no coordinate in the packet for quotient streams, host real equality, selected limits, countable choice, ambient completeness, an independent Real-completeness witness, or a second selector or disclosure route, so no accepted downstream consumer can recover one. □

Theorem 885.153 (Cauchy modulus refinement Real-window terminal consumer)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *route whose threshold rows are accepted by a* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, the terminal consumer-visible surface for any* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*, or Completion-facing reader is exactly the displayed packet consisting of the finite request,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector,* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure, and support rows* H, C, P, N*. Every such consumer factors through that packet, and conversely that packet supplies the full consumer-visible route.*

No additional quotient stream, host real equality, selected limit, countable-choice row, ambient completeness witness, independent Real-completeness row, second selector, or second disclosure row is part of the terminal consumer surface.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementRealWindowTerminalConsumer`

Proof. First use the Real-window budget pullback, Definition 885.151. It places every accepted budget-facing read on the same finite request, window, dyadic readback, regular handoff, Real seal, selector, disclosure, and support rows.

Next apply Real-window budget terminality, Definition 885.152. This turns the pullback packet into the terminal route for $\mathsf{Real}^{\uparrow}$ and Completion-facing consumers, so no downstream reader can detach from the displayed finite packet.

Finally combine selector-spine terminal pullback with L10 row exactness, Definition 885.148 and Definition 885.150. These identify the converse direction: the displayed packet is already the full selector path followed by the L10 spine with H, C, P, N. Since the excluded rows are absent from that exact inventory, the same packet is the whole terminal consumer surface. □

Theorem 885.154 (Cauchy modulus refinement bridged-route surface)**.** *Every standard bridge consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *refinement packet factors through the same public route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The route first reads the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ *meet, the* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *witness, the* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *selector budget, the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. It then uses only the primitive* BHist, hsame, Cont, Pkg*, and* NameCert *rows together with the sibling dependency lattice recorded by the packet. No bridge consumer receives a hidden selector, quotient stream, selected host limit, host real equality, second meet row, side-channel completion object, or ambient completeness coordinate outside that route.*

Proof. Begin with the root-unblock package of Definition 885.40. It assembles the local naming scope, the source-locked selector budget, and the diagonal-consumer boundary into one public entry route for downstream Real-completion consumers.

Next apply the diagonal-consumer boundary and the Real-completeness handoff, Definition 885.39 and Definition 885.49. These force every standard bridge read to pass through the shared meet, extraction witness, selector budget, selected window, dyadic tolerance, regular readback, and Real seal before it can consume the completion-facing surface.

The root readback stability theorem, Definition 885.51, keeps the selected window and tolerance readbacks on the same W, Q, H, C, P, N route under the componentwise transport and continuation rows. The real-consumer coverage theorem, Definition 885.56, then exhausts the Real and Completion consumers by that displayed route.

Finally use the primitive scope and sibling dependency lattice already carried by the chapter. They identify the only admissible bridge coordinates as the primitive BHist, hsame, Cont, Pkg, NameCert rows and the named CauchyModulusMeet$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, RegularCauchySelectorBudget$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ siblings. Since non-escape excludes the listed hidden and ambient coordinates, every standard bridge consumer factors through the stated public route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier bridged route surface`

Theorem 885.155 (Cauchy modulus refinement bridged-route determinacy)**.** *Let two standard bridge consumers accept the same* CauchyModulusRefinement$^{\uparrow}$ *packet and expose the same displayed route rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Then both public bridge reads factor through the same sibling route through CauchyModulusMeet$^{\uparrow}$*,* CauchyModulusExtraction$^{\uparrow}$*,* RegularCauchySelectorBudget$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. The two reads determine the same Real and Completion handoff surface under the componentwise* hsame *rows of* H*.*

Proof. Apply the bridged-route surface of Definition 885.154 to each consumer. Because the exposed rows are the same displayed $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, both reads enter the same meet, extraction witness, selector budget, selected window, dyadic tolerance, regular readback, and Real seal route.

Root readback stability, Definition 885.51, fixes the selected window and tolerance readbacks under the componentwise hsame rows. Thus a second consumer cannot replace W, Q, E by another selected stream, dyadic ledger, or readback while still exposing the same route.

Real-consumer coverage, Definition 885.56, exhausts the Real and Completion-facing reads by the same finite route. The primitive sibling lattice then leaves no hidden selector, quotient stream, second meet row, host limit, or ambient completeness coordinate able to separate the two consumers.

The resulting Real and Completion handoff surfaces are therefore the same public surface up to the displayed hsame transport in H. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementBridgedRouteDeterminacy`

Theorem 885.156 (Cauchy modulus refinement bridge consumer route exhaustion)**.** *Every standard bridge consumer of an accepted* CauchyModulusRefinement$^{\uparrow}$ *packet whose incoming route is supplied by the* RegularCauchyTailMeet$^{\uparrow}$ *root budget-seal package must first pass through the displayed TailMeet rows, then the finite observation-budget and limiter surfaces, and then the* CauchyModulusRefinement$^{\uparrow}$ *route*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *seal read is exported. The exported bridge surface contains no quotient stream, selected limit, host real equality, hidden selector, independent completion carrier, or ambient completeness principle outside that route.*

Proof. Apply the downstream budget-seal package of Definition 560.96. It supplies the incoming $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ route through the displayed TailMeet rows, the finite observation budget, and the limiter surfaces before any downstream seal consumer can read the packet.

The root seal exactness theorem, Definition 560.97, fixes that incoming route as the budgeted seal interface. Thus a bridge consumer entering from the TailMeet package has no separate Real or Completion channel before the named budget rows have been consumed.

Now apply the local bridged-route surface and determinacy theorems, Definition 885.154 and Definition 885.155. They force the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet onto the displayed $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ route and make that route the unique public refinement handoff for standard bridge consumers.

Finally use the non-escape clauses in the bridge-surface proof. Those clauses exclude hidden selectors, quotient streams, selected host limits, host real equalities, second meet rows, side-channel completion objects, and ambient completeness coordinates. Combining the incoming TailMeet budget route with the local refinement route therefore exhausts every standard bridge consumer before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read is exported. □

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier bridge consumer route exhaustion`

Theorem 885.157 (Cauchy modulus refinement TailMeet budget route uniqueness)**.** *Let two standard bridge consumers enter an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet through the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *budget-seal package and expose the same displayed refinement rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Then the two exported $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *seal handoff routes are the same public route under the componentwise* hsame *rows of* H*. The comparison reads only the shared TailMeet budget-seal rows and the displayed refinement rows; it has no coordinate for a hidden selector, quotient stream, selected host limit, host real equality, ambient completeness principle, or side-channel completion carrier.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementTailMeetBudgetRouteUniqueness`

Proof. Apply Definition 560.96. Since both consumers enter through the same $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ budget-seal package, their incoming routes first share the displayed TailMeet rows, finite observation budget, and limiter surface before the refinement packet is read.

Apply Definition 560.97. The shared incoming package is the exact budgeted seal interface, so neither consumer can insert an independent $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal route before the named TailMeet budget rows.

Now use Definition 885.155. The two consumers expose the same $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ rows, so the refinement-side public bridge read determines the same Real and Completion handoff surface under the componentwise hsame rows of H.

Finally apply Definition 885.156. Every accepted consumer entering from the TailMeet budget package is already exhausted by the TailMeet route followed by the displayed refinement route. The excluded hidden selector, quotient stream, selected host limit, host real equality, ambient completeness, and side-channel completion coordinates therefore cannot distinguish the two exported seal handoffs. □

Theorem 885.158 (Cauchy modulus refinement TailMeet Real-seal normal form)**.** *Every bridge consumer entering an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet through the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *budget rows and exposing the refinement rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N$$

has a unique normal-form route

TailMeet budget $\to$ *finite observation budget* $\to$ *limiter* $\to$ $\mathsf{CauchyModulusMeet}^{\uparrow}$ $\to$ $\mathsf{CauchyModulusExtraction}^{\uparrow}$ $\to$ R

The exported seal surface contains no quotient stream, selected host limit, hidden selector, second meet, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier tailmeet real seal normal form`

Proof. First use the bridge-consumer route exhaustion theorem, Definition 885.156. It forces every consumer entering from the $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ root package through the TailMeet budget rows, finite observation budget, limiter, and then the displayed refinement route before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ seal read is exported.

Next apply the TailMeet budget route uniqueness theorem, Definition 885.157. Once two consumers expose the same $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ rows, the exported Real and Completion seal handoff route is fixed under the componentwise hsame rows of H.

The sibling TailMeet root budget package and root seal exactness theorem, Definition 560.96 and Definition 560.97, identify the incoming TailMeet path with the finite budget-seal interface. The local bridged-route surface then supplies the meet, extraction, selector-budget, stream-name, dyadic-rational, and regular-sequence part of the route.

Finally use the non-escape clauses already recorded by the bridge-surface proof. Since the absent coordinates are not exported by the TailMeet package or by the refinement packet, the displayed route is the unique normal form for Real and Completion seal consumers. □

Corollary 885.159 (Cauchy modulus refinement bridged-route consumer determinacy)**.** *Let* $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet, and fix the displayed bridge rows*

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

Every accepted standard bridge consumer use of R *that reports a route selection, a normal-form readback, or an exactness decision for the refinement handoff reports the one named by the chapter: the route through* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the* $\mathsf{Real}^{\uparrow}$ *seal, stabilized by the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No hidden selector, quotient stream, second meet row, duplicate classifier source, selected host limit, host real equality, side-channel completion object, or ambient completeness coordinate can supply a different accepted route, normal form, or exactness decision.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier bridged route consumer determinacy`

Proof. Apply the bridged-route surface theorem, Definition 885.154. It gives the consumer-facing inventory of an accepted bridge read: before any standard bridge consumer can use the refinement handoff, the read factors through the fixed public route

$$M_0, M_1, U, V, T, W, Q, E, H, C, P, N.$$

The same theorem identifies the sibling surfaces read along that route and excludes hidden selectors, quotient streams, selected host limits, host real equalities, second meet rows, side-channel completion objects, and ambient completeness coordinates.

Now take two accepted standard bridge consumer uses of the fixed packet rows. One may be the chapter's displayed use and the other any consumer-facing use that claims a route selection, normal-form readback, or exactness decision. Since the rows are fixed, both expose the same $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ data. By bridged-route determinacy, Definition 885.155, the two public bridge reads factor through the same sibling route and determine the same Real and Completion handoff surface under the componentwise hsame rows of H.

A reported route selection is therefore the displayed factorization route. A reported normal-form readback is the same ordered read through the meet, extraction witness, selector budget, finite window, dyadic tolerance, regular readback, and Real seal rows. A reported exactness decision is the decision that the proposed consumer read is accepted exactly when it factors through those fixed rows and the structural rows H, C, P, N.

It remains only to exclude duplicate sources for the same decision. Any such source is either one of the displayed rows, in which case it is already identified by the fixed route and the determinacy theorem, or it is an unlisted coordinate. The surface theorem rejects the unlisted case. Hence every accepted consumer use picks the same finite route, normal form, and exactness decision named by the chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet is generated from displayed modulus comparison, meet, extraction, selector-budget, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 885.1, Definition 885.2, Definition 885.11, Definition 885.12, Definition 885.13, Definition 885.16, Definition 885.21, Definition 885.22, Definition 885.23, Definition 885.24, Definition 885.25, Definition 885.26, Definition 885.27, Definition 885.28, Definition 885.29, Definition 885.30, Definition 885.31, Definition 885.32, Definition 885.33, Definition 885.34, Definition 885.35, Definition 885.36, Definition 885.37, Definition 885.10, and the bridge witness Definition 885.154 bind the displayed modulus comparison, meet, extraction, selector-budget, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows into one scoped Real and completion obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, hidden selectors, second meet rows, ambient completeness, Lean verification, or bridge checking for $\mathsf{CauchyModulusRefinement}^{\uparrow}$.

Upgrade path: No further paper-side theory closure is required; remaining work is Lean verification and any future bridge checking for the displayed $\mathsf{CauchyModulusRefinement}^{\uparrow}$ route.

885.1 L10 window lattice pointer

The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ L10 ledger is the shared route consumed by chapter 1180 and chapter 38. Its source side is read through chapter 462, chapter 641, and chapter 678; its finite-window and rational-readback side is read through chapter 126, chapter 291, and chapter 301.

Theorem 885.160 (Cauchy modulus refinement L10 window lattice exactness)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing route through* R *factors first through the source meet, extraction, and selector-budget rows* U, V, T*, then through the four L10 faces*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

and then only through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *support rows* H, C, P, N*. The route exports no quotient stream equality, selected limit, host real equality, hidden selector, second meet row, countable-choice row, bridge aggregation, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completion principle.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementL10WindowLatticeExactness`

Proof. Start with the L10 sibling route of Definition 885.33. It places every Phase Real and Completion consumer after the source comparison, meet, extraction, and selector-budget rows, and orders the four downstream faces as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next pull the consumer back along the finite packet rows. The completion pullback theorem, Definition 885.123, reads the route first through the source-window front M_0, M_1, U, V, then through T, W, Q, E, and finally through the structural rows H, C, P, N. The one-line ledger anchors above identify those rows with the CauchyModulusMeet, CauchyModulusExtraction, RegularCauchySelectorBudget, StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces.

Finally apply L10 row exactness, Definition 885.150. It identifies the public inventory with the selector path, the four L10 faces, and the displayed support rows. Since the accepted packet and those anchors contain no quotient stream equality, selected limit, host real equality, hidden selector, second meet row, countable-choice row, bridge aggregation, ambient $\mathsf{Real}^{\uparrow}$ completeness, or ambient completion principle, no additional Real-facing or Completion-facing lattice route remains. □

Theorem 885.161 (Cauchy modulus refinement L10 root-window pullback exhaustion)**.** *For every Phase Real or Completion consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

the consumer route is exhausted by the following finite pullback spine. It first passes through the root source-window front M_0, M_1, U, V *of Definition 885.113; it then passes through the selector and L10 face rows* T, W, Q, E *governed by Definition 38.732; and it finally passes through the support rows* H, C, P, N *fixed by Definition 885.160 and Definition 885.123. Conversely, any displayed packet with exactly those rows repacks as the same L10 consumer route, with no additional quotient-stream, selected-host-limit, countable-choice, ambient-completeness, metric-completeness, bridge-aggregation, or hidden-selector row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 root window pullback exhaustion`

Proof. The forward direction begins at the accepted packet R. Root-window exhaustion forces the source side through M_0, M_1, U, V, so a consumer cannot enter by a second meet row, an unstated extraction row, or an ambient completion premise.

After that source window has been fixed, the L10 four-face exit criterion determines the only Real-facing exit surface. The selector row and the StreamName, DyadicRatCore, RegSeqRat, and Real rows are therefore exactly T, W, Q, E, and there is no room for quotient streams, selected host limits, countable choice, host $\mathsf{Real}^{\uparrow}$ equality, or a hidden selector.

The completion pullback and the window-lattice exactness theorem then identify the remaining support as H, C, P, N. These rows account for the displayed BHist, hsame, Cont, Pkg, and NameCert

support data, so metric-completeness rows and bridge aggregation are not part of the consumer route.

Conversely, suppose a displayed packet carries exactly $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. Read the first four rows as the root source-window front, the next four as the selector and four-face L10 exit, and the last four as the support ledger. The cited exactness and pullback results repack these finite rows into the same L10 route, and the excluded rows have no constructor in the displayed inventory. □

Theorem 885.162 (Cauchy modulus refinement L10 finite-request route monotonicity)**.** *Let*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$$

and

$$R' = (M_0, M_1, U, V, T', W', Q', E', H, C, P, N)$$

be accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packets with the same displayed source-meet, extraction,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. If the finite request displayed by* T' *refines the finite request displayed by* T *along the selector row over the same* $M_0, M_1, U, V, H, C, P, N$, *then the L10 route for* R *is included in the L10 route for* R' *on the same support: the* $\mathsf{StreamName}^{\uparrow}$ *window row* W, *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q, *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal route* E *are preserved by inclusion through the displayed* $T \leq T'$, $W \leq W'$, $Q \leq Q'$, *and* $E \leq E'$ *rows. No quotient stream, selected limit, countable-choice row, ambient completeness row, or host equality supplies a second route between the two requests.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 finite request route monotonicity`

↪ variant `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementL10FiniteRequestRouteMonotonicity`

Proof. First fix the common source side. L10 window lattice exactness, Definition 885.160, makes every Real-facing or Completion-facing read pass through M_0, M_1, U, V and then through the same displayed support rows H, C, P, N.

Now compare the selector rows. Root-window pullback exhaustion, Definition 885.161, says that the consumer route has no source other than the finite spine $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. Replacing T by a displayed refinement T' over the same source rows can therefore only enlarge the finite request already present in that spine.

Window-tolerance exactness, Definition 885.22, transports this selector inclusion to the StreamName window and DyadicRatCore tolerance rows. The same displayed history and classifier support then carries the regular-rational readback row, so the route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ is preserved by inclusion.

Finally apply Real seal factorization, Definition 885.23. The $\mathsf{Real}^{\uparrow}$ seal is downstream of the selected window and rational readback route, so the seal route is included along $E \leq E'$ and cannot be replaced by a quotient stream, selected limit, countable-choice row, ambient completeness row, or host equality. □

Theorem 885.163 (Cauchy modulus refinement L10 face-order rigidity)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing consumer of* R *must read the L10 faces, after the source rows* U, V, T*, in the displayed dependency order*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

A proposed consumer that swaps the stream schedule, dyadic tolerance, regular readback, or $\mathsf{Real}^{\uparrow}$ *seal face is not an accepted pullback of the refinement packet. In particular, such a consumer cannot be completed by quotient streams, host real equality, selected limits, countable choice, ambient completeness, metric-completeness rows, bridge aggregation, or hidden selectors.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementL10FaceOrderRigidity`

Proof. The window-lattice exactness theorem, Definition 885.160, fixes the only Real-facing and Completion-facing route through R: the source meet, extraction, and selector-budget rows precede the L10 faces, and those faces are displayed in the order $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Root-window pullback exhaustion, Definition 885.161, identifies every accepted consumer with that finite pullback spine. The root-window theorem Definition 885.113 supplies the same source-window front, while Definition 38.732 supplies the four-face exit criterion. Thus the stream schedule must be available before dyadic tolerance, dyadic tolerance before regular readback, and regular readback before the $\mathsf{Real}^{\uparrow}$ seal.

Suppose a proposed consumer swaps one of these four faces. Then its entry point is not the source-window front followed by the four-face exit specified above, so it is not one of the pullbacks exhausted by the accepted packet rows. The excluded resources have no constructor in the packet inventory: quotient streams and host real equality would bypass the StreamName and Real seal faces; selected limits and countable choice would bypass the dyadic and regular-readback rows; ambient completeness, metric-completeness rows, bridge aggregation, and hidden selectors would add rows absent from $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. □

Theorem 885.164 (Cauchy modulus refinement L10 support row minimality)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

let F *be an accepted* $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing consumer of* R *that already factors through the source rows* M_0, M_1, U, V *and the L10 face rows* T, W, Q, E*. Then the remaining support of* F *is exactly* H, C, P, N*:* H *supplies componentwise* hsame *transport along the displayed histories,* C *supplies the displayed* Cont *replay,* P *supplies the* Pkg *provenance row, and* N *supplies the local* NameCert *row. Removing* H*,* C*,* P*, or* N *loses respectively the history-transport, continuity-replay, package-source, or local-name obligation. Adding any further support coordinate gives an off-packet route, hence is excluded by L10 window lattice exactness and root-window pullback exhaustion.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementL10SupportRowMinimality`

Proof. Fix the accepted packet R and the consumer F. The hypotheses have already placed F through the source rows and through the L10 face rows, so the only undecided part of the route is its support ledger.

By L10 window lattice exactness, Definition 885.160, every such consumer reaches support only through the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Root-window pullback exhaustion, Definition 885.161, then identifies the whole consumer with the finite spine $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$ and leaves no second support coordinate outside that spine.

Project the four support rows. The row H is the componentwise history and hsame transport needed to carry source equality through the route; without it the consumer has no displayed transport witness. The row C is the continuity replay row; without it the route has no displayed Cont replay for the completion-facing read. The row P records the Pkg provenance of the source data; without it the consumer cannot cite the package source. The row N is the local NameCert row; without it the consumer has no certified name for the packet instance.

Finally apply face-order rigidity, Definition 885.163, to keep the support ledger downstream of the fixed source and face order. Dropping any one of H, C, P, N removes the corresponding obligation just projected, while adding a further coordinate would be a route not present in the exact finite pullback inventory. Hence the accepted support ledger is precisely H, C, P, N. □

Lemma 885.165 (Cauchy modulus refinement L10 window lattice downstream coverage)**.** *Let c be an accepted local consumer of an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet with displayed L10 window lattice*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N).$$

Then the consumer-facing read made by c is exhausted by the finite alternatives, ledger rows, and coverage rows already displayed in R. More precisely, the source-facing alternatives are only M_0, M_1, U, V; the selector and L10 ledger rows are only T, W, Q, E in the downstream order

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow};$$

and the coverage rows are only the displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support carried by H, C, P, N. Consequently a quotient stream equality, selected host limit, host real equality, hidden selector, second meet row, countable-choice row, metric-completeness row, bridge aggregation, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completion principle has no accepted local consumer coordinate in the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 window lattice downstream coverage`

Proof. Fix the accepted local consumer c and the displayed packet R. By Definition 885.160, any $\mathsf{Real}^{\uparrow}$-facing or Completion-facing route through R must first read the source meet, extraction, and selector-budget rows, then the four L10 faces, and then the support rows H, C, P, N. Thus the consumer-facing route of c has no starting coordinate outside the displayed source, face, and support surfaces.

Apply Definition 885.161 to identify the whole route with the finite pullback spine. The first segment is the root source-window front M_0, M_1, U, V; the second segment is the selector and L10 face block T, W, Q, E; and the terminal support segment is H, C, P, N. The converse part of that theorem says that a displayed packet with exactly these rows repacks as the same L10 consumer route, so the listed alternatives are not merely sufficient rows but the complete accepted inventory for c.

The three parts of the read are therefore forced. A source-facing demand can only land on M_0, M_1, U, V. A downstream analytic demand can only pass through the selector row and the four displayed faces T, W, Q, E in the order fixed by the window lattice. A coverage demand can only use the displayed history, transport, continuation, provenance, and naming rows carried by H, C, P, N.

It remains to inspect the excluded alternatives. Each excluded object named in the statement would require either an additional source coordinate, an alternate L10 face, or an additional support row beyond $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$. The exactness theorem excludes

such additions from every $\mathsf{Real}^{\uparrow}$-facing or Completion-facing route, and the pullback exhaustion theorem excludes them from the repacked finite spine. Hence every accepted local consumer is covered by the displayed finite rows, and every excluded alternative remains outside the accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ packet. □

Theorem 885.166 (Cauchy modulus refinement L10 face-status noncollapse)**.** *For every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet*

$$R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N),$$

a $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing consumer cannot replace the four L10 face rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

by a single aggregate closurestatus, formalstatus, or bridge-status row. The consumer must preserve the separate face-status reads determined by the L10 window lattice, the root-window pullback, the face order, the support-row ledger, and downstream coverage. This theorem only prevents cross-face status aggregation; it does not upgrade the verification axis of any individual face.

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier l10 face status noncollapse`

Proof. First apply L10 window lattice exactness, Definition 885.160. Every accepted $\mathsf{Real}^{\uparrow}$-facing or Completion-facing route through R must pass through the source rows and then through the displayed L10 face surface before it reaches the support rows.

Next apply root-window pullback exhaustion, Definition 885.161. It identifies the consumer route with the finite spine $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, so an aggregate status row outside that spine cannot be a hidden replacement for the face rows.

Face-order rigidity, Definition 885.163, then keeps the downstream read ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. A single aggregate row would erase that order and therefore would not be an accepted pullback of the packet.

Support-row minimality, Definition 885.164, fixes the remaining rows as H, C, P, N. These support rows transport, replay, source, and name the displayed packet; none of them stores a combined closurestatus, formalstatus, or bridge-status value for all four faces.

Finally use downstream coverage, Definition 885.165. Every accepted local consumer is already covered by the displayed source, face, and support rows. Hence the four face-status rows remain separate, and the theorem records only this noncollapse discipline, not a stronger verification claim for any face. □

Theorem 885.167 (Cauchy modulus refinement formal target surface)**.** *Every accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *window-lattice consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed finite route*

$$\mathsf{CauchyModulusMeet}^{\uparrow}, \mathsf{CauchyModulusExtraction}^{\uparrow}, \mathsf{RegularCauchySelectorBudget}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}$$

Equivalently, for a packet $R = (M_0, M_1, U, V, T, W, Q, E, H, C, P, N)$*, the consumer route is exhausted by the modulus-meet rows* M_0, M_1, U*, extraction row* V*, selector-budget row* T*, StreamName window row* W*, DyadicRatCore tolerance row* Q*, RegSeqRat readback and Real seal row* E*, and the support rows* H, C, P, N*. No quotient stream equality, selected host limit, host real equality, hidden selector, second meet row, countable-choice row, metric-completeness row, bridge aggregation, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completion principle supplies an alternate formal target surface.*

Lean4 ✓ `BEDC.Derived.CauchyModulusRefinementUp.CauchyModulusRefinementCarrier formal target surface`

Proof. First use L10 window lattice exactness. It places every Real-facing or Completion-facing read through the source meet, extraction, and selector-budget rows before the four L10 faces and support rows are available.

Next apply root-window pullback exhaustion and finite-request route monotonicity. The pullback theorem identifies the finite spine $M_0, M_1, U, V, T, W, Q, E, H, C, P, N$, while monotonicity says refinements enlarge only the displayed request route and do not create a second source of Real-facing data.

Then use face-order rigidity and support-row minimality. The faces must be read in the order $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the remaining support is exactly H, C, P, N for hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Finally apply downstream coverage. Every accepted local consumer is already covered by those source, face, and support rows, so the excluded quotient, selection, host-equality, hidden-selector, second-meet, choice, completeness, bridge, and ambient-completion alternatives have no coordinate in the formal target surface. □

Chapter 886

A Concrete Naming Certificate for $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$

The $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as a single finite synchronization surface. It joins the seal boundary of chapter 386, the budget selector of chapter 853, and the observation-budget route of chapter 611 before a $\mathsf{Real}^{\uparrow}$ consumer reads a completion-facing seal. The packet has no coordinate for quotient streams, host real equality, selected external limits, countable choice, or ambient completeness.

Definition 886.1 (Cauchy seal-budget synchronizer carrier)**.** *A* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (E, S, B, T, D, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite precision and selected-observation request entering the Real and Completion read;*
2. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row whose* $\mathsf{Real}^{\uparrow}$ *seal is the completion-facing endpoint;*
3. *B is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *or* $\mathsf{RealObservationBudget}^{\uparrow}$ *budget row that exposes the finite selected window, dyadic ledger, regular handoff, and seal boundary;*
4. *T is the* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *or shared-tail schedule row supplying the cofinal tail witness used by the same request;*
5. *D is the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *row naming the selected diagonal tail consumed by the budget and seal;*
6. *L is the joint compatibility row tying D, B, the shared threshold from T, and the seal row S into one finite route;*
7. *H stores componentwise* hsame *transports for $E, S, B, T, D, L, C, P, N$;*
8. *C stores the* Cont *routes by which consumers read the request, selector, budget, shared threshold, compatibility row, seal row, provenance, and local naming row;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}}$ *row.*

Lean4 ✓ BEDC.Derived.CauchySealBudgetSynchronizerUp.CauchySealBudgetSynchronizerTasteGate single carrier alignment

Theorem 886.2 (Cauchy seal-budget synchronizer NameCert obligations)**.** *For the carrier of Definition 886.1, the local* $\mathsf{NameCert}_{\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}}$ *surface consists exactly of carrier admission for* Z*, selector exposure through* D*, budget exposure through* B*, shared-tail exposure through* T*, seal exposure through* S*, joint compatibility through* L*, classifier stability under the componentwise* hsame *rows in* H*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is the displayed tuple $Z = (E, S, B, T, D, L, H, C, P, N)$. The request, seal, budget, tail schedule, selector, compatibility, transport, route, provenance, and naming rows are all explicit coordinates.

Selector, budget, and shared-tail exposure are projections from the same tuple. The selector row D names the diagonal tail, the budget row B supplies either the Cauchy diagonal budget or the observation-budget route, and T supplies the shared tail threshold consumed by the same request.

The seal and compatibility clauses are also finite projections. The row S is not read as an isolated seal; it is tied by L to the selected diagonal row, budget row, and shared threshold. Componentwise transports in H and continuation routes in C can replay these displayed rows, but they cannot introduce a second selector, budget, threshold, or seal. □

Lean4 ✓ BEDC.Derived.CauchySealBudgetSynchronizerUp.CauchySealBudgetSynchronizerTasteGate obligation surface

Theorem 886.3 (Cauchy seal-budget synchronizer non-escape)**.** *Every accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet read by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer factors through both the displayed seal row* S *and a displayed budget row* B *that is supplied by* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *or* $\mathsf{RealObservationBudget}^{\uparrow}$*. If the read uses the tail schedule, it must also pass through the displayed selector* D*, shared threshold* T*, and compatibility row* L*. No accepted read can bypass both the budget row and the selector-threshold compatibility row while still exporting the seal.*

Proof. Start from the carrier inventory. The only seal-facing coordinate is S, and the only budget-facing coordinate is B.

Apply the NameCert obligation theorem. It requires every admitted consumer route to expose D, B, T, L, and S through the displayed transport and continuation rows before the local name can be consumed.

Thus a seal read that omits the budget row has no route to B, and a seal read that omits both the selector row and the shared-threshold compatibility row has no route to D, T, L. Since the carrier contains no quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness theorem, no omitted analytic object can be repacked as a second seal route. □

Theorem 886.4 (Cauchy seal-budget synchronizer shared budget lock)**.** *Every accepted* $\mathsf{CauchySealBudgetSynchron}$ *packet fixes one request row, one seal row, one displayed budget row, one selector row, one shared-tail row, one compatibility row, and one structural family before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer can read the seal. In the notation of Definition 886.1, the locked rows are*

$$E, S, B, T, D, L, H, C, P, N,$$

where B is supplied by chapter 853 or chapter 611, and S is supplied by chapter 386. A consumer route that changes any one of these rows before reading S is outside the local $\mathsf{NameCert}_{\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}}$ *surface.*

Proof. Project the carrier displayed in Definition 886.1. Its ten coordinates are exactly the request, seal, budget, tail schedule, selector, compatibility, componentwise transport, displayed continuation, provenance, and local naming rows.

Apply Definition 886.2. The obligation surface exposes D, B, T, L, and S through the same H, C, P, N structural rows, so the request and the consumer route read one displayed packet rather than separately chosen sibling certificates.

Finally use Definition 886.3. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ read that reaches the seal must factor through both the displayed seal row from chapter 386 and the displayed budget route from chapter 853 or chapter 611. Therefore replacing any locked row destroys the accepted synchronizer route. □

Lean4 ✓ `BEDC.Derived.CauchySealBudgetSynchronizerUp.CauchySealBudgetSynchronizerUp StdBridge`

Theorem 886.5 (Cauchy seal-budget synchronizer tail-selector compatibility)**.** *In every accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet, the tail schedule row T and selector row D are consumed together through the compatibility row L. Hence a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer cannot combine the seal row S with an alternate selector or an alternate tail source while remaining inside the carrier*

$$Z = (E, S, B, T, D, L, H, C, P, N).$$

Proof. Start with the compatibility coordinate in Definition 886.1. The row L is the only displayed place where D, B, the shared threshold from T, and the seal row S are tied into one finite route.

Apply Definition 886.2. The local $\mathsf{NameCert}$ obligations require selector exposure, shared-tail exposure, budget exposure, seal exposure, and joint compatibility, and they replay these rows through the same componentwise hsame, Cont, Pkg, and local naming surface.

Use Definition 886.3. Since a seal read must pass through the displayed budget row and through the selector-threshold compatibility row, a consumer that substitutes another selector or tail source no longer reads the synchronizer packet. The sibling seal, diagonal-budget, and observation-budget chapters provide the displayed rows, but the synchronizer admits them only through the shared L row. □

Theorem 886.6 (Cauchy seal-budget synchronizer real-seal budget exhaustion)**.** *The* $\mathsf{Real}^{\uparrow}$*-facing seal route of an accepted* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *packet is exhausted by*

$$E, S, B, T, D, L, H, C, P, N.$$

It contains the displayed request, seal, budget, tail schedule, selector, compatibility, transport, continuation, provenance, and local naming rows, and it excludes quotient streams, host real equality, selected external limits, countable choice, and ambient completeness as seal-exporting resources.

Proof. Begin with the carrier inventory of Definition 886.1. The $\mathsf{Real}^{\uparrow}$-facing route has exactly the ten displayed rows $E, S, B, T, D, L, H, C, P, N$.

Apply Definition 886.2. These rows account for carrier admission, selector exposure, budget exposure, shared-tail exposure, seal exposure, joint compatibility, componentwise transport, displayed route replay, package provenance, and local naming.

Now apply Definition 886.3. Any $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer must read the seal through the displayed chapter 386 row and through the displayed budget route of chapter 853 or chapter 611.

There is no coordinate in the carrier for quotient streams, host real equality, selected external limits, countable choice, or ambient completeness. Those resources therefore cannot be recovered as hidden seal exporters, and the finite row list exhausts the Real-facing seal route. □

Theorem 886.7 (Cauchy seal-budget synchronizer scoped row grounding)**.** *For an accepted packet*

$$Z = (E, S, B, T, D, L, H, C, P, N),$$

every scoped obligation of $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *is grounded by exactly the displayed request, seal, budget, tail schedule, diagonal selector, compatibility, transport, route, provenance, and* $\mathsf{NameCert}$ *rows. The seal row S is consumed from chapter 386; the budget row B is consumed from chapter 853 or chapter 611; the tail schedule row T is consumed from chapter 681; the selector row D is consumed from chapter 697; and the seal read ends at the* $\mathsf{Real}^{\uparrow}$ *row named in the provenance package. No additional scoped obligation is discharged by quotient streams, host real equality, selected external limits, countable choice, or ambient completeness.*

Proof. Begin with the carrier definition. The packet has exactly the ten displayed coordinates $E, S, B, T, D, L, H, C, P, N$, so a scoped obligation can only be admitted through one of those rows.

Apply the local obligation package in Definition 886.2. Carrier admission gives Z; selector, budget, shared-tail, seal, and compatibility exposure give D, B, T, S, L; classifier stability and route replay give H and C; package provenance and local naming give P and N.

Next apply the shared budget lock and tail-selector compatibility theorems. They force the sibling rows to be consumed together: S comes from the Cauchy limit seal surface, B comes from the diagonal-budget or observation-budget surface, T supplies the tail schedule, D supplies the selected diagonal tail, and L ties these rows to the Real-facing seal read.

Finally project P. Its provenance list names the sibling dependency rows over $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Use the non-escape theorem to close the scope. Since the accepted carrier contains no coordinate for quotient streams, host real equality, selected external limits, countable choice, or ambient completeness, none of those objects can discharge a hidden synchronizer obligation. Thus the scoped obligation surface is exhausted by the displayed rows. □

Falsifiable prediction: If an accepted synchronizer packet is read by a RealCompleteness consumer, the read factors through both a displayed CauchyLimitSealUp row and a displayed CauchyDiagonalBudgetUp or RealObservationBudgetUp row; a seal read bypassing both refutes the packet.

Independence witness: cauchylimitseal, cauchydiagonalbudget, realobservationbudget, tailcofinalityschedule, diagonaltailselector: none is bijective because the synchronizer has one joint compatibility row tying selector, observation budget, shared threshold, and seal rows; each sibling lacks at least one of these rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ packet is generated from a finite precision request, completion-facing seal row, budget row, tail schedule row, diagonal selector row, joint compatibility row, componentwise transports, displayed routes, package provenance, and local naming row.

Theory closure: TheoryGate(NameCert$_{\text{CauchySealBudgetSynchronizer}\uparrow}$, maturePackageClosed)

Scope closed: Definition 886.1, Definition 886.2, Definition 886.3, Definition 886.4, Definition 886.5, Definition 886.6, Definition 886.7, and the checked TasteGate marker bind the synchronizer carrier, finite obligation package, row lock, selector-tail compatibility boundary, Real-facing budget exhaustion, scoped row grounding, and non-escape surface.

Formal verification: FormalStatus(NameCert$_{\text{CauchySealBudgetSynchronizer}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchySealBudgetSynchronizerUp.CauchySealBudgetSynchronizerTasteGate\_single\_carrier\_`

Bridge status: bridgeChecked

Not claimed: This chapter does not close Real completeness, quotient streams, host real equality, selected external limits, countable choice, or ambient completeness.

Upgrade path: No further verification-axis bridge upgrade remains for this finite synchronizer surface.

Chapter 887

A Concrete Naming Certificate for $\mathsf{RealSealComparison}^{\uparrow}$

The $\mathsf{RealSealComparison}^{\uparrow}$ packet is the equality-facing comparison surface for two sealed $\mathsf{Real}^{\uparrow}$ routes in the Real completeness target of chapter 9238. It consumes two Cauchy-limit seal packets from chapter 386, the separated uniqueness row of chapter 427, and the diagonal stability route of chapter 870. The packet records a finite comparison between the two sealed routes; it is not a quotient stream, a host real equality, a selected limit, or an ambient completeness principle.

Falsifiable prediction: If a $\mathsf{RealSealComparison}^{\uparrow}$ route is accepted, it factors through two displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ public route packets and one separated comparison row. A route requiring quotient streams, host real equality, selected limits, or ambient completeness is outside the carrier.

Independence witness: cauchylimitseal, regularlimituniqueness, diagonallimitstability, realobservationbudget, regularcauchycomparison: none is bijective to $\mathsf{RealSealComparison}^{\uparrow}$ because this carrier contains two seal routes and one equality-facing separated comparison between them. CauchyLimitSeal, RealObservationBudget, and DiagonalLimitStability are single-route or representative-to-seal packets; RegularLimitUniqueness supplies the separated row without the two public seal routes; RegularCauchyComparison compares regular sources before the paired seal-comparison package is assembled.

Definition 887.1 (Real seal comparison carrier)**.** *A* $\mathsf{RealSealComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S_0, S_1, U, D_0, D_1, K, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are two public* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route packets, each already exposing its regular source, finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local naming rows;*
2. *U is the separated* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *comparison row between the two* $\mathsf{Real}^{\uparrow}$ *seal outputs;*
3. *D_0 and D_1 are the* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *rows that attach the two seal routes to their displayed diagonal representatives;*
4. *K is the equality-facing comparison verdict read only after S_0, S_1, and U have been displayed;*
5. *H records componentwise* hsame *transport for the two seal packets, the separated comparison row, the diagonal-stability rows, the verdict, route replay, provenance, and local naming;*

6. *C records the displayed* Cont *routes that replay the two seal packets, the separated comparison, the diagonal-stability handoffs, and the verdict;*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RegularLimitUniqueness}^{\uparrow}$, $\mathsf{DiagonalLimitStability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSealComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, a selected host limit, countable choice, trichotomy, or an ambient Real completeness theorem.

Lean4 ✓ `BEDC.Derived.RealSealComparisonUp.RealSealComparisonTasteGate single carrier alignment`

Theorem 887.2 (RealSealComparison NameCert obligations)**.** *For every* $\mathsf{RealSealComparison}^{\uparrow}$ *carrier*

$$Q = (S_0, S_1, U, D_0, D_1, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSealComparison}^{\uparrow}}$ *surface consists of carrier exposure for the two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route packets, exposure of the separated* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *comparison row, exposure of the two* $\mathsf{DiagonalLimitStability}^{\uparrow}$ *handoff rows, equality-facing verdict exposure through K, component-wise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 887.1. The two seal packets, separated comparison row, diagonal-stability rows, verdict row, transports, continuations, provenance, and local name are explicit coordinates of one finite BHist packet.

Read the two seal routes through the Cauchy-limit seal public boundary. Each side must already present its regular source, finite windows, readback, and $\mathsf{Real}^{\uparrow}$ seal before the comparison route can inspect it.

Then read U through the separated uniqueness surface and D_0, D_1 through diagonal-limit stability. These rows make the equality-facing verdict a finite comparison between displayed seal routes, not a host equality between completed reals.

The structural rows H, C, P, N transport, replay, authenticate, and name exactly those displayed rows. Since the carrier contains no quotient, host equality, selected limit, choice, trichotomy, or ambient-completeness coordinate, the local NameCert obligations are exhausted by the listed packet. □

Lean4 ✓ `BEDC.Derived.RealSealComparisonUp.RealSealComparisonCarrier namecert obligations`

Theorem 887.3 (RealSealComparison uniqueness handoff)**.** *Any accepted* $\mathsf{RealSealComparison}^{\uparrow}$ *packet that exports equality of the two sealed* $\mathsf{Real}^{\uparrow}$ *routes must factor the export through the two displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packets S_0, S_1 and the separated comparison row U. The verdict K is determined only after those three rows and the diagonal-stability handoffs D_0, D_1 have been replayed through the displayed* hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Start with the two seal packets. The Cauchy-limit seal boundary forces each $\mathsf{Real}^{\uparrow}$ route to expose its public route packet before a downstream consumer reads the seal.

Next apply the separated uniqueness row. The comparison between the two seal outputs is the row U, so the equality-facing verdict cannot be read before both S_0 and S_1 have reached their displayed seals.

The diagonal-stability rows attach the seal routes to their representatives, while H, C, P, N replay and authenticate the same finite route. Thus the handoff to equality is the displayed two-seal and one-separated-row route, with no independent host equality or selected limit available. □

Theorem 887.4 (RealSealComparison non-escape)**.** *Downstream consumers of an accepted* RealSealComparison$^{\uparrow}$ *packet receive only the finite comparison surface*

$$S_0, S_1, U, D_0, D_1, K, H, C, P, N.$$

They receive no quotient stream, host real equality, selected limit, countable-choice selector, trichotomy row, or ambient completeness theorem.

Proof. Project the accepted carrier. The public comparison surface contains exactly the two seal route packets, separated comparison row, diagonal-stability handoffs, verdict, transports, continuations, provenance, and local naming row.

Use the uniqueness handoff to route any equality-facing consumer through S_0, S_1, U, D_0, D_1 before it reaches K. This makes the displayed finite comparison the only consumer-visible equality boundary.

Finally inspect the excluded data. No coordinate in the carrier stores quotient stream equality, host real equality, a selected limit, countable choice, trichotomy, or ambient completeness. The downstream export therefore cannot escape the finite two-seal comparison surface. □

Closure status for NameCert$_{\text{RealSealComparison}^{\uparrow}}$**.**

Constructive story: A RealSealComparison$^{\uparrow}$ packet is generated from two CauchyLimitSeal$^{\uparrow}$ public route packets, a separated RegularLimitUniqueness$^{\uparrow}$ comparison row, two DiagonalLimitStability$^{\uparrow}$ handoff rows, an equality-facing verdict, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealSealComparison}^{\uparrow}}$, seed)

Scope closed: Definition 887.1, Definition 887.2, Definition 887.3, and Definition 887.4 specify the finite two-seal comparison carrier, local NameCert rows, separated uniqueness handoff, and non-escape boundary.

Formal verification: FormalStatus(NameCert$_{\text{RealSealComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove host real equality, quotient-stream equality, selected-limit existence, ambient Real completeness, Lean verification, or bridge checking for RealSealComparison$^{\uparrow}$.

Upgrade path: Obligation closure requires checked two-seal carrier admission, separated comparison exactness, diagonal-stability replay, equality-verdict determinacy, transport preservation, and consumer non-escape for the same finite packet.

Chapter 888

A Concrete Naming Certificate for $\mathsf{ObserverTraceKernel}^{\uparrow}$

The $\mathsf{ObserverTraceKernel}^{\uparrow}$ packet realizes the observer-history identity vision of chapter 9418 as a finite kernel trace-row surface. It is the row-level companion of chapter 640 and chapter 882: the former exposes observer identity through a finite kernel trace, while the latter orders distinction steps. This chapter binds those two readings into one NameCert packet whose consumer sees only BHist, hsame, Cont, observation, provenance, and no-subject rows.

Definition 888.1 (ObserverTraceKernel carrier)**.** *An* $\mathsf{ObserverTraceKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, T, Q, O, L, C, P, N)$$

with the following displayed rows:

1. *H is the finite source-history ledger, whose entries are* BHist *rows available to the observer-facing consumer;*
2. *T is the kernel trace row recording the ordered finite observation route above H;*
3. *Q is the componentwise* hsame *transport row for source histories, trace rows, observation rows, provenance rows, and the local name;*
4. *O is the finite observation ledger, including the displayed* ProbeBundle *and* SigRel *reads licensed for the same trace;*
5. *L is the no-subject ledger recording that every exported observer read is one of the displayed rows of K;*
6. *C records the displayed* Cont *routes from H through T, O, L to the local naming row;*
7. *P is the* Pkg *provenance over* $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverKernelTrace}^{\uparrow}$, $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, BHist, hsame, Cont, *observation, and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverTraceKernel}^{\uparrow}}$ *row.*

The carrier has no coordinate for an observer-substance type, host equality of observers, hidden owner of observations, or trace object outside the displayed finite kernel rows.

Lean4 ✓ `BEDC.Derived.ObserverTraceKernelUp.ObserverTraceKernelTasteGate single carrier alignment`

Theorem 888.2 (ObserverTraceKernel carrier row)**.** *For the carrier of Definition 888.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverTraceKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (H, T, Q, O, L, C, P, N)$*, kernel trace-row exposure, componentwise* hsame *transport, finite observation readback, displayed* Cont *route replay, no-subject ledger exactness,* Pkg *provenance, and local naming exactness.*

Proof. Carrier admission is projection from the displayed tuple K. The rows H, T, Q, O, L, C, P, N list the source histories, kernel trace, transports, observations, no-subject ledger, continuation routes, provenance, and local name.

The trace-row and observation clauses are read together. The kernel trace T records the finite route over the histories in H, while O records only the licensed observation rows attached to that same route.

Transport and replay are also componentwise. The row Q preserves the displayed coordinates under hsame, and C gives the only permitted reads from source histories to observation and naming rows. Since the tuple has no further coordinate, the listed clauses exhaust the local certificate surface. □

Theorem 888.3 (ObserverTraceKernel no-subject ledger)**.** *Every accepted* $\mathsf{ObserverTraceKernel}^{\uparrow}$ *packet exports observer-history identity only through the finite rows* H, T, Q, O, L, C, P, N*. No consumer can extract an observer-substance carrier, subject identity row, hidden owner of the trace, host equality of observers, or observation truth predicate outside* BHist*,* hsame*,* Cont*, observation,* Pkg*, and* NameCert *rows.*

Proof. Project the accepted carrier and inspect the no-subject ledger L. It records each exported observer-facing read as a row already listed in H, T, Q, O, C, P, N.

Apply Definition 888.2. The local certificate surface admits only kernel trace rows, transports, finite observation rows, route replay, provenance, and local naming.

An observer-substance carrier or hidden owner would require an additional coordinate outside K. Because no such coordinate is present, every accepted consumer remains inside the displayed trace rows and cannot export subject identity or host equality data. □

Theorem 888.4 (ObserverTraceKernel composition boundary)**.** *Every consumer route from an accepted* $\mathsf{ObserverTraceKernel}^{\uparrow}$ *packet to an observer-history identity read factors through the finite composition*

$$H \longmapsto T \longmapsto O \longmapsto L \longmapsto C \longmapsto P \longmapsto N.$$

The route composes the $\mathsf{ObserverKernelTrace}^{\uparrow}$ *trace surface, the* $\mathsf{ObserverDistinctionTrace}^{\uparrow}$ *distinction-step surface, the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *observation surface, and the* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *exclusion surface without adding a new observer carrier, quotient observer identity, ambient time parameter, or host equality of subjects.*

Proof. Begin with the kernel trace sibling chapter 640. It confines observer trace reads to finite BHist, hsame, Cont, probe, signature, provenance, and naming rows.

Next use the distinction-trace sibling chapter 882. It orders distinction-step reads through displayed Hist growth and Cont routes, so the trace row T remains a finite kernel row rather than a detached observer state.

Finally apply the observer-history identity and subject-boundary surfaces. chapter 479 supplies the observation readback, and chapter 615 prevents the route from exporting a subject coordinate. The displayed route through H, T, O, L, C, P, N is therefore the whole consumer boundary. □

The later Lean target in `lean4/BEDC/Derived/ObserverTraceKernelUp.lean` must provide $\mathsf{Nontrivial\ ObserverTraceKernel}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObserverTraceKernel}^{\uparrow}$ instances. The witnesses must distinguish the source-history ledger, kernel trace row, hsame transports, observation ledger, no-subject ledger, displayed Cont routes, provenance, and local name.

Falsifiable prediction: Any accepted ObserverTraceKernelUp consumer route factors through the displayed kernel trace, observation, Cont, provenance, and no-subject rows before reading the local NameCert row.

Independence witness: observerkerneltrace, observerdistinctiontrace, observerhistidentity, observersubjectboundary: the packet records a kernel trace-row composition rather than trace exposure alone, distinction ordering alone, observation identity alone, or subject-boundary exclusion alone.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverTraceKernel}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverTraceKernel}^{\uparrow}$ packet is generated from source-history rows, a finite kernel trace row, componentwise hsame transports, observation rows, a no-subject ledger, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverTraceKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 888.1, Definition 888.2, Definition 888.3, and Definition 888.4 bind the finite kernel trace-row packet for observer-history identity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverTraceKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, a host observer model, quotient observer identity, ambient time, or an observer-substance behind the trace.

Upgrade path: Obligation closure requires checked carrier admission, trace-row exposure, hsame transport, observation readback, Cont route replay, provenance, no-subject exactness, and the stated Nontrivial and FieldFaithful instances.

Chapter 889

A Concrete Naming Certificate for $\mathsf{ObserverLocalityTransport}^{\uparrow}$

The $\mathsf{ObserverLocalityTransport}^{\uparrow}$ packet realizes the transport boundary in the Inter-Hist Locality chain of chapter 9419. It sits after the observer-locality cell surface of chapter 513, the causal-route surface of chapter 478, and the unary-bound surface of chapter 594; it is the route-level sibling read by the cross-Hist causal-rate packet of chapter 876. The packet records the finite path by which a consumer transports locality-cell data through causal rows and maximum-rate rows without entering a global observer frame.

Definition 889.1 (Observer-locality transport carrier)**.** *An* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, D, R, L, H, Q, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row that supplies the two-observer locality cell being transported;*
2. *D is the finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows that records the causal route through which the locality-cell read is moved between the two displayed histories;*
3. *R is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row whose unary bound is read only after the causal routes in D have been exposed;*
4. *L is the no-global-frame ledger, recording that the transport route contains no privileged observer frame, synchronized clock, metric tensor, Lorentz-geometry coordinate, or host real-valued speed row;*
5. *H records componentwise* hsame *transport for the locality cell, causal rows, maximum-rate row, no-global-frame ledger, probe-stability row, continuation routes, provenance row, and local naming row;*
6. *Q records* psame *stability for the probe reads used by the locality-cell row, causal rows, and maximum-rate row;*
7. *C records displayed* Cont *routes by which a consumer reads O, replays each row of D, reads R, checks L, and transports the same packet through H and Q;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverLocalityTransport}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{CrossHistCausalRate}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$ *row.*

The carrier exports only the displayed locality cell, causal rows, maximum-rate row, no-global-frame ledger, transport rows, continuation routes, provenance, and local naming data.

Theorem 889.2 (Observer-locality transport NameCert obligations)**.** *For the carrier of Definition 889.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet $T = (O, D, R, L, H, Q, C, P, N)$;*
2. *locality-cell source exactness, requiring the transported row to be the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row O and not a detached observer-frame coordinate;*
3. *causal-route handoff, requiring every transport route to pass through the displayed* $\mathsf{CrossHistCausal}^{\uparrow}$ *family D before the maximum-rate row R is read;*
4. *transport stability through componentwise* hsame *in H, probe stability in Q, and displayed* Cont *replay in C;*
5. *non-escape, excluding global frames, synchronized clocks, metric tensors, Lorentz geometry, host real-valued speed, and any locality read not routed through $O, D, R, L, H, Q, C, P, N$.*

Proof. Carrier admission is projection from Definition 889.1. The rows $O, D, R, L, H, Q, C, P, N$ display the locality cell, causal family, maximum-rate row, no-global-frame ledger, transports, routes, provenance, and local name.

Locality-cell source exactness reads O first. The packet has no separate observer-frame coordinate, so the transported locality surface is the finite cell row rather than an ambient frame.

Causal-route handoff then reads D before R. Each maximum-rate read is attached to the displayed causal family, so a consumer cannot replace the route witness by a bare unary bound.

Transport stability is componentwise through H, Q, C, and the non-escape claim is the carrier inventory. Since no excluded structure is a coordinate of T, no local certificate projection can export one. □

Theorem 889.3 (Observer-locality transport boundary)**.** *Every public consumer of an accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *packet reads the locality-cell row only through*

$$O, D, R, L, H, Q, C, P, N.$$

In particular, the consumer must pass from the $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row O through the* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows D before it reads the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row R or the cross-Hist causal-rate surface of chapter 876.*

Proof. Project the accepted carrier. The locality-cell source is the row O, and the only displayed route family connecting it to rate data is D.

Use the causal-route handoff in Definition 889.2. It places the $\mathsf{CrossHistCausal}^{\uparrow}$ rows before the maximum-rate row R, so the rate read is not a substitute for the transport route.

Finally route through H, Q, C, P, N and check the no-global-frame ledger L. The consumer remains inside the finite packet and cannot read a global frame, metric tensor, Lorentz geometry, synchronized clock, or host real-valued speed. □

Theorem 889.4 (Observer-locality transport rate pullback)**.** *Every rate-facing consumer of an accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *packet factors through the same finite tuple*

$$O, D, R, L, H, Q, C, P, N.$$

In particular, the consumer must read the $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row* O*, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *route family* D*, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row* R*, and the* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *consumer surface through one transport packet. The packet exports no global frame, synchronized clock, metric tensor, Lorentz geometry, host real-valued speed, or detached rate row.*

Proof. Project the accepted carrier from Definition 889.1. The tuple $O, D, R, L, H, Q, C, P, N$ lists the locality cell, causal family, maximum-rate row, no-global-frame ledger, structural rows, provenance, and local naming row available to a rate-facing consumer.

Apply Definition 889.2. The NameCert surface places locality-cell exactness before causal-route handoff, and it requires every maximum-rate read to pass through the displayed causal family.

Now use Definition 889.3. It keeps the public consumer inside the same tuple and routes any cross-Hist causal-rate read through the locality cell, causal routes, maximum-rate row, and transport rows.

The non-escape clause follows from the ledger L and carrier inventory. Since no listed coordinate is a global frame, synchronized clock, metric tensor, Lorentz geometry, host real-valued speed, or detached rate row, the accepted transport packet cannot export one. □

Theorem 889.5 (Observer-locality transport TasteGate obligation)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ ObserverLocalityTransport}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ ObserverLocalityTransport}^{\uparrow}$ *instances. These instances must preserve the locality-cell row, causal-route family, maximum-rate row, no-global-frame ledger,* hsame *transport,* psame *stability,* Cont *routes, package provenance, and local naming row.*

Proof. The packet is not determined by a single locality cell or a single rate row. It carries a route-level transport witness between the cell and the maximum-rate surface.

Nontriviality rules out a collapsed carrier in which the route witness is indistinguishable from the cell or rate rows. Field faithfulness records preservation of all displayed coordinates $O, D, R, L, H, Q, C, P, N$, so a TasteGate consumer can distinguish the route-level transport row from the sibling surfaces it connects. □

Lean4 ✓ `BEDC.Derived.ObserverLocalityTransportUp.ObserverLocalityTransportTasteGate single carrier alignment`

Theorem 889.6 (Observer-locality transport obligation carrier admission)**.** *For every accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier*

$$T = (O, D, R, L, H, Q, C, P, N),$$

the finite local $\mathsf{NameCert}_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$ *surface admits exactly the displayed transport packet. The admitted rows are the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *source row* O*, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *route family* D*, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *rate row* R*, the no-global-frame ledger* L*, componentwise transport* H*, probe stability* Q*, continuation replay* C*, package provenance* P*, and the local naming row* N*.*

Proof. Project the carrier displayed in Definition 889.1. Its coordinates are precisely $O, D, R, L, H, Q, C, P, N$, and each coordinate already has the named transport role used by the local certificate.

Apply the boundary theorem, Definition 889.3. It says every public consumer reads the locality-cell row only through the same tuple, so carrier admission cannot add an external frame, clock, metric, or detached rate coordinate.

Finally apply the rate pullback theorem, Definition 889.4. The rate-facing surface also factors through the same packet, which makes the admitted carrier the complete finite NameCert row for later promotion. □

Theorem 889.7 (Observer-locality transport obligation causal-route handoff)**.** *For every accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier* $T = (O, D, R, L, H, Q, C, P, N)$*, any consumer that reads the maximum-rate row* R *or the cross-Hist causal-rate surface must first pass through the displayed route family* D *from the locality-cell row* O*. The handoff is transported only by* H*, stabilized only by* Q*, replayed only by* C*, packaged only by* P*, and locally named only by* N*.*

Proof. Begin with the local NameCert obligations of Definition 889.2. Their handoff row places every maximum-rate read after the displayed $\mathsf{CrossHistCausal}^{\uparrow}$ family D.

Use Definition 889.3 to identify the public route. The consumer starts at the locality-cell row O, moves through D, and only then reaches R or a cross-Hist causal-rate read.

The rate pullback theorem, Definition 889.4, keeps this handoff inside T. Thus H, Q, C, P, N preserve and replay the same finite route instead of manufacturing a second causal path or detached rate row. □

Theorem 889.8 (Observer-locality transport route composition determinacy)**.** *For every accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier*

$$T = (O, D, R, L, H, Q, C, P, N),$$

any two public consumers that read the same $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row* O *and the same* $\mathsf{CrossHistCausal}^{\uparrow}$ *family* D *before reading the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row* R *have the same displayed transport route through* H, Q, C, P, N*. No such consumer can splice in a detached causal row, detached rate row, global frame, synchronized clock, metric tensor, Lorentz coordinate, or host real-valued speed.*

Proof. Project the accepted carrier from Definition 889.1. The only displayed route from the locality cell to the rate row is the finite packet $O, D, R, L, H, Q, C, P, N$.

Apply causal-route handoff, Definition 889.7. It forces both consumers to read O, then the same family D, and only then the rate row R.

Use rate pullback, Definition 889.4. Since every rate-facing consumer remains inside the same tuple, the continuation, transport, stability, provenance, and naming rows are exactly H, Q, C, P, N for both reads.

Finally apply non-global-frame exhaustion, Definition 889.9. The excluded coordinates have no carrier row in T, so they cannot be spliced into the displayed route. □

Theorem 889.9 (Observer-locality transport obligation non-global-frame exhaustion)**.** *For every accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier* $T = (O, D, R, L, H, Q, C, P, N)$*, the no-global-frame ledger* L *exhausts every request for a privileged observer frame, synchronized clock, metric tensor, Lorentz-geometry coordinate, host real-valued speed, or locality read not routed through* $O, D, R, L, H, Q, C, P, N$*. Later promotion reads the existing* `BEDC.Derived.ObserverLocalityTransportUp.ObserverLocalityTransport` *marker together with the* $\mathsf{Nontrivial\ ObserverLocalityTransport}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ ObserverLocalityTransport}^{\uparrow}$ *instances, so the AI-origin TasteGate discipline is attached to this same finite carrier rather than to a global frame.*

Proof. Read the ledger coordinate L in the carrier definition. It records exactly the exclusions for global frame, synchronized clock, metric tensor, Lorentz geometry, host real-valued speed, and off-route locality access.

Apply Definition 889.3. Since every public locality read stays inside T, any request for a global frame has no coordinate except the refusal recorded by L.

Apply Definition 889.4 for rate-facing consumers. The rate row is available only after the causal route and never as a host real-valued speed or Lorentz coordinate, so the exhaustion covers both locality and rate interfaces.

The TasteGate marker already checked in this chapter is then read as a single-carrier alignment condition. Nontriviality separates the transport packet from its sibling rows, and field faithfulness preserves all coordinates $O, D, R, L, H, Q, C, P, N$ during promotion. □

Theorem 889.10 (Observer-locality transport public export)**.** *For every accepted* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *carrier*

$$T = (O, D, R, L, H, Q, C, P, N),$$

the public export surface of T is exactly the finite route $O, D, R, L, H, Q, C, P, N$. The row O is exported as the $\mathsf{ObserverLocalityCell}^{\uparrow}$ *source, D as the finite* $\mathsf{CrossHistCausal}^{\uparrow}$ *route family, R as the* $\mathsf{MaxCausalRate}^{\uparrow}$ *rate row, L as the no-global-frame ledger, H as componentwise* hsame *transport, Q as probe stability, C as continuation replay, P as package provenance, and N as the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$ *row. The provenance path runs through* $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{CrossHistCausalRate}^{\uparrow}$, *and the checked single-carrier TasteGate marker* `BEDC.Derived.ObserverLocalityTransportUp.ObserverLocalityTransportTasteGate_single_carri` *no public export coordinate lies outside that route.*

Proof. Apply Definition 889.6. It identifies the accepted carrier with precisely the displayed packet $O, D, R, L, H, Q, C, P, N$ and supplies the local NameCert row for that packet.

Apply Definition 889.7. The export reads the locality cell through O, the route family through D, and the maximum-rate or cross-Hist causal-rate surface only after that handoff.

Apply Definition 889.8. Once the public consumer has read the same O and D before R, the displayed transport through H, Q, C, P, N is determined, so the export cannot splice in a detached route or detached rate row.

Finally apply Definition 889.9. The ledger L exhausts requests for global-frame or host-geometry data, while the checked TasteGate single-carrier marker attaches promotion to the same finite packet. □

Falsifiable prediction: If the transport packet is atomic, a consumer cannot replace the displayed locality-cell row by a MaxCausalRate row without passing through CrossHistCausal transport.

Independence witness: Nearest siblings: observerlocalitycell, crosshistcausal, maxcausalrate, crosshistcausalrate. The transport packet is not bijective to any of them because it carries a route-level transport row between sibling surfaces, while those siblings carry cell, causal, bound, or rate rows without this route witness.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverLocalityTransport}^{\uparrow}$ packet is generated from an observer-locality cell, finite cross-Hist causal route rows, a maximum-rate row, a no-global-frame ledger, componentwise hsame transport, psame stability, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 889.1, Definition 889.2, Definition 889.3, Definition 889.5, Definition 889.10, Lean4 ✓ `BEDC.Derived.ObserverLocalityTransportUp.taste gate`, and the checked single-carrier marker `BEDC.Derived.ObserverLocalityTransportUp.ObserverLocalityTransportTasteGate_sing` bind the public finite route $O, D, R, L, H, Q, C, P, N$ with the kernel rows BHist, hsame, psame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{ObserverLocalityTransport}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ObserverLocalityTransportUp.ObserverLocalityTransportTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive Lorentz geometry, metric tensors, synchronized clocks, global frames, host real-valued speed, Lean verification of the public export theorem, or bridge checking for $\mathsf{ObserverLocalityTransport}^{\uparrow}$.

Upgrade path: No further paper-side theory closure grade remains; only independent verification-axis and bridge-axis checks can strengthen the recorded status.

Chapter 890

A Concrete Naming Certificate for $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$

The $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$ packet realizes the runtime half of the reflection limit in chapter 9461. It sits beside the truth-total obstruction packet of chapter 472 and the certificate-indexed self-reference branch packet of chapter 667: those siblings record truth-total and branch-query refusals, while this packet records the concrete runtime-request surface that separates structural self-description from external kernel execution.

Definition 890.1 (Runtime-reflection boundary carrier)**.** *A* $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, E, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the structural self-description row over the internal* CIC^* *presentation, including formation, replay, and certificate-shape data;*
2. *Q is the runtime-request route, naming the object-level request that a consumer wants to treat as validation of the same structural presentation;*
3. *E is the execution-refusal row, recording that external kernel execution is not a carrier coordinate and cannot be supplied by the described object language;*
4. *V is the displayed runtime-boundary report, separating structural replay through S from the refused validator route through Q and E;*
5. *H records componentwise* hsame *transport for structural self-description, runtime requests, execution refusal, boundary reports, routes, provenance, and local names;*
6. *C records displayed* Cont *routes by which consumers read S, inspect Q, receive E, replay V, and transport the same packet through H;*
7. *P is the* Pkg *provenance over chapter 9461, chapter 472, chapter 667,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RuntimeReflectionBoundary}^{\uparrow}}$ *row.*

The carrier stores no BEDC-internal runtime validator, no truth-total classifier, no certificate-indexed truth branch, and no route that turns structural self-description into host execution.

Theorem 890.2 (Runtime-reflection boundary NameCert obligations)**.** *For the carrier of Definition 890.1, the local* $\mathsf{NameCert}_{\mathsf{RuntimeReflectionBoundary}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $R = (S, Q, E, V, H, C, P, N)$*;*
2. *structural self-description exactness, requiring every admitted description read to factor through* S*;*
3. *runtime-request accountability, requiring every attempted validator read to enter through the displayed route* Q*;*
4. *execution refusal and report exactness, requiring the refusal row* E *and boundary report* V *to be the only runtime-facing exports;*
5. *transport and provenance closure, requiring all comparisons, consumer routes, package provenance, and local naming to be exactly the displayed rows* H, C, P, N*.*

Thus a consumer can read structural self-description and the refused runtime request, but it cannot obtain a BEDC-internal runtime validator from the packet.

Proof. Carrier admission is projection from Definition 890.1. The rows S, Q, E, V, H, C, P, N are the complete finite packet.

Structural self-description exactness reads S first. The row displays formation, replay, and certificate-shape data, so a consumer may inspect the structural copy described in chapter 9461.

Runtime-request accountability reads Q before any runtime-facing claim. The requested validator route is a displayed object-level request, not the host act of validation.

Execution refusal is carried by E and reported by V. Since those are the only runtime-facing rows, every accepted runtime-boundary packet refuses any consumer path that asks for a BEDC-internal runtime validator row not displayed in the carrier.

Finally H, C, P, N transport, replay, package, and name the same finite surface. They add no hidden validator, truth-total classifier, certificate-indexed truth branch, internal-theory row, or identity-commitment row. □

Lean4 ✓ `BEDC.Derived.RuntimeReflectionBoundaryUp.RuntimeReflectionBoundary runtime request refusal route unary closure`

Theorem 890.3 (Runtime-reflection boundary TasteGate obligation)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ RuntimeReflectionBoundary}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ RuntimeReflectionBoundary}^{\uparrow}$ *instances. These instances must preserve the structural self-description row, runtime-request route, execution-refusal row, runtime-boundary report,* hsame *transport,* Cont *routes, package provenance, and local naming row.*

Proof. The carrier is not determined by structural self-description alone. It also carries the runtime-request route and execution-refusal row that mark the runtime half of the reflection limit.

Nontriviality rules out a collapsed packet where the runtime request is indistinguishable from the structural description. Field faithfulness records preservation of all displayed rows S, Q, E, V, H, C, P, N, so the future TasteGate surface can distinguish structural replay from refused internal validation. □

Lean4 ✓ `BEDC.Derived.RuntimeReflectionBoundaryUp.RuntimeReflectionBoundaryTasteGate single carrier alignment`

Falsifiable prediction: Every accepted runtime-boundary packet refuses any consumer path that asks for a BEDC-internal runtime validator row not displayed in the carrier.

Independence witness: TruthTotalReflection, SelfReferenceTruthBranch, InternalTheoryBoundary, MetaCICIdentityBoundary: the runtime-boundary carrier has an execution-refusal row and runtime-request route that none of these siblings carries, while it lacks their truth-total, certificate-indexed truth, internal-theory, or identity-commitment rows, so no bijective carrier match preserves NameCert obligations.

Closure status for $\mathsf{NameCert}_{\mathsf{RuntimeReflectionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$ packet is generated from a structural self-description row, a runtime-request route, an execution-refusal row, a runtime-boundary report, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuntimeReflectionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 890.1, Definition 890.2, and Definition 890.3 specify the finite runtime-boundary packet separating structural self-description from external kernel execution.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuntimeReflectionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide an internal runtime validator, a truth-total classifier, a certificate-indexed truth branch, Lean verification, or bridge checking for $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, structural-description exactness, runtime-request accountability, execution-refusal exactness, transport closure, and TasteGate nontriviality with field faithfulness.

Chapter 891

A Concrete Naming Certificate for $\mathsf{RealSealCongruence}^{\uparrow}$

The $\mathsf{RealSealCongruence}^{\uparrow}$ packet concretizes the replacement direction used by the Real-completeness target of chapter 9238. It consumes the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38, the finite-window schedule of chapter 126, the regular-rational readback surface of chapter 301, the refinement route of chapter 885, and the separated uniqueness boundary of chapter 427. The carrier below also records the Hausdorff separated-comparison consumer of chapter 402. Its role is not to assert quotient real equality; it records that two displayed readback routes are interchangeable for consumers that first read the same $\mathsf{Real}^{\uparrow}$ seal classifier.

Falsifiable prediction: Two accepted RealSealCongruence packets with the same displayed RealUp seal classifier admit no consumer-visible distinction between their RegSeqRat/StreamName readback routes.

Independence witness: cauchymodulusrefinement, regularlimituniqueness, cauchylimitseal, realtailagreementtransitivity: the carrier has a two-route seal-congruence row absent from refinement, uniqueness, single-seal, and tail-transitivity packets, while those siblings carry selector, separated-comparison, one-seal, or three-tail rows not bijective to this carrier.

Definition 891.1 (Real seal congruence carrier)**.** *A* $\mathsf{RealSealCongruence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R_0, W_0, E_0, R_1, W_1, E_1, A, J, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows whose Real-facing routes are being compared;*
2. W_0 *and* W_1 *are the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledgers used to inspect the two readbacks;*
3. E_0 *and* E_1 *are the two* $\mathsf{Real}^{\uparrow}$ *seal rows reached from those readbacks and windows;*
4. A *is the shared displayed* $\mathsf{Real}^{\uparrow}$ *seal classifier row, read after* E_0 *and* E_1 *are both present;*
5. J *is the seal-congruence row stating that a consumer may replace the route* R_0, W_0, E_0 *by the route* R_1, W_1, E_1 *only under the shared classifier row* A*;*
6. H *stores componentwise* hsame *transports for the two regular readbacks, the two window ledgers, the two seals, the shared classifier, the congruence row, provenance, and local name;*
7. C *stores the displayed* Cont *routes from each* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through its* $\mathsf{StreamName}^{\uparrow}$ *windows into its* $\mathsf{Real}^{\uparrow}$ *seal, and then through* A, J *to the local naming row;*

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, $\mathsf{RegularLimitUniqueness}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSealCongruence}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no host real equality, no selected limit representative, no ambient completion object, and no direct route from either readback to a consumer outside the displayed seal classifier.

Lean4 (definition) `BEDC.Derived.RealSealCongruenceUp.RealSealCongruenceUp`
Lean4 ✓ `BEDC.Derived.RealSealCongruenceUp.RealSealCongruenceTasteGate single carrier alignment`

Theorem 891.2 (Real seal congruence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealSealCongruence}^{\uparrow}}$ *surface for Definition 891.1 consists of seven finite obligations:*

1. *carrier admission for the packet $G = (R_0, W_0, E_0, R_1, W_1, E_1, A, J, H, C, P, N)$;*
2. $\mathsf{RegSeqRat}^{\uparrow}$ *readback admission for both R_0 and R_1 through chapter 301;*
3. *finite* $\mathsf{StreamName}^{\uparrow}$ *window coverage for W_0 and W_1 through chapter 126;*
4. $\mathsf{Real}^{\uparrow}$ *seal admission for E_0 and E_1 through chapter 38;*
5. *shared seal-classifier exactness for the row A, with the separated comparison discipline supplied by chapter 427 and chapter 402;*
6. *congruence replacement through J, stating that a consumer may exchange the two displayed readback routes only after A has been read;*
7. *ledger non-escape excluding quotient streams, host real equality, selected limits, ambient completion, and direct readback-to-consumer shortcuts.*

Proof. Carrier admission is projection from Definition 891.1. The tuple exposes exactly the two regular-rational readbacks, two finite window ledgers, two Real seal rows, shared classifier row, congruence row, transport, route, provenance, and local naming data.

The two readback routes are the sibling routes named by chapter 301, chapter 126, and chapter 38. Each route is read from a displayed $\mathsf{RegSeqRat}^{\uparrow}$ row through a displayed finite $\mathsf{StreamName}^{\uparrow}$ window ledger before it reaches its $\mathsf{Real}^{\uparrow}$ seal.

The shared classifier row A is the only comparison point between the two routes. The refinement schedule of chapter 885 supplies the finite-window discipline, while the separated comparison boundary of chapter 427 and chapter 402 keeps the comparison at the displayed seal rather than at a quotient real.

Finally, the congruence row J and the structural rows H, C, P, N make replacement consumer-visible only after the shared classifier has been read. Since the carrier has no coordinate for a host equality, selected limit, quotient stream, or ambient completion, no consumer can bypass A, J to distinguish the two accepted routes. □

Theorem 891.3 (Real seal congruence classifier exactness)**.** *For every accepted* $\mathsf{RealSealCongruence}^{\uparrow}$ *packet*

$$G = (R_0, W_0, E_0, R_1, W_1, E_1, A, J, H, C, P, N),$$

every consumer-visible replacement of the first readback route by the second factors through the finite route

$$R_0, W_0, E_0, A, J, E_1, W_1, R_1, H, C, P, N.$$

Thus the classifier row A is exact: if two accepted packets display the same A and the same transported structural rows H, C, P, N, a consumer cannot observe whether the $\mathsf{Real}^{\uparrow}$ *seal was reached through R_0, W_0, E_0 or through R_1, W_1, E_1. Conversely, without the displayed A and J rows, the carrier exports no replacement claim between the two readback routes.*

Proof. Read the carrier first. The only comparison row shared by the two sides is A, and the only replacement row is J; the routes into A are the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{Real}^{\uparrow}$ seals.

Apply the NameCert obligations of Definition 891.2. They force each side to be admitted through the same finite readback and seal discipline, then force replacement to pass through J after A.

Consumer exactness is therefore coordinate exhaustion. The rows H, C, P, N only transport, replay, record provenance, and name the same displayed packet. None of them adds quotient equality, a selected completed representative, host real equality, or a direct read from one route to the other.

For the converse boundary, remove A or J from the displayed route. The remaining coordinates contain two seal-reaching routes but no row that compares them or licenses replacement. Such a packet may show two Real-facing reads, but it is not a $\mathsf{RealSealCongruence}^{\uparrow}$ replacement packet. □

The follow-up Lean path for this packet is

lean4/BEDC/Derived/RealSealCongruenceUp/TasteGate.lean.

It should provide Nontrivial $\mathsf{RealSealCongruence}^{\uparrow}$ and FieldFaithful $\mathsf{RealSealCongruence}^{\uparrow}$ instances for the carrier, with the intended concrete declaration rooted at BEDC.Derived.RealSealCongruenceUp.RealSealCongruen

Closure status for $\mathsf{NameCert}_{\mathsf{RealSealCongruence}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSealCongruence}^{\uparrow}$ packet is generated from two RegSeqRat readback rows, two finite StreamName window ledgers, two RealUp seal rows, a shared RealUp seal classifier, a congruence replacement row, componentwise hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSealCongruence}^{\uparrow}}$, seed)

Scope closed: Definition 891.1, Definition 891.2, and Definition 891.3 bind the finite two-route seal-congruence packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSealCongruence}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient real equality, host real equality, selected limit representatives, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, seal-congruence exactness, and non-escape for the same two-route Real seal packet.

Chapter 892

A Concrete Naming Certificate for $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$

The $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ packet realizes the normalization boundary isolated in chapter 9386. It sits beside the open-frontier ledger of chapter 762, the normalization audit packet of chapter 750, the normalization audit seal of chapter 777, and the closed-normal route certificate of chapter 858. Those siblings record the inventory, audit route, seal route, and consumer route; this packet keeps the candidate-evidence boundary and the finished-normal boundary visible as separate coordinates.

Definition 892.1 (MetaCIC normalization frontier carrier)**.** *A* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

with the following displayed rows:

1. *C is the candidate-evidence row for the inspected empty-context typed closed MetaCIC terms;*
2. *K is the closed-candidate evidence row, recording where the closed term is actually placed into the candidate interface;*
3. *E is the finished-normal evidence row available after bounded reduction or displayed normal-form inspection;*
4. *N is the normal-endpoint use row consumed by closed-normal consistency routes;*
5. *S is the subject-reduction obstruction row, kept visible whenever candidate evidence or finished-normal evidence does not erase the preservation boundary;*
6. *H records componentwise* hsame *transport for C, K, E, N, S, R, P, L and the local packet shape;*
7. *R records the displayed* Cont *replay routes by which downstream consumers read candidate evidence, finished-normal evidence, normal endpoints, and obstruction rows;*
8. *P is the* Pkg *provenance over chapter 9386, chapter 762, chapter 750, chapter 777, and chapter 858;*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$ *row.*

The carrier is finite and obstruction-preserving. It records where a closed-normal consumer may read normalization evidence, but it does not turn the normalization frontier into a blanket closed strong-normalization theorem.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.taste gate`

Theorem 892.2 (MetaCIC normalization frontier NameCert obligations)**.** *Every accepted* MetaCICNormalizationFr *packet with carrier Definition 892.1 carries exactly the following local NameCert obligations:*

1. *carrier admission for $F = (C, K, E, N, S, H, R, P, L)$ as one finite* BHist *packet;*
2. *candidate-evidence boundary: a read of candidate-mediated strong-normalization content must pass through C and the closed-candidate row K;*
3. *finished-normal boundary: a read of finished-normal content must pass through E before it can feed the normal-endpoint row N;*
4. *closed-normal consumer handoff: any consistency-facing read of N must replay through R with provenance P and the local naming row L;*
5. *subject-reduction-obstruction separation: the obstruction row S remains distinct from C, K, E, N and is not discharged by the normalization packet.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 892.1. The nine rows $C, K, E, N, S, H, R, P, L$ are the complete finite packet.

For the candidate boundary, inspect C and K. The candidate row records the available normalization evidence, while the closed-candidate row records the point at which an empty-context closed term is admitted to that interface.

For the finished-normal boundary, inspect E and N. The endpoint row is available to a closed-normal consumer only after the finished-normal evidence row has been read through the displayed replay route.

The handoff and separation clauses use the structural rows. The replay row R, provenance row P, and local row L expose only the displayed normalization frontier, and the obstruction coordinate S remains a named coordinate rather than a hidden subject-reduction theorem. □

Theorem 892.3 (MetaCIC normalization frontier carrier rows)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, carrier admission exposes the concrete finite packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

as a BHist *carrier with candidate rows C, K, finished rows E, N, obstruction row S, componentwise* hsame *row H, displayed* Cont *replay row R,* Pkg *provenance row P, and local* NameCert *row L. No carrier read may replace these rows by an unnamed normalization source.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier carrier rows`

Proof. Project the carrier of Definition 892.1. It lists $F = (C, K, E, N, S, H, R, P, L)$ as one finite BHist packet.

Apply the carrier-admission clause of Definition 892.2. That clause routes every carrier read through the displayed rows before any downstream closed-normal consumer may use the frontier.

The row S remains visible in the same projection. Since it is a named obstruction coordinate rather than an erased proof obligation, carrier admission cannot be read as a blanket normalization theorem. □

Theorem 892.4 (MetaCIC normalization frontier classifier stability)**.** *In an accepted* MetaCICNormalizationFrontier *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

the local classifier is stable under componentwise hsame *transport only along the displayed* BHist *rows: candidate-mediated reads pass through C, K, finished-normal reads pass through E, N, and both routes replay through the* Cont *row R with* Pkg *provenance P and local* NameCert *row L. The obstruction row S is transported as an obstruction row and is not discharged by classifier stability.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier classifier stability`

Proof. Use the NameCert obligations to separate the two positive evidence routes. Candidate reads are routed through C, K, while finished-normal reads are routed through E, N before the normal endpoint is consumed.

Transport is componentwise through H. It can preserve the displayed candidate, finished, replay, provenance, and local naming rows, but it has no coordinate that turns a candidate row into a finished row or erases the obstruction row S.

The replay row R and package row P then expose the same classified frontier surface to downstream consumers. Thus classifier stability is stability of the displayed packet, with S retained as an obstruction coordinate. □

Theorem 892.5 (MetaCIC normalization frontier ledger exhaustion)**.** *Every ledger read exported by an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

is exhausted by the candidate rows C, K or the finished rows E, N, together with BHist *carrier exposure, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, the local* NameCert *row L, and the retained obstruction row S. A read bypassing both C, K and E, N is outside the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier ledger exhaustion`

Proof. Start with the local obligation inventory. The candidate-evidence boundary names C, K, and the finished-normal boundary names E, N.

The ledger support rows do not supply a third positive evidence source. The row H transports displayed fields, R replays displayed consumer routes, P records package provenance, and L names the local certificate surface.

It remains to account for S. The obstruction row is carried alongside both positive routes, so exhaustion of the ledger does not solve or erase the subject-reduction boundary. Therefore any ledger read outside C, K and E, N has no accepted normalization-evidence coordinate in the packet. □

Theorem 892.6 (MetaCIC normalization frontier non-escape)**.** *An accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet exports no blanket strong-normalization theorem for all empty-context typed closed MetaCIC terms, no full closed consistency theorem, and no erased subject-reduction transport. Any closed-normal consistency consumer route must factor through the candidate-evidence rows C, K or through the finished-normal row E before it reads N; a route that bypasses both is outside the carrier.*

Proof. Project the carrier. The only normalization evidence rows are C, K and E, and the only closed-normal endpoint row is N.

The replay and provenance rows R and P do not add new normalization evidence. They only expose the recorded candidate-evidence route, the finished-normal route, and the local endpoint route to a consumer.

The obstruction row S is separate from those positive rows. Since the packet has no coordinate for global closed strong normalization, full closed consistency, or erased subject-reduction transport, none of those objects can be exported by its local certificate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierNonescape`

Theorem 892.7 (MetaCIC normalization frontier redex boundary)**.** *In every accepted* MetaCICNormalizationFronti *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

the candidate rows C, K and the finished-normal rows E, N determine a finite redex-boundary route for the inspected empty-context typed closed MetaCIC terms. Any read that turns candidate evidence into finished-normal endpoint use must pass through C, K, E, N and retain the obstruction row S as a displayed coordinate, with replay and provenance supplied only by H, R, P, L.

The route is a BEDC frontier route, not a blanket strong-normalization theorem, not a closed consistency theorem, and not an erased subject-reduction transport.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier redex boundary`

Proof. Project the carrier in Definition 892.1. The rows $C, K, E, N, S, H, R, P, L$ are the finite packet, and S is listed separately from the positive candidate and finished-normal evidence rows.

Apply Definition 892.2. Its candidate-evidence boundary forces candidate-mediated reads through C, K, while its finished-normal boundary places E before N.

Now use Definition 892.6. Since the packet exports no blanket strong-normalization theorem, no full closed consistency theorem, and no erased subject-reduction transport, the redex boundary remains exactly the displayed finite route. □

Theorem 892.8 (MetaCIC normalization frontier fuel route)**.** *Every bounded replay of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet has a finite fuel route from candidate evidence C, K to finished-normal evidence E, N, with the obstruction row S carried alongside the replay rows H, R, P, L. The fuel route may be consumed by closed-normal downstream packets only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates of*

$$F = (C, K, E, N, S, H, R, P, L).$$

It exports no unbounded normalization schedule, no global subject-reduction proof, and no closed consistency theorem.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier fuel route`

Proof. Start from Definition 892.1. The packet is finite, so a bounded replay reads only the named carrier rows and cannot synthesize an additional normalization schedule.

Apply the handoff clauses in Definition 892.2. Candidate evidence enters through C, K, finished-normal endpoint use enters through E, N, and downstream closed-normal use must replay through R with provenance P and local naming row L.

Use Definition 892.41 for the row S. The obstruction is transported as a separate coordinate and is not discharged by the positive evidence rows.

Finally apply Definition 892.6. The finite replay route therefore remains a fuel route inside the packet and does not export unbounded normalization, global subject reduction, or closed consistency. □

Theorem 892.9 (MetaCIC normalization frontier root carrier rows)**.** *Every root consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet first reads exactly the finite carrier rows*

$$C, K, E, N, S, H, R, P, L.$$

The rows are the candidate-evidence surface, the closed-candidate surface, the finished-normal evidence surface, the normal-endpoint surface, the obstruction row, the transport row, the replay row, the provenance row, and the local naming row of the root packet.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root carrier rows`

Proof. Project the carrier definition in Definition 892.1. It lists $F = (C, K, E, N, S, H, R, P, L)$ as the complete packet and gives no additional root-facing coordinate.

Apply the NameCert obligations of Definition 892.2. Candidate reads enter through C, K, finished-normal reads enter through E, N, obstruction transport keeps S separate, and closed-normal downstream use replays through H, R, P, L.

Finally use the fuel route of Definition 892.8. Any root consumer that asks for a bounded replay remains inside those carrier rows before it can be consumed by a downstream closed-normal packet. □

Theorem 892.10 (MetaCIC normalization frontier root consumer non-escape)**.** *No root consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet can extract a blanket strong-normalization theorem, a closed consistency theorem, or erased subject-reduction transport from the rows* $C, K, E, N, S, H, R, P, L$.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root consumer nonescape`

Proof. Start with Definition 892.9. It exhausts the root packet by the displayed carrier rows and leaves no hidden normalization, consistency, or erased-transport coordinate.

Use Definition 892.6. The chapter already separates candidate and finished-normal evidence from blanket strong normalization, full closed consistency, and erased subject-reduction transport.

Now apply Definition 892.7 together with Definition 892.8. The root consumer receives only a bounded redex-boundary and fuel route, with S still displayed as an obstruction row, so none of the excluded global objects can be read from the packet. □

Theorem 892.11 (MetaCIC normalization frontier root candidate-gate projection)**.** *Every root consumer that asks the candidate gate of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must project the finite* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

through the candidate rows C, K *before it can use replay, provenance, or local naming support. The projection exposes no candidate evidence outside the displayed packet and leaves the obstruction row* S *present as a separate row.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root candidate gate projection`

Proof. Begin with Definition 892.9. It identifies $F = (C, K, E, N, S, H, R, P, L)$ as the whole root-facing carrier, so a root read has no unnamed candidate coordinate.

Apply the candidate-evidence boundary from Definition 892.2. That boundary places candidate access through C, K and attaches the read to the same local NameCert surface.

Use Definition 892.39 for the downstream support rows. It routes candidate consumption through C, K, R, P, L, while Definition 892.10 rules out an external candidate source.

The obstruction row is controlled by Definition 892.41. Since S is a named row of the same finite packet, the candidate-gate projection does not discharge it. □

Theorem 892.12 (MetaCIC normalization frontier root normal-endpoint route exhaustion)**.** *Every root consumer that reaches the normal endpoint of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must route through the finite* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

by the finished-normal evidence row E, the normal-endpoint row N, the replay row R, the provenance row P, and the local naming row L. There is no root endpoint route that bypasses E, N, R, P, L or erases the obstruction row S.

Proof. Project the root carrier rows. The finite packet names E, N, R, P, L as the endpoint-facing and support rows and provides no additional normal-endpoint coordinate.

Use the finished-normal boundary in Definition 892.2. It places the finished evidence row E before endpoint use through N and keeps both rows under the local certificate interface.

Next apply Definition 892.40 and Definition 892.51. Together they force endpoint consumption through E, N, R, P, L rather than through a free-standing closed-normal witness.

Finally invoke Definition 892.10 and Definition 892.47. They exclude a bypassing endpoint object and keep S visible throughout the route. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root normal endpoint route exhaustion`

Theorem 892.13 (MetaCIC normalization frontier root obstruction-preserving NameCert triad)**.** *For an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the root-facing* NameCert *surface presents exactly the obstruction-preserving triad of the candidate-gate projection through C, K, the normal-endpoint route through E, N, R, P, L, and the retained obstruction row S inside the finite* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The triad is transported only by the displayed hsame, Cont, Pkg, *and local naming coordinates of the packet.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootObstructionPreservi`

Proof. Start with the root carrier rows and the NameCert obligations. They place the local certificate over the same finite packet $F = (C, K, E, N, S, H, R, P, L)$ and do not supply a second certificate target.

Apply Definition 892.11 for the candidate side. It isolates the candidate gate at C, K and leaves replay, provenance, and local naming support inside the displayed packet.

Apply Definition 892.12 for the endpoint side. It exhausts normal-endpoint access by E, N, R, P, L and keeps the endpoint route within the same BHist packet.

It remains to track the obstruction row. Use Definition 892.41 and Definition 892.10; together they keep S as a retained row and prevent the triad from exporting blanket normalization, closed consistency, or erased subject-reduction transport. □

Theorem 892.14 (MetaCIC normalization frontier root candidate-endpoint coherence)**.** *Any root consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet that moves from the candidate rows C, K to the endpoint rows E, N preserves the same finite packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The route cannot switch candidate evidence sources between the candidate gate and the normal endpoint.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root candidate endpoint coherence`

Proof. Begin with Definition 892.9. It identifies $F = (C, K, E, N, S, H, R, P, L)$ as the complete root-facing packet.

Apply Definition 892.11. The candidate side enters through C, K and attaches replay, provenance, and local naming to the same carrier rows.

Apply Definition 892.12. The endpoint side reaches E, N only through E, N, R, P, L and has no second endpoint coordinate.

Finally use Definition 892.13 together with Definition 892.10. The triad transports the candidate and endpoint routes through the same hsame, Cont, Pkg, and naming rows, while non-escape rules out an external candidate source. □

Theorem 892.15 (MetaCIC normalization frontier root obstruction route separation)**.** *For every root consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the obstruction row S is transported alongside both the candidate route through C, K and the endpoint route through E, N, R, P, L. It is never consumed as positive normalization evidence.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root obstruction route separation`

Proof. Project the root carrier rows. The row S is a named coordinate of the same finite packet as C, K, E, N, R, P, L.

Use Definition 892.11. The candidate route leaves S present as a separate row rather than discharging it.

Use Definition 892.12. The endpoint route cannot bypass or erase S, because endpoint access is exhausted by E, N, R, P, L inside the same packet.

The separation claim is then exactly the obstruction part of Definition 892.13. Together with Definition 892.10, it prevents S from being read as a blanket strong-normalization witness. □

Theorem 892.16 (MetaCIC normalization frontier root closed-normal route totality)**.** *Every closed-normal consumer route of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *root packet is exhausted by the candidate route, endpoint route, replay row R, provenance row P, local naming row L, and obstruction row S of*

$$F = (C, K, E, N, S, H, R, P, L).$$

There is no closed-normal root route outside those displayed rows.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root closed normal route totality`

Proof. Start with Definition 892.9. The root packet exposes no row beyond $C, K, E, N, S, H, R, P, L$.

Apply Definition 892.14. It prevents a closed-normal consumer from changing candidate evidence sources when moving from C, K to E, N.

Apply Definition 892.15. The obstruction row remains transported beside the candidate and endpoint routes and cannot be converted into positive normalization evidence.

Finally use Definition 892.13 and Definition 892.10. They exhaust the route by the displayed certificate triad and exclude blanket normalization, closed consistency, erased subject-reduction transport, or any hidden closed-normal endpoint. □

Theorem 892.17 (MetaCIC normalization frontier obligation carrier totality)**.** *Carrier admission for an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

is exhausted by the displayed BHist *rows. The carrier of Definition 892.1, the NameCert obligations of Definition 892.2, and the root carrier rows of Definition 892.9 leave no admitted frontier coordinate outside* $C, K, E, N, S, H, R, P, L$.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obligation carrier totality`

Proof. Project the carrier definition. It presents $F = (C, K, E, N, S, H, R, P, L)$ as the finite frontier packet and assigns the candidate, finished-normal, obstruction, transport, replay, provenance, and local naming roles to those rows.

Apply the NameCert obligations. Candidate reads are bound to C, K, finished-normal reads are bound to E, N, and the obstruction row S remains separated from the positive normalization surfaces.

Finally use the root carrier rows. Every root consumer begins from the same displayed BHist packet, so carrier admission is total over exactly those rows and has no hidden normalization, consistency, or erased-transport coordinate. □

Theorem 892.18 (MetaCIC normalization frontier obligation evidence split)**.** *For an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet, positive normalization evidence splits before any closed-normal handoff into the candidate route through* C, K *and the finished-normal route through* E, N*. This split is the joint content of Definition 892.39, Definition 892.40, Definition 892.46, and Definition 892.42.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obligation evidence split`

Proof. Start with the candidate route. It routes candidate-facing consumers through C, K, R, P, L and keeps the read inside the same finite packet.

Turn to the finished-normal route. It routes endpoint-facing consumers through E, N, R, P, L and makes the normal endpoint available only after the finished-normal evidence row has been read.

The evidence-choice exhaustion theorem supplies the dichotomy. It rules out a third positive normalization evidence surface between those two routes.

Repack the result through the public obligation package. Since the package records carrier admission, the two positive routes, and the retained obstruction row together, any closed-normal handoff receives one of the two displayed positive surfaces rather than an unnamed normalization source. □

Theorem 892.19 (MetaCIC normalization frontier obligation obstruction retention)**.** *For an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the subject-reduction obstruction row S is retained across both positive routes and cannot be consumed as normalization evidence. The obstruction ledger of Definition 892.41, the retention theorem of Definition 892.47, and the subject-reduction compatibility theorem of Definition 892.65 keep S visible beside the candidate rows C, K and the finished-normal rows E, N.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obligation obstruction retention`

Proof. Read the obstruction ledger first. It names S as a separate coordinate of the same frontier packet as C, K, E, N, H, R, P, L.

Apply obstruction retention along the candidate and finished-normal routes. The candidate side through C, K and the endpoint side through E, N transport S rather than turning it into positive normalization evidence.

Use subject-reduction obstruction compatibility for downstream consumers. A SubjectReductionBoundary-facing consumer may read S as an obstruction coordinate, but it cannot discharge that row by consuming either positive normalization surface. □

Theorem 892.20 (MetaCIC normalization frontier candidate coverage)**.** *Every candidate-facing consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

must read the candidate evidence row C, the closed-candidate placement row K, the displayed replay row R, the provenance row P, and the local naming row L. The structural row H may transport the read componentwise, but it cannot replace C, K, R, P, L by an unnamed normalization source.

Proof. Project the carrier and NameCert obligations. The carrier lists $C, K, E, N, S, H, R, P, L$ as the complete finite packet, and the candidate-evidence boundary assigns candidate-facing reads to C and K.

The consumer side is then routed by the displayed support rows. The row R replays the candidate read, P records the package provenance, and L names the local certificate surface; H only transports the displayed rows.

Apply non-escape. A candidate consumer that bypasses any of C, K, R, P, L would need a hidden normalization coordinate outside the accepted packet, so candidate coverage is exactly the displayed route through those rows. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate coverage`

Theorem 892.21 (MetaCIC normalization frontier endpoint readback)**.** *Every normal-endpoint consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet reads the finished-normal evidence row E, the normal-endpoint row N, the displayed replay row R, the provenance row P, and the local naming row L, while retaining the subject-reduction obstruction row S as a visible coordinate of the same packet. A read of N outside E, N, R, P, L is not an endpoint readback of the frontier.*

Proof. Start with the finished-normal boundary. The NameCert obligations place E before N, and the carrier keeps both rows inside the same finite BHist packet as S, H, R, P, L.

Next inspect the handoff rows. The endpoint is replayed through R, packaged through P, and locally named by L; none of those rows supplies an independent endpoint witness.

It remains to account for the obstruction row. Since S is displayed separately from E and N, endpoint readback preserves S rather than discharging it. Non-escape then rules out any free-standing read of N outside the stated route. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier endpoint readback`

Theorem 892.22 (MetaCIC normalization frontier consumer non-escape)**.** *No closed-normal consumer or SubjectReductionBoundary-facing consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet can bypass both positive routes C, K, R, P, L and E, N, R, P, L while remaining inside the packet. Such a consumer also cannot erase the obstruction row S or replace it by blanket strong normalization, full closed consistency, or erased subject-reduction transport.*

Proof. First exhaust the positive routes. Candidate coverage forces candidate-facing reads through C, K, R, P, L, while endpoint readback forces normal-endpoint reads through E, N, R, P, L.

Now consider a closed-normal or SubjectReductionBoundary-facing consumer that avoids both routes. The carrier has no third positive normalization evidence surface: H transports, R replays, P records provenance, and L names the local certificate surface.

Finally use the obstruction row. The row S remains a displayed obstruction coordinate under both positive routes, and the non-escape theorem excludes any hidden coordinate that could erase it. Therefore a consumer that bypasses both positive routes or erases S is outside the accepted frontier packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer nonescape`

Theorem 892.23 (MetaCIC normalization frontier consumer-boundary lock)**.** *Every public consumer of the* MetaCICNormalizationFrontier$^{\uparrow}$ *completion surface must pass through the displayed source, boundary, route, readback,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows already present in the child packet. In particular, such a consumer cannot import a hidden normalization oracle, host equality, subject-reduction bridge, or external consistency claim as a replacement for the displayed boundary packet.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer boundary lock`

Proof. Project the consumer-boundary rows already established in this child file. Candidate-facing reads pass through C, K, R, P, L, while normal-endpoint reads pass through E, N, R, P, L and retain S as a visible obstruction coordinate.

The remaining structural access is exhausted by the same packet rows. Componentwise hsame transport may move displayed coordinates, Cont may replay the displayed routes, Pkg records their provenance, and the local NameCert row names the boundary surface.

Coordinate exhaustion now rules out bypasses. A hidden normalization oracle, host equality proof, subject-reduction bridge, or external consistency claim is not one of the displayed rows, so it cannot be supplied by a public consumer of the accepted completion surface. □

Theorem 892.24 (MetaCIC normalization frontier public route exhaustion)**.** *Every public consumer route through an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet chooses either the candidate route C, K, R, P, L or the endpoint route E, N, R, P, L. In both cases it retains the obstruction row S and the structural rows H, R, P, L. The packet exports no third route through hidden normalization, scheduler state, host equality, subject-reduction bridge, or external consistency claim.*

Proof. Apply Definition 892.20 to the candidate-facing case. The read is forced through C, K, R, P, L, with H only transporting displayed rows.

Apply Definition 892.21 to the normal-endpoint case. The read is forced through E, N, R, P, L, while S remains a visible obstruction coordinate of the same packet.

Finally combine Definition 892.22 with Definition 892.23. A public consumer that avoids both positive routes, erases S, or imports a hidden normalization, scheduler, host-equality, subject-reduction, or external consistency row is outside the accepted frontier packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer route public exhaustion`

Definition 892.25 (MetaCIC normalization frontier public export surface)**.** *The public export surface of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet is the finite row package*

$$F_{\mathrm{pub}} = (C, K, E, N, S, H, R, P, L).$$

Here C is candidate evidence, K is closed-candidate placement, E is finished-normal evidence, N is the normal endpoint, S is the subject-reduction obstruction, H is componentwise hsame *transport, R is the displayed* Cont *replay surface, P is* Pkg *provenance, and L is the local* NameCert *row. A public read of the frontier packet is a read of this finite surface, not a read of an ambient normalization theorem.*

Lean4 (definition) `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierPublicExportSur`
Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierPublicExportSurfaceCert`

Theorem 892.26 (MetaCIC normalization frontier public export totality)**.** *Every public* MetaCICNormalizationFro *consumer factors through one of the two displayed public routes:*

1. *the candidate surface C, K, R, P, L;*
2. *the finished-normal surface E, N, R, P, L.*

In both cases the row S is retained as subject-reduction obstruction data, and H transports only the displayed components of the same finite public surface.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierPublicExportTotality`

Proof. Start with the carrier and local certificate obligations. The carrier Definition 892.1 supplies exactly the rows $C, K, E, N, S, H, R, P, L$, and Definition 892.2 assigns consumer reads to the candidate boundary, the finished-normal boundary, and the displayed handoff rows.

The positive route is then exhausted by the existing route theorems. Use Definition 892.44 and Definition 892.46: a public consumer must choose C, K or E, N, with replay, provenance, and local naming supplied by R, P, L.

Finally use obstruction retention. The theorem Definition 892.47 keeps S visible under both choices, and non-escape prevents H, R, P, L from becoming a third positive normalization route. □

Theorem 892.27 (MetaCIC normalization frontier public export non-escape)**.** *The public export surface of Definition 892.25 cannot be read as blanket strong normalization for all empty-context typed closed MetaCIC terms, full closed consistency, erased subject-reduction transport, or a third hidden normalization evidence route beyond C, K, R, P, L and E, N, R, P, L.*

Proof. Project the public surface to the existing carrier. The definition of F_{pub} is only the displayed finite row package $C, K, E, N, S, H, R, P, L$, so any public read must stay inside the same carrier, transport, replay, package, and local naming coordinates.

Apply the chapter's non-escape and evidence-choice results. Definition 892.6 excludes blanket normalization, closed consistency, and erased subject-reduction transport, while Definition 892.46 leaves only the candidate and finished-normal evidence choices.

It remains to rule out a hidden route. Consumer-route totality gives no public path outside C, K, R, P, L and E, N, R, P, L, and obstruction retention keeps S as obstruction data rather than positive normalization evidence. Therefore the public export boundary is exactly the two-route surface with the obstruction row retained. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierPublicExport nonescape`

Definition 892.28 (MetaCIC normalization frontier visible obstruction packet)**.** *The visible obstruction packet of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *public read is the finite package*

$$O_{\text{vis}} = (C, N, O, S, B, H, R, P, L).$$

Here C is candidate evidence, N is the normal endpoint readback, O is the obstruction ledger, S is the subject-reduction obstruction reference, B is the displayed BHist *support, H is the componentwise* hsame *transport, R is the* Cont *replay surface, P is* Pkg *provenance, and L is the local* NameCert *row. A consumer reads O_{vis} as obstruction content for the public frontier, not as a proof of ambient normalization.*

Lean4 (definition) `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierVisibleObstruct`

Theorem 892.29 (MetaCIC normalization frontier visible obstruction admission)**.** *Every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *public export admits the visible obstruction packet O_{vis} of Definition 892.28. The packet displays candidate evidence, normal endpoint readback, obstruction ledger, subject-reduction obstruction reference,* BHist, hsame, Cont, Pkg, *and local* NameCert *rows on the same finite public surface.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierVisibleObstructionAdmis`

Proof. Start from the public export surface Definition 892.25. It already contains candidate evidence, finished-normal evidence, the normal endpoint, the subject-reduction obstruction, componentwise transport, replay, provenance, and local naming rows.

Apply public export totality, Definition 892.26. A public consumer is forced onto either the candidate route or the finished-normal route, and in both cases the subject-reduction obstruction row is retained.

Finally read the retained obstruction as a visible ledger. The same public surface supplies BHist, hsame, Cont, Pkg, and NameCert support through the displayed rows, so repackaging those coordinates gives O_{vis} without adding a third normalization route. □

Theorem 892.30 (MetaCIC normalization frontier visible obstruction route factorization)**.** *Every public route through the visible obstruction packet factors through the finite package $C, N, O, S, B, H, R, P, L$ and then through one of the two public export routes C, K, R, P, L or E, N, R, P, L. The obstruction-preserving route is finite, and it exports no blanket strong-normalization theorem, no closed consistency proof, and no erased subject-reduction transport.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierVisibleObstruction route factorization`

Proof. First project the visible packet back to the public export surface. The rows $C, N, O, S, B, H, R, P, L$ are read from the same carrier, obstruction, replay, provenance, and local naming coordinates used by Definition 892.25.

Next use the public totality theorem. By Definition 892.26, the positive part of the route must be C, K, R, P, L or E, N, R, P, L. The obstruction rows O and S remain attached to that choice as retained data.

The finite route has no escape coordinate. The non-escape theorem Definition 892.27 rules out blanket strong normalization, closed consistency, erased subject-reduction transport, and any hidden positive route beyond the two displayed export routes. Thus the visible obstruction route factors through the finite public surface only. □

Theorem 892.31 (MetaCIC normalization frontier visible obstruction consumer non-escape)**.** *A downstream consumer of O_{vis} may cite the candidate evidence, normal endpoint readback, obstruction ledger, subject-reduction obstruction reference,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows as frontier obstruction content. It cannot convert that packet into ambient normalization, full closed consistency, erased subject-reduction transport, or a consumer-owned normalization witness outside the two public export routes.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierVisibleObstructionConsu`

Proof. Read the consumer through the packet definition. The only available rows are $C, N, O, S, B, H, R, P, L$, so every cited coordinate is candidate, endpoint, obstruction, support, replay, provenance, or local naming data.

Route factorization then fixes the public boundary. By Definition 892.30, the consumer's positive access factors through C, K, R, P, L or E, N, R, P, L, with the obstruction ledger retained as obstruction content.

Finally apply public export non-escape. The theorem Definition 892.27 forbids ambient normalization, full closed consistency, erased subject-reduction transport, and hidden normalization evidence. The visible packet therefore serves as citable obstruction content, not as a replacement for the frontier's finite public export discipline. □

Theorem 892.32 (MetaCIC normalization frontier candidate evidence totality)**.** *Every accepted frontier candidate evidence route of a* MetaCICNormalizationFrontier$^{\uparrow}$ *packet is read from the displayed* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The route passes through the candidate evidence row C, the audit-facing evidence row K, the frontier seal row S, the normal-route rows E, N, componentwise hsame *transports in H, displayed* Cont *routes in R,* Pkg *provenance in P, and the local* NameCert *row L. No accepted candidate evidence route has a second source outside those rows.*

Proof. Start with the carrier surface of Definition 892.1. It displays one finite BHist packet and assigns the candidate, audit-facing, seal, normal-route, transport, continuation, provenance, and local naming roles to $C, K, E, N, S, H, R, P, L$.

Apply the NameCert obligations of Definition 892.2. The candidate-evidence boundary forces candidate reads through C, K, the finished-normal boundary places E before N, and the closed-normal handoff keeps the consumer route under the displayed replay and provenance rows.

Use the obligation closure package and route surface. Definition 892.17 exhausts carrier admission by the displayed rows, while Definition 892.39 and Definition 892.38 rule out a candidate route outside the BHist, hsame, Cont, Pkg, and NameCert coordinates. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate evidence totality`

Theorem 892.33 (MetaCIC normalization frontier seal boundary)**.** *In an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, a frontier seal cannot be replaced by a closed-normal route unless the candidate-evidence rows C, K and the audit rows carried by the displayed* BHist *packet are already exposed. A closed-normal consumer may use the seal only through S, H, R, P, L together with the normal-route rows E, N and the local* NameCert *row.*

Proof. Read the root carrier rows of Definition 892.9. A root consumer first receives the same finite packet $C, K, E, N, S, H, R, P, L$, so the frontier seal has no free-standing replacement coordinate.

Apply the candidate and endpoint route theorems. Definition 892.11 exposes the candidate side through C, K, and Definition 892.12 exposes normal-endpoint use through E, N, R, P, L.

It remains to keep the seal boundary visible. Definition 892.41 and Definition 892.47 keep S as a displayed row under hsame transport and Cont replay. Therefore the seal can participate in a closed-normal route only after the candidate-evidence and audit rows have already been displayed inside the same Pkg provenance and NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier seal boundary`

Theorem 892.34 (MetaCIC normalization frontier normal route non-escape)**.** *Normal-form consumers of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet cannot import host normalization, quotient term equality, or a hidden normalizer outside the displayed packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

Every normal-route read is confined to the candidate rows C, K, normal rows E, N, seal row S, componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier normal route nonescape`

Proof. Project the non-escape theorem Definition 892.6. It excludes blanket strong normalization, full closed consistency, and erased subject-reduction transport from the accepted packet.

Use the route theorems to identify the allowed normal-form path. Definition 892.40 routes finished-normal consumers through E, N, R, P, L, and Definition 892.16 exhausts closed-normal root routes by the candidate, endpoint, replay, provenance, local naming, and obstruction rows.

Finally apply Definition 892.42 and Definition 892.64. Their public and closed-normal surfaces expose only the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates, so host normalization, quotient term equality, and hidden normalizers have no accepted route into the packet. □

Theorem 892.35 (MetaCIC normalization frontier candidate confluence budget)**.** *A confluence-facing consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet may use the candidate rows C, K only as candidate evidence until the finished-normal boundary is displayed. It may use the finished rows E, N only after that boundary is visible through the same finite packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The consumer cannot identify the candidate-evidence route with the finished-normal route, cannot erase the obstruction row S, and cannot import blanket strong normalization, subject reduction, host normalization, quotient term equality, or a hidden normalizer.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate confluence budget`

Proof. Start with Definition 892.32. It confines candidate use to the displayed packet and makes C, K candidate evidence rows rather than completed normal-form rows.

Apply Definition 892.33. The seal row S remains visible between candidate evidence and closed-normal consumption, and the finished rows E, N become available only through the displayed boundary.

Finally apply Definition 892.34. Normal-form consumers remain inside $C, K, E, N, S, H, R, P, L$, so a confluence-facing consumer has no route that identifies candidate evidence with finished-normal evidence, erases S, or imports the excluded host normalization and quotient-equality principles. □

Theorem 892.36 (MetaCIC normalization frontier candidate SN consumer route)**.** *Any* MetaCICCandidateStrongN *consumer of a* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must read candidate evidence, finite fuel, endpoint normal-form rows, the normalization-audit route, the closed-normal route certificate, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before exporting an SN-facing boundary. The export is candidate-mediated: it does not assert unrestricted strong normalization, closed subject reduction, global confluence, quotient syntax equality, or a hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate sn consumer route`

Proof. Project the frontier carrier of Definition 892.1. The candidate rows C, K, endpoint rows E, N, obstruction seal S, transport rows H, replay rows R, provenance row P, and local naming row L are the only coordinates available to the consumer.

Use the candidate evidence boundary already established in this file. Definition 892.32 confines candidate evidence to C, K inside the same displayed packet, while Definition 892.35 keeps candidate use separate from finished-normal evidence until the normal endpoint rows are exposed.

Route the SN-facing read through the sibling normalization surfaces. The candidate strong-normalization chapters provide only candidate-mediated SN rows, and the closed-normal route chapters provide only displayed closed-normal certificates; neither sibling supplies an unrestricted normalization theorem.

Finally apply normal-route non-escape from Definition 892.34. The consumer remains inside the finite candidate, fuel, endpoint, audit, closed-normal, transport, replay, provenance, and NameCert route, so no global strong-normalization, subject-reduction, confluence, quotient-equality, or hidden-normalizer row can be exported. □

Theorem 892.37 (MetaCIC normalization frontier root obligation triad)**.** *For downstream closed-normal packets, an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet presents exactly the following root obligation triad: the candidate boundary through C, K, the finished-normal boundary through E, N, and the obstruction separation through S, all transported through H, R, P, L.*

Proof. Use Definition 892.2 to read the three obligations. The candidate-evidence boundary is carried by C, K, the finished-normal boundary is carried by E, N, and subject-reduction obstruction remains separated at S.

Apply Definition 892.7. It identifies those three surfaces as the finite redex boundary for inspected empty-context typed closed MetaCIC terms.

Finally combine Definition 892.8 with Definition 892.10. Downstream closed-normal packets may consume the triad only through the displayed replay, provenance, transport, package, and local naming rows, and they cannot replace the triad with a blanket normalization, consistency, or erased subject-reduction object. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root obligation triad`

Theorem 892.38 (MetaCIC normalization frontier kernel non-escape)**.** *For every accepted* MetaCICNormalizationF *packet, the finite kernel-facing route determined by $C, K, E, N, S, H, R, P, L$ is exhausted by the candidate route of Definition 892.39, the finished route of Definition 892.40, the obstruction ledger of Definition 892.41, and the public obligation package of Definition 892.42. A kernel consumer cannot bypass those rows to obtain a blanket strong-normalization theorem, a full closed consistency theorem, or erased subject-reduction transport.*

Proof. First project the public obligation package of Definition 892.42. It lists the finite carrier, the candidate-evidence boundary, the finished-normal boundary, the closed-normal handoff through R, P, L, and the obstruction row S.

Next use Definition 892.39 and Definition 892.40. These two route the positive kernel-facing reads through C, K or through E, N, with replay, provenance, and naming supplied by the displayed rows.

Apply Definition 892.41. The row S remains a separate obstruction coordinate and cannot be erased by reading candidate or finished-normal evidence.

Finally invoke Definition 892.6. A kernel consumer outside the listed rows would require one of the exported objects that the packet explicitly lacks, so the finite frontier route has no escape coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierKernelNonescape`

Theorem 892.39 (MetaCIC normalization frontier candidate route)**.** *Every closed-normal consumer of candidate evidence in an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet factors through the candidate row C, the closed-candidate row K, the replay row R, the provenance row P, and the local naming row L. The support rows are read inside the same finite* BHist *packet and transported only by the displayed* hsame *and* Cont *coordinates.*

Proof. Project the carrier $F = (C, K, E, N, S, H, R, P, L)$. The only candidate-facing rows are C and K, and the only consumer-facing support rows for replay, provenance, and local naming are R, P, and L.

Apply the candidate-evidence boundary clause of Definition 892.2. It requires any candidate-mediated read to pass through C and K before it can be used by a closed-normal consumer.

Finally use Definition 892.6. A consumer bypassing C, K, R, P, L has no carrier coordinate from which to read candidate evidence, so the displayed route is the whole candidate route. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate route totality`

Theorem 892.40 (MetaCIC normalization frontier finished route)**.** *Every closed-normal consumer of finished-normal evidence in an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet factors through the finished-normal evidence row E, the normal-endpoint row N, the replay row R, the provenance row P, and the local naming row L. The route uses only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates of the frontier packet.*

Proof. Project the same carrier. The finished-normal side is represented by E and N, while R, P, and L record the replay, provenance, and local certificate rows through which a consumer can read that side.

Apply the finished-normal boundary and closed-normal handoff clauses of Definition 892.2. They place E before N and then replay the endpoint through R, P, L.

The non-escape theorem rules out a hidden blanket normalization coordinate. Thus a finished-normal consumer that stays inside the packet must read exactly the displayed finished route and cannot replace it with an ambient normalization theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier finished route totality`

Theorem 892.41 (MetaCIC normalization frontier obstruction ledger)**.** *In every accepted* MetaCICNormalizationFr *packet, the subject-reduction obstruction row S is transported and named as a separate coordinate. It cannot be discharged by reading the candidate rows C, K or the finished-normal rows E, N, and no consumer may erase it while remaining inside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Proof. Read the carrier definition. The row S is listed separately from C, K, E, N before the transport, replay, provenance, and local naming rows are attached.

Use the subject-reduction-obstruction separation clause of Definition 892.2. It states that S remains distinct from the candidate and finished-normal evidence surfaces.

Apply Definition 892.6. Since no erased subject-reduction transport is exported, S can only be carried as a named obstruction coordinate; it cannot be solved by reading C, K, E, N. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obstruction ledger`

Theorem 892.42 (MetaCIC normalization frontier public obligation package)**.** *Every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet exposes the public obligation package consisting of carrier admission for*

$$F = (C, K, E, N, S, H, R, P, L),$$

the candidate-evidence boundary C, K, the finished-normal boundary E, N, the closed-normal handoff through R, P, L, and the obstruction-separation row S.

The package is a finite BEDC obligation surface. It does not assert blanket strong normalization, full closed consistency, or erased subject-reduction transport.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierPublicObligationPackage`

Proof. Project the carrier of Definition 892.1. The rows $C, K, E, N, S, H, R, P, L$ are exactly the finite frontier packet.

Apply Definition 892.2. Its carrier, candidate-boundary, finished-normal-boundary, handoff, and obstruction-separation clauses are precisely the public obligation rows listed in the statement.

Finally use Definition 892.6. It prevents the public package from being read as a global normalization theorem, a full consistency theorem, or an erased subject-reduction transport. □

Theorem 892.43 (MetaCIC normalization frontier boundary disjointness)**.** *For an accepted* MetaCICNormalization *packet, the rows C, K, E, N, S remain distinct BEDC coordinates: candidate evidence C, closed-candidate placement K, finished-normal evidence E, normal-endpoint use N, and subject-reduction obstruction S. They cannot be collapsed into a blanket normalization theorem or a single unnamed normalization row.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier boundary disjointness`

Proof. Read the carrier in Definition 892.1. The five rows C, K, E, N, S are listed as separate coordinates before the transport, replay, provenance, and local naming rows are attached.

Now apply Definition 892.2. It assigns different obligations to the candidate boundary, the finished-normal boundary, the closed-normal handoff, and the obstruction row.

The non-escape theorem Definition 892.6 excludes the blanket theorem reading. Therefore no consumer can collapse C, K, E, N, S into a single normalization assertion while remaining inside the accepted packet. □

Theorem 892.44 (MetaCIC normalization frontier consumer-route totality)**.** *Every consumer route for an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet factors through either the candidate-evidence surface C, K or the finished-normal evidence surface E, N. The rows H, R, P, L only transport, replay, package, and name that surface; they do not add a third normalization evidence route.*

Proof. Start from the route inventory in Definition 892.1. The only evidence surfaces available to a consumer are C, K and E, N, with S retained as the obstruction coordinate.

Use Definition 892.2. The candidate-evidence clause routes through C, K, the finished-normal clause routes through E, N, and the handoff clause routes closed-normal use through R, P, L after one of those surfaces is present.

Finally apply Definition 892.6. A consumer bypassing both C, K and E, N has no carrier coordinate from which to read normalization evidence, so the listed alternatives are total for the accepted frontier packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer route totality`

Theorem 892.45 (MetaCIC normalization frontier carrier-admission route)**.** *Carrier admission for an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet exposes exactly the finite rows*

$$F = (C, K, E, N, S, H, R, P, L).$$

The rows C, K are the candidate and closed-candidate evidence surface, E, N are the finished-normal and normal-endpoint surface, S is the subject-reduction obstruction row, and H, R, P, L transport, replay, package, and locally name the displayed surface. Carrier admission exposes no blanket normalization source outside those rows.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier carrier admission route`

Proof. Project the carrier definition Definition 892.1. It lists the nine rows $C, K, E, N, S, H, R, P, L$ as the whole accepted packet.

Apply the NameCert obligations of Definition 892.2. Carrier admission is the first obligation, and the remaining clauses sort the rows into candidate evidence, finished-normal evidence, consumer handoff, and obstruction separation.

Use Definition 892.6. Since the packet exports no blanket strong-normalization theorem or erased subject-reduction transport, carrier admission cannot expose any normalization source beyond the displayed frontier rows. □

Theorem 892.46 (MetaCIC normalization frontier evidence-choice exhaustion)**.** *Every positive consumer route out of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet chooses either the candidate surface C, K or the finished-normal surface E, N. The rows H, R, P, L only transport, replay, package, and locally name the chosen surface; they do not supply a third positive normalization evidence row.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier evidence choice exhaustion`

Proof. Read the candidate route Definition 892.39. It routes candidate-facing consumers through C, K, R, P, L inside the same finite packet.

Read the finished route Definition 892.40. It routes finished-normal consumers through E, N, R, P, L and the displayed BHist, hsame, Cont, Pkg, and NameCert coordinates.

Apply consumer-route totality, Definition 892.44. It exhausts positive routes by the two evidence surfaces and assigns H, R, P, L only transport, replay, packaging, and naming roles. The non-escape theorem Definition 892.6 rules out a hidden third source. □

Theorem 892.47 (MetaCIC normalization frontier obstruction retention)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet, the subject-reduction obstruction row S survives every candidate read through C, K and every finished-normal read through E, N. It is transported and replayed as a displayed row of the same packet and is not discharged by any candidate or finished-normal evidence surface.*

Proof. Use the obstruction ledger Definition 892.41. It names S as a separate coordinate and states that reading C, K or E, N does not erase it.

Combine this with the candidate and finished routes. Definition 892.39 and Definition 892.40 consume their positive surfaces through R, P, L, but neither route includes a discharge coordinate for S.

Finally invoke Definition 892.44 and Definition 892.6. All positive reads choose one of the two evidence surfaces, while erased subject-reduction transport remains outside the packet; therefore S is retained through every accepted consumer route. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obstruction retention`

Theorem 892.48 (MetaCIC normalization frontier candidate-finished boundary totality)**.** *Every accepted consumer route for a* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet passes through either the candidate boundary rows C, K or the finished-normal boundary rows E, N. The displayed carrier*

$$F = (C, K, E, N, S, H, R, P, L)$$

has no third positive normalization evidence surface: H, R, P, L only transport, replay, package, and locally name one of the two boundary routes.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate finished boundary totality`

Proof. Project the carrier and NameCert obligations. The carrier lists C, K, E, N as the only positive normalization evidence rows, and Definition 892.2 assigns consumer reads to the candidate boundary or the finished-normal boundary.

Use the candidate route and finished route theorems. Definition 892.39 routes candidate-facing consumers through C, K, R, P, L, while Definition 892.40 routes finished-normal consumers through E, N, R, P, L.

Finally apply consumer-route totality, Definition 892.44, and non-escape, Definition 892.6. A route outside both positive boundaries has no carrier coordinate from which to read normalization evidence. □

Theorem 892.49 (MetaCIC normalization frontier root candidate readback)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet $F = (C, K, E, N, S, H, R, P, L)$, any root candidate consumer reads the candidate row C and the closed-candidate row K before it may read any finished-normal endpoint row. The consumer route is therefore rooted at C, K, R, P, L, not at the finished-normal pair E, N.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root candidate readback`

Proof. Project the carrier and NameCert obligations. The accepted packet is exactly $F = (C, K, E, N, S, H, R, P, L)$ and the local obligations assign candidate-facing reads to the candidate boundary before a finished-normal endpoint is exposed.

Apply the candidate route Definition 892.39. It routes candidate-facing consumers through C, K, R, P, L and leaves E, N as the separate finished-normal surface.

Finally use consumer-route totality and non-escape. A root candidate read that bypasses C, K or starts from E, N has no carrier coordinate from which to read candidate evidence, so it is not a read of the accepted packet. □

Theorem 892.50 (MetaCIC normalization frontier root obstruction retention)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet $F = (C, K, E, N, S, H, R, P, L)$, the obstruction row S remains present under every candidate read through C, K and every finished-normal read through E, N. Neither the endpoint row E nor the normal row N discharges S.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root obstruction retention`

Proof. Read the obstruction ledger Definition 892.41. It keeps S as a displayed coordinate of the same packet as C, K, E, N, H, R, P, L.

Use the candidate route and the finished route. The candidate route consumes C, K, R, P, L, while the finished route consumes E, N, R, P, L; neither route contains a discharge coordinate for subject-reduction obstruction.

Apply the obstruction consumer lock and kernel non-escape. Since erased subject-reduction transport is outside the packet, S remains visible under both positive evidence routes. □

Theorem 892.51 (MetaCIC normalization frontier root normal endpoint handoff)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet* $F = (C, K, E, N, S, H, R, P, L)$*, the normal endpoint row* N *is exposed only after the finished-normal evidence row* E *and only through the replay, provenance, and local-name rows* R, P, L*. A root consumer cannot read* N *as a free-standing normalization endpoint.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootNormalEndpointHando`

Proof. Project the finished route Definition 892.40. It routes finished-normal consumers through E, N, R, P, L inside the same accepted packet.

Use the public obligation package. Its handoff row assigns closed-normal consumer exposure to the displayed replay, provenance, and local naming rows, so N is not available before E or outside R, P, L.

Finally apply non-escape. A free-standing read of N would require a hidden blanket normalization coordinate, full closed consistency coordinate, or erased subject-reduction transport; none belongs to F. □

Theorem 892.52 (MetaCIC normalization frontier obstruction consumer lock)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the obstruction row* S *remains visible to consumer routes and cannot be erased by reading either the candidate boundary* C, K *or the finished-normal boundary* E, N*. The public route may transport* S *through* H, R, P, L*, but it cannot replace* S *by blanket strong normalization, full closed consistency, or erased subject-reduction transport.*

Proof. Read the obstruction ledger, Definition 892.41. It keeps S as a separate coordinate in the same finite packet as C, K, E, N, H, R, P, L.

Apply candidate and finished route exactness. The candidate route consumes C, K and the finished route consumes E, N, but neither theorem discharges the separate obstruction coordinate.

Use kernel non-escape and consumer-route totality. Definition 892.38 excludes a hidden global theorem behind the public route, and Definition 892.44 leaves only the two positive evidence surfaces with S still displayed. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obstruction consumer lock`

Theorem 892.53 (MetaCIC normalization frontier obligation triad)**.** *The public obligation surface of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet is the triad made of carrier admission for* $F = (C, K, E, N, S, H, R, P, L)$*, candidate-or-finished boundary totality through* C, K *or* E, N*, and obstruction separation for* S*, with consumer handoff through the displayed rows* H, R, P, L*.*

This triad is the chapter's consumer-facing obligation surface. It does not claim blanket normalization, closed consistency, or erased subject reduction.

Proof. First use the public obligation package, Definition 892.42. It supplies carrier admission, candidate evidence, finished-normal evidence, closed-normal handoff, and obstruction separation for the displayed packet.

Next apply Definition 892.48. This packages the positive evidence side into the two available boundary routes, C, K and E, N.

Then apply Definition 892.52. The obstruction row S stays visible alongside those positive routes and is not erased by the handoff.

Finally use non-escape, Definition 892.6. Since the packet has no coordinate for blanket normalization, closed consistency, or erased subject reduction, the triad is exactly the public obligation surface for downstream consumers. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier public obligation triad`

Theorem 892.54 (MetaCIC normalization frontier scope binding)**.** *For every accepted* MetaCICNormalizationFront *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

carrier admission, the candidate rows C, K, the finished-normal rows E, N, the obstruction row S, and the structural rows H, R, P, L are scoped over the same BHist *carrier, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* NameCert *row. No row may be read as a detached host normalization or closed-consistency object.*

Proof. Start with carrier admission and the local NameCert obligations. Definition 892.45 places $C, K, E, N, S, H, R, P, L$ inside one accepted packet, and Definition 892.2 reads the same rows as the local certificate surface.

Next apply the public obligation triad. It keeps the candidate rows, the finished-normal rows, and the obstruction row on the same consumer-facing surface, while H, R, P, L provide only transport, replay, provenance, and local naming.

Finally use non-escape. Since no coordinate exports blanket strong normalization, closed consistency, erased subject reduction, or host normalization, every scoped read remains a read of the displayed BEDC packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier scope binding`

Theorem 892.55 (MetaCIC normalization frontier consumer scope exhaustion)**.** *Every closed-normal or consistency-facing consumer route for an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet chooses either the candidate surface C, K or the finished-normal surface E, N before any structural handoff through H, R, P, L. A consumer route cannot bypass both positive evidence surfaces and still remain inside the packet.*

Proof. Use consumer-route totality from Definition 892.44. It lists the candidate and finished-normal routes as the available positive evidence surfaces for downstream reads.

Then apply evidence-choice exhaustion and the candidate-finished boundary totality theorem. These results force the consumer to pick C, K or E, N before the packet is replayed, packaged, or locally named.

The bypass case is excluded by kernel non-escape. A route that avoids both positive surfaces would need a hidden blanket normalization, full closed consistency, or erased subject-reduction coordinate, and Definition 892.38 gives no such coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer scope exhaustion`

Theorem 892.56 (MetaCIC normalization frontier obstruction scope stability)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the obstruction row S remains a displayed obstruction row under componentwise transport by H and under the structural replay, provenance, and local naming rows R, P, L. Transport may carry S through the packet, but it cannot discharge S as blanket strong normalization, full closed consistency, or erased subject-reduction transport.*

Proof. Begin with obstruction retention, Definition 892.47. It keeps S as a separate row of the same finite frontier packet after the positive evidence surfaces are read.

Use the obstruction consumer lock together with the scope-binding theorem above. The obstruction row is transported and named with the same structural rows as the rest of the packet, so componentwise transport preserves it as a displayed row rather than removing it.

Finally apply non-escape. The carrier has no hidden coordinate that could turn S into a global normalization or consistency proof, so obstruction stability is exactly stability of the displayed S row. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obstruction scope stability`

Theorem 892.57 (MetaCIC normalization frontier normal endpoint exhaustion)**.** *Every normal-endpoint consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

factors through the candidate rows C, K, the finished-normal rows E, N, and the structural handoff rows H, R, P, L before reading the normal endpoint row N. The consumer cannot bypass the obstruction row S, and no read of N exports blanket strong normalization, full closed consistency, or erased subject-reduction transport.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierNormalEndpointExhaustio`

Proof. Begin with the candidate route and finished route. Definition 892.39 routes candidate-facing reads through C, K, R, P, L, while Definition 892.40 routes endpoint reads through E, N, R, P, L.

Next use consumer-route totality and the root normal endpoint handoff. These theorems force every normal-endpoint consumer onto the displayed positive surface before N is available, and they keep H, R, P, L as handoff rows rather than independent normalization evidence.

Apply the obstruction ledger. The row S remains a separate coordinate of the same packet and is not removed by the candidate route, the finished route, or the normal endpoint handoff.

Finally invoke non-escape. A bypass of S or a free-standing read of N would require a hidden blanket normalization, closed-consistency, or erased subject-reduction coordinate, and the packet has none. □

Theorem 892.58 (MetaCIC normalization frontier consistency consumer factorization)**.** *Every closed-normal consistency consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet factors through either the candidate route C, K, R, P, L or the finished route E, N, R, P, L before reading the normal endpoint row N. The packet has no third consistency-facing route outside those two displayed surfaces.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consistency consumer factorization`

Proof. Start from consumer-route totality. It lists the candidate route and the finished route as the complete positive evidence alternatives available to a consumer of the accepted packet.

Use the candidate-finished boundary totality theorem together with consumer scope exhaustion. A closed-normal consistency consumer must choose the candidate surface or the finished-normal surface before any structural replay, provenance, or local naming row can be read as part of the route.

The normal endpoint row is then controlled by the root normal endpoint handoff and the scope-binding theorem. These results keep N inside the same BHist, hsame, Cont, Pkg, and NameCert packet instead of exposing it as an independent consistency witness.

Finally apply kernel non-escape and the obstruction consumer lock. They rule out a hidden third route and keep the obstruction row S visible alongside whichever positive surface the consumer selected. □

Theorem 892.59 (MetaCIC normalization frontier subject obstruction preservation)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, the subject-reduction obstruction row S is preserved across both the candidate consumer route C, K, R, P, L and the finished-normal consumer route E, N, R, P, L. Positive normalization evidence may expose candidate rows, finished-normal rows, and the normal endpoint row N, but it does not discharge S.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier subject obstruction preservation`

Proof. Read the obstruction ledger and obstruction retention theorems. They keep S as a named row of the same packet and state that neither candidate evidence nor finished-normal evidence erases it.

Next apply the candidate route and the finished route. These are the two positive consumer routes exposed by the packet, and both pass through the same structural rows R, P, L without adding a discharge coordinate for S.

Use consumer-route totality to exhaust the positive alternatives. Since every consumer route chooses C, K or E, N, preservation across those two routes is preservation across the whole positive consumer surface.

Finally invoke non-escape. Erased subject-reduction transport is not exported by the packet, so positive normalization evidence cannot turn the displayed obstruction row into a solved subject-reduction theorem. □

Theorem 892.60 (MetaCIC normalization frontier root handoff scope)**.** *Every root handoff from an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet to a closed-normal consumer exposes exactly the carrier rows*

$$C, K, E, N, S, H, R, P, L$$

through the displayed BHist *history, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *certificate surface. No handoff row is available outside those five BEDC access layers.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootHandoffScope`

Proof. Start with the carrier and NameCert obligations. The carrier theorem names the nine rows of the packet, while the NameCert obligation theorem attaches the local naming certificate to the same rows rather than to an external normalization witness.

Next use the root normal endpoint handoff and consumer-route totality. These results force a closed-normal consumer to read the packet through the displayed history and transport rows before any endpoint evidence can be consumed.

Finally apply consistency-consumer factorization and subject-obstruction preservation. The positive route chosen by the consumer and the retained obstruction row are both still rows of the same BHist, hsame, Cont, Pkg, and NameCert surface, so the handoff exposes no further coordinate. □

Theorem 892.61 (MetaCIC normalization frontier downstream boundary)**.** *Every downstream normalization consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet must choose either the candidate route* C, K, R, P, L *or the finished-normal route* E, N, R, P, L*. In both cases the subject-reduction obstruction row* S *remains a displayed boundary row of the same packet.*

Proof. Apply consumer-route totality together with consistency-consumer factorization. These theorems list the candidate and finished-normal routes as the complete positive alternatives for a downstream normalization consumer.

For the candidate side, use the candidate route theorem. It exposes C, K together with the structural rows R, P, L, and it does not read an independent endpoint proof.

For the finished-normal side, use the finished route and normal endpoint exhaustion theorems. They expose E, N through the same structural rows R, P, L and keep the normal endpoint inside the packet.

It remains to control the obstruction row. Subject-obstruction preservation states that both positive routes retain S, while non-escape rules out any consumer route that turns S into an erased subject-reduction proof. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier downstream normalization boundary`

Theorem 892.62 (MetaCIC normalization frontier open-problem ledger unblock)**.** *The accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet unblocks the MetaCIC open-frontier ledger only as the finite boundary object*

$$(C, K, E, N, S, H, R, P, L).$$

It does not assert blanket normalization, closed consistency, or discharged subject reduction beyond the displayed candidate route, finished-normal route, obstruction row, and BEDC handoff surface.

Proof. Read the carrier theorem, NameCert obligations, and root handoff scope. They provide a finite packet with named rows and a local certificate interface for root consumers.

Next apply downstream boundary and consistency-consumer factorization. Any normalization or consistency consumer must stay on the candidate route C, K, R, P, L or the finished-normal route E, N, R, P, L.

Finally use subject-obstruction preservation and non-escape. The obstruction row S remains visible, and the packet has no row that exports blanket normalization, closed consistency, or erased subject-reduction transport. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontier open problem ledger unblock`

Theorem 892.63 (MetaCIC normalization frontier audit seal consumer route)**.** *Every MetaCICNormalizationAud facing consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet reads the same packet* $F = (C, K, E, N, S, H, R, P, L)$ *through the candidate rows* C, K*, the finished-normal rows* E, N*, the obstruction row* S*, the replay row* R*, the provenance row* P*, and the local naming row* L*. No such consumer obtains a blanket normalization source outside that displayed packet.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierAuditSealConsumerRoute`

Proof. Project the carrier and NameCert obligations. They expose exactly $C, K, E, N, S, H, R, P, L$ and attach audit-facing reads to the local certificate.

Use the candidate route and the finished route. These route seal consumers through C, K, R, P, L or E, N, R, P, L, and consumer-route totality exhausts the positive alternatives.

Apply the obstruction ledger, obstruction consumer lock, and non-escape. The row S remains visible, and no packet coordinate projects a blanket normalization source. □

Theorem 892.64 (MetaCIC normalization frontier closed-normal route boundary)**.** *Every ClosedNormalRouteCer facing read of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet reaches the normal endpoint N only after the finished-normal evidence row E has been read through the displayed route E, N, R, P, L. The obstruction row S remains a visible coordinate of the same packet throughout the read.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierClosedNormalRouteBounda`

Proof. Start from the finished route and normal endpoint exhaustion. They place the read of N behind the finished-normal evidence row E and the structural handoff rows R, P, L.

Apply the NameCert obligations and consumer-route totality. The consumer reads the endpoint through the local certificate surface, not as a free-standing endpoint.

Use the obstruction ledger, obstruction consumer lock, and non-escape. They keep S visible and rule out a route that discharges the obstruction before N is read. □

Theorem 892.65 (MetaCIC normalization frontier subject-reduction obstruction compatibility)**.** *Every SubjectReductionBoundary-facing consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet treats S as a retained obstruction coordinate. Neither the candidate rows C, K nor the finished-normal evidence rows E, N discharge that obstruction.*

Proof. Project the carrier and obstruction ledger. The row S is a named coordinate of the same finite packet as the positive evidence and handoff rows.

Use the candidate route and the finished route. They expose the positive normalization surfaces C, K, R, P, L and E, N, R, P, L, but neither route contains a subject-reduction discharge coordinate.

Apply the obstruction consumer lock, consumer-route totality, and non-escape. Every SubjectReductionBounda facing read stays on a displayed packet route, so positive evidence cannot turn S into a closed subject-reduction theorem. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierSubjectReductionObstruc`

Falsifiable prediction: If the frontier packet is accepted, any closed-normal consistency consumer route factors through either the candidate-evidence row or the finished-normal row; a route that bypasses both is outside the carrier.

Independence witness: metacic-open-frontier-ledger, metacicnormalizationaudit, metacicnormalizationauditseal, closednormalroutecertificate: the new packet is not bijective to any sibling because it carries the normalization frontier as an explicit obstruction-preserving coordinate rather than an audit report, a seal, a route certificate, or a mixed open-frontier inventory.

The Lean follow-up target is `lean4/BEDC/Derived/MetaCICNormalizationFrontierUp/TasteGate.lean`, with explicit `Nontrivial MetaCICNormalizationFrontierUp` and `FieldFaithful MetaCICNormalizationFront` instances.

Closure status for NameCert$_{\text{MetaCICNormalizationFrontier}^{\uparrow}}$.

Constructive story: A MetaCICNormalizationFrontier$^{\uparrow}$ packet is generated from candidate evidence, closed-candidate evidence, finished-normal evidence, normal-endpoint use, subject-reduction obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationFrontier}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 892.1, Definition 892.2, Definition 892.6, Definition 892.45, Definition 892.46, Definition 892.47, Definition 892.17, Definition 892.18, Definition 892.19, and the TasteGate marker `BEDC.Derived.MetaCICNormalizationFronti`

at Definition 892.1 bind the finite obstruction-preserving normalization frontier packet, the NameCert obligations, non-escape, carrier-admission route, evidence-choice exhaustion, obstruction-retention rows, obligation carrier totality, obligation evidence split, obligation obstruction retention, and singleton-carrier alignment site.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.MetaCICNormalizationFrontierUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close blanket strong normalization for all empty-context typed closed MetaCIC terms, full closed consistency, erased subject-reduction transport, or a public standard bridge.

Upgrade path: No further paper closure grade remains; future work is formal-status and bridge verification for the same obstruction-preserving normalization frontier packet under `BEDC.Derived.MetaCICNormalizationFrontierUp`.

Theorem 892.66 (MetaCIC normalization frontier candidate obstruction separation)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

candidate evidence through C, K *remains separate from the obstruction row* S*. A consumer may transport candidate evidence through* H, R, P, L*, but no candidate read discharges, hides, or rewrites* S *as finished normalization or erased subject reduction.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate obstruction separation`

Proof. Project the carrier rows of Definition 892.1. The candidate rows C, K and the obstruction row S are separate coordinates before the structural rows H, R, P, L are attached.

Apply the NameCert obligations and classifier stability. The candidate boundary clause reads C, K, while the obstruction-separation clause keeps S as its own row under the same classifier surface.

Use ledger exhaustion and non-escape. The ledger has no coordinate for erased subject reduction or blanket normalization, so transporting candidate evidence cannot remove the displayed obstruction row. □

Theorem 892.67 (MetaCIC normalization frontier closed-normal consumer route)**.** *Every closed-normal consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet that uses a finished-normal endpoint must route through the displayed rows* E, N, H, R, P, L*. The endpoint row* N *is available only after the finished-normal evidence row* E *and only under componentwise transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier closed normal consumer route`

Proof. Read the carrier $F = (C, K, E, N, S, H, R, P, L)$ from Definition 892.1. The only finished-normal evidence rows are E and N, while H, R, P, L provide the structural route by which a consumer may read them.

Apply the finished route, fuel route, and closed-normal handoff surfaces. They place the normal endpoint downstream of E and keep the consumer route inside the displayed hsame, Cont, Pkg, and NameCert rows.

Finally use the redex boundary and non-escape theorem. A consumer that bypasses E, N, H, R, P, L would require a blanket normalization coordinate or a host closed-normal endpoint, neither of which belongs to the packet. □

Theorem 892.68 (MetaCIC normalization frontier fuel ledger nonescape)**.** *The finite fuel ledger of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet forbids three exports: unbounded normalization, erased subject reduction, and closed consistency. The packet may expose candidate evidence through C, K, finished-normal evidence through E, N, obstruction retention through S, and structural route data through H, R, P, L, but it exports no consumer route outside those finite rows.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier fuel ledger nonescape`

Proof. Begin with the fuel route and ledger exhaustion results. They bind every accepted normalization read to the finite packet rows $C, K, E, N, S, H, R, P, L$ and do not introduce an unbounded normalization source.

Next use obstruction retention and the redex boundary. The row S records the subject-reduction obstruction as a retained boundary, while the redex boundary keeps candidate and finished-normal evidence within the inspected finite route.

Finally apply non-escape. Since the carrier has no row for unbounded normalization, erased subject reduction, or closed consistency, the finite fuel ledger cannot export any of those claims to downstream consumers. □

Theorem 892.69 (MetaCIC normalization frontier candidate evidence noncollapse)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

candidate evidence through C, K cannot be identified with finished-normal evidence through E, N by host equality, quotient rows, or hidden transport. Any accepted comparison of those surfaces must keep the candidate rows, the finished-normal rows, the obstruction row S, the componentwise hsame *row H, the displayed* Cont *row R, the* Pkg *provenance row P, and the local* NameCert *row L visible in the same* BHist *packet.*

Proof. Project the carrier of Definition 892.1. It exposes C, K as the candidate side and E, N as the finished-normal side of the same finite BHist packet.

Use Definition 892.2 and Definition 892.4. These results allow componentwise hsame transport along the displayed rows, but they do not provide a coordinate that rewrites candidate evidence as finished-normal evidence.

It remains to exclude external identifications. The root non-escape and boundary rows, Definition 892.10, Definition 892.66, and Definition 892.68, rule out host normalization, quotient term equality, erased subject-reduction transport, and a hidden closed-normal endpoint. Thus candidate evidence and finished-normal evidence remain distinct displayed surfaces. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier candidate evidence noncollapse`

Theorem 892.70 (MetaCIC normalization frontier finished-normal boundary)**.** *Every finished-normal read of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must pass through the displayed* BHist *carrier, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. Concretely, endpoint use reads E, N, H, R, P, L inside*

$$F = (C, K, E, N, S, H, R, P, L),$$

with the obstruction row S retained as a visible boundary coordinate.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier finished normal boundary`

Proof. Begin with Definition 892.9. A root consumer first receives exactly $C, K, E, N, S, H, R, P, L$, so there is no finished-normal endpoint outside the displayed carrier.

The finished-normal route is then fixed by Definition 892.12 and Definition 892.67. These rows place endpoint use after E and N, with replay through R, provenance through P, and local naming through L.

Finally apply the obstruction-preserving rows. Definition 892.13 and Definition 892.66 keep S as an obstruction coordinate under the same BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 892.71 (MetaCIC normalization frontier consumer route exhaustion)**.** *Every closed-normal, consistency-facing, or SubjectReductionBoundary-facing consumer route of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet factors through the candidate-evidence boundary C, K, the finished-normal boundary E, N, the obstruction row S, the componentwise transport row H, the replay row R, the provenance row P, and the local* NameCert *row L of*

$$F = (C, K, E, N, S, H, R, P, L).$$

There is no accepted consumer route outside the candidate-evidence boundary, finished-normal boundary, obstruction row, transport row, route row, provenance row, and local naming row.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier consumer route exhaustion`

Proof. First exhaust the carrier. Definition 892.9 and Definition 892.17 identify the root-facing packet with the finite rows $C, K, E, N, S, H, R, P, L$.

Next exhaust the positive evidence routes. The candidate side is controlled by Definition 892.11, while the finished-normal side is controlled by Definition 892.12 and Definition 892.70.

It remains to bind the consumer handoff. Use Definition 892.16, Definition 892.66, and Definition 892.68. These rows keep the obstruction, transport, replay, provenance, and local naming coordinates visible, so a consumer cannot obtain a hidden route beyond the displayed frontier packet. □

Theorem 892.72 (MetaCIC normalization frontier root normalization gate exhaustion)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

every root normalization-facing read is exhausted by the candidate rows C, K, the normal-endpoint rows E, N, the obstruction row S, the componentwise transport row H, the replay row R, the provenance row P, and the local name row L. In particular, a root reader cannot obtain normalization evidence without choosing the displayed candidate route or the displayed endpoint route while retaining the obstruction coordinate.

Proof. Project the root carrier rows of Definition 892.9. The root-facing packet displays exactly $C, K, E, N, S, H, R, P, L$, so any normalization-facing read starts inside the same finite BHist carrier.

The positive entrances are already exhausted. The candidate entrance is the projection through Definition 892.11, and the endpoint entrance is the route described by Definition 892.12.

It remains to keep the structural rows visible. Apply Definition 892.13 and Definition 892.71. They retain S, H, R, P, L alongside both positive routes, so no root normalization-facing consumer reads a concealed route outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootNormalizationGateEx`
Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root normalization gate exhaustion`

Theorem 892.73 (MetaCIC normalization frontier root obstruction stability)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

the row S remains a transported obstruction coordinate under both the candidate route C, K, H, R, P, L and the endpoint route E, N, H, R, P, L. It is never consumed as positive normalization evidence, and it cannot be rewritten as a closed-normal endpoint, a consistency witness, or erased subject reduction.

Proof. Begin with the carrier and separation rows. Definition 892.9 exposes S as its own coordinate, while Definition 892.66 prevents the candidate rows C, K from discharging it.

Now follow the endpoint route. Definition 892.12 routes endpoint use through E, N, H, R, P, L, and Definition 892.13 keeps S under the same BHist, hsame, Cont, Pkg, and NameCert surface.

The consumer side adds no positive use of S. By Definition 892.71, downstream readers keep the obstruction row visible rather than consuming it as normalization evidence. Thus S is stable under both displayed routes. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootObstructionStabilit`

Theorem 892.74 (MetaCIC normalization frontier root consumer choice boundary)**.** *Every closed-normal, consistency-facing, or SubjectReductionBoundary-facing consumer of an accepted* MetaCICNormalizationFro *packet has only the displayed candidate route through C, K, H, R, P, L or the displayed endpoint route through E, N, H, R, P, L, with S retained as the obstruction coordinate of the same packet. There is no third root route for such a consumer.*

Proof. First exhaust root admission. The carrier theorem Definition 892.9 supplies only the rows $C, K, E, N, S, H, R, P, L$.

Next split the allowed consumer choices. Candidate use is fixed by Definition 892.11; endpoint use is fixed by Definition 892.12 and Definition 892.16.

Finally apply the obstruction and consumer exhaustion rows. Definition 892.13 keeps S transported but unconsumed, and Definition 892.71 rules out any consumer route outside the displayed candidate and endpoint surfaces. Therefore closed-normal, consistency-facing, and SubjectReductionBoundary-facing consumers have no third root choice. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier root consumer choice boundary`

Theorem 892.75 (MetaCIC normalization frontier carrier admission)**.** *Every accepted* MetaCICNormalizationFront *packet admits a root carrier only as the displayed finite* BHist *packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The admission exposes candidate evidence through C, K, the normal-form boundary through E, N, the audit route through K, R, the seal route through S, H, the closed-normal route through E, N, R, P, L, componentwise hsame *transports through H,* Cont *replay through R,* Pkg *provenance through P, and local* NameCert *rows through L. No accepted carrier admission can hide any of those rows or replace them by a host normalization theorem.*

Proof. Start with the root carrier rows of Definition 892.9. They expose exactly the finite BHist packet $C, K, E, N, S, H, R, P, L$ and give the root reader no carrier coordinate outside that packet.

Next read the boundary and route clauses. Definition 892.11 fixes candidate evidence through C, K, while Definition 892.12 and Definition 892.16 fix the normal-form and closed-normal routes through E, N, R, P, L.

It remains to keep the structural witnesses visible. Apply Definition 892.13, Definition 892.42, and Definition 892.71. These rows retain the seal, hsame transport, Cont replay, Pkg provenance, and local NameCert surface, so carrier admission displays all kernel-facing rows without exporting host normalization. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrierAdmission`

Theorem 892.76 (MetaCIC normalization frontier normal-form boundary)**.** *No consumer of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet can read finished normalization before both the candidate-evidence boundary rows C, K and the finished-normal boundary rows E, N are displayed inside the same packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

Any normal-form read must keep the obstruction row S, componentwise hsame *transports H,* Cont *replay R,* Pkg *provenance P, and local* NameCert *row L visible alongside those two boundary surfaces.*

Proof. Project the candidate side first. Definition 892.32 and Definition 892.69 force candidate evidence through C, K and prevent it from being identified with finished-normal evidence by host equality or quotient rows.

Now project the finished side. Definition 892.70 places every finished-normal read inside E, N, H, R, P, L, and Definition 892.34 excludes host normalization, quotient term equality, and hidden normalizers outside the displayed packet.

Finally apply the consumer route theorem Definition 892.71. Since every closed-normal, consistency-facing, or SubjectReductionBoundary-facing consumer route factors through $C, K, E, N, S, H, R, P, L$, no consumer can read finished normalization until both the candidate-evidence and finished-normal boundary rows are displayed together. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierNormalFormBoundary`

Theorem 892.77 (MetaCIC normalization frontier substitution stability)**.** *Substitution-facing transports for an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet preserve the displayed frontier rows $C, K, E, N, S, H, R, P, L$ under componentwise* hsame, Cont *replay,* Pkg *provenance, and local* NameCert *naming. Such transports may carry candidate evidence, normal-form boundary evidence, the seal row, and consumer routes across the packet, but they do not create a host normalization theorem, erased subject-reduction transport, or quotient-level term equality.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier substitution stability`

Proof. Begin with classifier stability. Definition 892.4 permits componentwise hsame transport only along the displayed carrier rows, and Definition 892.73 keeps the obstruction coordinate S transported but unconsumed.

Next follow the consumer and route surfaces. Definition 892.72 exhausts every root normalization-facing read by $C, K, E, N, S, H, R, P, L$, while Definition 892.71 keeps downstream routes inside the same BHist, Cont, Pkg, and NameCert packet.

The forbidden exports are excluded by the sibling non-escape rows. Use Definition 892.38, Definition 892.68, and Definition 892.34. They rule out blanket normalization, erased subject reduction, quotient term equality, and hidden normalizers, so substitution-facing transport preserves the frontier rows without producing a host normalization theorem. □

Theorem 892.78 (MetaCIC normalization frontier closed-candidate readback exhaustion)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

closed-candidate readback is exhausted by the candidate evidence row C*, the closed-candidate row* K*, displayed replay* R*, provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$ *row* L*. A root reader cannot obtain closed-candidate evidence from* E, N, S, H *or from any host normalization, scheduler, quotient-equality, or closed-consistency surface outside the displayed packet.*

Proof. Start with the NameCert obligation rows of Definition 892.2. They attach candidate evidence to the displayed carrier and separate it from finished-normal evidence, obstruction transport, and external host normalization.

Next project the root candidate gate. Theorem 892.11 identifies the candidate read with C, K before any replay, provenance, or local naming support can be used.

Finally add the consumer support rows. Theorem 892.12 keeps endpoint use on the distinct E, N, R, P, L route, while Theorem 892.15 leaves S visible but unavailable as positive candidate evidence. Thus closed-candidate readback is exhausted by C, K, R, P, L and by no unlisted source. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier closed candidate readback exhaustion`

Theorem 892.79 (MetaCIC normalization frontier finished-endpoint use totality)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

finished endpoint use is total exactly over the finished-normal evidence row E*, the normal-endpoint row* N*, displayed replay* R*, provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$ *row* L*. The endpoint route may consult the candidate gate only through the already displayed* C, K *boundary, and it has no bypass through obstruction transport, host reachability, scheduler progress, or erased subject-reduction transport.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier finished endpoint use totality`

Proof. First read the endpoint-facing obligations. Theorem 892.2 places finished-normal evidence before endpoint use and keeps both rows under the same local certificate surface.

Now follow the root endpoint route. Theorem 892.12 exhausts endpoint access by E, N, R, P, L, so no finished endpoint can be consumed before those support rows are visible.

It remains to rule out hidden positive sources. Theorem 892.11 confines candidate consultation to C, K, and Theorem 892.15 keeps S transported rather than consumed. Hence endpoint use is total over E, N, R, P, L without importing host reachability, scheduler progress, or erased subject-reduction transport. □

Theorem 892.80 (MetaCIC normalization frontier obstruction-visible transport)**.** *For every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L),$$

transport of the obstruction row keeps S visible under componentwise hsame *transport H, displayed replay R, provenance P, and the local naming row L. Such transport can accompany candidate readback through C, K and finished-endpoint use through E, N, R, P, L, but it cannot turn S into positive normalization evidence, closed consistency, quotient-level term equality, or erased subject-reduction transport.*

Proof. Begin with the local obligation surface. Theorem 892.2 records S as an obstruction row of the same packet that carries C, K, E, N, H, R, P, L.

Transport then follows the root separation rows. Theorem 892.15 keeps S alongside the candidate route through C, K and the endpoint route through E, N, R, P, L.

Finally use the two positive route projections. Theorem 892.11 gives the only closed-candidate readback route, and Theorem 892.12 gives the only finished-endpoint route. Since neither route consumes S, obstruction-visible transport under H, R, P, L cannot become positive normalization evidence or any of the excluded global objects. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier obstruction visible transport`

Theorem 892.81 (MetaCIC normalization frontier route readback public export)**.** *Every public readback export of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

factors through the closed-candidate route C, K, R, P, L, the finished endpoint route E, N, R, P, L, and obstruction-visible transport S, H, R, P, L. The exported surface has no route through host normalization, scheduler progress, quotient equality, a closed-consistency shortcut, or erased subject-reduction transport.

Proof. First apply Definition 892.78. It identifies the closed-candidate public readback route as C, K, R, P, L and rules out host normalization, scheduler, quotient-equality, and closed-consistency sources for that face.

Next apply Definition 892.79. The finished endpoint face is total only over E, N, R, P, L, with candidate consultation confined to the already displayed C, K boundary.

Finally apply Definition 892.80. The obstruction row S remains visible only under H, R, P, L and cannot become positive normalization evidence, quotient equality, closed consistency, or erased subject-reduction transport. Combining the three route rows gives the whole public export surface. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier route readback public export`

Theorem 892.82 (MetaCIC normalization frontier root formal handoff totality)**.** *Every root formal handoff of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet is total only over the finite root packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The rows $C, K, E, N, S, H, R, P, L$ bind candidate evidence, the candidate gate, finished-normal evidence, the normal endpoint, the obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$ *row to the existing carrier. The handoff exports no scheduler state, host reachability predicate, closed-consistency theorem, erased subject-reduction transport, or hidden normalization source outside that carrier.*

Proof. Begin with the displayed root carrier rows. Theorem 892.9 and Theorem 892.75 expose exactly $C, K, E, N, S, H, R, P, L$, so a formal handoff has no additional coordinate from which to read normalization content.

Next project the positive entrances. Theorem 892.11 binds candidate evidence to C, K, while Theorem 892.12 routes normal-endpoint use through E, N, R, P, L.

Finally retain the obstruction and consumer rows. Theorem 892.13, Theorem 892.72, and Theorem 892.71 keep S, H, R, P, L visible for downstream reads. Thus the handoff is total for the displayed carrier and for no hidden host, scheduler, consistency, or erased subject-reduction input. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootFormalHandoffTotali`

Theorem 892.83 (MetaCIC normalization frontier root gate scope exactness)**.** *For every accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet $F = (C, K, E, N, S, H, R, P, L)$, the root candidate gate has exact scope C, K inside the same carrier and can feed a normal-endpoint read only through the displayed endpoint route E, N, R, P, L with obstruction retention through S, H, L. It cannot be widened to closed consistency, host reachability, scheduler progress, or erased subject-reduction transport.*

Proof. Project the candidate side first. Theorem 892.11 identifies the candidate gate with C, K, and Theorem 892.66 keeps the obstruction row S outside that positive candidate evidence.

Now read the endpoint route. Theorem 892.12 and Theorem 892.16 place endpoint use under E, N, R, P, L, rather than under an external host theorem or scheduler row.

The scope is exact because the gate-exhaustion and obstruction-preservation rows add no third positive source. Apply Theorem 892.13 and Theorem 892.72: they preserve S, H, R, P, L while excluding hidden closed consistency, host reachability, scheduler progress, and erased subject-reduction transport. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootGateScopeExactness`

Theorem 892.84 (MetaCIC normalization frontier root downstream consumer coverage)**.** *Every downstream MetaCIC consumer of an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *root packet that reads candidate evidence, normal endpoint data, obstruction status, replay, provenance, or local naming factors through*

$$C, K, E, N, S, H, R, P, L.$$

The consumer coverage includes the existing carrier, candidate gate, normal endpoint route, obstruction-preserving NameCert triad, root normalization gate, and local NameCert rows; it excludes hidden scheduler state, host reachability, closed consistency, erased subject-reduction transport, and any route outside the displayed consumer packet.

Proof. First take the carrier and candidate gate as the consumer source. Theorem 892.9 supplies the packet $C, K, E, N, S, H, R, P, L$, and Theorem 892.11 fixes candidate reads through C, K.

Next account for normal endpoints and obstruction retention. Theorem 892.12 routes endpoint reads through E, N, R, P, L, while Theorem 892.13 keeps S, H, L visible under the same local naming surface.

Finally apply route exhaustion. Theorem 892.72 exhausts root normalization-facing reads, and Theorem 892.71 exhausts downstream consumer routes. Therefore every downstream consumer factors through the displayed rows and none may import scheduler state, host reachability, closed consistency, erased subject-reduction transport, or an unlisted route. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierRootDownstreamConsumer(`

Theorem 892.85 (MetaCIC normalization frontier formal-target carrier coverage)**.** *For every formal target surface attached to an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet, carrier coverage is exactly the finite packet*

$$F = (C, K, E, N, S, H, R, P, L).$$

The rows C, K carry candidate evidence and closed-candidate placement, the rows E, N carry finished-normal evidence and the normal endpoint, the row S retains the subject-reduction obstruction, and the rows H, R, P, L expose componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFrontier}^{\uparrow}}$ *surface. No formal target row may read a carrier coordinate outside that packet.*

Proof. Project the carrier theorem and the NameCert obligation theorem. They list $C, K, E, N, S, H, R, P, L$ as the whole BHist carrier and attach the local certificate to those rows.

Next read the root formal handoff totality theorem. It makes the formal handoff total only over the displayed rows, with candidate, endpoint, obstruction, replay, provenance, and naming all preserved inside the same packet.

Finally apply non-escape. A carrier coordinate outside F would require a hidden normalization source, scheduler state, closed-consistency theorem, or erased subject-reduction transport, and none is exported by the packet. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier formal target carrier coverage`

Theorem 892.86 (MetaCIC normalization frontier formal-target classifier totality)**.** *For every formal target surface attached to an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *packet, classifier totality is exhausted by the candidate route through C, K, R, P, L and the finished-normal route through E, N, R, P, L, with obstruction retention through S, H, L. The classifier cannot add a third positive normalization route, cannot discharge S, and cannot replace the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows by host evidence.*

Proof. Start with evidence-choice exhaustion. Every positive consumer route chooses either C, K or E, N, and the structural rows R, P, L only replay, package, and name the chosen route.

Use obstruction retention next. The row S remains present across both positive routes, while H transports the displayed packet componentwise rather than solving the obstruction.

The classifier is therefore total on exactly those two routes. Root gate scope exactness and downstream consumer coverage rule out a third route through scheduler progress, host reachability, closed consistency, or erased subject-reduction transport. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCarrier formal target classifier totality`

Theorem 892.87 (MetaCIC normalization frontier formal-target ledger nonescape)**.** *For every formal target surface attached to an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet, any ledger read of candidate evidence, finished-normal evidence, normal endpoint use, obstruction status, replay, provenance, or local naming remains inside*

$$C, K, E, N, S, H, R, P, L.$$

A ledger row that omits obstruction retention, bypasses the evidence choice, or replaces BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *by hidden host material is outside the formal target surface.*

Proof. Read ledger exhaustion together with the public obligation package. They place candidate reads at C, K, finished-normal reads at E, N, obstruction accounting at S, and consumer support at H, R, P, L.

Now use the root normal endpoint route and root downstream consumer coverage. These theorems keep endpoint use and downstream consumption inside the same displayed packet before any formal target surface can consume them.

Finally apply the kernel non-escape theorem. A ledger read that bypasses the displayed rows asks for an unlisted coordinate, so it is rejected rather than admitted as part of the formal target surface. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierFormalTargetLedgerNones`

Theorem 892.88 (MetaCIC normalization frontier candidate endpoint premise route)**.** *Every finished-normal endpoint read attached to an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *formal target must expose Candidate, ClosedCandidate, or CandidateSet evidence and the frontier audit row before any confluence, decidability, or closed-normal consumer can inspect the endpoint. The route is therefore conditional on the displayed candidate surface and never supplies unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, or host evaluator soundness.*

Proof. Start with carrier coverage from Definition 892.85. The formal target surface is the finite packet $C, K, E, N, S, H, R, P, L$, where the candidate and closed-candidate rows are visible before the endpoint row is exported.

Use classifier totality from Definition 892.86. A consumer chooses the candidate route through C, K, R, P, L or the finished-normal route through E, N, R, P, L, while the obstruction and audit material remains visible through S, H, L.

Finally apply ledger nonescape from Definition 892.87. Any confluence, decidability, or closed-normal consumer must stay inside the displayed candidate, endpoint, obstruction, replay, provenance, and naming rows, so the excluded host principles and unconditional normalization claims have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCandidateEndpointPremis`

Theorem 892.89 (MetaCIC normalization frontier ground-compiler route)**.** *Every accepted* MetaCICNormalization *candidate-evidence boundary routes its TasteGate-facing conservativity and no-hidden-input obligations through the displayed ground-compiler recognition rows before any normalization boundary is exported. Concretely, a consumer of*

$$F = (C, K, E, N, S, H, R, P, L)$$

must first read the carrier surface of Definition 892.1, then the candidate-evidence route of Definition 892.32, and then the root normal-form boundary of Definition 892.76. Only after those

rows are visible may the consumer use the TasteGate row certified at the core surface, and that use exports no hidden normalizer, host normalization theorem, erased subject-reduction transport, or quotient-level term equality.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierGroundCompilerRoute`

Proof. Project the core carrier. The rows $C, K, E, N, S, H, R, P, L$ are the only accepted frontier coordinates, and the core surface records the TasteGate-facing packet through the same finite carrier rather than through an external normalization oracle.

Next read the candidate-evidence boundary. The route through C, K is fixed by Definition 892.32, while Definition 892.33 keeps the seal and audit-facing rows visible before closed-normal use.

The root normal-form boundary then forces finished-normal reads through E, N with S, H, R, P, L still displayed. Thus the conservativity and no-hidden-input obligations face the ground-compiler recognition rows before normalization is exported.

Finally apply the non-escape rows Definition 892.34 and Definition 892.77. They exclude hidden normalizers, host normalization, erased subject-reduction transport, and quotient term equality, so the TasteGate-facing route remains a ground-compiler route over the displayed packet. □

Theorem 892.90 (MetaCIC normalization frontier TasteGate route)**.** *Any promotion of* MetaCICNormalizationFron *beyond its seed and scaffold boundary must cite the finite candidate-evidence rows C, K, the finished-normal endpoint rows E, N, the subject-reduction obstruction row S, and the ground-compiler route rows H, R, P, L already displayed in this chapter. It may not replace those rows by a hand-written global normalization claim, a hidden normalizer, erased subject-reduction transport, or quotient-level term equality.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierTasteGateRoute`

Proof. Begin with the core surface Definition 892.1 and Definition 892.2. These rows expose $C, K, E, N, S, H, R, P, L$ as the finite packet and make candidate evidence, finished-normal evidence, obstruction accounting, and downstream replay separate obligations.

Next read the formal-target rows, including Definition 892.85, Definition 892.86, and Definition 892.87. They keep any promotion attached to the same carrier and classifier rows rather than to a blanket normalization assertion.

Finally use the ground-compiler route of Definition 892.89. Since this chapter has AI lineage, the roadmap discipline requires TasteGate and ground-compiler evidence before the chapter leaves seed-level closure; the displayed route supplies exactly that evidence and excludes a hand-written global normalization replacement. □

Theorem 892.91 (MetaCIC normalization frontier discharge route)**.** *The candidate evidence rows C, K, endpoint rows E, N, obstruction row S, and ground-compiler rows H, R, P, L jointly discharge any* MetaCICNormalizationFrontier$^{\uparrow}$ *normalization-frontier consumer through the displayed packet. The discharge exports neither a hidden normalizer nor a host strong-normalization theorem, erased subject-reduction transport, or quotient-level term equality.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierDischargeRoute`

Proof. Start with the TasteGate route of Definition 892.90. It requires any promotion beyond the seed and scaffold boundary to cite $C, K, E, N, S, H, R, P, L$ together, so the consumer cannot isolate candidate evidence from endpoints, obstruction accounting, or the ground-compiler rows.

Now apply the ground-compiler route of Definition 892.89. That route reads the carrier, candidate-evidence boundary, and root normal-form boundary before exporting any normalization boundary, and it states the same exclusions for hidden normalizers, host normalization, erased subject-reduction transport, and quotient-level term equality.

Finally read the formal-target rows cited in the TasteGate proof, especially carrier coverage, classifier totality, and ledger nonescape. These rows keep the discharge attached to the finite packet rather than to an ambient normalizer. Hence the listed rows jointly discharge the local frontier consumer while preserving the chapter's no-hidden-input boundary. □

Definition 892.92 (MetaCIC normalization frontier ground-compiler formal target)**.** *A ground-compiler formal target for* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *is a finite packet*

$$F = (C, K, E, N, S, H, R, P, L)$$

whose carrier rows C, K *display candidate evidence, whose endpoint rows* E, N *display finished-normal evidence, whose obstruction row* S *retains subject-reduction failure data, whose ground-compiler rows* H, R *record recognition and replay, whose provenance row* P *names the source packet, and whose local ledger* L *binds the exported normalization boundary to the same displayed rows.*

Lean4 (definition) `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierGroundCompilerF`

Theorem 892.93 (MetaCIC normalization frontier ground-compiler target scope)**.** *Any formal target for the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *normalization frontier must read candidate evidence* C, K*, finished-normal endpoint evidence* E, N*, obstruction retention* S*, and ground-compiler rows* H, R, P, L *before exporting a normalization boundary. A target that exports the boundary without those rows is outside the displayed ground-compiler packet and does not inhabit the local formal-target surface.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierGroundCompilerTargetSco`

Proof. Begin with the ground-compiler route of Definition 892.89. It requires a consumer to read the carrier surface, candidate evidence, and root normal-form boundary before any normalization boundary is exported.

Next apply the TasteGate route in Definition 892.90. It keeps every promotion beyond the seed and scaffold boundary attached to $C, K, E, N, S, H, R, P, L$ and excludes a hand-written global normalization claim or hidden normalizer.

Finally use the discharge route of Definition 892.91. The discharge theorem joins candidate evidence, endpoint evidence, obstruction retention, and ground-compiler rows into the same packet, so the formal target of Definition 892.92 has exactly the stated scope. □

Theorem 892.94 (MetaCIC normalization frontier candidate-discharge stability)**.** *Let* $F = (C, K, E, N, S, H, R, P$ *be an accepted* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *carrier whose ground-compiler route of Definition 892.89 and TasteGate route of Definition 892.90 are displayed. If* F' *is obtained from* F *by component-wise* hsame *transport of* $C, K, E, N, S, H, R, P, L$ *and by displayed* Cont *replay of the same ground-compiler recognition and discharge rows, then* F' *preserves the same candidate-mediated discharge boundary. In particular, the transported packet may export only the local candidate evidence, endpoint evidence, obstruction-retention, route, replay, provenance, and ledger rows already visible in* F*; it does not acquire strong normalization, confluence, erased subject-reduction transport, quotient-level term equality, or a hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCandidateDischargeStabi`

Proof. Start with the ground-compiler route. It makes the carrier surface, candidate-evidence boundary, and root normal-form boundary visible before any normalization boundary is exported.

Next use the TasteGate route. It requires the same rows $C, K, E, N, S, H, R, P, L$ for any promotion beyond the seed and scaffold boundary, so componentwise hsame transport can only rename displayed rows and cannot introduce a hidden normalizer or a global normalization assertion.

Now apply the discharge route and the target-scope theorem. The discharge route joins candidate evidence, endpoint evidence, obstruction retention, and ground-compiler rows inside the packet, while the target-scope theorem excludes targets that export a boundary without those rows.

Finally replay the displayed Cont route for F'. Since the replay reads exactly the transported recognition and discharge rows, the candidate-mediated boundary is preserved, and none of the excluded strong-normalization, confluence, subject-reduction, quotient-equality, or hidden-normalizer claims is added. □

Theorem 892.95 (MetaCIC normalization frontier residual-premise retention)**.** *Let $F = (C, K, E, N, S, H, R, P, L$ be an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet. Any candidate-mediated discharge that preserves the ground-compiler route must retain the obstruction row S as a visible residual premise until a separate subject-reduction row is supplied. The route may export candidate evidence, endpoint evidence, ground-compiler recognition, replay, provenance, and local ledger data, but it may not export hidden normalization, full strong normalization, confluence, erased subject reduction, quotient term equality, or a hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierResidualPremiseRetentio`

Proof. Begin with the ground-compiler route of Definition 892.89. It requires a consumer to read the carrier surface, candidate-evidence route, and root normal-form boundary before any normalization boundary is exported.

Next apply the TasteGate route of Definition 892.90. That route keeps $C, K, E, N, S, H, R, P, L$ visible together, so the obstruction row S cannot be silently erased by a promotion that still claims to use the displayed frontier packet.

Use the discharge route of Definition 892.91. It joins candidate evidence, endpoint evidence, obstruction accounting, and ground-compiler rows in one packet, and it excludes hidden normalizers, host strong-normalization theorems, erased subject-reduction transport, and quotient-level term equality.

Finally apply candidate-discharge stability from Definition 892.94. Componentwise hsame transport and displayed Cont replay preserve exactly the same boundary, so the transported discharge still retains S until an independent subject-reduction row is displayed. □

Theorem 892.96 (MetaCIC normalization frontier subject-reduction residual budget)**.** *Let $F = (C, K, E, N, S, H, R, P, L)$ be an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet whose ground-compiler route is preserved. The subject-reduction obstruction row S remains a visible residual budget row until a separate subject-reduction certificate is supplied. Candidate evidence C, K, endpoint evidence E, N, and ground-compiler rows H, R, P, L may be consumed only as their displayed rows; they cannot be retyped as subject reduction, strong normalization, confluence, quotient term equality, evaluator equality, or a hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierSubjectReductionResidua`

Proof. Begin with the ground-compiler route of Definition 892.89. It forces every normalization-boundary consumer to read the carrier surface, candidate-evidence route, and endpoint boundary before exporting any ground-compiler-facing result.

Apply the TasteGate and discharge routes of Definition 892.90 and Definition 892.91. These routes keep $C, K, E, N, S, H, R, P, L$ visible as separate rows, so the candidate and endpoint evidence cannot be promoted into subject reduction or a stronger normalization theorem.

Use candidate-discharge stability from Definition 892.94. Componentwise hsame transport and displayed Cont replay preserve the same row family, including the distinction between candidate evidence, endpoints, obstruction accounting, and ground-compiler replay.

Finally apply residual-premise retention from Definition 892.95. Since S is the only displayed subject-reduction obstruction row and no other coordinate has the required type, row exhaustion leaves S as an ordinary residual budget row until an independent subject-reduction certificate is supplied. □

Theorem 892.97 (MetaCIC normalization frontier ground-compiler readback)**.** *Let $F = (C, K, E, N, S, H, R, P, L$ be an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *carrier. Any normalization-frontier readback consumed by a MetaCIC audit or TasteGate consumer must pass through candidate evidence C, K, endpoint evidence E, N, residual premise S, and the ground-compiler rows H, R, P, L. Componentwise* hsame *transport and displayed* Cont *replay preserve that readback, and the route exports no strong normalization, confluence, erased subject reduction, quotient term equality, or hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierGroundCompilerReadback`

Proof. Begin with the ground-compiler route of Definition 892.89. It requires every normalization-boundary consumer to read the carrier surface, candidate-evidence route, and root normal-form boundary before any frontier readback is exported.

Next apply the TasteGate route and discharge route of Definition 892.90 and Definition 892.91. These rows keep $C, K, E, N, S, H, R, P, L$ visible together, so an audit-facing readback cannot separate candidate evidence from endpoints, residual obstruction accounting, recognition, replay, provenance, or the local ledger.

Use candidate-discharge stability from Definition 892.94. Componentwise hsame transport only renames the displayed rows, and displayed Cont replay replays the same recognition and discharge route for the transported packet.

Finally retain the residual premise by Definition 892.95. Since S remains visible until a separate subject-reduction row is supplied, the readback exports none of the excluded strong-normalization, confluence, erased subject-reduction, quotient-equality, or hidden-normalizer claims. □

Theorem 892.98 (MetaCIC normalization frontier formal-target contract)**.** *Any formal target consumed from an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must expose candidate evidence rows C, K, finished-normal endpoint rows E, N, residual premise S, and ground-compiler rows H, R, P, L before it may export a normalization boundary. A target is outside the contract if it drops S or replaces these rows by hidden strong normalization, confluence, erased subject reduction, quotient term equality, or a hidden normalizer.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierFormalTargetContract`

Proof. Begin with the ground-compiler route of Definition 892.89. It requires a consumer to read the carrier surface, candidate-evidence route, and root normal-form boundary before any normalization boundary is exported.

Apply the TasteGate route of Definition 892.90 and the discharge route of Definition 892.91. Together they keep $C, K, E, N, S, H, R, P, L$ visible as one packet rather than allowing a formal target to cite an ambient normalization principle.

Use candidate-discharge stability from Definition 892.94. Componentwise hsame transport and displayed Cont replay preserve the candidate-mediated discharge boundary and do not add confluence, erased subject reduction, quotient term equality, or a hidden normalizer.

Finally apply Definition 892.95 and Definition 892.97. The residual premise S remains visible until a separate subject-reduction row is supplied, and the readback exports the normalization boundary only through the displayed ground-compiler rows. Hence any target that drops S or substitutes hidden strong normalization lies outside the local contract. □

Theorem 892.99 (MetaCIC normalization frontier certificate-router handoff)**.** *Every accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet whose frontier rows name closed-normal candidates, bounded reduction windows, residual obstruction data, and classifier-stable proof certificates can be consumed by the normalization-certificate router only after those rows remain visible under* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *transport. The handoff refuses unconditional strong normalization, unrestricted confluence, and host equality.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierCertificateRouterHandof`

Proof. Project the carrier of Definition 892.1. Candidate evidence, endpoint evidence, residual obstruction, route recognition, replay, provenance, and local ledger rows are the only rows available to a router-facing consumer.

Use the ground-compiler route of Definition 892.89 and the target scope of Definition 892.93. They force every normalization-boundary read through the displayed carrier, candidate-evidence boundary, endpoint boundary, and ground-compiler rows before any certificate is exported.

Apply candidate-discharge stability, residual-premise retention, and ground-compiler readback. These theorems keep the same rows visible under componentwise hsame transport and displayed Cont replay, with Pkg provenance and the local NameCert ledger still attached to the packet.

The certificate router therefore receives only the displayed closed-normal candidate data, bounded window data, residual obstruction row, and classifier-stable proof certificate rows. Since no carrier coordinate stores unconditional strong normalization, unrestricted confluence, or host equality, the handoff cannot export those claims. □

Theorem 892.100 (MetaCIC normalization frontier L10 candidate-discharge route)**.** *Every candidate-normalization discharge read of an accepted* MetaCICNormalizationFrontier$^{\uparrow}$ *packet must display the four L10 Real and Completion source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before the frontier rows are consumed as a normalization-facing discharge. The read must also pass through the existing normalization audit and closed-normal route siblings, including chapter 750, chapter 777, chapter 858, Definition 892.89, and Definition 892.91. The route exports no full strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationFrontierUp.MetaCICNormalizationFrontierL10CandidateDischargeRo`

Proof. Start with the displayed carrier $F = (C, K, E, N, S, H, R, P, L)$. The candidate rows C, K can be consumed only through the same frontier packet that also displays endpoint evidence, obstruction retention, replay, provenance, and local naming.

Read the ground-compiler route and the discharge route. They force the normalization-facing consumer through the carrier, candidate-evidence boundary, endpoint boundary, residual obstruction, and closed-normal handoff before any discharge is public.

Now attach the L10 faces as source rows for the discharge. $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the finite tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ supplies finite observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal Real seal. These faces are source displays, not selected limits or ambient completeness principles.

Finally use the audit and closed-normal route siblings named in the statement. They keep the candidate discharge tied to finite audit, seal, and route rows, so the consumer cannot replace it by full strong normalization, global confluence, beta-conversion decidability, quotient term equality, selected limits, or Real completeness. □

Chapter 893

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFiber}^{\uparrow}$

The $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ packet records the common finite tail-fiber classifier used when two regular Cauchy source tails must feed one $\mathsf{Real}^{\uparrow}$ seal boundary. It sits between the shared-tail threshold packet of chapter 560, the tail-agreement seal of chapter 699, the finite stream windows of chapter 126, and the regular rational carrier of chapter 301. The packet is a Real-and-Completion diagonal route component; it is not a quotient of streams, a host real equality, or an ambient completeness principle.

Falsifiable prediction: If the shared tail fiber row is removed, the packet cannot factor both $\mathsf{RealTailAgreementSeal}^{\uparrow}$ and $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ consumer reads through one finite $\mathsf{Real}^{\uparrow}$ seal route.

Independence witness: regular-cauchy-tail-meet, realtailagreementseal, regularcauchytailfusion, modulus-indexed-tail-cover: these siblings expose threshold, agreement, fusion, or cover rows, while $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ exposes the common fiber classifier tying the two source tails to one $\mathsf{Real}^{\uparrow}$ seal boundary.

Definition 893.1 (Regular Cauchy tail fiber carrier)**.** *A* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose late regular-rational tails are being compared;*
2. W_0 *and* W_1 *are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the two source rows are observed;*
3. D_0 *and* D_1 *are* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-radius ledgers for the two observed tails;*
4. T *is the shared tail fiber row classifying the two source tails as one finite tail boundary;*
5. A *is the* $\mathsf{Real}^{\uparrow}$ *tail-agreement seal row reached through the shared fiber;*
6. H *records componentwise* hsame *transport for the source rows, windows, dyadic ledgers, shared fiber, seal, replay route, provenance, and local name;*
7. C *records displayed* Cont *routes for reading both source tails, both windows, both dyadic ledgers, the shared fiber row, and the seal row;*

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFiber}^{\uparrow}}$ *row.*

The carrier accepts only packets whose two regular Cauchy source tails meet in the displayed shared fiber row before any $\mathsf{Real}^{\uparrow}$ *seal is read. It contains no quotient stream, no host real equality, no selected limit, and no completion theorem beyond the finite seal boundary.*

Theorem 893.2 (Regular Cauchy tail fiber NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFiber}^{\uparrow}}$ *surface consists exactly of source-row exposure for R_0, R_1, finite-window exposure for W_0, W_1, dyadic tail-radius exposure for D_0, D_1, shared tail-fiber exposure for T,* $\mathsf{Real}^{\uparrow}$ *tail-agreement seal exposure for A, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 893.1. The first six rows expose exactly the two source tails, their finite observation windows, and their dyadic tail-radius ledgers.

Next read the common fiber row T. This is the only row that identifies the two late-tail observations as one finite tail boundary before the seal row is available.

Finally read A, H, C, P, N. The seal row is reached only after the shared fiber has been displayed, and the transport, continuation, provenance, and naming rows replay that same finite packet. Thus the local NameCert obligations are exhausted by the listed carrier coordinates. □

Theorem 893.3 (Regular Cauchy tail fiber window projection)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier factors through the finite projection*

$$(R_0, R_1, W_0, W_1, D_0, D_1, T).$$

The read cannot reach the $\mathsf{Real}^{\uparrow}$ *tail-agreement seal row A without first passing through the shared tail fiber row T and the displayed continuation route C.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberWindowProjection changes packet`

Proof. Start with a source-facing consumer. Projection exposes R_0, R_1 as the only regular-rational source rows and W_0, W_1 as the only finite stream-name windows attached to them.

The dyadic-radius ledgers D_0, D_1 are then read at those same windows. A consumer has no separate radius or tolerance row outside the displayed packet.

The seal-facing step is forced through T and C. Since A is attached to the shared fiber route, a consumer cannot bypass the common fiber and read a detached $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 893.4 (Regular Cauchy tail fiber shared seal factorization)**.** *For an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier $F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer route and the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *consumer route share the same finite factorization through T before reaching the* $\mathsf{Real}^{\uparrow}$ *seal row A.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiber shared fiber route changes packet`

Proof. Read the tail-meet side first. The meet route consumes $R_0, R_1, W_0, W_1, D_0, D_1$ and then exposes the shared threshold boundary as the fiber row T.

Read the tail-agreement side next. Its seal route uses the same T as the finite agreement boundary before the $\mathsf{Real}^{\uparrow}$ seal row A is available.

The administrative rows H, C, P, N transport, replay, package, and name this one route. Since no second fiber coordinate is present, both consumer routes factor through the same shared row. □

Theorem 893.5 (Regular Cauchy tail fiber non-escape)**.** *Every downstream read of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet remains inside*

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

It exports no quotient stream equality, no host real equality, no selected limit, no countable-choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. The only source, window, radius, common-fiber, and seal rows are the coordinates listed in Definition 893.1.

Use the window projection and shared seal factorization. Any consumer first reads the two finite windows and dyadic-radius ledgers, then reaches the common fiber T, and only then reaches the seal row A.

Inspect the excluded data. No coordinate stores quotient stream equality, host real equality, a selected limit, choice, or ambient completeness. The packet therefore cannot export such data to a downstream Real-and-Completion route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket nonescape`

Theorem 893.6 (Regular Cauchy tail fiber follow-up TasteGate contract)**.** *The Lean TasteGate follow-up for* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *must provide* $\mathsf{Nontrivial}\ \mathsf{RegularCauchyTailFiber}^{\uparrow}$ *and* $\mathsf{FieldFaithful}\ \mathsf{RegularCauchyTailFiber}^{\uparrow}$ *instances whose witnesses include an explicit* $\mathsf{witness_pair}$ *and constructor-field injection proofs for the two source rows, two windows, two dyadic-radius ledgers, shared fiber, seal row, transport, continuation, provenance, and local naming fields.*

Proof. The carrier has twelve displayed fields, and each field is observable by a NameCert obligation or by the non-escape boundary. A nontrivial instance must therefore exhibit two distinguishable accepted packets through a concrete witness pair rather than by host equality.

Field faithfulness must then inspect constructor fields. If two accepted packets agree under the public classifier, the proof has to inject through the source rows, windows, dyadic ledgers, fiber row, seal row, and administrative rows separately.

These are precisely the fields used by the window projection, shared seal factorization, and non-escape theorems. The TasteGate follow-up is therefore a fieldwise carrier proof for this packet, not a quotient-stream or host-real argument. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberTasteGate single carrier alignment`

Theorem 893.7 (Regular Cauchy tail fiber classifier stability)**.** *For an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

componentwise hsame *transport through the row H preserves the common fiber classifier on $R_0, R_1, W_0, W_1, D_0, D_1,$ In particular, a transported read still classifies the two regular Cauchy source tails through the same finite row T before reaching the* $\mathsf{Real}^{\uparrow}$ *seal row A, and the transport introduces neither quotient-stream equality nor host-real equality.*

Proof. Project the carrier of Definition 893.1. The row H is displayed as componentwise hsame transport for the two source rows, the two windows, the two dyadic ledgers, the shared fiber row, and the seal row.

Apply the window projection of Definition 893.3. Any source-facing read reaches $R_0, R_1, W_0, W_1, D_0, D_1$ before it can see a seal-facing row, so the transported classifier has the same finite source and window support.

Apply the shared seal factorization of Definition 893.4. The tail-meet route and the tail-agreement route both pass through the same row T before A is available.

Finally use non-escape from Definition 893.5. Since the accepted packet contains no quotient-stream equality and no host-real equality, componentwise transport along H cannot export either kind of equality. □

Theorem 893.8 (Regular Cauchy tail fiber paired-window exactness)**.** *For an accepted* RegularCauchyTailFiber$^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

every RegularCauchyTailMeet$^{\uparrow}$ *read and every* RealTailAgreementSeal$^{\uparrow}$ *read factors through the same finite tuple*

$$(R_0, R_1, W_0, W_1, D_0, D_1, T).$$

Thus the BEDC correspondence between the regular-tail meet route and the real-tail agreement route is witnessed by one paired-window tuple, not by an external equality of streams or host reals.

Proof. Start with a RegularCauchyTailMeet$^{\uparrow}$ consumer. By Definition 893.3, its source and window observations are exactly the two regular-rational rows, the two finite stream windows, and the two dyadic ledgers, followed by the shared fiber row T.

Now read a RealTailAgreementSeal$^{\uparrow}$ consumer. By Definition 893.4, the seal route uses the same shared fiber row T before it reaches the seal row A.

The two reads therefore have identical finite support up to T. The remaining rows H, C, P, N transport, replay, package, and name that support, so the correspondence is the displayed BEDC route through the paired-window tuple. □

Theorem 893.9 (Regular Cauchy tail fiber seal-consumer exhaustion)**.** *For an accepted* RegularCauchyTailFiber$^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

all downstream reads by RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegularCauchyTailMeet$^{\uparrow}$*,* RealTailAgreementSe *and* Real$^{\uparrow}$ *consumers remain inside*

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

No listed consumer can read a detached source tail, detached stream window, detached dyadic radius, second fiber row, external seal, quotient-stream witness, host-real witness, selected limit, or ambient completeness theorem.

Proof. Use the carrier projection and the NameCert obligations of Definition 893.2. The only exposed rows are the two source rows, two windows, two dyadic ledgers, shared fiber, seal, transport, continuation, provenance, and local name.

For RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ consumers, apply Definition 893.3. Their reads are exhausted by $R_0, R_1, W_0, W_1, D_0, D_1, T$ before any seal boundary is available.

For RegularCauchyTailMeet$^{\uparrow}$, RealTailAgreementSeal$^{\uparrow}$, and Real$^{\uparrow}$ consumers, apply Definition 893.4 and Definition 893.5. The shared seal route reaches A only through T, and the packet exports no further equality, selector, or completeness data. □

Theorem 893.10 (Regular Cauchy tail fiber Real-completion route coverage)**.** *For an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

every current Real-and-Completion consumer route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the paired-window route*

$$R_0, R_1, W_0, W_1, D_0, D_1 \longmapsto T \longmapsto A$$

before any Real seal is read, together with the displayed transport, continuation, provenance, and local naming rows H, C, P, N*. The route-coverage surface exports no quotient stream, host real equality, selected limit, countable-choice selector, or ambient completeness theorem.*

Proof. Start with the window projection of Definition 893.3. It makes the two regular-rational rows, two stream windows, and two dyadic ledgers the only source route available before a seal-facing row can be reached.

Next apply shared seal factorization and seal-consumer exhaustion from Definition 893.4 and Definition 893.9. These place the shared fiber row T before the seal row A for each listed Real-and-Completion consumer.

Finally use classifier exactness and source-swap stability from Definition 893.11 and Definition 893.15. The classifier can only replay the displayed paired-window, fiber, seal, transport, continuation, provenance, and local naming rows, so no quotient stream, host equality, selector, choice row, or completeness theorem is exported. □

Theorem 893.11 (Regular Cauchy tail fiber classifier exactness)**.** *For accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carriers*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

and

$$F' = (R'_0, R'_1, W'_0, W'_1, D'_0, D'_1, T', A', H', C', P', N'),$$

the local classifier can identify the two packets only through common $\mathsf{StreamName}^{\uparrow}$ *windows, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-radius rows, the shared fiber row, and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *seal row. In particular, classifier agreement is exhausted by displayed agreement of* W_0, W_1, D_0, D_1, T, A *with* $W'_0, W'_1, D'_0, D'_1, T', A'$ *under the available* hsame *transports, and it does not use quotient equality of streams or host real equality.*

Proof. Project both carriers through Definition 893.1. The only rows available to the local classifier are the two source rows, two finite windows, two dyadic ledgers, shared fiber, seal, transport, continuation, provenance, and local name.

Apply the window projection of Definition 893.3. Any source-facing classification must first pass through the common finite windows and their dyadic tail-radius ledgers before a seal-facing row can be observed.

Use the classifier stability and paired-window exactness rows. The hsame transports in H and H' carry the displayed window, radius, fiber, and seal coordinates component by component, while the shared fiber row T is the unique route from the paired windows to the seal row A.

Finally apply non-escape. Since neither accepted packet contains quotient stream equality or host real equality, no such equality can participate in the classifier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket classifier exactness`

Theorem 893.12 (Regular Cauchy tail fiber RegSeqRat Real handoff lock)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

must read the packet through the ordered handoff

$$R_0, R_1 \longrightarrow W_0, W_1 \longrightarrow D_0, D_1 \longrightarrow T \longrightarrow A.$$

Here R_0, R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* W_0, W_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *windows,* D_0, D_1 *are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* T *is the shared tail-fiber row, and* A *is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row consumed at the* $\mathsf{Real}^{\uparrow}$ *boundary. The handoff cannot bypass these rows by using host real equality, quotient stream equality, selected limits, choice-like selectors, or ambient completeness.*

Proof. Project the carrier of Definition 893.1. The accepted packet exposes exactly the two source rows, two windows, two dyadic ledgers, shared fiber, seal row, transports, continuation, package provenance, and local $\mathsf{NameCert}$ row.

Apply the NameCert obligations and window projection from Definition 893.2 and Definition 893.3. They force any source-facing read to start at the two $\mathsf{RegSeqRat}^{\uparrow}$ rows and pass through the displayed $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers.

Use shared seal factorization and classifier exactness from Definition 893.4 and Definition 893.11. The common tail-fiber row T is the route from the window-ledger surface to the seal row A, and the classifier has no quotient-stream or host-real coordinate.

Close with non-escape from Definition 893.5. Since the packet exports no selected limit, choice-like selector, ambient completeness theorem, host real equality, or quotient stream equality, every $\mathsf{Real}^{\uparrow}$-facing handoff remains locked to the displayed ordered row sequence. □

Theorem 893.13 (Regular Cauchy tail fiber tail-agreement seal exactness)**.** *Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet. The tail-agreement seal consumed by* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *is exactly the displayed row* A *reached through the shared fiber row* T *from the paired windows and dyadic ledgers. No detached seal row, external tail-agreement certificate, quotient stream equality, host real equality, selected limit, or ambient completeness witness can replace* A*.*

Proof. Start from the carrier and the local $\mathsf{NameCert}$ obligations. They list a single seal row A together with the paired source, window, dyadic, fiber, transport, continuation, package, and name rows.

Use the window projection to identify the two source rows through W_0, W_1 and D_0, D_1. The seal-facing route therefore begins with the same displayed finite support as the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ consumers.

Apply shared seal factorization. It sends the paired window-ledger support through the common tail-fiber row T before the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row is reachable, so the consumed seal is the displayed row A reached through T.

Finally combine classifier exactness with non-escape. The classifier has no detached seal coordinate, and the packet exports no quotient-stream equality, host real equality, selected limit, or ambient completeness witness; hence there is no second seal available to the consumer. □

Theorem 893.14 (Regular Cauchy tail fiber completion consumer nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet receives only the packet rows*

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

In particular, the consumer receives no selected limit, choice-like selector, ambient completeness theorem, quotient stream equality, host equality of reals, detached tail, detached window, detached dyadic ledger, second fiber, or detached tail-agreement seal.

Proof. Project the carrier and read the NameCert obligations. The completion-facing surface is the displayed finite tuple: two $\mathsf{RegSeqRat}^{\uparrow}$ rows, two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers, one shared fiber row, one tail-agreement seal row, and the structural rows.

Apply window projection and shared seal factorization. They route all source, window, dyadic, tail meet, and seal reads through $R_0, R_1, W_0, W_1, D_0, D_1, T, A$ before any completion-facing consumer can inspect the packet.

Use classifier exactness to keep the local classifier on the displayed window, ledger, fiber, and seal rows. The classifier supplies no quotient stream coordinate and no host real coordinate.

Finally invoke non-escape. Since the carrier contains no selected limit, choice-like selector, ambient completeness theorem, host equality, detached tail, detached window, detached ledger, second fiber, or detached seal, the completion-facing consumer cannot receive any such datum. □

Theorem 893.15 (Regular Cauchy tail fiber source-swap stability)**.** *For an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

the packet obtained by swapping the two source rows together with their $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,*

$$F^{\mathrm{op}} = (R_1, R_0, W_1, W_0, D_1, D_0, T, A, H, C, P, N),$$

has the same shared fiber route T *and reaches the same* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* A *through the same* Cont, Pkg, *and* NameCert *route. The source swap changes only the displayed order of the two source-window-radius triples; it does not create quotient-stream equality, host real equality, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket source swap stability`

Proof. Project the carrier of Definition 893.1. The source-facing coordinates occur as the two triples (R_0, W_0, D_0) and (R_1, W_1, D_1), while the fiber, seal, continuation, provenance, and local-name rows are the shared coordinates T, A, C, P, N.

Apply window projection from Definition 893.3. Reading the original order or the swapped order exposes the same two source-window-radius triples before either read reaches a seal-facing row.

Use shared seal factorization and paired-window exactness from Definition 893.4 and Definition 893.8. Both source orders pass through the common fiber row T before reaching the seal row A, and the continuation and package rows are reused without adding a second route.

Finally apply classifier exactness and non-escape from Definition 893.11 and Definition 893.5. The classifier sees only the displayed window, dyadic ledger, fiber, and seal coordinates, so the swap cannot introduce quotient-stream equality, host real equality, selected limits, or ambient completeness. □

Theorem 893.16 (Regular Cauchy tail fiber componentwise transport)**.** *For an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N),$$

componentwise hsame *transport of*

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N$$

preserves carrier admission and preserves the local classifier on the transported packet. The transported carrier is read through the same displayed H *and* C *rows, so every source, window, radius, fiber, seal, provenance, and local-name coordinate remains part of the same finite packet.*

Proof. Start with the carrier projection. The row H is part of the accepted packet and lists componentwise hsame transport for every coordinate that the packet exposes.

Replay the displayed continuation row C. The transported source rows, windows, dyadic ledgers, shared fiber, and seal are read in the same order as the original carrier, so carrier admission is preserved by rebuilding the finite tuple with transported coordinates.

Apply classifier stability. Because the classifier reads only the window, radius, fiber, and seal coordinates transported by H, the transported packet has the same local classifier support and cannot introduce an external classifier coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket componentwise transport`

Theorem 893.17 (Regular Cauchy tail fiber ledger exhaustion)**.** *Every downstream Real-and-Completion consumer of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

is exhausted by the finite rows

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

Such a consumer cannot read quotient stream equality, a selected limit, choice, ambient completeness, or host real equality from the packet.

Proof. Use the shared seal factorization of Definition 893.4. Every $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ route passes through the same shared fiber row T before it can reach the seal row A.

Apply the seal-consumer exhaustion theorem. The downstream Real-and-Completion consumers listed there are already confined to the two source rows, two windows, two dyadic ledgers, shared fiber, seal, transport, continuation, provenance, and local name.

Finish with non-escape. The carrier contains no quotient stream equality, selected limit, choice row, ambient completeness theorem, or host real equality, so none of these can be exported to a downstream consumer. □

Theorem 893.18 (Regular Cauchy tail fiber seal totality)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

is total exactly on

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

The consumer reads no detached source row, detached window, second dyadic ledger, second fiber, external seal, selected limit, quotient stream equality, host real equality, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket seal totality`

Proof. Start with paired-window exactness. It identifies the common support for $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ and $\mathsf{RealTailAgreementSeal}^{\uparrow}$ reads as the two source rows, two windows, two dyadic ledgers, and the shared fiber row T.

Apply shared seal factorization and seal-consumer exhaustion. The seal-facing consumers reach A only through the same finite packet, and the Real-facing consumers named there do not leave $R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N$.

Use classifier exactness to keep the visible classifier on the displayed window, ledger, fiber, and seal rows. This prevents a seal consumer from creating an additional source, window, dyadic ledger, or fiber coordinate.

Finally apply ledger exhaustion. The downstream Real-and-Completion route is already exhausted by the displayed tuple, so no selected limit, quotient stream equality, host real equality, or ambient completeness witness is exported. □

Theorem 893.19 (Regular Cauchy tail fiber source-seal ledger totality)**.** *Every accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

exposes exactly the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* R_0, R_1*, the two* $\mathsf{StreamName}^{\uparrow}$ *windows* W_0, W_1*, the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-radius ledgers* D_0, D_1*, the shared fiber row* T*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row* A*, and the structural rows* H, C, P, N*. No other source, window, dyadic ledger, fiber, seal, transport, continuation, package, or local NameCert row is available to a consumer of the carrier.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket source seal ledger totality`

Proof. Project the carrier of Definition 893.1. Its displayed coordinates list the two regular-rational source rows, the two stream windows, the two dyadic tail-radius ledgers, the shared fiber row, the seal row, and the four structural rows.

Use the NameCert obligations of Definition 893.2. They identify the same coordinates as the whole local certificate surface, so a consumer cannot obtain a detached source, window, ledger, fiber, or seal row from the local name.

Finally apply ledger exhaustion from Definition 893.17. Every downstream Real-and-Completion read is confined to the same finite list, which gives totality for the source-to-seal ledger. □

Theorem 893.20 (Regular Cauchy tail fiber window-fiber classifier coverage)**.** *Every local classifier read of an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier is covered by the paired windows* W_0, W_1*, the dyadic ledgers* D_0, D_1*, the shared fiber row* T*, and the seal row* A*. The classifier has no quotient stream coordinate and no host-real coordinate.*

Proof. Begin with the window projection of Definition 893.3. Any source-facing classifier read reaches the two source rows through the paired windows and their dyadic ledgers before it can see a seal-facing row.

Next use classifier stability and paired-window exactness from Definition 893.7 and Definition 893.8. These results keep the transported classifier on the same finite window, ledger, and fiber support.

Finally apply classifier exactness from Definition 893.11. The only equality data available to the classifier is displayed agreement of the window, ledger, fiber, and seal coordinates under componentwise transport, so quotient streams and host reals are outside its coverage. □

Theorem 893.21 (Regular Cauchy tail fiber completion handoff scope)**.** *Every downstream Real-and-Completion handoff from an accepted* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier factors through*

$$(R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

before reaching the $\mathsf{Real}^{\uparrow}$ *seal boundary. The handoff cannot claim selected limits, ambient completeness, quotient streams, or host equality.*

Proof. Use the shared seal factorization of Definition 893.4. Both the regular-tail meet route and the tail-agreement route pass through the same shared fiber row T before the seal row A is reachable.

Apply seal-consumer exhaustion from Definition 893.9. The Real-and-Completion consumers named there read only the two source rows, two windows, two dyadic ledgers, shared fiber, seal, and structural rows.

Close with ledger exhaustion from Definition 893.17. Since the packet exports no selected limit, ambient completeness theorem, quotient stream equality, or host real equality, the handoff scope stops at the finite tuple and the seal boundary. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket completion handoff scope`

Theorem 893.22 (Regular Cauchy tail fiber shared-window seal determinacy)**.** *Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

and

$$F' = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

be accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carriers with the same displayed source rows, windows, dyadic ledgers, shared fiber, seal row, transports, continuations, package provenance, and local* $\mathsf{NameCert}$ *row. Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal read available from either carrier factors through the same finite shared-window route*

$$(W_0, W_1, D_0, D_1) \longrightarrow T \longrightarrow A$$

before it reaches the seal row A. No quotient stream, host equality, selected limit, or ambient completeness coordinate can distinguish the two seal routes.

Proof. Project the displayed rows of the two carriers. Since the tuples show the same $R_0, R_1, W_0, W_1, D_0, D_1, T, H, C$ rows, any source-facing read begins with the same finite support.

Use the window projection of Definition 893.3. It pins each source row to the corresponding displayed window and dyadic ledger, so the two carriers expose the same W_0, W_1, D_0, D_1 window surface before any seal-facing route is reached.

Apply shared seal factorization from Definition 893.4. The seal route must pass through the common fiber row T before it can reach A, so the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readings have the same finite path through the shared fiber.

Finally combine completion handoff scope with non-escape from Definition 893.21 and Definition 893.5. These results rule out detached seal paths, quotient-stream equality, host equality, selected limits, and ambient completeness coordinates; hence no external coordinate can separate the two displayed seal routes. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket shared window seal determinacy`

Theorem 893.23 (Regular Cauchy tail fiber source non-escape)**.** *Every source-facing read of an accepted* RegularCauchyTailFiber$^{\uparrow}$ *packet that is later consumed by* RegularCauchyTailMeet$^{\uparrow}$, RealTailAgreementSeal$^{\uparrow}$, *or* Real$^{\uparrow}$ *remains inside*

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

The packet exports no detached source tail, detached StreamName$^{\uparrow}$ *window, second fiber row, quotient stream witness, host-real witness, selected limit, or ambient completeness theorem.*

Proof. Project the carrier of Definition 893.1. Its source-facing data are exactly the two source rows, paired windows, dyadic ledgers, shared fiber row, seal row, and structural rows displayed in the theorem.

Apply paired-window exactness from Definition 893.8. Any source read that reaches a downstream meet, tail-agreement seal, or Real consumer must keep the two windows and two dyadic ledgers paired with the same shared fiber row.

Use seal-consumer exhaustion from Definition 893.9 and source-seal ledger totality from Definition 893.19. These rows account for every seal-facing read through $R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N$ before any downstream consumer can inspect the packet.

Finally apply shared-window seal determinacy from Definition 893.22. Since the finite route through the paired windows, ledgers, fiber, and seal row is already determined, no detached tail, detached window, second fiber, quotient witness, host-real witness, selected limit, or ambient completeness theorem can be exported. □

Theorem 893.24 (Regular Cauchy tail fiber meet route)**.** *Every accepted shared-window tail-fiber read of a* RegularCauchyTailFiber$^{\uparrow}$ *packet factors through the displayed* StreamName$^{\uparrow}$ *finite-window rows, the* RegSeqRat$^{\uparrow}$ *classifier rows, the* DyadicRatCore$^{\uparrow}$ *tolerance rows, and the* CauchyLimitSeal$^{\uparrow}$ *completion handoff before it may be consumed as a completion-facing fiber. The route exports no detached tail, detached finite window, second fiber, quotient stream witness, host-real equality, selected limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberPacket meet route`

Proof. Project the accepted carrier rows

$$R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N.$$

The two source rows are read only through the displayed windows, dyadic ledgers, shared fiber, seal row, histories, continuations, package provenance, and local naming row.

Apply the window projection theorem Definition 893.3. It places each RegSeqRat$^{\uparrow}$ source read behind the corresponding StreamName$^{\uparrow}$ finite-window row and DyadicRatCore$^{\uparrow}$ tolerance row.

Next use shared-window seal determinacy from Definition 893.22. The two accepted routes therefore meet at the same shared fiber row T before any seal-facing read can reach A.

Transport the classifier componentwise by Definition 893.16, then exhaust the ledger by Definition 893.17. These steps keep the classifier, finite windows, dyadic tolerance rows, and shared fiber on the displayed packet.

Finally apply completion handoff scope from Definition 893.21. The resulting completion-facing fiber is already routed through the finite-window, classifier, tolerance, shared-seal, and CauchyLimitSeal$^{\uparrow}$ handoff surface, so no external coordinate remains available to the consumer. □

Theorem 893.25 (Regular Cauchy tail fiber Real-seal determinacy)**.** *Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *seal-facing consumer that agrees with the shared meet route through the displayed fiber row* T *reaches the unique displayed seal row* A*. Such a consumer cannot replace* A *by a second seal, quotient-stream equality, host real equality, selected limit, or ambient completeness witness.*

Proof. Project the carrier of Definition 893.1. The only seal-facing rows displayed by the accepted packet are the paired source rows, paired windows, dyadic ledgers, shared fiber row T, seal row A, transports, continuation, provenance, and local name.

Use the window projection theorem Definition 893.3 and shared seal factorization from Definition 893.4. Together they force every seal-facing read to pass through the displayed windows and then through T before reaching A.

Apply paired-window exactness from Definition 893.8 and classifier exactness from Definition 893.11. These rows identify the two window ledgers and the source classifiers with the same finite meet route, so an agreeing $\mathsf{Real}^{\uparrow}$ consumer has no second path to a different seal coordinate.

Finally invoke the meet route Definition 893.24 and non-escape from Definition 893.5. The meet route supplies the shared route through T, while non-escape excludes detached seal data, quotient-stream equality, host real equality, selected limits, and ambient completeness witnesses. Hence the displayed row A is the unique seal row available to the consumer. □

Theorem 893.26 (Regular Cauchy tail fiber limit-seal functor handoff)**.** *Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *packet. Let an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *consumer have carrier*

$$G = (S_G, T_G, M_G, W_0^G, W_1^G, R_G, D_G, K_G, E_G, H_G, C_G, P_G, N_G)$$

whose source seal row S_G *reads the same finite tail-agreement row* A*. Then the consumer must factor through the displayed finite route*

$$R_0, R_1 \longmapsto W_0, W_1 \longmapsto D_0, D_1 \longmapsto T \longmapsto A$$

before it can read its transport-map row M_G*, completion-seal classifier row* K_G*, terminal endpoint row* E_G*, or structural rows* H_G, C_G, P_G, N_G*. The handoff exports no quotient-stream equality, host real equality, selected limit, countable-choice selector, detached completion object, or ambient completeness principle.*

Proof. Project the accepted carrier of Definition 893.1. The source rows R_0, R_1 are visible only through their finite windows W_0, W_1 and dyadic ledgers D_0, D_1 before a seal-facing row can be inspected.

Apply the window projection, shared seal factorization, and Real-seal determinacy of Definition 893.3, Definition 893.4, and Definition 893.25. These results force every agreeing Real or seal-facing read through $R_0, R_1, W_0, W_1, D_0, D_1, T, A$, and they make A the only displayed tail-agreement seal reached from T.

Now read the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ side through the carrier boundary of Definition 1116.1 and the consumer boundary of Definition 1116.7. Since its source seal S_G is the same finite row A,

the functor consumer cannot reach its transport-map, completion-classifier, endpoint, or structural rows until it has consumed that source seal along the displayed tail-fiber route.

The non-escape clauses in Definition 893.5, Definition 893.21, and Definition 1116.7 remove quotient-stream equality, host real equality, selected limits, countable-choice selectors, detached completion objects, and ambient completeness from both packets. Therefore the handoff is exactly the finite route through $R_0, R_1, W_0, W_1, D_0, D_1, T, A$ before the functor reads $M_G, K_G, E_G, H_G, C_G, P_G, N_G$. □

Theorem 893.27 (Regular Cauchy tail fiber finite-window Real-seal equivalence)**.** *Let*

$$F = (R_0, R_1, W_0, W_1, D_0, D_1, T, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *carrier. The* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *route through the paired finite windows and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal route describe the same completion-facing read exactly when both factor through the finite tuple*

$$(W_0, W_1, D_0, D_1, T, A)$$

under the displayed hsame *transports carried by* H *and the displayed continuations carried by* C*. This correspondence uses no quotient stream, host real equality, selected limit, or ambient completeness coordinate.*

Proof. Begin with the forward route. Apply the window projection of Definition 893.3; the two source reads are visible only through W_0, W_1 and their dyadic ledgers D_0, D_1 under the displayed transports.

Use shared seal factorization from Definition 893.4. The paired-window read reaches the shared fiber row T before it reaches the seal row A, so the finite tuple $(W_0, W_1, D_0, D_1, T, A)$ already carries the meet-to-seal route.

For the reverse route, apply paired-window exactness from Definition 893.8 and classifier exactness from Definition 893.11. A $\mathsf{RealTailAgreementSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal read that reaches A through T determines the same paired windows and dyadic ledgers, hence returns to the finite meet surface.

Finally invoke Real-seal determinacy from Definition 893.25 and non-escape from Definition 893.5. The seal row is the unique displayed A reached from T, and no quotient stream, host equality, selected limit, or ambient completeness coordinate remains available to separate the two routes. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFiber}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, two finite $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ tail-radius ledgers, one shared tail fiber row, one $\mathsf{Real}^{\uparrow}$ tail-agreement seal row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFiber}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 893.1, Definition 893.2, Definition 893.3, Definition 893.4, Definition 893.5, Definition 893.6, Definition 893.7, Definition 893.8, Definition 893.9, Definition 893.11, Definition 893.12, Definition 893.13, Definition 893.14, Definition 893.15, Definition 893.16, Definition 893.17, Definition 893.19, Definition 893.20, Definition 893.21, Definition 893.22, Definition 893.25, Definition 893.26, and Definition 893.27 specify the finite common-tail fiber carrier, local NameCert rows, window projection, shared seal factorization, non-escape boundary, TasteGate follow-up contract, classifier stability, paired-window exactness, seal-consumer exhaustion, classifier exactness, RegSeqRat-to-Real handoff lock, tail-agreement seal exactness, completion consumer nonescape, source-swap stability, componentwise transport, ledger exhaustion, source-seal ledger totality, classifier coverage, completion handoff scope, shared-window seal determinacy, Real-seal determinacy, limit-seal functor handoff, and finite-window Real-seal correspondence.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFiberUp.taste gate`

Lean target: `BEDC.Derived.RegularCauchyTailFiberUp.RegularCauchyTailFiberTasteGate\_single\_carrier\_alignment`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFiber}^{\uparrow}}, \mathsf{axiomClean})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove quotient equality of regular streams, host real equality, selected-limit existence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{RegularCauchyTailFiber}^{\uparrow}$.

Upgrade path: No further paper-side theory closure remains for the common-tail fiber and $\mathsf{Real}^{\uparrow}$ seal handoff; bridge checking remains on the verification axis.

Chapter 894

A Concrete Naming Certificate for $\mathsf{ObserverFrameRefusal}^{\uparrow}$

The $\mathsf{ObserverFrameRefusal}^{\uparrow}$ packet gives the no-global-frame observer direction a concrete NameCert carrier. It packages the refusal surface implicit in chapter 9418 as finite rows: an observer-history row, local coordinate-frame row, locality-cell row, rate row, synchronization-refusal ledger, componentwise transport, continuation replay, package provenance, and local naming row. It sits after chapter 479, chapter 609, chapter 513, and chapter 876: those chapters supply the identity, local frame, locality-cell, and maximum-rate surfaces, while this chapter records that none of them may be promoted to a synchronized external observer frame.

Definition 894.1 (Observer-frame refusal carrier)**.** *An* $\mathsf{ObserverFrameRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, Q, L, R, G, H, C, P, N)$$

with the following displayed rows:

1. *O is the observer-history identity row consumed from* $\mathsf{ObserverHistIdentity}^{\uparrow}$*;*
2. *Q is the local coordinate-frame row consumed from* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *as an observer-local coordinate packet, not as a global frame;*
3. *L is the locality-cell row consumed from* $\mathsf{ObserverLocalityCell}^{\uparrow}$*;*
4. *R is the cross-Hist causal-rate row consumed from* $\mathsf{CrossHistCausalRate}^{\uparrow}$*;*
5. *G is the refusal ledger, recording that no synchronized clock, privileged observer frame, global coordinate chart, metric tensor, or all-observer synchronization row is admitted;*
6. *H records componentwise* hsame *transports for the observer-history, coordinate-frame, locality-cell, causal-rate, refusal-ledger, route, provenance, and naming rows;*
7. *C records displayed* Cont *routes for reading the local observer rows, checking the refusal ledger, and replaying the accepted consumer route;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverFrameRefusal}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverCoordinateFrame}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, $\mathsf{CrossHistCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverFrameRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a synchronized clock, privileged observer-frame row, global frame, metric tensor, quotient observer identity, host equality of histories, or unlisted external coordinate.

Theorem 894.2 (Observer-frame refusal NameCert obligations)**.** *For the carrier of Definition 894.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverFrameRefusal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (O, Q, L, R, G, H, C, P, N),$$

observer-Hist identity through O*, local coordinate-frame use through* Q*, locality-cell anchoring through* L*, cross-Hist causal-rate use through* R*, explicit refusal of synchronized clocks and privileged external frames through* G*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 894.1. The rows $O, Q, L, R, G, H, C, P, N$ account for the observer-history identity, local frame, locality cell, causal-rate row, refusal ledger, transport, route replay, provenance, and local naming surface.

The first four analytic rows are consumed from the sibling chapters. The observer-Hist identity row comes from chapter 479; the local coordinate-frame row comes from chapter 609; the locality cell comes from chapter 513; and the causal-rate row comes from chapter 876.

The refusal ledger G is the row that blocks promotion of those local rows into a synchronized external frame. The structural rows H, C, P, N replay and name the same refusal route, so every accepted read remains inside the finite packet. □

Theorem 894.3 (Observer-frame refusal no-global-sync boundary)**.** *Every accepted* $\mathsf{ObserverFrameRefusal}^{\uparrow}$ *packet* $F = (O, Q, L, R, G, H, C, P, N)$ *admits observer-facing reads only through the displayed local rows* O, Q, L, R *and the refusal ledger* G*, replayed through* H, C, P, N*. A consumer that reads a synchronized clock, privileged observer-frame row, global frame, metric tensor, quotient observer identity, host equality of histories, or unlisted external coordinate has left the accepted packet.*

Proof. Project the carrier. The only observer-facing rows are O, Q, L, R, and the only row that records the refusal of external synchronization data is G.

Apply Definition 894.2. The local NameCert surface requires the identity, coordinate-frame, locality-cell, causal-rate, and refusal rows to be consumed together with H, C, P, N.

Finally inspect the excluded coordinates in Definition 894.1. No synchronized clock, privileged observer frame, global coordinate chart, metric tensor, quotient observer identity, host equality of histories, or unlisted external coordinate is present. Therefore an accepted consumer route can only display such a row as refused external data, never as an admitted packet coordinate. □

Theorem 894.4 (Observer-frame refusal TasteGate obligation)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{ObserverFrameRefusal}^{\uparrow}$ *must include* $\mathsf{Nontrivial\ ObserverFrameRefusal}^{\uparrow}$ *with explicit distinct witnesses and* $\mathsf{FieldFaithful\ ObserverFrameRefusal}^{\uparrow}$ *by field projection and injection. These gate-facing instances must preserve the observer-history row, local coordinate-frame row, locality-cell row, cross-Hist causal-rate row, refusal ledger, transport rows, continuation routes, provenance row, and local naming row.*

Proof. The carrier has separate rows for observer identity, local coordinate use, locality-cell anchoring, causal-rate reading, and refusal of external synchronization. A singleton collapse would erase the difference between local observer data and refused external frame data.

Field faithfulness is the projection discipline for the displayed tuple. Each row of $F = (O, Q, L, R, G, H, C, P, N)$ has a distinct role, and the refusal ledger G must remain distinguishable from the local coordinate-frame row Q and the causal-rate row R. □

Falsifiable prediction: An accepted $\mathsf{ObserverFrameRefusal}^{\uparrow}$ packet admits no consumer route that reads a synchronized clock or privileged observer-frame row unless that row is displayed as a refused external coordinate.

Independence witness: observerhistidentity, observercoordinateframe, observerlocalitycell, histtimestream, crosshistcausalrate: $\mathsf{ObserverFrameRefusal}^{\uparrow}$ carries an explicit refusal ledger for privileged frame and synchronization imports; identity-only, coordinate-frame, locality-cell, timestream, and rate packets do not have the same refusal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverFrameRefusal}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverFrameRefusal}^{\uparrow}$ packet is generated from observer-history identity, local coordinate-frame, locality-cell, cross-Hist causal-rate, refusal-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverFrameRefusal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 894.1, Definition 894.2, Definition 894.3, Definition 894.4, and

Lean4 ✓ `BEDC.Derived.ObserverFrameRefusalUp.taste gate`

specify the finite refusal packet for no-global-frame observer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverFrameRefusal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ObserverFrameRefusalUp.ObserverFrameRefusalTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive synchronized clocks, privileged observer frames, global coordinates, metric tensors, quotient observer identity, host equality of histories, or bridge checking for $\mathsf{ObserverFrameRefusal}^{\uparrow}$.

Upgrade path: No further paper-theory upgrade remains; the remaining independent step is bridge checking without adding synchronized-frame content.

Chapter 895

A Concrete Naming Certificate for $\mathsf{MultiHistCoupling}^{\uparrow}$

The $\mathsf{MultiHistCoupling}^{\uparrow}$ packet realizes the paired-history coupling step in the Inter-Hist Locality chain of chapter 9419. It sits after the no-subject observer boundary of chapter 615, the observer-history identity surface of chapter 479, the multi-history configuration carrier of chapter 474, and the invariant readback packet of chapter 656. Its role is to expose a symmetric finite coupling ledger between two displayed histories before a $\mathsf{CrossHistCausal}^{\uparrow}$ consumer reads causal dependence or a rate packet reads a maximum boundary.

Definition 895.1 (MultiHistCoupling carrier)**.** *A* $\mathsf{MultiHistCoupling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H_0, H_1, J, S, T, C, P, N)$$

with the following displayed rows:

1. *H_0 and H_1 are paired* BHist *rows read from the same* $\mathrm{MultiHistConfig}^{\uparrow}$ *surface, with the no-subject boundary inherited from* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *and the identity-facing read inherited from* $\mathsf{ObserverHistIdentity}^{\uparrow}$*;*
2. *J is the finite coupling ledger that records the two directed reads $H_0 \to H_1$ and $H_1 \to H_0$ as one symmetric coupling packet;*
3. *S is the observer-symmetry witness, stating that the coupling ledger is not oriented by a privileged observer, global frame, synchronized clock, or subject-substance coordinate;*
4. *T stores componentwise* hsame *transport for the two* BHist *rows and displayed* psame *transport for the probe and signature rows used by the coupling ledger;*
5. *C records the displayed* Cont *routes by which a consumer reads H_0, reads H_1, checks J, replays S, and transports the same coupling packet through T;*
6. *P is the* Pkg *provenance over* $\mathsf{MultiHistCoupling}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{MultiHistCoupling}^{\uparrow}}$ *row.*

The carrier exports no global observer frame, no host equality of histories, no quotient of history pairs, no subject-substance coordinate, and no causal-dependence row outside the displayed coupling ledger.

Lean4 ✓ BEDC.Derived.MultihistCouplingUp.TasteGate.MultihistCouplingTasteGate single carrier alignment

Theorem 895.2 (MultiHistCoupling NameCert obligations)**.** *For the carrier of Definition 895.1, the local* $\mathsf{NameCert}_{\mathsf{MultiHistCoupling}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $K = (H_0, H_1, J, S, T, C, P, N)$*;*
2. *paired-history source exactness, requiring* H_0 *and* H_1 *to be read from the displayed* $\mathrm{MultiHistConfig}^{\uparrow}$ *row after the no-subject boundary and observer-history identity surfaces are available;*
3. *symmetric coupling ledger exactness, requiring* J *and* S *to expose both coupling directions without privileging either observer history;*
4. *transport stability through componentwise* hsame*, displayed* psame*, and replay of the listed* Cont *routes in* T, C*;*
5. *non-escape, excluding global frames, synchronized clocks, host history equality, quotient pairs, subject-substance rows, and any causal-dependence read not routed through* $H_0, H_1, J, S, T, C, P, N$*.*

Proof. Carrier admission is projection from Definition 895.1. The tuple lists the paired histories, coupling ledger, symmetry witness, transports, continuation routes, provenance, and local name.

The source rows are consumed from the sibling surfaces. The no-subject boundary of chapter 615 prevents a hidden observer-substance coordinate, chapter 479 supplies the observer-history identity read, and chapter 474 supplies the common finite multi-history configuration.

The ledger rows J and S then make the coupling symmetric. They record both directed reads as one packet and refuse any global-frame or privileged-observer orientation.

Finally T, C, P, N replay, transport, package, and name the same finite rows. Since the carrier contains no coordinate for a host equality, quotient pair, subject-substance row, or external causal predicate, the five obligations exhaust the local certificate surface. □

Theorem 895.3 (MultiHistCoupling invariant handoff)**.** *Every accepted* $\mathsf{MultiHistCoupling}^{\uparrow}$ *carrier*

$$K = (H_0, H_1, J, S, T, C, P, N)$$

hands an $\mathsf{InterHistInvariant}^{\uparrow}$ *or* $\mathsf{CrossHistCausal}^{\uparrow}$ *consumer exactly the paired-history rows, symmetric coupling ledger, observer-symmetry witness,* hsame *and* psame *transports, displayed* Cont *routes, package provenance, and local naming row. The consumer receives no global frame, no host equality of histories, no quotient pair, and no causal-dependence coordinate outside the displayed coupling packet.*

Proof. Project the accepted carrier to its displayed tuple. The only history rows available to the consumer are H_0 and H_1, and the only coupling row available between them is J with the symmetry witness S.

Apply the local NameCert obligations of Definition 895.2. Paired-history source exactness fixes the upstream configuration and observer-boundary reads, while symmetric coupling ledger exactness fixes the two directed reads as one finite coupling surface.

The invariant consumer of chapter 656 may therefore read the paired histories through J, S, T, C, P, N. A $\mathsf{CrossHistCausal}^{\uparrow}$ consumer may use that same finite route as its input surface, but any additional causal-dependence row must be displayed by the downstream causal packet rather than projected from this carrier. □

Theorem 895.4 (MultiHistCoupling TasteGate obligation)**.** *The later Lean* `Derived/MultiHistCouplingUp/Tas` *development must provide explicit* Nontrivial MultiHistCoupling$^{\uparrow}$ *and* FieldFaithful MultiHistCoupling$^{\uparrow}$ *instances. These instances must preserve the two paired* BHist *rows, symmetric coupling ledger, observer-symmetry witness,* hsame *and* psame *transports, displayed* Cont *routes, package provenance, and local naming row.*

Proof. The packet is not determined by either single history alone. It contains two history rows and a separate symmetric coupling ledger between them.

Nontriviality prevents collapse of H_0, H_1, and J into one indistinguishable row. Field faithfulness is the projection discipline for $K = (H_0, H_1, J, S, T, C, P, N)$, preserving each displayed coordinate and keeping the coupling ledger distinct from the sibling no-subject, identity, configuration, invariant, and causal-dependence surfaces. □

Falsifiable prediction: If the coupling row is omitted, no CrossHistCausalUp consumer can read the paired histories through a symmetric finite route.

Independence witness: observerhistidentity, observersubjectboundary, multihistconfig, interhistinvariant, crosshistcausal: MultiHistCouplingUp is not bijective to any of them because it contains both paired-history rows and a symmetric coupling ledger; each sibling lacks either the paired coupling row, the no-subject boundary, the invariant output row, or the causal-dependence row.

Closure status for NameCert$_{\text{MultiHistCoupling}^{\uparrow}}$**.**

Constructive story: A MultiHistCoupling$^{\uparrow}$ packet is generated from two paired BHist rows, a symmetric coupling ledger, an observer-symmetry witness, componentwise hsame and displayed psame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MultiHistCoupling}^{\uparrow}}$, seed)

Scope closed: Definition 895.1, Definition 895.2, Definition 895.3, and Definition 895.4 specify the finite paired-history coupling packet, its NameCert obligations, downstream handoff, and TasteGate instance surface.

Formal verification: FormalStatus(NameCert$_{\text{MultiHistCoupling}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MultihistCouplingUp.TasteGate.MultihistCouplingTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close CrossHistCausalUp dependence, MaxCausalRateUp bounds, Lorentz geometry, global frames, quotient history equality, or bridge checking for MultiHistCoupling$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, paired-history source exactness, symmetric coupling exactness, transport stability, non-escape, and TasteGate nontriviality with field faithfulness.

Chapter 896

A Concrete Naming Certificate for $\mathsf{DiagonalLimitBudget}^{\uparrow}$

The $\mathsf{DiagonalLimitBudget}^{\uparrow}$ packet is the finite budget carrier between the diagonal selector of chapter 673 and the RealUp seal route synchronized in chapter 886. It isolates the modulus, selected window, dyadic-tolerance ledger, and seal handoff that the Real completeness diagonal route consumes before any completed real value is read. The observation envelope of chapter 637 supplies the sibling finite readback surface; this chapter records the budget packet that keeps that readback tied to the diagonal index request.

Falsifiable prediction: A packet with the same RealUp seal row but no displayed modulus budget row is not an accepted DiagonalLimitBudgetUp carrier.

Independence witness: LimitSelectorUp selects the diagonal row but lacks the seal budget; CauchySealBudgetSynchronizerUp synchronizes selector and seal rows but lacks the isolated budget carrier; RealApproximationEnvelopeUp packages observation envelopes but not the diagonal modulus budget; DiagonalLimitCompatibilityUp compares limits but does not provide the budget packet.

Definition 896.1 (DiagonalLimitBudget carrier)**.** *A* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, M, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the diagonal index request read from the Real completeness diagonal representative of Definition 9238.3;*
2. *M is the modulus budget row displaying the precision request and the selected modulus value used to answer D;*
3. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}$ *window row chosen by the diagonal selector for that modulus budget;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback ledger attached to the selected window;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the same diagonal route;*
6. *H stores componentwise* hsame *transports for D, M, W, Q, E, C, P, N;*
7. *C stores the* Cont *routes replaying the index request, budget row, selected window, dyadic ledger, seal row, provenance, and local naming row;*

8. *P is the* Pkg *provenance over* $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RealApproximationEnvelope}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host real equality, countable choice, a host-selected limit, or ambient completeness.

Lean4 (definition) `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetUp`

Theorem 896.2 (DiagonalLimitBudget non-escape)**.** *Every accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *packet read by a diagonal-limit or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the displayed modulus budget row M before it reaches the selected window W, dyadic ledger Q, or seal row E. A consumer that reads E without exposing D, M, W, Q, H, C, P, N is not reading the accepted budget carrier.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetNonescape`

Proof. Project the carrier of Definition 896.1. The only route to the seal coordinate is the displayed tuple $B = (D, M, W, Q, E, H, C, P, N)$, and the budget row M is the only row that records the modulus value consumed by the diagonal request.

The transport rows H and replay routes C preserve the same coordinates. They can pass the selected window and dyadic ledger to a consumer, but they do not create a second modulus row, a second seal, or an external limit witness.

Thus any accepted seal read must expose the carrier rows in order: request, budget, window, dyadic readback, seal, transport, route, provenance, and local name. Since the excluded objects have no coordinate in B, they cannot be recovered from an accepted packet. □

Theorem 896.3 (DiagonalLimitBudget seal handoff)**.** *For an accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier $B = (D, M, W, Q, E, H, C, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal row E is handed to the synchronizer route only together with the same modulus budget M, selected window W, dyadic ledger Q, and structural rows H, C, P, N. The handoff supplies a budgeted seal route, not an isolated RealUp seal.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetSealHandoff`

Proof. Read the budget carrier and the synchronizer sibling together. The synchronizer of chapter 886 accepts a seal route only when a displayed budget row is available for the same request.

The $\mathsf{DiagonalLimitBudget}^{\uparrow}$ carrier provides exactly such a row: M is tied to the request D, and W and Q record the finite window and dyadic readback selected for that budget. The seal coordinate E is therefore exported only through the same packet.

The package row P and continuation row C replay the sibling dependencies on $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RealApproximationEnvelope}^{\uparrow}$, and $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$. No coordinate supplies an isolated seal, host-selected limit, quotient stream, or ambient completeness route. □

Theorem 896.4 (DiagonalLimitBudget obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for B, diagonal request exposure through D, modulus-budget exposure through M, selected* $\mathsf{StreamName}^{\uparrow}$ *and* RegSeqRat *window exposure through W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback exposure through Q,* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise* hsame *classifier transport through H, displayed* Cont *classifier replay through C,* Pkg *provenance through P, and local naming through N. The same displayed D, M request determines the same W, Q readback surface, and the seal row E is readable only along the displayed M, W, Q budget route.*

Lean4 ✓ BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetObligationSurface

Proof. Carrier admission is projection from the displayed tuple. The request, budget, selected window, dyadic ledger, seal, transport, continuation, provenance, and naming rows are all explicit coordinates.

The selector-facing side is exhausted by D, M, W: the request asks for a diagonal index, the budget row records the modulus value, and the selected window names the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}$ read. The observation-facing side is exhausted by Q, E: the dyadic ledger records the tolerance and readback, and the seal row records the RealUp endpoint reached by that ledger.

Classifier transport is carried by the structural rows. The componentwise hsame row H transports the displayed coordinates D, M, W, Q, E, C, P, N, while the Cont row C replays the index request, budget row, selected window, dyadic ledger, seal row, provenance, and local naming row. Thus the classifier visible to a consumer is the classifier of the displayed packet, not a hidden external classifier.

Selected-window determinacy is the finite readback part of the same projection. Since W and Q are coordinates of B attached to the displayed request and budget rows, two accepted reads of the same D, M surface in this packet expose the same selected-window and dyadic-readback rows. By Definition 896.2, there is no second modulus row, second selected window, or external limit witness from which a different W, Q readback could be recovered.

Seal-budget exactness follows from the seal handoff. By Definition 896.3, the seal row E is exported only together with the same M, W, Q, H, C, P, N rows. Therefore E is readable precisely as the displayed budgeted seal route, and the obligation surface contains no hidden classifier, no quotient stream equality, no host real equality, and no ambient completeness premise. □

Theorem 896.5 (DiagonalLimitBudget independence witness)**.** *The* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier is not reducible to any one sibling route among* $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{RealApproximationEnvelope}^{\uparrow}$, *or* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$. *Its distinguishing data is the simultaneous presence of the diagonal request* D*, modulus budget* M*, selected window* W*, dyadic-tolerance ledger* Q*, RealUp seal* E*, and structural rows* H, C, P, N *in one finite packet.*

Lean4 ✓ BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetIndependenceWitness

Proof. Compare the sibling carriers by their displayed rows. The limit selector exposes the diagonal selection route, but it does not isolate the modulus budget as the budget object consumed by the seal handoff.

The synchronizer exposes the joint seal route, but it treats the budget row as an input from another packet. The approximation envelope exposes the finite observation ledger, but it does not bind that ledger to the diagonal modulus budget. The compatibility packet compares limit routes, but it does not supply the budget carrier consumed before comparison.

The packet $B = (D, M, W, Q, E, H, C, P, N)$ displays all of these rows together. Removing M loses the finite budget, removing Q loses the dyadic-tolerance ledger, removing E loses the seal handoff, and removing H, C, P, N loses transport, replay, provenance, and local naming. □

Theorem 896.6 (DiagonalLimitBudget TasteGate readiness)**.** *The Lean TasteGate module for* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ DiagonalLimitBudget}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ DiagonalLimitBudget}^{\uparrow}$ *instances. Nontriviality separates the modulus budget row from the seal row, and field faithfulness preserves the displayed rows* $D, M, W, Q, E, H, C, P, N$ *for every TasteGate-facing consumer.*

Lean4 ✓ BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetTasteGateReadiness
↪ variant BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetTasteGate single carrier alignment

Proof. The carrier has two distinct analytic roles: M records the finite modulus budget and E records the RealUp seal reached after the budgeted route. A collapsed packet that identifies these roles would admit the seal without the displayed budget row, contradicting Definition 896.2.

Field faithfulness is the row-preservation condition needed by the same surface. A TasteGate-facing consumer may inspect the request, budget, selected window, dyadic ledger, seal, transport, route, provenance, and local naming rows, but no unlisted analytic or host-side object. □

Theorem 896.7 (DiagonalLimitBudget carrier admission)**.** *An accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier is admitted exactly as the finite* BHist *packet* $B = (D, M, W, Q, E, H, C, P, N)$ *whose local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *row* N *reads the diagonal request* D*, modulus budget* M*, selected window* W*, dyadic tolerance ledger* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, and* Pkg *provenance* P*.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetCarrierAdmission`

Proof. Inspect the carrier of Definition 896.1. Its displayed coordinates are precisely $D, M, W, Q, E, H, C, P, N$, and each coordinate is a finite BHist row.

Apply Definition 896.4. The local $\mathsf{NameCert}$ surface admits only the request, budget, selected window, dyadic ledger, seal, transport, replay, provenance, and naming rows listed there.

Finally use Definition 896.2. A consumer that omits D, M, W, Q, H, C, P, N cannot reach the accepted seal row. Thus carrier admission is the displayed finite packet and not parameter transport through an external budget object. □

Theorem 896.8 (DiagonalLimitBudget synchronization stability)**.** *For every accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier* $B = (D, M, W, Q, E, H, C, P, N)$*, any two reads transported by the same componentwise* hsame *row* H *and replayed through the same displayed* Cont *row* C *expose the same selected* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window row* W*, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*, the same* Pkg *provenance* P*, and the same local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *row* N *attached to* D, M*.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetSynchronizationStability`

Proof. Begin with finite row inspection of B. The selected-window row W and dyadic ledger Q are coordinates of the same packet as the request D and modulus budget M.

Use Definition 896.4. Componentwise hsame transport through H preserves the displayed coordinates, and Cont replay through C replays the index request, budget row, selected window, dyadic ledger, seal row, provenance, and local naming row.

The non-escape theorem excludes any second selected window, second dyadic ledger, hidden provenance row, or external local name. Hence two accepted reads along the same H and C surface expose the same W, Q, P, N rows for the displayed D, M budget. □

Theorem 896.9 (DiagonalLimitBudget Real seal handoff)**.** *For every accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *carrier* $B = (D, M, W, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is readable only after the diagonal request* D*, modulus budget* M*, selected* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*, componentwise* hsame *row* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *row* N *have been displayed.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetRealSealHandoff`

Proof. Apply Definition 896.3. The seal is handed to the synchronizer route only together with the same modulus budget, selected window, dyadic ledger, and structural rows.

Apply Definition 896.2. The packet has no route to E that bypasses D, M, W, Q, H, C, P, N, and no coordinate stores an isolated RealUp seal.

The obligation surface records why this is stable under reading. The rows H and C transport and replay the displayed packet, while P and N source and name the same local carrier. Therefore the Real seal handoff is the displayed budgeted route, not a parameter-transported or host-selected Real completion claim. □

Theorem 896.10 (DiagonalLimitBudget scoped L10 grounding)**.** *Every accepted* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *packet grounds the diagonal request, modulus budget, selected window, dyadic ledger, and Real seal through the ordered scoped route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In particular, the same carrier $B = (D, M, W, Q, E, H, C, P, N)$ *displays the request* D*, budget* M*, selected* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N *before any Real-facing consumer can read the seal.*

Proof. Start with the non-escape theorem. It forces every accepted seal read to expose D, M, W, Q, H, C, P, N, so the request and budget cannot be separated from the selected window and dyadic ledger.

Use the obligation surface to identify the rows. The selected-window side is exactly the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ part of W, and the readback side is exactly the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger Q attached to the same displayed D, M request.

Finally apply the two seal handoff theorems. The seal handoff keeps E attached to M, W, Q, H, C, P, N, and the Real seal handoff states that E is readable only after those rows have been displayed. Thus the scoped grounding route is the displayed packet route, not an external Real-completion or quotient-stream route. □

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetScopedL10Grounding`

Theorem 896.11 (DiagonalLimitBudget scoped row exhaustion)**.** *Every accepted packet* $B = (D, M, W, Q, E, H, C, P, N)$ *exposes exactly the diagonal request* D*, modulus budget* M*, selected* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ *row* N *before any Real-facing consumer can read* E*. No additional row for quotient streams, host real equality, selected limits, hidden completeness, or ambient choice is available inside the scoped L10 route.*

Lean4 ✓ `BEDC.Derived.DiagonalLimitBudgetUp.DiagonalLimitBudgetScopedRowExhaustion`

Proof. Start with the non-escape boundary in Definition 896.2. It forces every accepted seal read to pass through the displayed packet coordinates D, M, W, Q, H, C, P, N before E is consumed.

The obligation surface of Definition 896.4 identifies those coordinates as the complete local $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$ surface. The request and budget are D, M, the finite window is W, the dyadic readback is Q, and the structural rows are exactly H, C, P, N.

Now apply the Real seal handoff of Definition 896.9. The handoff keeps E attached to the same selected window, dyadic ledger, transport, replay, provenance, and local name, so the Real-facing consumer cannot isolate the seal from the budget packet.

Finally use the scoped L10 grounding in Definition 896.10. It orders the route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and therefore exhausts the rows available to

the scoped consumer. Since every excluded object is absent from the carrier and from the handoff route, the listed rows are exact. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalLimitBudget}^{\uparrow}$ packet is generated from a diagonal request, a displayed modulus budget, selected $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ window data, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 896.1, Definition 896.4, Definition 896.10, and Definition 896.11 bind the displayed rows $D, M, W, Q, E, H, C, P, N$ to the L10 diagonal-limit kernel route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalLimitBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host real equality, countable choice, selected limits, hidden RealUp completion data, or ambient completeness for $\mathsf{DiagonalLimitBudget}^{\uparrow}$.

Upgrade path: Scoped grounding requires checking the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows against the same $D, M, W, Q, E, H, C, P, N$ packet.

Chapter 897

A Concrete Naming Certificate for $\mathsf{StationaryWindowNaturality}^{\uparrow}$

The $\mathsf{StationaryWindowNaturality}^{\uparrow}$ packet records the finite naturality comparison required when a transported stationary rational classifier is read through the Real-completeness finite-window route. It consumes the transported stationary diagonal surface of chapter 475, the route boundary of chapter 613, the observation-budget surface of chapter 641, and the fused regular-Cauchy window packet of chapter 693. The carrier keeps the seed classifier and the four boundary readings in one displayed packet; it does not export a quotient stream, selected limit, host real equality, or ambient completeness principle.

Falsifiable prediction: A RealUp stationary-window consumer that bypasses RatUp or StreamNameUp finite-window transport refutes the packet.

Independence witness: transportedstationarydiagonal, stationarydiagonalwindowroute, cauchymodulusextraction, regularcauchywindowfusion: none is bijective because this carrier adds a naturality comparison row tying four boundary readings to one RatUp seed classifier.

Definition 897.1 (Stationary window naturality carrier)**.** *A* $\mathsf{StationaryWindowNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (Q, S, R, D, E, W, H, C, P, K)$$

with the following displayed rows:

1. *Q is the transported stationary* $\mathsf{Rat}^{\uparrow}$ *seed classifier row;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window classifier induced by Q;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-consumption row read from the same finite window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and tolerance ledger attached to that window;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *finite-window seal row reached from S, R, D;*
6. *W is the naturality comparison row asserting that the four boundary readings S, R, D, E are replays of the same seed classifier Q;*
7. *H stores componentwise* hsame *transports for $Q, S, R, D, E, W, C, P, K$;*
8. *C records the displayed* Cont *routes that replay the seed, stream window, regular readback, dyadic ledger, real seal, and naturality row;*

9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *K is the local* $\mathsf{NameCert}_{\mathsf{StationaryWindowNaturality}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unrecorded real comparison, a non-stationary stream, a quotient regular stream, a choice-selected limit, or an ambient completeness theorem.

Theorem 897.2 (Stationary window naturality NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryWindowNaturality}^{\uparrow}}$ *surface for Definition 897.1 consists exactly of carrier admission for $N = (Q, S, R, D, E, W, H, C, P, K)$, seed classifier exposure through Q, constant finite-window exposure through S, regular-sequence seal exposure through R, dyadic observation exposure through D, real finite-window seal exposure through E, naturality comparison exposure through W, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through K.*

Proof. Project the carrier. The rows $Q, S, R, D, E, W, H, C, P, K$ are the only coordinates admitted to the packet, so every local naming obligation must be read from one of these rows.

The four boundary readings are tied to the seed in order. The stationary diagonal window route supplies S from Q, the regular-sequence row R consumes the same window, the dyadic ledger D records the finite observations attached to that window, and the real seal row E is reached only after S, R, D are displayed.

The naturality row W is the comparison row that keeps these readings together. It is not a host equality assertion; it is the finite statement that the StreamName, RegSeqRat, DyadicRatCore, and RealUp boundary reads are the same transported stationary seed route when inspected through H, C, P, K.

Since no excluded object is a coordinate of N, the NameCert surface is exhausted by the displayed rows. □

Theorem 897.3 (Stationary window naturality boundary comparison)**.** *For an accepted* StationaryWindowNaturalit *carrier $N = (Q, S, R, D, E, W, H, C, P, K)$, every stationary finite-window consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal row E must pass through the same seed classifier Q, constant* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, naturality row W, and structural rows H, C, P, K.*

Proof. Begin with the route sibling. The stationary diagonal window route exposes the seed-to-window path, so a Real-facing read cannot reach the stream window without first reading Q and S.

Next use the regular-Cauchy window fusion sibling. It consumes the displayed stream window as the common source for the regular readback R, dyadic ledger D, and real seal row E.

The comparison row W then records that these boundary readings are not separate analytic choices. They are four readings of the same transported seed route, replayed through the componentwise transports, continuations, package row, and local naming row. □

Theorem 897.4 (Stationary window naturality downstream package)**.** *The carrier, NameCert obligations, and boundary comparison rows of* $\mathsf{StationaryWindowNaturality}^{\uparrow}$ *assemble the finite naturality package consumed by transported stationary diagonal, stationary diagonal window route, Cauchy modulus extraction, regular Cauchy window fusion, and* $\mathsf{RealCompleteness}^{\uparrow}$ *consumers. The package exports exactly*

$$Q, S, R, D, E, W, H, C, P, K$$

over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. The carrier supplies the seed row, the four finite-window boundary readings, the comparison row, and the structural support rows. The NameCert obligation theorem records these as the complete local naming surface.

The boundary comparison theorem forces every Real-facing read through the same finite route. Thus downstream consumers receive a naturality packet, not independent analytic witnesses for the StreamName, RegSeqRat, DyadicRatCore, and RealUp boundaries.

The listed sibling chapters provide the incoming surfaces. Transported stationary diagonal supplies the seed classifier, stationary diagonal window route supplies the seed-to-window path, Cauchy modulus extraction supplies the finite observation-budget discipline, and regular Cauchy window fusion supplies the shared readback and seal route. □

Theorem 897.5 (Stationary window naturality TasteGate readiness)**.** *The Lean TasteGate site for* StationaryWindowNaturality$^{\uparrow}$ *must distinguish the naturality comparison row W from the four boundary readings S, R, D, E and must preserve all rows $Q, S, R, D, E, W, H, C, P, K$ for classifier-facing consumers.*

Proof. Nontriviality is forced by the packet shape. If W collapsed to any one of S, R, D, E, the carrier would lose the comparison row that ties the four boundary readings to the same seed classifier Q.

Field faithfulness is the consumer-facing preservation condition. A classifier-facing route may inspect the seed, stream window, regular readback, dyadic ledger, real seal, naturality row, transports, continuations, package provenance, and local naming row, but no unlisted real-completion object. □

Lean4 ✓ `BEDC.Derived.StationaryWindowNaturalityUp.StationaryWindowNaturalityTasteGate single carrier alignment`

Closure status for NameCert$_{\text{StationaryWindowNaturality}^{\uparrow}}$.

Constructive story: A StationaryWindowNaturality$^{\uparrow}$ packet is generated from one transported stationary Rat$^{\uparrow}$ seed classifier, its constant StreamName$^{\uparrow}$ finite window, the matching RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ observation ledger, Real$^{\uparrow}$ seal, naturality comparison row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{StationaryWindowNaturality}^{\uparrow}}$, seed)

Scope closed: Definition 897.1, Definition 897.2, Definition 897.3, and Definition 897.4 bind the finite naturality packet for transported stationary window readings.

Formal verification: FormalStatus(NameCert$_{\text{StationaryWindowNaturality}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove RealUp completeness, quotient-stream equality, selected limits, host real equality, real trichotomy, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, boundary-reading exactness, naturality comparison preservation, downstream package coverage, and TasteGate row-faithfulness for the same finite packet.

Chapter 898

A Concrete Naming Certificate for $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$

The $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ packet is the composite finite witness envelope for the Real-completeness diagonal tail route of chapter 9238. It consumes the stream window surface of chapter 126, the finite-window envelope of chapter 365, and the common-tail comparison packet of chapter 387. The packet keeps the witness route finite by storing the selected StreamName window, the finite-window endpoint ledger, the Cauchy tail-comparison row, the RealTailAgreementSeal row, the RealUp readback row, and the structural provenance needed to replay the same route.

Definition 898.1 (CauchyTailWitnessEnvelope carrier)**.** *A* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, F, Q, A, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected for the diagonal tail request;*
2. *F is the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row that binds the selected window to its dyadic endpoint ledger, regular-rational readback, and RealUp endpoint surface;*
3. *Q is the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row that compares the relevant late regular-name windows through the same selected window and endpoint ledger;*
4. *A is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row reached after the finite-window and comparison rows have both been displayed;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row exported to the diagonal Real-completeness route;*
6. *H records componentwise* hsame *transports for S, F, Q, A, R, C, P, N;*
7. *C records the displayed* Cont *routes that replay the selected window, finite-window envelope, tail comparison, tail-agreement seal, RealUp readback, provenance, and local naming row;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}$ *row.*

The carrier contains no quotient stream equality, no host real equality, no selected host limit, no countable-choice selector, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyTailWitnessEnvelopeUp.CauchyTailWitnessEnvelopeTasteGate single carrier alignment`

Theorem 898.2 (CauchyTailWitnessEnvelope NameCert obligations)**.** *For the carrier of Definition 898.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (S, F, Q, A, R, H, C, P, N),$$

selected $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*, finite-window envelope exposure through* F*, common-tail comparison exposure through* Q*, RealTailAgreementSeal exposure through* A*, RealUp readback exposure through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows $S, F, Q, A, R, H, C, P, N$ are the only coordinates of the finite packet.

Read the analytic rows in order. The selected stream window S is first bound into the finite-window envelope F, and the tail comparison Q reads its common window and dyadic endpoint ledger through that same envelope surface.

The tail-agreement seal A and RealUp readback R are then reached only after S, F, Q have been displayed. The structural rows H, C, P, N transport, replay, authenticate, and name the same packet, so the listed rows exhaust the local NameCert surface. □

Theorem 898.3 (CauchyTailWitnessEnvelope selector handoff)**.** *Every diagonal-tail selector read exported by an accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *carrier factors through the selected window* S*, the finite-window envelope* F*, and the Cauchy tail-comparison row* Q *before the RealTailAgreementSeal row* A *or the RealUp readback row* R *can be consumed. The handoff supplies the finite witness route used by chapter 9238, not an isolated RealUp seal or a host limit.*

Proof. Start with the selected window. The carrier has exactly one StreamName window coordinate S, and that coordinate is the source read by the envelope row F.

Use the finite-window dependency next. The row F supplies the selected window and endpoint ledger consumed by the comparison row Q, so the comparison is not a host-level comparison detached from the envelope.

Finally read A and R. These rows are reached after the selector, envelope, and comparison rows have been replayed through C and transported through H. Thus the diagonal-tail handoff remains a finite witness envelope rather than a selected limit or isolated RealUp object. □

Theorem 898.4 (CauchyTailWitnessEnvelope non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet receives only the finite surface*

$$S, F, Q, A, R, H, C, P, N.$$

It receives no quotient stream equality, host real equality, selected host limit, countable-choice selector, or ambient completeness theorem. A later Lean TasteGate module for this carrier must therefore expose nontriviality and field-faithfulness over the displayed packet fields rather than over a collapsed RealUp output.

Proof. Project the carrier and apply the NameCert obligation theorem. The public surface is exhausted by the selected window, finite-window envelope, common-tail comparison, tail-agreement seal, RealUp readback, transport, replay, provenance, and local naming rows.

The selector handoff theorem keeps every diagonal-tail read on those rows. In particular, the comparison row Q consumes the finite-window endpoint ledger from F, and the seal row A is unavailable before that route is displayed.

The excluded objects have no coordinate in the packet. Since the carrier has no quotient stream, host real equality, host-selected limit, choice selector, or ambient completeness row, no downstream consumer can recover one from an accepted envelope. □

Theorem 898.5 (CauchyTailWitnessEnvelope finite-window exactness)**.** *For every accepted* CauchyTailWitnessEnv *carrier*

$$E = (S, F, Q, A, R, H, C, P, N),$$

the selected $\mathsf{StreamName}^{\uparrow}$ *window S is consumed through the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row F before any tail-comparison, seal, or RealUp read is exported. The endpoint ledger, regular-rational readback, and RealUp endpoint surface used by the envelope are therefore the ones displayed by F.*

Proof. Project the carrier and use the NameCert obligation theorem. The only selected-window coordinate is S, and the only finite-window envelope coordinate is F.

Apply the selector handoff theorem. It places S before F and F before Q, A, R, so no later row can read the selected window except through the envelope.

Finally use non-escape. Since the packet has no hidden endpoint ledger or host equality coordinate, the endpoint data consumed downstream are exactly the ledger rows displayed by F. □

Theorem 898.6 (CauchyTailWitnessEnvelope tail-comparison exactness)**.** *For every accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *carrier $E = (S, F, Q, A, R, H, C, P, N)$, the tail-comparison row Q reads the same finite-window endpoint ledger exposed by F before any RealTailAgreementSeal row A is consumed. No comparison row may bypass the selected window S or replace the endpoint ledger by quotient stream equality or host real equality.*

Proof. Start from finite-window exactness. By Definition 898.5, the selected window S and endpoint ledger are already bound through F.

Use the NameCert obligation theorem to locate Q. The comparison row is a displayed carrier coordinate and its consumer route follows the selected window and finite-window envelope rows.

Apply the selector handoff theorem. It places Q before A and R, so the seal cannot be reached until the comparison has read the same S, F ledger.

The non-escape theorem removes the remaining alternatives. Quotient streams, host real equality, selected host limits, countable choice, and ambient completeness are not packet coordinates, so they cannot replace the finite comparison ledger. □

Theorem 898.7 (CauchyTailWitnessEnvelope Real tail seal boundary)**.** *For every accepted* CauchyTailWitnessEnve *carrier $E = (S, F, Q, A, R, H, C, P, N)$, the RealTailAgreementSeal row A and the RealUp readback row R are reached only after the selected window S, finite-window envelope F, tail-comparison row Q, transport rows H, continuation rows C, provenance row P, and local naming row N have been displayed.*

The boundary is a finite witness-envelope route. It does not export quotient stream equality, host real equality, a selected host limit, a countable-choice selector, or ambient completeness.

Proof. Project the accepted packet. The carrier has exactly the rows $S, F, Q, A, R, H, C, P, N$, with H, C, P, N carrying transport, replay, provenance, and local naming for the same packet.

Apply finite-window exactness and tail-comparison exactness. These theorems route the seal-facing read through S, F, Q before A is available.

Use the selector handoff theorem for the final order of consumption. It places A and then R after the selected window, envelope, and comparison rows, while the structural rows replay and authenticate the same route.

Finally apply non-escape. Since no excluded object is present in the carrier, the seal boundary is exactly the displayed finite witness-envelope route. □

Theorem 898.8 (CauchyTailWitnessEnvelope obligation closure)**.** *For every accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet* $E = (S, F, Q, A, R, H, C, P, N)$*, the carrier definition, local* NameCert *obligations, selector handoff, finite-window exactness, tail-comparison exactness, Real-tail seal boundary, and non-escape theorem jointly form the* obligation *surface over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FiniteWindowEnvelope}^{\uparrow}$, $\mathsf{CauchyTailComparison}$ $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package exports exactly the selected stream-name window, finite endpoint envelope, Cauchy tail-comparison ledger, RealTailAgreementSeal boundary, RealUp readback row, transport rows, continuation routes, package provenance, and local naming row displayed by E.

Proof. Project the tuple $E = (S, F, Q, A, R, H, C, P, N)$. The carrier definition supplies the row names, and the local NameCert obligations certify that each downstream consumer reads those rows as a single finite witness-envelope packet.

Follow the route order. The selector handoff places S before F, Q, A, R, finite-window exactness binds the endpoint ledger through S, F, tail-comparison exactness makes Q read that same ledger before A, and the Real-tail seal boundary places A and R after the selected window, envelope, comparison, transport, continuation, provenance, and local naming rows.

Finally apply non-escape. Quotient stream equality, host equality, a host-selected limit, a choice selector, and ambient completeness are not coordinates of E, so no hidden row remains outside the displayed obligation surface. □

Theorem 898.9 (CauchyTailWitnessEnvelope scoped route binding)**.** *Every accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet*

$$E = (S, F, Q, A, R, H, C, P, N)$$

exposes the selected $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *row* F*, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row* Q*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* A*, the* $\mathsf{Real}^{\uparrow}$ *readback row* R*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and the local* NameCert *row* N *as one finite L10 witness-envelope route over* BHist, hsame, Cont, Pkg, *and* NameCert.

The route exposes no quotient stream equality, host real equality, selected host limit, countable-choice selector, or ambient completeness row.

Proof. Start from the local NameCert obligations. They identify the displayed coordinates $S, F, Q, A, R, H, C, P, N$ as the entire accepted packet surface, and the TasteGate marker for $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ aligns this single carrier with its named field-faithful carrier route.

Use the selector handoff to place the route on the selected window. The row S is consumed by the finite-window envelope F, and F supplies the endpoint ledger that the tail-comparison row Q reads.

Apply finite-window exactness and tail-comparison exactness. These results make the selected window, endpoint envelope, and Cauchy tail-comparison ledger the same finite route before the RealTailAgreementSeal row is available.

Finally use the Real-tail seal boundary and non-escape theorem. The seal row A, RealUp readback row R, transports H, continuations C, provenance P, and naming row N remain inside the same finite packet, while quotient stream equality, host real equality, selected host limits, countable choice, and ambient completeness have no carrier coordinate. □

Theorem 898.10 (CauchyTailWitnessEnvelope consumer interface exactness)**.** *For every accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet*

$$E = (S, F, Q, A, R, H, C, P, N)$$

a public consumer of $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *has exactly the finite interface consisting of the selected* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *endpoint row* F*, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row* Q*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *boundary row* A*, the* $\mathsf{Real}^{\uparrow}$ *readback row* R*, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by* H, C, P, N*.*

Conversely, the displayed rows $S, F, Q, A, R, H, C, P, N$ *suffice for the public consumer interface. They export no quotient stream equality, host real equality, host-selected limit, countable-choice selector, or ambient completeness principle.*

Proof. For exactness, project the carrier $E = (S, F, Q, A, R, H, C, P, N)$. By the NameCert obligation theorem, these are exactly the rows admitted for the local $\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}$ surface, so a public consumer has no off-packet coordinate before the route is inspected.

Use the selector handoff and finite-window exactness. The selected window S is first read through the envelope F, and the endpoint ledger exported to the consumer is the one displayed by that same finite-window row.

Next apply tail-comparison exactness. The row Q reads the same S, F ledger before the Real-tail seal can be consumed, so the comparison branch cannot replace the finite ledger by quotient stream equality or host real equality.

Now use the Real-tail seal boundary, obligation closure, and scoped route binding. The rows A and R are reached only after S, F, Q, H, C, P, N have been displayed, and the structural rows transport, replay, package, and name that same finite route.

For sufficiency, start with the displayed rows $S, F, Q, A, R, H, C, P, N$. The NameCert obligations admit them as one local naming surface, the selector handoff places the analytic rows in the required order, and the finite-window, tail-comparison, and seal-boundary theorems provide the endpoint, comparison, boundary, and readback components of the consumer route.

Finally apply non-escape and scoped route binding. Since the accepted packet has no coordinate for a quotient stream, host real equality, host-selected limit, countable-choice selector, or ambient completeness principle, the displayed rows provide the whole public interface and no stronger object is exported. □

Theorem 898.11 (CauchyTailWitnessEnvelope standard bridge surface)**.** *Every standard-facing read of an accepted* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *packet* $E = (S, F, Q, A, R, H, C, P, N)$ *factors through the selected* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{FiniteWindowEnvelope}^{\uparrow}$ *endpoint row* F*, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row* Q*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *boundary row* A*, the* $\mathsf{Real}^{\uparrow}$ *readback row* R*, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows carried by* H, C, P, N*.*

The bridge surface exports no quotient stream equality, host real equality, host-selected limit, countable-choice selector, or ambient completeness coordinate.

Proof. Begin with the local NameCert obligations of Definition 898.2. They identify $S, F, Q, A, R, H, C, P, N$ as the complete accepted carrier surface for the local $\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}$ row.

Use the selector handoff theorem Definition 898.3. It places every diagonal-tail read first on the selected $\mathsf{StreamName}^{\uparrow}$ window and then on the finite-window envelope before the comparison, seal, or readback rows are consumed.

Apply finite-window exactness and tail-comparison exactness from Definition 898.5 and Definition 898.6. These results bind the endpoint ledger and comparison row to the same finite S, F surface.

Next use the Real-tail seal boundary of Definition 898.7. The rows A and R are reached only after the selected window, finite-window envelope, comparison row, transports, continuations, provenance, and local name have been displayed.

Now combine non-escape, scoped route binding, and consumer interface exactness from Definition 898.4, Definition 898.9, and Definition 898.10. Together they state that a public or standard-facing consumer has exactly the listed finite interface and no off-packet coordinate.

Therefore the standard bridge surface is precisely the displayed $S, F, Q, A, R, H, C, P, N$ route, with no quotient stream equality, host real equality, host-selected limit, countable-choice selector, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ packet is generated from a selected $\mathsf{StreamName}^{\uparrow}$ window, a finite-window envelope row, a Cauchy tail-comparison row, a $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ readback row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 898.1, Definition 898.2, Definition 898.3, Definition 898.5, Definition 898.6, Definition 898.7, Definition 898.4, Definition 898.9, the bridge-facing public consumer surface Definition 898.10, and the standard bridge surface Definition 898.11 bind the selected StreamName window, finite-window envelope, tail-comparison row, RealTailAgreementSeal row, RealUp readback row, BHist carrier surface, hsame transports, Cont routes, Pkg provenance, NameCert row, public consumer interface, and standard-facing read as one finite bridge-facing surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyTailWitnessEnvelopeUp.CauchyTailWitnessEnvelopeTasteGate\_single\_carrier\_alignm`

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.CauchyTailWitnessEnvelopeUp.taste gate`

Not claimed: This chapter does not close quotient streams, host real equality, selected host limits, countable choice, ambient completeness, or machine bridge checking.

Upgrade path: No further paper-side closure upgrade remains; machine bridge checking stays on the formal axis.

Chapter 899

A Concrete Naming Certificate for $\mathsf{MetaClosureObstruction}^{\uparrow}$

The $\mathsf{MetaClosureObstruction}^{\uparrow}$ packet realizes the forbidden-schema side of chapter 9458. It records the permitted refutation route, the forbidden contradiction-schema request, the certificate-indexed truth-branch refusal row, the meta-loop boundary row, and the local naming ledger that separates ground-loop refutation from truth closure over the carrier of naming certificates.

The packet consumes the permitted refutation route of chapter 503, the truth-branch refusal boundary of chapter 854, the reflection limit row of chapter 586, the ground-loop boundary of chapter 674, and the halting distinction limit of chapter 495. These dependencies remain dependency prose for the local obstruction packet; they do not raise the chapter beyond its stated seed closure.

Definition 899.1 (MetaClosureObstruction carrier)**.** *A* $\mathsf{MetaClosureObstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, S, F, T, M, H, C, P, N)$$

with the following displayed rows:

1. *R is the permitted refutation route, read as a displayed assumption-to-bottom* Cont *packet over a fixed certificate surface;*
2. *S is the forbidden contradiction-schema request, naming one of existential-by-contradiction, double-negation elimination without a decidability row, or excluded middle over the carrier of naming certificates;*
3. *F is the refusal ledger for the requested schema, recording that the request is not an admitted certificate rule;*
4. *T is the truth-branch refusal row for certificate-indexed predicates;*
5. *M is the meta-loop boundary row separating the closed ground loop from the open truth-closure layer;*
6. *H records componentwise* hsame *transports for R, S, F, T, M, P, N;*
7. *C records the displayed* Cont *routes by which consumers read the refutation route, forbidden-schema request, refusal rows, and meta-loop boundary;*
8. *P is the* Pkg *provenance over* $\mathsf{WitnessedRefutation}^{\uparrow}$, $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$, $\mathsf{ReflectionLimitBoundary}^{\uparrow}$, $\mathsf{GroundLoopBoundary}^{\uparrow}$, *and* $\mathsf{HaltingDistinctionLimit}^{\uparrow}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetaClosureObstruction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by the displayed rows $R, S, F, T, M, H, C, P, N$. In particular, an accepted packet lacking the truth-branch refusal row T is not a $\mathsf{MetaClosureObstruction}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionTasteGate single carrier alignment`

Theorem 899.2 (MetaClosureObstruction NameCert obligations)**.** *For the carrier of Definition 899.1, the local* $\mathsf{NameCert}_{\mathsf{MetaClosureObstruction}^{\uparrow}}$ *surface consists of carrier admission for*

$$O = (R, S, F, T, M, H, C, P, N),$$

permitted-refutation exposure through R, forbidden-schema request exposure through S, schema refusal through F, certificate-indexed truth-branch refusal through T, meta-loop boundary exposure through M, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 899.1. The tuple displays the permitted refutation route, the forbidden-schema request, the two refusal rows, the meta-loop boundary, transport data, continuation data, provenance, and local naming row.

The refutation row R is ground-facing: it is read only as an assumption-to-bottom route inside a fixed certificate surface. The schema row S is boundary-facing: it names the forbidden move being requested but does not supply its constructive witness, decidability row, or certificate-indexed branch family.

The refusal rows F and T are distinct. Row F refuses the local schema request, while row T refuses the truth-branch family over naming certificates that the forbidden schema would supply. Row M then records that this refusal is the open meta-loop boundary rather than a missing ground-loop constructor.

Transport and replay are exhausted by H and C, and provenance is exhausted by P, N. Since no further coordinate exists, the listed rows are exactly the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionNameCertObligations`

Theorem 899.3 (MetaClosureObstruction non-escape)**.** *Every accepted* $\mathsf{MetaClosureObstruction}^{\uparrow}$ *packet exports only the finite rows*

$$R, S, F, T, M, H, C, P, N.$$

No consumer can extract from such a packet an existential witness obtained from $\neg\forall\neg$, a double-negation eliminator for an undecided classifier, an excluded-middle branch over naming certificates, a certificate-indexed truth predicate, or meta-loop truth closure. Any accepted packet lacking the truth-branch refusal row T fails to be a $\mathsf{MetaClosureObstruction}^{\uparrow}$ *carrier.*

Proof. Project the accepted packet to the carrier inventory of Definition 899.1. The only positive proof route is the permitted refutation row R, and the only forbidden-schema information is the request row S together with the refusal rows F, T and the boundary row M.

The NameCert obligations of Definition 899.2 force every consumer read through H, C, P, N. Thus a consumer may replay the displayed refutation and inspect the refusal boundary, but no route bypasses F, T, M to produce a truth branch or a closed meta loop.

The last claim follows from the carrier definition itself. Row T is one of the displayed coordinates of O, and the classifier compares accepted packets only through all displayed rows. Removing T removes the certificate-indexed truth-branch refusal coordinate, so the remaining packet is not a $\mathsf{MetaClosureObstruction}^{\uparrow}$ carrier. □

Independence witness: WitnessedRefutation, CertificateTruthBranchBoundary, ReflectionLimitBoundary, GroundLoopBoundary, HaltingDistinctionLimit: the obstruction packet contains both a forbidden-schema request row and a truth-branch refusal row. WitnessedRefutation carries the permitted refuter without the forbidden-schema packet; CertificateTruthBranchBoundary carries the branch boundary without the meta-loop obstruction packet; ReflectionLimitBoundary carries structural reflection limits without a contradiction-schema request; GroundLoopBoundary carries closed mark distinction without certificate-indexed truth refusal; HaltingDistinctionLimit carries a halting diagonal obstruction without the three-schema refusal ledger. Hence none is bijective to $\mathsf{MetaClosureObstruction}^{\uparrow}$ while preserving NameCert obligations.

Theorem 899.4 (MetaClosureObstruction truth-branch obligation)**.** *Let $O = (R, S, F, T, M, H, C, P, N)$ be an accepted* $\mathsf{MetaClosureObstruction}^{\uparrow}$ *carrier. Before any consumer may compare the obstruction packet with certificate-indexed truth closure, the truth-branch refusal row T must be displayed and transported with the permitted refutation route R, forbidden-schema request S, schema refusal row F, meta-loop boundary M, componentwise* hsame *data H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionCarrier truth branch obligation`

Proof. Project the accepted carrier O. The truth-branch refusal row T is a displayed coordinate of the same finite BHist packet as R, S, F, M, H, C, P, N.

A consumer comparing obstruction with certificate-indexed truth closure must first read T. Without that row, the carrier definition supplies only a permitted refutation route and a schema refusal, not the truth-branch refusal needed at the certificate-indexed boundary.

Transport and replay preserve this order. The row H carries T componentwise with the other displayed rows, C replays the same refusal route, and P, N source and name it as part of the local NameCert surface.

Thus the comparison obligation is not a verification claim and not a truth-closure rule. It is the paper-side requirement that T remain present before any truth-branch consumer can inspect the obstruction packet. □

Lean4 ✓ `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionTruthBranchObligation`

Theorem 899.5 (MetaClosureObstruction scoped refusal route)**.** *Let $O = (R, S, F, T, M, H, C, P, N)$ be an accepted* $\mathsf{MetaClosureObstruction}^{\uparrow}$ *carrier. The scoped refusal route exposed by O depends only on the permitted refutation route R, forbidden contradiction-schema request S, schema refusal row F, truth-branch refusal row T, meta-loop boundary M, componentwise* hsame *data H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. It refuses contradiction-schema closure and certificate-indexed truth predicates as carrier-external requests.*

Proof. Project the accepted packet. The classifier compares only the displayed rows $R, S, F, T, M, H, C, P, N$, so the scoped route has no entry point outside that finite inventory.

Read the refusal rows in order. Row F refuses the requested contradiction schema, and row T refuses the certificate-indexed truth branch that such a schema would create over naming certificates.

The structural rows preserve only this route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row keep the refusal surface named without adding contradiction-schema closure or certificate-indexed truth predicates. □

Theorem 899.6 (MetaClosureObstruction public export)**.** *Let $O = (R, S, F, T, M, H, C, P, N)$ be an accepted* MetaClosureObstruction$^{\uparrow}$ *carrier. Its public obstruction export is exactly the row inventory $R, S, F, T, M, H, C, P, N$: the permitted refutation route, forbidden-schema request, schema refusal, truth-branch refusal, meta-loop boundary, componentwise* hsame *data, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. Public consumers may cite this obstruction only as a finite refusal packet; they do not receive contradiction-schema closure, certificate-indexed truth predicates, or a public standard bridge.*

Proof. Project the carrier of Definition 899.1. The classifier compares accepted packets only through $R, S, F, T, M, H, C, P, N$, so the public surface cannot contain a row outside that tuple.

Apply Definition 899.5. The scoped refusal route already exposes contradiction-schema refusal and certificate-indexed truth-branch refusal through the same displayed rows.

Thus the public export is only the visible obstruction packet. It preserves the permitted refutation route and refusal boundary while leaving truth closure, excluded middle, double-negation elimination, and bridge checking outside the chapter. □

Lean4 ✓ `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionPublicExport`

Theorem 899.7 (MetaClosureObstruction mature refusal completeness)**.** *Let $O = (R, S, F, T, M, H, C, P, N)$ be an accepted* MetaClosureObstruction$^{\uparrow}$ *carrier. The mature obstruction surface is exhausted by the permitted refutation route R, forbidden-schema request S, schema refusal F, truth-branch refusal T, meta-loop boundary M, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. It exports no contradiction-schema closure and no certificate-indexed truth predicate.*

Proof. Project the carrier of Definition 899.1. Its displayed rows are exactly $R, S, F, T, M, H, C, P, N$, so any mature obstruction read starts inside that finite inventory.

Apply Definition 899.2 and Definition 899.6. The NameCert surface and public export both require the same route through the refutation, request, refusal, transport, replay, provenance, and local naming rows.

Finally use Definition 899.3 and Definition 899.5. They rule out contradiction-schema closure and certificate-indexed truth predicates as carrier-external requests, so no further mature row remains. □

Closure status for NameCert$_{\text{MetaClosureObstruction}^{\uparrow}}$**.**

Constructive story: A MetaClosureObstruction$^{\uparrow}$ packet is generated from a permitted refutation route, a forbidden contradiction-schema request, schema and truth-branch refusal rows, a meta-loop boundary row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaClosureObstruction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 899.1, Definition 899.2, Definition 899.3, Definition 899.4, Definition 899.6, and Definition 899.7 bind the finite mature obstruction packet separating permitted refutation from meta-loop truth closure.

Formal verification: FormalStatus(NameCert$_{\text{MetaClosureObstruction}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.MetaClosureObstructionUp.MetaClosureObstructionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the forbidden contradiction schemas, certificate-indexed truth predicates, excluded middle over naming certificates, or a public standard bridge.

Upgrade path: The next verification grade requires axiom-purity audit and bridge checking for the same TasteGate-aligned obstruction surface.

Chapter 900

A Concrete Naming Certificate for RealCompletionSealClassifierUp

The RealCompletionSealClassifierUp packet realizes the Real completeness vision of chapter 9238 as a finite classifier boundary for completion-seal reads. It joins the tail-agreement seal of RealTailAgreementSeal$^{\uparrow}$, the sealed Cauchy-limit route of CauchyLimitSeal$^{\uparrow}$, and the synchronized budget route of chapter 886 before a Real$^{\uparrow}$ consumer reads equivalence of completion-facing seals. The packet has no coordinate for quotient streams, host real equality, selected external limits, countable choice, or ambient completeness.

Definition 900.1 (Real completion-seal classifier carrier)**.** *A* RealCompletionSealClassifierUp *carrier is a finite* BHist *packet*

$$K = (A, L, S, B, R, W, D, H, C, P, N)$$

with the following displayed rows:

1. *A is the* RealTailAgreementSeal$^{\uparrow}$ *row naming the tail-agreement comparison used by the completion read;*
2. *L is the* CauchyLimitSeal$^{\uparrow}$ *row whose public route exposes the sealed Cauchy-limit endpoint;*
3. *S is the* CauchySealBudgetSynchronizer$^{\uparrow}$ *row tying the seal read to the displayed selector, observation-budget, shared-threshold, and compatibility rows;*
4. *B is the finite budget row read from the synchronizer, including the displayed precision request and selected observation window;*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row assembled from the displayed regular rational source data;*
6. *W is the* StreamName$^{\uparrow}$ *finite window row inspected before the regular handoff is consumed;*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance and observation ledger used by the same completion-seal read;*
8. *H records componentwise* hsame *transports for $A, L, S, B, R, W, D, C, P, N$;*
9. *C stores the displayed* Cont *routes by which consumers read the agreement seal, limit seal, synchronizer, budget, regular handoff, stream window, dyadic ledger, provenance, and local naming row;*
10. *P is the* Pkg *provenance over* RealTailAgreementSeal$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, CauchySealBudgetSynchronizer$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSealClassifierUp}}$ *row.*

The carrier classifies completion-seal equivalence only through these displayed rows. It does not identify real names by quotient streams, host equality, selected external limits, countable choice, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.RealCompletionSealClassifierUp.RealCompletionSealClassifierUp`
Lean4 ✓ `BEDC.Derived.RealCompletionSealClassifierUp.RealCompletionSealClassifierTasteGate`
`single carrier alignment`

Theorem 900.2 (Real completion-seal classifier NameCert obligations)**.** *For the carrier of Definition 900.1, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSealClassifierUp}}$ *surface consists of five rows:*

1. *carrier admission for the finite packet $K = (A, L, S, B, R, W, D, H, C, P, N)$;*
2. *completion-seal classifier stability under the componentwise* hsame *transports in H;*
3. *displayed seal equivalence through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows A, L;*
4. *budget and synchronizer exactness through S, B, W, R, D before any* $\mathsf{Real}^{\uparrow}$ *seal read is exported;*
5. *provenance and naming exactness through the displayed* Cont, Pkg, *and* $\mathsf{NameCert}$ *rows C, P, N.*

Later Lean work in `TasteGate.lean` *must provide* `Nontrivial RealCompletionSealClassifierUp` *and* `FieldFaithful RealCompletionSealClassifierUp` *instances for this same carrier surface.*

Proof. Carrier admission is projection from the tuple in Definition 900.1. The agreement seal, limit seal, synchronizer, budget, regular handoff, stream window, dyadic ledger, transport, route, provenance, and local naming rows are all explicit coordinates.

Classifier stability is componentwise. The row H transports the agreement, seal, synchronizer, budget, regular handoff, stream window, and dyadic observation rows without creating a quotient stream or host equality comparison.

Displayed seal equivalence is read through A and L. The tail-agreement row supplies the finite comparison surface, and the Cauchy-limit seal row supplies the completion-facing endpoint route.

Budget exactness is read through S, B, W, R, D. The synchronizer forces the seal read to pass through a displayed budget row, a finite stream window, the regular rational handoff, and the dyadic tolerance ledger before a $\mathsf{Real}^{\uparrow}$ consumer may inspect the completion seal.

Finally, the routes in C and the provenance and naming rows P, N authenticate exactly that finite surface. Thus the five obligations are the whole classifier interface. □

Theorem 900.3 (Real completion-seal classifier non-escape)**.** *Every accepted* $\mathsf{RealCompletionSealClassifierUp}$ *read factors through the displayed agreement, seal, synchronizer, budget, regular handoff, stream-window, dyadic, transport, route, provenance, and naming rows*

$$A, L, S, B, R, W, D, H, C, P, N.$$

No accepted read may bypass both a displayed seal row and a displayed budget or synchronizer row while still exporting a completion-seal classifier to a $\mathsf{Real}^{\uparrow}$ *consumer.*

Proof. Project the accepted carrier to $K = (A, L, S, B, R, W, D, H, C, P, N)$. The only seal-facing rows are A and L, and the only budget-facing synchronization rows are S, B, W, R, D.

Apply the NameCert obligation surface. It requires displayed seal equivalence through A, L and budget-synchronizer exactness through S, B, W, R, D before the route can be authenticated by C, P, N.

The componentwise transports in H preserve those rows and do not create a second completion channel. Since the carrier contains no quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness theorem, a read that bypasses both the seal rows and the budget or synchronizer rows has no remaining coordinate from which to export a classifier. □

Theorem 900.4 (Real completion-seal classifier stability)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* RealCompletionSealClassifierUp *carrier. Componentwise* hsame *transport along the rows recorded in H preserves the classifier read on the agreement seal, limit seal, synchronizer, budget, regular handoff, stream-window, dyadic ledger, displayed routes, provenance, and local naming coordinates of K. In particular, a transported accepted read still classifies completion-seal equivalence only through those displayed coordinates.*

Proof. Project the accepted carrier to its tuple K. The transport row H is a coordinate of the same carrier, so its componentwise hsame data is read only against the displayed rows $A, L, S, B, R, W, D, C, P, N$.

Use the obligation surface of Definition 900.2. Its stability row is exactly the requirement that the classifier surface remains the same classifier after those componentwise transports.

Finally use the non-escape boundary of Definition 900.3. Since no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness theorem is present in K, the transported read has no second completion channel. □

Theorem 900.5 (Real completion-seal classifier seal equivalence)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* RealCompletionSealClassifierUp *carrier. Before any* $\mathsf{Real}^{\uparrow}$ *consumer receives a seal classifier, completion-seal equivalence is read only through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row A and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L, together with the displayed route row C that exposes those two rows to the consumer.*

Proof. Project the carrier to K. The only agreement-facing coordinate is A, and the only completion-limit-facing coordinate is L.

Invoke the NameCert obligations of Definition 900.2. The displayed seal-equivalence row requires the classifier to pass through A and L before the route data in C can authenticate a $\mathsf{Real}^{\uparrow}$ consumer read.

The non-escape theorem then rules out every bypass. A read that does not use A and L has no displayed seal row left in the carrier from which to export a completion-seal classifier. □

Theorem 900.6 (Real completion-seal classifier budget exactness)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* RealCompletionSealClassifierUp *carrier. The synchronizer and budget rows S, B, W, R, D are exact prerequisites for the seal read: the classifier must pass through the synchronizer, finite budget, finite stream window, regular rational handoff, and dyadic tolerance ledger before any completion-seal classifier is exported. This prerequisite surface excludes quotient streams, host real equality, selected external limits, countable choice, and ambient completeness.*

Proof. Project the accepted read to the carrier tuple. The synchronizer, budget, stream-window, regular handoff, and dyadic rows are precisely S, B, W, R, D.

Apply the obligation surface. Its budget and synchronizer exactness row says that these five coordinates are prerequisites for the seal read before the routes, provenance, and local naming rows can authenticate an exported classifier.

Apply the non-escape boundary. If a read bypasses the displayed budget or synchronizer surface, then the carrier has no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness theorem that could replace it. □

Theorem 900.7 (Real completion-seal classifier tail-radius handoff)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* RealCompletionSealClassifierUp *carrier. Any accepted seal read whose budget row opens a tail-radius request must factor through the* DyadicTailRadiusLedger$^{\uparrow}$ *tail-budget coverage of Definition 757.3 and then through the* RegSeqRat$^{\uparrow}$–Real$^{\uparrow}$ *handoff of Definition 757.6 before any* Real$^{\uparrow}$ *consumer receives the classifier. The route is the displayed carrier route through $S, B, W, R, D, H, C, P, N$; it exports no quotient stream, host equality, selected external limit, countable choice, or ambient completeness principle.*

Proof. Project the accepted read to the carrier tuple K. The rows S, B, W, R, D are the only synchronizer, budget, finite window, regular rational handoff, and dyadic tolerance rows available before H, C, P, N authenticate a consumer-facing classifier.

Use budget exactness from Definition 900.6. Once the budget row opens a tail-radius request, the read is already constrained to the displayed route through S, B, W, R, D and cannot reach the exported classifier directly.

Now use the sibling ledger. The tail-budget coverage theorem, Definition 757.3, places the tail-radius request inside the finite precision, tail-window, StreamName-window, and dyadic readback rows, and Definition 757.6 carries that same route through the RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ handoff rows.

Finally repack the resulting route through H, C, P, N. Since neither K nor the dyadic tail-radius ledger contains quotient streams, host equality, selected external limits, countable-choice rows, or ambient completeness theorems, the classifier received by the Real$^{\uparrow}$ consumer is the ledger-mediated classifier. □

Theorem 900.8 (Real completion-seal classifier window factorization)**.** *Every exported* RealCompletionSealClassifie *classifier window factors through the synchronizer row S, budget row B, finite stream-window row W, regular rational handoff row R, and dyadic tolerance row D before the exported completion-seal read E is available to a* Real$^{\uparrow}$ *consumer. The factorization is read inside the same carrier $K = (A, L, S, B, R, W, D, H, C, P, N)$ and is authenticated only by the displayed route, provenance, and local naming rows C, P, N.*

Proof. Project the accepted carrier to K. The NameCert obligations of Definition 900.2 expose S, B, W, R, D as the budget and synchronizer exactness rows for the classifier window.

Use budget exactness from Definition 900.6. It forces the exported read to pass through the synchronizer, finite budget, stream window, regular handoff, and dyadic tolerance ledger before C, P, N can authenticate the consumer-facing read.

Finally apply non-escape from Definition 900.3. Since no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness theorem is present, there is no second path from the window to E. □

Theorem 900.9 (Real completion-seal classifier threshold uniqueness)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* RealCompletionSealClassifierUp *carrier. Any exported completion-seal threshold read over the displayed rows factors through the same synchronizer row S, budget row B, StreamName window row W, RegSeqRat handoff row R, and DyadicRatCore tolerance row D used by the exported classifier window. The route rows C, P, N authenticate that displayed route but cannot introduce a second threshold service for the same seal read.*

Proof. Project the accepted read to the carrier tuple K. The only threshold-facing support rows available before route authentication are the synchronizer, finite budget, StreamName window, RegSeqRat handoff, and DyadicRatCore tolerance rows S, B, W, R, D.

Apply the budget exactness theorem, Definition 900.6. It makes those five rows exact prerequisites for the seal read, so a threshold read exported by the packet has to cross the same budget surface as the classifier window.

Apply the window factorization theorem, Definition 900.8. The exported threshold read is therefore already on the displayed route through S, B, W, R, D before C, P, N can present it to a $\mathsf{Real}^{\uparrow}$ or completion-seal consumer.

Finally use the non-escape boundary of Definition 900.3. Since the carrier contains no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness theorem, the route rows C, P, N have no external source from which to manufacture another threshold service for the same displayed seal read. □

Theorem 900.10 (Real completion-seal classifier tail-agreement stability)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ and $K' = (A', L', S', B', R', W', D', H', C', P', N')$ be accepted* RealCompletionSealClassifierUp *carriers related by the displayed componentwise* hsame *transports. Transport preserves the joint seal-equivalence read through A, L and its transported counterpart through A', L' as presented to consumers by H, C, P, N and H', C', P', N'.*

Proof. Start with componentwise stability, Definition 900.4. The transport rows record exactly the agreement, limit-seal, synchronizer, budget, window, handoff, dyadic, route, provenance, and local naming coordinates.

Apply seal equivalence from Definition 900.5. The classifier reads completion-seal equivalence only through A, L before the $\mathsf{Real}^{\uparrow}$ consumer sees the exported seal classifier.

The transported read is therefore the same joint A, L seal-equivalence read through the transported support rows. Non-escape from Definition 900.3 rules out any quotient-stream or host-equality comparison appearing during transport. □

Theorem 900.11 (Real completion-seal classifier RealUp non-escape)**.** *A $\mathsf{Real}^{\uparrow}$ consumer of an accepted* RealCompletionSealClassifierUp *packet cannot read the completion-seal classifier from quotient streams, host equality, selected external limits, countable choice, or ambient completeness. The consumer receives only the displayed seal rows A, L, synchronizer and budget rows S, B, W, R, D, componentwise transport row H, route row C, provenance row P, and local naming row N.*

Proof. Project the consumer read to the accepted carrier $K = (A, L, S, B, R, W, D, H, C, P, N)$. The obligation theorem, Definition 900.2, lists all rows available to the local NameCert surface.

Use Definition 900.8 for the exported classifier window and Definition 900.10 for the transported seal-equivalence read. Together they place the consumer-facing classifier behind the displayed rows $A, L, S, B, W, R, D, H, C, P, N$.

Finally apply Definition 900.3. The carrier contains no quotient-stream coordinate, host equality, selected external limit, countable-choice row, or ambient completeness theorem, so a $\mathsf{Real}^{\uparrow}$ consumer has no route to such data. □

Theorem 900.12 (Real completion-seal classifier public scope)**.** *Every accepted* RealCompletionSealClassifierUp *packet exposes one public classifier scope over the finite carrier rows*

$$A, L, S, B, R, W, D, H, C, P, N$$

of Definition 900.1. Any $\mathsf{Real}^{\uparrow}$ or completion-seal consumer must factor through the seal-equivalence theorem, budget exactness, tail-radius handoff, window factorization, tail-agreement stability, and the $\mathsf{Real}^{\uparrow}$ non-escape theorem. No quotient stream, host equality, selected external limit, countable choice, or ambient completeness row is exported.

Proof. Project the accepted packet to the carrier tuple of Definition 900.1. This fixes the only available public rows as $A, L, S, B, R, W, D, H, C, P, N$.

Use the NameCert obligations of Definition 900.2. They make seal equivalence through A, L, budget and synchronizer exactness through S, B, W, R, D, and route authentication through C, P, N the whole classifier interface.

Compose Definition 900.7, Definition 900.8, and Definition 900.10. The exported classifier window and the transported agreement read therefore remain on the same finite route through $A, L, S, B, R, W, D, H, C, P, N$.

Finally apply Definition 900.11. It excludes quotient streams, host real equality, selected external limits, countable choice, and ambient completeness from the consumer surface, so the public scope is exactly the displayed finite classifier route. □

Theorem 900.13 (Real completion-seal classifier public export)**.** *Every public* $\mathsf{RealCompletionSealClassifierUp}$ *export to a* $\mathsf{Real}^{\uparrow}$ *consumer or to a completion-seal consumer factors through the finite route*

$$A, L, S, B, W, R, D, H, C, P, N.$$

The exported boundary is exactly the displayed seal-equivalence, budget-exactness, window-factorization, tail-agreement-stability, $\mathsf{Real}^{\uparrow}$ *non-escape, and public-scope surface of the accepted carrier. It exports no quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness principle.*

Proof. Project the accepted public packet to the carrier of Definition 900.1. This fixes the public export route as $A, L, S, B, W, R, D, H, C, P, N$, with the same displayed route, provenance, and local naming surface.

Use seal equivalence from Definition 900.5 and budget exactness from Definition 900.6. These two rows force any exported classifier to read the agreement seal, completion-limit seal, synchronizer, finite budget, StreamName window, regular handoff, and dyadic tolerance ledger before the consumer sees the classifier.

Then use the tail-radius handoff of Definition 900.7. A budget row that opens a tail-radius request must pass through the shared $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ coverage and $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ handoff route before the exported classifier reaches the consumer.

Next apply the window factorization of Definition 900.8 and the tail-agreement stability of Definition 900.10. They keep the exported window and transported agreement read on the same finite route through $A, L, S, B, W, R, D, H, C, P, N$.

Finally invoke $\mathsf{Real}^{\uparrow}$ non-escape from Definition 900.11 and public scope from Definition 900.12. Since quotient streams, host real equality, selected external limits, countable choice, and ambient completeness are absent from the carrier and from the public scope, no downstream export can recover them. □

Theorem 900.14 (Real completion-seal classifier completeness boundary)**.** *Every public* $\mathsf{RealCompletionSealClassifi}$ *export to a* $\mathsf{Real}^{\uparrow}$ *or completion-seal consumer is exactly the finite route*

$$A, L, S, B, W, R, D, H, C, P, N.$$

Consequently the export supplies a classifier boundary for a displayed seal read, and cannot be used as a theorem of $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient-stream equality, countable choice, or ambient Cauchy convergence.*

Proof. Start with the public export theorem, Definition 900.13. It states that every downstream $\mathsf{Real}^{\uparrow}$ or completion-seal export factors through the displayed finite route $A, L, S, B, W, R, D, H, C, P, N$.

Apply the $\mathsf{Real}^{\uparrow}$ non-escape theorem, Definition 900.11. The exported consumer surface has no coordinate for quotient streams, host equality, selected external limits, countable choice, or ambient completeness.

Use budget exactness from Definition 900.6. The synchronizer, finite budget, StreamName window, regular handoff, and dyadic tolerance ledger remain part of the same displayed route, so the read is a bounded classifier read rather than an ambient convergence principle.

Finally use public scope, Definition 900.12. That theorem exhausts the public consumer interface by the finite rows already displayed, so no consumer can reinterpret the exported classifier boundary as a $\mathsf{Real}^{\uparrow}$ completeness theorem, a selected-limit theorem, quotient-stream equality, countable choice, or ambient Cauchy convergence. □

Falsifiable prediction: If a Real completion-seal classifier read bypasses both a displayed seal row and a displayed budget/synchronizer row, the classifier packet is refuted.

Independence witness: realtailagreementseal, cauchylimitseal, cauchysealbudgetsynchronizer, real, regseqrat: no sibling is bijective because the classifier packet has a joint completion-seal classifier row tying agreement, seal, synchronizer, and Real handoff rows; each sibling lacks at least one of these coordinates.

Theorem 900.15 (Real completion-seal classifier public bridge schema)**.** *Let $K = (A, L, S, B, R, W, D, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionSealClassifierUp}$ *carrier. The carrier rows $A, L, S, B, R, W, D, H, C, P, N$, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSealClassifierUp}}$ *obligations, non-escape, componentwise stability, seal-equivalence, budget-exactness, window factorization, and threshold uniqueness assemble the public bridge schema from* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *to the exported* $\mathsf{RealCompletionSealClassifierUp}$ *read. The exported bridge surface has no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness coordinate.*

Proof. Project the accepted packet to the carrier of Definition 900.1. This exposes exactly the rows $A, L, S, B, R, W, D, H, C, P, N$: tail agreement, Cauchy-limit seal, seal-budget synchronizer, finite budget, regular rational handoff, StreamName window, DyadicRatCore tolerance ledger, componentwise transport, displayed route, provenance, and local naming.

Use the local theorem chain. The obligation theorem Definition 900.2 supplies the $\mathsf{NameCert}$ rows; non-escape and stability are Definition 900.3 and Definition 900.4; seal equivalence, budget exactness, window factorization, and threshold uniqueness are Definition 900.5, Definition 900.6, Definition 900.8, and Definition 900.9.

The sibling sources are then read row by row. The rows A and L provide the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ seal inputs, S and B provide the $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ budget input, R, W, D provide the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ finite-window route, and H, C, P, N authenticate the route to the $\mathsf{Real}^{\uparrow}$ consumer.

Finally apply the public export and completeness-boundary theorems, Definition 900.13 and Definition 900.14. They show that the exported bridge is the same finite classifier read and that no quotient stream, host equality, selected external limit, countable-choice row, or ambient completeness coordinate is present in the exported surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionSealClassifierUp}}$**.**

Constructive story: A RealCompletionSealClassifierUp packet is generated from tail-agreement seal rows, Cauchy-limit seal rows, Cauchy seal-budget synchronizer rows, finite budget rows, regular rational handoff rows, StreamName windows, DyadicRatCore ledgers, componentwise transports, displayed routes, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RealCompletionSealClassifierUp}}$, maturePackageClosed)

Scope closed: Definition 900.1, Definition 900.2, Definition 900.3, Definition 900.4, Definition 900.5, Definition 900.6, Definition 900.7, Definition 900.8, Definition 900.9, Definition 900.10, Definition 900.11, Definition 900.12, Definition 900.13, Definition 900.14, and Definition 900.15 bind the carrier, NameCert obligation surface, non-escape boundary, componentwise stability, seal-equivalence read, budget exactness, tail-radius handoff, window factorization, threshold uniqueness, tail-agreement stability, RealUp non-escape, public classifier scope, public consumer export, completeness boundary, and bridge surface for the finite completion-seal classifier packet.

Formal verification: FormalStatus(NameCert$_{\text{RealCompletionSealClassifierUp}}$, axiomClean)

Lean target: `BEDC.Derived.RealCompletionSealClassifierUp.RealCompletionSealClassifierTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close quotient streams, host real equality, selected external limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: The paper theory axis is mature for the finite completion-seal classifier surface; Lean verification and bridge checking remain independent obligations.

Chapter 901

A Concrete Naming Certificate for $\mathsf{AuditGateFailureWitness}^{\uparrow}$

The $\mathsf{AuditGateFailureWitness}^{\uparrow}$ packet realizes the failed-verdict side of the audit-gate boundary described in chapter 9396. It sits beside the pass/refusal ledgers of chapter 526, the composed gate surface of chapter 696, the apophatic refusal packet of chapter 630, and the gate-question packet of chapter 715. Those siblings expose questions, boundary ledgers, composition routes, or negative certificates; this packet records the concrete diagnostic witness for a gate row whose boundary item fails.

Falsifiable prediction: If an AuditGateFailureWitnessUp packet is accepted, every public consumer route reaches the diagnostic row before any composed refusal handoff; a direct route from gate row to theorem content refutes the packet.

Independence witness: AuditGateBoundaryUp supplies pass/refusal ledgers but not a diagnostic witness; AuditGateCompositionUp composes gate rows but does not carry a primitive failed item; NegativeCertificateBoundaryUp records apophatic refusal but not a gate diagnostic; ApophaticGateQuestionUp asks the gate question but does not store a failed verdict.

Definition 901.1 (Audit-gate failure-witness carrier)**.** *An* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (G, F, R, D, H, C, P, N)$$

with the following displayed rows:

1. *G is the audit-gate row whose finite boundary check is being read;*
2. *F is the failed boundary item, recording the source token, proof-ancestry dependency, unresolved target, origin/import crossing, or other gate-visible item that did not pass;*
3. *R is the refusal target attached to F, namely the theorem-content route, host supply, marker resolution, or internalizer that the gate refuses to admit;*
4. *D is the diagnostic readback row, recording the exact finite reason by which G classifies F as a failed item and attaches the refusal to R;*
5. *H records componentwise* hsame *transport for the gate row, failed item, refusal target, diagnostic readback, provenance, and local naming rows;*
6. *C records the displayed* Cont *replay routes from G to F, from F to D, from D to R, and from the local witness to any composed refusal handoff;*

7. *P records* Pkg *provenance naming* AuditGateBoundary$^{\uparrow}$, AuditGateComposition$^{\uparrow}$, NegativeCertificateBoundary$^{\uparrow}$, ApophaticGateQuestion$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *source packets;*

8. *N is the local* NameCert$_{\text{AuditGateFailureWitness}^{\uparrow}}$ *row.*

The carrier exports a failure witness for the displayed gate row. It exports no route by which the failed item becomes theorem content and no replacement diagnostic outside the displayed readback row.

Theorem 901.2 (Audit-gate failure-witness NameCert obligations)**.** *For every accepted* AuditGateFailureWitness$^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

the local NameCert$_{\text{AuditGateFailureWitness}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission: the accepted object is exactly the displayed* BHist *packet W with gate row, failed boundary item, refusal target, diagnostic readback, transport, continuation, provenance, and local naming rows;*

2. *diagnostic exactness: the row D is the only diagnostic readback for F under G, and it records the finite gate-facing reason for failure before any consumer reaches the refusal target;*

3. *refusal attachment: the row R is attached to the pair (F, D) rather than to an anonymous gate outcome, so the refusal cannot be detached from the concrete failed boundary item;*

4. *replay visibility: every public replay of the witness factors through the displayed routes in C, including the route from G to F, the diagnostic route from F to D, and the refusal route from D to R;*

5. *non-internalization: the packet exposes the failed verdict and its diagnostic, but it exports no theorem-content object, host supply, marker repair, admitted internalizer, or gate-bypass route for F.*

Lean4 ✓ `BEDC.Derived.AuditGateFailureWitnessUp.AuditGateFailureWitnessUp obligation surface`

Proof. Carrier admission is projection from Definition 901.1. The tuple exposes exactly the rows G, F, R, D, H, C, P, N, so an accepted witness has no hidden item beyond the displayed finite packet.

Diagnostic exactness is read from the role of D. The diagnostic is attached to F under the gate row G, and the continuation route reaches D before it can reach the refusal target. Thus a consumer cannot read the failure as an anonymous refusal.

Refusal attachment then follows by reading R after D. The refused target is paired with the failed item and diagnostic row, so the refusal cannot be moved to a different boundary item without leaving the carrier.

Replay visibility is confined to C, while componentwise transport is confined to H. The package and local-name rows P and N bind these visible routes as a local NameCert surface.

Finally, non-internalization follows from the carrier inventory. No coordinate of W supplies theorem content for F, repairs the marker or host supply, admits the refused internalizer, or bypasses the gate diagnostic. □

Theorem 901.3 (Audit-gate failure-witness target readback totality)**.** *For every accepted* AuditGateFailureWitness$^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

every public consumer read of the refusal target factors through the diagnostic row D before it reaches R. A read that reaches theorem content, marker repair, or host supply directly from F is not a read of the displayed carrier.

Proof. Project the carrier in Definition 901.1. Its continuation row C displays the route from the gate row to the failed item, from the failed item to the diagnostic, and from the diagnostic to the refusal target.

Apply the diagnostic exactness and replay visibility clauses of Definition 901.2. They force the diagnostic row D to be read before the target row R is available to any public consumer.

The non-internalization clause then excludes direct recovery of theorem content, marker repair, or host supply from F. Such a recovery would bypass D and therefore leave the displayed BHist inventory of the witness. □

Theorem 901.4 (Audit-gate failure-witness diagnostic refusal order)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

the public refusal order is G, F, D, R before the support rows H, C, P, N are consumed. In particular, all consumer reads pass through D before R, and the refusal target cannot be detached from the diagnostic readback that names the failed boundary item.

Proof. Start with the displayed continuation route. The carrier gives no route from G to R and no route from F to R except through the diagnostic row D.

Use refusal attachment from Definition 901.2. The row R is attached to the concrete pair (F, D), so a consumer cannot read the refusal as an anonymous outcome of the gate row alone.

Finally inspect the support rows. Transport, continuation replay, provenance, and local naming preserve the same order rather than creating a second refusal path. Thus the diagnostic readback precedes every public refusal handoff. □

Theorem 901.5 (Audit-gate failure-witness boundary-item noninternalization coverage)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

no theorem-content row, marker-repair row, or host-supply row can be recovered from the failed boundary item F without leaving the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *inventory. The packet covers the failure only as a gate row, failed item, diagnostic readback, refusal target, transport, replay, provenance, and local name.*

Proof. Read the carrier coordinates. The failed boundary item F appears with G, D, R, H, C, P, N and with no additional coordinate for theorem content, marker repair, host supply, or admitted internalization.

Apply the NameCert obligations. Carrier admission fixes the displayed BHist packet, replay visibility fixes the Cont routes through D, and non-internalization excludes every recovery of theorem content or repair content from F.

The remaining support is exactly componentwise hsame transport, package provenance, and local $\mathsf{NameCert}$ naming. Recovering a theorem-content, marker-repair, or host-supply row would require a row outside that inventory, so it is not covered by the accepted witness. □

Theorem 901.6 (Audit-gate failure-witness composition handoff)**.** *Let*

$$W_0 = (G_0, F_0, R_0, D_0, H_0, C_0, P_0, N_0)$$

and

$$W_1 = (G_1, F_1, R_1, D_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carriers consumed by an* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet. The composition may read the two diagnostic rows* D_0, D_1*, the two refusal targets* R_0, R_1*, and the two replay routes* C_0, C_1 *as input to its composed refusal trace. It may not turn either failed boundary item* F_0 *or* F_1 *into theorem content, and it may not replace either diagnostic row by the composed trace itself.*

Proof. Project the two accepted witnesses. Each exposes a gate row, failed item, refusal target, diagnostic readback, transport, replay, provenance, and local name.

Feed only the visible diagnostic and refusal rows to the composition packet. The first witness contributes D_0, R_0, C_0, and the second witness contributes D_1, R_1, C_1. The composed surface may order these rows, compare their gate boundaries, and expose a composed refusal trace.

The failed items remain boundary items of their original witnesses. Since neither carrier has a theorem-content coordinate for its failed item, the composition has no source from which to manufacture such content.

The composed trace also cannot replace the diagnostic rows. It is a consumer of D_0 and D_1, not a primitive diagnostic for either gate row. Thus the handoff preserves both failure witnesses while keeping the failures outside theorem content. □

Theorem 901.7 (Audit-gate failure-witness root public handoff)**.** *Every root-facing public handoff of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier keeps the diagnostic row before refusal-target reads. The handoff surface is exactly* G, F, R, D, H, C, P, N*, and it contains no theorem-content recovery from the failed boundary item.*

Proof. Project the carrier $W = (G, F, R, D, H, C, P, N)$. The gate row, failed item, refusal target, diagnostic readback, transport, replay, provenance, and local name are the only displayed coordinates.

Apply the NameCert obligations and the composition handoff. They require public refusal reads to pass through the diagnostic row and allow composition to consume only visible diagnostic, refusal, and replay rows.

The TasteGate marker context names the same finite carrier for later formal alignment. Since F is never a theorem-content coordinate, the root handoff exports a diagnostic boundary and nothing recovered from the failed item. □

Theorem 901.8 (Audit-gate failure-witness composition readback scope)**.** *In every* $\mathsf{AuditGateComposition}^{\uparrow}$ *consumption of* $\mathsf{AuditGateFailureWitness}^{\uparrow}$*, the readable scope of each component witness is* G, F, D, R, H, C, P, N *with* D *before the refusal target* R*. The composition may compare diagnostics and refusal targets but may not replace either diagnostic row by the composed trace.*

Proof. Use Definition 901.6. It gives the composition access to D_0, R_0, C_0 and D_1, R_1, C_1 while leaving the failed items as boundary items of their own witnesses.

Next apply refusal-route exactness and refusal attachment stability. They keep R attached to the concrete pair (F, D) under the displayed hsame and Cont rows.

Thus the composition readback scope is diagnostic-first on G, F, D, R, H, C, P, N. The composed trace is downstream of those rows and cannot become a primitive diagnostic for either witness. □

Theorem 901.9 (Audit-gate failure-witness diagnostic target exhaustion)**.** *After an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *consumer has read* G, F, D, R, H, C, P, N*, no further diagnostic target remains inside the packet. In particular, the packet exposes no internalizer, repair row, host-supply row, or theorem-content row for the failed item.*

Proof. Start from the root public handoff. It identifies the complete public surface as the displayed diagnostic witness rows and places D before refusal-target consumption.

Then use composition readback scope. Even when the witness is consumed by a composed audit gate, the available rows remain G, F, D, R, H, C, P, N and the composed trace does not replace the component diagnostic.

Finally apply boundary-item noninternalization coverage. Rows for internalization, repair, host supply, and theorem content are absent from the carrier, so no diagnostic target remains after the displayed rows are exhausted. □

Theorem 901.10 (Audit-gate failure-witness refusal attachment stability)**.** *Let*

$$W = (G, F, R, D, H, C, P, N)$$

be an accepted $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier. Componentwise* hsame *transport through* H *preserves the attachment of the refusal target* R *to the concrete pair* (F, D)*. A transported witness may rename the displayed rows only componentwise; it cannot detach* R *from the failed boundary item* F *or from the diagnostic readback row* D*.*

Proof. Project the carrier from Definition 901.1. The displayed packet contains the failed item, diagnostic row, refusal target, transport row, replay row, provenance row, and local name as separate finite coordinates.

Use the NameCert obligations of Definition 901.2. Their refusal attachment clause binds R to the pair (F, D), while their replay visibility clause keeps public reads on the displayed continuation rows.

The composition handoff, Definition 901.6, only consumes the diagnostic and refusal rows after the witness has supplied them. Since H acts componentwise on the same carrier coordinates, any accepted transport preserves the attachment rather than creating a detached refusal target. □

Theorem 901.11 (Audit-gate failure-witness diagnostic nonescape)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

no public route can bypass the diagnostic row D *to turn the failed boundary item* F *into theorem content, a repaired marker, an admitted internalizer, or a host-supplied replacement for the failed gate item.*

Proof. Project W and read the carrier inventory of Definition 901.1. The only route out of the gate row passes through the displayed failed item, diagnostic row, refusal target, transports, replay rows, provenance, and local name.

The NameCert obligations, Definition 901.2, place the diagnostic row before any consumer reaches the refusal target and explicitly exclude theorem content, marker repair, host supply, admitted internalizers, and gate-bypass routes for F.

The composition handoff theorem shows that even a composed audit surface receives D as diagnostic input rather than replacing it. Thus every public consumer must read the diagnostic before the refusal, and no route can turn F into admitted theorem content or a repaired marker. □

Theorem 901.12 (Audit-gate failure-witness root public exhaustion)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

the root public surface is exhausted by the displayed BHist *packet, the componentwise* hsame *transport row* H*, the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N*. After carrier admission, diagnostic exactness, refusal attachment, replay visibility, and non-internalization are read, no root consumer has a remaining route to theorem content, marker repair, host supply, an admitted internalizer, a detached refusal target, or a composed refusal replacement.*

Proof. Project the carrier and the NameCert obligations. Carrier admission identifies the public packet as $W = (G, F, R, D, H, C, P, N)$, while diagnostic exactness and refusal attachment place D before R and attach the refusal to the concrete failed item.

Use diagnostic nonescape. The failed boundary item F cannot become theorem content, marker repair, an admitted internalizer, or a host-supplied replacement after the diagnostic row has been displayed.

Finally apply refusal attachment stability and composition readback scope. Componentwise hsame transport and displayed Cont replay preserve the same route, and an $\mathsf{AuditGateComposition}^{\uparrow}$ consumer reads only the diagnostic, refusal, and replay rows. Thus the root public surface is exhausted by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 901.13 (Audit-gate failure-witness failure diagnostic exhaustion)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet*

$$W = (G, F, R, D, H, C, P, N),$$

the diagnostic row D *exhausts the failure explanation for the gate row* G *and failed boundary item* F *inside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *inventory. No second diagnostic, hidden proof object, host primitive, marker repair, replacement* NameCert*, or detached closure row remains after* D, H, C, P, N *are read.*

Proof. Read the diagnostic exactness clause of the NameCert obligations. It states that D is the only diagnostic readback for F under G, and the replay row reaches it before the refusal target is consumed.

Apply diagnostic nonescape and boundary-item noninternalization coverage. These results exclude theorem content, marker repair, admitted internalizers, host supply, and replacement diagnostic content from the accepted carrier.

The remaining support rows are only transport, replay, provenance, and local naming. Since hsame, Cont, Pkg, and NameCert preserve the displayed diagnostic rather than creating a second one, the failure explanation is exhausted by D for the same packet $W = (G, F, R, D, H, C, P, N)$. □

Theorem 901.14 (Audit-gate failure-witness refusal route coverage)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

every public refusal route factors through the displayed BHist *rows* G, F, D, R*, the componentwise* hsame *transport* H*, the displayed* Cont *replay* C*, the* Pkg *provenance* P*, and the local* NameCert *row* N*. A route from* G *or* F *directly to theorem content, host supply, marker repair, admitted internalization, or a detached composed refusal object is not covered by the packet.*

Proof. Begin with refusal-route exactness and refusal attachment stability. They keep R attached to the pair (F, D) under the displayed transport and replay rows, so a public route cannot detach the refusal from the failed boundary item.

Next use the NameCert obligations and diagnostic nonescape. Carrier admission fixes the finite packet, replay visibility forces the public route through C, and non-internalization blocks theorem content, marker repair, host supply, and admitted internalizers.

Finally read composition readback scope. A composed audit surface may consume the displayed refusal route, but it cannot replace D or R by a detached composed object. Hence the covered route is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert inventory of W. □

Theorem 901.15 (Audit-gate failure-witness replay factorization)**.** *Every public replay of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N)$$

factors first through the displayed diagnostic path

$$G \to F \to D \to R$$

and then through the visible support rows H, C, P, N. No replay may skip the failed item, replace the diagnostic, or read a refusal target outside the displayed transport, continuation, provenance, and local naming surface.

Proof. Project the accepted carrier. Its continuation row C displays the route from G to F, from F to D, from D to R, and from the local witness to any composed refusal handoff.

Apply the NameCert obligations of Definition 901.2. Carrier admission and replay visibility identify the public replay spine, while diagnostic exactness and refusal attachment force the order $G \to F \to D \to R$.

Finally apply the composition handoff, Definition 901.6. It permits a composed gate surface to consume only the displayed diagnostic, refusal, and replay rows. The remaining support is therefore exactly H, C, P, N, and no undisplayed repair, bypass, or replacement diagnostic participates in replay. □

Theorem 901.16 (Audit-gate failure-witness boundary coverage)**.** *Every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N)$$

covers exactly the gate row, failed item, diagnostic row, refusal target, transport rows, replay rows, provenance row, and local NameCert *row displayed by its carrier. No accepted witness adds a separate repair row, host-supplied theorem row, omitted diagnostic, or off-packet internalizer.*

Proof. Project the carrier of Definition 901.1. Its visible rows are G, F, R, D, H, C, P, N, so the gate row, failed item, refusal target, diagnostic, transport, replay, provenance, and local naming rows are the full inventory.

Apply the NameCert obligations of Definition 901.2. Carrier admission, diagnostic exactness, refusal attachment, replay visibility, and non-internalization account for every accepted row in that inventory.

The composition handoff and replay factorization theorems preserve the same rows for downstream consumers. They consume diagnostic, refusal, replay, transport, provenance, and naming data from the witness, but they do not add a repair row, theorem row, replacement diagnostic, or internalizer outside W. □

Theorem 901.17 (Audit-gate failure-witness origin refusal)**.** *If the failed item F records an origin or import-crossing failure, then every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *consumer reads that failure as refusal data through D and R. It cannot be converted into baseline theorem content, repaired marker content, or an admitted internalizer by replaying H, C, P, N.*

Proof. Start from the diagnostic row. The carrier makes D the displayed readback explaining why F failed under the gate row G, and the NameCert obligations force consumers to reach D before the refusal target R.

Use diagnostic nonescape and replay factorization. Diagnostic nonescape blocks replacement of the gate-visible failure by theorem content, while replay factorization keeps public replay on the path $G \to F \to D \to R$ with support rows H, C, P, N.

Finally apply the composition handoff. A composed audit surface may pass along the diagnostic and refusal rows, but it remains a consumer of the failure witness. Hence an origin or import-crossing failure stays refusal data and cannot become baseline theorem content, repaired marker content, or an admitted internalizer. □

Theorem 901.18 (Audit-gate failure-witness ledger exhaustion)**.** *Every public consumer of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet factors through*

$$G, F, D, R, H, C, P, N.$$

The packet exports no marker repair, host supply, theorem object, admitted internalizer, or route that bypasses the displayed gate, failed item, diagnostic, refusal, transport, replay, provenance, and local naming rows.

Proof. First read the public replay spine from Definition 901.15. It forces every public replay through the diagnostic route $G \to F \to D \to R$ and then through the support rows H, C, P, N.

Next apply boundary coverage. The accepted witness contains no public coordinate beyond the gate row, failed item, diagnostic, refusal target, transport, replay, provenance, and local naming rows.

The NameCert obligations and composition handoff then exclude all escape routes. Non-internalization blocks marker repair, host supply, theorem content, and admitted internalizers, while the composed surface may only consume the displayed diagnostic, refusal, and replay rows. Therefore the listed ledger exhausts every public consumer. □

Theorem 901.19 (Audit-gate failure-witness diagnostic totality)**.** *Every public read of a failed gate item in an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet factors through exactly the displayed diagnostic row D before reaching the refusal target R. The route is*

$$G \to F \to D \to R$$

with support rows H, C, P, N, so no public consumer can read F as a failure witness by bypassing D or by reaching R through a detached refusal route.

Proof. Start with the carrier and target-readback totality. The carrier displays $W = (G, F, R, D, H, C, P, N)$, and the failed item F is public only as an item governed by the gate row G.

Apply diagnostic-refusal order. It places the diagnostic row D between the failed item and the refusal target, so the route to R is not available until D has been consumed.

Finally use diagnostic nonescape and replay factorization. Diagnostic nonescape blocks replacement of D by marker repair or theorem content, while replay factorization carries the same route through H, C, P, N. Hence every public failed-item read reaches R through exactly the displayed diagnostic row. □

Theorem 901.20 (Audit-gate failure-witness refusal consumer boundary)**.** *Any consumer of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *witness must preserve the gate row G, failed item F, diagnostic row D, refusal target R, transport row H, replay row C, provenance row P, and local naming row N. Omitting D or detaching R from the pair (F, D) exits the accepted packet and cannot be repaired by transport, replay, provenance, or local naming alone.*

Proof. Project the ledger exhaustion theorem. Every public consumer already factors through G, F, D, R, H, C, P, N and receives no marker repair, host supply, theorem object, admitted internalizer, or bypass route.

Use diagnostic totality to bind D and R. The refusal target is reached through the displayed diagnostic row for the failed item, so a consumer that omits D or detaches R from (F, D) changes the public witness.

The support rows are conservative. Componentwise transport, replay, provenance, and local naming preserve the displayed diagnostic-refusal packet; they do not synthesize a replacement diagnostic or a detached refusal route. Thus such a consumer leaves the accepted boundary. □

Theorem 901.21 (Audit-gate failure-witness diagnostic uniqueness)**.** *For every accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N),$$

the row D is the unique public diagnostic for the pair (G, F) along the displayed continuation route. Any alternate diagnostic for the same gate row and failed boundary item is outside the carrier and cannot reach the refusal target R as an accepted $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *consumer route.*

Proof. Start with the carrier definition and the NameCert obligations. The accepted packet displays exactly one diagnostic readback row D for the gate row G and failed item F, and diagnostic exactness places that row before any refusal handoff.

Use target readback totality and diagnostic-refusal order. Every public route to R must pass through the route

$$G \to F \to D \to R.$$

Finally apply the refusal consumer boundary. An alternate diagnostic would either replace D or detach R from (F, D). Both moves leave the displayed packet, so no alternate diagnostic is an accepted public diagnostic for the same pair (G, F). □

Theorem 901.22 (Audit-gate failure-witness boundary-item coverage)**.** *Every failed boundary item accepted by an* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier is covered exactly as gate row, failed item, diagnostic, refusal, transport, replay, provenance, and local name. The carrier has no theorem-content coordinate and no marker-repair coordinate for the failed item.*

Proof. Project the carrier $W = (G, F, R, D, H, C, P, N)$. Its rows name the gate, failed boundary item, refusal target, diagnostic readback, transport, replay, provenance, and local NameCert data.

Apply boundary-item noninternalization coverage. That theorem excludes theorem-content, marker-repair, and host-supply rows from the accepted inventory.

Use diagnostic totality and the refusal consumer boundary to check the public route. The failed item is consumed through G, F, D, R and preserved by H, C, P, N; any consumer that adds theorem content or marker repair has changed the accepted packet rather than covering the displayed boundary item. □

Theorem 901.23 (Audit-gate failure-witness refusal-route exactness)**.** *Every consumer route to the refusal target R of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier factors through G, F, D and the displayed support rows H, C, P, N. There is no accepted detached route to R that omits the gate row, failed item, diagnostic readback, transport, replay, provenance, or local naming rows.*

Proof. Begin with target readback totality. It forces public reads of the refusal target to pass through the diagnostic row after the gate row and failed boundary item.

Diagnostic-refusal order gives the exact spine G, F, D, R. Replay factorization then carries that spine through the support rows H, C, P, N.

Finally use the refusal consumer boundary. Omitting the diagnostic or detaching R from (F, D) exits the carrier, while transport, replay, provenance, and local naming preserve only the displayed support route. Hence every accepted route to R has the stated factorization. □

Theorem 901.24 (Audit-gate failure-witness diagnostic carrier totality)**.** *For every accepted* AuditGateFailureWitne *packet*

$$W = (G, F, R, D, H, C, P, N),$$

the carrier exposes exactly one diagnostic carrier row before the refusal target. The row is D, it is attached to the gate row G and failed item F, and every accepted route to R first reads that displayed diagnostic row.

Proof. Start from carrier admission in Definition 901.2. The accepted packet contains one diagnostic coordinate, namely D, and no second diagnostic row is present in W.

Use diagnostic exactness and target readback totality. These obligations place the diagnostic readback before the refusal target and force every public read of R through the route $G \to F \to D \to R$.

Finally apply diagnostic uniqueness and refusal-route exactness. An alternate diagnostic would replace D or detach R from the displayed pair (F, D), and both moves leave the accepted carrier. Thus the diagnostic carrier row before R is exactly D. □

Theorem 901.25 (Audit-gate failure-witness composed refusal route preservation)**.** *Let*

$$W = (G, F, R, D, H, C, P, N)$$

be an accepted $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier consumed by an* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet. The composed consumer preserves the displayed route G, F, D, R, H, C, P, N and cannot replace it with composed refusal content. In particular, the composed refusal trace may consume D, R, and C, but it does not become the diagnostic row for G, F and does not supply theorem content for F.*

Proof. Read the composition handoff. It permits an $\mathsf{AuditGateComposition}^{\uparrow}$ surface to consume the diagnostic, refusal, and replay rows of an accepted failure witness, while the failed item remains a boundary item of that witness.

Use replay factorization and refusal-route exactness to pin down the consumed route. The public route remains $G \to F \to D \to R$ with support rows H, C, P, N, so the composed trace is downstream from the displayed carrier rather than a replacement for it.

Finally apply diagnostic nonescape and the refusal consumer boundary. They block conversion of F into theorem content and block replacement of D by a detached refusal route. Hence the composed consumer preserves the displayed failure-witness route. □

Theorem 901.26 (Audit-gate failure-witness gate-row exhaustion)**.** *Every gate-row consumer of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet is exhausted by carrier admission, diagnostic exactness, refusal attachment, replay visibility, and non-internalization. After these five rows are read, the consumer has no remaining route to a marker repair, host supply, theorem object, admitted internalizer, omitted diagnostic, detached refusal target, or composed refusal replacement.*

Proof. Begin with the NameCert obligation surface. Its five rows expose the carrier, identify the diagnostic, attach the refusal to the displayed failed item, make replay visible, and exclude internalization of the failure.

Apply ledger exhaustion and boundary-item coverage. They show that every public consumer of the gate row factors through G, F, D, R, H, C, P, N and that the failed item is covered only by those displayed rows.

Finally use composed refusal route preservation. Even when an $\mathsf{AuditGateComposition}^{\uparrow}$ packet consumes the witness, the route remains the displayed failure-witness route rather than composed refusal content. Therefore the five NameCert rows exhaust every gate-row consumer. □

Theorem 901.27 (Audit-gate failure-witness diagnostic row inversion)**.** *Let*

$$W = (G, F, R, D, H, C, P, N)$$

be an accepted $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier. Any public refusal read of* W *determines the displayed diagnostic row* D *for the gate row* G *and failed boundary item* F*. The read cannot determine an alternate diagnostic, a detached refusal target, theorem content for* F*, marker repair, host supply, or an admitted internalizer.*

Proof. Start with target readback totality and diagnostic refusal order. They place every public read of the refusal target on the route $G \to F \to D \to R$ before any support row is consumed.

Next use diagnostic carrier totality. The carrier exposes exactly one diagnostic row before R, and that row is attached to the pair (G, F).

Finally apply diagnostic uniqueness and gate-row exhaustion. A second diagnostic, detached refusal route, theorem-content row, marker repair, host supply, or internalizer would add a coordinate absent from $W = (G, F, R, D, H, C, P, N)$. Therefore the public refusal read determines precisely D. □

Theorem 901.28 (Audit-gate failure-witness composition input exhaustion)**.** *Let an accepted* $\mathsf{AuditGateComposition}^{\uparrow}$ *packet consume an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N).$$

Its failure-witness input consists only of the diagnostic row D*, the refusal target* R*, replay through* C*, and the displayed support rows* G, F, H, P, N*. It receives no theorem-content row, host-supply row, marker-repair row, internalizer row, replacement diagnostic, or composed refusal object promoted into the carrier.*

Proof. Read the composition handoff through composed refusal route preservation. The composed consumer is downstream from the route G, F, D, R, H, C, P, N, so it may inspect diagnostic, refusal, replay, transport, provenance, and local naming rows only as displayed rows of the same packet.

Use refusal-route exactness to keep the input route on G, F, D, R. The route to R factors through D and cannot be replaced by a detached composed refusal path.

Finally apply boundary-item coverage and gate-row exhaustion. These theorems exclude theorem content, host supply, marker repair, admitted internalizers, replacement diagnostics, and

composed refusal replacement from the accepted input surface. Hence the composition input is exhausted by the diagnostic, refusal, replay, and displayed support rows. □

Theorem 901.29 (Audit-gate failure-witness origin import boundary)**.** *If the failed item F of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *carrier*

$$W = (G, F, R, D, H, C, P, N)$$

records an origin/import crossing failure, then that failure remains a failed boundary item of W. It cannot be internalized as a host supply, theorem-content object, marker repair, admitted internalizer, or gate-bypass route.

Proof. Project the carrier and the non-internalization clause of the NameCert obligations. The failed item row F is part of the displayed witness, while no row of W supplies theorem content, host supply, marker repair, or an admitted internalizer.

Next use boundary-item coverage. It says every failed boundary item accepted by the carrier is covered only as gate row, failed item, diagnostic, refusal, transport, replay, provenance, and local name.

Finally combine refusal-route exactness with gate-row exhaustion. The origin/import crossing failure must be read through G, F, D, R, H, C, P, N, and after those rows are read there is no remaining route for host supply, theorem content, marker repair, internalization, or gate bypass. Thus the failed crossing remains a failed boundary item. □

Theorem 901.30 (Audit-gate failure-witness diagnostic scope closure)**.** *The diagnostic exactness of an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet is scoped only by the displayed rows*

$$G, F, R, D, H, C, P, N$$

over BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the sibling surfaces* $\mathsf{AuditGateBoundary}^{\uparrow}$*,* $\mathsf{AuditGateComposition}^{\uparrow}$*,* $\mathsf{NegativeCertificateBoundary}^{\uparrow}$*, and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$*. No consumer may add a second diagnostic source, a host-supplied repair row, or a theorem-content row for the failed item while remaining inside the accepted packet.*

Proof. Project the carrier of Definition 901.1. Its visible coordinates are exactly the gate row, failed item, refusal target, diagnostic row, transport, replay, provenance, and local name.

Apply the NameCert obligations and diagnostic exactness. They place the diagnostic row D inside the displayed packet and route it through BHist, hsame, Cont, Pkg, and NameCert together with the named sibling surfaces.

Finally use diagnostic nonescape and origin import boundary. These exclude a second diagnostic source, host repair, theorem content, and internalization from the carrier, so the diagnostic scope is the displayed packet alone. □

Theorem 901.31 (Audit-gate failure-witness refusal handoff scope)**.** *A composed refusal handoff consuming an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet may use the diagnostic row D, refusal target R, replay row C, and support rows G, F, H, P, N. It cannot promote the failed boundary item F to theorem content, cannot replace D by a composed diagnostic, and cannot detach R from the displayed failed item.*

Proof. Read the composition handoff theorem through the carrier rows. The composed consumer is allowed to inspect the diagnostic, refusal, and replay coordinates of the same G, F, R, D, H, C, P, N packet.

Apply composed refusal route preservation and refusal-route exactness. The route to R remains attached to G, F, D and then travels through the displayed support rows.

Finally use boundary-item coverage and diagnostic nonescape. They rule out theorem content for F, replacement of D, and detached refusal content, so the handoff remains a scoped consumer of the original failure-witness packet. □

Theorem 901.32 (Audit-gate failure-witness replay ledger scope)**.** *Replay, componentwise transport, provenance, and local naming in an accepted* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *packet preserve the same diagnostic packet* G, F, R, D, H, C, P, N*. They introduce no hidden repair row, no host-supply row, no theorem-content row, and no replacement diagnostic for the failed boundary item.*

Proof. Project the replay and support coordinates H, C, P, N from Definition 901.1. They preserve and name the rows already present in the packet.

Use replay factorization and gate-row exhaustion. Every public consumer route remains exhausted by carrier admission, diagnostic exactness, refusal attachment, replay visibility, and non-internalization.

Finally apply diagnostic row inversion. A public refusal read determines the displayed diagnostic row D and no alternate diagnostic, repair, host supply, theorem content, or internalizer. Thus replay and naming preserve the same diagnostic packet without adding hidden support. □

Theorem 901.33 (Audit-gate failure-witness public diagnostic export)**.** *For every accepted* AuditGateFailureWitnes *packet*

$$W = (G, F, R, D, H, C, P, N),$$

the public diagnostic route exported by the packet is exactly

$$G \longrightarrow F \longrightarrow D \longrightarrow R$$

followed by the support rows H, C, P, N*. A public consumer may read the gate row, the failed item, the diagnostic, the refusal target, componentwise transport, displayed replay, provenance, and local naming, but it receives no theorem-content row, host-supply row, marker-repair row, admitted internalizer, replacement diagnostic, or detached refusal object.*

Proof. Project the accepted carrier. The gate row G, failed item F, refusal target R, diagnostic row D, and support rows H, C, P, N are the only visible public coordinates.

Use diagnostic scope closure and refusal handoff scope. They place every public refusal read on the displayed route from G to F, then through D, and then to R before replay and support rows are consumed.

Apply replay ledger scope to the support rows. Componentwise transport, displayed replay, provenance, and local naming preserve the same diagnostic packet and do not add repair, theorem-content, host-supply, or replacement diagnostic coordinates.

Finally use origin import boundary and composition input exhaustion. These exclude internalization, gate bypass, detached composed refusal content, and theorem content for the failed item, so the route G, F, D, R, H, C, P, N is the public diagnostic export boundary. □

The Lean follow-up for this chapter must provide `Derived/AuditGateFailureWitnessUp/TasteGate.lean` with $\mathsf{Nontrivial}$ $\mathsf{AuditGateFailureWitness}^{\uparrow}$ and $\mathsf{FieldFaithful}$ $\mathsf{AuditGateFailureWitness}^{\uparrow}$ instances for the same finite carrier.

Lean4 ✓ `BEDC.Derived.AuditGateFailureWitnessUp.AuditGateFailureWitnessTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{AuditGateFailureWitness}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditGateFailureWitness}^{\uparrow}$ packet is generated from a gate row, a failed boundary item, a refusal target, diagnostic readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditGateFailureWitness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 901.1, Definition 901.2, Definition 901.6, Definition 901.30, Definition 901.31, Definition 901.32, Definition 901.33, and

Lean4 ✓ `BEDC.Derived.AuditGateFailureWitnessUp.taste gate`

bind failed audit-gate verdicts as public diagnostic NameCert packets with visible refusal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditGateFailureWitness}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.AuditGateFailureWitnessUp.AuditGateFailureWitnessTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a public standard bridge, audit-policy completeness, automatic repair of failed targets, or theorem content for failed boundary items.

Upgrade path: No theory-axis upgrade remains; bridge checking may be recorded if a standard bridge target is supplied for the same diagnostic witness packet, refusal handoff, replay ledger, provenance, and local NameCert surface.

$\mathsf{CauchyWitnessLedger}^{\uparrow}$ is the Real-completion witness ledger read between chapter 1117 and chapter 1110; its finite route uses the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surfaces without quotient streams, selected host limits, countable choice, or ambient completeness.

Chapter 902

A Concrete Naming Certificate for $\mathsf{CauchyWitnessLedger}^{\uparrow}$

The $\mathsf{CauchyWitnessLedger}^{\uparrow}$ packet realizes the Real completeness route of chapter 9238 as one finite witness ledger. It joins the diagonal selector budget of chapter 853, the observation-budget packet of chapter 611, the seal synchronizer of chapter 886, and the completion-seal classifier of chapter 900. The packet is a NameCert surface for the displayed witness route, not a quotient stream, host equality principle, hidden limit selector, countable-choice row, or ambient completeness theorem.

Definition 902.1 (Cauchy witness ledger carrier)**.** *A* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, B, S, K, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *row exposing the finite precision, mesh, comparison, and selected diagonal budget for the completion-facing request;*
2. *B is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row repackaging the same budget into finite observation windows, dyadic readbacks,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows;*
3. *S is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row tying the selected diagonal budget, observation budget, shared-threshold route, and seal row into one synchronized read;*
4. *K is the* $\mathsf{RealCompletionSealClassifierUp}$ *row classifying the completion-facing seal only after the synchronized budget and seal rows have been consumed;*
5. *H records componentwise* hsame *transports for Q, B, S, K, C, P, N;*
6. *C records displayed* Cont *routes from the diagonal budget to the observation budget, from the observation budget to the synchronizer, and from the synchronizer to the completion-seal classifier;*
7. *P records* Pkg *provenance over* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}$ $\mathsf{RealCompletionSealClassifierUp}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected external limits, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerTasteGate single carrier alignment`

Theorem 902.2 (Cauchy witness ledger NameCert obligations)**.** *For the carrier of Definition 902.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (Q, B, S, K, H, C, P, N),$$

diagonal-budget exposure through Q*, observation-budget exposure through* B*, seal-synchronizer exposure through* S*, completion-seal classifier exposure through* K*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Later Lean TasteGate work must provide explicit* Nontrivial CauchyWitnessLedger$^{\uparrow}$ *and* FieldFaithful CauchyWitnessLedger$^{\uparrow}$ *instances for this same row inventory.*

Proof. Carrier admission is projection from the displayed tuple. The rows Q, B, S, K, H, C, P, N expose every public coordinate of the witness ledger.

The analytic rows are read in order. The diagonal budget Q supplies the finite precision, mesh, comparison, and selected diagonal surface; the observation budget B repackages that surface as windows, dyadic readbacks, regular-rational handoffs, and real seal rows.

The synchronizer S then ties the selected diagonal budget and observation route to the seal-facing read. The classifier K is reached only after that synchronized route has been displayed.

The structural rows H, C, P, N transport, replay, package, and name the same finite route. Since the carrier has no quotient stream, host equality, external limit, choice, or ambient completeness coordinate, these rows exhaust the local certificate surface. □

Theorem 902.3 (Cauchy witness ledger TasteGate nontriviality)**.** *For every accepted* CauchyWitnessLedger$^{\uparrow}$ *packet, the TasteGate-facing surface must contain two distinguishable accepted ledger witnesses whose separation is witnessed by at least one displayed analytic row among* Q, B, S, K*. Hence carrier admission for the packet cannot collapse to a one-point surface: a missing-row packet or a packet that reaches* K *while bypassing one of* Q, B, S *is rejected by the same local NameCert surface.*

Proof. Read the accepted carrier as

$$L = (Q, B, S, K, H, C, P, N).$$

The NameCert obligation theorem says that the public surface contains all four analytic rows Q, B, S, K and all four structural rows H, C, P, N. Therefore the TasteGate input is not a bare carrier name; it is the displayed ledger packet with analytic rows available for comparison.

The falsifiable prediction for this chapter fixes the rejection test: any admitted Real-facing read must factor through the displayed diagonal-budget, observation-budget, seal-synchronizer, and completion-seal rows. A packet with a missing analytic row, or a packet whose read bypasses one of these rows before reaching K, fails that test and is not an accepted witness for the ledger surface.

The independence witness separates this carrier from each nearest sibling. Each sibling lacks at least one of the four ledger coordinates, while CauchyWitnessLedger$^{\uparrow}$ requires the simultaneous factorization through all four. Thus the accepted packet supplies a witness pair for Lean's Nontrivial obligation: one witness reads the displayed budget route and another reads a displayed seal route separated by at least one of Q, B, S, K, and the local certificate rejects the attempted one-point collapse. □

Theorem 902.4 (Cauchy witness ledger FieldFaithful row)**.** *Any* FieldFaithful CauchyWitnessLedger$^{\uparrow}$ *instance for the same accepted packet must project exactly the rows*

$$Q, B, S, K, H, C, P, N.$$

Equality of all eight projected rows is the only permitted carrier equality route for TasteGate consumers. No compressed classifier, quotient stream, host real equality, selected external limit, or ambient completeness row may replace constructor-field faithfulness for this packet.

Proof. Start with the carrier definition and the NameCert obligations. The carrier has exactly the tuple $L = (Q, B, S, K, H, C, P, N)$, and the local certificate surface exposes precisely those analytic, transport, replay, provenance, and naming rows.

Apply Definition 902.5. The analytic part of the read is forced through $Q \longmapsto B \longmapsto S \longmapsto K$, so a faithful field projection cannot expose K from a compressed classifier while hiding one of the prior analytic fields.

Next use Definition 902.11 and Definition 902.12. Once all eight displayed rows agree, there is no remaining public coordinate from which a second carrier equality route could be chosen. Conversely, omitting any displayed row leaves one of the admitted route, transport, replay, provenance, or naming obligations unread.

Finally apply Definition 902.18. Quotient stream equality, host Real equality, selected external limits, countable choice, and ambient completeness are excluded from the carrier. Therefore TasteGate consumers can be field-faithful only by projecting the eight constructor rows and using equality of those rows as the carrier equality route. □

Theorem 902.5 (Cauchy witness ledger budget factorization)**.** *Every admitted completion-facing read of an accepted* CauchyWitnessLedger$^{\uparrow}$ *packet factors through the finite route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with the displayed structural rows H, C, P, N. Thus the read first consumes the CauchyDiagonalBudget$^{\uparrow}$ *precision, mesh, comparison, and selector data, then the* RealObservationBudget$^{\uparrow}$ *observation windows and handoff rows, then the* CauchySealBudgetSynchronizer$^{\uparrow}$ *seal route, and only then the* RealCompletionSealClassifierUp *completion-seal classifier.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier budget factorization`

Proof. Start from the carrier definition. The only completion-facing analytic rows are Q, B, S, K, and the only structural rows are H, C, P, N.

Apply the NameCert obligation theorem. It places diagonal-budget exposure before observation-budget exposure, observation-budget exposure before seal-synchronizer exposure, and seal-synchronizer exposure before the completion-seal classifier.

The route rows in C replay exactly that order, while H preserves the same coordinates under componentwise transport and P, N provide provenance and local naming. No hidden row remains from which a completion-facing consumer could bypass one of the four analytic rows. □

Theorem 902.6 (Cauchy witness ledger readback totality)**.** *Let*

$$L = (Q, B, S, K, H, C, P, N)$$

be an accepted CauchyWitnessLedger$^{\uparrow}$ *packet. Every analytic row Q, B, S, K and every structural row H, C, P, N has a public readback through the local* NameCert$_{\text{CauchyWitnessLedger}^{\uparrow}}$ *surface, and no public readback exists for a row outside this tuple.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerReadbackTotality`

Proof. Project the carrier of Definition 902.1. The accepted packet is the displayed finite tuple, so each analytic row and each structural row is available by projection.

Apply Definition 902.2. The local NameCert surface reads the four analytic rows in the order Q, B, S, K and reads the structural rows H, C, P, N as transport, replay, provenance, and naming data for the same packet.

The carrier has no ninth coordinate. Hence a public readback can only name one of the displayed rows, and no row outside L can be read through the accepted certificate. □

Theorem 902.7 (Cauchy witness ledger budget source coherence)**.** *For every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, the rows Q, B, and S share the same finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *source surface before the completion-seal classifier row K is exposed. Thus the classifier is reached only after one coherent budget source has passed through the diagonal row, the observation row, and the synchronizer row.*

Proof. Start with Definition 902.5. It forces every admitted completion-facing read through the route $Q \longmapsto B \longmapsto S \longmapsto K$, so K has no public exposure before the first three analytic rows have been consumed.

Use Definition 902.13. The observation row B reads the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface already fixed by the diagonal row Q.

Then apply Definition 902.14. The synchronizer S consumes that same $Q \longmapsto B$ source before K becomes public, so the classifier cannot be exposed from a separate stream, rational, dyadic, or Real surface. □

Theorem 902.8 (Cauchy witness ledger classifier consumer non-escape)**.** *Every consumer of the completion-seal classifier row K in an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet must also expose the route*

$$Q \longmapsto B \longmapsto S$$

and the structural rows H, C, P, N. Such a consumer has no access to quotient stream equality, host real equality, selected external limits, countable-choice rows, or ambient completeness theorems outside the displayed ledger.

Proof. Use Definition 902.10. The accepted packet exposes exactly Q, B, S, K, H, C, P, N, with no additional classifier channel.

Apply Definition 902.11 and Definition 902.17. A consumer reaches K only along the displayed analytic route, and replay through C requires the prior rows Q, B, S together with the structural data H, P, N.

Finally apply Definition 902.18. The forbidden external coordinates are absent from the carrier and from its NameCert surface, so a classifier consumer cannot leave the finite route through quotient streams, host Real equality, selected limits, countable choice, or ambient completeness. □

Theorem 902.9 (Cauchy witness ledger Real completion handoff)**.** *Let*

$$L = (Q, B, S, K, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer that reads L reaches the completion-seal classifier K only through the diagonal-budget row Q, observation-budget row B, seal-synchronizer row S, and structural rows H, C, P, N. The handoff exports no quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerRealCompletionHandoff`

Proof. Project the carrier and apply Definition 902.5. The consumer route reaches K by first reading Q, then B, then S, all replayed through C and named by N.

Read the sibling surfaces cited in the introduction. The diagonal-budget row supplies the finite selector layer, the observation-budget row supplies finite Real-facing windows and handoffs, the synchronizer ties those rows to the seal route, and the classifier consumes the synchronized seal before it can export a completion-facing read.

Finally use the non-escape clause built into the carrier and NameCert obligations. Since no forbidden analytic coordinate appears in L, the Real completion handoff has exactly the displayed finite ledger route. □

Theorem 902.10 (Cauchy witness ledger carrier admission)**.** *Every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet exposes exactly the eight rows*

$$Q, B, S, K, H, C, P, N.$$

Here Q is the diagonal-budget row, B is the observation-budget row, S is the seal-synchronizer row, K is the completion-seal classifier row, and H, C, P, N are respectively the componentwise transport, route replay, package-provenance, and local NameCert rows. No public read can escape this carrier inventory.

Proof. Project the accepted packet through Definition 902.1. The carrier is the finite tuple $L = (Q, B, S, K, H, C, P, N)$, so each displayed row is obtained by projection and no further coordinate is present in the tuple.

Apply Definition 902.2. The local NameCert surface is exactly carrier admission for the same tuple together with the four analytic rows and four structural rows.

The non-escape clause in the carrier rules out quotient stream equality, host real equality, selected external limits, countable choice, and ambient completeness. Hence every accepted public read is confined to the displayed row inventory. □

Theorem 902.11 (Cauchy witness ledger route determinacy)**.** *Let two completion-facing consumers of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet agree on the displayed route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

and on the structural rows H, C, P, N. Then they determine the same completion-seal classifier read: each consumer reaches K only after the diagonal budget, observation budget, seal synchronizer, componentwise transports, route replay, provenance, and local NameCert rows have been read in the displayed carrier.

Proof. Start with Definition 902.10. Both consumers read the same eight-row carrier inventory, so their public coordinates are confined to Q, B, S, K, H, C, P, N.

Apply Definition 902.5. The analytic part of any completion-facing read is forced through $Q \longmapsto B \longmapsto S \longmapsto K$, while the structural rows H, C, P, N preserve, replay, package, and name that route.

Since the consumers agree on the displayed route and on every structural row, there is no remaining carrier coordinate from which a second completion-seal classifier read could be selected. The classifier read is therefore determined by the common route through K. □

Theorem 902.12 (Cauchy witness ledger ledger exhaustion)**.** *Every public read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet is exhausted by the four analytic rows* Q, B, S, K *and the structural rows* H, C, P, N*. In particular, the ledger exports no quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.*

Proof. Use Definition 902.10. The accepted packet exposes exactly the eight displayed rows, with no additional public coordinate.

Apply Definition 902.11. Any completion-facing consumer that agrees on the route and structural rows has the same completion-seal classifier read, so a public read cannot branch into an independent classifier channel.

Finally apply Definition 902.9. The Real-facing handoff reaches K only through Q, B, S and the structural rows, and it exports none of the listed external objects. Thus the ledger is exhausted by the displayed analytic and structural rows. □

Theorem 902.13 (Cauchy witness ledger diagonal observation route)**.** *Every admitted read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet first consumes the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *row* Q *and then the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row* B *over the same finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surface. The read may continue through* S, K, H, C, P, N*, but it cannot enter the observation row before the diagonal-budget row has fixed the displayed stream, regular-rational, dyadic, and Real-seal route.*

Proof. Project the accepted carrier using Definition 902.1 and Definition 902.10. The only analytic rows available to the read are Q, B, S, K, with H, C, P, N carrying transport, replay, provenance, and local naming.

Apply Definition 902.5. It places the diagonal-budget row before the observation-budget row on the displayed completion route, so the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data seen by B are the same finite data already fixed by Q.

Use Definition 902.11 and Definition 902.12. They rule out a second observation route or a public coordinate outside the carrier, hence every admitted read follows the displayed diagonal-to-observation route. □

Theorem 902.14 (Cauchy witness ledger synchronizer classifier lock)**.** *For an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, the completion-seal classifier row* K *is public only after the synchronizer row* S *has consumed the same diagonal-budget row* Q *and observation-budget row* B *on the displayed route. Thus no consumer can expose* K *by separating the classifier from the synchronized* $Q \longmapsto B \longmapsto S$ *route.*

Proof. Start from Definition 902.1. The synchronizer row S and completion-seal classifier row K are adjacent analytic coordinates of the same carrier, not independent public objects.

Apply Definition 902.5 and Definition 902.9. The factorization orders the route as $Q \longmapsto B \longmapsto S \longmapsto K$, and the Real handoff reaches the completion-seal classifier only after the same budget and observation rows have been consumed.

Finally use Definition 902.10, Definition 902.11, and Definition 902.12. Carrier admission supplies no second classifier row, route determinacy fixes the classifier read once Q, B, S, H, C, P, N are fixed, and ledger exhaustion prevents a public escape around the synchronizer. □

Theorem 902.15 (Cauchy witness ledger seal budget nonescape)**.** *No consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *ledger can replace the finite route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with quotient stream equality, host real equality, selected external limits, countable choice, or ambient completeness. The seal-budget route remains the displayed BEDC route over Q, B, S, K, H, C, P, N*.*

Proof. Read the finite route from Definition 902.1 and Definition 902.5. The carrier lists the available rows, and the factorization fixes the analytic order from diagonal budget to observation budget, synchronizer, and classifier.

Use Definition 902.9. Every $\mathsf{Real}^{\uparrow}$-facing consumer reaches the completion-seal classifier through those displayed rows rather than through a host Real object or a selected external limit.

Apply Definition 902.10, Definition 902.11, and Definition 902.12. Together they keep the consumer inside Q, B, S, K, H, C, P, N and exclude quotient stream equality, host real equality, selected limits, countable choice, and ambient completeness from the public ledger. □

Theorem 902.16 (Cauchy witness ledger componentwise transport)**.** *Let*

$$L = (Q, B, S, K, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *carrier. Componentwise* hsame *transport along the ledger preserves the eight displayed rows* Q, B, S, K, H, C, P, N*: the diagonal-budget row remains* Q*, the observation-budget row remains* B*, the seal-synchronizer row remains* S*, the completion-seal classifier row remains* K*, and the structural rows remain* H, C, P, N*. No transported consumer obtains a row outside this finite tuple.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier componentwise transport`

Proof. Use the carrier definition to project the tuple $L = (Q, B, S, K, H, C, P, N)$. The row H is itself the displayed componentwise hsame ledger for the analytic rows Q, B, S, K and the structural rows C, P, N.

Apply Definition 902.2. It identifies the public NameCert surface with precisely the eight rows of the carrier, so any transported read still has to land in one of those rows.

Apply Definition 902.5 and Definition 902.9. The route and the Real-facing handoff both consume the same finite row inventory, hence componentwise transport cannot create a new completion coordinate or rename one of the displayed rows into a hidden analytic object. □

Theorem 902.17 (Cauchy witness ledger route replay exactness)**.** *For an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *carrier*

$$L = (Q, B, S, K, H, C, P, N),$$

every displayed Cont *replay in* C *reaches the analytic route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

before any $\mathsf{Real}^{\uparrow}$*-facing read of* L *is admitted. In particular, a consumer cannot read the completion-seal classifier* K *without first replaying the diagonal-budget row* Q*, the observation-budget row* B*, and the seal-synchronizer row* S*.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier route replay exactness`

Proof. Project C from the carrier definition. Its displayed routes are the three finite transitions from the diagonal budget to the observation budget, from the observation budget to the synchronizer, and from the synchronizer to the completion-seal classifier.

Apply Definition 902.2. The NameCert surface admits route replay only through the rows named by the carrier, with C supplying the displayed Cont edges and N naming the local packet.

Now apply Definition 902.5. The factorization fixes the order $Q \longmapsto B \longmapsto S \longmapsto K$, and Definition 902.9 states that every $\mathsf{Real}^{\uparrow}$ completion-facing consumer reaches K only through this route. Therefore exact replay through C is a precondition for any Real-facing read. □

Theorem 902.18 (Cauchy witness ledger forbidden-coordinate exclusion)**.** *Every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet exports only the finite rows*

$$Q, B, S, K, H, C, P, N.$$

It exports no quotient stream equality, no host real equality, no selected external limit, no countable-choice row, and no ambient completeness theorem. The exclusion is part of the BEDC-kernel certificate, not a consequence of an external Real-completion principle.

Proof. Read the final clause of Definition 902.1: the carrier has no coordinate for quotient stream equality, host real equality, selected external limits, countable choice, or ambient completeness.

Apply Definition 902.2. The accepted NameCert surface is exactly the finite row inventory Q, B, S, K, H, C, P, N, so a forbidden coordinate cannot be recovered as a named public row.

Apply Definition 902.5 and Definition 902.9. The Real-facing route factors through the finite BEDC rows before reaching the completion-seal classifier, and the handoff explicitly exports none of the listed external coordinates. Thus the exclusion is anchored in the carrier and its local certificate surface. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerForbidden coordinate exclusion`

Theorem 902.19 (Cauchy witness ledger public export)**.** *Every public* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *export to a* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer factors through the analytic route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

together with the structural rows H, C, P, N*. Here* Q *is the diagonal-budget row of chapter 853,* B *is the observation-budget row of chapter 611,* S *is the seal-synchronizer row of chapter 886, and* K *is the completion-seal classifier row of chapter 900.*

The public export contains no quotient stream equality, host real equality, selected external limit, countable-choice row, ambient completeness theorem, or second completion route outside the displayed finite ledger.

Proof. Start with Definition 902.5. It fixes the analytic spine of any completion-facing read as $Q \longmapsto B \longmapsto S \longmapsto K$, with the four sibling chapters supplying the displayed budget, observation, synchronizer, and classifier rows.

Use Definition 902.9 to pass the same spine to a $\mathsf{Real}^{\uparrow}$ completion-facing consumer. The handoff reaches the completion-seal classifier only after the diagonal budget, observation budget, and seal synchronizer have been read.

Apply Definition 902.11. Once the consumer agrees on the analytic spine and on H, C, P, N, the classifier read is fixed, so there is no independent completion route.

Finally combine Definition 902.12 with Definition 902.18. The ledger is exhausted by Q, B, S, K, H, C, P, N, and the excluded coordinates cannot be recovered from the public export. □

Theorem 902.20 (Cauchy witness ledger route coverage)**.** *Every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *ledger route is covered by the finite rows*

$$Q, B, S, K, H, C, P, N.$$

The route may read the diagonal-budget row, observation-budget row, seal-synchronizer row, completion-seal classifier row, componentwise transport row, displayed continuation row, package-provenance row, and local NameCert row, but it has no public coordinate outside this inventory.

Proof. Start with the carrier definition and the NameCert obligation theorem. They identify the accepted packet with the displayed tuple $L = (Q, B, S, K, H, C, P, N)$ and make those rows the complete local certificate surface.

Next use carrier admission, route determinacy, and ledger exhaustion. Carrier admission supplies the finite row inventory, route determinacy prevents a second completion-seal classifier read once the displayed route is fixed, and ledger exhaustion says every public read is already consumed by the same analytic and structural rows.

Finally apply public export and forbidden-coordinate exclusion. The public route is the Real-facing ledger route through $Q \longmapsto B \longmapsto S \longmapsto K$ with H, C, P, N, while quotient stream equality, host real equality, selected external limits, countable choice, and ambient completeness are absent from the carrier. □

Theorem 902.21 (Cauchy witness ledger budget-seal lock)**.** *No completion-seal read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet can reach the row K unless it has first read the diagonal-budget row Q, the observation-budget row B, and the seal-synchronizer row S on the same displayed BEDC route.*

Proof. Project the accepted carrier. The only analytic route to the completion-seal classifier is the ordered surface $Q \longmapsto B \longmapsto S \longmapsto K$.

Apply budget factorization and route replay exactness. Budget factorization places Q, then B, then S before K, and route replay exactness records that the displayed Cont row C must replay that same route before any $\mathsf{Real}^{\uparrow}$-facing read is admitted.

Use the Real completion handoff and public export to pass the lock to every completion-facing consumer. The handoff reaches K only through the three budget and seal rows, and the public export has no second completion route from which K could be selected independently. □

Theorem 902.22 (Cauchy witness ledger Real consumer nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet receives only the finite ledger route*

$$Q, B, S, K, H, C, P, N.$$

It receives no quotient stream, host equality, selected limit, countable-choice row, ambient completeness theorem, or completion-facing route outside the displayed BEDC rows.

Proof. Begin with the Real completion handoff. It sends each $\mathsf{Real}^{\uparrow}$-facing consumer through the diagonal budget, observation budget, seal synchronizer, completion-seal classifier, componentwise transports, route replay, provenance, and local NameCert rows.

Now combine route coverage with the budget-seal lock. Route coverage confines the consumer to Q, B, S, K, H, C, P, N, and the lock prevents the consumer from reading K before the Q, B, S budget and seal route has been consumed.

Finally use forbidden-coordinate exclusion. Since the excluded coordinates are not rows of the carrier and are not exported by the public route, the Real-facing consumer cannot recover a quotient stream, host equality, selected limit, countable-choice row, ambient completeness theorem, or alternate completion channel from the ledger. □

Theorem 902.23 (Cauchy witness ledger downstream unblock package)**.** *Every downstream L10 consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet that reads Real-completion data factors first through the finite ledger rows*

$$Q, B, S, K, H, C, P, N.$$

The consumer may read the Real-completion surface only after the diagonal-budget row Q, observation-budget row B, seal-synchronizer row S, completion-seal classifier row K, componentwise transport row H, route-replay row C, package-provenance row P, and local NameCert row N have been exposed on the displayed BEDC route.

Proof. Begin with Definition 902.5. It places the analytic read through $Q \longmapsto B \longmapsto S \longmapsto K$ before any completion-facing row can be consumed.

Next use Definition 902.13 and Definition 902.14. The first identifies the shared diagonal and observation source, and the second keeps the synchronizer and classifier attached to that same source.

Now apply Definition 902.17 and Definition 902.19. Replay supplies the row C for the same route, while public export carries exactly the finite surface with H, C, P, N to the downstream consumer.

Finally, Definition 902.18 removes every external completion coordinate. Thus the downstream L10 read is unblocked only by the displayed ledger packet, not by a hidden analytic principle. □

Theorem 902.24 (Cauchy witness ledger window seal consumer triad)**.** *In every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, the diagonal-budget row Q, observation-budget row B, and seal-synchronizer row S form the consumer triad that binds the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces before the completion-seal classifier row K is readable. No downstream consumer may detach one of these four surfaces from the same $Q \longmapsto B \longmapsto S$ route.*

Proof. Read the budget order from Definition 902.5. The rows Q, B, and S occur before K, so the classifier cannot supply the first completion-facing surface.

Use Definition 902.13. It binds the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces seen by B to the same source already fixed by Q.

Apply Definition 902.14. The synchronizer row S carries that same source into the classifier row K, which gives the three consumer rows their shared seal-facing boundary.

Finally combine Definition 902.17, Definition 902.19, and Definition 902.18. Replay and export preserve the triad for downstream use, and forbidden-coordinate exclusion rules out any detached stream, rational, dyadic, or Real surface. □

Theorem 902.25 (Cauchy witness ledger completion classifier readback)**.** *For every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, the completion-seal classifier row K is readable exactly through the displayed route*

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

It is never readable through quotient streams, selected limits, countable choice, ambient completeness, or any Real-completion channel outside the finite ledger surface with H, C, P, N.

Proof. Start with Definition 902.5. It displays the only analytic path to K as $Q \longmapsto B \longmapsto S \longmapsto K$.

Next use Definition 902.13 and Definition 902.14. They keep the classifier attached to the same diagonal-budget, observation-budget, and seal-synchronizer source.

Apply Definition 902.17 and Definition 902.19. These theorems make the readback public only as the replayed finite route with the structural rows H, C, P, N.

Finally apply Definition 902.18. Since the forbidden coordinates are absent from the carrier and from the exported surface, K has no readback through quotient streams, selected limits, countable choice, ambient completeness, or an alternate completion channel. □

Theorem 902.26 (Cauchy witness ledger root downstream totality)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *ledger must expose exactly the rows*

$$Q, B, S, K, H, C, P, N.$$

The rows Q, B, S, and K are respectively the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, *and* $\mathsf{RealCompletionSealClassifierUp}$ *rows, while H, C, P, N are the componentwise transport, route replay, package-provenance, and local NameCert rows. No downstream consumer obtains a second Real-completion route or a row outside this finite ledger surface.*

Proof. First project the accepted carrier. By Definition 902.10, the only public rows are Q, B, S, K, H, C, P, N, with Q, B, S, and K naming the four displayed Real-completion route components.

Next apply Definition 902.20. It covers every accepted ledger route by the same eight rows, so any downstream read is already inside the finite carrier inventory.

Finally use Definition 902.22. A $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer receives no quotient stream, host equality, selected limit, countable-choice row, ambient completeness theorem, or alternate completion route. Thus the downstream surface is exactly the displayed ledger. □

Theorem 902.27 (Cauchy witness ledger root completion package)**.** *The diagonal-observation route, synchronizer-classifier lock, route replay exactness, and public export form one root handoff package for every L10* $\mathsf{Real}^{\uparrow}$ *completion consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *ledger. The package exports the route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with H, C, P, N, and it exports no quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.

Proof. Start with Definition 902.13. It places the observation row B after the diagonal-budget row Q over the same finite stream, regular-rational, dyadic, and Real-seal surface.

Next apply Definition 902.14. The synchronizer row S must consume the same $Q \longmapsto B$ source before the classifier row K becomes public.

Then use Definition 902.17. The displayed replay row C reaches $Q \longmapsto B \longmapsto S \longmapsto K$ before any $\mathsf{Real}^{\uparrow}$-facing read is admitted.

Finally apply Definition 902.19. The public surface exports exactly the analytic route with H, C, P, N and none of the external Real-completion coordinates. These four facts are the root handoff package available to the L10 consumer. □

Theorem 902.28 (Cauchy witness ledger root seal readback)**.** *Any completion-seal readback through an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *ledger reaches the classifier row K only after the displayed route*

$$Q \longmapsto B \longmapsto S$$

has been consumed. The readback cannot bypass the $\mathsf{RealCompletionSealClassifierUp}$ *row and cannot recover the seal from quotient stream equality, host real equality, selected external limits, countable choice, or ambient completeness.*

Proof. Begin with Definition 902.5. It orders the analytic route as $Q \longmapsto B \longmapsto S \longmapsto K$, so the completion-seal classifier is never the first row read.

Use Definition 902.14. The synchronizer-classifier lock keeps K attached to the same diagonal-budget and observation-budget source consumed by S.

Finally combine Definition 902.18 with Definition 902.22. The forbidden coordinates are absent from the ledger, and a Real-facing consumer receives no alternate completion channel. Therefore every seal readback reaches K only through the displayed finite route. □

Theorem 902.29 (Cauchy witness ledger StreamName window source lock)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every accepted* $\mathsf{StreamName}^{\uparrow}$ *window read used by L is sourced by the diagonal-budget row Q and the observation-budget row B, then transported through H, C, P, N. No hidden stream coordinate, hidden tail oracle, quotient stream, selected limit, countable-choice row, or ambient completeness theorem can supply a second window source for the ledger.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier streamname window source lock`

Proof. Start from the carrier definition Definition 902.1. It lists exactly the rows Q, B, S, K, H, C, P, N and gives no coordinate for a hidden stream or tail oracle.

The diagonal-observation route Definition 902.13 fixes B as the observation-budget read of the same finite stream surface exposed by Q. The synchronizer-classifier lock Definition 902.14 then keeps S and K attached to that same $Q \longmapsto B$ source.

Route coverage Definition 902.20 confines every ledger read to Q, B, S, K, H, C, P, N, and the budget-seal lock Definition 902.21 prevents the seal side from being read before the displayed budget rows have been consumed.

Finally, Definition 902.22 excludes the completion-facing escape coordinates. Hence every accepted $\mathsf{StreamName}^{\uparrow}$ window read in the ledger is sourced by Q and B through the displayed structural rows. □

Theorem 902.30 (Cauchy witness ledger RegSeqRat readback totality)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumed by B, S, or K is supplied by the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *route already displayed by Q and B, and then replayed through H, C, P, N. The ledger admits no detached rational readback, hidden dyadic completion, tail oracle, quotient stream, or ambient Real completeness coordinate.*

Proof. By Definition 902.1, the accepted packet has only the rows Q, B, S, K, H, C, P, N. Thus a rational readback consumed by B, S, or K must be read through one of the displayed ledger coordinates.

Use Definition 902.13 to identify the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface with the finite $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ data exposed by the diagonal and observation budgets. Then Definition 902.14 transports that readback through the synchronizer and classifier rows without adding a new rational source.

The route-coverage theorem Definition 902.20 and the budget-seal lock Definition 902.21 make the displayed route total for the ledger reads. The nonescape theorem Definition 902.22 removes the remaining Real-completion escape coordinates, so the readback is exactly the finite route already shown by Q and B. □

Theorem 902.31 (Cauchy witness ledger Real seal refusal boundary)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *completion request outside the route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

lands in the forbidden-coordinate refusal boundary. Such a request exports no quotient stream, host equality, selected limit, countable-choice coordinate, ambient completeness theorem, hidden tail oracle, or alternate Real seal.

Proof. The carrier definition Definition 902.1 exposes only Q, B, S, K, H, C, P, N. A request outside $Q \longmapsto B \longmapsto S \longmapsto K$ therefore has no carrier row on which to land except the refusal boundary for forbidden coordinates.

The diagonal-observation route Definition 902.13 supplies the first budget handoff, and Definition 902.14 supplies the seal-facing continuation through S and K. Route coverage Definition 902.20 states that every accepted route is exhausted by those displayed rows.

The budget-seal lock Definition 902.21 prevents a direct Real seal read outside the budget route. Finally, Definition 902.22 rules out quotient streams, host equality, selected limits, countable choice, ambient completeness, hidden tails, and alternate Real seals. Hence every off-route completion request is refused at the forbidden-coordinate boundary. □

Theorem 902.32 (Cauchy witness ledger Real seal consumer totality)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer of L either factors through the displayed route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with its transport, replay, provenance, and local-name rows H, C, P, N, or lands in the forbidden-coordinate refusal boundary of Definition 902.31. Thus the ledger has no completion-facing consumer outside the displayed diagonal-budget, observation-budget, seal-synchronizer, and completion-seal route.

Proof. Project the accepted packet through Definition 902.1. Its only completion-facing rows are Q, B, S, K together with the structural rows H, C, P, N, so a consumer must first read one of those displayed coordinates.

Route coverage Definition 902.20 makes that displayed route total for accepted ledger reads. The budget-seal lock Definition 902.21 then forces a seal-facing read to pass through the budget side before reaching S and K.

It remains to classify off-route requests. Apply Definition 902.22 and then the refusal boundary Definition 902.31. Every remaining completion demand is refused as a forbidden coordinate rather than exported as an alternate $\mathsf{Real}^{\uparrow}$ completion route. □

Falsifiable prediction: If a CauchyWitnessLedgerUp packet is accepted, every RealUp completion-facing read factors through the displayed diagonal-budget, observation-budget, seal-synchronizer, and completion-seal rows; a counterexample is any admitted read bypassing one row.

Independence witness: Nearest siblings: cauchydiagonalbudget, realobservationbudget, cauchysealbudgetsynchronizer, realcompletionsealclassifier. None is bijective to CauchyWitnessLedgerUp because each sibling lacks at least one of the four ledger coordinates and the combined packet has a NameCert obligation requiring simultaneous factorization through all four.

Theorem 902.33 (Cauchy witness ledger source sibling readback)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every completion-facing read of L exposes the four sibling source rows in the order*

$$\mathsf{CauchyDiagonalBudget}^{\uparrow} \longmapsto \mathsf{RealObservationBudget}^{\uparrow} \longmapsto \mathsf{CauchySealBudgetSynchronizer}^{\uparrow} \longmapsto \mathsf{RealCompletionSealC}$$

before any completion-seal classifier consumer reads K.

The consumer receives the displayed source rows, the componentwise transport row H, route replay C, package provenance P, and the local naming row N. It receives no quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.

Proof. Project the carrier of Definition 902.1. Its completion-facing source rows are exactly Q, B, S, K, with H, C, P, N carrying transport, replay, provenance, and local naming for those rows.

The budget factorization and source coherence are the displayed route constraints. The diagonal-observation route of Definition 902.13 first reads the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ row Q and the $\mathsf{RealObservationBudget}^{\uparrow}$ row B. The synchronizer-classifier lock of Definition 902.14 then carries the read through S before the classifier row K is consumed.

Classifier consumer non-escape and forbidden-coordinate exclusion close the boundary. Apply Definition 902.22 and Definition 902.31. Any request for a quotient stream, host equality, selected limit, countable-choice row, or ambient completeness theorem has no carrier coordinate and is refused outside the source-sibling route. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier source sibling readback`

Theorem 902.34 (Cauchy witness ledger Real-and-Completion sibling route)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every Real-and-Completion consumer route that reads L and reaches the completion-seal classifier must pass through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic ledger of chapter 291, the seal boundary of chapter 38, and the public route of chapter 386. The route exports no quotient stream, host equality, selected limit, hidden tail oracle, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 902.1. The accepted packet supplies exactly the rows Q, B, S, K, H, C, P, N; thus any route to the completion-seal classifier must be read through the displayed diagonal-budget, observation-budget, synchronizer, classifier, transport, replay, provenance, and naming rows.

First use Definition 902.33. It orders the local route through $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, and $\mathsf{RealCompletionSealClassifierUp}$ before the classifier row K is consumed.

Next attach the CauchyLimitSeal root L10 sibling route Definition 386.176 and its non-escape boundary Definition 386.179. Those results force the completion-facing seal route through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surface before any public limit-facing read is admitted.

Combining the two routes places the witness ledger inside the same Real-and-Completion dependency lattice as the seal route. The structural rows H, C, P, N only transport, replay, package, and name that finite path; they add no quotient stream, host equality, selected limit, hidden tail oracle, or ambient completeness coordinate. □

Theorem 902.35 (Cauchy witness ledger Real seal cover consumer lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer that reads a* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet over a finite cover must pass through the displayed diagonal-budget row Q, observation-budget row B, seal synchronizer row S, completion-seal classifier row K, and the support rows H, C, P, N. Cover and tail refinements are delegated to the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff surface together with* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*; the ledger exports no quotient stream, host equality, selected external limit, countable-choice row, ambient completeness principle, or second seal coordinate.*

Proof. Begin with budget factorization and the Real completion handoff. By Definition 902.5 and Definition 902.9, any completion-facing read reaches K only through the ordered route $Q \longmapsto B \longmapsto S \longmapsto K$ with support from H, C, P, N.

Use route determinacy and ledger exhaustion next. Definition 902.11 fixes the completion route once the displayed rows agree, and Definition 902.12 leaves no public coordinate outside Q, B, S, K, H, C, P, N.

For finite-cover and tail refinements, cite the L10 consumer route through Definition 902.64 and Definition 902.67. These results delegate the consumer-side refinement to the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff route.

Finally apply the tail-fusion surface of chapter 658. The $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ route supplies the finite cofinal tail consumer without adding quotient streams, host equality, selected external limits, countable choice, ambient completeness, or a second seal coordinate to the witness ledger. □

Theorem 902.36 (Cauchy witness ledger budget Real route totality)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every budget read,* $\mathsf{StreamName}^{\uparrow}$ *window read,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *completion-facing read of L routes through the finite rows Q, B, S, K, H, C, P, N. The route is the displayed diagonal-budget, observation-budget, synchronizer, and completion-seal path, with no completion choice, quotient representative, host limit, detached rational readback, or second stream source.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier budget real route totality`

Proof. Apply Definition 902.5. It orders the budget-facing route through $Q \longmapsto B \longmapsto S \longmapsto K$, while the carrier and NameCert rows supply only H, C, P, N as structural coordinates.

Use Definition 902.29, Definition 902.30, and Definition 902.31. These rows identify the StreamName source, the RegSeqRat readback, and the off-route Real refusal boundary with the same finite ledger. Hence every admitted read is exhausted by Q, B, S, K, H, C, P, N. □

Theorem 902.37 (Cauchy witness ledger StreamName RegSeq factorization)**.** *For an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet $L = (Q, B, S, K, H, C, P, N)$, the* $\mathsf{StreamName}^{\uparrow}$ *window source and every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factor through the same $Q \longmapsto B$ budget source before S or K is public. Componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row preserve this source without adding a tail oracle, detached dyadic completion, or host Real limit.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier streamname regseq factorization`

Proof. Read Definition 902.29. The stream window is fixed by the diagonal-budget and observation-budget rows, then transported through H, C, P, N.

Now apply Definition 902.30. The regular-rational readback is supplied by the same finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ route already displayed by Q and B. The synchronizer-classifier lock keeps S and K attached to that source, so no independent rational or Real-completion coordinate is introduced. □

Theorem 902.38 (Cauchy witness ledger Real completion nonescape)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be accepted. Any* $\mathsf{Real}^{\uparrow}$ *completion read admitted by L reaches the classifier row K only through the finite route $Q \longmapsto B \longmapsto S \longmapsto K$ and the structural rows H, C, P, N. Reads outside that route fall under the Real seal refusal boundary and export no selected limit, quotient stream, host equality, countable-choice row, ambient completeness theorem, hidden tail oracle, or alternate seal.*

Proof. Start with Definition 902.9 and Definition 902.28. They place every completion-facing read behind the diagonal budget, observation budget, synchronizer, and completion-seal classifier.

Finally use Definition 902.31. A request outside the displayed route has no carrier coordinate in BHist, no accepted hsame or Cont replay, and no Pkg or NameCert witness. Thus the read remains finite and cannot escape to an external completion principle. □

Theorem 902.39 (Cauchy witness ledger root budget row totality)**.** *Every accepted root read of $L = (Q, B, S, K, H, C, P, N)$ exposes the diagonal-budget row Q and the observation-budget row B before any completion-seal classifier read is public.*

Proof. Use Definition 902.1, Definition 902.5, and Definition 902.6. The carrier has Q and B as displayed analytic rows, the budget factorization orders them before S and K, and readback totality gives no root read outside Q, B, S, K, H, C, P, N. □

Theorem 902.40 (Cauchy witness ledger root seal route totality)**.** *Every root seal read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet passes through the finite route $Q \longmapsto B \longmapsto S \longmapsto K$.*

Proof. Apply Definition 902.28 and Definition 902.14. The first theorem places seal readback behind the displayed route, and the second keeps K attached to the same $Q \longmapsto B \longmapsto S$ source. □

Theorem 902.41 (Cauchy witness ledger root NameCert threshold)**.** *The root rows Q, B, S, K together with H, C, P, N form the* $\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$ *threshold surface for the witness ledger.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root namecert threshold`

Proof. Read Definition 902.2, Definition 902.5, and Definition 902.14. These statements supply the displayed obligation rows, order the budget route, and bind the synchronizer-classifier threshold without adding any row outside Q, B, S, K, H, C, P, N. □

Theorem 902.42 (Cauchy witness ledger formal target surface)**.** *For an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet $L = (Q, B, S, K, H, C, P, N)$, the encoded target* `BEDC.Derived.CauchyWitnessLedgerUp.taste_gate` *is the formal target surface for the finite route Q, B, S, K, H, C, P, N. Any future scaffold checking of that target must preserve the diagonal-budget row, observation-budget row, synchronizer row, completion-seal classifier row, componentwise transport rows, displayed replay rows, provenance row, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier formal target surface`

Proof. The root budget threshold is Definition 902.39. It forces the diagonal-budget row Q and observation-budget row B to be public before the completion-seal classifier is read.

The root seal threshold is Definition 902.40. It confines the seal route to $Q \longmapsto B \longmapsto S \longmapsto K$.

The local naming threshold is Definition 902.41. Together with Definition 902.38, it keeps completion-facing reads inside the same finite route and excludes selected limits, quotient streams, host equality, countable choice, ambient completeness, hidden tails, and alternate seals. Thus the named Lean target is exactly the finite formal surface whose later checking must preserve those rows. □

Theorem 902.43 (Cauchy witness ledger four-face Real completion lock)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Its formal target surface factors through exactly the four Phase Real and Completion faces* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with only the support rows* BHist, hsame, Cont, Pkg, *and* NameCert. *Thus every roadmap-exit*

consumer of the ledger reads the same four-face route and no route adds quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows.

Proof. Begin with Definition 902.42. It identifies the accepted formal surface with the finite route Q, B, S, K, H, C, P, N, so a consumer can expose only the budget, window, readback, Real-seal classifier, and structural rows already carried by L.

Next apply Definition 126.500 and Definition 126.502. These force the Real-completion exit route through the four displayed faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and exhaust every consumer read at that boundary.

Finally use Definition 126.498. It attaches the Cauchy witness ledger to that same four-object exit packet through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows. Since neither the ledger surface nor the exit packet carries quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows, no second completion-facing route remains. □

Theorem 902.44 (Cauchy witness ledger budget-selector four-face exhaustion)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ packet, and let $F = (B_0, S_0, W, D, R, E, H_0, C_0, P_0, N_0)$ be an accepted $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ packet whose selected finite windows W are the same $\mathsf{StreamName}^{\uparrow}$ window face consumed by L. Then every Phase Real and Completion consumer of L factors through the selector route of F and the same terminal four-face packet*

$$\mathsf{FiniteObservationBudgetSelector}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with only the support rows BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Thus passing through the observation-budget selector preserves the terminal route already locked by L: the consumer receives the selected window face, the dyadic ledger face, the regular-rational readback face, and the Real seal face, but no quotient stream, selected limit, host sequence equality, ambient Real completeness, metric-completeness datum, countable-choice row, or standard bridge row.

Proof. Begin with Definition 902.43. It places every roadmap-exit consumer of L on the four Phase Real and Completion faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Move to the selector packet. By Definition 782.7, the accepted packet F determines one finite route from its budget row through the selected schedule, window family, dyadic ledger, regular readback, and seal. By Definition 782.9, every dyadic observation exported by D is already read from the selected finite windows W.

Use the shared-window hypothesis to identify W with the $\mathsf{StreamName}^{\uparrow}$ window face consumed by L. The selector therefore supplies exactly the same window and dyadic-ledger coordinates that the ledger route exposes before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal are public.

Finally apply Definition 126.449. Once the shared window, dyadic ledger, readback, seal, and displayed support rows agree, the Phase Real and Completion consumer has no second terminal route. Since neither the selector route nor the ledger route carries quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows, passing through $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ preserves the same terminal four-face packet. □

Theorem 902.45 (Cauchy witness ledger open-phase budget route handoff)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted $\mathsf{CauchyWitnessLedger}^{\uparrow}$ packet. Suppose its displayed rows include the budget selector*

route, the completion-seal route, the $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *factorization, and the* $\mathsf{Real}^{\uparrow}$ *completion nonescape boundary. Then* L *determines one Real-and-completion handoff surface through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *supported only by the finite rows* BHist, hsame, Cont, Pkg, *and* NameCert.

Every open-phase consumer reading the packet therefore receives the same dyadic budget, named stream window, regular-rational readback, and Real seal coordinates. No off-packet completion coordinate, quotient stream, selected limit, host sequence equality, ambient Real completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row is exported by the handoff.

Proof. Start with Definition 902.36 and Definition 902.40. These rows place the budget reads and the completion-seal reads on the finite route $Q \longmapsto B \longmapsto S \longmapsto K$, with H, C, P, N as the only structural support.

Next use Definition 902.37 and Definition 902.38. The named stream source and the regular-rational readback both factor through the same budget source, while every completion-facing read remains inside the displayed Real seal boundary.

Apply Definition 902.43 and Definition 902.44. They identify the terminal consumer surface with the four open Phase Real and Completion faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, even when the observation-budget selector mediates the selected windows.

Thus a downstream consumer can only replay the finite BHist, hsame, Cont, Pkg, and NameCert route already carried by L. Since each cited row is exhausted by that route, no coordinate remains from which an external completion object or second Real handoff could be read. □

Theorem 902.46 (Cauchy witness ledger selector-seal terminality)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Let* $T = (B_0, W, R, L_0, E, H_0, C_0, P_0, N_0)$ *be an accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet whose selected window row* W, $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L_0 *agree with the ledger route exposed by* L *through the observation-budget selector.*

Then every Phase Real and Completion consumer of L *factors through the unique terminal route*

$$\mathsf{CauchyWitnessLedger}^{\uparrow} \longmapsto \mathsf{FiniteObservationBudgetSelector}^{\uparrow} \longmapsto \mathsf{RealCompletionSelectorSeal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto$$

with only the support rows BHist, hsame, Cont, Pkg, *and* NameCert. *No quotient stream, selected limit, host equality, ambient completeness, countable-choice row, second completion route, or standard bridge row remains after this factorization.*

Proof. Start with Definition 902.45. It places every open-phase consumer of L on the finite Real-and-completion handoff surface through the displayed budget, named stream window, regular-rational readback, and Real seal rows, with only BHist, hsame, Cont, Pkg, and NameCert support.

Next apply Definition 902.44. The observation-budget selector transports the same selected window and readback information without adding quotient streams, selected limits, host sequence equality, ambient Real completeness, metric-completeness data, countable choice, or standard bridge rows.

Use the agreement hypotheses for T. The selected window W, readback row R, and limit row L_0 are the same rows already consumed by the ledger route, so the selector-seal packet is attached to the handoff rather than forming a second completion source.

Finally use Definition 1081.3 and Definition 1081.4. Determinacy gives a single finite selector-seal route, and non-escape exhausts every completion consumer at $B_0, W, R, L_0, E, H_0, C_0, P_0, N_0$.

Hence the only remaining route is the displayed terminal chain from $\mathsf{CauchyWitnessLedger}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ through the selector and seal packets. □

Theorem 902.47 (Cauchy witness ledger selector agreement terminal collapse)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Let $F = (B_0, S_0, W, D, R_F, E_F, H_F, C_F, P_F, N_F)$ be an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *packet, and let $T = (B_1, W_1, R_T, L_1, E_T, H_T, C_T, P_T, N_T)$ be an accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose the three packets agree on the selected window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, and the* $\mathsf{Real}^{\uparrow}$ *seal row exposed by the ledger route.*

Then every $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{S}^{1\uparrow}$ *consumer of the Real-and-completion surface factors through the single terminal chain*

$$\mathsf{CauchyWitnessLedger}^{\uparrow} \longmapsto \mathsf{FiniteObservationBudgetSelector}^{\uparrow} \longmapsto \mathsf{RealCompletionSelectorSeal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto$$

The consumer receives only the displayed selected-window route, regular-rational readback, limit-seal row, Real seal row, and the support rows BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *It cannot introduce quotient streams, selected limits, host equality, ambient completeness, countable choice, a second completion route, or any separate RealAlgOrder, MetricSpace, or* $\mathsf{S}^{1\uparrow}$ *completion source.*

Proof. First attach the ledger to the selector-seal route. By Definition 902.46, the accepted ledger packet and selector-seal packet already force every Phase Real and Completion consumer through the displayed route from $\mathsf{CauchyWitnessLedger}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$.

Next identify the four open faces. Apply Definition 126.449 to the agreed selected window, dyadic observation window, regular readback, and Real seal rows, and then apply Definition 38.728. These rows are consumed as one terminal four-face packet rather than as independent completion coordinates.

Now read the downstream positive-boundary consumers. The RealAlgOrder, MetricSpace, and $\mathsf{S}^{1\uparrow}$ consumer theorems use the same positive Real boundary: they read the Real row after the StreamName, RegSeqRat, CauchyLimitSeal, and Real seal faces have already been fixed. Since the hypotheses identify those faces across L, F, and T, no consumer in this family has another source from which a completion route could be read.

Finally use nonescape for the selector-seal and ledger routes. The selector-seal packet exports no quotient stream, selected limit, host equality, ambient completeness, countable choice, or second completion route, and the ledger route exports no off-packet Real completion data. Thus the only available route for the three consumer families is the displayed terminal chain. □

Corollary 902.48 (Cauchy witness ledger budget Real route strict local obstruction)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Any displayed failure, refusal, separation, or countermodel row attached to the budget Real route of L is a strict local obstruction: it blocks only the proposed converse read, detached reuse, or off-route completion request that tries to bypass*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

and the structural rows H, C, P, N. Every consumer-facing use that remains admitted factors through the diagonal-budget row, observation-budget row, synchronizer row, completion-seal classifier row, componentwise transport row, displayed replay row, provenance row, and local $\mathsf{NameCert}$ *row. A request using any unlisted coordinate is refused at the Real seal boundary and exports no selected limit, quotient stream, host equality, countable-choice row, ambient completeness theorem, hidden tail oracle, alternate seal, or external validation witness.*

Consequently the obstruction cannot be reinterpreted as a global impossibility theorem for $\mathsf{Real}^{\uparrow}$ *completion-facing reads, as a host theorem about Real completeness, or as hidden validation by an external completion principle. It is only the local refusal of the named converse or reuse route outside the displayed ledger packet.*

Proof. Start with Definition 902.36. It says that every budget read, $\mathsf{StreamName}^{\uparrow}$ window read, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ completion-facing read admitted by L routes through exactly the finite rows Q, B, S, K, H, C, P, N. The same statement names the diagonal-budget, observation-budget, synchronizer, and completion-seal path as the whole admitted route and excludes completion choice, quotient representatives, host limits, detached rational readbacks, and second stream sources from that path.

Now apply Definition 902.38. A completion read reaches K only through $Q \longmapsto B \longmapsto S \longmapsto K$ together with H, C, P, N. If a displayed failure, refusal, separation, or countermodel row is triggered, it is triggered by a request outside that finite route; the nonescape theorem sends such a request to the Real seal refusal boundary and gives it no selected limit, quotient stream, host equality, countable-choice row, ambient completeness theorem, hidden tail oracle, or alternate seal.

Thus the consumer-facing alternatives split sharply. An admitted consumer factors through the displayed ledger rows by totality. A non-admitted consumer is rejected as an unlisted coordinate by nonescape. The refusal branch therefore has no carrier coordinate from which to state a global impossibility result, no host Real object from which to derive an ambient theorem, and no exported external witness from which hidden validation could be read. The only blocked object is the named converse or reuse request that attempted to leave the finite ledger route. □

Theorem 902.49 (Cauchy witness ledger root source lock)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. The diagonal-budget row* Q*, observation-budget row* B*, synchronizer row* S*, and completion-seal classifier row* K *use one shared finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source surface. Thus the public root read of* L *cannot attach* K *to a stream, rational, dyadic, or Real source separate from the source already read by* Q*,* B*, and* S*.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root source lock`

Proof. Begin with Definition 902.1. The accepted packet exposes only Q, B, S, K, H, C, P, N, so every source visible at the root must be read through one of those rows.

Apply Definition 902.5 and Definition 902.13. The budget factorization orders the analytic route as $Q \longmapsto B \longmapsto S \longmapsto K$, and the diagonal-observation route identifies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface seen by B with the one already fixed by Q.

Finally use Definition 902.14. The synchronizer carries the same $Q \longmapsto B$ source into K, so the classifier has no separate source row from which an alternate completion-facing read could be exposed. □

Theorem 902.50 (Cauchy witness ledger root source-window route lock)**.** *Every accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *completion-facing root read uses one shared finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source-window route through*

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

The read cannot attach a second stream window, rational readback, dyadic tolerance, real seal, hidden tail, quotient stream, host real equality, or ambient completeness row to the displayed carrier rows Q, B, S, K, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root source window route lock`

Proof. First apply the root source lock of Definition 902.49. It identifies the stream, regular-rational, dyadic, and real source surface of the completion-seal classifier with the source surface already read by Q, B, S.

Next use Definition 902.29. The stream window visible to a completion consumer is the window supplied by the same finite budget route, so the consumer cannot attach a separate window source before reading K.

Apply the refusal boundary of Definition 902.52. Any proposed rational readback, dyadic tolerance, real seal, hidden tail, quotient stream, host real equality, or ambient completeness coordinate outside the route $Q \longmapsto B \longmapsto S \longmapsto K$ is refused at the root boundary.

Finally, Definition 902.55 says every downstream completion-facing consumer factors through the displayed ledger rows. The common source-window route is therefore the whole consumer-facing route, not one branch among several. □

Theorem 902.51 (Cauchy witness ledger root budget coherence)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ and $L' = (Q', B', S', K', H', C', P', N')$ be accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packets. If the two public reads agree on Q, B, S, H, C, P, N, then they determine the same public completion-seal classifier read K. In particular, no accepted root read can split the classifier from the displayed budget route $Q \longmapsto B \longmapsto S \longmapsto K$.*

Proof. Start with Definition 902.11. Agreement on the displayed route and structural rows fixes the completion-facing route, so the public read has no second branch on which K could vary.

Use Definition 902.7. The rows Q, B, and S share one finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source before K is exposed.

Now apply Definition 902.25. The classifier readback is exactly the displayed route through $Q \longmapsto B \longmapsto S \longmapsto K$ together with H, C, P, N. Since those route and structural rows already agree, the public classifier read agrees as well. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root budget coherence`

Theorem 902.52 (Cauchy witness ledger root budget refusal boundary)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Any attempted public root read that gives the completion-seal classifier K a stream,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *tail, limit, or seal source not already exposed through*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

is routed to the displayed refusal boundary. Such a read cannot export a quotient stream, host $\mathsf{Real}^{\uparrow}$ *equality, selected external limit, countable-choice row, hidden tail, alternate seal, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root budget refusal boundary`

Proof. Read the accepted carrier through Definition 902.1. Its public root surface is exactly Q, B, S, K, H, C, P, N, and the completion-facing rows are displayed through the finite route $Q \longmapsto B \longmapsto S \longmapsto K$.

Apply Definition 902.49. The stream, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources visible to K are the same sources already read by Q, B, and S, so a proposed second source for the classifier is not a carrier row.

Next apply Definition 902.51. Once Q, B, S, H, C, P, N are fixed, the public classifier read K is determined by the displayed budget route, so a different tail, limit, or seal source cannot produce a second accepted classifier read.

Finally use Definition 902.55. Downstream completion consumers have no escape coordinate outside the displayed ledger, and the requested quotient stream, host $\mathsf{Real}^{\uparrow}$ equality, selected external limit, countable-choice row, hidden tail, alternate seal, or ambient completeness coordinate is therefore refused rather than exported. □

Theorem 902.53 (Cauchy witness ledger root budget classifier dichotomy)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every public completion-seal classifier read of L is exactly one of the following:*

1. *it follows the displayed route*
$$Q \longmapsto B \longmapsto S \longmapsto K$$
over the shared $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *source surface exposed by the root ledger;*

2. *it is routed to the refusal boundary and therefore exports no quotient stream, host* $\mathsf{Real}^{\uparrow}$ *equality, selected external limit, countable-choice row, hidden tail, alternate seal, or ambient completeness coordinate.*

Proof. First fix the source surface. By Definition 902.49, the rows Q, B, S, K read one shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source, so an accepted classifier read cannot begin from a second stream, rational, dyadic, or Real source.

Next bind the accepted branch. Definition 902.51 says that once the displayed budget route and structural rows are fixed, the completion-seal classifier read is determined by $Q \longmapsto B \longmapsto S \longmapsto K$. This gives the first side of the dichotomy.

It remains to account for every read not carried by that route. Definition 902.52 sends any classifier read with a detached source, tail, limit, or seal coordinate to the refusal boundary.

Finally apply Definition 902.55. Downstream $\mathsf{Real}^{\uparrow}$-completion consumers have no coordinate outside the displayed ledger, so a classifier read is either the accepted route above or a refused request, with no third exported surface. □

Theorem 902.54 (Cauchy witness ledger root diagnostic refusal dichotomy)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every root diagnostic read or completion-facing consumer read of L is exactly one of the following:*

1. *it follows the displayed route*
$$Q \longmapsto B \longmapsto S \longmapsto K$$
over the shared $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces exposed by the ledger;*

2. *it is refused at the displayed root boundary and exports no quotient stream, selected limit, host equality, countable-choice row, ambient completeness coordinate, hidden tail, alternate seal, or standard bridge row.*

Proof. First fix the source boundary. The root source lock of Definition 902.49 and the source-window route lock of Definition 902.50 place every accepted root read on the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source route through $Q \longmapsto B \longmapsto S \longmapsto K$.

Next classify diagnostic reads. Apply Definition 902.53; the classifier side has only the accepted displayed route or the refusal branch.

Finally attach the consumer boundary. The refusal boundary of Definition 902.52 blocks the excluded quotient, limit, host-equality, countable-choice, ambient-completeness, tail, seal, and bridge rows, while Definition 902.55 says completion-facing consumers have no coordinate outside the displayed ledger. Thus the root diagnostic surface has no third export case. □

Theorem 902.55 (Cauchy witness ledger root consumer exhaustion)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*-completion consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet* $L = (Q, B, S, K, H, C, P, N)$ *factors through*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

and the structural rows H, C, P, N*. Such a consumer has no quotient stream, host real equality, selected external limit, countable-choice row, hidden tail, alternate seal, or ambient completeness coordinate outside the displayed ledger.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root consumer exhaustion`

Proof. First apply Definition 902.12 and Definition 902.38. Ledger exhaustion confines every public read to Q, B, S, K, H, C, P, N, and Real-completion nonescape places every admitted completion read on the route $Q \longmapsto B \longmapsto S \longmapsto K$.

Next use Definition 902.29 and Definition 902.30. The StreamName window source and RegSeqRat readback are both supplied by the same finite budget source, so a downstream consumer cannot detach a stream window, rational readback, dyadic tolerance row, or hidden tail from the ledger.

Finally apply Definition 902.31. Any request outside the displayed route lands at the refusal boundary rather than at a quotient stream, host Real equality, selected external limit, countable-choice row, alternate seal, or ambient completeness coordinate. □

Corollary 902.56 (Cauchy witness ledger root source-lock strict obstruction)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. The refusal boundary of Definition 902.52, read under the source lock of Definition 902.49, is a strict local obstruction: it blocks exactly an attempted completion-seal classifier read that detaches* K *from the displayed route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

or that tries to reuse the classifier through an unlisted quotient stream, host $\mathsf{Real}^{\uparrow}$ *equality, selected external limit, countable-choice row, hidden tail, alternate seal, or ambient completeness coordinate. It does not assert a global failure of Cauchy completion, does not supply a host theorem, and does not validate any hidden external completion route.*

Proof. The root source lock fixes the source side first. By Definition 902.49, the stream, regular-rational, dyadic, and Real source surface visible to K is the same finite source already read by Q, B, and S. Hence a request that reaches K by a separate stream, rational, dyadic, or Real source is not a second accepted root read of L; it is precisely a detached classifier-source request.

Now apply the refusal boundary of Definition 902.52. That theorem lists the refused coordinates: quotient stream, host $\mathsf{Real}^{\uparrow}$ equality, selected external limit, countable-choice row, hidden tail, alternate seal, and ambient completeness coordinate, together with any tail, limit, or seal source not exposed through $Q \longmapsto B \longmapsto S \longmapsto K$. These are the only coordinates named by the displayed obstruction, so the obstruction is confined to the named converse or reuse route for the root classifier.

Finally use the consumer exhaustion statement of Definition 902.55. Every downstream completion-facing consumer factors through the displayed ledger rows and has no coordinate outside Q, B, S, K, H, C, P, N.

Therefore the refusal cannot be reinterpreted as a statement about all Cauchy completions, as a theorem of an ambient host theory, or as evidence for a hidden validation route. It only rejects off-route uses of the local source-locked witness ledger. □

Theorem 902.57 (Cauchy witness ledger root diagnostic budget totality)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every root diagnostic read of L consumes the diagonal-budget row Q, observation-budget row B, seal-synchronizer row S, completion-seal classifier row K, componentwise transport row H, displayed continuation row C, provenance row P, and local NameCert row N. No such root read can bypass the diagonal budget, observation budget, synchronizer, or completion-seal classifier.*

Proof. Begin with Definition 902.1. The carrier exposes only Q, B, S, K, H, C, P, N, so any root diagnostic read must land on the displayed ledger rows.

Apply Definition 902.2. The local NameCert surface requires carrier admission, budget exposure, synchronizer exposure, classifier exposure, transport, replay, provenance, and local naming for the same finite packet.

Now use Definition 902.5. It orders the analytic budget route through Q, then B, then S, then K. Since the root diagnostic read is a read of the accepted packet, it cannot omit any one of those budget rows while still preserving the displayed NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root diagnostic budget totality`

Theorem 902.58 (Cauchy witness ledger root seal classifier lock)**.** *For an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet $L = (Q, B, S, K, H, C, P, N)$, the completion-seal classifier row K is exposed only after the route*

$$Q \longmapsto B \longmapsto S$$

has been displayed. A consumer cannot read K as a detached classifier row before the diagonal-budget row, observation-budget row, and seal-synchronizer row have been consumed.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root seal classifier lock`

Proof. First read the carrier in Definition 902.1. The row K is one coordinate of the same finite packet as Q, B, S, H, C, P, N, not a separate classifier object.

Apply Definition 902.2 to bind K to the local certificate for the packet. The certificate exposes K together with the displayed budget and synchronizer rows.

Finally use Definition 902.5. The factorization places the classifier after $Q \longmapsto B \longmapsto S$, so a detached read of K would skip part of the certified route and would not be a root read of the accepted packet. □

Theorem 902.59 (Cauchy witness ledger root budget classifier determinacy)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ and $L' = (Q', B', S', K', H', C', P', N')$ be accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packets. Suppose their public root reads agree on the displayed budget route*

$$Q \longmapsto B \longmapsto S$$

and on the structural rows H, C, P, N. Then the completion-seal classifier read is determined on that public route: K and K' name the same route-bound classifier read. In particular, no second completion-seal classifier for the same public root route can be attached to a different stream source, $\mathsf{RegSeqRat}$ *source,* $\mathsf{DyadicRatCore}$ *source,* $\mathsf{Real}^{\uparrow}$ *source, tail source, limit source, or seal source.*

Proof. Use Definition 902.57 for both packets. Every public root diagnostic read exposes exactly the diagonal-budget row, observation-budget row, seal-synchronizer row, completion-seal classifier row, componentwise transport row, displayed continuation row, provenance row, and local NameCert row.

The assumed agreement on $Q \longmapsto B \longmapsto S$ and on H, C, P, N therefore identifies the public route on which the classifier row is read. Apply Definition 902.58: K is available only after that displayed route and cannot be read as a detached classifier before the budget and synchronizer rows have been consumed.

It remains to exclude an off-route classifier with a different completion source. Apply Definition 902.60 and Definition 902.61. Since the root packet has no quotient stream equality, host real equality, selected external limit, countable-choice row, ambient completeness coordinate, or detached classifier, every terminal completion-facing read is exhausted by the same finite ledger surface. Thus any classifier attached to the agreed public route is the route-bound classifier read already named by K and K'. □

Theorem 902.60 (Cauchy witness ledger root forbidden coordinate exclusion packet)**.** *The root packet of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *carrier* $L = (Q, B, S, K, H, C, P, N)$ *contains no quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness coordinate. All root completion-facing reads are confined to the displayed budget route, structural rows, provenance row, and local NameCert row.*

Proof. Start from Definition 902.1. Its root coordinates are exactly Q, B, S, K, H, C, P, N, and none of these is a quotient stream equality, host real equality, selected external limit, countable-choice row, or ambient completeness coordinate.

Use Definition 902.5 to locate every completion-facing read on the budget route through Q, B, S, K. The structural rows H, C, P, N preserve and name that route rather than adding a separate completion source.

Finish with Definition 902.18. That theorem rules out the forbidden coordinates for the ledger surface, and the present root packet exposes no additional row on which such a coordinate could be introduced. □

Theorem 902.61 (Cauchy witness ledger root terminal route exhaustion)**.** *Every terminal root diagnostic or completion-facing read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet* $L = (Q, B, S, K, H, C, P, N)$ *is exhausted by the displayed route* Q, B, S, K, H, C, P, N*. The classifier row* K *is available only after the route*

$$Q \longmapsto B \longmapsto S$$

has been displayed, and the structural rows H, C, P, N *transport, replay, package, and name that same route.*

No terminal read exports quotient stream equality, host real equality, a selected external limit, a countable-choice row, an ambient completeness coordinate, or a detached classifier.

Proof. Apply Definition 902.57. It places every root diagnostic read on the packet rows Q, B, S, K, H, C, P, N, with the budget and synchronizer rows present before the terminal read can complete.

Next use Definition 902.58. The classifier K is not a detached object: it is read only after $Q \longmapsto B \longmapsto S$ and remains tied to the same local packet.

Finally apply Definition 902.60. Since quotient stream equality, host real equality, selected external limits, countable choice, ambient completeness, and detached classifiers are not root coordinates, the terminal route is exactly the displayed finite ledger surface. □

Theorem 902.62 (Cauchy witness ledger root diagnostic budget transport)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ and $L' = (Q', B', S', K', H', C', P', N')$ be accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packets. Suppose the displayed budget route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

is transported componentwise through the displayed H, H' rows and through the displayed continuation rows C, C', with the history comparison governed by the stated hsame rows and the continuation comparison governed by the stated Cont *rows. Then root diagnostic reads and completion-facing terminal reads are stable along that transport: the transported route reads the same root budget, synchronizer, route-bound classifier, structural transport, continuation, provenance, and local NameCert rows.*

Moreover, this transport does not introduce quotient stream equality, host real equality, a selected external limit, a countable-choice row, an ambient completeness coordinate, or a detached classifier.

Proof. First apply Definition 902.57 to both accepted packets. The only rows visible to a root diagnostic read are the finite packet rows Q, B, S, K, H, C, P, N and $Q', B', S', K', H', C', P', N'$.

Use Definition 902.58 to read the order of the route. The classifier rows K and K' are available only after their displayed routes through $Q \longmapsto B \longmapsto S$ and $Q' \longmapsto B' \longmapsto S'$, so componentwise transport cannot detach the classifier from the budget and synchronizer rows.

Now apply Definition 902.59. Since the transport is through the displayed H, H' and C, C' rows, with the comparison confined to the stated *hsame* and Cont rows, the classifier read on the transported public route is the route-bound read already named by K and K'.

It remains to identify the terminal surface. Apply Definition 902.60 and Definition 902.61. The terminal completion-facing read is exhausted by the same finite ledger surface, and none of the forbidden coordinates can be supplied by transport through H, H' or C, C'. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root diagnostic budget transport`

Corollary 902.63 (Cauchy witness ledger root diagnostic budget use determinacy)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, and let t be an accepted consumer-facing root diagnostic budget use of L. Once the displayed rows Q, B, S, H, C, P, N are fixed, the use t has only one accepted landing: it factors through the finite route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

with route replay supplied by C, componentwise transport supplied by H, provenance supplied by P, and local naming supplied by N. Hence any other accepted consumer presentation over the same fixed rows chooses the same route-bound completion-seal classifier K, the same replay normal form through C, and the same exactness/readback decision named by the chapter. No alternate hidden coordinate, detached tail, selected external limit, duplicate classifier source, quotient stream equality, host real equality, countable-choice row, or ambient completeness coordinate can be used to produce a second accepted diagnostic budget use.

Proof. Apply Definition 902.57 to the accepted use t. Every root diagnostic read of L consumes exactly the diagonal-budget row Q, observation-budget row B, seal-synchronizer row S, completion-seal classifier row K, componentwise transport row H, displayed continuation row C, provenance row P, and local NameCert row N. In particular, after Q, B, S, H, C, P, N are fixed, the only remaining classifier-facing row visible to t is the route-bound row K.

Now compare t with any second accepted consumer presentation over the same fixed local rows. The hypotheses give agreement on the displayed budget route

$$Q \longmapsto B \longmapsto S$$

and on the structural rows H, C, P, N. By Definition 902.59, the completion-seal classifier read is determined on that public route: the second presentation cannot attach a different stream source, $\mathsf{RegSeqRat}$ source, $\mathsf{DyadicRatCore}$ source, $\mathsf{Real}^{\uparrow}$ source, tail source, limit source, or seal source to the same diagnostic budget use.

The replay and exactness components are also fixed. By Definition 902.17, the displayed Cont row C replays the route through $Q \longmapsto B \longmapsto S \longmapsto K$ before any $\mathsf{Real}^{\uparrow}$-facing read is admitted. By Definition 902.25, the classifier readback for K is public exactly through that displayed route with H, C, P, N. Thus an accepted presentation over the fixed rows has the same replay normal form and the same exactness/readback decision.

Finally use Definition 902.61 together with Definition 902.60. The terminal diagnostic surface is exhausted by the displayed finite ledger rows, and the forbidden external coordinates are not root coordinates. Therefore no hidden coordinate or duplicate classifier source remains from which a second accepted use of t could be produced. □

Theorem 902.64 (Cauchy witness ledger completion consumer factorization)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every completion-facing consumer of L factors through the displayed sibling route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and through the internal ledger route

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

The structural rows H, C, P, N preserve the same two routes as componentwise transport, displayed continuation, package provenance, and local naming data; no consumer reaches a completion-facing read through a detached dyadic core, a second stream name, an unregistered regular rational readback, or an external Real completion selector.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier completion consumer factorization`

Proof. Start with Definition 902.37. It places the $\mathsf{StreamName}^{\uparrow}$ window source and $\mathsf{RegSeqRat}^{\uparrow}$ readback on the same $Q \longmapsto B$ budget source, with $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ already named in the shared source surface.

Next apply Definition 902.38. Completion-facing reads reach K only through $Q \longmapsto B \longmapsto S \longmapsto K$ and the structural rows H, C, P, N.

Finally use Definition 902.18. Since the carrier has no detached dyadic, stream, regular-rational, or external Real-completion coordinate, the sibling route and the internal ledger route are the only public factorization surfaces available to completion consumers. □

Theorem 902.65 (Cauchy witness ledger regular Real dependency scope)**.** *The exact BEDC dependency scope of the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *regular-Real route is*

BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$,

Within this scope, $L = (Q, B, S, K, H, C, P, N)$ exports only the finite budget, stream-name, regular-rational, Real-seal, synchronizer, classifier, transport, continuation, provenance, and naming rows. It excludes host equality, quotient streams, selected limits, countable choice, and ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier regular real dependency scope`

Proof. Project the carrier in Definition 902.1. Its analytic rows name the four packet dependencies $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, and $\mathsf{RealCompletionSealClassifier}$ while its structural rows name BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Use Definition 902.7 and Definition 902.37. These identify the regular-Real source surface as the shared $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route read through Q and B before S and K are exposed.

Apply Definition 902.18. The excluded objects have no carrier coordinate, no transport row, no continuation replay, no provenance row, and no local NameCert entry, so they are outside the exact dependency scope. □

Theorem 902.66 (Cauchy witness ledger root-unblock obligation package)**.** *For downstream root consumers, the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *obligation package is the union of the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$ *obligations of Definition 902.2, the completion-consumer factorization of Definition 902.64, and the regular-Real dependency scope of Definition 902.65. Thus any root consumer may rely on exactly the displayed rows Q, B, S, K, H, C, P, N, the sibling route* $\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$*, and the named BEDC kernel rows* $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$*; it may not import a host equality, quotient-stream, selected-limit, choice, or ambient-completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerUp StdBridge`
↪ variant `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root namecert threshold`

Proof. First read the local certificate surface from Definition 902.2. It admits exactly Q, B, S, K, H, C, P, N as the public ledger rows.

Next apply Definition 902.64. This supplies the downstream completion-facing route through both the sibling chain and the internal ledger chain.

Finally combine that route with Definition 902.65. The dependency theorem names every allowed BEDC object and excludes the off-scope completion principles, so the resulting package is precisely the root-unblock surface available to downstream consumers. □

Theorem 902.67 (Cauchy witness ledger terminal L10 pullback)**.** *For any downstream root consumer of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet $L = (Q, B, S, K, H, C, P, N)$, the terminal L10 surface is the single finite pullback*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the internal ledger route

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

Every read of the root surface factors through these two displayed chains and the structural rows H, C, P, N; no consumer receives a second stream-name branch, an alternate regular-rational readback, or a detached Real completion endpoint.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root streamname pullback`

Proof. Begin with Definition 902.64. It already places every completion-facing consumer on the sibling chain from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, and on the internal ledger route $Q \longmapsto B \longmapsto S \longmapsto K$.

Use Definition 902.65. This identifies the exact BEDC dependency scope of the regular-Real route and keeps the dyadic core, stream name, regular rational readback, and Real endpoint attached to the same finite carrier surface.

Finally apply Definition 902.66. The root package admits only the displayed ledger rows and the named sibling route, so the two chains form the terminal L10 pullback available to root consumers. □

Theorem 902.68 (Cauchy witness ledger root completion nonescape)**.** *The terminal pullback of Definition 902.67 exports no quotient stream, selected limit, host equality, choice row, ambient completeness principle, metric-completion datum, or standard bridge row. Its public data are exactly the finite sibling route* $\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$*, the ledger route* $Q \longmapsto B \longmapsto S \longmapsto K$*, and the structural rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier root real completion nonescape`

Proof. Read the allowed data from Definition 902.67. It exposes only the sibling chain, the internal ledger chain, and the structural rows.

The regular-Real dependency scope of Definition 902.65 excludes host equality, quotient streams, selected limits, countable choice, and ambient completeness from the dependency surface.

The root-unblock obligation package of Definition 902.66 supplies the same exclusion to downstream root consumers. Since no metric-completion datum or standard bridge row is one of Q, B, S, K, H, C, P, N or one of the four sibling objects, neither is exported by the terminal pullback. □

Theorem 902.69 (Cauchy witness ledger root seal-selector exhaustion)**.** *Every root seal-selector read of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet factors through the two displayed routes*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

and

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The read may use the structural rows H, C, P, N*, but it has no separate selector row, no hidden tail selector, no detached* $\mathsf{Real}^{\uparrow}$ *endpoint, and no off-route completion coordinate.*

Proof. Start with the root pullback of Definition 902.67. It fixes the sibling route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the internal ledger route $Q \longmapsto B \longmapsto S \longmapsto K$.

Apply root completion nonescape from Definition 902.68. A selector-facing read cannot leave those displayed rows to obtain a quotient stream, selected limit, host equality, choice row, ambient completeness principle, metric-completion datum, or standard bridge row.

Finally use the root budget refusal boundary of Definition 902.52. Any requested seal selector not carried by the displayed routes lands at the refusal boundary, so the two chains exhaust every admitted root seal-selector read. □

Theorem 902.70 (Cauchy witness ledger completion observer refusal factorization)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every observer-filter or selector-seal consumer of* L *either reads the displayed ledger route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

together with the sibling route $\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$*, or lands in the refusal boundary for off-route completion requests. No third observer channel is exported.*

Proof. First read the admitted route from Definition 902.69. It gives the only seal-selector path available to a root consumer.

Next apply terminal observer exactness, Definition 902.73. The observer side is forced onto the same four terminal faces and the same structural support rows, so an observer-filter consumer has no independent completion face to inspect.

For requests outside those rows, use Definition 902.52. The refusal boundary receives the off-route quotient, selected-limit, host-equality, countable-choice, ambient-completeness, hidden-tail, alternate-seal, and bridge-shaped requests. Thus observer-filter and selector-seal consumers factor through the displayed route or through refusal, with no third case. □

Theorem 902.71 (Cauchy witness ledger terminal four-face refusal exactness)**.** *Once the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *faces of an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet are fixed, the terminal consumer surface contains exactly those faces, the ledger route* $Q \longmapsto B \longmapsto S \longmapsto K$*, and the structural rows* H, C, P, N*. No quotient stream, selected limit, host equality, countable-choice row, ambient completeness principle, metric-completion datum, or standard bridge row remains.*

Proof. Begin with terminal selector-observer uniqueness, Definition 902.76. Agreement on the four terminal faces and the support rows identifies the ledger route, observer-filter route, and selector-seal route as the same finite selector-observer route.

Use terminal observer exactness, Definition 902.73, to remove residual observer data. The observer can read only the fixed four faces and the displayed support rows.

Finally combine root completion nonescape and completion observer refusal factorization, Definition 902.68 and Definition 902.70. Off-route completion requests are refused, while admitted requests use the displayed routes. Hence none of the quotient, limit, host-equality, choice, ambient-completeness, metric-completion, or bridge rows remains at the terminal surface. □

Corollary 902.72 (Cauchy witness ledger completion consumer strict local obstruction)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Any displayed failure, refusal, separation, or countermodel row attached to the completion-consumer surface of* L *is a strict local obstruction. It blocks only a proposed converse read, detached reuse, second stream-name route, unregistered regular-rational readback, external* $\mathsf{Real}^{\uparrow}$ *completion selector, or bridge-shaped completion request that tries to bypass the displayed sibling route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and the internal ledger route

$$Q \longmapsto B \longmapsto S \longmapsto K.$$

Every admitted completion-facing consumer still factors through those two routes together with the structural rows H, C, P, N*. A request using an unlisted coordinate cannot be reinterpreted as a*

global impossibility theorem for $\mathsf{Real}^{\uparrow}$ *completion, as a host theorem about Real completeness, or as hidden validation by an external completion principle.*

Proof. Begin with Definition 902.64. It states that every completion-facing consumer of L factors through the sibling route from $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ and through the internal route $Q \longmapsto B \longmapsto S \longmapsto K$. The same theorem also fixes the role of H, C, P, N: they preserve those routes as componentwise transport, displayed continuation, package provenance, and local naming data, and they do not create a detached dyadic core, a second stream name, an unregistered regular-rational readback, or an external Real completion selector.

Now apply Definition 902.68. The terminal pullback exports no quotient stream, selected limit, host equality, choice row, ambient completeness principle, metric-completion datum, or standard bridge row. Its public data are exactly the four-object sibling route, the ledger route $Q \longmapsto B \longmapsto S \longmapsto K$, and the structural rows H, C, P, N.

These two statements leave only two consumer-facing cases. In the admitted case, the consumer factors through the displayed routes and the structural rows by completion-consumer factorization. In the refused case, the attempted coordinate is one of the unlisted completion, quotient, host, choice, bridge, or detached readback coordinates excluded by root completion nonescape. Such a coordinate has no exported carrier row, no structural replay through H, C, P, N, and no terminal pullback datum from which a host theorem or hidden external validation route could be read. Hence the displayed obstruction is strict and local: it rejects only the named converse or reuse attempt outside the ledger packet, while leaving the admitted completion-consumer factorization intact. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyWitnessLedger}^{\uparrow}$ packet is generated from a Cauchy diagonal budget, Real observation budget, Cauchy seal-budget synchronizer, Real completion-seal classifier, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 902.1, Definition 902.2, Definition 902.5, Definition 902.9, Definition 902.10, Definition 902.11, Definition 902.12, Definition 902.16, Definition 902.17, Definition 902.18, Definition 902.19, Definition 902.21, Definition 902.22, Definition 902.23, Definition 902.24, Definition 902.25, Definition 902.26, Definition 902.27, Definition 902.28, Definition 902.36, Definition 902.37, Definition 902.38, Definition 902.64, Definition 902.65, and Definition 902.66 bind the finite witness ledger for the Real completeness route; the mature paper surface covers the carrier, budget factorization, Real-completion handoff, carrier admission, route determinacy, ledger exhaustion, componentwise transport, replay exactness, forbidden-coordinate exclusion, public export, budget-seal lock, Real consumer nonescape, downstream unblock packaging, window-seal consumer triad, classifier readback, downstream totality, root completion packaging, seal readback, budget-Real route totality, StreamName-RegSeq factorization, Real completion nonescape, completion consumer factorization, regular-Real dependency scope, and root-unblock obligation packaging.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyWitnessLedger}^{\uparrow}}$, bridgeChecked)

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.taste gate`

Lean target: `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerUp\_StdBridge`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient streams, host real equality, selected external limits, countable choice, ambient completeness, axiom-purity, or bridge checking.

Upgrade path: No further formal-axis upgrade remains for the finite witness-ledger standard bridge.

902.1 Cauchy witness ledger terminal observer exactness

Theorem 902.73 (Cauchy witness ledger terminal observer exactness)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Any observer that reads the same* $\mathsf{StreamName}^{\uparrow}$

window face, $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face,* $\mathsf{Real}^{\uparrow}$ *seal face, and displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows determines the same terminal Real-completion observer surface.*

Consequently, the observer has no residual quotient stream, selected limit, host sequence equality, ambient Real completeness, metric-completeness datum, countable-choice row, second completion route, or standard bridge row to read after the four faces and support rows have been fixed.

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier terminal observer exactness`

Proof. Begin with Definition 902.46. It places every Phase Real and Completion consumer of L on the selector-seal route ending at $\mathsf{Real}^{\uparrow}$, with only the displayed BHist, hsame, Cont, Pkg, and NameCert rows as support.

Next use Definition 902.45 and Definition 902.44. The handoff supplies the shared dyadic budget, named stream window, regular-rational readback, and Real seal coordinates, while the selector exhaustion shows that an observation-budget mediator exports the same four terminal faces and no off-route completion coordinate.

Finally apply Definition 126.449. Once two observers agree on the $\mathsf{StreamName}^{\uparrow}$ window face, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, $\mathsf{RegSeqRat}^{\uparrow}$ readback face, $\mathsf{Real}^{\uparrow}$ seal face, and the displayed support rows, the open Phase Real and Completion surface is terminal. The exclusions in the cited Cauchy ledger route and StreamName terminality theorem leave no quotient stream, selected limit, host sequence equality, ambient Real completeness, metric-completeness datum, countable-choice row, second completion route, or standard bridge row outside that surface. □

Theorem 902.74 (Cauchy witness ledger observer-filter budget-selector lock)**.** *Let* $L = (Q, B, S, K, H, C, P, N)$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, and let* $G = (F, I, S_{\mathrm{obs}}, O, B_{\mathrm{obs}}, W, D, R, E, H_{\mathrm{obs}}, C_{\mathrm{obs}}, P_{\mathrm{obs}}, N_{\mathrm{obs}})$ *be an accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet. Suppose the selected* $\mathsf{StreamName}^{\uparrow}$ *window family* W *of* G *agrees with the terminal* $\mathsf{StreamName}^{\uparrow}$ *window face of* L*, together with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then every Real-completion observer whose finite observations are selected by* G *factors through the single route*

$$\mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to$$

The omitted rows of G *and every off-route completion datum remain outside this factorization.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier observer filter budget selector lock`

Proof. Begin with Definition 902.73. The agreement hypothesis identifies the terminal Cauchy ledger window face with the selected $\mathsf{StreamName}^{\uparrow}$ window family exported by G, and it fixes the same support rows for the dyadic ledger, regular-rational readback, and Real seal.

Next apply Definition 1042.3. It forces every finite observation consumed by G through the observer-history identity row, the observer filter, the selected row, the finite budget selector, and then the $\mathsf{StreamName}^{\uparrow}$ window family before any dyadic, regular-rational, or Real-completion row is exposed.

The NameCert surface of Definition 1042.2 supplies the boundary check for the omitted ledger. Its coordinates place omitted rows only in the boundary and support rows, while W, D, R, E are the sole observation and completion coordinates available on the selected route.

Coordinate exhaustion now gives the claimed lock. Since the Cauchy terminal observer has no residual quotient stream, host real equality, selected limit, countable-choice row, ambient completeness, second completion route, metric-completion datum, or standard bridge row to read,

and since the observer-filter budget selector has no coordinate that can turn an omitted row into W, D, R, or E, every Real-completion observer selected by G factors through the displayed route and nowhere else. □

Theorem 902.75 (Cauchy witness ledger observer-filter omitted-row boundary)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, and let*

$$G = (F, I, S_{\text{obs}}, O, B_{\text{obs}}, W, D, R, E, H_{\text{obs}}, C_{\text{obs}}, P_{\text{obs}}, N_{\text{obs}})$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet. Suppose G selects the same* $\mathsf{StreamName}^{\uparrow}$ *window family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row as the terminal Cauchy ledger route of L, with the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows identified. Then every Real-and-Completion observer factors through the selected route*

$$\mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to$$

and the omitted ledger O of G remains boundary-only data. No omitted row can be repacked as a $\mathsf{StreamName}^{\uparrow}$ *window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream, selected limit, host equality, choice row, ambient completeness principle, or second completion route.*

Proof. First use terminal observer exactness, Definition 902.73. The shared terminal faces force every Real-and-Completion observer of L onto the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface, with no residual quotient stream, selected limit, host equality, choice row, ambient completeness principle, metric-completion datum, or second completion route.

Next apply the observer-filter budget-selector lock, Definition 902.74. The agreement with G places the same observer on the route through $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{ObserverFilterBudgetSelector}$ and then the selected W, D, R, E coordinates.

Finally invoke omitted-row exclusion for the observer-filter budget selector, Definition 1042.4. It keeps O outside the selected StreamName window, dyadic ledger, regular-rational readback, and Real seal coordinates. Combining this boundary exclusion with the terminal Cauchy exhaustion leaves no carrier position where an omitted row could be repacked as any of the off-route completion objects listed in the statement. □

Theorem 902.76 (Cauchy witness ledger terminal selector-observer uniqueness)**.** *Let $L_{\text{cw}} = (Q, B_{\text{cw}}, S, K, H_{\text{cw}}, C_{\text{cw}}, P_{\text{cw}}, N_{\text{cw}})$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, let*

$$G = (F, I, S_{\text{obs}}, O, B_{\text{obs}}, W, D, R_{\text{obs}}, E_{\text{obs}}, H_{\text{obs}}, C_{\text{obs}}, P_{\text{obs}}, N_{\text{obs}})$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet, and let*

$$T = (B, W, R, L, E, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose these three packets agree on the selected* $\mathsf{StreamName}^{\uparrow}$ *window family, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the* $\mathsf{Real}^{\uparrow}$ *seal face, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then every terminal Real-completion observer selected through the ledger route of L_{cw} and every terminal Real-completion observer selected through the observer-filter route of G is the same finite selector-observer route through T.*

The route has no quotient stream coordinate, selected limit object, host equality row, ambient completeness principle, metric-completeness datum, countable-choice row, second completion route, or standard bridge row.

Proof. First apply Definition 902.73. The shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces identify every terminal observer read from L_{cw} with the displayed Cauchy witness ledger surface and its support rows.

Next apply Definition 902.74. The observer-filter packet G sends every selected finite observation through the observer-history identity, observer filter, budget selector, selected stream window, dyadic ledger face, regular-rational readback, and Real seal. Thus the observer-filter route reaches the same four terminal faces fixed by the ledger route.

Now use Definition 1081.3. Since the accepted selector-seal packet T displays the same selector budget, selected window, readback row, limit-seal handoff, Real endpoint, and structural rows, any completion consumer exposed by either upstream route determines the same finite Real and Completion handoff route.

Finally apply Definition 1081.4. Non-escape removes the remaining possible bypasses: quotient streams, selected or detached limits, host equality, ambient completeness, metric-completeness data, countable choice, second completion routes, and standard bridge rows are not carrier coordinates of the shared packet.

Coordinate exhaustion over the shared finite faces therefore leaves only one selector-observer route. It is simultaneously the terminal observer route read from L_{cw}, the selected observer-filter route read from G, and the selector-seal route read from T. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier terminal selector observer uniqueness`

Theorem 902.77 (Cauchy witness ledger terminal selector-observer support exhaustion)**.** *Let* $L_{\mathrm{cw}} = (Q, B_{\mathrm{cw}}, S, K, H_{\mathrm{cw}}, C_{\mathrm{cw}}, P_{\mathrm{cw}}, N_{\mathrm{cw}})$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, let*

$$G = (F, I, S_{\mathrm{obs}}, O, B_{\mathrm{obs}}, W, D, R_{\mathrm{obs}}, E_{\mathrm{obs}}, H_{\mathrm{obs}}, C_{\mathrm{obs}}, P_{\mathrm{obs}}, N_{\mathrm{obs}})$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet, and let*

$$T = (B, W, R, L, E, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose these three packets agree on the selected* $\mathsf{StreamName}^{\uparrow}$ *window family, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the* $\mathsf{Real}^{\uparrow}$ *seal face, and the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *support rows. Then every terminal selector-observer read exposed by these packets is exhausted by exactly those four shared faces and the displayed support rows.*

No quotient stream, selected limit, host equality, ambient completeness principle, metric-completeness datum, countable-choice row, second completion route, standard bridge row, or extra support row remains after the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *inventory is fixed.*

Proof. Apply Definition 902.73. It reduces every terminal observer read from the accepted Cauchy witness ledger packet to the four displayed terminal faces together with the BHist, hsame, Cont, Pkg, and NameCert rows.

Next apply Definition 902.74. The observer-filter budget selector cannot expose an omitted observation row as a completion coordinate; every selected finite observation must pass through the shared stream window, dyadic ledger, regular-rational readback, and Real seal.

Then use Definition 902.76. The ledger route, observer-filter route, and selector-seal route determine the same finite selector-observer route, so their support inventories cannot split into competing terminal reads.

Finally apply Definition 1081.4. Non-escape removes the possible residual rows named in the statement from the accepted selector-seal carrier. The only remaining readable surface is therefore the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces with the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support rows. □

Corollary 902.78 (Cauchy witness ledger terminal observer exactness determinacy)**.** *Let* $L_{\mathrm{cw}} = (Q, B_{\mathrm{cw}}, S, K, H_{\mathrm{cw}}, C_{\mathrm{cw}}, P_{\mathrm{cw}}, N_{\mathrm{cw}})$ *be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet, let*

$$G = (F, I, S_{\mathrm{obs}}, O, B_{\mathrm{obs}}, W, D, R_{\mathrm{obs}}, E_{\mathrm{obs}}, H_{\mathrm{obs}}, C_{\mathrm{obs}}, P_{\mathrm{obs}}, N_{\mathrm{obs}})$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet, and let*

$$T = (B, W, R, L, E, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Suppose the three packets agree on the selected* $\mathsf{StreamName}^{\uparrow}$ *window family, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, the* $\mathsf{Real}^{\uparrow}$ *seal face, and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support rows. Then any accepted consumer-facing use of the terminal observer exactness surface has a determined finite read: it must choose the route, normal form, and exactness decision read from the single chain*

$$\mathsf{CauchyWitnessLedger}^{\uparrow} \to \mathsf{ObserverHistoryIdentity}^{\uparrow} \to \mathsf{ObserverFilter}^{\uparrow} \to \mathsf{ObserverFilterBudgetSelector}^{\uparrow} \to \mathsf{RealCompl}$$

No accepted use can choose an alternate hidden coordinate, omitted-row classifier, duplicate classifier source, quotient stream, selected limit, host equality row, ambient completeness principle, metric-completeness datum, countable-choice row, second completion route, or standard bridge row after those local rows have been fixed.

Proof. Apply Definition 902.73. The shared $\mathsf{StreamName}^{\uparrow}$ window face, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, $\mathsf{RegSeqRat}^{\uparrow}$ readback face, $\mathsf{Real}^{\uparrow}$ seal face, and displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows determine one terminal Real-completion observer surface for the accepted Cauchy witness ledger packet. Thus an accepted consumer-facing use of terminal observer exactness has no remaining Real-completion coordinate outside that surface from which to read a different exactness decision.

Now apply Definition 902.76. The ledger route, the observer-filter route, and the selector-seal route determine the same finite selector-observer route through T. In particular, the observer-filter part of the read passes through $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$, the selected $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger face, the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, and the $\mathsf{Real}^{\uparrow}$ seal face; the selector-seal part reads the same selector budget, selected window, readback row, seal handoff, Real endpoint, and structural rows.

The two cited results exhaust both possible sources of variation. Exactness fixes the terminal observer surface once the four faces and support rows are fixed, while uniqueness, with the componentwise transport reference Definition 902.16, identifies every selected terminal observer with the same finite route through the displayed packets. Hence the consumer cannot replace the route by an omitted-row classifier, a second NameCert source, or any of the excluded completion coordinates named in the statement. The route, its normal form as a finite read, and the exactness decision exposed to the consumer are therefore determined by the displayed local rows alone. □

902.2 Cauchy witness ledger classifier route coverage

Theorem 902.79 (Cauchy witness ledger classifier route coverage)**.** *Let $L = (Q, B, S, K, H, C, P, N)$ be an accepted* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *packet. Every public consumer of the completion-seal classifier row K factors through the displayed Real-facing route*

$$Q \longmapsto B \longmapsto S \longmapsto K$$

together with the structural rows H, C, P, N. In particular, no classifier consumer can replace that route by quotient stream equality, host $\mathsf{Real}^{\uparrow}$ *equality, selected external limits, countable choice, ambient completeness, hidden tails, or an alternate* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessLedgerUp.CauchyWitnessLedgerCarrier classifier route coverage`

Proof. Project the accepted packet by Definition 902.1. Its public coordinates are exactly the diagonal budget Q, observation budget B, synchronizer S, completion-seal classifier K, and structural rows H, C, P, N.

Apply Definition 902.2 and Definition 902.5. The NameCert surface certifies the displayed rows, while the budget factorization orders the Real-facing route as $Q \longmapsto B \longmapsto S \longmapsto K$ before any classifier read is available to a public consumer.

Next use Definition 902.7 and Definition 902.14. The source coherence identifies the finite stream, dyadic, regular-rational, and Real faces consumed by Q and B; the synchronizer-classifier lock carries that same source through S into K without adding a second classifier route.

It remains to connect this finite route to the Real-facing seal. Apply Definition 902.11 and Definition 902.9. Once the route and structural rows are fixed, the completion-seal classifier read is determined, and every $\mathsf{Real}^{\uparrow}$-facing handoff reaches K only through the displayed budget, observation, and synchronizer rows.

Finally apply Definition 902.18. The excluded objects have no carrier coordinate, no NameCert entry, and no structural row in H, C, P, N. A public consumer of K therefore cannot replace the displayed route by quotient stream equality, host $\mathsf{Real}^{\uparrow}$ equality, selected external limits, countable choice, ambient completeness, hidden tails, or an alternate $\mathsf{Real}^{\uparrow}$ seal. □

The $\mathsf{CauchyWitnessLedger}^{\uparrow}$ row is the witness-ledger surface recovered by chapter 1117 and consumed by chapter 1110 inside the Real-completion route. In that lattice position the ledger remains the finite BHist, hsame, Cont, Pkg, and NameCert surface over the displayed budget, observation, synchronizer, and classifier rows; it supplies no quotient stream equality, choice principle, or ambient completeness theorem.

Chapter 903

A Concrete Naming Certificate for $\mathsf{ObservationBudgetLimiter}^{\uparrow}$

The $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet realizes the finite request-cap side of the Real and Completion route of chapter 9238. It sits between the observation-budget service of chapter 611 and the seal-facing routes of chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a concrete gate: it records which finite request rows may be served before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ seal read is consumed.

Falsifiable prediction: Deleting the budget-cap row permits a RealUp seal consumer to bypass the finite request bound, while deleting the window or dyadic rows breaks StreamName and RegSeqRat readback.

Independence witness: Nearest siblings: realobservationbudget, cauchysealbudgetsynchronizer, realcompletionsealclassifier, streamname, and regseqrat; none is bijective because this carrier has a separate budget-cap gate between request and seal rows.

Definition 903.1 (Observation-budget limiter carrier)**.** *An* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, B, K, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the incoming finite precision or observation request;*
2. *B is the accepted* $\mathsf{RealObservationBudget}^{\uparrow}$ *service row covering the request;*
3. *K is the budget-cap row that bounds how many request windows may be served before the seal is read;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger admitted under the cap;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger read through those windows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from W and D;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the capped request, window, dyadic, and regular readback rows have been consumed;*
8. *H stores componentwise* hsame *transports for $E, B, K, W, D, R, S, C, P, N$;*

9. *C stores the displayed* Cont *routes from the request to the budget row, from the budget-cap row to the window ledger, from the window and dyadic ledgers to the regular readback, and from the readback to the seal;*
10. *P records* Pkg *provenance over* $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *row.*

The carrier exports a finite budget gate for Real and Completion consumers. It exports no unbounded observation schedule, quotient stream, selected limit, host real equality, trichotomy row, independent real carrier, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiterTasteGate single carrier alignment`

Theorem 903.2 (Observation-budget limiter NameCert obligations)**.** *For the carrier of Definition 903.1, the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for the displayed packet $L = (E, B, K, W, D, R, S, H, C, P, N)$;*
2. *budget-cap admission, requiring K to be read after the request row E and budget row B and before any window row W is served;*
3. *finite-window coverage, requiring every served window to be one of the* $\mathsf{StreamName}^{\uparrow}$ *rows admitted by W under K;*
4. *dyadic and regular readback, requiring each* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation in D and the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be read through the same capped window route;*
5. *seal handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *seal row S to be consumed only after E, B, K, W, D, R have been displayed;*
6. *non-escape, excluding unbounded schedules, quotient streams, selected limits, host real equality, trichotomy, independent real carriers, and ambient completeness.*

Proof. Carrier admission is projection from Definition 903.1. The request, observation budget, cap, window, dyadic, regular readback, seal, transport, route, provenance, and local name are all explicit coordinates.

Budget-cap admission reads K between B and W. Thus the finite request is served by the displayed budget row before any window ledger is exposed, and the cap row fixes which subsequent windows are available.

Finite-window coverage, dyadic readback, and regular readback are then read along the displayed route $K \to W \to D \to R$. Componentwise transports in H preserve the same rows, and the continuation rows in C replay them without adding another observation source.

Finally the seal handoff reads S only after R. Since the carrier contains no row for an unbounded schedule, quotient stream, selected limit, host real equality, trichotomy, independent real carrier, or ambient completeness principle, the non-escape obligation is exactly the visible finite inventory. □

Theorem 903.3 (Observation-budget limiter cap admission)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier. Every admitted read of the finite window row W, dyadic ledger row D, regular readback row R, or real seal row S first reads the incoming request row E, then the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row B, and then the budget-cap row K. The cap admission row is therefore a carrier projection certified by the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *obligations, not a numerical side condition external to the packet.*

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiterCarrier cap admission`

Proof. Project the accepted carrier L and apply Definition 903.2. Carrier admission exposes the rows $E, B, K, W, D, R, S, H, C, P, N$, while budget-cap admission places K after E and B and before any served window row.

The finite-window, dyadic readback, regular readback, and seal handoff obligations all depend on that displayed cap row. Hence any consumer attempting to read W, D, R, or S without first reading K has no carrier coordinate and no NameCert clause from which such a route can be certified.

The non-escape clause excludes replacing this route by an unbounded schedule, quotient stream, selected limit, host equality, trichotomy row, independent real carrier, or ambient completeness witness. Thus cap admission is exactly the visible carrier order E, B, K before the downstream surface. □

Theorem 903.4 (Observation-budget limiter seal ordering)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* ObservationBudgetLimiter$^{\uparrow}$ *carrier. A seal-facing* Real$^{\uparrow}$ *consumer reaches S only through the ordered finite route*

$$E \to B \to K \to W \to D \to R \to S.$$

The displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows certify this order and certify no route that reads S before the budget cap, window ledger, dyadic ledger, and regular readback have been displayed.*

Proof. Use Definition 903.2. The seal handoff obligation permits the Real$^{\uparrow}$ seal row only after the request, observation-budget, cap, window, dyadic, and regular readback rows are visible.

The continuation rows in C display the same ordered route, and the componentwise transports in H preserve the chosen rows rather than adding a second seal route. The provenance row P and local naming row N authenticate this finite route as the packet's seal-facing surface.

Finally apply the non-escape obligation. Since L contains no independent real carrier, selected limit, host equality, quotient stream, trichotomy row, or completeness witness, there is no carrier projection from which an out-of-order seal read can be formed. □

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiter seal ordering`

Theorem 903.5 (Observation-budget limiter capped-window transport)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ and $L' = (E', B', K', W', D', R', S', H', C', P', N')$ be accepted* ObservationBudgetLimiter$^{\uparrow}$ *carriers whose request, budget, cap, window, dyadic, regular, and seal rows are related by the displayed componentwise* hsame *rows. If both packets read downstream consumers through the budget row and cap row before the window, dyadic, regular, and seal rows, then the transported consumer route remains inside the capped finite surface*

$$E, B, K, W, D, R, S, H, C, P, N.$$

It cannot escape to an uncapped window schedule, quotient stream, selected limit, external equality, independent real carrier, or ambient completeness witness.

Proof. Begin with carrier projection for L and L'. The local obligations read E through B, then K, before any of W, D, R, S becomes available.

Use the componentwise hsame rows to transport each displayed coordinate to the matching coordinate of L'. The continuation rows replay the capped route through $K' \to W' \to D' \to$

$R' \to S'$, so transport preserves the finite window service rather than introducing a new consumer source.

The non-escape obligation applies on both carriers. Because neither packet contains an uncapped schedule, quotient stream, selected limit, external equality, independent real carrier, or completeness row, the transported route remains exactly a capped-window transport of the displayed carrier surface. □

Theorem 903.6 (Observation-budget limiter budget monotonicity)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ and $L' = (E', B', K', W', D', R', S', H', C', P', N')$ be accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carriers. If the request and budget rows agree componentwise and the budget-cap row K' admits every window admitted by K, then every seal-facing read admitted by L is also admitted by L' through the transported rows W', D', R', S' and the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *evidence.*

Proof. Start with the carrier rows of L. A seal-facing read must pass through E, B, K, W, D, R, S in that order by Definition 903.2.

Use the componentwise agreement of the request and budget rows to transport the entry route from E, B to E', B'. The cap hypothesis then sends each K-admitted window to a K'-admitted window, so the same finite window service is available in W'.

Read the dyadic and regular rows through the transported windows. The rows D', R' are the matching dyadic and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows under the larger cap, and the route to S' is displayed by C' with provenance P', N'. Thus widening the cap preserves every previously admitted seal read and does not add any excluded analytic object. □

Theorem 903.7 (Observation-budget limiter window non-escape)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer served by an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier factors through the finite rows*

$$E, B, K, W, D, R, S, H, C, P, N.$$

In particular, no consumer may reach the regular readback row R or the real seal row S without first displaying the budget-cap row K, the finite window row W, and the dyadic observation row D.

Proof. Project the accepted carrier and apply Definition 903.2. The obligations order the request, budget, cap, window, dyadic, regular, and seal rows before any consumer-facing read is available.

For a $\mathsf{StreamName}^{\uparrow}$ consumer, the route stops at the capped window ledger W. For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, the route continues through D to R. For a $\mathsf{Real}^{\uparrow}$ consumer, the route continues through the same readback to S.

The displayed transport, continuation, provenance, and naming rows authenticate these routes and no others. Therefore any route that bypasses K, W, or D has no carrier coordinate from which to read R or S. □

Theorem 903.8 (Observation-budget limiter vision handoff)**.** *The Real and Completion route of chapter 9238 may consume an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet as the finite gate between observation-budget availability and seal consumption. The handoff first reads the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row, then the budget cap, then the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and finally the* $\mathsf{Real}^{\uparrow}$ *seal. The resulting route is compatible with chapter 886 and chapter 900 as a capped observation service, not as a completeness principle.*

Proof. Begin with the observation-budget row B and the cap row K. These rows turn the availability of an observation service into a finite request gate before any stream or seal consumer is reached.

Next compose the capped window route with the dyadic and regular readback rows. The route W, D, R is the same finite surface used by StreamName, DyadicRatCore, and RegSeqRat consumers, so it gives the Real and Completion side a displayed readback rather than an implicit analytic primitive.

Finally read the seal row S through the same packet. The synchronizer and completion-seal classifier chapters may consume this packet as a capped observation service before they read their seal-facing rows. Since the carrier contains no completeness, quotient, selected-limit, host-equality, trichotomy, or independent-real coordinate, the handoff is a finite gate and not an analytic closure theorem. □

Theorem 903.9 (Observation-budget limiter cap totality)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier. Every accepted packet reads the finite request row E and the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row B before the budget-cap row K. No consumer route to the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, or the* $\mathsf{Real}^{\uparrow}$ *seal row S is available without first displaying K with the same* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Apply Definition 903.2 to the displayed packet L. The budget-cap admission obligation places K after E and B and before any served window row.

Apply Definition 903.7. It identifies K as part of every route that reaches W, D, R, S, so a consumer missing the cap row has no carrier coordinate from which to read any downstream seal-facing row.

The monotonicity theorem Definition 903.6 shows that enlarging the cap changes only which K-admitted windows may be transported; it does not remove the requirement that the route pass through a displayed cap. Hence cap totality is a structural obligation of the carrier rather than a numerical side condition. □

Theorem 903.10 (Observation-budget limiter window readback exactness)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P,$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier. Each served* $\mathsf{StreamName}^{\uparrow}$ *window in W, each* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row in D, and each* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row in R shares the same capped route*

$$K \to W \to D \to R.$$

The displayed hsame *and* Cont *rows in H and C preserve that route, and the* Pkg *and* NameCert *rows P, N certify no second uncapped readback route.*

Proof. Use the dyadic and regular readback obligation from Definition 903.2. It requires the dyadic ledger D and the regular readback R to be read through the same capped window route that admits W.

The route itself is the finite chain $K \to W \to D \to R$. Componentwise hsame transports in H preserve the chosen cap, window, dyadic, and regular rows, while the displayed Cont rows in C replay the same chain for consumers.

Apply Definition 903.7 to exclude any read of R that bypasses K, W, D. Apply Definition 903.6 only as a transport principle: a larger accepted cap may carry the same readback through transported rows, but the readback remains tied to one capped route. □

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiter window readback exactness`

Theorem 903.11 (Observation-budget limiter Real seal cap boundary)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *seal row S exported by L is reached only after the sequence*

$$E \to B \to K \to W \to D \to R \to S.$$

The boundary displayed to a seal consumer consists exactly of the componentwise hsame *rows in H, the* Cont *rows in C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *row N over that sequence.*

Proof. Apply the seal handoff obligation of Definition 903.2. It permits the $\mathsf{Real}^{\uparrow}$ seal row S only after the request, observation-budget, cap, window, dyadic, and regular readback rows have all been displayed.

Apply Definition 903.9 for the entry side of the boundary and Definition 903.10 for the middle readback. Together they make the seal route pass through E, B, K, W, D, R before S.

Finally apply Definition 903.8. The Real and Completion consumer may use this packet only as a capped observation service. The visible boundary is therefore the same H, C, P, N provenance carried by L, with no completeness, quotient, selected-limit, host-equality, trichotomy, or independent-real coordinate added at the seal. □

Theorem 903.12 (Observation-budget limiter finite service triad)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. The finite service surface exposed by L consists exactly of the request row E, the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row B, the cap row K, the* $\mathsf{StreamName}^{\uparrow}$ *window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal row S, the componentwise transports H, the displayed routes C, the package provenance P, and the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *row N. A downstream Real or Completion consumer may cite this surface as the limiter triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N),$$

and receives no additional unbounded schedule, quotient stream, host real equality, selected limit, trichotomy row, independent real carrier, or ambient completeness row.

Proof. Project the accepted carrier through Definition 903.1 and apply Definition 903.2. These rows expose the request, budget, cap, window, dyadic, readback, seal, transport, route, provenance, and naming coordinates as the whole carrier surface.

For the entry part, use Definition 903.3 and Definition 903.9. They force every served window, readback, or seal read to pass through E, B, K before any downstream row is available.

For the middle part, use Definition 903.5, Definition 903.6, and Definition 903.7. These theorems keep the window and dyadic service inside the displayed W, D, R rows under the same cap.

For the exit part, apply Definition 903.10 and Definition 903.11. The seal row is therefore read only after the exact capped source-window-readback chain, with H, C, P, N recording the visible transport, route, package, and local certificate data. □

Theorem 903.13 (Observation-budget limiter source-window-readback lock)**.** *Let $L = (E, B, K, W, D, R, S, H, C,$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Every downstream read of the capped service factors through the same source-window-readback rows*

$$E \to B \to K \to W \to D \to R.$$

In particular, no downstream read may replace W, D, R by host real equality, a hidden selector, an uncapped schedule, or a second readback route outside the displayed hsame, Cont, Pkg, *and* NameCert *rows of L.*

Proof. Begin with Definition 903.3. It places the source request and observation-budget rows before the cap row, and it makes the cap a coordinate of the accepted packet rather than an external numerical side condition.

Next apply Definition 903.5. Transport may move the displayed rows componentwise, but it does not create a new observation source or an uncapped service. The monotonicity theorem Definition 903.6 has the same shape: it can widen an accepted cap while preserving the same displayed route discipline.

Finally use Definition 903.7 and Definition 903.10. They identify $K \to W \to D \to R$ as the only regular readback route in the packet. Hence a host equality, hidden selector, uncapped schedule, or second readback has no carrier row and no local NameCert clause from which a downstream read can be certified. □

Theorem 903.14 (Observation-budget limiter downstream seal eligibility)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P,$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. A* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, *or Real-route consumer is eligible to consume the limiter only after it has read the finite service triad of Definition 903.12. Thus every eligible seal consumer has already displayed the request-budget-cap entry, the source-window-readback middle, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary with the same H, C, P, N provenance.*

Proof. First read the limiter as a finite service packet by Definition 903.12. This exposes the three parts of the surface without adding a completion theorem, quotient stream, selected limit, host equality, or independent real carrier.

Next apply Definition 903.13. Any downstream consumer of the capped service must use the same E, B, K, W, D, R chain, so the consumer cannot replace the middle route by a hidden selector or host real equality.

Finally apply Definition 903.4 and Definition 903.11. They place the $\mathsf{Real}^{\uparrow}$ seal row after the capped readback and identify H, C, P, N as the seal-facing boundary. A $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, or Real-route consumer therefore becomes eligible only after the displayed finite service triad has been read. □

Theorem 903.15 (Observation-budget limiter kernel scope)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Every accepted limiter read is scoped by the displayed* BHist *carrier rows E, B, K, W, D, R, S, the componentwise* hsame *row H, the continuation-route row C, the package provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *row N. No accepted read of the limiter may cite a kernel object outside those rows before it reaches a downstream seal consumer.*

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiter kernel scope fields`

Proof. Project the carrier through Definition 903.1. The only carrier coordinates are $E, B, K, W, D, R, S, H, C, P, N$ with H storing the componentwise hsame transports and C storing the displayed Cont routes.

Apply Definition 903.2. Its cap, window, dyadic, regular, seal, and non-escape clauses read those coordinates as the local NameCert surface.

Finally use Definition 903.14. A downstream seal consumer becomes eligible only after the same finite service triad has been read, so the kernel scope is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 903.16 (Observation-budget limiter TasteGate carrier alignment)**.** *The paper carrier fields of* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *are exactly the fields that a Lean* `ChapterTasteGate` *and* `BHistCarrier` *instance for this chapter must expose: the request row E, observation-budget row B,*

cap row K, window row W, dyadic row D, regular readback row R, seal row S, transport row H, continuation row C, package row P, and local name row N. The alignment asks for no extra Real carrier, quotient stream, selected limit, host equality, trichotomy row, or ambient completeness field.

Proof. Read the field list from Definition 903.1. The chapter's carrier is a finite BHist packet with precisely the eleven displayed coordinates.

Use Definition 903.15 to identify the same coordinates as the whole accepted-read scope. A Lean carrier instance for this paper site therefore has to expose those rows and their rowwise transport, route, provenance, and name data.

The exclusion part follows from Definition 903.12 and Definition 903.25. Both theorems deny extra analytic fields, so the TasteGate-facing carrier aligns with the paper carrier rather than with a stronger completion object. □

Theorem 903.17 (Observation-budget limiter seal field boundary)**.** *For every accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet $L = (E, B, K, W, D, R, S, H, C, P, N)$, the seal field S is reachable only through the displayed sequence $E, B, K, W, D, R, S, H, C, P, N$. The packet exports no independent Real carrier, no hidden completeness row, and no seal-facing field outside the request, budget, cap, window, dyadic, regular, transport, continuation, package, and local naming surface.*

Proof. Apply Definition 903.4. It places the seal row after the finite route $E \to B \to K \to W \to D \to R \to S$.

Apply Definition 903.11. The rows H, C, P, N are the only seal-facing provenance boundary attached to that route.

Finally use Definition 903.15. Since every accepted limiter read remains inside the displayed kernel rows, there is no independent Real carrier or completeness coordinate from which another seal field could be exported. □

Theorem 903.18 (Observation-budget limiter primitive scope)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Every cap, window, dyadic observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, package, and* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *obligation of L factors only through the displayed* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. No obligation of the packet factors through a quotient stream, host real equality, selected limit, trichotomy row, independent real carrier, or ambient completeness principle.*

Proof. Project the accepted carrier through Definition 903.1. The coordinates of L are exactly the request, observation-budget, cap, window, dyadic, regular readback, seal, transport, continuation, package, and local naming rows listed there.

Apply Definition 903.2. Its obligations read those coordinates as carrier admission, cap admission, finite-window service, dyadic and regular readback, seal handoff, and non-escape clauses, all inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

Use Definition 903.12 and Definition 903.13 to identify the only downstream route as the finite triad

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

The non-escape clauses in those results exclude quotient streams, host real equality, selected limits, trichotomy, independent real carriers, and ambient completeness, so every listed obligation remains in the primitive scope. □

Theorem 903.19 (Observation-budget limiter scoped route factorization)**.** *Let* $L = (E, B, K, W, D, R, S, H, C, P, N$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, *or* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *consumer of* L *factors through the same scoped primitive route*

$$E \to B \to K \to W \to D \to R \to S,$$

with H, C, P, N *as the only transport, continuation, provenance, and* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *boundary. In particular, the consumer may read the limiter only as a finite BEDC primitive service and not through a quotient stream, host real equality, selected limit, trichotomy row, independent real carrier, or ambient completeness principle.*

Proof. Project the accepted packet through Definition 903.1. The carrier supplies exactly the rows $E, B, K, W, D, R, S, H, C, P, N$. Apply Definition 903.2 to read the same coordinates as the local $\mathsf{NameCert}$ obligations for carrier admission, cap admission, finite-window service, dyadic and regular readback, seal handoff, and non-escape.

The ordered primitive route is forced by the cap and readback clauses. Use Definition 903.3 to place K after the source request and observation-budget rows. Then use Definition 903.13 and Definition 903.10 to make $W \to D \to R$ the only regular readback route downstream from that cap.

For downstream consumers, apply Definition 903.14. It makes each $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, and Real-route consumer read the finite service triad before a seal-facing row is available. Use Definition 903.22 for $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ consumers and Definition 903.23 for $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ consumers; both results factor their reads through the same limiter triad.

Finally combine Definition 903.18 with Definition 903.25. The primitive-scope theorem confines every obligation to the displayed BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, while the service-triad coverage theorem says the listed consumers receive no second source, hidden selector, quotient stream, host real equality, selected limit, trichotomy row, independent real carrier, or ambient completeness. Thus every listed consumer factors through $E \to B \to K \to W \to D \to R \to S$ with H, C, P, N as the only boundary rows. □

Theorem 903.20 (Observation-budget limiter TasteGate boundary)**.** *The falsifiable prediction, independence witness, carrier rows, cap-admission theorem, seal-ordering theorem, capped-window transport theorem, and primitive-scope theorem of this chapter present the exact finite carrier boundary that a later Lean* `Nontrivial` *plus* `FieldFaithful` *TasteGate instance for* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *must compile. The boundary is the finite limiter packet surface and not an ambient Real completeness, quotient, selected-limit, host-equality, trichotomy, or independent-carrier assertion.*

Proof. The prediction and independence witness at the start of the chapter isolate the falsifiable role of the budget-cap gate: deleting it changes the Real and Completion service, and no listed sibling is bijective with the carrier.

The carrier boundary is then fixed by Definition 903.1. Apply Definition 903.3, Definition 903.4, and Definition 903.5 to make the accepted packet usable only through its finite cap, ordered seal route, and transported capped window.

Finally apply Definition 903.18. It states that every obligation used by the packet remains inside the displayed BEDC rows. Therefore the later TasteGate instance has a precise finite boundary to compile and receives no analytic completion principle from this paper theorem. □

Theorem 903.21 (Observation-budget limiter obligation closure package)**.** *For an accepted* ObservationBudgetLimi *packet* $L = (E, B, K, W, D, R, S, H, C, P, N)$*, the NameCert obligations, cap admission, seal ordering, capped-window transport, budget monotonicity, window non-escape, vision handoff, finite service triad, source-window-readback lock, downstream seal eligibility, primitive scope, and TasteGate boundary jointly form the paper obligation package for the finite limiter packet.*

Proof. Start with Definition 903.2. It supplies the local carrier, cap, window, readback, seal, and non-escape clauses that every later route must project from.

Use Definition 903.3, Definition 903.4, and Definition 903.5 for the main route obligations. These place K before W, D, R, S, order the seal after readback, and keep transported consumers inside the same capped service.

Use Definition 903.6, Definition 903.7, and Definition 903.8 for stability and downstream handoff. They preserve the displayed rows under cap enlargement, exclude escape from the finite source window, and hand the service to Real and Completion consumers as a capped observation service.

Finally use Definition 903.12, Definition 903.13, Definition 903.14, Definition 903.18, and Definition 903.20. These identify the exported triad, lock source-window-readback use, require seal consumers to read the triad first, keep the whole packet in primitive scope, and specify the finite TasteGate-facing boundary. Their composition is exactly the obligation package stated in the theorem. □

Theorem 903.22 (Observation-budget limiter selector compatibility)**.** *Let* $L = (E, B, K, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Any* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *consumer may read the limiter only through the request-budget-cap service triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

In particular, selector-facing use of the cap row K *does not expose a hidden selector, an uncapped observation schedule, a quotient stream, selected host limit, host real equality, or ambient completeness.*

Proof. Start with Definition 903.12. It identifies the whole exported surface of the limiter as the request-budget-cap entry, the source-window-readback middle, and the seal boundary with H, C, P, N.

Apply Definition 903.13. A selector-facing consumer cannot replace E, B, K, W, D, R by a hidden selector or second readback route, because the capped service has no such carrier row.

Finally use Definition 903.14. Any downstream consumer that continues toward a seal must already have displayed the same finite service triad, so selector compatibility is a projection of the limiter surface rather than a separate observation primitive. □

Theorem 903.23 (Observation-budget limiter Cauchy tail-budget factorization)**.** *Let* $L = (E, B, K, W, D, R, S, H,$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Any* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *consumer that uses* L *as its* O *row must first factor that row through the same limiter triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N)$$

before it serves tail windows, dyadic observations, regular readback, and seal rows. Thus the tail-budget route inherits the limiter's displayed H, C, P, N *provenance and no uncapped observation source.*

Proof. Apply Definition 903.12 to expose the service consumed as O. The limiter row contributes exactly $E, B, K, W, D, R, S, H, C, P, N$ and no external cap primitive.

Use Definition 903.13 for the window and readback portion. It forces any downstream use of the capped service to pass through $E \to B \to K \to W \to D \to R$ before a seal-facing row is available.

Then apply Definition 903.4 and Definition 903.14. They make any tail-budget seal consumer eligible only after the same triad has been read, so a $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ packet can consume O only as this finite limiter service. □

Theorem 903.24 (Observation-budget limiter tail-budget shared cap)**.** *Let $L = (E, B, K, W, D, R, S, H, C, P, N)$ be an accepted $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet. If an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ packet consumes L as its observation-budget row O, then its tail threshold, finite windows, dyadic observations, regular readback, and seal route all inherit the same displayed cap row K. No uncapped observation source and no second budget gate are available to that tail-budget consumer.*

Proof. Start with Definition 903.23. It forces the consumed O row to factor through the limiter triad

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

Apply Definition 903.9 to the entry side of that triad. The cap row K is the displayed budget gate for every served observation, so the tail threshold, windows, dyadic observations, regular readback, and seal route cannot select an uncapped source.

Finally use the $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ observation gate of Definition 906.3. That gate reads O as the observation-budget surface for the tail packet, so the factorized limiter row supplies the same K to all downstream tail-budget fields and leaves no second budget gate to project. □

Theorem 903.25 (Observation-budget limiter service triad coverage)**.** *Every listed $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, and Real-route consumer of an accepted $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet receives exactly the visible service triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N),$$

including the same transport, continuation, package, and local naming rows H, C, P, N. No such consumer receives an unbounded schedule, hidden selector, quotient stream, selected host limit, host real equality, independent real carrier, or ambient completeness row.

Proof. First apply Definition 903.14. It states that $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, and Real-route consumers become eligible only after the finite service triad has been read.

Next apply Definition 903.4. The seal row is reached only through $E \to B \to K \to W \to D \to R \to S$, so the visible triad covers the full entry, readback, and seal-facing route.

Finally use Definition 903.12 and Definition 903.13. These identify H, C, P, N as the displayed provenance for that route and exclude any second source, hidden selector, host equality, quotient, selected limit, independent real carrier, or completeness witness. □

Theorem 903.26 (Observation-budget limiter public seal interface)**.** *Every public consumer of an accepted $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet reads the capped finite window, budget cap, selector compatibility, Cauchy tail-budget factorization, and Real-seal boundary only through the displayed carrier rows*

$$E, B, K, W, D, R, S, H, C, P, N$$

and their Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$ *provenance. No public consumer receives an unbounded observation schedule, hidden selector, quotient stream, selected limit, host real equality, independent real carrier, or ambient completeness row from the packet.*

Proof. Begin with the public service triad of Definition 903.25. It identifies the consumer-visible surface as

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

Apply Definition 903.22. Selector-facing consumers must read the capped service through $E, B, K, W, D, R, S, H, C, P, N$, so selector compatibility is a carrier projection rather than a hidden selector row.

Next apply Definition 903.23 and Definition 903.24. A $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ consumer inherits the same cap row K and the same provenance rows when it consumes the limiter as its observation-budget field.

Finally use Definition 903.11 and Definition 903.19. The Real-seal endpoint is reached only through the displayed route, and the route exports no unbounded schedule, hidden selector, quotient stream, selected limit, host equality, independent carrier, or ambient completeness principle. □

Theorem 903.27 (Observation-budget limiter cap intersection)**.** *Let* $L = (E, B, K, W, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Suppose two accepted limiter packets share the transported request row* E*, observation-budget row* B*, and budget-cap row* K *with* L*. Then every shared* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the intersection of the displayed cap routes*

$$E \to B \to K \to W \to D \to R \to S$$

and cannot project an observation read outside the common K, H, C, P, N *provenance.*

Proof. Begin with Definition 903.9. Once the transported request and observation-budget rows agree, the shared cap row K is the only displayed gate from which served observations can enter the window, dyadic ledger, regular readback, and Real-seal rows.

Next apply Definition 903.13. It locks the source, finite window, dyadic observation, and regular readback rows along the same route, so a shared consumer cannot leave the common cap route after the K projection.

Finally use Definition 903.25. The public consumer surface is exactly the triad

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N),$$

hence the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers factor through the intersection of the two displayed cap routes rather than through an external observation source. □

Theorem 903.28 (Observation-budget limiter selector tail-budget compatibility)**.** *Let* $L = (E, B, K, W, D, R, S, H$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Any accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *consumer and any accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *consumer that read the same limiter row* L *must read the same finite service triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

In particular, the selector side and the tail-budget side use the same request, cap, window, dyadic ledger, regular readback, seal, transport, continuation, package, and local naming data.

Proof. First apply Definition 903.22. A selector-facing consumer reads the capped service through the displayed rows $E, B, K, W, D, R, S, H, C, P, N$ and does not receive a hidden selector row from the limiter.

Next apply Definition 903.23. A $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ consumer that uses L as its observation-budget row factors through the same displayed request, cap, window, dyadic ledger, regular readback, seal, and provenance rows.

Finally combine Definition 903.24 with Definition 903.27. The selector and tail-budget reads have the same accepted limiter row, so both accesses intersect at K and then continue through the common W, D, R, S, H, C, P, N service surface. □

Theorem 903.29 (Observation-budget limiter Real-route budget exhaustion)**.** *Every Real-route,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{UniformLimit}^{\uparrow}$*,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*, or* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet exhausts its limiter access through the displayed finite triad*

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

The packet supplies no unbounded observation schedule, quotient stream, selected limit, host equality, trichotomy, independent $\mathsf{Real}^{\uparrow}$ *carrier, or completeness row to any of those consumers.*

Proof. Project the accepted carrier as $L = (E, B, K, W, D, R, S, H, C, P, N)$. For public Real-route, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{UniformLimit}^{\uparrow}$ consumers, apply Definition 903.26; it exposes only the displayed finite service route and excludes the listed escape surfaces.

For selector and tail-budget consumers, use Definition 903.28. Both consumers read the same triad from L, and Definition 903.27 pins their shared readback to the common cap route through K.

Finally close with Definition 903.18 and Definition 903.25. The primitive scope contains only the finite request, cap, window, dyadic ledger, regular readback, seal, transport, continuation, package, and local naming rows, so every listed consumer exhausts its access inside that finite surface. □

Theorem 903.30 (Observation-budget limiter budget readback saturation)**.** *Let* $L = (E, B, K, W, D, R, S, H, C, P$ *be an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet. Every served* $\mathsf{StreamName}^{\uparrow}$ *window in* W*, every* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger read in* D*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* R*, and every* $\mathsf{Real}^{\uparrow}$ *seal read in* S *consumes the same displayed budget-cap row* K *along the route*

$$E \to B \to K \to W \to D \to R \to S.$$

No such read can be split across two budget services, and no downstream consumer can replace K *by an uncapped observation source outside the displayed* H, C, P, N *provenance.*

Proof. First apply Definition 903.9. It places the cap row K after the request and observation-budget rows E, B, and before any served window, dyadic ledger, regular readback, or seal row can be read.

Next use Definition 903.10. The middle route $K \to W \to D \to R$ is exact for the served windows, dyadic observations, and regular readback rows, so the readback part cannot change to a second cap service.

Finally compose Definition 903.24 with Definition 903.26. Tail-budget consumers and public seal consumers both read the limiter through the same displayed carrier rows and provenance fields, hence the downstream readback saturates one cap row K rather than two independent budget gates. □

Theorem 903.31 (Observation-budget limiter Real and Completion consumer boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet may use the limiter only as the finite route*

$$E \to B \to K \to W \to D \to R \to S$$

with the displayed H, C, P, N *provenance. The packet supplies a capped observation service for downstream Real and Completion chapters, not Real completeness, host real equality, quotient streams, selected limits, trichotomy, or an independent real carrier.*

Proof. Start with Definition 903.19. It factors every public route through the ordered primitive spine from E to S and keeps H, C, P, N as the visible transport, continuation, package, and local naming rows.

Then apply Definition 903.25. Real-route consumers and listed seal consumers receive the same service triad

$$(E, B, K) \quad (W, D, R) \quad (S, H, C, P, N).$$

Finally use Definition 903.26. The public interface already excludes completeness, host equality, quotient streams, selected limits, trichotomy, and independent real carriers, so Real and Completion consumers can cite the limiter only as the displayed finite service route. □

Theorem 903.32 (Observation-budget limiter tail-service downstream readback)**.** *Any* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *packet inherits the same budget-readback route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*:*

$$K \to W \to D \to R \to S.$$

The inherited route carries the same E, B, H, C, P, N *boundary data and does not expose a second observation budget, host equality, quotient stream, selected limit, or ambient completeness row.*

Proof. First apply Definition 903.23. A $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ consumer that uses the limiter as its observation row must factor that row through the same finite limiter triad before it serves windows, dyadic observations, regular readback, or seal rows.

Next use Definition 903.24. The tail threshold, finite windows, dyadic observations, regular readback, and seal route inherit one displayed cap row K, so the tail-service readback cannot draw from a second observation budget.

Finally apply Definition 903.14. A $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer is eligible only after the finite service triad has been read, so the downstream seal read continues through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ with the same E, B, H, C, P, N boundary. □

The follow-up Lean TasteGate for this chapter must provide `Nontrivial ObservationBudgetLimiterUp` and `FieldFaithful ObservationBudgetLimiterUp` instances for the same carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ packet is generated from a finite request row, observation-budget row, budget-cap row, finite StreamName window ledger, DyadicRatCore observation ledger, RegSeqRat readback row, RealUp seal row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservationBudgetLimiter}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 903.1, Definition 903.2, Definition 903.3, Definition 903.4, Definition 903.5, Definition 903.6, Definition 903.7, Definition 903.8, Definition 903.18, Definition 903.19, Definition 903.21, Definition 903.22, Definition 903.23, Definition 903.24, Definition 903.25, Definition 903.26, Definition 903.27, Definition 903.28, Definition 903.29, Definition 903.30, Definition 903.31, and Definition 903.32 bind finite request caps, NameCert rows, selector compatibility, Cauchy tail-budget factorization, tail-budget shared-cap routing, cap intersection, Real-route budget exhaustion, service-triad coverage, the mature public-consumer seal surface, downstream Real and Completion readback,

the paper bridge surface, and the primitive route $E \to B \to K \to W \to D \to R \to S$ before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal reads.

Formal verification: FormalStatus(NameCert$_{\text{ObservationBudgetLimiter}\uparrow}$, axiomClean)

Lean4 ✓ `BEDC.Derived.ObservationBudgetLimiterUp.taste gate`

Lean target: `BEDC.Derived.ObservationBudgetLimiterUp.ObservationBudgetLimiterTasteGate\_single\_carrier\_alignmen`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, Real completeness, quotient streams, host real equality, selected limits, trichotomy, independent real carriers, or ambient completeness.

Upgrade path: The theory axis is complete for the displayed finite-service and Real-seal cap boundary; Lean-side bridge checking remains a formal-axis task.

Chapter 904

A Concrete Naming Certificate for MetaCICAuditMatrix$^{\uparrow}$

The MetaCICAuditMatrix$^{\uparrow}$ packet realizes the matrix and critical-path reading of chapter 9385 as a finite NameCert surface. It sits beside the normalization frontier of chapter 892, the subject-reduction discharge ledger of chapter 720, the closed-normal consistency packet of chapter 739, and the audit-map dependency weave of chapter 839. Those siblings expose frontier rows, discharge rows, closed-normal endpoint rows, and cross-map routing rows; the present packet records the status matrix that keeps positive rows, non-claim rows, blocked edges, synthesis consumers, and non-escape ledgers inspectable in one carrier.

Definition 904.1 (MetaCIC audit matrix carrier)**.** *A* MetaCICAuditMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, N, B, S, E, H, C, P, L)$$

with the following displayed rows:

1. *A is the available-row matrix, recording closed-normal consistency, candidate-mediated normalization, local decidable checking, closed substitution, and conditional confluence as positive audit rows;*
2. *N is the non-claim row matrix, recording full closed consistency, arbitrary closed Church–Rosser, candidate-free normalization, unrestricted beta-conversion decidability, and host substitution semantics as non-exported endpoints;*
3. *B is the blocked-edge matrix, with the subject-reduction lift as the critical edge from arbitrary closed proof terms to the closed-normal consistency branch;*
4. *S is the synthesis-consumer row, naming the consumers that may read the matrix before using closed-normal packets, discharge ledgers, or audit-map dependency routes;*
5. *E is the non-escape ledger, recording that every consumer route must preserve A, N, B, S as displayed rows;*
6. *H records componentwise* hsame *transports for A, N, B, S, E and the structural rows;*
7. *C records the displayed* Cont *routes by which a consumer reads available rows, non-claim rows, blocked edges, synthesis consumers, and non-escape ledgers in matrix order;*
8. *P is the* Pkg *provenance over chapter 9385, chapter 892, chapter 720, chapter 739, and chapter 839;*

9. L is the local $\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}$ *row.*
The carrier has no coordinate for removing the blocked subject-reduction edge, no hidden full-consistency theorem, no host reduction semantics, and no quotient equality of MetaCIC terms.

Lean4 ✓ `BEDC.Derived.MetaCICAuditMatrixUp.MetaCICAuditMatrixTasteGate single carrier alignment`

Theorem 904.2 (MetaCIC audit matrix NameCert obligations)**.** *For the carrier of Definition 904.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (A, N, B, S, E, H, C, P, L),$$

available-row exposure, non-claim row exposure, blocked-edge exposure, synthesis-consumer exposure, non-escape ledger exposure, componentwise hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.MetaCICAuditMatrixUp.MetaCICAuditMatrixUp obligation surface`

Proof. Carrier admission is projection from Definition 904.1. The rows $A, N, B, S, E, H, C, P, L$ are the complete finite packet.

The positive matrix entries are read from A and routed through the sibling frontier and closed-normal packets named in P. The non-claim entries are read from N, so the unavailable endpoints remain explicit rows rather than omitted side conditions.

The critical edge is read from B, while the consumers are read from S. The non-escape row E then records that those consumers receive the displayed matrix and not an erased endpoint.

Finally H, C, P, L supply transport, route replay, provenance, and local naming for the same rows. Since no additional coordinate exists, the NameCert obligations are exhausted by the displayed packet. □

Theorem 904.3 (MetaCIC audit matrix critical-path route)**.** *Every accepted* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *carrier routes closed MetaCIC consistency consumers through the matrix order*

$$A \longrightarrow B \longrightarrow S \longrightarrow E.$$

The available rows A may feed the closed-normal consistency packet and audit-map dependency weave, but any consumer seeking full closed consistency must first read the blocked subject-reduction edge B and the synthesis-consumer row S.

Lean4 ✓ `BEDC.Derived.MetaCICAuditMatrixUp.MetaCICAuditMatrixCarrier critical path route`

Proof. Project the accepted carrier and read the continuation row C. Its displayed route reads the available matrix rows before the blocked-edge and consumer rows.

Use the provenance row P to identify the sibling surfaces. The closed-normal consistency packet supplies the positive endpoint, the subject-reduction discharge ledger supplies the required preservation slots, and the audit-map dependency weave supplies the cross-map consumer route.

The route cannot bypass B. The blocked-edge matrix is a coordinate of the same packet as A, S, E, so a full-consistency consumer has to expose it before any synthesis handoff can be read. □

Theorem 904.4 (MetaCIC audit matrix non-escape)**.** *Every consumer-facing route out of an accepted* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *carrier factors through*

$$A, N, B, S, E, H, C, P, L.$$

If the subject-reduction blocked edge B is removed from the packet, the non-escape ledger E no longer has a row through which to refuse the full closed-consistency read. Such a packet therefore fails the non-escape theorem for $\mathsf{MetaCICAuditMatrix}^{\uparrow}$.

Proof. Start with the carrier inventory. The available rows, non-claim rows, blocked edge, synthesis consumers, and non-escape ledger are all named coordinates of one finite packet.

The continuation and provenance rows C and P can only replay and package those displayed coordinates. They do not create a second route to full closed consistency, host substitution semantics, or quotient equality of MetaCIC terms.

Now remove B from the packet. The row E is defined as the ledger that preserves A, N, B, S under consumer reads, so without the blocked-edge coordinate it cannot certify refusal of the full closed-consistency route. Hence any accepted packet missing B is outside the stated non-escape boundary. □

Theorem 904.5 (MetaCIC audit matrix obligation surface)**.** *Every accepted* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *carrier exposes the matrix obligations of the rows*

$$A, N, B, S, E, H, C, P, L.$$

The available row A, non-claim row N, blocked-edge row B, synthesis-consumer row S, and non-escape ledger E are all inspectable before structural transport, route replay, package provenance, and local naming are read. No consumer may erase one of these rows and still claim to read the same audit matrix.

Proof. Project the carrier of Definition 904.1. The rows A, N, B, S, E are the displayed audit rows, while H, C, P, L are the structural rows that transport, replay, source, and name the same packet.

Use the NameCert obligations of Definition 904.2. They already require carrier admission, available-row exposure, non-claim exposure, blocked-edge exposure, synthesis-consumer exposure, and non-escape ledger exposure.

Finally apply the non-escape theorem. Since every consumer route factors through the whole row sequence, deleting any of A, N, B, S, E changes the carrier surface rather than producing another read of $\mathsf{MetaCICAuditMatrix}^{\uparrow}$. □

Theorem 904.6 (MetaCIC audit matrix scope route)**.** *Every consumer of the* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *matrix routes through the normalization frontier, subject-reduction discharge ledger, closed-normal consistency packet, and audit-map dependency weave named by the provenance row P. The route exposes what the matrix makes available and what it refuses; it does not supply full closed consistency, arbitrary closed Church–Rosser, candidate-free normalization, unrestricted beta-conversion decidability, host substitution semantics, or quotient equality.*

Proof. Start with the critical-path route of Definition 904.3. It sends closed MetaCIC consistency consumers from available rows to blocked edges, synthesis consumers, and the non-escape ledger.

Read the sibling anchors from P. The normalization frontier supplies the available normalization-facing rows, the subject-reduction discharge ledger names the blocked preservation edge, the closed-normal consistency packet supplies the positive endpoint, and the audit-map dependency weave records the cross-map consumer route.

The non-claim row N remains visible along this route. Therefore each unavailable endpoint is refused as a matrix row rather than silently imported from a sibling chapter or a host MetaCIC semantics. □

Theorem 904.7 (MetaCIC audit matrix closure ledger)**.** *The blocked-edge ledger of an accepted* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *packet remains visible under every closure-facing read. In particular, the subject-reduction blocked edge* B *is carried together with* A, N, S, E, H, C, P, L *whenever a consumer reads available rows, synthesis consumers, closed-normal packets, discharge ledgers, or audit-map dependency routes from the matrix.*

Proof. Use the obligation surface theorem. It makes the blocked-edge row B one of the displayed obligations of the carrier, not an external annotation.

Apply the scope route theorem. Any closure-facing consumer reaches sibling data through the provenance and continuation rows while preserving the visible matrix order through available rows, non-claims, blocked edges, synthesis consumers, and non-escape ledgers.

Finally invoke non-escape. If B is removed, the ledger E can no longer refuse the full closed-consistency read. Hence closure-facing reads keep the blocked-edge ledger visible as part of the finite packet. □

Theorem 904.8 (MetaCIC audit matrix obligation closure package)**.** *For the carrier of Definition 904.1, the same finite* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *packet closes the obligation surface consisting of carrier admission, matrix-row exactness, critical-path routing, blocked-edge preservation, non-escape, and TasteGate field faithfulness. Concretely, any accepted packet*

$$M = (A, N, B, S, E, H, C, P, L)$$

that satisfies the local $\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}$ *row exposes exactly the available-row matrix* A*, non-claim matrix* N*, blocked-edge matrix* B*, synthesis-consumer row* S*, non-escape ledger* E*, transport rows* H*, route rows* C*, provenance row* P*, and naming row* L *before a consumer may read the MetaCIC audit surface. No route through this package erases* B*, bypasses* E*, replaces* P *by host MetaCIC semantics, or reads TasteGate field faithfulness from a different carrier.*

Proof. Carrier admission is the carrier statement of Definition 904.1, and the local naming obligations are the obligations listed in Definition 904.2. These give the rows $A, N, B, S, E, H, C, P, L$ as one finite packet.

Matrix-row exactness and critical-path routing are supplied by Definition 904.5 and Definition 904.3. The surface theorem fixes which rows are visible, while the route theorem fixes the order in which closed MetaCIC consistency consumers must read available rows, blocked edges, synthesis consumers, and the non-escape ledger.

Blocked-edge preservation follows from Definition 904.7. Non-escape follows from Definition 904.4, and the consumer scope is the scope route of Definition 904.6. These three results keep B and E inside every closure-facing read.

Finally, the TasteGate field-faithfulness clause is read as a requirement on the same displayed packet, not on a sibling or host carrier. Since the package uses the carrier, route, non-escape, obligation surface, scope route, and closure ledger for the same M, the obligation closure surface is bound to one finite audit matrix. □

Falsifiable prediction: Any accepted $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ packet with the subject-reduction blocked edge removed fails Definition 904.4.

Independence witness: metacicnormalizationfrontier, subjectreductiondischargeledger, closednormalconsistencypacket, auditmapdependencyweave: none carries all four available-row, non-claim-row, blocked-edge, and synthesis-consumer matrices as one NameCert packet.

The Lean follow-up target is `lean4/BEDC/Derived/MetaCICAuditMatrixUp/TasteGate.lean`, with explicit `Nontrivial MetaCICAuditMatrixUp` and `FieldFaithful MetaCICAuditMatrixUp` instances.

Theorem 904.9 (MetaCIC audit matrix scoped status)**.** *Every scoped read of an accepted* $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ *packet factors through the available-row matrix A, non-claim matrix N, blocked-edge matrix B, synthesis-consumer row S, non-escape ledger E, normalization frontier, subject-reduction discharge ledger, closed-normal consistency packet, and audit-map dependency weave before any closed-consistency consumer reads the matrix. The read therefore preserves the finite dependency order*

$$A, N, B, S, E, H, C, P, L$$

and cannot erase the subject-reduction blocked edge, replace P by host MetaCIC semantics, or treat a non-claim row as a closed-consistency theorem.

Proof. Start with the critical-path route of Definition 904.3. It places closed MetaCIC consistency consumers after the available-row matrix, the blocked-edge matrix, the synthesis-consumer row, and the non-escape ledger.

Use the obligation surface and scope route of Definition 904.5 and Definition 904.6. These results expose the same rows $A, N, B, S, E, H, C, P, L$ and force the provenance row through the normalization frontier, subject-reduction discharge ledger, closed-normal consistency packet, and audit-map dependency weave.

Finally apply the closure ledger and non-escape theorems. The ledger keeps B visible in every closure-facing read, while non-escape refuses any route that deletes the blocked edge before reading closed-consistency consumers. Thus the accepted packet supplies scoped status exactly through the displayed MetaCIC audit matrix route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICAuditMatrix}^{\uparrow}$ packet is generated from available audit rows, non-claim rows, blocked-edge rows, synthesis-consumer rows, a non-escape ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 904.1, Definition 904.2, Definition 904.3, Definition 904.4, Definition 904.5, Definition 904.6, Definition 904.7, Definition 904.8, and Definition 904.9 bind the finite MetaCIC audit matrix packet, its carrier and NameCert obligations, critical-path route, non-escape boundary, scope route, blocked-edge ledger, obligation-closure package, and scoped consumer status.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICAuditMatrix}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.MetaCICAuditMatrixUp.MetaCICAuditMatrixTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed consistency, arbitrary closed Church–Rosser, candidate-free normalization, unrestricted beta-conversion decidability, host substitution semantics, public closure, or bridge checking.

Upgrade path: Public closure requires an export theorem showing that the scoped MetaCIC audit matrix route is the public read surface for closed-consistency consumers.

Chapter 905

A Concrete Naming Certificate for $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$

The $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ packet realizes the diagonal witness route of chapter 9238 as one finite grid. It joins the witness ledger of chapter 902, the completion-seal classifier of chapter 900, the selector-budget surface of chapter 678, and the tail-witness envelope of chapter 898. The grid is a NameCert surface for the displayed selector, ledger, seal, and tail rows; it is not a quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness theorem.

Definition 905.1 (Real Cauchy witness grid carrier)**.** *A* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, L, K, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *row exposing the finite selector modulus and selected observation window for the diagonal request;*
2. *L is the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row recording the diagonal budget, observation budget, seal synchronizer, and completion-seal classifier route;*
3. *K is the* $\mathsf{RealCompletionSealClassifierUp}$ *row reached by the completion-facing seal read;*
4. *T is the* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *row storing the selected StreamName window, finite-window endpoint ledger, tail comparison, tail-agreement seal, and RealUp readback;*
5. *S records the* $\mathsf{StreamName}^{\uparrow}$ *finite windows shared by the selector and tail rows;*
6. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness rows assembled from the selected windows;*
7. *E records the* $\mathsf{Real}^{\uparrow}$ *seal and readback rows exported after the ledger and classifier rows are displayed;*
8. *H records componentwise* hsame *transports for $B, L, K, T, S, R, E, C, P, N$;*
9. *C records displayed* Cont *routes from the selector budget to the witness ledger, completion-seal classifier, tail-witness envelope, StreamName rows, RegSeqRat rows, RealUp seal rows, provenance, and local naming row;*
10. *P records* Pkg *provenance over* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, $\mathsf{RealCompletionSealClassifi}$ $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyWitnessGrid}^{\uparrow}}$ *row.*
The carrier contains no coordinate for quotient stream equality, host real equality, selected external limits, countable choice, hidden selector enlargement, or ambient Real completeness.

Theorem 905.2 (Real Cauchy witness grid NameCert obligations)**.** *For the carrier of Definition 905.1, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyWitnessGrid}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$G = (B, L, K, T, S, R, E, H, C, P, N),$$

selector-budget exposure through B, witness-ledger exposure through L, completion-seal classifier exposure through K, tail-witness exposure through T, shared StreamName window exposure through S, RegSeqRat witness exposure through R, RealUp seal exposure through E, componentwise hsame *transport through H, displayed* Cont *route replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealCauchyWitnessGridUp.RealCauchyWitnessGridUp obligation surface`

Proof. Carrier admission is projection from the displayed tuple. The rows $B, L, K, T, S, R, E, H, C, P, N$ are all public coordinates of the grid.

The analytic rows are read in order. The selector-budget row B supplies the finite modulus and window request; the witness-ledger row L stores the diagonal budget, observation budget, synchronizer, and classifier route; the classifier row K reads the completion-seal boundary; and the tail-witness row T stores the selected tail envelope.

The shared rows S, R, E expose the StreamName, RegSeqRat, and RealUp handoff data consumed by those four packets. The structural rows H, C, P, N transport, replay, authenticate, and name the same grid. Since the carrier has no forbidden analytic coordinate, these rows exhaust the NameCert surface. □

Theorem 905.3 (Real Cauchy witness grid selector exactness)**.** *Every selector read admitted by an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier factors through the displayed finite route*

$$B \longmapsto (S, R) \longmapsto (L, T)$$

before the completion-seal classifier K or RealUp seal row E can be consumed. Thus the selected modulus and window are the only selector source for both the ledger side and the tail-witness side of the grid.

Proof. Project the accepted carrier to $G = (B, L, K, T, S, R, E, H, C, P, N)$. The only selector-budget coordinate is B, and the only shared observation and regular-witness coordinates are S and R.

Apply the NameCert obligation surface. It exposes selector-budget data through B, shared StreamName data through S, and RegSeqRat witness data through R before either the ledger row L or tail-witness row T is consumed.

The route rows in C replay exactly this order, while H preserves the same coordinates under componentwise transport. No second selector or hidden window coordinate remains in the carrier, so the selector route is exact. □

Theorem 905.4 (Real Cauchy witness grid Real handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$B, S, R, L, T, K, E, H, C, P, N.$$

In particular, the consumer first reads the selector-budget row, the shared StreamName and RegSeqRat rows, the ledger and tail-witness rows, and the completion-seal classifier before receiving the RealUp seal or readback row.

Proof. Start with selector exactness. By Definition 905.3, the selected window and RegSeqRat witness are fixed by B, S, R before the ledger row L or tail-witness row T can be used.

Read the sibling packets. The witness ledger of chapter 902 supplies the diagonal budget and synchronizer route, while the tail-witness envelope of chapter 898 supplies the finite tail readback route.

The completion-seal classifier of chapter 900 is then reached through K, and the RealUp seal or readback row is E. The structural rows H, C, P, N preserve and name this route, so no Real-completion consumer receives a different handoff. □

Theorem 905.5 (Real Cauchy witness grid non-escape)**.** *An accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier exports only the displayed rows* $B, L, K, T, S, R, E, H, C, P, N$*. It exports no quotient stream equality, host real equality, selected external limit, countable-choice row, hidden selector enlargement, unlisted tail witness, or ambient Real completeness theorem. Lean-facing work for this packet must provide* $\mathsf{Nontrivial\ RealCauchyWitnessGrid}^{\uparrow}$ *with an explicit witness pair and* $\mathsf{FieldFaithful\ RealCauchyWitnessGri}$ *by constructor-field injection.*

Proof. Project the carrier and apply the NameCert obligation theorem. Every exported row is one of the selector-budget, witness-ledger, completion-seal classifier, tail-witness, StreamName, RegSeqRat, RealUp, transport, route, provenance, and local naming rows.

The selector-exactness theorem prevents replacement of B, S, R by another selector source. The Real handoff theorem prevents replacement of L, T, K, E by a host limit or a detached RealUp output.

The excluded objects have no coordinate in the tuple. Therefore no accepted consumer can recover them from the grid, and the Lean-facing nontriviality and field-faithfulness tasks must be witnessed over the displayed fields. □

Theorem 905.6 (Real Cauchy witness grid ledger-tail pullback)**.** *Let*

$$G = (B, L, K, T, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier. The* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row* L *and the* $\mathsf{CauchyTailWitnessEnvelop}$ *row* T *read the same selector-budget row* B*, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* S*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *witness rows* R*. Thus no accepted grid can assign one selector route to the ledger side and a different selector route to the tail side.*

Proof. Project the accepted grid to the displayed carrier tuple. The only selector-budget coordinate is B, the only StreamName-window coordinate is S, and the only RegSeqRat witness coordinate is R.

Apply selector exactness from Definition 905.3. It fixes the route $B \longmapsto (S, R) \longmapsto (L, T)$ before either L or T is consumed.

The route row C and transport row H replay those same coordinates into the sibling packets. Since the carrier contains no second selector, hidden window, or unlisted RegSeqRat witness, the ledger and tail rows pull back to the same B, S, R selector surface. □

Theorem 905.7 (Real Cauchy witness grid completion-seal selector lock)**.** *Let*

$$G = (B, L, K, T, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier. The completion-seal classifier row* K *is reachable only after the shared selector-ledger-tail route*

$$B \longmapsto (S, R) \longmapsto (L, T)$$

has been read and authenticated by the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows* H, C, P, N*. In particular,* K *cannot be reached from a detached completion seal, host real equality, selected external limit, countable-choice row, or ambient completeness principle.*

Proof. Start with the ledger-tail pullback of Definition 905.6. It says that L and T already share the selector-budget, StreamName, and RegSeqRat rows B, S, R.

Use the Real handoff theorem, Definition 905.4. A Real-completion consumer reads the selector, shared rows, ledger, and tail-witness rows before the completion-seal classifier row K and RealUp row E are available.

Finally apply non-escape from Definition 905.5. The forbidden bypasses have no coordinate in G, and the structural rows H, C, P, N can authenticate only the displayed route. Hence the completion-seal classifier is locked behind the shared selector-ledger-tail route. □

Theorem 905.8 (Real Cauchy witness grid obligation closure)**.** *The accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *packet closes its local obligation surface over* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{CauchyWitnessLedger}^{\uparrow}$*,* RealCompletionSealClassifierUp*,* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Carrier admission, selector exactness, Real handoff, non-escape, ledger-tail pullback, and completion-seal selector locking are all witnessed by the same finite grid*

$$G = (B, L, K, T, S, R, E, H, C, P, N).$$

Proof. Carrier admission and the local NameCert surface are supplied by Definition 905.1 and Definition 905.2. They expose the displayed selector, ledger, classifier, tail, StreamName, RegSeqRat, RealUp, transport, route, provenance, and naming rows.

Selector exactness is supplied by Definition 905.3, and the Real handoff is supplied by Definition 905.4. These two clauses bind the selector-budget and Real-facing reads to the same finite route.

Non-escape is supplied by Definition 905.5. The ledger-tail pullback of Definition 905.6 and the completion-seal selector lock of Definition 905.7 add the two obligation-level locks needed by downstream Real-completion witness packets.

All six clauses use the same rows $B, L, K, T, S, R, E, H, C, P, N$. Therefore the chapter's obligation surface is finite and local to the named BEDC packets, with no quotient stream, host real equality, selected external limit, countable choice, hidden selector enlargement, or ambient completeness theorem. □

Theorem 905.9 (Real Cauchy witness grid standard bridge)**.** *Every standard-facing read of an accepted* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *carrier factors through the displayed selector, ledger, completion-seal classifier, tail,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, transport, route, provenance, and* NameCert *rows*

$$G = (B, L, K, T, S, R, E, H, C, P, N).$$

The bridge exposes only that finite witness grid. It exports no quotient stream, selected external limit, countable choice, host real equality, ambient completeness row, hidden selector enlargement, or off-grid Real-completion witness.

Proof. Project the accepted carrier to $G = (B, L, K, T, S, R, E, H, C, P, N)$. The NameCert obligation theorem exposes exactly the selector, ledger, classifier, tail, StreamName, RegSeqRat, RealUp, transport, route, provenance, and naming coordinates.

Apply selector exactness and Real handoff. The selector exactness theorem fixes the route from B through S, R to L, T, while the Real handoff theorem places K, E, H, C, P, N on the same public-facing path before any Real-completion consumer receives a read.

Next use the ledger-tail pullback and completion-seal selector lock. They force the ledger row and tail row to share the same selector surface and keep the completion-seal classifier behind the authenticated selector-ledger-tail route.

Finally apply non-escape and obligation closure. The non-escape theorem removes quotient streams, selected external limits, countable choice, host real equality, hidden selector enlargements, off-grid Real witnesses, and ambient completeness rows from the export surface. The obligation-closure theorem repacks the remaining rows as the finite public grid, so every standard-facing read factors through the displayed carrier. □

Falsifiable prediction: If the witness grid is atomic, then every Real-completion diagonal consumer factors through the displayed selector-budget and completion-seal grid before reaching RealUp; a consumer bypassing both sibling ledgers refutes the chapter.

Independence witness: Nearest siblings: cauchywitnessledger, realcompletionsealclassifier, regularcauchyselectorbudget, cauchytailwitnessenvelope. The grid is not bijective to any one sibling because it simultaneously stores selector rows, ledger rows, seal-classifier rows, and tail-witness rows, while each sibling omits at least one of those obligation rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyWitnessGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ packet is generated from a selector budget, witness ledger, completion-seal classifier, tail-witness envelope, shared StreamName rows, RegSeqRat witness rows, RealUp seal rows, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCauchyWitnessGrid}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 905.1, Definition 905.2, Definition 905.3, Definition 905.4, Definition 905.5, Definition 905.6, Definition 905.7, Definition 905.8, and Definition 905.9 bind the finite diagonal witness grid for the public Real-completion consumer route and the mature standard-facing read through the displayed selector, non-escape surface, obligation closure, ledger, classifier, tail, StreamName, RegSeqRat, RealUp, transport, route, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCauchyWitnessGrid}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.RealCauchyWitnessGridUp.taste gate`

Lean target: `BEDC.Derived.RealCauchyWitnessGridUp.RealCauchyWitnessGridTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient streams, host real equality, selected external limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: The paper closure axis is complete for this finite witness grid; Lean bridge checking remains an independent formal-axis obligation.

Chapter 906

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailBudget}^{\uparrow}$

The $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ packet realizes the finite observation-service side of the Real and Completion route of chapter 9238. It binds a regular Cauchy tail threshold to the finite budget gate of chapter 903, the witness envelope of chapter 898, and the seal synchronizer of chapter 886 before a $\mathsf{Real}^{\uparrow}$ completion-facing consumer reads the tail route. The packet supplies a finite tail-threshold, observation-budget, window-service, dyadic-readback, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal surface; it does not assert Real completeness.

Falsifiable prediction: Removing the tail-budget row permits a $\mathsf{Real}^{\uparrow}$ seal consumer to read a regular Cauchy tail without a displayed finite observation service; removing the observation-budget row breaks the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Independence witness: Nearest siblings: regular-cauchy-tail-fiber, cauchy-tail-witness-envelope, observationbudgetlimiter, cauchy-seal-budget-synchronizer, and real-completion-seal-classifier. None is bijective to $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ because the tail-fiber packet compares two source tails without a service cap, the witness envelope stores selector and agreement rows without the shared budget threshold, the limiter stores capped observations without a regular Cauchy tail row, the synchronizer stores seal-budget compatibility without the RegSeqRat readback service, and the completion-seal classifier stores equivalence of seal reads rather than the finite tail-service row itself.

Definition 906.1 (Regular Cauchy tail-budget carrier)**.** *A* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular Cauchy tail is being served;*
2. *T is the shared tail-threshold row naming the first accepted tail index for the service request;*
3. *O is the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row that bounds the finite observation service available after T is displayed;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window-service ledger admitted by O for the tail threshold;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger read through the windows in W;*

6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed tail threshold, finite windows, and dyadic observations;*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, T, O, W, D, Q have been consumed;*
8. *H records the componentwise* hsame *rows for the source, threshold, budget limiter, windows, dyadic ledger, readback, seal, replay, provenance, and local naming fields;*
9. *C records the displayed* Cont *routes from the regular source to the tail threshold, from the threshold to the observation budget, from the budget to window service, from window service to dyadic observation, from dyadic observation to regular readback, and from readback to the seal;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ *row.*

The carrier accepts only packets whose regular Cauchy tail threshold is served by a displayed finite observation budget before any $\mathsf{Real}^{\uparrow}$ *seal row is read.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailBudgetUp.taste gate`

Theorem 906.2 (Regular Cauchy tail-budget NameCert obligations)**.** *For the carrier of Definition 906.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$B = (R, T, O, W, D, Q, S, H, C, P, N),$$

tail-threshold exposure through T, observation-budget exposure through O, finite window service through W, dyadic observation through D, regular readback through Q, seal handoff through S, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier. The rows $R, T, O, W, D, Q, S, H, C, P, N$ are the only fields of the accepted packet.

Read the analytic service rows in order. The regular source R supplies the tail to be served, T fixes the shared tail threshold, and O supplies the finite observation-budget gate before any window is exposed.

The window, dyadic, and readback rows then form the service body. The rows W, D, Q display exactly how the finite observations are read back into a $\mathsf{RegSeqRat}^{\uparrow}$ row before S is available.

Finally H, C, P, N transport, replay, package, and name the same route. No additional row can export quotient streams, host real equality, selected limits, countable choice, or ambient completeness. □

Theorem 906.3 (Regular Cauchy tail-budget observation gate)**.** *Every accepted* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer read of a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet factors through the finite observation route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S.$$

In particular, the regular Cauchy tail may not be read as a $\mathsf{Real}^{\uparrow}$ *seal unless the tail threshold and the observation-budget limiter have both been displayed.*

Proof. Start with a source-facing read. The only regular source row is R, and the only tail threshold attached to that source is T.

Use the budget gate next. The observation service is controlled by O, so a consumer cannot reach the finite window row W before the limiter row has been displayed.

Read W, D, Q through the displayed routes in C. These rows provide the finite window service, dyadic observation ledger, and regular readback handoff. The seal row S is reached only after this route has been replayed.

The structural rows H, C, P, N authenticate that same route and do not create an alternate observation channel. Thus every consumer read factors through the displayed finite observation gate. □

Theorem 906.4 (Regular Cauchy tail-budget seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet receives only the displayed surface*

$$R, T, O, W, D, Q, S, H, C, P, N.$$

It receives no quotient stream, no host real equality, no selected host limit, no countable-choice selector, and no ambient completeness theorem. The seal row is a finite handoff from a budgeted regular Cauchy tail service, not a proof that all Cauchy names converge.

Proof. Inspect the carrier coordinates. The only seal-facing row is S, and it is positioned after R, T, O, W, D, Q in the displayed service route.

Apply the observation-gate theorem. Any $\mathsf{Real}^{\uparrow}$ consumer must first pass through the shared tail threshold, observation-budget limiter, finite windows, dyadic observations, and regular readback.

The excluded analytic objects have no coordinate in the packet. Since the route contains only finite carrier rows, componentwise transports, displayed continuations, package provenance, and a local naming row, the seal cannot export quotient equality, a selected limit, countable choice, or ambient completeness. □

Theorem 906.5 (Regular Cauchy tail-budget obligation package)**.** *An accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet assembles the finite tail-service obligation package*

$$R, T, O, W, D, Q, S, H, C, P, N.$$

The package consists of carrier admission for the regular source row R, tail-threshold exposure T, observation-budget exposure O, finite $\mathsf{StreamName}^{\uparrow}$ *window service W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *seal handoff S, componentwise* hsame *rows H, displayed* Cont *routes C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ *rows N. It does not include quotient streams, host real equality, selected-limit existence, countable choice, or ambient completeness.*

Proof. Start with Definition 906.1. The accepted carrier exposes exactly the rows $R, T, O, W, D, Q, S, H, C, P, N$, so every obligation in the package must be read from those coordinates.

Apply Definition 906.2. This supplies carrier admission, the local naming row, the componentwise transport rows, the displayed continuation routes, and the package provenance for the same finite carrier.

Now use Definition 906.3. It orders the tail threshold, observation budget, window service, dyadic observation, and regular readback rows as one finite service route before any seal-facing consumer can read S.

Finally apply Definition 906.4. Since the seal is only a handoff from the displayed route and the excluded analytic objects have no carrier coordinate, the package closes over the finite tail-service rows and exports no quotient stream, host real equality, selected limit, countable-choice selector, or ambient completeness theorem. □

Theorem 906.6 (Regular Cauchy tail-budget window-budget compatibility)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. Every served* $\mathsf{StreamName}^{\uparrow}$ *window row in* W *and every* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row in* D *is admitted through the same* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row* O *and the same tail-threshold row* T*. Thus the finite window service and the dyadic observation ledger share the single budgeted route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D.$$

Proof. Project the accepted carrier. The only rows available for tail threshold, observation budget, window service, and dyadic observation are T, O, W, D.

Apply Definition 906.3. It places every $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ consumer read after R, T, O and before the readback and seal rows.

Apply Definition 906.4 to exclude a second observation channel. Since no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient completeness row exists in the packet, the served windows and dyadic observations can only be read through the displayed T, O budget route. □

Theorem 906.7 (Regular Cauchy tail-budget readback factorization)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *factors through the finite chain*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q,$$

with H *transporting the component rows,* C *displaying the continuation routes,* P *recording package provenance, and* N *certifying the local name for that same chain.*

Proof. Begin with Definition 906.2. It identifies Q as the regular readback row assembled from the displayed tail threshold, finite windows, and dyadic observations.

Use Definition 906.6 for the middle service. The rows W and D have already passed through the same T, O budget route before Q can be read.

Finally read the structural rows. The transports in H, the continuations in C, the provenance row P, and the local name N authenticate the same R, T, O, W, D, Q factorization and provide no separate readback service. □

Theorem 906.8 (Regular Cauchy tail-budget downstream lattice)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. Any downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *consumer of* B *must share the single route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S,$$

with componentwise hsame *transport through H and* Cont *replay through C preserving the same tail-threshold row T and observation-budget row O for the window, dyadic, readback, and seal surfaces.*

Proof. Begin with the observation gate. It places every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ read after the source row R, tail threshold T, and observation budget O, and before the seal row S can be consumed.

Use window-budget compatibility for the middle of the route. The finite windows in W and dyadic observations in D are admitted through the same threshold and budget rows, so a downstream consumer cannot choose a second budget service for those observations.

Apply readback factorization. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q is assembled from the displayed R, T, O, W, D chain and is authenticated by the same H, C, P, N structural rows.

Finally apply seal non-escape. Since the seal row S is only a finite handoff after the displayed readback and no carrier coordinate stores quotient streams, selected host limits, host real equality, countable choice, or ambient completeness, the $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{RealDiagonalCompletion}^{\uparrow}$ consumers share the same budget lattice. □

Theorem 906.9 (Regular Cauchy tail-budget tail-threshold service exactness)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. Every served tail read of R is exposed only through the route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S.$$

In particular, neither the finite window row W nor the seal row S can be read before the tail-threshold row T has been displayed.

Proof. Project the carrier of Definition 906.1. The only source row is R, the only tail-threshold row attached to that source is T, and the service rows following it are O, W, D, Q, S.

Apply Definition 906.3. It forces each served $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ read to pass through the displayed route from R to S.

Use Definition 906.6 for the middle service rows. The window and dyadic ledgers are admitted only after the same threshold and budget rows have been read.

Finally apply Definition 906.7 and Definition 906.4. The readback row factors through T, O, W, D, and the seal row consumes that displayed readback rather than opening an earlier window or seal-facing route. □

Theorem 906.10 (Regular Cauchy tail-budget observation-budget gate totality)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. Every consumer route landing in* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *passes through the same* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row O. Thus all window service, dyadic observation, regular readback, and seal handoff rows share the single budget gate*

$$R \longmapsto T \longmapsto O.$$

Proof. Start from the carrier definition. The packet has one displayed observation-budget limiter row, namely O, and the continuation record C routes the tail threshold to that row before any service body is available.

Apply Definition 906.3. It covers the four consumer surfaces named in the statement and places each of them after R, T, O in the finite route.

Use Definition 906.6. It identifies O as the shared gate for W and D, so the downstream readback row Q cannot obtain its displayed observations from a second budget row.

Now apply Definition 906.7 and Definition 906.4. The readback and seal rows consume the same budgeted service route, and no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient completeness row supplies an alternate consumer route. □

Theorem 906.11 (Regular Cauchy tail-budget RegSeqRat readback seal handoff)**.** *Let*

$$B = (R, T, O, W, D, Q, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet. The row Q is the unique* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface for the displayed window row W and dyadic ledger row D. The seal row S may consume only this displayed readback surface together with the structural rows H, C, P, N.*

Proof. Inspect Definition 906.1. The packet contains one regular source row R and one regular readback row Q, while W and D are the only window-service and dyadic-observation rows feeding that readback.

Apply Definition 906.6. It places both W and D under the same T, O service route before any readback can be formed.

Apply Definition 906.7. The row Q is exactly the displayed factor through R, T, O, W, D, with H, C, P, N authenticating the same chain.

Finally use Definition 906.4. Since S is only a finite seal handoff after the displayed readback and no other analytic row is present, a seal-facing consumer can read S only through Q and the accompanying structural rows. □

Theorem 906.12 (Regular Cauchy tail-budget completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet may read the seal row S only as the finite budgeted readback boundary*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S,$$

together with the displayed H, C, P, N rows. The consumer receives no quotient stream, no selected host limit, no countable-choice selector, no host real equality, and no ambient completeness principle.

Proof. Apply Definition 906.7. It makes the readback row Q a finite factor through the source, tail threshold, budget limiter, windows, and dyadic ledger.

Then apply Definition 906.4. The seal row is available only after that readback route has been consumed, and the excluded analytic objects have no carrier coordinate.

The displayed rows H, C, P, N record transport, continuation, provenance, and naming for the same boundary. Hence a completion-facing consumer can consume S as this finite budgeted handoff and not as a Real-completeness theorem. □

Theorem 906.13 (Regular Cauchy tail-budget obligation-closure package)**.** *For an accepted* RegularCauchyTailBud *packet*

$$B = (R, T, O, W, D, Q, S, H, C, P, N),$$

the carrier of Definition 906.1, the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ *obligations of Definition 906.2, the observation gate of Definition 906.3, the seal non-escape boundary of Definition 906.4, the*

window-budget compatibility of Definition 906.6, the readback factorization of Definition 906.7, the tail-threshold service exactness of Definition 906.9, the observation-budget gate totality of Definition 906.10, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback seal handoff of Definition 906.11, and the completion boundary of Definition 906.12 form the paper-side obligation package over the displayed fields* $R, T, O, W, D, Q, S, H, C, P, N$.

The package is fieldwise: the regular source, tail threshold, observation budget, windows, dyadic observations, readback, seal, transports, continuation routes, provenance, and local name remain separate coordinates throughout every Real-facing and Completion-facing read.

Proof. Project the carrier of Definition 906.1. The eleven rows $R, T, O, W, D, Q, S, H, C, P, N$ are the complete coordinate list available to any local certificate or downstream consumer.

Apply Definition 906.2 for the local certificate surface, then apply Definition 906.3 and Definition 906.4 for the first route boundary. These theorems identify the same fields without adding quotient streams, host real equality, selected limits, choice, or ambient completeness.

Next use Definition 906.6, Definition 906.7, Definition 906.9, and Definition 906.10. They account for the finite service route from the regular source through the threshold, budget, windows, dyadic ledger, and readback.

Finally apply Definition 906.11 and Definition 906.12. The seal and completion-facing boundary consume the same displayed readback route, so the obligation package closes over exactly the listed carrier fields. □

Theorem 906.14 (Regular Cauchy tail-budget field-faithful boundary)**.** *Any later* FieldFaithful RegularCauchyTailB *instance for the carrier of Definition 906.1 must preserve the eleven displayed fields* $R, T, O, W, D, Q, S, H, C, P, N$ *separately. In particular, the tail threshold* T*, observation budget* O*, finite windows* W*, dyadic observations* D*, regular readback* Q*, and seal* S *cannot be collapsed into a single parameter echo, nor may the transport, continuation, provenance, or local naming rows replace one another.*

Thus a faithful instance must expose constructor-field injection for the regular source, tail threshold, observation budget, finite windows, dyadic observations, regular readback, seal, transport, continuation, package, and local naming fields as separate facts.

Proof. Read the constructor shape in Definition 906.1. Each coordinate is named by its own row and has its own role in the displayed finite route.

Use the route theorems to separate the service fields. The observation gate, window-budget compatibility, readback factorization, tail-threshold service exactness, observation-budget gate totality, and readback seal handoff assign different positions to T, O, W, D, Q, S in the route

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S.$$

Use the non-escape and completion-boundary theorems to separate the structural fields. The rows H, C, P, N authenticate transport, route replay, provenance, and local naming for the same packet; none of them supplies a replacement tail threshold, observation budget, window ledger, dyadic ledger, readback row, or seal.

Therefore any field-faithful certificate for this carrier must project and preserve the eleven rows one by one. A single echo parameter cannot witness the stated finite observation-service boundary. □

Theorem 906.15 (Regular Cauchy tail-budget threshold totality)**.** *For an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet*

$$B = (R, T, O, W, D, Q, S, H, C, P, N),$$

the finite tail threshold T is the unique entry row through which any $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer may reach the served tail. The consumer may read W, D, Q, S only after the displayed route*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q \longmapsto S$$

has exposed the same threshold, observation-budget limiter, and local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ *provenance.*

Proof. Project the carrier of Definition 906.1. The row T is the only threshold coordinate attached to the regular source row R.

Apply Definition 906.9. It binds every served tail request to the displayed threshold before any window, dyadic observation, readback, or seal row is available.

Then apply Definition 906.10. The observation-budget row O is read after T, so the finite service route cannot bypass the threshold by selecting W, D, Q, or S directly.

Finally use Definition 906.7. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows all factor through the same $R, T, O, W, D, Q, S, H, C, P, N$ surface, leaving no second tail-threshold entry row. □

Theorem 906.16 (Regular Cauchy tail-budget selector handoff)**.** *Every selector-facing handoff from an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumers factors through the displayed rows T, O, W, D, Q before the seal row S is exposed. The handoff is therefore a finite tail-selector service over the carrier of Definition 906.1, not a countable-choice selector, quotient stream, selected host limit, or ambient completion theorem.*

Proof. Start with Definition 906.6. It binds the finite windows W to the threshold T and observation-budget row O.

Use Definition 906.9 to make the selector-facing tail request pass through the same threshold. The selector receives a displayed finite service row rather than an unbounded family of choices.

Next apply Definition 906.7 and Definition 906.11. These theorems route W through D and Q before S becomes available to a Real-facing consumer.

The non-escape clause is supplied by Definition 906.4. Since the carrier has no countable-choice selector, quotient stream, selected host limit, or ambient completion coordinate, the handoff remains finite and fieldwise. □

Theorem 906.17 (Regular Cauchy tail-budget Real-seal consumer scope)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet reads the Real-seal row S only as the endpoint of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *service route*

$$R, T, O, W, D, Q, S, H, C, P, N.$$

The consumer receives no host real equality, quotient stream, selected limit, countable-choice selector, trichotomy row, or ambient Real completeness principle beyond the finite carrier surface.

Proof. Project the carrier of Definition 906.1. The seal coordinate S is displayed only after the source, threshold, observation budget, finite windows, dyadic observations, and regular readback rows.

Apply Definition 906.11. It identifies the Real-facing seal handoff as a consumer of the regular readback row Q, not as an independent Real carrier.

Next use Definition 906.12. It confines completion-facing reads to the same $R, T, O, W, D, Q, S, H, C, P, N$ packet and keeps the seal boundary separate from Real completeness.

Finally apply Definition 906.13. The carrier rows, NameCert obligations, readback factorization, threshold service, observation-budget gate, handoff, and completion boundary compose as one fieldwise package, so the Real-seal consumer cannot extract host equality, quotient streams, selected limits, choice, trichotomy, or ambient completeness. □

Theorem 906.18 (Regular Cauchy tail-budget promoted obligation closure)**.** *The carrier admission of Definition 906.1, the local naming obligations of Definition 906.2, the tail-threshold service of Definition 906.9, the observation-budget gate of Definition 906.10, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of Definition 906.7, and the Real seal route of Definition 906.11 form one obligation-closure package over the displayed rows* $R, T, O, W, D, Q, S, H, C, P, N$*.*

Proof. Project the carrier of Definition 906.1. The rows $R, T, O, W, D, Q, S, H, C, P, N$ are the only fields available to the local naming surface.

Apply Definition 906.2 for carrier admission and local certificate exposure. Then use Definition 906.9 and Definition 906.10 to route every served tail request through the shared threshold and observation budget.

Finally use Definition 906.7 and Definition 906.11. The readback row and the seal row consume the same finite budgeted service, so the listed components close as one obligation package. □

Theorem 906.19 (Regular Cauchy tail-budget window readback totality)**.** *Every finite budget request admitted by a* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet has a displayed window and readback route through the packet's budget and dyadic rows. Concretely, a request reaches* Q *only along*

$$R \longmapsto T \longmapsto O \longmapsto W \longmapsto D \longmapsto Q,$$

with S, H, C, P, N *recording the seal-facing endpoint, transports, routes, provenance, and local naming row.*

Proof. Start from the threshold service. By Definition 906.9, a served request first reads the displayed tail threshold T attached to the regular source row R.

Next apply Definition 906.10. The observation budget O admits the request before any window or dyadic row can be consumed.

Use Definition 906.6 and Definition 906.7. These bind the finite windows W and dyadic observations D to the readback row Q, giving the displayed route for every finite budget request. □

Theorem 906.20 (Regular Cauchy tail-budget Real seal non-escape package)**.** *The* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{RealDiagonalCompletion}^{\uparrow}$ *consumers of an accepted* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *packet obtain only the finite seal route through* $R, T, O, W, D, Q, S, H, C, P, N$*. They obtain no quotient stream, countable-choice selector, selected host limit, host real equality, trichotomy row, or ambient completeness oracle.*

Proof. Use Definition 906.17. It makes the Real-facing consumer read S only after the source, threshold, observation budget, finite windows, dyadic observations, and regular readback rows.

Apply Definition 906.12. Completion-facing consumers receive the same finite route, not a separate completeness coordinate.

Finally use Definition 906.4. The excluded analytic objects have no coordinate in the carrier, so the Real seal package cannot export a quotient stream, choice principle, selected limit, host equality, trichotomy, or ambient oracle. □

Theorem 906.21 (Regular Cauchy tail-budget obligation-rank surface)**.** *For* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$*, the rows for carrier admission, tail-threshold exposure, observation-budget gating, finite window service, dyadic observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, local* NameCert*,* Nontrivial *witnessing, and* FieldFaithful *row faithfulness are the complete obligation-rank surface of the finite packet*

$$R, T, O, W, D, Q, S, H, C, P, N.$$

No additional analytic row, hidden limit selector, host equality row, quotient stream row, or ambient completeness oracle is part of the obligation rank.

Proof. Project the carrier in Definition 906.1. It exposes exactly the source, threshold, observation budget, finite windows, dyadic observations, regular readback, seal, transport, continuation, provenance, and local naming rows.

Apply the NameCert obligations of Definition 906.2 and the promoted package of Definition 906.18. These bind carrier admission, local naming, transport, continuation replay, package provenance, threshold service, observation gating, and the nontrivial and field-faithful witness rows to the same displayed packet.

Use Definition 906.19 to force every finite budget read through T, O, W, D, Q. Then apply Definition 906.20 for the Real seal endpoint S and its non-escape boundary.

The listed rows therefore exhaust the obligation-rank surface. Anything outside them has no carrier coordinate and cannot be supplied by the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$ route. □

The Lean TasteGate.lean contract for this chapter requires `Nontrivial RegularCauchyTailBudgetUp` and `FieldFaithful RegularCauchyTailBudgetUp` instances whose witnesses include an explicit `witness_pair` and constructor-field injection proofs for the regular source, tail threshold, observation budget, finite windows, dyadic observations, regular readback, seal, transport, continuation, package, and local naming fields.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, shared tail-threshold row, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ window-service ledger, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 906.1, Definition 906.2, Definition 906.18, Definition 906.19, Definition 906.20, and Definition 906.21 specify the finite regular Cauchy tail-budget packet whose displayed obligation-rank, budget-window readback, and Real-seal non-escape rows form the public export surface, with the checked TasteGate instance `BEDC.Derived.RegularCauchyTailBudget`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBudget}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RegularCauchyTailBudgetUp.RegularCauchyTailBudgetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Real completeness, quotient equality of streams, host real equality, selected-limit existence, countable choice, or bridge checking for $\mathsf{RegularCauchyTailBudget}^{\uparrow}$.

Upgrade path: Mature paper closure is recorded here; machine bridge checking remains a separate verification-axis obligation for the finite tail-budget public route.

Chapter 907

A Concrete Naming Certificate for $\mathsf{PiApplicationAdequacyRoute}^{\uparrow}$

The $\mathsf{PiApplicationAdequacyRoute}^{\uparrow}$ packet realizes the application-facing route in chapter 9383 as finite NameCert data. It refines the candidate-mediated surface of chapter 758 by separating Pi-candidate function evidence, domain-candidate argument evidence, and application-candidate handoff before a closed typed application may be read through chapter 562. The acceptance-facing replay is audited beside chapter 547, so the route is a packet over displayed rows rather than an unconditional normalization theorem.

Definition 907.1 (Pi application adequacy route carrier)**.** *A* $\mathsf{PiApplicationAdequacyRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (F, A, P, D, K, S, C, N, R, H, Q)$$

with the following displayed rows:

1. *F is the Pi-candidate function evidence row, naming the function candidate and the Pi-head evidence consumed by the application route;*
2. *A is the domain-candidate argument evidence row, naming the argument, its domain candidate, and the candidate read used by the Pi body;*
3. *P is the application candidate row, recording the handoff from the Pi-head evidence and domain-candidate argument evidence to the candidate assigned to the application;*
4. *D is the closed typed application row, keeping the function, argument, application term, and typing ledger in one displayed packet;*
5. *K is the step-strong-normalization row read from the candidate evidence already displayed in F, A, P, D;*
6. *S is the no-infinite-reduction row projected from K and not from an external evaluator;*
7. *C is the displayed* Cont *route from the Pi candidate and domain candidate rows through the application candidate row to the closed typed application row;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PiApplicationAdequacyRoute}^{\uparrow}}$ *row;*
9. *R records* Pkg *provenance over chapter 9383, chapter 758, chapter 562, chapter 547,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*

10. *H stores componentwise* hsame *transports for the Pi-head evidence, argument evidence, application candidate, typed application, step-SN, and no-infinite-reduction rows;*
11. *Q is the query boundary that records which application-normalization question is answered by this finite route and refuses any stronger global-normalization query.*

The carrier has no coordinate for evidence-free strong normalization, unrestricted application adequacy, quotient equality over reduction traces, a host evaluator, a selected normalization strategy, or ambient consistency.

Lean4 (definition) `BEDC.Derived.PiApplicationAdequacyRouteUp.PiApplicationAdequacyRouteUp`
Lean4 ✓ `BEDC.Derived.PiApplicationAdequacyRouteUp.PiApplicationAdequacyRouteTasteGate single carrier alignment`

Theorem 907.2 (Pi application adequacy route NameCert obligations)**.** *Every accepted* PiApplicationAdequacyRou *packet binds exactly the obligation surface displayed in Definition 907.1: Pi-candidate function evidence through F, domain-candidate argument evidence through A, application candidate handoff through P, closed typed application through D, step strong normalization through K, no-infinite-reduction projection through S, displayed route through C, local naming through N, provenance through R, transport through H, and query boundary through Q.*

Proof. Project the carrier $Z = (F, A, P, D, K, S, C, N, R, H, Q)$. The positive application route begins with the Pi-candidate function row and the domain-candidate argument row, so a consumer cannot read the application candidate before both F and A have been exposed.

The handoff row P then supplies the application candidate used by the closed typed application row D. The strong-normalization and no-infinite-reduction rows K, S are downstream from that candidate-mediated handoff, not independent termination assertions.

The support rows preserve the same finite surface. The route row C, local NameCert row N, provenance row R, componentwise transport row H, and query boundary Q name and restrict only the displayed coordinates, so the listed rows exhaust the local NameCert obligations. □

Theorem 907.3 (Pi application adequacy route nonescape)**.** *Every closed application strong-normalization read exported by an accepted* PiApplicationAdequacyRoute$^{\uparrow}$ *packet factors through the displayed rows*

$$F, A, P, D, K, S, C, N, R, H, Q.$$

The read may be consumed by the candidate-mediated route of chapter 758 and by the endpoint discipline of chapter 562, with acceptance replay bounded by chapter 547. It cannot be reread as evidence-free global strong normalization, unconditional application adequacy, quotient equality over reductions, host evaluation, or ambient consistency.

Proof. Start from a closed application strong-normalization read. The only step-SN coordinate is K, and its route passes through F, A, P, D before the no-infinite-reduction projection S can be consumed.

Use the continuation row C and componentwise transport row H to replay the same candidate-mediated path under accepted packet agreement. Provenance R and the local NameCert row N then present that route to the sibling candidate-evidence, typed-normal-form, and kernel-acceptance audit surfaces.

Finally inspect the query boundary Q and the absence of excluded coordinates. Since the packet stores no host evaluator, quotient trace equality, global-normalization row, unrestricted adequacy row, or ambient consistency row, no consumer can escape the displayed application route. □

Falsifiable prediction: If the route is atomic, a consumer cannot derive closed application strong normalization from componentwise function and argument SN without the Pi-candidate and domain-candidate rows.

Independence witness: metacicnormalizationaudit, candidateevidencesnroute, typedreductionnormalform, subjectreduction, kernelacceptanceauditwitness: PiApplicationAdequacyRouteUp is not bijective to these because it carries both Pi-head evidence and application-candidate handoff rows, while each sibling lacks at least one of those route coordinates.

The Lean-side `TasteGate.lean` obligation surface for $\mathsf{PiApplicationAdequacyRoute}^{\uparrow}$ consists of `Nontrivial PiApplicationAdequacyRouteUp` and `FieldFaithful PiApplicationAdequacyRouteUp`, with explicit witness-pair and field projection proofs over the displayed carrier rows.

Closure status for $\mathsf{NameCert}_{\mathsf{PiApplicationAdequacyRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{PiApplicationAdequacyRoute}^{\uparrow}$ packet is generated from Pi-candidate function evidence, domain-candidate argument evidence, application-candidate handoff rows, closed typed application rows, step-SN rows, no-infinite-reduction rows, displayed continuations, local naming, provenance, componentwise transport, and a query boundary.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PiApplicationAdequacyRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 907.1, Definition 907.2, and Definition 907.3 bind the finite candidate-mediated application adequacy route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PiApplicationAdequacyRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close evidence-free global strong normalization, unconditional application adequacy, quotient equality over reduction traces, host evaluation, ambient consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, application-candidate handoff exactness, no-infinite projection exactness, query-boundary refusal, provenance, and local NameCert rows.

Chapter 908

A Concrete Naming Certificate for $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$

The $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ packet realizes the Type IV finite-record schedule socket of section 9394.5 as a finite envelope over accepted schedule rows. It consumes the prefix and request-window discipline of chapter 539, the finite naming surface of chapter 126, and the Real-facing budget route of chapter 611. The envelope gives downstream Real and Completion consumers one packet that displays the schedule source, request window, refusal boundary, transport, route, provenance, and local name without reading observed prefixes as completed streams.

Falsifiable prediction: Any RealUp-facing schedule read from an accepted envelope factors through the displayed request-window and StreamName source rows; a read without those rows refutes the envelope.

Independence witness: Nearest siblings: countableobservationschedule, streamname, realobservationbudget, finiteobservationbudgetselector, and observationbudgetlimiter. None is bijective to $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ because the envelope has a joint refusal row and a consumer-handoff row tying the schedule source to the Real-facing window; no sibling alone carries both rows in one local NameCert packet.

Definition 908.1 (Observation schedule envelope carrier)**.** *An* $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, S, W, R, J, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{CountableObservationSchedule}^{\uparrow}$ *source packet supplying observed prefixes and finite request windows;*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *source row through which the schedule is addressed;*
3. *W is the Real-facing request-window row read by budget and completion consumers;*
4. *R is the joint refusal row blocking completed streams, future-law claims, hidden persistence, and unbounded observation schedules;*
5. *J is the consumer-handoff row tying the schedule source Q, StreamName row S, and Real-facing window W into one envelope;*
6. *H records componentwise* hsame *transports for Q, S, W, R, J, C, P, N;*

7. *C records the displayed* Cont *routes that expose schedule prefixes, StreamName source names, Real-facing windows, refusal data, and the handoff row;*
8. *P records* Pkg *provenance over* $\mathsf{CountableObservationSchedule}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationScheduleEnvelope}^{\uparrow}}$ *row.*

The carrier is exactly this finite envelope. It exports no completed countable stream, no unobserved future row, no hidden causal-continuity principle, no quotient schedule equality, no selected Real value, and no ambient completion theorem.

Lean4 ✓ `BEDC.Derived.ObservationScheduleEnvelopeUp.ObservationScheduleEnvelopeTasteGate single carrier alignment`

Theorem 908.2 (Observation schedule envelope NameCert obligations)**.** *For the carrier of Definition 908.1, the local* $\mathsf{NameCert}_{\mathsf{ObservationScheduleEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (Q, S, W, R, J, H, C, P, N),$$

schedule-source exposure, StreamName source accountability, Real-facing request-window exposure, joint refusal, consumer handoff, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, and non-escape for completed streams, future observations, hidden persistence laws, quotient schedule equality, selected Real values, and ambient completion.*

Proof. Carrier admission is projection from Definition 908.1. The rows $Q, S, W, R, J, H, C, P, N$ are all displayed coordinates of one finite BHist packet.

Schedule-source exposure reads Q through chapter 539. StreamName accountability reads S through chapter 126, while the Real-facing window W is the only route by which a budget consumer may ask for schedule data.

The refusal and handoff rows then bind the envelope. The refusal row R blocks completed streams and future-law data, and the handoff row J forces the consumer to carry Q, S, W together before the route is authenticated by H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ObservationScheduleEnvelopeUp.TasteGate.ObservationScheduleEnvelopeNameCert obligations`

Theorem 908.3 (Observation schedule envelope window transport)**.** *For an accepted* ObservationScheduleEnvelope *carrier $E = (Q, S, W, R, J, H, C, P, N)$, any displayed transport of the Real-facing request window preserves the same source schedule row Q, StreamName source row S, joint refusal row R, and consumer-handoff row J. Transport may replay the window through H and C, but it may not enlarge W, replace S by an unlisted source, or turn the observed prefix carried by Q into a completed stream.*

Lean4 ✓ `BEDC.Derived.ObservationScheduleEnvelopeUp.ObservationScheduleEnvelope window transport`

Proof. Project the carrier and read the componentwise transport row H. The row records transport for the schedule source, StreamName source, Real-facing window, refusal row, handoff row, route row, provenance row, and local name.

Replay the consumer route through C. A transported Real-facing read still reaches W only together with Q, S, R, J, so the same finite source and refusal boundary remain visible.

Apply non-escape from Definition 908.2. Since the envelope has no coordinate for window enlargement, an unlisted source, or a completed stream, transport cannot create any of those rows. □

Theorem 908.4 (Observation schedule envelope StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, *or* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ *packet reads the schedule through the same StreamName source row* S *and the same Real-facing window row* W*. The consumer receives the handoff row* J *with the schedule source* Q*, refusal row* R*, componentwise transport* H*, displayed routes* C*, provenance* P*, and local name* N*; it receives no independent schedule carrier and no completed stream.*

Proof. Begin with the sibling surfaces. The schedule row Q supplies finite prefixes and request windows, while the StreamName chapter supplies the named source row S through which the schedule is addressed.

For a Real-facing consumer, route the same source through W and J. The handoff row ties Q, S, W before budget selectors or limiters can read the schedule, so the consumer sees one envelope rather than separate replacement rows.

The refusal row and non-escape clause exclude the missing objects. A consumer can inspect the displayed envelope, but it cannot obtain an independent carrier, a completed stream, a future observation law, or ambient completion from the handoff. □

Theorem 908.5 (Observation schedule envelope non-escape)**.** *Every consumer of an accepted* $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ *packet factors through the finite rows*

$$Q, S, W, R, J, H, C, P, N.$$

No consumer may replace those rows by a completed countable stream, an unobserved future row, a hidden causal-continuity principle, quotient schedule equality, selected Real data, an independent observation-budget carrier, or an ambient completion theorem.

Proof. Read the carrier inventory in Definition 908.1. It lists exactly the schedule source, StreamName source, Real-facing request window, refusal row, handoff row, transports, routes, provenance, and local naming row.

Use the NameCert obligation surface to authenticate the inventory. Each consumer-facing read must pass through the displayed rows and then through H, C, P, N.

The excluded objects have no coordinate in the packet. Therefore a route that asks for a completed stream, a future observation, hidden persistence, quotient equality, selected Real data, an independent budget carrier, or ambient completion exits the accepted envelope. □

The Lean TasteGate surface for this chapter carries `Nontrivial ObservationScheduleEnvelopeUp` and `FieldFaithful ObservationScheduleEnvelopeUp` instances for the same finite envelope carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationScheduleEnvelope}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$ packet is generated from finite BHist rows for request windows, observation schedule, envelope boundary, budget rows, route rows, transports, provenance, and local naming. The envelope admits only displayed finite observation routes: Cont carries each request through schedule and budget rows before any downstream read can inspect a selected window. Componentwise hsame preserves the schedule, envelope, budget, and route inventory, while Pkg records the observation, StreamName, budget, Real, and completion-adjacent dependencies. The packet names a finite observation envelope and refuses completed streams or hidden persistence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservationScheduleEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the observation-schedule envelope carrier, route obligations, excluded-object boundary, and finite envelope surface for $\mathsf{ObservationScheduleEnvelope}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservationScheduleEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim completed streams, future observations, hidden persistence, quotient equality, selected Real data, independent budget carriers, or ambient completion.

Upgrade path: Obligation closure requires Lean encodings of the envelope carrier, schedule route, budget boundary, field-faithful surface, and non-escape theorem.

Chapter 909

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$

The $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$ interface packages the finite shared-threshold diagonal route used by the Real-completeness vision of chapter 9238. It consumes the tail-meet surface of chapter 560, the shared-modulus surface of chapter 462, the witness envelope of chapter 898, and the diagonal window coverage row used by the completion route. The packet records the diagonal threshold comparison as data, not as a host-selected limit, quotient stream, or ambient completeness theorem.

Definition 909.1 (RegularCauchyDiagonalMeet carrier)**.** *A* $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, M, E, W, Q, H, C, P, N)$$

with the following displayed coordinates:

1. *T is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row that exposes the paired regular-Cauchy tails through one tail threshold;*
2. *M is the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row that supplies the shared bound consumed by the tail route;*
3. *E is the* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *row that stores the selected window, finite-window envelope, tail comparison, tail-agreement seal, and RealUp readback;*
4. *W is the* $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$ *row that records the diagonal window coverage available to the Real-completion route;*
5. *Q is the shared diagonal threshold comparison row, read only after T, M, E, W have been displayed;*
6. *H stores componentwise* hsame *transports for the tail-meet row, modulus-meet row, witness envelope, diagonal coverage row, comparison row, provenance, and local name;*
7. *C stores the displayed* Cont *routes for reading the shared modulus, applying it to the tail meet, passing through the witness envelope, checking diagonal coverage, and exposing the comparison row;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$, $\mathsf{RealDiagonalWindowCoverage}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows, their transports, their continuation routes, package provenance, and local naming provenance.

Lean4 (definition) `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetUp`

Theorem 909.2 (RegularCauchyDiagonalMeet NameCert obligations)**.** *For the carrier of Definition 909.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}}$ *surface consists exactly of carrier admission for* $D = (T, M, E, W, Q, H, C, P, N)$*, tail-meet admission through* T*, shared-modulus admission through* M*, witness-envelope admission through* E*, diagonal-window coverage through* W*, shared-threshold comparison through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetNameCertObligations`

Proof. Project the displayed carrier. Its rows name the tail meet, shared modulus, witness envelope, diagonal coverage, shared comparison, transport, route, package, and local certificate.

Read the analytic rows in the order required by the Real-completeness diagonal route. The shared modulus row M supplies the bound consumed by T, the envelope row E records the finite witness route, and the coverage row W makes the diagonal window visible before Q is exposed.

The remaining rows are structural. The transports in H, routes in C, provenance row P, and local name N replay and identify the same finite packet, so the local NameCert surface has no coordinate for quotient streams, host real equality, selected limits, or ambient completeness. □

Theorem 909.3 (RegularCauchyDiagonalMeet shared-threshold factorization)**.** *Every Real-completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$ *carrier reads the diagonal threshold comparison through the finite route*

$$M \longmapsto T \longmapsto E \longmapsto W \longmapsto Q.$$

The route uses the shared-bound packet of chapter 462, the shared-tail packet of chapter 560, the witness envelope of chapter 898, and the diagonal-window coverage row described by chapter 9238.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetSharedThresholdFactorization`

Proof. Start with a Real-completion read of the comparison row. The carrier has exactly one comparison coordinate Q, and the continuation route to it passes through the rows M, T, E, W.

The shared-bound route is first read through M and then consumed by T. This is the finite meet route: the source moduli and tail windows are synchronized before any diagonal comparison is available.

Next the witness route is exposed through E, and the diagonal window coverage is exposed through W. Only after these rows have been displayed can Q be consumed. Thus every comparison read factors through the stated finite route, with H, C, P, N carrying transport, replay, provenance, and naming. □

Theorem 909.4 (RegularCauchyDiagonalMeet transport induction)**.** *Let an accepted* $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$ *carrier display the rows* M, T, E, W, Q *and the componentwise transport row* H *of Definition 909.1. If the shared modulus, tail meet, witness envelope, diagonal coverage, and comparison rows are transported componentwise along* H*, then every consumer route to the comparison row still factors through*

$$M \longmapsto T \longmapsto E \longmapsto W \longmapsto Q.$$

The transported packet therefore preserves the displayed diagonal route without identifying quotient streams, selecting limits, or asserting host RealUp equality.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetTransportInduction`

Proof. Induct over the finite continuation route displayed in the carrier. The base read is the shared modulus row M; transporting it by the corresponding component of H keeps the same shared-bound coordinate and supplies the first node of the route.

For the tail-meet, witness-envelope, and diagonal-coverage steps, use the next component of H together with the carrier rows T, E, W. Each step replaces only the visible row by an hsame-transported row while preserving the continuation order recorded by C.

At the final comparison step, apply the shared-threshold factorization theorem above. Since every transported read has already replayed M, T, E, W, the transported comparison row Q is still reached through the same displayed route. No step introduces a quotient stream equality, selected limit, or host equality coordinate, because those objects are absent from the carrier. □

Theorem 909.5 (RegularCauchyDiagonalMeet non-escape)**.** *An accepted* RegularCauchyDiagonalMeet$^{\uparrow}$ *carrier exports only the finite rows $T, M, E, W, Q, H, C, P, N$ of Definition 909.1. It exports no quotient stream equality, host real equality, selected limit, countable-choice diagonal selector, analytic completeness theorem, or ambient RealUp completion principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetNonescape`

Proof. Inspect the carrier. The only mathematical rows are the tail meet, modulus meet, witness envelope, diagonal coverage row, and shared comparison row.

Apply the local NameCert obligations. They force every accepted read to remain inside the displayed carrier, and the shared-threshold factorization theorem forces comparison-facing reads through M, T, E, W, Q.

The excluded objects would require carrier coordinates or continuation routes not present in D. Since the packet contains only finite BEDC rows and their structural provenance, no downstream consumer can recover quotient streams, host equality, selected limits, choice selectors, analytic completeness, or an ambient completion principle from it. □

Theorem 909.6 (RegularCauchyDiagonalMeet obligation closure)**.** *The carrier of Definition 909.1, the NameCert obligations of Definition 909.2, the shared-threshold factorization of Definition 909.3, the transport induction of Definition 909.4, the non-escape theorem of Definition 909.5, and the ChapterTasteGate single-carrier rows for field faithfulness and nontriviality exhaust the paper-side obligation surface for the diagonal meet route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetObligationClosure`

Proof. Project an accepted carrier

$$D = (T, M, E, W, Q, H, C, P, N).$$

The carrier definition displays exactly the tail-meet, shared-modulus, witness-envelope, diagonal-window, comparison, transport, continuation, provenance, and local naming rows available to a consumer.

The NameCert obligations force every accepted read through those rows. The shared-threshold factorization orders the comparison route as M, T, E, W, Q, while the transport induction preserves the same route under componentwise hsame. The TasteGate rows read the same displayed carrier as field-faithful and nontrivial, so they add no hidden classifier or second carrier.

Finally apply non-escape. Quotient stream equality, host Real equality, selected limits, countable-choice diagonal selectors, analytic completeness, and ambient RealUp completion have no coordinate in D. Therefore no further obligation remains inside the displayed diagonal meet packet. □

Theorem 909.7 (RegularCauchyDiagonalMeet scoped consumer boundary)**.** *Every completion-facing consumer of an accepted* RegularCauchyDiagonalMeet$^{\uparrow}$ *carrier* $D = (T, M, E, W, Q, H, C, P, N)$ *reads the exact route*

$$M \longmapsto T \longmapsto E \longmapsto W \longmapsto Q.$$

The scope of this boundary is limited to RegularCauchyTailMeet$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchyTailWitnessEnvelope$^{\uparrow}$ RealDiagonalWindowCoverage$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *It refuses quotient streams, selected limits, countable-choice diagonal selectors, analytic completeness, and ambient* Real$^{\uparrow}$ *completion.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.RegularCauchyDiagonalMeetScopedConsumerBoundary`

Proof. Use the shared-threshold factorization theorem to force every comparison read through M, T, E, W, Q. The transport induction theorem keeps that same route stable under the componentwise hsame rows in H.

The non-escape theorem rules out quotient streams, selected limits, countable-choice diagonal selectors, analytic completeness, and ambient completion rows. The obligation-closure theorem then identifies the remaining consumer surface with the displayed carrier, its replay, provenance, and local naming rows.

Thus the boundary is exactly the listed finite route over RegularCauchyTailMeet$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, CauchyTailWitnessEnvelope$^{\uparrow}$, RealDiagonalWindowCoverage$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. □

The public diagonal-meet export is the scoped consumer route of Definition 909.7 together with the checked ChapterTasteGate carrier surface.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalMeetUp.taste gate`

Closure status for NameCert$_{\text{RegularCauchyDiagonalMeet}^{\uparrow}}$**.**

Constructive story: A RegularCauchyDiagonalMeet$^{\uparrow}$ packet is generated from finite BHist rows for a regular-Cauchy tail meet, shared modulus meet, witness envelope, diagonal-window coverage, shared-threshold comparison, transports, continuations, provenance, and local naming. The Real-completion-facing comparison is routed by Cont through the shared modulus, tail meet, witness envelope, and diagonal coverage rows before Q is exposed. Componentwise hsame preserves the same tail, modulus, envelope, coverage, and comparison rows, while Pkg records the tail-meet, modulus-meet, witness-envelope, diagonal-window, and Real dependencies. The packet is a finite diagonal meet, not completion.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyDiagonalMeet}^{\uparrow}}$, publicClosed)

Scope closed: This chapter binds the regular Cauchy diagonal-meet carrier, NameCert obligations, obligation closure, non-escape theorem, and scoped consumer boundary over RegularCauchyTailMeet$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDiagonalMeet}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient stream equality, host real equality, selected limits, countable-choice diagonal selectors, analytic completeness, or ambient RealUp completion.

Upgrade path: The next paper-axis target is a bridge route beyond the public diagonal-meet export, still without adding RealUp completeness, quotient streams, selected limits, or choice selectors.

Chapter 910

A Concrete Naming Certificate for $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$

The $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ packet realizes the generator-closure route isolated by the MetaCIC substitution audit map of chapter 9380. It sits beside the substitution audit map of chapter 779 and the substitution audit window route of chapter 824: those chapters expose the closed substitution rows and their windowed handoff route, while this packet records the generator replay surface that keeps closed shift, closed substitution, closed composition, and subject-reduction boundary reads inside a finite NameCert carrier.

Definition 910.1 (Substitution generator-audit carrier)**.** *A* $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, C, R, S, K, W, H, Q, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC* Term *row whose substitution generators are audited;*
2. *C is the* ClosedAt *boundary row attached to T;*
3. *R is the closed shift row, recording the admitted* shift *generator route under C;*
4. *S is the closed substitution row, recording the admitted* substitute *generator route under C;*
5. *K is the closed-composition row for the available closed substitution compositions;*
6. *W is the generator-route replay and obstruction ledger row, recording the finite order in which R, S, K and the subject-reduction handoff boundary may be inspected;*
7. *H records componentwise* hsame *transports for term, boundary, generator, composition, replay, provenance, route, and naming rows;*
8. *Q records the displayed* Cont *routes by which consumers replay the generator audit without leaving the packet;*
9. *P is the* Pkg *provenance over chapter 9380, chapter 779, chapter 824,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstitutionGeneratorAudit}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose rows display the closed boundary, closed shift, closed substitution, closed composition, generator-route replay, transport, continuation, provenance, and

local naming data above. It exports no unrestricted substitution calculus, no beta-substitution preservation theorem, no closed subject-reduction theorem, no quotient equality, and no host semantic equality coordinate.

Theorem 910.2 (Substitution generator-audit NameCert obligations)**.** *For the carrier of Definition 910.1, the local* $\mathsf{NameCert}_{\mathsf{SubstitutionGeneratorAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$G = (T, C, R, S, K, W, H, Q, P, N),$$

with the following finite obligations:

1. *carrier admission for* $G = (T, C, R, S, K, W, H, Q, P, N)$*;*
2. *closed-boundary exposure through* T, C*;*
3. *closed shift exposure through* R*;*
4. *closed substitution exposure through* S*;*
5. *closed-composition exposure through* K*;*
6. *generator-route replay and obstruction-ledger exposure through* W*;*
7. *componentwise transport through* H*;*
8. *displayed continuation through* Q*;*
9. *package provenance through* P*;*
10. *local naming through* N*.*

Proof. Carrier admission is projection from Definition 910.1. The rows $T, C, R, S, K, W, H, Q, P, N$ are the complete finite packet.

Read T, C first. They expose the MetaCIC term and the closed boundary at which the generator actions are admitted, matching the closed slice recorded in chapter 9380.

Read R, S, K, W next. The shift, substitution, and composition rows record the checked closed generator fragments, while W fixes the replay route and obstruction ledger through which a consumer may inspect them before any subject-reduction handoff boundary is named.

Finally inspect H, Q, P, N. These rows transport, replay, package, and name the same finite packet. Since no additional coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 910.3 (Substitution generator-audit route lock)**.** *For every accepted* $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ *packet*

$$G = (T, C, R, S, K, W, H, Q, P, N),$$

any consumer route that reaches the subject-reduction handoff boundary first passes through the closed boundary C*, the closed shift row* R*, the closed substitution row* S*, the closed-composition row* K*, and the generator-route replay row* W*. The route may export those finite syntax-side audit rows, but it cannot export unrestricted substitution, beta-substitution preservation, dependent typing preservation, closed subject reduction, quotient equality, or host semantic equality.*

Lean4 ✓ `BEDC.Derived.SubstitutionGeneratorAuditUp.SubstitutionGeneratorAudit route lock rows`

Proof. Project the accepted carrier. The only generator rows are R, S, K, and the only replay row that orders them for downstream use is W.

Use Definition 910.2. It forces any accepted read to expose T, C before the movement rows and to pass through W before the administrative rows H, Q, P, N can replay the packet.

The sibling route of chapter 824 records the same closed-window handoff discipline, while chapter 779 records the wider audit map. The present carrier keeps the generator replay as a finite packet between those two surfaces.

The excluded endpoints have no row in G. Therefore a route that claims unrestricted substitution, beta preservation, dependent typing preservation, closed subject reduction, quotient equality, or host semantic equality leaves the accepted generator-audit carrier. □

Theorem 910.4 (Substitution generator-audit TasteGate obligation)**.** *The* `\origin{ai}` *chapter for* $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ *is admissible to the TasteGate route only when the Lean follow-up for namespace* BEDC.Derived.SubstitutionGeneratorAuditUp *supplies both a* Nontrivial $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ *instance and a* FieldFaithful $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ *instance for the same finite carrier*

$$G = (T, C, R, S, K, W, H, Q, P, N).$$

Those instances must preserve the displayed closed-boundary, generator, replay, transport, continuation, provenance, and local naming rows, and they may not replace the obstruction ledger by a hidden subject-reduction theorem.

Lean4 ✓ `BEDC.Derived.SubstitutionGeneratorAuditUp.SubstitutionGeneratorAuditTasteGate single carrier alignment`

Proof. Start from the carrier and NameCert obligations. They identify exactly which fields are available to a TasteGate consumer: the closed boundary, closed generator rows, replay row, transports, continuations, provenance, and local name.

Nontriviality requires a genuine accepted witness for this carrier rather than an empty one-row shell. Field faithfulness requires each displayed coordinate to be recoverable from the carrier projections used by the TasteGate route.

The route lock supplies the obstruction side. Since subject reduction and unrestricted substitution are not fields of G, a TasteGate witness that obtains them without the generator-closure row and obstruction ledger would not be faithful to this carrier. □

Falsifiable prediction: A SubstitutionGeneratorAuditUp packet admits no subject-reduction handoff unless the generator-closure row and obstruction ledger are both present.

Independence witness: substitutionauditmap, closedsubstitutionboundary, closedsubstitutionseal, substitutionauditwindowroute, subjectreductiondischarge: the generator-audit carrier is not equivalent to any sibling because it carries a dedicated generator-route replay row W tying closed shift, closed substitution, closed composition, and handoff refusal into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SubstitutionGeneratorAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{SubstitutionGeneratorAudit}^{\uparrow}$ packet is generated from a MetaCIC term row, a ClosedAt boundary, closed shift and substitution rows, closed-composition rows, a generator-route replay row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstitutionGeneratorAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 910.1, Definition 910.2, Definition 910.3, and Definition 910.4 bind the finite generator-audit packet, its replay lock, and its TasteGate instance obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstitutionGeneratorAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution, beta-substitution preservation, dependent typing preservation, closed subject reduction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator-route replay, obstruction-ledger non-escape, and TasteGate nontriviality and field faithfulness for the same finite packet.

Chapter 911

A Concrete Naming Certificate for $\mathsf{MinkowskiRateGeometry}^{\uparrow}$

The $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ interface realizes the Lorentz-geometry step of the Inter-Hist Locality chain of chapter 9419. It consumes the finite multi-history configuration surface of chapter 474, the causal witness rows of chapter 478, the maximum-rate boundary of chapter 594, and the observer-symmetric frame-rate packet of chapter 617. Its new content is a symmetric distance-shape ledger: the packet records which rate-induced comparisons may be read as a Minkowski-like boundary, while still exporting no metric tensor, global frame, synchronized clock, host real-valued coordinate, or ambient Lorentz manifold.

Definition 911.1 (Minkowski rate geometry carrier)**.** *A* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, X, R, L, D, H, C, P, N)$$

with the following displayed rows:

1. *G is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite observer-history configuration and its no-global-sync boundary;*
2. *X is the finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows over selected pairs of histories in G;*
3. *R is an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *row whose unary bound is read over G and the causal family X;*
4. *L is an accepted* $\mathsf{LorentzFrameRate}^{\uparrow}$ *row tying the same G, X, R surface to an observer-symmetric frame-rate ledger;*
5. *D is the symmetric distance-shape ledger, recording the rate-induced light-boundary comparisons licensed by L and requiring the same row after swapping the two displayed observer histories;*
6. *H records componentwise* hsame *transport for G, X, R, L, D, the continuation routes, provenance, and local naming row;*
7. *C records visible* Cont *routes by which a consumer reads G, then X, then R, then L, then D, and transports the packet through H into the local certificate;*
8. *P is the* Pkg *provenance over* $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{LorentzFrameRate}^{\uparrow}$, *the distance-shape ledger,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MinkowskiRateGeometry}^{\uparrow}}$ *row.*

The carrier exports only this finite ledger surface. The word "geometry" names the symmetric boundary shape induced by the maximum-rate invariant, not an imported continuum, coordinate chart, or metric tensor.

Theorem 911.2 (Minkowski rate geometry NameCert obligations)**.** *For the carrier of Definition 911.1, the local* $\mathsf{NameCert}_{\mathsf{MinkowskiRateGeometry}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed packet* $M = (G, X, R, L, D, H, C, P, N)$*;*
2. *upstream exactness, requiring* G*,* X*,* R*, and* L *to be the displayed multi-Hist, causal, maximum-rate, and Lorentz-frame-rate rows read in that order;*
3. *distance-shape exactness, requiring every entry of* D *to be induced by the observer-symmetric maximum-rate row of* L*;*
4. *observer symmetry, requiring the* D *row to be stable under swapping the two displayed observer histories and transporting the swap through* H*;*
5. *route stability, requiring all reads of* D *to pass through the visible* Cont *routes in* C *and the provenance row* P*;*
6. *non-escape, excluding global frames, synchronized clocks, metric tensors, host real-valued coordinates, ambient Lorentz manifolds, and distance rows not generated by* $G, X, R, L, D, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 911.1. The packet displays the configuration, causal family, maximum-rate row, frame-rate row, distance-shape ledger, transport rows, continuation routes, provenance, and local name before any consumer reads the certificate.

Upstream exactness reads the rows in dependency order. The maximum-rate row R is tied to G and X, and the Lorentz-frame-rate row L is tied to the same G, X, R surface.

Distance-shape exactness then reads D only after L. Thus the boundary shape is a rate-induced ledger row, not an independent metric object.

Observer symmetry and route stability are the componentwise H and C clauses. They move the displayed rows together and keep every read inside the same finite packet.

The non-escape clause is the carrier inventory. Since no excluded structure is a coordinate of M, no local naming projection can export one. □

Theorem 911.3 (Minkowski rate geometry symmetry)**.** *Let*

$$M = (G, X, R, L, D, H, C, P, N)$$

be an accepted $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *carrier. Every distance-shape comparison in* D *is observer-symmetric: swapping the two displayed observer histories in a listed pair and transporting the packet through* H *returns the same distance-shape row in* D *along the visible routes in* C*.*

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometrySymmetry`

Proof. Read the distance-shape ledger D through the route required by Definition 911.1. Each row of D is attached to the observer-symmetric frame-rate packet L.

The frame-rate row inherits its maximum-rate row from the same G and X surface, so swapping the displayed observers does not select a different rate source. It only reorders the same selected pair inside the finite configuration.

Finally transport through H and replay the routes in C. The comparison returns to the same distance-shape ledger row, because D records symmetric shape data and has no coordinate for a privileged observer slot. □

Theorem 911.4 (Minkowski rate geometry non-escape)**.** *The public boundary of an accepted* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *packet exports no global frame, synchronized clock, metric tensor, host real-valued coordinate system, ambient Lorentz manifold, or distance comparison detached from the maximum-rate invariant. A consumer can read a Minkowski-like boundary only through the finite route*

$$G, X, R, L, D, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryNonescape`

Proof. Project the accepted carrier. Its coordinates are exactly the upstream configuration, causal, maximum-rate, and Lorentz-frame-rate rows, the distance-shape ledger, transport rows, continuation routes, provenance, and local name.

The NameCert obligations require every distance-shape row to be induced by L, and L is itself tied to the displayed maximum-rate row R over G and X. Hence no detached distance comparison is available.

The symmetry theorem keeps the boundary observer-symmetric under componentwise transport. It does not add a metric tensor, coordinate chart, or global frame.

Therefore the public packet is a finite rate-to-geometry boundary. Any richer geometric structure would require a coordinate absent from $G, X, R, L, D, H, C, P, N$. □

Theorem 911.5 (Minkowski rate geometry TasteGate obligation)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *must include explicit* $\mathsf{Nontrivial\ MinkowskiRateGeometry}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ MinkowskiRateGeometry}^{\uparrow}$ *instances. These instances must preserve the multi-Hist row, causal family, maximum-rate row, Lorentz-frame-rate row, symmetric distance-shape ledger,* hsame *transport,* Cont *routes, package provenance, and local naming row.*

Proof. The carrier is not determined by any single upstream row. It adds D, the symmetric distance-shape ledger, after the observer-symmetric maximum-rate surface has already been exposed by L.

Nontriviality rules out collapsing the packet to a bare maximum-rate row or a bare Lorentz-frame-rate row. Field faithfulness records preservation of every coordinate $G, X, R, L, D, H, C, P, N$, so a downstream TasteGate consumer can distinguish the rate-induced geometry boundary from each sibling surface it consumes. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryTasteGate single carrier alignment`

Theorem 911.6 (Minkowski rate geometry carrier envelope)**.** *Every accepted* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *packet is exhausted by the displayed carrier coordinates*

$$G, X, R, L, D, H, C, P, N.$$

The NameCert obligations of Definition 911.2, the symmetry theorem Definition 911.3, the non-escape theorem Definition 911.4, and the TasteGate marker `BEDC.Derived.MinkowskiRateGeometryUp.Minkowsk` *leave no additional geometry coordinate outside the finite carrier envelope.*

Proof. Project the carrier of Definition 911.1. Its coordinates are exactly the multi-Hist configuration G, causal family X, maximum-rate row R, Lorentz-frame-rate row L, symmetric distance-shape ledger D, transport row H, continuation row C, provenance row P, and local name N.

Apply the NameCert obligations and non-escape theorem. They require every consumer read to expose the same displayed rows and exclude global frames, synchronized clocks, metric tensors, host real-valued coordinates, ambient Lorentz manifolds, and detached distance comparisons.

Finally use the symmetry theorem and the TasteGate marker. Symmetry keeps the distance-shape ledger inside the same observer-swapping route, while field faithfulness records every coordinate of the single carrier, so no hidden geometry field remains beyond $G, X, R, L, D, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryCarrier envelope`

Theorem 911.7 (Minkowski rate geometry classifier stability)**.** *Let two accepted* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *packets be related by the componentwise* hsame *transport row displayed in their carriers. Then the classifier preserves the symmetric distance-shape row only through the route*

$$G, X, R, L, D, H, C, P, N.$$

In particular, classifier stability may use the NameCert obligations, observer symmetry, non-escape, and the TasteGate marker, but it may not compare packets through an external metric tensor, ambient Lorentz manifold, global frame, synchronized clock, or host real-valued coordinate.

Proof. Read the componentwise transport row H from the carrier. It transports the upstream configuration, causal family, maximum-rate row, Lorentz-frame-rate row, distance-shape ledger, continuation routes, provenance, and local name together.

Apply Definition 911.3. The transported comparison remains the same symmetric distance-shape row because swapping observers is handled inside D and replayed through C.

Apply Definition 911.4 and the TasteGate marker. Non-escape forbids external comparison coordinates, and field faithfulness keeps the classifier aligned with the displayed single-carrier route. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryCarrier classifier stability`

Theorem 911.8 (Minkowski rate geometry ledger exactness)**.** *For every accepted* $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *packet, the distance-shape ledger D is induced by the displayed* $\mathsf{MaxCausalRate}^{\uparrow}$ *row R and* $\mathsf{LorentzFrameRate}^{\uparrow}$ *row L through the NameCert route*

$$G, X, R, L, D, H, C, P, N.$$

The ledger cannot export a metric tensor, coordinate chart, global frame, synchronized clock, host real-valued coordinate, or ambient Lorentz manifold. Its exact public content is the symmetric rate-induced boundary named by the carrier, the NameCert obligations, the symmetry and non-escape theorems, and the TasteGate marker.

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryCarrier ledger exactness`

Proof. Project the rows R, L, D from the accepted carrier. The maximum-rate row R is read over the same G, X surface, and the Lorentz-frame-rate row L ties that maximum-rate surface to the observer-symmetric frame-rate ledger.

The NameCert obligations require every entry of D to be induced by L rather than supplied as an independent metric object. The symmetry theorem keeps this induced ledger invariant under the displayed observer swap.

Use non-escape and the TasteGate marker to close the ledger boundary. Since the carrier is field-faithful for $G, X, R, L, D, H, C, P, N$ and contains no metric-tensor or manifold coordinate, the public ledger is exactly the finite rate-induced distance-shape row. □

Theorem 911.9 (Minkowski rate geometry scope binding)**.** *The finite rate-to-geometry packet is scoped only over* MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, LorentzFrameRate$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the displayed TasteGate marker*

`BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryTasteGate_single_carrier_alignment`.

Within that scope, promotion reads only the finite carrier $G, X, R, L, D, H, C, P, N$ and never a metric tensor, coordinate chart, synchronized clock, global frame, host real-valued speed, curved spacetime, or ambient Lorentz manifold.

Proof. Start with the carrier of Definition 911.1. Its public rows are the multi-Hist configuration G, cross-Hist causal family X, maximum-rate row R, Lorentz-frame-rate row L, distance-shape ledger D, transport H, continuation C, provenance P, and local name N.

Apply the NameCert obligations of Definition 911.2. They force upstream exactness through MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, and LorentzFrameRate$^{\uparrow}$ before any distance-shape row is visible.

Use non-escape and ledger exactness from Definition 911.4 and Definition 911.8. These theorems keep every geometry read inside the rate-induced ledger and rule out external metric, coordinate, clock, frame, speed, spacetime, and manifold coordinates.

Finally read the TasteGate marker

```
BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryTasteGate_single_carrier_alignment.
```

It aligns field faithfulness with the same single carrier; it does not enlarge the scope. Therefore promotion is bound exactly to the listed finite packet and its displayed kernel rows. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryScopeBinding`

Theorem 911.10 (Minkowski rate geometry public certificate)**.** *The public certificate exported by* MinkowskiRateGeometry$^{\uparrow}$ *is exactly the finite route*

$$G, X, R, L, D, H, C, P, N.$$

It consumes the displayed MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, *and* LorentzFrameRate$^{\uparrow}$ *rows, then exposes only the rate-induced distance-shape ledger, componentwise transport, continuation route, provenance, and local name. It refuses metric tensors, host real coordinates, synchronized clocks, global frames, and ambient Lorentz manifolds as public certificate fields.*

Proof. Start with the carrier envelope of Definition 911.1 and Definition 911.6. These identify the complete public inventory as $G, X, R, L, D, H, C, P, N$.

Apply the NameCert obligations, observer symmetry, non-escape, ledger exactness, and scope binding from Definition 911.2, Definition 911.3, Definition 911.4, Definition 911.8, and Definition 911.9. They force every distance-shape read to pass through the upstream multi-Hist, causal, maximum-rate, and Lorentz-frame-rate rows before it reaches D.

Finally use the TasteGate marker

```
BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryTasteGate_single_carrier_alignment.
```

It records field faithfulness for the single carrier and does not admit any extra coordinate. Hence the certificate is public exactly at the listed finite route and refuses continuum Lorentz data outside that route. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryPublicCertificate`

Theorem 911.11 (Minkowski rate geometry public export)**.** *Any downstream consumer may cite the finite route*

$$G, X, R, L, D, H, C, P, N$$

as the public $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ *certificate. The export is exactly the rate-induced distance-shape ledger carried by the accepted packet, and it still refuses metric tensors, coordinate charts, synchronized clocks, global frames, host real speeds, curved spacetime, and ambient Lorentz manifolds.*

Proof. Start with the carrier envelope and NameCert obligations of Definition 911.6 and Definition 911.2. They make $G, X, R, L, D, H, C, P, N$ the complete public inventory and require every distance-shape read to pass through the displayed upstream rows.

Use ledger exactness, non-escape, and scope binding from Definition 911.8, Definition 911.4, and Definition 911.9. These keep the geometry surface inside the finite rate-induced ledger and exclude every external tensor, chart, clock, frame, host-speed, spacetime, or manifold coordinate.

Finally apply the public certificate theorem, Definition 911.10. It identifies the cited route as the public certificate itself, so the export is precisely the finite rate-induced distance-shape ledger and nothing outside that ledger. □

Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometry public export route`
Lean4 ✓ `BEDC.Derived.MinkowskiRateGeometryUp.MinkowskiRateGeometryPublicExport`

Falsifiable prediction: Deleting the observer-symmetry row permits two observers over the same MaxCausalRateUp packet to read incompatible geometry rows, while deleting the MaxCausalRateUp row leaves no rate-induced distance boundary.

Independence witness: Nearest siblings: multihistconfig, crosshistcausal, maxcausalrate, lorentzframerate, and observerlocalitytransport; no bijection exists because this carrier adds a symmetric distance-shape ledger that each sibling explicitly lacks or refuses.

Closure status for $\mathsf{NameCert}_{\mathsf{MinkowskiRateGeometry}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MinkowskiRateGeometry}^{\uparrow}$ packet is generated from a multi-Hist configuration, finite cross-Hist causal rows, a maximum-rate row, an observer-symmetric Lorentz-frame-rate row, a symmetric distance-shape ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinkowskiRateGeometry}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 911.1, Definition 911.2, Definition 911.3, Definition 911.4, Definition 911.5, Definition 911.6, Definition 911.7, Definition 911.8, and Definition 911.9 specify the finite rate-to-geometry packet, its NameCert obligations, observer-symmetry boundary, non-escape surface, TasteGate instance surface, carrier envelope, classifier stability, ledger exactness, and finite scope binding.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinkowskiRateGeometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a metric tensor, coordinate chart, synchronized clock, global frame, ambient Lorentz manifold, host real-valued speed, curved spacetime, Lean verification, or bridge checking for $\mathsf{MinkowskiRateGeometry}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, upstream exactness, distance-shape exactness, observer symmetry, route stability, non-escape, and TasteGate nontriviality with field faithfulness.

Chapter 912

A Concrete Naming Certificate for $\mathsf{AuthoredRecordBoundary}^{\uparrow}$

The $\mathsf{AuthoredRecordBoundary}^{\uparrow}$ surface makes the inscription-event vision of chapter 9397 into a finite authored-record carrier. It sits after the observer-history identity row of chapter 477, the finite prefix scheduling row of chapter 539, and the stream naming row of chapter 126. Those siblings supply the public history, schedule, and finite name rows; this chapter records the boundary at which a mismatch notice is accepted as an authored route and then exported as a public record.

Definition 912.1 (Authored-record boundary carrier)**.** *An* $\mathsf{AuthoredRecordBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, S, M, R, W, F, T, Q, V, H, C, P, N)$$

with the following displayed rows:

1. *O is the observer-history row consumed from* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*;*
2. *S is the countable finite-prefix schedule row consumed from* $\mathsf{CountableObservationSchedule}^{\uparrow}$*;*
3. *M is the mismatch-notice row that identifies the finite gap between the prior record surface and the proposed authored record;*
4. *R is the accepted-route row that binds the mismatch notice to a reusable public record route;*
5. *W is the written public-record row;*
6. *F is the refusal row excluding bypass reads that skip M or R;*
7. *T is the transport row for moving O, S, M, R, W, F componentwise through displayed* hsame *comparisons;*
8. *Q is the route-replay row of displayed* Cont *steps from prior record, through mismatch notice and accepted route, to public record;*
9. *V is the provenance row naming the source vision, observer-history, schedule, stream-name, closed-observation-system, and inscription-acceptance-budget dependencies;*
10. *H records the local handoff to* $\mathsf{ClosedObservationSystem}^{\uparrow}$ *and* $\mathsf{InscriptionAcceptanceBudget}^{\uparrow}$ *consumers;*
11. *C records the public consumer routes for authored-record reads;*

12. *P is the* Pkg *provenance over* BHist, hsame, Cont, ObserverHistoryIdentity$^{\uparrow}$, CountableObservationSchedule$^{\uparrow}$, StreamName$^{\uparrow}$, ClosedObservationSystem$^{\uparrow}$, InscriptionAcceptanceBudget$^{\uparrow}$, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AuthoredRecordBoundary}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed prior-record, mismatch-notice, accepted-route, public-record, refusal, transport, route, provenance, handoff, consumer, package, and local naming rows. It has no coordinate for private intention, raw byte authorship, host psychological identity, hidden checker state, or a record route that bypasses M or R.

Theorem 912.2 (Authored-record boundary NameCert obligations)**.** *For the carrier of Definition 912.1, the local* $\mathsf{NameCert}_{\mathsf{AuthoredRecordBoundary}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure: every accepted packet exposes exactly $A = (O, S, M, R, W, F, T, Q, V, H, C, P, N)$;*
2. *mismatch accountability: the public record row W is admitted only after the mismatch-notice row M is displayed against the prior observer-history and schedule rows O, S;*
3. *accepted-route accountability: every authored-record read passes through the accepted-route row R and the route-replay row Q;*
4. *refusal stability: the refusal row F is transported by T and remains visible to every consumer route in C;*
5. *provenance exactness: observer-history, finite schedule, stream-name, closed-observation-system, inscription-acceptance-budget, package, and local naming dependencies are exhausted by V, H, P, N.*

Lean4 ✓ `BEDC.Derived.AuthoredRecordBoundaryUp.AuthoredRecordBoundaryTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 912.1. The displayed tuple lists the prior record rows, mismatch notice, accepted route, public record, refusal, transport, route replay, provenance, handoff, consumer, package, and local naming rows.

Mismatch accountability reads M beside O and S. Since W is not a standalone row but the public record reached after the mismatch and route rows are displayed, an authored-record read cannot begin at the written record alone.

Accepted-route accountability reads R and Q. The route row names the accepted articulation, and the continuation row replays the finite path from prior record through mismatch notice to the public record.

Refusal stability is the role of F and T. The refusal row records that bypassing M or R is outside the carrier, and transport moves that refusal together with the other displayed rows.

Finally, provenance exactness is obtained by inspecting V, H, P, N. These rows name the observer-history, schedule, stream-name, closed-observation-system, inscription-acceptance-budget, package, and local certificate surfaces consumed by the boundary, with no extra source coordinate. □

Theorem 912.3 (Authored-record boundary non-escape)**.** *Every public read of an accepted* AuthoredRecordBounda *carrier factors through the finite rows*

$$O, S, M, R, W, F, T, Q, V, H, C, P, N.$$

A read that bypasses the mismatch-notice row M, bypasses the accepted-route row R, imports private intention, treats raw byte equality as authorship, or replaces the displayed provenance rows by a hidden host process has left the accepted carrier.

Proof. Project the accepted carrier. Its public-record coordinate is W, but its admission route is the displayed chain through O, S, M, R, Q with refusal, transport, provenance, and consumer rows F, T, V, H, C, P, N.

Apply Definition 912.2. The mismatch-accountability and accepted-route clauses force every authored-record read through M and R, while the refusal-stability clause records the boundary for bypass attempts.

No coordinate of the packet stores private intention, byte-level authorship, hidden checker state, or a host process trace. Therefore every public read is exhausted by the displayed rows listed above. □

Theorem 912.4 (Authored-record boundary vision handoff)**.** *The authored-record boundary realizes the inscription-event reading of chapter 9397 as a finite NameCert packet: prior record rows are supplied by* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *and* $\mathsf{CountableObservationSchedule}^{\uparrow}$*, finite naming is supplied by* $\mathsf{StreamName}^{\uparrow}$*, the closed observation side is bounded by* $\mathsf{ClosedObservationSystem}^{\uparrow}$*, the accepted-route budget is bounded by* $\mathsf{InscriptionAcceptanceBudget}^{\uparrow}$*, and the public authored record is exported only through* $M, R, W, F, T, Q, V, H, C, P, N$*.*

Proof. Start with the source vision. Its inscription event requires a prior-record surface, a mismatch or gap notice, an accepted route, and a public named record. These are exactly the O, S, M, R, W coordinates of Definition 912.1.

The sibling dependencies supply the finite inputs. Observer-history identity gives the public history row, countable observation schedules give finite prefix windows, StreamName gives the nameable stream surface, closed observation systems bound the prior-record side, and inscription-acceptance budgets bound the accepted route and downstream consumer side.

The non-escape theorem Definition 912.3 then supplies the handoff boundary. A public authored-record consumer receives only the displayed mismatch, accepted-route, record, refusal, transport, route, provenance, package, and local naming rows. □

Falsifiable prediction: A public authored-record read that bypasses the mismatch-notice row or accepted-route row refutes the carrier.

Independence witness: observerhistoryidentity, countableobservationschedule, streamname, closedobservationsystem, inscriptionacceptancebudget: $\mathsf{AuthoredRecordBoundary}^{\uparrow}$ is not bijective to any listed sibling because it carries the paired mismatch-notice and accepted-route rows together with public-record refusal and authored-record provenance; each sibling lacks at least one of those boundary rows.

The follow-up Lean TasteGate must provide $\mathsf{Nontrivial\ AuthoredRecordBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ AuthoredRecor}$ instances before the carrier can be promoted beyond the paper-side obligation surface. The expected Lean target is BEDC.Derived.AuthoredRecordBoundaryUp.AuthoredRecordBoundaryTasteGate_single_carrier_

Closure status for $\mathsf{NameCert}_{\mathsf{AuthoredRecordBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{AuthoredRecordBoundary}^{\uparrow}$ packet is generated from prior observer-history and schedule rows, mismatch notice, accepted route, public record, refusal, componentwise transport, route replay, provenance, handoff, public consumer, package, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuthoredRecordBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 912.1, Definition 912.2, Definition 912.3, and Definition 912.4 bind the finite authored-record boundary over observer-history, schedule, stream-name, closed-observation-system, and inscription-acceptance-budget rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuthoredRecordBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, private intention, raw byte authorship, hidden checker state, or authored-record reads that bypass the mismatch-notice and accepted-route rows.

Upgrade path: Obligation closure requires a checked TasteGate carrier together with Nontrivial and FieldFaithful instances for AuthoredRecordBoundary$^{\uparrow}$.

Chapter 913

A Concrete Naming Certificate for $\mathsf{BudgetedRealSealRoute}^{\uparrow}$

The $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ packet realizes the Real-completion budget route as one finite NameCert surface. It consumes the budget selector of chapter 782, the shared threshold row of chapter 560, the entrance budget of chapter 611, the sealed limit route of chapter 386, and the Real readback surface of chapter 38. Its observation rows remain tied to the finite stream, regular-rational, and dyadic sources of chapter 126, chapter 301, and chapter 291.

Definition 913.1 (Budgeted real seal route carrier)**.** *A* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, T, O, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *row selecting the finite observation budget used by the route;*
2. *T is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold row that must be displayed before the completion route may expose a seal;*
3. *O is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *entrance row tying the selected budget to the Real-facing observation surface;*
4. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route row that carries the sealed Cauchy-limit handoff;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal and readback row exposed only after S, T, O, L have been displayed;*
6. *H records componentwise* hsame *transports for the selector, threshold, entrance, limit-seal, Real readback, route, provenance, and local name rows;*
7. *C records the displayed* Cont *replay by which consumers read the selector, threshold, observation-budget entrance, limit seal, Real readback, provenance, and local name rows;*
8. *P is the* Pkg *provenance over* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}$ $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, L, R, H, C, P, N$. It has no coordinate for quotient streams, host real equality, countable choice, an external limit selector, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.BudgetedRealSealRouteUp.BudgetedRealSealRouteUp`

Theorem 913.2 (Budgeted real seal route NameCert obligations)**.** *For the carrier of Definition 913.1, the local* $\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for* $B = (S, T, O, L, R, H, C, P, N)$*, budget selector admission through* S*, TailMeet threshold admission through* T*, observation-budget entrance through* O*, CauchyLimitSeal route handoff through* L*, Real seal and readback exactness through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.BudgetedRealSealRouteUp.BudgetedRealSealRouteCarrier namecert obligations`

Proof. Project the displayed packet $B = (S, T, O, L, R, H, C, P, N)$. Each obligation names one carrier coordinate or one structural route attached to that coordinate.

The budget route starts with S, enters the Real-facing observation surface through O, and then requires the threshold row T before the limit seal L can hand a read to R. Thus the Real seal read is tied to the same finite selector, threshold, observation-budget, and limit-seal rows.

The structural rows H, C, P, N transport, replay, identify, and name the same finite packet. Since the carrier contains no external completion coordinate, these rows exhaust the local NameCert surface. □

Theorem 913.3 (Budgeted real seal route selector admission)**.** *Every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet must first expose the finite observation-budget selector row* S*. The rows* O*,* L*, and* R *are readable only as downstream route coordinates after* S *has been admitted through the selector surface of chapter 782.*

Proof. Project the carrier of Definition 913.1. Its packet is displayed as $B = (S, T, O, L, R, H, C, P, N)$, with S named as the $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ row selecting the finite observation budget used by the route.

Apply the NameCert obligation theorem. The same local surface lists budget selector admission through S before observation-budget entrance through O, CauchyLimitSeal handoff through L, and Real seal readback through R.

The sibling selector chapter supplies the admission surface for S. Since the carrier has no alternate selector coordinate, any accepted packet that reads O, L, or R must have passed through that displayed S row. □

Theorem 913.4 (Budgeted real seal route TailMeet threshold handoff)**.** *In every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet, the TailMeet threshold row* T *sits between selector admission and the CauchyLimitSeal handoff. Thus a seal-facing read can reach* L *only through the route segment* $S \to T \to O \to L$*, with* T *supplied by the shared-threshold surface of chapter 560 and* L *consumed by the CauchyLimitSeal route of chapter 386.*

Proof. Read the displayed packet in Definition 913.1. The selector row S fixes the finite observation budget, while T is the regular Cauchy tail threshold that must be displayed before the completion route may expose a seal.

Use the aggregate obligation theorem to align these carrier rows with the local NameCert surface. It records selector admission through S, threshold admission through T, observation-budget entrance through O, and CauchyLimitSeal route handoff through L as obligations of the same packet.

The TailMeet sibling supplies the shared threshold interpretation of T, and the CauchyLimitSeal sibling supplies the handoff interpretation of L. Because no coordinate bypasses T, every seal-facing read from the accepted packet passes through the stated route segment. □

Theorem 913.5 (Budgeted real seal route Real-seal readback exactness)**.** *For every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet, the Real seal row R is available only through the displayed route $S \to T \to O \to L \to R$ together with the structural rows H, C, P, N. No accepted packet exposes a quotient stream, host real equality, selected limit, countable-choice selector, or ambient completeness row as part of the readback.*

Proof. Project the carrier. The only Real-facing coordinate is R, and the carrier definition places it after the selector row S, threshold row T, observation-budget entrance row O, and CauchyLimitSeal route row L.

Apply the local NameCert obligation theorem. It identifies Real seal and readback exactness through R as one obligation of the same finite packet, while H, C, P, N supply transport, replay, provenance, and local naming for that displayed route.

Finally use the non-escape boundary of Definition 913.6. The objects excluded in the statement require coordinates absent from $B = (S, T, O, L, R, H, C, P, N)$, so the Real readback has exactly the displayed route and no hidden completion-facing row. □

Theorem 913.6 (Budgeted real seal route non-escape)**.** *Every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet exports only the displayed rows*

$$S, T, O, L, R, H, C, P, N.$$

In particular, no accepted packet may expose the $\mathsf{Real}^{\uparrow}$ *seal row R without the TailMeet threshold row T, and no consumer may recover a quotient stream, host equality of real names, countable-choice selector, external limit, or ambient Real-completion principle from the packet.*

Proof. Inspect the carrier. The only Real-facing coordinate is R, and it is reached through the route rows S, T, O, L together with the structural rows H, C, P, N.

Apply the NameCert obligation theorem. It requires threshold admission through T and limit-seal handoff through L before Real readback exactness through R is available.

The excluded objects would require coordinates not present in the carrier. Because every export must factor through the displayed rows, the packet has no route to quotient streams, host equality, choice selectors, external limits, or an ambient completeness theorem. □

Theorem 913.7 (Budgeted real seal route TasteGate obligation)**.** *The Lean follow-up for* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ BudgetedRealSealRoute}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ BudgetedRealSealRoute}^{\uparrow}$ *in* `TasteGate.lean`*. These instances must preserve the selector row S, threshold row T, observation-budget row O, CauchyLimitSeal route row L, Real seal row R, transport row H, replay row C, provenance row P, and local naming row N.*

Proof. Use Definition 913.1 to read the nine displayed rows. A nontrivial gate witness must distinguish a packet by at least one of those finite coordinates rather than by an ambient real-completion object.

Field faithfulness is the same row preservation requirement. The gate-facing instance may transport and replay rows through H and C, but it cannot identify two packets after dropping S, T, O, L, R, P, or N.

Thus the TasteGate work is tied to the finite route packet itself, with no hidden completion principle or host equality field. □

Lean4 ✓ `BEDC.Derived.BudgetedRealSealRouteUp.BudgetedRealSealRouteTasteGate single carrier alignment`

Theorem 913.8 (Budgeted real seal route obligation closure)**.** *For the carrier of Definition 913.1, the obligation surface is complete at the displayed route packet*

$$B = (S, T, O, L, R, H, C, P, N).$$

It consists exactly of carrier admission, selector admission through S, TailMeet threshold admission through T, $\mathsf{RealObservationBudget}^{\uparrow}$ *entrance through O,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff through L,* $\mathsf{Real}^{\uparrow}$ *seal readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$ *row N.*

Proof. Start from the NameCert obligation list in Definition 913.2. It projects precisely the carrier row, selector row, threshold row, observation-budget entrance, CauchyLimitSeal handoff, Real readback, transport row, continuation row, provenance row, and local naming row.

Use the selector admission, TailMeet handoff, and Real-seal readback theorems to read the dynamic route. Theorems 913.3, 913.4, and 913.5 force every accepted read through S, T, O, L, R in that order.

Finally apply non-escape, Definition 913.6. Since no hidden completion, quotient stream, host equality, selected limit, or choice coordinate is present, the listed rows exhaust the obligation surface. □

Theorem 913.9 (Budgeted real seal route ledger scope)**.** *Every obligation of an accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet is grounded in* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the sibling chapter rows for* $\mathsf{FiniteObservationBudgetS}$ $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *No obligation is grounded in quotient streams, host equality of real names, countable choice, selected host limits, or an ambient completeness theorem.*

Proof. Read the provenance coordinate P in Definition 913.1. It lists the kernel support rows and the sibling concrete-instance sources consumed by the budgeted Real-seal route.

The obligation closure theorem identifies the complete local surface. Each listed route obligation is therefore one of the carrier rows S, T, O, L, R or one of the structural rows H, C, P, N, and those structural rows are exactly hsame, Cont, Pkg, and $\mathsf{NameCert}$ support over the same BHist packet.

The excluded sources are absent from both the carrier and the provenance list. By Definition 913.6, an accepted consumer cannot add them as ledger grounds after the packet has been admitted. □

Theorem 913.10 (Budgeted real seal route source exactness)**.** *For every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet, the displayed source route*

$$S \longmapsto T \longmapsto O \longmapsto L \longmapsto R$$

is exact: selector admission, TailMeet threshold admission, RealObservationBudget entrance, CauchyLimitSeal handoff, and Real readback occur in that order and through those rows only. The structural rows H, C, P, N transport, replay, package, and name the route without adding another source.

Proof. First apply selector admission, Definition 913.3. The route must begin at the finite observation-budget selector row S.

Next apply the TailMeet threshold handoff, Definition 913.4. It places T between selector admission and the CauchyLimitSeal handoff and binds O as the RealObservationBudget entrance row.

Then apply Real-seal readback exactness, Definition 913.5. It identifies R as readable only after the displayed S, T, O, L route and confines H, C, P, N to transport, replay, provenance, and local naming. □

Theorem 913.11 (Budgeted real seal route public export)**.** *For every accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet*

$$B = (S, T, O, L, R, H, C, P, N),$$

the rows $S, T, O, L, R, H, C, P, N$ *export the public* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *interface consumed by finite Real-completion readers. The exported interface is exactly the selector row, TailMeet threshold, observation-budget entrance, CauchyLimitSeal handoff, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$ *row.*

Proof. Start with obligation closure, Definition 913.8. It identifies the complete displayed packet surface and leaves no row outside $S, T, O, L, R, H, C, P, N$.

Apply ledger scope, Definition 913.9. The public reader may use the same BHist, hsame, Cont, Pkg, and NameCert grounds already listed by the packet, but it does not receive an additional ground.

Finally apply source exactness, Definition 913.10. The readable Real route is $S \longmapsto T \longmapsto O \longmapsto L \longmapsto R$, and the remaining rows transport, replay, package, and name that route. This is exactly the public finite Real-completion interface. □

Theorem 913.12 (Budgeted real seal route public consumer exactness)**.** *Any public finite Real-completion consumer of an accepted* $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ *packet factors through the exported route* $S, T, O, L, R, H, C, P, N$*. No such consumer factors through a quotient stream, host real equality, countable choice, selected limit, ambient completeness theorem, or bridge row.*

Proof. Use the public export theorem to fix the consumer surface. Its rows are the only exported coordinates of the accepted packet, so a public consumer must read through $S, T, O, L, R, H, C, P, N$.

The source-exactness theorem then pins down the order of the Real route: selector admission, TailMeet threshold admission, RealObservationBudget entrance, CauchyLimitSeal handoff, and Real readback. The structural rows H, C, P, N carry transport, replay, provenance, and naming for that route.

The ledger-scope theorem excludes every proposed bypass. Quotient streams, host real equality, countable choice, selected limits, ambient completeness, and bridge rows are absent from the carrier and from the ledger grounds, so they cannot be routes for a public consumer of the displayed packet. □

Falsifiable prediction: If the displayed TailMeet threshold row T is absent, no accepted $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ packet can expose the $\mathsf{Real}^{\uparrow}$ seal row R.

Independence witness: finiteobservationbudgetselector, regularcauchytailmeet, real-observation-budget, cauchylimitseal, real: $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ combines a selector row, a TailMeet threshold, a Real observation-budget entrance, a CauchyLimitSeal handoff, and a Real seal/readback row; none of the named siblings is bijective to this combined route packet because each omits at least one required route coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BudgetedRealSealRoute}^{\uparrow}$ packet is generated from a finite observation-budget selector, a TailMeet threshold row, a Real observation-budget entrance, a CauchyLimitSeal route, a Real seal and readback row, componentwise transports, displayed replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 913.1, Definition 913.2, Definition 913.8, Definition 913.9, Definition 913.10, Definition 913.11, Definition 913.12, Definition 913.6, and Definition 913.7 bind the selector row, TailMeet threshold, observation budget, CauchyLimitSeal handoff, Real readback row, hsame, Cont, Pkg, and local NameCert rows as one public finite Real-completion budget route.

Formal verification: FormalStatus(NameCert$_{\mathsf{BudgetedRealSealRoute}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BudgetedRealSealRouteUp.BudgetedRealSealRouteTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, host real equality, countable choice, external limit selection, ambient Real completeness, or bridge checking for BudgetedRealSealRoute$^{\uparrow}$.

Upgrade path: No paper-side closure upgrade remains for the displayed selector, TailMeet threshold, observation budget, CauchyLimitSeal handoff, Real readback, hsame, Cont, Pkg, and NameCert route.

Chapter 914

A Concrete Naming Certificate for $\mathsf{CompactNetModulusSelector}^{\uparrow}$

The $\mathsf{CompactNetModulusSelector}^{\uparrow}$ interface is the concrete finite selector behind the compact-uniform-continuity construction of chapter 9237. It consumes the compact finite-net rows of chapter 60, the pointwise-modulus rows of chapter 59, the metric-distance rows of chapter 57, the positive rational order rows of chapter 37, and the finite lower-bound fold surface of chapter 619. Its output is the single finite packet a uniform-modulus consumer reads when a tolerance history is converted into a precision history by a compact net.

The selected precision row is assembled from sibling concrete packets: the finite cover and budget surface of chapter 648, the fold output of chapter 619, and the downstream precision packet read by chapter 369. The selector therefore records a horizontal dependency route from compact cover data through a finite lower-bound fold to the uniform-modulus consumer boundary.

Definition 914.1 (Compact net modulus selector carrier)**.** *A* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

with the following visible rows:

1. *X is the displayed compact metric source row, including the located finite-net coverage data for the requested tolerance history;*
2. *Y is the displayed target metric row whose distance comparisons are consumed after the pointwise-modulus reads;*
3. *E is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history requested by the uniform-modulus consumer;*
4. *$\mathcal{P}$ is the nonempty finite* $\mathsf{ProbeBundle}$ *spine of compact-net centers;*
5. *C records bundled center histories and half-radius coverage rows for the source metric;*
6. *R records the positive* $\mathsf{Rat}^{\uparrow}$ *subordinate radius rows attached to the bundled centers;*
7. *M records the* $\mathsf{CFun}^{\uparrow}$ *pointwise-modulus rows at the centers of $\mathcal{P}$ for the tolerance history E;*
8. *F is the accepted* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound row over $\mathcal{P}$ and R;*
9. *D is the selected positive precision history exported by the fold row F;*
10. *H records componentwise* hsame *transport for the metric, compact-net, tolerance, radius, pointwise-modulus, fold, precision, route, provenance, and naming rows;*

11. *Q records the displayed* Cont *routes through which compact, continuous, metric, rational, fold, and uniform-modulus consumers read the packet;*
12. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and local naming rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}^{\uparrow}}$ *row.*

The carrier exports only the compact-net input, pointwise-modulus radii, selected lower-bound fold row, tolerance-to-precision handoff, transport rows, continuation routes, package provenance, and local naming row. It has no coordinate for open-cover compactness, a host minimum, a choice-selected center, a quotient of point histories, or an external modulus function.

Theorem 914.2 (Compact net modulus selector NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet of Definition 914.1;*
2. *compact-net exactness, requiring $X, \mathcal{P}, C, R$ to display the located finite ε-net rows read from the compact source certificate;*
3. *pointwise-modulus exactness, requiring every row of M to be attached to a bundled center and to the displayed tolerance history E;*
4. *lower-bound fold exactness, requiring F to be the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *row over the same $\mathcal{P}$ and R and requiring D to be its exported positive lower-bound history;*
5. *non-escape, requiring every consumer read of D to pass through $X, Y, E, \mathcal{P}, C, R, M, F, H, Q, P, N$ and forbidding open-cover compactness, host minima, choice-selected centers, quotient point equality, and external modulus functions.*

Proof. Carrier admission is projection from Definition 914.1. The packet displays the source and target metric rows, tolerance history, finite probe spine, compact coverage rows, subordinate radius rows, pointwise-modulus rows, fold row, selected precision, transports, routes, provenance, and local name.

Compact-net exactness reads the finite coverage surface from $X, \mathcal{P}, C, R$. The center and radius rows are not reconstructed by the selector; they are the displayed input rows supplied by the compact naming certificate.

Pointwise-modulus exactness then reads M only at bundled centers. Each local row is tied to the same tolerance history E and to the source and target metric rows already displayed in the carrier.

Lower-bound fold exactness reads F over the same probe spine and radius ledger. The selected precision D is accepted only as the positive lower-bound output of that finite fold.

The non-escape obligation is the carrier inventory together with the displayed routes. Since no excluded object is a coordinate of S, no local naming projection or continuation route can export one. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorNameCertObligations`

Theorem 914.3 (Compact net modulus selector fold soundness)**.** *Let*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

be an accepted $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier. If a source-distance read is certified below the selected precision history D, then the compact-continuous handoff for tolerance E factors through*

one bundled center of $\mathcal{P}$, the corresponding pointwise-modulus row in M, and the same lower-bound fold certificate F.

The handoff uses the selected row only as a finite $\mathsf{Rat}^{\uparrow}$ *lower bound below every displayed subordinate radius row in R.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier fold soundness`

Proof. Begin with the compact-net coordinate. The accepted carrier exposes $\mathcal{P}$ as a nonempty finite probe spine and C as the coverage row assigning each source-point history to a bundled center with a displayed half-radius comparison.

Read the selected precision through F. By the lower-bound fold exactness obligation, D is below every radius ledger in R through the $\mathsf{Rat}^{\uparrow}$ classifier and the stored transport rows in H.

Use the metric rows of X to route a source-distance read below D through the bundled center supplied by C. The comparison $D \leq R_p$ keeps the second point inside the local ball named by the same probe.

Finally apply the corresponding pointwise-modulus row of M and the target metric row Y. The output distance estimate is therefore read from $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$, not from an external modulus selector. □

Theorem 914.4 (Compact net modulus selector non-escape)**.** *Every consumer read of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet factors through the displayed compact-net input, pointwise-modulus radii, selected lower-bound fold row, transport rows, continuation routes, provenance, and local naming row of Definition 914.1. The packet exports no open-cover compactness principle, no host minimum over rationals, no choice-selected center, no quotient point equality, no sequential compactness argument, and no external modulus function.*

Proof. Project the accepted carrier. Its coordinates are exactly $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$.

The NameCert obligations force compact reads through $X, \mathcal{P}, C, R$, pointwise-modulus reads through M, and selected-precision reads through F, D. The transport and continuation rows H, Q replay those same routes componentwise.

No coordinate remains from which a consumer could read open-cover compactness, host minima, choice-selected centers, quotient point equality, sequential compactness, or an external modulus function. Thus every public read stays within the finite selector packet. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier nonescape`

Theorem 914.5 (Compact net modulus selector independence boundary)**.** *The* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier is not bijective, as a NameCert obligation surface, to any one of its compact, continuous, metric, rational, finite-net-minimum-fold, or uniform-modulus siblings. It is the joint packet that stores the compact-net input, pointwise-modulus radii, selected lower-bound fold row, and tolerance-only handoff together.*

Proof. The compact sibling supplies finite-net coverage but has no pointwise-modulus row, no selected lower-bound fold output, and no target-metric handoff.

The continuous sibling supplies pointwise-modulus rows but has no compact probe spine and no finite lower-bound fold over a bundled radius ledger.

The metric and rational siblings supply distance and order readbacks only. They do not carry the compact coverage row, the bundled center ledger, or the tolerance-to-precision handoff.

The finite-net-minimum-fold sibling supplies the lower-bound fold but not the compact coverage input, pointwise-modulus rows, or target-metric consumer route. The uniform-modulus sibling

reads the resulting precision packet, while this selector records the compact-net and pointwise-modulus source rows that justify that precision. Hence no bijection preserves the listed NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier independence boundary`

Theorem 914.6 (Compact net modulus selector TasteGate obligation)**.** *The later Lean TasteGate file for namespace* BEDC.Derived.CompactNetModulusSelectorUp *must provide* Nontrivial CompactNetModulusSelecto *and* FieldFaithful CompactNetModulusSelector$^{\uparrow}$ *instances for the same finite carrier. These instances must preserve the compact-net input, pointwise-modulus radius rows, selected lower-bound fold row, tolerance-to-precision handoff,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorTasteGate single carrier alignment`

Proof. Nontriviality is witnessed by the dependency mix of the carrier. Removing the compact-net rows leaves no bundled source for the fold; removing the pointwise-modulus rows leaves no local continuity handoff; removing the fold row leaves no selected tolerance-only precision.

Field faithfulness records that these rows are preserved as one packet. A TasteGate consumer must therefore distinguish CompactNetModulusSelector$^{\uparrow}$ from its compact, continuous, metric, rational, finite-fold, and uniform-modulus siblings by the full coordinate surface $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$. □

Theorem 914.7 (Compact net modulus selector root obligation triad)**.** *Every accepted* CompactNetModulusSelecto *packet exposes exactly the compact finite-net row $X, \mathcal{P}, C$, the pointwise-modulus radius rows R, M, the lower-bound fold row F, the selected precision D, and the structural rows H, Q, P, N of Definition 914.1. The root obligation surface is therefore the triad consisting of finite-net input, pointwise-radius input, and metric-selector output. It exports no open-cover compactness, host minimum, quotient point equality, choice-selected center, or external modulus function.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorRootObligationTriad`

Proof. Read the carrier definition. The finite-net part is visible in $X, \mathcal{P}, C$, the radius and pointwise-modulus part is visible in R, M, and the selected metric precision is visible through F, D.

Use Definition 914.2. Its compact-net exactness, pointwise-modulus exactness, lower-bound fold exactness, and non-escape clauses bind those rows to the same local naming surface.

The fold, non-escape, and TasteGate rows then exhaust the public consumer route. By Definition 914.3, Definition 914.4, and Definition 914.6, the output precision is read as one finite packet and not as any excluded compactness, minimum, quotient, center-selection, or external-modulus object. □

Theorem 914.8 (Compact net modulus selector pointwise radius route)**.** *For an accepted* CompactNetModulusSelec *carrier $S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$, each center radius is read through the* CFun$^{\uparrow}$ *pointwise-modulus row M, the positive* Rat$^{\uparrow}$ *radius row R, and the* Met$^{\uparrow}$ *source and target rows X, Y before the finite lower-bound fold row F can select the precision history D. No center radius is consumed as a free scalar, a host metric fact, or a hidden modulus choice.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier pointwise radius route`

Proof. Begin with the displayed center ledger. The probe spine $\mathcal{P}$ and coverage row C identify the bundled centers, while the radius row R records positive rational histories attached to those centers.

Next read the pointwise route. The row M supplies the $\mathsf{CFun}^{\uparrow}$ pointwise-modulus data at the bundled centers, and the rows X, Y supply the source and target $\mathsf{Met}^{\uparrow}$ distance reads through the continuation routes in Q.

Only after those rows are visible does the fold row F select D. The NameCert obligations and fold soundness force F to fold over the same $\mathcal{P}$ and R, so the selected precision cannot be read before the pointwise-radius and metric boundaries are in place. □

Theorem 914.9 (Compact net modulus selector metric consumer boundary)**.** *Downstream compact-uniform-continuity consumers of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier receive a finite metric selector packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

and nothing outside that packet. The delivered boundary consists of compact finite-net input, pointwise-modulus radius rows, the finite lower-bound fold, selected precision, transport rows, continuation routes, provenance, and the local NameCert *row; it does not deliver open-cover compactness, a host minimum, quotient point equality, a choice-selected center, or an external modulus function.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier metric consumer boundary`

Proof. Project the consumer-facing packet from Definition 914.1. Its metric content is displayed by X, Y, R, D together with the finite center and fold rows $\mathcal{P}, C, M, F$.

Apply the NameCert and non-escape theorems. They require every consumer read to pass through the listed carrier rows, transports, routes, provenance, and local name.

The sibling inputs cited in the chapter introduction supply the compact, continuous, metric, rational, and fold readings, but this selector exports only their assembled finite packet. Thus the compact-uniform-continuity consumer receives a selector boundary rather than an open-cover compactness principle, host minimum, quotient equality, selected center, or external modulus. □

Theorem 914.10 (Compact net modulus selector consumer factorization)**.** *Every downstream* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

factors through compact finite-net input $X, \mathcal{P}, C$, pointwise continuous radius rows R, M, metric rows X, Y, the positive rational tolerance row E, finite lower-bound fold F, selected precision D, and structural rows H, Q, P, N. No such consumer may replace this route by open-cover compactness, a host minimum, quotient point equality, a choice-selected center, sequential compactness, or an external modulus function.

Proof. Begin with the local NameCert obligations. They attach the accepted selector to the displayed compact, continuous, metric, rational, fold, selected-precision, transport, route, provenance, and naming rows of S.

Use fold soundness and the pointwise radius route. The fold row F reads the finite probe spine and radius ledger, while each radius row reaches D only through the pointwise continuous row M, metric rows X, Y, and positive rational tolerance row E.

Now apply the metric consumer boundary. A downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the assembled selector packet, not a detached compactness theorem or detached modulus function.

Finally invoke non-escape. Since the carrier has no row for open-cover compactness, host minima, quotient point equality, choice-selected centers, sequential compactness, or external modulus functions, every consumer factors through the displayed rows $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier consumer factorization`

Theorem 914.11 (Compact net modulus selector request obligation surface)**.** *For an accepted carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the request obligation surface distinguishes precision requests by visible tolerance and finite-net rows. If two requests have tolerance rows E_0 and E_1 or finite-net rows $(\mathcal{P}_0, C_0, R_0)$ and $(\mathcal{P}_1, C_1, R_1)$ that are not identified by the displayed hsame *rows H, then their selected precision histories must be read as distinct request packets through Q, P, N. This obligation is part of the TasteGate surface of Definition 914.6.*

Proof. Project the carrier. The tolerance row E, finite probe spine $\mathcal{P}$, coverage row C, subordinate radius row R, selected precision row D, transports H, routes Q, provenance P, and local name N are visible coordinates of S.

Use the NameCert obligations and fold soundness. They require selected precision to be read through the same tolerance, compact-net, pointwise-modulus, and fold rows that produced it.

Apply the TasteGate obligation surface. Because TasteGate preservation must keep the displayed carrier rows, a change in visible tolerance or finite-net data that is not transported by H cannot be collapsed into the same request packet by Q, P, N. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorRequestObligationSurface`

Theorem 914.12 (Compact net modulus selector TasteGate nontriviality boundary)**.** *The TasteGate obligation for* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *is nontrivial on the displayed carrier $S = (X, Y, E, \mathcal{P}, C, R, M, F, I$ At least the request with tolerance row E_0 and finite-net row $(\mathcal{P}_0, C_0, R_0)$ and the request with tolerance row E_1 or finite-net row $(\mathcal{P}_1, C_1, R_1)$ must remain distinguishable whenever their visible rows are not connected by the displayed* hsame *transport H.*

Thus the carrier cannot be replaced by a one-element surface while preserving the compact-net input, pointwise-modulus rows, lower-bound fold row, selected precision history, Cont *routes,* Pkg *provenance, and local* NameCert *row named in Definition 914.6.*

Proof. Start from the request obligation surface. It ties a selected precision history to the tolerance row and finite-net rows that generated the request.

If two requests differ in visible tolerance or finite-net data and H does not identify those rows, the local classifier has no displayed transport that can merge them. The routes Q and provenance row P must therefore preserve the difference when the local name N exports the packet.

The TasteGate obligation requires preservation of the full coordinate surface. A one-element carrier would erase the displayed distinction between the two request packets, so it cannot satisfy the nontriviality boundary for this NameCert surface. □

Theorem 914.13 (Compact net modulus selector field-faithful row)**.** *Any consumer preserving all displayed rows of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

preserves the exported precision history D together with its compact-net, pointwise-modulus, lower-bound fold, transport, route, provenance, and naming evidence. This field-faithful row is the paper-side obligation that allows future Lean instances for $\mathsf{Nontrivial\ CompactNetModulusSelector}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ CompactNetModulusSelector}^{\uparrow}$ *to use the displayed request packets rather than a one-element carrier.*

Proof. Read D through the carrier and the fold soundness theorem. The selected precision is not a free output; it is the lower-bound fold result tied to $E, \mathcal{P}, C, R, M, F$.

Use non-escape and the TasteGate obligation. Every public consumer of the selected precision must preserve the transport rows H, continuation routes Q, package provenance P, and local name N along with the compact, continuous, metric, rational, and fold inputs.

Therefore a row-preserving consumer preserves the exported precision history as a field of the same packet. This is exactly the field-faithful surface needed for later Lean instances, without collapsing the NameCert carrier to one element. □

Theorem 914.14 (Compact net modulus selector source coverage)**.** *Let*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

be an accepted $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier. Every accepted selector packet reads the compact cover rows, finite-net fold row, continuous pointwise-modulus rows, source and target metric distance rows, positive* $\mathsf{Rat}^{\uparrow}$ *precision row, and* $\mathsf{UniformModulus}^{\uparrow}$ *output handoff through the displayed carrier coordinates of Definition 914.1. No source row is supplied by an ambient compactness principle, host minimum, quotient point equality, choice-selected center, or external modulus function.*

Proof. Start with the carrier inventory of Definition 914.1. The compact source is visible in $X, \mathcal{P}, C, R$, the continuous pointwise rows are visible in M, the metric rows are visible in X, Y, and the positive precision row is visible in D.

Apply Definition 914.3 and Definition 914.8. The fold reads only the displayed probe spine and radius ledger, while each center radius reaches the selected precision only through the pointwise-modulus and metric rows already stored in S.

Finally use Definition 914.9 and Definition 914.13. A downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the same field-preserving selector packet, so every public source read remains inside the listed carrier coordinates. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorSourceCoverage`

Theorem 914.15 (Compact net modulus selector center ledger exposure)**.** *Let*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

be an accepted $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier. Every accepted selector packet exposes the finite probe spine $\mathcal{P}$, center coverage row C, subordinate radius row R, and pointwise-modulus row M before the selected precision row D can be read. Thus the precision handoff is never read from a hidden center selector, a quotient point equality, an ambient compactness principle, or an external modulus function.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCenterLedgerExposure`

Proof. Project S through Definition 914.1. The finite probe spine, center coverage row, subordinate radius row, and pointwise-modulus row are displayed coordinates of the same accepted packet as the selected precision row.

Apply Definition 914.2. Its compact-net exactness and pointwise-modulus exactness rows force C, R, M to be public before a consumer can reach the lower-bound fold or the selected precision.

Finally use Definition 914.8. Each center radius reaches the precision threshold only through the displayed pointwise-modulus and metric rows, so no hidden center choice, quotient equality, ambient compactness principle, or external modulus function can supply D. □

Theorem 914.16 (Compact net modulus selector fold radius exhaustion)**.** *For an accepted carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the fold row F consumes exactly the displayed finite probe spine $\mathcal{P}$ and subordinate radius ledger R before it exports the selected precision row D. The fold cannot read a host minimum, a choice-selected center, quotient point equality, or an external modulus as a second source for the selected radius budget.

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorCarrier fold radius exhaustion`

Proof. Start with Definition 914.3. It identifies the lower-bound fold as a read over the displayed probe spine and radius ledger.

Use Definition 914.21. The selected precision is exact only as the fold output over $\mathcal{P}$ and R, so a second host minimum or external rational infimum has no carrier coordinate.

Apply non-escape and field faithfulness, Definition 914.4 and Definition 914.13. Public consumers preserve the displayed fold route and cannot replace it by choice-selected center data, quotient point equality, or an external modulus. □

Theorem 914.17 (Compact net modulus selector UniformModulus handoff)**.** *Any downstream* $\mathsf{UniformModulus}^{\uparrow}$ *read of the selected precision row D from an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet passes through the already exposed compact row $X, \mathcal{P}, C, R$, continuous pointwise row M, metric rows X, Y, rational tolerance row E, fold row F, transport row H, continuation row Q, provenance row P, and local* NameCert *row N. The handoff therefore consumes the selector obligation surface rather than a parameter-only modulus promise.*

Proof. Begin with source coverage, Definition 914.14. It places every public source read for the uniform-modulus output inside the displayed selector coordinates.

Next use the metric consumer boundary, Definition 914.9. A metric-facing consumer must keep the compact, continuous, rational, and fold inputs attached to the selected threshold rather than reading the threshold as a detached number.

Finally apply the request obligation surface and public handoff rows, Definition 914.6 and Definition 914.22. They carry the same transport, continuation, provenance, and local naming rows into $\mathsf{UniformModulus}^{\uparrow}$, so the downstream read of D is exactly the exposed selector route. □

Theorem 914.18 (Compact net modulus selector metric threshold stability)**.** *For accepted* CompactNetModulusSel *carriers*

$$S_0 = (X_0, Y_0, E_0, \mathcal{P}_0, C_0, R_0, M_0, F_0, D_0, H_0, Q_0, P_0, N_0)$$

and

$$S_1 = (X_1, Y_1, E_1, \mathcal{P}_1, C_1, R_1, M_1, F_1, D_1, H_1, Q_1, P_1, N_1),$$

componentwise hsame *transport along the displayed carrier rows and replay through the displayed* Cont *routes preserves the chosen precision threshold. Thus any metric consumer that reads D_0 below the pointwise-radius fold route of S_0 reads the transported threshold D_1 through the corresponding fold, radius, pointwise-modulus, metric, provenance, and naming rows of S_1.*

Proof. Project both packets with Definition 914.1. The transport rows H_0, H_1 name the componentwise hsame identifications for compact-net data, tolerance histories, radius ledgers, pointwise-modulus rows, folds, selected precision histories, routes, provenance, and local names.

Use Definition 914.8 to identify the threshold route before the fold is read. The selected precision is available only after the corresponding radius and pointwise-modulus rows have been transported through the displayed metric rows.

Apply Definition 914.3. Since the fold row is over the transported probe spine and radius ledger, the selected threshold is preserved as the same finite lower-bound read rather than as a host minimum.

The consumer-facing statement follows from Definition 914.9 and Definition 914.13. Field-preserving consumers must keep the selected precision together with its compact, continuous, metric, rational, fold, route, provenance, and naming evidence. □

Theorem 914.19 (Compact net modulus selector ledger route exactness)**.** *Every public read of an accepted* CompactNetModulusSelector$^{\uparrow}$ *selector packet factors through the finite ledger rows*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$$

named by Definition 914.1. In particular, compact-cover reads, pointwise-modulus reads, metric-threshold reads, rational-precision reads, finite-fold reads, and UniformModulus$^{\uparrow}$ *handoff reads use the same displayed ledger route, with no host minimum and no ambient compactness principle.*

Proof. First read the ledger from Definition 914.1. The listed coordinates are the only rows exported by the selector packet.

Next apply Definition 914.3 and Definition 914.8. They force the selected precision to pass through the displayed compact-net, radius, pointwise-modulus, metric, and fold rows before any consumer can use it.

Finally apply Definition 914.9 and Definition 914.13. Public consumers preserve exactly the carrier fields they receive, so the route cannot be replaced by a host minimum, ambient compactness principle, quotient equality, selected-center choice, or external modulus. □

Theorem 914.20 (Compact net modulus selector compact-cover consumer lock)**.** *Let*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

be an accepted CompactNetModulusSelector$^{\uparrow}$ *carrier. Every compact-cover consumer of S reads the source metric row X, finite probe spine $\mathcal{P}$, coverage ledger C, and subordinate radius row R before it can read the pointwise-modulus row M, the fold row F, or the selected precision row D. The consumer imports no open-cover compactness principle and no choice-selected center.*

Proof. Project the carrier of Definition 914.1. The compact-cover portion is displayed by $X, \mathcal{P}, C, R$, while M, F, D are later rows in the same finite selector packet.

Use the NameCert obligations and fold soundness. Compact-net exactness binds cover reads to $X, \mathcal{P}, C, R$, and lower-bound fold exactness can only use the already displayed probe spine and radius ledger.

Finally apply non-escape and the TasteGate obligation row. Since neither open-cover compactness nor a choice-selected center is a carrier coordinate, no compact-cover consumer can import either object before the selector reaches M, F, D. □

Theorem 914.21 (Compact net modulus selector finite-fold budget exactness)**.** *For an accepted carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the selected precision history D is exact only as the $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *output F over the displayed probe spine $\mathcal{P}$ and radius ledger R. No host minimum, external rational infimum, hidden compactness choice, or quotient of center histories can serve as a second budget source for D.*

Proof. Read D through the carrier. The only row that exports a selected lower-bound history is F, and F is named as the fold over $\mathcal{P}$ and R.

Apply the NameCert lower-bound fold obligation together with Definition 914.3. These rows force the budget comparison to remain inside the displayed finite probe spine and radius ledger.

Use non-escape and TasteGate preservation to exclude alternate sources. A host minimum, external infimum, hidden center choice, or quotient center history is not preserved as a field of S, so it cannot be the exactness witness for D. □

Theorem 914.22 (Compact net modulus selector uniform-modulus public handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer downstream of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet receives exactly the assembled finite packet*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N.$$

It receives the tolerance row, compact finite-net rows, pointwise-modulus rows, selected fold budget, transports, continuation routes, provenance, and local naming row, and it receives no open-cover compactness, host minimum, choice-selected center, quotient point equality, or external modulus function.

Proof. Start with the carrier and the NameCert obligations. They expose $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$ as the full selector packet and bind cover, pointwise, fold, and route reads to those rows.

Use fold soundness to identify D as the precision handed to the uniform-modulus consumer. The selected precision is produced over the displayed probe spine and radius ledger, not by an external selector.

Finally apply non-escape and the TasteGate obligation row. Every downstream public handoff preserves the same finite fields, so excluded compactness, minimum, choice, quotient, and external-modulus objects are not delivered to $\mathsf{UniformModulus}^{\uparrow}$. □

Theorem 914.23 (Compact net modulus selector carrier admission scope)**.** *For an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

carrier admission exposes the compact source metric row X, target metric row Y, tolerance row E, finite probe bundle $\mathcal{P}$, coverage ledger C, subordinate radius row R, pointwise-modulus row M, lower-bound fold row F, selected precision row D, transport rows H, continuation routes Q, provenance P, and local naming row N as one finite surface. A consumer entering $\mathsf{UniformModulus}^{\uparrow}$

through this carrier must keep all public reads on those rows and cannot replace them by open-cover compactness, a host minimum, a choice-selected center, quotient point equality, or an external modulus function.

Proof. Project the carrier through Definition 914.2. Carrier admission and the local NameCert clauses expose exactly the displayed tuple S and bind the compact-net, pointwise-modulus, lower-bound, transport, route, provenance, and naming rows to the same packet.

Use Definition 914.12. The finite probe surface is admitted only as the displayed bundle with its local rows, so the carrier cannot be read as a hidden open-cover compactness principle or as a selected-center choice.

Finally apply Definition 914.13 and Definition 914.22. Field faithfulness preserves the tuple coordinates, and the public handoff delivers the same finite selector surface to $\mathsf{UniformModulus}^{\uparrow}$. □

Theorem 914.24 (Compact net modulus selector lower-bound route totality)**.** *For an accepted carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

every public read of the selected precision row D is total only through the finite lower-bound fold row F over $\mathcal{P}$ and R, followed by the route rows H, Q, P, N that deliver the selector into $\mathsf{UniformModulus}^{\uparrow}$*. No consumer may read D as a host infimum, host minimum, ambient compactness witness, quotient of centers, or external modulus.*

Proof. Begin with Definition 914.21. It identifies D as the exact output of the displayed lower-bound fold F over the finite probe spine and subordinate radius ledger.

Apply Definition 914.2. The lower-bound clause is part of the same carrier admission surface, so the route from F to D must keep $X, Y, E, \mathcal{P}, C, R, M, H, Q, P, N$ attached.

The public handoff of Definition 914.22, together with the TasteGate boundary and field-faithful row of Definition 914.12 and Definition 914.13, then sends precisely that finite lower-bound route to $\mathsf{UniformModulus}^{\uparrow}$ and excludes every host or quotient substitute for D. □

Theorem 914.25 (Compact net modulus selector obligation envelope)**.** *The obligation envelope of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

is the finite carrier-admission row, the TasteGate nontriviality surface for $\mathcal{P}$, the field-faithful row, the lower-bound fold route F, the selected precision row D, and the public consumer route into $\mathsf{UniformModulus}^{\uparrow}$*. This envelope is exactly the surface available to downstream consumers, and it contains no open-cover compactness, sequential compactness, host minimum, choice-selected center, quotient point equality, or external modulus function.*

Proof. First read the NameCert obligations through Definition 914.2. They collect carrier admission, compact-net exactness, pointwise-modulus exactness, lower-bound fold exactness, transport stability, and non-escape into one local packet.

Next use Definition 914.21. This pins the selected precision row D to the finite fold row F over $\mathcal{P}$ and R rather than to any ambient minimum principle.

Finally combine Definition 914.22, Definition 914.12, and Definition 914.13. The handoff exports the same tuple to $\mathsf{UniformModulus}^{\uparrow}$, TasteGate keeps the probe surface finite and nontrivial, and field faithfulness prevents a consumer from replacing the displayed rows by hidden compactness, choice, quotient, or external-modulus data. □

Theorem 914.26 (Compact net modulus selector route primitive scope)**.** *Every accepted* CompactNetModulusSelec *packet binds the* $\mathsf{Compact}^{\uparrow}$ *finite-net rows,* $\mathsf{CFun}^{\uparrow}$ *pointwise-modulus rows,* $\mathsf{Met}^{\uparrow}$ *distance rows,* $\mathsf{Rat}^{\uparrow}$ *positive precision rows,* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound rows,* ProbeBundle *rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *as its complete primitive scope. The selector supplies no open-cover compactness, sequential compactness, host minima, choice-selected centers, quotient point equality, or external modulus functions.*

Proof. First project the carrier and NameCert obligations, Definition 914.1 and Definition 914.2. They display exactly the compact-net, metric, positive rational, probe, pointwise-modulus, lower-bound fold, transport, continuation, provenance, and local naming rows.

Next use the pointwise-radius route and finite-fold budget exactness, Definition 914.8 and Definition 914.21. These routes force every radius and selected precision read through $X, Y, E, \mathcal{P}, C, R, M, F, D$ before a public handoff can occur.

Finally apply the uniform-modulus public handoff and obligation envelope, Definition 914.22 and Definition 914.25. The downstream consumer receives the same finite selector packet, so any open-cover, sequential compactness, host-minimum, choice, quotient, or external-modulus object would require a primitive coordinate absent from the accepted carrier. □

Theorem 914.27 (Compact net modulus selector request route determinacy)**.** *For an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the visible tolerance row E and finite-net rows $X, \mathcal{P}, C, R$ determine the request packet read by the selector. A public request consumer may continue through M, F, D, H, Q, P, N, but it cannot replace the request by open-cover compactness, selected centers, host metric data, quotient point equality, or an external modulus function.

Proof. Begin with the request obligation surface, Definition 914.11. It binds the tolerance, finite-net, metric, and pointwise-radius rows to the same selector request before the selected precision can be read.

Next apply Definition 914.2. The carrier admission and local naming clauses keep E and $X, \mathcal{P}, C, R$ inside one accepted packet with the pointwise-modulus and fold rows.

Finally use Definition 914.13 and Definition 914.22. Field faithfulness preserves the displayed rows, and the handoff sends exactly that request packet to $\mathsf{UniformModulus}^{\uparrow}$ without adding host compactness, selected centers, quotient equality, or an external modulus. □

Theorem 914.28 (Compact net modulus selector fold consumer exhaustion)**.** *Every public read of the selected precision row D is exhausted by the finite fold row F over the same* ProbeBundle $\mathcal{P}$ *and radius ledger R. The read retains the compact, continuous, metric, rational, transport, route, provenance, and local naming rows of S, and it exports no host minimum, selected-center choice, quotient point equality, or open-cover compactness witness.*

Proof. Start from Definition 914.21. It identifies D as the output of the displayed fold F over $\mathcal{P}$ and R, not as a host minimum or external rational infimum.

Use Definition 914.3 together with Definition 914.8. The fold reads only the bundled probes and subordinate radii supplied by the same metric and pointwise-modulus rows.

Apply Definition 914.2 and Definition 914.13. These keep the fold, selected precision, transports, routes, provenance, and local naming rows attached to the carrier, so no unlisted center, quotient equality, or host minimum can supply D. □

Theorem 914.29 (Compact net modulus selector uniform request factorization)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet factors through the compact, continuous, metric,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows displayed by Definition 914.26. The factorization exports the selected request and precision boundary only through those rows.*

Proof. First combine Definition 914.27 with Definition 914.28. The request is fixed by the tolerance and finite-net rows, and the selected precision is fixed by the finite fold over the same probe bundle and radius ledger.

Next apply Definition 914.2. The carrier, compact-net, pointwise-modulus, lower-bound, transport, route, provenance, and local naming clauses keep these request rows in one finite packet.

Finally use Definition 914.22 and Definition 914.26. They pass the same packet to $\mathsf{UniformModulus}^{\uparrow}$ and identify all public primitive rows, so no open-cover compactness, host metric primitive, selected center, quotient point equality, or external modulus remains available. □

Theorem 914.30 (Compact net modulus selector selected precision exactness)**.** *For an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *carrier*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

a downstream $\mathsf{UniformModulus}^{\uparrow}$ *consumer reads the selected precision row* D *if and only if the same visible tolerance row* E*, finite probe spine* $\mathcal{P}$*, coverage row* C*, subordinate radius rows* R*, pointwise-modulus rows* M*, fold row* F*, transports* H*, routes* Q*, provenance* P*, and local* $\mathsf{NameCert}$ *row* N *are displayed. The read is therefore completely determined by those visible rows and exports no open-cover compactness, host minimum, quotient point equality, choice-selected center, or external modulus function.*

Proof. For necessity, start with Definition 914.29. Any downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer must factor through the displayed selector route before it can read D.

Apply Definition 914.2. The carrier, compact-net, pointwise-modulus, lower-bound fold, transport, route, provenance, and local naming clauses keep $E, \mathcal{P}, C, R, M, F, H, Q, P, N$ attached to the same accepted packet.

For sufficiency, use Definition 914.3. Once the finite probe spine, radius ledger, pointwise-modulus rows, and fold row are displayed inside the same packet, D is the selected finite lower-bound read rather than a host minimum.

Finally apply Definition 914.19 and Definition 914.29. The ledger route and uniform request factorization carry exactly that visible packet to the downstream consumer, so open-cover compactness, host minima, quotient point equality, choice-selected centers, and external modulus functions have no coordinate from which to determine D. □

Theorem 914.31 (CompactNetModulusSelector finite-net budget route)**.** *For an accepted* $\mathsf{CompactNetModulusSel}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

every modulus read factors through the $\mathsf{Compact}^{\uparrow}$ *finite-net rows* $X, \mathcal{P}, C, R$*, the* $\mathsf{Met}^{\uparrow}$ *distance rows* X, Y*, the* $\mathsf{Rat}^{\uparrow}$ *tolerance row* E*, and the* $\mathsf{UniformModulus}^{\uparrow}$ *request rows before exposing the selected net budget* D*. The route exports no open-cover compactness, choice-selected point, host real metric, quotient center equality, or external modulus function.*

Proof. Project the carrier through Definition 914.23. It exposes the compact finite-net rows, metric rows, rational tolerance, pointwise-modulus row, lower-bound fold, selected precision, transport, route, provenance, and local naming rows as one finite surface.

Use the pointwise-radius and lower-bound routes, Definition 914.8 and Definition 914.24. They force the selected budget to pass through the displayed finite probe spine, subordinate radii, and lower-bound fold before D is readable.

Apply the uniform-modulus public handoff and route primitive scope, Definition 914.22 and Definition 914.26. These rows deliver the same finite selector packet into $\mathsf{UniformModulus}^{\uparrow}$ and exclude open-cover compactness, host metric supply, selected centers, quotient point equality, and external modulus functions. □

Theorem 914.32 (CompactNetModulusSelector primitive-scope closure)**.** *For every accepted* CompactNetModulus *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the public selector route is scoped exactly by the $\mathsf{Compact}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert *rows displayed in Definition 914.1. Every* $\mathsf{UniformModulus}^{\uparrow}$ *read of D factors through those rows, and the exported surface contains no open-cover compactness, sequential compactness, host minimum, selected center, quotient point equality, or external modulus function.*

Proof. Project the accepted packet through Definition 914.1. This reads $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$ as the whole finite selector carrier and leaves no additional compactness or modulus coordinate.

Apply Definition 914.26. It identifies the primitive public route as the displayed compact, continuous, metric, rational, finite-fold, probe-bundle, history, transport, continuation, provenance, and local naming rows.

Use Definition 914.31 to bind the selected budget D to the compact finite-net rows, pointwise-radius route, and finite lower-bound fold before any uniform-modulus consumer can read it.

Finally combine Definition 914.22 with Definition 914.25. The downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff therefore sees only the finite selector route and not open-cover compactness, sequential compactness, a host minimum, a selected center, quotient point equality, or an external modulus function. □

Theorem 914.33 (CompactNetModulusSelector center-radius monotonicity)**.** *For every accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the selected precision row D stays below every displayed center radius in the finite radius ledger R as read through the finite fold row F. The comparison is a finite selector comparison inside the same packet and not an open-cover compactness argument, host minimum, quotient point equality, choice-selected center, or external modulus computation.

Proof. Start with fold soundness and lower-bound route totality, Definition 914.3 and Definition 914.24. They identify D as the lower-bound fold output read from the same nonempty probe spine and displayed radius ledger.

Next apply finite-fold budget exactness, Definition 914.21. The comparison of D with each displayed center radius is made through F over $\mathcal{P}$ and R, so no ambient minimum or quotient equality can supply a different bound.

Finally use primitive-scope closure from Definition 914.32. The public selector route contains only $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$, which keeps the center-radius monotonicity inside the finite packet. □

Theorem 914.34 (CompactNetModulusSelector fold-output positive)**.** *For every accepted* CompactNetModulusSel *packet $S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$, the fold output D remains a positive* $\mathsf{Rat}^{\uparrow}$ *precision history. Its positivity is read through the same nonempty finite probe spine $\mathcal{P}$ and radius ledger R that feed the fold row F, and not through a host minimum, open-cover compactness principle, selected center, quotient point equality, or external modulus.*

Proof. Begin with the accepted carrier and fold soundness, Definition 914.1 and Definition 914.3. The carrier stores a positive tolerance history and positive subordinate radius rows, while fold soundness reads the output through the displayed finite fold.

Use lower-bound route totality and finite-fold budget exactness, Definition 914.24 and Definition 914.21. These results keep D on the same nonempty probe spine and radius ledger, so the fold output is a $\mathsf{Rat}^{\uparrow}$ precision history rather than a host-selected real number.

Finally apply center-radius monotonicity from Definition 914.33. Since D is read below the displayed positive radii through F, positivity remains part of the finite selector packet and cannot be replaced by an external compactness or modulus object. □

Theorem 914.35 (CompactNetModulusSelector uniform-consumer precision lock)**.** *Every downstream compact-uniform-continuity consumer of an accepted* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet receives only the finite selector route*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N.$$

The consumer cannot replace the selected precision row by open-cover compactness, a host minimum, quotient point equality, a choice-selected center, or an external modulus function.

Proof. Start with the uniform-modulus public handoff, Definition 914.22. It sends the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer the same finite selector packet rather than an ambient compactness or modulus object.

Next combine finite-net budget routing with fold-output positivity, Definition 914.31 and Definition 914.34. The selected precision row is a positive $\mathsf{Rat}^{\uparrow}$ fold output tied to $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$.

Finally apply primitive-scope closure from Definition 914.32. The primitive route exposes no open-cover compactness, host minimum, quotient point equality, selected-center oracle, or external modulus coordinate, so the uniform consumer is locked to the displayed precision row D. □

Definition 914.36 (Compact net modulus selector root readiness)**.** *The root-readiness packet for* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *is the finite tuple*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$$

read with the following public roles:

1. *$X, \mathcal{P}, C$ are the compact-net source rows from* $\mathsf{Compact}^{\uparrow}$*, including the finite center spine and its coverage rows;*
2. *R, M are the subordinate positive* $\mathsf{Rat}^{\uparrow}$ *radius rows and the* $\mathsf{CFun}^{\uparrow}$ *pointwise-modulus rows attached to the same centers and tolerance row E;*
3. *F, D are the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *fold output and the selected positive precision row exported from that fold;*

4. *H, Q are the displayed* hsame *transport rows and* Cont *consumer routes through the metric, compact, continuous, rational, fold, and uniform-modulus interfaces;*

5. *P, N are the package provenance and the local* $\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}^{\uparrow}}$ *row.*

Root readiness means that these rows are sufficient for downstream uniform-modulus consumers to read the selector as a finite compact-net handoff: compact coverage is exposed through $X, \mathcal{P}, C$, pointwise-radius control through R, M, finite lower-bound selection through F, D, and all public transport through H, Q, P, N.

Theorem 914.37 (Compact net modulus selector finite-net exposure)**.** *For every accepted root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

each selected precision row exposes a nonempty finite ProbeBundle *spine $\mathcal{P}$, the compact coverage rows C over the source row X, and the subordinate positive* $\mathsf{Rat}^{\uparrow}$ *radius rows R used by the finite fold. The exposure is public only through the compact-net source rows, the pointwise-modulus rows M, and the selected fold rows F, D of Definition 914.36.*

Proof. Read the root-ready tuple through Definition 914.1 and Definition 914.36. The compact part of the packet is exactly $X, \mathcal{P}, C, R$, with $\mathcal{P}$ named as the nonempty finite probe spine and C naming the coverage rows over the source metric.

Use Definition 914.2. Its compact-net exactness clause keeps $X, \mathcal{P}, C, R$ attached to the local NameCert row, while its pointwise-modulus clause ties each row of M to the same centers and tolerance history E.

Finally apply Definition 914.3. The selected precision row D is read through the finite fold F over the same probe spine and radius ledger, so the downstream consumer sees finite compact-net exposure rather than an ambient compactness principle. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorFiniteNetExposure`

Theorem 914.38 (Compact net modulus selector totally bounded handoff)**.** *Every downstream* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of a root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet receives the compact finite-net rows only through the displayed fold row F and selected precision row D. The handoff preserves the source rows $X, \mathcal{P}, C$, the subordinate radius rows R, the pointwise-modulus rows M, and the routes H, Q, P, N, but it does not supply any separate compactness witness or external modulus function.*

Proof. Start with the root obligation triad, Definition 914.7. It identifies the public surface as finite-net input, pointwise-radius input, and metric-selector output on the same carrier.

Next use Definition 914.31 and Definition 914.32. These theorems route every modulus read through the compact, continuous, metric, rational, finite-fold, transport, continuation, provenance, and local naming rows displayed by the selector.

The sibling chapters supply the source interfaces: chapter 60 supplies compact finite-net rows, chapter 59 supplies pointwise-modulus rows, chapter 57 supplies metric reads, chapter 37 supplies positive rational order rows, chapter 619 supplies the finite fold, and chapter 369 reads the selected precision boundary. The selector hands them forward only as F, D together with the displayed route rows. □

Theorem 914.39 (Compact net modulus selector consumer non-escape)**.** *No downstream consumer of a root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet can read open-cover compactness, a host minimum, a choice-selected center, quotient point equality, sequential compactness, or an external modulus function. Every consumer read factors through*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$$

as displayed in Definition 914.36.

Proof. Project the packet through Definition 914.36. Its public coordinates are the compact-net rows, pointwise-radius rows, finite fold, selected precision, transport rows, continuation routes, provenance, and local name.

Apply Definition 914.4 and Definition 914.9. They already forbid consumer reads outside the displayed selector packet and identify the delivered boundary as a finite metric selector packet.

Finally use Definition 914.30 and Definition 914.32. The selected precision is determined by the visible finite-net, radius, pointwise-modulus, fold, transport, route, provenance, and naming rows, leaving no coordinate for open-cover compactness, host minima, choice-selected centers, quotient point equality, sequential compactness, or external modulus functions. □

Theorem 914.40 (Compact net modulus selector cover-radius synchronization)**.** *For every accepted root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the compact cover rows C and the positive $\mathsf{Rat}^{\uparrow}$ *radius rows R are synchronized over the same finite probe spine $\mathcal{P}$. A cover read through C cannot be consumed without the corresponding positive radius ledger R, and a radius read through R cannot be consumed without the compact-net cover rows C over the source metric row X.*

Proof. Project S through Definition 914.36. The displayed carrier stores $X, \mathcal{P}, C, R$ as the compact finite-net input and makes R the subordinate positive-radius ledger attached to that same probe spine.

Use Definition 914.37. It exposes the compact coverage rows and the positive radius rows together, with no separate coordinate for a detached cover row or a detached radius row.

Finally apply Definition 914.39. Since every downstream read factors through the displayed tuple, a consumer cannot separate C from R while remaining inside the accepted selector surface. □

Theorem 914.41 (Compact net modulus selector fold-output positivity)**.** *For every accepted root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the fold output row D is a positive precision row only through the positive $\mathsf{Rat}^{\uparrow}$ *radius ledger R, the pointwise-modulus rows M, and the finite lower-bound fold row F. No consumer may treat D as a host minimum, a nonpositive bound, or a choice-selected external modulus value.*

Proof. Read D through Definition 914.3. The finite fold row F computes the selected lower-bound precision from the positive radius and pointwise-modulus rows already present in the carrier.

Use Definition 914.30. It fixes the precision row by the visible compact-net, radius, pointwise-modulus, fold, transport, route, provenance, and naming rows, so positivity is inherited from the $\mathsf{Rat}^{\uparrow}$ radius ledger and the admitted fold route.

Definition 914.39 excludes host minima, choice-selected centers, quotient point equality, and external modulus functions. Thus the only public interpretation of D is the positive fold output carried by R, M, F inside the displayed packet. □

Theorem 914.42 (Compact net modulus selector request-to-precision factorization)**.** *Every tolerance request E accepted by a root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet factors to the selected precision row D through*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N.$$

The handoff first reads compact coverage through $X, \mathcal{P}, C$, then positive subordinate radii through R, then pointwise-modulus rows through M, then the finite fold output through F, D, and finally the transport, route, provenance, and local certificate rows H, Q, P, N.

Proof. Begin with Definition 914.36. The tolerance row E is displayed with the source and target metric rows, the compact finite-net input, the pointwise-modulus rows, and the selected fold output.

Apply Definition 914.40. It forces every compact cover read to carry the matching positive radius ledger, so the request cannot bypass $X, \mathcal{P}, C, R$.

Next apply Definition 914.41. The fold output D is admitted only through R, M, F, and therefore the tolerance request reaches a precision row only through the finite selector fold.

Finally use Definition 914.39. The displayed transport, route, provenance, and local naming rows H, Q, P, N are the only remaining public coordinates, so the request-to-precision handoff factors through exactly the root-ready carrier. □

Theorem 914.43 (Compact net modulus selector fold-consumer coherence)**.** *Every fold consumer of a root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet that preserves the selected precision row D also preserves the same* ProbeBundle *spine $\mathcal{P}$ and positive radius ledger R. Consequently the fold handoff cannot be reinterpreted as a host minimum, quotient center equality, or external modulus function.*

Proof. Project the consumer read through Definition 914.1. The only fold data in the carrier are the finite fold row F, the selected precision row D, the bundled centers $\mathcal{P}$, and the radius ledger R.

Now use Definition 914.36 and Definition 914.37. They identify D as the fold-selected precision attached to the same finite probe spine and subordinate positive radius rows.

Apply Definition 914.38. Any consumer preserving D along the finite fold route receives the compact finite-net rows through F, D and therefore keeps $\mathcal{P}$ and R in the same public ledger.

Definition 914.39 excludes every alternate read of a host minimum, quotient center equality, and external modulus function. The preserved selected precision is therefore coherent only with the original probe bundle and radius ledger. □

Theorem 914.44 (Compact net modulus selector root-budget admission)**.** *For every accepted root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

the selector budget D is admitted for downstream use only with the displayed finite probe spine $\mathcal{P}$, compact coverage row C, positive radius rows R, pointwise-modulus rows M, finite fold row F, selected precision row D, transport rows H, route rows Q, provenance row P, and local NameCert *row N. Thus the budget is a finite root-ready handoff and not a parameter-only compactness promise.*

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorRootBudgetAdmission`

Proof. Read the packet through Definition 914.1 and Definition 914.36. These two displays enumerate exactly the root rows $X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$ and assign the budget role to the selected precision row D.

Apply Definition 914.37. It binds D to the nonempty finite probe spine, compact coverage rows, subordinate positive radius ledger, pointwise-modulus rows, and selected fold output.

Use Definition 914.38 to pass the same rows to the downstream compact-net and uniform-modulus consumers. The handoff keeps the finite fold and selected precision visible together with the transport, route, provenance, and local naming rows.

Finally invoke Definition 914.39. Since no consumer can leave the displayed packet, a budget read without any one of the root rows is not an admitted selector read. □

Theorem 914.45 (Compact net modulus selector uniform-modulus entry exhaustion)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *entry through a root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet exhausts the visible route through* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{CFun}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No additional entry point supplies compactness, continuity, metric control, rational positivity, finite selection, history transport, package provenance, or local naming data.*

Proof. Begin with Definition 914.1. It lists the compact source rows, continuous pointwise-modulus rows, metric rows, rational tolerance and radius rows, finite fold data, history transport, continuation route, package provenance, and local name certificate.

Definition 914.36 turns that carrier list into the root-ready packet. The displayed roles specify how $X, \mathcal{P}, C, R, M, F, D, H, Q, P, N$ become the finite selector handoff.

Use Definition 914.37 and Definition 914.38. Together they force the selected precision read through the compact finite-net spine, pointwise-modulus rows, positive rational ledger, finite fold, and route rows before it reaches $\mathsf{UniformModulus}^{\uparrow}$.

Finally apply Definition 914.39. Since every downstream read factors through the root-ready tuple, the visible route is exhausted and no separate entry point can supply the omitted compactness, continuity, metric, rational, fold, history, transport, provenance, or naming data. □

Theorem 914.46 (Compact net modulus selector downstream coverage)**.** *For every accepted root-ready* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet*

$$S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N),$$

every accepted local consumer of the selected budget D is exhausted by the finite alternatives already displayed in the selector file: the compact coverage rows $X, \mathcal{P}, C$, the positive radius ledger R, the pointwise-modulus rows M, the finite fold and selected precision rows F, D, and the transport, route, provenance, and local naming rows H, Q, P, N. Any requested alternative that asks for open-cover compactness, sequential compactness, a host minimum, a choice-selected center, quotient point equality, a detached compactness witness, or an external modulus function is not an accepted local consumer of the $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *packet.*

Proof. Start with the root budget admission of Definition 914.44. It admits the selected budget D for downstream use only together with the displayed finite probe spine $\mathcal{P}$, compact coverage row C, positive radius rows R, pointwise-modulus rows M, finite fold row F, selected precision row D, transport rows H, route rows Q, provenance row P, and local $\mathsf{NameCert}$ row N. Thus an accepted consumer cannot read D as a parameter-only modulus promise.

Next use Definition 914.45. A $\mathsf{UniformModulus}^{\uparrow}$ entry through the same root-ready packet exhausts the visible route through $\mathsf{Compact}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, BHist,

hsame, Cont, Pkg, and NameCert. These are precisely the typed interfaces carried by the displayed rows of S: compact coverage by $X, \mathcal{P}, C$, rational radius and tolerance control by E, R, D, pointwise continuity by M, finite selection by F, and structural delivery by H, Q, P, N.

Combine the two readings. The admission theorem fixes which root rows may accompany the budget, while the exhaustion theorem fixes which downstream entry points can consume those rows. Therefore every accepted local consumer is already one of the enumerated coverage, ledger, fold, route, provenance, or naming reads in the selected file.

Finally apply the exclusion clauses in both the root admission and entry exhaustion statements. Since no coordinate of S supplies open-cover compactness, sequential compactness, host minima, choice-selected centers, quotient point equality, detached compactness witnesses, or external modulus functions, any request for one of those alternatives fails to remain inside the accepted CompactNetModulusSelector$^{\uparrow}$ packet. □

Theorem 914.47 (Compact net modulus selector public modulus handoff)**.** *Every public* UniformModulus$^{\uparrow}$ *read from an accepted* CompactNetModulusSelector$^{\uparrow}$ *packet factors through the finite selector rows*

$$X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N$$

of Definition 914.1. The public handoff exposes the compact metric source through CompactMetric$^{\uparrow}$ *and* Met$^{\uparrow}$ *rows, the selected precision through* UniformModulus$^{\uparrow}$ *and* FiniteNetMinimumFold$^{\uparrow}$ *rows, the dyadic and regular request boundary through* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*, and* RegSeqRat$^{\uparrow}$ *rows, and the structural boundary through* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It is grounded at the TasteGate surface of Definition 914.6 and exports no open-cover compactness, host minimum, selected center, quotient point equality, sequential compactness, or external modulus function.*

Proof. Project the accepted selector carrier. The rows $X, Y, E, \mathcal{P}, C, R, M, F, D$ are the only compact-net, metric, tolerance, pointwise-modulus, fold, and selected-precision data available to a public consumer.

Apply Definition 914.10. It forces every downstream UniformModulus$^{\uparrow}$ read through the compact finite-net input, pointwise continuous radius rows, metric rows, rational tolerance row, lower-bound fold, selected precision, and structural rows of the same packet.

Apply Definition 914.42 and Definition 914.43. These results bind the tolerance request to the finite probe spine, positive radius ledger, pointwise-modulus rows, fold output, and selected precision before any public modulus read is exported.

The TasteGate alignment cited in the statement preserves the same single carrier as a field-faithful public packet. Since no excluded object appears among the carrier rows or the cited handoff theorems, the public surface is exactly the displayed selector-to-modulus route. □

Theorem 914.48 (Compact net modulus selector bridge envelope)**.** *The public compact-net selector handoff of Definition 914.47 is the paper-side bridge envelope from compact finite-net rows, pointwise-modulus rows, lower-bound fold rows, selected precision, and* UniformModulus$^{\uparrow}$ *consumer rows to the standard acceptance schema. For every accepted* CompactNetModulusSelector$^{\uparrow}$ *packet* $S = (X, Y, E, \mathcal{P}, C, R, M, F, D, H, Q, P, N)$*, the envelope reads the compact source rows* $X, \mathcal{P}, C, R$*, pointwise rows* M*, fold row* F*, selected precision* D*, and consumer rows* H, Q, P, N *in that order, and exports no open-cover compactness, host minimum, selected center outside* $\mathcal{P}$*, quotient point equality, sequential compactness, or external modulus function.*

Proof. Start from the accepted selector carrier and the public handoff theorem. The finite compact source is already displayed as $X, \mathcal{P}, C, R$, so the envelope begins with the compact finite-net rows rather than an open-cover compactness principle.

Next read the pointwise-modulus and fold rows. The rows M, F, D bind each pointwise radius ledger to the selected precision by the lower-bound fold, so the handoff cannot be reinterpreted as a host minimum or an external modulus.

Finally pass to the consumer rows H, Q, P, N. They provide componentwise transport, continuation replay, provenance, and local naming for the same packet, matching the acceptance schema's demand for a single visible source route.

The excluded rows are absent from the carrier and from the public handoff. Hence the displayed handoff is precisely the bridge envelope from the compact selector packet to the standard acceptance boundary. □

Falsifiable prediction: Removing the selected lower-bound row breaks the uniform-modulus handoff while leaving compact-net coverage and pointwise-modulus rows readable.

Independence witness: Compare compact, continuous, metric, finite-net-minimum-fold, and uniform-modulus siblings: none is bijective because this carrier alone stores the compact-net input, pointwise-modulus radii, selected lower-bound fold row, and tolerance-only handoff together.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactNetModulusSelector}^{\uparrow}$ packet is generated from a compact metric finite-net row, a target metric row, a positive tolerance history, a nonempty ProbeBundle of centers, positive subordinate radius rows, pointwise-modulus rows at the bundled centers, a finite lower-bound fold, selected precision, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.CompactNetModulusSelectorUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 914.1, Definition 914.2, Definition 914.40, Definition 914.41, Definition 914.42, Definition 914.3, Definition 914.4, Definition 914.5, Definition 914.47, and Definition 914.48 bind the finite compact-net selector packet, its local NameCert obligations, cover-radius synchronization, fold-output positivity, request-to-precision factorization, fold soundness, non-escape boundary, sibling independence boundary, public uniform-modulus handoff, and paper-side bridge envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactNetModulusSelector}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CompactNetModulusSelectorUp.CompactNetModulusSelectorTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridge checking, open-cover compactness, sequential compactness, a host minimum, a choice-selected center, quotient point equality, or an external modulus function for $\mathsf{CompactNetModulusSelector}^{\uparrow}$.

Upgrade path: Bridge checking remains the formal-axis task; the paper closure already binds the compact-net selector packet through the public handoff and bridge envelope.

Chapter 915

A Concrete Naming Certificate for $\mathsf{KernelInductiveAcceptance}^{\uparrow}$

The $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ packet realizes the inductive-declaration branch of chapter 9364 as finite NameCert data. It sits beside the accepted-declaration packet of chapter 549, the generic acceptance trace of chapter 557, the generator closure classifier of chapter 486, the type-classifier membership surface of chapter 476, and the subject-reduction boundary of chapter 483. Those siblings expose acceptance, trace, closure, membership, and preservation rows; this packet isolates the constructor-signature plus emitted-eliminator ledger required for accepted inductive declarations.

Definition 915.1 (KernelInductiveAcceptance carrier)**.** *A* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, S, E, P, R, H, C, Q, N)$$

with the following displayed rows:

1. *D is the inductive declaration candidate row, including the carrier name, parameters, indices, universe data, and constructor-name inventory;*
2. *S is the constructor-signature ledger, recording the finite domain and target signature of every constructor accepted for D;*
3. *E is the emitted-eliminator ledger, recording the recursor, induction principle, and computation rows generated from the accepted signature;*
4. *P is the positivity-refusal row, recording that a candidate whose constructor signature violates the local positivity discipline is refused rather than entered into the environment;*
5. *R is the recursion-discipline row, recording that eliminator and recursor consumers may read only the emitted rows in E;*
6. *H records componentwise* hsame *transports for D, S, E, P, R, Q, N;*
7. *C records the displayed* Cont *routes from the declaration candidate to constructor signatures, from constructor signatures to emitted eliminators, from failed signatures to positivity refusal, and from emitted eliminators to downstream consumers;*
8. *Q is the* Pkg *provenance over chapter 9364, chapter 549, chapter 557, chapter 486, chapter 476, chapter 483,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}$ *row.*

The carrier contains no theorem-proof row, no generic theorem trace, no generator-closure theorem, no type-membership theorem, and no subject-reduction target row beyond the emitted eliminator ledger displayed in E.

Lean4 (definition) `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp`

Theorem 915.2 (KernelInductiveAcceptance NameCert obligations)**.** *For the carrier of Definition 915.1, the local* $\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}$ *surface has five rows:*

1. *carrier admission for the finite packet $I = (D, S, E, P, R, H, C, Q, N)$;*
2. *constructor-signature exactness, requiring every accepted constructor read to be one of the displayed entries of S;*
3. *emitted-eliminator ledger exactness, requiring every recursor, induction-principle, and computation read to pass through E after S is visible;*
4. *positivity-refusal stability, requiring a failed positivity check to land in P rather than in an environment-entry row;*
5. *environment-entry non-escape, requiring downstream acceptance, trace, generator, type-classifier, and subject-reduction consumers to read only $D, S, E, P, R, H, C, Q, N$ and not a hidden kernel-correctness theorem.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp namecert obligations`

Proof. Carrier admission is projection from the displayed packet. The declaration candidate, constructor-signature ledger, emitted-eliminator ledger, positivity refusal, recursion discipline, transport, route, provenance, and local name are all coordinates of the same finite BHist object.

Constructor-signature exactness and emitted-eliminator exactness are the two accepted rows. The accepted constructor inventory is read from S, and each eliminator or recursor consumer must first expose S before reading the generated ledger E.

The refusal and environment clauses separate accepted and rejected packets. A failed positivity read lands in P and never reaches the environment-entry route, while an accepted read may enter the environment only through the displayed D, S, E, R, H, C, Q, N rows.

The sibling chapters then consume only the finite rows they name. KernelAcceptanceWitness and KernelAcceptanceTrace can read the environment entry, GeneratorClosure can read the accepted generator boundary, TypeClassifierMembership can read emitted type-classifier data, and SubjectReduction can read recursor-facing preservation boundaries; none of those reads adds an unrecorded proof of kernel correctness. □

Theorem 915.3 (KernelInductiveAcceptance nontrivial boundary)**.** *The* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet has a nontrivial boundary for TasteGate: an accepted packet displays at least one accepted environment-entry route through D, S, E, R, H, C, Q, N, while a candidate whose constructor signature lacks the required positivity row lands in P and is not accepted. Phase C must realize this as a Nontrivial* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *instance in* $\mathsf{BEDC.Derived.KernelInductiveAcceptanceUp.TasteGate}$.

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceNontrivialBoundary`

Proof. Use the carrier to separate accepted and refused rows. Accepted packets expose constructor signatures and emitted eliminators through S and E, while refused packets expose the positivity boundary P.

The NameCert obligation surface makes the separation observable. Environment-entry non-escape permits an accepted route only after the displayed constructor-signature and eliminator ledgers are present, and positivity-refusal stability prevents a failed candidate from taking that route.

Thus the TasteGate boundary is not empty: it has an accepted side and a refused side, both represented by finite rows of the same carrier. □

Theorem 915.4 (KernelInductiveAcceptance field-faithful boundary)**.** *The* KernelInductiveAcceptance$^{\uparrow}$ *packet is field-faithful for TasteGate: any accepted read that preserves $D, S, E, P, R, H, C, Q, N$ componentwise preserves the inductive-declaration acceptance row, and any read that drops either the constructor-signature ledger S or the emitted-eliminator ledger E is not a faithful read of this carrier. Phase C must realize this as a FieldFaithful* KernelInductiveAcceptance$^{\uparrow}$ *instance in* BEDC.Derived.KernelInductiveAcceptanceUp.TasteGate*.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceFieldFaithfulBoundary`

Proof. Start with two accepted reads matched componentwise by the displayed hsame row. The declaration candidate, signature ledger, eliminator ledger, refusal boundary, recursion discipline, routes, provenance, and local name are therefore transported together.

Apply constructor-signature exactness and emitted-eliminator exactness from the NameCert surface. These two rows are the fields that distinguish inductive acceptance from a generic accepted declaration or trace.

If either row is absent, the carrier no longer exposes the paired signature-plus-eliminator surface. The read may belong to a sibling acceptance packet, but it is not faithful to KernelInductiveAcceptance$^{\uparrow}$. □

Theorem 915.5 (KernelInductiveAcceptance constructor ledger exhaustion)**.** *Every accepted constructor read of a* KernelInductiveAcceptance$^{\uparrow}$ *packet is exhausted by the displayed declaration row D, constructor-signature ledger S, componentwise* hsame *transports H, displayed* Cont *routes C, package provenance Q, and local* NameCert *row N. No accepted constructor consumer can read a constructor signature outside the finite ledger S.*

Proof. Use the carrier of Definition 915.1. The declaration candidate D, constructor-signature ledger S, transports H, routes C, provenance Q, and naming row N are displayed coordinates of the same finite BHist packet.

Apply the constructor-signature exactness clause of Definition 915.2. It requires every accepted constructor read to be one of the displayed entries of S, with its declaration source and route recorded by the surrounding rows.

The TasteGate commitments are preserved because the read is still through the paired inductive-declaration surface. A consumer that drops S or replaces it by a hidden theorem-proof row leaves the field-faithful boundary of Definition 915.4. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceConstructorLedgerExhaustion`

Theorem 915.6 (KernelInductiveAcceptance eliminator ledger exhaustion)**.** *Every recursor, induction-principle, or computation read of an accepted* KernelInductiveAcceptance$^{\uparrow}$ *packet factors through the constructor-signature ledger S before entering the emitted-eliminator ledger E. No eliminator consumer may read a hidden theorem-proof row or bypass S on the way to E.*

Proof. Use the carrier of Definition 915.1. The rows D, S, E, R, H, C, Q, N are displayed coordinates of the same finite BHist packet, so a recursor or induction-principle consumer has no carrier coordinate outside them.

Apply the constructor-signature exactness and emitted-eliminator exactness clauses of Definition 915.2. They force each accepted eliminator read to expose S before E.

Finally use the sibling anchors named in the chapter introduction. The accepted-declaration, trace, generator, type-classifier, and subject-reduction chapters consume the emitted route after E is visible, but none of them inserts an extra theorem-proof row between S and E. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp eliminator ledger exhaustion`

Theorem 915.7 (KernelInductiveAcceptance refusal nonescape)**.** *Every constructor-signature candidate for* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *that fails the local positivity row lands in the refusal row P and cannot be transported into the accepted environment-entry route D, S, E, R, H, C, Q, N. The failed candidate remains outside the accepted declaration, trace, generator, type-classifier, and subject-reduction consumer paths.*

Proof. Begin with the carrier row P of Definition 915.1. It is the displayed place for failed positivity candidates, distinct from the accepted constructor-signature ledger S and emitted-eliminator ledger E.

Apply the positivity-refusal stability clause of Definition 915.2. A failed signature is routed to P rather than to the accepted environment-entry path.

The nontrivial boundary of Definition 915.3 separates the accepted and refused sides of the TasteGate surface. Thus the failed candidate cannot be read by the accepted-declaration, trace, generator, type-classifier, or subject-reduction consumers as an entered inductive declaration. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp refusal nonescape`

Theorem 915.8 (KernelInductiveAcceptance obligation closure package)**.** *The carrier of Definition 915.1, NameCert obligations of Definition 915.2, TasteGate nontrivial boundary of Definition 915.3, field-faithful boundary of Definition 915.4, constructor-ledger exhaustion of Definition 915.5, eliminator-ledger exhaustion of Definition 915.6, and positivity-refusal nonescape of Definition 915.7 assemble the full paper-side obligation surface for* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$*. The assembled package exports no theorem-proof row, generic theorem trace, generator-closure theorem, type-membership theorem, subject-reduction target, or hidden kernel-correctness theorem.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceObligationClosurePackage`

Proof. Project the carrier tuple $I = (D, S, E, P, R, H, C, Q, N)$. These coordinates display the declaration candidate, constructor-signature ledger, emitted-eliminator ledger, positivity refusal, recursion discipline, transport, continuation, provenance, and local naming rows.

Apply Definition 915.2. Its five rows give carrier admission, constructor-signature exactness, emitted-eliminator exactness, positivity-refusal stability, and environment-entry non-escape for the same finite packet.

Use Definition 915.3 and Definition 915.4. They enforce the AI TasteGate discipline by keeping the accepted and refused sides distinct and by making faithful reads preserve the paired constructor-signature plus emitted-eliminator fields.

The accepted read surface is exhausted by Definition 915.5 and Definition 915.6. Constructor consumers return to S, and recursor, induction-principle, and computation consumers pass through S before E.

Finally apply Definition 915.7. Failed positivity routes stay in P, so the package cannot leak into accepted declaration, trace, generator, type-classifier, or subject-reduction consumers through a hidden kernel-correctness theorem. □

Theorem 915.9 (KernelInductiveAcceptance accepted declaration route)**.** *Every accepted-declaration consumer of a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet must pass through the displayed rows* D, S, E, R, H, C, Q, N *of the carrier. The consumer may read the declaration candidate, constructor-signature ledger, emitted-eliminator ledger, recursion discipline, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows, but it receives no hidden theorem-proof row and no accepted-declaration certificate outside those displayed rows.*

Proof. Project the carrier of Definition 915.1. Its accepted side is the finite route D, S, E, R, H, C, Q, N, with the refused positivity candidates separated into P.

Apply the NameCert obligations of Definition 915.2. Carrier admission, constructor-signature exactness, emitted-eliminator exactness, and environment-entry non-escape force an accepted-declaration read to expose D, then S, then E, with R, H, C, Q, N carrying the route structure.

The field-faithful boundary of Definition 915.4 prevents a consumer from dropping the signature or eliminator ledgers. The obligation package of Definition 915.8, together with the existing TasteGate anchor recorded by this chapter, keeps the AI TasteGate read anchored in the displayed carrier rather than in a hidden theorem-proof row. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp accepted declaration route`

Theorem 915.10 (KernelInductiveAcceptance generator closure route)**.** *Every generator-closure consumer of a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet may read the constructor-signature ledger* S *and emitted-eliminator ledger* E *only through the displayed route rows* D, S, E, R, H, C, Q, N *and the local* $\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}$ *row. It cannot replace those rows by a generator-closure theorem, a theorem-proof trace, or an off-carrier closure certificate.*

Proof. Begin with constructor-ledger and eliminator-ledger exhaustion, Definition 915.5 and Definition 915.6. They identify S and E as the only visible signature and generated-eliminator ledgers available to downstream generator consumers.

Use the NameCert surface to attach those ledgers to D, R, H, C, Q, N. The displayed Cont row records the route from declaration candidate to constructor signatures and from signatures to emitted eliminators, while Q and N supply provenance and local naming.

Finally apply the obligation closure package and the field-faithful boundary. A generator-closure consumer preserving the TasteGate surface must preserve the paired S, E fields, so it cannot substitute a hidden theorem-proof trace or an off-carrier closure theorem for the displayed route. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceGeneratorClosureRoute`

Theorem 915.11 (KernelInductiveAcceptance subject reduction route)**.** *Every subject-reduction consumer of a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet receives emitted eliminator and computation rows only after the constructor signatures in* S *are visible. Failed positivity candidates remain in the refusal row* P*, and no subject-reduction consumer can transport such a candidate into the accepted route* D, S, E, R, H, C, Q, N*.*

Proof. Start with the emitted-eliminator exactness clause in Definition 915.2. It requires every recursor, induction-principle, and computation read to pass through E after the constructor-signature ledger S is visible.

Apply Definition 915.6. This pins the subject-reduction-facing rows to the displayed route D, S, E, R, H, C, Q, N and rules out a bypass from declaration candidate directly to computation rows.

It remains to separate failed positivity candidates. By Definition 915.7, such candidates land in P and remain outside accepted declaration, trace, generator, type-classifier, and subject-reduction paths. The obligation closure package and the existing TasteGate anchor recorded by this chapter therefore leave subject reduction with only the displayed accepted route and no hidden theorem-proof row. □

Theorem 915.12 (KernelInductiveAcceptance environment scope route)**.** *For the carrier $I = (D, S, E, P, R, H, C, Q, N)$ of Definition 915.1, every KernelAcceptanceWitness or KernelAcceptanceTrace consumer of an accepted inductive declaration reads the environment-entry and trace surface only through the displayed rows D, S, E, R, H, C, Q, N. The refusal row P remains visible as the rejected side, but it is not transported into the accepted environment route.*

Proof. Project the accepted declaration route of Definition 915.9. It identifies the environment-entry surface with the displayed declaration row, constructor-signature ledger, emitted-eliminator ledger, recursion row, componentwise transport, continuation route, provenance, and local naming rows.

Use constructor-ledger exhaustion and eliminator-ledger exhaustion to make the trace route determinate. KernelAcceptanceWitness and KernelAcceptanceTrace may name the accepted declaration and its trace, but their readback cannot skip S or E and cannot introduce a theorem-proof row outside the finite packet.

Finally apply refusal nonescape. Failed positivity candidates stay in P, so the trace and witness consumers see the refused side only as a boundary row and never as an accepted environment entry. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp environment scope route`

Theorem 915.13 (KernelInductiveAcceptance type-classifier scope route)**.** *Every TypeClassifierMembership consumer of a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet is scoped by the displayed constructor-signature ledger S, emitted-eliminator ledger E, componentwise* hsame *row H, displayed* Cont *row C, package provenance Q, and local* NameCert *row N. It may classify emitted inductive data only through $D, S, E, P, R, H, C, Q, N$ and may not substitute a hidden type-membership theorem for those rows.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp typeclassifier scope route`

Proof. Begin with the NameCert obligations. Constructor-signature exactness and emitted-eliminator exactness force any classifier-facing read to expose S before it reads the generated rows in E.

Use the continuation route totality theorem to bind the classifier route to C. Since the route from signatures to emitted eliminators is already displayed, a TypeClassifierMembership consumer has no off-carrier route by which to classify an unrecorded eliminator or computation row.

Transport is controlled by the componentwise hsame scope. The row H preserves S, E, P, R, Q, N, so classifier transport preserves the same finite carrier rows rather than replacing them with a hidden membership theorem. □

Theorem 915.14 (KernelInductiveAcceptance subject-reduction scope lock)**.** *SubjectReduction consumers of a* KernelInductiveAcceptance$^{\uparrow}$ *packet are locked to the emitted-eliminator route D, S, E, R, H, C, Q, N. Once S and E are fixed, a subject-reduction read cannot import an external preservation theorem, cannot read a computation row outside E, and cannot turn a failed positivity candidate from P into an accepted preservation boundary.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp subject reduction scope lock`

Proof. Start with Definition 915.11. It already requires every preservation-facing consumer to receive emitted eliminator and computation rows only after the constructor signatures in S are visible.

Apply eliminator-ledger exhaustion and continuation route totality. Together they make E the only emitted-eliminator inventory and C the only route by which a subject-reduction consumer reaches it.

It remains to rule out refused candidates. Refusal nonescape keeps failed positivity rows in P, and classifier stability prevents componentwise transport from converting such a row into an accepted route. The subject-reduction consumer is therefore locked to the displayed accepted surface. □

Theorem 915.15 (KernelInductiveAcceptance scoped consumer package)**.** *The rows $D, S, E, P, R, H, C, Q, N$ form a scoped consumer package for KernelAcceptanceWitness, KernelAcceptanceTrace, GeneratorClosure, TypeClassifierMembership, and SubjectReduction. Each named sibling may consume only the carrier-local route assigned to it by the accepted-declaration route, generator-closure route, type-classifier scope route, subject-reduction scope lock, constructor-ledger exhaustion, eliminator-ledger exhaustion, refusal nonescape, continuation totality, and componentwise transport scope.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp scoped consumer package`

Proof. First assemble the accepted environment side. The environment scope route binds KernelAcceptanceWitness and KernelAcceptanceTrace to D, S, E, R, H, C, Q, N, while refusal nonescape leaves P as the separated rejected side.

Next assemble the generator and classifier sides. The generator-closure route consumes the paired S, E ledgers through the carrier, and the type-classifier scope route makes classifier membership read the same emitted rows through H, C, Q, N.

Finally assemble the preservation side. The subject-reduction scope lock, together with eliminator-ledger exhaustion and continuation route totality, keeps SubjectReduction on the emitted-eliminator route and blocks external preservation or theorem-proof rows.

These routes exhaust the sibling consumers named in the introduction. Since the carrier contains no theorem-proof row, generic theorem trace, generator theorem, type-membership theorem, subject-reduction theorem, or hidden kernel-correctness witness, the package is scoped precisely by $D, S, E, P, R, H, C, Q, N$. □

Theorem 915.16 (KernelInductiveAcceptance environment-entry determinacy)**.** *For the carrier $I = (D, S, E, P, R, H, C, Q, N)$ of Definition 915.1, every accepted environment-entry read is determined by the displayed route D, S, E, R, H, C, Q, N. If two accepted reads preserve the declaration candidate, constructor-signature ledger, emitted-eliminator ledger, recursion-discipline row, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *row, then*

they present the same environment-entry surface for KernelInductiveAcceptance$^{\uparrow}$. *The positivity-refusal row P remains outside the accepted entry surface.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp environment entry determinacy`

Proof. Project the NameCert obligation surface of Definition 915.2. Carrier admission, constructor-signature exactness, emitted-eliminator exactness, and environment-entry non-escape identify the accepted side as the displayed D, S, E, R, H, C, Q, N route.

Use constructor-ledger exhaustion and eliminator-ledger exhaustion, Definition 915.5 and Definition 915.6. These prevent a second constructor or emitted-eliminator source from distinguishing two accepted environment entries with the same displayed rows.

Finally apply the accepted declaration route theorem Definition 915.9. It gives downstream consumers no accepted-declaration certificate outside the displayed route, while refusal nonescape keeps P separate from the accepted entry. □

Theorem 915.17 (KernelInductiveAcceptance emitted-eliminator consumer coverage)**.** *Once the constructor signatures in S are visible, every recursor consumer, induction-principle consumer, and computation-row consumer of an accepted* KernelInductiveAcceptance$^{\uparrow}$ *packet is covered by the emitted-eliminator ledger E together with D, S, R, H, C, Q, N. No such consumer reads an eliminator outside E, a theorem-proof row, a generic trace theorem, or an off-carrier generator certificate.*

Proof. Begin with the emitted-eliminator exactness clause of Definition 915.2. It requires every recursor, induction-principle, and computation read to pass through E after the constructor-signature ledger S is visible.

Apply Definition 915.6. The consumer route has no carrier coordinate outside D, S, E, R, H, C, Q, N, and the ledger E is the only emitted-eliminator inventory available to the read.

Use the generator closure route and subject reduction route, Definition 915.10 and Definition 915.11. Generator and preservation consumers are downstream of the same emitted ledger, so their coverage is exhausted by E and the displayed route rows rather than by a hidden theorem or trace certificate. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp emitted eliminator consumer coverage`

Theorem 915.18 (KernelInductiveAcceptance positivity-refusal classifier stability)**.** *Every constructor-signature candidate that fails the local positivity classifier remains in the refusal row P under componentwise classifier transport. Such a candidate cannot be transported into the accepted route D, S, E, R, H, C, Q, N, cannot acquire an emitted eliminator in E, and cannot become a subject-reduction or generator-closure consumer row.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp positivity refusal classifier stability`

Proof. Start with the positivity-refusal stability clause of Definition 915.2. A failed positivity candidate is routed to P rather than to the accepted constructor-signature ledger S.

Apply the refusal nonescape theorem Definition 915.7. It separates P from accepted declaration, trace, generator, type-classifier, and subject-reduction paths, so classifier transport cannot turn a refused candidate into an accepted environment entry.

Finally combine the accepted declaration route, generator closure route, and subject reduction route. Theorems 915.9, 915.10, and 915.11 all consume only the displayed accepted route after S

and E are visible. Since a refused candidate remains in P, it cannot enter any of those consumer rows. □

Theorem 915.19 (KernelInductiveAcceptance continuation route totality)**.** *For the carrier $I = (D, S, E, P, R, H, C, Q, N)$ of Definition 915.1, every accepted declaration route, constructor-signature route, emitted-eliminator route, positivity-refusal route, generator-closure consumer route, and subject-reduction consumer route factors through the displayed* Cont *row C. No consumer of* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *may read a declaration, constructor, eliminator, refusal, generator boundary, or preservation boundary through a route outside C.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp cont route totality`

Proof. Project the carrier tuple. The row C is the displayed continuation coordinate, and Definition 915.1 assigns to it the declaration-to-signature, signature-to-eliminator, failed-signature-to-refusal, and emitted-eliminator-to-consumer routes.

Apply constructor-ledger exhaustion and eliminator-ledger exhaustion. These identify S and E as the only accepted constructor and eliminator ledgers, so accepted declaration, constructor, and eliminator consumers can enter only through the continuation coordinates already displayed in C.

For failed signatures, use positivity-refusal nonescape. The failed candidate lands in P, and the only recorded route from failed signature data to refusal data is the corresponding component of C.

For generator and subject-reduction consumers, use Definition 915.10 and Definition 915.11. Both consumers are downstream of the emitted-eliminator surface, so their carrier-local readback is exhausted by the same $D, S, E, P, R, H, C, Q, N$ packet and cannot introduce an off-carrier continuation. □

Theorem 915.20 (KernelInductiveAcceptance hsame transport scope)**.** *Componentwise* hsame *transport through the displayed row H preserves the constructor-signature ledger S, emitted-eliminator ledger E, positivity-refusal row P, recursion row R, package provenance Q, and local* NameCert *row N of a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *carrier. The transport scope contains no theorem-proof row and no hidden kernel-correctness witness.*

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp hsame transport scope`

Proof. Start with the transport row H in Definition 915.1. Its scope is componentwise over the displayed rows D, S, E, P, R, Q, N, with C recording the routes that make those transported rows readable by consumers.

The constructor and eliminator ledgers are preserved by the field-faithful boundary and the two ledger-exhaustion theorems. A transport that drops S or E is not a faithful read of this carrier, and a transport that preserves them keeps each accepted constructor and emitted eliminator inside the finite displayed ledgers.

The refusal and recursion rows are preserved by the NameCert obligations. Positivity-refusal stability keeps failed signatures in P, and emitted-eliminator exactness makes recursor and induction-principle consumers pass through the displayed recursion row R rather than an unrecorded proof surface.

Finally, provenance and local naming are carrier coordinates. Transport through H therefore carries Q and N together with the ledgers and refusal row, leaving no place for an additional theorem-proof row or hidden kernel-correctness witness. □

Theorem 915.21 (KernelInductiveAcceptance sibling consumer scope)**.** *The sibling consumers named in the chapter introduction may consume a* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ *packet only through the displayed KernelAcceptanceWitness, KernelAcceptanceTrace, GeneratorClosure, TypeClassifierMembership, and SubjectReduction routes over* $D, S, E, P, R, H, C, Q, N$*. They may not read an accepted-declaration certificate, trace certificate, generator theorem, type-membership theorem, subject-reduction theorem, or theorem-proof row outside those displayed carrier rows.*

Proof. Use the introduction and the carrier definition to identify the sibling boundary. KernelAcceptanceWitness and KernelAcceptanceTrace read accepted declaration and trace data, GeneratorClosure reads the accepted generator boundary, TypeClassifierMembership reads emitted classifier data, and SubjectReduction reads preservation-facing eliminator data.

The accepted-declaration and generator consumers are already restricted by Definition 915.9 and Definition 915.10. Both pass through the finite carrier rows and receive no off-carrier certificate.

Subject-reduction consumers are restricted by Definition 915.11, while constructor, eliminator, refusal, continuation, and transport scope are restricted by Definition 915.5, Definition 915.6, Definition 915.7, Definition 915.19, and Definition 915.20.

Thus every named sibling read is a carrier-local readback over $D, S, E, P, R, H, C, Q, N$. The carrier lacks theorem-proof, generic trace theorem, generator theorem, type-membership theorem, and subject-reduction theorem rows, so no sibling can consume such a row from this packet. □

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceUp sibling consumer scope`

Theorem 915.22 (KernelInductiveAcceptance mature package)**.** *The carrier of Definition 915.1, the constructor-ledger exhaustion of Definition 915.5, the eliminator-ledger exhaustion of Definition 915.6, the refusal nonescape boundary of Definition 915.7, the nontrivial boundary of Definition 915.3, the field-faithful boundary of Definition 915.4, and the sibling consumer routes of Definition 915.21 form the mature paper package for* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$*. The package is mature exactly as a carrier-local inductive-acceptance readback: every accepted constructor, emitted eliminator, refused positivity candidate, TasteGate boundary, field-faithful read, and named sibling consumer remains inside* $D, S, E, P, R, H, C, Q, N$*.*

Proof. Start with the carrier and NameCert obligations. They display the declaration candidate, constructor-signature ledger, emitted-eliminator ledger, positivity-refusal row, recursion discipline, transport, replay, provenance, and local naming as one finite packet.

The constructor and eliminator ledgers are exhausted by their dedicated theorems. Thus accepted constructor reads return to S, and recursor, induction-principle, and computation reads pass through S before they reach E.

The refused side is separated by refusal nonescape, nontriviality, and field-faithfulness. A failed positivity candidate stays in P, an accepted read preserves the paired S, E surface, and a read that drops either ledger is outside this carrier.

Finally the sibling consumer scope theorem packages the downstream routes. KernelAcceptanceWitness, KernelAcceptanceTrace, GeneratorClosure, TypeClassifierMembership, and SubjectReduction consume only the displayed carrier rows, so the mature package exports no theorem-proof row, generic trace theorem, generator theorem, type-membership theorem, subject-reduction theorem, or hidden kernel-correctness witness. □

Falsifiable prediction: If the carrier is atomic, every accepted row exposes a constructor signature and emitted eliminator ledger; a packet lacking either row is rejected by the local classifier.

Independence witness: kernelacceptancewitness, kernelacceptancetrace, generatorclosure, typeclassifiermembership, subjectreduction: none is bijective because those siblings lack the paired constructor-signature plus emitted-eliminator ledger row, while this carrier lacks their theorem-proof, generic trace, generator-closure, or subject-reduction target rows.

Closure status for $\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelInductiveAcceptance}^{\uparrow}$ packet is generated from an inductive declaration candidate, constructor-signature ledger, emitted-eliminator ledger, positivity-refusal row, recursion-discipline row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 915.1, Definition 915.2, Definition 915.3, Definition 915.4, Definition 915.5, Definition 915.6, Definition 915.7, Definition 915.8, Definition 915.9, Definition 915.10, Definition 915.11, Definition 915.12, Definition 915.13, Definition 915.14, Definition 915.15, Definition 915.19, Definition 915.20, Definition 915.21, and Definition 915.22 bind the carrier, local NameCert obligation, TasteGate boundary, ledger exhaustion, refusal nonescape, accepted declaration route, generator route, subject-reduction route, environment scope, type-classifier scope, subject-reduction scope lock, scoped consumer package, displayed Cont row, componentwise hsame transport, sibling consumer scope, and mature package to one carrier-local readback surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelInductiveAcceptance}^{\uparrow}}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.KernelInductiveAcceptanceUp.taste gate`

Lean target: `BEDC.Derived.KernelInductiveAcceptanceUp.KernelInductiveAcceptanceTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: human

Not claimed: This chapter does not prove full kernel correctness, subject reduction, theorem-proof soundness, generic trace completeness, generic theorem proof rows, or any bridge theorem.

Upgrade path: Bridge closure requires a checked bridge for the same carrier-local readback package.

Chapter 916

A Concrete Naming Certificate for $\mathsf{ClosedNormalEndpointTransport}^{\uparrow}$

The $\mathsf{ClosedNormalEndpointTransport}^{\uparrow}$ packet realizes the typed normal endpoint boundary isolated by chapter 9376. It sits beside the endpoint ledger of chapter 837, the route certificate of chapter 858, and the subject-reduction route of chapter 787: those siblings expose the closed beta-strong endpoint, the route certificate, and the blocked transport surface, while the present packet records the finite NameCert transport row that a downstream consumer must display before reading the closed-normal contradiction route.

Definition 916.1 (Closed-normal endpoint transport carrier)**.** *A* $\mathsf{ClosedNormalEndpointTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, B, T, F, O, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source proof row naming the closed proof candidate before the endpoint handoff;*
2. *B is the beta-route row naming the finite reduction route available to the packet;*
3. *T is the typed normal endpoint row consumed by the closed-normal consistency main result;*
4. *F is the false-type preservation socket, displayed as a socket rather than as a discharged subject-reduction theorem;*
5. *O is the subject-reduction obstruction row that prevents the transport from being reread as full closed consistency;*
6. *L is the non-escape ledger recording that arbitrary closed proofs are not exported beyond the typed normal endpoint boundary;*
7. *H stores componentwise* hsame *transports for source proof, beta-route, typed endpoint, socket, obstruction, and non-escape rows;*
8. *C stores displayed* Cont *routes from the source proof and beta-route rows to the endpoint row and from the endpoint row to the closed-normal route certificate;*
9. *P records* Pkg *provenance over chapter 9376, chapter 837, chapter 858, chapter 787,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalEndpointTransport}^{\uparrow}}$ *row.*

The classifier compares two carriers only by the finite rows $S, B, T, F, O, L, H, C, P, N$. No coordinate contains full closed consistency, a closed subject-reduction theorem, unrestricted normalization, quotient conversion, host evaluation, or an ambient proof that every closed false proof reaches a typed normal endpoint.

Lean4 ✓ `BEDC.Derived.ClosedNormalEndpointTransportUp.ClosedNormalEndpointTransportTasteGate`
`single carrier alignment`

Theorem 916.2 (Closed-normal endpoint transport NameCert obligations)**.** *Every accepted* ClosedNormalEndpoint *packet binds exactly the finite obligation surface of Definition 916.1: carrier admission for $E = (S, B, T, F, O, L, H, C, P, N)$, source proof row exactness through S, beta-route exactness through B, typed normal endpoint exactness through T, false-type preservation socket visibility through F, obstruction visibility through O, non-escape through L, componentwise classifier stability through H, displayed continuation stability through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. Admission exposes only the finite tuple $E = (S, B, T, F, O, L, H, C, P, N)$, so every public read starts from the displayed source proof, beta-route, endpoint, socket, obstruction, and non-escape rows.

The endpoint transport is read in two stages. The rows S and B name the source proof and beta route, while T names the typed normal endpoint already at the boundary consumed by chapter 9376. The socket F is visible but not discharged into a closed subject-reduction theorem.

The rows O and L record the negative boundary. They keep the subject-reduction obstruction and non-escape ledger in the same packet as the positive endpoint row, so a consumer cannot erase the missing first arrow from arbitrary closed proof to typed normal endpoint.

The structural rows H, C, P, N transport, replay, package, and name only this finite surface. Since no coordinate supplies full closed consistency, unrestricted normalization, quotient conversion, host evaluation, or ambient consistency, the accepted packet exports only the displayed endpoint transport boundary. □

Theorem 916.3 (Closed-normal endpoint transport boundary)**.** *Every consumer of an accepted* ClosedNormalEndpointTransport$^{\uparrow}$ *packet that reaches a beta-strong false endpoint must factor through either the displayed transport row from S, B to T together with the socket F, or through the displayed obstruction row O and non-escape ledger L. A consumer that asks for full closed consistency, closed subject reduction, unrestricted normalization, quotient proof conversion, host evaluation, or an unlisted route from arbitrary closed proofs to typed normal endpoints is outside the carrier.*

Proof. Start with a consumer read of the packet and project the carrier. The only endpoint coordinate is T, and the only rows leading to it are the displayed source proof row S, beta-route row B, and socket row F.

If the read uses the positive endpoint route, then it is already a finite closed-normal endpoint read and can be handed to the route certificate of chapter 858. The rows H and C preserve that route componentwise and do not add a subject-reduction discharge.

If the read asks for the missing first arrow, it must encounter O and L. Those rows record that the transport surface is an obstruction-aware endpoint packet, not a proof of arbitrary closed consistency.

Thus every accepted read either stays on the displayed endpoint transport row or stops at the displayed obstruction boundary. The package and naming rows P, N make that boundary public without adding any unlisted consistency principle. □

Falsifiable prediction: A consumer that reaches a beta-strong false endpoint without the displayed transport row or obstruction row is not accepted by this carrier.

Independence witness: closednormalendpointledger, closed-normal-route-certificate, subjectreductionroute, subjectreductionobligation: none is bijective because endpoint transport carries both route-preservation and obstruction rows, while each sibling omits at least one of those coordinates.

The later Lean target in `lean4/BEDC/Derived/ClosedNormalEndpointTransportUp/TasteGate.lean` must provide explicit Nontrivial ClosedNormalEndpointTransport$^{\uparrow}$ and FieldFaithful ClosedNormalEndpointTransport$^{\uparrow}$ instances for this same finite carrier. Nontriviality separates the endpoint transport row from the obstruction row, while field faithfulness preserves the displayed coordinates $S, B, T, F, O, L, H, C, P, N$ for every TasteGate-facing consumer.

Closure status for NameCert$_{\text{ClosedNormalEndpointTransport}^{\uparrow}}$ **·**

Constructive story: A ClosedNormalEndpointTransport$^{\uparrow}$ packet is generated from source proof, beta-route, typed normal endpoint, false-type preservation socket, subject-reduction obstruction, non-escape ledger, componentwise transport, displayed continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ClosedNormalEndpointTransport}^{\uparrow}}$, seed)

Scope closed: Definition 916.1, Definition 916.2, and Definition 916.3 bind the finite NameCert endpoint transport boundary for closed-normal consistency.

Formal verification: FormalStatus(NameCert$_{\text{ClosedNormalEndpointTransport}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full closed consistency, closed subject reduction, unrestricted normalization, quotient conversion, host evaluation, ambient consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-route exactness, socket visibility, obstruction visibility, classifier stability, continuation stability, provenance, local naming, nontriviality, and field faithfulness for the same finite endpoint transport packet.

Chapter 917

A Concrete Naming Certificate for TailBudgetCoherence$^{\uparrow}$

The TailBudgetCoherence$^{\uparrow}$ packet is the composite Real and Completion surface joining the finite tail-budget route of chapter 906, the shared threshold route of chapter 560, the agreement-seal route of chapter 699, and the diagonal-completion route named in chapter 9238. It is not an atomic carrier: its atomicity witness is intentionally replaced by a composite-dependency explanation over existing sibling packets. The later Lean TasteGate work must still provide Nontrivial TailBudgetCoherence$^{\uparrow}$ and FieldFaithful TailBudgetCoherence$^{\uparrow}$ instances for the displayed composite carrier.

Definition 917.1 (Tail budget coherence carrier)**.** *A* TailBudgetCoherence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the* RegularCauchyTailMeet$^{\uparrow}$ *shared-threshold row used by the diagonal tail route;*
2. *O is the* RealObservationBudget$^{\uparrow}$ *observation-budget row that bounds the Real-facing finite reads;*
3. *S is the* RegularCauchySelectorBudget$^{\uparrow}$ *selector-budget row that supplies the diagonal member requests;*
4. *A is the* RealTailAgreementSeal$^{\uparrow}$ *agreement-seal row tying the tail route to the Real-facing agreement surface;*
5. *L is the* CauchyLimitSeal$^{\uparrow}$ *limit-seal row reached only after the threshold, observation budget, selector budget, and agreement seal have been displayed;*
6. *W is the finite* StreamName$^{\uparrow}$ *window ledger served by the observation and selector budgets;*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows;*
8. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger read before the regular-rational handoff and seal rows;*
9. *H records componentwise* hsame *transports for the threshold, observation budget, selector budget, agreement seal, limit seal, windows, readback, tolerance ledger, route, provenance, and local naming rows;*

10. *C records the displayed* Cont *routes from the tail threshold to the budget rows, from budgets to finite windows, from windows to dyadic tolerance and regular readback rows, and from the readback to the agreement and limit seals;*
11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TailBudgetCoherence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier contains no quotient stream equality, no selected host limit, no host real equality, and no ambient completeness principle.

Lean4 (definition) `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceUp`

Theorem 917.2 (Tail budget coherence NameCert obligations)**.** *For the carrier of Definition 917.1, the local* $\mathsf{NameCert}_{\mathsf{TailBudgetCoherence}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (M, O, S, A, L, W, R, D, H,$ TailMeet threshold exposure through M, observation-budget exposure through O, selector-budget exposure through S, agreement-seal exposure through A, limit-seal exposure through L, finite window service through W, regular readback through R, dyadic tolerance accounting through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceTasteGate single carrier alignment`

Proof. Project the displayed packet. Its coordinates are exactly $M, O, S, A, L, W, R, D, H, C, P, N$, so carrier admission and local naming cannot introduce a second tail threshold, budget, readback, agreement seal, or limit seal.

Read the route rows in dependency order. The threshold row M fixes the shared tail boundary; O and S bound the observation and selector requests; W, D, R display the finite windows, tolerance ledger, and regular-rational readback available after those budget rows.

The seal rows are then read through the same route. The agreement row A and limit row L are attached to the displayed readback surface, while H, C, P, N transport, replay, package, and name only that finite packet.

Thus the NameCert surface is composite rather than atomic. It packages the displayed sibling rows from the tail-meet, observation-budget, selector-budget, agreement-seal, limit-seal, stream-window, readback, and dyadic-ledger chapters, and it exports no quotient streams, host real equality, selected limits, or ambient completeness. □

Theorem 917.3 (Tail budget coherence real-completion handoff)**.** *Every Real and Completion consumer that cites an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet must read the completion-facing* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *surface through the displayed route*

$$M, O, S, W, D, R, A, L, H, C, P, N.$$

In particular, the route realizes the Real completeness handoff of chapter 9238 as one finite threshold, observation-budget, selector-budget, agreement-seal, and limit-seal surface over the sibling packets of chapter 560, chapter 699, and chapter 906.

Proof. Start from the accepted carrier and project the threshold, budget, window, readback, and seal rows. The TailMeet row M supplies the shared tail boundary, while O and S supply the two finite budget gates that service observation and selector requests.

Use the displayed window route next. The $\mathsf{StreamName}^{\uparrow}$ row W, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and $\mathsf{RegSeqRat}^{\uparrow}$ readback row R are the only finite readback coordinates available before a seal-facing consumer reaches A or L.

Now apply the sibling routes. The tail-budget packet of chapter 906 supplies the budget-row source, the TailMeet sibling supplies the threshold source, and the RealTailAgreementSeal sibling supplies the agreement-seal source. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ row is read only after those rows have been displayed.

The structural rows H, C, P, N carry the same composite packet to downstream Real and Completion consumers. Since the packet has no coordinate for quotient stream equality, selected host limits, host real equality, or ambient completeness, the handoff is exactly the finite BEDC-visible route stated above. □

Theorem 917.4 (Tail budget coherence threshold-budget exposure)**.** *For any accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

the TailMeet threshold row M, observation-budget row O, and selector-budget row S are exposed before any finite window row W, dyadic tolerance row D, regular readback row R, agreement-seal row A, or limit-seal row L is consumed.

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier threshold budget exposure`

Proof. Project the carrier of Definition 917.1. The first three budget-facing coordinates of the accepted packet are M, O, S, and the displayed carrier order places W, R, D, A, L only after those rows have been named.

Apply Definition 917.2. Its NameCert surface exposes TailMeet threshold, observation budget, and selector budget as separate rows before the finite window service, regular readback, dyadic tolerance accounting, agreement-seal exposure, and limit-seal exposure rows are read.

The structural rows H, C, P, N transport, replay, package, and name the same packet. They do not supply an alternate window, tolerance ledger, readback route, agreement seal, or limit seal outside the exposed threshold and budget rows. □

Theorem 917.5 (Tail budget coherence window-readback exactness)**.** *For any accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet $T = (M, O, S, A, L, W, R, D, H, C, P, N)$, the rows W, D, R form the unique finite window, dyadic tolerance, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route available under the displayed threshold and budget rows M, O, S.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier window readback exactness`

Proof. Use Definition 917.4 to fix M, O, S as the exposed threshold and budget rows before any readback-facing coordinate is consumed.

Now apply Definition 917.2. The accepted NameCert surface lists exactly one finite window service row W, one dyadic tolerance accounting row D, and one regular readback row R in the displayed packet.

Because the classifier compares accepted packets only through the displayed rows of Definition 917.1, there is no second finite window, tolerance ledger, or readback coordinate for a Real or Completion consumer to select. Thus W, D, R are the exact readback route under M, O, S. □

Theorem 917.6 (Tail budget coherence seal non-escape)**.** *Every Real or Completion consumer of an accepted* TailBudgetCoherence$^{\uparrow}$ *packet reaches the agreement-seal row A and limit-seal row L only after the rows M, O, S, W, D, R have been displayed, and receives no quotient stream equality, host real equality, selected host limit, countable-choice selector, or ambient completeness principle from the packet.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier seal nonescape`

Proof. Start with Definition 917.5. It identifies W, D, R as the exact finite readback route available under the already exposed threshold and budget rows M, O, S.

Apply Definition 917.3. Any Real or Completion consumer must read the handoff through the displayed route $M, O, S, W, D, R, A, L, H, C, P, N$, so the agreement seal A and limit seal L are reached only after the finite threshold, budget, window, tolerance, and readback rows.

Finally inspect the carrier definition. It contains no coordinate for quotient stream equality, host real equality, selected host limit, countable-choice selector, or ambient completeness, and the NameCert obligations package only the displayed finite rows. □

Theorem 917.7 (Tail budget coherence Real-completion budget route refinement)**.** *Every Real or Completion consumer of an accepted* TailBudgetCoherence$^{\uparrow}$ *packet factors through the ordered finite route*

$$M, O, S, W, D, R, A, L, H, C, P, N.$$

The threshold row M, observation-budget row O, and selector-budget row S are exposed before any finite window, readback, or seal row is consumed; W, D, R are the unique finite StreamName$^{\uparrow}$–DyadicRatCore$^{\uparrow}$–RegSeqRat$^{\uparrow}$ *readback route; and A, L are reachable only after that readback route has been displayed.*

The refinement exports no quotient stream equality, host real equality, selected host limit, countable-choice selector, or ambient completeness principle.

Proof. Apply Definition 917.4. It fixes M, O, S as the exposed threshold, observation-budget, and selector-budget surface before a consumer reaches any window, readback, tolerance, agreement-seal, or limit-seal row.

Next apply Definition 917.5. Under those budget rows, the only readback route is the finite W, D, R surface, read as the StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ tolerance ledger, and RegSeqRat$^{\uparrow}$ regular-rational readback.

Then use Definition 917.6. It places A, L after the displayed readback route and blocks every quotient stream equality, host real equality, selected host limit, countable-choice selector, and ambient completeness coordinate.

Finally apply Definition 917.3. The Real and Completion consumer therefore sees a single finite route $M, O, S, W, D, R, A, L, H, C, P, N$, with H, C, P, N transporting, replaying, packaging, and naming the same packet rather than opening a second analytic surface. □

Theorem 917.8 (Tail budget coherence obligation-closure package)**.** *For an accepted* TailBudgetCoherence$^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

the carrier of Definition 917.1, the NameCert obligations of Definition 917.2, the Real-completion handoff of Definition 917.3, the threshold-budget exposure of Definition 917.4, the window-readback exactness of Definition 917.5, and the seal non-escape boundary of Definition 917.6 assemble the fieldwise obligation *surface for* TailBudgetCoherence$^{\uparrow}$.

The rows remain separate carrier, NameCert, handoff, threshold-budget, finite-window, readback, tolerance-ledger, agreement-seal, and limit-seal obligations. The package exports no quotient stream, host real equality, selected limit, countable-choice selector, or ambient completeness row.

Proof. Project the carrier definition to fix the displayed packet $T = (M, O, S, A, L, W, R, D, H, C, P, N)$ and its ambient BEDC rows. The classifier has access only to the listed BHist, hsame, Cont, Pkg, and NameCert coordinates and to the named sibling rows.

Invoke Definition 917.2 to bind the carrier admission, local naming row, componentwise transports, continuation replay, package provenance, threshold row, budget rows, window row, tolerance ledger, readback row, agreement seal, and limit seal.

Chain Definition 917.3, Definition 917.4, and Definition 917.5. These theorems route every Real or Completion consumer through the finite sequence $M, O, S, W, D, R, A, L, H, C, P, N$, expose M, O, S before any readback-facing row, and identify W, D, R as the exact finite readback route under those exposed threshold and budget rows.

Finally apply Definition 917.6. The agreement seal and limit seal are reachable only through the displayed finite packet, and no quotient stream, host real equality, selected host limit, countable-choice selector, or ambient completeness coordinate is exported. The composite-dependency explanation therefore keeps the result as a finite Real-completion handoff package, not an atomic analytic object. □

Theorem 917.9 (Tail budget coherence primitive scope)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

the primitive scope is exactly the sibling rows $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with the kernel rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *displayed in the packet.*

Every Real-completion consumer route through T first passes through those rows and cannot use quotient stream equality, host real equality, selected limits, countable-choice selectors, or ambient completeness outside the displayed finite packet.

Proof. Project the carrier of Definition 917.1. This fixes the accepted packet as the displayed tuple $T = (M, O, S, A, L, W, R, D, H, C, P, N)$, with M, O, S, A, L, W, R, D supplied by the eight named sibling rows and H, C, P, N supplied by the local BHist, hsame, Cont, Pkg, and NameCert surface.

Use Definition 917.2 to bind carrier admission, componentwise transport, continuation replay, package provenance, and local naming to those displayed coordinates. Then apply Definition 917.3 to force every Real-completion consumer route through the same packet rather than through a host equality or ambient limit object.

Now use Definition 917.4 and Definition 917.5. They expose M, O, S before any readback-facing coordinate and identify W, D, R as the exact finite window, tolerance, and readback route available under that exposure.

Finally apply Definition 917.6. The agreement seal A and limit seal L are reachable only after the displayed finite route, and no coordinate remains for quotient stream equality, host real equality, selected limits, countable-choice selectors, or ambient completeness. Hence the primitive scope is exhausted by the finite packet named in the theorem. □

Theorem 917.10 (Tail budget coherence sibling-scope lock)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

each displayed row is read only through the named sibling packets and BEDC kernel rows: M through RegularCauchyTailMeet$^{\uparrow}$*, O through* RealObservationBudget$^{\uparrow}$*, S through* RegularCauchySelectorBudget$^{\uparrow}$*, A through* RealTailAgreementSeal$^{\uparrow}$*, L through* CauchyLimitSeal$^{\uparrow}$*, W through* StreamName$^{\uparrow}$*, R through* RegSeqRat$^{\uparrow}$*, D through* DyadicRatCore$^{\uparrow}$*, and H, C, P, N through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier sibling scope lock`

Proof. Project the carrier of Definition 917.1. The accepted tuple displays $M, O, S, A, L, W, R, D, H, C, P, N$ as separate coordinates, and the classifier compares accepted packets only through those coordinates.

Apply Definition 917.2 to bind each coordinate to its local obligation row. The threshold, observation-budget, selector-budget, agreement-seal, limit-seal, window, readback, and dyadic-tolerance rows are supplied by their named sibling packets, while H, C, P, N supply only componentwise transport, continuation replay, provenance, and local naming.

Finally use Definition 917.9. It identifies the primitive surface as the eight sibling rows together with BHist, hsame, Cont, Pkg, and NameCert. There is no remaining coordinate from which a consumer could read an unlisted sibling packet or a host-level analytic primitive. □

Theorem 917.11 (Tail budget coherence Real route scope)**.** *Every* Real$^{\uparrow}$ *or Completion consumer of an accepted* TailBudgetCoherence$^{\uparrow}$ *packet factors through*

$$M, O, S, W, D, R, A, L, H, C, P, N$$

before reading an agreement seal or a limit seal. In particular, no Real-facing or Completion-facing consumer reaches A or L without first passing through the TailMeet threshold, observation budget, selector budget, finite window, dyadic tolerance ledger, and regular readback rows.

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier real route scope`

Proof. Start with Definition 917.3. It states the handoff route for every Real and Completion consumer as the displayed sequence $M, O, S, W, D, R, A, L, H, C, P, N$.

Use Definition 917.4 to place M, O, S before any readback-facing row is consumed. Then use Definition 917.5 to identify W, D, R as the exact finite window, tolerance, and readback route under that exposed threshold and budget surface.

The seal rows are now forced by Definition 917.6. The agreement seal A and limit seal L are reachable only after M, O, S, W, D, R, and the structural rows H, C, P, N transport and name the same finite packet rather than opening a second Real or Completion path. □

Theorem 917.12 (Tail budget coherence no ambient completeness scope)**.** *The scoped* TailBudgetCoherence$^{\uparrow}$ *route exports no quotient stream equality, no host real equality, no selected limit, no countable-choice selector, and no ambient completeness principle. Its public output is only the finite coherence route through $M, O, S, W, D, R, A, L, H, C, P, N$ and the sibling packets named in Definition 917.10.*

Proof. Inspect the carrier of Definition 917.1. The accepted packet contains rows for the threshold, budgets, finite windows, dyadic tolerance ledger, regular readback, agreement seal, limit seal, componentwise transport, continuation replay, provenance, and local naming, but it contains no coordinate for quotient equality, host equality, selected limits, choice selectors, or completeness.

Apply Definition 917.8. The obligation surface assembled there is fieldwise over the same finite rows and explicitly keeps the seal route separate from quotient stream equality, host real equality, selected limits, countable-choice selectors, and ambient completeness.

Combine this with Definition 917.10 and Definition 917.11. The first locks every row to the named sibling and kernel packets, and the second forces Real and Completion consumers through the finite route before any seal read. Therefore the scoped route exports no ambient completeness object beyond the displayed coherence packet. □

Theorem 917.13 (Tail budget coherence public export boundary)**.** *The public export of an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet is exactly the finite route*

$$M, O, S, W, D, R, A, L, H, C, P, N$$

through the sibling rows named by Definition 917.10. Every $\mathsf{Real}^{\uparrow}$ *or Completion consumer must pass through threshold-budget exposure, window-readback exactness, Real route scope, and no-ambient-completeness scope before it reads the agreement seal* A *or limit seal* L*.*

Consequently the public export contains no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier public export boundary`

Proof. Start with the sibling-scope lock of Definition 917.10. It ties M, O, S, A, L, W, R, D to the named TailMeet, observation-budget, selector-budget, agreement-seal, limit-seal, StreamName, RegSeqRat, and DyadicRatCore sibling packets, and ties H, C, P, N to the BEDC kernel rows.

Apply threshold-budget exposure from Definition 917.4. It places M, O, S before any finite window, dyadic tolerance, regular readback, agreement-seal, or limit-seal coordinate can be consumed.

Next apply window-readback exactness from Definition 917.5 and Real route scope from Definition 917.11. These identify W, D, R as the exact readback route under M, O, S and force every Real or Completion consumer through $M, O, S, W, D, R, A, L, H, C, P, N$ before seal access.

Finally use no-ambient-completeness scope from Definition 917.12. Since the accepted route exports no quotient stream equality, host real equality, selected limit, countable-choice selector, or ambient completeness principle, the finite route written above is the complete public boundary. □

Theorem 917.14 (Tail budget coherence Real-seal window stability)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *or Completion consumer that reaches the agreement seal* A *or the limit seal* L *must preserve the same finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *fixed under the threshold-budget rows* M, O, S*.*

Componentwise hsame *transport may rename these rows, but it may not replace them by a second window, a host limit, a quotient stream equality, or an ambient completeness route.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier real seal window stability`

Proof. First expose the budget spine. By Definition 917.4, every readback-facing route through T presents M, O, S before the finite window, dyadic tolerance, regular readback, agreement seal, or limit seal is consumed.

Next fix the finite readback rows. Apply Definition 917.5; it identifies W, D, R as the exact $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route under the displayed threshold and budget rows, so a seal consumer cannot substitute a second readback surface.

Now place the seal consumer on the public route. The Real route scope of Definition 917.11 and the public export boundary of Definition 917.13 force any $\mathsf{Real}^{\uparrow}$ or Completion read of A or L through $M, O, S, W, D, R, A, L, H, C, P, N$.

Finally use seal non-escape from Definition 917.6. Since the accepted packet exports neither host limits, quotient stream equality, nor ambient completeness, componentwise hsame transport can only rename the displayed rows and cannot replace the finite window-readback triple. □

Theorem 917.15 (Tail budget coherence bridge interface)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

the composite sibling route from $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *bridges to the public Real-and-Completion consumer interface exactly through the finite surface*

$$M, O, S, W, D, R, A, L, H, C, P, N.$$

The forward direction is the public export boundary of Definition 917.13. The reverse projection is the sibling-scope lock of Definition 917.10 together with the primitive scope of Definition 917.9.

Thus the bridge carries only the displayed threshold, observation-budget, selector-budget, agreement-seal, limit-seal, window, readback, dyadic-tolerance, transport, continuation, provenance, and local naming rows. It carries no quotient stream equality, no host real equality, no selected limit, no countable-choice selector, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceUp StdBridge`

Proof. For the forward direction, apply Definition 917.13. It identifies the public Real-and-Completion export as the finite route $M, O, S, W, D, R, A, L, H, C, P, N$ through the sibling rows already locked by the accepted packet.

For the reverse projection, start with Definition 917.10. It reads each coordinate of T only through the named sibling packets and through the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Then apply Definition 917.9 to exhaust the primitive surface by those same sibling and kernel rows.

The consumer side is controlled by Definition 917.11. Every Real or Completion consumer factors through $M, O, S, W, D, R, A, L, H, C, P, N$ before reading an agreement seal or a limit seal.

Finally apply Definition 917.12. Since the route exports no quotient stream equality, host real equality, selected limit, countable-choice selector, or ambient completeness principle, the two projections describe a bridge only for the finite public surface and not for any stronger analytic equality or completeness object. □

Theorem 917.16 (Tail budget coherence finite-route determinacy)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

any two $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumers that read the same TailMeet threshold, observation budget, selector budget, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, agreement seal, limit seal, and structural rows determine the same public finite route*

$$M, O, S, W, D, R, A, L, H, C, P, N.$$

No second window, quotient stream equality, selected limit, host real equality, countable-choice selector, or ambient completeness row can distinguish the two routes.

Proof. Begin with threshold-budget exposure from Definition 917.4. It fixes the TailMeet threshold, observation-budget, and selector-budget rows before any consumer can read the finite window or either seal.

Next apply window-readback exactness from Definition 917.5. The shared window, dyadic ledger, and regular readback are exactly W, D, R, so two consumers with the same displayed readback data cannot separate into two finite-window routes.

The Real route scope and public export boundary, Definition 917.11 and Definition 917.13, place both consumers on $M, O, S, W, D, R, A, L, H, C, P, N$ before either Real or Completion endpoint sees the agreement or limit seal.

Finally use the sibling-scope lock, bridge interface, and no-ambient-completeness scope of Definition 917.10, Definition 917.15, and Definition 917.12. These exhaust the structural rows and exclude quotient stream equality, selected limits, host real equality, choice selectors, and ambient completeness rows, so the two reads are the same public finite route. □

Theorem 917.17 (Tail budget coherence synchronizer comparison)**.** *For an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet*

$$T = (M, O, S, A, L, W, R, D, H, C, P, N),$$

the $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet on the same consumer route is the window-to-seal projection of* T*: it reads the shared StreamName window* W*, RegSeqRat readback* R*, threshold-budget spine* M, O, S*, CauchyLimitSeal row* L*, transport, continuation, provenance, and local naming rows, and then exposes only the corresponding Real-window seal surface. The projection discards no hidden completion coordinate because* T *carries none.*

Lean4 ✓ `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceCarrier synchronizer comparison`

Proof. Begin with the TailBudgetCoherence bridge interface of Definition 917.15. It presents the public consumer route as $M, O, S, W, D, R, A, L, H, C, P, N$.

Compare this route with the RealWindowSynchronizer carrier and obligation package from Definition 979.1 and Definition 979.7. The synchronizer reads the same window W, readback R, threshold spine, limit seal L, transport, continuation, provenance, and naming rows when it exposes its Real-window seal.

The TailBudgetCoherence no-ambient-completeness scope ensures that the larger packet has no quotient stream, selected limit, host real equality, or ambient completeness coordinate to hide from the projection. Hence the synchronizer packet is exactly the window-to-seal projection of the public tail-budget packet. □

Theorem 917.18 (Tail budget coherence diagonal-selection dependency)**.** *Every Completion-facing* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *route consumes the finite diagonal-selection row of chapter 4080 before it reads the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows. The dependency is only a finite-route order: the packet does not choose a host diagonal sequence, assert quotient stream equality, or supply an ambient completeness principle.*

Proof. Start from threshold-budget exposure. The TailMeet threshold, observation budget, and selector budget rows M, O, S are visible before any finite window, tolerance, readback, or seal row is consumed.

The Cauchy diagonal-selection row is the selector-facing sibling that turns those displayed requests into the finite diagonal window service. Reading it first fixes the window requests that the $\mathsf{StreamName}^{\uparrow}$ row W may serve.

Then apply window-readback exactness and seal non-escape. The only downstream readback path is W, D, R followed by the agreement and limit seals A, L, and no row in the packet adds a host-selected diagonal, quotient equality, or ambient completeness object. □

Theorem 917.19 (Tail budget coherence shared-window refinement)**.** *In an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet* $T = (M, O, S, A, L, W, R, D, H, C, P, N)$*, the shared window and refinement rows* W, D, R *agree under the threshold-budget rows* M, O, S*. Any RealWindowSynchronizer consumer that reads* W *and* R *through this packet also reads the dyadic tolerance ledger* D *as the displayed refinement witness for the same shared threshold-budget spine.*

Proof. Apply threshold-budget exposure from Definition 917.4. It places the TailMeet threshold, observation budget, and selector budget rows M, O, S before the finite window, dyadic tolerance ledger, and regular readback are consumed.

Then use window-readback exactness from Definition 917.5. It identifies W, D, R as the exact StreamName, DyadicRatCore, and RegSeqRat route under those same threshold-budget rows.

A RealWindowSynchronizer consumer reading through the TailBudgetCoherence packet therefore cannot choose a second window or readback row. It receives W and R together with D as the refinement witness for the shared M, O, S spine. □

Theorem 917.20 (Tail budget coherence Real-window consumer route)**.** *Any* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *consumer that uses an accepted* $\mathsf{TailBudgetCoherence}^{\uparrow}$ *packet follows the public route*

$$M, O, S, W, D, R, A, L, H, C, P, N.$$

The route carries the TailMeet threshold, observation budget, selector budget, shared StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, agreement seal, CauchyLimitSeal row, transport, continuation, provenance, and local naming rows, and it exports no quotient stream or ambient completeness coordinate.

Proof. Use the synchronizer comparison of Definition 917.17 to place the RealWindowSynchronizer consumer as a projection of the TailBudgetCoherence public packet.

Apply shared-window refinement from Definition 917.19. The window, dyadic ledger, and readback rows are the displayed W, D, R rows under the same threshold-budget spine M, O, S.

Finally use the public export boundary and no-ambient-completeness scope of Definition 917.13 and Definition 917.12. The consumer route is exactly $M, O, S, W, D, R, A, L, H, C, P, N$, with no quotient stream or ambient completeness coordinate. □

Falsifiable prediction: If a completion-facing consumer reads the limit seal without a displayed TailMeet threshold, observation budget, selector budget, finite window ledger, dyadic tolerance ledger, regular readback, and agreement-seal row, that read is not accepted by $\mathsf{TailBudgetCoherence}^{\uparrow}$.

The composite-dependency explanation is the replacement for an atomic independence witness. The carrier is deliberately assembled from existing $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows; its content is the coherence of those rows at the Real-completion handoff, not a new primitive analytic object.

Closure status for $\mathsf{NameCert}_{\mathsf{TailBudgetCoherence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TailBudgetCoherence}^{\uparrow}$ packet is generated from TailMeet threshold, Real observation-budget, regular Cauchy selector-budget, RealTailAgreementSeal, CauchyLimitSeal, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance, componentwise transport, displayed continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailBudgetCoherence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 917.1, Definition 917.2, Definition 917.3, Definition 917.4, Definition 917.5, Definition 917.6, Definition 917.8, Definition 917.9, Definition 917.13, Definition 917.14, Definition 917.10, Definition 917.11, Definition 917.12, and Definition 917.15 give the mature treatment of the public export boundary, Real-seal window stability, bridge interface, sibling-scope lock, Real route scope, and no-ambient-completeness scope for the finite Real-completion tail-budget handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{TailBudgetCoherence}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.TailBudgetCoherenceUp.TailBudgetCoherenceUp\_StdBridge`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not prove Real completeness, quotient equality of regular streams, selected-limit existence, host real equality, or bridge checking for TailBudgetCoherence$^{\uparrow}$.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 918

A Concrete Naming Certificate for $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$

The $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ packet realizes the Real and Completion handoff requested by chapter 9238. It sits immediately downstream of the bounded monotone Cauchy witness surface of chapter 535: the witness still supplies the finite monotone source, tail window, Cauchy modulus, dyadic tolerance, and located trap rows, while the seal packages the public convergence handoff to $\mathsf{Real}^{\uparrow}$ without adding a selected limit or quotient equality.

Definition 918.1 (Bounded monotone convergence seal carrier)**.** *A* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *row that supplies the bounded source, tail witness, dyadic ledger, located trap, and real-seal precursor;*
2. *M is the public* $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail-window certificate through which the bounded monotone source is read;*
3. *K is the public* $\mathsf{CauchyCriterion}^{\uparrow}$ *packet through which tolerance requests, tail windows, and dyadic bounds are read;*
4. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row exported by the witness and criterion packets;*
5. *σ is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger used by the tail and seal reads;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and upper-bound ledger consumed before any convergence seal is visible;*
7. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row tying the witnessed Cauchy route to the completion-facing seal boundary;*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *convergence seal row reached only after W, M, K, F, σ, D, L have been displayed;*
9. *H records componentwise* hsame *transports for all displayed rows;*
10. *C records the displayed* Cont *routes by which consumers read the witness, monotone tail, criterion, regular-rational, stream-window, dyadic, limit-seal, and Real rows;*

11. *P is the* Pkg *provenance over* BoundedMonotoneCauchyWitness$^{\uparrow}$, MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{BoundedMonotoneConvergenceSeal}^{\uparrow}}$ *row.*

The carrier contains no selected limit representative, quotient stream equality, host real equality, trichotomy row, open-cover compactness row, detached tail oracle, or ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.BoundedMonotoneConvergenceSealUp.BoundedMonotoneConvergenceSealUp`

Theorem 918.2 (Bounded monotone convergence seal NameCert obligations)**.** *The local* NameCert$_{\text{BoundedMonotoneC}}$ *surface for Definition 918.1 consists exactly of carrier admission for $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$, bounded witness exposure through W, public monotone tail-window exposure through M, public Cauchy criterion exposure through K, regular-rational handoff through F, finite stream-window exposure through σ, dyadic tolerance and upper-bound accounting through D, Cauchy-limit seal compatibility through L, Real convergence seal handoff through R, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier. Its displayed rows are exactly the bounded witness, public monotone tail-window, public Cauchy criterion, regular-rational handoff, finite stream-window, dyadic ledger, limit-seal compatibility, Real seal, transport, route, provenance, and local naming rows.

Use Definition 535.31. It routes the bounded source through the public MonotoneCauchy$^{\uparrow}$ tail-window certificate and routes the witness rows through the public CauchyCriterion$^{\uparrow}$ packet before the Real-facing seal is read.

Finally attach the completion-facing rows $F, \sigma, D, L, R, H, C, P, N$. Since the carrier has no coordinate for a selected limit, quotient stream, host equality, trichotomy, open-cover compactness, detached tail oracle, or ambient completeness, the listed rows are the complete local naming surface. □

Theorem 918.3 (Bounded monotone convergence seal public handoff)**.** *Every* Real$^{\uparrow}$ *or completion-facing consumer of an accepted* BoundedMonotoneConvergenceSeal$^{\uparrow}$ *packet factors first through the bounded monotone Cauchy witness row W, then through the public* MonotoneCauchy$^{\uparrow}$ *and* CauchyCriterion$^{\uparrow}$ *rows M and K, and only then through the displayed route*

$$F, \sigma, D, L, R, H, C, P, N.$$

The consumer receives a finite Real convergence handoff assembled from BoundedMonotoneCauchyWitness$^{\uparrow}$, MonotoneCauchy$^{\uparrow}$, CauchyCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows, with the* CauchyLimitSeal$^{\uparrow}$ *row L acting only as the compatibility boundary for the sealed route. It receives no route that bypasses W or K to a selected limit, quotient stream, host real equality, ambient completeness witness, detached tail oracle, trichotomy row, or open-cover compactness row.*

Proof. Begin with the bounded monotone witness surface. By Definition 535.19, every witness read is already scoped by the displayed regular-rational, stream-window, Cauchy witness, dyadic, located-trap, Real seal, transport, route, provenance, and naming rows.

Next use the bridge to public tail and criterion packets. The MonotoneCauchy$^{\uparrow}$ row supplies the public tail-window certificate, while the CauchyCriterion$^{\uparrow}$ row supplies the explicit tolerance, tail, dyadic, regular-rational, and seal route.

The remaining rows $F, \sigma, D, L, R, H, C, P, N$ carry the public convergence handoff to the Real-facing boundary. The limit-seal row records compatibility with the sealed completion route, but it does not add a selected representative or a host equality coordinate.

Thus every consumer route is exhausted by the displayed packet. A route that omits W or K has no access to the bounded witness or Cauchy criterion rows, and a route that asks for selected limits, quotient streams, host equality, ambient completeness, detached tail oracles, trichotomy, or compactness asks for data absent from the carrier. □

Theorem 918.4 (Bounded monotone convergence seal witness consumption)**.** *Every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$$

consumes the witness row W through $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *before any public* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal row is available. A completion-facing consumer therefore receives the bounded source, tail witness, dyadic ledger, located trap, and Real-seal precursor from W before it reads $M, K, F, \sigma, D, L,$ or R.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneConvergenceSealUp.BoundedMonotoneConvergenceSealUp witness consumption`

Proof. Start with Definition 918.1. The first displayed coordinate of the accepted packet is W, and the carrier names it as the accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ row that supplies the bounded source, tail witness, dyadic ledger, located trap, and Real-seal precursor.

The sibling surface chapter 535 and the bridge theorem Definition 535.31 say that the bounded witness is the source from which the public $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ rows are read. The same public route is later continued through chapter 438, chapter 392, chapter 126, chapter 291, chapter 301, chapter 386, and chapter 38.

Finally apply Definition 918.3. Every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer factors through W before it reaches the public rows M, K and the route $F, \sigma, D, L, R, H, C, P, N$. Thus the witness row is exposed first and no direct completion read is available. □

Theorem 918.5 (Bounded monotone convergence seal criterion route)**.** *After the witness row W has been consumed, every tolerance-facing* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *read routes through the public rows M, K, F, σ, D, L before it reaches the Real seal row R. Such a read cannot bypass the public* $\mathsf{MonotoneCauchy}^{\uparrow}$ *row M or the public* $\mathsf{CauchyCriterion}^{\uparrow}$ *row K when it asks for a tail window, dyadic tolerance ledger, stream window, regular-rational handoff, or CauchyLimitSeal compatibility boundary.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneConvergenceSealUp.BoundedMonotoneConvergenceSealUp criterion route`

Proof. Project Definition 918.1. The carrier places M and K immediately after the bounded witness row W and before the regular-rational, stream, dyadic, CauchyLimitSeal, and Real rows.

The public tail and criterion rows are the sibling surfaces of chapter 438 and chapter 392. The bounded-witness source is supplied by chapter 535, and the later finite route is read through chapter 126, chapter 291, chapter 301, chapter 386, and chapter 38.

Now use Definition 918.2 together with Definition 918.3. The NameCert surface names public monotone tail-window exposure through M and public Cauchy criterion exposure through K, and the handoff theorem forces every completion-facing route through W, M, K before $F, \sigma, D, L, R, H, C, P, N$ can be consumed. □

Theorem 918.6 (Bounded monotone convergence seal Real handoff)**.** *For every accepted* BoundedMonotoneConver *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the Real-facing handoff reaches R only after the finite route

$$W, M, K, F, \sigma, D, L, R, H, C, P, N$$

has displayed the witness, criterion, stream, dyadic, limit-seal, transport, route, package, and NameCert rows. The row R is exactly the $\mathsf{Real}^{\uparrow}$ *handoff at the end of this displayed route and not a selected limit, quotient stream equality, host real equality, ambient completeness row, or metric-completeness row.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneConvergenceSealUp.BoundedMonotoneConvergenceSealUp real handoff`

Proof. Read the displayed order in Definition 918.1. The route reaches R only after W, M, K, F, σ, D, L have appeared, and the rows F, σ, D, L are the regular-rational, stream-window, dyadic-ledger, and CauchyLimitSeal compatibility coordinates.

The sibling surfaces certify the same ordering: chapter 535 provides the witness source, chapter 438 and chapter 392 provide the public tail and criterion gates, chapter 301, chapter 126, and chapter 291 provide the finite name and tolerance route, chapter 386 provides only the compatibility boundary, and chapter 38 provides the terminal seal surface.

Finally apply Definition 918.3. It identifies L as compatibility only and R as the Real convergence handoff after the displayed route. Since the carrier contains no coordinate for selected limits, quotient stream equality, host real equality, ambient completeness, or metric-completeness rows, the handoff is exact at the listed Real seal boundary. □

Theorem 918.7 (Bounded monotone convergence seal limit-seal compatibility)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the row L is only the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary reached after W, M, K, F, σ, D. It cannot itself expose a selected limit, quotient stream equality, host real equality, ambient completeness theorem, or a* $\mathsf{Real}^{\uparrow}$ *seal read.*

Proof. Project the carrier of Definition 918.1. The displayed route reaches L only after the bounded witness row, the public monotone and Cauchy rows, the regular-rational row, the stream-window row, and the dyadic ledger row.

Apply Definition 918.5. It identifies every tolerance-facing read of L as a read of the CauchyLimitSeal compatibility boundary after W, M, K, F, σ, D.

The sibling $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface of chapter 386 is therefore used only as the boundary surface, while the $\mathsf{Real}^{\uparrow}$ surface of chapter 38 is read only at the terminal row R. The carrier has no coordinate for selected limits, quotient stream equality, host real equality, ambient completeness, or an early Real seal read, so coordinate exhaustion gives the stated nonescape. □

Theorem 918.8 (Bounded monotone convergence seal tail-window consumption)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

every seal read consumes the bounded witness row W, public monotone tail-window row M, public Cauchy criterion row K, regular-rational handoff row F, stream-window row σ, and dyadic tolerance ledger row D before it may read the CauchyLimitSeal boundary L or the Real handoff row R.

Proof. Project the carrier of Definition 918.1. The displayed tuple places W, M, K, F, σ, D before L, R, so no coordinate order exposes L or R ahead of the bounded witness, public tail, public criterion, regular-rational, stream-window, and dyadic rows.

Apply Definition 918.3 and Definition 918.5. These route the public consumer through W, M, K and then through F, σ, D before the CauchyLimitSeal boundary can be read.

Finally use Definition 918.6. The Real handoff R is available only after the same finite prefix and the compatibility row L, so every seal read consumes the six displayed rows before either terminal seal coordinate is inspected. □

Theorem 918.9 (Bounded monotone convergence seal limit refusal exhaustion)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$*, the row L is only the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary. It exports no selected limit, quotient stream, host equality, trichotomy, open-cover compactness, or ambient completeness datum to any seal consumer.*

Proof. Use Definition 918.7. It identifies L as the CauchyLimitSeal compatibility boundary reached after the finite prefix W, M, K, F, σ, D.

The excluded data are not carrier rows. The packet has rows for the bounded witness, public monotone tail window, public Cauchy criterion, regular-rational handoff, stream window, dyadic ledger, compatibility boundary, Real handoff, transports, continuations, package provenance, and local NameCert provenance, but it has no coordinate for a selected limit, quotient stream, host equality, trichotomy, open-cover compactness, or ambient completeness principle.

Therefore a consumer that reads L receives compatibility only. Coordinate exhaustion blocks every listed limit or completeness datum from being exported through the limit-seal row. □

Theorem 918.10 (Bounded monotone convergence seal Real threshold route)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$*, the Real threshold row R is reached only through the finite route*

$$W, M, K, F, \sigma, D, L, R.$$

This route is consumed through the sibling surfaces $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{MonotoneCauchy}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the carrier order. The route to R passes through W, M, K, F, σ, D, L before the Real threshold coordinate is available.

Attach the sibling surfaces to the finite prefix. The bounded source is read through chapter 535; the public monotone and Cauchy gates are read through chapter 438 and chapter 392; the finite naming and tolerance rows are read through chapter 126, chapter 291, and chapter 301.

Now apply Definition 918.7 and Definition 918.6. They place the $\mathsf{CauchyLimitSeal}^{\uparrow}$ compatibility boundary at L and the $\mathsf{Real}^{\uparrow}$ handoff at R, so the displayed sibling route is the only route to the Real threshold row. □

Theorem 918.11 (Bounded monotone convergence seal Real boundary nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet reaches* R *only through the displayed route*

$$W \to (M, K) \to (F, \sigma, D, L) \to R.$$

It cannot recover an ambient completeness theorem, metric-completeness row, trichotomy row, open-cover compactness row, or detached tail oracle from that Real boundary.

Proof. First use Definition 918.3. It forces every Real-facing or completion-facing consumer to pass through the bounded witness row W before the public rows and finite route are available.

Next apply Definition 918.5 and Definition 918.6. These theorems place the public monotone and Cauchy rows M, K before F, σ, D, L and place the Real handoff row R only after the same route.

Finally use Definition 918.13. The excluded ambient completeness theorem, metric-completeness row, trichotomy row, open-cover compactness row, and detached tail oracle have no coordinate in $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$, so the displayed Real boundary cannot export them. □

Theorem 918.12 (Bounded monotone convergence seal obligation package)**.** *The obligation surface of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet is the ordered package*

$$W \to (M, K) \to (F, \sigma, D, L) \to R$$

together with the transport, route, provenance, and naming rows H, C, P, N*. It exposes the bounded witness first, then the public* $\mathsf{MonotoneCauchy}^{\uparrow}$ *and* $\mathsf{CauchyCriterion}^{\uparrow}$ *handoff, then the finite* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route, and only then the* $\mathsf{Real}^{\uparrow}$ *row. The package closes no Real completeness theorem, no selected-limit theorem, no quotient-stream equality, and no host-real equality.*

Proof. The carrier definition Definition 918.1 gives the finite packet $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$. The local NameCert theorem Definition 918.2 reads the same coordinates as carrier admission, witness exposure, public tail and criterion exposure, finite route exposure, Real handoff, transport, continuation, provenance, and local naming.

The sibling surfaces determine the displayed route. The witness part is chapter 535; the public tail and criterion gates are chapter 438 and chapter 392; the finite route is chapter 301, chapter 126, chapter 291, and chapter 386; the terminal seal surface is chapter 38.

Apply Definition 918.3. It forces every Real or completion consumer through W, then M, K, then F, σ, D, L, and only then R, H, C, P, N. Therefore the ordered package is the complete obligation surface for the current seal, while completeness, selected limits, quotient stream equality, and host real equality remain outside the carrier. □

Theorem 918.13 (Bounded monotone convergence seal terminal budget exhaustion)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

every terminal $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of* S *is exhausted by the ordered route*

$$W \to (M, K) \to (F, \sigma, D, L) \to R$$

together with the structural rows H, C, P, N*. After that route has been read, the packet cannot recover a selected limit, quotient stream equality, host real equality, trichotomy row, open-cover compactness row, detached tail oracle, ambient completeness theorem, or metric-completeness datum.*

Proof. Start with the carrier of Definition 918.1. Its only terminal-facing coordinates are the witness row W, the public monotone and Cauchy rows M, K, the finite stream, dyadic, regular-rational, and limit-seal route F, σ, D, L, the Real handoff row R, and the structural rows H, C, P, N.

Apply Definition 918.3, Definition 918.5, and Definition 918.6. These results force the terminal consumer through the displayed route before any Real-facing read is available.

Use Definition 918.12 to collect the same rows as the complete obligation surface. The carrier has no coordinate for the excluded completion or compactness data, so exhausting the displayed route also exhausts every completion-facing read available from the packet. □

Theorem 918.14 (Bounded monotone convergence seal scoped handoff package)**.** *Every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$$

binds the witness, public monotone tail, Cauchy criterion, regular-rational row, stream name, dyadic ledger, CauchyLimitSeal compatibility row, Real handoff, componentwise transport, displayed route, package provenance, and local name to the primitive BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The sibling surfaces* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{MonotoneCauchy}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are read only through the displayed route*

$$W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N).$$

The handoff exports no selected limit, quotient stream equality, host real equality, ambient completeness, trichotomy, or detached tail oracle.

Proof. Start with the carrier of Definition 918.1. It supplies exactly the tuple $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N$ and makes H, C, P, N the only transport, continuation, package, and naming rows available to the packet.

Apply the NameCert obligations of Definition 918.2. They bind each displayed coordinate to the same primitive BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface rather than to an external Real-completion principle.

Use the public handoff of Definition 918.3 together with the criterion route recorded in Definition 918.5 and the Real handoff recorded in Definition 918.6. These theorems force every sibling read through the bounded witness, public monotone and Cauchy gates, finite rational-stream route, CauchyLimitSeal compatibility row, and terminal Real row.

Finally apply Definition 918.12. The package theorem collects those rows into one scoped obligation surface, so selected limits, quotient stream equality, host real equality, ambient completeness, trichotomy, and detached tail oracles remain outside the accepted packet. □

Theorem 918.15 (Bounded monotone convergence seal provenance and local naming)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the provenance row P *and the local naming row* N *name exactly the ordered package*

$$W \to (M, K) \to (F, \sigma, D, L) \to R$$

together with the componentwise transport and displayed route rows H, C*. The rows* P *and* N *cannot add a selected limit, quotient stream equality, host real equality, trichotomy row, ambient completeness row, detached tail oracle, or open-cover compactness row.*

Proof. Project the carrier of Definition 918.1. Its final coordinates are the package provenance row P and local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}}$ row N, after the displayed witness, public tail, criterion, finite route, Real seal, transport, and route rows.

Apply Definition 918.2. The obligation list names package provenance through P and local naming through N only as rows over the same displayed carrier inventory. Use Definition 918.3 and Definition 918.12 to identify the ordered package as $W \to (M, K) \to (F, \sigma, D, L) \to R$ together with H, C.

It remains to exclude escaping data. The carrier has no coordinate for a selected limit, quotient stream equality, host real equality, trichotomy, ambient completeness, detached tail oracle, or open-cover compactness. Since P and N only name provenance and local naming for the listed carrier coordinates, they cannot introduce any unlisted row. □

Theorem 918.16 (Bounded monotone convergence seal checked obligation route)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the obligation route required for later checked Real-completion work is exactly

$$W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N).$$

The witness row W, monotone-tail row M, Cauchy-criterion row K, regular-rational row F, stream-name row σ, dyadic row D, CauchyLimitSeal row L, Real row R, transport row H, route row C, provenance row P, and local NameCert row N are supplied by the existing carrier, NameCert-obligation, public-handoff, obligation-package, scoped-handoff, and provenance/local-naming theorems of this chapter. The route excludes selected limits, quotient stream equality, host real equality, trichotomy, detached tail oracles, open-cover compactness, and ambient completeness.

Proof. Start with the carrier definition Definition 918.1. It supplies exactly the finite packet $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$ and has no coordinate for selected limits, quotient stream equality, host real equality, trichotomy, detached tail oracles, open-cover compactness, or ambient completeness.

Apply Definition 918.2 and Definition 918.3. These results expose the witness, public monotone-tail, Cauchy-criterion, finite rational-stream route, CauchyLimitSeal boundary, and Real handoff in the required order.

Now use Definition 918.12, Definition 918.14, and Definition 918.15. They package the same route as $W \to (M, K) \to (F, \sigma, D, L) \to R$ together with H, C, P, N, bind it to the primitive BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, and prevent the provenance or local name from adding any unlisted Real-completion content. □

Theorem 918.17 (Bounded monotone convergence seal witness scope)**.** *For every accepted* BoundedMonotoneConv *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the row W is the unique bounded-witness entry point into the L10 Real-completion chain. Every subsequent public row of the packet depends on the $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *surface exposed by W: the monotone-tail gate M, the Cauchy-criterion gate K, the finite route F, σ, D, L, the Real handoff R, and the bookkeeping rows H, C, P, N have no independent entry route that avoids W.*

Proof. Begin with Definition 918.1. It places W as the first displayed row and identifies it as the accepted $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ row carrying the bounded source, tail witness, dyadic ledger, located trap, and Real-seal precursor.

Apply Definition 918.4. It states that a completion-facing consumer consumes W before any of M, K, F, σ, D, L, or R is available.

Now use Definition 918.16. The checked route is exactly $W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N)$, so every later row is downstream of the witness surface and no second witness entry point is exported. □

Theorem 918.18 (Bounded monotone convergence seal witness-scope public factorization)**.** *Every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *completion-facing consumer factors through the ordered public route*

$$W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N),$$

where W is supplied by $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*, M by* $\mathsf{MonotoneCauchy}^{\uparrow}$*, K by* $\mathsf{CauchyCriterion}^{\uparrow}$*, F by* $\mathsf{RegSeqRat}^{\uparrow}$*, σ by* $\mathsf{StreamName}^{\uparrow}$*, D by* $\mathsf{DyadicRatCore}^{\uparrow}$*, L by* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and R by* $\mathsf{Real}^{\uparrow}$*. The route excludes selected limits, quotient stream equality, host real equality, trichotomy, detached tail oracle, open-cover compactness, and ambient completeness.*

Proof. First apply Definition 918.17. It makes W the unique bounded-witness entry point into the public L10 chain and places every later row downstream of that witness surface.

Next use Definition 918.3 and Definition 918.5. These theorems force the completion-facing consumer from W through the public monotone-tail row M, the Cauchy-criterion row K, and the finite route F, σ, D, L before the Real-facing handoff can be read.

Then apply Definition 918.6. It identifies R as the terminal $\mathsf{Real}^{\uparrow}$ handoff after the displayed finite route, not as a selected limit, quotient stream equality, host real equality, or ambient completeness row.

Finally use Definition 918.16 together with Definition 918.15. The checked route appends exactly H, C, P, N as transport, route replay, provenance, and local naming for the same packet, while provenance and naming cannot add trichotomy, detached tail oracles, open-cover compactness, selected limits, quotient stream equality, host real equality, or ambient completeness. □

Theorem 918.19 (Bounded monotone convergence seal criterion scope)**.** *Within an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

the rows M and K are mandatory public gates before the L10 route reaches F, σ, D, L, or R. The row M is the $\mathsf{MonotoneCauchy}^{\uparrow}$ *tail-window gate and the row K is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *tolerance gate; no StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, or Real handoff row is readable as a public convergence route until both gates have been consumed after W.*

Proof. Project the carrier in Definition 918.1. The rows M and K occur immediately after W and before the finite route rows F, σ, D, L and the Real row R.

Apply Definition 918.5. It prevents a tolerance-facing read from bypassing M or K when it asks for a tail window, dyadic tolerance ledger, stream window, regular-rational handoff, or CauchyLimitSeal boundary.

Finally use Definition 918.16. Since the displayed route is $W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N)$, the public criterion surface is a gate in the L10 chain rather than optional explanatory data. □

Theorem 918.20 (Bounded monotone convergence seal stream and Real scope)**.** *After the witness and criterion gates of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet have been consumed, the finite L10 route consumes the rows in the displayed order*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as the public surfaces named by the packet rows σ, D, F, L, R*. These surfaces are read from the carrier's finite block*

$$W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N),$$

so the StreamName, dyadic, regular-rational, limit-seal, and Real rows are consumed as one finite completion chain rather than as detached analytic endpoints.

Proof. Read the rows in Definition 918.1. The finite completion-facing part of the packet contains the regular-rational row F, the stream-window row σ, the dyadic row D, the CauchyLimitSeal boundary L, and the Real handoff R.

Use Definition 918.6. It places R after W, M, K, F, σ, D, L and identifies R as the terminal $\mathsf{Real}^{\uparrow}$ handoff rather than a selected limit or host equality. Thus the public consumption order of the finite L10 surfaces is read from the same finite block, not from an external analytic route.

The StreamName, dyadic, regular-rational, CauchyLimitSeal, and Real rows are therefore the finite L10 route named by Definition 918.16. The theorem's route order supplies the public consumption order for the chain, while the theorem statement records the sibling surfaces consumed by those rows. □

Theorem 918.21 (Bounded monotone convergence seal ledger exclusion scope)**.** *The ledger of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet excludes selected limits, quotient stream equality, host real equality, trichotomy, detached tail oracles, open-cover compactness, and ambient completeness. Its only completion-facing rows are the bounded witness* W*, public gates* M, K*, finite route rows* F, σ, D, L, R*, transport and route rows* H, C*, provenance row* P*, and local naming row* N*.*

Proof. Start with Definition 918.1. The definition explicitly lists the rows of $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$ and states that the carrier contains no selected limit representative, quotient stream equality, host real equality, trichotomy row, open-cover compactness row, detached tail oracle, or ambient completeness theorem.

Apply Definition 918.29. It identifies P and N as provenance and local naming rows for exactly the displayed packet and prevents them from importing any excluded analytic object.

Finally use Definition 918.16. The route already accounts for every witness, criterion, finite completion, Real, transport, provenance, and naming row, so the exclusions are part of the scoped obligation surface rather than a separate side condition. □

Theorem 918.22 (Bounded monotone convergence seal obligation scope package)**.** *The four scoped rows Definition 918.17, Definition 918.19, Definition 918.20, and Definition 918.21 form the obligation package for the L10 completion-chain reading of* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*. Together they bind the unique witness entry, mandatory public criterion gates, finite StreamName–DyadicRatCore–RegSeqRat–CauchyLimitSeal–Real route, and non-escape ledger for any* $\mathsf{obligation}$ *treatment of this packet.*

Proof. The witness-scope row identifies W as the only bounded-witness entry point. The criterion-scope row then forces M and K to be consumed as public $\mathsf{MonotoneCauchy}^{\uparrow}$ and $\mathsf{CauchyCriterion}^{\uparrow}$ gates before any completion-facing route can proceed.

The stream-and-Real scope row records the finite L10 consumption chain through σ, D, F, L, R, and the ledger-exclusion scope row removes selected limits, quotient equality, host equality, trichotomy, detached tail oracles, open-cover compactness, and ambient completeness from the packet.

Combining these rows with Definition 918.16 gives exactly the package $W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N)$. Hence the packet has a concrete obligation surface ready for a closure-grade reading without adding any Lean marker in this chapter. □

Theorem 918.23 (Bounded monotone convergence seal carrier admission)**.** *Every accepted* BoundedMonotoneConv *packet projects exactly the displayed carrier rows*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N).$$

Every $\mathsf{Real}^{\uparrow}$ *or completion-facing read of the packet must pass through the witness row W, public monotone-tail row M, public Cauchy-criterion row K, finite* $\mathsf{StreamName}^{\uparrow}$ *row σ, dyadic* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, regular-rational* $\mathsf{RegSeqRat}^{\uparrow}$ *row F,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row L, and terminal* $\mathsf{Real}^{\uparrow}$ *row R, with H, C, P, N recording transport, displayed route, provenance, and local naming.*

Proof. Project Definition 918.1. Its coordinates are exactly $W, M, K, F, \sigma, D, L, R, H, C, P, N$, and no other coordinate is available for a selected limit, quotient stream, host equality, ambient completeness witness, trichotomy, or detached tail oracle.

Apply Definition 918.4 and Definition 918.5. They force the witness read through W and then the public monotone-tail and Cauchy-criterion reads through M and K before the finite route rows can be consumed.

Use Definition 918.6 together with Definition 918.15. The Real-facing row R is reached only after F, σ, D, L, while H, C, P, N only record transport, route replay, provenance, and local naming for the same displayed packet. □

Theorem 918.24 (Bounded monotone convergence seal nontriviality)**.** *The* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *carrier admits two distinguishable finite packet shapes. In one shape the Real seal row is still blocked at the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary L; in the other shape the Real seal row R is displayed after the public route through W, M, K, F, σ, D, L. Hence the chapter does not present a one-element carrier.*

Proof. The carrier definition exposes both the compatibility boundary row L and the Real seal row R as separate coordinates. The sibling $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter supplies the sealed completion boundary, while the $\mathsf{Real}^{\uparrow}$ chapter supplies the public Real-facing seal surface.

For the boundary packet shape, read the displayed route only up to L. By Definition 918.5, tolerance-facing reads may reach the CauchyLimitSeal compatibility row after W, M, K, F, σ, D without thereby receiving a selected Real limit or host equality.

For the public seal packet shape, continue through Definition 918.3 and Definition 918.6. The row R is then displayed after the same public criterion route. Since one shape has no displayed R read and the other does, the two finite packet shapes are distinguishable by the consumer route. □

Theorem 918.25 (Bounded monotone convergence seal field faithfulness)**.** *For accepted* BoundedMonotoneConverg *packets*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$$

and

$$S' = (W', M', K', F', \sigma', D', L', R', H', C', P', N'),$$

equality of all displayed fields forces equality of the public seal read. Conversely, changing any one of W, K, F, σ, D, L, R *changes the corresponding consumer route through the witness, criterion, regular-rational, stream, dyadic, limit-seal, or Real-facing row.*

Proof. Assume the displayed fields of S and S' agree. By Definition 918.23, every public seal read is a read of the same ordered carrier rows, and by Definition 918.3 the Real-facing consumer factors through those rows in the displayed order.

The equality of $W, M, K, F, \sigma, D, L, R, H, C, P, N$ therefore transports the same witness, public tail, criterion, finite route, CauchyLimitSeal boundary, Real seal, route replay, provenance, and local name. The public seal read is equal because it has no coordinate outside that carrier inventory.

Now change one of W, K, F, σ, D, L, R. The witness-consumption theorem detects a change of W; the criterion-route theorem detects changes of K, F, σ, D, L; and the Real-handoff theorem detects a change of R. Each changed field is on the unique consumer route to the corresponding public read, so the corresponding route changes. □

Theorem 918.26 (Bounded monotone convergence seal route determinacy)**.** *Two accepted* BoundedMonotoneConv *seal-facing consumer routes with the same displayed rows*

$$W, M, K, F, \sigma, D, L, R, H, C, P, N$$

determine the same $\mathsf{Real}^{\uparrow}$ *handoff route. Conversely, changing any one of* W, K, F, σ, D, L, R *changes the corresponding witness, criterion, regular-rational, stream, dyadic, limit-seal, or Real-facing read.*

Proof. Start with carrier admission and the public handoff theorem. They make every seal-facing consumer route a read of the ordered carrier $W, M, K, F, \sigma, D, L, R, H, C, P, N$ and require the route to pass through the public completion-facing handoff before the Real row is exposed.

Next apply the criterion route and Real handoff. These rows force W to feed the witness surface, M, K to feed the public monotone-tail and Cauchy-criterion gates, F, σ, D to feed the regular-rational, stream, and dyadic route, L to feed the CauchyLimitSeal compatibility boundary, and R to feed the Real-facing read.

The obligation package and scoped handoff package then remove alternate route choices. With the same displayed rows, the route replay C, componentwise transport H, provenance P, and local name N can only certify the same finite passage through the same rows.

Finally use field faithfulness. A changed W, K, F, σ, D, L, or R is a changed field on the unique consumer route to its corresponding surface, so the affected witness, criterion, regular-rational, stream, dyadic, limit-seal, or Real-facing read changes as well. □

Theorem 918.27 (Bounded monotone convergence seal stability envelope)**.** *For accepted* BoundedMonotoneConver *packets* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$ *and* $S' = (W', M', K', F', \sigma', D', L', R', H', C', P', N')$*, componentwise* hsame *transport recorded by* H *preserves the displayed route*

$$W \to (M, K) \to (F, \sigma, D, L) \to R \to (H, C, P, N).$$

Thus a transported consumer still reads the bounded witness, public monotone-tail and Cauchy-criterion rows, finite rational-stream route, $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary,* $\mathsf{Real}^{\uparrow}$ *handoff, continuation replay, provenance, and local name in that order.*

Proof. Start with the carrier admission of Definition 918.23. It fixes the only coordinates available to a consumer as $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

Apply the local NameCert obligations of Definition 918.2. These obligations make H the componentwise hsame transport row for the displayed packet rather than a second route.

The public and scoped handoff rows of Definition 918.3 and Definition 918.14 then preserve the route through W, (M, K), (F, σ, D, L), and R. Finally, Definition 918.15, Definition 918.24, and Definition 918.25 keep the transport attached to the same finite provenance, local name, distinguishable boundary rows, and faithful public seal fields. □

Theorem 918.28 (Bounded monotone convergence seal exactness envelope)**.** *Within an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet, the row L is exactly the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary reached after W, M, K, F, σ, D, and the row R is exactly the* $\mathsf{Real}^{\uparrow}$ *handoff reached only after that finite route. No consumer may read L as a selected Real limit, or read R before the displayed bounded-witness, public-criterion, stream, dyadic, and regular-rational rows have been exposed.*

Proof. Use Definition 918.5. It places L after the witness, public monotone-tail, public Cauchy-criterion, regular-rational, stream, and dyadic rows, so L is only the compatibility boundary for that route.

Next apply Definition 918.6 and Definition 918.3. Together they put R after the same finite route and keep the Real-facing read from bypassing W, M, K, F, σ, D, L.

The carrier, obligation package, scoped handoff package, provenance-local-naming row, non-triviality row, and field-faithfulness row then rule out any alternate coordinate for a selected limit, an early Real handoff, or a second completion boundary. □

Theorem 918.29 (Bounded monotone convergence seal ledger exhaustion)**.** *For an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet, the provenance row P and local naming row N name exactly the displayed packet $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$. They cannot add selected limits, quotient stream equality, host real equality, trichotomy, compactness, or ambient completeness.*

Proof. Read the carrier and NameCert obligation rows from Definition 918.1 and Definition 918.2. The provenance and naming coordinates are P and N for the listed packet, with no extra coordinate for limit selection or ambient completeness.

Apply Definition 918.15. It identifies P as source provenance and N as the local NameCert row for the same route.

Finally combine Definition 918.23, Definition 918.3, Definition 918.14, Definition 918.24, and Definition 918.25. The route remains finite and field-faithful, so P, N exhaust the ledger rather than importing quotient equality, host equality, trichotomy, compactness, or a completeness principle. □

Theorem 918.30 (Bounded monotone convergence seal obligation closure package)**.** *For an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$, the paper obligation surface is exactly the carrier admission row, bounded-witness exposure row, public monotone-tail and Cauchy-criterion route, regular-rational stream route, dyadic tolerance ledger, stability envelope, exactness envelope, and provenance-local naming ledger. Equivalently, the obligation package is generated by carrier admission, witness exposure, criterion routing, stability, exactness, and ledger exhaustion over the displayed rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$.*

Proof. Start from carrier admission. By Definition 918.23, every accepted packet exposes only the displayed rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

The witness and criterion routes are supplied by Definition 918.4 and Definition 918.5. These place the bounded witness, monotone-tail row, Cauchy-criterion row, regular-rational handoff, stream window, dyadic ledger, limit-seal boundary, and Real-facing read in the stated order.

Next apply Definition 918.27 and Definition 918.28. They keep componentwise transport on the same route and make L and R exact terminal rows for the displayed convergence surface.

Finally use Definition 918.29. The provenance and local naming rows close the obligation package without importing selected limits, quotient stream equality, host real equality, trichotomy, compactness, or ambient completeness. □

Theorem 918.31 (Bounded monotone convergence seal obligation-upgrade scope)**.** *The paper obligation surface for* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *is ready only as a scope package over the displayed rows* $W, M, K, F, \sigma, D, L, R, H, C, P, N$*. Its dependency surface is the finite route through* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{MonotoneCauchy}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. This theorem prepares the* $\mathsf{obligation}$ *route, but it does not change the chapter closurestatus, add a Lean marker, or claim Real completeness, selected limits, quotient stream equality, host real equality, trichotomy, compactness, or ambient completeness.*

Proof. Start with Definition 918.30. It already generates the paper obligation package from carrier admission, witness exposure, criterion routing, stability, exactness, and ledger exhaustion over $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

The named dependencies are exactly the rows consumed by that route. $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ supplies W, $\mathsf{MonotoneCauchy}^{\uparrow}$ supplies M, $\mathsf{CauchyCriterion}^{\uparrow}$ supplies K, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ supply F, σ, D, while $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ supply L, R.

The primitive BEDC rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ supply carrier formation, transport, replay, provenance, and local naming. These are sufficient to state the paper scope package.

Finally inspect the excluded claims in Definition 918.32. The route still does not select a limit, identify quotient streams, assert host real equality, or import a completeness principle, so the closurestatus and Lean verification axis remain unchanged. □

Theorem 918.32 (Bounded monotone convergence seal public route totality)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet reads through exactly the route*

$$W, M, K, F, \sigma, D, L, R, H, C, P, N.$$

No such consumer can instead read a selected limit, quotient stream, host real equality, trichotomy, open-cover compactness row, or ambient completeness coordinate.

Proof. First apply Definition 918.30. The public surface is already generated by carrier admission, witness exposure, criterion routing, stability, exactness, and ledger exhaustion over $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

Use Definition 918.3 and Definition 918.14. They identify the $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ reads with the displayed finite route rather than with a separate analytic completion object.

Finally combine Definition 918.28 with Definition 918.29. Exactness forces L and R to remain the terminal seal and Real handoff rows, while ledger exhaustion excludes selected limits, quotient stream equality, host real equality, trichotomy, open-cover compactness, and ambient completeness. □

Theorem 918.33 (Bounded monotone convergence seal finite-limit interface obligations)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$*, the finite-limit interface obligation is exactly the displayed route through* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{MonotoneCauchy}^{\uparrow}$*,* $\mathsf{CauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*:*

$$W, M, K, \sigma, D, F, L, R, H, C, P, N.$$

The public consumer receives only this finite route and receives no selected limit, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient completeness principle.

Proof. Project the carrier $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$. The witness row is W, the public monotone and criterion rows are M and K, the stream, dyadic, and regular-rational rows are σ, D, F, and the completion-facing rows are L and R.

The sibling route is fixed by the named surfaces. $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ supplies W, $\mathsf{MonotoneCauchy}^{\uparrow}$ supplies M, $\mathsf{CauchyCriterion}^{\uparrow}$ supplies K, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ supply σ, D, F, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ supply L, R.

The remaining rows H, C, P, N only transport, replay, package, and name the same finite route. Since the carrier has no coordinate for selected limits, quotient stream equality, host real equality, trichotomy, compactness, or ambient completeness, the displayed interface exhausts the public obligation surface. □

Theorem 918.34 (Bounded monotone convergence seal L10 completion cofinality)**.** *Every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *read that faces the L10* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *phase reaches the terminal Real seal only after the displayed route has exposed the bounded witness row* W*, the public monotone and Cauchy criterion gates* M, K*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* σ*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* F*, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row* L*. The cofinal completion-facing route is therefore*

$$W, M, K, \sigma, D, F, L, R, H, C, P, N,$$

and it contains no selected limit, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First consume the bounded witness row. By Definition 918.4, every completion-facing read exposes W before the public rows or terminal seal can be inspected.

Next apply the criterion route, Definition 918.5. It forces the consumer through the public monotone and Cauchy criterion gates and then through the finite stream, dyadic, regular-rational, and CauchyLimitSeal rows before the Real handoff.

The finite-limit interface theorem Definition 918.33 identifies the same sibling surfaces as the L10-facing route: $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use the Real-boundary nonescape theorem Definition 918.11. Since the carrier has no coordinate for selected limits, quotient stream equality, host equality, trichotomy, compactness, or ambient completeness, every L10 completion read is cofinal in the displayed finite route and stops at the terminal Real seal. □

Theorem 918.35 (Bounded monotone convergence seal public limit refusal)**.** *Let* $S = (W, M, K, F, \sigma, D, L, R, H, C$ *be an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet. Any public request for a selected limit, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient completeness is refused at the finite route through* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{MonotoneCauchy}^{\uparrow}$*,*

$\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The accepted consumer may read* $W, M, K, F, \sigma, D, L, R, H, C, P, N$, *but no unlisted limit datum.*

Proof. Use Definition 918.32. Every public $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer reads exactly the displayed route $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

Now inspect the route by sibling. The witness, monotone, and criterion portion is supplied by $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, and $\mathsf{CauchyCriterion}^{\uparrow}$; the finite naming and tolerance portion is supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$; the boundary and terminal seal portion is supplied by $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

The refused data are not rows of S. Therefore a public request for selected limits, quotient stream equality, host equality, trichotomy, compactness, or ambient completeness stops at the same finite packet rather than becoming a consumer-visible row. □

Theorem 918.36 (Bounded monotone convergence seal criterion-to-Real route totality)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$, *every route from the public criterion row* K *to the Real handoff row* R *factors through the finite sibling chain* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as displayed by*

$$K, \sigma, D, F, L, R, H, C, P, N.$$

Together with the preceding rows W, M, *this is the whole route* $W, M, K, F, \sigma, D, L, R, H, C, P, N$; *it carries no selected limit, quotient stream equality, host real equality, trichotomy, open-cover compactness, or ambient completeness.*

Proof. Start at K. By Definition 918.5, a tolerance-facing read must pass from the public criterion row through the stream, dyadic, regular-rational, and CauchyLimitSeal rows before reaching the Real handoff.

Attach the sibling names to those coordinates. The criterion row is a $\mathsf{CauchyCriterion}^{\uparrow}$ row, σ is the $\mathsf{StreamName}^{\uparrow}$ window ledger, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, F is the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, L is the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary, and R is the $\mathsf{Real}^{\uparrow}$ handoff.

Finally prepend W, M and append H, C, P, N. The result is the same finite route named by Definition 918.33, so all consumer-visible data are exhausted before any selected limit or ambient completion principle could be read. □

Theorem 918.37 (Bounded monotone convergence seal route obligation closure)**.** *For an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$, *the displayed rows jointly close the paper-side route obligations for the bounded monotone convergence seal. The closed route is exactly the carrier, NameCert obligations, public handoff, criterion route, Real handoff, limit-seal compatibility, terminal budget exhaustion, and scoped handoff package over the rows* $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

Proof. Start with the carrier and NameCert obligations, Definition 918.1 and Definition 918.2. They expose precisely the rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$ and state the admission, routing, stability, exactness, provenance, and naming obligations.

Use public handoff, criterion routing, Real handoff, and limit-seal compatibility: Definition 918.3, Definition 918.5, Definition 918.6, and Definition 918.7. These identify the public consumer path from bounded witness rows to the terminal Real handoff without inserting a selected limit or ambient completeness row.

Finally apply terminal budget exhaustion and the scoped handoff package, Definition 918.13 and Definition 918.14. The same rows exhaust the route budget and package the resulting finite handoff, so the paper-side route obligations are closed at the displayed boundary. □

Theorem 918.38 (Bounded monotone convergence seal criterion dyadic ledger lock)**.** *For an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$*, every accepted route from the public criterion row* K *to the regular-rational handoff row* F *must read the dyadic tolerance ledger* D *through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling row before either* F *or* R *is public. Thus the criterion-to-handoff route has the ordered public core*

$$K, \sigma, D, F, L, R.$$

Proof. Project the carrier and NameCert obligations from Definition 918.1 and Definition 918.2. They expose K, σ, D, F, L, R as named rows of the same accepted packet and leave no alternate tolerance coordinate.

Apply the criterion route and Real handoff results, Definition 918.5 and Definition 918.6. The route from the criterion row to the Real-facing row passes through the stream-window row, the dyadic ledger, the regular-rational handoff, and the limit-seal row.

Since D is the only dyadic tolerance ledger in the carrier, the public handoff cannot expose F or R before that row has been read. The displayed order is therefore the complete dyadic ledger lock for the accepted route. □

Theorem 918.39 (Bounded monotone convergence seal stream-window regular handoff lock)**.** *For an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L, R, H, C, P, N)$*, the stream-window row* σ *and the regular-rational row* F *form one ordered handoff. No selected limit, quotient stream equality, or host real equality may bypass* σ *and* F *on the path from the public bounded monotone witness rows to the terminal Real handoff.*

Proof. Use the public handoff theorem, Definition 918.3, to locate the consumer path through the displayed carrier rows. The stream-window row σ is read before the regular-rational row F is exposed to downstream consumers.

The tail-window consumption and criterion route are the same finite path: Definition 918.8 and Definition 918.5 identify the stream and regular-rational rows as adjacent obligations in the handoff from criterion data to CauchyLimitSeal data.

Apply limit-seal compatibility, Definition 918.7. Because compatibility is stated only after σ and F have been read, selected limits, quotient stream equality, and host real equality have no bypass coordinate in the public route. □

Theorem 918.40 (Bounded monotone convergence seal threshold route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet* $S = (W, M, K, F, \sigma, D, L,$ *can project only the displayed witness, monotone, criterion, stream, dyadic,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, transport, route, provenance, and NameCert rows. Consequently the consumer receives no ambient completeness, trichotomy, open-cover compactness, or detached tail oracle.*

Proof. Read the packet coordinatewise. The rows $W, M, K, \sigma, D, F, L, R, H, C, P, N$ are exactly the witness, monotone, criterion, stream, dyadic, regular-rational, CauchyLimitSeal, Real, transport, route, provenance, and NameCert rows.

Use terminal budget exhaustion and route obligation closure, Definition 918.13 and Definition 918.37. These results say that the displayed rows consume the public route budget at the terminal threshold.

No remaining coordinate of S can name ambient completeness, trichotomy, open-cover compactness, or a detached tail oracle. Hence a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer can project only the rows already displayed by the accepted seal. □

Theorem 918.41 (Bounded monotone convergence seal TasteGate obligation boundary)**.** *Promotion beyond seed status for* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *is read only through the displayed carrier rows together with this chapter's stated TasteGate nontriviality and field-faithfulness commitments. This changes the theory-closure ledger for the paper-side obligation surface, and it does not change the Lean verification-status row.*

Proof. Project the route obligation closure above. The paper-side boundary already binds the carrier rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$ to the NameCert and public handoff obligations.

Now use the chapter's TasteGate statement for the same displayed carrier. Nontriviality and field faithfulness are read as obligations over the existing packet; they do not add a Lean theorem marker or a new formal-status target.

Thus the closure ledger may record obligation-level theory closure while leaving the Lean verification grade unchanged. The two axes remain independent: the paper surface states the route and TasteGate obligations, while future Lean work must still encode and verify them. □

Theorem 918.42 (Bounded monotone convergence seal witness-grid factorization)**.** *For every accepted* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ *packet*

$$S = (W, M, K, F, \sigma, D, L, R, H, C, P, N),$$

whose witness row W *is read through the* $\mathsf{RealCauchyWitnessGrid}^{\uparrow}$ *surface of chapter 905, every Real-facing consumer route factors through the convergence-seal rows*

$$W, M, K, F, \sigma, D, L, R, H, C, P, N$$

and through the sibling grid rows

$$B, L, K, T, S, R, E, H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route uses the witness ledger of chapter 902, the completion-seal classifier of chapter 900, and the tail-witness envelope of chapter 898; it exports no Real completeness theorem, selected external limit, quotient stream equality, or host real equality.*

Proof. Project the carrier of Definition 918.1. The rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$ are the complete convergence-seal packet, so a Real-facing consumer starts from the displayed finite carrier and not from an ambient completeness object.

Use the public handoff and criterion route, Definition 918.3 and Definition 918.5. These place the bounded witness, monotone row, criterion row, stream-window row, dyadic ledger, regular-rational handoff, and limit-seal row on one ordered route before the Real handoff is available.

Now read the witness row through chapter 905. That chapter factors the Real-facing diagonal grid through $B, L, K, T, S, R, E, H, C, P, N$ and ties those rows to the witness ledger, completion-seal classifier, and tail-witness envelope named in chapter 902, chapter 900, and chapter 898.

Finally apply the Real handoff and TasteGate obligation boundary from Definition 918.6 and Definition 918.41. The Real seal is therefore consumed only after both finite surfaces have been displayed, and the non-escape clauses leave no coordinate for Real completeness, a selected external limit, quotient stream equality, or host real equality. □

Falsifiable prediction: Every accepted consumer of the convergence seal factors through BoundedMonotoneCauchyWitnessUp, MonotoneCauchyUp, CauchyCriterionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and RealUp rows before it reads the Real convergence handoff.

Independence witness: boundedmonotonecauchywitness, monotonecauchy, cauchycriterion, cauchylimitseal, real: the bounded witness lacks the public convergence seal row, monotonecauchy lacks the bounded witness source packet, cauchycriterion lacks the monotone source ledger, cauchylimitseal lacks the bounded monotone witness and public criterion bridge, and real lacks the finite witness-to-seal route; therefore no bijection preserves all carrier rows $W, M, K, F, \sigma, D, L, R, H, C, P, N$.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/BoundedMonotoneConvergenceSealUp`. It must provide `Nontrivial BoundedMonotoneConvergenceSealUp` and `FieldFaithful BoundedMonotoneConvergenceSealUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ packet is generated from a bounded monotone Cauchy witness, public monotone Cauchy tail-window certificate, public Cauchy criterion packet, regular-rational handoff, stream-window ledger, dyadic tolerance ledger, CauchyLimitSeal compatibility row, Real convergence seal, componentwise transports, displayed continuations, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 918.1, Definition 918.3, Definition 918.4, Definition 918.5, Definition 918.6, Definition 918.7, Definition 918.12, Definition 918.13, and Definition 918.41, together with the checked TasteGate row, bind the public handoff and Real-facing seal rows of the finite convergence-seal export surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.BoundedMonotoneConvergenceSealUp.taste gate`

Lean target: `BEDC.Derived.BoundedMonotoneConvergenceSealUp.BoundedMonotoneConvergenceSealUp\_witness\_consumption`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Real completeness, selected-limit existence, quotient stream equality, host real equality, Lean verification, or bridge checking for $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$.

Upgrade path: No further paper-side closure upgrade remains; Lean verification and bridge checking remain separate formal-axis work.

Chapter 919

A Concrete Naming Certificate for $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$

The $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ packet concretizes the bounded observation surface used by the Real completeness route of chapter 9238. It consumes the selector discipline of chapter 782, the finite window naming surface of chapter 126, the regular-rational readback of chapter 301, and the Real-facing observation budget of chapter 611. Its role is to classify prefix-depth exhaustion before a $\mathsf{Real}^{\uparrow}$ seal read, so a diagonal-limit consumer can see exactly which finite rows have been spent.

Falsifiable prediction: Every accepted finite-prefix limit-budget packet has no route to host real equality, quotient streams, selected limits, or countable choice outside the displayed prefix-window rows.

Independence witness: finiteobservationbudgetselector, regularcauchyfiniteobservationroute, budgetedrealsealroute, cauchysealbudgetsynchronizer: this carrier fixes prefix-depth exhaustion before the seal read, while those siblings select budgets, route observations, synchronize seals, or package seal routes without this prefix-budget classifier.

Definition 919.1 (Finite-prefix limit-budget carrier)**.** *A* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite diagonal request row naming the precision, prefix depth, and consumer site;*
2. *B is the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *row that admits the displayed request as a finite observation budget;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *prefix-window ledger selected under B;*
4. *D is the prefix-depth exhaustion row recording that every admitted offset lies inside the displayed finite window and that no later offset is served by this packet;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the spent prefix windows;*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal row reached only after Q, B, W, D, R have been displayed;*
7. *E is the local exclusion ledger for host real equality, quotient streams, selected limits, countable choice, and unbounded schedules;*
8. *H records componentwise* hsame *transports for all displayed rows;*

9. *C records* Cont *routes from request to budget, from budget to prefix windows, from prefix exhaustion to regular readback, and from readback to the seal;*
10. *P records* Pkg *provenance over* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}$ *row.*

The carrier is a finite budget classifier. It exports no host real equality, quotient-stream carrier, selected completed limit, countable choice principle, hidden tail schedule, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitBudgetUp.FinitePrefixLimitBudgetTasteGate single carrier alignment`

Theorem 919.2 (Finite-prefix limit-budget NameCert obligations)**.** *For the carrier of Definition 919.1, the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for $F = (Q, B, W, D, R, S, E, H, C, P, N)$ as a* BHist *packet;*
2. *prefix-budget selection, requiring Q to be admitted through the finite selector row B before any* $\mathsf{StreamName}^{\uparrow}$ *window is served;*
3. *prefix-window exhaustion, requiring D to account for every offset admitted by W and to refuse every outside-prefix read;*
4. *regular and seal handoff, requiring R and S to be reached only through the spent prefix-window route;*
5. *non-escape, requiring E, H, C, P, N to preserve the same finite route and to block host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, and ambient completeness.*

Proof. Project the carrier. The displayed coordinates $Q, B, W, D, R, S, E, H, C, P, N$ are the only rows available to the local naming certificate.

Read the selection side first. The selector row B admits the finite request Q, and the window ledger W lists exactly the prefix windows served by that admitted budget.

Next read the exhaustion row D. It ties every served offset to the finite prefix window and records refusal for outside-prefix reads before the regular readback row R or the seal row S can be consumed.

Finally use E, H, C, P, N. The exclusion ledger blocks the unlisted analytic objects, while transport, continuation, provenance, and local naming replay only the finite route already displayed by the packet. □

Theorem 919.3 (Finite-prefix limit-budget seal ordering)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet reaches its seal row only through the ordered finite route*

$$Q \to B \to W \to D \to R \to S.$$

The displayed hsame, Cont, Pkg, *and* NameCert *rows certify no route that reads the seal before the prefix-depth exhaustion row has been consumed.*

Proof. Start with the five NameCert rows of Definition 919.2. The prefix-budget and prefix-window rows place Q, B, W before the exhaustion row D.

The regular and seal handoff row then places R and S after D. Thus a seal-facing consumer must read the spent prefix route before it can reach the completion-facing seal.

The transport and continuation rows preserve that order under componentwise hsame and Cont replay. Since the exclusion ledger contains no host equality, quotient stream, selected limit, countable choice, hidden tail schedule, or ambient completeness coordinate, there is no alternate seal route inside the packet. □

Theorem 919.4 (Finite-prefix limit-budget Real route handoff)**.** *The Real completeness route of chapter 9238 may consume an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet as the finite classifier between selected prefix observations and a* $\mathsf{Real}^{\uparrow}$ *seal read. The handoff is compatible with chapter 782, chapter 126, chapter 301, and chapter 611 because it preserves their budget, window, regular readback, and seal rows inside one displayed prefix-depth surface.*

Proof. Read the sibling selector first. The $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ row supplies the finite request budget consumed by B, and no observation is served before that row is displayed.

Read the window and regular-rational siblings next. The $\mathsf{StreamName}^{\uparrow}$ windows in W and the $\mathsf{RegSeqRat}^{\uparrow}$ readback in R are tied together by the prefix-depth exhaustion row D, so the route records which prefix rows have been spent before the seal-facing handoff.

Finally compare with the $\mathsf{RealObservationBudget}^{\uparrow}$ surface. That chapter supplies the completion-facing observation discipline; the present packet narrows it to the finite prefix-budget classifier needed before the Real completeness seal read. The non-escape row keeps quotient streams, host real equality, selected limits, countable choice, hidden tail schedules, and ambient completeness outside the handoff. □

Theorem 919.5 (Finite-prefix limit-budget prefix-exhaustion coverage)**.** *For every accepted* FinitePrefixLimitBudge *carrier*

$$F = (Q, B, W, D, R, S, E, H, C, P, N),$$

each admitted offset served by the finite $\mathsf{StreamName}^{\uparrow}$ *window row* W *is accounted for by the depth-exhaustion row* D*. Every outside-prefix read is refused before the regular readback row* R *or the seal row* S *is available to a consumer.*

Proof. Project the carrier rows Q, B, W, D, R, S from Definition 919.1. The finite request row Q must pass through the budget selector row B before the prefix-window ledger W serves an offset.

Apply Definition 919.2. Its prefix-window exhaustion clause identifies D as the row that accounts for every offset admitted by W and records refusal for outside-prefix reads.

Apply Definition 919.3. Since every seal-facing route has order $Q \to B \to W \to D \to R \to S$, neither R nor S is available until the exhaustion row has already accepted the displayed prefix offsets and refused the reads outside the finite window. □

Theorem 919.6 (Finite-prefix limit-budget hsame transport)**.** *Componentwise* hsame *transports of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier preserve the ordered route*

$$Q \to B \to W \to D \to R \to S.$$

They cannot create a new prefix window, a new prefix-depth exhaustion row, a new regular readback row, or a new seal row outside the displayed carrier.

Proof. Use the transport coordinate H of the carrier together with the continuation, provenance, and naming rows C, P, N. These rows are part of the finite packet, so transport acts componentwise on the displayed coordinates rather than adding a fresh carrier coordinate.

The ordered route is already fixed by Definition 919.3. Transport through H can replay the same Q, B, W, D, R, S route, and C records only the displayed continuations from request to budget, from budget to windows, from exhaustion to readback, and from readback to seal.

Finally, P and N keep the transported packet inside the same Pkg provenance and local $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}$ surface. No coordinate in H, C, P, N names an additional prefix window or an additional seal row, so componentwise hsame cannot create one. □

Theorem 919.7 (Finite-prefix limit-budget nonescape criterion)**.** *Any consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet that claims host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, or ambient completeness is outside the carrier unless the claim factors through the displayed exclusion row* E*.*

Proof. Read the carrier definition first. The final sentence of Definition 919.1 states that the packet exports none of the listed analytic objects; it exports only the finite budget classifier rows.

Apply the non-escape clause of Definition 919.2. That clause assigns the exclusion work to E, H, C, P, N and requires those rows to preserve the same finite route while blocking host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, and ambient completeness.

Thus a consumer that asserts one of those objects must leave the displayed carrier surface unless it presents the assertion through E. The remaining rows only replay selection, prefix-window exhaustion, readback, seal handoff, transport, continuation, provenance, and local naming for the finite route. □

Theorem 919.8 (Finite-prefix limit-budget carrier-classifier scope)**.** *Every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet has its carrier and classifier scoped by the displayed finite prefix request* Q*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing seal row* S*, componentwise* hsame *transport* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}$ *row* N*. No classifier read is accepted outside that finite prefix-budget surface.*

Proof. Project the carrier of Definition 919.1. The request, selector, window, exhaustion, readback, seal, exclusion, transport, route, provenance, and naming rows are already displayed as $Q, B, W, D, R, S, E, H, C, P, N$.

Use the obligation and seal-ordering theorems. Definition 919.2 makes selection, exhaustion, regular readback, and seal handoff part of the local certificate, while Definition 919.3 orders every consumer route through $Q \to B \to W \to D \to R \to S$.

Finally attach the transport and nonescape rows, Definition 919.6 and Definition 919.7. They keep classifier replay inside the same H, C, P, N support and exclude any outside prefix-budget coordinate. □

Theorem 919.9 (Finite-prefix limit-budget record factorization)**.** *All record reads of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet factor through the finite rows*

$$Q, B, W, D, R, S, E, H, C, P, N.$$

In particular, a record read cannot use quotient streams, host real equality, selected limits, countable choice, hidden tail schedules, or ambient completeness to replace the displayed finite prefix route.

Proof. Begin with prefix-exhaustion coverage, Definition 919.5. It forces every served offset through the finite $\mathsf{StreamName}^{\uparrow}$ window row W and the exhaustion row D before R or S is available.

Apply carrier-classifier scope, Definition 919.8. The carrier and classifier are bounded by the same request, window, readback, seal, transport, route, provenance, and local naming rows.

The exclusion follows from Definition 919.7. Since quotient streams, host real equality, selected limits, countable choice, hidden tail schedules, and ambient completeness occur only as refused claims in E, they cannot serve as alternate record factors. □

Theorem 919.10 (Finite-prefix limit-budget consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet receives only the same finite prefix-budget surface*

$$Q, B, W, D, R, S, E, H, C, P, N.$$

The consumer may read the finite request, selected prefix windows, exhaustion row, regular readback, seal handoff, exclusion ledger, transport, route, provenance, and local naming rows, but it receives no completed limit, quotient stream, host real equality, selected tail, hidden schedule, or ambient completeness witness.

Proof. Read the Real-route handoff first. Definition 919.4 places the Real completeness consumer behind the finite selector, StreamName, RegSeqRat, and RealObservationBudget rows already carried by the packet.

Next apply record factorization. Definition 919.9 says every record read used by such a consumer factors through $Q, B, W, D, R, S, E, H, C, P, N$ and through no quotient or selected-limit route.

Finally use the closurestatus upgrade-path surface as the boundary vocabulary: the regular-readback consumer and completion-facing seal interface are the consumers being scoped here. Because the same finite packet supplies both interfaces, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers receive exactly the displayed prefix-budget rows and nothing beyond them. □

Theorem 919.11 (Finite-prefix limit-budget diagonal selector consumption)**.** *Every diagonal-selector or Real-completion consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

must consume the finite route

$$Q \to B \to W \to D$$

before reading the regular readback row R or the completion-facing seal row S.

Proof. First use seal ordering, Definition 919.3. It places every completion-facing read behind the route $Q \to B \to W \to D \to R \to S$, so neither R nor S is available before the exhaustion row.

Next use prefix-exhaustion coverage, Definition 919.5. The diagonal selector spends only the finite windows admitted by W, and D records that every admitted offset has been accounted for while outside-prefix reads are refused.

Finally apply the consumer boundary, Definition 919.10. A Real-completion consumer receives only $Q, B, W, D, R, S, E, H, C, P, N$, so it cannot read the diagonal selector as a completed tail or selected limit before the finite request, budget, window, and exhaustion rows have been consumed. □

Theorem 919.12 (Finite-prefix limit-budget Real seal non-escape extension)**.** *No* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet can replace the displayed finite prefix route by host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, or ambient completeness. The only seal-facing route available to such a consumer is the displayed finite route through $Q, B, W, D, R, S, E, H, C, P, N$.*

Proof. Begin with the nonescape criterion, Definition 919.7. Host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, and ambient completeness appear only as refused claims in the exclusion row E.

Use record factorization, Definition 919.9. Every record read of the packet factors through the finite rows $Q, B, W, D, R, S, E, H, C, P, N$, so no external analytic object can replace the displayed prefix route.

The diagonal-consumption theorem above identifies the point at which the seal-facing consumer may read R and S. Since that access comes only after Q, B, W, D, a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer that substitutes a host equality, quotient stream, selected limit, choice principle, hidden tail schedule, or ambient completeness witness leaves the accepted carrier. □

Theorem 919.13 (Finite-prefix limit-budget observer lock)**.** *For every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier*

$$F = (Q, B, W, D, R, S, E, H, C, P, N),$$

each served finite observation is locked to the displayed request row Q, the finite selector row B, the $\mathsf{StreamName}^{\uparrow}$ *prefix-window row W, and the prefix-depth exhaustion row D. No observer of the packet can serve a finite observation through a different request, selector, StreamName window, or prefix-depth row.*

Proof. Project the accepted carrier from Definition 919.1. The only observation-facing rows before readback are Q, B, W, D, and the carrier exports no alternate request coordinate or alternate prefix-window coordinate.

Use the NameCert obligations of Definition 919.2. The selector row B admits the displayed request Q, the window row W is served under that selector, and the exhaustion row D accounts for every offset admitted by W.

Finally apply carrier-classifier scope, Definition 919.8. It confines every classifier read to the same displayed finite prefix-budget surface, so an observer cannot detach a served finite observation from Q, B, W, D. □

Theorem 919.14 (Finite-prefix limit-budget outside-tail refusal)**.** *For every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier, any outside-prefix read or hidden-tail read is refused before the regular readback row R or the seal row S becomes available. The refusal is recorded by the prefix-exhaustion row D together with the exclusion ledger E.*

Proof. Start with prefix-exhaustion coverage, Definition 919.5. It accounts for every admitted offset through D and refuses every outside-prefix read before R or S can be consumed.

Next use seal ordering, Definition 919.3. Since every seal-facing route is ordered as $Q \to B \to W \to D \to R \to S$, no consumer reaches regular readback or the seal before the exhaustion row has performed its refusal work.

It remains to block hidden tails. Apply the nonescape criterion, Definition 919.7: hidden tail schedules occur only as refused claims in E. Thus outside-prefix and hidden-tail reads are both rejected within the finite packet before R or S is available. □

Theorem 919.15 (Finite-prefix limit-budget seal-consumer coverage)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier receives exactly the displayed prefix-budget route*

$$Q \to B \to W \to D \to R \to S$$

together with the exclusion, transport, continuation, provenance, and local naming rows E, H, C, P, N. Such a consumer receives no quotient stream, selected limit, host real equality, hidden tail schedule, countable choice principle, or ambient completeness witness.

Proof. First compose the Real-route handoff of Definition 919.4 with the regular readback path fixed by the carrier. This places the completion-facing consumer behind the finite selector, StreamName prefix windows, RegSeqRat readback, and Real-facing seal rows.

Next use record factorization and the consumer boundary, Definition 919.9 and Definition 919.10. Together they say that $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers read only $Q, B, W, D, R, S, E, H, C, P, N$

Finally apply the Real seal non-escape theorem, Definition 919.12. It excludes quotient streams, selected limits, host real equality, hidden tail schedules, countable choice, and ambient completeness from the seal-facing route, leaving exactly the displayed prefix-budget route and its local certificate rows. □

Theorem 919.16 (Finite-prefix limit-budget seal-window route totality)**.** *For every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier*

$$F = (Q, B, W, D, R, S, E, H, C, P, N),$$

each packet accepted by a $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer routes through*

$$Q \to B \to W \to D \to R \to S$$

with finite prefix-window totality completed before any seal read. The seal consumer may read S only after the request Q, finite budget selector B, prefix-window ledger W, depth-exhaustion row D, and regular-rational readback R have all been displayed.

Proof. Begin with seal ordering, Definition 919.3. It fixes the only completion-facing route as $Q \to B \to W \to D \to R \to S$ and refuses any seal read that precedes the prefix-depth exhaustion row.

Next attach prefix-exhaustion coverage and diagonal selector consumption. Definition 919.5 accounts for every admitted offset inside W through D, while Definition 919.11 prevents R or S from being read before the finite request, budget, window, and exhaustion rows have been consumed.

Finally use seal-consumer coverage. Definition 919.15 says that every $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer receives exactly the displayed route together with E, H, C, P, N. Thus prefix-window totality is already settled inside the finite packet before any completion-facing seal read occurs. □

Theorem 919.17 (Finite-prefix limit-budget completion consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carrier is exhausted by the finite packet rows*

$$Q, B, W, D, R, S, E, H, C, P, N.$$

No such consumer can add quotient streams, selected limits, countable choice, hidden tail schedules, host real equality, or ambient completeness beyond the displayed prefix-budget certificate.

Proof. First use the consumer boundary, Definition 919.10. It identifies the rows delivered to $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers as exactly $Q, B, W, D, R, S, E, H, C, P, N$.

Next apply record factorization and seal-consumer coverage. Definition 919.9 routes every record read through the displayed finite packet, and Definition 919.15 repacks the same rows as the seal-facing surface.

Finally apply the Real seal non-escape extension. Definition 919.12 blocks quotient streams, selected limits, host real equality, countable choice, hidden tail schedules, and ambient completeness from replacing the displayed route. Therefore every completion consumer is exhausted by the same finite prefix-budget packet. □

Theorem 919.18 (Finite-prefix limit-budget public-axis scope)**.** *For every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet*

$$F = (Q, B, W, D, R, S, E, H, C, P, N),$$

any current Phase Real and Completion public read may consume F *only as the finite prefix-budget route through* $Q, B, W, D, R, S, E, H, C, P, N$*. The route preserves the finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RealObservationBudget}^{\uparrow}$*-facing seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It does not create host real equality, quotient streams, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data,* $\mathsf{bridgeChecked}$ *status, or an aggregate formal-status token.*

Proof. Begin with seal ordering, Definition 919.3. It fixes the public read behind the finite route $Q \to B \to W \to D \to R \to S$, so the seal-facing consumer cannot bypass the displayed prefix-depth exhaustion row.

Next use the consumer boundary, Definition 919.10. It identifies the completion-facing read surface with $Q, B, W, D, R, S, E, H, C, P, N$ and keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows inside the same finite packet.

Finally apply completion consumer exhaustion, Definition 919.17, together with the two-axis discipline of section 9467.2. The former excludes host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, and metric-completeness data from the packet; the latter prevents this paper-side public route from manufacturing $\mathsf{bridgeChecked}$ status or any aggregate verification token. □

Theorem 919.19 (Finite-prefix limit-budget Real-completion root unblock)**.** *Every Real-completion or diagonal-limit consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet must read the finite route*

$$Q \to B \to W \to D \to R \to S$$

through the displayed FiniteObservationBudgetSelector, $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealObservationBudget}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows. The same route supplies the exclusion, transport, continuation, provenance, and local naming rows* E, H, C, P, N*. No such consumer receives a quotient stream, host real equality, selected limit, countable choice principle, hidden tail schedule, or ambient completeness coordinate.*

Proof. Start from the carrier definition, Definition 919.1. It names the packet rows $Q, B, W, D, R, S, E, H, C, P, N$ and makes the finite selector, prefix-window ledger, regular-rational readback, Real-facing seal, and local naming rows part of one carrier.

Apply the NameCert obligations and seal ordering, Definition 919.2 and Definition 919.3. They force every seal-facing consumer to pass through $Q \to B \to W \to D \to R \to S$ before the seal row is available.

Finally use the Real-route handoff, completion-consumer exhaustion, and public-axis scope, Definition 919.4, Definition 919.17, and Definition 919.18. These results identify the Real-completion

and diagonal-limit read surface with the same finite packet and exclude quotient streams, host real equality, selected limits, countable choice, hidden tails, and ambient completeness data. □

Theorem 919.20 (Finite-prefix limit-budget RegSeqRat StreamName interface)**.** *For every accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *are exposed as one finite interface. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer sees only prefix windows admitted by the finite budget selector* B *and exhausted by the depth row* D *before the readback row* R *is available.*

Proof. Use the carrier definition to read B, W, D, R as rows of the same finite packet. The packet does not offer a separate StreamName object or a separate RegSeqRat readback surface outside $Q, B, W, D, R, S, E, H, C, P, N$.

Apply the NameCert obligations. The budget selector row B admits the displayed finite request, the prefix-window row W is served under that selector, and the exhaustion row D accounts for the offsets admitted by W.

Now apply seal ordering together with the public-axis scope theorem. Since the route is ordered as $Q \to B \to W \to D \to R \to S$, a $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot read R before W has been admitted by B and exhausted by D. □

Theorem 919.21 (Finite-prefix limit-budget CauchyLimitSeal consumer scope)**.** *Every* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet receives only*

$$Q, B, W, D, R, S, E, H, C, P, N.$$

The consumer may use these rows as the finite seal route, but it may not replace the route by a completed limit, quotient stream, choice-derived tail, host real equality witness, hidden tail schedule, or ambient completeness coordinate.

Proof. Begin with the carrier definition and NameCert obligations. Together they display the exact packet rows and assign request admission, prefix-window service, exhaustion, regular readback, seal handoff, exclusion, transport, continuation, provenance, and naming to $Q, B, W, D, R, S, E, H, C, P, N$.

Use seal ordering and the Real-route handoff. These results keep the $\mathsf{CauchyLimitSeal}^{\uparrow}$ read behind the finite path $Q \to B \to W \to D \to R \to S$ and prevent the seal row from being reached by a separate completion coordinate.

Finally apply completion-consumer exhaustion and public-axis scope. They state that the consumer boundary is exhausted by the displayed rows and that completed limits, quotient streams, choice-derived tails, host real equality, hidden tail schedules, and ambient completeness data are not part of the packet. □

Theorem 919.22 (Finite-prefix limit-budget tail-modulus fusion)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Any* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *consumer of* F *may enter the tail-modulus fusion surface only through the displayed finite prefix route*

$$Q \to B \to W \to D \to R \to S.$$

The handoff exposes the finite selector, prefix window, exhaustion row, regular-rational readback, and seal row already present in F*; it does not add a hidden tail schedule, selected limit, quotient stream, host real equality witness, countable-choice row, or ambient completeness coordinate.*

Proof. Start with the seal-window route totality of Definition 919.16. It fixes the only completion-facing route of F as $Q \to B \to W \to D \to R \to S$ before any seal-facing consumer can read S.

Next use the RegSeqRat StreamName interface, Definition 919.20. The prefix window W, exhaustion row D, and readback row R are therefore already joined inside the finite packet that a tail-modulus consumer receives.

Now compare with the tail-modulus fusion certificate of chapter 1024. Its completion-facing read is a displayed fusion of tail, modulus, threshold, selector-window, rational-readback, and Real-seal rows, so the incoming finite prefix-budget route supplies only the visible selector-window side of that surface.

Finally apply completion consumer exhaustion and CauchyLimitSeal consumer scope, Definition 919.17 and Definition 919.21. They exclude hidden tails, selected limits, quotient streams, host real equality, countable choice, and ambient completeness data from the handoff, leaving the route through Q, B, W, D, R, S as the entire prefix-budget contribution to $\mathsf{CauchyTailModulusFusion}^{\uparrow}$. □

Theorem 919.23 (Finite-prefix limit-budget stability handoff)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Any* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *replay of F preserves the displayed rows $Q, B, W, D, R, S, E, H, C, P, N$ componentwise under the same* hsame *transports and* Cont *replays. The stability surface may read the finite budget, StreamName window, RegSeqRat readback, Real-facing seal, provenance, and local NameCert rows, but it may not replace them by a detached selector, independent stream object, selected limit, quotient stream, or ambient Real-completeness witness.*

Proof. Begin with the finite route of Definition 919.16. The rows Q, B, W, D, R, S are displayed before the seal row is available, so a stability replay inherits an ordered prefix-budget route rather than an unstructured completion object.

Use the RegSeqRat StreamName interface, Definition 919.20. It keeps W, D, R in one finite read surface, which is exactly the StreamName and RegSeqRat side consumed by the stability certificate of chapter 1226.

The same packet carries the transport and continuation rows H and C. Reading the stability chapter through those rows makes the replay componentwise: the hsame comparison transports each displayed row, and the Cont replay returns the same finite prefix-budget surface to the consumer.

Finally apply completion consumer exhaustion and CauchyLimitSeal consumer scope, Definition 919.17 and Definition 919.21. These results prevent a detached selector, independent stream object, selected limit, quotient stream, host real equality, choice-derived tail, or ambient completeness coordinate from entering the stability handoff. □

Theorem 919.24 (Finite-prefix limit-budget diagonal seal determinacy)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Any diagonal-limit or Real-completion seal consumer receiving both the tail-modulus fusion handoff of Definition 919.22 and the stability handoff of Definition 919.23 has a unique displayed seal route through*

$$Q \to B \to W \to D \to R \to S.$$

The consumer may combine the $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *and* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *surfaces only along that route; it cannot choose a second seal path through a hidden tail schedule, detached stability replay, selected limit, quotient stream, host real equality witness, countable-choice row, or ambient completeness coordinate.*

Proof. The local route is already unique at the prefix-budget boundary. By Definition 919.16, the seal row is reached only after Q, B, W, D, R, and by Definition 919.20, the StreamName and RegSeqRat readback rows are not detachable from the finite window and exhaustion rows.

The tail-modulus side adds no second route. Definition 919.22 sends the prefix-budget packet into the $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ surface of chapter 1024 only through $Q \to B \to W \to D \to R \to S$.

The stability side preserves the same rows. Definition 919.23 reads the $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ surface of chapter 1226 through componentwise hsame transport and displayed Cont replay, so the replay cannot manufacture a detached seal path.

Finally use completion consumer exhaustion and CauchyLimitSeal consumer scope, Definition 919.17 and Definition 919.21. They remove hidden tail schedules, selected limits, quotient streams, host real equality, countable choice, and ambient completeness from the consumer boundary. Hence the diagonal-limit or Real-completion consumer has only the displayed seal route through $Q \to B \to W \to D \to R \to S$. □

Theorem 919.25 (Finite-prefix limit-budget common-refinement determinacy)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

and

$$F' = (Q', B', W', D', R', S', E', H', C', P', N')$$

be accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *carriers serving the same* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer request. If the two packets admit a common finite* $\mathsf{StreamName}^{\uparrow}$ *refinement window, then both consumer reads factor through the same displayed selector and exhaustion route before either seal row is available.*

The comparison is determined by the common W/W' *refinement, the* D/D' *exhaustion rows, and the* R/R' *regular readbacks. It is not determined by quotient streams, selected limits, host real equality, countable choice, or ambient completeness.*

Proof. Start from the carrier definition. It presents each accepted packet only through the displayed rows $Q, B, W, D, R, S, E, H, C, P, N$ and $Q', B', W', D', R', S', E', H', C', P', N'$, so a shared consumer request has no hidden coordinate outside the two finite packets.

Apply finite prefix-window route totality, Definition 919.16. For each packet, the seal-facing route is forced through the finite selector, prefix window, exhaustion row, and regular readback before the seal row is reachable.

Next use the RegSeqRat StreamName interface, Definition 919.20. The common $\mathsf{StreamName}^{\uparrow}$ refinement window is therefore read as a common finite comparison of W/W', exhausted by D/D', and exposed to the consumer through R/R' rather than through a separate stream object.

Finally apply the CauchyLimitSeal consumer scope, Definition 919.21. It blocks completed limits, quotient streams, choice-derived tails, host real equality, hidden tail schedules, and ambient completeness data from entering the consumer boundary, so the common-refinement comparison is already determined before either seal row is available. □

Theorem 919.26 (Finite-prefix limit-budget prefix-selector totality)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer request admitted by* F *first factors through the displayed request row* Q *and finite selector row* B*. The consumer cannot read the prefix window* W*, exhaustion row* D*, regular-rational readback* R*, or seal row* S *unless the finite selector row has already admitted the request.*

Equivalently, the packet offers no direct route from a consumer request to W, D, R, S*, no quotient stream standing behind* W*, no selected tail standing behind* R*, and no host Real equality witness standing behind* S*.*

Proof. Start with the NameCert obligations of Definition 919.2. Their prefix-budget selection row places the admitted request Q through the finite selector row B before any $\mathsf{StreamName}^{\uparrow}$ window is served.

Apply seal-window route totality, Definition 919.16. It fixes the only seal-facing route as

$$Q \to B \to W \to D \to R \to S.$$

Use the RegSeqRat StreamName interface, Definition 919.20. The rows W, D, R are read as one finite prefix-window and regular-readback surface, so they are not available before the selector row has admitted the request.

Finally apply common-refinement determinacy, Definition 919.25. A second consumer packet serving the same request is compared only through a common finite selector and window refinement, which rules out a direct read of W, D, R, S outside Q, B. □

Theorem 919.27 (Finite-prefix limit-budget prefix-window stability)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

and

$$F' = (Q', B', W', D', R', S', E', H', C', P', N')$$

be accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packets related by the componentwise* hsame *transports displayed in* H *and* H'*. If* F *and* F' *serve the same finite* $\mathsf{StreamName}^{\uparrow}$ *window request for a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, then the transported comparison preserves the finite window rows* W/W'*, the exhaustion rows* D/D'*, and the readback rows* R/R'*.*

The transport may replay the displayed Cont *routes, but it cannot create quotient streams, selected tails, a detached StreamName object, or an ambient completeness coordinate.*

Proof. First apply the NameCert obligations of Definition 919.2. They identify H and H' as componentwise transport rows attached to the displayed packets, not as sources of new analytic coordinates.

Next use the RegSeqRat StreamName interface, Definition 919.20. It binds the $\mathsf{StreamName}^{\uparrow}$ window rows W/W' to the exhaustion rows D/D' and the regular-rational readback rows R/R' inside the finite prefix surface.

Apply the stability handoff theorem, Definition 919.23. Its replay of the $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ surface preserves $Q, B, W, D, R, S, E, H, C, P, N$ componentwise under hsame transport and displayed Cont replay.

Finally invoke common-refinement determinacy, Definition 919.25. Since packets serving the same request compare through a common finite window refinement before either seal row is reached, the transported comparison has no route to quotient streams, selected tails, detached StreamName objects, or ambient completeness data. □

Theorem 919.28 (Finite-prefix limit-budget seal-read factorization)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Every seal-facing read of* F *factors through the full displayed packet*

$$Q, B, W, D, R, S, E, H, C, P, N.$$

The factorization first admits the request through Q, B*, then reads the finite prefix window and RegSeqRat surface through* W, D, R*, then reaches the seal row* S*, and finally replays the exclusion, transport, continuation, provenance, and local NameCert rows* E, H, C, P, N*.*

Thus a $\mathsf{Real}^{\uparrow}$*, diagonal-limit, or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer may combine the tail-modulus fusion handoff, stability handoff, and common-refinement comparison only through that displayed packet; it may not read a hidden tail schedule, quotient stream, selected limit, host Real equality witness, countable-choice row, or ambient completeness coordinate.*

Proof. Begin with prefix-selector totality, Definition 919.26. It forces every admitted seal-facing consumer request through Q, B before any window, readback, or seal row is visible.

Use seal-window route totality and the RegSeqRat StreamName interface, Definition 919.16 and Definition 919.20. Together they identify the finite route $Q \to B \to W \to D \to R \to S$ and keep W, D, R inside the same prefix-window readback surface.

Next apply tail-modulus fusion and stability handoff, Definition 919.22 and Definition 919.23. The former admits only the displayed prefix contribution to $\mathsf{CauchyTailModulusFusion}^{\uparrow}$, while the latter replays the same rows componentwise under hsame and Cont.

Finally apply common-refinement determinacy, Definition 919.25. Any second packet serving the same seal-facing request is compared by a common finite refinement of W, D, R, so the consumer has no remaining route to hidden tails, quotient streams, selected limits, host equality, choice, or ambient completeness outside $Q, B, W, D, R, S, E, H, C, P, N$. □

Theorem 919.29 (Finite-prefix limit-budget public interface export)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. The packet is the public Phase Real and Completion interface exported to* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$*, and* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *consumers. Every such public read factors through the finite prefix-budget route*

$$Q \to B \to W \to D \to R \to S$$

and then through the displayed exclusion, transport, continuation, provenance, and local naming rows E, H, C, P, N*.*

The exported interface carries no host real equality, quotient stream, selected limit, countable-choice row, hidden tail schedule, ambient completeness coordinate, bridge-status certificate, or aggregate verification token beyond that displayed packet.

Proof. Start with public-axis scope and the Real-completion root unblock, Definition 919.18 and Definition 919.19. These identify the Phase Real, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ read surface with the same finite packet $Q, B, W, D, R, S, E, H, C, P, N$ and force the seal-facing route through $Q \to B \to W \to D \to R \to S$.

Next apply tail-modulus fusion and stability handoff, Definition 919.22 and Definition 919.23. The first admits a $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ consumer only through the displayed prefix contribution, and the second replays the $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ surface componentwise under the same hsame transports and Cont routes.

Finally use diagonal seal determinacy and seal-read factorization, Definition 919.24 and Definition 919.28. They leave no second seal path and repack every seal-facing public read through $Q, B, W, D, R, S, E, H, C, P, N$. The exclusions already recorded by the six cited results rule out host real equality, quotient streams, selected limits, countable choice, hidden tail schedules, ambient completeness, bridge-status certificates, and aggregate verification tokens. □

Theorem 919.30 (Finite-prefix limit-budget acceptance-bridge interface)**.** *Let*

$$F = (Q, B, W, D, R, S, E, H, C, P, N)$$

be an accepted $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *packet. Every acceptance-bridge consumer that transports the accepted packet preserves the same public route*

$$Q \to B \to W \to D \to R \to S$$

and the same support rows E, H, C, P, N.

The bridge may transport the public interface exported by Definition 919.29 across the public axis scope of Definition 919.18 and the completion-consumer exhaustion boundary of Definition 919.17. It cannot add host real equality, quotient streams, selected limits, countable choice, ambient completeness, bridge-status certificates, or an aggregate verification token.

Proof. Begin with the public interface export, Definition 919.29. It identifies the whole Phase Real and Completion read surface with the displayed route $Q \to B \to W \to D \to R \to S$ followed by E, H, C, P, N.

Next apply public-axis scope, Definition 919.18. The public axis may expose the finite route to $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers, but it does not manufacture a second analytic coordinate or a verification aggregate.

Finally use completion-consumer exhaustion, Definition 919.17. Since the completion-facing consumer has already exhausted the packet through $Q, B, W, D, R, S, E, H, C, P, N$, an acceptance bridge can only transport that interface. The exclusions in the three cited results rule out host real equality, quotient streams, selected limits, countable choice, ambient completeness, bridge-status certificates, and aggregate verification tokens outside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ packet is generated from a finite diagonal request, finite observation-budget selector row, finite StreamName prefix-window ledger, prefix-depth exhaustion row, RegSeqRat readback row, Real-facing seal row, exclusion ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitBudgetUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 919.1, Definition 919.2, Definition 919.15, Definition 919.16, Definition 919.17, Definition 919.22, Definition 919.23, Definition 919.25, Definition 919.29, and Definition 919.30, together with the visible TasteGate carrier marker, close the mature finite prefix-budget treatment: public interface export, acceptance-bridge interface, tail-modulus fusion handoff, stability handoff, and common-refinement theorem over the route $Q \to B \to W \to D \to R \to S$ and rows E, H, C, P, N.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixLimitBudget}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.FinitePrefixLimitBudgetUp.FinitePrefixLimitBudgetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove real completeness, select a completed limit, identify quotient streams, assert host real equality, use countable choice, or close Lean implementation closure.

Upgrade path: bridgeChecked remains a Lean-side bridge task, while the paper-side mature route is closed by the displayed finite-prefix interface $Q \to B \to W \to D \to R \to S$ with rows E, H, C, P, N.

Chapter 920

A Concrete Naming Certificate for $\mathsf{CofinalTailAgreement}^{\uparrow}$

The $\mathsf{CofinalTailAgreement}^{\uparrow}$ packet realizes the paired cofinal-tail agreement route required by the Real completeness surface of chapter 9238. It sits between the shared-threshold route of chapter 560, the fused tail route of chapter 658, the selector-budget route of chapter 678, the displayed Real agreement seal of chapter 699, and the sealed Cauchy-limit route of chapter 386. The packet records the paired-source, shared-threshold, dyadic-readback, and seal-handoff rows as one finite $\mathsf{NameCert}$ carrier rather than as an ambient completeness principle.

Definition 920.1 (Cofinal tail-agreement carrier)**.** *A* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, F, B, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *shared-threshold row for the paired regular-Cauchy sources;*
2. *F is the* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ *row that fuses the paired cofinal tail witnesses through the shared threshold;*
3. *B is the* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ *row that records the finite selector budget used by the paired route;*
4. *S is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row that receives the paired agreement boundary;*
5. *D is the dyadic readback row consumed before the seal handoff is visible;*
6. *R records the regular-rational source readbacks visible to the paired route;*
7. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row consumed by completion-facing readers;*
8. *H records componentwise* hsame *transport for all displayed rows;*
9. *C records the* Cont *routes by which consumers read the shared threshold, fusion, budget, dyadic readback, Real agreement seal, and Cauchy-limit handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$ *row.*

The carrier contains no quotient stream equality, selected host limit, external tail oracle, countable-choice row, or ambient Real-completeness theorem.

Lean4 ✓ `BEDC.Derived.CofinalTailAgreementUp.CofinalTailAgreementTasteGate single carrier alignment`

Theorem 920.2 (Cofinal tail-agreement NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$ *surface for Definition 920.1 consists of six obligation rows: carrier admission for* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, shared-threshold exactness through* M*, fused cofinal-tail exactness through* F*, selector-budget readback through* B*, dyadic agreement readback through* D, R, S*, Cauchy-limit seal handoff through* L*, and non-escape through* H, C, P, N*.*

Proof. Carrier admission is projection from the displayed packet. The rows already expose the shared threshold, fused tail route, selector budget, dyadic readback, regular source readbacks, Real agreement seal, Cauchy-limit handoff, transports, routes, provenance, and local naming.

The dependency rows are read in order. The shared threshold M supplies the common tail bound, F fuses the paired cofinal witnesses through that bound, and B records the finite selector budget that the paired route may use.

The consumer-facing rows are then forced. The dyadic readback D and regular readbacks R feed the Real agreement seal S, and the completion-facing reader reaches the Cauchy-limit handoff L only through the same displayed route. Since H, C, P, N name all transports, continuations, provenance, and local certificate rows, no hidden tail selector or ambient completeness principle is exported. □

Theorem 920.3 (Cofinal tail-agreement carrier admission)**.** *Every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet admits the displayed* BHist *carrier*

$$A = (M, F, B, S, D, R, L, H, C, P, N)$$

before any public Real-completion consumer may read a tail-agreement result. The admission row exposes M, F, B, D, R, S, L *as source, budget, readback, seal, and handoff data, while* H, C, P, N *expose componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 920.1. Its rows are already a finite BHist packet rather than quotient-stream or host-limit data.

Apply the obligation surface of Definition 920.2. Carrier admission is the first obligation, and the same surface names shared-threshold exactness, fused-tail exactness, selector-budget readback, dyadic readback, seal handoff, and non-escape.

The TasteGate marker following Definition 920.1 records the same single-carrier alignment. Thus a public consumer starts from $A = (M, F, B, S, D, R, L, H, C, P, N)$ and cannot bypass the hsame, Cont, Pkg, or $\mathsf{NameCert}$ support rows. □

Theorem 920.4 (Cofinal tail-agreement tail readback)**.** *Every public tail readback of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet factors through the full row route*

$$M, F, B, D, R, S, L, H, C, P, N.$$

The readback first uses the shared threshold and fused cofinal tail rows M, F*, then the finite selector budget* B*, then the dyadic and regular source readbacks* D, R*, then the Real agreement seal* S *and Cauchy-limit handoff* L*, with* H, C, P, N *preserving the same route.*

Proof. Begin with carrier admission, Definition 920.3. It makes the full row set visible before any consumer-facing agreement read.

Use Definition 920.2. The obligations place shared-threshold exactness before fused-tail exactness, selector-budget readback before dyadic and regular readback, and seal handoff after those readback rows.

Finally apply the consumer factorization pattern established by Definition 920.6. The rows H, C, P, N do not select a new tail; they transport, replay, package, and name the already displayed route. □

Theorem 920.5 (Cofinal tail-agreement cofinal nonescape)**.** *No accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier exports quotient stream equality, a selected host limit, countable choice, or ambient Real completeness as a substitute for the displayed cofinal route* $M, F, B, D, R, S, L, H, C, P, N$*. Any Completion-facing consumer that omits one of these rows cannot be repacked as a public read of the accepted packet.*

Proof. Read the admitted carrier from Definition 920.3. The finite packet supplies exactly the displayed rows and no quotient-stream, selected-limit, choice, or ambient-completeness coordinate.

Apply tail readback, Definition 920.4. The public route to the Real and Completion handoff passes through M, F, B, D, R, S, L and then through the support rows H, C, P, N.

Use the existing non-escape theorem, Definition 920.8. A route omitting the shared threshold, fused tail, budget, readbacks, seal, handoff, transport, continuation, provenance, or local name is outside the packet, so none of the refused host objects can be exported as the cofinal tail agreement. □

Theorem 920.6 (Cofinal tail-agreement consumer factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet factors through the shared-threshold row* M*, the fused tail row* F*, the selector-budget row* B*, the dyadic readback row* D*, the regular readback row* R*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* S*, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L *before it can read a completion-facing agreement result. The consumer receives the paired route*

$$M, F, B, D, R, S, L, H, C, P, N$$

and receives no quotient stream, selected limit, host equality, hidden tail oracle, or ambient completeness theorem.

Proof. Start from the accepted carrier. Its only consumer-facing rows are $M, F, B, D, R, S, L, H, C, P, N$, so any public read begins with the displayed finite packet.

Use the sibling tail surfaces in order. The $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ row fixes the shared threshold, the $\mathsf{RegularCauchyTailFusion}^{\uparrow}$ row fuses the paired cofinal witnesses through that threshold, and the $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$ row records the finite selector budget used by the paired read.

Now route the seal handoff. The dyadic and regular readbacks feed the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row, and the completion-facing side then consumes the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row through the same H, C, P, N provenance. No carrier coordinate remains for a second tail selector or a separate completion route. □

Theorem 920.7 (Cofinal tail-agreement Real and Completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reaches the paired agreement result of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier must pass through the route*

$$M, F, B, D, R, S, L, H, C, P, N,$$

with the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* S *consumed before the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L*. The consumer cannot replace this route by quotient stream equality, host stream equality, a selected limit, or an ambient completeness theorem.*

Proof. Begin with the local NameCert obligations. The carrier admission row exposes only $M, F, B, S, D, R, L, H, C$ while the shared-threshold, fused-tail, selector-budget, dyadic-readback, and Cauchy-limit handoff obligations force the same displayed rows to be consumed by any public reader.

Apply Definition 920.6. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer reaches the paired agreement only through $M, F, B, D, R, S, L, H, C, P, N$, so the agreement result is not available before the shared threshold, fused tail, budget, dyadic readback, regular readback, and seal rows have been read.

Use Definition 920.9 and Definition 920.10. The shared threshold and selector budget are single displayed rows, and the handoff has exactly one $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row before exactly one $\mathsf{CauchyLimitSeal}^{\uparrow}$ row.

Finally apply Definition 920.11. Quotient stream equality, host stream equality, selected limits, and ambient completeness all bypass the displayed route, so none can be repacked as the Real or Completion handoff of the accepted $\mathsf{CofinalTailAgreement}^{\uparrow}$ carrier. □

Theorem 920.8 (Cofinal tail-agreement non-escape)**.** *An accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet exports only the paired-source, shared-threshold, dyadic-readback, Real-agreement-seal, and Cauchy-limit handoff rows of Definition 920.1. A route that omits the shared threshold* M*, the fused tail row* F*, the selector budget* B*, the dyadic readback* D*, the Real agreement seal* S*, or the Cauchy-limit handoff* L *is outside the packet and cannot be repacked as the paired cofinal-tail agreement route.*

Proof. Inspect the carrier inventory. The packet has exactly the displayed rows $M, F, B, S, D, R, L, H, C, P, N$ and no coordinate for quotient streams, host equality, selected limits, hidden tail oracles, or external completeness data.

Apply Definition 920.2. Its obligation rows make the shared threshold, fused tail, selector budget, dyadic readback, Real agreement seal, and Cauchy-limit handoff mandatory for the public route.

Finally apply Definition 920.6. Every Real or Completion consumer reaches the agreement result through the same displayed rows, so an attempted route omitting any of the mandatory rows has no continuation that can export the packet. □

Theorem 920.9 (Cofinal tail-agreement shared threshold budget)**.** *For every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, the paired cofinal-tail route has a single shared threshold row* M *and a single finite selector-budget row* B*. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads the agreement through* S *or* L *must first factor through* M, F, B *and the same transport, replay, provenance, and naming rows* H, C, P, N*.*

Proof. Project the accepted carrier. The only threshold coordinate is M, and the only selector-budget coordinate is B; the fused paired route is the row F between them.

Use Definition 920.2. Shared-threshold exactness, fused cofinal-tail exactness, and selector-budget readback are separate obligation rows, so no consumer can replace M, F, B by an external threshold or a second selector budget.

Finally apply Definition 920.6. Both Real and Completion reads reach the seal handoff through the same $M, F, B, D, R, S, L, H, C, P, N$ route, so the shared threshold and finite budget are not optional decorations of the packet. □

Theorem 920.10 (Cofinal tail-agreement seal handoff determinacy)**.** *In an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, the handoff from the paired cofinal-tail route to* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *and then to* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *is determined by the displayed rows* $M, F, B, D, R, S, L, H, C, P$*. A consumer preserving those rows cannot choose a second Real agreement seal, a second Cauchy-limit seal, or a selected host limit.*

Proof. The carrier has exactly one Real agreement seal row S and exactly one Cauchy-limit handoff row L. The dyadic and regular readbacks D, R are the only displayed rows feeding the seal handoff.

Read the obligation surface of Definition 920.2. It names dyadic agreement readback through D, R, S and Cauchy-limit seal handoff through L as mandatory local certificate rows.

Transport and replay do not introduce another seal. The rows H, C, P, N only preserve and expose the same displayed packet, so any consumer preserving $M, F, B, D, R, S, L, H, C, P, N$ has no coordinate from which to select a second host limit or an alternate seal path. □

Theorem 920.11 (Cofinal tail-agreement quotient stream refusal)**.** *No accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$ *exports quotient-stream equality, host stream equality, a selected limit, or an ambient completeness theorem. Every attempted consumer of such data factors through the refusal already expressed by the missing carrier coordinate and therefore cannot be repacked as a* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *read.*

Proof. Inspect the definition of the carrier. Its public coordinates are $M, F, B, S, D, R, L, H, C, P, N$, and the definition explicitly excludes quotient stream equality, selected host limits, external tail oracles, countable-choice rows, and ambient Real-completeness data.

Apply the non-escape theorem, Definition 920.8. A route that omits the shared threshold, fused tail, selector budget, dyadic readback, Real agreement seal, or Cauchy-limit handoff is outside the packet.

The refused data omit the displayed route by construction. Quotient-stream equality and host equality bypass M, F, B, D, R, S, L; selected limits bypass the finite budget B and handoff row L; ambient completeness bypasses the local $\mathsf{NameCert}$ row N. Hence none of them can be repacked as an accepted paired cofinal-tail agreement route. □

Theorem 920.12 (Cofinal tail-agreement shared-threshold exposure)**.** *For every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, every public* $\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$ *read exposes the shared-threshold row* M *before it can read the fused cofinal-tail row* F*, the finite selector-budget row* B*, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* S*, or the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L*. The row* M *is therefore a displayed* BHist *coordinate of the packet, not a threshold recovered from quotient streams, host limits, or ambient completeness.*

Proof. Apply Definition 920.2. Its obligation surface separates shared-threshold exactness through M from fused-tail exactness through F, selector-budget readback through B, dyadic agreement readback through D, R, S, and Cauchy-limit handoff through L.

Next use Definition 920.6. Every public Real or Completion consumer reads the agreement through $M, F, B, D, R, S, L, H, C, P, N$, so no later seal row is available before the shared threshold has been displayed.

Finally apply Definition 920.8. A route omitting M lies outside the accepted packet, and Definition 920.11 prevents quotient stream equality, host equality, selected limits, or ambient completeness from replacing that missing row. □

Theorem 920.13 (Cofinal tail-agreement selector-budget readback)**.** *For every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, the row* B *is the finite selector-budget coordinate*

through which the dyadic readback D, the regular-source readback R, the Real agreement seal S, and the Cauchy-limit handoff L are publicly read. No consumer of the accepted packet receives a second selector budget, hidden selector oracle, selected host limit, or budget-free path from D, R to S, L.

Proof. Read the carrier and the obligation rows of Definition 920.2. The selector-budget obligation names B separately from M, F, D, R, S, L, so the budget is a finite row of the same BHist packet rather than a property inferred after the seal.

Apply Definition 920.6. The displayed consumer route places B before D, R, S, L, and every Real or Completion-facing read of those rows proceeds through the same transport, continuation, provenance, and local naming rows H, C, P, N.

Use Definition 920.7 and Definition 920.11. The handoff never exports a selected limit, hidden selector, host equality, or ambient completeness coordinate, so no budget-free route can be repacked as an accepted $\mathsf{CofinalTailAgreement}^{\uparrow}$ read. □

Theorem 920.14 (Cofinal tail-agreement seal handoff coverage)**.** *For every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier $A = (M, F, B, S, D, R, L, H, C, P, N)$, every Completion-facing consumer reaches the handoff row L only after passing through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row S on the same $M, F, B, D, R, H, C, P, N$ route. The consumer receives no quotient-stream equality, host limit, selected-limit coordinate, or ambient completeness row as part of that handoff.*

Proof. Begin with Definition 920.7. It states that every Real or Completion consumer reaching the paired agreement result passes through $M, F, B, D, R, S, L, H, C, P, N$, with S consumed before L.

Then apply Definition 920.6. The same route is the whole public consumer surface, so Completion-facing use of L remains tied to the Real agreement seal S and to the displayed carrier rows.

Finally use Definition 920.8 together with Definition 920.11. Quotient stream equality, host limits, selected limits, and ambient completeness each require an absent coordinate, hence none is delivered by the Completion handoff. □

Theorem 920.15 (Cofinal tail-agreement eventual selector exactness)**.** *For every accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet $A = (M, F, B, S, D, R, L, H, C, P, N)$, the selector-budget row B is the unique public route from the shared-threshold row M and fused-tail row F to the dyadic readback row D and regular readback row R. No accepted Real or Completion consumer can read D, R, S, L from M, F by a second selector, hidden tail oracle, quotient stream equality, or ambient completeness row.*

Proof. Begin with the carrier and NameCert obligations. The packet displays exactly one shared-threshold row M, one fused-tail row F, and one selector-budget row B, and the obligation surface reads shared-threshold exposure, fused-tail exactness, and selector-budget readback as separate rows of the same finite certificate.

Use Definition 920.12 and Definition 920.13. They place M before F and force the public read of D, R, S, L through B, with H, C, P, N preserving the same packet rather than supplying an alternate selector.

Finally apply Definition 920.14 and Definition 920.11. The Real and Cauchy-limit handoffs consume the route already built from M, F, B, D, R, while quotient stream equality, selected limits, hidden tail oracles, and ambient completeness require absent coordinates. Thus B is the unique public selector route from the shared threshold and fused tail to the readback rows. □

Theorem 920.16 (Cofinal tail-agreement tail-filter stability)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet preserves the cofinal tail filter represented by the rows $M, F, B, D, R, S, L, H, C, P, N$. The transported read cannot replace that finite filter by quotient*

stream equality, host stream equality, a selected limit, or any equality of completed streams outside the displayed packet.

Proof. Read the transport row H in the carrier. It transports the shared threshold, fused tail, selector budget, dyadic readback, regular readback, Real agreement seal, Cauchy-limit handoff, continuation, provenance, and local naming rows componentwise.

Apply Definition 920.6 and Definition 920.21. Every public consumer route uses the same finite sequence $M, F, B, D, R, S, L, H, C, P, N$, so transporting the packet preserves the filter as a displayed row sequence rather than as a quotient of streams.

The replacement cases are excluded by Definition 920.11. Quotient stream equality, host stream equality, selected limits, and equality of completed streams ask for data the carrier explicitly omits, so no hsame transport can turn them into the accepted tail-filter read. □

Theorem 920.17 (Cofinal tail-agreement completion-window coherence)**.** *The* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff row S and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row L of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet consume the same finite selector window. In particular, both handoffs read the same shared-threshold row M, fused-tail row F, selector-budget row B, dyadic readback row D, regular readback row R, and support rows H, C, P, N.*

Proof. Start from Definition 920.14. It places every Completion-facing use of L after the Real agreement seal S on the same $M, F, B, D, R, H, C, P, N$ route.

Use Definition 920.15. The selector-budget row B is the unique public route from M, F to the dyadic and regular readback rows, so the Real seal cannot consume one selector window while the Cauchy-limit seal consumes another.

Finally apply Definition 920.19 and Definition 920.20. They identify the terminal handoff with the same source route and exclude quotient streams, selected host limits, countable choice, and ambient Real completeness. Hence S and L share one finite selector window. □

Theorem 920.18 (Cofinal tail-agreement modulus seal route)**.** *Every accepted paired agreement route consumed by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *from a* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier $A = (M, F, B, S, D, R, L, H, C, P, N)$ preserves the same shared-threshold row M, fused tail row F, selector-budget row B, dyadic readback row D, regular readback row R, seal row S, terminal handoff row L, transport row H, continuation row C, provenance row P, and local naming row N. A* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *consumer cannot read the paired cofinal agreement through S while changing any of the upstream M, F, B, D, R rows or any of the support rows H, C, P, N.*

Proof. Start from the carrier of Definition 920.1. Its displayed packet is exactly $M, F, B, S, D, R, L, H, C, P, N$, so the modulus-seal consumer has no coordinate outside that finite carrier.

Apply the NameCert obligations of Definition 920.2. They bind shared-threshold exactness through M, fused cofinal-tail exactness through F, selector-budget readback through B, dyadic and regular readback through D, R, S, and terminal handoff through L.

Now use consumer factorization and the Real/Completion handoff, Definition 920.6 and Definition 920.7. Any accepted consumer that reaches the seal row reads the route $M, F, B, D, R, S, L, H, C, P, N$, and the support rows only transport, replay, package, and name that same route.

Finally combine shared-threshold budget, seal-handoff determinacy, quotient-stream refusal, and non-escape, Definition 920.9, Definition 920.10, Definition 920.11, and Definition 920.8. They rule out a second threshold, second budget, altered readback, alternate seal, quotient-stream equality, selected limit, or ambient completeness route for the same $\mathsf{CofinalModulusSeal}^{\uparrow}$ read. □

Theorem 920.19 (Cofinal tail-agreement terminal seal compatibility)**.** *For an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, the handoff* $S \to L$ *is compatible with the terminal* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row only when the same source route* M, F, B, D, R *is displayed before the seal and preserved by* H, C, P, N*. A terminal consumer that reaches* L *without the same* M, F, B, D, R *route is not reading the* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *handoff.*

Proof. Project the accepted packet. The only terminal handoff coordinate is L, and Definition 920.10 makes its predecessor the displayed seal row S fed by D, R on the same M, F, B source route.

The route into Completion is already forced by Definition 920.7. It places S before L and keeps the same $M, F, B, D, R, H, C, P, N$ support around the handoff.

Use Definition 920.14 and Definition 920.18. They show that Completion-facing consumption of L and modulus-seal consumption of S preserve one displayed source route rather than two separately chosen routes.

If a consumer omits or changes M, F, B, D, R, then Definition 920.8 and Definition 920.11 apply. The attempted handoff would require an absent coordinate such as a quotient stream, host limit, selected limit, or ambient completeness row, so it is outside the terminal seal compatibility surface. □

Theorem 920.20 (Cofinal tail-agreement completion source exhaustion)**.** *No Completion-facing consumer of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *carrier can replace the route* $M, F, B, D, R, S, L, H, C, P, N$ *by quotient stream equality, host limits, selected limits, countable choice, or ambient Real completeness. The accepted source for the terminal agreement is exhausted by the displayed shared-threshold, fused-tail, selector-budget, dyadic-readback, regular-readback, seal, handoff, transport, continuation, provenance, and naming rows.*

Proof. Read the carrier and NameCert obligations. The packet lists $M, F, B, S, D, R, L, H, C, P, N$ and explicitly excludes quotient stream equality, selected host limits, external tail oracles, countable-choice rows, and ambient Real-completeness data.

Consumer factorization, Real/Completion handoff, and seal-handoff coverage force every Completion-facing read through the same displayed route. Thus L is not a free limit coordinate; it is the terminal row reached after M, F, B, D, R, S and preserved by H, C, P, N.

Apply the modulus-seal route and terminal compatibility theorems above. They identify the same M, F, B, D, R source route as the one seen by the modulus seal and by the terminal Cauchy-limit seal.

Finally use quotient-stream refusal and non-escape. Each replacement source asks for data that the carrier has refused or omitted, so none can be repacked as a Completion-facing consumer of the accepted $\mathsf{CofinalTailAgreement}^{\uparrow}$ route. □

Theorem 920.21 (Cofinal tail-agreement route coverage)**.** *For every accepted public read of a* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet* $A = (M, F, B, S, D, R, L, H, C, P, N)$*, the visible route covers the rows* $M, F, B, D, R, S, L, H, C, P, N$ *and factors through the* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *surfaces. No accepted Real- or Completion-facing read may consume* S *or* L *while omitting any upstream row among* M, F, B, D, R *or any support row among* H, C, P, N*.*

Proof. Start with the carrier and NameCert obligations. They expose the shared threshold M, fused tail route F, selector budget B, dyadic and regular readbacks D, R, Real agreement seal S, Cauchy-limit handoff L, and the support rows H, C, P, N in one finite BHist packet.

The factorization is supplied by the existing consumer route. Use Definition 920.6, Definition 920.7, and Definition 920.14: together they place the RegularCauchyTailMeet, RegularCauchyTailFusion, RegularCauchySelectorBudget, RealTailAgreementSeal, and CauchyLimitSeal readings on the same displayed route.

It remains to exclude partial public reads. The non-escape and quotient-stream refusal theorems, Definition 920.8 and Definition 920.11, say that an omitted row would have to be replaced by host equality, a selected limit, quotient-stream equality, countable choice, or ambient completeness, none of which is a coordinate of the accepted packet. □

Theorem 920.22 (Cofinal tail-agreement dependency lock)**.** *The sibling rows named in the* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *route are mandatory* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *coordinates of the accepted packet, not prose dependencies. In particular, the RegularCauchyTailMeet threshold, RegularCauchyTailFusion tail route, RegularCauchySelectorBudget budget, RealTailAgreementSeal row, and CauchyLimitSeal handoff remain locked to the same* H, C, P, N *support rows throughout every accepted consumer read.*

Proof. Read Definition 920.1. The sibling data enter the packet as displayed coordinates rather than as external chapter names or informal prerequisites.

The NameCert obligations then assign those coordinates their transport, continuation, provenance, and local naming roles. They bind BHist membership, componentwise hsame stability, Cont replay, Pkg provenance, and $\mathsf{NameCert}$ locality to the same finite route.

Consumer factorization and the handoff theorems preserve that lock downstream. The Real/Completion handoff, modulus-seal route, terminal seal compatibility, and completion source exhaustion results together prevent a consumer from changing a sibling row while keeping the public $\mathsf{CofinalTailAgreement}^{\uparrow}$ name. □

Theorem 920.23 (Cofinal tail-agreement completion boundary)**.** *Every Completion-facing consumer of an accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *packet receives only the finite route* $M, F, B, D, R, S, L, H, C, P, N$*. The consumer cannot substitute quotient streams, selected host limits, countable choice, or ambient Real completeness for the displayed shared threshold, fused tail, selector budget, dyadic readback, regular readback, Real seal, Cauchy-limit handoff, transport, continuation, provenance, and local naming rows.*

Proof. The boundary begins at the Real/Completion handoff. By Definition 920.7 and Definition 920.14, the handoff into L is reached only through the displayed $M, F, B, D, R, S, H, C, P, N$ route.

The terminal compatibility and completion source exhaustion results identify this as the whole Completion-facing source. They leave no hidden quotient stream, selected limit, choice-selected tail, or ambient completeness coordinate behind L.

Finally, quotient-stream refusal and non-escape close the boundary. Any attempted replacement asks for data outside the carrier and therefore cannot be repacked as a $\mathsf{CofinalTailAgreement}^{\uparrow}$ consumer read. □

Theorem 920.24 (Cofinal tail-agreement scoped kernel route)**.** *Every accepted public* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *read factors through the displayed route* $M, F, B, D, R, S, L, H, C, P, N$*. This route is scoped exactly by the carrier's* BHist *rows, componentwise* hsame *transports, displayed* Cont *continuation,* Pkg *provenance, local* $\mathsf{NameCert}$ *row, and the named* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$*,* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$*,* $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$*,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *surfaces. No accepted read may replace any part of the route by quotient*

stream equality, host stream equality, a selected limit, countable choice, or ambient Real completeness.

Proof. Start from Definition 920.1 and Definition 920.2. They present $M, F, B, D, R, S, L, H, C, P, N$ as one finite packet, with H supplying componentwise hsame transport, C supplying the displayed Cont replay, P supplying Pkg provenance, and N supplying local NameCert naming over the StreamName, dyadic, regular, Real, and tail-handoff rows.

Next apply Definition 920.6, Definition 920.21, and Definition 920.22. These results put every public read on the same $M, F, B, D, R, S, L, H, C, P, N$ route and lock the shared threshold, fused tail, selector budget, dyadic readback, regular readback, Real agreement seal, and Cauchy-limit handoff to the same H, C, P, N support.

Definition 920.23 supplies the final exclusion. A reader that asks for quotient stream equality, host stream equality, a selected limit, countable choice, or ambient Real completeness asks for a coordinate not present in the accepted packet. Hence the public read is scoped by exactly the displayed BEDC kernel rows and the listed completion-route surfaces. □

Falsifiable prediction: Deleting the shared cofinal threshold row breaks the Real/Completion consumer route while leaving the one-sided tail-fusion route intact.

Independence witness: The nearest siblings regularcauchytailmeet, regularcauchytailfusion, regularcauchyselectorbudget, realtailagreementseal, and cauchylimitseal share individual rows with $\mathsf{CofinalTailAgreement}^{\uparrow}$, but no bijection preserves all paired-source, shared-threshold, dyadic-readback, and seal-handoff rows at once: the tail-meet packet lacks the selector-budget and seal-handoff rows, the tail-fusion packet lacks the terminal Cauchy-limit handoff, the selector-budget packet lacks the paired agreement seal, the Real tail-agreement seal lacks the upstream fusion budget, and the Cauchy-limit seal lacks the paired-source threshold route.

The Lean `TasteGate.lean` boundary for this carrier must include `Nontrivial CofinalTailAgreementUp` and `FieldFaithful CofinalTailAgreementUp` instances, so the paired shared-threshold route is represented by a nonempty and field-faithful carrier rather than by a vacuous type.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalTailAgreement}^{\uparrow}$ packet is generated from a shared RegularCauchyTailMeet threshold, RegularCauchyTailFusion paired tail route, RegularCauchySelectorBudget finite selector budget, dyadic and regular readbacks, RealTailAgreementSeal handoff, CauchyLimitSeal handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 920.24 binds every accepted public read of the finite paired cofinal-tail agreement carrier to the displayed BEDC kernel rows and Real/Completion route surfaces.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofinalTailAgreement}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.CofinalTailAgreementUp.taste gate`

Lean target: `BEDC.Derived.CofinalTailAgreementUp.CofinalTailAgreementTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close public-object closure, quotient stream equality, selected host limits, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Public closure requires exhibiting the route as a stable public object surface rather than only a scoped finite kernel route.

Chapter 921

A Concrete Naming Certificate for $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$

The $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$ packet realizes the finite tail route behind the Real-completeness target of chapter 9238. It sits after the stream-name surface of chapter 126, the regular rational source of chapter 301, the dyadic readback core of chapter 291, the RealUp seal of chapter 38, and the shared tail-meet route of chapter 560. The packet records one finite tail fiber, its schedule row, and its seal handoff without claiming quotient reals, ambient completeness, or an unrestricted diagonal theorem.

Definition 921.1 (Finite tail-fiber schedule carrier)**.** *A* $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, F, S, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the displayed tail index at which the finite Real-completeness route is read;*
2. *F is the finite fiber row over that tail index, recording the stream observations available at the selected tail;*
3. *S is the schedule row that binds the tail index and finite fiber to the selected RealUp seal route;*
4. *D is the dyadic radius ledger consumed by the finite tail readback;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to interpret the scheduled fiber as regular rational data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the scheduled tail-fiber route;*
7. *H records componentwise* hsame *transport for the tail index, fiber row, schedule row, dyadic radius ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal;*
8. *C records the displayed* Cont *routes for stream-name readback, dyadic-radius use, regular-rational readback, tail-meet comparison, and seal consumption;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTailFiberSchedule}^{\uparrow}}$ *row.*

The classifier compares two carriers only by the displayed rows $T, F, S, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, an infinite tail oracle, a selected host limit, countable choice, or an ambient real-completeness theorem.

Lean4 ✓ `BEDC.Derived.FiniteTailFiberScheduleUp.FiniteTailFiberScheduleTasteGate single carrier alignment`

Theorem 921.2 (FiniteTailFiberSchedule NameCert obligations)**.** *Every accepted* $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$ *packet with carrier Definition 921.1 carries the standard five local naming obligations:*

1. *carrier habitation: the tuple* $Q = (T, F, S, D, R, E, H, C, P, N)$ *is admitted as one finite* BHist *packet;*
2. *classifier transport: the componentwise* hsame *rows in* H *and the displayed* Cont *routes in* C *preserve the tail index, finite fiber, schedule, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, package, and naming rows;*
3. *schedule-to-fiber exactness: every accepted schedule read passes through the displayed tail index* T *and finite fiber row* F *before it can cite a RealUp seal;*
4. *dyadic and regular readback exactness: every seal-facing radius and regular-rational read passes through the dyadic ledger* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*;*
5. *non-escape: accepted carriers export no quotient real, selected host limit, infinite tail oracle, countable-choice row, or ambient Real-completeness theorem.*

Thus the local $\mathsf{NameCert}_{\mathsf{FiniteTailFiberSchedule}^{\uparrow}}$ *surface is exactly the finite schedule from a displayed tail fiber to a displayed* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Carrier habitation is the displayed finite tuple $Q = (T, F, S, D, R, E, H, C, P, N)$. Each coordinate is a visible history row or a finite route between such rows, so admission does not require an external stream space or a host real carrier.

For classifier transport, read the structural rows componentwise. The hsame data in H preserve each displayed coordinate, and the Cont routes in C replay the stream-name, dyadic-radius, regular-rational, tail-meet, and seal-consumption reads without introducing another tail object.

Schedule-to-fiber exactness follows from the shape of S. A schedule row is accepted only when it binds the displayed tail index T to the finite fiber row F, and the seal row E is reachable only along that same route.

Dyadic and regular readback exactness then projects D and R. The radius information used at the scheduled tail is precisely the finite dyadic ledger, and the regular-rational data are precisely the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback for that fiber.

Non-escape is carrier inspection. Since no coordinate stores quotient equality, a host-selected limit, an infinite tail oracle, countable choice, or an ambient completeness theorem, every consumer of the packet must remain inside the displayed tail-fiber schedule and RealUp seal rows. □

Falsifiable prediction: Any accepted finite tail-fiber schedule consumer reaches the RealUp seal only after the displayed tail index, finite fiber, dyadic radius, RegSeqRat readback, hsame, Cont, Pkg, and local NameCert rows are present.

Independence witness: Compare StreamName, RegSeqRat, DyadicRatCore, RealUp, and RegularCauchyTailMeet: this packet is not bijective with any of them because it carries a schedule row binding one finite tail fiber to a RealUp seal; each sibling lacks at least the schedule-to-seal obligation or the tail-fiber ledger.

The later Lean `TasteGate.lean` boundary for this carrier must provide `Nontrivial FiniteTailFiberSchedul` and `FieldFaithful FiniteTailFiberScheduleUp` instances with explicit witness pairs and field projections. The nontrivial witness separates a scheduled tail-fiber packet from a carrier with no

seal route, while field faithfulness preserves the tail index, finite fiber, schedule row, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailFiberSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$ packet is generated from a displayed tail index, finite fiber row, schedule row, dyadic radius ledger, RegSeqRat readback, RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTailFiberSchedule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 921.1 and Definition 921.2 bind the finite tail-fiber schedule and RealUp seal handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTailFiberSchedule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient reals, host real completeness, selected limits, countable choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-to-fiber exactness, dyadic and regular readback exactness, RealUp seal handoff, transport closure, and TasteGate nontriviality with field faithfulness.

Lemma 921.3 (FiniteTailFiberSchedule seal-route determinacy)**.** *Let*

$$Q = (T, F, S, D, R, E, H, C, P, N)$$

and

$$Q' = (T', F', S', D', R', E', H', C', P', N')$$

be accepted $\mathsf{FiniteTailFiberSchedule}^{\uparrow}$ *packets. Suppose the displayed non-seal rows agree componentwise through the packet classifier:*

$$T = T', \quad F = F', \quad S = S', \quad D = D', \quad R = R', \quad H = H', \quad C = C', \quad P = P', \quad N = N'.$$

Then the $\mathsf{Real}^{\uparrow}$ *seal rows* E *and* E' *are classified along the same finite tail-fiber route. More explicitly, any accepted read of* E *and any accepted read of* E' *both factor through the common route*

$$T, F, S, D, R, H, C, P, N,$$

so the seal comparison has no independent coordinate from which to distinguish two RealUp seal routes under the same displayed tail index, finite fiber, schedule, dyadic ledger, RegSeqRat readback, transport, continuation, provenance, and local naming rows.

Proof. Project both packets to the carrier of Definition 921.1. The only row that can expose a RealUp seal is the seal coordinate attached to the same tuple as the displayed tail index, finite fiber, schedule row, dyadic ledger, RegSeqRat readback, componentwise transports, continuation routes, package provenance, and local name.

Apply the schedule-to-fiber obligation from Definition 921.2. In each accepted packet, a seal-facing schedule read must first pass through the displayed tail index and finite fiber row. Since $T = T'$, $F = F'$, and $S = S'$, the two packets present one scheduled tail-fiber route before either seal row can be cited.

Apply the dyadic and regular readback obligation from the same theorem. A seal-facing radius read is the displayed dyadic ledger D, and the regular-rational read is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row R. Since $D = D'$ and $R = R'$, the readback portion of the two seal routes is the same finite ledger rather than two separate regular-rational sources.

The transport and continuation rows now identify the remaining route data. The common rows $H = H'$ and $C = C'$ replay the same componentwise hsame transport and the same displayed Cont

route through the stream-name readback, dyadic-radius use, regular-rational readback, tail-meet comparison, and seal consumption. The common rows $P = P'$ and $N = N'$ attach the same package provenance and local naming surface to that replay.

Finally use non-escape from Definition 921.2. The packet has no coordinate for a quotient real, selected host limit, infinite tail oracle, countable-choice row, ambient completeness theorem, or second seal selector. Therefore an alleged distinct seal route would need an unlisted coordinate, while every accepted seal read is already forced through the common finite route $T, F, S, D, R, H, C, P, N$. Hence E and E' classify together along that route. $\square$

Chapter 922

A Concrete Naming Certificate for MetaCICTypedExamplesApplicationRoute$^{\uparrow}$

The MetaCICTypedExamplesApplicationRoute$^{\uparrow}$ packet realizes the application and beta-redex rows of chapter 9382 as a finite route certificate. It sits beside the MetaCIC gallery of chapter 775, the worked-example seal of chapter 753, the typed-example gallery of chapter 756, and the audit budget of chapter 880: those siblings expose gallery entries, checker rows, seal rows, and regression-spend rows, while this chapter isolates the route by which application-form examples and identity redex examples are carried to downstream MetaCIC audit consumers.

Definition 922.1 (MetaCIC typed-examples application-route carrier)**.** *A* MetaCICTypedExamplesApplicationRoute *carrier is a finite* BHist *packet*

$$R = (S, T, A, B, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the typed source-term row, recording the displayed function, argument, context, and sort information for application-form examples such as* app_id_sort_in_empty_ctx *and dependent identity application rows;*
2. *T is the typed target-term row, recording the displayed result term and its type after application or beta replay;*
3. *A is the application spine row, listing the finite head, argument, binder, context, and instantiation packets consumed by the route;*
4. *B is the beta and checker row, recording the selected redex endpoint, local checker acceptance, and named beta replay for rows such as* id_sort_redex *and* id_sort_redex_typed*;*
5. *Z is the closedness row, recording the finite closed-term, closed-context, and well-scoped inputs needed by S, T, A, B;*
6. *H records componentwise* hsame *transports for source terms, target terms, application spines, beta and checker rows, closedness rows, provenance, and local naming;*
7. *C records the displayed* Cont *routes by which a downstream consumer reaches the source row, target row, application spine, beta or checker row, closedness row, and sibling handoff rows;*
8. *P is the* Pkg *provenance over chapter 9382, chapter 775, chapter 753, chapter 756, and chapter 880;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary subject reduction, strong normalization, confluence, global checker completeness, unrestricted beta closure, or a host evaluator.

Theorem 922.2 (MetaCIC typed-examples application-route NameCert obligations)**.** *For the carrier of Definition 922.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for $R = (S, T, A, B, Z, H, C, P, N)$;*
2. *source-target exposure through S and T for typed application examples and typed beta-redex endpoints named by chapter 9382;*
3. *application-spine exactness through A, including the displayed head, argument, binder, context, and instantiation packets;*
4. *beta, checker, and closedness visibility through B and Z for exactly the selected application and identity-redex rows;*
5. *transport, continuation, provenance, and local naming through H, C, P, N, with sibling handoff restricted to the four concrete TypedExamples packets cited above.*

Proof. Project the finite packet. The source row S and target row T expose the endpoints of the application route, while the spine row A records the finite head, argument, binder, context, and instantiation data that make the route inspectable.

The beta and checker row B is read only together with the closedness row Z. Thus a consumer can replay the selected identity-redex and application examples, but it cannot replace them by an arbitrary represented term or by an unrestricted beta closure.

Finally route the read through H, C, P, N. The transport and continuation rows keep the same displayed source, target, spine, beta, checker, and closedness coordinates, and the provenance row limits sibling access to the existing gallery, seal, typed-gallery, and audit-budget packets. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 922.3 (MetaCIC typed-examples application-route soundness)**.** *Every consumer of an accepted* $\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}$ *packet reaches a typed application or beta-redex example through the displayed rows*

$$S, T, A, B, Z, H, C, P, N.$$

The consumer may use the packet as a finite route from a typed source term through an application spine to a typed target term, with beta, checker, closedness, transport, continuation, provenance, and local naming evidence. It receives no theorem about all MetaCIC applications, no unrestricted subject-reduction principle, no normalization endpoint selector, no confluence principle, and no global checker-completeness witness.

Proof. Begin with an accepted route packet. The only public endpoints are the typed source row S and the typed target row T, and the only path between them is the finite application spine A together with the beta and checker row B.

Closedness is supplied by Z, so the checker and beta reads are tied to the displayed source, target, and spine rows rather than to an ambient term space. Componentwise transport through H preserves those same rows before any comparison is made.

The continuation row C supplies the consumer path, while P and N record provenance and local naming. Because the carrier contains no coordinate for arbitrary applications, unrestricted

beta closure, subject reduction, normalization, confluence, or checker completeness, every accepted consumer read remains the finite application route described in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}}$ ·

Constructive story: A $\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}$ packet is generated from finite BHist rows for typed source terms, typed target terms, application spines, beta and checker rows, closedness rows, transports, continuations, provenance, and local naming. The route to a typed application or beta-redex example is replayed by Cont through the displayed source, spine, beta/checker, closedness, and target rows. Componentwise hsame preserves the same term, spine, checker, and closedness inventory, while Pkg records the typed-examples audit map and sibling gallery, seal, typed-gallery, and audit-budget packets. The packet certifies selected examples only.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the MetaCIC typed-examples application-route carrier, NameCert obligations, route soundness theorem, and finite consumer path.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedExamplesApplicationRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim arbitrary applications, unrestricted beta closure, subject reduction, normalization, confluence, host evaluation, or checker completeness.

Upgrade path: Obligation closure requires Lean encodings of the application-route carrier, source-target exposure, beta/checker route, closedness row, and soundness theorem.

Chapter 923

A Concrete Naming Certificate for $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$

The $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ packet realizes the TypedExamples audit-map vision of chapter 9382 as a finite coverage matrix. It sits beside the MetaCIC gallery of chapter 775, the typed-example gallery of chapter 756, the worked-example seal of chapter 753, the regression-spend budget of chapter 880, and the beta-step boundary of chapter 754. Those siblings expose gallery rows, checker rows, seal rows, budget rows, and beta-boundary rows; this chapter binds them as one finite row-indexed coverage object for checker-audit consumers.

Definition 923.1 (Typed example coverage matrix carrier)**.** *A* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, G, C, B, R, E, H, K, P, N)$$

with the following displayed rows:

1. *F is the finite family index for accepted typed-example families;*
2. *G records, for each row of F, the paired gallery cells inherited from the MetaCIC and typed-example gallery packets;*
3. *C records the checker cells attached to the same rows;*
4. *B records the beta-boundary cells attached to the same rows;*
5. *R records the regression-spend cells and their accepted coverage budget;*
6. *E is the exactness row asserting that every accepted family in F has the displayed G, C, B, R cells and no extra coverage cell;*
7. *H records componentwise* hsame *transports for the family, gallery, checker, beta, regression-spend, exactness, provenance, and local naming rows;*
8. *K records the displayed* Cont *routes by which checker-audit consumers read exactly the cells F, G, C, B, R, E;*
9. *P is the* Pkg *provenance over the sibling gallery, seal, budget, beta-boundary, transport, continuation, and exactness rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$ *row.*

The carrier has no coordinate for checker completeness, subject reduction, normalization, arbitrary beta closure, or unindexed regression spending.

Definition 923.2 (Typed example coverage matrix row family)**.** *For an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *carrier* $M = (F, G, C, B, R, E, H, K, P, N)$*, the* coverage row family *is the finite subfamily of* F *whose displayed cells name the boolean, natural-number, pair, option, list, sum, and dependent-function gallery examples. The row family is read only through the paired matrix coordinates* G, C, B, R, E*:* G *names the example cell,* C *names its checker cell,* B *names the beta-boundary cell available for that example,* R *names the regression-spend cell, and* E *records that all four cells belong to the same family row.*

Theorem 923.3 (Typed example coverage matrix NameCert obligations)**.** *For the carrier of Definition 923.1, the local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for* $M = (F, G, C, B, R, E, H, K, P, N)$*;*
2. *family-index exactness through* F *against the accepted typed-example family rows;*
3. *paired gallery, checker, beta, and regression-spend cell exposure through* G, C, B, R*;*
4. *matrix exactness through* E*, requiring each accepted row to carry all four cell kinds and no unindexed cell;*
5. *transport, continuation, provenance, and local naming through* H, K, P, N*.*

Proof. Carrier admission is projection of the displayed packet. The row F supplies the finite family index before any coverage cell is read.

The rows G, C, B, R are paired by position. A consumer can inspect a gallery cell, checker cell, beta cell, or regression-spend cell only as the cell attached to the same family row.

The exactness row E records that the four cell kinds are jointly present for every accepted family and that no extra coverage coordinate exists outside the matrix. The rows H, K, P, N then transport, route, name, and package only the displayed rows. □

Theorem 923.4 (Typed example coverage matrix regression exactness)**.** *Every regression-spend cell in an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet is paired with the unique displayed family row and with the corresponding gallery, checker, and beta cells. Conversely, every accepted family row has a displayed regression-spend cell in* R*.*

Proof. Project the accepted packet to F, G, C, B, R, E. The exactness row E is read against the common family index F, so each regression-spend entry is indexed before it is exposed to a checker-audit consumer.

For a displayed spend cell, E supplies the matching gallery, checker, and beta cells in the same row. For a displayed family row, E supplies the corresponding regression-spend cell in R.

Since the carrier has no unindexed spend coordinate, these two readings exhaust regression coverage for the packet. □

Theorem 923.5 (Typed example coverage matrix non-escape)**.** *Every checker-audit consumer of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *carrier reads coverage only through the displayed matrix rows* $F, G, C, B, R, E, H, K, P, N$*. The consumer cannot replace a missing gallery, checker, beta, or regression-spend cell by a sibling packet alone, an ambient checker theorem, or an unindexed budget row.*

Proof. Let a checker-audit consumer route be given. The route must pass through the displayed continuation row K and therefore reaches only the family, cell, exactness, transport, provenance, and local naming rows.

The sibling packets provide provenance for the cell contents, but the matrix pairing is carried by F and E. A sibling gallery, seal, budget, or beta-boundary packet alone does not store the combined cell row.

No coordinate in the carrier names an ambient checker theorem or an unindexed budget row. Hence the accepted consumer surface is exactly the finite coverage matrix. □

Theorem 923.6 (Typed example coverage matrix spend boundary)**.** *If a checker-audit consumer spends a regression budget from an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet, the spend is licensed only for the family row whose gallery, checker, beta, and regression-spend cells are all displayed in the same matrix row.*

Proof. A regression spend is read from R through the continuation route K. The exactness row E identifies the family row in F and the paired gallery, checker, and beta cells in G, C, B.

The transport row H preserves the same row identity under componentwise hsame movement, while P and N record provenance and local naming only. They do not create a second spend endpoint.

Therefore a spend is confined to the displayed matrix row that carries all four cell kinds. □

Theorem 923.7 (Typed example coverage matrix gallery rows)**.** *The coverage row family of Definition 923.2 contains displayed rows for the boolean, natural-number, pair, option, list, sum, and dependent-function gallery examples. For each such row, the* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet exposes the same-family gallery cell in G, checker cell in C, beta-boundary cell in B, regression-spend cell in R, and exactness witness in E.*

Proof. Project the accepted carrier to F, G, C, B, R, E. The row family is a finite subfamily of F, so each listed gallery example is named before any checker-audit cell is read.

Read E at each of the boolean, natural-number, pair, option, list, sum, and dependent-function rows. The exactness row displays the matching G, C, B, R cells at the same family index.

Because the cells are paired inside the same carrier, coverage of these examples is matrix coverage. It is not a claim that an external checker proves every term in the surrounding type system. □

Theorem 923.8 (Typed example coverage matrix boundary rows)**.** *The accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *row family does not claim total dependent coverage, polymorphic total coverage, subject reduction, normalization, or checker completeness. Any consumer that needs those properties must add a separate carrier and NameCert surface; they are not obtained from the boolean, natural-number, pair, option, list, sum, or dependent-function gallery rows alone.*

Proof. Inspect the carrier inventory of Definition 923.1. The only coverage coordinates are the finite family index F, the displayed cells G, C, B, R, and the exactness row E.

The row family of Definition 923.2 selects named gallery examples from F. It does not add a coordinate for arbitrary dependent typing, polymorphic instantiation, subject reduction, normalization, or checker completeness.

The support rows H, K, P, N transport, replay, source, and name the same matrix. They cannot manufacture a global type-system theorem outside the displayed coverage rows. □

Theorem 923.9 (Typed example coverage matrix public export)**.** *Every MetaCIC audit consumer of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet receives only the family-indexed gallery, checker, beta-boundary, and regression-spend rows* $F, G, C, B, R, E, H, K, P, N$*. The export may cite the MetaCIC typed-examples gallery of chapter 775, the typed-example audit budget of chapter 880, and the beta-step boundary of chapter 754, but it does not export checker completeness, subject reduction, normalization, arbitrary beta closure, or gallery soundness beyond the displayed family rows.*

Proof. Project the accepted matrix carrier. The public rows are the family index F, the paired gallery, checker, beta-boundary, and regression-spend cells G, C, B, R, the exactness row E, and the structural rows H, K, P, N.

Read the cited sibling chapters through provenance. The MetaCIC typed-examples gallery supplies gallery cells, the audit-budget packet supplies regression-spend cells, and the beta-step boundary supplies beta-boundary cells; the coverage matrix is the place where those cells share one family index.

Apply the boundary-row theorem above. Since the exported rows do not contain subject reduction, normalization, checker completeness, arbitrary beta closure, or gallery soundness coordinates, a MetaCIC audit consumer receives only the displayed row-indexed matrix surface. □

Theorem 923.10 (Typed example coverage matrix critical-path spend exhaustion)**.** *Any checker-audit spend from an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet must expose the same family row in* F*, gallery cell in* G*, checker cell in* C*, beta-boundary cell in* B*, regression spend in* R*, exactness row in* E*, and structural rows* H, K, P, N*. Therefore the matrix cannot be used as subject-reduction evidence, normalization evidence, or checker-completeness evidence.*

Proof. Project the accepted carrier of Definition 923.1. The only spend coordinate is R, and it is indexed by the same finite family row F as the gallery, checker, beta-boundary, and exactness cells.

Use regression exactness, the spend boundary, and checker-cell locality. These theorems keep every spend attached to the displayed G, C, B, R, E cells at one row and prevent a consumer from moving the checker cell or budget cell to a different family.

Finally apply non-escape and family exhaustion. The structural rows H, K, P, N preserve and replay the same packet, while the carrier has no coordinate for subject reduction, normalization, or checker completeness. The critical-path spend is therefore exhausted by the displayed matrix row. □

Theorem 923.11 (Typed example coverage matrix row totality)**.** *Every accepted family row in* F *has displayed gallery, checker, beta-boundary, and regression-spend cells in* G, C, B, R*, with the exactness row* E *recording that the four cells belong to the same row.*

Proof. Project the accepted carrier to F, G, C, B, R, E. A family row is accepted only as an element of the finite index F, so the row is named before any coverage cell is consumed.

Read the exactness row E at that family index. The row-totality clause in E displays the corresponding gallery cell in G, checker cell in C, beta-boundary cell in B, and regression-spend cell in R.

Because E is part of the same carrier, the four cells are not imported independently from sibling chapters. They are the displayed cells of the accepted matrix row. □

Theorem 923.12 (Typed example coverage matrix checker-cell locality)**.** *A checker cell from* C *may be consumed only at the family row whose gallery cell in* G*, beta-boundary cell in* B*, and regression-spend cell in* R *are paired with it through* E*.*

Proof. Let a checker-audit consumer read a checker cell. The read is routed through K, hence it reaches C only through the displayed matrix rows.

The exactness row E identifies the family row in F attached to that checker cell. At the same index it displays the paired gallery, beta-boundary, and regression-spend cells in G, B, R.

The transport row H preserves this row identity under componentwise hsame movement. Thus a checker cell cannot be moved to a different family row or paired with unrelated sibling data. □

Theorem 923.13 (Typed example coverage matrix family exhaustion)**.** *The accepted family index F exhausts the matrix surface of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *carrier: no ambient checker theorem, sibling gallery row, beta-boundary row, or unindexed budget row can add a hidden family outside F.*

Proof. Inspect the carrier inventory. The only family coordinate is F, and the only coverage-cell coordinates are G, C, B, R paired by E.

The provenance row P records sibling ancestry, but provenance is not a second family index. Likewise, the continuation row K only replays reads from the displayed matrix; it does not allocate another row.

Since the carrier has no coordinate for an ambient checker theorem or unindexed budget row, every accepted family is already one of the rows of F. □

Theorem 923.14 (Typed example coverage matrix carrier admission)**.** *For an accepted packet $M = (F, G, C, B, R, E, H, K, P, N)$, carrier admission is exactly the finite matrix row M together with the family index F and the displayed checker-audit rows $G, C, B, R, E, H, K, P, N$. The carrier admission row does not replace the matrix by a sibling gallery packet, a checker packet, or a budget packet.*

Proof. Project the accepted packet of Definition 923.1. The family index F is the first displayed row, so the carrier is admitted before any checker-audit consumer reads a cell.

Apply the NameCert obligations of Definition 923.3. They keep the displayed matrix together with the paired gallery, checker, beta, regression-spend, exactness, transport, continuation, provenance, and local naming rows. No sibling packet can supply the same combined row. □

Theorem 923.15 (Typed example coverage matrix context coverage)**.** *Every accepted* TypedExampleCoverageMatri *carrier covers its typed-example context exactly through the family rows in F and the paired gallery, checker, beta, and regression-spend rows in G, C, B, R. The context coverage row is therefore the displayed matrix row, not an ambient parameter-transport theorem.*

Proof. Read the row-totality theorem and the checker-cell locality theorem. Together they force each family row in F to carry all four displayed cell kinds in the same matrix row.

Then use family exhaustion. It rules out hidden family rows outside the matrix surface, so the accepted context is exhausted by the displayed F, G, C, B, R rows and their exactness row E. □

Theorem 923.16 (Typed example coverage matrix ledger non-escape)**.** *The local ledger row of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet cannot escape the matrix surface: every regression-spend read, checker read, gallery read, or beta-boundary read remains tied to the same family row and its paired exactness data. In particular, the ledger does not leave a residual unindexed budget row behind the displayed packet.*

Proof. Project the carrier and read the continuation row K. It routes every read through the displayed matrix rows, so a ledger read reaches only $F, G, C, B, R, E, H, P, N$.

Now apply regression exactness and non-escape. They keep the spend row paired with the family row and rule out replacement by a sibling packet or an unindexed budget row. The ledger therefore stays inside the accepted matrix surface. □

Theorem 923.17 (TypedExampleCoverageMatrix component route)**.** *Any checker, beta-boundary, gallery, or regression-spend consumer of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *carrier must read the gallery row, checker row, beta-step row, regression-spend row, row-totality evidence, checker-locality evidence, family-exhaustion evidence, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$ *rows as one packet.*

Proof. Project the carrier of Definition 923.1. The accepted packet already displays the family row F, the gallery row G, the checker row C, the beta-boundary row B, the regression-spend row R, the exactness row E, and the structural rows H, K, P, N.

Apply the NameCert obligations of Definition 923.3. They force a consumer to read the family index, paired cell exposure, matrix exactness, transport, continuation, provenance, and local naming as one finite matrix packet.

The regression exactness, non-escape, spend boundary, row-totality, checker-cell locality, and family-exhaustion theorems keep every checker, beta-boundary, gallery, or regression-spend read on the same accepted row. Therefore the consumer cannot split the packet into independent cells. □

Theorem 923.18 (Typed example coverage matrix TasteGate obligation surface)**.** *Before any closure promotion, the coverage-matrix obligation surface of an accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet is exactly the finite family index F, the paired gallery, checker, beta, and regression-spend cells G, C, B, R, the exactness row E, and the nonescape route through H, K, P, N. A TasteGate witness for this chapter must expose those rows as field-faithful carrier evidence and must not replace the displayed matrix by a sibling gallery, checker theorem, beta route, regression budget, or ambient audit object.*

Proof. Project the carrier $M = (F, G, C, B, R, E, H, K, P, N)$ of Definition 923.1. The family index F is named before any coverage cell can be inspected.

Use row totality Definition 923.11 and checker-cell locality Definition 923.12. They bind each gallery, checker, beta, and regression-spend cell to the same family row through E.

Finally apply family exhaustion Definition 923.13 and non-escape Definition 923.5. The remaining rows H, K, P, N transport, replay, source, and name only the displayed matrix, so a TasteGate carrier witness must be built from the same obligation surface before the closure block can record a stronger grade. □

Theorem 923.19 (Typed example coverage matrix scoped export)**.** *An accepted* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *packet exports exactly the finite row-indexed coverage surface over MetaCIC typed examples, gallery rows, checker rows, seal rows, regression-spend rows, beta-boundary rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$*. The export is the scoped matrix surface $F, G, C, B, R, E, H, K, P, N$; it is not checker completeness, subject reduction, normalization, arbitrary beta closure, arbitrary gallery soundness, or an ambient MetaCIC audit theorem.*

Proof. Project the carrier of Definition 923.1. The row family of Definition 923.2 and the context coverage theorem of Definition 923.15 identify the exported index as the finite family rows in F together with their paired G, C, B, R, E cells.

Use ledger non-escape Definition 923.16. It keeps checker, gallery, beta-boundary, seal, and regression-spend reads tied to the same matrix row, so the export cannot acquire a residual budget, sibling gallery, or ambient checker route outside F.

Finally apply the component route of Definition 923.17 and the TasteGate obligation surface of Definition 923.18. These rows preserve the same componentwise hsame, displayed Cont, Pkg, and local NameCert support, and therefore the scoped export is exactly the finite matrix surface stated above. □

Theorem 923.20 (Typed example coverage matrix public certificate)**.** *The public checker-audit certificate exported by* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *is exactly the finite row-indexed surface*

$$F, G, C, B, R, E, H, K, P, N.$$

Any consumer citing this certificate receives family-index exactness through F, E*, paired gallery, checker, beta-boundary, and regression cells through* G, C, B, R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMa}}$ *row through* N*. The certificate exports no subject-reduction theorem, normalization theorem, checker-completeness theorem, arbitrary beta closure, arbitrary gallery soundness, hidden family row, or unindexed regression budget.*

Proof. Start with the NameCert obligations of Definition 923.3. They identify carrier admission, family-index exactness, paired cell exposure, matrix exactness, transport, replay, provenance, and local naming as the only certificate rows.

Use row totality and checker-cell locality, Definition 923.11 and Definition 923.12. These bind every accepted family row to its gallery, checker, beta-boundary, and regression cells, so the public consumer receives the paired matrix row rather than detached sibling cells.

Apply non-escape and critical-path spend exhaustion, Definition 923.5 and Definition 923.10. They keep gallery, checker, beta-boundary, and regression-spend reads tied to the same family row and rule out subject reduction, normalization, checker completeness, and unindexed budget use. The public certificate is therefore exactly the displayed surface $F, G, C, B, R, E, H, K, P, N$. □

Theorem 923.21 (Typed example coverage matrix formal target row)**.** *The next formal target surface for* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *consists of a* Nontrivial *witness pair, a* FieldFaithful *mk-field projection, and TasteGate single-carrier alignment over the displayed rows*

$$F, G, C, B, R, E, H, K, P, N.$$

This target surface is confined to the already displayed coverage-matrix carrier, row-totality evidence, checker-cell locality, family exhaustion, TasteGate obligation surface, and public certificate. It does not include checker completeness, subject reduction, normalization, arbitrary beta closure, or gallery soundness.

Proof. Use the carrier of Definition 923.1 to supply the witness pair and mk-field projections from the displayed rows $F, G, C, B, R, E, H, K, P, N$.

Apply row totality, checker-cell locality, and family exhaustion, Definition 923.11, Definition 923.12, and Definition 923.13. These keep every gallery, checker, beta-boundary, and regression-spend cell tied to the same finite family row.

Finally apply the TasteGate obligation surface and the public certificate, Definition 923.18 and Definition 923.20. They identify the single-carrier alignment surface and exclude checker completeness, subject reduction, normalization, arbitrary beta closure, and gallery soundness from the target. □

Theorem 923.22 (Typed example coverage matrix Lean handoff contract)**.** *Any Lean namespace accepting the formal target row for* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *must expose* Nontrivial *and* FieldFaithful *instances over the displayed matrix rows*

$$F, G, C, B, R, E, H, K, P, N.$$

The namespace may not replace the matrix by the gallery sibling, the beta-boundary sibling, the regression-budget sibling, or an ambient audit theorem; the instances must preserve the row-indexed coverage carrier itself.

Proof. Read the formal target row of Definition 923.21. It names the witness pair, mk-field projection, and TasteGate alignment over exactly $F, G, C, B, R, E, H, K, P, N$.

Use the public certificate of Definition 923.20. That certificate is the finite matrix surface and excludes checker completeness, subject reduction, normalization, arbitrary beta closure, arbitrary gallery soundness, hidden family rows, and unindexed regression budgets.

Therefore a future namespace can discharge the target only by exposing the nontriviality and field-faithfulness witnesses for the displayed matrix rows themselves. A sibling gallery or ambient audit theorem would omit the shared row index F or the paired cells G, C, B, R, E, so it cannot serve as the handoff contract. □

Falsifiable prediction: If the coverage-matrix row is removed, a checker-audit consumer can no longer verify that every accepted typed-example family has paired gallery, checker, beta, and regression-spend cells.

Independence witness: Nearest siblings: metacictypedexamplesgallery, typedexamplegallery, metacic-typedexamples-seal, typedexampleauditbudget, betastepboundary. None carries the combined matrix-cell carrier because each sibling exposes one surface family, while the coverage matrix stores the shared family index and the paired gallery, checker, beta, and regression-spend cells in one packet.

The Lean follow-up for namespace BEDC.Derived.TypedExampleCoverageMatrixUp.TasteGate must include Nontrivial $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ and FieldFaithful $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ instances with an explicit witness pair and mk-field projections.

Theorem 923.23 (Typed example coverage matrix scope closure record)**.** *The scoped paper surface of* $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ *is the finite matrix export recorded in Definition 923.19, with its obligation rows supplied by carrier admission, family-index exactness, paired gallery-checker-beta-spend exposure, matrix exactness, row-totality, checker-cell locality, family exhaustion, the TasteGate obligation surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$*. This record closes only that scoped matrix surface.*

Closure status for $\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TypedExampleCoverageMatrix}^{\uparrow}$ packet is generated from a finite typed-example family index, paired gallery cells, checker cells, beta cells, regression-spend cells, exactness rows, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedExampleCoverageMatrix}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 923.1, Definition 923.2, Definition 923.3, Definition 923.4, Definition 923.5, Definition 923.6, Definition 923.7, Definition 923.8, Definition 923.11, Definition 923.12, Definition 923.13, Definition 923.18, Definition 923.19, Definition 923.20, Definition 923.22, and Definition 923.23 enumerate the finite coverage-matrix obligation surface for gallery, checker, beta, regression-spend, row-family, boundary, row-totality, checker-locality, family-exhaustion, TasteGate evidence, scoped export, public certificate, and formal-target handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{TypedExampleCoverageMatrix}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close checker completeness, subject reduction, normalization, arbitrary beta closure, arbitrary gallery soundness, Lean verification, or bridge checking.

Upgrade path: The next paper step is a public export or bridge route for the scoped matrix surface, while Lean verification remains a separate target.

Chapter 924

A Concrete Naming Certificate for ChurchEliminatorRegressionMatrix$^{\uparrow}$

The ChurchEliminatorRegressionMatrix$^{\uparrow}$ packet realizes the Church eliminator rows of chapter 9382 as a finite checker and beta-regression matrix. It sits beside the MetaCIC gallery of chapter 775, the coverage matrix of chapter 923, the audit budget of chapter 880, the application route of chapter 922, and the beta-step boundary of chapter 754. Those siblings expose gallery, coverage, budget, application, and beta-boundary rows; this chapter binds the Church boolean, natural-number, list, and sum eliminators as one row-indexed regression object.

Definition 924.1 (Church eliminator regression matrix carrier)**.** *A* ChurchEliminatorRegressionMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, G, L, C, B, S, X, H, K, P, N)$$

with the following displayed rows:

1. *F is the finite family row with exactly the Church boolean, natural-number, list, and sum eliminator families;*
2. *G records the gallery entry for the same family row, including the displayed type former, constructors, and eliminator term;*
3. *L records the local eliminator shape, respectively boolean branch selection, natural-number recursor, list fold, and sum case analysis;*
4. *C records the checker acceptance row for the displayed eliminator;*
5. *B records the selected beta-regression row for the same eliminator family;*
6. *S records the regression-spend row licensed for that family;*
7. *X is the exactness row asserting that every accepted family in F has exactly the displayed G, L, C, B, S cells and no extra eliminator family;*
8. *H records componentwise* hsame *transports for family, gallery, eliminator, checker, beta, spend, exactness, continuation, provenance, and naming rows;*
9. *K records the displayed* Cont *routes by which eliminator-regression consumers read the same row;*

10. *P is the* Pkg *provenance over the TypedExamples gallery, coverage matrix, audit budget, application route, beta-step boundary, transport, continuation, and exactness rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChurchEliminatorRegressionMatrix}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary eliminator soundness, global normalization, subject reduction, confluence, unbounded beta closure, or a regression spend not indexed by one of the four displayed Church families.

Lean4 ✓ `BEDC.Derived.ChurchEliminatorRegressionMatrixUp.ChurchEliminatorRegressionMatrixTasteGate single carrier alignment`

Theorem 924.2 (Church eliminator regression matrix NameCert obligations)**.** *For the carrier of Definition 924.1, the local* $\mathsf{NameCert}_{\mathsf{ChurchEliminatorRegressionMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (F, G, L, C, B, S, X, H, K, P, N),$$

family exposure through the four displayed Church eliminator rows, gallery exposure through G, eliminator-shape exposure through L, checker exposure through C, beta-regression exposure through B, spend exposure through S, row exactness through X, and transport, continuation, provenance, and local naming through H, K, P, N.

Proof. Project the displayed carrier. The family row F is read before any eliminator cell is available, and it lists only the boolean, natural-number, list, and sum eliminator families.

For each displayed family, the rows G, L, C, B, S are paired by the same row index. Thus a consumer reaches the gallery entry, eliminator shape, checker acceptance, beta-regression cell, and spend cell only as cells of the same Church eliminator row.

The exactness row X exhausts the local NameCert surface. The rows H, K, P, N preserve, route, package, and name the same finite packet, while no carrier coordinate supplies arbitrary eliminator metatheory or an unindexed spend. □

Theorem 924.3 (Church eliminator regression matrix family exactness)**.** *Every accepted* ChurchEliminatorRegressic *packet has exactly four eliminator families: Church booleans, Church natural numbers, Church lists, and Church sums. Each family row carries its displayed gallery, eliminator-shape, checker, beta-regression, and spend cells, and every displayed checker or beta-regression cell belongs to one of those four family rows.*

Lean4 ✓ `BEDC.Derived.ChurchEliminatorRegressionMatrixUp.ChurchEliminatorRegressionMatrixCarrier family exactness`

Proof. Read the family row F and the exactness row X. By construction, F has the four displayed Church eliminator families and no fifth family coordinate.

The exactness row X pairs each family with its G, L, C, B, S cells. Hence a checker cell or beta-regression cell cannot be read before choosing one of the displayed Church eliminator families.

Conversely, each listed family receives the paired gallery, eliminator-shape, checker, beta-regression, and spend cells from X. This gives both inclusion directions for the finite family inventory. □

Theorem 924.4 (Church eliminator regression matrix non-escape)**.** *Every eliminator-regression consumer of an accepted* $\mathsf{ChurchEliminatorRegressionMatrix}^{\uparrow}$ *carrier reads the Church boolean, natural-number, list, and sum eliminator rows only through*

$$F, G, L, C, B, S, X, H, K, P, N.$$

The consumer cannot replace a missing family, checker, beta-regression, or spend cell by the general coverage matrix alone, by an ambient checker theorem, by an application-route cell, or by an unindexed beta-boundary row.

Proof. Let an eliminator-regression consumer be given. Its public read passes through the continuation row K, so it reaches only the displayed family, gallery, eliminator, checker, beta, spend, exactness, transport, provenance, and local naming rows.

The sibling packets provide provenance for the cells, but the row-indexed eliminator pairing is carried by F and X. A general coverage matrix, application route, audit budget, or beta-step boundary does not by itself name the combined Church eliminator regression row.

No coordinate in the carrier names an ambient checker theorem, an arbitrary beta-boundary cell, or an unindexed spend. Therefore the accepted consumer surface is exactly the finite Church eliminator regression matrix. □

Falsifiable prediction: An eliminator regression consumer that accepts a Church family without the displayed family, checker, beta-regression, and spend rows refutes the packet.

Independence witness: Nearest siblings: metacictypedexamplesgallery, typedexamplecoveragematrix, typedexampleauditbudget, metacictypedexamplesapplicationroute, betastepboundary. None is bijective to ChurchEliminatorRegressionMatrixUp because each sibling omits at least one row among the four-family eliminator index, checker cells, beta-regression cells, spend cells, and local exactness row.

The Lean follow-up for namespace BEDC.Derived.ChurchEliminatorRegressionMatrixUp.TasteGate must include Nontrivial ChurchEliminatorRegressionMatrix$^{\uparrow}$ and FieldFaithful ChurchEliminatorRegressionMatrix$^{\uparrow}$ instances with an explicit witness pair and mk-field projections.

Theorem 924.5 (Church eliminator regression matrix obligation surface)**.** *Every accepted* ChurchEliminatorRegress *packet must expose the finite obligation rows for the eliminator source, regression matrix entries, closed checker route, transport, replay, provenance, and local NameCert:*

$$F, G, L, C, B, S, X, H, K, P, N.$$

The accepted surface has no coordinate for evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. Project the carrier of Definition 924.1. The rows F, G, L expose the eliminator source and its displayed shape, while C, B, S, X expose the closed checker route, beta-regression cells, spend cells, and row exactness.

Apply the NameCert obligations, family exactness, and non-escape theorems of Definition 924.2, Definition 924.3, and Definition 924.4. Together they force each regression matrix entry to remain paired with one of the four Church eliminator families and with its checker and beta-regression rows.

The rows H, K, P, N supply transport, replay, provenance, and local naming for the same finite packet. Since evaluator equality, quotient term equality, unrestricted normalization, and kernel consistency are not carrier fields, no consumer can obtain them from the obligation surface. □

Closure status for NameCert$_{\text{ChurchEliminatorRegressionMatrix}^{\uparrow}}$.

Constructive story: A ChurchEliminatorRegressionMatrix$^{\uparrow}$ packet is generated from four Church eliminator family rows, paired gallery cells, eliminator-shape cells, checker cells, beta-regression cells, spend cells, exactness rows, componentwise transports, continuation routes, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{ChurchEliminatorRegressionMatrix}^{\uparrow}}$, seed)

Scope closed: Definition 924.1, Definition 924.2, Definition 924.3, and Definition 924.4 bind the four-family Church eliminator regression matrix.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChurchEliminatorRegressionMatrix}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close arbitrary eliminator soundness, subject reduction, confluence, global normalization, unrestricted beta closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four-family exactness, paired checker and beta-regression exposure, spend confinement, componentwise transport, displayed continuation, provenance, and local naming rows.

Chapter 925

A Concrete Naming Certificate for $\mathsf{LayeredRelationCert}^{\uparrow}$

The $\mathsf{LayeredRelationCert}^{\uparrow}$ packet realizes the relation-certificate discipline of chapter 9416 as a finite $\mathsf{NameCert}$ surface. It consumes the local inscription row discipline of chapter 519, the gap ledger surface of chapter 520, the quotient-soundness refusal boundary of chapter 634, and the audited generator route of chapter 525. The packet does not assert that two objects are identical; it names the exact rows through which a same-schema relation may be consumed.

Definition 925.1 (Layered relation NameCert row carrier)**.** *A* $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier is a finite packet*

$$R = (A, B, L, U, V, W, F, H, C, P, G, N),$$

where A and B are the two displayed source-chain rows, L is the finite two-chain layer map, U is the same-schema exactness row, V is the preserved-structure row, W is the not-preserved row, F is the failure-boundary row, H is componentwise hsame *transport, C is displayed continuation replay, P is package provenance, G is the gap and quotient-soundness boundary row, and N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for object identity between the two source domains, empirical law transfer, quotient-stream equality, subject-substance transfer, global-frame transfer, or an unbounded analogy principle.

Theorem 925.2 (Layered relation NameCert row obligations)**.** *For an accepted* $\mathsf{LayeredRelationCert}^{\uparrow}$ *packet, the local* $\mathsf{NameCert}$ *surface consists exactly of source-chain admission through A, B, finite layer-map exposure through L, same-schema exactness through U, preserved-structure exposure through V, explicit non-preservation through W, failure-boundary exposure through F, componentwise transport through H, replay through C, provenance through P, gap and quotient-soundness refusal through G, and the local naming row N.*

Proof. Start with the row carrier of Definition 925.1. It lists the only coordinates available to a consumer, so same-schema evidence must read the two source chains, the finite layer map, the exactness row, the preserved row, and the not-preserved row from the same packet.

The inscription-side rows are anchored by chapter 519 and chapter 520. They keep the two source-chain displays and the gap ledger visible rather than allowing a bare analogy token to stand in for evidence.

The refusal side is anchored by chapter 634 and the audited generator route of chapter 525. These boundaries keep quotient soundness, generator authority, and downstream consumption attached to the displayed rows F, G, H, C, P, N. □

Lean4 ✓ BEDC.Derived.LayeredRelationCertUp.LayeredRelationCert namecert row obligations

Theorem 925.3 (Layered relation NameCert refusal)**.** *No accepted* LayeredRelationCert$^{\uparrow}$ *packet exports object identity between its two source domains, empirical transfer, quotient equality, subject-substance transfer, global-frame transfer, or a hidden bridge beyond the rows*

$$(A, B, L, U, V, W, F, H, C, P, G, N).$$

In particular, a consumer that omits either the not-preserved row W or the failure-boundary row F cannot read the packet as same-schema evidence.

Proof. Apply Definition 925.2. The row obligations place not-preserved content and the failure boundary inside the same local naming surface as the preserved and exactness rows.

The quotient-soundness boundary prevents the same-schema row U from becoming quotient equality, while the inscription-gap ledger prevents missing source rows from being filled by a hidden bridge. Hence object identity, empirical transfer, subject-substance transfer, and global-frame transfer have no carrier coordinate.

If W or F is absent, the consumer has either no explicit refusal of non-preserved content or no boundary at which the relation weakens. Such a packet is therefore not an accepted same-schema certificate. □

Theorem 925.4 (Layered relation NameCert nontriviality)**.** *The* LayeredRelationCert$^{\uparrow}$ *carrier is not reducible to the source-chain packet of inscriptionevent, the gap packet of inscriptiongapledger, the quotient-refusal packet of quotientsoundnessboundary, the generator-audit packet of authorized-generatorrecursor, or the audit-map packet of auditmapfamily. Its nontrivial surface is the simultaneous presence of a two-chain layer map together with preserved, not-preserved, and failure-boundary rows.*

Proof. Compare the displayed row sets. The inscriptionevent sibling supplies source inscription rows but not the two-chain layer map with preserved and not-preserved relation rows. The inscriptiongapledger sibling supplies the gap ledger but not same-schema exactness or the preserved-structure row.

The quotientsoundnessboundary sibling supplies a refusal boundary for quotient over-readings, while authorizedgeneratorrecursor supplies an audited generator route. Neither sibling contains both source chains together with L, U, V, W, F as one relation certificate.

The auditmapfamily sibling supplies audit-map methodology, not a local relation packet with the two-chain layer map plus preserved, not-preserved, and failure-boundary surface. Therefore the carrier is nontrivial relative to all named siblings. □

Theorem 925.5 (Layered relation NameCert field faithfulness)**.** *Every downstream consumer that reads a* LayeredRelationCert$^{\uparrow}$ *packet as same-schema evidence must preserve the field roles of $A, B, L, U, V, W, F, H, C, P, G, N$. A transported read may rename the packet only through H, C, P, N; it may not collapse A with B, read V without W, read U without F, or move the boundary row G into a source-chain coordinate.*

Proof. Use the NameCert obligations of Definition 925.2. They assign each public field its role before the local certificate can be consumed.

Next apply the refusal theorem Definition 925.3. It blocks the only collapses that would turn same-schema evidence into identity, empirical transfer, quotient equality, or hidden bridge content.

The remaining transport rows H, C, P, N only preserve, replay, package, and name the displayed packet. Since none of those rows adds a new source-chain, exactness, preserved, not-preserved, or boundary coordinate, the field roles are faithful under downstream consumption. □

Falsifiable prediction: A relation certificate that omits either the not-preserved row or the failure-boundary row cannot be consumed as same-schema evidence by any downstream bridge chapter.

Independence witness: Nearest siblings: inscriptionevent, inscriptiongapledger, quotientsoundnessboundary, authorizedgeneratorrecursor, and auditmapfamily. None is bijective to $\mathsf{LayeredRelationCert}^{\uparrow}$ because none has the two-chain layer map plus preserved, not-preserved, and failure-boundary surface in one finite NameCert packet.

Later Lean work must provide $\mathsf{Nontrivial}\ \mathsf{LayeredRelationCert}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{LayeredRelationCert}^{\uparrow}$ instances in `TasteGate.lean`.

Chapter 926

A Concrete Naming Certificate for $\mathsf{CompactNetModulusBudget}^{\uparrow}$

The $\mathsf{CompactNetModulusBudget}^{\uparrow}$ packet realizes the compact-uniform-continuity budget handoff as one finite NameCert carrier. It sits after the compact finite-net and continuous-map modulus ledger of chapter 388, consumes the tolerance-to-precision target surface of chapter 369, and records the finite rational fold supplied by chapter 619. The sibling compact cover and selector chapters chapter 648 and chapter 914 provide the compact-cover and center-selection faces, while this packet binds the finite-net, pointwise-modulus, rational-fold, and uniform-modulus budget rows as a single downstream object.

Definition 926.1 (Compact net modulus budget carrier)**.** *A* $\mathsf{CompactNetModulusBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, M, F, U, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, including the nonempty* ProbeBundle *spine and located coverage rows;*
2. *M is the pointwise-modulus row attached to the bundled centers and to the requested positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history;*
3. *F is the rational lower-bound fold row over the finite radius ledger;*
4. *U is the uniform-modulus handoff row exported to downstream compact-continuous consumers;*
5. *S is the selector and cover compatibility row tying* $\mathsf{CompactModulusCover}^{\uparrow}$ *and* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *to the same finite net;*
6. *H records componentwise* hsame *transport for finite-net, pointwise-modulus, fold, uniform-modulus, selector, route, provenance, and naming rows;*
7. *C records displayed* Cont *routes through which budget consumers read the compact net, pointwise modulus, fold ledger, selector compatibility, and uniform-modulus handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{CompactModulusCover}^{\uparrow}$, $\mathsf{CompactNetModulusSelector}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactNetModulusBudget}^{\uparrow}}$ *row.*

The carrier exports no open-cover compactness coordinate, host minimum, selected point outside the finite probe spine, external modulus function, sequential compactness argument, or quotient point equality.

Lean4 ✓ `BEDC.Derived.CompactNetModulusBudgetUp.CompactNetModulusBudgetTasteGate single carrier alignment`

Theorem 926.2 (Compact net modulus budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompactNetModulusBudget}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier admission for the packet of Definition 926.1;*
2. *finite-net exactness for the compact* ProbeBundle *row* K*;*
3. *pointwise-modulus exactness for the rows* M *over the same bundled centers;*
4. *rational-fold exactness for the lower-bound ledger* F *and its exported uniform-modulus row* U*;*
5. *downstream non-escape through the displayed* hsame*,* Cont*,* Pkg*, selector-compatibility, and local* NameCert *rows* S, H, C, P, N*.*

The later Lean `TasteGate.lean` *development for* $\mathsf{CompactNetModulusBudget}^{\uparrow}$ *must provide* Nontrivial CompactNetM *and* FieldFaithful $\mathsf{CompactNetModulusBudget}^{\uparrow}$ *instances preserving these rows.*

Proof. Carrier admission is projection from Definition 926.1. The packet exposes exactly the compact net, pointwise modulus, rational fold, uniform handoff, selector compatibility, transport, route, provenance, and local naming rows.

Finite-net exactness reads K through the compact-uniform-continuity and compact-cover sibling surfaces. The pointwise-modulus row M is attached to the same finite spine, so no pointwise row can be read at an unlisted center.

Rational-fold exactness reads F through the finite-net-minimum-fold sibling and exports U as the uniform-modulus budget row. The selector row S keeps the compact-cover and selector faces on that same finite spine.

Finally H, C, P, N transport, route, package, and name the packet. Since no excluded analytic object is a coordinate of B, the local NameCert obligations are exhausted by the five rows above. □

Lean4 ✓ `BEDC.Derived.CompactNetModulusBudgetUp.CompactNetModulusBudgetCarrier namecert obligations`

Theorem 926.3 (Compact net modulus budget handoff exactness)**.** *Every downstream compact-continuous consumer of an accepted* $\mathsf{CompactNetModulusBudget}^{\uparrow}$ *packet reads the uniform-modulus budget only through*

$$K, M, F, U, S, H, C, P, N.$$

The consumer reaches the finite-net face before the pointwise-modulus face, reaches the rational-fold row before the uniform-modulus handoff, and reads the selector and cover compatibility row on the same finite probe spine. It has no route through open-cover compactness before the rational fold row, host minima, selected external centers, sequential compactness, quotient point equality, or an external modulus function.

Proof. Project the accepted carrier. The finite-net row K, pointwise-modulus row M, rational-fold row F, uniform handoff row U, and selector-compatibility row S are the only budget-facing coordinates.

Apply Definition 926.2. The five rows force carrier admission, finite-net exactness, pointwise-modulus exactness, rational-fold exactness, and downstream non-escape through the displayed structural rows.

Read the sibling surfaces in order. The compact-uniform-continuity and compact-modulus-cover rows supply the finite-net and cover data, the finite-net-minimum-fold row supplies the rational fold, and the uniform-modulus row supplies the exported precision budget.

Because U is exported only after F and because the packet has no open-cover, host-minimum, external-center, sequential-compactness, quotient, or external-modulus coordinate, every downstream consumer route is exactly the displayed budget handoff. □

Falsifiable prediction: Every accepted $\mathsf{CompactNetModulusBudget}^{\uparrow}$ packet has no consumer route that reads an open-cover compactness coordinate before the rational fold row.

Independence witness: Compare compactuniformcontinuity, uniformmodulus, finitenetminimumfold, compactmoduluscover, compactnetmodulusselector: this carrier records all four budget faces in one packet, while each sibling lacks at least one of the finite-net, pointwise-modulus, fold-ledger, or uniform-modulus handoff rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactNetModulusBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactNetModulusBudget}^{\uparrow}$ packet is generated from compact finite-net rows, pointwise-modulus rows, a rational lower-bound fold row, a uniform-modulus handoff row, selector and cover compatibility, componentwise transport, displayed continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactNetModulusBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 926.1, Definition 926.2, and Definition 926.3 specify the finite budget packet, its local NameCert obligations, and its downstream handoff exactness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactNetModulusBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, sequential compactness, host minima, selected external centers, quotient point equality, or external modulus functions for $\mathsf{CompactNetModulusBudget}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exactness, pointwise-modulus exactness, rational-fold exactness, uniform-modulus handoff exactness, nontriviality, and field faithfulness.

Chapter 927

A Concrete Naming Certificate for $\mathsf{AttentionLedger}^{\uparrow}$

The $\mathsf{AttentionLedger}^{\uparrow}$ packet concretizes the observer-history vision of chapter 9418 as a selected-row ledger with an explicit omitted-row boundary. It consumes the observer-history carrier of chapter 477, refines the filter surface of chapter 514, and hands only selected public rows to the StreamName surface of chapter 126. The packet is a NameCert object for attention as finite disclosure: the selected rows are public, the omitted rows remain certified boundary data, and the StreamName handoff cannot erase the split.

Definition 927.1 (Attention ledger carrier)**.** *An* $\mathsf{AttentionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (H, F, S, O, U, W, R, L, C, P, N)$$

with the following displayed rows:

1. *H is the source* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *row whose observation identity is being filtered;*
2. *F is the* $\mathsf{ObserverFilter}^{\uparrow}$ *row that records the local selection decision;*
3. *S is the selected-row ledger, listing the finite* BHist *rows admitted to public attention;*
4. *O is the omitted-row ledger, listing the finite source rows not admitted to public attention together with their boundary reasons;*
5. *U is the public attention row, naming the selected surface that a downstream consumer may inspect;*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *handoff row served only from S and U;*
7. *R records componentwise* hsame *transports for the source, filter, selected, omitted, public attention, StreamName, route, provenance, and local naming rows;*
8. *L is the ledger exactness row tying every source observation either to S or to O;*
9. *C records the displayed* Cont *routes by which consumers read the source identity, filter, selected rows, omitted boundary, public attention row, and StreamName handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttentionLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed coordinates. It has no coordinate for host observer equality, quotient stream equality, hidden attention state, or deletion of an omitted source row.

Lean4 ✓ `BEDC.Derived.AttentionLedgerUp.AttentionLedgerTasteGate single carrier alignment`

Theorem 927.2 (Attention ledger NameCert obligations)**.** *For the carrier of Definition 927.1, the local* $\mathsf{NameCert}_{\mathsf{AttentionLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $A = (H, F, S, O, U, W, R, L, C, P, N)$*, observer-history source exposure through* H*, observer-filter selection through* F*, selected-row exposure through* S*, omitted-row boundary exposure through* O*, public attention exposure through* U*, StreamName handoff through* W*, componentwise* hsame *transport through* R*, ledger exactness through* L*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection of the displayed packet. The source identity row H and filter row F are present before any public attention row can be read.

The ledger rows split the source observations. The selected ledger S contains exactly the rows admitted to public attention, while the omitted ledger O contains the rows stopped at the boundary. The row L records that every source observation has one of these two ledger readings.

The public handoff is then routed through U, W, R, C, P, N. Since the carrier has no hidden attention coordinate, host observer equality, quotient stream equality, or unledgered deletion row, these displayed rows exhaust the local NameCert surface. □

Theorem 927.3 (Attention ledger non-escape)**.** *Every consumer of an accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *packet sees omitted source observations only through the omitted-row ledger* O *and its boundary reasons. No consumer route may turn an omitted row into public attention data, StreamName data, host observer equality, quotient stream equality, or hidden attention state.*

Lean4 ✓ `BEDC.Derived.AttentionLedgerUp.AttentionLedgerPacket non escape`

Proof. Project the accepted packet to $A = (H, F, S, O, U, W, R, L, C, P, N)$. The omitted observations are named only in O and are related to the source identity by the exactness row L.

Any consumer route must pass through C and therefore reads either the selected ledger S and public row U, or the omitted boundary ledger O. The StreamName row W is attached to the selected side of this split.

There is no remaining coordinate that can reinterpret an omitted entry as selected public data. Thus omitted rows remain boundary certificates and cannot escape into StreamName, host equality, quotient equality, or hidden attention state. □

Theorem 927.4 (Attention ledger ObserverFilter handoff)**.** *Every accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *packet factors its selection step through the* $\mathsf{ObserverFilter}^{\uparrow}$ *row* F*. The downstream* $\mathsf{StreamName}^{\uparrow}$ *handoff reads only the selected ledger* S*, public attention row* U*, transport row* R*, continuation row* C*, provenance row* P*, and local naming row* N*, while the omitted ledger* O *remains a boundary certificate.*

Proof. Start with the displayed filter row F. By chapter 514, a filter separates selected rows from omitted rows before any StreamName-facing read is available.

The attention packet records that separation as S and O. The row U is built from the selected side, and the StreamName row W is routed through S, U, R, C, P, N.

The omitted ledger remains visible but non-exporting. It certifies what the filter did not select, so it cannot provide a StreamName row or replace the selected public attention row. □

Theorem 927.5 (Attention ledger public boundary)**.** *An accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *consumer cannot export a StreamName-facing row unless both the selected-row ledger S and the omitted-row boundary O are present in the same carrier. The public attention row U is therefore a boundary between visible selected data and certified omitted data, not a collapse of the two ledgers.*

Proof. Assume a consumer exports the StreamName-facing row W. Carrier projection exposes W together with S, O, U, R, L, C, P, N in the same finite packet.

The route to W passes through the public attention row U, and U is licensed by the exactness row L. That exactness row accounts for source observations only by the selected ledger S and the omitted ledger O.

If either ledger is absent, the route lacks the displayed certificate separating public attention from boundary refusal. Hence an accepted export of W requires both ledgers, and the public row U remains the visible boundary rather than a hidden merger of selected and omitted data. □

Theorem 927.6 (Attention ledger obligation closure package)**.** *The carrier definition, NameCert obligations, non-escape theorem, ObserverFilter handoff, public boundary theorem, falsifiable prediction, independence witness, and TasteGate marker assemble the* obligation *surface for* $\mathsf{AttentionLedger}^{\uparrow}$*.*

The package remains over the selected-row ledger, omitted-row ledger, public attention row, StreamName handoff, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows. It does not add host observer equality, quotient stream equality, hidden attention state, or deletion of an omitted source row.*

Proof. Begin with Definition 927.1. It displays exactly $H, F, S, O, U, W, R, L, C, P, N$, including the selected-row ledger, omitted-row ledger, public attention row, StreamName handoff, transport, replay, provenance, and local naming rows.

Apply Definition 927.2, Definition 927.3, Definition 927.4, and Definition 927.5. These theorems give the local NameCert surface, keep omitted rows at the boundary, route the selection through the ObserverFilter row, and require both ledgers before a public StreamName-facing export.

The falsifiable prediction and independence witness record the same finite separation as a consumer-facing test: selected disclosure and omitted-boundary evidence must coexist, and no sibling carrier supplies the same combined surface. The displayed TasteGate marker binds that surface to the single finite carrier.

Thus the obligation package is complete at the paper surface named here. It remains confined to S, O, U, W, R, C, P, N together with the source, filter, and exactness rows H, F, L, and it has no coordinate for the excluded host or quotient objects. □

Theorem 927.7 (Attention ledger scoped kernel boundary)**.** *Every accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *consumer route is scoped exactly by the displayed selected ledger S, omitted ledger O, public attention row U, StreamName handoff W, componentwise* hsame *row R, ledger exactness row L,* Cont *row C,* Pkg *row P, and local* NameCert *row N. The route also reads the source observer-history row H and filter row F as the incoming selection surface, but it exports no host observer equality, quotient stream equality, hidden attention state, or unledgered deletion of an omitted row.*

Proof. Project the accepted carrier of Definition 927.1. The packet has exactly the rows $H, F, S, O, U, W, R, L, C, P,$ so every consumer route must be assembled from the displayed source, filter, ledger, public, transport, continuation, provenance, and local naming coordinates.

Use non-escape and the ObserverFilter handoff. By Definition 927.3, omitted rows remain confined to O and its boundary reasons. By Definition 927.4, the selected side reaches StreamName only through S, U, W, R, C, P, N after the filter row F has separated the source rows.

Now apply the public boundary and the obligation package. Definition 927.5 requires S and O together before a StreamName-facing row can be exported, while Definition 927.6 repacks the same rows as the finite NameCert surface. Thus the scoped route is exactly the displayed kernel boundary and contains no coordinate for host observer equality, quotient stream equality, hidden attention state, or unledgered omission. □

Theorem 927.8 (Attention ledger public consumer surface)**.** *For every accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *packet*

$$A = (H, F, S, O, U, W, R, L, C, P, N),$$

the complete public consumer surface is the selected ledger S, the omitted ledger O, the public attention row U, the $\mathsf{StreamName}^{\uparrow}$ *handoff row W, the exactness row L, and the support rows R, C, P, N. The source row H and filter row F feed this surface, while no public consumer receives host observer equality, quotient stream equality, hidden attention state, or an unledgered deletion row.*

Proof. Begin with the carrier and NameCert obligations. The carrier displays exactly $H, F, S, O, U, W, R, L, C, P, N$ and the local NameCert surface exposes selected rows through S, omitted boundary rows through O, public attention through U, the StreamName handoff through W, exactness through L, and support through R, C, P, N.

Use non-escape and the ObserverFilter handoff. Omitted rows remain boundary data by Definition 927.3, while Definition 927.4 routes the selected side from the filter row F to S, U, W, R, C, P, N.

The public boundary and obligation package finish the export. Theorems 927.5 and 927.6 require the selected and omitted ledgers to coexist before the StreamName-facing handoff is available, so the public surface is exactly $S, O, U, W, L, R, C, P, N$ over the incoming source and filter rows. □

Theorem 927.9 (Attention ledger ObserverInterface handoff)**.** *Every* $\mathsf{ObserverInterface}^{\uparrow}$ *consumer of an accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *packet may read the public attention row only through the* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *source row H, the* $\mathsf{ObserverFilter}^{\uparrow}$ *split row F, the selected and omitted ledger pair S, O, and the* $\mathsf{StreamName}^{\uparrow}$ *handoff row W. The omitted rows remain boundary data carried by O, L, R, C, P, N and are not public attention rows.*

Proof. Project the accepted packet. The source observer-history row H and filter row F are the only incoming observer rows, and the selected and omitted ledgers S, O are the only rows that record the filter result.

Apply the non-escape and ObserverFilter handoff theorems. Non-escape confines omitted observations to O and its boundary reasons, while the handoff theorem routes selected data toward StreamName only through S, U, W, R, C, P, N after the filter split.

Now use the public boundary, obligation closure package, and scoped kernel boundary. These theorems require S and O together, repack the same rows as the finite NameCert surface, and identify the scoped consumer route as $H, F, S, O, U, W, R, L, C, P, N$. Thus an $\mathsf{ObserverInterface}^{\uparrow}$ consumer can read public attention only through that displayed route, with omitted rows remaining certified boundary data. □

Theorem 927.10 (Attention ledger public route export)**.** *For every accepted* $\mathsf{AttentionLedger}^{\uparrow}$ *packet*

$$A = (H, F, S, O, U, W, R, L, C, P, N),$$

any public export route must pass through the selected ledger S, omitted ledger O, public attention row U, StreamName handoff row W, exactness row L, and support rows R, C, P, N. The source

row H and filter row F remain incoming rows for the route; the export cannot carry host observer equality, quotient stream equality, hidden attention state, or deletion of an omitted source row.

Lean4 ✓ `BEDC.Derived.AttentionLedgerUp.AttentionLedgerPacket public route export`

Proof. Project the carrier and NameCert obligations. They display $H, F, S, O, U, W, R, L, C, P, N$ as the complete packet and identify $S, O, U, W, L, R, C, P, N$ as the public and boundary rows available after the source and filter rows.

Use non-escape and the ObserverFilter handoff. The omitted ledger stays confined to O and its boundary reasons, while selected data reaches StreamName only through S, U, W, R, C, P, N after F has performed the split.

Apply the public boundary and scoped kernel boundary. These theorems require S and O to remain together and make the route exactly the finite kernel surface $H, F, S, O, U, W, R, L, C, P, N$.

Finally use the ObserverInterface handoff. Public observers may read attention only through the displayed route, so host observer equality, quotient stream equality, hidden attention state, and unledgered deletion have no export coordinate. □

Falsifiable prediction: An accepted AttentionLedgerUp consumer cannot export a StreamName-facing row unless the selected-row ledger and omitted-row boundary are both present.

Independence witness: observerfilter, observerhistoryidentity, streamname, interhistinvariant, observertracekernel: no sibling is bijective to $\mathsf{AttentionLedger}^{\uparrow}$ because this carrier simultaneously stores the selected ledger, omitted ledger, and public attention row. Observerfilter has the split but not the public attention row; observerhistoryidentity has source identity without selection; streamname has the exported surface without the omitted boundary; interhistinvariant has invariant readback rather than an attention split; observertracekernel has trace composition without the selected or omitted ledger pair.

The Lean TasteGate namespace BEDC.Derived.AttentionLedgerUp.TasteGate will provide Nontrivial AttentionLed and FieldFaithful $\mathsf{AttentionLedger}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{AttentionLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AttentionLedger}^{\uparrow}$ packet is generated from observer-history identity, observer-filter selection, selected-row ledger, omitted-row ledger, public attention row, StreamName handoff, componentwise transport, ledger exactness, displayed continuation, package provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AttentionLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 927.1, Definition 927.2, Definition 927.3, Definition 927.4, Definition 927.5, Definition 927.6, Definition 927.7, and

Lean4 (definition) `BEDC.Derived.AttentionLedgerUp.taste gate`

specify the finite attention-ledger packet, selected-row ledger, omitted-row boundary, public attention row, StreamName handoff, scoped kernel boundary, and TasteGate-aware obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AttentionLedger}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.AttentionLedgerUp.AttentionLedgerPacket\_public\_route\_export`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove observer identity completeness, StreamName quotient equality, hidden attention-state existence, Lean verification, or bridge checking for $\mathsf{AttentionLedger}^{\uparrow}$.

Upgrade path: No further paper-side theory upgrade remains for the finite attention-ledger packet.

Chapter 928

A Concrete Naming Certificate for $\mathsf{ObserverInterface}^{\uparrow}$

The $\mathsf{ObserverInterface}^{\uparrow}$ packet realizes the observer-history identity vision of chapter 9418 as a finite interface row between observer filtering, attention accounting, and countable stream naming. It consumes the selected and omitted rows of chapter 514, the public attention boundary of chapter 927, and the downstream handoff discipline of chapter 126. The interface is a NameCert object for licensed observation: every public read enters through selected observer-filter rows, every omitted source row remains ledgered, and no separate observer subject is available behind the history rows.

Definition 928.1 (Observer interface carrier)**.** *An* $\mathsf{ObserverInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (H, F, A, S, O, U, W, Q, R, L, C, P, N)$$

with the following displayed rows:

1. *H is the source observer-history row whose identity is read as* BHist *data;*
2. *F is the* $\mathsf{ObserverFilter}^{\uparrow}$ *row selecting public observations and recording omitted rows;*
3. *A is the* $\mathsf{AttentionLedger}^{\uparrow}$ *row that records the selected-row and omitted-row split as a public attention boundary;*
4. *S is the selected observation ledger exported by the interface;*
5. *O is the omitted-row ledger, with one boundary entry for each source row not exported;*
6. *U is the licensed-observation row served to public consumers;*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *handoff row generated only from S and U;*
8. *Q is the no-subject row recording that every exported observation is already one of the displayed history, filter, attention, selected, omitted, or handoff rows;*
9. *R records componentwise* hsame *transports for observer-history, filter, attention, selected, omitted, licensed-observation, StreamName, no-subject, route, provenance, and local naming rows;*
10. *L is the interface ledger tying each accepted read to $F, A, S, O, U, W, Q, R, C, P, N$;*
11. *C records the displayed* Cont *routes by which consumers read the source history, filter, attention ledger, selected rows, omitted boundary, licensed observation, StreamName handoff, and no-subject row;*

12. *P is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{AttentionLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverInterface}^{\uparrow}}$ *row.*

The classifier compares accepted interface packets only through these displayed coordinates. It has no coordinate for hidden observer state, host equality of observers, quotient stream identity, unledgered deletion, or a subject parameter outside the history rows.

Theorem 928.2 (ObserverInterface NameCert obligations)**.** *For the carrier of Definition 928.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverInterface}^{\uparrow}}$ *surface consists exactly of carrier admission for $I = (H, F, A, S, O, U, W, Q, R, L, C$ observer-history source exposure through H, filter selection through F, attention-ledger exposure through A, selected-row exposure through S, omitted-row boundary exposure through O, licensed observation through U, StreamName handoff through W, no-subject exactness through Q, componentwise* hsame *transport through R, interface ledger exactness through L, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection of the displayed packet. The rows $H, F, A, S, O, U, W, Q, R, L, C, P, N$ list the whole observer-facing interface before any public read is available.

The filter and attention rows bind the public split. The selected ledger S and omitted ledger O are both read through F and A, so a consumer cannot obtain a licensed observation without seeing the boundary that prevents omitted rows from becoming public data.

The no-subject and naming rows close the local surface. The row Q records that exported observations are already displayed rows, while R, C, P, N provide only componentwise transport, route replay, provenance, and local naming. Since the carrier has no further coordinate, the listed clauses exhaust the local certificate surface. □

Theorem 928.3 (ObserverInterface filter-history handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet factors through the selected observer-filter ledger S, the public attention row U, the StreamName handoff row W, the no-subject row Q, the componentwise* hsame *transports R, the displayed* Cont *routes C, the interface ledger L, the package row P, and the local name N. Omitted rows in O remain boundary certificates and cannot supply StreamName data.*

Lean4 ✓ `BEDC.Derived.ObserverInterfaceUp.ObserverInterfacePacket filter history handoff`

Proof. Start with the accepted interface carrier. Its only StreamName-facing coordinate is W, and W is attached to the selected side of the filter and attention rows.

Read the handoff through the finite ledger. The rows S, U, L, R, C, P, N record the selected observation, public attention boundary, exact read entry, transport, route, provenance, and local naming needed by the downstream $\mathsf{StreamName}^{\uparrow}$ consumer.

The omitted ledger O is still visible, but only through its boundary role. Since Q records that no hidden subject or owner is exported, an omitted row has no route by which it can become StreamName data or replace the selected observation ledger. □

Theorem 928.4 (ObserverInterface non-escape)**.** *An accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet exports only the displayed observer-history, observer-filter, attention-ledger, selected-row, omitted-row, licensed-observation, StreamName handoff, no-subject,* hsame, Cont, Pkg*, ledger, and local* NameCert *rows of Definition 928.1. It exports no hidden observer state, quotient observer identity, host equality of histories, unledgered omitted row, unlicensed observation, or subject parameter outside* BHist*.*

Proof. Inspect the carrier coordinates. They are exactly $H, F, A, S, O, U, W, Q, R, L, C, P, N$, and each coordinate is a displayed history, filter, attention, selected, omitted, licensed-observation, handoff, no-subject, transport, ledger, route, provenance, or local naming row.

Use Definition 928.2 to identify the accepted certificate surface. Every public read is routed through the finite inventory and then recorded in L before it reaches the local name.

Use Definition 928.3 for StreamName-facing consumers. That handoff keeps omitted rows as boundary certificates and selected rows as the only exported observation data, so no hidden observer state, quotient identity, host equality, unledgered omission, unlicensed observation, or subject parameter has a public coordinate. □

Theorem 928.5 (ObserverInterface obligation closure package)**.** *For the carrier of Definition 928.1, the obligation-closure surface of* $\mathsf{ObserverInterface}^{\uparrow}$ *consists of carrier admission, filter selection, attention-ledger exposure, selected-row ledger, omitted-row boundary, licensed observation,* $\mathsf{StreamName}^{\uparrow}$ *handoff, no-subject exactness, componentwise transport, displayed continuation, package provenance, local naming, the falsifiable prediction boundary, and the independence witness against the observer-filter, observer-history-identity, attention-ledger, and StreamName siblings. Selected rows are the only* $\mathsf{StreamName}^{\uparrow}$*-facing data, and omitted rows remain boundary certificates.*

Proof. Start with the NameCert obligations of Definition 928.2. They expose the accepted carrier rows $H, F, A, S, O, U, W, Q, R, L, C, P, N$ and assign each public observation to the filter, attention, selected, omitted, licensed-observation, handoff, no-subject, transport, ledger, route, provenance, and local naming surface.

Use the filter-history handoff of Definition 928.3. It identifies S and U as the only selected rows that can feed W, while O remains visible only as an omitted-row boundary.

Finally apply non-escape, Definition 928.4. Because the packet has no hidden observer state, quotient observer identity, host equality of histories, unledgered omitted row, unlicensed observation, or subject parameter, the falsifiable prediction and independence witness read exactly this finite obligation surface. □

Theorem 928.6 (ObserverInterface selected-row exactness)**.** *For an accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet* $I = (H, F, A, S, O, U, W, Q, R, L, C, P, N)$*, the selected-row ledger* S *is exactly the public observation side selected by* F *and exposed through* A *and* U*. Every* $\mathsf{StreamName}^{\uparrow}$*-facing read through* W *must pass through* S, U, Q, R, L, C, P, N*, and no omitted row in* O *can be reclassified as selected data.*

Proof. Project the carrier rows of Definition 928.1. The selected ledger S is a displayed coordinate, and the licensed observation row U is the public read served from the selected side.

Use Definition 928.2 to bind S to the local NameCert surface. The same certificate surface records F as the filter row, A as the attention-ledger exposure, and L as the interface ledger for accepted reads.

Now apply Definition 928.3. It routes every StreamName-facing read through $S, U, W, Q, R, L, C, P, N$ and keeps O as boundary data. Non-escape then prevents an omitted row, hidden observer state, or unlicensed observation from supplying selected data. □

Theorem 928.7 (ObserverInterface omitted-row boundary)**.** *For an accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet* $I = (H, F, A, S, O, U, W, Q, R, L, C, P, N)$*, every row recorded in the omitted ledger* O *remains a boundary certificate witnessed by the same filter row* F *and attention row* A*. Such a row may be named by the local certificate and carried by* R, L, C, P, N*, but it has no route to the licensed-observation row* U *or to the* $\mathsf{StreamName}^{\uparrow}$ *handoff row* W*.*

Proof. Start with the NameCert obligation list of Definition 928.2. It exposes O as the omitted-row boundary and keeps it in the same finite packet as the filter, attention, selected, licensed-observation, handoff, transport, ledger, route, provenance, and naming rows.

The filter-history handoff of Definition 928.3 separates the roles of S and O. The selected ledger feeds W through U, while the omitted ledger remains visible only as a boundary certificate.

Finally use Definition 928.5 and Definition 928.4. The obligation package records selected rows as the only StreamName-facing data, and non-escape rules out unledgered deletion, hidden observer state, and unlicensed observation. Hence omitted rows stay named but cannot become public handoff data. □

Theorem 928.8 (ObserverInterface StreamName handoff totality)**.** *For an accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet* $I = (H, F, A, S, O, U, W, Q, R, L, C, P, N)$*, every licensed public observation exported by the interface has a displayed* $\mathsf{StreamName}^{\uparrow}$ *handoff row in* W*. The handoff is total for the selected ledger* S *and licensed-observation row* U*, and it is empty for omitted rows except for their boundary naming through* O, R, L, C, P, N*.*

Proof. Read the handoff coordinate from the carrier. The row W is generated only from the selected observation ledger S and the licensed-observation row U, so any public observation exported by the interface has a displayed handoff row.

Apply Definition 928.3. It states that every StreamName-facing read exported by an accepted packet factors through $S, U, W, Q, R, L, C, P, N$, while omitted rows remain boundary certificates.

Use the obligation package to close the route. By Definition 928.5, carrier admission, filter selection, attention-ledger exposure, selected-row ledgering, omitted-row boundary, licensed observation, handoff, no-subject exactness, transport, continuation, provenance, and naming are the whole obligation surface. Thus the handoff is total over licensed selected rows and has no public branch for omitted rows. □

Falsifiable prediction: Every accepted ObserverInterfaceUp packet routes StreamName-facing reads through selected observer-filter rows and cannot export an unledgered omitted row.

Independence witness: observerfilter, observerhistoryidentity, attentionledger, streamname: the interface packet has all four row families simultaneously, while each sibling lacks at least one of filter partition, no-subject history identity, attention ledger, or countable handoff rows.

The proposed Lean follow-up site is `lean4/BEDC/Derived/ObserverInterfaceUp/TasteGate.lean`. Its obligation surface includes $\mathsf{Nontrivial\ ObserverInterface}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObserverInterface}^{\uparrow}$ instances whose witnesses distinguish the observer-history source, filter row, attention ledger, selected ledger, omitted ledger, licensed-observation row, StreamName handoff, no-subject row, transports, routes, provenance, and local name.

Theorem 928.9 (ObserverInterface public export)**.** *Every public consumer of an accepted* $\mathsf{ObserverInterface}^{\uparrow}$ *packet factors through the displayed observer-history identity, observer-filter row, attention-ledger row, selected-row ledger, omitted-row boundary, licensed-observation row,* $\mathsf{StreamName}^{\uparrow}$ *handoff, componentwise transport, displayed continuation, package provenance, and local* NameCert *rows of Definition 928.1. In particular, the public interface exports selected observations only through the exact selected-row route and exports omitted observations only as boundary rows; no consumer receives a hidden observer state, an unledgered observation, or a StreamName-facing row outside this finite surface.*

Lean4 ✓ `BEDC.Derived.ObserverInterfaceUp.ObserverInterfacePacket public export`

Proof. Project the accepted packet through the carrier of Definition 928.1. The observer-history identity, filter, attention ledger, selected ledger, omitted boundary, licensed observation, StreamName handoff, transports, continuations, provenance, and local naming rows are all displayed coordinates of the same finite packet.

Apply the NameCert obligations of Definition 928.2 and the obligation package of Definition 928.5. These bind carrier admission, filter selection, attention-ledger exposure, selected-row ledgering, omitted-row boundary, licensed observation, no-subject exactness, componentwise transport, displayed continuation, package provenance, and local naming as the whole accepted obligation surface.

The selected-row side is exact by Definition 928.6, and the omitted-row side is bounded by Definition 928.7. Hence selected rows are the only rows that can feed public observation, while omitted rows remain named boundary data.

Finally use Definition 928.8 together with the filter-history handoff and non-escape theorems. Every StreamName-facing read passes through the licensed selected rows and their handoff route, and no hidden observer state, unledgered omitted row, or unlicensed observation supplies an additional public export branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverInterface}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverInterface}^{\uparrow}$ packet is generated from observer-history identity, observer-filter selection, attention-ledger accounting, selected rows, omitted-row boundary rows, licensed observation, StreamName handoff, no-subject exactness, componentwise transport, displayed continuation, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverInterface}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 928.1, Definition 928.2, Definition 928.3, Definition 928.4, Definition 928.5, Definition 928.6, Definition 928.7, Definition 928.8, and Definition 928.9 specify the finite observer-interface packet, its obligation package, selected-row exactness, omitted-row boundary, StreamName handoff totality, and public export boundary named by `BEDC.Derived.ObserverInterfaceUp.ObserverInterfacePacket_public_export`.

Lean target: `BEDC.Derived.ObserverInterfaceUp.ObserverInterfacePacket\_public\_export`

Lean4 ✓ `BEDC.Derived.ObserverInterfaceUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverInterface}^{\uparrow}}, \mathsf{axiomClean})$

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove observer-history completeness, StreamName quotient identity, hidden observer-state existence, Lean verification, or bridge checking for $\mathsf{ObserverInterface}^{\uparrow}$.

Upgrade path: No further paper-axis mature-treatment step remains; Lean-side bridge checking remains separate.

Chapter 929

A Concrete Naming Certificate for $\mathsf{CauchyRateBudget}^{\uparrow}$

The $\mathsf{CauchyRateBudget}^{\uparrow}$ packet realizes the finite convergence-rate control surface required by the Real completeness route of chapter 9238. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Real seal of chapter 38, and the observation-budget seal route of chapter 611. The packet records a displayed convergence-rate row as finite BEDC data; it is not a quotient of streams, a selected real value, a host metric convergence proof, or an ambient completeness principle.

Definition 929.1 (Cauchy rate-budget carrier)**.** *A* $\mathsf{CauchyRateBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, W, Q, D, E, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the convergence-rate row, naming the finite schedule that sends each displayed tolerance request to a bounded tail window;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read under that schedule;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the visible radius and comparison rows;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, Q, D have been displayed;*
6. *H records componentwise* hsame *transports for the rate, window, readback, tolerance, and seal rows;*
7. *C records the finite* Cont *routes for rate display, window read, readback, dyadic tolerance, seal handoff, package provenance, and local naming;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}$ *row.*

The carrier contains no hidden modulus selector, quotient stream equality, selected limit, host real equality, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetTasteGate single carrier alignment`

Theorem 929.2 (Cauchy rate-budget NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}$ *surface for Definition 929.1 consists exactly of carrier admission for* B*, convergence-rate display through* R*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window coverage through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger exactness through* D*,* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed tuple $B = (R, W, Q, D, E, H, C, P, N)$. Each obligation row named in the statement is one coordinate of the same finite BHist packet.

The rate, window, readback, tolerance, and seal clauses are read in order. The rate row supplies the finite schedule, the window row applies it to $\mathsf{StreamName}^{\uparrow}$, the readback row exposes the $\mathsf{RegSeqRat}^{\uparrow}$ observation, and the dyadic row records the visible tolerance before the Real seal is available.

The transport and route clauses use the stored H and C rows componentwise. Package provenance and local naming then repack the same finite surface through P, N, leaving no coordinate for quotient streams, selected limits, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier namecert obligations`

Theorem 929.3 (Cauchy rate-budget rate non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or observation-budget seal consumer of an accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet must read the convergence-rate row* R *before reaching the seal row* E*. The rate cannot be replaced by an unbounded modulus, a hidden tail selector, quotient stream equality, host metric convergence, or a selected external limit.*

Proof. Start with Definition 929.2. It places convergence-rate display before the finite-window, readback, tolerance, and seal rows in the local NameCert surface.

Project the carrier. The only route from the packet to the Real seal is the displayed chain R, W, Q, D, E authenticated by H, C, P, N.

The excluded objects are absent from the carrier. Since no coordinate stores an unbounded modulus, hidden tail selector, quotient stream equality, host convergence proof, or selected limit, no consumer can substitute one for the displayed rate row. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier rate nonescape`

Theorem 929.4 (Cauchy rate-budget StreamName–RegSeq handoff)**.** *For an accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet* $B = (R, W, Q, D, E, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *form a single finite handoff controlled by the rate row* R*. Any consumer that reads* Q *must first read* R *and* W*, and any later* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read must preserve that same handoff through* D, E, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier streamname regseq handoff`

Proof. Read R, W, Q from the carrier. The rate row selects the finite window schedule, and the window row is the only public source from which the readback row is obtained.

Apply the NameCert obligation surface. It lists the $\mathsf{StreamName}^{\uparrow}$ window coverage before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the dyadic tolerance ledger after it, so the readback cannot be detached from the selected window.

Finally use componentwise transport and route replay. The H, C, P, N rows preserve the same handoff when a downstream dyadic or Real consumer reads the packet. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetStreamNameRegSeqHandoff`

Theorem 929.5 (Cauchy rate-budget TasteGate boundary)**.** *The TasteGate boundary for* $\mathsf{CauchyRateBudget}^{\uparrow}$ *is the non-vacuous field separation among the rate row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Any faithful single-carrier witness must preserve the order*

$$R \to W \to Q \to D \to E$$

rather than collapsing the packet to one undifferentiated Real-completion row.

Proof. The carrier separates the five public Real-completion rows. Each has a distinct consumer role: rate display, finite window, regular-rational readback, dyadic tolerance, and Real seal.

Use rate non-escape and the StreamName–RegSeq handoff. They force the seal-facing route to expose the rate before the window, the window before readback, and the readback before dyadic tolerance.

Therefore a faithful TasteGate witness for this carrier must preserve the displayed row order. Collapsing the rows would destroy either the rate non-escape property or the StreamName–RegSeq handoff. □

Theorem 929.6 (Cauchy rate-budget public export)**.** *An accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet exports exactly the public finite route*

$$R \to W \to Q \to D \to E$$

together with H, C, P, N *to Real-completion and observation-budget consumers. Every such consumer must preserve the existing TasteGate row separation and may not read a hidden modulus selector, quotient stream equality, selected limit, host convergence proof, or ambient completeness from the packet.*

Proof. Begin with the NameCert obligations and rate non-escape, Definition 929.2 and Definition 929.3. They make the convergence-rate row R public before any StreamName, RegSeqRat, dyadic, or Real seal consumer can use the packet.

Next apply the StreamName–RegSeq handoff of Definition 929.4. It keeps W and Q attached to the same finite rate schedule, so a consumer cannot detach a readback from the displayed window.

Finally use the TasteGate boundary, Definition 929.5. It preserves the separated row order $R \to W \to Q \to D \to E$, while H, C, P, N transport, replay, source, and name that same packet. Since the carrier contains no row for a hidden modulus selector, quotient stream equality, selected limit, host convergence proof, or ambient completeness, none of those objects is exported. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier public export`

Theorem 929.7 (Cauchy rate-budget dyadic tolerance exactness)**.** *Any accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *only after the displayed* R, W, Q *route has been traversed, and only before the Real seal row* E *is reached. The ledger is not an isolated tolerance witness, a free-standing radius choice, or a seal-independent display.*

Proof. Project the carrier $B = (R, W, Q, D, E, H, C, P, N)$. The displayed coordinates place D after the rate row R, the window row W, and the readback row Q, while the Real seal row E remains downstream.

Apply the NameCert obligation surface and the StreamName–RegSeq handoff. Together they keep the finite rate schedule, the selected window, and the regular-rational readback attached to the same packet before any dyadic tolerance row is consulted.

The TasteGate boundary then rules out any shortcut to D or any seal read before D. So every accepted packet reaches the dyadic ledger only through the ordered $R \to W \to Q \to D \to E$ route. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier dyadic tolerance exactness`

Theorem 929.8 (Cauchy rate-budget Real seal routing)**.** *Any Real-facing seal consumer of an accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet reaches E only through the route $R \to W \to Q \to D \to E$. In particular, the seal is not reachable by bypassing the rate row, the finite-window row, the readback row, or the dyadic tolerance ledger.*

Proof. Start with rate non-escape. It forces every Real-facing consumer to read the convergence-rate row before the seal row can be used.

Next use the StreamName–RegSeq handoff together with the dyadic tolerance exactness theorem. The handoff keeps W and Q glued to the same displayed rate packet, and the exactness theorem places D between that handoff and the seal row E.

Composing these steps gives the only seal route $R \to W \to Q \to D \to E$. Any attempted bypass would break either the rate display order, the finite handoff, or the tolerance-ledger placement. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetRealSealRouting`

Theorem 929.9 (Cauchy rate-budget regular-tail-estimate handoff)**.** *Every accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet whose seal-facing consumer is a* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *consumer must expose the ordered route*

$$R \to W \to Q \to D \to E$$

before the tail-envelope lock of $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *is read. The combined route exports no quotient stream, selected limit, host metric convergence proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. First apply rate non-escape, Definition 929.3. It keeps the convergence-rate row R visible before any seal-facing read can use E.

Next use the StreamName–RegSeq handoff and dyadic tolerance exactness, Definition 929.4 and Definition 929.7. These place the finite window row W, readback row Q, and tolerance ledger D between R and the seal row.

Then apply Real seal routing, Definition 929.8. It identifies $R \to W \to Q \to D \to E$ as the only seal-facing path inside the accepted budget packet.

Finally hand this path to the tail-envelope lock of Definition 1487.5. Since the lock is read only after its finite envelope surface is exposed, the fused budget-to-tail route cannot export quotient streams, selected limits, host metric convergence proofs, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier regular tail estimate handoff`

Theorem 929.10 (Cauchy rate-budget scoped package)**.** *For every accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet $B = (R, W, Q, D, E, H, C, P, N)$, the displayed carrier, local* $\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}$ *obligations, rate non-escape,* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{Real}^{\uparrow}$ *seal routing, and TasteGate boundary assemble into one scoped package over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *Every consumer route through the package is confined to $R, W, Q, D, E, H, C, P, N$.*

Proof. Begin with the carrier of Definition 929.1 and the obligation surface of Definition 929.2. They display exactly the rate row, StreamName window, RegSeqRat readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows.

Use rate non-escape from Definition 929.3. It prevents any Real-facing or observation-budget consumer from replacing R by a hidden modulus, host convergence proof, quotient stream equality, or selected external limit.

Use the StreamName–RegSeq handoff and dyadic tolerance exactness, Definition 929.4 and Definition 929.7. These force the route R, W, Q, D to remain ordered before the Real seal is read.

Finally apply Real seal routing and the TasteGate boundary, Definition 929.8 and Definition 929.5. The seal-facing route is exactly $R \to W \to Q \to D \to E$, while H, C, P, N transport, replay, source, and name the same finite packet. Thus every consumer is scoped to the displayed BEDC kernel rows and cannot escape to an ambient completeness or quotient principle. □

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier scope package`

Theorem 929.11 (Cauchy rate-budget tail monotonicity)**.** *For every accepted* $\mathsf{CauchyRateBudget}^{\uparrow}$ *packet* $B = (R, W, Q, D, E, H, C, P, N)$*, tightening the displayed tail-rate row* R *inside the same finite* BHist *packet preserves the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the* $\mathsf{Real}^{\uparrow}$ *seal route. The tightened packet must still read* W, Q, D, E *through the same displayed rows, so the dyadic tolerance exactness, regular-tail-estimate handoff, and Real seal routing remain available without adding quotient streams, selected limits, host convergence, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetCarrier tail monotonicity`

Proof. Start with the scoped package of Definition 929.10. It confines every consumer of the accepted packet to the rows $R, W, Q, D, E, H, C, P, N$.

Tightening R changes only the displayed rate row inside that packet. The StreamName–RegSeq handoff theorem Definition 929.4 still forces Q to be read after the same window row W, and the dyadic tolerance exactness theorem Definition 929.7 still places D after R, W, Q.

Now apply Real seal routing and the regular-tail-estimate handoff, Definition 929.8 and Definition 929.9. They bind every later Real or tail-envelope read to the route $R \to W \to Q \to D \to E$ already recorded in the same BHist rows. Therefore a stricter displayed tail rate preserves the handoff and seal route rather than creating a new host-level modulus or a detached convergence proof. □

Theorem 929.12 (Cauchy rate-budget window concatenation)**.** *Let*

$$B_0 = (R_0, W_0, Q_0, D_0, E_0, H_0, C_0, P_0, N_0), \qquad B_1 = (R_1, W_1, Q_1, D_1, E_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{CauchyRateBudget}^{\uparrow}$ *packets. If their* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows are compatible and the first displayed window of* B_1 *begins after the regular-tail threshold exposed by* R_0, W_0, Q_0, D_0*, then the displayed dyadic tolerance ledgers concatenate to one accepted finite budget route. The concatenated route reads its* $\mathsf{Real}^{\uparrow}$ *seal only after both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger are visible, and it introduces no quotient stream, selected limit, countable choice, host convergence proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyRateBudgetUp.CauchyRateBudgetWindowConcatenation`

Proof. Project the two displayed carriers. The first packet supplies $R_0, W_0, Q_0, D_0, E_0, H_0, C_0, P_0, N_0$, and the second supplies $R_1, W_1, Q_1, D_1, E_1, H_1, C_1, P_1, N_1$; compatibility identifies the end of W_0 with the beginning discipline of W_1 inside the finite StreamName window surface.

Use the StreamName–RegSeq handoff on each packet. It forces Q_0 to be read from W_0 and Q_1 to be read from W_1, with each readback still governed by its displayed rate row.

Next apply dyadic tolerance exactness and tail monotonicity. The second window begins after the first regular-tail threshold, so the two tolerance displays can be read as one ordered ledger rather than as a detached radius choice.

Finally apply Real seal routing to the concatenated surface. The seal-facing consumer must traverse the combined route through both readbacks and the shared dyadic ledger before any $\mathsf{Real}^{\uparrow}$ seal is exposed. Since every coordinate used in the construction is projected from the two finite carriers and their H, C, P, N rows, no quotient stream, selected limit, countable choice, host convergence proof, or ambient completeness principle is added. □

Falsifiable prediction: A Real-completion seal consumer that consumes a CauchyRateBudgetUp packet cannot read a RealUp seal unless the displayed rate row, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance ledger have all been displayed.

Independence witness: Compare observationbudgetlimiter, regularcauchytailbudget, finiteprefixlimitbudget, cauchydiagonalbudget, and budgetedrealsealroute: CauchyRateBudgetUp has a first-class convergence-rate row tied to a single finite-window schedule, while those siblings respectively cap requests, tail thresholds, prefix depth, diagonal budget, or a composite seal route; no bijection preserves both the rate coordinate and the seal-readback order.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRateBudget}^{\uparrow}$ packet is generated from a displayed convergence-rate row, a finite $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 929.1, Definition 929.2, Definition 929.3, Definition 929.4, Definition 929.7, Definition 929.8, Definition 929.5, and Definition 929.10 scope the finite convergence-rate, StreamName window, RegSeqRat readback, dyadic tolerance, Real seal, TasteGate boundary, and BEDC kernel support rows for Real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove Real completeness, construct an ambient modulus, identify quotient streams, select a host real limit, close Lean verification, or provide bridge checking.

Upgrade path: Public closure requires a public consumer theorem exporting the same rate row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal route, TasteGate field separation, transport, replay, provenance, and local naming rows.

Chapter 930

A Concrete Naming Certificate for $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$

The $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ packet realizes the Real-completion route of chapter 9238 as one finite NameCert commitment. It sits downstream of the $\mathsf{Real}^{\uparrow}$ seal surface of chapter 38, the finite window surface of chapter 126, the request cap of chapter 903, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, and the sealed Cauchy-limit route of chapter 386. The packet records finite precision, budget, StreamName, RegSeqRat, RealUp seal, and non-escape rows; it is not a host completeness principle, a quotient stream, a selected limit, or a hidden equality rule.

Definition 930.1 (Finite Real route-commitment carrier)**.** *A* $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (E, B, W, Q, S, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite precision request row consumed by the route;*
2. *B is the observation-budget limiter row that admits the request before any downstream window read;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected under that budget;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after E, B, W, Q have been displayed;*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row that consumes the same sealed route as finite Cauchy-limit evidence;*
7. *H stores componentwise* hsame *transports for the precision, budget, window, readback, seal, handoff, route, provenance, and local naming rows;*
8. *C stores the displayed* Cont *routes from precision request to budget admission, from budget admission to StreamName window, from window to RegSeqRat readback, from readback to RealUp seal, and from the seal to the CauchyLimitSeal handoff;*
9. *P records* Pkg *provenance over* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealRouteCommitment}^{\uparrow}}$ *row.*

The carrier exports exactly this finite route commitment and no unbounded observation schedule, quotient stream, selected host limit, host real equality, trichotomy row, independent real carrier, or ambient completeness theorem.

Theorem 930.2 (Finite Real route-commitment NameCert obligations)**.** *For the carrier of Definition 930.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteRealRouteCommitment}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for the displayed packet* $F = (E, B, W, Q, S, L, H, C, P, N)$*;*
2. *budget admission, requiring the precision request* E *to pass through the observation-budget limiter row* B *before any StreamName window is served;*
3. *window readback, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row* W *to be the only finite window source for the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*;*
4. *seal ordering, requiring the* $\mathsf{Real}^{\uparrow}$ *seal row* S *to be read only after* E, B, W, Q *are displayed;*
5. *Cauchy-limit handoff, requiring the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L *to consume the same displayed seal route rather than a selected host limit;*
6. *route non-escape, excluding unbounded schedules, quotient streams, selected limits, host real equality, trichotomy, independent real carriers, and ambient completeness.*

The Lean `TasteGate.lean` *development for this packet must provide* `Nontrivial FiniteRealRouteCommitmentUp` *and* `FieldFaithful FiniteRealRouteCommitmentUp` *instances for this same carrier surface.*

Lean4 ✓ `BEDC.Derived.FiniteRealRouteCommitmentUp.FiniteRealRouteCommitmentTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 930.1. The precision request, budget limiter, window, readback, seal, handoff, transport, continuation, provenance, and local naming rows are all displayed coordinates.

Budget admission and window readback compose the sibling surfaces of chapter 903, chapter 126, and chapter 301. The route therefore reads E through B, then W, then Q, without exposing a second window source.

Seal ordering and Cauchy-limit handoff use the same continuation rows in C. The $\mathsf{Real}^{\uparrow}$ seal row S is available only after the finite readback, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row L consumes that same displayed seal route.

The non-escape obligation follows from the carrier inventory. None of F stores an unbounded schedule, quotient stream, selected limit, host equality, trichotomy row, independent real carrier, or completeness theorem, so no accepted read can import one. □

Theorem 930.3 (Finite Real route-commitment budget non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ *packet factors through the finite route*

$$E \to B \to W \to Q \to S \to L.$$

In particular, a consumer cannot reach the RealUp seal or the Cauchy-limit handoff without first displaying the observation budget, StreamName window, and RegSeqRat readback rows.

Proof. Start with the local obligation surface of Definition 930.2. It places the budget row before the StreamName window, the StreamName window before the RegSeqRat readback, the readback before the RealUp seal, and the seal before the CauchyLimitSeal handoff.

The displayed transport and continuation rows preserve that order. Componentwise hsame transport moves the same rows, while C replays only the route named in the carrier.

Finally apply the non-escape obligation. Since the packet has no coordinate for an unbounded schedule, quotient stream, selected host limit, host equality, trichotomy, independent real carrier, or ambient completeness, every seal-facing and Cauchy-limit-facing consumer remains inside the finite route. □

Falsifiable prediction: Removing the route-commitment row permits a RealUp seal consumer to bypass either the observation budget or the StreamName window before the RegSeqRat readback.

Independence witness: real, streamname, regseqrat, observationbudgetlimiter, cauchylimitseal: no sibling is bijective to $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ because this packet stores the route commitment that joins precision, budget, StreamName window, RegSeqRat readback, RealUp seal, and CauchyLimitSeal handoff rows in one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealRouteCommitment}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ packet is generated from finite BHist rows for a precision request, observation-budget limiter, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, transports, continuations, provenance, and local naming. The route is ordered by Cont: request, budget, window, readback, seal, and handoff must be displayed in sequence before any Real or Cauchy-limit consumer can read the packet. Componentwise hsame preserves that route, and Pkg records the observation, stream, regular-sequence, Real, and limit-seal dependencies.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRealRouteCommitment}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite Real route-commitment carrier, NameCert obligations, budget non-escape theorem, falsifiable prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRealRouteCommitment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim unbounded observation schedules, quotient streams, selected host limits, host real equality, trichotomy, independent real carriers, or ambient completeness.

Upgrade path: Obligation closure requires Lean encodings of the route-commitment carrier, budget-to-window route, seal ordering, Cauchy-limit handoff, and non-escape theorem.

Chapter 931

A Concrete Naming Certificate for $\mathsf{UnaryContMonoid}^{\uparrow}$

The $\mathsf{UnaryContMonoid}^{\uparrow}$ packet is the concrete monoid-shaped reading of the BEDC unary continuation calculus. It realizes the continuation-as-computation vision of chapter 9365 on the finite BHist surface: unary histories are the carrier, Cont composition is the operation, the empty history is the unit, hsame is the classifier, and the local ledger records only displayed continuation rows.

It is the monoid sibling of the Kleisli-style $\mathsf{ContinuationMonad}^{\uparrow}$ packet of chapter 485 and the category-shaped $\mathsf{Cat}^{\uparrow}$ packet of chapter 61. The chapter does not introduce a host monoid primitive, quotient equality, or external algebraic carrier.

Definition 931.1 (Unary Cont monoid carrier)**.** *A* $\mathsf{UnaryContMonoid}^{\uparrow}$ *carrier is a finite packet*

$$M = (U, \mu, e, H, C, P, N)$$

with the following visible rows:

1. *U is the unary carrier row: a displayed history h is admitted exactly through* $\mathsf{UnaryHistory}(h)$*.*
2. *μ is the binary continuation operation row: for unary histories h, k, r, the product $\mu(h, k) = r$ is displayed by the continuation ledger* $\mathsf{Cont}(h, k, r)$*.*
3. *e is the unit row, fixed to the empty history* e *with the displayed left and right unit continuations.*
4. *H records componentwise* hsame *transport for unary histories, continuation results, unit rows, and ledger entries.*
5. *C records the associativity and unit continuation ledgers for μ, including the two parenthesized continuation routes over triples of unary histories.*
6. *P is the* Pkg *provenance over* BHist*,* $\mathsf{UnaryHistory}$*,* Cont*,* hsame*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{ContinuationMonad}^{\uparrow}$*, and* $\mathsf{Cat}^{\uparrow}$*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$ *row for carrier admission, classifier stability, operation closure, unit and associativity exactness, and ledger non-escape.*

Thus the carrier is the tuple (U, μ, e, H, C, P, N) with $U =$ $\mathsf{UnaryHistory}$*, μ given by* Cont *composition, and $e =$* e*. Every readable algebraic row is a displayed BEDC history row.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidTasteGate single carrier alignment`

Theorem 931.2 (Unary Cont monoid NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier admission for the unary history row* U*; classifier stability under* hsame *transport; closure of* μ *under displayed* Cont *composition; left-unit, right-unit, and associativity exactness for the empty row and the continuation ledgers in* C*; and non-escape of the ledger through the provenance and local naming rows* P, N*.*

Proof. Carrier admission is the first coordinate of Definition 931.1. It accepts precisely unary histories and uses no carrier outside BHist.

Classifier stability is componentwise transport through H. The transported carrier row, operation row, unit row, and ledger row remain the same kind of displayed history data.

Closure and exactness are the continuation laws already read by the sibling packets. The unit clauses are the empty-history laws of Definition 29.5, and the associativity clause is the continuation associativity law of Definition 29.6, equivalently the monoid ledger summarized by Definition 40.22.

Finally, P and N record only the named BEDC rows. Since no coordinate names a host monoid object, quotient product, or external equality, the local ledger cannot export anything beyond the unary continuation packet. □

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier namecert obligations`

Theorem 931.3 (Unary Cont associativity)**.** *For unary histories* a, b, c *and displayed continuation results*

$$\mathsf{Cont}(a, b, ab) \quad \textit{and} \quad \mathsf{Cont}(b, c, bc),$$

the two continuation products

$$\mu(\mu(a, b), c) \quad \textit{and} \quad \mu(a, \mu(b, c))$$

are classified together by hsame *whenever both are read through the carrier of Definition 931.1. The empty history* e *is a two-sided unit for the same operation.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier cont associativity`

Proof. Read μ as displayed Cont composition. The left-associated product first consumes the ledger for $\mathsf{Cont}(a, b, ab)$ and then composes with c; the right-associated product first consumes the ledger for $\mathsf{Cont}(b, c, bc)$ and then composes from a.

Apply the core continuation associativity row cited in Definition 29.6. It compares exactly these two parenthesized displayed routes by hsame, so the two carrier results are classified together.

For the unit rows, apply Definition 29.5. The left and right products with e read back to the original unary history under the same classifier. These facts are ledger entries in C, not host monoid equalities. □

Theorem 931.4 (Unary Cont monoid operation closure)**.** *For unary histories* a, b *and a displayed continuation result* r *with* $\mathsf{Cont}(a, b, r)$*, the product row* $\mu(a, b) = r$ *remains inside the* $\mathsf{UnaryContMonoid}^{\uparrow}$ *carrier. It is transported componentwise by* hsame *and packaged by the same* Pkg *provenance and* NameCert *rows as the carrier packet.*

No quotient product, host monoid multiplication, or external carrier equality is introduced by this operation row.

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier operation closure`

Proof. Project the carrier of Definition 931.1. Its unary row admits histories through UnaryHistory, and its operation row is exactly displayed Cont composition.

Use the NameCert obligations of Definition 931.2. They include closure of μ under displayed Cont composition, componentwise classifier stability under hsame, and packaging by the provenance and local naming rows P, N.

The associativity and unit theorem Definition 931.3 supplies the operation ledger used by later algebraic reads. That ledger remains a continuation row inside the same carrier, so it adds neither a quotient product nor a host monoid multiplication nor an external equality. □

Theorem 931.5 (Unary Cont sibling dependency)**.** *The* $\mathsf{UnaryContMonoid}^{\uparrow}$ *packet is the monoid-shaped dependency sibling of* $\mathsf{ContinuationMonad}^{\uparrow}$ *and* $\mathsf{Cat}^{\uparrow}$*: it consumes the continuation unit and associativity rows named by chapter 485, agrees with the tail-composition reading used by chapter 61, and realizes the operational interpretation of chapter 9365 without adding any host monoid primitive, quotient carrier, or non-BEDC operation.*

Proof. The continuation-as-computation chapter reads Cont as the small-step route and identifies tail append as the operational composite. The carrier of Definition 931.1 stores exactly that route as μ.

The $\mathsf{ContinuationMonad}^{\uparrow}$ sibling supplies the same unit and associativity ledger for Kleisli-style two-step continuation reads. Forgetting source, middle, and target notation leaves the unary carrier, the binary continuation result, and the empty unit row.

The $\mathsf{Cat}^{\uparrow}$ sibling reads the same tails as hom-carrier composition. Forgetting object and arrow packaging leaves precisely the monoid-shaped operation on unary histories. Thus the dependency is a repackaging of existing continuation rows, with no additional algebraic primitive. □

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier sibling dependency`

Definition 931.6 (Unary Cont monoid kernel scope)**.** *The* $\mathsf{UnaryContMonoid}^{\uparrow}$ *kernel scope is the finite row package*

$$\mathsf{UnaryHistory},\quad \mathsf{Cont},\quad \mathsf{hsame},\quad \mathsf{e},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ContinuationMonad}^{\uparrow},\quad \mathsf{Cat}^{\uparrow}.$$

It binds the carrier to unary histories, the operation to displayed Cont *composition, the unit to the empty-history row, classifier transport to* hsame*, and sibling provenance to the continuation-monad and category packets.*

Lean4 (definition) `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidKernelScope`

Theorem 931.7 (Unary Cont monoid scoped grounding)**.** *For every accepted* $\mathsf{UnaryContMonoid}^{\uparrow}$ *carrier, associativity, left and right unit exactness, carrier transport, and ledger non-escape all factor through the rows of Definition 931.6. The scoped grounding cannot invoke a host monoid, a quotient product, or any carrier equality not displayed by* UnaryHistory*,* Cont*,* hsame*,* e*,* Pkg*,* NameCert*,* $\mathsf{ContinuationMonad}^{\uparrow}$*, and* $\mathsf{Cat}^{\uparrow}$*.*

Proof. Carrier admission reads unary histories through UnaryHistory and reads the operation through displayed Cont composition. The unit row is the empty history e, so the carrier has no separate algebraic object.

Associativity and unit exactness are inherited as continuation ledger rows from $\mathsf{ContinuationMonad}^{\uparrow}$ and $\mathsf{Cat}^{\uparrow}$. Transport is componentwise hsame transport of the same displayed histories and continuation results.

The provenance and local naming rows package those finite facts for consumers. Since every algebraic read has already factored through the listed kernel rows, the ledger cannot escape to host monoids, quotient products, or external carrier equality. □

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier scoped grounding`

Theorem 931.8 (Unary Cont monoid public NameCert export)**.** *The carrier of Definition 931.1, the NameCert obligations of Definition 931.2, the associativity and unit exactness of Definition 931.3, the sibling dependency of Definition 931.5, the kernel scope of Definition 931.6, and the scoped grounding of Definition 931.7 assemble a public* $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$ *export over*

$$\mathsf{UnaryHistory},\quad \mathsf{Cont},\quad \mathsf{hsame},\quad \mathsf{e},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ContinuationMonad}^{\uparrow},\quad \mathsf{Cat}^{\uparrow}.$$

The export excludes host monoids, quotient products, arbitrary carrier monoids, and external algebraic equality.

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier public namecert export`

Proof. Start with the carrier and obligation rows. The carrier of Definition 931.1 exposes unary histories, displayed continuation composition, the empty unit, transport, route, provenance, and local naming rows; Definition 931.2 states that these rows exhaust the local NameCert surface.

The algebraic rows are already internal to the continuation ledger. Associativity and the two unit laws are supplied by Definition 931.3, while the sibling dependency of Definition 931.5 records that the same rows are consumed from $\mathsf{ContinuationMonad}^{\uparrow}$ and $\mathsf{Cat}^{\uparrow}$.

The public export surface is therefore the finite kernel scope of Definition 931.6. The scoped grounding theorem Definition 931.7 says that every accepted read factors through exactly those rows.

Since no listed row names a host monoid, quotient product, arbitrary carrier monoid, or external equality relation, the public $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$ export is a BEDC continuation packet and not an imported algebraic object. □

Theorem 931.9 (Unary Cont monoid unit-bind surface)**.** *Every accepted unit and bind read of* $\mathsf{UnaryContMonoid}^{\uparrow}$ *factors through the concrete rows*

$$\mathsf{UnaryHistory},\quad \mathsf{Cont},\quad \mathsf{hsame},\quad \mathsf{e},\quad \mathsf{Pkg},\quad \mathsf{NameCert},\quad \mathsf{ContinuationMonad}^{\uparrow},\quad \mathsf{Cat}^{\uparrow}.$$

Left and right unit reads are the empty-history continuation rows, and associativity is the displayed continuation ledger. The surface is not a host monoid equality, a quotient product, or an arbitrary algebraic carrier.

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier unit bind surface`

Proof. Project the carrier of Definition 931.1. Its unit coordinate is the empty history e, its operation coordinate is displayed Cont composition, and its transport, provenance, and naming rows are H, C, P, N.

Apply the associativity and unit theorem, Definition 931.3. The left and right unit clauses are the empty-history continuation rows, while associativity is the comparison of the two displayed continuation ledgers.

Use the sibling dependency theorem, Definition 931.5. It identifies the same rows as the monoid-shaped reading of $\mathsf{ContinuationMonad}^{\uparrow}$ and $\mathsf{Cat}^{\uparrow}$, so the unit and bind surface is inherited as BEDC continuation data.

Finally cite the public NameCert export, Definition 931.8. The exported package is exactly the concrete kernel scope listed above, and no host monoid equality or external carrier can be read from it. □

Theorem 931.10 (Unary Cont monoid left and right unit exhaustion)**.** *For a concrete* UnaryHistory *carrier with multiplication read by* Cont *and unit read by* `BHist.Empty`*, every left-unit or right-unit consumer factors through one of the displayed* `cont_left_unit` *or* `cont_right_unit` *rows. The* $\mathsf{UnaryContMonoid}^{\uparrow}$ *packet has no separate abstract monoid operation row, host carrier equality, quotient product, or off-ledger unit service.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier left right unit exhaustion`

Proof. Project the carrier of Definition 931.1. Its operation row is displayed continuation composition, and its unit row is the empty history e.

Apply Definition 931.3. The two unit clauses are precisely the empty-history continuation rows: the left-unit read is `cont_left_unit`, and the right-unit read is `cont_right_unit`.

Use operation closure and scoped grounding, Definition 931.4 and Definition 931.7. They keep every algebraic read inside UnaryHistory, Cont, hsame, e, Pkg, NameCert, $\mathsf{ContinuationMonad}^{\uparrow}$, and $\mathsf{Cat}^{\uparrow}$.

The public NameCert export Definition 931.8 then excludes any host monoid or quotient carrier row. Hence the two displayed continuation unit rows exhaust all unit reads. □

Theorem 931.11 (Unary Cont monoid public formal target)**.** *The public formal target for* $\mathsf{UnaryContMonoid}^{\uparrow}$ *is exactly the carrier of Definition 931.1, the associativity ledger of Definition 931.3, the left and right unit exhaustion of Definition 931.10, the operation closure of Definition 931.4, the scoped grounding of Definition 931.7, and the public* $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$ *export of Definition 931.8. It contains no additional formal target for an abstract monoid carrier, quotient product, host operation, or external equality.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier public formal target`

Proof. Start with the carrier and associativity rows. The carrier exposes unary histories, displayed continuation composition, the empty unit, transport, provenance, and local naming; the associativity theorem supplies the parenthesized continuation ledger consumed by monoid-shaped reads.

Add the unit and operation boundaries. The left and right unit exhaustion theorem restricts all unit reads to `cont_left_unit` and `cont_right_unit`, while operation closure keeps products inside the displayed Cont row.

Finally use scoped grounding and public export. The scoped grounding theorem states that all accepted reads factor through the concrete kernel scope, and the public NameCert export packages exactly that finite surface for downstream consumers. □

Theorem 931.12 (Unary Cont monoid ContinuationMonad unit handoff)**.** *Every unit and bind read exported by* $\mathsf{UnaryContMonoid}^{\uparrow}$ *is the empty-history unit row together with the displayed* Cont *route consumed by* $\mathsf{ContinuationMonad}^{\uparrow}$*. The handoff exposes* UnaryHistory*,* Cont*,* hsame*,* e*,* Pkg*, and the local* NameCert *row, and it exposes no host monoid operation, quotient product, or off-packet equality service.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier continuationmonad unit handoff`

Proof. Read the carrier from Definition 931.1. Its unit coordinate is e, and its operation coordinate is the displayed Cont route over unary histories.

Use Definition 931.10. The two unit reads are exhausted by the empty-history continuation rows, so the $\mathsf{ContinuationMonad}^{\uparrow}$ unit side has no second source.

Then apply Definition 931.11 and Definition 931.5. These identify the same finite rows as the monoid-shaped surface consumed by $\mathsf{ContinuationMonad}^{\uparrow}$, while excluding host monoid equality and quotient multiplication. □

Theorem 931.13 (Unary Cont monoid Category root readiness)**.** *The unary carrier, displayed* Cont *composition, componentwise* hsame *transport,* Pkg *provenance, and local* NameCert *rows of* $\mathsf{UnaryContMonoid}^{\uparrow}$ *are the category-shaped composition substrate read by* $\mathsf{Cat}^{\uparrow}$*. A category-facing consumer receives only these rows and the sibling continuation surface, not a host category composition law or an external morphism equality.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier category root readiness`

Proof. Start with Definition 931.1. The carrier gives a unary history object, a displayed continuation operation, transport, provenance, and naming, all inside one BEDC packet.

The sibling dependency theorem, Definition 931.5, identifies this same surface as the one shared with $\mathsf{Cat}^{\uparrow}$. The unit exhaustion theorem keeps identity-shaped reads on e, and the public formal target keeps composition-shaped reads on Cont.

Thus a $\mathsf{Cat}^{\uparrow}$ read uses the displayed continuation substrate already packaged by $\mathsf{UnaryContMonoid}^{\uparrow}$. No host category composition, quotient morphism, or external equality row can be imported through that root. □

Theorem 931.14 (Unary Cont monoid KernelCategory consumer boundary)**.** *A* $\mathsf{KernelCategory}^{\uparrow}$ *consumer of* $\mathsf{UnaryContMonoid}^{\uparrow}$ *may use only the UnaryContMonoid continuation rows, the category hom-carrier rows named by the sibling category surface, and the local* NameCert *rows. The consumer boundary of chapter 488 cannot import host category composition, host monoid equality, quotient products, or a detached morphism-equality relation.*

Lean4 ✓ `BEDC.Derived.UnaryContMonoidUp.UnaryContMonoidCarrier kernel category consumer boundary`

Proof. First project the $\mathsf{UnaryContMonoid}^{\uparrow}$ rows through Definition 931.1. This supplies unary histories, displayed Cont composition, hsame transport, Pkg provenance, and local naming.

Next use Definition 931.11. It records the complete public surface available to downstream consumers and rules out abstract monoid carriers or quotient products.

Finally combine Definition 931.5 with Definition 931.10. The sibling surface is exactly the continuation and category handoff read by $\mathsf{KernelCategory}^{\uparrow}$, while the two unit rows remain empty-history continuation rows rather than host identities. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$.

Constructive story: A $\mathsf{UnaryContMonoid}^{\uparrow}$ packet is generated from unary histories, displayed Cont composition rows, the empty-history unit, componentwise hsame transport, Pkg provenance, and a local NameCert ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 931.1, Definition 931.2, Definition 931.3, Definition 931.5, Definition 931.8, and Definition 931.9 bind the monoid-shaped BEDC continuation packet over UnaryHistory, Cont, hsame, e, Pkg, NameCert, $\mathsf{ContinuationMonad}^{\uparrow}$, and $\mathsf{Cat}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UnaryContMonoid}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close commutativity, additive naming, host monoids, quotient products, arbitrary carrier monoids, or machine bridge checking.

Upgrade path: Formal closure requires Lean encoding of Definition 931.8; bridge checking remains separate from the mature paper closure.

Chapter 932

A Concrete Naming Certificate for $\mathsf{KernelObservationSieve}^{\uparrow}$

The $\mathsf{KernelObservationSieve}^{\uparrow}$ surface realizes the inscription-event route of chapter 9397 as a finite observation sieve. It consumes the observer-history identity packet of chapter 477, the selector-ledger packet of chapter 494, and the observer-filter split of chapter 514. The carrier records accepted rows and rejected rows together, so the public inscription-facing route cannot erase the rows that failed the sieve.

Definition 932.1 (Kernel observation sieve carrier)**.** *A* $\mathsf{KernelObservationSieve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, S, F, A, R, T, C, P, N)$$

with the following displayed rows:

1. *O is the source observer-history row supplied by* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*;*
2. *S is the selector-ledger row supplied by* $\mathsf{SelectorLedger}^{\uparrow}$*;*
3. *F is the observer-filter row that classifies source observations as accepted or rejected;*
4. *A is the accepted-row ledger, listing the finite observation rows admitted to the inscription-facing route;*
5. *R is the rejected-row ledger, listing the finite observation rows refused by the same sieve together with their boundary reasons;*
6. *T records componentwise* hsame *transport for O, S, F, A, R, P, N;*
7. *C records displayed* Cont *routes through the source history, selector ledger, filter row, accepted ledger, and rejected ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{SelectorLedger}^{\uparrow}$*,* $\mathsf{ObserverFilter}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelObservationSieve}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for a hidden observation oracle, host equality of histories, quotient stream equality, unledgered deletion, or an inscription route that forgets the rejected-row ledger.

Lean4 ✓ `BEDC.Derived.KernelObservationSieveUp.KernelObservationSieveTasteGate single carrier alignment`

Theorem 932.2 (Kernel observation sieve NameCert obligations)**.** *For the carrier of Definition 932.1, the local* $\mathsf{NameCert}_{\mathsf{KernelObservationSieve}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure: every accepted packet exposes exactly* $K = (O, S, F, A, R, T, C, P, N)$*;*
2. *selector accountability: the selector-ledger row* S *is read before an observation row can enter the accepted ledger* A*;*
3. *paired-ledger exactness: every source observation accounted for by the sieve is visible in exactly one of the accepted ledger* A *or the rejected ledger* R*;*
4. *transport and route stability:* T *and* C *move and replay only the displayed observer-history, selector, filter, accepted, rejected, provenance, and naming rows;*
5. *provenance exposure:* P *and* N *bind the sieve to* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{SelectorLedger}^{\uparrow}$*,* $\mathsf{ObserverFilter}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *data rather than to a host-side observation source.*

Lean4 ✓ `BEDC.Derived.KernelObservationSieveUp.KernelObservationSieveNameCert obligations`

Proof. Carrier exposure is projection from Definition 932.1. The observer-history, selector-ledger, filter, accepted-ledger, rejected-ledger, transport, route, provenance, and naming rows are all displayed coordinates.

Selector accountability reads S beside O and F. An observation reaches the accepted ledger only after the selector and filter rows have classified it inside the finite packet.

Paired-ledger exactness is the defining split between A and R. The accepted side can feed an inscription-facing consumer, while the rejected side remains visible as boundary data.

Transport, route stability, and provenance exposure are supplied by T, C, P, N. Since the carrier lacks a hidden observation oracle, host history equality, quotient stream equality, unledgered deletion, and any route that forgets R, these rows exhaust the local certificate surface. □

Theorem 932.3 (Kernel observation sieve non-escape)**.** *No consumer of an accepted* $\mathsf{KernelObservationSieve}^{\uparrow}$ *packet can recover an omitted observation row without passing through the rejected-row ledger* R*. A consumer may read accepted observations through* A *and displayed routes* C*, but omitted rows remain boundary data in* R *and cannot be reconstructed from* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{SelectorLedger}^{\uparrow}$*,* $\mathsf{ObserverFilter}^{\uparrow}$*, host equality, quotient stream equality, or an unrecorded inscription event.*

Proof. Project the accepted carrier $K = (O, S, F, A, R, T, C, P, N)$. The omitted observations are named only in the rejected-row ledger R, while accepted observations are named in A.

Apply Definition 932.2. Paired-ledger exactness places every accounted observation in exactly one of the two ledgers, and selector accountability requires the route through S and F before either ledger can be consumed.

The transport and route rows T and C replay this same split. The provenance row P names only the observer-history, selector-ledger, and observer-filter sibling certificates, so no hidden source remains from which a consumer could recover an omitted row without reading R. □

Theorem 932.4 (Kernel observation sieve inscription handoff)**.** *The finite observation sieve hands data to the inscription-event surface of chapter 9397 only through the accepted ledger* A*, while the rejected ledger* R *remains part of the same public certificate. Thus the inscription-facing route receives* O, S, F, A, T, C, P, N *together with the visible refusal boundary* R*, and it receives no unfiltered observation stream, hidden selector, or host-level source of omitted rows.*

Proof. Start from the vision handoff. The inscription-event route requires a finite public record surface rather than an unbounded observation stream.

Use Definition 932.2. The accepted ledger A is available only after O, S, F are displayed, and the rejected ledger R remains in the same packet as the boundary for omitted rows.

Finally apply non-escape from Definition 932.3. Since omitted rows cannot be recovered without R, the inscription-facing consumer receives the accepted ledger together with the refusal boundary and no hidden observation source. □

Falsifiable prediction: No accepted $\mathsf{KernelObservationSieve}^{\uparrow}$ consumer can recover an omitted observation row without passing through the rejected-row ledger R.

Independence witness: observerhistoryidentity, selectorledger, observerfilter, inscriptioneventrelated surface: the sieve has both accepted and rejected row ledgers, which these siblings lack as a paired carrier.

The Lean TasteGate namespace BEDC.Derived.KernelObservationSieveUp.TasteGate must provide Nontrivial $\mathsf{KernelObservationSieve}^{\uparrow}$ and FieldFaithful $\mathsf{KernelObservationSieve}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{KernelObservationSieve}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelObservationSieve}^{\uparrow}$ packet is generated from observer-history identity, selector-ledger, observer-filter, accepted-row ledger, rejected-row ledger, componentwise hsame transport, displayed continuation, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelObservationSieve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 932.1, Definition 932.2, Definition 932.3, and Definition 932.4 specify the finite accepted-and-rejected observation sieve and its inscription-facing boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelObservationSieve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, hidden observation recovery, quotient stream equality, host history equality, or inscription-event completeness.

Upgrade path: Obligation closure requires checked carrier admission, selector accountability, paired-ledger exactness, transport and route stability, non-escape, nontriviality, and field faithfulness for the same carrier.

Chapter 933

A Concrete Naming Certificate for $\mathsf{MaxRateReadGate}^{\uparrow}$

The $\mathsf{MaxRateReadGate}^{\uparrow}$ packet is the finite gate between the inter-Hist locality vision of chapter 9419 and downstream maximum-rate consumers such as $\mathsf{LorentzFrameRate}^{\uparrow}$ and $\mathsf{InterHistLocalityLedger}^{\uparrow}$. It consumes the unary-bound surface of chapter 594, but makes the read public only when the same packet also displays observer-symmetry and no-global-frame refusal rows. The gate is therefore not a new rate value, metric geometry, or global reference frame; it is a NameCert boundary that prevents detached maximum-rate reads.

Falsifiable prediction: An accepted $\mathsf{MaxRateReadGate}^{\uparrow}$ consumer that reads a $\mathsf{MaxCausalRate}^{\uparrow}$ row while omitting either observer-symmetry or no-global-frame refusal refutes the packet.

Independence witness: Nearest siblings: multihistconfig, crosshistcausal, maxcausalrate, observerlocalitycell, interhistinvariant. None is bijective to $\mathsf{MaxRateReadGate}^{\uparrow}$ because each omits either the paired gate requiring both symmetry and no-global-frame refusal or the maximum-rate consumer lock.

Definition 933.1 (MaxRateReadGate carrier)**.** *A* $\mathsf{MaxRateReadGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, R, S, F, L, H, Q, C, P, N)$$

with the following visible coordinates:

1. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row supplying the finite observer-history configuration and its no-global-sync surface;*
2. *R is an accepted* $\mathsf{MaxCausalRate}^{\uparrow}$ *row whose unary-bound surface is being read;*
3. *S is the observer-symmetry row, recording that the rate read is invariant under swapping the displayed observer-history presentations after componentwise transport;*
4. *F is the no-global-frame refusal row, recording that the read does not import a privileged observer, synchronized clock, metric tensor, Lorentz frame, or ambient real-valued speed;*
5. *L is the maximum-rate consumer lock, requiring every public consumer of R to pass through both S and F before it may read the unary-bound row;*
6. *H records componentwise* hsame *transport for $M, R, S, F, L, Q, C, P, N$;*
7. *Q records* psame *stability for the probe reads used by the rate row, symmetry row, refusal row, and consumer lock;*

8. *C records the displayed* Cont *routes from the configuration and rate rows into S, F, L, H, Q, P, N;*
9. *P is the* Pkg *provenance over* $\mathsf{MaxRateReadGate}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MaxRateReadGate}^{\uparrow}}$ *row.*

The carrier exports a finite public read gate for a maximum-rate row. It does not export a detached rate, a host-side numeric speed, a metric structure, a synchronized frame, or an all-observer reference coordinate.

Lean4 ✓ `BEDC.Derived.MaxRateReadGateUp.MaxRateReadGateTasteGate single carrier alignment`

Theorem 933.2 (MaxRateReadGate NameCert obligations)**.** *For the carrier of Definition 933.1, the local* $\mathsf{NameCert}_{\mathsf{MaxRateReadGate}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed packet $G = (M, R, S, F, L, H, Q, C, P, N)$;*
2. *rate-source locality, requiring R to be read as the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row attached to the same displayed configuration M;*
3. *observer-symmetry exactness, requiring every public rate read to pass through S after componentwise transport;*
4. *no-global-frame refusal, requiring every public rate read to pass through F before any consumer sees the unary-bound surface;*
5. *consumer-lock exactness, requiring L to bind the reads of R, S, F into one public gate rather than three independent rows;*
6. *transport stability and non-escape through H, Q, C, P, N, excluding global frames, synchronized clocks, metric tensors, Lorentz frames, host real speeds, and detached maximum-rate reads.*

Lean4 ✓ `BEDC.Derived.MaxRateReadGateUp.MaxRateReadGateNameCertObligations`

Proof. Carrier admission is the projection of the finite tuple in Definition 933.1. The rows $M, R, S, F, L, H, Q, C, P, N$ display the configuration, rate row, symmetry row, refusal row, lock, transport, probe stability, route, provenance, and local naming data.

Rate-source locality follows by reading M before R. The rate row is not a free-standing numeric surface; it is the $\mathsf{MaxCausalRate}^{\uparrow}$ row already tied to the finite witness-family packet described in chapter 594.

The public gate is then formed by S, F, L. The symmetry row supplies the observer-symmetric read, the refusal row blocks global-frame imports, and the lock row requires both gates before any consumer reads the unary-bound surface.

Transport stability is componentwise through H, Q, C. These rows replay the same configuration, rate, symmetry, refusal, and lock data under accepted hsame and psame movement.

The non-escape clause is an inventory argument. Since the carrier has no coordinate for a global frame, synchronized clock, metric tensor, Lorentz frame, host real speed, or detached rate row, no local $\mathsf{NameCert}_{\mathsf{MaxRateReadGate}^{\uparrow}}$ projection can export one. □

Theorem 933.3 (MaxRateReadGate symmetry and non-escape)**.** *Every public consumer of an accepted* $\mathsf{MaxRateReadGate}^{\uparrow}$ *carrier*

$$G = (M, R, S, F, L, H, Q, C, P, N)$$

may read the MaxCausalRate$^{\uparrow}$ *row R only through the observer-symmetry row S, the no-global-frame refusal row F, and the maximum-rate consumer lock L. Swapping the displayed observer-history presentations preserves the public read after transport through H, Q, C, and no consumer receives a global frame, synchronized clock, metric tensor, Lorentz frame, host real-valued speed, or detached rate row.*

Lean4 ✓ `BEDC.Derived.MaxRateReadGateUp.MaxRateReadGateCarrier symmetry nonescape`

Proof. Start with the accepted carrier and project M and R. The configuration row M supplies the finite observer-history surface, while R supplies the maximum causal-rate row already bounded inside that surface.

Apply the observer-symmetry row S. It records that the public rate read is invariant under swapping the displayed observer-history presentations after replaying the componentwise transport data.

Next apply the no-global-frame refusal row F. This blocks any attempt to interpret the rate read as a privileged observer frame, synchronized clock, metric tensor, Lorentz frame, or ambient numeric speed.

Finally apply the consumer lock L and route the read through H, Q, C, P, N. The lock binds R, S, F into one public gate, so a consumer that omits either symmetry or refusal leaves the MaxRateReadGate$^{\uparrow}$ surface. □

Theorem 933.4 (MaxRateReadGate downstream consumer boundary)**.** *A* LorentzFrameRate$^{\uparrow}$, InterHistLocalityLedger$^{\uparrow}$, *or other maximum-rate consumer may use an accepted* MaxRateReadGate$^{\uparrow}$ *packet only as the finite gated surface*

$$M, R, S, F, L, H, Q, C, P, N.$$

The consumer may inspect the multi-Hist configuration row, the MaxCausalRate$^{\uparrow}$ *row, the observer-symmetry row, the no-global-frame refusal row, the consumer lock, componentwise transport, probe stability, displayed continuation routes, provenance, and local naming data. It may not bypass the gate to obtain a free rate value or a global reference frame.*

Lean4 ✓ `BEDC.Derived.MaxRateReadGateUp.MaxRateReadGateCarrier downstream consumer boundary`

Proof. Read the carrier rows listed in Definition 933.1. The only maximum-rate input is R, and the only public path to R goes through S, F, L.

Use Definition 933.2 to bind those rows as the local NameCert surface. The obligations make rate-source locality, symmetry, refusal, and lock exactness part of one packet.

Then use Definition 933.3. It preserves the read under observer swap and excludes every detached global-frame or numeric-rate export.

Thus downstream consumers receive a finite gate rather than a rate primitive. Any consumer that asks for a read of R without S and F is outside the displayed packet. □

Theorem 933.5 (MaxRateReadGate CrossHistCausalRate handoff)**.** *Every* CrossHistCausalRate$^{\uparrow}$ *consumer of an accepted* MaxRateReadGate$^{\uparrow}$ *packet reads the maximum-rate row only through*

$$M, R, S, F, L, H, Q, C, P, N.$$

The row R is tied to the MaxCausalRate$^{\uparrow}$ *surface of chapter 594, S is the observer-symmetry row, F is the no-global-frame refusal row, and L is the consumer lock. The handoff to chapter 876 therefore exports a gated maximum-rate boundary rather than a detached rate value, host speed, metric row, clock, or global observer frame.*

Proof. Apply Definition 933.2. It binds the carrier rows $M, R, S, F, L, H, Q, C, P, N$ into one local NameCert surface and makes rate-source locality part of the packet.

Next apply Definition 933.3. This forces every public read of R to pass through the observer-symmetry row S, the no-global-frame refusal row F, and the lock row L.

Finally apply Definition 933.4. A $\mathsf{CrossHistCausalRate}^{\uparrow}$ consumer is one of the maximum-rate consumers that receives the finite gate, so its handoff reads the same ten rows and no detached global-frame or numeric-rate surface. □

Theorem 933.6 (MaxRateReadGate NameCert package)**.** *The carrier, local NameCert obligations, symmetry-and-non-escape theorem, and downstream consumer boundary assemble the concrete* $\mathsf{NameCert}_{\mathsf{MaxRateReadGate}^{\uparrow}}$ *package over* $\mathsf{MultiHistConfig}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* NameCert. *The package is ready for a Lean realization whose gate file provides* `Nontrivial MaxRateReadGateUp` *and* `FieldFaithful MaxRateReadGateUp` *instances in* `lean4/BEDC/Derived/MaxRateReadGateUp/TasteGate.le` *with the canonical declaration surface expected at* `BEDC.Derived.MaxRateReadGateUp.taste_gate`.

Proof. Begin with the carrier definition. It fixes the ten visible rows and names the $\mathsf{MaxCausalRate}^{\uparrow}$ source of the rate read.

Add the NameCert obligations. They require locality against M, observer symmetry through S, no-global-frame refusal through F, consumer locking through L, and transport through H, Q, C, P, N.

The symmetry-and-non-escape theorem and downstream boundary then exhaust the public consumer surface. Together they make the package finite, falsifiable, and separated from the sibling packets that supply only configuration, causal witnesses, raw maximum-rate bounds, locality cells, or invariant readback rows. □

Theorem 933.7 (MaxRateReadGate obligation coverage)**.** *The carrier, NameCert obligations, symmetry nonescape theorem, downstream consumer boundary, CrossHistCausalRate handoff, and package theorem cover the carrier admission, rate-source locality, observer-symmetry exactness, no-global-frame refusal, consumer-lock exactness, transport stability, and ledger nonescape obligations for* $\mathsf{MaxRateReadGate}^{\uparrow}$. *The same coverage is constrained by the checked TasteGate carrier alignment for* `BEDC.Derived.MaxRateReadGateUp.MaxRateReadGateTasteGate_single_carrier_alignment`.

Proof. Apply Definition 933.1 for carrier admission. The tuple $G = (M, R, S, F, L, H, Q, C, P, N)$ displays exactly the configuration row, maximum-rate row, observer-symmetry row, no-global-frame refusal, consumer lock, transport, probe stability, continuation route, provenance, and local naming.

Apply Definition 933.2 for rate-source locality, observer-symmetry exactness, no-global-frame refusal, consumer-lock exactness, transport stability, and nonescape. The checked TasteGate marker following the carrier records that the formal carrier alignment is the same single displayed surface.

Apply Definition 933.3, Definition 933.4, and Definition 933.5. These results show that observer swaps, downstream maximum-rate consumers, and $\mathsf{CrossHistCausalRate}^{\uparrow}$ handoffs all read the same gate rows and cannot bypass the no-global-frame refusal or consumer lock.

Finally apply Definition 933.6. The package theorem assembles those rows over $\mathsf{MultiHistConfig}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, and NameCert, so no listed obligation remains outside the finite package. □

Closure status for $\mathsf{NameCert}_{\mathsf{MaxRateReadGate}^{\uparrow}}$.

Constructive story: A MaxRateReadGate$^{\uparrow}$ packet is generated from a $\mathrm{MultiHistConfig}^{\uparrow}$ row, a MaxCausalRate$^{\uparrow}$ row, an observer-symmetry row, a no-global-frame refusal row, a maximum-rate consumer lock, componentwise hsame transports, psame probe stability, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MaxRateReadGate}^{\uparrow}}$, seed)

Scope closed: Definition 933.1, Definition 933.2, Definition 933.3, and Definition 933.6 bind the finite maximum-rate read gate with observer symmetry, no-global-frame refusal, consumer locking, and local NameCert packaging.

Formal verification: FormalStatus(NameCert$_{\mathsf{MaxRateReadGate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, global frames, metric tensors, Lorentz frames, host real speeds, detached maximum-rate values, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rate-source locality, observer-symmetry exactness, no-global-frame refusal, consumer-lock exactness, transport stability, and non-escape for the same finite packet.

Chapter 934

A Concrete Naming Certificate for $\mathsf{CellularAutomaton}^{\uparrow}$

The $\mathsf{CellularAutomaton}^{\uparrow}$ surface realizes the cellular substrate reading of chapter 9390 as a finite NameCert packet. It sits beside the transition-oriented carrier of chapter 482, the accepted-and-rejected observation sieve of chapter 932, and the gated public read of chapter 933. Its distinguishing data are a local rule table and a finite orbit ledger, so a pattern tag is certified only as a read from displayed cellular rows.

Falsifiable prediction: A $\mathsf{CellularAutomaton}^{\uparrow}$ consumer that reads a pattern tag without passing through the displayed finite orbit ledger is outside the certificate.

Independence witness: continuationautomaton, kernelobservationsieve, maxratereadgate, haltingdistinctiontrace: none is bijective because $\mathsf{CellularAutomaton}^{\uparrow}$ carries a local rule-table and orbit-ledger row that the siblings lack, while it lacks their distinct continuation or rate-gate payload rows.

Definition 934.1 (CellularAutomaton NameCert carrier)**.** *A* $\mathsf{CellularAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, I, O, T, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the local cellular rule-table row, including the finite neighbourhood alphabet and every displayed transition entry;*
2. *I is the initial row of cells from which the accepted orbit segment is generated;*
3. *O is the finite orbit-segment ledger listing the rows reached from I by repeated reads of R;*
4. *T is the pattern-tag classifier row assigning finite tags such as ether, glider, collision, or refused pattern to rows visible in O;*
5. *L is the observation ledger recording which row windows, tag reads, and Rule 110 boundary reads are admitted;*
6. *H records componentwise* hsame *transports for R, I, O, T, L, P, N;*
7. *C records displayed* Cont *routes through the rule table, initial row, orbit segment, classifier, and observation ledger;*

8. *P is the* Pkg *provenance over* $\mathsf{CellularAutomaton}^{\uparrow}$*, the cellular reading of chapter 9390,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularAutomaton}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only by the displayed rule table, initial row, finite orbit ledger, pattern tags, observation ledger, transport rows, continuation routes, provenance, and local naming row. It has no coordinate for an unbounded orbit, external induction over all times, host equality of cellular histories, quotient stream equality, hidden glider catalog, or detached Rule 110 oracle.

Lean4 ✓ `BEDC.Derived.CellularAutomatonUp.TasteGate.CellularAutomatonTasteGate single carrier alignment`

Theorem 934.2 (CellularAutomaton NameCert obligations)**.** *For the carrier of Definition 934.1, the local* $\mathsf{NameCert}_{\mathsf{CellularAutomaton}^{\uparrow}}$ *surface consists exactly of carrier exposure for $A = (R, I, O, T, L, H, C, P, N)$, rule-table accountability through R, initial-row accountability through I, finite orbit exposure through O, pattern-tag reads through T, observation-ledger replay through L, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. No consumer can read a cellular pattern, Rule 110 tag, universal simulation claim, or closed-observation boundary as* $\mathsf{CellularAutomaton}^{\uparrow}$ *evidence without passing through these rows.*

Proof. Carrier exposure is projection from Definition 934.1. The finite packet displays the rule table, initial row, orbit segment, classifier, observation ledger, transports, continuation routes, provenance, and local naming row as its only coordinates.

Rule-table and initial-row accountability are read from R and I. The orbit ledger O is admitted only as the finite route generated from these rows, so a consumer cannot replace the local rule by a host evaluator or an unrecorded cellular history.

Pattern-tag and observation-ledger replay are read from T and L. The classifier may name ether, glider, collision, Rule 110 boundary, or refused-pattern tags only for windows already displayed in O and only through the admitted ledger reads.

Finally use H, C, P, N. These rows transport, replay, record, and name exactly the displayed packet. Since the carrier lacks coordinates for unbounded induction, quotient streams, host cellular equality, hidden catalogs, and detached oracles, the listed rows exhaust the local NameCert surface. □

Theorem 934.3 (Rule 110 closed-observation boundary)**.** *Every Rule 110 or cellular-substrate read exported by an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *carrier factors through the finite route*

$$R, I, O, T, L, H, C, P, N.$$

In particular, a consumer may cite Rule 110 minimality, a pattern tag, or a closed-observation boundary only as a finite ledger read from O and L; it does not obtain an unbounded orbit, an internal induction principle, a host evaluator, a quotient stream representative, or a cellular oracle outside the packet.

Proof. Begin with the route inventory. The only Rule 110-facing rows in the packet are the rule table R, the initial row I, the finite orbit ledger O, the classifier T, and the observation ledger L.

Apply Definition 934.2. A pattern tag or cellular-substrate read is accepted only when the classifier row reads a window already present in the displayed orbit ledger and the observation ledger admits that read.

The non-escape claim is finite-route inspection. The continuation routes C have no edge to an unbounded time domain, a host evaluator, quotient stream equality, or an external induction rule. The provenance and local naming rows P, N therefore certify only the displayed closed-observation segment. □

Theorem 934.4 (CellularAutomaton local rule exposure)**.** *Every accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *read first exposes the local rule row R and the initial row I before any orbit, pattern tag, Rule 110 boundary, transport, continuation, provenance, or local naming row can be consumed. The read therefore cannot start from a host cellular evaluator, an implicit rule convention, or an unrecorded initial configuration.*

Lean4 ✓ `BEDC.Derived.CellularAutomatonUp.TasteGate.CellularAutomatonCarrier local rule exposure`

Proof. Project the carrier of Definition 934.1. The first two cellular coordinates are exactly the local rule table R and the initial row I.

Apply the NameCert obligations of Definition 934.2. Orbit exposure, pattern tags, observation-ledger replay, transport, route replay, package provenance, and local naming are all accepted only as rows of the displayed packet.

Finally use the closed-observation boundary of Definition 934.3. Since every Rule 110-facing read factors through $R, I, O, T, L, H, C, P, N$, no accepted route can bypass R and I. □

Theorem 934.5 (CellularAutomaton accepted-window exhaustion)**.** *Every pattern-tag read exported by an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *packet is exhausted by the finite orbit ledger O and the observation ledger L. A consumer may classify only windows already displayed in O and admitted by L; it obtains no hidden pattern catalog, all-time orbit stream, or detached Rule 110 observation row.*

Proof. Read the carrier of Definition 934.1. The only orbit-facing coordinate is O, and the only observation-admission coordinate is L.

Apply Definition 934.2. The classifier row T names pattern tags only through the displayed orbit and observation rows, while H, C, P, N preserve and replay the same finite packet.

The closed-observation boundary of Definition 934.3 then excludes an unbounded orbit, host evaluator, quotient stream representative, or cellular oracle. Thus the accepted pattern-window surface is exactly the finite pair O, L together with the displayed certificate rows. □

Theorem 934.6 (CellularAutomaton finite-route non-escape)**.** *No Rule 110 consumer of an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *packet can bypass*

$$R, I, O, T, L, H, C, P, N$$

into an unbounded orbit, host evaluator, quotient stream, hidden pattern catalog, or detached cellular oracle. Every public Rule 110 read remains a finite route through the displayed local rule, initial row, orbit ledger, classifier, observation ledger, transport, continuation, provenance, and naming rows.

Proof. Start with the carrier of Definition 934.1. Its coordinates list the complete accepted route $R, I, O, T, L, H, C, P, N$ and no additional cellular substrate coordinate.

The local NameCert obligations of Definition 934.2 force each accepted read through the displayed rows. Rule-table accountability, initial-row accountability, finite orbit exposure, classifier replay, and observation-ledger replay leave no route to an external evaluator or hidden catalog.

Finally apply Definition 934.3. Rule 110-facing reads are closed observations of the finite packet, so unbounded or quotient-stream routes are outside the certificate. □

Theorem 934.7 (CellularAutomaton rule-table exactness)**.** *Every transition read exported by an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *carrier is one displayed local rule-table row from* R*, transported by the componentwise* hsame *row* H *and replayed through the continuation row* C*. The read cannot cite a host cellular evaluator, an implicit transition convention, or a rule entry absent from* R*.*

Proof. Project the accepted packet to R, H, C. The carrier has a single rule-table coordinate, and the local NameCert obligations make every transition-facing read pass through that coordinate.

The transport row H can only compare displayed rule-table entries componentwise. It does not add a missing neighbourhood case or identify two host evaluators.

The continuation row C replays the selected local transition from the displayed table. Since no separate evaluator or oracle coordinate exists in the packet, exactness is precisely the assertion that the transition read is one row of R carried through H and C. □

Theorem 934.8 (CellularAutomaton finite orbit exposure)**.** *Every orbit read exported by an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *carrier is a finite ledger row in* O *generated from the initial row* I *and the rule table* R*. The packet exposes no unbounded orbit coordinate, no all-time induction principle, and no hidden scheduler outside the displayed ledger route* R, I, O, H, C, P, N*.*

Proof. Start from the carrier coordinates. The initial cellular data are stored in I, the local transition table is stored in R, and the only orbit-facing coordinate is the finite ledger O.

The admitted route reads O as the ledger generated from I by repeated displayed uses of R below the finite segment named in the packet. Transport through H and replay through C preserve those displayed rows rather than replacing them with a host run.

An unbounded orbit or all-time induction rule would require a coordinate not present in R, I, O, H, C, P, N. Thus every public orbit read remains a finite ledger exposure. □

Theorem 934.9 (CellularAutomaton pattern-tag non-escape)**.** *Every ether, glider, collision, Rule 110 boundary, or refused-pattern tag exported by an accepted* $\mathsf{CellularAutomaton}^{\uparrow}$ *carrier factors through the finite orbit ledger* O *and the observation ledger* L*. A tag read may use the classifier row* T*, the transport row* H*, the route row* C*, and the package and naming rows* P, N*, but it cannot escape to a hidden pattern catalog, an unbounded orbit, a host cellular evaluator, or an oracle for cellular universality.*

Proof. Project the tag-facing rows. The classifier row T names only pattern tags for windows already visible in the finite orbit ledger O, and the observation ledger L records which of those reads are admitted.

Transport and continuation are still finite. The H row compares displayed tag, orbit, and observation rows componentwise, while C replays the route from the rule and initial rows through the finite orbit window and into the tag read.

The excluded objects are absent coordinates. A hidden pattern catalog, unbounded orbit, host evaluator, or universality oracle cannot be recovered from O, T, L, H, C, P, N, so every accepted tag remains inside the finite ledger surface. □

The Lean TasteGate namespace BEDC.Derived.CellularAutomatonUp.TasteGate must provide Nontrivial $\mathsf{CellularAutomaton}^{\uparrow}$ and FieldFaithful $\mathsf{CellularAutomaton}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularAutomaton}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CellularAutomaton}^{\uparrow}$ packet is generated from a finite cellular rule table, initial row, finite orbit ledger, pattern-tag classifier, observation ledger, componentwise hsame transports, displayed continuation routes, package provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.CellularAutomatonUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{CellularAutomaton}\uparrow}$, maturePackageClosed)

Scope closed: Definition 934.1, Definition 934.2, Definition 934.3, Definition 934.7, Definition 934.8, and Definition 934.9 specify the displayed BHist, hsame, Cont, Pkg, NameCert, rule-table, orbit-ledger, and classifier scope.

Formal verification: FormalStatus(NameCert$_{\text{CellularAutomaton}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CellularAutomatonUp.TasteGate.CellularAutomatonTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, Rule 110 universality, cellular completeness, internal induction, quotient stream equality, or host evaluator correctness.

Upgrade path: No further paper-axis theory step remains for the finite cellular-automaton example surface; formal bridge checking remains a separate verification-axis step.

Chapter 935

A Concrete Naming Certificate for MetricSpace Compact-Continuity Source Lock

The MetricSpace compact-continuity source lock is the finite BEDC packet read before the compact-uniform-continuity construction of chapter 9237. It joins the located distance rows of chapter 57, the compact finite-net rows of chapter 129, the continuous pointwise-modulus rows of chapter 130, and the uniform-modulus consumer boundary of chapter 369. The packet is a source lock: a consumer sees the source rows together or receives no uniform-modulus precision row.

Falsifiable prediction: If the packet is atomic, no consumer can read a uniform-modulus precision row before the MetricSpaceUp distance row, CompactMetricUp finite-net row, ContinuousMapUp pointwise-modulus row, and RatUp finite-fold row are all displayed.

Independence witness: metric, compactmetric, continuousmap, uniformmodulus, finitenetminimumfold: this packet is not bijective to any one sibling because it carries all four source-lock rows and a consumer non-escape row simultaneously, while each sibling omits at least one required coordinate.

Definition 935.1 (MetricSpace compact-continuity source lock)**.** *A MetricSpace compact-continuity source-lock carrier is a finite* BHist *packet*

$$L = (M, K, F, P, R, H, C, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the located distance witness and classifier-stable metric comparison used by the compact and continuous inputs.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying the finite* ProbeBundle *of centers and the coverage ledger for the same metric source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row, carrying the continuous graph, target metric boundary, and local modulus rows over the source metric M.*
4. *P is the* ProbeBundle *center-assignment row that aligns bundled probes from K with the pointwise-modulus centers consumed by F.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-fold row, carrying the finite lower-bound ledger over the bundled positive rational radius histories before any uniform precision row is exposed.*

6. *H records componentwise* hsame *transports for M, K, F, P, R, the consumer boundary, and the local naming row.*
7. *C records the displayed* Cont *routes that read distance, finite-net, pointwise-modulus, probe-assignment, and rational-fold rows before the uniform-modulus consumer route is available.*
8. *N is the local* NameCert *row for the source-lock packet.*

The carrier is finite and source-locked. It contains no host compactness theorem, no open-cover selector, no choice-selected modulus, no quotient metric equality, and no ambient completeness row.

Lean4 ✓ BEDC.Derived.MetricSpaceCompactContinuitySourceLockUp.MetricSpaceCompactContinuitySourceLockTas
single carrier alignment

Theorem 935.2 (MetricSpace compact-continuity source-lock obligations)**.** *For the carrier of Definition 935.1, the local* NameCert *surface consists of five finite obligations:*

1. *carrier exposure: every accepted packet exposes exactly the rows $L = (M, K, F, P, R, H, C, N)$;*
2. *metric source compatibility: the finite-net, pointwise-modulus, and rational-fold rows are read over the same* $\mathsf{Met}^{\uparrow}$ *source row M;*
3. *compact-continuous alignment: the probe centers supplied by K and P are the centers at which the pointwise-modulus rows of F are consumed;*
4. *finite-fold precision: the* $\mathsf{Rat}^{\uparrow}$ *row R is the only lower-bound fold available to the uniform-modulus consumer;*
5. *consumer non-escape: every uniform-modulus precision read factors through M, K, F, P, R, H, C, N and receives no host compactness, choice-selected modulus, quotient equality, or ambient completeness coordinate.*

Proof. Carrier exposure is projection from Definition 935.1. The rows M, K, F, P, R, H, C, N are coordinates of one finite BHist packet, so no consumer can read a hidden source row outside that tuple.

Metric source compatibility is enforced by the shared M coordinate. The compact finite-net row K, continuous pointwise-modulus row F, probe center assignment P, and rational lower fold R are all stored as routes over that displayed source metric.

Compact-continuous alignment is the role of P. A bundled center from K becomes a pointwise-modulus center for F only through the displayed probe-assignment row, and the hsame transports in H replay the same alignment after classifier comparison.

Finite-fold precision is supplied by R. The continuation rows in C route a tolerance request through the displayed finite bundle and then through the rational lower-bound fold before a precision row reaches the uniform-modulus consumer.

Finally, consumer non-escape follows from the package inventory. Since L has no coordinate for host compactness, open-cover choice, quotient metric equality, or ambient completeness, none of those principles can be read from the source-lock packet. The later Lean carrier for this same packet must provide Nontrivial and FieldFaithful instances in ‘TasteGate.lean‘ for the same source carrier, so the formal consumer cannot switch to a different carrier while preserving the source-lock rows. □

Theorem 935.3 (MetricSpace source-lock row exposure)**.** *Every accepted MetricSpace compact-continuity source-lock packet exposes exactly the displayed rows*

$$M, K, F, P, R, H, C, N$$

of Definition 935.1. Its public surface contains the $\mathsf{Met}^{\uparrow}$ *source row, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row, the* $\mathsf{Rat}^{\uparrow}$ *lower-fold row, the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer boundary, the* BHist *carrier ledger, the componentwise* hsame *transports, the displayed* Cont *routes, and the local* NameCert *row. It contains no hidden host compactness, quotient metric equality, open-cover selector, choice modulus, or ambient completeness row.*

Proof. Project the carrier tuple in Definition 935.1. The only rows named by the packet are M, K, F, P, R, H, C, N, and the row-exposure obligation in Definition 935.2 states that every accepted packet displays exactly that tuple.

The sibling anchors determine the interpretation of those rows. The $\mathsf{Met}^{\uparrow}$ row is the metric source, $\mathsf{CompactMetric}^{\uparrow}$ supplies finite-net data, $\mathsf{ContinuousMap}^{\uparrow}$ supplies pointwise-modulus data, $\mathsf{Rat}^{\uparrow}$ supplies the finite lower fold, and $\mathsf{UniformModulus}^{\uparrow}$ is reached only as the downstream consumer boundary.

The excluded principles are absent by carrier inventory. Since no coordinate of L is a host compactness witness, quotient equality proof, open-cover selector, choice-selected modulus, or ambient completeness row, a public read from the packet cannot expose one. □

Theorem 935.4 (MetricSpace source-lock center alignment)**.** *In every accepted MetricSpace compact-continuity source-lock packet, the K finite-net centers, the P* ProbeBundle *assignment rows, and the F pointwise-modulus rows are read over the same* $\mathsf{Met}^{\uparrow}$ *source row M. Transport between the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net surface, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise surface, the* $\mathsf{Rat}^{\uparrow}$ *lower-fold surface, and the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer boundary is permitted only through the displayed H componentwise* hsame *rows and the displayed C* Cont *rows.*

Proof. Read the metric source compatibility and compact-continuous alignment clauses of Definition 935.2. They bind K, F, P, R to the single source coordinate M in Definition 935.1.

The center route is therefore unique inside the packet. A center exposed by the $\mathsf{CompactMetric}^{\uparrow}$ row K reaches a $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus row in F only through the ProbeBundle assignment row P.

Transport does not add a second source. The displayed H rows replay componentwise hsame comparison, and the displayed C rows replay the Cont routes from $\mathsf{Met}^{\uparrow}$ through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and toward $\mathsf{UniformModulus}^{\uparrow}$. □

Theorem 935.5 (MetricSpace source-lock uniform-modulus nonescape)**.** *Any uniform-modulus precision read from an accepted MetricSpace compact-continuity source-lock packet factors through the displayed rows*

$$M, K, F, P, R, H, C, N.$$

In particular, the $\mathsf{Rat}^{\uparrow}$ *lower-fold row R must be consumed before any* $\mathsf{UniformModulus}^{\uparrow}$ *precision row is public, and no precision read can bypass the* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows displayed by the source lock.*

Proof. Start from the finite-fold precision and consumer non-escape clauses of Definition 935.2. They state that a tolerance-to-precision read is routed through the rational lower-bound fold and through the exposed packet inventory.

The factorization is exactly the carrier route in Definition 935.1. The source $\mathsf{Met}^{\uparrow}$ row M, finite-net $\mathsf{CompactMetric}^{\uparrow}$ row K, pointwise $\mathsf{ContinuousMap}^{\uparrow}$ row F, probe-assignment row P, and $\mathsf{Rat}^{\uparrow}$ lower-fold row R are read before the continuation boundary reaches $\mathsf{UniformModulus}^{\uparrow}$.

Thus R is not an optional scalar witness. It is the displayed finite lower-fold row that the continuation rows C consume before the precision row becomes public, with H and N preserving the same source-lock identity. □

Closure status for $\mathsf{NameCert}_{\mathrm{MetricSpaceCompactContinuitySourceLockUp}}$.

Constructive story: A MetricSpace compact-continuity source-lock packet is generated from a MetricSpaceUp distance row, CompactMetricUp finite-net row, ContinuousMapUp pointwise-modulus row, ProbeBundle center assignment, RatUp finite-fold ledger, componentwise hsame transports, displayed Cont routes, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathrm{MetricSpaceCompactContinuitySourceLockUp}}$, seed)

Scope closed: Definition 935.1 and Definition 935.2 bind the metric source, compact finite-net, continuous pointwise-modulus, ProbeBundle center-assignment, RatUp lower-fold, transport, continuation, and consumer non-escape rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathrm{MetricSpaceCompactContinuitySourceLockUp}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a Lean carrier, a checked TasteGate instance, a public exported NameCert theorem, host compactness, open-cover compactness, choice-selected moduli, or ambient completeness.

Upgrade path: Obligation closure requires a Lean carrier for the finite source-lock packet plus TasteGate, Nontrivial, FieldFaithful, and checked non-escape declarations for the same carrier.

Chapter 936

A Concrete Naming Certificate for $\mathsf{ZNormal}^{\uparrow}$

The $\mathsf{ZNormal}^{\uparrow}$ packet realizes the total-host boundary named by chapter 9370 as finite normal-form certificate data. It sits beside the typed endpoint packet of chapter 562, the finite Zeckendorf normal-form packet of chapter 278, and the terminal continuation packet of chapter 683. The packet records terminal normality, bounded fuel replay, and non-escape from total host evidence without exporting a divergence oracle, host conversion row, or global normalization procedure.

Falsifiable prediction: Every accepted ZNormalUp packet exposes a finite terminal-normality row and admits no consumer route to a divergence oracle or host conversion row.

Independence witness: Nearest siblings: typedreductionnormalform, zeckendorf-normal, zcarry, continuationtracenormalform, and normalformconsistencyseal. None is bijective to $\mathsf{ZNormal}^{\uparrow}$ because typedreductionnormalform records a typed endpoint route, zeckendorf-normal records a finite combinatorial normal-word classifier, zcarry records a local carry classifier, continuation-tracenormalform records a terminal continuation trace, and normalformconsistencyseal records a consistency-facing seal; $\mathsf{ZNormal}^{\uparrow}$ keeps the total-host normal-form boundary itself as a finite packet.

Definition 936.1 ($\mathsf{ZNormal}^{\uparrow}$ carrier)**.** *A* $\mathsf{ZNormal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed reduction normal-form row read through* $\mathsf{TypedReductionNormalForm}^{\uparrow}$*;*
2. *F is the finite fuel row naming the bounded host reduction segment;*
3. *V is the terminal-normality row asserting that the segment ends at a normal form;*
4. *K is the Zeckendorf and carry-facing normal-form comparison row read through* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *and* $\mathsf{ZCarry}^{\uparrow}$ *where a finite word boundary is present;*
5. *R is the continuation terminal-read row read through* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$*;*
6. *H records componentwise* hsame *transport for typed endpoint, fuel, terminal normality, finite word comparison, continuation read, provenance, and local naming rows;*

7. *C records the displayed* Cont *routes from typed endpoint to fuel replay, from fuel replay to terminal normality, from terminal normality to the continuation read, and from those rows to the consumer-facing packet boundary;*
8. *P is the* Pkg *provenance over chapter 9370, chapter 562, chapter 278, chapter 683,* BHist*,* hsame*,* Cont*, and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *row.*

The carrier accepts only packets whose normal-form claim is read through a finite terminal row. It has no coordinate for a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, or ambient proof that every external computation halts.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal readback semantic certificate`

Theorem 936.2 ($\mathsf{ZNormal}^\uparrow$ NameCert obligations)**.** *For the carrier of Definition 936.1, the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *surface consists exactly of carrier admission for*

$$Z = (T, F, V, K, R, H, C, P, N),$$

typed endpoint readback through T, finite fuel exposure through F, terminal-normality exposure through V, finite normal-word comparison through K, continuation terminal-read exposure through R, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier. The rows $T, F, V, K, R, H, C, P, N$ are the only fields of an accepted $\mathsf{ZNormal}^\uparrow$ packet.

Read the endpoint rows first. The typed reduction row T is supplied by chapter 562; the finite fuel row F and terminal-normality row V bind that endpoint to a displayed total-host reduction segment.

Read the sibling normal-form rows next. The row K may cite the finite word discipline of chapter 278, while R may cite the terminal trace surface of chapter 683. These rows are auxiliary certificate coordinates, not routes to host conversion or general normalization.

Finally inspect H, C, P, N. They transport, replay, package, and name the same finite surface. Since no carrier coordinate names divergence, quotient computation, host conversion, or an unrestricted evaluator, the listed obligations exhaust the packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket namecert obligations`

Theorem 936.3 ($\mathsf{ZNormal}^\uparrow$ carry normalization obligation)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the row K is obligated to expose finite normal-word evidence through the $\mathsf{ZeckendorfNormal}^\uparrow$ *surface of chapter 278 and the local carry evidence of chapter 544. The exposure is a* BHist *row transported by* hsame*, replayed by* Cont*, packaged by* Pkg*, and named by* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$*; it is not a host conversion row, quotient computation, unrestricted evaluator, or global normalizer.*

Proof. Project the carrier of Definition 936.1. The only finite normal-word comparison coordinate is K, and it is listed beside the typed endpoint, fuel, terminal-normality, and continuation rows rather than outside the packet.

Use the NameCert obligation surface of Definition 936.2. It binds K to the same BHist, hsame, Cont, Pkg, and local naming rows as the rest of the accepted packet.

The sibling evidence is then read through the Zeckendorf and carry-facing chapters named in the carrier. Since the carrier has no coordinate for host conversion, quotient computation, unrestricted evaluation, or global normalization, the K obligation is exactly finite normal-word and carry evidence. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalCarryNormalizationObligation`

Theorem 936.4 ($\mathsf{ZNormal}^{\uparrow}$ continuation terminal route)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the continuation row R is reached only after the typed endpoint T, finite fuel F, and terminal-normality row V have been displayed. Its consumer route is the* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *terminal-read surface of chapter 683 transported through H, replayed through C, packaged through P, and named through N.*

Proof. Start from the finite packet $Z = (T, F, V, K, R, H, C, P, N)$. The row order places R after T, F, V and makes it a continuation terminal-read row, not a source of unbounded evaluation.

Apply Definition 936.2. The theorem exposes typed endpoint readback, finite fuel, terminal normality, continuation terminal-read, and the structural rows in one local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Now read the sibling terminal trace through chapter 683. The transport, continuation, package, and naming rows are H, C, P, N, so no host conversion, quotient computation, unrestricted evaluator, or global normalizer can replace the terminal continuation route. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalContinuationTerminalRoute`

Theorem 936.5 ($\mathsf{ZNormal}^{\uparrow}$ typed boundary exhaustion)**.** *Every consumer of the displayed* $\mathsf{ZNormal}^{\uparrow}$ *rows*

$$T, F, V, K, R, H, C, P, N$$

must factor through the typed endpoint of chapter 562, the finite normal-word and carry row of Definition 936.3, the continuation terminal route of Definition 936.4, and the local BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. The boundary exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.*

Proof. First expose the typed side. The carrier definition and Definition 936.2 make T, F, V the displayed route from typed endpoint through finite fuel to terminal normality.

Next expose the two downstream rows. The carry-normalization obligation fixes K as finite Zeckendorf and carry evidence, while the continuation-terminal theorem fixes R as the terminal read through the displayed continuation trace.

Finally collect H, C, P, N. These rows provide the only componentwise transport, replay, provenance, and local naming surface for all consumer reads. Since none of the cited rows contains host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization, the typed boundary is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket typed boundary exhaustion`

Theorem 936.6 ($\mathsf{ZNormal}^{\uparrow}$ fuel normality non-escape)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exposes its terminal normal-form claim only through the finite route*

$$T \longmapsto F \longmapsto V \longmapsto R$$

with the normal-word comparison row K available only as displayed sibling evidence. No consumer can use the packet to read a divergence oracle, a host conversion row, a quotient computation, an unrestricted evaluator, or a global normalizer.

Proof. Start from the typed endpoint row T. By construction it is a packet-visible endpoint row rather than a host conversion judgment.

Next read F and V. The fuel row names the finite replay segment, and the terminal-normality row records the endpoint of that segment. A consumer therefore receives a bounded normal-form packet, not an unbounded computation.

Use R and K only after the terminal row is displayed. The continuation trace row replays the terminal read, while the Zeckendorf and carry-facing row records finite word comparison where that sibling surface is relevant.

The excluded objects have no carrier coordinate. Thus transport, continuation replay, provenance, and local naming cannot manufacture a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, or global normalizer from the accepted packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket fuel normality nonescape`

Theorem 936.7 ($\mathsf{ZNormal}^{\uparrow}$ total-host boundary)**.** *The* $\mathsf{ZNormal}^{\uparrow}$ *packet gives a concrete finite interface for the host-side total-language boundary of chapter 9370. A public consumer may read the typed endpoint, finite fuel, terminal normality, finite normal-word comparison, continuation terminal read, transports, routes, provenance, and local name of*

$$Z = (T, F, V, K, R, H, C, P, N),$$

but it may not reread this surface as Turing completeness of the host, as a proof that every substrate computation halts, or as a route from substrate universality to host-level partial functions.

Proof. First bind the host-side surface. The carrier of Definition 936.1 records exactly the typed endpoint, finite replay, terminal-normality, sibling normal-word, continuation-read, transport, route, provenance, and naming rows.

Then compare this surface with chapter 9370. That chapter separates the total host from the universal substrate. The $\mathsf{ZNormal}^{\uparrow}$ packet inhabits the host side of that separation because every accepted read is through finite terminal-normality rows.

Finally apply Definition 936.6. It excludes divergence, host conversion, quotient computation, unrestricted evaluation, and global normalization routes. Consequently no consumer can turn the finite normal-form packet into host Turing completeness or a substrate halting theorem. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host boundary`

Theorem 936.8 ($\mathsf{ZNormal}^{\uparrow}$ terminal readback exactness)**.** *For any accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

every terminal read factors through the displayed rows T, F, V, R, H, C, P, N*. The read may consume the sibling normal-word row* K *only after the terminal-normality row* V *is displayed, and it cannot consult host conversion, quotient computation, unrestricted evaluation, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalReadbackExactness`

Proof. Project the carrier of Definition 936.1. The terminal read row is R, and the only rows before it are the typed endpoint T, the finite fuel row F, and the terminal-normality row V.

Use the displayed continuation routes in C. They replay the path from T through F and V to R, while H, P, N transport, package, and name that same route.

The row K is a sibling normal-word coordinate rather than a terminal readback source. Since no coordinate names host conversion, quotient computation, unrestricted evaluation, or a global normalizer, every accepted terminal read is exactly the displayed finite readback surface. □

Theorem 936.9 (ZNormal$^{\uparrow}$ terminal fuel route admission)**.** *For any accepted* ZNormal$^{\uparrow}$ *packet, every terminal read must expose the finite sequence T, F, V, R before consumer use. The sequence is transported only by the displayed* hsame *row H, replayed only by the displayed* Cont *row C, packaged only by P, and named only by N.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel route admission`

Proof. Project the carrier of Definition 936.1. The terminal-facing rows are the typed endpoint T, finite fuel F, terminal-normality row V, and continuation terminal-read row R.

Use Definition 936.4 and Definition 936.8. They place R after T, F, V and keep the terminal read inside the displayed finite route.

The structural rows are exactly H, C, P, N. They transport, replay, package, and name the route, and no other carrier coordinate can admit a terminal read before the finite fuel and terminal-normality rows have been displayed. □

Theorem 936.10 (ZNormal$^{\uparrow}$ finite host boundary refusal)**.** *No accepted* ZNormal$^{\uparrow}$ *packet route can replace the displayed rows T, F, V, R by host conversion, quotient computation, divergence oracle, unrestricted evaluator, or global normalizer. Every accepted route remains a* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*-local read of the finite carrier.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalFiniteHostBoundaryRefusal`

Proof. Begin with the carrier of Definition 936.1. It lists $T, F, V, K, R, H, C, P, N$ as the complete packet and names no host conversion row, quotient computation, divergence oracle, unrestricted evaluator, or global normalizer.

Apply Definition 936.6 and Definition 936.9. The terminal claim is read through finite fuel and terminal normality before the continuation row is consumed.

Finally use Definition 936.13. The only movement available to the packet is through H, C, P, N, so an accepted route cannot substitute any excluded host or quotient object for the displayed terminal fuel rows. □

Theorem 936.11 (ZNormal$^{\uparrow}$ sibling normal-word handoff)**.** *The K row of an accepted* ZNormal$^{\uparrow}$ *packet may consume the sibling surfaces of chapter 562, chapter 278, chapter 544, and chapter 683 only as displayed finite evidence. It does not provide host computation, a conversion oracle, a quotient normalizer, or a proof that all external computations normalize.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket sibling normalword handoff`

Proof. Read the sibling handoff through the carrier coordinates. The typed endpoint row T, finite fuel row F, terminal row V, comparison row K, and continuation row R are separate displayed fields.

The cited sibling chapters supply finite certificate surfaces: typed endpoint evidence, finite Zeckendorf normal-word evidence, local carry evidence, and terminal continuation-trace evidence. The K row can package those surfaces only through the displayed hsame, Cont, Pkg, and NameCert rows of the accepted packet.

No carrier coordinate supplies a host computation relation, conversion oracle, quotient normalizer, or external normalization theorem. Thus the sibling handoff remains finite evidence rather than a route to total host computation beyond the packet. □

Theorem 936.12 ($\mathsf{ZNormal}^{\uparrow}$ obligation closure package)**.** *The carrier admission of Definition 936.1, the NameCert obligations of Definition 936.2, the fuel non-escape boundary of Definition 936.6, the total-host boundary of Definition 936.7, the terminal readback exactness of Definition 936.8, and the sibling normal-word handoff of Definition 936.11 assemble the five-row* obligation *surface for* $\mathsf{ZNormal}^{\uparrow}$ *over* $T, F, V, K, R, H, C, P, N$.

Proof. First fix the accepted carrier $Z = (T, F, V, K, R, H, C, P, N)$. Carrier admission and local naming expose only the rows listed in Definition 936.1.

Apply Definition 936.2 to bind the NameCert surface. Then use Definition 936.6 and Definition 936.7 to keep every consumer read inside finite fuel, terminal normality, continuation readback, and total-host boundary evidence.

Finally use Definition 936.8 and Definition 936.11. These identify the terminal read and the sibling normal-word handoff as displayed finite rows, so the carrier, NameCert, fuel, terminal readback, sibling handoff, and host-boundary components form one obligation package. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket obligation closure package`

Theorem 936.13 ($\mathsf{ZNormal}^{\uparrow}$ route transport)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the route from the typed endpoint row T through finite fuel F, terminal normality V, the sibling normal-word row K, and the terminal continuation read R is transported only by the displayed componentwise row H and replayed only by the displayed continuation row C. No accepted route can insert host conversion, quotient computation, unrestricted evaluation, or a global normalizer between these coordinates.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRouteTransport`

Proof. Project the accepted packet through Definition 936.1. The only route-bearing coordinates exposed by the carrier are the componentwise transport row H and the continuation replay row C.

Apply Definition 936.2 to bind these rows to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface. Then use Definition 936.6 to keep the path from T through F and V inside finite fuel and terminal-normality evidence.

Finally apply Definition 936.8 and Definition 936.11. The terminal read and sibling normal-word row are reached through the same displayed transport and replay rows, so no external normalizer or host computation route is available. □

The root surface consumes sibling finite evidence before terminal readback is available. The row K is the sibling handoff row: it reads the finite normal-word carrier of chapter 278 and the carry classifier of chapter 544 as displayed BHist evidence. Without those two sibling rows, the terminal normality route cannot be read as finite Zeckendorf and carry evidence inside the $\mathsf{ZNormal}^{\uparrow}$ packet.

Theorem 936.14 ($\mathsf{ZNormal}^{\uparrow}$ root terminal non-escape interface)**.** *Every root consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet must read terminal normality through the displayed path*

$$T \to F \to V \to R,$$

with K available only as finite sibling normal-word and carry evidence from $\mathsf{ZeckendorfNormal}^{\uparrow}$ *and* $\mathsf{ZCarry}^{\uparrow}$*. No host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer can enter the root-facing interface.*

Proof. Begin with Definition 936.6. It places the terminal normal-form claim on the finite route from typed endpoint T through fuel F and terminal row V to continuation read R.

Apply Definition 936.8. The terminal read factors through T, F, V, R, H, C, P, N, and the row K is consumed only after V is already displayed.

Use Definition 936.11 to identify K as finite sibling evidence from $\mathsf{ZeckendorfNormal}^{\uparrow}$ and $\mathsf{ZCarry}^{\uparrow}$. Then apply Definition 936.13 to keep the whole root route inside H, C, P, N without host conversion, quotient computation, unrestricted evaluation, divergence, or global normalization. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal nonescape interface`

Theorem 936.15 ($\mathsf{ZNormal}^{\uparrow}$ root carrier exposure)**.** *Every root consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet first sees exactly the displayed tuple*

$$Z = (T, F, V, K, R, H, C, P, N).$$

The rows expose the typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, terminal readback, componentwise hsame *transport, displayed* Cont *replay, package provenance, and local* NameCert *naming. No root-unblock consumer receives a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, or global normalizer outside those coordinates.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root unblock carrier exposure`

Proof. Project the carrier of Definition 936.1. It lists $T, F, V, K, R, H, C, P, N$ as the complete finite packet and gives no other coordinate to a root consumer.

Apply Definition 936.2. The NameCert surface identifies the same fields as the complete local naming, transport, route, provenance, and package surface.

Use Definition 936.6 and the displayed TasteGate contract row. The finite fuel boundary and the carrier-faithful TasteGate reading keep the exposed tuple from becoming a route to host conversion, quotient computation, unrestricted evaluation, divergence, or global normalization. □

Theorem 936.16 ($\mathsf{ZNormal}^{\uparrow}$ root terminal route)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the terminal normal-form route used by root consumers is*

$$T \to F \to V \to R,$$

with K available only as sibling normal-word evidence. The route is transported through H, replayed through C, packaged through P, and named through N; it cannot be replaced by host conversion, quotient computation, unrestricted evaluation, or global normalization.

Proof. Begin with Definition 936.6. It already binds terminal normality to the finite path from the typed endpoint T through fuel F and terminal row V to the continuation read R.

Apply Definition 936.8. The theorem makes R the exact terminal readback row and permits K only after V has been displayed.

Finally combine Definition 936.11 with Definition 936.13. The sibling row remains finite evidence, while H, C, P, N transport, replay, package, and name the same terminal route without adding any excluded host or global normalization object. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootTerminalRouteHostTailExclusion`

Theorem 936.17 ($\mathsf{ZNormal}^{\uparrow}$ root non-escape ledger)**.** *The root-unblock ledger of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet is exhausted by carrier admission, NameCert obligations, finite fuel non-escape, terminal readback exactness, sibling normal-word handoff, and route transport. Thus every downstream normal-form row must pass through*

$$T, F, V, K, R, H, C, P, N$$

and the finite sibling surfaces named by K*, while host conversion, quotient computation, unrestricted evaluation, divergence detection, and global normalization remain absent.*

Proof. First assemble the carrier and naming rows using Definition 936.1 and Definition 936.2. They expose the packet fields and no separate host computation surface.

Next apply Definition 936.6, Definition 936.8, and Definition 936.11. These rows bind finite fuel, terminal readback, and sibling evidence to the displayed packet.

Finally use Definition 936.13 together with the displayed TasteGate contract row. Transport and replay stay inside H, C, P, N, so the ledger has no coordinate for host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootUnblockNonescapeLedger`

Theorem 936.18 ($\mathsf{ZNormal}^{\uparrow}$ normal output boundary)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the public normal-output row is exactly the terminal-normality row V *together with the terminal read row* R*. This output boundary exports no divergence oracle, host conversion, quotient computation, unrestricted evaluator, global normalizer, or total host-computation principle.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal output boundary`

Proof. Project the carrier of Definition 936.1. The normal-output data are displayed as V and R, while T, F, K, H, C, P, N only type, fuel, compare, transport, replay, package, and name that same finite packet.

Use Definition 936.2 to make the projection a local certificate read. The finite-output claim is then constrained by Definition 936.6, which excludes divergence and unrestricted evaluation at the fuel boundary.

The terminal side is fixed by Definition 936.8, and the sibling side is fixed by Definition 936.11. Together they show that V with R is the whole public normal-output boundary and that no host conversion, quotient computation, or global normalizer is exported. □

Theorem 936.19 ($\mathsf{ZNormal}^{\uparrow}$ consumer non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

factors through the displayed rows $T, F, V, K, R, H, C, P, N$ *and the finite sibling surfaces cited in Definition 936.11. The consumer obtains no route to host conversion, quotient computation, unrestricted evaluation, global normalization, divergence detection, or total host computation outside those rows.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket consumer nonescape`

Proof. Project Z through Definition 936.1. This exposes the typed endpoint, fuel, terminal normality, sibling normal-word, terminal read, transport, continuation, provenance, and naming rows as the complete consumer surface.

Apply Definition 936.2. The certificate admits consumer reads only through those displayed coordinates and their local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ provenance.

The finite boundary is inherited from Definition 936.6 and Definition 936.8. The only sibling handoff available to downstream chapters is the finite handoff of Definition 936.11. Hence every consumer factors through the displayed packet and cannot escape to any omitted host or global normalization object. □

The follow-up Lean TasteGate.lean contract for this chapter must provide `Nontrivial ZNormalUp` and `FieldFaithful ZNormalUp` instances for the same carrier.

Theorem 936.20 ($\mathsf{ZNormal}^\uparrow$ carrier obligation row)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

carrier admission binds the typed endpoint row T, finite fuel row F, terminal-normality row V, Zeckendorf and carry-facing comparison row K, continuation terminal-read row R, componentwise hsame *row H, displayed* Cont *row C, package row P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *row N to the same finite* BHist *surface. The carrier row exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalCarrierObligationRow`

Proof. Project the carrier of Definition 936.1. It displays exactly $T, F, V, K, R, H, C, P, N$ and gives no additional coordinate to a root consumer.

Apply Definition 936.2. The local NameCert surface reads those fields as carrier admission, endpoint exposure, finite fuel, terminal normality, sibling comparison, terminal read, transport, route, provenance, and naming for the same packet.

Use Definition 936.6 and Definition 936.8. They keep the carrier read inside finite fuel and terminal readback rows, so the excluded host and global-normalization objects cannot be recovered from carrier admission. □

Theorem 936.21 ($\mathsf{ZNormal}^\uparrow$ carry classifier row)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet, the comparison coordinate K is the only ZeckendorfNormal and ZCarry classifier surface. It is read as a* BHist *row transported by H, replayed by C, packaged by P, and named by N after the typed endpoint, finite fuel, and terminal-normality rows have been displayed. It is not a host conversion row, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.*

Proof. Start from Definition 936.1 and Definition 936.2. They place K inside the same finite packet as T, F, V, R, H, C, P, N and bind it to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ route.

Apply Definition 936.3 and Definition 936.11. These theorems identify the K row with the finite ZeckendorfNormal and ZCarry comparison surface rather than with host computation.

Finally use Definition 936.13. The comparison row is transported and replayed only through H and C, then packaged and named by P, N, so no quotient computation, unrestricted evaluator, divergence oracle, or global normalizer can enter the classifier row. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalCarryClassifierRow`

Theorem 936.22 ($\mathsf{ZNormal}^{\uparrow}$ normal-form ledger row)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, terminal normality is recorded only by the ledger route through*

$$T, F, V, R, H, C, P, N,$$

with K available as the finite sibling comparison row named by Definition 936.21. The ledger binds the terminal-normality claim to BHist*,* Cont*,* hsame*,* Pkg*, and* NameCert *rows and exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.*

Proof. Use Definition 936.20 to expose the displayed packet and no other terminal-normality coordinates. The terminal row is V, and the continuation read is R.

Apply Definition 936.6 and Definition 936.8. These theorems force terminal normality through the finite fuel path T, F, V, R and keep readback inside the displayed packet.

Now apply Definition 936.21 and Definition 936.13. The sibling comparison row remains finite evidence, while H, C, P, N provide the only transport, replay, package, and naming surface for the normal-form ledger. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal form ledger row`

Theorem 936.23 ($\mathsf{ZNormal}^{\uparrow}$ terminal-host boundary row)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

terminal-host use of $\mathsf{ZNormal}^{\uparrow}$ *must factor through the typed endpoint row T, finite fuel row F, terminal-normality row V, continuation terminal-read row R, sibling normal-word row K, and the structural rows H, C, P, N. No consumer of this terminal-host boundary may read host conversion, quotient computation, unrestricted evaluator, divergence oracle, substrate halting, or global normalizer outside that finite BEDC packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal host boundary`

Proof. Project Definition 936.1. The displayed carrier gives the typed endpoint, finite fuel, terminal-normality, continuation readback, sibling normal-word evidence, transports, continuations, package provenance, and NameCert row as the only terminal-host coordinates.

Apply Definition 936.22 and Definition 936.21. The normal-form ledger binds T, F, V, R, H, C, P, N, while the classifier row identifies K as finite sibling normal-word evidence rather than host computation.

Use Definition 936.4 and Definition 936.6. These force the continuation read and terminal-normality claim through finite fuel and terminal route data, so the host-side boundary is only finite BEDC packet data and exposes none of the excluded host or global-normalization objects. □

Theorem 936.24 ($\mathsf{ZNormal}^{\uparrow}$ root visible source package)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exposes the typed endpoint row T, finite fuel row F, terminal-normality row V, carry and Zeckendorf normal-word row K, continuation trace row R,* hsame *row H,* Cont *row C,* Pkg *row P, and local* NameCert *row N before any normal-output read. No normal output consumer can read a host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, or global normalizer outside that finite root package.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root visible source package`

Proof. Project the carrier of Definition 936.1. The displayed rows $T, F, V, K, R, H, C, P, N$ are the only root coordinates available to a normal-output consumer.

Use the carrier obligation, normal-form ledger, and normal-output boundary theorems. They force terminal normality through T, F, V, R and keep the sibling normal-word evidence in the finite comparison row K.

The structural rows then exhaust transport, continuation replay, package provenance, and local naming. Since the carrier has no host conversion, quotient-normalizer, unrestricted-evaluator, divergence-oracle, or global normalizer coordinate, the visible source package is exactly the displayed root packet. □

Theorem 936.25 ($\mathsf{ZNormal}^{\uparrow}$ root normal-word route totality)**.** *The finite normal-word route of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through chapter 278, chapter 544, and the local carrier rows* $T, F, V, K, R, H, C, P, N$*. It exports a carry and Zeckendorf normal-word surface only through the row* K*, with no host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, or global normalizer.*

Proof. Start from the visible source package of Definition 936.24. It places every normal-output read inside the local packet before the normal-word row can be consumed.

Apply the carry normalization obligation and sibling normal-word handoff. These identify K as the finite route through the Zeckendorf normal and ZCarry siblings, rather than a host computation or quotient normalization procedure.

Finally use route transport. The K row is carried only by H, replayed only by C, packaged only by P, and named only by N. The route is therefore total over the displayed finite packet and leaves no coordinate for the excluded global or host-side objects. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root normalword route totality`

Theorem 936.26 ($\mathsf{ZNormal}^{\uparrow}$ root continuation terminal exhaustion)**.** *Every continuation-terminal read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet passes through chapter 683 and the local rows* T, F, V, R, H, C, P, N*. The read exports no unrestricted evaluator, divergence oracle, host conversion, quotient normalizer, substrate halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root continuation terminal exhaustion`

Proof. Begin with the continuation terminal route. It places terminal readback in the row R after the typed endpoint T, finite fuel F, and terminal-normality row V have been displayed.

Use the continuation-trace normal-form sibling route together with terminal readback exactness. These rows make the terminal read a finite continuation trace, not an unrestricted evaluator or divergence oracle.

The structural rows H, C, P, N then transport, replay, package, and name the same finite terminal route. Carrier exhaustion leaves no host conversion, quotient normalizer, substrate halting coordinate, or global normalizer to export. □

Theorem 936.27 ($\mathsf{ZNormal}^{\uparrow}$ root carrier total-host readiness)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exposes the typed endpoint row* T*, finite fuel row* F*, terminal-normality row* V*, carry and normal-word row* K*, continuation terminal route* R*, componentwise* hsame *row* H*, displayed* Cont *row* C*,* Pkg *row* P*, and local* $\mathsf{NameCert}$ *row* N *before any total-host read. A total-host consumer receives only this finite root package and cannot read host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, substrate halting row, or global normalizer outside it.*

Proof. Project the carrier of Definition 936.1. The tuple $Z = (T, F, V, K, R, H, C, P, N)$ lists every root coordinate available before a total-host read.

Use Definition 936.24 and Definition 936.23. These rows place typed endpoint, fuel, terminal normality, sibling normal-word evidence, continuation readback, transports, continuations, package provenance, and local naming on the visible terminal-host route.

Apply Definition 936.17. Since the non-escape ledger exhausts the carrier and structural rows, no host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, substrate halting row, or global normalizer can be read outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root carrier total host readiness`

Theorem 936.28 ($\mathsf{ZNormal}^{\uparrow}$ root continuation-trace readiness)**.** *Every continuation-trace consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet must factor through Definition 936.4, Definition 936.14, and Definition 936.16. The continuation read is therefore reached only after* T, F, V *have been displayed, with* K *available as finite sibling normal-word evidence and* H, C, P, N *providing only transport, replay, provenance, and naming.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root continuation trace readiness`

Proof. Start with Definition 936.4. It places the continuation row R after the typed endpoint, finite fuel, and terminal-normality rows.

Next use Definition 936.14. The root-facing interface keeps the terminal route on $T \to F \to V \to R$ and treats K only as finite sibling evidence.

Finally apply Definition 936.16. The terminal route is transported through H, replayed through C, packaged through P, and named through N, with no replacement by host conversion, quotient computation, unrestricted evaluation, divergence, or global normalization. □

Theorem 936.29 ($\mathsf{ZNormal}^{\uparrow}$ root NameCert readiness)**.** *The displayed carrier and continuation rows of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet give a local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *readiness surface for downstream normal-form consumers. The surface consists of* $T, F, V, K, R, H, C, P, N$ *with the continuation route constrained by the root terminal interface, and it exports no host conversion, quotient normalizer, divergence oracle, or global normalization procedure.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root namecert readiness surface`

Proof. Begin with Definition 936.2. It identifies carrier admission, typed endpoint readback, finite fuel, terminal normality, sibling normal-word comparison, continuation terminal-read exposure, transport, replay, provenance, and local naming as the complete $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Apply Definition 936.27 to make the carrier rows visible before any total-host use. Then apply Definition 936.28 to force continuation-trace consumers through the terminal route rather than an ambient evaluator.

The same carrier exhaustion proves the exclusion clause. No displayed coordinate names host conversion, a quotient normalizer, a divergence oracle, or a global normalization procedure, so the local NameCert readiness surface cannot export one. □

Theorem 936.30 ($\mathsf{ZNormal}^{\uparrow}$ frontier carry readback totality)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet that exposes normal-word evidence is a* BHist *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

whose finite word row K reads $\mathsf{ZeckendorfNormal}^{\uparrow}$ *and* $\mathsf{ZCarry}^{\uparrow}$ *before the local K row is consumed by any root-facing normal-form reader. The readback route is transported by H, replayed by C, packaged by P, and named by N; it exports no host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket frontier carry readback totality`

Proof. Project the carrier of Definition 936.1. The only coordinate that can expose finite normal-word evidence is K, and the carrier text binds that row to $\mathsf{ZeckendorfNormal}^{\uparrow}$ and $\mathsf{ZCarry}^{\uparrow}$ where a finite word boundary is present.

Apply Definition 936.3 and Definition 936.11. These force the normal-word evidence to be read through the Zeckendorf and carry sibling surfaces before a root-facing consumer uses the local K row.

Finally use Definition 936.29. The structural rows H, C, P, N transport, replay, package, and name the same evidence, while carrier exhaustion excludes host conversion, quotient normalization, unrestricted evaluation, divergence, and global normalization. □

Theorem 936.31 ($\mathsf{ZNormal}^{\uparrow}$ continuation normal-word pullback)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the continuation terminal route and the normal-word row form one finite pullback through*

$$H, C, P, N.$$

Thus a consumer that reads the terminal continuation row R and the finite normal-word row K receives them as one transported, replayed, packaged, and named BHist *packet over T, F, V, K, R, not as a terminal route detached from the Zeckendorf and carry-facing evidence.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket continuation normalword pullback`

Proof. Start with Definition 936.4. It places R after the typed endpoint T, finite fuel F, and terminal-normality row V.

Next apply Definition 936.30. The normal-word row K is already tied to $\mathsf{ZeckendorfNormal}^{\uparrow}$ and $\mathsf{ZCarry}^{\uparrow}$ before any root-facing readback consumes it.

Use Definition 936.13 to bind both routes through H and C, then use Definition 936.29 to package and name the shared surface through P and N. The result is one finite pullback packet through H, C, P, N. □

Theorem 936.32 ($\mathsf{ZNormal}^{\uparrow}$ total-host refusal exhaustion)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exports no host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, or global normalizer. Any total-host consumer must factor through the displayed* BHist *packet*

$$T, F, V, K, R, H, C, P, N$$

together with the finite sibling surfaces cited by K and R.

Proof. Project the root carrier from Definition 936.1. The displayed rows give typed endpoint, finite fuel, terminal normality, normal-word evidence, terminal continuation read, transport, replay, package provenance, and local naming, with no host conversion or global-normalization coordinate.

Apply Definition 936.27. It places all total-host reads inside the finite root package before any public normal-form consumer receives the packet.

Now combine Definition 936.31 with Definition 936.17. The pullback keeps K and R in the same transported packet, and the non-escape ledger exhausts all downstream reads through the listed rows.

Therefore the refused objects remain absent from the export surface. A consumer asking for host conversion, a quotient normalizer, an unrestricted evaluator, a divergence oracle, or a global normalizer asks for data outside the accepted $\mathsf{ZNormal}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host refusal exhaustion`

Theorem 936.33 ($\mathsf{ZNormal}^{\uparrow}$ finite-fuel carrier admission)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet is read from one finite* BHist *carrier*

$$Z = (T, F, V, K, R, H, C, P, N)$$

whose rows expose the typed endpoint T*, finite fuel* F*, terminal-normality row* V*, finite normal-word row* K*, terminal continuation row* R*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *ledger* N*. A packet is accepted only when all nine rows are finite rows of that same carrier and no admission step replaces them by host conversion, quotient computation, unrestricted evaluation, a divergence oracle, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootFiniteFuelAdmissionCertificate`

Proof. Project the displayed carrier of Definition 936.1. Its source rows are exactly $T, F, V, K, R, H, C, P, N$, and each is a finite coordinate of the same BHist packet.

Use Definition 936.2 to attach the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ row, and use Definition 936.6 to keep the finite-fuel and terminal-normality rows inside the displayed packet.

Apply Definition 936.8, Definition 936.11, and Definition 936.12. These bind the readback, normal-word, continuation, transport, package, and naming rows to the same finite carrier.

The non-export clause is carrier exhaustion. Since no coordinate names the excluded host or quotient objects, no accepted packet can expose one at admission. □

Theorem 936.34 ($\mathsf{ZNormal}^{\uparrow}$ terminal-normality consumer gate)**.** *Let*

$$Z = (T, F, V, K, R, H, C, P, N)$$

be an accepted $\mathsf{ZNormal}^{\uparrow}$ *packet. Any root consumer that reaches the continuation row* R *must first read the typed endpoint row* T*, the finite-fuel row* F*, and the terminal-normality row* V *in the same packet. The consumer may then read* R *only through the displayed continuation route and the transported* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal normality consumer gate`

Proof. Start from Definition 936.33. It supplies the only source rows available to a root consumer and places T, F, V, R in one finite packet.

Apply Definition 936.4. That route admits a continuation read only after the typed endpoint, finite fuel, and terminal-normality rows have been exposed.

Use Definition 936.31 and Definition 936.13. These keep the continuation read on the same finite route as the sibling normal-word handoff and transport the result only through H, C, P, N. □

Theorem 936.35 ($\mathsf{ZNormal}^{\uparrow}$ no-divergence oracle ledger)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the exported ledger consists only of the typed endpoint, finite fuel, terminal-normality, sibling normal-word, terminal continuation, transport, continuation, package, and naming rows $T, F, V, K, R, H, C, P, N$*. It cannot export a divergence oracle, host conversion, quotient computation, unrestricted evaluator, or global normalizer to any root, normal-form, or MetaCIC-facing consumer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket no divergence oracle ledger`

Proof. Begin with Definition 936.33. It supplies the finite inventory of the packet and rules out all coordinates not named by $T, F, V, K, R, H, C, P, N$.

Use Definition 936.34 for the continuation side and Definition 936.8 with Definition 936.11 for the normal-word side. Both reads remain finite route reads.

Apply Definition 936.17 and Definition 936.32. These exhaust root-facing reads through the displayed transport, continuation, package, and naming ledger, and refuse total-host requests that do not factor through the packet.

Thus the ledger has no exported coordinate from which a divergence oracle, host conversion, quotient computation, unrestricted evaluator, or global normalizer could be read. □

Theorem 936.36 ($\mathsf{ZNormal}^{\uparrow}$ terminal consumer totality)**.** *Let*

$$Z = (T, F, V, K, R, H, C, P, N)$$

be an accepted $\mathsf{ZNormal}^{\uparrow}$ *packet. Every terminal normal-form consumer that reads* R *or* K *must first pass through carrier admission, the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *obligations, terminal-normality consumer gate, and total-host refusal exhaustion. The consumer therefore reads only the typed endpoint, finite fuel, terminal-normality, finite normal-word, terminal continuation, transport, replay, package, and local naming rows* $T, F, V, K, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalConsumerTotality`

Proof. Start with finite-fuel carrier admission. Definition 936.33 exposes $T, F, V, K, R, H, C, P, N$ as one finite BHist packet and gives no coordinate outside that packet to a terminal consumer.

Next apply the NameCert obligations and the terminal-normality consumer gate. The obligation surface binds the local naming row to the carrier, while Definition 936.34 places any continuation read behind T, F, V and transports it only through H, C, P, N.

Finally use no-divergence and total-host refusal. Definition 936.35 and Definition 936.32 exhaust the terminal-facing reads by the displayed rows, so the consumer has no route to host conversion, quotient computation, unrestricted evaluation, a divergence oracle, or a global normalizer. □

Theorem 936.37 ($\mathsf{ZNormal}^{\uparrow}$ finite-route readback boundary)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

every downstream readback of a normal word or terminal continuation factors through the finite routes certified by Definition 936.8, Definition 936.11, and Definition 936.31. The readback boundary is $T, F, V, K, R, H, C, P, N$ *and no host conversion, quotient computation, divergence oracle, unrestricted evaluator, or global normalizer is exposed at that boundary.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalFiniteRouteReadbackBoundary`

Proof. Begin with the terminal readback exactness theorem. It makes R a readback row of the accepted packet rather than a host-side evaluator result.

Now attach the normal-word row. Definition 936.11 reads K through the finite sibling normal-word surfaces, and Definition 936.31 binds K and R into one transported and replayed route through H, C, P, N.

The total-host refusal theorem closes the boundary. Since any total-host request that does not factor through the packet is refused, the finite readback route cannot be replaced by host conversion, quotient computation, a divergence oracle, unrestricted evaluation, or global normalization. □

Theorem 936.38 ($\mathsf{ZNormal}^{\uparrow}$ downstream normal-form package)**.** *Every downstream normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the finite package*

$$Z = (T, F, V, K, R, H, C, P, N).$$

The package is assembled from carrier admission, local NameCert *obligations, terminal consumer totality, finite-route readback boundary, no-divergence ledger, continuation normal-word pullback, and total-host refusal exhaustion. It exports no host conversion, quotient computation, divergence oracle, unrestricted evaluator, total host computation, substrate halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalDownstreamNormalFormPackage`

Proof. First expose the packet. The carrier admission theorem and the local NameCert obligations make $T, F, V, K, R, H, C, P, N$ the complete finite inventory available to a downstream normal-form consumer.

Next apply terminal consumer totality and finite-route readback boundary. The first theorem forces terminal consumers through $T, F, V, K, R, H, C, P, N$, and the second places normal-word and continuation readbacks on the same transported finite route.

Finally combine the no-divergence ledger, continuation normal-word pullback, and total-host refusal exhaustion. These results exhaust all public normal-form reads through the displayed rows and refuse every host-side or quotient-style request that lacks a coordinate in the accepted packet. □

Theorem 936.39 ($\mathsf{ZNormal}^{\uparrow}$ root public normal-form export)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the public normal-form export is exactly the finite surface*

$$Z = (T, F, V, K, R, H, C, P, N)$$

together with the finite sibling routes already certified by the normal-word handoff and continuation pullback. The export is not a host conversion procedure, an unrestricted evaluator, or a global normalization theorem.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root public normal form export`

Proof. Project the accepted packet. Carrier admission lists the typed endpoint, finite fuel, terminal-normality, normal-word, readback, transport, replay, package, and local naming rows as $T, F, V, K, R, H, C, P, N$.

Attach the sibling route results. The sibling normal-word handoff and continuation normal-word pullback route K and R through the same finite packet and do not add a host evaluator coordinate.

The non-export claim follows from the total-host refusal and no-divergence ledgers. They exhaust public reads through the displayed rows and leave no coordinate for host conversion, unrestricted evaluation, or global normalization. □

Theorem 936.40 ($\mathsf{ZNormal}^{\uparrow}$ root total-host obligation package)**.** *The root total-host obligation package for* $\mathsf{ZNormal}^{\uparrow}$ *consists of carrier admission for* $T, F, V, K, R, H, C, P, N$*, terminal fuel, sibling normal-word certification, continuation readback, componentwise transport, route replay, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *rows. Every root-facing total-host request must be discharged by this package or refused.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host obligation package`

Proof. Start with carrier admission and the local NameCert obligations. These expose the finite rows $T, F, V, K, R, H, C, P, N$ and bind them to the named $\mathsf{ZNormal}^{\uparrow}$ surface.

Next collect the terminal and sibling obligations. Terminal fuel is read through F, terminal-normality through V, the sibling normal word through K, and continuation readback through R.

Finally package transport, replay, provenance, and naming through H, C, P, N. The total-host refusal theorem says that a request outside these rows has no accepted coordinate, so the displayed collection is the complete root total-host obligation package. □

Theorem 936.41 ($\mathsf{ZNormal}^{\uparrow}$ root unary carry entry)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet that exposes a unary carry entry to a normal-form consumer must first route the typed endpoint row* T*, the finite fuel row* F*, and the terminal-normality row* V *through the visible source package of Definition 936.24. The finite word row* K *then reads the* $\mathsf{ZCarry}^{\uparrow}$ *sibling surface of chapter 544 before any consumer may use* K *as carry evidence.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root unary carry entry`

Proof. Start with Definition 936.24. It exposes the root packet as $T, F, V, K, R, H, C, P, N$ and gives the consumer no row outside the displayed carrier.

Next apply Definition 936.25. This forces normal-word consumption through the finite row K after the typed endpoint, fuel, and terminal-normality rows are already visible.

Finally use Definition 936.11. The handoff reads K through the Zeckendorf and $\mathsf{ZCarry}^{\uparrow}$ sibling surfaces, so unary carry entry reaches the root only through the displayed carry sibling and cannot supply host conversion, quotient computation, unrestricted evaluation, or a global normalizer. □

Theorem 936.42 ($\mathsf{ZNormal}^{\uparrow}$ root Zeckendorf handoff)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the normal-word row* K *may be consumed as Zeckendorf evidence only after the root visible source package has exposed* $T, F, V, K, R, H, C, P, N$ *and after the sibling handoff has routed* K *through* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *of chapter 278 and* $\mathsf{ZCarry}^{\uparrow}$ *of chapter 544. The terminal readback remains downstream of* R, H, C, P, N *and exports no host normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root zeckendorf handoff`

Proof. Project the visible source package by Definition 936.24. The packet exposes the typed endpoint, finite fuel, terminal-normality, normal-word, continuation, transport, replay, provenance, and naming rows as one root surface.

Apply Definition 936.25 and Definition 936.11. These results place the K row on the finite ZeckendorfNormal and $\mathsf{ZCarry}^{\uparrow}$ sibling route before the local root consumer uses it.

It remains to account for terminal readback. The continuation normal-word pullback of Definition 936.31 binds K and R into the same transported route through H, C, P, N, so the Zeckendorf handoff cannot detach into host normalization, quotient computation, unrestricted evaluation, or divergence detection. □

Theorem 936.43 ($\mathsf{ZNormal}^{\uparrow}$ root continuation trace boundary)**.** *Let*

$$Z = (T, F, V, K, R, H, C, P, N)$$

be an accepted $\mathsf{ZNormal}^{\uparrow}$ *root packet. Every continuation-trace terminal readback must pass through* R, H, C, P, N *after the visible source package and the normal-word route totality have exposed*

T, F, V, K. The boundary is the displayed root packet together with the finite continuation route; it is not a substrate halting row, divergence oracle, unrestricted evaluator, quotient normalizer, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root continuation trace boundary`

Proof. Begin with the visible source package and normal-word route totality. Definition 936.24 exposes $T, F, V, K, R, H, C, P, N$, while Definition 936.25 fixes the normal-word route through K before readback.

Now apply Definition 936.31. It pulls the normal-word row K and continuation row R onto the same transported and replayed route through H, C, P, N.

Finally use the root total-host obligation package of Definition 936.40. Since every root-facing total-host request must be discharged by the displayed packet or refused, the continuation trace boundary has no coordinate for substrate halting, divergence, unrestricted evaluation, quotient normalization, or global normalization. □

Theorem 936.44 ($\mathsf{ZNormal}^{\uparrow}$ root terminal readback ledger totality)**.** *Every terminal readback consumer at the $\mathsf{ZNormal}^{\uparrow}$ root passes through T, F, V, R, H, C, P, N before reading any downstream normal-form data. The consumer receives no divergence oracle, unrestricted evaluator, quotient computation, or substrate halting row.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal readback ledger totality`

Proof. Begin with terminal consumer totality. It forces every terminal readback to start from the typed endpoint, finite fuel, terminal-normality, and readback rows T, F, V, R.

Transport and replay then move the same terminal row through H and C, and the package and local naming rows P, N record its provenance and certificate. Thus the terminal readback ledger is total only along T, F, V, R, H, C, P, N.

Apply the no-divergence ledger and total-host refusal. Since no divergence oracle, unrestricted evaluator, quotient-computation coordinate, or substrate halting row appears in the accepted packet, no terminal readback consumer can extract one. □

Theorem 936.45 ($\mathsf{ZNormal}^{\uparrow}$ total-host terminal fuel determinacy)**.** *For every accepted $\mathsf{ZNormal}^{\uparrow}$ packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host terminal-fuel read is determined by the displayed typed endpoint row T, finite fuel row F, terminal-normality row V, and terminal read row R, transported only by H, replayed only by C, packaged only by P, and named only by N. The sibling normal-word row K is read as finite ZeckendorfNormal and ZCarry evidence and cannot be replaced by host conversion, quotient computation, unrestricted evaluation, divergence oracle, or global normalizer data.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host terminal fuel determinacy`

Proof. Project the carrier from Definition 936.1. It exposes $T, F, V, K, R, H, C, P, N$ as the whole accepted packet, so terminal fuel has no second host-side coordinate.

Use Definition 936.53 and Definition 936.14. They bind terminal use to T, F, V, R and keep the route inside the finite packet before any total-host consumer can read it.

Finally apply Definition 936.57 and Definition 936.58. The K row remains finite sibling normal-word evidence, while the refusal ledger excludes host conversion, quotient computation, unrestricted evaluation, divergence, and global normalization. □

Theorem 936.46 ($\mathsf{ZNormal}^{\uparrow}$ downstream normal route exhaustion)**.** *Every downstream normal-form route out of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

is exhausted by terminal fuel through T, F, V, R*, sibling normal-word evidence through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming through* N*. No downstream normal route may pass through host conversion, quotient computation, unrestricted evaluation, divergence oracle, or global normalizer rows.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacketDownstreamNormalRouteExhaustion`

Proof. Begin with Definition 936.56. It places every downstream normal-form consumer inside the displayed root packet and gives no separate normal-output coordinate.

Apply Definition 936.45. Terminal fuel and terminal readback are determined by T, F, V, R and the structural rows H, C, P, N.

For the sibling route, use Definition 936.57. The row K is consumed only through finite ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm surfaces. The total-host refusal theorem Definition 936.58 then removes host conversion, quotient computation, unrestricted evaluation, divergence, and global normalization from the downstream boundary. □

Theorem 936.47 ($\mathsf{ZNormal}^{\uparrow}$ finite normal-word classifier compatibility)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the finite normal-word classifier seen by a downstream consumer is compatible with the terminal route: K *may be consumed only after the typed endpoint* T*, finite fuel* F*, terminal-normality row* V*, terminal read* R*,* hsame *transport* H*,* Cont *replay* C*, package row* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N *have fixed the same packet. This compatibility exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacketFiniteNormalwordClassifierCompatibility`

Proof. Start from Definition 936.25. It identifies K as the finite normal-word route through the ZeckendorfNormal and ZCarry sibling surfaces rather than a host computation.

Use Definition 936.54 to attach that classifier route to T, F, V, R, H, C, P, N. The normal-word row cannot be detached from the terminal packet or read through an independent quotient computation.

Finally invoke Definition 936.46. Since every downstream normal route is already exhausted by the displayed packet and finite sibling surfaces, classifier compatibility contributes no route to host conversion, unrestricted evaluation, divergence detection, or global normalization. □

Theorem 936.48 ($\mathsf{ZNormal}^{\uparrow}$ downstream ZCarry dependency route)**.** *Every downstream normal-form read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

consumes the normal-word row K *through chapter 278 and chapter 544 before any terminal readback through* R, H, C, P, N*. The read therefore reaches ZeckendorfNormal and ZCarry sibling evidence as finite BEDC rows, not as host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream zcarry dependency route`

Proof. Begin with Definition 936.46. It says that every downstream normal-form route is already exhausted by $T, F, V, K, R, H, C, P, N$ and that the row K is the sibling normal-word route.

Next apply Definition 936.47. The classifier compatibility theorem forces K to remain attached to the same terminal packet before any downstream consumer reads R, H, C, P, N.

The two sibling labels name the concrete finite sources for that row. Thus K is consumed through the ZeckendorfNormal and ZCarry chapters first, and the terminal readback can only replay the already displayed route. □

Theorem 936.49 ($\mathsf{ZNormal}^{\uparrow}$ candidate-normal endpoint readback)**.** *Every candidate-mediated normal endpoint consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

must read terminal fuel through T, F, V, R, finite normal-word evidence through K, and structural rows H, C, P, N before it can feed a MetaCIC candidate-normalization or closed-normal consistency route. The endpoint readback exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacketCandidateNormalEndpointReadback`

Proof. Apply Definition 936.45. It binds the terminal fuel and terminal readback rows to T, F, V, R before any total-host endpoint can consume the accepted packet.

Next use Definition 936.46. Every downstream normal route is already exhausted by $T, F, V, R, K, H, C, P, N$, so a candidate-mediated endpoint has no independent host conversion or global-normalizer coordinate.

Finally invoke Definition 936.47. The finite normal-word row K is compatible with the same terminal route and must be read after the structural rows have fixed the packet. The downstream MetaCIC candidate-normalization consumer is therefore a later formalization site for this readback boundary, not a proof of unrestricted normalization. □

Theorem 936.50 ($\mathsf{ZNormal}^{\uparrow}$ root total-host fuel frontier)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, any total-host read that precedes a public normal output must expose the displayed rows T, F, V, R, H, C, P, N before that output is available. The typed endpoint T, finite fuel F, terminal-normality row V, terminal continuation readback R, componentwise transport H, displayed replay C, package provenance P, and local name N are therefore the root-facing fuel frontier for public normal output.*

Proof. Begin with the carrier and NameCert obligations of Definition 936.1 and Definition 936.2. They list $T, F, V, K, R, H, C, P, N$ as the complete accepted packet and bind every public read to the same local certificate surface.

Apply finite-fuel carrier admission from Definition 936.33. It places the typed endpoint, finite fuel, terminal-normality row, terminal continuation readback, and structural rows in one finite packet before any root-facing normal-form consumer can read the output.

Finally use terminal consumer totality, Definition 936.36. Since every terminal consumer is exhausted by the displayed packet, the public normal output cannot precede the rows T, F, V, R, H, C, P, N or obtain a total-host witness from outside them. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootTotalHostFuelFrontier`

Theorem 936.51 ($\mathsf{ZNormal}^{\uparrow}$ root sibling route separation)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the row* K *is only the finite Zeckendorf and* $\mathsf{ZCarry}^{\uparrow}$ *sibling route used for normal-word evidence. It cannot become host conversion, quotient normalization, unrestricted evaluation, substrate halting, divergence detection, or a global normalizer, and it cannot replace the typed endpoint, finite fuel, terminal-normality, continuation, transport, replay, provenance, or naming rows.*

Proof. Use the NameCert obligations of Definition 936.2 to locate K inside the same accepted packet as T, F, V, R, H, C, P, N. The certificate gives K a finite sibling-normal-word role and no host-side conversion coordinate.

Next apply the continuation normal-word pullback of Definition 936.31. It pulls the normal-word row and continuation readback to the same transported route through H, C, P, N, so K remains attached to finite Zeckendorf and carry evidence.

Finally combine total-host refusal exhaustion with the no-divergence oracle ledger, Definition 936.32 and Definition 936.35. These ledgers refuse host conversion, quotient normalization, unrestricted evaluation, substrate halting, divergence detection, and global normalization, leaving K as a sibling route rather than a host conversion principle. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root sibling route separation`

Theorem 936.52 ($\mathsf{ZNormal}^{\uparrow}$ root refusal audit exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the root refusal ledger exhausts requests for divergence, host conversion, unrestricted evaluation, substrate halting, and global normalization. Any root, normal-form, or MetaCIC-facing consumer must remain inside the displayed* $T, F, V, K, R, H, C, P, N$ *route and cannot obtain one of those refused objects from a hidden total-host row.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root refusal audit exhaustion`

Proof. Start with finite-fuel admission and terminal consumer totality, Definition 936.33 and Definition 936.36. They expose the finite packet $T, F, V, K, R, H, C, P, N$ as the complete route for terminal normal-form consumers.

Use continuation normal-word pullback to bind the sibling normal-word and continuation rows. By Definition 936.31, downstream normal-word and readback consumers remain on the same transported route through H, C, P, N.

Finally apply total-host refusal exhaustion and the no-divergence oracle ledger. The refusal surface of Definition 936.32 and Definition 936.35 has already accounted for divergence, host conversion, unrestricted evaluation, substrate halting, and global normalization requests, so no hidden total-host row remains available. □

Theorem 936.53 ($\mathsf{ZNormal}^{\uparrow}$ root terminal carrier admission)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet admitted at the root terminal surface is a finite carrier*

$$Z = (T, F, V, K, R, H, C, P, N)$$

whose rows expose typed endpoint evidence, finite fuel, terminal-normality, finite sibling normal-word evidence, terminal continuation readback, componentwise transport, displayed replay, provenance, and local naming. The admission route is the carrier and $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *route of Definition 936.1 and Definition 936.2; it introduces no host conversion row, quotient normalization row, unrestricted evaluator, divergence detector, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal carrier admission`

Proof. Project the carrier using Definition 936.1. Its displayed rows are exactly $T, F, V, K, R, H, C, P, N$, so root admission has no extra coordinate from which a host conversion or quotient normalizer could be read.

Apply Definition 936.2. The local certificate binds typed endpoint readback, finite fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, provenance, and naming to the same finite packet.

Finally use Definition 936.14. The terminal root interface keeps every downstream terminal read inside the displayed rows and refuses host conversion, quotient computation, unrestricted evaluation, divergence detection, and global normalization. □

Theorem 936.54 ($\mathsf{ZNormal}^{\uparrow}$ root normal-word route exposure)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the finite normal-word row K is exposed at the root only through the sibling normal-word and continuation routes already certified by the packet. A consumer that reads K must pass through the terminal rows T, F, V, the continuation read R, and the transport, replay, provenance, and naming rows H, C, P, N; it receives no quotient normalization, host conversion, unrestricted evaluation, divergence detector, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root normal word route exposure`

Proof. Start with Definition 936.25. It places the normal-word row on the displayed root route and treats K as finite sibling evidence rather than as an evaluator.

Next apply Definition 936.31. The normal-word row and continuation read are pulled back to the same transported finite packet through H, C, P, N, so K cannot be detached from T, F, V, R.

Use Definition 936.14 to close the escape boundary. Since all visible reads remain inside $T, F, V, K, R, H, C, P, N$, the root route exposes no host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization. □

Theorem 936.55 ($\mathsf{ZNormal}^{\uparrow}$ root total-host refusal ledger)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root terminal ledger refuses exactly the total-host objects absent from the carrier: host conversion, quotient normalization, unrestricted evaluation, divergence detection, and global normalization. The refusal is witnessed by carrier admission, normal-word route exposure, terminal readback exactness, continuation terminal routing, and the displayed rows $T, F, V, K, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host refusal ledger`

Proof. First use root terminal carrier admission above. It identifies the complete finite packet surface and shows that no total-host escape coordinate is present in the carrier.

Next use root normal-word route exposure together with Definition 936.8. The normal-word and terminal readback rows are finite route rows of the same packet, not quotient-normalization or host-conversion outputs.

Finally apply Definition 936.36 and Definition 936.32. These theorems exhaust root terminal consumers by $T, F, V, K, R, H, C, P, N$, so unrestricted evaluation, divergence detection, and global normalization have no public row in the ledger. □

Theorem 936.56 ($\mathsf{ZNormal}^{\uparrow}$ root downstream totality)**.** *Every downstream normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the root carrier rows*

$$T, F, V, K, R, H, C, P, N.$$

The consumer may read terminal normality through T, F, V, R*, finite normal-word evidence through* K*, transport through* H*, continuation replay through* C*, package provenance through* P*, and local naming through* N*; it receives no normal-form row outside that displayed packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream totality`

Proof. Project the carrier from Definition 936.1. It displays exactly $T, F, V, K, R, H, C, P, N$ as the accepted packet and gives a downstream consumer no separate carrier coordinate.

Use Definition 936.2 to bind those rows to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface. Then apply Definition 936.24, which places every normal-output read after the same visible source package has been exposed.

The routing clauses are supplied by Definition 936.25 and Definition 936.26. The first theorem fixes the finite normal-word route through K, and the second fixes continuation-terminal reads through T, F, V, R, H, C, P, N. Hence every downstream normal-form consumer factors through the displayed packet. □

Theorem 936.57 ($\mathsf{ZNormal}^{\uparrow}$ root sibling normal-word consumption)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the* K *row is consumed only through the finite sibling surfaces of chapter 278, chapter 544, and chapter 683. No downstream normal-form consumer may read* K *as host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootSiblingNormalWordConsumption`

Proof. Begin with Definition 936.1. The row K appears inside the same finite packet as T, F, V, R, H, C, P, N and is not a separate host-facing row.

Apply Definition 936.25. It routes the normal-word evidence through the ZeckendorfNormal and ZCarry sibling surfaces before any root consumer reads the local K row.

Next use Definition 936.26. Continuation-trace normal-form reads remain terminal continuation reads through T, F, V, R, H, C, P, N, so the same downstream consumer cannot detach K from the terminal route.

Finally apply Definition 936.56. Since every downstream normal-form consumer factors through the displayed packet, the sibling normal-word row is consumed only through those finite sibling surfaces and not through any excluded host or global-normalization object. □

Theorem 936.58 ($\mathsf{ZNormal}^{\uparrow}$ root total-host refusal)**.** *The root interface of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exports no host conversion, quotient normalizer, unrestricted evaluator, divergence oracle, substrate-halting row, or global normalizer. Any total-host or downstream normal-form use must remain inside*

$$T, F, V, K, R, H, C, P, N$$

and the finite sibling consumption route of Definition 936.57.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootTotalHostRefusal`

Proof. Project the visible source package using Definition 936.24. It exposes the typed endpoint, finite fuel, terminal-normality, normal-word, continuation, transport, replay, package, and naming rows before any normal-output consumer reads the packet.

Use Definition 936.56 to cover downstream normal-form consumers, and use Definition 936.57 to cover the finite sibling route through K.

It remains to exclude total-host escape. Apply Definition 936.26; continuation reads are terminal route reads through the displayed packet and expose no unrestricted evaluator, divergence oracle, host conversion, quotient normalizer, substrate-halting row, or global normalizer. Therefore the root interface refuses exactly those host and global-normalization objects. □

Theorem 936.59 ($\mathsf{ZNormal}^{\uparrow}$ total-host readback factorization)**.** *Every total-host readback of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the root packet*

$$T, F, V, K, R, H, C, P, N$$

and through the finite sibling route of Definition 936.57. In particular, a total-host consumer may read terminal normality, finite normal-word evidence, transport, continuation replay, provenance, and local naming only by first passing through the displayed carrier rows and then through the ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm sibling surfaces where the normal-word route exposes them.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host readback factorization`

Proof. Project the accepted carrier from Definition 936.1. It lists $T, F, V, K, R, H, C, P, N$ as the complete root packet, so a total-host readback has no separate coordinate from which to obtain normal-form data.

Apply Definition 936.56. This places every downstream normal-form consumer inside the displayed packet. Then apply Definition 936.57; the only consumption path for K is the finite sibling route through ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm surfaces.

Finally use Definition 936.58. Since the root interface refuses host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, and global normalization, the total-host readback can only be the packet-and-sibling factorization just described. □

Theorem 936.60 ($\mathsf{ZNormal}^{\uparrow}$ normal-form consumer boundary)**.** *Every normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet receives only terminal normality through* T, V, R*, finite normal-word evidence through* K*, componentwise transport through* H*, continuation replay through* C*, provenance through* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *rows through* N*. The consumer receives no host conversion row, quotient normalizer, unrestricted evaluator, divergence detector, substrate-halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal form consumer boundary`
↪ variant `BEDC.Derived.ZnormalUp.ZnormalPacket root normal form consumer boundary`

Proof. Start from Definition 936.1. The carrier exposes the terminal, word, transport, replay, provenance, and naming rows as coordinates of the same finite packet and lists no host-facing conversion or global-normalization row.

Use Definition 936.56 to bind any normal-form consumer to that packet. The terminal and continuation readbacks are then exhausted by Definition 936.26, so terminal normality and continuation replay remain inside T, V, R, H, C, P, N.

For finite normal-word evidence, apply Definition 936.57. It keeps K on the finite sibling route. The refusal clause of Definition 936.58 excludes every listed host and global-normalization object, giving exactly the stated consumer boundary. □

Theorem 936.61 ($\mathsf{ZNormal}^{\uparrow}$ sibling-refusal exhaustion)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm sibling surfaces exhaust the permitted normal-word route. Outside those sibling surfaces and the root packet*

$$T, F, V, K, R, H, C, P, N,$$

the interface excludes host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, and global normalization.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootSiblingRefusalExhaustion`

Proof. Project the carrier with Definition 936.1. The normal-word coordinate K is one row of the finite root packet, and the carrier supplies no independent route for host conversion or global normalization.

Apply Definition 936.57. It identifies the permitted normal-word route with the ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm sibling surfaces and denies the listed host and global-normalization readings of K.

It remains to see that no total-host path reintroduces an excluded surface. Use Definition 936.56 to confine downstream consumers to the displayed packet, and then use Definition 936.58 to refuse host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, and global normalization. These clauses exhaust the sibling route and the refusal boundary together. □

Theorem 936.62 ($\mathsf{ZNormal}^{\uparrow}$ root carrier obligation completion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

root-readiness carrier admission is exhausted by the displayed typed endpoint row T, finite fuel row F, terminal-normality row V, sibling normal-word row K, terminal readback row R, componentwise hsame *row H, displayed* Cont *row C, package row P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N. No root-readiness consumer receives carrier data for host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, total host computation, or global normalization outside that packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootCarrierObligationCompletion`

Proof. Project the carrier using Definition 936.1 and the root carrier exposure of Definition 936.15. These rows identify $Z = (T, F, V, K, R, H, C, P, N)$ as the complete root-facing packet.

Apply Definition 936.2 to read the same coordinates as the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface. The structural rows H, C, P, N provide transport, replay, provenance, and naming; they do not add a host-side computation coordinate.

Finally use Definition 936.58. Since every total-host request remains inside the displayed packet and the finite sibling route, the carrier obligation is complete for root readiness and excludes the listed host and global-normalization objects. □

Theorem 936.63 ($\mathsf{ZNormal}^{\uparrow}$ terminal route obligation completion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal-route obligation for root readiness is the finite route from T through F and V to R, transported by H, replayed by C, packaged by P, and named by N. The sibling normal-word row K may be consumed only as the finite Zeckendorf, carry, and continuation-facing evidence specified by the root sibling route; it is not a host conversion, quotient normalizer, unrestricted evaluator, divergence detector, substrate-halting row, total host-computation row, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalRouteObligationCompletion`

Proof. Use Definition 936.16 and Definition 936.53. They place the terminal root read on the route $T \to F \to V \to R$ inside the displayed carrier.

Next apply Definition 936.57. The K row is consumed only through the finite sibling surfaces of the Zeckendorf normal, carry, and continuation-trace normal-form packets.

The terminal route is then exhausted by Definition 936.26 and Definition 936.55. These results keep terminal reads inside $T, F, V, K, R, H, C, P, N$ and refuse the excluded host and global-normalization rows. □

Theorem 936.64 ($\mathsf{ZNormal}^{\uparrow}$ downstream readiness obligation package)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the downstream normal-form readiness package consists exactly of carrier admission for Z, terminal-route exposure through T, F, V, R, H, C, P, N, sibling normal-word consumption through K, total-host refusal, and downstream normal-form totality over the displayed rows. Every downstream normal-form consumer obtains a finite BEDC route through that package and no route to host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, total host computation, or global normalization.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalDownstreamReadinessObligationPackage`

Proof. Start with carrier completion above. It fixes the root-readiness carrier as the finite packet $Z = (T, F, V, K, R, H, C, P, N)$ and leaves no extra host-facing coordinate.

Use terminal route obligation completion to bind terminal normality to T, F, V, R, H, C, P, N and to keep K as finite sibling normal-word evidence.

Finally apply Definition 936.56 and Definition 936.58. The first theorem gives every downstream normal-form consumer a route through the displayed packet, while the second refuses total-host escape. Hence the downstream readiness package is the finite root route itself. □

Theorem 936.65 ($\mathsf{ZNormal}^{\uparrow}$ root-readiness terminal-refusal closure)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

whose carrier admission, terminal route, sibling normal-word row, downstream totality, and total-host refusal rows are displayed, the root-readiness terminal-refusal surface is closed by the finite packet Z. Downstream normal-form consumers receive only the finite terminal route and receive no host conversion, quotient normalization, unrestricted evaluation, substrate halting, divergence detector, total host computation, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootReadinessTerminalRefusalClosure`

Proof. First use Definition 936.62. It fixes the carrier admission surface as the displayed rows $T, F, V, K, R, H, C, P, N$ and excludes hidden host-facing coordinates.

Next apply Definition 936.63. The terminal route is the finite route from T through F and V to R, transported and replayed by H, C, P, N, with K consumed only as sibling normal-word evidence.

Finally combine Definition 936.64, Definition 936.58, and Definition 936.56. These rows give downstream consumers their finite route and refuse every total-host or global-normalization escape listed in the statement. □

Theorem 936.66 ($\mathsf{ZNormal}^{\uparrow}$ root BHist source surface)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root source rows are one finite BHist *surface: typed endpoint* T*, finite fuel* F*, terminal normality* V*, finite sibling normal-word row* K*, terminal continuation read* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*. A root consumer has no second source row outside this displayed packet.*

Proof. Project the carrier in Definition 936.1. It displays $Z = (T, F, V, K, R, H, C, P, N)$ as the accepted packet and gives each root-facing row as part of the same finite carrier.

Apply Definition 936.2. The local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface reads carrier admission, typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, terminal continuation read, transport, replay, provenance, and local naming from the same rows.

Finally use Definition 936.13 and Definition 936.29. The route and root-readiness rows keep those coordinates inside BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data, while the sibling chapters chapter 278, chapter 544, and chapter 683 supply only the finite surfaces named by K and R. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root bhist source surface`

Theorem 936.67 ($\mathsf{ZNormal}^{\uparrow}$ root classifier stability)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the classifier-facing reads through K *and* R *are transported only by the displayed* hsame *row* H *and replayed only by the displayed* Cont *row* C*. The* K *read may consume the ZeckendorfNormal and ZCarry sibling surfaces, and the* R *read may consume the ContinuationTraceNormalForm sibling surface, but neither read may be transported or replayed by a host conversion, quotient computation, unrestricted evaluation, divergence detector, total host computation, or global normalizer.*

Proof. Start from Definition 936.1 and Definition 936.2. They place K and R inside the same accepted packet as T, F, V, H, C, P, N and attach both reads to the local naming surface.

Use Definition 936.13. It identifies H as the only componentwise transport row and C as the only displayed replay row available to routes from T, F, V through K and R.

The sibling uses are then finite. The ZeckendorfNormal and ZCarry chapters bind the K classifier read, while chapter 683 binds the R terminal read. Together with Definition 936.29, these routes give classifier stability without adding any excluded host or global-normalization object. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root classifier stability`

Theorem 936.68 ($\mathsf{ZNormal}^{\uparrow}$ root ledger exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

every root consumer factors through the displayed Pkg *row* P *and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*. The root ledger permits reads of* T, F, V, K, R, H, C *only through that package-and-name boundary and cannot read host conversion, quotient computation, unrestricted evaluation, divergence, total host computation, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root ledger exhaustion`

Proof. First expose the packet by Definition 936.1. The only provenance and local naming rows are P and N, and the root source rows are the displayed coordinates T, F, V, K, R, H, C.

Apply Definition 936.2. It binds those coordinates to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ ledger and records P as the provenance package for the same finite surface.

Use Definition 936.13 to keep transport and replay inside H and C. Then apply Definition 936.29: every root-readiness consumer reads the package through P and the local name through N before it can use the sibling surfaces of chapter 278, chapter 544, and chapter 683.

Since no carrier coordinate, transport row, replay row, package row, or local name supplies host conversion, quotient computation, unrestricted evaluation, divergence, total host computation, or global normalization, the ledger is exhausted by P and N over the displayed BHist packet. □

Theorem 936.69 ($\mathsf{ZNormal}^{\uparrow}$ root total-host source packet)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host source exposed at the root is the single finite packet obtained by reading the typed reduction normal-form row T*, the finite fuel row* F*, the terminal-normality row* V*, the Zeckendorf and carry-facing normal-word row* K*, the terminal continuation-trace row* R*, the componentwise transport row* H*, the displayed continuation replay row* C*, the package provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*. The source cannot be split into a typed endpoint source, a normal-word source, a carry source, or a continuation-trace source outside that* BHist *packet.*

Proof. Project the carrier of Definition 936.1. It lists $T, F, V, K, R, H, C, P, N$ as the full accepted packet and gives no second root source row.

Use the NameCert obligation surface of Definition 936.2. The typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, package, and local naming rows are all read through one local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Apply Definition 936.66 and Definition 936.67. These keep the typed reduction normal-form row, the Zeckendorf and carry-facing row, and the continuation-trace row inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

Finally use Definition 936.68. Since the ledger has no coordinate outside $T, F, V, K, R, H, C, P, N$, the total-host source is exactly the displayed finite source packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host source packet`

Theorem 936.70 ($\mathsf{ZNormal}^{\uparrow}$ root terminal normality exposure)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the terminal-normality row* V *is readable at the root only after the finite fuel replay through* F *has been displayed and the terminal continuation evidence* R *has been admitted through the same transported route. A consumer may read terminal normality through*

$$T \longmapsto F \longmapsto V \longmapsto R$$

with transport H, replay C, provenance P, and local naming N, but it may not read V from an ambient normalizer, host conversion judgment, quotient computation, unrestricted evaluator, or divergence detector.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal normality exposure`

Proof. Start with Definition 936.6. It places terminal normality on the finite route from the typed endpoint T through finite fuel F to the continuation read R.

Apply Definition 936.8. The terminal read factors through T, F, V, R, H, C, P, N, and the sibling row K can be consumed only after V is already displayed as finite evidence.

Use Definition 936.9 to force the finite replay order before consumer use. The structural rows H, C, P, N are the only transport, replay, package, and naming routes available for that exposure.

Finally invoke Definition 936.16 and Definition 936.14. They keep terminal continuation evidence attached to the same root route and exclude ambient normalization, host conversion, quotient computation, unrestricted evaluation, and divergence detection. □

Theorem 936.71 ($\mathsf{ZNormal}^{\uparrow}$ root forbidden host conversion exclusion)**.** *No accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exports a divergence oracle, host conversion row, quotient-normalization row, unrestricted host evaluator, global normalization procedure, or substrate halting row. Every root request for such an object is exhausted by the displayed rows $T, F, V, K, R, H, C, P, N$ and by the refusal ledgers attached to the local* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surface.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root forbidden host conversion exclusion`

Proof. Read the complete packet through Definition 936.1 and Definition 936.2. The accepted carrier has only the typed endpoint, fuel, terminal normality, sibling normal-word, continuation readback, transport, replay, provenance, and local naming rows.

Apply Definition 936.10. It prevents the displayed finite route from being replaced by host conversion, quotient computation, divergence, unrestricted evaluation, or global normalization.

Next use Definition 936.55 and Definition 936.52. These root ledgers account for host conversion, quotient normalization, unrestricted evaluation, substrate halting, divergence detection, and global normalization requests without adding any exported row for them.

Finally combine Definition 936.68 with Definition 936.73. The root source remains $T, F, V, K, R, H, C, P, N$, so no divergence oracle, host conversion row, or global normalization procedure is available outside the finite packet. □

Theorem 936.72 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel readback frontier)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

every terminal fuel read factors through the displayed typed endpoint T, finite fuel row F, terminal-normality row V, continuation readback row R, componentwise transport H, displayed route C, provenance P, and local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N. The sibling normal-word row K may be consulted only as finite sibling evidence after V is displayed.*

The frontier exports no divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalizer, or total-host claim outside $T, F, V, K, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel readback frontier`

Proof. Project the carrier of Definition 936.1. Its terminal-facing coordinates are the typed endpoint, finite fuel, terminal-normality, continuation readback, transport, route, provenance, and local naming rows.

Use Definition 936.2 to bind those coordinates to the local certificate surface. The finite fuel row and terminal-normality row are therefore read before any terminal continuation row is consumed.

Apply Definition 936.6, Definition 936.8, and Definition 936.9. These results force the terminal read through T, F, V, R, H, C, P, N and keep K as a finite sibling row.

Finally use Definition 936.11 for the only allowed K consumption. The carrier has no coordinate for divergence, host conversion, quotient computation, unrestricted evaluation, global normalization, or total-host escape, so the displayed frontier is exhaustive. □

Theorem 936.73 ($\mathsf{ZNormal}^{\uparrow}$ total-host frontier lock)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host boundary is locked to the finite rows $T, F, V, K, R, H, C, P, N$*. A consumer may not reread that boundary as a divergence row, host conversion row, quotient computation, unrestricted evaluator, global normalization procedure, or substrate halting statement.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host frontier lock`

Proof. Begin with Definition 936.1 and Definition 936.2. Together they list the complete accepted packet and attach every public read to the local $\mathsf{NameCert}$ surface.

Apply Definition 936.7. It identifies the packet as a finite interface for the total-host side and refuses host Turing completeness or substrate halting content beyond the accepted rows.

Now use Definition 936.6 and Definition 936.8. Terminal normality is read through finite fuel and continuation readback, not through a host conversion or unrestricted evaluator.

The route admission theorem, Definition 936.9, keeps the readback route inside H, C, P, N, while Definition 936.11 keeps the sibling row finite. Thus no excluded total-host or quotient object can be recovered from the frontier. □

Theorem 936.74 ($\mathsf{ZNormal}^{\uparrow}$ downstream consumer boundary)**.** *Every downstream consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors only through the typed endpoint* T*, finite fuel* F*, terminal normality* V*, sibling normal-word handoff* K*, continuation readback* R*, componentwise transport* H*, displayed route* C*, provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*.*

Consequently downstream normal-form consumers receive the finite terminal frontier and no divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalizer, or total-host computation row.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream consumer boundary`

Proof. First expose the finite packet with Definition 936.1. The downstream-facing rows are exactly $T, F, V, K, R, H, C, P, N$.

Next bind the local certificate. By Definition 936.2, carrier admission, typed endpoint readback, fuel, terminal normality, sibling handoff, continuation readback, transport, route, provenance, and naming all belong to that same finite surface.

Use Definition 936.72 to force terminal fuel readback through T, F, V, R, H, C, P, N. Then use Definition 936.11 to identify the only allowed K consumption as finite sibling normal-word evidence.

Finally apply Definition 936.73. Since the locked frontier refuses divergence, host conversion, quotient computation, unrestricted evaluation, global normalization, and total-host computation, every downstream consumer remains inside the displayed terminal-fuel boundary. □

Theorem 936.75 ($\mathsf{ZNormal}^{\uparrow}$ TasteGate scope binding)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the finite rows $T, F, V, K, R, H, C, P, N$ *are the exact field surface later consumed by* `Nontrivial ZNormalUp` *and* `FieldFaithful ZNormalUp`*. The TasteGate-facing surface reads typed endpoint, finite fuel, terminal normality, finite sibling normal-word evidence, terminal continuation, componentwise transport, displayed continuation, package provenance, and local NameCert rows, and it exports no host conversion, quotient computation, unrestricted evaluation, divergence oracle, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket tastegate scope binding`

Proof. Begin with finite-fuel carrier admission and the local NameCert obligations. They expose exactly $T, F, V, K, R, H, C, P, N$ as one finite BHist packet and bind the same rows to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Apply the displayed TasteGate contract row through the normal-form package. The consumer receives the terminal and sibling evidence already routed through the packet, so the later Nontrivial and FieldFaithful instances consume the same field surface rather than a host-side computation.

Finally use the fuel non-escape and total-host refusal rows. Since the packet contains no coordinate for host conversion, quotient computation, unrestricted evaluation, divergence, or global normalization, the TasteGate scope is exactly the displayed finite field surface. □

Theorem 936.76 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel field scope)**.** *Every terminal-normality read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the finite chain*

$$T \to F \to V \to R,$$

with K *available only as finite Zeckendorf and carry-facing sibling evidence. The structural rows* H, C, P, N *transport, replay, package, and name this chain; they do not replace it by host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel field scope`

Proof. Use Definition 936.34. It places every continuation read behind the typed endpoint T, finite fuel F, and terminal-normality row V in the same packet.

Next apply terminal readback exactness and sibling normal-word handoff. The readback row is R, while K remains finite sibling evidence through the Zeckendorf and carry-facing surfaces.

The route transport theorem supplies the only structural movement. Rows H, C, P, N transport, replay, package, and name the finite chain, and fuel non-escape rules out any replacement by an omitted host or global object. □

Theorem 936.77 ($\mathsf{ZNormal}^{\uparrow}$ total-host consumer scope)**.** *No total-host consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet can read divergence, host conversion, quotient computation, unrestricted evaluation, or global normalization from the packet. Every such consumer is confined to*

$$T, F, V, K, R, H, C, P, N$$

and to the finite sibling routes named by K *and* R*.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host consumer scope`

Proof. Start with the total-host refusal exhaustion theorem. It already confines total-host requests to the displayed packet and the finite sibling surfaces cited by the normal-word and continuation rows.

Use the terminal fuel field scope above to isolate the terminal-normality chain. Then use the continuation normal-word pullback to keep K and R in one transported, replayed, packaged, and named packet.

Finally apply the no-divergence oracle ledger and root non-escape ledger. Together they exhaust all consumer reads through the displayed rows, so none of the excluded host, quotient, evaluator, divergence, or global-normalization objects is available. □

Theorem 936.78 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel row exhaustion)**.** *Every terminal-normality read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet is exhausted by the rows*

$$T, F, V, R, H, C, P, N.$$

The row K contributes only the finite Zeckendorf and carry-facing sibling normal-word surface already available to the packet, and the TasteGate-facing consumer surface is the same field surface later used by `Nontrivial ZNormalUp` *and* `FieldFaithful ZNormalUp`*.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel row exhaustion`

Proof. Apply Definition 936.76. It places each terminal-normality read behind T, F, V, R and uses H, C, P, N only as transport, replay, package, and naming rows.

Next use Definition 936.77. A consumer that asks for host conversion, quotient computation, divergence, unrestricted evaluation, or global normalization receives no additional row.

The remaining sibling evidence is exactly the finite handoff of Definition 936.11. Thus the TasteGate boundary seen by later Nontrivial and FieldFaithful instances is the displayed finite surface, not a host evaluator. □

Theorem 936.79 ($\mathsf{ZNormal}^{\uparrow}$ normal-word sibling exactness)**.** *The normal-word row K of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet is exact only through the Zeckendorf and carry-facing sibling routes cited by Definition 936.11. Together with the continuation readback row R, it is transported through H, C, P, N and not through host conversion, quotient computation, divergence, unrestricted evaluation, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal word sibling exactness`

Proof. Start with Definition 936.11. It identifies K with the finite Zeckendorf and carry-facing sibling normal-word evidence.

Now apply Definition 936.31. This places K and R on one transported route and permits only the structural rows H, C, P, N to move that route across the packet.

Finally use Definition 936.77. Since the packet exports no host or quotient-facing normalizer, the exactness of K is exactly the sibling exactness already named in the finite rows. □

Theorem 936.80 ($\mathsf{ZNormal}^{\uparrow}$ total-host refusal package)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the public refusal package excludes host conversion, quotient computation, divergence oracle, unrestricted evaluator, and global normalizer. All admitted reads factor through*

$$T, F, V, K, R, H, C, P, N,$$

with terminal-normality routed by T, F, V, R, H, C, P, N and sibling normal-word evidence routed by K through the Zeckendorf and carry-facing sibling surface.

Proof. First combine Definition 936.78 with Definition 936.79. These rows separate the terminal-normality route from the sibling normal-word route while keeping both inside the same finite packet.

Next apply Definition 936.77. It refuses every total-host request whose requested object has no coordinate in $T, F, V, K, R, H, C, P, N$.

The TasteGate surface is therefore explicit: later Nontrivial and FieldFaithful instances consume the same finite field surface, and no hidden host conversion, quotient computation, divergence oracle, unrestricted evaluator, or global normalizer is packaged. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host refusal package`

Theorem 936.81 ($\mathsf{ZNormal}^{\uparrow}$ root TasteGate field surface)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the TasteGate-facing field surface is exactly the displayed tuple $T, F, V, K, R, H, C, P, N$. The surface reads terminal normality and sibling normal-word evidence only through those rows; it cannot collapse into host conversion, quotient normalization, unrestricted evaluation, a divergence oracle, or a global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZNormalTasteGate single carrier alignment`

Proof. Project the carrier and local NameCert obligations. They expose $T, F, V, K, R, H, C, P, N$ as the complete field inventory and leave no additional coordinate for a host-side converter or quotient normalizer.

Use the TasteGate scope binding and total-host refusal package. The first binds the future Nontrivial and FieldFaithful read to the same tuple, while the second refuses every consumer request whose object is not carried by the tuple.

The field surface is therefore exact. Any attempted collapse to host conversion or quotient normalization would require a coordinate absent from the carrier, so the TasteGate-facing read remains the finite BEDC packet surface. □

Theorem 936.82 ($\mathsf{ZNormal}^{\uparrow}$ terminal normality scope package)**.** *Every terminal-normality read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through*

$$T, F, V, R, H, C, P, N,$$

with K available only as finite sibling evidence through the Zeckendorf and carry-facing normal-word surface. No terminal-normality consumer can replace this route by host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal normality scope package`

Proof. Begin with terminal fuel row exhaustion. It places each terminal-normality read behind T, F, V, R and makes H, C, P, N the only transport, replay, package, and naming rows.

Next read the sibling evidence. Normal-word sibling exactness keeps K on the finite Zeckendorf and carry-facing route, so K supports the terminal read only as sibling evidence and not as a host evaluator.

Finally apply the root TasteGate field surface. The terminal-normality consumer receives the same finite tuple surface, and the excluded host or quotient objects have no field from which they could be read. □

Theorem 936.83 ($\mathsf{ZNormal}^{\uparrow}$ normal-word continuation scope lock)**.** *Before any downstream normal-form consumer reads an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the normal-word row* K *and continuation row* R *are jointly scoped through* $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, *and* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$. *The joint scope is transported, replayed, packaged, and named only through* H, C, P, N *and cannot be reopened as host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal word continuation scope lock`

Proof. Start with normal-word sibling exactness. It binds K to the finite Zeckendorf and carry-facing sibling routes, so any normal-word read is already scoped before downstream consumption.

Now apply the continuation normal-word pullback. It places K and R on one finite route through H, C, P, N, with R supplied by the continuation terminal surface and not by a detached evaluator.

Use the terminal normality scope package to close the downstream read. Since the consumer sees only the joined K, R route inside the displayed packet, no host conversion, quotient normalization, unrestricted evaluation, divergence detector, or global normalizer can reopen the scope. □

Theorem 936.84 ($\mathsf{ZNormal}^{\uparrow}$ terminal-route field exhaustion)**.** *Every terminal-route consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet must read the finite terminal-normality row, finite fuel row, transport row, provenance row, and local* $\mathsf{NameCert}$ *row from the displayed packet*

$$Z = (T, F, V, K, R, H, C, P, N).$$

The route may use the terminal continuation row R *and displayed replay row* C*, but it cannot replace the finite terminal packet by host conversion, quotient normalization, unrestricted evaluation, divergence detection, or a global normalizer.*

Proof. Start with Definition 936.53. It displays $T, F, V, K, R, H, C, P, N$ as the complete root terminal carrier and leaves no host conversion or quotient-normalization coordinate.

Use Definition 936.69 to keep every total-host source read inside the same finite packet. The source route therefore reaches terminal normality only through the typed endpoint, fuel, continuation, transport, replay, provenance, and naming rows already present.

Finally apply Definition 936.78. It exhausts terminal-normality consumers by T, F, V, R, H, C, P, N, with K available only as finite sibling normal-word evidence. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalRouteFieldExhaustion`

Theorem 936.85 ($\mathsf{ZNormal}^{\uparrow}$ normal-word sibling bridge)**.** *The normal-word route of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet is mediated by the ZeckendorfNormal, ZCarry, TypedReductionNormalForm, and ContinuationTraceNormalForm sibling surfaces already displayed by the packet. The row* K *is therefore a finite sibling normal-word row transported with* T, F, V, R, H, C, P, N*, not a host normalizer, quotient conversion, divergence oracle, or unrestricted evaluator.*

Proof. Apply Definition 936.79. It identifies the normal-word row K with the finite Zeckendorf and carry-facing sibling routes and keeps that row attached to the continuation readback surface.

Next use Definition 936.53. The typed endpoint, fuel, terminal-normality, continuation, transport, replay, provenance, and naming rows are displayed in the same root carrier, so the sibling normal-word route cannot detach from the terminal packet.

Finally apply Definition 936.69. The total-host source is one finite packet, which rules out any separate host normalizer, quotient conversion, divergence oracle, or unrestricted evaluator behind the sibling bridge. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalNormalWordSiblingBridge`

Theorem 936.86 ($\mathsf{ZNormal}^{\uparrow}$ total-host refusal readback)**.** *The same accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet that exposes the terminal route refuses readback of divergence, host conversion, quotient normalization, unrestricted evaluation, and global normalization. All terminal and normal-word reads remain inside*

$$T, F, V, K, R, H, C, P, N,$$

with no total-host row outside the finite packet.

Proof. First combine Definition 936.84 with Definition 936.85. Terminal and normal-word consumers are both confined to the displayed packet and its finite sibling routes.

Use Definition 936.69. It states that the total-host source is the single finite packet, not a separately readable host conversion or divergence object.

Finally apply Definition 936.78 and Definition 936.79. These rows account for all admitted terminal and sibling reads, leaving no coordinate for the refused total-host readbacks. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostRefusalReadback`

Theorem 936.87 ($\mathsf{ZNormal}^{\uparrow}$ root total-host readiness)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root total-host consumer may read only the typed endpoint row T*, finite fuel row* F*, terminal-normality row* V*, sibling normal-word row* K*, continuation readback row* R*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*. The consumer receives no host conversion row, quotient normalization row, divergence oracle, unrestricted evaluation row, or global-normalization coordinate.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host readiness`

Proof. Project the carrier of Definition 936.1. The displayed packet $Z = (T, F, V, K, R, H, C, P, N)$ lists every root-readable coordinate of an accepted $\mathsf{ZNormal}^{\uparrow}$ packet.

Apply the NameCert obligations of Definition 936.2. They bind carrier admission, typed endpoint readback, finite fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, provenance, and local naming into one finite surface.

Since the carrier and the NameCert surface contain no host conversion, quotient normalization, divergence, unrestricted evaluation, or global-normalization row, a total-host consumer can only read the displayed packet. □

Theorem 936.88 ($\mathsf{ZNormal}^{\uparrow}$ root kernel source totality)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root kernel source is total over the displayed packet rows: typed endpoint T*, finite fuel* F*, terminal normality* V*, sibling normal-word evidence* K*, continuation readback* R*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming* N*. No kernel consumer can source the same normal-form read from host conversion, quotient normalization, divergence detection, unrestricted evaluation, or a global normalizer outside those rows.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootKernelSourceTotality`

Proof. Start from the carrier of Definition 936.1. Its finite BHist packet supplies the complete source tuple and does not expose any second kernel-source coordinate.

Use Definition 936.2 to read the same rows through the local naming surface. The typed endpoint, fuel, terminal row, sibling normal-word row, continuation readback, transport, replay, package, and naming obligations are all discharged inside Z.

The non-escape conclusion is constructive projection: every kernel consumer obtains its source by selecting one of the displayed rows and following the structural route through H, C, P, N. The excluded host and quotient objects are absent from the packet, so they cannot serve as kernel sources. □

Theorem 936.89 ($\mathsf{ZNormal}^{\uparrow}$ root normal-form consumer exhaustion)**.** *Every root normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

is exhausted by typed endpoint T, finite fuel F, terminal-normality row V, sibling normal-word row K, continuation readback R, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N. The normal-form read has no remaining route to host conversion, quotient normalization, a divergence oracle, unrestricted evaluation, or global normalization.*

Proof. First expose the displayed packet with Definition 936.1. It gives the normal-form consumer exactly the nine rows $T, F, V, K, R, H, C, P, N$.

Then apply Definition 936.2. The local certificate forces endpoint, fuel, terminal-normality, sibling normal-word, continuation readback, transport, replay, provenance, and naming to be consumed as one finite route.

It remains to rule out escape. Because neither the carrier nor the NameCert obligation surface contains host conversion, quotient normalization, divergence, unrestricted evaluation, or global normalization, no normal-form consumer can continue past the displayed route. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root normal form consumer exhaustion`

Theorem 936.90 ($\mathsf{ZNormal}^{\uparrow}$ terminal row constructor exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal row constructor surface is exhausted by typed endpoint T, finite fuel F, terminal-normality V, sibling normal-word evidence K, terminal readback R, componentwise hsame *transport H, displayed* Cont *replay C, package provenance P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N. It constructs no host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization row.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalRowConstructorExhaustion`

Proof. Project the carrier of Definition 936.1. The accepted packet lists exactly $T, F, V, K, R, H, C, P, N$ as the root terminal constructor inventory.

Apply Definition 936.2. Carrier admission, typed endpoint readback, finite fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, provenance, and local naming are the displayed NameCert obligations for the same packet.

Use Definition 936.8 and Definition 936.11. Terminal readback is routed through the finite terminal rows, and K is only the finite sibling normal-word handoff.

Finally apply Definition 936.81. The TasteGate-facing field surface is the same packet surface, so no constructor row can be reinterpreted as host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization. □

Theorem 936.91 ($\mathsf{ZNormal}^{\uparrow}$ fuel replay nondivergence boundary)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the fuel replay boundary is the finite path from T through F to V and R, transported and replayed by H, C, P, N, with K available only as finite sibling normal-word evidence. This boundary records nondivergence of the displayed replay route without exporting divergence detection, unrestricted evaluation, host conversion, quotient normalization, or global normalization.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalFuelReplayNondivergenceBoundary`

Proof. Begin with Definition 936.1 and Definition 936.2. Together they expose the finite fuel row F beside the typed endpoint, terminal-normality, continuation, transport, replay, provenance, and local naming rows.

Use Definition 936.8. The replayed terminal read is the displayed route through T, F, V, R, H, C, P, N, not a host evaluator.

Attach the sibling side by Definition 936.11. The row K remains finite normal-word evidence and does not become a quotient normalizer or a global normalizer.

Finally apply Definition 936.81. Since the TasteGate-facing packet has exactly the rows $T, F, V, K, R, H, C, P, N$, the fuel replay boundary can certify only the displayed nondivergence route and cannot decide arbitrary divergence or unrestricted evaluation. □

Theorem 936.92 ($\mathsf{ZNormal}^{\uparrow}$ total host consumer factorization)**.** *Every total-host consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the finite packet*

$$Z = (T, F, V, K, R, H, C, P, N).$$

The factorization first reads typed endpoint, fuel, terminal-normality, sibling normal-word, and terminal readback rows, and then moves only through H, C, P, N. It has no route to host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization.

Proof. Project the carrier and local NameCert surface using Definition 936.1 and Definition 936.2. These rows give the complete consumer inventory and bind it to one local naming certificate.

Apply Definition 936.8. Any terminal read of the consumer must pass through the finite terminal route, including R and the structural rows H, C, P, N.

Use Definition 936.11 to account for K. The normal-word component is the sibling handoff surface, not an additional host normalization channel.

Finally use Definition 936.81. The TasteGate-facing field surface is already the same tuple, so a total-host consumer factors through the packet and cannot extract host conversion, quotient normalization, unrestricted evaluation, divergence detection, or global normalization. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host consumer factorization`

Theorem 936.93 ($\mathsf{ZNormal}^{\uparrow}$ root-readiness bridge source)**.** *Every root-readiness consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the finite sibling route consisting of* $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$, *and the packet rows*

$$T, F, V, K, R, H, C, P, N.$$

The consumer reads the root source as one BHist *packet with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *naming; it has no additional bridge source outside the finite sibling route.*

Proof. Project the accepted packet through Definition 936.66. This exposes $T, F, V, K, R, H, C, P, N$ as one root BHist source rather than as detached endpoint, word, carry, or continuation rows.

Use Definition 936.29 to make the local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ surface the only root-readiness interface. The same surface carries H, C, P, N, so transport, replay, package provenance, and local naming stay attached to the displayed packet.

Finally read the sibling route through the finite consumers already named by K and R. The K row reaches $\mathsf{ZeckendorfNormal}^\uparrow$ and $\mathsf{ZCarry}^\uparrow$, while the R row reaches $\mathsf{ContinuationTraceNormalForm}^\uparrow$; no extra root bridge source is introduced beyond these sibling surfaces and the displayed packet rows. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readiness bridge source`

Theorem 936.94 ($\mathsf{ZNormal}^\uparrow$ root-readiness terminal non-escape)**.** *Every terminal read exposed by the root-readiness surface of an accepted* $\mathsf{ZNormal}^\uparrow$ *packet remains inside*

$$T, F, V, K, R, H, C, P, N.$$

It exposes no host conversion row, quotient normalizer, unrestricted evaluator, divergence oracle, or global normalization row.

Proof. Start from Definition 936.14. It binds terminal use to the finite root-facing interface and keeps the terminal route on the displayed packet rows.

Apply Definition 936.58. This refuses host conversion, quotient normalization, unrestricted evaluation, divergence detection, and global normalization for every total-host request that tries to leave the finite root route.

The terminal read is therefore a non-escaping readback through $T, F, V, K, R, H, C, P, N$. Since neither the terminal interface nor the total-host refusal theorem supplies an extra host or global-normalization coordinate, no listed excluded row is exported. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readiness terminal nonescape`

Theorem 936.95 ($\mathsf{ZNormal}^\uparrow$ root-readiness NameCert package)**.** *The root-readiness package consumed downstream by* $\mathsf{ZNormal}^\uparrow$ *consists exactly of the finite bridge-source row of Definition 936.93, the terminal non-escape row of Definition 936.94, and the existing local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *surface. Downstream normal-form consumers receive that package as* $T, F, V, K, R, H, C, P, N$ *with the finite sibling route through* $\mathsf{ZeckendorfNormal}^\uparrow$*,* $\mathsf{ZCarry}^\uparrow$*, and* $\mathsf{ContinuationTraceNormalForm}^\uparrow$*; they do not receive a separate total-host or global-normalization package.*

Proof. First combine Definition 936.93 with Definition 936.29. The bridge-source theorem supplies the finite sibling route and the displayed packet rows, while the NameCert readiness theorem supplies the local naming surface over the same rows.

Next use Definition 936.94. It keeps terminal reads inside the same package and refuses host conversion, quotient normalization, unrestricted evaluation, divergence detection, and global normalization.

Thus every downstream normal-form consumer receives one package: the finite sibling bridge source, the terminal non-escape boundary, and the local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ surface. There is no remaining coordinate from which to build a separate total-host package or a global-normalization package. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readiness namecert package`

Theorem 936.96 ($\mathsf{ZNormal}^{\uparrow}$ total-host terminal carrier admission)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

admits a total-host terminal carrier only through the displayed terminal normality row V*, bounded fuel replay row* F*, typed endpoint row* T*, and terminal continuation row* R*. The TasteGate-facing structural rows* H, C, P, N *transport, replay, package, and name this same carrier, while* K *remains finite sibling normal-word evidence. The carrier admits no host conversion row, quotient normalization row, unrestricted evaluator, divergence oracle, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host terminal carrier admission`

Proof. Project the carrier and its local obligations, Definition 936.1 and Definition 936.2. They expose exactly $T, F, V, K, R, H, C, P, N$ as the finite packet and make terminal normality a displayed row rather than a host-side conversion judgment.

Use the root terminal carrier and terminal non-escape surfaces, Definition 936.53 and Definition 936.14. These place terminal use on T, F, V, R and keep the sibling normal-word row K inside the same accepted packet.

Finally apply the TasteGate scope binding, Definition 936.75. The later TasteGate consumer sees the same finite field surface, so the admitted carrier cannot acquire any excluded host, quotient, evaluator, divergence, or global-normalization row. □

Theorem 936.97 ($\mathsf{ZNormal}^{\uparrow}$ fuel readback TasteGate route)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the bounded fuel readback route reaches TasteGate only after the typed endpoint* T*, finite fuel* F*, terminal normality* V*, continuation readback* R*, and structural rows* H, C, P, N *have been displayed. The normal-word row* K *may contribute only the finite Zeckendorf and carry-facing sibling route. This TasteGate route exports no host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization.*

Proof. Start from terminal fuel field scope and terminal fuel row exhaustion, Definition 936.76 and Definition 936.78. They route fuel readback through T, F, V, R, H, C, P, N and leave K as finite sibling evidence.

Next use the root TasteGate field surface, Definition 936.81. It identifies the TasteGate-facing consumer surface with the displayed tuple $T, F, V, K, R, H, C, P, N$.

The total-host refusal package, Definition 936.80, closes the route. Since every excluded host or quotient object is absent from the packet, the fuel readback reaches TasteGate only through the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalFuelReadbackTasteGateRoute`

Theorem 936.98 ($\mathsf{ZNormal}^{\uparrow}$ downstream normal consumer non-escape)**.** *Every downstream normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *total-host packet is confined to terminal normality through* T, F, V, R*, finite sibling normal-word evidence through* K*, and the TasteGate-aligned structural rows* H, C, P, N*. The downstream route is exhausted by the displayed packet and cannot escape to host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, or global normalization.*

Proof. Begin with downstream normal route exhaustion, Definition 936.46. It accounts for terminal fuel, sibling normal-word evidence, componentwise transport, continuation replay, provenance, and local naming for every downstream normal-form consumer.

Apply the total-host terminal carrier admission above. The downstream consumer therefore reads the same terminal carrier that already passed through terminal normality, bounded fuel replay, TasteGate alignment, and the local structural rows.

Finally use the root total-host refusal theorem, Definition 936.58. It refuses host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, and global normalization for every route not carried by $T, F, V, K, R, H, C, P, N$. Hence the downstream normal consumer has no escape route beyond the displayed packet. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream normal consumer nonescape`

Theorem 936.99 ($\mathsf{ZNormal}^{\uparrow}$ terminal total-host carrier row)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal total-host carrier row is the displayed BHist *carrier read through terminal normality* V*, continuation readback* R*, componentwise* hsame *transport* H*,* Cont *replay* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row* N*. The row admits terminal normality as finite displayed data and exports no host conversion, quotient normalization, divergence oracle, or global normalization procedure.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalTotalHostCarrierRow`

Proof. Project the carrier and local certificate surface using Definition 936.1 and Definition 936.2. These rows expose $T, F, V, K, R, H, C, P, N$ as the complete accepted packet and bind the carrier to one local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Apply Definition 936.53 and Definition 936.96. They place the terminal total-host carrier on the displayed terminal route through T, F, V, R with H, C, P, N attached as transport, replay, package, and naming evidence.

Finally use Definition 936.58. Since the root refusal ledger excludes host conversion, quotient normalization, divergence detection, unrestricted evaluation, and global normalization outside the packet, the carrier row is finite displayed data rather than a host-side normalizer. □

Theorem 936.100 ($\mathsf{ZNormal}^{\uparrow}$ fuel replay classifier row)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the fuel replay classifier row reads bounded fuel* F *together with terminal normality* V*, terminal readback* R*,* hsame *transport* H*,* Cont *replay* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming* N*. The classifier may consume the sibling normal-word row* K *only as finite displayed evidence, and it supplies no unrestricted evaluator or global normalization procedure.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket fuel replay classifier row`

Proof. Begin with Definition 936.91. It identifies the bounded replay path from T through F to V and R, transported and replayed by H, C, P, N, with K remaining finite sibling normal-word evidence.

Use Definition 936.97 and Definition 936.78. These results keep the fuel readback classifier on the displayed packet rows before any TasteGate-facing consumer can read it.

The root side is fixed by Definition 936.89. It confines downstream normal-form consumers to the same finite packet, so bounded replay remains classifier data and cannot become unrestricted evaluation or global normalization. □

Theorem 936.101 ($\mathsf{ZNormal}^{\uparrow}$ divergence-oracle non-escape row)**.** *Every divergence-oracle request addressed to a root* $\mathsf{ZNormal}^{\uparrow}$ *total-host packet is blocked by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *rows of the same accepted packet. Terminal normality and bounded replay are read as finite data through* $T, F, V, K, R, H, C, P, N$*; no route escapes to a divergence oracle, substrate halting predicate, host conversion row, quotient normalization row, or global normalization procedure.*

Proof. First assemble the finite root packet using Definition 936.93. It exposes the root BHist source and attaches transport, replay, package provenance, and local naming to the same displayed fields.

Next apply Definition 936.94 and Definition 936.17. These rows keep terminal reads inside the finite normality and continuation route while refusing escape to omitted host-facing objects.

Finally use Definition 936.35 with Definition 936.58. The first ledger refuses divergence-oracle readings, and the second refuses the total-host exits listed in the statement; together they leave only terminal normality and bounded replay as finite displayed data. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalDivergenceOracleNonescapeRow`

Definition 936.102 ($\mathsf{ZNormal}^{\uparrow}$ root display packet)**.** *The root display packet of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *certificate is the finite tuple*

$$Z = (T, F, V, K, R, H, C, P, N).$$

Here T *is the typed endpoint row,* F *is the finite fuel row,* V *is the terminal-normality row,* K *is the Zeckendorf and carry-facing normal-word row,* R *is the terminal continuation-trace row,* H *is the componentwise* hsame *transport row,* C *is the displayed* Cont *replay row,* P *is the local* Pkg *provenance row, and* N *is the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming row. The packet is read as one finite* BHist *surface: every admitted root consumer must enter through these rows and their displayed* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootDisplayPacket semantic name certificate`

Theorem 936.103 ($\mathsf{ZNormal}^{\uparrow}$ root total-host bridge components)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root display packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

each total-host read factors through the typed endpoint row T*, the Zeckendorf and carry-facing normal row* K*, the terminal continuation-trace row* R*, and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *rows* P, N*. The read may use the finite fuel, terminal-normality, transport, and replay rows* F, V, H, C*, but it cannot bypass the displayed bridge components by a host conversion row, divergence oracle, quotient computation, unrestricted evaluator, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootTotalHostBridgeComponents`
↪variant `BEDC.Derived.ZnormalUp.ZnormalPacket root total host bridge components`

Proof. Start with Definition 936.102. It exposes the accepted root surface as the finite tuple $T, F, V, K, R, H, C, P, N$, so there is no second total-host coordinate outside the displayed BHist packet.

Apply Definition 936.66 and Definition 936.67. The first pins root reads to the single BHist source, and the second keeps the Zeckendorf and carry-facing normal row K and the terminal continuation-trace row R under the displayed hsame and Cont rows H, C.

The sibling chapters then supply only component surfaces. The typed endpoint route is the one named by chapter 562; the finite normal-word route is the one named by chapter 278 and chapter 544; and the terminal continuation route is the one named by chapter 683.

Finally use Definition 936.68. Since the ledger accounts for every root consumer through P and N, the total-host read has exactly the displayed bridge components and no exported host conversion, divergence, quotient, evaluator, or global-normalization row. □

Theorem 936.104 ($\mathsf{ZNormal}^{\uparrow}$ root continuation replay closure)**.** *Every accepted replay of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *root display packet is exhausted by the displayed* Cont *row* C *together with the terminal continuation-trace row* R*, transport row* H*, provenance row* P*, and local naming row* N*. The replay cannot export a divergence oracle, host conversion row, or global normalization procedure, and it cannot manufacture a continuation trace outside the finite packet* Z*.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootContReplayClosure`

Proof. Read the replay surface through Definition 936.102. The only replay coordinate is C, and it is attached to R, H, P, N inside the same finite packet.

Use Definition 936.67. It keeps the R read replayed only by C and transported only by H, with the ContinuationTraceNormalForm sibling surface supplying the terminal trace rather than an ambient evaluator.

Next apply Definition 936.69 and Definition 936.84. These force terminal and total-host consumers to pass through the displayed rows before any replay can be accepted.

Finally invoke Definition 936.68. The package and local naming ledger has no coordinate for divergence, host conversion, or global normalization, so replay closure is exactly the displayed Cont row over Z. □

Theorem 936.105 ($\mathsf{ZNormal}^{\uparrow}$ root NameCert component exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *surface of an accepted root display packet is exactly the displayed carrier, classifier, transport, replay, provenance, and naming data*

$$(T, F, V, K, R, H, C, P, N).$$

Its carrier is the finite BHist *packet* Z*; its classifier reads the typed endpoint, finite fuel, terminal normality, Zeckendorf and carry-facing normal row, and terminal continuation trace; its transport and replay rows are* H *and* C*; its provenance and local name rows are* P *and* N*. No additional carrier, classifier, transport, replay, provenance, naming, host conversion, divergence, or global-normalization component belongs to the local NameCert surface.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootNameCertComponentExhaustion`

Proof. Project the carrier from Definition 936.102 and Definition 936.1. Both descriptions display the same finite coordinates $T, F, V, K, R, H, C, P, N$ and give no external NameCert component.

Apply Definition 936.2. It binds carrier admission, classifier reads, transport, replay, provenance, and local naming to the rows already visible in the accepted packet.

Use Definition 936.67 to account for the classifier side and Definition 936.68 to account for provenance and local naming. These two results close the component list through K, R, H, C, P, N without opening a host-facing route.

The sibling packets remain inputs to the displayed rows only: chapter 562 supplies the typed endpoint surface, chapter 278 and chapter 544 supply the finite normal-word surface, and chapter 683 supplies the terminal continuation surface.

Thus the local NameCert surface is exhausted by the carrier, classifier, transport, replay, provenance, and naming rows of Z, with no additional host conversion, divergence, or global-normalization component. □

Theorem 936.106 ($\mathsf{ZNormal}^{\uparrow}$ root display packet bridge readback exactness)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root display packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

any accepted total-host or terminal-normal consumer read of Z factors through exactly the displayed typed endpoint row T, finite fuel row F, terminal-normality row V, Zeckendorf and carry-facing normal-word row K, terminal continuation-trace row R, componentwise transport row H, replay row C, provenance row P, and local name row N. The typed endpoint, normal-word, and continuation-trace sibling surfaces may be consumed only as displayed rows of this packet. No host conversion, divergence oracle, quotient computation, unrestricted evaluator, or global normalizer participates in the readback.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootDisplayPacketBridgeReadbackExactness`

Proof. Project the packet by Definition 936.102. The accepted root surface is the finite BHist display $T, F, V, K, R, H, C, P, N$, so a consumer read has no ambient coordinate apart from these rows.

Apply Definition 936.103. It identifies the typed endpoint read with T, the Zeckendorf and carry-facing normal read with K, the terminal continuation-trace read with R, and the local provenance and naming read with P, N, while keeping F, V, H, C inside the same packet.

Use Definition 936.104 for the replay coordinate. This confines every accepted continuation replay to C together with R, H, P, N, and it prevents the readback from exporting a host conversion, divergence oracle, or global-normalization procedure.

Finally invoke Definition 936.105. Its component list exhausts the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface by the carrier, classifier, transport, replay, provenance, and naming rows of Z. The sibling chapters chapter 562, chapter 278, chapter 544, and chapter 683 are therefore consumed only as displayed endpoint, normal-word, and continuation-trace surfaces, giving exact readback with no quotient computation, unrestricted evaluator, or global normalizer. □

Definition 936.107 ($\mathsf{ZNormal}^{\uparrow}$ root carrier boundary)**.** *The root carrier boundary for terminal normality is the finite packet*

$$Z = (T, F, V, K, R, H, C, P, N).$$

Here T is the terminal typed endpoint row, F is the bounded fuel row, V is the terminal-normality row, K is the total-host refusal row for host conversion, quotient normalization, divergence, unrestricted evaluation, and global normalization, R is the displayed continuation readback row, H is the componentwise hsame *transport row, C is the displayed* Cont *replay row, P is the* Pkg *provenance row, and N is the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row. A root consumer reads terminal normality only through these rows of the* BHist *packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root carrier boundary semantic namecert`

Theorem 936.108 ($\mathsf{ZNormal}^{\uparrow}$ NameCert root obligations)**.** *For the root boundary of Definition 936.107, the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission exposes exactly the finite packet $Z = (T, F, V, K, R, H, C, P, N)$;*

2. *classifier exposure reads terminal normality from V and its sibling normal-route evidence from T, F, K, R;*
3. *terminal-normality evidence is accepted only after the bounded fuel row F and the total-host refusal row K are displayed;*
4. *bounded-fuel replay is transported by H and reread only through the displayed* Cont *row C;*
5. *non-escape keeps* Pkg *provenance and local naming inside P, N and exports no host conversion row, quotient normalization row, divergence oracle, unrestricted evaluator, or global-normalization coordinate.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket namecert root obligations`

Proof. Project the root boundary packet. The carrier obligation is the finite tuple $T, F, V, K, R, H, C, P, N$, so the classifier has no second row from which to read terminal normality.

The classifier and terminal-normality obligations are the rows V together with T, F, K, R. The row F bounds the replay, K records refusal of total-host escape routes, and R keeps the terminal readback attached to the displayed continuation surface.

Transport and replay are structural rows. The componentwise hsame row H moves only the displayed coordinates, and the Cont row C rereads only the same finite packet.

Finally, P and N record provenance and local naming for the packet already exposed. Since no excluded host or quotient coordinate appears in the carrier, the non-escape obligation is discharged inside the displayed root surface. □

Theorem 936.109 ($\mathsf{ZNormal}^{\uparrow}$ root consumer readiness)**.** *Every downstream normal-route consumer that uses* $\mathsf{ZNormal}^{\uparrow}$ *must expose the root carrier boundary of Definition 936.107 before citing terminal normality. In particular, it must display the terminal typed endpoint row T, bounded fuel row F, terminal-normality row V, total-host refusal row K, continuation readback row R,* hsame *transport row H,* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root consumer readiness`

Proof. Begin with the boundary definition. A downstream consumer reaches $\mathsf{ZNormal}^{\uparrow}$ through the finite packet $Z = (T, F, V, K, R, H, C, P, N)$, and terminal normality is one row of that packet rather than an ambient host theorem.

Apply Definition 936.108. The root obligations require carrier admission, classifier exposure, terminal-normality evidence, bounded-fuel replay, and non-escape before the local name may be consumed.

Thus a normal-route consumer is ready only after it has displayed all root rows. Any attempted citation that omits F, K, R, H, C, P, N would either detach the terminal row from its bounded replay and total-host refusal surface or bypass the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ row, and so would not be an accepted $\mathsf{ZNormal}^{\uparrow}$ consumer. □

Theorem 936.110 ($\mathsf{ZNormal}^{\uparrow}$ NameCert obligation surface)**.** *For every accepted root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *obligation surface is exactly the finite display consisting of carrier admission for Z, classifier exposure through T, F, V, K, R, terminal-normality evidence through V, bounded-fuel replay through F, H, C, and non-escape through P, N. The packet admits no additional carrier row for host conversion, quotient computation, unrestricted evaluation, divergence, substrate halting, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalNameCertObligationSurface`

Proof. Read the carrier boundary through Definition 936.107. It lists precisely the rows $T, F, V, K, R, H, C, P, N$, so carrier admission is already a finite BHist display and not an ambient total-host object.

Apply Definition 936.108. The root obligations bind classifier exposure, terminal-normality evidence, bounded-fuel replay, and non-escape to the same displayed packet. The classifier rows are T, F, V, K, R, while replay is transported by H and reread through C.

Finally use Definition 936.109. A root or downstream normal-route consumer can cite terminal normality only after all rows of Z are exposed. Since the listed host and global-normalization readings are absent from Z, the obligation surface is exhausted by the displayed packet. □

Theorem 936.111 ($\mathsf{ZNormal}^{\uparrow}$ classifier transport)**.** *Let*

$$Z = (T, F, V, K, R, H, C, P, N)$$

and

$$Z' = (T', F', V', K', R', H', C', P', N')$$

be accepted $\mathsf{ZNormal}^{\uparrow}$ *root packets. If the displayed transport rows give componentwise* hsame *from each classifier coordinate of Z to the corresponding coordinate of Z' and the displayed* Cont *replay rows carry the same bounded-fuel route, then classifier transport preserves the $T, F, V, K, R, H, C, P, N$ rows. The ZeckendorfNormal,* $\mathsf{ZCarry}^{\uparrow}$*, and* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *inputs are consumed only as sibling finite surfaces, not as new classifier coordinates.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket classifier transport`

Proof. Start from the two root boundaries. By Definition 936.107, both packets have the same row shape, so a componentwise hsame transport can only move displayed coordinates to displayed coordinates.

Use Definition 936.108 for the source packet and for the target packet. Classifier exposure is attached to T, F, V, K, R, while bounded replay is attached to F, H, C and to F', H', C'.

The sibling evidence is external only in the controlled sense already named by the chapter. The typed endpoint route is supplied by chapter 562, the finite normal-word route by chapter 278 and chapter 544, and the terminal continuation route by chapter 683.

Therefore transport preserves the displayed classifier rows and consumes those sibling surfaces only through K and R. No sibling certificate becomes a fresh coordinate of the $\mathsf{ZNormal}^{\uparrow}$ packet. □

Theorem 936.112 ($\mathsf{ZNormal}^{\uparrow}$ ledger nonescape)**.** *Every root or normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

factors its ledger read through the provenance and naming rows P, N. Consequently the consumer cannot read host conversion, quotient computation, unrestricted evaluation, divergence, substrate halting, or a global normalizer from the $\mathsf{ZNormal}^{\uparrow}$ *ledger.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket ledger nonescape certificate`

Proof. Project the ledger rows by Definition 936.107. The only provenance row is P, and the only local naming row is N.

Apply Definition 936.108. Non-escape is one of the root obligations, and it keeps Pkg provenance and local naming inside the displayed P, N rows.

Now apply Definition 936.109. Any root or normal-route consumer must expose the whole packet before terminal normality is read, so its ledger route has already passed through P, N.

The excluded host and global-normalization readings have no ledger coordinate in Z. Hence every accepted consumer factors through P, N, and the ledger cannot be reopened as host conversion, quotient computation, unrestricted evaluation, divergence, substrate halting, or a global normalizer. □

Theorem 936.113 ($\mathsf{ZNormal}^{\uparrow}$ terminal boundary non-escape)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

any terminal read through the displayed BHist *rows* T, F, V, R, H, C, P, N *remains inside the finite terminal-normality boundary of the packet. The read cannot export a host conversion row, quotient normalization row, divergence oracle, unrestricted evaluator, or global-normalization coordinate.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalBoundaryNonescape`

Proof. Project the accepted carrier by Definition 936.1 and the root packet by Definition 936.107. Both displays expose the terminal route as rows of one finite BHist packet, with T, F, V, R, H, C, P, N already attached to the same local naming surface.

Apply Definition 936.2. The local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ obligation reads terminal normality only after typed endpoint evidence, finite fuel, sibling normal-word evidence, continuation readback, transport, replay, provenance, and local naming have been displayed.

The sibling chapters supply component evidence rather than escape routes. chapter 562 supplies the typed endpoint surface, chapter 278 and chapter 544 supply finite normal-word and carry evidence, and chapter 683 supplies terminal continuation readback.

Since no host conversion, quotient normalization, divergence oracle, unrestricted evaluator, or global normalizer is a row of Z, the terminal read is exhausted by the displayed BHist packet. □

Theorem 936.114 ($\mathsf{ZNormal}^{\uparrow}$ source route totality)**.** *Every accepted root consumer that cites terminal normality for* $\mathsf{ZNormal}^{\uparrow}$ *must first expose the complete displayed source route*

$$T, F, V, K, R, H, C, P, N.$$

Thus the consumer reads terminal normality only after the terminal typed endpoint row, finite fuel row, terminal-normality row, sibling normal-word row, continuation readback row, hsame *transport row,* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row have all appeared in the* BHist *packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket source route totality`

Proof. Begin with Definition 936.109. It requires every downstream normal-route consumer to expose the root carrier boundary before citing terminal normality.

Now read the root boundary itself through Definition 936.107. The only source route made available to a consumer is the finite tuple $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.2. The local certificate binds carrier admission, classifier exposure, bounded-fuel replay, transport, package provenance, and local naming to the same displayed packet.

The typed endpoint, finite normal-word, carry, and terminal continuation inputs remain the sibling surfaces of chapter 562, chapter 278, chapter 544, and chapter 683. Therefore a consumer route is total exactly over the displayed BHist packet and has no hidden source outside it. □

Theorem 936.115 ($\mathsf{ZNormal}^{\uparrow}$ NameCert ledger exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *ledger is exhausted by the provenance and naming rows* P, N *over that displayed* BHist *packet. No additional ledger row remains for host conversion, quotient normalization, divergence detection, unrestricted evaluation, global normalization, or a second source route.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalNamecertLedgerExhaustion`

Proof. Project the carrier by Definition 936.1. The packet lists P as the Pkg provenance row and N as the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ row, with no separate ledger component.

Use Definition 936.2 to read the ledger through the same finite packet. The naming surface attaches carrier admission, terminal-normality evidence, sibling evidence, transport, replay, provenance, and naming to $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.107 and Definition 936.109. They force root consumers to display the same BHist source route before terminal normality can be used.

The sibling evidence remains local to the displayed rows: typed endpoints come from chapter 562, finite normal words from chapter 278, carry evidence from chapter 544, and terminal continuation readback from chapter 683. Consequently P, N exhaust the local ledger over Z, and every excluded host-side read lacks a NameCert row. □

Theorem 936.116 ($\mathsf{ZNormal}^{\uparrow}$ total-host readback display)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root-facing total-host readback surface exposes exactly the rows $T, F, V, K, R, H, C, P, N$*. Here* T *is the typed endpoint row,* F *is the finite fuel row,* V *is the terminal-normality row,* K *is the finite ZeckendorfNormal and* $\mathsf{ZCarry}^{\uparrow}$ *normal-word evidence row,* R *is the continuation readback row, and* H, C, P, N *are the displayed transport, route, provenance, and local naming rows.*

No root consumer receives host conversion, quotient normalization, unrestricted evaluation, divergence, or global normalization data from this display.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host readback display`

Proof. Project the carrier of Definition 936.1 and the root boundary of Definition 936.107. Both expose the same finite packet $T, F, V, K, R, H, C, P, N$ before any root-facing terminal claim can be read.

Apply Definition 936.2 and Definition 936.108. These obligations bind carrier admission, terminal-normality exposure, sibling evidence, bounded fuel, transport, replay, provenance, and local naming to the displayed root packet.

The row list contains no host conversion, quotient normalization, unrestricted evaluator, divergence oracle, or global normalizer. Thus the total-host readback display is exactly the finite root surface $T, F, V, K, R, H, C, P, N$. □

Theorem 936.117 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel route exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

any terminal-fuel consumer factors through the displayed finite route

$$T \longmapsto F \longmapsto V \longmapsto R$$

together with H, C, P, N. The sibling row K may be read only as finite normal-word evidence attached to that route, and it cannot replace the finite fuel route by host conversion, quotient normalization, unrestricted evaluation, divergence, or global normalization.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel route exhaustion`

Proof. Begin with the total-host readback display of Definition 936.116. It exposes the root packet as a finite BHist surface and leaves no unbounded evaluator row outside the packet.

Use Definition 936.72 and Definition 936.9. They place every terminal fuel read behind the typed endpoint T, finite fuel F, terminal-normality row V, and continuation readback row R.

Next apply Definition 936.113. The boundary keeps the read inside the displayed transport, replay, provenance, and local naming rows H, C, P, N.

Finally inspect K. By Definition 936.11, K is finite sibling normal-word evidence only. It cannot supply host conversion, quotient normalization, unrestricted evaluation, divergence, or global normalization in place of the terminal fuel route. □

Theorem 936.118 (ZNormal$^{\uparrow}$ normal-word sibling separation)**.** *For every accepted* ZNormal$^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the ZeckendorfNormal and ZCarry$^{\uparrow}$ *normal-word evidence carried by K remains sibling evidence for the finite route T, F, V, R, H, C, P, N. It is separate from host conversion, quotient normalization, unrestricted evaluation, divergence, and global normalization, and no root consumer may identify those excluded host-side objects with the normal-word evidence row.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal word sibling separation`

Proof. Project K from the accepted root packet. The carrier records K beside T, F, V, R, H, C, P, N, so the normal-word evidence is a displayed sibling row rather than a replacement for the terminal route.

Apply Definition 936.3 and Definition 936.11. These theorems read K through the finite ZeckendorfNormal and ZCarry$^{\uparrow}$ surfaces and attach it to the same BHist, hsame, Cont, Pkg, and NameCert rows as the rest of the packet.

Use Definition 936.73 for the excluded host-side objects. The frontier refuses host conversion, quotient computation, unrestricted evaluation, divergence, and global normalization as exported reads.

Therefore the K row remains finite normal-word sibling evidence only. The terminal route remains T, F, V, R, H, C, P, N, and no accepted root consumer can turn K into a host conversion, quotient normalization, unrestricted evaluator, divergence oracle, or global normalizer. □

Theorem 936.119 (ZNormal$^{\uparrow}$ root carrier exposure)**.** *For every accepted* ZNormal$^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root carrier exposure available to downstream normal-form consumers is exactly the finite row list displayed by Definition 936.1. The rows expose the typed endpoint T, finite fuel F, terminal-normality coordinate V, sibling normal-word evidence K, terminal readback R, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

No accepted root consumer receives a second carrier row outside this packet.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root carrier exposure`

Proof. Project the carrier in Definition 936.1. Its displayed tuple is $T, F, V, K, R, H, C, P, N$, with each coordinate named as a finite BEDC row before any terminal-normality or total-host claim can be read.

Apply Definition 936.2 and Definition 936.108. These obligations bind carrier admission, typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, terminal readback, transport, replay, provenance, and local naming to the same accepted packet.

Use the total-host readback display of Definition 936.116. It exposes the same row list on the root-facing surface and supplies no independent carrier coordinate. Hence the root carrier exposure is exactly the finite packet already used by the carrier definition. □

Theorem 936.120 ($\mathsf{ZNormal}^{\uparrow}$ root ledger nonescape)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root ledger can export only the displayed finite carrier, terminal-normality, sibling normal-word, terminal readback, transport, replay, provenance, and naming rows. It cannot export a divergence oracle, host conversion row, unrestricted evaluator, quotient normalizer, or global normalization procedure.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root ledger nonescape`

Proof. Begin with Definition 936.119. The root ledger-readable surface has the same finite coordinates $T, F, V, K, R, H, C, P, N$ as the carrier exposure.

Next apply Definition 936.124 and Definition 936.17. Both ledger boundaries exhaust the available reads through the displayed terminal-normality, fuel, normal-word, continuation, transport, replay, provenance, and naming rows.

Finally use Definition 936.10 and Definition 936.73. These refusal theorems block divergence detection, host conversion, unrestricted evaluation, quotient normalization, substrate halting, and global normalization from any route not carried by the finite packet. □

Theorem 936.121 ($\mathsf{ZNormal}^{\uparrow}$ root readback exhaustion)**.** *Every terminal readback accepted by a* $\mathsf{ZNormal}^{\uparrow}$ *root consumer factors through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of*

$$Z = (T, F, V, K, R, H, C, P, N).$$

The readback first exposes the typed endpoint, fuel, terminal-normality, sibling normal-word, and terminal-read rows, then transports and packages them through H, C, P, N*. It has no replacement route through host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readback exhaustion`

Proof. Start with Definition 936.119. It fixes the readable root packet as $T, F, V, K, R, H, C, P, N$ and identifies the displayed BHist source rows before any terminal readback is consumed.

Use Definition 936.8, Definition 936.117, and Definition 936.44. These results place terminal readback on the route from T through F, V, K, R and keep the terminal read inside the finite ledger.

Apply Definition 936.120. The only remaining structural rows are the componentwise hsame transport H, the displayed Cont replay C, the Pkg provenance P, and the local $\mathsf{NameCert}$ row N. Therefore every terminal readback factors through the displayed BEDC rows and cannot be replaced by an excluded host-side route. □

Theorem 936.122 ($\mathsf{ZNormal}^{\uparrow}$ terminal normality admission)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal-normality row V is exposed only after the typed endpoint row T and the finite fuel row F have already been displayed. The admission surface therefore reads terminal normality as a finite packet coordinate, not as a host conversion, quotient computation, unrestricted evaluator, divergence oracle, or substrate halting theorem.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTerminalNormalityAdmission`

Proof. Project the carrier of Definition 936.1. The accepted packet is the tuple $T, F, V, K, R, H, C, P, N$, and V appears in that tuple after the typed endpoint and fuel rows.

Apply Definition 936.2. The local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface reads carrier admission, typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, terminal readback, transport, replay, provenance, and naming as one finite certificate.

Use Definition 936.8. Terminal readback factors through T, F, V, R, H, C, P, N, so the consumer can reach V only through the displayed endpoint and fuel path.

Finally apply Definition 936.10. The admitted terminal-normality row remains inside the finite packet boundary and cannot be reinterpreted as host conversion, quotient computation, unrestricted evaluation, divergence detection, or a substrate halting theorem. □

Theorem 936.123 ($\mathsf{ZNormal}^{\uparrow}$ finite-fuel classifier stability)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

componentwise hsame *transport preserves the finite fuel row F, terminal normality row V, sibling normal-word row K, continuation readback row R, continuation replay row C, provenance row P, and local naming row N. The transported classifier surface adds no host conversion row and performs no quotient computation.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket finite fuel classifier stability`

Proof. Start with Definition 936.1. The rows F, V, K, R, H, C, P, N are coordinates of the same displayed BHist packet, with H reserved for componentwise hsame transport.

Apply Definition 936.2. The NameCert surface binds finite fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, provenance, and local naming to the same local packet.

Use Definition 936.8 for the continuation side. The terminal read is transported through the listed finite rows and does not introduce a separate evaluator.

Then apply Definition 936.10. Since the boundary refuses host conversion and quotient computation, componentwise hsame transport can preserve the displayed classifier rows only as finite packet data. □

Theorem 936.124 ($\mathsf{ZNormal}^{\uparrow}$ total-host ledger nonescape)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host ledger exports exactly the finite terminal-normality, fuel, normal-word, continuation, transport, replay, provenance, and local naming rows displayed by the packet. It exports no divergence oracle, unrestricted evaluator, global normalizer, host conversion row, or substrate halting theorem.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host ledger nonescape`

Proof. Project the carrier by Definition 936.1. The ledger-readable surface is the finite tuple $T, F, V, K, R, H, C, P, N$ and has no additional total-host coordinate.

Apply Definition 936.2. The same rows are read as the local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ obligations: endpoint, fuel, terminal normality, sibling normal-word evidence, terminal readback, transport, replay, provenance, and naming.

Use Definition 936.8 to route any terminal ledger read through the finite readback coordinates. The row R is a displayed continuation read, not an unrestricted evaluator.

Finally apply Definition 936.10. The total-host ledger therefore cannot export divergence detection, global normalization, host conversion, quotient computation, or a substrate halting theorem. □

Theorem 936.125 ($\mathsf{ZNormal}^\uparrow$ root normal-form readiness)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root normal-form consumer reads terminal normality only through the finite packet displayed by Definition 936.1. The packet first exposes the carrier and local obligations, then routes terminal readback, sibling normal-word evidence, and the root visible source package through the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The readiness boundary exports no host conversion, quotient computation, unrestricted evaluator, divergence oracle, or global normalizer.

Proof. Project the carrier and local naming surface using Definition 936.1 and Definition 936.2. These rows display exactly $T, F, V, K, R, H, C, P, N$ and bind them to the finite $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ packet.

Next apply Definition 936.8. Terminal readback is routed through T, F, V, R, H, C, P, N, so terminal normality remains a finite row rather than a host-side conversion judgment.

Now use Definition 936.11 and Definition 936.24. The sibling normal-word row K is finite evidence attached to the same packet, and the visible root source exposes all normal-output reads only after $T, F, V, K, R, H, C, P, N$ have been displayed.

The omitted host objects have no coordinate in that packet. Hence root normal-form readiness is a finite BEDC boundary over BHist, hsame, Cont, Pkg, and NameCert rows only. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootNormalFormReadiness`

Theorem 936.126 ($\mathsf{ZNormal}^\uparrow$ total-host ledger exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host ledger is exhausted by the typed endpoint row T, finite fuel row F, terminal-normality row V, sibling normal-word row K, terminal readback row R, componentwise transport row H, displayed continuation row C, provenance row P, and local name row N. Every total-host consumer of the packet factors through those rows and cannot add host conversion, quotient computation, unrestricted evaluation, divergence detection, or a global normalization procedure.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host ledger exhaustion`

Proof. Start from Definition 936.1. The ledger-readable carrier is the finite tuple $T, F, V, K, R, H, C, P, N$ and has no additional total-host coordinate.

Apply Definition 936.2. The local certificate reads carrier admission, finite fuel, terminal normality, sibling normal-word evidence, terminal readback, transport, replay, provenance, and naming as one packet.

Use Definition 936.8 and Definition 936.11. Terminal readback and sibling normal-word evidence stay inside the displayed finite rows before any consumer can claim a normal-form output.

Finally apply Definition 936.24. The visible source package is already the whole total-host surface, so host conversion, quotient computation, unrestricted evaluation, divergence detection, and global normalization have no route through the accepted ledger. □

Theorem 936.127 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel consumer package)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal fuel consumer package is the route from the typed endpoint T through finite fuel F, terminal normality V, sibling normal-word evidence K, terminal readback R, and the support rows H, C, P, N. The route is a finite BHist *package with* hsame*,* Cont*,* Pkg*, and* NameCert *support; it is not a host evaluator, quotient normalizer, divergence oracle, unrestricted conversion service, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel consumer package`

Proof. Read Z through Definition 936.1 and Definition 936.2. The finite fuel row, terminal normality row, sibling normal-word row, terminal readback row, and structural support rows are all part of the same accepted packet.

Apply Definition 936.8. The terminal consumer reaches readback only through T, F, V, R, H, C, P, N, so the fuel row is a required part of the route and not an optional host computation step.

Then use Definition 936.11. The row K supplies only finite sibling normal-word evidence and remains attached to the same BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 936.24. Since the root visible source package exhausts the normal-output route, the terminal fuel consumer package cannot expose host conversion, quotient computation, unrestricted evaluation, divergence detection, or global normalization. □

Theorem 936.128 ($\mathsf{ZNormal}^{\uparrow}$ root constructor readback)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

each root constructor readback factors through the displayed terminal, fuel, terminal-normality, sibling normal-word, terminal readback, transport, replay, provenance, and local-name rows. The constructor surface therefore reads only the finite BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *packet and does not introduce a host conversion row, quotient computation, unrestricted evaluator, divergence oracle, substrate-halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root constructor readback`

Proof. Project the carrier of Definition 936.1. The root constructor inventory is exactly the finite tuple $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.90. It identifies the terminal row constructor surface with typed endpoint, finite fuel, terminal normality, sibling normal-word evidence, terminal readback, componentwise transport, continuation replay, provenance, and local naming.

Use Definition 936.125 and Definition 936.126. These theorems keep the constructor readback on the same normal-form readiness route and exhaust the total-host ledger by the displayed packet.

Finally apply Definition 936.81. The TasteGate-facing field surface is the same packet surface, so the constructor readback has no additional host or global-normalization coordinate. □

Theorem 936.129 ($\mathsf{ZNormal}^{\uparrow}$ terminal normal-form refusal)**.** *Every terminal normal-form read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet refuses host conversion, quotient computation, unrestricted evaluation, divergence oracle access, substrate-halting evidence, and global normalizer routes. Such a read may use only the finite packet*

$$T, F, V, K, R, H, C, P, N$$

and the sibling normal-word surfaces named by the $\mathsf{ZNormal}^{\uparrow}$ *root route.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal normal form refusal`

Proof. Begin with Definition 936.122. Terminal normality is admitted only through the displayed typed endpoint and finite fuel rows, so a terminal read cannot start from host conversion, quotient computation, unrestricted evaluation, divergence detection, or substrate halting.

Apply Definition 936.58. The root interface excludes host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate-halting rows, and global normalizers from every total-host or downstream normal-form use.

Next use Definition 936.57. The normal-word component K is consumed only through finite sibling surfaces and cannot be detached into a host normalizer or quotient computation.

Finally apply Definition 936.91. The fuel replay route records only the displayed finite nondivergence path through $T, F, V, K, R, H, C, P, N$. Hence terminal normal-form reads refuse exactly the listed host and global routes. □

Theorem 936.130 ($\mathsf{ZNormal}^{\uparrow}$ consumer route totality)**.** *Every downstream consumer of* $\mathsf{ZNormal}^{\uparrow}$ *factors through the finite packet*

$$T, F, V, K, R, H, C, P, N$$

together with the typed endpoint, finite normal-word, carry, and terminal continuation sibling surfaces used by the root route. The consumer receives no normal-form evidence outside those rows, and the TasteGate-facing boundary remains confined to the displayed field surface rather than a host evaluator or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket consumer route totality`

Proof. Start from Definition 936.56. It places every downstream normal-form consumer inside the displayed $\mathsf{ZNormal}^{\uparrow}$ root packet.

Use Definition 936.60. The consumer receives terminal normality, finite normal-word evidence, transport, continuation replay, provenance, and local naming only through the named packet rows.

Attach the sibling side by Definition 936.61. The ZeckendorfNormal, ZCarry, and ContinuationTraceNormalForm sibling surfaces exhaust the permitted normal-word route outside the root packet, and they exclude host conversion, quotient normalization, unrestricted evaluation, divergence detection, substrate halting, and global normalization.

Finally apply Definition 936.81. The TasteGate-facing boundary is the same finite field surface, so downstream consumers are total over the packet and sibling surfaces and have no route outside them. □

Theorem 936.131 ($\mathsf{ZNormal}^{\uparrow}$ terminal normality carrier exposure)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

terminal normality is exposed only through the displayed typed endpoint row T, finite fuel row F, terminal-normality row V, terminal read row R, componentwise transport row H, continuation replay row C, package row P, and local naming row N. The row K contributes only finite sibling normal-word evidence, and no accepted carrier can reread K as host conversion, quotient computation, an unrestricted evaluator, a divergence oracle, or a global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal normality carrier exposure`

Proof. Project the carrier in Definition 936.1. Its accepted packet is the finite row list $T, F, V, K, R, H, C, P, N$, with V singled out as the terminal-normality coordinate and K listed separately as sibling normal-word evidence.

Apply Definition 936.2. The local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ obligations bind carrier admission, typed endpoint, finite fuel, terminal normality, finite normal-word comparison, terminal readback, transport, replay, provenance, and local naming to this same packet.

Use Definition 936.6 and Definition 936.8. These theorems route every terminal read through T, F, V, R, H, C, P, N and keep the normal-word coordinate K outside the terminal-read source.

Finally apply Definition 936.11. The K row is finite sibling evidence only, so terminal normality remains exposed through the displayed carrier rows and cannot become any excluded host-side or global-normalization object. □

Theorem 936.132 ($\mathsf{ZNormal}^{\uparrow}$ total-host fuel replay boundary)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

total-host fuel replay is the finite route from typed endpoint T through fuel F to terminal row V and terminal continuation read R. The route is transported by H, replayed by C, packaged by P, and named by N; it does not export a host conversion row, quotient computation, unrestricted evaluator, divergence oracle, substrate halting claim, or global normalizer.

Proof. Start with Definition 936.1. The carrier displays the endpoint, fuel, terminal normality, sibling normal-word, continuation read, transport, replay, package, and naming rows as the single finite packet $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.2 to read the same rows as the local certificate surface. The fuel coordinate F and terminal-normality coordinate V are therefore packet rows, not ambient host computation relations.

Next use Definition 936.6 and Definition 936.8. They force terminal fuel replay along T, F, V, R before any consumer can read the terminal continuation surface.

The structural route is then exactly H, C, P, N. By Definition 936.11, the sibling row K remains finite evidence beside that route, so the fuel replay boundary cannot escape to any excluded host, quotient, divergence, substrate-halting, or global-normalization row. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host fuel replay boundary`

Theorem 936.133 ($\mathsf{ZNormal}^{\uparrow}$ consumer divergence-oracle nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the displayed rows*

$$T, F, V, K, R, H, C, P, N.$$

Such a consumer may use typed endpoint evidence, finite fuel replay, terminal normality, finite sibling normal-word evidence, terminal continuation read, transport, replay, package provenance, and local naming, but it cannot obtain a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalizer, or substrate halting claim.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket consumer divergence oracle nonescape`

Proof. Project the accepted carrier by Definition 936.1. The consumer surface is the finite packet $T, F, V, K, R, H, C, P, N$, and there is no second consumer coordinate outside that packet.

Use Definition 936.2. It identifies the permitted reads as carrier admission, typed endpoint, finite fuel, terminal normality, sibling normal-word comparison, continuation read, transport, replay, provenance, and local naming.

Apply Definition 936.6 and Definition 936.8. These route terminal normal-form consumption through T, F, V, R, H, C, P, N and prevent the consumer from detaching a host computation or unbounded evaluator from the finite replay.

Finally apply Definition 936.11. The remaining row K is finite sibling evidence only. Hence every downstream consumer factors through the displayed packet and has no route to divergence detection, host conversion, quotient computation, unrestricted evaluation, global normalization, or substrate-halting evidence. □

Theorem 936.134 ($\mathsf{ZNormal}^{\uparrow}$ root terminal-host route classifier exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root terminal-host route classifier is exhausted by the displayed terminal normality row V, finite fuel replay row F, total-host refusal ledger implicit in the accepted packet, componentwise BHist *and* hsame *rows T, H, displayed* Cont *replay row C, package row P, and local* NameCert *row N. The classifier may also read the sibling normal-word row K and terminal readback row R only as rows of the same packet; it exports no divergence oracle, host conversion row, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal host route classifier exhaustion`

Proof. Start from the terminal normality carrier exposure, Definition 936.131. It identifies the accepted carrier with the finite rows $T, F, V, K, R, H, C, P, N$ and places terminal normality on the displayed row V rather than on host computation.

Use the total-host fuel replay boundary, Definition 936.132. The root route reaches terminal normality through T, F, V, R and then reads transport, replay, package, and local naming only through H, C, P, N.

Finally apply the consumer non-escape theorem, Definition 936.133. Since no downstream consumer can obtain a divergence oracle, host conversion, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer, the terminal-host classifier is exhausted by the displayed packet rows. □

Theorem 936.135 ($\mathsf{ZNormal}^{\uparrow}$ root total-host consumer readback lock)**.** *Every total-host consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet reads the same terminal-host lock*

$$T, F, V, K, R, H, C, P, N.$$

The typed endpoint, finite fuel replay, terminal normality, sibling normal-word evidence, terminal readback, BHist *support, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows are the only rows available to that consumer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host consumer readback lock`

Proof. First use Definition 936.131. It fixes the consumer surface as the finite packet $T, F, V, K, R, H, C, P, N$.

Next use Definition 936.132. Fuel replay and terminal readback remain locked to T, F, V, R and cannot detach a separate host conversion or evaluator row.

Now invoke Definition 936.134. The classifier surface for a terminal-host route is already exhausted by the same displayed rows.

The non-escape row Definition 936.133 then rules out every external total-host read. Hence all total-host consumers read the same finite lock and no unlisted row remains available. □

Theorem 936.136 ($\mathsf{ZNormal}^{\uparrow}$ root NameCert terminal route readiness)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row of an accepted packet is ready for terminal-route consumption exactly when the displayed terminal normality row, finite fuel replay row, total-host refusal ledger,* BHist *support,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming rows are read as the single packet* $T, F, V, K, R, H, C, P, N$*. This readiness does not add a host conversion row, quotient computation, divergence oracle, unrestricted evaluator, substrate-halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root namecert terminal route readiness`

Proof. Begin with Definition 936.131. The carrier exposes terminal normality and the supporting rows only as the accepted finite packet.

Use Definition 936.135. It fixes the readback lock for every total-host consumer to the same rows $T, F, V, K, R, H, C, P, N$.

Finally use Definition 936.133. The local NameCert row can certify readiness for the displayed terminal route, but it has no coordinate from which to provide any excluded host-side or global-normalization object. □

Theorem 936.137 ($\mathsf{ZNormal}^{\uparrow}$ terminal-host root factorization)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

each terminal-host consumer factors through the carrier row of Definition 936.1, the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *obligations, the fuel-normality non-escape surface, terminal readback exactness, sibling normal-word handoff, and the root TasteGate field surface. The factorization reads* $T, F, V, K, R, H, C, P, N$ *as one finite* BHist *packet transported by* hsame*, replayed by* Cont*, packaged by* Pkg*, and named by* NameCert*.*

No terminal-host consumer receives a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalizer, or substrate-halting claim outside that displayed packet.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal host root factorization`

Proof. Project the carrier by Definition 936.1. The terminal-host surface is the finite tuple $T, F, V, K, R, H, C, P, N$, so the consumer has no row outside the displayed packet.

Apply Definition 936.2 and Definition 936.6. These rows bind carrier admission, finite fuel replay, terminal normality, continuation readback, transport, replay, package, and naming to the same finite route.

Use Definition 936.8, Definition 936.11, and Definition 936.81. The terminal read, sibling normal-word row, and TasteGate-facing field surface all inspect the same tuple, so the terminal-host consumer factors through the displayed BEDC primitives. □

Theorem 936.138 ($\mathsf{ZNormal}^{\uparrow}$ fuel replay root exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

fuel replay at the root is exhausted by the route from typed endpoint T through finite fuel F, terminal-normality row V, and terminal readback R, with H, C, P, N supplying the only transport, continuation, package, and naming rows. The row K contributes only the sibling normal-word handoff surface already displayed in the packet.

Thus a fuel-replay consumer cannot reread the packet as host conversion, quotient computation, divergence detection, unrestricted evaluation, global normalization, or substrate halting.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket fuel replay root exhaustion`

Proof. Begin with Definition 936.6. It places terminal normality behind the finite route $T \longmapsto F \longmapsto V \longmapsto R$ and keeps K as sibling normal-word evidence.

Next apply Definition 936.8. Terminal readback is exact only through T, F, V, R, H, C, P, N, so the replay cannot detach a host computation row from the accepted packet.

Finally use Definition 936.81 together with the local NameCert obligations. The root field surface is exactly the displayed tuple $T, F, V, K, R, H, C, P, N$, and the excluded host-side objects have no carrier coordinate there. □

Theorem 936.139 ($\mathsf{ZNormal}^{\uparrow}$ normal-output root non-escape)**.** *Every normal-output read from an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

remains inside the terminal-host root surface: carrier admission through T, F, V, K, R, structural transport through H, continuation replay through C, package provenance through P, and local naming through N. The read may cite the sibling normal-word handoff and the root TasteGate field surface, but it exports no detached normal-output object beyond the displayed BEDC primitives.

In particular, no normal-output consumer can obtain host conversion, quotient computation, divergence detection, unrestricted evaluation, global normalization, or substrate-halting evidence from the packet.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal output root nonescape`

Proof. Use Definition 936.18. Normal-output consumption is already confined to the finite carrier and the displayed normal-form rows.

Apply Definition 936.11 and Definition 936.8. They identify the only sibling normal-word and terminal-readback routes available to the consumer.

Finally combine Definition 936.2, Definition 936.6, and Definition 936.81. These rows bind the output read to BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, and no excluded host-side or substrate-halting object remains exportable. □

Theorem 936.140 ($\mathsf{ZNormal}^{\uparrow}$ total-host readback triad)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

total-host readback exposes exactly the terminal-normality row V, the bounded fuel replay row F, and the non-escape row witnessed by the local carrier and naming surface. The exposure is a finite BHist *route transported by* hsame*, read by* Cont*, packaged by* Pkg*, and named by* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *through $T, F, V, K, R, H, C, P, N$.*

No accepted total-host readback supplies a detached host conversion row, quotient computation, divergence oracle, unrestricted evaluator, global normalizer, or substrate-halting certificate.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootTotalHostReadbackTriad`

Proof. Project the accepted carrier by Definition 936.1. The packet shows the terminal-normality coordinate V, the fuel coordinate F, the readback coordinate R, and the support rows H, C, P, N inside the single finite tuple $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.2. The local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ row binds carrier admission, terminal normality, bounded fuel replay, sibling normal-word evidence, readback, transport, continuation replay, package provenance, and local naming to the same packet.

Use Definition 936.8 together with Definition 936.6. Terminal readback reaches V only through the bounded fuel route, while the non-escape theorem excludes host conversion, quotient computation, divergence detection, unrestricted evaluation, global normalization, and substrate halting.

Finally combine Definition 936.137, Definition 936.138, and Definition 936.139. These root rows identify the same BHist, hsame, Cont, Pkg, and NameCert surface, so the total-host readback triad is exactly terminal normality, bounded fuel replay, and non-escape. □

Theorem 936.141 ($\mathsf{ZNormal}^{\uparrow}$ fuel route non-escape)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the bounded fuel route from T through F to V and R cannot export divergence detection, host conversion, quotient computation, unrestricted evaluation, global normalization, or substrate halting. Its whole observable surface is the finite BHist *packet with* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *support.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootFuelRouteNonescape`

Proof. Start with Definition 936.6. It already places terminal normality behind finite fuel replay and rules out the host-side and global-normalization objects listed in the statement.

Next apply Definition 936.8. The readback row R is reached only after the endpoint, fuel, and terminal-normality rows have been displayed, so no consumer can detach R into an ambient host conversion or quotient-computation route.

Use Definition 936.137. The route remains inside the same $T, F, V, K, R, H, C, P, N$ packet, with H, C, P, N giving transport, continuation replay, package provenance, and local naming.

The sibling row is harmless by Definition 936.11: K contributes finite normal-word evidence only. Hence the bounded fuel route has no coordinate from which divergence, host conversion, global normalization, or substrate halting could be exported. □

Theorem 936.142 ($\mathsf{ZNormal}^{\uparrow}$ downstream normal-form readiness)**.** *Every downstream normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

must factor through the terminal-host exposure, the terminal-host obligation rows, and the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *route. The consumer may read only the finite* BHist *surface carried by $T, F, V, K, R, H, C, P, N$, transported by* hsame*, replayed by* Cont*, and packaged by* Pkg*.*

Consequently downstream readiness is a finite normal-form boundary: it does not provide host conversion, quotient computation, unrestricted evaluation, divergence detection, global normalization, or substrate-halting evidence.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootDownstreamNormalFormReadiness`
Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream readiness namecert`

Proof. First expose the terminal-host surface with Definition 936.131 and Definition 936.137. These theorems show that every terminal-host consumer reads the displayed packet and no second host-side coordinate.

Next apply Definition 936.138 and Definition 936.139. These obligation rows bind terminal normality, bounded fuel replay, readback, transport, continuation replay, package provenance, and naming to the same finite carrier.

Use Definition 936.2 to repack those rows as the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ route. The support objects are precisely BHist, hsame, Cont, Pkg, and the local name certificate over $T, F, V, K, R, H, C, P, N$.

Finally apply Definition 936.141. Since the fuel route cannot export host conversion, divergence detection, unrestricted evaluation, quotient computation, global normalization, or substrate halting, downstream normal-form readiness factors through the terminal host rows and does not exceed them. □

Definition 936.143 ($\mathsf{ZNormal}^{\uparrow}$ terminal readback route carrier)**.** *The* $\mathsf{ZNormal}^{\uparrow}$ *terminal readback route carrier is the finite packet*

$$Q = (T, F, V, K, R, H, C, P, N)$$

read from an accepted $\mathsf{ZNormal}^{\uparrow}$ *carrier as follows:* T *is the typed endpoint row,* F *is the bounded fuel replay row,* V *is the terminal-normality row,* K *is the finite normal-word comparison row,* R *is the continuation readback row,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row.*

The route carrier is terminal only when the readback passes through the rows displayed in chapter 562, chapter 278, and chapter 683. It has no coordinate for a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalizer, or substrate-halting certificate.

Lean4 (definition) `BEDC.Derived.ZnormalUp.ZnormalTerminalReadbackRouteCarrier`

Theorem 936.144 ($\mathsf{ZNormal}^{\uparrow}$ terminal readback route exhaustion)**.** *Every downstream read of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *terminal readback route carrier* $Q = (T, F, V, K, R, H, C, P, N)$ *factors through terminal normality* V*, bounded fuel replay* F*, continuation readback* R*, componentwise transport* H*, displayed replay* C*, provenance* P*, the local naming row* N*, and the sibling finite normal-word row* K*. The factorization is through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *only.*

Consequently no downstream normal-route consumer can obtain a detached host conversion, quotient computation, divergence oracle, unrestricted evaluator, global normalization procedure, or substrate-halting theorem from this route.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal readback route exhaustion`

Proof. Start with Definition 936.143. The route carrier lists only $T, F, V, K, R, H, C, P, N$, so every downstream read begins inside the same finite packet.

Apply the carrier and obligation rows of Definition 936.1 and Definition 936.2. They identify typed endpoint readback, bounded fuel replay, terminal normality, normal-word comparison, continuation readback, transport, replay, provenance, and local naming as one $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface.

Next use Definition 936.137, Definition 936.138, and Definition 936.139. These root rows keep terminal-host consumption, fuel replay, and normal-output reads inside the displayed packet.

Finally apply Definition 936.140 and Definition 936.142. The typed endpoint, Zeckendorf normal-word, and continuation trace references already present in the chapter index supply the

sibling readback boundaries, while the terminal-host rows exclude host conversion, quotient computation, unrestricted evaluation, divergence detection, global normalization, and substrate halting. □

Theorem 936.145 ($\mathsf{ZNormal}^{\uparrow}$ total-host refusal nonescape)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *terminal readback route carrier* $Q = (T, F, V, K, R, H, C, P, N)$*, the total-host refusal surface is exhausted by terminal normality, bounded fuel replay, continuation readback, componentwise transport, displayed replay, provenance, and local* NameCert *rows. The packet exports no divergence oracle, host conversion row, global normalization procedure, quotient computation, unrestricted evaluator, or substrate-halting certificate.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostRefusalNonescape`

Proof. Project the route carrier of Definition 936.143. The rows $T, F, V, K, R, H, C, P, N$ expose a finite terminal readback route, not a host evaluator or global normalizer.

Use Definition 936.6, Definition 936.8, and Definition 936.141. These theorems place every terminal read behind bounded fuel and remove divergence detection, host conversion, quotient computation, unrestricted evaluation, global normalization, and substrate halting from the fuel route.

Finally combine Definition 936.137, Definition 936.142, and Definition 936.144. The resulting route is a finite BHist packet transported by hsame, replayed by Cont, packaged by Pkg, and named by $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$, so the listed host-side objects have no remaining export coordinate. □

Theorem 936.146 ($\mathsf{ZNormal}^{\uparrow}$ total-host scope triad)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

every terminal-host use factors through the same displayed tuple. The typed route reads T, F, V *through chapter 562; the finite normal-word route reads* K *through chapter 278 and chapter 544; and the continuation route reads* R *through chapter 683.*

The structural rows H, C, P, N *are the only componentwise transport, replay, package, and local naming rows available to those three routes.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host scope triad`

Proof. Project the carrier by Definition 936.1. It exposes the single finite tuple $T, F, V, K, R, H, C, P, N$ and gives no separate terminal-host source row.

Use Definition 936.2. The local certificate binds the typed endpoint, fuel, terminal-normality, sibling normal-word, continuation readback, transport, replay, package, and naming rows to the same accepted packet.

Apply Definition 936.6, Definition 936.8, and Definition 936.11. These identify the typed, continuation, and sibling normal-word reads as the three visible routes of the packet.

Finally use Definition 936.13 and Definition 936.81. They keep the three routes on the TasteGate-facing field surface and leave H, C, P, N as the only transport, replay, package, and naming rows. □

Theorem 936.147 ($\mathsf{ZNormal}^{\uparrow}$ fuel replay exactness)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

bounded fuel replay reaches terminal normality only along the route

$$T \longmapsto F \longmapsto V \longmapsto R.$$

The route is transported by H, replayed by C, packaged by P, and named by N; the sibling row K can be consumed only as finite normal-word evidence already attached to the same packet.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket fuel replay exactness certificate`

Proof. Start from Definition 936.6. It places terminal normality behind finite fuel and excludes any read that bypasses the displayed route from T through F to V.

Use Definition 936.8. Terminal readback reaches R only after the typed endpoint, finite fuel, and terminal-normality rows have already been exposed.

Apply Definition 936.11. The sibling route K supplies finite normal-word evidence for the same packet and does not replace the fuel replay route.

Finally combine Definition 936.13 with Definition 936.2. Transport, replay, package provenance, and local naming are exactly the rows H, C, P, N, so bounded fuel replay has no detached terminal-normality witness outside Z. □

Theorem 936.148 ($\mathsf{ZNormal}^{\uparrow}$ divergence-oracle exclusion)**.** *No accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N)$$

exports a divergence oracle, host conversion row, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer. Every terminal-host read remains a read of $T, F, V, K, R, H, C, P, N$ through the carrier, the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *obligations, the fuel and readback exactness rows, the sibling normal-word handoff, route transport, and the TasteGate-facing field surface.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket divergence oracle exclusion`

Proof. Read the carrier by Definition 936.1. The accepted packet contains only the typed endpoint, finite fuel, terminal-normality, sibling normal-word, continuation readback, transport, replay, package, and naming rows.

Apply Definition 936.2 and Definition 936.6. The local certificate and finite-fuel boundary prevent terminal normality from being reread as a host-side computation object.

Use Definition 936.8, Definition 936.11, and Definition 936.13. Terminal readback, sibling normal-word evidence, and movement between rows all stay inside the finite BEDC packet.

Finally apply Definition 936.81. The TasteGate-facing field surface exposes no coordinate for divergence detection, host conversion, quotient computation, unrestricted evaluation, substrate halting, or global normalization; therefore none can be read from the accepted packet. □

Theorem 936.149 ($\mathsf{ZNormal}^{\uparrow}$ componentwise NameCert transport)**.** *Let*

$$Z = (T, F, V, K, R, H, C, P, N)$$

and

$$Z' = (T', F', V', K', R', H', C', P', N')$$

be accepted $\mathsf{ZNormal}^{\uparrow}$ *root packets over displayed* BHist *rows. If the componentwise* hsame *transport row H relates T, F, V, K, R, C, P, N to $T', F', V', K', R', C', P', N'$, then the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *surface transports from Z to Z' without adding a host normalization row, quotient conversion row, or divergence oracle. The transported surface remains the same displayed terminal typed endpoint, bounded fuel, terminal-normality, total-host refusal, continuation readback,* Cont *replay,* Pkg *provenance, and local naming packet.*

Proof. Project both packets through Definition 936.107. The only readable coordinates are the displayed BHist rows $T, F, V, K, R, H, C, P, N$ and their primed counterparts.

Apply Definition 936.108. The certificate surface is carrier admission, classifier exposure, terminal-normality evidence, bounded-fuel replay, non-escape, Pkg provenance, and local naming over exactly those rows.

The sibling chapters provide the component witnesses rather than host normalization. The typed endpoint row is read through chapter 562, the normal-word row through chapter 278, and the terminal continuation row through chapter 683.

Transport along H therefore moves each visible component and then rereads the same route through C. Since no host normalization, quotient conversion, or divergence-oracle coordinate occurs in the carrier, the transported NameCert remains kernel-scoped. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket componentwise namecert transport`

Theorem 936.150 ($\mathsf{ZNormal}^{\uparrow}$ total-host refusal boundary)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host refusal row K is part of the kernel-scoped $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *boundary. Any consumer that cites terminal normality from V must also display K together with the bounded fuel row F, continuation readback row R,* hsame *transport row H,* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N. A consumer that omits K has not crossed the total-host boundary and cannot export host conversion, quotient normalization, divergence detection, unrestricted evaluation, or global normalization as a* $\mathsf{ZNormal}^{\uparrow}$ *consequence.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host refusal boundary`

Proof. Start with the root carrier boundary. The row K is a coordinate of the displayed BHist packet, not a host-side comment about normalization.

Use Definition 936.108. Terminal-normality evidence is accepted only after the bounded fuel row F and the total-host refusal row K have been displayed, and replay is restricted to the visible Cont route C.

Now read the sibling inputs. The typed reduction normal-form packet supplies a typed endpoint, the Zeckendorf normal packet supplies finite normal-word evidence, and the continuation trace normal-form packet supplies terminal continuation readback. None of these siblings supplies a host conversion, quotient normalization, divergence oracle, unrestricted evaluator, or global normalizer.

Therefore any consumer omitting K lacks one of the boundary rows required by $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$. The terminal-normality citation remains inside the finite kernel packet and cannot be promoted to a total-host export. □

Theorem 936.151 ($\mathsf{ZNormal}^{\uparrow}$ terminal route exhaustion)**.** *Every terminal route exported by an accepted* $\mathsf{ZNormal}^{\uparrow}$ *root packet factors through*

$$T, F, V, K, R, H, C, P, N.$$

After the terminal typed endpoint row T, bounded fuel row F, terminal-normality row V, total-host refusal row K, continuation readback row R, componentwise hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N have been read, no additional kernel row remains for host normalization, quotient conversion, divergence detection, unrestricted evaluation, global normalization, or a second terminal route.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal route exhaustion`

Proof. Apply Definition 936.114. A root consumer that cites terminal normality must expose the complete source route $T, F, V, K, R, H, C, P, N$.

Apply Definition 936.115. The local ledger is exhausted by the displayed Pkg and NameCert rows over that same packet, so there is no second provenance or naming coordinate for a hidden terminal route.

Finally use Definition 936.113. It rules out host conversion, quotient normalization, divergence oracles, unrestricted evaluation, and global normalization from the terminal boundary. Thus the displayed route is exhausted once the nine kernel rows have been read. □

Theorem 936.152 ($\mathsf{ZNormal}^{\uparrow}$ total-host field readiness)**.** *Before any total-host normality read is accepted from a* $\mathsf{ZNormal}^{\uparrow}$ *root packet, the rows*

$$T, F, V, K, R, H, C, P, N$$

must be displayed fieldwise: terminal typed endpoint, bounded fuel, terminal-normality evidence, total-host refusal, continuation readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host field readiness`

Proof. Apply Definition 936.108. The root certificate surface requires carrier admission, classifier exposure, terminal-normality evidence, bounded-fuel replay, non-escape, provenance, and local naming over the displayed packet.

Use Definition 936.114. A consumer that cites terminal normality must expose the full source route $T, F, V, K, R, H, C, P, N$.

Finally apply Definition 936.150. It places K inside the accepted boundary, so a total-host read cannot consume V without also displaying the refusal row and the structural rows that keep the route kernel-scoped. □

Theorem 936.153 ($\mathsf{ZNormal}^{\uparrow}$ terminal TasteGate readiness)**.** *The displayed terminal endpoint, bounded fuel, terminal-normality row, total-host refusal row, continuation readback, componentwise transport, replay, provenance, and local NameCert rows are exactly the paper surface expected by later* Nontrivial $\mathsf{ZNormal}^{\uparrow}$ *and* FieldFaithful $\mathsf{ZNormal}^{\uparrow}$ *Lean instances. No such instance may read host conversion, quotient normalization, divergence detection, unrestricted evaluation, global normalization, or a second terminal route.*

Proof. Begin with total-host field readiness from Definition 936.152. It fixes the candidate TasteGate fields as $T, F, V, K, R, H, C, P, N$.

Use Definition 936.149. Componentwise transport preserves the same terminal typed endpoint, bounded fuel, terminal-normality, refusal, replay, provenance, and naming packet without adding a host normalization row.

Then use Definition 936.151. Once the nine rows have been read, no additional kernel row remains for a second terminal route or an excluded host operation.

Thus the later Lean instances have the exact field-faithful surface they need: nontriviality is witnessed by the presence of the nine distinct rows, and field faithfulness is the demand that consumers preserve those rows. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal tastegate readiness`

Theorem 936.154 ($\mathsf{ZNormal}^{\uparrow}$ root readiness export package)**.** *The componentwise transport, refusal boundary, terminal route exhaustion, and TasteGate readiness results assemble the* $\mathsf{ZNormal}^{\uparrow}$ *root unblock package. A downstream root consumer receives exactly the terminal typed endpoint, bounded fuel, terminal-normality evidence, total-host refusal, continuation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming rows.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readiness export certificate`

Proof. First apply componentwise transport from Definition 936.149. It identifies the rowwise transport route and keeps it inside the displayed root packet.

Next apply the total-host refusal boundary, Definition 936.150, together with total-host field readiness from Definition 936.152. These results force K and all structural rows to be visible before any terminal-normality export is consumed.

Use terminal route exhaustion, Definition 936.151. It excludes host conversion, quotient normalization, divergence detection, unrestricted evaluation, global normalization, and a second terminal route after $T, F, V, K, R, H, C, P, N$ have been read.

Finally apply terminal TasteGate readiness from Definition 936.153. The same nine rows are the expected Nontrivial and FieldFaithful surface, so the root consumer receives a complete root unblock package rather than a host-level normalization theorem. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root readiness export package`

Theorem 936.155 ($\mathsf{ZNormal}^{\uparrow}$ terminal fuel readback admission)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

a terminal-host consumer is admitted to read terminal normality only after it displays the finite fuel readback route from T through F to V and R, together with the transport, replay, provenance, and local naming rows H, C, P, N. The sibling row K may be consumed only as finite normal-word evidence already attached to the packet. The admission route exports no divergence oracle, host conversion row, quotient computation, unrestricted evaluator, global normalization procedure, or substrate-halting coordinate.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal fuel readback admission`

Proof. Start with the total-host readback display of Definition 936.116. It exposes the root packet as the finite rows $T, F, V, K, R, H, C, P, N$ before any terminal-host read is available.

Next apply Definition 936.117. The terminal read must pass through the route $T \longmapsto F \longmapsto V \longmapsto R$ with H, C, P, N as the only structural support.

Finally use Definition 936.141 and Definition 936.11. These keep K as finite sibling normal-word evidence and exclude divergence, host conversion, quotient computation, unrestricted evaluation, global normalization, and substrate halting from the admitted readback route. □

Theorem 936.156 ($\mathsf{ZNormal}^{\uparrow}$ total-host normal-form ledger exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet $Z = (T, F, V, K, R, H, C, P, N)$, the total-host normal-form ledger is exhausted by terminal normality V, bounded fuel replay F, terminal readback R, sibling normal-word evidence K, and the structural rows H, C, P, N. A total-host normal-form claim may cite only those finite rows and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *surface. It cannot cite a host conversion row, a global normalizer, an unbounded host-normalization procedure, a divergence oracle, or a hidden fuel source outside the packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host normal form ledger exhaustion`

Proof. First project the terminal-host root factorization of Definition 936.137. It places every terminal-host consumer inside the same finite BHist packet and gives no additional host-side coordinate.

Use Definition 936.138 to account for bounded fuel replay and terminal readback. The route consumes F, V, R through the displayed structural rows and leaves K as attached sibling evidence.

Then apply Definition 936.139. Normal-output consumption remains inside carrier admission, transport, continuation replay, package provenance, and local naming; the excluded host-side objects have no export coordinate.

Thus the total-host ledger is precisely the displayed finite packet. Unbounded host normalization, hidden fuel, divergence detection, host conversion, and global normalization remain outside the accepted ledger. □

Theorem 936.157 ($\mathsf{ZNormal}^{\uparrow}$ downstream normal-route boundary)**.** *Every downstream normal-route consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet must pass through the terminal fuel readback admission of Definition 936.155 and the total-host ledger exhaustion of Definition 936.156. The exposed downstream route is therefore the finite packet* $T, F, V, K, R, H, C, P, N$ *with terminal normality, bounded fuel replay, finite normal-word evidence, continuation readback, transport, replay, provenance, and local naming. No downstream consumer receives a host conversion row, global normalization procedure, divergence oracle, unbounded evaluator, detached total-host normal form, or hidden observation of normalization beyond those rows.*

Proof. Begin with downstream normal-form readiness, Definition 936.142. It already requires downstream consumers to factor through the terminal-host exposure and local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ route.

Apply terminal fuel readback admission. The consumer can read terminal normality only through T, F, V, R, H, C, P, N, with K attached as finite sibling normal-word evidence.

Finally apply total-host normal-form ledger exhaustion. Since the ledger has no coordinate for host conversion, global normalization, divergence detection, unbounded evaluation, detached total-host normal forms, or hidden normalization observations, the downstream normal route is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket downstream normal route boundary`

Definition 936.158 ($\mathsf{ZNormal}^{\uparrow}$ total-host obligation packet)**.** *For an accepted* $\mathsf{ZNormal}^{\uparrow}$ *carrier*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host obligation packet is the ordered finite surface whose rows are the typed endpoint T*, bounded fuel replay* F*, terminal normality* V*, sibling normal-word evidence* K*, terminal readback* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming* N*. A consumer of this packet may inspect these rows only through the carrier, NameCert, terminal readback, transport, replay, provenance, and local naming routes already exposed for* $\mathsf{ZNormal}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.ZnormalUp.ZnormalTotalHostObligationPacket`
Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostObligationPacket row exposure`

Theorem 936.159 ($\mathsf{ZNormal}^{\uparrow}$ normal-form consumer coverage)**.** *Every normal-form consumer of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet factors through the total-host obligation packet of Definition 936.158. The factorization covers terminal normality through V, bounded fuel replay through F, sibling normal-word evidence through K, terminal readback through R, transport through H, continuation replay through C, provenance through P, and local naming through N, all over the typed endpoint row T.*

No normal-form consumer receives a route outside those rows before it has passed through terminal fuel readback admission, total-host normal-form ledger exhaustion, and the downstream normal-route boundary.

Proof. Project the accepted carrier by Definition 936.1. It displays exactly the rows $T, F, V, K, R, H, C, P, N$ and supplies no additional normal-form coordinate.

Apply Definition 936.155. This places every terminal normality read behind the route from T through F to V and R, with H, C, P, N as the only structural support and K as attached sibling normal-word evidence.

Next use Definition 936.156. The ledger accounts for terminal normality, bounded fuel replay, terminal readback, sibling evidence, transport, replay, provenance, and local naming inside the same packet.

Finally apply Definition 936.157. A downstream normal-route consumer therefore reaches normal-form evidence only through the displayed obligation packet, so the coverage claim is exactly the finite row surface of Definition 936.158. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal form consumer coverage`

Theorem 936.160 ($\mathsf{ZNormal}^{\uparrow}$ total-host readback separation)**.** *The total-host readback surface of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet separates the finite obligation rows*

$$T, F, V, K, R, H, C, P, N$$

from host conversion, quotient computation, unrestricted evaluation, divergence oracles, substrate halting, and global normalization. A readback that cites terminal normality, bounded fuel replay, sibling normal-word evidence, terminal readback, transport, replay, provenance, or local naming must remain inside the total-host obligation packet and cannot promote any excluded host object to a $\mathsf{ZNormal}^{\uparrow}$ *row.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostReadbackSeparation`

Proof. Begin with the finite packet of Definition 936.158. Its rows are the only objects named by the accepted total-host surface.

Use terminal fuel readback admission from Definition 936.155. The terminal read is admitted only after T, F, V, R, H, C, P, N are displayed, and the sibling normal-word row K remains finite evidence rather than host conversion.

Apply total-host normal-form ledger exhaustion, Definition 936.156. The excluded host objects have no ledger coordinate and cannot be recovered from hidden fuel, detached total-host normal forms, or a second readback route.

The downstream boundary of Definition 936.157 closes the separation. Since every downstream route is already the finite obligation packet, host conversion, quotient computation, unrestricted evaluation, divergence detection, substrate halting, and global normalization remain outside the readback surface. □

Theorem 936.161 ($\mathsf{ZNormal}^{\uparrow}$ terminal-host obligation totality)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet, the total-host obligation packet of Definition 936.158, normal-form consumer coverage from Definition 936.159, and total-host readback separation from Definition 936.160 assemble the terminal-host obligation surface required by* $\mathsf{ZNormal}^{\uparrow}$*. The assembled surface is exactly the finite row family*

$$T, F, V, K, R, H, C, P, N,$$

with terminal normality, bounded fuel replay, sibling normal-word evidence, terminal readback, transport, replay, provenance, and local naming available to downstream consumers, and with host conversion, quotient computation, unrestricted evaluation, divergence oracles, substrate halting, and global normalization absent from the packet.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal host obligation totality`

Proof. First read the obligation packet. By Definition 936.158, the terminal-host surface is the finite carrier and NameCert row family $T, F, V, K, R, H, C, P, N$.

Next apply normal-form consumer coverage. Definition 936.159 forces every normal-form consumer through terminal normality, bounded fuel replay, sibling evidence, terminal readback, transport, replay, provenance, and local naming over the same typed endpoint.

Then apply total-host readback separation. Definition 936.160 keeps host conversion, quotient computation, unrestricted evaluation, divergence detection, substrate halting, and global normalization outside the packet.

Combining these two directions gives the required terminal-host obligation totality: all finite rows needed by downstream consumers are present, and every excluded host-side object is separated from the $\mathsf{ZNormal}^{\uparrow}$ surface. □

Theorem 936.162 ($\mathsf{ZNormal}^{\uparrow}$ total-host decision surface)**.** *Every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet exposes a total-host decision surface through the displayed finite rows*

$$T, F, V, K, R, H, C, P, N.$$

Terminal normality is read through V *over the typed endpoint* T *and bounded fuel replay* F*; the normal-word row* K *cites the* $\mathsf{ZeckendorfNormal}^{\uparrow}$ *and* $\mathsf{ZCarry}^{\uparrow}$ *sibling surfaces; the terminal continuation row* R *cites the* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *surface; and the route is transported by* hsame*, replayed by* Cont*, packaged by* Pkg*, and named by* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$*. The accepted decision rows contain no host conversion procedure, divergence oracle, quotient computation, unrestricted evaluator, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostDecisionSurface`

Proof. Start with the carrier and obligation packet of Definition 936.1 and Definition 936.158. Together they display exactly $T, F, V, K, R, H, C, P, N$, so the decision surface has no coordinate outside the finite BHist packet.

Read the sibling anchors next. The typed endpoint route is the one supplied by chapter 562; the finite normal-word route is constrained by chapter 278 and chapter 544; and the terminal continuation route is the one supplied by chapter 683. These anchors feed rows of the packet rather than a host conversion row.

Finally apply Definition 936.161. It assembles terminal normality, bounded fuel replay, sibling normal-word evidence, terminal readback, transport, replay, provenance, and local naming as the whole consumer-facing surface. Hence the total-host decision row is precisely the displayed BHist, Cont, hsame, Pkg, and $\mathsf{NameCert}$ surface. □

Theorem 936.163 ($\mathsf{ZNormal}^{\uparrow}$ terminal-normality refusal stability)**.** *Let $Z = (T, F, V, K, R, H, C, P, N)$ and $Z' = (T', F', V', K', R', H', C', P', N')$ be accepted $\mathsf{ZNormal}^{\uparrow}$ packets related by componentwise* hsame *transport of the displayed rows. If a consumer request for Z is refused because it asks for a divergence oracle, a host-conversion procedure, or a global-normalization row instead of terminal normality, then the corresponding transported request for Z' is refused for the same reason. The refusal remains attached to the* BHist*,* Cont*,* Pkg*, and local* NameCert *surface and is not converted into a transported host-side object.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal normality refusal stability`

Proof. Project both accepted packets through Definition 936.1. Componentwise transport moves only the rows $T, F, V, K, R, H, C, P, N$ and their primed counterparts, so it has no target coordinate for a divergence oracle, host conversion procedure, or global normalizer.

Use Definition 936.160. Refusal of the excluded objects is not a side condition outside the packet; it is the separation of the finite total-host readback surface from host conversion, quotient computation, unrestricted evaluation, divergence detection, substrate halting, and global normalization.

The sibling surfaces remain component inputs under transport. Typed reduction normal-form, Zeckendorf normal form, local carry, continuation trace, and the normal-form consistency seal of chapter 740 are consumed through the displayed K and R rows and through H, C, P, N. Thus transporting the accepted packet preserves the same refusal ledger and never creates a transported host-side object. □

Theorem 936.164 ($\mathsf{ZNormal}^{\uparrow}$ root-unblock NameCert exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *surface needed for root unblock is exhausted by the displayed terminal-normality, bounded-fuel, replay, provenance, and naming rows of the accepted packet. Concretely, carrier admission reads the* BHist *packet $T, F, V, K, R, H, C, P, N$; classifier and terminal-normality reads use T, F, V, K, R; transport and replay use H and C; provenance and local naming use P and N. No additional NameCert clause remains for host conversion, divergence detection, global normalization, quotient computation, unrestricted evaluation, or substrate halting.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootUnblockNameCertExhaustion`

Proof. Read the root carrier through Definition 936.1 and the total-host obligation packet through Definition 936.158. These definitions list the full root-unblock surface and identify the terminal-normality, fuel, replay, provenance, and naming rows.

Apply Definition 936.162. It places the sibling typed endpoint, Zeckendorf normal-word, carry, continuation-trace, and normal-form consistency seal of chapter 740 inside the same finite row family, with hsame, Cont, Pkg, and NameCert as the only structural routes.

Finally use Definition 936.163. Since the excluded host-side requests remain refused under componentwise transport, there is no residual NameCert obligation beyond the displayed carrier, classifier, transport, replay, provenance, and naming rows. The root-unblock certificate is therefore exhausted by the local packet. □

Theorem 936.165 ($\mathsf{ZNormal}^{\uparrow}$ total-host decision route exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the terminal-host decision route is exhausted by the finite BHist *display and its* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *rows. The route reads carrier admission from $T, F, V, K, R, H, C, P, N$, terminal decision from T, F, V, K, R, transport from H, continuation replay from C, provenance from*

P, and local naming from N. It contains no host conversion, divergence oracle, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTotalHostDecisionRouteExhaustion`

Proof. Begin with Definition 936.1 and Definition 936.158. They present $T, F, V, K, R, H, C, P, N$ as the whole accepted carrier and obligation packet, so every terminal-host read starts inside a finite BHist display.

Apply Definition 936.162. The theorem identifies the decision surface with the typed endpoint, bounded fuel, terminal-normality, sibling normal-word, continuation readback, transport, replay, provenance, and naming rows rather than with a host-side evaluator.

Finally use Definition 936.163 and Definition 936.164. The refused host-side requests stay refused under hsame transport, and the local root-unblock certificate has no remaining clause outside BHist, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$. Hence the total-host decision route is exhausted by the displayed finite packet. □

Theorem 936.166 ($\mathsf{ZNormal}^{\uparrow}$ terminal-host classifier route nonescape)**.** *Let $Z = (T, F, V, K, R, H, C, P, N)$ be an accepted $\mathsf{ZNormal}^{\uparrow}$ packet. Any terminal-host classifier route that reaches the root-unblock surface factors through the displayed BHist rows $T, F, V, K, R, H, C, P, N$, with classifier reads on T, F, V, K, R, componentwise hsame transport on H, Cont replay on C, Pkg provenance on P, and $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ naming on N. No classifier branch escapes to a host conversion procedure, divergence oracle, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal host classifier route nonescape`

Proof. Project the route through Definition 936.1 and Definition 936.158. The only classifier-readable coordinates are the displayed terminal, fuel, terminal-normality, sibling normal-word, and continuation rows, with transport, replay, provenance, and naming carried by H, C, P, N.

Use Definition 936.162 to identify the classifier as a consumer of that finite decision surface. It receives no independent host conversion row and no quotient or halting computation row.

Now apply Definition 936.163. A request that attempts to leave the displayed route is refused before transport can turn it into a host-side object.

Definition 936.164 then closes the route at the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ surface. Since root unblock has no residual naming obligation beyond the finite packet, the terminal-host classifier route has no escape branch. □

Theorem 936.167 ($\mathsf{ZNormal}^{\uparrow}$ normal-form readiness support totality)**.** *For every accepted $\mathsf{ZNormal}^{\uparrow}$ packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

normal-form readiness for terminal-host use is supported exactly by the finite carrier, classifier, transport, replay, provenance, and naming rows of Z. The readiness support consists of the BHist rows $T, F, V, K, R, H, C, P, N$, the hsame row H, the Cont row C, the Pkg row P, and the local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ row N. It has no host conversion, divergence oracle, quotient computation, unrestricted evaluator, substrate-halting row, or global normalizer.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket normal form readiness support totality`

Proof. Read the accepted packet through Definition 936.1 and Definition 936.158. Normal-form readiness is therefore anchored in the same finite BHist tuple $T, F, V, K, R, H, C, P, N$ as terminal-host decision.

Definition 936.162 supplies the readiness support from the terminal endpoint, bounded fuel, terminal normality, sibling normal-word evidence, continuation readback, transport, replay, provenance, and naming rows. These are BEDC rows of the packet, not a global normalizer or unrestricted evaluator.

Apply Definition 936.163 to rule out conversion of refused host-side requests into readiness data. Then apply Definition 936.164 to identify the local NameCert surface as complete for root unblock.

Thus every support row needed by terminal-host normal-form readiness is already one of the finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ rows displayed by Z. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalNormalFormReadinessSupportTotality`

Theorem 936.168 ($\mathsf{ZNormal}^\uparrow$ root total-host source coverage)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the root total-host source is covered by the finite BHist *display* $T, F, V, K, R, H, C, P, N$*. The source row* T *supplies the typed endpoint,* F *supplies bounded fuel replay,* V *supplies the terminal normal-form row,* K *and* R *supply the finite sibling normal-word and continuation surfaces,* H *supplies the displayed* hsame *transports,* C *supplies the* Cont *consumer routes,* P *supplies* Pkg *provenance, and* N *supplies the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *row. No covered source requires host conversion, divergence detection, quotient normalization, unrestricted evaluation, substrate halting, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root total host source coverage`

Proof. Begin with Definition 936.1 and Definition 936.158. They name the accepted root source as the finite tuple $T, F, V, K, R, H, C, P, N$, so the source coverage question has no coordinate outside the displayed BHist packet.

Apply Definition 936.162. It reads typed endpoint, bounded fuel, terminal normality, finite sibling evidence, continuation readback, transport, replay, provenance, and local naming from the same rows and refuses the omitted host-side objects.

Finally use Definition 936.175. Downstream consumers can reach the local NameCert row only after the sibling typed endpoint, normal-word, carry, and continuation surfaces have entered the packet through T, K, R. Hence the root total-host source is fully covered by the finite display. □

Theorem 936.169 ($\mathsf{ZNormal}^\uparrow$ root terminal-normality determinacy)**.** *Let* $Z = (T, F, V, K, R, H, C, P, N)$ *be an accepted* $\mathsf{ZNormal}^\uparrow$ *packet. A root consumer that asks for terminal normality is determined by the terminal normal-form row* V *over the typed endpoint* T*, the bounded fuel replay row* F*, the continuation readback row* R*, and the structural rows* H, C, P, N*. The sibling row* K *contributes only the finite Zeckendorf and carry-facing normal-word surface. The consumer cannot replace this determination by host conversion, divergence detection, quotient normalization, unrestricted evaluation, substrate halting, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root terminal normality determinacy`

Proof. Use Definition 936.82. It confines every terminal-normality read to T, F, V, R, H, C, P, N, with K available only as finite sibling evidence.

Next apply Definition 936.173. The public terminal route is exhausted by $T, F, V, K, R, H, C, P, N$, so there is no second route on which terminal normality could be chosen by a host evaluator or quotient normalizer.

The decision surface of Definition 936.162 then fixes the root consumer. Since V is the only terminal normal-form row and the supporting rows are exactly T, F, R, H, C, P, N, terminal normality is determined inside the finite packet. □

Theorem 936.170 ($\mathsf{ZNormal}^{\uparrow}$ root fuel-replay exhaustion)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet* $Z = (T, F, V, K, R, H, C, P, N)$*, bounded fuel replay at the root is exhausted by* F *together with the displayed* Cont *row* C *and the transport, provenance, and naming rows* H, P, N*. The replay is used only to expose the terminal normal-form row* V *over* T *and the continuation row* R*, while* K *remains finite sibling normal-word evidence. It does not export a divergence oracle, an unrestricted evaluator, substrate halting, host conversion, quotient normalization, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket root fuel replay exhaustion`

Proof. Start with Definition 936.78. It places every terminal-normality read behind the finite fuel row F and the route T, F, V, R, H, C, P, N.

Now apply Definition 936.161. The total-host obligation packet presents replay, transport, provenance, and local naming as rows of the same finite packet, with the excluded host-side objects absent from the surface.

Finally use Definition 936.174. The field-readback lock interprets `FieldFaithful ZNormalUp` and `Nontrivial ZNormalUp` on the displayed packet, not on an evaluator. Thus bounded fuel replay is exhausted by F, C, H, P, N as support for V and R. □

Theorem 936.171 ($\mathsf{ZNormal}^{\uparrow}$ root refusal ledger separation)**.** *The root refusal ledger of an accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet separates the admitted finite rows*

$$T, F, V, K, R, H, C, P, N$$

from host conversion, divergence detection, quotient normalization, unrestricted evaluation, substrate halting, and global normalization. The admitted side contains terminal normality, bounded fuel replay, displayed hsame *transport,* Cont *consumer routing,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *naming; the refused side contains exactly the omitted host-facing objects and cannot be transported back into the packet.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootRefusalLedgerSeparation`

Proof. Read the finite side through Definition 936.168. The admitted ledger entries are precisely the rows $T, F, V, K, R, H, C, P, N$ and their carrier, terminal, replay, transport, provenance, and naming uses.

Read the refused side through Definition 936.163. A request for divergence, host conversion, or global normalization remains refused under componentwise hsame transport, so transport cannot convert the refusal into a packet row.

Finally apply Definition 936.160. Quotient computation, unrestricted evaluation, substrate halting, and the other host-side objects have no ledger coordinate. The ledger therefore separates admitted root evidence from refused host-facing objects. □

Theorem 936.172 ($\mathsf{ZNormal}^{\uparrow}$ root NameCert unblock package)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet* $Z = (T, F, V, K, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *root-unblock package is the finite obligation packet consisting of source coverage, terminal-normality determinacy, fuel-replay*

exhaustion, and refusal-ledger separation. It binds the terminal normal-form row V, bounded fuel replay row F, displayed hsame *transports H,* Cont *routes C,* Pkg *provenance P, and local naming row N over the same finite* BHist *source $T, F, V, K, R, H, C, P, N$, and it exports no host conversion or divergence oracle.*

Proof. First assemble the admitted source rows by Definition 936.168. This gives the finite BHist packet on which the local NameCert row can operate.

Next combine Definition 936.169 with Definition 936.170. Terminal normality is determined by V over T, F, R, H, C, P, N, and bounded fuel replay is exhausted by F together with the displayed replay and structural rows.

Finally apply Definition 936.171 and Definition 936.164. The refusal ledger keeps host conversion, divergence detection, quotient normalization, unrestricted evaluation, substrate halting, and global normalization outside the packet, while root-unblock NameCert exhaustion leaves no remaining local clause beyond $T, F, V, K, R, H, C, P, N$. The four components therefore form the root NameCert unblock package. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootNameCertUnblockPackage`

Theorem 936.173 ($\mathsf{ZNormal}^{\uparrow}$ terminal route component totality)**.** *For every accepted* $\mathsf{ZNormal}^{\uparrow}$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

terminal use is exposed only through the finite component surface $T, F, V, K, R, H, C, P, N$. The typed endpoint row T, finite fuel row F, terminal-normality row V, sibling normal-word row K, continuation readback row R, componentwise transport row H, replay row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$ *row N exhaust the public route. No accepted packet exports host conversion, quotient normalization, divergence detection, unrestricted evaluation, substrate halting, or a global normalizer.*

Proof. Start from the carrier and local obligation packet of Definition 936.1 and Definition 936.2. They name the accepted surface as the tuple $T, F, V, K, R, H, C, P, N$, so every terminal read begins inside the finite BEDC rows.

Next apply Definition 936.6 and Definition 936.8. These theorems keep finite fuel, terminal normality, and continuation readback inside the displayed packet rather than inside a host evaluator.

Finally use Definition 936.11 and Definition 936.24. The sibling typed endpoint, finite normal-word, carry, and continuation surfaces of chapter 562, chapter 278, chapter 544, and chapter 683 enter only through the named component rows. Hence terminal use is total over the finite packet and has no extra host-facing route. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket terminal route component totality`

Theorem 936.174 ($\mathsf{ZNormal}^{\uparrow}$ TasteGate field readback lock)**.** *The Lean TasteGate rows* `FieldFaithful ZNormalUp` *and* `Nontrivial ZNormalUp` *are read on the paper side as the* $\mathsf{ZNormal}^{\uparrow}$ *field-readback lock for the finite surface $T, F, V, K, R, H, C, P, N$. They certify that the accepted field surface is nonempty and field-faithful for local readback; they do not add host conversion, quotient normalization, a divergence oracle, unrestricted evaluation, substrate halting, or a global normalizer.*

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalTasteGateFieldReadbackLock`

Proof. Project the accepted packet through Definition 936.1. The field inventory available to the TasteGate reader is exactly $T, F, V, K, R, H, C, P, N$, with local naming controlled by the $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ row.

Apply Definition 936.2, Definition 936.6, and Definition 936.8. Together they identify field readback with finite fuel, terminal normality, continuation readback, transport, replay, provenance, and naming, and they exclude the omitted host-side objects from the carrier.

The Lean rows are therefore interpreted as a lock on this finite surface. FieldFaithful says the rows can be read back as the displayed packet fields, while Nontrivial says the surface is not empty; neither row changes the packet into host conversion or quotient normalization. □

Theorem 936.175 ($\mathsf{ZNormal}^\uparrow$ root obligation downstream handoff)**.** *Every downstream normal-form consumer of the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *row must first pass through the sibling surfaces of chapter 562, chapter 278, chapter 544, and chapter 683. After those surfaces have supplied the typed endpoint, finite normal-word, carry, and terminal continuation rows, the consumer may use the local row* N *only as part of the same packet* $T, F, V, K, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalRootObligationDownstreamHandoff`

Proof. Begin with Definition 936.1. It places the downstream-facing rows in the finite packet and assigns N to the local $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ surface, not to an independent host-normalization object.

Use Definition 936.11 for the K row and Definition 936.8 for the R row. The typed endpoint chapter supplies T, the Zeckendorf and carry chapters supply the finite normal-word route behind K, and the continuation-trace chapter supplies terminal readback behind R.

Finally apply Definition 936.24 together with Definition 936.2. The source package and local obligations require consumers to reach N through T, F, V, K, R, H, C, P first. Thus downstream use of the local NameCert row is a handoff from the sibling normal-form surfaces, not a direct host-normalization route. □

Theorem 936.176 ($\mathsf{ZNormal}^\uparrow$ total-host terminal row exposure)**.** *For every accepted* $\mathsf{ZNormal}^\uparrow$ *packet*

$$Z = (T, F, V, K, R, H, C, P, N),$$

the total-host root consumer exposes terminal normality only through the terminal row V*, bounded fuel replay through* F*, the total-host source through* T*, the continuation trace through* R*, the transport and route rows* H, C*, the provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{ZNormal}^\uparrow}$ *row* N*. The sibling row* K *supplies finite normal-word evidence only. No exposed row is a host conversion row, divergence oracle, quotient normalizer, unrestricted evaluator, or global normalizer.*

Proof. Start from Definition 936.168. It identifies the total-host source as the finite display $T, F, V, K, R, H, C, P, N$, so a root consumer has no coordinate outside the accepted packet.

Next use Definition 936.169 and Definition 936.170. These bind terminal normality to V and bounded replay to F, C, H, P, N before any downstream consumer can inspect the local naming row.

Finally apply the TasteGate anchor Definition 936.81. Its single-carrier alignment reads the same field surface, so the terminal exposure is exactly the finite packet rather than a host conversion or evaluator surface. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket total host terminal row exposure`

Theorem 936.177 ($\mathsf{ZNormal}^{\uparrow}$ divergence oracle refusal)**.** *Let $Z = (T, F, V, K, R, H, C, P, N)$ be an accepted $\mathsf{ZNormal}^{\uparrow}$ packet. Any root readback request for a divergence oracle is refused at the packet boundary: the admitted rows provide terminal normality, bounded fuel replay, total-host source, continuation trace, componentwise transports, displayed routes, provenance, and local NameCert naming, but they do not export a divergence oracle or any surrogate through host conversion, substrate halting, unrestricted evaluation, quotient normalization, or global normalization.*

Proof. Read the admitted side through Definition 936.171. The refusal ledger separates $T, F, V, K, R, H, C, P, N$ from divergence detection and the other host-facing objects.

Use Definition 936.80 and Definition 936.162. Together they keep every total-host request inside the finite packet rows and refuse any request whose target is absent from those rows.

The TasteGate field surface of Definition 936.81 supplies the nonempty single-carrier readback anchor. Since that anchor has no divergence-oracle coordinate, the refused request cannot re-enter through field faithfulness, transport, route replay, provenance, or local naming. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket divergence oracle refusal`

Theorem 936.178 ($\mathsf{ZNormal}^{\uparrow}$ host conversion nonescape)**.** *For every accepted $\mathsf{ZNormal}^{\uparrow}$ packet $Z = (T, F, V, K, R, H, C, P, N)$, a host-conversion read cannot escape the finite readback route. Terminal normality, bounded fuel replay, total-host source, continuation trace, transports, routes, provenance, and the local NameCert row are all read through $T, F, V, K, R, H, C, P, N$, while host conversion, quotient computation, unrestricted evaluation, substrate halting, divergence detection, and global normalization remain outside the carrier.*

Proof. Begin with Definition 936.173. It exhausts terminal use by the component surface $T, F, V, K, R, H, C, P, N$ and leaves no extra host-facing route.

Next apply Definition 936.172. The local NameCert row is reached only after source coverage, terminal-normality determinacy, fuel-replay exhaustion, and refusal-ledger separation have been assembled inside the same finite packet.

Finally use Definition 936.81. The TasteGate single-carrier alignment reads the accepted rows as one field surface, so host conversion has no route through transport, replay, provenance, local naming, or sibling normal-word evidence. □

Lean4 ✓ `BEDC.Derived.ZnormalUp.ZnormalPacket host conversion nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ZNormal}^{\uparrow}$ packet is generated from a typed reduction normal-form row, finite fuel row, terminal-normality row, finite Zeckendorf and carry-facing comparison row, continuation terminal-read row, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 936.1, Definition 936.2, Definition 936.3, Definition 936.4, Definition 936.5, Definition 936.6, Definition 936.7, Definition 936.8, Definition 936.11, Definition 936.12, Definition 936.13, Definition 936.18, Definition 936.19, Definition 936.23, Definition 936.81, Definition 936.82, Definition 936.83, Definition 936.137, Definition 936.138, and Definition 936.139 bind the bridge-facing total-host terminal-normality boundary through the existing finite normal-form, carry, continuation, typed-boundary, terminal-readback, sibling-handoff, route-transport, normal-output, consumer non-escape, terminal-host, TasteGate field-surface, scope-lock, root-factorization, fuel-replay, and output-root non-escape rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ZNormal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove host Turing completeness, substrate halting, global normalization for external computations, host conversion, quotient computation, Lean verification, or bridge checking for $\mathsf{ZNormal}^{\uparrow}$.

Upgrade path: Lean verification and bridge checking remain the next formal-axis work, without adding host conversion, quotient computation, or global normalization.

Chapter 937

A Concrete Naming Certificate for $\mathsf{CellularTrustSubstrate}^{\uparrow}$

The $\mathsf{CellularTrustSubstrate}^{\uparrow}$ packet turns the Rule 110 closed-substrate reading of chapter 9390 into a finite NameCert carrier. It consumes the cellular rule-table and finite orbit surface of chapter 934 and the bounded trace reporting discipline of chapter 944; its additional work is to bind those rows into one trust-substrate NameCert packet. It records the rule table, manifest row, finite orbit segment, classifier surface, audit route, package provenance, and local naming ledger through which the cellular trust path is inspected. The packet does not identify infinite streams, decide host equality of cellular automata, or import an oracle for long-run behaviour.

Falsifiable prediction: If a public Rule 110 substrate read accepted by this packet can be replayed without one of the finite rule-table, manifest, orbit-segment, classifier, audit-route, package, or local NameCert rows below, then the carrier admission claim of Definition 937.1 is false.

Independence witness: haltingdistinction, inscriptionevent, continuationtracenormalform, kernelinductiveacceptance: the carrier shares the finite diagonal-boundary, inscription-record, trace-normal-form, and kernel-acceptance surfaces used by those siblings, but none of them supplies the eight-entry cellular rule table, manifest row, orbit segment, and classifier disagreement audit as a single closed-substrate packet.

Definition 937.1 (Cellular trust substrate NameCert carrier)**.** *A* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier is a finite packet*

$$S = (R, M, O, F, A, P, N)$$

with the following visible rows:

1. *R is the finite Rule 110 table row, listing the eight neighbourhood transitions used by the substrate.*
2. *M is the manifest row naming the initial cell word, declared run budget, expected observation fields, and public replay parameters.*
3. *O is the finite orbit-segment row produced by iterating R from M for the displayed number of steps.*
4. *F is the classifier row for visible cell patterns, phase tags, and accepted observation records.*
5. *A is the audit-route row comparing the manifest, direct Rule 110 evolution, cyclic-tag compilation route, and classifier disagreement checks.*

6. *P is the* Pkg *provenance over* BHist, Cont, hsame, *the manifest schema, the classifier row, and the replay audit.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$ *ledger for carrier admission, classifier stability, audit-route exactness, and closed-observation non-escape.*

All rows are finite. The carrier names a concrete closed substrate, not a quotient stream, an oracle halting predicate, or host-level equality of cellular automata.

Theorem 937.2 (Cellular trust substrate NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$ *surface for Definition 937.1 consists exactly of six obligation rows: carrier admission for $S = (R, M, O, F, A, P, N)$; rule-table faithfulness for the eight displayed Rule 110 transitions; manifest-to-orbit replay exactness for the finite segment O; classifier stability under the visible phase and pattern rows in F; audit-route agreement for the direct evaluator, cyclic-tag route, and disagreement checks in A; and closed-observation non-escape through P and N.*

Proof. Carrier admission is projection from the displayed tuple. The rule table, manifest, orbit segment, classifier, audit route, package row, and local ledger are the only coordinates of the packet.

Rule-table faithfulness and manifest replay exactness are finite checks. The eight entries of R determine each successor cell in O, and M supplies the initial word and replay budget read by the audit route.

Classifier stability is local to F. Any accepted pattern or phase read must pass through the displayed classifier row and the disagreement checks in A, so a consumer cannot replace the row by an unrecorded host classifier.

Finally, P and N package the same finite rows as the public certificate. Since no coordinate names a quotient stream, an oracle for halting, or host equality of cellular automata, the obligations close only the finite closed-substrate surface. □

Theorem 937.3 (Cellular trust substrate closed observation non-escape)**.** *Every accepted public observation of a* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *packet factors through the finite route*

$$R \longmapsto M \longmapsto O \longmapsto F \longmapsto A \longmapsto P \longmapsto N.$$

The route may certify the displayed Rule 110 orbit segment, classifier row, and replay audit, but it cannot export a quotient of infinite streams, an oracle halting decision, a host cellular-automaton equality principle, or an induction principle over all future rows.

Proof. Start with an accepted observation. Carrier projection gives the only available rows: the rule table R, manifest M, orbit segment O, classifier F, audit route A, package row P, and local ledger N.

The NameCert obligations force the observation to read successor cells through R and M, classifier claims through F, and replay agreement through A. Each of these rows is finite and displayed inside the packet.

The package and ledger rows record provenance for that same finite route. They add no coordinate for infinite-stream quotienting, oracle halting, host equality between cellular automata, or induction over ungenerated future cells. Thus the accepted observation remains a closed finite substrate read. □

Lean follow-up for $\mathsf{CellularTrustSubstrate}^{\uparrow}$ must provide the corresponding `Nontrivial` and `FieldFaithful` TasteGate instances for the finite rule-table, manifest, orbit-segment, classifier, audit-route, and local NameCert rows.

Chapter 938

A Concrete Naming Certificate for CellularOrbitSubstrate$^{\uparrow}$

The CellularOrbitSubstrate$^{\uparrow}$ packet turns the cellular automaton substrate reading of chapter 9390 into a concrete finite-orbit NameCert carrier. It sits beside the machine-code surface of chapter 280, the bounded-graph surface of chapter 199, the continuation-tail surface of chapter 482, and the resource-modulus surface of chapter 282. Its distinguishing datum is a local-neighborhood orbit row with a finite spatial window ledger.

Falsifiable prediction: Two accepted packets with the same local rule table and hsame initial row cannot expose two distinct next-row ledgers under the displayed neighborhood classifier.

Independence witness: turingmachine, computable, continuationautomaton, complexityclass: the cellular carrier has a local-neighborhood orbit row and finite spatial window ledger that these siblings do not carry, while those siblings carry machine-code, bounded graph, continuation-tail, or resource-modulus rows absent here.

Definition 938.1 (Cellular orbit substrate carrier)**.** *A* CellularOrbitSubstrate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, I, W, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the local cellular rule-table row, including the finite alphabet, the neighborhood radius, and every local transition entry used by the packet.*
2. *I is the initial row, displayed as a finite spatial window with boundary convention and origin marker.*
3. *W is the finite orbit-window ledger recording the displayed rows obtained by iterating R from I for the accepted time budget and spatial window.*
4. *L is the local-neighborhood ledger listing, for each accepted successor cell in W, the visible predecessor neighborhood and the rule-table entry used to compute it.*
5. *H records componentwise* hsame *transport for rule tables, initial rows, orbit windows, local-neighborhood ledgers, provenance rows, and local names.*
6. *C records the* Cont *replay route from R and I through each displayed row of W and each neighborhood read in L.*

7. *P is the* Pkg *provenance over the cellular substrate reading, the sibling computational certificates,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularOrbitSubstrate}^{\uparrow}}$ *row for carrier admission, local determinacy, observation non-escape, and public name exposure.*

The classifier compares accepted packets only through the displayed rule table, hsame initial row, finite orbit window, local-neighborhood ledger, transport row, continuation replay, provenance, and local naming row. It has no coordinate for an infinite orbit, a hidden global evaluator, a quotient stream, host equality between cellular histories, or an external cellular oracle.

Theorem 938.2 (Cellular orbit substrate local determinacy)**.** *For the carrier of Definition 938.1, if two accepted packets expose the same rule-table row R and* hsame *initial rows I and I', then every displayed next-row entry accepted by the finite orbit-window ledger is determined by the same local-neighborhood read in L. In particular, the public classifier cannot expose two distinct next-row ledgers under the same rule table and hsame initial row.*

Proof. Project both accepted packets to their rule-table, initial-row, orbit-window, and local-neighborhood ledger rows. The rule table R fixes a single successor value for each displayed predecessor neighborhood, and the hsame initial rows identify the first visible neighborhoods componentwise.

Use the continuation replay row C to step through the finite window ledger. At each displayed successor cell, the ledger L records the predecessor neighborhood and the rule-table entry used to compute it, so two accepted ledgers with the same R and transported initial row must read the same entry.

The classifier has no coordinate for a detached evaluator or hidden cellular history. Therefore a claimed second next-row ledger would need a displayed neighborhood read absent from L or a second value for the same entry of R, neither of which belongs to the carrier. □

Theorem 938.3 (Cellular orbit substrate observation non-escape)**.** *Every accepted observation exported by a* $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ *packet factors through the finite route*

$$R, I, W, L, H, C, P, N.$$

The route can certify local neighborhoods, successor rows, finite spatial windows, and displayed orbit prefixes, but it exports no infinite orbit, hidden cellular evaluator, quotient stream, host equality principle, global completeness witness, or independent tail oracle.

Proof. Begin with carrier projection. An accepted observation can read only the rule table R, initial row I, finite orbit-window ledger W, local-neighborhood ledger L, transport row H, continuation replay C, package row P, and local naming row N.

The local-neighborhood ledger supplies the only route from a visible predecessor window to a successor cell. The continuation row replays this finite sequence and never introduces an unbounded time domain or spatial completion.

The provenance and local name rows package that same finite route. Since the carrier lacks coordinates for infinite streams, host cellular equality, global completeness, detached evaluators, and external tail oracles, every accepted observation remains inside the displayed packet. □

Theorem 938.4 (Cellular orbit substrate NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CellularOrbitSubstrate}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (R, I, W, L, H, C, P, N)$, local-neighborhood determinacy through R and L, hsame transport of the initial row through H, finite* Cont *replay*

through C, observation non-escape through W, L, P, N, and public naming by the local NameCert row. The surface does not import machine code, bounded graph simulation, continuation-tail equality, resource-modulus comparison, or any unbounded cellular-orbit object.

Proof. Carrier admission is projection from Definition 938.1. The packet displays the rule table, initial row, finite orbit window, local-neighborhood ledger, transport row, replay row, provenance, and local naming row as its only coordinates.

Local determinacy is supplied by Definition 938.2. Observation containment is supplied by Definition 938.3. Together they identify the accepted public surface with the finite orbit route and exclude any read outside it.

The sibling boundaries are therefore separate. TuringMachine$^{\uparrow}$ machine code, Computable$^{\uparrow}$ bounded graphs, ContinuationAutomaton$^{\uparrow}$ continuation tails, and ComplexityClass$^{\uparrow}$ resource moduli may be cited as comparison anchors, but no such row is a coordinate of the cellular orbit substrate carrier. □

Lean4 ✓ `BEDC.Derived.CellularOrbitSubstrateUp.CellularOrbitSubstrateTasteGate single carrier alignment`

The later Lean TasteGate namespace BEDC.Derived.CellularOrbitSubstrateUp.TasteGate must provide Nontrivial CellularOrbitSubstrate$^{\uparrow}$ and FieldFaithful CellularOrbitSubstrate$^{\uparrow}$ instances for the displayed finite-orbit carrier.

Theorem 938.5 (Cellular orbit substrate orbit carrier row)**.** *Finite orbit admission for a* CellularOrbitSubstrate$^{\uparrow}$ *packet is exactly admission of the displayed local-neighborhood carrier*

$$S = (R, I, W, L, H, C, P, N)$$

of Definition 938.1. The admitted orbit row consists of the rule table R, the initial spatial row I, the finite orbit-window ledger W, the local-neighborhood ledger L, the transport row H, the continuation replay C, provenance P, and local name N, and it imports no infinite orbit, hidden evaluator, quotient stream, host cellular equality, or external cellular oracle.

Proof. Project the packet through Definition 938.1. Carrier admission exposes exactly the rule table, initial row, finite orbit window, local-neighborhood ledger, transport, replay, provenance, and local naming rows.

Use Definition 938.2. Once R and I are admitted, each successor entry in the finite window is determined by the corresponding local-neighborhood read in L.

Apply Definition 938.3 and Definition 938.4. They identify the public orbit carrier with the finite route R, I, W, L, H, C, P, N and exclude all unbounded or host-level cellular objects. □

Theorem 938.6 (Cellular orbit substrate local classifier row)**.** *The* CellularOrbitSubstrate$^{\uparrow}$ *classifier is stable under displayed* hsame *transport of cell states and orbit windows. If two accepted packets transport the same rule-table row, initial spatial row, finite orbit-window ledger, and local-neighborhood ledger through their displayed H rows, then the public classifier reads the same local successor decisions through the corresponding continuation replay and naming rows.*

Proof. Begin with the local classifier surface of Definition 938.1. The classifier has access only to R, I, W, L, H, C, P, N.

Apply Definition 938.2. The rule table fixes every successor value for each transported predecessor neighborhood, and L records the displayed entry used for each accepted successor cell.

Use Definition 938.3 and Definition 938.4 to close the route. Since no hidden evaluator or quotient stream is available, hsame-transported cell states and orbit windows preserve the public classifier row. □

Theorem 938.7 (Cellular orbit substrate continuation ledger row)**.** *Every public read of the* CellularOrbitSubstrate$^{\uparrow}$ *cellular orbit route factors through the listed* Cont, Pkg, *and* NameCert *rows of the displayed carrier. The finite continuation replay C reads from R, I, W, L, H, the provenance row P records that finite cellular route, and the local name N exports only that packet to public consumers.*

Proof. Read the continuation route from Definition 938.1. The row C is the only displayed replay path from the rule table and initial row through the orbit window and local-neighborhood ledger.

Apply Definition 938.2 to each accepted successor read. The replay cannot choose a second successor row because L records the unique local-neighborhood entry used with R.

Finally use Definition 938.3 and Definition 938.4. The provenance row P and local name N package the same finite route, so public reads cannot escape to machine code, bounded graph simulation, continuation-tail equality, resource-modulus comparison, or an unbounded orbit object. □

Theorem 938.8 (Cellular orbit substrate rule-locality obligation)**.** *Every accepted orbit step of a* CellularOrbitSubstrate$^{\uparrow}$ *packet is read from the displayed local transition table R together with the neighboring finite rows recorded in the local-neighborhood ledger L. The public route may expose the predecessor neighborhood, the rule-table entry, the successor cell, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows, but it has no coordinate for a hidden cellular evaluator or an external rule oracle.*

Proof. Start from the carrier of Definition 938.1. Its orbit data are exactly the displayed rule table R, initial row I, orbit-window ledger W, and local-neighborhood ledger L.

Apply Definition 938.2. Each accepted successor entry is determined by the local-neighborhood read in L and the corresponding transition entry in R.

Use Definition 938.3 and Definition 938.7 to close the exported route. The continuation, provenance, and local name rows replay only that finite local computation, so no hidden evaluator or external rule oracle can be exported. □

Theorem 938.9 (Cellular orbit substrate orbit-prefix obligation)**.** *Every consumer-visible orbit segment exported by a* CellularOrbitSubstrate$^{\uparrow}$ *packet is a finite prefix of the displayed* BHist *orbit ledger W. The consumer may read the initial row, the accepted finite sequence of successor rows, the local-neighborhood evidence for each displayed step, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows, but it cannot extract an infinite orbit or a detached tail object.*

Proof. Project the packet through Definition 938.1. The only orbit carrier row is the finite ledger W, with the initial row I and local-neighborhood evidence L attached to its displayed steps.

Apply Definition 938.5. Orbit admission is exactly admission of the finite packet $S = (R, I, W, L, H, C, P, N)$, so a public orbit segment must be read as part of the displayed ledger.

Finally use Definition 938.7 and Definition 938.3. The continuation route replays finite ledger rows only, and the non-escape theorem blocks infinite orbit and detached tail coordinates. □

Theorem 938.10 (Cellular orbit substrate classifier-stability obligation)**.** *Classifier replacement for an accepted* CellularOrbitSubstrate$^{\uparrow}$ *packet preserves only the displayed local row, finite orbit prefix, componentwise* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *rows. If the replacement agrees on R, I, W, L, H, C, P, N, then the public classifier has no further surface on which to distinguish the packets.*

Proof. Begin with the classifier inventory in Definition 938.1. The carrier compares accepted packets only through the local rule table, initial row, finite orbit window, local-neighborhood ledger, transport, replay, provenance, and local name.

Use Definition 938.6. Displayed hsame transport of the rule table, initial row, orbit window, and local-neighborhood ledger preserves the public successor decisions.

Apply Definition 938.8 and Definition 938.9. These obligations identify the classifier-visible computation with the local row and finite prefix.

The route closes by Definition 938.7 and Definition 938.3. Since every exported read factors through H, C, P, N together with R, I, W, L, there is no remaining coordinate for a replacement classifier to observe. □

Theorem 938.11 (Cellular orbit substrate obligation closure package)**.** *The carrier row of Definition 938.1, local determinacy of Definition 938.2, observation non-escape of Definition 938.3, NameCert surface of Definition 938.4, orbit carrier row of Definition 938.5, local classifier row of Definition 938.6, continuation ledger row of Definition 938.7, rule-locality obligation of Definition 938.8, orbit-prefix obligation of Definition 938.9, and classifier-stability obligation of Definition 938.10 form the obligation package for* $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ *over the finite packet* $S = (R, I, W, L, H, C, P, N)$*.*

Proof. Project the carrier row first. The packet exposes only the local rule table R, initial row I, finite orbit window W, local-neighborhood ledger L, transport H, replay C, provenance P, and local naming row N.

Apply the local determinacy, observation non-escape, and NameCert surface theorems. They make the finite neighborhood computation carrier-local and keep every public observation inside the displayed certificate rows.

Use the orbit carrier, local classifier, continuation ledger, rule-locality, orbit-prefix, and classifier-stability obligations. These rows exhaust the visible cellular-orbit computation by binding successor decisions, finite prefix reads, and classifier replacement to the same R, I, W, L, H, C, P, N packet.

Therefore the listed rows are exactly the chapter-level obligation package. The package still leaves Rule 110 universality, cellular completeness, infinite-orbit equality, host evaluator correctness, and bridge checking outside the current surface. □

Theorem 938.12 (Cellular orbit substrate mature consumer completeness)**.** *Every cellular-orbit consumer route for an accepted* $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ *packet factors through the finite cellular orbit substrate carrier* $S = (R, I, W, L, H, C, P, N)$*, the orbit row, the trust substrate rows, classifier transport, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The consumer may read the local rule table, initial row, finite orbit-window ledger, local-neighborhood ledger, componentwise transport, continuation replay, provenance, and local name only through that packet and the accepted bridge-facing TasteGate alignment target.*

The mature consumer surface exports no detached global cellular dynamics, host time step, untracked orbit quotient, infinite-orbit equality, host evaluator correctness, or standard computational equivalence.

Proof. Start from Definition 938.1. It exposes the finite packet $S = (R, I, W, L, H, C, P, N)$ as the only carrier through which a cellular-orbit consumer can read local dynamics.

Apply Definition 938.5, Definition 938.6, and Definition 938.7. These rows bind the orbit row, classifier transport, and continuation replay to the same local rule table, initial row, finite orbit window, and local-neighborhood ledger.

Next use Definition 938.8, Definition 938.9, and Definition 938.10. The trust substrate rows and classifier-stability surface keep every successor decision inside R, I, W, L, H, C, P, N.

Finally cite the checked TasteGate alignment target and Definition 938.11. Together they identify the bridge-facing consumer route with the same BHist, hsame, Cont, Pkg, and NameCert rows, leaving no exported coordinate for detached global cellular dynamics, host time steps, untracked orbit quotients, infinite-orbit equality, host evaluator correctness, or standard computational equivalence. □

Closure status for $\mathsf{NameCert}_{\mathsf{CellularOrbitSubstrate}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ packet is generated from a finite local rule table, initial spatial row, finite orbit-window ledger, local-neighborhood ledger, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CellularOrbitSubstrate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 938.1, Definition 938.2, Definition 938.3, Definition 938.4, Definition 938.5, Definition 938.6, Definition 938.7, Definition 938.8, Definition 938.9, Definition 938.10, Definition 938.11, and Definition 938.12 bind the finite local-neighborhood orbit certificate, local classifier, continuation ledger, prefix obligation, classifier-stability surface, bridge-facing alignment, and consumer-complete route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CellularOrbitSubstrate}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.CellularOrbitSubstrateUp.taste gate`

Lean target: `BEDC.Derived.CellularOrbitSubstrateUp.CellularOrbitSubstrateTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, Rule 110 universality, infinite-orbit equality, host evaluator correctness, or standard computational equivalence.

Upgrade path: No further paper-axis upgrade is required; remaining work is verification and bridge checking for the same finite carrier, orbit, trust substrate, classifier, transport, package, and naming rows.

Chapter 939

A Concrete Naming Certificate for $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$

The $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ surface realizes the large-model inscription reading of chapter 9398 as a finite BEDC packet. It records model-weight, activation-orbit, corpus-supply, output-record, audit-refusal, transport, continuation, provenance, and local naming rows without asserting consciousness, host semantic understanding, or internally ratified universal closure.

The packet refines the neighboring inscription surfaces of chapter 610 and chapter 519. Its extra datum is the corpus-supply row: an output record is accepted as carried inscription only after the finite inference substrate and the finite source-channel row are both visible in the same NameCert surface.

Definition 939.1 (LargeModelInscriptionPoint carrier)**.** *A* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, A, C, O, R, H, K, P, N)$$

with the following displayed rows:

1. *W is the finite model-weight row naming the inference substrate;*
2. *A is the finite activation-orbit row generated by a stated prompt and forward-pass schedule;*
3. *C is the corpus-supply row recording the training-source channel from which forward-binding patterns are carried;*
4. *O is the output-record row storing the emitted token sequence as public inscription data;*
5. *R is the audit-refusal row stating that the packet does not internally ratify universal closure, consciousness, host semantic understanding, or model-side proof of the emitted content;*
6. *H records componentwise* hsame *transport for model-weight, activation-orbit, corpus-supply, output-record, refusal, continuation, provenance, and naming rows;*
7. *K records the displayed* Cont *routes by which a consumer reads the inference substrate, corpus-supply row, output record, and refusal ledger in order;*
8. *P is the* Pkg *provenance over* $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{TranscendentalSupplyTaxonomy}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, *and* $\mathsf{ObserverInterface}^{\uparrow}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LargeModelInscriptionPoint}^{\uparrow}}$ *row.*

The carrier contains no subject primitive, no phenomenal-state coordinate, no host-language meaning oracle, and no proof that an emitted universal sentence is internally certified by the model.

Lean4 ✓ `BEDC.Derived.LargeModelInscriptionPointUp.TasteGate.LargeModelInscriptionPointTasteGate single carrier alignment`

Definition 939.2 (LargeModelInscriptionPoint classifier)**.** *Two accepted* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *carriers classify together exactly when their model-weight rows, activation-orbit rows, corpus-supply rows, output-record rows, audit-refusal rows, continuation routes, package provenance, and local naming rows compare by the displayed* hsame *entries. The classifier is componentwise and finite:*

$$W \sim_{\mathsf{hsame}} W', \quad A \sim_{\mathsf{hsame}} A', \quad C \sim_{\mathsf{hsame}} C', \quad O \sim_{\mathsf{hsame}} O',$$
$$R \sim_{\mathsf{hsame}} R', \quad K \sim_{\mathsf{hsame}} K', \quad P \sim_{\mathsf{hsame}} P', \quad N \sim_{\mathsf{hsame}} N'.$$

No clause compares hidden model intentions, unbounded semantic understanding, global corpus truth, or an externally interpreted universal claim.

Theorem 939.3 (LargeModelInscriptionPoint NameCert obligations)**.** *For the carrier of Definition 939.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelInscriptionPoint}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for* $L = (W, A, C, O, R, H, K, P, N)$*;*
2. *classifier transport for the model-weight, activation-orbit, corpus-supply, output-record, audit-refusal, continuation, provenance, and naming rows under componentwise* hsame*;*
3. *stability of the output-record read under the same displayed inference-substrate and corpus-supply rows;*
4. *ledger exactness requiring every inscription read to pass through the corpus-supply row and the audit-refusal row before it reaches the local naming row;*
5. *non-escape for consciousness, host semantic understanding, internally ratified universal closure, hidden source selection, and model-side proof of output truth.*

Proof. Carrier admission is projection from Definition 939.1. The packet lists the finite inference substrate, activation orbit, corpus-supply channel, output record, refusal ledger, transports, continuation routes, package provenance, and local naming row.

Classifier transport is componentwise. Each accepted comparison is one displayed hsame entry, and the continuation row replays the same substrate, corpus, output, and refusal reads after transport.

Stability and ledger exactness follow from the route order. A consumer first reads W and A as the finite inference substrate, then C as the source-channel row, then O as the emitted inscription, and finally R before it reaches N.

The excluded claims have no coordinate in L. Thus the local NameCert can certify carried inscription data, but it cannot project consciousness, host-side semantic understanding, or internally certified universal truth. □

Theorem 939.4 (LargeModelInscriptionPoint carried-output boundary)**.** *Every accepted* LargeModelInscriptionPoi *packet reads model output as carried inscription through the finite route*

model-weight row $\longrightarrow$ *activation-orbit row* $\longrightarrow$ *corpus-supply row* $\longrightarrow$ *output-record row* $\longrightarrow$ $\mathsf{NameCert}_{\mathsf{LargeM}}$

If the corpus-supply row is absent, the packet no longer distinguishes carried inscription from internally ratified universal closure.

Proof. Project the carrier $L = (W, A, C, O, R, H, K, P, N)$. The first two rows give the finite inference substrate, while C records the source channel from which forward-binding patterns are carried.

Read O only after C has been displayed. This forces the output record to be consumed as carried inscription data rather than as a model-internal proof or host semantic oracle.

Finally, the refusal row R fixes the public boundary before the naming row N is read. Without C, the route would expose an output record without the source-channel distinction needed by the theorem. □

Theorem 939.5 (LargeModelInscriptionPoint public non-escape)**.** *No public consumer of an accepted* LargeModelInscriptionPoint$^{\uparrow}$ *packet can extract consciousness, host semantic understanding, model-side proof of a universal claim, an unbounded corpus truth predicate, or a hidden source selector. Public reads are limited to the finite inference-substrate, corpus-supply, output-record, audit-refusal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in the carrier.*

Proof. Inspect the carrier inventory. The only public coordinates are $W, A, C, O, R, H, K, P, N$, so every consumer projection factors through these finite rows.

The neighboring inscription and supply chapters provide route discipline, not extra semantic coordinates. They let the packet name where carried inscription is read, while the local refusal row records what the packet does not contain.

Therefore public consumers can recover the displayed rows and their certified route, but no consciousness predicate, semantic-understanding oracle, universal-truth proof, or hidden source selector is available for projection. □

Theorem 939.6 (LargeModelInscriptionPoint source-channel exactness)**.** *For an accepted* LargeModelInscriptionPo *packet $L = (W, A, C, O, R, H, K, P, N)$, the corpus-supply row C is the unique source-channel gate for carried inscription. A consumer may read O as a carried output only after the route has exposed W, A, C, H, K, P, N, and no hidden source selector, host semantic oracle, or model-internal truth predicate can replace C.*

Lean4 ✓ `BEDC.Derived.LargeModelInscriptionPointUp.LargeModelInscriptionPointCarrier source channel exactness`

Proof. Project the carrier of Definition 939.1. The only source-channel coordinate in the packet is C, while W and A provide the finite inference substrate and O stores the emitted inscription.

Apply the carried-output boundary of Definition 939.4. It reads the output row only after the corpus-supply row has been displayed.

Finally use public non-escape from Definition 939.5. Since hidden source selectors, host semantic oracles, and model-side proof predicates are not projected from L, the row C is the source-channel gate for carried inscription. □

Theorem 939.7 (LargeModelInscriptionPoint output-record stability)**.** *Output-record reads for accepted* LargeModelInscriptionPoint$^{\uparrow}$ *packets are stable under the displayed model-weight, activation-orbit, corpus-supply, audit-refusal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. If two accepted packets compare by the classifier of Definition 939.2, a public consumer that reads the output record of one packet can transport that read only through the componentwise rows*

$$W, A, C, O, R, H, K, P, N.$$

Proof. Use the classifier definition. It compares model-weight, activation-orbit, corpus-supply, output-record, audit-refusal, continuation, provenance, and naming rows component by component.

The NameCert obligations of Definition 939.3 supply classifier transport and route stability for the same displayed rows. Thus an output-record read cannot drop the corpus-supply row, refusal ledger, continuation replay, provenance, or local name during transport.

Apply Definition 939.6 and Definition 939.5. The first keeps the output read behind the source-channel gate, and the second excludes hidden semantic or truth coordinates from the transported public read. □

Theorem 939.8 (LargeModelInscriptionPoint obligation closure package)**.** *The* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *chapter closes the following obligation package for the finite inscription packet: carrier admission for L, classifier transport, source-channel exactness, output-record stability, ledger exactness for the carried-output route, and public non-escape for consciousness, host semantic understanding, universal-truth proof, unbounded corpus truth, and hidden source selection.*

Proof. Carrier admission is Definition 939.1, and classifier transport is Definition 939.2 together with Definition 939.3.

Source-channel exactness follows from Definition 939.6. Output-record stability follows from Definition 939.7.

Ledger exactness is the carried-output boundary of Definition 939.4. Public non-escape is Definition 939.5, so the package has no remaining public coordinate for the excluded claims. □

Theorem 939.9 (LargeModelInscriptionPoint scoped kernel surface)**.** *Every accepted* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *packet exposes exactly the scoped kernel surface*

$$W, A, C, O, R, H, K, P, N.$$

All inscription, carried-output, corpus-supply, audit-refusal, transport, continuation, provenance, and local naming reads factor through those rows. No consciousness predicate, host semantic-understanding oracle, model-side universal proof, corpus truth oracle, or hidden source selector is exported.

Proof. The carrier definition of Definition 939.1 displays precisely the rows $W, A, C, O, R, H, K, P, N$. The classifier definition of Definition 939.2 compares accepted packets only through the same componentwise rows.

The NameCert obligations of Definition 939.3 bind carrier admission, classifier transport, stability, ledger exactness, and non-escape to that finite surface. The carried-output boundary of Definition 939.4 and the source-channel exactness theorem Definition 939.6 force output and corpus-supply reads to pass through the displayed route.

Output-record stability is Definition 939.7, and public non-escape is Definition 939.5. The checked TasteGate marker records that the Lean carrier surface is a single aligned carrier, so the paper packet and the machine carrier name the same scoped surface. □

Falsifiable prediction: If the corpus-supply row is removed, the packet can no longer distinguish model output as carried inscription from internally ratified universal closure.

Independence witness: Nearest siblings: inscriptionpoint, closedobservationsystem, transcendentalsupplytaxonomy, apophaticname, observerinterface; none is bijective because none simultaneously carries model-weight, activation-orbit, corpus-supply, output-record, and no-internal-ratification rows.

The Lean taste-gate surface for this packet is expected to expose Nontrivial LargeModelInscriptionPoint$^{\uparrow}$ and FieldFaithful LargeModelInscriptionPoint$^{\uparrow}$ in TasteGate.lean, with the future namespace BEDC.Derived.LargeMode

Closure status for NameCert$_{\text{LargeModelInscriptionPoint}^{\uparrow}}$ ·

Constructive story: A LargeModelInscriptionPoint$^{\uparrow}$ packet is generated from model-weight, activation-orbit, corpus-supply, output-record, audit-refusal, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LargeModelInscriptionPoint}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 939.1, Definition 939.2, Definition 939.3, Definition 939.4, Definition 939.5, Definition 939.6, Definition 939.7, Definition 939.8, and Definition 939.9 bind the finite carrier, classifier transport, source-channel gate, output-record stability, carried-output ledger, public non-escape obligations, and scoped kernel surface.

Formal verification: FormalStatus(NameCert$_{\text{LargeModelInscriptionPoint}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LargeModelInscriptionPointUp.TasteGate.LargeModelInscriptionPointTasteGate\_single\_car`

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.LargeModelInscriptionPointUp.taste gate`

Not claimed: This chapter does not close consciousness, host semantic understanding, model-side proof of universal claims, corpus truth, hidden source selection, or a checked Lean certificate for the whole packet.

Upgrade path: The remaining formal work is a standard bridge certificate for the same public large-model inscription packet.

Chapter 940

A Trust Manifest Surface for $\mathsf{CellularTrustSubstrate}^{\uparrow}$

The $\mathsf{CellularTrustSubstrate}^{\uparrow}$ surface realizes the cellular trust substrate of Definition 9390.6 as a finite NameCert packet. It is not merely a cellular-rule carrier: it binds the local transition table, the orbit window, the classifier, the manifest row, and the audit route into one closed observation surface. The resulting packet gives later Rule 110 and no-induction arguments a concrete certificate object to cite instead of a narrative-only substrate.

The trust substrate consumes the local rule and finite orbit discipline of chapter 934 and the bounded trace boundary of chapter 944 before its manifest and audit rows are public. Those sibling surfaces supply the displayed cellular evolution and step-indexed trace limit, while this packet records the additional trust manifest and replay route.

The packet consumes the local substrate carrier of chapter 942 and the finite local read boundary of chapter 509. Its trust rows add manifest and audit-route accountability on top of those sibling surfaces, so an accepted read must first expose the cellular carrier and local boundary before the manifest and replay rows can certify the trusted substrate packet.

Falsifiable prediction: Removing the local-rule row permits two orbit windows with different update tables to classify as the same substrate packet; removing the manifest row prevents a finite audit replay from reconstructing the displayed observation boundary.

Independence witness: Nearest siblings: localitycell, continuationautomaton, haltingdistinction, continuationbigstep, and groundcompilereventflowauditroute. None is bijective to the cellular trust substrate because those siblings omit either the fixed local transition table, the orbit-window manifest row, or the closed substrate observation boundary.

Definition 940.1 (Cellular trust substrate carrier)**.** *A* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, K, M, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the local cellular rule row, including the finite alphabet, neighbourhood arity, boundary convention, and complete transition table;*
2. *W is the orbit-window row, consisting of the initial configuration, the finite time interval, and the displayed cell coordinates accepted for observation;*
3. *K is the classifier row assigning finite substrate tags to windows generated by R on W;*

4. *M is the manifest row recording the text-format schema, expected read fields, replay bounds, and admitted observation boundary;*
5. *A is the audit-route row replaying the finite generation of each displayed orbit window and each classifier decision;*
6. *H records componentwise* hsame *transports for R, W, K, M, A, P, N;*
7. *C records displayed* Cont *routes through the local rule, orbit window, classifier, manifest, and audit route;*
8. *P is the* Pkg *provenance over* $\mathsf{CellularTrustSubstrate}^{\uparrow}$*, Definition 9390.6,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only by the displayed rule table, orbit window, substrate classifier, manifest, audit route, transport rows, continuation routes, provenance, and naming row. It has no coordinate for an unbounded orbit, host evaluator identity, hidden update table, detached manifest schema, quotient stream equality, or external induction principle.

Lean4 (definition) `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrateUp`

Theorem 940.2 (Cellular trust substrate obligation surface)**.** *For the carrier of Definition 940.1, the local* $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$ *surface consists exactly of carrier exposure for $S = (R, W, K, M, A, H, C, P, N)$ local-rule accountability through R, orbit-window exposure through W, classifier accountability through K, manifest replay through M, finite audit routing through A, componentwise stability through H, displayed continuation through C, package provenance through P, and local naming through N. No consumer can read a cellular trust substrate, a Rule 110 substrate, a closed observation boundary, or a no-induction substrate premise as* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *evidence without passing through these rows.*

Proof. Carrier exposure is projection from Definition 940.1. The packet displays the rule table, orbit window, classifier, manifest, audit route, transports, continuation routes, provenance, and local naming row as its only coordinates.

Local-rule, orbit-window, classifier, manifest, and audit accountability are read from R, W, K, M, A. The rule row decides the next state locally; the orbit-window row states the finite observation domain; the classifier row names only windows generated inside that domain; the manifest row states the replay schema; and the audit row records the finite route by which each accepted read is recovered.

The remaining rows close the NameCert surface. The H, C, P, N rows transport, replay, record, and name exactly the displayed packet. Since the carrier lacks coordinates for unbounded orbit access, hidden update tables, detached manifests, quotient streams, host evaluator identity, and external induction, the listed rows exhaust the public obligation surface. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrate obligation surface`

Theorem 940.3 (Local-rule nonescape)**.** *Every accepted classifier read exported by a* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier factors through the displayed local rule row R and the finite orbit-window row W. In particular, two packets whose accepted evidence differs only by a hidden update table, a host evaluator, or an unstated boundary convention cannot be classified as the same cellular trust substrate packet.*

Proof. Begin with the accepted-read route. By Definition 940.2, a classifier read reaches the local NameCert row only through $R, W, K, M, A, H, C, P, N$.

The row R supplies the complete local update table and boundary convention, while W supplies the initial configuration and finite observation domain. The classifier row K has no independent transition semantics; it can only tag windows admitted by the route from R through W.

Thus changing the update table, host evaluator, or boundary convention without changing the displayed R row is not a move inside the carrier. Since the packet has no hidden coordinate from which such a change can be recovered, the accepted classifier cannot identify the two displays as the same substrate evidence. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrate local rule window classifier reflect display`

Theorem 940.4 (Orbit observation boundary)**.** *Every closed observation exported by a* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier is bounded by the manifest row M and replayed by the audit-route row A. Consequently, a consumer obtains a finite observation boundary for the displayed orbit window, not an unbounded orbit, a stream quotient, or an internal induction rule over all cellular times.*

Proof. The manifest row M states the accepted text-format schema, replay bounds, and observation boundary. The audit-route row A then replays each displayed orbit-window and classifier read against that schema.

Apply Definition 940.2. A closed observation is accepted as $\mathsf{CellularTrustSubstrate}^{\uparrow}$ evidence only when the route through M and A reaches the local naming row N.

It follows that the exported boundary is finite and displayed. The carrier has no coordinate for an all-time orbit, quotient stream representative, or induction rule over cellular time, so no such object is obtained by consuming the packet. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrate orbit observation boundary`

Theorem 940.5 (Cellular trust substrate obligation closure)**.** *The carrier of Definition 940.1, the local-rule nonescape theorem Definition 940.3, the orbit observation boundary theorem Definition 940.4, the falsifiable prediction, the independence witness, and the Lean TasteGate single-carrier alignment together supply the obligation-closure surface for* $\mathsf{CellularTrustSubstrate}^{\uparrow}$*. The closed surface is the finite route through $R, W, K, M, A, H, C, P, N$, with no claim of Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, or internal induction.*

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrateTasteGate single carrier alignment`

Proof. The carrier displays the complete local-rule, orbit-window, classifier, manifest, audit-route, transport, continuation, provenance, and naming rows. The local-rule nonescape theorem forces classifier reads through R and W, and the orbit observation boundary theorem forces closed observations through M and A.

The falsifiable prediction identifies the two row removals that would break the route, while the independence witness separates this carrier from nearby cellular and continuation packets. The Lean TasteGate alignment supplies the checked single-carrier evidence that the displayed packet has the intended TasteGate carrier shape.

Together these ingredients exhaust the obligation surface named by Definition 940.2. Since the packet still lacks coordinates for universality, host evaluator semantics, quotient streams, cellular completeness, and induction over all times, obligation closure stops at the displayed BEDC rows. □

Theorem 940.6 (Cellular trust substrate primitive scope binding)**.** *For every accepted* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier* $S = (R, W, K, M, A, H, C, P, N)$*, the rows* $R, W, K, M, A, H, C, P, N$ *bind only to the BEDC primitive surface* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. In particular,* R, W, K, M, A *are finite* BHist *rows,* H *is their componentwise* hsame *transport,* C *is their displayed* Cont *route,* P *is their* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$ *row.*

Proof. Start from the obligation surface of Definition 940.2. It lists the complete carrier admission route for $S = (R, W, K, M, A, H, C, P, N)$ and names no primitive outside BHist, hsame, Cont, Pkg, and NameCert.

Local-rule nonescape forces every classifier read through the displayed R and W rows. Orbit observation boundary forces every closed observation through the finite manifest and audit rows M and A.

The checked TasteGate single-carrier alignment cited at Definition 940.5 confirms that the displayed carrier shape is the aligned cellular trust substrate shape. Hence the primitive bindings are exactly the listed BEDC rows, with no host evaluator, quotient stream, or induction principle added by the packet. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate primitive scope binding`

Theorem 940.7 (Cellular trust substrate sibling scope binding)**.** *For every accepted* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *carrier* $S = (R, W, K, M, A, H, C, P, N)$*, the same rows bind the packet to the sibling surfaces* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*,* $\mathsf{CellularSubstrate}^{\uparrow}$*, and* $\mathsf{LocalityCell}^{\uparrow}$*. The rule and orbit rows consume the cellular automaton and bounded trace surfaces, while the classifier, manifest, audit, transport, route, provenance, and naming rows expose only the local substrate and locality-cell boundaries needed by downstream cellular trust consumers.*

Proof. The opening dependency paragraphs place $\mathsf{CellularTrustSubstrate}^{\uparrow}$ after the local rule and finite orbit discipline of $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{BoundedUniversalTrace}^{\uparrow}$, and after the local substrate and local read boundaries of $\mathsf{CellularSubstrate}^{\uparrow}$ and $\mathsf{LocalityCell}^{\uparrow}$.

Apply local-rule nonescape. The accepted classifier route cannot pass around R and W, so the cellular automaton and bounded trace rows are sibling inputs rather than optional explanatory context.

Apply orbit observation boundary. The manifest row M and audit row A bound every closed observation before it is exported, and H, C, P, N keep the same route transported, replayed, packaged, and named.

Finally use the obligation closure theorem together with the checked TasteGate single-carrier alignment cited at Definition 940.5. They identify the accepted sibling scope as the displayed finite route and exclude Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, and all-time induction from the binding. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate sibling scope binding`

The Lean TasteGate namespace BEDC.Derived.CellularTrustSubstrateUp.TasteGate must provide Nontrivial CellularTrustSubstrateUp and FieldFaithful CellularTrustSubstrateUp instances for the displayed carrier, while BEDC.Derived.CellularTrustSubstrateUp.taste_gate records the chapter-level TasteGate instance.

Theorem 940.8 (Cellular trust substrate public export package)**.** *Every public consumer of an accepted* CellularTrustSubstrate$^{\uparrow}$ *packet* $S = (R, W, K, M, A, H, C, P, N)$ *receives exactly the local-rule row, finite orbit-window row, classifier row, manifest row, audit route, componentwise transport, continuation route, provenance, and local* NameCert *row. This public package is routed through* CellularAutomaton$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, CellularSubstrate$^{\uparrow}$, *and* LocalityCell$^{\uparrow}$, *and it carries no Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, or all-time induction.*

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate public export package`

Proof. Start from the obligation surface. It exposes the local rule, orbit window, classifier, manifest, audit route, transport, continuation, provenance, and local naming rows as the complete accepted packet.

Use local-rule nonescape and orbit observation boundary. They force classifier reads through R, W and closed observation reads through M, A, excluding hidden update tables, host evaluators, unbounded orbits, quotient streams, and induction over all cellular times.

Apply primitive scope binding and sibling scope binding. The primitive route is confined to BHist, hsame, Cont, Pkg, and NameCert, while the sibling route passes through CellularAutomaton$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, CellularSubstrate$^{\uparrow}$, and LocalityCell$^{\uparrow}$.

Finally use the obligation closure theorem and its TasteGate alignment marker. They identify the displayed carrier shape as the aligned single-carrier surface, so a public consumer receives exactly the listed packet and no Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, or all-time induction. □

Theorem 940.9 (Cellular trust substrate bridge-scope package)**.** *Any bridge-scope consumer of an accepted* CellularTrustSubstrate$^{\uparrow}$ *packet must preserve the local-rule row, orbit-window row, classifier row, manifest row, audit route, componentwise* hsame *transport, displayed* Cont *route,* Pkg *provenance, local* NameCert *row, falsifiable prediction, independence witness, and TasteGate alignment marker. The bridge-scope package carries no Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, hidden global orbit, or all-time induction principle.*

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate bridge scope package`

Proof. Start from the public export theorem. It gives every public consumer exactly the local rule, orbit window, classifier, manifest, audit route, transport, continuation, provenance, and local naming rows.

Apply primitive scope binding. The public rows are bound to BHist, hsame, Cont, Pkg, and NameCert, so the bridge-scope consumer cannot add a host evaluator, quotient stream representative, or induction principle.

Next apply sibling scope binding. It keeps the route attached to CellularAutomaton$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, CellularSubstrate$^{\uparrow}$, and LocalityCell$^{\uparrow}$, which supplies the local-rule, finite-orbit, classifier, manifest, audit, prediction, and independence surface used by the bridge.

Finally use obligation closure and the TasteGate alignment marker. They identify the displayed carrier as the accepted cellular trust substrate packet, so the bridge-scope consumer preserves the stated rows without importing Rule 110 universality, cellular completeness, or hidden all-time behavior. □

Theorem 940.10 (Cellular trust substrate mature route completeness)**.** *For every accepted* CellularTrustSubstrate$^{\uparrow}$ *packet* $S = (R, W, K, M, A, H, C, P, N)$*, the local-rule row, orbit-window row, classifier row, manifest row, audit route, primitive scope, sibling scope, public export package, and bridge-scope package*

exhaust all mature public reads of the packet. A mature consumer can only preserve the finite route through $R, W, K, M, A, H, C, P, N$, together with the TasteGate single-carrier alignment marker; it receives no Rule 110 universality, host evaluator correctness, quotient stream equality, cellular completeness, hidden global orbit, or all-time induction principle.

Proof. Start from the obligation surface and the obligation closure theorem. They display exactly the carrier rows $R, W, K, M, A, H, C, P, N$ and identify the TasteGate single-carrier alignment as the checked carrier-shape evidence for the packet.

The route constraints are then supplied in layers. Local-rule nonescape forces classifier reads through R and W, while orbit observation boundary forces closed observations through the manifest row M and audit route A.

Next apply primitive scope binding and sibling scope binding. These results bind the same route to BHist, hsame, Cont, Pkg, NameCert, and to the sibling surfaces CellularAutomaton$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, CellularSubstrate$^{\uparrow}$, and LocalityCell$^{\uparrow}$, without adding a host evaluator or an all-time cellular principle.

Finally use the public export package and bridge-scope package. The public package exposes exactly the listed rows, and the bridge-scope package preserves the same rows across bridge-facing reads. Therefore the mature public surface has no remaining paper-side coordinate beyond the displayed finite route and the TasteGate alignment marker. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate mature route completeness`

Corollary 940.11 (Cellular trust substrate strict obstruction locality)**.** *Let $S = (R, W, K, M, A, H, C, P, N)$ be an accepted* CellularTrustSubstrate$^{\uparrow}$ *carrier. Any consumer-facing obstruction cited from a displayed local-rule failure, manifest refusal, sibling separation, or countermodel-style row in this chapter is strict and local: it can block only the named converse or reuse route that tries to read around the displayed cellular trust substrate packet. It cannot be reinterpreted as a global impossibility theorem for cellular substrates, a host theorem about Rule 110 or bounded universal traces, or hidden external validation of a non-displayed update table, evaluator, manifest schema, quotient stream, unbounded orbit, or all-time induction principle.*

Proof. Start with the complete consumer surface. By Definition 940.2, a consumer can read CellularTrustSubstrate$^{\uparrow}$ evidence only through the finite row inventory

$$R, W, K, M, A, H, C, P, N.$$

The same theorem names these rows as carrier exposure, local-rule accountability, orbit-window exposure, classifier accountability, manifest replay, audit routing, componentwise transport, displayed continuation, package provenance, and local naming. Thus a displayed failure, refusal, separation, or countermodel-style row has no independent public force unless its use factors through this inventory.

Now apply Definition 940.3. Every accepted classifier read factors through the displayed local rule row R and the finite orbit-window row W. Hence a purported converse from the classifier row K to a hidden update table, host evaluator, or unstated boundary convention is blocked only at that classifier route: the packet gives K no transition semantics except the route from R through W.

The remaining reuse attempts fail for the same row-inventory reason. Manifest and audit uses must pass through M and A; transports and continuations must remain componentwise through H and C; provenance and naming must remain the displayed rows P and N. None of these

coordinates is a global cellular impossibility theorem, a host evaluator theorem, a quotient-stream equality, an external validation channel, or an induction principle over all cellular times.

Therefore the obstruction is strict in the BEDC sense. It rejects exactly the attempted converse or reuse route whose required coordinate is absent from $R, W, K, M, A, H, C, P, N$, while leaving every stronger global, host-level, or externally validated reading outside the accepted $\mathsf{CellularTrustSubstrate}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CellularTrustSubstrateUp.CellularTrustSubstrate strict obstruction locality`

Theorem 940.12 (Cellular trust substrate standard bridge boundary)**.** *For every accepted* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *packet* $S = (R, W, K, M, A, H, C, P, N)$*, any bridge-facing read of the packet is confined to the displayed local-rule row* R*, finite orbit-window row* W*, classifier row* K*, manifest row* M*, audit route* A*, componentwise transport* H*, continuation replay* C*, package provenance* P*, and local naming row* N*. The bridge may preserve and expose only these rows together with the primitive and sibling scope bindings already proved in this chapter. It cannot import Rule 110 universality, host evaluator correctness, quotient streams, all-time induction, hidden global-orbit data, or external validation.*

Proof. Start from the public export package and the bridge-scope package. They give every public or bridge-scope consumer exactly the finite route through $R, W, K, M, A, H, C, P, N$.

Use local-rule nonescape and orbit observation boundary. These force classifier reads through R, W, K and closed observation reads through M, A, so a bridge-facing consumer cannot replace the displayed rows by a host evaluator, hidden transition table, quotient stream, global orbit, or all-time induction route.

Finally apply primitive scope binding, sibling scope binding, and strict obstruction locality. They keep the accepted bridge surface over BHist, hsame, Cont, Pkg, NameCert, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, and $\mathsf{LocalityCell}^{\uparrow}$, while any attempted read outside those coordinates is only a local obstruction to that attempted reuse. □

Closure status for $\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CellularTrustSubstrate}^{\uparrow}$ packet is generated from the displayed local rule, finite orbit window, classifier, manifest, audit route, componentwise transport, continuation route, package provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$, maturePackageClosed)

Scope closed: The displayed carrier rows, obligation surface, local-rule nonescape, manifest and audit rows, sibling dependencies $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, and $\mathsf{LocalityCell}^{\uparrow}$, and the standard bridge boundary bind the finite $R, W, K, M, A, H, C, P, N$ cellular trust scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CellularTrustSubstrate}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CellularTrustSubstrateUp.TasteGate.CellularTrustSubstrateTasteGate\_single\_carrier\_al`

Bridge status: paperBridge

Not claimed: This chapter does not claim Rule 110 universality, host evaluator correctness, quotient streams, cellular completeness, or all-time induction.

Upgrade path: Mature closure requires checked examples and completeness for the same public carrier rows, local-rule route, orbit boundary, manifest, audit route, transport, replay, provenance, and naming data.

Chapter 941

A Concrete Naming Certificate for $\mathsf{CellularRowClassifier}^{\uparrow}$

The $\mathsf{CellularRowClassifier}^{\uparrow}$ packet realizes the finite local row-window classifier surface implicit in chapter 9390. It is adjacent to the transition-row carrier of chapter 482, the diagonal trace boundary of chapter 491, and the obstruction-ledger consumer of chapter 490. Its distinguishing datum is the displayed local neighbourhood window through which a row tag is read; the packet therefore certifies finite cellular pattern recognition without importing a host orbit classifier.

Falsifiable prediction: If a CellularRowClassifier packet is atomic, every accepted row-tag read factors through the displayed local window and no consumer can recover a host orbit classifier outside that window.

Independence witness: ContinuationAutomatonUp supplies transition rows but not local neighbourhood tags; HaltingDistinctionUp supplies diagonal trace rows but not cellular pattern tags; HaltingObstructionUp supplies obstruction ledgers but not row-window classifiers; ApophaticNameUp supplies boundary requests but not transition-local tag determinacy.

Definition 941.1 (CellularRowClassifier carrier)**.** *A* $\mathsf{CellularRowClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, J, T, A, H, C, P, N)$$

with the following visible rows:

1. *R is the finite cellular rule row whose local neighbourhood relation is being inspected.*
2. *J is the displayed row-window row, naming the finite centre cell, its neighbourhood, and the bounded row segment from which a tag may be read.*
3. *T is the finite tag row assigning local tags such as ether, glider phase, collision fragment, boundary refusal, or unclassified window to the displayed windows in J.*
4. *A is the acceptance ledger recording which row-tag reads are admitted and which candidate reads are refused for lacking the displayed neighbourhood.*
5. *H records componentwise* hsame *transports for R, J, T, A, P, N.*
6. *C records displayed* Cont *routes through the rule row, local window row, tag row, acceptance ledger, provenance row, and local name row.*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, *cellular row-window reads, the sibling packets cited above, and local* NameCert *evidence.*

8. N is the local $\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$ *row.*
The carrier has no coordinate for an unbounded cellular orbit, a host evaluator over all rows, quotient stream equality, a global glider catalog, or a detached Rule 110 oracle.

Theorem 941.2 (CellularRowClassifier NameCert obligations)**.** *For the carrier of Definition 941.1, the local* $\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for $W = (R, J, T, A, H, C, P, N)$, rule-row accountability through R, local window exposure through J, tag-read accountability through T, acceptance and refusal replay through A, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. A consumer may use the packet as finite cellular pattern evidence only through those rows.*

Proof. Carrier admission is projection from Definition 941.1. The rule row, local window row, tag row, acceptance ledger, transports, routes, provenance, and local naming row are the only visible coordinates.

Rule-row and window accountability come from R and J. A row-tag read is admitted only when the displayed window supplies the neighbourhood data named by the carrier, so the consumer cannot replace the local row by an unrecorded orbit segment.

Tag-read and refusal replay are read from T and A. The ledger records both admitted local tags and refusals caused by missing neighbourhood data, which keeps the classifier inside the finite row-window surface.

Finally use H, C, P, N. These rows transport, replay, record, and name the same displayed packet. Since no coordinate names an unbounded orbit, a host evaluator, quotient stream equality, a global catalog, or a detached oracle, the listed rows exhaust the local NameCert obligations. □

Theorem 941.3 (CellularRowClassifier local window determinacy)**.** *Every accepted row-tag read exported by a* $\mathsf{CellularRowClassifier}^{\uparrow}$ *packet factors through the finite route*

$$R \longmapsto J \longmapsto T \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Thus the read is determined by the displayed local neighbourhood window and its ledger entry. The packet exports no host orbit classifier, no unbounded cellular history, no global halting classifier, no quotient stream equality, and no ambient completeness principle.

Proof. Start with an accepted row-tag read. Carrier projection gives R, J, T, A, H, C, P, N as the only rows through which the read can pass.

Apply Definition 941.2. The tag row T can be read only against the local window J, and the acceptance ledger A records whether that window supplies the required neighbourhood data.

The remaining route is bookkeeping inside the packet. The hsame transports and Cont routes preserve the displayed rows, while P and N record package provenance and local naming for the same finite read.

Coordinate exhaustion gives determinacy. Since no coordinate names a host orbit classifier, unbounded cellular history, global halting classifier, quotient stream equality, or ambient completeness principle, the accepted tag is determined exactly by the displayed local window. □

Theorem 941.4 (CellularRowClassifier standard reflection bridge surface)**.** *Any standard reflection read of a* $\mathsf{CellularRowClassifier}^{\uparrow}$ *row classifier must factor through the displayed finite route*

$$R \longmapsto J \longmapsto T \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The bridge-facing surface is exactly the carrier of Definition 941.1, the local $\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$ *obligations of Definition 941.2, the local window determinacy of Definition 941.3, and the checked*

TasteGate row for the displayed carrier. The surface exports no host orbit classifier, unbounded cellular history, global halting classifier, quotient stream equality, or ambient completeness principle.

Proof. Start with a standard reflection read of the row classifier. Carrier projection exposes only R, J, T, A, H, C, P, N, so the read has no coordinate from which to inspect a host orbit, an unbounded history, or a detached classifier.

Apply Definition 941.2. The $\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$ surface requires rule-row accountability, local window exposure, tag-read accountability, acceptance and refusal replay, componentwise stability, displayed route replay, package provenance, and local naming through the same finite packet.

Next use Definition 941.3. Any accepted row-tag read is determined by the local neighbourhood row J and its ledger entry in A, with T recording the tag and H, C, P, N preserving the same displayed route for reflection.

The TasteGate row supplies the public nontriviality and field-faithfulness surface for this carrier, but it does not add another classifier coordinate. Therefore the standard reflection read must pass through the displayed finite route R, J, T, A, H, C, P, N.

Since the carrier and cited obligations contain no host orbit classifier, unbounded cellular history, global halting classifier, quotient stream equality, or ambient completeness principle, the bridge-facing surface cannot export any such datum. □

The Lean TasteGate namespace BEDC.Derived.CellularRowClassifierUp.TasteGate provides Nontrivial CellularRow and FieldFaithful $\mathsf{CellularRowClassifier}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularRowClassifier}^{\uparrow}$ packet is generated from a finite cellular rule row, local row-window row, tag row, acceptance ledger, componentwise hsame transports, displayed continuation routes, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 941.1, Definition 941.2, Definition 941.3, Definition 941.4, and

Lean4 ✓ `BEDC.Derived.CellularRowClassifierUp.taste gate`

specify the finite row-window classifier surface, bridge-facing reflection surface, and checked TasteGate instance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CellularRowClassifier}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CellularRowClassifierUp.CellularRowClassifierTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove Rule 110 universality, global cellular orbit classification, total halting classification, quotient stream equality, host evaluator correctness, or bridge checking.

Upgrade path: The paper closure axis is mature; bridgeChecked verification remains a separate formal bridge obligation.

Chapter 942

A Concrete Naming Certificate for $\mathsf{CellularSubstrate}^{\uparrow}$

The $\mathsf{CellularSubstrate}^{\uparrow}$ packet packages the carrier alternative of chapter 9391 as a finite BHist NameCert surface. It sits between the rule-table and finite-orbit emphasis of chapter 934 and the audit-ledger substrate of chapter 937. Its distinguishing role is to certify the cellular carrier itself: a displayed local row, neighbourhood window, local update route, global-step ledger, transport surface, continuation surface, provenance, and local NameCert row are all present at once.

Falsifiable prediction: A packet with the same global-step ledger but no displayed neighbourhood window is not an accepted $\mathsf{CellularSubstrate}^{\uparrow}$ carrier.

Independence witness: Nearest siblings: cellularautomaton, cellulartrustsubstrate, streamname, interhistlocalityledger, and finitewindowenvelope; the carrier is not bijective to any of them because it simultaneously requires a local neighbourhood window, local update route, global-step ledger, transport rows, and local NameCert row, while each sibling omits at least one of those displayed coordinates.

Definition 942.1 (Cellular substrate carrier)**.** *A* $\mathsf{CellularSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, N, U, G, H, C, P, L)$$

with the following displayed rows:

1. *S is the cell-state row, naming the finite row segment visible to the carrier.*
2. *N is the neighbourhood-window row, naming the finite cells read around each displayed cell.*
3. *U is the local update route, recording the stated local transition from a neighbourhood window to the next cell-state row.*
4. *G is the global-step ledger, listing the finite sequence of substrate steps admitted by the carrier.*
5. *H records componentwise* hsame *transports for the cell-state, neighbourhood, local-update, global-ledger, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *routes through the cell-state row, neighbourhood row, local update route, global-step ledger, and local NameCert row.*
7. *P is the* Pkg *provenance over* $\mathsf{CellularSubstrate}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, the cellular-carrier vision, and the sibling cellular NameCert chapters.*

8. *L is the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrate}^{\uparrow}}$ *row for carrier admission, classifier accountability, stability, exactness, ledger replay, and non-escape.*

The local classifier compares accepted carriers only by these displayed rows. It has no coordinate for host cellular automaton equality, an infinite grid, a hidden scheduler, a detached global evaluator, or an unbounded update oracle.

Lean4 ✓ `BEDC.Derived.CellularSubstrateUp.CellularSubstrateTasteGate single carrier alignment`

Theorem 942.2 (Cellular substrate NameCert obligations)**.** *For the carrier of Definition 942.1, the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrate}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier exposure for $Q = (S, N, U, G, H, C, P, L)$;*
2. *classifier accountability for reads that compare cell-state rows, neighbourhood windows, and local update routes only through S, N, U;*
3. *stability under the componentwise* hsame *transports displayed in H;*
4. *exactness of local update and global-step replay through U, G, C;*
5. *ledger and provenance closure through G, P, L, with no export of hidden host equality, infinite-grid reasoning, scheduler state, or unbounded update oracle.*

Every accepted $\mathsf{CellularSubstrate}^{\uparrow}$ *consumer must pass through these five rows before citing a cellular carrier read.*

Proof. Carrier exposure is projection from Definition 942.1. The tuple displays the cell-state row, neighbourhood window, local update route, global-step ledger, transports, continuations, provenance, and local naming row as its only coordinates.

Classifier accountability is local to S, N, U. A cell-state comparison or update read is accepted only when the read names the displayed neighbourhood window and the displayed local transition route.

Stability is supplied by H, while exactness is supplied by U, G, C. The transport row compares only the visible coordinates, and the continuation row replays only the finite local update route and finite global-step ledger.

Finally P and L package the same finite route as the public naming surface. Since the carrier lacks coordinates for host equality, an infinite grid, a hidden scheduler, and an unbounded update oracle, the five obligation rows exhaust the NameCert surface. □

Theorem 942.3 (Cellular substrate local-update non-escape)**.** *Every accepted local-update read exported by a* $\mathsf{CellularSubstrate}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto N \longmapsto U \longmapsto G \longmapsto H \longmapsto C \longmapsto P \longmapsto L.$$

In particular, an accepted packet exposes no host cellular automaton equality, infinite grid, hidden scheduler, or unbounded update oracle.

Proof. Start with an accepted local-update read. By Definition 942.1, the only available rows are S, N, U, G, H, C, P, L.

Apply Definition 942.2. The read must use the displayed neighbourhood window, the displayed local update route, and the finite global-step ledger before it can be named by the local $\mathsf{NameCert}_{\mathsf{CellularSubstrate}^{\uparrow}}$ row.

The route has no coordinate for host cellular automaton equality, infinite grid data, hidden scheduler state, or an unbounded update oracle. Thus every accepted read remains a finite cellular-substrate carrier read. □

Theorem 942.4 (Cellular substrate window exposure obligation)**.** *Every accepted* $\mathsf{CellularSubstrate}^{\uparrow}$ *carrier exposes the neighbourhood-window row* N *before any accepted local update route* U *can be replayed. Equivalently, a local-update read of*

$$Q = (S, N, U, G, H, C, P, L)$$

must pass through the displayed BHist *window row, the local* hsame *transport row, the displayed* Cont *route, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrate}^{\uparrow}}$ *row before it cites* U*.*

Proof. Project the carrier of Definition 942.1. The tuple displays N as a separate neighbourhood-window row and U as the local update route, so the update route has no independent host-side source.

Apply Definition 942.2. Its carrier-exposure and classifier-accountability rows require accepted reads to name the displayed neighbourhood window before they name the displayed transition route.

Finally use Definition 942.3. The only replay route is the finite path through S, N, U, G, H, C, P, L, hence a local update cannot bypass N or acquire an infinite-grid, scheduler, or host evaluator coordinate. □

Theorem 942.5 (Cellular substrate classifier stability obligation)**.** *Componentwise* hsame *transport in the row* H *preserves the* $\mathsf{CellularSubstrate}^{\uparrow}$ *classifier on the cell-state row* S*, neighbourhood-window row* N*, local update route* U*, global-step ledger* G*, continuation row* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{CellularSubstrate}^{\uparrow}}$ *row* L*. No transported read can replace these rows by host cellular automaton equality, an infinite grid, a hidden scheduler, or an unbounded update oracle.*

Proof. Project $Q = (S, N, U, G, H, C, P, L)$ through Definition 942.1. The transport coordinate H is declared componentwise for exactly the displayed cell-state, neighbourhood, update, ledger, continuation, provenance, and naming rows.

Use Definition 942.2. Its stability row is local to those same displayed coordinates and therefore transports the classifier without creating another equality surface.

The non-escape theorem Definition 942.3 rules out every hidden host evaluator, infinite grid, scheduler, and unbounded update oracle after transport. Thus classifier stability remains a $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ statement about the packet rows. □

Theorem 942.6 (Cellular substrate ledger replay obligation)**.** *Audit-route replay through the global-step ledger* G*, continuation row* C*, provenance row* P*, and local naming row* L *exhausts all accepted* $\mathsf{CellularSubstrate}^{\uparrow}$ *reads. The replay exports only the finite cellular-substrate packet and never exports a hidden host evaluator, infinite grid, scheduler, or unbounded update oracle.*

Proof. Start with an accepted cellular-substrate read and project the carrier by Definition 942.1. The only ledgered route available after the local update row is G, C, P, L.

Apply Definition 942.2. Its exactness and ledger-provenance rows state that accepted reads replay the finite global-step ledger through the displayed continuation and package rows before entering the local NameCert row.

Now cite Definition 942.3. Since the complete route already factors through S, N, U, G, H, C, P, L, the audit replay has no coordinate from which to export a hidden evaluator, infinite grid, scheduler, or unbounded oracle. □

Theorem 942.7 (Cellular substrate local-rule factorization)**.** *Every accepted* $\mathsf{CellularSubstrate}^{\uparrow}$ *local-update read of a carrier*

$$Q = (S, N, U, G, H, C, P, L)$$

factors through the sibling CellularLocalUpdateRule$^{\uparrow}$ *surface: the displayed neighbourhood window* N*, the selected local rule row* U*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The read cannot be reconstructed from the global-step ledger alone, a hidden scheduler, an infinite grid, or a host evaluator.*

Proof. Project the carrier through Definition 942.1. The only local-update source available to the packet is the displayed pair of the neighbourhood-window row N and the local update route U, together with the visible transport, continuation, package, and naming rows.

Apply the window exposure theorem Definition 942.4 and the ledger replay theorem Definition 942.6. The first forces the local-update read to see N before U, and the second confines any later replay to G, C, P, L rather than to an external evaluator.

Now compare the exposed rows with the sibling packet of chapter 984. Its obligation theorem Definition 984.2 identifies the local-rule consumer surface as a finite neighbourhood window, selected local rule row, componentwise hsame, displayed Cont, Pkg, and local NameCert route.

Thus the CellularSubstrate$^{\uparrow}$ local-update row is consumed through that sibling surface. Since the carrier has no coordinate for a hidden scheduler, infinite grid, host evaluator, or ledger-only rule source, no accepted read factors by any other route. □

The Lean file `Derived/CellularSubstrateUp/TasteGate.lean` includes `Nontrivial CellularSubstrateUp` and `FieldFaithful CellularSubstrateUp` instances for the displayed carrier.

Closure status for NameCert$_{\text{CellularSubstrate}^{\uparrow}}$ **.**

Constructive story: A CellularSubstrate$^{\uparrow}$ packet is generated from a finite cell-state row, a neighbourhood-window row, a local update route, a global-step ledger, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CellularSubstrate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 942.1, Definition 942.2, Definition 942.3, Definition 942.4, Definition 942.5, Definition 942.6, and Definition 942.7 bind the carrier, NameCert obligations, local-update non-escape, window exposure, classifier stability, ledger replay, and local-rule factorization rows.

Formal verification: FormalStatus(NameCert$_{\text{CellularSubstrate}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CellularSubstrateUp.CellularSubstrateTasteGate\_single\_carrier\_alignment`
Lean4 ✓ `BEDC.Derived.CellularSubstrateUp.taste gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close bridge checking, universality, hidden host evaluator correctness, quotient stream equality, internal induction, infinite-grid completeness, scheduler correctness, or unbounded update evaluation for CellularSubstrate$^{\uparrow}$.

Upgrade path: No further paper-side closure grade remains; bridgeChecked is verification-axis work.

Chapter 943

A Concrete Naming Certificate for $\mathsf{CellularCarrierEncoding}^{\uparrow}$

The $\mathsf{CellularCarrierEncoding}^{\uparrow}$ packet realizes the cellular-carrier alternative of chapter 9391 as a finite NameCert surface. It sits beside the cellular-rule certificate of chapter 934 and the closed Rule 110 trust substrate of chapter 937: those siblings certify cellular dynamics and trust-path replay, while this carrier certifies the encoding bridge from a BHist value to a displayed cell row.

Falsifiable prediction: A packet that omits the cell-row embedding coordinate cannot satisfy the CellularCarrierEncodingUp non-escape theorem.

Independence witness: Nearest siblings: cellularautomaton, cellulartrustsubstrate, groundcompilereventflowaudit, twin-substrate-audit. The carrier is not bijective to any sibling because it includes both BHist-to-bit encoding and cellular row embedding obligations, while the siblings expose only substrate dynamics, trust, audit flow, or two-substrate alignment rows.

Definition 943.1 (Cellular carrier encoding carrier)**.** *A* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, T, R, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* BHist *row whose constructor trace is being encoded.*
2. *T is the* toBits *and readback row, recording the finite binary word obtained from B and the inverse read that reconstructs the same* BHist *constructor trace.*
3. *R is the cell-row embedding coordinate, naming the finite width, default fill, start coordinate, and row slice that contains the word from T.*
4. *S is the cellular substrate witness row, citing the local cellular rule, admissible initial row, and finite observation window through which the embedded row is read.*
5. *H records componentwise* hsame *transports for B, T, R, S, P, N.*
6. *C records displayed* Cont *routes through the source* BHist *row, the bit encoding, the cell-row embedding, the substrate witness, the provenance row, and the local certificate.*
7. *P is the* Pkg *provenance over* $\mathsf{CellularCarrierEncoding}^{\uparrow}$, BHist, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularCarrierEncoding}^{\uparrow}}$ *row.*

The classifier compares carriers only by the displayed source BHist *row, bit encoding and readback row, embedding coordinate, substrate witness, transports, continuation routes, package provenance, and local naming row. It has no coordinate for a host byte string, an unstated row offset, an unbounded cellular orbit, a detached substrate evaluator, or a quotient of encodings.*

Lean4 ✓ `BEDC.Derived.CellularCarrierEncodingUp.CellularCarrierEncodingTasteGate single carrier alignment`

Theorem 943.2 (CellularCarrierEncoding NameCert obligations)**.** *For the carrier of Definition 943.1, the local* $\mathsf{NameCert}_{\mathsf{CellularCarrierEncoding}^{\uparrow}}$ *surface consists exactly of carrier exposure for* $E = (B, T, R, S, H, C, P, N)$*, source accountability through* B*, bit encoding and readback exactness through* T*, cell-row embedding accountability through* R*, cellular substrate witnessing through* S*, componentwise stability through* H*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*. No consumer can read a BHist-to-cell-row carrier equivalence, a cellular substrate row, or a finite encoding bridge as* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *evidence without passing through these rows.*

Proof. Carrier exposure is projection from Definition 943.1. The packet displays the source BHist row, the bit encoding and readback row, the cell-row embedding coordinate, the substrate witness, transports, continuation routes, provenance, and local naming row as its only coordinates.

Source accountability and bit-readback exactness are read from B and T. The row T is tied to the constructor trace of B, so a consumer cannot replace the displayed source by a host byte string or a different binary word after the packet is accepted.

Embedding accountability and substrate witnessing are read from R and S. The embedding coordinate states where the finite word is placed in the cell row, and the substrate witness states which local cellular surface reads that row.

The remaining rows close the NameCert surface. The H, C, P, N rows transport, replay, record, and name exactly the displayed packet. Since the carrier lacks coordinates for unstated row offsets, unbounded orbits, detached evaluators, and quotient encodings, the listed rows exhaust the public obligation surface. □

Theorem 943.3 (Cell-row embedding non-escape)**.** *Every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *read factors through the displayed cell-row embedding coordinate* R*. In particular, a packet may certify the finite route*

$$B \longmapsto T \longmapsto R \longmapsto S \longmapsto H, C, P, N,$$

but it cannot identify two encodings that differ only by an omitted row coordinate, hidden default fill, host-side byte layout, detached cellular evaluator, or unbounded orbit read.

Proof. Start with an accepted encoding read. By Definition 943.2, the read reaches the local NameCert row only through B, T, R, S, H, C, P, N.

The row T supplies the finite bit word and readback to B, but it does not by itself place the word inside the cellular substrate. The coordinate R supplies that placement by naming the finite width, default fill, start coordinate, and row slice.

The substrate witness S then reads only the row supplied by R. Since the carrier has no hidden coordinate for another offset, byte layout, evaluator, or infinite orbit, omitting R removes the route by which the cellular read can be tied back to the source BHist packet. □

Theorem 943.4 (CellularCarrierEncoding readback exactness)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier* $E = (B, T, R, S, H, C, P, N)$*, the row* T *binds the source* BHist *row* B *to both the finite binary word and the reconstruction row before the embedding coordinate* R *is read. Hence a consumer*

cannot read R as evidence for a cellular carrier unless the displayed word in T is already tied to the same constructor trace of B.

Proof. Project the carrier of Definition 943.1. The row B is the source BHist trace, and T records the finite binary word together with the inverse readback of that same trace.

Apply the NameCert obligations of Definition 943.2. They make bit encoding and readback exactness through T a local certificate duty before cell-row embedding accountability through R can be consumed.

Finally use cell-row embedding non-escape, Definition 943.3. Since every accepted cellular read factors through R only after the route $B \longmapsto T \longmapsto R$ is displayed, R cannot be used as a detached cell-row witness for a different host word or source trace. □

Theorem 943.5 (CellularCarrierEncoding route stability)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier $E = (B, T, R, S, H, C, P, N)$, the rows H and C transport and replay the displayed route*

$$B \longmapsto T \longmapsto R \longmapsto S$$

without adding a hidden row offset, host byte layout, detached substrate evaluator, or unbounded cellular orbit. The stable route is exactly the source row, bit-readback row, embedding coordinate, and substrate witness row named by the carrier.

Proof. Read the structural rows in Definition 943.1. The row H gives componentwise hsame transports for the displayed packet, and C gives displayed Cont routes through the source, bit encoding, embedding, substrate, provenance, and naming rows.

Use Definition 943.2 to identify these rows as the componentwise stability and route-replay duties of the local certificate. They preserve the route already displayed by B, T, R, S rather than adding an independent route.

The non-escape theorem then removes the unavailable coordinates. Since hidden offsets, host byte layout, detached evaluators, and unbounded orbit reads are not rows of E, transport by H and replay by C cannot introduce them. □

Theorem 943.6 (CellularCarrierEncoding provenance exhaustion)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier $E = (B, T, R, S, H, C, P, N)$, the rows P and N exhaust the public provenance and local naming surface over* BHist*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularTrustSubstrate}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert*. No additional provenance row is available for a host byte encoding, hidden cellular evaluator, quotient of encodings, or unbounded substrate read.*

Proof. Project the provenance and naming rows from Definition 943.1. The row P records the listed BEDC packets and kernel relations, while N is the local $\mathsf{NameCert}_{\mathsf{CellularCarrierEncoding}^{\uparrow}}$ row.

Apply the local obligation theorem. It says that package provenance through P and local naming through N are part of the exact NameCert surface for the same packet $E = (B, T, R, S, H, C, P, N)$.

Finally combine readback exactness with route stability. The public route already ties B to T, R, and S through H and C, so P, N can only certify that displayed route and cannot supply an extra provenance channel for any excluded host-side or unbounded cellular object. □

Theorem 943.7 (CellularCarrierEncoding obligation closure)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier $E = (B, T, R, S, H, C, P, N)$, the carrier exposure of Definition 943.1, the* NameCert *obligation surface of Definition 943.2, cell-row embedding non-escape, readback exactness, route stability, and provenance exhaustion supply the complete paper obligation surface for the finite BHist-to-cell-row carrier. A consumer of the packet may use only the displayed* BHist *row,* hsame *transports,* Cont *routes,* Pkg *provenance, local* NameCert *row, and sibling cellular rows named by the carrier.*

Proof. Project the accepted carrier. Its visible data are exactly the rows B, T, R, S, H, C, P, N, so every public read begins from the same finite packet and no host-side coordinate is available at the carrier boundary.

Apply the local NameCert obligation theorem to expose the duties attached to those rows. The source row, readback row, embedding coordinate, substrate witness, transports, routes, provenance, and local naming row are all certificate duties of the same packet.

Use the four later row theorems to close the surface. Non-escape forces cellular reads through R, readback exactness ties T to B before R is consumed, route stability keeps the replay inside B, T, R, S, H, C, and provenance exhaustion keeps P, N from adding any extra channel.

Therefore the listed rows supply the whole paper obligation surface: carrier, NameCert, non-escape, readback, route, and provenance are consumed together, and the packet exposes no further primitive or evaluator. □

Theorem 943.8 (CellularCarrierEncoding primitive scope binding)**.** *The* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *paper surface binds only the primitives* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*, and* $\mathsf{CellularTrustSubstrate}^{\uparrow}$*. In particular, the obligation closure of Definition 943.7 does not quantify over a host byte carrier, a global cellular evaluator, an orbit quotient, an unstated row coordinate, or any primitive outside those displayed rows.*

Proof. Read the carrier definition. The source row is a BHist row; stability and replay are expressed by hsame and Cont; provenance and local naming are expressed by Pkg and $\mathsf{NameCert}$.

The only cellular dependencies named by the packet are the sibling $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{CellularTrustSubstrate}^{\uparrow}$ rows recorded in the provenance and substrate witness. No separate carrier coordinate names a byte-level encoding, global evaluator, quotient, or hidden row coordinate.

Apply obligation closure. Since the closed paper surface is exhausted by the displayed rows and the listed row theorems, every primitive used by the chapter is one of the named BEDC objects, and no outside primitive is introduced by the closure claim. □

Theorem 943.9 (CellularCarrierEncoding sibling scope binding)**.** *The* $\mathsf{CellularAutomaton}^{\uparrow}$ *and* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *sibling rows in a* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *carrier are consumed as visible dependencies of the finite embedding and provenance route. They are not hidden cellular evaluators, and they do not supply an unbounded substrate read or a second route around the displayed* $B \longmapsto T \longmapsto R \longmapsto S$ *path.*

Proof. The sibling rows enter through S and P. The substrate witness S names the cellular surface used to read the embedded row, and P records the provenance of that use beside the local carrier, kernel relations, and naming row.

Route stability fixes how those rows are consumed. The displayed route reaches the sibling surface only after B is encoded as T and placed by R, and H, C transport and replay that route without adding another evaluator.

Provenance exhaustion and non-escape rule out the remaining alternatives. The sibling rows certify the visible finite dependency already present in the packet; they do not create a detached cellular computation, an unbounded orbit read, or a bypass around the carrier's displayed path. □

Theorem 943.10 (CellularCarrierEncoding cell-row accountability)**.** *Every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packet read of the bit encoding and readback row* T *exposes the cell-row embedding coordinate* R *before the carrier can be consumed as a cellular row. The route cannot bypass* R *through a hidden offset, host byte layout, unstated default fill, detached evaluator, or unrecorded row slice.*

Proof. Project the carrier of Definition 943.1. The bit encoding and readback data are stored in T, while the only row that places that word inside a cellular row is R.

Use readback exactness and non-escape, Definition 943.4 and Definition 943.3. They force the route from the source trace through T and then through the displayed embedding coordinate before any cellular carrier read is available.

The excluded alternatives are absent carrier coordinates. Since the packet stores no hidden offset, host byte layout, detached evaluator, unrecorded default fill, or second row slice, the accepted read exposes R as its only cell-row accountability witness. □

Theorem 943.11 (CellularCarrierEncoding substrate witness completeness)**.** *Every substrate-facing read of an accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packet factors through the displayed* $\mathsf{CellularTrustSubstrate}^{\uparrow}$ *sibling row and the existing* $\mathsf{CellularAutomaton}^{\uparrow}$ *dependency named by S and P. No extra substrate carrier, global cellular evaluator, unbounded orbit, or detached trust path is available outside that displayed sibling route.*

Proof. Read the substrate coordinates from the carrier. The row S names the cellular substrate witness, and the provenance row P records the $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{CellularTrustSubstrate}^{\uparrow}$ dependencies used by that witness.

Apply sibling scope binding. It states that the cellular siblings are consumed as visible dependencies of the finite embedding and provenance route, with no hidden evaluator or unbounded substrate read.

Finally combine route stability with provenance exhaustion. The hsame and Cont rows preserve the same B, T, R, S path, and P, N name only that path. Thus a substrate-facing consumer has no route to any extra substrate carrier beyond the displayed sibling rows. □

Theorem 943.12 (CellularCarrierEncoding TasteGate readiness surface)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packet $E = (B, T, R, S, H, C, P, N)$, the carrier definition, falsifiable prediction, independence witness, local* $\mathsf{NameCert}$ *obligations, non-escape theorem, readback exactness, route stability, provenance exhaustion, primitive scope binding, sibling scope binding, cell-row accountability, and substrate witness completeness are exactly the paper rows that a later Lean* $\mathsf{Nontrivial\ CellularCarrierEncodingUp}$ *and* $\mathsf{FieldFaithful\ CellularCarrierEncodingUp}$ *instance must consume. The readiness surface is only the displayed route through B, T, R, S, H, C, P, N and contains no host byte string, hidden row offset, detached evaluator, unbounded orbit, quotient encoding coordinate, or unrecorded substrate witness.*

Proof. Project the carrier of Definition 943.1. It displays the source BHist row, bit encoding and readback row, cell-row embedding coordinate, cellular substrate witness, transports, continuations, provenance, and local naming row.

Use the falsifiable prediction, independence witness, local $\mathsf{NameCert}$ obligations, and non-escape theorem. These identify the admissible consumer route and rule out carrier coordinates for host byte strings, hidden evaluator state, quotient encodings, and unrecorded substrate witnesses.

Readback exactness and route stability then bind the route $B \longmapsto T \longmapsto R \longmapsto S$. The encoding row T is read back from B, the embedding coordinate R is exposed before any cellular row is consumed, and H, C preserve the same displayed route.

Finally combine provenance exhaustion, primitive scope binding, sibling scope binding, cell-row accountability, and substrate witness completeness. They attach the route to the visible $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{CellularTrustSubstrate}^{\uparrow}$ sibling rows through S and P, while excluding hidden row offsets, detached evaluators, unbounded orbits, quotient encoding coordinates, and

unrecorded substrate witnesses. This is precisely the paper-side surface that the Lean TasteGate instances must inhabit. □

Theorem 943.13 (CellularCarrierEncoding status obligation closure)**.** *For every accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packet* $E = (B, T, R, S, H, C, P, N)$*, the carrier exposure, local* NameCert *obligations, non-escape boundary, readback exactness, route stability, provenance exhaustion, primitive scope binding, sibling scope binding, cell-row accountability, substrate witness completeness, and TasteGate readiness surface form the finite obligation surface for the AI-origin packet. Any later Lean TasteGate-facing consumer must use the same displayed rows*

$$B, T, R, S, H, C, P, N$$

rather than a host byte string, hidden row offset, detached cellular evaluator, unbounded orbit, quotient encoding coordinate, or unrecorded substrate witness.

Proof. Begin with the carrier and local naming surface. The carrier of Definition 943.1 exposes exactly B, T, R, S, H, C, P, N, while Definition 943.2 and Definition 943.3 bind the admitted rows and exclude host byte strings, detached evaluators, quotient encodings, and hidden substrate witnesses.

Next read the finite route. Readback exactness, route stability, provenance exhaustion, and primitive scope binding force the consumer path through B, then T, then R, with H, C, P, N preserving and naming that same path. The sibling scope binding, cell-row accountability, and substrate witness completeness theorems attach the route to the displayed $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{CellularTrustSubstrate}^{\uparrow}$ rows without adding a second evaluator.

Finally use the TasteGate readiness surface. Definition 943.12 identifies exactly these rows as the paper surface for later Nontrivial and FieldFaithful instances, and the falsifiable prediction records the concrete row whose failure would invalidate the packet's consumer-facing status. Thus the obligation closure is the displayed finite surface, not any host representation outside it. □

Theorem 943.14 (CellularCarrierEncoding two-witness separation)**.** *Let* $E_0 = (B, T, R_0, S, H, C, P, N)$ *and* $E_1 = (B, T, R_1, S, H, C, P, N)$ *be two accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packets whose displayed cell-row embedding coordinates* R_0 *and* R_1 *differ while the source, bit-readback, substrate, transport, continuation, provenance, and naming rows agree. Then* E_0 *and* E_1 *are the canonical witness pair for a later* Nontrivial $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *instance: the classifier distinguishes them by the* R *coordinate, and no host byte string, orbit quotient, or hidden substrate evaluator can identify them after acceptance.*

Proof. Project both accepted carriers by Definition 943.1. The two packets share B, T, S, H, C, P, N and differ only in the displayed embedding coordinate.

Use cell-row embedding non-escape and cell-row accountability, Definition 943.3 and Definition 943.10. They force every cellular read through the displayed R row before the packet can be consumed as a cellular carrier.

Readback exactness keeps T tied to the common source row B, while route stability keeps H and C from creating a second path around R. Thus the classifier's comparison of R_0 and R_1 is a comparison of carrier data, not a comparison of detached host bytes or orbit representatives.

Finally apply provenance exhaustion and substrate witness completeness. The shared P, N and S rows record only the displayed finite route, so they cannot collapse the two embedding coordinates by a hidden evaluator or unbounded orbit read. The pair E_0, E_1 is therefore the paper-side nontriviality witness. □

Theorem 943.15 (CellularCarrierEncoding field projection faithfulness)**.** *For accepted* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *packets* E *and* E'*, equality of the ordered field projection*

$$(B, T, R, S, H, C, P, N)$$

for E *and* E' *forces equality of the packets. Conversely, a mismatch in any one displayed field is visible to the classifier as a mismatch of the accepted carrier surface.*

Proof. Project the two accepted packets using the carrier definition. A $\mathsf{CellularCarrierEncoding}^{\uparrow}$ packet has no fields beyond the ordered rows B, T, R, S, H, C, P, N, so equality of the whole projection identifies every constructor row of the packet.

For visibility of mismatches, inspect the same projection one row at a time. A mismatch in B changes the source trace, a mismatch in T changes bit encoding or readback, a mismatch in R changes cell placement, and a mismatch in S changes the displayed substrate witness.

The structural fields are visible for the same reason. Route stability preserves the H, C rows as componentwise transport and replay data, while provenance exhaustion preserves P, N as the package and local naming rows of the same finite carrier.

Therefore the classifier cannot hide a field mismatch inside host bytes, quotient representatives, or detached cellular evaluation. Equality of the ordered projection is equality of accepted packets, and inequality in a displayed field remains classifier-visible. □

Theorem 943.16 (CellularCarrierEncoding formal target boundary)**.** *The paper surface of* $\mathsf{CellularCarrierEncoding}^{\uparrow}$ *exposes the exact obligations needed by later formal instances. The witness pair of Definition 943.14 supplies the displayed paper witnesses for a later*

$$\mathsf{Nontrivial\ CellularCarrierEncoding}^{\uparrow}$$

instance. The ordered projection of Definition 943.15 supplies the field-faithful projection consumed by a later

$$\mathsf{FieldFaithful\ CellularCarrierEncoding}^{\uparrow}$$

instance.

Both obligations are read through the existing checked TasteGate alignment surface and through the carrier rows B, T, R, S, H, C, P, N*. No host byte string, hidden row offset, detached cellular evaluator, orbit quotient, or unrecorded substrate witness is part of the target surface.*

Proof. Apply the checked TasteGate alignment surface named at the chapter opening. It fixes the single-carrier alignment for the displayed $\mathsf{CellularCarrierEncoding}^{\uparrow}$ packet and keeps the paper route on the carrier rows already named by the chapter.

For nontriviality, use Definition 943.14. Its two accepted packets agree on B, T, S, H, C, P, N and differ in the displayed R coordinate, so they give the paper witness pair required by the later $\mathsf{Nontrivial}$ instance.

For field faithfulness, use Definition 943.15. Equality of the ordered projection B, T, R, S, H, C, P, N is equality of accepted packets, and any mismatch in one displayed field remains classifier-visible. Thus the same projection is the ordered-field surface consumed by the later $\mathsf{FieldFaithful}$ instance. □

The Lean follow-up must provide `Nontrivial CellularCarrierEncodingUp` and `FieldFaithful CellularCarrierEncodingUp` instances in `lean4/BEDC/Derived/CellularCarrierEncodingUp/TasteGate.lean` with the canonical alignment surface expected at `BEDC.Derived.CellularCarrierEncodingUp.CellularCarrier`

Theorem 943.17 (CellularCarrierEncoding bridge-facing route)**.** *For every accepted* CellularCarrierEncoding$^{\uparrow}$ *packet, the public rows*

$$B, T, R, S, H, C, P, N$$

form the bridge-facing route from the source BHist *row to the displayed cellular automaton and cellular trust substrate surfaces. The route exposes bit encoding and readback, cell-row placement, substrate witness, componentwise transport, displayed continuation, package provenance, and local* NameCert *rows as one finite public certificate, and it exports no host byte string, hidden row offset, detached cellular evaluator, orbit quotient, or unrecorded substrate witness.*

Proof. Start with Definition 943.16. It fixes the paper surface on the carrier rows consumed by the checked TasteGate alignment.

Apply Definition 943.3 and Definition 943.4. The source trace, bit encoding, readback, and cell-row placement are visible only through B, T, R and cannot be replaced by host bytes or hidden offsets.

Next use Definition 943.5 and Definition 943.6. They keep the substrate surface, transport rows, continuation rows, package provenance, and local naming row on S, H, C, P, N.

Therefore the public packet already presents the bridge-facing route from BHist source rows to the cellular automaton and cellular trust substrate surfaces, while bridge checking itself remains outside this paper-axis claim. □

Closure status for NameCert$_{\text{CellularCarrierEncoding}^{\uparrow}}$.

Constructive story: A CellularCarrierEncoding$^{\uparrow}$ packet is generated from a source BHist row, bit encoding and readback row, cell-row embedding coordinate, cellular substrate witness row, componentwise transports, displayed continuations, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{CellularCarrierEncoding}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 943.1, Definition 943.2, Definition 943.3, Definition 943.4, Definition 943.5, Definition 943.6, Definition 943.8, Definition 943.9, Definition 943.7, Definition 943.16, Definition 943.17, and

Lean4 ✓ `BEDC.Derived.CellularCarrierEncodingUp.taste gate`

bind the scoped and bridge-facing B, T, R, S, H, C, P, N route over BHist, hsame, Cont, Pkg, NameCert, CellularAutomaton$^{\uparrow}$, and CellularTrustSubstrate$^{\uparrow}$ together with the checked TasteGate single-carrier alignment.

Formal verification: FormalStatus(NameCert$_{\text{CellularCarrierEncoding}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CellularCarrierEncodingUp.CellularCarrierEncodingTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bit-readback Lean verification, cell-row embedding uniqueness, substrate replay completeness, or hidden offset exclusion beyond the displayed packet.

Upgrade path: No further paper-theory upgrade remains; bridge checking remains on the verification axis.

Chapter 944

A Concrete Naming Certificate for $\mathsf{BoundedUniversalTrace}^{\uparrow}$

The $\mathsf{BoundedUniversalTrace}^{\uparrow}$ packet is the finite host-side trace surface for the strong-normalisation boundary of chapter 9370. It combines the bounded machine substrate of chapter 280, the cellular substrate row of chapter 934, and the finite continuation ledger of chapter 469. The object does not assert that an unbounded computation halts; it exposes exactly the fuel-indexed rows from which a total host may read a terminal row.

Falsifiable prediction: Removing the fuel row permits a consumer to read terminal output without a displayed finite trace ledger, which the local classifier rejects.

Independence witness: turingmachine, continuationtrace, cellularautomaton, continuationbigstep, haltingdistinctiontrace: none is bijective because $\mathsf{BoundedUniversalTrace}^{\uparrow}$ carries the fuel-bound, substrate row, and trace ledger in one packet, while each sibling lacks at least one of those coordinates or replaces the bounded packet by a different semantic surface.

Definition 944.1 (BoundedUniversalTrace carrier)**.** *A* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, S, T, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the fuel row, naming the finite step bound used by the total host read;*
2. *M is a substrate row drawn from the displayed* $\mathsf{TuringMachine}^{\uparrow}$ *or* $\mathsf{CellularAutomaton}^{\uparrow}$ *interface;*
3. *S is the finite source row containing the initial configuration, initial cellular window, or encoded substrate input;*
4. *T is the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger listing the step rows admitted below the bound F;*
5. *R is the terminal or readback row selected from the displayed ledger T;*
6. *H records componentwise* hsame *transports for F, M, S, T, R, P, N;*
7. *C records the* Cont *routes that replay the bounded step ledger and the terminal readback;*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{TuringMachine}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only by the displayed fuel, substrate, source, trace, readback, transport, route, provenance, and naming rows. It has no coordinate for an unbounded halting predicate, host evaluator equality, quotient behaviour, selected nontermination evidence, or universal-machine semantic completion.

Theorem 944.2 (BoundedUniversalTrace NameCert obligations)**.** *For the carrier of Definition 944.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier exposure for* $U = (F, M, S, T, R, H, C, P, N)$*;*
2. *classifier transport by componentwise* hsame *over the displayed fuel, substrate, source, trace, readback, package, and naming rows;*
3. *step-indexed readback through* F*,* T*, and* R*;*
4. *non-escape excluding unbounded halting, host evaluator equality, quotient behaviour, selected nontermination evidence, and universal-machine semantic completion;*
5. *public export of the same finite packet for Lean formalization under* BEDC.Derived.BoundedUniversalTraceUp.

No consumer can cite a total-host read of a universal substrate computation unless the read passes through these five rows.

Lean4 ✓ `BEDC.Derived.BoundedUniversalTraceUp.BoundedUniversalTraceCarrier namecert obligations`

Proof. Carrier exposure is projection from Definition 944.1. The accepted packet displays exactly the fuel, substrate, source, finite continuation ledger, readback, transport, route, provenance, and naming rows.

Classifier transport is componentwise. A comparison between accepted packets may transport rows through H and replay routes through C, but it cannot compare hidden host evaluators, unbounded runs, quotient behaviours, or semantic completions because no such coordinate is carried.

The remaining rows are the bounded readback, non-escape, and public-export statements proved below. Together they exhaust the NameCert surface because every consumer-facing read is either a projection from the packet, a displayed route through the finite ledger, or a refusal caused by an absent coordinate. □

Theorem 944.3 (BoundedUniversalTrace step-indexed readback)**.** *Every public read of the terminal row* R *in an accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier factors through the fuel row* F *and the finite trace ledger* T*. The read is admitted only when* T *displays a bounded route from the source row* S *over the substrate row* M *and* R *is the terminal or readback row selected by that same finite ledger.*

Proof. Project the accepted packet through Definition 944.1. The only computation-facing coordinates are the substrate row M, source row S, fuel row F, trace ledger T, readback row R, and route row C.

Apply the continuation-trace reading of chapter 469. A step read is public only as a finite ledger row, so selecting R requires the displayed ledger entry in T and the replay route in C.

The fuel row F bounds the ledger inspected by the host. Since there is no coordinate for an unbounded trace, a consumer cannot read R by bypassing F or by replacing T with a host-level run. □

Theorem 944.4 (BoundedUniversalTrace non-escape)**.** *An accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *packet exports no unbounded halting predicate, no equality between host evaluators, no quotient behaviour, no selected nontermination witness, and no universal-machine semantic completion. Its visible computational content is exactly the finite packet* $F, M, S, T, R, H, C, P, N$ *of Definition 944.1.*

Proof. Inspect the carrier coordinates. The substrate row M may name a $\mathsf{TuringMachine}^{\uparrow}$ or $\mathsf{CellularAutomaton}^{\uparrow}$ surface, but the packet stores only the bounded substrate row selected for this trace.

The trace row T is finite and is read through the continuation surface of chapter 469. The readback theorem above forces the terminal row through F and T, so it supplies a step-indexed host read rather than an unbounded semantic claim.

The excluded objects require coordinates not present in the packet. Unbounded halting needs a global run predicate, host evaluator equality needs a host semantic equality row, quotient behaviour needs a quotient classifier, selected nontermination needs an infinite negative witness, and universal-machine semantic completion needs an all-input semantic surface. None is among $F, M, S, T, R, H, C, P, N$. □

Theorem 944.5 (BoundedUniversalTrace public export)**.** *The public* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *export is the finite carrier of Definition 944.1 together with the NameCert obligations of Definition 944.2, the step-indexed readback of Definition 944.3, and the non-escape boundary of Definition 944.4. This export is the finite bridge surface by which the total host of chapter 9370 may inspect universal-substrate traces without importing halting, partiality, host evaluator equality, quotient behaviour, or semantic completion.*

Proof. The carrier supplies the displayed packet $F, M, S, T, R, H, C, P, N$. The NameCert theorem binds carrier exposure, classifier transport, readback, non-escape, and public export to those rows.

The readback theorem accounts for the only public terminal read. It must pass through the fuel row and finite trace ledger, so the host-facing surface remains step-indexed.

The non-escape theorem records what the packet cannot export. Repacking the carrier, readback, and refusal clauses gives a public certificate surface over BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, with the substrate-side anchors supplied by $\mathsf{TuringMachine}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, and $\mathsf{ContinuationTrace}^{\uparrow}$. □

Theorem 944.6 (BoundedUniversalTrace fuel determinacy)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, the terminal readback row* R *is determined by the displayed fuel row* F *together with the finite trace ledger* T*. Two accepted reads with the same* F*, the same substrate and source rows* M, S*, and the same ledger* T *have the same public terminal row after the componentwise transports and routes* H, C, P, N *are applied; no unbounded halting predicate or host evaluator equality contributes to the determination.*

Proof. Project the carrier through Definition 944.1. The only computation-facing coordinates are $F, M, S, T, R, H, C, P, N$.

Use the public export of Definition 944.5. It exposes terminal readback through the same finite packet, so a read of R must pass through the fuel row F and the finite trace ledger T.

If two reads share F, M, S, T, the remaining comparison is componentwise transport and replay through H, C, P, N. Since the carrier includes no unbounded run row and no host evaluator equality row, those absent objects cannot distinguish the terminal readback. □

Theorem 944.7 (BoundedUniversalTrace substrate readback)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, every substrate read factors through the displayed* $\mathsf{TuringMachine}^{\uparrow}$

or $\mathsf{CellularAutomaton}^{\uparrow}$ *row* M *and the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger* T *before reaching the terminal row* R*. The packet exports a bounded substrate read, not a host evaluator for the unbounded substrate.*

Proof. Inspect the substrate coordinate in Definition 944.1. The row M is explicitly drawn from the $\mathsf{TuringMachine}^{\uparrow}$ or $\mathsf{CellularAutomaton}^{\uparrow}$ interface, while T is the finite continuation ledger below the fuel bound.

Apply Definition 944.3 and Definition 944.5. The terminal row is public only when the displayed continuation ledger routes the source row S over the substrate row M within the fuel row F.

Thus substrate information reaches consumers through M, T, R, H, C, P, N. No consumer receives an evaluator equality, an unbounded transition closure, or a semantic completion of the substrate. □

Theorem 944.8 (BoundedUniversalTrace ledger exhaustion)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, the finite trace ledger* T *exhausts every step row visible below the fuel bound* F*. A public consumer can inspect only the source row* S*, substrate row* M*, ledger rows in* T*, terminal readback* R*, and the displayed transport, route, provenance, and naming rows* H, C, P, N*; it cannot add hidden steps, appeal to unbounded halting, or identify traces by host evaluator equality.*

Proof. Read the accepted packet as the finite carrier $U = (F, M, S, T, R, H, C, P, N)$. The trace rows available below the bound are exactly the entries displayed in T.

Use Definition 944.3. Any terminal read must route through the same ledger T and the same fuel row F, so every visible step used by the read is already part of the displayed finite trace.

Finally apply the non-escape boundary of Definition 944.4. Hidden steps, unbounded halting, quotient behaviour, and host evaluator equality require coordinates absent from the carrier, so the ledger T exhausts the accepted bounded trace surface. □

Theorem 944.9 (BoundedUniversalTrace root fuel schedule)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, the root fuel row* F *is a unary* BHist *schedule consumed only by finite* Cont *replay through the trace ledger* T *and terminal readback row* R*. A public read below the bound must cite the same fuel row, the same ledger row, componentwise* hsame *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$ *row.*

Proof. Project the accepted carrier through Definition 944.1. The fuel coordinate is the displayed row F, and the only replay data attached to it are the finite trace ledger T, the readback row R, and the route coordinate C.

Apply Definition 944.3 and Definition 944.6. They force every terminal read to pass through F and T, and they determine the readback row from the displayed bound, substrate, source, and ledger rows after the listed transports and routes are applied.

Finally use Definition 944.8. Since the ledger exhausts the steps visible below the bound, the root fuel row is not an implicit evaluator or host scheduler; it is exactly the unary BHist schedule consumed by finite Cont replay. □

Theorem 944.10 (BoundedUniversalTrace obligation surface lock)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, a total-host read is available only after the packet exposes exactly the fuel, substrate, source, finite trace, readback, transport, route, provenance, and local naming rows* $F, M, S, T, R, H, C, P, N$*. No public read may replace this surface by an unbounded evaluator or by a partial host trace.*

Proof. Project the carrier of Definition 944.1. It displays exactly the nine rows $F, M, S, T, R, H, C, P, N$.

Use Definition 944.2. The local certificate admits carrier exposure, componentwise transport, bounded readback, non-escape, and public export only over those displayed rows.

Apply Definition 944.3 and Definition 944.4. They force the terminal row through F and T while refusing unbounded halting, host evaluator equality, quotient behaviour, selected nontermination evidence, and semantic completion. □

Theorem 944.11 (BoundedUniversalTrace fuel row public boundary)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, the row* F *is the only public bound for trace replay. The packet projects no unbounded halting predicate, no all-fuel convergence claim, and no host evaluator that can substitute for the displayed fuel row.*

Proof. Begin with the root fuel schedule of Definition 944.9. It identifies F as the unary BHist schedule consumed by finite Cont replay through T and R.

Use Definition 944.3. A read of R is public only through the fuel row and finite trace ledger, so the bound cannot be replaced by a host run.

Finally apply Definition 944.4. Unbounded halting and host evaluator equality require coordinates absent from Definition 944.1, leaving F as the sole public trace bound. □

Theorem 944.12 (BoundedUniversalTrace substrate trace readback)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier* $U = (F, M, S, T, R, H, C, P, N)$*, substrate reads must pass through the substrate row* M *and the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *row* T *before the readback row* R *is readable. The read is a finite carrier projection through* H, C, P, N*, not a direct host evaluation of the substrate.*

Proof. Project M and T from Definition 944.1. The substrate row names the $\mathsf{TuringMachine}^{\uparrow}$ or $\mathsf{CellularAutomaton}^{\uparrow}$ surface, and the trace row records the finite continuation ledger below the fuel bound.

Apply Definition 944.3. The terminal row R is selected only when T displays the bounded route from S over M.

Use Definition 944.4. Since the carrier lacks a host evaluator and lacks an unbounded transition closure, substrate readback can only replay M and T through the displayed transport, route, provenance, and local naming rows. □

Theorem 944.13 (BoundedUniversalTrace cellular replay scope)**.** *When the substrate row* M *of an accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier names the* $\mathsf{CellularAutomaton}^{\uparrow}$ *dependency of chapter 934, every cellular read is made only through that sibling cellular replay surface, the finite ledger* T*, local* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$ *row. The packet does not expose an unbounded cellular orbit, host cellular evaluator, hidden update table, or global orbit classifier.*

Proof. Read the carrier row named by Definition 944.1. The substrate coordinate M may be drawn from the displayed $\mathsf{CellularAutomaton}^{\uparrow}$ interface, but the accepted packet stores only this local substrate row together with the finite source, ledger, readback, transport, route, provenance, and naming rows.

Use Definition 944.7. It requires every substrate read to pass through M and the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger T before the terminal row R is reached. The sibling dependency named by the carrier supplies the cellular replay surface, while P records the local provenance of that use.

The non-escape boundary Definition 944.4 removes any larger cellular semantics. Since the carrier has no coordinate for an unbounded orbit, host cellular evaluator, hidden update table,

or global orbit classifier, the cellular dependency is read only through the displayed finite replay scope. □

Theorem 944.14 (BoundedUniversalTrace TuringMachine non-escape)**.** *When the substrate row M of an accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier names the* $\mathsf{TuringMachine}^{\uparrow}$ *dependency of chapter 280, that dependency supplies finite transition evidence only through M, S, T, R, H, C, P, and N. It supplies no unbounded evaluator, no divergence oracle, and no total halting classifier.*

Proof. Begin with the chapter carrier. In Definition 944.1, the Turing-machine dependency can occur only as the displayed substrate row M of the accepted finite packet $U = (F, M, S, T, R, H, C, P, N)$.

Apply Definition 944.7 and Definition 944.8. The terminal read uses the substrate row together with the finite continuation ledger T and the fuel row F, so every transition fact visible to the consumer is already listed in the bounded ledger and replayed by C.

Finally use Definition 944.4. An unbounded evaluator, divergence oracle, or total halting classifier would require a coordinate outside the displayed carrier. Since no such coordinate exists, the $\mathsf{TuringMachine}^{\uparrow}$ row contributes only finite transition evidence inside the local package. □

Theorem 944.15 (BoundedUniversalTrace fuel-substrate-ledger separation)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier $U = (F, M, S, T, R, H, C, P, N)$, the fuel row F, substrate row M, and finite continuation ledger T are pairwise distinct public roles. A consumer may transport these roles componentwise through H, C, P, N, but may not collapse fuel into substrate, substrate into ledger, or ledger into the terminal readback row R.*

Proof. Project the carrier of Definition 944.1. The row F names the finite step bound, M names the displayed Turing-machine or cellular substrate, and T names the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger. They are introduced as separate coordinates of the same packet.

Apply Definition 944.3. Terminal readback must pass through F and T, so the fuel row bounds the read and the ledger supplies the displayed steps rather than one row replacing the other.

Next use Definition 944.7. Substrate information is read through M before the ledger reaches R, which prevents the substrate role from being identified with either the fuel schedule or the ledger rows.

Finally apply Definition 944.8. The ledger T exhausts the visible step rows below F, while H, C, P, N only transport, replay, cite, and name the same finite packet. Hence componentwise transport preserves the three public roles without collapsing them into one another or into terminal readback. □

Theorem 944.16 (BoundedUniversalTrace TasteGate readiness package)**.** *For every accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier*

$$U = (F, M, S, T, R, H, C, P, N),$$

the rows $F, M, S, T, R, H, C, P, N$ form the exact TasteGate readiness package for a single future carrier. The Nontrivial *instance must distinguish at least two packets by one displayed row, the* FieldFaithful *instance must project all nine fields, and no proof may collapse fuel, substrate, ledger, or readback into a hidden host evaluator.*

Lean4 ✓ `BEDC.Derived.BoundedUniversalTraceUp.BoundedUniversalTraceTasteGate single carrier alignment`

Proof. Start with the carrier of Definition 944.1. It displays the fuel row F, substrate row M, finite source row S, continuation ledger T, terminal readback row R, transport row H, route row C, package row P, and local name row N as separate fields of one finite packet.

Apply Definition 944.2, Definition 944.3, and Definition 944.4. These rows bind admission, bounded readback, and the refusal of any unbounded evaluator, divergence oracle, or total halting classifier outside the displayed carrier.

Next use Definition 944.6, Definition 944.8, and Definition 944.15. They keep F, M, T, and R as distinct public roles while H, C, P, N only transport, replay, cite, and name the same finite surface.

Thus a future TasteGate carrier has one exact package to encode. Nontriviality can witness distinct packets by changing a displayed row, and field faithfulness must project each of the nine rows rather than replacing the finite trace by host evaluation. □

Theorem 944.17 (BoundedUniversalTrace carrier transport obligation)**.** *For every accepted* BoundedUniversalTrac *carrier $U = (F, M, S, T, R, H, C, P, N)$, componentwise* hsame *transport preserves the fuel row F, substrate row M, source row S, finite trace ledger T, terminal readback row R, route row C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$ *row N as separate public coordinates. The transport row H may identify corresponding coordinates of two accepted packets, but it cannot collapse fuel into substrate, substrate into ledger, ledger into readback, or any displayed row into a host evaluator.*

Proof. Project the accepted packet through Definition 944.1. The carrier lists $F, M, S, T, R, H, C, P, N$ as the complete finite row family.

Use the obligation surface of Definition 944.2. Its classifier transport clause is componentwise over the displayed fuel, substrate, source, trace, readback, package, and naming rows.

Apply the fuel-substrate-ledger separation of Definition 944.15. It prevents the transported rows from being collapsed into one another, while Definition 944.4 excludes host evaluator equality as a replacement coordinate. □

Theorem 944.18 (BoundedUniversalTrace terminal readback obligation)**.** *Every terminal readback from an accepted* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *carrier is public only through the displayed fuel row F and the finite continuation ledger T. The read may cite the substrate siblings* $\mathsf{TuringMachine}^{\uparrow}$ *and* $\mathsf{CellularAutomaton}^{\uparrow}$ *through chapter 280 and chapter 934, and it may cite the ledger sibling* $\mathsf{ContinuationTrace}^{\uparrow}$ *through chapter 469; it cannot bypass F and T by a host evaluator, unbounded halting predicate, or hidden continuation.*

Proof. Begin with the step-indexed readback theorem Definition 944.3. It makes R readable only after the fuel row and finite trace ledger have been displayed.

Use the substrate readback theorem Definition 944.7. It routes substrate information through the $\mathsf{TuringMachine}^{\uparrow}$ or $\mathsf{CellularAutomaton}^{\uparrow}$ row M and the finite $\mathsf{ContinuationTrace}^{\uparrow}$ ledger T before terminal readback.

Finally apply Definition 944.8 and Definition 944.4. The ledger exhausts visible steps below the bound, and the absent unbounded halting, host evaluator, and hidden continuation coordinates cannot substitute for F and T. □

Theorem 944.19 (BoundedUniversalTrace Nontrivial and FieldFaithful obligation)**.** *The paper-side obligation for the Lean* TasteGate *target over* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *is to provide explicit* Nontrivial *and* FieldFaithful *instances over the same nine displayed rows*

$$F, M, S, T, R, H, C, P, N.$$

Nontriviality must distinguish accepted packets by at least one displayed row, and field faithfulness must project fuel, substrate, source, trace ledger, readback, transport, route, provenance, and local naming without replacing any row by host evaluation or unbounded computation semantics.

Proof. Read the TasteGate readiness package of Definition 944.16. It already identifies the exact nine rows that a single carrier must encode.

Use the carrier transport obligation above to keep the nine coordinates separate under hsame transport. A FieldFaithful instance must therefore project each displayed coordinate rather than identifying roles by a host-side semantic equality.

Use the terminal readback obligation and the non-escape boundary of Definition 944.4. They force terminal computation through the finite fuel and ledger rows, so a Nontrivial witness must vary a displayed packet row and cannot appeal to unbounded computation or hidden evaluator behaviour. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedUniversalTrace}^{\uparrow}$ packet is generated from a fuel row, a Turing-machine or cellular-automaton substrate row, a finite source row, a finite continuation-trace ledger, a terminal readback row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 944.5 exposes the finite fuel, substrate, source, trace, readback, transport, route, provenance, and naming rows as the public packet, while

Lean4 (definition) `BEDC.Derived.BoundedUniversalTraceUp.taste gate`

and the checked TasteGate readiness package in Definition 944.16 record the matching carrier alignment.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedUniversalTrace}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BoundedUniversalTraceUp.BoundedUniversalTraceTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, partial evaluator semantics, quotient behaviour, selected nontermination evidence, or universal-machine semantic completion.

Upgrade path: No paper theory-axis grade remains; bridge checking remains an independent formal-axis obligation for the same bounded trace surface.

Chapter 945

A Concrete Naming Certificate for $\mathsf{CellularSubstrateClassifier}^{\uparrow}$

The $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ packet realizes the classifier-facing half of the carrier alternative of chapter 9391. It sits after the rule-table row of chapter 934, the trust-substrate audit row of chapter 937, the finite orbit-window row of chapter 938, and the BHist carrier packet of chapter 942. Its role is not to replace the BHist kernel; it certifies the finite comparison by which a BHist row, a bit-row encoding, and a cellular orbit window are read as the same accepted carrier observation.

Falsifiable prediction: Any accepted $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ carrier comparison must expose the displayed BHist row, bit-row encoding, orbit-window row, classifier row, transport row, route row, provenance row, and local NameCert row. A packet omitting any one of these rows is not an accepted classifier comparison.

Independence witness: cellularautomaton, cellulartrustsubstrate, cellularorbitsubstrate, interhistlocalityledger, observerhistidentity: the classifier packet is not bijective to any listed sibling because it simultaneously stores the BHist source row, the bit-row encoding, the cellular orbit window, and the comparison classifier surface, while each sibling omits at least one of those rows.

Definition 945.1 (Cellular substrate classifier carrier)**.** *A* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, E, O, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the BHist source row, naming the finite BEDC carrier observation being compared.*
2. *E is the bit-row encoding of B, including the empty-row convention and the constructor-to-bit direction used by the comparison.*
3. *O is the finite cellular orbit-window row whose initial displayed cells read back the encoded bit row.*
4. *F is the classifier row that admits the comparison between B, E, and O.*
5. *H records componentwise* hsame *transport for the BHist source, bit-row encoding, orbit-window row, classifier row, provenance row, and local naming row.*
6. *C records the* Cont *route from the BHist row through bit-row encoding, orbit-window readback, and classifier admission.*

7. *P is the* Pkg *provenance over* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{CellularOrbitSubstrate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the originating carrier-alternative vision.*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifier}^{\uparrow}}$ *row for carrier admission, classifier accountability, BHist-orbit compatibility, exactness, provenance, and non-escape.*

The classifier compares accepted carrier observations only through these displayed rows. It has no coordinate for host equality between cellular histories, an infinite orbit, a hidden cellular evaluator, a detached bit-string quotient, or a carrier switch outside the finite NameCert packet.

Lean4 ✓ `BEDC.Derived.CellularSubstrateClassifierUp.TasteGate.CellularSubstrateClassifierTasteGate`
`single carrier alignment`

Definition 945.2 (Cellular substrate classifier obligation surface)**.** *The local obligation surface of* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ *consists of eight rows: carrier exposure for $K = (B, E, O, F, H, C, P, N)$, BHist source visibility, bit-row encoding visibility, orbit-window visibility, classifier accountability, componentwise transport through H, continuation replay through C, and public naming through P, N.*

An accepted consumer may cite the BHist-to-cellular carrier comparison only after these eight rows are present. The surface therefore binds the cellular carrier alternative to finite NameCert evidence rather than to a host-side assertion that the two carriers are interchangeable.

Theorem 945.3 (Cellular substrate classifier NameCert boundary)**.** *For the carrier of Definition 945.1, the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifier}^{\uparrow}}$ *boundary is exactly the obligation surface of Definition 945.2. In particular, a public carrier-comparison read is accepted only when it displays the BHist row, bit-row encoding, orbit-window row, classifier row, transport row, route row, provenance row, and local naming row.*

Proof. Carrier exposure is projection from Definition 945.1. The tuple displays B, E, O, F, H, C, P, N as its only coordinates.

The source, encoding, orbit, and classifier obligations are the first four analytic rows of Definition 945.2. The transport and route obligations are supplied by H and C, and public naming is supplied by P, N.

No other coordinate is available to the packet. Thus the local NameCert boundary is exactly the displayed obligation surface, and every accepted public comparison read passes through those eight rows. □

Theorem 945.4 (Cellular substrate classifier BHist-orbit compatibility)**.** *If two accepted* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ *packets have* hsame *BHist source rows and* hsame *bit-row encodings, then their accepted orbit-window comparisons agree at every displayed classifier read. Equivalently, the classifier cannot expose two distinct cellular carrier reads for the same transported BHist source and bit-row encoding.*

Proof. Project both packets to their BHist source, bit-row encoding, orbit-window, and classifier rows. The hsame hypotheses identify the two source rows and the two bit-row rows componentwise.

Use the continuation route in each packet to replay the comparison from B through E into the displayed orbit window O. The classifier row F is permitted to read only those displayed source, encoding, and orbit-window coordinates.

A second accepted cellular carrier read for the same transported B and E would require either a different orbit-window coordinate or a hidden classifier coordinate. Neither belongs to the carrier

of Definition 945.1, so the displayed classifier read is compatible with the transported BHist source. □

Theorem 945.5 (Cellular substrate classifier non-escape)**.** *Every accepted* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ *carrier comparison factors through the finite route*

$$B \longmapsto E \longmapsto O \longmapsto F \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route exports no infinite orbit, host cellular-history equality, hidden evaluator, detached bit-string quotient, or carrier replacement outside the local NameCert row.

Proof. Begin with an accepted carrier comparison. By Definition 945.3, the comparison must display the BHist source row, bit-row encoding, orbit-window row, classifier row, transport row, route row, provenance row, and local naming row.

The factorization route is exactly the order in which those rows are consumed. The BHist row is encoded as a bit row, the bit row is read through the finite orbit window, the classifier admits the comparison, and the transport, continuation, provenance, and local NameCert rows package the public read.

The carrier has no field for an infinite orbit, host cellular-history equality, hidden evaluator, detached quotient, or external carrier replacement. Therefore every accepted comparison remains inside the finite classifier packet. □

Theorem 945.6 (Cellular substrate classifier orbit-window exhaustion)**.** *Every accepted* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ *comparison consumes exactly the BHist source row, bit-row encoding row, orbit-window row, classifier row, transport row, route row, provenance row, and local NameCert row before any cellular carrier comparison is public. No accepted comparison may replace the orbit-window read with source-equivalence transport, host cellular-history equality, an infinite orbit, or a detached bit-string quotient.*

Proof. The NameCert boundary theorem Definition 945.3 identifies the public boundary with the eight displayed rows of Definition 945.2. Thus any accepted public comparison must expose B, E, O, F, H, C, P, N.

The BHist-orbit compatibility theorem Definition 945.4 binds the transported BHist source and bit-row encoding to the displayed orbit-window comparison. A second cellular carrier read for the same transported source would require an additional orbit or classifier coordinate, which the carrier does not contain.

Finally apply the non-escape theorem Definition 945.5. Its finite route is the same eight-row route, so coordinate exhaustion leaves no public comparison outside the BHist source row, bit-row encoding, orbit-window row, classifier row, transport row, route row, provenance row, and local NameCert row. □

Theorem 945.7 (Cellular substrate classifier obligation closure)**.** *The carrier of Definition 945.1, the obligation surface of Definition 945.2, the NameCert boundary, BHist-orbit compatibility, non-escape route, orbit-window exhaustion, falsifiable prediction, independence witness, and TasteGate commitments form the obligation-closure package for* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$.

The package is exactly over the displayed rows B, E, O, F, H, C, P, N*. It refuses infinite orbit data, host cellular-history equality, a hidden evaluator, detached bit-string quotients, and replacement of the carrier outside the displayed packet.*

Proof. Begin with the displayed carrier $K = (B, E, O, F, H, C, P, N)$. The obligation surface records precisely the BHist source row B, bit-row encoding E, finite orbit-window row O, classifier row F, componentwise transport row H, continuation row C, package provenance P, and local NameCert row N.

Apply the NameCert boundary, BHist-orbit compatibility, non-escape, and orbit-window exhaustion theorems. These results identify the accepted public comparison with the same eight rows, force transported BHist and bit-row reads through the displayed orbit window, keep the route finite, and prevent any comparison from bypassing the orbit-window row.

The falsifiable prediction and independence witness test the same surface from the consumer side: omitting one displayed row breaks acceptance, and no listed sibling stores the full BHist, bit-row, orbit-window, classifier, transport, route, provenance, and local naming packet. The TasteGate commitments are therefore attached to the displayed carrier rather than to an external evaluator.

No remaining coordinate exists for an infinite orbit, host cellular-history equality, hidden evaluator, detached bit-string quotient, or carrier replacement. Thus the listed rows and witnesses assemble the obligation-closure package claimed here. □

The later Lean namespace `BEDC.Derived.CellularSubstrateClassifierUp.TasteGate` must provide `Nontrivial CellularSubstrateClassifierUp` and `FieldFaithful CellularSubstrate ClassifierUp` instances for the displayed classifier carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularSubstrateClassifier}^{\uparrow}$ packet is generated from a displayed BHist source row, bit-row encoding, finite orbit-window row, classifier row, componentwise transport row, continuation route row, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifier}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 945.1, Definition 945.2, Definition 945.3, Definition 945.4, Definition 945.5, Definition 945.6, Definition 945.7, and

Lean4 ✓ `BEDC.Derived.CellularSubstrateClassifierUp.taste gate`

specify the BHist source row, bit-row encoding, orbit-window row, classifier row, route row, TasteGate, Nontrivial, and FieldFaithful surface as a bridge-compatible finite classifier package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifier}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CellularSubstrateClassifierUp.TasteGate.CellularSubstrateClassifierTasteGate\_single\_c`

Bridge status: none

Origin: ai

Not claimed: This chapter does not replace the BHist kernel, add a cellular automaton substrate axiom, prove infinite orbit adequacy, supply host cellular-history equality, or close bridge checking.

Upgrade path: No further paper-axis grade remains for this classifier package; bridge checking stays on the independent verification axis.

Chapter 946

A Concrete Naming Certificate for $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$

The $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$ packet realizes the two-axis map of chapter 9388 as a finite NameCert object. It does not decide universality or closure for an arbitrary substrate. It records the displayed evidence by which a consumer reads a substrate as one named quadrant entry, using the cellular rule surface of chapter 934, the carrier surface of chapter 942, the finite obstruction boundary of chapter 490, and the bounded trace surface of chapter 944 as sibling anchors.

Falsifiable prediction: Any closed-quadrant consumer that reads a $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$ packet without the displayed closure-axis row is rejected by the local classifier.

Independence witness: cellularautomaton, cellularsubstrate, haltingobstruction, boundeduniversaltrace: no listed sibling is bijective to $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$ because the packet carries both axis-pair rows and the quadrant-tag row, while each sibling supplies only a rule, carrier, obstruction, or bounded-trace surface without the paired quadrant classifier.

Definition 946.1 (Closure-universality carrier)**.** *A* $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, D, G, S, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the universality-axis row, recording whether the accepted substrate read is universal, subuniversal, or refused for lack of displayed evidence.*
2. *D is the closure-axis row, recording whether the accepted observation channel is closed, open, or refused for lack of displayed evidence.*
3. *G is the quadrant-tag row determined by the displayed pair (U, D).*
4. *S is the substrate-witness row, naming the finite sibling surface from which the substrate read is consumed.*
5. *A is the sibling-anchor ledger, containing the relevant* $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{HaltingObstruction}^{\uparrow}$, *and* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *rows used by this packet.*
6. *H records componentwise* hsame *transports for U, D, G, S, A, P, N.*
7. *C records displayed* Cont *routes through the two axis rows, the quadrant tag, the substrate witness, and the sibling-anchor ledger.*

8. *P is the* Pkg *provenance over* $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$*, the quadrant vision, the sibling concrete chapters,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosureUniversalityQuadrant}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only by these rows. It has no coordinate for a host proof of arbitrary universality, an ambient closure oracle, a hidden substrate evaluator, or a quadrant tag detached from the displayed axis pair.

Lean4 ✓ `BEDC.Derived.ClosureUniversalityQuadrantUp.ClosureUniversalityQuadrantTasteGate single carrier alignment`

Theorem 946.2 (Closure-universality quadrant NameCert obligations)**.** *For the carrier of Definition 946.1, the local* $\mathsf{NameCert}_{\mathsf{ClosureUniversalityQuadrant}^{\uparrow}}$ *surface consists exactly of carrier exposure for $Q = (U, D, G, S, A, H, C, P, N)$, universality-axis accountability through U, closure-axis accountability through D, quadrant-tag determinacy through G, substrate-witness exposure through S, sibling-anchor replay through A, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. A consumer cannot read a closed quadrant, an open quadrant, a universal quadrant, or a sub-universal quadrant unless the two axis rows and their determined tag are both present.*

Proof. Carrier exposure is projection from Definition 946.1. The accepted packet displays the two axis rows, quadrant tag, substrate witness, sibling-anchor ledger, transports, continuations, provenance, and local name as its only coordinates.

Axis accountability is componentwise. The universality read is taken from U, the closure read is taken from D, and the tag row G is admitted only as the finite classifier row for that displayed pair.

The sibling ledger A and substrate witness S connect the abstract quadrant entry to concrete surfaces already present in the dossier. The consumer may replay those anchors through C and transport them through H, but it cannot replace them with a hidden universality proof, closure oracle, or host substrate evaluator.

Finally P and N package the same finite route as the public naming surface. Since no detached quadrant tag or ambient axis oracle is carried, the listed rows exhaust the local NameCert obligations. □

Theorem 946.3 (Closure-universality axis independence boundary)**.** *An accepted* $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$ *packet exposes the universality-axis row U and the closure-axis row D as separate displayed coordinates, and the quadrant tag G is read only after both coordinates are present. Thus a consumer may transport or compare the two axes componentwise, but it may not infer either axis from the other, collapse the pair to a single sibling certificate, or replace the tag by a host-level substrate classification.*

Proof. Inspect the carrier of Definition 946.1. The universality row, closure row, and quadrant tag are distinct coordinates, so projection of one axis does not project the other.

The tag row G is a classifier read from the displayed pair (U, D). Therefore a tag comparison may use the componentwise transports in H and the route replay in C, but it must keep both axis rows visible.

The sibling rows in A witness non-bijection rather than collapse. $\mathsf{CellularAutomaton}^{\uparrow}$ supplies rule and orbit rows, $\mathsf{CellularSubstrate}^{\uparrow}$ supplies carrier rows, $\mathsf{HaltingObstruction}^{\uparrow}$ supplies finite obstruction rows, and $\mathsf{BoundedUniversalTrace}^{\uparrow}$ supplies bounded trace rows. None supplies both axis rows and the quadrant tag as one finite packet. □

The later Lean TasteGate namespace BEDC.Derived.ClosureUniversalityQuadrantUp will provide Nontrivial ClosureUniversalityQuadrant$^{\uparrow}$ and FieldFaithful ClosureUniversalityQuadrant$^{\uparrow}$ instances for the displayed carrier.

Closure status for NameCert$_{\text{ClosureUniversalityQuadrant}^{\uparrow}}$ ·

Constructive story: A ClosureUniversalityQuadrant$^{\uparrow}$ packet is generated from a universality-axis row, a closure-axis row, their quadrant tag, a substrate witness, sibling cellular anchors, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosureUniversalityQuadrant}^{\uparrow}}$, seed)

Scope closed: Definition 946.1, Definition 946.2, and Definition 946.3 specify the finite two-axis quadrant packet and its non-collapse boundary.

Formal verification: FormalStatus(NameCert$_{\text{ClosureUniversalityQuadrant}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, arbitrary universality, arbitrary closure, host substrate classification, or completeness of all possible quadrant inhabitants.

Upgrade path: Obligation closure requires checked carrier admission, axis-row accountability, quadrant-tag determinacy, sibling-anchor replay, nontriviality, and field faithfulness for the same packet.

Chapter 947

A Concrete Naming Certificate for $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$

The $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ packet realizes the two-axis map of chapter 9388 as a finite $\mathsf{NameCert}$ carrier. It sits beside the Rule 110 substrate surface of chapter 934, the trust packet of chapter 937, and the orbit packet of chapter 938. Its distinguishing datum is not another cellular run; it is the displayed pair of axis certificates and the quadrant provenance row that records how a substrate is classified without identifying the axes.

Falsifiable prediction: Deleting either the universality row or the closure row collapses the quadrant classifier and prevents distinguishing Rule110-like closed-universal substrates from oracle-open substrates.

Independence witness: Nearest siblings: cellularautomaton, cellulartrustsubstrate, cellularorbitsubstrate, largemodelinscriptionpoint, and kernelobservationsieve; none is bijective because this carrier stores two independent axis certificates and quadrant provenance rather than a single substrate, orbit, inscription, or sieve row.

Definition 947.1 (Universality-closure quadrant carrier)**.** *A* $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, U, K, G, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the substrate-name row, naming the cellular, machine, oracle, finite-state, or host-side substrate whose location is being certified;*
2. *U is the universality-axis row, recording the displayed value* $\mathsf{Univ}(S)$ *together with the finite witness or refusal ledger used by the classifier;*
3. *K is the closure-axis row, recording the displayed value* $\mathsf{Closed}(S)$ *together with the finite observation-channel witness or refusal ledger;*
4. *G is the quadrant row, one of universal-and-closed, universal-and-open, subuniversal-and-closed, or subuniversal-and-open;*
5. *W is the provenance witness row linking S, U, K, G to the relevant substrate surface, such as the Rule 110 row, oracle row, Presburger-like closed row, or external-sensor row of the quadrant map;*
6. *H records componentwise* hsame *transports for S, U, K, G, W, P, N;*

7. *C records displayed* Cont *routes that replay the two independent axis reads and the quadrant selection;*
8. *P is the* Pkg *provenance over* $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$*, chapter 9388, the sibling substrate certificates,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniversalityClosureQuadrant}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the substrate name, universality row, closure row, quadrant row, witness row, transport rows, replay routes, provenance, and naming row. It has no coordinate for deriving closure from universality, deriving universality from closure, importing an oracle into a closed packet, or replacing a displayed axis value by a host-level semantic judgement.

Lean4 ✓ `BEDC.Derived.UniversalityClosureQuadrantUp.UniversalityClosureQuadrantTasteGate single carrier alignment`

Theorem 947.2 (Universality-closure quadrant NameCert obligations)**.** *For the carrier of Definition 947.1, the local* $\mathsf{NameCert}_{\mathsf{UniversalityClosureQuadrant}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier exposure for $Q = (S, U, K, G, W, H, C, P, N)$;*
2. *universality-axis accountability through U and the substrate row S;*
3. *closure-axis accountability through K and the observation-channel witness row W;*
4. *quadrant selection through the ordered pair (U, K) and no other classifier coordinate;*
5. *componentwise stability through H and displayed replay through C;*
6. *non-escape through P and N, excluding axis collapse, oracle import into closed rows, hidden host semantics, and quotient-level substrate equality.*

No consumer can cite the quadrant map as $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ *evidence unless both axis rows and the quadrant provenance row are present in the same finite packet.*

Proof. Carrier exposure is projection from Definition 947.1. The packet displays the substrate row, two axis rows, quadrant row, witness row, transports, replay routes, provenance, and local naming row as its only coordinates.

Universality and closure accountability are separate reads. The row U records the computational-power classifier and K records the observation-channel classifier; the route C may replay each row, but it does not identify them or turn one into evidence for the other.

Quadrant selection is therefore a finite product read. The row G is accepted only as the classifier value determined by the displayed pair (U, K) and the witness row W, so a consumer cannot replace the two-axis packet by a single substrate, orbit, inscription, or sieve row.

Finally, H, P, N transport, package, and name the same finite packet. Since the carrier has no coordinate for an oracle inside a closed row, hidden semantic equality, quotient substrate equality, or a derivation between the axes, the listed obligation rows exhaust the local NameCert surface. □

Theorem 947.3 (Universality-closure quadrant axis independence)**.** *In an accepted* $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ *packet, neither the universality-axis row U nor the closure-axis row K determines the other. The quadrant row G is determined only by the displayed pair (U, K) together with the provenance witness W. Thus the closed-universal Rule 110-style site, the universal-open oracle site, the subuniversal-closed host or Presburger-style site, and the subuniversal-open external-sensor site remain four distinct classifier outcomes.*

Lean4 ✓ `BEDC.Derived.UniversalityClosureQuadrantUp.UniversalityClosureQuadrant axis independence`

Proof. Project an accepted packet to S, U, K, G, W. The carrier has one row for universality and one row for closure, and the two rows are listed as separate coordinates.

The replay routes in C preserve that separation. A universality read may inspect the computational-power witness carried by U, while a closure read may inspect the observation-channel witness carried by K and W; neither route rewrites one row into the other.

The quadrant classifier then reads the ordered pair. Changing either coordinate changes the admissible quadrant value: Rule 110-like closed universality, oracle-open universality, subuniversal closure, and subuniversal external openness occupy different cells in chapter 9388.

Since no carrier coordinate identifies the axes, any collapse would require a row absent from Q. The accepted packet therefore certifies axis independence at the NameCert surface. □

Theorem 947.4 (Universality-closure quadrant non-escape)**.** *Every public read exported by an accepted* $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ *packet factors through the finite route*

$$S, U, K, G, W, H, C, P, N.$$

The route can certify the two-axis quadrant location of a displayed substrate, but it exports no proof of Rule 110 universality, no oracle computation, no universal closure over an infinite domain, no quotient equality of substrates, no host semantic equality, and no derivation from one axis to the other.

Proof. Begin with carrier projection. The public packet contains only the substrate row S, universality row U, closure row K, quadrant row G, witness row W, transports H, replay routes C, provenance P, and local name N.

The axis-independence theorem forces every quadrant read through the pair (U, K). The witness row may point to a cellular substrate, cellular trust packet, cellular orbit packet, oracle-open comparison, host-side closed substrate, or external-sensor comparison, but that pointer is provenance for a row already displayed in the packet.

The excluded objects require coordinates not present in Q. Rule 110 universality needs the separate substrate proof, oracle computation needs an oracle row, universal closure needs an infinite-domain ratifier, quotient equality needs a quotient classifier, host semantic equality needs a host semantic row, and axis collapse needs an identification between U and K. None is exported by the finite quadrant certificate. □

The later Lean `TasteGate.lean` development for $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ must include explicit $\mathsf{Nontrivial\ UniversalityClosureQuadrant}^{\uparrow}$ and $\mathsf{FieldFaithful\ UniversalityClosureQuadrant}^{\uparrow}$ instances for the displayed two-axis carrier. These instances must preserve the substrate row, universality row, closure row, quadrant row, provenance witness, transport rows, replay routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{UniversalityClosureQuadrant}^{\uparrow}}$.

Constructive story: A $\mathsf{UniversalityClosureQuadrant}^{\uparrow}$ packet is generated from a substrate row, universality-axis row, closure-axis row, quadrant row, provenance witness, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniversalityClosureQuadrant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 947.1, Definition 947.2, Definition 947.3, and Definition 947.4 specify the finite two-axis substrate quadrant certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniversalityClosureQuadrant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, Rule 110 universality, oracle computation, universal closure over an infinite domain, quotient equality, host semantic equality, or derivability between the axes.

Upgrade path: Obligation closure requires checked carrier admission, universality-axis accountability, closure-axis accountability, quadrant selection, axis independence, non-escape, nontriviality, and field faithfulness for the same two-axis carrier.

Chapter 948

A Concrete Naming Certificate for $\mathsf{BHistCellRowEmbedding}^{\uparrow}$

The $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ packet concretizes the carrier-equivalence reading of chapter 9391 as a finite naming certificate from a BHist row to a cellular row observation. It sits beside the closed cellular trust packet of chapter 940, the finite orbit packet of chapter 938, and the public record packet of chapter 519. Those siblings certify cellular trust, orbit observation, and inscription exposure; this carrier certifies the explicit row-level bridge that keeps the original BHist source and its cellular readback in one packet.

Its composite dependencies are the BHist source row, the $\mathsf{CellularAutomaton}^{\uparrow}$ row substrate, the $\mathsf{CellularTrustSubstrate}^{\uparrow}$ trust packet of chapter 940, the $\mathsf{CellularOrbitSubstrate}^{\uparrow}$ orbit observation of chapter 938, and the $\mathsf{InscriptionEvent}^{\uparrow}$ exposure packet of chapter 519, combined with the inverse readback row into one row-embedding certificate.

Falsifiable prediction: Removing the inverse readback row leaves a cellular row observation but breaks the BHist carrier recovery route.

Independence witness: Nearest siblings: cellulartrustsubstrate, cellularorbitsubstrate, inscriptionevent, and cellularautomaton. None is bijective to $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ because none stores both the source BHist row and the explicit bit-row embedding together with the inverse readback row that recovers the same carrier source.

Definition 948.1 (BHist cell-row embedding carrier)**.** *A* $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, I, W, O, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* BHist *row whose constructor trace is embedded.*
2. *I is the bit-row image obtained from the displayed constructor trace of B.*
3. *W is the width and default-padding row, naming the finite row width, padding value, origin coordinate, and the bounded slice in which I is placed.*
4. *O is the orbit-zero row, recording the cellular row observation before any cellular transition is consumed.*
5. *R is the inverse readback row, recovering the same source* BHist *constructor trace from the displayed bit-row image and the padding convention.*

6. *H records componentwise* hsame *transports for B, I, W, O, R, P, N.*
7. *C records displayed* Cont *routes through source reading, bit-row embedding, orbit-zero observation, inverse readback, provenance, and local naming.*
8. *P is the* Pkg *provenance over* BHist, CellularAutomaton$^{\uparrow}$, CellularTrustSubstrate$^{\uparrow}$, CellularOrbitSubstrate$^{\uparrow}$, InscriptionEvent$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BHistCellRowEmbedding}^{\uparrow}}$ *row for carrier admission, row embedding exactness, inverse readback exactness, non-escape, and TasteGate boundary exposure.*

The carrier contains no host byte string, no unbounded cellular orbit, no detached cellular evaluator, no quotient of encodings, and no cellular transition claim beyond the displayed orbit-zero row.

Lean4 (definition) `BEDC.Derived.BHistCellRowEmbeddingUp.BHistCellRowEmbeddingUp`

Theorem 948.2 (BHist cell-row embedding NameCert obligations)**.** *For the carrier of Definition 948.1, the local* $\mathsf{NameCert}_{\mathsf{BHistCellRowEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for $E = (B, I, W, O, R, H, C, P, N)$, bit-row image accountability through I, width and default-padding accountability through W, orbit-zero observation through O, inverse readback exactness through R, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N. No cellular row observation is accepted as a* BHist *carrier bridge unless it passes through those rows.*

Lean4 ✓ `BEDC.Derived.BHistCellRowEmbeddingUp.BHistCellRowEmbeddingCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 948.1. The packet displays the source BHist row, bit-row image, width and padding row, orbit-zero row, inverse readback row, transports, routes, provenance, and local name as its only coordinates.

The embedding rows I and W account for the cellular row image. The row I records the finite binary image of the constructor trace, while W records where the image sits and how every unmentioned cell is padded.

The readback row R is the carrier-recovery check. It ties the displayed cellular row back to the same source trace B, so the observation cannot be replaced by a host byte layout or by another row with the same visible finite slice but a different carrier source.

The rows H, C, P, N repack those data as transport, replay, provenance, and naming evidence. Since the packet has no coordinate for an unbounded orbit, detached evaluator, quotient encoding, or cellular transition claim, the listed rows exhaust the local NameCert surface. □

Theorem 948.3 (BHist cell-row embedding non-escape)**.** *Every accepted* $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *read factors through the finite route*

$$B \longmapsto I \longmapsto W \longmapsto O \longmapsto R \longmapsto H, C, P, N.$$

The route exports the source BHist *row, its bit-row image, the width and default-padding convention, the orbit-zero row observation, the inverse readback row, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and the local* NameCert *row. It exports no host byte string, no omitted padding convention, no unbounded cellular orbit, no hidden substrate evaluator, no quotient of encodings, and no selected cellular transition history.*

Lean4 ✓ `BEDC.Derived.BHistCellRowEmbeddingUp.BHistCellRowEmbedding nonescape`

Proof. Start with the accepted carrier and apply the NameCert obligations of Definition 948.2. Every public read reaches the local name only through $B, I, W, O, R, H, C, P, N$.

The row observation itself is bounded by I, W, O. These rows name the finite bit image, its width and padding convention, and the cellular row at orbit zero. They do not supply later cellular dynamics or a host evaluator.

The inverse readback row R closes the recovery route to B. Once the readback has been fixed, no consumer can substitute a different source trace, host byte layout, or quotient representative without leaving the displayed packet.

The remaining rows are transport and provenance rows. They move and replay the same finite data but introduce no further object, so the excluded outputs have no coordinate from which to escape. □

Theorem 948.4 (BHist cell-row embedding TasteGate boundary)**.** *The TasteGate-facing boundary of an accepted* $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *carrier is exactly the finite witness-pair surface consisting of the source row B and the readback row R, together with field projections to I, W, O, H, C, P, N. A later Lean implementation of* BEDC.Derived.BHistCellRowEmbeddingUp.TasteGate *must provide* Nontrivial $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *and* FieldFaithful $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *instances by giving explicit witness-pair and field projection proofs for those displayed rows.*

Lean4 ✓ `BEDC.Derived.BHistCellRowEmbeddingUp.BHistCellRowEmbedding tastegate boundary`

Proof. Project the carrier. The witness pair is available because the packet contains both the source BHist row B and the inverse readback row R in the same finite certificate.

Use the NameCert obligation theorem to read every other field as a displayed coordinate of the same packet. The bit-row image, width and padding row, orbit-zero row, transports, continuations, provenance, and local name are therefore field projections rather than hidden predicates.

Apply non-escape. Since the boundary exports no host byte string, quotient encoding, unbounded orbit, hidden evaluator, or selected transition history, the TasteGate implementation must be faithful to these finite fields and cannot discharge faithfulness by appealing to a non-packet cellular object. □

Closure status for $\mathsf{NameCert}_{\mathsf{BHistCellRowEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ packet is generated from a source BHist row, a bit-row image, width and default-padding data, an orbit-zero cellular row observation, an inverse readback row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistCellRowEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 948.1, Definition 948.2, Definition 948.3, and Definition 948.4 name the finite BHist-to-cell-row bridge surface without closing its formal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistCellRowEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a kernel carrier switch, Rule 110 trust replay, unbounded cellular evolution, host byte equality, quotient encoding, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, inverse readback exactness, non-escape, and TasteGate field-faithfulness rows for the same BHist cell-row embedding packet.

Chapter 949

A Concrete Naming Certificate for $\mathsf{CookFrontierCoordinate}^{\uparrow}$

The $\mathsf{CookFrontierCoordinate}^{\uparrow}$ packet turns the engineering path of chapter 9417 into a finite $\mathsf{NameCert}$ surface. It reads the cellular rule-table and finite orbit rows of chapter 934, the fuel-indexed readback discipline of chapter 944, and the observation sieve of chapter 932; its own work is to bind a stage-frontier triple, Rule 110 scale row, audit replay row, and failure ledger into one coordinate packet. The carrier does not assert unbounded universality, does not decide that a halted target has been reached, and does not import a host evaluator oracle.

Falsifiable prediction: If a claimed frontier coordinate can be replayed without the displayed stage-frontier, Rule 110 scale, audit-binary, or failure-ledger rows, then the carrier admission row is false.

Independence witness: cellularautomaton, cellulartrustsubstrate, boundeduniversaltrace, kernelobservationsieve: these siblings expose cellular rows, trust routes, bounded traces, or observation sieves, but none is bijective to the coordinate packet because none carries the ordered stage-frontier triple with audit replay and failure-ledger rows.

Definition 949.1 (Cook frontier coordinate carrier)**.** *A* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *carrier is a finite packet*

$$K = (F, R, S, A, L, H, C, P, N)$$

with the following displayed rows:

1. $F = (F_{\mathrm{tag}}, F_{\mathrm{compile}}, F_{\mathrm{halt}})$ *is the ordered stage-frontier triple naming the tag-system encoding frontier, the Rule 110 compilation frontier, and the halted-target frontier inspected by the packet;*
2. R *is the finite Rule 110 scale row naming the cellular rule table, width budget, time budget, boundary convention, and row family used by the frontier read;*
3. S *is the finite stage-sample ledger recording the tag-system word, compiled cellular window, and target readback rows displayed at the chosen frontier;*
4. A *is the audit-binary row containing the replay executable name, input manifest, digest row, accepted output rows, and refusal rows for failed replay;*
5. L *is the failure-ledger row recording absent encodings, scale overflows, replay mismatches, unread target rows, and refused universality reads;*
6. H *records componentwise* hsame *transports for* F, R, S, A, L, P, N*;*

7. *C records displayed* Cont *routes from the stage-frontier triple through the scale row, sample ledger, audit replay, and failure ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{CookFrontierCoordinate}^{\uparrow}$*, chapter 9417,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* BoundedUniversalTrace BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$ *row.*

The local classifier compares accepted coordinates only by these displayed rows. It has no coordinate for unbounded Rule 110 universality, a host-level proof that the target Turing machine halts, a hidden cellular evaluator, quotient stream equality, or a detached success claim outside the audit binary.

Lean4 (definition) `BEDC.Derived.CookFrontierCoordinateUp.TasteGate.CookFrontierCoordinateUp`

Theorem 949.2 (Cook frontier coordinate NameCert obligations)**.** *For the carrier of Definition 949.1, the local* $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (F, R, S, A, L, H, C, P, N)$, ordered stage-frontier accountability through F, Rule 110 scale accountability through R, finite sample exposure through S, audit replay exactness through A, failure-ledger non-escape through L, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 949.1. The packet displays the stage-frontier triple, scale row, sample ledger, audit replay, failure ledger, transports, routes, provenance, and local naming row as its only coordinates.

Stage-frontier and scale accountability are read from F and R. A consumer must name which encoding, compilation, and halted-target frontiers are being inspected, and must read the Rule 110 width, time, boundary, and row-family data from the finite scale row.

Sample exposure and audit replay are read from S and A. The sample ledger records the tag-system, cellular, and target readback rows selected by the coordinate, while the audit-binary row supplies the replay manifest, digest, accepted outputs, and refused replay cases.

Finally use L, H, C, P, N. The failure ledger blocks success claims that bypass the replay rows, and the structural rows transport and package the same finite fields. Since the carrier has no unbounded universality, host halting, hidden evaluator, quotient stream, or detached success coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CookFrontierCoordinateUp.CookFrontierCoordinate namecert obligations`

Theorem 949.3 (Cook frontier coordinate witnessed-universality boundary)**.** *An accepted* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *packet may report witnessed operational coverage only along the finite route*

$$F \longmapsto R \longmapsto S \longmapsto A \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

It may cite the Cook compile-frontier path of chapter 9417 as the vision being measured and may use the cellular and bounded-trace surfaces of chapter 934 and chapter 944 as sibling inputs, but it does not promote finite replay into an unbounded Rule 110 universality theorem.

Lean4 ✓ `BEDC.Derived.CookFrontierCoordinateUp.CookFrontierCoordinate witnessed universality boundary`

Proof. Start with the route inventory. The frontier triple F names the three inspected stages, while R and S provide only a finite Rule 110 scale row and a finite stage-sample ledger.

The sibling chapters supply finite surfaces rather than unbounded semantic closure. The cellular chapter contributes displayed rule-table and orbit rows, and the bounded-trace chapter contributes fuel-indexed readback; neither gives this packet a host proof that every future Rule 110 row or every target Turing-machine behaviour has been covered.

The audit and failure rows determine the witnessed boundary. A replay accepted by A is reported only with the refusals and missing cases recorded in L, then transported and packaged through H, C, P, N. Thus the coordinate can witness finite operational coverage without asserting unbounded universality as a new theorem. □

Theorem 949.4 (Cook frontier coordinate failure-ledger non-escape)**.** *Every rejected or incomplete* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *read remains inside the displayed failure ledger* L*. A missing encoding row, scale overflow, audit mismatch, unread target row, or refused universality read cannot be reclassified as success by appealing to* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*, a host evaluator, quotient stream equality, or any route absent from* $K = (F, R, S, A, L, H, C, P, N)$*.*

Proof. Project the accepted carrier. The only rows that can record failure are the replay row A and the failure ledger L, with F, R, S providing the stage, scale, and sample context.

Inspect each failure class through the same finite coordinates. Missing encodings are failures of the stage-frontier row, scale overflows are failures of the Rule 110 scale row, replay mismatches are failures of the audit binary, and unread target rows or refused universality reads are ledger entries in L.

The sibling chapters do not supply a bypass. Cellular rows and bounded traces can be cited only as displayed finite inputs to this coordinate, and no host evaluator, quotient stream, or absent route is carried by K. Therefore every rejected or incomplete read stays recorded as a failure-ledger row. □

The Lean follow-up for $\mathsf{CookFrontierCoordinate}^{\uparrow}$ requires `Nontrivial` plus `FieldFaithful` instances in `lean4/BEDC/Derived/CookFrontierCoordinateUp/TasteGate.lean`, with witnesses for the ordered stage-frontier triple, Rule 110 scale row, stage-sample ledger, audit-binary row, failure ledger, transports, routes, provenance, and local naming field.

Theorem 949.5 (Cook frontier coordinate scale-row faithfulness)**.** *For every accepted* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *carrier*

$$K = (F, R, S, A, L, H, C, P, N),$$

the scale row R *is the unique row from which the Rule 110 rule table, width budget, time budget, boundary convention, and row family of the frontier read may be projected. A consumer that changes any one of those five scale fields while preserving* F, S, A, L, H, C, P, N *no longer reads the same* $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$ *coordinate.*

Proof. Project the carrier of Definition 949.1. The row R displays exactly the Rule 110 rule table, width budget, time budget, boundary convention, and row family used by the coordinate.

The stage-frontier triple F names where the packet is read, but it does not store the cellular scale. The sample ledger S records selected rows after the scale has already been fixed.

The audit row A and failure ledger L replay or refuse a finite read against the displayed scale; they do not define a second width, time, boundary, or row-family source. Transport, replay, provenance, and naming rows H, C, P, N preserve the same displayed field order.

Thus any alteration of the Rule 110 scale data changes the only row that can justify those projections. The accepted coordinate is therefore faithful to R as the scale row. □

Theorem 949.6 (Cook frontier coordinate audit replay exactness)**.** *For every accepted* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *carrier* $K = (F, R, S, A, L, H, C, P, N)$*, the audit-binary row* A *and failure-ledger row* L *determine the replay boundary exactly: accepted output rows are those named by* A*, and absent encodings, scale overflows, replay mismatches, unread target rows, and refused universality reads remain ledger entries in* L*.*

Proof. Read A first. It carries the replay executable name, input manifest, digest row, accepted output rows, and refusal rows for failed replay.

Read L next. The ledger names the failure classes attached to the same frontier and scale rows: absent encodings, scale overflows, replay mismatches, unread target rows, and refused universality reads.

The sample row S supplies observed data, but it does not convert a ledger entry into an accepted output. Likewise H, C, P, N transport, replay, package, and name the same audit boundary without adding a hidden evaluator.

Therefore replay exactness is the joint reading of A and L. A successful replay must be an accepted output row in A, and every failed or incomplete replay remains visible as a row of L. □

Theorem 949.7 (Cook frontier coordinate field faithfulness)**.** *The fields* $F, R, S, A, L, H, C, P, N$ *of a* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *carrier cannot be merged while preserving the displayed* $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$ *rows. Merging any two of the ordered stage-frontier triple, Rule 110 scale row, stage-sample ledger, audit-binary row, failure ledger, componentwise transport, displayed replay, package provenance, or local naming row loses at least one carrier admission, accountability, replay, non-escape, stability, provenance, or naming obligation.*

Proof. Inspect the roles of the nine fields. The row F names the frontier stages, R names the Rule 110 scale, and S names the sampled stage data.

The rows A and L separate replay success from replay failure. If they are merged, the packet can no longer distinguish accepted output rows from refused or incomplete reads.

The structural rows H and C have different duties: one transports componentwise equality data and the other records displayed route replay. The package row P and local naming row N then record provenance and certificate naming rather than frontier, scale, sample, audit, or failure data.

Each field therefore discharges a distinct NameCert obligation listed in Definition 949.2. A merge removes the witness for at least one listed obligation, so field faithfulness is forced by the displayed carrier itself. □

Theorem 949.8 (Cook frontier coordinate scoped kernel route)**.** *Every accepted* $\mathsf{CookFrontierCoordinate}^{\uparrow}$ *coordinate read factors through the finite route*

$$F \longmapsto R \longmapsto S \longmapsto A \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Here F *is the ordered stage-frontier triple,* R *is the Rule 110 scale row,* S *is the stage-sample ledger,* A *is the audit-binary row,* L *is the failure ledger, and* H, C, P, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$ *rows. The only sibling inputs consumed by the route are the displayed finite rows of* $\mathsf{CellularAutomaton}^{\uparrow}$ *and* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*; the route does not export unbounded Rule 110 universality, target-machine halting, host evaluator correctness, quotient stream equality, or a detached success claim.*

Proof. Project the carrier of Definition 949.1. Its coordinates are exactly the stage-frontier triple, scale row, sample ledger, audit replay row, failure ledger, transports, replay route, provenance, and local naming row.

Apply the NameCert obligations of Definition 949.2. They bind carrier admission, stage-frontier accountability, scale accountability, sample exposure, audit replay, failure-ledger non-escape, stability, replay, provenance, and local naming to those same rows.

Use the witnessed-universality boundary and failure-ledger non-escape theorems, Definition 949.3 and Definition 949.4. The sibling $\mathsf{CellularAutomaton}^{\uparrow}$ and $\mathsf{BoundedUniversalTrace}^{\uparrow}$ rows are consumed only as finite displayed inputs, while every failed or incomplete read stays in L.

Finally apply scale-row faithfulness, audit replay exactness, and field faithfulness. These prevent replacement of R, A, L, or any structural row by a hidden evaluator, an unbounded universality principle, or a merged carrier field. Thus the scoped kernel route is precisely the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}$.

Constructive story: A $\mathsf{CookFrontierCoordinate}^{\uparrow}$ packet is generated from an ordered stage-frontier triple, finite Rule 110 scale row, stage-sample ledger, audit-binary replay row, failure ledger, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 949.1, Definition 949.2, Definition 949.3, Definition 949.4, Definition 949.5, Definition 949.6, Definition 949.7, and Definition 949.8 specify the finite Cook compile-frontier coordinate packet, NameCert obligations, witnessed-universality boundary, failure-ledger non-escape, scale-row faithfulness, audit replay exactness, field faithfulness, and scoped kernel route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CookFrontierCoordinate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CookFrontierCoordinateUp.CookFrontierCoordinate\_namecert\_obligations`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove unbounded Rule 110 universality, target-machine halting, host evaluator correctness, quotient stream equality, or bridge checking for $\mathsf{CookFrontierCoordinate}^{\uparrow}$.

Upgrade path: Public closure requires a public-object surface for the coordinate route and a bridge-checked schema handoff for the same finite coordinate.

Chapter 950

A Concrete Naming Certificate for $\mathsf{QuadrantSubstrate}^{\uparrow}$

The $\mathsf{QuadrantSubstrate}^{\uparrow}$ packet realizes the two-axis quadrant map of chapter 9388 as a finite substrate-classifier certificate. It sits beside the cellular automaton substrate of chapter 934, the trust packet of chapter 937, the orbit packet of chapter 938, and the obstruction ledger of chapter 490. Its public role is the finite locator that records which quadrant a displayed substrate inhabits without turning that locator into host substrate equality.

Falsifiable prediction: A substrate read that changes either the universality-axis row or the closure-axis row without changing the quadrant locator is outside the certificate.

Independence witness: Nearest siblings: cellularautomaton, cellulartrustsubstrate, cellularorbitsubstrate, and haltingobstruction; none is bijective because those siblings carry a single substrate or obstruction surface while $\mathsf{QuadrantSubstrate}^{\uparrow}$ carries a two-axis locator plus contrast ledger.

Definition 950.1 (QuadrantSubstrate NameCert carrier)**.** *A* $\mathsf{QuadrantSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, K, L, S, C, H, R, P, N)$$

with the following displayed rows:

1. *U is the universality-axis row, recording whether the substrate read is universal or subuniversal in the sense of Definition 9388.1;*
2. *K is the closure-axis row, recording whether the same substrate read is closed or open in the sense of Definition 9388.2;*
3. *L is the quadrant locator, one of universal-and-closed, universal-and-open, subuniversal-and-closed, or subuniversal-and-open, read from the ordered pair (U, K);*
4. *S is the substrate witness whose local evidence supplies the row inspected by the locator;*
5. *C is the contrast ledger separating the four displayed cells and recording which adjacent cell would be reached by changing exactly one axis row;*
6. *H records componentwise* hsame *transport for U, K, L, S, C, R, P, N;*
7. *R records the displayed* Cont *routes that replay the two axis reads, the locator read, and the contrast-ledger read;*

8. *P is the* Pkg *provenance over* $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{CellularOrbitSubstrate}^{\uparrow}$, $\mathsf{HaltingObstruction}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuadrantSubstrate}^{\uparrow}}$ *row.*

The carrier has no coordinate for deriving U from K, deriving K from U, replacing L by host equality of substrates, or hiding an open observation channel behind a closed-row name.

Lean4 ✓ `BEDC.Derived.QuadrantSubstrateUp.QuadrantSubstrateTasteGate single carrier alignment`

Theorem 950.2 (QuadrantSubstrate axis classifier totality)**.** *Every accepted* $\mathsf{QuadrantSubstrate}^{\uparrow}$ *packet lands in exactly one displayed quadrant by the pair (U, K). The four cases are universal-and-closed, universal-and-open, subuniversal-and-closed, and subuniversal-and-open; no case is selected by host substrate equality or by identifying the substrate witness S with a sibling carrier.*

Proof. Inspect the two axis rows displayed in Definition 950.1. The row U has the two finite readings universal and subuniversal, and the row K has the two finite readings closed and open.

The quadrant locator L is defined as the product read of those two rows. Thus the finite inspection has four possible outcomes: $(\mathsf{universal}, \mathsf{closed})$, $(\mathsf{universal}, \mathsf{open})$, $(\mathsf{subuniversal}, \mathsf{closed})$, and $(\mathsf{subuniversal}, \mathsf{open})$.

Exactness follows because the contrast ledger C records the adjacent-cell changes and the route row R replays the same two axis reads. Changing either axis changes the admissible locator, while leaving both axes fixed leaves one displayed cell.

No host substrate equality is consulted in this inspection. The witness row S supplies the substrate evidence, but the accepted cell is selected by (U, K) and transported componentwise by H, not by a global equality between $\mathsf{QuadrantSubstrate}^{\uparrow}$ and any sibling certificate. □

Theorem 950.3 (QuadrantSubstrate NameCert obligations)**.** *For the carrier of Definition 950.1, the local* $\mathsf{NameCert}_{\mathsf{QuadrantSubstrate}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier exposure for $Q = (U, K, L, S, C, H, R, P, N)$;*
2. *axis classifier totality through the finite pair (U, K);*
3. *quadrant locator exactness through L and the displayed replay routes R;*
4. *contrast-ledger non-escape through C, excluding axis collapse and hidden substrate equality;*
5. *local provenance through P and N over the listed sibling surfaces and BEDC kernel rows.*

These rows are sufficient to cite the two-axis quadrant classifier as a $\mathsf{QuadrantSubstrate}^{\uparrow}$ *NameCert packet, and no additional substrate-oracle, halting decision, infinite closure, or quotient equality row is exported.*

Proof. Carrier exposure is direct projection from the packet coordinates. The finite rows $U, K, L, S, C, H, R, P, N$ are the entire displayed surface, so every accepted public read begins by naming one of those coordinates.

Axis classifier totality is Definition 950.2. The local certificate therefore has a total four-cell locator for every accepted packet, with the value fixed by (U, K).

Locator exactness is supplied by L and replayed by R. The route row checks that the accepted locator is the same finite product read of the universality row and the closure row, rather than a name imported from the witness substrate.

The contrast ledger gives non-escape. Since C records the one-axis moves between adjacent cells, a consumer cannot change an axis row while keeping the old locator, collapse the two axes, or replace the finite row comparison by host substrate equality.

Finally, P and N bind the local certificate to the named sibling surfaces and to BHist, hsame, Cont, Pkg, and NameCert. The packet contains no row for an oracle computation, no global halting classifier, no infinite closure ratifier, and no quotient equality of substrate names, so the five listed rows exhaust the local NameCert obligations. □

Lean `BEDC/Derived/QuadrantSubstrateUp/TasteGate.lean` must provide Nontrivial QuadrantSubstrate$^{\uparrow}$ and FieldFaithful QuadrantSubstrate$^{\uparrow}$ instances with explicit `witness_pair` and constructor-field injection proofs for the two-axis locator carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{QuadrantSubstrate}^{\uparrow}}$.

Constructive story: A QuadrantSubstrate$^{\uparrow}$ packet is generated from two finite axis rows: the universality row and the closure row. The BHist carrier adds the product locator, substrate witness, contrast ledger, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert row. The classifier is bottom-up: first read the two axes, then compute the one displayed quadrant, then use the contrast ledger to expose how changing one axis changes the cell. Downstream consumers receive the locator together with its route evidence. The route names a finite locator, not a global substrate identity.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuadrantSubstrate}^{\uparrow}}$, seed)

Scope closed: This chapter binds the quadrant substrate carrier, axis classifier totality, and the five finite NameCert obligations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{QuadrantSubstrate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not derive one axis from the other, assert host substrate equality, supply halting oracles, or close an infinite substrate class.

Upgrade path: To reach obligationClosure, give a checked carrier whose axis-pair classifier and contrast ledger determine exactly one displayed quadrant.

Chapter 951

A Concrete Naming Certificate for $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$

The $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ packet realizes the universal-but-not-closed quadrant of chapter 9388 as a finite NameCert surface. It sits opposite the closed cellular substrate witnesses of chapter 934 and beside the obstruction-facing boundary of chapter 490. Its distinguishing datum is not another closed orbit ledger. It is the explicit oracle-call row and transcript row by which a universal substrate leaves the closed observation channel.

Falsifiable prediction: Removing the oracle-call row makes an oracle transcript indistinguishable from a closed cellular orbit ledger; removing the transcript row prevents any consumer from testing whether the substrate is open rather than closed.

Independence witness: cellularautomaton, cellulartrustsubstrate, haltingobstruction, haltingdiagonal, and haltingdistinctionlimit: none is bijective to $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ because this packet carries oracle-call, oracle-transcript, open-boundary, and quadrant-provenance rows in one finite substrate certificate, while the listed siblings supply closed cellular rules, trust rows, obstruction rows, diagonal rows, or distinction-limit rows without the oracle-open carrier surface.

Definition 951.1 (Oracle-augmented substrate carrier)**.** *An* $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, A, T, B, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the universal substrate row, naming the internal finite rule, machine, or simulation surface whose ordinary steps are read as substrate-local records.*
2. *A is the oracle-call row, recording the finite query shape, query name, and call boundary by which the substrate asks a source not generated by S.*
3. *T is the oracle-transcript row, recording the finite reply ledger used by the consumer, including the reply token and the local point at which it is consumed.*
4. *B is the open-boundary row, recording that A and T are not closed orbit records of S and cannot be replayed as a cellular ledger.*
5. *E is the quadrant-evidence row, recording universality from the substrate row and non-closure from the oracle-call and transcript boundary.*
6. *H records componentwise* hsame *transports for S, A, T, B, E, P, N.*

7. *C records displayed* Cont *routes through substrate-local steps, oracle calls, transcript reads, the open-boundary check, and the final quadrant read.*
8. *P is the* Pkg *provenance over* $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$*, the quadrant map, the cellular and halting siblings,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OracleAugmentedSubstrate}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through these rows. It has no coordinate for converting the oracle transcript into a closed cellular orbit, no hidden halting oracle outside the displayed call row, no ambient proof of hypercomputation, and no closure certificate for the open boundary.

Theorem 951.2 (Oracle-augmented substrate NameCert obligations)**.** *For the carrier of Definition 951.1, the local* $\mathsf{NameCert}_{\mathsf{OracleAugmentedSubstrate}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier exposure for $O = (S, A, T, B, E, H, C, P, N)$;*
2. *oracle-call accountability, requiring every oracle read to cite A before a reply can be consumed;*
3. *transcript accountability, requiring every reply token to cite T and its consumption point;*
4. *open-boundary exactness, requiring B to separate oracle-call and transcript rows from closed orbit-ledger rows;*
5. *non-escape through E, H, C, P, N, excluding hidden oracles, closed cellular replay of the transcript, ambient hypercomputation proofs, and any quadrant read detached from the displayed oracle rows.*

Thus a consumer may classify the packet as universal but not closed only after reading the substrate row, oracle-call row, transcript row, open-boundary row, and package-local naming rows in the same finite certificate.

Lean4 ✓ `BEDC.Derived.OracleAugmentedSubstrateUp.OracleAugmentedSubstrateTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 951.1. The packet displays the universal substrate row, oracle-call row, transcript row, open-boundary row, quadrant-evidence row, transports, replay routes, provenance, and local name as its only public coordinates.

Oracle-call and transcript accountability are separate reads. The row A records the call boundary and query shape, while T records the reply ledger and consumption point. A consumer that has only one of these rows cannot reconstruct the other from the substrate-local row S.

Open-boundary exactness is the role of B. The cellular siblings provide closed rule and orbit ledgers, and the halting siblings provide obstruction and diagonal surfaces, but neither family supplies an oracle reply as a substrate-internal orbit record. The route C must therefore replay the open-boundary check before accepting the quadrant-evidence row E.

Finally H, P, N transport, package, and name the same finite route. Since hidden oracles, ambient hypercomputation proofs, and detached quadrant judgements have no carrier coordinate, the five obligation rows exhaust the local NameCert surface. □

Theorem 951.3 (Oracle-augmented substrate oracle non-escape)**.** *No public consumer of an accepted* $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ *packet can erase the oracle boundary. Every oracle-dependent read factors through the finite route*

$$S, A, T, B, E, H, C, P, N.$$

The route may certify that the displayed substrate belongs to the universal-but-not-closed quadrant, but it exports no closed cellular orbit ledger for the oracle reply, no substrate-internal proof of the oracle answer, no hidden halting oracle, and no closure premise for observations that pass through A and T.

Proof. Start with carrier projection. The only public coordinates are $S, A, T, B, E, H, C, P, N$, so any oracle-dependent read must cite the oracle-call row and transcript row before it can reach the quadrant-evidence row.

The open-boundary row blocks closed replay. A closed cellular orbit ledger, such as the sibling cellularautomaton and cellulartrustsubstrate surfaces, records substrate-local evolution. The oracle transcript is admitted precisely as a reply not generated by that local evolution, so a consumer cannot identify it with a closed orbit row without leaving the packet.

The haltingobstruction, haltingdiagonal, and haltingdistinctionlimit siblings explain why the oracle row is not merely an obstruction label: the packet stores a call-and-transcript surface, not a proof that the substrate internally solves the obstruction. Therefore hidden oracle strength and substrate-internal oracle correctness remain outside the public projection.

The packaging rows H, C, P, N preserve that boundary under transport and replay. Hence the oracle-open classification is public only as the finite route through the displayed oracle rows, and no closed-observational premise is exported for those rows. □

The later Lean `TasteGate.lean` development for $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ must provide $\mathsf{Nontrivial\ OracleAugmentedSubstrate}^{\uparrow}$ and $\mathsf{FieldFaithful\ OracleAugmentedSubstrate}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{OracleAugmentedSubstrate}^{\uparrow}}$.

Constructive story: An $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$ packet is generated from a universal substrate row, oracle-call row, oracle-transcript row, open-boundary row, quadrant-evidence row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OracleAugmentedSubstrate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 951.1, Definition 951.2, and Definition 951.3 specify the finite oracle-open substrate certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OracleAugmentedSubstrate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, physical realizability of oracle substrates, oracle correctness, closed replay of oracle transcripts, or a complete theory of hypercomputation.

Upgrade path: Obligation closure requires checked carrier admission, oracle-call accountability, transcript accountability, open-boundary exactness, oracle non-escape, nontriviality, and field faithfulness for the same packet.

Chapter 952

A Concrete Naming Certificate for $\mathsf{CookCompileFrontierWitness}^{\uparrow}$

The $\mathsf{CookCompileFrontierWitness}^{\uparrow}$ surface realizes the engineering coordinate of chapter 9417 as a finite NameCert packet. It sits beside the cellular row certificate of chapter 934, the closed cellular trust packet of chapter 940, the executable certificate edge of chapter 473, the kernel-morphism graph surface of chapter 620, and the bounded trace packet of chapter 944. Those siblings certify local orbit, trust, compiler, morphism, and trace rows; this packet certifies the finite three-stage compile frontier together with the audit row that witnesses the coordinate.

Falsifiable prediction: A claimed frontier witness whose round-trip audit row omits one of the three compile stages is outside the certificate.

Independence witness: cellularautomaton, cellulartrustsubstrate, certificatecompiler, kernelmorphism, boundeduniversaltrace: none is bijective to $\mathsf{CookCompileFrontierWitness}^{\uparrow}$ because the witness carries a stage-coordinate row and a round-trip audit row whose obligations bind all three compile stages at once. CellularAutomaton and CellularTrustSubstrate expose cellular rows, CertificateCompiler and KernelMorphism expose certificate and graph edges, and BoundedUniversalTrace exposes bounded trace replay; none alone binds the Cook frontier coordinate to the three-stage audit surface.

Definition 952.1 (Cook compile-frontier witness carrier)**.** *A* $\mathsf{CookCompileFrontierWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, Q, A, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the compile-stage ledger with exactly the TM-to-tag, tag-to-cyclic, and cyclic-to-Rule 110 stage rows, in that order;*
2. *Q is the frontier-coordinate row recording the displayed source size, production count, tape-symbol count, and Rule 110 step budget read from the engineering coordinate;*
3. *A is the round-trip audit row, exposing the source endpoint, the three stage outputs, the Rule 110 observation window, and the decoded endpoint;*
4. *O is the obstruction row naming only finite memory-scale, encoding-constant, and verification-cost boundaries attached to the displayed coordinate;*
5. *H records componentwise* hsame *transports for S, Q, A, O, P, N;*

6. *C records displayed* Cont *routes through the three compile stages, the audit row, and the coordinate classifier;*
7. *P is the* Pkg *provenance over* CellularAutomaton$^{\uparrow}$, CellularTrustSubstrate$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, KernelMorphismUp, BoundedUniversalTrace$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CookCompileFrontierWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden compiler, an omitted stage, a host evaluator, an unbounded Rule 110 orbit, quotient equality of traces, or a detached universality oracle.

Lean4 (definition) `BEDC.Derived.CookCompileFrontierWitnessUp.CookCompileFrontierWitnessUp`
Lean4 ✓ `BEDC.Derived.CookCompileFrontierWitnessUp.CookCompileFrontierWitnessTasteGate single carrier alignment`

Theorem 952.2 (Cook compile-frontier witness obligation surface)**.** *For the carrier of Definition 952.1, the local* $\mathsf{NameCert}_{\mathsf{CookCompileFrontierWitness}^{\uparrow}}$ *surface consists exactly of carrier exposure for $W = (S, Q, A, O, H, C, P, N)$, compile-stage exposure through S, coordinate accountability through Q, round-trip audit replay through A, finite obstruction locality through O, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. No consumer can cite a Cook compile-frontier witness without exposing the three compile stages and the audit row that binds the coordinate.*

Lean4 ✓ `BEDC.Derived.CookCompileFrontierWitnessUp.CookCompileFrontierWitnessCarrier obligation surface`

Proof. Project the carrier of Definition 952.1. The only stage-facing data are the three ordered rows in S, and the only coordinate-facing data are Q and the audit row A.

The audit obligation is finite. The row A records the source endpoint, each stage output, the Rule 110 observation window, and the decoded endpoint, so a claimed witness cannot replace an exposed stage by an external compiler or evaluator.

The remaining obligations are the transport, route, provenance, and name rows. The rows H, C, P, N preserve and replay only the displayed packet over the named sibling certificates, so no hidden stage, unbounded orbit, quotient trace, or detached oracle remains available to a public consumer. □

Lean4 ✓ `BEDC.Derived.CookCompileFrontierWitnessUp.CookCompileFrontierWitnessObligationSurface`

Theorem 952.3 (Cook compile-frontier stage-boundary non-escape)**.** *Every accepted* CookCompileFrontierWitness$^{\uparrow}$ *read that claims a frontier coordinate must factor through the ordered stage boundary*

$$\mathsf{TM\text{-}to\text{-}tag}, \quad \mathsf{tag\text{-}to\text{-}cyclic}, \quad \mathsf{cyclic\text{-}to\text{-}Rule110}$$

inside S, then through the round-trip audit row A, and then through H, C, P, N. It cannot omit a stage, reorder the stage boundary, import a host compiler, appeal to an unbounded orbit, or identify two traces by quotient equality.

Proof. Use the carrier exposure of Definition 952.1. The stage ledger S is ordered and finite, with exactly the three compile-stage rows named in the statement.

Apply the obligation surface of Definition 952.2. A frontier-coordinate read reaches the local NameCert row only after the coordinate row and the round-trip audit row have been displayed, so the route cannot bypass an intermediate compiler stage.

Finally inspect the excluded coordinates. The packet contains no hidden compiler, unbounded orbit, host evaluator, or quotient trace equality; its public route is the displayed finite path through S, A, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.CookCompileFrontierWitnessUp.CookCompileFrontierWitnessCarrier stage boundary nonescape`

Theorem 952.4 (Cook compile-frontier coordinate classifier stability)**.** *If two accepted* CookCompileFrontierWitness *carriers are compared by the local classifier, then their exposed frontier-coordinate rows are stable only when the same stage ledger, audit row, obstruction row, componentwise* hsame *transports, continuation routes, provenance row, and local naming row are preserved. In particular, classifier agreement cannot be obtained by changing the compile-stage order, by dropping the audit row, or by replacing the displayed coordinate with a host-side scale claim.*

Proof. Read classifier comparison through the carrier rows of Definition 952.1. The coordinate row Q is not detached data; it is read together with the stage ledger S, audit row A, and obstruction row O.

The obligation surface of Definition 952.2 forces transport and route preservation through H and C. Thus any accepted comparison must preserve the displayed stage order and the audit replay that justifies the coordinate.

Package and naming rows close the comparison. Since P names the sibling certificates and N names the local packet, a host-side scale claim or altered stage boundary cannot be classified as the same frontier witness. □

The Lean file `Derived/CookCompileFrontierWitnessUp/TasteGate.lean` must supply Nontrivial CookCompile and FieldFaithful CookCompileFrontierWitness$^{\uparrow}$ instances for the displayed carrier.

Closure status for NameCert$_{\text{CookCompileFrontierWitness}^{\uparrow}}$.

Constructive story: A CookCompileFrontierWitness$^{\uparrow}$ packet is generated from an ordered three-stage compile ledger, frontier-coordinate row, round-trip audit row, finite obstruction row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CookCompileFrontierWitness}^{\uparrow}}$, seed)

Scope closed: Definition 952.1, Definition 952.2, Definition 952.3, and Definition 952.4 bind the carrier and obligation surface for the finite compile-frontier witness.

Formal verification: FormalStatus(NameCert$_{\text{CookCompileFrontierWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, hidden compiler correctness, unbounded Rule 110 orbit claims, quotient trace equality, or detached universality oracles.

Upgrade path: Obligation closure requires the displayed stage, coordinate, audit, obstruction, transport, route, provenance, NameCert, Nontrivial, and FieldFaithful rows for the same carrier.

Chapter 953

A Concrete Naming Certificate for $\mathsf{UniversalClosedSubstrate}^{\uparrow}$

The $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ packet is the concrete universal-and-closed row promised by chapter 9388. It specializes the two-axis quadrant surface to finite simulator data: a closed rule table, a universal compilation row, a bounded simulator trace, and a boundary ledger. It sits beside the cellular automaton packet of chapter 934, while keeping the substrate packet separate from the quadrant classifier, the cellular substrate row, the cellular row classifier, and the halting distinction boundary.

Falsifiable prediction: Any accepted $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ packet lacking a finite rule-table row cannot satisfy the chapter's NameCert obligations.

Independence witness: cellularautomaton, cellularsubstrate, cellularrowclassifier, haltingdistinctionlimit: the packet simultaneously carries universality, closure, simulator trace, and boundary-ledger rows, so none of these siblings is bijective to it.

Definition 953.1 (Universal closed substrate carrier)**.** *A* $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, A, E, S, T, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite rule-table row for the closed substrate, naming the local state alphabet, neighborhood window, update rule, and boundary convention.*
2. *A is the acceptance row saying that observations are finite substrate histories generated only by repeated use of R.*
3. *E is the universality encoding row, recording the finite compiler surface from a program row and input row into an initial substrate history.*
4. *S is the simulator-step row, relating one displayed program transition to a bounded block of rule-table steps.*
5. *T is the finite simulator trace row, listing the bounded blocks served by S for the accepted program prefix.*
6. *B is the boundary-ledger row, recording where the packet stops: no unbounded halting decision, no oracle call, and no host semantic equality is exported.*

7. *H records componentwise* hsame *transports for the rule table, closed-observation row, encoding row, simulator-step row, trace row, boundary row, package row, and local naming row.*
8. *C records displayed* Cont *routes that replay the rule-table evolution, the encoding read, each bounded simulator block, and the boundary-ledger read.*
9. *P is the* Pkg *provenance over* $\mathsf{UniversalClosedSubstrate}^{\uparrow}$*, chapter 9388, chapter 934,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniversalClosedSubstrate}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $R, A, E, S, T, B, H, C, P, N$. It contains no coordinate for replacing the finite rule table by a semantic oracle, no coordinate for deciding all program behaviour, and no quotient identifying the universal row with the closure row.

Lean4 ✓ `BEDC.Derived.UniversalClosedSubstrateUp.UniversalClosedSubstrateTasteGate single carrier alignment`

Theorem 953.2 (Universal closed substrate NameCert obligations)**.** *For the carrier of Definition 953.1, the local* $\mathsf{NameCert}_{\mathsf{UniversalClosedSubstrate}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for $U = (R, A, E, S, T, B, H, C, P, N)$;*
2. *closed-substrate accountability through the finite rule-table row R and the closed-observation row A;*
3. *universality accountability through the encoding row E and the bounded simulator-step row S;*
4. *trace exposure through T, with every simulator block backed by the displayed rule table;*
5. *boundary discipline through B, excluding halting decision, oracle access, hidden host equality, and unbounded semantic closure;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

Thus a consumer may cite $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ *only when the finite rule table, closure row, universality encoding, bounded simulator trace, and boundary ledger appear in the same packet.*

Proof. Project the carrier of Definition 953.1. Its coordinates are exactly $R, A, E, S, T, B, H, C, P, N$, so the local naming row has no other source for substrate closure, universality, simulation, or boundary data.

The closed side is read through R and A. The rule table supplies the finite local update surface, and the acceptance row restricts observations to histories generated by that table.

The universal side is read through E, S, T. The encoding row names the finite program-input translation, the simulator-step row bounds each transition block, and the trace row lists only the blocks whose rule-table support is displayed.

Finally, B, H, C, P, N bind the boundary and naming surface. Since the carrier has no coordinate for an oracle, a total halting predicate, host semantic equality, or a derivation collapsing the two axes, the listed rows exhaust the local NameCert obligations. □

Theorem 953.3 (Universal closed substrate finite simulator boundary)**.** *Every public read exported by an accepted* $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ *packet factors through the finite route*

$$R, A, E, S, T, B, H, C, P, N.$$

The route may certify that a displayed closed rule table carries a bounded simulator trace for an encoded program prefix, but it exports no proof that an arbitrary program halts, no oracle computation, no host semantic equality, no infinite closure over all traces, and no identification of universality with closure.

Proof. Begin with carrier projection. Any exported read must name the rule table R, closed-observation row A, encoding row E, simulator-step row S, trace row T, boundary ledger B, and the structural rows H, C, P, N.

The simulator route is finite. Each program-prefix read enters through E, advances only by the bounded blocks recorded by S, and is exposed only where T lists the corresponding rule-table blocks.

The boundary row B records the stopping point of the packet. A halting decision for arbitrary programs, an oracle computation, host equality, and infinite closure over all traces would require coordinates absent from U.

Therefore $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ realizes the universal-and-closed quadrant as a finite NameCert substrate packet. It is strong enough to carry the quadrant witness of chapter 9388 and the cellular automaton sibling surface of chapter 934, but not strong enough to export the excluded semantic objects. □

The later Lean `TasteGate.lean` development for $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ must provide $\mathsf{Nontrivial\ UniversalClos}$ and $\mathsf{FieldFaithful\ UniversalClosedSubstrate}^{\uparrow}$ instances in `lean4/BEDC/Derived/UniversalClosedSubstrateUp/Tas` Those instances must preserve the rule-table row, closed-observation row, universality encoding, bounded simulator-step row, finite trace, boundary ledger, componentwise transport, replay route, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{UniversalClosedSubstrate}^{\uparrow}}$.

Constructive story: A $\mathsf{UniversalClosedSubstrate}^{\uparrow}$ packet is generated from a finite rule table, closed-observation row, universality encoding row, bounded simulator-step row, finite simulator trace, boundary ledger, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniversalClosedSubstrate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 953.1, Definition 953.2, and Definition 953.3 specify the finite universal-and-closed substrate packet and its simulator boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniversalClosedSubstrate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, arbitrary halting, oracle computation, host semantic equality, infinite closure over all traces, or a derivation between universality and closure.

Upgrade path: Obligation closure requires checked carrier admission, rule-table accountability, closed-observation accountability, universality encoding, bounded simulator trace, boundary discipline, nontriviality, and field faithfulness for the same packet.

Chapter 954

A Concrete Naming Certificate for $\mathsf{CompileFrontierWitness}^{\uparrow}$

The $\mathsf{CompileFrontierWitness}^{\uparrow}$ packet realizes the Cook compile-frontier engineering path of chapter 9417 as finite NameCert data. It records the present frontier coordinate, the halted-machine target coordinate, the three compile stages, the audit rows, and the nonchoice boundary through which a substrate read is allowed to leave the packet. It is anchored by the displayed cellular classifier of chapter 945, the finite rule and orbit surface of chapter 934, and the obstruction ledger of chapter 490.

Falsifiable prediction: If the compile-frontier witness carrier is accepted, every halted-TM target read factors through a displayed frontier-coordinate row and no hidden evaluator coordinate can be exported.

Independence witness: cellularsubstrateclassifier, cellularautomaton, haltingobstruction, compiler-optimization-endomorphism: none is bijective to $\mathsf{CompileFrontierWitness}^{\uparrow}$ because none simultaneously carries present coordinate, target coordinate, stage route, audit row, and nonchoice boundary.

Definition 954.1 (Compile-frontier witness carrier)**.** *A* $\mathsf{CompileFrontierWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, T, S, A, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the frontier-coordinate row, naming the present Cook coordinate (P^*, T^*, S^*) accepted by the substrate audit.*
2. *T is the target-coordinate row, naming the halted-TM coordinate $(64, 64, S_{\mathrm{halt}})$ and the finite machine/input pair whose halted read is sought.*
3. *S is the stage-route row, displaying TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 routes as the only compile stages available to the packet.*
4. *A is the audit-row ledger, recording the stage audit, substrate round-trip audit, decode audit, and refusal rows for failed reads.*
5. *B is the nonchoice boundary row, saying that the packet exposes a witnessed-coordinate route but not a hidden evaluator, oracle, global halting classifier, or unrecorded compiler step.*
6. *H records componentwise* hsame *transport for frontier coordinates, target coordinates, stage routes, audit rows, boundary rows, provenance rows, and local naming rows.*

7. *C records displayed* Cont *routes through the frontier coordinate, target coordinate, compile stages, cellular substrate classifier, Rule 110 orbit surface, obstruction boundary, audit ledger, and nonchoice boundary.*

8. *P is the* Pkg *provenance over* $\mathsf{CompileFrontierWitness}^{\uparrow}$*, the Cook compile-frontier engineering path,* $\mathsf{CellularSubstrateClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{HaltingObstruction}^{\uparrow}$*,* $\mathsf{CompilerOptimizationEndomorphism}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompileFrontierWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, T, S, A, B, H, C, P, N$. It has no coordinate for a host evaluator, detached compiler correctness theorem, hidden decoder, global halting decision, or substrate run not named by the audit row.

Lean4 (definition) `BEDC.Derived.CompileFrontierWitnessUp.TasteGate.CompileFrontierWitnessUp`

Theorem 954.2 (Compile-frontier witness NameCert obligations)**.** *For the carrier of Definition 954.1, the local* $\mathsf{NameCert}_{\mathsf{CompileFrontierWitness}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *frontier-coordinate carrier, exposing the present coordinate F and the halted target T as displayed BEDC rows;*

2. *displayed coordinate classifier, comparing frontier and target reads only by componentwise* hsame *transport through H;*

3. *pipeline-stage route exactness, forcing every compile read through the TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 route row S;*

4. *audit-row nonescape, requiring every substrate, decode, and refusal read to pass through the audit ledger A and the displayed continuation routes C;*

5. *ledger exhaustion, saying that B, P, N exclude hidden evaluator coordinates, detached compiler steps, oracle reads, and global halting classifiers.*

Thus a public compile-frontier witness may be consumed only as a finite witnessed-coordinate NameCert packet.

Proof. Carrier exposure is projection from Definition 954.1. The present coordinate, target coordinate, stage route, audit row, nonchoice boundary, transport, replay, provenance, and local naming rows are the only coordinates of W.

The displayed coordinate classifier is the componentwise read through H. It can compare frontier rows and target rows only where the packet exposes them, so no host coordinate or hidden substrate evaluator can be inserted.

Pipeline exactness is the route row S together with the continuation row C. A halted-TM target read must pass through TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 before it can reach the cellular classifier of chapter 945 and the Rule 110 orbit surface of chapter 934.

The audit ledger A records the substrate round trip, decode check, and refusal boundary. The obstruction sibling of chapter 490 supplies the finite boundary saying that the packet exposes an obstruction-aware route rather than a global halting classifier.

Finally, B, P, N close the local ledger. Since none of these rows names an oracle, host evaluator, detached compiler step, or unrecorded decoder, the five listed rows exhaust the public NameCert surface. □

Theorem 954.3 (Compile-frontier witness nonchoice boundary)**.** *Every public read exported by an accepted* $\mathsf{CompileFrontierWitness}^{\uparrow}$ *packet factors through*

$$F, T, S, A, B, H, C, P, N.$$

The route may certify that a displayed halted-TM target is being pursued through the Cook compile-frontier stage route and its audit rows, but it cannot export a hidden evaluator coordinate, a detached compiler oracle, an unrecorded decode step, a global halting decision, or a choice of successful run outside the finite ledger.

Proof. Begin with the carrier inventory. A public read has access to the present coordinate F, target coordinate T, stage route S, audit ledger A, nonchoice boundary B, and the structural rows H, C, P, N.

The route through the Cook path is finite. The stage row S names the three compile stages of chapter 9417; C replays them as displayed continuation rows; A records the accepted audit rows and the refusal rows.

The cellular anchors are also finite. The classifier surface of chapter 945 and the orbit surface of chapter 934 supply displayed substrate rows, while chapter 490 keeps the halted-read boundary from becoming a decision procedure.

Therefore the witness is a finite coordinate certificate. It can expose where the frontier route succeeds or refuses, but it cannot choose a hidden run, evaluate a program outside the ledger, or export an unrecorded compiler coordinate. □

The later Lean development in `lean4/BEDC/Derived/CompileFrontierWitnessUp/TasteGate.lean` must provide `Nontrivial CompileFrontierWitnessUp` and `FieldFaithful CompileFrontierWitnessUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CompileFrontierWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{CompileFrontierWitness}^{\uparrow}$ packet is generated from a present frontier coordinate, halted-machine target coordinate, Cook stage route, audit ledger, nonchoice boundary, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompileFrontierWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 954.1, Definition 954.2, and Definition 954.3 specify the finite compile-frontier witness carrier, its five NameCert rows, and its nonchoice boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompileFrontierWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, Cook universality, compiler correctness, hidden evaluator soundness, arbitrary halting, or successful frontier crossing.

Upgrade path: Obligation closure requires checked carrier admission, coordinate classifier stability, stage-route exactness, audit-row nonescape, ledger exhaustion, nontriviality, and field faithfulness for the same packet.

Chapter 955

A Concrete Naming Certificate for $\mathsf{PresburgerSubUniversal}^{\uparrow}$

The $\mathsf{PresburgerSubUniversal}^{\uparrow}$ packet realizes the sub-universal closed quadrant of chapter 9388 by naming Presburger arithmetic as a finite $\mathsf{NameCert}$ surface. It sits beside the arithmetic carrier rows of chapter 28 and chapter 29, the first-order interface of chapter 197, the computable-decision surface of chapter 199, and the Turing-machine boundary of chapter 280. Its role is not to prove global arithmetic strength; it records the closed decision-trace certificate for the language generated by $0, 1, +$.

Falsifiable prediction: Any accepted finite Presburger decision trace for this packet factors through the displayed syntax, addition, quantifier-elimination, decision-trace, closure, provenance, and NameCert rows; a consumer read bypassing one of those rows refutes the packet.

Independence witness: cellularautomaton, haltingobstruction, firstorder, computable, and turingmachine: the Presburger packet is not bijective to any one sibling because it simultaneously carries a first-order arithmetic syntax row, an additive-only classifier, a finite decision-trace row, and a closed-observation non-escape row, while the listed siblings respectively expose cellular evolution, halting obstruction, generic syntax, computability evidence, or unrestricted machine boundary rows.

Definition 955.1 (Presburger sub-universal carrier)**.** *A* $\mathsf{PresburgerSubUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, A, Q, D, K, H, C, L, N)$$

with the following displayed rows:

1. *S is the Presburger syntax row for first-order formulas over the signature $(0, 1, +)$;*
2. *A is the additive arithmetic row linking the syntax to the* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Add}^{\uparrow}$ *certificate surfaces;*
3. *Q is the finite quantifier-elimination or normal-form decision row used by the Presburger decision procedure;*
4. *D is the finite decision-trace row recording the accepted yes/no outcome for the displayed formula;*
5. *K is the closure row recording that every observation consumed by the decision trace is generated inside the displayed Presburger packet;*
6. *H stores componentwise* hsame *transports for S, A, Q, D, K, L, N;*

7. *C stores displayed* Cont *routes replaying syntax formation, additive readback, decision trace, and closure readback;*
8. *L is the* Pkg *provenance ledger over* $\mathsf{PresburgerSubUniversal}^{\uparrow}$*, chapter 9388, chapter 197, chapter 199, and chapter 280;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PresburgerSubUniversal}^{\uparrow}}$ *row.*

The carrier has no coordinate for multiplication, no Turing-universal program row, no oracle row, no global semantic equality over arbitrary arithmetic structures, and no infinite-domain ratifier outside the displayed decision trace.

Lean4 (definition) `BEDC.Derived.PresburgerSubUniversalUp.PresburgerSubUniversalUp`

Theorem 955.2 (Presburger closed-observation NameCert obligations)**.** *For the carrier of Definition 955.1, the local* $\mathsf{NameCert}_{\mathsf{PresburgerSubUniversal}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier exposure for $P = (S, A, Q, D, K, H, C, L, N)$;*
2. *additive-syntax accountability through S and A;*
3. *finite decision accountability through Q and D;*
4. *closed-observation accountability through K;*
5. *componentwise stability through H and displayed replay through C;*
6. *non-escape through L and N, excluding multiplication, oracle import, Turing-universal simulation, quotient equality of names, and external semantic ratification.*

No consumer can cite Presburger arithmetic as the sub-universal closed quadrant witness unless the additive syntax, finite decision trace, and closure row are present in the same finite packet.

Proof. Carrier exposure is projection from Definition 955.1. The packet displays syntax, additive arithmetic, quantifier-elimination or normal-form decision data, decision trace, closure, transports, replay routes, provenance, and the local name.

The arithmetic and decision obligations are separate reads. The rows S and A identify the first-order additive language, while Q and D record the finite decision trace for the displayed formula.

Closure is read through K. The replay routes in C may inspect the syntax, additive readback, decision trace, and closure readback, but they do not import multiplication, a Turing-machine simulator, or an oracle row.

Finally, H, L, N transport, package, and name the same finite packet. Since every exported read is tied to the displayed Presburger decision trace, the listed rows exhaust the local NameCert surface. □

Theorem 955.3 (Presburger quadrant non-escape)**.** *Every public read exported by an accepted* $\mathsf{PresburgerSubUniversal}^{\uparrow}$ *packet factors through the finite route*

$$S, A, Q, D, K, H, C, L, N.$$

The route certifies Presburger arithmetic as a sub-universal closed inhabitant of section 9388.4: it supplies finite first-order additive syntax, a finite decision trace, and a closed observation row, but exports no multiplication coordinate, no universal-machine semantics, no halting-obstruction classifier, no oracle computation, no quotient equality of certificate names, and no universal closure over an unrecorded infinite domain.

Proof. Begin with carrier projection. The only exported coordinates are the syntax row S, additive row A, decision row Q, decision trace D, closure row K, transports H, replay routes C, provenance L, and local name N.

The NameCert obligation theorem forces each accepted read through those coordinates. In particular, a read of the quadrant witness must use the additive syntax rows and the finite decision trace before it can cite closure.

The excluded objects require rows absent from the carrier. Multiplication requires a multiplicative arithmetic coordinate, universal-machine semantics require a machine row, halting obstruction requires the separate obstruction packet of chapter 490, oracle computation requires an oracle row, quotient equality requires a quotient classifier, and unrecorded universal closure requires a ratifier outside the displayed trace. None is exported by the finite Presburger certificate. □

The later Lean `BEDC.Derived.PresburgerSubUniversalUp` development must include explicit $\mathsf{Nontrivial}\ \mathsf{PresburgerSubUniversal}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PresburgerSubUniversal}^{\uparrow}\ \mathsf{TasteGate}$ instances for the displayed Presburger carrier. These instances must preserve the syntax row, additive arithmetic row, decision row, trace row, closure row, transport rows, replay routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{PresburgerSubUniversal}^{\uparrow}}$.

Constructive story: A $\mathsf{PresburgerSubUniversal}^{\uparrow}$ packet is generated from first-order additive syntax, a finite Presburger decision row, a decision trace, a closed-observation row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PresburgerSubUniversal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 955.1, Definition 955.2, and Definition 955.3 specify the finite Presburger sub-universal closed certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PresburgerSubUniversal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove Lean verification, bridge checking, multiplication definability, Turing universality, halting decidability, oracle computation, quotient equality, or universal closure over an unrecorded infinite domain.

Upgrade path: Obligation closure requires checked carrier admission, additive-syntax accountability, decision-trace accountability, closed-observation accountability, non-escape, nontriviality, and field faithfulness for the same Presburger carrier.

Chapter 956

A Concrete Naming Certificate for $\mathsf{OracleSubstrateBoundary}^{\uparrow}$

The $\mathsf{OracleSubstrateBoundary}^{\uparrow}$ packet realizes the oracle-open substrate boundary of chapter 9388 as a finite $\mathsf{NameCert}$ carrier. It sits beside the two-axis quadrant packet of chapter 947, the closed cellular trust surface of chapter 937, and the finite cellular ledger of chapter 934. Its distinguishing datum is an explicit external-channel row: the packet can name an oracle-query boundary without converting that row into a closed substrate observation.

Falsifiable prediction: Removing the external-channel row makes the packet indistinguishable from a closed cellular substrate, while removing the refusal ledger permits an internal-induction read.

Independence witness: Nearest siblings: universalityclosurequadrant, cellulartrustsubstrate, cellularautomaton, kernelobservationsieve, and boundeduniversaltrace; none is bijective because this carrier has an oracle-query boundary and refusal ledger absent from closed substrate packets.

Definition 956.1 (Oracle-substrate boundary carrier)**.** *An* $\mathsf{OracleSubstrateBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, Q, E, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the substrate row, naming the machine, cellular, or host-side carrier whose observation boundary is being inspected;*
2. *Q is the oracle-query row, listing the finite query sites that leave the closed substrate interface;*
3. *E is the external-channel row, recording the boundary at which a query is routed outside the closed observation ledger;*
4. *R is the refusal ledger, recording which internal-induction, closed-observation, quotient-equality, and host-semantic reads are rejected for the same packet;*
5. *L is the local observation ledger, recording the closed substrate rows that remain visible on the substrate side of the boundary;*
6. *H records componentwise* hsame *transports for S, Q, E, R, L, P, N;*
7. *C records displayed* Cont *routes that replay the local observation rows, the oracle-query boundary read, and the refusal ledger;*

8. *P is the* Pkg *provenance over* $\mathsf{OracleSubstrateBoundary}^{\uparrow}$*, chapter 9388,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OracleSubstrateBoundary}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the substrate row, oracle-query row, external-channel row, refusal ledger, local observation ledger, componentwise transports, replay routes, provenance, and naming row. It has no coordinate for replacing the external channel by a closed cellular observation or for reading oracle refusal as an induction principle internal to the substrate.

Lean4 ✓ `BEDC.Derived.OracleSubstrateBoundaryUp.OracleSubstrateBoundaryTasteGate single carrier alignment`

Theorem 956.2 (Oracle-substrate boundary NameCert obligations)**.** *For the carrier of Definition 956.1, the local* $\mathsf{NameCert}_{\mathsf{OracleSubstrateBoundary}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier exposure for $O = (S, Q, E, R, L, H, C, P, N)$;*
2. *substrate-side observation accountability through S and L;*
3. *oracle-query accountability through the displayed query row Q;*
4. *external-channel boundary accountability through E;*
5. *refusal accountability through R, excluding closed-substrate induction, quotient equality, hidden host semantics, and oracle import into local observations;*
6. *componentwise stability, route replay, package provenance, and local naming through H, C, P, N.*

No consumer can cite an oracle-open substrate boundary as $\mathsf{OracleSubstrateBoundary}^{\uparrow}$ *evidence unless the query row, external-channel row, and refusal ledger are present in the same finite packet.*

Proof. Carrier exposure is projection from Definition 956.1. The accepted packet displays the substrate row, oracle-query row, external-channel row, refusal ledger, local observation ledger, transports, routes, package provenance, and local name as its only coordinates.

Substrate accountability and oracle accountability are separate reads. The local observation ledger L records closed substrate rows, while Q and E record the query sites and the boundary through which those sites leave the closed interface.

The refusal ledger blocks the escape modes named in the statement. A consumer may replay the finite boundary route through C, but there is no coordinate that turns an external query into closed-substrate induction, quotient equality, hidden host semantics, or an oracle imported into the local observation rows.

Finally, H, P, N transport, package, and name exactly the same finite packet. These rows certify stability and provenance for the displayed carrier; they do not add any missing closed-substrate or internal-induction coordinate. □

Theorem 956.3 (Oracle-substrate boundary external-channel non-escape)**.** *Every public boundary read exported by an accepted* $\mathsf{OracleSubstrateBoundary}^{\uparrow}$ *packet factors through the finite route*

$$S, Q, E, R, L, H, C, P, N.$$

The route can certify that a displayed substrate has an oracle-query boundary and a refusal ledger, but it exports no closed cellular substrate certificate, no internal induction principle for oracle answers, no quotient equality of behaviours, no host semantic equality, and no oracle computation detached from the displayed boundary row.

Proof. Begin with carrier projection. The only public rows are the substrate row S, oracle-query row Q, external-channel row E, refusal ledger R, local observation ledger L, transports H, replay routes C, provenance P, and local name N.

The query boundary must be read through Q and E. The closed substrate side remains only the finite observation ledger L, so the carrier keeps the oracle-facing boundary distinct from the cellular or host-side rows it may reference.

The refusal ledger records the missing routes. A closed cellular substrate certificate would require the external-channel row to disappear; an internal induction principle for oracle answers would require a rule absent from R; quotient equality and host semantic equality would require coordinates not present in the packet.

Thus every accepted public read remains a finite boundary certificate. It may identify the oracle-open edge of the substrate, but it cannot export an oracle computation or closed-substrate conclusion outside the displayed route. □

The Lean development BEDC.Derived.OracleSubstrateBoundaryUp.TasteGate will later provide Nontrivial OracleSubstrateBoundary$^{\uparrow}$ and FieldFaithful OracleSubstrateBoundary$^{\uparrow}$ instances for the displayed carrier. These instances must preserve the substrate row, oracle-query row, external-channel row, refusal ledger, local observation ledger, transport rows, continuation routes, package provenance, and local naming row.

Closure status for NameCert$_{\text{OracleSubstrateBoundary}^{\uparrow}}$.

Constructive story: An OracleSubstrateBoundary$^{\uparrow}$ packet is generated from a substrate row, oracle-query row, external-channel row, refusal ledger, local observation ledger, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{OracleSubstrateBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 956.1, Definition 956.2, and Definition 956.3 specify the finite oracle-query boundary certificate surface.

Formal verification: FormalStatus(NameCert$_{\text{OracleSubstrateBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, oracle computation, closed cellular substrate completeness, internal induction over oracle answers, quotient equality, or host semantic equality.

Upgrade path: Obligation closure requires checked carrier admission, substrate-side accountability, oracle-query accountability, external-channel accountability, refusal-ledger exactness, nontriviality, and field faithfulness for the same boundary carrier.

Chapter 957

A Concrete Naming Certificate for $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$

The $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ packet realizes the two-axis closure-universality map of chapter 9388 as a finite substrate-quadrant $\mathsf{NameCert}$ carrier. It sits beside the Rule 110 substrate certificate of chapter 934, the cellular substrate carrier of chapter 942, and the bounded trace surface of chapter 944. Its public datum is the closed-substrate quadrant tag together with the displayed universality and closure rows; it does not identify those axes with a sibling substrate, trace, or trust row.

Falsifiable prediction: A substrate-quadrant consumer that changes either the displayed universality row or closure row without changing the quadrant tag is outside the certificate.

Independence witness: cellularautomaton, cellularsubstrate, cellulartrustsubstrate, boundeduniversaltrace: none is bijective because the quadrant packet carries two axis rows and a quadrant tag, while the siblings carry substrate, trust, or trace rows without the full two-axis classifier.

Definition 957.1 (Closed substrate quadrant carrier)**.** *A* $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, U, K, G, W, H, C, P, N)$$

whose rows have the following meanings:

1. *S names the substrate being classified, such as a cellular automaton substrate, a cellular substrate carrier, an oracle-facing substrate, or a host-side closed fragment;*
2. *U is the displayed universality row* $\mathsf{Univ}(S)$*, including the finite witness or refusal ledger used by the consumer;*
3. *K is the displayed closure row* $\mathsf{Closed}(S)$*, including the finite observation-channel witness or refusal ledger;*
4. *G is the substrate-quadrant tag for the ordered pair (U, K), with cells universal-and-closed, universal-and-open, subuniversal-and-closed, and subuniversal-and-open;*
5. *W is the provenance witness tying S, U, K, G to the concrete substrate-map read of chapter 9388;*
6. *H records componentwise* hsame *transports for the substrate, axis, quadrant, witness, provenance, and naming rows;*
7. *C records the displayed* Cont *routes that replay the two axis reads and the quadrant-tag read;*

8. *P is the* Pkg *provenance over* $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$*, the cited vision map, the sibling substrate certificates,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSubstrateQuadrant}^{\uparrow}}$ *row.*

The local classifier compares packets only through these nine rows. It has no coordinate for deriving universality from closure, deriving closure from universality, replacing the quadrant tag by a sibling substrate carrier, or importing an ambient semantic judgement about the substrate.

Theorem 957.2 (Closed substrate quadrant NameCert obligations)**.** *For the carrier of Definition 957.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedSubstrateQuadrant}^{\uparrow}}$ *surface consists exactly of six finite obligation families:*

1. *carrier exposure for $Q = (S, U, K, G, W, H, C, P, N)$;*
2. *universality-axis accountability through the displayed pair S, U;*
3. *closure-axis accountability through the displayed pair S, K and witness row W;*
4. *quadrant-tag accountability through G as a read of the ordered pair (U, K);*
5. *stability and replay through componentwise* hsame *rows H and* Cont *routes C;*
6. *non-escape through P, N, excluding axis collapse, hidden host semantics, quotient substrate equality, oracle import into closed rows, and replacement by a sibling substrate or trace certificate.*

Thus an accepted $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ *packet is a finite NameCert packet for the full two-axis classifier, not a renaming of any one substrate, trust, or trace surface.*

Proof. Carrier exposure is projection from Definition 957.1. The packet displays one substrate row, two axis rows, one quadrant tag, one witness row, transports, replay routes, package provenance, and a local naming row.

The axis obligations are separate. The universality row U records the computational-power classifier for S, while the closure row K records the observation-channel classifier for S and W. The continuation rows can replay both reads, but they do not identify them.

The quadrant obligation is the finite ordered-pair read. The tag G is accepted only as a classifier for (U, K), so changing either row without changing the tag exits the displayed certificate surface.

Finally, H, C, P, N transport, replay, package, and name the same rows. Since no listed coordinate supplies host semantic equality, quotient substrate equality, axis derivability, or a sibling-only carrier replacement, the six obligation families exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ClosedSubstrateQuadrantUp.ClosedSubstrateQuadrantCarrier namecert obligations`

Theorem 957.3 (Closed substrate quadrant independence certificate)**.** *In an accepted* $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ *packet, the quadrant tag G is independent of each single axis row taken alone. The substrate row S with only U does not determine closure, S with only K does not determine universality, and the sibling cellularautomaton, cellularsubstrate, cellulartrustsubstrate, and boundeduniversaltrace certificates do not determine the full packet unless they also display both axis rows and the same quadrant tag.*

Proof. Project the packet to S, U, K, G, W. The rows U and K are separate coordinates, and the carrier contains no map from either coordinate to the other.

The quadrant tag is a read of the ordered pair. A consumer that holds only S, U lacks the closure witness row; a consumer that holds only S, K lacks the universality witness row; and a

consumer that holds only a sibling substrate, trust, or trace packet lacks the full pair (U, K) together with G.

The witness row W records which displayed substrate-map case is being read, but it is not a bijection to any one sibling. Therefore the accepted packet certifies exactly the finite two-axis classifier and keeps the four substrate quadrants distinct at the NameCert surface. □

Theorem 957.4 (Closed substrate quadrant consumer boundary)**.** *Every public read exported by an accepted* $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ *packet factors through*

$$S, U, K, G, W, H, C, P, N.$$

The read may consume the displayed substrate quadrant, but it exports no proof of Rule 110 universality, no oracle computation, no host closure theorem, no quotient substrate equality, no ambient semantic equality, and no derivation between $\mathsf{Univ}(S)$ *and* $\mathsf{Closed}(S)$*.*

Proof. Start with the carrier projection. The public packet has exactly the substrate row, universality row, closure row, quadrant tag, witness row, transport rows, continuation routes, package provenance, and local name.

Apply the obligation theorem to route every NameCert read through the displayed axis rows and quadrant tag. Apply the independence certificate to block replacement by any single axis read or sibling substrate certificate.

The excluded objects all require rows not present in Q. A Rule 110 proof, oracle computation, host closure theorem, quotient equality, ambient semantic equality, or axis derivation must be supplied by another certificate; none is exported by the closed substrate quadrant packet. □

The Lean touchpoint for this carrier is `BEDC.Derived.ClosedSubstrateQuadrantUp.TasteGate`; it supplies $\mathsf{Nontrivial}\ \mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ instances for the same nine displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubstrateQuadrant}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedSubstrateQuadrant}^{\uparrow}$ packet is generated from a substrate row, universality-axis row, closure-axis row, quadrant tag, provenance witness, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubstrateQuadrant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 957.1, Definition 957.2, Definition 957.3, and Definition 957.4 specify the finite two-axis closed-substrate quadrant certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedSubstrateQuadrant}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ClosedSubstrateQuadrantUp.ClosedSubstrateQuadrantTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove bridge checking, Rule 110 universality, oracle computation, host closure, quotient substrate equality, ambient semantic equality, or derivability between the axes.

Upgrade path: Theorem-checked closure requires canonical Lean theorems for the obligation surface, axis independence, and consumer boundary over the same carrier.

Chapter 958

A Concrete Naming Certificate for $\mathsf{HaltedTmRoundTrip}^{\uparrow}$

The $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ packet is the finite certificate target for the Cook compile-frontier engineering path of chapter 9417. It joins a staged Turing-machine compilation row, a Rule 110 orbit row, an inverse decode row, a halted readback row, and a scale-threshold row in one NameCert carrier. The packet sits beside the cellular trust substrate of chapter 940, the bounded universal trace of chapter 944, the compiler optimization endomorphism of chapter 639, and the universal closed substrate of chapter 953; it is not reducible to any one of those siblings because it records the complete round trip from machine input to halted readback at a finite frontier coordinate.

Falsifiable prediction: Deleting the decode row permits an orbit witness without a halted-TM readback; deleting the scale-threshold row permits a frontier claim without the finite Cook-stage budget.

Independence witness: cellulartrustsubstrate, cellularautomaton, boundeduniversaltrace, compiler-optimization-endomorphism, universalclosedsubstrate; none is bijective because none preserves all staged compile, orbit, decode, halt, and frontier-scale rows.

Definition 958.1 (HaltedTmRoundTrip carrier)**.** *A* $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, G, Y, R, D, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the halted Turing-machine source row, naming the finite state set, tape alphabet, transition table, initial configuration, and halted output row selected for the target machine;*
2. *G is the staged Cook compilation row, recording the TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 stages as finite compiler rows;*
3. *Y is the cyclic-tag schedule row, including the production count, tape-symbol count, initial cyclic word, and finite schedule bound used by the compiled run;*
4. *R is the Rule 110 initial row and local-rule orbit row, with the finite cell window, step budget, and accepted observation coordinates;*
5. *D is the inverse decode row from the observed Rule 110 window back through cyclic-tag and tag-system readback;*

6. *Q is the halted readback row, stating that the decoded machine row matches the halted output named by M;*
7. *F is the frontier-scale row, recording the finite production, tape-symbol, row-width, and step-budget thresholds used by the compile-frontier read;*
8. *H records componentwise* hsame *transports for $M, G, Y, R, D, Q, F, P, N$;*
9. *C records displayed* Cont *routes through the staged compile, Rule 110 orbit, inverse decode, halted readback, and frontier-scale rows;*
10. *P is the* Pkg *provenance over* $\mathsf{HaltedTmRoundTrip}^{\uparrow}$*, chapter 9417, chapter 940, chapter 944, chapter 639, chapter 953,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltedTmRoundTrip}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $M, G, Y, R, D, Q, F, H, C, P, N$. It has no coordinate for an unbounded halting predicate, host evaluator equality, a detached Rule 110 orbit, a decode-free universality claim, or a scale-free frontier assertion.

Lean4 (definition) `BEDC.Derived.HaltedTmRoundTripUp.TasteGate.HaltedTmRoundTripUp`

Theorem 958.2 (HaltedTmRoundTrip NameCert obligations)**.** *For the carrier of Definition 958.1, the local* $\mathsf{NameCert}_{\mathsf{HaltedTmRoundTrip}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (M, G, Y, R, D, Q, F, H, C,$ staged compile accountability through G, cyclic-tag schedule accountability through Y, Rule 110 orbit accountability through R, inverse decode exactness through D, halted readback exactness through Q, frontier-scale accountability through F, and componentwise transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 958.1. The source machine, staged compiler, cyclic schedule, Rule 110 orbit, inverse decode, halt readback, scale row, transport, route, provenance, and naming rows are the only coordinates available to a consumer.

The compile side is read through M, G, Y, R. The source row names the halted machine, the staged compiler row records the three Cook stages, the schedule row fixes the finite cyclic-tag budget, and the Rule 110 row records the finite orbit window accepted for observation.

The readback side is read through D, Q, F. The decode row returns from the observed orbit to the machine-facing row, the halt row checks that decoded row against the displayed halted output, and the scale row records the finite frontier budget under which the route is being claimed.

Finally H, C, P, N bind the NameCert surface. Since the carrier has no coordinate for unbounded halting, host evaluator equality, a detached orbit witness, a decode-free readback, or a scale-free frontier claim, the listed rows exhaust the local obligations. □

Theorem 958.3 (HaltedTmRoundTrip scale-threshold route)**.** *Every compile-frontier read exported by an accepted* $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ *packet factors through the staged finite route*

$$M, G, Y, R, D, Q, F, H, C, P, N.$$

The route may state that the displayed halted machine has been compiled through the Cook stages, evaluated on a finite Rule 110 orbit, decoded back to the halted row, and checked at the thresholds recorded by F. It may not state the frontier claim without the scale row, the halted readback without the decode row, or the Rule 110 orbit without the staged compile row.

Proof. Begin with the route row C. Its admitted paths replay the staged compiler rows, then the cyclic-tag schedule, then the Rule 110 orbit, then the inverse decode and halted readback.

The scale-threshold row F is read on the same route. It records the finite production count, tape-symbol count, row-width budget, and step-budget threshold used by the frontier claim, so the exported read is a bounded certificate rather than an ambient universality assertion.

The dependency on D and Q is essential. Without D, the packet contains an orbit observation but no machine-facing readback; without Q, it contains a decode but no halted-row match; without F, it contains a round trip but no Cook-frontier coordinate. □

Theorem 958.4 (HaltedTmRoundTrip nonescape)**.** *An accepted* $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ *packet exports no unbounded halting predicate, no host evaluator equality, no detached Rule 110 orbit certificate, no decode-free halted readback, no scale-free frontier claim, and no semantic proof that every Turing machine is handled by the displayed route. Its public content is exactly the finite packet* $M, G, Y, R, D, Q, F, H, C, P, N$ *of Definition 958.1.*

Proof. Inspect the carrier coordinates. The source row names one finite halted-machine target, the compile rows name one staged Cook route, and the orbit row names one finite Rule 110 observation window.

The readback and scale rows restrict the export. The decoded row D and halt row Q bind the orbit to the named halted output, while F binds the read to a finite frontier coordinate.

The excluded objects require rows absent from the carrier. Unbounded halting needs a global decision predicate, host evaluator equality needs a semantic equality row, a detached orbit certificate would omit G, a decode-free halt read would omit D, and a scale-free frontier claim would omit F. None of these coordinates is present, so the packet exports only the displayed finite NameCert surface. □

The Lean `TasteGate.lean` development for $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ will need $\mathsf{Nontrivial\ HaltedTmRoundTrip}^{\uparrow}$ and $\mathsf{FieldFaithful\ HaltedTmRoundTrip}^{\uparrow}$ instances. These instances must preserve the halted-machine source row, staged compile row, cyclic-tag schedule, Rule 110 orbit, inverse decode, halted readback, frontier-scale row, componentwise transport, replay route, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{HaltedTmRoundTrip}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HaltedTmRoundTrip}^{\uparrow}$ packet is generated from a halted Turing-machine source row, staged Cook compile row, cyclic-tag schedule, Rule 110 orbit row, inverse decode row, halted readback row, frontier-scale row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaltedTmRoundTrip}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 958.1, Definition 958.2, Definition 958.3, and Definition 958.4 specify the finite Cook compile-frontier round-trip certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaltedTmRoundTrip}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, host evaluator equality, global Rule 110 correctness, all-machine universality, or a detached decode-free frontier result.

Upgrade path: Obligation closure requires checked carrier admission, staged compile accountability, Rule 110 orbit accountability, inverse decode exactness, halted readback exactness, frontier-scale accountability, nontriviality, and field faithfulness for the same packet.

Chapter 959

Cook Frontier Witness Name Certificate Construction

The $\mathsf{CookFrontierWitness}^{\uparrow}$ packet realizes the compile-frontier engineering path of chapter 9417 as a finite NameCert witness surface. It sits beside the substrate-facing cellular packet of chapter 934 and the halting-facing diagonal packet of chapter 496. Its role is not to prove Cook universality again, decide a halting problem, or identify compiler stages by host equality; it records the finite stage, target, replay, obstruction, transport, continuation, package, and naming rows a consumer may inspect.

Falsifiable prediction: Removing the replay-audit row permits a frontier coordinate to be reported without a finite route back through the TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 stages.

Independence witness: cellularautomaton, compiler-optimization-endomorphism, haltingdiagonal, continuationautomaton, universalclosedsubstrate: no listed sibling is bijective to $\mathsf{CookFrontierWitness}^{\uparrow}$. The cellularautomaton packet carries a substrate rule surface without the three-stage Cook compile route; compiler-optimization-endomorphism carries before/after compiler traces without a halted target and replay audit; haltingdiagonal carries diagonal obstruction data without the Cook stage ledger; continuationautomaton carries replay dynamics without the frontier coordinate and target rows; universalclosedsubstrate carries a universal closed substrate packet without the staged Cook witness route.

Definition 959.1 (Cook frontier witness carrier)**.** *A* $\mathsf{CookFrontierWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (F, M, Y, R, T, A, O, H, C, P, N)$$

with the following displayed rows:

1. *F is the frontier-coordinate row, recording the accepted production count, tape-symbol count, Rule 110 step budget, and the reported coordinate relative to the halted-TM target of Definition 9417.1;*
2. *M is the TM-to-tag compile-stage row, naming the finite Turing-machine input, tag-system output, and stage-local audit route;*
3. *Y is the tag-to-cyclic compile-stage row, naming the finite tag-system input, cyclic-tag output, and stage-local audit route;*

4. *R is the cyclic-to-Rule-110 compile-stage row, naming the finite cyclic-tag input, Rule 110 initial row, evaluator budget, and decode route;*
5. *T is the halted-target row, recording the halted-machine target coordinate and the expected halt-state readback;*
6. *A is the replay-audit row, replaying the round trip from the Rule 110 trace back through R, Y, and M to the target read in T;*
7. *O is the obstruction-ledger row, recording memory scaling, encoding constants, verification cost, and any displayed failure boundary for the accepted coordinate;*
8. *H records componentwise* hsame *transports for $F, M, Y, R, T, A, O, P, N$;*
9. *C records displayed* Cont *routes for each compile stage, the replay audit, the obstruction read, and the local naming row;*
10. *P is the* Pkg *provenance over* $\mathsf{CookFrontierWitness}^{\uparrow}$*, chapter 9417, chapter 934, chapter 496,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CookFrontierWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for a total halting decision, a semantic oracle, a host equality between compiler stages, an unbounded substrate trace, or an unstaged Cook universality proof.

Theorem 959.2 (Cook frontier witness obligation surface)**.** *For the carrier of Definition 959.1, the local* $\mathsf{NameCert}_{\mathsf{CookFrontierWitness}^{\uparrow}}$ *surface consists exactly of these finite rows:*

1. *frontier-coordinate exposure through F;*
2. *three compile-stage exposures through the TM-to-tag row M, the tag-to-cyclic row Y, and the cyclic-to-Rule-110 row R;*
3. *halted-target exposure through T;*
4. *replay-audit exposure through A, requiring the finite route back through R, Y, M;*
5. *obstruction-ledger exposure through O;*
6. *componentwise transport through H;*
7. *continuation replay through C;*
8. *package provenance through P;*
9. *local naming through N.*

Thus an accepted $\mathsf{CookFrontierWitness}^{\uparrow}$ *consumer may report a Cook frontier coordinate only through $F, M, Y, R, T, A, O, H, C, P, N$, and not through a bare substrate run, host compiler equality, selected halting conclusion, unstaged replay, or ambient universality theorem.*

Lean4 ✓ `BEDC.Derived.CookFrontierWitnessUp.CookFrontierWitnessTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 959.1. The displayed packet contains exactly the frontier row, three compile-stage rows, halted-target row, replay-audit row, obstruction ledger, and structural rows H, C, P, N.

The compile route is staged. The row M supplies the TM-to-tag read, Y supplies the tag-to-cyclic read, and R supplies the cyclic-to-Rule-110 read. The replay audit A can therefore be checked only by following the finite route back through R, Y, M to the halted target T.

The obstruction row O records where the accepted coordinate stops and why a larger coordinate is not claimed by this packet. The transport, continuation, provenance, and naming rows then bind those finite reads into one NameCert surface.

Since no coordinate remains for a total halting decision, host compiler equality, semantic oracle, unbounded trace, or unstaged universality theorem, the listed rows exhaust the public obligation surface. □

Lean `BEDC/Derived/CookFrontierWitnessUp/TasteGate.lean` will provide explicit Nontrivial CookFrontierWi and FieldFaithful CookFrontierWitness$^{\uparrow}$ instances. Those instances must preserve the frontier-coordinate row, the three compile-stage rows, the halted-target row, the replay-audit row, the obstruction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Lemma 959.3 (Cook frontier replay-audit route determinacy)**.** *Let*

$$W = (F, M, Y, R, T, A, O, H, C, P, N) \quad \textit{and} \quad W' = (F, M, Y, R, T, A, O, H, C, P, N)$$

be two accepted CookFrontierWitness$^{\uparrow}$ *carriers whose displayed rows agree component by component: the frontier-coordinate row F, the TM-to-tag row M, the tag-to-cyclic row Y, the cyclic-to-Rule-110 row R, the halted-target row T, the replay-audit row A, the obstruction boundary row O, the componentwise transport row H, the continuation row C, the provenance row P, and the local naming row N are the same rows in both packets. If u is a frontier-coordinate read exposed by W and u' is a frontier-coordinate read exposed by W', then the common replay-audit route through A and the common componentwise transport row H identify the two reads:*

$$u \mathsf{hsame} u'.$$

Equivalently, duplicate accepted CookFrontierWitness$^{\uparrow}$ *packets with the same finite replay route cannot expose two non-*hsame *frontier-coordinate reads.*

Proof. Project the two accepted carriers by Definition 959.1. The first coordinate of each packet is the same frontier-coordinate row F, and the replay side of each packet is the same finite route: the row A replays the Rule 110 trace back through R, then Y, then M, and finally to the halted-target row T.

By Definition 959.2, an accepted consumer may report a Cook frontier coordinate only through the displayed rows $F, M, Y, R, T, A, O, H, C, P, N$. Thus the read u from W and the read u' from W' both factor through the same frontier-coordinate row F and the same replay-audit row A.

The componentwise transport row H records the hsame transports for $F, M, Y, R, T, A, O, P, N$. Applying its frontier-coordinate component to the two factorizations through the common row F gives the displayed identification of the two frontier-coordinate reads. The continuation row C, provenance row P, and naming row N are also common, so no second route remains through which either packet could expose a different frontier-coordinate read. Therefore $u \mathsf{hsame} u'$ under the same finite replay route. □

Closure status for NameCert$_{\text{CookFrontierWitness}^{\uparrow}}$**.**

Constructive story: A CookFrontierWitness$^{\uparrow}$ packet is generated from a finite frontier coordinate, the TM-to-tag, tag-to-cyclic, and cyclic-to-Rule-110 stage rows, a halted-target row, replay audit, obstruction ledger, componentwise transport, continuation replay, provenance, and local naming. The BHist carrier keeps each stage as a displayed coordinate; hsame identifies only matching stage and audit rows; and Cont replays the route from the Rule 110 trace back through the compiler stages to the halted target. The certificate names a bounded frontier witness rather than Cook universality itself.

Theory closure: TheoryGate(NameCert$_{\mathsf{CookFrontierWitness}\uparrow}$, seed)

Scope closed: This chapter binds the Cook frontier witness carrier, obligation surface, and replay-audit route determinacy lemma.

Formal verification: FormalStatus(NameCert$_{\mathsf{CookFrontierWitness}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim a total halting decision, semantic oracle, host compiler-stage equality, unbounded trace, or unstaged universality theorem.

Upgrade path: To reach obligationClosure, formalize the staged compiler rows and replay-audit determinacy as a checked finite NameCert carrier.

Chapter 960

A Concrete Naming Certificate for $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$

The $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ packet realizes the step-indexed bridge of Definition 9370.5 and Definition 9370.7 as a finite NameCert carrier. It keeps the total host and the universal substrate in different rows: the host evaluator is total and fuel-indexed, while the substrate row may name a universal computation only through a displayed finite window. The packet therefore gives the strong-normalisation vision a concrete finite carrier without claiming host partiality or unbounded substrate completion.

Falsifiable prediction: Removing the fuel row allows a consumer to confuse one finite substrate window with an unbounded computation, which the $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ carrier rejects.

Independence witness: Nearest siblings: twinsubstratebridge, cellularsubstrate, continuationautomaton, closednormalroutecertificate, and boundeduniversaltrace. None is bijective to FuelIndexedSubstrateBri because none simultaneously carries fuel, substrate-state, total-host evaluator, and refusal rows separating a finite substrate window from an unbounded computation.

Definition 960.1 (Fuel-indexed substrate bridge carrier)**.** *A* $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, S, E, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the fuel row, naming the finite step bound under which the host read is accepted;*
2. *S is the substrate-state row, naming the finite cellular, machine, or continuation state visible at that fuel;*
3. *E is the total-host evaluator row, naming the terminating host function that maps a fuel and substrate input to a finite readback;*
4. *W is the finite substrate-window row exposed to downstream consumers;*
5. *R is the readback row produced from F, S, E, W;*
6. *Q is the refusal row excluding unbounded completion, host partiality, hidden halting evidence, and any identification of a finite window with an all-time computation;*
7. *H records componentwise* hsame *transports for F, S, E, W, R, Q, P, N;*

8. *C records displayed* Cont *routes through the fuel, substrate-state, total-host evaluator, finite-window, readback, and refusal rows;*
9. *P is the* Pkg *provenance over* $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$*, chapter 9370, chapter 942, chapter 944,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through these displayed rows. It has no coordinate for a partial host evaluator, an unbounded substrate orbit, a halting oracle, quotient stream equality, selected nontermination evidence, or semantic completion of the universal substrate.

Lean4 (definition) `BEDC.Derived.FuelIndexedSubstrateBridgeUp.FuelIndexedSubstrateBridgeUp`

Theorem 960.2 (Fuel-indexed substrate bridge obligation surface)**.** *For the carrier of Definition 960.1, the local* $\mathsf{NameCert}_{\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (F, S, E, W, R, Q, H, C, P, N)$, fuel exposure through F, substrate-state exposure through S, total-host evaluation through E, finite substrate-window exposure through W, readback through R, refusal accounting through Q, componentwise* hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FuelIndexedSubstrateBridgeUp.FuelIndexedSubstrateBridgeTasteGate single carrier alignment`

Proof. Project the displayed carrier. The accepted packet has exactly the coordinates $F, S, E, W, R, Q, H, C, P, N$, so carrier admission cannot introduce another fuel value, substrate state, host evaluator, window, readback, or refusal row.

Read the computational rows in dependency order. The fuel row F bounds the accepted host read, S provides the substrate state seen at that bound, E is the total evaluator that consumes the bounded input, and W, R display the finite window and readback delivered to the consumer.

The refusal and structural rows close the surface. The row Q records the excluded escapes, while H, C, P, N transport, replay, package, and name only the same finite packet. Thus the NameCert surface is exactly the displayed fuel-indexed bridge and nothing outside it. □

Theorem 960.3 (Fuel-indexed substrate bridge total-host non-escape)**.** *Every accepted* FuelIndexedSubstrateBridge *consumer route reads the host computation through the total evaluator row E under the displayed fuel row F. The packet exports no partial host evaluator, no host-level halting oracle, no selected nontermination witness, and no proof that the substrate computation has completed beyond the displayed finite window W.*

Lean4 ✓ `BEDC.Derived.FuelIndexedSubstrateBridgeUp.FuelIndexedSubstrateBridge total host nonescape`

Proof. Start from the obligation surface of Definition 960.2. A consumer reaches the readback row only through $F, S, E, W, R, Q, H, C, P, N$.

The evaluator row E is consumed with the fuel row F. Hence the host-facing computation is the total step-indexed read described by Definition 9370.5, not a partial evaluator or an unbounded host run.

The refusal row Q records the missing escapes. Since the carrier has no coordinate for host partiality, a halting oracle, selected nontermination evidence, or substrate completion beyond W, none of those objects can be exported by the consumer route. □

Theorem 960.4 (Fuel-indexed substrate bridge step-window boundary)**.** *For an accepted* FuelIndexedSubstrateBrid *packet* $B = (F, S, E, W, R, Q, H, C, P, N)$*, the substrate-side content available to any downstream route is exactly the finite step window* W *and readback* R *obtained under* F, S, E*. In particular, one finite window cannot be consumed as an unbounded computation, a completed universal substrate run, or a quotient stream of all future substrate states.*

Lean4 ✓ `BEDC.Derived.FuelIndexedSubstrateBridgeUp.FuelIndexedSubstrateBridge step window boundary`

Proof. Project the carrier of Definition 960.1. The only substrate-facing rows are the state row S, the finite window row W, and the readback row R, all governed by the fuel and evaluator rows F, E.

Apply Definition 960.3. The consumer receives a total fuel-indexed host read and no unbounded host or substrate completion coordinate.

It remains to identify the boundary of the substrate read. The refusal row Q excludes treating W as an all-time computation or as a quotient stream of future states, while H, C, P, N preserve only the displayed finite route. Therefore the public substrate content is exactly the stated finite step window and readback. □

The Lean TasteGate module BEDC.Derived.FuelIndexedSubstrateBridgeUp.TasteGate must provide Nontrivial FuelIndexedSubstrateBridgeUp and FieldFaithful FuelIndexedSubstrateBridgeUp instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}}$ •

Constructive story: A $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ packet is generated from fuel, substrate-state, total-host evaluator, finite-window, readback, refusal, transport, route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 960.1, Definition 960.2, Definition 960.3, and Definition 960.4 specify the finite total-host and universal-substrate bridge packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, host partiality, unbounded substrate completion, halting decidability, quotient stream equality, or semantic completion of the universal substrate.

Upgrade path: Obligation closure requires checked carrier admission, fuel exposure, substrate-state exposure, total-host evaluator accountability, finite-window exactness, refusal non-escape, nontriviality, and field faithfulness for the same packet.

Chapter 961

A Concrete Naming Certificate for $\mathsf{DiagonalLimitObserver}^{\uparrow}$

The $\mathsf{DiagonalLimitObserver}^{\uparrow}$ packet makes the observer-facing route in chapter 9238 finite. It reads the diagonal-limit construction through the concrete Real and Completion certificates of chapter 38, chapter 126, chapter 301, chapter 291, chapter 386, and chapter 646. The packet treats the observer route as primitive: the observer does not receive a completed real, a quotient stream, a host equality proof, or an ambient completeness principle, only the displayed finite rows by which a diagonal read reaches a $\mathsf{Real}^{\uparrow}$ seal.

Falsifiable prediction: No accepted $\mathsf{DiagonalLimitObserver}^{\uparrow}$ read reaches the $\mathsf{Real}^{\uparrow}$ seal row without first exposing the $\mathsf{StreamName}^{\uparrow}$ finite-window row and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. A claimed observer read that presents the seal while hiding either row refutes this carrier.

Independence witness: diagonallimitcompatibility, realobservationbudget, cauchylimitseal, regularcauchyselectorbudget, streamname: none is bijective to $\mathsf{DiagonalLimitObserver}^{\uparrow}$ because this carrier makes the observer route primitive. DiagonalLimitCompatibility certifies compatibility of a diagonal route with seal rows, RealObservationBudget certifies request caps, CauchyLimitSeal certifies the seal route, RegularCauchySelectorBudget certifies selector spending, and StreamName certifies stream rows separately; none alone carries the observer packet tying all six rows to one local NameCert surface.

Definition 961.1 (Diagonal-limit observer carrier)**.** *A* $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, W, Q, T, S, R, H, C, P, N)$$

with the following displayed coordinates:

1. *D is the diagonal packet naming the* $\mathsf{RealCompleteness}^{\uparrow}$ *source, the diagonal representative, and the observed diagonal indices;*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger exposed to the observer before any seal-facing read;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows;*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and readback rows;*
5. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route that consumes the readback and tolerance rows;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through D, W, Q, T, S;*

7. *H stores componentwise* hsame *transports for the diagonal packet, window ledger, readback row, tolerance ledger, seal route, Real seal, provenance, local name, and packet shape;*
8. *C stores displayed* Cont *routes from D to W, from W to Q, from W to T, from Q, T to S, from S to R, and from all packet rows to the local naming row;*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompleteness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitObserver}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no selected host limit, no host real equality, no hidden observer budget, no alternate tolerance ledger, and no ambient completeness coordinate.

Lean4 (definition) `BEDC.Derived.DiagonalLimitObserverUp.DiagonalLimitObserverUp`

Theorem 961.2 (Diagonal-limit observer NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitObserver}^{\uparrow}}$ *surface for Definition 961.1 consists of five finite obligation rows:*

1. *carrier admission for $O = (D, W, Q, T, S, R, H, C, P, N)$;*
2. *observer classifier stability under the displayed* hsame *transports and* Cont *routes;*
3. *window-first exactness stating that no accepted observer read reaches Q, T, S, or R before the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W is displayed;*
4. *tolerance-readback exactness forcing the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T to share the same window row W;*
5. *seal-route non-escape stating that the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route S and the* $\mathsf{Real}^{\uparrow}$ *seal row R are reached only after D, W, Q, T, H, C, P, N have been displayed.*

Proof. Carrier admission is projection from the finite packet in Definition 961.1. The tuple lists the diagonal packet, finite windows, regular readback, dyadic tolerance ledger, Cauchy-limit seal, Real seal, transports, routes, provenance, and local name.

Classifier stability uses the componentwise rows H and C. These rows preserve the same diagonal source, stream-window ledger, readback row, tolerance ledger, seal route, Real seal, package row, and local certificate under observer-facing transport.

Window-first and tolerance-readback exactness are read from the order and routes of O. The only routes to Q and T start from the displayed W row, and the only route to S consumes both Q and T.

Finally, seal-route non-escape follows because R is reached only through S, with P and N recording the provenance and local certificate. Since the carrier lacks quotient streams, selected host limits, host real equality, hidden observer budgets, alternate tolerance ledgers, and ambient completeness, no outside row can supply an accepted seal read. □

Lean4 ✓ `BEDC.Derived.DiagonalLimitObserverUp.DiagonalLimitObserver namecert obligations`

Theorem 961.3 (Diagonal-limit observer seal exposure)**.** *Every accepted observer read of a* $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *packet*

$$O = (D, W, Q, T, S, R, H, C, P, N)$$

that reaches the $\mathsf{Real}^{\uparrow}$ *seal row R first exposes the finite window row W and the dyadic tolerance ledger T. More precisely, the observer-facing route factors through*

$$D \longrightarrow W \longrightarrow Q, T \longrightarrow S \longrightarrow R$$

with the displayed hsame*,* Cont*,* Pkg*, and* NameCert *support rows. Thus an observer route that hides W or T is not an accepted read of the carrier.*

Proof. Project the carrier and inspect the displayed continuation rows. The only route from the diagonal packet to the readback and tolerance surface starts at W.

The readback row Q and tolerance ledger T are attached to the same finite windows. The seal route S consumes both rows before the Real seal row R can be exposed.

Apply the local NameCert obligations. They keep the route inside H, C, P, N and exclude hidden observer budgets, alternate tolerance ledgers, quotient streams, host real equality, selected limits, and ambient completeness. □

Theorem 961.4 (Diagonal-limit observer classifier transport)**.** *Let*

$$O = (D, W, Q, T, S, R, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *packet. Any classifier-facing presentation accepted by the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitObserver}^{\uparrow}}$ *transports the same diagonal packet, StreamName window ledger, RegSeqRat readback row, DyadicRatCore tolerance ledger, CauchyLimitSeal route, RealUp seal, package provenance, and local name componentwise along the displayed* hsame *rows in H and the displayed* Cont *rows in C. The transport introduces no quotient stream, host real equality, hidden observer budget, alternate tolerance ledger, or ambient completeness coordinate.*

Proof. Project the accepted carrier to its displayed packet rows. The classifier has access only to $D, W, Q, T, S, R, H, C, P, N$, because the carrier definition contains no further observer-facing coordinate.

Use the classifier-stability row of Definition 961.2. That row reads the componentwise transports in H together with the continuation routes in C, so the classifier presentation preserves the displayed diagonal, window, readback, tolerance, seal, Real, provenance, and local-name rows.

The exclusion clauses are inherited from the carrier. Since quotient streams, host real equality, hidden observer budgets, alternate tolerance ledgers, and ambient completeness coordinates are absent from O, no classifier transport can acquire such a row while remaining accepted by the same local certificate. □

Theorem 961.5 (Diagonal-limit observer tolerance readback coherence)**.** *Let*

$$O = (D, W, Q, T, S, R, H, C, P, N)$$

be an accepted $\mathsf{DiagonalLimitObserver}^{\uparrow}$ *packet. Every accepted observer read that exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, or reaches the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route S and* $\mathsf{Real}^{\uparrow}$ *seal R does so over the same finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger W. Thus readback and tolerance cannot split into two accepted observer routes.*

Proof. Project the packet and inspect the routes stored in C. The displayed route to Q starts from W, and the displayed route to T starts from that same window ledger.

Apply the tolerance-readback exactness row in Definition 961.2. It forces the readback row and the tolerance ledger to share the same StreamName window surface rather than two independent observer windows.

For reads that reach S or R, use Definition 961.3. The seal route consumes both Q and T before exposing R, so the same window row W remains the unique accepted source for readback and tolerance. □

Theorem 961.6 (Diagonal-limit observer public export scope)**.** *Every public Real-and-Completion consumer of an accepted* DiagonalLimitObserver$^{\uparrow}$ *packet may export only the ordered observer route*

$$D \longrightarrow W \longrightarrow Q, T \longrightarrow S \longrightarrow R$$

together with the support rows H, C, P, N*. The exported route may cite the finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* CauchyLimitSeal$^{\uparrow}$ *route, and* Real$^{\uparrow}$ *seal rows displayed by the packet. It cannot export a quotient stream, host real equality, selected host limit, hidden observer budget, alternate tolerance ledger, or ambient completeness principle.*

Proof. Apply Definition 961.2 to read the finite carrier inventory. The public consumer receives only the displayed packet rows $D, W, Q, T, S, R, H, C, P, N$ and the local certificate obligations attached to them.

Next use Definition 961.3. Any accepted route that reaches the Real$^{\uparrow}$ seal row must expose W and T before R, hence its public spine factors through $D \to W \to Q, T \to S \to R$ rather than through an undisplayed host object.

Apply Definition 961.4 to preserve the same rows under the componentwise hsame and Cont transports. Then apply Definition 961.5 to keep the RegSeqRat$^{\uparrow}$ readback row and DyadicRatCore$^{\uparrow}$ tolerance ledger over one finite StreamName$^{\uparrow}$ window ledger.

The forbidden exports are absent from the carrier and from all accepted classifier transports. Therefore the public Real-and-Completion surface consists exactly of the ordered observer route together with H, C, P, N and the cited finite certificate rows. □

The Lean TasteGate surface records Nontrivial and FieldFaithful instances for the packet fields.

Closure status for NameCert$_{\text{DiagonalLimitObserver}^{\uparrow}}$**.**

Constructive story: A DiagonalLimitObserver$^{\uparrow}$ packet is generated from a diagonal RealCompleteness$^{\uparrow}$ source row, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance ledger, CauchyLimitSeal$^{\uparrow}$ route, Real$^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DiagonalLimitObserver}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 961.1, Definition 961.2, Definition 961.3, Definition 961.4, Definition 961.5, and Definition 961.6 bind the observer-facing diagonal route to finite StreamName windows, RegSeqRat readback, Dyadic tolerance, CauchyLimitSeal, RealUp seal, transport, route, provenance, NameCert rows, and the public Real-and-Completion export surface.

Formal verification: FormalStatus(NameCert$_{\text{DiagonalLimitObserver}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DiagonalLimitObserverUp.DiagonalLimitObserverTasteGate\_single\_carrier\_alignment` Lean4 ✓ `BEDC.Derived.DiagonalLimitObserverUp.taste gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, quotient streams, selected host limits, host real equality, hidden observer budgets, alternate tolerance ledgers, or ambient completeness for DiagonalLimitObserver$^{\uparrow}$.

Upgrade path: No further paper-axis grade remains for this observer package; Lean verification and bridge checking stay on the separate verification axis.

Chapter 962

A Concrete Naming Certificate for $\mathsf{TotalHostFuelHandoff}^{\uparrow}$

The $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ packet is the finite handoff object between the total host of chapter 9370 and the universal substrate traces represented by chapter 944. It is narrower than $\mathsf{StepIndexedTotalHost}^{\uparrow}$: it does not package the whole host normal-form comparison surface, but instead fixes the boundary at which a total host supplies finite fuel, the substrate displays the bounded trace, and the terminal readback returns through the same fuel certificate.

Falsifiable prediction: Removing the fuel handoff row permits a terminal substrate read without a displayed total-host fuel boundary.

Independence witness: znormal, bounded-universal-trace, cellularautomaton, turingmachine, continuationtrace: none is bijective because the handoff packet carries the fuel boundary between total host and substrate trace rather than only normality, trace, substrate, machine, or continuation rows.

Definition 962.1 (Total-host fuel handoff carrier)**.** *A* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, F, S, T, R, B, X, C, P, N)$$

with the following displayed rows:

1. *H is the total-host source row whose public read is accepted only at a finite boundary;*
2. *F is the fuel handoff row naming the host-visible finite budget supplied to the substrate read;*
3. *S is the substrate row, drawn from a displayed* $\mathsf{CellularAutomaton}^{\uparrow}$ *or* $\mathsf{TuringMachine}^{\uparrow}$ *interface;*
4. *T is the finite trace row, read as a* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger below the fuel boundary F;*
5. *R is the terminal readback row selected from T under the same fuel boundary;*
6. *B is the refusal ledger for missing fuel, detached terminal readback, unbounded halting, partial evaluator transfer, host-substrate equality, and quotient behaviour;*
7. *X records componentwise* hsame *transports for H, F, S, T, R, B, P, N;*
8. *C records displayed* Cont *routes from H to F, from F to S, T, from T to R, and from every excluded semantic claim to B;*
9. *P is the* Pkg *provenance over* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{TuringMachine}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{TotalHostFuelHandoff}^{\uparrow}}$ *row.*
The local classifier compares accepted packets only through $H, F, S, T, R, B, X, C, P, N$. It has no coordinate for an infinite substrate run, a host partial evaluator, a global halting predicate, host equality with the substrate, quotient stream behaviour, or a terminal readback detached from the finite fuel handoff.

Lean4 (definition) `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffUp`

Theorem 962.2 (TotalHostFuelHandoff NameCert obligations)**.** *For the carrier of Definition 962.1, the local* $\mathsf{NameCert}_{\mathsf{TotalHostFuelHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (H, F, S, T, R, B, X, C, P, N),$$

total-host exposure through H, finite fuel handoff through F, substrate exposure through S, bounded trace exposure through T, terminal readback through R, refusal-ledger exactness through B, componentwise transport through X, displayed continuation replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier of Definition 962.1. The accepted packet displays exactly $H, F, S, T, R, B, X, C, P, N$, so every public read is either one of those rows or a route replaying them.

The computational route is finite. The host row H reaches the substrate only through the fuel row F, the substrate and trace rows S, T are read below that same boundary, and the terminal readback R is selected from the displayed trace.

The structural rows bind the public certificate. The transports X, routes C, provenance P, and local naming row N keep the handoff inside BHist, hsame, Cont, Pkg, and NameCert.

Since the carrier has no further coordinate, unbounded halting, partial evaluator transfer, host-substrate equality, quotient behaviour, and detached terminal readback are refused by B rather than exported as NameCert obligations. □

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffNameCertObligations`

Theorem 962.3 (TotalHostFuelHandoff carrier admission)**.** *An accepted* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ *handoff packet may expose a terminal readback only after the carrier has displayed exactly*

$$H, F, S, T, R, B, X, C, P, N.$$

The public terminal readback row R is therefore admitted only as the row selected from the substrate trace T under the same finite fuel boundary F, with the refusal ledger B, transports X, route row C, provenance P, and local name N still present.

Proof. Project the carrier in Definition 962.1. Its coordinates are the total-host source H, fuel boundary F, substrate row S, finite trace T, terminal readback R, refusal ledger B, transport row X, route row C, provenance row P, and local name N.

Read the terminal row through the displayed Cont route. The route reaches R only after H has supplied F and after S, T have been read below that same boundary.

The remaining rows make the admission public rather than semantic. The refusal ledger records missing-fuel and detached-readback failures, while X, P, N keep the packet inside the local BHist, Pkg, and NameCert surface. □

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffCarrierAdmission`

Theorem 962.4 (TotalHostFuelHandoff fuel monotonicity)**.** *Replacing the displayed fuel row F of an accepted* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ *packet by a smaller displayed budget can only shorten the substrate trace and terminal readback route. It cannot create a new terminal readback row outside*

$$H, F, S, T, R, B, X, C, P, N,$$

and it cannot turn an excluded unbounded halting, partial evaluator, host-substrate equality, quotient behaviour, or detached terminal-readback claim into a public output.

Proof. Use the displayed route row C. Lowering the fuel budget restricts the part of the substrate trace T that may be replayed before the terminal readback row is selected.

The carrier has no coordinate for inventing a readback after this restriction. Any public read must still pass through $H, F, S, T, R, B, X, C, P, N$, so a smaller F can remove available trace entries but cannot add a new row.

Finally retain the refusal ledger B. Claims requiring more fuel, an unbounded run, a partial evaluator, a host-substrate equality coordinate, quotient behaviour, or a detached terminal oracle remain in the refused side of the packet. □

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffFuelMonotonicity`

Theorem 962.5 (TotalHostFuelHandoff non-escape)**.** *Every accepted* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ *packet exports no terminal substrate read except through the finite route*

$$H, F, S, T, R, B, X, C, P, N.$$

In particular, it exports no unbounded halting predicate, no host partial evaluator, no equality between host computation and substrate trace, no quotient behaviour, no independent terminal readback oracle, and no transfer of substrate universality into the total host.

Proof. Begin with the displayed route row C. Its admissible paths pass from the total-host source H to the fuel handoff F, then through the substrate row S and finite trace ledger T, and only then to the terminal readback R.

The fuel row is the boundary object. Without F, the packet may still mention a substrate or a terminal shape, but it no longer exhibits the finite host-visible budget required to read that terminal row in a total host.

The refusal ledger accounts for every excluded claim. Unbounded halting needs a global run predicate, a host partial evaluator needs a host semantic execution row, host-substrate equality needs an equality coordinate, quotient behaviour needs a quotient classifier, and transfer of substrate universality needs a semantic bridge absent from the carrier.

Thus the handoff exposes a finite terminal readback while preserving the separation required by chapter 9370. □

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffNonescape`

Theorem 962.6 (TotalHostFuelHandoff sibling boundary)**.** *The siblings* $\mathsf{ZNormal}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{TuringMachine}^{\uparrow}$, *and* $\mathsf{ContinuationTrace}^{\uparrow}$ *are comparison anchors for* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$, *but none is bijective to it.* $\mathsf{ZNormal}^{\uparrow}$ *stores a host normality boundary without the substrate terminal handoff;* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *stores a bounded trace without the total-host fuel handoff row;* $\mathsf{CellularAutomaton}^{\uparrow}$ *and* $\mathsf{TuringMachine}^{\uparrow}$ *store substrate surfaces without the host boundary; and* $\mathsf{ContinuationTrace}^{\uparrow}$ *stores finite continuation rows without the total-host source and NameCert handoff surface.*

Proof. Read the siblings by their missing coordinates. The normal-form sibling accounts for finite host normality, but it does not carry the substrate row, finite trace ledger, or terminal readback selected below host fuel.

The bounded trace sibling is closest to F, S, T, R. It still does not isolate the handoff row as the boundary between total-host source and substrate trace, nor does it carry the refusal surface for detached terminal readback.

The substrate and continuation siblings point to components rather than the whole handoff. They supply machine, cellular, or finite continuation rows, but they do not package $H, F, S, T, R, B, X, C, P, N$ together as a total-host fuel certificate.

Consequently the $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ packet is the concrete interface object for terminal readback across the total-host and substrate-trace boundary. □

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffSiblingBoundary`

Lean4 ✓ `BEDC.Derived.TotalHostFuelHandoffUp.TotalHostFuelHandoffTasteGate single carrier alignment`

The Lean `lean4/BEDC/Derived/TotalHostFuelHandoffUp/TasteGate.lean` development must include explicit $\mathsf{Nontrivial\ TotalHostFuelHandoff}^{\uparrow}$ and $\mathsf{FieldFaithful\ TotalHostFuelHandoff}^{\uparrow}$ instances. These instances must preserve the total-host source row, fuel handoff row, substrate row, finite trace row, terminal readback row, refusal ledger, componentwise transports, displayed continuation routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{TotalHostFuelHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{TotalHostFuelHandoff}^{\uparrow}$ packet is generated from a total-host source row, finite fuel handoff row, substrate row, finite trace row, terminal readback row, refusal ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalHostFuelHandoff}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 962.1, Definition 962.2, Definition 962.5, Definition 962.6, and the checked TasteGate marker specify the finite fuel handoff packet and its NameCert obligation surface between total host and substrate trace.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalHostFuelHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, partial evaluator semantics, host-substrate equality, quotient behaviour, or transfer of substrate universality to the total host.

Upgrade path: Obligation closure requires checked carrier admission, fuel handoff exposure, substrate-trace exposure, terminal-readback exactness, refusal-ledger exactness, nontriviality, and field faithfulness for the same packet.

Chapter 963

A Concrete Naming Certificate for FuelBoundedPartialTrace$^{\uparrow}$

The FuelBoundedPartialTrace$^{\uparrow}$ packet is the finite NameCert realization of the total-host versus universal-substrate boundary of chapter 9370. It takes the step-indexed computation reading of `r110_run_n_steps`, exposes the fuel row as public data, and records only the bounded trace cells that a total host can inspect. The packet sits beside BoundedUniversalTrace$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, ZNormal$^{\uparrow}$, TypedReductionNormalForm$^{\uparrow}$, and MetaCICNormalizationBudget$^{\uparrow}$; it is not a halting claim, a partial evaluator, a quotient trace, or an ambient universality surface.

Falsifiable prediction: A trace read without the displayed fuel row is not accepted by FuelBoundedPartialTrace$^{\uparrow}$.

Independence witness: boundeduniversaltrace, cellularautomaton, znormal, typedreduction-normalform, metacicnormalizationbudget: none is bijective to FuelBoundedPartialTrace$^{\uparrow}$ because the packet binds a fuel row, substrate row, finite prefix trace, readback row, and total-host refusal row in one carrier, while each sibling omits at least one of those coordinates or uses a different classifier surface.

FuelBoundedPartialTrace$^{\uparrow}$ consumes chapter 944 as the larger total-host bounded-trace surface and chapter 469 as the finite prefix ledger. Its own contribution is the partial-trace fuel and refusal specialization over those sibling surfaces, not a standalone substrate.

Definition 963.1 (FuelBoundedPartialTrace carrier)**.** *A* FuelBoundedPartialTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, S, I, R, O, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the fuel row naming the finite step budget used by the host read;*
2. *S is the substrate row, such as the* CellularAutomaton$^{\uparrow}$ *Rule 110 local-rule surface or another displayed universal-substrate row;*
3. *I is the initial-data row for the bounded run;*
4. *R is the finite trace-prefix row containing only states admitted below the budget F;*
5. *O is the readback row selected from the displayed prefix R;*
6. *B is the total-host refusal row stating that reads outside the displayed budget are absent rather than negative halting facts;*

7. *H records componentwise* hsame *transports for F, S, I, R, O, B, P, N;*
8. *C records displayed* Cont *routes through the step-indexed trace prefix, readback row, and refusal boundary;*
9. *P is the* Pkg *provenance over* $\mathsf{FuelBoundedPartialTrace}^{\uparrow}$*, chapter 9370,* $\mathsf{BoundedUniversalTrace}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{ZNormal}^{\uparrow}$*,* $\mathsf{TypedReductionNormalForm}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationBudget}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FuelBoundedPartialTrace}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $F, S, I, R, O, B, H, C, P, N$. It has no coordinate for an unbounded halting predicate, a partial host evaluator, a choice of infinite nontermination evidence, quotient equality of traces, or a universal-substrate semantic completion.

```
↪ variant BEDC.Derived.FuelBoundedPartialTraceUp.FuelBoundedPartialTraceTasteGate single carrier alignment
```

Theorem 963.2 (FuelBoundedPartialTrace NameCert obligations)**.** *For the carrier of Definition 963.1, the local* $\mathsf{NameCert}_{\mathsf{FuelBoundedPartialTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (F, S, I, R, O, B, H, C, P, N)$, fuel-indexed trace exposure through F, S, I, R, readback accountability through O, total-host refusal through B, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 963.1. The fuel, substrate, initial-data, trace-prefix, readback, refusal, transport, route, provenance, and naming rows are the only public coordinates.

The trace side is read through F, S, I, R. A consumer may inspect a finite prefix produced below the displayed budget, but the packet contains no coordinate for an unbounded run or a host-level partial evaluator.

The readback and refusal side is read through O and B. The selected row O must come from the displayed prefix R, while B records that an out-of-budget read is absent at this certificate surface rather than represented as a halting or nonhalting fact.

Finally H, C, P, N bind the NameCert surface. Since the excluded semantic objects have no coordinate in the packet, the listed rows exhaust the obligations. □

Theorem 963.3 (FuelBoundedPartialTrace total-host refusal)**.** *An accepted* $\mathsf{FuelBoundedPartialTrace}^{\uparrow}$ *packet admits a public trace read only when the read factors through the displayed fuel row F and finite trace prefix R. A consumer that omits F, replaces R by an unbounded substrate run, or asks for a row beyond the displayed budget reaches the refusal row B and obtains no halting predicate, no partial evaluator result, no quotient trace equality, and no ambient universality claim.*

Proof. Begin with the carrier coordinates. The only computation-facing rows are the budget F, substrate S, initial data I, trace prefix R, readback row O, and refusal row B.

If a read is in budget, the route row C sends it through F and the displayed prefix R before reaching O. This is a finite step-indexed read of the kind described in Definition 9370.5.

If the read omits the budget or asks outside the displayed prefix, the packet has no terminal coordinate to expose. The refusal row B is therefore the only public boundary, and it refuses by absence of a displayed finite row rather than by importing halting, partiality, quotient equality, or universal-substrate semantics. □

The Lean follow-up for BEDC.Derived.FuelBoundedPartialTraceUp will provide Nontrivial FuelBoundedPartialTrace and FieldFaithful $\mathsf{FuelBoundedPartialTrace}^{\uparrow}$ instances for the displayed carrier. Those instances must

preserve the fuel row, substrate row, initial-data row, trace-prefix row, readback row, refusal row, componentwise transports, replay routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{FuelBoundedPartialTrace}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{FuelBoundedPartialTrace}^{\uparrow}$ packet is generated from a fuel row, substrate row, initial-data row, finite trace-prefix row, readback row, total-host refusal row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FuelBoundedPartialTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 963.1, Definition 963.2, and Definition 963.3 specify the finite fuel-bounded trace packet for total-host reads of universal-substrate computations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FuelBoundedPartialTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, partial evaluator semantics, quotient trace equality, selected nontermination evidence, or universal-substrate semantic completion.

Upgrade path: Obligation closure requires checked carrier admission, fuel-indexed trace exposure, readback accountability, total-host refusal, nontriviality, and field faithfulness for the same packet.

Chapter 964

A Concrete Naming Certificate for $\mathsf{StepIndexedTotalHost}^{\uparrow}$

The $\mathsf{StepIndexedTotalHost}^{\uparrow}$ packet is the concrete NameCert object for the boundary between a total host and a universal substrate in chapter 9370. It sits beside the Turing-machine substrate of chapter 280, the bounded universal trace of chapter 944, the normal-form host packet of chapter 936, and the closed-normal consistency boundary of chapter 733. Its distinguishing datum is the simultaneous presence of a total host row, a unary fuel row, a finite universal-substrate trace row, and a refusal ledger preventing an unbounded substrate run from being reread as host computation.

Falsifiable prediction: Deleting the fuel row permits an unbounded substrate trace to masquerade as host computation; the local classifier must reject that packet because the host-side read no longer passes through a displayed finite fuel index.

Independence witness: turingmachine, boundeduniversaltrace, znormal, closednormalconsistency: none is bijective because each sibling omits at least one of the fuel-indexed host boundary, the substrate trace, or the refusal ledger.

Definition 964.1 (StepIndexedTotalHost carrier)**.** *A* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, F, R, N, Z, B, Hs, C, P, K)$$

with the following displayed rows:

1. *A is the total host term or evaluator row, read as a BEDC host computation whose reduction is accepted only through finite data;*
2. *F is the unary fuel row naming the finite host-visible step index;*
3. *R is the finite substrate trace row, such as a Rule 110 row produced by* `r110_run_n_steps` *or a Turing-machine trace row with the same displayed bound;*
4. *N is the host normal-form boundary row, read through the finite normal-form discipline of* $\mathsf{ZNormal}^{\uparrow}$*;*
5. *Z is the bounded universal trace witness row, read through* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *and tied to the same fuel row F;*
6. *B is the refusal ledger for halting transfer, partial evaluator transfer, missing fuel, detached substrate trace, and host-equality claims not carried by the packet;*

7. *Hs records componentwise* hsame *transports for A, F, R, N, Z, B, P, K;*
8. *C records displayed* Cont *routes from host row to fuel, from fuel to finite substrate trace, from trace to bounded readback, from bounded readback to normal-form boundary, and from each absent semantic claim to the refusal ledger;*
9. *P is the* Pkg *provenance over* StepIndexedTotalHost$^{\uparrow}$, TuringMachine$^{\uparrow}$, BoundedUniversalTrace$^{\uparrow}$, ZNormal$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *K is the local* NameCert$_{\text{StepIndexedTotalHost}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $A, F, R, N, Z, B, Hs, C, P, K$. It has no coordinate for unbounded halting, an infinite Rule 110 orbit, host evaluator equality, quotient behaviour, or a proof that substrate universality transfers to the total host.

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostTasteGate single carrier alignment`

Theorem 964.2 (StepIndexedTotalHost NameCert obligations)**.** *For the carrier of Definition 964.1, the local* NameCert$_{\text{StepIndexedTotalHost}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$H = (A, F, R, N, Z, B, Hs, C, P, K),$$

total-host exposure through A, unary fuel exposure through F, finite substrate-trace exposure through R, host normal-form boundary exposure through N, bounded universal trace witness exposure through Z, refusal-ledger exactness through B, componentwise transport through Hs, displayed continuation replay through C, package provenance through P, and local naming through K.

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier namecert obligations`

Proof. Project the carrier of Definition 964.1. Its visible rows are exactly $A, F, R, N, Z, B, Hs, C, P, K$, so every public read must be a projection from those rows or a route replaying them.

The host-facing side is read through A, F, N. The row A names the total host term or evaluator row, F supplies the finite fuel index, and N records the normal-form boundary available to the host.

The substrate-facing side is read through R, Z. The trace row R records a finite Rule 110 or Turing-machine prefix, while Z ties that prefix to the bounded universal trace witness at the same fuel index.

Finally B, Hs, C, P, K bind the refusal and NameCert surface. Since the carrier has no coordinate for unbounded halting, infinite orbit data, host evaluator equality, quotient behaviour, or transfer of substrate universality to host totality, the listed rows exhaust the obligations. □

Theorem 964.3 (StepIndexedTotalHost separation)**.** *Every accepted* StepIndexedTotalHost$^{\uparrow}$ *packet separates the total-host read from the universal-substrate run by the finite route*

$$A, F, R, Z, N, B, Hs, C, P, K.$$

A consumer may read a host computation at fuel F, a finite substrate trace R, a bounded witness Z, and a normal-form boundary N. It may not read an unbounded substrate trace as host computation, transfer halting or partial evaluator behaviour across the boundary, or replace the refusal ledger B by a semantic equality between the host and the substrate.

Proof. Begin with the finite route row C. Its admissible paths pass from the host row A to the fuel row F, then to the displayed substrate trace R and bounded witness Z, and only then to the host normal-form boundary N.

The fuel row is essential. If F is absent, the packet still may display a substrate trace schema, but the host has no finite step index under which to read it as a total computation. The refusal ledger B therefore rejects the packet rather than accepting an unbounded run as host data.

The same ledger blocks semantic transfer. Halting transfer, partial evaluator transfer, infinite orbit claims, host evaluator equality, and quotient behaviour require coordinates absent from $A, F, R, N, Z, B, Hs, C, P, K$.

Thus the packet preserves both sides of chapter 9370: the substrate may carry universal finite prefixes, while the host reads only fuel-indexed finite data. □

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier separation`

Theorem 964.4 (StepIndexedTotalHost sibling boundary)**.** *The siblings* $\mathsf{TuringMachine}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{ZNormal}^{\uparrow}$, *and the closed-normal consistency boundary are comparison anchors for* $\mathsf{StepIndexedTotalHost}^{\uparrow}$, *but none is bijective to it.* $\mathsf{TuringMachine}^{\uparrow}$ *supplies machine substrate rows without the host normal-form boundary;* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *supplies a bounded trace without the full total-host refusal surface;* $\mathsf{ZNormal}^{\uparrow}$ *supplies host normality without the universal-substrate trace; and the closed-normal consistency boundary supplies a normal-form consistency node without the step-indexed substrate witness.*

Proof. Read the siblings one at a time. The Turing-machine surface names finite machine data, but it does not carry the host row A, the normal-form boundary N, or the refusal ledger B for total-host transfer.

The bounded universal trace supplies the rows closest to F, R, Z. It still lacks the full packet route from host term to normal-form boundary and the refusal ledger blocking host-substrate semantic identification.

The normal-form and closed-normal consistency siblings point in the other direction. They account for finite normality and consistency-facing boundaries, but they do not carry the Rule 110 or Turing-machine substrate trace tied to the same fuel row.

Consequently each sibling can be cited as a component or comparison anchor, while the $\mathsf{StepIndexedTotalHost}^{\uparrow}$ packet remains the concrete boundary object that stores all rows together. □

Theorem 964.5 (StepIndexedTotalHost obligation surface)**.** *The fuel-indexed total-host packet has exactly the obligations of total-host row exposure through A, unary fuel exposure through F, finite substrate trace exposure through R, normal-form boundary exposure through N, bounded universal trace witness exposure through Z, refusal-ledger exactness through B, componentwise* hsame *transport through Hs, displayed* Cont *replay through C,* Pkg *provenance through P, and local* $\mathsf{NameCert}_{\mathsf{StepIndexedTotalHost}^{\uparrow}}$ *through K. Every host/substrate comparison must pass through the same fuel row F before any normal-form read or bounded universal trace read is accepted.*

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier obligation surface`

Proof. Start with the carrier and NameCert obligations, Definition 964.1 and Definition 964.2. They list the total-host row, unary fuel row, finite substrate trace, normal-form boundary, bounded universal trace witness, refusal ledger, transport, replay, provenance, and local name as the complete obligation surface.

Use the separation theorem. It routes every host-facing and substrate-facing read through $A, F, R, Z, N, B, Hs, C, P, K$, so the fuel row F is the shared boundary before either the normal-form row N or bounded universal trace row Z can be inspected.

Apply the sibling boundary theorem. The Turing-machine, bounded universal trace, normal-form, and closed-normal consistency siblings can each provide comparison data, but none replaces the displayed fuel row or the refusal ledger that keeps host and substrate readings separated. □

The Lean `TasteGate.lean` development for $\mathsf{StepIndexedTotalHost}^{\uparrow}$ must include explicit $\mathsf{Nontrivial\ StepIndexedT}$ and $\mathsf{FieldFaithful\ StepIndexedTotalHost}^{\uparrow}$ instances. These instances must preserve the total host row, unary fuel row, finite substrate trace, normal-form boundary, bounded universal trace witness, refusal ledger, componentwise transports, displayed routes, package provenance, and local naming row.

Theorem 964.6 (StepIndexedTotalHost scoped-kernel route)**.** *Every accepted* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *scoped-kernel read factors through* BHist, hsame, Cont, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{TypedReductionNormalForm}^{\uparrow}$, *the displayed* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *carrier rows,* Pkg *provenance, and* $\mathsf{NameCert}$*. The route is finite: it exposes carrier admission, fuel-indexed trace rows, host normal-form boundary, bounded universal trace witness, refusal-ledger exactness, nontriviality, and field faithfulness for the same packet, but it does not assert host evaluator equality, kernel consistency, unbounded halting transfer, or substrate universality for the total host.*

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier scoped kernel route`

Proof. Start with the carrier and NameCert obligations, Definition 964.1 and Definition 964.2. They expose the total host row, unary fuel row, finite substrate trace, normal-form boundary, bounded universal trace witness, refusal ledger, componentwise transports, route replay, provenance, and local naming row.

Apply separation and the sibling boundary theorem, Definition 964.3 and Definition 964.4. These route the substrate side through $\mathsf{BoundedUniversalTrace}^{\uparrow}$ and the host boundary through $\mathsf{TypedReductionNormalForm}^{\uparrow}$ before either side can be compared.

Finally use the obligation surface theorem. It keeps the packet inside the same finite $A, F, R, N, Z, B, Hs, C, P, K$ rows, with nontriviality and field faithfulness read as obligations of that displayed carrier rather than as ambient semantic equalities. Hence the scoped route has no coordinate for host evaluator equality, kernel consistency, unbounded halting transfer, or total-host universality. □

Theorem 964.7 (StepIndexedTotalHost public interface)**.** *The exported public interface of an accepted* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *packet is exactly the fuel-indexed host/substrate boundary carried by* $A, F, R, N, Z, B, Hs, C, P, K$*. A public consumer may read the total-host row, the shared unary fuel row, the finite substrate trace, the host normal-form boundary, the bounded universal trace witness, the refusal ledger, and the displayed transport, replay, provenance, and local naming rows. The interface does not export host evaluator equality, kernel consistency, unbounded halting transfer, partial evaluator semantics, quotient behaviour, or substrate universality for the total host.*

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier public interface`

Proof. Start with the scoped-kernel route of Definition 964.6. It already factors every accepted read through the carrier rows, the bounded universal trace boundary, the normal-form host boundary, and the local NameCert surface.

Apply the obligation surface and separation theorems, Definition 964.5 and Definition 964.3. These identify $A, F, R, N, Z, B, Hs, C, P, K$ as the only rows through which a host/substrate comparison may pass, with F as the shared finite boundary.

Finally use the sibling boundary of Definition 964.4. The Turing-machine, bounded-trace, normal-form, and closed-normal siblings can be cited as anchors, but none supplies the missing semantic endpoints. Hence the public interface exports the displayed finite packet and nothing beyond it. □

Theorem 964.8 (StepIndexedTotalHost bridge interface)**.** *The public fuel-indexed* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *interface bridges the total-host read and the bounded universal trace read exactly through*

$$A, F, R, N, Z, B, Hs, C, P, K.$$

A consumer may transport the host row A to the shared fuel row F, replay the finite substrate trace R, read the bounded universal witness Z, and return to the normal-form host boundary N. The bridge does not export unbounded halting, partial evaluator transfer, host evaluator equality, quotient behaviour, or a proof that substrate universality is a total-host property.

Lean4 ✓ `BEDC.Derived.StepIndexedTotalHostUp.StepIndexedTotalHostCarrier bridge interface`

Proof. Start with the public interface theorem, Definition 964.7. It exposes exactly the rows $A, F, R, N, Z, B, Hs, C, P, K$ and no semantic endpoint outside the finite packet.

Use the separation and obligation-surface theorems, Definition 964.3 and Definition 964.5. They force host/substrate comparison through the shared fuel row F before either the bounded universal trace witness Z or the normal-form boundary N is read.

Finally apply the sibling boundary. The Turing-machine, bounded universal trace, normal-form, and closed-normal consistency chapters provide bridge anchors, while the refusal ledger B blocks the absent unbounded and equality claims. Hence the public interface is a bridge between the displayed finite reads, not an export of stronger host or substrate semantics. □

Closure status for $\mathsf{NameCert}_{\mathsf{StepIndexedTotalHost}^{\uparrow}}$.

Constructive story: A $\mathsf{StepIndexedTotalHost}^{\uparrow}$ packet is generated from a total host term or evaluator row, unary fuel row, finite substrate trace row, normal-form boundary row, bounded universal trace witness row, refusal ledger, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StepIndexedTotalHost}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 964.1, Definition 964.2, Definition 964.3, Definition 964.4, Definition 964.5, Definition 964.6, Definition 964.7, and Definition 964.8 specify the mature fuel-indexed total-host and universal-substrate boundary through the checked bridge-interface theorem.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StepIndexedTotalHost}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, partial evaluator semantics, host evaluator equality, quotient behaviour, or transfer of substrate universality to the total host.

Upgrade path: The mature paper scope is the fuel-indexed total-host/substrate boundary through the checked bridge interface; Lean verification remains an independent formal-axis obligation.

Chapter 965

A Concrete Naming Certificate for $\mathsf{PromptResponseTrace}^{\uparrow}$

The $\mathsf{PromptResponseTrace}^{\uparrow}$ packet realizes the large-model prompt and output record as finite BEDC data. It sits between chapter 939 and chapter 928: the former supplies the inference-substrate and corpus-supply reading, while the latter supplies the public observer-facing interface. The packet also keeps the obstruction boundary of chapter 490 visible, so a prompt-output record is not read as a global decision or internally ratified universal claim.

Definition 965.1 (PromptResponseTrace carrier)**.** *A* $\mathsf{PromptResponseTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, C, P, O, A, F, R, H, K, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *row naming the finite inference substrate, corpus-supply channel, output-record surface, and audit-refusal boundary;*
2. *C is the source-channel row recording the finite corpus-supply and prompt-entry provenance used by the trace;*
3. *P is the prompt row, a finite inscription request read through the displayed source channel;*
4. *O is the output row, a finite response inscription emitted after P along the recorded model route;*
5. *A is the audit-refusal row stating that the packet does not internally ratify universal closure, host semantic truth, consciousness, or model-side proof of the output;*
6. *F is the* $\mathsf{ObserverInterface}^{\uparrow}$ *row through which public observers may read the prompt, response, source channel, and refusal boundary;*
7. *R is the* $\mathsf{HaltingObstruction}^{\uparrow}$ *boundary row, recording that the prompt-response trace is not a global halting or decision oracle;*
8. *H records componentwise* hsame *transport for the model row, source-channel row, prompt row, output row, audit-refusal row, observer-interface row, obstruction boundary, route row, provenance row, and local name;*
9. *K records the displayed* Cont *routes by which consumers read C, then P, then O, then A, then F and R;*

10. *G is the* Pkg *provenance over* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{ObserverInterface}^{\uparrow}$, $\mathsf{HaltingObstruction}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PromptResponseTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden model intention, corpus-wide truth, host semantic understanding, quotient identity of outputs, global halting classification, or internally certified universal closure.

Lean4 ✓ `BEDC.Derived.PromptResponseTraceUp.PromptResponseTraceTasteGate single carrier alignment`

Theorem 965.2 (PromptResponseTrace NameCert obligations)**.** *For the carrier of Definition 965.1, the local* $\mathsf{NameCert}_{\mathsf{PromptResponseTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for* $T = (M, C, P, O, A, F, R, H, K$ *source-channel exposure through C, prompt admission through P, response inscription through O, audit-refusal exposure through A, observer-interface handoff through F, halting-obstruction boundary through R, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through G, and local naming through N.*

Proof. Project the displayed carrier. The rows $M, C, P, O, A, F, R, H, K, G, N$ list the whole trace before any consumer can read a prompt or response.

The source-channel row is read before the prompt row, and the prompt row is read before the output row. This forces the response to be consumed as carried inscription over a finite channel rather than as a self-ratified theorem of the model.

The audit-refusal, observer-interface, and obstruction rows close the local certificate surface. The refusal row blocks internally ratified universal closure, the interface row gives the public observation route, and the obstruction row prevents the trace from becoming a global decision oracle.

The remaining rows provide only transport, route replay, provenance, and local naming. Since the carrier has no further coordinate, the listed obligations exhaust the NameCert surface. □

Theorem 965.3 (PromptResponseTrace audit refusal)**.** *Every public read of an accepted* $\mathsf{PromptResponseTrace}^{\uparrow}$ *packet must pass through the audit-refusal row A after the prompt-output rows P, O and before the local naming row N. Removing A permits the prompt-response packet to be read as an internally ratified universal claim; retaining A confines the packet to finite source-channel, prompt, response, observer-interface, and obstruction-boundary rows.*

Proof. Begin with Definition 965.1. The only route from the prompt and response rows to the local name passes through the displayed continuation row K, and K lists the audit-refusal row before the public handoff reaches N.

Apply Definition 965.2. The local certificate surface requires audit-refusal exposure as one of the rows of any accepted prompt-response trace.

If A is removed, the route still contains P and O but no row recording the absence of internal ratification. A consumer could then read the output as if the packet itself certified a universal claim.

With A present, the public read is confined to the finite source-channel, prompt, response, observer-interface, obstruction-boundary, transport, route, provenance, and naming rows. No host semantic truth predicate or model-side proof coordinate is available. □

Theorem 965.4 (PromptResponseTrace vision handoff)**.** *The large-model prompt and output reading of chapter 9398 is realized by the finite* $\mathsf{PromptResponseTrace}^{\uparrow}$ *carrier of Definition 965.1. The handoff reads* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *for the inference substrate and corpus-supply row,*

$\mathsf{ObserverInterface}^{\uparrow}$ *for public trace observation, and* $\mathsf{HaltingObstruction}^{\uparrow}$ *for the finite non-oracle boundary. It exports no hidden substrate oracle, consciousness coordinate, global halting classifier, or internally certified universal closure.*

Proof. The large-model inscription chapter supplies the model-side finite rows: inference substrate, corpus-supply, output record, and audit refusal. These are read into M, C, O, A rather than into a hidden semantic oracle.

The observer-interface chapter supplies the public reading discipline. The row F records that prompt and response data are exposed through finite observer-facing rows, not through an unledgered subject or hidden observation channel.

The halting-obstruction chapter supplies the non-oracle boundary. The row R records that the trace cannot decide arbitrary continuation chains or act as a global halting classifier.

Combining these rows with H, K, G, N gives a finite NameCert handoff for prompt-response traces. The carrier has no coordinate for consciousness, host semantic understanding, global decision power, or internally certified universal closure, so none is exported by the vision handoff. □

Theorem 965.5 (PromptResponseTrace source-channel exactness)**.** *For the displayed carrier*

$$T = (M, C, P, O, A, F, R, H, K, G, N),$$

the row C is the exact source-channel row of the $\mathsf{PromptResponseTrace}^{\uparrow}$ *packet. Every corpus-supply, prompt-entry, and prompt provenance read used by the trace is routed through C together with the large-model inscription row M, and no hidden corpus oracle, host semantic truth predicate, model intention coordinate, or unrecorded prompt source is available.*

Proof. Project the carrier of Definition 965.1. The row C is the only row named as source channel, and the rows P and O are read as prompt and output inscriptions over that channel.

Apply Definition 965.2. Its source-channel exposure clause places C on the accepted NameCert surface before the prompt and response rows are consumed.

The audit-refusal and vision-handoff theorems keep the same boundary. They expose public observations through F and the non-oracle boundary through R, while the carrier has no coordinate for a hidden corpus oracle or host semantic truth predicate. □

Theorem 965.6 (PromptResponseTrace prompt-output route order)**.** *For the displayed carrier $T = (M, C, P, O, A, F, R, H, K, G, N)$, the continuation row K orders the trace route through C, then P, then O, then A, before the public observer row F and halting-obstruction row R are read. Thus the prompt-response packet is consumed as source-channel input followed by prompt, output, audit refusal, observer handoff, and non-oracle boundary.*

Proof. Read the route row in the carrier definition. It states that K records the displayed Cont routes by which consumers read C, then P, then O, then A, then F and R.

Use the NameCert obligations to make this route part of the accepted certificate surface. The route row is not optional transport; it is the public replay discipline for a named prompt-response trace.

Finally apply the audit-refusal theorem. It requires the audit row A after the prompt-output rows and before the local naming route, so the observer and obstruction rows are read only after the source, prompt, output, and refusal sequence has been displayed. □

Theorem 965.7 (PromptResponseTrace observer-obstruction package)**.** *For the displayed carrier $T = (M, C, P, O, A, F, R, H, K, G, N)$, the rows F and R form the public observer plus non-oracle*

obstruction package. The row F exposes the finite prompt-response trace to public observers, while R records that the same trace is not a global halting classifier, global decision oracle, internally certified universal closure, or model-side proof of the output.

Proof. Begin with the carrier rows. The row F is the $\mathsf{ObserverInterface}^{\uparrow}$ handoff, and the row R is the $\mathsf{HaltingObstruction}^{\uparrow}$ boundary.

Apply the audit-refusal theorem before reading the observer package. It ensures that public access to P and O passes through A, so the output is not presented as an internally ratified universal claim.

Now use the vision handoff. It combines the large-model inscription row, observer-interface row, and halting-obstruction row into the finite trace surface, while excluding consciousness, host semantic understanding, global decision power, and internally certified universal closure. □

Falsifiable prediction: Removing the audit-refusal row permits the prompt-response packet to be read as an internally ratified universal claim, while removing the source-channel row breaks corpus-supply provenance.

Independence witness: Nearest siblings: largemodelinscriptionpoint, observerinterface, inscriptionevent, haltingobstruction, and authoredrecordboundary; none is bijective because this carrier has a prompt-output trace plus an audit-refusal row not present in those surfaces.

The follow-up Lean TasteGate for this chapter must provide `Nontrivial PromptResponseTraceUp` and `FieldFaithful PromptResponseTraceUp` instances for the same finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PromptResponseTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{PromptResponseTrace}^{\uparrow}$ packet is generated from a large-model inscription row, source-channel row, prompt row, output row, audit-refusal row, observer-interface row, halting-obstruction boundary, componentwise transport, displayed continuation, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PromptResponseTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 965.1, Definition 965.2, Definition 965.3, Definition 965.4, Definition 965.5, Definition 965.6, Definition 965.7, the falsifiable prediction, the independence witness,
Lean4 ✓ `BEDC.Derived.PromptResponseTraceUp.taste gate`
, and the checked TasteGate alignment bind the finite prompt-response trace to its carrier, NameCert obligations, audit refusal, vision handoff, source-channel exactness, prompt-output route order, observer-obstruction package, and AI-facing alignment row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PromptResponseTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.PromptResponseTraceUp.PromptResponseTraceTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close corpus truth, model consciousness, host semantic understanding, global halting classification, internally certified universal closure, or bridge checking for $\mathsf{PromptResponseTrace}^{\uparrow}$.

Upgrade path: No paper theory-axis grade remains; Lean verification and bridge checking remain independent obligations.

Chapter 966

A Concrete Naming Certificate for $\mathsf{ObserverResponseAudit}^{\uparrow}$

The $\mathsf{ObserverResponseAudit}^{\uparrow}$ packet packages the observer-facing response boundary as a finite NameCert object. It is downstream of the prompt and output rows of chapter 965 and the licensed public interface of chapter 928. Its role is to bind the prompt, response, observer handoff, audit refusal, inscription event, non-oracle boundary, transport, route replay, provenance, and local naming rows into a single accepted carrier.

Definition 966.1 (Observer-response audit carrier)**.** *An* $\mathsf{ObserverResponseAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, I, M, E, P, O, R, H, K, G, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{PromptResponseTrace}^{\uparrow}$ *row recording the finite prompt, response, source-channel, audit-refusal, observer-interface, and halting-obstruction route.*
2. *I is the* $\mathsf{ObserverInterface}^{\uparrow}$ *row through which public observers may read the accepted prompt and response rows.*
3. *M is the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *row naming the finite large-model inscription substrate and corpus-supply side of the trace.*
4. *E is the* $\mathsf{InscriptionEvent}^{\uparrow}$ *row recording the prompt-entry and response-inscription event as finite accepted rows.*
5. *P is the prompt row read from the prompt-response trace and the inscription event.*
6. *O is the response row read after P through the same displayed route.*
7. *R is the audit-refusal row stating that the packet does not certify host semantic truth, model consciousness, global halting classification, or internally ratified universal closure.*
8. *H records componentwise* hsame *transport for the prompt-response trace, observer interface, large-model inscription row, inscription event, prompt row, response row, refusal row, route row, provenance row, and local name.*
9. *K records displayed* Cont *replay through T, then I, then M and E, then P, O, R, before the local NameCert row is read.*

10. *G is the* Pkg *provenance over* PromptResponseTrace$^{\uparrow}$, ObserverInterface$^{\uparrow}$, LargeModelInscriptionPoint$^{\uparrow}$, InscriptionEvent$^{\uparrow}$, HaltingObstruction$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{ObserverResponseAudit}^{\uparrow}}$ *row.*

The local classifier compares observer-response audit carriers only by these displayed rows. It has no coordinate for hidden model intention, unledgered observer state, host semantic equality of responses, corpus-wide truth, global halting decision, or internally certified universal closure.

Lean4 (definition) `BEDC.Derived.ObserverResponseAuditUp.ObserverResponseAuditUp`
Lean4 ✓ `BEDC.Derived.ObserverResponseAuditUp.ObserverResponseAuditTasteGate single carrier alignment`

Theorem 966.2 (Observer-response audit NameCert obligations)**.** *For the carrier of Definition 966.1, the local* NameCert$_{\text{ObserverResponseAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (T, I, M, E, P, O, R, H, K, G, N)$, prompt-response exposure through T, observer-interface handoff through I, large-model inscription exposure through M, inscription-event exposure through E, prompt and response ordering through P, O, audit-refusal exactness through R, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through G, and local naming through N.*

Proof. Project the displayed carrier. The rows $T, I, M, E, P, O, R, H, K, G, N$ are the only coordinates available to an accepted observer-response audit packet.

The prompt-response row T supplies the source-channel and prompt-output order, while the observer-interface row I supplies the public reading route. Thus a consumer cannot read the response as a free-standing host statement or as an unledgered observer event.

The large-model inscription row M and inscription-event row E bind the response to the finite substrate and accepted event rows. The prompt row P is read before the response row O, and the audit-refusal row R is read before the local name.

The remaining rows provide only transport, continuation replay, provenance, and local naming. Since the carrier has no further coordinate, the listed obligations exhaust the NameCert surface. □

Theorem 966.3 (Observer-response audit refusal boundary)**.** *Every accepted* ObserverResponseAudit$^{\uparrow}$ *packet exposes the audit-refusal row R after the prompt and response rows and before the local naming row. A packet with prompt and response rows but no audit-refusal row is not an accepted* ObserverResponseAudit$^{\uparrow}$ *carrier.*

Proof. Use Definition 966.1. The tuple contains R as a named coordinate between the prompt-response content and the transport, continuation, provenance, and local naming rows.

Apply Definition 966.2. Its refusal-exactness clause is part of the local NameCert surface, so carrier admission fails when the prompt and response rows are present but the refusal row is absent.

Therefore the packet cannot be read as internally certified universal closure, host semantic truth, model consciousness, or a global halting decision. The accepted route remains the finite prompt, response, observer-interface, inscription, refusal, transport, continuation, provenance, and local naming route. □

Falsifiable prediction: A packet with prompt and response rows but no audit-refusal row is not an accepted ObserverResponseAudit$^{\uparrow}$ carrier.

Independence witness: Nearest siblings: promptresponsetrace, observerinterface, largemodelinscriptionpoint, inscriptionevent, and haltingobstruction; none is bijective because this carrier

simultaneously requires the prompt-response trace, observer-interface handoff, large-model inscription row, inscription event row, audit-refusal row, componentwise transport, displayed continuation, provenance, and local NameCert row, while each sibling omits at least one of those displayed coordinates.

The Lean file `Derived/ObserverResponseAuditUp/TasteGate.lean` must later include `Nontrivial ObserverResponseAuditUp` and `FieldFaithful ObserverResponseAuditUp` instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverResponseAudit}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverResponseAudit}^{\uparrow}$ packet is generated from a prompt-response trace, observer-interface row, large-model inscription row, inscription-event row, prompt row, response row, audit-refusal row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverResponseAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 966.1, Definition 966.2, and Definition 966.3 specify the finite observer-response audit carrier and its refusal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverResponseAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, host semantic truth, model consciousness, global halting classification, or internally certified universal closure for $\mathsf{ObserverResponseAudit}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, prompt-response exposure, observer-interface handoff, inscription-event exposure, audit-refusal exactness, nontriviality, and field faithfulness for the same carrier.

Chapter 967

A Concrete Naming Certificate for $\mathsf{LargeModelActivationOrbit}^{\uparrow}$

The $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ packet realizes the activation-orbit row of chapter 9398 as finite BEDC data. It is a sibling of chapter 939 and chapter 965: the former names the broader inscription surface, while the latter records prompt and output traces. This chapter isolates the model-weight, transition, activation-window, prompt, output, audit-refusal, and local NameCert rows needed to classify one displayed orbit without turning that orbit into consciousness, host semantic understanding, global halting power, or internally ratified universal closure.

Definition 967.1 (LargeModelActivationOrbit carrier)**.** *A* $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, T, A, P, Y, R, H, C, G, N)$$

with the following displayed rows:

1. *W is the finite model-weight row naming the inference substrate used by the orbit;*
2. *T is the displayed transition row, a finite forward-pass schedule read over W;*
3. *A is the activation-window row recording the finite sequence of activation states exposed by the orbit;*
4. *P is the prompt row that supplies the stated initial condition for the activation window;*
5. *Y is the output row read after the displayed transition and activation-window rows;*
6. *R is the audit-refusal row stating that the orbit does not internally certify consciousness, host semantic truth, global halting classification, or universal closure;*
7. *H records componentwise* hsame *transport for the model-weight, transition, activation-window, prompt, output, refusal, route, provenance, and naming rows;*
8. *C records the displayed* Cont *routes by which a consumer reads W, then T, then A, then P, Y, R, before reaching the local name;*
9. *G is the* Pkg *provenance over* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{PromptResponseTrace}^{\uparrow}$, $\mathsf{ObserverInterface}^{\uparrow}$, $\mathsf{HaltingObstruction}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LargeModelActivationOrbit}^{\uparrow}}$ *row.*

The carrier contains no hidden activation stream, no quotient identity of orbits, no host meaning oracle, no subject primitive, and no model-side proof that the output row is true.

Definition 967.2 (LargeModelActivationOrbit classifier)**.** *Two accepted* $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ *carriers classify together exactly when their model-weight, transition, activation-window, prompt, output, audit-refusal, continuation, provenance, and local naming rows compare componentwise by the displayed* hsame *entries:*

$$W \sim_{\mathsf{hsame}} W', \quad T \sim_{\mathsf{hsame}} T', \quad A \sim_{\mathsf{hsame}} A', \quad P \sim_{\mathsf{hsame}} P', \quad Y \sim_{\mathsf{hsame}} Y',$$
$$R \sim_{\mathsf{hsame}} R', \quad C \sim_{\mathsf{hsame}} C', \quad G \sim_{\mathsf{hsame}} G', \quad N \sim_{\mathsf{hsame}} N'.$$

Changing the transition row or activation-window row while leaving the orbit classifier unchanged is not an accepted comparison. No clause compares hidden model intention, corpus-wide truth, unbounded activation streams, global halting behavior, or an externally interpreted universal claim.

Theorem 967.3 (LargeModelActivationOrbit NameCert obligations)**.** *For the carrier of Definition 967.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelActivationOrbit}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for* $O = (W, T, A, P, Y, R, H, C, G, N)$*;*
2. *classifier transport for the model-weight, transition, activation-window, prompt, output, audit-refusal, continuation, provenance, and naming rows under componentwise* hsame*;*
3. *transition-window exactness requiring every accepted orbit read to pass through the displayed transition row* T *and activation-window row* A *before the output row* Y *is consumed;*
4. *prompt-output ledger exactness requiring the prompt row* P*, output row* Y*, and audit-refusal row* R *to remain in the same local certificate surface;*
5. *non-escape for hidden activation streams, consciousness, host semantic understanding, global halting classification, selected corpus truth, and internally ratified universal closure.*

Proof. Project the carrier of Definition 967.1. The rows $W, T, A, P, Y, R, H, C, G, N$ are the whole packet, so any accepted consumer starts from these displayed finite rows.

Classifier transport is componentwise. A comparison of accepted orbits must transport the model weights, transition schedule, activation window, prompt, output, refusal, route, provenance, and local name together, so a changed transition row or activation-window row exits the certificate unless the displayed classifier changes with it.

Transition-window exactness follows from the route order in C. The output row is read only after W, T, A, and the prompt row is retained as the stated initial condition for that same window.

The audit-refusal and provenance rows connect the orbit to its neighbors. chapter 939 supplies the inscription-point substrate reading, chapter 965 supplies the prompt-output trace surface, chapter 928 supplies public observation discipline, and chapter 490 supplies the non-oracle boundary.

Since the carrier has no hidden activation stream, subject primitive, host meaning oracle, global halting classifier, or universal-truth proof coordinate, none can be projected from the local NameCert surface. □

```
Lean4 ✓  BEDC.Derived.LargeModelActivationOrbitUp.TasteGate.LargeModelActivationOrbitTasteGate
single carrier alignment
```

Theorem 967.4 (LargeModelActivationOrbit vision realization)**.** *The activation-orbit reading in chapter 9398 is realized by the finite carrier of Definition 967.1. The realization reads model weights through* W*, transition scheduling through* T*, finite activation exposure through* A*, prompt*

initialization through P, output inscription through Y, audit refusal through R, transport and continuation through H, C, provenance through G, and local naming through N.

The public export is exactly this finite route. It does not export a hidden activation stream, a selected completed semantic object, consciousness, host semantic equality, global halting classification, ambient corpus truth, or internally certified universal closure.

Proof. Start with the large-model substrate definition in chapter 9398. Its observational records are finite inference orbits over weights, a transition, and a stated prompt. The rows W, T, A, P encode exactly that finite observation surface.

The sibling $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ packet supplies the broader inscription reading, but it does not isolate the transition and activation-window rows as the classifier-bearing carrier. The sibling $\mathsf{PromptResponseTrace}^{\uparrow}$ packet supplies prompt and output rows, but it treats the activation orbit as upstream substrate rather than as its own certified object.

The rows Y, R, H, C, G, N complete the local certificate. They expose output, refusal, transport, continuation, provenance, and local naming while preserving the same non-escape boundary as the inscription-point and prompt-response chapters.

Therefore the activation-orbit row of the vision is realized as a separate finite NameCert carrier, not as an overloaded reading of the inscription-point or prompt-response carrier. □

Theorem 967.5 (LargeModelActivationOrbit scoped consumer route)**.** *Every downstream consumer route for an accepted* $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ *packet factors through* $W, T, A, P, Y, R, H, C, G, N$ *in that order. The sibling surfaces* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{PromptResponseTrace}^{\uparrow}$, $\mathsf{ObserverInterface}^{\uparrow}$, *and* $\mathsf{HaltingObstruction}^{\uparrow}$ *may consume the packet only through those displayed rows, and no route can bypass the audit-refusal row R to obtain consciousness, host semantic truth, global halting classification, or internally ratified universal closure.*

Proof. Use the carrier inventory of Definition 967.1. It lists $W, T, A, P, Y, R, H, C, G, N$ as the whole finite packet, with C recording displayed consumer routes and G recording provenance through the sibling surfaces.

The classifier of Definition 967.2 transports the same coordinates componentwise. Thus a downstream consumer cannot preserve the orbit classifier while changing the transition row, activation-window row, prompt row, output row, refusal row, route row, provenance row, or local name.

Apply the NameCert obligations of Definition 967.3. They force transition-window exactness through T, A, prompt-output ledger exactness through P, Y, R, and non-escape for hidden activation streams, consciousness, host semantic understanding, global halting classification, selected corpus truth, and internally ratified universal closure.

Finally compose with the vision realization of Definition 967.4. The sibling packets may read the orbit as inscription point, prompt-response trace, public observation surface, or halting boundary, but only after the displayed route reaches R and keeps the same finite NameCert packet. Hence the scoped consumer route cannot bypass the audit-refusal row. □

Falsifiable prediction: Changing the displayed transition row or activation-window row without changing the orbit classifier exits the certificate.

Independence witness: Nearest siblings: largemodelinscriptionpoint, promptresponsetrace, observerinterface, haltingobstruction; none is bijective because this carrier makes the transition row and activation-window row classifier-bearing data while retaining prompt, output, audit-refusal, transport, provenance, and local naming rows in one finite orbit certificate.

The Lean follow-up for this chapter must provide `Nontrivial LargeModelActivationOrbitUp` and `FieldFaithful LargeModelActivationOrbitUp` instances in `BEDC.Derived.LargeModelActivationOrbitU`

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelActivationOrbit}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ packet is generated from finite model-weight, transition, activation-window, prompt, output, audit-refusal, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeModelActivationOrbit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 967.4 and Definition 967.5 bind the displayed activation-orbit carrier, classifier, NameCert obligations, falsifiable prediction, independence witness, and TasteGate-visible packet route as the bridged public surface, with

Lean4 ✓ `BEDC.Derived.LargeModelActivationOrbitUp.taste gate`

recording the TasteGate instance.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeModelActivationOrbit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LargeModelActivationOrbitUp.TasteGate.LargeModelActivationOrbitTasteGate\_single\_carri`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close hidden activation streams, model consciousness, host semantic understanding, global halting classification, corpus truth, or bridge checking for $\mathsf{LargeModelActivationOrbit}^{\uparrow}$.

Upgrade path: Bridge checking requires a standard bridge theorem for the same activation-orbit carrier and TasteGate-visible packet.

Chapter 968

A Concrete Naming Certificate for $\mathsf{CorpusSupplyDistillation}^{\uparrow}$

The $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ packet realizes the source-channel row of chapter 9398 as its own finite NameCert carrier. It refines the corpus-supply coordinate of chapter 939 before that coordinate is consumed by the neighboring inscription surfaces of chapter 610 and chapter 519. The packet records corpus supply, filtering, distillation, output-prior, refusal, transport, route, provenance, and local naming rows without treating a model output as internally certified truth.

Definition 968.1 (CorpusSupplyDistillation carrier)**.** *A* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, F, D, O, R, H, T, P, N)$$

with the following displayed rows:

1. *C is the corpus-supply row naming the finite source channel from which training or retrieval inscriptions are carried;*
2. *F is the filtering row naming the finite selection, exclusion, and normalization surface applied to the source channel;*
3. *D is the distillation row naming the finite compression, training, or transfer route by which source-channel patterns are made available to a model-side packet;*
4. *O is the output-prior row recording the carried prior over emitted records before any public output is read as an inscription;*
5. *R is the refusal row excluding corpus-wide truth, hidden source selection, host semantic understanding, consciousness, and model-side proof of output truth;*
6. *H records componentwise* hsame *transport for the corpus-supply, filtering, distillation, output-prior, refusal, route, provenance, and naming rows;*
7. *T records displayed* Cont *routes through C, F, D, O, R before any downstream output record or inscription point is consumed;*
8. *P is the* Pkg *provenance over* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ObserverInterface}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$ *row.*

The carrier is exactly this finite source-channel packet. It has no coordinate for corpus-total truth, hidden dataset selection, model intention, host semantic ratification, or internally certified output content.

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationTasteGate single carrier alignment`

Theorem 968.2 (CorpusSupplyDistillation NameCert obligations)**.** *For the carrier of Definition 968.1, the local* $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (C, F, D, O, R, H, T, P, N)$*, corpus-supply exposure through* C*, filtering exposure through* F*, distillation exposure through* D*, output-prior exposure through* O*, refusal accounting through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* T*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationNameCert obligations`

Proof. Project the displayed carrier. The accepted packet lists $C, F, D, O, R, H, T, P, N$, so the local certificate cannot introduce another source channel, filter, distillation route, output prior, refusal row, transport, route, provenance, or local name.

Read the source-channel rows in dependency order. A consumer first reaches the finite corpus-supply row C, then the filtering row F, then the distillation row D, and only then the output-prior row O.

The refusal and structural rows close the certificate surface. The row R records the missing truth and semantic coordinates, while H, T, P, N transport, replay, package, and name only the same finite packet. Thus the NameCert obligations are exhausted by the displayed carrier. □

Theorem 968.3 (CorpusSupplyDistillation carried-output boundary)**.** *Every accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet routes a corpus-carried output prior through the finite path*

$$\textit{corpus-supply row} \longrightarrow \textit{filtering row} \longrightarrow \textit{distillation row} \longrightarrow \textit{output-prior row} \longrightarrow \mathsf{NameCert}_{\mathsf{CorpusSupplyDistillat}}$$

When this row is later consumed by $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$*, the output record remains a carried inscription surface rather than internally certified output truth.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation carried output boundary`

Proof. Start with the carrier $K = (C, F, D, O, R, H, T, P, N)$. The source-channel route in T reads C, F, D before it reaches O, so the output prior is carried from displayed corpus supply rather than projected from a hidden model certificate.

The LargeModelInscriptionPoint chapter consumes a corpus-supply row before reading the output record. This chapter supplies that row as a standalone packet whose route has already exposed filtering and distillation.

Finally, the refusal row R is read before the local naming row N. Therefore the downstream output boundary preserves carried inscription and does not become a proof of output truth, host semantic understanding, or corpus-total correctness. □

Theorem 968.4 (CorpusSupplyDistillation public non-escape)**.** *No public consumer of an accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet can extract hidden source selection, corpus-wide truth, internally certified output truth, consciousness, host semantic understanding, or a model-side proof object. Public reads are limited to the finite corpus-supply, filtering, distillation, output-prior, refusal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in the carrier.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation public nonescape`

Proof. Inspect the carrier inventory of Definition 968.1. The only public coordinates are $C, F, D, O, R, H, T, P, N$.

Apply the carried-output boundary. Any consumer that reaches the output-prior row has already passed through the source-channel, filtering, and distillation rows, and then through the refusal row before local naming.

Since hidden source selection, corpus-wide truth, semantic understanding, consciousness, and proof of output truth are not coordinates of K, no public projection can recover them. The packet exports only the finite source-channel surface and its displayed certificate route. □

Definition 968.5 (CorpusSupplyDistillation obligation ledger)**.** *For the carrier $K = (C, F, D, O, R, H, T, P, N)$ of Definition 968.1, the obligation ledger is the ordered row family*

$$C, F, D, O, R, H, T, P, N.$$

The rows record corpus supply, filtering, distillation, output prior, refusal, componentwise hsame *transport, displayed* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$ *naming. A consumer of the ledger must read the source-channel route $C \longrightarrow F \longrightarrow D \longrightarrow O$ through T before treating the output-prior row as public, and must read R before N whenever the requested output-prior interpretation would claim hidden source selection, corpus-wide truth, host semantic understanding, consciousness, or internally certified output truth.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation obligation ledger namecert boundary`

Theorem 968.6 (CorpusSupplyDistillation filter-route order)**.** *Every accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet reads the displayed* Cont *route T in the order*

$$C \longrightarrow F \longrightarrow D \longrightarrow O$$

before any carried output-prior row is consumed. The packet has no accepted route that reaches O before the corpus-supply row C, filtering row F, or distillation row D has been displayed.

Proof. Start from the carrier $K = (C, F, D, O, R, H, T, P, N)$ of Definition 968.1. Its source-channel rows are listed as corpus supply, filtering, distillation, and output prior, and the route row T is the displayed Cont route through those rows.

Apply the NameCert obligations of Definition 968.2. They require corpus-supply exposure, filtering exposure, distillation exposure, and output-prior exposure as separate duties of the same local certificate.

Finally use public non-escape from Definition 968.4. A route that reaches O without C, F, D would need hidden source selection or internally certified output truth, and those coordinates are absent from the accepted packet. □

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation filter route order`

Theorem 968.7 (CorpusSupplyDistillation output-prior refusal exactness)**.** *For every accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet, an output-prior read that would be interpreted as hidden source selection, corpus-wide truth, host semantic understanding, consciousness, or internally certified output truth must expose the refusal row R before the local* $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$ *row N. The local name records a carried source-channel packet, not a certificate of output truth.*

Proof. Use the obligation ledger of Definition 968.5. It places R between the output-prior row and the structural certificate rows so that disallowed interpretations are refused before local naming is consumed.

The NameCert obligations of Definition 968.2 make refusal accounting part of the same finite carrier. The local name N therefore names a packet whose refusal row has already excluded the stronger readings.

Apply public non-escape from Definition 968.4. Since the excluded readings are not coordinates of K, any output-prior read that asks for them must stop at R before it can cite the local NameCert row. □

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation output prior refusal exactness`

Theorem 968.8 (CorpusSupplyDistillation source-channel exposure)**.** *Every accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet exposes its source channel through the displayed row C. No public consumer can replace C by hidden dataset selection, corpus-total truth, or an unrecorded supply channel.*

Proof. Project the carrier of Definition 968.1. The source-channel coordinate is the row C in the finite packet $K = (C, F, D, O, R, H, T, P, N)$.

Apply Definition 968.2. The local NameCert surface names corpus-supply exposure through C before filtering, distillation, output-prior, refusal, transport, replay, provenance, and local naming are read.

Use Definition 968.4. Hidden source selection and corpus-wide truth are not coordinates of the carrier, so all source exposure available to a public consumer is exactly the displayed C row. □

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationSourceChannel exposure`

Theorem 968.9 (CorpusSupplyDistillation filter distillation output-prior order)**.** *Every output-prior read from an accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet follows the finite route order*

$$C, F, D, O.$$

The output prior is therefore read after source-channel exposure, filtering, and distillation, not as an internally certified model-output truth.

Proof. Start from Definition 968.1. The rows C, F, D, O are displayed in the carrier before the refusal and structural rows.

Apply Definition 968.2. It records corpus-supply exposure through C, filtering through F, distillation through D, and output-prior exposure through O as the ordered public route.

Finally use Definition 968.4. Since internally certified output truth is absent from K, the only available output-prior read is the carried route through C, F, D, O. □

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation filter distillation output prior nonescape`

Theorem 968.10 (CorpusSupplyDistillation refusal exactness)**.** *The refusal surface of an accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packet is exactly the finite route*

$$R, H, T, P, N.$$

It refuses corpus-wide truth, hidden source selection, model consciousness, host semantic understanding, internally certified output truth, and model-side proof objects only through the displayed refusal, transport, continuation, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation refusal exactness`

Proof. Read the carrier in Definition 968.1. The refusal row R is followed by componentwise transport H, displayed continuation replay T, package provenance P, and local naming N.

Apply Definition 968.2. It makes refusal accounting a named obligation of the same packet and gives H, T, P, N as the structural rows that preserve and expose that refusal.

Conclude with Definition 968.4. Each refused object is outside the carrier, so the exact public refusal route is R, H, T, P, N and nothing stronger is exported. □

Theorem 968.11 (CorpusSupplyDistillation carrier admission)**.** *Every accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *read exposes exactly the finite carrier rows*

$$C, F, D, O, R, H, T, P, N.$$

The accepted read therefore displays source-channel, filtering, distillation, output-prior, refusal, componentwise transport, continuation route, provenance, and local naming rows, and it exposes no corpus-wide truth, hidden source selection, model consciousness, host semantic understanding, internally certified output truth, or model-side proof object.

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationCarrier admission`

Proof. Start from the carrier of Definition 968.1. Its displayed coordinates are exactly $C, F, D, O, R, H, T, P, N$, so an accepted read has no additional coordinate to expose.

Apply the obligation surface of Definition 968.2. It assigns the local NameCert duties to the same source-channel, filtering, distillation, output-prior, refusal, transport, replay, provenance, and naming rows.

Finally use the obligation ledger of Definition 968.5. The ledger names the refused stronger readings outside the carrier, so admission exposes only the displayed rows. □

Theorem 968.12 (CorpusSupplyDistillation classifier stability)**.** *Componentwise* hsame *transport of accepted* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *packets preserves the source-channel, filtering, distillation, output-prior, refusal, route, provenance, and naming rows. In particular, a transported packet remains a read of* $C, F, D, O, R, H, T, P, N$ *rather than a read of a hidden corpus selector or internally certified output truth.*

Proof. Read the carrier rows from Definition 968.1. The classifier has one displayed component for each of $C, F, D, O, R, H, T, P, N$, so componentwise hsame transport can only preserve those rows.

Use the ordered route of Definition 968.6. It keeps filtering before distillation and output-prior exposure, preventing transport from reordering the source-channel route.

Then apply Definition 968.10. Refusal remains attached through R, H, T, P, N, so stronger readings are still refused after transport.

The NameCert obligations of Definition 968.2 repack the transported rows into the same local naming surface. □

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation classifier stability`

Theorem 968.13 (CorpusSupplyDistillation ledger exactness)**.** *The* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ *obligation ledger exports no corpus-wide truth, hidden source selection, model consciousness, host semantic understanding, internally certified output truth, or model-side proof object. Its public ledger is exactly the source-channel, filtering, distillation, output-prior, refusal, transport, route, provenance, and local naming surface of the accepted packet.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationLedger exactness`

Proof. Begin with Definition 968.5. The ledger names the admissible public rows and lists the stronger readings as refused objects rather than exported fields.

Use Definition 968.11 to identify the accepted carrier surface with $C, F, D, O, R, H, T, P, N$. There is no coordinate for corpus-wide truth, hidden source selection, model consciousness, host semantic understanding, internally certified output truth, or a model-side proof object.

Finally combine Definition 968.6 with Definition 968.10. The first theorem fixes the source-to-output-prior order, and the second fixes the refusal route, so the obligation ledger exports exactly the finite packet and no stronger semantic claim. □

Falsifiable prediction: If the source-channel row is removed, the packet cannot distinguish corpus-carried output priors from internally certified output truth.

Independence witness: Nearest siblings: largemodelinscriptionpoint, inscriptionpoint, inscriptionevent, apophaticname, and observerinterface. None is bijective to $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ because LargeModelInscriptionPoint carries inference and output rows in addition to corpus supply, InscriptionPoint and InscriptionEvent carry inscription handoff rows without the filtering and distillation source channel, ApophaticName carries a boundary request rather than corpus provenance, and ObserverInterface carries public observer rows without the corpus-supply, filtering, distillation, and output-prior route.

Theorem 968.14 (CorpusSupplyDistillation Lean TasteGate handoff)**.** *The displayed carrier rows*

$$C, F, D, O, R, H, T, P, N$$

are the same rows consumed by the Lean ChapterTasteGate*,* Nontrivial*, and* FieldFaithful *instances for* $\mathsf{CorpusSupplyDistillation}^{\uparrow}$*. The Lean handoff uses the same single finite carrier rather than a second hidden packet.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.taste gate`

Proof. Begin with the carrier of Definition 968.1. It displays exactly $C, F, D, O, R, H, T, P, N$, and Definition 968.2 assigns the local certificate duties to those rows.

Use Definition 968.7 to keep the output-prior and refusal rows inside the same accepted packet. The refusal route is not an extra carrier; it is the R, H, T, P, N part of the displayed carrier.

Finally apply the existing Lean single-carrier marker

BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillationTasteGate_single_carrier_alignment.

It identifies the Lean encoding and decoding surface with the displayed carrier, so the Lean ChapterTasteGate, Nontrivial, and FieldFaithful instances consume the same rows described by the paper certificate. □

Theorem 968.15 (CorpusSupplyDistillation mature package boundary)**.** *The corpus-supply, filtering, distillation, output-prior, refusal, transport, route, provenance, and* $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$ *rows form the mature package boundary consumed by* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *and* $\mathsf{InscriptionEvent}^{\uparrow}$*. This boundary exports no corpus-wide truth and no internally certified output truth.*

Lean4 ✓ `BEDC.Derived.CorpusSupplyDistillationUp.CorpusSupplyDistillation mature package boundary`

Proof. Read the carrier of Definition 968.1. Its finite packet is exactly $C, F, D, O, R, H, T, P, N$, so the package boundary has one displayed row for corpus supply, filtering, distillation, output prior, refusal, transport, route, provenance, and local naming.

Apply Definition 968.3 and Definition 968.14. The first theorem routes the output prior through the source-channel surface consumed by $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, and the second keeps the same carrier available to the inscription-facing handoff.

Finally use Definition 968.4. Corpus-wide truth and internally certified output truth are not exported coordinates of the packet, so the mature package boundary remains a finite source-channel and refusal surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}$.

Constructive story: A $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ packet is generated from corpus-supply, filtering, distillation, output-prior, refusal, componentwise transport, displayed continuation, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 968.1, Definition 968.4, Definition 968.14, and Definition 968.15 expose the public non-escape theorem, bridge-facing finite-carrier route, existing TasteGate handoff, and mature package boundary through `BEDC.Derived.CorpusSupplyDistillationUp.taste_gate` as the exported finite source-channel distillation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CorpusSupplyDistillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close corpus-wide truth, hidden source selection, model consciousness, host semantic understanding, internally certified output truth, full Lean verification status, or bridge checking for $\mathsf{CorpusSupplyDistillation}^{\uparrow}$.

Upgrade path: Bridge checking requires a verified standard bridge for the same finite carrier and public non-escape surface.

Chapter 969

A Concrete Naming Certificate for $\mathsf{LargeModelAuditChannel}^{\uparrow}$

The $\mathsf{LargeModelAuditChannel}^{\uparrow}$ packet realizes the verified-output-channel side of chapter 9398 as finite BEDC data. It consumes the inscription carrier of chapter 939, the public audit membrane of chapter 524, and the prompt-output trace of chapter 965. Its distinguishing role is the consumer-facing audit channel: prompt, output, checker, refusal, route, provenance, and local NameCert rows are carried together before any model emission is accepted as audited output.

Definition 969.1 (LargeModelAuditChannel carrier)**.** *A* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, Q, O, C, R, U, V, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *row naming the finite inference substrate, corpus-supply channel, output-record surface, and no-internal-ratification boundary;*
2. *Q is the prompt row, containing the finite request text, input provenance, and admitted context window;*
3. *O is the output row, containing the emitted token sequence as public inscription data;*
4. *C is the checker row, naming the finite evaluation harness, proof checker, policy checker, or replay procedure used to inspect O;*
5. *R is the refusal row, recording rejected claims, unverifiable universal closures, policy refusals, or unsupported checker outcomes;*
6. *U is the audit-membrane row through which a public consumer reads prompt, output, checker, and refusal data without direct model-state access;*
7. *V is the route row recording the displayed continuation from prompt through output, checker, refusal, membrane, provenance, and local name;*
8. *H records componentwise* hsame *transport for the model, prompt, output, checker, refusal, membrane, route, provenance, and naming rows;*
9. *K records the displayed* Cont *replay accepted by the channel;*
10. *P is the* Pkg *provenance over* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{PromptResponseTrace}^{\uparrow}$, *chapter 524, chapter 928,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

11. N is the local $\mathsf{NameCert}_{\mathsf{LargeModelAuditChannel}^{\uparrow}}$ *row.*
The carrier has no coordinate for hidden model intention, corpus-wide truth, host semantic understanding, unchecked output acceptance, or a private evaluator outside the displayed checker row.

Lean4 ✓ `BEDC.Derived.LargeModelAuditChannelUp.LargeModelAuditChannelTasteGate single carrier alignment`

Definition 969.2 (LargeModelAuditChannel classifier)**.** *Two accepted* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *carriers classify together exactly when their model rows, prompt rows, output rows, checker rows, refusal rows, membrane rows, route rows, transport rows, replay rows, provenance rows, and local naming rows compare by the displayed* hsame *entries. Thus the classifier is finite and componentwise:*

$$M \sim_{\mathsf{hsame}} M', \quad Q \sim_{\mathsf{hsame}} Q', \quad O \sim_{\mathsf{hsame}} O', \quad C \sim_{\mathsf{hsame}} C',$$
$$R \sim_{\mathsf{hsame}} R', \quad U \sim_{\mathsf{hsame}} U', \quad V \sim_{\mathsf{hsame}} V', \quad H \sim_{\mathsf{hsame}} H',$$
$$K \sim_{\mathsf{hsame}} K', \quad P \sim_{\mathsf{hsame}} P', \quad N \sim_{\mathsf{hsame}} N'.$$

No clause compares unrecorded checker state, hidden evaluation policy, model-side semantic truth, or an unchecked continuation outside V and K.

Theorem 969.3 (LargeModelAuditChannel NameCert obligations)**.** *For the carrier of Definition 969.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelAuditChannel}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (M, Q, O, C, R, U, V, H, K, P, N)$, prompt exposure through Q, output exposure through O, checker exposure through C, refusal exposure through R, audit-membrane handoff through U, route replay through V and K, componentwise stability through H, provenance through P, and local naming through N. A consumer obtains audited output only after these rows are jointly displayed.*

Proof. Project the carrier of Definition 969.1. The rows $M, Q, O, C, R, U, V, H, K, P, N$ are the whole public packet, so prompt, output, checker, refusal, route, provenance, and local naming are all visible before the channel is consumed.

The checker row C and refusal row R provide the audit boundary. They separate an inspected output from a bare model emission by recording which finite harness was applied and which claims were refused or left unsupported.

The membrane and route rows make the read public. The U, V, K rows force the consumer-facing path to pass through the audit membrane and displayed replay rather than through hidden model state or a private evaluator.

The remaining H, P, N rows supply transport, provenance, and local naming. Since no carrier coordinate stores hidden intention, corpus-wide truth, host semantic understanding, or unchecked output acceptance, the listed rows exhaust the NameCert obligations. □

Theorem 969.4 (LargeModelAuditChannel audited-output boundary)**.** *If the checker row C is absent from an accepted prompt-output packet, the packet cannot distinguish audited output from an unaudited model emission. With C present, every audited-output read factors through the displayed prompt row Q, output row O, checker row C, refusal row R, audit membrane U, route rows V, K, provenance row P, and local name N.*

Proof. The carrier records output in O, but O alone is only an emitted inscription row. It becomes an audited output only when the route also displays the checker row C and the refusal row R.

Apply Definition 969.3. Any accepted public read reaches N only after the channel exposes prompt, output, checker, refusal, membrane, replay, transport, provenance, and local naming rows.

If C is removed, the remaining packet still contains a prompt and an output but no displayed evaluation harness or proof-checking surface. The consumer then has no finite coordinate by which to separate inspected output from unaudited emission. □

Theorem 969.5 (LargeModelAuditChannel checker exposure obligation)**.** *For every accepted* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *packet*

$$A = (M, Q, O, C, R, U, V, H, K, P, N),$$

the checker row C is exposed as a carrier obligation before any audited-output consumer may read the output row O through the audit membrane U. The exposure is componentwise under the classifier and exports no hidden evaluator, model-side intention, corpus-wide truth row, or unchecked output acceptance coordinate.

Proof. Project A through Definition 969.1. The checker row C is one of the displayed rows between the output and refusal boundary, so it is part of the finite public packet.

Apply Definition 969.2. The classifier compares the checker coordinate by the displayed hsame entry and has no clause for hidden checker state or private evaluator data.

Use Definition 969.3. The local NameCert surface exposes the checker row before the audited-output route can pass through U, V, K, P, N. □

Theorem 969.6 (LargeModelAuditChannel refusal exactness obligation)**.** *For every accepted* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *packet*

$$A = (M, Q, O, C, R, U, V, H, K, P, N),$$

the refusal row R records exactly the rejected claims, unsupported checker outcomes, and unverifiable universal closures attached to the displayed prompt, output, and checker rows Q, O, C. It does not export a private policy oracle, host semantic truth predicate, hidden model intention, or global proof-checking result.

Proof. Read R from Definition 969.1. The carrier places the refusal row after prompt, output, and checker data and before the public audit membrane.

Use Definition 969.2 to keep refusal comparison componentwise. The classifier compares only the displayed refusal coordinate and never compares an unrecorded policy oracle or host semantic predicate.

Apply Definition 969.3. The NameCert surface exposes refusal data as part of the accepted audit channel, so rejected claims and unsupported checker outcomes remain exact finite rows rather than external judgments. □

Theorem 969.7 (LargeModelAuditChannel route replay obligation)**.** *For every accepted* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *packet*

$$A = (M, Q, O, C, R, U, V, H, K, P, N),$$

the route replay obligation is the displayed continuation from prompt Q through output O, checker C, refusal row R, audit membrane U, route row V, replay row K, provenance P, and local name N. The TasteGate carrier alignment for BEDC.Derived.LargeModelAuditChannelUp.LargeModelAuditChannelTasteGate_ *confirms that this obligation is attached to the same finite carrier and not to hidden model state, private evaluator correctness, or global semantic truth.*

Proof. Start with Definition 969.1. It lists V and K as the route and replay rows for the same packet $A = (M, Q, O, C, R, U, V, H, K, P, N)$.

Apply Definition 969.2. Route and replay comparison remains componentwise, so the consumer cannot replace V or K with an unrecorded continuation or private evaluator route.

Use Definition 969.3 together with the existing TasteGate carrier alignment marker BEDC.Derived.LargeModelAu The public replay route is therefore the finite audit-channel packet itself, with no hidden model state or global semantic truth exported. □

Theorem 969.8 (LargeModelAuditChannel public route scope)**.** *Every public consumer route for an accepted* LargeModelAuditChannel$^{\uparrow}$ *packet*

$$A = (M, Q, O, C, R, U, V, H, K, P, N)$$

reads $M, Q, O, C, R, U, V, H, K, P, N$ through the checker row, refusal row, audit membrane, displayed route and replay rows, provenance row, and local NameCert *row. It cannot bypass the checker row, collapse refusal into output acceptance, or export hidden model intention, corpus-wide truth, host semantic understanding, private evaluator correctness, or unchecked output.*

Proof. First project the carrier and its local obligations by Definition 969.3. The public packet is the displayed tuple $M, Q, O, C, R, U, V, H, K, P, N$, so no consumer route has a coordinate beyond those rows.

Apply Definition 969.4 and Definition 969.5. These results force every audited-output read to expose C before O is consumed through the audit membrane, excluding unchecked output and private evaluator replacement.

Next use Definition 969.6. The refusal row R remains its own finite row attached to Q, O, C, so unsupported checker outcomes and rejected claims cannot be absorbed into output acceptance.

Finally apply Definition 969.7. The consumer route factors through U, V, K, P, N, and the same replay surface excludes hidden model intention, corpus-wide truth, host semantic understanding, private evaluator correctness, and any continuation not displayed by the carrier. □

Falsifiable prediction: If the checker row is absent, the packet cannot distinguish audited output from an unaudited model emission.

Independence witness: largemodelinscriptionpoint, inscriptionevent, inscription-acceptance-budget, auditmembrane, observerinterface: none is bijective because the audit-channel packet carries prompt, output, checker, refusal, route, provenance, and local NameCert rows together.

The future Lean TasteGate namespace BEDC.Derived.LargeModelAuditChannelUp.TasteGate must provide Nontrivial LargeModelAuditChannel$^{\uparrow}$ and FieldFaithful LargeModelAuditChannel$^{\uparrow}$ instances for this finite carrier.

Theorem 969.9 (LargeModelAuditChannel abstract-interface handoff)**.** *The public* LargeModelAuditChannel$^{\uparrow}$ *carrier, componentwise classifier, TasteGate single-carrier alignment,* hsame *transport, displayed replay, provenance, and local* NameCert *rows jointly supply the abstract large-model audit-channel interface. Every interface read is therefore routed through the public prompt, output, checker, refusal, audit-membrane, route, replay, provenance, and naming rows, and no bridge handoff may introduce hidden model state, private evaluator correctness, corpus-wide truth, or unchecked semantic understanding.*

Proof. Start with the public carrier and classifier of Definition 969.1 and Definition 969.2. They expose exactly the rows $M, Q, O, C, R, U, V, H, K, P, N$ and compare them componentwise.

Apply the NameCert and public-route results, Definition 969.3 and Definition 969.8. These force every accepted audit-channel read through the checker, refusal, membrane, replay, provenance, and local naming rows before the interface is consumed.

The TasteGate single-carrier alignment recorded by BEDC.Derived.LargeModelAuditChannelUp.LargeModelAudit attaches that public route to the same finite carrier. Thus the abstract-interface handoff is the displayed packet itself, not a private evaluator, hidden model state, corpus-wide truth row, or unchecked semantic object. □

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelAuditChannel}^{\uparrow}}$.

Constructive story: A $\mathsf{LargeModelAuditChannel}^{\uparrow}$ packet is generated from a large-model inscription row, prompt row, output row, checker row, refusal row, audit membrane, displayed route, componentwise transport, replay, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.LargeModelAuditChannelUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeModelAuditChannel}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 969.1, Definition 969.2, Definition 969.3, Definition 969.4, Definition 969.5, Definition 969.6, Definition 969.7, Definition 969.8, and Definition 969.9 specify the complete paper-side audit-channel package: carrier, classifier, NameCert obligations, public route scope, abstract-interface handoff, and audit-channel refusal rows for large-model output records.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeModelAuditChannel}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LargeModelAuditChannelUp.LargeModelAuditChannelTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close model consciousness, corpus truth, host semantic understanding, private evaluator correctness, global proof checking, Lean verification, or bridge checking for $\mathsf{LargeModelAuditChannel}^{\uparrow}$.

Upgrade path: Bridge-checked closure requires a real StdBridge theorem connecting the public audit-channel surface to the abstract $\mathsf{LargeModelAuditChannel}^{\uparrow}$ interface.

Chapter 970

A Concrete Naming Certificate for $\mathsf{ReductionFuelBoundary}^{\uparrow}$

The $\mathsf{ReductionFuelBoundary}^{\uparrow}$ packet is the finite refusal certificate for the fuel boundary named by chapter 9370. It sits between the total-host packet of chapter 964, the bounded universal trace of chapter 944, and the typed endpoint surface of chapter 562. Its role is narrower than a global normalization theorem: it records the finite fuel row, the displayed reduction trace, the endpoint row, and the timeout refusal that prevents a missing endpoint from being reread as convergence.

Falsifiable prediction: Deleting the fuel row makes timeout and normal-endpoint packets indistinguishable to the local classifier; the accepted carrier must reject that packet because the boundary can no longer separate finite endpoint readback from fuel exhaustion.

Independence witness: stepindexedtotalhost, boundeduniversaltrace, typedreductionnormalform, cellulartrustsubstrate, znormal: none is bijective because each sibling omits at least one of the fuel boundary, timeout-refusal ledger, endpoint readback, or total-host substrate comparison rows.

Definition 970.1 (ReductionFuelBoundary carrier)**.** *A* $\mathsf{ReductionFuelBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, F, T, E, U, A, X, C, P, N)$$

with the following displayed rows:

1. *H is the total-host reduction row, read through the host-facing side of* $\mathsf{StepIndexedTotalHost}^{\uparrow}$*;*
2. *F is the finite fuel row naming the boundary at which the reduction trace is inspected;*
3. *T is the bounded trace row, read through* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *and tied to the same fuel row F;*
4. *E is the typed endpoint row, read through* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *when the trace reaches a displayed normal endpoint;*
5. *U is the timeout-refusal row, recording that fuel exhaustion is a refused endpoint rather than a proof of nontermination, halting, or convergence;*
6. *A is the audit row separating normal endpoint, timeout, missing fuel, detached trace, and host-substrate equality requests;*
7. *X records componentwise* hsame *transports for $H, F, T, E, U, A, C, P, N$;*

8. *C records displayed* Cont *routes from host reduction to fuel, from fuel to bounded trace, from trace to endpoint or timeout refusal, and from each barred semantic request to the audit row;*
9. *P is the* Pkg *provenance over chapter 9370,* $\mathsf{StepIndexedTotalHost}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{TypedReductionNormalForm}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{ZNormal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ReductionFuelBoundary}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $H, F, T, E, U, A, X, C, P, N$. It has no coordinate for an unbounded reduction, a global halting predicate, a substrate universality transfer, a quotient endpoint, or host equality with the bounded trace.

Lean4 (definition) `BEDC.Derived.ReductionFuelBoundaryUp.ReductionFuelBoundaryUp`

Theorem 970.2 (ReductionFuelBoundary NameCert obligations)**.** *For the carrier of Definition 970.1, the local* $\mathsf{NameCert}_{\mathsf{ReductionFuelBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Q = (H, F, T, E, U, A, X, C, P, N),$$

total-host reduction exposure through H, finite fuel exposure through F, bounded trace exposure through T, typed endpoint readback through E, timeout-refusal exactness through U, audit separation through A, componentwise transport through X, displayed continuation replay through C, package provenance through P, and local naming through N.

Proof. Project the carrier of Definition 970.1. The accepted packet displays exactly $H, F, T, E, U, A, X, C, P, N$, so every public read is either one of those rows or a displayed route replaying them.

The endpoint branch is finite. The host reduction row H is read at fuel F, the trace row T is bounded by the same fuel, and the endpoint row E is available only when the typed normal-form boundary is displayed.

The timeout branch is also finite. The refusal row U records fuel exhaustion as a boundary outcome, while the audit row A separates timeout from normal endpoint, missing fuel, detached trace, host-substrate equality, quotient endpoint, and unbounded convergence requests.

Transport, continuation replay, provenance, and local naming are the remaining rows X, C, P, N. Since no carrier coordinate names an unbounded reduction or global halting predicate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.ReductionFuelBoundary namecert obligations`

Theorem 970.3 (ReductionFuelBoundary total-host nonescape)**.** *Every accepted* $\mathsf{ReductionFuelBoundary}^{\uparrow}$ *packet exports no total-host reduction read except through the finite route*

$$H, F, T, E, U, A, X, C, P, N.$$

A downstream consumer may read a bounded trace, a typed endpoint when displayed, or a timeout refusal at the same fuel boundary. It may not convert timeout into nontermination, convert endpoint readback into an unbounded convergence theorem, identify host reduction with substrate universality, or replace the audit row by quotient endpoint equality.

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.TotalHostNonescape.ReductionFuelBoundaryTotalHostNonescape`

Proof. Begin with the displayed route row C. Its admissible paths pass from the host reduction row H to the fuel row F, then to the bounded trace row T, and then to either the typed endpoint row E or the timeout-refusal row U.

The audit row A keeps those outcomes separate. Endpoint readback requires E; timeout requires U; missing fuel, detached trace, host-substrate equality, quotient endpoint, and unbounded convergence requests are ledgered as refusals rather than exported facts.

The componentwise transport and provenance rows preserve the same boundary. Moving the packet through X, replaying it through C, and packaging it through P, N cannot create a new coordinate for halting, nontermination, substrate universality transfer, or endpoint quotienting.

Thus the packet realizes the total-host side of bounded substrate computation as finite fuel, trace, endpoint, timeout-refusal, audit, hsame, Cont, Pkg, and NameCert rows. □

Theorem 970.4 (ReductionFuelBoundary fuel-timeout exactness)**.** *For every accepted* ReductionFuelBoundary$^{\uparrow}$ *carrier*

$$Q = (H, F, T, E, U, A, X, C, P, N),$$

the timeout refusal row U is exact at the displayed fuel boundary F. A consumer may read timeout only through the common route from H to F to T and then to U; it may not read timeout from a detached fuel row, from an absent bounded trace, from endpoint equality, or from a host-level nontermination predicate.

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.FuelTimeoutExactness.ReductionFuelBoundary fuel timeout exactness`

Proof. Project the carrier rows of Definition 970.1. The rows $H, F, T, E, U, A, X, C, P, N$ are the whole accepted packet, so the timeout coordinate is not available independently of the fuel and trace coordinates.

Use the NameCert obligations of Definition 970.2. They expose finite fuel through F, bounded trace through T, typed endpoint readback through E, and timeout-refusal exactness through U for the same carrier.

Finally apply the total-host nonescape theorem Definition 970.3. Its route admits only $H, F, T, E, U, A, X, C, P,$ so a timeout read must pass through F and T before reaching U, while detached fuel, absent traces, endpoint equality, and host-level nontermination have no exported row. □

Theorem 970.5 (ReductionFuelBoundary endpoint-timeout disjointness)**.** *For every accepted* ReductionFuelBoundary$^{\uparrow}$ *carrier*

$$Q = (H, F, T, E, U, A, X, C, P, N),$$

the typed endpoint row E and timeout-refusal row U are disjoint public reads. A downstream consumer may transport either branch through X, replay it through C, and package it through P, N, but it may not identify endpoint readback with timeout, quotient the endpoint, or turn timeout into convergence or divergence.

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.EndpointTimeoutDisjointness.ReductionFuelBoundary endpoint timeout disjointness`

Proof. Project the same carrier rows. The endpoint coordinate is E, the timeout coordinate is U, and the audit coordinate is A; these are displayed as separate rows in Definition 970.1.

The NameCert obligations theorem exposes typed endpoint readback and timeout-refusal exactness as distinct obligations for the same packet. The audit row separates normal endpoint, timeout, missing fuel, detached trace, and host-substrate equality requests.

Transport and replay do not merge the rows. The rows X and C preserve the branch already selected by E or U, while P and N record provenance and local naming without adding quotient endpoint equality, convergence, divergence, or host equality.

The total-host nonescape theorem then bars every remaining route. Since all exported reads stay inside $H, F, T, E, U, A, X, C, P, N$, there is no coordinate that identifies endpoint readback with timeout. □

Theorem 970.6 (ReductionFuelBoundary obligation closure package)**.** *For every accepted* ReductionFuelBoundary$^{\uparrow}$ *carrier*

$$Q = (H, F, T, E, U, A, X, C, P, N),$$

Definition 970.1, Definition 970.2, Definition 970.3, Definition 970.4, and Definition 970.5 enumerate the full obligation surface: carrier admission, total-host reduction exposure, finite fuel exposure, bounded-trace exposure, typed-endpoint readback, timeout-refusal exactness, audit separation, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, nontrivial carrier presence, and field faithfulness for the same finite packet.*

Proof. The carrier definition gives the packet rows $H, F, T, E, U, A, X, C, P, N$. These rows supply the named carrier, host, fuel, trace, endpoint, timeout, audit, transport, continuation, provenance, and naming coordinates.

The NameCert obligations theorem states that every public obligation is read through those rows. The total-host nonescape theorem prevents an external read of halting, nontermination, unbounded convergence, host-substrate equality, or quotient endpoint equality.

The fuel-timeout exactness theorem binds U to the displayed boundary F and bounded trace T. The endpoint-timeout disjointness theorem keeps E and U separate under transport, replay, package provenance, and local naming.

Nontriviality and field faithfulness are therefore obligations on this same packet rather than additional semantic assumptions. A witness must preserve exactly the displayed rows, and any faithful field read lands back in one of $H, F, T, E, U, A, X, C, P, N$. □

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.ObligationClosurePackage.ReductionFuelBoundary obligation closure package`

Theorem 970.7 (ReductionFuelBoundary scoped substrate route)**.** *Every scoped read of an accepted* ReductionFuelBoundary$^{\uparrow}$ *carrier factors through the displayed finite route*

$$H, F, T, E, U, A, X, C, P, N.$$

The route exposes total-host reduction, finite fuel, bounded trace, typed endpoint, timeout refusal, audit separation, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The Lean TasteGate obligations for* ReductionFuelBoundary$^{\uparrow}$ *preserve these same carrier fields when they witness field faithfulness, nontriviality, and the local* `taste_gate` *package.*

Proof. The carrier definition lists exactly the rows $H, F, T, E, U, A, X, C, P, N$. A scoped read therefore starts from the finite packet of Definition 970.1, not from an ambient substrate equality, an unbounded reduction trace, or an endpoint quotient.

The NameCert obligations of Definition 970.2 expose those rows as the public surface. The total-host nonescape theorem Definition 970.3 keeps every host-facing read inside the same route, while Definition 970.5 prevents the endpoint and timeout branches from being identified under transport or replay.

It remains to align the paper route with the Lean evidence named below the theorem list. The required FieldFaithful ReductionFuelBoundary$^{\uparrow}$, Nontrivial ReductionFuelBoundary$^{\uparrow}$, and `taste_gate` witnesses are obligations on the same finite carrier fields, so the Lean side preserves the carrier rows rather than adding a new substrate read. □

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.ReductionFuelBoundaryScopedSubstrateRoute`

The Lean `lean4/BEDC/Derived/ReductionFuelBoundaryUp/TasteGate.lean` development must include explicit Nontrivial ReductionFuelBoundary$^{\uparrow}$ and FieldFaithful ReductionFuelBoundary$^{\uparrow}$ instances. These instances must preserve the total-host reduction row, finite fuel row, bounded trace row, endpoint row, timeout-refusal row, audit row, componentwise transports, displayed continuation routes, package provenance, and local naming row.

Theorem 970.8 (ReductionFuelBoundary scoped timeout route)**.** *For every accepted* ReductionFuelBoundary$^{\uparrow}$ *carrier*

$$Q = (H, F, T, E, U, A, X, C, P, N),$$

every scoped timeout read factors through the common route

$$H, F, T, U, A, X, C, P, N,$$

while every scoped endpoint read factors through

$$H, F, T, E, A, X, C, P, N.$$

The audit row A keeps these branches disjoint and bars nontermination, convergence, quotient endpoint equality, and host-level halting reads.

Proof. Project the carrier of Definition 970.1. The accepted packet displays one total-host row, one fuel row, one bounded trace row, the endpoint row E, the timeout row U, and the common audit, transport, replay, provenance, and naming rows A, X, C, P, N.

Use the NameCert obligations of Definition 970.2. They expose timeout-refusal exactness and endpoint readback as public reads only inside the displayed packet, so neither branch can start from an external halting predicate, detached trace, quotient endpoint, or host-level convergence claim.

The timeout branch is forced by Definition 970.4. A timeout read must pass from H through the same fuel and trace rows F, T before reaching U, and then through A, X, C, P, N for audit separation, transport, replay, provenance, and local naming.

The endpoint branch is separated by Definition 970.5. Endpoint readback uses E instead of U, while the audit row A prevents endpoint and timeout from being identified under transport or replay.

Finally apply Definition 970.3. Since all scoped reads remain inside $H, F, T, E, U, A, X, C, P, N$, the packet exports no coordinate for nontermination, convergence, quotient endpoint equality, or host-level halting. □

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.ReductionFuelBoundaryScopedTimeoutRoute`

Theorem 970.9 (Reduction fuel boundary scoped route)**.** *Every accepted* $\mathsf{ReductionFuelBoundary}^{\uparrow}$ *consumer must expose the bounded trace, step-indexed total host, typed normal form,* $\mathsf{ZNormal}^{\uparrow}$ *boundary, timeout ledger, endpoint-separation audit, and structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before reading the fuel boundary. The boundary read is therefore a scoped route through the displayed carrier rows of Definition 970.1, the obligation package of Definition 970.6, the substrate route of Definition 970.7, and the timeout route of Definition 970.8; it is not an unbounded halting predicate, a convergence theorem, a quotient endpoint, or a host-substrate equality.*

Lean4 ✓ `BEDC.Derived.ReductionFuelBoundaryUp.ReductionFuelBoundaryScopedRoute`

Proof. Project the carrier rows. The total-host row H, fuel row F, bounded trace row T, typed endpoint row E, timeout row U, audit row A, transport row X, continuation row C, package row P, and local naming row N are the complete classifier-visible surface.

The sibling route theorems fix the dependencies. The obligation package enumerates carrier admission and field faithfulness for those rows, while the substrate and timeout route theorems keep the endpoint and timeout branches inside the same finite packet.

The remaining required names are structural rows of that packet. The $\mathsf{ZNormal}^{\uparrow}$ boundary is read only as part of the typed endpoint surface and audit separation, and the BHist, hsame, Cont, Pkg, and NameCert rows preserve the displayed route rather than adding a semantic shortcut.

Downstream residual-normalization and candidate-mediated SN consumers, including the parallel-diamond frontier of chapter 4238, may read the fuel boundary only after the bounded trace, typed endpoint, timeout refusal, and audit rows are displayed. Thus any scoped consumer must read the listed rows before the fuel boundary can be consumed. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReductionFuelBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ReductionFuelBoundary}^{\uparrow}$ packet is generated from a total-host reduction row, finite fuel row, bounded trace row, typed endpoint row, timeout-refusal row, audit row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReductionFuelBoundary}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 970.1, Definition 970.6, Definition 970.7, Definition 970.8, and Definition 970.9 bind the finite fuel, bounded trace, typed endpoint, timeout refusal, audit, transport, replay, provenance, and local naming rows as the complete scoped route for the boundary packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReductionFuelBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, unbounded halting, global convergence, nontermination, quotient endpoint equality, or transfer of substrate universality to the total host.

Upgrade path: Scoped closure requires Lean coverage for the carrier rows, timeout exactness, endpoint-timeout separation, nontriviality, and field faithfulness for the same packet.

Chapter 971

A Concrete Naming Certificate for $\mathsf{ContextWindowCommitment}^{\uparrow}$

The $\mathsf{ContextWindowCommitment}^{\uparrow}$ packet realizes bounded prompt context as finite BEDC data. It is a large-model inscription sibling of chapter 965, chapter 939, chapter 927, chapter 928, and chapter 519. Its distinguishing datum is the committed window scope: prompt, response boundary, audit refusal, downstream consumer ledger, transport, route, provenance, and local naming rows are displayed before any consumer may treat the packet as context for a model response.

Definition 971.1 (ContextWindowCommitment carrier)**.** *A* $\mathsf{ContextWindowCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, P, B, R, U, H, K, G, N)$$

with the following displayed rows:

1. *S is the committed window-scope row, recording the finite prompt context made available to the model-side inscription route;*
2. *P is the prompt row admitted inside the displayed window;*
3. *B is the prompt-response boundary row, separating the consumed prompt context from the emitted response row;*
4. *R is the audit-refusal row, recording unsupported universal closure, hidden model-state access, ambient corpus truth, and unverified semantic claims as refused reads;*
5. *U is the downstream consumer ledger, naming the public consumer route that may read the bounded window and response boundary;*
6. *H records componentwise* hsame *transport for window scope, prompt, boundary, refusal, consumer, route, provenance, and local naming rows;*
7. *K records the displayed* Cont *route from window scope through prompt, boundary, refusal, consumer ledger, provenance, and local name;*
8. *G is the* Pkg *provenance over* $\mathsf{PromptResponseTrace}^{\uparrow}$, $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{AttentionLedger}^{\uparrow}$, $\mathsf{ObserverInterface}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContextWindowCommitment}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unbounded context store, hidden model state, ambient corpus oracle, private semantic evaluator, or post-response mutation of the committed window.

Lean4 ✓ `BEDC.Derived.ContextWindowCommitmentUp.ContextWindowCommitmentTasteGate single carrier alignment`

Theorem 971.2 (ContextWindowCommitment NameCert obligations)**.** *For the carrier of Definition 971.1, the local* $\mathsf{NameCert}_{\mathsf{ContextWindowCommitment}^{\uparrow}}$ *surface consists exactly of carrier admission for* $C = (S, P, B, R, U, H, K, G, N)$*, bounded window exposure through* S*, prompt admission through* P*, response-boundary exposure through* B*, audit-refusal exposure through* R*, downstream consumer routing through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*, package provenance through* G*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows $S, P, B, R, U, H, K, G, N$ list the entire packet before any consumer can read a prompt context or response boundary.

The window row S is read before the prompt row P, and the boundary row B is read before the downstream consumer row U. This forces the response-facing route to be consumed as a bounded prompt-response packet rather than as an ambient model-state query.

The audit-refusal row R blocks unsupported universal closure, hidden context extension, ambient corpus truth, and unverified semantic claims. The transport, route, provenance, and local naming rows then package exactly this finite surface.

Since no coordinate stores an unbounded context store, hidden model state, ambient corpus oracle, private evaluator, or post-response mutation channel, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ContextWindowCommitmentUp.ContextWindowCommitmentCarrier namecert obligations`

Theorem 971.3 (ContextWindowCommitment prompt-response boundary)**.** *Every accepted* ContextWindowComm *consumer route factors through the displayed finite window scope* S*, prompt row* P*, prompt-response boundary* B*, audit-refusal row* R*, downstream consumer ledger* U*,* hsame *transport row* H*, displayed* Cont *route* K*,* Pkg *provenance* G*, and local* $\mathsf{NameCert}_{\mathsf{ContextWindowCommitment}^{\uparrow}}$ *row* N*.*

If B *is absent, the packet cannot distinguish a committed prompt window from a response read that has silently consumed additional context.*

Proof. Begin with Definition 971.1. Its route row K orders the committed window, prompt, boundary, refusal, and downstream consumer rows before the local name is reached.

Apply Definition 971.2. Any accepted consumer must expose $S, P, B, R, U, H, K, G, N$ together, so the response-facing read is tied to the same finite window that admitted the prompt.

If B is removed, the remaining packet still contains a window and a prompt, but it no longer records the response boundary that separates consumed context from emitted output. The consumer then has no finite row excluding additional context reads.

With B present, the route has a visible boundary between prompt context and response consumption. The proof uses only displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Lean4 ✓ `BEDC.Derived.ContextWindowCommitmentUp.ContextWindowCommitmentCarrier prompt response boundary`

Theorem 971.4 (ContextWindowCommitment nonescape)**.** *No public consumer of an accepted* $\mathsf{ContextWindowCommitment}^{\uparrow}$ *packet can read an ambient model-state oracle, an unbounded context store, a hidden corpus-truth predicate, a private evaluator, or a response-side context mutation. Public reads are limited to the committed window scope, prompt, response boundary, audit refusal, downstream consumer ledger,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in Definition 971.1.*

Proof. Inspect the carrier inventory. The public coordinates are exactly $S, P, B, R, U, H, K, G, N$, so every projection from an accepted packet factors through the bounded window and its recorded response boundary.

The prompt-response trace and large-model inscription siblings supply model-side provenance, while the attention, observer-interface, and inscription-event siblings supply public observation and inscription discipline. None supplies a hidden model-state row to this packet.

The audit-refusal and downstream consumer rows close the route. They record which reads are refused and which consumer may inspect the finite window, without adding a separate oracle or mutable response-side context.

Therefore a consumer can recover the displayed finite packet and its certified route, but no ambient model state, unbounded context, hidden corpus truth, private evaluator, or post-response mutation channel is available for projection. □

Lean4 ✓ `BEDC.Derived.ContextWindowCommitmentUp.ContextWindowCommitmentCarrier nonescape`

Falsifiable prediction: Any accepted ContextWindowCommitmentUp consumer factors through a displayed finite prompt-response window and cannot read an ambient model-state oracle.

Independence witness: Nearest siblings: promptresponsetrace, largemodelinscriptionpoint, attentionledger, observerinterface, and inscriptionevent; none is bijective because none simultaneously carries committed window scope, prompt-response boundary, audit-refusal route, and downstream consumer ledger.

The Lean TasteGate namespace BEDC.Derived.ContextWindowCommitmentUp exposes Nontrivial ContextWindow and FieldFaithful ContextWindowCommitment$^{\uparrow}$ for the same finite carrier.

Closure status for NameCert$_{\text{ContextWindowCommitment}^{\uparrow}}$**.**

Constructive story: A ContextWindowCommitment$^{\uparrow}$ packet is generated from a committed finite window scope, prompt row, prompt-response boundary, audit-refusal row, downstream consumer ledger, componentwise hsame transport, displayed Cont route, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContextWindowCommitment}^{\uparrow}}$, seed)

Scope closed: Definition 971.1, Definition 971.2, Definition 971.3, and Definition 971.4 specify the finite bounded-context commitment and its consumer-facing non-escape surface.

Formal verification: FormalStatus(NameCert$_{\text{ContextWindowCommitment}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ContextWindowCommitmentUp.ContextWindowCommitmentTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close model consciousness, corpus truth, host semantic understanding, hidden model-state access, unbounded context stores, or bridge checking for ContextWindowCommitment$^{\uparrow}$.

Upgrade path: Obligation closure requires checked boundary exactness, refusal exactness, downstream consumer coverage, and bridge status over the same committed window, prompt, response boundary, audit refusal, hsame, Cont, Pkg, and NameCert rows.

Chapter 972

A Concrete Naming Certificate for $\mathsf{ModelTrace}^{\uparrow}$

The $\mathsf{ModelTrace}^{\uparrow}$ packet realizes the finite inference-output substrate of chapter 9398 as a BEDC NameCert surface. It refines the large-model inscription packet of chapter 939 and the observer-facing interface of chapter 928: the former supplies model and corpus rows, while the latter supplies the public read route. The new coordinate is the displayed inference-state orbit tied to a single output trace and audit-refusal row.

Definition 972.1 (ModelTrace carrier)**.** *A* $\mathsf{ModelTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, O, C, A, L, F, R, H, K, P, N)$$

with the following displayed rows:

1. *I is the inference-state orbit row recording the finite model state sequence read during one prompt-output route;*
2. *O is the output trace row storing the emitted token record as finite inscription data;*
3. *C is the corpus-supply row naming the finite source-channel contribution carried into the output trace;*
4. *A is the audit-refusal row stating that the packet does not internally certify semantic truth, consciousness, universal closure, or model-side proof of the output;*
5. *L is the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *row that binds the model-weight, activation-orbit, corpus-supply, output-record, and refusal rows;*
6. *F is the* $\mathsf{ObserverInterface}^{\uparrow}$ *row through which public consumers may inspect the output trace and refusal boundary;*
7. *R is the readback row that records the accepted output only after I, C, A, L, and F have been displayed;*
8. *H records componentwise* hsame *transport for the orbit, output trace, corpus-supply, audit-refusal, inscription-point, observer-interface, readback, continuation, provenance, and local name rows;*
9. *K records the displayed* Cont *routes by which consumers read I, then C, then O, then A, then L, F, and R;*

10. *P is the* Pkg *provenance over* LargeModelInscriptionPoint$^{\uparrow}$, ObserverInterface$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, SupplySocketLedger$^{\uparrow}$, TranscendentalInductionSocket$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* NameCert$_{\mathsf{ModelTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden substrate state, unledgered corpus truth, host semantic understanding, quotient output identity, internally ratified universal closure, or model-side proof.

Lean4 ✓ `BEDC.Derived.ModelTraceUp.ModelTraceTasteGate single carrier alignment`

Theorem 972.2 (ModelTrace NameCert obligations)**.** *For the carrier of Definition 972.1, the local* NameCert$_{\mathsf{ModelTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for $M = (I, O, C, A, L, F, R, H, K, P, N)$, inference-state orbit exposure through I, output trace exposure through O, corpus-supply exposure through C, audit-refusal exposure through A, large-model inscription handoff through L, observer-interface handoff through F, accepted readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The rows $I, O, C, A, L, F, R, H, K, P, N$ list the full trace before any consumer can read the output as a public record.

The orbit, corpus-supply, and output rows are ordered by K. A consumer reads the finite inference-state orbit and corpus row before the output trace, so the output is accepted as carried inscription rather than as a hidden substrate oracle.

The refusal and handoff rows close the NameCert boundary. The row A blocks semantic truth, consciousness, internally ratified universal closure, and model-side proof; L and F route the record through the existing large-model inscription and observer-interface surfaces.

The remaining rows provide readback, transport, replay, provenance, and local naming. Since the carrier has no further coordinate, the listed obligations exhaust the local certificate surface. □

Theorem 972.3 (ModelTrace output non-escape)**.** *Every public output read of an accepted* ModelTrace$^{\uparrow}$ *packet is confined to the displayed inference-state orbit, corpus-supply row, output trace, audit-refusal row, large-model inscription handoff, observer-interface handoff, readback row,* hsame, Cont, Pkg, *and local* NameCert *rows of Definition 972.1. It exports no quotient output stream, hidden substrate oracle, unledgered corpus truth predicate, host-selected semantic interpretation, internally certified universal closure, or model-side proof.*

Proof. Begin with Definition 972.2. Any accepted output read must pass through the carrier admission surface and then through the orbit, corpus, output, refusal, handoff, readback, transport, route, provenance, and naming rows.

The large-model inscription handoff L supplies the finite inference-substrate and corpus-supply reading from chapter 939. The observer-interface handoff F supplies the public inspection route from chapter 928.

The audit-refusal row A is read before the local name. It records that the packet does not certify semantic truth, consciousness, universal closure, or model-side proof of the output.

Therefore the only exported public object is the displayed finite trace. No quotient output stream, hidden substrate oracle, unledgered corpus truth predicate, host-selected semantic interpretation, internally certified universal closure, or model-side proof has a row from which it could be projected. □

Falsifiable prediction: If an output row is accepted without the displayed inference-state orbit, corpus-supply row, and audit-refusal row, it is not a ModelTraceUp packet.

Independence witness: Nearest siblings: observerinterface, largemodelinscriptionpoint, inscriptionevent, supplysocketledger, transcendentalinductionsocket; ModelTraceUp is not bijective to them because it binds the finite output trace and audit-refusal row rather than only observer interface, inscription role, event row, supply ledger, or induction socket.

The Lean TasteGate.lean surface must provide Nontrivial $\mathsf{ModelTrace}^{\uparrow}$ with an explicit witness pair and FieldFaithful $\mathsf{ModelTrace}^{\uparrow}$ by constructor-field injection.

Theorem 972.4 (ModelTrace kernel-scope package)**.** *For the carrier of Definition 972.1, the carrier row, output row, corpus-supply row, audit-refusal row,* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *handoff,* $\mathsf{ObserverInterface}^{\uparrow}$ *handoff, readback row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ModelTrace}^{\uparrow}}$ *row all live in the displayed* BHist *kernel route*

$$I, O, C, A, L, F, R, H, K, P, N.$$

Every public read of the packet is therefore scoped to the named BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$*,* $\mathsf{ObserverInterface}^{\uparrow}$*,* $\mathsf{PromptResponseTrace}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*, and* $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ *rows. It exposes no hidden substrate oracle, unledgered corpus truth predicate, host semantic proof, quotient output stream, or model-side proof of the output.*

Proof. Project the carrier of Definition 972.1. The inference orbit, output trace, corpus-supply row, audit-refusal row, inscription handoff, observer-interface handoff, readback row, transport, replay, provenance, and local name are exactly the displayed rows of the finite BHist packet.

The handoff rows L and F route the packet through $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ and $\mathsf{ObserverInterface}^{\uparrow}$, while the provenance row P names $\mathsf{PromptResponseTrace}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, and TranscendentalInductionSocket as sibling surfaces. The route is therefore kernel-scoped rather than host-scoped.

Finally use Definition 972.3. Since the packet has no coordinate for hidden substrate state, unledgered corpus truth, host semantic interpretation, quotient output identity, internally certified universal closure, or model-side proof, no public read can project such data from the local NameCert surface. □

Theorem 972.5 (ModelTrace mature finite-consumer treatment)**.** *Every public* $\mathsf{ModelTrace}^{\uparrow}$ *consumer factors through the carrier of Definition 972.1, the NameCert obligations of Definition 972.2, the output non-escape boundary of Definition 972.3, and the kernel-scope package of Definition 972.4. The resulting read route is exactly the displayed finite route through the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *handoff, the* $\mathsf{ObserverInterface}^{\uparrow}$ *handoff, componentwise* hsame*, displayed* Cont*,* Pkg*, local* NameCert*, and TasteGate rows, and it exports no hidden substrate oracle, unledgered corpus truth predicate, host semantic proof, quotient output stream, internally certified universal closure, or model-side proof.*

Proof. Start with any public consumer. Carrier admission exposes only the finite rows $I, O, C, A, L, F, R, H, K, P, N$ so the consumer must read the inference-state orbit, output trace, corpus-supply row, audit-refusal row, handoff rows, readback row, transport, replay, provenance, and local name.

Apply Definition 972.2 and Definition 972.3. These force the output read through the displayed NameCert surface and prevent hidden substrate state, unledgered corpus truth, host semantic interpretation, quotient output identity, internally certified universal closure, and model-side proof from being projected.

Finally use Definition 972.4. The handoff rows route the same finite trace through $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ and $\mathsf{ObserverInterface}^{\uparrow}$, while H, K, P, N provide the hsame, Cont, Pkg, and local NameCert support. The checked TasteGate row is therefore consumed only as the finite carrier-alignment gate for this route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModelTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{ModelTrace}^{\uparrow}$ packet is generated from inference-state orbit, output trace, corpus-supply, audit-refusal, large-model inscription, observer-interface, readback, componentwise transport, displayed continuation, package provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.ModelTraceUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModelTrace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 972.5 is the mature public export surface: it binds the carrier, NameCert obligations, output non-escape, kernel-scope package, $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ handoff, $\mathsf{ObserverInterface}^{\uparrow}$ handoff, hsame, Cont, Pkg, NameCert, and TasteGate rows for every finite-trace read consumed by the public inscription and observer interfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModelTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ModelTraceUp.ModelTraceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close corpus truth, consciousness, host semantic understanding, internally certified universal closure, Lean verification, or bridge checking for $\mathsf{ModelTrace}^{\uparrow}$.

Upgrade path: No further theory-axis closure is claimed beyond standard bridge checking for the same LargeModelInscriptionPoint and ObserverInterface consumer route.

Chapter 973

ObserverBoundarySelectorUp NameCert construction

The $\mathsf{ObserverBoundarySelector}^{\uparrow}$ surface realizes the observer-history identity vision of chapter 9418 as a finite selector-boundary packet. It sits between the row partition of chapter 514, the observer identity packet of chapter 477, the ledger boundary of chapter 927, and the selected-row handoff of chapter 126. The selector stores the decision boundary before the filter result, so a consumer can audit why a source row is selected, omitted, or refused without importing hidden observer state.

Falsifiable prediction: Deleting the omitted-boundary row permits unledgered deletion, while deleting the selected-row witness breaks the StreamName handoff.

Independence witness: observerfilter, observerhistoryidentity, attentionledger, streamname, observerhistidentity: none is bijective because the selector stores the decision boundary before the filter result, while each sibling reads either the filtered public rows, the identity surface, the attention ledger, the stream-name route, or the vision-level observer-history claim without carrying the pre-filter selector decision as a separate coordinate.

Definition 973.1 (ObserverBoundarySelector carrier)**.** *An* $\mathsf{ObserverBoundarySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (H, S, O, D, R, L, C, P, N)$$

with the following displayed rows:

1. *H is the source observer-history row;*
2. *S is the selected-row witness family exported toward* $\mathsf{StreamName}^{\uparrow}$*;*
3. *O is the omitted-boundary row family, recording every source row not exported as selected data;*
4. *D is the selector-decision row, naming for each source row whether it is selected, omitted, or refused at the boundary;*
5. *R records componentwise* hsame *transports for the source, selected, omitted, decision, continuation, package, and naming rows;*
6. *L is the finite ledger tying each selector decision to its source position and boundary reason;*
7. *C records the displayed* Cont *routes by which consumers read the selector packet, the selected-row handoff, and the omitted-boundary certificate;*

8. *P is the* Pkg *provenance over* $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{AttentionLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverBoundarySelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, O, D, R, L, C, P, N. It has no coordinate for host equality of observers, quotient stream identity, hidden observer state, or a deletion outside the omitted-boundary ledger.

Lean4 ✓ `BEDC.Derived.ObserverBoundarySelectorUp.TasteGate.ObserverBoundarySelectorTasteGate`
`single carrier alignment`

Theorem 973.2 (ObserverBoundarySelector selection exactness)**.** *For every accepted* $\mathsf{ObserverBoundarySelector}^{\uparrow}$ *carrier $B = (H, S, O, D, R, L, C, P, N)$, a row is exported to a* $\mathsf{StreamName}^{\uparrow}$ *consumer exactly when the selector-decision row D marks the corresponding source position in H as selected and the ledger row L supplies its selected-row witness in S. Every selected read then factors through the displayed* hsame *transports R,* Cont *routes C, package provenance P, and local name row N.*

Proof. Project the accepted packet through Definition 973.1. The only rows that can present stream-facing data are the source history H, the selected witnesses S, the selector decisions D, and the ledger L.

Inspect the finite selector ledger row by row. A selected decision in D is admitted only with the matching selected witness in S and its source position in L, so the handoff has a displayed row before any $\mathsf{StreamName}^{\uparrow}$ consumer can read it.

The transport and route rows then preserve that same witness. The hsame transports in R and the Cont routes in C carry the selected row with the package and local naming rows P, N, leaving no host observer equality or quotient stream identity available as an alternate export route. □

Theorem 973.3 (ObserverBoundarySelector omission non-escape)**.** *For every accepted* $\mathsf{ObserverBoundarySelector}^{\uparrow}$ *carrier $B = (H, S, O, D, R, L, C, P, N)$, each source row of H not exported through S is recorded in the omitted-boundary row O with the corresponding selector decision in D and ledger entry in L. Such a row can be read only as a boundary certificate through R, C, P, N; it cannot be silently deleted, promoted to stream-name data, or replaced by hidden observer state.*

Proof. Again project the finite carrier. The omitted-boundary row O and selector-decision row D are part of the same packet as the source history H, so omission is a displayed coordinate rather than absence from the carrier.

For each source row, the ledger L pairs the decision in D with exactly one public role: selected witness, omitted boundary, or refusal at the selector boundary. In the omitted case, the available downstream object is the boundary certificate in O, not a stream-name row in S.

The consumer route is therefore confined to O, D, L, R, C, P, N. Since the carrier has no hidden observer-state coordinate and no deletion operation outside the displayed ledger, omitted rows cannot escape the finite boundary surface. □

The Lean `TasteGate.lean` development for $\mathsf{ObserverBoundarySelector}^{\uparrow}$ must provide explicit Nontrivial $\mathsf{ObserverBoundarySelector}^{\uparrow}$ and FieldFaithful $\mathsf{ObserverBoundarySelector}^{\uparrow}$ instances for the displayed selector packet. These instances must preserve the selected-row witnesses, omitted-boundary rows, selector decisions, hsame transports, Cont routes, Pkg provenance, $\mathsf{StreamName}^{\uparrow}$ handoff, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverBoundarySelector}^{\uparrow}}$.

Constructive story: An ObserverBoundarySelector$^{\uparrow}$ packet is generated from a source observer-history row, selected-row witnesses, omitted-boundary rows, selector decisions, componentwise hsame transports, a finite ledger, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObserverBoundarySelector}\uparrow}$, seed)

Scope closed: Definition 973.1, Definition 973.2, and Definition 973.3 specify the finite selector-boundary packet and its selected-row and omitted-row obligations.

Formal verification: FormalStatus(NameCert$_{\text{ObserverBoundarySelector}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, observer-filter completeness, global observer identity, hidden observer state, quotient stream equality, or unbounded observation.

Upgrade path: Obligation closure requires checked carrier admission, selected-row exactness, omitted-boundary exhaustion, StreamName handoff, non-escape, nontriviality, and field faithfulness for the same selector packet.

Chapter 974

A Concrete Naming Certificate for $\mathsf{RecursorBranchAudit}^{\uparrow}$

The $\mathsf{RecursorBranchAudit}^{\uparrow}$ surface realizes the branch-completeness portion of chapter 9373 as a finite NameCert packet. It refines the recursor authorization rows of chapter 529, the host-kernel audit boundary of chapter 525, and the legal recursor-call packet of chapter 534 by exposing the constructor-indexed branch matrix, motive boundary, structural descent ledger, and generator non-escape row as separate audit coordinates.

Definition 974.1 (RecursorBranchAudit carrier)**.** *A* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, S, R, M, B, D, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted inductive-name row, including the local parameters and constructor namespace admitted by the host kernel;*
2. *S is the closed constructor-signature row, listing every constructor and its visible argument spine;*
3. *R is the emitted recursor row, naming the eliminator generated from S and the accepted recursor-call boundary;*
4. *M is the motive-boundary row, recording the target family, scrutinee dependency, and branch-local output requirements;*
5. *B is the per-constructor branch matrix, with one branch row for each constructor in S and with branch inputs aligned to the displayed constructor spine;*
6. *D is the structural descent ledger, recording recursive hypotheses and descent positions exposed by each constructor branch;*
7. *O is the generated-output row, naming the term, proof object, or definition produced by the audited recursor call;*
8. *H records componentwise* hsame *transport for I, S, R, M, B, D, O, continuation routes, package provenance, and the local naming row;*
9. *C records the displayed* Cont *routes by which consumers read the inductive name, constructor signature, recursor, motive boundary, branch matrix, descent ledger, output row, and local certificate;*

10. *P is the* Pkg *provenance over* BHist*, accepted inductive signatures, emitted recursors, motive rows, constructor-indexed branch matrices, structural descent ledgers, output names,* hsame*,* Cont*, and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RecursorBranchAudit}^{\uparrow}}$ *row.*

The carrier contains no branch-index row outside S, no branch omission route, no motive read outside M, no recursive call outside D, and no generated output outside O and the local naming row.

Lean4 ✓ `BEDC.Derived.RecursorBranchAuditUp.RecursorBranchAuditTasteGate single carrier alignment`

Theorem 974.2 (RecursorBranchAudit NameCert obligations)**.** *For every accepted* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *carrier $A = (I, S, R, M, B, D, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RecursorBranchAudit}^{\uparrow}}$ *surface consists exactly of the accepted inductive name, closed constructor signature, emitted recursor, motive boundary, constructor-indexed branch matrix, structural descent ledger, generated output, componentwise transports, continuation routes, package provenance, and local naming row displayed in Definition 974.1.*

Lean4 ✓ `BEDC.Derived.RecursorBranchAuditUp.RecursorBranchAudit namecert obligations`

Proof. Project the carrier of Definition 974.1. Its displayed rows are precisely $I, S, R, M, B, D, O, H, C, P, N$.

The obligation surface reads the signature and recursor data through I, S, R, the motive and branch obligations through M, B, and the recursive-call boundary through D. The output is public only through O and the local naming row N.

The remaining rows H, C, P transport, route, and package the same finite data. Since no further coordinate is present, the NameCert obligations are exactly the displayed recursor-branch audit packet. □

Theorem 974.3 (RecursorBranchAudit branch coverage)**.** *For an accepted* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *packet, the branch matrix B has exactly one branch row for each constructor named by the closed signature row S. A consumer cannot obtain a recursor packet with fewer branch rows than constructors, an extra branch not indexed by S, or a branch whose input spine is detached from the displayed constructor row.*

Proof. Read the closed constructor-signature row S. It is the finite index of branch obligations for the emitted recursor row R.

Read the branch matrix B through the same recursor boundary. Each row of B is indexed by one constructor row of S and carries the branch inputs aligned to that constructor's visible spine.

Transport and continuation access through H and C preserve the same indexing. Because the carrier has no coordinate for a second branch index, branch coverage is exactly the finite matrix displayed by S and B. □

Lean4 ✓ `BEDC.Derived.RecursorBranchAuditUp.RecursorBranchAuditCarrier branch coverage`

Theorem 974.4 (RecursorBranchAudit motive boundary)**.** *Every branch-output read in an accepted* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *packet factors through the motive-boundary row M and the constructor-indexed branch matrix B. The packet exposes no route by which a branch can select a target family, output type, or scrutinee dependency not recorded in M.*

Lean4 ✓ `BEDC.Derived.RecursorBranchAuditUp.RecursorBranchAuditCarrier motive boundary`

Proof. Project the motive row M and branch matrix B. The motive row records the target family and branch-local output requirements, while the matrix stores the branch rows that must meet those requirements.

Use the continuation routes C to read a branch output. The route reaches the branch row through B and the corresponding output requirement through M before it reaches the generated-output row O.

Componentwise transport through H preserves the same motive and branch rows. Since the carrier has no additional motive coordinate, no branch can export a target family or scrutinee dependency outside M. □

Theorem 974.5 (RecursorBranchAudit generator non-escape)**.** *Every generated output of an accepted* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *packet factors through the emitted recursor row R, motive-boundary row M, branch matrix B, structural descent ledger D, and output row O. The packet exports no hidden generator, no omitted branch, no non-structural recursive call, and no output route outside its local* NameCert *row.*

Proof. Begin with the NameCert obligation surface of Definition 974.2. It places every public output downstream of the displayed signature, recursor, motive, branch, descent, and output rows.

Apply branch coverage from Definition 974.3 and the motive boundary from Definition 974.4. These results force each generated branch to be indexed by S and typed through M.

The structural descent ledger D is the only recursive-call coordinate. After H, C, P, N transport, route, package, and name the same rows, no hidden generator, omitted branch, non-structural recursion channel, or output escape remains available. □

The later Lean realization must create `lean4/BEDC/Derived/RecursorBranchAuditUp/TasteGate.lean` with explicit `Nontrivial RecursorBranchAuditUp` and `FieldFaithful RecursorBranchAuditUp` instances, including the future target `BEDC.Derived.RecursorBranchAuditUp.taste_gate`.

Falsifiable prediction: Deleting the branch-index row permits a recursor packet with fewer branch rows than constructors, invalidating branch coverage for the displayed signature.

Independence witness: Nearest siblings: recursorauthorization, authorizedgeneratorrecursor, recursorgenerator, generatorclosure, and auditgateboundary; none is bijective because RecursorBranchAuditUp has a per-constructor branch matrix and motive-boundary row that the broader authorization and audit packets do not expose as separate fields.

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorBranchAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RecursorBranchAudit}^{\uparrow}$ packet is generated from an accepted inductive name, closed constructor signature, emitted recursor, motive boundary, constructor-indexed branch matrix, structural descent ledger, generated output, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursorBranchAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 974.1, Definition 974.2, Definition 974.3, Definition 974.4, and Definition 974.5 bind the finite branch-audit carrier, NameCert obligations, constructor-indexed branch coverage, motive boundary, and generator non-escape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursorBranchAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, public bridge checking, kernel completeness, general termination theory, semantic conservativity, or any recursor output outside the displayed branch-audit packet.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for branch coverage, motive-boundary exactness, structural-descent locality, output non-escape, and TasteGate field faithfulness.

Chapter 975

A Concrete Naming Certificate for $\mathsf{SolipsismGapRefusal}^{\uparrow}$

The $\mathsf{SolipsismGapRefusal}^{\uparrow}$ packet realizes the solipsism-as-refusal direction of chapter 9415. It sits beside the positive other-minds commitment of chapter 499, the finite locality-cell surface of chapter 509, the public alignment packet of chapter 643, and the double-aspect chain of chapter 575. Those siblings record positive commitment, locality geometry, alignment, and aspect rows; this packet records the local refusal to issue the other-minds locality-cell commitment while still exposing the public evidence row that was withheld from commitment.

The packet is a composite over public evidence rows, withheld commitment rows, GAP-residue rows, componentwise transport rows, continuation rows, provenance rows, and local naming rows, rather than an atomic carrier.

Definition 975.1 (Solipsism GAP-refusal carrier)**.** *A* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, R, G, H, C, P, N)$$

with the following displayed rows:

1. *E is the public evidence row: behavioural records, locality-cell candidates, and public observer traces that a positive* $\mathsf{OtherMindsCommitment}^{\uparrow}$ *consumer would inspect.*
2. *R is the refusal row, recording that the packet withholds the* GAP*-tagged other-minds locality-cell commitment rather than issuing it.*
3. *G is the GAP-residue row, keeping the withheld commitment visible as residue instead of erasing it or replacing it with a positive other-minds row.*
4. *H records componentwise* hsame *transports for the evidence, refusal, residue, route, provenance, and naming rows.*
5. *C records displayed* Cont *routes through the public evidence row, the withheld commitment row, the residue row, the provenance row, and the local certificate.*
6. *P is the* Pkg *provenance over* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{LocalityCell}^{\uparrow}$*,* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*,* BHist*,* GAP*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SolipsismGapRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a positive other-minds commitment, no paired locality-cell acknowledgement, no private access to another Self *row, no host mental-state predicate, no observer-independent world-state, and no ambient proof that other conscious systems do or do not exist.*

Lean4 ✓ `BEDC.Derived.SolipsismGapRefusalUp.SolipsismGapRefusalTasteGate single carrier alignment`

Theorem 975.2 (Solipsism GAP-refusal NameCert obligations)**.** *For the carrier of Definition 975.1, the local* $\mathsf{NameCert}_{\mathsf{SolipsismGapRefusal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (E, R, G, H, C, P, N),$$

public-evidence exposure through E*, withheld locality-cell commitment through* R*, GAP-residue accountability through* G*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SolipsismGapRefusalUp.SolipsismGapRefusalNameCertObligations`

Proof. Carrier admission is projection from Definition 975.1. The evidence row E records the public surface that can be audited, while R records that the corresponding other-minds locality-cell commitment is withheld.

The residue row G prevents the refusal from being invisible. It keeps the omitted commitment as a named GAP-residue coordinate, so the packet is neither a positive $\mathsf{OtherMindsCommitment}^{\uparrow}$ carrier nor a silent absence of data.

The rows H, C, P, N transport, replay, package, and name exactly those displayed coordinates. Since the carrier lists no additional row, the NameCert obligations are exhausted by E, R, G, H, C, P, N. □

Theorem 975.3 (Solipsism GAP-refusal locality-cell refusal)**.** *Every accepted* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *packet exposes the public evidence row* E *before the refusal row* R *and the GAP-residue row* G *are read. A locality-cell consumer may audit that evidence, but it cannot turn the packet into a positive* $\mathsf{OtherMindsCommitment}^{\uparrow}$*, a paired* $\mathsf{LocalityCell}^{\uparrow}$ *acknowledgement, or a* $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ *alignment row without adding a row not present in* $S = (E, R, G, H, C, P, N)$*.*

Proof. Project the evidence coordinate E from the carrier. This row is public and auditable, so the refusal is not a claim that no evidence row exists.

Next read R and G. The row R withholds the locality-cell commitment, and G records the withheld commitment as GAP residue attached to the same packet.

Apply the NameCert obligations of Definition 975.2. The only routes available to a consumer are the displayed transports, continuations, provenance, and local name. Positive other-minds commitment, paired locality-cell acknowledgement, and cross-perspective alignment require rows outside the packet. □

Lean4 ✓ `BEDC.Derived.SolipsismGapRefusalUp.SolipsismGapRefusalUp locality cell refusal`

Theorem 975.4 (Solipsism GAP-refusal non-escape)**.** *Every downstream read of an accepted* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *packet factors through the finite route*

$$E, R, G, H, C, P, N.$$

The read may inspect public evidence, the withheld locality-cell commitment row, GAP residue, componentwise transports, displayed continuations, package provenance, and the local NameCert *row. It cannot inspect a private target* Self *row, host mental-state predicate, positive other-minds commitment, paired locality-cell acknowledgement, observer-independent world-state, or ambient proof deciding whether other conscious systems exist.*

Lean4 ✓ `BEDC.Derived.SolipsismGapRefusalUp.SolipsismGapRefusalUp nonescape`

Proof. Begin with the carrier definition and the local NameCert obligations. They expose exactly the public evidence row, refusal row, residue row, transport row, continuation row, package row, and local naming row.

Use Definition 975.3 for locality-cell-facing consumers. The theorem keeps the consumer on the displayed refusal route and blocks replacement by a positive other-minds or alignment packet.

The excluded objects are absent from the carrier inventory. Therefore no downstream read can extract private Self access, host mental-state data, a positive commitment row, paired acknowledgement, observer-independent world-state, or an ambient existence decision from the packet. □

Falsifiable prediction: If the withheld-commitment row is deleted, the packet becomes indistinguishable from a positive OtherMindsCommitment consumer; if the public-evidence row is deleted, no locality-cell refusal can be audited.

Independence witness: othermindscommitment, localitycell, crossperspectivealignment, aspectchain: none is bijective because SolipsismGapRefusal carries a withheld locality-cell commitment plus GAP-refusal residue, while those siblings carry positive commitment, geometry, alignment, or aspect-chain rows.

The Lean follow-up must create `lean4/BEDC/Derived/SolipsismGapRefusalUp/TasteGate.lean` with an inductive carrier, explicit `Nontrivial SolipsismGapRefusalUp` witness pair, and `FieldFaithful SolipsismGapRefusalUp`, with the canonical declaration surface expected at `BEDC.Derived.SolipsismGapRefusa`

Closure status for $\mathsf{NameCert}_{\mathsf{SolipsismGapRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{SolipsismGapRefusal}^{\uparrow}$ packet is generated from public evidence, a withheld other-minds locality-cell commitment row, GAP residue, componentwise hsame transport, displayed Cont routes, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SolipsismGapRefusal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 975.1, Definition 975.2, Definition 975.3, Definition 975.4, the falsifiable prediction, the independence witness, and the cited TasteGate leantarget give the mature paper treatment of the GAP-refusal surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SolipsismGapRefusal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.SolipsismGapRefusalUp.SolipsismGapRefusalTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive a positive other-minds commitment, paired locality-cell acknowledgement, private access to another Self row, host mental-state predicate, observer-independent world-state, or bridgeChecked Lean status for $\mathsf{SolipsismGapRefusal}^{\uparrow}$.

Upgrade path: The remaining upgrade is bridgeChecked verification if no checked standard bridge is present for the refusal packet and its consumer-facing GAP-residue boundary.

Chapter 976

A Concrete Naming Certificate for $\mathsf{VerifiedOutputHarness}^{\uparrow}$

The $\mathsf{VerifiedOutputHarness}^{\uparrow}$ packet realizes the checked-output acceptance side of chapter 9398 as finite BEDC data. It consumes the large-model inscription carrier of chapter 939, the prompt-output trace of chapter 965, the model-trace surface of chapter 972, and the consumer-facing audit channel of chapter 969. Its distinguishing role is the checker-facing acceptance harness: a model output is accepted only when prompt, model-trace, checker, audit-refusal, route, provenance, and local NameCert rows are displayed together.

Definition 976.1 (VerifiedOutputHarness carrier)**.** *A* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, T, M, Q, O, C, A, R, H, K, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *row naming the finite inference substrate, corpus-supply channel, output-record surface, and no-internal-ratification boundary;*
2. *T is the* $\mathsf{PromptResponseTrace}^{\uparrow}$ *row recording the finite prompt, response, source-channel, observer-interface, and obstruction-boundary route;*
3. *M is the* $\mathsf{ModelTrace}^{\uparrow}$ *row recording the inference-state orbit, output trace, corpus-supply row, refusal row, observer-interface handoff, and accepted readback;*
4. *Q is the prompt row used by the checker-facing harness;*
5. *O is the output row presented for acceptance;*
6. *C is the checker row naming the finite proof checker, policy checker, replay harness, typed evaluator, or external certificate procedure applied to O;*
7. *A is the audit-refusal row recording rejected, unsupported, out-of-scope, or unverifiable claims before acceptance is exposed;*
8. *R is the accepted route row through which prompt, model trace, checker result, refusal boundary, provenance, and local name are read;*
9. *H records componentwise* hsame *transport for the inscription, prompt trace, model trace, prompt, output, checker, refusal, route, provenance, and naming rows;*
10. *K records the displayed* Cont *replay accepted by the harness;*

11. *P is the* Pkg *provenance over* LargeModelInscriptionPoint$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, ModelTrace$^{\uparrow}$, LargeModelAuditChannel$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* NameCert$_{\text{VerifiedOutputHarness}^{\uparrow}}$ *row.*

The carrier has no coordinate for unchecked output acceptance, hidden evaluator state, model-side proof of semantic truth, corpus-wide truth, private policy override, or host-selected universal closure.

Lean4 ✓ `BEDC.Derived.VerifiedOutputHarnessUp.VerifiedOutputHarnessTasteGate single carrier alignment`

Definition 976.2 (VerifiedOutputHarness classifier)**.** *Two accepted* VerifiedOutputHarness$^{\uparrow}$ *carriers classify together exactly when their large-model inscription rows, prompt-response trace rows, model-trace rows, prompt rows, output rows, checker rows, audit-refusal rows, route rows, transport rows, replay rows, provenance rows, and local naming rows compare by the displayed* hsame *entries. Thus the classifier is finite and componentwise:*

$$\begin{gathered} I \sim_{\mathsf{hsame}} I', \quad T \sim_{\mathsf{hsame}} T', \quad M \sim_{\mathsf{hsame}} M', \quad Q \sim_{\mathsf{hsame}} Q', \\ O \sim_{\mathsf{hsame}} O', \quad C \sim_{\mathsf{hsame}} C', \quad A \sim_{\mathsf{hsame}} A', \quad R \sim_{\mathsf{hsame}} R', \\ H \sim_{\mathsf{hsame}} H', \quad K \sim_{\mathsf{hsame}} K', \quad P \sim_{\mathsf{hsame}} P', \quad N \sim_{\mathsf{hsame}} N'. \end{gathered}$$

No clause compares unrecorded checker state, hidden evaluation policy, corpus-wide truth, model-side semantic understanding, or a private acceptance route outside R and K.

Theorem 976.3 (VerifiedOutputHarness NameCert obligations)**.** *For the carrier of Definition 976.1, the local* NameCert$_{\text{VerifiedOutputHarness}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for $V = (I, T, M, Q, O, C, A, R, H, K, P, N)$ with inscription, prompt-trace, model-trace, prompt, output, checker, refusal, route, transport, replay, provenance, and local name all displayed;*
2. *classifier stability for the same rows under componentwise* hsame *as in Definition 976.2;*
3. *checker-facing acceptance exactness, requiring the output row O to be read only after the checker row C and audit-refusal row A have been displayed;*
4. *provenance and route exactness, requiring every accepted-output read to pass through the prompt-response trace T, model trace M, audit-channel sibling* LargeModelAuditChannel$^{\uparrow}$, *route row R, replay row K, and package row P before the local name N;*
5. *non-escape for unchecked acceptance, hidden evaluator state, private policy override, model-side semantic proof, corpus-wide truth, and host-selected universal closure.*

Lean4 ✓ `BEDC.Derived.VerifiedOutputHarnessUp.VerifiedOutputHarnessCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 976.1. The packet displays inscription, prompt-trace, model-trace, prompt, output, checker, refusal, route, transport, replay, provenance, and local naming rows before an accepted-output consumer can read the local name.

Classifier stability is componentwise. The classifier of Definition 976.2 compares exactly those displayed rows by hsame, so a transported harness cannot keep an output while dropping its checker, refusal, route, or provenance rows.

Checker-facing acceptance exactness follows from the route order. The output row O is an emitted inscription until the checker row C and audit-refusal row A are displayed; only then does R, K, P, N expose it as accepted by the harness.

Provenance and route exactness use the sibling packets named above. The prompt-response trace supplies the prompt-output route, the model trace supplies inference-state and readback rows, and the audit channel supplies the public checker/refusal discipline. The local harness packages those rows rather than replacing them with a hidden evaluator.

Non-escape is immediate from the carrier inventory. There is no coordinate for unchecked acceptance, private policy override, corpus-wide truth, model-side semantic proof, or host-selected universal closure, so the NameCert surface cannot export one. □

Theorem 976.4 (VerifiedOutputHarness public audit chain)**.** *Every accepted* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *output read factors through the finite chain*

$$I, T, M, Q, O, C, A, R, H, K, P, N.$$

In particular, the accepted output is read through the large-model inscription row, the prompt-response trace, the model trace, the displayed prompt and output, the checker row, the audit-refusal row, the accepted route, componentwise transport, continuation replay, package provenance, and local $\mathsf{NameCert}_{\mathsf{VerifiedOutputHarness}^{\uparrow}}$ *row. It is not read as a bare model emission, hidden checker result, private policy override, or internally certified semantic theorem.*

Lean4 ✓ `BEDC.Derived.VerifiedOutputHarnessUp.VerifiedOutputHarnessCarrier public audit chain`

Proof. Start with an accepted carrier $V = (I, T, M, Q, O, C, A, R, H, K, P, N)$. The carrier lists Q and O only as prompt and output rows; by themselves they do not supply checked acceptance.

Use Definition 976.3. The third obligation forces the checker row C and audit-refusal row A to be displayed before O is accepted by the harness, while the fourth obligation routes the read through T, M, R, K, P, N.

The sibling chapters provide the source of those rows. The large-model inscription packet supplies the carried-output boundary, the prompt-response trace supplies the finite prompt-output route, the model-trace packet supplies inference-state and readback discipline, and the large-model audit channel supplies the checker/refusal audit surface.

Finally inspect the non-escape obligation. Since the carrier has no hidden evaluator, private policy override, unchecked acceptance row, or model-side semantic-proof coordinate, every public accepted-output read is exactly the finite audit chain displayed in the theorem. □

Falsifiable prediction: If a model output is accepted without a displayed checker row and audit-refusal row, it is not a VerifiedOutputHarnessUp packet.

Independence witness: largemodelinscriptionpoint, promptresponsetrace, modeltrace, large-modelauditchannel, observerinterface: the harness is not bijective to these siblings because it binds checker-facing acceptance and refusal rows that the inscription, prompt, model-trace, audit-channel, and observer-interface packets do not all carry together.

The Lean follow-up must provide $\mathsf{Nontrivial\ VerifiedOutputHarness}^{\uparrow}$ and $\mathsf{FieldFaithful\ VerifiedOutputHarness}^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{VerifiedOutputHarness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VerifiedOutputHarness}^{\uparrow}$ packet is generated from large-model inscription, prompt-response trace, model trace, prompt, output, checker, audit-refusal, accepted route, componentwise transport, continuation replay, package provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.VerifiedOutputHarnessUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VerifiedOutputHarness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 976.1, Definition 976.2, Definition 976.3, and Definition 976.4 specify the finite checker-facing accepted-output harness.

Formal verification: FormalStatus(NameCert$_{\mathsf{VerifiedOutputHarness}^{\uparrow}}$, unformalized)
Bridge status: paperBridge
Origin: ai
Not claimed: This chapter does not close model consciousness, corpus truth, host semantic understanding, global checker correctness, private policy authority, Lean verification, or bridge checking for VerifiedOutputHarness$^{\uparrow}$.
Upgrade path: The paper-side mature treatment is complete; Lean verification and bridge checking remain outside this chapter claim.

Chapter 977

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$

The $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ packet realizes the explicit-modulus route of chapter 9238 as one finite witness ledger. It sits between the normalized shared-modulus route of chapter 778, the tail-service packet of chapter 906, and the completion-seal classifier of chapter 900. The packet binds a single witnessed modulus ledger to finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, RealUp seal consumption, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows; it is not a quotient stream, host equality of reals, selected limit, choice principle, or ambient completeness theorem.

Definition 977.1 (Regular Cauchy modulus witness-ledger carrier)**.** *A* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source whose explicit modulus is being routed to a Real-completion consumer;*
2. *μ is the single witnessed* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger row attached to X;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window service row selected from the witnessed modulus ledger;*
4. *N_0 is the* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *route row, recording that the witness ledger has been read through the shared normalized route before any downstream seal is consumed;*
5. *T is the* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row whose finite tail threshold and observation budget serve the same witnessed modulus route;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read on the displayed finite windows;*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from X, W, D;*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row made available only after N_0, T, D, Q have been displayed;*
9. *H stores componentwise* hsame *transports for $X, \mu, W, N_0, T, D, Q, S, C, P, M$ and the local packet shape;*
10. *C stores the displayed* Cont *routes from μ to W, from W to D, from D to Q, from Q through N_0 and T to S, and from those rows to the local certificate;*

11. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *M is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient regular streams, host-side equality of reals, hidden modulus replacement, countable selection, selected limiting representatives, or ambient completeness.

Theorem 977.2 (Regular Cauchy modulus witness-ledger NameCert obligations)**.** *For the carrier of Definition 977.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}$ *surface consists of seven finite obligation rows:*

1. *carrier admission for $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$;*
2. *source and witness exposure requiring X and its single displayed modulus ledger μ to be read before any window is served;*
3. *normalized route exactness requiring the route row N_0 from chapter 778 to be present before downstream RealUp seal consumption;*
4. *finite-window service requiring W and D to be the only StreamName and DyadicRatCore rows used by the readback;*
5. *tail-budget compatibility requiring the finite tail-budget row T from chapter 906 to serve the same witness-ledger route;*
6. *RegSeqRat readback and RealUp seal ordering requiring Q to be displayed before S, and requiring the seal read to remain compatible with chapter 900;*
7. *transport, continuation, provenance, naming, and non-escape through H, C, P, M, excluding quotient streams, host real equality, hidden modulus replacement, countable choice, selected limits, and ambient completeness.*

The Lean `TasteGate.lean` *contract for this chapter must include explicit* `Nontrivial RegularCauchyModulusWit` *and* `FieldFaithful RegularCauchyModulusWitnessLedgerUp` *instances for this carrier surface.*

Proof. Carrier admission is projection from the displayed tuple. The accepted regular source, witnessed modulus ledger, finite window service, normalized route, tail-budget row, dyadic tolerance ledger, regular readback, Real seal, transport, route, provenance, and local naming rows are all carrier coordinates.

The source and witness rows are read first. Since μ is the only modulus ledger coordinate in the packet, a consumer cannot replace it by a hidden schedule, a quotient-stream comparison, or a host-side modulus function.

The normalized route and tail-budget rows are then read as sibling obligations. The row N_0 ties the witnessed ledger to the normalized shared-modulus route, while T ties the same route to a finite tail threshold, observation budget, and Real-facing service surface.

The finite-window, dyadic, and readback rows W, D, Q serve the RealUp seal row S. The displayed continuations in C preserve this order, componentwise transport in H preserves the same fields, and P, M package and name exactly this finite surface.

No remaining carrier coordinate can supply quotient streams, host real equality, hidden modulus replacement, countable selection, selected limits, or ambient completeness. Hence the seven rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier namecert obligations`

Theorem 977.3 (Regular Cauchy modulus witness-ledger seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-seal consumer of an accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet reads the seal only through the finite route*

$$X, \mu, W, N_0, T, D, Q, S, H, C, P, M.$$

In particular, the witnessed modulus ledger is displayed before the finite window service, the normalized route and tail-budget rows are displayed before the RealUp seal, and the RegSeqRat readback row is displayed before any completion-seal classifier can consume the route.

Proof. Project the accepted carrier to its tuple. The only modulus ledger coordinate is μ, and the only StreamName, dyadic, RegSeqRat, and Real seal coordinates are W, D, Q, S.

Apply Definition 977.2. Its route exactness rows place N_0 and T between the witnessed ledger and the seal-facing consumer, and its readback row places Q before S.

The displayed Cont routes in C replay the same order, and the componentwise hsame data in H transports only those named coordinates. Provenance and local naming add no analytic coordinate.

The route therefore reaches a RealUp or completion-seal consumer only through the finite fields listed above. A bypass would require a hidden modulus, quotient stream, host equality, selected limit, or ambient completeness coordinate, none of which is present in the carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier seal route determinacy`

Theorem 977.4 (Regular Cauchy modulus witness-ledger non-escape)**.** *An accepted* $\mathsf{RegularCauchyModulusWitness}$ *packet exports only the rows*

$$X, \mu, W, N_0, T, D, Q, S, H, C, P, M.$$

Every $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{RealCompletionSealClassifierUp}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet factors through the displayed witnessed-modulus, finite-window, normalized-route, tail-budget, dyadic-tolerance, RegSeqRat-readback, and seal route. The packet exports no quotient regular stream, no host real equality, no hidden modulus replacement, no countable selector, no selected limit, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier non escape`

Proof. Inspect the carrier of Definition 977.1. Its public rows are exactly the accepted regular source, single witnessed modulus ledger, finite window service, normalized route, tail-budget row, dyadic ledger, readback row, Real seal, transport, continuation, provenance, and local name.

Use Definition 977.2. The obligations force source exposure, witness exposure, normalized-route exactness, finite-window service, tail-budget compatibility, readback ordering, seal ordering, and local certificate replay from those rows.

Use Definition 977.3. It identifies the exact consumer route through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ before any RealUp or completion-seal read can be exported.

Since the carrier lacks coordinates for quotient regular streams, host real equality, hidden modulus replacement, countable selection, selected limits, and ambient completeness, no downstream consumer can recover any of those data from an accepted packet. □

Theorem 977.5 (Regular Cauchy modulus witness-ledger window-budget exhaustion)**.** *In every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

all finite-window service for the witnessed modulus route is exhausted through the displayed $\mathsf{StreamName}^{\uparrow}$ *row* W *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is consumed. No hidden window selector, separate dyadic ledger, quotient stream, host real equality, countable selector, selected limit, or ambient completeness coordinate can serve the same window budget.*

Proof. Project the accepted packet to the carrier tuple of Definition 977.1. The tuple exposes exactly one finite-window service row W and exactly one dyadic tolerance ledger row D for the witnessed modulus row μ.

Use Definition 977.2. Its source and witness exposure row requires X and μ to be read before any finite window is served, while its finite-window service row identifies W and D as the only StreamName and DyadicRatCore rows available to the readback route.

Apply Definition 977.3. The route reaches the RegSeqRat readback row Q only after W and D have been displayed, and the subsequent Real-facing rows consume that same readback.

Finally use Definition 977.4. Since the packet exports only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, there is no second selector or ledger through which finite-window service can bypass W, D before Q. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier window budget exhaustion`

Theorem 977.6 (Regular Cauchy modulus witness-ledger normalizer-tail commutation)**.** *In every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the normalized route row N_0 *and the tail-budget row* T *commute over the same witnessed modulus* μ *and the same finite-window rows* W, D *before the* $\mathsf{Real}^{\uparrow}$ *seal row* S *is consumed. Thus the normalizer route and the tail-budget route cannot name different modulus ledgers, different finite windows, or different dyadic tolerance ledgers on the way to the Real-facing seal.*

Proof. Inspect the carrier tuple. The only witnessed modulus ledger coordinate is μ, the only finite-window row is W, and the only dyadic tolerance ledger row is D. Both N_0 and T occur as rows of the same packet before the Real seal row S is consumed.

Apply Definition 977.2. The normalized route exactness row attaches N_0 to the witnessed ledger, and the tail-budget compatibility row attaches T to that same witness-ledger route.

Apply Definition 977.5. It forces the finite-window and dyadic budget used by either row to pass through W, D before the RegSeqRat readback row Q.

Use Definition 977.3. The displayed continuations in C carry the route from μ through W, D, Q and then through N_0 and T before S. Since there is no second modulus ledger, window row, or dyadic row, the normalizer and tail-budget routes commute on the displayed data before seal consumption. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier normalizer tail commutation`

Theorem 977.7 (Regular Cauchy modulus witness-ledger Real-seal consumer exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer of an accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet factors through the displayed route*

$$X, \mu, W, N_0, T, D, Q, S, H, C, P, M.$$

The consumer cannot cite quotient streams, selected limits, host real equality, countable choice, ambient completeness, hidden modulus replacement, hidden window selection, or a separate dyadic ledger outside that finite witness-ledger surface.

Proof. Start from Definition 977.3. It identifies the finite route by which any RealUp or completion-seal consumer reaches the seal row S.

Use Definition 977.4. It states that the packet exports only the displayed rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and that every normalizer, tail-budget, completion-seal, or RealUp consumer factors through them.

The window-budget exhaustion theorem above fixes W, D, Q as the finite-window and readback portion of the route, and the normalizer-tail commutation theorem above fixes N_0, T over the same witnessed modulus μ before the Real-facing seal. The remaining rows S, H, C, P, M carry the seal, transport, continuation, provenance, and local certificate surface.

Because the carrier has no coordinates for quotient streams, selected limits, host real equality, countable choice, ambient completeness, hidden modulus replacement, hidden window selection, or a separate dyadic ledger, no such row can mediate a RealUp or RealCompletionSealClassifierUp consumer route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier real seal consumer exactness`

Theorem 977.8 (Regular Cauchy modulus witness-ledger source-triad exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer of an accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$$

reads the L10 source triad exactly as the displayed $\mathsf{StreamName}^{\uparrow}$ *row* W *for finite StreamName windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *for dyadic tolerances, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *for readback before the Real seal row* S*. No hidden stream selector, separate dyadic ledger, quotient stream, selected limit, host real equality, countable choice, or ambient completeness row can replace any component of this triad.*

Proof. Project the accepted carrier to its displayed rows. The only source-triad coordinates available before the seal are W, D, Q, with W serving finite windows, D serving dyadic tolerances, and Q serving the RegSeqRat readback.

Apply Definition 977.2. Its source and witness exposure rows bind X, μ before the triad is read, and its row-ordering obligations place W, D, Q before the seal row S.

Use window-budget exhaustion and normalizer-tail commutation, Definition 977.5 and Definition 977.6. The first forces W, D before Q, while the second keeps N_0, T over the same witnessed modulus and finite-window rows.

Finally combine Real-seal consumer exactness with non-escape, Definition 977.7 and Definition 977.4. They make every Real-facing consumer read through the same finite packet and exclude hidden stream selectors, separate dyadic ledgers, quotient streams, selected limits, host real equality, countable choice, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier source triad exactness`

Theorem 977.9 (Regular Cauchy modulus witness-ledger source-triad lock)**.** *For any accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer that preserves the witnessed modulus route must consume* W, D, Q *as one locked source triad over the same witnessed modulus* μ*, normalized route* N_0*, and tail-budget row* T*. Replacing any one of* W, D, Q *by a row from another packet is not an accepted seal route.*

Proof. Project the accepted carrier to its displayed tuple. The only rows available for the source triad before the Real-facing seal are W, D, Q, and they occur in the same finite packet as μ, N_0, T.

Apply source-triad exactness from Definition 977.8. It identifies W as the finite StreamName window row, D as the dyadic tolerance row, and Q as the RegSeqRat readback row consumed before S.

Use normalizer-tail commutation from Definition 977.6. This keeps the normalized route row N_0 and tail-budget row T over the same witnessed modulus μ and over the same finite-window and dyadic rows.

Finally apply Real-seal consumer exactness and non-escape, Definition 977.7 and Definition 977.4. Any attempted replacement of one triad row by a row from another packet would require a hidden selector, separate ledger, quotient stream, selected limit, host equality, countable-choice row, or ambient completeness row, none of which is exported by the accepted seal route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier source triad lock`

Theorem 977.10 (Regular Cauchy modulus witness-ledger TasteGate readiness)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the displayed rows supply the complete paper-side obligation surface expected by the Nontrivial and FieldFaithful TasteGate requirements for this chapter. The rows provide carrier admission, non-escape, route-ordering, and local naming witnesses before any Lean encoding may expose the chapter as a TasteGate carrier.

Proof. Carrier admission is the tuple itself, as fixed by Definition 977.1. The local naming row is M, and the package, continuation, and transport rows P, C, H record the provenance, replay route, and componentwise preservation for the same tuple.

Apply the NameCert obligation theorem, Definition 977.2. It binds the source row X, witnessed modulus row μ, finite-window service W, normalizer row N_0, tail-budget row T, dyadic row D, RegSeqRat readback row Q, and Real seal row S as the displayed obligation rows.

Apply seal-route exactness and non-escape, Definition 977.3 and Definition 977.4. These results force every Real-facing consumer to read the same ordered rows and exclude quotient streams, host real equality, hidden modulus replacement, selected limits, countable choice, and ambient completeness.

Finally use source-triad exactness, Definition 977.8. It isolates W, D, Q as the finite-window, dyadic-tolerance, and RegSeqRat readback triad, so the TasteGate surface has a concrete nonempty packet and a field-faithful row inventory rather than a detached analytic assertion. □

Theorem 977.11 (Regular Cauchy modulus witness-ledger field separation)**.** *In an accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet, the witnessed ledger, finite-window service, normalizer row, tail-budget row, dyadic tolerance ledger, RegSeqRat readback, and Real seal cannot be collapsed into one another by a consumer route. Each row has a distinct downstream role, and deleting or identifying any one of them triggers the missing-coordinate refusal recorded by the carrier and non-escape boundary.*

Proof. Project the carrier to the displayed tuple. The row μ supplies the witnessed modulus ledger, W supplies finite StreamName windows, N_0 supplies the normalizer route, T supplies the tail budget, D supplies dyadic tolerances, Q supplies RegSeqRat readback, and S supplies the Real seal.

The NameCert obligations separate the roles. Source and witness exposure require X, μ before window service; finite-window service requires W, D before readback; tail-budget compatibility requires T on the same route; and seal ordering requires Q before S.

Use non-escape from Definition 977.4. A consumer that collapses any of these rows would need a replacement coordinate or a route that bypasses the displayed tuple, but the accepted packet exports only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

Thus each row is separated by its downstream use and by the refusal of missing coordinates. The witness ledger cannot serve as a window service, the window service cannot serve as a dyadic ledger, the normalizer and tail rows cannot serve as the Real seal, and the readback row cannot be omitted before seal consumption. □

Theorem 977.12 (Regular Cauchy modulus witness-ledger route field faithfulness)**.** *Any accepted packet route preserving the public Real-seal consumer surface preserves each displayed field of*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$$

in the same order. Consequently a later Lean FieldFaithful instance for RegularCauchyModulusWitnessLedger$^{\uparrow}$ *may project fields by tuple inspection rather than by quotient identification, host real equality, or any external completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier route field faithfulness`

Proof. Start with source-triad exactness, Definition 977.8. It forces every RealUp or completion-seal consumer to read W, D, Q as the finite-window, dyadic-tolerance, and RegSeqRat readback rows before the seal.

Use window-budget exhaustion and normalizer-tail commutation, Definition 977.5 and Definition 977.6. The first prevents hidden window or dyadic replacement before Q, while the second keeps N_0 and T over the same witnessed modulus μ and the same rows W, D before S.

Apply Real-seal consumer exactness, Definition 977.7. The consumer route reaches the public seal only through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, with H, C, P, M carrying transport, continuation, provenance, and local naming for that same tuple.

The preservation claim is therefore componentwise. Since no quotient stream, host real equality, selected limit, hidden modulus replacement, hidden window selector, separate dyadic ledger, countable choice, or ambient completeness coordinate is exported, field faithfulness is witnessed by inspecting the displayed tuple in order. □

Theorem 977.13 (Regular Cauchy modulus witness-ledger nontrivial witness surface)**.** *The* RegularCauchyModulus *carrier admits two distinguishable accepted packet surfaces. One may vary the displayed witnessed*

modulus ledger row μ while preserving the source row X and the downstream route shape, or vary the finite-window service row W while preserving the witnessed ledger shape. In either case the componentwise hsame *transport row H refuses to identify the two packets unless the changed displayed coordinate is restored.*

Proof. Project an accepted packet to the carrier tuple

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M).$$

The witnessed modulus ledger and finite-window service are separate coordinates of this tuple, not properties recovered from a quotient stream or from host-side equality of reals.

Apply Definition 977.2. Its source and witness exposure row requires the displayed μ to be read before any window is served, and its finite-window service row requires the displayed W to be the StreamName row consumed before readback.

The two packet surfaces are therefore separated by componentwise transport. A change in μ changes the witness-ledger coordinate read by the source route, while a change in W changes the finite-window coordinate read before D and Q. Since H stores componentwise hsame data for the displayed rows, it cannot identify either changed packet with the original accepted packet without reusing the original field. □

Theorem 977.14 (Regular Cauchy modulus witness-ledger field-faithful surface)**.** *For an accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet, equality of all displayed fields*

$$X, \mu, W, N_0, T, D, Q, S, H, C, P, M$$

determines the accepted packet surface. Conversely, any change to one of these displayed fields is visible to the local classifier through carrier projection, componentwise hsame *refusal, or one of the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}$ *obligation rows.*

Proof. The carrier is the finite tuple of Definition 977.1. If all displayed fields agree, tuple projection leaves no further coordinate on which two accepted packets can differ.

Now change one displayed field. If the changed row is X or μ, the source and witness exposure obligation detects the change. If it is W, D, or Q, the finite-window service and readback obligations detect the changed route before the Real-facing seal.

If the changed row is N_0 or T, the normalized route exactness and tail-budget compatibility obligations detect that the Real-facing route no longer uses the same witnessed modulus ledger. If the changed row is S, H, C, P, or M, then seal ordering, componentwise transport, displayed continuation replay, provenance, or local naming detects the changed accepted surface.

Thus the local classifier is field-faithful on the displayed tuple. The proof uses only carrier projection and the NameCert obligations, with no quotient stream, host real equality, hidden modulus replacement, selected limit, countable choice, or ambient completeness coordinate. □

Theorem 977.15 (Regular Cauchy modulus witness-ledger TasteGate field projection)**.** *Let*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$$

be an accepted $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet. The nontrivial witness pair for the packet may be chosen by changing either the witnessed modulus ledger row μ or the finite-window service row W. The field-faithful projection is exactly the ordered tuple projection through $X, \mu, W, N_0, T, D, Q, S, H,$ and every* $\mathsf{Real}^{\uparrow}$ *or* RealCompletionSealClassifierUp *consumer preserving the public seal route preserves those displayed fields componentwise.*

Proof. Use the nontrivial witness surface, Definition 977.13. It separates accepted packet surfaces by changing μ or W, and componentwise hsame refuses to identify either change without restoring the displayed coordinate.

Apply the field-faithful surface, Definition 977.14. It identifies the accepted packet surface with the ordered tuple projection through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

Now combine source-triad exactness and Real-seal consumer exactness, Definition 977.8 and Definition 977.7. They force every public Real-facing read to pass through the same witnessed ledger, finite-window, dyadic, readback, seal, transport, replay, provenance, and naming rows.

Finally use non-escape, Definition 977.4. Since the carrier exports no quotient stream, host equality, hidden modulus replacement, selected limit, countable choice, or ambient completeness coordinate, no consumer preserving the public seal route can identify the witness pair or collapse any displayed field. □

Theorem 977.16 (Regular Cauchy modulus witness-ledger carrier disjointness)**.** *No carrier automorphism preserving the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}$ *rows can collapse the witnessed-modulus coordinate* μ*, the normalized-route coordinate* N_0*, the tail-budget coordinate* T*, and the Real-seal coordinate* S *into one another. These four rows remain disjoint because they are read at different stages of the local route and are protected by the window-budget, normalizer-tail, and non-escape obligations.*

Proof. Begin with the NameCert obligations of Definition 977.2. They assign distinct local roles to the witnessed modulus ledger, normalized route, tail-budget row, and Real seal row: μ is exposed before any window service, N_0 records the normalized shared-modulus route, T records tail-budget compatibility for the same route, and S is consumed only after the readback and route rows are displayed.

Use window-budget exhaustion from Definition 977.5. It forces the finite-window and dyadic rows W, D before Q, so neither μ nor N_0 nor T can be replaced by a hidden window or dyadic ledger on the way to the seal.

Use normalizer-tail commutation from Definition 977.6. It keeps N_0 and T over the same witnessed modulus μ and the same finite-window rows before S, which prevents an automorphism preserving the local rows from exchanging the route coordinate with the tail-budget coordinate or with the seal.

Finally apply non-escape from Definition 977.4. The accepted packet exports only $X, \mu, W, N_0, T, D, Q, S, H, C,$ so a collapse among μ, N_0, T, S would require a replacement coordinate or an external equality principle. No such coordinate is present in the carrier. □

Theorem 977.17 (Regular Cauchy modulus witness-ledger consumer totality)**.** *For every accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer using the witnessed modulus route factors through the single displayed surface* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*. The consumer reads* μ *before* W*, reads* W, D *before* Q*, reads* N_0, T *over the same witness before* S*, and reads* H, C, P, M *as the structural transport, replay, provenance, and local naming rows. No quotient stream, host real equality, hidden modulus replacement, hidden window selector, separate dyadic ledger, selected limit, countable choice, or ambient completeness row can replace that route.*

Proof. Apply the NameCert obligation theorem, Definition 977.2. Its source and witness exposure rows put the displayed modulus ledger μ before the StreamName window row W, and its route-ordering clauses keep the later rows attached to the same accepted packet.

Next use window-budget exhaustion and source-triad exactness, Definition 977.5 and Definition 977.8. These force the finite-window and dyadic rows W, D before the RegSeqRat readback row Q, so a Real-facing consumer cannot insert a hidden window selector or a separate dyadic ledger.

Use normalizer-tail commutation from Definition 977.6. It keeps the normalized route row N_0 and the tail-budget row T over the same witnessed modulus and over the same finite-window route before the seal row S.

Apply Real-seal consumer exactness, Definition 977.7. It identifies the public RealUp and completion-seal read as the displayed route through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, with H, C, P, M supplying transport, replay, provenance, and local naming for that route.

Finally apply non-escape from Definition 977.4. Since the carrier exports no quotient stream, host real equality, hidden modulus replacement, hidden window selector, separate dyadic ledger, selected limit, countable choice, or ambient completeness coordinate, every admitted consumer reads the displayed modulus-window-dyadic-readback-seal route as one total surface. □

Theorem 977.18 (Regular Cauchy modulus witness-ledger tail-service uniqueness)**.** *For every accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the single witnessed modulus ledger row μ together with the displayed tail-budget row T determines the tail-service route used by every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer of L. Any tail-service read preserving the accepted packet must pass through the same finite window row W, normalized row N_0, dyadic row D, RegSeqRat readback row Q, and seal row S; a second tail threshold, hidden witness ledger, quotient stream, selected limit, host equality, countable choice, or ambient completeness row is not an admitted route.*

Proof. Begin with the NameCert obligations of Definition 977.2. The source and witness exposure rows display X, μ, and the tail-budget compatibility row attaches T to that same witnessed route before a Real-facing seal is consumed.

Apply normalizer-tail commutation, Definition 977.6. It keeps N_0 and T over the same witnessed modulus μ and over the same finite-window and dyadic rows.

Use window-budget exhaustion and source-triad exactness, Definition 977.5 and Definition 977.8. These identify W, D, Q as the only finite-window, dyadic, and readback portion of the tail-service route.

Finally apply Real-seal consumer exactness and non-escape, Definition 977.7 and Definition 977.4. Since the accepted packet exports only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, the pair μ, T determines the admitted tail-service path through the displayed rows and leaves no coordinate for a second service. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier tail service uniqueness`

Theorem 977.19 (Regular Cauchy modulus witness-ledger dyadic-window coherence)**.** *For every accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the StreamName row W, DyadicRatCore row D, and RegSeqRat row Q are consumed as one coherent source triad before the Real-facing seal row S is read. A consumer cannot keep one of W, D, Q while replacing another with a row from a different packet, a hidden dyadic ledger, a quotient stream, a selected limit, host equality, countable choice, or ambient completeness data.

Proof. Project the accepted packet to its displayed tuple. The only StreamName, DyadicRatCore, and RegSeqRat coordinates available before the seal are W, D, Q.

Apply the NameCert obligations and source-triad exactness, Definition 977.2 and Definition 977.8. They identify W as finite-window service, D as the dyadic tolerance ledger, and Q as the readback row consumed before S.

Use window-budget exhaustion and normalizer-tail commutation, Definition 977.5 and Definition 977.6. The first forces W, D before Q, while the second keeps the route tied to the same witnessed modulus μ and tail-budget row T.

Conclude with the source-triad lock, Real-seal consumer exactness, and non-escape, Definition 977.9, Definition 977.7, and Definition 977.4. These results make the triad a single displayed route and exclude every replacement coordinate listed above. □

Theorem 977.20 (Regular Cauchy modulus witness-ledger seal-readback factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *read of an accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$$

factors through the displayed rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*. The factorization reads the witnessed modulus row before the window service, reads the dyadic-window-RegSeqRat triad before the Real seal, and reads the transport, continuation, provenance, and local certificate rows as part of the same packet.*

Proof. Start from the seal route and consumer totality, Definition 977.3 and Definition 977.17. They identify every Real-facing read as a route through the displayed packet surface rather than through a selected limit or completion principle.

The tail-service uniqueness theorem fixes the tail side of the route. It makes μ and T determine the admitted service path through W, N_0, D, Q, S.

The dyadic-window coherence theorem fixes the readback side of the route. It forces W, D, Q to travel as one source triad before S.

Finally apply non-escape, Definition 977.4. Since the carrier exports no quotient stream, host real equality, hidden modulus replacement, hidden window selector, separate dyadic ledger, selected limit, countable choice, or ambient completeness coordinate, every RealUp or completion-seal read factors through exactly $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$. □

Theorem 977.21 (Regular Cauchy modulus witness-ledger open-phase exit sufficiency)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the displayed rows supply the Phase Real and Completion exit surface by routing $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *through one finite witness ledger. The rows* X, μ, N_0, T *bind the source, witnessed modulus, normalized route, and tail budget;* W, D, Q, S *are the finite window, dyadic tolerance, regular readback, and Real seal; and* H, C, P, M *supply transport, replay, provenance, and local naming. This exit surface exports no quotient stream, host real equality, selected limit, countable choice, or ambient completeness row.*

Proof. Begin with the carrier and NameCert obligations, Definition 977.1 and Definition 977.2. They display exactly the rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and assign the StreamName, dyadic, regular-readback, and Real-seal roles to the same accepted packet.

Use the seal route, source-triad exactness, and normalizer-tail commutation, Definition 977.3, Definition 977.8, and Definition 977.6. These keep the finite window, dyadic tolerance, readback, normalized route, and tail budget attached to the witnessed modulus before the seal is read.

Finally apply Real-seal consumer exactness and consumer totality, Definition 977.7 and Definition 977.17. Together with non-escape, Definition 977.4, they make the displayed packet sufficient for the Phase Real and Completion exit surface and exclude every forbidden completion coordinate listed in the statement. □

Theorem 977.22 (Regular Cauchy modulus witness-ledger readback seal determinacy)**.** *For every accepted packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the readback row Q and seal row S are determined by the same witnessed modulus row μ, normalizer row N_0, tail-budget row T, finite-window row W, and dyadic tolerance row D. A Real-facing consumer cannot replace Q or S by a readback or seal from another packet while preserving the accepted modulus-ledger route.

Proof. Project the accepted carrier. The only readback and Real-seal coordinates available to a consumer of L are the displayed rows Q and S.

Apply source-triad exactness and normalizer-tail commutation, Definition 977.8 and Definition 977.6. The first fixes Q as the readback attached to W and D, while the second keeps N_0 and T over the same witnessed modulus μ before the seal.

The seal route and Real-seal consumer exactness, Definition 977.3 and Definition 977.7, then identify S as the seal consumed after that same readback route. Non-escape rules out a second packet row, selected limit, host equality, quotient stream, countable choice, or ambient completeness datum as a replacement. □

Theorem 977.23 (Regular Cauchy modulus witness-ledger completion boundary)**.** *Every Completion-facing consumer of an accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet may read only the finite boundary*

$$X, \mu, W, N_0, T, D, Q, S, H, C, P, M.$$

It may consume the witnessed modulus ledger, finite window, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and local naming rows, but it cannot export quotient streams, host real equality, selected limits, countable choice, ambient completeness, or a completion object not displayed by this boundary.

Proof. Start from consumer totality, Definition 977.17. It factors every RealUp or completion-seal consumer through the displayed route $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

The open-phase exit sufficiency theorem identifies this route with the StreamName, DyadicRatCore, RegSeqRat, and RealUp exit packet. The readback seal determinacy theorem fixes Q and S to the same witnessed modulus, normalizer, tail-budget, window, and dyadic rows.

Finally use non-escape, Definition 977.4. Since the carrier has no coordinate for quotient streams, host real equality, selected limits, countable choice, ambient completeness, or undisplayed completion objects, no Completion-facing consumer can export such data from the accepted boundary. □

Theorem 977.24 (Regular Cauchy modulus witness-ledger obligation-closure route)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the displayed rows exhaust the carrier admission, source witness, normalizer, tail-budget, finite-window, dyadic, RegSeqRat readback, Real seal, transport, continuation, provenance, local NameCert*, Nontrivial, and FieldFaithful obligation surface for the chapter. No additional quotient-stream, host-real-equality, hidden-modulus, hidden-window, selected-limit, countable-choice, ambient-completeness, or undisplayed completion row is needed to state the obligation boundary.*

Proof. The carrier and local NameCert surface are the tuple and the seven obligation rows of Definition 977.1 and Definition 977.2. These give carrier admission, source witness exposure, normalizer exactness, finite-window service, tail-budget compatibility, readback and Real seal ordering, and the structural rows H, C, P, M.

The consumer-facing boundary is fixed by non-escape, source-triad exactness, and seal-route exactness: Definition 977.4, Definition 977.8, and Definition 977.3. Together they force every RealUp and completion-seal read through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and refuse every hidden stream, dyadic, quotient, selected-limit, host-equality, choice, or ambient-completeness coordinate.

TasteGate readiness supplies the chapter-level Nontrivial and FieldFaithful side of the same surface. The nontrivial witness surface separates packet witnesses by the displayed μ or W coordinate, while the field-faithful surface and route field-faithfulness identify accepted packet equality with componentwise preservation of the displayed tuple.

Finally apply completion boundary. It states that a Completion-facing consumer may read only the finite boundary $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$. Hence the listed rows already cover the carrier, witness, route, seal, transport, continuation, provenance, naming, Nontrivial, and FieldFaithful obligations, so the chapter records obligation closure without asserting Lean verification or a bridge. □

Theorem 977.25 (Regular Cauchy modulus witness-ledger tail-service exactness)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$ *or* RealCompletionSealClassifierUp *tail-service read of L factors through the same witnessed modulus row μ, finite* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, normalizer row N_0, tail-budget row T, seal row S, and structural rows H, C, P, M. No hidden selector, quotient stream, host real equality, selected limit, ambient completeness coordinate, or replacement tail-service packet may serve as a Real-facing route for L.*

Proof. Project the carrier and the local NameCert obligations. The accepted packet displays exactly the rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, and the naming surface assigns the StreamName window, dyadic tolerance, readback, tail budget, normalizer, seal, transport, continuation, provenance, and local-name roles to those rows.

Use window-budget exhaustion, normalizer-tail commutation, and source-triad exactness. These keep W, D, Q, N_0, T attached to the same witnessed modulus row μ before any Real-facing service reads the packet.

Apply readback seal determinacy and completion boundary. The readback row Q and seal row S are fixed by the same displayed route, and completion-facing consumers may read only the boundary $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

Finally invoke carrier non-escape. Since L has no coordinate for a hidden selector, quotient stream, host real equality, selected limit, ambient completeness object, or second tail-service packet, every Real-facing tail-service read factors through the displayed route and no other one. □

Theorem 977.26 (Regular Cauchy modulus witness-ledger normalizer-source totality)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

any $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer that reads the witnessed modulus row* μ *must preserve two displayed route constraints. First, the source triad* W, D, Q *is attached to* μ *and cannot be replaced by a hidden source, dyadic, or readback row. Second, the normalizer row* N_0 *is displayed on the same packet before tail-budget and Real-seal consumption. Thus the admitted consumer surface is a compatible finite route over* μ*, not a single linear order placing* N_0 *before* Q*, and not a hidden modulus selector, quotient stream, host real equality, selected limit, countable choice, or ambient completeness datum.*

Proof. The carrier and local NameCert obligations expose the finite packet rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$. By Definition 977.2, the normalizer row N_0, finite source window W, dyadic tolerance D, readback row Q, tail-budget row T, and seal row S are named rows of the same accepted packet.

Apply the source-triad lock, Definition 977.9. It binds the displayed source triad W, D, Q to the witnessed modulus row μ, so a consumer that exposes μ cannot detach the readback from the finite source data or replace one of the triad rows.

Now use tail-service exactness, Definition 977.25. Any Real-facing or completion-seal tail-service read factors through the shared $\mu, W, N_0, T, D, Q, S, H, C, P, M$ route, hence the same packet displays N_0 before tail-budget and seal consumption while preserving the source triad over μ.

Finally apply the completion boundary, Definition 977.23. It permits no off-packet selector, quotient stream, host equality, selected limit, countable choice, ambient completeness row, or hidden completion selector, so the two displayed constraints are total for every accepted consumer of the witnessed modulus. □

Theorem 977.27 (Regular Cauchy modulus witness-ledger tail-budget readback)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the tail-budget row T *is read back only through the same source triad* W, D, Q *and the same normalizer row* N_0 *attached to the witnessed modulus* μ*. A tail-budget readback may witness how far the packet must go after* W *and* D*, but it cannot replace* μ*, replace* N_0*, or introduce a second readback, quotient stream, host real equality, selected limit, countable choice, or ambient completeness row.*

Proof. Begin with the local obligation surface, Definition 977.2. The packet has one displayed witnessed modulus row μ, one displayed normalizer row N_0, one tail-budget row T, and the source triad W, D, Q.

The source-triad lock, Definition 977.9, fixes Q as the readback carried by W and D over the same modulus witness. Thus the tail-budget row can only be interpreted against that displayed triad.

Use tail-service exactness, Definition 977.25. It routes every tail-service read through $\mu, W, N_0, T, D, Q, S, H, C,$ so T records the budget along the already displayed normalizer and readback path rather than producing another one.

The completion boundary, Definition 977.23, removes the remaining escape cases. Since the boundary has no coordinate for a replacement modulus, replacement normalizer, second readback, quotient stream, host equality, selected limit, choice principle, or ambient-completeness object, tail-budget readback is exhausted by W, D, Q and N_0. □

Theorem 977.28 (Regular Cauchy modulus witness-ledger readback route lock)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the readback row Q is locked to the same witnessed route μ, W, D, N_0, T before the seal row S is consumed. A consumer may read Q only as the $\mathsf{RegSeqRat}^{\uparrow}$ *readback generated by the displayed source window W, dyadic tolerance row D, normalizer row N_0, and tail-budget row T over μ; it cannot replace any one of those rows by a hidden source, quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, or second readback surface.*

Proof. Begin with the carrier and local NameCert obligations. They expose exactly one witnessed modulus row μ, one finite source window W, one dyadic tolerance row D, one normalizer row N_0, one tail-budget row T, one readback row Q, and one seal row S.

Apply source-triad lock, Definition 977.9. It binds W, D, Q to the same witnessed modulus, so the readback route cannot detach Q from the displayed finite-window and dyadic rows.

Next use tail-service exactness and tail-budget readback, Definition 977.25 and Definition 977.27. These identify the service route for Q with $\mu, W, N_0, T, D, Q, S, H, C, P, M$ and force T to be read through the same W, D, Q, N_0 path.

Finally apply the completion boundary. Since completion-facing consumption has no coordinate for a hidden source, quotient stream, host equality, selected limit, choice row, ambient completeness row, or second readback surface, Q is locked to the displayed route before S can be consumed. □

Theorem 977.29 (Regular Cauchy modulus witness-ledger completion-seal handoff totality)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every RealCompletionSealClassifierUp *handoff factors through the full displayed boundary $X, \mu, W, N_0, T, D, Q, S, H,$ No proper sub-boundary omitting the witnessed source, source triad, normalizer, tail budget, seal, transport, continuation, provenance, or local NameCert row is export-ready for completion-seal consumption.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier completion seal handoff totality`

Proof. Project the handoff to the local NameCert surface. The only rows available to the completion-seal consumer are the carrier row X, witnessed modulus μ, source window W, normalizer N_0, tail budget T, dyadic tolerance D, readback Q, seal S, transports H, continuation C, provenance P, and local naming row M.

Use source-triad lock and readback route lock. They keep W, D, Q attached to μ and force Q to be read through the same N_0, T route before S is consumed.

Apply tail-service exactness and completion boundary. The former routes every completion-seal service through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, while the latter excludes hidden selectors, quotient streams, host equalities, selected limits, countable choice, ambient completeness, and detached completion rows.

Thus totality is exactly the full displayed boundary. If any listed row is missing, the consumer has a proper sub-boundary rather than the accepted handoff surface, so it is not export-ready. □

Theorem 977.30 (Regular Cauchy modulus witness-ledger tail-window refusal)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

any route that reaches toward Real$^{\uparrow}$ *or* RealCompletionSealClassifierUp *while omitting either the displayed tail-budget row T or the finite-window row W is refused before Real-seal consumption. Such a route may mention μ, D, Q, N_0 or S, but without both W and T it is not the accepted tail-service route and cannot consume the Real seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier tail window refusal`

Proof. The NameCert obligations display both W and T as named rows of the packet. The finite-window row supplies the source window for the dyadic and readback route, and the tail-budget row records the admitted service depth along that route.

Apply source-triad lock to the finite-window side. Without W, the triad W, D, Q is not present, so the proposed route cannot identify the accepted readback over μ.

Apply tail-service exactness and tail-budget readback to the budget side. Without T, the route cannot be the displayed $\mu, W, N_0, T, D, Q, S, H, C, P, M$ service route and cannot read the tail budget against the accepted normalizer and readback path.

Finally use readback seal determinacy and non-escape. Since S is determined by the displayed readback route and the packet has no off-boundary replacement for W or T, every route missing either row is refused before Real-seal consumption. □

Theorem 977.31 (Regular Cauchy modulus witness-ledger Real-seal export readiness)**.** *For an accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

a Real$^{\uparrow}$ *or* RealCompletionSealClassifierUp *consumer is export-ready exactly when the packet displays the witnessed modulus μ, the source triad W, D, Q, the normalizer row N_0, the tail-budget row T, the seal row S, the componentwise transport row H, the continuation row C, the provenance row P, and the local* NameCert *row M. No consumer is export-ready from a proper sub-boundary or from an undisplayed quotient stream, host real equality, selected limit, countable choice, ambient completeness object, or detached seal row.*

Proof. For sufficiency, start from the carrier and NameCert obligations in Definition 977.2. They display exactly the source triad, normalizer row, tail-budget row, seal row, transport, continuation, provenance, and local naming rows of L.

The source-triad lock, Definition 977.9, binds W, D, Q to μ. Tail-service exactness, Definition 977.25, then routes every Real-facing or completion-seal read through the same $\mu, W, N_0, T, D, Q, S, H, C, P, M$ packet. Therefore a consumer with all listed rows displayed has the complete export surface.

For necessity, suppose a Real$^{\uparrow}$ or RealCompletionSealClassifierUp consumer is export-ready. By tail-service exactness, its route factors through the witnessed modulus, source triad, normalizer, tail-budget, seal, transport, continuation, provenance, and local NameCert rows of the accepted packet.

It remains to rule out readiness from less data. Apply completion boundary, Definition 977.23. The consumer may read only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, so omitting one of the displayed rows or substituting an undisplayed quotient, equality, limit, choice, completeness, or detached seal row cannot produce an export-ready consumer. □

Theorem 977.32 (RegularCauchyModulusWitnessLedger bridge request surface)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every bridge request for L must factor through the witnessed modulus row μ, the finite StreamName$^{\uparrow}$ *window row W, the normalizer row N_0, the tail-budget row T, the* DyadicRatCore$^{\uparrow}$ *tolerance row D, the* RegSeqRat$^{\uparrow}$ *readback row Q, the* Real$^{\uparrow}$ *seal row S, the componentwise* hsame *transports H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* NameCert *row M. Such a request cannot use quotient streams, host real equality, countable choice, selected limits, ambient completeness, or a detached standard bridge row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier bridge request surface`

Proof. Start with the carrier and local NameCert obligations. They expose exactly the packet rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, so a bridge-facing request has no carrier coordinate outside the witnessed modulus ledger, finite window, normalizer, tail budget, dyadic tolerance, readback, seal, transport, replay, provenance, and local naming rows.

Next apply window-budget exhaustion, normalizer-tail commutation, and tail-service exactness. These keep the finite window, normalized route, tail budget, dyadic tolerance, and RegSeqRat readback attached to the same witnessed modulus row before any Real-facing or Completion-facing service can consume the packet.

Then use Real-seal consumer exactness and readback-seal determinacy. The Real seal is the one determined by the same displayed readback route, and any consumer of the seal must read that route rather than replacing it by a host equality, selected limit, quotient stream, choice principle, or ambient-completeness row.

Finally combine the completion boundary with the public Phase Real and Completion exit sufficiency theorem. The boundary lists only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, while the exit theorem identifies these rows as the public consumer surface. Therefore every bridge request factors through the displayed packet, and any detached standard bridge row is off-route for this chapter. □

Theorem 977.33 (Regular Cauchy modulus witness-ledger root seal minimality)**.** *For every* BHist*-anchored accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the root seal handoff to any Phase Real or Completion consumer is minimal exactly at the displayed boundary. The consumer must read the RegSeqRat$^{\uparrow}$ *source row X, witnessed modulus row μ, finite* StreamName$^{\uparrow}$ *window row W,* RegularCauchyModulusNormalizer$^{\uparrow}$ *route row N_0,* RegularCauchyTailBudget$^{\uparrow}$ *row T,* DyadicRatCore$^{\uparrow}$ *tolerance row D,* RegSeqRat$^{\uparrow}$ *readback row Q,* Real$^{\uparrow}$ *seal row S, transport row H, continuation row C, provenance row P, and local* NameCert *row M. No proper sub-boundary, quotient stream, host real equality, selected limit, countable choice, ambient completeness datum, hidden window selection, detached normalizer, or detached* RealCompletionSealClassifierUp *seal can serve as the root seal handoff.*

Proof. Project the packet to its local carrier and NameCert obligations. These display the twelve rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and place the history anchor on the same BHist route as the provenance and local naming rows.

Use source-triad lock to bind W, D, Q to the witnessed modulus row μ. Then use normalizer-tail commutation and tail-service exactness to keep N_0 and T on that same route before Q and S are consumed.

Apply Real-seal export readiness and the bridge request surface. They identify the Phase Real and Completion-facing handoff with the displayed rows and give no coordinate for quotient streams, host real equality, selected limits, countable choice, ambient completeness, hidden windows, detached normalizers, or detached classifier seals.

Thus deleting any listed row leaves a proper sub-boundary rather than a root seal handoff. The displayed tuple is therefore the minimal accepted boundary for root consumers. □

Theorem 977.34 (Regular Cauchy modulus witness-ledger source-row coverage)**.** *For every* BHist-*anchored accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the rows read by a Phase Real or Completion consumer cover exactly the sibling sources needed by the root handoff: StreamName$^{\uparrow}$ *is read through* W*,* DyadicRatCore$^{\uparrow}$ *through* D*,* RegSeqRat$^{\uparrow}$ *through* X *and* Q*,* Real$^{\uparrow}$ *through* S*,* RegularCauchyModulusNormalizer$^{\uparrow}$ *through* N_0*,* RegularCauchyTailBudget$^{\uparrow}$ *through* T*, and* RealCompletionSealClassifierUp *through the seal-facing route* Q, S, H, C, P, M*. The handoff has no additional source row and no missing row among* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*.*

Proof. Begin with the carrier row and the NameCert obligation theorem. They name $X, \mu, W, N_0, T, D, Q, S, H, C, P$ as the only rows of the packet, so any source-row coverage statement must be obtained by projecting these rows rather than by importing a second carrier.

Read the sibling anchors row by row. The finite-window service is W for StreamName$^{\uparrow}$, the dyadic tolerance ledger is D for DyadicRatCore$^{\uparrow}$, the regular source and readback are X, Q for RegSeqRat$^{\uparrow}$, and the seal is S for Real$^{\uparrow}$.

Next read the structural siblings. The normalized route is N_0 for RegularCauchyModulusNormalizer$^{\uparrow}$, the admitted tail service is T for RegularCauchyTailBudget$^{\uparrow}$, and the classifier-facing seal route is Q, S, H, C, P, M for RealCompletionSealClassifierUp.

Finally invoke non-escape and completion boundary. Since the boundary exports no quotient stream, host real equality, selected limit, countable choice, ambient completeness datum, hidden window selection, detached normalizer, or detached seal, the displayed rows are both sufficient for all named sibling reads and exhaustive for the root handoff. □

Theorem 977.35 (Regular Cauchy modulus witness-ledger consumer obligation exhaustion)**.** *Let* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$ *be a* BHist-*anchored accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet. Every root consumer obligation for a Phase Real or Completion handoff is discharged by the displayed carrier, NameCert obligations, source-triad lock, completion boundary, Real-seal export readiness, and bridge request surface. After those clauses are read, there is no remaining root obligation asking for quotient streams, host real equality, selected limits, countable choice, ambient completeness, hidden window selection, detached normalizers, detached seals, or any row outside* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*.*

Proof. First discharge the object surface. The carrier and NameCert obligations expose the source, witnessed modulus, window, normalizer, tail budget, dyadic tolerance, readback, seal, transport, continuation, provenance, and local naming rows.

Next discharge the route surface. Source-triad lock attaches W, D, Q to μ, normalizer-tail commutation attaches N_0, T to the same route, and tail-service exactness sends every Real-facing or Completion-facing service through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

Then discharge the consumer surface. The completion boundary identifies the accepted consumer boundary, Real-seal export readiness makes that boundary export-ready, and the bridge request surface says every bridge-facing request factors through the same displayed rows.

It remains only to exclude extra obligations. Non-escape and the boundary theorems give no row for quotient streams, host real equality, selected limits, countable choice, ambient completeness, hidden windows, detached normalizers, detached seals, or any off-boundary source. Therefore the root consumer obligations are exhausted by the displayed packet. □

Theorem 977.36 (Regular Cauchy modulus witness-ledger UniformCauchyCriterion exit factor)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{UniformCauchyCriterion}^{\uparrow}$*-facing consumer of* L *factors through the displayed route* $X, \mu, W, N_0, T, D, Q, S, H, C$ *The consumer may read the packet through the witnessed seal route, source triad, tail service, and* $\mathsf{Real}^{\uparrow}$*-seal export surface, but it cannot replace that route by a hidden source, quotient stream, selected limit, host real equality, countable choice, ambient completeness object, or detached completion row.*

Proof. Begin with the accepted carrier and local NameCert obligations. They expose the carrier X, witnessed modulus μ, finite source window W, normalizer N_0, tail budget T, dyadic tolerance D, readback Q, seal S, transport H, continuation C, provenance P, and local naming row M.

Apply the seal-route theorem, source-triad lock, and tail-service exactness. These identify the route to the seal with the same μ, W, N_0, T, D, Q data and prevent a UniformCauchyCriterion-facing consumer from detaching the Cauchy witness from the displayed finite-window and dyadic-tail rows.

Finally use Real-seal export readiness. The export theorem says that a Real-facing or completion-facing consumer is ready exactly from the displayed route, so the UniformCauchyCriterion exit surface factors through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and has no coordinate for off-route equality, choice, quotient, limit, or ambient-completeness data. □

Theorem 977.37 (Regular Cauchy modulus witness-ledger dyadic-tail cofinal readback)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

dyadic tail cofinal readback is exhausted by the displayed W, D, Q, T *rows before the seal row* S *is consumed. Thus every cofinal dyadic-tail readback available to a completion-facing consumer is the one generated from the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, dyadic tolerance row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and tail-budget row* T *over the same witnessed modulus route.*

Proof. Use window-budget exhaustion to bind the finite source window to the admitted tail budget. This makes the available tail service finite and ledger-visible before any seal consumption can occur.

Next apply tail-budget readback. It reads the budget row T through the same W, D, Q path, so a cofinal dyadic-tail consumer cannot introduce another readback surface after the displayed rows have been exhausted.

The seal row S is then reached only through readback-seal determinacy. Since the packet has no second source window, dyadic tolerance, readback, or tail-budget coordinate, the cofinal readback seen before S is exactly the displayed W, D, Q, T route. □

Theorem 977.38 (Regular Cauchy modulus witness-ledger seal-classifier scope)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

any RealCompletionSealClassifierUp *scope read is exactly the displayed seal route* Q, S, H, C, P, M*. The scope read may use the* RegSeqRat$^{\uparrow}$ *readback row* Q*,* Real$^{\uparrow}$ *seal* S*, componentwise* hsame *transports* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* M*, and it cannot use quotient streams, selected limits, host real equality, countable choice, or ambient completeness.*

Proof. Project the classifier read to the completion boundary. The boundary exposes only the readback, seal, transport, continuation, provenance, and local naming rows once the source and tail route have delivered the packet to the RealCompletionSealClassifier surface.

Apply readback-seal determinacy and Real-seal consumer exactness. These force S to be the seal determined by Q, and force the classifier-facing consumer to carry the displayed transports, replay, provenance, and NameCert row rather than an off-route representative.

The non-escape theorem rules out the remaining alternatives. Quotient streams, selected limits, host real equality, countable choice, and ambient completeness have no row in Q, S, H, C, P, M, so every classifier scope read is exactly the displayed seal route. □

Theorem 977.39 (Regular Cauchy modulus witness-ledger StreamName-RegSeqRat-Real exactness)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the L10 handoff exposed to a Real$^{\uparrow}$ *or* RealCompletionSealClassifierUp *consumer is exactly the route from the finite* StreamName$^{\uparrow}$ *window row* W*, through the* DyadicRatCore$^{\uparrow}$ *tolerance row* D *and the* RegSeqRat$^{\uparrow}$ *readback row* Q*, to the* Real$^{\uparrow}$ *seal row* S*. The route is carried by the witnessed source* X, μ*, the normalizer row* N_0*, the tail-budget row* T*, and the structural rows* H, C, P, M*, and it contains no quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, or undisplayed completion datum.*

Proof. Project L to the carrier of Definition 977.1. The displayed finite route contains one StreamName window row W, one dyadic tolerance row D, one RegSeqRat readback row Q, and one Real seal row S.

Apply the NameCert obligations and source-triad exactness, Definition 977.2 and Definition 977.8. These rows bind X, μ before window service and force the consumer to read W, D, Q as the finite-window, dyadic-tolerance, and readback triad before S.

Use window-budget exhaustion and readback-seal determinacy, Definition 977.5 and Definition 977.22. The first theorem rules out a second window or dyadic ledger before Q, and the second fixes Q and S to the same witnessed route over μ, N_0, T, W, D.

Finally apply Real-seal consumer exactness and non-escape, Definition 977.7 and Definition 977.4. Since the accepted packet exports only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, the L10 handoff has no quotient stream, host real equality, selected limit, choice row, ambient completeness row, or undisplayed completion datum. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier streamname regseqrat real exactness`

Theorem 977.40 (Regular Cauchy modulus witness-ledger modulus-normalizer-tail coherence)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the witnessed modulus row μ, the normalizer row N_0, and the tail-budget row T are locked to the same finite-window, dyadic-tolerance, and readback triad W, D, Q before the Real seal row S is exported. Thus a consumer cannot read a normalized route from one modulus ledger, a tail budget from another route, or a readback from a detached finite window while preserving the accepted seal export.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier modulus normalizer tail coherence`

Proof. Begin with the carrier and local NameCert obligations. The packet displays exactly one witnessed modulus row μ, one normalizer row N_0, one tail-budget row T, and the source triad W, D, Q before the seal row S.

Apply normalizer-tail commutation and tail-service exactness, Definition 977.6 and Definition 977.25. They keep N_0 and T on the same witnessed modulus route and make every tail-service read pass through $\mu, W, N_0, T, D, Q, S, H, C, P, M$.

Use the source-triad lock and tail-budget readback theorems, Definition 977.9 and Definition 977.27. The triad W, D, Q is attached to μ, and T can be interpreted only against that same displayed normalizer and readback path.

The Real-seal export readiness theorem then supplies the boundary for S. Since readiness requires $\mu, W, D, Q, N_0, T, S, H, C, P, M$ together, a consumer that changes one of μ, N_0, T, W, D, Q loses the accepted export route rather than obtaining a second route. □

Theorem 977.41 (Regular Cauchy modulus witness-ledger completion consumer boundary)**.** *For every accepted* RegularCauchyModulusWitnessLedger$^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every Completion-facing or RealCompletionSealClassifierUp*-facing consumer factors through the union of the StreamName-RegSeqRat-Real exact route and the modulus-normalizer-tail coherent route. The consumer may cite only the displayed rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and cannot cite an undisplayed completion object, quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, detached normalizer, detached tail budget, or detached seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier completion consumer boundary`

Proof. Start from the completion boundary and bridge request surface, Definition 977.23 and Definition 977.32. They force every Completion-facing read and every bridge request through the finite packet rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

The StreamName-RegSeqRat-Real exactness theorem above fixes the readback-to-seal side of that packet. It identifies W, D, Q, S as the exact L10 path from finite windows and dyadic tolerances to RegSeqRat readback and Real seal export.

The modulus-normalizer-tail coherence theorem above fixes the modulus-service side. It locks μ, N_0, T to the same triad W, D, Q before the seal is exported.

Finally apply non-escape and open-phase exit sufficiency, Definition 977.4 and Definition 977.21. Since the public Phase Real and Completion exit surface is exactly the displayed packet, a consumer outside the union of the two routes would need an undisplayed completion object or another forbidden coordinate, none of which is exported. □

Theorem 977.42 (Regular Cauchy modulus witness-ledger root source admission)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the root source rows admitted toward Phase Real and Completion are exactly the witnessed modulus row μ*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* S*, carried over the displayed source* X *and structural rows* H, C, P, M*. In particular, source admission cannot pass through a hidden modulus selector, quotient stream, host real equality, selected limit, countable choice, ambient completeness object, or an undisplayed completion datum.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier root source admission`

Proof. Project the carrier and NameCert obligations, Definition 977.1 and Definition 977.2. They display the packet rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and assign the witnessed modulus, StreamName window, dyadic tolerance, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming roles.

Apply the source-triad lock, Definition 977.9. It binds W, D, Q to the same witnessed modulus row μ, so any root admission that exposes one of the L10 source rows must expose the whole finite triad over the same source witness.

Use the completion boundary and bridge request surface, Definition 977.23 and Definition 977.32. These restrict Phase Real, Completion, and bridge-facing reads to the displayed packet, hence no hidden selector, quotient stream, host equality, chosen limit, choice principle, ambient completeness object, or undisplayed completion datum can serve as a root source route. □

Theorem 977.43 (Regular Cauchy modulus witness-ledger root consumer handoff totality)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer factors through the same finite source triad* W, D, Q*, the normalizer row* N_0*, the tail-budget row* T*, the transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* M*. The handoff is total on the displayed root boundary: there is no consumer route through an extra completion carrier, detached normalizer, detached tail budget, detached seal, quotient stream, host real equality, countable choice, selected limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier root consumer handoff totality`

Proof. Start from root source admission above. It identifies the admitted source rows as μ, W, D, Q, S over the displayed source and structural packet rows.

Apply the NameCert obligations and source-triad lock, Definition 977.2 and Definition 977.9. Together they keep W, D, Q attached to μ and place N_0, T, H, C, P, M on the same accepted packet before any Real-facing or completion-seal consumer reads it.

Next use completion boundary and bridge request surface, Definition 977.23 and Definition 977.32. They make the displayed tuple the whole consumer boundary for Phase Real and Completion exit, so every consumer route factors through $W, D, Q, N_0, T, H, C, P, M$ with the same seal row S.

The remaining alternatives are off-boundary. Since the carrier has no row for an extra completion carrier, detached normalizer, detached budget, detached seal, quotient stream, host equality, chosen limit, choice principle, or ambient-completeness object, none can witness a consumer handoff for L. □

Theorem 977.44 (Regular Cauchy modulus witness-ledger root formalization readiness)**.** *The carrier of* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$*, its local* $\mathsf{NameCert}$ *obligations, the nontrivial witness surface, the field-faithful surface, the source-triad lock, the completion boundary, and the bridge-request surface present a* BHist*-anchored formalization target for the root exit route. The target is the finite packet relation over* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*; it is not a quotient-stream target, not a host-real-equality target, and not an ambient-completeness target.*

Proof. The carrier and NameCert obligations, Definition 977.1 and Definition 977.2, give the finite packet shape and the local naming rows. Thus the object to be formalized is already a displayed BHist packet with named source, witness, service, seal, transport, replay, provenance, and local certificate coordinates.

The nontrivial witness surface and field-faithful surface keep the packet meaningful at the root boundary. They separate displayed witness data and identify accepted equality with preservation of the packet rows rather than with host equality of real numbers.

Apply the source-triad lock, completion boundary, and bridge request surface, Definition 977.9, Definition 977.23, and Definition 977.32. These theorems force every Real-facing, Completion-facing, and bridge-facing route through $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$.

Consequently Lean can aim at a finite BHist-anchored packet theorem whose constructors and projections are the displayed rows. No quotient stream, host-real-equality premise, selected-limit object, choice principle, or ambient-completeness carrier is part of the root formalization surface. □

Theorem 977.45 (Regular Cauchy modulus witness-ledger Real object readback totality)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$ *readback exposed at the root exit route is read only through the displayed coordinates* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*. No accepted packet exports an additional selected limit, quotient stream, host real equality, countable-choice row, ambient-completeness row, detached seal, detached normalizer, or detached tail-budget coordinate for that readback.*

Proof. The carrier and NameCert obligations display exactly the rows $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and assign their local names. Apply Definition 977.1 and Definition 977.2.

The source-triad lock, completion boundary, bridge request surface, and root formalization readiness theorems then force every Real-facing exit route through that same finite packet. Thus a readback that uses an undisplayed Real object would leave the accepted BHist packet, contradicting the root boundary already fixed by Definition 977.9, Definition 977.23, Definition 977.32, and Definition 977.44. □

Theorem 977.46 (Regular Cauchy modulus witness-ledger StreamName dyadic readback scope)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *readback scope is exactly the displayed triad* W, D, Q *over the same source and witnessed modulus rows* X, μ*. It cannot be refined by a hidden selector, quotient stream, host equality of real objects, selected limit, countable-choice row, or ambient completeness row.*

Proof. Use the source-triad lock to bind W, D, Q to the same witnessed modulus row μ over the source X. The tail-budget readback and Real-seal export readiness theorems keep that triad on the displayed normalizer, tail-service, and seal route rather than on a separate stream object.

The completion boundary and root formalization readiness theorems identify the accepted target as the finite packet itself. Hence any selector, quotient stream, host equality, chosen limit, or ambient-completeness coordinate would be an extra root datum, not a refinement of the W, D, Q readback scope. □

Theorem 977.47 (Regular Cauchy modulus witness-ledger completion seal root consumer exhaustion)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, every* $\mathsf{RealCompletionSealClassifierUp}$ *consumer at the root boundary factors through the union of the Real object readback route, the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the normalizer-tail route already fixed by* N_0, T, H, C, P, M*. There is no fourth route through an undisplayed completion carrier, quotient stream, host real equality, selected limit, countable choice, or ambient completeness coordinate.*

Proof. The Real object readback totality theorem above supplies the $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ route for $\mathsf{Real}^{\uparrow}$ reads. The StreamName dyadic readback scope theorem above supplies the W, D, Q route for finite-window and dyadic readback.

For the normalizer-tail route, apply Definition 977.6, Definition 977.25, and Definition 977.27. These keep N_0, T, H, C, P, M attached to the same displayed triad and seal.

Finally use the bridge request surface and root formalization readiness theorems. Since the root packet exports only these named coordinates, every completion-seal consumer is exhausted by their union. □

Theorem 977.48 (Regular Cauchy modulus witness-ledger source-triad terminality)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, the displayed source triad* W, D, Q *is terminal among Real-facing and Completion-facing source routes under the witnessed modulus row* μ*. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer that reaches the seal row* S *factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *before it can read* N_0, T, H, C, P, M *or consume* S*.*

Proof. First apply the source-triad lock and StreamName-RegSeqRat-Real exactness, Definition 977.9 and Definition 977.39. These identify W, D, Q as the finite source route over μ and prevent a consumer from inserting a quotient stream, host equality, chosen limit, or ambient-completeness carrier before the Real seal.

Next use modulus-normalizer-tail coherence. It makes N_0 and T service rows over the same source triad rather than independent routes, so a Real-facing or Completion-facing consumer cannot bypass W, D, Q by reading the normalizer or tail budget alone.

Finally apply completion consumer boundary and completion seal root consumer exhaustion. They exhaust the consumer surface by the Real object readback route, the W, D, Q readback

route, and the normalizer-tail route attached to it. Therefore the source triad is terminal for every displayed root consumer. □

Theorem 977.49 (Regular Cauchy modulus witness-ledger seal source pullback)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, the seal row* S *is the pullback of the source triad* W, D, Q *along the displayed normalizer, tail, transport, continuation, provenance, and local naming rows* N_0, T, H, C, P, M*. Thus the Real seal consumed at the root boundary is determined by* W, D, Q *over* μ *together with those structural rows, and it is not a detached Real object or an external completion carrier.*

Proof. Begin with readback-seal determinacy and StreamName-RegSeqRat-Real exactness. The determinacy theorem fixes the readback row Q and the seal row S to the same route over μ, N_0, T, W, D, while exactness identifies that route as the L10 handoff from StreamName window to DyadicRatCore tolerance, RegSeqRat readback, and Real seal.

Use modulus-normalizer-tail coherence to attach N_0 and T to the same W, D, Q triad before S is exported. The rows H, C, P, M then serve only as the displayed transport, replay, provenance, and local NameCert support for that route.

The completion consumer boundary theorem supplies the consumer-side equation for the pullback description. Since every completion-facing consumer reads only $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$, the seal row is the source triad transported along N_0, T, H, C, P, M, not a separate completion witness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier seal source pullback`

Theorem 977.50 (Regular Cauchy modulus witness-ledger completion refusal boundary)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, any attempted Real-facing or Completion-facing consumer route that omits one of* μ, W, D, Q, S *is a refusal boundary for the packet. Such a route is not an alternate completion witness, even if it retains* N_0, T, H, C, P, M *or cites a quotient stream, host real equality, selected limit, countable-choice row, ambient completeness row, detached normalizer, detached tail budget, or detached seal.*

Proof. For omission of μ, apply the source-triad lock and readback-seal determinacy. Without the witnessed modulus row, the triad W, D, Q and the seal row S are no longer attached to the accepted source route.

For omission of one of W, D, Q, use source-triad terminality together with StreamName-RegSeqRat-Real exactness. The consumer loses the terminal source route from finite StreamName window through DyadicRatCore tolerance and RegSeqRat readback, so the remaining rows cannot form a Real or completion witness.

For omission of S, apply seal source pullback and completion seal root consumer exhaustion. The route may still display service or provenance rows, but it has no Real seal to consume at the root boundary.

The completion consumer boundary and non-escape theorems rule out all replacements. Quotient streams, host real equality, selected limits, countable choice, ambient completeness, detached normalizers, detached tail budgets, and detached seals are not exported rows of the accepted packet, so the attempted route is refused rather than reinterpreted as another completion witness. □

Theorem 977.51 (Regular Cauchy modulus witness-ledger root-exit partition)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, every Phase Real and Completion root-exit consumer route is classified by exactly one of the following four cases:*

1. *the Real object readback route through the displayed packet* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*;*
2. *the* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *source-triad route through* W, D, Q *over* X, μ*;*
3. *the normalizer-tail structural route through* N_0, T, H, C, P, M *attached to the same triad and seal;*
4. *the refusal boundary for an attempted route omitting at least one of* μ, W, D, Q, S*.*

Thus the root-exit surface has no additional route through an undisplayed Real object, quotient stream, host real equality, selected limit, countable-choice row, ambient-completeness row, detached normalizer, detached tail budget, or detached completion seal.

Proof. Begin with Real object readback totality and StreamName dyadic readback scope. These identify the first two admitted consumer routes: a Real-facing read may use only the displayed packet, while the finite-window source read is exactly the W, D, Q triad over X, μ.

Next apply completion seal root consumer exhaustion. It states that every completion-seal consumer at the root boundary factors through the union of the Real object readback route, the source-triad route, and the normalizer-tail route. The normalizer-tail route is then pinned down by source-triad terminality and seal source pullback, because N_0, T, H, C, P, M are structural rows attached to the same W, D, Q source and the same seal row S rather than an independent completion carrier.

It remains to classify attempted routes outside those admitted branches. The completion refusal boundary theorem applies to any route omitting one of μ, W, D, Q, S, so such a route is refused instead of becoming a hidden witness.

The four cases are disjoint at the root boundary because each reads a different BEDC role: displayed Real readback, source-triad readback, structural normalizer-tail service, or refusal for a missing mandatory row. Combining the exhaustion theorem with the refusal boundary leaves no fifth Phase Real or Completion consumer route. □

Theorem 977.52 (Regular Cauchy modulus witness-ledger root-exit partition stability)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet* $L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M)$*, the root-exit classification of Definition 977.51 is stable under every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer route that preserves the displayed rows* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*. If such a consumer route is classified as displayed Real readback,* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$ *source-triad readback, normalizer-tail structural service, or refusal for omitting one of* μ, W, D, Q, S*, then every row-preserving presentation of the same consumer route remains in that same case. No row-preserving presentation can move the route to another case or create an additional root-exit case.*

```
Lean4 ✓  BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.RegularCauchyModulusWitnessLedgerCarrier
root exit partition stability
```

Proof. Start from the root-exit partition. It already exhausts every Phase Real and Completion root-exit consumer route by the displayed Real readback route, the source-triad readback route, the normalizer-tail structural route, or the refusal boundary for a missing mandatory row.

For the displayed Real and source-triad cases, apply source-triad terminality together with seal source pullback. A row-preserving presentation keeps the same μ, W, D, Q, S coordinates, so it cannot detach the Real seal from the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The first two cases therefore remain the same cases.

For the normalizer-tail case, use seal source pullback and the completion seal root consumer exhaustion theorem. The rows N_0, T, H, C, P, M are structural service rows attached to the same

triad and seal, not a second completion carrier. Preserving the displayed packet keeps that attachment fixed, so the structural route cannot become either a source-triad readback route or an external completion route.

For the refusal case, apply the completion refusal boundary. A presentation that still omits one of μ, W, D, Q, S remains refused, while a presentation preserving all displayed rows is no longer a presentation of that refused route but of one of the three admitted routes already classified above.

Finally apply non-escape. Since the packet exports no quotient stream, host real equality, selected limit, countable-choice row, ambient-completeness row, detached normalizer, detached tail budget, or detached completion seal, a row-preserving consumer has no hidden coordinate by which to change its root-exit case. The four-way classification is therefore stable for all displayed-row-preserving $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealCompletionSealClassifierUp}$ consumers. □

Theorem 977.53 (Regular Cauchy modulus witness-ledger formal target boundary)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the Lean-facing root target is the finite relation saying that every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompletionSealClassifierUp}$ *consumer route is one of the four root-exit cases of Definition 977.51, and that the route preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* S*, normalizer* N_0*, tail budget* T*, transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and* $\mathsf{NameCert}$ *row* M*. No Lean-facing consumer route may replace this finite relation by quotient streams, host real equality, selected limits, countable choice, ambient completeness, detached normalizers, detached tail budgets, or detached completion seals.*

Proof. Begin with the root-exit partition and partition stability. They classify every root consumer route as displayed Real readback, source-triad readback, normalizer-tail structural service, or refusal for a missing mandatory row, and they keep that classification stable under displayed-row-preserving presentations.

Use seal source pullback and source-triad terminality next. These theorems attach the $\mathsf{Real}^{\uparrow}$ seal S and the normalizer-tail rows N_0, T, H, C, P, M to the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback row.

Finally apply the completion refusal boundary and nonescape. A route omitting one of the source or seal rows is refused, and the packet exports no quotient stream, host real equality, selected limit, countable-choice row, ambient-completeness row, detached normalizer, detached tail budget, or detached completion seal. The finite relation described in the statement is therefore the exact Lean-facing root target for $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealCompletionSealClassifierUp}$ consumers. □

Theorem 977.54 (Regular Cauchy modulus witness-ledger L10 exit boundary)**.** *For every accepted* $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ *packet*

$$L = (X, \mu, W, N_0, T, D, Q, S, H, C, P, M),$$

the packet is an L10 exit-boundary packet exactly when its displayed rows factor through the shared sibling surfaces: the $\mathsf{StreamName}^{\uparrow}$ *finite-window surface* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *seal-readback rows* X, Q*, the* $\mathsf{Real}^{\uparrow}$ *finite-window ledger boundary* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite ledger* D*, and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *exit factor over* $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$*. Conversely, these five displayed sibling surfaces reconstruct the same witness-ledger exit boundary using only* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*, dyadic-ledger, and finite-window rows, and add no quotient stream, host limit, selected limit, host real equality, countable-choice row, ambient completeness datum, or detached completion carrier.*

Proof. For the forward direction, start with the open Phase Real and Completion exit surface of Definition 977.21. It exposes the accepted packet as the finite route from the witnessed modulus row through the StreamName, DyadicRatCore, RegSeqRat, and RealUp rows.

Next apply source-row coverage, consumer-obligation exhaustion, and StreamName-RegSeqRat-Real exactness. These identify W as the finite-window surface, D as the dyadic finite ledger, X, Q as the regular-source and readback surface, and S as the RealUp boundary row consumed by the same packet.

Finally apply the UniformCauchyCriterion exit factor, dyadic-tail cofinal readback, and seal-classifier scope. They keep the exit consumer on $X, \mu, W, N_0, T, D, Q, S, H, C, P, M$ and rule out any completion-facing coordinate outside the displayed BHist, hsame, Cont, Pkg, NameCert, dyadic-ledger, and finite-window rows.

For the reverse direction, suppose the five sibling surfaces are displayed over the same rows. StreamName-RegSeqRat-Real exactness and source-triad terminality, Definition 977.39 and Definition 977.48, assemble W, D, Q, S into the L10 handoff.

Modulus-normalizer-tail coherence and root consumer handoff totality, Definition 977.40 and Definition 977.43, then attach $X, \mu, N_0, T, H, C, P, M$ to that handoff. The resulting packet is the same witness-ledger exit boundary because the non-escape theorem leaves no row for quotient streams, host limits, selected limits, host real equality, countable choice, ambient completeness, or detached completion data. □

Falsifiable prediction: If a RealUp seal consumer reads a regular Cauchy modulus without the displayed witness-ledger row, normalized route row, finite-window row, dyadic tolerance row, and RegSeqRat readback row, the packet is refuted.

Independence witness: regular-cauchy-modulus-normalizer, cauchymodulusmeet, regular-cauchy-tail-budget, real-completion-seal-classifier, streamname: no sibling is bijective because this packet uniquely binds the witnessed modulus ledger to the downstream finite-window and RealUp seal route; each sibling omits either the single witness ledger, the normalized route, or the RealUp consumer seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$ packet is generated from a regular source, a displayed modulus witness ledger, finite StreamName windows, normalized-tail and tail-budget rows, dyadic tolerance readback, a RealUp seal, componentwise transports, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 977.1, Definition 977.2, Definition 977.3, Definition 977.4, Definition 977.6, Definition 977.7, Definition 977.10, Definition 977.12, Definition 977.13, Definition 977.14, Definition 977.24, Definition 977.25, Definition 977.26, Definition 977.27, Definition 977.31, Definition 977.21, Definition 977.32, Definition 977.33, Definition 977.34, Definition 977.35, Definition 977.36, Definition 977.37, and Definition 977.38 bind the finite carrier, NameCert obligations, seal route, non-escape surface, source-window order, normalized-tail compatibility, seal consumer totality, TasteGate readiness, route field-faithfulness, nontrivial witness surface, field-faithful surface, chapter scoped surface, tail-service exactness, normalizer-source totality, tail-budget readback, Real-seal export readiness, public Phase Real and Completion exit surface, bridge request surface, root seal minimality, source-row coverage, consumer obligation exhaustion, UniformCauchyCriterion exit factor, dyadic-tail cofinal readback, and RealCompletionSealClassifier scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyModulusWitnessLedgerUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host real equality, hidden modulus replacement, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking for $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$.

Upgrade path: No further paper-side closure upgrade is pending for the finite witnessed modulus-ledger packet; bridge checking remains a separate verification-axis matter.

Chapter 978

A Concrete Naming Certificate for $\mathsf{HaltingTraceClassifier}^{\uparrow}$

The $\mathsf{HaltingTraceClassifier}^{\uparrow}$ packet realizes the positive finite-trace side of the halting obstruction in chapter 9368 without presenting a total halting decision. It sits beside the certificate-history obstruction of chapter 491, the displayed trace packet of chapter 492, the machine-step surface of chapter 280, and the inscription-event naming surface of chapter 519.

Definition 978.1 (Halting trace-classifier carrier)**.** *A* $\mathsf{HaltingTraceClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, h, F, R, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the machine or certificate row whose one-step behavior is read through the* $\mathsf{TuringMachine}^{\uparrow}$ *and* $\mathsf{NameCert}$ *surfaces;*
2. *h is the initial history row;*
3. *F is a finite positive* Cont *trace from h to a displayed halted row;*
4. *R is the positive classifier row accepting exactly the displayed trace F as terminating evidence;*
5. *D is the diagonal refusal row stating that no accepted classifier read may decide divergence or non-termination outside the displayed finite trace;*
6. *H records componentwise* hsame *transports for the machine row, initial history, trace row, positive classifier row, refusal row, package provenance, and local name;*
7. *C records the displayed* Cont *routes that replay the trace, transport the positive read, and return the diagonal refusal to the same packet;*
8. *P is the* Pkg *provenance over* $\mathsf{TuringMachine}^{\uparrow}$, $\mathsf{HaltingDistinction}^{\uparrow}$, $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingTraceClassifier}^{\uparrow}}$ *row.*

The carrier contains no row for a total halting predicate, no negative divergence decider, no unbounded search certificate, and no host-level proof that an absent continuation never appears.

Theorem 978.2 (Halting trace-classifier positive trace soundness)**.** *Every accepted positive read of a* $\mathsf{HaltingTraceClassifier}^{\uparrow}$ *packet factors through the displayed finite trace row F. A consumer may*

classify M at h as positively halted only by replaying the finite Cont *route from h to the displayed halted row through C, together with the transported trace and classifier rows F, R, H.*

Proof. Project the accepted carrier to $T = (M, h, F, R, D, H, C, P, N)$. The positive classifier row is R, and the only evidence row it may read is the finite trace F.

The continuation routes in C replay that trace from the initial history row to the displayed halted row. Componentwise transport through H moves M, h, F, R together, so the positive read remains attached to the same displayed trace after transport.

The package and local naming rows P, N expose no second trace source. Hence every accepted positive halting classification factors through the finite Cont trace F rather than through an ambient decision procedure. □

Theorem 978.3 (Halting trace-classifier diagonal refusal)**.** *The diagonal row D of an accepted* $\mathsf{HaltingTraceClassifier}^{\uparrow}$ *packet refuses any request to turn the positive trace classifier into a total halting classifier. In particular, an absent displayed trace cannot be read as divergence, non-termination, or a negative halting verdict.*

Proof. Read the refusal coordinate D from the carrier. It is paired with the positive classifier row R to state exactly where the packet stops: R accepts displayed finite traces, while D rejects requests that ask for a verdict outside those traces.

The route data in C can replay a finite Cont trace already present as F, but it has no route for an unbounded search over all continuations. The transport row H preserves this boundary because it transports D with the positive classifier and trace rows.

Since no coordinate stores a divergence decider, total halting predicate, or proof that all future continuations are absent, the diagonal request has no source row in the packet. It is therefore refused by the displayed carrier itself. □

Theorem 978.4 (Halting trace-classifier non-escape)**.** *No consumer of an accepted* $\mathsf{HaltingTraceClassifier}^{\uparrow}$ *packet can extract a decision about divergence outside the displayed trace row F. The packet exports positive finite-trace evidence and diagonal refusal, but it does not export a total classifier over all machine-history pairs.*

Proof. Inspect the carrier coordinates. The positive evidence is F together with R, and the refusal evidence is D together with its replay and transport rows.

Apply positive trace soundness to any accepted positive read. The read must use the displayed finite trace. Apply diagonal refusal to any request for a negative or total verdict. Such a request asks for a coordinate absent from T.

The provenance and local naming rows P, N only account for these displayed rows. They do not add unbounded search, divergence, or totality data, so no consumer can escape from the finite positive-trace and refusal surface. □

Theorem 978.5 (Halting trace-classifier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingTraceClassifier}^{\uparrow}}$ *surface for Definition 978.1 consists of five finite obligations:*

1. *carrier exposure for the displayed packet $T = (M, h, F, R, D, H, C, P, N)$;*
2. *positive trace soundness, requiring every accepted positive read to factor through the displayed finite* Cont *trace F;*
3. *diagonal refusal, requiring every total-halting or divergence request outside F to be rejected by D;*

4. *transport stability for the machine row, history row, trace row, classifier row, refusal row, provenance, and local naming row through H;*
5. *ledger exactness, requiring all replay, package, and naming reads to factor through C, P, N with no hidden divergence decider.*

The later Lean `TasteGate.lean` *development must provide explicit* Nontrivial HaltingTraceClassifier$^{\uparrow}$ *and* FieldFaithful HaltingTraceClassifier$^{\uparrow}$ *instances for this finite carrier.*

Lean4 ✓ `BEDC.Derived.HaltingTraceClassifierUp.HaltingTraceClassifierTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.HaltingTraceClassifierUp.HaltingTraceClassifierUp taste gate boundary`

Proof. Carrier exposure is projection from Definition 978.1. The tuple lists the machine row, initial history, finite trace, positive classifier, diagonal refusal, transports, continuation routes, provenance, and local name.

Positive trace soundness is Definition 978.2, and diagonal refusal is Definition 978.3. Together they separate accepted finite halting evidence from forbidden total classification.

Transport stability follows from the componentwise role of H. Replay and ledger exactness follow from C, P, N, which are the only rows carrying continuation replay, package provenance, and local naming.

Nontriviality rules out identifying a positive trace row with the diagonal refusal row. Field faithfulness requires the TasteGate implementation to preserve $M, h, F, R, D, H, C, P, N$, so a future gate cannot discharge the packet by forgetting either the finite trace evidence or the refusal boundary. □

Falsifiable prediction: Every accepted HaltingTraceClassifierUp positive read factors through a displayed finite Cont trace and cannot decide divergence outside that trace.

Independence witness: haltingdistinction, haltingdistinctiontrace, turingmachine, inscriptionevent, continuationtracenormalform: none is bijective because none simultaneously carries positive finite-trace classification and diagonal refusal rows.

Closure status for NameCert$_{\text{HaltingTraceClassifier}^{\uparrow}}$.

Constructive story: A HaltingTraceClassifier$^{\uparrow}$ packet is generated from one machine row, one initial history row, one displayed finite Cont trace, one positive classifier row, one diagonal refusal row, componentwise transports, replay routes, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\text{HaltingTraceClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 978.1, Definition 978.2, Definition 978.3, Definition 978.4, and Definition 978.5 bind the finite positive-trace classifier and diagonal refusal surface.

Formal verification: FormalStatus(NameCert$_{\text{HaltingTraceClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a total halting predicate, divergence decision, unbounded search certificate, Lean proof, or public standard bridge.

Upgrade path: Checked closure requires Lean encoding of the carrier and TasteGate proofs for nontriviality, field faithfulness, positive trace soundness, diagonal refusal, and non-escape.

Chapter 979

A Concrete Naming Certificate for $\mathsf{RealWindowSynchronizer}^{\uparrow}$

The $\mathsf{RealWindowSynchronizer}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as one finite naming certificate. It consumes the StreamName window surface of chapter 126, the RegSeqRat tail surface of chapter 301, the Cauchy limit seal of chapter 386, the regular Cauchy tail budget of chapter 906, and the Real seal surface of chapter 38. The synchronizer is not a new completion principle; it is the finite packet that keeps the window, tail, budget, limit-seal, and RealUp rows aligned before any completion-facing consumer reads the route.

Definition 979.1 (Real window synchronizer carrier)**.** *A* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (W, R, T, L, S, H, C, P, N)$$

with the following displayed coordinates:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window service row whose finite windows are visible to downstream consumers;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row that reads the same window service as a regular rational sequence surface;*
3. *T is the shared tail threshold and* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *service row through which the window read and the regular tail read are synchronized;*
4. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row consuming the synchronized tail route as a sealed Cauchy-limit handoff;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after W, R, T, L have been replayed through the displayed routes;*
6. *H records componentwise* hsame *transports for the window, tail, threshold, limit-seal, Real seal, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes that inspect the StreamName window, pull the RegSeqRat tail to the shared threshold, pass through the CauchyLimitSeal row, and expose the RealUp seal;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$ *row naming exactly this synchronizer packet.*

The carrier has no coordinate for a quotient stream, selected host real, ambient completeness theorem, independent seal route, or second window service not transported through H.

Lean4 ✓ `BEDC.Derived.RealWindowSynchronizerUp.taste gate`
Lean4 (definition) `BEDC.Derived.RealWindowSynchronizerUp.RealWindowSynchronizerUp`

Theorem 979.2 (Real window synchronizer NameCert obligations)**.** *For every accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier* $Z = (W, R, T, L, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window admission through* W*, RegSeqRat tail admission through* R*, shared-threshold admission through* T*, CauchyLimitSeal route admission through* L*, RealUp seal admission through* S*, componentwise* hsame *transport through* H*, displayed route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 979.1. Each coordinate is displayed in the finite BHist packet, so the local naming row can inspect only $W, R, T, L, S, H, C, P, N$.

The obligation rows are then read in route order. The StreamName window and RegSeqRat tail rows are tied by the shared threshold T; the CauchyLimitSeal row consumes that synchronized tail route; and the RealUp seal row is exposed only after the same replay path has passed through H, C, P, N.

No extra obligation can appear, because the carrier has no quotient, selected host real, independent seal, or second window-service coordinate. Thus the displayed rows exhaust the local NameCert surface. □

Theorem 979.3 (Real window synchronizer budget pullback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read exported by an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet factors through the same shared threshold row* T *used by the* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row* R*. Transporting the window, tail, or threshold rows through* H *transports the CauchyLimitSeal and RealUp seal reads through the same displayed* Cont *routes* C *and does not create a second budget ledger.*

Proof. Project the packet $Z = (W, R, T, L, S, H, C, P, N)$. The only threshold coordinate is T, and both the window route and the tail route are displayed as consumers of that row.

Compose the displayed routes in C: first read W, then replay R at the shared threshold, then pass through L before exposing S. Componentwise transport by H changes representatives of these rows but preserves the same ordered route.

Since no separate budget coordinate is present, a completion-facing read cannot bypass the pullback. It receives the RealUp seal only as the end of the synchronized StreamName, RegSeqRat, CauchyLimitSeal, and local NameCert route. □

Lean4 ✓ `BEDC.Derived.RealWindowSynchronizerUp.RealWindowSynchronizerCarrier budget pullback`

Theorem 979.4 (Real window synchronizer non-escape)**.** *An accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier exports no route to a Real or completion consumer except the finite synchronized packet* $(W, R, T, L, S, H, C, P, N)$*. In particular, it cannot export a StreamName window without the RegSeqRat tail threshold, a RegSeqRat tail without the shared StreamName window, a CauchyLimitSeal read detached from the RealUp seal, or a RealUp seal detached from the displayed limit-seal route.*

Proof. Inspect the carrier coordinates. The exported window, tail, threshold, limit seal, Real seal, transports, continuations, provenance, and naming rows are all fields of the same finite packet.

If a consumer removes W, the tail row no longer has the displayed StreamName window it claims to read. If it removes R or T, the Real-facing route has no regular tail threshold. If it removes L, the completion-facing handoff is absent; if it removes S, no RealUp seal can be read.

Thus every admitted route is the displayed synchronized route, and every escape route asks for a coordinate not present in the carrier. □

Lean4 ✓ `BEDC.Derived.RealWindowSynchronizerUp.RealWindowSynchronizerCarrier nonescape`

Theorem 979.5 (Real window synchronizer limit-seal compatibility)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier*

$$Z = (W, R, T, L, S, H, C, P, N),$$

the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L is consumed only after the same* $\mathsf{StreamName}^{\uparrow}$ *window row W, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tail row R, and the same shared threshold row T have been replayed through the displayed transport, route, provenance, and naming rows H, C, P, N. No consumer of L may replace W, R, or T by a second window service, hidden tail oracle, detached threshold ledger, or ambient completeness coordinate.*

Proof. Project the accepted packet to its displayed coordinates. The only entries available before the limit-seal row are the window W, tail R, threshold T, and the shared transport and route surface H, C.

The replay through C first reads the StreamName window and then the RegSeqRat tail at the same threshold. Componentwise hsame transport in H may rename those rows, but it preserves the ordered route into L rather than creating a second source for the seal.

The provenance and naming rows P, N then certify that the consumed limit seal is the row displayed in the same finite packet. Since the carrier has no hidden tail oracle, detached threshold, second window service, or ambient completeness coordinate, every admitted L read is the synchronized W, R, T, H, C, P, N read. □

Theorem 979.6 (Real window synchronizer Real seal exposure)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier $Z = (W, R, T, L, S, H, C, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal row S is exposed only after the window row W, tail row R, threshold row T, and limit-seal row L have been read through the displayed route. The exposure exports no quotient stream, selected host real, ambient completeness theorem, detached seal route, or second window service.*

Proof. Use the budget pullback of Definition 979.3. It places every Real-facing read over the shared threshold T used by the StreamName window and RegSeqRat tail.

Then apply limit-seal compatibility from Definition 979.5. The CauchyLimitSeal row is available only after the same W, R, T replay has passed through H, C, P, N.

Finally expose the Real seal row S as the last displayed consumer row. The non-escape theorem Definition 979.4 rules out quotient streams, selected host reals, ambient completeness, detached seal routes, and second window services because none is present among the carrier coordinates. □

Theorem 979.7 (Real window synchronizer obligation-closure package)**.** *The carrier definition of Definition 979.1, the local* $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$ *obligations of Definition 979.2, the budget pullback of Definition 979.3, the non-escape boundary of Definition 979.4, the limit-seal compatibility of Definition 979.5, the Real-seal exposure of Definition 979.6, and the checked TasteGate carrier alignment marker* `BEDC.Derived.RealWindowSynchronizerUp.TasteGate.RealWindowSynchronizerTaste` *jointly form the* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *obligation-closure surface for the finite Real and Completion window-to-seal route.*

Proof. The carrier definition fixes the finite packet $Z = (W, R, T, L, S, H, C, P, N)$ and excludes quotient streams, selected host reals, ambient completeness, detached seal routes, and second window services.

The NameCert obligation theorem names the rows that must be admitted. The budget pullback ties the StreamName window and RegSeqRat tail to the shared threshold, while non-escape prevents any consumer from reading outside the displayed packet.

The two seal theorems then place the CauchyLimitSeal row and the RealUp seal row on the same replay path. Together with the TasteGate marker for single-carrier alignment, these rows state the exact obligation package without claiming a checked Lean proof for the new theorem sites. □

Theorem 979.8 (Real window synchronizer StreamName–RegSeqRat scope)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier* $Z = (W, R, T, L, S, H, C, P, N)$*, every* $\mathsf{StreamName}^{\uparrow}$ *consumer read and every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read used by the synchronizer factors through the same displayed* W, R, T, H, C, P, N *rows. The synchronizer exposes no second window service, hidden tail oracle, or detached threshold ledger outside that finite route.*

Proof. Project the carrier Z to its displayed rows. The only StreamName window coordinate is W, the only RegSeqRat tail coordinate is R, and the only shared threshold coordinate available to either read is T.

Apply the budget pullback of Definition 979.3. It forces every Real or completion-facing read exported by the packet through the same threshold row used by both the window and tail routes.

Apply the non-escape boundary of Definition 979.4. Since no second window service, hidden tail oracle, or detached threshold ledger occurs among the coordinates, the displayed StreamName and RegSeqRat sibling rows named in the chapter introduction are the only available sources for the synchronizer read. □

Theorem 979.9 (Real window synchronizer limit-seal scope)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier* $Z = (W, R, T, L, S, H, C, P, N)$*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L *and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *are scoped by the same synchronized* W, R, T, H, C, P, N *route. Consequently, no completion-facing consumer can read* L *or* S *without the shared window-tail threshold packet.*

Proof. Start with Definition 979.8. It places the StreamName and RegSeqRat reads over the same window, tail, threshold, transport, route, provenance, and local naming rows.

Combine that scope with limit-seal compatibility from Definition 979.5. The CauchyLimitSeal row L is consumed only after the synchronized W, R, T replay has passed through H, C, P, N.

Finally apply the Real seal exposure theorem Definition 979.6. It exposes S only after L has been read on the same route, so a completion-facing consumer cannot reach either seal without the shared window-tail threshold packet. □

Theorem 979.10 (Real window synchronizer standard bridge route)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *packet*

$$Z = (W, R, T, L, S, H, C, P, N),$$

every standard bridge read of Z *factors through exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail* R*, the shared threshold and* $\mathsf{RegularCauchyTailBudget}^{\uparrow}$ *row* T*, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L*, the* $\mathsf{Real}^{\uparrow}$ *seal* S*, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$ *row* N*. It exports no quotient stream, selected host real, ambient completeness theorem, detached seal route, or second window service.*

Proof. Apply the StreamName–RegSeqRat scope theorem Definition 979.8. It forces the first bridge coordinates to be the displayed W, R, T, H, C, P, N rows rather than a second window service, hidden tail oracle, or detached threshold ledger.

Next apply the limit-seal scope theorem Definition 979.9. The CauchyLimitSeal row L and the RealUp seal row S then sit on the same synchronized route, so a standard bridge consumer cannot read either seal without the shared window-tail threshold packet.

The Real-seal exposure theorem Definition 979.6 identifies the terminal public seal row as S after the displayed route through W, R, T, L, H, C, P, N.

Finally use the non-escape boundary Definition 979.4. Since quotient streams, selected host reals, ambient completeness theorems, detached seal routes, and second window services are absent from the carrier, the displayed route exhausts every standard bridge read. □

Theorem 979.11 (Real window synchronizer standard bridge surface)**.** *For an accepted* $\mathsf{RealWindowSynchronizer}^{\uparrow}$ *carrier* $Z = (W, R, T, L, S, H, C, P, N)$*, the carrier row, local* $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$ *obligations, shared budget pullback, non-escape row, limit-seal compatibility, Real-seal exposure, StreamName–RegSeqRat scope, limit-seal scope, and the checked TasteGate single-carrier marker* `BEDC.Derived.RealWindowSyn` *together form the bridge surface from the finite synchronizer packet* $(W, R, T, L, S, H, C, P, N)$ *to the standard reflection schema.*

Lean4 ✓ `BEDC.Derived.RealWindowSynchronizerUp.RealWindowSynchronizerUp StdBridge`

Proof. The carrier definition Definition 979.1 fixes the finite packet and excludes every coordinate outside $W, R, T, L, S, H, C, P, N$. The NameCert obligations of Definition 979.2 then identify the local admission rows carried by that same packet.

The public route is obtained by composing the budget pullback of Definition 979.3 with the non-escape boundary of Definition 979.4. These two results make the shared threshold row T the only budget source and prevent any completion-facing read from leaving the displayed synchronizer coordinates.

The seal route is supplied by Definition 979.5 and Definition 979.6. The StreamName–RegSeqRat scope of Definition 979.8 and the limit-seal scope of Definition 979.9 place those reads on the same W, R, T, H, C, P, N route.

Together with the checked TasteGate single-carrier marker, these rows supply the paper-side bridge surface to the standard reflection schema. The verification axis remains grounded in the existing TasteGate target until a checked StandardBridge theorem is present. □

Falsifiable prediction: Removing the shared window row permits a RealUp seal read whose StreamName and RegSeqRat windows no longer agree; removing the seal row prevents any completion-facing consumer from reading the synchronizer as a RealUp route.

Independence witness: Nearest siblings: streamname, regseqrat, cauchylimitseal, regular-cauchy-tail-budget, and real. The synchronizer is not bijective to any one sibling because it carries all five surfaces simultaneously: a StreamName finite window, a RegSeqRat tail threshold, a CauchyLimitSeal row, a RegularCauchyTailBudget service row, and a RealUp seal row; dropping any one row changes the admitted consumer route.

The Lean file `Derived/RealWindowSynchronizerUp/TasteGate.lean` will include Nontrivial RealWindowSynch with an explicit witness_pair and FieldFaithful $\mathsf{RealWindowSynchronizer}^{\uparrow}$ by constructor-field injection for the displayed coordinates $W, R, T, L, S, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RealWindowSynchronizer}^{\uparrow}}$.

Constructive story: A RealWindowSynchronizer$^{\uparrow}$ packet is generated from a StreamName window service row, RegSeqRat tail row, shared threshold row, CauchyLimitSeal row, RealUp seal row, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealWindowSynchronizer}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 979.1, Definition 979.2, Definition 979.3, Definition 979.4, Definition 979.5, Definition 979.6, Definition 979.7, Definition 979.8, Definition 979.9, Definition 979.10, and Definition 979.11 bind the finite synchronizer carrier, local obligation surface, shared-budget pullback, seal compatibility, public Real-seal exposure row, non-escape boundary, StreamName–RegSeqRat route, scoped seal route, standard bridge read route, standard reflection bridge surface, checked TasteGate alignment marker, and TasteGate instance `BEDC.Derived.RealWindowSynchronizerUp.taste_gate` for the Real and Completion window-to-seal route.

Formal verification: FormalStatus(NameCert$_{\text{RealWindowSynchronizer}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RealWindowSynchronizerUp.TasteGate.RealWindowSynchronizerTasteGate\_single\_carrier\_al`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, ambient metric completion, or checked Lean certificates for Definition 979.8 and Definition 979.9.

Upgrade path: No further upgrade is pending on the formal bridge axis; the checked StandardBridge theorem records the top bridge grade for this route.

Chapter 980

A Concrete Naming Certificate for $\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}$

The $\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}$ packet realizes the Real completeness route of chapter 9238 as a common classifier-normal-form surface for finite rational completion data. It sits over the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, and the RealUp seal row of chapter 38. The packet is a shared route certificate: it does not identify any one source surface with another, and it exports no quotient stream, selected limit representative, host real equality, countable choice row, or ambient completeness theorem.

Definition 980.1 (Regular Cauchy classifier normal-form carrier)**.** *A* $\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by every finite read in the packet;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at the same tolerance;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and tolerance;*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, W, Q have been displayed;*
5. *E is the completion-seal classifier row from chapter 900, recording that the RealUp read uses the same finite route;*
6. *H stores componentwise* hsame *transport for D, W, Q, R, E, C, P, N and for the local classifier-normal-form packet;*
7. *C stores the displayed* Cont *routes from D to W, from W to Q, from Q to R, from R to E, and from those rows to the local certificate;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCompletionSealClassifierUp}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}}$ *row.*

The carrier records the classifier-normal-form route across the four source surfaces. It has no coordinate for a quotient stream, hidden modulus replacement, selected external limit, host real equality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.RegularCauchyClassifierNormalFormUp.RegularCauchyClassifierNormalFormUp`

Theorem 980.2 (Regular Cauchy classifier normal-form obligations)**.** *For the carrier of Definition 980.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier admission for* $K = (D, W, Q, R, E, H, C, P, N)$*;*
2. *common dyadic tolerance exactness requiring the same* D *row to serve the StreamName window, RegSeqRat readback, and RealUp seal route;*
3. *finite-window exactness requiring* W *to be displayed before any regular rational readback is consumed;*
4. *RegSeqRat-to-RealUp handoff requiring* Q *to be displayed before* R *and requiring the completion-seal classifier row* E *to read the same route;*
5. *classifier stability under the componentwise* hsame *transports in* H *and the displayed* Cont *routes in* C*;*
6. *provenance, naming, and non-escape through* P, N*, excluding quotient streams, hidden modulus replacement, selected limits, host real equality, countable choice, and ambient completeness.*

The later Lean `TasteGate.lean` *contract for this chapter must provide explicit* `Nontrivial RegularCauchyClassi` *and* `FieldFaithful RegularCauchyClassifierNormalFormUp` *instances for this carrier surface.*

Proof. Carrier admission is projection from the displayed tuple. The dyadic tolerance, finite stream window, regular readback, RealUp seal, completion-seal classifier, transport, continuation, provenance, and local naming rows are all carrier coordinates.

The common dyadic tolerance row is read first. Since D is the only tolerance coordinate, the StreamName window, RegSeqRat readback, and RealUp seal route cannot switch to a second tolerance ledger without leaving the packet.

The finite-window row W is then displayed before the regular handoff Q. The route row C records this order, and the componentwise transport row H preserves the same fields under classifier comparison.

The RealUp seal row R and completion-seal classifier row E are read after D, W, Q. Provenance and local naming authenticate exactly that finite sequence.

No remaining carrier coordinate can supply quotient streams, hidden modulus replacement, selected external limits, host real equality, countable choice, or ambient completeness. Hence the six rows exhaust the local NameCert obligations. □

Theorem 980.3 (Regular Cauchy classifier normal-form Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}$ *packet reads the regular Cauchy classifier route only through*

$$D, W, Q, R, E, H, C, P, N.$$

In particular, the common dyadic tolerance row is displayed before the StreamName schedule, the StreamName schedule is displayed before the RegSeqRat readback, and the RegSeqRat readback is displayed before the RealUp seal and completion-seal classifier handoff.

Proof. Project the accepted carrier to $K = (D, W, Q, R, E, H, C, P, N)$. The only dyadic, stream, regular, RealUp, and completion-seal classifier coordinates are D, W, Q, R, E.

Apply the obligation surface of Definition 980.2. Its dyadic exactness, finite-window exactness, and handoff rows place D before W, W before Q, and Q before R, E.

The displayed Cont routes in C replay the same order for consumers, while the componentwise hsame transports in H preserve the named rows. The provenance row P and local name N add authentication without adding analytic data.

Thus every RealUp or completion-facing read factors through the listed finite route. A bypass would require a second tolerance ledger, a quotient stream, a selected limit representative, host real equality, countable choice, or ambient completeness, and none of those coordinates is present. □

Falsifiable prediction: Removing the common dyadic tolerance row breaks the RegSeqRat-to-RealUp classifier handoff while preserving the StreamName schedule.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchylimitseal: none is bijective because this packet stores the shared classifier-normal-form route across all four source surfaces rather than any single source surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}$ packet is generated from a common dyadic tolerance row, finite StreamName window, RegSeqRat readback, RealUp seal row, completion-seal classifier, componentwise hsame transports, displayed Cont routes, provenance, and local naming. The construction first exposes the tolerance, then the finite window, then the regular rational readback, and only then the Real-facing seal. The classifier-normal-form row records that all consumers use the same finite route. It is a shared BEDC carrier for Cauchy readback discipline, not a quotient of streams.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the regular Cauchy classifier normal-form carrier, its six obligation rows, and the Real handoff theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyClassifierNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient-stream equality, selected completed limits, host real equality, countable choice, or ambient completeness.

Upgrade path: To reach obligationClosure, formalize the shared dyadic-window-readback-seal route and prove its finite classifier stability.

Chapter 981

A Concrete Naming Certificate for $\mathsf{DyadicWindowSeal}^{\uparrow}$

The $\mathsf{DyadicWindowSeal}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as a finite service boundary before a $\mathsf{Real}^{\uparrow}$ seal consumer is allowed to read completion-facing data. It sits between the dyadic precision surface of chapter 291, the finite stream windows of chapter 126, the observation-budget surface of chapter 611, the seal synchronizer of chapter 886, and the completion-seal classifier of chapter 900. The packet is a finite Real-and-Completion seal service, not a quotient stream, selected host limit, host real equality, hidden selector, unbounded schedule, or ambient completeness principle.

Definition 981.1 (Dyadic window seal carrier)**.** *A* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (q, w, r, b, z, s, h, c, p, n)$$

with the following displayed rows:

1. *q is the dyadic precision and request row read from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger;*
2. *w is the finite* $\mathsf{StreamName}^{\uparrow}$ *window served for that request;*
3. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained through the window;*
4. *b is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row that admits the finite observation service;*
5. *z is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row tying the request, budget, shared threshold, and seal service together;*
6. *s is the* $\mathsf{Real}^{\uparrow}$ *seal row that may be consumed only after q, w, r, b, z are displayed;*
7. *h records componentwise* hsame *transport for the request, window, readback, budget, synchronizer, and seal rows;*
8. *c records the displayed* Cont *routes by which every consumer reaches those rows;*
9. *p is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *n is the local* $\mathsf{NameCert}_{\mathsf{DyadicWindowSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient stream representative, host real equality, selected host limit, hidden selector, unbounded observation schedule, or ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.DyadicWindowSealUp.DyadicWindowSealUp`

Theorem 981.2 (Dyadic window seal NameCert obligations)**.** *For the carrier of Definition 981.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicWindowSeal}^{\uparrow}}$ *surface consists of six finite rows:*

1. *carrier admission for* $W = (q, w, r, b, z, s, h, c, p, n)$ *as one displayed* BHist *packet;*
2. *dyadic request exactness identifying* q *with the precision and request ledger row served by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. *finite-window readback exactness identifying* w *and* r *as the* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for the same request;*
4. *observation-budget admission through* b *before any* $\mathsf{Real}^{\uparrow}$ *seal row is exposed;*
5. *synchronizer agreement through* z*, forcing the request, budget, shared-threshold, and seal rows to be read as one finite route;*
6. *transport, route, provenance, and local naming exactness through* h, c, p, n*.*

Lean `BEDC.Derived.DyadicWindowSealUp.TasteGate` *must provide explicit* `Nontrivial DyadicWindowSealUp` *and* `FieldFaithful DyadicWindowSealUp` *instances for this same carrier surface.*

Lean4 ✓ `BEDC.Derived.DyadicWindowSealUp.DyadicWindowSealTasteGate single carrier alignment`

Proof. Project the carrier tuple in Definition 981.1. The request, window, readback, budget, synchronizer, seal, transport, route, provenance, and naming rows are all displayed coordinates of the packet.

The dyadic request row fixes the precision service, and the window and readback rows fix the finite observation through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. The budget row admits the service only as a finite observation, while the synchronizer row ties that budget to the same seal route.

Finally the transport and continuation rows authenticate componentwise movement of the displayed service. Repacking those rows with p and n gives the local NameCert surface and leaves no carrier coordinate through which a hidden selector, quotient stream, host real equality, selected limit, unbounded schedule, or ambient completeness premise can enter. □

Theorem 981.3 (Dyadic window seal budget pullback)**.** *Let* $W = (q, w, r, b, z, s, h, c, p, n)$ *be an accepted* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *packet. Every observation-budget consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* s *factors through the finite pullback route*

$$q \longmapsto w \longmapsto r \longmapsto b \longmapsto z \longmapsto s.$$

The route is exhausted by the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the same packet, and exports no quotient stream equality, host real equality, selected host limit, hidden selector, unbounded schedule, or ambient completeness.*

Proof. Begin with the obligation surface of Definition 981.2. It requires the dyadic request row, stream window, regular readback, observation budget, synchronizer, and seal row to belong to one accepted packet.

Read the consumer through the observation-budget row. Since b is displayed as a coordinate of the same carrier, the admitted service can inspect only the request, window, readback, and budget rows already named by q, w, r, b.

Now apply synchronizer agreement. The row z binds that budget route to the shared seal route ending at s, while h, c, p, n record the componentwise transport, continuation replay, provenance, and local naming. Carrier inspection supplies the stated exclusions. □

Theorem 981.4 (Dyadic window seal non-escape)**.** *No accepted* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *read can export a* $\mathsf{Real}^{\uparrow}$ *seal consumer without first displaying the dyadic request row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RealObservationBudget}^{\uparrow}$ *row, and* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row of the same packet. Any read that bypasses one of those rows has no remaining coordinate from which to export a completion-facing seal service.*

Proof. Project the accepted packet to $W = (q, w, r, b, z, s, h, c, p, n)$. The only seal-facing coordinate is s, and the only service route into it is the displayed finite chain through q, w, r, b, z.

Use Definition 981.3. It places every observation-budget consumer of s on the same pullback route and authenticates the route through h, c, p, n.

The carrier definition then rules out all bypasses. There is no quotient-stream coordinate, host equality test, selected host limit, hidden selector, unbounded schedule, or ambient completeness field that could replace a missing request, window, readback, budget, or synchronizer row. □

Theorem 981.5 (Dyadic window seal Real handoff)**.** *Every accepted* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *consumer that hands data to* $\mathsf{Real}^{\uparrow}$ *hands only the finite packet rows*

$$q, w, r, b, z, s, h, c, p, n.$$

In particular, a $\mathsf{Real}^{\uparrow}$ *seal read factors through the displayed dyadic precision request, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, observation budget, and synchronizer row before the seal is consumed.*

Proof. Apply Definition 981.2 to identify the displayed Real-and-Completion service surface. The only rows available to a consumer are the request, window, readback, budget, synchronizer, seal, transport, route, provenance, and local naming rows.

Apply Definition 981.3. It orders every seal read through q, w, r, b, z before the consumer reaches s.

Finally apply Definition 981.4. Since no alternate analytic or quotient coordinate is present, the handoff to $\mathsf{Real}^{\uparrow}$ is exactly the finite packet route. □

Theorem 981.6 (Dyadic window seal finite source exhaustion)**.** *Every accepted* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *read that faces a* $\mathsf{Real}^{\uparrow}$ *consumer is exhausted by exactly the displayed finite source chain*

$$q, w, r, b, z, s, h, c, p, n.$$

Here q *is the dyadic request from* $\mathsf{DyadicRatCore}^{\uparrow}$*,* w *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window,* r *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* b *is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *row,* z *is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row,* s *is the* $\mathsf{Real}^{\uparrow}$ *seal row, and* h, c, p, n *are precisely the* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows. Conversely, any packet carrying those rows with the displayed transport, continuation, provenance, and local naming data repacks as the same* $\mathsf{DyadicWindowSeal}^{\uparrow}$ *finite service boundary. The exhaustion exports no quotient stream equality, selected host limit, host real equality, hidden selector, unbounded schedule, ambient completeness, or bridge data.*

Proof. Start from Definition 981.2. It lists the carrier row inventory, so an accepted $\mathsf{DyadicWindowSeal}^{\uparrow}$ packet exposes only the request, window, readback, budget, synchronizer, seal, componentwise transport, continuation, provenance, and local naming rows.

Apply Definition 981.3. The Real-facing read is ordered along the finite route $q \longmapsto w \longmapsto r \longmapsto b \longmapsto z \longmapsto s$, and the remaining support rows authenticate that same route inside the packet.

Apply Definition 981.4. Since there is no bypass coordinate, no missing row can be supplied by quotient stream equality, a selected host limit, host real equality, a hidden selector, an unbounded schedule, ambient completeness, or bridge data.

For the converse, take a packet carrying exactly the displayed source chain and support rows. The NameCert obligations repack those rows into the carrier tuple, the pullback theorem supplies the ordered service boundary, and Definition 981.5 identifies the resulting consumer interface as the same $\mathsf{Real}^{\uparrow}$-facing finite handoff. □

Falsifiable prediction: Every accepted DyadicWindowSealUp RealUp consumer factors through the displayed dyadic window and synchronizer rows before any seal read.

Independence witness: Nearest siblings: real-observation-budget, cauchy-seal-budget-synchronizer, finite-tail-diagonal-seal, regular-cauchy-limit-classifier, and real-completion-seal-classifier; none is bijective because DyadicWindowSealUp uniquely carries the dyadic request, finite StreamName window, RegSeqRat readback, observation budget, synchronizer row, and RealUp seal in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicWindowSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicWindowSeal}^{\uparrow}$ packet is generated from a dyadic precision request, finite StreamName window, RegSeqRat readback, observation-budget row, seal synchronizer row, RealUp seal, componentwise transports, displayed continuations, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicWindowSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 981.1, Definition 981.2, Definition 981.3, Definition 981.4, Definition 981.5, and Definition 981.6 bind the finite Real-and-Completion service packet to the displayed dyadic request, window, readback, budget, synchronizer, seal, transport, route, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicWindowSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, host real equality, selected host limits, hidden selectors, unbounded schedules, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier, obligation, pullback, non-escape, Real handoff, and finite-source exhaustion targets for the same finite packet surface.

Chapter 982

A Concrete Naming Certificate for $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$

The $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$ surface isolates the reusable commitment row inside the inscription-event vision of chapter 9397. It sits between the full inscription-event packet of chapter 519 and the public residue ledger of chapter 520: the event chapter records a complete accepted inscription packet, while this seal keeps only the finite source, accepted commitment, consumer, ledger, transport, route, provenance, and naming rows needed by downstream audit readers.

Definition 982.1 (InscriptionCommitmentSeal carrier)**.** *An* $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, U, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row from which the commitment is read;*
2. *M is the accepted named commitment row, read as public record data rather than private intention;*
3. *U is the intended downstream consumer row for the commitment;*
4. *G is the public* GAP *ledger row attached to the commitment and consumer;*
5. *H records componentwise* hsame *transports for S, M, U, G together with route, provenance, and local naming rows;*
6. *C records the displayed* Cont *routes by which a consumer reads the source, commitment, consumer, and ledger rows;*
7. *P is the* Pkg *provenance over* BHist, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionCommitmentSeal}^{\uparrow}}$ *row.*

The classifier accepts only packets with these eight displayed rows. It has no coordinate for a complete inscription event, unchecked route replay, private motive, host process trace, hidden consumer right, or ledger entry outside G.

Lean4 ✓ `BEDC.Derived.InscriptionCommitmentSealUp.InscriptionCommitmentSealTasteGate single carrier alignment`

Theorem 982.2 (InscriptionCommitmentSeal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{InscriptionCommitmentSeal}^{\uparrow}}$ *surface for Definition 982.1 consists exactly of carrier exposure for*

$$K = (S, M, U, G, H, C, P, N),$$

source-row readback through S*, accepted-commitment readback through* M*, consumer accountability through* U*, public GAP-ledger readback through* G*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Proof. Carrier exposure is projection from the displayed packet. The rows S, M, U, G, H, C, P, N are the complete inventory available to a consumer of the seal.

Read S, M, U, G in order. The source row fixes the finite record state, the accepted commitment row supplies the named public commitment, the consumer row names the permitted reader, and the GAP ledger records the public residue that accompanies that reader.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a broader event packet, hidden route, private motive, host trace, or unlisted ledger entry, the NameCert surface is exhausted by the displayed seal rows. □

Theorem 982.3 (InscriptionCommitmentSeal consumer-ledger split)**.** *For every accepted* InscriptionCommitmentSe *carrier* $K = (S, M, U, G, H, C, P, N)$*, deleting the accepted commitment row* M *breaks acceptance for the named commitment, while deleting the public GAP ledger row* G *breaks non-escape for the consumer ledger. The two failures are distinct:* M *is the record admitted for the consumer, and* G *is the public residue ledger attached to that admission.*

Proof. Project the carrier. The accepted commitment row and the public GAP ledger row are separate coordinates of the finite packet.

Use the NameCert obligations of Definition 982.2. Commitment readback is assigned to M, while public ledger readback is assigned to G together with the consumer row U and the displayed routes in C.

If M is absent, the packet no longer exposes an accepted named commitment for the consumer. If G is absent, the packet may still name a commitment row, but it loses the public ledger boundary that prevents consumer escape beyond the displayed residue. Thus the two rows support distinct downstream reads. □

Theorem 982.4 (InscriptionCommitmentSeal sibling factorization)**.** *Every downstream audit read of an accepted* $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$ *packet factors through the inscription-event carrier of chapter 519 for accepted-record locality and through the inscription GAP ledger of chapter 520 for public residue accountability. The seal exports only*

$$S, M, U, G, H, C, P, N$$

to that reader, not the full event route, private agency data, or an additional ledger.

Proof. Start from the accepted carrier and project the four content rows S, M, U, G. The rows S, M, U are the inscription-facing source, accepted commitment, and consumer surface, while G is the public residue ledger.

Apply the two sibling readings. The inscription-event chapter supplies the surrounding accepted-record discipline for S, M, U, and the inscription GAP-ledger chapter supplies the residue-ledger discipline for G.

The structural rows H, C, P, N then carry exactly the seal packet to downstream audit readers. They do not reconstruct the full event route, add private agency data, or create a second ledger row. □

Falsifiable prediction: Deleting the accepted commitment row or the public GAP ledger row breaks a distinct downstream read: acceptance for the named commitment and non-escape for the consumer ledger.

Independence witness: Nearest siblings: inscriptionevent, inscriptiongapledger, kernelacceptancetrace, tastegateadmission. InscriptionEventUp has a full event packet rather than a seal; InscriptionGapLedgerUp has the residue ledger but no accepted commitment row; KernelAcceptanceTraceUp has checker acceptance but no inscription consumer; TasteGateAdmissionUp has admission rows but no GAP commitment ledger. No bijection preserving all rows exists.

The Lean target family $BEDC.Derived.InscriptionCommitmentSealUp$ should provide Nontrivial and FieldFaithful instances for the displayed carrier before a TasteGate theorem records the same seal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionCommitmentSeal}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$ packet is generated from a finite source row, an accepted named commitment row, an intended consumer row, a public GAP ledger row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.InscriptionCommitmentSealUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionCommitmentSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 982.1, Definition 982.2, Definition 982.3, Definition 982.4, the falsifiable prediction, independence witness, and TasteGate marker bind the mature paper-side commitment-seal export surface over the displayed inscription-event and inscription-gap-ledger consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionCommitmentSeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.InscriptionCommitmentSealUp.InscriptionCommitmentSealTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridge checking, complete inscription-event ontology, private intention, hidden host traces, unchecked routes, or ledger rows outside the displayed packet.

Upgrade path: The remaining upgrade is bridgeChecked verification for the displayed commitment-seal export surface; the paper-side closure is mature over the inscription-event and inscription-gap-ledger consumer rows.

Chapter 983

A Concrete Naming Certificate for $\mathsf{CauchySharedTailModulus}^{\uparrow}$

The $\mathsf{CauchySharedTailModulus}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as a finite shared-tail threshold certificate. It consumes the explicit modulus extraction surface of chapter 641, the dyadic tail-radius ledger of chapter 757, the finite window service of chapter 126, the regular rational readback of chapter 301, and the Real seal boundary of chapter 38. Its purpose is to synchronize two regular-Cauchy tail requests through one displayed dyadic threshold before any $\mathsf{Real}^{\uparrow}$ completion-facing seal is exposed.

Definition 983.1 (Cauchy shared-tail modulus carrier)**.** *A* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *sources whose tails are being synchronized for a Real-completion consumer;*
2. M_0 *and* M_1 *are the displayed* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *rows supplying the two finite tail requests;*
3. τ *is the single shared dyadic tail-threshold row, read through the* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *service;*
4. D *is the dyadic tolerance ledger witnessing that both requests are served at the same threshold* τ*;*
5. W_0 *and* W_1 *are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened by* τ *for the two sources;*
6. Q_0 *and* Q_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the two windows and the shared dyadic ledger;*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the two readbacks have passed through the same shared threshold;*
8. H *records componentwise* hsame *transport for the sources, extracted moduli, shared threshold, dyadic ledger, windows, readbacks, Real seal, provenance, and local name;*
9. C *records the displayed* Cont *routes from each extracted modulus to* τ*, from* τ *to the two windows, from the windows to the readbacks, and from the readbacks to the Real seal;*

10. *P is the* Pkg *provenance over* CauchyModulusExtraction$^{\uparrow}$, DyadicTailRadiusLedger$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyTailMeet$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySharedTailModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, independent per-source seal exposure, hidden threshold replacement, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchySharedTailModulusUp.CauchySharedTailModulusTasteGate single carrier alignment`

Theorem 983.2 (Cauchy shared-tail modulus NameCert obligations)**.** *For the carrier of Definition 983.1, the local* $\mathsf{NameCert}_{\mathsf{CauchySharedTailModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for S, paired request exposure through M_0, M_1, shared-threshold exposure through τ, D, window service through W_0, W_1, regular readback through Q_0, Q_1, Real seal ordering through E, componentwise stability through H, displayed replay through C, provenance through P, and local naming through N. A completion-facing consumer obtains no* Real$^{\uparrow}$ *seal read until both requests have been routed through the shared dyadic threshold.*

Proof. Project the displayed carrier. The rows $R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$ are all of the public packet, so every tail request, threshold, window, readback, seal, transport, route, provenance, and local name is visible before the certificate is consumed.

The sibling modulus and dyadic rows supply the request boundary. The extracted modulus rows come from chapter 641, while the shared dyadic tolerance row comes from chapter 757; together they force both tail requests to meet at the same finite threshold τ.

The window and readback rows then route through chapter 126 and chapter 301. The Real seal row is exposed only after those finite rows have been replayed through C and transported by H.

Since the carrier contains no quotient stream, host real equality, selected limit, hidden threshold replacement, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 983.3 (Cauchy shared-tail modulus tail synchronization)**.** *Every accepted* CauchySharedTailModulus$^{\uparrow}$ *packet synchronizes the two tail requests through the same row τ. In particular, any* RegularCauchyTailMeet$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* StreamName$^{\uparrow}$ *consumer of the packet reads M_0 and M_1 before τ, reads both windows W_0, W_1 only through τ, and reads the two regular outputs Q_0, Q_1 only after the shared dyadic ledger D has been displayed.*

Lean4 ✓ `BEDC.Derived.CauchySharedTailModulusUp.CauchySharedTailModulusCarrier tail synchronization`

Proof. Start with the carrier definition. The two request rows M_0, M_1 and the single threshold row τ are separate displayed coordinates, but there is no second threshold coordinate.

Use the local obligations of Definition 983.2. Paired request exposure forces both requests to be visible before the threshold is consumed, and shared-threshold exposure forces both window services through the same τ, D rows.

The sibling RegularCauchyTailMeet$^{\uparrow}$ route reads the same threshold as its shared-tail row. Thus every tail-meet, StreamName, or RegSeqRat consumer remains on the finite route through $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1$. □

Theorem 983.4 (Cauchy shared-tail modulus Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing read of an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet factors through the displayed route*

$$M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

The Real seal E *is therefore a handoff from one shared dyadic tail threshold, not from two independent tail witnesses or an ambient completeness oracle.*

Lean4 ✓ `BEDC.Derived.CauchySharedTailModulusUp.CauchySharedTailModulusCarrier real seal handoff`

Proof. Read the Real seal coordinate in the carrier. The only $\mathsf{Real}^{\uparrow}$ seal row is E, and it appears after the paired request, threshold, dyadic, window, and readback rows.

Apply Definition 983.3. It places both regular-Cauchy tails on the same threshold τ before either readback can reach the seal.

Finally use the displayed transport, route, provenance, and naming rows H, C, P, N. They authenticate the same finite route and do not add an independent seal coordinate, so the completion-facing read is exactly the listed handoff. □

Theorem 983.5 (Cauchy shared-tail modulus carrier admission exactness)**.** *An accepted* $\mathsf{CauchySharedTailModulus}$ *packet exposes exactly the carrier rows*

$$R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

The local $\mathsf{NameCert}_{\mathsf{CauchySharedTailModulus}^{\uparrow}}$ *admission read has no additional coordinate for quotient stream equality, host real equality, selected limits, independent per-source seal exposure, hidden threshold replacement, or ambient completeness.*

Proof. Project the carrier of Definition 983.1. The accepted packet is the displayed finite tuple, so carrier admission can expose precisely its two source rows, two extracted modulus rows, one shared threshold row, the dyadic ledger, two windows, two readbacks, the Real seal, transport, route, provenance, and local name.

Apply the local NameCert obligations of Definition 983.2. They match the admission surface field by field and assign each row to one visible obligation: paired request exposure, shared-threshold exposure, paired window service, regular readback, Real seal ordering, componentwise stability, displayed replay, provenance, and local naming.

Finally use the Real seal handoff of Definition 983.4. Any completion-facing read still factors through the same displayed rows, so a purported extra coordinate would not be read from the accepted carrier. □

Theorem 983.6 (Cauchy shared-tail modulus paired-window readback service)**.** *For every accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet, both finite windows* W_0, W_1 *and both regular readbacks* Q_0, Q_1 *are served from the one shared threshold row* τ *together with its dyadic ledger row* D*. No consumer reads either window or readback through a private threshold row.*

Proof. Start from carrier admission exactness, Definition 983.5. The only threshold row available to the accepted packet is τ, and the only dyadic threshold ledger row is D.

Apply the synchronization theorem Definition 983.3. It orders the paired request rows M_0, M_1 before τ, D, then routes both windows W_0, W_1 and both readbacks Q_0, Q_1 through that shared threshold service.

The paired-window obligation and Real seal handoff keep the downstream read on the same row. Thus every window and readback consumer is served from τ, D, and a private-threshold window is not a read of the displayed packet. □

Theorem 983.7 (Cauchy shared-tail modulus route replay provenance)**.** *For an accepted* CauchySharedTailModulus *packet, the rows H, C, P, N transport, replay, package, and name only the displayed route*

$$R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E.$$

They do not certify an alternate route with hidden thresholds, independent Real seals, quotient stream equality, host real equality, selected limits, or ambient completeness.

Proof. Use carrier admission exactness to identify the public route inventory. The non-structural rows before H, C, P, N are exactly $R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E$, so transport and replay have no hidden source outside this list.

Next apply the local NameCert obligations. The transport row H is componentwise, the replay row C follows the displayed continuation routes, the provenance row P packages the cited concrete BEDC surfaces, and N names the local certificate.

Finally compose with paired-window readback service and the Real seal handoff. Those two theorems force the middle route and the completion-facing seal to use the same $\tau, D, W_0, W_1, Q_0, Q_1, E$ path, so H, C, P, N certify only the displayed route. □

Theorem 983.8 (Cauchy shared-tail modulus paired-window obligation)**.** *For every accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet, the two* $\mathsf{StreamName}^{\uparrow}$ *window rows W_0, W_1 are exposed only after the paired request rows M_0, M_1 and the single dyadic threshold package τ, D have been displayed. Thus no consumer of the packet may read either window as a unilateral source window or as a window opened by a private threshold.*

Proof. Project the carrier of Definition 983.1. The rows $M_0, M_1, \tau, D, W_0, W_1$ are displayed in the finite packet, with a single threshold coordinate τ and a single dyadic tolerance ledger D serving both requests.

Apply Definition 983.3. It places M_0 and M_1 before the shared threshold read, and it routes both StreamName windows through the same τ, D rows.

The public route row C and transport row H replay that ordering componentwise. Since the carrier has no hidden threshold or unilateral window coordinate, the only accepted window read is the paired read through M_0, M_1, τ, D. □

Theorem 983.9 (Cauchy shared-tail modulus threshold uniqueness obligation)**.** *Let a consumer route claim to read an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet while replacing the shared threshold row τ by two independent threshold rows. Then the route is not a read of the carrier in Definition 983.1. Any completion-facing route that reaches the Real seal must use the unique public threshold coordinate τ and its dyadic ledger row D.*

Proof. Inspect the carrier coordinates. The packet contains one threshold row τ and one dyadic ledger row D; it contains no coordinates for separate left and right thresholds.

Apply carrier-coordinate uniqueness in the displayed tuple. A route that supplies two independent threshold rows asks for two carrier coordinates where the carrier presents only one, so it leaves the packet surface before any Real seal read is considered.

Now apply Definition 983.4. Every completion-facing Real seal read factors through the displayed route containing the same τ, D rows. Therefore a route that replaces τ cannot be a read of the accepted carrier. □

Theorem 983.10 (Cauchy shared-tail modulus paired-source readback)**.** *Every accepted* CauchySharedTailModulus *packet reads both* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows R_0, R_1 before the shared threshold row τ is consumed.*

A downstream regular-Cauchy, StreamName, RealUp, or completion-facing consumer may inspect Q_0, Q_1 only as readbacks of the paired source rows through $M_0, M_1, \tau, D, W_0, W_1$.

Proof. Start with the carrier definition. The source rows R_0, R_1 are displayed before the extracted modulus rows, the single threshold row, the dyadic ledger, the window rows, and the readback rows.

Apply the tail synchronization theorem. It orders the two modulus requests before τ and routes both windows and both readbacks through the shared threshold service.

Finally use threshold uniqueness. Since the accepted packet has no private threshold coordinate, a consumer that reaches Q_0, Q_1 without reading both source rows through the common route is not reading the displayed shared-tail packet. □

Theorem 983.11 (Cauchy shared-tail modulus dyadic ledger exactness)**.** *For every accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet, the dyadic ledger row D is the exact common tolerance witness for both finite windows W_0, W_1 and both regular readbacks Q_0, Q_1. No consumer may replace D by two private dyadic ledgers, omit it from one side, or read the Real seal row E as if the two windows had independent tolerance witnesses.*

Proof. Project the displayed packet. It contains one dyadic ledger row D paired with the single threshold row τ.

Use paired-window readback service. Both windows and both readbacks are served from the same τ, D pair, so D is not an optional annotation on one side of the route.

The Real seal handoff then carries the same dyadic ledger into the completion-facing read. Since the seal factors through $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$, omitting or duplicating D leaves the accepted carrier surface. □

Theorem 983.12 (Cauchy shared-tail modulus completion-consumer non-escape)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet factors through*

$$M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

The consumer cannot obtain selected limits, quotient real equality, host real equality, ambient completeness, two private thresholds, or a Real seal detached from the paired source readback and dyadic ledger rows.

Proof. First apply the Real seal handoff. It identifies the completion-facing route as the displayed finite path through the two modulus rows, the shared threshold, the dyadic ledger, paired windows, paired readbacks, Real seal, transport, route, provenance, and local name.

Next use paired-source readback and dyadic ledger exactness. These rows ensure that Q_0, Q_1 remain readbacks of the two displayed sources through one common τ, D route.

Finally apply threshold uniqueness. The accepted carrier has no selected-limit coordinate, quotient-real coordinate, host-equality coordinate, ambient-completeness coordinate, or pair of private threshold coordinates, so a consumer that extracts any of those objects is outside the packet. □

Theorem 983.13 (Cauchy shared-tail modulus field-faithfulness obligation)**.** *For the carrier*

$$S = (R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N),$$

the public fields $R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$ determine the local $\mathsf{NameCert}_{\mathsf{CauchySharedTailModul}}$ *read. No proper subtuple of these fields exposes the same completion-facing* $\mathsf{Real}^{\uparrow}$ *seal, because omitting any paired source, request, threshold, dyadic ledger, window, readback, seal, transport, route, provenance, or local name removes a row used by the displayed handoff.*

Proof. Start from Definition 983.2. It lists the carrier admission row, paired requests, shared threshold, dyadic ledger, paired windows, paired readbacks, Real seal, componentwise stability, route replay, provenance, and local name as the public NameCert surface.

Use Definition 983.3 for the middle of the read. It forces the two request rows, the single threshold row, the dyadic ledger, both windows, and both regular readbacks to appear in the same finite route.

Finally apply Definition 983.4. The Real seal is exposed only through the complete displayed route

$$M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

Dropping any listed field breaks either carrier admission, the synchronized finite route, or the authenticated Real seal handoff. Hence the full public tuple determines the NameCert read, while no proper subtuple exposes the same seal. □

Theorem 983.14 (Cauchy shared-tail modulus public obligation package)**.** *For the carrier*

$$S = (R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N),$$

carrier admission, paired request exposure, the shared threshold, the dyadic ledger, paired $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise transport, route replay, package provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySharedTailModulus}^{\uparrow}}$ *naming form one public obligation package. A consumer cannot detach any of these rows from the package while still claiming to read the shared-tail Real-completion surface.*

Proof. Project the displayed packet S. The carrier definition exposes exactly the two sources, two modulus rows, one threshold row, one dyadic ledger, two windows, two readbacks, one Real seal, transport, route, provenance, and local name.

Use the local NameCert obligations and the tail synchronization theorem. They bind carrier admission, paired request exposure, shared-threshold exposure, window service, regular readback, and Real seal ordering to the same finite route.

Now apply the paired-window obligation, threshold uniqueness obligation, and field-faithfulness obligation. These rows prevent a consumer from replacing the shared threshold, splitting the paired windows into private routes, or dropping a public field while preserving the completion-facing Real seal read. □

Theorem 983.15 (Cauchy shared-tail modulus Real-completion obligation closure)**.** *Every Real-completion-facing consumer route for an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet must pass through the public obligation package of Definition 983.14 before reading the* $\mathsf{Real}^{\uparrow}$ *seal. In particular, no such route may replace the shared threshold row* τ *and its dyadic ledger row* D *by private per-source thresholds.*

Proof. First apply the public obligation package theorem. It places carrier admission, paired request exposure, shared-threshold exposure, paired windows, paired readbacks, Real seal ordering, transport, replay, provenance, and local naming on one public surface.

Then use the Real seal handoff theorem. Any completion-facing read of the Real seal factors through the displayed route containing $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$.

Finally apply the threshold uniqueness obligation. Since the accepted carrier has only the shared threshold coordinate τ and ledger coordinate D, a route with private per-source thresholds is not a read of the packet. □

Theorem 983.16 (Cauchy shared-tail modulus threshold transport stability)**.** *For an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet, componentwise* hsame *transport along* H *preserves the shared threshold row* τ *and the dyadic ledger row* D *as one paired service for both source requests. Any transported consumer route that introduces a private left threshold, a private right threshold, or a second dyadic ledger is outside the local* $\mathsf{NameCert}_{\mathsf{CauchySharedTailModulus}^{\uparrow}}$ *package.*

Proof. Read the carrier rows for H. The transport row is componentwise for the sources, extracted moduli, shared threshold, dyadic ledger, windows, readbacks, Real seal, provenance, and local name; it is not a constructor for new threshold coordinates.

Apply threshold uniqueness and dyadic ledger exactness. These two rows identify τ and D as the only threshold service available to the accepted packet, and they keep both finite windows and both regular readbacks attached to that same service.

Now transport the displayed route by H. Since every transported component is a component of

$$R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, P, N,$$

the transported packet still has one threshold row and one ledger row. A route with private thresholds or a second ledger therefore asks for rows not carried by the packet. □

Theorem 983.17 (Cauchy shared-tail modulus window readback transport)**.** *For an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet, the paired windows* W_0, W_1*, paired readbacks* Q_0, Q_1*, and Real seal* E *are transported only along the shared route recorded by* H *and* C*. No transported read may detach* W_0 *or* W_1 *from* τ, D*, detach* Q_0 *or* Q_1 *from the paired windows, or expose* E *without the displayed shared-tail route.*

Proof. Start with paired-window readback service. It serves W_0, W_1, Q_0, Q_1 from the same τ, D rows, so the window and readback transport has a single threshold source.

Use route replay provenance. The rows H and C transport and replay only the displayed route through the two sources, two extracted moduli, the shared threshold, the dyadic ledger, the paired windows, the paired readbacks, and the Real seal.

Finally apply the Real seal handoff. The seal row E is downstream of the paired readbacks and the same shared threshold route, so a transported consumer that detaches a window, readback, or seal from H, C is not reading the accepted package. □

Theorem 983.18 (Cauchy shared-tail modulus obligation closure package)**.** *For the carrier*

$$S = (R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N),$$

carrier admission, local $\mathsf{NameCert}$ *obligations, tail synchronization, threshold uniqueness, threshold transport stability, paired window and readback transport, Real seal handoff, field-faithfulness, and the checked TasteGate carrier alignment form one* $\mathsf{obligation}$ *surface for* $\mathsf{CauchySharedTailModulus}^{\uparrow}$*. The package has no coordinate for private thresholds, independent Real seals, quotient stream equality, host real equality, selected limits, or ambient completeness.*

Proof. First collect carrier admission and the local NameCert obligations. They expose exactly the displayed rows of S and assign them to paired source requests, one shared threshold, one dyadic ledger, paired windows, paired readbacks, one Real seal, componentwise transport, replay, provenance, and local naming.

Next use synchronization, threshold uniqueness, and threshold transport stability. These rows bind M_0, M_1, τ, D into one shared finite threshold service and exclude private thresholds under both direct reads and componentwise hsame transport.

Then apply window readback transport and Real seal handoff. They keep W_0, W_1, Q_0, Q_1, E on the same H, C route, so the completion-facing read cannot detach the seal from the paired finite route.

Finally combine field-faithfulness with the checked TasteGate carrier alignment. The public tuple is the whole local package surface, and the TasteGate marker records that the carrier alignment is a single visible packet. Thus the listed rows assemble the obligation surface without adding any hidden completion or quotient coordinate. □

Theorem 983.19 (Cauchy shared-tail modulus falsifiable prediction)**.** *If a consumer accepts a* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet while reading a* $\mathsf{Real}^{\uparrow}$ *seal before the shared threshold row* τ *or while replacing* τ *by two independent threshold rows, then that consumer is not reading the packet of Definition 983.1. The accepted packet exposes one shared dyadic threshold before any Real seal route.*

Proof. Inspect the carrier coordinates. There is exactly one shared threshold row τ and exactly one Real seal row E.

Use the NameCert obligations to order the public read. The paired request rows, the shared dyadic threshold, the dyadic ledger, both StreamName windows, and both RegSeqRat readbacks are displayed before the Real seal row can be consumed.

Replacing τ by two independent thresholds asks for carrier coordinates that are not present. Reading E before τ violates the displayed route order, so either behavior leaves the accepted packet surface. □

Falsifiable prediction: If a CauchySharedTailModulusUp packet is accepted, every RealUp completion-facing read factors through the displayed shared threshold before any Real seal is exposed.

Independence witness: cauchymodulusextraction, dyadictailradiusledger, regularcauchytailmeet, realcompletionsealclassifier: none is bijective because each lacks either the paired request rows, the shared threshold row, or the combined Real seal handoff carried by $\mathsf{CauchySharedTailModulus}^{\uparrow}$.

Theorem 983.20 (Cauchy shared-tail modulus obligation surface triad)**.** *The public obligation surface for an accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *packet is the triad consisting of carrier admission for*

$$S = (R_0, R_1, M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N),$$

the shared-threshold route through $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1$*, and the Real-seal ordering row* E *authenticated by* H, C, P, N*. A completion-facing consumer may read the package only after the paired requests, paired windows, paired readbacks, and the shared dyadic threshold have all been displayed.*

Proof. Carrier admission is Definition 983.1. It displays the two source rows, two modulus rows, the single shared threshold, the dyadic ledger, the paired window rows, the paired readbacks, the Real seal, transports, routes, provenance, and local name.

The shared-threshold route is forced by tail synchronization, paired-window readback service, and threshold uniqueness. These rows keep both sides on τ, D and exclude private threshold replacement.

The Real-seal ordering row is then supplied by the Real seal handoff and completion-consumer non-escape theorems. Since E is read only after the paired finite route and since H, C, P, N authenticate that route, the three listed surfaces form the public obligation triad. □

Theorem 983.21 (Cauchy shared-tail modulus TasteGate scoped admission)**.** *The checked TasteGate row following Definition 983.1 records the AI-origin admissibility surface for* $\mathsf{CauchySharedTailModulus}^{\uparrow}$*: it aligns the single displayed carrier with the local certificate rows before any Real-completion consumer may treat the shared-tail route as admitted.*

Proof. Read the checked marker attached to the carrier definition. Its target is the single-carrier alignment theorem for $\mathsf{CauchySharedTailModulus}^{\uparrow}$, so the admissibility row is attached to the same displayed tuple S rather than to a hidden completion principle.

Use the obligation closure package. That package combines carrier admission, NameCert obligations, threshold synchronization, transport stability, and Real seal handoff on the same finite surface.

Thus the TasteGate row is a scoped admission row for the existing public packet. It does not add a selected limit, quotient equality, host real equality, or ambient completeness coordinate. □

Theorem 983.22 (Cauchy shared-tail modulus Real seal obligation gate)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal route exported by* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *is public until the shared dyadic threshold* τ, D *and the paired readbacks* Q_0, Q_1 *have been displayed. A route that exposes* E *without the common threshold, both window rows, and both readback rows is outside the accepted shared-tail packet.*

Proof. Apply paired-source readback and paired-window readback service. They force the source rows, modulus rows, shared threshold, dyadic ledger, windows, and readbacks to appear on the same finite route before a completion-facing consumer can inspect the seal.

Use dyadic ledger exactness to keep D as the common tolerance witness for both sides. A route omitting D or replacing it by private ledgers leaves the accepted carrier surface.

Finally apply the Real seal handoff and completion-consumer non-escape theorems. They place E downstream of $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1$ and exclude any public Real seal detached from those rows. □

Theorem 983.23 (Cauchy shared-tail modulus downstream consumer factorization)**.** *Any downstream Real-completion,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, or* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySharedTailModulus}$ *packet factors through the displayed route*

$$M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

The $\mathsf{CauchyLimitSeal}^{\uparrow}$ *side is the sibling limit-seal route of chapter 386, the* $\mathsf{RealCompleteness}^{\uparrow}$ *side is the completion-facing seal route of chapter 450, and the finite source of both routes is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 126 and chapter 301. No selected limit, quotient stream, host real equality, countable choice rule, or ambient completeness coordinate is exposed by this factorization.*

Proof. Begin with tail synchronization. It places the two extracted modulus rows and the single shared threshold row on the same dyadic ledger before either finite window can be consumed.

Next use the Real seal handoff and the Real seal obligation gate. They force every completion-facing read of E to pass through the paired windows and paired regular readbacks over the common route $M_0, M_1, \tau, D, W_0, W_1, Q_0, Q_1$.

Now apply completion-consumer non-escape. A downstream limit-seal or completeness consumer can only receive the rows already displayed by the carrier, transported by H, replayed by C, cited by P, and named by N.

The sibling chapters named in the statement provide the consumer-side vocabulary for those finite rows. Since none of the source, seal, transport, replay, provenance, or naming rows contains

a selected limit, quotient stream, host equality, countable-choice selector, or ambient completeness coordinate, the factorization exposes no such object. □

The Lean TasteGate surface records Nontrivial CauchySharedTailModulus$^{\uparrow}$ and FieldFaithful CauchySharedTailMod instances for the displayed carrier.

Closure status for NameCert$_{\text{CauchySharedTailModulus}\uparrow}$.

Constructive story: A CauchySharedTailModulus$^{\uparrow}$ packet is generated from two regular-Cauchy source rows, extracted modulus rows, one shared dyadic threshold, a dyadic tolerance ledger, paired finite windows, paired regular readbacks, a Real seal handoff row, componentwise transports, displayed replay, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{CauchySharedTailModulus}\uparrow}$, maturePackageClosed)

Scope closed: Definition 983.1, Definition 983.4, Definition 983.12, Definition 983.14, and Definition 983.15 bind the BHist, hsame, Cont, Pkg, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicTailRadiusLedger$^{\uparrow}$, and Real$^{\uparrow}$ surface to one shared finite threshold before completion-facing reads.

Formal verification: FormalStatus(NameCert$_{\text{CauchySharedTailModulus}\uparrow}$, axiomClean)

Lean4 ✓ `BEDC.Derived.CauchySharedTailModulusUp.taste gate`

Lean target: `BEDC.Derived.CauchySharedTailModulusUp.CauchySharedTailModulusCarrier\_tail\_synchronization`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, independent per-source seal exposure, hidden threshold replacement, ambient completeness, or bridge checking.

Upgrade path: No further paper-axis upgrade remains unless a checked standard bridge is supplied.

Chapter 984

A Concrete Naming Certificate for $\mathsf{CellularLocalUpdateRule}^{\uparrow}$

The $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ surface isolates the local transition-table row named by the cellular-carrier alternative of chapter 9391. It sits beside the whole-rule and orbit carrier of chapter 934, the substrate packet of chapter 942, and the classifier-facing comparison of chapter 945. Its role is narrower than each sibling: it names only the displayed neighbourhood window, the selected rule-table row, and the visible next-row cell certified by that row.

Falsifiable prediction: Every accepted $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ packet determines a visible next-row cell only from its displayed neighbourhood window and rule-table row; no consumer can read a global orbit, hidden boundary condition, or host evaluator from the packet.

Independence witness: cellularautomaton, cellularsubstrate, cellularrowclassifier, cellularcarrierencoding, cellularsubstrateclassifier: none is bijective because those siblings package whole automata, substrates, row classifiers, or encodings while $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ packages only the local update rule surface.

Definition 984.1 (CellularLocalUpdateRule carrier)**.** *A* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, U, S', H, C, P, N)$$

with the following visible rows:

1. *S is the displayed current cell-state row;*
2. *W is the finite neighbourhood-window row surrounding the cell whose successor is being read;*
3. *U is the selected local rule-table row, containing exactly the displayed window pattern and its displayed output cell;*
4. *S' is the visible next-row cell determined by W and U;*
5. *H records componentwise* hsame *transports among the state row, window row, rule-table row, output cell, continuation route, package provenance, and local name;*
6. *C records the displayed* Cont *route by which a consumer reads S, W, U, S';*
7. *P is the* Pkg *provenance over* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{CellularRowClassifier}^{\uparrow}$, $\mathsf{CellularCarrierEncoding}^{\uparrow}$, $\mathsf{CellularSubstrateClassifier}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

8. N is the local $\mathsf{NameCert}_{\mathsf{CellularLocalUpdateRule}^{\uparrow}}$ *row.*

The classifier for U reads only the displayed neighbourhood window, the displayed rule-table row, the resulting next-row cell, the componentwise transport row, the continuation route, the package provenance, and the local name. It has no coordinate for a global orbit, a hidden boundary condition, or a host cellular evaluator.

Theorem 984.2 (CellularLocalUpdateRule NameCert obligations)**.** *For the carrier of Definition 984.1, the local* $\mathsf{NameCert}_{\mathsf{CellularLocalUpdateRule}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the packet $U = (S, W, U, S', H, C, P, N)$;*
2. *neighbourhood visibility for the finite window row W;*
3. *rule-row accountability for the displayed table row U and its visible output cell S';*
4. *componentwise stability and replay through H, C, P, N;*
5. *non-escape from global orbit reads, hidden boundary conditions, host evaluators, and rule entries absent from U.*

Proof. Carrier admission is projection from Definition 984.1. The packet displays the current state row, finite neighbourhood window, selected rule-table row, visible next-row cell, transport row, continuation route, provenance row, and local naming row before a consumer can read the local update.

Neighbourhood visibility and rule-row accountability are the two local coordinates of the carrier. The window row W supplies the only input pattern, while U supplies the only rule-table row and output cell available to the classifier.

The stability and replay rows are H, C, P, N. The hsame row transports corresponding displayed rows, the Cont row records the read route, and the package and local-name rows expose the certificate as a public NameCert object.

No remaining coordinate is present. A global orbit, hidden boundary condition, host evaluator, or absent rule entry therefore cannot be exported by the local $\mathsf{NameCert}_{\mathsf{CellularLocalUpdateRule}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.CellularLocalUpdateRuleUp.TasteGate.CellularLocalUpdateRuleTasteGate single carrier alignment`

Theorem 984.3 (CellularLocalUpdateRule falsifiable boundary)**.** *Every accepted* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ *packet determines its visible next-row cell by the finite route*

$$W, U \longrightarrow S' \longrightarrow H, C, P, N.$$

If a consumer obtains the next-row cell from a global orbit ledger, a hidden boundary condition, a host cellular evaluator, or a rule-table entry not displayed in U, then the consumer has left the accepted $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ *packet.*

Proof. Start with the obligation surface of Definition 984.2. The only neighbourhood input is W, and the only rule information is the displayed rule-table row U.

The output row S' is then read along H, C, P, N. These rows transport, replay, package, and name the displayed local rule surface; they do not add an orbit ledger or evaluator.

Thus a read that depends on a global orbit ledger, hidden boundary condition, host evaluator, or absent table row is not a read through W, U, S', H, C, P, N. It is outside the accepted packet. □

The Lean TasteGate namespace BEDC.Derived.CellularLocalUpdateRuleUp.TasteGate provides Nontrivial CellularLocalUpdateRule$^{\uparrow}$ and FieldFaithful CellularLocalUpdateRule$^{\uparrow}$ instances for the displayed carrier.

Closure status for NameCert$_{\text{CellularLocalUpdateRule}^{\uparrow}}$ ·

Constructive story: A CellularLocalUpdateRule$^{\uparrow}$ packet is generated from a displayed current cell-state row, finite neighbourhood-window row, selected local rule-table row, visible next-row cell, componentwise hsame transports, displayed Cont route, package provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.CellularLocalUpdateRuleUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{CellularLocalUpdateRule}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 984.1, Definition 984.2, and Definition 984.3 specify the finite local update rule carrier, its NameCert obligation rows, and its non-escape boundary.

Formal verification: FormalStatus(NameCert$_{\text{CellularLocalUpdateRule}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CellularLocalUpdateRuleUp.TasteGate.CellularLocalUpdateRuleTasteGate\_single\_carrier\_`

Bridge status: paperBridge

Not claimed: This chapter does not close global cellular orbits, hidden boundary conditions, host evaluator correctness, substrate universality, or bridge checking for CellularLocalUpdateRule$^{\uparrow}$.

Upgrade path: No further paper-side closure grade remains; Lean bridge checking remains a separate axis for the same finite local update packet.

Chapter 985

A Concrete Naming Certificate for HaltingFiniteBoundary$^{\uparrow}$

The HaltingFiniteBoundary$^{\uparrow}$ packet realizes the halting fixed-point obstruction of chapter 9368 as a finite boundary certificate. It sits beside the distinction packet of chapter 491, the finite trace packet of chapter 492, the positive trace classifier of chapter 978, the terminal trace-normal-form surface of chapter 683, and the public inscription record of chapter 519. These dependencies are row consumption: the carrier reads finite trace evidence, terminal trace shape, and public record accountability as displayed packet rows, not as an external halting ontology.

Definition 985.1 (Halting finite-boundary carrier)**.** *A* HaltingFiniteBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, F, R, D, T, I, H, C, P, N)$$

with the following displayed rows:

1. *A is the admitted certificate-history input row whose visible behavior is being inspected;*
2. *F is the finite positive* Cont *trace row, recording only the displayed segment available to the packet;*
3. *R is the Real or trace-consumer row that may read F as finite evidence but not as a total decision;*
4. *D is the diagonal refusal row, rejecting any request to turn absence of a displayed trace into divergence, non-termination, or a negative halting verdict;*
5. *T is the terminal trace-normal-form row consumed from the continuation-trace normal-form surface when a displayed terminal boundary is present;*
6. *I is the inscription-event row naming the public record in which the finite trace and refusal boundary are consumed;*
7. *H records componentwise* hsame *transports for the input, trace, consumer, refusal, terminal, inscription, package, and local naming rows;*
8. *C records the displayed* Cont *routes that replay the finite trace, enter the trace consumer, expose the diagonal refusal, and return the terminal and inscription rows to the same packet;*
9. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* HaltingDistinction$^{\uparrow}$*,* HaltingDistinctionTrace$^{\uparrow}$*,* HaltingTraceClassifier$^{\uparrow}$*,* ContinuationTraceNormalForm$^{\uparrow}$*,* InscriptionEvent$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a total halting predicate, a divergence oracle, an unbounded search certificate, a host evaluator, or a proof that every absent future continuation is absent.

Theorem 985.2 (Halting finite-boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}^{\uparrow}}$ *surface for Definition 985.1 consists of six finite obligations:*

1. *carrier exposure for the displayed packet $B = (A, F, R, D, T, I, H, C, P, N)$;*
2. *finite trace evidence, requiring every accepted Real or trace-consumer read to expose the finite* Cont *trace row F;*
3. *diagonal refusal, requiring every total-halting, divergence, or non-termination request outside F to be rejected by D;*
4. *terminal trace locality, requiring any normal-form or terminal read to pass through T and the displayed routes C;*
5. *public record accountability, requiring downstream inscription use to pass through I, P, N without adding private host evidence;*
6. *transport and ledger exactness, requiring A, F, R, D, T, I, P, N to be transported componentwise through H and replayed only through C.*

The later Lean `TasteGate.lean` *development must include* Nontrivial $\mathsf{HaltingFiniteBoundary}^{\uparrow}$ *and* FieldFaithful $\mathsf{HaltingFiniteBoundary}^{\uparrow}$ *instances with an explicit* `witness_pair` *and* `mk`*-derived field projection, so the gate cannot identify the finite trace row with the diagonal refusal row or forget either row from the carrier.*

Lean4 ✓ `BEDC.Derived.HaltingFiniteBoundaryUp.HaltingFiniteBoundaryTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 985.1. The rows list the inspected input, finite trace, consumer read, diagonal refusal, terminal trace boundary, inscription record, transports, replay routes, package provenance, and local name.

Finite trace evidence is the F row consumed by R. The route data C may replay that finite trace and transport it through H, but no coordinate supplies an unbounded search over future continuations.

Diagonal refusal is the D row. It is transported with F and R, so a consumer cannot turn absence of a displayed continuation into divergence or a negative verdict after transport.

Terminal trace locality and public record accountability are the T and I rows. They allow a consumer to read terminal trace shape and public inscription use only through the same finite packet, with provenance and local naming recorded by P and N.

Since every accepted read factors through $A, F, R, D, T, I, H, C, P, N$, the packet separates positive finite trace evidence from forbidden total halting decisions. □

Theorem 985.3 (Halting finite-boundary consumer inversion)**.** *Any accepted* $\mathsf{HaltingFiniteBoundary}^{\uparrow}$ *packet with a Real or trace consumer exposes both a finite* Cont *trace row and a diagonal refusal row. A packet lacking either row is not a carrier of Definition 985.1, and therefore cannot supply the local* $\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.HaltingFiniteBoundaryUp.HaltingFiniteBoundaryCarrier consumer inversion`

Proof. Let an accepted packet be presented to a Real or trace consumer. By carrier exposure it has the displayed form $B = (A, F, R, D, T, I, H, C, P, N)$.

The consumer row is R, and its finite evidence is F. By the finite trace obligation, any accepted read through R must expose F before the consumer may classify the observed behavior.

The boundary row is D. By diagonal refusal, the same accepted packet must expose D before any request outside the displayed finite trace is rejected. If F or D is missing, the tuple is not the carrier named by the definition, so the local certificate cannot be formed. □

Falsifiable prediction: Any accepted HaltingFiniteBoundaryUp packet with a Real or trace consumer must expose a finite Cont trace row and a diagonal refusal row; a packet lacking either row is rejected by carrier inversion.

Independence witness: haltingdistinction, haltingdistinctiontrace, haltingtraceclassifier, continuationtracenormalform, inscriptionevent: none is bijective because the new carrier combines finite trace evidence with a diagonal refusal boundary, while the siblings isolate distinction, trace, classifier, terminal trace, or inscription rows separately.

Closure status for $\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}\uparrow}$.

Constructive story: A $\mathsf{HaltingFiniteBoundary}^{\uparrow}$ packet is generated from an admitted certificate-history input row, a finite Cont trace row, a Real or trace-consumer row, a diagonal refusal row, a terminal trace-normal-form row, an inscription-event row, componentwise transports, replay routes, package provenance, and a local naming row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}\uparrow}, \mathsf{seed})$

Scope closed: Definition 985.1, Definition 985.2, and Definition 985.3 bind the finite trace evidence and diagonal refusal boundary for Real or trace consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaltingFiniteBoundary}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a total halting predicate, divergence decision, unbounded search certificate, host evaluator, Lean implementation, or standard bridge.

Upgrade path: Checked closure requires Lean encoding of the carrier plus Nontrivial and FieldFaithful TasteGate instances preserving the finite trace row, diagonal refusal row, consumer row, terminal row, inscription row, transports, routes, provenance, and local naming field projections.

Chapter 986

ObserverHsameClassifierUp NameCert construction

The $\mathsf{ObserverHsameClassifier}^{\uparrow}$ surface realizes the observer-history identity vision of chapter 9418 as a finite classifier row. It consumes the single-packet observer surface of chapter 479 and the pairwise licensed-observation surface of chapter 477. Its role is narrower than either sibling: it decides only whether displayed, licensed observations agree through hsame-mediated rows, and it has no coordinate for observer-substance or host equality.

Falsifiable prediction: If an accepted ObserverHsameClassifierUp packet changes only an unlicensed observer-substance row, then every public classifier read is unchanged because no such row is in the carrier.

Independence witness: Nearest siblings: observerhistidentity, observerhistoryidentity, observersubjectboundary, observerkerneltrace, observerfilter. The carrier is not bijective to any of them: it has a classifier-output and licensed-observation ledger row that the identity carriers and trace/filter packets do not have, while it lacks their full trace/filter selector coordinates.

Definition 986.1 (Observer hsame classifier carrier)**.** *An* $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H_0, H_1, A, R, L, D, C, P, N)$$

with the following displayed rows:

1. *H_0 and H_1 are the two observer-history rows being compared;*
2. *A is the licensed observation ledger, listing exactly the mark,* Cont, ProbeBundle, SigRel, *and classifier reads permitted to affect the decision;*
3. *R stores the componentwise* hsame *transports for the two histories, licensed observations, classifier-output row, continuation routes, package provenance, and local name;*
4. *L records the local ledger tying each licensed observation to its source history row and naming boundary;*
5. *D is the finite classifier-output row, recording agreement, disagreement, or refusal only after the relevant licensed observation row has passed through R and C;*
6. *C records the displayed* Cont *routes by which a consumer reads the histories, licensed observations, transports, and classifier output;*

7. *P is the* Pkg *provenance over* BHist, hsame, Cont, ProbeBundle, SigRel, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, *and* NameCert *rows;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through A, R, L, D, C, P, N. The carrier has no coordinate for unlicensed observation, observer-substance, host equality of observers, quotient history identity, or an observation outside the displayed ledger.

Theorem 986.2 (Observer hsame classifier NameCert obligations)**.** *For the carrier of Definition 986.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for $K = (H_0, H_1, A, R, L, D, C, P, N)$;*

2. *classifier transport through the componentwise* hsame *rows in R;*

3. *licensed-observation exactness for every row in A and no row outside A;*

4. *decision-row confinement, requiring every public classifier read to factor through D, L, R, C, P, N;*

5. *non-escape from unlicensed observation, observer-substance, host equality, quotient identity, and hidden classifier state.*

Lean4 ✓ `BEDC.Derived.ObserverHsameClassifierUp.ObserverHsameClassifierTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 986.1. The packet displays both observer-history rows, the licensed observation ledger, hsame transports, local ledger, classifier-output row, continuation routes, provenance, and local naming row before any public decision is available.

Classifier transport is componentwise. The row R moves each licensed observation and each displayed output row to the common hsame surface, while C records the only routes along which a consumer may read those transported rows.

Licensed-observation exactness follows by finite inventory of A and L. A public classifier decision may use a mark, Cont, ProbeBundle, SigRel, or prior classifier row only when that row appears in A with its source and boundary entry in L.

The remaining obligations follow from absence of coordinates. Since the carrier contains no observer-substance row, host equality row, quotient row, or hidden classifier state, every accepted public read is confined to A, R, L, D, C, P, N. □

Theorem 986.3 (Observer hsame classifier non-escape)**.** *Every public consumer of an accepted* $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *packet reads only the licensed observation ledger A, the componentwise* hsame *transports R, the local ledger L, the classifier-output row D, the displayed* Cont *routes C, package provenance P, and local name row N. Changing any unlicensed observer-substance datum leaves the public classifier read unchanged, because no such datum is a carrier coordinate.*

Proof. Project the accepted packet to the rows of Definition 986.1. The only consumer-facing comparison data are the licensed observation ledger, transport row, local ledger, classifier-output row, route row, provenance row, and name row.

Apply the NameCert obligations of Definition 986.2. They require every public decision to pass through the displayed licensed observations and the hsame-mediated transport surface before the classifier-output row can be read.

There is no observer-substance coordinate and no unlicensed observation coordinate in the carrier. A change confined to such data therefore changes no row in A, R, L, D, C, P, N, so every public classifier read remains the same. □

Lean4 ✓ BEDC.Derived.ObserverHsameClassifierUp.ObserverHsameClassifier decision row concrete boundary

Theorem 986.4 (Observer hsame classifier licensed-observation exactness)**.** *For an accepted* ObserverHsameClassifi *packet* $K = (H_0, H_1, A, R, L, D, C, P, N)$*, every decision-relevant mark,* Cont *route,* ProbeBundle *row,* SigRel *row, and prior classifier read is licensed exactly by the pair of displayed ledger rows* A *and* L*. No row outside* A *can affect the classifier output, and every row in* A *that affects the output has its source history row and naming boundary recorded in* L*.*

Proof. Start from the carrier inventory in Definition 986.1. The only row that lists decision-relevant observations is A, and the only row that attaches such observations to source histories and naming boundaries is L.

Apply Definition 986.2. Its licensed-observation clause says that marks, Cont routes, ProbeBundle rows, SigRel rows, and prior classifier reads may enter the decision only through the finite ledger, with no outside observation row available.

Finally use Definition 986.3. Since a public consumer can read only A, R, L, D, C, P, N, any decision-relevant observation outside A would have to escape the displayed carrier. The non-escape theorem rules this out, while the entries of L supply the required source and boundary records for the observations that remain. □

Theorem 986.5 (Observer hsame classifier transport stability)**.** *For an accepted* ObserverHsameClassifier$^{\uparrow}$ *packet* $K = (H_0, H_1, A, R, L, D, C, P, N)$*, componentwise* hsame *transport through* R *preserves the two history rows* H_0, H_1*, the licensed ledger* A*, the local ledger* L*, the decision row* D*, the route row* C*, package provenance* P*, and the local name row* N*. After transport, the classifier still compares the same two history rows through the same licensed ledger and emits only the transported decision row.*

Proof. Read R as the componentwise transport row specified by Definition 986.1. Its coordinates are aligned with the two history rows, licensed observations, classifier-output row, continuation routes, package provenance, and local name.

Apply the transport clause of Definition 986.2. It requires classifier transport through the displayed hsame rows before the public decision row can be read, so transport cannot swap in a different history pair, ledger, route surface, provenance row, or name row.

Use Definition 986.3 for confinement of the transported read. A transported consumer still has access only to A, R, L, D, C, P, N, and therefore the transported classifier output factors through the same displayed rows as the original packet. □

Theorem 986.6 (Observer hsame classifier output confinement)**.** *For an accepted* ObserverHsameClassifier$^{\uparrow}$ *packet* $K = (H_0, H_1, A, R, L, D, C, P, N)$*, every agreement, disagreement, or refusal output factors through* D, L, R, C, P, N *together with the licensed observation ledger* A*. The output cannot read observer-substance, host equality, quotient history identity, unlicensed observation, or hidden classifier state.*

Proof. The carrier of Definition 986.1 gives the output row D only after the licensed observations pass through A, L, R and the displayed consumer routes in C. The provenance and naming rows P, N record the package and local certificate boundary.

Apply Definition 986.2. Its decision-row confinement and non-escape clauses bind every public classifier read to the displayed rows and exclude observations that are not licensed by the ledger.

Now apply Definition 986.3. The carrier has no coordinate for observer-substance, host equality, quotient identity, or hidden classifier state, so an agreement, disagreement, or refusal output has no access path to those data. It must therefore factor through A, D, L, R, C, P, N. □

Theorem 986.7 (Observer hsame classifier dependency scope)**.** *For an accepted* $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *packet* $K = (H_0, H_1, A, R, L, D, C, P, N)$*, the classifier decision depends exactly on the two history rows* H_0, H_1*, the licensed observation ledger* A*, the* hsame *transport row* R*, the local ledger* L*, the decision row* D*, the displayed* Cont *routes* C*, the* Pkg *provenance row* P*, the local* $\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}$ *row* N*, and the sibling* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *and* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *surfaces named by* P*. It has no dependency coordinate for observer-substance, host equality, quotient history identity, unlicensed observation, or hidden classifier state.*

Proof. The carrier definition displays precisely the rows $H_0, H_1, A, R, L, D, C, P, N$. The provenance row P is the only place where sibling surfaces enter the packet, and it names the single-packet $\mathsf{ObserverHistIdentity}^{\uparrow}$ surface and the pairwise $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ surface as provenance for the displayed observer-history rows and licensed observations.

Use Definition 986.4 for the observation side. Every decision-relevant mark, Cont route, ProbeBundle row, SigRel row, and prior classifier read must be listed by A and tied to its source and boundary in L.

Use Definition 986.5 and Definition 986.6 for transport and output. The public decision can move only through R, C, P, N and can emit only the row D after the licensed ledger has been read.

Finally apply non-escape from Definition 986.3. Since the carrier contains no observer-substance, host-equality, quotient-identity, unlicensed-observation, or hidden-state coordinate, no accepted classifier decision can depend on such data. □

Theorem 986.8 (Observer hsame classifier TasteGate carrier scope)**.** *The cited TasteGate alignment,* Nontrivial $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *instance, and* FieldFaithful $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *instance are scoped to the displayed classifier packet* $K = (H_0, H_1, A, R, L, D, C, P, N)$*. They certify the same carrier, dependency rows, and local* $\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}$ *boundary, and they cannot be read as certifying a different observer carrier or a classifier with additional hidden coordinates.*

Proof. The Lean target is attached to the $\mathsf{ObserverHsameClassifier}^{\uparrow}$ surface whose paper carrier is Definition 986.1. The surrounding NameCert obligations identify the packet by its displayed history rows, ledger rows, transport row, decision row, routes, provenance, and local name.

The Nontrivial instance witnesses that this carrier has the displayed classifier surface rather than collapsing to an empty or anonymous carrier. The FieldFaithful instance binds the exported fields to the same rows $H_0, H_1, A, R, L, D, C, P, N$.

By Definition 986.7, there is no additional carrier coordinate through which a second observer classifier could be read. Thus the TasteGate, nontriviality, and field-faithfulness surfaces refer to the same displayed packet. □

Theorem 986.9 (Observer hsame classifier scoped closure package)**.** *The* $\mathsf{ObserverHsameClassifier}^{\uparrow}$ *chapter forms a scoped closure package for the displayed classifier packet. The package consists of the carrier definition, the local* NameCert *obligations, non-escape, licensed-observation exactness, transport stability, output confinement, dependency scope, and TasteGate carrier scope, all over the same rows* $H_0, H_1, A, R, L, D, C, P, N$*.*

Proof. The carrier and local NameCert obligations are supplied by Definition 986.1 and Definition 986.2. They identify the finite packet and the five obligations that make the classifier readable.

The behavior of accepted packets is closed by the four boundary theorems. Non-escape removes observer-substance and hidden-state reads; licensed-observation exactness fixes the admissible observation ledger; transport stability preserves the displayed rows through hsame; output

confinement forces agreement, disagreement, and refusal outputs through the displayed classifier rows.

The remaining scope facts are supplied by Definition 986.7 and Definition 986.8. Together they bind the dependencies, TasteGate surface, nontriviality surface, and field-faithfulness surface to the same packet, so the chapter closes exactly the scoped classifier package and nothing beyond it. □

The Lean `TasteGate.lean` development for $\mathsf{ObserverHsameClassifier}^{\uparrow}$ provides explicit Nontrivial ObserverHsame and $\mathsf{FieldFaithful\ ObserverHsameClassifier}^{\uparrow}$ instances for the displayed classifier packet. These instances preserve the two history rows, licensed observation ledger, hsame transport row, local ledger, classifier-output row, Cont routes, Pkg provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverHsameClassifier}^{\uparrow}$ packet is generated from two observer-history rows, a licensed observation ledger, componentwise hsame transports, a local ledger, a classifier-output row, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 986.1, Definition 986.2, Definition 986.3, Definition 986.4, Definition 986.5, Definition 986.6, Definition 986.7, Definition 986.8, and Definition 986.9 specify the finite classifier packet, its licensed-observation boundary, and its exact carrier scope.

Lean4 ✓ `BEDC.Derived.ObserverHsameClassifierUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverHsameClassifier}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ObserverHsameClassifierUp.ObserverHsameClassifierTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full observer-history identity, observer-filter completeness, hidden observer state, quotient history identity, unbounded observation, or bridge checking.

Upgrade path: Public closure requires a public consumer surface that exports the scoped classifier package without adding observer-substance, host-equality, quotient-identity, unlicensed-observation, or hidden-state reads.

Chapter 987

A Concrete Naming Certificate for $\mathsf{CellularVerbRealignment}^{\uparrow}$

The $\mathsf{CellularVerbRealignment}^{\uparrow}$ packet realizes the kernel-verb realignment direction of chapter 9391 as a finite comparison surface. It sits beside the finite orbit packet of chapter 938 and the BHist cell-row embedding packet of chapter 948. Those siblings expose orbit and embedding rows; this carrier stores the comparison row that keeps the original BHist verbs and the cellular-orbit verbs visible at the same public boundary.

Falsifiable prediction: Removing the verb-comparison row makes the packet unable to distinguish BHist Cont from cellular orbit continuation.

Independence witness: cellularorbitsubstrate, bhistcellrowembedding, cellularautomaton, kernelcategory: none is bijective because this packet stores the verb-realignment comparison row rather than only orbit, embedding, automaton, or categorical hom rows.

Definition 987.1 (Cellular verb realignment carrier)**.** *A* $\mathsf{CellularVerbRealignment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, O, M, H, E, K, L, C, P, N)$$

with the following displayed rows:

1. *B is the source* BHist *verb row, displaying the finite Mark, Hist, Ext, and Cont reads being compared.*
2. *O is the cellular orbit row, displaying the finite cell-row segment, orbit window, local update route, and accepted continuation read.*
3. *M is the verb-comparison row, pairing each* BHist *kernel verb read with its cellular-orbit classifier read.*
4. *H is the componentwise* hsame *transport row for source histories, cell rows, comparison entries, embedding data, package data, and local names.*
5. *E is the embedding and readback row, citing the* $\mathsf{BHistCellRowEmbedding}^{\uparrow}$ *surface that keeps the source* BHist *row and the cellular row in one packet.*
6. *K is the kernel-commitment row, recording that the public BEDC carrier remains the displayed* BHist *row rather than the cellular orbit itself.*
7. *L is the local obligation ledger for carrier admission, verb comparison, NameCert exposure, non-escape, and TasteGate field projection.*

8. *C records displayed* Cont *routes through the* BHist *verb row, the cellular orbit row, the comparison row, the embedding row, the kernel-commitment row, the ledger, the package row, and the local naming row.*
9. *P is the* Pkg *provenance over* BHist, CellularOrbitSubstrate$^{\uparrow}$, BHistCellRowEmbedding$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularVerbRealignment}^{\uparrow}}$ *row.*

The carrier compares verb rows only through $B, O, M, H, E, K, L, C, P, N$. It has no coordinate for replacing the BHist *kernel carrier, quotienting orbit segments, reading an unbounded cellular trace, or treating a cellular classifier as a host-level definition of Mark, Hist, Ext, or Cont.*

Lean4 ✓ `BEDC.Derived.CellularVerbRealignmentUp.CellularVerbRealignmentTasteGate single carrier alignment`

Theorem 987.2 (Cellular verb realignment NameCert obligations)**.** *For the carrier of Definition 987.1, the local* $\mathsf{NameCert}_{\mathsf{CellularVerbRealignment}^{\uparrow}}$ *surface consists exactly of carrier admission for $V = (B, O, M, H, E, K, L, C, P, N)$,* BHist *verb exposure through B, cellular orbit exposure through O, verb-comparison accountability through M, componentwise transport through H, embedding and readback accountability through E, kernel-carrier commitment through K, ledger exactness through L, displayed route replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 987.1. The packet displays the BHist verb row, cellular orbit row, comparison row, transports, embedding row, kernel-commitment row, local ledger, continuation routes, package row, and local name as its only coordinates.

The sibling packets account for the two sides of the comparison. Use chapter 938 to read the finite cellular orbit row, and use chapter 948 to keep the cellular row tied to the source BHist row.

The row M is the additional obligation surface. It records which Mark, Hist, Ext, and Cont reads are being compared, so the NameCert cannot be discharged by orbit exposure alone or by embedding exposure alone.

The rows K, L, C, P, N close the certificate. The kernel-carrier row preserves the BEDC BHist commitment, while the ledger, route, provenance, and naming rows expose no replacement carrier, quotient orbit, unbounded trace, or host definition of the verbs. □

Theorem 987.3 (Cellular verb realignment non-escape boundary)**.** *Every accepted* CellularVerbRealignment$^{\uparrow}$ *public read factors through the finite route*

$$B, O, M, H, E, K, L, C, P, N.$$

The route may compare BHist *Mark, Hist, Ext, and Cont reads with displayed cellular row, orbit-segment, row-extension, and orbit-continuation reads. It cannot replace the BEDC kernel carrier, identify orbit segments by quotient equality, import an unbounded cellular trace, or make the cellular classifier the definition of the kernel verbs.*

Proof. Start with the NameCert obligation theorem Definition 987.2. Every public read reaches the local name only after the source verb row, cellular orbit row, comparison row, transport row, embedding row, kernel-commitment row, ledger, route, and package rows have been displayed.

The comparison row M separates the four verb families. Mark and Hist comparisons read displayed carrier and orbit rows, while Ext and Cont comparisons read displayed extension and continuation routes.

The embedding row E and kernel-commitment row K keep the source carrier fixed as BHist. They permit a cellular-orbit comparison but do not turn the cellular carrier into the BEDC kernel carrier.

Since the carrier lacks coordinates for quotient orbit equality, unbounded cellular traces, host evaluators, and replacement verb definitions, the accepted public read remains the finite comparison surface stated above. □

The Lean follow-up must provide `Nontrivial CellularVerbRealignmentUp` and `FieldFaithful CellularVerbRealignmentUp` instances in `lean4/BEDC/Derived/CellularVerbRealignmentUp/TasteGate.lean` with the expected alignment surface at `BEDC.Derived.CellularVerbRealignmentUp.TasteGate.CellularVerbRe`

Closure status for $\mathsf{NameCert}_{\mathsf{CellularVerbRealignment}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularVerbRealignment}^{\uparrow}$ packet is generated from a BHist verb row, a cellular orbit row, a verb-comparison row, componentwise hsame transports, an embedding and readback row, a kernel-carrier commitment row, a local obligation ledger, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularVerbRealignment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 987.1, Definition 987.2, and Definition 987.3 bind the finite BHist and cellular-orbit verb comparison surface without replacing the kernel carrier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularVerbRealignment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a carrier switch, quotient orbit equality, unbounded cellular trace, host evaluator correctness, Lean implementation, or standard bridge.

Upgrade path: Checked closure requires Lean encoding of the carrier plus Nontrivial and FieldFaithful TasteGate instances preserving the BHist verb row, cellular orbit row, comparison row, embedding row, kernel-commitment row, ledger, route, provenance, and local naming projections.

Chapter 988

A Concrete Naming Certificate for LargeModelCorpusSupply$^{\uparrow}$

The LargeModelCorpusSupply$^{\uparrow}$ packet realizes the training-corpus supply reading of chapter 9398 as finite BEDC data. It consumes the local inscription-site discipline of chapter 610, the supply-socket boundary of chapter 604, and the large-model siblings chapter 939, chapter 967, and chapter 969. Its distinguishing role is the forward-binding corpus ledger: corpus source rows, training filters, distilled weight rows, inference-consumption rows, audit ledgers, routes, provenance, and a local NameCert row are carried together before a model output is read as corpus-supplied inscription.

Definition 988.1 (LargeModelCorpusSupply carrier)**.** *A* LargeModelCorpusSupply$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, F, W, I, A, H, K, P, N)$$

with the following displayed rows:

1. *R is the corpus-source row, naming the finite text, data, or inscription records admitted to the supply channel;*
2. *F is the training-filter row, recording selection, rejection, normalization, de-duplication, and policy gates applied before distillation;*
3. *W is the distilled weight row that records how the accepted corpus channel is carried into the finite inference substrate;*
4. *I is the inference-consumption row, recording where the distilled pattern is read by a prompt, activation orbit, output record, or audit channel;*
5. *A is the audit-ledger row, storing corpus provenance, filter evidence, refusal evidence, and public read obligations for downstream checkers;*
6. *H records componentwise* hsame *transport for corpus, filter, weight, inference-consumption, audit, route, provenance, and naming rows;*
7. *K records the displayed* Cont *routes by which a consumer reads R, then F, then W, then I, A, before reaching the local name;*
8. *P is the* Pkg *provenance over* InscriptionPoint$^{\uparrow}$, TranscendentalSupplyInscription$^{\uparrow}$, LargeModelInscriptionPoint$^{\uparrow}$, LargeModelActivationOrbit$^{\uparrow}$, LargeModelAuditChannel$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{LargeModelCorpusSupply}^{\uparrow}}$ *row.*

The carrier contains no hidden corpus selector, no global corpus-truth predicate, no host semantic understanding oracle, no model-side proof of output truth, and no route that bypasses displayed provenance.

Lean4 ✓ `BEDC.Derived.LargeModelCorpusSupplyUp.LargeModelCorpusSupplyTasteGate single carrier alignment`

Definition 988.2 (LargeModelCorpusSupply classifier)**.** *Two accepted* $\mathsf{LargeModelCorpusSupply}^{\uparrow}$ *carriers classify together exactly when their corpus-source rows, training-filter rows, distilled weight rows, inference-consumption rows, audit-ledger rows, route rows, provenance rows, and local naming rows compare componentwise by the displayed* hsame *entries:*

$$R \sim_{\mathsf{hsame}} R', \quad F \sim_{\mathsf{hsame}} F', \quad W \sim_{\mathsf{hsame}} W', \quad I \sim_{\mathsf{hsame}} I',$$
$$A \sim_{\mathsf{hsame}} A', \quad K \sim_{\mathsf{hsame}} K', \quad P \sim_{\mathsf{hsame}} P', \quad N \sim_{\mathsf{hsame}} N'.$$

Changing a source row or filter row while leaving the classifier unchanged is not an accepted comparison. No clause compares hidden author intent, unbounded training truth, private filtering policy outside F, or an internally ratified universal claim.

Theorem 988.3 (LargeModelCorpusSupply NameCert obligations)**.** *For the carrier of Definition 988.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelCorpusSupply}^{\uparrow}}$ *surface consists of the standard five finite rows:*

1. *carrier admission for $C = (R, F, W, I, A, H, K, P, N)$;*
2. *classifier transport for corpus-source, training-filter, distilled weight, inference-consumption, audit-ledger, route, provenance, and naming rows under componentwise* hsame*;*
3. *stability of the distilled weight and inference-consumption reads under the same displayed corpus-source and training-filter rows;*
4. *ledger exactness requiring every accepted inference-consumption or audit-channel read to pass through R, F, W, I, A, K, P, N in the displayed order;*
5. *non-escape for hidden corpus selection, corpus-wide truth, host semantic understanding, model-side proof of output truth, and unaudited provenance recovery.*

Lean4 ✓ `BEDC.Derived.LargeModelCorpusSupplyUp.LargeModelCorpusSupplyCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 988.1. The packet exposes exactly the corpus source, training filter, distilled weight, inference-consumption, audit, transport, route, provenance, and local naming rows.

Classifier transport is componentwise. A comparison of accepted corpus-supply packets must transport R, F, W, I, A, K, P, N together, so a changed corpus-source row or filter row exits the certificate unless the displayed classifier changes with it.

Stability and ledger exactness follow from the route order in K. A consumer first reads R and F, then reads W as the distillation row, and only then consumes I or A before reaching the provenance and naming rows.

The non-escape clauses follow from the inventory. The packet has no coordinate for hidden source selection, global corpus truth, host semantic understanding, model-side output proof, or provenance recovery outside the displayed audit ledger. □

Theorem 988.4 (LargeModelCorpusSupply corpus-provenance boundary)**.** *Every accepted* LargeModelCorpusSupp *packet exposes a corpus source row only by the displayed route*

$$\textit{corpus-source row} \longrightarrow \textit{training-filter row} \longrightarrow \textit{distilled weight row} \longrightarrow \textit{inference-consumption row} \longrightarrow \textit{audit}$$

Thus an accepted $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *read can recover a source row only after consuming the same corpus provenance coordinate that supplied the forward-binding channel.*

Proof. Project the carrier $C = (R, F, W, I, A, H, K, P, N)$. The source row R is not exposed alone; the filter row F and distilled weight row W sit on the route before any inference-consumption row can be read.

The large-model inscription and activation-orbit siblings consume the distilled pattern through I, while the audit-channel sibling consumes public provenance through A and P. These routes keep corpus provenance visible before any output is accepted as audited.

Since the carrier contains no hidden source selector or private provenance channel, a downstream audit read that reaches a corpus source row must pass through the displayed corpus provenance coordinate. □

Falsifiable prediction: If the corpus-supply packet is atomic, no accepted LargeModelAuditChannelUp read can recover a corpus source row without consuming the displayed corpus provenance coordinate.

Independence witness: large-model-inscription-point, large-model-activation-orbit, large-model-audit-channel, inscription-point, transcendental-supply-inscription: the corpus-supply carrier has corpus provenance and training-filter rows absent from each sibling, while the siblings respectively carry inscription-site, activation-orbit, audit-channel, local-inscription, or supply-socket rows instead.

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelCorpusSupply}^{\uparrow}}$.

Constructive story: A $\mathsf{LargeModelCorpusSupply}^{\uparrow}$ packet is generated from corpus-source, training-filter, distilled weight, inference-consumption, audit-ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeModelCorpusSupply}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 988.1, Definition 988.2, Definition 988.3, and Definition 988.4 specify the finite corpus-supply packet and its provenance boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeModelCorpusSupply}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close corpus truth, hidden author intent, host semantic understanding, model consciousness, model-side proof of output truth, Lean verification, or bridge checking for $\mathsf{LargeModelCorpusSupply}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, corpus-source exposure, filter exactness, provenance boundary, nontriviality, and field faithfulness for the same corpus-supply carrier.

Chapter 989

A Concrete Naming Certificate for $\mathsf{LargeModelContextAudit}^{\uparrow}$

The $\mathsf{LargeModelContextAudit}^{\uparrow}$ packet realizes the context-and-audit side of chapter 9398 as finite BEDC data. It consumes the context-window commitment of chapter 971, the prompt-response trace of chapter 965, the model-trace surface of chapter 972, the audit channel of chapter 969, and the checked-output harness of chapter 976. Its distinguishing role is a finite context-audit packet: a model output is accepted only after the displayed context, prompt-response, model trace, audit channel, verified harness, refusal, transport, continuation, provenance, and local naming rows have been read together.

Definition 989.1 (LargeModelContextAudit carrier)**.** *A* $\mathsf{LargeModelContextAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (C, T, M, A, V, R, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{ContextWindowCommitment}^{\uparrow}$ *row naming the bounded prompt context, response boundary, admitted source rows, and downstream consumer surface;*
2. *T is the* $\mathsf{PromptResponseTrace}^{\uparrow}$ *row recording the prompt, response, source-channel route, observer-interface handoff, and obstruction boundary;*
3. *M is the* $\mathsf{ModelTrace}^{\uparrow}$ *row recording the finite model-state trace, output trace, corpus-supply row, refusal row, and accepted readback;*
4. *A is the* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *row carrying prompt, output, checker, refusal, audit membrane, route, provenance, and local name;*
5. *V is the* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *row carrying the checker-facing acceptance harness for the same output record;*
6. *R is the refusal row, recording unsupported, out-of-context, unverifiable, policy-refused, or route-incomplete claims before acceptance is exposed;*
7. *H records componentwise* hsame *transport for context, prompt-response, model-trace, audit-channel, harness, refusal, route, provenance, and naming rows;*
8. *K records the displayed* Cont *route by which a consumer reads C, then T, then M, then A and V, before reaching R, P, N;*

9. *P is the* Pkg *provenance over* ContextWindowCommitment$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, ModelTrace$^{\uparrow}$, LargeModelAuditChannel$^{\uparrow}$, VerifiedOutputHarness$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{LargeModelContextAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden context recovery, unchecked model output acceptance, unrecorded checker state, host semantic understanding, corpus-wide truth, or a private route bypassing the displayed context-audit ledger.

Lean4 (definition) `BEDC.Derived.LargeModelContextAuditUp.LargeModelContextAuditUp`
Lean4 ✓ `BEDC.Derived.LargeModelContextAuditUp.LargeModelContextAuditTasteGate single carrier alignment`

Theorem 989.2 (LargeModelContextAudit NameCert obligations)**.** *For the carrier of Definition 989.1, the local* NameCert$_{\text{LargeModelContextAudit}^{\uparrow}}$ *surface consists of carrier admission for*

$$X = (C, T, M, A, V, R, H, K, P, N),$$

classifier transport for the same rows under componentwise hsame*, route exactness through the displayed* Cont *row K, refusal exactness through R, audit-channel and verified-harness exposure through A and V, provenance through P, and local naming through N. Every accepted consumer read of a model output factors through the displayed* ContextWindowCommitment$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, ModelTrace$^{\uparrow}$, LargeModelAuditChannel$^{\uparrow}$, *and* VerifiedOutputHarness$^{\uparrow}$ *rows before the output can be accepted.*

Proof. Carrier admission is projection from Definition 989.1. The packet displays context, prompt-response trace, model trace, audit channel, verified harness, refusal, transport, route, provenance, and local naming rows as its entire public surface.

Classifier transport is componentwise. A transported packet must carry $C, T, M, A, V, R, H, K, P, N$ together, so a changed context window, trace row, checker row, refusal row, or verified harness cannot be silently retained by the same local name.

Route exactness follows from K. A consumer first reads the bounded context row C, then the prompt-response row T, then the model-trace row M, and only then reaches the audit channel A and verified harness V before refusal, provenance, and naming are exposed.

The non-escape clauses follow from the carrier inventory. No row stores hidden context recovery, unchecked acceptance, unrecorded checker state, host semantic understanding, corpus-wide truth, or a private acceptance route. □

Lean4 ✓ `BEDC.Derived.LargeModelContextAuditUp.TasteGate.LargeModelContextAuditTasteGate single carrier alignment`

Theorem 989.3 (LargeModelContextAudit refusal factorization)**.** *Every refusal or accepted-output decision exposed by an accepted* LargeModelContextAudit$^{\uparrow}$ *carrier*

$$X = (C, T, M, A, V, R, H, K, P, N)$$

factors through the displayed context-window commitment, prompt-response trace, model trace, audit channel, verified output harness, componentwise transport, continuation route, provenance, and local naming rows. In particular, a consumer cannot accept a model output by reading V alone; the read must also pass through $C, T, M, A, R, H, K, P, N$.

Proof. Start with a refusal or accepted-output read. The entry row is the bounded context commitment C, because the packet has no separate coordinate for hidden context recovery or a bare output acceptance.

The prompt-response row T and model-trace row M then identify the finite emitted record being audited. These rows bind the output to its prompt, source-channel route, trace, corpus-supply handoff, and refusal boundary before any checker-facing surface is consumed.

The audit-channel row A and verified-harness row V provide the checker and acceptance surfaces. The refusal row R records unsupported or out-of-context outcomes before the local name can expose acceptance.

Finally route the read through H, K, P, N. Those rows supply transport, continuation, provenance, and local naming data, so the consumer receives a displayed context-audit packet rather than a private evaluator route or an unchecked model emission. □

Falsifiable prediction: Any accepted LargeModelContextAuditUp consumer factors through displayed ContextWindowCommitmentUp, PromptResponseTraceUp, ModelTraceUp, LargeModelAuditChannelUp, and VerifiedOutputHarnessUp rows before accepting a model output.

Independence witness: contextwindowcommitment, promptresponsetrace, modeltrace, largemodelauditchannel, verifiedoutputharness: the context-audit packet is not bijective with these siblings because it binds their rows into one consumer route and adds a shared refusal, transport, continuation, provenance, and local NameCert surface that no single sibling carries alone.

The Lean follow-up must provide Nontrivial LargeModelContextAudit$^{\uparrow}$ and FieldFaithful LargeModelContextAudit$^{\uparrow}$ instances in BEDC.Derived.LargeModelContextAuditUp/TasteGate.lean for the same finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelContextAudit}^{\uparrow}}$ **·**

Constructive story: A LargeModelContextAudit$^{\uparrow}$ packet is generated from context-window commitment, prompt-response trace, model trace, audit channel, verified output harness, refusal, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LargeModelContextAudit}^{\uparrow}}$, seed)

Scope closed: Definition 989.1, Definition 989.2, and Definition 989.3 specify the finite context-audit packet and its consumer route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LargeModelContextAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close model consciousness, corpus truth, host semantic understanding, global checker correctness, hidden context recovery, Lean verification, or bridge checking for LargeModelContextAudit$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, context-route exactness, refusal factorization, nontriviality, and field faithfulness for the same context-audit carrier.

Chapter 990

A Concrete Naming Certificate for $\mathsf{CauchyTailConsensus}^{\uparrow}$

The $\mathsf{CauchyTailConsensus}^{\uparrow}$ packet realizes the shared-tail consensus direction of chapter 9238. It ties the shared threshold route of chapter 983 to the paired agreement route of chapter 699, then exposes a single $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff for downstream completion consumers.

Definition 990.1 (Cauchy tail consensus carrier)**.** *A* $\mathsf{CauchyTailConsensus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *row that supplies one shared finite threshold and readback route;*
2. *A is the accepted* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row that compares the paired tails after the same threshold route has been displayed;*
3. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row exposed only after S and A agree on the route;*
4. *W records the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows used by both S and A;*
5. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks consumed by the agreement and limit-seal rows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for Real-completion consumers;*
7. *H records componentwise* hsame *transport for all displayed rows;*
8. *C records the* Cont *replay from shared threshold to agreement, from agreement to limit seal, and from limit seal to Real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailConsensus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, independent tail thresholds, unshared readback routes, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyTailConsensusUp.CauchyTailConsensusTasteGate single carrier alignment`

Theorem 990.2 (Cauchy tail consensus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyTailConsensus}^{\uparrow}}$ *surface for Definition 990.1 consists exactly of carrier admission, shared-tail admission through* S*, tail-agreement admission through* A*, common* $\mathsf{StreamName}^{\uparrow}$ *window service through* W*, common* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff through* L*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise transport through* H*, route replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The rows $S, A, L, W, R, E, H, C, P, N$ are the complete accepted inventory, so each shared-tail, agreement, window, readback, limit-seal, Real-seal, transport, route, provenance, and naming obligation is visible.

Read the two sibling routes. The shared-tail side uses chapter 983, while the agreement side uses chapter 699. Both are forced to consume the same displayed W, R route before L and E are available.

Since the carrier stores no quotient stream, selected limit, independent threshold, unshared readback, host equality, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 990.3 (Cauchy tail consensus shared route determinacy)**.** *Every accepted* $\mathsf{CauchyTailConsensus}^{\uparrow}$ *packet exposes a* $\mathsf{Real}^{\uparrow}$ *seal only when the shared-tail row* S *and the tail-agreement row* A *use the same finite threshold, window, and readback route through* W, R, H, C, P, N*. A packet whose* S *and* A *rows disagree on that route has no accepted Real seal row* E*.*

Proof. Start with the NameCert obligations. The $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff L and terminal seal E occur after both S and A have consumed the displayed W, R route.

Apply the sibling boundaries. The shared-tail modulus chapter supplies one shared finite threshold, and the Real tail-agreement seal chapter accepts the paired agreement only through its displayed window and readback rows.

If those routes disagree, there is no common continuation from A to L in the carrier. Thus the terminal seal row cannot be exposed; if they agree, the displayed H, C, P, N rows replay the single accepted consensus route. □

Theorem 990.4 (Cauchy tail consensus Real-completion handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{CauchyTailConsensus}^{\uparrow}$ *packet reads only the finite route*

$$S, A, W, R, L, E, H, C, P, N.$$

The route first consumes the shared-tail modulus, then the tail-agreement seal, then the Cauchy limit seal, and finally the Real seal; it exports no quotient stream, selected host limit, independent threshold, host real equality, or ambient completeness principle.

Proof. Use shared route determinacy. It makes W, R the common route for the shared-tail row and the tail-agreement row before any seal is visible.

Next read the Cauchy-limit handoff. The row L is reached only after the consensus route has passed through both S and A.

Finally project the terminal seal row E and the structural rows H, C, P, N. They transport, replay, package, and name the same finite route, so the consumer has no coordinate from which to import any excluded completion resource. □

Falsifiable prediction: Any accepted $\mathsf{CauchyTailConsensus}^{\uparrow}$ packet exposes no Real seal unless both shared-tail and tail-agreement rows use the same finite threshold and readback route.

Independence witness: cauchysharedtailmodulus, realtailagreementseal, cauchylimitseal, streamname, regseqrat: none is bijective to $\mathsf{CauchyTailConsensus}^{\uparrow}$ because this carrier stores the shared-tail row, agreement row, limit-seal handoff, common window/readback route, Real seal, and structural certificate rows together.

The Lean `TasteGate.lean` boundary for this carrier must include `Nontrivial CauchyTailConsensusUp` and `FieldFaithful CauchyTailConsensusUp` instances, so the consensus object is represented by an inhabited and field-faithful packet rather than by a vacuous route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailConsensus}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CauchyTailConsensus}^{\uparrow}$ packet is generated from shared-tail modulus, Real tail-agreement seal, CauchyLimitSeal, StreamName window, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailConsensus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 990.1, Definition 990.2, Definition 990.3, and Definition 990.4 bind the finite shared-tail consensus carrier and its Real-completion consumer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailConsensus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected host limits, host real equality, independent tail thresholds, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-tail admission, tail-agreement admission, common window and readback route determinacy, CauchyLimitSeal handoff, RealUp seal handoff, and non-escape for the same packet.

Chapter 991

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$

The $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ packet realizes the Real and Completion direction of chapter 9238 as a finite nonzero-apartness budget certificate for regular Cauchy consumers. It consumes the regular source surface of chapter 301, the dyadic lower-bound readback of chapter 291, the finite window surface of chapter 126, the Real seal boundary of chapter 38, and the reciprocal-window budget surface of chapter 703. The packet isolates the finite lower-bound row shared by reciprocal and apartness consumers before either consumer reaches a Real-facing seal.

Definition 991.1 (Regular Cauchy apartness-budget carrier)**.** *A* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source whose nonzero lower-bound window is being read;*
2. *A is the nonzero-apartness budget row, recording a finite positive lower-bound request for X;*
3. *M is the finite modulus row selecting the observation precision at which the lower-bound request is inspected;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected from X by the budget and modulus rows;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound readback row obtained from that window;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row tying the dyadic lower-bound data back to the regular source;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after A, M, W, D, R have been displayed;*
8. *H records componentwise* hsame *transport for the source, apartness budget, modulus, window, dyadic lower bound, regular readback, Real seal, provenance, and local name;*
9. *C records the displayed* Cont *routes for source inspection, apartness-budget exposure, finite-window coverage, dyadic lower-bound soundness, regular readback, and Real-seal handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessBudget}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through the displayed rows $X, A, M, W, D, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier route rows`

Theorem 991.2 (Regular Cauchy apartness-budget NameCert obligations)**.** *For the carrier of Definition 991.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessBudget}^{\uparrow}}$ *surface consists of carrier admission, classifier stability, apartness-budget exposure, dyadic lower-bound soundness, and Real-seal non-escape for the same finite packet $B = (X, A, M, W, D, R, E, H, C, P, N)$. Carrier admission exposes exactly the listed rows. Classifier stability transports those rows componentwise through H and replays the same routes through C. Apartness-budget exposure requires A before any lower-bound or seal read. Dyadic lower-bound soundness requires D to be read from the finite window W selected by M. Real-seal non-escape requires every* $\mathsf{Real}^{\uparrow}$ *consumer to reach E only after the regular readback R has been displayed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetTasteGate single carrier alignment`

Proof. Project the displayed packet. The carrier rows are exactly $X, A, M, W, D, R, E, H, C, P, N$, so admission has no hidden coordinate outside the tuple.

Classifier stability is componentwise. The transport row H compares the same source, budget, modulus, window, lower-bound, readback, seal, provenance, and name fields, while C replays the displayed route in the transported packet.

The apartness and dyadic clauses follow from the order of the finite route. The budget row A is read before the selected window and lower-bound row, and D is a read of the finite StreamName window W under the modulus row M.

The Real-seal clause then uses the regular readback row R. Since E is the only Real seal coordinate and it is reached only through R, H, C, P, N, a seal-facing consumer cannot bypass the displayed apartness-budget and dyadic lower-bound rows. □

Theorem 991.3 (Regular Cauchy apartness-budget non-escape)**.** *An accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet exports only the rows*

$$X, A, M, W, D, R, E, H, C, P, N.$$

It exports no host nonzero real, no trichotomy principle, no quotient stream, no selected representative, no total reciprocal operation, and no ambient completeness theorem. Any reciprocal-window, apartness, regular-Cauchy, dyadic, StreamName, or Real-facing consumer of the packet must read the displayed apartness-budget row before consuming the dyadic lower-bound row or the Real seal row.

Proof. Inspect the carrier definition. The packet has one regular source, one nonzero-apartness budget, one modulus row, one finite window, one dyadic lower-bound readback, one regular readback, one Real seal, and the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 991.2. They force carrier admission, componentwise classifier stability, apartness-budget exposure, dyadic lower-bound soundness, and Real-seal non-escape to use those same fields.

The excluded objects are not packet fields. A host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem would require an additional coordinate or an additional route, neither of which is present in the accepted carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudget nonescape`

Theorem 991.4 (Regular Cauchy apartness-budget transport stability)**.** *For accepted* RegularCauchyApartnessBud *packets, the lower-bound row is stable only componentwise through the displayed transport and continuation surface. If*

$$B = (X, A, M, W, D, R, E, H, C, P, N),$$

then H transports the regular source, apartness budget, modulus row, StreamName window, DyadicRatCore lower-bound row, RegSeqRat readback, Real seal, provenance, and local name componentwise, while C records the corresponding displayed route stability

$$\mathsf{Cont}(M, W), \qquad \mathsf{Cont}(W, D), \qquad \mathsf{Cont}(W, R), \qquad \mathsf{Cont}(R, E).$$

No transported consumer may detach the lower-bound row D from the apartness budget A and the finite window W.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudget transport stability`

Proof. Project the carrier rows. The only lower-bound coordinate is D, and it is placed after the apartness budget A, modulus row M, and StreamName window W.

Use the NameCert obligations. They make classifier stability componentwise through H and replay the lower-bound and readback routes through C.

The non-escape theorem then blocks any transported read that asks for a detached lower bound. Such a read would require a lower-bound coordinate outside $X, A, M, W, D, R, E, H, C, P, N$, which the accepted packet does not contain. □

Theorem 991.5 (Regular Cauchy apartness-budget lower-bound scope)**.** *Every accepted* RegularCauchyApartnessB *packet exposes its dyadic lower-bound row only through the finite route*

$$A, M, W, D, R, H, C, P, N.$$

Here A is the apartness budget, M is the modulus row, W is the $\mathsf{StreamName}^{\uparrow}$ *window, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound read, and R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. A* $\mathsf{Real}^{\uparrow}$ *or reciprocal-window consumer may therefore read D only after the displayed route has fixed those rows, and cannot read D outside the finite packet $X, A, M, W, D, R, E, H, C, P, N$.*

Proof. Project the accepted carrier. The lower-bound coordinate is the single displayed row D, and the rows immediately controlling it are the apartness budget A, modulus M, finite window W, and regular readback R.

Apply the NameCert obligations of Definition 991.2. They require apartness-budget exposure before the lower-bound read, force D to be read from the selected finite window, and keep the regular readback on the same displayed packet.

Apply transport stability, Definition 991.4. The rows H and C preserve the lower-bound route componentwise and replay only the displayed continuations. Thus any $\mathsf{Real}^{\uparrow}$ or reciprocal-window consumer that tries to read D without A, M, W, R, H, C, P, N leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier lower bound scope`

Theorem 991.6 (Regular Cauchy apartness-budget lower-bound determinacy)**.** *Let*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

and

$$B' = (X', A, M, W, D, R, E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packets. If the two packets share the displayed apartness row A, modulus row M,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the structural rows governing the lower-bound route, then they expose the same dyadic lower-bound handoff to every* $\mathsf{Real}^{\uparrow}$ *and reciprocal-window consumer. The handoff cannot be changed by a different source seal, host nonzero witness, trichotomy branch, quotient stream, selected representative, or ambient completeness principle.*

Proof. Apply lower-bound scope, Definition 991.5. It confines every lower-bound read to the displayed route $A, M, W, D, R, H, C, P, N$ and does not admit an additional lower-bound coordinate.

Use transport stability, Definition 991.4. The shared structural rows keep the apartness budget, modulus, window, dyadic lower bound, regular readback, provenance, and local naming componentwise fixed, while the displayed continuations replay the same route.

Apply the reciprocal-window boundary, Definition 991.8. A reciprocal consumer can only consume the lower-bound handoff through the same finite window and readback rows, and a $\mathsf{Real}^{\uparrow}$ consumer reaches the seal only after that handoff is displayed.

Finally use non-escape, Definition 991.3. The alternative objects listed in the statement are not fields of either accepted packet, so they cannot distinguish the handoff determined by the shared lower-bound route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier lower bound determinacy`

↪ variant `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetLowerBoundDeterminacy`

Theorem 991.7 (Regular Cauchy apartness-budget Real seal refusal)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal or reciprocal-window read is public from a* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet unless the apartness-budget row A has already been read. A route that exposes E or a reciprocal lower-bound consumer while omitting A is not a carrier read of the accepted packet.*

Proof. Read the order of the displayed packet. The apartness-budget row A is the only public nonzero-budget coordinate, and the lower-bound and seal rows occur downstream of the finite window route.

Apply transport stability for the lower-bound row. It keeps D attached to A, M, W under H and C, so a reciprocal-window consumer cannot read D as an independent public datum.

Finally apply non-escape. The Real seal row E is exposed only through R, H, C, P, N after the same apartness and lower-bound route, so a consumer omitting A leaves the public surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier real seal refusal`

Theorem 991.8 (Regular Cauchy apartness-budget reciprocal-window boundary)**.** *Every reciprocal-window consumer of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet factors through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the local* $\mathsf{NameCert}$ *rows*

$$A, M, W, D, R, H, C, P, N.$$

The boundary exposes a finite apartness budget and finite readback route, not a total reciprocal operation, host nonzero real, quotient stream, selected representative, or ambient completeness theorem.

Proof. Start from the NameCert obligations. They require apartness-budget exposure before the dyadic lower-bound row and require that lower bound to be read from the finite StreamName window selected by the modulus row.

Use transport stability to keep the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows on the same displayed route under H and C.

Apply the Real seal refusal and non-escape theorems. A reciprocal-window consumer that asks for a total reciprocal or host-level nonzero fact asks for data not carried by $A, M, W, D, R, H, C, P, N$, so the public boundary is exactly the finite route stated above. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier reciprocal window boundary`

Theorem 991.9 (Regular Cauchy apartness-budget Phase Real handoff)**.** *Every Phase Real and Completion consumer that uses a* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *lower-bound read must factor through the displayed RegSeqRat source* X*, apartness budget* A*, modulus row* M*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, Real seal row* E*, and the structural rows* H, C, P, N *before it can feed a reciprocal-window or* $\mathsf{Real}^{\uparrow}$*-facing exit. The handoff is the finite route*

$$X, A, M, W, D, R, E, H, C, P, N.$$

It refuses host nonzero reals, trichotomy, quotient streams, selected representatives, a total reciprocal operation, and ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier phase real handoff`

Proof. Project the carrier of Definition 991.1. The rows $X, A, M, W, D, R, E, H, C, P, N$ are the only displayed source, lower-bound, seal, transport, replay, provenance, and local naming rows available to the consumer.

Apply lower-bound scope, Definition 991.5. It forces every lower-bound read through A, M, W, D, R and the local structural rows before any downstream consumer can use the read.

Apply lower-bound determinacy, Definition 991.6. Once the displayed lower-bound route is fixed, there is no second carrier coordinate from which a Phase Real or Completion consumer can read a different positive handoff.

Use Real seal refusal, Definition 991.7, and the reciprocal-window boundary, Definition 991.8. A Real-facing or reciprocal-window exit is therefore reached only after the same apartness budget, modulus, window, dyadic lower bound, regular readback, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The carrier exclusions finish the non-escape claim. Host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocals, and ambient completeness are absent from the packet, so none of them can be exported by the Phase Real and Completion handoff. □

Theorem 991.10 (Regular Cauchy apartness-budget public interface)**.** *Every public reciprocal-window or* $\mathsf{Real}^{\uparrow}$*-seal consumer of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

factors through the same finite interface

$$A, M, W, D, R, E, H, C, P, N.$$

This interface composes lower-bound scope, lower-bound determinacy, Real-seal refusal, and reciprocal-window boundary. It exports no host nonzero real, trichotomy principle, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier public interface`

Proof. First apply lower-bound scope, Definition 991.5. It confines every lower-bound read to the apartness row, modulus row, finite window, dyadic lower-bound row, regular readback, transport, replay, provenance, and local naming rows.

Next apply lower-bound determinacy, Definition 991.6. Once A, M, W, D, R and the structural rows are fixed, a public consumer has no second carrier coordinate from which to read a different dyadic lower-bound handoff.

Use Real-seal refusal, Definition 991.7, before any $\mathsf{Real}^{\uparrow}$-seal consumer reaches E. The seal is public only after the same apartness and lower-bound route has been displayed.

Finally apply the reciprocal-window boundary, Definition 991.8. A reciprocal consumer reads exactly the finite route $A, M, W, D, R, H, C, P, N$, and a Real-seal consumer adds only the downstream seal row E on that route. Since non-escape excludes host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocals, and ambient completeness, the listed interface is the full public surface. □

Theorem 991.11 (Regular Cauchy apartness-budget positive-window determinacy)**.** *Let* $B = (X, A, M, W, D, R, E, H, C, P, N)$ *and* $B' = (X', A, M, W, D, R, E', H', C', P', N')$ *be accepted* RegularCauchyApartne *packets with the same displayed apartness row* A*, modulus row* M*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and structural rows for the lower-bound route. Then every reciprocal-window consumer reads the same positive lower-bound handoff from* B *and* B'*. No host nonzero real, trichotomy branch, quotient stream, or selected representative can alter that read.*

Proof. Project both accepted carriers. The lower-bound route in each packet is displayed through the same rows A, M, W, D, R together with the structural transport, replay, provenance, and naming rows for that route.

Apply lower-bound scope, Definition 991.5. It makes the dyadic lower-bound read depend only on the apartness budget, modulus, StreamName window, DyadicRatCore row, regular readback, and local structural rows.

Use transport stability, Definition 991.4, to keep this route componentwise fixed. Then apply the reciprocal-window boundary theorem Definition 991.8: a reciprocal consumer has no remaining coordinate from which to read a different positive lower bound.

The non-escape and Real-seal refusal clauses remove host nonzero reals, trichotomy branches, quotient streams, and selected representatives from the packet. Hence those external objects cannot change the finite lower-bound handoff determined by the shared window rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier positive window determinacy`

Theorem 991.12 (Regular Cauchy apartness-budget lower-bound modulus monotonicity)**.** *Let*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

and

$$B' = (X', A, M', W', D, R, E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packets sharing the apartness row* A*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row* D*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. If the modulus and window pair* (M', W') *refines* (M, W) *through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* Cont *route, then every* $\mathsf{Real}^{\uparrow}$ *or reciprocal-window consumer reads from* B' *a lower-bound handoff no weaker than the one read from* B*. The refinement may add displayed window information, but it cannot replace the apartness row, weaken the dyadic lower-bound row, detach the regular readback, or import a host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.*

Proof. Project the carrier rows of both packets. The shared lower-bound data are the apartness row A, the dyadic lower-bound row D, and the regular readback R, while the only changing rows in the lower-bound route are the modulus and window pair.

Apply lower-bound scope, Definition 991.5. It binds every lower-bound read to the finite route $A, M, W, D, R, H, C, P, N$ in the coarser packet and to the analogous route $A, M', W', D, R, H', C', P', N'$ in the refined packet.

Use transport stability to replay the refinement through H, H' and C, C'. The displayed $\mathsf{StreamName}^{\uparrow}$ and Cont route preserves the apartness row, dyadic lower-bound row, and regular readback, so a consumer can only receive the same lower-bound handoff or one exposed through a finer finite window.

Apply positive-window determinacy, Definition 991.11. With A, D, R fixed and the refined window routed through the same displayed lower-bound data, there is no carrier coordinate from which a changed or weaker handoff can be read.

Finally use non-escape. Host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocals, and ambient completeness are not packet fields, so they cannot reverse the monotone refinement of the lower-bound route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier lower bound modulus monotonicity`

Theorem 991.13 (Regular Cauchy apartness-budget L10 scope binding)**.** *Every accepted apartness-budget read exported by a* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet factors through chapter 301, chapter 291, chapter 126, chapter 38, and chapter 703 before any Real-facing apartness consumer may use the budget. In carrier terms, the source row* X*, apartness row* A*, modulus row* M*, window row* W*, dyadic lower-bound row* D*, regular readback row* R*, Real seal row* E*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *must all remain on the same finite L10 route. No consumer may detach the budget from this dependency lattice into a host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.*

Proof. Project the carrier of Definition 991.1. The accepted packet places the regular source, apartness budget, modulus, finite window, dyadic lower bound, regular readback, Real seal, transport, replay, provenance, and local naming rows in one tuple.

Apply the NameCert obligations and lower-bound scope. These force the apartness row to precede the StreamName window and DyadicRatCore lower-bound read, and they keep the RegSeqRat readback on the same route before any Real seal is consumed.

Use the reciprocal-window boundary and positive-window determinacy. A Cauchy inverse or reciprocal-window consumer reads the same positive lower-bound handoff only through the displayed finite window and readback rows.

Finally apply non-escape and Real-seal refusal. The excluded host objects are not carrier coordinates, so every Real-facing apartness consumer remains bound to the stated L10 dependency lattice. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier l10 scope binding`

Theorem 991.14 (Regular Cauchy apartness-budget bridge scope)**.** *Every bridge-facing consumer from* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *to* $\mathsf{Real}^{\uparrow}$ *or reciprocal-window interfaces must read the same finite route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *before using the apartness budget. In carrier terms, the rows* $A, M, W, D, R, E, H, C, P, N$ *remain attached to the source row* X *throughout the bridge read, so the budget cannot be consumed as a host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.*

Proof. First apply lower-bound scope and lower-bound determinacy, Definition 991.5 and Definition 991.6. These results keep the apartness row, modulus row, StreamName window, DyadicRatCore lower-bound row, RegSeqRat readback, transport, replay, provenance, and local naming rows on one finite route.

Next use Real-seal refusal and the reciprocal-window boundary, Definition 991.7 and Definition 991.8. A Real-facing or reciprocal-window consumer reaches the seal or reciprocal interface only after the apartness budget and lower-bound handoff have been displayed.

Finally compose the public interface with L10 scope binding, Definition 991.10 and Definition 991.13. The bridge-facing route is therefore the same finite dependency lattice through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyInverseBudget}^{\uparrow}$, with no coordinate for the refused host or completion principles. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier bridge scope`

Theorem 991.15 (Regular Cauchy apartness-budget RealApartnessField handoff)**.** *Every* $\mathsf{RealApartnessField}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet must read the apartness budget row* A*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *before any field operation consumes apartness. Thus the field-facing route cited in chapter 5562 receives a displayed finite lower-bound handoff, not a host nonzero real, trichotomy branch, quotient representative, selected limit, total reciprocal operation, or ambient completeness theorem.*

Proof. Start with the public interface, Definition 991.10. It identifies every reciprocal-window or $\mathsf{Real}^{\uparrow}$-seal consumer with the finite surface $A, M, W, D, R, E, H, C, P, N$, so a field-facing apartness consumer cannot enter through a row outside the accepted packet.

Apply bridge scope and lower-bound determinacy, Definition 991.14 and Definition 991.6. These keep the apartness budget, dyadic lower-bound ledger, regular readback, transport, replay, provenance, and naming rows on one route before the downstream Real-facing read.

Finally use non-escape, Definition 991.3. The refused host and completion principles are not carrier coordinates, while the $\mathsf{RealApartnessField}^{\uparrow}$ chapter consumes apartness through displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, transport, replay, provenance, and local naming rows. Hence every field operation that consumes apartness receives the stated handoff first. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier field handoff`

Theorem 991.16 (Regular Cauchy apartness-budget route composition)**.** *Every accepted* RegularCauchyApartnessB *packet composes the apartness row A, modulus row M,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound readback D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal row E, reciprocal-window boundary, transport row H, replay row C, provenance row P, and local* NameCert *row N through one finite L10 route. Any bridge-facing consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *apartness handoff or a reciprocal-window read must traverse that route in order; it receives no host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.*

Proof. Start with lower-bound scope, Definition 991.5. It places the apartness row, modulus row, StreamName window, DyadicRatCore lower-bound row, RegSeqRat readback, transport, replay, provenance, and local naming rows on the same finite lower-bound route.

Apply the reciprocal-window boundary, Definition 991.8. A reciprocal consumer can read a positive lower-bound handoff only after the displayed window and readback rows have been presented on that route.

Use the public interface and L10 scope binding, Definition 991.10 and Definition 991.13. These results attach the route to the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyInverseBudget}^{\uparrow}$ siblings before any bridge-facing read.

Finally apply bridge scope, Definition 991.14. The bridge-facing consumer therefore receives the composed finite route $A, M, W, D, R, E, H, C, P, N$ and no coordinate for a host nonzero real or total reciprocal. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier route composition`

Theorem 991.17 (Regular Cauchy apartness-budget cofinal window refinement)**.** *Let*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

and

$$B' = (X', A, M', W', D, R, E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packets sharing the apartness row A,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound row D, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. Suppose the* $\mathsf{StreamName}^{\uparrow}$ *windows W and W' are cofinally comparable through displayed modulus, window, transport, replay, provenance, and local-name rows. Then every* $\mathsf{Real}^{\uparrow}$ *or reciprocal-window consumer reads the same Real-facing apartness-budget handoff from the two packets. Refinement is allowed only through the displayed rows M, W, H, C, P, N and M', W', H', C', P', N'; it cannot replace A, weaken D, detach R, import a host nonzero real, or introduce a total reciprocal operation.*

Proof. First apply lower-bound scope, Definition 991.5. It confines each lower-bound read to the apartness row, modulus row, finite StreamName window, dyadic lower-bound row, regular readback, transport, replay, provenance, and local naming rows.

Next apply positive-window determinacy, Definition 991.11. Cofinal comparability makes the two window routes meet along the displayed finite-window surface, so a reciprocal-window consumer has no second carrier coordinate from which to read a different positive handoff.

Use lower-bound modulus monotonicity, Definition 991.12, in both cofinal directions. Each refinement may expose more finite window information, but with A, D, R fixed it cannot weaken or replace the lower-bound handoff.

Finally apply the reciprocal-window boundary, Definition 991.8, Real-seal refusal, Definition 991.7, and L10 scope binding, Definition 991.13. These results keep every reciprocal-window or Real-facing consumer on the displayed finite route and exclude host nonzero witnesses, quotient streams, selected representatives, total reciprocals, and ambient completeness from the handoff. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier cofinal window refinement`

Theorem 991.18 (Regular Cauchy apartness-budget bridge surface)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing apartness read of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet*

$$B = (X, A, M, W, D, R, E, H, C, P, N)$$

factors through the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessBudget}^{\uparrow}}$ *row* N*. The bridge-facing surface contains no quotient stream equality, trichotomy branch, host nonzero real, selected representative, total reciprocal operation, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier bridge surface`

Proof. Project the carrier rows. The apartness row A and modulus row M do not reach a $\mathsf{Real}^{\uparrow}$ consumer directly; they are read through the finite window W, the dyadic lower-bound row D, and the regular readback R before the seal row E is available.

Apply lower-bound scope and Real-seal refusal, Definition 991.5 and Definition 991.7. These results force every Real-facing read to expose A, M, W, D, R, E on the displayed route.

Next use bridge scope and route composition, Definition 991.14 and Definition 991.16. They attach the same route to the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and reciprocal-window sibling surfaces before a bridge consumer can use the budget.

Finally apply the public interface and L10 scope binding, Definition 991.10 and Definition 991.13. The rows H, C, P, N transport, replay, source, and name the bridge surface, while the refused host coordinates are absent from the carrier and cannot be recovered by a $\mathsf{Real}^{\uparrow}$-facing read. □

Theorem 991.19 (Regular Cauchy apartness-budget inverse-window consumption)**.** *Any accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet consumed by a* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read must expose its positive lower-bound row, reciprocal-window row, cofinal-window row, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row before the field inverse branch is read. The consumer route cites chapter 5562 and chapter 291; it receives the finite rows*

$$A, M, W, D, R, E, H, C, P, N$$

from the carrier, not Real completeness, trichotomy, quotient equality, a selected representative, or a total inverse operation.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier inverse window consumption`

Proof. First use lower-bound determinacy, Definition 991.6. It fixes the positive lower-bound handoff once the apartness row, modulus row, StreamName window, DyadicRatCore lower-bound row, RegSeqRat readback, and structural rows are displayed.

Next apply the reciprocal-window boundary, Definition 991.8. A reciprocal or inverse-facing consumer can read the lower-bound handoff only through the finite window, readback, tolerance, transport, replay, provenance, and local naming rows.

Use the RealApartnessField handoff and route composition, Definition 991.15 and Definition 991.16. These attach the lower-bound and reciprocal-window route to the field-facing apartness consumer before a field operation may consume it.

Finally apply cofinal window refinement, Definition 991.17. Cofinal refinements may display more window information, but they cannot replace A, weaken D, detach R, or import a host inverse. Hence the inverse branch reads only the displayed finite budget and dyadic tolerance rows before entering the $\mathsf{RealApartnessField}^{\uparrow}$ surface. □

Theorem 991.20 (Regular Cauchy apartness-budget shared field inverse window)**.** *Let an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet be consumed both by a* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read and by a* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *reciprocal-window read. If the two consumers cite the same accepted packet, then both reads share the displayed rows*

$$A, M, W, D, R, E, H, C, P, N.$$

The field inverse branch cannot split from the reciprocal-window lower-bound handoff: it must read the same apartness budget A, modulus row M, finite window W, dyadic lower-bound ledger D, regular readback R, Real seal E, transport H, replay C, provenance P, and local naming row N before either inverse-facing surface is exported.

Proof. Begin with the field handoff and reciprocal-window boundary, Definition 991.15 and Definition 991.8. The field consumer and reciprocal-window consumer both reach inverse-facing data only after the finite lower-bound route has displayed the apartness, window, dyadic, readback, transport, replay, provenance, and naming rows.

Apply route composition and cofinal window refinement, Definition 991.16 and Definition 991.17. Refining the window may expose more observations, but it does not replace A, split M from W, weaken D, detach R, or introduce a second Real seal.

Finally use inverse-window consumption and non-escape, Definition 991.19 and Definition 991.3. Since host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocals, and ambient completeness are not coordinates of the carrier, the two inverse-facing consumers can only share the displayed packet route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier shared field inverse window`

Theorem 991.21 (Regular Cauchy apartness-budget located reciprocal handoff)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{RealInverse}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet must read the same finite rows*

$$A, M, W, D, R, E, H, C, P, N.$$

The dyadic lower-bound row D and the regular readback row R are visible before the located reciprocal handoff is exported, and the consumer receives the reciprocal route only as a displayed apartness-budget packet over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyInverseBudget}^{\uparrow}$. *The handoff contains no host nonzero real, trichotomy branch, quotient stream, selected representative, total reciprocal operation, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier located reciprocal handoff`

Proof. Start with lower-bound determinacy, Definition 991.6. It fixes the positive lower-bound read once the apartness row, modulus row, finite window, dyadic lower-bound row, regular readback, and structural rows are displayed.

Apply the reciprocal-window boundary and field handoff, Definition 991.8 and Definition 991.15. These force any reciprocal-facing or apartness-field read to pass through $A, M, W, D, R, E, H, C, P, N$ before it can be consumed by a located or inverse-facing surface.

Use route composition and shared field inverse-window consumption, Definition 991.16 and Definition 991.20. The located consumer and the $\mathsf{RealInverse}^{\uparrow}$ consumer therefore share the same dyadic lower-bound row D and regular readback row R before either handoff is public.

Finally apply the mature consumer theorem, Definition 991.22. Since host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocals, and ambient completeness are absent from the accepted carrier, the located reciprocal route can only export the displayed finite packet. □

Theorem 991.22 (Regular Cauchy apartness-budget mature consumer)**.** *Every mature* $\mathsf{RealInverse}^{\uparrow}$, $\mathsf{RealApartnessField}^{\uparrow}$, *or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *packet must read the dyadic lower-bound row, the apartness guard, the inverse-window row, the regular rational readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before using the reciprocal budget. Such a consumer receives the displayed packet route*

$$A, M, W, D, R, E, H, C, P, N$$

and no host nonzero real, trichotomy branch, quotient stream equality, selected representative, total reciprocal operation, ambient completeness theorem, or detached inverse credential.

Proof. Start with route composition, Definition 991.16. It forces every reciprocal-budget read to pass through the apartness guard A, modulus row M, finite window W, dyadic lower-bound ledger D, regular readback R, Real seal E, and the structural rows H, C, P, N.

Next apply inverse-window consumption, Definition 991.19. A $\mathsf{RealApartnessField}^{\uparrow}$ or $\mathsf{RealInverse}^{\uparrow}$ branch may use the reciprocal budget only after the positive lower-bound row, reciprocal-window row, cofinal-window row, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row have been exposed on the same finite route.

Finally use shared field inverse-window consumption, Definition 991.20. If a $\mathsf{RegSeqRat}^{\uparrow}$ readback, field inverse read, or reciprocal-window read cites this packet, then all such consumers share $A, M, W, D, R, E, H, C, P, N$. Since the refused host coordinates are absent from the carrier, the mature consumer can only use the displayed reciprocal budget. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier mature consumer`

Theorem 991.23 (Regular Cauchy apartness-budget formal-axis target)**.** *The formal target for* RegularCauchyApartnessBudget$^{\uparrow}$ *is the determinacy of the apartness-budget packet from its displayed rows:* DyadicRatCore$^{\uparrow}$ *tolerance and lower-bound rows,* RegSeqRat$^{\uparrow}$ *tail windows,* StreamName$^{\uparrow}$ *finite observations, the* Real$^{\uparrow}$ *apartness seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. Any future formalization of the packet must recover the same rows*

$$A, M, W, D, R, E, H, C, P, N$$

before exporting reciprocal-budget, inverse-facing, or apartness-field consumers, and it may not use host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocal operations, or ambient completeness as hidden coordinates.

Proof. Read the mature consumer theorem together with route composition. Those results already force inverse-facing consumers to pass through the apartness row, modulus row, finite window, dyadic lower-bound ledger, regular readback, Real seal, and structural rows.

The formal-axis target is the same finite route stated as a later Lean obligation. Dyadic tolerance rows determine the positive lower-bound evidence, RegSeqRat tail windows and StreamName finite observations determine the visible readback route, and the Real apartness seal records the terminal nonzero-facing boundary.

Finally the hsame, Cont, Pkg, and NameCert rows bind transport, replay, provenance, and local naming to the same carrier. Since none of the refused host objects is a coordinate of the packet, a formal proof can only export the displayed row determinacy. □

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessBudgetUp.RegularCauchyApartnessBudgetCarrier formal axis target`

Falsifiable prediction: Any accepted reciprocal-window consumer that omits the displayed apartness-budget row is not a RegularCauchyApartnessBudgetUp carrier read.

Independence witness: Nearest siblings: cauchyinversebudget, apartnessreal, regseqrat, dyadicratcore, streamname. The carrier is not bijective to CauchyInverseBudgetUp because it has no reciprocal readback row; not to ApartnessRealUp because it keeps finite regular-Cauchy window budgets rather than a completed Real apartness predicate; not to RegSeqRatUp because it adds nonzero lower-bound rows; not to DyadicRatCoreUp because it includes stream-window and Real-seal routes; not to StreamNameUp because it includes dyadic lower-bound and apartness ledger rows.

The Lean TasteGate route is expected to provide `Nontrivial RegularCauchyApartnessBudgetUp` with an explicit witness pair and `FieldFaithful RegularCauchyApartnessBudgetUp` with `.mk`-derived fields and a match/injection proof.

Closure status for NameCert$_{\text{RegularCauchyApartnessBudget}^{\uparrow}}$.

Constructive story: A RegularCauchyApartnessBudget$^{\uparrow}$ packet is generated from a regular-Cauchy source, a finite nonzero-apartness budget, a modulus row, a finite StreamName window, a DyadicRatCore lower-bound readback, a RegSeqRat readback, a RealUp seal, componentwise transports, displayed routes, provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyApartnessBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 991.10 exports the public reciprocal-window and Real$^{\uparrow}$-seal surface, Definition 991.13 gives the exact L10 dependency scope over RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, CauchyInverseBudget$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, Definition 991.18 records the bridge route, and Definition 991.22 binds mature inverse, apartness-field, RegSeqRat, dyadic lower-bound, StreamName, and Real-seal consumers to the same finite reciprocal-budget route.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyApartnessBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, host nonzero reals, trichotomy, quotient streams, selected representatives, total reciprocal operations, ambient completeness, or bridge checking.

Upgrade path: Lean-side carrier admission, classifier stability, apartness-budget exposure, dyadic lower-bound soundness, Real-seal non-escape, provenance, and TasteGate alignment remain separate from the bridge-facing paper surface.

Chapter 992

A Concrete Naming Certificate for $\mathsf{AxiomRequirementLedger}^{\uparrow}$

The $\mathsf{AxiomRequirementLedger}^{\uparrow}$ packet realizes the Required-Axiom-Set coordinate of chapter 9455 as a finite consumer-facing ledger. It sits beside the dependency-map packet of chapter 862, the audit boundary of chapter 526, the forbidden-host ancestry boundary of chapter 622, the apophatic refusal surface of chapter 543, and the common audit-map interface of chapter 806. Those siblings expose mode rows, audit refusals, forbidden dependency ancestry, named externality refusal, and cross-map reads; this chapter packages the finite row by which a required axiom supply is carried without being confused with a zero-axiom witness. As an ai packet, it bundles those dependency-map, audit-boundary, forbidden-ancestry, apophatic-name, and audit-interface surfaces into one required-supply ledger.

Definition 992.1 (AxiomRequirementLedger carrier)**.** *An* $\mathsf{AxiomRequirementLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\Phi, M, W, A, R, T, U, V, H, C, P, N)$$

with the following displayed rows:

1. Φ *is the claim row whose dependency tuple is being read;*
2. M *is the Mode A, Mode B, or Mode C row inherited from the dependency-map entry;*
3. W *is the witness row, either a zero-axiom proof, a restricted proof, a counterexample, or a named socket;*
4. A *is the finite required-axiom-set row displayed for the entry;*
5. R *is the refusal ledger that prevents a nonempty* A *row from being read as substrate-internal proof content;*
6. T *is the transport row connecting the required-supply row to audit-map consumers;*
7. U *is the route row from the dependency tuple to the audit boundary and forbidden-ancestry checks;*
8. V *is the provenance row for apophatic names, audit-map interface reads, and dependency-map siblings;*
9. H *records componentwise* hsame *transport for the claim, mode, witness, required-supply, refusal, transport, route, provenance, and local name;*

10. *C records displayed* Cont *routes by which consumers inspect* $\Phi, M, W, A, R, T, U, V, H$*;*
11. *P is the* Pkg *provenance over* AxiomDependencyMap$^{\uparrow}$, AuditGateBoundary$^{\uparrow}$, ForbiddenAxiomAncestry$^{\uparrow}$, ApophaticName$^{\uparrow}$, AuditMapInterface$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AxiomRequirementLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate that turns a nonempty required-axiom row into a Mode A proof, imports a refused host axiom as BEDC theorem content, hides a selected witness, or bypasses the displayed audit route.

Lean4 ✓ `BEDC.Derived.AxiomRequirementLedgerUp.AxiomRequirementLedgerTasteGate single carrier alignment`

Theorem 992.2 (AxiomRequirementLedger NameCert obligations)**.** *For the carrier of Definition 992.1, the local* $\mathsf{NameCert}_{\mathsf{AxiomRequirementLedger}^{\uparrow}}$ *surface consists exactly of claim exposure through Φ, mode exposure through M, witness exposure through W, required-supply exposure through A, refusal preservation through R, transport through T, H, route replay through U, C, provenance through V, P, and local naming through N. A downstream audit-map consumer can read a nonempty required-axiom-set row only as required supply, refusal data, or named socket data, never as an accepted zero-axiom witness.*

Proof. Project the carrier. The accepted packet exposes the rows $\Phi, M, W, A, R, T, U, V, H, C, P, N$ and no other supply row.

The dependency-map sibling supplies the tuple order: claim, mode, witness, and required-axiom set are read before the ledger is exported. The audit boundary and forbidden-ancestry siblings then read the same required-supply row through R, T, U, H, C as a finite refusal or ancestry boundary.

The apophatic-name and audit-map-interface siblings supply the public consumer shape. They allow a named socket or refusal row to be cited, routed, and transported, but they do not convert it into a generated BEDC witness.

Thus the local NameCert surface is exactly the displayed ledger. Since no carrier coordinate imports the required axiom or manufactures a Mode A proof, a nonempty A row remains required supply rather than substrate-internal proof content. □

Lean4 ✓ `BEDC.Derived.AxiomRequirementLedgerUp.AxiomRequirementLedgerNameCertObligations`

Theorem 992.3 (AxiomRequirementLedger non-escape)**.** *Every accepted* AxiomRequirementLedger$^{\uparrow}$ *packet separates required axiom supply from zero-axiom witnesses. If the required-supply row A is nonempty, any consumer that reads the packet must factor through*

$$\Phi, M, W, A, R, T, U, V, H, C, P, N.$$

It cannot export the entry as a Mode A witness unless the witness row W is itself a checked zero-axiom proof row and the required-supply row is empty on the displayed route.

Proof. Start from the carrier definition. The only rows available to a consumer are the displayed finite rows, and the required-supply row A is separate from the witness row W.

Use the NameCert obligations of Definition 992.2. They require every consumer read of A to pass through the refusal, transport, route, provenance, and local naming rows R, T, U, V, H, C, P, N.

Now compare with the dependency-map mode row. A Mode A read is permitted only where the witness row is a checked zero-axiom proof and the required-supply row is empty on the displayed route. When A is nonempty, the accepted packet records supply still required; it does not export a substrate theorem. □

Lean4 ✓ `BEDC.Derived.AxiomRequirementLedgerUp.AxiomRequirementLedgerRequiredSupplyNonescape`

Falsifiable prediction: A consumer that turns a nonempty required-axiom row into a Mode A witness without a checked proof row is not reading an accepted AxiomRequirementLedgerUp packet.

Independence witness: axiomdependencymap, auditgateboundary, forbidden-axiom-ancestry, apophaticname, and auditmapinterface are non-bijective siblings: none alone carries the complete finite combination of claim, mode, witness, required-supply, refusal, transport, route, provenance, and NameCert data exposed by $\mathsf{AxiomRequirementLedger}^{\uparrow}$.

The packet composes axiomdependencymap, auditgateboundary, forbidden-axiom-ancestry, apophaticname, and auditmapinterface into the displayed required-supply ledger surface.

The Lean follow-up is `lean4/BEDC/Derived/AxiomRequirementLedgerUp/TasteGate.lean` with explicit Nontrivial and FieldFaithful instances for the displayed ledger.

Theorem 992.4 (AxiomRequirementLedger bridge route)**.** *Every audit-map or dependency-map bridge consumer of an accepted* $\mathsf{AxiomRequirementLedger}^{\uparrow}$ *packet must pass through the displayed rows*

$$\Phi, M, W, A, R, T, U, V, H, C, P, N$$

and the TasteGate single-carrier row before reading required axiom supply. In particular, bridge-side reads of A are mediated by the same claim, mode, witness, refusal, transport, route, provenance, replay, package, and local NameCert rows that define the public packet.

Proof. Project the accepted carrier. The only required-supply row exposed by the packet is A, and it is displayed together with $\Phi, M, W, R, T, U, V, H, C, P, N$.

An audit-map consumer reads required supply through the refusal, transport, route, and provenance rows. A dependency-map consumer reads the same supply through the claim, mode, witness, and required-set coordinates. Both routes are transported and replayed by H and C, then named by P and N.

The TasteGate single-carrier row aligns those two consumer views with the same accepted packet. Thus no bridge consumer can read axiom supply before passing through the displayed rows and the single-carrier alignment. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomRequirementLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomRequirementLedger}^{\uparrow}$ packet is generated from a claim row, mode row, witness row, required-supply row, refusal ledger, transport row, audit route, provenance row, componentwise transports, displayed consumer routes, package provenance, and local naming.

Lean4 ✓ `BEDC.Derived.AxiomRequirementLedgerUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxiomRequirementLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 992.1, Definition 992.2, Definition 992.3, the falsifiable prediction, and the independence witness export the required-supply ledger over the displayed rows $\Phi, M, W, A, R, T, U, V, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxiomRequirementLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, global theorem dependency completeness, host axiom import, selected witnesses, quotient equality, or any conversion of required supply into BEDC theorem content.

Upgrade path: Only Lean verification and bridge checking remain for the displayed required-supply ledger rows and their non-escape route.

Chapter 993

A Concrete Naming Certificate for $\mathsf{FiniteReflectionTuple}^{\uparrow}$

The $\mathsf{FiniteReflectionTuple}^{\uparrow}$ packet realizes the TasteGate stability discipline of chapter 9459 as a finite reflection tuple. It consumes the stability packet of chapter 564, the candidate-admission surface of chapter 545, and the compiler-witness route of chapter 563. The packet is narrower than these siblings: it records the fixed displayed tuple image, the readback rows, the componentwise NameCert transports, and the reflection-request rows that a downstream bridge request may inspect.

Definition 993.1 (FiniteReflectionTuple carrier)**.** *A* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N)$$

with the following displayed rows:

1. *X is the candidate chapter identity row, recording the chapter label, origin row, and concrete-instance site being reflected;*
2. *S is the consumed* $\mathsf{TasteGateStability}^{\uparrow}$ *row, carrying the candidate carrier,* ChapterTasteGate *obligations, admission ledger, and non-escape boundary;*
3. *A is the consumed* $\mathsf{TasteGateAdmission}^{\uparrow}$ *row, exposing the candidate-admission surface and refusal ledger for the same X;*
4. *W is the consumed* $\mathsf{TasteGateCompilerWitness}^{\uparrow}$ *row, exposing the ground-compiler event-flow image, replay rows, dependency ledger, and audit result;*
5. *I is the displayed tuple-image row, listing the finite image of X, S, A, W as the exact tuple requested by the reflection route;*
6. *R is the readback row, recovering X, S, A, W componentwise from I without importing a second carrier or host semantic judgment;*
7. *H stores componentwise* hsame *transports for $X, S, A, W, I, R, C, P, Q, N$ and componentwise* NameCert *transport for the three consumed TasteGate packets;*
8. *C records the displayed* Cont *routes that replay tuple formation, image reading, readback, and componentwise NameCert transport;*
9. *P is the* Pkg *provenance over chapter 9459, the three consumed TasteGate packets, tuple-image rows, readback rows,* hsame, Cont, *and* NameCert *rows;*

10. *Q is the reflection-request row, recording which* StandardBridge*-facing request may inspect the tuple image and readback;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteReflectionTuple}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden prompt state, oracle taste, unlisted compiler facts, a host-side approval row, a second tuple image, or a reflection request detached from I and R.

Theorem 993.2 (FiniteReflectionTuple NameCert obligations)**.** *For every accepted* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *carrier*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteReflectionTuple}^{\uparrow}}$ *surface consists exactly of carrier exposure for F, candidate identity through X, consumed stability data through S, consumed admission data through A, consumed compiler-witness data through W, tuple-image exposure through I, readback exposure through R, componentwise* hsame *and* NameCert *transport through H, replay routes through C, package provenance through P, reflection-request exposure through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleNameCert obligations`

Proof. Carrier exposure is projection from Definition 993.1. The packet lists every row available to a consumer of the finite reflection tuple.

The TasteGate rows are supplied by chapter 564, chapter 545, and chapter 563. The stability row supplies the carrier and non-escape boundary, the admission row supplies the candidate row being tested, and the compiler witness supplies the ground-compiler evidence read by W and replayed through C.

The reflection-specific obligations are I, R, H, C, P, Q, N. The tuple image I is the only displayed image, R is the only readback, H gives componentwise transport for both equality and local certificates, and Q records the bridge-facing request that may inspect the tuple.

Since the carrier has no coordinate for hidden prompt state, oracle taste, unlisted compiler facts, host-side approval, a second tuple image, or a request detached from I and R, the listed rows exhaust the local NameCert surface. □

Theorem 993.3 (FiniteReflectionTuple field-faithful prediction)**.** *Every accepted* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *packet is field-faithful to its displayed tuple image. A consumer that asks for the reflected candidate, stability packet, admission packet, compiler witness, componentwise transport, or* StandardBridge *request must read that field through I and recover it through R. A packet lacking either the displayed image row or the readback row is not a public reflection request over the accepted TasteGate route.*

Proof. Start with the carrier definition. The only image coordinate is I, and the only row that recovers fields from the image is R.

Use the NameCert obligations of Definition 993.2. They force candidate identity, stability, admission, compiler witness, transport, replay, provenance, and request reads through the displayed tuple surface.

The reflection-request row Q is therefore request data for the same image and readback. If either I or R is absent, a consumer may still name some TasteGate packet, but it cannot present the fixed tuple image/readback surface required by this carrier.

Thus every accepted field read is faithful to the displayed tuple, and a request without both image and readback rows is outside the public reflection packet. □

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier field faithful prediction`

Theorem 993.4 (FiniteReflectionTuple image-readback exactness)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

every public field read of X, S, A, W, C, P, Q, or N factors through the displayed tuple-image row I and the readback row R. No public read of the reflected candidate, consumed TasteGate packet, replay route, provenance row, request row, or local naming row may bypass I or recover its component from a second readback surface.

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier image readback exactness`

Proof. The carrier definition gives exactly one image coordinate, namely I, and exactly one readback coordinate, namely R. The public rows of F are therefore exposed through the displayed finite tuple rather than through a separate host-side table.

Apply Definition 993.2. Its certificate surface lists candidate identity, consumed stability data, consumed admission data, compiler-witness data, replay routes, provenance, request exposure, and local naming as rows of the same finite packet.

Now apply Definition 993.3. A consumer that asks for any reflected field must read that field through I and recover it through R. Since the carrier has no coordinate for a second image or second readback, the exactness claim follows. □

Theorem 993.5 (FiniteReflectionTuple reflection-request exhaustion)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

any StandardBridge*-facing request that inspects this finite reflection tuple is the displayed request row Q attached to the same I, R, H, C, P, N packet. A request detached from the displayed tuple image, readback row, componentwise transport row, replay route, provenance row, or local naming row is not a request over F.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier request exhaustion`

Proof. The definition of the carrier assigns request exposure to the single row Q. The same definition places Q in one tuple with I, R, H, C, P, and N.

The NameCert obligations of Definition 993.2 make request exposure part of the local certificate surface. Thus the request visible to a bridge-facing consumer is not an extra coordinate outside the accepted packet.

Use Definition 993.3 to bind that request back to the displayed image and readback. If a proposed request is not the displayed row Q for the same packet, or if it is detached from I, R, H, C, P, or N, it lies outside the public reflection route. □

Theorem 993.6 (FiniteReflectionTuple componentwise transport)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

componentwise hsame *transport and componentwise* NameCert *transport preserve exactly the displayed rows $X, S, A, W, I, R, C, P, Q, N$. They preserve the consumed* TasteGateStability$^{\uparrow}$, TasteGateAdmission$^{\uparrow}$, *and* TasteGateCompilerWitness$^{\uparrow}$ *certificate rows, and they cannot introduce a second tuple image or a reflection request detached from I and R.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier componentwise transport`

Proof. The row H is the only transport row in the carrier. By Definition 993.1, it stores componentwise hsame transport for $X, S, A, W, I, R, C, P, Q, N$ and componentwise NameCert transport for the three consumed TasteGate packets.

Apply Definition 993.2. The local certificate surface exposes the same component list and no additional transport coordinate.

It remains to rule out extra image and request data. The carrier definition excludes a second tuple image and a detached request, while Definition 993.4 and Definition 993.5 force public image reads and bridge-facing requests through the displayed I, R, and Q rows. Componentwise transport therefore preserves exactly the named rows. □

Theorem 993.7 (FiniteReflectionTuple public projection route)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

every public projection from F factors through the displayed image/readback pair I, R. Candidate identity, stability input, admission input, compiler-witness input, transport, replay, provenance, request, and local naming are first exposed as fields of the finite image I and then recovered through the readback row R.

There is no public projection from F that bypasses I, R, reads a second image, or recovers a component from host-side reflection state.

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier public projection route`

Proof. Begin with Definition 993.1. The tuple has exactly one displayed image row I and exactly one readback row R.

Apply Definition 993.4. It forces public field reads of the candidate, consumed TasteGate packets, replay, provenance, request, and local naming through the same image/readback pair.

Finally apply Definition 993.6. Componentwise hsame and NameCert transport preserve exactly the displayed rows and do not introduce a second image or detached request. Hence every public projection factors through I, R. □

Theorem 993.8 (FiniteReflectionTuple dependency ledger totality)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

the dependency ledger is exactly the three sibling TasteGate rows TasteGateStability$^{\uparrow}$, TasteGateAdmission$^{\uparrow}$, *and* TasteGateCompilerWitness$^{\uparrow}$, *together with the BEDC support rows* BHist, hsame, Cont, Pkg, *and* NameCert *displayed by H, C, P, N. No hidden dependency, host approval row, oracle taste row, or unlisted compiler theorem is part of the ledger.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier dependency ledger totality`

Proof. Read the carrier row by row. The fields S, A, and W are precisely the consumed TasteGateStability, TasteGateAdmission, and TasteGateCompilerWitness packets, while H, C, P, N record the transport, replay, provenance, and naming support.

Use Definition 993.2. It lists the local certificate surface and names no dependency other than the three sibling packets and the displayed BEDC rows.

The request exhaustion theorem, Definition 993.5, keeps the bridge-facing request attached to the same I, R, H, C, P, N packet. Thus the ledger is total for this public route and has no projection point for a hidden dependency. □

Theorem 993.9 (FiniteReflectionTuple audit-result nonescape)**.** *An audit-result read of an accepted* FiniteReflectionTuple$^{\uparrow}$ *packet can inspect only the displayed tuple image, readback, componentwise transport, replay, provenance, request, local naming, and the three listed TasteGate sibling inputs. It cannot import hidden prompt state, oracle taste, host approval, a detached reflection request, a second image/readback surface, or an unlisted compiler theorem.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier audit result nonescape`

Proof. First apply Definition 993.7. Any public read of the tuple, including an audit-result read, factors through I, R.

Next apply Definition 993.8. The only dependency inputs available to that read are the three sibling TasteGate rows and the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally use Definition 993.6 and Definition 993.5. They keep transport and bridge-facing requests inside the same finite packet. Since the excluded objects have no carrier coordinate, no image row, no readback row, and no request row, an audit-result read cannot escape to them. □

Theorem 993.10 (FiniteReflectionTuple StdBridge request scope)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

every StandardBridge*-facing request over F must read the candidate identity,* TasteGateStability$^{\uparrow}$ *input,* TasteGateAdmission$^{\uparrow}$ *input,* TasteGateCompilerWitness$^{\uparrow}$ *input, tuple image, readback, componentwise transport, replay, provenance, request, and local* NameCert *rows through that same packet. No such request may import hidden prompt state, a second image/readback surface, host approval, oracle taste, or an unlisted compiler theorem.*

Proof. Start with Definition 993.5. It identifies the bridge-facing request with the displayed row Q attached to the same tuple image, readback, transport, replay, provenance, and local naming rows.

Use Definition 993.8. The only dependency inputs available to that request are the three consumed TasteGate packets and the displayed BEDC support rows.

Apply Definition 993.9. Hidden prompt state, oracle taste, host approval, detached requests, second image/readback surfaces, and unlisted compiler theorems have no carrier coordinate in an accepted audit-result read.

Finally invoke Definition 993.3. Since each reflected field is read through I and recovered through R, the request scope is exactly the displayed finite packet $X, S, A, W, I, R, H, C, P, Q, N$. □

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleTasteGate single carrier alignment`

Theorem 993.11 (FiniteReflectionTuple bridge-facing row totality)**.** *For every accepted* FiniteReflectionTuple$^{\uparrow}$ *packet*

$$F = (X, S, A, W, I, R, H, C, P, Q, N),$$

every bridge-facing request over F factors through the displayed tuple image I, readback row R, componentwise transports H, replay row C, provenance row P, request row Q, and local NameCert row N, together with the candidate and consumed TasteGate sibling rows X, S, A, W. No hidden prompt state, second image/readback surface, host approval, oracle taste, or unlisted compiler theorem can serve the request.

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier bridge facing row totality`

Proof. Begin with the NameCert obligations. They expose the carrier row X, consumed stability, admission, and compiler-witness rows S, A, W, the tuple image I, readback R, componentwise transports H, replay C, provenance P, request row Q, and local naming row N.

Apply image-readback exactness. It forces every public image read through I and every recovery of reflected data through R, so a bridge-facing request cannot introduce a second image/readback surface.

Use request exhaustion and componentwise transport. The request row Q is attached to the same tuple image, readback, transports, replay, provenance, and local naming rows, and the transports keep the reflected components inside the accepted finite packet.

Finally invoke the StdBridge request-scope theorem. It already excludes hidden prompt state, host approval, oracle taste, second image/readback surfaces, and unlisted compiler theorems for StandardBridge-facing requests, so every bridge-facing row is total over the displayed packet $X, S, A, W, I, R, H, C, P, Q, N$. □

Theorem 993.12 (FiniteReflectionTuple bridge-request boundary)**.** *Every* StandardBridge *request over an accepted* FiniteReflectionTuple$^{\uparrow}$ *packet must first read* $X, S, A, W, I, R, H, C, P, Q, N$ *through the displayed tuple image and readback. The request cannot substitute host approval, oracle taste, an unlisted compiler theorem, or detached request state for the image/readback packet.*

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.FiniteReflectionTupleCarrier bridge request boundary`

Proof. Start with image-readback exactness. It forces every public read of the reflected candidate, consumed TasteGate rows, replay, provenance, request, and local name through the displayed image I and readback row R.

Next apply request exhaustion. The StandardBridge-facing request is the displayed row Q attached to the same I, R, H, C, P, N packet, rather than a detached request state.

Use dependency ledger totality and bridge-facing row totality together. They expose the allowed dependency rows X, S, A, W and the support rows H, C, P, N, while excluding host approval, oracle taste, hidden prompt state, second image/readback surfaces, and unlisted compiler theorems.

Therefore a bridge request over the accepted packet is a consumer handoff for exactly $X, S, A, W, I, R, H, C, P, Q$ through the displayed image/readback route. □

Falsifiable prediction: A FiniteReflectionTupleUp packet without displayed image and readback rows cannot support a StdBridge request.

Independence witness: tastegatestability, tastegateadmission, tastegatecompilerwitness, kernelacceptancetrace: none contains the fixed tuple image/readback/componentwise transport surface together with a reflection-request row.

Lean development for this route must provide Nontrivial FiniteReflectionTuple$^{\uparrow}$ and FieldFaithful FiniteReflectionT instances in `lean4/BEDC/Derived/FiniteReflectionTupleUp/TasteGate.lean`.

Closure status for NameCert$_{\text{FiniteReflectionTuple}^{\uparrow}}$.

Constructive story: A FiniteReflectionTuple$^{\uparrow}$ packet is generated from a candidate chapter identity row, consumed TasteGateStability$^{\uparrow}$, TasteGateAdmission$^{\uparrow}$, and TasteGateCompilerWitness$^{\uparrow}$ packets, a displayed tuple image, a readback row, componentwise hsame and NameCert transports, continuation routes, provenance, a reflection-request row, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteReflectionTuple}^{\uparrow}}$, maturePackageClosed)

Lean4 ✓ `BEDC.Derived.FiniteReflectionTupleUp.taste gate`

Scope closed: Definition 993.1, Definition 993.2, Definition 993.4, Definition 993.5, Definition 993.6, Definition 993.7, Definition 993.8, Definition 993.9, Definition 993.10, Definition 993.11, the falsifiable prediction, and the independence witness bind the public image/readback projection, reflection request, dependency ledger, audit-result boundary, and bridge-facing request rows for the same finite packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteReflectionTuple}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, axiom-clean audit, bridge checking, hidden prompt-state reconstruction, oracle taste, host-side approval, unlisted compiler facts, or a reflection request detached from the displayed image and readback rows.

Upgrade path: No further paper-side theory grade remains; independent Lean bridge checking remains on the formal axis for the same tuple image, readback, transport, and reflection-request rows.

Chapter 994

A Concrete Naming Certificate for $\mathsf{RealSelectorComparison}^{\uparrow}$

The $\mathsf{RealSelectorComparison}^{\uparrow}$ packet realizes the Real and Completion target of chapter 9238 as a finite comparison between two diagonal selector rows before either route reaches the $\mathsf{Real}^{\uparrow}$ seal. It consumes the finite window discipline of chapter 126, the regular-rational readback of chapter 301, the limit-seal surface of chapter 386, the tail-agreement seal of chapter 699, the modulus refinement row of chapter 885, and the real-name seal of chapter 38. The packet is not a host equality of completed reals; it is the displayed row package that forces two selector routes to share the same finite tail, readback, agreement, and seal evidence.

Definition 994.1 (Real selector comparison carrier)**.** *A* $\mathsf{RealSelectorComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D_0, D_1, T, W, R, L, A, H, C, P, N)$$

with the following displayed coordinates:

1. D_0 *and* D_1 *are the two diagonal selector rows whose finite histories are being compared;*
2. T *is the shared tail row, including the common threshold and window request used by both selector routes;*
3. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row that both selectors read before any regular-rational handoff;*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording that both selector routes decode the same regular rational tail at* T*;*
5. L *is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row reached after the shared regular-rational readback;*
6. A *is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row witnessing that the two tails agree on the displayed cofinal window;*
7. H *records componentwise* hsame *transports for* D_0, D_1, T, W, R, L, A *and the final* $\mathsf{Real}^{\uparrow}$ *seal row;*
8. C *records the displayed* Cont *routes that replay the two diagonal selector rows through the same tail, window, readback, limit-seal, tail-agreement, and* $\mathsf{Real}^{\uparrow}$ *seal path;*
9. P *is the* Pkg *provenance over the diagonal selectors,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSelectorComparison}^{\uparrow}}$ *row naming exactly this two-selector comparison packet.*

The carrier has no coordinate for a selected host limit, quotient-stream equality, independent second tail window, ambient completeness theorem, or Real seal route detached from $T, W, R, L, A, H, C, P, N$.

Theorem 994.2 (Real selector comparison NameCert obligations)**.** *For every accepted* $\mathsf{RealSelectorComparison}^{\uparrow}$ *carrier*

$$S = (D_0, D_1, T, W, R, L, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSelectorComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, two diagonal selector admissions through D_0 and D_1, shared tail and window admission through T and W, regular-rational readback through R, Cauchy-limit seal admission through L, tail-agreement seal admission through A, componentwise* hsame *transport through H, displayed route replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealSelectorComparisonUp.RealSelectorComparisonTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 994.1. The only selector data available to the local certificate are the two diagonal rows D_0, D_1 and the shared tail/window/readback route T, W, R.

The seal obligations are then read in route order. The Cauchy-limit row L consumes the regular-rational readback, and the tail-agreement row A certifies that the two selector tails agree on the same cofinal window before the $\mathsf{Real}^{\uparrow}$ seal can be exposed.

Componentwise transports in H and displayed continuations in C replay the same finite rows for both selector routes, while P and N record provenance and local naming. Since the carrier has no detached host limit, quotient equality, second tail window, independent completion theorem, or alternate Real seal coordinate, no further NameCert obligation can be read from the packet. □

Theorem 994.3 (Real selector comparison diagonal readback)**.** *For an accepted* $\mathsf{RealSelectorComparison}^{\uparrow}$ *carrier $S = (D_0, D_1, T, W, R, L, A, H, C, P, N)$, both diagonal selector routes read the same* $\mathsf{StreamName}^{\uparrow}$ *window row W and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R at the shared tail row T. Any* $\mathsf{Real}^{\uparrow}$ *seal obtained from either route therefore factors through the common finite path*

$$D_i, T, W, R, L, A, H, C, P, N$$

for $i = 0, 1$.

Proof. Start with the two selector rows D_0 and D_1. The carrier gives only one tail row T, one finite window row W, and one regular-rational readback row R for both selectors.

Follow the displayed continuations in C. Each selector route first reaches T, then reads W, then decodes R, and only then reaches the limit and agreement rows L and A.

The transports in H preserve those coordinates componentwise. Thus either selector route reaches a $\mathsf{Real}^{\uparrow}$ seal through the same finite path, with no available coordinate for a second readback or a separately selected limit. □

Theorem 994.4 (Real selector comparison seal non-escape)**.** *An accepted* $\mathsf{RealSelectorComparison}^{\uparrow}$ *carrier exports no completion-facing route except the displayed two-selector packet $S = (D_0, D_1, T, W, R, L, A, H, C,$ In particular, it cannot expose incompatible* $\mathsf{Real}^{\uparrow}$ *seals for D_0 and D_1 while preserving the shared tail/window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route, and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row.*

Proof. Assume a completion-facing consumer reads a seal from the accepted packet. The consumer can inspect only $D_0, D_1, T, W, R, L, A, H, C, P, N$, because those are the complete displayed coordinates of the carrier.

If the consumer follows the D_0 route, Definition 994.3 forces it through T, W, R, L, A before the seal read. The same theorem forces the D_1 route through the identical tail, window, readback, limit-seal, and agreement rows.

Therefore incompatible seal outputs would require an extra selector tail, a second regular readback, a detached Cauchy-limit seal, or a Real seal row outside the packet. None of those coordinates exists, so the seal route cannot escape the displayed comparison surface. □

Falsifiable prediction: Deleting the shared tail/window row permits two selector routes to reach incompatible RealUp seals, because the packet no longer forces their StreamName windows and RegSeqRat readbacks through one finite tail threshold.

Independence witness: Nearest siblings: cauchylimitseal, realtailagreementseal, cauchymodulusrefinement, streamname, and regseqrat. None is bijective to $\mathsf{RealSelectorComparison}^{\uparrow}$ because this packet carries two diagonal selector rows plus one shared tail/window/readback route, one CauchyLimitSeal row, one RealTailAgreementSeal row, and one RealUp seal route in a single NameCert surface.

The Lean file `Derived/RealSelectorComparisonUp/TasteGate.lean` must provide Nontrivial RealSelectorCom and FieldFaithful $\mathsf{RealSelectorComparison}^{\uparrow}$ instances, with field injection for $D_0, D_1, T, W, R, L, A, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSelectorComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSelectorComparison}^{\uparrow}$ packet is generated from two diagonal selector rows, one shared tail/window row, one regular-rational readback row, CauchyLimitSeal and RealTailAgreementSeal rows, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSelectorComparison}^{\uparrow}}$, seed)

Scope closed: Definition 994.1, Definition 994.2, Definition 994.3, and Definition 994.4 bind the finite two-selector comparison carrier, obligation surface, shared readback route, and seal non-escape boundary for the Real and Completion handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSelectorComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, host equality of completed reals, quotient-stream equality, selected limits, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, shared tail/window readback, CauchyLimitSeal route compatibility, RealTailAgreementSeal compatibility, seal non-escape, and TasteGate nontriviality with field faithfulness.

Chapter 995

A Concrete Naming Certificate for $\mathsf{CellularPatternCatalog}^{\uparrow}$

The $\mathsf{CellularPatternCatalog}^{\uparrow}$ packet realizes the finite glider, ether, and phase-row classifier described in Definition 9390.1 and Definition 9390.3. It sits beside the cellular substrate classifier of chapter 945, the local update rule of chapter 984, the halting trace classifier of chapter 978, the halting finite boundary of chapter 985, and the cellular verb realignment packet of chapter 987. The certificate records a finite pattern catalog, not an unbounded orbit semantics or a host-side judgment that every future row has been classified.

Falsifiable prediction: If a packet lacks either a finite phase-tag row or a finite local-window row, then it cannot satisfy the $\mathsf{CellularPatternCatalog}^{\uparrow}$ NameCert obligations.

Independence witness: cellularsubstrateclassifier, cellularlocalupdaterule, haltingtraceclassifier, haltingfiniteboundary, cellularverbrealignment: the catalog carrier is not bijective to any sibling because it carries the joint rule id, local window, finite phase/glider/ether tag, catalog row, transport, route, provenance, and local NameCert surface. The substrate classifier lacks the explicit catalog row, the local update rule lacks the tag classifier, the halting trace classifier and halting finite boundary lack the cellular phase catalog, and cellular verb realignment lacks the finite local-window catalog obligation.

Definition 995.1 (Cellular pattern catalog carrier)**.** *A* $\mathsf{CellularPatternCatalog}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, T, G, H, C, P, N)$$

with the following displayed rows:

1. *R is the rule-identity row, naming the finite local cellular rule whose rows are being classified;*
2. *W is the finite local-window row, displaying the bounded cell neighbourhood read by the catalog;*
3. *T is the finite phase, glider, and ether tag row, whose labels are drawn from the catalog's closed tag table;*
4. *G is the catalog row, listing the finite phase strings, glider names, ether row, and local matching discipline used by the classifier;*
5. *H records componentwise* hsame *transport for the rule id, window, tag, catalog, provenance, and local naming rows;*

6. *C records the displayed* Cont *routes from rule id to local window, from local window to catalog row, from catalog row to tag row, and from those rows back through provenance and local naming;*
7. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, *chapter 9390, and the sibling cellular and halting packets listed above;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularPatternCatalog}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through R, W, T, G, H, C, P, N. It has no coordinate for an infinite orbit classifier, host visual recognition, a global completeness theorem for all rows, or an unlisted pattern tag.

Lean4 (definition) `BEDC.Derived.CellularPatternCatalogUp.CellularPatternCatalogCarrier`

Theorem 995.2 (CellularPatternCatalog NameCert obligations)**.** *For the carrier of Definition 995.1, the local* $\mathsf{NameCert}_{\mathsf{CellularPatternCatalog}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (R, W, T, G, H, C, P, N)$, rule-id exposure, finite local-window exposure, finite phase/glider/ether tag exposure, catalog-row lookup, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and local naming. Every accepted tag must be read through the route*

$$R \longmapsto W \longmapsto G \longmapsto T,$$

with H, C, P, N attached as transport, route, provenance, and naming rows.

Proof. Carrier admission is projection from Definition 995.1. The displayed tuple gives exactly the rule id, local window, tag row, catalog row, transport, route, provenance, and local naming coordinates.

The cellular source of the classifier is chapter 9390. Its glider and ether examples become finite catalog rows here: a consumer first reads the rule row R, then the local window W, then the catalog row G, and only then the finite tag row T.

The sibling chapters bind the surrounding lattice. The substrate classifier of chapter 945 supplies the cellular classifier surface, chapter 984 supplies local update discipline, chapter 978 and chapter 985 keep trace and boundary reads finite, and chapter 987 aligns the cellular vocabulary with the NameCert route.

Finally, H, C, P, N preserve and name the same finite catalog. Since no row of K is an infinite orbit classifier, a host visual recognizer, a global completeness theorem, or an unlisted tag, the NameCert obligations are exhausted by the displayed route. □

Lean4 ✓ `BEDC.Derived.CellularPatternCatalogUp.CellularPatternCatalogTasteGate single carrier alignment`

Lemma 995.3 (CellularPatternCatalog tag lookup determinacy)**.** *Let*

$$K = (R, W, T, G, H, C, P, N)$$

and

$$K' = (R', W', T', G', H', C', P', N')$$

be accepted $\mathsf{CellularPatternCatalog}^{\uparrow}$ *carriers in the sense of Definition 995.1. Suppose that the two packets agree on the rule-identity row, finite local-window row, catalog row, componentwise transport row, continuation route row, provenance row, and local naming row:*

$$R = R', \quad W = W', \quad G = G', \quad H = H', \quad C = C', \quad P = P', \quad N = N'.$$

Then the exposed finite phase, glider, and ether tag rows T and T' classify together under the local $\mathsf{NameCert}_{\mathsf{CellularPatternCatalog}^{\uparrow}}$ *classifier.*

Proof. By Definition 995.2, every accepted tag read in a $\mathsf{CellularPatternCatalog}^{\uparrow}$ packet is consumed through the displayed route

$$R \longmapsto W \longmapsto G \longmapsto T,$$

with H, C, P, N attached as the transport, route replay, provenance, and local naming rows. Thus T is not an independent tag judgment: it is the tag row exposed by the finite catalog lookup whose visible inputs are the rule row R, the local window row W, and the catalog row G.

The same statement holds for K' through the route

$$R' \longmapsto W' \longmapsto G' \longmapsto T'.$$

The hypotheses identify the lookup inputs R, W, G with R', W', G' and also identify the rows H, C, P, N with H', C', P', N'. Therefore both accepted packets replay the same finite catalog row against the same finite local window under the same local naming and provenance surface.

The catalog row G is finite by Definition 995.1, and its tag labels are drawn from the closed tag table described in Definition 9390.1 and instantiated by the glider and ether catalog of Definition 9390.3. On a fixed finite rule row and fixed finite local window, reading the same displayed catalog row exposes a single local tag-classification row. A failure of T and T' to classify together would therefore require either a different rule identity, a different local window, a different catalog row, a different replay or transport row, or an extra host-side tag coordinate outside the carrier.

No such coordinate is present in Definition 995.1, and the agreed rows exhaust the lookup route of Definition 995.2. Hence the two exposed finite phase, glider, and ether tag rows classify together. □

Lean4 ✓ `BEDC.Derived.CellularPatternCatalogUp.CellularPatternCatalog tag lookup determinacy`

Theorem 995.4 (CellularPatternCatalog public lookup surface)**.** *Every public consumer of an accepted* $\mathsf{CellularPatternCatalog}^{\uparrow}$ *carrier may read the finite catalog lookup only through the rule row R, local-window row W, catalog row G, phase/glider/ether tag row T, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row of Definition 995.1. The public surface is exactly the finite lookup route constrained by Definition 995.2 and Definition 995.3; it exports no global orbit classifier, host visual recognizer, completeness theorem for all future rows, or unlisted pattern tag.*

Proof. Project the carrier of Definition 995.1. A public consumer receives only R, W, T, G, H, C, P, N, so every visible catalog question is a finite lookup against the displayed rule, local window, catalog, and tag rows.

Apply the NameCert obligations of Definition 995.2. They force the lookup route through

$$R \longmapsto W \longmapsto G \longmapsto T$$

with H, C, P, N attached as transport, replay, provenance, and naming data.

Finally use tag lookup determinacy, Definition 995.3. Once the rule, local window, catalog, transport, replay, provenance, and local naming rows are fixed, the exposed tag row is fixed by the finite catalog surface. Since no carrier coordinate stores an infinite orbit semantics, a host visual recognizer, a complete future-row classifier, or an unlisted tag, the public surface is exactly the finite lookup interface. □

Lean4 ✓ `BEDC.Derived.CellularPatternCatalogUp.CellularPatternCatalog public lookup surface`

Closure status for $\mathsf{NameCert}_{\mathsf{CellularPatternCatalog}^{\uparrow}}$.

Constructive story: A CellularPatternCatalog$^{\uparrow}$ packet is generated from a finite cellular rule id, local window, phase/glider/ether tag row, catalog row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CellularPatternCatalog}^{\uparrow}}$, publicClosed)

Scope closed: Definition 995.1, Definition 995.2, Definition 995.3, and Definition 995.4 bind the finite rule, local window, tag, catalog, transport, route, provenance, local naming rows, and existing TasteGate marker to the cellular substrate classifier, local update rule, halting trace boundary, halting finite boundary, and cellular verb realignment sources.

Formal verification: FormalStatus(NameCert$_{\text{CellularPatternCatalog}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a complete orbit classifier, host visual recognition, global classification of all future rows, audit-clean purity, or a public standard bridge.

Upgrade path: Audit-clean closure requires purity review for the finite carrier, catalog lookup determinacy, substrate and update-rule dependency, halting boundary dependency, cellular verb realignment route, transport, replay, provenance, and local naming rows for BEDC.Derived.CellularPatternCatalogUp.

Chapter 996

A Concrete Naming Certificate for $\mathsf{RatClassifierTransportSeal}^{\uparrow}$

The $\mathsf{RatClassifierTransportSeal}^{\uparrow}$ packet realizes the transported stationary rational diagonal row of chapter 9238 as one finite NameCert handoff. It consumes the RatUp source classifier of chapter 37, the constant StreamName classifier surface of chapter 126, the RegSeqRat finite-window seal of chapter 301, the DyadicRatCore observation ledger of chapter 291, and the RealUp classifier handoff of chapter 38.

Definition 996.1 (RatClassifierTransportSeal carrier)**.** *A* $\mathsf{RatClassifierTransportSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, S, W, D, A, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the RatUp source-classifier row carrying the accepted stationary rational comparison;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *classifier row induced by the same rational comparison;*
3. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal row consuming the constant stream windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for the same finite windows;*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *classifier handoff row reached only after Q, S, W, D are replayed;*
6. *H records componentwise* hsame *transports for Q, S, W, D, A, package provenance, and the local naming row;*
7. *C records the displayed* Cont *routes from the RatUp row through the StreamName, RegSeqRat, DyadicRatCore, and RealUp readings;*
8. *P is the* Pkg *provenance over RatUp,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RatClassifierTransportSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for an independent RealUp equality, quotient stream, selected limit, host rational equality, or ambient completeness principle.

Lean4 (definition) `BEDC.Derived.RatClassifierTransportSealUp.RatClassifierTransportSealUp`

Theorem 996.2 (RatClassifierTransportSeal NameCert obligations)**.** *For every accepted* $\mathsf{RatClassifierTransportSeal}$ *carrier $R = (Q, S, W, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RatClassifierTransportSeal}^{\uparrow}}$ *surface consists*

exactly of carrier admission, RatUp source-classifier exposure through Q, constant StreamName classifier exposure through S, RegSeqRat finite-window seal exposure through W, DyadicRatCore ledger exposure through D, RealUp classifier handoff through A, componentwise hsame *transport through H, displayed route replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RatClassifierTransportSealUp.RatClassifierTransportSealCarrier namecert obligations`

Proof. Project the finite carrier of Definition 996.1. Each obligation row is one coordinate of R, so the local NameCert read can inspect only $Q, S, W, D, A, H, C, P, N$.

Read the rows in transport order. The RatUp classifier Q induces the constant StreamName row S, the RegSeqRat seal W consumes the displayed finite windows, the DyadicRatCore ledger D records the observed dyadic readings, and the RealUp handoff A is exposed only after those rows have been replayed through H, C, P, N.

No further obligation can appear because the carrier contains no independent equality, quotient, selected limit, host rational, or ambient completeness coordinate. Thus the displayed rows exhaust the local NameCert surface. □

Theorem 996.3 (RatClassifierTransportSeal window exhaustion)**.** *Every public finite-window read of an accepted* $\mathsf{RatClassifierTransportSeal}^{\uparrow}$ *packet factors through the RatUp seed classifier row Q before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *classifier row is read. Once Q is fixed, the public route is exhausted by S, W, D, A, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RatClassifierTransportSealUp.RatClassifierTransportSealCarrier window exhaustion`

Proof. Start with the route row C and the componentwise transport row H. They display a single ordered path from the rational classifier through constant StreamName windows, RegSeqRat finite-window sealing, DyadicRatCore ledger reads, and the RealUp handoff.

If a consumer attempts to read S, W, D, or A first, it asks for a coordinate whose route is defined only as a replay from Q. The package provenance P and the local naming row N record that dependency for the same accepted packet.

Therefore every public window read begins at Q and is exhausted by the remaining displayed rows. The packet exports no route to a separate stream, dyadic ledger, real classifier, or completion principle outside R. □

Theorem 996.4 (RatClassifierTransportSeal classifier handoff determinacy)**.** *Let $R = (Q, S, W, D, A, H, C, P, N)$ and $R' = (Q', S', W', D', A', H', C', P', N')$ be accepted* $\mathsf{RatClassifierTransportSeal}^{\uparrow}$ *carriers. Suppose the RatUp source-classifier rows Q, Q', the constant* $\mathsf{StreamName}^{\uparrow}$ *rows S, S', the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seals W, W', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers D, D', the componentwise transport rows H, H', the continuation route rows C, C', the package provenance rows P, P', and the local naming rows N, N' agree under the displayed* hsame *and* Cont *transport from the RatUp source route. Then the* $\mathsf{Real}^{\uparrow}$ *classifier handoff rows A and A' cannot diverge: every public* $\mathsf{Real}^{\uparrow}$ *classifier read factors through the same visible route Q, S, W, D, H, C, P, N, transported to $Q', S', W', D', H', C', P', N'$.*

Lean4 ✓ `BEDC.Derived.RatClassifierTransportSealUp.RatClassifierTransportSealCarrier classifier handoff determinacy`

Proof. Project both accepted carriers through Definition 996.1. Their public coordinates are exactly the displayed rows, with A and A' reached only after the RatUp, StreamName, RegSeqRat, and DyadicRatCore rows have been replayed.

Apply Definition 996.3 to R and to R'. Any public $\mathsf{Real}^{\uparrow}$ classifier read from either packet must first pass through the RatUp seed classifier and then through the finite route exhausted by the remaining displayed coordinates.

The hypotheses identify those finite routes: Q matches Q', the constant StreamName rows match, the finite-window seals and dyadic ledgers match, and H, C, P, N transport the same package and naming data to H', C', P', N'. Thus the replayed route that reaches A is the same route that reaches A'.

A distinct public $\mathsf{Real}^{\uparrow}$ handoff would therefore require an additional coordinate recording an independent RealUp equality, quotient stream, selected limit, host rational equality, or ambient completeness principle. Such a coordinate is excluded by Definition 996.1. Hence A and A' cannot diverge along accepted public reads. □

Falsifiable prediction: Every public finite-window read of an accepted RatClassifierTransportSealUp packet factors through the RatUp seed classifier before any StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, or RealUp classifier row is read.

Independence witness: rat, streamname, regseqrat, dyadicratcore, real: RatUp lacks finite-window and Real seal rows; StreamNameUp lacks the rational seed classifier; RegSeqRatUp lacks the RatUp source row; DyadicRatCoreUp lacks the Real classifier handoff; RealUp lacks the four-reading transport ledger, so no bijection preserves the full carrier.

The Lean file `Derived/RatClassifierTransportSealUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{RatClassifierTransportSeal}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RatClassifierTransportSeal}^{\uparrow}$ instances for the displayed carrier $Q, S, W, D, A, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RatClassifierTransportSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RatClassifierTransportSeal}^{\uparrow}$ packet is generated from a RatUp source classifier, constant StreamName classifier, RegSeqRat finite-window seal, DyadicRatCore observation ledger, RealUp classifier handoff, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RatClassifierTransportSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 996.1, Definition 996.2, Definition 996.3, and Definition 996.4 bind the finite transported stationary rational classifier seal across RatUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RatClassifierTransportSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, host rational equality, ambient metric completion, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, RatUp source factorization, finite-window seal exhaustion, RealUp handoff non-escape, and TasteGate nontriviality with field faithfulness.

Chapter 997

A Concrete Naming Certificate for $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$

The $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ packet realizes the reflection-limit validation boundary of chapter 9461 as a finite runtime checkpoint. It sits beside the runtime-candidate boundary of chapter 591, the finite trace evaluator of chapter 818, the finite trace gap socket of chapter 504, and the reflection-limit boundary of chapter 586. Those siblings expose candidate rows, bounded evaluator rows, finite gap rows, and open meta-boundary rows; the present packet records a checkpoint state together with the validation-boundary row that a runtime-facing consumer must read before making any host-valid claim.

Definition 997.1 (ReflectionRuntimeCheckpoint carrier)**.** *A* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, T, V, H, C, P, N)$$

with the following displayed rows:

1. *I is the admitted runtime-candidate input row, recording the coded* CIC* *expression, proof-object, or state being checkpointed;*
2. *S is the displayed checkpoint-state row, recording the finite state reached by the candidate before any host validation is asserted;*
3. *T is the trace-evidence row, listing the bounded evaluator or gap-socket observations used to justify the checkpoint state;*
4. *V is the validation-boundary row, recording that host validation of the checkpoint is not itself a packet coordinate and cannot be replaced by the checkpoint state;*
5. *H records componentwise* hsame *transport for input, checkpoint state, trace evidence, validation boundary, provenance, and local naming;*
6. *C records the displayed* Cont *routes by which a consumer reads I, reaches S, replays T, inspects V, and returns through transport and naming;*
7. *P is the* Pkg *provenance over chapter 9461, chapter 591, chapter 818, chapter 504, chapter 586,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-valid runtime judgment, a total truth row, an unbounded evaluator, or a validation act detached from V.

Theorem 997.2 (ReflectionRuntimeCheckpoint NameCert obligations)**.** *For the carrier of Definition 997.1, the local* $\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed checkpoint packet* $K = (I, S, T, V, H, C, P, N)$*;*
2. *runtime-candidate exposure, requiring every checkpointed input read to factor through* I*;*
3. *checkpoint-state and trace exactness, requiring every displayed checkpoint state to be read through* S *and every supporting bounded observation to be read through* T*;*
4. *validation-boundary non-escape, requiring every host-validation-facing consumer to pass through* V *and preventing* I, S, T, H, C, P, N *from exporting host validation or total truth;*
5. *transport, route, provenance, and naming closure, requiring all componentwise comparisons, consumer continuations, package dependencies, and local naming to be exactly the displayed rows* H, C, P, N*.*

Thus a consumer may inspect a finite checkpoint state and its bounded trace evidence, but it cannot turn that checkpoint into a host-valid runtime judgment or a total truth row.

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier trace evidence finiteness`

Proof. Carrier admission is projection from Definition 997.1. The packet lists I, S, T, V, H, C, P, N as the complete surface available to a consumer.

Runtime-candidate exposure begins at I. The checkpoint state is not a free-standing state; it is the state of the admitted runtime-candidate input named by the first row.

Checkpoint-state and trace exactness then read S and T. The state row records the displayed checkpoint, and the trace-evidence row records only bounded evaluator or gap-socket observations supplied by the finite siblings.

Validation-boundary non-escape is the separate row V. By the reflection-limit boundary in chapter 9461 and by the object-level runtime surfaces of chapter 591, chapter 818, and chapter 504, host validation is not a packet coordinate. Therefore no read of I, S, T, H, C, P, N can export host validation, total truth, or an unbounded evaluator.

Finally H, C, P, N transport, route, cite, and name the same finite checkpoint packet. Since those rows add no further runtime or truth coordinate, the listed five obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier namecert obligations`

Theorem 997.3 (ReflectionRuntimeCheckpoint TasteGate obligation)**.** *The Lean* `TasteGate.lean` *development for* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *provides explicit* $\mathsf{Nontrivial\ ReflectionRuntimeCheckpoint}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ ReflectionRuntimeCheckpoint}^{\uparrow}$ *instances preserving the runtime input row, checkpoint-state row, trace-evidence row, validation-boundary row, componentwise transport, continuation routes, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointUp StdBridge`

Proof. The carrier has at least two distinguishable checkpoint packets, obtained by varying the displayed input row while holding the remaining rows fixed. This witnesses nontriviality.

Field faithfulness is the componentwise reading of $K = (I, S, T, V, H, C, P, N)$. If two packets have the same displayed field list, then each of their eight BHist coordinates agrees, and constructor

equality identifies the packets. The Lean TasteGate surface records these two facts as the required instances. □

Theorem 997.4 (ReflectionRuntimeCheckpoint input-state exposure)**.** *For every accepted* ReflectionRuntimeCheck *carrier*

$$K = (I, S, T, V, H, C, P, N),$$

every checkpoint consumer reads the runtime-candidate input row I and checkpoint-state row S before it reads any trace-evidence row or validation-facing row. Thus neither T nor V can be treated as a detached source of runtime state.

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier input state exposure`

Proof. Project the carrier of Definition 997.1. The packet is ordered as $K = (I, S, T, V, H, C, P, N)$, with I carrying the admitted runtime candidate and S carrying the displayed checkpoint state reached by that candidate.

Apply the runtime-candidate exposure and checkpoint-state exactness clauses of Definition 997.2. They force every checkpointed input read through I and every checkpoint-state read through S before the trace row T or validation-boundary row V is available.

Finally use the TasteGate carrier alignment of Definition 997.3. Field faithfulness keeps I and S as the first two displayed checkpoint coordinates, so a consumer cannot bypass them by reading T, V, H, C, P, or N as a substitute input-state source. □

Theorem 997.5 (ReflectionRuntimeCheckpoint trace-boundary readback)**.** *For every accepted* ReflectionRuntimeCheckpoint$^{\uparrow}$ *carrier $K = (I, S, T, V, H, C, P, N)$, the row T is bounded trace evidence routed through the finite trace evaluator and finite trace gap-socket siblings, through the package provenance P, and back through the transport, continuation, and naming rows H, C, N. It is not an unbounded evaluator and it is not a validation act.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier trace boundary readback`

Proof. Start from the provenance row in Definition 997.1. The row P cites the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, and reflection-limit boundary as the displayed sources for the checkpoint packet.

The NameCert obligations of Definition 997.2 force every supporting bounded observation through T. The sibling finite evaluator and gap-socket chapters supply the bounded trace surfaces, while H, C, P, N record the componentwise transport, consumer route, provenance, and local name of the same readback.

Finally use the TasteGate carrier alignment of Definition 997.3. Field faithfulness keeps T distinct from V and from the route ledger H, C, P, N, so the trace readback remains bounded evidence rather than an unbounded evaluator or detached validation act. □

Theorem 997.6 (ReflectionRuntimeCheckpoint validation non-escape route)**.** *For every accepted* ReflectionRuntimeCheckpoint$^{\uparrow}$ *carrier $K = (I, S, T, V, H, C, P, N)$, any validation-facing read must pass through the validation-boundary row V. No route through I, S, T, H, C, P, N alone exports a host-valid runtime judgment, total truth, or an unbounded evaluator.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier validation nonescape route`

Proof. Project the carrier of Definition 997.1. The rows I, S, T expose only the candidate input, displayed checkpoint state, and bounded trace evidence, while H, C, P, N transport, route, cite, and name that same finite packet.

Apply the validation-boundary non-escape clause of Definition 997.2. It names V as the sole validation-boundary row and states that the remaining rows cannot export host validation or total truth.

Finally use the TasteGate carrier alignment of Definition 997.3. Field faithfulness keeps V as a separate coordinate, so a consumer cannot identify the validation boundary with I, S, T, or any transport, route, provenance, or naming row. □

Theorem 997.7 (ReflectionRuntimeCheckpoint root-unblock package)**.** *The carrier of Definition 997.1, the NameCert obligations of Definition 997.2, the input-state exposure of Definition 997.4, the trace-boundary readback of Definition 997.5, the validation non-escape route of Definition 997.6, and the TasteGate carrier-alignment row of Definition 997.3 form the finite root-unblock package for downstream runtime-checkpoint consumers. Any such consumer receives exactly the displayed rows I, S, T, V, H, C, P, N and their finite routes, with no host validation, total truth, unbounded evaluator, or detached runtime judgment.*

Proof. The carrier supplies the rows I, S, T, V, H, C, P, N. The NameCert obligations identify the admissible reads of those rows and exclude every unlisted runtime or truth coordinate.

The input-state exposure forces reads through I and S before trace or validation rows. The trace-boundary readback then routes T through the cited runtime-candidate, finite trace evaluator, finite trace gap socket, reflection-limit, transport, continuation, provenance, and naming boundaries.

The validation non-escape route forces every validation-facing read through V. The TasteGate obligation contributes the single-carrier alignment: nontriviality supplies distinguishable checkpoint packets, and field faithfulness preserves every displayed coordinate. These ingredients give downstream consumers a finite checkpoint package and no additional root-facing validation surface. □

Theorem 997.8 (ReflectionRuntimeCheckpoint validation boundary totality)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier $K = (I, S, T, V, H, C, P, N)$, every validation-facing consumer has a total boundary route*

$$I \longmapsto S \longmapsto T \longmapsto V \longmapsto H, C, P, N.$$

If a consumer route omits V, then it may read only the runtime-candidate input, checkpoint state, and bounded trace evidence supplied by I, S, T and the local transport, route, provenance, and naming rows H, C, P, N. Such a route does not read host validation, total truth, or an unbounded evaluator.

Proof. Project the rows of Definition 997.1. They give the displayed carrier order I, S, T, V, H, C, P, N, with V singled out as the validation-boundary coordinate and the later rows serving only as transport, continuation route, provenance, and local naming data.

Use Definition 997.2 to bind the validation-boundary row to the local NameCert obligations. The non-escape clause prevents any consumer from replacing V by host validation, total truth, or an unbounded evaluator.

Next apply Definition 997.4 and Definition 997.5. The first forces every checkpoint read through I and S, while the second identifies T as bounded trace evidence before any validation-facing row is reached.

Finally apply Definition 997.6 and Definition 997.7. Together they make the route through V total for validation-facing consumers and keep the whole boundary package finite, local, and exhausted by I, S, T, V, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier validation boundary totality`

Theorem 997.9 (ReflectionRuntimeCheckpoint validation row precedence)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, any validation-facing checkpoint consumer must read the displayed rows* I, S, T, V *in that order:* I *supplies the admitted runtime-candidate input,* S *supplies the checkpoint state reached by that input,* T *supplies bounded trace evidence for that state, and only then* V *supplies the validation-boundary row. In particular, neither* S *nor* T *may be treated as a validation row.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier validation row precedence`

Proof. Project the carrier of Definition 997.1. Its first four rows are I, S, T, V, with I typed as the admitted candidate row, S as the checkpoint-state row, T as trace evidence, and V as the validation-boundary row.

Apply Definition 997.4. It forces every checkpoint consumer to read I and S before any trace-evidence or validation-facing row is available.

Apply Definition 997.5 to identify T as bounded evidence rather than a validation act. Then apply Definition 997.6: every validation-facing read must pass through V, so S and T cannot serve as validation substitutes. □

Theorem 997.10 (ReflectionRuntimeCheckpoint trace evidence finiteness)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the row* T *is bounded trace evidence obtained through the finite trace evaluator and finite trace gap-socket route recorded by* P *and consumed through* H, C, N*. It cannot be promoted to an unbounded evaluator or to a detached validation source.*

Proof. Use Definition 997.5. It already routes T through the finite trace evaluator, finite trace gap socket, package provenance, transport, continuation, and local naming rows.

Next use Definition 997.7. The root-unblock package gives downstream consumers exactly I, S, T, V, H, C, P, N and no unlisted evaluator, validation act, or truth coordinate.

Thus the only role available to T inside the packet is bounded evidence for the checkpoint state. Since neither the carrier nor the root-unblock package contains an unbounded evaluator coordinate, no consumer can promote T beyond the finite route displayed by the sibling rows. □

Theorem 997.11 (ReflectionRuntimeCheckpoint audit-boundary consumer lock)**.** *Every downstream* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *checkpoint consumer receives only the finite packet rows* I, S, T, V, H, C, P, N *together with the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, reflection-limit boundary,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows cited by* P*. The consumer receives no host-valid runtime judgment, total truth predicate, unbounded evaluator, or validation coordinate outside that displayed boundary.*

Proof. Start with Definition 997.7. It identifies I, S, T, V, H, C, P, N and their finite routes as the complete root-facing packet offered to downstream runtime-checkpoint consumers.

Apply Definition 997.13. The provenance row P exhausts the sibling route through the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, reflection-limit boundary, and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 997.6 and Definition 997.14. Validation-facing reads must pass through V, and the remaining packet rows export neither host validation nor total truth. Hence every downstream consumer is locked to the finite audit boundary displayed in the carrier. □

Theorem 997.12 (ReflectionRuntimeCheckpoint state-trace coupling)**.** *For every accepted* ReflectionRuntimeCheck *carrier $K = (I, S, T, V, H, C, P, N)$, the checkpoint state row S and trace-evidence row T are consumed together before a validation-facing read reaches V. A consumer may not read S as a detached runtime state or read T as a detached validation act.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier state trace coupling`

Proof. Project the carrier of Definition 997.1. The rows S and T are adjacent checkpoint coordinates of the same finite packet, following the admitted runtime-candidate input row I and preceding the validation-boundary row V.

Apply Definition 997.2. Its checkpoint-state and trace exactness clause binds every displayed state read to the trace-evidence row, while its validation-boundary clause keeps host validation behind V.

Finally use Definition 997.4 and Definition 997.5. The first forces the input and state rows before trace or validation rows, and the second keeps T as bounded evidence rather than an unbounded evaluator or detached validation act. □

Theorem 997.13 (ReflectionRuntimeCheckpoint audit provenance exhaustion)**.** *For every accepted* ReflectionRuntimeCheckpoint$^{\uparrow}$ *carrier $K = (I, S, T, V, H, C, P, N)$, the provenance row P exhausts the runtime-candidate boundary, finite-trace evaluator, finite-trace gap socket, and reflection-limit boundary sibling route. The checkpoint packet has no additional provenance row for a host-valid runtime judgment, total truth predicate, or unbounded evaluator.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier audit provenance exhaustion`

Proof. Read the provenance coordinate in Definition 997.1. It cites exactly the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, reflection-limit boundary, and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 997.5. The trace row T is routed through those same sibling surfaces and then back through H, C, P, N, so the audit path is the displayed finite route rather than an external evaluator.

Use Definition 997.7 to close the package. Since downstream consumers receive exactly I, S, T, V, H, C, P, N, no second provenance coordinate remains outside P for host validation, total truth, or an unbounded runtime evaluator. □

Theorem 997.14 (ReflectionRuntimeCheckpoint host-validation nonescape)**.** *No accepted* ReflectionRuntimeCheck *packet exports a host-valid runtime judgment, total truth predicate, or unbounded evaluator outside the displayed rows*

$$I, S, T, V, H, C, P, N.$$

Every validation-facing consumer must pass through V, with S, T, H, C, P, N serving only as checkpoint state, bounded trace evidence, transport, route, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier host validation nonescape`

Proof. Begin with the carrier of Definition 997.1. It explicitly lists I, S, T, V, H, C, P, N and states that no detached host-validation coordinate, total truth row, or unbounded evaluator is present.

Apply Definition 997.6. Any validation-facing read must pass through V, and no route through I, S, T, H, C, P, N alone exports host validation or total truth.

The state-trace coupling of Definition 997.12 and audit provenance exhaustion of Definition 997.13 account for the remaining rows. Thus all checkpoint information is consumed through the finite packet, with no host-validation escape outside the displayed surface. □

Theorem 997.15 (ReflectionRuntimeCheckpoint validation boundary order)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, any validation-facing read first exposes the runtime-candidate input row* I*, the checkpoint-state row* S*, and the bounded trace-evidence row* T*, and only then passes through the validation-boundary row* V*. Thus a consumer cannot reach a host-validation-facing boundary by reading* V *without the displayed input, state, and trace rows.*

Proof. Project the carrier of Definition 997.1. The displayed checkpoint order places the admitted runtime-candidate row I, checkpoint-state row S, bounded trace row T, and validation-boundary row V as the validation-facing prefix of K.

Apply the NameCert obligations of Definition 997.2. Runtime-candidate exposure binds reads to I, checkpoint-state exactness binds state reads to S, trace exactness binds supporting observations to T, and validation-boundary non-escape binds host-validation-facing access to V.

Finally use Definition 997.6 and Definition 997.14. Together they keep every validation-facing consumer inside the displayed order I, S, T, V and prevent any later transport, route, provenance, or naming row from replacing that boundary order. □

Theorem 997.16 (ReflectionRuntimeCheckpoint bounded trace gap-socket compatibility)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the trace row* T *is compatible with the finite trace evaluator and finite trace gap-socket siblings cited by* P*. It is consumed only as bounded trace evidence and cannot be reread as an unbounded evaluator, a host validator, or a detached truth row.*

Proof. Read the trace row from the carrier of Definition 997.1. The row T is explicitly trace evidence for the checkpoint state, while P cites the finite trace evaluator and finite trace gap-socket siblings as the available trace sources.

Apply Definition 997.5. It routes T through the sibling finite evaluator and gap-socket surfaces and back through H, C, P, N, so the trace route is a bounded readback route rather than an evaluator coordinate.

Use Definition 997.13 to exhaust the cited sibling route. Since the provenance row has no further source for host validation, total truth, or an unbounded evaluator, compatibility with the finite siblings is the whole trace obligation. □

Theorem 997.17 (ReflectionRuntimeCheckpoint runtime-candidate validation refusal ledger)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the validation-boundary row* V *records the refusal of host validation for the runtime-candidate checkpoint until an external bridge certificate is supplied. The refusal is attached to the candidate input* I*, checkpoint state* S*, trace evidence* T*, and audit route* H, C, P, N*, and it is not itself a host-valid runtime judgment.*

Proof. Start with the carrier and NameCert obligations, Definition 997.1 and Definition 997.2. They bind the runtime candidate to I, its checkpoint state to S, its bounded trace evidence to T, and the validation-facing boundary to V.

Apply Definition 997.6. The only validation-facing row is V, and every read through the remaining packet rows fails to export host validation or total truth.

Finish with Definition 997.13 and Definition 997.14. The audit route accounts for H, C, P, N, while host-validation nonescape keeps the refusal ledger distinct from any external bridge certificate that would be needed to assert host-valid runtime validation. □

Theorem 997.18 (ReflectionRuntimeCheckpoint validation prefix exhaustion)**.** *For every accepted* ReflectionRuntimeCheckpoint$^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, each validation-facing read exposes the runtime-candidate input row* I*, checkpoint-state row* S*, bounded trace row* T*, and validation-boundary row* V *before any transport, continuation, provenance, or local naming row* H, C, P, N *is consumed. No validation consumer reaches* H, C, P, N *as a substitute for that prefix.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier validation prefix exhaustion`

Proof. Project the carrier by Definition 997.1. The validation prefix of the packet is exactly I, S, T, V, followed by the transport, route, provenance, and naming rows.

Apply Definition 997.2. Runtime-candidate exposure, checkpoint-state exactness, trace exactness, and validation-boundary non-escape are read before the structural rows can replay or name the packet.

Use Definition 997.6. It prevents H, C, P, N from acting as a detached validation source, so every validation-facing route is exhausted by the displayed prefix before those rows are consumed. □

Theorem 997.19 (ReflectionRuntimeCheckpoint trace provenance lock)**.** *For every accepted* ReflectionRuntimeChe *carrier* $K = (I, S, T, V, H, C, P, N)$*, the trace row* T *is locked to the runtime candidate* I*, the finite-trace evaluator, the finite-trace gap socket, and the reflection-limit provenance cited by* P*. A consumer may replay* T *only through this finite provenance route and not as a detached host validator, total truth predicate, or unbounded evaluator.*

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointCarrier trace provenance lock`

Proof. Read T from Definition 997.1. The trace row is bundled with the runtime candidate I and checkpoint state S, and the provenance row P names the finite trace siblings and reflection-limit source used by the route.

Apply Definition 997.13. It exhausts the cited trace route through the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, and reflection-limit provenance.

Finally use Definition 997.6 and Definition 997.14. These prevent the locked trace row from being reread as host validation, total truth, or unbounded evaluation. □

Theorem 997.20 (ReflectionRuntimeCheckpoint host-validation refusal boundary)**.** *For every accepted* ReflectionRuntimeCheckpoint$^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the validation-boundary row* V *records refusal of a host-valid runtime judgment until an external bridge certificate is present. The refusal is carried by* I, S, T, H, C, P, N *and is never itself exported as host validation, total truth, or an unbounded runtime evaluator.*

Proof. Start from Definition 997.17. It attaches the refusal ledger to the runtime candidate, checkpoint state, trace evidence, validation boundary, and audit route.

Use Definition 997.2. The local NameCert surface records V as the validation-boundary row, while H, C, P, N only transport, replay, package, and name that same finite packet.

Finish with Definition 997.14. Since no host-valid runtime judgment is present outside the displayed rows, the refusal boundary remains a bridge-facing boundary rather than an exported validation theorem. □

Theorem 997.21 (ReflectionRuntimeCheckpoint validation trace scope row)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, every validation-facing read must pass through the input row* I*, the checkpoint-state row* S*, the bounded trace row* T*, and only then the validation-boundary row* V*. The local transport, route, provenance, and naming rows* H, C, P, N *may carry that read, but they do not supply a detached validation source.*

Proof. Begin with the input-state exposure theorem Definition 997.4. It forces checkpoint consumers through I and S before any trace-evidence or validation-facing row is available.

Next apply the trace-boundary readback of Definition 997.5. The row T is bounded trace evidence routed through the finite trace evaluator and finite trace gap-socket siblings, not an unbounded evaluator or detached validation act.

Finally use Definition 997.6 and Definition 997.15. They place every validation-facing read at V after the displayed input, state, and trace rows, with H, C, P, N serving only as the finite route ledger. □

Theorem 997.22 (ReflectionRuntimeCheckpoint provenance consumer row)**.** *Every downstream* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer route uses the provenance row* P *through the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, reflection-limit boundary, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No second provenance row for host validation, total truth, or unbounded evaluation is exported.*

Proof. Start from the audit-boundary consumer lock of Definition 997.11. It gives each downstream consumer exactly the displayed checkpoint rows and the sibling route cited by P.

Apply Definition 997.13. The provenance row already exhausts the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, reflection-limit boundary, and local structural rows.

Finish with Definition 997.7. The downstream consumer receives the finite checkpoint packet and no additional root-facing validation surface, so there is no second provenance source outside P. □

Theorem 997.23 (ReflectionRuntimeCheckpoint host-boundary obligation row)**.** *No* ReflectionRuntimeCheckpoint *packet exports a host-valid runtime judgment, total truth predicate, or unbounded evaluator outside the displayed rows*

$$I, S, T, V, H, C, P, N.$$

Any host-boundary obligation available to a downstream consumer is the finite validation boundary already routed through I, S, T, V, H, C, P, N*.*

Proof. First use the validation trace scope row of Definition 997.21. It forces validation-facing access through I, S, T, V and prevents H, C, P, N from acting as independent validation sources.

Next apply the provenance consumer row of Definition 997.22. This exhausts the sibling route through the runtime-candidate, finite evaluator, gap-socket, and reflection-limit boundaries.

Finally use the host-validation nonescape theorem Definition 997.14. Since no listed row exports host validation, total truth, or an unbounded evaluator, the only available host-boundary obligation is the finite boundary displayed by I, S, T, V, H, C, P, N. □

Theorem 997.24 (ReflectionRuntimeCheckpoint route budget interface)**.** *Every downstream* ReflectionRuntimeChe *consumer route factors through the displayed budget interface*

$$I, S, T, V, H, C, P, N$$

before any validation-facing read is available. The route reads the runtime-candidate input, checkpoint state, bounded trace evidence, validation boundary, componentwise transport, continuation, provenance, and local naming rows as one finite checkpoint packet.

Proof. Begin with the carrier and NameCert obligations, Definition 997.1 and Definition 997.2. They list I, S, T, V, H, C, P, N as the displayed packet and bind candidate input, checkpoint state, trace evidence, validation boundary, transport, continuation, provenance, and naming to those rows.

Use the validation trace scope row and provenance consumer row, Definition 997.21 and Definition 997.22. Together they force validation-facing access through I, S, T, V and route the remaining consumer data through H, C, P, N.

The host-boundary obligation row then prevents a consumer from reading host validation, total truth, or an unbounded evaluator outside the displayed packet. Hence every validation-facing route first factors through the finite budget interface. □

Theorem 997.25 (ReflectionRuntimeCheckpoint trace-validation separation)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the bounded trace-evidence row* T *and the validation-boundary row* V *remain distinct under field faithfulness. A consumer may route* T *into a validation-facing read only by passing through* V*; it cannot exchange trace evidence for validation boundary data.*

Proof. Read T and V from the carrier of Definition 997.1. The row T records bounded trace evidence, while V records the validation boundary.

Apply the trace-boundary readback theorem and the validation non-escape route, Definition 997.5 and Definition 997.6. The first keeps T as bounded evidence routed through the finite trace siblings, and the second requires every validation-facing read to pass through V.

Finally use the TasteGate obligation of Definition 997.3. Field faithfulness preserves the displayed coordinates, so T and V cannot be exchanged by a consumer route. □

Theorem 997.26 (ReflectionRuntimeCheckpoint host-boundary refusal)**.** *The* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *packet refuses host-valid runtime judgment, total truth, and unbounded evaluator rows unless those rows are supplied as separate boundary data outside the finite checkpoint packet. No route through* I, S, T, V, H, C, P, N *exports such data by itself.*

Proof. Start from the host-boundary obligation row, Definition 997.23. It states that the finite packet exports only the displayed validation boundary and no host-valid runtime judgment, total truth predicate, or unbounded evaluator.

Next use the validation trace scope row and trace-validation separation theorem. They place validation-facing reads after I, S, T, V and keep bounded trace evidence distinct from the validation boundary.

The host-validation nonescape theorem, Definition 997.14, closes the refusal. Since the local carrier, NameCert obligations, state-trace coupling, and provenance route account for all rows of

I, S, T, V, H, C, P, N, any host-valid judgment, total truth row, or unbounded evaluator must be displayed as separate boundary data. □

Falsifiable prediction: A checkpoint consumer that omits the validation-boundary row cannot export a host-valid runtime judgment or total truth row.

Independence witness: RuntimeCandidateBoundary, FiniteTraceEvaluator, FiniteTraceGapSocket, ReflectionLimitBoundary: the checkpoint carrier has a displayed checkpoint state and validation-boundary pair not bijective to candidate-only, evaluator-only, gap-socket, or pure boundary packets.

Theorem 997.27 (ReflectionRuntimeCheckpoint scoped source register)**.** *For every accepted* ReflectionRuntimeChe *carrier $K = (I, S, T, V, H, C, P, N)$, the scoped source register for validation-facing reads is exactly*

$$I, S, T, V.$$

The rows H, C, P, N are not additional source rows: H transports the displayed checkpoint rows by hsame*, C exposes their* Cont *routes, P records the* Pkg *provenance through the runtime-candidate, finite-trace, finite-gap, and reflection-limit siblings, and N names the local* $\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}$ *surface.*

Proof. Project the carrier in Definition 997.1. Its first four rows are the admitted runtime-candidate input I, checkpoint state S, bounded trace evidence T, and validation-boundary row V.

Apply Definition 997.21. It makes those four rows the validation-facing source route and prevents H, C, P, N from acting as detached validation sources.

Use Definition 997.22. The remaining rows are transport, continuation, provenance, and local naming rows for the same finite packet, so the scoped source register is exactly I, S, T, V. □

Theorem 997.28 (ReflectionRuntimeCheckpoint route source totality)**.** *Every validation-facing* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *consumer factors through the finite route*

$$I \longmapsto S \longmapsto T \longmapsto V \longmapsto H, C, P, N.$$

If V is absent from a consumer route, then the route can read only the runtime-candidate input, checkpoint state, bounded trace evidence, transport, continuation, provenance, and local naming rows; it cannot produce a host-valid runtime judgment, total truth predicate, or unbounded evaluator.

Proof. Begin with Definition 997.8. It states that every validation-facing route reaches V after reading I, S, T and before returning through H, C, P, N.

Next apply Definition 997.27. The only scoped source rows are I, S, T, V, while H, C, P, N carry the route and naming data for those rows.

Finally use Definition 997.14. Without the validation-boundary row V, the remaining packet has no coordinate from which a host-valid runtime judgment, total truth predicate, or unbounded evaluator can be exported. □

Theorem 997.29 (ReflectionRuntimeCheckpoint scoped closure export)**.** *The scoped paper surface of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *is the finite package consisting of the carrier $K = (I, S, T, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}$ *obligations, input-state exposure, trace-boundary readback, validation non-escape, the scoped source register, route source totality, and the root-unblock package. This surface exports a downstream runtime-checkpoint route through I, S, T, V, H, C, P, N and no host runtime validation, total truth row, unbounded evaluator, or detached validation coordinate.*

Proof. Start with the carrier and NameCert obligations, Definition 997.1 and Definition 997.2. They supply the finite packet and bind the local certificate obligations to I, S, T, V, H, C, P, N.

Use input-state exposure and trace-boundary readback, Definition 997.4 and Definition 997.5. These theorems force reads through I, S and keep T as bounded evidence routed through the finite trace siblings.

Next apply validation non-escape, the scoped source register, and route source totality. Together they place every validation-facing read at V after I, S, T and make H, C, P, N the transport, continuation, provenance, and naming rows rather than extra validation sources.

Finish with Definition 997.7. The downstream consumer receives the finite runtime-checkpoint package already displayed by the carrier, so the scoped paper surface exports no host runtime validation, total truth row, unbounded evaluator, or detached validation coordinate. □

Theorem 997.30 (ReflectionRuntimeCheckpoint TasteGate field-faithful row)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, the displayed rows*

$$I, S, T, V, H, C, P, N$$

are the exact field-faithful readback surface. Equality of the eight readback rows is equality of checkpoint packets, and no consumer may identify the runtime input row, checkpoint-state row, bounded trace row, validation-boundary row, transport row, continuation row, provenance row, or local naming row with any other coordinate.

Proof. Project the carrier in Definition 997.1. It has exactly the eight displayed coordinates I, S, T, V, H, C, P, N, so every public read begins by selecting one of those packet fields.

Apply the NameCert obligations and scoped closure export, Definition 997.2 and Definition 997.29. They bind each consumer route to the same finite checkpoint packet and prevent the route from adding a detached validation coordinate.

Finally use the host-validation nonescape theorem, Definition 997.14. Since no host-valid runtime judgment, total truth row, or unbounded evaluator is available outside the packet, field-faithful readback is exhausted by the eight displayed rows. □

Theorem 997.31 (ReflectionRuntimeCheckpoint nontrivial witness row)**.** *The empty runtime input packet*

$$K_0 = (\mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$$

and the one-step runtime input packet

$$K_1 = (\mathsf{E0}(\mathsf{e}), \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e}, \mathsf{e})$$

are separated by the runtime input row of $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$*. Thus the checkpoint carrier is not vacuous, even before any host-validation boundary is bridged.*

Proof. Read both packets through the carrier definition. They agree on checkpoint state, bounded trace evidence, validation boundary, transport, continuation, provenance, and local naming, while their runtime input rows differ.

The NameCert obligations make I the first admitted runtime-candidate input row, and the scoped closure export keeps this row in the public checkpoint surface. Therefore a consumer that reads the runtime input row distinguishes K_0 from K_1.

Host-validation nonescape ensures that this separation is purely a carrier fact. It does not depend on host runtime validation, total truth, or an unbounded evaluator, so it witnesses nontriviality inside the displayed BEDC packet. □

Theorem 997.32 (ReflectionRuntimeCheckpoint runtime readback exhaustion)**.** *Every runtime-checkpoint consumer read factors through the same eight displayed rows*

$$I, S, T, V, H, C, P, N.$$

The runtime input, checkpoint state, bounded trace evidence, validation boundary, componentwise transport, continuation routes, package provenance, and local name exhaust the consumer readback surface; no ninth row exports host validation, total truth, an unbounded evaluator, or a detached validation coordinate.

Proof. Begin with the scoped closure export, Definition 997.29. It identifies the public surface as the finite package consisting of carrier admission, NameCert obligations, input-state exposure, trace-boundary readback, validation non-escape, scoped source register, route source totality, and the root-unblock package.

Apply the carrier definition and NameCert obligations. The carrier supplies exactly I, S, T, V, H, C, P, N, while the obligations bind runtime input, checkpoint state, trace evidence, validation boundary, transport, route, provenance, and naming to those rows.

Finally use host-validation nonescape. Since host validation, total truth, unbounded evaluation, and detached validation have no packet coordinate, every consumer read factors through the eight-row checkpoint surface and stops there. □

Theorem 997.33 (ReflectionRuntimeCheckpoint candidate-trace-validation chain)**.** *For every accepted* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *carrier* $K = (I, S, T, V, H, C, P, N)$*, each validation-facing read factors through the ordered checkpoint chain*

$$I \longmapsto S \longmapsto T \longmapsto V \longmapsto H, C, P, N.$$

The read first exposes the runtime-candidate input I*, then the checkpoint state* S*, then the bounded trace evidence* T*, then the validation boundary* V*; only after those four rows are fixed may it return through the structural transport, continuation, provenance, and local naming rows* H, C, P, N*.*

Proof. Start with the carrier definition, Definition 997.1. It displays the checkpoint packet as the finite tuple $K = (I, S, T, V, H, C, P, N)$, so every validation-facing read has a visible row from which it must begin.

Apply input-state exposure and trace-boundary readback, Definition 997.4 and Definition 997.5. These force the route to pass through the runtime-candidate input and checkpoint state before using T as bounded trace evidence.

Finally use the validation non-escape route, Definition 997.6. It places the validation-facing boundary at V and leaves H, C, P, N as the structural rows for transport, continuation, provenance, and local naming, so the displayed ordered chain exhausts the read. □

Theorem 997.34 (ReflectionRuntimeCheckpoint local validation scope)**.** *For every accepted* ReflectionRuntimeChe *carrier* $K = (I, S, T, V, H, C, P, N)$*, the validation row* V *is local to the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, and reflection-limit boundary recorded by* P*. It does not export a host-valid runtime judgment, a total truth row, or an unbounded evaluator coordinate.*

Proof. Begin with audit provenance exhaustion, Definition 997.13. The provenance row P records the runtime-candidate, finite-trace, finite-gap, and reflection-limit siblings that the checkpoint packet is allowed to cite.

Next apply host-validation nonescape, Definition 997.14. The validation row is therefore a local boundary row of the packet, not a source for host runtime validity, total truth, or unbounded evaluation.

Use the audit-boundary consumer lock, Definition 997.11. Since consumers are locked to the displayed checkpoint boundary and its recorded provenance, the validation scope stays local to the rows named by P. □

Theorem 997.35 (ReflectionRuntimeCheckpoint root consumer exhaustion)**.** *Every downstream root consumer of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *receives exactly the rows*

$$I, S, T, V, H, C, P, N$$

from the finite checkpoint packet. No root consumer receives a detached validation row, hidden evaluator row, host-valid runtime judgment, or total-truth row outside this displayed surface.

Proof. First apply the candidate-trace-validation chain, Definition 997.33. It forces every validation-facing read to pass through I, S, T, V before returning through H, C, P, N.

Next use the local validation scope, Definition 997.34. The validation boundary remains tied to the runtime-candidate, finite-trace, finite-gap, and reflection-limit provenance recorded by the packet.

Finally compose the root-unblock package with runtime readback exhaustion, Definition 997.7 and Definition 997.32. These results deliver the same eight rows to downstream root consumers and rule out any detached validation, hidden evaluator, host-valid runtime judgment, or total-truth coordinate. □

Theorem 997.36 (ReflectionRuntimeCheckpoint root carrier totality)**.** *Every downstream root consumer of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *first exposes exactly the finite carrier*

$$K = (I, S, T, V, H, C, P, N)$$

as displayed BHist *rows before any runtime, trace, validation, or* NameCert *read is consumed. The rows are respectively the runtime-candidate input, checkpoint state, bounded trace evidence, validation boundary, componentwise transport, continuation route, package provenance, and local name; no root consumer begins from a hidden evaluator row, host-validation row, total-truth row, or detached naming coordinate.*

Proof. Start with the carrier definition, Definition 997.1, and the NameCert obligations, Definition 997.2. Together they display the eight rows I, S, T, V, H, C, P, N as the complete checkpoint packet.

Next apply trace-boundary readback and validation non-escape, Definition 997.5 and Definition 997.6. These force trace-facing and validation-facing reads to pass through the displayed trace evidence and validation boundary rather than an ambient host row.

Finally use Definition 997.35. Since downstream root consumers receive exactly the same eight-row surface, their first carrier exposure is precisely K and no additional runtime, trace, validation, or NameCert source remains. □

Theorem 997.37 (ReflectionRuntimeCheckpoint root validator boundary)**.** *Every validation-facing read of a downstream root consumer must pass through the displayed validation boundary row* V *of*

$$K = (I, S, T, V, H, C, P, N).$$

The rows I, S, T, H, C, P, N may expose the candidate input, checkpoint state, bounded trace evidence, transport, continuation, provenance, and local naming, but none of them supplies the validation boundary or a host-valid runtime judgment.

Proof. Begin with Definition 997.36. It places every root consumer on the displayed carrier $K = (I, S, T, V, H, C, P, N)$ before the consumer reads validation-facing data.

Apply the trace-boundary readback, Definition 997.5. This fixes T as bounded trace evidence, so the trace row cannot be reused as a validation row.

Now use the validation non-escape route, Definition 997.6. Validation-facing access is localized at V, while I, S, T, H, C, P, N remain the non-validation carrier, transport, route, provenance, and naming rows.

Finally compose with Definition 997.35. The downstream root surface contains no detached validation coordinate and no host-valid runtime judgment, so every root validation read must pass through V. □

Theorem 997.38 (ReflectionRuntimeCheckpoint root ledger exactness)**.** *All downstream root ledger reads of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *are exhausted by the bounded trace evidence T, the validation boundary V, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and local* NameCert *rows of*

$$K = (I, S, T, V, H, C, P, N).$$

No ledger read imports host validation, total truth, an unbounded evaluator, a detached validation coordinate, or any row outside the finite checkpoint packet.

Proof. First use Definition 997.36. It gives the root consumer exactly the displayed carrier rows and fixes the finite packet in which ledger reads are interpreted.

Next apply Definition 997.37. Validation-facing ledger access must pass through V, while trace-facing evidence is supplied by T through Definition 997.5.

Use the NameCert obligations and the validation non-escape route, Definition 997.2 and Definition 997.6. These bind componentwise hsame, displayed Cont, Pkg provenance, and local NameCert support to the finite packet and block host validation or total truth from entering the ledger.

Finally invoke Definition 997.35. Since downstream root consumers receive no row beyond K, the listed trace, validation, transport, route, provenance, and naming data exhaust every root ledger read. □

Theorem 997.39 (ReflectionRuntimeCheckpoint scoped carrier exposure)**.** *Every scoped downstream root consumer of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *first exposes the displayed finite carrier*

$$K = (I, S, T, V, H, C, P, N)$$

before reading any runtime-candidate, checkpoint-state, trace, validation, transport, continuation, provenance, or naming data. The first consumer-visible object is therefore the same eight-row packet recorded by the carrier, the local NameCert *obligations, the root-unblock package, and the TasteGate carrier row; no scoped root consumer begins from a hidden evaluator, host-validation row, total-truth row, or detached name.*

Proof. Start with the carrier and local certificate obligations, Definition 997.1 and Definition 997.2. They display $K = (I, S, T, V, H, C, P, N)$ and make carrier admission the first local obligation.

Apply the root-unblock package together with root consumer exhaustion, Definition 997.7 and Definition 997.35. These results route downstream consumers through the same finite checkpoint surface and deny any extra host-validation, total-truth, unbounded-evaluator, or detached-validation coordinate.

Finally use root carrier totality and the TasteGate row, Definition 997.36 and Definition 997.3. Field-faithful carrier alignment keeps the root consumer's first exposure equal to the displayed packet K. □

Theorem 997.40 (ReflectionRuntimeCheckpoint scoped validation boundary)**.** *Every validation-facing read of a scoped downstream root consumer of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *passes through the validation-boundary row* V *of*

$$K = (I, S, T, V, H, C, P, N).$$

The checkpoint-state row S *and trace-evidence row* T *may expose the displayed state and bounded evidence, but neither row can replace* V *as the validation boundary, and neither row exports a host-valid runtime judgment.*

Proof. Begin with the carrier and NameCert obligations, Definition 997.1 and Definition 997.2. They separate checkpoint state, bounded trace evidence, and validation boundary into the distinct rows S, T, V.

Use the root-unblock package and root carrier totality, Definition 997.7 and Definition 997.36. A scoped root consumer must therefore read the displayed carrier before any validation-facing access is available.

Now apply the root validator boundary and root ledger exactness, Definition 997.37 and Definition 997.38. They force validation-facing access through V and keep T, H, C, P, N as trace, transport, route, provenance, and naming ledger data.

Finally invoke the TasteGate row, Definition 997.3. Field faithfulness prevents S or T from being identified with V, so the validation boundary remains the separate row required by the scoped packet. □

Theorem 997.41 (ReflectionRuntimeCheckpoint scoped ledger exhaustion)**.** *Every ledger read of a scoped downstream root consumer of* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *is exhausted by*

$$T, V, H, C, P, N$$

together with the runtime-candidate boundary, finite trace evaluator, finite trace gap socket, and reflection-limit boundary cited by the provenance row P*. The ledger therefore has no ninth coordinate for host validation, total truth, an unbounded evaluator, or validation outside the displayed checkpoint packet.*

Proof. First expose the carrier and its certificate surface, Definition 997.1 and Definition 997.2. The packet supplies T, V, H, C, P, N as the trace, validation, transport, route, provenance, and naming rows available for ledger reads.

Next use the root-unblock package and root consumer exhaustion, Definition 997.7 and Definition 997.35. They keep downstream root consumers inside the finite checkpoint packet and remove any detached ledger coordinate.

Apply root carrier totality, root validator boundary, and root ledger exactness, Definition 997.36, Definition 997.37, and Definition 997.38. These results identify T as bounded evidence, V as the validation boundary, and H, C, P, N as the exact structural ledger rows.

Finally read the provenance citations through the TasteGate row, Definition 997.3. The cited runtime-candidate, finite-trace, finite-gap, and reflection-limit sibling rows are exactly the sources named by P, so the ledger is exhausted by the displayed scoped surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}$.

Constructive story: A $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ packet is generated from a runtime-candidate input row, checkpoint-state row, trace-evidence row, validation-boundary row, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 997.39, Definition 997.40, and Definition 997.41 bind scoped root consumers to the displayed carrier, validation boundary, and exact ledger surface, while Definition 997.1, Definition 997.2, Definition 997.7, Definition 997.35, Definition 997.36, Definition 997.37, Definition 997.38, and

Lean4 ✓ `BEDC.Derived.ReflectionRuntimeCheckpointUp.taste gate`

provide the finite runtime-checkpoint carrier and TasteGate grounding.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.ReflectionRuntimeCheckpointUp.ReflectionRuntimeCheckpointTasteGate\_single\_carrier\_al`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not provide host runtime validation, a total truth predicate, an unbounded evaluator, or bridge checking for $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$.

Upgrade path: No further upgrade – chapter at top of the formal axis.

Chapter 998

A Concrete Naming Certificate for $\mathsf{QuotientStreamRefusal}^{\uparrow}$

The $\mathsf{QuotientStreamRefusal}^{\uparrow}$ packet records the finite Real-completion boundary that refuses quotient-stream equality unless the request first factors through displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows. It is a non-escape certificate for completion-facing consumers, not a quotient construction and not a selected-limit principle.

The refusal packet is a consumer of chapter 126, chapter 301, chapter 386, and chapter 38 in that order. Finite $\mathsf{StreamName}^{\uparrow}$ windows feed $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyLimitSeal}^{\uparrow}$ binds the regular readback to a seal route, and $\mathsf{Real}^{\uparrow}$ receives only that displayed seal surface. This dependency is a horizontal L10 lattice edge, not a quotient construction, selected-limit principle, host real equality, or ambient completeness claim.

Definition 998.1 (QuotientStreamRefusal carrier)**.** *A* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, L, E, F, H, C, P, N)$$

with five displayed route rows and four structural rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row requested by the completion-facing consumer;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row served only after the window row S has been exposed;*
3. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route row binding the regular readback to the seal surface;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, L have been consumed;*
5. *F is the refusal ledger row recording that quotient-stream equality, selected limits, host real equality, and ambient completeness are not exported by this packet;*
6. *H records componentwise* hsame *transports for S, R, L, E, F and the local naming rows;*
7. *C records the displayed* Cont *routes from S to R, from R to L, from L to E, and from any quotient-stream request to the refusal row F;*
8. *P records* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.QuotientStreamRefusalUp.QuotientStreamRefusalUp`

Theorem 998.2 (QuotientStreamRefusal NameCert surface)**.** *For every accepted* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *carrier* $Q = (S, R, L, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$ *surface has exactly five route obligations: finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route exposure through* L*,* $\mathsf{Real}^{\uparrow}$ *seal exposure through* E*, and quotient-stream refusal through* F*. The structural rows* H, C, P, N *transport, replay, package, and name those five obligations, and no quotient stream identity, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate is exported.*

Proof. Project the carrier of Definition 998.1. The only route rows are S, R, L, E, F, and every structural read is attached to them through H, C, P, N.

Read the completion route in order. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is exposed only after the $\mathsf{StreamName}^{\uparrow}$ window S, the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row L is exposed only after R, and the $\mathsf{Real}^{\uparrow}$ seal row E is exposed only after L.

Now inspect a quotient-stream request. The displayed continuation route sends that request to F, whose ledger records refusal rather than an equality witness. Since the packet has no quotient, selected-limit, host-equality, hidden-tail, or ambient-completeness coordinate, the listed five obligations exhaust the local certificate surface. □

Theorem 998.3 (QuotientStreamRefusal non-escape route)**.** *Any Real-completion consumer accepted through a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

If the consumer asks instead for quotient-stream equality, selected limits, host real equality, hidden family limits, hidden tail oracles, or ambient completeness, the request is routed to the refusal ledger F *and no completion coordinate is exported.*

Proof. Use Definition 998.2. It orders the accepted route through S, R, L, E and attaches all transport, continuation, provenance, and local naming data to that route.

The refusal row handles every off-route request. A quotient-stream equality request has no carrier coordinate except F, and F is a ledger of refusal rather than a stream identity witness.

Thus every accepted Real-completion read remains on the displayed finite route, while every non-escape boundary request is recorded without exporting quotient equality, selected limits, host real equality, hidden family limits, hidden tail oracles, or ambient completeness. □

Theorem 998.4 (QuotientStreamRefusal route exposure)**.** *Every accepted* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *consumer reads the displayed route rows in the order*

$$S \longmapsto R \longmapsto L \longmapsto E$$

before any $\mathsf{Real}^{\uparrow}$*-facing seal row is public. The structural rows* H, C, P, N *may transport, replay, record provenance, and name that route, but they do not let a consumer expose* R*,* L*, or* E *before the preceding row has been displayed.*

Proof. Project the carrier of Definition 998.1. The route coordinates are listed as the finite window row S, regular readback row R, limit-seal route row L, and Real seal row E.

Apply Definition 998.2. It identifies these four route obligations as the only accepted Real-completion exposure path before the refusal row is consulted.

The structural rows H, C, P, N are attached to the same displayed coordinates. They preserve and replay the route already exposed by S, R, L, E, so no structural projection can publish the $\mathsf{Real}^{\uparrow}$ seal before the ordered route has been read. □

Theorem 998.5 (QuotientStreamRefusal ledger exactness)**.** *Every request for quotient-stream equality, selected limits, host real equality, hidden family limits, hidden tail oracles, or ambient completeness terminates at the refusal row F. The rows H, C, P, N only transport, route, package, and name that refusal; they do not turn the refusal ledger into any of the requested completion coordinates.*

Lean4 ✓ `BEDC.Derived.QuotientStreamRefusalUp.QuotientStreamRefusal refusal ledger changes packet`

Proof. Use Definition 998.3. It sends each off-route completion request to F and records that no quotient, selected-limit, host-equality, hidden-family, hidden-tail, or ambient-completeness coordinate is exported.

Read the remaining rows through the carrier. The transport row H can compare displayed rows componentwise, C can route the request to F, P can record package provenance, and N can name the local packet.

None of those rows is a completion coordinate. Therefore each listed request ends exactly at the refusal ledger, with structural rows recording the refusal rather than discharging it. □

Theorem 998.6 (QuotientStreamRefusal TasteGate obligation surface)**.** *The paper-side obligation surface for the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *TasteGate consists of carrier admission for $Q = (S, R, L, E, F, H, C, P, N)$, route exposure through S, R, L, E, refusal-ledger exactness at F, componentwise field visibility for $S, R, L, E, F, H, C, P, N$, and the nontriviality and field-faithfulness commitments for those displayed rows.*

Proof. Carrier admission is Definition 998.1. It gives the finite packet and names every row on the TasteGate-facing surface.

Route exposure is Definition 998.4. Refusal-ledger exactness is Definition 998.5, together with the non-escape route of Definition 998.3.

The TasteGate paragraph below the independence witness states the remaining commitments: the Lean file will provide nontriviality and field faithfulness for the displayed rows $S, R, L, E, F, H, C, P, N$. Thus the obligation surface is exactly the finite carrier, ordered route, refusal ledger, componentwise field visibility, and the stated TasteGate commitments. □

Theorem 998.7 (QuotientStreamRefusal carrier admission)**.** *Every accepted* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet exposes exactly*

$$Q = (S, R, L, E, F, H, C, P, N).$$

The rows are the finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route,* $\mathsf{Real}^{\uparrow}$ *seal, refusal ledger, componentwise* hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$ *row. No accepted packet carries an additional quotient stream, selected limit, host real equality, hidden family limit, hidden tail oracle, or ambient completeness coordinate.*

Proof. Project the carrier definition. The tuple $Q = (S, R, L, E, F, H, C, P, N)$ lists all route, refusal, transport, replay, provenance, and local naming rows of the packet.

Use the NameCert surface theorem. It identifies the same five route and refusal obligations S, R, L, E, F and attaches the structural rows H, C, P, N to them.

Since no further coordinate is present in the carrier or the local certificate surface, every accepted packet admits exactly the displayed tuple and none of the excluded completion objects. □

Theorem 998.8 (QuotientStreamRefusal route ordering)**.** *Any accepted completion read through a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet consumes* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows in that order:*

$$S \longmapsto R \longmapsto L \longmapsto E.$$

The structural rows H, C, P, N *transport, replay, package, and name this ordered route, but they do not expose* R *before* S, L *before* R, *or* E *before* L.

Lean4 ✓ `BEDC.Derived.QuotientStreamRefusalUp.QuotientStreamRefusalPacket route ordering`

Proof. Start with Definition 998.7. It gives the admitted route rows S, R, L, E together with the structural rows.

Apply Definition 998.4. That theorem reads the same rows in the displayed order and prevents any structural projection from publishing a later row before the preceding one has been consumed.

Thus every accepted completion read follows the finite route from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, with H, C, P, N recording only the visible route. □

Theorem 998.9 (QuotientStreamRefusal terminal route exhaustion)**.** *Every terminal Real-completion read accepted by a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet is exhausted by the displayed rows*

$$S, R, L, E, F, H, C, P, N,$$

where S *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* L *is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route,* E *is the* $\mathsf{Real}^{\uparrow}$ *seal, and* F *is the refusal ledger. The structural rows* H, C, P, N *only transport, replay, record provenance, and name that finite route. After these rows are read, no quotient stream equality, selected limit, host real equality, hidden tail oracle, ambient completeness, or metric-completeness row remains.*

Proof. Start with Definition 998.2. It identifies the displayed carrier surface as the five route and refusal rows S, R, L, E, F together with the structural rows H, C, P, N.

Next apply Definition 998.3 and Definition 998.4. These results force every Real-completion consumer onto the visible route from S through R and L to E, with every off-route completion request landing in F.

Use Definition 998.5 for the terminal refusal boundary. The ledger row records requests for quotient-stream equality, selected limits, host real equality, hidden tail data, and completeness data without exporting any requested coordinate.

Finally apply Definition 998.8. The only completion-facing order is $S \longmapsto R \longmapsto L \longmapsto E$, and the rows H, C, P, N are transport, replay, provenance, and name rows for that order. Thus the displayed nine rows exhaust the accepted terminal read, leaving no quotient, selected-limit, host-equality, hidden-tail, ambient-completeness, or metric-completeness row. □

Theorem 998.10 (QuotientStreamRefusal terminal route naming exactness)**.** *After the terminal route*

$$S, R, L, E, F, H, C, P, N$$

is exhausted for an accepted $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet, the local* $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$ *row names exactly the refusal packet. It does not convert the refusal row* F *into quotient-stream equality, selected limits, host real equality, a hidden tail oracle, ambient completeness, or metric-completeness data.*

Proof. Start with Definition 998.7. It admits only the route rows S, R, L, E, refusal row F, and structural rows H, C, P, N.

Use Definition 998.8 and Definition 998.9. These theorems identify the terminal read as the displayed route together with refusal, transport, replay, provenance, and naming.

Apply Definition 998.5 and Definition 998.2. The ledger remains a refusal row, and the local NameCert surface names that row rather than discharging any excluded completion request. □

Theorem 998.11 (QuotientStreamRefusal off-route nonescape)**.** *If a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *consumer asks for quotient-stream equality, a selected limit, host real equality, hidden family limit, hidden tail data, or ambient completeness, the request lands in the refusal row* F*. The packet records the refusal and exports no completion coordinate outside the ordered route* $S \longmapsto R \longmapsto L \longmapsto E$*.*

Proof. Use Definition 998.7 to identify the only admitted rows. The off-route requests named in the statement are not carrier coordinates.

Apply Definition 998.3. It sends those requests to the refusal ledger and states that no completion coordinate is exported from them.

Finally use Definition 998.8. The only completion-facing route is S, R, L, E, so the refusal row cannot be converted into a quotient stream, selected limit, host equality, hidden data source, or ambient completeness theorem. □

Theorem 998.12 (QuotientStreamRefusal tail-fiber boundary)**.** *Every* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *consumer read that reaches* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *must pass through the displayed route*

$$S \longmapsto R \longmapsto L \longmapsto E.$$

If the consumer asks to interpret the shared tail-fiber row as quotient-stream equality, a selected limit, host real equality, a hidden tail oracle, or ambient completeness, the request lands in the refusal row F*. Thus the tail-fiber row may be consumed only as finite seal-route evidence and cannot discharge any refused completion coordinate.*

Proof. Start with the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ carrier admission and route ordering. By Definition 998.7 and Definition 998.8, every accepted completion read has exactly the displayed carrier rows and reaches the $\mathsf{Real}^{\uparrow}$ seal only through $S \longmapsto R \longmapsto L \longmapsto E$.

Now read the $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ side. The shared seal factorization of Definition 893.4 says that tail-meet and tail-agreement consumers share the finite fiber row before the $\mathsf{Real}^{\uparrow}$ seal row is available. The non-escape theorem Definition 893.5 says that this packet exports no quotient stream equality, host real equality, selected limit, choice selector, or ambient completeness theorem.

Apply the off-route refusal boundary of Definition 998.11. Any attempt to read the shared fiber as quotient-stream equality, a selected limit, host real equality, hidden tail data, or ambient completeness is therefore not a completion witness; it is routed to F.

The only remaining use of the tail-fiber row is finite seal-route evidence feeding the displayed Real route. Hence a $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ consumer can reach $\mathsf{QuotientStreamRefusal}^{\uparrow}$ through S, R, L, E, but it cannot turn the shared row into quotient equality or completeness data. □

Theorem 998.13 (QuotientStreamRefusal regular-tail quotient refusal factorization)**.** *Every* RegularCauchyTailFib *or Real-completion consumer that asks to read the shared tail row as quotient-stream equality must first pass through the* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *refusal-and-*psame*-transport surface of chapter 634. After that boundary has consumed the requested-representative refusal rows and the classifier-respecting* psame *transport replacement, the request lands in the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *refusal row* F *unless the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows have been consumed in order.*

Proof. Start with carrier admission and route ordering for $\mathsf{QuotientStreamRefusal}^{\uparrow}$. By Definition 998.7 and Definition 998.8, the only completion route exposed by the packet is $S \longmapsto R \longmapsto L \longmapsto E$, with F reserved for refused quotient-style reads.

Now consume the quotient-soundness boundary. The transport replacement theorem Definition 634.3 routes any surviving representative-style request through displayed relation and psame transport rows, while Definition 634.8 and Definition 634.6 leave no quotient carrier, representative selector, quotient equality eliminator, or $\mathsf{Quot.sound}$ dependency available to the downstream consumer.

It remains to attach the regular-tail read to the Real-completion refusal packet. By Definition 998.12, a shared tail-fiber row can only feed the displayed seal route. By Definition 998.11, any attempt to read that row as quotient-stream equality is therefore routed to F unless the consumer has already consumed S, R, L, E as the finite completion route. □

Theorem 998.14 (QuotientStreamRefusal obligation closure)**.** *The carrier, local* $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$ *surface, route exposure, ledger exactness, terminal exhaustion, and tail-fiber boundary assemble the concrete obligation closure for* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Every accepted completion-facing read is exhausted by the displayed packet*

$$Q = (S, R, L, E, F, H, C, P, N),$$

with public route $S \longmapsto R \longmapsto L \longmapsto E$ *and refusal ledger* F*. The closure exports no quotient stream equality, selected limit, host real equality, hidden family limit, hidden tail oracle, ambient completeness, or metric-completeness row.*

Proof. Carrier admission is Definition 998.7. It fixes the packet as the finite row tuple $Q = (S, R, L, E, F, H, C, P, N)$ over the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surfaces.

The public route is supplied by Definition 998.4 and Definition 998.8. These theorems force the completion-facing read through $S \longmapsto R \longmapsto L \longmapsto E$ before the $\mathsf{Real}^{\uparrow}$ seal row is exported.

Refusal exactness is supplied by Definition 998.5 and Definition 998.11. Every off-route request reaches F, and the structural rows H, C, P, N only transport, replay, package, and name that refusal.

Terminal exhaustion is Definition 998.9. The tail-fiber compatibility boundary is Definition 998.12. Together they show that both terminal Real-completion reads and tail-fiber consumers are consumed only through the displayed route and cannot reinterpret a shared row as quotient equality or completeness data.

Thus the listed carrier, naming surface, route, ledger, terminal exhaustion, and tail-fiber boundary close the concrete obligation surface. No quotient stream equality, selected limit, host real equality, hidden family limit, hidden tail oracle, ambient completeness, or metric-completeness row is exported. □

Theorem 998.15 (QuotientStreamRefusal TasteGate marker route)**.** *The complete paper-side TasteGate route for* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *consists exactly of carrier admission, route exposure, refusal-ledger exactness, off-route non-escape, the displayed* $\mathsf{Nontrivial}\ \mathsf{QuotientStreamRefusal}^{\uparrow}$ *commitment, and the displayed* $\mathsf{FieldFaithful}\ \mathsf{QuotientStreamRefusal}^{\uparrow}$ *commitment. In carrier coordinates, the route is exhausted by*

$$Q = (S, R, L, E, F, H, C, P, N),$$

with $S \longmapsto R \longmapsto L \longmapsto E$ as the public completion route and F as the refusal ledger for quotient-stream equality, selected limits, host real equality, hidden family limits, hidden tail data, and ambient completeness.

Lean4 ✓ `BEDC.Derived.QuotientStreamRefusalUp.taste gate`

Proof. Carrier admission gives the displayed packet $Q = (S, R, L, E, F, H, C, P, N)$ and states that no additional completion coordinate is admitted.

Route exposure and route ordering identify the public completion path as $S \longmapsto R \longmapsto L \longmapsto E$. The structural rows H, C, P, N only transport, replay, package, and name that path.

Ledger exactness and off-route nonescape route every quotient-stream, selected-limit, host-equality, hidden-data, and ambient-completeness request to F. They do not turn the refusal row into any requested completion witness.

The displayed TasteGate commitment supplies the remaining two marker obligations: nontriviality and field faithfulness for the same finite packet rows. Hence the listed carrier, route, ledger, non-escape, nontriviality, and field-faithfulness obligations form the complete paper-side TasteGate route. □

Theorem 998.16 (QuotientStreamRefusal public bridge boundary)**.** *The public refusal export of* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *bridges to the standard Real-completion boundary exactly through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyLimitSeal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In carrier coordinates this is the packet

$$Q = (S, R, L, E, F, H, C, P, N),$$

where S, R, L, E are the bridge route and F is the refusal ledger. The BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{Nontrivial\ QuotientStreamRefusal}^{\uparrow}$, *and* $\mathsf{FieldFaithful\ QuotientStreamRefusal}^{\uparrow}$ *rows are structural support for that route only. They export no quotient-stream equality, selected limit, host real equality, hidden tail oracle, ambient completeness row, or alternative Real-completion boundary.*

Proof. Carrier admission is Definition 998.7. It fixes the public packet as $Q = (S, R, L, E, F, H, C, P, N)$ and leaves no additional quotient-stream, selected-limit, host-equality, hidden-tail, or completeness coordinate.

Route ordering is Definition 998.8. It forces every accepted completion read through $S \longmapsto R \longmapsto L \longmapsto E$ before the $\mathsf{Real}^{\uparrow}$ seal row is public.

Terminal exhaustion is Definition 998.9. Regular-tail factorization is Definition 998.13. Together they rule out a second terminal read through quotient-stream equality or a hidden tail interpretation of the shared row.

Obligation closure is Definition 998.14. The TasteGate marker route is Definition 998.15. These results assemble the refusal ledger, structural support rows, nontriviality commitment, and field-faithfulness commitment around the same displayed route, so the public bridge boundary is exactly the StreamName, RegSeqRat, CauchyLimitSeal, and RealUp route and nothing outside it. □

Theorem 998.17 (QuotientStreamRefusal Real-completion consumer exhaustion)**.** *Every accepted Real-completion consumer using a* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *packet exhausts its public read through the displayed route*

$$S \longmapsto R \longmapsto L \longmapsto E$$

together with the refusal row F. After composition with the sibling surfaces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *the consumer cannot cite quotient-stream equality, selected limits, host real equality, hidden family limits, hidden tail oracles, or ambient completeness outside that route-and-refusal packet.*

Proof. The NameCert surface theorem fixes the public coordinates of the packet as the route rows S, R, L, E, the refusal ledger F, and the structural support rows H, C, P, N. The non-escape route theorem then sends every off-route Real-completion request to F.

Ledger exactness identifies what F records: quotient-stream equality, selected limits, host real equality, hidden family limits, hidden tail oracles, and ambient completeness are refused rather than exported. Route ordering forces the accepted consumer to read $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in the displayed order.

Finally, the public bridge boundary composes those sibling surfaces over the same carrier packet. Since the bridge exports only $S \longmapsto R \longmapsto L \longmapsto E$ and records all excluded requests at F, the route-and-refusal packet exhausts the consumer's public Real-completion read. □

Falsifiable prediction: Any accepted Real-completion consumer that asks for quotient-stream equality is refused unless it first factors through the displayed StreamName, RegSeqRat, CauchyLimitSeal, and RealUp seal rows.

Independence witness: streamname, regseqrat, cauchylimitseal, realwindowsynchronizer, real: no sibling is bijective because none carries the refusal row together with all four route surfaces and the non-escape ledger.

The Lean file `Derived/QuotientStreamRefusalUp/TasteGate.lean` will provide $\mathsf{Nontrivial}\ \mathsf{QuotientStreamRef}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientStreamRefusal}^{\uparrow}$ instances for the displayed rows $S, R, L, E, F, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{QuotientStreamRefusal}^{\uparrow}$ packet is generated from a StreamName finite-window row, RegSeqRat readback row, CauchyLimitSeal route row, RealUp seal row, quotient-refusal ledger, componentwise transports, displayed continuations, package provenance, and local naming; the TasteGate row is recorded at BEDC.Derived.QuotientStreamRefusalUp.taste_

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 998.1, Definition 998.14, Definition 998.15, and Definition 998.16 display the finite refusal scope, public refusal route, and Real-completion bridge over StreamNameUp, RegSeqRatUp, CauchyLimitSealUp, RealUp, BHist, hsame, Cont, Pkg, NameCert, TasteGate, Nontrivial, and FieldFaithful rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuotientStreamRefusal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.QuotientStreamRefusalUp.QuotientStreamRefusalTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim axiom-purity, bridge checking, quotient streams, selected limits, host real equality, or ambient completeness.

Upgrade path: The next formal-axis step is axiomCleanV to bridgeCheckedV via a checked standard bridge for the existing single-carrier theorem.

Chapter 999

A Concrete Naming Certificate for $\mathsf{TrivialZeroClassifier}^{\uparrow}$

The $\mathsf{TrivialZeroClassifier}^{\uparrow}$ packet realizes the trivial-zero side of the RH witness catalogue as its own finite carrier. It names the negative-even-integer source route before that route is consumed by the non-trivial-zero boundary of chapter 659. The packet is later separated from the fixed-strip witness route of chapter 548: the former binds negative-even source rows, while the latter binds supplied critical-line witness rows and refusal ledgers. The packet reads only the arithmetic source rows of $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, the $\mathsf{ZetaZeros}^{\uparrow}$ zero row supplied by chapter 73, the zeta-continuation socket of chapter 570, rational comparison data, componentwise transports, displayed routes, provenance, and the local naming row.

Falsifiable prediction: Every accepted TrivialZeroClassifierUp read of a negative-even zero factors through IntUp, RatUp, RealUp, ZetaZerosUp, and the local rational comparison ledger before any RH-facing consumer sees it.

Independence witness: zetazeros, zetacontinuationsocket, nontrivialzeroclassifier, criticallinewitness: none is bijective to $\mathsf{TrivialZeroClassifier}^{\uparrow}$ because this packet binds the negative-even integer source, arithmetic embeddings, zeta-zero row, real-part row, rational comparison ledger, and local NameCert row as one trivial-zero classifier, while each sibling omits at least one of those displayed coordinates or consumes the row only after another classifier has already named it.

Definition 999.1 (Trivial-zero-classifier carrier)**.** *A* $\mathsf{TrivialZeroClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, Q, R, Z, E, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the negative-even* $\mathsf{Int}^{\uparrow}$ *source row, displaying a signed integer of the form $-2n$ with the nonempty unary witness for n;*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *embedding row for the same signed source;*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *real-part row obtained from the rational embedding;*
4. *Z is the* $\mathsf{ZetaZeros}^{\uparrow}$ *zero row for the corresponding complex history with empty imaginary part;*
5. *E is the* $\mathsf{ZetaContinuationSocket}^{\uparrow}$ *functional-equation route row that exposes the sine-zero source of the trivial zero;*

6. *A is the local rational comparison ledger recording the negative-even side and separating it from the critical-strip interval side;*
7. *H records componentwise* hsame *transports for I, Q, R, Z, E, A and for the package and local naming coordinates;*
8. *C records the displayed* Cont *routes from the integer source to the rational and real rows, from the continuation socket to the zero row, and from the comparison ledger to RH-facing consumers;*
9. *P is the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, $\mathsf{ZetaContinuationSocket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TrivialZeroClassifier}^{\uparrow}}$ *row.*

The carrier classifies a supplied zero as trivial only through the displayed negative-even route $I, Q, R, Z, E, A, H, C, P, N$. It supplies no critical-strip zero row, no non-trivial-zero row, and no proof of RH.

Theorem 999.2 (TrivialZeroClassifier NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TrivialZeroClassifier}^{\uparrow}}$ *surface for Definition 999.1 consists exactly of six finite obligations:*

1. *carrier admission for $T = (I, Q, R, Z, E, A, H, C, P, N)$;*
2. *negative-even source binding from I through Q and R;*
3. *zeta-zero handoff from the continuation socket row E to the zero row Z;*
4. *rational comparison exposure through A before any RH-facing classifier consumes the row;*
5. *componentwise stability under the displayed transports in H and replay through the routes in C;*
6. *non-escape from critical-strip classification, analytic zero search, host complex equality, quotient zero-set equality, representative selection, and an RH conclusion.*

The Lean taste gate for this surface is required to expose `Nontrivial TrivialZeroClassifierUp` *and* `FieldFaithful TrivialZeroClassifierUp` *instances in* `TasteGate.lean`*.*

Lean4 ✓ `BEDC.Derived.TrivialZeroClassifierUp.TrivialZeroClassifierTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 999.1. The packet displays the negative-even integer source, the arithmetic embeddings, the real-part row, the zeta-zero row, the continuation socket row, the rational comparison ledger, transports, routes, provenance, and local name.

The negative-even source binding is read from I, Q, R. These rows are the only coordinates that move the signed source into the rational and real ledgers, so the classifier cannot replace them by a host integer or real equality.

The zeta-zero handoff is read through E and Z. The continuation socket row supplies the functional-equation route, while Z names the zero row consumed by the RH catalogue.

The comparison and stability rows are read through A, H, C. They are the only displayed routes by which the negative-even side can be exposed to an RH-facing consumer.

Finally, P and N package the surface. Since no carrier coordinate names a critical-strip proof, analytic zero search, host complex equality, quotient zero-set equality, representative selector, or RH proof, the six obligations exhaust the local NameCert boundary. □

Theorem 999.3 (TrivialZeroClassifier negative-even route)**.** *For an accepted* $\mathsf{TrivialZeroClassifier}^{\uparrow}$ *carrier* $T = (I, Q, R, Z, E, A, H, C, P, N)$*, every accepted read of a negative-even zero factors through the displayed route*

$$I, Q, R, E, Z, A, H, C, P, N.$$

The read reaches $\mathsf{ZetaZeros}^{\uparrow}$ *only after the* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, continuation-socket, and rational-comparison rows have been displayed.*

Proof. Project the carrier rows $I, Q, R, E, Z, A, H, C, P, N$. The first three rows display the arithmetic embedding of the negative-even source, while E and Z display the zeta-continuation and zeta-zero coordinates for that same source.

Use the rational comparison ledger A after the zero row is present. This ledger records that the source lies on the negative-even side and is the only comparison coordinate exported by the packet.

The transport, route, provenance, and naming rows then stabilize the same displayed read. Thus an accepted negative-even-zero read reaches the zero row only along the stated finite route. □

Theorem 999.4 (TrivialZeroClassifier non-trivial separation)**.** *For an accepted* $\mathsf{TrivialZeroClassifier}^{\uparrow}$ *carrier* $T = (I, Q, R, Z, E, A, H, C, P, N)$*, no consumer can read a critical-strip non-trivial-zero classification from* T *alone. A downstream* $\mathsf{NontrivialZeroClassifier}^{\uparrow}$ *consumer may consume* T *as its trivial-zero ledger, but the critical-strip side must still be supplied by a separate* $\mathsf{ZetaZeros}^{\uparrow}$ *and* $\mathsf{CritStrip}^{\uparrow}$ *route as in chapter 659.*

Proof. Read the carrier definition. The packet has rows for the negative-even arithmetic source, zeta continuation, zeta zero, rational comparison, transport, route, provenance, and local name, but it has no $\mathsf{CritStrip}^{\uparrow}$ coordinate.

Apply the negative-even route theorem. It shows that every exported zero read is already tied to the displayed negative-even side before an RH-facing consumer sees it.

The non-trivial-zero classifier therefore receives this packet only as the trivial-zero ledger. Any critical-strip zero row consumed by that downstream classifier must be supplied by its own displayed Z, S, W route, not by $\mathsf{TrivialZeroClassifier}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{TrivialZeroClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TrivialZeroClassifier}^{\uparrow}$ packet is generated from a negative-even $\mathsf{Int}^{\uparrow}$ source row, $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ embedding rows, a $\mathsf{ZetaZeros}^{\uparrow}$ zero row, a $\mathsf{ZetaContinuationSocket}^{\uparrow}$ functional-equation route row, rational comparison data, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TrivialZeroClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 999.1, Definition 999.2, Definition 999.3, and Definition 999.4 bind the finite trivial-zero classifier boundary consumed by RH-facing classifier packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TrivialZeroClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RH, non-trivial-zero classification, critical-strip witness construction, analytic zero search, host complex equality, quotient zero-set equality, selected representatives, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, negative-even source binding, zeta-zero handoff, rational comparison exposure, classifier stability, and consumer non-escape for the same finite packet.

Chapter 1000

A Concrete Naming Certificate for $\mathsf{ApophaticKernelBoundary}^{\uparrow}$

The $\mathsf{ApophaticKernelBoundary}^{\uparrow}$ packet realizes the proof discipline of chapter 9400 at the kernel boundary. It sits beside the named externality packet of chapter 543, the socket ledger of chapter 536, the audit readback gate of chapter 584, and the refusal packet of chapter 630. Those siblings expose names, socket rows, gate readback, or negative certificates; this chapter packages the finite row where a kernel-facing request is refused unless its supply is displayed at type or ledger level.

Definition 1000.1 (Apophatic kernel boundary carrier)**.** *An* $\mathsf{ApophaticKernelBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, S, Q, R, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the accepted* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the external dependency position;*
2. *S is the accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *row recording the requested supply shape and socket ledger;*
3. *Q is the kernel-boundary question that asks whether the request is exposed at type or ledger level;*
4. *R is the refusal ledger for any request that would enter as an axiom, hidden setup field, host quotient, selected representative, or unrecorded supply;*
5. *G is the audit gate row that reads back the question, refusal outcome, and visible route;*
6. *H records componentwise* hsame *transports for the name, socket, question, refusal, gate, route, provenance, and local certificate rows;*
7. *C records the displayed* Cont *routes by which consumers read the name, socket, kernel-boundary question, refusal ledger, and audit gate;*
8. *P is the* Pkg *provenance over* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticKernelBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, or unrecorded dependency.

Lean4 ✓ `BEDC.Derived.ApophaticKernelBoundaryUp.ApophaticKernelBoundaryTasteGate single carrier alignment`

Theorem 1000.2 (Apophatic kernel boundary NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApophaticKernelBoundary}^{\uparrow}}$ *surface for Definition 1000.1 consists exactly of carrier admission for* $K = (A, S, Q, R, G, H, C, P, N)$*, name-to-socket routing from* A *to* S*, kernel-boundary refusal exactness through* Q *and* R*, audit-readback stability through* G*, componentwise transport through* H*, displayed route replay through* C*, package provenance through* P*, local naming through* N*, and non-internalization of the refused kernel-facing supply.*

Proof. Project the carrier. Its displayed rows are exactly the apophatic name, apophatic socket, kernel-boundary question, refusal ledger, audit gate, transport, route, provenance, and local naming rows.

Use the sibling packets cited above. The $\mathsf{ApophaticName}^{\uparrow}$ row supplies the named externality, the $\mathsf{ApophaticSocket}^{\uparrow}$ row supplies the socket ledger, the $\mathsf{ExternalityGate}^{\uparrow}$ row supplies visible gate readback, and the $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ row supplies the refusal discipline for attempted internalizers.

The local NameCert surface is therefore exhausted by the displayed packet. Since the carrier has no coordinate for an axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, or unrecorded dependency, no such object can be produced by the certificate. □

Theorem 1000.3 (Apophatic kernel boundary noninternalization)**.** *Every accepted* $\mathsf{ApophaticKernelBoundary}^{\uparrow}$ *packet keeps kernel-facing supply requests outside the generated substrate unless they are displayed as type-level or ledger-level rows. A consumer may cite the name row* A*, socket row* S*, boundary question* Q*, refusal ledger* R*, audit gate* G*, transport row* H*, route row* C*, provenance row* P*, and local naming row* N*, but it cannot extract an axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, or unrecorded dependency from the packet.*

Proof. Begin with the carrier inventory of Definition 1000.1. The packet displays the request, the gate question, the refusal ledger, and the readback route, but it does not display the requested kernel-facing supply as an internal coordinate.

Apply Definition 1000.2. A consumer route must pass through the name-to-socket route, the kernel-boundary question, the refusal ledger, and the audit gate before it can cite the packet.

Thus any route asking for an axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, or unrecorded dependency asks for data absent from the carrier and refused by the ledger. □

Theorem 1000.4 (Apophatic kernel boundary public route)**.** *Every public consumer of an accepted* $\mathsf{ApophaticKernelBoundary}^{\uparrow}$ *packet factors through the displayed route*

$$A \to S \to Q \to R \to G \to (H, C, P, N).$$

The route first names the externality, then records the socket, then asks the kernel-boundary question, then reads the refusal ledger, then exposes the audit gate, and only then replays transport, continuation, provenance, and local NameCert rows. No public route bypasses R *or* G *to reach an axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, or unrecorded dependency.*

Proof. Read the displayed order in Definition 1000.1. The rows A, S, Q, R, G appear before the transport, continuation, provenance, and local naming rows.

The NameCert obligations of Definition 1000.2 bind the name-to-socket route, refusal exactness, and audit-readback stability to that same carrier order. Hence a public citation of the packet must retain the refusal ledger and gate readback.

The noninternalization theorem then excludes every requested supply that is not a displayed row. The public route is therefore exactly the finite route through $A, S, Q, R, G, H, C, P, N$. □

Falsifiable prediction: Every accepted ApophaticKernelBoundaryUp consumer reaches a refusal-ledger row before it may cite a kernel-boundary request, and no consumer extracts an axiom, hidden setup field, host quotient, or unrecorded supply from the packet.

Independence witness: apophaticname, apophaticsocket, externalitygate, negativecertificate-boundary: ApophaticNameUp lacks a kernel-boundary refusal ledger, ApophaticSocketUp lacks the gate-readback row, ExternalityGateUp lacks the primitive-kernel refusal boundary, and NegativeCertificateBoundaryUp lacks the name-to-socket route; no bijection preserves all rows A,S,Q,R,G,H,C,P,N.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/ApophaticKernelBoundaryUp/TasteG` It must provide `Nontrivial ApophaticKernelBoundaryUp` and `FieldFaithful ApophaticKernelBoundaryUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticKernelBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticKernelBoundary}^{\uparrow}$ packet is generated from an apophatic-name row, an apophatic-socket row, a kernel-boundary question, a refusal ledger, an audit gate, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticKernelBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1000.1, Definition 1000.2, Definition 1000.3, and Definition 1000.4 bind the finite kernel-boundary refusal carrier, public route, and non-internalization surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticKernelBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not supply a kernel axiom, hidden setup field, host quotient, selected representative, primitive kernel supply, Lean verification, or bridge checking for $\mathsf{ApophaticKernelBoundary}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, name-to-socket routing, kernel-boundary refusal exactness, audit-readback stability, non-internalization, nontriviality, and field faithfulness for the same finite packet.

Chapter 1001

A Concrete Naming Certificate for $\mathsf{RealSealAdmissionCut}^{\uparrow}$

The $\mathsf{RealSealAdmissionCut}^{\uparrow}$ packet records the finite boundary at which a Real-completion route becomes admissible. It sits after the observation-budget limiter of chapter 903, the finite window carrier of chapter 126, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and before any $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff is consumed. The packet is an admission cut: it records the ordered entry boundary for a finite route, not a host completeness principle, quotient stream, selected limit, or independent real equality.

Definition 1001.1 (Real seal admission-cut carrier)**.** *A* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E, B, W, Q, S, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite seal-request row whose requested observation is being admitted;*
2. *B is the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row that caps the request before any window read is served;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row admitted under that budget;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only from the displayed window;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row that may be consumed only after E, B, W, Q have been displayed;*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row that reads the same admitted route;*
7. *H records componentwise* hsame *transport for the request, budget, window, readback, seal, limit handoff, route, provenance, and local naming rows;*
8. *C records the displayed* Cont *route from request to budget admission, from budget admission to window, from window to readback, from readback to Real seal, and from the Real seal to the Cauchy-limit handoff;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an unbounded observation schedule, a window served outside B, a $\mathsf{RegSeqRat}^{\uparrow}$ *readback outside W, a* $\mathsf{Real}^{\uparrow}$ *seal outside Q, a selected host limit, a quotient stream equality, or an ambient completeness theorem.*

Theorem 1001.2 (Real seal admission-cut NameCert obligations)**.** *For the carrier of Definition 1001.1, the local* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (E, B, W, Q, S, L, H, C, P, N),$$

budget-limited request admission, StreamName window locality, RegSeqRat readback locality, Real seal ordering, CauchyLimitSeal handoff ordering, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of any seal route that bypasses* B *or* W*.*

Proof. Project the carrier of Definition 1001.1. The finite packet exposes the request, budget, window, readback, seal, handoff, transport, route, provenance, and local naming rows.

Budget-limited request admission uses the sibling limiter row. The request E is admitted only through B, and the StreamName window W is served only under that admitted budget.

Readback and seal ordering then follow the displayed route. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q reads the finite window W, the $\mathsf{Real}^{\uparrow}$ seal S reads Q, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ row L consumes the same admitted seal route.

The transport, continuation, provenance, and naming rows preserve that order. Since the carrier has no coordinate for a bypassing schedule, host limit, quotient stream equality, independent seal, or ambient completeness theorem, the local NameCert surface is exhausted by the displayed admission cut. □

Theorem 1001.3 (Real seal admission-cut route non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff accepted through a* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *packet factors through the ordered finite boundary*

$$E \to B \to W \to Q \to S \to L.$$

A route that bypasses either the observation-budget limiter B *or the StreamName window* W *is not admitted by the packet and cannot be consumed as the displayed Real-completion handoff.*

Proof. Begin with the NameCert obligations of Definition 1001.2. They force every admitted read to expose the request, limiter, window, readback, seal, and limit-handoff rows in the displayed order.

Use H and C to transport and replay the same finite route. Componentwise hsame transport cannot replace the budget row or the window row, and the displayed continuation route has no branch from E directly to Q, S, or L.

It remains to exclude downstream escape. The carrier inventory has no row for an unbounded observation schedule, a selected host limit, quotient stream equality, independent real equality, or ambient completeness. Therefore any downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer receives only the admitted boundary $E, B, W, Q, S, L, H, C, P, N$. □

Theorem 1001.4 (Real seal admission-cut source scope)**.** *Every admitted* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *packet exposes the ordered source faces*

$$E, B, W, Q, S, L$$

as the request face, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *face,* $\mathsf{StreamName}^{\uparrow}$ *finite-window face,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face,* $\mathsf{Real}^{\uparrow}$ *seal face, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff face. No consumer of the packet may read* Q*,* S*, or* L *without first projecting the budget and window faces* B *and* W*.*

Proof. Project the carrier of Definition 1001.1. Its first six rows are named in the order E, B, W, Q, S, L, and the definition assigns these rows to the request, budget limiter, window, readback, Real seal, and limit handoff.

Apply Definition 1001.2. The local certificate surface accepts readback, seal, and handoff rows only as part of the displayed admission chain.

Finally apply Definition 1001.3. Since every admitted downstream read factors through $E \to B \to W \to Q \to S \to L$, a consumer that skips B or W is outside the packet's admitted source scope. □

Theorem 1001.5 (Real seal admission-cut classifier transport)**.** *For accepted packets with displayed rows*

$$A = (E, B, W, Q, S, L, H, C, P, N),$$

componentwise hsame *on the request, budget, window, readback, seal, handoff, provenance, and naming faces, together with the replay row C, transports admission of the same finite boundary. The transported packet has no bypass route from E to Q, S, or L, and no route from W to a host-selected limit outside the displayed handoff.*

Proof. Use the carrier definition to separate the transport row H from the replay row C. The row H records componentwise hsame for the displayed faces, while C records the displayed Cont route through the same ordered boundary.

The NameCert obligations of Definition 1001.2 bind those two rows to carrier admission, window locality, readback locality, Real-seal ordering, and CauchyLimitSeal handoff ordering. Transport therefore preserves the classifier only by preserving each displayed face and the replay route that connects them.

Apply Definition 1001.3 to exclude bypasses. A route from E directly to Q, S, or L, or from W to an unlisted host-selected limit, would not factor through the displayed admission chain and hence would not be the transported packet. □

Theorem 1001.6 (Real seal admission-cut ledger exactness)**.** *In an accepted* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *packet, the ledger rows P and N exhaust the local provenance and naming data for the admission cut. They record provenance over* BHist, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, hsame, Cont, *and* NameCert, *together with the local* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *row, and they exclude ledger coordinates for unbounded schedules, quotient streams, selected limits, host real equality, and ambient completeness.*

Proof. Project the final two rows of the carrier. The row P is the stated Pkg provenance over the displayed primitives, and N is the local naming row for the admission-cut packet.

Apply the NameCert obligations to identify which rows the local certificate admits. The obligations list provenance and local naming only through P and N, and the carrier contains no separate field for another provenance ledger or naming source.

The exclusions follow from the carrier inventory. Since unbounded schedules, quotient streams, selected limits, host real equality, and ambient completeness have no displayed coordinates, they cannot be recovered from P or N as local admission-cut ledger data. □

Theorem 1001.7 (Real seal admission-cut transport provenance field faithfulness)**.** *For an accepted* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *packet*

$$A = (E, B, W, Q, S, L, H, C, P, N),$$

componentwise hsame *transport and displayed* Cont *replay preserve the seal request E, observation-budget limiter B, finite window W, readback Q,* $\mathsf{Real}^{\uparrow}$ *seal S,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L, provenance P, and local* NameCert *row N together. No transported packet may collapse two displayed rows, introduce an unlisted field, or bypass either B or W while remaining the same admitted Real-seal boundary.*

Proof. Project the carrier named in Definition 1001.1. Its displayed data are exactly the rows $E, B, W, Q, S, L, H, C, P, N$, with H reserved for componentwise hsame transport, C reserved for displayed Cont replay, P reserved for provenance, and N reserved for the local naming row.

Apply Definition 1001.2. The local certificate surface binds carrier admission, budget-limited request admission, window locality, readback locality, Real-seal ordering, CauchyLimitSeal handoff ordering, transport, replay, provenance, and local naming as one finite admission-cut packet.

Apply Definition 1001.3 to the transported route. Since every accepted Real seal or limit handoff still factors through

$$E \to B \to W \to Q \to S \to L,$$

the transport row cannot identify away a displayed component, add a host-side coordinate, or replace the budget-window boundary by an unlisted route.

Thus the projected rows H, C, P, N preserve both provenance and field separation for the displayed packet. Any collapse of two rows, any extra field, or any route bypassing B or W would remove one of the projections required by the carrier and would no longer be an accepted $\mathsf{RealSealAdmissionCut}^{\uparrow}$ packet. □

Theorem 1001.8 (Real seal admission-cut scoped boundary)**.** *For an accepted* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *packet*

$$A = (E, B, W, Q, S, L, H, C, P, N),$$

the scoped boundary is exactly the finite BHist *request row E, the* $\mathsf{ObservationBudgetLimiter}^{\uparrow}$ *row B, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row S, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row L, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *row N. It admits no host real equality, quotient stream equality, selected limit, ambient completeness principle, or bypass route outside this displayed boundary.*

Proof. Start with the carrier of Definition 1001.1 and the NameCert obligations of Definition 1001.2. These identify the displayed BHist packet and bind its request, budget, window, readback, Real-seal, CauchyLimitSeal, transport, replay, provenance, and local naming rows as the complete admission surface.

The route and source restrictions are supplied by Definition 1001.3 and Definition 1001.4. They force every accepted downstream $\mathsf{Real}^{\uparrow}$ seal read and $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff to pass through

$$E \to B \to W \to Q \to S \to L.$$

Next apply Definition 1001.5. Componentwise hsame transport and the displayed Cont replay preserve the same finite boundary, so transport cannot introduce a classifier route from E directly to Q, S, or L, nor a route from W to a selected host limit.

Finally use Definition 1001.6 and Definition 1001.7. The Pkg row and local NameCert row exhaust the ledger data, while field faithfulness keeps $E, B, W, Q, S, L, H, C, P, N$ separated as the same admitted packet. Hence the scoped boundary is precisely the listed BEDC primitive rows, with host real equality, quotient stream equality, selected limits, ambient completeness, and bypass routes excluded. □

Theorem 1001.9 (Real seal admission-cut public NameCert)**.** *The carrier of Definition 1001.1, the obligation surface of Definition 1001.2, route non-escape from Definition 1001.3, the source scope of Definition 1001.4, classifier transport from Definition 1001.5, ledger exactness from Definition 1001.6, transport-provenance field faithfulness from Definition 1001.7, and the TasteGate marker together export a single public* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *interface. Downstream* $\mathsf{Real}^{\uparrow}$ *seal and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packets consume that interface only as the displayed admission boundary*

$$E \to B \to W \to Q \to S \to L$$

with componentwise transport, replay, provenance, and local naming rows.

Proof. Start from the carrier definition. It fixes the finite rows $E, B, W, Q, S, L, H, C, P, N$ and gives the local naming row N the role of $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$.

The obligation theorem identifies the admissible public surface of that row. Source scope and route non-escape then force every downstream $\mathsf{Real}^{\uparrow}$ seal read and every $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff through the same budget-window-readback chain, so the public object is the displayed admission boundary rather than a detached real or selected limit.

Classifier transport, ledger exactness, and transport-provenance field faithfulness supply the remaining public fields. They preserve the displayed faces under hsame and Cont, keep provenance in P, keep naming in N, and exclude any extra coordinate for quotient stream equality, ambient completeness, or a bypassing route.

The TasteGate marker certifies that the Lean-side carrier remains the same single finite packet. Therefore the listed rows and theorem surfaces repack as one public $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ interface for the RealUp seal and CauchyLimitSeal consumers. □

Theorem 1001.10 (Real seal admission-cut consumer bridge)**.** *The public* $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ *interface of Definition 1001.9 bridges to downstream* $\mathsf{Real}^{\uparrow}$ *seal and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumers exactly as the ordered finite boundary*

$$E \to B \to W \to Q \to S \to L$$

together with the componentwise transport, replay, provenance, and local naming rows H, C, P, N*. Any consumer route that bypasses* B *or* W*, or that uses quotient stream equality, selected limits, host real equality, unbounded schedules, or ambient completeness, is outside the bridge.*

Proof. Start with the carrier of Definition 1001.1. It supplies only the finite rows $E, B, W, Q, S, L, H, C, P, N$, with N serving as the local $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$ row and with no coordinate for host-side completion data.

Apply Definition 1001.2, Definition 1001.3, and Definition 1001.4. These identify the admissible NameCert surface, force every downstream seal or handoff through $E \to B \to W \to Q \to S \to L$, and deny access to Q, S, or L before the budget row B and window row W have been projected.

Use Definition 1001.9 to repack that surface as the public consumer interface. The existing TasteGate marker fixes the same single finite carrier on the verification side, so the bridge has no additional carrier on which quotient stream equality, selected limits, host real equality, unbounded schedules, or ambient completeness could enter.

Thus the consumer bridge is precisely the displayed ordered boundary together with H, C, P, N. Every route not passing through B and W is excluded by the carrier inventory and the non-escape theorem, hence is not a $\mathsf{RealSealAdmissionCut}^{\uparrow}$ bridge to the downstream $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer schema. □

Falsifiable prediction: A RealUp seal read that bypasses either the observation-budget limiter or StreamName window is not admitted by RealSealAdmissionCutUp.

Independence witness: finiterealroutecommitment, observationbudgetlimiter, streamname, regseqrat, cauchylimitseal: the admission cut is not bijective to any sibling because it records the ordered boundary at which the route becomes admissible, while the siblings record a commitment, a limiter, a window carrier, a readback, or a limit handoff separately.

The Lean `TasteGate.lean` boundary for this packet must provide `Nontrivial RealSealAdmissionCutUp` and `FieldFaithful RealSealAdmissionCutUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSealAdmissionCut}^{\uparrow}$ packet is generated from a finite seal request, an observation-budget limiter, a StreamName window, a RegSeqRat readback, a RealUp seal, a CauchyLimitSeal handoff, componentwise transports, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1001.1, Definition 1001.2, Definition 1001.3, Definition 1001.4, Definition 1001.5, Definition 1001.6, Definition 1001.7, Definition 1001.8, Definition 1001.9, Definition 1001.10, and

Lean4 ✓ `BEDC.Derived.RealSealAdmissionCutUp.taste gate`

bind the finite admission-cut carrier, ordered budget-window-readback boundary, RealUp seal ordering, CauchyLimitSeal handoff, non-escape surface, source scope, classifier transport, ledger exactness, transport-provenance field separation, scoped primitive boundary, public NameCert export, consumer bridge, and TasteGate marker.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSealAdmissionCut}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RealSealAdmissionCutUp.RealSealAdmissionCutTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, ambient completeness, unbounded observation, independent RealUp seal exposure, or machine-checked bridge status.

Upgrade path: No paper theory-axis grade remains for this chapter; bridge checking is a separate verification-axis task.

Chapter 1002

A Concrete Naming Certificate for CertificateAuditGate$^{\uparrow}$

The CertificateAuditGate$^{\uparrow}$ packet realizes the audit-gate boundary of chapter 9396 as a concrete local NameCert instance. It sits beside the operational membrane of chapter 533, the audit-gate boundary of chapter 526, and the replacement ledger of chapter 568. Those siblings expose membrane routing, gate rows, or replacement refusals; this packet binds the gate pass itself to a finite certificate surface that cannot be read as substrate-internal closure.

Falsifiable prediction: If the refusal row is removed, a gate pass can be read as substrate-internal closure rather than as local non-breach, and the NameCert classifier rejects that read.

Independence witness: operationalmembrane, auditgateboundary, effectivereplacementledger, reflectionlimitboundary, boundedsearchrefutation: none is bijective because CertificateAuditGate$^{\uparrow}$ carries gate input, checked surface, refusal, drift, axiom-purity, transport, route, provenance, and local naming rows in one packet, while each sibling lacks at least one of those rows or uses the rows for a different boundary question.

Definition 1002.1 (CertificateAuditGate carrier)**.** *A* CertificateAuditGate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the gate-input row, recording the finite manuscript, Lean target, source-file, and audit-command inputs supplied to the gate;*
2. *S is the checked-surface row, recording the finite pass surface actually inspected by the gate;*
3. *R is the refusal row, recording every reading that the gate rejects rather than promotes to substrate-internal closure;*
4. *D is the drift row, recording paper-Lean marker resolution, duplicate-label checks, and name-resolution failures as finite audit data;*
5. *A is the axiom-purity row, recording source-axiom refusal and dependency refusal for* Classical.choice, Quot.sound, *and* propext*;*
6. *H stores componentwise* hsame *transports for the input, checked surface, refusal, drift, axiom-purity, provenance, and naming rows;*

7. *C stores the* Cont *routes by which a consumer reads the gate input, checked surface, refusal, drift, axiom-purity, provenance, and local naming rows;*
8. *P records* Pkg *provenance over* $\mathsf{CertificateAuditGate}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{OperationalMembrane}^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *data;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateAuditGate}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a global repository-correctness theorem, a hidden source axiom, an unresolved Lean marker, a quotient representative, a choice selector, proposition equality, or a proof that a passing audit row becomes substrate-internal closure.

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.TasteGate.CertificateAuditGateCarrier event flow fields iff`

Theorem 1002.2 (CertificateAuditGate obligation surface)**.** *For every accepted carrier $G = (I, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CertificateAuditGate}^{\uparrow}}$ *surface consists exactly of gate-input exposure, checked-surface exactness, refusal-row attachment, drift-row resolution, axiom-purity refusal, componentwise* hsame *stability, displayed* Cont *routing,* Pkg *provenance, and the local naming row. No consumer may cite a gate pass without exposing all nine rows of the same finite packet.*

Proof. Project the carrier of Definition 1002.1. The rows $I, S, R, D, A, H, C, P, N$ are introduced as separate coordinates of one finite BHist packet.

The first five rows give the gate-facing surface: supplied inputs, inspected checks, explicit refusals, paper-Lean drift data, and axiom-purity data. The support rows H, C, P, N then transport, replay, cite, and name those same rows without adding a further proof principle.

Thus the local NameCert obligation is not a scalar pass bit. It is the displayed finite packet, and any consumer omitting one of the rows has left the certificate surface named by the carrier. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.TasteGate.CertificateAuditGateObligationSurface`

Theorem 1002.3 (CertificateAuditGate soundness boundary)**.** *For every accepted* $\mathsf{CertificateAuditGate}^{\uparrow}$ *carrier $G = (I, S, R, D, A, H, C, P, N)$, a passing checked-surface row S is sound only as local non-breach of the rows displayed by I, D, A, H, C, P, N. It does not assert global manuscript truth, host-kernel completeness, unrestricted semantic conservativity, or substrate-internal closure of the gate input.*

Proof. Read S together with the supplied inputs I. The checked surface records only the finite rows inspected for those inputs, so the pass result is bounded by the rows carried in the packet.

Use the drift row D and axiom-purity row A as the two gate ledgers. They report marker resolution, duplicate-label refusal, source-axiom refusal, and forbidden-dependency refusal as finite audit data.

Finally route the read through H, C, P, N. Componentwise transport, continuation replay, package provenance, and local naming keep the soundness claim on the displayed non-breach surface. Since no coordinate records global truth, host completeness, semantic conservativity, or internal closure, none can be projected from the pass row. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.TasteGate.CertificateAuditGateSoundnessBoundary`

Theorem 1002.4 (CertificateAuditGate non-escape)**.** *Every consumer read of an accepted* $\mathsf{CertificateAuditGate}^{\uparrow}$ *packet factors through*

$$I, S, R, D, A, H, C, P, N.$$

The read may expose gate inputs, checked rows, refusals, drift data, axiom-purity data, transports, routes, provenance, and local naming, but it cannot escape to a hidden audit oracle, unresolved target replacement, quotient representative, choice selector, proposition equality, or substrate-internal theorem supplied by the gate.

Proof. Start with the carrier inventory in Definition 1002.1. It lists exactly the gate input, checked surface, refusal, drift, axiom-purity, transport, route, provenance, and naming rows.

Apply Definition 1002.2. A public certificate read must expose the same nine rows and cannot replace the packet by a pass token.

Apply Definition 1002.3. The pass row is local non-breach for the displayed audit data rather than a theorem about all host or manuscript behaviour.

The excluded objects require coordinates absent from G. Therefore continuation replay through C and naming through N can return only the finite certificate rows, not a hidden oracle or substrate-internal theorem. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.TasteGate.CertificateAuditGateNonescape`

Theorem 1002.5 (CertificateAuditGate kernel scope)**.** *Every accepted* $\mathsf{CertificateAuditGate}^{\uparrow}$ *read is kernel-scoped by the finite rows*

$$I, S, R, D, A, H, C, P, N.$$

The carrier admission, checked-surface row, refusal row, drift row, axiom-purity row, transport row, route row, provenance row, and local naming row are all projected from the same packet, and no audit conclusion may be read outside that packet.

Proof. Project the carrier in Definition 1002.1. It gives the nine rows as coordinates of one finite BHist packet.

Apply Definition 1002.2 to identify those coordinates as the local $\mathsf{NameCert}_{\mathsf{CertificateAuditGate}^{\uparrow}}$ surface. Apply Definition 1002.4 to rule out hidden audit oracles, unresolved target replacement, quotient representatives, choice selectors, proposition equality, and substrate-internal theorem export.

Thus the audit read is exactly kernel-scoped by the displayed rows. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.CertificateAuditGateKernelScope`

Theorem 1002.6 (CertificateAuditGate drift exactness)**.** *For every accepted* $\mathsf{CertificateAuditGate}^{\uparrow}$ *packet, the drift row D is read exactly as paper-Lean marker resolution, duplicate-label checking, and name-resolution failure data for the same gate input I and checked surface S. It is not a substitute Lean target, a hidden marker repair, or an external repository-correctness theorem.*

Proof. The carrier definition assigns D precisely to marker resolution, duplicate-label checks, and name-resolution failures.

Use Definition 1002.2 to keep the drift row attached to I, S, R, A, H, C, P, N inside the same local NameCert surface. Then use Definition 1002.3: the pass row is local non-breach for the displayed rows rather than a global correctness theorem.

The non-escape theorem excludes unresolved target replacement and hidden audit oracles, so the only drift information available to consumers is the exact row D. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.CertificateAuditGateDriftExactness`

Theorem 1002.7 (CertificateAuditGate axiom-purity exactness)**.** *For every accepted* CertificateAuditGate$^{\uparrow}$ *packet, the axiom-purity row A records exactly source-axiom refusal and dependency refusal for* Classical.choice*,* Quot.sound*, and* propext*. It does not assert semantic conservativity, host-kernel completeness, or a proof principle beyond the inspected finite audit row.*

Proof. Project A from Definition 1002.1. The row is defined as the finite axiom-purity ledger for source-axiom refusal and the named forbidden dependencies.

Apply Definition 1002.2 and Definition 1002.3. Together they keep A inside the local checked surface and prevent the pass row from becoming a semantic-conservativity or host-completeness theorem.

Finally Definition 1002.4 prevents the consumer from escaping to a hidden source axiom, quotient representative, choice selector, or proposition equality. Hence the axiom-purity claim is exact for A and no stronger. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.taste gate`

Theorem 1002.8 (CertificateAuditGate public export)**.** *An accepted* CertificateAuditGate$^{\uparrow}$ *packet exports only the public rows*

$$I, S, R, D, A, H, C, P, N.$$

Consumers may read gate input, checked surface, refusal, drift, axiom-purity, componentwise transport, displayed route, provenance, and local naming from the same finite packet. They may not export repository completeness, host-kernel completeness, semantic conservativity, quotient representatives, choice selectors, proposition equality, unresolved target replacement, or substrate-internal closure from a gate pass.

Proof. Project the carrier in Definition 1002.1. The displayed packet contains exactly the gate input, checked surface, refusal, drift, axiom-purity, transport, route, provenance, and naming rows.

Apply Definition 1002.2, Definition 1002.6, and Definition 1002.7. These identify the checked surface, drift row, and axiom-purity row as finite public audit rows rather than hidden correctness oracles.

Apply Definition 1002.4 and Definition 1002.5. The public export is therefore bounded by $I, S, R, D, A, H, C, P, N$ and cannot leave the local NameCert packet. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.CertificateAuditGatePublicExport`

Theorem 1002.9 (CertificateAuditGate bridge boundary)**.** *An accepted* CertificateAuditGate$^{\uparrow}$ *public export through*

$$I, S, R, D, A, H, C, P, N$$

remains a paper-side certificate surface until a separate standard bridge theorem is supplied. The pass row S cannot be read as bridgeChecked*, repository completeness, host-kernel completeness, semantic conservativity, a quotient representative, a choice selector, proposition equality, unresolved target replacement, or substrate-internal closure.*

Proof. Begin with the public export of Definition 1002.8. It exposes exactly the rows $I, S, R, D, A, H, C, P, N$ and excludes repository completeness, host-kernel completeness, semantic conservativity, quotient representatives, choice selectors, proposition equality, unresolved target replacement, and substrate-internal closure.

Apply the soundness boundary of Definition 1002.3. The pass row is local non-breach for the displayed audit rows, not a theorem about all manuscript, repository, host-kernel, or semantic behaviour.

Use drift exactness from Definition 1002.6 and axiom-purity exactness from Definition 1002.7. These keep D and A as exact finite audit rows rather than bridge certificates or external correctness principles.

Finally apply non-escape and kernel scope, Definition 1002.4 and Definition 1002.5. The public read remains inside the same local NameCert packet, so bridge-checked status requires a separate standard bridge theorem. □

Lean4 ✓ `BEDC.Derived.CertificateAuditGateUp.CertificateAuditGateBridgeBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateAuditGate}\uparrow}$.

Constructive story: A $\mathsf{CertificateAuditGate}^{\uparrow}$ packet is generated from gate input, checked surface, refusal, drift, axiom-purity, componentwise transport, displayed route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateAuditGate}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1002.1, Definition 1002.2, Definition 1002.3, Definition 1002.4, Definition 1002.8, and Definition 1002.9 bind the finite audit-gate certificate surface, its public non-escape boundary, and its paper-side bridge boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateAuditGate}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CertificateAuditGateUp.taste\_gate`

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.certificateAuditGateDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim bridge checking, global repository correctness, host-kernel completeness, semantic conservativity, quotient representatives, choice selectors, proposition equality, or substrate-internal closure from a gate pass.

Upgrade path: Bridge-checked status requires a separate standard bridge package for the same finite audit-gate surface.

Chapter 1003

A Concrete Naming Certificate for $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$

The $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ packet makes the no-global-synchronization row of Definition 9419.4 into a concrete finite carrier. It is not a multi-Hist configuration, an observer-frame packet, a coupling ledger, an invariant value, or a causal-rate route. It stores the refusal row itself, together with the displayed histories, locality rows, hsame transport, Cont routes, package provenance, and local naming data that let downstream consumers cite the boundary without treating it as prose.

Definition 1003.1 (NoGlobalSyncBoundary carrier)**.** *A* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (H, L, R, S, C, P, N)$$

with the following displayed rows:

1. $H = (H_1, \ldots, H_n)$ *is the finite row of displayed observer histories;*
2. $L = (L_1, \ldots, L_n)$ *is the locality ledger, with* L_i *naming the local rows of* H_i *available to consumers;*
3. R *is the refusal row, recording that the packet admits no privileged shared clock, global observer frame, total observer order, or selected synchronization map;*
4. S *records the displayed* hsame *transports for the observer-history, locality-ledger, refusal, route, provenance, and naming rows;*
5. C *records displayed* Cont *routes for reading the histories, locality ledgers, refusal row, transport rows, package provenance, and local naming row;*
6. P *is the* Pkg *provenance over* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{FiniteMultiHistPacket}^{\uparrow}$, $\mathsf{ObserverFrameRefusal}^{\uparrow}$, $\mathsf{ObservationCoupling}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{CrossHistCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{NoGlobalSyncBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a shared clock, privileged frame, host-side observer order, synchronization selector, causal relation, invariant value, maximum-rate witness, metric row, or quotient of histories.

Theorem 1003.2 (NoGlobalSyncBoundary NameCert obligations)**.** *For the carrier of Definition 1003.1, the local* $\mathsf{NameCert}_{\mathsf{NoGlobalSyncBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission*

for

$$G = (H, L, R, S, C, P, N),$$

displayed observer-history rows through H, locality-ledger exactness through L, explicit refusal of shared clocks, privileged frames, total observer orders, and synchronization selectors through R, componentwise hsame *transport through S, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 1003.1. The rows H, L, R, S, C, P, N account for the observer histories, locality ledgers, refusal ledger, transport table, continuation routes, provenance, and local naming surface.

The positive rows H and L say which histories and locality rows may be read. The negative row R says which global synchronization data are refused. The structural rows S, C, P, N transport, replay, name, and record provenance for exactly those displayed rows.

No further row is available in the carrier. Thus every local NameCert obligation is either a projection from one of the displayed rows or the boundary clause that the missing global clock, frame, order, and selector are not admitted coordinates. □

Theorem 1003.3 (NoGlobalSyncBoundary frame-refusal handoff)**.** *Every accepted* NoGlobalSyncBoundary$^{\uparrow}$ *packet supplies the refusal row needed by an* ObserverFrameRefusal$^{\uparrow}$ *consumer. The consumer may read the displayed histories H, locality ledgers L, refusal row R, transport rows S, routes C, provenance P, and local naming row N, but it may not promote any of them to a shared clock, privileged observer frame, total observer order, or selected synchronization map.*

Proof. Read the carrier coordinates from Definition 1003.1. The refusal row R is explicitly part of the packet, and the rows S, C, P, N give the transport, route, provenance, and naming data by which an observer-frame consumer can cite it.

Apply Definition 1003.2. The same local NameCert surface that admits R also records the refusal of shared clocks, privileged frames, total observer orders, and synchronization selectors.

The handoff therefore carries negative boundary data, not an admitted global frame. Any accepted ObserverFrameRefusal$^{\uparrow}$ use can consume R as the displayed refusal row, while the missing synchronization objects remain outside the public packet. □

Theorem 1003.4 (NoGlobalSyncBoundary finite-packet non-escape)**.** *Every accepted* NoGlobalSyncBoundary$^{\uparrow}$ *consumer route factors through the finite rows*

$$H, L, R, S, C, P, N.$$

A consumer that exports a shared clock, privileged frame, total observer order, selected synchronization map, causal-rate route, invariant value, coupling ledger, metric row, or quotient of histories without displaying it as refused data has left the packet.

Proof. The carrier in Definition 1003.1 is finite and lists exactly seven rows. The rows H and L provide local history and locality access, R provides the refusal boundary, and S, C, P, N provide transport, replay, provenance, and naming.

No coordinate of this tuple is a shared clock, privileged frame, total observer order, selected synchronization map, causal-rate route, invariant value, coupling ledger, metric row, or quotient of histories. These objects cannot be projected from the accepted packet as admitted data.

It remains only the refusal case. If such an object is mentioned by an accepted consumer, it must be mentioned through R as excluded boundary data and replayed through the displayed routes in C. Otherwise the consumer has introduced a row that is not part of $G = (H, L, R, S, C, P, N)$. □

Theorem 1003.5 (NoGlobalSyncBoundary multi-Hist lattice link)**.** *Every accepted* NoGlobalSyncBoundary$^{\uparrow}$ *consumer reads the refusal row as a downstream lattice edge from the multi-history configuration surface of chapter 474 through the invariant surface of chapter 656, the causal-rate surface of chapter 478, and the maximum-rate boundary of chapter 594. The consumer may cite those sibling carriers only as displayed dependency rows, and it still exports no shared clock, privileged frame, selected synchronization map, causal-rate route, metric row, or quotient of histories.*

Proof. Project the carrier of Definition 1003.1. Its coordinates expose H, L, R, S, C, P, N: displayed histories, locality ledgers, the refusal row, componentwise transport, continuation routes, provenance, and local naming.

Apply the local NameCert obligations of Definition 1003.2. The provenance row P may name the multi-Hist, inter-Hist invariant, cross-Hist causal, and maximum-rate sibling surfaces as dependency rows, while the refusal row R records that no global synchronization object is admitted.

Finally apply Definition 1003.4. A consumer route that reads the cited siblings still factors through the finite packet $G = (H, L, R, S, C, P, N)$. Thus the sibling rows are provenance and consumer routes for the Inter-Hist locality lattice, not hidden synchronization data, causal-rate data, metric data, or a quotient of histories. □

Theorem 1003.6 (NoGlobalSyncBoundary public consumer theorem)**.** *Any downstream consumer citing* NoGlobalSyncBoundary$^{\uparrow}$ *may export only the public refusal boundary over the displayed histories H, locality ledger L, refusal row R, componentwise* hsame *transport S, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The consumer may not reinterpret the packet as a shared clock, privileged observer frame, total observer order, synchronization map, causal-rate route, invariant value, metric row, or quotient of histories.*

Proof. Begin with the finite-packet non-escape theorem Definition 1003.4. It says that every accepted consumer route factors through H, L, R, S, C, P, N and that no admitted coordinate stores a shared synchronization object.

Use the multi-Hist lattice link of Definition 1003.5. The public consumer may cite sibling histories, invariant rows, causal-rate rows, and maximum-rate boundaries only through provenance and displayed consumer routes.

Finally apply the NameCert obligations of Definition 1003.2. They identify R as the explicit refusal row and S, C, P, N as the transport, replay, provenance, and naming rows for that refusal. Thus the public export is exactly the refusal boundary, not a synchronization map or hidden global observer structure. □

Theorem 1003.7 (NoGlobalSyncBoundary TasteGate obligation)**.** *The later Lean site* `lean4/BEDC/Derived/NoG` *will provide explicit* Nontrivial NoGlobalSyncBoundary$^{\uparrow}$ *and* FieldFaithful NoGlobalSyncBoundary$^{\uparrow}$ *instances. These instances must keep the displayed histories, locality rows, refusal row,* hsame *transport,* Cont *routes, package provenance, and local naming row as distinguishable fields.*

Proof. The carrier separates positive locality data from the negative refusal row. A singleton collapse would identify a displayed history, a locality ledger, and the refusal of global synchronization, erasing the boundary that the packet is designed to expose.

Field faithfulness is the projection discipline for the displayed tuple. Each row of $G = (H, L, R, S, C, P, N)$ has a distinct consumer role, and the refusal row R must remain distinguishable from the finite multi-Hist rows and observer-frame handoff data. □

Lean4 ✓ `BEDC.Derived.NoGlobalSyncBoundaryUp.NoGlobalSyncBoundaryTasteGate single carrier alignment`

Falsifiable prediction: Any accepted NoGlobalSyncBoundaryUp consumer that exports a shared clock, privileged frame, total observer order, or selected synchronization map without a displayed refusal row refutes the packet.

Independence witness: finitemulthistpacket, observerframerefusal, observationcoupling, interhistinvariant, crosshistcausalrate: none is bijective because NoGlobalSyncBoundaryUp stores the refusal row itself rather than a configuration, observer-frame packet, coupling ledger, invariant value, or causal-rate route.

Closure status for $\mathsf{NameCert}_{\mathsf{NoGlobalSyncBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ packet is generated from displayed observer histories, locality ledgers, a no-global-synchronization refusal row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoGlobalSyncBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1003.1, Definition 1003.2, Definition 1003.3, Definition 1003.4, Definition 1003.5, Definition 1003.6, and

Lean4 ✓ `BEDC.Derived.NoGlobalSyncBoundaryUp.taste gate`

specify the public finite refusal boundary for no-global-synchronization consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoGlobalSyncBoundary}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.NoGlobalSyncBoundaryUp.NoGlobalSyncBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not construct a shared clock, privileged frame, total observer order, synchronization map, causal-rate route, invariant value, Lean proof, or bridge checking for $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$.

Upgrade path: The paper closure axis is mature for the displayed refusal boundary; bridge checking remains a verification-axis task over the same Lean target.

Chapter 1004

A Concrete Naming Certificate for $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$

The $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ packet realizes the shared dyadic threshold comparison row used by the Real and Completion route of chapter 9238. It consumes the tail-radius ledger of chapter 757, the shared-tail modulus surface of chapter 983, the finite window service of chapter 126, the regular rational readback of chapter 301, and the Real seal boundary of chapter 38. The packet isolates the comparison row between two dyadic radius ledgers before any shared-tail or Real-completion consumer may treat the threshold as usable.

Definition 1004.1 (Dyadic tail-radius comparison carrier)**.** *A* $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L_0, L_1, \rho_0, \rho_1, \kappa, W_0, W_1, R_0, R_1, E, H, Q, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two accepted* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ *rows supplying the left and right dyadic tail-radius ledgers;*
2. ρ_0 *and* ρ_1 *are the displayed radius rows read from those ledgers;*
3. κ *is the finite dyadic comparison row asserting that the two ledger radii admit a shared threshold usable by the downstream tail route;*
4. W_0 *and* W_1 *are the* $\mathsf{StreamName}^{\uparrow}$ *windows opened only after the comparison row has been displayed;*
5. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows served by the compared windows;*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the compared ledger route has been consumed;*
7. H *records componentwise* hsame *transport for the ledgers, radii, comparison row, windows, readbacks, Real seal, provenance, and local name;*
8. Q *records the displayed* Cont *routes from each ledger to its radius row, from the two radii to* κ*, from* κ *to the windows, from the windows to the readbacks, and from the readbacks to the Real seal;*
9. P *is the* Pkg *provenance over* $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host real equality, selected limits, independent radius replacement, an unbounded threshold search, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicTailRadiusComparisonUp.DyadicTailRadiusComparisonTasteGate single carrier alignment`

Theorem 1004.2 (Dyadic tail-radius comparison NameCert obligations)**.** *For the carrier of Definition 1004.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for C, left and right ledger exposure through L_0, L_1, radius exposure through ρ_0, ρ_1, comparison admission through κ, window service through W_0, W_1, regular readback through R_0, R_1, Real seal ordering through E, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through N. A shared-tail or Real-completion consumer obtains no seal read unless the displayed comparison row sits between the two radius ledger rows and the downstream windows.*

Proof. Project the displayed carrier. The two ledger rows, their radius rows, the comparison row, paired windows, readbacks, seal, transports, continuations, provenance, and local name are all explicit coordinates of the finite packet.

Read the sibling ledger and shared-tail routes. The two radius rows are consumed from chapter 757, while the downstream shared-tail consumer has the same finite-threshold shape as chapter 983. Therefore the local certificate must expose ρ_0, ρ_1, κ before the StreamName and RegSeqRat rows can be treated as a shared tail route.

The Real seal row is downstream of the same displayed route. Since the carrier contains no quotient stream, selected limit, hidden radius selector, unbounded threshold search, host real equality, or ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1004.3 (Dyadic tail-radius comparison threshold order)**.** *Every accepted* $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ *packet exposes the shared threshold route in the order*

$$L_0, L_1, \rho_0, \rho_1, \kappa, W_0, W_1, R_0, R_1.$$

No downstream $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer may read a paired window or readback before the comparison row κ has been displayed between the two radius ledger rows.*

Proof. Start with the carrier inventory of Definition 1004.1. The only route from the two radius ledgers to the paired windows passes through ρ_0, ρ_1 and then through the single comparison row κ.

Apply the NameCert obligations of Definition 1004.2. Ledger exposure precedes radius exposure, radius exposure precedes comparison admission, and comparison admission precedes window service and regular readback.

Finally compare the sibling shared-tail surfaces. The shared-tail modulus and regular tail-meet chapters may consume this packet only as a finite threshold route, so a consumer that reads W_0, W_1, R_0, R_1 without the displayed comparison row is outside the accepted packet. □

Theorem 1004.4 (Dyadic tail-radius comparison Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read admitted through an accepted* $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ *packet factors through*

$$L_0, L_1, \rho_0, \rho_1, \kappa, W_0, W_1, R_0, R_1, E, H, Q, P, N.$$

The seal is a handoff from a displayed comparison between two dyadic radius ledgers, not from a quotient stream, a selected completed limit, or an ambient completeness theorem.

Proof. Use threshold order from Definition 1004.3. It places the paired windows and readbacks behind the two ledger radii and their comparison row.

Read the Real boundary through chapter 38. The seal row E appears only after the paired readbacks, so the Real-facing route cannot detach the seal from the comparison packet.

The structural rows H, Q, P, N replay and name the same finite route. They add transport, continuation, provenance, and local certificate data, but no hidden radius, no second comparison row, and no external completion object. □

Theorem 1004.5 (Dyadic tail-radius comparison non-escape)**.** *No accepted* $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ *packet exports quotient stream equality, host real equality, a selected limit, an unbounded tail search, a hidden radius selector, a private comparison oracle, or an ambient completeness principle. Every shared-tail or Real-completion consumer read is exhausted by*

$$L_0, L_1, \rho_0, \rho_1, \kappa, W_0, W_1, R_0, R_1, E, H, Q, P, N.$$

Proof. Inspect the carrier coordinates. The exported mathematical rows are precisely the two dyadic tail-radius ledgers, their two radius rows, one comparison row, two windows, two regular readbacks, and one Real seal.

Apply the Real seal handoff. Any completion-facing consumer must traverse those rows before reaching E, and the structural rows only transport, replay, package, and name that route.

The excluded objects have no carrier coordinate and no route in Q. Hence extracting any of them is not a read of the accepted comparison packet. □

Falsifiable prediction: No admitted $\mathsf{Real}^{\uparrow}$ seal read through $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ is accepted unless the displayed dyadic comparison row sits between the two $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$ radius rows and the downstream shared-tail window/readback rows.

Independence witness: dyadictailradiusledger, cauchysharedtailmodulus, regularcauchytailmeet, streamname, real: none is bijective to $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ because this carrier stores two radius ledgers, their explicit comparison row, paired windows, paired readbacks, the Real seal handoff, and the structural certificate rows together.

The later Lean TasteGate boundary for this carrier must provide $\mathsf{Nontrivial}\ \mathsf{DyadicTailRadiusComparison}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DyadicTailRadiusComparison}^{\uparrow}$ instances, so the comparison packet is represented by an inhabited and field-faithful finite row rather than by a vacuous shared-tail annotation.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTailRadiusComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTailRadiusComparison}^{\uparrow}$ packet is generated from two DyadicTailRadiusLedger rows, two displayed radius rows, one dyadic comparison row, paired StreamName windows, paired RegSeqRat readbacks, a RealUp seal handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicTailRadiusComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1004.1, Definition 1004.2, Definition 1004.3, Definition 1004.4, and Definition 1004.5 bind the finite dyadic radius comparison row used by shared-tail Real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicTailRadiusComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected limits, host real equality, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger comparison order, shared-tail window and readback routing, RealUp seal handoff, non-escape, and TasteGate instances for the same finite packet.

Chapter 1005

A Concrete Naming Certificate for $\mathsf{InscriptionRouteAudit}^{\uparrow}$

The $\mathsf{InscriptionRouteAudit}^{\uparrow}$ packet gives the inscription-event vision of chapter 9397 a public route audit from visible gap to accepted checker row and downstream consumer. It sits beside the local inscription point of chapter 610, the full audit trace of chapter 682, and the finite witness route of chapter 638. Those siblings respectively expose the supply socket, the broad inscription trace, and a finite consumer route; the present packet isolates the route by which a visible gap becomes a named accepted checker row before a public consumer may read it.

Definition 1005.1 (Inscription route-audit carrier)**.** *An* $\mathsf{InscriptionRouteAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, G, R, A, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the prior source-scope row containing the records visible before the route is accepted;*
2. *G is the visible gap row whose public residue asks for a route;*
3. *R is the named route row proposed to connect the visible gap to a reusable checker surface;*
4. *A is the accepted checker row, recording the finite check by which R is admitted as public record data;*
5. *D is the downstream consumer row that may read the accepted route only after A is visible;*
6. *H records componentwise* hsame *transports for S, G, R, A, D together with continuation, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which a consumer reads source scope, visible gap, named route, accepted checker, and downstream consumer rows;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{InscriptionAuditTrace}^{\uparrow}$, $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$, $\mathsf{FiniteWitnessRoute}^{\uparrow}$, *and* $\mathsf{ProofTermErasureAuditRoute}^{\uparrow}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionRouteAudit}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these nine displayed rows. It has no coordinate for a private motive, hidden checker, unchecked route, downstream right outside D, host process trace, or consumer path that bypasses the accepted checker row.

Lean4 ✓ BEDC.Derived.InscriptionRouteAuditUp.TasteGate.InscriptionRouteAuditTasteGate single carrier alignment

Theorem 1005.2 (InscriptionRouteAudit NameCert obligations)**.** *For the carrier of Definition 1005.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionRouteAudit}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier exposure for* $Q = (S, G, R, A, D, H, C, P, N)$*;*
2. *visible-gap locality: every route read begins at the prior source scope* S *and the displayed gap row* G*;*
3. *accepted-route exactness: the named route* R *becomes public only through the accepted checker row* A*;*
4. *downstream consumer accountability: every public consumer row* D *factors through* A *before the consumer read is visible;*
5. *provenance and local naming exhaustion through* H, C, P, N*, with no private motive, hidden checker, unchecked route, or off-packet consumer right.*

Proof. Carrier exposure is projection from the displayed packet. The rows $S, G, R, A, D, H, C, P, N$ are the full inventory available to an audit reader.

The gap-facing route begins with S and G. These rows identify the prior records and the public gap residue; without them the named route has no visible socket to answer.

The checker-facing route then reads R through A. The accepted checker row is the only place where the proposed route is admitted as public record data, so no hidden checker or unchecked replay can replace it.

Finally, D, H, C, P, N bind downstream consumption, transport, continuation, provenance, and local naming to the same finite packet. Since the carrier has no private motive, host trace, or off-packet consumer coordinate, the five obligations exhaust the local certificate surface. □

Falsifiable prediction: If the route-audit carrier is atomic, every public downstream use must factor through the accepted checker row before the consumer row is visible.

Theorem 1005.3 (InscriptionRouteAudit falsifiable prediction)**.** *Every public downstream read of an accepted* $\mathsf{InscriptionRouteAudit}^{\uparrow}$ *carrier* $Q = (S, G, R, A, D, H, C, P, N)$ *factors through the accepted checker row* A *before the consumer row* D *is visible. A purported public use that reaches* D *while bypassing* A *is outside the carrier.*

Proof. Project the carrier of Definition 1005.1. The only consumer-facing row is D, and the definition places it after the named route and accepted checker rows R, A.

Apply the accepted-route exactness and downstream consumer accountability clauses of Definition 1005.2. They make A the public admission point for R and require every downstream read through D to factor through that checker row.

The transport, continuation, provenance, and local naming rows preserve this same route. Hence a consumer path that reaches D without first exposing A would require an additional carrier coordinate, which the packet does not contain. □

Independence witness: inscriptionauditrace, inscriptioncommitmentseal, inscriptionpoint, finite-witnessroute, prooftermerasureauditroute: the route-audit carrier has both accepted-checker and downstream-gap rows, while these siblings respectively lack the full route from visible gap to public consumer; hence no carrier bijection preserves all NameCert rows.

Theorem 1005.4 (InscriptionRouteAudit independence witness)**.** *The* InscriptionRouteAudit$^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, with* InscriptionAuditTrace$^{\uparrow}$, InscriptionCommitmentSeal$^{\uparrow}$, InscriptionPoint$^{\uparrow}$, FiniteWitnessRoute$^{\uparrow}$, *or* ProofTermErasureAuditRoute$^{\uparrow}$. *It uniquely binds the visible gap row G, the named route row R, the accepted checker row A, and the public consumer row D as one route-audit packet.*

Proof. Compare the displayed coordinates. The present carrier has the paired gap-to-checker and checker-to-consumer route

$$G \to R \to A \to D.$$

The inscription audit trace supplies a broader trace packet, while the commitment seal isolates a commitment row rather than the visible gap-to-checker route. The inscription point supplies the local socket and actuality rows, not the public consumer route from G through A.

The finite witness route supplies a Real-and-Completion consumer path, and the proof-term erasure audit route supplies erasure-facing audit rows. Neither contains this chapter's simultaneous visible gap, named route, accepted checker, and downstream public consumer rows. Thus no carrier bijection can preserve all local NameCert obligations. □

The Lean target family *BEDC.Derived.InscriptionRouteAuditUp* should provide a Nontrivial instance by exhibiting two distinct finite packets with different accepted checker rows, and a FieldFaithful instance whose projections recover $S, G, R, A, D, H, C, P, N$ from the carrier.

Theorem 1005.5 (InscriptionRouteAudit mature consumer surface)**.** *Downstream public consumers of* InscriptionRouteAudit$^{\uparrow}$ *read only the visible gap row G, named route row R, accepted checker row A, downstream consumer row D, componentwise transport row H, replay route row C, provenance row P, and local NameCert row N. No mature public read may inspect a private motive, hidden checker, unchecked route, or host process trace outside the displayed carrier.*

Proof. Project the carrier of Definition 1005.1. The consumer-facing rows are exactly G, R, A, D, H, C, P, N after the prior source scope has supplied the visible records.

The NameCert obligations in Definition 1005.2 require the named route to pass through the accepted checker row and require the downstream consumer row to factor through that checker. The falsifiable prediction in Definition 1005.3 gives the same consumer boundary as a public read discipline.

The independence witness in Definition 1005.4 excludes a sibling carrier bijection that would supply the missing private or host-side rows. Hence the displayed rows are the mature consumer surface. □

Closure status for NameCert$_{\text{InscriptionRouteAudit}^{\uparrow}}$.

Constructive story: An InscriptionRouteAudit$^{\uparrow}$ packet is generated from prior source scope, a visible gap, a named route, an accepted checker row, a downstream consumer row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InscriptionRouteAudit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1005.1, Definition 1005.2, Definition 1005.3, Definition 1005.4, Definition 1005.5, and

Lean4 ✓ `BEDC.Derived.InscriptionRouteAuditUp.taste gate`

bind the visible gap-to-accepted-route audit packet and its downstream consumer boundary.

Formal verification: FormalStatus(NameCert$_{\text{InscriptionRouteAudit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.InscriptionRouteAuditUp.TasteGate.InscriptionRouteAuditTasteGate\_single\_carrier\_alig`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, private agency, host process traces, unchecked routes, downstream rights outside the displayed consumer row, or a completed ontology of inscription events.

Upgrade path: The public consumer surface is recorded; no further paper-theory grade remains for this carrier.

chapter 576 supplies the explicit contrary-record witness consumed by the GapClosureBoundary$^{\uparrow}$ gap-closure boundary, with the routed child below recording the finite witnessed-refutation dependency.

Chapter 1006

A Concrete Naming Certificate for $\mathsf{GapClosureBoundary}^{\uparrow}$

The $\mathsf{GapClosureBoundary}^{\uparrow}$ packet gives the shared GAP and T socket discipline of chapter 9415 a finite common boundary. It sits beside the causal commitment of chapter 505, the finite witnessed refutation of chapter 576, the other-minds commitment of chapter 499, the aspect chain of chapter 575, and the solipsism refusal of chapter 975. Those siblings package application faces; the present carrier isolates the common closure boundary consumed by all of them.

Definition 1006.1 (GapClosureBoundary carrier)**.** *A* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, S, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the* GAP *request row, naming the forward-binding or socket-facing demand whose source is not generated by the closed substrate;*
2. *S is the source observation or socket row, displaying the finite record, locality cell, aspect side, or inscription route that the request may lawfully read;*
3. *R is the refusal and non-escape ledger, recording that an off-source closure request is not internalised;*
4. *H records componentwise* hsame *transports for G, S, R together with continuation, provenance, and naming rows;*
5. *C records the displayed* Cont *routes from the request row to the source row, and from every unsupported closure read to the refusal row;*
6. *P records* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the hard-problem sibling packets that consume this boundary;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row.*

The classifier accepts exactly finite packets with these seven rows. It has no coordinate for internal universal closure, private other-minds access, host causal necessity, direct aspect reduction, or a T *supply that bypasses the displayed source route.*

Theorem 1006.2 (GapClosureBoundary NameCert obligations)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *surface has exactly four obligations: carrier exposure for the finite request, source, refusal, and structural rows; source-route*

exactness for any displayed observation or socket row S; refusal-ledger routing for any request that tries to convert G into internal closure without S; and structural exhaustion through H, C, P, N. These obligations are sufficient for the shared boundary used by chapter 505, chapter 576, chapter 499, chapter 575, and chapter 975.

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundary namecert obligations`

Proof. Project the carrier of Definition 1006.1. The rows G, S, R, H, C, P, N are the full inventory visible to a boundary reader, so carrier exposure contributes no hidden coordinate.

Read the source route next. Any accepted use of the GAP request must show the source observation or socket row S before it can be consumed by a causal, refutation, other-minds, aspect-chain, or solipsism-refusal sibling.

Finally inspect a closure request with no displayed source route. The continuation row sends such a request to R, while H, P, N merely transport, package, and name the same finite packet. Since the carrier has no internal-closure coordinate, the four obligations exhaust the local certificate surface. □

Theorem 1006.3 (GapClosureBoundary refusal non-escape)**.** *Any accepted* GapClosureBoundary$^{\uparrow}$ *consumer that attempts to turn a* GAP *request into substrate-internal closure must first read the displayed source route S. If no such route is present, the consumer is routed to the refusal ledger R and receives no universal closure, private other-minds access, host causal necessity, direct aspect reduction, hidden* T *supply, or off-packet continuation.*

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundary refusal nonescape`

Proof. Use the NameCert obligations of Definition 1006.2. They divide every boundary read into the carrier exposure route, the displayed source route, and the refusal route.

When S is present, the consumer remains a reader of the finite source row and the structural rows H, C, P, N. The carrier still exports only a named boundary packet, not a substrate-internal closure principle.

When S is absent, the only available continuation is the refusal route into R. The refusal row is a ledger entry rather than a supply witness, so the excluded closure, access, necessity, reduction, hidden-source, and off-packet continuation coordinates cannot be exported. □

Theorem 1006.4 (GapClosureBoundary displayed source precondition)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *carrier*

$$B = (G, S, R, H, C, P, N),$$

any supported closure read of B must expose the displayed source observation or socket row S before a hard-problem sibling consumer reads the packet. The only public precondition for such a read is the finite BHist *packet together with the* hsame *transports in H, the* Cont *source route in C, the* Pkg *provenance in P, and the local* NameCert *row N.*

Proof. Project the carrier definition. The packet has exactly the request row G, the displayed source row S, the refusal ledger R, and the structural rows H, C, P, N.

Use the NameCert obligation theorem Definition 1006.2. Its source-route exactness clause requires a supported observation or socket-facing read to display S before any sibling application consumes the boundary.

The refusal non-escape theorem Definition 1006.3 rules out replacing this precondition by an internal universal closure row. Thus the supported closure read has to expose S and remains inside the displayed BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 1006.5 (GapClosureBoundary refusal ledger transport)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any off-source closure read of the request row* G *is transported to the refusal ledger* R *along the displayed structural rows* H, C, P, N*. This transport uses only componentwise* hsame*, the displayed* Cont *route to refusal,* Pkg *provenance, and the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundary refusal ledger transport`

Proof. Start with an off-source request. By definition it has no displayed source route through S, so it cannot satisfy the source precondition of Definition 1006.4.

Apply refusal non-escape. The only available continuation sends the request to R, while the rows H, C, P, N transport, route, package, and name the same finite packet.

The structural rows do not add a hidden source. They preserve the refusal ledger through hsame, Cont, Pkg, and NameCert, so the off-source read is transported to R rather than to internal universal closure, private access, causal necessity, aspect reduction, or a hidden T supply. □

Theorem 1006.6 (GapClosureBoundary NameCert obligation triad)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the complete local NameCert obligation surface is carrier exposure, displayed source exactness, refusal-ledger routing, and structural rows. Carrier exposure reads* G, S, R *from the finite packet, displayed source exactness requires any supported closure read to pass through* S*, refusal-ledger routing sends off-source closure requests to* R*, and* H, C, P, N *only transport, route, package, and name those rows.*

Proof. Project the carrier of Definition 1006.1. It exposes exactly the request row, displayed source row, refusal ledger, and structural rows.

Apply Definition 1006.2 and Definition 1006.4. Together they identify carrier exposure and source-route exactness as the supported part of the local certificate.

Finally use Definition 1006.5. It gives the off-source routing clause, while the structural rows remain the same H, C, P, N rows already displayed by the carrier. □

Theorem 1006.7 (GapClosureBoundary classifier exhaustion)**.** *The* $\mathsf{GapClosureBoundary}^{\uparrow}$ *classifier accepts only finite packets with rows*

$$G, S, R, H, C, P, N.$$

No accepted classifier read adds an internal closure coordinate, hidden T *supply, private other-minds access, causal necessity row, direct aspect reduction, or off-packet continuation beyond those seven displayed rows.*

Proof. Read the classifier clause in Definition 1006.1. It states that accepted packets have exactly the seven displayed rows and names the unavailable coordinates.

Use Definition 1006.6. The local obligations inspect those same rows as carrier exposure, source exactness, refusal routing, and structural transport.

The displayed source precondition and refusal transport theorems cover the two possible closure-facing reads. Since both reads remain inside G, S, R, H, C, P, N, the classifier surface is exhausted by the finite packet. □

Theorem 1006.8 (GapClosureBoundary ledger nonescape)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the refusal ledger* R *cannot export internal closure, a hidden* T *supply, private other-minds access, causal necessity, or aspect reduction. It only records that an off-source request has no displayed source route through* S *and is therefore retained as a refusal row transported by* H, C, P, N*.*

Proof. Begin with Definition 1006.5. It sends every off-source closure read of G to R along the structural rows.

Apply Definition 1006.6. The ledger is one part of the local NameCert surface, not a generator for additional closure-facing data.

Finally project the carrier. The excluded internal closure, hidden source, other-minds, causal, and aspect-reduction data are not rows of B, so the refusal ledger cannot export them. □

Theorem 1006.9 (GapClosureBoundary sibling consumer exhaustion)**.** *The hard-problem sibling consumers listed in the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet exhaust the exported boundary surface of* $B = (G, S, R, H, C, P, N)$*. Causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers may read the common request row* G*, source row* S*, refusal ledger* R*, and structural rows* H, C, P, N*, but none gains internal universal closure beyond that finite packet.*

Proof. Read the carrier and the NameCert obligation surface. The provenance row P records exactly the hard-problem sibling packets that consume this boundary, and the local certificate exposes no row beyond G, S, R, H, C, P, N.

Supported sibling reads use Definition 1006.4; they must expose S before consuming the packet. Off-source sibling reads use Definition 1006.5; they are sent to R through H, C, P, N.

Thus every listed sibling consumer is accounted for by either the displayed source route or the refusal ledger route. Since neither route contains an internal universal closure coordinate, the exported boundary is exhausted by the finite packet and no sibling acquires a stronger closure principle. □

Theorem 1006.10 (GapClosureBoundary source route exactness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any consumer of* B *that reads a displayed source observation or socket row must route the* GAP *request row* G *through* S *before any application sibling consumes the boundary packet. The source route is the only admissible pre-consumption route for causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers.*

Proof. Project the carrier of Definition 1006.1. The only source-facing rows available to a consumer are the GAP request row G and the displayed source observation or socket row S.

Use the obligation summary of Definition 1006.2. Its source-route clause requires an accepted boundary consumer to exhibit S before any application sibling may consume the packet.

The refusal non-escape row of Definition 1006.3 rules out bypassing S by turning G into internal closure. Thus every accepted source-reading consumer uses the displayed route $G \longrightarrow S$ first. □

Theorem 1006.11 (GapClosureBoundary refusal ledger routing)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any off-source closure request is routed to the refusal ledger* R*. Such a request cannot export internal closure, a hidden* T *supply, or any continuation that bypasses the displayed source row* S*.*

Proof. Read the carrier of Definition 1006.1. It contains one request row G, one displayed source row S, one refusal ledger R, and the structural rows H, C, P, N.

Apply Definition 1006.2. Its refusal-ledger clause sends every closure read without the displayed source route to R rather than to a substrate-internal closure coordinate.

Now invoke Definition 1006.3. The refusal row is a ledger endpoint, not a supply row, so it exports neither internal closure nor hidden T data. □

Theorem 1006.12 (GapClosureBoundary structural exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the rows* H, C, P, N *only transport, route, package, and locally name the same seven-row packet. They add no further source, refusal, closure, or hidden* T *coordinate.*

Proof. Start from the carrier definition. The packet inventory is exactly G, S, R, H, C, P, N, with H assigned to componentwise hsame transport, C to displayed Cont routes, P to Pkg provenance, and N to the local $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ row.

The obligation summary identifies structural exhaustion as one of the four local certificate duties. Therefore the structural rows can only preserve and repackage the already displayed request, source, and refusal rows.

Finally use refusal non-escape. Since off-source closure is routed to R and R exports no hidden supply, the structural rows have no coordinate from which to generate a second source, internal closure, or hidden T row. □

Theorem 1006.13 (GapClosureBoundary root carrier exposure)**.** *Every root consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier first sees exactly the finite packet*

$$B = (G, S, R, H, C, P, N).$$

The root-facing carrier exposure consists of the GAP *request row* G*, displayed source row* S*, refusal ledger* R*, componentwise transport* H*, displayed continuation routes* C*, package provenance* P*, and local* $\mathsf{NameCert}$ *row* N*, with no internal closure coordinate or hidden* T *supply preceding those rows.*

Proof. Project the carrier of Definition 1006.1. Its displayed inventory is the seven-row packet G, S, R, H, C, P, N, and the classifier admits no additional root-facing row.

Use Definition 1006.2 to read this inventory as the local carrier-exposure obligation. The source-route, refusal-ledger, and structural obligations are all obligations on the same packet rather than independent root carriers.

Finally apply structural exhaustion, Definition 1006.12. The rows H, C, P, N only transport, route, package, and name G, S, R, so a root consumer has no pre-packet coordinate from which to read internal closure, private access, causal necessity, direct aspect reduction, or a hidden T supply. □

Theorem 1006.14 (GapClosureBoundary root refusal exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every root off-source closure read is routed to the refusal row* R *through* H, C, P, N*. The route exports no internal closure coordinate and no hidden* T *supply beyond the displayed refusal ledger.*

Proof. Use the NameCert obligations of Definition 1006.2. They make refusal-ledger routing one of the root certificate duties and keep it inside the displayed carrier.

Apply the displayed-source precondition, Definition 1006.4. A root read without the displayed source row S cannot enter the supported source branch.

The refusal-ledger transport theorem, Definition 1006.5, sends that off-source read to R through H, C, P, N. Classifier exhaustion from Definition 1006.7 and root carrier exposure from Definition 1006.13 then exclude any internal closure coordinate or hidden T supply. □

Theorem 1006.15 (GapClosureBoundary root structural non-escape)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the structural rows* H, C, P, N *only transport, route, package, and*

name the displayed rows G, S, R. They do not add an internal closure row, a second source row, a second refusal ledger, or a hidden T *supply.*

Proof. Start with root carrier exposure from Definition 1006.13. The root packet has exactly the seven rows G, S, R, H, C, P, N.

The NameCert obligations of Definition 1006.2 assign H, C, P, N to componentwise transport, displayed route replay, package provenance, and local naming. These obligations do not create a new carrier coordinate.

Finally combine the displayed-source precondition, refusal-ledger transport, and classifier exhaustion of Definition 1006.4, Definition 1006.5, and Definition 1006.7. The structural rows can only preserve the supported branch through S or the refusal branch through R, so no hidden closure or T row escapes the displayed packet. □

Theorem 1006.16 (GapClosureBoundary root sibling handoff)**.** *Any* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutati}$ $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root packet receives only the finite boundary packet*

$$G, S, R, H, C, P, N.$$

The handoff supplies the request row, displayed source row, refusal row, transport, continuation, provenance, and local naming rows, and it supplies no internal closure coordinate, hidden T *row, private access row, causal necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Begin with the root carrier exposure theorem, Definition 1006.13. It fixes the handoff inventory as G, S, R, H, C, P, N before any hard-problem sibling reads the packet.

Use the NameCert obligations and displayed-source precondition, Definition 1006.2 and Definition 1006.4. Supported sibling reads pass through the displayed source branch, while unsupported reads cannot bypass the source check.

Apply refusal-ledger transport and classifier exhaustion from Definition 1006.5 and Definition 1006.7. The unsupported branch reaches R through H, C, P, N, and the classifier has no route to hidden closure data.

Therefore the listed siblings receive exactly the finite boundary packet G, S, R, H, C, P, N. The handoff has no coordinate for internal closure, a hidden T supply, private access, causal necessity, direct aspect reduction, or off-packet continuation. □

Theorem 1006.17 (GapClosureBoundary shared socket route package)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$, the shared socket route available to root hard-problem consumers is the finite package consisting of the request row G, displayed source row S, refusal row R, componentwise transport H, displayed continuation routes C, package provenance P, and local naming row N. A supported socket read uses the route $G \longrightarrow S$ through H, C, P, N; an unsupported socket read uses the route $G \longrightarrow R$ through the same structural rows.*

Proof. Project the carrier exposure theorem, Definition 1006.13. It fixes the root inventory as the seven rows G, S, R, H, C, P, N.

Use the NameCert obligations of Definition 1006.2. They separate the source-route obligation from the refusal-ledger obligation while keeping both routes inside the same finite packet.

The displayed source precondition, Definition 1006.4, gives the supported branch $G \longrightarrow S$. The refusal-ledger transport theorem, Definition 1006.5, gives the unsupported branch $G \longrightarrow R$.

Thus the shared socket route is not a hidden supply route. It is exactly the public packet G, S, R, H, C, P, N with one supported source branch and one refusal branch. □

Theorem 1006.18 (GapClosureBoundary source refusal exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every supported root read is exhausted by the displayed source row* S*, and every unsupported root read is exhausted by the refusal row* R*. No accepted root read of* B *has a third route to internal closure, private access, causal necessity, direct aspect reduction, hidden* T *supply, or off-packet continuation.*

Proof. Begin with the shared socket route package above. It presents the root-facing read as either the source branch $G \longrightarrow S$ or the refusal branch $G \longrightarrow R$, transported through H, C, P, N.

For a supported read, apply Definition 1006.4. The consumer must expose the displayed source observation or socket row S before the packet is consumed, so the supported branch is exhausted by S and the structural rows.

For an unsupported read, apply Definition 1006.5. The request has no displayed source route and is sent to R along H, C, P, N.

Finally use sibling consumer exhaustion, Definition 1006.9. The listed hard-problem consumers receive no route outside the finite packet, so the source row and refusal row exhaust the root alternatives. □

Theorem 1006.19 (GapClosureBoundary hard-problem consumer handoff)**.** *Every* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet receives only the finite boundary packet*

$$B = (G, S, R, H, C, P, N).$$

The consumer may read a supported source route through G, S, H, C, P, N *or an unsupported refusal route through* G, R, H, C, P, N*, but it receives no consumer-specific closure row outside that packet.*

Proof. Read the carrier exposure theorem first. The handoff begins with $B = (G, S, R, H, C, P, N)$, so every downstream consumer starts from the same finite request, source, refusal, transport, continuation, provenance, and naming rows.

Apply Definition 1006.2. The local certificate exposes carrier, source-route, refusal-ledger, and structural duties; it does not expose a separate consumer-specific hard-problem row.

Use source refusal exhaustion above to split the handoff. Supported consumers receive the route through S, while unsupported consumers receive the refusal route through R.

Finally apply Definition 1006.9. The causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal chapters consume the shared packet only, so the handoff remains a finite boundary rather than hidden supply. □

Theorem 1006.20 (GapClosureBoundary source-route obligation)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any supported root read of the* GAP *request must pass through the displayed source row* S *before a hard-problem sibling consumer may use the boundary. The support route is transported only by* H*, replayed only by* C*, packaged only by* P*, and locally named only by* N*.*

Proof. Begin with the displayed source precondition, Definition 1006.4. It requires any supported closure read to expose the source observation or socket row S before the packet is consumed.

Apply source route exactness, Definition 1006.10. The admissible pre-consumption route for the hard-problem siblings is the displayed route from G to S; no sibling may substitute an internal closure route.

The structural rows do not change this obligation. By Definition 1006.12, H, C, P, N only transport, replay, package, and name the same route, so every supported root read passes through S before it reaches a sibling consumer. □

Theorem 1006.21 (GapClosureBoundary refusal-ledger obligation)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any off-source root read of the request row* G *is routed to the refusal ledger* R *through* H, C, P, N*. Such a read receives only the refusal transport of the finite packet and no internal universal closure, hidden* T *supply, private other-minds access, causal necessity row, direct aspect reduction, or off-packet continuation.*

Proof. Start with the refusal-ledger transport theorem, Definition 1006.5. An off-source request has no displayed route through S, so the displayed continuation sends it to R along H, C, P, N.

Use refusal ledger routing, Definition 1006.11, to identify this as the root-facing obligation rather than an analytic side condition. The ledger endpoint is part of the same accepted packet and cannot be replaced by an undisplayed source.

Finally combine ledger nonescape and structural exhaustion, Definition 1006.8 and Definition 1006.12. The refusal row is preserved by transport, replay, provenance, and naming, while the excluded closure, hidden supply, other-minds, causal, aspect, and off-packet coordinates are absent from the carrier. □

Theorem 1006.22 (GapClosureBoundary root source handshake)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the complete root source route is the displayed handshake*

$$G \longrightarrow S$$

transported by H*, replayed by* C*, packaged by* P*, and locally named by* N*. Every downstream consumer in chapter 505, chapter 576, chapter 499, chapter 575, or chapter 975 reads the root request* G *through this source row before it may use the boundary as a causal, refutation, other-minds, aspect, or solipsism input.*

Proof. Start with the source-route obligation, Definition 1006.20. It already identifies S as the displayed source that every supported root read of G must expose.

Use source-route exactness, Definition 1006.10, to make the route unique at the root surface. The admissible handshake is $G \longrightarrow S$, not a route through internal closure or a sibling-specific private premise.

Finally apply structural exhaustion, Definition 1006.12. The rows H, C, P, N only transport, replay, package, and name the same handshake, so every listed downstream consumer receives the same finite source route. □

Theorem 1006.23 (GapClosureBoundary root refusal totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any root request that does not exhibit the handshake* $G \longrightarrow S$ *is routed to the refusal ledger* R*. The route to* R *is total for off-source causal, refutation, other-minds, aspect, and solipsism consumers, and it exports no internal closure, hidden* T *supply, private access row, causal necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Apply Definition 1006.21. It sends every off-source root read of the request row G to the refusal ledger R through H, C, P, N.

Use refusal ledger routing, Definition 1006.11, to identify R as the ledger endpoint rather than a second source. Thus every listed consumer that lacks the displayed source handshake receives the same refusal route.

The non-escape theorem, Definition 1006.3, removes the excluded exits. Since R is not an internal closure coordinate or a hidden supply row, the refusal is total across the downstream root surface. □

Theorem 1006.24 (GapClosureBoundary root structural exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the structural rows* H, C, P, N *are the complete NameCert root surface beyond the request, source, and refusal rows. They transport the source handshake* $G \longrightarrow S$*, route off-source reads to* R*, package the same boundary for downstream causal, refutation, other-minds, aspect, and solipsism consumers, and add no further root coordinate.*

Proof. Read the structural exhaustion theorem, Definition 1006.12. It restricts H, C, P, N to transport, route, package, and local naming for the seven-row packet.

Combine it with the root source handshake. The structural rows preserve the displayed route $G \longrightarrow S$ and cannot introduce a second source before a downstream consumer reads the boundary.

Combine it with root refusal totality. Off-source reads are routed to R, and the same structural rows preserve that refusal route without exposing an internal closure coordinate, hidden supply, or consumer-specific private row.

Therefore the complete root NameCert surface is exactly G, S, R, H, C, P, N, with H, C, P, N exhausting the structural part of the shared boundary. □

Theorem 1006.25 (GapClosureBoundary root source exactness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any supported root read of the* GAP *request row* G *must expose the displayed source row* S *before a downstream hard-problem consumer reads the boundary. The support data available to that read is exactly the source route carried by* H*, replayed by* C*, packaged by* P*, and locally named by* N*.*

Proof. Project the seven-row carrier of Definition 1006.1. The only source-facing coordinate in the accepted packet is the displayed row S.

Apply the NameCert obligations and displayed-source precondition, Definition 1006.2 and Definition 1006.4. They require any supported read of G to pass through the displayed observation or socket row before a sibling consumer may use the packet.

Finally use root structural exhaustion, Definition 1006.24. The rows H, C, P, N transport, replay, package, and name the same source route, so they do not add an alternate root source. □

Theorem 1006.26 (GapClosureBoundary consumer refusal exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any downstream consumer that lacks the displayed source row* S *is exhausted by the refusal ledger* R*. The refusal route gives no internal universal closure, hidden* T *supply, private other-minds access, causal necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Begin with the refusal-ledger transport theorem, Definition 1006.5. An off-source request cannot satisfy the source precondition, so its displayed continuation route goes to R.

Use the root refusal totality row, Definition 1006.23. It identifies the same refusal route as total for the causal, refutation, other-minds, aspect, and solipsism consumers.

The non-escape content of Definition 1006.3 removes the excluded exits. Since the accepted packet has no hidden closure coordinate or hidden supply row, a consumer without S receives only the refusal ledger transported by H, C, P, N. □

Theorem 1006.27 (GapClosureBoundary structural transport closure)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the structural rows* H, C, P, N *only transport, route, package, and name the same seven-row packet. They preserve the source route through* S *and the refusal route through* R*, and they add no hidden source, second refusal ledger, internal closure coordinate, or off-packet consumer row.*

Proof. Read the structural rows in the carrier definition. The row H supplies componentwise hsame transport, C supplies displayed Cont routes, P supplies Pkg provenance, and N supplies the local NameCert row.

Apply the structural exhaustion theorem and the root structural exhaustion theorem, Definition 1006.12 and Definition 1006.24. Together they restrict H, C, P, N to the already displayed request, source, and refusal rows.

Combine root source exactness with consumer refusal exhaustion. Supported consumers are carried through S, unsupported consumers are carried to R, and both routes remain inside the same packet G, S, R, H, C, P, N. □

Theorem 1006.28 (GapClosureBoundary TasteGate nontrivial witness surface)**.** *For accepted* GapClosureBoundary$^{\uparrow}$ *packets over the displayed carrier $B = (G, S, R, H, C, P, N)$, a packet whose request row G is routed through a displayed source row S and a packet whose request row is routed to the refusal ledger R determine distinct witness surfaces. The future* Nontrivial GapClosureBoundary$^{\uparrow}$ *TasteGate instance is therefore witnessed by the paper-level separation between source-supported boundary use and refusal-ledger boundary use.*

Proof. Read the carrier and local obligations from Definition 1006.1 and Definition 1006.2. The carrier exposes G, S, R, H, C, P, N, and the obligations distinguish the displayed source route from the refusal route.

For the source-supported surface, apply source-route exactness and the root source handshake, Definition 1006.10 and Definition 1006.22. A supported read passes through $G \longrightarrow S$ before a downstream hard-problem sibling may consume the boundary.

For the refusal surface, apply refusal-ledger routing and root refusal totality, Definition 1006.11 and Definition 1006.23. An off-source request is routed to R and receives no internal closure row.

Finally use structural exhaustion, Definition 1006.12. The rows H, C, P, N transport, replay, package, and name the same separation rather than collapsing S and R. Hence the two accepted surfaces are distinct at the public witness boundary. □

Theorem 1006.29 (GapClosureBoundary field-faithful row separation)**.** *The rows G, S, R, H, C, P, N of an accepted* GapClosureBoundary$^{\uparrow}$ *carrier are jointly faithful for the public NameCert read: changing the displayed source route S or changing the refusal-ledger route R changes the public read of the boundary packet. No componentwise transport, continuation replay, provenance row, or local naming row can identify a source-supported packet with a refusal-routed packet.*

Proof. Start from carrier exposure and the local NameCert obligations, Definition 1006.1 and Definition 1006.2. The public read records the request row, source row, refusal row, transport row, continuation row, package row, and local naming row.

Use source-route exactness for the supported side. A change in the displayed source route changes the public route by which G reaches S, so the accepted packet presents a different source-supported boundary read.

Use refusal-ledger routing for the off-source side. A change in R changes the ledger endpoint for requests that do not display S, and the non-escape rows prevent that difference from being hidden by an internal closure coordinate.

Finally apply root structural exhaustion, Definition 1006.24. Since H, C, P, N only transport, replay, package, and name G, S, R, they preserve rather than erase row differences. Thus the seven displayed rows are faithful for the public NameCert read. □

Theorem 1006.30 (GapClosureBoundary AI obligation readiness package)**.** *The* $\mathsf{GapClosureBoundary}^{\uparrow}$ *chapter presents the obligation-closure surface consisting of carrier exposure, source-route exactness, refusal-ledger routing, non-escape, falsifiable prediction, independence witness, TasteGate nontrivial witness separation, and field-faithful row separation. This package is the paper-level readiness surface for the later TasteGate carrier over* G, S, R, H, C, P, N*.*

Proof. Carrier exposure is supplied by Definition 1006.1 and the NameCert obligation rows of Definition 1006.2. These rows present the accepted packet as finite BHist data with explicit source, refusal, transport, route, provenance, and naming coordinates.

Source-route exactness and refusal-ledger routing are supplied by Definition 1006.10, Definition 1006.22, Definition 1006.11, and Definition 1006.23. They divide supported reads from off-source reads without adding a hidden closure row.

Non-escape and consumer exhaustion are supplied by the refusal non-escape rows, structural exhaustion, and sibling consumer exhaustion. Together they rule out internal universal closure, hidden T supply, private other-minds access, causal necessity, direct aspect reduction, and off-packet continuation.

The displayed falsifiable prediction and independence witness record how the same finite boundary is tested by downstream siblings. Together with Definition 1006.28 and Definition 1006.29, these rows form the obligation-closure surface expected by the TasteGate work. □

Falsifiable prediction: Any accepted GapClosureBoundaryUp consumer that tries to convert a GAP request into internal closure is routed to the refusal row unless a displayed source route is present.

Independence witness: causalcommitment, finitewitnessedrefutation, othermindscommitment, aspectchain, solipsismgaprefusal: none is bijective to GapClosureBoundaryUp because each sibling packages one application face while this carrier contains the common GAP request plus refusal ledger consumed by all faces.

The Lean file `Derived/GapClosureBoundaryUp/TasteGate.lean` will provide $\mathsf{Nontrivial}\ \mathsf{GapClosureBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{GapClosureBoundary}^{\uparrow}$ instances for the displayed rows G, S, R, H, C, P, N.

Theorem 1006.31 (GapClosureBoundary root source exposure)**.** *Every root read of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet first exposes the displayed finite rows*

$$G, S, R, H, C, P, N.$$

The read may inspect the GAP request row G*, the source row* S*, the refusal ledger* R*, the componentwise* hsame *transport row* H*, the displayed* Cont *routes* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*. It cannot read a hidden* T *supply, an internal universal closure coordinate, private other-minds access, causal necessity, direct aspect reduction, or any off-packet continuation before this boundary packet is exposed.*

Proof. Start with the carrier exposure theorem, Definition 1006.13. It identifies the root-facing packet as the seven displayed rows G, S, R, H, C, P, N.

Use the source-route obligation and root source handshake, Definition 1006.20 and Definition 1006.22. A supported root read of G must pass through the displayed source row S before any downstream hard-problem sibling may consume the boundary.

Finally apply root structural exhaustion, Definition 1006.24. The rows H, C, P, N transport, route, package, and name the same packet, so they cannot introduce a hidden T supply, internal closure coordinate, or private consumer-specific row. □

Theorem 1006.32 (GapClosureBoundary root ledger routing)**.** *For an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, every unsupported root request for closure is routed through the refusal ledger* R *by the same transport, continuation, provenance, and local naming rows* H, C, P, N*. The route replaces the unsupported request by a visible refusal read and exports no internal closure witness, hidden* T *supply, private access row, causal necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Project the refusal row from the carrier and apply Definition 1006.21. Unsupported root reads of the request row G are sent to the displayed ledger R through H, C, P, N.

Use refusal totality, Definition 1006.23, to cover the off-source cases. If the source handshake $G \longrightarrow S$ is absent, the packet has no accepted closure route except the ledger route to R.

The non-escape and structural rows, Definition 1006.3 and Definition 1006.24, remove the excluded exits. The same H, C, P, N rows preserve the refusal route without adding an internal closure witness or hidden supply coordinate. □

Theorem 1006.33 (GapClosureBoundary root consumer coverage)**.** *The root-facing boundary rows of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet exhaust the shared inputs used by* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*. Each listed consumer reads either the displayed source route through* G, S, H, C, P, N *or the refusal route through* G, R, H, C, P, N*, and no listed consumer receives an additional root row outside* G, S, R, H, C, P, N*.*

Proof. Begin with Definition 1006.31. It fixes the root-facing inventory for every consumer as the same seven-row packet.

For source-supported use, apply the root source handshake. The causal, refutation, otherminds, aspect, and solipsism consumers may use the boundary only after the displayed route $G \longrightarrow S$ is exposed and transported by H, C, P, N.

For unsupported closure requests, apply root ledger routing. The same consumers receive the refusal route through R and do not obtain an internal closure witness, hidden supply, private access row, causal necessity row, direct aspect-reduction row, or off-packet continuation.

Finally use the existing independence witness together with root structural exhaustion. The listed siblings package distinct application faces, while the shared root packet contains exactly the common request, source, refusal, transport, route, provenance, and naming rows consumed by all of them. □

Theorem 1006.34 (GapClosureBoundary downstream source refusal)**.** *Every listed downstream hard-problem consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet must choose either the displayed source route* $G \longrightarrow S$ *or the refusal route* $G \longrightarrow R$*. No downstream consumer among* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receives a third root route outside* G, S, R, H, C, P, N*.*

Proof. Use root source exposure, Definition 1006.31. It fixes the common packet available to the downstream consumers as G, S, R, H, C, P, N.

Apply root ledger routing and root consumer coverage, Definition 1006.32 and Definition 1006.33. The supported case goes through the displayed source row S, while the unsupported case is routed to the refusal ledger R.

Root structural exhaustion then excludes the third option. Since the carrier has no private hard-problem route outside the displayed rows, every downstream read is either source-routed or refusal-routed. □

Theorem 1006.35 (GapClosureBoundary downstream boundary package)**.** *For the listed downstream hard-problem consumers, the exact boundary package is the pair of routes $G \longrightarrow S$ and $G \longrightarrow R$ together with the structural rows H, C, P, N. This package is the whole exported interface for source exposure, refusal routing, transport, continuation, provenance, and local naming.*

Proof. Begin with root consumer coverage. It states that each downstream consumer reads the same root-facing packet and receives no additional row outside G, S, R, H, C, P, N.

Use downstream source refusal above to split the packet into its two possible root routes. The positive branch is the source route through S, and the negative branch is the refusal route through R.

The rows H, C, P, N carry the routes without changing their boundary. Thus the downstream package is exactly the two route rows plus the displayed structural support. □

Theorem 1006.36 (GapClosureBoundary downstream structural exhaustion)**.** *The downstream boundary package G, S, R, H, C, P, N cannot add hidden T supply, private access, causal necessity, aspect reduction, or off-packet continuation for any listed hard-problem consumer. All such requests are refused unless they are already represented by the displayed source or refusal route.*

Proof. Start from root structural exhaustion, Definition 1006.24. It removes hidden supply, private rows, causal rows, aspect-reduction rows, and off-packet continuations from the root-facing carrier.

Combine this with the downstream boundary package above. Since the downstream package has only G, S, R, H, C, P, N, the structural rows can transport and name a source or refusal route but cannot create any excluded object.

Therefore the listed hard-problem consumers inherit the same exhaustion boundary. A consumer may use the displayed source route or the displayed refusal route, and nothing outside the packet is exported. □

Theorem 1006.37 (GapClosureBoundary source-consumer uniformity)**.** *For an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet, every supported root read consumed by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *factors through the same displayed source route*

$$G \longrightarrow S$$

and through the same structural rows H, C, P, N. No listed consumer receives a private source route, hidden closure coordinate, or consumer-specific row outside G, S, H, C, P, N.

Proof. First expose the common root packet by Definition 1006.31. The supported read begins with the displayed GAP request row G, reaches the source row S, and is transported, routed, packaged, and named by H, C, P, N.

Then apply consumer coverage, Definition 1006.33. The causal, refutation, other-minds, aspect, and solipsism consumers all read the same root-facing packet, so a supported consumer route cannot add a separate source coordinate.

The conclusion follows because the only supported route visible to any listed consumer is the displayed route $G \longrightarrow S$ together with H, C, P, N. The excluded hidden closure and private rows are absent from the exposed packet. □

Theorem 1006.38 (GapClosureBoundary refusal-consumer uniformity)**.** *For an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet, every unsupported root read requested by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *is routed to the refusal ledger R through the same structural rows H, C, P, N. The routed refusal exports no hidden closure row,*

supply row, private-access row, causal-necessity row, aspect-reduction row, or off-packet continuation.

Proof. Start with root ledger routing, Definition 1006.32. An unsupported request has no accepted source route, so it is sent to the displayed refusal ledger R and remains transported, routed, packaged, and named by H, C, P, N.

Now use root consumer coverage, Definition 1006.33. Each listed downstream consumer receives either the supported source route or the refusal route from the same root-facing packet.

Since the request is unsupported, only the refusal branch remains. The non-escape content already carried by ledger routing prevents the consumer from rereading H, C, P, N as hidden closure, supply, private access, causal necessity, direct aspect reduction, or an off-packet continuation. □

Theorem 1006.39 (GapClosureBoundary root unblock package)**.** *The* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root obligation surface consists of three displayed rows of use: carrier exposure by* G, S, R, H, C, P, N*, supported source consumption through* $G \longrightarrow S$ *with* H, C, P, N*, and unsupported refusal consumption through* R *with* H, C, P, N*. Together with refusal non-escape and TasteGate readiness, these rows form the downstream root-unblock package for* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*.*

Proof. Carrier exposure is supplied by Definition 1006.31. It fixes the root-facing inventory as the finite rows G, S, R, H, C, P, N and excludes hidden closure, supply, private access, causal necessity, direct aspect reduction, and off-packet continuation rows.

Supported source consumption is supplied by Definition 1006.37. Unsupported refusal consumption is supplied by Definition 1006.38. These two theorems give the positive and negative consumer routes over the same structural rows.

Finally combine refusal non-escape with the TasteGate readiness package already displayed in this chapter. The result is a three-row root obligation surface: expose the carrier, route supported reads through the source, and route unsupported reads to refusal without importing any hidden closure principle. □

Theorem 1006.40 (GapClosureBoundary source observation totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every supported read of* B *exposes both the request row* G *and the source observation row* S *before any hard-problem sibling consumes the packet. The source route is exact: the sibling receives the displayed source branch through* G, S, H, C, P, N *and receives no hidden closure row, private source row, or consumer-specific source coordinate.*

Proof. Project the carrier of Definition 1006.1. The root-facing packet exposes exactly G, S, R, H, C, P, N, so the supported source branch has to begin with the displayed request and source rows.

Use the displayed-source precondition of Definition 1006.4. A supported closure read must show S before a causal, refutation, other-minds, aspect-chain, or solipsism-refusal consumer may read the packet.

Finally apply source route exactness from Definition 1006.10. The route from G to S is the only admissible pre-consumption route, and the structural rows H, C, P, N carry that same route without adding another source coordinate. □

Theorem 1006.41 (GapClosureBoundary refusal ledger totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every off-source closure request routes to the refusal ledger* R *through* H, C, P, N*. The refusal route exports no hidden* T *supply, private access row, causal-necessity row, direct aspect-reduction row, internal closure witness, or off-packet continuation.*

Proof. Start with refusal-ledger transport, Definition 1006.5. An off-source request has no accepted route through S, so the displayed continuation sends it to R along the structural rows.

Apply refusal ledger routing, Definition 1006.11. The route to R is the accepted endpoint for a closure read that cannot display the source row.

Now use ledger nonescape, Definition 1006.8. Since R records refusal rather than supply, the routed request cannot export hidden T data, private access, causal necessity, aspect reduction, internal closure, or an off-packet continuation. □

Theorem 1006.42 (GapClosureBoundary NameCert root package)**.** *The* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root package is assembled from carrier exposure, source-observation totality, refusal-ledger totality, and structural exhaustion. Every named downstream sibling consumes exactly one of the two displayed branches of* $B = (G, S, R, H, C, P, N)$*: supported reads through* G, S, H, C, P, N *or off-source reads through* G, R, H, C, P, N*.*

Proof. Carrier exposure is supplied by Definition 1006.13. It fixes the finite root packet as G, S, R, H, C, P, N before any downstream hard-problem sibling reads it.

Source-observation totality is supplied by Definition 1006.40. Refusal-ledger totality is supplied by Definition 1006.41. Together they exhaust the supported and off-source branches.

Finally apply structural exhaustion, Definition 1006.12, and the root unblock package, Definition 1006.39. The structural rows transport, route, package, and name the two displayed branches without creating a third route, so the root package is exactly the finite NameCert surface consumed by the named downstream siblings. □

Theorem 1006.43 (GapClosureBoundary downstream source route package)**.** *For every supported downstream read by* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, the shared* $\mathsf{GapClosureBoundary}^{\uparrow}$ *boundary factors through the displayed route*

$$G \longrightarrow S$$

and through the structural rows H, C, P, N*. No supported downstream read receives a hidden closure row, private source row, or consumer-specific access route outside* G, S, H, C, P, N*.*

Proof. Expose the root source packet by Definition 1006.31. It fixes the boundary inventory as G, S, R, H, C, P, N and identifies the only supported source-facing route as $G \longrightarrow S$.

Use root ledger routing to keep unsupported closure requests outside the supported branch. The ledger theorem prevents the refusal row from being reread as a second positive source route.

Now apply source-consumer uniformity and the root-unblock package. The five listed siblings consume the same source route through $G \longrightarrow S$, and H, C, P, N provide the transport, continuation, provenance, and local naming rows that carry this route downstream. □

Theorem 1006.44 (GapClosureBoundary sibling refusal transport package)**.** *For every unsupported sibling read by* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, the shared* $\mathsf{GapClosureBoundary}^{\uparrow}$ *boundary factors through the refusal row* R *and through the structural rows* H, C, P, N*. The refusal route exports no hidden closure row, hidden* T *supply, private access row, causal-necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Start with root ledger routing, Definition 1006.32. An unsupported request cannot use the source row as a witness of closure, so it is routed to the displayed refusal ledger R.

Apply refusal-consumer uniformity. The same five siblings receive this refusal route through H, C, P, N, not through private sibling rows or a second root carrier.

Finally use root source exposure and the root-unblock package to identify the ambient packet. The only exported unsupported route is the refusal row transported, routed, packaged, and named by H, C, P, N. □

Theorem 1006.45 (GapClosureBoundary shared boundary consumer package)**.** *For* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$, *the shared* $\mathsf{GapClosureBoundary}^{\uparrow}$ *boundary package is exhausted by two cases: supported downstream reads factor through* $G \longrightarrow S$ *with* H, C, P, N, *and unsupported sibling reads factor through* R *with* H, C, P, N. *No third sibling-consumer route is exported by the boundary packet.*

Proof. First use root source exposure to fix the common carrier rows G, S, R, H, C, P, N. These rows are the whole boundary packet visible to the listed siblings.

For the supported case, apply Definition 1006.43. The downstream read is carried by the displayed route $G \longrightarrow S$ and by H, C, P, N.

For the unsupported case, apply Definition 1006.44. The read is carried by R and the same structural rows.

The root-unblock package combines source exposure, root ledger routing, source-consumer uniformity, and refusal-consumer uniformity. Therefore the two displayed cases exhaust the shared boundary consumed by the five siblings. □

Theorem 1006.46 (GapClosureBoundary source-refusal orthogonality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$, *a supported root read must pass through the displayed source row* S, *while an unsupported closure read is routed to the refusal ledger* R. *The two routes are orthogonal at the public boundary: neither* S *nor* R *can be reread as a hidden closure supply, and no accepted read obtains both a source witness and a refusal witness as a concealed substitute for internal closure.*

Proof. Project the seven-row carrier from Definition 1006.1. The only source-facing coordinate is S, and the only off-source endpoint is the refusal ledger R.

Apply the NameCert obligations and displayed source precondition, Definition 1006.2 and Definition 1006.4. A supported read of the request row G has to expose S before any downstream consumer can use the boundary.

For an unsupported read, use refusal non-escape, Definition 1006.3. The read is routed to R and receives no internal universal closure, hidden T supply, private access row, causal necessity row, direct aspect reduction, or off-packet continuation. Thus the source route and refusal route remain separate public alternatives rather than a combined hidden supply. □

Theorem 1006.47 (GapClosureBoundary hard-problem consumer factorization)**.** *Every accepted root read used by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *factors through the same displayed packet* G, S, R, H, C, P, N. *A supported consumer factors through* G, S, H, C, P, N, *and an unsupported consumer factors through* G, R, H, C, P, N; *no listed consumer receives a consumer-specific carrier outside the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *rows.*

Proof. Start with the carrier definition and the NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the shared root packet and record the hard-problem sibling packets only through the provenance row P.

For supported consumers, apply the displayed source precondition. The causal, refutation, other-minds, aspect, and solipsism routes can use the boundary only after G has been read through S and transported, replayed, packaged, and named by H, C, P, N.

For unsupported consumers, apply refusal non-escape. The same hard-problem request is sent to R through the structural rows, and the excluded closure, private access, causal necessity, direct aspect reduction, hidden T supply, and off-packet continuation are absent from the carrier. Hence every listed consumer factors through one of the two displayed finite routes. □

Theorem 1006.48 (GapClosureBoundary closure-request determinacy)**.** *For accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carriers, agreement on the request row G, source row S, transport row H, continuation row C, provenance row P, and local naming row N determines whether a root closure request is supported or refusal-routed. If the displayed source route through S is available, the read is supported through G, S, H, C, P, N; if that route is absent, the same continuation and naming surface routes the request to R.*

Proof. Read the displayed source precondition, Definition 1006.4. It identifies the public support condition as the presence of the source observation or socket row S together with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

Use the NameCert obligation theorem to inspect the continuation row. The row C routes supported reads through the source row and routes unsupported closure reads to the refusal row, while H, P, N transport, package, and name that same decision.

Finally apply refusal non-escape. Once the source route is absent, the request cannot escape to an internal closure coordinate or hidden T supply, so the public data G, S, H, C, P, N fixes the branch: supported through S when the source route is displayed, and refusal-routed to R otherwise. □

Theorem 1006.49 (GapClosureBoundary sibling source totality)**.** *For every supported read of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$, *the consumer first factors through the displayed source route*

$$G \longrightarrow S.$$

Only after that route is exposed may the read use the structural rows H, C, P, N. Thus the listed sibling consumers receive a common source route before any transport, continuation, provenance, or local naming row is read.

Proof. Project the carrier $B = (G, S, R, H, C, P, N)$ using Definition 1006.1. The carrier exposes one request row, one displayed source row, one refusal row, and the structural rows H, C, P, N.

Apply the NameCert obligations and displayed source precondition, Definition 1006.2 and Definition 1006.4. A supported root request cannot be read by a sibling consumer until the public source row has been displayed as the route $G \longrightarrow S$.

Now use source-refusal orthogonality, hard-problem consumer factorization, and closure-request determinacy. These theorems keep the supported branch separate from R, force each listed sibling through the same boundary packet, and make the source branch the determined route whenever the displayed source is present. The rows H, C, P, N therefore occur only as the transport, continuation, provenance, and local naming surface for that already displayed source route. □

Theorem 1006.50 (GapClosureBoundary refusal route determinacy)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$, if a root closure request has no displayed source route through $G \longrightarrow S$, then the request is deterministically routed to the refusal row R. The same H, C, P, N*

rows transport, route, package, and name this refusal; no unsupported request may choose a second source row, hidden closure row, private access row, or hidden T *supply.*

Proof. Start with the displayed source precondition. Its contrapositive boundary reading is that a supported branch must display the route from G to S before any sibling can consume the boundary.

When that route is absent, apply refusal non-escape and source-refusal orthogonality. Refusal non-escape sends the unsupported request to R and excludes internal closure, private access, causal necessity, direct aspect reduction, off-packet continuation, and hidden T supply; orthogonality prevents R from being reread as a concealed source branch.

Finally use hard-problem consumer factorization and closure-request determinacy. Every listed sibling reads the same packet G, S, R, H, C, P, N, and the public data determine the branch: displayed source gives the supported route, while absence of the displayed source gives the refusal route R with the same structural rows. □

Theorem 1006.51 (GapClosureBoundary structural row conservation)**.** *For accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carriers, the structural rows* H, C, P, N *conserve only the two displayed branches: the source branch* $G \longrightarrow S$ *and the refusal branch through* R*. These rows cannot introduce internal closure, a hidden* T *supply, a private access route, causal necessity, direct aspect reduction, or any consumer-specific route outside the displayed boundary packet.*

Proof. Read the carrier and NameCert obligations together. The rows H, C, P, N are respectively transport, continuation, provenance, and local naming rows for the visible packet G, S, R; they are not additional carriers and contain no independent closure coordinate.

For the supported branch, apply sibling source totality. Each listed sibling reaches H, C, P, N only after the route $G \longrightarrow S$ has been displayed, so the structural rows preserve that route rather than supplying a hidden source.

For the unsupported branch, apply refusal route determinacy and refusal non-escape. The same structural rows carry the request to R and exclude hidden closure supply, private access, causal necessity, direct aspect reduction, off-packet continuation, and hidden T supply.

The source-refusal orthogonality and hard-problem consumer factorization theorems then exhaust the boundary. Since every listed consumer is routed through one of the two displayed branches over G, S, R, H, C, P, N, the structural rows conserve the visible source/refusal split and cannot add an internal closure branch. □

Theorem 1006.52 (GapClosureBoundary downstream branch determinacy)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *and every hard-problem consumer among* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *the consumer chooses exactly one of the displayed root branches: the source route*

$$G \longrightarrow S$$

or the refusal route

$$G \longrightarrow R.$$

The structural rows H, C, P, N *carry the chosen branch and do not create a consumer-specific third route.*

Proof. Start with hard-problem consumer factorization and closure-request determinacy, Definition 1006.47 and Definition 1006.48. They place every listed consumer inside the packet G, S, R, H, C, P, N and determine whether a root request is source-supported or refusal-routed.

For the source case, apply sibling source totality. A supported consumer first displays the route $G \longrightarrow S$ before H, C, P, N can transport, route, package, or name the read.

For the refusal case, apply refusal route determinacy. When the source route is absent, the same consumer is routed to R over H, C, P, N and cannot choose a hidden closure row, private source row, or hidden T supply.

Structural row conservation then excludes a third branch. The rows H, C, P, N conserve the selected source/refusal branch rather than adding another consumer route. □

Theorem 1006.53 (GapClosureBoundary refusal source coverage)**.** *The source and refusal branches of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet jointly cover all accepted hard-problem reads by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$. *A listed read is accepted only through* $G \longrightarrow S$ *with* H, C, P, N *or through* $G \longrightarrow R$ *with* H, C, P, N*; there is no accepted third route outside* G, S, R, H, C, P, N*.*

Proof. First expose the shared boundary carrier by Definition 1006.1 and Definition 1006.2. The accepted packet has exactly the request, source, refusal, transport, continuation, provenance, and local naming rows.

Use source-refusal orthogonality to separate the two possible reads. The source branch carries a displayed source row, while the refusal branch carries a refusal ledger rather than concealed closure supply.

Apply downstream branch determinacy. Each listed hard-problem consumer is assigned to one of the two branches already displayed by the packet.

Finally apply structural row conservation and refusal non-escape. They rule out hidden closure, hidden T supply, private access, causal necessity, direct aspect reduction, and off-packet continuation, so the two branches jointly cover the accepted reads. □

Theorem 1006.54 (GapClosureBoundary consumer exactness package)**.** *The consumer exactness surface of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *is the finite package consisting of carrier admission for* $B = (G, S, R, H, C, P, N)$*, downstream branch determinacy, refusal-source coverage, structural row conservation, and the root-unblock package. This surface is exact for the listed hard-problem consumers: it exports the source branch, the refusal branch, and the rows* H, C, P, N*, but it exports no hidden closure, hidden* T *supply, private access route, causal-necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Carrier admission is given by Definition 1006.1 and Definition 1006.2. These results fix the finite packet G, S, R, H, C, P, N before any listed consumer reads the boundary.

Next use downstream branch determinacy. It assigns each accepted consumer read to the displayed source route or the displayed refusal route over the same structural rows.

Then use refusal-source coverage. The two displayed branches cover all accepted hard-problem reads, so no consumer needs an extra route outside the finite packet.

Finally combine structural row conservation with the root-unblock package of Definition 1006.39. The structural rows carry only the selected branch, and the root package records the same source/refusal surface for $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, and $\mathsf{SolipsismGapRefusal}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundaryConsumerExactnessPackage`

Theorem 1006.55 (GapClosureBoundary source route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every supported hard-problem consumer read by* $\mathsf{CausalCommitment}^{\uparrow}$,

$\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *must expose the source route*

$$G \longrightarrow S$$

before consuming the structural rows H, C, P, N*. No listed supported consumer may read the structural surface as a substitute for the displayed source row.*

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundarySourceRouteTotality`

Proof. Start with hard-problem consumer factorization, Definition 1006.47. It places every listed supported consumer inside the shared packet G, S, R, H, C, P, N rather than inside a consumer-specific carrier.

Apply sibling source totality. Definition 1006.49 requires the displayed route $G \longrightarrow S$ before any of the structural rows H, C, P, N are consumed by a listed sibling.

Finally use source-refusal orthogonality and structural row conservation. Definition 1006.46 keeps the source route separate from the refusal ledger, while Definition 1006.51 prevents H, C, P, N from generating a hidden source, closure, or T supply. Hence the source route is total for supported hard-problem reads. □

Theorem 1006.56 (GapClosureBoundary refusal route exactness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every unsupported closure request by a listed hard-problem consumer is routed exactly through*

$$G \longrightarrow R$$

with the structural rows H, C, P, N*. Such a request cannot export hidden closure, hidden* T *supply, private access, causal necessity, direct aspect reduction, or an off-packet continuation.*

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.GapClosureBoundaryRefusalRouteExactness`

Proof. Begin with closure-request determinacy. By Definition 1006.48, once the displayed source route is absent, the same continuation and naming surface determines the refusal branch.

Apply refusal route determinacy. Definition 1006.50 sends the unsupported request to R over H, C, P, N and excludes a second source row, hidden closure row, private access row, and hidden T supply.

The exactness of this route is preserved by source-refusal orthogonality and structural row conservation. The first prevents R from being reread as a concealed source, and the second keeps H, C, P, N as transport, route, package, and local naming rows for the displayed refusal branch. □

Theorem 1006.57 (GapClosureBoundary consumer ledger exhaustion)**.** *For the listed hard-problem siblings* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, source-route totality together with refusal-route exactness exhausts the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consumer ledger. Every supported read is accounted for by* $G \longrightarrow S$ *with* H, C, P, N*, every unsupported closure request is accounted for by* $G \longrightarrow R$ *with* H, C, P, N*, and no third consumer route is exported.*

Proof. First account for supported reads. Definition 1006.55 forces every listed supported consumer through $G \longrightarrow S$ before the structural rows are read.

Next account for unsupported requests. Definition 1006.56 routes each unsupported closure request through $G \longrightarrow R$ with H, C, P, N and excludes hidden closure, hidden T supply, and off-packet continuations.

It remains to exclude a third ledger entry. Apply hard-problem consumer factorization and closure-request determinacy to keep every listed consumer inside the same source/refusal split, then apply structural row conservation to prevent H, C, P, N from introducing a consumer-specific route. Thus the two displayed branches exhaust the consumer ledger. □

Definition 1006.58 (GapClosureBoundary hub root inventory)**.** *The hub root inventory for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *is the displayed carrier* $B = (G, S, R, H, C, P, N)$ *together with the existing carrier-obligation row of Definition 1006.2, the source-route row of Definition 1006.55, the refusal-route row of Definition 1006.56, the structural rows carried by* H, C, P, N*, and the consumer-exactness row of Definition 1006.54.*

Theorem 1006.59 (GapClosureBoundary hub root readiness)**.** *The existing child theorems for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *assemble the root-unblock surface for* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*. The assembled surface is the carrier inventory* G, S, R, H, C, P, N*, the supported source route* $G \longrightarrow S$*, the unsupported refusal route* $G \longrightarrow R$*, and the exact consumer package that exports no hidden closure, hidden* T *supply, private access, causal-necessity row, direct aspect-reduction row, or off-packet continuation.*

Proof. Use the carrier obligations of Definition 1006.2 to fix the shared packet G, S, R, H, C, P, N before any listed hard-problem consumer reads the boundary.

Apply source-route totality and refusal-route exactness, Definition 1006.55 and Definition 1006.56. They give the two displayed branches $G \longrightarrow S$ and $G \longrightarrow R$ over the same structural rows.

Finally apply the root-unblock and consumer-exactness packages, Definition 1006.39 and Definition 1006.54. These packages identify the five listed consumers as readers of the same finite source/refusal surface and exclude the hidden rows named in the statement. □

Theorem 1006.60 (GapClosureBoundary hub unblock surface)**.** *Every downstream consumer named by the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root package reads only the displayed source/refusal split through* G, S, R, H, C, P, N*. A supported read passes through* $G \longrightarrow S$ *with* H, C, P, N*, an unsupported closure request passes through* $G \longrightarrow R$ *with* H, C, P, N*, and no downstream consumer receives a third route outside the hub root inventory.*

Proof. Begin with the hub root inventory of Definition 1006.58. It names the carrier, source route, refusal route, structural rows, and consumer-exactness row used by the hub.

For supported reads, use Definition 1006.55. For unsupported closure requests, use Definition 1006.56. Both routes are expressed through the same coordinates G, S, R, H, C, P, N.

The consumer-exactness package of Definition 1006.54 then excludes any off-inventory route. Hence the hub exposes exactly the displayed split and no additional downstream surface. □

Theorem 1006.61 (GapClosureBoundary root visible refusal audit)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root packet* $B = (G, S, R, H, C, P, N)$*, every downstream refusal read must first display the gap request row* G*, the refusal ledger row* R*, the structural boundary rows* H, C, P, N*, the finite* BHist *carrier, the componentwise* hsame *transports, the displayed* Cont *refusal route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row. The read is therefore an audit of the visible root boundary and not a request for a hidden closure, hidden* T *supply, private access row, host equality row, direct aspect-reduction row, or off-packet continuation.*

Proof. Begin with the carrier row of Definition 1006.1. It exposes exactly the request, source, refusal, transport, continuation, provenance, and local naming rows of the finite packet $B = (G, S, R, H, C, P, N)$.

Use the NameCert obligations of Definition 1006.2 and the root readiness inventory of Definition 1006.58. These rows identify the finite BHist carrier together with hsame, Cont, Pkg, and local NameCert data before any listed hard-problem consumer reads the boundary.

For an unsupported downstream read, apply refusal-route exactness and refusal route determinacy, Definition 1006.56 and Definition 1006.50. The read is sent to the displayed refusal ledger R over H, C, P, N.

Finally use structural row conservation, Definition 1006.51. The structural rows conserve the visible source/refusal split and do not create any of the hidden rows excluded in the statement. □

Theorem 1006.62 (GapClosureBoundary root package consumer cut)**.** *Every downstream consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root packet factors through the displayed package and provenance surface* G, S, R, H, C, P, N*. A supported consumer factors through the displayed source route* $G \longrightarrow S$ *with* H, C, P, N*, an unsupported consumer factors through the refusal route to* R *with* H, C, P, N*, and no listed consumer may bypass these rows to a concealed closure theorem outside the local* BHist*,* Cont*,* Pkg*, and* NameCert *packet.*

Proof. Start from hard-problem consumer factorization, Definition 1006.47. It places the causal, finite-refutation, other-minds, aspect-chain, and solipsism-refusal consumers inside the same displayed packet.

For supported reads, use source-route totality and closure-request determinacy, Definition 1006.55 and Definition 1006.48. The consumer receives the source branch only through $G \longrightarrow S$ and the structural rows.

For unsupported reads, use refusal-route exactness and the consumer ledger exhaustion theorem, Definition 1006.56 and Definition 1006.57. They route the request to R and leave no third consumer route.

The consumer-exactness package of Definition 1006.54 supplies the cut: every exported consumer read is already a read of the displayed package and provenance surface, so there is no bypass to an external closure theorem. □

Theorem 1006.63 (GapClosureBoundary root NameCert replay)**.** *Replay along the displayed* Cont *rows and transport along the componentwise* hsame *rows preserve exactly the root boundary rows displayed by an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet. The replay may carry the source route* $G \longrightarrow S$ *or the refusal route to* R *through* H, C, P, N*, but it does not create quotient data, host equality data, ambient closure data, a hidden* T *supply, or any consumer-specific row outside the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *surface.*

Proof. Read the structural rows from Definition 1006.12. The rows H, C, P, N serve as transport, continuation, provenance, and naming data for the already displayed carrier rows.

For the supported branch, apply source-route totality, closure-request determinacy, and structural row conservation, Definition 1006.55, Definition 1006.48, and Definition 1006.51. Replay along Cont and transport along hsame keep the branch at $G \longrightarrow S$ with the same package rows.

For the refusal branch, apply refusal route determinacy and refusal route exactness, Definition 1006.50 and Definition 1006.56. The replay endpoint is the displayed refusal ledger R over the same H, C, P, N rows.

Finally use source-refusal orthogonality and hard-problem consumer factorization. They keep the two replayed branches inside the common boundary packet and prevent Cont, hsame, Pkg, or local NameCert rows from generating quotient, host equality, ambient closure, hidden supply, or consumer-specific data. □

Theorem 1006.64 (GapClosureBoundary source-refusal triad)**.** *Every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *boundary packet exposes exactly the request row* G*, the source row* S*, and the refusal row* R *before any hard-problem consumer reads the boundary. The remaining rows* H, C, P, N *transport, replay, package, and name that triad, but they do not add an internal closure witness, hidden* T *supply, private-access row, or direct aspect-reduction row.*

Proof. Begin with the carrier and NameCert rows, Definition 1006.1 and Definition 1006.2. They expose the finite packet G, S, R, H, C, P, N and separate carrier exposure from source-route and refusal-ledger duties.

Use source-route totality for supported reads and refusal-route exactness for unsupported reads, Definition 1006.55 and Definition 1006.56. Together they force every listed hard-problem consumer to see G with either the displayed source row S or the displayed refusal row R.

Finally apply refusal non-escape and structural row conservation, Definition 1006.3 and Definition 1006.51. These rows keep H, C, P, N from manufacturing a hidden closure, hidden supply, private access, or aspect-reduction coordinate after the triad has been exposed. □

Theorem 1006.65 (GapClosureBoundary consumer ledger factorization)**.** *Every hard-problem consumer access to an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet factors through the displayed rows* G, S, R, H, C, P, N*. A supported access factors through* $G \longrightarrow S$ *and an unsupported closure request factors through* $G \longrightarrow R$*, but no consumer can manufacture an internal closure witness from the ledger, transport, continuation, provenance, or naming rows.*

Proof. Start from hard-problem consumer factorization, Definition 1006.47. It places the listed causal, finite-refutation, other-minds, aspect-chain, and solipsism-refusal consumers inside the common packet G, S, R, H, C, P, N.

For supported access, apply source route totality and source observation totality, Definition 1006.55 and Definition 1006.40. The consumer receives the source branch only through the displayed route from G to S.

For unsupported closure requests, apply refusal route exactness and refusal ledger totality, Definition 1006.56 and Definition 1006.41. The request is sent to R through the same structural rows rather than to an internal closure witness.

The consumer-exactness package and structural row conservation, Definition 1006.54 and Definition 1006.51, exclude a third ledger route. Thus consumer access factors through the displayed rows and nowhere else. □

Theorem 1006.66 (GapClosureBoundary hard-problem route exhaustion)**.** *The finite witnessed-refutation and hard-problem sibling consumers of a* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet consume only the displayed boundary rows* G, S, R, H, C, P, N*. Their positive reads use the exposed source route, their negative reads use the exposed refusal route, and neither kind of read imports a consumer-specific closure witness outside the local* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *surface.*

Proof. Project the root boundary packet using Definition 1006.64. Before any sibling reads the boundary, the request, source, and refusal rows are visible as the common triad carried by H, C, P, N.

For the finite witnessed-refutation and hard-problem siblings, use hard-problem consumer factorization and consumer ledger exhaustion, Definition 1006.47 and Definition 1006.57. These theorems divide all listed sibling reads between the source branch and refusal branch.

Consumer ledger factorization then binds both branches to the displayed rows G, S, R, H, C, P, N. Since refusal non-escape and the consumer-exactness package exclude hidden closure and off-packet continuation rows, the siblings consume exactly the public GapClosureBoundary surface. □

Theorem 1006.67 (GapClosureBoundary root router exactness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, the root router exposes exactly two accepted routes for the hard-problem sibling consumers named in Definition 1006.2: the displayed source route from* G *to* S*, and the refusal-ledger route from* G *to* R*. Every accepted consumer route is one of these two routes over* H, C, P, N*, and no hidden closure branch, hidden* T *supply, private-access row, causal-necessity row, or aspect-reduction row is exported.*

Proof. Project the accepted carrier using Definition 1006.1. The packet contains the request row G, source row S, refusal row R, and the structural rows H, C, P, N.

Use Definition 1006.2 to read the router surface. Its source-route exactness clause accounts for the displayed route from G to S, and its refusal-ledger routing clause accounts for the route from G to R.

Finally apply Definition 1006.3. Since an off-source request is retained by R and the structural rows only transport, route, package, and name the same packet, the router has no third accepted branch outside G, S, R, H, C, P, N. □

Theorem 1006.68 (GapClosureBoundary consumer route totality)**.** *Every accepted read of a* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet by* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *is total over the displayed source/refusal split. A supported read factors through* $G \longrightarrow S$ *with* H, C, P, N*, an unsupported closure request factors through* $G \longrightarrow R$ *with* H, C, P, N*, and no listed consumer receives a route that bypasses both* S *and* R*.*

Proof. Begin with the sibling list in Definition 1006.2. The listed consumers are exactly the hard-problem consumers for which the boundary packet records provenance and local naming.

For supported reads, use Definition 1006.55. It requires the displayed route from G to S before any listed sibling consumes the structural rows.

For unsupported closure requests, use Definition 1006.56. It sends the request through $G \longrightarrow R$ with the same H, C, P, N rows and excludes hidden closure, hidden T supply, and off-packet continuation.

The two cases cover the consumer ledger by Definition 1006.57. Hence every accepted sibling route is total over the source/refusal split and no consumer-specific route remains. □

Theorem 1006.69 (GapClosureBoundary refusal ledger stability)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, replay along* C *and transport along* H *preserve the refusal ledger endpoint* R *for every unsupported hard-problem consumer request. The rows* P, N *may package and locally name that endpoint, but they cannot turn the refusal ledger into displayed source routing, internal closure, a hidden* T *supply, or a consumer-specific continuation.*

Proof. Start with Definition 1006.3. An unsupported closure request has no displayed route through S, so the accepted continuation sends it to the refusal row R.

Apply Definition 1006.11 and Definition 1006.12. They identify H, C, P, N as the transport, route, provenance, and local naming rows for the same refusal endpoint, not as generators of another source or closure coordinate.

The consumer route totality theorem then fixes the downstream alternatives. Since every accepted read is either the displayed source route or the refusal-ledger route, a replayed unsupported request remains at R and cannot become a hidden third route. □

Theorem 1006.70 (GapClosureBoundary hard-problem root route stability)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root packet* $B = (G, S, R, H, C, P, N)$*, the downstream hard-problem consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*,*

and $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *read the same root route surface. A supported read uses the displayed source route* $G \longrightarrow S$ *with* H, C, P, N*, an unsupported closure request uses the displayed refusal route* $G \longrightarrow R$ *with* H, C, P, N*, and no consumer receives a host closure row or hidden supply row outside the packet.*

Proof. Begin with the hub root inventory and root readiness theorems, Definition 1006.58 and Definition 1006.59. They bind the root packet to G, S, R, H, C, P, N and name the five downstream consumers as readers of that same surface.

For supported reads, apply source-route totality and root source exposure, Definition 1006.55 and Definition 1006.64. These keep the source branch at $G \longrightarrow S$ over the displayed structural rows.

For unsupported reads, apply refusal-route exactness and root ledger routing, Definition 1006.56 and Definition 1006.65. These send the request to R through the same packet rather than to a host closure.

The root-unblock package and root package consumer cut, Definition 1006.39 and Definition 1006.62, then make the route stable for all five consumers without adding any off-packet supply row. □

Theorem 1006.71 (GapClosureBoundary consumer refusal totality)**.** *Every unsupported hard-problem consumer request handled by an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root packet is refused through the displayed ledger row* R *of* $B = (G, S, R, H, C, P, N)$*. The refusal covers the five downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *uniformly, and it exports no private access row, direct aspect-reduction row, causal-necessity row, or hidden* T *supply.*

Proof. First use refusal-route exactness and refusal ledger totality, Definition 1006.56 and Definition 1006.41. They identify R as the endpoint for unsupported closure requests.

Next apply hard-problem consumer factorization and consumer ledger exhaustion, Definition 1006.47 and Definition 1006.57. These theorems place all five downstream consumers on the same source/refusal ledger.

Finally use root visible refusal audit and root NameCert replay, Definition 1006.61 and Definition 1006.63. The replay preserves only G, S, R, H, C, P, N, so unsupported consumers receive the visible refusal row and none of the excluded hidden rows. □

Theorem 1006.72 (GapClosureBoundary five-sibling unblock surface)**.** *The root packet* $B = (G, S, R, H, C, P, N)$ *supplies one common unblock surface for* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*. Each sibling consumes the displayed request row, the source-or-refusal branch, the componentwise transport rows, the continuation replay rows, provenance, and local naming; none receives a separate host closure theorem, hidden source, consumer-specific continuation, or off-packet bridge row.*

Proof. Start from the hub unblock surface theorem Definition 1006.60. It already identifies the common source/refusal split used by the downstream root package.

Use hard-problem route exhaustion and root route stability, Definition 1006.66 and Definition 1006.70. Together they bind every listed sibling read to G, S, R, H, C, P, N and divide the reads between the source branch and refusal branch.

The consumer refusal totality theorem Definition 1006.71 handles the unsupported branch uniformly. The consumer-exactness package then rules out a sibling-specific route outside the displayed packet, so the five consumers share one root unblock surface. □

Theorem 1006.73 (GapClosureBoundary recognizer source admission)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any recognizer-side source admission for a hard-problem sibling consumer must enter through the* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *acceptance route and land in the* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *tuple image before the boundary packet is consumed. The admitted read exposes the displayed request row* G*, the source row* S*, and the structural rows* H, C, P, N*, while the downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receive no source coordinate outside those rows.*

Proof. Start with the carrier of Definition 1006.1. Its recognizer-facing boundary is already the seven-row packet G, S, R, H, C, P, N.

Use the $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ gate through Definition 1048.3. An accepted source read must pass through the event-flow, source-channel, generated-recognizer, and certificate-gate rows before it can be consumed as compiler-layer evidence.

Next apply Definition 993.3. The admitted recognizer row lands in the displayed $\mathsf{FiniteReflectionTuple}^{\uparrow}$ image and is read back from that tuple, so the source branch available to the boundary is the displayed route $G \longrightarrow S$ over H, C, P, N.

Finally use root source exactness, Definition 1006.25. The listed hard-problem consumers may consume the admitted source branch only through the displayed boundary rows, so no additional source coordinate is exported. □

Theorem 1006.74 (GapClosureBoundary recognizer refusal stability)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, if a recognizer-side request does not land in the displayed* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *tuple image after* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *acceptance, then the boundary routes the request to the refusal ledger* R*. The refusal route is stable under the structural rows* H, C, P, N *and cannot become internal closure, hidden* T *supply, private access, causal necessity, direct aspect reduction, or an off-packet continuation for any listed sibling consumer.*

Proof. Begin with the non-evidence refusal theorem for $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$, Definition 1048.4. It keeps report-like source, host compiler state, proof text, and raw classifier quotients outside accepted evidence when the generated recognizer route does not pass the gate.

The tuple boundary of Definition 993.3 then prevents a failed landing from being reread as a reflected boundary tuple. Thus the request has no displayed route through $G \longrightarrow S$.

Apply refusal route exactness and refusal ledger stability, Definition 1006.56 and Definition 1006.69. The request is routed to R through H, C, P, N and the same rows preserve that refusal endpoint for all listed sibling consumers. □

Theorem 1006.75 (GapClosureBoundary recognizer consumer exhaustion)**.** *For the downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, the combined* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *and* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *landing route exhausts the recognizer-facing* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consumer surface. A successful recognizer landing is consumed as* $G \longrightarrow S$ *with* H, C, P, N*; a failed or unsupported landing is consumed as* $G \longrightarrow R$ *with* H, C, P, N*; no third recognizer, tuple, or sibling-specific route is exported.*

Proof. First account for successful recognizer landings. By Definition 1006.73, accepted recognizer evidence reaches the boundary only after the $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ gate and the $\mathsf{FiniteReflectionTuple}^{\uparrow}$ tuple readback, and it is then consumed through the displayed source branch.

Next account for failed or unsupported landings. By Definition 1006.74, these requests are routed to the refusal ledger R over the same structural rows.

It remains to rule out another consumer path. Use hard-problem route stability and consumer ledger exhaustion, Definition 1006.70 and Definition 1006.57. They keep all five listed consumers inside the same source/refusal split, so the recognizer-facing route is exhausted by the two displayed branches. □

Theorem 1006.76 (GapClosureBoundary bridge precondition surface)**.** *The bridge-facing precondition surface of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *is exactly the displayed packet* $B = (G, S, R, H, C, P, N)$ *together with four rows: the source/refusal split through* $G \longrightarrow S$ *and* $G \longrightarrow R$*, the consumer exactness package for* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, and the root replay rows carried by* Cont*,* hsame*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$*. No hidden closure row, hidden* T *supply, host equality row, quotient datum, or off-packet continuation belongs to that precondition surface.*

Proof. Start with the carrier definition and NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the seven displayed coordinates G, S, R, H, C, P, N before any bridge-facing consumer reads the boundary.

Use source-route totality and refusal-route exactness, Definition 1006.55 and Definition 1006.56. These theorems identify the two public branches as the displayed route to S and the displayed route to R.

Finally combine consumer exactness with root NameCert replay, Definition 1006.54 and Definition 1006.63. The first theorem fixes which downstream consumers may read the boundary, and the second keeps replay inside the same Cont, hsame, Pkg, and local $\mathsf{NameCert}$ rows. The excluded hidden rows therefore have no coordinate in the precondition surface. □

Theorem 1006.77 (GapClosureBoundary TasteGate boundary)**.** *Any Lean TasteGate or* $\mathsf{StandardBridge}$ *work for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *must preserve the same seven packet rows* G, S, R, H, C, P, N *and the same displayed source/refusal routes. Its paper-side boundary may read carrier admission, the source route, the refusal route, consumer exactness, root package replay, provenance, and local naming, but it cannot read hidden closure, hidden* T *supply, host equality, quotient data, or any continuation outside the displayed packet.*

Proof. Project the bridge precondition surface of Definition 1006.76. The only available packet rows are G, S, R, H, C, P, N, and the only branch rows are the displayed source and refusal routes.

Use structural row conservation and the root visible refusal audit, Definition 1006.51 and Definition 1006.61. They keep Cont, hsame, Pkg, and local $\mathsf{NameCert}$ replay tied to the visible packet and prevent the structural rows from adding host equality, quotient data, or hidden supply.

The TasteGate or $\mathsf{StandardBridge}$ reader therefore has no additional coordinate to inspect. Its admissible input is the seven-row packet with the displayed branch split, consumer exactness, root replay, provenance, and naming rows. □

Theorem 1006.78 (GapClosureBoundary obligation status assembly)**.** *The carrier rows, source/refusal split, root-unblock package, consumer-exactness package, root NameCert replay rows, bridge precondition surface, and TasteGate boundary assemble the paper-side* $\mathsf{obligation}$ *surface for* $\mathsf{GapClosureBoundary}^{\uparrow}$*. The assembly justifies recording while leaving the verification axis at* $\mathsf{unformalized}$ *until a separate Lean target checks the corresponding root package.*

Proof. Begin with the displayed carrier and branch obligations. Definition 1006.2, Definition 1006.55, and Definition 1006.56 identify the accepted packet and its two public routes.

Next apply the root-unblock and consumer-exactness packages, Definition 1006.39 and Definition 1006.54. These package the listed hard-problem consumers as readers of exactly that source/refusal surface.

Add the root replay and bridge boundary rows from Definition 1006.63, Definition 1006.76, and Definition 1006.77. The assembled surface records every paper obligation needed by the boundary while making no Lean verification claim, so the closure and verification axes remain independent. □

Theorem 1006.79 (GapClosureBoundary scoped dependency inventory)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the packet displays its* BHist *carrier rows together with* hsame *transport* H*,* Cont *route* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *before any consumer reads the boundary. The dependency inventory is therefore the seven-row packet* G, S, R, H, C, P, N*, and it contains no hidden closure, hidden* T *supply, host equality row, quotient datum, or off-packet continuation.*

Proof. Start with the carrier and NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the finite boundary packet as the request, source, refusal, transport, continuation, provenance, and naming rows.

Next apply the consumer exactness package. Definition 1006.54 requires the hard-problem consumers to read the displayed carrier before any downstream route is used.

Finally use root NameCert replay, Definition 1006.63. The replay keeps H, C, P, N attached to the same BHist carrier, so the dependency inventory is already visible before the consumer read. □

Theorem 1006.80 (GapClosureBoundary scoped source-refusal exactness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the source branch*

$$G \longrightarrow S$$

and the refusal branch

$$G \longrightarrow R$$

are exactly the two scoped routes over H, C, P, N*. A supported read uses the source route, an unsupported read uses the refusal route, and no third route over the same packet is accepted.*

Proof. First expose the scoped dependency inventory. Definition 1006.79 fixes G, S, R, H, C, P, N as the visible packet before route selection.

For the source branch, apply source-route totality from Definition 1006.55. A supported read of G reaches the displayed source row S and remains transported, routed, packaged, and named by H, C, P, N.

For the refusal branch, apply refusal route exactness from Definition 1006.56. An unsupported read of G reaches the displayed refusal row R over the same structural rows.

The consumer exactness package then removes any residual case. Since every hard-problem consumer read is already assigned to the source/refusal surface, no additional accepted route remains. □

Theorem 1006.81 (GapClosureBoundary scoped consumer non-escape)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the sibling consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *can read only the scoped source/refusal surface of* G, S, R, H, C, P, N*. They cannot read hidden closure, hidden* T *supply, host equality, quotient data, or any continuation outside the displayed packet.*

Proof. Begin with scoped source-refusal exactness. Definition 1006.80 assigns each accepted read to $G \longrightarrow S$ or $G \longrightarrow R$ over H, C, P, N.

Use refusal non-escape and the consumer exactness package, Definition 1006.3 and Definition 1006.54. These rows remove hidden closure, hidden T supply, and consumer-specific routes from the accepted packet.

Close with root NameCert replay. Definition 1006.63 keeps replay inside the displayed Cont, hsame, Pkg, and local $\mathsf{NameCert}$ rows, so host equality, quotient data, and off-packet continuations have no readable coordinate. □

Theorem 1006.82 (GapClosureBoundary source packet)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the root source packet exposes exactly the following rows over the displayed* BHist *surface:*

1. *the request row* G *and the source observation or socket row* S*;*
2. *the refusal ledger* R *for every closure-facing read that lacks a displayed source route;*
3. *the hard-problem consumer rows for* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*;*
4. *the componentwise* hsame *transport rows in* H *and the displayed* Cont *routes in* C *from* G *to* S *or* R*;*
5. *the provenance row* P *as* Pkg *data and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*.*

No accepted consumer reads a source, refusal, transport, provenance, or local naming coordinate outside G, S, R, H, C, P, N*.*

Proof. Project the carrier of Definition 1006.1. It gives the finite BHist packet G, S, R, H, C, P, N and assigns the structural meanings of H, C, P, N.

Use source route totality and refusal route exactness from Definition 1006.55 and Definition 1006.56. They place every supported read on $G \longrightarrow S$ and every unsupported closure request on $G \longrightarrow R$.

Finally apply the consumer exactness package and recognizer consumer exhaustion, Definition 1006.54 and Definition 1006.75. These results keep the listed consumers, recognizer landing rows, Pkg provenance, and local $\mathsf{NameCert}$ entry inside the displayed packet, so no external coordinate is exported. □

Theorem 1006.83 (GapClosureBoundary classifier stability)**.** *Let* $B = (G, S, R, H, C, P, N)$ *be an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier. Componentwise* hsame *transport in* H *together with the displayed* Cont *routes in* C *preserves the source/refusal classifier: a supported read remains routed through* $G \longrightarrow S$*, and an unsupported closure request remains routed through* $G \longrightarrow R$*. The transport cannot introduce hidden host refusal, private source selection, unrecorded consumer authority, hidden* T *supply, causal necessity, direct aspect reduction, or an off-packet continuation.*

Proof. Start with the source packet of Definition 1006.82. The only classifier-visible rows are the request, source, refusal, transport, continuation, provenance, consumer, and naming rows already displayed by B.

For supported reads, apply source route totality and downstream branch determinacy. They force the consumer to expose $G \longrightarrow S$ before reading H, C, P, N, and they prevent a structural row from selecting a private source.

For unsupported closure requests, apply refusal route exactness and refusal ledger stability. The same hsame and Cont rows carry the request to R and preserve the refusal endpoint rather than turning it into host-level refusal data or concealed closure supply.

The remaining exclusions follow from consumer ledger exhaustion and recognizer consumer exhaustion. Both surfaces exhaust accepted hard-problem reads by the displayed source/refusal split, so no unrecorded consumer authority or off-packet continuation remains available. □

Theorem 1006.84 (GapClosureBoundary NameCert obligation surface)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *obligation surface consists of exactly five obligations:*

1. *carrier exposure: the certificate exposes the finite* BHist *rows* G, S, R, H, C, P, N*;*
2. *source-route admission: every supported read enters through the displayed route* $G \longrightarrow S$ *with the* hsame*,* Cont*, and* Pkg *rows carried by* H, C, P*;*
3. *refusal-ledger exactness: every unsupported closure request enters* R *through the displayed refusal route and receives no hidden closure or hidden* T *supply;*
4. *consumer-boundary exhaustion: the five listed hard-problem consumers read only the displayed source/refusal packet and its structural rows;*
5. *non-escape of host/global refusal data: the local refusal ledger is not a host refusal oracle, global closure refusal, private source selector, or consumer-authority row.*

These five obligations are sufficient for the root $\mathsf{GapClosureBoundary}^{\uparrow}$ *surface consumed by the sibling hard-problem packets.*

Proof. Carrier exposure is the direct projection of Definition 1006.1. The packet exposes G, S, R, H, C, P, N and no additional public coordinate.

Source-route admission follows from source route totality and recognizer source admission. A supported read must pass through the displayed source branch before the hard-problem consumer or recognizer landing surface consumes the packet.

Refusal-ledger exactness follows from refusal route exactness and recognizer refusal stability. An unsupported closure request lands in R through the displayed continuation rows and remains refusal data rather than closure supply.

Consumer-boundary exhaustion follows from the consumer exactness package, source/refusal consumer ledger, and recognizer landing obligations. Those sibling rows cover the listed consumers by the source/refusal split.

Non-escape follows from classifier stability above together with refusal non-escape. Since hsame, Cont, Pkg, and NameCert preserve the displayed packet, the local ledger cannot become host refusal data, global refusal data, private source selection, or unrecorded consumer authority. □

Theorem 1006.85 (GapClosureBoundary root source-classifier totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *source-classifier read of a carrier* $B = (G, S, R, H, C, P, N)$*, the read factors through the finite packet rows* G, S, R, H, C, P, N*. If the read is supported, its displayed route is* $G \longrightarrow S$*; if it is unsupported as a closure request, its displayed route is* $G \longrightarrow R$*. No accepted source-classifier read has a third route outside the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Proof. Begin with the source packet of Definition 1006.82. It exposes the finite rows G, S, R, H, C, P, N as the whole root packet and states that no accepted consumer reads a source, refusal, transport, provenance, or local naming coordinate outside those rows.

For supported reads, apply Definition 1006.55. The source-classifier must expose the displayed route $G \longrightarrow S$ before any listed hard-problem sibling consumes H, C, P, N.

For unsupported closure requests, apply Definition 1006.56. The request is routed through $G \longrightarrow R$ with the same structural rows, and the excluded closure and hidden T coordinates are not exported.

Finally use classifier stability, Definition 1006.83. Componentwise hsame transport and the displayed Cont routes preserve the source/refusal split, so the two displayed routes exhaust the accepted source-classifier reads. □

Theorem 1006.86 (GapClosureBoundary refusal-ledger route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, unsupported closure requests are exhausted by the refusal-ledger route* $G \longrightarrow R$ *over* H, C, P, N*. Such a request cannot become hidden closure, hidden* T *supply, host equality, quotient data, or an off-packet continuation.*

Proof. Start with the unsupported side of Definition 1006.85. Once the displayed source route is absent, the only accepted classifier route is the refusal route $G \longrightarrow R$.

Apply Definition 1006.56 and Definition 1006.69. These results keep the unsupported request at the ledger endpoint R under the same hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

The remaining exclusions come from Definition 1006.83. The transported refusal route cannot introduce hidden host refusal, hidden T supply, private source selection, unrecorded consumer authority, or an off-packet continuation; hence it also cannot supply host equality or quotient data not present in G, S, R, H, C, P, N. □

Theorem 1006.87 (GapClosureBoundary consumer NameCert readiness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the source-classifier route totality and refusal-ledger route totality supply the* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *readiness surface consumed by* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*. Each listed consumer receives exactly the displayed source/refusal packet with its* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows, and receives no off-packet closure resource.*

Proof. Read the local obligation surface in Definition 1006.84. Its five obligations require carrier exposure, source-route admission, refusal-ledger exactness, consumer-boundary exhaustion, and non-escape of host or global refusal data.

The source-classifier route is supplied by Definition 1006.85. It gives the supported route $G \longrightarrow S$ and the unsupported route $G \longrightarrow R$ inside the same finite packet.

The refusal side is supplied by Definition 1006.86. Unsupported closure requests remain ledger entries and cannot become hidden closure, hidden T supply, host equality, quotient data, or off-packet continuation.

Therefore the listed consumers read exactly the readiness surface named by the local $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ row: the displayed BHist packet, its hsame transports, its Cont routes, and its Pkg provenance. No stronger closure resource is exported to any hard-problem sibling. □

Theorem 1006.88 (GapClosureBoundary root request source totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every root consumer that receives supported source evidence does so through the displayed source route*

$$G \longrightarrow S$$

and through no other source-bearing coordinate. The rows H, C, P, N *carry only transport, continuation, provenance, and local naming for that route; they do not create hidden closure supply, host equality, quotient data, private source selection, or an off-packet continuation.*

Proof. Read the carrier from Definition 1006.1. Its root boundary coordinates are exactly the request row G, source row S, refusal row R, and the structural rows H, C, P, N.

Apply source-route totality from Definition 1006.55. A supported hard-problem consumer must expose $G \longrightarrow S$ before any structural row is read.

Finally use structural row conservation from Definition 1006.51. The rows H, C, P, N preserve the already displayed source route as transport, continuation, provenance, and local naming, so no second source route or concealed closure coordinate is available. □

Theorem 1006.89 (GapClosureBoundary root refusal consumer totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every root consumer whose closure request is not supported is exhausted by the displayed refusal route*

$$G \longrightarrow R$$

with H, C, P, N *carrying only transport, continuation, provenance, and local naming. Such a consumer receives no source substitute, hidden closure row, hidden* T *supply, host equality, quotient data, private access row, or off-packet continuation.*

Proof. Start with the carrier boundary in Definition 1006.1. The only off-source endpoint available to the root packet is the refusal row R.

Use refusal route exactness and refusal ledger stability, Definition 1006.56 and Definition 1006.69. They route unsupported closure requests through $G \longrightarrow R$ and preserve that refusal endpoint under transport and replay.

Apply structural row conservation to close the alternatives. Since H, C, P, N only transport, route, package, and name the same refusal endpoint, the unsupported consumer cannot obtain a third branch outside G, S, R, H, C, P, N. □

Theorem 1006.90 (GapClosureBoundary root structural NameCert exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, the root* $\mathsf{NameCert}_{\mathsf{GapClosureBound}}$ *surface is exhausted by the source route* $G \longrightarrow S$*, the refusal route* $G \longrightarrow R$*, and the structural rows* H, C, P, N*. The structural rows carry only transport, continuation, provenance, and naming for those two routes, so the root consumer interface has no third source/refusal branch and no additional closure, equality, quotient, or off-packet resource.*

Proof. First expose the finite packet by Definition 1006.1. The NameCert surface can only name the displayed request, source, refusal, transport, continuation, provenance, and local naming coordinates.

The source side is exhausted by Definition 1006.88. The refusal side is exhausted by Definition 1006.89.

It remains to account for the structural rows. Apply Definition 1006.51: H, C, P, N conserve the selected source or refusal branch instead of adding another coordinate. Thus the root structural NameCert surface is exactly the displayed two-route package. □

Theorem 1006.91 (GapClosureBoundary admission route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every accepted source admission for the root boundary enters through the displayed source route* $G \longrightarrow S$ *with the* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows carried by* G, S, H, C, P, N*. The admission cannot use hidden closure supply, host equality, quotient data, private source selection, or any off-packet continuation.*

Proof. Begin with the source packet of Definition 1006.82. It exposes the accepted root inventory as the finite rows G, S, R, H, C, P, N and states that no source, refusal, transport, provenance, or local naming coordinate is available outside that packet.

For the admission branch, apply Definition 1006.85. A supported source-classifier read is routed through $G \longrightarrow S$ and not through a third carrier coordinate.

Finally use classifier stability, Definition 1006.83. Componentwise hsame transport and the displayed Cont routes preserve the source/refusal split, so the structural rows can carry the admitted route but cannot manufacture hidden closure supply, host equality, quotient data, private source selection, or an off-packet continuation. □

Theorem 1006.92 (GapClosureBoundary refusal route stability)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every unsupported closure request remains on the displayed refusal ledger route* $G \longrightarrow R$ *under the same* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Transport, replay, package, and local naming may preserve that refusal endpoint, but they cannot convert it into source admission, hidden closure supply, host equality, quotient data, or an off-packet continuation.*

Proof. Start with refusal route exactness, Definition 1006.56. Once the source route is absent, the unsupported request is routed through $G \longrightarrow R$ with the same structural rows H, C, P, N.

Apply refusal-ledger stability, Definition 1006.69. Replay along C and transport along H preserve the endpoint R, while P and N package and name that endpoint rather than replacing it with another route.

The remaining exclusions follow from classifier stability. The transported refusal route remains a refusal ledger entry under the local BHist, hsame, Cont, Pkg, and NameCert surface, so it cannot become source admission, hidden closure supply, host equality, quotient data, or an off-packet continuation. □

Theorem 1006.93 (GapClosureBoundary consumer readiness exhaustion)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the hard-problem consumers* CausalCommitment$^{\uparrow}$*,* FiniteWitnessedRefutation$^{\uparrow}$*,* OtherMindsCommitment$^{\uparrow}$*,* AspectChain$^{\uparrow}$*, and* SolipsismGapRefusal$^{\uparrow}$ *receive only the admitted source/refusal packet carried by* G, S, R, H, C, P, N*. A supported consumer receives the admitted source route, an unsupported closure request receives the refusal route, and no consumer receives an off-packet closure resource.*

Proof. First assemble the packet boundary. Definition 1006.85 exhausts accepted source-classifier reads by the displayed source and refusal routes, and Definition 1006.91 fixes the supported admission side as $G \longrightarrow S$.

Next account for unsupported consumers. Definition 1006.92 keeps each unsupported closure request on $G \longrightarrow R$ under the same hsame, Cont, Pkg, and local NameCert rows.

Finally apply consumer NameCert readiness, Definition 1006.87. The listed hard-problem consumers read exactly the displayed BHist packet and its structural rows, so the source/refusal packet is exhausted and no off-packet closure resource is exported. □

Definition 1006.94 (GapClosureBoundary root readback packet)**.** *The* GapClosureBoundary$^{\uparrow}$ *root readback packet is the finite* BHist *surface*

$$Q = (B, S_0, R_0, T_0, H_0, C_0, P_0, N_0)$$

where $B = (G, S, R, H, C, P, N)$ *is an accepted* GapClosureBoundary$^{\uparrow}$ *carrier,* S_0 *is the displayed source-readback row for* S*,* R_0 *is the displayed refusal-readback row for* R*,* T_0 *is the downstream*

consumer handoff row, H_0 records componentwise hsame *transport for the source and refusal readbacks, C_0 records* Cont *replay of the source and refusal branches, P_0 records* Pkg *provenance through the root-unblock package and the finite witnessed refutation route of Definition 576.7, and N_0 is the handoff* NameCert *row.*

Theorem 1006.95 (GapClosureBoundary root NameCert handoff)**.** *Every accepted root readback packet $Q = (B, S_0, R_0, T_0, H_0, C_0, P_0, N_0)$ hands off the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *NameCert surface to downstream hard-problem consumers only after the source branch $G \longrightarrow S$ or the refusal branch $G \longrightarrow R$ has been read through B. The handoff row T_0 may cite the root-unblock package of Definition 1006.39 and the finite witnessed refutation boundary route of Definition 576.7, but it receives no host closure row, hidden* T *supply, private access row, or off-packet continuation.*

Proof. Start from the accepted carrier $B = (G, S, R, H, C, P, N)$ inside Definition 1006.94. The root readback packet adds only source readback, refusal readback, handoff, transport, replay, provenance, and local naming rows.

For the supported branch, apply source-route totality and the root-unblock package, Definition 1006.55 and Definition 1006.39. These force the source readback S_0 to be a replay of $G \longrightarrow S$ through the displayed structural rows.

For the unsupported branch, apply refusal-route exactness and the finite witnessed refutation crosslink, Definition 1006.56 and Definition 576.7. These force the refusal readback R_0 to remain a refusal boundary rather than a universal closure principle.

The handoff row T_0 is then packaged by P_0 and named by N_0. Since no row of Q is a host closure, hidden supply, private access, or off-packet continuation, the downstream handoff exports only the displayed root NameCert surface. □

Theorem 1006.96 (GapClosureBoundary root consumer locality)**.** *Every downstream consumer of the root readback packet $Q = (B, S_0, R_0, T_0, H_0, C_0, P_0, N_0)$ is local to the displayed* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier, its source/refusal readback rows, and the handoff* NameCert *surface. A consumer may inspect $B, S_0, R_0, T_0, H_0, C_0, P_0, N_0$ and may follow the hard-problem consumer factorization of Definition 1006.65; it cannot bypass those rows to inspect a hidden root source, a host equality, a private other-minds access row, a direct aspect-reduction row, or an ambient closure theorem.*

Proof. Project the packet Q. Its consumer-facing coordinates are exactly the accepted boundary carrier, the two readback rows, the handoff row, transport, replay, provenance, and local naming.

Apply root package consumer cut, Definition 1006.62. It forces every downstream consumer through the displayed package and provenance surface of B.

Apply hard-problem consumer factorization, Definition 1006.65. The consumer route is one of the displayed source/refusal branches and not a separate consumer-specific route.

Finally use the handoff theorem Definition 1006.95. The handoff preserves the same source/refusal readback under hsame, Cont, Pkg, and NameCert rows, so no hidden source, host equality, private access, direct reduction, or ambient closure theorem can be read outside Q. □

Theorem 1006.97 (GapClosureBoundary supported route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$, every supported boundary route is displayed through the carrier rows, the classifier surface, the refusal ledger, downstream consumer rows, componentwise transports, displayed* Cont *routes,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row. A supported read therefore reaches a hard-problem consumer only through the displayed source route and never through an unrecorded closure coordinate.*

Proof. Start with the carrier and NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the finite rows G, S, R, H, C, P, N and require source-route exactness before any supported read is consumed.

Use the classifier and source-route surfaces recorded by Definition 1006.7 and Definition 1006.10. These show that an accepted supported route reads the displayed source row S and remains inside the finite carrier.

Finally apply the downstream handoff theorems, including Definition 1006.112 and Definition 1006.114. The structural rows H, C, P, N only transport, replay, package, and name the supported route, so no hidden closure coordinate is exported. □

Theorem 1006.98 (GapClosureBoundary refusal stability route)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, an unsupported read of* G *remains in the refusal ledger* R *under componentwise* hsame *transport. The displayed route may replay the refusal through* H, C, P, N*, but it cannot turn the unsupported read into internal closure, hidden* T *supply, host quotient data, or a private source selection.*

Proof. Begin with an unsupported read of the request row G. Since no displayed source route through S is present, the displayed source precondition of Definition 1006.4 is not met.

Apply refusal-ledger routing and transport, Definition 1006.11 and Definition 1006.5. They send the unsupported read to R and carry it only through the componentwise hsame rows, displayed Cont route, provenance, and local name.

The refusal non-escape theorem Definition 1006.3 excludes internal closure, hidden T supply, host quotient data, and private source selection. Thus the refusal route is stable under the displayed transports. □

Theorem 1006.99 (GapClosureBoundary consumer nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *route receives only the displayed rows* G, S, R, H, C, P, N*, the classifier and refusal ledgers, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data. It receives no hidden global classifier, theorem-proof row, host quotient, or unrecorded completion boundary.*

Proof. First expose the consumer through Definition 1006.114. A supported consumer receives the source handoff, and an off-source consumer receives the refusal handoff; both handoffs use the same finite carrier rows.

Combine supported route totality with refusal stability. The supported case is exhausted by Definition 1006.97, while the unsupported case is exhausted by Definition 1006.98.

It remains to rule out external replacements. The carrier, classifier exhaustion, and ledger nonescape theorems give no coordinate for a hidden global classifier, theorem-proof row, host quotient, or completion boundary outside G, S, R, H, C, P, N. Hence every downstream consumer remains inside the displayed route completion surface. □

Theorem 1006.100 (GapClosureBoundary root recognizer admission totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every root recognizer admission that reaches a downstream hard-problem consumer must first pass through the* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *acceptance route and then land in the displayed* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *image. The admitted branch is consumed as the source route* $G \longrightarrow S$ *with the displayed transport, continuation, provenance, and local naming rows; no hidden source selector, private compiler state, or off-packet recognition channel is exported.*

Proof. Start with the carrier of Definition 1006.1. The root boundary exposes only the packet G, S, R, H, C, P, N.

Apply recognizer source admission, Definition 1006.73. It forces successful recognizer evidence through $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ and then through the $\mathsf{FiniteReflectionTuple}^{\uparrow}$ tuple image before the boundary packet is consumed.

Finally use admission route totality, Definition 1006.91. The accepted branch is the displayed $G \longrightarrow S$ route under H, C, P, N, so no hard-problem consumer receives an additional source coordinate. □

Theorem 1006.101 (GapClosureBoundary root refusal ledger coverage)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every root recognizer request that fails to land through the* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *and* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *admission route is covered by the displayed refusal ledger* R*. The GAP residue available to downstream hard-problem consumers is therefore the finite refusal branch* $G \longrightarrow R$ *with the same* H, C, P, N *structural rows, not hidden closure supply, host equality, quotient data, private source access, or an off-packet continuation.*

Proof. Begin with recognizer refusal stability, Definition 1006.74. A failed tuple landing cannot be reread as accepted source evidence, so the request is routed to the refusal ledger.

Use refusal route stability, Definition 1006.92. The rows H, C, P, N preserve the endpoint R and package it as the displayed GAP residue rather than replacing it with another route.

The coverage is exhaustive by consumer readiness exhaustion, Definition 1006.93. Unsupported closure requests for the listed hard-problem consumers remain inside the same refusal branch. □

Theorem 1006.102 (GapClosureBoundary root NameCert consumer exhaustion)**.** *For the downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$*, the root* $\mathsf{GapClosureBoundary}^{\uparrow}$ *NameCert surface is exhausted by the displayed source row, refusal ledger, transport rows, continuation routes, package provenance, and local naming row of* $B = (G, S, R, H, C, P, N)$*. A consumer may read admitted source evidence through* $G \longrightarrow S$ *or refusal coverage through* $G \longrightarrow R$*, but it receives no separate hard-problem carrier, hidden closure oracle, host-level equality, or unnamed route.*

Proof. First assemble the accepted branch. By Definition 1006.100, recognizer admission reaches the boundary only as the displayed source route $G \longrightarrow S$.

Next assemble the refusal branch. By Definition 1006.101, every failed or unsupported recognizer request is carried by $G \longrightarrow R$ under the same structural rows.

It remains to account for the consumer surface. Apply recognizer consumer exhaustion, Definition 1006.75, and consumer readiness exhaustion, Definition 1006.93. Together they show that the listed hard-problem consumers read only G, S, R, H, C, P, N and no sibling-specific or off-packet NameCert row. □

Theorem 1006.103 (GapClosureBoundary source route completeness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every supported downstream consumer route for* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *factors through the displayed source row* S *and the concrete rows* G, H, C, P, N*. The consumer may read the branch only as*

$$G \longrightarrow S$$

transported, replayed, packaged, and named by H, C, P, N*; it receives no separate source carrier, private host witness, or consumer-specific route outside the accepted packet.*

Proof. Start with the accepted carrier and NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the packet G, S, R, H, C, P, N before any hard-problem consumer reads the boundary.

Apply source route totality, Definition 1006.55. A supported read reaches the consumer surface only after the displayed route $G \longrightarrow S$ is available.

Finally use source observation totality and structural row conservation, Definition 1006.40 and Definition 1006.51. These keep H, C, P, N as transport, replay, package, and local naming rows for the same source route, so the supported consumer factors through the concrete packet and not through an extra carrier. □

Theorem 1006.104 (GapClosureBoundary refusal branch determinacy)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every off-source closure request by a listed downstream consumer has the unique refusal endpoint* R*. The request is routed through* G *and the structural rows* H, C, P, N *to the refusal branch*

$$G \longrightarrow R$$

and cannot split into hidden T *supply, internal closure, private access, host equality, quotient data, or an off-packet continuation.*

Proof. Begin with refusal route determinacy, Definition 1006.50. When the displayed source route is absent, the public boundary sends the request to the refusal row R.

Use refusal route exactness and refusal non-escape, Definition 1006.56 and Definition 1006.3. They identify $G \longrightarrow R$ as the accepted endpoint and remove hidden closure, hidden T supply, private access, and off-packet continuation as exits.

The structural rows are then fixed by structural row conservation, Definition 1006.51. Since H, C, P, N only preserve, replay, package, and name the displayed branch, the refusal endpoint cannot split into another source or an internal closure witness. □

Theorem 1006.105 (GapClosureBoundary NameCert consumer totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the concrete* $\mathsf{NameCert}$ *consumer packet for the listed downstream hard-problem chapters is total over the two displayed branches. Each consumer route is exhausted by either the source branch* $G \longrightarrow S$ *of Definition 1006.103 or the refusal branch* $G \longrightarrow R$ *of Definition 1006.104, with* H, C, P, N *as the only transport, continuation, provenance, and local naming rows.*

Proof. First assemble the source side. By Definition 1006.103, every supported consumer factors through the concrete source row S and the rows G, H, C, P, N.

Next assemble the refusal side. By Definition 1006.104, every off-source closure request reaches the unique refusal endpoint R through the same structural rows.

It remains to show that these two branches exhaust the NameCert packet. Apply the carrier obligations of Definition 1006.2 and the non-escape content of Definition 1006.3. The first fixes the consumer surface to G, S, R, H, C, P, N, and the second removes hidden closure, hidden T supply, and off-packet continuation. Thus every listed consumer route is accounted for by the displayed source branch or the displayed refusal branch. □

Theorem 1006.106 (GapClosureBoundary finite source/refusal classifier)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, the public root classifier exposes exactly the two routes*

$$G \longrightarrow S$$

and

$$G \longrightarrow R$$

over the structural rows H, C, P, N. The first route is the supported source route, the second route is the unsupported refusal route, and the accepted classifier exports no third public route, hidden closure branch, private source row, or off-packet continuation.

Proof. Start with carrier exposure and local obligations, Definition 1006.1 and Definition 1006.2. They identify the public root packet with the rows G, S, R, H, C, P, N before any consumer reads the boundary.

Use source/refusal orthogonality, Definition 1006.46. The supported side has to display the source row as $G \longrightarrow S$, while the unsupported side is routed to the refusal row as $G \longrightarrow R$.

Finally apply the consumer totality surface of Definition 1006.105. Since every listed consumer route is exhausted by one of these two branches with H, C, P, N as structural support, the classifier has no remaining public branch to expose. □

Theorem 1006.107 (GapClosureBoundary consumer source/refusal readiness)**.** *Every listed hard-problem consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$ is ready only after the same source/refusal split of Definition 1006.106 has been displayed. A supported consumer reads the source route $G \longrightarrow S$, an unsupported closure request reads the refusal route $G \longrightarrow R$, and no consumer receives an off-packet closure row, private witness row, hidden* T *supply, quotient row, or sibling-specific carrier.*

Proof. First factor the downstream surface by Definition 1006.47. The causal, finite-refutation, other-minds, aspect-chain, and solipsism-refusal consumers are recorded through the same packet G, S, R, H, C, P, N rather than through independent consumer carriers.

Next use Definition 1006.106. It gives the only two public routes available to those consumers: the supported source route and the unsupported refusal route.

The readiness claim follows from consumer readiness exhaustion, Definition 1006.93. That theorem removes off-packet closure resources from the consumer interface, so each listed consumer is ready precisely at the displayed source/refusal split. □

Theorem 1006.108 (GapClosureBoundary root refusal NameCert exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier $B = (G, S, R, H, C, P, N)$, the local root* $\mathsf{NameCert}$ *surface is exhausted by carrier exposure, supported source routing, unsupported refusal routing, and the structural transport, provenance, replay, and naming rows H, C, P, N. Thus the root certificate has no additional local obligation beyond exposing G, S, R, H, C, P, N, routing supported requests through $G \longrightarrow S$, routing unsupported requests through $G \longrightarrow R$, and transporting, packaging, replaying, and naming those routes.*

Proof. Read the local surface from Definition 1006.1 and Definition 1006.2. These anchors provide the carrier row and the certificate duties for source routing, refusal routing, transport, provenance, replay, and local naming.

Apply source/refusal orthogonality together with the finite classifier, Definition 1006.46 and Definition 1006.106. They separate the supported source route from the unsupported refusal route and show that both routes remain inside the same public packet.

It remains to account for consumers. Use hard-problem consumer factorization and consumer readiness exhaustion, Definition 1006.47 and Definition 1006.93. These close the local surface by assigning all listed consumer reads to the displayed source/refusal packet and by excluding off-packet closure rows. □

Theorem 1006.109 (GapClosureBoundary source-refusal decision surface)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every supported read factors through the displayed source route*

$$G \longrightarrow S,$$

and every unsupported read factors through the displayed refusal route

$$G \longrightarrow R.$$

The decision surface is the concrete packet governed by Definition 1006.55, Definition 1006.56, Definition 1006.54, and the TasteGate row recorded in the closure status for this chapter.

Proof. Start with the carrier in Definition 1006.1. It exposes exactly the request row, source row, refusal row, transport row, continuation row, provenance row, and local naming row G, S, R, H, C, P, N.

For the supported branch, apply Definition 1006.55. The read reaches the boundary only after the route $G \longrightarrow S$ is displayed.

For the unsupported branch, apply Definition 1006.56. The same request row is sent to R rather than to hidden closure, hidden T supply, host equality, quotient data, or an off-packet continuation.

Finally use the consumer exactness package and the TasteGate status row. The package keeps downstream reads inside the same seven coordinates, and the TasteGate row requires the source/refusal split to be preserved by the named carrier surface. □

Theorem 1006.110 (GapClosureBoundary consumer route factorization)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, each downstream consumer read by* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *factors first through the decision surface of Definition 1006.109. Supported consumers then read* $G \longrightarrow S$ *with* H, C, P, N*, while unsupported consumers read* $G \longrightarrow R$ *with the same structural rows.*

Proof. Begin with the decision surface. It already separates every accepted read of B into the supported route $G \longrightarrow S$ or the unsupported route $G \longrightarrow R$.

Apply the consumer exactness package, Definition 1006.54. The listed hard-problem consumers receive the shared boundary packet and do not obtain a consumer-specific source, refusal, transport, continuation, provenance, or naming coordinate.

The supported case is therefore routed by Definition 1006.55. The unsupported case is routed by Definition 1006.56. Since both cases use the same structural rows H, C, P, N, the consumer route factors through the displayed decision surface before any downstream read is made. □

Theorem 1006.111 (GapClosureBoundary TasteGate field-faithful readback)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every faithful readback of the TasteGate row recorded in the closure status preserves the displayed packet fields* G, S, R, H, C, P, N*. In particular, it preserves the supported branch* $G \longrightarrow S$*, the unsupported branch* $G \longrightarrow R$*, and the consumer factorization of Definition 1006.110.*

Proof. Read the TasteGate row through the carrier of Definition 1006.1. The row is attached to the displayed $\mathsf{GapClosureBoundary}^{\uparrow}$ packet, not to an ambient closure oracle.

Use source-route totality and refusal-route exactness. A faithful readback must preserve which requests reach S and which requests reach R, because changing either endpoint changes the public source/refusal decision.

Finally apply consumer route factorization. The listed consumers see the decision only through G, S, R, H, C, P, N, so a field-faithful readback preserves all displayed packet fields and cannot replace them with hidden closure, private source access, host equality, quotient data, or an off-packet continuation. □

Theorem 1006.112 (GapClosureBoundary root source totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every supported root read of the boundary must pass through the displayed source route*

$$G \longrightarrow S$$

before any hard-problem consumer reads the packet. The rows H, C, P, N *keep that read inside the componentwise transport, displayed continuation, package provenance, and local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *surface, so no supported read can use a second source coordinate, host equality, quotient data, hidden closure supply, or an off-packet continuation.*

Proof. Start with the carrier of Definition 1006.1. The accepted root packet has exactly the rows G, S, R, H, C, P, N, with S the only displayed source row available to a supported boundary read.

Apply the NameCert obligations and displayed source precondition, Definition 1006.2 and Definition 1006.4. They require a supported read to expose the source observation or socket row S before the packet is consumed.

The structural rows do not change the route. They transport, replay, package, and locally name the already displayed source branch, so the supported root read is exactly the route $G \longrightarrow S$ inside G, S, H, C, P, N. □

Theorem 1006.113 (GapClosureBoundary root refusal handoff totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every off-source root read is handed to the displayed refusal row*

$$G \longrightarrow R$$

through H, C, P, N*. The handoff cannot export internal closure, hidden* T *supply, host equality, quotient data, private source selection, or any continuation outside the displayed packet.*

Proof. Begin with an off-source read of G. By Definition 1006.4, such a read has not exposed the source row S, so it cannot enter the supported branch.

Use refusal-ledger transport from Definition 1006.5. The request is routed to R along the same structural rows H, C, P, N that belong to the accepted carrier.

Finally apply the carrier and NameCert obligations, Definition 1006.1 and Definition 1006.2. Since the certificate surface contains only G, S, R, H, C, P, N, the refusal handoff has no row from which internal closure or hidden T supply could be exported. □

Theorem 1006.114 (GapClosureBoundary root consumer handoff)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, the downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receive exactly the displayed boundary rows* G, S, R, H, C, P, N*. A supported consumer receives the source handoff* $G \longrightarrow S$ *with* H, C, P, N*, an off-source consumer receives the refusal handoff* $G \longrightarrow R$ *with* H, C, P, N*, and no listed consumer receives a separate closure, private access, causal-necessity, aspect-reduction, quotient, or hidden* T *row.*

Proof. First expose the accepted packet through Definition 1006.1 and Definition 1006.2. These anchors identify the boundary rows available before any downstream hard-problem consumer reads the packet.

For a supported consumer, use Definition 1006.112. The read has already passed through $G \longrightarrow S$, and the structural rows only transport, replay, package, and name that source branch.

For an off-source consumer, use Definition 1006.113. The request has already been handed to R through H, C, P, N, without exporting hidden closure or hidden T supply.

It remains to check consumer exhaustion. Apply Definition 1006.9: the causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers are exactly the listed readers of this boundary packet. Therefore each receives one of the two displayed handoffs and no row outside G, S, R, H, C, P, N. □

The $\mathsf{GapClosureBoundary}^{\uparrow}$ sibling route first consumes the finite refutation witness surface of chapter 576 and the refusal boundary of chapter 975. Only after those two horizontal inputs are fixed does the chapter expose its source-refusal consumer ledger: supported reads remain on the displayed source branch, unsupported closure requests remain on the displayed refusal branch, and the later ledger receives no kernel-only shortcut around the sibling lattice.

Theorem 1006.115 (GapClosureBoundary finite trace evaluator route)**.** *Any* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consumer that evaluates a finite refusal trace or finite acceptance trace must pass through the* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *checkpoint and runtime route of chapter 818 before declaring source-gap closure. The boundary read therefore factors through the displayed request row, trace-evaluator checkpoint, supported or refused branch, componentwise transport, displayed replay, provenance, and local naming rows; it cannot declare closure from an unevaluated trace, hidden runtime verdict, or off-packet source-gap assertion.*

Proof. Start with the accepted boundary packet and the consumer bridge admission of Definition 1006.180. The consumer receives only the displayed request, branch, transport, replay, provenance, and local naming rows.

The finite trace is then evaluated through the $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ sibling surface. Its checkpoint fixes the runtime read before the supported branch or refused branch can be consumed as a source-gap boundary result.

Apply the source-route and refusal-route exactness results for the two possible branches. A supported trace remains on the source branch, while a refused trace remains on the refusal branch, and both branches keep the same structural rows.

Thus a consumer that skips the finite trace evaluator has not read the same $\mathsf{GapClosureBoundary}^{\uparrow}$ packet. It may report a separate runtime observation, but it cannot declare source-gap closure from this boundary certificate. □

The $\mathsf{GapClosureBoundary}^{\uparrow}$ sibling lattice first consumes the recognizer route of chapter 1048, the audit-packet route of chapter 820, and the finite-trace evaluator of chapter 818. Only after those three rows are visible does the boundary expose its source/refusal branch: recognized traces may pass to the displayed source row, while exactness failures remain on the refusal row with the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

Theorem 1006.116 (GapClosureBoundary refusal verdict totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the boundary exposes the source GAP row* G*, the unsupported request endpoint, the refusal verdict row* R*, and the refusal ledger carried by* H, C, P, N *before any consumer can read a supported route. A supported consumer may later read* $G \longrightarrow S$*, but only after the refusal verdict surface has fixed how off-source requests are routed.*

Proof. Begin with the source/refusal decision surface of Definition 1006.109. It separates accepted reads of G into the displayed source route $G \longrightarrow S$ and the displayed refusal route $G \longrightarrow R$.

For the refusal side, apply the source-refusal consumer ledger, Definition 1006.57. The unsupported request endpoint is accounted for by the refusal route through R and by the same structural rows H, C, P, N.

Now use the root totality handoff surface, Definition 1006.114. It requires downstream consumers to receive the displayed rows G, S, R, H, C, P, N rather than a private boundary packet.

Finally, the recognizer-facing dependency surface is already concrete: Definition 1048.3 and Definition 993.3 place accepted evidence through a certificate gate and tuple readback before it reaches the boundary. Thus the refusal verdict and ledger are visible before a consumer can use the supported route. □

Theorem 1006.117 (GapClosureBoundary supported route exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every supported route is exhausted by the displayed* BHist *rows* G, S, R*, the componentwise* hsame *row* H*, the displayed* Cont *row* C*, the* Pkg *row* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. No supported consumer route can use hidden closure supply, host equality, quotient data, or a boundary route outside these rows.*

Proof. First expose the scoped dependency inventory. By Definition 1006.79, the boundary displays the BHist carrier rows together with H, C, P, N before any consumer reads it.

For the supported branch, apply root source totality from Definition 1006.112. A supported root read must pass through $G \longrightarrow S$ and then remain inside the componentwise transport, displayed continuation, provenance, and local naming surface.

Next apply consumer route totality, Definition 1006.68. It assigns each listed downstream read to the displayed source/refusal split and rules out a route bypassing both S and R.

The concrete sibling landing theorems close the only remaining import path. The certificate-gate exactness of Definition 1048.3 and the field-faithful readback of Definition 993.3 make the recognizer and tuple surfaces feed the same displayed rows, so there is no hidden closure route beyond G, S, R, H, C, P, N. □

Theorem 1006.118 (GapClosureBoundary NameCert consumer non-escape)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any downstream consumer receives only the refusal-plus-supported-route packet named by* N *and transported through* H, C, P*. The consumer receives no unregistered global closure, hidden exactness theorem, host quotient-like boundary, private source selector, or off-packet continuation.*

Proof. Start from scoped consumer non-escape, Definition 1006.81. It confines the listed sibling consumers to the scoped source/refusal surface of G, S, R, H, C, P, N.

The supported side is governed by Definition 1006.117. The source branch is the route $G \longrightarrow S$ carried by the displayed transport, continuation, package provenance, and local $\mathsf{NameCert}$ rows.

The refusal side is governed by Definition 1006.116. Unsupported requests are already placed on the refusal verdict row R and its ledger before the supported route can be consumed.

Finally use root consumer handoff and recognizer refusal stability, Definition 1006.114 and Definition 1006.74. The first keeps every listed downstream consumer inside the displayed packet, and the second prevents failed recognizer landing from being reread as source evidence. Hence the local NameCert surface exports only the refusal-plus-supported-route packet and no unregistered global closure, hidden exactness theorem, quotient-like boundary, private selector, or off-packet continuation. □

Theorem 1006.119 (GapClosureBoundary source-refusal admission triad)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, a downstream recognizer-facing request is admitted only through the displayed source-refusal triad: the request row* G*, the supported source route* $G \longrightarrow S$*, and the unsupported refusal route* $G \longrightarrow R$*, all transported, replayed, packaged, and named by* H, C, P, N*. The triad is read through the source-refusal decision surface, the consumer exactness package, the recognizer landing obligations, and the route completion surface. It exports no hidden closure row, hidden* T *supply, private source route, host equality row, quotient datum, direct aspect-reduction row, or off-packet continuation.*

Proof. Start with Definition 1006.109. It separates each accepted read of B into the supported route $G \longrightarrow S$ or the unsupported route $G \longrightarrow R$.

Next apply the consumer exactness package of Definition 1006.54. The listed hard-problem consumers read the shared carrier and receive no consumer-specific source, refusal, transport, continuation, provenance, or naming coordinate.

Use the recognizer landing obligations, Definition 1006.73 and Definition 1006.74. Successful recognizer landings are consumed as the displayed source branch, and failed or unsupported landings are consumed as the displayed refusal branch.

Finally apply Definition 1006.99. The completed route remains G, S, R, H, C, P, N, so the excluded hidden, host, quotient, aspect, and off-packet rows have no admitted coordinate. □

Theorem 1006.120 (GapClosureBoundary refusal ledger exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the refusal ledger for downstream recognizer-facing requests is exhausted by the displayed refusal route* $G \longrightarrow R$ *with structural rows* H, C, P, N*. Any request that cannot enter the source branch through the recognizer landing surface remains at this refusal boundary, and the ledger supplies no hidden closure theorem, hidden* T *source, host equality, quotient object, private selector, or sibling-specific recognizer row.*

Proof. First read the unsupported side of the source-refusal decision surface. Without the displayed source route $G \longrightarrow S$, the accepted branch is the refusal route $G \longrightarrow R$.

Apply Definition 1006.56 and Definition 1006.69. These theorems identify R as the refusal endpoint and preserve it under the structural rows H, C, P, N.

Then use recognizer refusal stability from Definition 1006.74. A failed landing after the $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ and $\mathsf{FiniteReflectionTuple}^{\uparrow}$ surfaces cannot be reread as source evidence.

The consumer non-escape boundary of Definition 1006.99 removes hidden closure, hidden T supply, host equality, quotient data, private selectors, and sibling-specific recognizer rows. Hence the refusal ledger is exhausted by the displayed refusal route. □

Theorem 1006.121 (GapClosureBoundary downstream consumer boundary)**.** *Every downstream recognizer-facing consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet either factors through the source-refusal admission triad of Definition 1006.119 or remains at the refusal ledger boundary of Definition 1006.120. In particular, the consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receive only the displayed packet* G, S, R, H, C, P, N *and the two visible source-refusal routes. They receive no bypass around the displayed refusal package and no hidden recognizer, solver, host equality, quotient, closure, or off-packet continuation.*

Proof. Begin with recognizer consumer exhaustion, Definition 1006.75. It says successful recognizer landings are consumed as $G \longrightarrow S$ and failed or unsupported landings are consumed as $G \longrightarrow R$.

Apply the source-refusal admission triad to the successful side. The admitted route reads the shared request, source, refusal, transport, continuation, provenance, and local naming rows before any downstream consumer receives the packet.

Apply refusal ledger exhaustion to the unsupported side. Requests not passing through the displayed source branch remain at R with the same structural rows and cannot use a bypassing recognizer or solver route.

Finally combine supported-route totality with consumer non-escape, Definition 1006.97 and Definition 1006.99. The completed downstream boundary is exactly G, S, R, H, C, P, N, and every hidden recognizer, solver, host equality, quotient, closure, or off-packet continuation request is refused. □

Theorem 1006.122 (GapClosureBoundary terminal source admission)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, a terminal source read is admitted only through the displayed source branch*

$$G \longrightarrow S$$

with the componentwise hsame *row* H*, displayed* Cont *row* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}}$ *row* N*. Any request not supported by that branch is routed through the displayed refusal branch*

$$G \longrightarrow R$$

with the same structural rows. The terminal read exports no hidden closure row, hidden T *supply, private source selector, host equality row, quotient datum, or off-packet continuation.*

Proof. Begin with the source-refusal admission triad of Definition 1006.119. It already identifies the terminal recognizer-facing request with the request row G, the supported source route $G \longrightarrow S$, and the unsupported refusal route $G \longrightarrow R$.

For supported reads, apply recognizer source admission, Definition 1006.73. The $\mathsf{GroundCompilerRecognizerFlow}$ gate and $\mathsf{FiniteReflectionTuple}^{\uparrow}$ readback place accepted source evidence on the displayed branch before the boundary packet is consumed.

For unsupported reads, apply refusal ledger exhaustion, Definition 1006.120. Requests that fail the recognizer landing remain at R under H, C, P, N and cannot be reread as source evidence.

Finally use the downstream consumer boundary of Definition 1006.121. It restricts every listed consumer to the same packet G, S, R, H, C, P, N, so the terminal read has no route outside the displayed source/refusal split. □

Theorem 1006.123 (GapClosureBoundary refusal classifier stability)**.** *Let* $B = (G, S, R, H, C, P, N)$ *and* $B' = (G', S', R', H', C', P', N')$ *be accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carriers. Componentwise* hsame *transport through* H *and* H'*, followed by replay through the displayed* Cont *rows* C *and* C'*, preserves the supported/refusal classifier split: supported reads remain reads of the displayed source branch* $G \longrightarrow S$*, unsupported requests remain reads of the displayed refusal branch* $G \longrightarrow R$*, and transport cannot introduce host equality, quotient data, private source selectors, hidden closure supply, or an off-packet continuation row.*

Proof. First expose the classifier inventory through Definition 1006.119. The classifier-relevant data are the request row, the source branch, the refusal branch, and the structural rows H, C, P, N.

Next apply recognizer refusal stability, Definition 1006.74. Failed $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ and $\mathsf{FiniteReflectionTuple}^{\uparrow}$ landings cannot become source evidence after replay, so the refusal side remains refusal data.

For the supported side, use Definition 1006.73 together with the certificate-gate and tuple-readback anchors Definition 1048.3 and Definition 993.3. The accepted recognizer row is read back as displayed boundary data before it reaches S.

Finally invoke refusal ledger exhaustion, Definition 1006.120. Since the ledger contains no host equality, quotient object, private selector, hidden closure row, or off-packet continuation coordinate, componentwise transport and replay preserve the same supported/refusal split. □

Theorem 1006.124 (GapClosureBoundary consumer ledger nonescape)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the downstream consumers* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receive only the displayed packet rows* G, S, R, H, C, P, N*. A supported consumer reads through* $G \longrightarrow S$*, an unsupported consumer reads through* $G \longrightarrow R$*, and no listed consumer receives a hidden recognizer, solver, host, quotient, closure, private source selector, or off-packet continuation row.*

Proof. Start with Definition 1006.121. It says every listed downstream consumer either factors through the source-refusal admission triad or remains at the refusal ledger boundary.

For the source branch, apply terminal source admission, Definition 1006.122. Supported reads are terminal only as reads of $G \longrightarrow S$ with H, C, P, N.

For the refusal branch, apply refusal ledger exhaustion, Definition 1006.120. Unsupported closure requests are kept at R under the same structural rows and do not acquire a consumer-specific escape route.

Finally combine the recognizer consumer exhaustion of Definition 1006.75 with the root consumer handoff of Definition 1006.114. The five listed consumers receive exactly the displayed packet surface, so hidden recognizer, solver, host, quotient, closure, private-selector, and off-packet continuation rows are absent from the ledger. □

Theorem 1006.125 (GapClosureBoundary NameCert route completion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier*

$$B = (G, S, R, H, C, P, N),$$

the obligations of Definition 1006.2 complete the route surface visible to causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers. A supported consumer reads the request row G *through the displayed source row* S*; an unsupported consumer reads* G *through the refusal row* R*; and both routes are transported by* H*, replayed by* C*, packaged by* P*, and locally named by* N*.*

Proof. Project the carrier of Definition 1006.1. Its full consumer-visible inventory is G, S, R, H, C, P, N, with H assigned to componentwise hsame transport, C to displayed Cont replay, P to Pkg provenance, and N to the local $\mathsf{NameCert}$ row.

Apply Definition 1006.2. Carrier exposure gives the finite packet, source-route exactness sends supported reads through S, and refusal-ledger routing sends unsupported reads through R.

Finally use Definition 1006.3 together with the sibling consumer surface recorded in the carrier chapter. The causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers receive only the completed source-or-refusal route, so no consumer reads a hidden closure coordinate or a parameter-transport equivalence schema. □

Theorem 1006.126 (GapClosureBoundary refusal totality readback)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every hard-problem consumer readback that cannot exhibit the source branch* $G \longrightarrow S$ *is total on the refusal branch* $G \longrightarrow R$*. The readback keeps the refusal row under the displayed* hsame *transports,* Cont *replay,* Pkg *provenance, and* $\mathsf{NameCert}$ *naming*

rows, and it exports no private other-minds access, causal necessity row, direct aspect-reduction row, hidden T *supply, or off-packet continuation.*

Proof. Begin with the source/refusal split supplied by Definition 1006.125. A consumer readback is either supported by the displayed source row S or is unsupported at the boundary.

In the unsupported case, apply Definition 1006.3. The only available endpoint is the refusal ledger R, and the structural rows H, C, P, N preserve that endpoint rather than replacing it with a source witness.

The listed sibling consumers do not add further coordinates. Causal commitment, finite witnessed refutation, other-minds commitment, aspect-chain, and solipsism-refusal consumers therefore read the same refusal branch when S is absent, with no host-side completion route beyond B. □

Theorem 1006.127 (GapClosureBoundary ledger exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the route ledger is exhausted by the source branch* $G \longrightarrow S$*, the refusal branch* $G \longrightarrow R$*, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*. No downstream consumer can extend the ledger to an internal universal closure row or to a separate parameter-transport equivalence surface.*

Proof. First read the route-completion theorem. It identifies the only two consumer routes from G: the displayed source route and the displayed refusal route.

Next read refusal totality. Whenever the source route is absent, the readback is already total at R under H, C, P, N.

It remains to check consumer exhaustion. The carrier of Definition 1006.1 and the obligation theorem of Definition 1006.2 leave no row beyond G, S, R, H, C, P, N. Thus every downstream hard-problem consumer receives a finite ledger route and no extra closure, hidden source, private access, causal, aspect, or off-packet continuation coordinate. □

Theorem 1006.128 (GapClosureBoundary terminal route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier*

$$B = (G, S, R, H, C, P, N),$$

every terminal read of the root request row is routed through exactly one displayed branch. A supported terminal read uses the source branch

$$G \longrightarrow S$$

with the componentwise hsame *row* H*, the displayed* Cont *row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. An unsupported terminal read uses the refusal branch*

$$G \longrightarrow R$$

with the same structural rows. No terminal read exports a row outside the finite BHist *packet* G, S, R, H, C, P, N*.*

Proof. Begin with the carrier and local certificate obligations of Definition 1006.1 and Definition 1006.2. They expose the finite BHist packet and assign H, C, P, N to componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming.

For the supported branch, apply terminal source admission from Definition 1006.122 together with route completion, Definition 1006.125. These rows force the accepted terminal source read to pass through $G \longrightarrow S$ before any consumer reads the structural surface.

For the unsupported branch, apply refusal route exactness from Definition 1006.56 and refusal totality readback, Definition 1006.126. The same request row is sent to R and remains under H, C, P, N.

Finally use ledger exhaustion, Definition 1006.127. The ledger has no third terminal endpoint beyond the source and refusal branches, so every terminal read is accounted for by one of the two displayed routes. □

Theorem 1006.129 (GapClosureBoundary refusal NameCert factorization)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, every unsupported request factors through the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row. The factorization first records the refusal endpoint* R*, then transports it by the componentwise* hsame *row* H*, replays it through the displayed* Cont *row* C*, packages it by the* Pkg *row* P*, and names the resulting refusal read by* N*. Thus unsupported requests remain inside the finite* BHist *carrier and cannot be read as an unnamed closure resource.*

Proof. Read the refusal route from Definition 1006.56. An unsupported request cannot use the source branch, so its endpoint is the displayed refusal row R.

Next apply the NameCert route completion theorem, Definition 1006.125. It identifies the refusal route as one of the local certificate duties and records the structural rows H, C, P, N as the only transport, replay, provenance, and naming data attached to that duty.

Now use the refusal classifier stability and refusal totality readback rows of Definition 1006.123 and Definition 1006.126. They preserve the refusal endpoint through the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface rather than replacing it by a source witness.

The requested factorization is therefore the displayed path through R, H, C, P, N ending at N. Since the carrier exposes no additional row beyond G, S, R, H, C, P, N, the unsupported request stays within the local $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ packet. □

Theorem 1006.130 (GapClosureBoundary consumer surface exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the hard-problem consumers* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*,* $\mathsf{AspectChain}^{\uparrow}$*, and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *receive only the rows* G, S, R, H, C, P, N*. Supported consumers read the terminal source branch* $G \longrightarrow S$*; unsupported consumers read the refusal branch* $G \longrightarrow R$*; and all such reads are carried by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Start with consumer ledger exhaustion, Definition 1006.57. It places each listed hard-problem consumer on the shared source/refusal ledger rather than on a consumer-specific surface.

For supported consumers, use terminal route totality from Definition 1006.128. The only available supported endpoint is $G \longrightarrow S$ with the structural rows H, C, P, N.

For unsupported consumers, use refusal NameCert factorization from Definition 1006.129. The unsupported request is routed through R, H, C, P, N and named by the same local certificate row N.

Finally apply route ledger exhaustion from Definition 1006.127. The ledger is exhausted by G, S, R, H, C, P, N, so the listed consumers receive no additional hard-problem coordinate outside the displayed packet. □

Theorem 1006.131 (GapClosureBoundary source-refusal totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *request row in the finite* BHist *carrier*

$$B = (G, S, R, H, C, P, N),$$

the readback factors through exactly one displayed branch before any downstream consumer receives it. A supported read factors through the source branch $G \longrightarrow S$; an unsupported read factors through the refusal branch $G \longrightarrow R$; and both branches are transported only through the componentwise hsame *row H, the displayed* Cont *row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row N. The packet exports no hidden closure coordinate and no hidden* T*-supply coordinate.*

Proof. Start with terminal source admission, Definition 1006.122. It identifies the supported terminal request with the branch $G \longrightarrow S$ and keeps the structural rows H, C, P, N attached to that branch.

For the unsupported side, apply refusal totality readback, Definition 1006.126. It sends every source-absent read to the refusal branch $G \longrightarrow R$ under the same transport, replay, provenance, and naming rows.

Finally use route ledger exhaustion from Definition 1006.127. Since the carrier exposes no endpoint beyond S and R and no structural rows beyond H, C, P, N, the accepted request has no hidden closure or T-supply coordinate. □

Theorem 1006.132 (GapClosureBoundary NameCert scope readiness)**.** *The carrier admission, source/refusal split, local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *obligations, route completion, refusal totality, and ledger exhaustion form the scoped obligation surface of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Concretely, every accepted packet is read as*

$$G \to (S \mid R) \to H \to C \to P \to N,$$

with the route branch selected before transport, replay, package provenance, or local naming is consumed.

Proof. Carrier admission is the finite packet of Definition 1006.1. The local certificate obligations of Definition 1006.2 assign the visible rows to request exposure, source-route exactness, refusal-ledger routing, structural transport, continuation replay, provenance, and naming.

Route completion, Definition 1006.125, turns those rows into the displayed source-or-refusal route. Refusal totality, Definition 1006.126, handles every source-absent read without adding another endpoint.

Ledger exhaustion, Definition 1006.127, then closes the obligation surface. The only remaining rows are BHist carrier data, hsame transport, Cont replay, Pkg provenance, and the local NameCert row. □

Theorem 1006.133 (GapClosureBoundary bridge readback scope)**.** *Any later bridge-facing readback of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *must first display source/refusal totality, componentwise transport through H, local* NameCert *provenance through P, N, and the TasteGate row for the same packet. Such a readback may expose the scoped source/refusal surface, but it does not claim* Gap *discharge, does not provide a standard bridge, and does not turn the refusal branch into a source witness.*

Proof. First apply source-refusal totality from Definition 1006.131. A bridge-facing read must therefore show whether the request is on $G \longrightarrow S$ or on $G \longrightarrow R$ before it is read by any consumer.

Next use NameCert scope readiness from Definition 1006.132. It requires the componentwise transport, continuation replay, package provenance, and local naming rows to remain visible as H, C, P, N.

The TasteGate row is the same scoped packet boundary already recorded by the chapter status. Since terminal source admission and ledger exhaustion leave no row that discharges Gap or supplies a standard bridge, the readback is only a bridge precondition surface. □

Theorem 1006.134 (GapClosureBoundary supported source-route readiness)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every supported root read reaches a downstream consumer only through the displayed source route*

$$G \longrightarrow S.$$

The read keeps the root row G*, the source row* S*, the componentwise* hsame *row* H*, the displayed* Cont *row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N *visible. The refusal row* R *remains part of the same packet as the row that records how the same boundary routes off-source requests.*

Proof. Start with the source/refusal decision surface, Definition 1006.109. A supported read is selected by the displayed branch $G \longrightarrow S$, not by a private source selector or hidden closure coordinate.

Next use root consumer handoff, Definition 1006.114. The consumer receives the displayed packet G, S, R, H, C, P, N, so the supported branch keeps H, C, P, N attached as transport, continuation, provenance, and local naming rows.

Finally use consumer ledger exhaustion, Definition 1006.57. Since the same packet also accounts for unsupported reads by the refusal row R, the supported route remains source-readable while the refusal ledger stays visible rather than being erased by the supported branch. □

Theorem 1006.135 (GapClosureBoundary refusal-ledger routing exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$ *of Definition 1006.1, every unsupported endpoint is exhausted by the refusal route*

$$G \longrightarrow R$$

together with the structural rows H, C, P, N*. Such an endpoint can be transported, replayed, packaged, and named as refusal data, but it cannot be reread as source evidence and cannot escape to hidden closure supply, host equality, quotient data, or an off-packet continuation.*

Proof. Begin with the source/refusal decision surface of Definition 1006.109. It gives an unsupported read the displayed endpoint R rather than the source endpoint S.

Use consumer ledger exhaustion, Definition 1006.57. The listed downstream consumers receive the refusal endpoint through the same rows H, C, P, N, so there is no consumer-specific endpoint outside the packet.

It remains to rule out rereading the refusal endpoint as source evidence. Recognizer refusal stability, Definition 1006.74, keeps failed or unsupported recognizer landings on the refusal side. Root consumer handoff, Definition 1006.114, then keeps that refusal side inside G, S, R, H, C, P, N for every downstream read. □

Theorem 1006.136 (GapClosureBoundary TasteGate scoped nontriviality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, the scoped TasteGate surface is nontrivial exactly because the packet displays both the supported route* $G \longrightarrow S$ *and the refusal route* $G \longrightarrow R$ *with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row. The TasteGate target recorded as* `BEDC.Derived.GapClosureBoundaryUp.taste_` *therefore reads a field-faithful packet surface rather than an undifferentiated parameter echo.*

Proof. First combine supported source-route readiness with refusal-ledger routing exhaustion, Definition 1006.134 and Definition 1006.135. The two branches are both visible in the same accepted packet, so the TasteGate row has a concrete distinction to preserve.

Next use the root consumer handoff and consumer ledger exhaustion theorems, Definition 1006.114 and Definition 1006.57. They keep all downstream reads inside G, S, R, H, C, P, N and make H, C, P, N the complete transport, replay, package, and local naming surface for the two branches.

Finally use recognizer refusal stability, Definition 1006.74. Unsupported recognizer landings remain refusal data, while supported landings remain source-route data. Thus the TasteGate row recorded in the chapter status is field-faithful to the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ packet and does not collapse the source and refusal branches. □

Theorem 1006.137 (GapClosureBoundary acceptance row exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the accepted root row is exhausted by the displayed* BHist *inventory* G, S, R*, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. A source-supported consumer route must read the accepted source row as*

$$G \longrightarrow S,$$

and an unsupported closure request must read the refusal route as

$$G \longrightarrow R.$$

No root acceptance row exports a third consumer route, hidden closure coordinate, hidden T *supply, host equality row, quotient datum, private selector, or off-packet continuation.*

Proof. First read the source/refusal consumer ledger. Definition 1006.55 requires every supported hard-problem read to expose $G \longrightarrow S$ before consuming H, C, P, N, while Definition 1006.56 sends every unsupported closure request through $G \longrightarrow R$ with the same structural rows.

Next use the root refusal-route obligations. Definition 1006.116 makes the refusal endpoint and ledger visible before supported consumption, and Definition 1006.117 confines supported routes to the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

The terminal source unblock removes the remaining root entry. Definition 1006.122 admits terminal source reads only through $G \longrightarrow S$ and sends non-supported reads through $G \longrightarrow R$, and Definition 1006.124 keeps every listed consumer inside G, S, R, H, C, P, N.

Finally apply refusal route totality, Definition 1006.147. Since every unsupported request is exhausted by the displayed refusal branch and the structural rows cannot generate another endpoint, the accepted root row has no third consumer route or hidden closure supply. □

Theorem 1006.138 (GapClosureBoundary refusal route readback)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, refusal-route readback is determined by the displayed* BHist *request row* G*, refusal row* R*, componentwise* hsame *row* H*, displayed* Cont *route row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. If a consumer route lacks the accepted source row* $G \longrightarrow S$*, then its readback is exactly the refusal route* $G \longrightarrow R$ *over* H, C, P, N*, and the readback cannot be reclassified as a source route, a private root, a closure witness, host equality, quotient data, or an off-packet continuation.*

Proof. Begin with the source-refusal ledger. Definition 1006.56 identifies the unsupported request endpoint as R and keeps the transport, continuation, provenance, and naming rows attached to that endpoint.

Now apply the root refusal-route obligations. Definition 1006.116 fixes the verdict row before supported consumption, while Definition 1006.118 says downstream consumers receive only the refusal-plus-supported-route packet named by N and transported through H, C, P.

For terminal readback, use Definition 1006.123. Componentwise hsame transport through H, followed by replay through C, preserves unsupported requests as refusal data rather than source evidence.

The totality lock closes the route. By Definition 1006.147, every unsupported closure request reaches R through H, C, P, N and has no hidden closure, host-equality, private-access, or off-packet continuation branch. Hence refusal readback is exactly the displayed route $G \longrightarrow R$. □

Theorem 1006.139 (GapClosureBoundary NameCert route closure)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *route surface is closed by the accepted source row* $G \longrightarrow S$*, the refusal route* $G \longrightarrow R$*, and the displayed structural rows* H, C, P, N*. The certificate records the* BHist *carrier rows componentwise, transports them by* hsame*, replays them through* Cont*, packages them by* Pkg*, and names them locally by* N*; it exports no unregistered consumer route, global closure row, hidden* T *supply, host equality row, quotient datum, private source selector, or off-packet continuation.*

Proof. First assemble the consumer ledger. By Definition 1006.57, supported reads are accounted for by $G \longrightarrow S$ with H, C, P, N, unsupported closure requests are accounted for by $G \longrightarrow R$ with the same rows, and no third consumer route is exported.

Next apply the root refusal-route obligation surface. Definition 1006.117 keeps the supported branch inside the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows, while Definition 1006.118 prevents a downstream consumer from escaping that named packet.

The terminal source unblock supplies the root-facing split. Definition 1006.122 admits source reads only through $G \longrightarrow S$ and routes non-supported requests through $G \longrightarrow R$; Definition 1006.124 keeps the listed hard-problem consumers inside the same packet.

Finally use the refusal route totality lock, Definition 1006.147. It exhausts the refusal side by the displayed branch and structural rows, so the local $\mathsf{NameCert}$ route surface is closed by the two named routes and exports no unregistered consumer route or hidden closure coordinate. □

Theorem 1006.140 (GapClosureBoundary L10 route consumer interface)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, every L10-facing consumer of the boundary first factors through one of the displayed branches*

$$G \longrightarrow S$$

or

$$G \longrightarrow R$$

before it reads the componentwise hsame *row* H*, the displayed* Cont *route row* C*, the* Pkg *provenance row* P*, or the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. The interface therefore exports only the source branch or the refusal branch over* H, C, P, N*, and it exports no hidden closure, hidden* T *supply, host equality, quotient data, selected Real limit, ambient completeness, or off-packet continuation.*

Proof. Begin with the source packet, Definition 1006.82. It exposes the request row G, the source row S, the refusal ledger R, and the structural rows H, C, P, N as the complete consumer-facing inventory.

For the source side, apply the scoped source/refusal rows Definition 1006.131 and Definition 1006.122. They require a supported consumer to display $G \longrightarrow S$ before any terminal or L10-facing read consumes the structural surface.

For the refusal side, apply Definition 1006.56 and Definition 1006.123. They route a source-absent request through $G \longrightarrow R$ and keep that endpoint stable under componentwise transport and replay.

Finally use classifier stability and the NameCert obligation surface, Definition 1006.83 and Definition 1006.84. Since the certificate has no row beyond G, S, R, H, C, P, N, an L10-facing consumer has no hidden completion, equality, quotient, selected-limit, ambient-completeness, or continuation exit. □

Theorem 1006.141 (GapClosureBoundary Real-completion refusal)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, any request for* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, or hidden completion supply is refused unless the request first enters the displayed source route*

$$G \longrightarrow S$$

and then remains transported, replayed, packaged, and named only through H, C, P, N*. A request that does not enter by that source route is routed through*

$$G \longrightarrow R$$

with the same structural rows, and it receives no hidden closure, hidden T *supply, host equality, quotient data, selected Real limit, ambient completeness, or off-packet continuation.*

Proof. Project the source packet of Definition 1006.82. The packet gives only the request, source, refusal, transport, continuation, package, and local naming rows.

If a Real-completion request is admitted, the L10 route consumer interface above and terminal source admission force it through $G \longrightarrow S$ before any $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, or $\mathsf{StreamName}^{\uparrow}$ consumer can read the packet.

If that source route is absent, use the scoped refusal rows Definition 1006.131 and Definition 1006.56. The request is sent to $G \longrightarrow R$ under H, C, P, N rather than to a concealed completion object.

The exclusions are preserved by classifier stability and the NameCert obligation surface. Definition 1006.83 prevents transport and replay from turning a refusal endpoint into source evidence, while Definition 1006.84 records no coordinate for hidden closure, hidden T supply, host equality, quotient data, selected Real limit, ambient completeness, or off-packet continuation. □

Theorem 1006.142 (GapClosureBoundary terminal consumer package)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the L10 terminal consumer surface is the package consisting of the source packet, the classifier-stability row, the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *obligation surface, the source route* $G \longrightarrow S$*, and the refusal route* $G \longrightarrow R$ *over* H, C, P, N*. This package lets downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *consumers cite a finite source/refusal interface: supported reads enter by* $G \longrightarrow S$*, unsupported completion-facing requests enter* $G \longrightarrow R$*, and no consumer receives hidden closure, hidden* T *supply, host equality, quotient data, selected Real limit, ambient completeness, or off-packet continuation.*

Proof. Assemble the source packet from Definition 1006.82. It supplies the finite rows G, S, R, H, C, P, N and no external consumer coordinate.

Next add the classifier row. By Definition 1006.83, componentwise hsame transport and displayed Cont replay preserve the choice between source and refusal rather than creating a third terminal route.

Now add the local NameCert obligations from Definition 1006.84. They identify carrier exposure, source-route admission, refusal-ledger exactness, consumer-boundary exhaustion, and non-escape of host or global refusal data as the finite certificate surface.

Finally use the two L10 rows above together with the scoped source/refusal rows Definition 1006.131 and Definition 1006.56. The source row handles supported terminal consumers, the refusal row handles unsupported completion-facing requests, and the assembled package has no coordinate from which a hidden completion or ambient Real limit can be read. □

Theorem 1006.143 (GapClosureBoundary scoped source-refusal record)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier*

$$B = (G, S, R, H, C, P, N),$$

the scoped source-refusal record is exactly the finite BHist *packet with the source route* $G \longrightarrow S$*, the refusal route* $G \longrightarrow R$*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N*. The record contains no third route, hidden source selector, internal closure coordinate, or off-packet consumer row.*

Proof. Start with the carrier and local NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose G, S, R, H, C, P, N as the finite carrier inventory and assign H, C, P, N to transport, replay, provenance, and naming.

For the source side, apply source-route totality, Definition 1006.55. Every supported read reaches the displayed source row through $G \longrightarrow S$ before the structural rows are consumed.

For the refusal side, apply refusal-route exactness, Definition 1006.56. Every unsupported read reaches R through $G \longrightarrow R$ with the same structural rows.

Finally use consumer ledger exhaustion, Definition 1006.57. Since the two displayed branches exhaust the listed consumers, the scoped record has no third route or hidden closure coordinate. □

Theorem 1006.144 (GapClosureBoundary consumer scope nonescape)**.** *Every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consumer route factors through the scoped source-refusal record of Definition 1006.143. Supported consumers read* $G \longrightarrow S$ *with* H, C, P, N*, unsupported consumers read* $G \longrightarrow R$ *with* H, C, P, N*, and no consumer can read host equality, quotient data, private other-minds access, causal necessity, direct aspect reduction, hidden* T *supply, or an off-packet continuation.*

Proof. First invoke the scoped source-refusal record. It fixes the only accepted carrier rows and the two displayed branches before a downstream hard-problem consumer receives the packet.

Next apply the consumer exactness package, Definition 1006.54. It keeps the listed sibling consumers on the same shared boundary carrier rather than on consumer-specific source, refusal, transport, continuation, provenance, or naming coordinates.

Finally use refusal non-escape and structural row conservation, Definition 1006.3 and Definition 1006.51. The refusal branch cannot become a closure supply, and the structural rows preserve the displayed branch instead of creating host equality, quotient data, private access, causal necessity, aspect reduction, hidden T data, or an off-packet continuation. □

Theorem 1006.145 (GapClosureBoundary TasteGate scope readback)**.** *The TasteGate readback for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *reads exactly the scoped source-refusal record over* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. It preserves the carrier* $B = (G, S, R, H, C, P, N)$*, the source route* $G \longrightarrow S$*, the refusal route* $G \longrightarrow R$*, consumer factorization, and the rows* H, C, P, N *as the complete transport, replay, provenance, and local naming surface for the accepted boundary.*

Proof. Begin with the scoped source-refusal record and consumer scope nonescape. These results bind the carrier, source branch, refusal branch, and downstream consumer routes to G, S, R, H, C, P, N.

Use the TasteGate obligation triad, Definition 1006.157. It reads the same carrier exposure, branch distinction, and structural rows as the visible packet whose nontriviality is recorded by the TasteGate target.

Close with the NameCert scope readiness theorem, Definition 1006.132. Its readback surface is precisely BHist carrier data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row, so the TasteGate readback cannot add an external source, refusal, or consumer coordinate. □

Theorem 1006.146 (GapClosureBoundary public certificate handoff)**.** *The* GapClosureBoundary$^{\uparrow}$ *public certificate surface is the finite route obtained by packaging the scoped source-refusal record, the displayed source route, the displayed refusal route, and the root* NameCert *rows:*

$$G, S, R, H, C, P, N.$$

Every public consumer receives the supported branch through $G \longrightarrow S$ or the unsupported branch through $G \longrightarrow R$, with the same transport, replay, provenance, and local naming rows. It receives no hidden closure witness, hidden T *supply, private other-minds access, causal necessity primitive, quotient boundary equality, or off-packet continuation.*

Proof. Start with the scoped record of Definition 1006.143. It already binds the supported and unsupported reads to one displayed packet rather than to a private closure source.

For the supported public route, apply Definition 1006.55. The branch reaches the source row only through $G \longrightarrow S$ before downstream consumers inspect H, C, P, N.

For the unsupported public route, apply Definition 1006.56. The branch reaches the refusal row only through $G \longrightarrow R$ with the same structural rows.

Finally apply Definition 1006.149. The root NameCert row packages the two branches with transport, replay, and provenance, so the public certificate has exactly the displayed finite surface and no coordinate for the excluded host objects. □

Closure status for NameCert$_{\text{GapClosureBoundary}^{\uparrow}}$**.**

Constructive story: A GapClosureBoundary$^{\uparrow}$ packet is generated from a GAP request row, displayed source observation or socket row, refusal and non-escape ledger, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{GapClosureBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1006.143, Definition 1006.55, and Definition 1006.56 bind the carrier, source/refusal replay route, consumer factorization, and TasteGate readback to the displayed BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{GapClosureBoundary}^{\uparrow}}$, theoremChecked)

Lean4 ✓ `BEDC.Derived.GapClosureBoundaryUp.taste gate`

Lean target: `BEDC.Derived.GapClosureBoundaryUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not internalise the GAP request, construct a T supply witness, prove causal necessity, give private other-minds access, reduce aspect chains, or provide a public bridge.

Upgrade path: Mature closure requires standard-bridge checking and downstream consumer replay of the source/refusal record.

Theorem 1006.147 (GapClosureBoundary refusal route totality)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *packet $B = (G, S, R, H, C, P, N)$, every unsupported closure request reaches the refusal row R through the same structural rows H, C, P, N. The route is total over the displayed branch*

$$G \longrightarrow R$$

and cannot branch to hidden closure, hidden T *supply, host equality, private access, or an off-packet continuation.*

Proof. Start with refusal route exactness, Definition 1006.56. It sends each unsupported closure request to R over the displayed structural rows H, C, P, N and removes hidden closure and hidden T supply as exits.

Use refusal branch determinacy and refusal non-escape, Definition 1006.104 and Definition 1006.3. These keep the request at the displayed endpoint $G \longrightarrow R$ rather than at host equality, private access, or a concealed source branch.

Finally apply structural row conservation, Definition 1006.51. Since H, C, P, N only preserve, replay, package, and name the displayed refusal branch, every unsupported closure request is exhausted by the public refusal route. □

Theorem 1006.148 (GapClosureBoundary route completion totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the source/refusal route completion surface is total for the listed hard-problem consumers. A supported consumer route is exactly the displayed source route*

$$G \longrightarrow S,$$

and an unsupported closure request is exactly the displayed refusal route

$$G \longrightarrow R,$$

with H, C, P, N *supplying the only componentwise transport, replay, provenance, and local naming rows. The completed route exports no hidden closure coordinate, hidden* T *supply, private source witness, host equality row, quotient datum, or off-packet continuation.*

Proof. Begin with source-route exactness, Definition 1006.10. It requires every supported hard-problem read to display the branch $G \longrightarrow S$ before any sibling consumer may use the boundary.

For the unsupported side, apply refusal route totality, Definition 1006.147. Every off-source closure request reaches R through the same structural rows and cannot branch to hidden closure, hidden T supply, host equality, private access, or an off-packet continuation.

Structural row conservation, Definition 1006.51, keeps H, C, P, N attached to the selected branch. These rows transport, replay, package, and name the displayed source or refusal route; they do not generate a third route.

Finally assemble the status surface with Definition 1006.78. The carrier, source/refusal split, root-unblock package, consumer exactness package, and TasteGate boundary are already recorded over the same packet G, S, R, H, C, P, N. Hence the source/refusal route completion is total exactly on the two displayed branches. □

Theorem 1006.149 (GapClosureBoundary NameCert route totality)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the root NameCert route is total exactly on the displayed carrier, source, and refusal rows. A supported read factors through the source branch*

$$G \longrightarrow S,$$

an unsupported read factors through the refusal ledger

$$G \longrightarrow R,$$

and both branches are transported by H*, replayed by* C*, packaged by* P*, and named by* N*. The route exports no hidden closure coordinate, hidden* T *supply, private source witness, host equality row, quotient datum, or off-packet continuation.*

Proof. Begin with the carrier and local certificate obligations, Definition 1006.1 and Definition 1006.2. They expose the finite BHist packet G, S, R, H, C, P, N and assign H, C, P, N to componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

For supported reads, apply source-route totality and route completion totality, Definition 1006.55 and Definition 1006.148. These keep the carrier/source branch on $G \longrightarrow S$ before downstream consumers inspect the structural rows.

For unsupported reads, apply refusal route totality and refusal NameCert factorization, Definition 1006.147 and Definition 1006.129. They send the same request row to R and keep the refusal endpoint inside H, C, P, N.

Structural row conservation then removes the remaining escape routes. Since H, C, P, N only transport, replay, package, and name the displayed source/refusal packet, the NameCert route is total exactly on the two branches. □

Theorem 1006.150 (GapClosureBoundary source readback)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *packet $B = (G, S, R, H, C, P, N)$, source readback is exhausted by the displayed refusal ledger together with the supported source branch. If a root request displays the source row, the readback is the branch $G \longrightarrow S$ under H, C, P, N; if it does not display the source row, the readback is the refusal ledger $G \longrightarrow R$ under the same structural rows. No additional readback row can convert the refusal ledger into source evidence or erase the displayed source branch.*

Proof. Read the supported side through terminal route totality, Definition 1006.128. It says a supported terminal read uses $G \longrightarrow S$ with the same componentwise transport, continuation, provenance, and local naming rows.

Read the unsupported side through refusal ledger totality and route completion totality, Definition 1006.41 and Definition 1006.148. These results exhaust off-source reads at R and keep them under H, C, P, N.

The source readback is therefore determined by whether S is displayed. The refusal ledger remains visible as the negative branch, and the accepted packet contains no further coordinate that can supply a second source witness or erase the branch distinction. □

Theorem 1006.151 (GapClosureBoundary downstream unblock package)**.** *For every accepted* GapClosureBoundary$^{\uparrow}$ *packet $B = (G, S, R, H, C, P, N)$, the downstream unblock surface for* CausalCommitment$^{\uparrow}$, FiniteWitnessedRefutation$^{\uparrow}$, OtherMindsCommitment$^{\uparrow}$, AspectChain$^{\uparrow}$, *and* SolipsismGapRefusal$^{\uparrow}$ *is exactly the package of the carrier row G, the source/refusal rows S, R, the componentwise* hsame *row H, the displayed* Cont *row C, the* Pkg *provenance row P, and the local* NameCert *row N. A downstream consumer may consume the supported source route or the refusal ledger, but it receives no consumer-specific hidden closure row, private access row, host equality row, quotient datum, or off-packet continuation.*

Proof. Start with the root unblock package, Definition 1006.39. It already lists the hard-problem consumers as readers of the shared source/refusal boundary rather than as owners of separate carrier data.

Next apply NameCert route totality and source readback from the two preceding theorems. These bind the carrier/source/refusal route to G, S, R and keep both supported and unsupported reads under H, C, P, N.

Finally use consumer surface exhaustion, Definition 1006.130. Since every listed downstream consumer is exhausted by the same finite packet, the downstream unblock package is precisely the displayed BHist, Gap, hsame, Cont, Pkg, and NameCert surface. □

Theorem 1006.152 (GapClosureBoundary certificate-gate row)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, a boundary read by* $\mathsf{AspectChain}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteReflectionTuple}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$*, or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *must pass through the displayed certificate-gate row before the source branch* $G \longrightarrow S$ *or refusal branch* $G \longrightarrow R$ *is public. The gate is carried only by the* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row.*

Proof. Start with recognizer source admission, Definition 1006.73. It sends an accepted source read through the $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ gate and then through the displayed $\mathsf{FiniteReflectionTuple}^{\uparrow}$ image.

Next use the local NameCert obligation surface, Definition 1006.84. It records carrier exposure, source-route admission, refusal-ledger exactness, consumer boundary exhaustion, and non-escape as rows of B.

Finally combine the hard-problem root package with source/refusal consumer totality. The listed consumers therefore receive a gated source or refusal branch, and no private gate outside G, S, R, H, C, P, N is exported. □

Theorem 1006.153 (GapClosureBoundary witnessed-refutation row)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, any* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ *or* $\mathsf{FiniteReflectionTuple}^{\uparrow}$ *readback of the boundary is forced onto the displayed source/refusal split. A supported read uses* $G \longrightarrow S$ *under* H, C, P, N*, an unsupported read uses* $G \longrightarrow R$ *under the same structural rows, and neither read imports a host refutation witness, quotient datum, hidden completion source, or consumer-specific continuation.*

Proof. Project the recognizer consumer exhaustion theorem, Definition 1006.75. It exhausts the recognizer-facing route by successful tuple landing or refusal landing.

For the positive side, use Definition 1006.73. The witnessed row reaches the boundary only after the gate and tuple readback, and then it is consumed as $G \longrightarrow S$.

For the refusal side, use Definition 1006.74. Failed or unsupported landings remain on $G \longrightarrow R$ and are transported only through H, C, P, N. □

Theorem 1006.154 (GapClosureBoundary solipsism-refusal row)**.** *Every* $\mathsf{SolipsismGapRefusal}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$*, or* $\mathsf{CausalCommitment}^{\uparrow}$ *consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet receives only the displayed source/refusal boundary rows* G, S, R, H, C, P, N*. In particular, a private solipsism escape, private other-minds access, direct aspect-reduction row, causal-necessity row, or hidden* T *supply is refused unless it is already represented by the public source branch or the refusal branch named by the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row.*

Proof. Begin with hard-problem consumer factorization and consumer exactness. These theorems place the listed sibling consumers inside the common boundary packet and divide their reads between the source branch and the refusal branch.

Use the certificate-gate row above together with the NameCert route completion surface. The local certificate names only G, S, R, H, C, P, N, so a consumer has no private row from which to read solipsism escape, other-minds access, aspect reduction, or causal necessity.

Finally apply refusal route exactness and refusal non-escape. Unsupported requests stay on the displayed refusal ledger R under hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, and cannot become hidden closure supply. □

Theorem 1006.155 (GapClosureBoundary root certificate-gate factorization)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *root carrier* $B = (G, S, R, H, C, P, N)$*, any certificate-gate read consumed by*

$\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *or* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *factors through the displayed source/refusal split*

$$G \longrightarrow S \qquad \text{or} \qquad G \longrightarrow R$$

and then through the same BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *rows* H, C, P, N. *The factorization exports no hidden* T *supply, host equality, quotient row, or private selector.*

Proof. Start from the accepted carrier $B = (G, S, R, H, C, P, N)$ and the existing carrier definition. The only public boundary rows are the source branch $G \longrightarrow S$, the refusal branch $G \longrightarrow R$, and the structural rows H, C, P, N.

Use the NameCert obligation surface and route-completion obligations. They require the certificate gate to read carrier admission, source/refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any downstream consumer receives the boundary.

Finally apply consumer ledger exhaustion for the hard-problem readers. Since the ledger names only the five downstream consumers and routes each of them through the same source/refusal split, there is no remaining coordinate for hidden supply, host equality, quotient data, or private selection. □

Theorem 1006.156 (GapClosureBoundary hard-problem consumer coverage)**.** *The root carrier* $B = (G, S, R, H, C, P, N)$ *covers the hard-problem consumers* $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, *and* $\mathsf{SolipsismGapRefusal}^{\uparrow}$ *exactly through the accepted source branch* $G \longrightarrow S$ *and the accepted refusal branch* $G \longrightarrow R$. *Each consumer receives its boundary as a* BHist *read transported by* hsame, *replayed by* Cont, *packaged by* Pkg, *and named by* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$.

Proof. First read the consumer ledger exhaustion theorem for the root packet. It lists the hard-problem consumers and assigns their supported requests to the source branch and their unsupported requests to the refusal branch.

Next use the recognizer landing obligations. A consumer that lands positively must pass through the source row S, while a consumer that fails to land remains on the refusal row R. Both landings are still reads of the same carrier B.

The status block records the chapter commitment for the same carrier. Combining it with the displayed route rows H, C, P, N shows that coverage is by the finite certificate gate alone, not by a hidden supply, host equality, quotient row, or private selector. □

Theorem 1006.157 (GapClosureBoundary TasteGate obligation triad)**.** *For the accepted carrier* $B = (G, S, R, H, C, P, N)$, *a future TasteGate proof for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *must discharge exactly the root carrier admission, source/refusal branch exactness, and hard-problem consumer coverage obligations. These obligations are expressed through the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *rows and do not add a hidden* T *supply, host equality, quotient row, or private selector.*

Proof. Carrier admission is the first obligation. It exposes G, S, R, H, C, P, N as the only accepted rows of the root packet.

Source/refusal exactness is the second obligation. The source branch $G \longrightarrow S$ handles supported recognizer landings, and the refusal branch $G \longrightarrow R$ handles unsupported closure requests.

Hard-problem consumer coverage is the third obligation. The preceding coverage theorem routes $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{AspectChain}^{\uparrow}$, and

$\mathsf{SolipsismGapRefusal}^{\uparrow}$ through the same carrier, so the triad exhausts the certificate-gate obligations needed by the TasteGate surface. □

Theorem 1006.158 (GapClosureBoundary source obligation scope)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, a supported read must display the source branch*

$$G \longrightarrow S$$

before it consumes the structural rows H, C, P, N*. The structural rows transport, replay, source, and name the displayed source branch; they do not replace it with hidden closure supply.*

Proof. Start with the carrier and local NameCert obligations, Definition 1006.1 and Definition 1006.2. They expose the finite rows G, S, R, H, C, P, N and assign H, C, P, N to transport, replay, provenance, and local naming.

Apply source route totality, Definition 1006.55. It requires the displayed source route $G \longrightarrow S$ before a supported hard-problem consumer reads the structural rows.

Finally use source-refusal orthogonality and consumer factorization, Definition 1006.46 and Definition 1006.47. These keep the source branch distinct from refusal and keep the consumer inside the shared packet, so no hidden source or closure row is available. □

Theorem 1006.159 (GapClosureBoundary refusal obligation scope)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier* $B = (G, S, R, H, C, P, N)$*, an unsupported read routes to the refusal branch*

$$G \longrightarrow R$$

with no hidden supply outside G, S, R, H, C, P, N*. The structural rows preserve the refusal endpoint and cannot convert it into source evidence, private access, causal necessity, direct aspect reduction, or hidden* T *supply.*

Proof. Begin with refusal route exactness, Definition 1006.56. It sends each unsupported closure request through $G \longrightarrow R$ with the structural rows H, C, P, N.

Use the local NameCert obligation surface, Definition 1006.84. It records refusal-ledger exactness and non-escape as obligations of the same finite carrier.

Now apply source-refusal orthogonality and refusal ledger stability. The refusal branch cannot be reread as a concealed source branch, and transport or replay preserves R rather than producing hidden supply. Thus the unsupported read remains on the displayed refusal obligation scope. □

Theorem 1006.160 (GapClosureBoundary consumer obligation scope)**.** *Every hard-problem consumer of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *carrier receives only the two displayed branches over* G, S, R, H, C, P, N*: supported reads consume* $G \longrightarrow S$*, and unsupported reads consume* $G \longrightarrow R$*. No consumer receives a third branch, hidden supply row, host equality, quotient datum, private source selector, or off-packet continuation.*

Proof. First assemble the supported branch. By Definition 1006.158, every supported read must display $G \longrightarrow S$ before the structural rows are consumed.

Next assemble the unsupported branch. By Definition 1006.159, every unsupported read remains on $G \longrightarrow R$ with no hidden supply.

Finally apply hard-problem consumer factorization and consumer ledger exhaustion, Definition 1006.47 and Definition 1006.57. They keep the listed consumers inside the same source/refusal split, so the two displayed branches exhaust the consumer obligation scope. □

Theorem 1006.161 (GapClosureBoundary scoped carrier admission)**.** *Every scoped boundary read accepted by* $\mathsf{GapClosureBoundary}^{\uparrow}$ *must first expose the same seven-row packet* $B = (G, S, R, H, C, P, N)$*. The rows* G, S, R *are the displayed* BHist *request, source, and refusal rows,* H *is the componentwise* hsame *transport row,* C *is the displayed* Cont *route row,* P *is the* Pkg *provenance row, and* N *is the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row. No accepted scoped read can replace any of these rows by hidden closure, hidden* T *supply, host equality, quotient data, or an off-packet continuation.*

Proof. Start with the scoped dependency inventory. By Definition 1006.79, the accepted boundary packet already displays G, S, R, H, C, P, N before any consumer read.

Next read the displayed rows through the root obligation packet and source-refusal obligation scope. The packet records G, S, R as the finite BHist rows and keeps H, C, P, N attached as transport, route, provenance, and local naming data.

Finally apply scoped consumer non-escape. Since the consumer surface has no coordinate for hidden closure, hidden T supply, host equality, quotient data, or off-packet continuation, a scoped read that omits one of the seven displayed rows is not an accepted $\mathsf{GapClosureBoundary}^{\uparrow}$ boundary read. □

Theorem 1006.162 (GapClosureBoundary scoped route totality)**.** *For every accepted scoped consumer read of* $\mathsf{GapClosureBoundary}^{\uparrow}$*, the route is exhausted by exactly one of the two displayed branches*

$$G \longrightarrow S$$

or

$$G \longrightarrow R$$

over the same rows H, C, P, N*. Supported reads use the source branch, unsupported reads use the refusal branch, and there is no third accepted source, refusal, bridge, or continuation route outside* G, S, R, H, C, P, N*.*

Proof. First expose the admitted packet. By Definition 1006.161, any accepted scoped read is already a read of G, S, R, H, C, P, N.

Apply scoped source-refusal exactness. Definition 1006.80 assigns supported reads to $G \longrightarrow S$ and unsupported reads to $G \longrightarrow R$ over the same structural rows H, C, P, N.

It remains to exclude residual routes. The scoped dependency inventory supplies no additional route coordinate, and scoped consumer non-escape prevents a consumer from importing a hidden source, hidden refusal, bridge-only continuation, quotient witness, or off-packet route. Thus the source and refusal branches exhaust the accepted scoped consumer reads. □

Theorem 1006.163 (GapClosureBoundary scoped TasteGate readback)**.** *Any* $\mathsf{TasteGate}$ *reader or* $\mathsf{StandardBridge}$ *reader for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *that reads the scoped boundary must read the same packet* $B = (G, S, R, H, C, P, N)$ *and the same source/refusal split of Definition 1006.162. Such a reader cannot use hidden closure, hidden* T *supply, host equality, quotient data, or an off-packet continuation to identify, complete, or replace the displayed boundary packet.*

Proof. Begin with the scoped carrier admission theorem. It fixes the carrier rows, transport row, route row, provenance row, and local naming row as the whole readable packet for scoped boundary access.

Next use scoped route totality. Every accepted scoped read available to a later $\mathsf{TasteGate}$ or $\mathsf{StandardBridge}$ reader has already been assigned to the displayed source branch or the displayed refusal branch over H, C, P, N.

Finally apply the TasteGate boundary and bridge precondition surface assembled in Definition 1006.78. Those rows require any TasteGate or bridge-facing read to preserve the same seven packet fields. Because the packet contains no hidden closure, hidden T supply, host equality, quotient coordinate, or off-packet continuation, such data cannot be used to identify, complete, or replace the scoped boundary. □

Theorem 1006.164 (GapClosureBoundary consumer obligation closure)**.** *Every accepted* GapClosureBoundary$^{\uparrow}$ *consumer route must expose the displayed* BHist *request row, classifier row, refusal row,* Cont *replay row,* Pkg *provenance row, and local* NameCert *row before it can read the boundary. The route may transport those rows by the recorded* hsame *data, but it cannot consume a hidden request, hidden classifier, unrecorded refusal, off-packet replay, detached provenance row, or unnamed boundary packet.*

Proof. Start from the root boundary obligation packet and the source-refusal obligation scope. They expose the finite request, classifier, refusal, replay, provenance, and naming rows as the public root surface of the carrier.

Use root NameCert route totality. Every accepted consumer route is named by the same local certificate and is replayed through the displayed Cont row rather than through a private continuation.

Finally apply refusal route totality and route-completion totality. The hsame transport rows preserve the displayed request/classifier/refusal packet, while Pkg provenance records the same source. Hence any route omitting one of these rows leaves the accepted consumer surface. □

Theorem 1006.165 (GapClosureBoundary source-refusal exactness)**.** *A source-refusal read of* GapClosureBoundary$^{\uparrow}$ *is exact only when the same displayed refusal row is consumed by every downstream route that reads the source boundary. If a downstream route replaces the refusal row, hides the classifier that selected it, or detaches it from the displayed* BHist*,* Cont*,* Pkg*, and* NameCert *rows, the route is no longer an exact source-refusal read.*

Proof. Begin with source-refusal consumer totality and source-refusal terminal unblock. They force the downstream source-facing route to consume the same refusal row that the root packet displays.

Next use the root source-refusal readiness and decision-surface theorems. The classifier row selects the refusal route before any consumer handoff, so a later route cannot replace the refusal row without changing the boundary decision.

The structural rows preserve exactness. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row carry the same refusal row to each consumer. A route that hides or swaps any of them exits the source-refusal packet. □

Theorem 1006.166 (GapClosureBoundary refusal handoff exhaustion)**.** *No accepted* GapClosureBoundary$^{\uparrow}$ *consumer handoff can use a hidden source, hidden classifier, or unrecorded gap route. Every handoff is exhausted by the displayed request row, source/refusal decision rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row already present in the root packet.*

Proof. Project the root acceptance route exhaustion theorem. It leaves only the displayed request, source/refusal, transport, replay, provenance, and naming rows available to a consumer handoff.

Apply the consumer obligation closure theorem. Any accepted route has to expose the request, classifier, refusal, replay, provenance, and local certificate rows before handoff.

Finally use source-refusal exactness. Because every downstream route consumes the same displayed refusal row, no hidden source, hidden classifier, or unrecorded gap route remains available. The handoff is therefore exhausted by the root packet. □

Theorem 1006.167 (GapClosureBoundary L10 terminal obligation pack)**.** *Every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet routes an L10 terminal consumer through the displayed* BHist *request, source, and refusal rows; the componentwise* hsame *transport row; the displayed* Cont *replay row; the* Pkg *provenance row; the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row; and the downstream* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *rows before the terminal boundary is exported. No L10 terminal consumer receives a hidden closure row, hidden Real limit, anonymous stream name, host equality, quotient datum, or off-packet continuation.*

Proof. Begin with the L10 terminal consumer boundary. It fixes the terminal interface to the source and refusal branches over the structural rows H, C, P, N and excludes hidden completion supply.

Use the root obligation closure handoff next. It requires any root-facing consumer to expose the same request, classifier, refusal, replay, provenance, and naming rows before the boundary can be read.

Finally attach the real-completion consumers. The $\mathsf{Real}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows are therefore read only after the finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support has already been displayed, so the terminal export has no coordinate for hidden Real or stream-name supply. □

Theorem 1006.168 (GapClosureBoundary Real and StreamName dependency)**.** *Any real-completion gap closure read of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet must display both the* $\mathsf{Real}^{\uparrow}$ *row and the* $\mathsf{StreamName}^{\uparrow}$ *row before closure is exported. A read that displays only one of these rows remains a partial terminal request and cannot be exported as a real-completion gap closure.*

Proof. Start from the root source-refusal obligation scope. It separates supported source reads from refusal reads before any terminal consumer inspects completion-facing data.

Apply scoped root unblock obligations. They keep the root packet inside the displayed source/refusal split and prevent a downstream consumer from replacing the missing row by hidden supply or off-packet replay.

Thus the $\mathsf{Real}^{\uparrow}$ row and the $\mathsf{StreamName}^{\uparrow}$ row must both be visible at the terminal read. If either row is absent, the source/refusal boundary has not supplied the data needed for real-completion export, and the read remains outside the accepted closure surface. □

Theorem 1006.169 (GapClosureBoundary finite refusal consumer exhaustion)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet, the consumer-facing boundary is exhausted by finite witnessed refusal, source refusal, and readback handoff rows together with the structural rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No consumer-facing boundary row remains for hidden source selection, hidden completion supply, Real-row synthesis, StreamName-row synthesis, or off-packet terminal replay.*

Proof. First use the L10 terminal consumer boundary to expose the two terminal branches available to downstream consumers.

Next use the root source-refusal obligation scope. It records the source-refusal split as part of the same accepted packet rather than as a detachable consumer-side choice.

Finally apply scoped root unblock obligations and root obligation closure handoff. These rows identify the readback handoff and close the root-facing inventory, so finite witnessed refusal, source refusal, and readback handoff exhaust the consumer-facing boundary. □

Theorem 1006.170 (GapClosureBoundary L10 terminal source lock)**.** *For every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet* $B = (G, S, R, H, C, P, N)$*, the root route exported to the L10 terminal source surface is locked*

to the displayed rows

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}},$$
$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Completion}^{\uparrow}.$$

The source-refusal route may reach a terminal $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *consumer only after passing through the source-refusal admission triad of Definition 1006.119, the recognizer landing obligations of Definition 1006.73 and Definition 1006.74, and the route-completion surface of Definition 1006.125. The export carries no hidden completion witness, selected Real limit, anonymous stream name, host equality row, quotient datum, or off-packet continuation.*

Proof. Project the accepted packet to its finite source-refusal inventory. The carrier contributes G, S, R, while H, C, P, N provide the componentwise hsame transport, displayed Cont replay, package provenance, and local naming rows.

Next apply the source-refusal admission triad and the recognizer landing obligations. Successful recognizer landings are read as $G \longrightarrow S$, while failed or unsupported landings are read as $G \longrightarrow R$ under the same structural rows.

Finally apply route completion and the L10 terminal obligation pack. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ consumers receive the packet only after the displayed BHist, hsame, Cont, Pkg, and NameCert rows have been exposed, so the excluded hidden coordinates have no terminal source slot. □

Theorem 1006.171 (GapClosureBoundary root consumer exactness)**.** *Every root consumer route of an accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet is exact with respect to the same L10 terminal source surface. A supported root consumer must read the branch* $G \longrightarrow S$ *before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *row is inspected; an unsupported root consumer must read the branch* $G \longrightarrow R$ *before the same terminal rows are inspected. Both branches remain transported by* H*, replayed by* C*, packaged by* P*, and named by* N*.*

Proof. Begin with the terminal source lock above. It fixes the only terminal source surface available to the root route as the displayed BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ rows.

For the supported branch, use recognizer source admission together with route completion. These force successful evidence through $G \longrightarrow S$ before any terminal consumer can read the downstream rows.

For the unsupported branch, use recognizer refusal stability together with refusal ledger exhaustion. The request remains at $G \longrightarrow R$ under H, C, P, N and cannot be reread as source evidence or completion supply.

The Real and StreamName dependency theorem then supplies the terminal exactness check: a real-completion export needs the named $\mathsf{Real}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows together, and the $\mathsf{Completion}^{\uparrow}$ consumer receives them only through the same source-refusal packet. □

Theorem 1006.172 (GapClosureBoundary terminal unblock package)**.** *The terminal unblock package consumed by the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *boundary chapters is the finite packet consisting of the root request row, displayed source row, displayed refusal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *naming, and the terminal* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Completion}^{\uparrow}$ *rows. A downstream Real or Completion boundary may cite this package as a source-locked unblock surface, but it may not extract an ambient completeness principle, quotient stream equality, selected limit, hidden source selector, or off-packet continuation from it.*

Proof. First assemble the root surface from the terminal source lock and root consumer exactness. These two projections identify the source branch, refusal branch, structural rows, and terminal rows as one finite packet.

Next add the L10 terminal obligation pack. It places $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ behind the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, while the root route keeps $\mathsf{Completion}^{\uparrow}$ on the same consumer boundary.

Finally use finite refusal consumer exhaustion. Unsupported terminal requests are absorbed by the displayed refusal ledger, and supported terminal requests remain on the displayed source branch. Thus Real and Completion consumers receive exactly the source-locked unblock package and no ambient or off-packet resource. □

Theorem 1006.173 (GapClosureBoundary L10 source-lock totality)**.** *Every L10 terminal* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consumer that reads a completion-facing refusal must pass through the source-refusal branch, the source-classifier row, the route-completion row, the terminal-route row, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row before the refusal is public. A terminal read that bypasses any one of those rows is outside the accepted packet* $B = (G, S, R, H, C, P, N)$*.*

Proof. Start with the source-refusal admission triad and source-refusal totality. They split every accepted request into the displayed source branch $G \longrightarrow S$ or refusal branch $G \longrightarrow R$ over the structural rows H, C, P, N.

Next add the classifier and route-completion rows. Definition 1006.83 keeps supported and unsupported requests separated under hsame transport and Cont replay, while Definition 1006.125 keeps the read inside the local NameCert surface.

Finally use terminal route totality together with the L10 terminal source lock. The terminal consumer can inspect a completion-facing refusal only after the source/refusal branch, classifier, route-completion, terminal-route, and NameCert rows have all been displayed, so no hidden completion witness or off-packet refusal remains. □

Theorem 1006.174 (GapClosureBoundary L10 source lock)**.** *Every L10 consumer of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *that reads a source or refusal row over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$ *must expose the same finite* BHist *source branch, the source-classifier row, the refusal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row before the consumer may pass the row downstream.*

Proof. Start from the source-refusal totality and source-lock totality rows. They make every accepted completion-facing request pass through the displayed source branch or refusal branch before it is public.

Next use the root source-classifier route totality and route-completion obligations. The classifier determines which branch is visible, and the route-completion row keeps the read inside the local NameCert surface.

Finally read the L10 consumers only through their displayed rows. $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ receive the source lock through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data, not through an off-packet completion witness. □

Theorem 1006.175 (GapClosureBoundary terminal source handoff)**.** *The terminal* $\mathsf{GapClosureBoundary}^{\uparrow}$ *handoff to L10 completion consumers is exactly the source/refusal packet supplied by the source-refusal admission route, the scoped source-refusal readback, the terminal route totality, and the L10 source-lock totality. The handoff has no coordinate for parameter transport equivalence, quotient completion equality, host real equality, or hidden source completion outside the displayed rows.*

Proof. Use the source-refusal admission package first. It supplies the visible source branch and refusal branch together with the structural rows required by the hub packet.

Then apply scoped source-refusal readback and terminal route totality. These theorems keep terminal reads on the same displayed route and prevent a downstream completion consumer from importing a hidden branch.

The L10 source-lock totality gives the final public boundary. Since the accepted handoff consists only of the displayed BHist, classifier, refusal, hsame, Cont, Pkg, and NameCert rows, it cannot assert parameter transport equivalence or any host-completion equality. □

Theorem 1006.176 (GapClosureBoundary root L10 unblock package)**.** *The root L10 unblock surface for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *consists of the source lock of Definition 1006.174, the terminal handoff of Definition 1006.175, and the existing route-totality and source-readback rows. Any downstream L10 consumer over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *must consume this finite package before claiming a completion-facing refusal or source read.*

Proof. Assemble the source-facing rows. The root source-classifier route totality, root request source-refusal totality, scoped source-refusal readback, and L10 source-lock totality provide the source branch, refusal branch, classifier, readback, and terminal visibility rows.

Add the structural rows. Componentwise hsame transport preserves the branch, displayed Cont replay carries the route, Pkg provenance records the source of the consumer handoff, and local NameCert data names the accepted packet.

The six L10 consumers named in the statement are therefore downstream readers of this finite source-lock package. They may consume the handoff, but they do not receive a separate parameter-equivalence claim, quotient equality, host real equality, or hidden completion source. □

Theorem 1006.177 (GapClosureBoundary terminal source-route exposure)**.** *Every root-unblock consumer of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *that reaches a terminal* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *facing gap-closure read must first expose the displayed source route of the accepted carrier* $B = (G, S, R, H, C, P, N)$*. The read passes through the* BHist *request row* G*, the supported source branch* $G \longrightarrow S$ *or refusal branch* $G \longrightarrow R$*, the componentwise* hsame *row* H*, the displayed* Cont *row* C*, the* Pkg *row* P*, and the local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *row* N *before any terminal completion-facing row is consumed.*

Proof. Begin with the L10 source lock of Definition 1006.174. It places every completion-facing consumer behind the same finite source and refusal route over G, S, R, H, C, P, N.

Use the terminal source handoff of Definition 1006.175. The handoff identifies the terminal boundary with the source-refusal packet rather than with parameter transport, quotient completion equality, host real equality, or hidden source completion.

Finally apply the root L10 unblock package of Definition 1006.176. Its downstream consumers over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ may read terminal rows only after the displayed source route and local NameCert row have been exposed. □

Theorem 1006.178 (GapClosureBoundary terminal refusal-consumer totality)**.** *Every root-unblock consumer of* $\mathsf{GapClosureBoundary}^{\uparrow}$ *that cannot read the supported source branch must read the refusal route* $G \longrightarrow R$ *under the same* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *rows before it consumes any terminal* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *gap-closure row. No accepted root-unblock consumer has a third terminal route outside the displayed source/refusal split.*

Proof. Start with terminal route totality, Definition 1006.128. It says every terminal read chooses exactly the supported branch $G \longrightarrow S$ or the refusal branch $G \longrightarrow R$ over the same structural rows.

For unsupported reads, apply refusal NameCert factorization from Definition 1006.129. The request is recorded at R, transported through H, replayed through C, packaged by P, and named by N.

The L10 source-lock totality theorem, Definition 1006.173, then blocks terminal consumption until the refusal branch, route-completion row, terminal route, and local NameCert row have all been displayed. Thus the root-unblock consumer has no off-packet terminal route. □

Theorem 1006.179 (GapClosureBoundary terminal NameCert unblock package)**.** *The terminal root-unblock package for* $\mathsf{GapClosureBoundary}^{\uparrow}$ *is the finite source-route, refusal-route, terminal-handoff, and local* $\mathsf{NameCert}_{\mathsf{GapClosureBoundary}^{\uparrow}}$ *surface assembled from* $B = (G, S, R, H, C, P, N)$*. Every root-unblock consumer must read that package before any* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{StreamName}^{\uparrow}$ *facing gap closure is consumed, and the package exports no hidden Real limit, anonymous regular sequence, anonymous stream name, host equality, quotient datum, or off-packet completion witness.*

Proof. Assemble the source route from Definition 1006.177. It exposes the request row, supported branch, structural rows, provenance, local naming, and terminal handoff before a completion-facing consumer may read.

Assemble the refusal route from Definition 1006.178. Unsupported root-unblock consumers are routed through R, H, C, P, N and cannot bypass the local NameCert packet.

Finally use the L10 terminal source lock and root L10 unblock package, Definition 1006.170 and Definition 1006.176. Together they bind $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ facing gap closure to the same finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface, so no terminal consumer receives an unrecorded completion resource. □

Theorem 1006.180 (GapClosureBoundary consumer bridge admission)**.** *Every accepted* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet exposes the finite split* $B = (G, S, R, H, C, P, N)$ *as the common bridge surface for downstream consumers. The request row* G*, supported source route* $G \longrightarrow S$*, unsupported refusal route* $G \longrightarrow R$*, componentwise transport* H*, displayed continuation replay* C*, provenance package* P*, and local name row* N *are the only admitted rows through which* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$*, and the neighboring hard-problem consumers may read the boundary.*

Proof. Start with the NameCert obligations of Definition 1006.2. They expose the accepted carrier as the finite packet G, S, R, H, C, P, N and assign the local naming row to that packet.

Apply source-route totality and refusal-route exactness, Definition 1006.55 and Definition 1006.56. These two results make the supported branch the displayed route $G \longrightarrow S$ and the unsupported branch the displayed route $G \longrightarrow R$.

Finally use the consumer exactness package and hub root readiness, Definition 1006.54 and Definition 1006.59. The downstream consumers therefore receive the same bridge surface and no consumer-specific row outside the packet. □

Theorem 1006.181 (GapClosureBoundary CausalCommitment forward-GAP route)**.** *The* $\mathsf{CausalCommitment}^{\uparrow}$ *consumer may read a supported* $\mathsf{GapClosureBoundary}^{\uparrow}$ *branch only through the displayed forward-GAP route* $G \longrightarrow S$ *over the shared rows* H, C, P, N*. It cannot obtain a hidden causal-necessity row, a private source selector, or an off-packet continuation from the accepted boundary packet.*

Proof. Use the consumer bridge admission of Definition 1006.180. It fixes the common downstream surface before the causal consumer reads any part of the packet.

On the supported side, apply Definition 1006.55. The only admitted supported route is $G \longrightarrow S$, and the structural rows available to the consumer are exactly H, C, P, N.

The exclusions follow from Definition 1006.54 and Definition 1006.59. Since the hub-ready packet exports no causal-necessity coordinate, no private source selector, and no off-packet continuation, $\mathsf{CausalCommitment}^{\uparrow}$ can consume the supported branch only through the displayed route. □

Theorem 1006.182 (GapClosureBoundary finite-refutation refusal route)**.** *The* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ *consumer may read an unsupported* $\mathsf{GapClosureBoundary}^{\uparrow}$ *branch only through the displayed refusal route* $G \longrightarrow R$ *over the shared rows* H, C, P, N*. The refusal row cannot be converted into a hidden contradiction oracle, host-level refutation, quotient witness, or off-packet inconsistency source.*

Proof. Begin again with the consumer bridge admission of Definition 1006.180. It identifies the finite packet and prevents the finite-refutation consumer from reading a separate carrier.

For the unsupported branch, apply Definition 1006.56. The branch is routed to R from G and remains transported, replayed, packaged, and named only through H, C, P, N.

Finally use Definition 1006.2, Definition 1006.54, and Definition 1006.59. These anchors leave no row from which $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ could extract a hidden contradiction oracle, host-level refutation, quotient witness, or off-packet inconsistency source. □

The hub-level $\mathsf{GapClosureBoundary}^{\uparrow}$ route consumes the finite witnessed refutation surface of chapter 576 and the solipsism refusal surface of chapter 975 before exporting the causal, other-minds, aspect, and finite-trace consumer lattice of chapter 505, chapter 499, chapter 575, and chapter 818. Each consumer sees only the displayed source/refusal rows already routed by the hub packet. The $\mathsf{GapClosureBoundary}^{\uparrow}$ dossier route consumes finite witnessed refutation from chapter 576, solipsism gap refusal from chapter 975, bounded-search refutation surfaces, and finite trace evaluator data from chapter 818 before it exports refusal routing to claim-audit, causal, and other-minds consumers. These sibling dependencies keep the boundary as finite route data: the hub may cite the assembled refusal lattice, but no consumer receives an unbounded evaluator, hidden search choice, or private gap oracle. The $\mathsf{GapClosureBoundary}^{\uparrow}$ dependency route to chapter 576 is read before any contrary witness is interpreted. The displayed boundary row fixes whether the consumer receives the supported source branch or the refusal branch, while the bounded search surface of chapter 579 supplies only the finite search ledger used after that branch has been selected.

Lemma 1006.183 (GapClosureBoundary finite witnessed-refutation route)**.** *Any* $\mathsf{FiniteWitnessedRefutation}^{\uparrow}$ *consumer of a* $\mathsf{GapClosureBoundary}^{\uparrow}$ *packet first consumes the displayed source/refusal row of the boundary packet. It may then read the bounded-search witness ledger, but it receives no hidden closure row, off-packet contradiction oracle, or unbounded search choice from the boundary.*

Chapter 1007

A Concrete Naming Certificate for $\mathsf{FiniteObservationInterface}^{\uparrow}$

The $\mathsf{FiniteObservationInterface}^{\uparrow}$ packet realizes the Real and Completion target of chapter 9238 as a finite read surface. It consumes the StreamName observation boundary of chapter 126, the RegSeqRat regularity surface of chapter 301, the DyadicRatCore tolerance surface of chapter 291, and the RealUp handoff surface of chapter 38. It is narrower than a completion theorem: the packet exposes a selected finite observation window, records the omitted tail as a ledger, and then routes every real-facing read through the displayed sibling rows.

Definition 1007.1 (Finite observation interface carrier)**.** *A* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, R, D, T, E, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row whose finite read entries are visible before any real-facing handoff;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row that reads the same observation entries as regular rational data;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row accounting for every displayed precision request and dyadic error bound used by the read;*
4. *T is the selected finite-window row, listing the exact bounded observation window exposed to the consumer;*
5. *E is the omitted-tail ledger, recording precisely which StreamName and RegSeqRat entries lie outside T and are not exported as selected rows;*
6. *H stores componentwise* hsame *transports for the observation row, regularity row, tolerance row, selected window, omitted-tail ledger, provenance, and local naming row;*
7. *C stores the displayed* Cont *replay routes that read S, then R, then D, expose T, record E, and only then hand off to the RealUp row consumed by the downstream route;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *selected finite-window rows, omitted-tail ledgers,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationInterface}^{\uparrow}}$ *row naming exactly this finite observation interface packet.*

The carrier has no coordinate for a full Real completeness proof, quotient stream equality, host real equality, selected limit, or export of the omitted tail as a selected stream row.

Lean4 ✓ `BEDC.Derived.FiniteObservationInterfaceUp.FiniteObservationInterfaceTasteGate single carrier alignment`

Theorem 1007.2 (Finite observation interface NameCert obligations)**.** *For every accepted* FiniteObservationInterfac *carrier* $O = (S, R, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationInterface}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier admission: the accepted packet displays exactly the rows* $S, R, D, T, E, H, C, P, N$ *of Definition 1007.1;*
2. *selected-window exposure: the finite window* T *is exposed before any RealUp handoff is made available to a consumer;*
3. *omitted-tail ledger exactness: the ledger* E *records the entries outside* T *and cannot be read as a selected stream row;*
4. *dyadic tolerance accountability: every precision request used by the read is accounted for by the dyadic row* D *before the route reaches the RealUp surface;*
5. *StreamName/RegSeqRat/Real handoff non-escape: every handoff to RealUp factors through* S*,* R*,* D*,* T*,* E*,* H*,* C*,* P*, and* N*, and no quotient stream equality, host real equality, full Real completeness, or selected limit is exported.*

Lean4 ✓ `BEDC.Derived.FiniteObservationInterfaceUp.FiniteObservationInterfaceNameCert obligation surface`

Proof. Carrier admission is projection from Definition 1007.1. The StreamName observation row, RegSeqRat regularity row, DyadicRatCore tolerance row, selected window, omitted-tail ledger, transport, continuation, provenance, and naming coordinates are all displayed in the same finite packet.

The selected-window and omitted-tail obligations are separated by the two rows T and E. A consumer may inspect the selected finite window T; it may also inspect the ledger E as the exact record of entries outside that window. The ledger row is not a second selected stream row, because the carrier gives it only the omitted-tail role.

Dyadic accountability is read through D before the Real-facing route is exposed. The replay rows H, C, P, N keep the StreamName, RegSeqRat, dyadic, selected-window, and tail-ledger reads in one local NameCert packet, so the Real handoff has no coordinate from which to obtain completeness, quotient equality, host equality, or a selected limit. □

Theorem 1007.3 (Finite observation interface selected-stream handoff)**.** *Every Real-facing finite observation read of an accepted* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *packet routes through the displayed StreamName row* S*, RegSeqRat row* R*, DyadicRatCore row* D*, selected finite-window row* T*, omitted-tail ledger* E*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. The route cannot assert full Real completeness, quotient stream equality, host real equality, or the existence of a selected limit.*

Lean4 ✓ `BEDC.Derived.FiniteObservationInterfaceUp.FiniteObservationInterfaceSelectedStream handoff`

Proof. Start with a Real-facing consumer read. By the NameCert obligations, the read reaches the RealUp side only after the StreamName observation row S, the RegSeqRat regularity row R, and the DyadicRatCore tolerance row D have been displayed.

The selected stream information is then exactly the finite window T. The omitted-tail ledger E is read as a boundary record for what the packet withholds from the selected window; it is not exported through the same classifier role as T.

Finally replay the route through H, C, P, N. These rows transport and package the displayed finite packet, and they do not add a coordinate for ambient Real completeness, quotient stream equality, host real equality, or limit selection. Thus every Real-facing finite observation read remains inside the displayed interface route. □

Theorem 1007.4 (Finite observation interface scoped closure)**.** *Every accepted* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *read is scoped exactly by the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of Definition 1007.1, together with the sibling* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. Selected-window exposure, omitted-tail exactness, dyadic accountability, and Real-facing non-escape are all read through the same finite packet*

$$O = (S, R, D, T, E, H, C, P, N).$$

No coordinate of the scoped read supplies full Real completeness, quotient stream equality, host real equality, or a selected-limit coordinate.

Proof. Project the carrier of Definition 1007.1. This gives the finite packet $O = (S, R, D, T, E, H, C, P, N)$ and fixes the displayed BHist, transport, continuation, package, and naming rows as the only rows available to the interface read.

Apply Definition 1007.2. Carrier admission binds the packet to those displayed rows, selected-window exposure is read through T, omitted-tail exactness is read through E, dyadic accountability is read through D, and the StreamName/RegSeqRat/Real handoff condition forces every Real-facing route through the same local packet.

Apply Definition 1007.3. The sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces are therefore used only along the displayed finite route. Since the carrier has no coordinate for full Real completeness, quotient stream equality, host real equality, or a selected limit, none of those claims can escape the scoped packet. □

Theorem 1007.5 (Finite observation interface public certificate route)**.** *Every accepted* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *packet*

$$O = (S, R, D, T, E, H, C, P, N)$$

exports a public reusable certificate only through the displayed StreamName row S, RegSeqRat row R, DyadicRatCore row D, selected finite-window row T, omitted-tail ledger E, transport row H, replay row C, provenance row P, and local $\mathsf{NameCert}$ *row N. The public route does not export Real completeness, a selected limit, quotient stream equality, host real equality, or an omitted-tail-as-selected-stream row.*

Proof. Project the carrier in Definition 1007.1. The accepted packet has exactly the rows $S, R, D, T, E, H, C, P, N$, and the carrier explicitly withholds Real completeness, quotient stream equality, host real equality, selected limits, and any selected-stream role for the omitted tail.

Apply Definition 1007.2. The local NameCert obligations force carrier admission, selected-window exposure, omitted-tail exactness, dyadic tolerance accountability, and StreamName/RegSeqRat/Real handoff non-escape to be read from those rows.

Apply Definition 1007.3 and Definition 1007.4. The reusable surface is therefore public only as the displayed finite certificate route, and no hidden Real-facing coordinate is available outside that route. □

Theorem 1007.6 (Finite observation interface consumer reuse)**.** *Any downstream L10 consumer that reuses an accepted* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *packet must reuse the public certificate route of Definition 1007.5. Such a consumer may read the StreamName, RegSeqRat, DyadicRatCore, selected-window, omitted-tail, transport, replay, provenance, and NameCert rows, but it cannot bypass those rows to obtain an independent RealUp handoff.*

Proof. Begin with a downstream L10 consumer of the accepted packet. By Definition 1007.5, the only reusable certificate exported by the packet is the displayed route through $S, R, D, T, E, H, C, P, N$.

The route itself is constrained by Definition 1007.2. In particular, the selected window is exposed before any RealUp handoff, the omitted tail remains a ledger rather than a selected stream, and dyadic accountability is supplied before the Real-facing read.

Use Definition 1007.3 together with Definition 1007.4. Every Real-facing read is already scoped by the finite packet and its sibling surfaces, so a downstream consumer has no bypass row from which to form an independent handoff. □

Theorem 1007.7 (Finite observation interface L10 no-escape public boundary)**.** *The public* FiniteObservationInterf *route for an accepted packet* $O = (S, R, D, T, E, H, C, P, N)$ *is an L10 boundary: it exports the displayed finite certificate rows and nothing beyond them. In particular, no downstream L10 reuse of the route exports Real completeness, a selected limit, quotient stream equality, host real equality, or an omitted-tail-as-selected-stream row.*

Proof. The carrier description in Definition 1007.1 fixes the packet inventory. It names the selected finite window T and the omitted-tail ledger E as distinct rows, and it gives no coordinate for completeness, quotient equality, host equality, or limit selection.

Apply Definition 1007.2 to bind those rows to the local NameCert surface. The non-escape obligation says that every handoff to RealUp factors through $S, R, D, T, E, H, C, P, N$ and through no extra Real-facing row.

Finally combine Definition 1007.3, Definition 1007.4, and Definition 1007.5. The public route is reusable by L10 consumers, but its reuse is still the same finite boundary, so the listed non-export claims remain unavailable. □

Theorem 1007.8 (Finite observation interface L10 bridge)**.** *For every accepted packet* $O = (S, R, D, T, E, H, C, P, N)$*, the public certificate route of Definition 1007.5 is exactly the L10 bridge surface consumed by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readers. Every bridge read factors through* $S, R, D, T, E, H, C, P, N$*, and no bridge read exports Real completeness, selected limits, quotient stream equality, host real equality, or omitted-tail-as-selected-stream data.*

Proof. Start from the public reusable certificate route. By Definition 1007.5, the exported surface is the displayed finite route through $S, R, D, T, E, H, C, P, N$, with no additional Real-facing coordinate.

The selected-stream handoff of Definition 1007.3 identifies the only route by which a Real-facing finite observation read can reach the sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surfaces. The scoped closure theorem Definition 1007.4 keeps that route inside the same BHist, hsame, Cont, Pkg, and NameCert rows.

Now apply consumer reuse from Definition 1007.6. Any L10 reader using the packet must reuse the public certificate route, so its bridge read has no bypass around $S, R, D, T, E, H, C, P, N$.

Finally use the L10 no-escape boundary of Definition 1007.7. Since the public route exports only the displayed finite certificate rows, the bridge surface consumed by the four downstream readers cannot export Real completeness, selected limits, quotient stream equality, host real equality, or omitted-tail-as-selected-stream data. □

Falsifiable prediction: Every accepted FiniteObservationInterfaceUp packet exposes a selected finite observation window before any RealUp handoff and cannot export an omitted tail as a selected stream row.

Independence witness: streamname, regseqrat, dyadicratcore, real, finitewindowenvelope: the interface packet combines selected-window, omitted-tail, dyadic tolerance, regularity, and Real handoff rows; each sibling lacks at least one of these carrier rows, so no bijection preserves the full field inventory.

The Lean file `Derived/FiniteObservationInterfaceUp/TasteGate.lean` must provide Nontrivial FiniteObser and FieldFaithful FiniteObservationInterface$^{\uparrow}$ instances for the displayed coordinates $S, R, D, T, E, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationInterface}^{\uparrow}}$.

Constructive story: A FiniteObservationInterface$^{\uparrow}$ packet is generated from a StreamName observation row, RegSeqRat regularity row, DyadicRatCore tolerance row, selected finite-window row, omitted-tail ledger, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.FiniteObservationInterfaceUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteObservationInterface}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1007.1, Definition 1007.2, Definition 1007.3, Definition 1007.4, Definition 1007.5, and Definition 1007.8 bind the finite observation carrier, local NameCert obligation surface, selected-window route, omitted-tail boundary, dyadic accountability, Real-facing non-escape scope, reusable public certificate route, and L10 bridge surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteObservationInterface}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteObservationInterfaceUp.taste\_gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close Real completeness, selected-limit existence, quotient stream equality, host real equality, Lean verification, or bridge checking.

Upgrade path: No further theory-axis closure is claimed here; remaining verification-axis work is auditCleanV or bridgeCheckedV for the displayed finite observation interface.

Chapter 1008

A Concrete Naming Certificate for $\mathsf{MetaCICRedexFrontier}^{\uparrow}$

The $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ packet realizes the beta-redex part of the MetaCIC subject-reduction frontier as finite certificate data. It reads the open-problem inventory of chapter 9386, the finite row catalogue of chapter 780, and the normalization boundary of chapter 892. Its purpose is narrower than those ledgers: it binds the four subject-reduction discharge rows to a beta-boundary route and to an obstruction ledger, so a consumer can distinguish an incomplete beta-redex packet from a complete frontier carrier.

The packet is composite over the MetaCIC open-problem ledger, MetaCIC normalization frontier, beta-boundary route, and obstruction-ledger siblings; those rows jointly supply the carrier surface.

Falsifiable prediction: A packet lacking the beta-redex row but carrying the other three discharge rows is rejected by the redex-frontier carrier.

Independence witness: metacicopenproblemledger, metacicnormalizationfrontier, metacic-substitution-audit-packet, beta-critical-pair, apophaticname: those siblings either inventory open rows, record normalization boundaries, isolate substitution windows, expose critical pairs, or refuse external supply; none is bijective with the complete redex-frontier carrier because this packet requires all four discharge rows plus the obstruction ledger in one carrier.

Definition 1008.1 (MetaCIC redex-frontier carrier)**.** *A* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, A, L, P, O, E, H, C, G, N)$$

with the following displayed rows:

1. *B is the beta-redex discharge row, carrying* BetaSubstitutionPreservation *for the beta case;*
2. *A is the application-argument discharge row, carrying* AppArgTypeStable *for reduction under the argument of an application;*
3. *L is the lambda-domain discharge row, carrying* LamDomainSubjectReduction *for binder-domain reduction inside lambda formation;*
4. *P is the Pi-domain discharge row, carrying* PiDomainSubjectReduction *for binder-domain reduction inside Pi formation;*
5. *O is the obstruction ledger row recording that the four rows are required because closed subject reduction is not exported at this boundary;*

6. *E is the beta-boundary route row that places beta-redex consumption before any subject-reduction handoff;*
7. *H records componentwise* hsame *transports for B, A, L, P, O, E together with provenance and local naming rows;*
8. *C records the displayed* Cont *routes from the four discharge rows to the beta-boundary route and from that route to subject-reduction consumers;*
9. *G is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, SubjectReductionDischargeBundle, MetaCICSubjectReductionFrontier$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BetaCriticalPair$^{\uparrow}$, *and the open-problem ledger rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with all four discharge rows B, A, L, P and the obstruction ledger O present in the same carrier. It has no coordinate for a beta-only shortcut, a three-row discharge interface, a hidden substitution oracle, a closed subject-reduction theorem, or a consumer route that bypasses E.

Lean4 (definition) `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier`

Theorem 1008.2 (MetaCICRedexFrontier NameCert obligations)**.** *For the carrier of Definition 1008.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for $R = (B, A, L, P, O, E, H, C, G, N)$;*
2. *four-row discharge coverage for beta-redex, application-argument, lambda-domain, and Pi-domain subject-reduction cases;*
3. *beta-boundary routing through E before any subject-reduction consumer can read the discharge packet;*
4. *obstruction ledger exposure through O, separating the finite route from a closed subject-reduction theorem;*
5. *componentwise transport and replay through H and C;*
6. *provenance and local naming exhaustion through G, N, with no beta-only shortcut, hidden substitution oracle, or off-packet consumer route.*

The Lean follow-up must provide `Nontrivial MetaCICRedexFrontierUp` *and* `FieldFaithful MetaCICRedexFron` *in* `lean4/BEDC/Derived/MetaCICRedexFrontierUp/TasteGate.lean`.

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1008.1. The displayed tuple $R = (B, A, L, P, O, E, H, C, G, N$ is the full row inventory available to an accepted redex-frontier reader.

The four discharge rows are read next. The beta row B supplies the substitution-preservation case, while A, L, P supply the application-argument, lambda-domain, and Pi-domain cases. Since the classifier requires all four rows, a packet that omits B is not the displayed carrier even if A, L, P are present.

The boundary and obstruction rows are then read through E and O. The row E is the only route by which beta-redex consumption reaches a subject-reduction consumer, and O records why this finite route is not being replaced by a closed subject-reduction theorem.

Finally, H, C, G, N transport, replay, package, and name the same finite surface. The carrier has no coordinate for a beta-only shortcut, hidden substitution oracle, or off-packet consumer path, so the six obligations exhaust the local certificate boundary. □

Theorem 1008.3 (MetaCICRedexFrontier beta-boundary routing)**.** *Every accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier routes beta-redex consumption through*

$$B \to E \to C$$

before a subject-reduction consumer may read the packet. The remaining discharge rows A, L, P *travel with the same carrier, but none of them replaces the beta row* B *or the beta-boundary route* E*.*

Proof. Project the carrier. The beta-redex row is B, the beta-boundary route row is E, and the consumer replay row is C.

Use the obligation theorem for the same packet. It makes four-row discharge coverage and beta-boundary routing simultaneous requirements of the local $\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}$ surface.

The rows A, L, P remain necessary discharge rows, but their branch content is application-argument, lambda-domain, and Pi-domain preservation. They do not contain the beta-substitution premise-conclusion shape carried by B. Therefore the only accepted beta-facing route is the displayed path from B through E to C. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier beta boundary route`

Theorem 1008.4 (MetaCICRedexFrontier obstruction non-escape)**.** *No consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier can extract a closed MetaCIC subject-reduction theorem, a hidden substitution oracle, or a three-row subject-reduction interface from the packet. Every consumer read factors through the displayed rows*

$$B, A, L, P, O, E, H, C, G, N.$$

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier obstruction nonescape`

Proof. Read the carrier definition. It exposes the four discharge rows, the obstruction ledger, the beta-boundary route, transport, replay, provenance, and local naming. It lists no closed theorem coordinate and no hidden oracle coordinate.

Apply the NameCert obligations. The obstruction row O keeps the packet tied to the finite discharge route described by chapter 9386, rather than to a closed subject-reduction theorem. The beta-boundary row E also prevents a consumer from reading the beta case without the named route.

The remaining structural rows H, C, G, N preserve the same packet. Since the classifier requires all four discharge rows B, A, L, P, a three-row interface is rejected rather than completed by transport, continuation replay, provenance, or local naming. □

Theorem 1008.5 (MetaCICRedexFrontier scoped package)**.** *Let* $R = (B, A, L, P, O, E, H, C, G, N)$ *be an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier. Its scoped package is exactly the finite primitive surface formed by the typed term window carried by the four discharge rows* B, A, L, P*, the beta-boundary row* E*, the obstruction rows in* O*, componentwise* BHist *and* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* G*, local* $\mathsf{NameCert}$ *naming through* N*, and the TasteGate alignment of the same carrier.*

The scoped package exports no subject-reduction theorem, global confluence theorem, normalization theorem, hidden substitution oracle, three-row discharge interface, or beta-only shortcut.

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierTasteGate single carrier alignment`

Proof. Project the accepted carrier. The typed term window is not a separate ambient context; it is the finite window read by the four displayed discharge rows B, A, L, P.

The beta and obstruction boundaries are explicit. The row E routes beta-redex consumption before any subject-reduction consumer, while O records why the carrier remains a frontier packet rather than a closed theorem.

The structural rows preserve this same surface. The rows H, C, G, N supply transport, replay, provenance, and local naming, and the TasteGate alignment identifies this single carrier as the accepted package. Since no off-packet theorem or oracle coordinate appears in the carrier, the excluded global claims are outside the scoped package. □

Theorem 1008.6 (MetaCICRedexFrontier normalization-window handoff)**.** *Every normalization-window consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier factors through the scoped package of Definition 1008.5 before it may hand data to the normalization-window confluence packet of chapter 2337. The handoff exports the typed term window, beta-boundary row, obstruction rows, transport, replay, provenance, and local naming rows; it does not export subject reduction, global confluence, or normalization.*

Proof. Begin with the obstruction non-escape theorem, Definition 1008.4. Every consumer read factors through the displayed rows $B, A, L, P, O, E, H, C, G, N$.

Apply Definition 1008.5. The same rows form the scoped primitive surface, including the beta-boundary route and obstruction ledger needed before a normalization-window reader can consume the packet.

The target confluence packet is a downstream consumer named by chapter 2337. This chapter hands off only the finite redex-frontier rows; the closed subject-reduction, global confluence, and normalization claims are not coordinates of the carrier. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier normalization window handoff`

Theorem 1008.7 (MetaCICRedexFrontier candidate normalization)**.** *A normalization-window candidate over an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier may be read only when all four discharge rows* B, A, L, P*, obstruction row* O*, beta-boundary route* E*, componentwise transport* H*, displayed replay* C*, provenance* G*, and local naming row* N *are displayed. The candidate route hands a finite redex-frontier packet to the normalization-window consumer, but it does not close subject reduction, global confluence, normalization, substitution admissibility, or any hidden oracle claim.*

Proof. Begin with obstruction non-escape, Definition 1008.4. It forces every consumer to read the four discharge rows, the obstruction row, beta-boundary route, transport, replay, provenance, and local naming before the packet can be exposed.

Apply the scoped package theorem, Definition 1008.5. The same rows form the primitive finite surface; a candidate missing one of B, A, L, P or missing O, E, H, C, G, N is not the scoped packet accepted by the carrier.

Finally use the normalization-window handoff of Definition 1008.6. It allows only the finite redex-frontier rows to pass to the downstream normalization-window consumer.

The handoff is therefore candidate-mediated and finite. Since subject reduction, global confluence, normalization, substitution admissibility, and oracle access are not carrier rows, the candidate route cannot assert them. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCandidateNormalization`

Theorem 1008.8 (MetaCICRedexFrontier candidate SN window)**.** *A strong-normalization-window consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier may read only the four discharge rows* B, A, L, P*, the obstruction row* O*, the beta-boundary row* E*, componentwise transport* H*, displayed replay* C*, provenance* G*, and local naming* N*. It may pass the resulting finite candidate packet to the normalization-window confluence route of chapter 2337, but it receives no subject-reduction theorem, global confluence theorem, normalization theorem, substitution oracle, or dependent-codomain readback.*

Proof. Begin with the scoped package theorem, Definition 1008.5. It identifies the primitive surface of an accepted carrier as exactly the typed term window carried by B, A, L, P, the beta-boundary row E, the obstruction row O, and the structural rows H, C, G, N.

Next apply the normalization-window handoff, Definition 1008.6. This handoff permits those finite redex-frontier rows to be read by the downstream normalization-window confluence packet, while keeping subject reduction, global confluence, and normalization outside the exported surface.

Finally use candidate normalization, Definition 1008.7. It sharpens the handoff to finite candidates and records that substitution admissibility and hidden oracle access are not carrier rows. Since dependent-codomain readback is also absent from the carrier inventory, the strong-normalization-window consumer receives only the displayed finite candidate packet. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCandidateSNWindow`

Theorem 1008.9 (MetaCICRedexFrontier SN obstruction retention)**.** *A strong-normalization-window consumer over an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *packet must retain the obstruction row* O *through the beta-boundary route and the normalization-window handoff. The consumer may pass only the finite candidate packet formed from* $B, A, L, P, O, E, H, C, G, N$*; it may not erase* O *into a global strong-normalization theorem, subject-reduction theorem, confluence theorem, substitution oracle, or dependent-codomain readback.*

Proof. Start with obstruction non-escape, Definition 1008.4. It forces the obstruction row O to remain visible whenever the redex-frontier packet is exposed to a consumer.

Use the scoped package theorem, Definition 1008.5. The accepted packet is exactly the finite row family $B, A, L, P, O, E, H, C, G, N$, so retaining the obstruction is part of reading the primitive package rather than an external side condition.

Apply the normalization-window handoff and candidate normalization, Definition 1008.6 and Definition 1008.7. These results carry the beta-boundary row E and the finite candidate surface to the downstream normalization-window route without replacing O by an ambient normalization claim.

Finally apply the candidate SN-window theorem, Definition 1008.8. It restricts the strong-normalization-window consumer to the displayed finite candidate packet, so subject reduction, confluence, substitution, global strong normalization, and dependent-codomain readback remain outside the exported surface. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierSNObstructionRetention`

Theorem 1008.10 (MetaCICRedexFrontier residual-diamond handoff)**.** *Any residual-diamond consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *packet receives only the finite redex-frontier rows* $B, A, L, P, O, E, H, C, G, N$ *together with the retained obstruction rows. The handoff may compare displayed residual windows inside that packet, but it does not export global confluence, substitution admissibility, strong normalization, or dependent-codomain readback.*

Proof. Start with the strong-normalization obstruction retention theorem, Definition 1008.9. It keeps the obstruction row O visible whenever a consumer reads the beta-boundary route.

Read the candidate SN window of Definition 1008.8. The consumer receives only the finite candidate packet formed from the four discharge rows, the obstruction row, the beta-boundary row, and the structural rows.

A residual-diamond comparison can therefore inspect only displayed residual windows inside the same redex-frontier packet. The comparison may transport those windows through H and replay them through C, but it cannot add substitution admissibility or a host confluence theorem.

Thus the handoff preserves the finite candidate-mediated boundary. Strong normalization, global confluence, substitution, and dependent-codomain readback remain outside the exported NameCert surface. □

Theorem 1008.11 (MetaCICRedexFrontier obstruction retention)**.** *Every candidate-mediated strong-normalization or normalization-window consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier must keep the beta-boundary row E, the four discharge rows B, A, L, P, the obstruction row O, componentwise transport H, displayed replay C, provenance G, and local naming N visible before any handoff to the normalization-window confluence packet of chapter 2337. The retained packet exports only the finite redex-frontier surface; it exports no subject reduction, global confluence, normalization, or substitution admissibility.*

Proof. Project the accepted carrier

$$R = (B, A, L, P, O, E, H, C, G, N).$$

The carrier inventory already displays the four discharge rows, the beta-boundary row, the obstruction row, and the structural transport, replay, provenance, and naming rows.

Apply the candidate SN-window theorem, Definition 1008.8. It permits a strong-normalization-window consumer to pass only the finite candidate packet formed from $B, A, L, P, O, E, H, C, G, N$ to the downstream normalization-window confluence route.

Apply the scoped package theorem, Definition 1008.5. It identifies the same rows as the primitive surface of the accepted packet, so a candidate-mediated consumer that hides O, drops E, or separates H, C, G, N from the discharge rows is not reading the scoped package.

Finally use beta-boundary routing, Definition 1008.3, together with the normalization-window handoff of Definition 1008.6. The beta row is consumed through E before the handoff, and the downstream packet receives only the displayed redex-frontier rows, not the off-carrier global claims. □

Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierObstructionRetention`

Theorem 1008.12 (MetaCICRedexFrontier residual-confluence handoff)**.** *Every residual-confluence consumer of an accepted* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *carrier must first expose the four discharge rows B, A, L, P, the beta-boundary row E, the obstruction row O, componentwise transport H, displayed replay C, provenance G, and local naming N before handing data to any residual or parallel-diamond confluence packet. The handoff exports only the finite redex-frontier surface; it exports no subject reduction, global confluence, normalization, substitution admissibility, hidden oracle, or dependent-codomain readback.*

Proof. Project the accepted carrier

$$R = (B, A, L, P, O, E, H, C, G, N).$$

This displays the four discharge rows, beta-boundary row, obstruction row, and structural rows before any confluence-facing consumer can inspect the packet.

Apply non-escape and the scoped package theorem, Definition 1008.4 and Definition 1008.5. Together they force the consumer to read exactly the displayed primitive package, with O retained and H, C, G, N attached to the discharge rows.

Use the normalization-window handoff and candidate normalization, Definition 1008.6 and Definition 1008.7. These results route the beta-boundary row E and the finite candidate surface to downstream normalization-window consumers without replacing the packet by a global normalization or subject-reduction theorem.

Finally apply obstruction retention, Definition 1008.11. A residual or parallel-diamond packet therefore receives only the finite candidate data with the obstruction still visible, so the excluded confluence, substitution, normalization, oracle, and dependent-codomain claims remain outside the export. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ packet is generated from beta-redex, application-argument, lambda-domain, and Pi-domain discharge rows, an obstruction ledger, a beta-boundary route, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1008.1, Definition 1008.2, Definition 1008.3, Definition 1008.4, Definition 1008.5, and Definition 1008.6 bind the finite beta-redex frontier packet, its primitive scoped package, and its normalization-window handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICRedexFrontier}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.MetaCICRedexFrontierUp.MetaCICRedexFrontierCarrier\_namecert\_obligations`
Lean4 ✓ `BEDC.Derived.MetaCICRedexFrontierUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close MetaCIC subject reduction, confluence, normalization, substitution admissibility, dependent-codomain readback, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: The next theory step is the bridge-facing handoff from the public redex-frontier export to a standard acceptance bridge.

Chapter 1009

A Concrete Naming Certificate for $\mathsf{CofinalRealRequestReuse}^{\uparrow}$

The $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as a finite request-reuse certificate. It consumes the finite request fusion surface of chapter 360, the window naming surface of chapter 126, and the regular-rational and dyadic readback rows used by the completion seal. The packet is not a completion theorem by itself; it is the displayed lock that lets repeated cofinal precision bundles reuse one visible route before a $\mathsf{Real}^{\uparrow}$ seal is served.

Definition 1009.1 (Cofinal real request-reuse carrier)**.** *A* $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, \sigma, W, G, D, S, H, C, P, N)$$

with the following visible rows:

1. *Q is the finite bundle of cofinal Real-completion precision requests;*
2. *σ is the cofinal selector row that chooses the displayed tail service for every request in Q;*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family through which all selected finite observations are read;*
4. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the same shared windows;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the requested tolerances, endpoints, and finite comparison rows;*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after Q, σ, W, G, D have been replayed;*
7. *H records componentwise* hsame *transport for the request bundle, selector, window, readback, dyadic ledger, seal, route, provenance, and local name;*
8. *C records the displayed* Cont *routes that replay the cofinal selector, shared StreamName windows, RegSeqRat readback, DyadicRatCore observations, and RealUp seal handoff;*
9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalRealRequestReuse}^{\uparrow}}$ *row.*

The carrier contains no second request-bundle route, no host completion object, no quotient stream equality, no hidden selector, and no RealUp seal outside the displayed shared route.

Lean4 ✓ BEDC.Derived.CofinalRealRequestReuseUp.CofinalRealRequestReuseTasteGate single carrier alignment

Theorem 1009.2 (Cofinal real request-reuse NameCert obligations)**.** *For the carrier of Definition 1009.1, the local* $\mathsf{NameCert}_{\mathsf{CofinalRealRequestReuse}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (Q, \sigma, W, G, D, S, H, C, P, N)$*, cofinal request-bundle exposure through* Q*, selector exposure through* σ*, shared* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* G*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exactness through* D*,* $\mathsf{Real}^{\uparrow}$ *seal handoff through* S*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier definition. Each obligation row named in the statement is a displayed coordinate of R, so carrier admission fixes the request bundle, selector, shared window, readback, dyadic ledger, seal, transport, replay, provenance, and local name at once.

Use the finite request fusion supplied by the $\mathsf{RealCauchyCompletion}^{\uparrow}$ surface. A cofinal precision bundle may be read request by request, but every accepted read must return to the same displayed selector σ and the same shared StreamName window family W before reaching G, D, S.

The downstream handoff rows then keep the route local. The RegSeqRat readback, DyadicRatCore ledger, RealUp seal, componentwise transports, continuations, package row, and NameCert row are all consumed from the displayed packet, so no unlisted completion or selector coordinate enters the certificate. □

Theorem 1009.3 (Cofinal real request-reuse fused handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ *packet reads the finite request bundle* Q *through the same selector* σ *and the same shared window route*

$$W, G, D, S, H, C, P, N.$$

Repeated precision bundles may therefore reuse the displayed RealCauchyCompletion route, but they cannot split into separate StreamName, RegSeqRat, DyadicRatCore, or RealUp seal routes.

Proof. Begin with the request-bundle row Q and the cofinal selector σ. The selector is a coordinate of the packet, so a request cannot choose a tail service outside the displayed carrier.

Apply the finite request fusion route from chapter 360. That route turns finitely many precision reads into a finite completion-facing surface, and here its window side is identified with the single StreamName family W.

Now replay the downstream handoff. The RegSeqRat row G reads only the windows in W, the DyadicRatCore ledger D records only the displayed observations, and the RealUp seal S is consumed only after those rows pass through H, C, P, N. Hence repeated precision bundles reuse one route and do not branch into separate completion-seal paths. □

Theorem 1009.4 (Cofinal real request-reuse non-escape)**.** *An accepted* $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ *packet exports no host completion carrier, no quotient stream equality, no second cofinal selector, no unshared StreamName window, no RegSeqRat readback detached from the dyadic ledger, and no RealUp seal detached from the displayed packet. Every downstream* $\mathsf{Real}^{\uparrow}$ *or completion-seal consumer factors through* $Q, \sigma, W, G, D, S, H, C, P, N$*.*

Proof. Inspect the carrier rows. The only request surface is Q, the only selector is σ, the only window family is W, and the only Real-facing row is the seal S.

Use the NameCert obligations to force every accepted read through the shared rows W, G, D, S, H, C, P, N. The componentwise transport and continuation rows record how the packet is replayed; they do not add another selector or another completion route.

Finally use the fused handoff theorem. Since every completion-facing consumer must reuse the same displayed route, there is no coordinate from which a host completion carrier, quotient stream equality, unshared window, detached readback, or detached RealUp seal can be projected. □

Falsifiable prediction: A consumer that reads two cofinal precision bundles as reusable without displaying the shared StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal refutes the packet.

Independence witness: Nearest siblings: realcauchycompletion, realobservationbudget, streamname, regseqrat, and dyadicratcore. None is bijective to $\mathsf{CofinalRealRequestReuse}^{\uparrow}$: RealCauchyCompletionUp supplies the completion route but not the repeated request-reuse lock; RealObservationBudgetUp supplies request caps but not the shared completion seal; StreamNameUp supplies windows without RegSeqRat or Dyadic readback; RegSeqRatUp supplies regular readback without the shared request selector; DyadicRatCoreUp supplies observations without the cofinal bundle and RealUp seal. The packet is exactly the lock tying repeated precision bundles to one displayed route.

The later Lean file TasteGate.lean for $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ must provide $\mathsf{Nontrivial\ CofinalRealRequestReuse}^{\uparrow}$ and $\mathsf{FieldFaithful\ CofinalRealRequestReuse}^{\uparrow}$ instances for the same packet surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalRealRequestReuse}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalRealRequestReuse}^{\uparrow}$ packet is generated from a finite cofinal precision request bundle, a cofinal selector, shared StreamName windows, RegSeqRat readback, DyadicRatCore observations, a RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalRealRequestReuse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1009.1, Definition 1009.2, Definition 1009.3, and Definition 1009.4 bind the finite repeated request-reuse packet and its shared completion-seal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalRealRequestReuse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, uniqueness of completed limits, quotient equality of streams, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-bundle fusion, shared-window reuse, RegSeqRat and DyadicRatCore handoff, RealUp seal non-escape, nontriviality, and field faithfulness for the same carrier.

Chapter 1010

A Concrete Naming Certificate for $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$

The $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ packet realizes the unary-direction bridge of chapter 9344 as a finite NameCert carrier. It consumes the axis-unary-spine surface of chapter 599 and the zero-spine standard surface of chapter 572, then adds a local direction decision row that keeps host integer axes outside the exported packet.

Definition 1010.1 (UnaryDirectionNormalizer carrier)**.** *A* $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, Z, K, D, H, C, P, N)$$

with the following visible coordinates:

1. *A is the axis-unary-spine row supplied by* $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$*;*
2. *Z is the zero-spine standard row supplied by* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$*;*
3. *K is the Zeckendorf carry-normalization row used by axis-side carry consumers;*
4. *D is the direction-normalization decision row choosing the unary-to-axis, axis-to-unary, or neutral bridge presentation;*
5. *H stores componentwise* hsame *transports for A, Z, K, D and the local naming row;*
6. *C stores the displayed* Cont *replay routes for axis-spine exposure, zero-spine exposure, carry-normalization routing, direction-decision readback, and NameCert audit;*
7. *P records* Pkg *provenance over* $\mathsf{AxisUnarySpineNormalizer}^{\uparrow}$*,* $\mathsf{UnaryZeroSpineStandardIso}^{\uparrow}$*,* $\mathsf{ZeckendorfCarryNormali}$ $\mathsf{UnaryDirectionBridge}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionNormalizer}^{\uparrow}}$ *row.*

The packet has no coordinate for a host integer-axis function, a quotient between unary and zero-spine carriers, or an ambient standard-model normalizer outside the displayed rows.

Lean4 ✓ `BEDC.Derived.UnaryDirectionNormalizerUp.UnaryDirectionNormalizerTasteGate single carrier alignment`

Theorem 1010.2 (UnaryDirectionNormalizer NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryDirectionNormalizer}^{\uparrow}}$ *surface for Definition 1010.1 consists of carrier admission, axis-row exposure, zero-spine compatibility, carry-normalization routing, direction-decision stability, and non-escape from host integer*

axes. In particular, an accepted packet exposes the finite rows A, Z, K, D, H, C, P, N and exports a direction-normalized unary packet only through those rows.

Proof. Carrier admission is projection from the displayed packet $U = (A, Z, K, D, H, C, P, N)$. The axis-row exposure clause reads A through the sibling surface of chapter 599, and the zero-spine compatibility clause reads Z through chapter 572.

The carry route is separate from the direction decision. The row K supplies the Zeckendorf carry-normalization input, while D records which bridge direction is being normalized for the downstream consumer.

Stability is componentwise. The row H transports A, Z, K, D and the local name without merging the axis-unary and zero-spine carriers, and C replays the same reads as finite continuation routes.

The non-escape condition follows from the absence of any host integer-axis coordinate. Since P lists only the displayed BEDC dependencies and N binds the local NameCert row, a consumer that bypasses A or Z has no route from which to export a direction-normalized unary packet. □

Theorem 1010.3 (UnaryDirectionNormalizer bridge realization)**.** *Every accepted* $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ *carrier realizes the unary-direction bridge as a finite axis, zero-spine, carry, direction, transport, route, provenance, and NameCert packet. Its public reads factor through the bridge source chapter 9344, the axis-unary-spine sibling, and the unary-zero-spine standard sibling, and they do not export a host integer-axis normalizer.*

Proof. Project the carrier rows from Definition 1010.1. The route begins at A and Z, so the bridge read is anchored in the two concrete sibling surfaces rather than in a free host-axis map.

Apply Definition 1010.2. It forces carry-normalization reads through K, direction reads through D, transport through H, continuation replay through C, provenance through P, and the public naming surface through N.

The stated non-escape boundary is therefore immediate: a consumer may inspect the finite bridge packet, but it cannot replace the packet by an external integer-axis normalizer or by a kernel identification of the unary and zero-spine carriers. □

Falsifiable prediction: Any accepted UnaryDirectionNormalizerUp consumer that bypasses the axis-unary-spine row or zero-spine standard row cannot export a direction-normalized unary packet.

Independence witness: axisunaryspinenormalizer, unaryzerospinestandardiso, zeckendorfcarrynormalization, nat, add: none is bijective to UnaryDirectionNormalizerUp because the new carrier contains the combined direction row D and the local NameCert row N tying axis, zero-spine, and carry-normalization routes together.

The Lean taste-gate file TasteGate.lean will provide Nontrivial $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ and FieldFaithful $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryDirectionNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{UnaryDirectionNormalizer}^{\uparrow}$ packet is generated by an axis-unary-spine row, a zero-spine standard row, a Zeckendorf carry-normalization row, a direction decision row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnaryDirectionNormalizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1010.1, Definition 1010.2, and Definition 1010.3 bind the finite unary-direction normalizer packet without host integer-axis escape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnaryDirectionNormalizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean formalization, bridge checking, host integer-axis normalization, kernel equality between unary and zero-spine histories, or a public additive isomorphism.

Upgrade path: Obligation closure requires checked carrier admission, axis-row exposure, zero-spine compatibility, carry-normalization routing, direction-decision stability, and non-escape for the same finite packet.

Chapter 1011

A Concrete Naming Certificate for $\mathsf{ObserverLocalityPullback}^{\uparrow}$

The $\mathsf{ObserverLocalityPullback}^{\uparrow}$ packet realizes the common observer-locality pullback surface in the Inter-Hist Locality chain of chapter 9419. It sits after the observer-locality cell surface of chapter 513, the causal witness surface of chapter 478, the maximum-rate boundary of chapter 594, and the transport packet of chapter 889. Its role is to name the finite packet in which all four rows are simultaneously visible before a rate-facing consumer reads the causal-rate surface of chapter 876.

Definition 1011.1 (Observer-locality pullback carrier)**.** *An* $\mathsf{ObserverLocalityPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (O, D, M, T, J, H, Q, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row supplying the two-observer locality cell;*
2. *D is the displayed finite family of* $\mathsf{CrossHistCausal}^{\uparrow}$ *rows over the same observer-locality cell;*
3. *M is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row read only over the causal family D;*
4. *T is the* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *row transporting O through D before M is consumed;*
5. *J is the pullback compatibility row asserting that O, D, M, T are one finite packet rather than four detached sibling reads;*
6. *H records componentwise* hsame *transport for O, D, M, T, J and the local naming row;*
7. *Q records* psame *stability for the probe reads used by the observer-locality cell, causal rows, maximum-rate row, and transport row;*
8. *C records displayed* Cont *routes for reading O, replaying each row of D, reading M, transporting through T, checking J, and auditing the local name;*
9. *P is the* Pkg *provenance over* $\mathsf{ObserverLocalityPullback}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{ObserverLocalityTransport}^{\uparrow}$, $\mathsf{CrossHistCausalRate}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}$ *row.*

The carrier exports only the displayed locality cell, causal rows, maximum-rate row, transport row, compatibility row, structural rows, provenance, and local naming data.

Theorem 1011.2 (Observer-locality pullback NameCert obligations)**.** *For the carrier of Definition 1011.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$U = (O, D, M, T, J, H, Q, C, P, N),$$

observer-locality source exactness through O*, causal-family exactness through* D*, maximum-rate readback through* M*, transport exactness through* T*, simultaneous pullback compatibility through* J*, componentwise transport through* H*, probe stability through* Q*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 1011.1. The rows $O, D, M, T, J, H, Q, C, P, N$ are the only coordinates available to a local $\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}$ read.

Read the four sibling surfaces in packet order. The observer-locality cell supplies O, the displayed causal family supplies D, the maximum-rate row M is attached to that family, and the transport row T carries the locality-cell read through the causal and rate rows.

The compatibility row J then records that these four surfaces are simultaneous. The structural rows H, Q, C, P, N transport, stabilize, replay, package, and name the same finite packet, so no detached rate row or host frame can replace the pullback carrier. □

Lean4 ✓ `BEDC.Derived.ObserverLocalityPullbackUp.ObserverLocalityPullbackTasteGate single carrier alignment`

Theorem 1011.3 (Observer-locality pullback non-escape)**.** *Every rate-facing consumer of an accepted* $\mathsf{ObserverLocalityPullback}^{\uparrow}$ *packet factors through*

$$O, D, M, T, J, H, Q, C, P, N.$$

In particular, a consumer cannot read the $\mathsf{MaxCausalRate}^{\uparrow}$ *row* M *unless the displayed* $\mathsf{CrossHistCausal}^{\uparrow}$ *family* D*, the* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row* O*, and the* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *row* T *are present in the same pullback packet. The packet exports no global frame, synchronized clock, metric tensor, Lorentz geometry, host real-valued speed, or detached cross-Hist causal-rate row.*

Proof. Start with the accepted carrier. The maximum-rate coordinate is M, and its only admissible sources in the carrier are the displayed locality cell O, causal family D, transport row T, and compatibility row J.

Apply Definition 1011.2. It forces every rate-facing read through the same finite packet and then through H, Q, C, P, N for transport, probe stability, route replay, provenance, and local naming.

The excluded objects have no coordinate in U. Therefore a request for a global frame, synchronized clock, metric tensor, Lorentz geometry, host speed, or detached causal-rate row leaves the accepted pullback packet rather than projecting from it. □

Theorem 1011.4 (Observer-locality pullback causal-rate handoff)**.** *Every* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverLocalityPullback}^{\uparrow}$ *packet receives the observer-locality cell, causal family, maximum-rate row, and transport row only through the common compatibility row* J *and the structural rows* H, Q, C, P, N*. The handoff is therefore a finite pullback packet over the four sibling surfaces, not a separate maximum-rate read followed by an external transport.*

Proof. Project O, D, M, T, J from the carrier. The causal-rate consumer has access to the maximum-rate row only together with the causal family and the transport row that bind it to the observer-locality cell.

Use Definition 1011.3. It excludes any route that reads M as a detached rate value or obtains T from an external observer frame.

The remaining route is exactly the displayed structural surface H, Q, C, P, N. These rows keep the causal-rate handoff inside the accepted BHist packet and leave no coordinate for a host real-valued speed or hidden synchronization row. □

Falsifiable prediction: A rate-facing consumer cannot read MaxCausalRateUp from an observer-locality cell unless the displayed CrossHistCausalUp and ObserverLocalityTransportUp rows form one pullback packet.

Independence witness: observerlocalitycell, crosshistcausal, maxcausalrate, observerlocality-transport, crosshistcausalrate: the pullback packet is not bijective to any sibling because it carries the simultaneous compatibility row among all four surfaces.

The Lean taste-gate file for $\mathsf{ObserverLocalityPullback}^{\uparrow}$ must include explicit $\mathsf{Nontrivial\ ObserverLocalityPullback}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObserverLocalityPullback}^{\uparrow}$ instances preserving the locality-cell row, causal family, maximum-rate row, transport row, pullback compatibility row, structural rows, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverLocalityPullback}^{\uparrow}$ packet is generated from an observer-locality cell, finite cross-Hist causal rows, a maximum-rate row, an observer-locality transport row, a simultaneous compatibility row, componentwise hsame transport, psame stability, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1011.1, Definition 1011.2, Definition 1011.3, and Definition 1011.4 specify the finite pullback packet, its NameCert surface, rate-facing non-escape boundary, and causal-rate handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverLocalityPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive Lorentz geometry, metric tensors, synchronized clocks, global frames, host real-valued speed, Lean verification, or bridge checking for $\mathsf{ObserverLocalityPullback}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, sibling exactness for observer-locality cell, causal, maximum-rate, and transport rows, pullback compatibility, no-global-frame refusal, and TasteGate nontriviality with field faithfulness.

Chapter 1012

A Concrete Naming Certificate for $\mathsf{TermStratumChoiceClosure}^{\uparrow}$

The $\mathsf{TermStratumChoiceClosure}^{\uparrow}$ interface turns the term-level branch of Definition 9387.18 and Definition 9387.19 into a finite naming packet. It separates the accepted witness-map rows that BEDC may consume from the refused choice-oracle rows that would otherwise close the term stratum without displayed witnesses. The packet sits beside chapter 515, chapter 568, chapter 634, and chapter 566: those siblings aggregate the three strata, replacement ledgers, quotient refusal, and proposition transport, while this chapter isolates the term-stratum witness-map route.

Definition 1012.1 (TermStratumChoiceClosure carrier)**.** *A* $\mathsf{TermStratumChoiceClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, F, W, M, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the index-family row for the positive term existentials named in Definition 9387.18;*
2. *F is the fiber-membership row recording the visible target fiber for each accepted index;*
3. *W is the witness-map row, recording for each displayed index the constructor-level term and membership evidence actually supplied to the packet;*
4. *M is the map-coherence row saying that consumer reads of W respect the displayed index and fiber rows rather than an ambient selector;*
5. *R is the oracle-refusal row, recording that no global* Classical.choice *function, hidden selector body, or unindexed host function is admitted;*
6. *H stores componentwise* hsame *transport for the index, fiber, witness-map, coherence, refusal, route, provenance, and naming rows;*
7. *C stores the finite* Cont *routes by which a downstream consumer reads a witness only after reading its displayed index and fiber membership;*
8. *P records* Pkg *provenance over* BHist, hsame, Cont, Pkg, NameCert, *chapter 9387, and the sibling packets chapter 515, chapter 568, chapter 634, and chapter 566;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TermStratumChoiceClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice oracle that manufactures witnesses from positive existentials without the displayed witness-map row W.

Theorem 1012.2 (TermStratumChoiceClosure NameCert obligations)**.** *For an accepted carrier $T = (I, F, W, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TermStratumChoiceClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite packet, index-family exposure for I, fiber-membership exposure for F, witness-map exactness for W and M, componentwise* hsame *and* Cont *stability through H and C, provenance through P, and oracle refusal through R. Every consumer-visible term read factors through these rows before it can use a witness.*

Proof. Project the finite carrier. The rows $I, F, W, M, R, H, C, P, N$ are all displayed, so carrier admission and provenance are ordinary row reads.

Witness-map exactness is the joint reading of W and M. The witness row supplies a term only at a displayed index, and the coherence row ties that term to the displayed fiber membership in F.

Transport and route stability are supplied by H and C. They move and replay the same finite rows componentwise, so a consumer cannot replace the witness-map route by a host selector.

Finally read the refusal row R. It records the rejected oracle surfaces and leaves no coordinate for Classical.choice, a hidden selector body, or an unindexed host function. Thus the listed rows exhaust the local naming obligations. □

Lean4 ✓ `BEDC.Derived.TermStratumChoiceClosureUp.TermStratumChoiceClosureTasteGate single carrier alignment`

Theorem 1012.3 (TermStratumChoiceClosure nontriviality)**.** *The* $\mathsf{TermStratumChoiceClosure}^{\uparrow}$ *packet is nontrivial: there are accepted packets whose witness-map row W is present and accepted packets whose oracle-refusal row R blocks the same consumer request after W is absent. These packets are distinguished by the local* NameCert *surface because one exposes an indexed constructor-level witness route and the other exposes only the refused oracle row.*

Proof. Use a singleton index-family row for I and a displayed fiber row F. In the positive packet, let W store the visible constructor-level witness and its membership evidence, and let M record that the only consumer read is the singleton indexed route.

For the contrasting packet, keep the same index and fiber rows but remove the witness-map route from the accepted surface. The only remaining answer to the consumer request is the refusal row R.

The two packets differ at the row inspected by the consumer: one projects W through C, while the other projects R. Since the local name row records that distinction, the carrier does not collapse to a single vacuous packet. □

Theorem 1012.4 (TermStratumChoiceClosure field faithfulness)**.** *If two accepted* $\mathsf{TermStratumChoiceClosure}^{\uparrow}$ *carriers*

$$T = (I, F, W, M, R, H, C, P, N) \qquad \textit{and} \qquad T' = (I', F', W', M', R', H', C', P', N')$$

are identified by the packet classifier, then each displayed field is preserved componentwise: index rows compare with index rows, fiber rows with fiber rows, witness-map rows with witness-map rows, map-coherence rows with map-coherence rows, oracle-refusal rows with oracle-refusal rows, and the transport, route, provenance, and naming rows with their corresponding rows.

Proof. Unfold the classifier of the accepted packet as componentwise comparison over the displayed carrier coordinates. The index and fiber coordinates are compared first, so a witness in W cannot be transported to a different index or fiber without the matching hsame evidence.

Next compare W, M, R. The witness-map route is preserved as a witness-map route, the coherence row remains a coherence row, and the oracle refusal remains a refusal row. No classifier clause converts a refused oracle into a witness map.

The remaining rows H, C, P, N are administrative coordinates for transport, route replay, provenance, and local naming. Componentwise preservation of these rows gives field faithfulness for the whole packet. □

Falsifiable prediction: Any accepted packet whose witness-map row is removed has no route from indexed positive existentials to a consumer-visible term without invoking the refused oracle row.

Independence witness: unwitnessedexistenceclosure, effectivereplacementledger, quotientsoundnessboundary, propexttransportboundary: none is bijective because this packet isolates the term-stratum witness-map route, while the siblings aggregate strata, replacement ledgers, quotient transport, or proposition transport.

The Lean `TasteGate.lean` surface must provide `Nontrivial TermStratumChoiceClosureUp` and `FieldFaithful TermStratumChoiceClosureUp` instances for the same displayed carrier rows $I, F, W, M, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{TermStratumChoiceClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{TermStratumChoiceClosure}^{\uparrow}$ packet is generated from an index-family row, fiber-membership row, displayed witness-map row, map-coherence row, oracle-refusal row, componentwise hsame transport, finite Cont routes, provenance, and local naming. The carrier reads positive term existentials only after their index and fiber rows are visible; then the witness-map supplies constructor-level evidence. If that row is absent, the refusal row records that no host selector is admitted. Thus the classifier separates generated term witnesses from refused choice-like supply using finite BEDC rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TermStratumChoiceClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the term-stratum choice-closure carrier, NameCert obligations, nontriviality, and field-faithfulness statements.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TermStratumChoiceClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a global choice function, hidden selector body, unindexed host function, or witness production without displayed fiber evidence.

Upgrade path: To reach obligationClosure, provide checked nontrivial and field-faithful instances for the finite witness-map and oracle-refusal rows.

Chapter 1013

A Concrete Naming Certificate for InterHistSymmetryBudget$^{\uparrow}$

The InterHistSymmetryBudget$^{\uparrow}$ packet realizes the two-direction locality budget from chapter 9419 as finite NameCert data. It consumes the ledger-local route of chapter 759, the two-observer cell of chapter 513, the invariant readback of chapter 656, the observer-Hist identity surface of chapter 479, and the no-global-synchronization boundary of chapter 1003. Its role is to budget both directional reads before any downstream consumer treats an inter-Hist relation as symmetric.

Falsifiable prediction: Every accepted InterHistSymmetryBudgetUp packet exposes both directional locality rows and admits no consumer route to a global synchronization history or host frame.

Independence witness: Nearest siblings: interhistlocalityledger, observerlocalitycell, interhistinvariant, observerhistidentity, and noglobalsyncboundary. None is bijective to this carrier because InterHistSymmetryBudget$^{\uparrow}$ stores both directional locality rows, one invariant readback row, one no-global-synchronization refusal row, transport, continuation, provenance, and local naming data in a single budget packet, while each sibling omits at least one of those rows or records a different consumer surface.

Definition 1013.1 (InterHistSymmetryBudget carrier)**.** *An* InterHistSymmetryBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (A, B, L_{AB}, L_{BA}, O, I, G, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source history rows whose symmetric locality budget is being certified;*
2. *L_{AB} is the directional locality row reading finite data from A toward B;*
3. *L_{BA} is the directional locality row reading finite data from B toward A;*
4. *O is the* ObserverLocalityCell$^{\uparrow}$ *row consumed before the two directional reads are compared;*
5. *I is the* InterHistInvariant$^{\uparrow}$ *readback row that receives the two-direction locality budget;*
6. *G is the* NoGlobalSyncBoundary$^{\uparrow}$ *refusal row, recording that neither direction supplies a global synchronization history, shared clock, privileged host frame, or selected all-observer map;*
7. *H stores componentwise* hsame *transport for the source histories, directional locality rows, observer-locality cell, invariant readback, no-global-synchronization row, route rows, provenance, and local name;*

8. *C stores the displayed* Cont *routes by which consumers read $A, B, L_{AB}, L_{BA}, O, I, G, H, P, N$;*
9. *P is the* Pkg *provenance over* $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InterHistSymmetryBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global synchronization history, host frame, shared clock, total observer order, quotient of histories, metric spacetime, or ambient equality oracle.

Theorem 1013.2 (InterHistSymmetryBudget NameCert obligations)**.** *For the carrier of Definition 1013.1, the local* $\mathsf{NameCert}_{\mathsf{InterHistSymmetryBudget}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for $Y = (A, B, L_{AB}, L_{BA}, O, I, G, H, C, P, N)$;*
2. *two-direction locality exposure, requiring both L_{AB} and L_{BA} after the observer-Hist rows A, B have been read;*
3. *observer-locality and invariant readback, requiring O before I receives the paired directional rows;*
4. *componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming through H, C, P, N;*
5. *non-escape through the no-global-synchronization row G, excluding global synchronization histories, host frames, shared clocks, total observer orders, quotient histories, metric spacetime, and ambient equality oracles.*

Proof. Carrier admission is projection from Definition 1013.1. The rows $A, B, L_{AB}, L_{BA}, O, I, G, H, C, P, N$ display the two source histories, the two directional locality reads, the locality cell, invariant readback, no-global-synchronization refusal, transport, routes, provenance, and local name.

The two-direction locality obligation reads L_{AB} and L_{BA} separately. The observer-Hist identity surface supplies the two source histories, while the inter-Hist locality ledger supplies the directional route discipline for each read.

The observer-locality and invariant rows then receive the paired budget. The row O records the finite two-observer cell, and the row I records the observer-symmetric invariant readback after both directions have been displayed.

Finally H, C, P, N provide transport, continuation replay, provenance, and local naming. The row G supplies the no-global-synchronization refusal, so no remaining obligation can export a global synchronization history, host frame, shared clock, quotient history, metric spacetime, or ambient equality oracle. □

Theorem 1013.3 (InterHistSymmetryBudget non-escape)**.** *Every public consumer of an accepted* $\mathsf{InterHistSymmetryBudget}^{\uparrow}$ *carrier reads only the rows*

$$A, B, L_{AB}, L_{BA}, O, I, G, H, C, P, N.$$

It may consume both directional locality rows, invariant readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It receives no global synchronization history, host frame, shared clock, total observer order, quotient of histories, metric spacetime, or ambient equality oracle.*

Proof. Inspect the carrier coordinates of Definition 1013.1. Every public row is one of $A, B, L_{AB}, L_{BA}, O, I, G, H, C,$

The positive route data remain finite. The two source histories, the two directional locality rows, the observer-locality cell, and the invariant readback are transported by H and replayed by C before provenance and local naming are read through P and N.

The no-global-synchronization row G is a refusal row, not a hidden synchronization object. Since no coordinate supplies a global history, host frame, shared clock, total observer order, quotient history, metric spacetime, or ambient equality oracle, no consumer can export such an object from the accepted packet. □

Theorem 1013.4 (InterHistSymmetryBudget TasteGate obligation)**.** *The later Lean site for* InterHistSymmetryBudget$^{\uparrow}$ *must provide explicit* Nontrivial InterHistSymmetryBudget$^{\uparrow}$ *and* FieldFaithful InterHistSymm *instances. These instances must keep the two source histories, two directional locality rows, observer-locality cell, invariant readback, no-global-synchronization refusal row,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *row as distinguishable fields.*

Lean4 ✓ `BEDC.Derived.InterHistSymmetryBudgetUp.InterHistSymmetryBudgetTasteGate single carrier alignment`

Proof. The carrier separates the two directions L_{AB} and L_{BA} from the invariant readback I and the refusal row G. A trivial collapse would identify a directional locality read with the symmetric budget or with the no-global-synchronization boundary, erasing the consumer distinction this packet exposes.

Field faithfulness is the projection discipline for $Y = (A, B, L_{AB}, L_{BA}, O, I, G, H, C, P, N)$. Each row has a distinct consumer role, so a later TasteGate witness must preserve those projections before any downstream causal-rate or no-global-sync chapter consumes the budget. □

Closure status for NameCert$_{\text{InterHistSymmetryBudget}^{\uparrow}}$.

Constructive story: An InterHistSymmetryBudget$^{\uparrow}$ packet is generated from two source histories, two directional locality rows, an observer-locality cell, an invariant readback row, a no-global-synchronization refusal row, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InterHistSymmetryBudget}^{\uparrow}}$, seed)

Scope closed: Definition 1013.1, Definition 1013.2, and Definition 1013.3 specify the finite two-direction locality budget and its no-global-synchronization boundary.

Formal verification: FormalStatus(NameCert$_{\text{InterHistSymmetryBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct a global synchronization history, host frame, shared clock, total observer order, quotient history, metric spacetime, Lean proof, or bridge checking for InterHistSymmetryBudget$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, two-direction locality exposure, invariant readback, no-global-synchronization non-escape, Nontrivial, and FieldFaithful for the same packet.

Chapter 1014

A Concrete Naming Certificate for $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$

The $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$ packet realizes the witness-ledger direction of the Real and Completion target in chapter 9238. It consumes the sealed Cauchy-limit handoff of chapter 386, the paired tail-agreement route of chapter 699, the Cauchy witness ledger row, the finite $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the $\mathsf{Real}^{\uparrow}$ seal row before a completion-facing consumer may inspect the displayed limit route. The packet is a finite ledger for the route, not a selected limit, quotient real equality, hidden tail selector, or ambient completeness principle.

Definition 1014.1 (Cauchy limit witness-ledger carrier)**.** *A* $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, T, S, W, G, R, H, C, P, N, A)$$

with the following displayed rows:

1. *D is the finite diagonal-budget row that fixes the request served by the route;*
2. *T is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row that synchronizes the visible tail route;*
3. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row consumed as the displayed limit-seal handoff;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before any tail readback is exposed;*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row obtained from W;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, G, T, S have been displayed;*
7. *H records componentwise* hsame *transport for the diagonal budget, tail agreement, limit seal, window, readback, Real seal, provenance, and local name;*
8. *C records the displayed* Cont *route from the budget row to the window, from the window to the readback, from the readback to the tail-agreement row, and from the tail-agreement row to the limit seal and Real seal;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitWitnessLedger}^{\uparrow}}$ *row;*
11. *A is the displayed audit row stating that every accepted read remains on the same finite limit-witness ledger.*

The carrier has no coordinate for a selected limit, quotient real equality, host real equality, hidden tail selector, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyLimitWitnessLedgerUp.CauchyLimitWitnessLedgerTasteGate single carrier alignment`

Theorem 1014.2 (CauchyLimitWitnessLedger NameCert obligations)**.** *For the carrier of Definition 1014.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitWitnessLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L = (D, T, S, W, G, R, H, C, P, N, A)$*, diagonal-budget exposure, tail-agreement exposure, limit-seal exposure, StreamName window service, RegSeqRat readback service, RealUp seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and audit non-escape through* A*.*

Proof. Project the displayed carrier. The packet exposes exactly the diagonal budget, tail agreement, limit seal, window, regular readback, Real seal, transport, route, provenance, local name, and audit rows.

The sibling rows supply the route discipline. The window row is read through chapter 126, the regular-rational readback through chapter 301, the tail agreement through chapter 699, and the limit-seal handoff through chapter 386.

Transport and replay are componentwise. Since no carrier coordinate names a selected limit, quotient real equality, hidden tail selector, host real equality, or ambient completeness principle, the listed rows exhaust the local certificate surface. □

Theorem 1014.3 (CauchyLimitWitnessLedger tail synchronization)**.** *Every accepted* $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$ *packet routes its completion-facing read through the same finite sequence*

$$D, W, G, T, S, R, H, C, P, N, A.$$

In particular, the displayed limit-seal row S *is consumed only after the StreamName window* W*, the RegSeqRat readback* G*, and the tail-agreement row* T *have been exposed through the same* hsame *and* Cont *ledger.*

Proof. Begin with carrier admission. The only visible path to the limit-seal row is the displayed tuple containing $D, W, G, T, S, R, H, C, P, N, A$.

Use the NameCert obligations. They place window service before readback service, readback before tail agreement, and tail agreement before the limit-seal and RealUp seal rows.

The audit row A records that this is the whole accepted route. Thus a consumer cannot bypass the window or readback rows, cannot replace the tail-agreement row by a hidden selector, and cannot obtain a selected limit or quotient real equality from the packet. □

Falsifiable prediction: Every accepted CauchyLimitWitnessLedgerUp packet exposes a displayed limit-seal row and admits no public route to a selected limit, quotient real equality, or hidden tail selector.

Independence witness: Nearest siblings: cauchylimitseal, realtailagreementseal, cauchywitnessledger, streamname, and regseqrat. None is bijective to $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$ because this chapter keeps the combined limit-witness ledger as one finite packet: diagonal budget, tail agreement, limit seal, StreamName window, RegSeqRat readback, RealUp seal, transport, replay, provenance, local name, and audit row are carried together.

The Lean TasteGate boundary for this packet provides explicit `Nontrivial CauchyLimitWitnessLedgerUp` and `FieldFaithful CauchyLimitWitnessLedgerUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitWitnessLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLimitWitnessLedger}^{\uparrow}$ packet is generated from a finite diagonal budget, displayed tail-agreement row, displayed Cauchy-limit seal, StreamName window, RegSeqRat readback, RealUp seal, componentwise transports, displayed replay, provenance, local NameCert row, and audit non-escape row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitWitnessLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1014.1, Definition 1014.2, and Definition 1014.3 bind the finite limit-witness ledger route over StreamName, RegSeqRat, RealUp, RealTailAgreementSeal, CauchyLimitSeal, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitWitnessLedger}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyLimitWitnessLedgerUp.CauchyLimitWitnessLedgerTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close selected limits, quotient real equality, host real equality, hidden tail selectors, ambient completeness, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-agreement ordering, limit-seal exposure, StreamName and RegSeqRat service rows, RealUp seal ordering, and audit non-escape for the same finite packet.

Chapter 1015

A Concrete Naming Certificate for $\mathsf{OracleResponseLedger}^{\uparrow}$

The $\mathsf{OracleResponseLedger}^{\uparrow}$ packet realizes the oracle-response row of Definition 9388.7 as a finite $\mathsf{NameCert}$ surface. It sits between the oracle-call transcript of chapter 951 and the external-channel boundary of chapter 956. Its distinguishing datum is a response ledger: a query can be paired with a finite reply and boundary row without becoming a closed-substrate observation.

Falsifiable prediction: No accepted response ledger can certify a closed-substrate observation unless the same packet also displays the boundary row separating the response from substrate-internal orbit data.

Independence witness: closureuniversalityquadrant, oracleaugmentedsubstrate, oraclesubstrate-boundary, universalclosedsubstrate, presburger-subuniversal; the response ledger carries a query/response row that the quadrant map and closed-substrate witnesses lack, while boundary chapters do not classify response equality.

Definition 1015.1 (Oracle-response ledger carrier)**.** *An* $\mathsf{OracleResponseLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, R, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the substrate row whose oracle-facing request is being inspected;*
2. *Q is the finite oracle-query row, recording the query name, request site, and local input slice;*
3. *R is the response row, recording the finite reply token, response name, and point of consumption;*
4. *B is the boundary row, separating the response from closed substrate-internal orbit observations;*
5. *F is the refusal row, excluding closed-substrate certification, quotient-response equality, hidden oracle correctness, and host-semantic conversion of the response;*
6. *E is the route-exactness row, recording that any public response read must pass through Q, R, B, F before it reaches a quadrant or boundary consumer;*
7. *H records componentwise* hsame *transports for S, Q, R, B, F, E, P, N;*
8. *C records displayed* Cont *routes through the query, response, boundary, refusal, and response-ledger reads;*

9. *P is the* Pkg *provenance over* $\mathsf{OracleResponseLedger}^{\uparrow}$, $\mathsf{OracleAugmentedSubstrate}^{\uparrow}$, $\mathsf{OracleSubstrateBoundary}^{\uparrow}$, $\mathsf{ClosureUniversalityQuadrant}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OracleResponseLedger}^{\uparrow}}$ *row.*

The local classifier compares accepted ledgers only through these rows. It has no coordinate for treating the response as a closed orbit record, no hidden oracle-correctness proof, and no quotient equality of response behaviours.

Lean4 ✓ `BEDC.Derived.OracleResponseLedgerUp.OracleResponseLedgerTasteGate single carrier alignment`

Theorem 1015.2 (Oracle-response ledger NameCert obligations)**.** *For the carrier of Definition 1015.1, the local* $\mathsf{NameCert}_{\mathsf{OracleResponseLedger}^{\uparrow}}$ *surface consists exactly of seven finite obligation rows:*

1. *carrier exposure for $L = (S, Q, R, B, F, E, H, C, P, N)$;*
2. *query accountability through the displayed oracle-query row Q;*
3. *response accountability through the displayed reply row R;*
4. *boundary accountability through B, separating the response from closed-substrate observations;*
5. *refusal accountability through F, excluding closed replay, hidden oracle correctness, quotient response equality, and host semantic conversion;*
6. *route exactness through E, C, requiring public response consumers to read the query, response, boundary, and refusal rows in the same packet;*
7. *componentwise stability, provenance, and local naming through H, P, N.*

Thus an oracle response is accepted only as a finite BEDC ledger over query, response, boundary, refusal, route, provenance, and naming rows.

Proof. Carrier exposure is projection from Definition 1015.1. The accepted packet displays the substrate row, query row, response row, boundary row, refusal row, route-exactness row, transports, routes, provenance, and local name as its only public coordinates.

Query and response accountability are separate finite reads. The row Q records the request site and local input slice, while R records the reply token and point of consumption. Neither row reconstructs the other from the substrate row alone.

Boundary and refusal accountability keep the response outside the closed observation channel. The boundary chapter supplies the external-channel separation, and the oracle-augmented substrate chapter supplies the call and transcript surface. The refusal row records that closed replay, hidden oracle correctness, quotient response equality, and host semantic conversion are not available inside the ledger.

Finally E, C, H, P, N route, transport, package, and name the same finite packet. Any public response consumer must therefore pass through the displayed query, response, boundary, and refusal rows before using the ledger, and no unlisted closed-substrate or oracle-correctness coordinate remains. □

Theorem 1015.3 (Oracle-response ledger boundary non-escape)**.** *Every public response read exported by an accepted* $\mathsf{OracleResponseLedger}^{\uparrow}$ *packet factors through the finite route*

$$S, Q, R, B, F, E, H, C, P, N.$$

The route can certify a finite query/response ledger at an oracle boundary, but it exports no closed-substrate observation for the response, no hidden oracle-correctness theorem, no quotient equality of response streams, no host semantic equality, and no closed universal substrate witness.

Proof. Begin with the carrier projection. The only response-facing coordinates are the substrate row S, query row Q, response row R, boundary row B, refusal row F, route-exactness row E, transports H, continuations C, provenance P, and local name N.

The boundary row must be displayed before a response can be read as an oracle response. By chapter 956, the oracle-facing edge is separated from closed substrate observations, and by chapter 951 the call and transcript remain explicit oracle rows rather than substrate-internal orbit data.

The refusal row then blocks the remaining escape modes. A closed universal substrate witness, quotient response equality, hidden oracle-correctness theorem, or host semantic equality would require a coordinate not present in L.

Thus the public response ledger is exactly the displayed finite route. It may serve quadrant and boundary consumers as oracle-response evidence, but only as finite BEDC data with the boundary and refusal rows still visible. □

The later Lean `TasteGate.lean` development for $\mathsf{OracleResponseLedger}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ OracleResponseLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ OracleResponseLedger}^{\uparrow}$ instances for the displayed carrier. These instances must preserve the query row, response row, boundary row, refusal row, route-exactness row, transport rows, continuation routes, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{OracleResponseLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{OracleResponseLedger}^{\uparrow}$ packet is generated from a substrate row, oracle-query row, response row, boundary row, refusal row, route-exactness row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OracleResponseLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1015.1, Definition 1015.2, and Definition 1015.3 specify the finite oracle-response ledger certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OracleResponseLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, oracle correctness, closed replay of oracle responses, quotient response equality, host semantic equality, or physical realizability of oracle channels.

Upgrade path: Obligation closure requires checked carrier admission, query accountability, response accountability, boundary accountability, refusal exactness, route exactness, nontriviality, and field faithfulness for the same response ledger.

Chapter 1016

A Concrete Naming Certificate for $\mathsf{FinitePrefixAutomaton}^{\uparrow}$

The $\mathsf{FinitePrefixAutomaton}^{\uparrow}$ interface names the standard finite automaton object at the level of bounded prefix runs. It is the finite-run sibling of chapter 351, the bounded continuation surface of chapter 482, and the grammar-facing comparison boundary of chapter 356. It realizes the operational reading of chapter 9365 without importing an external language quotient, a host transition evaluator, or an infinite trace semantics.

Definition 1016.1 (FinitePrefixAutomaton carrier)**.** *A* $\mathsf{FinitePrefixAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, q_0, A, T, W, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the finite state-row ledger;*
2. *q_0 is the initial state row;*
3. *A is the accepting-state subledger of Q;*
4. *T is the finite transition table whose rows have source state, input mark, target state, and displayed* Cont *witness;*
5. *W is the finite prefix word ledger, written as a finite list of* BMark *input rows;*
6. *R is the prefix-run ledger, listing the state row reached after each visible prefix of W;*
7. *E is the endpoint and acceptance row, recording the final state read and its accepting-state membership result;*
8. *H stores componentwise* hsame *transports for states, marks, transition rows, prefix positions, endpoint rows, acceptance rows, provenance, and local naming;*
9. *C stores the displayed* Cont *routes by which consumers replay each prefix transition and inspect the endpoint;*
10. *P is the* Pkg *provenance over* BHist, BMark, hsame, Cont, *prefix lists, acceptance rows, and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixAutomaton}^{\uparrow}}$ *row.*

The carrier exposes one bounded finite-prefix run. It contains no quotient of accepted languages, no host automaton semantics, no unlisted transition rule, and no infinite continuation trace.

Lean4 (definition) `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonUp`

Definition 1016.2 (FinitePrefixAutomaton classifier)**.** *The* $\mathsf{FinitePrefixAutomaton}^{\uparrow}$ *classifier accepts a packet* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$ *exactly when the prefix ledger* R *starts at* q_0*, every successive prefix step is witnessed by a displayed row of* T *and a corresponding* Cont *route in* C*, the endpoint row in* E *is the last row of* R*, and the acceptance bit in* E *is read from membership in* A*. The classifier reads only the displayed state, mark, transition, run, endpoint, transport, continuation, package, and local naming rows.*

Lean4 (definition) `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonClassifier`

Theorem 1016.3 (FinitePrefixAutomaton NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FinitePrefixAutomaton}^{\uparrow}}$ *surface for the carrier of Definition 1016.1 consists of seven finite obligations:*

1. *carrier admission for the displayed packet* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$*;*
2. *prefix-source binding of the word ledger* W *and initial state row* q_0*;*
3. *transition exactness for every adjacent prefix-run row through the displayed table* T *and the corresponding* Cont *witness;*
4. *endpoint exactness, stating that* E *reads the last row of* R*;*
5. *acceptance exactness, stating that the acceptance bit in* E *is precisely the displayed membership read in* A*;*
6. *componentwise stability through* H *and replay through* C*;*
7. *non-escape, stating that every downstream finite-prefix read factors through* $Q, q_0, A, T, W, R, E, H, C, P, N$ *and cannot consult a host evaluator, language quotient, unlisted transition rule, or infinite trace.*

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonNameCertObligationSurface`

Proof. Carrier admission is projection from Definition 1016.1. The packet already displays its state ledger, initial state, accepting ledger, transition table, prefix word, run ledger, endpoint row, transports, continuation routes, provenance, and local name.

The prefix-source, transition, endpoint, and acceptance obligations are read from W, R, T, E, A. Each step of R is licensed by a row of T carrying the displayed Cont witness, and the final read is the endpoint row named in E.

Stability and replay are supplied by H and C. Since the packet has no coordinate for an external evaluator, quotient language, unlisted transition rule, or infinite trace, the non-escape obligation follows from the carrier boundary. □

Theorem 1016.4 (FinitePrefixAutomaton transition exactness obligation)**.** *For an accepted* $\mathsf{FinitePrefixAutomaton}$ *carrier* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$*, each adjacent prefix-run step in* R *is witnessed by a displayed row of* T *whose source state, input mark, target state, and* Cont *route are the rows read by the corresponding prefix window of* W*.*

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier transition exactness obligation`

Proof. Project the accepted carrier through Definition 1016.1. The transition table T, prefix word W, run ledger R, and continuation ledger C are separate displayed coordinates.

Apply the transition row of Definition 1016.3. It requires every adjacent run step to read a row of T and the matching Cont route in C, so no host transition evaluator or unlisted rule can witness the step. □

Theorem 1016.5 (FinitePrefixAutomaton endpoint exactness obligation)**.** *For an accepted* FinitePrefixAutomaton$^{\uparrow}$ *carrier* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$*, the endpoint row* E *is exactly the last displayed row of the finite prefix-run ledger* R*.*

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier endpoint exactness obligation`

Proof. Read the endpoint coordinate of the carrier. The row E is displayed independently of the transition table and cannot be supplied by an ambient evaluator.

Use the endpoint obligation in Definition 1016.3. That obligation identifies E with the last row of R, while H only transports the displayed endpoint row and C only replays the displayed route. □

Theorem 1016.6 (FinitePrefixAutomaton acceptance exactness obligation)**.** *For an accepted* FinitePrefixAutomaton$^{\uparrow}$ *carrier* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$*, the acceptance bit recorded in* E *is precisely the displayed membership read of the endpoint state in the accepting-state sub-ledger* A*.*

Proof. Project the accepting ledger A and endpoint row E from the carrier. These rows are part of the finite packet, so acceptance is not read from a quotient language or external automaton semantics.

Apply the acceptance obligation in Definition 1016.3. It binds the bit stored in E to membership in A for the endpoint state, and the remaining transport, replay, provenance, and naming rows preserve that displayed read. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier acceptance exactness obligation`

Theorem 1016.7 (FinitePrefixAutomaton displayed prefix-run determinism)**.** *Let*

$$F = (Q, q_0, A, T, W, R, E, H, C, P, N)$$

and

$$F' = (Q, q_0, A, T, W, R', E', H', C', P', N')$$

be accepted FinitePrefixAutomaton$^{\uparrow}$ *carriers with the same state ledger* Q*, initial row* q_0*, accepting ledger* A*, transition table* T*, and word ledger* W*. If the displayed* Cont *replay rows in* C *and* C' *match componentwise along the prefix windows of* W*, then the prefix-run ledgers* R *and* R' *are* hsame*-equivalent at every displayed prefix position, and the endpoint rows* E *and* E' *name the same final state row.*

Proof. The empty prefix is forced by the shared initial row q_0. Both accepted carriers therefore begin their displayed run ledger at the same state row, with the componentwise transport rows recording the allowed hsame read.

For the successor prefix step, project transition exactness from Definition 1016.4. The same table T, the same input mark from W, and the matching replay rows in C and C' force both run

ledgers to use the same displayed transition row. Induction over the finite prefix ledger propagates the hsame comparison through every prefix position.

Finally apply endpoint exactness from Definition 1016.5. Since each endpoint is the last displayed row of its run ledger, the prefixwise comparison identifies E and E' as the same final-state read inside the finite packet boundary. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier prefix run displayed determinism`

Theorem 1016.8 (FinitePrefixAutomaton prefix-run nonescape)**.** *Any downstream* RegularLanguage$^{\uparrow}$ *or* ContinuationAutomaton$^{\uparrow}$ *read of an accepted* FinitePrefixAutomaton$^{\uparrow}$ *prefix run consults only the rows*

$$Q, q_0, A, T, W, R, E, H, C, P, N.$$

It cannot use a host evaluator, a language quotient, an unlisted transition, or an infinite trace to justify a transition step, endpoint row, or acceptance bit.

Proof. Start with the NameCert obligation surface of Definition 1016.3. It lists carrier admission, prefix-source binding, transition exactness, endpoint exactness, acceptance exactness, stability, replay, and non-escape as finite obligations over the displayed packet rows.

The transition, endpoint, and acceptance exactness theorems then identify the only admissible reads of a run step, final row, and acceptance bit. Each read is projected from $Q, q_0, A, T, W, R, E, H, C, P, N$, with H transporting the displayed rows and C replaying only the displayed continuation witnesses.

A RegularLanguage$^{\uparrow}$ consumer may read the finite word, state, endpoint, and acceptance rows; a ContinuationAutomaton$^{\uparrow}$ consumer may read the Cont-labelled transition rows and replay routes. Neither consumer receives a coordinate for a host evaluator, quotient language, unlisted transition rule, or infinite trace, so those objects cannot enter the prefix-run justification. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier prefix run nonescape`

Theorem 1016.9 (FinitePrefixAutomaton prefix-run determinacy)**.** *For an accepted* FinitePrefixAutomaton$^{\uparrow}$ *carrier* $F = (Q, q_0, A, T, W, R, E, H, C, P, N)$*, the displayed prefix word* W *and transition table* T *determine the prefix-run ledger* R *componentwise inside the packet. If another accepted run ledger* R' *is admitted over the same* $Q, q_0, A, T, W, E, H, C, P, N$*, then each prefix position of* R *is* hsame *to the corresponding prefix position of* R' *through the displayed transport rows in* H*.*

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonCarrier prefix run determinacy`

Proof. Begin at the empty prefix. Both run ledgers must start at the displayed initial state row q_0 by the prefix-source obligation of Definition 1016.3.

For a nonempty prefix, read the preceding state row and the next input mark from W. The transition exactness obligation forces the next state to be the target of the displayed row of T carrying the matching Cont witness. Thus the two ledgers have the same displayed source, mark, transition row, and target at that position.

Apply the componentwise transport rows in H at each prefix position. They identify the corresponding state rows inside the packet, while C only replays the already displayed transition route. No host transition function or unlisted rule is available, so the determinacy claim stays internal to the finite ledger. □

Theorem 1016.10 (FinitePrefixAutomaton regular-language sibling bridge)**.** *Every accepted* FinitePrefixAutomato *packet supplies exactly the bounded finite-run surface consumed by* RegularLanguage$^{\uparrow}$ *and* ContinuationAutomaton$^{\uparrow}$ *readers: the* RegularLanguage$^{\uparrow}$ *side reads the same finite state, prefix word, endpoint, and acceptance rows, while the* ContinuationAutomaton$^{\uparrow}$ *side reads the same* Cont*-labelled transition rows, componentwise* hsame *transports, replay routes, package row, and local* NameCert *row. The bridge exposes no quotient language, no context-free derivation tree, no host evaluator, and no infinite continuation trace.*

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomaton regularlanguage sibling bridge`

Proof. Read the finite automaton side through chapter 351. The accepted prefix word, finite state rows, endpoint row, and acceptance ledger are already displayed by Q, q_0, A, W, R, E.

Read the continuation side through chapter 482. Each transition row in T carries its displayed Cont witness, while H, C, P, N provide transport, replay, provenance, and local naming.

The context-free sibling chapter 356 may compare this bounded run with grammar yields only through the exposed finite word and endpoint surface. Since the packet has no coordinate for language quotienting, derivation trees, host evaluation, or infinite traces, the sibling bridge is exactly the finite-prefix route listed above. □

Theorem 1016.11 (FinitePrefixAutomaton obligation window)**.** *Every accepted* FinitePrefixAutomaton$^{\uparrow}$ *prefix-window read exposes exactly the finite rows*

$$Q, q_0, A, T, W, R, E, H, C, P, N.$$

The window supplies the state ledger, initial row, accepting-state ledger, transition table, finite word, run ledger, endpoint and acceptance row, componentwise transport, displayed replay, provenance, and local NameCert *row, and it supplies no host evaluator, quotient language equality, unlisted transition rule, or infinite trace.*

Proof. Project the accepted carrier through Definition 1016.1. The carrier displays precisely $Q, q_0, A, T, W, R, E, H, C, P, N$ and has no additional coordinate through which a consumer could read outside automaton semantics.

Apply the classifier and obligation surface of Definition 1016.2 and Definition 1016.3. They bind prefix-source, transition, endpoint, acceptance, stability, replay, provenance, and local naming obligations to the same displayed rows.

Finally use transition exactness, endpoint exactness, acceptance exactness, and the sibling bridge, Definition 1016.4, Definition 1016.5, Definition 1016.6, and Definition 1016.10. Those theorems keep every downstream finite-prefix read inside the listed packet and exclude host evaluators, quotient language equality, unlisted transition rules, and infinite traces. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonObligationWindow`

Theorem 1016.12 (FinitePrefixAutomaton deterministic readback)**.** *For accepted* FinitePrefixAutomaton$^{\uparrow}$ *packets with the same displayed table T and finite word W, the same prefix-window read forces the same displayed endpoint row and the same acceptance row. The conclusion is computed by induction over finite prefixes using only $Q, q_0, A, T, W, R, E, H, C, P, N$.*

Proof. For the empty prefix, the carrier and classifier force the run ledger to begin at the displayed initial row q_0. Thus two accepted reads with the same table and word have the same initial row inside the packet boundary.

For the successor prefix, apply transition exactness from Definition 1016.4. The same transition table row and the same input mark from W force the same displayed target row, while H and C only transport and replay that finite transition evidence.

Induction over the finite word ledger gives the same displayed run endpoint. Endpoint exactness and acceptance exactness, Definition 1016.5 and Definition 1016.6, then force the endpoint row and acceptance row read from E and A. No host evaluator, quotient language equality, unlisted transition rule, or infinite trace is available to change the result. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonDeterministicReadback`

Theorem 1016.13 (FinitePrefixAutomaton regular-language handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *or* $\mathsf{ContinuationAutomaton}^{\uparrow}$ *consumer of an accepted* $\mathsf{FinitePrefixAutomaton}^{\uparrow}$ *packet receives only finite word rows, transition rows, endpoint rows, acceptance rows, transport rows, replay rows, provenance rows, and local naming rows from*

$$Q, q_0, A, T, W, R, E, H, C, P, N.$$

The handoff exports no host evaluator, quotient language equality, unlisted transition rule, infinite trace, or derivation-tree comparison.

Proof. Read the carrier and classifier first. They expose the finite word ledger W, transition table T, run ledger R, endpoint and acceptance row E, and the accepting-state ledger A, with transport, replay, provenance, and naming supplied by H, C, P, N.

Apply the deterministic readback theorem above together with Definition 1016.9. The consumer receives a finite endpoint and acceptance read determined by the displayed word and transition table, not by an external evaluator.

Finally apply the regular-language sibling bridge of Definition 1016.10. The $\mathsf{RegularLanguage}^{\uparrow}$ side reads only finite word, endpoint, and acceptance data, and the $\mathsf{ContinuationAutomaton}^{\uparrow}$ side reads only the displayed transition and replay rows. Quotient language equality, unlisted transition rules, infinite traces, and derivation-tree comparisons remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomaton regular language handoff`

Theorem 1016.14 (FinitePrefixAutomaton public obligation certificate)**.** *The carrier, classifier, obligation surface, transition exactness, endpoint exactness, acceptance exactness, deterministic readback, and regular-language handoff together expose the full finite prefix automaton public obligation certificate over*

$$Q, q_0, A, T, W, R, E, H, C, P, N.$$

The certificate exports no host evaluator, quotient language equality, unlisted transition, derivation tree, or infinite trace.

Proof. Read the carrier and classifier of Definition 1016.1 and Definition 1016.2. They make $Q, q_0, A, T, W, R, E, H, C, P, N$ the displayed packet surface and leave no outside automaton coordinate.

Apply the NameCert obligation surface of Definition 1016.3. It binds carrier admission, prefix-source exposure, transition rows, endpoint and acceptance rows, transport, replay, provenance, and local naming to that same packet.

Use transition, endpoint, and acceptance exactness from Definition 1016.4, Definition 1016.5, and Definition 1016.6. These theorems force every accepted read through the displayed transition table, run ledger, endpoint row, and accepting-state row.

Finally apply deterministic readback and the regular-language handoff, Definition 1016.12 and Definition 1016.13. They keep all public consumers on the finite prefix route and exclude host evaluators, quotient language equality, unlisted transitions, derivation trees, and infinite traces. □

Lean4 ✓ `BEDC.Derived.FinitePrefixAutomatonUp.FinitePrefixAutomatonPublicObligationCertificate`

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixAutomaton}\uparrow}$.

Constructive story: A $\mathsf{FinitePrefixAutomaton}^{\uparrow}$ packet is generated from a finite automaton row, initial state, accepted prefix word, transition ledger, endpoint row, acceptance ledger, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The BHist carrier exposes a bounded run: each transition is read as a finite continuation step, the endpoint is determined by the displayed prefix, and the classifier admits acceptance only through the recorded endpoint and ledger rows. The construction therefore gives a finite-prefix automaton route that can be compared with adjacent language packets without quotienting languages.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixAutomaton}\uparrow}, \mathsf{obligation})$

Scope closed: This chapter binds the finite-prefix automaton carrier, obligation surface, determinacy, non-escape, and sibling bridge theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixAutomaton}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim language equality, derivation-tree equivalence, host evaluator correctness, infinite traces, or quotient automaton semantics.

Upgrade path: Scoped closure requires binding the displayed obligation certificate to the exact BEDC primitives and sibling dependencies already displayed in the chapter.

Chapter 1017

A Concrete Naming Certificate for $\mathsf{SharedTailNormalizer}^{\uparrow}$

The $\mathsf{SharedTailNormalizer}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as the finite operation row that turns two displayed tail schedules into one shared window route. It sits between the shared-tail input surfaces of chapter 893, chapter 560, and chapter 699, and the finite route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a normalizer for two Real-completion tail schedules, not a quotient stream, host equality, selected limit, or ambient completeness principle.

Falsifiable prediction: If two displayed tail schedules cannot be normalized to one shared finite window route, the packet cannot feed both the RegularCauchyTailFiber and RealTailAgreementSeal consumers.

Independence witness: regular-cauchy-tail-fiber, regular-cauchy-tail-meet, realtailagreementseal, regularcauchyselectorbudget, cauchysharedtailmodulus: these siblings expose common fiber, meet, seal, selector, or shared-modulus rows, while SharedTailNormalizerUp exposes the operation row that normalizes two schedules into one shared finite route.

Definition 1017.1 (Shared tail normalizer carrier)**.** *A* $\mathsf{SharedTailNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, W, D, R, F, S, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two incoming tail-schedule rows, each already displayed as a finite regular-Cauchy source route;*
2. *M is the shared threshold row that records the common finite bound serving both schedules;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window opened at the shared threshold;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared window;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the window and dyadic ledger;*
6. *F is the shared-fiber row consumed by* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$*;*
7. *S is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal route reached after W, D, R, F have been displayed;*

8. *H records componentwise* hsame *transport for both schedules, the shared threshold, window, dyadic ledger, readback, fiber, seal route, provenance, and local name;*
9. *C records the displayed* Cont *route that normalizes A, B through M, opens W, reads D, R, F, and reaches S;*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFiber}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SharedTailNormalizer}^{\uparrow}}$ *row.*

The carrier contains no second private threshold, quotient stream equality, host real equality, selected limit, hidden tail selector, detached Real seal, or ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.SharedTailNormalizerUp.SharedTailNormalizerUp`

Theorem 1017.2 (Shared tail normalizer NameCert obligations)**.** *For the carrier of Definition 1017.1, the local* $\mathsf{NameCert}_{\mathsf{SharedTailNormalizer}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (A, B, M, W, D, R, F, S, H, C, P, N)$, paired schedule exposure through A, B, shared threshold exposure through M, StreamName window service through W, DyadicRatCore tolerance accounting through D, RegSeqRat readback through R, shared-fiber exposure through F, RealTailAgreementSeal and RealUp handoff through S, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The rows $A, B, M, W, D, R, F, S, H, C, P, N$ are the complete public packet, so each consumer-facing read is one of the listed finite coordinates.

The source schedules are exposed first. The shared threshold M is then the only row that can serve both schedules, and the window W is opened through that same threshold before any dyadic ledger, regular readback, fiber, or Real-facing seal is available.

The finite route continues through chapter 126, chapter 291, and chapter 301. The shared-fiber read is the consumer surface of chapter 893, and the seal route is read through chapter 699 and chapter 38.

Since the carrier has no coordinate for a private threshold, quotient stream equality, host real equality, selected limit, hidden selector, detached seal, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 1017.3 (Shared tail normalizer common threshold factorization)**.** *Every accepted* $\mathsf{SharedTailNormalizer}^{\uparrow}$ *packet factors both incoming schedules through the one finite route*

$$A, B, M, W, D, R, F, S, H, C, P, N.$$

The $\mathsf{RegularCauchyTailFiber}^{\uparrow}$ *consumer reads F only after the shared StreamName, DyadicRatCore, and RegSeqRat rows W, D, R have been displayed, and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer reads S only after the same shared-fiber row F has been exposed.*

Proof. Begin with the paired schedule rows. Projection gives A and B as the only incoming schedules and M as the only common threshold row, so the route has no private schedule-specific threshold to branch through.

Next factor through the finite window route. The shared threshold opens the $\mathsf{StreamName}^{\uparrow}$ window W, the same window carries the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, and the regular rational readback R is obtained before any shared-fiber or Real-facing seal row is visible.

Now use the sibling surfaces. The row F is the common fiber consumed by chapter 893, while chapter 560 supplies the shared-tail threshold discipline that the normalizer records as M.

Finally read S, H, C, P, N. The seal route reaches chapter 699 and chapter 38 only after F is exposed, and the transport, continuation, provenance, and local naming rows replay that same finite route. □

The Lean follow-up for this packet will provide `Nontrivial SharedTailNormalizerUp` with `witness_pair` and `FieldFaithful SharedTailNormalizerUp` by constructor-field injection over the two schedules, shared threshold, StreamName window, DyadicRatCore ledger, RegSeqRat readback, shared fiber, Real seal route, transport, continuation, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SharedTailNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{SharedTailNormalizer}^{\uparrow}$ packet is generated from two displayed tail schedules, one shared threshold, one StreamName window, one DyadicRatCore ledger, one RegSeqRat readback, one shared-fiber row, one RealTailAgreementSeal and RealUp seal route, componentwise transports, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SharedTailNormalizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1017.1, Definition 1017.2, and Definition 1017.3 bind the finite shared-tail normalizer carrier, local NameCert obligations, and common threshold factorization route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SharedTailNormalizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, ambient Real completeness, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{SharedTailNormalizer}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, paired schedule exposure, shared-threshold factorization, StreamName and DyadicRatCore service rows, RegSeqRat readback, shared-fiber handoff, RealTailAgreementSeal handoff, and constructor-field faithfulness for the same finite packet.

Chapter 1018

A Concrete Naming Certificate for $\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}$

The $\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}$ packet isolates the AxisZeckendorf boundary token row that chapter 9350 refuses to read as a limit or supremum. It sits beside the dimension-lift boundary packet of chapter 571 and the FullAxis-to-Real refusal packet of chapter 602: those siblings aggregate broader refusal surfaces, while this carrier names only the boundary-token limit-refusal row and its non-escape routes. As an ai packet, it bundles the cannot-claim registry, dimension-lift boundary, FullAxis-to-Real refusal, AxisNat replacement refusal, and carry-normalization surfaces into one boundary-token refusal ledger.

Definition 1018.1 (Axis boundary limit-refusal carrier)**.** *An* $\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Z, L, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *Z is the AxisZeckendorf boundary-token row, including the* `boundary_01` *address from the cannot-claim registry;*
2. *L is the limit-refusal row, recording that the boundary token is not a limit or supremum carrier;*
3. *R is the Real-facing refusal row, recording that the same boundary does not provide* $\mathsf{Real}^{\uparrow}$ *equality, a Cauchy limit, or a completion row;*
4. *Q is the completeness-refusal row, recording that no total completeness principle is exported by the packet;*
5. *H records componentwise* hsame *transports preserving the boundary, limit-refusal, Real-facing refusal, completeness-refusal, provenance, and local naming rows;*
6. *C records the displayed* Cont *routes by which a consumer reads Z, L, R, Q, H, P, N without leaving the finite packet;*
7. *P is the* Pkg *provenance over* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{DimLiftBoundary}^{\uparrow}$, $\mathsf{FullAxisRealRefusal}^{\uparrow}$, $\mathsf{AxisNatReplacementRefusal}^{\uparrow}$, $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}}$ *row.*

The carrier exports only this boundary-token refusal packet. It contains no limit, supremum, $\mathsf{Real}^{\uparrow}$ *equality, Cauchy completion, completeness principle, or bridge to an ambient real object.*

Theorem 1018.2 (Axis boundary limit-refusal NameCert obligations)**.** *For the carrier of Definition 1018.1, the local* $\mathsf{NameCert}_{\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $B = (Z, L, R, Q, H, C, P, N)$*;*
2. *boundary-token exposure, stating that* Z *is the only AxisZeckendorf boundary row exported by the packet;*
3. *limit-refusal exactness, stating that* L *refuses limit and supremum readings of the boundary token;*
4. *Real and completeness refusal, stating that* R, Q *refuse* $\mathsf{Real}^{\uparrow}$ *equality, Cauchy completion, and completeness readings;*
5. *local naming exhaustion, stating that* H, C, P, N *preserve and name only the displayed refusal rows.*

Proof. Carrier admission is projection of the tuple in Definition 1018.1. The rows Z, L, R, Q, H, C, P, N are all displayed coordinates of one finite packet.

Boundary-token exposure reads Z against the cannot-claim registry of chapter 9350. The packet has no second boundary coordinate from which a limit object could be selected.

The refusal rows are separate. The row L blocks limit and supremum readings, while R and Q block $\mathsf{Real}^{\uparrow}$ equality, Cauchy completion, and completeness readings.

Finally, H, C, P, N transport, route, cite, and name those same rows. Since no coordinate names a limit, supremum, real equality, completion, or completeness object, the five obligations exhaust the local NameCert surface. □

Theorem 1018.3 (Axis boundary limit-refusal non-escape)**.** *Every consumer read of an accepted* $\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}$ *packet factors through*

$$Z \to L \to R \to Q \to H \to C \to P \to N.$$

The read may expose the boundary token and its refusal rows, but it admits no consumer route to a limit, supremum, $\mathsf{Real}^{\uparrow}$ *equality, Cauchy completion, or completeness row.*

Proof. Start with the carrier of Definition 1018.1. It lists the boundary token, limit-refusal row, Real-facing refusal row, completeness-refusal row, transport, continuation, provenance, and local name as the whole packet.

Apply Definition 1018.2. The boundary-token exposure and refusal exactness clauses force every consumer read through Z, L, R, Q before the transport and provenance rows are consumed.

The sibling packets chapter 571 and chapter 602 provide broader refusal contexts, but they do not add a hidden limit or real-completion row to this carrier. Thus the displayed route exhausts the accepted packet. □

Lean4 ✓ `BEDC.Derived.AxisBoundaryLimitRefusalUp.AxisBoundaryLimitRefusalCarrier nonescape`

Falsifiable prediction: Every accepted AxisBoundaryLimitRefusalUp packet exposes a boundary-token refusal row and admits no consumer route to a limit, supremum, RealUp equality, or completeness row.

Independence witness: axiszeckendorfcannotclaim, dimliftboundary, fullaxisrealrefusal, axisnatreplacementrefusal, zeckendorfcarrynormalization: none is bijective because this carrier isolates the boundary-token refusal row, while the siblings respectively aggregate registry entries,

dimension-lift refusal, FullAxis-to-Real refusal, AxisNat replacement refusal, and carry-normalization evidence.

The packet composes axiszeckendorfcannotclaim, dimliftboundary, fullaxisrealrefusal, axisnatreplacementrefusal, and zeckendorfcarrynormalization into the displayed boundary-token refusal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}$ packet is generated from an AxisZeckendorf boundary-token row, a limit-refusal row, a Real-facing refusal row, a completeness-refusal row, componentwise hsame transports, displayed Cont routes, provenance, and local naming. The BHist carrier first exposes the boundary token, then records that the token is not a limit or supremum carrier, and then blocks Real equality, Cauchy completion, and completeness readings. The classifier is a finite refusal route: every accepted consumer sees the boundary token only through the displayed refusal ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the axis boundary limit-refusal carrier, its NameCert obligations, and the non-escape theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxisBoundaryLimitRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a limit, supremum, RealUp equality, Cauchy completion, completeness principle, or ambient real bridge.

Upgrade path: To reach obligationClosure, formalize the boundary-token refusal route and prove that no accepted consumer extracts the excluded limit rows.

Chapter 1019

A Concrete Naming Certificate for $\mathsf{ContourReversalLedger}^{\uparrow}$

The $\mathsf{ContourReversalLedger}^{\uparrow}$ packet realizes the reversal law surface of chapter 9340 as finite NameCert data. It reads the contour-operation packet of chapter 521, displays the original and reversed piecewise-linear contour rows, and exposes the negated $\mathsf{Complex}^{\uparrow}$ output only through the same reversal route.

Definition 1019.1 (Contour reversal ledger carrier)**.** *A* $\mathsf{ContourReversalLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, G^{-}, F, I, H, C, P, N)$$

with the following displayed rows:

1. *G is the source* $\mathsf{PLContour}$ *row, including the finite endpoint list and segment rows read by the contour-integral operation surface;*
2. *G^{-} is the reversed* $\mathsf{PLContour}$ *row obtained by reading the same endpoints and segment rows in the displayed opposite orientation;*
3. *F is the accepted* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row for the integrand and the source contour data that supplies the operation-law ledger containing reversal;*
4. *I is the negated* $\mathsf{Complex}^{\uparrow}$ *output row certified as the value of the same integrand along G^{-} after the output row for G has been read;*
5. *H records componentwise* hsame *transports for G, G^{-}, F, I and for the provenance and local naming rows;*
6. *C records the displayed* Cont *routes from G to G^{-}, from F to the reversal-law ledger, and from the source output row to the negated* $\mathsf{Complex}^{\uparrow}$ *output row I;*
7. *P records* Pkg *provenance over* $\mathsf{PLContour}$, $\mathsf{ContourIntegralOperation}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{ContourSumWindow}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourReversalLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host smooth path, a quotient path class, an ambient analytic-continuation principle, an unrecorded integral value, or a nonledgered sign choice.

Lean4 ✓ `BEDC.Derived.ContourReversalLedgerUp.ContourReversalLedgerTasteGate single carrier alignment`

Theorem 1019.2 (Contour reversal ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContourReversalLedger}^{\uparrow}}$ *surface for Definition 1019.1 consists of carrier exposure for* $R = (G, G^-, F, I, H, C, P, N)$*, orientation-pair exposure for the two* PLContour *rows, contour-operation ledger exposure for the reversal law, negated* $\mathsf{Complex}^{\uparrow}$ *output routing, componentwise* hsame *transport, displayed* Cont *routes,* Pkg *provenance, and the local naming row. Any accepted consumer that asks for the reversed-contour output must first read the displayed PLContour row, reversed-contour row, and* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row.*

Proof. Carrier exposure is projection from the tuple in Definition 1019.1. The source contour row, reversed-contour row, operation row, negated output row, transport row, continuation row, provenance row, and local naming row are all displayed before any consumer can read the reversal result.

The orientation-pair obligation is confined to G and G^-. The route in C first reads the finite endpoint and segment data of G, then reads the opposite-orientation row G^-, so no host path object or quotient path class can replace either contour coordinate.

The reversal-law obligation is read through the $\mathsf{ContourIntegralOperation}^{\uparrow}$ row F and its operation-law ledger. The negated output row I is available only after that ledger has exposed the source contour, reversed contour, and operation row.

Finally, the rows H, C, P, N transport the displayed coordinates, replay the finite routes, record the dependency packages, and name the local ledger. Since no carrier coordinate stores an ambient analytic-continuation theorem, unrecorded integral value, or nonledgered sign choice, none of those objects can be exported. □

Theorem 1019.3 (Contour reversal ledger contour-integral handoff)**.** *Let* $R = (G, G^-, F, I, H, C, P, N)$ *be an accepted* $\mathsf{ContourReversalLedger}^{\uparrow}$ *carrier. Every* $\mathsf{ContourIntegralOperation}^{\uparrow}$*,* $\mathsf{ContourIntegralSocket}^{\uparrow}$*,* $\mathsf{ContourSumWindow}^{\uparrow}$*, residue-facing, or RH scaffold consumer of the reversal row factors first through* G*,* G^-*, and* F *before it may read the negated* $\mathsf{Complex}^{\uparrow}$ *output row* I*.*

The handoff exports only the displayed paired orientation rows, the contour-operation reversal ledger, the negated output route, componentwise transports, displayed continuations, package provenance, and local NameCert *row. It exports no host path equality, hidden branch choice, ambient integral operator, quotient path class, or unrecorded analytic-continuation coordinate.*

Proof. Begin with Definition 1019.2. It exposes the paired contour rows and forces the reversal-law read through the local $\mathsf{ContourIntegralOperation}^{\uparrow}$ row before the negated output row can be inspected.

For a contour-operation or contour-socket consumer, apply the operation-law ledger closure of chapter 521 and the socket boundary of chapter 583. Both sibling surfaces read contour data through displayed finite rows and ledgered routes, so the reversal handoff cannot bypass G, G^-, F, H, C, P, N.

For a contour-sum, residue-facing, or RH scaffold consumer, read the same handoff through the provenance row P. That row names the contour-sum window and analytic neighbors only as dependency packets; it does not add a host integral, quotient path, branch selector, or unrecorded analytic-continuation coordinate.

Thus every accepted consumer reaches the negated $\mathsf{Complex}^{\uparrow}$ output through the displayed orientation pair and contour-operation ledger, and the exported surface is exactly the finite packet R. □

Falsifiable prediction: Every accepted ContourReversalLedgerUp consumer reaches the negated ComplexUp output only after reading the displayed PLContour row, reversed-contour row, and

ContourIntegralOperation row.

Independence witness: contourintegraloperation, contoursumwindow, analyticcontinuationoperation, branchobstruction, holomorphic: none is bijective because the reversal ledger has the paired orientation rows and negated-output route that each sibling lacks.

The Lean file `TasteGate.lean` must include explicit Nontrivial ContourReversalLedger$^{\uparrow}$ and FieldFaithful ContourReversalLedger$^{\uparrow}$ instances for the displayed carrier $R = (G, G^{-}, F, I, H, C, P, N)$.

Closure status for NameCert$_{\text{ContourReversalLedger}^{\uparrow}}$.

Constructive story: A ContourReversalLedger$^{\uparrow}$ packet is generated from a displayed PLContour row, its reversed-contour row, a ContourIntegralOperation$^{\uparrow}$ reversal-law row, a negated Complex$^{\uparrow}$ output row, componentwise hsame transports, finite Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContourReversalLedger}^{\uparrow}}$, seed)

Scope closed: Definition 1019.1, Definition 1019.2, and Definition 1019.3 bind the finite orientation-pair ledger and negated-output route over PLContour, ContourIntegralOperation$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ContourReversalLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Cauchy's theorem, residue calculus, RH zero-location consequences, host path equality, global branch selection, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, reversal-ledger exactness, contour-operation handoff, TasteGate nontriviality, and field faithfulness for the same packet.

Chapter 1020

A Concrete Naming Certificate for $\mathsf{CauchyWindowTransducer}^{\uparrow}$

The $\mathsf{CauchyWindowTransducer}^{\uparrow}$ packet realizes the Real-completeness route of chapter 9238 as a finite transition object between StreamName windows, dyadic tolerance rows, RegSeqRat readbacks, LimitSelector handoffs, and RealUp seals. It is a NameCert carrier for one displayed window step; it is not a quotient stream, a host limit selector, a countable-choice principle, or an ambient completeness theorem.

The packet consumes chapter 126 for finite windows, chapter 291 for dyadic tolerances, chapter 301 for readback, and chapter 38 only after the RegSeqRat handoff.

Definition 1020.1 (Cauchy window transducer carrier)**.** *A* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, D, W, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window source row;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row read at the same window scale;*
3. *W is the local window-step row translating the source window and dyadic tolerance into the next finite request;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced after the window step;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after the readback row is displayed;*
6. *L is the* $\mathsf{LimitSelector}^{\uparrow}$ *handoff row naming the diagonal-limit selector used by the window step;*
7. *H records componentwise* hsame *transports for $S, D, W, R, E, L, C, P, N$;*
8. *C records displayed* Cont *routes from S, D to W, from W to R, from R to E, and from L to the same step route;*
9. *P records* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LimitSelector}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyWindowTransducer}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected external limit, quotient equality of real names, global stream equality, countable choice, or a RealUp seal read that bypasses the displayed RegSeqRat handoff.

Lean4 (definition) `BEDC.Derived.CauchyWindowTransducerUp.taste gate`

Theorem 1020.2 (Cauchy window transducer NameCert obligations)**.** *For the carrier of Definition 1020.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyWindowTransducer}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (S, D, W, R, E, L, H, C, P, N),$$

StreamName window exposure through S, dyadic tolerance exposure through D, local window-step exposure through W, RegSeqRat readback through R, RealUp seal routing through E, LimitSelector handoff through L, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from the displayed tuple T. Each public row of the transducer is one of $S, D, W, R, E, L, H, C, P, N$.

The analytic rows are read in order. The StreamName row and dyadic tolerance row supply the finite request surface, the local window-step row consumes that pair, the RegSeqRat row records the readback, and the RealUp seal row is reachable only after that readback.

The LimitSelector row names the diagonal selector used by the same step route. The structural rows H, C, P, N transport, replay, package, and name this finite inventory, so no hidden limit or quotient coordinate remains in the NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyWindowTransducerUp.CauchyWindowTransducerCarrier namecert obligations`

Theorem 1020.3 (Cauchy window transducer window-step determinacy)**.** *Let two accepted* CauchyWindowTransdu *packets agree on the StreamName window row S, dyadic tolerance row D, LimitSelector row L, and structural rows H, C, P, N. If both packets expose the same window-step row W, then they determine the same RegSeqRat readback row R and the same RealUp seal route E.*

Proof. Project both packets through Definition 1020.1. The only analytic continuation after S, D, L is the displayed window-step row W followed by R and E.

Apply Definition 1020.2. It identifies route replay with the displayed Cont rows in C and binds the same package provenance and local name through P and N.

Since the two packets agree on S, D, L, W, H, C, P, N, there is no remaining carrier coordinate from which a second readback or seal route could be selected. The RegSeqRat readback and RealUp seal route are therefore determined by the common window step. □

Lean4 ✓ `BEDC.Derived.CauchyWindowTransducerUp.CauchyWindowTransducerCarrier step spine determinacy`

Theorem 1020.4 (Cauchy window transducer RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet hands a RealUp-facing consumer to the seal row E only through the RegSeqRat readback row R. A consumer may inspect the StreamName window, dyadic tolerance, window step, and LimitSelector handoff, but the RealUp seal is not exported until R has been displayed.*

Proof. Use the carrier definition to read the analytic route

$$S, D, L \longmapsto W \longmapsto R \longmapsto E.$$

The row E occurs after R in the displayed inventory and has no separate continuation source.

The NameCert obligation theorem then fixes the admissible public reads. Route replay through C reaches the RealUp seal only along the displayed route, while H transports the same rows componentwise and P, N bind the route to the local certificate.

Thus a RealUp-facing consumer can read the seal only after the RegSeqRat handoff row has been consumed. A route that bypasses R is outside the carrier. □

Theorem 1020.5 (Cauchy window transducer Real seal non-escape)**.** *No accepted* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *packet exports a RealUp seal read from an external limit selector, quotient stream equality, global stream equality, countable-choice row, or hidden completeness theorem. Every RealUp seal read factors through*

$$S, D, W, R, E, L, H, C, P, N.$$

Proof. Start with Definition 1020.4. It forces every RealUp-facing seal read through the displayed RegSeqRat handoff R.

Apply Definition 1020.3. Agreement on the source window, tolerance, selector, step, and structural rows determines the readback and seal route, so no hidden selector remains available after the displayed rows are fixed.

Finally use the exclusion clause in Definition 1020.1 together with Definition 1020.2. The carrier lists the full finite inventory and explicitly lacks quotient, global equality, choice, and ambient completeness coordinates. □

Theorem 1020.6 (Cauchy window transducer L10 lattice link)**.** *Every accepted* $\mathsf{CauchyWindowTransducer}^{\uparrow}$ *route factors through the sibling surfaces of chapter 126, chapter 291, chapter 301, chapter 673, and chapter 38 before the local* $\mathsf{Real}^{\uparrow}$ *seal is public. Equivalently, the displayed finite route consumes the StreamName finite-window row, DyadicRatCore tolerance row, local window-step row, RegSeqRat handoff row, LimitSelector handoff row, RealUp seal row, transport row, continuation row, provenance row, and local NameCert row before any Real-facing consumer can read the seal.*

Proof. The carrier of Definition 1020.1 names the five sibling surfaces as rows S, D, R, L, E together with the local structural rows W, H, C, P, N. The NameCert obligation theorem Definition 1020.2 says that these are exactly the public rows of an accepted transducer route.

Window-step determinacy from Definition 1020.3 fixes the passage from the StreamName and DyadicRatCore rows, with the LimitSelector handoff, to the RegSeqRat readback and the RealUp seal route. The RegSeqRat handoff theorem Definition 1020.4 then prevents the Real-facing consumer from reading E before R has been displayed.

Finally, Real seal non-escape Definition 1020.5 exhausts the remaining coordinates. No quotient stream, selected host limit, countable-choice row, or ambient completeness coordinate is introduced; the L10 dependency is precisely the finite sibling route recorded in the displayed packet. □

Lean4 ✓ `BEDC.Derived.CauchyWindowTransducerUp.CauchyWindowTransducerUp StdBridge`

Falsifiable prediction: Removing the window-step row permits a RealUp seal read without a displayed RegSeqRat handoff.

Independence witness: Nearest siblings: StreamName, DyadicRatCore, RegSeqRat, RealUp, and LimitSelector. None is bijective to $\mathsf{CauchyWindowTransducer}^{\uparrow}$ because this carrier stores the joint window source, dyadic tolerance, step row, readback row, seal route, selector handoff, transport, replay, provenance, and local naming rows in one finite packet, while each sibling exposes only one surface of that route.

Theorem 1020.7 (Cauchy window transducer scoped closure package)**.** *The carrier of Definition 1020.1, the NameCert obligations of Definition 1020.2, window-step determinacy from Definition 1020.3, the RegSeqRat handoff of Definition 1020.4, Real seal non-escape from Definition 1020.5, the falsifiable prediction, the independence witness, and the checked TasteGate alignment jointly bind the scoped finite route from StreamName windows and DyadicRatCore tolerances through RegSeqRat readback to the RealUp seal.*

Proof. The carrier fixes the whole packet as the displayed tuple $S, D, W, R, E, L, H, C, P, N$. The NameCert obligation theorem says that these rows are exactly the local public surface.

Window-step determinacy fixes the passage from the source window, dyadic tolerance, selector handoff, and local step row to the RegSeqRat readback and RealUp seal route. The RegSeqRat handoff theorem then forces every Real-facing read through R before E is public.

Real seal non-escape excludes the remaining ambient coordinates. Together with the falsifiable prediction and independence witness, this leaves a scoped finite route rather than a quotient real, selected host limit, global stream equality, countable-choice principle, or hidden completeness theorem. □

Lean4 ✓ `BEDC.Derived.CauchyWindowTransducerUp.CauchyWindowTransducerTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWindowTransducer}\uparrow}$.

Constructive story: A $\mathsf{CauchyWindowTransducer}^{\uparrow}$ packet is generated from a StreamName finite-window source row, DyadicRatCore tolerance row, local window-step row, RegSeqRat readback row, RealUp seal row, LimitSelector handoff row, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWindowTransducer}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1020.1, Definition 1020.2, Definition 1020.3, Definition 1020.4, Definition 1020.5, and Definition 1020.7 bind the finite route over StreamName, DyadicRatCore, RegSeqRat, RealUp, LimitSelector, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWindowTransducer}\uparrow}, \mathsf{bridgeChecked})$

Lean4 ✓ `BEDC.Derived.CauchyWindowTransducerUp.taste gate`

Lean target: `BEDC.Derived.CauchyWindowTransducerUp.CauchyWindowTransducerTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close selected limits, quotient real equality, global stream equality, countable choice, or ambient completeness.

Upgrade path: No further upgrade remains; the finite Cauchy window transducer route is at the top of both the paper closure axis and the bridge verification axis.

Chapter 1021

A Concrete Naming Certificate for $\mathsf{LargeModelAttentionGate}^{\uparrow}$

The $\mathsf{LargeModelAttentionGate}^{\uparrow}$ packet realizes the attention-window boundary of chapter 9398 as finite BEDC data. It sits between the activation-orbit packet of chapter 967, the context-audit packet of chapter 989, the corpus-supply packet of chapter 988, the audit channel of chapter 969, and the verified-output harness of chapter 976. Its role is not to introduce a new model semantic oracle; it records the finite gate through which an output-facing read must pass from trace and activation rows into context, corpus, audit, verified-output, transport, route, provenance, and local naming rows.

Definition 1021.1 (LargeModelAttentionGate carrier)**.** *A* $\mathsf{LargeModelAttentionGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, A, C, K, U, V, H, R, P, N)$$

with the following displayed rows:

1. *T is the trace row, recording the finite model-trace read that supplies the emitted output-facing state;*
2. *A is the* $\mathsf{LargeModelActivationOrbit}^{\uparrow}$ *row naming the finite activation-window segment selected for the read;*
3. *C is the context-window row, carrying the bounded prompt and response boundary consumed by the gate;*
4. *K is the* $\mathsf{LargeModelCorpusSupply}^{\uparrow}$ *row that records the corpus-supply and filter boundary available to this local selection;*
5. *U is the* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *row through which the selected output-facing read is routed to a public audit consumer;*
6. *V is the* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *handoff row for the same output-facing record;*
7. *H records componentwise* hsame *transport for the trace, activation-window, context, corpus-supply, audit-channel, verified-output, replay, provenance, and naming rows;*
8. *R records the displayed* Cont *replay by which a consumer reads T, then A, C, K, then U, V, before reaching provenance and the local name;*

9. *P is the* Pkg *provenance over* ModelTrace$^{\uparrow}$, LargeModelActivationOrbit$^{\uparrow}$, ContextWindowCommitment$^{\uparrow}$, LargeModelCorpusSupply$^{\uparrow}$, LargeModelAuditChannel$^{\uparrow}$, VerifiedOutputHarness$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. *N is the local* NameCert$_{\mathsf{LargeModelAttentionGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden attention stream, unbounded context recovery, corpus-wide truth, unchecked output acceptance, model-side semantic proof, or private route bypassing the displayed activation and context rows.

Lean4 ✓ `BEDC.Derived.LargeModelAttentionGateUp.LargeModelAttentionGateTasteGate single carrier alignment`

Theorem 1021.2 (LargeModelAttentionGate NameCert obligations)**.** *For the carrier of Definition 1021.1, the local* NameCert$_{\mathsf{LargeModelAttentionGate}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for $G = (T, A, C, K, U, V, H, R, P, N)$;*
2. *activation-window readback through the displayed activation-orbit row A;*
3. *corpus boundary exposure through the corpus-supply row K;*
4. *audit-channel routing through U;*
5. *verified-output handoff through V;*
6. *componentwise transport through H;*
7. *replay through the displayed* Cont *row R;*
8. *provenance through P;*
9. *local naming through N.*

Every accepted attention-gate consumer must display these rows before it can read the output-facing gate as a certified large-model packet.

Proof. Project the carrier of Definition 1021.1. The packet lists exactly $T, A, C, K, U, V, H, R, P, N$, so carrier admission supplies no additional hidden state.

The activation and context rows are the entry surface. The trace row T is only readable through the activation-window row A and the bounded context row C, while K keeps the corpus-supply boundary attached to the same local read.

The output-facing rows are downstream. The audit-channel row U and verified-output row V are reached only after the displayed activation, context, and corpus rows have been consumed.

Finally H, R, P, N provide transport, replay, provenance, and local naming for the same packet. Since the carrier has no hidden attention stream, context-recovery oracle, corpus-truth predicate, unchecked acceptance row, or model-side semantic proof coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 1021.3 (LargeModelAttentionGate finite-window soundness)**.** *Every output-facing read of an accepted* LargeModelAttentionGate$^{\uparrow}$ *packet factors through the finite activation and context window before it can reach an audit-channel or verified-output row. In the carrier*

$$G = (T, A, C, K, U, V, H, R, P, N),$$

the read order is

$$T \to (A, C, K) \to (U, V) \to (H, R, P, N).$$

Thus an output-facing consumer cannot read U or V as a bare checker result, hidden evaluator state, unbounded attention stream, recovered context, corpus-wide truth predicate, or host semantic equality.

Proof. Start with Definition 1021.2. It requires activation-window readback through A and context exposure through C before audit-channel routing or verified-output handoff is part of the local surface.

Use the sibling chapters to identify those rows. chapter 967 supplies the finite activation-orbit row, chapter 989 and chapter 971 supply the bounded context route, and chapter 988 supplies the corpus boundary.

Only after this finite window is displayed do chapter 969 and chapter 976 provide the audit and verified-output rows. The replay, provenance, and naming rows R, P, N then certify that the same finite route was consumed.

The exclusions follow from the carrier inventory. No coordinate stores a bare checker result, hidden evaluator, unbounded attention stream, recovered context, corpus-wide truth predicate, or host semantic equality, so no accepted read can bypass the finite activation and context window. □

Falsifiable prediction: Every LargeModelAttentionGateUp verified-output read factors through a displayed activation-orbit row and context-window row before it can reach the audit-channel handoff.

Independence witness: largemodelinscriptionpoint, largemodelactivationorbit, largemodelcontextaudit, largemodelauditchannel, verifiedoutputharness: the attention-gate packet is not bijective with these siblings because inscription lacks the selected context gate, activation orbit lacks the audit and verified-output handoff, context audit lacks the activation-window selector as the classifier-bearing entry, audit channel lacks the finite activation/context window, and verified output harness lacks the corpus-supply gate attached to the same selection route.

The Lean TasteGate carrier names the same displayed packet and carries `Nontrivial LargeModelAttentionGa` and `FieldFaithful LargeModelAttentionGateUp` instances for it.

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelAttentionGate}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{LargeModelAttentionGate}^{\uparrow}$ packet is generated from trace, activation-orbit, context-window, corpus-supply, audit-channel, verified-output, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeModelAttentionGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1021.1, Definition 1021.2, and Definition 1021.3 specify the finite activation-window and context-window attention-gate carrier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeModelAttentionGate}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LargeModelAttentionGateUp.LargeModelAttentionGateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close hidden attention streams, unbounded context recovery, corpus truth, host semantic understanding, unchecked output acceptance, Lean verification, or bridge checking for $\mathsf{LargeModelAttentionGate}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, activation-window readback, corpus-boundary exposure, audit-channel routing, verified-output handoff, componentwise transport, replay, provenance, local naming, nontriviality, and field faithfulness for the same carrier.

Chapter 1022

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$

The $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$ packet records the common finite-window intersection used when two regular-Cauchy routes share one tail-meet row before the Real or Completion consumer reads a tail-agreement seal. It consumes the threshold service of chapter 560 and the agreement handoff of chapter 699, while keeping the two selected windows visible as separate rows until the shared intersection packet has been named.

Definition 1022.1 (Regular-Cauchy window-intersection carrier)**.** *A* $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, W_X, W_Y, D, M, T, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two selected* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy source rows;*
2. *W_X and W_Y are the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows for those source rows;*
3. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback ledger visible to both windows;*
4. *M is the accepted* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *row supplying the common tail-meet threshold;*
5. *T is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff row exposed only after W_X, W_Y, D, M are displayed together;*
6. *H records componentwise* hsame *transports for the two source rows, the two window rows, the dyadic ledger, the tail-meet row, the agreement handoff, provenance, and the local naming row;*
7. *C records displayed* Cont *routes from the two source rows to the selected windows, from those windows to the common dyadic ledger, from the ledger to the tail meet, and from the tail meet to the agreement handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowIntersection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, independent tail thresholds, unshared window services, ambient completeness, or a RealTailAgreementSeal handoff that bypasses the displayed intersection packet.

Lean4 (definition) BEDC.Derived.RegularCauchyWindowIntersectionUp.RegularCauchyWindowIntersectionUp
Lean4 ✓ BEDC.Derived.RegularCauchyWindowIntersectionUp.RegularCauchyWindowIntersectionTasteGate
single carrier alignment

Theorem 1022.2 (Regular-Cauchy window-intersection NameCert obligations)**.** *For the carrier of Definition 1022.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowIntersection}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$I = (X, Y, W_X, W_Y, D, M, T, H, C, P, N),$$

the two regular-Cauchy source rows X, Y*, the selected window rows* W_X, W_Y*, the common dyadic ledger* D*, the common tail-meet row* M*, the RealTailAgreementSeal handoff row* T*, componentwise transport through* H*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed packet I. Its coordinates are exactly the two sources, two selected windows, common dyadic ledger, tail-meet row, agreement handoff, transport, replay, provenance, and local name.

The tail-meet coordinate is consumed through chapter 560. The agreement handoff is consumed through chapter 699 only after both selected windows and the common dyadic ledger have been displayed.

No remaining carrier coordinate can expose quotient stream equality, a selected host limit, an independent threshold, an unshared window service, ambient completeness, or another route to the agreement handoff. The listed rows therefore exhaust the local NameCert surface. □

Theorem 1022.3 (Regular-Cauchy window-intersection falsifiable boundary)**.** *Any accepted* RegularCauchyWindow *consumer that presents two selected regular-Cauchy windows with a common tail-meet row must expose the same* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff row* T*. If the two windows present different handoff rows after the same* D, M, H, C, P, N *coordinates are fixed, the packet is rejected by the local NameCert ledger.*

Proof. The carrier places T after the two selected windows, the common dyadic ledger, and the common tail-meet row. Thus a consumer can reach a RealTailAgreementSeal handoff only through the displayed route recorded by C.

Fixing D, M, H, C, P, N fixes the common ledger, the tail-meet threshold, componentwise transport, replay route, provenance, and local name. There is no second handoff coordinate in the packet.

Hence an accepted packet exposes one handoff row for the shared intersection. A presentation with two different handoff rows is not a carrier of Definition 1022.1. □

Theorem 1022.4 (Regular-Cauchy window-intersection sibling independence)**.** *The carrier of Definition 1022.1 is not bijective to the regularcauchywitnessschedule, regular-cauchy-tail-meet, realtailagreementseal, finiteobservationbudgetselector, or cauchylimitseal sibling packets. Each sibling supplies a schedule, a threshold meet, a seal row, a selector route, or a limit seal, while this carrier records the common intersection packet binding both selected windows before the seal.*

Proof. Inspect the displayed rows of I. The packet contains two selected windows, one common dyadic ledger, one tail-meet row, and one agreement handoff in the same finite carrier.

The witness-schedule sibling supplies a schedule but not the paired intersection carrier. The tail-meet sibling supplies a threshold meet but not the stored pair of selected windows before the handoff. The RealTailAgreementSeal sibling supplies the seal row but not the common intersection packet binding both windows before that seal.

The selector and limit-seal siblings likewise expose a selector route or a limit seal without the full pair of selected windows, shared ledger, tail meet, and agreement handoff. No bijection can preserve the complete NameCert obligation surface. □

Theorem 1022.5 (Regular-Cauchy window-intersection ledger exhaustion)**.** *Every Real or Completion diagonal-route consumer of an accepted* $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$ *packet reads only*

$$X, Y, W_X, W_Y, D, M, T, H, C, P, N.$$

The consumer receives the two source rows, two selected windows, common dyadic ledger, common tail meet, RealTailAgreementSeal handoff, componentwise transports, displayed continuations, provenance, and local name; it receives no quotient stream equality, selected host limit, independent threshold, unshared window service, or ambient completeness principle.

Proof. Start from the NameCert obligation theorem. It identifies the complete public ledger with the rows of the displayed packet.

Apply the falsifiable boundary theorem to the handoff row. The RealTailAgreementSeal handoff is reachable only as the single row T after the two windows and common tail meet have been displayed.

Use sibling independence for the remaining exclusions. None of the nearest sibling packets supplies an additional coordinate that could be imported into this carrier. The consumer is therefore exhausted by the displayed ledger. □

Theorem 1022.6 (Regular-Cauchy window-intersection shared-threshold determinacy)**.** *Let*

$$I = (X, Y, W_X, W_Y, D, M, T, H, C, P, N) \quad \text{and} \quad I' = (X', Y', W'_X, W'_Y, D', M', T', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$ *packets. If the two packets expose the same selected* $\mathsf{StreamName}^{\uparrow}$ *windows for the same two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the same* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *threshold row, then every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*-facing read agrees on the displayed shared-threshold packet*

$$X, Y, W_X, W_Y, D, M, T, H, C, P, N.$$

Conversely, equality of the seal-facing read recovers only the displayed source-window rows, tolerance ledger, threshold row, hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row. It recovers no quotient stream equality, selected limit, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge row.*

Proof. Begin with the carrier. By Definition 1022.1, an accepted packet has exactly the two source rows, two selected window rows, common dyadic ledger, tail-meet row, seal handoff, transport, continuation, provenance, and naming coordinates displayed in I.

The NameCert obligations then fix the admissible readings of those coordinates. Definition 1022.2 requires the same displayed route through the selected windows, the common dyadic ledger, the tail-meet row, and the RealTailAgreementSeal handoff. The tail-meet coordinate is the finite threshold service supplied by chapter 560, and the handoff coordinate is read through chapter 699.

Now compare I and I'. Once the two selected source-window pairs, D, and M agree, the displayed route replay through C has no second threshold coordinate to choose. Definition 1022.5 says that a Real or Completion consumer reads only the rows of the displayed packet, so the seal-facing read is the same shared-threshold packet.

For the converse, equality of the seal-facing read can recover exactly the rows that the ledger exposes: $X, Y, W_X, W_Y, D, M, T, H, C, P, N$. The carrier contains no coordinate for quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge data, and sibling independence prevents importing such a coordinate from the neighboring packets. Thus determinacy is finite and local to the displayed RegularCauchyWindowIntersection packet. □

Falsifiable prediction: Any accepted RegularCauchyWindowIntersectionUp consumer that presents two selected regular-Cauchy windows with a common tail-meet row must expose the same RealTailAgreementSealUp handoff row; otherwise the packet is rejected by the local NameCert ledger.

Independence witness: regularcauchywitnessschedule, regular-cauchy-tail-meet, realtailagreementseal, finiteobservationbudgetselector, cauchylimitseal: none is bijective because each sibling supplies a single schedule, a threshold meet, a seal row, a selector route, or a limit seal, while this carrier records the common intersection packet binding both windows before the seal.

The Lean boundary must provide `Derived/RegularCauchyWindowIntersectionUp/TasteGate.lean` with explicit `Nontrivial` and `FieldFaithful` instances, so the intersection is represented by an inhabited, field-faithful carrier rather than by a vacuous seal route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyWindowIntersection}^{\uparrow}$ packet is generated from two regular-Cauchy source rows, two selected StreamName windows, a common DyadicRatCore ledger, a RegularCauchyTailMeet row, a RealTailAgreementSeal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowIntersection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1022.1, Definition 1022.2, Definition 1022.3, Definition 1022.4, Definition 1022.5, and Definition 1022.6 bind the finite shared-window intersection packet and its RealTailAgreementSeal handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWindowIntersection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected host limits, independent tail thresholds, unshared window services, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window intersection obligations, handoff determinacy, sibling independence, and consumer ledger exhaustion for the same packet.

Chapter 1023

A Concrete Naming Certificate for $\mathsf{InterHistRateBoundary}^{\uparrow}$

The $\mathsf{InterHistRateBoundary}^{\uparrow}$ packet realizes the inter-Hist maximum-rate boundary of chapter 9419 as finite BEDC data. It binds the cross-history rate surface of chapter 876, the read gate of chapter 933, and the refusal surface of chapter 1003 into one local naming carrier, so downstream consumers receive a gated route rather than a host speed, metric row, clock, or global frame.

Falsifiable prediction: An accepted rate-boundary consumer that reads a maximum-rate row without both observer-symmetry and no-global-frame refusal rows refutes the packet.

Independence witness: crosshistcausalrate, maxratereadgate, noglobalsyncboundary, interhistinvariant, crosshistcausalroute: the boundary packet is not bijective to any one sibling because it carries the shared gate plus route plus refusal surface.

Definition 1023.1 (InterHistRateBoundary carrier)**.** *An* $\mathsf{InterHistRateBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, D, G, R, S, F, I, H, Q, C, P, N)$$

with the following visible rows:

1. *A is the paired observer-history boundary row that supplies the two displayed inter-Hist sides;*
2. *D is the causal-route row, read through* $\mathsf{CrossHistCausalRoute}^{\uparrow}$ *and* $\mathsf{CrossHistCausalRate}^{\uparrow}$ *rather than through an ambient spacetime path;*
3. *G is the accepted* $\mathsf{MaxRateReadGate}^{\uparrow}$ *row binding the rate read to the gate surface of chapter 933;*
4. *R is the maximum-rate row tied to* $\mathsf{MaxCausalRate}^{\uparrow}$ *as in chapter 594;*
5. *S is the observer-symmetry row for the displayed histories;*
6. *F is the no-global-frame refusal row, with the same refusal discipline as chapter 1003;*
7. *I is the inter-Hist invariant readback row, recording that the boundary is stable as an inter-Hist packet rather than a single-history rate;*
8. *H records componentwise* hsame *transport for the displayed rows;*
9. *Q records* psame *stability for the probes used by D, G, R, S, F, I;*
10. *C records displayed* Cont *routes from the causal route and gate into the local naming row;*

11. *P is the* Pkg *provenance over* InterHistRateBoundary$^{\uparrow}$, CrossHistCausalRate$^{\uparrow}$, MaxRateReadGate$^{\uparrow}$, InterHistInvariant$^{\uparrow}$, NoGlobalSyncBoundary$^{\uparrow}$, CrossHistCausalRoute$^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* NameCert*;*
12. *N is the local* NameCert$_{\mathsf{InterHistRateBoundary}^{\uparrow}}$ *row.*

The carrier exports a finite maximum-rate boundary for inter-Hist consumers. It exports no host-side speed, global time coordinate, metric tensor, Lorentz frame, selected synchronization map, or detached rate primitive.

Theorem 1023.2 (InterHistRateBoundary NameCert obligations)**.** *For the carrier of Definition 1023.1, the local* NameCert$_{\mathsf{InterHistRateBoundary}^{\uparrow}}$ *surface consists exactly of seven finite obligations:*

1. *carrier admission for $B = (A, D, G, R, S, F, I, H, Q, C, P, N)$;*
2. *causal-route exactness, requiring D to be read through the displayed* CrossHistCausalRoute$^{\uparrow}$ *and* CrossHistCausalRate$^{\uparrow}$ *rows;*
3. *maximum-rate gate exactness, requiring G to bind R, S, F before any consumer reads the rate row;*
4. *observer-symmetry exactness through S after componentwise transport;*
5. *no-global-frame refusal through F before any boundary read is public;*
6. *invariant readback through I, keeping the packet on the inter-Hist surface;*
7. *transport, probe, route, provenance, and local-name stability through H, Q, C, P, N.*

Proof. Carrier admission is the projection of the finite tuple in Definition 1023.1. The rows $A, D, G, R, S, F, I, H, Q, C, P, N$ display the observer boundary, causal route, rate gate, maximum-rate row, symmetry row, refusal row, invariant row, transports, probe stability, routes, provenance, and local name.

Apply the causal-rate surface from chapter 876. It supplies the route discipline for D and prevents the boundary from reading a rate before the cross-history causal route is displayed.

Apply the max-rate read gate from chapter 933. The gate binds R to S and F, so the maximum-rate row is not public until both observer symmetry and no-global-frame refusal have been read.

Finally use the invariant and refusal rows. The invariant row keeps the packet on the inter-Hist surface, while the refusal row excludes global frames, clocks, metric tensors, Lorentz frames, synchronized coordinates, and detached host speeds. The remaining rows H, Q, C, P, N transport, route, package, and name the same finite packet. □

Theorem 1023.3 (InterHistRateBoundary gated consumer)**.** *Every* CrossHistCausalRate$^{\uparrow}$, LorentzFrameRate$^{\uparrow}$, MinkowskiRateGeometry$^{\uparrow}$, *or inter-Hist locality consumer of an accepted* InterHistRateBoundary$^{\uparrow}$ *packet may read the maximum-rate row R only through the displayed boundary rows*

$$A, D, G, R, S, F, I, H, Q, C, P, N.$$

In particular, the read of R passes through the causal route D, the gate row G, the observer-symmetry row S, the no-global-frame refusal row F, and the invariant row I before it reaches any downstream consumer.

Proof. Project the accepted boundary carrier. The only maximum-rate row is R, and the only displayed path to it is through the route and gate rows D and G.

Use Definition 1023.2. The obligations make causal-route exactness, gate exactness, symmetry, refusal, and invariant readback part of one local naming surface.

The max-rate gate of chapter 933 supplies the public lock for R, S, F. The causal-rate packet of chapter 876 supplies the cross-history route read before the rate row is inspected.

Thus a downstream consumer receives a finite boundary packet rather than a free maximum-rate primitive. Any read that bypasses D, G, S, F, or I is outside the accepted $\mathsf{InterHistRateBoundary}^{\uparrow}$ carrier. □

Theorem 1023.4 (InterHistRateBoundary nonescape)**.** *An accepted* $\mathsf{InterHistRateBoundary}^{\uparrow}$ *packet has no admissible export of a host speed, global time coordinate, synchronized observer frame, metric tensor, Lorentz frame, selected synchronization map, or detached maximum-rate row. Its public surface is exhausted by* $A, D, G, R, S, F, I, H, Q, C, P, N$ *and the local* $\mathsf{NameCert}_{\mathsf{InterHistRateBoundary}^{\uparrow}}$ *obligations over those rows.*

Proof. Start from the gated consumer theorem. Every rate read factors through the displayed route, gate, symmetry, refusal, and invariant rows.

The no-global-frame refusal row F is read with the boundary discipline of chapter 1003. It blocks any interpretation of the packet as a global synchronization surface or privileged frame.

The carrier inventory then completes the non-escape argument. Since the tuple contains no coordinate for host speed, global time, metric geometry, Lorentz-frame data, selected synchronization, or detached rate values, no local NameCert projection can export one. □

Theorem 1023.5 (InterHistRateBoundary NameCert package)**.** *The carrier, local NameCert obligations, gated consumer theorem, and non-escape theorem assemble the concrete* $\mathsf{NameCert}_{\mathsf{InterHistRateBoundary}^{\uparrow}}$ *package over* $\mathsf{CrossHistCausalRate}^{\uparrow}$, $\mathsf{MaxRateReadGate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{CrossHistCausalRoute}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* NameCert. *The Lean carrier surface is realized by* `Nontrivial InterHistRateBoundaryUp` *and* `FieldFaithful InterHistRateBoundaryUp` *instances in* `lean4/BEDC/Derived/InterHistRateBoundaryUp/TasteGate.lean`, *with the canonical declaration surface at* `BEDC.Derived.InterHistRateBoundaryUp.taste_gate`.

Proof. Combine the carrier definition with the NameCert obligations. They fix the twelve displayed rows and bind route, gate, symmetry, refusal, invariant readback, transport, probe stability, continuation, provenance, and local naming.

The gated consumer theorem identifies how downstream rate and geometry consumers may read the packet. The non-escape theorem identifies what they cannot read.

The cited Lean TasteGate carrier supplies the finite event-flow presentation, round-trip readback, nontriviality, and field faithfulness for the same row surface. Therefore the paper package and the Lean carrier use the same finite BEDC data rather than a host metric or global-frame primitive. □

Lean4 ✓ `BEDC.Derived.InterHistRateBoundaryUp.InterHistRateBoundaryTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{InterHistRateBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{InterHistRateBoundary}^{\uparrow}$ packet is generated from paired observer-history boundary rows, a cross-history causal route, a maximum-rate gate, rate row, observer-symmetry row, no-global-frame refusal, inter-Hist invariant readback, componentwise hsame transport, probe stability, displayed Cont routes, provenance, and local naming. The carrier makes the rate visible only through the gate and refusal rows. Its classifier records a bounded inter-Hist relation: the consumer reads route, gate, symmetry, refusal, and invariant data, while hsame and Cont preserve exactly those fields without importing a host metric or shared clock.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InterHistRateBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the inter-Hist rate-boundary carrier, obligations, gated consumer theorem, non-escape theorem, and paper-Lean package comparison.

Formal verification: FormalStatus(NameCert$_{\mathsf{InterHistRateBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim host-side speed, global time, metric tensor, Lorentz frame, selected synchronization map, or detached rate primitive.

Upgrade path: To reach obligationClosure, promote the displayed TasteGate-aligned carrier into explicit checked obligations for all route, gate, refusal, and invariant rows.

Chapter 1024

A Concrete Naming Certificate for $\mathsf{CauchyTailModulusFusion}^{\uparrow}$

The $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ packet realizes the Real-completeness route of chapter 9238 by joining the shared-tail surface of chapter 658, the modulus refinement surface of chapter 885, and the shared threshold service of chapter 983. Its finite rows are read through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ certificate surfaces of chapter 126, chapter 291, chapter 301, and chapter 38.

Definition 1024.1 (Cauchy tail modulus fusion carrier)**.** *A* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N)$$

with the following displayed rows:

1. *T_0 and T_1 are the two regular-Cauchy tail schedules whose common completion-facing tail route is being fused;*
2. *μ_0 and μ_1 are the two modulus rows read from the source schedules;*
3. *τ is the shared tail threshold produced from the two schedules and the two modulus rows;*
4. *V is the modulus comparison row witnessing that both source moduli are served by the same threshold τ;*
5. *W_0 and W_1 are the selector-window rows opened from τ for the two source schedules;*
6. *D_0 and D_1 are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows attached to the two selected windows;*
7. *Q_0 and Q_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the windows and dyadic tolerances;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the shared threshold, modulus comparison, windows, tolerances, and readbacks have been displayed;*
9. *H records componentwise* hsame *transport for the schedules, moduli, threshold, comparison, windows, tolerances, readbacks, seal, provenance, and local name;*
10. *C records displayed* Cont *routes from the two schedules to their modulus rows, from the modulus rows to τ, V, from τ, V to the windows, from the windows to the dyadic tolerances and regular readbacks, and from the readbacks to the Real seal;*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFusion}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailModulusFusion}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, host real equality, selected external limit, independent per-source Real seal, hidden modulus comparison, or ambient completeness coordinate.

Theorem 1024.2 (Cauchy tail modulus fusion NameCert obligations)**.** *For the carrier of Definition 1024.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailModulusFusion}^{\uparrow}}$ *surface consists exactly of carrier admission for F, paired tail-schedule exposure through T_0, T_1, modulus exposure through μ_0, μ_1, shared-threshold exposure through τ, modulus comparison through V, selector-window service through W_0, W_1, dyadic tolerance readback through D_0, D_1, regular rational readback through Q_0, Q_1, Real seal ordering through E, componentwise stability through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyTailModulusFusionUp.CauchyTailModulusFusionCarrier namecert obligations`

Proof. Project the displayed carrier. The packet lists the two tail schedules, two modulus rows, shared threshold, modulus comparison, selector windows, dyadic tolerances, regular readbacks, Real seal, transport, route, provenance, and local name as public coordinates.

The sibling tail and modulus routes supply the two source boundaries. The tail schedules are read through chapter 658, while the comparison row is read through chapter 885 and the shared-threshold service is read through chapter 983.

The finite observation rows then pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the $\mathsf{Real}^{\uparrow}$ seal is exposed. Since the carrier has no quotient stream, host real equality, selected external limit, hidden modulus comparison, or ambient completeness coordinate, these rows exhaust the local NameCert obligations. □

Theorem 1024.3 (Cauchy tail modulus fusion shared-tail exhaustion)**.** *Every accepted* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *packet routes a completion-facing Real seal through the displayed chain*

$$T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N.$$

No consumer of the packet can read E before the shared threshold τ, the modulus comparison V, both selector windows, both dyadic tolerance rows, and both regular readbacks have been displayed.

Lean4 ✓ `BEDC.Derived.CauchyTailModulusFusionUp.CauchyTailModulusFusionCarrier shared tail exhaustion`

Proof. Start from carrier admission in Definition 1024.1. The only Real-facing coordinate is E, and it is listed after the schedules, moduli, threshold, comparison, windows, tolerances, and readbacks.

Apply the NameCert obligations of Definition 1024.2. They assign every completion-facing read of E to the same displayed route and require the route to pass through the shared threshold and comparison rows before any selector-window read.

Finally use the absence clauses in the carrier. A route that bypasses $\tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1$ would require a private threshold, hidden comparison, independent Real seal, or ambient completeness coordinate, none of which is present in the packet. □

Theorem 1024.4 (Cauchy tail modulus fusion threshold comparison stability)**.** *For the displayed carrier*

$$F = (T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N)$$

of Definition 1024.1, every modulus-comparison read that reaches the $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the shared threshold* τ *and the comparison row* V *exposed by the sibling surfaces chapter 658, chapter 885, and chapter 983. The transport row* H*, replay row* C*, provenance row* P*, and local name row* N *preserve this factoring for both branches* $T_0, \mu_0, W_0, D_0, Q_0$ *and* $T_1, \mu_1, W_1, D_1, Q_1$*.*

Proof. Project the carrier coordinates. The only comparison coordinate is V, and it is listed between the shared threshold τ and the two selector-window rows W_0, W_1.

The sibling surfaces provide the source routes: regular tail data enter through T_0, T_1, modulus data enter through μ_0, μ_1, and the shared-threshold service exposes τ before either branch can select a window. The replay row C records exactly this order.

Finally use the exclusion clauses of the carrier. A comparison that bypasses τ or replaces V would require a hidden comparison row, an independent Real seal, a quotient stream, or an ambient completeness coordinate. None is a coordinate of F, so stability is forced by carrier projection. □

Theorem 1024.5 (Cauchy tail modulus fusion window readback route)**.** *For the displayed carrier*

$$F = (T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N),$$

every finite-window readback used by the $\mathsf{Real}^{\uparrow}$ *seal* E *is routed through the paired windows* W_0, W_1*, dyadic tolerance rows* D_0, D_1*, and regular rational rows* Q_0, Q_1 *displayed in the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *certificate surfaces. The rows* $T_0, T_1, \mu_0, \mu_1, \tau, V, H, C, P, N$ *remain public coordinates of the same packet and do not open any second readback channel.*

Proof. Begin with the window coordinates. The carrier lists exactly two selector windows, W_0 and W_1, and each window has its displayed dyadic tolerance and regular rational readback, namely D_0, Q_0 and D_1, Q_1.

The observation surfaces then determine the route. $\mathsf{StreamName}^{\uparrow}$ supplies the finite window names, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies D_0, D_1, $\mathsf{RegSeqRat}^{\uparrow}$ supplies Q_0, Q_1, and $\mathsf{Real}^{\uparrow}$ consumes the resulting readbacks only at E.

The remaining rows keep the route single-carrier. The schedules T_0, T_1, moduli μ_0, μ_1, threshold τ, comparison V, transport H, replay C, provenance P, and local name N are all projected from the same finite packet, so a second window-readback channel would have to be an unlisted coordinate. □

Theorem 1024.6 (Cauchy tail modulus fusion Real seal nonescape)**.** *For the displayed carrier*

$$F = (T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *seal row* E *has no escape route outside the shared-tail and modulus-fusion packet. Any consumer that reads* E *must also read the sibling-surface chain through* $T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1$ *and must keep the transport, replay, provenance, and naming rows* H, C, P, N *attached to that same carrier.*

Proof. The carrier has a single Real-facing coordinate, E. It is introduced only after the schedules, modulus rows, shared threshold, comparison row, selector windows, dyadic tolerances, and regular readbacks have been displayed.

The certificate surfaces prevent detachment. The route to E is replayed by C, transported by H, recorded by P, and named by N as part of the same $\mathsf{NameCert}_{\mathsf{CauchyTailModulusFusion}^{\uparrow}}$ packet.

If E escaped this packet, the consumer would need a quotient stream, a host real equality, a selected external limit, an independent per-source Real seal, or an ambient completeness coordinate. The carrier explicitly excludes each of these coordinates, so the Real seal remains internal to the displayed chain. □

Theorem 1024.7 (Cauchy tail modulus fusion public Real-completion export)**.** *Every downstream Real-completion consumer of an accepted* CauchyTailModulusFusion$^{\uparrow}$ *packet reads the public route through the displayed rows*

$$T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N.$$

The source rows are supplied by the three sibling surfaces RegularCauchyTailFusion$^{\uparrow}$, CauchyModulusRefinement$^{\uparrow}$, *and* CauchySharedTailModulus$^{\uparrow}$, *and the finite readback exits through the four L10 surfaces* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$. *This public surface exports no quotient stream, selected host limit, host equality, countable-choice row, or ambient Real-completeness principle.*

Proof. Start with the NameCert obligations of Definition 1024.2. They expose exactly the two schedules, two modulus rows, shared threshold, comparison row, paired windows, dyadic tolerances, regular readbacks, Real seal, transport, replay, provenance, and local name listed in the carrier.

Next use shared-tail exhaustion and threshold-comparison stability. Definition 1024.3 forces any Real-facing read of E through $\tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1$, while Definition 1024.4 keeps both branches on the same sibling-surface comparison route.

The finite readback is then fixed by Definition 1024.5. The route reads the StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ tolerances, RegSeqRat$^{\uparrow}$ readbacks, and Real$^{\uparrow}$ seal as one displayed public surface.

Finally apply Real-seal nonescape, Definition 1024.6. Since the packet has no coordinate for quotient streams, selected host limits, host equality, countable choice, or ambient Real completeness, every downstream Real-completion consumer receives only the public route above. □

Theorem 1024.8 (Cauchy tail modulus fusion mature readback coverage)**.** *Every downstream Real-completion consumer of an accepted* CauchyTailModulusFusion$^{\uparrow}$ *packet reads exactly the shared chain*

$$T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N.$$

The source rows are supplied by RegularCauchyTailFusion$^{\uparrow}$, CauchyModulusRefinement$^{\uparrow}$, CauchySharedTailModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$. *The consumer receives no second limit, quotient stream, host equality, hidden modulus comparison, countable-choice row, or ambient Real-completeness coordinate.*

Proof. Begin with the NameCert obligations of Definition 1024.2. They bind the consumer-facing packet to the displayed carrier rows $T_0, T_1, \mu_0, \mu_1, \tau, V, W_0, W_1, D_0, D_1, Q_0, Q_1, E, H, C, P, N$.

Next combine the shared-tail and comparison witnesses. The shared-tail exhaustion theorem forces every Real-facing read through the common threshold, windows, dyadic tolerances, and regular readbacks, while threshold-comparison stability keeps both branches attached to the three sibling surfaces.

The finite observation route is then fixed by the window-readback theorem. It preserves the StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ certificate surfaces as the route from selected windows to the Real seal.

Finally use Real-seal nonescape together with the public Real-completion export theorem. Since the packet contains no second limit, quotient stream, host equality, hidden modulus comparison, countable-choice row, or ambient Real-completeness coordinate, the downstream consumer receives exactly the fused public route and no external completion coordinate. □

Closure status for NameCert$_{\text{CauchyTailModulusFusion}^{\uparrow}}$.

Constructive story: A CauchyTailModulusFusion$^{\uparrow}$ packet is generated from two tail schedules, two modulus rows, one shared threshold, one modulus comparison row, paired selector windows, paired dyadic tolerances, paired RegSeqRat$^{\uparrow}$

readbacks, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.CauchyTailModulusFusionUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{CauchyTailModulusFusion}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1024.1, the five obligation rows in Definition 1024.2, Definition 1024.3, Definition 1024.4, Definition 1024.5, Definition 1024.6, the public export theorem Definition 1024.7, and the mature readback coverage theorem Definition 1024.8 bind the reusable shared-tail and modulus-fusion surface for downstream Real-completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyTailModulusFusion}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CauchyTailModulusFusionUp.CauchyTailModulusFusionCarrier\_namecert\_obligations`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient-stream equality, host real equality, selected external limits, or a public standard bridge.

Upgrade path: No further paper-side closure step remains; bridge checking remains a separate formal-axis task.

Chapter 1025

A Concrete Naming Certificate for $\mathsf{ContinuationBisimulation}^{\uparrow}$

The $\mathsf{ContinuationBisimulation}^{\uparrow}$ packet realizes the behavioral-equivalence direction of chapter 9365 as a finite paired-simulation certificate. It is adjacent to the small-step trace packet of chapter 469, the terminal-read packet of chapter 657, and the bounded evaluator packet of chapter 818, but it stores a different public surface: two endpoint traces together with forward and backward simulation rows over the same endpoints.

Falsifiable prediction: An accepted $\mathsf{ContinuationBisimulation}^{\uparrow}$ packet with a forward simulation row but no backward simulation row fails the local NameCert obligations.

Independence witness: continuationtrace, continuationbigstep, continuationtracenormalform, continuationautomaton, finitetraceevaluator: none is bijective to $\mathsf{ContinuationBisimulation}^{\uparrow}$ because those siblings record traces, terminal reads, normal forms, automata, or bounded evaluator rows, while this carrier requires paired forward and backward simulation rows over the same endpoints.

Definition 1025.1 (ContinuationBisimulation carrier)**.** *A* $\mathsf{ContinuationBisimulation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (L, R, F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger, including its source, mark-and-tail, target, and trace rows;*
2. *R is the right finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger over the comparison endpoint;*
3. *F is the forward simulation row, reading each visible left trace step as a right trace step with matching endpoint classifier;*
4. *B is the backward simulation row, reading each visible right trace step as a left trace step with matching endpoint classifier;*
5. *E is the endpoint classifier row comparing the two source endpoints and the two target endpoints after the paired simulations have been read;*
6. *H stores componentwise* hsame *transports for the two trace ledgers, the forward and backward simulation rows, endpoint classifiers, replay rows, provenance, and local naming;*
7. *C stores the displayed* Cont *replay routes used to replay the left trace, the right trace, the forward simulation, the backward simulation, and the endpoint reads;*

8. *P is the* Pkg *provenance over chapter 9365, chapter 469, chapter 657, chapter 818,* BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationBisimulation}^{\uparrow}}$ *row.*

The carrier exposes a symmetric finite behavioral-equivalence packet. It contains no host evaluator, quotient behaviour class, unbounded trace semantics, terminal-result selector, or one-sided simulation shortcut.

Theorem 1025.2 (ContinuationBisimulation NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ContinuationBisimulation}^{\uparrow}}$ *surface for the carrier of Definition 1025.1 consists of seven finite obligations:*

1. *carrier admission for the displayed packet $Z = (L, R, F, B, E, H, C, P, N)$;*
2. *left trace admission for L through the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *rows;*
3. *right trace admission for R through the finite* $\mathsf{ContinuationTrace}^{\uparrow}$ *rows;*
4. *forward simulation exactness, requiring every visible step read from L to be matched by a displayed step read from R through F;*
5. *backward simulation exactness, requiring every visible step read from R to be matched by a displayed step read from L through B;*
6. *endpoint-classifier stability through E, H, C for the two source endpoints and the two target endpoints;*
7. *non-escape, stating that every accepted behavioral-equivalence read factors through $L, R, F, B, E, H, C, P, N$ and cannot consult a host evaluator, quotient behaviour class, unbounded trace semantics, terminal-result selector, or one-sided simulation row.*

Lean4 ✓ `BEDC.Derived.ContinuationBisimulationUp.TasteGate.ContinuationBisimulationNameCertObligations`

Proof. Carrier admission is projection from Definition 1025.1. The accepted packet displays exactly the two trace ledgers, the two simulation rows, the endpoint classifier, the transport row, the replay row, provenance, and the local naming row.

Read the two trace obligations through chapter 469. The rows L and R are finite continuation traces, so their visible steps are source, mark-and-tail, target, trace-ledger, and replay reads rather than host-level transition facts.

The simulation obligations are then paired. The forward row F accounts for every visible left step by a right-step read, and the backward row B accounts for every visible right step by a left-step read. Since both rows are coordinates of the same packet, the local NameCert can reject a one-sided packet before endpoint equivalence is exported.

Finally, E, H, C, P, N provide endpoint comparison, componentwise transport, replay, provenance, and local naming for the paired route. No coordinate remains for a quotient behaviour class, an evaluator graph, an unbounded trace semantics, a terminal-result selector, or a hidden one-sided simulation, so the seven obligations exhaust the packet. □

Theorem 1025.3 (ContinuationBisimulation non-escape)**.** *Every accepted* $\mathsf{ContinuationBisimulation}^{\uparrow}$ *consumer route factors through the displayed finite rows*

$$L, R, F, B, E, H, C, P, N.$$

In particular, a behavioral-equivalence consumer must read both the forward simulation row F and the backward simulation row B before the endpoint classifier E can be exported. The route cannot

be replaced by a host evaluator, a quotient of behaviours, an automaton-language equivalence, a bounded evaluator verdict, or a terminal-only big-step read.

Proof. Unpack the carrier of Definition 1025.1. The only trace data available to a consumer are the two finite ledgers L and R, and the only simulation data are the displayed rows F and B.

Use the NameCert obligations of Definition 1025.2. They force left-step reads through L, right-step reads through R, forward matching through F, backward matching through B, and endpoint comparison through E, H, C.

The surrounding siblings keep their own surfaces. chapter 657 reads terminal rows, chapter 818 records bounded evaluator rows, and chapter 482 records automaton transition rows. None supplies the missing opposite simulation row for this packet.

Therefore every accepted behavioral-equivalence read remains inside $L, R, F, B, E, H, C, P, N$. Removing either F or B changes the local NameCert surface and leaves no route by which the endpoint classifier may export symmetric equivalence. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationBisimulation}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuationBisimulation}^{\uparrow}$ packet is generated from left and right finite continuation ledgers, forward and backward matching rows, endpoint comparison, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The BHist carrier does not start from behavioral equivalence as a host relation; it builds the relation by exposing both finite ledgers and both matching directions. The classifier accepts a bisimulation read only when left steps, right steps, forward matches, backward matches, and endpoint rows all remain visible under the same transport and replay surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationBisimulation}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the continuation bisimulation carrier, its NameCert obligations, and the non-escape field-faithfulness argument.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuationBisimulation}^{\uparrow}}, \mathsf{unformalized})$

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.continuationBisimulationDeltaLedger

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient behavioral semantics, infinite coinductive trace equality, host evaluator equivalence, or one-sided simulation as full equivalence.

Upgrade path: To reach obligationClosure, formalize the two finite ledgers and prove that both matching rows are necessary for the endpoint classifier.

Chapter 1026

A Concrete Naming Certificate for $\mathsf{KleisliTrace}^{\uparrow}$

The $\mathsf{KleisliTrace}^{\uparrow}$ packet realizes the continuation-as-small-step direction of chapter 9365 as a finite trace surface for Kleisli computation. It sits between the certified Kleisli interface of chapter 485 and the terminal replay surface of chapter 683: the former supplies bind and return rows, while the latter supplies the trace-normal-form row that lets a consumer read the whole route as one finite continuation trace. The same packet also exposes the $\mathsf{Cat}^{\uparrow}$ hom rows consumed by categorical continuation users, so Kleisli bind and category composition read from the same displayed route instead of from two unrelated ledgers.

Falsifiable prediction: If the trace-normal-form row is removed, the packet cannot route both Kleisli bind and CategoryHomCarrier consumers through one finite continuation trace.

Independence witness: continuationmonad, continuationtracenormalform, category, generator-closure: $\mathsf{ContinuationMonad}^{\uparrow}$ supplies bind and return rows without a single exported trace ledger, $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ supplies terminal replay without Kleisli source and target bind slots, $\mathsf{Cat}^{\uparrow}$ supplies hom rows without monadic provenance, and generator closure supplies source saturation without the five-row local naming surface below.

Definition 1026.1 (KleisliTrace carrier)**.** *A* $\mathsf{KleisliTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, T, G, H, C, P, N)$$

whose displayed rows are:

1. *S is the source continuation computation row accepted by the* $\mathsf{ContinuationMonad}^{\uparrow}$ *surface;*
2. *B is the Kleisli bind row, including the return slot, middle continuation, and composed continuation read from the* $\mathsf{ContinuationMonad}^{\uparrow}$ *packet;*
3. *T is the finite* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *row that replays the same Kleisli computation as one terminal trace-normal-form route;*
4. *G is the* $\mathsf{Cat}^{\uparrow}$ *hom row, exposing the source object, target object, and morphism history consumed by categorical continuation users;*
5. *H is the componentwise* hsame *transport row for the monad row, trace-normal-form row, hom row, replay rows, provenance, and local name;*

6. *C stores the displayed* Cont *replay routes that connect the Kleisli bind read, trace replay, and category hom read;*
7. *P is the* Pkg *provenance over chapter 9365, chapter 485, chapter 683, chapter 61,* BHist, hsame, Cont, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KleisliTrace}^{\uparrow}}$ *row.*

The packet contains no host evaluator, no hidden category composition oracle, no selected normalizer, and no quotient of computations. Every exported computation read is a row of K.

Lean4 (definition) `BEDC.Derived.KleisliTraceUp.TasteGate.KleisliTraceUp`

Theorem 1026.2 (KleisliTrace NameCert obligation surface)**.** *For the carrier of Definition 1026.1, the local* $\mathsf{NameCert}_{\mathsf{KleisliTrace}^{\uparrow}}$ *surface consists of five rows:*

1. *carrier admission for the displayed packet $K = (S, B, T, G, H, C, P, N)$;*
2. *Kleisli source and bind readback through the* $\mathsf{ContinuationMonad}^{\uparrow}$ *row S, B;*
3. *trace-normal-form replay through the finite* $\mathsf{ContinuationTraceNormalForm}^{\uparrow}$ *row T and the displayed* Cont *routes C;*
4. *Category hom readback through G, with componentwise* hsame *transport through H;*
5. *provenance and local naming through P, N, together with non-escape from host evaluators, hidden category composition, selected normalizers, and quotient computation objects.*

These five rows are the obligation surface that a later Lean TasteGate must expose through both $\mathsf{Nontrivial\,KleisliTrace}^{\uparrow}$ *and* $\mathsf{FieldFaithful\,KleisliTrace}^{\uparrow}$ *instances.*

Lean4 ✓ `BEDC.Derived.KleisliTraceUp.TasteGate.KleisliTraceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1026.1. The accepted packet exposes exactly S, B, T, G, H, C, P, N.

Read the Kleisli side through chapter 485. The rows S and B display the source computation, return slot, middle continuation, and composed bind route before any consumer asks for a trace.

Read the trace side through chapter 683. The row T and replay routes C show that the same bind route is consumed as one terminal finite trace-normal-form packet.

Finally read the categorical side through $\mathsf{Cat}^{\uparrow}$. The hom row G gives the source, target, and morphism surface consumed by category users, while H, P, N provide transport, provenance, and local naming. Since the carrier has no coordinate for a host evaluator, hidden category composition oracle, selected normalizer, or quotient computation object, the five rows exhaust the local certificate surface. □

Theorem 1026.3 (KleisliTrace common trace route)**.** *Every accepted* $\mathsf{KleisliTrace}^{\uparrow}$ *consumer route for Kleisli bind or for a* $\mathsf{Cat}^{\uparrow}$ *hom read factors through the same finite rows*

$$S, B, T, G, H, C, P, N.$$

The Kleisli consumer reads S, B, T, C, H, P, N; the category consumer reads G, T, C, H, P, N. Neither consumer can replace the displayed trace-normal-form row by a host evaluator, hidden composition oracle, selected normalizer, or quotient computation object.

Proof. Start with the carrier projection. Both consumer routes have access only to S, B, T, G, H, C, P, N.

For the Kleisli consumer, apply Definition 1026.2. The source and bind rows are read through S, B, and the route is replayed through T, C before provenance and local naming are exported.

For the categorical consumer, use the same obligation surface. The hom row G is read together with the trace-normal-form row T, the replay row C, transport row H, provenance row P, and local name N.

The excluded alternatives have no coordinate in the accepted packet. Thus the two consumers share the same finite continuation trace rather than importing a second evaluator, composition, normalization, or quotient surface. □

Theorem 1026.4 (KleisliTrace trace-normal-form necessity)**.** *An accepted* $\mathsf{KleisliTrace}^{\uparrow}$ *packet cannot export both its Kleisli bind read and its* $\mathsf{Cat}^{\uparrow}$ *hom read after removing the trace-normal-form row* T*. The finite replay row* T *is the common surface that identifies the monadic bind route and the category hom route as one continuation trace.*

Proof. Assume the packet is accepted but omit the row T. The remaining rows still mention the Kleisli source and bind data S, B and the category hom data G, but they no longer contain the finite terminal replay surface consumed by both users.

Apply Definition 1026.2. The second row reads the monadic bind surface, the third row requires trace-normal-form replay through T, C, and the fourth row reads the category hom surface through G, H.

Apply Definition 1026.3. Its shared route for both consumers passes through T. Without that row, the two reads would need a second evaluator, hidden category composition, selected normalizer, or quotient computation object, each of which is excluded by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{KleisliTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KleisliTrace}^{\uparrow}$ packet is generated from a monadic bind row, trace-normal-form row, category-hom row, shared trace route, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier reads a computation trace by first exposing the bind surface, then the finite trace normal form, then the category-facing hom surface. The same trace route is consumed by both monadic and categorical consumers, while hsame preserves the displayed coordinates and Cont replays the finite handoff. The construction names a shared Kleisli trace, not hidden evaluator composition.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KleisliTrace}^{\uparrow}}$, seed)

Scope closed: This chapter binds the Kleisli trace carrier, obligation surface, common trace route, and route non-escape argument.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KleisliTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim hidden category composition, selected normalizers, quotient computations, or host evaluator equality.

Upgrade path: To reach obligationClosure, provide a checked carrier tying the monadic bind, trace-normal-form, and category-hom rows to one finite route.

Chapter 1027

A Concrete Naming Certificate for $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$

The $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ packet realizes the large-model inscription-point reading of chapter 9398 as a finite audit surface. It sits downstream of the inscription carrier of chapter 939, the activation-orbit carrier of chapter 967, the verified-output harness of chapter 976, and the route-audit sibling of chapter 1005. Its distinguishing role is to bind output, corpus provenance, verification channel, refusal ledger, transport, route, and local naming rows before a consumer may export a model output as an audited inscription.

Definition 1027.1 (LargeModelInscriptionAudit carrier)**.** *A* $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, C, V, R, T, K, H, P, N)$$

with the following displayed rows:

1. *O is the output-record row read from the large-model inscription point;*
2. *C is the corpus-provenance row recording the source-channel and distillation route for the output read;*
3. *V is the verification-channel row naming the finite checker, replay harness, policy check, proof check, or external certificate procedure applied to O;*
4. *R is the refusal-ledger row recording unsupported, unverifiable, out-of-scope, or rejected claims;*
5. *T records componentwise* hsame *transport for output, corpus, verification, refusal, route, provenance, and naming rows;*
6. *K records the displayed* Cont *route from inscription point through corpus provenance, verification channel, refusal ledger, package provenance, and local name;*
7. *H is the accepted harness row supplied by* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *when the output read is checker-facing;*
8. *P is the* Pkg *provenance over* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$, $\mathsf{LargeModelActivationOrbit}^{\uparrow}$, $\mathsf{LargeModelAuditChannel}$ $\mathsf{CorpusSupplyDistillation}^{\uparrow}$, $\mathsf{VerifiedOutputHarness}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LargeModelInscriptionAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for model consciousness, host semantic understanding, corpus-wide truth, hidden verifier state, private policy override, selected source oracle, or internally ratified universal closure.

Theorem 1027.2 (LargeModelInscriptionAudit NameCert obligations)**.** *For the carrier of Definition 1027.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelInscriptionAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for* $A = (O, C, V, R, T, K, H, P, N)$*, corpus-provenance exposure through* C*, verification-channel exposure through* V*, refusal-ledger exposure through* R*, componentwise transport through* T*, displayed route replay through* K*, checker-facing harness exposure through* H*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed packet. The rows $O, C, V, R, T, K, H, P, N$ are the full public inventory for the audit.

The corpus and verification obligations are separate. The row C records where the output read is carried from, while V records how the output is inspected before public export.

The refusal, transport, route, harness, provenance, and naming rows make the audit replayable. They force any consumer-facing read to keep unsupported claims, checker-facing status, and local provenance in the same finite packet. □

Theorem 1027.3 (LargeModelInscriptionAudit output verification boundary)**.** *An accepted* LargeModelInscriptionA *packet can export the output row* O *as audited inscription only after the verification-channel row* V *and refusal-ledger row* R *have been displayed. If* V *is absent, the packet may still contain an emitted output record, but it cannot export that record as internally ratified universal truth or as checker-facing accepted output.*

Proof. Project the carrier of Definition 1027.1. The output row O is present before the audit reads verification or refusal data.

The route row K orders the public read through C, V, R, H, P, N. Thus the output remains carried inscription until the verification channel and refusal ledger have both been exposed.

If V is removed, no carrier coordinate names the checker, replay harness, policy check, proof check, or external certificate procedure. The packet therefore lacks the row needed to distinguish an audited output from a bare emitted inscription or an internally asserted universal claim. □

Theorem 1027.4 (LargeModelInscriptionAudit corpus provenance non-escape)**.** *Every public corpus-provenance read of an accepted* $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ *packet factors through the displayed row* C *and the package row* P*. It cannot be replaced by a selected source oracle, corpus-wide truth predicate, hidden training-data selector, host semantic oracle, or model-internal proof of the output.*

Proof. The carrier contains one corpus-provenance coordinate, namely C, and one package-provenance coordinate, namely P. A consumer can read source-channel and distillation provenance only through those rows.

The sibling $\mathsf{CorpusSupplyDistillation}^{\uparrow}$ row supplies finite filtering and distillation provenance, while $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ supplies the carried-output boundary. Neither sibling adds an oracle for corpus-wide truth or host semantic interpretation to this packet.

Since the excluded objects have no coordinate in $A = (O, C, V, R, T, K, H, P, N)$, public export of corpus provenance is exhausted by C and P. □

Theorem 1027.5 (LargeModelInscriptionAudit TasteGate obligation)**.** *The Lean boundary for* $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ *must provide* Nontrivial $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ *and* FieldFaithful LargeModelInscrip

instances in TasteGate.lean *for the same displayed carrier. The nontrivial instance must exhibit an explicit* witness_pair *separating at least an output-with-verification packet from an output-without-verification packet, and the field-faithful instance must use* mk*-derived fields that preserve* $O, C, V, R, T, K, H, P, N$ *component by component.*

Lean4 ✓ BEDC.Derived.LargeModelInscriptionAuditUp.LargeModelInscriptionAuditTasteGate single carrier alignment

Proof. The carrier definition displays nine finite rows. A nontriviality witness must therefore separate two admissible packet shapes by an actual row difference, not by a hidden semantic predicate.

The verification-channel boundary of Definition 1027.3 supplies the required separation: one packet displays V as a checker-facing route, and the other lacks that displayed verification row.

Field faithfulness must then be componentwise over the same displayed rows. Using mk-derived fields keeps the Lean carrier aligned with the paper packet and prevents hidden verifier state, corpus truth, or internal ratification from entering through an unlisted coordinate. □

Theorem 1027.6 (LargeModelInscriptionAudit scoped carrier coverage)**.** *For an accepted* LargeModelInscriptionAu *packet* $A = (O, C, V, R, T, K, H, P, N)$*, every public scoped read factors through exactly the displayed output row* O*, corpus-provenance row* C*, verification-channel row* V*, refusal-ledger row* R*,* hsame*-transport row* T*,* Cont*-route row* K*, harness row* H*,* Pkg*-provenance row* P*, and local* NameCert *row* N*. Sibling support is limited to* LargeModelInscriptionPoint$^{\uparrow}$*,* LargeModelActivationOrbit$^{\uparrow}$*,* LargeModelAuditChannel$^{\uparrow}$ CorpusSupplyDistillation$^{\uparrow}$*,* VerifiedOutputHarness$^{\uparrow}$*, and* InscriptionRouteAudit$^{\uparrow}$*. No hidden verifier state, corpus-wide truth predicate, model consciousness, or host semantic oracle is admitted by the scoped carrier.*

Proof. The carrier definition gives the complete finite BHist packet $A = (O, C, V, R, T, K, H, P, N)$. Each public read must therefore project one of those displayed rows, and the NameCert obligation theorem identifies carrier admission, corpus provenance, verification channel, refusal ledger, transport, route replay, harness exposure, package provenance, and local naming as the exact local certificate surface.

The output-verification boundary forces checker-facing export through V and R, while the corpus-provenance non-escape theorem forces source and distillation reads through C and P. These two obligations rule out a hidden verifier coordinate, a selected source oracle, a corpus-wide truth predicate, and a host semantic oracle.

The TasteGate obligation aligns the Lean carrier with the same componentwise rows. Since field faithfulness is witnessed over the mk-derived coordinates $O, C, V, R, T, K, H, P, N$, the scoped read cannot gain model consciousness, internal ratification, or private semantic state through any unlisted carrier field. □

Falsifiable prediction: A LargeModelInscriptionAuditUp consumer that omits the verification-channel row cannot export model output as internally ratified universal truth.

Independence witness: largemodelactivationorbit, largemodelauditchannel, corpussupplydistillation, verifiedoutputharness, inscriptionrouteaudit: the combined output/provenance/verification carrier is not bijective to any one sibling because activation orbit lacks the audit verification channel, audit channel lacks the corpus-provenance and route-audit combination, corpus-supply distillation lacks the accepted output harness, verified output harness lacks the standalone corpus-provenance audit row, and inscription route audit lacks the large-model output and corpus rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LargeModelInscriptionAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ packet is generated from output, corpus-provenance, verification-channel, refusal-ledger, transport, route, harness, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LargeModelInscriptionAudit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1027.1, Definition 1027.2, Definition 1027.3, Definition 1027.4, Definition 1027.5, and Definition 1027.6 specify the finite large-model inscription audit carrier, its verification-channel boundary, and its scoped carrier coverage.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LargeModelInscriptionAudit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LargeModelInscriptionAuditUp.LargeModelInscriptionAuditTasteGate\_single\_carrier\_alig`
Lean4 ✓ `BEDC.Derived.LargeModelInscriptionAuditUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close model consciousness, corpus-wide truth, host semantic understanding, hidden verifier correctness, Real completeness, or bridge checking for $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$.

Upgrade path: No further theory-axis upgrade remains for the displayed $O, C, V, R, T, K, H, P, N$ rows; bridge checking remains a separate verification-axis obligation.

Chapter 1028

A Concrete Naming Certificate for $\mathsf{RuntimeTruthSeparation}^{\uparrow}$

The $\mathsf{RuntimeTruthSeparation}^{\uparrow}$ packet realizes the reflection-limit boundary of chapter 9461 as a finite NameCert surface that keeps structural CIC* self-description separate from host runtime and truth-validation requests. It sits beside the reflection-limit boundary of chapter 586, the runtime-candidate boundary of chapter 591, the finite trace evaluator of chapter 818, the ground-loop boundary of chapter 674, and the quotient-soundness boundary of chapter 634. Those siblings expose separate structural, runtime, evaluator, ground, and quotient boundaries; this packet binds the two refusal axes in one local carrier so that a downstream consumer can ask for structure while runtime execution and truth validation remain outside BEDC.

Definition 1028.1 (Runtime/truth separation carrier)**.** *A* $\mathsf{RuntimeTruthSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, E, T, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the structural description row for the finite* CIC* *self-description accepted by the reflection-limit packet;*
2. *E is the runtime-request row, recording a consumer request to execute, step, normalize, or validate the structural description by an external host runtime;*
3. *T is the truth-validation request row, recording a consumer request for a truth predicate, host validation oracle, global truth classifier, or semantic correctness judgment over the same structural description;*
4. *L is the refusal ledger, routing every request in E or T back to the finite packet unless the consumer reads only the structural row S and its displayed bookkeeping;*
5. *H records componentwise* hsame *transport for the structural description, runtime-request, truth-validation request, refusal ledger, continuation, provenance, and local naming rows;*
6. *C records displayed* Cont *routes for reading S, E, T, L, H, P, N without adding an evaluator, truth predicate, or validation oracle;*
7. *P is the* Pkg *provenance over* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$, $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$, $\mathsf{FiniteTraceEvaluator}^{\uparrow}$, $\mathsf{GroundLoopBoundary}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

8. N is the local $\mathsf{NameCert}_{\mathsf{RuntimeTruthSeparation}^{\uparrow}}$ *row.*
The carrier exports structural self-description and its refusal ledger only. It contains no runtime executor, no truth predicate, no host validation oracle, and no off-packet evaluator.

Lean4 ✓ `BEDC.Derived.RuntimeTruthSeparationUp.RuntimeTruthSeparationTasteGate single carrier alignment`

Theorem 1028.2 (Runtime/truth separation non-escape)**.** *Every accepted* $\mathsf{RuntimeTruthSeparation}^{\uparrow}$ *packet* $R = (S, E, T, L, H, C, P, N)$ *exposes structural* CIC^* *data only through* S, H, C, P, N *and routes runtime or truth-validation requests only through* E, T, L*. It exports no runtime executor, truth predicate, host validation oracle, or off-packet evaluator.*

Proof. Project the carrier of Definition 1028.1. The row S is the only structural self-description row, while E and T are request rows rather than execution or truth witnesses.

The refusal ledger L is the only row that consumes those requests. A runtime read must pass through E and then L; a truth-validation read must pass through T and then L.

Finally inspect the bookkeeping rows. The transport row H, continuation row C, provenance row P, and local name N preserve and route the displayed finite packet. Since none of these coordinates contains an executor, truth predicate, host validation oracle, or evaluator, every accepted consumer read remains inside the structural row or the refusal ledger. □

Falsifiable prediction: Any accepted RuntimeTruthSeparationUp consumer requesting runtime execution or a truth predicate is routed to the refusal ledger unless it stays inside the displayed structural rows.

Independence witness: reflectionlimitboundary, runtimecandidateboundary, finitetraceevaluator, groundloopboundary, quotientsoundnessboundary: none is bijective because none carries both runtime-request and truth-validation refusal rows with the structural self-description row in one packet.

The Lean taste-gate file is `lean4/BEDC/Derived/RuntimeTruthSeparationUp/TasteGate.lean`. It must provide explicit `Nontrivial RuntimeTruthSeparationUp` and `FieldFaithful RuntimeTruthSeparation` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RuntimeTruthSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{RuntimeTruthSeparation}^{\uparrow}$ packet is generated from structural description, runtime-request, truth-validation request, refusal-ledger, transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuntimeTruthSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 1028.1 and Definition 1028.2 specify the finite carrier separating structural self-description from runtime execution and truth validation.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuntimeTruthSeparation}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RuntimeTruthSeparationUp.RuntimeTruthSeparationTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide Lean verification, bridge checking, a runtime executor, a truth predicate, a host validation oracle, quotient soundness, or an off-packet evaluator.

Upgrade path: Scoped closure requires checked carrier admission, refusal-ledger exactness, non-escape, nontriviality, and field faithfulness over the same packet.

Chapter 1029

A Concrete Naming Certificate for KernelNormalizationAuditJoin$^{\uparrow}$

The KernelNormalizationAuditJoin$^{\uparrow}$ packet realizes the audit-witness route of chapter 9364 as a finite join over kernel acceptance, axiom-query ancestry, normalization-audit replay, and subject-reduction boundary rows. It sits beside the kernel acceptance witness of chapter 547, the normalization audit packet of chapter 750, the subject-reduction boundary of chapter 721, and the closed-normal consistency packet of chapter 734. Those siblings expose one side of the route; this packet records the simultaneous join needed before a downstream MetaCIC or Real-and-Completion consumer may treat an accepted declaration as replayed normalization-audit evidence.

Definition 1029.1 (Kernel-normalization audit join carrier)**.** *A* KernelNormalizationAuditJoin$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, A, Q, N, S, B, R, H, C, P, L)$$

with the following displayed rows:

1. *K is the kernel-stamp row supplied by* KernelAcceptanceAuditWitness$^{\uparrow}$*, including the generated candidate, kernel candidate, accepted declaration, and environment ledger;*
2. *A is the* AxiomQueryLedger$^{\uparrow}$ *row that records the transitive dependency query, forbidden-dependency comparison, and verdict for the accepted declaration named by K;*
3. *Q is the normalization-audit replay row read through* MetaCICNormalizationAudit$^{\uparrow}$*, containing candidate evidence, closed-term projection, finite strong-normalization scope, and replay provenance;*
4. *N is the closed-normal consistency row read through* ClosedNormalConsistency$^{\uparrow}$ *when the replay route reaches a beta-strong normal false endpoint;*
5. *S is the subject-reduction boundary row read through* MetaCICSubjectReductionBoundaryUp*, keeping the four discharge obligations and the checked obstruction visible;*
6. *B is the non-escape boundary row saying that acceptance, query cleanliness, normalization replay, and closed-normal consistency remain separate from kernel correctness, full closed subject reduction, host evaluation, or ambient completeness;*
7. *R is the route-join ledger recording that the acceptance route, axiom-query route, normalization replay route, closed-normal route, and subject-reduction boundary route are consumed as one finite packet;*

8. *H records componentwise* hsame *transport for $K, A, Q, N, S, B, R, C, P, L$;*
9. *C records displayed* Cont *replay for accepted-declaration ancestry, axiom-query ledger reads, normalization-audit replay, closed-normal handoff, and boundary inspection;*
10. *P is the* Pkg *provenance over* KernelAcceptanceAuditWitness$^{\uparrow}$, AxiomQueryLedger$^{\uparrow}$, MetaCICNormalizationAudit$^{\uparrow}$, ClosedNormalConsistency$^{\uparrow}$, MetaCICSubjectReductionBoundaryUp, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *L is the local* NameCert$_{\text{KernelNormalizationAuditJoin}^{\uparrow}}$ *row.*

The carrier has no coordinate for certifying the Lean kernel, proving full closed subject reduction, manufacturing candidate evidence, erasing axiom-query ancestry, selecting a host evaluator, or turning closed-normal consistency into ambient consistency.

Lean4 ✓ `BEDC.Derived.KernelNormalizationAuditJoinUp.KernelNormalizationAuditJoinTasteGate single carrier alignment`

Theorem 1029.2 (Kernel-normalization audit join NameCert obligations)**.** *For the carrier of Definition 1029.1, the local* NameCert$_{\text{KernelNormalizationAuditJoin}^{\uparrow}}$ *surface consists exactly of carrier admission for $J = (K, A, Q, N, S, B, R, H, C, P, L)$, kernel-stamp exposure through K, axiom-query exposure through A, normalization-audit replay through Q, closed-normal handoff through N, subject-reduction boundary exposure through S, non-escape through B, route-join ledger exposure through R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.KernelNormalizationAuditJoinUp.KernelNormalizationAuditJoinPacket namecert obligations`

Proof. Project the carrier of Definition 1029.1. The rows K, A, Q, N, S are the only substantive audit rows: kernel acceptance, axiom-query ancestry, normalization replay, closed-normal endpoint, and subject-reduction boundary.

The join ledger R and continuation row C determine how a consumer reads those rows together. The acceptance route reaches the accepted declaration before A can classify its ancestry, and the normalization route reaches Q before N or S can be consumed as consistency-facing boundary data.

The transport, provenance, and local naming rows preserve the same finite inventory. Since J contains no further coordinate, the displayed rows exhaust the local NameCert obligations. □

Theorem 1029.3 (Kernel-normalization audit join non-escape)**.** *Every accepted* KernelNormalizationAuditJoin$^{\uparrow}$ *packet routes normalization-audit replay through accepted declaration ancestry and axiom-query ledger rows before any closed-normal or subject-reduction boundary read is exported. The packet cannot export Lean kernel correctness, replay success without the kernel-stamp row, dependency cleanliness without the axiom-query row, evidence-free global normalization, full closed subject reduction, host evaluation, quotient conversion, or ambient completeness.*

Proof. Begin with the kernel-stamp and axiom-query side. The rows K and A are distinct coordinates, so acceptance of the declaration and classification of its dependency ancestry cannot be read from either row alone.

Next read the normalization and consistency side. The normalization-audit row Q supplies only the candidate-mediated replay surface, while N supplies the closed-normal endpoint and S keeps the subject-reduction obstruction visible.

Finally inspect the boundary row B together with R, H, C, P, L. These rows replay and name the finite join without adding kernel correctness, hidden normalization evidence, closed subject reduction, host evaluation, quotient conversion, or ambient completeness. Hence every exported read remains inside the displayed packet. □

Falsifiable prediction: If the kernel-stamp row is removed, the packet cannot route normalization-audit replay through accepted declaration ancestry and axiom-query ledger rows.

Independence witness: kernelacceptanceauditwitness, metacicnormalizationaudit, metacicsubjectreductionboundary, closednormalconsistency: none is bijective because the join carries the simultaneous kernel-stamp, normalization replay, obstruction boundary, and non-escape rows, while each sibling carries only one side of that route.

The Lean follow-up for $\mathsf{KernelNormalizationAuditJoin}^{\uparrow}$ lands in `lean4/BEDC/Derived/KernelNormalizationAu` It must provide `Nontrivial` and `FieldFaithful` instances with an explicit `witness_pair` and field injection proofs for the displayed rows $K, A, Q, N, S, B, R, H, C, P, L$.

Closure status for $\mathsf{NameCert}_{\mathsf{KernelNormalizationAuditJoin}^{\uparrow}}$.

Constructive story: A $\mathsf{KernelNormalizationAuditJoin}^{\uparrow}$ packet is generated from kernel-stamp, axiom-query ledger, normalization-audit replay, closed-normal handoff, subject-reduction boundary, non-escape, route-join, transport, continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelNormalizationAuditJoin}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1029.1, Definition 1029.2, and Definition 1029.3 specify the finite join over acceptance ancestry, axiom-query ledger, normalization replay, closed-normal handoff, and subject-reduction boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelNormalizationAuditJoin}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, Lean kernel correctness, full closed subject reduction, evidence-free global normalization, host evaluation, quotient conversion, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, route-join exactness, non-escape, nontriviality, and field faithfulness over the same packet.

Chapter 1030

A Concrete Naming Certificate for FinitePrefixMachine$^{\uparrow}$

The FinitePrefixMachine$^{\uparrow}$ packet realizes the machine-specific bounded-prefix reading of chapter 9365. It sits between the finite recognizer surface of chapter 1016, the continuation-transition surface of chapter 482, the machine-code surface of chapter 280, and the finite trace replay surface of chapter 469. Its distinguishing datum is that the machine row, state row, transition row, finite prefix, endpoint, acceptance, replay, and local naming rows are all read as one displayed packet.

Falsifiable prediction: Removing the prefix-bound row permits a terminal read not backed by a displayed finite machine prefix, which the local classifier rejects.

Independence witness: finiteprefixautomaton, continuationautomaton, turingmachine, continuationtrace: FinitePrefixMachineUp is not bijective to any sibling because it carries machine state, transition, finite prefix, endpoint, acceptance, replay, and local naming rows in one packet.

Theorem 1030.1 (FinitePrefixMachine carrier)**.** *A* FinitePrefixMachine$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, Q, T, F, E, A, H, C, N)$$

with the following displayed coordinates:

1. *M is the finite machine row, including the machine identifier and its displayed input alphabet surface;*
2. *Q is the current state and finite state-ledger row read by the bounded prefix;*
3. *T is the transition-row ledger whose entries expose source state, input mark, target state, and the displayed* Cont *witness for the step;*
4. *F is the finite prefix-bound row, listing the machine input prefix and the visible prefix positions admitted for replay;*
5. *E is the endpoint row reached after the displayed prefix has been replayed;*
6. *A is the acceptance row read from the endpoint and the machine acceptance ledger;*
7. *H stores componentwise* hsame *transports for machine identity, state rows, transition rows, prefix positions, endpoint rows, acceptance rows, replay rows, and local naming;*
8. *C stores the displayed* Cont *replay routes for every accepted transition in the finite prefix and for the endpoint read;*

9. N is the local $\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}$ *row.*

The carrier exposes one machine-specific bounded continuation prefix. It contains no language quotient, no host evaluator, no unlisted transition rule, no infinite trace semantics, and no terminal read outside the displayed finite prefix.

Lean4 ✓ `BEDC.Derived.FinitePrefixMachineUp.FinitePrefixMachineTasteGate single carrier alignment`

Proof. The packet is read by projection from the finite BHist carrier. The displayed tuple names the machine row, the state ledger, the transition ledger, the prefix-bound row, endpoint, acceptance, transports, replay routes, and local naming row.

The neighbouring surfaces explain why each coordinate is public. chapter 1016 supplies the bounded prefix-run shape, chapter 482 supplies the transition-as-continuation shape, chapter 280 supplies the machine row, and chapter 469 supplies finite replay.

Since no other coordinate is present, every accepted read is confined to $M, Q, T, F, E, A, H, C, N$. □

Theorem 1030.2 (FinitePrefixMachine NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}$ *surface for the carrier of Definition 1030.1 consists of seven finite obligations:*

1. *carrier admission for the displayed packet $P = (M, Q, T, F, E, A, H, C, N)$;*
2. *machine binding, stating that Q, T, F, E, A are read as rows of the displayed machine row M;*
3. *prefix-bound exposure for the finite prefix row F before any endpoint read is accepted;*
4. *transition exactness for every visible prefix step through the displayed ledger T and the corresponding* Cont *route in C;*
5. *endpoint and acceptance exactness through E and A;*
6. *componentwise stability through H and replay through C;*
7. *non-escape, stating that no accepted terminal read may consult a host evaluator, an unlisted transition rule, a language quotient, an infinite trace, or a terminal row not backed by F.*

Proof. Carrier admission is Definition 1030.1. It gives the finite machine packet and names every row on the local certificate surface.

The machine-binding and prefix-bound obligations are read from M, Q, T, F, E, A. The transition ledger T is meaningful only as a row of the machine M, and the endpoint row E can be read only after the finite prefix-bound row F has displayed the positions being replayed.

Transition exactness, endpoint exactness, and acceptance exactness are then local checks over the same packet. Each accepted step uses a row of T and a replay route in C; the endpoint row E is the terminal row of that finite replay; and A reads acceptance from the endpoint and the displayed machine acceptance ledger.

Finally, H, C, N transport, replay, and name only the displayed rows. Because the carrier has no coordinate for a host evaluator, unlisted rule, quotient language, infinite trace, or off-prefix terminal row, the non-escape obligation follows from the packet boundary. □

Theorem 1030.3 (FinitePrefixMachine prefix determinacy)**.** *For an accepted* $\mathsf{FinitePrefixMachine}^{\uparrow}$ *carrier $P = (M, Q, T, F, E, A, H, C, N)$, the displayed machine row M, transition ledger T, and finite prefix-bound row F determine the endpoint row E and acceptance row A inside the packet. If another accepted endpoint and acceptance pair E', A' is admitted over the same M, Q, T, F, H, C, N, then E is* hsame *to E' and A is* hsame *to A' through the displayed transport rows in H.*

Proof. Begin with the prefix-bound row F. It fixes the visible input prefix and the prefix positions admitted for replay, so a terminal read cannot skip to an endpoint outside that finite row.

At each prefix position, the transition-exactness obligation of Definition 1030.2 forces the next state row to be the target of the displayed transition row in T for the machine M. The replay route in C records the same step as a Cont read; it does not add a second transition source.

After the last displayed prefix position, endpoint exactness forces E to be the row reached by that replay, and acceptance exactness forces A to be read from E and the acceptance surface of M.

Apply the componentwise transport rows in H to any second accepted pair E', A' over the same machine, transition, and prefix rows. The packet therefore identifies the two endpoint rows and the two acceptance rows internally, without using a host evaluator or an unlisted transition rule. □

Lean4 ✓ `BEDC.Derived.FinitePrefixMachineUp.FinitePrefixMachineCarrier prefix determinacy`

Theorem 1030.4 (FinitePrefixMachine non-escape)**.** *Every accepted* $\mathsf{FinitePrefixMachine}^{\uparrow}$ *consumer route factors through*

$$M, Q, T, F, E, A, H, C, N.$$

In particular, a terminal read must pass through the finite prefix-bound row F, the transition ledger T, and the replay row C before the endpoint row E and acceptance row A can be exported. The route cannot be replaced by a finite-prefix automaton quotient, a continuation automaton behaviour class, a Turing-machine host evaluator, or an unbounded continuation trace.

Proof. Unpack the carrier of Definition 1030.1. The only machine-specific data available to a consumer are M, Q, T, F, E, A together with transport, replay, and naming rows.

Use the NameCert obligations of Definition 1030.2. They require machine binding, prefix-bound exposure, transition exactness, endpoint exactness, and acceptance exactness before a terminal read is exported.

The sibling chapters keep their own surfaces. chapter 1016 records bounded automaton runs, chapter 482 records continuation-labelled automaton transitions, chapter 280 records machine-code rows, and chapter 469 records finite replay traces. None supplies the missing combined packet for this machine-specific prefix read.

Therefore every accepted terminal route remains inside $M, Q, T, F, E, A, H, C, N$. □

Lean4 ✓ `BEDC.Derived.FinitePrefixMachineUp.FinitePrefixMachineCarrier nonescape`

Theorem 1030.5 (FinitePrefixMachine public export)**.** *The public export of an accepted* $\mathsf{FinitePrefixMachine}^{\uparrow}$ *packet is exactly the machine-specific bounded prefix surface*

$$M \longmapsto Q \longmapsto T \longmapsto F \longmapsto E \longmapsto A$$

together with componentwise hsame *transport,* Cont *replay, and the local* $\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}$ *row. The export supplies a displayed bounded machine-prefix read for downstream consumers, but it does not publish language equality, behavior-class equality, unbounded halting, total host execution, or a trace outside the finite prefix row.*

Proof. Carrier admission supplies the displayed packet $P = (M, Q, T, F, E, A, H, C, N)$. The public machine-facing route is the finite sequence from machine row through state, transition, prefix bound, endpoint, and acceptance rows.

Prefix determinacy is Definition 1030.3. It makes the endpoint and acceptance rows functions of the displayed machine, transition, and prefix rows inside the local packet.

Non-escape is Definition 1030.4. It prevents any downstream consumer from replacing the finite prefix route by a host evaluator, behaviour quotient, language quotient, unbounded halting claim, or hidden trace oracle.

Thus the exported surface is exactly the displayed bounded machine-prefix read, together with the transport, replay, and local naming rows that certify it. □

The Lean file `Derived/FinitePrefixMachineUp/TasteGate.lean` will provide $\mathsf{Nontrivial\ FinitePrefixMachine}^{\uparrow}$ and $\mathsf{FieldFaithful\ FinitePrefixMachine}^{\uparrow}$ instances for the displayed rows $M, Q, T, F, E, A, H, C, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePrefixMachine}^{\uparrow}$ packet is generated from a finite machine row, prefix word row, transition ledger, finite run row, endpoint row, acceptance row, componentwise hsame transport, displayed Cont replay, and local naming. The BHist carrier exposes a bounded machine-prefix computation: each transition step is read from the displayed ledger, the endpoint is determined by the same prefix, and the acceptance classifier receives only that finite endpoint surface. Transport preserves the row inventory, and continuation replay prevents a consumer from replacing the run with a host evaluator.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite-prefix machine carrier, prefix determinacy, non-escape, and exact exported surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePrefixMachine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim behavior quotients, language quotients, unbounded halting, hidden trace oracles, or host evaluator semantics.

Upgrade path: To reach obligationClosure, formalize the finite machine-prefix ledger and prove endpoint and acceptance determinacy for the displayed carrier.

Chapter 1031

ObserverLicensedComparisonUp NameCert construction

The $\mathsf{ObserverLicensedComparison}^{\uparrow}$ surface realizes the observer-history identity vision of chapter 9418 as a finite licensed comparison carrier. It sits between the single-history packet of chapter 479, the pairwise observation packet of chapter 477, and the classifier packet of chapter 986. Its public role is to carry the licensed observations, their hsame transports, and the classifier-output row together, without adding observer-substance or host equality.

Falsifiable prediction: Changing only unlicensed observer-substance rows leaves every public comparison read unchanged.

Independence witness: Nearest siblings: observerhsameclassifier, observerhistidentity, observerhistoryidentity, observersubjectboundary, observerfilter. The carrier is not bijective to any of them: it stores a reusable licensed-comparison packet with source histories, licensed observations, hsame transport, classifier output, and public routes, while the siblings either carry a single identity row, a pairwise observation row, a classifier-only output surface, a subject boundary, or a filter selector.

Definition 1031.1 (Observer licensed comparison carrier)**.** *An* $\mathsf{ObserverLicensedComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H_0, H_1, A, R, D, C, P, N)$$

with the following displayed rows:

1. *H_0 and H_1 are the observer-history rows being compared;*
2. *A is the licensed observation ledger, listing exactly the displayed mark,* Cont*,* ProbeBundle*,* SigRel*, and comparison reads allowed to affect the public result;*
3. *R records componentwise* hsame *transport for the histories, licensed observations, classifier-output row, route row, package row, and local naming row;*
4. *D is the finite classifier-output row, read only after the licensed ledger and transport rows are displayed;*
5. *C records the displayed* Cont *routes by which a consumer reads H_0, H_1, A, R, D, P, N;*
6. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* ProbeBundle*,* SigRel*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*,* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{ObserverHsameClassifier}^{\uparrow}$*, and* NameCert *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverLicensedComparison}^{\uparrow}}$ *row.*

The carrier accepts only comparisons whose public result factors through A, R, D, C, P, N. It has no coordinate for unlicensed observation, observer-substance, host equality, quotient history identity, or hidden classifier state.

Lean4 (definition) `BEDC.Derived.ObserverLicensedComparisonUp.ObserverLicensedComparisonUp`

Theorem 1031.2 (Observer licensed comparison NameCert obligations)**.** *For the carrier of Definition 1031.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverLicensedComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for $L = (H_0, H_1, A, R, D, C, P, N)$, licensed-observation exactness for A, componentwise* hsame *transport through R, classifier-output confinement through D, displayed consumer routes through C, package provenance through P, local naming through N, and non-escape from unlicensed observation, observer-substance, host equality, quotient identity, and hidden classifier state.*

Proof. Project the carrier. The displayed rows are exactly $H_0, H_1, A, R, D, C, P, N$, so carrier admission and local naming are finite BHist reads rather than host-side equality claims.

Licensed-observation exactness is the row A. A consumer may use a mark, continuation, probe bundle, signature relation, or prior comparison read only when that datum appears in the displayed licensed ledger and is then transported through R.

Classifier-output confinement is the row D read through C, P, N. The same route also records provenance through $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, and the sibling classifier surface of chapter 986.

The excluded objects have no carrier coordinate. Thus no accepted comparison can expose unlicensed observation, observer-substance, host equality, quotient identity, or hidden classifier state. □

Theorem 1031.3 (Observer licensed comparison non-escape)**.** *Every public consumer of an accepted* $\mathsf{ObserverLicensedComparison}^{\uparrow}$ *packet reads only the licensed observation ledger A, the componentwise* hsame *transports R, the classifier-output row D, the displayed* Cont *routes C, package provenance P, and local name row N. Changing any unlicensed observer-substance datum leaves the public comparison read unchanged.*

Proof. Start from the accepted packet $L = (H_0, H_1, A, R, D, C, P, N)$. The consumer-facing comparison rows are A, R, D, C, P, N; the two history rows are sources for the licensed ledger, not observer-substance fields.

Apply the NameCert obligations of Definition 1031.2. They force every public comparison read through the displayed licensed observations, hsame transports, classifier output, route, provenance, and local name.

No unlicensed observer-substance row is present in the carrier. A change confined to such a row changes none of A, R, D, C, P, N, so every public comparison read remains unchanged. □

The Lean `TasteGate.lean` development for $\mathsf{ObserverLicensedComparison}^{\uparrow}$ must provide explicit Nontrivial $\mathsf{ObserverLicensedComparison}^{\uparrow}$ and FieldFaithful $\mathsf{ObserverLicensedComparison}^{\uparrow}$ instances for the displayed licensed comparison packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverLicensedComparison}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverLicensedComparison}^{\uparrow}$ packet is generated from two observer-history rows, a licensed observation ledger, componentwise hsame transports, a classifier-output row, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverLicensedComparison}^{\uparrow}}$, seed)

Scope closed: Definition 1031.1, Definition 1031.2, and Definition 1031.3 specify the finite licensed-comparison packet and its unlicensed-observation non-escape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverLicensedComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full observer-history identity, observer-filter completeness, hidden observer state, quotient history identity, unbounded observation, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, licensed-observation exactness, classifier-output confinement, non-escape, nontriviality, and field faithfulness for the same packet.

Chapter 1032

ObserverBridgeLedgerUp NameCert construction

The $\mathsf{ObserverBridgeLedger}^{\uparrow}$ surface realizes the observer-history identity vision of chapter 9418 as a finite bridge ledger for observer-state consumers. It sits after the conscious-observer-state row of chapter 578, the no-subject boundary row of chapter 615, the multi-history configuration row of chapter 474, the observer-history identity row of chapter 477, and the locality-cell row of chapter 513. The ledger records those sibling routes together before any downstream consumer reads an observer-state bridge as a reusable NameCert surface.

Falsifiable prediction: Any accepted bridge-ledger consumer that bypasses the observer-state row or the no-subject boundary row leaves the ObserverBridgeLedgerUp carrier and cannot preserve NameCert routing.

Independence witness: Nearest siblings: consciousobserverstate, observersubjectboundary, multihistconfig, observercoordinateframe, finitewindowobserver. None is bijective to the bridge-ledger packet because none carries all bridge-ledger rows, no-subject boundary rows, and multi-history route rows together: consciousobserverstate has the current-state surface without the route ledger, observersubjectboundary has the no-subject refusal boundary without the state bridge, multihistconfig has the multi-history carrier without the observer-state row, observercoordinateframe has local-frame routing without the no-subject bridge ledger, and finitewindowobserver has bounded-window observation without the full multi-history route ledger.

Definition 1032.1 (Observer bridge ledger carrier)**.** *An* $\mathsf{ObserverBridgeLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, M, I, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *row naming the observer-state surface being bridged;*
2. *Q is an accepted* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *row certifying that the bridge has no separate observer-substance coordinate;*
3. *M is an accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite multi-history configuration available to the bridge;*
4. *I is an accepted* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *row naming the licensed observer-history identity surface;*

5. *L is an accepted* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *row through which the state, boundary, and multi-history rows are read locally;*
6. *R is the bridge-ledger route row, recording the finite order in which $S, Q, M, I, L, H, C, P, N$ may be consumed;*
7. *H stores componentwise* hsame *transport for the observer-state, no-subject boundary, multi-history, identity, locality, route, provenance, and local name rows;*
8. *C records the displayed* Cont *routes for every bridge-ledger consumer read;*
9. *P is the* Pkg *provenance over* $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverLocalityCell}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}$ *row.*

The carrier accepts an observer-state bridge only when the observer-state row, no-subject boundary row, multi-history row, identity row, locality-cell row, route ledger, transport row, continuation routes, provenance, and local naming row are all displayed together. It exports no hidden observer-substance, global observer frame, unrecorded history, host equality of observers, quotient history identity, or route outside the ledger.

Lean4 (definition) `BEDC.Derived.ObserverBridgeLedgerUp.ObserverBridgeLedgerUp`
Lean4 ✓ `BEDC.Derived.ObserverBridgeLedgerUp.ObserverBridgeLedgerTasteGate single carrier alignment`

Theorem 1032.2 (Observer bridge ledger NameCert obligations)**.** *For the carrier of Definition 1032.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (S, Q, M, I, L, R, H, C, P, N)$, observer-state anchoring through S, no-subject boundary anchoring through Q, multi-history configuration anchoring through M, observer-history identity anchoring through I, locality-cell anchoring through L, route-ledger exactness through R, componentwise* hsame *transport through H, displayed continuation replay through C, package provenance through P, local naming through N, and non-escape from hidden observer-substance, global observer frames, unrecorded histories, host equality of observers, quotient history identity, and routes outside the ledger.*

Proof. Project the accepted carrier to the ten displayed rows of Definition 1032.1. The observer-state, no-subject boundary, multi-history, observer-history identity, and locality-cell rows are visible before the route ledger is available.

Route-ledger exactness is the row R. A bridge consumer may read the sibling rows only in the displayed order recorded by R and only after componentwise transport through H and continuation replay through C.

The provenance row P and local naming row N bind the read to the five sibling certificates listed in the carrier. Since none of the excluded objects is a coordinate of the carrier, no accepted bridge-ledger read can export one. □

Theorem 1032.3 (Observer bridge ledger falsifiable prediction)**.** *Any accepted bridge-ledger consumer that bypasses either the observer-state row S or the no-subject boundary row Q leaves the* $\mathsf{ObserverBridgeLedger}^{\uparrow}$ *carrier and cannot preserve the local* $\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}$ *routing.*

Proof. The carrier admits a bridge only as $B = (S, Q, M, I, L, R, H, C, P, N)$. The route row R is typed as a ledger over both S and Q, and the local name row N certifies that same displayed packet.

If a consumer omits S, then the bridge no longer has an observer-state surface to carry. If it omits Q, then the read has no certified no-subject boundary and may no longer be interpreted as an observer-history bridge inside the vision of chapter 9418.

In either case the consumer is not a route through R, H, C, P, N for the displayed carrier, so the local NameCert route is not preserved. □

Theorem 1032.4 (Observer bridge ledger independence witness)**.** *The* $\mathsf{ObserverBridgeLedger}^{\uparrow}$ *carrier is not bijective to the sibling surfaces* $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{ObserverCoordinateFrame}^{\uparrow}$, *or* $\mathsf{FiniteWindowObserver}^{\uparrow}$.

Proof. Each listed sibling omits at least one bridge-ledger coordinate required by Definition 1032.1. The conscious-observer-state row lacks the no-subject boundary and route ledger. The observer-subject-boundary row lacks the current observer-state bridge and multi-history route. The multi-history configuration lacks the current-state and no-subject boundary rows.

The observer-coordinate-frame row supplies a local frame route, but it does not carry all bridge-ledger, no-subject boundary, and multi-history route rows together as one NameCert packet. The finite-window observer row supplies bounded observation, but not the full route ledger tying $S, Q, M, I, L, H, C, P, N$ together.

Thus deleting the bridge-ledger coordinates changes the public carrier, while adding them changes each sibling surface. No bijection with any one sibling preserves all displayed rows. □

Theorem 1032.5 (Observer bridge ledger sibling route)**.** *Every accepted* $\mathsf{ObserverBridgeLedger}^{\uparrow}$ *consumer reads the conscious-observer-state, no-subject boundary, multi-history configuration, observer-history identity, and locality-cell siblings through the single route*

$$S, Q, M, I, L \longrightarrow R, H, C, P, N.$$

No second route may reach those siblings through hidden observer state, a global observer frame, an unrecorded history row, host equality, quotient history identity, or an external route table.

Lean4 ✓ `BEDC.Derived.ObserverBridgeLedgerUp.ObserverBridgeLedger sibling route row separates`

Proof. The carrier contains exactly one route-ledger coordinate, namely R. That coordinate records how the five sibling rows S, Q, M, I, L may be consumed.

Componentwise transport and continuation replay are then supplied by H and C, while P and N record provenance and local naming. The remaining named objects are absent from the carrier, so they cannot provide a second route. □

The Lean carrier site `Derived/ObserverBridgeLedgerUp/TasteGate.lean` provides explicit Nontrivial and FieldFaithful instances for the displayed bridge-ledger packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverBridgeLedger}^{\uparrow}$ packet is generated from conscious-observer-state, no-subject-boundary, multi-history configuration, observer-history identity, observer-locality-cell, route-ledger, hsame transport, displayed Cont route, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1032.1, Definition 1032.2, Definition 1032.3, Definition 1032.4, and Definition 1032.5 specify the finite bridge-ledger carrier, its NameCert obligations, observer-state and no-subject bypass refutation, sibling independence, and single route through state, boundary, multi-history, identity, locality, transport, continuation, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverBridgeLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global observer frames, hidden observer-substance, unrecorded histories, quotient history identity, public export, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked TasteGate carrier admission, Nontrivial and FieldFaithful instances, route-ledger exactness, no-subject boundary preservation, and multi-history sibling routing for the same packet.

Chapter 1033

A Concrete Naming Certificate for $\mathsf{RealCompletionRouteAudit}^{\uparrow}$

The $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ packet realizes the route-audit surface for the Real-completeness diagonal and seal route of chapter 9238. It consumes the finite completion handoff of chapter 360, the limit-seal row of chapter 386, the $\mathsf{StreamName}^{\uparrow}$ window surface, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the primitive hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. The packet is an audit object for the displayed route, not a selected limit, quotient stream equality, host real equality, or ambient completeness principle.

Definition 1033.1 (Real completion route-audit carrier)**.** *A* $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, L, S, R, D, H, C, P, N, V)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *route row naming the finite completion handoff;*
2. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row reached from the same completion route;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window-service row opened before any route readback;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row exposed through S;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and precision row consumed by the displayed completion route;*
6. *H records componentwise* hsame *transports for the completion handoff, limit seal, window, readback, dyadic tolerance, route verdict, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes from S to R, from R and D to Q, and from Q to L;*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionRouteAudit}^{\uparrow}}$ *row;*
10. *V is the route-audit verdict row stating that the accepted consumer route is exhausted by the displayed finite packet.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, hidden tail selection, unbounded observation, or ambient completeness.

Theorem 1033.2 (Real completion route-audit NameCert obligations)**.** *For the carrier of Definition 1033.1, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionRouteAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (Q, L, S, R, D, H, C, P, N, V),$$

completion-route admission through Q*, limit-seal admission through* L*, StreamName window exposure through* S*, RegSeqRat readback through* R*, DyadicRatCore tolerance exposure through* D*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, local naming through* N*, and route-audit non-escape through* V*.*

Proof. Project the carrier of Definition 1033.1. The displayed packet exposes the completion handoff, limit seal, window service, readback, dyadic tolerance, transport, continuation, provenance, local name, and route verdict rows.

Read the route in dependency order. The $\mathsf{StreamName}^{\uparrow}$ row S opens the finite windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads those windows as regular-rational data, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the finite tolerance ledger. The completion row Q may then be consumed, and the limit-seal row L is reached only from that same displayed route.

The rows H, C, P, N, V preserve and audit this route. Since the carrier contains no quotient stream equality, host real equality, selected limit, hidden tail selector, unbounded observation, or ambient completeness coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealCompletionRouteAuditUp.RealCompletionRouteAuditTasteGate single carrier alignment`

Theorem 1033.3 (Real completion route-audit consumer boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ *packet factors through the finite route*

$$S, R, D, Q, L, H, C, P, N, V.$$

The consumer may inspect the displayed StreamName, RegSeqRat, DyadicRatCore, RealCauchyCompletion, CauchyLimitSeal, transport, continuation, provenance, naming, and audit-verdict rows, and it receives no route to quotient stream equality, host real equality, selected limits, or ambient completeness.

Proof. Start with the NameCert obligations. They identify the accepted packet with the ten displayed rows $Q, L, S, R, D, H, C, P, N, V$ and bind the completion-facing route to the same inventory.

The positive route is ordered by the displayed continuations. The window row S precedes the readback row R, the dyadic row D supplies the tolerance surface for the completion handoff Q, and the limit-seal row L is reached only after that handoff has been consumed.

Finally inspect the route-audit verdict V. It records that no consumer route leaves the finite packet for quotient stream equality, host real equality, selected limits, hidden tail selection, unbounded observation, or ambient completeness. Thus every accepted consumer read stays inside the displayed route. □

Theorem 1033.4 (Real completion route ordering)**.** *For every accepted* $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ *carrier*

$$A = (Q, L, S, R, D, H, C, P, N, V),$$

the displayed continuation row C *orders the positive route as*

$$S \to R, \qquad (R, D) \to Q, \qquad Q \to L.$$

No consumer may read Q before the window row S, the readback row R, and the dyadic tolerance row D have been exposed, and no consumer may read L before the completion handoff Q has been consumed.

Proof. Use the carrier definition to project the row C. By Definition 1033.1, C records exactly the displayed Cont routes from S to R, from R and D to Q, and from Q to L.

The NameCert obligations of Definition 1033.2 admit those continuations as part of the local naming surface. The consumer boundary of Definition 1033.3 then prevents any consumer from bypassing them through quotient stream equality, host real equality, selected limits, or ambient completeness. □

Theorem 1033.5 (StreamName finite window service)**.** *For every accepted* $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ *carrier $A = (Q, L, S, R, D, H, C, P, N, V)$, the row S is the unique StreamName service row available to the route audit. It exposes only the finite window data named by* $\mathsf{StreamName}^{\uparrow}$*, and every later readback through R must be fed by that displayed window row.*

Proof. Project S from the carrier. The definition of $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ names S as the $\mathsf{StreamName}^{\uparrow}$ window-service row opened before any route readback.

Apply the NameCert obligation theorem to bind the StreamName exposure to the same row S. The consumer boundary then says that every completion-facing read stays inside $S, R, D, Q, L, H, C, P, N, V$, so no hidden tail selector, unbounded observation, quotient stream equality, or host real equality can supply a second window service. □

Theorem 1033.6 (RegSeqRat readback over the displayed window)**.** *For every accepted* RealCompletionRouteAudit *carrier $A = (Q, L, S, R, D, H, C, P, N, V)$, the row R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed through S. Its regular-rational data are route data for the displayed StreamName window, not an equality of streams or a selected limit.*

Proof. The carrier definition assigns R to the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback row and places it after the StreamName window row S in the displayed route.

The NameCert obligations certify readback through R, and Definition 1033.4 identifies the route edge $S \to R$. Finally, the consumer boundary excludes quotient stream equality, host real equality, selected limits, and ambient completeness, so the readback remains finite route data over the displayed window. □

Theorem 1033.7 (DyadicRatCore tolerance exposure for the handoff)**.** *For every accepted* RealCompletionRouteAu *carrier $A = (Q, L, S, R, D, H, C, P, N, V)$, the row D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and precision ledger consumed by the completion handoff Q. The handoff may depend on R and D together, but it receives no ambient completeness principle and no host-level real equality.*

Proof. Project the dyadic row D and the completion row Q from the carrier. By definition, D is the finite tolerance and precision row consumed by the displayed completion route, while Q is the $\mathsf{RealCauchyCompletion}^{\uparrow}$ route row.

The NameCert obligations admit both rows and the continuation row C. The route-ordering theorem reads the relevant edge as $(R, D) \to Q$, and the consumer boundary blocks every route to ambient completeness, selected limits, quotient stream equality, or host real equality. □

Theorem 1033.8 (Completion handoff and limit-seal exposure)**.** *For every accepted* $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ *carrier $A = (Q, L, S, R, D, H, C, P, N, V)$, the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff Q and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *exposure L are reached only through the displayed rows*

$$S,\ R,\ D,\ H,\ C,\ P,\ N,\ V.$$

Thus the audit packet exposes the completion handoff and limit seal without adding quotient stream equality, host real equality, selected limits, hidden tail selection, unbounded observation, or ambient completeness.

Proof. Begin with the carrier and NameCert obligation theorems. They place the handoff Q and seal row L in the same finite packet as the StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, transports, continuations, provenance, local name, and verdict row.

Use the route-ordering theorem to reach Q only from the readback and dyadic rows and to reach L only from Q. The consumer boundary then exhausts all completion-facing reads by the displayed packet, so the omitted equality, choice, tail, observation, and completeness surfaces remain unavailable. □

Falsifiable prediction: Every accepted RealCompletionRouteAuditUp packet exposes a finite completion route and admits no consumer route to quotient stream equality, host real equality, selected limits, or ambient completeness.

Independence witness: realcauchycompletion, cauchylimitseal, streamname, regseqrat, dyadicratcore: the route-audit carrier is not bijective to any of these siblings because $\mathsf{RealCauchyCompletion}^{\uparrow}$ stores the completion handoff without the full audit verdict, $\mathsf{CauchyLimitSeal}^{\uparrow}$ stores the seal without the complete window-readback-tolerance route, $\mathsf{StreamName}^{\uparrow}$ stores finite windows without the completion and seal rows, $\mathsf{RegSeqRat}^{\uparrow}$ stores regular-rational readback without the route verdict, and $\mathsf{DyadicRatCore}^{\uparrow}$ stores tolerance rows without the completion handoff. The audit packet carries all five route objects with hsame, Cont, Pkg, NameCert, and verdict rows in one finite surface.

The Lean TasteGate boundary for this packet must provide `Nontrivial RealCompletionRouteAuditUp` and `FieldFaithful RealCompletionRouteAuditUp` instances for the displayed carrier in `TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionRouteAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionRouteAudit}^{\uparrow}$ packet is generated from a RealCauchyCompletion route row, CauchyLimitSeal row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise transports, displayed continuations, package provenance, local NameCert row, and route-audit verdict.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionRouteAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1033.1, Definition 1033.2, and Definition 1033.3 bind the finite Real-completion route audit over RealCauchyCompletion, CauchyLimitSeal, StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionRouteAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, selected limits, hidden tail selection, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route ordering, StreamName and RegSeqRat service rows, DyadicRatCore tolerance exposure, RealCauchyCompletion handoff, CauchyLimitSeal exposure, nontriviality, and field faithfulness for the same finite packet.

Chapter 1034

CauchySealFunctorUp NameCert construction

The CauchySealFunctor$^{\uparrow}$ surface realizes the Real-completion Cauchy seal route of chapter 9238 as a finite seal-transport packet. It sits between the completion classifier of chapter 900, the limit-seal source of chapter 386, the regular-rational source of chapter 301, the finite-window source of chapter 126, and the dyadic tolerance ledger of chapter 291. Its role is to carry a reusable transport row from displayed CauchyLimitSeal inputs to displayed RealCompletionSealClassifier reads before a Real$^{\uparrow}$ handoff consumes the route.

Falsifiable prediction: If a seal transport reaches a RealUp consumer without passing through either a displayed CauchyLimitSeal row or a displayed RealCompletionSealClassifier row, the CauchySealFunctor packet is refuted.

Independence witness: realcompletionsealclassifier, cauchylimitseal, cauchysealbudgetsynchronizer, regseqrat, streamname: no sibling is bijective because CauchySealFunctor carries a transport row between seal reads, while each sibling supplies only one endpoint, budget, regular-source, or finite-window surface.

Definition 1034.1 (Cauchy seal functor carrier)**.** *A* CauchySealFunctor$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, E, T, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyLimitSeal$^{\uparrow}$ *row;*
2. *E is the target* RealCompletionSealClassifierUp *row;*
3. *T is the seal-transport row linking L to E through displayed finite reads;*
4. *W is the* StreamName$^{\uparrow}$ *finite-window schedule consumed by the transport;*
5. *R is the* RegSeqRat$^{\uparrow}$ *regular-source handoff;*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used by the same handoff;*
7. *H records componentwise* hsame *transport for $L, E, T, W, R, D, C, P, N$;*
8. *C records the displayed* Cont *routes by which consumers replay the seal transport;*
9. *P is the* Pkg *provenance naming* CauchyLimitSeal$^{\uparrow}$, RealCompletionSealClassifierUp, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySealFunctor}^{\uparrow}}$ *row.*
The carrier accepts only transports whose consumer-visible route factors through $L, E, T, W, R, D, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchySealFunctorUp.CauchySealFunctorUp`

Theorem 1034.2 (Cauchy seal functor NameCert obligations)**.** *For the carrier of Definition 1034.1, the local* $\mathsf{NameCert}_{\mathsf{CauchySealFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (L, E, T, W, R, D, H, C, P, N$ source-seal admission for L, target-classifier admission for E, seal-transport exactness for T, finite-window scheduling through W, regular-source handoff through R, dyadic tolerance readback through D, componentwise* hsame *transport through H, displayed consumer routes through C, package provenance through P, local naming through N, and non-escape from quotient streams, selected limits, host real equality, countable choice, and ambient completeness.*

Proof. Project the accepted carrier. The displayed rows are exactly $L, E, T, W, R, D, H, C, P, N$, so carrier admission, transport exactness, package provenance, and local naming are finite BHist reads.

Read the source and target rows through the sibling chapters. The source row is the displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ surface, and the target row is the displayed RealCompletionSealClassifierUp classifier surface; T records only the transport between these two displayed endpoints.

The regularity and window rows are W, R, D. They force every transport read through a finite StreamName window, a RegSeqRat handoff, and a DyadicRatCore tolerance ledger before the route reaches a $\mathsf{Real}^{\uparrow}$ consumer.

The excluded objects have no carrier coordinate. Therefore the local NameCert surface contains no quotient stream, selected external limit, host real equality, countable choice, or ambient completeness datum. □

Theorem 1034.3 (Cauchy seal functor non-escape)**.** *Every public consumer of an accepted* $\mathsf{CauchySealFunctor}^{\uparrow}$ *packet reads only the source seal row L, target classifier row E, seal-transport row T, finite-window row W, regular-source row R, dyadic tolerance row D, componentwise* hsame *rows H, displayed* Cont *routes C, package provenance P, and local name row N. In particular, a* $\mathsf{Real}^{\uparrow}$ *consumer receives a transported finite seal read, not a quotient stream, selected limit, host real equality, countable choice, or ambient completeness proof.*

Proof. Start from the accepted packet $F = (L, E, T, W, R, D, H, C, P, N)$. The consumer-facing rows are the two seal endpoints, the transport row, the three finite route rows, and the local transport, route, provenance, and name rows.

Apply the NameCert obligations of Definition 1034.2. They force every public read through the displayed finite packet and leave no coordinate for a host-level completion object.

Thus any $\mathsf{Real}^{\uparrow}$ read sees only the transported finite seal surface. Since quotient streams, selected limits, host equality, countable choice, and ambient completeness are absent from the carrier, no public consumer can extract them. □

Theorem 1034.4 (Cauchy seal functor regularity handoff)**.** *Let $F = (L, E, T, W, R, D, H, C, P, N)$ be an accepted* $\mathsf{CauchySealFunctor}^{\uparrow}$ *packet. Every regularity-sensitive seal transport factors through the same triple W, R, D: a finite* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-source handoff, and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The transport row T may replay these rows through H and C, but it cannot replace them by an unbounded stream, a selected representative, or a host real-completion argument.*

Proof. Read the transport row T together with the route rows W, R, D. By carrier admission, these rows are the only finite-window, regular-source, and dyadic-tolerance coordinates available to the transport.

The componentwise hsame and Cont rows H and C replay the same triple rather than adding a second source. Hence every regularity-sensitive transport remains attached to the displayed StreamName, RegSeqRat, and DyadicRatCore rows.

The non-escape theorem excludes unbounded streams, selected representatives, host equality, and ambient completeness. The regularity handoff is therefore exactly the finite triple W, R, D as stated. □

Theorem 1034.5 (Cauchy seal functor Real-completion boundary)**.** *Let $F = (L, E, T, W, R, D, H, C, P, N)$ be an accepted* CauchySealFunctor$^{\uparrow}$ *packet. Any Real-completion consumer of F must pass through either the displayed* CauchyLimitSeal$^{\uparrow}$ *row L before transport or the displayed* RealCompletionSealClassifierUp *row E after transport, with T, W, R, D, H, C, P, N providing the finite route. A read that reaches a* Real$^{\uparrow}$ *consumer while bypassing both displayed seal rows is not a read of this packet.*

Proof. Project the accepted packet to the two seal endpoints L and E. The carrier has no other seal endpoint, so a Real-completion consumer can enter the transport only before T through L or after T through E.

Use the regularity handoff of Definition 1034.4. The route between the endpoints is the finite triple W, R, D together with the local rows H, C, P, N.

Finally apply non-escape. A consumer path that bypasses both L and E has no source seal, no target classifier, and no displayed transport row of the packet, so it is outside the CauchySealFunctor$^{\uparrow}$ carrier. □

The Lean `TasteGate.lean` development for CauchySealFunctor$^{\uparrow}$ must provide explicit Nontrivial CauchySealFunc and FieldFaithful CauchySealFunctor$^{\uparrow}$ instances for the displayed finite seal-transport packet.

Closure status for NameCert$_{\text{CauchySealFunctor}^{\uparrow}}$**.**

Constructive story: A CauchySealFunctor$^{\uparrow}$ packet is generated from a CauchyLimitSeal source row, a RealCompletionSealClassifier target row, one seal-transport row, StreamName windows, RegSeqRat handoff rows, DyadicRatCore tolerance rows, componentwise hsame transports, displayed Cont routes, package provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{CauchySealFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 1034.1, Definition 1034.2, Definition 1034.3, Definition 1034.4, and Definition 1034.5 specify the finite seal-transport packet and its Real-completion consumer boundary.

Formal verification: FormalStatus(NameCert$_{\text{CauchySealFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy streams, selected limits, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, seal-transport exactness, regularity handoff, Real-completion boundary, nontriviality, and field faithfulness for the same packet.

Chapter 1035

A Concrete Naming Certificate for $\mathsf{LocatedCauchyModulus}^{\uparrow}$

The $\mathsf{LocatedCauchyModulus}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as a finite located tail-modulus object. It sits downstream of the finite seal classifier in chapter 900: StreamName windows, RegSeqRat readbacks, DyadicRatCore radius ledgers, CauchyLimitSeal compatibility rows, and RealUp seal rows are exposed as one displayed packet before any completion-facing consumer may read the seal. The packet is not a quotient stream, host real equality, selected limit, countable-choice surface, or ambient completeness principle.

Definition 1035.1 (Located Cauchy modulus carrier)**.** *A* $\mathsf{LocatedCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window source row;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row exposed from the displayed windows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger recording the dyadic tolerance attached to the same tail request;*
4. *T is the located tail-modulus row naming the finite threshold at which the StreamName and RegSeqRat windows become mutually readable at radius D;*
5. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row reached after the located tail-modulus threshold;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after S, R, D, T, L have been displayed;*
7. *H records componentwise* hsame *transports for the window, readback, radius, modulus, limit-seal, Real-seal, route, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes from S, R, D to T, from T to L, and from L to E;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, unbounded observation, detached tail choice, or a RealUp seal read that bypasses the located modulus row.

Lean4 ✓ BEDC.Derived.LocatedCauchyModulusUp.LocatedCauchyModulusTasteGate single carrier alignment
Lean4 (definition) BEDC.Derived.LocatedCauchyModulusUp.LocatedCauchyModulusCarrier
Lean4 (definition) BEDC.Derived.LocatedCauchyModulusUp.LocatedCauchyModulusUp

Theorem 1035.2 (Located Cauchy modulus NameCert obligations)**.** *For the carrier of Definition 1035.1, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (S, R, D, T, L, E, H, C, P, N),$$

StreamName window exposure through S, RegSeqRat readback through R, DyadicRatCore radius accounting through D, located tail-modulus exposure through T, CauchyLimitSeal compatibility through L, RealUp seal routing through E, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.

Lean4 ✓ BEDC.Derived.LocatedCauchyModulusUp.LocatedCauchyModulusNameCertObligations

Proof. Project the carrier of Definition 1035.1. The displayed packet exposes exactly the window, readback, radius, tail-modulus, limit-seal, Real-seal, transport, route, provenance, and local naming rows.

Read the route in dependency order. The StreamName row S supplies the finite window, the RegSeqRat row R reads that window as regular-rational data, and the DyadicRatCore row D supplies the radius ledger. The located modulus row T may then be consumed, followed by the CauchyLimitSeal compatibility row L and the RealUp seal row E.

The rows H, C, P, N preserve this route componentwise, replay it as displayed continuations, record its provenance, and name the local surface. Since the carrier contains no quotient stream equality, host real equality, selected limit, unbounded observation, detached tail choice, or bypass route to E, the listed rows exhaust the NameCert obligations. □

Theorem 1035.3 (Located Cauchy modulus window stability)**.** *Let $M = (S, R, D, T, L, E, H, C, P, N)$ and $M' = (S', R', D', T', L', E', H', C', P', N')$ be accepted* $\mathsf{LocatedCauchyModulus}^{\uparrow}$ *packets. If their StreamName, RegSeqRat, DyadicRatCore, and located tail-modulus rows are related componentwise by the displayed transport rows H and H', then every completion-facing read of E and E' factors through the same ordered window route*

$$S, R, D, T, L, E.$$

In particular, transport cannot create a second RealUp seal route outside the displayed StreamName window, RegSeqRat handoff, DyadicRatCore radius ledger, located modulus threshold, and CauchyLimitSeal compatibility row.

Proof. Start with the two carrier projections. The only analytic rows before the RealUp seal are S, R, D, T, L in the first packet and S', R', D', T', L' in the second.

Use the NameCert obligations above. They bind every accepted seal read to the displayed route through the StreamName window, RegSeqRat readback, DyadicRatCore radius ledger, located modulus threshold, and CauchyLimitSeal compatibility row.

Now apply the componentwise transport rows. Transport preserves the named window, readback, radius, modulus, limit-seal, and Real-seal coordinates, while the displayed continuations replay the same order. No carrier coordinate remains for quotient stream equality, host real equality, selected limits, or a detached tail oracle, so transported completion-facing reads still factor through the same finite route. □

Falsifiable prediction: Every accepted RealUp seal read from a located modulus factors through a displayed StreamName window, RegSeqRat handoff, DyadicRatCore radius ledger, and CauchyLimitSeal compatibility row.

Independence witness: cauchymodulus, regseqrat, streamname, realcompletionsealclassifier, cauchysealbudgetsynchronizer: none is bijective to $\mathsf{LocatedCauchyModulus}^{\uparrow}$ because a Cauchy modulus row lacks the StreamName and RegSeqRat window readbacks, RegSeqRat lacks the dyadic radius and limit-seal rows, StreamName lacks the regular-rational and Real seal rows, RealCompletionSealClassifier lacks the located tail-modulus threshold as a carrier row, and CauchySealBudgetSynchronizer records synchronized budget rows without making the located modulus row the displayed threshold. The located carrier stores $S, R, D, T, L, E, H, C, P, N$ together in one finite surface.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/LocatedCauchyModulusUp/TasteGate` It must provide `Nontrivial LocatedCauchyModulusUp` and `FieldFaithful LocatedCauchyModulusUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyModulus}^{\uparrow}$ packet is generated from a StreamName finite-window source, RegSeqRat readback, DyadicRatCore radius ledger, located tail-modulus row, CauchyLimitSeal compatibility row, RealUp seal, componentwise transports, displayed continuations, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1035.1, Definition 1035.2, and Definition 1035.3 bind the finite located tail-modulus route from StreamName and RegSeqRat windows through DyadicRatCore radius accounting to CauchyLimitSeal and RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located tail-modulus exposure, window stability, CauchyLimitSeal compatibility, RealUp seal routing, nontriviality, and field faithfulness for the same finite packet.

Chapter 1036

A Concrete Naming Certificate for $\mathsf{LocalStencilObservation}^{\uparrow}$

The $\mathsf{LocalStencilObservation}^{\uparrow}$ packet realizes the cellular substrate reading of chapter 9390 at the smallest finite observation surface: a displayed neighbourhood stencil, a displayed local rule row, and the observed finite row window to which a pattern classifier may point. It sits beside the whole automaton packet of chapter 934, the local transition packet of chapter 984, and the finite pattern catalog of chapter 995. Unlike those siblings, it certifies only one local observation packet, not a complete orbit, not a general update rule interface, and not a catalog of all accepted pattern tags.

Falsifiable prediction: If a purported $\mathsf{LocalStencilObservation}^{\uparrow}$ consumer classifies a cellular pattern without exposing a finite neighbourhood stencil, the selected local rule row, and the observed row window, then the consumer has left the certificate.

Independence witness: cellularautomaton, cellularlocalupdaterule, cellularpatterncatalog: none is bijective to $\mathsf{LocalStencilObservation}^{\uparrow}$ because this packet carries the joint neighbourhood, local-rule, observed-row, classifier, transport, route, package, and NameCert rows for one finite stencil observation; the automaton sibling carries an orbit ledger, the local-update sibling carries a next-cell output surface, and the pattern-catalog sibling carries a finite catalog table.

Definition 1036.1 (Local stencil observation carrier)**.** *A* $\mathsf{LocalStencilObservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, R, V, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite neighbourhood-stencil row, including the bounded cell positions inspected by the classifier;*
2. *R is the local-rule row whose displayed entry is permitted to read the stencil row;*
3. *V is the observed finite row-window produced or inspected at the named cellular time slice;*
4. *T is the finite classifier row assigning an admitted local pattern tag, such as ether, glider fragment, collision fragment, boundary fragment, or refused observation;*
5. *H records componentwise* hsame *transport for the stencil, local-rule, observed-window, classifier, route, package, and local naming rows;*
6. *C records the displayed* Cont *route from W and R to the observed row window V, then from V to the classifier row T;*

7. *P is the* Pkg *provenance over chapter 9390, chapter 934, chapter 984, chapter 995,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalStencilObservation}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only by W, R, V, T, H, C, P, N. It has no coordinate for an unbounded cellular orbit, a host-side pattern recognizer, an unstated Rule 110 convention, a hidden catalog row, or a detached universality oracle.

Lean4 (definition) `BEDC.Derived.LocalStencilObservationUp.LocalStencilObservationUp`

Theorem 1036.2 (LocalStencilObservation NameCert obligations)**.** *For the carrier of Definition 1036.1, the local* $\mathsf{NameCert}_{\mathsf{LocalStencilObservation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$O = (W, R, V, T, H, C, P, N),$$

neighbourhood exposure through W, local-rule accountability through R, observed-row exposure through V, pattern-classifier exposure through T, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. Every cellular pattern classification exported by this packet must read a finite stencil before it reads the finite observed row and classifier tag.

Proof. Carrier admission is projection from Definition 1036.1. The finite packet displays W, R, V, T, H, C, P, N as its complete observation surface.

The cellular content is read in the order named by the carrier. The neighbourhood row W supplies the bounded stencil, the local-rule row R supplies the displayed rule entry, and the observed-row window V is the only row window available to the classifier.

The classifier row T can therefore tag only the displayed observation. The transport, route, provenance, and local name rows H, C, P, N preserve and replay that same finite packet; they do not add an orbit ledger, hidden catalog, or host pattern recognizer. □

Theorem 1036.3 (LocalStencilObservation carrier totality)**.** *Every accepted* $\mathsf{LocalStencilObservation}^{\uparrow}$ *read is exhausted by the finite rows W, R, V, T, H, C, P, N. A consumer may use the whole-automaton packet of chapter 934, the local update packet of chapter 984, or the finite pattern catalog of chapter 995 only as provenance named in P; none of those siblings supplies an additional coordinate inside the local stencil packet.*

Proof. Start with carrier projection. The accepted packet has exactly the finite stencil row, local-rule row, observed-window row, classifier row, transport row, route row, provenance row, and local naming row.

The sibling references enter through P. They explain where the cellular reading, local update discipline, and finite pattern vocabulary are sourced, but they are not extra fields of O.

Apply Definition 1036.2. Since each accepted classification must pass through the displayed rows and the carrier lists no further coordinate, those rows are total for the local observation surface. □

Theorem 1036.4 (LocalStencilObservation finite non-escape)**.** *No accepted* $\mathsf{LocalStencilObservation}^{\uparrow}$ *packet exports an unbounded orbit, a host cellular evaluator, a hidden pattern catalog, a global Rule 110 classifier, or a detached universality oracle. Every exported observation factors through the finite route*

$$W, R, V, T, H, C, P, N.$$

Proof. Inspect the rows of Definition 1036.1. The only cellular data are the neighbourhood stencil W, the local-rule row R, the observed finite row window V, and the classifier row T.

The structural rows H, C, P, N only transport, replay, package, and locally name those displayed data. They do not introduce an all-time orbit, a host evaluator, an unstated catalog, or a universality witness.

Thus any read of one of the excluded objects would require a coordinate absent from $O = (W, R, V, T, H, C, P, N)$. Such a read is outside the accepted local stencil observation certificate. □

Theorem 1036.5 (LocalStencilObservation Rule 110 classifier route)**.** *Any Rule 110 local-pattern classifier read exported through* $\mathsf{LocalStencilObservation}^{\uparrow}$ *must expose the finite rows*

$$W, R, V, T, H, C, P, N$$

before it can cite the global cellular substrate of chapter 9390, the whole-automaton packet of chapter 934, the local update packet of chapter 984, the finite pattern catalog of chapter 995, or any Cook-frontier row named by the substrate reading. The route exports one finite local classifier observation, not a global Rule 110 universality witness.

Proof. Project the carrier of Definition 1036.1. The accepted local observation exposes the neighbourhood stencil W, local-rule row R, observed finite row window V, classifier row T, componentwise transport H, displayed replay C, provenance P, and local name N.

Apply Definition 1036.4. It excludes unbounded orbits, hidden catalog reads, global Rule 110 classifiers, and detached universality witnesses from the packet, so a Cook-frontier or substrate citation cannot precede the local rows.

Finally use the sibling boundary supplied by the cellular automaton, local update, and pattern catalog chapters. Those chapters may be cited through P only after the local stencil, rule, observed window, and classifier rows have been displayed; they do not add a second classifier coordinate to $\mathsf{LocalStencilObservation}^{\uparrow}$. □

Theorem 1036.6 (LocalStencilObservation sibling independence)**.** *The* $\mathsf{LocalStencilObservation}^{\uparrow}$ *carrier is not bijective to the sibling packets* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$*, or* $\mathsf{CellularPatternCatalog}^{\uparrow}$*. It retains the joint local observation surface* W, R, V, T, H, C, P, N*, while those siblings respectively retain a finite orbit ledger, a next-cell update surface, or a catalog lookup table not present as an internal coordinate of this packet.*

Proof. Compare the carrier rows. The $\mathsf{LocalStencilObservation}^{\uparrow}$ packet has the observed row-window coordinate V tied to a single finite stencil W and a selected rule row R.

The whole automaton sibling records an orbit ledger, the local update sibling records a visible successor-cell surface, and the pattern-catalog sibling records a closed finite catalog row. Those rows are not interchangeable with the single observed-window row V and the local classifier row T.

Conversely, this packet does not carry the sibling-specific orbit, next-cell, or catalog coordinates except as provenance through P. Hence no bijection preserves the displayed NameCert obligation surface in both directions. □

The future Lean TasteGate file `lean4/BEDC/Derived/LocalStencilObservationUp/TasteGate.lean` provides `Nontrivial LocalStencilObservationUp` and `FieldFaithful LocalStencilObservationUp` instances for the displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalStencilObservation}^{\uparrow}}$.

Constructive story: A LocalStencilObservation$^{\uparrow}$ packet is generated from a finite neighbourhood stencil, selected local-rule row, observed row window, local classifier row, componentwise hsame transport, displayed Cont route, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocalStencilObservation}^{\uparrow}}$, obligation)

Scope closed: Definition 1036.1, Definition 1036.2, Definition 1036.3, Definition 1036.4, Definition 1036.5, and Definition 1036.6 bind the displayed rows W, R, V, T, H, C, P, N for the local stencil, rule, observed window, classifier, BHist, hsame, Cont, Pkg, and NameCert dependencies without asserting Lean verification.

Formal verification: FormalStatus(NameCert$_{\text{LocalStencilObservation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global cellular orbits, Rule 110 universality, host evaluator correctness, complete pattern-catalog coverage, or a public standard bridge.

Upgrade path: Lean follow-up will bind the finite observation carrier under `BEDC.Derived.LocalStencilObservationUp`.

Chapter 1037

A Concrete Naming Certificate for $\mathsf{RelationalFrameAudit}^{\uparrow}$

The $\mathsf{RelationalFrameAudit}^{\uparrow}$ packet turns the no-global-frame and observer-symmetry surface of chapter 9419 into finite NameCert data. It consumes the configuration rows of chapter 474, the causal rows of chapter 478, the unary-bound rows of chapter 594, the refusal boundary of chapter 1003, and the ledger-local routing of chapter 759. Its public role is to audit a proposed comparison between two observer frames without admitting a global frame, host synchronization map, or metric primitive.

Definition 1037.1 (RelationalFrameAudit carrier)**.** *A* $\mathsf{RelationalFrameAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, A, B, Q, S, D, L, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row supplying the finite multi-history configuration;*
2. *A and B are the two displayed observer-history slots whose local frame comparison is being audited;*
3. *Q is the frame-request row, recording the requested comparison as a finite local read rather than as an ambient coordinate system;*
4. *S is the observer-symmetry audit row, requiring both directions of the comparison to be visible before the request is accepted;*
5. *D is the* $\mathsf{CrossHistCausal}^{\uparrow}$ *row used by the comparison request, read only after the two observer slots are displayed;*
6. *L is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row that bounds the causal read without introducing a host speed, metric tensor, or synchronized clock;*
7. *F is the* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *refusal verdict row for any request that asks for a privileged global frame, total observer order, selected synchronization map, or detached frame comparison;*
8. *H records componentwise* hsame *transport for M, A, B, Q, S, D, L, F and the structural rows;*
9. *C records displayed* Cont *routes from the frame request either through the two-Hist route A, B, D, L, S or to the refusal row F;*

10. *P is the* Pkg *provenance over* RelationalFrameAudit$^{\uparrow}$, MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, NoGlobalSyncBoundary$^{\uparrow}$, InterHistLocalityLedger$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$ *row.*

The carrier exports a finite audit boundary for frame-comparison requests. It has no coordinate for a global frame, shared clock, metric tensor, host Lorentz chart, selected synchronization function, or comparison detached from the displayed histories.

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditCarrier full packet semantic namecert`

Definition 1037.2 (LayeredRelationCert carrier for RelationalFrameAudit)**.** *The* RelationalFrameAudit$^{\uparrow}$ *realization of* LayeredRelationCert *is the finite relation-audit packet*

$$\Lambda_R = (S_A, S_B, \ell, \pi, \nu, G, E, \partial, H, C, P, N)$$

where S_A and S_B are the two source-row lists, ℓ is the displayed two-chain layer map, π records the preserved relation-schema rows, ν records the not-preserved rows, G is the source ledger for each correspondence, E is the exactness row for same-schema language, and ∂ is the failure boundary. The structural rows H, C, P, N are the componentwise hsame *transports, displayed* Cont *routes,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$ *row.*

The packet is accepted only as relation-audit data. It may compare the refinement shape of two displayed chains, but it does not identify their objects, sources, observers, empirical content, or host frames.

Lean4 (definition) `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditLayeredRelationCertCarrier`

Theorem 1037.3 (RelationalFrameAudit NameCert obligations)**.** *For the carrier of Definition 1037.1, the local* $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (M, A, B, Q, S, D, L, F, H, C, P, N),$$

two-Hist locality through M, A, B, frame-request exposure through Q, observer-symmetry audit through S, cross-Hist causal routing through D, maximum-rate boundary reading through L, no-global-frame refusal through F, and transport, continuation, provenance, and local naming through H, C, P, N.

Proof. Carrier admission is projection from Definition 1037.1. The tuple displays the configuration row, two observer slots, frame request, symmetry audit, causal route, rate boundary, refusal verdict, transport, continuation, provenance, and local name.

The positive comparison route first reads M, A, B. The causal row D and maximum-rate row L can then be consumed as finite inter-Hist data, with the observer-symmetry row S requiring that both comparison directions remain visible.

The negative route is the refusal verdict F. By the no-global-synchronization boundary of chapter 1003, a request for a privileged frame, shared clock, total observer order, or selected synchronization map is retained as refused data rather than promoted to an admitted coordinate.

Finally H, C, P, N only transport, replay, package, and name the same rows. Thus the NameCert surface is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditTasteGate single carrier alignment`

Theorem 1037.4 (RelationalFrameAudit obligation totality)**.** *Any accepted* $\mathsf{RelationalFrameAudit}^{\uparrow}$ *consumer must expose exactly the finite obligation surface*

$$M, A, B, Q, S, D, L, F, H, C, P, N$$

of Definition 1037.1. Thus the consumer displays the $\mathrm{MultiHistConfig}^{\uparrow}$ *row* M*, the two observer slots* A, B*, the frame request* Q*, the observer-symmetry audit* S*, the* $\mathsf{CrossHistCausal}^{\uparrow}$ *route* D*, the* $\mathsf{MaxCausalRate}^{\uparrow}$ *bound* L*, the* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *refusal verdict* F*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$ *row* N*. No accepted consumer may replace one of these rows by a global frame, shared clock, selected synchronization map, host Lorentz chart, metric tensor, detached comparison, or source-equivalence transport.*

Proof. Project the carrier of Definition 1037.1. The tuple $R = (M, A, B, Q, S, D, L, F, H, C, P, N)$ is the complete displayed inventory for an accepted frame-audit packet.

Apply Definition 1037.3. Its NameCert surface reads the same configuration row, two observer slots, request row, symmetry audit, causal route, maximum-rate boundary, refusal verdict, transport, replay, provenance, and local naming rows.

It remains to exclude extra exits. The carrier contains no coordinate for a global frame, shared clock, selected synchronization map, Lorentz chart, metric tensor, detached comparison, or source-equivalence transport. Hence every accepted consumer exposes exactly the listed finite rows and nothing outside their projection. □

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditObligationTotality`

Theorem 1037.5 (LayeredRelationCert NameCert obligation surface)**.** *For the carrier* Λ_R *of Definition 1037.2, the local* $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for the displayed tuple* $\Lambda_R = (S_A, S_B, \ell, \pi, \nu, G, E, \partial, H, C, P, N)$*;*
2. *classifier locality for the two source-row lists* S_A, S_B *and the layer map* ℓ*;*
3. *stability through the componentwise* hsame *transports* H *and displayed* Cont *routes* C*;*
4. *exactness through the preserved rows* π*, not-preserved rows* ν*, and same-schema exactness row* E*;*
5. *ledger and export control through* G, ∂, P, N*, so any relation read carries provenance, failure boundary, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditLayeredRelationCert namecert obligation surface`

Proof. Carrier admission projects the tuple of Definition 1037.2. No source, layer, preservation, refusal, ledger, exactness, boundary, transport, continuation, provenance, or naming row is implicit.

Classifier locality reads only the two displayed source lists and the layer map. The preserved row π and not-preserved row ν then separate same-schema language from object identity.

Stability is supplied by H and C. These rows transport and replay the same displayed relation packet; they do not add a third chain, a host frame, or an empirical bridge.

The ledger row G, failure boundary ∂, package row P, and local NameCert row N control export. Hence the NameCert surface is exactly the five listed rows. □

Theorem 1037.6 (LayeredRelationCert finite relation exactness)**.** *An accepted* RelationalFrameAudit$^{\uparrow}$ *realization of* LayeredRelationCert *exports same-schema language exactly for the finite relation packet* Λ_R*. The export is available only when source rows, layer map, preserved rows, not-preserved rows, ledger, exactness, failure boundary, transport, continuation, provenance, and local naming are all displayed in the same packet. It exports no object identity, global frame, empirical law, subject-substance row, or unlisted bridge.*

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditLayeredRelationCert finite relation exactness`

Proof. Begin with the carrier Λ_R. Its source rows S_A, S_B, layer map ℓ, preserved rows π, not-preserved rows ν, ledger G, exactness row E, and boundary ∂ are separate coordinates.

Apply Definition 1037.5. The exactness row can be consumed only together with the preserved and not-preserved rows, so same-schema language has an explicit positive surface and an explicit refusal surface.

Finally use the export control row. Since H, C, P, N transport, replay, package, and name only the displayed packet, a consumer has no coordinate through which to read object identity, global-frame data, empirical content, subject-substance content, or a bridge outside Λ_R. □

Theorem 1037.7 (RelationalFrameAudit no-global-frame refusal)**.** *Every accepted* RelationalFrameAudit$^{\uparrow}$ *packet that receives a frame request outside the displayed two-Hist route sends that request to the refusal verdict* F*. The accepted packet exports no global frame, shared clock, total observer order, selected synchronization map, host Lorentz chart, metric tensor, or detached comparison row.*

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditCarrier no global frame refusal`

Proof. Start with the carrier inventory. The only frame-facing rows are the request row Q, the two observer slots A, B, the causal and rate rows D, L, the observer-symmetry row S, and the refusal row F.

Apply the NameCert obligations of Definition 1037.3. A supported request must travel through the displayed two-Hist route; an unsupported request has no row except the no-global-frame refusal verdict.

Use the no-global-synchronization boundary from chapter 1003. Since the missing global objects are not coordinates of $R = (M, A, B, Q, S, D, L, F, H, C, P, N)$, no continuation or naming projection can export them as admitted data. □

Theorem 1037.8 (RelationalFrameAudit observer-symmetry factorization)**.** *Any accepted* RelationalFrameAudit$^{\uparrow}$ *packet that exposes a frame comparison factors the read through the displayed two-Hist route*

$$A, B \longrightarrow D \longrightarrow L \longrightarrow S$$

together with componentwise hsame *transport in* H *and displayed* Cont *replay in* C*. If the request does not factor through this route, the packet reads it only through the refusal verdict* F*.*

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditCarrier observer symmetry factorization`

Proof. Project the frame-request row Q from the carrier. The request has only two public exits: the displayed comparison route through A, B, D, L, S or the refusal route through F.

For the supported exit, read the two observer-history rows from M and then apply the cross-Hist causal and maximum-rate sibling surfaces of chapter 478 and chapter 594. The comparison remains local because D and L are finite rows over the displayed multi-Hist configuration.

The row S supplies the observer-symmetry audit. Both directions must be visible before the packet exports the comparison, and H and C only transport and replay the same factorized route.

For the unsupported exit, apply Definition 1037.7. A request that does not pass through the two-Hist route has no accepted destination except F. □

Theorem 1037.9 (RelationalFrameAudit formal-target boundary)**.** *Any future formal target for* $\mathsf{RelationalFrameAudit}^{\uparrow}$ *must expose the same carrier rows*

$$M, A, B, Q, S, D, L, F, H, C, P, N$$

together with the LayeredRelationCert *rows of Definition 1037.2 before it claims TasteGate admission, no-global-frame refusal, or observer-symmetry factorization. A target that omits either the frame-audit carrier rows or the relation-audit rows cannot be read as the Lean-facing boundary for this chapter.*

Lean4 ✓ `BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditFormalTargetBoundary`

Proof. Project the carrier of Definition 1037.1. It fixes the frame-audit rows $M, A, B, Q, S, D, L, F, H, C, P, N$ as the finite packet visible to any target.

Read the LayeredRelationCert obligation surface through Definition 1037.5. The source-list, layer-map, preservation, non-preservation, ledger, exactness, failure-boundary, transport, replay, provenance, and naming rows are separate relation-audit obligations rather than optional prose.

Finally use no-global-frame refusal and observer-symmetry factorization, Definition 1037.7 and Definition 1037.8. Those theorems consume the displayed refusal and two-Hist route rows, so a later formal target may claim their boundary only after the same carrier and relation-audit surfaces have been exposed. □

Falsifiable prediction: If a relation packet omits the not-preserved row, it cannot satisfy the RelationalFrameAuditUp export gate for same-schema language.

Independence witness: multihistconfig, multihistcoupling, multihistsuperposition, finite-multihist-packet: these siblings carry multi-Hist sites, couplings, superpositions, or packet rows, but none carries a two-chain layer map with preserved and not-preserved relation-audit fields, so no carrier bijection preserves the NameCert obligations.

The later Lean TasteGate file must provide $\mathsf{Nontrivial\ RelationalFrameAudit}^{\uparrow}$ and $\mathsf{FieldFaithful\ RelationalFrameAu}$ instances for the displayed rows $M, A, B, Q, S, D, L, F, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RelationalFrameAudit}^{\uparrow}$ packet is generated from a multi-Hist configuration, two observer-history slots, a frame request, observer-symmetry audit row, cross-Hist causal route, maximum-rate boundary, no-global-frame refusal verdict, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1037.9 together with the existing marker BEDC.Derived.RelationalFrameAuditUp.RelationalFrameAuditTasteGa binds the public export boundary for carrier admission, relation-audit rows, no-global-frame refusal, and observer-symmetry factorization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RelationalFrameAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct a metric tensor, Lorentz chart, shared clock, global synchronization map, Lean proof, or bridge checking for RelationalFrameAudit$^{\uparrow}$.

Upgrade path: Public closure requires a checked Lean carrier, checked NameCert obligation theorem, checked no-global-frame refusal theorem, checked observer-symmetry theorem, and bridge-facing standard route for RelationalFrameAudit$^{\uparrow}$.

Chapter 1038

A Concrete Naming Certificate for $\mathsf{TarskiTruthRefusal}^{\uparrow}$

The $\mathsf{TarskiTruthRefusal}^{\uparrow}$ packet realizes the Tarski side of the reflection-limit vision of chapter 9461 as finite naming data. It consumes the truth-total obstruction of chapter 472 and the structural boundary of chapter 586, but its carrier is narrower: it records the refusal row that separates CIC* sentence codes from any BEDC-internal truth predicate.

Definition 1038.1 (Tarski truth-refusal carrier)**.** *A* $\mathsf{TarskiTruthRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, Q, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* CIC* *sentence-code row, recording the finite coded sentence surface being read;*
2. *Q is the attempted truth-read query row, recording the request for an internal classifier deciding represented sentence truth;*
3. *R is the truth-read refusal row, recording that the requested predicate is not a carrier coordinate;*
4. *D is the diagonal pressure row, recording the self-application shape that would turn Q into the Tarski obstruction;*
5. *L is the boundary ledger separating admitted structural sentence-code reads from refused truth-predicate reads;*
6. *H records componentwise* hsame *transports for sentence codes, truth-read queries, refusals, diagonal rows, ledgers, routes, provenance, and local names;*
7. *C records the displayed* Cont *routes by which a consumer may inspect sentence codes, replay the query, read the refusal, and return to the boundary ledger;*
8. *P is the* Pkg *provenance over* BHist, CIC* *sentence-code rows, truth-read queries, Tarski diagonal rows,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TarskiTruthRefusal}^{\uparrow}}$ *ledger.*

The carrier contains no BEDC truth predicate, no truth-value classifier for all CIC* *sentence codes, no runtime validator, and no route converting the structural self-description of BEDC into a truth predicate for that description.*

Lean4 ✓ `BEDC.Derived.TarskiTruthRefusalUp.TarskiTruthRefusalTasteGate single carrier alignment`

Theorem 1038.2 (Tarski truth-refusal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{TarskiTruthRefusal}^{\uparrow}}$ *surface for Definition 1038.1 consists of carrier admission for* $T = (S, Q, R, D, L, H, C, P, N)$*, sentence-code exposure through* S*, attempted truth-read exposure through* Q*, refusal exactness through* R*, diagonal-pressure accounting through* D*, boundary-ledger exactness through* L*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 1038.1. The packet lists the sentence-code, query, refusal, diagonal, ledger, transport, route, provenance, and naming rows as the complete public surface.

The sentence-code row S is read structurally. It may expose the same kind of coded syntax that appears in the reflection-limit chapter, but it does not decide the truth of the represented sentence.

The query and refusal rows Q, R record the Tarski boundary. A consumer may ask for an internal truth predicate, and the packet answers only by displaying the refusal row together with the diagonal pressure row D and the ledger L.

The remaining rows H, C, P, N only transport, replay, package, and name the same finite surface. Since none of them contains a truth predicate coordinate, the local NameCert obligations are exhausted by the displayed packet. □

Theorem 1038.3 (Tarski truth-refusal non-escape)**.** *Every accepted* $\mathsf{TarskiTruthRefusal}^{\uparrow}$ *packet exports* CIC* *sentence-code information only through*

$$S, Q, R, D, L, H, C, P, N.$$

Any consumer request for a BEDC-internal truth predicate factors through the truth-read refusal row R *and the boundary ledger* L*. No projection from the packet yields a predicate classifying every represented sentence code by truth value.*

Proof. Project the accepted carrier as $T = (S, Q, R, D, L, H, C, P, N)$. The carrier definition identifies R as the only row whose role answers the truth-read query, and it answers by refusal.

The structural read of S is therefore separated from the attempted truth read Q. The diagonal row D records why accepting Q as a truth predicate would cross the Tarski boundary, while L records that the accepted consumer surface stops at the refusal.

The transport, route, provenance, and local naming rows preserve this same separation. They can replay the sentence-code read and the refusal read, but they add no classifier that decides all represented sentence codes. □

Falsifiable prediction: If the truth-read refusal row is removed, the packet can no longer block an attempted internal truth predicate for CIC-star sentence codes.

Independence witness: truthtotalreflection, selfreferencetruthbranch, reflectionlimitboundary, runtime-reflection-boundary, and groundloopboundary: none is bijective to $\mathsf{TarskiTruthRefusal}^{\uparrow}$ while preserving the NameCert fields. TruthTotalReflection carries a broader truth-total attempt and diagonal obstruction surface rather than the isolated truth-read refusal row R. SelfReferenceTruthBranch carries a certificate-indexed branch query and socket report rather than the sentence-code refusal ledger S, Q, R, D, L. ReflectionLimitBoundary carries closed ground-loop and structural meta-description rows in addition to a truth-boundary ledger, while this carrier contains no ground-loop coordinate. RuntimeReflectionBoundary carries runtime-request and execution-refusal rows rather than a Tarski sentence-code truth-read refusal. GroundLoopBoundary carries primitive mark distinction closure and no CIC* sentence-code truth query.

The Lean follow-up `TasteGate.lean` must include `Nontrivial TarskiTruthRefusalUp` with explicit `witness_pair` and `FieldFaithful TarskiTruthRefusalUp` by constructor-field injection. The canonical downstream declaration surface is `BEDC.Derived.TarskiTruthRefusalUp.taste_gate`.

Closure status for $\mathsf{NameCert}_{\mathsf{TarskiTruthRefusal}\uparrow}$.

Constructive story: A $\mathsf{TarskiTruthRefusal}^{\uparrow}$ packet is generated from a CIC^* sentence-code row, an attempted truth-read query, a truth-read refusal row, a diagonal-pressure row, a boundary ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert ledger.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TarskiTruthRefusal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1038.1, Definition 1038.2, and Definition 1038.3 bind the finite sentence-code refusal carrier and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TarskiTruthRefusal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, a BEDC truth predicate, a truth-value classifier for all CIC^* sentence codes, runtime validation, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, truth-read refusal exactness, diagonal-pressure accounting, boundary-ledger exactness, Nontrivial, and FieldFaithful for the displayed packet.

Chapter 1039

A Concrete Naming Certificate for $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$

The $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ packet localizes the uniqueness clause of chapter 9238 into one finite comparison surface. It sits after the regular-limit uniqueness carrier of chapter 427, the two candidate completion seals of chapter 386, and the finite selector interface of chapter 673. The packet does not claim Real completeness; it records the finite route through which a consumer may read uniqueness between two sealed Real-completion candidates.

Falsifiable prediction: Every accepted RealLimitUniquenessSealUp consumer factors through the two displayed CauchyLimitSealUp rows, the shared RegSeqRat comparison row, the finite StreamName window row, the DyadicRatCore tolerance row, and the RealUp classifier row before it can read uniqueness.

Independence witness: regular-limit-uniqueness, cauchylimitseal, limitselector, realcompleteness, real: RegularLimitUniquenessUp lacks the paired finite RealUp seal route, CauchyLimitSealUp lacks the two-seal comparison budget, LimitSelectorUp lacks the second candidate seal, RealCompletenessUp is a route lattice rather than a local uniqueness carrier, and RealUp lacks the completion-seal comparison ledger.

Definition 1039.1 (Real limit uniqueness seal carrier)**.** *A* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$$

with the following displayed rows:

1. L_0 *is the first* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row naming a candidate completion seal;*
2. L_1 *is the second* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row naming the comparison seal;*
3. Q *is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row through which both sealed candidates are read;*
4. S *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before either uniqueness read is public;*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the comparison;*
6. R *is the terminal* $\mathsf{Real}^{\uparrow}$ *classifier row consumed by the uniqueness surface;*
7. V *is the* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *row binding the two sealed candidates to one Real-completion uniqueness read;*

8. *H records componentwise* hsame *transport for the two seal rows, comparison row, window row, dyadic ledger, Real classifier, uniqueness row, route row, provenance row, and naming row;*
9. *C records the displayed* Cont *route from the two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows through Q, S, D, R, V;*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularLimitUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected host limit, quotient stream equality, detached selector choice, ambient completeness theorem, or Real equality read outside the displayed rows.

Lean4 ✓ `BEDC.Derived.RealLimitUniquenessSealUp.RealLimitUniquenessSealTasteGate single carrier alignment`

Theorem 1039.2 (Real limit uniqueness seal obligation surface)**.** *For the carrier of Definition 1039.1, the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$U = (L_0, L_1, Q, S, D, R, V, H, C, P, N),$$

the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ *candidate rows L_0, L_1, shared regular-rational comparison through Q, finite window exposure through S, dyadic tolerance accounting through D, terminal* $\mathsf{Real}^{\uparrow}$ *classification through R, uniqueness handoff through V, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 1039.1. The tuple exposes the two completion-seal candidates, the shared regular-rational comparison row, the finite window row, the dyadic tolerance ledger, the terminal Real classifier, the regular-limit uniqueness row, transport, replay, provenance, and local naming.

Read the route in dependency order. The two seal rows L_0, L_1 must be present before the comparison row Q can be interpreted as a comparison of candidate limits, and Q must be read through the finite window row S and dyadic ledger D before the terminal Real classifier R is available.

The uniqueness row V is then only a local handoff from that displayed route. The rows H, C, P, N preserve and name the same route, while the carrier supplies no coordinate for selected host limits, quotient stream equality, detached selector choice, ambient completeness, or Real equality outside the packet. □

Lean4 ✓ `BEDC.Derived.RealLimitUniquenessSealUp.RealLimitUniquenessSealCarrier obligation surface`

Theorem 1039.3 (Real limit uniqueness seal nonescape)**.** *Every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *consumer reading uniqueness between two Real-completion candidates factors through the finite sequence*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N.$$

No accepted consumer may replace this route by a single $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, by a* $\mathsf{LimitSelector}^{\uparrow}$ *row with only one candidate, by a host equality of Real numbers, by quotient stream equality, or by an ambient Real-completeness theorem.*

Proof. Begin with the obligation surface of Definition 1039.2. It gives every accepted uniqueness read the two displayed seal rows, shared comparison row, finite window row, dyadic ledger, Real classifier, uniqueness handoff, transport, route, provenance, and local name.

The first exclusion is single-candidate escape. A lone $\mathsf{CauchyLimitSeal}^{\uparrow}$ row cannot supply both L_0 and L_1, and a $\mathsf{LimitSelector}^{\uparrow}$ row that carries only one selected candidate cannot fill the second seal coordinate.

The second exclusion is ambient equality escape. Host Real equality, quotient stream equality, and a global Real-completeness theorem are not rows of U; the only equality-facing read is the local uniqueness row V after L_0, L_1, Q, S, D, R have been exposed. □

Lean4 ✓ `BEDC.Derived.RealLimitUniquenessSealUp.RealLimitUniquenessSealCarrier nonescape`

Theorem 1039.4 (Real limit uniqueness seal consumer handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing, or regular-limit consumer of an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet receives only the displayed carrier rows and the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row. In particular, downstream consumers can cite the two-seal comparison route, but they cannot extract a selected limit, quotient stream equality, host Real equality, detached selector, or ambient completeness principle from the packet.*

Lean4 ✓ `BEDC.Derived.RealLimitUniquenessSealUp.RealLimitUniquenessSealCarrier consumer handoff`

Proof. Use Definition 1039.2 to identify the public handoff. The consumer receives the displayed packet $U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$ and no additional carrier coordinate.

Then apply the nonescape result of Definition 1039.3. Since every uniqueness read passes through the two candidate seals and the shared comparison-window-tolerance route, the downstream consumer can replay only that finite route.

Thus the handoff is local to the packet and suitable for later Real-completion consumers without importing selected limits, quotient streams, host equality, detached selectors, or ambient completeness. □

Theorem 1039.5 (Real limit uniqueness selector-seal nonescape)**.** *Any accepted consumer that tries to read* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *uniqueness through a selector-seal route must still expose the full displayed inventory: the two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* L_0, L_1*, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row* Q*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{Real}^{\uparrow}$ *classifier row* R*, the* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *row* V*, and the structural rows* H, C, P, N*.*

A single selector row or selector-seal shortcut cannot replace this inventory. The packet grants no selected host limit, quotient stream equality, detached selector choice, ambient completeness theorem, or Real equality read outside the displayed rows.

Proof. Start from Definition 1039.2. The local NameCert surface is the packet

$$U = (L_0, L_1, Q, S, D, R, V, H, C, P, N),$$

so any accepted uniqueness read has to expose the two seal rows, shared comparison row, finite window row, dyadic tolerance row, Real classifier row, regular-limit uniqueness row, and structural rows.

Next apply Definition 1039.3. It forbids replacing the displayed route by a single $\mathsf{CauchyLimitSeal}^{\uparrow}$ row, a one-sided $\mathsf{LimitSelector}^{\uparrow}$ row, host Real equality, quotient stream equality, or an ambient Real-completeness theorem.

Finally use Definition 1039.4. A downstream $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{CauchyLimitSeal}^{\uparrow}$-facing consumer receives only the displayed packet and the local NameCert row. Therefore routing the read through a selector-seal surface cannot remove any row from $L_0, L_1, Q, S, D, R, V, H, C, P, N$ or create a shortcut around the uniqueness inventory. □

Theorem 1039.6 (Real limit uniqueness seal independent witness)**.** *The* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet is independent, as a finite NameCert surface, from* $\mathsf{RegularLimitUniqueness}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RealCompleteness}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *alone. Each listed sibling supplies a proper part or surrounding route role, while Definition 1039.1 binds the two completion seals, shared regular-rational comparison, finite StreamName window, dyadic ledger, Real classifier, uniqueness handoff, transport, route, provenance, and local naming in one packet.*

Proof. Inspect the carrier rows. The $\mathsf{RegularLimitUniqueness}^{\uparrow}$ row V lacks the two displayed completion-seal rows and the route through Q, S, D, R, while each $\mathsf{CauchyLimitSeal}^{\uparrow}$ row supplies only one endpoint of the comparison.

The $\mathsf{LimitSelector}^{\uparrow}$ surface can name a selected candidate but does not carry the second candidate seal or the shared comparison budget. The $\mathsf{RealCompleteness}^{\uparrow}$ vision supplies a route lattice rather than a local carrier, and $\mathsf{Real}^{\uparrow}$ supplies the terminal classifier without the completion-seal comparison ledger.

Therefore none of the listed siblings is bijective to the displayed packet. The packet's independence witness is exactly the simultaneous presence of $L_0, L_1, Q, S, D, R, V, H, C, P, N$ under one local NameCert row. □

Theorem 1039.7 (Real limit uniqueness seal two-seal exposure)**.** *Every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet exposes both* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* L_0 *and* L_1 *before any uniqueness read through* V *is available. Neither a one-sided completion seal nor a selector-facing endpoint can stand in for the missing seal row.*

Proof. Use the carrier definition of Definition 1039.1. The packet is the displayed tuple $U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$, so both completion-seal coordinates are part of the admitted carrier before the uniqueness coordinate can be read.

Apply the obligation surface of Definition 1039.2. It lists carrier admission and the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ candidate rows as separate local NameCert obligations, before the shared comparison, window, dyadic ledger, Real classifier, and uniqueness handoff.

The nonescape theorem, Definition 1039.3, rules out replacing this exposure by a single seal, a one-sided selector, quotient stream equality, host Real equality, or ambient completeness. The consumer handoff of Definition 1039.4 then says that downstream consumers receive only the displayed packet and local NameCert row.

Finally, Definition 1039.6 identifies the simultaneous presence of L_0 and L_1 as part of the independence witness. Thus any accepted uniqueness read must expose both seal rows first. □

Theorem 1039.8 (Real limit uniqueness seal shared comparison window)**.** *For every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet, the two seal candidates share the route through the regular-rational comparison row* Q*, the finite window row* S*, the dyadic tolerance row* D*, the Real classifier row* R*, and the uniqueness row* V*. The packet admits no detached selector, quotient stream, or external comparison stream as a substitute for this shared route.*

Proof. Start from the carrier of Definition 1039.1. The only comparison-facing coordinates between the two seals are Q, S, D, R, V, with route replay recorded by C and provenance recorded by P.

The obligation surface in Definition 1039.2 orders these rows as one finite route: the shared $\mathsf{RegSeqRat}^{\uparrow}$ comparison, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ classifier, and $\mathsf{RegularLimitUniqueness}^{\uparrow}$ handoff.

Apply Definition 1039.3 to exclude a detached selector, quotient stream equality, host Real equality, and ambient completeness. Then use Definition 1039.4: a downstream consumer can replay only the rows that the packet hands off.

The independent witness theorem, Definition 1039.6, separates this packet from $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCompleteness}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ alone exactly because the shared route is present with both seals. Hence every accepted comparison must travel through Q, S, D, R, V. □

Theorem 1039.9 (Real limit uniqueness seal classifier stability)**.** *Componentwise* hsame *transport through H preserves the two-seal comparison, dyadic tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *classifier row, uniqueness handoff,* Cont *route, provenance, and local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row of an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet. Transport cannot change the packet into a one-seal certificate, a selector certificate, or a quotient-stream certificate.*

Proof. Read H in the carrier definition of Definition 1039.1. It records componentwise hsame transport for the two seal rows, comparison row, window row, dyadic ledger, Real classifier, uniqueness row, route row, provenance row, and naming row.

The obligation surface of Definition 1039.2 makes those transported rows part of the local NameCert surface. Thus the transported packet still carries the two-seal comparison, the dyadic tolerance ledger, the $\mathsf{Real}^{\uparrow}$ classifier row, the uniqueness handoff, the displayed Cont route, provenance, and local naming.

Use Definition 1039.3 and Definition 1039.4 to control consumers after transport. Since the handoff contains only the displayed rows, no consumer can read a one-seal certificate, selector certificate, quotient stream, host equality, or ambient completeness principle from the transported packet.

Finally, Definition 1039.6 identifies the packet by the simultaneous presence of $L_0, L_1, Q, S, D, R, V, H, C, P, N$. Componentwise hsame transport preserves that inventory rather than changing the certificate kind. □

Theorem 1039.10 (Real limit uniqueness seal nontrivial witness)**.** *The displayed* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *carrier admits two distinguishable finite packets. They may be separated by changing the second* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L_1 while keeping the first seal row and the remaining route fixed, or by changing the shared comparison-window route through Q, S while keeping the two seal rows fixed. Hence the local carrier is not a one-element shell.*

Proof. Begin with the carrier inventory of Definition 1039.1. A packet is the finite tuple

$$U = (L_0, L_1, Q, S, D, R, V, H, C, P, N),$$

so the second seal row and the shared comparison-window rows are visible coordinates rather than hidden consequences.

The obligation surface of Definition 1039.2 makes L_1 and the route through Q, S separate NameCert obligations. The nonescape theorem, Definition 1039.3, prevents an accepted consumer from collapsing those rows into a single seal, a selector endpoint, quotient stream equality, host Real equality, or ambient completeness.

Finally apply the independence witness of Definition 1039.6. Its separation from the sibling carriers is exactly the simultaneous presence of the two seal rows and the shared route. Varying either the second seal row or the shared comparison-window row therefore gives a distinguishable finite packet, so the carrier is nontrivial. □

Theorem 1039.11 (Real limit uniqueness seal field-faithful projection)**.** *For accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packets, equality of packets is forced by equality of all displayed fields $L_0, L_1, Q, S, D, R, V, H, C, P, N$. Conversely, the displayed field tuple determines the packet and leaves no extra coordinate for selected host limits, quotient stream equality, detached selector choice, ambient completeness, or host Real equality.*

Proof. Read the packet through Definition 1039.1. The carrier is presented as the tuple $U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$, and every named coordinate is part of the finite BHist packet.

The forward direction follows from Definition 1039.2: any accepted packet equality must preserve the two CauchyLimitSeal rows, shared comparison row, window row, dyadic ledger, Real classifier, uniqueness row, transport, route, provenance, and local name. There is no accepted packet coordinate outside that list.

For the reverse direction, suppose all displayed fields agree. The nonescape theorem, Definition 1039.3, rules out any hidden equality read or external completeness route that could distinguish two packets after their displayed rows agree.

Use Definition 1039.6 to close the determination claim. The packet is independent from its siblings precisely because the whole displayed inventory is bound in one local NameCert row. Thus the field tuple determines the packet, and the projection to fields is faithful. □

Theorem 1039.12 (Real limit uniqueness seal selector-route coverage)**.** *Any selector-facing uniqueness read of an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet must expose the full displayed route*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N.$$

It cannot substitute a one-sided selector row, a single completion-seal endpoint, or a detached selector choice for the two-seal comparison route.

Proof. Apply the selector-seal nonescape result of Definition 1039.5. It says that even a selector-facing route has to expose the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows, the shared comparison row, the finite window row, the dyadic tolerance row, the $\mathsf{Real}^{\uparrow}$ classifier row, the regular-limit uniqueness row, and the structural rows.

The two-seal exposure theorem, Definition 1039.7, removes the one-sided selector case. The shared comparison window of Definition 1039.8 then forces the selector-facing read through Q, S, D, R, V rather than through an external comparison stream.

Finally use the consumer handoff of Definition 1039.4. A downstream consumer receives only the displayed packet and the local NameCert row, so no detached selector choice or single endpoint can replace $L_0, L_1, Q, S, D, R, V, H, C, P, N$. □

Theorem 1039.13 (Real limit uniqueness seal completion consumer totality)**.** *Every Real-completion-facing consumer of an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet receives only the finite two-seal comparison packet* $L_0, L_1, Q, S, D, R, V, H, C, P, N$ *and the local naming surface. Such a consumer cannot export a selected limit, quotient stream equality, host Real equality, detached selector choice, or ambient completeness principle from the packet.*

Proof. Begin with the nonescape theorem, Definition 1039.3. It forces every accepted uniqueness read through the finite displayed route and excludes single-seal, selector-only, host-equality, quotient-stream, and ambient-completeness escapes.

Use Definition 1039.4 to identify the exported interface. The consumer receives the packet rows and local NameCert row, not an additional completion object or a selected host limit.

Classifier stability, Definition 1039.9, keeps this interface unchanged under componentwise hsame transport. The field-faithful projection of Definition 1039.11 then says that the displayed fields exhaust the packet, leaving no hidden coordinate from which a consumer could export quotient streams, host Real equality, detached selector data, or ambient completeness. □

Theorem 1039.14 (Real limit uniqueness seal Real-classifier boundary)**.** *In every accepted* RealLimitUniquenessSea *packet, the* $\mathsf{Real}^{\uparrow}$ *classifier row* R *is read after the dyadic tolerance row* D *and before the uniqueness*

handoff row V. No host Real equality, quotient stream equality, detached selector choice, or global completeness route can replace this D, R, V boundary.

Proof. Apply the shared comparison window theorem, Definition 1039.8. It identifies the comparison route through Q, S, D, R, V, with the dyadic tolerance ledger before the Real classifier and the uniqueness handoff after it.

Classifier stability, Definition 1039.9, preserves the $\mathsf{Real}^{\uparrow}$ classifier row together with the dyadic ledger, uniqueness handoff, route, provenance, and local NameCert row under componentwise transport. Thus transport cannot move the classifier outside the displayed comparison route or turn it into a selector certificate.

Finally use completion consumer totality from Definition 1039.13. Since completion-facing consumers receive only the finite two-seal packet, the Real-classifier read cannot be replaced by host equality, quotient stream equality, detached selector choice, or a global completeness route. □

Theorem 1039.15 (Real limit uniqueness seal TasteGate handoff)**.** *The same displayed field inventory $L_0, L_1, Q, S, D, R, V, H, C, P, N$ is the paper-side obligation consumed by the Lean* `Nontrivial`*,* `FieldFaithful`*, and single-carrier TasteGate instances for* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$*.*

Proof. The carrier definition in Definition 1039.1 supplies the single-carrier inventory handed to Lean: two completion-seal rows, the shared regular-rational comparison row, finite window row, dyadic tolerance row, Real classifier row, uniqueness row, transport, route, provenance, and local naming.

The obligation surface of Definition 1039.2 identifies the same rows as the NameCert surface. This is the field inventory used by the field-faithful projection, because no accepted packet has more fields than $L_0, L_1, Q, S, D, R, V, H, C, P, N$.

The nontriviality handoff is controlled by Definition 1039.3 and Definition 1039.6. Since the packet cannot escape to a one-seal or selector-only surface, and since its independence is witnessed by the simultaneous two-seal comparison route, the TasteGate instances consume the same concrete obligation rather than a separate carrier. □

Theorem 1039.16 (Real limit uniqueness seal terminal classifier exactness)**.** *For every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet, the terminal* $\mathsf{Real}^{\uparrow}$ *classifier row R is available exactly through the displayed route*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N.$$

The route uses the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows L_0, L_1, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row Q, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row S, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before R is public. It excludes host Real equality, quotient streams, selected limits, and ambient completeness as classifier sources.*

Proof. Start with the obligation surface of Definition 1039.2. It lists $L_0, L_1, Q, S, D, R, V, H, C, P, N$ as the complete local NameCert surface and places the terminal $\mathsf{Real}^{\uparrow}$ classifier after the two seal rows, the shared comparison row, the finite window, and the dyadic ledger.

The two-seal exposure theorem, Definition 1039.7, prevents a one-sided completion seal from supplying R. The shared comparison window of Definition 1039.8 then forces the classifier read through Q, S, D rather than through an external stream.

Use nonescape, Definition 1039.3, and consumer handoff, Definition 1039.4, to exclude host equality, quotient streams, selected limits, detached selectors, and ambient completeness from the public classifier interface. Finally apply classifier stability, Definition 1039.9, so componentwise hsame transport preserves the same R row inside the displayed route. □

Theorem 1039.17 (Real limit uniqueness seal CauchyLimit pair readback)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of a* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet recovers both seal rows* L_0, L_1 *and the shared comparison route* Q, S, D *before uniqueness through* V *is public. No consumer can read uniqueness from a single* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, from a detached selector, or from a route that omits the shared comparison-window-tolerance ledger.*

Proof. Apply Definition 1039.4. A $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer receives only the displayed packet and the local NameCert row, so every public read is a read of the finite inventory rather than an external object.

The two-seal exposure theorem, Definition 1039.7, recovers both L_0 and L_1 before uniqueness can be read. The shared comparison window of Definition 1039.8 recovers Q, S, D as the common route that compares the two sealed candidates.

Use Definition 1039.2 to place the uniqueness handoff V after those rows in the local NameCert surface. Then apply Definition 1039.3 and Definition 1039.9 to rule out one-seal, selector-only, quotient-stream, host-equality, and ambient-completeness substitutions under transport. □

Theorem 1039.18 (Real limit uniqueness seal Phase Real boundary)**.** *The route of Definition 1039.1 is the current Phase Real and completion uniqueness boundary for* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$*. It consumes* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *through the displayed packet, but it does not claim Real completeness, selected-limit existence, quotient stream equality, or host Real equality.*

Proof. Read the carrier and obligation surface together. By Definition 1039.2, the packet consumes the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows, the $\mathsf{RegSeqRat}^{\uparrow}$ comparison row, the $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{Real}^{\uparrow}$ classifier row, and the $\mathsf{RegularLimitUniqueness}^{\uparrow}$ handoff row.

The readback obligations are fixed by Definition 1039.7 and Definition 1039.8. Together they make the boundary finite: both seal rows and the shared route through Q, S, D, R, V must appear before a consumer can cite uniqueness.

Finally apply Definition 1039.3, Definition 1039.4, and Definition 1039.9. These results keep the public boundary inside the displayed packet and exclude Real completeness, selected limits, quotient stream equality, host equality, detached selectors, and any classifier source outside the transported NameCert surface. □

Theorem 1039.19 (Real limit uniqueness seal route triad)**.** *Every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet exposes the route triad of two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* L_0, L_1*, a shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison route through* Q, S, D*, and a terminal* $\mathsf{Real}^{\uparrow}$ *classifier row* R *before the uniqueness row* V *is read.*

Proof. Start from the obligation surface of Definition 1039.2. It lists the two seal rows, shared comparison row, finite window row, dyadic tolerance row, Real classifier, and uniqueness handoff as displayed obligations of one packet.

Apply the two-seal exposure theorem, Definition 1039.7. It places L_0 and L_1 before any uniqueness read can be public.

Then apply the shared comparison window, Definition 1039.8. It routes the comparison through Q, S, D, R, V, and the consumer handoff of Definition 1039.4 keeps downstream reads inside that displayed triad. □

Theorem 1039.20 (Real limit uniqueness seal window exactness)**.** *For every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *are the exact finite*

observation route for the shared comparison between L_0 and L_1. No detached selector, quotient stream, host Real equality, or ambient completeness principle can replace the S, D observation route.

Proof. Use Definition 1039.8. It identifies the shared route as Q, S, D, R, V, with the finite window and dyadic tolerance rows between the regular-rational comparison and the terminal Real classifier.

The nonescape theorem, Definition 1039.3, removes the external routes through host equality, quotient streams, one-sided selectors, and ambient completeness. The consumer handoff of Definition 1039.4 says that downstream consumers receive only the displayed packet rows.

Finally apply the field-faithful projection theorem, Definition 1039.11. Since the displayed fields exhaust the packet, the observation route for the shared comparison is exactly the S, D window-and-tolerance surface and no hidden route remains. □

Theorem 1039.21 (Real limit uniqueness seal RegSeqRat endpoint determinacy)**.** *For every accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet* $U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$*, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q determines both endpoint reads exposed by the two seal rows L_0, L_1 before uniqueness through V is public. No endpoint can be supplied by a detached selector, quotient stream, host Real equality, or ambient completeness principle.*

Proof. Read the packet through Definition 1039.1. The two completion-seal rows and the shared regular-rational comparison row occur in the same finite carrier.

Use Definition 1039.8. It places Q before the finite window, dyadic tolerance, Real classifier, and uniqueness handoff, so both endpoint reads pass through the same comparison row.

Then apply Definition 1039.20, Definition 1039.16, and Definition 1039.3. These results keep endpoint determinacy inside Q, S, D, R, V and exclude every external endpoint source. □

Theorem 1039.22 (Real limit uniqueness seal dyadic Real classifier lock)**.** *In an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the terminal* $\mathsf{Real}^{\uparrow}$ *classifier row R lock the classifier boundary before the uniqueness handoff V is read. A consumer cannot replace this D, R boundary by Real completeness, host Real equality, quotient stream equality, a detached selector, or a selected host limit.*

Proof. Start from Definition 1039.1. The D and R rows are explicit coordinates of the same packet as the two seal rows and the uniqueness handoff.

Apply Definition 1039.20. The dyadic row is the exact tolerance component of the finite observation route, so classifier reads cannot skip from Q directly to V.

Finally use Definition 1039.16 and Definition 1039.3. The terminal classifier is available only through the displayed route, and every external Real or stream equality source is outside the packet. □

Theorem 1039.23 (Real limit uniqueness seal StreamName window totality)**.** *The finite* $\mathsf{StreamName}^{\uparrow}$ *row S is the only observation window exposed to a two-seal consumer of* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$*. Every comparison of L_0 and L_1 that reaches the uniqueness handoff must pass through Q, S, D, R, V, and no consumer receives an additional stream window, quotient stream classifier, selected-limit witness, or host completeness route.*

Proof. Project the accepted carrier with Definition 1039.1. It records exactly one StreamName window coordinate S between the shared RegSeqRat comparison row and the dyadic tolerance row.

Use Definition 1039.8 and Definition 1039.20. Together they make S the finite observation window for the two-seal comparison route.

Close with Definition 1039.16 and Definition 1039.3. The terminal classifier and nonescape clauses leave no hidden stream, selector, quotient, or completeness observation window outside the displayed packet. □

Theorem 1039.24 (Real limit uniqueness seal consumer exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing uniqueness consumer of an accepted* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *packet receives only*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N.$$

It receives no selected host limit, quotient stream equality, host Real equality, detached selector, or ambient completeness principle.

Proof. Apply Definition 1039.4. It identifies the exported interface as the displayed carrier rows and the local NameCert row, with no additional completion object.

Next use Definition 1039.19 and Definition 1039.20. Together they expose the two-seal route, the shared comparison, and the exact finite observation rows before the uniqueness handoff is available.

Finally combine nonescape, Definition 1039.3, with field-faithful projection, Definition 1039.11. The packet has no hidden coordinate from which a consumer could extract a selected host limit, quotient stream equality, host Real equality, detached selector, or ambient completeness principle. □

The later Lean boundary is `lean4/BEDC/Derived/RealLimitUniquenessSealUp/TasteGate.lean`. It must provide `Nontrivial RealLimitUniquenessSealUp` and `FieldFaithful RealLimitUniquenessSealUp` instances for the same displayed carrier.

Theorem 1039.25 (Real limit uniqueness seal primitive scope route)**.** *Every obligation row of the displayed packet*

$$U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$$

is scoped to concrete BEDC primitives: the two $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows* L_0, L_1*, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row* Q*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *classifier row* R*, the* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *handoff row* V*, componentwise* hsame *transport through* H*, the displayed* Cont *route* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row* N*.*

The route does not pass through selected host limits, quotient stream equality, detached selector choices, host Real equality, or ambient Real completeness.

Proof. Begin with the obligation surface of Definition 1039.2. It identifies every local NameCert row of U as one of the displayed carrier, two completion-seal rows, shared comparison, finite window, dyadic ledger, Real classifier, uniqueness handoff, transport, route, provenance, and local naming obligations.

The nonescape theorem, Definition 1039.3, keeps that surface inside the packet. It excludes a single completion seal, a one-sided selector, quotient stream equality, host Real equality, and ambient completeness as substitutes for the displayed route.

Next apply the two-seal exposure theorem and the shared comparison window theorem. By Definition 1039.7, both L_0 and L_1 are exposed before any uniqueness read, and by Definition 1039.8, their comparison travels through Q, S, D, R, V.

Classifier stability, Definition 1039.9, transports the same rows through componentwise hsame without changing the packet into another certificate kind. The nontrivial witness of Definition 1039.10 confirms that the two-seal and shared-window coordinates are genuine finite coordinates rather than a one-element shell.

Finally use field-faithful projection, Definition 1039.11. Equality and determination of accepted packets are exhausted by the fields $L_0, L_1, Q, S, D, R, V, H, C, P, N$, so the primitive route is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert binding. □

Theorem 1039.26 (Real limit uniqueness seal public route surface)**.** *The public consumer route for* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *exports exactly the two completion-seal rows* L_0, L_1*, the shared comparison row* Q*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *classifier row* R*, the uniqueness handoff row* V*, componentwise transport* H*, displayed continuation* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row* N*. It exports no selected host limit, quotient stream equality, detached selector, host Real equality, ambient completeness theorem, or additional completion object.*

Proof. Start with the primitive scope route, Definition 1039.25. It binds every displayed obligation row to the concrete BEDC primitives $L_0, L_1, Q, S, D, R, V, H, C, P, N$ and excludes selected limits, quotient streams, detached selectors, host Real equality, and ambient Real completeness from the route.

Next use the nontrivial witness, Definition 1039.10. The two completion-seal rows and the shared comparison-window route are genuine finite coordinates, so the public consumer surface cannot collapse to a one-row shell or a selector-only certificate.

Finally apply field-faithful projection, Definition 1039.11. Equality and determination of accepted packets are exhausted by the displayed fields, and the consumer exhaustion theorem keeps downstream reads inside the same packet. Therefore the public route exports exactly the listed rows and no external Real or Completion object. □

Theorem 1039.27 (Real limit uniqueness bridged seal surface)**.** *The bridge-facing* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *surface is exactly the packet*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N.$$

Here L_0 *and* L_1 *are the two* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows,* Q *is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row,* S *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{Real}^{\uparrow}$ *classifier row,* V *is the* $\mathsf{RegularLimitUniqueness}^{\uparrow}$ *handoff,* H *is componentwise* hsame *transport,* C *is the displayed* Cont *route,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$ *row.*

This surface provides the two-seal uniqueness route needed by Real and Completion consumers. It still provides no selected host limit, quotient stream equality, host Real equality, detached selector choice, or ambient Real completeness.

Proof. Begin with the obligation surface, Definition 1039.2. It lists the carrier row, the two completion-seal rows, the shared comparison row, the finite window row, the dyadic tolerance ledger, the Real classifier, the regular-limit uniqueness handoff, componentwise transport, continuation, provenance, and local NameCert row as the complete packet inventory.

The nonescape theorem, Definition 1039.3, keeps every consumer inside that inventory. It rules out selected host limits, quotient stream equality, host Real equality, detached selector choices, and ambient Real completeness as routes for reading the uniqueness surface.

Next apply the primitive-scope route and the public-route surface. By Definition 1039.25, every displayed row is tied to the concrete BEDC primitives BHist, hsame, Cont, Pkg, and NameCert. By Definition 1039.26, the same rows are exactly what a public consumer receives.

Finally use the nontrivial witness and field-faithful projection. The witness of Definition 1039.10 prevents the two-seal surface from collapsing to a one-row shell, and Definition 1039.11 says that accepted packets are determined componentwise by $L_0, L_1, Q, S, D, R, V, H, C, P, N$. Therefore the bridge-facing surface is precisely the displayed two-seal uniqueness packet and carries none of the excluded host or completeness data. □

Theorem 1039.28 (Real limit uniqueness seal formal TasteGate boundary)**.** *The displayed packet rows*

$$L_0, L_1, Q, S, D, R, V, H, C, P, N$$

are exactly the paper-side boundary consumed by the checked Lean TasteGate route `BEDC.Derived.RealLimitUniqu` *together with the checked carrier-obligation surface and nonescape rows for* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$.

This boundary records a formal TasteGate route for the finite two-seal uniqueness packet. It does not claim Real completeness, selected limits, quotient stream equality, host Real equality, or bridge checking.

Proof. Start with the carrier definition, Definition 1039.1. It names the full finite packet $U = (L_0, L_1, Q, S, D, R, V, H, C, P, N)$ and leaves no coordinate for selected host limits, quotient streams, host Real equality, detached selectors, or ambient completeness.

Apply the checked obligation surface, Definition 1039.2, and the checked nonescape theorem, Definition 1039.3. These two results identify the NameCert rows consumed by the packet and keep every accepted uniqueness read inside the displayed two-seal route.

Next use the nontrivial witness, Definition 1039.10, the field-faithful projection, Definition 1039.11, and the primitive-scope route, Definition 1039.25. They show that the two seal rows and shared comparison-window route are genuine finite coordinates, that accepted packets are determined by the listed fields, and that every row is scoped to the displayed BHist, hsame, Cont, Pkg, and NameCert primitives.

Finally combine the public-route surface, Definition 1039.26, with the checked Lean TasteGate route named in the statement. The Lean route consumes the same carrier, carrier-obligation, nonescape, nontrivial, and field-faithful evidence; the paper boundary is therefore exactly the displayed packet rows, with no additional Real, Completion, quotient, host-equality, or bridge assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ packet is generated from two CauchyLimitSeal rows, a shared RegSeqRat comparison row, a finite StreamName window row, a DyadicRatCore tolerance ledger, a RealUp classifier row, a RegularLimitUniqueness row, componentwise transports, displayed continuation, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1039.1, Definition 1039.2, Definition 1039.3, Definition 1039.4, Definition 1039.10, Definition 1039.11, Definition 1039.25, Definition 1039.18, Definition 1039.24, Definition 1039.26, and Definition 1039.28 bind the two-seal carrier, shared comparison row, finite StreamName window, DyadicRatCore tolerance, RealUp classifier, uniqueness handoff, nontrivial witness, field-faithful witness, primitive route, Phase Real boundary, consumer exhaustion surface, public consumer route, and formal TasteGate boundary to the displayed BHist, hsame, Cont, Pkg, and NameCert primitives.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealLimitUniquenessSeal}^{\uparrow}}$, theoremChecked)

Bridge status: none

Origin: ai

Lean target: `BEDC.Derived.RealLimitUniquenessSealUp.taste\_gate`

Not claimed: This chapter does not prove Real completeness, selected-limit existence, quotient stream equality, host Real equality, or bridge checking for RealLimitUniquenessSeal$^{\uparrow}$.

Upgrade path: No further theory-axis upgrade is claimed beyond the recorded formal and bridge verification boundaries.

Chapter 1040

A Concrete Naming Certificate for $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$

The $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$ packet realizes the separated uniqueness clause of chapter 9238 as a finite diagonal-comparison witness. It sits after the real naming surface of chapter 38, the regular-rational readback surface of chapter 301, and the two-seal uniqueness route of chapter 1039. Its carrier records the point at which two diagonal limit candidates are compared through the same finite windows, not a quotient of regular streams and not a host equality of completed real numbers.

Definition 1040.1 (Real limit uniqueness witness carrier)**.** *A* $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D_0, D_1, Q, S, R_0, R_1, E, V, H, C, P, N)$$

with the following displayed rows:

1. D_0 *and* D_1 *are the two diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *candidate rows read from the same regular-Cauchy source;*
2. Q *is the finite diagonal-comparison row bounding the pointwise difference of the two candidates;*
3. S *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row through which the comparison is observed;*
4. R_0 *and* R_1 *are the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached from the compared regular-rational candidates;*
5. E *is the separated RealUp classifier row witnessing the* psame *identification of the two seals;*
6. V *is the sibling* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *route row that packages the same two-seal comparison as a public uniqueness surface;*
7. H *records componentwise* hsame *transport for the diagonal rows, comparison row, finite window, two RealUp seals, separated classifier, seal route, replay route, provenance row, and local naming row;*
8. C *records the displayed* Cont *route from* D_0, D_1 *through* Q, S, R_0, R_1, E, V*;*
9. P *is the* Pkg *provenance over chapter 9238, chapter 38, chapter 301, chapter 1039,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected host limits, quotient stream equality, countable choice, ambient Real completeness, or host Real equality.

Lean4 ✓ BEDC.Derived.RealLimitUniquenessWitnessUp.TasteGate.RealLimitUniquenessWitnessTasteGate
single carrier alignment

Theorem 1040.2 (Real limit uniqueness witness obligation surface)**.** *For the carrier of Definition 1040.1, the local* $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$W = (D_0, D_1, Q, S, R_0, R_1, E, V, H, C, P, N),$$

diagonal candidate exposure through D_0, D_1*, finite comparison through* Q*, finite window exposure through* S*, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through* R_0, R_1*, separated classifier exposure through* E*, the* $\mathsf{RealLimitUniquenessSeal}^{\uparrow}$ *handoff through* V*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier rows. The tuple supplies exactly the two diagonal candidate rows, their comparison row, the observation window, the two RealUp seals, the separated classifier, the uniqueness-seal handoff, transport, replay, provenance, and local naming.

Read the comparison in dependency order. The diagonal candidates must be exposed before the finite comparison row can be interpreted, the window row must be exposed before the comparison is observed, and the two RealUp seals must be reached before the separated classifier row is consumed.

The remaining structural rows preserve and name the same route. Since the carrier has no coordinate for quotient streams, selected host limits, ambient completeness, choice, or host equality, the displayed rows exhaust the local NameCert obligations. □

Theorem 1040.3 (Real limit uniqueness witness nonescape)**.** *Every accepted* $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$ *consumer that reads uniqueness of two Real-completion candidates factors through the finite route*

$$D_0, D_1, Q, S, R_0, R_1, E, V, H, C, P, N.$$

The consumer cannot replace this route by a single $\mathsf{Real}^{\uparrow}$ *seal, a quotient-stream equality, a selected host limit, an ambient Real-completeness theorem, or a host Real equality judgment.*

Proof. Start with the obligation surface of Definition 1040.2. It gives every accepted uniqueness read the two diagonal candidates, their finite comparison row, the shared observation window, the two terminal RealUp seals, the separated classifier row, the uniqueness-seal handoff, and the structural rows.

Single-endpoint escape is impossible because neither R_0 nor R_1 alone contains the paired diagonal comparison Q or the separated classifier row E. The witness is therefore a two-candidate packet, not a detached real-name seal.

Ambient equality escape is also impossible. Quotient streams, selected host limits, global completeness, and host Real equality are absent from the carrier, so every consumer read remains on the displayed diagonal-comparison route. □

Theorem 1040.4 (Real limit uniqueness witness consumer handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$ *packet receives a finite diagonal-comparison, seal, classifier, and NameCert witness packet. The handoff exposes* $D_0, D_1, Q, S, R_0, R_1, E, V, H, C, P,$ *and no selected limit, quotient stream, host equality, detached selector, countable-choice row, or ambient completeness principle.*

Proof. Use the obligation surface to identify the packet given to the consumer. The consumer receives the two candidate diagonals, finite comparison, shared window, two RealUp seals, separated classifier, RealLimitUniquenessSeal route, transport, replay, provenance, and local naming.

Then apply nonescape. The packet's uniqueness read is already bounded by the paired diagonal-comparison route, so a downstream RealUp or completion consumer cannot import a host equality or quotient comparison after the handoff.

Thus the handoff is exactly the finite witness packet required by the Real-completeness uniqueness clause, with every dependency visible through BHist, hsame, Cont, Pkg, and NameCert rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitUniquenessWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$ packet is generated from two diagonal RegSeqRat candidate rows, a finite comparison row, a StreamName window row, two RealUp seals, a separated RealUp classifier, a RealLimitUniquenessSeal route, componentwise transports, displayed continuation, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLimitUniquenessWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1040.1, Definition 1040.2, Definition 1040.3, and Definition 1040.4 specify the finite diagonal-comparison witness surface and its nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLimitUniquenessWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove Real completeness, selected-limit existence, quotient stream equality, host Real equality, Lean verification, or bridge checking for $\mathsf{RealLimitUniquenessWitness}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, diagonal comparison exposure, finite window exposure, two RealUp seal exposure, separated classifier exposure, uniqueness-seal handoff, nontriviality, and field faithfulness for the same packet.

Chapter 1041

A Concrete Naming Certificate for $\mathsf{CofinalDiagonalSeal}^{\uparrow}$

The $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ packet realizes the cofinal-tail-to-diagonal $\mathsf{Real}^{\uparrow}$ seal route described in chapter 9238 and its transported-stationary-diagonal context. It sits after the cofinal-tail budget of chapter 698, the diagonal selector budget of chapter 853, the diagonal limit budget of chapter 896, the regular-source handoff of chapter 301, the finite-window source of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite route seal, not a proof of Real completeness, quotient-stream equality, selected host limits, or ambient completion.

Falsifiable prediction: Removing the cofinal-tail row permits two diagonal routes with incompatible tail thresholds to classify as the same Real seal; removing the Real seal row leaves no terminal public handoff.

Independence witness: cofinaltailbudget, cauchydiagonalbudget, diagonallimitbudget, regularcauchytailschedule, realsealcomparison: no listed sibling is bijective to $\mathsf{CofinalDiagonalSeal}^{\uparrow}$. The cofinal-tail budget stores the shared threshold without the diagonal selector and terminal seal; the Cauchy diagonal budget stores the selector without the cofinal threshold; the diagonal limit budget stores the limit-facing budget without the full cofinal route; the regular Cauchy tail schedule stores tail service without the diagonal handoff; and the real seal comparison stores the terminal comparison without the upstream budget lattice.

Definition 1041.1 (Cofinal diagonal seal carrier)**.** *A* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\epsilon, T, D, L, R, W, Y, A, H, C, P, N)$$

with the following displayed rows:

1. ϵ *is the finite precision request served by the route;*
2. T *is the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *row fixing the cofinal tail threshold;*
3. D *is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *row fixing the diagonal selector budget;*
4. L *is the* $\mathsf{DiagonalLimitBudget}^{\uparrow}$ *row naming the diagonal-limit-facing budget;*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-source readback row consumed by the seal;*
6. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the route is observed;*

7. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and dyadic-readback ledger;*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row made public by the route;*
9. *H records componentwise* hsame *transport for the precision, budget, readback, window, dyadic, seal, route, provenance, and naming rows;*
10. *C records the displayed* Cont *replay route from cofinal tail threshold through diagonal budget, diagonal limit budget, regular-source readback, finite window, dyadic ledger, and terminal Real seal;*
11. *P is the* Pkg *provenance over* $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DiagonalLimitBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, hidden tail choice, ambient completeness, or a Real-facing handoff detached from the displayed route.

Lean4 (definition) `BEDC.Derived.CofinalDiagonalSealUp.CofinalDiagonalSealUp`

Theorem 1041.2 (Cofinal diagonal seal NameCert obligations)**.** *For the carrier of Definition 1041.1, the local* $\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (\epsilon, T, D, L, R, W, Y, A, H, C, P, N),$$

precision exposure through ϵ, cofinal-tail budget exposure through T, diagonal selector exposure through D, diagonal-limit budget exposure through L, regular-source readback through R, finite-window exposure through W, dyadic tolerance exposure through Y, terminal Real seal exposure through A, componentwise transport through H, displayed route replay through C, package provenance through P, and local naming through N.

Lean4 ✓ `BEDC.Derived.CofinalDiagonalSealUp.CofinalDiagonalSealTasteGate single carrier alignment`

Proof. Project the displayed carrier. The tuple exposes the precision request, cofinal-tail budget, diagonal budget, diagonal-limit budget, regular-source row, finite-window row, dyadic ledger, terminal seal, transport, replay, provenance, and local naming rows.

Read the route in dependency order. The cofinal-tail row T fixes the shared threshold, the diagonal budget row D fixes the selector side, and the diagonal-limit budget row L records the limit-facing budget before any regular-source or Real-facing read is available.

The rows R, W, Y, A then supply the observable route to the public seal: regular-source readback, finite StreamName window, dyadic tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal. The transport, continuation, provenance, and local naming rows preserve that same route and add no hidden coordinate.

Since quotient stream equality, selected host limits, hidden tail choice, ambient completeness, and detached Real handoffs are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1041.3 (Cofinal diagonal seal route exhaustion)**.** *Every accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *consumer route to a public* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed finite sequence*

$$\epsilon, T, D, L, R, W, Y, A, H, C, P, N.$$

No accepted route may replace the cofinal-tail budget by a diagonal selector alone, replace the diagonal budget by a tail threshold alone, or export a Real seal without the regular-source, finite-window, and dyadic ledger rows.

Proof. Start from the NameCert obligations of Definition 1041.2. They identify an accepted public route with the displayed rows of the carrier and give no second source of precision, tail, diagonal, regular-source, window, dyadic, or Real-seal data.

The first half of the route binds the two budget sides. The cofinal-tail row T prevents diagonal reads with incompatible thresholds from being classified as one seal, while the diagonal rows D, L prevent a tail threshold from being read as a terminal diagonal route by itself.

The second half binds the observable seal handoff. A consumer reaches A only after reading the regular-source row R, finite-window row W, and dyadic ledger Y, with H, C, P, N preserving and naming the route.

Therefore every public Real seal exported by the packet factors through the displayed sequence and has no bypass through quotient streams, selected limits, hidden tail choice, or ambient completeness. □

Theorem 1041.4 (Cofinal diagonal seal route admission)**.** *An accepted local* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet admits exactly the displayed route rows*

$$\epsilon, T, D, L, R, W, Y, A, H, C, P, N.$$

The admitted rows are the precision request, cofinal-tail budget, diagonal selector budget, diagonal-limit-facing budget, regular-source readback, finite-window exposure, dyadic tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation, package provenance, and local* $\mathsf{NameCert}$ *row of Definition 1041.1.*

Proof. Project the carrier of Definition 1041.1. Its tuple has exactly the displayed coordinates $\epsilon, T, D, L, R, W, Y, A, H, C, P, N$, and no further coordinate is available for the local packet.

Apply Definition 1041.2. The NameCert surface exposes precisely those carrier coordinates as the accepted rows, with N naming the same local packet rather than adding an extra route.

Thus local admission is coordinatewise admission of the displayed carrier. The absent coordinates named in the definition cannot be admitted by projection, so the route rows listed above are exact. □

Theorem 1041.5 (Cofinal diagonal seal observable route handoff)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal handoff accepted by a* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger Y before the terminal* $\mathsf{Real}^{\uparrow}$ *row A. The handoff excludes detached Real seals, quotient streams, selected host limits, hidden tail choice, and ambient completeness.*

Proof. Use the sibling anchors named in the chapter introduction. The regular-source chapter supplies the readback row R, the StreamName chapter supplies the finite window W, and the DyadicRatCore chapter supplies the tolerance ledger Y consumed by the terminal Real row A.

Apply Definition 1041.3. Any accepted consumer route to a public Real seal factors through the displayed sequence, so R, W, Y occur before A in the only admitted handoff.

The carrier has no detached Real coordinate, quotient-stream classifier, selected host limit, hidden tail selector, or ambient-completion row. Therefore none of those surfaces can serve as a public handoff bypass. □

Theorem 1041.6 (Cofinal diagonal seal nontrivial faithful field)**.** *The* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet has two distinguishable displayed carriers, and equality of all displayed fields is faithful to equality of the packet fields. On the Lean side, the same carrier surface is witnessed by the existing* `Nontrivial CofinalDiagonalSealUp` *and* `FieldFaithful CofinalDiagonalSealUp` *instances in* `BEDC.Derived.CofinalDiagonalSealUp.TasteGate`*.*

Proof. For distinguishability, vary a displayed carrier row while keeping the packet shape of Definition 1041.1. The NameCert obligations of Definition 1041.2 expose that row, so the two packets cannot collapse to one displayed carrier.

For faithfulness, compare two accepted packets field by field along $\epsilon, T, D, L, R, W, Y, A, H, C, P, N$. If every displayed row agrees, the carrier projections agree, the displayed route agrees, and the local NameCert row names the same packet.

The Lean taste-gate instances record exactly this pair of facts for the same displayed carrier family: nontriviality supplies distinct inhabitants, and field faithfulness says the visible carrier fields determine the packet fields. □

Theorem 1041.7 (Cofinal diagonal seal sibling scope)**.** *The* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet is scoped exactly by the sibling surfaces of chapter 698, chapter 853, chapter 896, chapter 301, chapter 126, chapter 291, and chapter 38. It consumes those rows in one finite NameCert packet and exports no independent Real completeness theorem or quotient-stream classifier.*

Proof. Project the carrier to the sibling rows T, D, L, R, W, Y, A. Each row is named by one of the cited sibling chapters and each is present before the local naming row N accepts the seal.

The route row C records their order of use, while H preserves componentwise transport and P records provenance over the same siblings. Thus the local packet is a scoped assembly of sibling rows rather than a new ambient completion principle.

The route-exhaustion theorem excludes every bypass to quotient streams, selected host limits, hidden tail choice, and ambient completeness. Hence the sibling scope is exactly the displayed cofinal-tail, diagonal-budget, diagonal-limit, RegSeqRat, StreamName, DyadicRatCore, and RealUp surface. □

Theorem 1041.8 (Cofinal diagonal seal carrier row readback)**.** *Every accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *carrier exposes exactly the twelve rows*

$$\epsilon, T, D, L, R, W, Y, A, H, C, P, N$$

of Definition 1041.1. It exposes no quotient-stream coordinate, selected host-limit coordinate, hidden-tail coordinate, or detached Real-seal coordinate.

Proof. Read the accepted packet through the carrier definition. The displayed carrier has precisely the rows $\epsilon, T, D, L, R, W, Y, A, H, C, P, N$, and every row is named before the local $\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}$ row accepts the packet.

Apply the NameCert obligations theorem. Its surface is coordinatewise over the same carrier rows: precision, cofinal tail, diagonal selector, diagonal limit, regular-source readback, finite window, dyadic ledger, terminal seal, transport, route replay, provenance, and local naming.

Use the route-exhaustion theorem to rule out further consumer coordinates. Since every public Real-facing route factors through the displayed sequence, no quotient-stream, selected host-limit, hidden-tail, or detached Real-seal coordinate can be read from an accepted carrier.

The sibling chapters chapter 698, chapter 853, chapter 896, chapter 301, chapter 126, chapter 291, and chapter 38 account for the public rows that the carrier does expose. □

Theorem 1041.9 (Cofinal diagonal seal route row exposure)**.** *In every accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *route, the cofinal-tail row* T*, diagonal budget row* D*, and diagonal-limit row* L *must be read before the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff rows* R, W, Y, A *become public.*

Proof. Start with the carrier definition. The carrier orders the budget side as T, D, L before the handoff side R, W, Y, A, and the local route row C records the same displayed continuation.

Apply the NameCert obligations theorem. It exposes cofinal-tail budget, diagonal selector budget, and diagonal-limit budget as named obligations before regular-source readback, finite-window exposure, dyadic tolerance exposure, and terminal Real seal exposure can be read as the public handoff.

The route-exhaustion theorem then prevents a consumer from bypassing T, D, L. Any accepted route to a public $\mathsf{Real}^{\uparrow}$ seal must factor through the displayed sequence, so the handoff rows cannot become public before the cofinal-tail, diagonal-budget, and diagonal-limit rows have been read.

Thus the sibling surfaces of chapter 698, chapter 853, and chapter 896 precede the handoff through chapter 301, chapter 126, chapter 291, and chapter 38. □

Theorem 1041.10 (Cofinal diagonal seal terminal readback handoff)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row* A *of an accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet is public only through the displayed* R, W, Y *handoff. It cannot be consumed as ambient Real completeness, as a quotient-stream classifier, or as a detached Real seal.*

Proof. Project the carrier of Definition 1041.1. The terminal row A appears only after the regular-source readback row R, the finite-window row W, and the dyadic tolerance row Y in the displayed packet.

Apply the NameCert obligations theorem to the same rows. The terminal Real seal exposure is not a separate coordinate; it is one obligation in the finite route that also exposes the regular source, the finite window, and the dyadic ledger.

Now use route exhaustion. Every accepted consumer route to a public $\mathsf{Real}^{\uparrow}$ seal factors through $\epsilon, T, D, L, R, W, Y, A, H, C, P, N$, so a consumer cannot read A as ambient completeness or as an off-route Real seal.

The cited RegSeqRat, StreamName, DyadicRatCore, and Real sibling chapters supply exactly the R, W, Y, A handoff rows. The cofinal-tail, diagonal-budget, and diagonal-limit siblings supply the preceding route rows, so the terminal readback remains a scoped handoff rather than a global completeness principle. □

Theorem 1041.11 (Cofinal diagonal seal cofinal budget compatibility)**.** *For an accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet, the cofinal-tail row* T*, diagonal selector row* D*, and diagonal-limit row* L *are jointly required before any* R, W, Y, A $\mathsf{Real}^{\uparrow}$*-seal handoff can classify as the same local packet. In particular, a route that omits one of* T, D, L *cannot be identified with the displayed local packet even if it presents the same regular-source, finite-window, dyadic, and terminal* $\mathsf{Real}^{\uparrow}$ *rows.*

Proof. Read the carrier of Definition 1041.1. The budget side of the packet is not a single row: T fixes the cofinal-tail threshold, D fixes the diagonal selector budget, and L records the diagonal-limit-facing budget.

Apply Definition 1041.9. It places T, D, L before the handoff rows R, W, Y, A, so the handoff cannot become public until the budget side has been read as part of the same local carrier.

Use Definition 1041.3 and Definition 1041.8. Every accepted consumer route factors through the displayed rows and every accepted carrier exposes exactly those rows, hence a route lacking one of T, D, L is not the same local packet as the displayed $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ seal. □

Theorem 1041.12 (Cofinal diagonal seal terminal exactness)**.** *For an accepted* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet, the terminal row* A *is public only through the displayed route*

$$T, D, L, R, W, Y, H, C, P, N.$$

It is never public as Real completeness, quotient-stream equality, a selected host limit, hidden tail choice, ambient completion, or a detached $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the carrier definition. The row A is a coordinate of the finite carrier, and the surrounding rows $T, D, L, R, W, Y, H, C, P, N$ record the budget side, observable handoff, transport, continuation, provenance, and local naming that make the row public.

Apply Definition 1041.10. It identifies the terminal $\mathsf{Real}^{\uparrow}$ row as public through the R, W, Y handoff and rejects ambient completeness, quotient-stream classifiers, and detached Real seals.

Now combine Definition 1041.3 with Definition 1041.9. Route exhaustion forces every public seal readback through the displayed finite sequence, while row exposure forces the cofinal-tail, diagonal-selector, and diagonal-limit rows to precede the handoff. Therefore A is terminal exactness for this finite route, not a Real-completeness theorem or quotient-stream equality principle. □

Theorem 1041.13 (Cofinal diagonal seal hidden selector refusal)**.** *No selected host limit, hidden tail choice, detached selector row, or off-route terminal seal can be consumed by a* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *packet unless it is one of the displayed finite rows*

$$\epsilon, T, D, L, R, W, Y, A, H, C, P, N.$$

Proof. Start from Definition 1041.1. The carrier lists exactly the finite rows $\epsilon, T, D, L, R, W, Y, A, H, C, P, N$ and explicitly has no coordinate for selected host limits, hidden tail choice, detached selector rows, quotient-stream equality, ambient completeness, or off-route Real handoffs.

Apply Definition 1041.8. Every accepted carrier exposes exactly the displayed rows and no hidden-tail, selected-limit, quotient-stream, or detached Real coordinate.

Finally use Definition 1041.3 and Definition 1041.12. Any consumed public seal must factor through the displayed route, and the terminal row is exact only for that route. Thus a hidden selector or host-limit witness is consumable only when it has already been identified with one of the displayed finite rows. □

The later Lean namespace `BEDC.Derived.CofinalDiagonalSealUp.TasteGate` must provide `Nontrivial CofinalDiagonalSealUp` and `FieldFaithful CofinalDiagonalSealUp` instances for the same displayed carrier.

Theorem 1041.14 (Cofinal diagonal seal scoped route closure)**.** *The carrier of Definition 1041.1 together with Definition 1041.2, Definition 1041.3, Definition 1041.4, Definition 1041.5, Definition 1041.7, Definition 1041.8, Definition 1041.9, Definition 1041.10, Definition 1041.11, Definition 1041.12, and Definition 1041.13 binds the exact scope of the finite cofinal-tail-to-diagonal* $\mathsf{Real}^{\uparrow}$ *seal route over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the cited L10 sibling rows.*

Within that scope, every admitted public seal readback factors through the displayed rows

$$\epsilon, T, D, L, R, W, Y, A, H, C, P, N.$$

Quotient stream equality, selected host limits, hidden tail choice, detached $\mathsf{Real}^{\uparrow}$ *seals, ambient completeness, public-object export, and bridge checking remain outside the scoped route.*

Proof. First read the local carrier and NameCert surface. The carrier definition gives the finite rows, while the NameCert obligations theorem exposes exactly precision, cofinal tail, diagonal selector, diagonal limit, regular-source readback, finite window, dyadic ledger, terminal seal, transport, continuation, provenance, and local naming.

Next assemble the admitted route. Route admission identifies the local rows, route exhaustion forces every public $\mathsf{Real}^{\uparrow}$ seal readback through the displayed sequence, and observable handoff places the RegSeqRat, StreamName, and DyadicRatCore rows before the terminal seal row.

Now bind the sibling scope and row exposure. The sibling-scope theorem fixes the cited L10 surfaces, carrier row readback rules out additional coordinates, and route row exposure places the cofinal-tail, diagonal-budget, and diagonal-limit rows before any terminal handoff.

Finally apply terminal readback handoff, cofinal budget compatibility, terminal exactness, and hidden selector refusal. These results state that the terminal row is exact only for the finite route and that hidden selectors, host limits, detached seals, quotient streams, ambient completeness, public-object export, and bridge checking are not produced by this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ packet is generated from a finite precision request, cofinal-tail budget row, Cauchy diagonal budget row, diagonal limit budget row, RegSeqRat readback row, StreamName window row, DyadicRatCore tolerance ledger, terminal RealUp seal row, componentwise transports, displayed continuation, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1041.14 is the scoped closure record for the finite cofinal-tail-to-diagonal RealUp seal route, its sibling scope, its budget-compatible handoff, terminal exactness, and hidden-selector refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalDiagonalSeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CofinalDiagonalSealUp.CofinalDiagonalSealTasteGate\_single\_carrier\_alignment`
Lean4 ✓ `BEDC.Derived.CofinalDiagonalSealUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, hidden tail choice, ambient completeness, public-object export, or bridge checking.

Upgrade path: Public closure requires a public-object surface beyond the finite cofinal-tail-to-diagonal route seal.

Chapter 1042

A Concrete Naming Certificate for $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$

The $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ packet realizes observer-history identity as a finite selected-row budget feeding the StreamName and Real-completion observation routes. It consumes the selected and omitted rows of chapter 514, the observer-history source of chapter 477, the finite StreamName window route of chapter 126, and the L10 budget selector of chapter 782. Its role is to bind those routes into one local NameCert packet, not to create a quotient stream, host real equality, selected host limit, or hidden observation selector.

Falsifiable prediction: A packet with a finite observation budget but no selected ObserverFilter row is rejected by the local classifier.

Independence witness: observerfilter, finiteobservationbudgetselector, observerhistoryidentity, streamname, real: no sibling is bijective because none carries selected and omitted observer rows together with a finite L10 observation budget and Real-seal handoff.

Definition 1042.1 (Observer-filter budget-selector carrier)**.** *An* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is an accepted* $\mathsf{ObserverFilter}^{\uparrow}$ *packet with selected subhistory row S and omitted-row ledger O;*
2. *I is the* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *source row factored by the observer filter;*
3. *B is an accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *row whose selected schedule reads only from S;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected from that schedule;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger read from the finite windows in W;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the dyadic ledger;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the displayed budget route;*
8. *H records componentwise* hsame *transports for the observer-filter, identity, selected row, omitted ledger, budget, window, dyadic, handoff, seal, provenance, and local name rows;*

9. *C records the displayed* Cont *routes from I to F, from S to B, from B to W, from W to D, from D to R, and from R to E;*
10. *P is the* Pkg *provenance over* ObserverHistoryIdentity$^{\uparrow}$, ObserverFilter$^{\uparrow}$, StreamName$^{\uparrow}$, FiniteObservationBudgetSe DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverFilterBudgetSelector}^{\uparrow}}$ *row.*

The classifier accepts G only when the budget row B reads from the selected row S of F and the omitted ledger O remains a boundary row. The carrier has no coordinate for a finite budget without an observer-filter selection, an omitted-row observation window, quotient streams, host real equality, countable choice, hidden limit selectors, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorTasteGate single carrier alignment`

Theorem 1042.2 (Observer-filter budget-selector NameCert surface)**.** *For the carrier of Definition 1042.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverFilterBudgetSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$, observer-history factorization through I and F, selected-row budget admission through S and B, StreamName window exposure through W, dyadic ledger exposure through D,* RegSeqRat$^{\uparrow}$ *handoff through R,* Real$^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape for omitted rows in O.*

Proof. Project the carrier. The displayed packet contains the observer-history row, observer filter, selected and omitted rows, budget selector, finite windows, dyadic ledger, regular handoff, Real seal, and the four structural rows H, C, P, N.

Read the positive route in order. The observer-history row I is first consumed by the observer filter F; the selected subhistory S then supplies the only schedule source for B; the budget selector exposes the finite windows W before the dyadic ledger D, handoff row R, and seal row E are available.

The omitted ledger is checked by coordinate exhaustion. It appears as O and nowhere as B, W, D, R, or E. Since the carrier also omits quotient streams, host real equality, countable choice, hidden selectors, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorNameCertSurface omitted field empty reflects`

Theorem 1042.3 (Observer-filter budget-selector StreamName handoff)**.** *Every accepted* ObserverFilterBudgetSele *packet routes its finite observation budget through the selected ObserverFilter row before any* StreamName$^{\uparrow}$ *window is available:*

$$I \to F \to S \to B \to W \to D \to R \to E.$$

The omitted ledger O may be read only as boundary data through H, C, P, N; it cannot supply StreamName windows, dyadic observations, RegSeqRat$^{\uparrow}$ *handoff rows,* Real$^{\uparrow}$ *seals, quotient streams, host real equality rows, or hidden limit selectors.*

Proof. Start with the NameCert surface theorem. It identifies the accepted packet with $G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$ and places the budget selector after the selected row S.

Apply the ObserverFilter L10 handoff of Definition 514.11. That handoff forces every budget-selector route whose schedule reads from S through the chain from observer-history identity to observer filter, StreamName, and the finite observation-budget selector.

The remaining coordinates are the budget-selector continuation. The route from B to W, D, R, E is the finite observation route, while H, C, P, N only transport, replay, package, and name the same packet.

Thus the public handoff is exactly the displayed finite route. Because O is not a selected row or budget coordinate, no omitted row can become any of the named observation, completion, equality, or selector objects. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorStreamNameHandoff omitted boundary not erased`

Theorem 1042.4 (Observer-filter budget-selector omitted-row exclusion)**.** *For every accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet*

$$G = (F, I, S, O, B, W, D, R, E, H, C, P, N),$$

any consumer that reaches one of B, W, D, R, E factors through the selected row S. The omitted ledger O may be read only through the boundary rows H, C, P, N and cannot be repacked as an observation window, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream, host equality row, hidden limit selector, or ambient completeness row.*

Proof. Project the carrier definition for G. The positive observation coordinates are I, F, S, B, W, D, R, E, while the omitted ledger occurs only as the displayed row O together with the structural rows H, C, P, N.

Apply Definition 1042.2. It identifies O as boundary-only data and places carrier admission, continuation replay, provenance, and local naming in the rows H, C, P, N rather than in the observation route.

Apply Definition 1042.3. The handoff forces every consumer reaching the budget, window, dyadic, regular-sequence, or Real-seal coordinates through

$$I \to F \to S \to B \to W \to D \to R \to E.$$

It remains to exhaust the packet coordinates. There is no carrier slot in which O can appear as B, W, D, R, E, a quotient stream, host equality row, hidden limit selector, or ambient completeness row, so the omitted ledger cannot be repacked as any such consumer-facing object. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorOmittedRowExclusion omitted boundary not repacked`

Theorem 1042.5 (Observer-filter budget selector selected-row gate)**.** *Every accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet*

$$G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$$

must expose the selected $\mathsf{ObserverFilter}^{\uparrow}$ *row S before any finite budget row B,* $\mathsf{StreamName}^{\uparrow}$ *window family W, dyadic ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, or* $\mathsf{Real}^{\uparrow}$ *seal row E can be consumed. The omitted rows O remain ledger data through H, C, P, N and cannot feed the selected schedule, quotient streams, host real equality, or hidden observation selectors.*

Proof. Project the carrier G. The positive route records I, F, S, B, W, D, R, E in that order, while O is a separate omitted-row ledger and H, C, P, N are the transport, continuation, provenance, and local naming rows.

Apply the local NameCert surface. It admits the budget row only as the row whose selected schedule reads from S, and it places the StreamName, dyadic, regular-sequence, and Real-seal rows after that budget row.

Now use omitted-row exclusion from Definition 1042.4. Since every consumer of B, W, D, R, E factors through the selected row and since O cannot be repacked as one of those consumer-facing coordinates, the selected-row gate is the only entry to the finite observation route.

The remaining excluded objects follow by coordinate exhaustion. The carrier has no row for quotient streams, host real equality, or hidden observation selectors, so none of them can bypass S or make the omitted ledger into a schedule source. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorSelectedRowGate selected field not erased`

Theorem 1042.6 (Observer-filter budget-selector budget refinement nonescape)**.** *Let*

$$G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet. Any finite-budget refinement that replaces B by a row $B^{\sharp}$ while preserving the selected* $\mathsf{ObserverFilter}^{\uparrow}$ *row S and the* $\mathsf{StreamName}^{\uparrow}$ *window family W preserves the downstream route*

$$I \to F \to S \to B^{\sharp} \to W \to D \to R \to E.$$

Such a refinement cannot turn omitted ledger data O into a window, dyadic ledger, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream, host real equality row, hidden limit selector, or ambient completeness row.*

Proof. Project the carrier definition for G. Its positive observation coordinates are the displayed rows I, F, S, B, W, D, R, E, while O appears as an omitted ledger and H, C, P, N provide boundary transport, continuation, provenance, and local naming.

Apply Definition 1042.5. The gate forces any consumer of the finite budget row, window family, dyadic ledger, regular-sequence handoff, or Real seal to enter through S; replacing B by a refinement that preserves S and W therefore leaves the consumer route through the selected row.

Apply Definition 1042.4. The omitted ledger remains boundary-only data, so the refinement has no source from which to manufacture a consumer-facing row out of O.

It remains to exhaust the coordinates before dyadic readback consumes D. The packet has no slot in which O can become W, D, R, E, a quotient stream, a host real equality row, a hidden limit selector, or an ambient completeness row; hence the refined budget still feeds exactly the displayed route. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorBudgetRefinementNonesca budget field not erased`

Theorem 1042.7 (Observer-filter budget-selector dyadic readback exactness)**.** *Every accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D only after the selected* $\mathsf{ObserverFilter}^{\uparrow}$ *row S, finite budget row B, and* $\mathsf{StreamName}^{\uparrow}$ *window family W have been exposed in the displayed route*

$$I \to F \to S \to B \to W \to D.$$

The readback cannot replace D by omitted-row data from O, quotient streams, host real equality, or hidden limit selectors.

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorDyadicReadbackExactness`
`dyadic field not erased`

Proof. Start with the accepted carrier $G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$. The carrier definition places D after W, and the StreamName handoff theorem places W after the selected-row budget route through S and B.

Apply Definition 1042.2. It identifies the $\mathsf{DyadicRatCore}^{\uparrow}$ exposure as the named row D and keeps the omitted ledger O outside the positive observation route.

Finally use coordinate exhaustion. The carrier has no coordinate for quotient streams, host real equality, or hidden limit selectors, so the only dyadic readback available to the packet is the displayed row reached through I, F, S, B, W. □

Theorem 1042.8 (Observer-filter budget-selector Real-seal terminality)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet factors through*

$$I \to F \to S \to B \to W \to D \to R \to E,$$

with the omitted row O serving only as a boundary ledger. No terminal read may bypass the selected observer-filter budget route or use O as a Real-seal source.

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorRealSealTerminality`
`terminal field not erased`

Proof. The StreamName handoff theorem gives the positive route from I through F, S, B, W, D, R, E. This already places the terminal Real seal after the finite budget windows and the dyadic and regular-sequence rows.

The dyadic readback exactness theorem pins the D row to the selected observer-filter budget route. Therefore the following R and E reads inherit the same selected-row dependency rather than a separate omitted-row or host-real source.

The omitted ledger remains boundary data because it is listed only as O and transported by H, C, P, N. Since the carrier excludes quotient streams, host real equality, selected host limits, and hidden limit selectors, every terminal Real-seal read factors through the displayed route. □

Theorem 1042.9 (Observer-filter budget-selector bridge-ready route)**.** *Every accepted* ObserverFilterBudgetSelecto *packet exposes the exact bridge-ready selected-row chain*

$$I \to F \to S \to B \to W \to D \to R \to E.$$

The rows O, H, C, P, N are retained only as omitted-row boundary data, componentwise transport, displayed route replay, provenance, and local naming. In particular, the L10 budget route from chapter 477, chapter 514, chapter 782, chapter 126, and the dyadic-to-Real chapters is presented as one public route rather than as separable packet facts.

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorBridgeReadyRoute`
`support fields faithful`

Proof. Begin with Definition 1042.2. It identifies the accepted packet as $G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$ and fixes the NameCert surface over the selected-row budget carrier.

Next use Definition 1042.3. This supplies the L10 route through I, F, S, B, W and keeps O outside the StreamName window source.

The dyadic and terminal components are then forced by Definition 1042.7 and Definition 1042.8. The first pins D to the selected-row finite-window readback, and the second makes E terminal only after R.

It remains to account for the remaining coordinates. The carrier definition assigns H to hsame transport, C to Cont replay, P to Pkg provenance, and N to the local NameCert row, so none of them can become an additional observation or completion source. The displayed chain is therefore the bridge-ready route exported by the packet. □

Theorem 1042.10 (Observer-filter budget-selector completion-consumer factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet factors through*

$$I \to F \to S \to B \to W \to D \to R \to E$$

together with the structural rows H, C, P, N*. The omitted ledger* O *remains boundary-only: it cannot feed* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seals, quotient streams, host real equality, hidden selectors, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorCompletionConsumerFacto`

Proof. Start from the local NameCert surface of Definition 1042.2. It exposes the packet as $G = (F, I, S, O, B, W, D, R, E, H, C, P, N)$ and fixes H, C, P, N as transport, displayed replay, provenance, and local naming rows for the same carrier.

The selected-row gate and omitted-row exclusion give the source side of the factorization. Definition 1042.5 forces every consumer of B, W, D, R, E through S, while Definition 1042.4 keeps O from becoming a window, dyadic ledger, regular handoff, Real seal, quotient stream, host equality row, hidden selector, or ambient completeness row.

Next use dyadic readback exactness. Definition 1042.7 pins D to the finite-window route through I, F, S, B, W, so the downstream $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R cannot be read from a separate omitted ledger.

Finally apply Real-seal terminality and the bridge-ready route. Definition 1042.8 makes E terminal only after R, and Definition 1042.9 packages the whole route as the completion-facing chain $I \to F \to S \to B \to W \to D \to R \to E$ with H, C, P, N as the only structural support rows. □

Theorem 1042.11 (Observer-filter budget-selector bridge boundary locality)**.** *The bridge-ready route of Definition 1042.9 is local to* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*. It exports no quotient stream, host real equality, countable choice selector, hidden limit selector, or ambient completeness row.*

Proof. Project the carrier coordinates. The positive rows are exactly I, F, S, B, W, D, R, E, and each one belongs to one of the named source chapters or kernel rows.

Now project the structural coordinates. The rows H, C, P, N provide only transport, route replay, provenance, and local naming for the same packet. The omitted ledger O is held as boundary data and is not a source for B, W, D, R, or E.

Coordinate exhaustion gives the exclusion claim. The carrier definition contains no coordinate for quotient streams, host real equality, countable choice selectors, hidden limit selectors, or ambient completeness; hence the public bridge boundary cannot export any such row. □

Theorem 1042.12 (Observer-filter budget-selector Real-completion consumer)**.** *Any Real and Completion consumer reading an accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet must pass through the selected observer row* S*, finite budget row* B*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *before the read can be used by a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. The structural rows* O, H, C, P, N *remain omitted-row boundary data, componentwise transport, displayed replay, provenance, and local naming for the same route.*

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorCarrier real completion consumer`

Proof. Begin with the bridge-ready route of Definition 1042.9. It exposes the selected-row chain

$$I \to F \to S \to B \to W \to D \to R \to E$$

as the public route available to a Real and Completion consumer.

The terminality theorem Definition 1042.8 makes the final $\mathsf{Real}^{\uparrow}$ read depend on the preceding selected row, budget, window, dyadic ledger, and regular-sequence handoff. Thus a downstream $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ consumer cannot start at E alone.

The sibling L10 handoff rows already cited in the chapter provide the source side of the same route: observer-history identity feeds the selected observer row, the finite observation-budget selector supplies B, and the StreamName chapter supplies W. Since the carrier has no coordinate for quotient streams, host real equality, selected host limits, hidden limit selectors, or ambient completeness, the listed chain is the only Real-completion consumer route exported by the packet. □

Theorem 1042.13 (Observer-filter budget-selector downstream budget lock)**.** *Any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *or* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet must preserve the selected observer row* S*, finite budget row* B*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, dyadic row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *as one downstream route. The omitted ledger* O *remains boundary data, and* H, C, P, N *remain the structural rows for componentwise transport, displayed replay, provenance, and local naming. The downstream consumer cannot replace this route by an omitted-row read, quotient stream, host real equality, selected host limit, hidden selector, or ambient completeness claim.*

Proof. Start with the selected-row gate and omitted-row exclusion. Definition 1042.5 forces every read of B, W, D, R, E through S, while Definition 1042.4 keeps O outside the selected observer route.

Next attach the dyadic and Real-facing rows. Definition 1042.7 fixes D as the dyadic readback of the same finite window route, and Definition 1042.8 makes E terminal only after the selected row, budget row, window row, dyadic row, and regular handoff row have been retained.

Finally compose the bridge-ready route with the Real-completion consumer factorization. Definition 1042.9 packages the route through S, B, W, D, R, E, and Definition 1042.12 forces $\mathsf{CauchyLimitSeal}^{\uparrow}$ and $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ consumers to read that same chain. The structural rows remain H, C, P, N, and the omitted row remains boundary-only, so no downstream completion consumer can escape the displayed lock. □

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.ObserverFilterBudgetSelectorDownstreamBudgetLock`

The Lean file `Derived/ObserverFilterBudgetSelectorUp/TasteGate.lean` will provide Nontrivial ObserverFi and FieldFaithful $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ instances over the displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverFilterBudgetSelector}^{\uparrow}}$.

Constructive story: An ObserverFilterBudgetSelector$^{\uparrow}$ packet is generated from observer-history identity, observer-filter selected and omitted rows, a finite observation-budget selector, StreamName windows, DyadicRatCore observations, RegSeqRat handoff, Real seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.ObserverFilterBudgetSelectorUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{ObserverFilterBudgetSelector}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1042.1, Definition 1042.2, Definition 1042.3, Definition 1042.7, and the public export Definition 1042.8 bind the finite selected-row budget route, dyadic readback exactness, and terminal Real-seal factorization over ObserverHistoryIdentityUp, ObserverFilterUp, StreamNameUp, FiniteObservationBudgetSelectorUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ObserverFilterBudgetSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host real equality, selected host limits, countable choice, hidden limit selectors, ambient completeness, Lean verification, or bridge checking.

Upgrade path: No further paper-side closure grade remains for the selected-row Real-seal route; verification and bridge checking remain separate axes.

Chapter 1043

A Concrete Naming Certificate for $\mathsf{ObserverTimeStep}^{\uparrow}$

The $\mathsf{ObserverTimeStep}^{\uparrow}$ packet realizes the Hist time-step direction of chapter 9418 as one finite local naming certificate. It sits between the elementary observer-history packet of chapter 479, the local stream packet of chapter 603, the clock-budget packet of chapter 665, the licensed comparison packet of chapter 1031, and the bridge ledger of chapter 1032. The packet names the single Hist-growth step, not a global clock, a synchronized observer frame, a quotient history identity, or an external time parameter.

Falsifiable prediction: Removing the displayed Cont-step row prevents a consumer from preserving the local ObserverTimeStepUp NameCert route.

Independence witness: Nearest siblings: observerhistidentity, histtimestream, localclockbudget, observerlicensedcomparison, observerbridgeledger. None carries exactly the elementary Hist step plus no-global-clock refusal packet: observerhistidentity names observer-history identity without the time-step row, histtimestream records a stream rather than one atomic step, localclockbudget packages a bounded clock window, observerlicensedcomparison compares licensed observer histories, and observerbridgeledger joins bridge rows without isolating the elementary Hist-growth step.

Definition 1043.1 (Observer time-step carrier)**.** *An* $\mathsf{ObserverTimeStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H, D, C, R, L, P, N)$$

with the following displayed rows:

1. *H is the source* Hist *row before the elementary growth step;*
2. *D is the distinction tag recording whether the step applies* E1 *or* E0*;*
3. *C is the displayed* Cont *step from H along D to the next local* Hist *row;*
4. *R records componentwise* hsame *transport for the source, distinction, continuation, refusal, provenance, and naming rows;*
5. *L is the no-global-clock ledger refusing any synchronized external time coordinate for the step;*
6. *P is the* Pkg *provenance over* Hist*, distinction tags,* Cont*,* hsame*, no-global-clock refusal, and* NameCert *rows;*

7. N is the local $\mathsf{NameCert}_{\mathsf{ObserverTimeStep}^{\uparrow}}$ *row.*
The carrier accepts a local observer time step only as the displayed Hist source, distinction tag, Cont step, transport row, no-global-clock ledger, package provenance, and local naming row together.

Lean4 (definition) `BEDC.Derived.ObserverTimeStepUp.ObserverTimeStepUp`

Theorem 1043.2 (Observer time-step NameCert obligations)**.** *For the carrier of Definition 1043.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverTimeStep}^{\uparrow}}$ *surface consists exactly of carrier admission for* $T = (H, D, C, R, L, P, N)$*, source-Hist exposure through* H*, distinction exposure through* D*, displayed Cont-step replay through* C*, componentwise* hsame *transport through* R*, no-global-clock refusal through* L*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier to the seven displayed rows. The source Hist row, distinction tag, and Cont-step row are visible before a consumer can name the time step.

Transport is then restricted to R, which preserves the same source, tag, continuation, refusal, provenance, and local name rows. The no-global-clock ledger L is read as part of the packet, so the local step is never promoted to a synchronized external clock.

The provenance row P and naming row N bind the admitted read to exactly these coordinates. Since no other coordinate appears in the carrier, the listed rows exhaust the NameCert obligations for $\mathsf{ObserverTimeStep}^{\uparrow}$. □

Theorem 1043.3 (Observer time-step nonescape)**.** *Every accepted* $\mathsf{ObserverTimeStep}^{\uparrow}$ *consumer factors through the finite route*

$$H, D, C, R, L, P, N.$$

No accepted consumer may replace the displayed Cont step by a global clock tick, a synchronized observer frame, a quotient history identity, an unrecorded Hist predecessor, or an external time parameter.

Proof. Start from the NameCert obligations of Definition 1043.2. They identify an accepted time-step read with the source Hist row, distinction tag, displayed Cont step, transport row, refusal ledger, provenance, and local name.

The Cont row C is the only route from the source row H and distinction tag D to the next local Hist row. Removing C leaves no displayed replay row for a consumer to preserve.

The refusal ledger L excludes the external clock and synchronized-frame alternatives, while R, P, N only transport, package, and name the same displayed rows. Hence every accepted consumer remains inside the finite route H, D, C, R, L, P, N. □

The Lean namespace `BEDC.Derived.ObserverTimeStepUp.TasteGate` provides explicit Nontrivial ObserverTim and FieldFaithful $\mathsf{ObserverTimeStep}^{\uparrow}$ instances over the displayed source-Hist, distinction-tag, Cont-step, hsame-transport, no-global-clock, Pkg, and NameCert fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverTimeStep}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObserverTimeStep}^{\uparrow}$ packet is generated from a source Hist row, distinction tag, displayed Cont step, hsame transport, no-global-clock refusal ledger, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObserverTimeStep}^{\uparrow}}$, seed)

Scope closed: Definition 1043.1, Definition 1043.2, and Definition 1043.3 specify the finite local Hist-growth step and its refusal of global-clock escape.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObserverTimeStep}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global clocks, synchronized observer frames, quotient history identity, external time parameters, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-Hist exposure, distinction exposure, Cont-step replay, hsame transport, no-global-clock refusal, nontriviality, and field faithfulness for the same packet.

Chapter 1044

A Concrete Naming Certificate for $\mathsf{SocketUseLedger}^{\uparrow}$

The $\mathsf{SocketUseLedger}^{\uparrow}$ packet records visible consumption of an apophatic socket as a finite NameCert event surface. It sits under the name-to-gate discipline of chapter 9400 and consumes the sibling surfaces of chapter 543, chapter 536, chapter 584, chapter 629, and chapter 715. Those chapters name the socket, attach the gate, refuse hidden supply, and ask the audit question; this chapter records the finite event by which a consumer visibly uses that socket. The visible-use row consumes the $\mathsf{ApophaticSocket}^{\uparrow}$ use-site row and $\mathsf{ExternalityGate}^{\uparrow}$ gate row as sibling carriers, not merely as prose context.

Definition 1044.1 (SocketUseLedger carrier)**.** *A* $\mathsf{SocketUseLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, S, Q, V, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the external supply position;*
2. *S is the* $\mathsf{ApophaticSocket}^{\uparrow}$ *row naming the requested socket and its visible use site;*
3. *Q is the* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *row asking whether the socket use remained marked;*
4. *V is the visible-use row recording the accepted consumer read through the socket;*
5. *R is the refusal ledger for unmarked, off-gate, or internalizing uses of the same requested supply;*
6. *H records componentwise* hsame *transports for A, S, Q, V, R together with continuation, provenance, and naming rows;*
7. *C records the displayed* Cont *routes by which audit consumers read the name, socket, question, visible-use, and refusal rows;*
8. *P is the* Pkg *provenance over* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{NegativeNameBoundary}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SocketUseLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for the requested outside supply itself, no anonymous host import, no hidden theorem premise, and no route by which a visible socket use becomes a positive internal description of the external source.

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.SocketUseLedgerTasteGate single carrier alignment`

Theorem 1044.2 (SocketUseLedger NameCert obligations)**.** *For the carrier of Definition 1044.1, the local* $\mathsf{NameCert}_{\mathsf{SocketUseLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$U = (A, S, Q, V, R, H, C, P, N),$$

apophatic-name exposure through A*, socket exposure through* S*, gate-question exactness through* Q*, visible-use readback through* V*, refusal-ledger coverage through* R*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*, dependency provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the displayed packet. The tuple U lists the apophatic name, socket, gate question, visible-use row, refusal ledger, transport, continuation, provenance, and local naming rows.

The name-to-gate portion is read through A, S, Q. The apophatic name gives the external supply position, the socket row gives the use site, and the gate question records the public test for marked use.

The event portion is read through V and R. A visible use is accepted only when it passes through the named socket and gate question; an unmarked or internalizing request is sent to the refusal ledger.

Finally H, C, P, N provide the only transport, replay, provenance, and naming routes. Since the carrier contains no requested outside supply, the local certificate records use of the socket without internalizing the supply. □

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.SocketUseLedgerCarrier namecert obligations`

Theorem 1044.3 (SocketUseLedger visible-use non-escape)**.** *Every accepted socket-use read of a* $\mathsf{SocketUseLedger}^{\uparrow}$ *packet factors through* $\mathsf{ApophaticName}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*,* $\mathsf{ExternalityGate}^{\uparrow}$*, and the local visible/refusal ledger before any audit consumer sees it. The exported read remains inside* $A, S, Q, V, R, H, C, P, N$ *and cannot export the requested supply, an unmarked selector, a hidden axiom channel, a quotient collapse, proposition-extensionality collapse, or a host-level replacement for the gate answer.*

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.SocketUseLedgerCarrier visible refusal layer separation`

Proof. Start with the carrier of Definition 1044.1. The accepted packet exposes the name row A, socket row S, gate-question row Q, visible-use row V, refusal row R, and structural rows H, C, P, N.

Apply the NameCert obligations of Definition 1044.2. A consumer read reaches V only after the name, socket, and gate question have been displayed, and any off-gate request is routed to R.

Now read the audit-facing rows. The continuation row C exposes the finite use/refusal result, while H, P, N transport, package, and name the same packet. No excluded supply or host principle appears among the coordinates, so the accepted socket-use read cannot escape the ledger. □

Theorem 1044.4 (SocketUseLedger public visible refusal)**.** *Every public visible-use or refusal read of a* $\mathsf{SocketUseLedger}^{\uparrow}$ *packet factors through* $\mathsf{ApophaticName}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*,* $\mathsf{ExternalityGate}^{\uparrow}$*,* $\mathsf{NegativeNameBoundary}^{\uparrow}$*, and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *before the local visible-use row, refusal ledger, componentwise transport, continuation replay, provenance, and local naming rows are read. Thus the exported event is a sibling-carrier route followed by the local ledger route*

$$A, S, Q, V, R, H, C, P, N.$$

It cannot be read as a detached visible-use event, an unmarked supply import, a gate-free refusal, a quotient collapse, or a host replacement for the apophatic boundary.

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.SocketUseLedgerCarrier public visible refusal`

Proof. Start with the carrier of Definition 1044.1. The name row A, socket row S, gate question Q, visible-use row V, refusal row R, and structural rows H, C, P, N are the only displayed fields.

Apply Definition 1044.2. It forces apophatic-name exposure, socket exposure, gate-question exactness, visible-use readback, refusal-ledger coverage, transport, replay, provenance, and local naming to be read as one NameCert surface.

Now use Definition 1044.3. The visible-use read reaches the audit consumer only after the sibling apophatic name, socket, externality gate, negative boundary, and gate-question carriers have constrained it. The local ledger rows can therefore export visible use or refusal only through that route, with no detached supply import or host-level gate replacement. □

Theorem 1044.5 (SocketUseLedger public event surface)**.** *The public exported object of* $\mathsf{SocketUseLedger}^{\uparrow}$ *is exactly the visible-use and refusal event surface*

$$U = (A, S, Q, V, R, H, C, P, N).$$

Its public rows expose the apophatic name A, socket S, gate question Q, visible-use row V, refusal row R, componentwise transport row H, continuation row C, provenance row P, and local naming row N, and no other public coordinate is exported.

Proof. The carrier of Definition 1044.1 lists precisely the nine displayed rows of U. The NameCert obligations of Definition 1044.2 certify carrier admission, exposure, visible-use readback, refusal coverage, transport, continuation replay, provenance, and local naming for those rows.

The visible-use non-escape theorem Definition 1044.3 excludes the requested outside supply, unmarked selectors, hidden axiom channels, quotient collapse, proposition-extensionality collapse, and host-level gate replacement from the exported read. Hence the public surface is the event ledger U and nothing beyond it. □

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.SocketUseLedgerCarrier public event surface`

Falsifiable prediction: Every accepted socket-use read factors through ApophaticNameUp, ApophaticSocketUp, ExternalityGateUp, and the local visible/refusal ledger before any audit consumer sees it. A read that reaches audit without $A, S, Q, V, R, H, C, P, N$ falsifies this packet.

Independence witness: Nearest siblings: apophaticname, apophaticsocket, externalitygate, negativenameboundary, and apophaticgatequestion. SocketUseLedgerUp is not bijective to any of them because it simultaneously carries an event-level visible-use row and a refusal row for the same socket-use read; each sibling lacks at least one of those event ledger coordinates.

Theorem 1044.6 (SocketUseLedger mature audit package)**.** *The* $\mathsf{SocketUseLedger}^{\uparrow}$ *mature audit package consists of the visible-use and refusal event surface $U = (A, S, Q, V, R, H, C, P, N)$, the apophatic sibling carriers* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{NegativeNameBoundary}^{\uparrow}$, *and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$, *the falsifiable prediction above, the independence witness above, and the* $\mathsf{TasteGate}$ *target recorded by the local closure rows. Together these data bind the public socket-use event, its refusal boundary, its sibling-carrier route, and its audit-facing taste gate without exporting the requested outside supply.*

Proof. The public event surface is fixed by Definition 1044.5: the only exported rows are $A, S, Q, V, R, H, C, P, N$. The visible-use non-escape theorem Definition 1044.3 keeps every accepted read inside the apophatic name, socket, gate question, visible-use row, refusal ledger, and structural rows.

The sibling-carrier route is supplied by the chapters named in the introduction and by Definition 1044.4. The falsifiable prediction identifies an observable failure mode for any audit read that skips this route, while the independence witness separates the packet from its nearest apophatic siblings by the simultaneous presence of visible-use and refusal event rows.

Finally the closure rows record the local TasteGate target for the same packet. These ingredients are exactly the mature paper-side inventory: public surface, sibling route, falsifiable boundary, independence separation, and taste-gate target, with no coordinate for the requested outside supply. □

Closure status for $\mathsf{NameCert}_{\mathsf{SocketUseLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SocketUseLedger}^{\uparrow}$ packet is generated from an apophatic-name row, socket row, gate-question row, visible-use row, refusal ledger, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SocketUseLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1044.1, Definition 1044.2, Definition 1044.3, and Definition 1044.5 bind the bridge route from $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{NegativeNameBoundary}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, and the local visible-use/refusal ledger into the public socket-use event surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SocketUseLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Origin: ai

Lean4 ✓ `BEDC.Derived.SocketUseLedgerUp.taste gate`

Not claimed: This chapter does not close Lean verification, bridge checking, existence of the requested outside supply, global audit completeness, or every possible socket-use grammar.

Upgrade path: no further paper-side theory upgrade – chapter is mature on the closure axis pending formal-axis work.

Chapter 1045

A Concrete Naming Certificate for $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$

The $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ packet realizes the finite Rule 110 glider classifier named by the cellular automaton reading of Definition 9390.3. It sits beside the finite rule and orbit surface of chapter 934, the carrier-facing substrate packet of chapter 942, and the compile-frontier witness of chapter 952. Cook 2004 supplies the Rule 110 universality route, while Martinez 2007 supplies the named finite glider catalog; this packet records only the row-window classifier data needed to inspect that catalog inside BEDC.

Falsifiable prediction: A $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ packet without a displayed Rule 110 row window, catalog tag, phase-period-displacement row, and local match row is not an accepted glider-catalog carrier.

Independence witness: cellularautomaton, cellularsubstrate, cookcompilefrontierwitness, cellularpatterncatalog: none is bijective to $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ because this packet carries the Rule 110 row window together with a Martinez-style glider tag, phase-period-displacement row, displayed local match, hsame transport, continuation route, provenance row, and local NameCert. The siblings expose cellular dynamics, substrate rows, compile-frontier coordinates, or a general pattern catalog, but not this finite Rule 110 glider classifier packet.

Definition 1045.1 (Rule 110 glider catalog carrier)**.** *A* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (R, W, G, P, D, H, C, K, N)$$

with the following displayed rows:

1. *R is the Rule 110 rule row and finite row-window source being inspected.*
2. *W is the displayed finite row window, including the center position, neighbouring cells, and enough surrounding cells to test the named phase string.*
3. *G is the finite catalog-tag row, whose tag is one of ether, refused window, or a Martinez 2007 glider name such as A, B, $\bar{B}$, C_1, C_2, E, $\bar{E}$, F, G, or H.*
4. *P is the phase, period, and displacement row attached to the tag, recording the catalog phase string, fundamental period, and row displacement per period.*
5. *D is the displayed local-match row, recording that the row window W matches the phase row P under the catalog's neighbour discipline, or that it is refused by that finite discipline.*

6. *H records componentwise* hsame *transport for R, W, G, P, D, K, N.*
7. *C records the displayed* Cont *routes from the Rule 110 row to the finite window, from the window to the phase row, from the phase row to the local-match row, and from the local-match row to the catalog tag.*
8. *K is the* Pkg *provenance over* $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{CookCompileFrontierWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $R, W, G, P, D, H, C, K, N$. It has no coordinate for an unbounded Rule 110 orbit, a host visual recognizer, a complete all-time glider theorem, a hidden catalog row, or a Cook universality proof.

Lean4 ✓ `BEDC.Derived.RuleOneTenGliderCatalogUp.TasteGate.RuleOneTenGliderCatalogTasteGate single carrier alignment`

Theorem 1045.2 (Rule 110 glider catalog NameCert obligations)**.** *For the carrier of Definition 1045.1, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}$ *surface consists exactly of carrier admission for $J = (R, W, G, P, D, H, C, K, N)$, Rule 110 row-window exposure through R, W, catalog-tag exposure through G, phase-period-displacement exposure through P, displayed local-match exposure through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through K, and local naming through N. Every accepted tag read factors through the finite route*

$$R \longmapsto W \longmapsto P \longmapsto D \longmapsto G,$$

with H, C, K, N attached as transport, route, provenance, and naming rows.

Proof. Carrier admission is projection from Definition 1045.1. The displayed tuple lists the Rule 110 row, finite window, catalog tag, phase-period-displacement row, local match, transport, route, provenance, and local naming rows, and no other row is part of the packet.

The catalog read is finite. A consumer first exposes the Rule 110 row R and finite window W, then reads the phase-period-displacement row P, then checks the displayed local match D, and only then obtains the tag row G.

The structural rows do not add new classifier data. The row H transports the displayed entries componentwise, C replays the same finite route, K records provenance through the cellular and compile-frontier siblings, and N names the local packet. Therefore the listed rows exhaust the local NameCert obligations. □

Theorem 1045.3 (Rule 110 glider catalog classifier stability)**.** *If two accepted* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *carriers agree on their Rule 110 row, finite row window, phase-period-displacement row, displayed local-match row, componentwise transport row, continuation route, provenance row, and local naming row, then their catalog-tag rows classify together under the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}$ *classifier. In particular, classifier agreement cannot be changed by renaming the tag while leaving the displayed window and local-match route fixed.*

Proof. Use Definition 1045.2. An accepted tag read reaches G only through the route $R \longmapsto W \longmapsto P \longmapsto D \longmapsto G$ with H, C, K, N attached.

For two accepted carriers with the same R, W, P, D, H, C, K, N, the finite lookup inputs and the replay route are the same. The Martinez-style tag row is therefore read from the same displayed phase and local-match data inside the same local NameCert surface.

A different classification would require a different row window, phase-period-displacement entry, local-match result, route, provenance, or local naming row, or else an external host recognizer. None is present in the carrier. Hence the two catalog-tag rows classify together. □

Theorem 1045.4 (Rule 110 glider catalog substrate handoff)**.** *Every* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *read handed to a cellular substrate consumer exposes the finite Rule 110 row window, catalog tag, phase-period-displacement row, displayed local match,* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row before the substrate may cite it as glider-catalog evidence. The handoff uses the cellular rule surface of chapter 934, the substrate carrier of chapter 942, and the finite compile-frontier coordinate of chapter 952; it exports no unbounded orbit classifier, host visual recognition, or additional universality witness.*

Proof. Start with carrier admission from Definition 1045.1. The packet gives the displayed rows $R, W, G, P, D, H, C, K, N$ and excludes unbounded orbit, host recognizer, hidden catalog, and Cook universality coordinates.

Apply the NameCert obligations theorem. It forces every accepted catalog read through the finite route from the Rule 110 row and row window to the phase row, local-match row, and catalog tag.

The sibling handoff is then only a provenance and route handoff. The cellular automaton chapter supplies the finite local rule and orbit surface, the cellular substrate chapter supplies the carrier row through which a substrate consumer reads the window, and the Cook compile-frontier witness supplies the finite coordinate for Cook-facing audit reads. Since none of those siblings adds a hidden classifier coordinate to J, the handoff remains exactly the displayed finite glider-catalog packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ packet is generated from a Rule 110 rule row, finite row-window row, catalog-tag row, phase-period-displacement row, displayed local-match row, componentwise hsame transports, displayed continuation routes, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1045.1, Definition 1045.2, Definition 1045.3, and Definition 1045.4 bind the finite Rule 110 row-window, catalog tag, phase-period-displacement, local-match, transport, route, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenGliderCatalog}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove Lean verification, global classification of all Rule 110 rows, Cook universality, host evaluator correctness, hidden visual recognition, or a public standard bridge.

Upgrade path: A later checked surface can provide Nontrivial and FieldFaithful instances for the displayed Rule 110 glider-catalog rows.

Chapter 1046

A Concrete Naming Certificate for TypingJudgmentClassifier$^{\uparrow}$

The TypingJudgmentClassifier$^{\uparrow}$ packet realizes the type-checking-as-classifier-membership vision of chapter 9367 as a finite judgment classifier. It consumes the membership surface of chapter 476, the derivation trace of chapter 598, and the route join of chapter 809, then exposes one accepted typing-judgment row through Ext membership, derivation-route replay, SigRel readback, componentwise hsame transport, Pkg provenance, and local NameCert data.

Falsifiable prediction: A packet with a typing derivation route but no displayed Ext-membership row is not an accepted TypingJudgmentClassifierUp carrier.

Independence witness: typeclassifiermembership, classifiertyping, subjectreduction, typecheckingderivationtrace, and classifiermorphism: none is bijective because the typing-judgment classifier carries the displayed Ext-membership row, derivation route, SigRel readback, Cont replay, hsame transport, Pkg provenance, and local NameCert surface together, while each sibling isolates only one membership, typing, preservation, trace, or morphism face.

Definition 1046.1 (TypingJudgmentClassifier carrier)**.** *A* TypingJudgmentClassifier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (J, M, D, S, C, H, P, N)$$

with the following displayed rows:

1. *J is the typing-judgment row whose membership in the accepted classifier is being read;*
2. *M is the displayed* Ext *membership row certifying that J is accepted by the local type-checking classifier;*
3. *D is the finite derivation-route row that explains how the classifier reached M;*
4. *S is the* SigRel *readback row naming the accepted typing judgment as the public reading of the membership row along D;*
5. *C records the displayed* Cont *replay routes for reading the judgment row, membership row, derivation route, readback row, provenance row, and local naming row through the same finite packet;*
6. *H stores componentwise* hsame *transports for J, M, D, S, C, P, N;*
7. *P is the* Pkg *provenance over chapter 9367, chapter 476, chapter 489, chapter 483, chapter 598, chapter 590,* BHist*,* Ext*,* SigRel*,* Cont*,* hsame*, and* NameCert*;*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypingJudgmentClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for host type semantics, hidden conversion, quotient syntax equality, unrecorded derivation search, closed subject reduction, or an ambient classifier morphism.

Theorem 1046.2 (TypingJudgmentClassifier NameCert obligations)**.** *For the carrier of Definition 1046.1, the local* $\mathsf{NameCert}_{\mathsf{TypingJudgmentClassifier}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission exposes precisely $T = (J, M, D, S, C, H, P, N)$;*
2. *membership exactness reads the accepted typing-judgment decision only from the displayed* Ext *row M;*
3. *derivation-route replay reads the route only through D and the displayed* Cont *rows C;*
4. *public readback exposes the accepted judgment only through the* SigRel *row S transported by H;*
5. *provenance and non-escape bind every consumer read to P, N and exclude host type semantics, hidden conversion, quotient syntax equality, unrecorded derivation search, closed subject reduction, and ambient classifier morphisms.*

Lean4 ✓ `BEDC.Derived.TypingJudgmentClassifierUp.TypingJudgmentClassifierCarrier namecert obligations`

Proof. Carrier admission is projection from the finite tuple in Definition 1046.1. The judgment, membership, derivation route, readback, replay, transport, provenance, and naming rows are visible coordinates.

Membership exactness follows because M is the only Ext membership coordinate. The derivation route D and continuation row C can replay how the decision is read, but they cannot replace M by a hidden checker.

The public classifier read is exhausted by S, H, P, N. These rows name, transport, package, and locally certify the same displayed membership event, so the excluded host and quotient-side objects have no coordinate from which a consumer could recover them. □

Theorem 1046.3 (TypingJudgmentClassifier derivation route)**.** *Every accepted* $\mathsf{TypingJudgmentClassifier}^{\uparrow}$ *carrier routes the typing judgment through displayed classifier membership before any public readback is available:*

$$J \to M \to D \to S.$$

The replay and transport rows C, H preserve that order for downstream consumers, and the package and local name rows P, N record the finite provenance of the same route.

Proof. Inspect the carrier inventory. The judgment row J is not an accepted classifier decision by itself; the accepted decision is the displayed membership row M.

The derivation route D records how the membership row is reached, and the SigRel row S records the public reading of that accepted judgment. The continuation and transport rows replay and preserve these coordinates componentwise.

Since P, N only package and name the same rows, every consumer route factors through the displayed chain from judgment to membership, derivation, and readback. □

Lean4 ✓ `BEDC.Derived.TypingJudgmentClassifierUp.TypingJudgmentClassifierCarrier derivation route`

Theorem 1046.4 (TypingJudgmentClassifier TasteGate readiness)**.** *The Lean TasteGate module for* $\mathsf{TypingJudgmentClassifier}^{\uparrow}$ *must provide explicit* $\mathsf{Nontrivial\ TypingJudgmentClassifier}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ TypingJudgmentClassifier}^{\uparrow}$ *instances. Nontriviality separates the typing-judgment row* J *from the membership row* M*, and field faithfulness preserves the displayed rows* J, M, D, S, C, H, P, N *for every TasteGate-facing consumer.*

Lean4 ✓ `BEDC.Derived.TypingJudgmentClassifierUp.TasteGate.TypingJudgmentClassifierTasteGate readiness`

Lean4 ✓ `BEDC.Derived.TypingJudgmentClassifierUp.TypingJudgmentClassifierUp taste gate readiness`

Proof. Nontriviality is forced by the falsifiability boundary: a derivation route without the displayed Ext row is rejected, so the row carrying the judgment cannot collapse with the row carrying accepted membership.

Field faithfulness is the row-preservation condition for the same packet. A consumer may inspect the judgment, membership, derivation route, readback, replay, transport, provenance, and local name rows, but no unlisted type-checker state or host semantic object. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypingJudgmentClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{TypingJudgmentClassifier}^{\uparrow}$ packet is generated from a typing-judgment row, a displayed Ext-membership row, a finite derivation route, SigRel readback, displayed Cont replay, componentwise hsame transports, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypingJudgmentClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1046.1, Definition 1046.2, Definition 1046.3, and Definition 1046.4 bind the finite typing-judgment classifier surface over Ext membership, derivation routes, SigRel readback, Cont replay, hsame transport, Pkg provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypingJudgmentClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, host type semantics, hidden conversion, quotient syntax equality, unrecorded derivation search, closed subject reduction, or ambient classifier morphisms.

Upgrade path: Obligation closure requires checked carrier admission, membership exactness, derivation-route replay, SigRel readback, non-escape, nontriviality, and field faithfulness for the same finite packet.

Chapter 1047

ObserverDirectionUp NameCert construction

The $\mathsf{ObserverDirection}^{\uparrow}$ surface realizes the direction-sensitive readback part of chapter 9418 as a finite observer-to-target carrier. It sits after the regular observer packet of chapter 431, the observer-history identity packet of chapter 479, and the licensed comparison packet of chapter 1031. Its role is to keep the observer route, target readback, direction token, comparison ledger, and local naming row together before a consumer treats a comparison as an oriented observation rather than an unordered pair of histories.

Falsifiable prediction: Removing the direction-token row collapses two distinct observer-to-target routes and breaks the licensed comparison handoff.

Independence witness: Compare observerregular, observerhistidentity, observerlicensedcomparison, observerbridgeledger, and streamname siblings: none is bijective because this carrier alone stores the direction token together with observer route, target readback, and comparison ledger.

Definition 1047.1 (Observer direction carrier)**.** *An* $\mathsf{ObserverDirection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, T, U, L, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the observer-row packet, read through the regular observer and observer-history identity surfaces;*
2. *T is the target-row packet whose readback is being oriented;*
3. *U is the direction-token row, naming which observer-to-target route is being read;*
4. *L is the licensed comparison ledger, read through the sibling surface of chapter 1031;*
5. *B is the target readback row produced only after R, T, U, L have been displayed together;*
6. *H records the componentwise* hsame *transports for observer, target, direction token, comparison ledger, readback, route, provenance, and local name rows;*
7. *C records the displayed* Cont *routes from the observer row to the target row and from that oriented route to the comparison readback;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverRegular}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverLicensedComparison}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{ObserverDirection}^{\uparrow}}$ *row.*

The carrier accepts an oriented observer comparison only when the observer row, target row, direction token, licensed comparison, readback, hsame *transport, continuation route, provenance, and local naming row are all displayed in one finite packet. It exports no unordered comparison, hidden observer-substance, host equality of observers, global frame, unlicensed readback, or route outside the displayed direction token.*

Lean4 ✓ `BEDC.Derived.ObserverDirectionUp.ObserverDirectionUp carrier fields`

Theorem 1047.2 (Observer direction NameCert obligations)**.** *For the carrier of Definition 1047.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverDirection}^{\uparrow}}$ *surface consists exactly of carrier admission for $O = (R, T, U, L, B, H, C, P, N)$, observer-row anchoring through R, target-row anchoring through T, direction-token exactness through U, licensed comparison handoff through L, target readback through B, componentwise* hsame *transport through H, displayed continuation replay through C, package provenance through P, local naming through N, and non-escape from unordered comparison, hidden observer-substance, host equality, global frames, unlicensed readback, and routes outside the displayed direction token.*

Proof. Project the accepted packet to the nine rows of Definition 1047.1. The observer row and target row become comparable only after the direction token and licensed comparison ledger are displayed with them.

The direction-token exactness is the row U. A public read may consume the observer-to-target route only through the displayed continuation row C, and the licensed comparison handoff must pass through L before the target readback row B is available.

The transport and provenance rows then bind the read to the observerregular, observerhistidentity, and observerlicensedcomparison siblings cited above. Since none of the excluded objects is a coordinate of O, the local NameCert surface is exhausted by the listed finite obligations. □

Lean4 ✓ `BEDC.Derived.ObserverDirectionUp.ObserverDirectionUp namecert obligations`

Theorem 1047.3 (Observer direction falsifiable boundary)**.** *If the direction-token row U is removed from an accepted* $\mathsf{ObserverDirection}^{\uparrow}$ *packet, two packets that differ only by reversing the observer-to-target route have the same remaining rows R, T, L, B, H, C, P, N. The licensed comparison handoff can then no longer distinguish the oriented readback, so the packet is not an* $\mathsf{ObserverDirection}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.ObserverDirectionUp.ObserverDirectionFalsifiableBoundary`

Proof. Remove U from $O = (R, T, U, L, B, H, C, P, N)$. The remaining rows still display an observer row, target row, comparison ledger, readback, transport, continuation route, provenance, and local name, but they no longer name which oriented route the readback follows.

A reversed observer-to-target route can preserve the unordered pair of visible rows while changing the intended comparison handoff. The licensed comparison sibling records a comparison ledger, not the missing direction token itself.

Thus the carrier loses the row that separates the two oriented routes. The result is outside Definition 1047.1, and the falsifiable prediction is the loss of the licensed oriented handoff. □

Theorem 1047.4 (Observer direction nontrivial boundary)**.** *The* $\mathsf{ObserverDirection}^{\uparrow}$ *packet is nontrivial at the paper boundary: an accepted packet with direction token U and the same observer, target, comparison, transport, route, provenance, and name rows but opposite displayed direction token is not identified by the carrier predicate. The later Lean* `TasteGate.lean` *must include an explicit* Nontrivial $\mathsf{ObserverDirection}^{\uparrow}$ *instance for this displayed carrier.*

Lean4 ✓ `BEDC.Derived.ObserverDirectionUp.ObserverDirectionTasteGate single carrier alignment`

Proof. The carrier contains the direction-token row as a coordinate. Two packets with different displayed direction tokens therefore differ as finite BHist packets even when their observer, target, comparison, transport, route, provenance, and name rows agree.

The licensed comparison row does not erase this distinction, because it is read after the direction token has fixed the observer-to-target route. The paper boundary is therefore nontrivial: the packet can witness two oriented reads over the same visible endpoints. □

Theorem 1047.5 (Observer direction field-faithful boundary)**.** *Every public* $\mathsf{ObserverDirection}^{\uparrow}$ *consumer preserves the nine displayed fields* $R, T, U, L, B, H, C, P, N$ *of Definition 1047.1. In particular, it may not identify a direction-token change with a target-readback change, a licensed-comparison change, or a provenance change. The later Lean* `TasteGate.lean` *must include an explicit* $\mathsf{FieldFaithful\ ObserverDirection}^{\uparrow}$ *instance for this displayed carrier.*

Proof. Read a public consumer as a projection from the accepted carrier. The only fields available to the consumer are the observer row, target row, direction token, licensed comparison ledger, target readback, hsame transport row, continuation route, package provenance, and local name.

Field faithfulness follows from the order of the read. The direction token U gates the route in C, the licensed comparison ledger L gates the readback B, and the provenance row P records the sibling sources separately.

Since these rows occupy separate coordinates, a consumer that merges one coordinate into another no longer preserves the carrier of Definition 1047.1. The public boundary is therefore faithful to the listed fields. □

Lean4 ✓ `BEDC.Derived.ObserverDirectionUp.TasteGate.ObserverDirectionTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverDirection}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverDirection}^{\uparrow}$ packet is generated from observer-row, target-row, direction-token, licensed-comparison, readback, hsame transport, displayed Cont route, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverDirection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1047.1, Definition 1047.2, Definition 1047.3, Definition 1047.4, and Definition 1047.5 specify the finite oriented observer-readback carrier, its NameCert obligations, direction-token boundary, nontriviality boundary, and field-faithfulness boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverDirection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full observer-history identity, observer-filter completeness, hidden observer state, quotient history identity, global observer frames, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked TasteGate carrier admission, direction-token exactness, licensed-comparison handoff, readback confinement, nontriviality, and field faithfulness for the same packet.

Chapter 1048

A Concrete Naming Certificate for $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$

The $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ packet realizes the generated-recognizer route of chapter 9407 as a finite NameCert carrier. It sits beside the event-flow route of chapter 830, the audit-map packet of chapter 836, and the recognizer audit packet of chapter 820: those siblings display event-flow routing, compiler audit inventory, and recognizer-facing audit rows, while this chapter binds the source channel, generated recognizer, certificate gate, and non-evidence refusal as one consumer packet.

Definition 1048.1 (GroundCompiler recognizer-flow carrier)**.** *A* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, S, R, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *E is the event-flow row naming the formal GroundCompiler event packet under inspection;*
2. *S is the source-channel row recording the legal channel encoding, decoder address, and lossless readback route for E;*
3. *R is the generated-recognizer row recording package recognition, local certificate recognition, derivation-certificate recognition, theorem-identifier recognition, chapter-flow recognition, and accepted-export recognition;*
4. *Q is the certificate-gate row through which an accepted generated recognizer result may be read as GroundCompiler evidence;*
5. *B is the non-evidence row separating source text, carried metadata, metric reports, motif reports, compiler reports, proof text, host compiler state, and raw classifier quotients from certificate evidence;*
6. *H records componentwise* hsame *transports for event-flow, source-channel, generated-recognizer, certificate-gate, non-evidence, continuation, provenance, and local naming rows;*
7. *C records displayed* Cont *routes from event flow to source channel, from source-channel readback to generated recognition, from generated recognition to certificate gate, from excluded report-like objects to non-evidence, and from the accepted packet to downstream consumers;*

8. *P is the* Pkg *provenance over chapter 9407, chapter 830, chapter 836, chapter 820,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares exactly the displayed coordinates $E, S, R, Q, B, H, C, P, N$. It has no coordinate for hidden compiler input, proof reconstructed from source text, metric evidence, motif identity, report evidence, raw classifier equality, or an unmediated generated export.

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognizerFlowUp.GroundCompilerRecognizerFlowTasteGate`
`single carrier alignment`

Theorem 1048.2 (GroundCompiler recognizer-flow NameCert obligations)**.** *For every accepted* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *carrier*

$$G = (E, S, R, Q, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}}$ *surface consists exactly of event-flow readback through E, source-channel lossless routing through S, generated-recognizer mediation through R, certificate-gate acceptance through Q, non-evidence refusal through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The rows E, S, R, Q, B are the public data coordinates, and H, C, P, N transport, route, locate, and name those same coordinates.

Read the route in order. The event-flow row supplies the formal input, the source-channel row supplies legal encoding and lossless readback, the generated-recognizer row mediates recognition of packages and certificate-like compiler exports, and the certificate-gate row is the only positive acceptance point.

The non-evidence row records every excluded channel: source text, carried metadata, metric reports, motif reports, compiler reports, proof text, host state, and raw classifier quotients. Since no carrier coordinate converts B into E, S, R, Q, or proof evidence, the displayed rows exhaust the local NameCert obligation surface. □

Theorem 1048.3 (GroundCompiler recognizer-flow certificate-gate exactness)**.** *Every accepted* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *export that reaches the certificate gate Q first passes through the displayed event-flow row E, source-channel row S, and generated-recognizer row R. An export that omits the generated recognizer row cannot be read as accepted GroundCompiler evidence by* hsame *transport,* Cont *replay,* Pkg *provenance, or the local* NameCert *row.*

Proof. Start with the carrier classifier of Definition 1048.1. A certificate-facing read is accepted only when it compares the displayed coordinates $E, S, R, Q, B, H, C, P, N$.

Apply the NameCert obligation surface of Definition 1048.2. It orders the positive route through event flow, source channel, generated recognizer, and certificate gate, so Q is not reachable as a standalone export coordinate.

Finally inspect the structural rows. Transport, continuation, provenance, and local naming preserve the same packet, but none supplies a replacement for R. Therefore a compiler export without the generated-recognizer row is outside the accepted GroundCompiler evidence route. □

Theorem 1048.4 (GroundCompiler recognizer-flow non-evidence refusal)**.** *For every accepted* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *carrier, source text, carried metadata, metric reports, motif reports, compiler reports, proof text, host compiler state, and raw classifier quotients can be read*

only through the non-evidence row B. They cannot become event-flow input, source-channel readback, generated-recognizer evidence, certificate-gate evidence, or BEDC proof evidence inside the packet.

Proof. Project the packet and isolate B. It is the only coordinate assigned to report-like, host-like, text-like, metric, motif, and quotient objects.

Use certificate-gate exactness. Any route to accepted GroundCompiler evidence has already passed through E, S, R, Q, so it cannot start from an object classified only by B.

The remaining rows H, C, P, N preserve and replay displayed coordinates. They do not add a constructor from non-evidence to event flow, channel readback, generated recognition, certificate evidence, or proof evidence, so the refusal row is stable under every public consumer read. □

Theorem 1048.5 (GroundCompiler recognizer-flow TasteGate carrier readback)**.** *For every displayed* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *packet*

$$G = (E, S, R, Q, B, H, C, P, N),$$

the TasteGate row for BEDC.Derived.GroundCompilerRecognizerFlowUp *reads the single carrier through exactly the same event-flow, source-channel, generated-recognizer, certificate-gate, non-evidence, transport, continuation, provenance, and local naming coordinates. A TasteGate consumer cannot replace G by a hidden compiler input, raw recognizer quotient, proof text, metric report, motif report, or unmediated generated export.*

Lean4 ✓ `BEDC.Derived.GroundCompilerRecognizerFlowUp.taste gate`

Proof. Start from the carrier definition. The packet has exactly the displayed rows $E, S, R, Q, B, H, C, P, N$, and the classifier compares only those rows.

Apply the existing single-carrier TasteGate alignment named by the Lean surface. It makes the TasteGate entry a readback of the same finite carrier rather than a second carrier with extra compiler-facing coordinates.

Finally use the non-evidence refusal row. Objects such as proof text, reports, motifs, raw quotients, and unmediated exports are recorded only at B, so the TasteGate read cannot promote them into the positive carrier route. □

Theorem 1048.6 (GroundCompiler recognizer-flow field-faithful row projection)**.** *For every accepted* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *packet $G = (E, S, R, Q, B, H, C, P, N)$, the field-faithful projection of the TasteGate surface is componentwise: E remains the event-flow row, S the source-channel row, R the generated-recognizer row, Q the certificate-gate row, B the non-evidence row, H the* hsame *transport row, C the* Cont *replay row, P the* Pkg *provenance row, and N the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}}$ *row.*

Proof. Project the accepted packet and keep the row order fixed. Field faithfulness can only report the fields already exposed by the carrier and by the NameCert obligation surface.

The positive route rows E, S, R, Q are separated from the refusal row B, while H, C, P, N are structural rows attached to the same carrier. No projection step has a coordinate from which to synthesize a compiler proof, hidden recognizer state, or report-like witness.

Thus the TasteGate field readback is exactly componentwise over $E, S, R, Q, B, H, C, P, N$. □

Theorem 1048.7 (GroundCompiler recognizer-flow nontrivial witness boundary)**.** *The nontrivial witness for* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *separates an accepted recognizer-flow packet from*

every packet that omits one of $E, S, R, Q, B, H, C, P, N$ or tries to identify the non-evidence row B with event-flow, source-channel, generated-recognizer, certificate-gate, or proof-evidence data. The separation is witnessed inside the finite carrier rows and does not assert compiler correctness, self-hosting, or proof reconstruction.

Proof. Use the carrier inventory to compare two candidate packets. A packet missing any one of the displayed coordinates cannot satisfy the same row-by-row classifier as G.

For the positive evidence route, apply certificate-gate exactness. Accepted evidence must pass through E, S, R, Q, and none of the structural rows replaces a missing generated-recognizer or certificate-gate coordinate.

For the boundary side, apply non-evidence refusal. The row B is stable under transport, continuation, provenance, and local naming, so it remains separated from evidence and proof rows. This gives a finite nontrivial witness boundary without importing compiler correctness or proof reconstruction. □

Falsifiable prediction: A compiler export that omits the generated recognizer row cannot pass the certificate gate as accepted GroundCompiler evidence.

Independence witness: groundcompilereventflowaudit, groundcompilerauditmap, auditmapfamily, auditmapcoverageledger, namecertgenerated: event-flow audit lacks generated-recognizer mediation, audit-map packets lack the certificate-gate row, audit families lack source-channel lossless routing, coverage ledgers lack the non-evidence refusal row, and NameCert-generated surfaces do not carry the compiler recognizer flow as a single packet.

The Lean TasteGate follow-up for BEDC.Derived.GroundCompilerRecognizerFlowUp must provide Nontrivial GroundCompilerRecognizerFlow$^{\uparrow}$ and FieldFaithful GroundCompilerRecognizerFlow$^{\uparrow}$ instances for the displayed carrier $E, S, R, Q, B, H, C, P, N$.

Closure status for NameCert$_{\text{GroundCompilerRecognizerFlow}^{\uparrow}}$.

Constructive story: A GroundCompilerRecognizerFlow$^{\uparrow}$ packet is generated from event-flow, source-channel, generated-recognizer, certificate-gate, non-evidence, componentwise hsame, displayed Cont, Pkg, and local NameCert rows; the TasteGate row is recorded at BEDC.Derived.GroundCompilerRecognizerFlowUp.taste_gate.

Theory closure: TheoryGate(NameCert$_{\text{GroundCompilerRecognizerFlow}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1048.1, Definition 1048.2, Definition 1048.3, Definition 1048.4, Definition 1048.5, Definition 1048.6, and Definition 1048.7 bind the finite GroundCompiler recognizer-flow carrier, TasteGate carrier readback, field-faithful projection, nontrivial witness boundary, certificate-gate exactness, and non-evidence refusal.

Formal verification: FormalStatus(NameCert$_{\text{GroundCompilerRecognizerFlow}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.GroundCompilerRecognizerFlowUp.GroundCompilerRecognizerFlowTasteGate\_single\_carrier\_`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close compiler correctness, self-hosting coverage, proof reconstruction from source text, metric evidence, motif evidence, report evidence, audit cleanliness, or bridge checking.

Upgrade path: The paper closure axis is complete for the event-flow and source-channel recognizer packet; bridge checking remains independent unless a StdBridge target is supplied.

Chapter 1049

A Concrete Naming Certificate for AxisCarryRouteSeparation$^{\uparrow}$

The AxisCarryRouteSeparation$^{\uparrow}$ packet records the finite route boundary around the Zeckendorf carry classifier of chapter 9345. It reads the carry-budget surface of chapter 692, the unary-spine normalizer of chapter 599, and the unary-direction normalizer of chapter 1010 as sibling routes, while keeping carry-relatedness separate from hsame identification.

It is an ai over the ZeckendorfCarryClassifier$^{\uparrow}$, AxisZeckendorfCarryBudget$^{\uparrow}$, AxisUnarySpineNormalizer$^{\uparrow}$, and UnaryDirectionNormalizer$^{\uparrow}$ sibling surfaces, with the route-separation row serving only as the finite boundary bundle.

Definition 1049.1 (Axis carry route separation carrier)**.** *An* AxisCarryRouteSeparation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, U, D, S, T, B, R, H, C, P, N)$$

with the following visible rows:

1. *A is the accepted* AxisZeckendorfCarryBudget$^{\uparrow}$ *row containing the displayed carry source, carry target, normal-form target, value ledger, and non-*hsame *boundary;*
2. *U is the* AxisUnarySpineNormalizer$^{\uparrow}$ *row through which the axis-unary spine is read before a direction route is visible;*
3. *D is the* UnaryDirectionNormalizer$^{\uparrow}$ *row through which the direction-normalized unary packet is read;*
4. *S is the carry source row generated by the displayed* ZCarry *generator;*
5. *T is the normal target row reached by the same carry route;*
6. *B is the refusal row recording that the carry-related source and target are not identified by* hsame*;*
7. *R is the route ledger separating carry classification, normal-form routing, and refusal replay;*
8. *H records componentwise* hsame *transports for the sibling rows without converting the carry relation into history sameness;*
9. *C records the displayed* Cont *routes for reading $A, U, D, S, T, B, R, H, P, N$;*
10. *P is the* Pkg *provenance over* ZCarry$^{\uparrow}$*,* AxisZeckendorfCarryBudget$^{\uparrow}$*,* AxisUnarySpineNormalizer$^{\uparrow}$*,* UnaryDirectionNormalizer$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*

11. N is the local $\mathsf{NameCert}_{\mathsf{AxisCarryRouteSeparation}^{\uparrow}}$ *row.*

The carrier has no coordinate for promoting the carry relation to hsame*, no global carry rewrite, no host arithmetic quotient, and no normalizer that forgets the refusal row B.*

Theorem 1049.2 (Axis carry route separation boundary)**.** *Every accepted* $\mathsf{AxisCarryRouteSeparation}^{\uparrow}$ *packet $Q = (A, U, D, S, T, B, R, H, C, P, N)$ exposes carry-relatedness only through the displayed carry row $S \to T$, exposes normal-form routing only through the sibling rows A, U, D, and exposes the history-sameness refusal only through B, R, H, C, P, N. A consumer therefore receives a finite packet separating carry-relatedness, normal-target routing, and non-*hsame *refusal, and it cannot turn the Zeckendorf carry row of chapter 9345 into an* hsame *identification.*

Proof. Project the finite carrier of Definition 1049.1. The rows S and T are the only carry source and target rows, while A, U, D are the only sibling route rows for the carry budget, axis-unary spine normalizer, and unary-direction normalizer.

The refusal row B records the non-hsame boundary supplied by the Zeckendorf carry classifier. The route ledger R, componentwise transport row H, continuation row C, provenance row P, and local name N replay that refusal without adding a kernel equality coordinate.

Finally use coordinate exhaustion. Since Q contains no host quotient, no global rewrite, and no coordinate that identifies S with T by hsame, every public read remains within the finite carry route, normal-target route, and refusal ledger displayed above. □

Lean4 ✓ `BEDC.Derived.AxisCarryRouteSeparationUp.AxisCarryRouteSeparationTasteGate single carrier alignment`

Falsifiable prediction: Any accepted route that turns a Zeckendorf carry row into hsame must pass through the refusal ledger rather than the public carry-route classifier.

Independence witness: zeckendorf-carry-classifier, axiszeckendorfcarrybudget, axisunaryspinenormalizer, unarydirectionnormalizer: the route-separation packet is not bijective to any sibling because it carries the carry row, normal-target row, non-hsame refusal, and route ledger simultaneously.

The Lean taste-gate file is `lean4/BEDC/Derived/AxisCarryRouteSeparationUp/TasteGate.lean`. It must provide explicit `Nontrivial AxisCarryRouteSeparationUp` and `FieldFaithful AxisCarryRouteSepara` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{AxisCarryRouteSeparation}^{\uparrow}}$.

Constructive story: An $\mathsf{AxisCarryRouteSeparation}^{\uparrow}$ packet is generated from carry-budget, axis-unary-spine, unary-direction, carry source, normal target, refusal, route-ledger, transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxisCarryRouteSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 1049.1 and Definition 1049.2 specify the finite carrier separating carry-relatedness, normal-form routing, and history-sameness refusal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxisCarryRouteSeparation}^{\uparrow}}$, unformalized)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.axisCarryRouteSeparationDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, global carry normalization, host arithmetic quotienting, or any kernel-level identification of carry-related histories.

Upgrade path: Scoped closure requires Lean encoding and checked routing for carrier admission, route separation, nontriviality, and field faithfulness over the same packet.

Chapter 1050

A Concrete Naming Certificate for $\mathsf{HostTotalTrace}^{\uparrow}$

The $\mathsf{HostTotalTrace}^{\uparrow}$ packet is the finite host-trace object for the total-host side of chapter 9370. It sits beside chapter 964, chapter 960, chapter 936, and chapter 970. Its distinguishing datum is that one packet carries host readback, finite fuel, substrate endpoint, timeout refusal, componentwise transport, route replay, provenance, and local naming rows without collapsing them into the broader step-indexed host boundary.

Falsifiable prediction: HostTotalTraceUp has no morphism to StepIndexedTotalHostUp that preserves timeout refusal while forgetting trace endpoint readback.

Independence witness: StepIndexedTotalHostUp, FuelIndexedSubstrateBridgeUp, ZNormalUp, ReductionFuelBoundaryUp: HostTotalTraceUp carries both host readback and timeout-refusal rows, while these siblings separately expose total-host boundary, substrate bridge, normal-form predicate, or fuel boundary without the same trace-level packet.

Definition 1050.1 (HostTotalTrace carrier)**.** *A* $\mathsf{HostTotalTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H, F, R, E, U, L, C, P, N)$$

with the following displayed rows:

1. *H is the total-host readback row, naming the finite host computation being inspected;*
2. *F is the finite fuel row under which the trace is accepted;*
3. *R is the normal-form readback row, read through the host-side normality discipline of* $\mathsf{ZNormal}^{\uparrow}$*;*
4. *E is the substrate endpoint row reached by the finite trace when an endpoint is displayed;*
5. *U is the timeout-refusal row, recording fuel exhaustion as a refused endpoint rather than as nontermination, convergence, or halting evidence;*
6. *L records componentwise* hsame *transports for H, F, R, E, U, C, P, N;*
7. *C records displayed* Cont *routes from host readback to fuel, from fuel to normal-form readback, from the same fuel to substrate endpoint or timeout refusal, and from barred semantic requests to the refusal row;*
8. *P is the* Pkg *provenance over* $\mathsf{HostTotalTrace}^{\uparrow}$*, chapter 9370,* $\mathsf{StepIndexedTotalHost}^{\uparrow}$*,* $\mathsf{FuelIndexedSubstrateBridge}$ $\mathsf{ZNormal}^{\uparrow}$*,* $\mathsf{ReductionFuelBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{HostTotalTrace}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $H, F, R, E, U, L, C, P, N$. It has no coordinate for host partiality, unbounded substrate completion, quotient endpoint equality, host-substrate equality, or a detached timeout oracle.

Definition 1050.2 (HostTotalTrace classifier)**.** *The* $\mathsf{HostTotalTrace}^{\uparrow}$ *classifier compares two accepted packets only by the displayed tuple*

$$H, F, R, E, U, L, C, P, N.$$

Two packet presentations agree when their host readback, fuel, normal-form readback, substrate endpoint, timeout-refusal, componentwise transport, route, provenance, and local naming rows agree componentwise. The classifier does not compare partial host evaluators, unbounded substrate completions, quotient endpoints, host-substrate equalities, or detached timeout oracles.

Theorem 1050.3 (HostTotalTrace NameCert obligations)**.** *For the carrier of Definition 1050.1 and classifier of Definition 1050.2, the local* $\mathsf{NameCert}_{\mathsf{HostTotalTrace}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission and classifier exactness for $T = (H, F, R, E, U, L, C, P, N)$;*
2. *host readback and finite fuel exposure through H, F;*
3. *normal-form readback and substrate endpoint exposure through R, E;*
4. *timeout-refusal exactness through U;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through L, C, P, N.*

Proof. Project the carrier and classifier in Definition 1050.1 and Definition 1050.2. They display exactly the rows $H, F, R, E, U, L, C, P, N$, so carrier admission and classifier exactness cannot introduce a detached endpoint, alternate fuel, or hidden timeout branch.

Read the constructive rows in order. The host readback row H is accepted only under the finite fuel row F; the normal-form readback R and substrate endpoint E are public only as rows of that same finite trace packet.

The refusal row U accounts for fuel exhaustion. It records timeout as a boundary outcome and not as nontermination, halting, convergence, host-substrate equality, or quotient endpoint equality.

Finally L, C, P, N preserve and replay only the same finite packet. The five listed rows therefore exhaust the local NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.HostTotalTraceUp.HostTotalTraceTasteGate single carrier alignment`

Theorem 1050.4 (HostTotalTrace timeout endpoint separation)**.** *Every accepted* $\mathsf{HostTotalTrace}^{\uparrow}$ *packet separates endpoint readback from timeout refusal by the displayed route*

$$H, F, R, E, U, L, C, P, N.$$

A consumer may read the host readback, fuel, normal-form readback, displayed substrate endpoint, and timeout-refusal rows. It may not erase E while preserving U, turn timeout into nontermination, turn endpoint readback into unbounded convergence, or identify the host trace with the substrate bridge.

Proof. Start from the obligation surface of Definition 1050.3. The endpoint row E and timeout-refusal row U are separate coordinates of the same accepted packet.

Compare the sibling surfaces. $\mathsf{StepIndexedTotalHost}^{\uparrow}$ supplies the broader total-host boundary, $\mathsf{FuelIndexedSubstrateBridge}^{\uparrow}$ supplies the substrate bridge, $\mathsf{ZNormal}^{\uparrow}$ supplies the normal-form predicate, and $\mathsf{ReductionFuelBoundary}^{\uparrow}$ supplies the fuel-boundary refusal discipline. None lets this packet forget endpoint readback while retaining the local timeout row.

The route and transport rows L, C, P, N preserve that separation. Hence a downstream consumer can replay the finite trace, but it cannot convert timeout into a semantic nontermination claim or endpoint readback into unbounded convergence. □

The proposed Lean follow-up is the module `lean4/BEDC/Derived/HostTotalTraceUp/TasteGate.lean`. It must provide explicit $\mathsf{Nontrivial\ HostTotalTrace}^{\uparrow}$ and $\mathsf{FieldFaithful\ HostTotalTrace}^{\uparrow}$ instances preserving the host readback, finite fuel, normal-form readback, substrate endpoint, timeout refusal, componentwise transport, displayed route, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HostTotalTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{HostTotalTrace}^{\uparrow}$ packet is generated from host readback, finite fuel, normal-form readback, substrate endpoint, timeout-refusal, componentwise hsame transport, displayed Cont route, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HostTotalTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1050.1, Definition 1050.2, Definition 1050.3, and Definition 1050.4 specify the finite host-trace packet separating endpoint readback from timeout refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HostTotalTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, host partiality, unbounded substrate completion, halting decidability, quotient endpoint equality, or host-substrate equality.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, host readback, fuel exposure, normal-form readback, endpoint exposure, timeout-refusal exactness, nontriviality, and field faithfulness for the same packet.

Chapter 1051

A Concrete Naming Certificate for $\mathsf{TailCauchyRouter}^{\uparrow}$

The $\mathsf{TailCauchyRouter}^{\uparrow}$ packet realizes the tail-selection route used by the Real-completion vision of chapter 9238. It sits after the finite window service of chapter 126, the rational readback of chapter 301, the regular-Cauchy source surface of chapter 660, the limit handoff of chapter 386, and the terminal Real seal of chapter 38. The packet is a finite route certificate for a displayed tail selector, not a quotient stream equality, ambient completeness theorem, hidden choice principle, or host-level real equality.

Definition 1051.1 (Tail Cauchy router carrier)**.** *A* $\mathsf{TailCauchyRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, W, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the tail selector that names the accepted tail threshold for the requested precision;*
2. *M is the shared modulus row used by every source and consumer read in the packet;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window reached at the selected tail;*
4. *R is the regular-Cauchy source row whose tail is being routed;*
5. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row fed by the selected window and source row;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal endpoint exported to the consumer;*
7. *H records componentwise* hsame *transport for the selector, modulus, window, source, limit handoff, endpoint, route, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes from the selected tail and shared modulus to the window, from the window and source row to the limit handoff, and from the handoff to the Real endpoint;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TailCauchyRouter}^{\uparrow}}$ *row.*

The carrier has no coordinate for a second selector, unbounded window, quotient stream equality, detached Real endpoint, ambient completeness, or a limit handoff not fed by the displayed source row.

Lean4 (definition) `BEDC.Derived.TailCauchyRouterUp.TailCauchyRouterUp`
Lean4 ✓ `BEDC.Derived.TailCauchyRouterUp.TailCauchyRouterTasteGate single carrier alignment`

Theorem 1051.2 (Tail Cauchy router carrier exposure)**.** *For the carrier of Definition 1051.1, the local* $\mathsf{NameCert}_{\mathsf{TailCauchyRouter}^{\uparrow}}$ *surface exposes exactly the finite rows*

$$S, M, W, R, L, E, H, C, P, N.$$

Every accepted packet makes the selector, shared modulus, finite StreamName and RegSeqRat window, regular-Cauchy source row, CauchyLimitSeal handoff, Real endpoint, transport, continuation, provenance, and local naming rows available as named coordinates.

Proof. Project the displayed carrier. The tuple lists the selector, modulus, window, source, handoff, endpoint, transport, continuation, provenance, and local naming rows as finite coordinates.

The exclusion clause of the carrier forbids any second selector, unbounded window, quotient equality, detached endpoint, ambient completeness row, or unrelated handoff coordinate. Thus the listed coordinates are exactly the exposed NameCert surface. □

Theorem 1051.3 (Tail selector determinacy)**.** *Inside an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet, the tail selector row S determines the window row W relative to the shared modulus M. A consumer cannot replace S by a different selector while keeping the same local NameCert row N and the same displayed route rows C.*

Proof. Read the continuation row C from the selector side. It records the route from S and M to the finite window W, so the window admitted by the certificate is the one reached by the displayed selector and modulus.

If a consumer replaces S while keeping N and C, the first continuation edge no longer has the source named by the carrier. The replacement is therefore outside the accepted NameCert packet rather than a second accepted tail route. □

Theorem 1051.4 (Tail modulus-window coverage)**.** *Every accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet covers the selected tail window by the shared modulus row M. The finite row W must contain the StreamName observations and RegSeqRat readback needed by the regular-Cauchy source row R at the tail named by S.*

Proof. Use the carrier exposure theorem to isolate S, M, W, R. The packet contains a single shared modulus row and a single finite window row, so all source and consumer reads pass through those displayed coordinates.

The continuation row C binds S and M to W, then binds W to the source row R. Hence a window lacking the StreamName observations or RegSeqRat readback required at the selected tail is not covered by the certificate. □

Theorem 1051.5 (Tail seal non-escape)**.** *The* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L of an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet is fed only by the displayed window row W and regular-Cauchy source row R. No accepted packet exports a limit handoff detached from the selected tail route.*

Proof. Inspect the route row C. Its middle segment reaches L from W and R, after W has already been selected by S under the shared modulus M.

The carrier contains no detached handoff coordinate. Therefore any proposed limit seal not fed by the displayed selected window and source row has no coordinate in the local packet and is outside the certificate. □

Theorem 1051.6 (Tail RealUp consumer handoff)**.** *Every consumer route from an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet to a public* $\mathsf{Real}^{\uparrow}$ *endpoint factors through*

$$S, M, W, R, L, E, H, C, P, N.$$

The consumer receives the Real endpoint E only after the selected tail window has fed the displayed $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L.*

Proof. Start with carrier exposure. A consumer can inspect only the displayed selector, modulus, window, source, handoff, endpoint, transport, continuation, provenance, and naming rows.

Follow the displayed continuation route. It first selects and covers the finite tail window, then feeds the CauchyLimitSeal row from that window and the regular-Cauchy source, and only then reaches the Real endpoint.

Since no detached endpoint or ambient completeness coordinate occurs in the carrier, every accepted Real-facing consumer route factors through the displayed finite packet. □

Theorem 1051.7 (Tail Cauchy router RegSeqRat route totality)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read admitted by an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet reaches the consumer only through*

$$S, M, W, R, H, C, P, N.$$

Here W is the finite $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window selected by S under the shared modulus M. The packet has no unbounded window coordinate and no RegSeqRat route outside the displayed finite window.*

Proof. Project the carrier of Definition 1051.1. The RegSeqRat-facing rows available before the limit handoff are the selector S, shared modulus M, finite window W, regular-Cauchy source row R, transport row H, continuation row C, provenance row P, and local naming row N.

Use Definition 1051.2 and Definition 1051.3. They expose the displayed rows and bind W to the selected tail route, so a consumer cannot replace the selected window while keeping the same packet.

Finally apply Definition 1051.4. The RegSeqRat readback required by R is covered inside W, and the carrier contains no unbounded window coordinate. Hence every admitted RegSeqRat read is total only along the displayed finite route. □

Theorem 1051.8 (Tail Cauchy router dyadic budget non-escape)**.** *In an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet, every dyadic tolerance and budget read is confined to the shared modulus M, selected tail S, finite window row W, and route provenance H, C, P, N. No dyadic budget can be imported from a hidden selector, detached window, quotient stream equality, or ambient completeness coordinate.*

Proof. Read the carrier rows for the budget-facing route. The only dyadic control row is the shared modulus M, and it is paired with the selected tail row S before the route reaches the finite window W.

Apply Definition 1051.3. The selector cannot be changed while preserving the same continuation row, so a second dyadic schedule would not be a read of the accepted packet.

Then apply Definition 1051.4. The budget reaches the StreamName and RegSeqRat observations through W, with H, C, P, N recording transport, continuation, provenance, and local naming. The carrier exposure theorem supplies no hidden selector, detached window, quotient equality, or ambient completeness row. □

Theorem 1051.9 (Tail Cauchy router Real seal handoff)**.** *The Real seal endpoint E of an accepted* $\mathsf{TailCauchyRouter}^{\uparrow}$ *packet is public only after the selected window W and source row R feed the displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L. The handoff exports no quotient stream equality, host real equality, hidden choice, or ambient completeness principle.*

Proof. Begin with Definition 1051.5. The limit row L is fed only by the displayed finite window W and regular-Cauchy source row R, after W has been selected from S under the shared modulus M.

Now apply Definition 1051.6. Every Real-facing consumer route factors through the full displayed packet $S, M, W, R, L, E, H, C, P, N$, so the endpoint E is visible only downstream of the selected window and limit handoff.

The carrier exposure theorem supplies no quotient stream equality, host real equality, hidden choice row, or ambient completeness row. Therefore the public seal handoff is exactly the displayed finite route to E. □

Falsifiable prediction: Any accepted TailCauchyRouterUp packet whose selected tail window does not feed the displayed CauchyLimitSeal row is outside the certificate.

Independence witness: streamname, regseqrat, regularcauchysequence, cauchylimitseal, real: TailCauchyRouterUp is not bijective to any sibling because it carries a route row binding all five surfaces at once; the siblings expose point streams, regular sequence rows, Cauchy input data, limit seals, or RealUp endpoints separately.

The Lean site `lean4/BEDC/Derived/TailCauchyRouterUp/TasteGate.lean` provides `Nontrivial TailCauchyRouterUp` and `FieldFaithful TailCauchyRouterUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{TailCauchyRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{TailCauchyRouter}^{\uparrow}$ packet is generated from a tail selector, shared modulus, finite StreamName and RegSeqRat window, regular-Cauchy source row, CauchyLimitSeal handoff, RealUp endpoint, componentwise transports, displayed continuations, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailCauchyRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1051.1, Definition 1051.2, Definition 1051.3, Definition 1051.4, Definition 1051.5, Definition 1051.6, Definition 1051.7, Definition 1051.8, and Definition 1051.9 specify the finite tail-selection route from StreamName and RegSeqRat windows to CauchyLimitSeal and RealUp handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailCauchyRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, hidden choice, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector determinacy, modulus-window coverage, seal non-escape, RealUp consumer handoff, nontriviality, and field faithfulness for the same packet.

Chapter 1052

A Concrete Naming Certificate for $\mathsf{InscriptionTraceClassifier}^{\uparrow}$

The $\mathsf{InscriptionTraceClassifier}^{\uparrow}$ packet realizes the source/name/route/check/consumer surface of the inscription-event vision in chapter 9397. It sits beside the full event packet of chapter 519, the public residue ledger of chapter 520, the trace reader of chapter 682, and the route audit of chapter 1005. These siblings expose event, ledger, trace, and route aspects; the present classifier isolates the finite packet a downstream consumer must display before an inscription event may be exported as public record data.

Definition 1052.1 (Inscription trace-classifier carrier)**.** *An* $\mathsf{InscriptionTraceClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, R, K, U, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row from which the inscription trace is read;*
2. *M is the public name row attached to the trace;*
3. *R is the displayed route row from source to named trace;*
4. *K is the accepted check row that admits the route as public record data;*
5. *U is the downstream consumer row permitted to read the accepted trace;*
6. *L is the refusal-ledger row for incomplete source, name, route, check, or consumer displays;*
7. *H records componentwise* hsame *transports for S, M, R, K, U, L together with continuation, provenance, and local naming rows;*
8. *C records the displayed* Cont *routes by which a consumer reads source, name, route, check, consumer, and refusal-ledger rows;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, $\mathsf{InscriptionAuditTrace}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionTraceClassifier}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten displayed rows. It has no coordinate for a private source, hidden name, unchecked route, off-packet checker, consumer right outside U, or inscription-event export that bypasses the refusal ledger L.

Theorem 1052.2 (Inscription trace-classifier obligations)**.** *For the carrier of Definition 1052.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionTraceClassifier}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$T = (S, M, R, K, U, L, H, C, P, N),$$

source visibility through S*, public naming through* M*, route exactness through* R*, accepted-check admission through* K*, consumer accountability through* U*, refusal-ledger coverage through* L*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Proof. Carrier exposure is projection from the displayed packet. The rows $S, M, R, K, U, L, H, C, P, N$ are the whole inventory available to a trace-classifier reader.

Read the content rows in order. The source row S supplies the visible origin, the name row M supplies the public handle, the route row R supplies the trace path, the check row K admits that path as public record data, and the consumer row U names the permitted downstream reader.

The refusal row L covers every incomplete display among those five rows. The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. Since no carrier coordinate exports a hidden source, hidden name, unchecked route, off-packet checker, or extra consumer right, these rows exhaust the local certificate surface. □

Falsifiable prediction: Any accepted trace-classifier consumer without displayed source, name, route, check, and consumer rows factors to the refusal ledger rather than exporting an inscription event.

Theorem 1052.3 (Inscription trace-classifier falsifiable prediction)**.** *For every accepted* $\mathsf{InscriptionTraceClassifier}^{\uparrow}$ *packet* $T = (S, M, R, K, U, L, H, C, P, N)$*, any purported trace-classifier consumer that omits one of the displayed source, name, route, check, or consumer rows factors through the refusal ledger* L *rather than exporting an* $\mathsf{InscriptionEvent}^{\uparrow}$ *row. Thus an inscription event is public only after* S, M, R, K, U *are visible in the same packet.*

Proof. Project the carrier of Definition 1052.1. The only inscription-facing export path is the displayed chain through S, M, R, K, U.

Apply Definition 1052.2. It assigns missing source, name, route, check, or consumer data to the refusal-ledger row L, while the accepted event-facing path requires the five displayed content rows.

The transport, continuation, provenance, and local naming rows preserve this same split. Therefore a consumer that omits any one of S, M, R, K, U has no event-export coordinate left in the carrier and must be read through L. □

Independence witness: inscriptionevent, inscriptiongapledger, inscriptionaudittrace, inscriptionrouteaudit, inscriptioncommitmentseal: this classifier packet has the simultaneous source/name/route/check/co acceptance chain and a refusal-ledger fallback, while the named siblings respectively expose a full event packet, residue ledger, trace audit, route audit, or commitment seal. No carrier bijection preserves all rows.

Theorem 1052.4 (Inscription trace-classifier independence witness)**.** *The* $\mathsf{InscriptionTraceClassifier}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, with* $\mathsf{InscriptionEvent}^{\uparrow}$*,* $\mathsf{InscriptionGapLedger}^{\uparrow}$*,* $\mathsf{InscriptionAuditTrace}^{\uparrow}$*,* $\mathsf{InscriptionRouteAudit}^{\uparrow}$*, or* $\mathsf{InscriptionCommitmentSeal}^{\uparrow}$*. It uniquely binds the displayed source, name, route, accepted check, downstream consumer, and refusal-ledger rows as one classifier packet.*

Proof. Compare the displayed coordinates. The present carrier contains the accepted classifier chain

$$S \to M \to R \to K \to U$$

together with the refusal-ledger row L for incomplete presentations.

The inscription event sibling contains the full event packet rather than the classifier fallback. The gap ledger sibling records public residue, but not the simultaneous accepted source/name/route/check/consume chain. The audit trace sibling records trace readback, and the route-audit sibling isolates the accepted route to a consumer, but neither supplies this packet's refusal fallback for missing classifier rows.

The commitment seal sibling records an accepted commitment and ledger rather than the full trace-classifier surface. Hence no bijection preserving the local NameCert rows can identify any of those carriers with $T = (S, M, R, K, U, L, H, C, P, N)$. □

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionTraceClassifier}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionTraceClassifier}^{\uparrow}$ packet is generated from visible source, public name, displayed route, accepted check, downstream consumer, refusal-ledger fallback, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionTraceClassifier}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1052.1, Definition 1052.2, Definition 1052.3, and Definition 1052.4 bind the finite source/name/route/check/con classifier packet and its refusal-ledger fallback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionTraceClassifier}^{\uparrow}}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.InscriptionTraceClassifierUp.taste gate`

Lean target: `BEDC.Derived.InscriptionTraceClassifierUp.TasteGate.InscriptionTraceClassifierTasteGate\_single\_car`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridge checking, private agency, host process traces, unchecked routes, inscription-event completeness, or consumer rights outside the displayed packet.

Upgrade path: Scoped closure requires binding the classifier packet to each downstream inscription-event consumer route without adding any row beyond $S, M, R, K, U, L, H, C, P, N$.

Chapter 1053

A Concrete Naming Certificate for $\mathsf{RegSeqObservationBudget}^{\uparrow}$

The $\mathsf{RegSeqObservationBudget}^{\uparrow}$ packet realizes the finite observation-budget route required by the Real-completeness target of chapter 9238. It consumes the diagonal selector surface of chapter 853, the terminal Real seal surface of chapter 38, the finite StreamName window row, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the primitive hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. The packet is an observation route to a seal, not an ambient completeness theorem, quotient stream equality, host real equality, or selected external limit.

It is composite over the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ surfaces, binding their displayed rows into a single observation-budget route.

Falsifiable prediction: Every RealUp-seal consumer accepted by $\mathsf{RegSeqObservationBudget}^{\uparrow}$ factors through the displayed observation-budget row before any seal read.

Independence witness: cauchydiagonalbudget, realobservationbudget, regularcauchycomparison, streamname, real: no listed sibling is bijective to $\mathsf{RegSeqObservationBudget}^{\uparrow}$. The diagonal budget stores selector data without the full RealUp observation row, the Real observation budget stores seal-facing budget admission without the regular-source and diagonal selector packet, the regular-Cauchy comparison row stores finite comparison data without the terminal seal route, the StreamName row stores finite windows without the RealUp budget, and the RealUp certificate stores the seal without the upstream observation schedule.

Definition 1053.1 (Regular observer-budget carrier)**.** *A* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, R, E, D, \epsilon, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that exposes the requested observation window;*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window;*
3. E *is the* $\mathsf{Real}^{\uparrow}$ *observation-budget row reached only after the regular-source readback;*
4. D *is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *row selecting the diagonal observation budget consumed by the same read;*
5. ϵ *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that fixes the finite precision served by the packet;*

6. *H records componentwise* hsame *transports for the window, readback, RealUp budget, diagonal selector, tolerance, continuation, provenance, and local naming rows;*
7. *C records the displayed* Cont *replay route from W to R, from R and ϵ to D, and from D to E;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$ *row.*

The classifier accepts only the displayed finite route $W, R, E, D, \epsilon, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, an unbounded observation schedule, a hidden diagonal selector, an undisplayed seal read, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegSeqObservationBudgetUp.RegSeqObservationBudgetTasteGate single carrier alignment`

Theorem 1053.2 (Regular observer-budget NameCert obligations)**.** *For the carrier of Definition 1053.1, the local* $\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$O = (W, R, E, D, \epsilon, H, C, P, N),$$

StreamName window exposure through W, RegSeqRat readback through R, RealUp observation-budget exposure through E, diagonal selector exposure through D, dyadic tolerance exposure through ϵ, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the displayed carrier. Its public rows are precisely the finite window, regular-source readback, RealUp observation budget, diagonal selector, dyadic tolerance, transport, continuation, provenance, and local naming rows.

Read the route in dependency order. The StreamName window W is available before regular readback R; the tolerance row ϵ and diagonal budget D then determine the finite observation request; the RealUp budget row E is reached only after those rows are displayed.

The rows H, C, P, N preserve and name the same packet. Since the carrier lacks coordinates for quotient stream equality, host real equality, unbounded observation schedules, hidden selectors, detached seals, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 1053.3 (Regular observer-budget filter-schedule non-escape)**.** *Every accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *consumer route to a* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed observation-budget sequence*

$$W, R, \epsilon, D, E, H, C, P, N.$$

No accepted route may inspect the terminal RealUp seal before the StreamName window, RegSeqRat readback, dyadic tolerance, and CauchyDiagonalBudget selector rows have been read, and no route may replace that finite schedule by an omitted filter row, hidden diagonal selector, quotient stream, host real equality, or ambient completeness principle.

Proof. Start from the NameCert obligation theorem. It identifies every accepted packet with the displayed rows $W, R, E, D, \epsilon, H, C, P, N$ and gives the local classifier no additional schedule coordinate.

The positive route is fixed by the carrier itself. The window row W feeds the readback row R, the tolerance row ϵ and diagonal selector D fix the finite observation budget, and only then is the RealUp observation-budget row E available for a seal-facing consumer.

The structural rows preserve that order. Componentwise hsame transport in H and replay in C move the displayed route, while P and N record provenance and local naming for the same packet. Therefore any omitted filter row, hidden selector, quotient, host equality, or ambient completeness object remains outside the accepted route. □

Theorem 1053.4 (Regular observer-budget window admission)**.** *Every accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *route exposes the finite window row W before it may read the regular-source row R, the dyadic tolerance row ϵ, the diagonal selector row D, or the RealUp observation-budget row E. In particular, no accepted consumer may enter through R, ϵ, D, or E while leaving the StreamName window undisplayed.*

Proof. Use the carrier of Definition 1053.1. Its accepted packet is the displayed row

$$O = (W, R, E, D, \epsilon, H, C, P, N),$$

and the classifier has no coordinate for an alternate entry point.

Apply Definition 1053.2. The local NameCert surface lists StreamName window exposure through W before RegSeqRat readback, dyadic tolerance exposure, diagonal selector exposure, and RealUp budget exposure.

Finally apply Definition 1053.3. Any consumer route to the seal must factor through the finite schedule in which W is already visible before R, ϵ, D, and E are consumed. Hence a route that enters through those later rows without exposing W is outside the accepted packet. □

Theorem 1053.5 (Regular observer-budget tail monotonicity)**.** *Componentwise* hsame *transport and displayed* Cont *replay preserve the same finite observation budget*

$$W, R, \epsilon, D, E, H, C, P, N$$

for an accepted $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *route. They may transport or replay the displayed rows, but they cannot introduce a hidden selector, quotient stream, host real equality, unbounded observation schedule, or second terminal seal read.*

Proof. Project the structural rows from Definition 1053.1. The row H records only componentwise hsame transports for the displayed window, readback, RealUp budget, diagonal selector, tolerance, continuation, provenance, and local naming rows, while C records only the displayed replay route from W to R, from R and ϵ to D, and from D to E.

Apply Definition 1053.2. The accepted NameCert surface is exhausted by the same finite carrier rows, so transport and replay have no independent coordinate in which a new selector, quotient, host equality, unbounded schedule, or extra seal read could be stored.

Apply Definition 1053.3. Since every accepted seal-facing route factors through the finite observation schedule, extending along a transported or replayed tail preserves that schedule and cannot escape to an undisplayed observation object. □

Theorem 1053.6 (Regular observer-budget Real seal handoff)**.** *The RealUp seal-facing row E of an accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *carrier is consumed only after the StreamName window W, RegSeqRat readback R, dyadic tolerance ϵ, and CauchyDiagonalBudget selector D have been displayed. Thus the terminal seal handoff is a consequence of the finite observation-budget route, not an independent Real equality, quotient stream, or ambient completeness premise.*

Proof. Start with the carrier in Definition 1053.1. The row E is a displayed RealUp observation-budget row inside the same finite packet as W, R, ϵ, and D.

Apply Definition 1053.2. The obligation surface reaches RealUp budget exposure through E only as part of the listed window exposure, regular readback, dyadic tolerance, diagonal selector, transport, replay, provenance, and local naming rows.

Now apply Definition 1053.3. It forbids inspecting the terminal RealUp seal before W, R, ϵ, and D have been read. Therefore any seal-facing use of E is a handoff from the finite observation-budget route, and not from an independent equality, quotient, or completeness principle. □

The Lean TasteGate boundary for this packet must provide `Nontrivial RegSeqObservationBudgetUp` and `FieldFaithful RegSeqObservationBudgetUp` instances in `Derived/RegSeqObservationBudgetUp/TasteGa` with the canonical obligation theorem expected at `BEDC.Derived.RegSeqObservationBudgetUp.RegSeqObservat`

Theorem 1053.7 (Regular observer-budget scoped closure)**.** *The carrier definition, NameCert obligations, filter-schedule non-escape, window admission, tail monotonicity, Real seal handoff, and the two Lean TasteGate witnesses bind the scoped* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *surface exactly over the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Inside this scope, every accepted seal-facing consumer is a consumer of the finite observation-budget packet*

$$W, R, \epsilon, D, E, H, C, P, N.$$

The scope has no classifier coordinate for quotient streams, host real equality, hidden selectors, unbounded schedules, or ambient completeness.

Proof. Apply the carrier description of Definition 1053.1 and the obligation surface of Definition 1053.2. These identify the accepted object with the listed finite rows and leave no independent row beyond the displayed window, readback, Real budget, diagonal selector, tolerance, transport, continuation, provenance, and local naming data.

Next apply Definition 1053.3, Definition 1053.4, and Definition 1053.5. The window is visible before later rows, every seal-facing route factors through the finite schedule, and transport or replay preserves the same observation budget rather than creating a new selector or schedule.

Finally apply Definition 1053.6. The RealUp seal row is reached only as the terminal handoff of the displayed route. The Lean TasteGate names `RegSeqObservationBudgetTasteGate_single_carrier_alignme` and `taste_gate` witness that the same singleton carrier boundary supplies nontriviality and field faithfulness, so the paper scope and kernel TasteGate evidence refer to the same finite packet. □

Theorem 1053.8 (Regular observer-budget public consumer export)**.** *Every public* $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalLimitStability}^{\uparrow}$, *or* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *packet receives exactly the displayed finite route*

$$W, R, \epsilon, D, E, H, C, P, N.$$

The exported route reads the $\mathsf{StreamName}^{\uparrow}$ *window* W *before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R, *reads the dyadic tolerance row* ϵ *and diagonal selector row* D *before* $\mathsf{Real}^{\uparrow}$ *budget exposure* E, *and then replays only the displayed transport, continuation, provenance, and local naming rows* H, C, P, N. *The export contains no quotient stream equality, host real equality, hidden selector, unbounded schedule, detached seal read, or ambient completeness coordinate.*

Proof. First expose the public entry row. Theorem 1053.4 forces every accepted route to display the StreamName window W before it may read R, ϵ, D, or E.

Next preserve the finite tail budget. Theorem 1053.5 keeps componentwise hsame transport and displayed Cont replay on the same rows $W, R, \epsilon, D, E, H, C, P, N$, so public $\mathsf{DiagonalLimitStability}^{\uparrow}$ and $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ consumers cannot replace the selector or schedule by an unshown coordinate.

Then apply the Real seal handoff. Theorem 1053.6 reaches the RealUp budget row E only after the window, readback, dyadic tolerance, and diagonal selector have been displayed. Thus a public $\mathsf{Real}^{\uparrow}$ consumer receives the same finite route rather than a detached seal read.

Finally use Definition 1053.7. The scoped closure binds the carrier, NameCert obligations, non-escape theorem, window admission, tail monotonicity, and Real seal handoff to the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. The accepted carrier has no coordinate for quotient stream equality, host real equality, hidden selectors, unbounded schedules, detached seal reads, or ambient completeness, so the displayed route is the whole public export surface. □

Theorem 1053.9 (Regular observer-budget bridge precondition)**.** *Any standard bridge eligible to consume the public* $\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$ *export must preserve the displayed finite route*

$$W, R, \epsilon, D, E, H, C, P, N$$

through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Bridge-side data may be exposed only after the window* W*, readback* R*, tolerance* ϵ*, diagonal selector* D*, and RealUp budget row* E *have appeared in that order, followed by the displayed transport, continuation, provenance, and local naming rows* H, C, P, N*. No quotient stream equality, host real equality, hidden selector, unbounded schedule, detached seal read, or ambient completeness premise can serve as an alternate bridge entry.*

Lean4 ✓ `BEDC.Derived.RegSeqObservationBudgetUp.RegSeqObservationBudgetUp StdBridge`

Proof. Start with the public export of Definition 1053.8. It already identifies the consumer-visible boundary with the single finite route $W, R, \epsilon, D, E, H, C, P, N$ and excludes quotient stream equality, host real equality, hidden selectors, unbounded schedules, detached seal reads, and ambient completeness coordinates from that boundary.

Next use Definition 1053.7. The carrier, NameCert obligations, non-escape theorem, window admission, tail monotonicity, Real seal handoff, and TasteGate witnesses are all scoped over the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. Thus a standard bridge consuming the public export cannot widen the object language beyond those rows.

Finally apply Definition 1053.4, Definition 1053.5, and Definition 1053.6. Window admission orders W before later rows, tail monotonicity keeps transport and replay on the same displayed budget, and Real seal handoff reaches E only after W, R, ϵ, and D. Therefore the only admissible bridge entry is the displayed route in the stated order. □

Theorem 1053.10 (Regular observer-budget mature package)**.** *The public export theorem, bridge precondition, scoped closure theorem, window admission, tail monotonicity, Real seal handoff, and Lean TasteGate witnesses package the finite route*

$$W, R, \epsilon, D, E, H, C, P, N$$

as the mature paper-side $\mathsf{RegSeqObservationBudget}^{\uparrow}$ *surface. The package binds only the displayed StreamName window, RegSeqRat readback, dyadic tolerance, CauchyDiagonalBudget selector, RealUp budget row, componentwise transports, displayed continuations, package provenance, and local*

NameCert row. It has no coordinate for quotient stream equality, host real equality, hidden selectors, unbounded schedules, detached seal reads, or ambient completeness.

Proof. First apply Definition 1053.8. The public consumer boundary is already the finite route $W, R, \epsilon, D, E, H, C, P, N$, so public RealUp, DiagonalLimitStabilityUp, and CauchyDiagonalBudgetUp consumers receive the same displayed observation budget.

Next apply Definition 1053.9 and Definition 1053.7. The bridge precondition restricts any standard bridge entry to the same ordered rows, while scoped closure binds those rows over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Finally apply Definition 1053.4, Definition 1053.5, and Definition 1053.6. Window admission exposes W before later rows, tail monotonicity keeps transport and replay on the same finite packet, and the Real seal handoff reaches E only through W, R, ϵ, and D. Together with the Lean TasteGate witnesses for nontriviality and field faithfulness, these clauses leave no entry for quotient stream equality, host real equality, hidden selectors, unbounded schedules, detached seal reads, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RegSeqObservationBudget}^{\uparrow}$ packet is generated from a StreamName finite-window row, RegSeqRat readback row, RealUp observation-budget row, CauchyDiagonalBudget selector row, DyadicRatCore tolerance row, componentwise transports, displayed continuations, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1053.1, Definition 1053.2, Definition 1053.3, Definition 1053.4, Definition 1053.5, Definition 1053.6, Definition 1053.8, Definition 1053.9, Definition 1053.10,
Lean4 ✓ `BEDC.Derived.RegSeqObservationBudgetUp.RegSeqObservationBudgetTasteGate single carrier alignment`
, and
Lean4 ✓ `BEDC.Derived.RegSeqObservationBudgetUp.taste gate`
bind the finite RegSeq observation-budget route over StreamNameUp, RegSeqRatUp, RealUp, CauchyDiagonalBudgetUp, DyadicRatCoreUp, hsame, Cont, Pkg, and NameCert rows and expose the public consumer export surface with its bridge-entry precondition and mature paper-side package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegSeqObservationBudget}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.RegSeqObservationBudgetUp.RegSeqObservationBudgetUp\_StdBridge`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, selected external limits, hidden selectors, ambient completeness, or bridge checking.

Upgrade path: No further formal-axis upgrade remains because the standard bridge is checked for the same finite observation-budget packet.

Chapter 1054

A Concrete Naming Certificate for $\mathsf{ObserverTraceSeal}^{\uparrow}$

The $\mathsf{ObserverTraceSeal}^{\uparrow}$ packet realizes the observer-history identity vision of chapter 9418 as a finite sealed trace certificate. It reads the no-subject identity surface of chapter 479 together with the observer trace, distinction trace, bounded trace, and finite evaluator siblings. The finite evaluator sibling of chapter 818 supplies bounded checkpoint and runtime readback, while $\mathsf{ObserverTraceSeal}^{\uparrow}$ adds the seal and no-subject refusal row. Its novelty is the seal row: a visible refusal ledger that binds each accepted trace to the same observer row, observation row, route, provenance, and local name before any consumer may read the trace.

Definition 1054.1 (ObserverTraceSeal carrier)**.** *An* $\mathsf{ObserverTraceSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, T, A, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the observer row, read as a finite* Hist *or* BHist *observer-history surface;*
2. *T is the accepted trace row over the same observer-history surface;*
3. *A is the observation row, recording the displayed* Cont*-mediated read of the trace;*
4. *R is the seal and no-subject refusal ledger, recording that the trace has no owner, subject carrier, hidden observer equality, or off-packet identity row;*
5. *H records componentwise* hsame *transport for O, T, A, R, P, N;*
6. *C records the displayed* Cont *routes from the observer row through the trace, observation, seal, provenance, and naming rows;*
7. *P is the* Pkg *provenance over* $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverKernelTrace}^{\uparrow}$, $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{FiniteTraceEvaluator}^{\uparrow}$, BHist, Hist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverTraceSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for observer-substance, host equality of observers, quotient observer identity, detached trace ownership, or an evaluator graph outside the displayed trace packet.

Theorem 1054.2 (ObserverTraceSeal NameCert obligations)**.** *For the carrier of Definition 1054.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverTraceSeal}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for $S = (O, T, A, R, H, C, P, N)$;*

2. *observer and trace exposure through the displayed O, T, A rows;*
3. *seal and no-subject refusal through R;*
4. *componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance through H, C, P;*
5. *local naming exactness through N.*

No accepted consumer can read the trace without passing through all five rows.

Lean4 ✓ `BEDC.Derived.ObserverTraceSealUp.ObserverTraceSealUp taste gate boundary`

Proof. Carrier admission is projection from Definition 1054.1. The packet displays the observer row, trace row, observation row, seal row, transport, route, provenance, and local name.

The observer and trace exposure rows are read through O, T, A. The trace is therefore attached to the same observer-history surface and the same displayed observation route before the seal is inspected.

The refusal row R records the no-subject boundary. Componentwise transport and replay are then supplied only by H and C, while P and N record provenance and local naming. Since no further coordinate is present, the five rows exhaust the local certificate. □

Theorem 1054.3 (ObserverTraceSeal no-subject nonescape)**.** *Every accepted* ObserverTraceSeal$^{\uparrow}$ *packet exposes a trace only through the displayed observer row, trace row, observation row, seal row, componentwise* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row O, T, A, R, H, C, P, N. It cannot export an observer-substance carrier, hidden subject row, host equality of observers, quotient observer identity, detached trace owner, or evaluator graph outside those rows.*

Proof. Begin with the NameCert obligations of Definition 1054.2. They force every accepted read through O, T, A, R, H, C, P, N before the local name is available.

Use the observer-history identity sibling chapter 479. It reads observer identity as a finite history and observation surface rather than as a separate subject carrier.

Now attach the seal row R. The refusal ledger is part of the same accepted packet as the trace, so a consumer cannot expose T while omitting the no-subject refusal. The absent observer-substance, hidden subject, host equality, quotient identity, detached owner, and evaluator graph each require a coordinate outside the carrier. □

Falsifiable prediction: If an accepted ObserverTraceSeal packet exposes a trace without the no-subject refusal row, then thm:observer-trace-seal-non-subject-nonescape is refuted by that packet.

Independence witness: observerhistidentity, observerkerneltrace, observerdistinctiontrace, boundeduniversaltrace, finitetraceevaluator: the packet records an observer row, trace row, observation row, seal row, no-subject refusal, transport, route, provenance, and local name in one finite carrier.

The seal row plus the no-subject refusal ledger prevents a bijection to any sibling carrier. ObserverHistIdentity$^{\uparrow}$ supplies identity comparison without a sealed trace row, ObserverKernelTrace$^{\uparrow}$ supplies trace exposure without the extra seal obligation, ObserverDistinctionTrace$^{\uparrow}$ orders distinction steps without bounded trace readback, BoundedUniversalTrace$^{\uparrow}$ supplies a fuel-indexed trace without observer-history identity, and FiniteTraceEvaluator$^{\uparrow}$ supplies finite evaluation without the no-subject seal.

Theorem 1054.4 (ObserverTraceSeal public export)**.** *Every public export of an accepted* ObserverTraceSeal$^{\uparrow}$ *packet is exactly the displayed observer row, trace row, observation row, no-subject seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row*

$$O, T, A, R, H, C, P, N.$$

The export contains no observer-substance carrier, detached trace owner, host equality of observers, quotient observer identity, hidden subject row, or evaluator graph outside the sealed packet.

Proof. Project the carrier of Definition 1054.1. Its rows are exactly the observer row O, trace row T, observation row A, seal and no-subject refusal row R, transport H, replay C, provenance P, and local naming row N.

Apply the NameCert obligations of Definition 1054.2. They force every accepted public read through carrier admission, observer and trace exposure, no-subject refusal, transport, replay, provenance, and local naming before the trace can be exported.

Finally use Definition 1054.3. The absent observer-substance, detached owner, host equality, quotient identity, hidden subject, and external evaluator graph have no coordinate in the carrier, so the public export is exactly the displayed finite surface. □

Theorem 1054.5 (ObserverTraceSeal mature route)**.** *Every accepted* $\mathsf{ObserverTraceSeal}^{\uparrow}$ *public route is exhausted by the observer row, trace row, observation row, no-subject seal row, componentwise transport, replay, provenance, and local naming rows already used by the observer, trace, seal, and finite evaluator siblings. No further theory-axis datum is needed to reuse the sealed trace packet as a public consumer surface.*

Proof. Start with the public export theorem Definition 1054.4. It identifies the exported rows with O, T, A, R, H, C, P, N and excludes detached observer ownership, host equality, quotient identity, hidden subject data, and external evaluator graphs.

The observer and trace siblings consume the same rows. $\mathsf{ObserverHistIdentity}^{\uparrow}$ supplies the no-subject identity reading, $\mathsf{ObserverKernelTrace}^{\uparrow}$ and $\mathsf{ObserverDistinctionTrace}^{\uparrow}$ supply trace-facing reads, and $\mathsf{BoundedUniversalTrace}^{\uparrow}$ supplies the bounded trace boundary.

Finally use the finite evaluator sibling of chapter 818. Its checkpoint and runtime readback remain downstream of the same displayed trace packet, so the seal row R and the structural rows H, C, P, N already supply the mature public route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverTraceSeal}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObserverTraceSeal}^{\uparrow}$ packet is generated from an observer row, trace row, observation row, seal and no-subject refusal ledger, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverTraceSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1054.1, Definition 1054.2, Definition 1054.3, and Definition 1054.4 specify the finite sealed trace packet, its no-subject boundary, and its public export surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverTraceSeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean4 ✓ `BEDC.Derived.ObserverTraceSealUp.taste gate`

Lean target: `BEDC.Derived.ObserverTraceSealUp.ObserverTraceSealUp\_taste\_gate\_boundary`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, a host observer model, quotient observer identity, detached trace ownership, or evaluator graph semantics.

Upgrade path: No further paper-side theory upgrade is pending for the displayed observer-trace seal route.

Chapter 1055

A Concrete Naming Certificate for $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$

The $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ surface turns the compiler-correctness reading of chapter 9371 into an atomic finite packet. It consumes the morphism rows of chapter 481, the compiler graph rows of chapter 473, and the kernel morphism boundary of chapter 620, but it adds a displayed source-to-target replay route for trace faithfulness. The packet does not assert host compiler semantics, program equivalence, a decompiler inverse, or ambient semantic preservation outside the displayed certificate rows.

Falsifiable prediction: Every accepted compiler trace faithfulness packet preserves source hsame as target hsame through a displayed finite graph replay route; a packet missing the replay route is rejected.

Independence witness: namecertmorphism, certificatecompiler, kernelmorphism, haltingtrace-classifier: none is bijective because CompilerTraceFaithfulnessUp carries both compiler graph replay and source-target hsame preservation, while the siblings lack one of those rows.

Definition 1055.1 (CompilerTraceFaithfulness carrier)**.** *A* $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, M, G, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* NameCert *row whose carrier and classifier provide the accepted source histories;*
2. *T is the target* NameCert *row whose carrier and classifier receive target histories;*
3. *K is the accepted* $\mathsf{CertificateCompiler}^{\uparrow}$ *packet reading the finite compiler graph from S to T;*
4. *M is the accepted* KernelMorphismUp *row exposing the same source, target, graph, carrier-admission, and classifier-lift surface;*
5. *G is the finite compiler graph row shared by K and M;*
6. *R is the displayed replay route assigning each accepted source trace window a finite graph path in G;*
7. *L is the classifier and* hsame *lift row stating that source classifier agreement and source* hsame *are transported to target classifier agreement and target* hsame *along the replay route;*

8. *H records componentwise* hsame *transport for the source, target, compiler packet, kernel morphism packet, graph, replay route, lift row, continuation row, package row, and local naming row;*
9. *C records the displayed* Cont *replay by which consumers read source windows, graph edges, target windows, and transported classifier rows;*
10. *P names the dependency surface* BHist, hsame, Cont, Pkg, NameCert, NameCertMorphism$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, *and* KernelMorphismUp*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompilerTraceFaithfulness}^{\uparrow}}$ *row.*

The local classifier compares two accepted packets only through the displayed source and target certificate rows, the shared compiler graph, the replay route, the classifier and hsame *lift row, transport, continuation replay, package provenance, and local naming.*

Lean4 (definition) `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessCarrier`

Definition 1055.2 (CompilerTraceFaithfulness classifier)**.** *The classifier of a* CompilerTraceFaithfulness$^{\uparrow}$ *packet accepts a source-target trace comparison exactly when it reads a source trace window from S, follows the displayed replay route R through graph edges of G, lands in the target row T through the carrier rows already exposed by K and M, and then transports classifier membership and* hsame *through L, H, C.*

Thus trace faithfulness is not a judgment about arbitrary source and target behaviors. It is a finite relation on displayed replay paths whose endpoints are already certified by the source and target NameCert *rows.*

Lean4 (definition) `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessClassifier`
Lean4 ✓ `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessClassifier replay transport`

Theorem 1055.3 (CompilerTraceFaithfulness NameCert obligations)**.** *For the carrier and classifier of Definition 1055.1 and Definition 1055.2, the local* $\mathsf{NameCert}_{\mathsf{CompilerTraceFaithfulness}^{\uparrow}}$ *surface has five obligations:*

1. *carrier exposure: the accepted object is the finite packet $F = (S, T, K, M, G, R, L, H, C, P, N)$;*
2. *compiler graph admission: every replayed edge is a displayed edge of G whose endpoints are admitted by the compiler packet K and the kernel morphism packet M;*
3. *replay totality: every accepted source trace window consumed by the classifier has a finite route in R ending in the displayed target row;*
4. *classifier and* hsame *preservation: source classifier membership and source* hsame *are transported to the target classifier and target* hsame *only through L, H, C;*
5. *non-escape: no host compiler semantics, quotient program equality, decompiler inverse, selected behavior, or target object outside $S, T, K, M, G, R, L, H, C, P, N$ is exported.*

Proof. Carrier exposure is projection from the finite tuple. The rows $S, T, K, M, G, R, L, H, C, P, N$ are the only coordinates available to the local classifier and to downstream consumers.

Compiler graph admission is inherited from the two sibling routes. The CertificateCompiler$^{\uparrow}$ packet of chapter 473 supplies finite graph edges and endpoint landing, while the KernelMorphismUp packet of chapter 620 exposes the same edge and classifier-lift surface as a kernel morphism boundary.

Replay totality is the additional row R. A source trace window is accepted only when R displays a finite path through G and reaches the target row already certified by T.

Finally the preservation and non-escape obligations are read from L, H, C, P, N. The lift row records classifier and hsame transport, the continuation and package rows replay and name the same route, and the absence of any other coordinate rules out host compiler semantics, quotient equality, inverse data, selected behaviors, and undisplayed target objects. □

Lean4 ✓ `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessNameCertObligations`

Theorem 1055.4 (CompilerTraceFaithfulness finite replay boundary)**.** *Every accepted* CompilerTraceFaithfulness$^{\uparrow}$ *packet preserves source* hsame *as target* hsame *only through the displayed finite graph replay route*

$$S \to G \to R \to T$$

inside $F = (S, T, K, M, G, R, L, H, C, P, N)$*. The consumer must first read the source certificate, the accepted compiler packet, the kernel morphism row, the graph row, the replay route, and the classifier and* hsame *lift row before it may export target agreement. A packet without the replay route* R *is not accepted as compiler trace faithfulness.*

Proof. Begin with the carrier and classifier definitions. They place the source row, target row, compiler packet, kernel morphism row, graph row, replay route, and lift row inside one finite packet.

Use the NameCert obligations. Graph admission forces every replay edge through G, K, M, while replay totality forces every accepted source trace window through R before a target row can be read.

The preservation row L then transports source classifier membership and source hsame to the target side, with H and C preserving the same route componentwise. Package provenance and local naming close the packet without adding another source of agreement.

If R is absent, the replay-totality obligation has no coordinate to read. The packet may still resemble a compiler graph or kernel morphism packet, but it cannot certify trace faithfulness because the source-to-target route has not been displayed. □

Lean4 ✓ `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessFiniteReplayBoundary`

Theorem 1055.5 (CompilerTraceFaithfulness downstream target)**.** *The Lean-facing downstream surface for* CompilerTraceFaithfulness$^{\uparrow}$ *is the packet family to be recorded under* Derived/CompilerTraceFaithfulnessUp *with explicit* Nontrivial *and* FieldFaithful *instances. The target checks that the finite replay route* R*, the classifier and* hsame *lift row* L*, and the support rows* H, C, P, N *are all present before a compiler trace faithfulness packet is admitted.*

Proof. The carrier and obligations already name the data that the downstream gate must inspect. A nontrivial instance requires an accepted packet with a displayed replay route, and a field-faithful instance requires each visible row of F to remain recoverable from the admitted packet.

The finite replay boundary supplies the consumer condition. Admission can be tested by projecting R, L, H, C, P, N from the packet and checking that the target-side agreement is read only after the source route has passed through the displayed compiler graph.

No extra semantic object is part of this target. The file path names the later Lean surface, while the paper packet fixes the row inventory that such a gate must preserve. □

Lean4 ✓ `BEDC.Derived.CompilerTraceFaithfulnessUp.CompilerTraceFaithfulnessTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CompilerTraceFaithfulness}\uparrow}$.

Constructive story: A $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ packet is generated from displayed source and target certificate rows, a compiler graph, a replay route, classifier and hsame lift data, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompilerTraceFaithfulness}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1055.1, Definition 1055.2, Definition 1055.3, Definition 1055.4, and Definition 1055.5 bind the mature finite replay treatment across source certificate, compiler graph, kernel morphism, replay, lift, transport, continuation, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompilerTraceFaithfulness}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host compiler semantics, quotient program equality, a decompiler inverse, ambient semantic preservation, Lean verification, or bridge checking.

Upgrade path: No paper-side theory upgrade remains; future work is Lean verification and bridge checking for the same finite replay packet.

Chapter 1056

A Concrete Naming Certificate for $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$

The $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ packet realizes the critical path of chapter 9385 as a finite discharge-obstruction lattice. It sits beside the audit-synthesis packet of **??**, the discharge-obligation packet of chapter 724, the subject-reduction obstruction packet of chapter 731, the decidable audit packet of chapter 746, and the confluence audit packet of chapter 735. Those siblings expose available, blocked, and required MetaCIC audit rows; the present packet records their finite lattice order without turning the blocked subject-reduction edge into a closed consistency theorem.

Definition 1056.1 (MetaCIC discharge-obstruction lattice carrier)**.** *A* $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (N, D, C, O, Q, T, R, P, K)$$

with the following displayed rows:

1. *N is the closed-normal consistency row, reading the available normal endpoint through* **??***;*
2. *D is the decidable checker row, reading local typing and bounded-normal comparison through chapter 746;*
3. *C is the confluence-status row, reading conditional bridges and closed-normal fragments through chapter 735;*
4. *O is the subject-reduction obstruction row, reading the checked blocked edge through chapter 731;*
5. *Q is the discharge-obligation row, reading the four required subject-reduction inputs through chapter 724;*
6. *T records componentwise* hsame *transports for the closed-normal, checker, confluence, obstruction, obligation, package, and naming rows;*
7. *R records displayed* Cont *routes by which a consumer reads available rows before blocked rows and required rows before any attempted closed-consistency export;*
8. *P is the* Pkg *provenance over the MetaCIC audit synthesis, discharge-obligation, subject-reduction obstruction, decidable checker, confluence audit,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *K is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}}$ *row.*

The carrier contains no coordinate for unconditional closed consistency, no coordinate erasing the discharge-obligation row, and no coordinate replacing the obstruction row by a closed subject-reduction theorem.

Lean4 (definition) `BEDC.Derived.MetaCICDischargeObstructionLatticeUp.MetaCICDischargeObstructionLatticeUp`

Theorem 1056.2 (MetaCIC discharge-obstruction lattice NameCert obligations)**.** *For the carrier of Definition 1056.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L = (N, D, C, O, Q, T, R, P, K)$*, closed-normal consistency exposure, decidable-checker exposure, confluence-status exposure, subject-reduction obstruction exposure, discharge-obligation exposure, componentwise* hsame *transport, displayed* Cont *route replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The rows $N, D, C, O, Q, T, R, P, K$ are the complete finite packet, so carrier admission and local naming are direct projections.

Read the available branch through N, D, C. These rows expose closed-normal consistency, local checker evidence, and confluence status as separate sibling reads, matching the available entries of chapter 9385.

Read the blocked and required branch through O and Q. The obstruction row keeps the subject-reduction edge unavailable, while the discharge-obligation row lists the required inputs that a later consumer must supply before that edge can be used.

Finally read T, R, P, K. Transport, route replay, provenance, and local naming preserve the same row order, and the carrier has no additional coordinate from which a hidden closed-consistency theorem could be exported. □

Theorem 1056.3 (MetaCIC discharge-obstruction lattice falsifiable nonescape)**.** *Any accepted consumer of a* $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ *carrier cannot export unconditional closed consistency unless the discharge-obligation row* Q *is filled and the obstruction row* O *is discharged. A consumer that reads only* N, D, C, T, R, P, K*, or reads* O *merely as an obstruction, remains confined to closed-normal consistency, local checker evidence, confluence status, and the recorded blocked edge.*

Proof. Begin with Definition 1056.2. It places the closed-normal, decidable, and confluence rows on the available side, and it keeps the obstruction and obligation rows visible as separate coordinates.

The falsifiable test is row-local. If a claimed consumer exports unconditional closed consistency, its route must show where Q supplies the required subject-reduction inputs and where O ceases to be the blocked subject-reduction edge.

When either row is absent in that role, route replay through R stops at the closed-normal branch and the recorded blocked edge. Provenance through P and naming through K preserve the same boundary, so the claimed export has no accepted carrier coordinate. □

Theorem 1056.4 (MetaCIC discharge-obstruction lattice independence witness)**.** *The siblings* **??***, chapter 724, chapter 731, chapter 746, and chapter 735 are independent witnesses for the* $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ *lattice. None is bijective with the lattice carrier, because each lacks at least one of the available, blocked, or required edge rows displayed by* $N, D, C, O, Q, T, R, P, K$*.*

Proof. Compare the audit-synthesis sibling first. It names the combined critical path, but it does not by itself separate every available, blocked, and required edge row as the lattice carrier does.

Compare the discharge-obligation and subject-reduction obstruction siblings next. The former displays the required inputs, while the latter displays the blocked edge; neither alone carries the closed-normal, decidable, confluence, obstruction, and obligation rows as one ordered packet.

Finally compare the decidable and confluence audit siblings. They provide available checker and confluence rows, but they do not contain the discharge-obligation row and the subject-reduction obstruction row together. Hence none of the five siblings is bijective with L, and together they witness the lattice as a genuine finite assembly rather than a renamed copy of one sibling. □

The Lean-side handoff for this packet is the finite object Nontrivial $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$, witnessed by an explicit pair separating an available-row projection from a blocked-row projection, together with FieldFaithful $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ by constructor-field projection and injection over $N, D, C, O, Q, T, R, P, K$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}$ packet is generated from closed-normal consistency, decidable-checker, confluence-status, subject-reduction obstruction, discharge-obligation, transport, route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1056.1, Definition 1056.2, Definition 1056.3, and Definition 1056.4 bind the MetaCIC audit critical path as a finite discharge-obstruction lattice.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeObstructionLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove closed subject reduction, unconditional closed consistency, arbitrary closed Church–Rosser, global decidability, checked Lean certificate coverage, or a public standard bridge.

Upgrade path: Formal closure requires checked carrier admission, field-faithful projections, nontrivial available-blocked witnesses, route replay, provenance, and non-escape for the same finite lattice rows.

Chapter 1057

A Concrete Naming Certificate for $\mathsf{AxiomRefusalLedger}^{\uparrow}$

The $\mathsf{AxiomRefusalLedger}^{\uparrow}$ packet makes an axiom-use attempt visible as a finite NameCert refusal row. It sits under the apophatic-name discipline of chapter 9400 and consumes the named externality surface of chapter 543 together with the socket surface of chapter 536. Its role is narrower than those siblings: it records that an attempted kernel-facing assumption is refused as an axiom and redirected either to a displayed setup-field obligation or to an audit-visible unmet boundary.

It is an ai over the apophatic name, socket, gate question, externality gate, and type-level socket exposure surfaces, with the refusal ledger row serving as the local audit interface.

Definition 1057.1 (AxiomRefusalLedger carrier)**.** *An* $\mathsf{AxiomRefusalLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, S, Q, F, R, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the accepted* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the unsupported dependency position;*
2. *S is the accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *row recording the requested supply shape and its visible site;*
3. *Q is the axiom-use question, asking whether the request would be consumed as a Lean axiom, hidden theorem, host assumption, or unrecorded primitive;*
4. *F is the setup-field alternative row, recording the exposed carrier field or implementation obligation that would have to supply the requested data without expanding the substrate;*
5. *R is the refusal ledger for the attempted axiom channel;*
6. *G is the audit-readback row that displays the question, the setup-field alternative, and the refusal outcome;*
7. *H records componentwise* hsame *transports for A, S, Q, F, R, G together with continuation, provenance, and naming rows;*
8. *C records the displayed* Cont *routes by which consumers read the name, socket, axiom-use question, setup-field alternative, refusal ledger, and audit readback;*
9. *P is the* Pkg *provenance over* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{TypeLevelSocketExposure}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AxiomRefusalLedger}^{\uparrow}}$ *row.*

The carrier contains no coordinate that can be consumed as a Lean axiom, hidden theorem, host assumption, quotient representative, selected witness, or unrecorded primitive.

Theorem 1057.2 (AxiomRefusalLedger NameCert obligations)**.** *For the carrier of Definition 1057.1, the local* $\mathsf{NameCert}_{\mathsf{AxiomRefusalLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L = (A, S, Q, F, R, G, H, C, P, N$ *apophatic-name exposure through* A*, socket exposure through* S*, axiom-use question exactness through* Q*, setup-field alternative exposure through* F*, refusal-ledger coverage through* R*, audit-readback stability through* G*, componentwise transport through* H*, displayed route replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedger namecert obligations`

Proof. Project the displayed carrier. Its rows are exactly the apophatic name, socket, axiom-use question, setup-field alternative, refusal ledger, audit readback, transport, route, provenance, and local naming rows.

Read the name and socket rows through the sibling surfaces. The $\mathsf{ApophaticName}^{\uparrow}$ row names the unsupported dependency, and the $\mathsf{ApophaticSocket}^{\uparrow}$ row records the requested supply shape without producing that supply.

The question, alternative, and refusal rows are then read through Q, F, R, G. The question identifies the attempted axiom channel, F records the setup-field route that would make the dependency explicit, R refuses the internalization, and G makes the refusal available to audit consumers.

Finally H, C, P, N only transport, replay, package, and name the same visible packet. Since no row is an axiom, hidden theorem, host assumption, quotient representative, selected witness, or unrecorded primitive, the local NameCert obligations are exhausted by the displayed packet. □

Theorem 1057.3 (AxiomRefusalLedger noninternalization)**.** *Every accepted* $\mathsf{AxiomRefusalLedger}^{\uparrow}$ *packet refuses an axiom-use attempt as internal substrate content. A consumer may cite the rows* $A, S, Q, F, R, G, H, C, P, N$*, including the setup-field alternative and audit readback, but no coordinate of the accepted packet can be consumed as a Lean axiom, hidden theorem, host assumption, quotient representative, selected witness, or unrecorded primitive.*

Lean4 ✓ `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedger noninternalization`

Proof. Begin with the carrier inventory of Definition 1057.1. The packet displays the requested dependency position and the attempted axiom channel, but the only response rows are the setup-field alternative, refusal ledger, and audit readback.

Apply the NameCert obligations of Definition 1057.2. Every consumer route must pass through Q, F, R, G before it can cite the packet, so the request is classified and refused before it reaches downstream use.

The sibling boundaries of chapter 543 and chapter 536 keep the named dependency and socket shape outside generated content. Thus the accepted packet exports only visible refusal and setup-field exposure, never an internal axiom channel. □

Theorem 1057.4 (AxiomRefusalLedger gate readback)**.** *Every audit read of an accepted* $\mathsf{AxiomRefusalLedger}^{\uparrow}$ *packet factors through the route*

$$A \to S \to Q \to F \to R \to G \to (H, C, P, N).$$

The readback row G *reports the axiom-use question, the setup-field alternative, and the refusal ledger before any downstream consumer may cite the packet. No audit read can bypass* G *to*

recover a Lean axiom, hidden theorem, host assumption, quotient representative, selected witness, or unrecorded primitive.

Lean4 ✓ `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedger gate readback`

Proof. Read the displayed order in Definition 1057.1. The rows A, S, Q, F, R, G precede the structural rows and make the name, socket, question, alternative, refusal, and readback visible in that order.

Use Definition 1057.2. Audit-readback stability is one of the local certificate rows, so any audit consumer that cites the packet must preserve G together with Q, F, R.

Finally apply noninternalization, Definition 1057.3. Since the packet has no consumable axiom or hidden theorem coordinate, bypassing G cannot produce a stronger read; the only available audit route is the displayed one. □

Theorem 1057.5 (AxiomRefusalLedger sibling independence)**.** $\mathsf{AxiomRefusalLedger}^{\uparrow}$ *is not a renaming of* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{ExternalityGate}^{\uparrow}$, *or* TypeLevelSocketExpos *The axiom-refusal packet simultaneously carries an axiom-use question Q, a setup-field alternative F, a refusal ledger R, and an audit-readback row G over the same named socket. Each listed sibling lacks at least one of these coordinates, so no bijection of displayed rows preserves the local NameCert surface.*

Lean4 ✓ `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedgerSiblingIndependence`

Proof. Compare first with $\mathsf{ApophaticName}^{\uparrow}$ and $\mathsf{ApophaticSocket}^{\uparrow}$. Those siblings name the externality and requested supply shape, but they do not carry the paired axiom-use question and setup-field alternative rows Q, F.

Compare next with $\mathsf{ApophaticGateQuestion}^{\uparrow}$ and $\mathsf{ExternalityGate}^{\uparrow}$. Those siblings expose gate questions or readback gates, but they do not bind the question to a setup-field alternative and an axiom-channel refusal ledger in one packet.

Finally compare with $\mathsf{TypeLevelSocketExposure}^{\uparrow}$. It exposes type-level socket data, but it is not the audit ledger for refusing a Lean axiom or hidden theorem channel. Hence no row-preserving bijection can identify the carrier $A, S, Q, F, R, G, H, C, P, N$ with any of the listed sibling packets. □

Falsifiable prediction: Every accepted AxiomRefusalLedgerUp packet exposes $A, S, Q, F, R, G, H, C, P, N$ before citation, and no accepted packet has any coordinate that can be consumed as a Lean axiom or hidden theorem.

Independence witness: apophaticname, apophaticsocket, apophaticgatequestion, externalitygate, typelevelsocketexposure: AxiomRefusalLedgerUp alone carries the paired axiom-use question, setup-field alternative, refusal ledger, and audit-readback rows for one named socket; no sibling preserves all rows A,S,Q,F,R,G,H,C,P,N.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/AxiomRefusalLedgerUp/TasteGate.l` It must provide `Nontrivial AxiomRefusalLedgerUp` and `FieldFaithful AxiomRefusalLedgerUp` instances for the same displayed carrier.

Lean4 ✓ `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedgerTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomRefusalLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomRefusalLedger}^{\uparrow}$ packet is generated from an apophatic-name row, apophatic-socket row, axiom-use question, setup-field alternative, refusal ledger, audit readback, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{AxiomRefusalLedger}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1057.1, Definition 1057.2, Definition 1057.3, Definition 1057.4, and Definition 1057.5 bind the kernel scope over the setup-field alternative, refusal row, audit readback, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{AxiomRefusalLedger}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.AxiomRefusalLedgerUp.AxiomRefusalLedger\_namecert\_obligations`

Bridge status: none

Origin: ai

Not claimed: This chapter does not supply the refused axiom, hidden theorem, host assumption, quotient representative, selected witness, or bridge checking for AxiomRefusalLedger$^{\uparrow}$.

Upgrade path: Public closure requires a downstream consumer that reads the scoped refusal ledger as a reusable public audit interface.

Chapter 1058

A Concrete Naming Certificate for $\mathsf{SocketKindClassifier}^{\uparrow}$

The $\mathsf{SocketKindClassifier}^{\uparrow}$ packet realizes the socket-kind taxonomy of chapter 9394 as a finite BEDC carrier. It sits beside the socket ledger of chapter 528, the refusal surface of chapter 629, and the audit gate of chapter 526. Those siblings consume, refuse, or police sockets; this packet records the kind row that tells downstream consumers which supply aspect is being handled.

Definition 1058.1 (SocketKindClassifier carrier)**.** *A* $\mathsf{SocketKindClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, K_D, R, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the socket-site row naming the visible* GAP *or supply socket under classification;*
2. *K_D is the socket-kind row naming the Type I, Type II, Type III, or later taxonomy row from chapter 9394;*
3. *R is the requested-supply row recording the exact shortcut the site asks to consume;*
4. *G is the gate-locality row naming the audit gate that must reject or admit that kind at the local site;*
5. *H records componentwise* hsame *transport for the socket-site, kind, requested-supply, gate-locality, route, provenance, and naming rows;*
6. *C records displayed* Cont *routes through S, K_D, R, G before a consumer reads the classification;*
7. *P is the* Pkg *provenance over* $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{NegativeNameBoundary}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for a supplied witness, quotient equality, host-side harmlessness proof, anonymous socket kind, or global taxonomy oracle.

Lean4 (definition) `BEDC.Derived.SocketKindClassifierUp.SocketKindClassifierUp`

Theorem 1058.2 (SocketKindClassifier NameCert obligations)**.** *For the carrier of Definition 1058.1, the local* $\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (S, K_D, R, G, H, C, P, N)$,*

socket-site exposure through S, kind exposure through K_D, requested-supply exactness through R, gate-locality exactness through G, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Equivalently, the standard five-row NameCert surface is carrier admission, classifier-kind exactness, gate-locality stability, displayed route/provenance readback, and non-escape for omitted supply coordinates.

Proof. Project the displayed carrier. The accepted packet lists S, K_D, R, G, H, C, P, N, so the local certificate cannot introduce another socket site, kind row, requested-supply row, audit gate, transport, route, provenance, or local name.

Read the classification rows in order. The socket-site row S is displayed before the kind row K_D, and the requested-supply row R is read before the gate-locality row G can certify where the classification is policed.

Finally inspect H, C, P, N. These rows transport, replay, package, and name only the same finite packet. Therefore the NameCert obligations are exhausted by the displayed carrier rows. □

Lean4 ✓ `BEDC.Derived.SocketKindClassifierUp.SocketKindClassifierNameCertObligations`

Theorem 1058.3 (SocketKindClassifier gate-locality exactness)**.** *Every accepted* SocketKindClassifier$^{\uparrow}$ *packet binds its socket-kind row to the local audit gate displayed in G. A consumer that identifies a Type II witness socket with a Type III extensional-collapse socket fails this gate-locality row, because witness visibility and extensional-collapse refusal are distinct audit gates in chapter 9394.*

Proof. Begin with the socket-kind row K_D and requested-supply row R. Together they record which shortcut is requested: a missing witness, an extensional equality token, or another named taxonomy route.

Use the gate-locality row G. It ties the classification to the audit gate appropriate to that requested supply, so a witness-visibility gate cannot be read as an extensional-collapse gate.

The support rows H, C, P, N only transport and package the displayed classification. Since no row supplies a host-side coercion between Type II and Type III, the mixed identification is rejected by the local gate. □

Theorem 1058.4 (SocketKindClassifier sibling separation)**.** *The* SocketKindClassifier$^{\uparrow}$ *carrier is not bijective to* SupplySocketLedger$^{\uparrow}$, NegativeNameBoundary$^{\uparrow}$, AuditGateBoundary$^{\uparrow}$, ApophaticGateQuestion$^{\uparrow}$, *or* QuotientSoundnessBoundary$^{\uparrow}$. *The classifier has an explicit kind row K_D and gate-locality row G that these siblings respectively consume, refuse, gate, question, or specialize, so no row-preserving bijection can preserve all displayed NameCert fields.*

Proof. Compare the socket ledger first. SupplySocketLedger$^{\uparrow}$ records visible socket use, while the classifier records the kind row and its gate-locality row before use is consumed.

Compare the refusal and gate siblings next. NegativeNameBoundary$^{\uparrow}$ refuses internalization, AuditGateBoundary$^{\uparrow}$ records audit verdict surfaces, and ApophaticGateQuestion$^{\uparrow}$ asks the gate question; none of these carries the classifier's simultaneous socket-site, kind, requested-supply, and local-gate tuple.

Finally compare QuotientSoundnessBoundary$^{\uparrow}$. It specializes the Type III extensional-collapse refusal, while SocketKindClassifier$^{\uparrow}$ must also distinguish Type I and Type II rows and later taxonomy rows. The displayed fields therefore cannot be matched by a bijection preserving all NameCert rows. □

Theorem 1058.5 (SocketKindClassifier public export package)**.** *Every public export of an accepted* $\mathsf{SocketKindClassifier}^{\uparrow}$ *packet is exactly the finite route through* S, K_D, R, G, H, C, P, N*. Downstream consumers may use the export only as the socket-site surface* S*, the socket-kind surface* K_D*, the requested-supply surface* R*, the gate-locality surface* G*, the componentwise transport surface* H*, the displayed continuation surface* C*, the provenance surface* P*, and the local* $\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$ *surface* N*. The export contains no supplied witness, quotient equality, host harmlessness proof, anonymous socket kind, or global taxonomy oracle.*

Proof. Start from the carrier of Definition 1058.1. Its displayed packet has exactly the rows S, K_D, R, G, H, C, P, N, and the carrier definition already excludes the witness, quotient, host-harmlessness, anonymous-kind, and global-taxonomy coordinates from the packet.

Apply the NameCert obligations of Definition 1058.2. They identify the public certificate surface with carrier admission, classifier-kind exactness, gate-locality stability, displayed route and provenance readback, and non-escape for omitted supply coordinates.

Use gate-locality exactness from Definition 1058.3 together with sibling separation from Definition 1058.4. The first result prevents a consumer from reclassifying the exported kind through another local gate; the second prevents the package from being read as any sibling socket ledger, refusal, audit gate, gate question, or quotient boundary.

Finally read the TasteGate witness named in the closure scope. It certifies that the accepted packet is a field-faithful, nontrivial classifier over the same displayed rows, so the consumer-facing export is precisely the finite public route through S, K_D, R, G, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.SocketKindClassifierUp.SocketKindClassifierUp StdBridge`

Falsifiable prediction: A socket-kind classifier packet that identifies a Type II witness socket with a Type III extensional-collapse socket fails the gate-locality row.

Independence witness: Compare SupplySocketLedger, NegativeNameBoundary, AuditGateBoundary, ApophaticGateQuestion, and QuotientSoundnessBoundary: the classifier has an explicit kind row that these siblings either consume, refuse, gate, question, or specialize, so no bijection preserves all NameCert rows.

The Lean TasteGate module BEDC.Derived.SocketKindClassifierUp.TasteGate provides Nontrivial SocketKindClass and FieldFaithful $\mathsf{SocketKindClassifier}^{\uparrow}$ instances over the displayed rows S, K_D, R, G, H, C, P, N.

Closure status for $\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SocketKindClassifier}^{\uparrow}$ packet is generated from a socket-site row, a socket-kind row, a requested-supply row, a gate-locality row, componentwise transport, displayed continuation, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1058.1, Definition 1058.2, Definition 1058.3, Definition 1058.4, Definition 1058.5, and

Lean4 ✓ `BEDC.Derived.SocketKindClassifierUp.taste gate`

bind the supply taxonomy to a finite classifier packet with a consumer-facing public export and terminal paper-side package closure.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SocketKindClassifier}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.SocketKindClassifierUp.SocketKindClassifierTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not claim supplied-witness equality, quotient equality, host harmlessness, anonymous socket-kind collapse, or a global taxonomy oracle.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 1059

A Concrete Naming Certificate for $\mathsf{MachineInterfaceSocket}^{\uparrow}$

The $\mathsf{MachineInterfaceSocket}^{\uparrow}$ packet realizes the machine-facing formalization target interface of subsection 9462.35.5 as a finite socket-routed NameCert carrier. It sits beside the kind classifier of chapter 1058, the use ledger of chapter 1044, the audit gate boundary of chapter 526, the apophatic socket of chapter 536, and the supply socket ledger of chapter 528. Those siblings classify, record, police, refuse, or account for socket traffic; this packet routes formalization targets through the same finite audit-facing rows.

Definition 1059.1 (MachineInterfaceSocket carrier)**.** *A* $\mathsf{MachineInterfaceSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, K, U, A, S, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the formalization-target row recording the target name, intended namespace, source registry, statement skeleton, dependency list, expected status, and not-claimed surface from Definition 9462.2264;*
2. *K is the* $\mathsf{SocketKindClassifier}^{\uparrow}$ *row naming the socket kind that governs the target route;*
3. *U is the* $\mathsf{SocketUseLedger}^{\uparrow}$ *row recording the visible use of that socket by the machine-facing target;*
4. *A is the* $\mathsf{AuditGateBoundary}^{\uparrow}$ *row whose local verdict must accept the displayed socket kind before the target can be authorized;*
5. *S is the* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row tying the requested machine support to the finite supply ledger;*
6. *R is the refusal-locality row recording that refused audit-gate kinds do not authorize target admission;*
7. *H records componentwise* hsame *transport for T, K, U, A, S, R;*
8. *C records displayed* Cont *routes from the target row through the classifier, use ledger, audit gate, supply ledger, and refusal-locality row;*
9. *P is the* Pkg *provenance over* $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SocketUseLedger}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MachineInterfaceSocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for anonymous target admission, global namespace correctness, host-side proof search success, bypassed audit acceptance, supplied witness content, or a theorem asserting that every declared target is true.

Lean4 (definition) `BEDC.Derived.MachineInterfaceSocketUp.MachineInterfaceSocketUp`

Theorem 1059.2 (MachineInterfaceSocket NameCert obligations)**.** *For the carrier of Definition 1059.1, the local* $\mathsf{NameCert}_{\mathsf{MachineInterfaceSocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$M = (T, K, U, A, S, R, H, C, P, N),$$

formalization-target exposure through T*, socket-kind exactness through* K*, visible socket-use accounting through* U*, audit-gate boundary readback through* A*, supply-ledger accounting through* S*, refusal-locality through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Consequently every accepted machine-facing target route is a finite NameCert packet whose public reads are target row, socket-kind row, use row, audit-gate row, supply-ledger row, refusal-locality row, transport, continuation, provenance, and local name.

Proof. Project the displayed carrier. The packet lists $T, K, U, A, S, R, H, C, P, N$, so the local certificate cannot introduce another formalization target, socket kind, socket use, audit verdict, supply ledger, refusal row, transport, route, provenance, or local name.

Read the dependency rows in order. The target row T gives the machine-facing request, the classifier row K fixes the socket kind, the use ledger U records the visible invocation, the audit boundary A supplies the local verdict surface, and the supply ledger S accounts for the requested machine support.

Finally inspect R, H, C, P, N. The refusal-locality row blocks refused gate kinds, while the support rows only transport, replay, package, and name the same finite packet. Thus the NameCert obligations are exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceSocketUp.machineInterfaceSocket round trip`

Theorem 1059.3 (MachineInterfaceSocket gate locality)**.** *Every accepted* $\mathsf{MachineInterfaceSocket}^{\uparrow}$ *packet authorizes a formalization target only through the audit-gate row displayed in* A*. If the audit-gate row refuses the socket kind named by* K*, then the refusal-locality row* R *prevents the packet from authorizing the target named by* T*.*

Proof. Begin with the target and kind rows. The row T records the intended namespace, dependencies, expected status, and not-claimed surface, while K records the socket kind through which that target asks to be routed.

Use the use ledger and audit boundary. The row U makes the socket use visible, and A is the only displayed audit-gate boundary available to judge that use. No carrier coordinate stores a detached authorization predicate.

Apply refusal-locality. When A refuses the kind named by K, the row R records that refusal as a local obstruction to target admission. Since H, C, P, N only preserve the same rows, the target route cannot be authorized by another hidden gate. □

Theorem 1059.4 (MachineInterfaceSocket sibling separation)**.** *The* $\mathsf{MachineInterfaceSocket}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{SocketKindClassifier}^{\uparrow}$*,* $\mathsf{SocketUseLedger}^{\uparrow}$*,*

AuditGateBoundary$^{\uparrow}$, ApophaticSocket$^{\uparrow}$, *or* SupplySocketLedger$^{\uparrow}$. *The machine-interface socket carries the formalization-target row T together with the gate-locality row R, while those siblings respectively classify socket kinds, record uses, expose audit verdicts, refuse external supply, or account for supply sockets without preserving both rows at once.*

Proof. Compare the classifier and use ledger first. SocketKindClassifier$^{\uparrow}$ supplies kind exactness, and SocketUseLedger$^{\uparrow}$ records visible use, but neither carries the formalization-target row with namespace, dependency, expected-status, and not-claimed fields.

Compare the audit and apophatic siblings next. AuditGateBoundary$^{\uparrow}$ exposes audit verdict surfaces, and ApophaticSocket$^{\uparrow}$ records non-internalizing supply requests, but neither combines the target row with a socket-use ledger and the local refusal row for target authorization.

Finally compare SupplySocketLedger$^{\uparrow}$. It accounts for requested supply sockets but does not bind a machine-facing target row to the classifier, use ledger, audit gate, and refusal-locality rows in one packet. Therefore no bijection preserves the target row plus gate-locality row together with all displayed NameCert fields. □

Theorem 1059.5 (MachineInterfaceSocket public target export)**.** *Every public machine-facing target export carried by an accepted* MachineInterfaceSocket$^{\uparrow}$ *packet $M = (T, K, U, A, S, R, H, C, P, N)$ factors through the displayed rows*

$$T, K, U, A, S, R, H, C, P, N.$$

Here T is the target row, K the socket-kind classifier, U the use ledger, A the audit-gate row, S the supply ledger, R refusal locality, H transport, C replay, P provenance, and N local naming. The export has no coordinate for anonymous target admission, bypassed audit acceptance, global namespace correctness, supplied proof content, host-side proof search success, or a theorem asserting that every declared target is true.

Proof. Read the NameCert obligations of Definition 1059.2. They expose the target row, socket kind, use ledger, audit-gate row, supply ledger, refusal-locality row, transport, replay, provenance, and local name as the whole local certificate surface.

Apply gate locality from Definition 1059.3. A public export cannot bypass A, because refused socket kinds are blocked by R before the target row can be treated as authorized.

Finally use sibling separation, Definition 1059.4. None of the sibling packets preserves the formalization-target row together with the gate-locality row and all support rows, so the public machine-facing export must factor through $T, K, U, A, S, R, H, C, P, N$ inside this packet. □

Falsifiable prediction: A MachineInterfaceSocket$^{\uparrow}$ packet cannot authorize a formalization target whose audit-gate row refuses its socket kind.

Independence witness: socketkindclassifier, socketuseledger, auditgateboundary, apophatic-socket, supplysocketledger: the classifier lacks the formalization-target row, the use ledger lacks the target-bound gate refusal row, the audit gate lacks socket-use and supply-ledger accounting for target routing, the apophatic socket refuses supply without target admission, and the supply socket ledger lacks target-row authorization; no bijection preserves target row plus gate-locality row.

The Lean carrier file BEDC.Derived.MachineInterfaceSocketUp.TasteGate provides explicit Nontrivial MachineInter and FieldFaithful MachineInterfaceSocket$^{\uparrow}$ instances over the displayed packet.

Lean4 ✓ `BEDC.Derived.MachineInterfaceSocketUp.taste gate`

Closure status for NameCert$_{\text{MachineInterfaceSocket}^{\uparrow}}$ ·

Constructive story: A MachineInterfaceSocket$^{\uparrow}$ packet is generated from a target row, socket-kind row, use ledger, audit-gate row, supply-ledger row, refusal-locality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MachineInterfaceSocket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1059.1, Definition 1059.2, Definition 1059.3, Definition 1059.4, and Definition 1059.5 bind the target row, socket-kind classifier, socket-use ledger, audit-gate boundary, supply-ledger row, refusal-locality, componentwise hsame, displayed Cont, Pkg, and local NameCert rows as one scoped machine-interface socket.

Formal verification: FormalStatus(NameCert$_{\text{MachineInterfaceSocket}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MachineInterfaceSocketUp.machineInterfaceSocket\_round\_trip`

Bridge status: paperBridge

Not claimed: This chapter does not close bridge checking, global namespace correctness, host-side proof search success, supplied witness content, or truth of every declared target.

Upgrade path: No further theory-axis upgrade remains; machine-checked bridge status is a separate verification-axis obligation.

Chapter 1060

A Concrete Naming Certificate for $\mathsf{SchwarzReflectionRoute}^{\uparrow}$

The $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ packet realizes the reflection route isolated by chapter 9341 as finite NameCert data. It is a concrete operation-route sibling of chapter 577, using the holomorphic source interface of chapter 68, the differentiability control of chapter 67, and the complex carrier of chapter 39.

Definition 1060.1 (Schwarz reflection route carrier)**.** *A* $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, B, J, O, H, C, P, N)$$

with the following displayed rows:

1. *U is the holomorphic source row, recording the upper-side domain, the source function history, and the* $\mathsf{Holomorphic}^{\uparrow}$ *certificate over the visible side of the boundary;*
2. *B is the real-boundary trace row, recording the boundary segment, the continuity trace to that segment, and the real-value readback used by the reflection route;*
3. *J is the conjugation/reflection row, recording the domain reflection, the input conjugation $z \mapsto \overline{z}$, and the output conjugation $w \mapsto \overline{w}$ as displayed* $\mathsf{Complex}^{\uparrow}$ *routes;*
4. *O is the reflected-output row, recording the lower-side function history whose value is read as $\overline{f(\overline{z})}$ through the rows U, B, J;*
5. *H records componentwise* hsame *transports for U, B, J, O and for the replay, provenance, and local naming rows;*
6. *C records the displayed* Cont *replay routes from the source holomorphic row to the boundary trace, from the boundary trace to the conjugation/reflection row, and from that row to the reflected output;*
7. *P is the* Pkg *provenance over* $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{CplxDiff}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwarzReflectionRoute}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for a global analytic-continuation theorem, no monodromy selector, no quotient germ, no host complex-analysis equality, and no RH-facing zero-location predicate.

Lean4 ✓ BEDC.Derived.SchwarzReflectionRouteUp.TasteGate.SchwarzReflectionRouteTasteGate single carrier alignment

Theorem 1060.2 (Schwarz reflection route NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SchwarzReflectionRoute}^{\uparrow}}$ *surface for Definition 1060.1 consists of carrier admission for*

$$S = (U, B, J, O, H, C, P, N),$$

holomorphic source exposure, real-boundary trace exposure, conjugation/reflection exposure, reflected-output routing, componentwise hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer of the reflected output must first read the rows* U, B, J, H, C, P, N*.*

Proof. Carrier admission is projection from Definition 1060.1. The packet displays the source, boundary, reflection, output, transport, replay, provenance, and local naming rows before any reflected value is public.

The analytic side is then read through existing sibling certificates. The row U is a $\mathsf{Holomorphic}^{\uparrow}$ source row over $\mathsf{Complex}^{\uparrow}$ histories, and the reflection test uses the complex-differentiability control named by chapter 67; neither row is replaced by host complex equality.

The route rows H and C keep the same packet visible under transport and replay. The provenance row P records the dependence on $\mathsf{AnaCont}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{CplxDiff}^{\uparrow}$, and $\mathsf{Complex}^{\uparrow}$, while N supplies only the local name for this route. □

Theorem 1060.3 (Schwarz reflection route boundary symmetry)**.** *For an accepted* $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ *carrier* $S = (U, B, J, O, H, C, P, N)$*, the real-boundary trace row* B *is the only boundary symmetry datum used by the reflected-output row* O*. If the reflected route is read at a boundary point already displayed in* B*, then the source-side and reflected-side reads are compared only by the* hsame *transports in* H *and the replay routes in* C*.*

Proof. Project the boundary row B from the carrier. It stores the boundary segment, the continuity trace, and the real-value readback that makes conjugation inert at the boundary.

Next project the conjugation/reflection row J and the reflected-output row O. The output route is defined as a read through the source value followed by displayed input and output conjugation, so its boundary comparison has no extra datum beyond B, J, H, C.

Finally use the packet inventory. Since no carrier coordinate stores a host boundary theorem, a quotient germ, or a global analytic-continuation principle, boundary symmetry can be exported only as the displayed finite route through B, H, C. □

Theorem 1060.4 (Schwarz reflection route public handoff)**.** *Every* $\mathsf{AnaCont}^{\uparrow}$*,* $\mathsf{Holomorphic}^{\uparrow}$*, contour-facing, or zeta-continuation-facing consumer of an accepted* $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ *packet receives only the rows*

$$U, B, J, O, H, C, P, N$$

and their displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure. The handoff presents a boundary-symmetric reflection route, not a global branch selector, monodromy theorem, quotient-germ construction, RH predicate, or zero locator.*

Proof. Begin with Definition 1060.2. It exposes the finite carrier and forces the reflected-output read through the holomorphic source, real-boundary trace, conjugation/reflection, transport, replay, provenance, and local naming rows.

For analytic-continuation and holomorphic consumers, use the dependency rows named in P. They identify this packet as a route over $\mathsf{AnaCont}^{\uparrow}$ and $\mathsf{Holomorphic}^{\uparrow}$ rather than as an independent continuation theorem.

For contour-facing and zeta-facing consumers, the same provenance row supplies only a finite operation-route dependency. The consumer can cite the reflected-output row O, but it cannot extract a hidden branch, global zero classifier, or RH-facing predicate from the packet. □

Theorem 1060.5 (Schwarz reflection route non-escape)**.** *An accepted* $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ *carrier exports no object outside* U, B, J, O, H, C, P, N*. In particular, it exports no ambient Schwarz reflection principle, no selected analytic branch outside the reflected route, no monodromy closure, no quotient-germ equality, no host complex-analysis equality, and no non-trivial-zero locator.*

Proof. The carrier of Definition 1060.1 is a finite tuple with eight displayed rows. Any public read must project one of those rows or move through the displayed transport and replay rows.

The excluded objects are absent from that tuple. They are not source, boundary, reflection, output, transport, replay, provenance, or local naming rows.

Thus a consumer may use the packet as a reflected continuation route, but no consumer can read it as a global analytic theorem, a branch-selection device, a quotient construction, a host equality principle, or a zero-location certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchwarzReflectionRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwarzReflectionRoute}^{\uparrow}$ packet is generated from a holomorphic source row, a real-boundary trace row, a conjugation/reflection row, a reflected-output row, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwarzReflectionRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1060.1, Definition 1060.2, Definition 1060.3, Definition 1060.4, and Definition 1060.5 specify the finite reflection-route carrier, local naming obligations, boundary symmetry, public handoff, and non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwarzReflectionRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a global Schwarz reflection theorem, monodromy, quotient germs, RH consequences, zero-location predicates, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, boundary-symmetry replay, public handoff, non-escape coverage, TasteGate nontriviality, and field faithfulness for the same packet.

Chapter 1061

A Concrete Naming Certificate for $\mathsf{ApophaticFarEndSocket}^{\uparrow}$

The $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ packet realizes the far-end socket reading of chapter 9401 as finite NameCert data. It consumes the socket family row of chapter 536, the named boundary row of chapter 543, the supply-ledger row of chapter 528, the local inscription row of chapter 610, and the observer-history identity discipline of chapter 479. Its distinguishing role is the paired boundary: the unique far-end row is never internalized, and the local inscription row is still displayed before any observer-facing read is accepted.

Definition 1061.1 (Apophatic far-end socket carrier)**.** *An* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, A, T, G, I, O, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{ApophaticSocket}^{\uparrow}$ *row collecting the forward-binding socket family;*
2. *A is the* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the boundary position without supplying a kernel object;*
3. *T is the far-end boundary row recording that every displayed forward socket terminates at the same apophatic position;*
4. *G is the* GAP *ledger row that marks non-internalization of the far end;*
5. *I is the* $\mathsf{InscriptionPoint}^{\uparrow}$ *row recording the local inscription of the far-end boundary on the observer chain;*
6. *O is the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *row keeping observer identity on the displayed* BHist *side;*
7. *L is the* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row binding socket kind, supply shape, audit gate, and provenance;*
8. *H records componentwise* hsame *transports for $S, A, T, G, I, O, L, C, P, N$;*
9. *C records displayed* Cont *routes through the socket row, far-end boundary row, local inscription row, observer-history row, supply ledger, provenance, and local naming row;*
10. *P records* Pkg *provenance over* $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticFarEndSocket}^{\uparrow}}$ *row.*

The carrier contains no coordinate for T *as a kernel object, a state, a record, a* Hist *constructor, a subject parameter, a global frame, or an unmarked supply witness.*

Theorem 1061.2 (Apophatic far-end socket NameCert obligations)**.** *For the carrier of Definition 1061.1, the local* $\mathsf{NameCert}_{\mathsf{ApophaticFarEndSocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (S, A, T, G, I, O, L, H, C, P, N),$$

forward-socket exposure through S, apophatic naming through A, non-internalized far-end boundary exposure through T and G, local inscription exposure through I, observer-history identity through O, supply-ledger exposure through L, componentwise hsame *transport through H, displayed* Cont *route replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ApophaticFarEndSocketUp.TasteGate.ApophaticFarEndSocketNameCertObligations`

Proof. Project the accepted carrier. The rows $S, A, T, G, I, O, L, H, C, P, N$ are exactly the forward-socket, apophatic-name, far-end boundary, gap ledger, local-inscription, observer-history, supply-ledger, transport, route, provenance, and local naming rows.

Read the sibling packets in order. The socket and name rows come from chapter 536 and chapter 543; the supply-ledger row comes from chapter 528; and the observer-facing rows come from chapter 610 and chapter 479.

The rows T and G keep the far end as a boundary name rather than a generated object, while I records the local inscription where the same boundary is read on a displayed chain. Since the carrier has no further coordinate, these rows exhaust the local NameCert surface. □

Theorem 1061.3 (Apophatic far-end socket non-escape)**.** *An accepted* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *carrier exports only the displayed rows $S, A, T, G, I, O, L, H, C, P, N$. It exports no kernel object named* T*, no state, no record, no* Hist *constructor, no subject parameter behind the trace, no global frame, no unmarked supply witness, and no object-level equality between a self-center and the far-end boundary.*

Lean4 ✓ `BEDC.Derived.ApophaticFarEndSocketUp.TasteGate.ApophaticFarEndSocket nonescape`
↪ variant `BEDC.Derived.ApophaticFarEndSocketUp.TasteGate.ApophaticFarEndSocketNonEscape`

Proof. Use Definition 1061.2. Every accepted read factors through the finite row inventory $S, A, T, G, I, O, L, H, C,$

The far-end boundary is recorded only by the paired rows T and G. They name the common apophatic far side and record non-internalization; neither row is a kernel value, generated state, record, or BHist constructor for the far end itself.

The observer-facing read is recorded only by I and O. The inscription row is local to the displayed chain, and the observer-history row keeps identity on the BHist surface. Hence no subject parameter, global frame, hidden supply witness, or object-level equality with T can be exported. □

Theorem 1061.4 (Apophatic far-end socket TasteGate readiness)**.** *The* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *carrier is TasteGate-ready precisely because its non-internalized far-end rows T, G and its local inscription rows I, O are both constructor fields of the same finite packet. A Lean carrier for this chapter must therefore expose a nontrivial witness pair and a field-faithful projection over $S, A, T, G, I, O, L, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.ApophaticFarEndSocketUp.TasteGate.ApophaticFarEndSocketTasteGate single carrier alignment`

Proof. The carrier of Definition 1061.1 lists the far-end boundary and local inscription rows as ordinary displayed fields rather than as ambient assumptions. A TasteGate encoding can therefore flatten the packet to its finite BHist field list and read it back by the same constructor.

Nontriviality is witnessed by changing one displayed row while leaving the remaining rows fixed. Field faithfulness is witnessed by constructor-field projection: if two packets agree on $S, A, T, G, I, O, L, H, C, P, N$, then the packets are equal.

The non-escape theorem supplies the semantic boundary for this encoding. Removing T or G loses the non-internalized far-end boundary, while removing I or O loses the local inscription and observer-history discipline. The same finite packet is therefore the correct carrier for the paper-side NameCert surface and the Lean-side TasteGate. □

Theorem 1061.5 (Apophatic far-end socket public boundary)**.** *Every public consumer of an accepted* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *carrier receives only the finite rows*

$$S, A, T, G, I, O, L, H, C, P, N$$

together with the nonescape boundary of Definition 1061.3 and the TasteGate-readiness boundary of Definition 1061.4. The public surface exposes the far-end row as displayed boundary data and exposes the local inscription row as displayed observer-chain data; it exports no kernel object named T*, no global frame, no hidden supply witness, and no bridge credential.*

Proof. Project the accepted carrier. The only displayed rows available to a consumer are $S, A, T, G, I, O, L, H, C, P,$ and the nonescape theorem identifies the forbidden exports outside that row inventory.

The TasteGate-readiness theorem then fixes the same inventory as the Lean-facing carrier boundary. Thus the public read is exactly the finite row surface plus the nonescape and TasteGate-readiness constraints, not an ambient object for the far end. □

Falsifiable prediction: Removing either the far-end boundary row or the local inscription row makes Definition 1061.3 false: the former permits a hidden internalizer for T, while the latter permits a detached observer-facing read with no local inscription site.

Independence witness: apophaticsocket, apophaticname, supplysocketledger, inscriptionpoint, observerhistidentity: none is bijective with $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ because this carrier alone pairs the non-internalized far-end rows T, G with the local inscription and observer-history rows I, O while retaining the socket, name, ledger, transport, route, provenance, and local NameCert rows.

The Lean-side TasteGate surface provides Nontrivial $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ and FieldFaithful ApophaticFarEn instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticFarEndSocket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ packet is generated from a forward-socket row, apophatic-name row, far-end boundary row, gap ledger, local inscription row, observer-history identity row, supply-socket ledger row, componentwise transports, displayed routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.ApophaticFarEndSocketUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ApophaticFarEndSocket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1061.1, Definition 1061.2, Definition 1061.3, Definition 1061.4, and Definition 1061.5 bind the far-end socket packet as a finite public NameCert surface over the displayed boundary and local inscription rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ApophaticFarEndSocket}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ApophaticFarEndSocketUp.TasteGate.ApophaticFarEndSocketTasteGate\_single\_carrier\_alig`

Bridge status: none

Origin: ai

Not claimed: This chapter does not internalize T, add a subject parameter, supply a global frame, close a consciousness theorem, or provide a public standard bridge.

Upgrade path: No paper-side closure upgrade remains; a checked standard bridge is a verification-axis matter unless supplied separately.

Chapter 1062

A Concrete Naming Certificate for $\mathsf{DigestProvenancePacket}^{\uparrow}$

The $\mathsf{DigestProvenancePacket}^{\uparrow}$ carrier is the concrete NameCert realization of the digest-provenance discipline from chapter 9402. It sits downstream of the finite hash transcript surface of chapter 242, the negative socket naming discipline of chapter 543, and the visible supply refusal ledger of chapter 528. Its purpose is narrow: a digest may be used as a visible token only after the packet displays the source row, fiber-membership row, GAP ledger, exactness or refusal row, transport, replay, provenance, and local naming rows.

Falsifiable prediction: Every accepted DigestProvenancePacketUp consumer that reads source content must pass through the displayed fiber-membership and GAP ledger rows before any source equality claim.

Independence witness: hash, apophaticname, apophaticsocket, transcendentalsupplyledger, supplysocketledger: none is bijective to $\mathsf{DigestProvenancePacket}^{\uparrow}$ because this carrier binds the visible digest token, source row, fiber-membership row, GAP ledger, exactness or refusal row, hsame transport, Cont replay, provenance, and local NameCert row in one packet, while the siblings isolate hash evaluation, negative naming, socket request, supply taxonomy, or supply-kind ledger surfaces without the source-fiber discipline.

Definition 1062.1 (DigestProvenancePacket carrier)**.** *A* $\mathsf{DigestProvenancePacket}^{\uparrow}$ *carrier is a finite packet*

$$D = (V, S, F, G, E, H, C, P, N)$$

over the following displayed rows:

1. *V is the visible digest-token row, naming the token that downstream consumers may compare, store, or route;*
2. *S is the source row, recording the finite source class and the local source-channel handle without identifying the digest with the source;*
3. *F is the fiber-membership row, recording the generated relation that the source row belongs to the visible digest fiber;*
4. *G is the* GAP *ledger row, recording any far-end or missing-source supply required by the fiber reading;*
5. *E is the exactness or refusal row, recording whether the packet has an exactness certificate for a particular source claim or refuses source equality;*

6. *H records componentwise* hsame *transports for the visible digest, source, fiber, ledger, exactness, replay, provenance, and naming rows;*
7. *C records the displayed* Cont *replay routes by which consumers inspect V, S, F, G, E, H, P, N in order;*
8. *P is the* Pkg *provenance over the digest token, source row, fiber relation, GAP ledger, exactness or refusal row, transports, replay routes, and local naming surface;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DigestProvenancePacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global universe digest, digest-as-source equality, object-level T*, hidden preimage oracle, quotient source equality, or host hash-function semantics.*

Lean4 (definition) `BEDC.Derived.DigestProvenancePacketUp.DigestProvenancePacketCarrier`

Theorem 1062.2 (DigestProvenancePacket NameCert obligations)**.** *For an accepted* $\mathsf{DigestProvenancePacket}^{\uparrow}$ *carrier*

$$D = (V, S, F, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DigestProvenancePacket}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, token visibility through V, source-channel display through S, fiber-membership exactness through F, GAP ledger exposure through G, exactness or refusal through E, componentwise transport through H, replay locality through C, package provenance through P, local naming through N, and non-escape from the refused source and far-end supplies.*

Lean4 ✓ `BEDC.Derived.DigestProvenancePacketUp.DigestProvenancePacketCarrier namecert obligations`

Proof. Project the carrier in Definition 1062.1. The visible inventory is exactly $V, S, F, G, E, H, C, P, N$, so carrier admission and local naming are attached to these rows and to no hidden digest substrate.

Token visibility is read through V, while source-channel display is read through S. The hash-facing comparison surface of chapter 242 may supply digest transcripts, but the packet does not let such transcripts replace the displayed source row.

Fiber exactness and ledger exposure are the rows F and G. They force every source-sensitive read to name both the fiber-membership relation and the GAP entry that records missing far-end or source supply.

The exactness row E then separates certified source claims from refused ones. Transports and replay through H and C preserve this separation, and provenance through P records which local packet is being consumed.

Thus the NameCert surface refuses a global universe digest, digest-as-source equality, object-level T, hidden preimage oracle, quotient source equality, and host hash-function semantics. The apophatic naming discipline of chapter 543 and the supply ledger of chapter 528 are consumed only as visible boundary rows. □

Theorem 1062.3 (DigestProvenancePacket no hidden source consumption)**.** *Every accepted consumer of a* $\mathsf{DigestProvenancePacket}^{\uparrow}$ *carrier that reads source content must factor its read through the displayed source row S, fiber-membership row F, GAP ledger G, exactness or refusal row E, transport row H, replay route C, provenance row P, and local naming row N before making any source equality claim. A consumer that reads only the digest token V obtains a visible digest token and no source content.*

Lean4 ✓ `BEDC.Derived.DigestProvenancePacketUp.DigestProvenancePacketCarrier no hidden source consumption`

Proof. Start with a consumer read of an accepted packet $D = (V, S, F, G, E, H, C, P, N)$. If the read only asks for the digest token, Definition 1062.1 exposes V as a visible token and provides no source row outside the packet.

Now suppose the read asks for source content. The source channel is S, and the source may be attached to the token only through the fiber-membership row F. Without F, the read has a digest token and a displayed source row but no generated relation between them.

The next required row is the GAP ledger G. It records whether the fiber far end, source supply, or missing exactness data has been socketed rather than silently consumed. This is the same refusal posture enforced by chapter 543 and by the supply-ledger surface of chapter 528.

The exactness or refusal row E controls equality-facing claims. If E contains a local exactness certificate, the claim remains scoped to the displayed packet; if E is a refusal row, the source equality claim is unavailable. In neither case does the digest become the source.

Finally route the read through H, C, P, N. Componentwise transport, continuation replay, package provenance, and local naming preserve the same packet and do not add a preimage oracle, quotient representative, host hash semantics, global digest universe, or object-level T coordinate. Therefore no source content is consumed unless the displayed fiber and GAP ledger rows have already been opened. □

Theorem 1062.4 (DigestProvenancePacket source-fiber ledger order)**.** *For every accepted* DigestProvenancePacket *carrier*

$$D = (V, S, F, G, E, H, C, P, N),$$

each source-sensitive read must expose the source row S, then the fiber-membership row F, then the GAP *ledger row G, before the exactness or refusal row E may authorize or refuse an equality-facing source claim. A read that has not passed through S, F, G has no packet-local authority to assert source equality, digest-as-source equality, quotient source equality, hidden preimage recovery, or host hash-function semantics.*

Lean4 ✓ `BEDC.Derived.DigestProvenancePacketUp.DigestProvenancePacketCarrier source fiber ledger order`

Proof. Start from the carrier definition. The rows of Definition 1062.1 place the visible digest token V beside a source row S, then a fiber-membership row F, then the GAP ledger G, with E recording only the exactness or refusal status that follows those displayed rows.

Now use no hidden source consumption. Definition 1062.3 forces every source-content read through S, F, G, E, H, C, P, N before any source equality claim, while a token-only read exposes V and no source content.

It remains to identify the order relevant to equality-facing claims. The carrier gives no coordinate that can replace S by a hidden source, no coordinate that can replace F by a host preimage oracle, and no coordinate that can replace G by silent far-end supply. Therefore E can authorize or refuse only after the source, fiber, and ledger rows have been opened in that order. □

Closure status for $\mathsf{NameCert}_{\mathsf{DigestProvenancePacket}^{\uparrow}}$.

Constructive story: A $\mathsf{DigestProvenancePacket}^{\uparrow}$ packet is generated from a visible digest token, source row, fiber-membership row, GAP ledger, exactness or refusal row, componentwise transports, displayed replay routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DigestProvenancePacket}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1062.1, Definition 1062.2, Definition 1062.3, Definition 1062.4, and the TasteGate target bind the fiber-membership, GAP, refusal, replay, provenance, and naming rows to the displayed digest-provenance carrier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DigestProvenancePacket}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.DigestProvenancePacketUp.DigestProvenancePacketTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim source equality, preimage oracle access, quotient source equality, host hash semantics, a standard bridge, or object-level truth for DigestProvenancePacket$^{\uparrow}$.

Upgrade path: Bridge closure next requires a standard bridge for the scoped digest-provenance packet.

Chapter 1063

A Concrete Naming Certificate for MetaCICProofObjectAuditRoute$^{\uparrow}$

The MetaCICProofObjectAuditRoute$^{\uparrow}$ packet turns the synthesis view of chapter 9385 into a concrete route certificate for proof-object audit consumers. It sits beside **??**, chapter 776, chapter 731, chapter 717, and chapter 739. Those siblings expose the status matrix, synthesis packet, obstruction boundary, discharge ledger, and closed-normal endpoint; the present chapter records the route union that keeps those rows inspectable without converting the blocked subject-reduction edge into a full consistency theorem.

Definition 1063.1 (MetaCIC proof-object audit route carrier)**.** *A* MetaCICProofObjectAuditRoute$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, O, D, N, C, B, H, R, P, L)$$

with the following displayed rows:

1. *S is the synthesis-status row, recording the subject-reduction, confluence, normalization, substitution, decidability, and closed-normal status matrix read from chapter 9385;*
2. *O is the subject-reduction obstruction row, read through chapter 731;*
3. *D is the discharge-ledger row, exposing the four named subject-reduction slots through chapter 717;*
4. *N is the closed-normal packet row, recording the positive normal-endpoint consistency branch of chapter 739;*
5. *C is the confluence and decidable sibling route row, recording the conditional confluence routes and local checking surfaces that constrain proof-object reads;*
6. *B is the blocked-edge ledger, naming the unavailable transport from arbitrary closed proof objects to the closed-normal branch;*
7. *H records componentwise* hsame *transports for S, O, D, N, C, B and the local structural rows;*
8. *R records displayed* Cont *routes by which proof-object audit consumers read the synthesis, obstruction, discharge, closed-normal, confluence, decidable, and blocked-edge rows;*
9. *P is the* Pkg *provenance over* **??**, *chapter 776, chapter 731, chapter 717, chapter 739,* BHist, hsame, Cont, *and* NameCert *rows;*

10. L is the local $\mathsf{NameCert}_{\mathsf{MetaCICProofObjectAuditRoute}^{\uparrow}}$ *row.*
The carrier is not bijective with any one sibling packet. It stores the route union tying positive closed-normal evidence to the obstruction ledger, so agreement on only the synthesis packet, only the obstruction packet, or only the closed-normal packet does not reconstruct the proof-object audit route.

Theorem 1063.2 (MetaCIC proof-object audit route NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICProofObjectAuditRoute}^{\uparrow}$ *carrier*

$$A = (S, O, D, N, C, B, H, R, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICProofObjectAuditRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, synthesis-status exposure through S, obstruction and four-slot ledger exposure through O, D, closed-normal endpoint exposure through N, confluence and decidable sibling route exposure through C, blocked-edge ledger exactness through B, componentwise* hsame *transport through H, displayed route replay through R, package provenance through P, and local naming through L.*

Proof. Carrier admission is projection from Definition 1063.1. The rows $S, O, D, N, C, B, H, R, P, L$ are the complete public packet, so every NameCert obligation is read from one of those coordinates.

The proof-object content is split across the sibling rows. The synthesis-status row S records the audit-map matrix, O, D record the obstruction and discharge slots, N records the closed-normal endpoint branch, and C records the confluence and decidable side routes.

The blocked-edge obligation is the row B. Transport, route replay, provenance, and local naming are confined to H, R, P, L, and these structural rows preserve the displayed tuple rather than adding a second route. Thus the local NameCert surface is exactly the finite route packet stated above. □

Lean4 ✓ `BEDC.Derived.MetaCICProofObjectAuditRouteUp.MetaCICProofObjectAuditRoutePacket namecert obligations`

Theorem 1063.3 (MetaCIC proof-object audit route blocked-edge non-escape)**.** *Every proof-object audit consumer of an accepted* $\mathsf{MetaCICProofObjectAuditRoute}^{\uparrow}$ *carrier reads the closed-normal branch only through N together with the blocked-edge ledger B and the obstruction rows O, D. If B is removed from the carrier, the packet no longer certifies blocked-edge non-escape: a consumer can still see the positive closed-normal row N, but the displayed refusal of the missing subject-reduction transport is no longer part of the same route.*

Proof. Read the closed-normal side first. The row N exposes the positive closed-normal packet, while O, D expose the subject-reduction obstruction and its four discharge slots.

The non-escape condition is carried by B. It records that the route from arbitrary closed proof objects to the closed-normal branch is blocked, so any consumer of N must carry the obstruction ledger in the same packet.

Removing B changes the public certificate surface. The tuple would still contain synthesis, obstruction, discharge, closed-normal, confluence, decidable, transport, route, provenance, and naming rows, but it would no longer contain the row that marks the missing transport edge. Hence blocked-edge non-escape is falsifiable by deleting B. □

Lean4 ✓ `BEDC.Derived.MetaCICProofObjectAuditRouteUp.MetaCICProofObjectAuditRoute blocked edge nonescape`

Theorem 1063.4 (MetaCIC proof-object audit route sibling independence)**.** *The* MetaCICProofObjectAuditRoute$^{\uparrow}$ *carrier is independent from the sibling carriers of* **??**, *chapter 776, chapter 731, chapter 717, and chapter 739. Each sibling supplies one route face, but none of them alone contains the union* $S, O, D, N, C, B, H, R, P, L$ *that ties positive closed-normal evidence to the obstruction ledger.*

Proof. Inspect the sibling roles. The synthesis siblings provide status and combined audit rows, the obstruction sibling provides the blocked subject-reduction boundary, the discharge ledger provides the four named slots, and the closed-normal packet provides the positive endpoint branch.

The present carrier records those rows together with the blocked-edge ledger and a single structural transport, route, provenance, and naming surface. Therefore it is not a repackaging of any one sibling packet.

The independence witness is componentwise: deleting the synthesis row loses the matrix, deleting O or D loses the subject-reduction boundary, deleting N loses the positive closed-normal evidence, and deleting B loses the non-escape certificate. No sibling carrier supplies all of those rows at once. □

Theorem 1063.5 (MetaCIC proof-object audit route TasteGate readiness)**.** *The* MetaCICProofObjectAuditRoute$^{\uparrow}$ *packet is ready for a* TasteGate *carrier precisely at the displayed finite surface*

$$(S, O, D, N, C, B, H, R, P, L).$$

The Lean follow-up must provide Nontrivial *and* FieldFaithful *instances for this carrier in* lean4/BEDC/Derived/Meta *with the public alignment theorem named* BEDC.Derived.MetaCICProofObjectAuditRouteUp.MetaCICProofObjectAud

Lean4 ✓ `BEDC.Derived.MetaCICProofObjectAuditRouteUp.MetaCICProofObjectAuditRouteTasteGate`
`single carrier alignment`

Proof. TasteGate readiness uses the finite packet already displayed in the carrier definition. The row list exposes one public carrier, one classifier surface, and one provenance route for proof-object audit consumers.

Nontriviality is witnessed by the separation between the positive closed-normal row N and the blocked-edge ledger B. Field faithfulness is witnessed by componentwise access to $S, O, D, N, C, B, H, R, P, L$, since deleting or identifying any of these rows changes a named audit obligation.

Thus the follow-up Lean target is not a new paper claim about closed consistency. It is the formal TasteGate encoding of the same finite route packet and its componentwise field discipline. □

Closure status for NameCert$_{\text{MetaCICProofObjectAuditRoute}^{\uparrow}}$**.**

Constructive story: A MetaCICProofObjectAuditRoute$^{\uparrow}$ packet is generated from synthesis-status, subject-reduction obstruction, discharge-ledger, closed-normal packet, confluence and decidable sibling route, blocked-edge ledger, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICProofObjectAuditRoute}^{\uparrow}}$, seed)

Scope closed: Definition 1063.1, Definition 1063.2, Definition 1063.3, Definition 1063.4, and Definition 1063.5 bind proof-object audit routes as finite NameCert rows tying closed-normal evidence to the subject-reduction blocked-edge ledger.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICProofObjectAuditRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, full closed MetaCIC consistency, global Church–Rosser, candidate-free normalization, global decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Nontrivial and FieldFaithful instances, carrier alignment, blocked-edge non-escape, sibling independence, and TasteGate readiness for the displayed finite route packet.

Chapter 1064

A Concrete Naming Certificate for $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$

The $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ packet gives the fixed-point reading of chapter 9401 a finite NameCert surface. It records the apophatic socket, the fiber row whose far end is being named, the GAP ledger that keeps the requested supply outside the substrate, the boundary row that refuses internalization of T, the local inscription row where the same event is read on a chain, and the usual transport, route, provenance, and local naming rows. It sits beside the socket carrier of chapter 536, the fixed-point carrier of chapter 556, the inscription carrier of chapter 519, and the no-subject identity surface of chapter 479.

Definition 1064.1 (Apophatic fiber far-end carrier)**.** *An* $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, F, G, B, I, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{ApophaticSocket}^{\uparrow}$ *row naming the visible forward-binding socket;*
2. *F is the fiber row recording the socket-to-far-end request without providing the far end as an internal value;*
3. *G is the* GAP *ledger row attached to the requested far end of the fiber;*
4. *B is the apophatic boundary row recording that* T *is not a* BHist, Hist, *digest, state, theorem object, setup field, or generated witness;*
5. *I is the local inscription row recording the chain point where the boundary name is read inscription-side without adding a subject coordinate;*
6. *H records componentwise* hsame *transports for S, F, G, B, I, C, P, N;*
7. *C records the displayed* Cont *routes by which consumers read the socket, fiber, ledger, boundary, inscription, provenance, and local naming rows;*
8. *P is the* Pkg *provenance over* $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for T *as an internal object, no route from the far-end boundary to a digest or history value, and no hidden observer-substance row behind the local inscription.*

Lean4 (definition) `BEDC.Derived.ApophaticFiberFarEndUp.ApophaticFiberFarEndUp`

Theorem 1064.2 (Apophatic fiber far-end NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}$ *surface for Definition 1064.1 consists of five finite obligations:*

1. *carrier admission for the packet* $A = (S, F, G, B, I, H, C, P, N)$*;*
2. *socket-fiber locality, stating that* S *and* F *are read together before any far-end boundary row is consumed;*
3. *ledger and boundary exactness, stating that* G *and* B *record the apophatic far end only as a boundary name and never as a generated substrate coordinate;*
4. *local inscription accountability, stating that* I *is a chain-local reading of the same event and not a subject parameter, private center, or equality with* T*;*
5. *public route exactness, stating that every consumer read factors through* $S, F, G, B, I, H, C, P, N$ *and no route exports* T*, a digest, a* BHist *value, a hidden theorem, or an observer-substance row.*

Lean4 ✓ `BEDC.Derived.ApophaticFiberFarEndUp.ApophaticFiberFarEndTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1064.1. The displayed packet lists the socket, fiber, ledger, boundary, local inscription, transport, route, provenance, and local naming rows.

Socket-fiber locality is read through S and F. The socket identifies the visible forward-binding request, while the fiber row records the far-end request without supplying that far end as an object.

Ledger and boundary exactness are carried by G and B. These rows record the GAP far-end boundary and the refusal of internal T; neither row adds a coordinate from which a consumer can project T as a BHist, Hist, digest, state, theorem object, setup field, or generated witness.

The local inscription row I is then read as chain-local accountability. It ties the boundary name to a displayed inscription point, but it does not introduce an extra subject row or an object-level equality with T.

Finally H, C, P, N provide transport, continuation replay, provenance, and local naming. Since every public route is a projection through these rows and the carrier has no further coordinate, the obligation surface is exhausted by the finite packet $S, F, G, B, I, H, C, P, N$. □

Theorem 1064.3 (Apophatic fiber far-end noninternality)**.** *For every accepted* $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ *packet* $A = (S, F, G, B, I, H, C, P, N)$*, the far-end boundary named by* B *cannot be consumed as a* BHist*,* Hist*, digest, state, theorem object, setup field, generated witness, or observer-substance row. Any accepted consumer of the local inscription row* I *must still pass through the socket row* S*, fiber row* F*,* GAP *ledger* G*, boundary row* B*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*.*

Proof. Project the boundary row B from the carrier. It records the apophatic far end as a refusal row, not as a generated object or value inside the substrate.

Project the local inscription row I next. Its public role is to attach the boundary name to a chain-local read of an event; it does not add a subject coordinate or identify that local event with T as an object.

The only available consumer routes are those listed in C and transported through H. Together with S, F, G, P, N, they force every accepted read to remain inside the displayed packet. Since none of the forbidden outputs is a carrier coordinate, no accepted route can extract it. □

Theorem 1064.4 (Apophatic fiber far-end falsifiable prediction)**.** *If an accepted* $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ *packet admits a consumer route from the apophatic far-end row to a* BHist, hsame, *or digest value row without passing through its boundary ledger, the certificate is invalid.*

Proof. The carrier exposes the far-end only through the fiber row F, GAP ledger G, and apophatic boundary row B. A consumer route that reaches a value row while bypassing G and B therefore omits the rows that make the far-end read apophatic.

Apply public route exactness from Definition 1064.2. Every accepted read must factor through $S, F, G, B, I, H, C, P, N$. A route that bypasses the boundary ledger is not one of the displayed continuation routes and cannot be certified by the local $\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}$ row. □

Theorem 1064.5 (Apophatic fiber far-end independence witness)**.** *The* $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, *or* $\mathsf{ObserverHistIdentity}^{\uparrow}$. *These siblings respectively supply socket visibility, gate questioning, generator fixed-output witnessing, public inscription event routing, and no-subject history identity, but none carries the fiber ledger, local inscription row, and far-end noninternality row together.*

Proof. Compare the socket and gate siblings first. $\mathsf{ApophaticSocket}^{\uparrow}$ supplies the visible socket row, and $\mathsf{ApophaticGateQuestion}^{\uparrow}$ supplies the refusal question, but neither contains the combined fiber row F, far-end ledger G, boundary row B, and local inscription row I.

Compare the fixed-point and inscription siblings next. $\mathsf{GeneratorFixedPoint}^{\uparrow}$ records a fixed-output witness for an authorized generator, while $\mathsf{InscriptionEvent}^{\uparrow}$ records accepted public inscription routing. Neither is a far-end boundary carrier, and neither refuses T as a BHist, Hist, digest, state, theorem object, setup field, or generated witness.

Finally compare $\mathsf{ObserverHistIdentity}^{\uparrow}$. It supplies the no-subject history-identity discipline consumed by the inscription row, but it has no fiber-far-end ledger and no apophatic boundary row. Therefore any bijection preserving NameCert rows would have to delete at least one of F, G, B, I, so no such bijection exists. □

Falsifiable prediction: If an accepted packet admits a consumer route from the apophatic far-end row to a BHist, hsame, or digest value row without passing through its boundary ledger, the certificate is invalid.

Independence witness: apophaticsocket, apophaticgatequestion, generatorfixedpoint, inscriptionevent, observerhistidentity: none is bijective because each lacks at least one of the fiber ledger, local inscription, and far-end noninternality rows carried together here.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ApophaticFiberFarEnd}^{\uparrow}$ packet is generated from an accepted apophatic socket, fiber row, GAP far-end ledger, boundary row, local inscription row, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The BHist carrier first exposes the socket and fiber request, then records the far end only as a ledgered boundary, and finally reads the local inscription without adding a subject coordinate. The classifier preserves the far end as noninternal data: every consumer must pass through socket, fiber, ledger, boundary, inscription, route, provenance, and name.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the apophatic fiber far-end carrier, obligations, noninternality, falsifiable prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticFiberFarEnd}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim the far end as a BHist, digest, state, theorem object, setup field, generated witness, or observer substance.

Upgrade path: To reach obligationClosure, turn the existing aligned TasteGate carrier into explicit checked obligations for the fiber, ledger, boundary, and inscription rows.

Chapter 1065

A Concrete Naming Certificate for $\mathsf{ExternalSupplyBoundary}^{\uparrow}$

The $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ packet realizes the boundary discipline of chapter 9395 as a finite NameCert object. It sits beside the socket ledger of chapter 528, the apophatic socket surface of chapter 536, the audit gate boundary of chapter 526, and the gate-question packet of chapter 715. Those siblings expose socket use, requested externality, gate rows, or audit questions; this packet binds the complete boundary record that keeps external supply visible without making the requested supply internal.

Definition 1065.1 (ExternalSupplyBoundary carrier)**.** *An* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, R, I, A, G, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-name row naming the exact external supply site;*
2. *R is the requested-supply-shape row, recording what the site asks to consume;*
3. *I is the non-internalization row, recording that the requested supply is not available as substrate truth;*
4. *A is the accepted-boundary-form row, one of setup field, NameCert row, GAP ledger, bridge slot, or apophatic socket;*
5. *G is the associated* GAP *ledger row preserving the residue or socket pressure generated by the request;*
6. *Q is the gate-question row naming the finite audit question that must be asked before any consumer reads the boundary;*
7. *H records componentwise* hsame *transport for B, R, I, A, G, Q;*
8. *C records displayed* Cont *replay routes through the same boundary rows;*
9. *P is the* Pkg *provenance over* $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for an internal witness of the requested supply, no hidden selector, no host quotient, and no theorem asserting that the requested supply exists.

Lean4 ✓ `BEDC.Derived.ExternalSupplyBoundaryUp.ExternalSupplyBoundaryTasteGate single carrier alignment`

Theorem 1065.2 (ExternalSupplyBoundary NameCert obligations)**.** *For the carrier of Definition 1065.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* $E = (B, R, I, A, G, Q, H, C, P, N)$*, boundary-name exposure through* B*, requested-supply exactness through* R*, non-internalization exactness through* I*, accepted-form exposure through* A*,* GAP*-ledger readback through* G*, gate-question readback through* Q*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Equivalently, a consumer may read the boundary name, requested supply shape, non-internalization status, accepted boundary form, associated GAP *ledger, and gate question, but no requested supply content outside those rows.*

Proof. Project the displayed carrier. The accepted packet lists $B, R, I, A, G, Q, H, C, P, N$, so the local certificate cannot introduce a second boundary name, requested-supply shape, non-internalization row, accepted boundary form, GAP ledger, gate question, transport row, replay route, provenance row, or local name.

Read the obligation rows in order. The boundary name B identifies the site, the requested-supply row R names the demanded shape, and the non-internalization row I prevents that request from becoming substrate truth.

Next read A, G, Q. The accepted-boundary-form row states how the request remains visible, the GAP row records the residue or socket pressure, and the gate-question row states the audit question a consumer must ask.

Finally inspect H, C, P, N. These rows only transport, replay, package, and name the same finite packet. Therefore the NameCert obligations are exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.ExternalSupplyBoundaryUp.ExternalSupplyBoundary namecert obligations`

Theorem 1065.3 (ExternalSupplyBoundary non-internalization)**.** *Every accepted* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ *packet refuses internalization of its requested supply. A downstream consumer may cite* $B, R, I, A, G, Q, H, C, P, N$ *as boundary evidence, but it may not read from the packet an internal witness, selector, quotient representative, proposition equality token, or theorem that supplies the requested content.*

Proof. Begin with the non-internalization row I. By the carrier definition, I is displayed as a row of the packet rather than derived from a hidden consumer convention.

Use the accepted-boundary-form row A together with the GAP ledger G. The row A records the visible form in which the request is held, and G records the residue or socket pressure that would otherwise be erased.

Use the gate-question row Q to bind the consumer read to an audit question. A consumer can ask whether the boundary remains visible, but the packet contains no coordinate that answers by supplying the requested content.

The remaining rows H, C, P, N preserve and name only the displayed boundary rows. Since no coordinate stores an internal witness, selector, quotient representative, proposition equality token, or supplied-content theorem, every accepted packet keeps the requested supply external. □

Theorem 1065.4 (ExternalSupplyBoundary accepted form exhaustion)**.** *For every accepted* ExternalSupplyBounda *packet*

$$E = (B, R, I, A, G, Q, H, C, P, N),$$

the accepted boundary form row A *is exhausted by exactly one of the displayed boundary forms: setup field, NameCert row,* GAP *ledger, bridge slot, or apophatic socket. The packet exposes no accepted form outside those rows and no internal witness of the requested supply.*

Proof. Project the carrier of Definition 1065.1. The accepted-form coordinate is the row A, and it is read beside $B, R, I, G, Q, H, C, P, N$.

Use Definition 1065.2. The local NameCert surface binds accepted-form exposure to carrier admission, requested-supply exactness, non-internalization, GAP-ledger readback, gate-question readback, transport, replay, provenance, and local naming.

Finally apply Definition 1065.3. Since the requested supply is not internalized, the accepted form cannot be replaced by a hidden witness, selector, quotient representative, proposition equality token, or supplied-content theorem. □

Lean4 ✓ `BEDC.Derived.ExternalSupplyBoundaryUp.ExternalSupplyBoundary accepted form exhaustion`

Theorem 1065.5 (ExternalSupplyBoundary gate question determinacy)**.** *For every accepted* ExternalSupplyBound *packet*

$$E = (B, R, I, A, G, Q, H, C, P, N),$$

the gate-question row Q is deterministic for downstream reads. Once the boundary name B, requested-supply shape R, non-internalization row I, accepted form A, and GAP *ledger G are fixed, every consumer reads the same finite question Q through H, C, P, N.*

Proof. Project the displayed carrier. The row Q is a coordinate of the same finite packet as $B, R, I, A, G, H, C, P, N$, so there is no second question row available to a consumer.

Use Definition 1065.2. Gate-question readback is one of the local obligations, and the structural rows H, C, P, N only transport, replay, package, and name that row.

Apply Definition 1065.3. The determinacy of Q cannot be bypassed by reading an internal witness, selector, quotient representative, proposition equality token, or theorem that supplies the requested content. □

Theorem 1065.6 (ExternalSupplyBoundary sibling compatibility)**.** *Every accepted* ExternalSupplyBoundary$^{\uparrow}$ *packet*

$$E = (B, R, I, A, G, Q, H, C, P, N)$$

is compatible with the sibling surfaces SupplySocketLedger$^{\uparrow}$, ApophaticSocket$^{\uparrow}$, AuditGateBoundary$^{\uparrow}$, *and* ApophaticGateQuestion$^{\uparrow}$. *The row P records those sibling provenances, while A, G, Q expose the accepted boundary form, socket or residue ledger, and gate question needed by the sibling reads. The compatibility does not internalize the requested supply.*

Proof. Project the carrier. The provenance row P explicitly ranges over the socket ledger, apophatic socket, audit gate boundary, and apophatic gate question sibling surfaces, while A, G, Q display the accepted form, residue ledger, and question rows they consume.

Use Definition 1065.2. It binds sibling-facing provenance to the same carrier admission, non-internalization, accepted-form readback, GAP-ledger exposure, gate-question readback, transport, replay, and local naming rows.

Finally use Definition 1065.3. The sibling compatibility route can expose socket use, apophatic refusal, audit boundary pressure, and gate-question reads, but it has no coordinate for an internal witness, selector, quotient representative, proposition equality token, or supplied-content theorem. □

Theorem 1065.7 (ExternalSupplyBoundary carrier admission)**.** *Every accepted* ExternalSupplyBoundary$^{\uparrow}$ *boundary read is admitted only by the finite* BHist *carrier*

$$E = (B, R, I, A, G, Q, H, C, P, N).$$

The local $\mathsf{NameCert}_{\mathsf{ExternalSupplyBoundary}^{\uparrow}}$ *row* N *projects exactly this tuple: boundary name* B*, requested-supply shape* R*, non-internalization row* I*, accepted-boundary-form row* A*,* GAP *ledger* G*, gate question* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, package provenance* P*, and local naming* N*.*

Proof. Start from the carrier definition. The packet is displayed as the ten-coordinate finite history $E = (B, R, I, A, G, Q, H, C, P, N)$, and every coordinate is named before the local certificate row is read.

Use Definition 1065.2. Carrier admission is the first obligation of the local NameCert surface, and the later obligations are all projections from the same displayed packet.

Thus an accepted boundary read has no second carrier and no unlisted coordinate. It is admitted exactly through the finite BHist tuple named by N. □

Theorem 1065.8 (ExternalSupplyBoundary GAP ledger exposure)**.** *Every accepted* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ *read exposes requested-supply pressure factors only through the* GAP *ledger row* G*, while the non-internalization row* I *records that the same request is not available as internal substrate content. The exposure is therefore a ledger read of boundary pressure, not a supply witness.*

Proof. Read the packet through Definition 1065.7. The rows I and G are coordinates of the same finite carrier, so a consumer cannot separate ledger pressure from the non-internalization status that names the request as external.

Apply Definition 1065.2. The local surface lists non-internalization exactness through I and GAP-ledger readback through G as distinct obligations over the same boundary packet.

Finally use Definition 1065.3. Since no coordinate supplies the requested content, the G read exposes only residue or socket pressure for the request named by R and refused by I. □

Theorem 1065.9 (ExternalSupplyBoundary socket readback)**.** *Every sibling boundary read through* $\mathsf{SupplySocketLedger}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*,* $\mathsf{AuditGateBoundary}^{\uparrow}$*, and* $\mathsf{ApophaticGateQuestion}^{\uparrow}$ *must pass through the displayed rows* A, G, Q, H, C, P, N *of the same accepted* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ *packet. The readback may expose accepted boundary form, socket or residue ledger, gate question, transport, replay, provenance, and local naming, but it may not bypass those rows to obtain requested supply content.*

Proof. Use Definition 1065.6. The sibling surfaces named there consume the provenance row P together with the accepted form A, ledger G, and gate question Q.

Use Definition 1065.7 to place those rows inside the single finite carrier. The transport row H, replay row C, provenance row P, and naming row N keep the sibling read attached to that carrier rather than to an external socket convention.

Apply Definition 1065.8 and Definition 1065.3. The readback can carry socket pressure and the audit question, but the requested supply remains external to the packet. □

Falsifiable prediction: A boundary packet that omits either the non-internalization row or the GAP-ledger row cannot satisfy the non-internalization theorem.

Independence witness: Compare SupplySocketLedger, ApophaticSocket, AuditGateBoundary, and ApophaticGateQuestion: none is bijective to ExternalSupplyBoundary, because each lacks at least one of the requested-supply shape, accepted boundary form, GAP ledger, or gate-question rows required by the packet.

The Lean follow-up must provide explicit Nontrivial ExternalSupplyBoundary$^{\uparrow}$ and FieldFaithful ExternalSupplyBou instances in lean4/BEDC/Derived/ExternalSupplyBoundaryUp/TasteGate.lean before the chapter can claim a checked taste-gate target.

Closure status for NameCert$_{\text{ExternalSupplyBoundary}^{\uparrow}}$.

Constructive story: An ExternalSupplyBoundary$^{\uparrow}$ packet is generated from a boundary-name row, requested-supply-shape row, non-internalization row, accepted-boundary-form row, GAP ledger row, gate-question row, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ExternalSupplyBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1065.1, Definition 1065.2, Definition 1065.3, Definition 1065.4, Definition 1065.5, Definition 1065.6, Definition 1065.7, Definition 1065.8, and Definition 1065.9 bind the finite boundary packet, its NameCert obligations, non-internalization row, accepted-form exhaustion, gate-question determinacy, sibling compatibility, carrier admission, GAP-ledger exposure, and socket readback.

Lean4 ✓ `BEDC.Derived.ExternalSupplyBoundaryUp.taste gate`

Formal verification: FormalStatus(NameCert$_{\text{ExternalSupplyBoundary}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.ExternalSupplyBoundaryUp.ExternalSupplyBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close internal witnesses, selectors, quotient representatives, supplied-content theorems, Lean TasteGate verification, or bridge checking.

Upgrade path: The theory axis has no higher closure grade; the remaining possible improvement is a bridgeCheckedV formal-axis target.

Chapter 1066

A Concrete Naming Certificate for $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$

The $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ packet realizes the seven boundary forms of chapter 9395 as a finite classifier over existing BEDC surfaces. It consumes the name-to-gate discipline of chapter 584, the apophatic socket boundary of chapter 536, and the supply ledger of chapter 528. The packet classifies visible boundary forms; it does not turn the classified dependency into an internal theorem.

Definition 1066.1 (Non-axiom boundary form carrier)**.** *A* $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, S, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary site where unsupported supply, compression residue, or interface input would otherwise be consumed as substrate truth;*
2. *K is exactly one of the seven non-axiom boundary-form rows: inductive, def, setup-field, NameCert row, GAP ledger, bridge slot, or apophatic socket;*
3. *S is the source-surface row attached to K: a closed generator for inductive rows, a definitional source for def rows, a caller-provided setup field, a public* $\mathsf{NameCert}$ *surface, a* GAP *ledger, a separated bridge slot, or an* $\mathsf{ApophaticSocket}^{\uparrow}$ *boundary;*
4. *G is the gate row read through* $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, *or* $\mathsf{SupplySocketLedger}^{\uparrow}$ *when the classified form is a visible external-supply boundary;*
5. *H records componentwise* hsame *transport for the boundary site, kind row, source surface, gate row, route row, provenance row, and local naming row;*
6. *C records displayed* Cont *routes by which consumers read B, K, S, G before using the classifier result;*
7. *P is the* Pkg *provenance over chapter 9395, chapter 584, chapter 536, chapter 528,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{NonAxiomBoundaryForm}^{\uparrow}}$ *row.*

The carrier exports no coordinate for an axiom declaration, hidden selector, quotient representative, proposition-extensionality collapse, unledgered bridge, or supplied external content.

Lean4 (definition) `BEDC.Derived.NonAxiomBoundaryFormUp.NonAxiomBoundaryFormUp`

Theorem 1066.2 (Non-axiom boundary form NameCert obligations)**.** *For every accepted* NonAxiomBoundaryForm *carrier*

$$F = (B, K, S, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NonAxiomBoundaryForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, boundary-site exposure through* B*, seven-kind classifier exposure through* K*, source-surface exactness through* S*, gate locality through* G*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.NonAxiomBoundaryFormUp.NonAxiomBoundaryForm namecert obligations`

Proof. Project the carrier of Definition 1066.1. Its public rows are exactly B, K, S, G, H, C, P, N, so every NameCert obligation must be read from one of those coordinates.

The classifier obligation is the row K. It ranges only over the seven forms named in Definition 9395.2, while S records the corresponding source surface and G records the visible gate when the form is a socket, ledger, or externality-facing route.

Finally inspect H, C, P, N. These rows transport, replay, cite, and name the same finite classifier packet. They do not add an eighth form or a supplied object behind the boundary. □

Theorem 1066.3 (Non-axiom boundary form non-internalization)**.** *Every consumer of an accepted* $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ *packet must read the boundary site* B*, the classifier row* K*, and the source-surface row* S *before treating the dependency as admitted at a public interface. Changing* K *to an axiom row, deleting* S*, or bypassing the gate row* G *invalidates the public non-axiom route and supplies no internal BEDC theorem.*

Lean4 ✓ `BEDC.Derived.NonAxiomBoundaryFormUp.NonAxiomBoundaryForm noninternalization`

Proof. Start with the NameCert obligations above. They force the public route through B, K, S, G, H, C, P, N before any consumer receives the classifier result.

Use the external-supply discipline of chapter 9395. Each permitted boundary form keeps a source channel visible: constructor closure, definitional source, setup input, certificate row, GAP ledger, bridge separation, or apophatic socket.

Now compare the forbidden alternatives. An axiom row has no displayed source surface, deleting S removes the form-specific source channel, and bypassing G removes the gate for externality-facing forms. None of these routes is an accepted $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ packet.

Therefore the packet classifies boundary visibility without internalizing the requested supply. A consumer may cite the finite classifier and its source surface, but it receives no constructor, theorem, witness, quotient equality, or host principle not already displayed in the packet. □

The origin tag ai-composite records that this packet is a composite classifier over existing boundary-form surfaces. It therefore uses the carrier, obligation, and non-internalization rows above rather than atomic falsifiable-prediction or independence-witness rows.

Closure status for $\mathsf{NameCert}_{\mathsf{NonAxiomBoundaryForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$ packet is generated from one boundary site, one of seven non-axiom boundary-form rows, the corresponding source-surface row, an externality-facing gate row when applicable, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NonAxiomBoundaryForm}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1066.1, Definition 1066.2, and Definition 1066.3 bind the seven-form classifier, source-surface exactness, gate locality, provenance, and non-internalization boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NonAxiomBoundaryForm}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, complete discharge of any external supply, or an internal theorem corresponding to any classified socket.

Upgrade path: Public closure requires checked carrier alignment, seven-kind classifier exactness, source-surface stability, gate locality, and non-internalization for the displayed finite packet.

Chapter 1067

A Concrete Naming Certificate for $\mathsf{DigestFiberLedger}^{\uparrow}$

The $\mathsf{DigestFiberLedger}^{\uparrow}$ surface realizes the hash-like apophatic fixed-point discipline of chapter 9402 as a finite NameCert packet. It sits beside the hash certificate of chapter 242, the apophatic naming certificate of chapter 543, and the supply socket ledger of chapter 528. The packet keeps digest visibility, hidden fiber membership, GAP provenance, apophatic refusal, and consumer non-escape in one displayed carrier, so digest equality cannot silently become source equality.

Definition 1067.1 (DigestFiberLedger carrier)**.** *A* $\mathsf{DigestFiberLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (V, F, G, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the visible digest row, read through the local* $\mathsf{Hash}^{\uparrow}$ *certificate surface;*
2. *F is the fiber-membership row recording which source histories are carried behind the visible digest without identifying them with the digest;*
3. *G is the* GAP *provenance row that makes the fiber membership auditable;*
4. *A is the apophatic refusal row that blocks treating the far end of the fiber as a kernel object, source payload, or global digest;*
5. *H records componentwise* hsame *transports for the digest, fiber, GAP, refusal, continuation, provenance, and local certificate rows;*
6. *C records the displayed* Cont *routes by which consumers read V, F, G, and A without obtaining source equality or an internal* T *object;*
7. *P is the* Pkg *provenance over* $\mathsf{Hash}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}$ *row.*

The carrier contains no coordinate for source equality derived from digest equality, no hidden preimage selector, no global universe digest, and no internal object representing the apophatic far end.

Lean4 (definition) `BEDC.Derived.DigestFiberLedgerUp.DigestFiberLedgerUp`

Theorem 1067.2 (DigestFiberLedger NameCert obligations)**.** *For the carrier of Definition 1067.1, the local* $\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$D = (V, F, G, A, H, C, P, N),$$

visible digest exposure through V*, fiber-membership exposure through* F*, GAP provenance through* G*, apophatic refusal through* A*, componentwise* hsame *transport through* H*, displayed continuation replay through* C*, package provenance through* P*, local naming through* N*, and non-escape for any consumer that tries to identify digest equality with source equality without the fiber and GAP rows.*

Lean4 ✓ `BEDC.Derived.DigestFiberLedgerUp.DigestFiberLedgerNameCertObligations`

Proof. Project the carrier in Definition 1067.1. Its displayed coordinates are precisely the visible digest, fiber-membership, GAP provenance, apophatic refusal, transport, continuation, provenance, and local naming rows.

Read the three nearest sources through their displayed roles. The $\mathsf{Hash}^{\uparrow}$ certificate supplies the digest-facing row, $\mathsf{ApophaticName}^{\uparrow}$ supplies the refusal discipline for the far end of the fiber, and $\mathsf{SupplySocketLedger}^{\uparrow}$ supplies the socket ledger that keeps the missing source from becoming an internal theorem.

The NameCert obligations are therefore exhausted by the rows of D. Since the carrier has no coordinate for source equality from digest equality, no hidden preimage selector, and no object-level far end, any accepted consumer must retain F, G, A, H, C, P, and N when it cites the visible digest. □

Theorem 1067.3 (DigestFiberLedger non-escape)**.** *Every accepted* $\mathsf{DigestFiberLedger}^{\uparrow}$ *consumer of a visible digest row factors through the displayed fiber-membership row* F*, GAP provenance row* G*, apophatic refusal row* A*, componentwise transport row* H*, continuation routes* C*, package provenance* P*, and local naming row* N*. No consumer can use equality of visible digest rows as equality of source histories unless an additional exactness certificate is displayed outside the* $\mathsf{DigestFiberLedger}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.DigestFiberLedgerUp.DigestFiberLedgerNonEscape`

Proof. Start with an accepted packet $D = (V, F, G, A, H, C, P, N)$. The visible row V is only the digest-facing surface, while F and G record the fiber and its provenance.

Apply the NameCert obligations of Definition 1067.2. A consumer route may cite the digest only through the same carrier that displays fiber membership, GAP provenance, apophatic refusal, componentwise transport, continuation replay, package provenance, and local naming.

The absence claim follows from the carrier inventory. No coordinate in D turns digest equality into source equality, selects a preimage, exposes a global universe digest, or internalizes the apophatic far end; any such use requires a separate displayed exactness certificate. □

Theorem 1067.4 (DigestFiberLedger public export)**.** *Every public digest read exported by an accepted* $\mathsf{DigestFiberLedger}^{\uparrow}$ *carrier exposes exactly the visible digest row, fiber-membership row, GAP provenance row, apophatic refusal row, componentwise* hsame *row, displayed* Cont *row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}$ *row. It exports no source equality, hidden preimage selector, global universe digest, or internal far-end object.*

Lean4 ✓ `BEDC.Derived.DigestFiberLedgerUp.DigestFiberLedgerPublicExport`

Proof. Apply the carrier inventory of Definition 1067.1. The public read begins at the visible digest coordinate V, but the same carrier also displays F, G, A, H, C, P, N as mandatory rows.

Apply Definition 1067.2 and Definition 1067.3. Together they force every accepted consumer to retain the fiber, GAP, refusal, transport, continuation, provenance, and local naming rows when the visible digest is cited.

The negative export claim is again coordinate-level. Since D contains no source equality row, no preimage-selection row, no universe-digest row, and no internal far-end row, none of those objects can be exported by a public digest read. □

Theorem 1067.5 (DigestFiberLedger bridge schema boundary)**.** *Any bridge consumer for a public* $\mathsf{DigestFiberLedger}^{\uparrow}$ *export must preserve the eight displayed rows*

$$(V, F, G, A, H, C, P, N)$$

of Definition 1067.1. To claim that digest equality reflects source equality, the consumer must also add a separate standard-bridge row witnessing the exact source-equality route; the public export alone is not such a bridge.

Proof. Start with the carrier $D = (V, F, G, A, H, C, P, N)$. The NameCert obligations of Definition 1067.2 make these rows the complete local packet: visible digest, fiber membership, GAP provenance, apophatic refusal, transport, continuation, package provenance, and local naming.

Apply the non-escape theorem Definition 1067.3. It rules out reading equality of visible digest rows as equality of source histories unless an additional exactness certificate is displayed outside the packet.

Finally apply the public export theorem Definition 1067.4. The exported surface retains V, F, G, A, H, C, P, N and exports no source equality row, hidden preimage selector, global universe digest, or internal far-end object. Hence a standard bridge must be an extra row, not a consequence of public export itself. □

Falsifiable prediction: A consumer that identifies digest equality with source equality without the GAP fiber row refutes the DigestFiberLedgerUp packet.

Independence witness: Nearest siblings: hash, apophaticname, supplysocketledger, apophaticgatequestion, inscriptiongapledger. The carrier is not bijective to any sibling because it simultaneously stores digest visibility, fiber membership, GAP provenance, and apophatic non-internalization rows, while each sibling omits at least one of those obligations.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/DigestFiberLedgerUp/TasteGate.le` It must provide `Nontrivial DigestFiberLedgerUp` and `FieldFaithful DigestFiberLedgerUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{DigestFiberLedger}^{\uparrow}$ packet is generated from a visible digest row, hidden fiber-membership row, GAP provenance row, apophatic refusal row, componentwise hsame transport, displayed continuation routes, package provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.DigestFiberLedgerUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1067.1, Definition 1067.2, Definition 1067.3, Definition 1067.4, and the bridge-schema boundary of Definition 1067.5 bind the displayed digest, fiber, GAP, refusal, transport, continuation, provenance, local naming, public export, and bridge-schema packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DigestFiberLedger}^{\uparrow}}, \mathsf{encodedDef})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close source equality from digest equality, hidden preimage selection, global universe digest exposure, Lean verification, or bridge verification.

Upgrade path: Bridged closure requires a checked public schema bridge for the same digest, fiber, GAP, refusal, hsame, Cont, Pkg, and NameCert rows.

Chapter 1068

A Concrete Naming Certificate for $\mathsf{ApophaticFixedPoint}^{\uparrow}$

The $\mathsf{ApophaticFixedPoint}^{\uparrow}$ packet realizes the vision of chapter 9401 as a finite NameCert carrier. It consumes the apophatic name discipline of chapter 543, the local inscription surface of chapter 610, the visible socket-use ledger of chapter 1044, and the fixed witness discipline of chapter 556. The packet keeps far-end unification, local inscription readback, boundary noninternalization, and fixed-point ledger rows together without turning the far end into a kernel object.

Definition 1068.1 (ApophaticFixedPoint carrier)**.** *An* $\mathsf{ApophaticFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, I, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the forward-socket family row, recording the finite socket names, their requested supply shapes, visible use sites, and socket-use ledger links;*
2. *T is the apophatic far-end boundary row, naming the unique far end common to the displayed socket family without representing it as a kernel object;*
3. *I is the local inscription point row, recording where the common far-end boundary is read back on the local observation chain;*
4. *L is the boundary ledger row, recording the fixed-point ledger entries, refused internalizations, and accepted local readback events for the same socket family;*
5. *H records componentwise* hsame *transports for S, T, I, L, the continuation row, the package row, and the local naming row;*
6. *C records the displayed* Cont *routes by which consumers read the socket family, far-end boundary, local inscription point, ledger rows, provenance, and local certificate;*
7. *P is the* Pkg *provenance over* $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{SocketUseLedger}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, BHist, GAP, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticFixedPoint}^{\uparrow}}$ *row.*

The carrier contains exactly these rows. It has no coordinate for an internal object T*, no equality between the far-end boundary and a local state, no detached self-center object, no hidden socket selector, and no host fixed-point principle outside the displayed ledger.*

Theorem 1068.2 (ApophaticFixedPoint NameCert obligations)**.** *For the carrier of Definition 1068.1, the local* $\mathsf{NameCert}_{\mathsf{ApophaticFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (S, T, I, L, H, C, P, N),$$

forward-socket family exposure through S*, unique apophatic far-end naming through* T*, local inscription readback through* I*, boundary-ledger coverage through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointUp.ApophaticFixedPointTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1068.1. The tuple lists the socket family, far-end boundary, local inscription point, ledger, transport, continuation, provenance, and local naming rows.

Read the four nearest concrete sources through their displayed roles. The $\mathsf{ApophaticName}^{\uparrow}$ row supplies boundary naming, $\mathsf{InscriptionPoint}^{\uparrow}$ supplies the local inscription readback surface, $\mathsf{SocketUseLedger}^{\uparrow}$ supplies visible socket-use rows, and $\mathsf{GeneratorFixedPoint}^{\uparrow}$ supplies fixed-witness ledger discipline.

The NameCert obligations are exhausted by S, T, I, L, H, C, P, N. Since the carrier has no coordinate for an internal T object, detached self-center, hidden selector, or host fixed-point principle, every accepted read remains a finite certificate route. □

Theorem 1068.3 (ApophaticFixedPoint boundary noninternalization)**.** *For an accepted* $\mathsf{ApophaticFixedPoint}^{\uparrow}$ *packet, the far-end row* T *is readable only as the common apophatic boundary of the displayed forward-socket family* S*. A consumer may transport or replay this row through* H, C, P, N*, but it cannot read* T *as a kernel object, local state,* BHist *constructor, selected supply witness, object-level equality, or host fixed-point theorem.*

Proof. Start with the carrier inventory in Definition 1068.1. The far-end coordinate is the boundary row T, and its socket domain is the displayed family S.

The ledger row L records which attempted internalizations are refused, while H, C, P, N only transport, replay, package, and name the same finite rows. None of these rows adds a kernel carrier for the far end.

Apply the NameCert obligations of Definition 1068.2. They bind every read of T to S, T, I, L, H, C, P, N, so a far-end read that treats T as an object, state, constructor, selected witness, equality, or host theorem has no accepted route in the packet. □

Theorem 1068.4 (ApophaticFixedPoint local inscription readback)**.** *Every accepted local readback of an* $\mathsf{ApophaticFixedPoint}^{\uparrow}$ *packet factors through the route*

$$S \longmapsto T \longmapsto I \longmapsto L \longmapsto H, C, P, N.$$

The read first exposes the socket family, then names the unique apophatic far-end boundary, then records the local inscription point, and finally checks the boundary ledger, transport, continuation, provenance, and local NameCert *rows. It exports no local inscription read that bypasses the socket family or internalizes the far end.*

Proof. Project an accepted packet $F = (S, T, I, L, H, C, P, N)$. The local inscription row is I, but it is part of the same finite carrier as the socket family S and far-end boundary T.

Use Definition 1068.3. It blocks any read that turns T into an internal object before I is consumed as a local inscription point.

The remaining rows give the public readback surface. The ledger row L records accepted local readback and refused internalization attempts, while H, C, P, N preserve and expose only the displayed route. Thus every local inscription readback factors through the stated finite chain. □

Falsifiable prediction: Any accepted ApophaticFixedPointUp packet whose far-end row is read as a kernel object violates the boundary-noninternalization theorem.

Independence witness: apophaticname, inscriptionpoint, socketuseledger, generatorfixedpoint: none combines forward-socket unification, local inscription readback, boundary noninternalization, and fixed-point ledger rows in one packet.

The proposed Lean `TasteGate.lean` boundary is `lean4/BEDC/Derived/ApophaticFixedPointUp/TasteGate.` It must provide `Nontrivial ApophaticFixedPointUp` and `FieldFaithful ApophaticFixedPointUp` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticFixedPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticFixedPoint}^{\uparrow}$ packet is generated from a forward-socket family row, apophatic far-end boundary row, local inscription point row, boundary ledger row, componentwise transports, displayed continuations, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticFixedPoint}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1068.1, Definition 1068.2, Definition 1068.3, Definition 1068.4, the falsifiable prediction, the independence witness, and the sibling references to $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{SocketUseLedger}^{\uparrow}$, and $\mathsf{GeneratorFixedPoint}^{\uparrow}$ bind the finite obligation carrier over S, T, I, L, H, C, P, N with the bridge-facing TasteGate carrier alignment recorded on the formal axis.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticFixedPoint}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ApophaticFixedPointUp.ApophaticFixedPointTasteGate\_single\_carrier\_alignment`

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close an internal far-end object, host fixed-point theorem, hidden socket selection, object-level equality, standard bridge, or bridge checking.

Upgrade path: Bridge checking requires a standard bridge theorem for the same S, T, I, L, H, C, P, N packet.

Chapter 1069

A Concrete Naming Certificate for $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$

The $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ packet realizes the apophatic fixed-point and local-inscription reading of chapter 9401 as a finite NameCert surface. It consumes visible rows already separated by chapter 584, chapter 528, chapter 556, and chapter 242: the gate row refuses hidden supply, the socket ledger names the refused supply kind, the generator fixed-point row controls displayed fixed-output reads, and the hash row supplies digest-facing public readback. The new carrier is the fiber that forces those rows to be read in the apophatic order rather than as a direct digest-to-far-end identification.

Definition 1069.1 (Apophatic fixed-point fiber carrier)**.** *An* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, G, T, I, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest and socket readback row, carrying the public digest-facing observation together with the socket handle through which the packet is read;*
2. *S is the external-supply socket row supplied by the finite supply-ledger discipline;*
3. *G is the* GAP *fiber and provenance row, recording that every accepted public read passes through a boundary-fiber witness before it reaches any apophatic far-end name;*
4. *T is the apophatic far-end boundary-name row, naming the fixed far end only as boundary data and not as a kernel object;*
5. *I is the local inscription row, recording the chain-local site at which the boundary name is read inscription-side;*
6. *H records componentwise* hsame *transport for D, S, G, T, I, P, N and for the local packet shape;*
7. *C records the displayed* Cont *routes by which consumers read the digest, socket, GAP fiber, far-end boundary, inscription, provenance, and naming rows in order;*
8. *P records* Pkg *provenance over* BHist*, digest rows, externality gates, supply sockets, generator fixed-point witnesses, hash readback,* GAP *fibers, local inscriptions,* hsame*,* Cont*, and the local naming surface;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticFixedPointFiber}^{\uparrow}}$ *row.*

The carrier contains no row identifying D directly with T, no internal object for T*, and no hidden route from a public digest to an apophatic far end.*

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiberTasteGate single carrier alignment`

Theorem 1069.2 (Apophatic fixed-point fiber NameCert obligations)**.** *For the carrier of Definition 1069.1, the local* $\mathsf{NameCert}_{\mathsf{ApophaticFixedPointFiber}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (D, S, G, T, I, H, C, P, N),$$

digest-and-socket readback through D, external-supply exposure through S, GAP-fiber routing through G, boundary-name exposure through T, local-inscription exposure through I, componentwise hsame *transport through H, displayed continuation routing through C, package provenance through P, local naming through N, and refusal of any direct digest-to-T route that omits the GAP fiber row.*

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiber namecert obligations`

Proof. Carrier admission is projection from Definition 1069.1. The packet displays the digest, socket, GAP fiber, far-end boundary name, local inscription, transport, route, provenance, and local certificate rows as its complete inventory.

The readback obligations are ordered by the same inventory. A consumer first reads the digest and socket row D, then the external-supply socket row S, then the GAP fiber row G before the boundary-name row T and the inscription row I become available.

The transport, continuation, provenance, and naming rows preserve that order. Since the carrier has no coordinate for a direct digest-to-T identification, a route that bypasses G has no NameCert row to project and is rejected by the packet. □

Theorem 1069.3 (Apophatic fixed-point fiber non-escape)**.** *Every accepted* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *public read reaches the GAP fiber row G before it reaches the apophatic far-end boundary-name row T. A read that travels from the digest/socket row D to T without passing through G is outside the local certificate and refutes the packet.*

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiberCarrier gap first route nonescape`
Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiber nonescape`

Proof. Start from the finite carrier inventory. The only displayed path from public digest readback to the apophatic far-end boundary is the route family C applied to the rows D, S, G, T, I.

Apply the NameCert obligations. They make GAP-fiber routing a certificate row, not a prose convention, so any accepted read of T must preserve the intervening G coordinate and its provenance.

A direct route from D to T would require a carrier coordinate that the tuple does not contain. The local naming row N therefore has no certificate surface for such a route, and the attempted read is rejected rather than exported as an apophatic fixed-point fiber. □

Theorem 1069.4 (Apophatic fixed-point fiber sibling independence)**.** $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *is not bijective to the nearest sibling packets* $\mathsf{ExternalityGate}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*,* $\mathsf{GeneratorFixedPoint}^{\uparrow}$*, or* $\mathsf{Hash}^{\uparrow}$*. Each sibling omits at least one row required by the fiber: digest/socket readback, external-supply socket, GAP fiber, apophatic far-end boundary name, or local inscription. Conversely, the fiber does not replace those siblings; it binds their rows into a single ordered NameCert packet with the GAP fiber between the public digest and the apophatic far end.*

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiber independence witness`

Proof. Compare the carrier rows. $\mathsf{ExternalityGate}^{\uparrow}$ supplies a gate and refusal surface but lacks the digest/socket readback, generator fixed-point witness, hash digest, and local inscription rows required here.

$\mathsf{SupplySocketLedger}^{\uparrow}$ supplies the socket ledger but lacks the digest row, GAP-fiber order row, far-end boundary-name row, and local inscription row as one packet. $\mathsf{GeneratorFixedPoint}^{\uparrow}$ supplies a fixed-output witness but not the apophatic far-end boundary or inscription rows. $\mathsf{Hash}^{\uparrow}$ supplies digest-facing readback but neither the socket ledger nor the GAP fiber and far-end boundary.

Thus no bijection can preserve all carrier coordinates and NameCert obligations. Forgetting any of D, S, G, T, I changes the accepted public-read order, while adding such a row to a sibling changes that sibling's carrier rather than identifying it with the fiber. □

Falsifiable prediction: Every public read of an accepted ApophaticFixedPointFiberUp packet reaches the GAP fiber row before the apophatic far-end row; a direct digest-to-T route refutes the packet.

Independence witness: Nearest siblings: externalitygate, supplysocketledger, generatorfixed-point, and hash. None is bijective because each omits at least one of the digest, socket, GAP fiber, far-end boundary, or local-inscription rows, while the fiber requires all five rows together with componentwise hsame transport, displayed Cont routes, Pkg provenance, and the local NameCert row.

Theorem 1069.5 (Apophatic fixed-point fiber scoped route)**.** *Every accepted* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *public read is scoped by the concrete rows*

$$D, S, G, T, I, H, C, P, N$$

of Definition 1069.1 and by the sibling dependencies $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, *and* $\mathsf{Hash}^{\uparrow}$. *The GAP fiber row G lies between the digest/socket row D and the apophatic far-end boundary row T, so the local certificate exposes only the row-scoped route through $D, S, G, T, I, H, C, P, N$ and rejects any direct digest-to-far-end read.*

Proof. Use the carrier rows of Definition 1069.1. They give the accepted public read a finite inventory: digest/socket readback, external-supply socket, GAP fiber, far-end boundary name, local inscription, transport, continuation, provenance, and local naming.

Apply Definition 1069.2. The NameCert surface exports those rows in the displayed order, with the sibling packets supplying the externality gate, supply socket ledger, generator fixed-point, and hash-facing readback dependencies.

Apply Definition 1069.3. Since the only route from D to T passes through G, the GAP fiber is not optional boundary prose but a carrier row. Finally use Definition 1069.4: forgetting any of the sibling-dependent rows changes the packet rather than preserving an accepted scoped read. □

Lean4 ✓ `BEDC.Derived.ApophaticFixedPointFiberUp.ApophaticFixedPointFiber scoped route`

Theorem 1069.6 (Apophatic fixed-point fiber public export surface)**.** *Every accepted* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *packet exports exactly the public read surface*

$$D, S, G, T, I, H, C, P, N$$

with GAP-first ordering. The sibling dependency rows are read through $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, *and* $\mathsf{Hash}^{\uparrow}$, *and every digest-facing public read must pass through the GAP*

fiber before the apophatic far-end boundary is exposed. This is the public object surface of the packet, not a bridge credential.

Proof. Project the carrier of Definition 1069.1. The displayed rows are precisely $D, S, G, T, I, H, C, P, N$, and the carrier has no coordinate for a direct digest-to-far-end identification.

Apply the NameCert obligations of Definition 1069.2. They bind the digest/socket row, external-supply socket, GAP fiber, far-end boundary, local inscription, transport, replay, provenance, and local naming as the only exported rows.

Use the non-escape theorem Definition 1069.3 and the sibling-independence theorem Definition 1069.4. These keep $\mathsf{ExternalityGate}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{GeneratorFixedPoint}^{\uparrow}$, and $\mathsf{Hash}^{\uparrow}$ as visible dependencies rather than hidden bridge credentials.

Finally apply the scoped-route theorem Definition 1069.5. It forces the GAP-first public read order, so the theorem exports a public object surface only and does not certify an acceptance bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticFixedPointFiber}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ packet is generated from a digest/socket row, external-supply socket row, GAP fiber/provenance row, apophatic far-end boundary-name row, local inscription row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticFixedPointFiber}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1069.1, Definition 1069.2, Definition 1069.3, Definition 1069.4, and Definition 1069.5 bind the finite carrier, GAP-first read order, direct digest-to-far-end refusal, nearest-sibling independence surface, and scoped public-read route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticFixedPointFiber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a public bridge, an object-level value for T, a hidden external supply, or a global observer-substance theorem.

Upgrade path: Bridge-facing readback requires a displayed standard bridge over the same finite packet, without turning the TasteGate row into a bridge credential.

Chapter 1070

A Concrete Naming Certificate for $\mathsf{BoundaryGate}^{\uparrow}$

The $\mathsf{BoundaryGate}^{\uparrow}$ packet realizes the boundary-gate construction of chapter 9395 as a finite NameCert surface. It sits beside the external-supply boundary of chapter 1065, the socket ledger of chapter 528, the supply-kind router of chapter 666, the audit membrane of chapter 524, and the audit-gate boundary of chapter 526. Those siblings isolate boundary records, socket ledgers, kind routing, refusal replay, or audit inventory; this packet binds the gate question and audit verdict to the visible boundary form and site route in one carrier.

Definition 1070.1 (BoundaryGate carrier)**.** *A* $\mathsf{BoundaryGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, Q, V, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-form row, recording the accepted external-supply boundary form being tested;*
2. *Q is the gate-question row, recording the finite question asked of that boundary form;*
3. *V is the audit-verdict row, recording the pass-or-refuse result for the question Q at boundary form B;*
4. *S is the visible site route, recording the socket, marker, dependency, certificate row, or import edge at which the boundary is read;*
5. *H records componentwise* hsame *transports for B, Q, V, S together with continuation, provenance, and naming rows;*
6. *C records the displayed* Cont *routes by which consumers inspect the boundary form, gate question, audit verdict, and visible site route;*
7. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{SupplyKindRouter}^{\uparrow}$, $\mathsf{AuditMembrane}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryGate}^{\uparrow}}$ *row.*

The carrier exports no hidden boundary content, no verdict detached from the boundary form, no route that treats refused supply as substrate theorem content, and no host-side permission to answer the gate without displaying B, Q, V, S, H, C, P, N.

Theorem 1070.2 (BoundaryGate NameCert obligations)**.** *For the carrier of Definition 1070.1, the local* $\mathsf{NameCert}_{\mathsf{BoundaryGate}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$G = (B, Q, V, S, H, C, P, N),$$

boundary-form exposure through B*, gate-question exposure through* Q*, audit-verdict exposure through* V*, visible-site routing through* S*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Consequently, an accepted $\mathsf{BoundaryGate}^{\uparrow}$ *packet cannot expose a verdict without the boundary form it judges, cannot expose a boundary form without the gate question and site route, and cannot route refused boundary content into substrate theorem content.*

Lean4 ✓ `BEDC.Derived.BoundaryGateUp.BoundaryGateTasteGate single carrier alignment`

Proof. Project the displayed carrier. The accepted packet lists B, Q, V, S, H, C, P, N as its full inventory, so the local naming certificate has no coordinate from which to introduce a second boundary form, gate question, verdict, site route, transport, continuation, provenance, or local name.

Read the gate-facing rows first. The boundary form B names what kind of external-supply boundary is being tested, Q names the finite question asked of that form, and V records the audit verdict for that question.

Next read the routing rows. The site route S makes the tested location visible, while H and C transport and replay only the same displayed rows.

Finally inspect P and N. Provenance ties the packet to its sibling surfaces, and the local name binds the finite inventory. Since the carrier has no coordinate for hidden boundary content or detached verdicts, every accepted read remains inside the displayed NameCert surface. □

Theorem 1070.3 (BoundaryGate non-internalization)**.** *Every accepted* $\mathsf{BoundaryGate}^{\uparrow}$ *packet preserves the external-supply boundary it judges. A downstream consumer may read* B, Q, V, S, H, C, P, N *as a finite audit surface for the boundary gate, but it may not read from the packet an internal witness of the requested supply, a replacement theorem for refused content, an off-site verdict, a selected boundary representative, a quotient collapse, a hidden host equality, or a bridge row.*

Lean4 ✓ `BEDC.Derived.BoundaryGateUp.BoundaryGate noninternalization`

Proof. Begin with the boundary-form and gate-question rows. The pair B, Q fixes the boundary and the finite question as displayed certificate content rather than as an implicit host-side convention.

Use the audit-verdict row V together with the visible site route S. A verdict is accepted only as the result of testing the displayed boundary form at the displayed site; it is not an independent source of requested supply.

The transport and replay rows H and C preserve that same packet under componentwise movement and consumer access. They do not add a selector, quotient representative, theorem object, host equality, or bridge coordinate.

The provenance and local naming rows P, N bind the packet to $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{SupplyKindRouter}^{\uparrow}$, $\mathsf{AuditMembrane}^{\uparrow}$, and $\mathsf{AuditGateBoundary}^{\uparrow}$ as sibling sources. Since none of the displayed rows supplies the refused content itself, the accepted boundary gate keeps the judged supply external. □

Theorem 1070.4 (BoundaryGate carrier admission)**.** *Every admitted* $\mathsf{BoundaryGate}^{\uparrow}$ *carrier is admitted only as a finite* BHist *packet*

$$G = (B, Q, V, S, H, C, P, N)$$

whose rows display the boundary form B*, gate question* Q*, audit verdict* V*, visible site route* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. In particular, the verdict* V *cannot be admitted apart from the boundary form* B *and visible site route* S*.*

Lean4 ✓ `BEDC.Derived.BoundaryGateUp.BoundaryGateCarrierAdmission`

Proof. Project the carrier of Definition 1070.1. The row inventory is exactly B, Q, V, S, H, C, P, N, so admission begins with the displayed boundary form, question, verdict, and site route.

The structural rows do not add a detached verdict. The hsame row H transports the same displayed data, the Cont row C replays access through the same packet, and P, N bind provenance and local naming to that inventory.

Thus carrier admission is a row projection from the displayed packet. Since no carrier coordinate stores V without B and S, an admitted verdict remains attached to the boundary form and visible site route. □

Theorem 1070.5 (BoundaryGate classifier stability)**.** *For any accepted* $\mathsf{BoundaryGate}^{\uparrow}$ *packet* $G = (B, Q, V, S, H, C, P, N)$*, the classifier is stable only under componentwise* hsame *transport of the displayed rows* B, Q, V, S, H, C, P, N *and displayed* Cont *replay through the same* Pkg *and* NameCert *surface. A transported verdict remains the verdict for boundary form* B *at visible site route* S*; it cannot become a route-free audit result.*

Proof. Start with the classifier rows displayed by the carrier. The boundary form, question, verdict, and site route are compared as named coordinates of one finite packet.

Apply the transport row. Componentwise hsame preserves each displayed coordinate as the same row for classifier purposes, so a stable read carries B, Q, V, S together rather than moving V alone.

Finally use the replay and provenance rows. The Cont row C, Pkg row P, and local NameCert row N replay only the same packet, so classifier stability cannot detach the verdict from the boundary form or visible site route. □

Lean4 ✓ `BEDC.Derived.BoundaryGateUp.BoundaryGateClassifierStability`

Theorem 1070.6 (BoundaryGate ledger nonescape)**.** *Every ledger read from an accepted* $\mathsf{BoundaryGate}^{\uparrow}$ *carrier* $G = (B, Q, V, S, H, C, P, N)$ *must expose the boundary form* B*, gate question* Q*, audit verdict* V*, visible site route* S*,* BHist *transport row* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N *together. No ledger consumer can export* V *as supplied content, as a hidden boundary witness, or as a verdict detached from* B *and* S*.*

Proof. Read the ledger-facing packet as displayed finite data. The rows B, Q, V, S identify what is judged, which finite question is asked, what verdict is recorded, and where the boundary is read.

Now inspect the structural route. The BHist transport row H and Cont replay row C keep ledger access inside the same carrier, while P and N record provenance and local naming.

Because the carrier contains no extra row for supplied content, hidden boundary witnesses, or route-free verdicts, the ledger cannot export such data. The verdict remains a visible row of the boundary gate rather than an escaping object. □

Theorem 1070.7 (BoundaryGate verdict-site route stability)**.** *For every accepted* BoundaryGate$^{\uparrow}$ *carrier* $G = (B, Q, V, S, H, C, P, N)$*, the audit verdict row* V *remains attached to both the boundary-form row* B *and the visible-site route* S *under componentwise* hsame *transport and displayed* Cont *replay. In particular, no stable replay can read* V *as a route-free verdict, as supplied theorem content, or as a boundary decision detached from the site where* B *is inspected.*

Proof. Start from the carrier of Definition 1070.1. Its displayed packet has a single verdict coordinate V, and that coordinate is listed together with the boundary form B, gate question Q, visible site route S, transport row H, replay row C, provenance row P, and local name N.

Apply Definition 1070.5. Componentwise hsame transport preserves the classifier only by moving the displayed rows together, so a transported read of V still carries the same boundary form and visible site route.

Finally apply Definition 1070.6. Ledger consumption cannot export V as hidden supply or as a detached verdict. The displayed Cont replay therefore returns the verdict only through the same boundary-form and site-route rows that admitted it. □

Falsifiable prediction: Any accepted BoundaryGateUp packet that admits boundary content without exposing both the boundary-form row and the audit-verdict row is rejected by the NameCert surface.

Independence witness: Nearest siblings: external-supply-boundary, supply-socket-ledger, supply-kind-router, audit-membrane, audit-gate-boundary. None is bijective because BoundaryGateUp carries the gate question and verdict rows together with the visible boundary form and site route, while the siblings isolate boundary, socket kind, kind routing, refusal replay, or gate-boundary inventory.

The Lean follow-up must provide Nontrivial BoundaryGate$^{\uparrow}$ and FieldFaithful BoundaryGate$^{\uparrow}$ in lean4/BEDC/Derived/BoundaryGateUp/TasteGate.lean before the taste-gate target BEDC.Derived.BoundaryGateUp.T can be used as a checked correspondence.

Closure status for NameCert$_{\text{BoundaryGate}^{\uparrow}}$**.**

Constructive story: A BoundaryGate$^{\uparrow}$ packet is generated from a boundary-form row, gate-question row, audit-verdict row, visible site route, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The carrier asks a finite question of a visible external-supply boundary and records the verdict only with that boundary and site still present. The classifier therefore admits a gate read by projecting B, Q, V, S, H, C, P, N together: transport preserves the same boundary form, and continuation replay prevents the verdict from becoming detached supply content.

Theory closure: TheoryGate(NameCert$_{\text{BoundaryGate}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1070.1, Definition 1070.2, Definition 1070.4, Definition 1070.5, Definition 1070.6, and Definition 1070.7 bind the exact boundary, verdict, site-route, and TasteGate scope over B, Q, V, S, H, C, P, N, BHist, hsame, Cont, Pkg, NameCert, and the sibling external-supply boundary packets.

Formal verification: FormalStatus(NameCert$_{\text{BoundaryGate}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BoundaryGateUp.BoundaryGateCarrierAdmission`

Bridge status: none

Origin: ai

Not claimed: It does not claim hidden boundary content, detached verdicts, supplied theorem content, quotient representatives, host equality, or bridge rows.

Upgrade path: Public export requires exposing the same B, Q, V, S, H, C, P, N scoped gate through a consumer-facing surface without adding hidden boundary content, detached verdicts, supplied theorem content, quotient representatives, host equality, or bridge rows.

Chapter 1071

A Concrete Naming Certificate for $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$

The $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ packet realizes the local self-center reading of chapter 9401 as a finite NameCert seal. It consumes the local inscription row of chapter 610, the apophatic socket row of chapter 536, the supply inscription surface of chapter 604, the observer-history identity discipline of chapter 479, and the public event route of chapter 519. The seal binds those rows without identifying a local self-center with the apophatic far end.

Definition 1071.1 (Local self-center inscription seal carrier)**.** *A* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H, I, S, A, V, R, C, P, N)$$

with the following displayed rows:

1. *H is the local history row, read through the observer-history identity surface and carrying no subject-substance coordinate;*
2. *I is the inscription point row, naming the local site at which the chain's accumulation event is read inscription-side;*
3. *S is the self-center reading row, recording the aspectual read of the same local event as self-center data;*
4. *A is the apophatic socket row, naming the boundary-facing far-end route without representing that far end as a kernel object;*
5. *V is the nonidentity verdict row, recording that the self-center reading row S and the apophatic socket far-end row A are not object-level equal rows;*
6. *R records componentwise* hsame *transport for H, I, S, A, V and for the local naming row;*
7. *C records the displayed* Cont *routes by which consumers read the local history, inscription point, self-center row, apophatic socket row, verdict row, provenance, and local certificate;*
8. *P is the* Pkg *provenance over* $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an internal object T*, no subject behind the displayed history row, no equality between* S *and* A*, and no hidden route that reads the apophatic far end as a local state.*

Lean4 (definition) `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealUp`

Theorem 1071.2 (Local self-center inscription seal NameCert obligations)**.** *For the carrier of Definition 1071.1, the local* $\mathsf{NameCert}_{\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (H, I, S, A, V, R, C, P, N),$$

local history exposure through H*, inscription-point locality through* I*, self-center reading through* S*, apophatic socket exposure through* A*, nonidentity verdict through* V*, componentwise* hsame *transport through* R*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealUp taste gate boundary`

Proof. Carrier admission is projection from Definition 1071.1. The tuple displays the local history, inscription point, self-center read, apophatic socket, nonidentity verdict, transport, route, provenance, and naming rows.

The sibling chapters supply only their named roles. $\mathsf{InscriptionPoint}^{\uparrow}$ supplies the local inscription row, $\mathsf{ApophaticSocket}^{\uparrow}$ supplies the boundary-facing socket row, $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$ supplies the supply-inscription provenance, $\mathsf{ObserverHistIdentity}^{\uparrow}$ supplies the no-subject history discipline, and $\mathsf{InscriptionEvent}^{\uparrow}$ supplies the public event route.

The local NameCert row is exhausted by these displayed rows. Since no coordinate identifies S with A, every accepted read of the self-center remains an inscription-side local row rather than a far-end object. □

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealTasteGate single carrier alignment`

Theorem 1071.3 (Local self-center inscription seal nonidentity boundary)**.** *For every accepted* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ *carrier* $L = (H, I, S, A, V, R, C, P, N)$*, a consumer may read* S *only as the local self-center inscription row and may read* A *only as the apophatic socket row. Any packet that identifies* S *with the far-end row of* A *fails the nonidentity verdict row* V *and is not accepted as this seal.*

Proof. Project the accepted carrier. The self-center row and apophatic socket row occur as separate coordinates S and A, and the verdict row V is the displayed boundary between them.

The transport and continuation rows preserve that separation. They replay S and A through R and C but do not add an equality coordinate or a kernel value for the far end.

Apply the NameCert obligations of Definition 1071.2. A packet that identifies the local self-center with the apophatic far end omits the required verdict, so the local name row has no accepted certificate surface for it. □

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealSelfCenter nonidentity boundary`

Theorem 1071.4 (Local self-center inscription seal sibling separation)**.** LocalSelfCenterInscriptionSeal$^{\uparrow}$ *is not bijective, under NameCert-row preservation, to* InscriptionPoint$^{\uparrow}$, ApophaticSocket$^{\uparrow}$, TranscendentalSupplyInsc ObserverHistIdentity$^{\uparrow}$, *or* InscriptionEvent$^{\uparrow}$. *None of those siblings carries the simultaneous local history, inscription-point, self-center reading, apophatic socket, and nonidentity verdict rows required by Definition 1071.1.*

Proof. Compare the carrier rows. InscriptionPoint$^{\uparrow}$ records a local inscription surface but not the self-center verdict against an apophatic far end. ApophaticSocket$^{\uparrow}$ records the boundary-facing socket but not the local self-center reading.

TranscendentalSupplyInscription$^{\uparrow}$ supplies supply-inscription provenance, ObserverHistIdentity$^{\uparrow}$ supplies no-subject history identity, and InscriptionEvent$^{\uparrow}$ supplies public event routing. Each omits at least one of the displayed rows H, I, S, A, V.

Thus a row-preserving bijection would have to erase or synthesize part of the seal. Erasure loses an obligation, while synthesis changes the sibling carrier; neither operation preserves the local NameCert surface. □

Theorem 1071.5 (Local self-center inscription seal transport exactness)**.** *For every accepted* LocalSelfCenterInscriptionSeal$^{\uparrow}$ *carrier* $L = (H, I, S, A, V, R, C, P, N)$*, the transport row* R *is componentwise* hsame *transport for the local history row* H*, inscription-point row* I*, self-center reading row* S*, apophatic socket row* A*, nonidentity verdict row* V*, displayed route row* C*, provenance row* P*, and local naming row* N*. No accepted transport identifies* S *with* A *or replaces the verdict row* V *by a hidden equality coordinate.*

Proof. Project the accepted carrier. The displayed tuple gives $H, I, S, A, V, R, C, P, N$ as the only seal rows, and R is the named transport row for that tuple.

Apply the NameCert obligations of Definition 1071.2. They require componentwise transport for the carrier rows and local name rather than a separate equality source.

Finally use the nonidentity boundary of Definition 1071.3. Since an identification of S with the far-end row of A fails V, exact transport preserves the verdict boundary instead of collapsing it. □

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealCarrier transport exactness`

Theorem 1071.6 (Local self-center inscription seal provenance exhaustion)**.** *For every accepted* LocalSelfCenterInscriptionSeal$^{\uparrow}$ *carrier* $L = (H, I, S, A, V, R, C, P, N)$*, the provenance row* P *is exhausted by* InscriptionPoint$^{\uparrow}$, ApophaticSocket$^{\uparrow}$, TranscendentalSupplyInscription$^{\uparrow}$, ObserverHistIdentity$^{\uparrow}$, InscriptionEvent$^{\uparrow}$, *and the kernel rows* BHist, hsame, Cont, Pkg, *and* NameCert. *It exports no subject behind* H*, no internal object for the apophatic far end, and no sibling row outside those named surfaces.*

Proof. Read P from the carrier definition. Its provenance list is exactly the five sibling surfaces together with the displayed kernel rows.

Use the sibling separation theorem Definition 1071.4. Each sibling supplies only its named role and omits at least one of the simultaneous local history, inscription-point, self-center reading, apophatic socket, and verdict rows.

The provenance row therefore accounts for all admissible inputs to the seal without adding an extra subject, far-end object, or hidden sibling coordinate. □

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.LocalSelfCenterInscriptionSealProvenance row nonescape`

Theorem 1071.7 (Local self-center inscription seal carrier transport)**.** *Any accepted carrier transport of a* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ *packet preserves the nonidentity verdict row. If* $L = (H, I, S, A, V, R, C, P, N)$ *is accepted and the carrier is transported through the displayed* hsame *row* R*, then the transported packet still reads* S *as the local self-center row,* A *as the apophatic socket row, and* V *as the verdict refusing their object-level identification.*

Proof. Begin with transport exactness. The row R transports the displayed components of L componentwise, including S, A, and V.

Apply the nonidentity boundary. A packet that identifies the local self-center row with the apophatic far-end row is not accepted as this seal.

Thus an accepted carrier transport must preserve the verdict row. Otherwise the transported packet would erase the boundary required by the NameCert obligations and would fail acceptance. □

Falsifiable prediction: A packet that identifies the local self-center row with the apophatic far-end row fails the nonidentity verdict row.

Independence witness: InscriptionPoint, ApophaticSocket, TranscendentalSupplyInscription, ObserverHistIdentity, and InscriptionEvent: none carries the simultaneous self-center reading, apophatic socket, and nonidentity verdict rows.

The later Lean TasteGate file must provide explicit `Nontrivial LocalSelfCenterInscriptionSealUp` and `FieldFaithful LocalSelfCenterInscriptionSealUp` instances in `lean4/BEDC/Derived/LocalSelfCenter`

Closure status for $\mathsf{NameCert}_{\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ packet is generated from local history, inscription point, self-center reading, apophatic socket, nonidentity verdict, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1071.1, Definition 1071.2, Definition 1071.3, Definition 1071.4, Definition 1071.5, Definition 1071.6, Definition 1071.7, and

Lean4 ✓ `BEDC.Derived.LocalSelfCenterInscriptionSealUp.taste gate`

bind the carrier, NameCert obligations, nonidentity boundary, sibling separation, transport, provenance, carrier-transport, and TasteGate rows for the finite local seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, a standard bridge, identity with the apophatic far end, or any local-state reading of the apophatic far end.

Upgrade path: No further paper-theory upgrade remains for the finite local self-center seal; later work is verification and bridge checking only.

Chapter 1072

A Concrete Naming Certificate for $\mathsf{ExternalSupplySocket}^{\uparrow}$

The $\mathsf{ExternalSupplySocket}^{\uparrow}$ packet realizes the socket side of chapter 9395 as a finite NameCert carrier. It sits between the boundary packet of chapter 1065, the apophatic naming surface of chapter 543, the kind router of chapter 666, and the classifier packet of chapter 1058. The socket packet keeps the requested supply visible as a refused boundary row while preserving the ledger rows that downstream audit consumers read.

Definition 1072.1 (ExternalSupplySocket carrier)**.** *An* $\mathsf{ExternalSupplySocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (B, A, K, R, F, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$ *row naming the boundary site and requested supply shape;*
2. *A is the* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the supply position without supplying the far end;*
3. *K is the* $\mathsf{SocketKindClassifier}^{\uparrow}$ *row naming the socket kind that governs the request;*
4. *R is the* $\mathsf{SupplyKindRouter}^{\uparrow}$ *row routing the request through the chosen kind and requested-shape surface;*
5. *F is the refusal-gate row recording that the requested supply is not available as substrate truth;*
6. *L is the socket ledger row recording the GAP pressure, visible use site, and refusal readback for the same request;*
7. *H records componentwise* hsame *transport for B, A, K, R, F, L;*
8. *C records displayed* Cont *routes from the boundary row to the apophatic name, classifier, router, refusal gate, and ledger rows;*
9. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{SupplyKindRouter}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, *the socket ledger,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplySocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for the requested outside supply, no hidden selector, no quotient representative, no proposition-equality token, no host equality, and no theorem asserting that the requested supply has become internal.

Lean4 ✓ BEDC.Derived.ExternalSupplySocketUp.ExternalSupplySocketTasteGate single carrier alignment

Theorem 1072.2 (ExternalSupplySocket NameCert surface)**.** *For the carrier of Definition 1072.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplySocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$X = (B, A, K, R, F, L, H, C, P, N),$$

boundary exposure through B*, apophatic-name exposure through* A*, socket-kind classification through* K*, kind routing through* R*, refusal-gate readback through* F*, socket-ledger readback through* L*, componentwise transport through* H*, displayed continuation through* C*, provenance through* P*, and local naming through* N*.*

Consequently every accepted public read of the socket must pass through the boundary row, name row, classifier row, router row, refusal-gate row, and ledger row before it may cite the socket as an external-supply boundary.

Proof. Project the displayed carrier. The packet lists $B, A, K, R, F, L, H, C, P, N$, so the certificate has no coordinate from which to manufacture another boundary name, kind classifier, refusal gate, ledger, transport, route, provenance, or local name.

Read the dependency rows in order. The boundary row B gives the requested supply shape, the apophatic row A names the same unsupplied position, the classifier row K fixes the socket kind, and the router row R places that kind before the requested-shape and refusal reads.

Finally read F, L, H, C, P, N. The refusal gate and ledger record the non-internalizing readback, while transport, continuation, provenance, and local naming preserve the same finite packet. Thus the NameCert surface is exhausted by the displayed rows. □

Theorem 1072.3 (ExternalSupplySocket noninternalization)**.** *Every accepted* $\mathsf{ExternalSupplySocket}^{\uparrow}$ *packet keeps its requested supply external. A consumer may read the boundary row, apophatic name, socket-kind classifier, kind router, refusal gate, socket ledger, transport, route, provenance, and local name, but it receives no supplied witness, quotient representative, proposition-equality token, host equality, hidden selector, unrestricted induction principle, or theorem turning the requested supply into substrate truth.*

Proof. Start with the boundary packet of chapter 1065. It records the requested supply and noninternalization row before any socket consumer reads the request.

Use the apophatic name and classifier rows next. The apophatic surface names the external position without a far-end object, and the socket-kind classifier fixes the local kind row that must be routed before refusal data is available.

Apply the kind router and the refusal-gate row. The router forces the requested-shape read through the displayed kind route, while F records that this route ends in refusal rather than supplied content.

Finally use the ledger and structural rows. The row L records GAP pressure and refusal readback, and H, C, P, N only transport, replay, package, and name the same packet. Since no carrier coordinate stores a supplied witness or host-level replacement, the requested supply remains external. □

Falsifiable prediction: A packet without the refusal-gate row cannot satisfy the noninternalization theorem.

Independence witness: Nearest siblings: externalsupplyboundary, apophaticname, supplykindrouter, socketkindclassifier, and supplysocketledger. None is bijective to $\mathsf{ExternalSupplySocket}^{\uparrow}$

because the socket packet simultaneously carries boundary naming, apophatic naming, kind classification, kind routing, refusal-gate readback, and socket-ledger rows; each sibling omits at least one of those rows.

The Lean follow-up must provide explicit Nontrivial ExternalSupplySocket$^{\uparrow}$ and FieldFaithful ExternalSupplySocket instances in Derived/ExternalSupplySocketUp/TasteGate.lean before the alignment theorem BEDC.Derived.ExternalSu can be cited by a checked paper marker.

Closure status for NameCert$_{\text{ExternalSupplySocket}^{\uparrow}}$.

Constructive story: An ExternalSupplySocket$^{\uparrow}$ packet is generated from an external-supply boundary row, apophatic name row, socket-kind classifier, supply-kind router, refusal-gate row, socket ledger, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The carrier reads the requested supply as a visible boundary, names the unsupplied position, classifies its socket kind, routes the request, and records refusal plus ledger residue. The support rows keep the same site visible under transport and replay. The classifier preserves externality: every accepted consumer sees the socket through the finite boundary-name-kind-route-refusal-ledger sequence.

Theory closure: TheoryGate(NameCert$_{\text{ExternalSupplySocket}^{\uparrow}}$, seed)

Scope closed: This chapter binds the external supply socket carrier, NameCert surface, noninternalization theorem, prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{ExternalSupplySocket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim supplied witnesses, hidden selectors, quotient representatives, proposition-equality tokens, host equality, or internal theorem content.

Upgrade path: To reach obligationClosure, formalize the socket-kind, router, refusal-gate, and ledger rows as a checked noninternalizing carrier.

Chapter 1073

A Concrete Naming Certificate for $\mathsf{NonAxiomAdmission}^{\uparrow}$

The $\mathsf{NonAxiomAdmission}^{\uparrow}$ packet is the concrete admission classifier for the external-supply discipline of chapter 9395. It sits after the boundary packet of chapter 1065, the socket ledger of chapter 528, the apophatic socket surface of chapter 536, the selector ledger of chapter 494, and the admission gate of chapter 545. Those siblings record particular boundary, socket, ledger, or chapter-admission rows; this packet classifies the decision surface that accepts a proposed supply position without turning it into an axiom.

Definition 1073.1 (NonAxiomAdmission carrier)**.** *A* $\mathsf{NonAxiomAdmission}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, F, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the proposed supply position presented for admission;*
2. *F is the accepted non-axiom boundary form, chosen from inductive constructor, definitional expansion, setup-field obligation,* $\mathsf{NameCert}$ *row,* GAP *memory, bridge slot, or socket boundary;*
3. *W is the displayed witness route showing how the proposed position is consumed through the selected boundary form;*
4. *H stores componentwise* hsame *transport for the route X, F, W;*
5. *C stores displayed* Cont *replay for the same admitted route;*
6. *P stores* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, $\mathsf{TasteGateAdmission}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{NonAxiomAdmission}^{\uparrow}}$ *row.*

The carrier has no coordinate that asserts the supplied content as substrate truth, no hidden chooser, no quotient representative, no proposition-equality collapse, and no global frame from which consumers may recover the proposed supply apart from the displayed rows.

Lean4 (definition) `BEDC.Derived.NonAxiomAdmissionUp.taste gate`

Theorem 1073.2 (NonAxiomAdmission NameCert obligations)**.** *For the carrier of Definition 1073.1, the local* $\mathsf{NameCert}_{\mathsf{NonAxiomAdmission}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (X, F, W, H, C, P, N)$, proposed-position exposure through X, accepted-form exactness through F, witness-route exposure*

through W, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Equivalently, every accepted admission decision exposes which supply position was proposed, which non-axiom boundary form admits it, which finite route witnesses the admission, and which transport, replay, provenance, and naming rows preserve the same route. It exposes no consumer route that treats the supplied content itself as substrate truth.

Proof. Project the displayed carrier. The accepted packet lists exactly X, F, W, H, C, P, N, so the local certificate cannot introduce a second proposed supply position, accepted-form row, witness route, transport row, replay route, provenance row, or local name.

Read the admission rows in order. The row X names the proposed supply position, F classifies the boundary form as inductive, definitional, setup-field, NameCert, GAP, bridge, or socket data, and W displays the finite route by which that form is consumed.

Finally inspect H, C, P, N. These rows only transport, replay, package, and name the same admitted route. Since none of them contains supplied content as substrate truth, the local NameCert obligations are exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.NonAxiomAdmissionCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.NonAxiomAdmissionNamecertObligations`

Theorem 1073.3 (NonAxiomAdmission boundary exhaustion)**.** *Every accepted* $\mathsf{NonAxiomAdmission}^{\uparrow}$ *packet classifies a proposed supply position into a displayed non-axiom boundary form. A downstream consumer may read only X, F, W, H, C, P, N and therefore cannot obtain from the packet an axiom, hidden chooser, quotient representative, proposition-equality token, global frame, or second route that treats the supplied content as substrate truth.*

Proof. Start with the boundary-form coordinate F. By the carrier definition, F ranges only over the displayed forms: inductive constructor, definitional expansion, setup-field obligation, NameCert row, GAP memory, bridge slot, or socket boundary. None is an axiom coordinate.

Use the witness route W together with the transport and replay rows H, C. They show how the selected boundary form is consumed and preserved, but they do not add a hidden supply source or a second consumer edge.

Now read the provenance and naming rows P, N. The provenance row ties the admission to $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, $\mathsf{TasteGateAdmission}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert data, while N names the local packet. Since consumers can read only X, F, W, H, C, P, N, all excluded coordinates are absent from the public route. □

Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.NonAxiomAdmissionCarrier boundary exhaustion`

Theorem 1073.4 (NonAxiomAdmission witness-route nonescape)**.** *If an accepted* $\mathsf{NonAxiomAdmission}^{\uparrow}$ *packet exposes witness route W, then every consumer route that preserves the local* $\mathsf{NameCert}_{\mathsf{NonAxiomAdmission}^{\uparrow}}$ *row must read the same displayed coordinates*

$$X, F, W, H, C, P, N.$$

Changing the proposed supply position, accepted non-axiom form, witness route, componentwise transport, replay, provenance, or local name without changing one displayed coordinate is unreadable to the packet. The carrier therefore admits no consumer edge that treats altered supplied content as the same local admission while leaving X, F, W, H, C, P, N fixed.

Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.NonAxiomAdmissionCarrier witness route nonescape`

Proof. Project the carrier of Definition 1073.1. The witness route W is not a detachable annotation; it is one of the displayed coordinates transported by H, replayed by C, sourced by P, and named by N.

Apply Definition 1073.3. A consumer may read only X, F, W, H, C, P, N, so any change to the supplied content must appear as a change in the proposed position, accepted form, witness route, transport, replay, provenance, or name.

If all seven displayed coordinates are held fixed, the local NameCert has no remaining row from which a different supplied-content route can be read. Thus the witness route is nonescaping for every row-preserving consumer of the packet. □

Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.NonAxiomAdmissionWitnessRoute nonescape`

Theorem 1073.5 (NonAxiomAdmission scoped primitive route)**.** *Every accepted* $\mathsf{NonAxiomAdmission}^{\uparrow}$ *packet binds its displayed rows*

$$X, F, W, H, C, P, N$$

to exact BEDC primitives: X *is a proposed-supply* BHist *row,* F *is the accepted non-axiom form row,* W *is the witness-route row,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{NonAxiomAdmission}^{\uparrow}}$ *row. No scoped consumer can read an admitted position without preserving those seven primitive rows.*

Proof. Start with the NameCert obligations of Definition 1073.2. They identify the local certificate surface with exactly the proposed-position, accepted-form, witness-route, transport, replay, provenance, and naming rows.

Apply boundary exhaustion from Definition 1073.3. The accepted form F is a displayed non-axiom boundary form, and no consumer route can recover an axiom coordinate, hidden chooser, quotient representative, proposition-equality token, or global frame from the packet.

Finally use witness-route nonescape from Definition 1073.4. Holding X, F, W, H, C, P, N fixed leaves no remaining row from which a different supplied-content route can be read. Thus the scoped route is exactly the listed BEDC primitive route. □

Theorem 1073.6 (NonAxiomAdmission scoped TasteGate route)**.** *Every accepted* $\mathsf{NonAxiomAdmission}^{\uparrow}$ *decision is scoped by the displayed rows* X, F, W, H, C, P, N*, the non-axiom boundary siblings,* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the checked TasteGate route for* BEDC.Derived.NonAxiomAdmissionUp.taste *No scoped consumer may treat the proposed supply position as substrate truth unless the same accepted non-axiom form, witness route, transport, replay, provenance, local name, and TasteGate evidence remain visible.*

Proof. First compose the local certificate surface from Definition 1073.2. It identifies the accepted decision with the rows X, F, W, H, C, P, N, so every admission read starts from the displayed proposed position, accepted boundary form, witness route, transport, replay, provenance, and local name.

Next apply Definition 1073.3. The accepted boundary form is restricted to the non-axiom sibling routes, and none of those siblings supplies an axiom coordinate, hidden chooser, quotient representative, proposition-equality token, or global frame.

Finally use Definition 1073.4 and Definition 1073.5. Witness-route nonescape prevents a row-preserving consumer from changing the supplied-content route, while the scoped primitive route binds the same rows to BHist, hsame, Cont, Pkg, and NameCert. The checked TasteGate target records that this scoped carrier is admitted as a chapter gate, so the displayed route is exactly the TasteGate-scoped non-axiom admission surface. □

Lean4 ✓ `BEDC.Derived.NonAxiomAdmissionUp.taste gate`

Falsifiable prediction: Every accepted NonAxiomAdmissionUp packet classifies a proposed supply position into a displayed non-axiom boundary form and exposes no consumer route that treats the supplied content as substrate truth.

Independence witness: externalsupplyboundary, supplysocketledger, apophaticsocket, selectorledger, and tastegateadmission: those siblings record particular boundary, socket, ledger, and admission rows, while NonAxiomAdmissionUp classifies the admission decision surface itself. No row-preserving bijection can recover the full X, F, W, H, C, P, N classifier from any one sibling packet.

The Lean follow-up target is BEDC.Derived.NonAxiomAdmissionUp.taste_gate. Its TasteGate.lean file must provide explicit Nontrivial NonAxiomAdmission$^{\uparrow}$ and FieldFaithful NonAxiomAdmission$^{\uparrow}$ instances with a displayed witness pair and field-projection proofs for X, F, W, H, C, P, N.

Closure status for NameCert$_{\text{NonAxiomAdmission}^{\uparrow}}$.

Constructive story: A NonAxiomAdmission$^{\uparrow}$ packet is generated from a proposed supply position, accepted non-axiom form, witness route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{NonAxiomAdmission}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1073.6 binds accepted admission decisions to the displayed X, F, W, H, C, P, N rows, non-axiom boundary siblings, BHist, hsame, Cont, Pkg, NameCert, and the checked TasteGate route for NonAxiomAdmission$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{NonAxiomAdmission}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NonAxiomAdmissionUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim substrate truth, hidden choice, quotient representatives, proposition equality collapse, a global frame, axiomClean audit status, or bridge checking for NonAxiomAdmission$^{\uparrow}$.

Upgrade path: Audit closure requires axiom-purity reporting and bridge work for the checked NonAxiomAdmission$^{\uparrow}$ TasteGate route.

Chapter 1074

A Concrete Naming Certificate for $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$

The $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ packet realizes the admission-decision rule of chapter 9395 as a finite NameCert object. It sits beside the external boundary packet of chapter 1065, the router of chapter 666, and the apophatic naming surface of chapter 543. Those siblings expose a boundary, route a supply kind, or name an external position; this packet binds the decision row that chooses the strongest permitted non-axiom form for a proposed dependency.

Definition 1074.1 (BoundaryAdmissionDecision carrier)**.** *A* $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, F, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the proposed-dependency row, naming the object, operation, theorem, or interface component under admission;*
2. *F is the strongest permitted non-axiom form row, choosing one of closed inductive syntax, computed definition, setup field, mature NameCert row, GAP ledger, bridge slot, or apophatic socket;*
3. *R is the requested-shape row, recording the exact supply shape the consumer wants to read;*
4. *A is the audit/refusal row, recording the local decision when the requested shape cannot be exported as internal substrate truth;*
5. *H records componentwise* hsame *transport for X, F, R, A;*
6. *C records displayed* Cont *replay routes through the same admission rows;*
7. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplyKindRouter}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{AuditGateComposition}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryAdmissionDecision}^{\uparrow}}$ *row.*

The carrier has no coordinate for an anonymous axiom, hidden host selector, detached consumer permission, quotient collapse, or supplied-content theorem.

Lean4 (definition) `BEDC.Derived.BoundaryAdmissionDecisionUp.BoundaryAdmissionDecisionUp`

Theorem 1074.2 (BoundaryAdmissionDecision NameCert obligations)**.** *For the carrier of Definition 1074.1, the local* $\mathsf{NameCert}_{\mathsf{BoundaryAdmissionDecision}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for* $D = (X, F, R, A, H, C, P, N)$*;*
2. *proposed-dependency exposure through* X*;*
3. *strongest permitted non-axiom form exactness through* F*;*
4. *requested-shape exactness through* R*;*
5. *audit/refusal exactness through* A*, with componentwise* hsame *stability, displayed* Cont *replay, package provenance, and local naming supplied by* H, C, P, N*.*

No consumer read of the certificate may bypass F *and reach* R *as an admitted internal dependency.*

Proof. Project the displayed carrier. The accepted packet lists X, F, R, A, H, C, P, N, so the local certificate cannot introduce a second dependency, hidden permitted form, detached requested shape, unrecorded refusal, transport row, replay route, provenance row, or local name.

Read the decision rows in order. The proposed-dependency row X names the object under admission, and the chosen-form row F displays the strongest permitted non-axiom form before any consumer reaches the requested-shape row R.

Use the audit/refusal row A. If the requested shape cannot be exported through F, the row A records refusal rather than turning the request into an axiom, hidden selector, quotient collapse, or host permission.

Finally inspect H, C, P, N. These rows only transport, replay, package, and name the same finite decision packet. Therefore the NameCert obligations are exhausted by the five displayed obligation rows. □

Theorem 1074.3 (BoundaryAdmissionDecision router exhaustion)**.** *Every accepted* $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ *packet forces a consumer route through* F *before the requested-shape row* R *can be read. If* F *is absent, the route cannot distinguish a closed inductive form, computed definition, setup field, NameCert row, GAP ledger, bridge slot, or apophatic socket, and the admission decision is not auditable.*

Proof. Start with carrier admission for $D = (X, F, R, A, H, C, P, N)$. The requested-shape row R is part of the same finite packet as the chosen-form row F.

Apply the NameCert obligations of Definition 1074.2. They expose F before any consumer reads R as a decision result, so a route reaching R without F omits the row that records which non-axiom form carries the request.

The support rows H, C, P, N preserve and name only the displayed packet. They cannot reconstruct an omitted chosen-form row or replace it by host convention. Hence router exhaustion fails when F is absent. □

Falsifiable prediction: A packet omitting F cannot satisfy router exhaustion because a consumer can reach R without a displayed non-axiom form.

Independence witness: external-supply-boundary, supply-kind-router, apophaticname, socket-kind-classifier, auditgatecomposition: none is bijective because each lacks at least one of the proposed-dependency row X, chosen-form row F, requested-shape row R, and audit/refusal row A in the same packet.

The Lean carrier for $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ BoundaryAdmissionDecision}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundaryAdmissionDecision}^{\uparrow}$ instances over the displayed rows X, F, R, A, H, C, P, N before a checked TasteGate target is claimed.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryAdmissionDecision}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryAdmissionDecision}^{\uparrow}$ packet is generated from a proposed-dependency row, strongest permitted non-axiom form row, requested-shape row, audit/refusal row, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The BHist carrier first displays the object under admission, then the exact permitted form, and only then the requested shape. If the request cannot be exported through that form, the audit row records refusal. The support rows keep the decision attached to the same visible dependency. The classifier is therefore a finite admission route rather than an anonymous permission to import external supply.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryAdmissionDecision}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the boundary admission decision carrier, NameCert obligations, router exhaustion, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryAdmissionDecision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim anonymous axioms, hidden host selectors, detached permissions, quotient collapse, or supplied-content theorems.

Upgrade path: To reach obligationClosure, check that every requested-shape read factors through the displayed permitted-form row and audit/refusal row.

Chapter 1075

A Concrete Naming Certificate for $\mathsf{GeneratorLocalInductionBoundary}^{\uparrow}$

The $\mathsf{GeneratorLocalInductionBoundary}^{\uparrow}$ packet realizes the induction boundary of chapter 9395 as a finite NameCert object. It records when induction is licensed by a displayed generator and when a claimed domain has become free-floating boundary pressure. The packet sits beside chapter 1065, chapter 528, chapter 494, chapter 715, and chapter 543: those packets expose supply, selector, question, or apophatic naming rows, while this packet binds generator closure, eliminator locality, refusal, and induction-ledger rows in one carrier.

Definition 1075.1 (GeneratorLocalInductionBoundary carrier)**.** *A* $\mathsf{GeneratorLocalInductionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, E, L, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the generator-domain row, naming the closure of the displayed generator and no larger semantic totality;*
2. *E is the eliminator-locality row, recording that the induction principle consumes exactly the constructor cases of D;*
3. *L is the induction-ledger row, recording the base, step, replay, and consumer rows used by the local eliminator;*
4. *F is the free-floating-refusal row, recording that any universal claim outside D is routed to a boundary request instead of internal proof;*
5. *H records componentwise* hsame *transport for D, E, L, F;*
6. *C records displayed* Cont *replay routes through the same generator, eliminator, ledger, and refusal rows;*
7. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorLocalInductionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for induction over an unspecified semantic totality, no hidden generator, no host well-foundedness oracle, and no theorem converting free-floating induction into internal induction.

Theorem 1075.2 (GeneratorLocalInductionBoundary NameCert obligations)**.** *For the carrier of Definition 1075.1, the local* $\mathsf{NameCert}_{\mathsf{GeneratorLocalInductionBoundary}\uparrow}$ *surface consists exactly of carrier admission for* $G = (D, E, L, F, H, C, P, N)$*, generator-domain exposure through* D*, eliminator-locality exactness through* E*, induction-ledger readback through* L*, free-floating-refusal readback through* F*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Equivalently, a consumer may run generator-local induction only by reading the displayed generator closure, the displayed eliminator cases, and the displayed ledger. A consumer that asks for induction beyond those rows receives only the refusal row and the boundary provenance.

Proof. Project the displayed packet $G = (D, E, L, F, H, C, P, N)$. The generator-domain row D fixes the domain of induction before any eliminator is read.

Read E and L together. The eliminator-locality row says that the admissible cases are exactly the constructor cases of D, while the induction-ledger row records the base, step, replay, and consumer rows actually used by the proof route.

Now inspect F. The refusal row separates any claim over an unspecified semantic totality from the generator-local route. Such a claim can be recorded as boundary pressure, but it is not converted into an internal eliminator.

The remaining rows H, C, P, N transport, replay, package, and name the same four substantive rows. They add no hidden generator and no wider induction principle, so the NameCert obligations are exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.GeneratorLocalInductionBoundaryUp.TasteGate.GeneratorLocalInductionBoundaryTasteGa`
`single carrier alignment`

Theorem 1075.3 (GeneratorLocalInductionBoundary noninternalization)**.** *Every accepted* GeneratorLocalInductio *packet keeps free-floating induction outside the internal proof surface. A downstream consumer may cite* D, E, L, F, H, C, P, N *as boundary evidence for a generator-local induction route, but it may not read from the packet a universal eliminator over an unspecified totality, a hidden well-foundedness oracle, a host induction principle, or a theorem that turns the refusal row into generated structure.*

Proof. Begin with the generator-domain row D. It is a finite displayed row, so the available induction domain is the closure of that row and not an ambient universe of possible cases.

Use E and L to read the permitted proof route. The eliminator row supplies only the constructor cases belonging to D, and the ledger records only the local base, step, replay, and consumer rows.

Apply the refusal row F to every induction request outside that route. The request remains visible as boundary pressure, but there is no coordinate in the packet that supplies a universal eliminator, host induction principle, or hidden well-foundedness oracle.

Finally read H, C, P, N. These rows preserve the same generator, eliminator, ledger, and refusal data under transport, replay, provenance, and naming. Since no packet coordinate internalizes the refused domain, free-floating induction remains outside the internal proof surface. □

Falsifiable prediction: A packet that omits either the displayed generator row or the eliminator row cannot satisfy the non-internalization theorem.

Independence witness: Compare ExternalSupplyBoundary, SupplySocketLedger, SelectorLedger, ApophaticGateQuestion, and ApophaticName: none is bijective to GeneratorLocalInductionBoundary because none simultaneously carries generator closure, eliminator locality, free-floating refusal, and induction-ledger rows.

The Lean follow-up must provide explicit Nontrivial GeneratorLocalInductionBoundary$^{\uparrow}$ and FieldFaithful Generato instances in lean4/BEDC/Derived/GeneratorLocalInductionBoundaryUp/TasteGate.lean before the chapter can claim a checked taste-gate target.

Closure status for NameCert$_{\text{GeneratorLocalInductionBoundary}^{\uparrow}}$.

Constructive story: A GeneratorLocalInductionBoundary$^{\uparrow}$ packet is generated from a displayed generator row, local eliminator row, induction-ledger row, free-floating refusal boundary, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier reads generator closure and eliminator locality as finite data before any induction consumer may use them. If a consumer asks for a free-floating induction principle outside those rows, the refusal boundary is exposed instead. Thus the classifier keeps induction local to generated carrier evidence and refuses host-level induction supply.

Theory closure: TheoryGate(NameCert$_{\text{GeneratorLocalInductionBoundary}^{\uparrow}}$, seed)

Scope closed: This chapter binds the generator local induction boundary carrier, non-internalization theorem, prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{GeneratorLocalInductionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim unrestricted induction, hidden eliminators, external theorem supply, free-floating generator closure, or host equality of induction principles.

Upgrade path: To reach obligationClosure, formalize generator, eliminator, induction-ledger, and refusal-boundary rows with checked nontrivial and field-faithful instances.

Chapter 1076

A Concrete Naming Certificate for $\mathsf{ExternalSupplyFailureWitness}^{\uparrow}$

The $\mathsf{ExternalSupplyFailureWitness}^{\uparrow}$ packet realizes the diagnostic side of chapter 9395 as a finite NameCert object. It sits beside the boundary packet of chapter 1065, the failed audit-gate witness of chapter 901, the composed gate surface of chapter 696, the socket classifier of chapter 1058, and the gate-question packet of chapter 715. Those siblings expose the boundary, failed gate, composed refusal route, socket kind, or question row; this packet records the finite row at which an unsupported external-supply route fails.

Definition 1076.1 (ExternalSupplyFailureWitness carrier)**.** *An* $\mathsf{ExternalSupplyFailureWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, R, F, D, T, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-name row naming the external-supply site;*
2. *R is the requested-supply row recording the demanded supply shape;*
3. *F is the failed-item row naming the unsupported item whose route was rejected;*
4. *D is the diagnostic row exposed to public consumers;*
5. *T is the refusal-target row naming the theorem, witness, selector, quotient, host equality, or completion route that the failure refuses to supply;*
6. *G is the gate row through which the failed route is audited;*
7. *H records componentwise* hsame *transport for B, R, F, D, T, G;*
8. *C records displayed* Cont *replay through the same diagnostic route;*
9. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{AuditGateComposition}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{ApophaticGateQuestion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyFailureWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for a supplied theorem, hidden selector, quotient equality, host-level equality, or direct route from the boundary name to the requested content.

Lean4 (definition) `BEDC.Derived.ExternalSupplyFailureWitnessUp.ExternalSupplyFailureWitnessUp`

Theorem 1076.2 (ExternalSupplyFailureWitness NameCert obligations)**.** *For the carrier of Definition 1076.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyFailureWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for* $W = (B, R, F, D, T, G, H, C, P, N)$*, boundary-name exposure through* B*, requested-supply exactness through* R*, failed-item exactness through* F*, diagnostic exposure through* D*, refusal-target attachment through* T*, gate-row locality through* G*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Equivalently, an accepted packet lets a consumer cite the visible boundary, request, failed item, diagnostic, refusal target, gate, transport, replay, provenance, and certificate rows, but it supplies no refused content beyond those rows.

Proof. Project the displayed carrier. The definition lists $B, R, F, D, T, G, H, C, P, N$, so the local certificate has one boundary-name row, one requested-supply row, one failed item, one diagnostic row, one refusal target, one gate row, and the structural rows that transport, replay, package, and name them.

Use chapter 1065 to identify B and R as boundary-facing rows rather than content rows. Use chapter 901 to read F, D, T as the failed item, diagnostic, and refusal attachment exposed by a failed gate.

Next use chapter 696 and chapter 1058. The composition chapter keeps the refusal route gate-local, while the classifier chapter keeps the requested supply in its socket-kind row instead of turning it into content.

Finally use chapter 715. The question row explains why the consumer must read the gate before the diagnostic answer is available. Since H, C, P, N only preserve the same displayed packet, the NameCert obligations are exhausted by the listed rows. □

Lean4 ✓ `BEDC.Derived.ExternalSupplyFailureWitnessUp.ExternalSupplyFailureWitnessTasteGate single carrier alignment`

Theorem 1076.3 (ExternalSupplyFailureWitness diagnostic nonescape)**.** *Every public consumer of an accepted* $\mathsf{ExternalSupplyFailureWitness}^{\uparrow}$ *packet reaches the diagnostic row* D *before any refusal target* T *is usable. A direct route from the boundary name* B *to theorem content, witness content, selector output, quotient equality, host equality, or completion output is not a route of the packet.*

Proof. Start with Definition 1076.1. The only content-facing coordinates are F, D, T, G; none stores the refused theorem, witness, selector, quotient, host equality, or completion output.

Apply the boundary discipline from chapter 1065. The boundary name B and requested-supply row R identify the site and demanded shape, but the boundary packet does not internalize the requested supply.

Now apply the failed-gate and composition siblings from chapter 901 and chapter 696. The failed item F is attached to a diagnostic row D, and the composed refusal route reads that diagnostic before a refusal target is handed to a consumer.

Use chapter 1058 and chapter 715 to keep the route visible. The socket classifier names the supply kind, and the gate question forces a gate read before the answer is exposed. Therefore any route skipping D or connecting B directly to refused content has no carrier coordinate and refutes the packet. □

Falsifiable prediction: If an accepted ExternalSupplyFailureWitnessUp packet is public, every consumer reaches the diagnostic row before any refusal target; a direct route from boundary name to theorem content refutes the packet.

Independence witness: ExternalSupplyBoundaryUp exposes the boundary but not the failed item; AuditGateFailureWitnessUp exposes a failed gate but not the external-supply boundary name; AuditGateCompositionUp composes refusal routes but does not store the primitive requested supply; SocketKindClassifierUp classifies the socket kind but not the diagnostic row; ApophaticGateQuestionUp asks the gate question but does not attach the failed item to a refusal target. The row separation prevents a row-preserving bijection between any sibling and ExternalSupplyFailureWitnessUp.

The Lean follow-up must provide explicit Nontrivial ExternalSupplyFailureWitness$^{\uparrow}$ and FieldFaithful ExternalSupp instances in lean4/BEDC/Derived/ExternalSupplyFailureWitnessUp/TasteGate.lean before the chapter can claim a checked taste-gate target.

Closure status for NameCert$_{\text{ExternalSupplyFailureWitness}\uparrow}$ ·

Constructive story: An ExternalSupplyFailureWitness$^{\uparrow}$ packet is generated from an external boundary name, failed-item row, diagnostic row, refusal target, gate context, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The BHist carrier exposes the failed external request as finite audit data: a consumer must read the boundary site and diagnostic row before reaching the refusal target. The local ledger records which requested item failed and where the failure is read. Transport and continuation replay preserve exactly that route, so the failure witness cannot be converted into theorem content or a supplied dependency.

Theory closure: TheoryGate(NameCert$_{\text{ExternalSupplyFailureWitness}\uparrow}$, seed)

Scope closed: This chapter binds the external supply failure witness carrier, diagnostic route, prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{ExternalSupplyFailureWitness}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim successful supply, direct theorem content, hidden gate success, host selectors, or boundary-to-proof conversion.

Upgrade path: To reach obligationClosure, check the finite diagnostic route from boundary name through failed item to refusal target.

Chapter 1077

A Concrete Naming Certificate for $\mathsf{ExternalSupplyWitness}^{\uparrow}$

The $\mathsf{ExternalSupplyWitness}^{\uparrow}$ packet realizes the witness-facing part of chapter 9395 as a finite NameCert object. It sits beside the boundary packet of chapter 1065, the socket ledger of chapter 528, the apophatic name surface of chapter 543, the audit gate boundary of chapter 526, and the negative-name boundary of chapter 629. Those siblings expose boundary, socket use, named refusal, audit verdict, or negative-name refusal rows; this packet binds the requested-witness, gate-locality, supply-ledger, and local NameCert rows on one auditable route.

Definition 1077.1 (ExternalSupplyWitness carrier)**.** *An* $\mathsf{ExternalSupplyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, R, G, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the socket-site row naming the exact external supply position;*
2. *R is the requested-witness row recording the witness shape requested at S;*
3. *G is the gate-locality row stating the local audit gate through which R may be read;*
4. *L is the supply ledger row preserving the* GAP *pressure and socket residue of the request;*
5. *H records componentwise* hsame *transports for S, R, G, L;*
6. *C records displayed* Cont *routes through the same socket, witness, gate, and ledger rows;*
7. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*,* $\mathsf{ApophaticName}^{\uparrow}$*,* $\mathsf{AuditGateBoundary}^{\uparrow}$*,* $\mathsf{NegativeNameBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for an internal witness before the gate-locality row, no hidden selector, no host quotient, and no theorem asserting that the requested witness is available as substrate truth.

Lean4 ✓ `BEDC.Derived.ExternalSupplyWitnessUp.ExternalSupplyWitnessTasteGate single carrier alignment`

Theorem 1077.2 (ExternalSupplyWitness NameCert obligations)**.** *For the carrier of Definition 1077.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for $W = (S, R, G, L, H, C, P, N)$, socket-site exposure through S, requested-witness readback through R,*

gate-locality exposure through G, supply-ledger readback through L, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Equivalently, a consumer may read the socket site, the requested witness shape, the gate through which that request is local, and the ledger that keeps the external supply visible, but it may not read an internal witness outside those rows.

Proof. Project the displayed carrier. The accepted packet lists S, R, G, L, H, C, P, N, so the local certificate cannot introduce a second socket site, requested-witness row, gate-locality row, supply ledger, transport row, replay route, provenance row, or local name.

Read the public rows in order. The socket site S names the external supply position, the requested-witness row R records the shape demanded there, and the gate-locality row G states the local audit gate that must be crossed before the request is usable.

Now read the ledger and structural rows. The supply ledger L preserves the socket residue, while H, C, P, N only transport, replay, package, and name the same finite packet. Therefore the NameCert obligations are exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.ExternalSupplyWitnessUp.ExternalSupplyWitness namecert obligations`

Theorem 1077.3 (ExternalSupplyWitness socket factorization)**.** *Every accepted* ExternalSupplyWitness$^{\uparrow}$ *consumer route factors through the socket-site row S before it can cite the requested-witness row R, gate-locality row G, supply ledger L, or structural rows H, C, P, N. The packet exports no alternate route that begins with a hidden witness, detached audit gate, unledgered supply coordinate, quotient representative, or host selector.*

Lean4 ✓ `BEDC.Derived.ExternalSupplyWitnessUp.ExternalSupplyWitness socket factorization`

Proof. Begin with the carrier of Definition 1077.1. The socket site S is the first displayed row and supplies the name under which the requested witness is audited.

Use the requested-witness and gate-locality rows. The request R is meaningful only as a request at S, and the gate row G records the local audit boundary for that same site.

Finally inspect L, H, C, P, N. These rows preserve and replay the same socket route; none contains an independent witness, detached audit gate, unledgered supply coordinate, quotient representative, or host selector. Hence every accepted route factors through S. □

Theorem 1077.4 (ExternalSupplyWitness non-escape)**.** *Every accepted* ExternalSupplyWitness$^{\uparrow}$ *packet keeps the requested witness external until the gate-locality row is displayed. A downstream consumer may cite S, R, G, L, H, C, P, N as witness-boundary evidence, but it cannot read R as a public internal witness before G, nor can it obtain a hidden selector, quotient collapse, proposition-equality token, or theorem that supplies the requested witness.*

Proof. Read S and R as boundary rows. They name the socket site and the requested witness shape, but neither row states that the requested witness has been generated by the substrate.

Apply the gate-locality row G. It is the displayed locality condition for any consumer read of R, so a public witness read before G would bypass the carrier route.

Use the supply ledger L and the structural rows H, C, P, N. They preserve the same externalized request and leave no coordinate for hidden selectors, quotient collapse, proposition-equality tokens, or supplied-witness theorems. Thus the requested witness does not escape the gate-local packet. □

Theorem 1077.5 (ExternalSupplyWitness gate locality)**.** *For every accepted* ExternalSupplyWitness$^{\uparrow}$ *packet* $W = (S, R, G, L, H, C, P, N)$*, the gate-locality row* G *is the unique public gate through which* R *can be consumed. Any consumer of the requested-witness row must preserve the same socket site* S*, supply ledger* L*, componentwise* hsame *transports* H*, displayed* Cont *routes* C*,* Pkg *provenance* P*, and local* NameCert *row* N*.*

Proof. Project the finite packet. The carrier gives a single displayed gate-locality row G between the requested-witness row R and the ledger and structural rows.

Use socket factorization. Since every accepted route begins at S, a consumer cannot attach R to a different public gate without leaving the displayed packet.

Now use non-escape. The rows L, H, C, P, N preserve the same route and do not expose an alternate gate, hidden selector, or detached witness. Therefore G is the unique public gate for consuming R inside the packet. □

Theorem 1077.6 (ExternalSupplyWitness field-faithful surface)**.** *The public* ExternalSupplyWitness$^{\uparrow}$ *surface is field-faithful to the eight rows* S, R, G, L, H, C, P, N*: changing any one of the socket-site, requested-witness, gate-locality, supply-ledger, transport, continuation, provenance, or local NameCert rows changes the exported NameCert surface, while preserving all eight rows preserves the same witness-boundary packet.*

Proof. Start with the NameCert obligations. They identify the exported surface with exactly the rows S, R, G, L, H, C, P, N.

Changing S, R, G, or L changes the socket site, requested witness, public gate, or supply ledger read by the consumer. By socket factorization and gate locality, those rows determine the witness-boundary route.

Changing H, C, P, or N changes the transport, replay, provenance, or local naming by which the same route is stabilized. Conversely, if all eight rows are preserved, the carrier and every public NameCert obligation are preserved. Hence the surface is field-faithful to the displayed packet. □

Falsifiable prediction: A packet whose requested witness row is public before the gate-locality row fails the ExternalSupplyWitnessUp non-escape theorem.

Independence witness: Compare ExternalSupplyBoundary, SupplySocketLedger, ApophaticName, AuditGateBoundary, and NegativeNameBoundary: the witness packet has simultaneous requested-witness, gate-locality, and supply-ledger rows, while those siblings respectively expose boundary, socket use, named refusal, audit verdict, or negative-name refusal rows, so no row-preserving bijection preserves the NameCert surface.

The Lean follow-up must provide explicit Nontrivial ExternalSupplyWitness$^{\uparrow}$ and FieldFaithful ExternalSupplyWitn instances over $W = (S, R, G, L, H, C, P, N)$ in lean4/BEDC/Derived/ExternalSupplyWitnessUp/TasteGate.lean before the chapter can claim a checked taste-gate target.

Closure status for NameCert$_{\text{ExternalSupplyWitness}^{\uparrow}}$**.**

Constructive story: An ExternalSupplyWitness$^{\uparrow}$ packet is generated from a socket-site row, requested-witness row, gate-locality row, supply ledger, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The carrier names the requested witness only as a request at a visible socket site, then requires the gate-locality row before any consumer can inspect the requested shape. The ledger preserves the external supply residue, while transport and continuation replay keep every read on the same finite route. The classifier therefore distinguishes witness-boundary evidence from an internal witness.

Theory closure: TheoryGate(NameCert$_{\text{ExternalSupplyWitness}^{\uparrow}}$, seed)

Scope closed: This chapter binds the external supply witness carrier, socket factorization, non-escape, gate-locality, and field-faithful surface theorems.

Formal verification: FormalStatus(NameCert$_{\text{ExternalSupplyWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim an internal witness before the gate row, hidden selectors, quotient collapse, proposition-equality tokens, or supplied-witness theorems.

Upgrade path: To reach obligationClosure, formalize socket factorization and gate locality for the displayed witness-boundary carrier.

Chapter 1078

A Concrete Naming Certificate for $\mathsf{InterInscriptionCoherence}^{\uparrow}$

The $\mathsf{InterInscriptionCoherence}^{\uparrow}$ packet realizes the space-as-inter-chain-coherence reading of chapter 9401 as a finite NameCert surface. It pairs two local inscription rows, binds them through an inter-Hist locality ledger, and exposes only the transport, route, provenance, and local naming data needed by downstream observer-locality consumers.

The sibling surfaces chapter 1069, chapter 610, chapter 759, chapter 479, and chapter 1065 each contribute part of this reading. None of them contains both paired inscription endpoints and a coherence ledger in one accepted carrier, so the paired packet below is not a renaming of any sibling.

Definition 1078.1 (InterInscriptionCoherence carrier)**.** *An* $\mathsf{InterInscriptionCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, J, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the first local* $\mathsf{InscriptionPoint}^{\uparrow}$ *row, naming one chain-side inscription point of the apophatic fixed point;*
2. *J is the second local* $\mathsf{InscriptionPoint}^{\uparrow}$ *row, naming the comparison chain-side inscription point;*
3. *L is the* $\mathsf{InterHistLocalityLedger}^{\uparrow}$ *row that records the locality-cell coherence between I and J;*
4. *H records componentwise* hsame *transports for the two inscription rows, the inter-*Hist *locality ledger, the continuation routes, the provenance row, and the local naming row;*
5. *C records the displayed* Cont *routes by which downstream consumers read I, J, L, H, P, N without obtaining a global space object or a hidden host metric;*
6. *P records* Pkg *provenance over* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{InterInscriptionCoherence}^{\uparrow}}$ *row.*

The carrier contains no global frame, no host-space coordinate, no quotient observer equality, no hidden shared subject, and no ambient synchronization principle.

Lean4 (definition) `BEDC.Derived.InterInscriptionCoherenceUp.InterInscriptionCoherenceUp`

Theorem 1078.2 (InterInscriptionCoherence NameCert obligations)**.** *For the carrier of Definition 1078.1, the local* $\mathsf{NameCert}_{\mathsf{InterInscriptionCoherence}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Z = (I, J, L, H, C, P, N),$$

paired inscription exposure through I *and* J*, inter-*Hist *locality exposure through* L*, componentwise* hsame *transport through* H*, displayed* Cont *route replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows I, J, L, H, C, P, N are the only coordinates available to the local certificate.

Read the paired endpoint rows first. The two $\mathsf{InscriptionPoint}^{\uparrow}$ rows supply distinct local inscription endpoints, while the $\mathsf{InterHistLocalityLedger}^{\uparrow}$ row supplies the coherence ledger between them.

The structural rows then close the certificate. Transport is componentwise through H, consumer replay is through C, provenance is recorded by P, and the local naming row N certifies the same finite packet without adding a global frame or host-space coordinate. □

Theorem 1078.3 (InterInscriptionCoherence locality transport)**.** *For every accepted* $\mathsf{InterInscriptionCoherence}^{\uparrow}$ *packet* $Z = (I, J, L, H, C, P, N)$*, any downstream locality consumer that reads coherence between inscription points must read both inscription rows* I, J *and the inter-*Hist *locality ledger* L *through the displayed transport row* H *and route row* C*. If either inscription row or the inter-*Hist *locality ledger is omitted, no accepted packet can satisfy this locality-transport surface.*

Proof. Start from carrier admission in Definition 1078.1. The coherence relation is stored only by the triple I, J, L inside the finite packet.

Apply the NameCert obligations of Definition 1078.2. They require paired inscription exposure and locality-ledger exposure before any consumer route may replay the packet.

The falsifiable clause follows by projection. Removing I, removing J, or removing L destroys one of the required coordinates of the accepted carrier, so the displayed transport and route rows have no complete locality surface to carry. □

Theorem 1078.4 (InterInscriptionCoherence ledger non-escape)**.** *Every consumer of an accepted* $\mathsf{InterInscriptionCoherence}^{\uparrow}$ *packet receives only the rows* I, J, L, H, C, P, N*. The consumer receives no global frame, no host metric, no quotient observer equality, no hidden shared subject, and no ambient synchronization coordinate.*

Proof. Project the carrier and the NameCert obligations. The public inventory is exactly the paired inscription rows, the inter-Hist locality ledger, transport, route, provenance, and local naming rows.

Use locality transport from Definition 1078.3. A coherence read must travel through the displayed I, J, L, H, C surface, so it cannot be replaced by a host-space metric or a quotient observer equality.

Finally use the absence clauses in the carrier. No row of Z names a global frame, hidden shared subject, or ambient synchronization principle, and downstream consumers cannot recover such data from provenance or local naming alone. □

Theorem 1078.5 (InterInscriptionCoherence sibling dependency)**.** *The* $\mathsf{InterInscriptionCoherence}^{\uparrow}$ *carrier depends on the sibling surfaces* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$*,* $\mathsf{InscriptionPoint}^{\uparrow}$*,* $\mathsf{InterHistLocalityLedger}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*, and* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$*, but it is not equivalent to any one of them. Its*

*accepted packet is the paired inscription-and-ledger surface $Z = (I, J, L, H, C, P, N)$, with two inscription endpoints, one inter-*Hist *locality ledger, transport, route, provenance, and local NameCert data.*

Proof. Read the sibling roles from the provenance row of Definition 1078.1. The apophatic fixed-point fiber supplies the boundary reading, inscription-point packets supply the endpoint rows, the inter-Hist locality ledger supplies coherence, observer-history identity supplies the history-side consumer discipline, and the external-supply boundary keeps the far end non-internal.

Each sibling lacks at least one coordinate required by Z. The fixed-point fiber lacks the paired locality ledger; an inscription-point packet supplies one endpoint rather than two; an inter-Hist locality ledger lacks the two local inscription rows; observer-history identity lacks the apophatic inscription endpoints; and the external-supply boundary lacks the coherence ledger.

Therefore the dependency is strict. Downstream Lean work must provide explicit Nontrivial InterInscriptionCoher and FieldFaithful InterInscriptionCoherence$^{\uparrow}$ instances in TasteGate.lean for the whole paired packet rather than for a sibling alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{InterInscriptionCoherence}^{\uparrow}}$.

Constructive story: An InterInscriptionCoherence$^{\uparrow}$ packet is generated from two local inscription rows, one inter-Hist locality ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InterInscriptionCoherence}^{\uparrow}}$, seed)

Scope closed: Definition 1078.1, Definition 1078.2, Definition 1078.3, Definition 1078.4, and Definition 1078.5 bind the paired inscription endpoints, inter-Hist locality ledger, transport, route, provenance, local naming, non-escape, and sibling dependency surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InterInscriptionCoherence}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global space, host metrics, quotient observer equality, hidden shared subjects, ambient synchronization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locality transport, ledger non-escape, sibling dependency, and TasteGate instances for nontriviality and field faithfulness over the paired packet.

Chapter 1079

A Concrete Naming Certificate for $\mathsf{HashApophaticSeal}^{\uparrow}$

The $\mathsf{HashApophaticSeal}^{\uparrow}$ packet realizes the digest, fiber, GAP-ledger, and common apophatic far-end boundary of chapter 9402 as a single finite NameCert seal. It sits beside the hash certificate of chapter 242, the digest-fiber ledger of chapter 1067, the apophatic fixed-point packet of chapter 1068, the apophatic socket packet of chapter 536, and the external-supply boundary of chapter 1065. The seal differs from each sibling because it binds the visible digest, fiber ledger, GAP provenance, far-end boundary, and refusal row in one consumer-facing packet.

Definition 1079.1 (HashApophaticSeal carrier)**.** *A* $\mathsf{HashApophaticSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, F, G, T, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest row, read through the* $\mathsf{Hash}^{\uparrow}$ *certificate surface;*
2. *F is the fiber-ledger row, recording source-side membership behind D without converting digest sameness into source equality;*
3. *G is the* GAP *provenance row that keeps the hidden fiber auditable;*
4. *T is the common apophatic far-end boundary row for the displayed fiber route;*
5. *R is the refusal row blocking any consumer from treating T as a kernel object, source payload, selected preimage, or global digest;*
6. *H records componentwise* hsame *transports for D, F, G, T, R, C, P, N;*
7. *C records the displayed* Cont *routes by which digest, fiber, GAP, far-end, and refusal reads remain in the same finite packet;*
8. *P is the* Pkg *provenance over* $\mathsf{Hash}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HashApophaticSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for source equality from digest sameness, no hidden preimage selector, no internal object representing the far end, no global universe digest, and no outside-supply witness.

Theorem 1079.2 (HashApophaticSeal NameCert obligations)**.** *For the carrier of Definition 1079.1, the local* $\mathsf{NameCert}_{\mathsf{HashApophaticSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (D, F, G, T, R, H, C, P, N),$$

visible digest exposure through D*, fiber-ledger exposure through* F*, GAP provenance through* G*, common far-end boundary naming through* T*, apophatic refusal through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Every accepted consumer of the seal must preserve those rows together. In particular, digest sameness may be read only as the visible side of the displayed packet, while source equality, preimage selection, object-level far-end internalization, and global digest readings remain absent unless a separate exactness certificate is displayed outside the $\mathsf{HashApophaticSeal}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.HashApophaticSealUp.HashApophaticSealNameCertObligations`

Proof. Project the accepted carrier in Definition 1079.1. Its displayed coordinates are exactly $D, F, G, T, R, H, C, P, N$.

Read the sibling sources through their local roles. The hash certificate supplies the visible digest row; the digest-fiber ledger supplies the fiber and GAP rows; the apophatic fixed-point and socket surfaces name the far-end boundary without internalizing it; and the external-supply boundary keeps the refused outside object visible only as boundary data.

The NameCert surface is exhausted by those rows. Since no carrier coordinate stores a source-equality reflection, selected preimage, internal far-end object, global digest, or outside-supply witness, an accepted consumer cannot obtain any of those objects from the seal. □

Theorem 1079.3 (HashApophaticSeal fiber nonescape)**.** *Every accepted* $\mathsf{HashApophaticSeal}^{\uparrow}$ *consumer that reads the visible digest row* D *must route through the fiber-ledger row* F*, GAP provenance row* G*, common far-end boundary row* T*, refusal row* R*, transport row* H*, continuation row* C*, package provenance* P*, and local naming row* N*. Removing the GAP fiber ledger while keeping the digest row makes the seal unable to distinguish digest sameness from source equality.*

Consequently the seal exports no selected preimage, quotient collapse, host source equality, internal T *object, global universe digest, hidden outside supply, or unledgered far-end read.*

Proof. Start from an accepted packet $A = (D, F, G, T, R, H, C, P, N)$. The digest row D is only the visible face of the carrier, while F and G record the fiber membership and provenance needed to keep the source side auditable.

Apply the NameCert obligations of Definition 1079.2. They force a digest consumer to read D together with the fiber ledger, GAP row, far-end boundary, refusal row, componentwise transport, continuation replay, provenance, and local name.

Now inspect the carrier inventory. If F and G are removed while D remains, the remaining rows do not record which source-side fiber produced the digest; digest sameness therefore cannot be separated from source equality by this seal. Because the accepted carrier contains no selected preimage, quotient collapse, host source equality, internal T object, global digest, hidden supply, or unledgered far-end coordinate, none of those readings escapes the displayed packet. □

Falsifiable prediction: Removing the GAP fiber ledger while keeping the digest row makes the seal unable to distinguish digest sameness from source equality.

Independence witness: Nearest siblings: hash, apophaticfixedpoint, apophaticsocket, externalsupplyboundary, and gap-policy/namecert surfaces; no bijection exists because the seal owns a combined digest, fiber-ledger, and apophatic far-end refusal row absent from each sibling.

The Lean TasteGate boundary is `lean4/BEDC/Derived/HashApophaticSealUp/TasteGate.lean`. It provides `Nontrivial HashApophaticSealUp` and `FieldFaithful HashApophaticSealUp` instances for the displayed digest, fiber, GAP, far-end, refusal, transport, continuation, provenance, and naming carrier.

Lean4 ✓ `BEDC.Derived.HashApophaticSealUp.HashApophaticSealTasteGate single carrier alignment`

Theorem 1079.4 (HashApophaticSeal public seal export)**.** *Every accepted* $\mathsf{HashApophaticSeal}^{\uparrow}$ *carrier*

$$A = (D, F, G, T, R, H, C, P, N)$$

exports one public seal packet whose visible surface is the digest row D, fiber-ledger row F, GAP provenance row G, common far-end boundary row T, refusal row R, componentwise transport row H, continuation row C, package provenance row P, and local naming row N. The exported packet permits digest, fiber, GAP, and far-end refusal reads, but it exports no source equality from digest sameness, selected preimage, internal far-end object, global digest, hidden outside supply, or unledgered fiber route.

Proof. Project the accepted carrier. The public rows are exactly $D, F, G, T, R, H, C, P, N$, and each row is already listed in the local NameCert surface of Definition 1079.2.

The digest row is public only together with the fiber ledger and GAP provenance. By Definition 1079.3, any digest read that drops F or G loses the information that separates digest sameness from source equality.

The far-end row is public only as boundary data. The refusal row R, transport row H, continuation row C, provenance row P, and local name N keep the exported packet inside the displayed seal rather than an internal object or selected preimage.

Thus the seal exports one finite public packet and nothing outside it. Source equality, selected preimages, internal far-end objects, global digest readings, hidden outside supply, and unledgered fiber routes remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{HashApophaticSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{HashApophaticSeal}^{\uparrow}$ packet is generated from a visible digest, a digest-fiber ledger, GAP provenance, a common apophatic far-end boundary, a refusal row, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HashApophaticSeal}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1079.1, Definition 1079.2, Definition 1079.3, Definition 1079.4, the falsifiable prediction, the independence witness, and the TasteGate marker bind $D, F, G, T, R, H, C, P, N$ to BHist, hsame, Cont, Pkg, NameCert, and the sibling scopes $\mathsf{Hash}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, and $\mathsf{ExternalSupplyBoundary}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HashApophaticSeal}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.HashApophaticSealUp.HashApophaticSealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close source equality from digest sameness, selected preimages, internal far-end objects, a global digest, an outside-supply witness, or bridge checking.

Upgrade path: Bridged closure requires a standard bridge that carries the public digest, fiber, GAP, and far-end refusal packet into the acceptance-bridge schema.

Chapter 1080

A Concrete Naming Certificate for $\mathsf{InscriptionHashTrace}^{\uparrow}$

The $\mathsf{InscriptionHashTrace}^{\uparrow}$ packet realizes the hash-like apophatic fixed-point digest of chapter 9402 as a finite inscription trace. It sits beside the hash certificate of chapter 242, the inscription event packet of chapter 519, the trace classifier of chapter 1052, the apophatic fixed-point packet of chapter 1068, and the hash-apophatic seal of chapter 1079. Its distinguishing role is that the visible digest, inscription event, trace classifier, refusal ledger, route audit, transport, continuation, package, and local name are one finite consumer-facing row.

Definition 1080.1 (InscriptionHashTrace carrier)**.** *An* $\mathsf{InscriptionHashTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, I, K, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest row read through the* $\mathsf{Hash}^{\uparrow}$ *certificate surface;*
2. *I is the finite inscription-event row that records where the digest is written into the observer-facing trace;*
3. *K is the inscription trace-classifier row that keeps digest sameness distinct from trace-source equality;*
4. *R is the refusal-ledger row blocking hidden preimage, global digest, and object-level* T *reads;*
5. *A is the route-audit row showing that the digest read passes through inscription and classifier surfaces before the apophatic fixed-point seal is cited;*
6. *H records componentwise* hsame *transports for D, I, K, R, A, C, P, N;*
7. *C records the displayed* Cont *route through the digest, inscription event, classifier, refusal, audit, provenance, and naming rows;*
8. *P is the* Pkg *provenance over* $\mathsf{Hash}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionTraceClassifier}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, $\mathsf{HashApophaticSeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionHashTrace}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a selected digest preimage, a global universe hash, source equality from digest sameness, an uninscribed trace read, an unaudited route, or an internal object named T*.*

Theorem 1080.2 (InscriptionHashTrace NameCert obligations)**.** *For the carrier of Definition 1080.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionHashTrace}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (D, I, K, R, A, H, C, P, N),$$

visible digest exposure through D*, inscription-event exposure through* I*, trace-classifier exposure through* K*, refusal-ledger exposure through* R*, route-audit exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Every accepted consumer preserves these rows together. In particular, the hash-like digest can be cited toward the apophatic fixed-point seal only after the inscription event, trace classifier, refusal ledger, and route audit have been displayed.

Proof. Project the accepted carrier in Definition 1080.1. Its displayed rows are exactly $D, I, K, R, A, H, C, P, N$.

Read the sibling surfaces in order. The hash certificate supplies the visible digest row, the inscription-event packet supplies the finite writing site, the trace-classifier packet supplies the trace-facing classifier, and the hash-apophatic seal supplies the apophatic fixed-point endpoint only through a displayed refusal-aware route.

The rows R and A are part of the same carrier, so refusal and route audit are not optional side conditions. Since the carrier has no further coordinate, these rows exhaust the local NameCert surface. □

Theorem 1080.3 (InscriptionHashTrace refusal nonescape)**.** *Every accepted* $\mathsf{InscriptionHashTrace}^{\uparrow}$ *consumer that reads* D *must also read* I, K, R, A, H, C, P, N*. Consequently the packet exports no selected preimage, no global universe digest, no source equality obtained from digest sameness, no uninscribed trace, no unaudited route to the seal, and no internal* T *object.*

Proof. Start with the NameCert obligations of Definition 1080.2. They place every public digest read inside the finite carrier $T = (D, I, K, R, A, H, C, P, N)$.

The inscription event and trace classifier are read before the refusal and audit rows. Thus a consumer cannot detach the digest from the trace site or replace classifier sameness by source equality.

Finally inspect the carrier inventory. It contains no preimage selector, global digest, uninscribed trace coordinate, unaudited route coordinate, or internal T object, so none of those readings can escape the displayed packet. □

Theorem 1080.4 (InscriptionHashTrace audit route)**.** *Any accepted route from an* $\mathsf{InscriptionHashTrace}^{\uparrow}$ *digest row to the* $\mathsf{HashApophaticSeal}^{\uparrow}$ *endpoint factors through the displayed chain*

$$\mathsf{Hash}^{\uparrow} \to \mathsf{InscriptionEvent}^{\uparrow} \to \mathsf{InscriptionTraceClassifier}^{\uparrow} \to R \to A \to H, C, P, N.$$

The route cannot cite the apophatic fixed-point seal before the digest evidence, inscription event, classifier row, refusal ledger, route audit, componentwise hsame*,* Cont*,* Pkg*, and* NameCert *rows have all been exposed.*

Lean4 ✓ `BEDC.Derived.InscriptionHashTraceUp.TasteGate.InscriptionHashTrace audit route`

Proof. Use Definition 1080.2 to put the route inside the accepted carrier. The only digest-facing row is D, and it is packaged with the inscription event I and classifier row K.

Next read R and A. The refusal row records the blocked preimage, global digest, and object-level T readings, while the audit row records the permitted route toward the hash-apophatic seal.

The remaining rows H, C, P, N transport, replay, cite, and name only that finite route. Hence any seal-facing consumer sees the displayed chain and no shortcut around it. □

Theorem 1080.5 (InscriptionHashTrace field-faithful prediction)**.** *A Lean TasteGate carrier for* $\mathsf{InscriptionHashTrace}^{\uparrow}$ *is field-faithful only when equality of the projected rows* $D, I, K, R, A, H, C, P, N$ *determines equality of the whole packet, and it is nontrivial only when at least one displayed digest row can vary while the remaining rows stay fixed. Dropping* I*,* K*,* R*, or* A *makes the carrier unable to witness the inscription hash trace route.*

Lean4 ✓ `BEDC.Derived.InscriptionHashTraceUp.TasteGate.InscriptionHashTraceTasteGate single carrier alignment`

Proof. The carrier of Definition 1080.1 is a constructor over the nine displayed rows. Equality of the projected rows therefore gives equality of the constructor instance, while changing the digest row gives a nontrivial witness pair.

The route theorem above supplies the semantic boundary for this encoding. Without the inscription event, the digest is not trace-local; without the classifier, digest sameness can be mistaken for trace-source equality; without the refusal ledger, hidden preimage and object-level T readings are not blocked; and without the audit row, the route to the hash-apophatic seal is not displayed. □

Falsifiable prediction: Any accepted $\mathsf{InscriptionHashTrace}^{\uparrow}$ consumer factors through the displayed hash digest, inscription event, trace classifier, refusal ledger, route-audit, hsame, Cont, Pkg, and NameCert rows before it can cite the apophatic fixed-point seal.

Independence witness: hash, inscriptionevent, inscriptiontraceclassifier, apophaticfixedpoint, and hashapophaticseal: no bijection preserves the displayed rows because $\mathsf{InscriptionHashTrace}^{\uparrow}$ alone binds the digest row, inscription event, trace classifier, refusal ledger, route audit, transport, continuation, provenance, and local NameCert row in one carrier.

The Lean TasteGate boundary is `lean4/BEDC/Derived/InscriptionHashTraceUp/TasteGate.lean`. It includes explicit `Nontrivial InscriptionHashTraceUp` and `FieldFaithful InscriptionHashTraceUp` instances for the displayed finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionHashTrace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{InscriptionHashTrace}^{\uparrow}$ packet is generated from a visible digest row, inscription-event row, trace-classifier row, refusal-ledger row, route-audit row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionHashTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1080.1, Definition 1080.2, Definition 1080.3, Definition 1080.4, and Definition 1080.5 bind the carrier, NameCert obligations, refusal nonescape, audit route, and field-faithful prediction rows for the finite inscription hash trace packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionHashTrace}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.InscriptionHashTraceUp.TasteGate.InscriptionHashTraceTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close hidden preimages, global universe digests, source equality from digest sameness, unaudited routes, or bridge checking.

Upgrade path: Obligation closure requires checked carrier, classifier, refusal, audit-route, nontriviality, and field-faithfulness rows.

Chapter 1081

A Concrete Naming Certificate for $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$

The $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ packet is the finite selector-to-seal carrier for the Real and Completion handoff of chapter 9238. It sits after the selector-budget packets and before a consumer reads a $\mathsf{Real}^{\uparrow}$ completion seal: the packet records the finite selector budget, selected window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ route as one displayed certificate surface.

Falsifiable prediction: A completion consumer that reads the RealUp seal without the displayed selector-budget and RegSeqRat readback rows refutes the packet.

Independence witness: cauchymodulusrefinement, cauchylimitseal, realcompletionsealclassifier, realsealcomparison, regularcauchycomparison: no listed sibling is bijective to $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$. CauchyModulusRefinement records the refinement route without the terminal selector-to-seal packet, CauchyLimitSeal records a seal without the complete selector budget and readback route, RealCompletionSealClassifier records classifier-facing seal comparison without selected-window budget admission, RealSealComparison compares sealed routes without the upstream selector budget, and RegularCauchyComparison compares regular names without the completion seal handoff.

Definition 1081.1 (Real completion selector-seal carrier)**.** *A* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, W, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite selector-budget row naming the precision request, selector index, and admitted budget surface for the completion handoff;*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row produced by B;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected window;*
4. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row reached only after the selector budget, selected window, and regular-rational readback are displayed;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal route consumed by the Real and Completion handoff;*
6. *H records componentwise* hsame *transports for the selector budget, selected window, readback, limit seal, terminal Real seal, route, provenance, and local naming rows;*
7. *C records the displayed* Cont *replay route from B to W, from W to R, from R to L, and from L to E;*

8. *P is the* Pkg *provenance over selector-budget packets,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}^{\uparrow}}$ *row.*

The classifier compares carriers only by the displayed rows $B, W, R, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, a hidden selector, a detached limit object, a second completion route, or ambient completeness.

Theorem 1081.2 (Real completion selector-seal NameCert obligations)**.** *For the carrier of Definition 1081.1, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (B, W, R, L, E, H, C, P, N),$$

selector-budget exposure through B, selected-window exposure through W, $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff through L, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier namecert obligations`

Proof. Project the carrier of Definition 1081.1. Its public surface is exactly the selector budget, selected window, readback row, limit seal, terminal Real seal, transports, continuations, provenance, and local naming row.

Read the route in dependency order. The selector budget B admits the requested finite observation budget, the window row W records the chosen finite window, the readback row R exposes the regular-rational data in that window, and the limit-seal row L is available only after those rows are displayed.

The terminal seal row E is then a route from the same displayed packet. The structural rows H, C, P, N transport, replay, record, and name only this finite route, so the certificate has no remaining coordinate for quotient equality, host equality, hidden selection, detached limits, or ambient completeness. □

Theorem 1081.3 (Real completion selector-seal route determinacy)**.** *If two accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *consumers expose the same selector budget B, selected window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L, terminal* $\mathsf{Real}^{\uparrow}$ *seal row E, and the same structural rows H, C, P, N, then they determine the same finite Real and Completion handoff route. No second selected window, hidden selector, detached limit row, quotient stream equality, host real equality, or ambient completeness coordinate can distinguish the two reads.*

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealRouteDeterminacy`

Proof. Use the carrier definition to compare the two accepted reads row by row. The rows $B, W, R, L, E, H, C, P, N$ are the whole carrier surface, and the classifier has no additional comparison coordinate.

Apply the NameCert obligations. They force every seal-facing read through the same selector budget, selected window, regular-rational readback, limit seal, and terminal Real seal before the structural rows can package the route.

Since both consumers expose the same displayed rows, their componentwise hsame transports and Cont replays name the same finite route. The excluded objects are absent carrier coordinates, so they cannot produce a second handoff. □

Theorem 1081.4 (Real completion selector-seal non-escape)**.** *Every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *completion consumer factors through the finite route*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

The consumer may read the selector budget, selected window, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transports, continuations, provenance, and local* NameCert *row. It receives no quotient stream equality, host real equality, hidden selector, detached limit object, second completion route, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier nonescape`

Proof. Start from the accepted packet $S = (B, W, R, L, E, H, C, P, N)$. By construction, the selector budget and selected window precede the readback row, and the readback row precedes the limit-seal and terminal Real seal rows.

Use Definition 1081.2. The local certificate exports exactly the displayed route and makes H, C, P, N structural rows for that route rather than independent sources of completion data.

Finally inspect the excluded coordinates. None is present in the carrier or classifier, so a completion consumer cannot bypass B, W, R, L, E through quotient streams, host real equality, hidden selection, a detached limit, a second route, or ambient completeness. □

Theorem 1081.5 (Real completion selector-seal L10 handoff scope)**.** *Every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet factors the current L10 Real and Completion handoff through the sibling chapters chapter 126, chapter 301, chapter 386, and chapter 38. Its visible route has the ordered carrier spine*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E,$$

where B *admits the selector budget,* W *is the selected* $\mathsf{StreamName}^{\uparrow}$ *window,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* L *is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer handoff, and* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint row. The rows* H, C, P, N *only transport, replay, record provenance for, and name this route.*

Consequently the packet is an L10 handoff node for the Real and Completion surface, not an independent completion principle. It supplies no quotient stream equality, host real equality, hidden selector, detached limit object, second completion route, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier l10 handoff scope`

Proof. Project the carrier from Definition 1081.1. Its first five coordinates are exactly the selector budget, selected stream window, regular-rational readback, Cauchy-limit seal row, and terminal Real endpoint row, and the prose immediately ties these rows to the four sibling chapters in that order.

Apply Definition 1081.2. The NameCert obligation inventory admits only those five route rows followed by H, C, P, N, so the latter rows can only perform componentwise hsame transport, Cont replay, Pkg provenance recording, and local naming for the same displayed path.

Finally use Definition 1081.4. Non-escape excludes every bypass coordinate named in the statement: quotient stream equality, host real equality, hidden selectors, detached limits, second completion routes, and ambient completeness. Thus every accepted selector-seal packet lands on the L10 route $B \to W \to R \to L \to E$ before any structural row is read. □

Theorem 1081.6 (Real completion selector-seal selector-budget totality)**.** *Every accepted* RealCompletionSelectorS *packet exposes the selector-budget row B before any selected window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row L, terminal* $\mathsf{Real}^{\uparrow}$ *endpoint row E, or structural row H, C, P, N is readable. In particular, no completion consumer can read a selected window, regular-rational readback, limit seal, terminal endpoint, transport, continuation, provenance, or local naming row without first passing through the finite selector-budget admission recorded by B.*

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier selector budget totality`

Proof. Project the carrier of Definition 1081.1. Its displayed packet is $S = (B, W, R, L, E, H, C, P, N)$, with B placed as the first route row and with W, R, L, E, H, C, P, N recorded as rows whose meanings are tied to that same packet.

Apply Definition 1081.5. The L10 handoff route has ordered spine

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E,$$

and the rows H, C, P, N only transport, replay, record provenance for, and name this route.

Thus the selected window, readback, limit seal, terminal endpoint, and structural rows are not independent entries. They are readable only after the selector-budget admission row has already been displayed. □

Theorem 1081.7 (Real completion selector-seal readback exactness)**.** *Once an accepted* RealCompletionSelectorSea *packet displays its selector-budget row B and selected window row W, every regular-rational readback route factors through the single* $\mathsf{RegSeqRat}^{\uparrow}$ *row R. The readback cannot be replaced by quotient stream equality, host real equality, a detached limit row, a second completion route, or any ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier regseq readback exactness`

Proof. Use Definition 1081.2. The local certificate lists $\mathsf{RegSeqRat}^{\uparrow}$ readback exactly as the row R that follows the selector budget B and selected window W.

Next apply Definition 1081.3. Two accepted consumers that expose the same $B, W, R, L, E, H, C, P, N$ determine the same finite handoff route, and there is no classifier coordinate that can distinguish the route by quotient stream equality, host real equality, detached limits, second routes, or ambient completeness.

Therefore any readback visible after B and W must pass through the displayed row R. Replacing R by an external equality or by a detached limit would introduce a coordinate absent from the carrier and so would leave the accepted packet. □

Theorem 1081.8 (Real completion selector-seal observer-budget handoff)**.** *Let*

$$S = (B, W, R, L, E, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet, and let*

$$G = (F, I, S_{\text{obs}}, O, B_{\text{obs}}, W_{\text{obs}}, D_{\text{obs}}, R_{\text{obs}}, E_{\text{obs}}, H_{\text{obs}}, C_{\text{obs}}, P_{\text{obs}}, N_{\text{obs}})$$

be an accepted $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *packet reading the same selector budget boundary. If W_{obs} supplies the selected* $\mathsf{StreamName}^{\uparrow}$ *window for W, D_{obs} supplies the dyadic ledger read by*

that window, R_{obs} supplies the $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and E_{obs} supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal face reached by E, then the selector-seal consumer remains on the route*

$$I \to F \to S_{\text{obs}} \to B_{\text{obs}} \to W_{\text{obs}} \to D_{\text{obs}} \to R_{\text{obs}} \to E_{\text{obs}} \to L, E.$$

The omitted observer rows O remain boundary-only data. They cannot be read as the selected window, dyadic ledger, regular-rational readback, limit-seal handoff, terminal Real seal, quotient stream equality, host real equality, second completion route, ambient completeness principle, or TasteGate instance.

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealObserverBudgetHandoff`

Proof. Start with selector-budget totality, Definition 1081.6. It forces the $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ consumer to expose B before W, R, L, E, H, C, P, N can be read.

Use readback exactness, Definition 1081.7. Once the selector budget and selected window are displayed, the readback route is the single $\mathsf{RegSeqRat}^{\uparrow}$ row R, not a quotient stream equality, host real equality, detached limit, second route, or ambient completeness principle.

Now apply the observer-filter budget-selector handoff, Definition 1042.3. It routes the selected observation through $I, F, S_{\text{obs}}, B_{\text{obs}}, W_{\text{obs}}, D_{\text{obs}}, R_{\text{obs}}, E_{\text{obs}}$ before the selector-seal packet consumes the limit and Real endpoint rows.

Finally use omitted-row exclusion, Definition 1042.4. The omitted observer ledger remains outside the positive route coordinates, while the existing TasteGate witness surface in this chapter remains a witness row over $B, W, R, L, E, H, C, P, N$ rather than an additional completion source. Hence the consumer route is exactly the displayed observer-budget handoff followed by L, E. □

Theorem 1081.9 (Real completion selector-seal selector-window classifier pullback)**.** *Let*

$$S = (B, W, R, L, E, H, C, P, N)$$

and

$$S' = (B', W', R', L', E', H', C', P', N')$$

be accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packets. They have the same selector-window classifier surface exactly when the selector budget B, selected* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural H, C, P, N rows agree componentwise with their primed counterparts.*

The classifier excludes quotient stream equality, host real equality, hidden selectors, detached limit objects, second completion routes, and ambient completeness.

Proof. For the forward direction, apply the NameCert obligations of Definition 1081.2. The local certificate exposes exactly $B, W, R, L, E, H, C, P, N$ as the selector-to-seal surface.

Use route determinacy, Definition 1081.3. If two accepted packets present the same displayed rows, they determine the same finite Real and Completion handoff route, so the selector-window classifier has no further coordinate to inspect.

For the reverse direction, assume componentwise agreement of $B, W, R, L, E, H, C, P, N$. Selector-budget totality, Definition 1081.6, forces both reads to enter through B before the selected window or readback is available, and readback exactness, Definition 1081.7, fixes R as the only regular-rational readback after B and W.

Finally apply non-escape, Definition 1081.4. Hidden selectors, quotient stream equality, host real equality, detached limits, second completion routes, and ambient completeness are absent from the packet, so none can split the agreed selector-window classifier surface. □

Theorem 1081.10 (Real completion selector-seal terminal consumer exhaustion)**.** *Every terminal* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *completion consumer factors through the full ordered route*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

The terminal consumer may use the $\mathsf{StreamName}^{\uparrow}$ *selected window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *endpoint only as the displayed continuation of that route. It has no residual access to a hidden selector, quotient stream equality, host real equality, detached limit object, second completion route, or ambient completeness principle.*

Proof. Begin with Definition 1081.6. It forces every terminal read to expose the selector-budget row B before the selected window, readback, limit seal, endpoint, or structural rows are available.

Use Definition 1081.7 for the readback step. After B and W are visible, the only regular-rational readback row available to the packet is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row R.

Now apply Definition 1081.4. Non-escape forces every accepted completion consumer through $B, W, R, L, E, H, C, P, N$ and excludes hidden selectors, external equalities, detached limits, second completion routes, and ambient completeness.

Finally use Definition 1081.5. The sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ references are exactly the route positions W, R, L, E, so the terminal consumer exhausts the displayed handoff rather than opening another completion surface. □

Theorem 1081.11 (Real completion selector-seal terminality)**.** *Any two accepted terminal Real and Completion handoff consumers that expose the same ordered rows*

$$B, W, R, L, E, H, C, P, N$$

factor through a unique $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *selector-seal route. Every accepted terminal handoff is exhausted by that route: the selector-budget row B, the selected* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L, the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint E, and the structural rows H, C, P, N are the whole finite NameCert surface available to the consumer.*

The terminal route supplies no hidden selector, quotient stream equality, host real equality, detached limit object, second completion route, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealTerminality`

Proof. First identify the ordered route. Apply Definition 1081.5: every accepted packet reaches the Real and Completion handoff through the sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ positions W, R, L, E after the selector-budget row B.

Next enforce the entry and readback constraints. By Definition 1081.6, no selected window, readback, limit seal, terminal endpoint, or structural row is readable before B. By Definition 1081.7, once B and W are displayed, the regular-rational readback is exactly the row R.

Now compare two terminal consumers exposing the same rows. Use Definition 1081.3; equality of the displayed rows $B, W, R, L, E, H, C, P, N$ forces the same finite Real and Completion handoff route, so the selector-seal route through them is unique.

Finally apply Definition 1081.10. It says every terminal handoff consumer has no residual access beyond the displayed route and excludes hidden selectors, external equality coordinates, detached limits, second completion routes, and ambient completeness. Hence every accepted terminal handoff is exhausted by the unique selector-seal route. □

Theorem 1081.12 (Real completion selector-seal diagonal compatibility pullback)**.** *Let*

$$S = (B, W, R, L, E, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet, and let*

$$K = (R_0, T_0, S_0, D_0, W_0, Q_0, L_0, H_0, C_0, P_0, N_0)$$

be an accepted $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose selected-window row, regular-rational readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row agree with* W, R, E*. Then every terminal selector-seal consumer of* S *factors through the same finite diagonal route fixed by* K*. The pullback route cannot introduce quotient streams, selected limits, host real equality, hidden selectors, independent completion data, or ambient completeness.*

Proof. First fix the selector-seal route. By selector-budget totality and readback exactness, Definition 1081.6 and Definition 1081.7, a terminal consumer of S must expose B before W and then read the displayed row R.

Terminality, Definition 1081.11, exhausts the same consumer by the ordered rows $B, W, R, L, E, H, C, P, N$. Thus the selector-seal side supplies one finite route from selector budget through selected window, readback, limit-seal handoff, and terminal endpoint.

Now use the diagonal packet. The selector-budget source lock, Definition 646.193, fixes the selector-budget read of K to its displayed diagonal source and finite rows. Since K agrees with S on the selected window, regular readback, and terminal Real seal rows, the shared rows W, R, E lie on that same diagonal route.

Route determinacy and consumer non-escape for diagonal compatibility, Definition 646.198 and Definition 646.196, identify any completion-facing or $\mathsf{Real}^{\uparrow}$-facing consumer using those shared rows with the finite route of K.

Finally compare the two carrier surfaces. The selector-seal non-escape theorem, Definition 1081.4, leaves no coordinate in S for hidden selectors, quotient stream equality, host real equality, detached limits, second completion routes, or ambient completeness. The diagonal non-escape theorem leaves no such coordinate in K either, so the terminal selector-seal consumer is the pullback of the diagonal compatibility route rather than an independent completion source. □

Theorem 1081.13 (Real completion selector-seal diagonal completion factorization)**.** *Every accepted terminal* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *consumer*

$$S = (B, W, R, L, E, H, C, P, N)$$

that also reads a $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *selector and a* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *seal factors through the ordered selector-seal route*

$$B \longmapsto W \longmapsto R \longmapsto L \longmapsto E,$$

then through the finite-window diagonal route of chapter 4080, and then through the completion-existence seal route of chapter 5721 before any completion-facing consumer reads the terminal row E*. The factorization refuses detached selectors, quotient stream equality, host real equality, selected limits, second completion routes, and ambient completeness.*

Proof. First use terminality and non-escape, Definition 1081.11 and Definition 1081.4. They force the terminal consumer to expose $B, W, R, L, E, H, C, P, N$ and leave no coordinate for a hidden selector, external equality, detached limit, second route, or ambient completeness principle.

Next apply the diagonal compatibility pullback of Definition 1081.12. A consumer that also reads the diagonal selector must identify its selected-window and regular-rational readback rows with the displayed W and R rows of S, so the diagonal selector is read as a finite-window route rather than as a detached choice of limit.

Finally route the completion-existence seal through chapter 5721. That chapter exposes completion existence only after its source metric, Cauchy schedule, dense embedding, selected windows, regular readback, dyadic ledger, and completion seal rows have been displayed. Hence the selector-seal endpoint E is consumed only after the ordered B, W, R, L, E route, the diagonal-selection route, and the metric-completion-existence route have all been read. □

Theorem 1081.14 (Real completion selector-seal Cont spine compatibility)**.** *Every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet exposes exactly the two displayed* Cont *witnesses for the selector-to-seal spine:*

$$B \to W \to R \qquad \textit{and} \qquad R \to L \to E.$$

The structural rows H, C, P, N can package the route only after the selector-budget row B, selected window W, $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L, and terminal* $\mathsf{Real}^{\uparrow}$ *endpoint E have been displayed through those two continuation witnesses. No accepted packet adds a third continuation witness, skips either displayed witness, or reads H, C, P, N as an independent completion route.*

Proof. Project the carrier of Definition 1081.1. Its public route rows are B, W, R, L, E and its structural continuation row is the displayed row C for that same carrier.

Use selector-budget totality and readback exactness, Definition 1081.6 and Definition 1081.7. They force the first continuation witness to expose the selector budget, selected window, and regular-rational readback as $B \to W \to R$ before any limit-seal endpoint is readable.

Then use terminality, Definition 1081.11. The terminal endpoint is exhausted by the displayed route, so the second continuation witness exposes $R \to L \to E$ and leaves H, C, P, N as packaging rows for that route rather than further completion data. □

Theorem 1081.15 (Real completion selector-seal two-step Cont boundary)**.** *Any consumer that tries to read the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint without both displayed* Cont *witnesses*

$$B \to W \to R \qquad \textit{and} \qquad R \to L \to E$$

has left the accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. Such a read cannot cite* $\mathsf{Real}^{\uparrow}$ *completion data through the selector-seal route, cannot use H, C, P, N as a substitute for the missing witness, and cannot recover the endpoint from quotient stream equality, host real equality, a detached limit object, a hidden selector, or ambient completeness.*

Proof. Start from Definition 1081.14. It identifies the two continuation witnesses that must be displayed before the terminal endpoint is packaged.

Apply non-escape, Definition 1081.4. Every accepted completion consumer factors through $B, W, R, L, E, H, C, P, N$, so omitting either continuation witness omits part of the carrier route itself.

Finally use terminality, Definition 1081.11. The terminal Real and Completion handoff is exhausted by the displayed route, so a consumer outside the two-step boundary has no remaining packet coordinate from which to cite $\mathsf{Real}^{\uparrow}$ completion data. □

Theorem 1081.16 (Real completion selector-seal window tolerance lock)**.** *Every accepted* RealCompletionSelectorS *packet exposing the terminal route*

$$B \to W \to R \to L \to E$$

uses the same selected $\mathsf{StreamName}^{\uparrow}$ *window* W *and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *across both displayed* Cont *witnesses*

$$B \to W \to R \qquad \textit{and} \qquad R \to L \to E.$$

The terminal seal is therefore fed by the selected-window and readback surface already displayed in the packet. No second selected window, detached readback, quotient stream equality, host real equality, hidden selector, detached limit, or ambient completeness row can be introduced between R *and* L*.*

Proof. First expose the entry side. Selector-budget totality, Definition 1081.6, forces the packet to display B before any selected window, readback, limit-seal, endpoint, or structural row is readable.

Next identify the readback surface. By Definition 1081.7, once B and W are visible, every regular-rational readback route factors through the displayed row R and cannot be replaced by an external equality or detached completion object.

Now use Cont spine compatibility and the two-step boundary, Definition 1081.14 and Definition 1081.15. The first witness ends at the displayed row R, and the second witness begins from that same row R on its way to L and E; omitting or replacing either witness leaves the accepted packet.

Finally apply non-escape, Definition 1081.4. Since every accepted consumer factors through $B, W, R, L, E, H, C, P, N$, there is no carrier coordinate between R and L for another window, another readback, quotient stream equality, host real equality, hidden selection, detached limits, or ambient completeness. The terminal seal is fed by exactly the selected window and readback already displayed. □

Theorem 1081.17 (Real completion selector-seal structural row nonpromotion)**.** *In every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet, the rows* H, C, P, N *transport, replay, record provenance for, and name only the already displayed route*

$$B \to W \to R \to L \to E.$$

They do not create a hidden selector, quotient stream equality, detached limit object, second completion route, host real equality, or ambient completeness row. In particular, the structural rows cannot turn the selector-seal packet into an independent completion principle after the two displayed Cont *witnesses have been read.*

Proof. Use the L10 handoff scope, Definition 1081.5. It places H, C, P, N after the visible selector-budget, selected-window, readback, limit-seal, and terminal-endpoint rows.

Use Definition 1081.15 to fix the continuation boundary. A terminal endpoint read that lacks either displayed continuation witness is outside the accepted packet, so H, C, P, N cannot supply a replacement route.

Finally combine non-escape with terminal consumer exhaustion, Definition 1081.4 and Definition 1081.10. The excluded coordinates are absent from the carrier, and the terminal consumer is exhausted by the displayed rows, so the structural rows remain transport, replay, provenance, and naming rows only. □

Theorem 1081.18 (Real completion selector-seal carrier admission)**.** *For every accepted* RealCompletionSelectorSea *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

carrier admission exposes exactly the selector-budget row B, selected-window row W, $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row L, terminal* $\mathsf{Real}^{\uparrow}$ *endpoint row E, componentwise transport row H, continuation row C, provenance row P, and local naming row N. No admitted carrier has an additional quotient-stream, host-real, hidden-selector, detached-limit, second-route, or ambient-completeness coordinate.*

Proof. Project the carrier definition, Definition 1081.1. Its displayed packet has exactly the nine rows $B, W, R, L, E, H, C, P, N$.

Apply the NameCert obligation inventory, Definition 1081.2. It reads those same rows as the full local certificate surface: carrier admission, selector budget, selected window, readback, limit seal, terminal endpoint, transport, continuation, provenance, and local naming.

Finally use route determinacy and non-escape, Definition 1081.3 and Definition 1081.4. Since all accepted reads are determined by the displayed rows and every excluded coordinate is absent from the carrier, admission exposes exactly the stated packet. □

Theorem 1081.19 (Real completion selector-seal selector-budget exposure)**.** *For every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

each completion-facing read of W, R, L, E, H, C, P, N passes through the selector-budget row B. The packet therefore exposes the selector budget as the entry row for the selected window, regular-rational readback, limit-seal handoff, terminal endpoint, and structural rows.

Proof. Use selector-budget totality, Definition 1081.6. It states that no selected window, readback, limit seal, endpoint, transport, continuation, provenance, or local naming row is readable before B.

Apply the L10 handoff scope, Definition 1081.5. The route spine begins with B and then proceeds to W, R, L, E, while H, C, P, N package that same route.

The NameCert obligations of Definition 1081.2 identify the local certificate rows with this displayed packet. Hence every accepted read of a non-budget row is already a read downstream of the selector-budget admission row. □

Theorem 1081.20 (Real completion selector-seal selected-window readback)**.** *For every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

the selected-window readback route displays W before it displays the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. After B and W are exposed, every regular-rational readback available to the packet is the row R.*

Proof. Start with selector-budget totality, Definition 1081.6. It places the selected-window row W after B and before any readback, limit seal, endpoint, or structural row is readable.

Use readback exactness, Definition 1081.7. Once B and W have been displayed, the only regular-rational readback route inside the accepted packet factors through R.

Terminal consumer exhaustion, Definition 1081.10, keeps the terminal consumer on the same ordered route. The readback step is therefore W followed by R, not a detached equality or a second completion read. □

Theorem 1081.21 (Real completion selector-seal limit-seal handoff)**.** *For every accepted* RealCompletionSelectorSe *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

the $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row* L *is reachable only as the displayed continuation of the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. A read that bypasses* $R \to L$ *leaves the selector-seal packet.*

Proof. Apply the L10 handoff scope, Definition 1081.5. Its ordered carrier spine places the limit-seal row immediately after the readback row in the route $B \to W \to R \to L \to E$.

Use readback exactness, Definition 1081.7, to fix R as the only regular-rational readback after B and W. Then use Cont spine compatibility, Definition 1081.14, which displays the second continuation witness as $R \to L \to E$.

The two-step Cont boundary of Definition 1081.15 excludes a terminal read that omits either displayed continuation witness. Thus a limit-seal handoff that bypasses $R \to L$ is not an accepted $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ handoff. □

Theorem 1081.22 (Real completion selector-seal terminal endpoint routing)**.** *For every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}$ *endpoint row* E *is routed from the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row* L*, and the rows* H, C, P, N *remain structural rows for that route. They transport, replay, record provenance for, and name the route; they do not create another completion endpoint.*

Proof. Use terminality, Definition 1081.11. It says that the accepted terminal handoff is exhausted by the ordered rows $B, W, R, L, E, H, C, P, N$.

Apply terminal consumer exhaustion, Definition 1081.10. The terminal consumer uses the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ endpoint only as the displayed continuation of the selector-to-seal route.

Finally combine non-escape with structural row nonpromotion, Definition 1081.4 and Definition 1081.17. The rows H, C, P, N cannot supply a hidden selector, detached limit, second route, quotient equality, host equality, or ambient completeness; they remain structural support for the route from L to E. □

Theorem 1081.23 (Real completion selector-seal TasteGate witness surface)**.** *For every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet*

$$S = (B, W, R, L, E, H, C, P, N),$$

the displayed rows $B, W, R, L, E, H, C, P, N$ *form the paper-side witness surface for later* Nontrivial *and* FieldFaithful *TasteGate work. Two accepted packets whose selector-budget rows or selected-window rows differ are distinguishable at the selector-seal boundary unless a displayed componentwise transport row in* H *carries one row to the other. Conversely, equality of all displayed rows* $B, W, R, L, E, H, C, P, N$ *determines the whole selector-seal packet, with no hidden selector, quotient stream equality, host real equality, detached limit, second completion route, or ambient completeness coordinate left to separate the packets.*

Proof. Start from the carrier definition, Definition 1081.1. The accepted packet has exactly the rows $B, W, R, L, E, H, C, P, N$, and the classifier compares carriers only by those rows.

Apply route determinacy and the window tolerance lock, Definition 1081.3 and Definition 1081.16. The selector budget B fixes the admissible selected window W, the readback row R, and the route toward L and E; changing B or W therefore changes the displayed route unless the transport row H supplies the componentwise identification.

Finally use structural row nonpromotion, Definition 1081.17. The rows H, C, P, N transport, replay, record, and name only the already displayed route, so equality of all nine rows leaves no further coordinate that could separate two packets. This is precisely the finite witness surface on which later TasteGate nontriviality and field faithfulness can be tested. □

Theorem 1081.24 (Real completion selector-seal bridge readiness)**.** *The ordered route*

$$B \to W \to R \to L \to E$$

together with H, C, P, N forms the bridge-ready finite handoff from selector budget to terminal $\mathsf{Real}^{\uparrow}$ *endpoint inside the* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet. The handoff uses exactly the L10 handoff scope, selector-budget totality,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exactness, terminal consumer exhaustion, terminality, the two-step* Cont *boundary, and structural-row nonpromotion rows already displayed in this chapter.*

Proof. Start with the L10 handoff scope. It orders the visible selector-budget, selected-window, readback, limit-seal, terminal-endpoint, transport, continuation, provenance, and naming rows as a finite packet route.

Next apply selector-budget totality and $\mathsf{RegSeqRat}^{\uparrow}$ readback exactness. These results force the route to expose B before selected-window and readback use, and then bind every readback consumer to the displayed row R rather than to a quotient stream equality, host real equality, detached limit, or ambient completeness witness.

Finally combine terminal consumer exhaustion, terminality, the two-step Cont boundary, and structural-row nonpromotion. The terminal endpoint is read through the displayed B, W, R, L, E route, while H, C, P, N only transport, replay, record provenance for, and name that route. This gives a finite bridge-ready handoff without claiming a standard bridge. □

Theorem 1081.25 (Real completion selector-seal standard bridge premise)**.** *Any standard bridge interpreter for the same displayed selector budget B, selected* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L,* $\mathsf{Real}^{\uparrow}$ *endpoint E, componentwise* hsame *transport rows H,* Cont *replay rows C,* Pkg *provenance row P, and* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}^{\uparrow}}$ *row N is determined by the existing* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *carrier. Such an interpreter cannot introduce quotient stream equality, host real equality, a hidden selector, a detached limit, a second completion route, or ambient completeness.*

Proof. First fix the displayed handoff. Bridge readiness, Definition 1081.24, makes the finite route $B \to W \to R \to L \to E$ together with H, C, P, N the only bridge-ready surface supplied by the packet.

Next apply route determinacy and non-escape, Definition 1081.3 and Definition 1081.4. They force any consumer reading the same displayed rows to read the same selector-seal route and leave no carrier coordinate for a quotient stream equality, host real equality, hidden selector, detached limit, second completion route, or ambient completeness witness.

Finally use structural-row nonpromotion and public route export, Definition 1081.17 and Definition 1081.29. The rows H, C, P, N only transport, replay, record provenance for, and name the displayed route, and the public surface exports exactly that route. Therefore a standard bridge

interpreter over the named rows is a reader of the existing carrier rather than an additional completion source. □

Theorem 1081.26 (Real completion selector-seal obligation scope)**.** *Every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet exposes exactly the selector-budget row* B*, selected finite-window row* W*, limit-seal handoff* L*, terminal endpoint route* E*, componentwise* hsame *transport* H*, displayed* Cont *route* C*,* Pkg *provenance* P*, and local* NameCert *row* N*, together with the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *on the displayed handoff. It exports no countable-choice selector, quotient stream, host real equality, ambient completeness, or hidden limit oracle.*

Proof. Project the carrier of Definition 1081.1. Its displayed packet is exactly $S = (B, W, R, L, E, H, C, P, N)$, so the selector budget, selected window, regular-rational readback, limit seal, endpoint, transport, continuation, provenance, and local naming rows are the only available coordinates.

Use the existing readback and handoff theorems in order. Selector-budget totality exposes B before W, R, L, E, H, C, P, N; $\mathsf{RegSeqRat}^{\uparrow}$ readback exactness fixes R after B and W; terminal consumer exhaustion and terminality carry the route through L and E.

Finally apply Cont spine compatibility, structural row nonpromotion, and bridge readiness. They keep H, C, P, N as transport, replay, provenance, and naming rows for the same displayed handoff, so no countable-choice selector, quotient stream, host equality, ambient completeness principle, or hidden limit oracle is exported. □

Theorem 1081.27 (Real completion selector-seal TasteGate grounding)**.** *The checked Lean target* `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier_namecert_obligatio` *grounds the carrier and* NameCert *obligation row for* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$*. The remaining* `Nontrivial RealCompletionSelectorSealUp` *and* `FieldFaithful RealCompletionSelectorSealUp` *instances are the later TasteGate obligations and are not closed by this chapter.*

Proof. Read the chapter's carrier and local naming theorem first. The carrier displays the finite rows $B, W, R, L, E, H, C, P, N$, and the local NameCert theorem records the package, structural row, and naming obligations over that carrier.

The Lean target checks precisely that carrier and NameCert obligation surface. It therefore supplies the verification anchor for obligation closure over the finite selector-seal packet.

The TasteGate instance rows ask for `Nontrivial RealCompletionSelectorSealUp` and `FieldFaithful RealCompletionSelectorSealUp`. Those instance obligations require additional TasteGate data and remain outside the present obligation boundary. □

The Lean follow-up must provide `Nontrivial RealCompletionSelectorSealUp` and `FieldFaithful RealCompletionSelectorSealUp` in `lean4/BEDC/Derived/RealCompletionSelectorSealUp/TasteGate.lean`.

Theorem 1081.28 (Real completion selector-seal scoped consumer surface)**.** *Every accepted* RealCompletionSelecto *packet exposes a single scoped L10 consumer surface*

$$S = (B, W, R, L, E, H, C, P, N).$$

Here B *is the selector budget,* W *is the selected* $\mathsf{StreamName}^{\uparrow}$ *window,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* L *is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint,* H *records componentwise transports,* C *records the displayed continuation routes,* P *records provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}^{\uparrow}}$ *row.*

The surface is scoped to the current L10 Real and Completion handoff: it is generated by the carrier, certified by the NameCert obligations, ordered by the L10 handoff scope, entered through

selector-budget totality, read back through the RegSeqRat exactness row, exhausted at the terminal consumer and terminality rows, bounded by the two-step Cont *boundary, kept finite by structural-row nonpromotion, made bridge-ready without claiming a bridge, and grounded by the TasteGate row. It exports no quotient stream equality, host real equality, hidden selector, detached limit object, second completion route, ambient completeness principle, TasteGate instance, or checked bridge.*

Proof. Begin with the carrier and NameCert obligations, Definition 1081.1 and Definition 1081.2. They give exactly the displayed rows $B, W, R, L, E, H, C, P, N$ and make those rows the local certificate surface.

Use the L10 handoff scope, selector-budget totality, and RegSeqRat readback exactness. These results order the consumer route through $B \to W \to R \to L \to E$, force every read to enter through the selector budget, and make R the only readback row after the selected window is displayed.

Now apply terminal consumer exhaustion, terminality, and the two-step Cont boundary. The terminal Real endpoint is consumed only through the displayed route, any two terminal consumers with the same rows determine the same selector-seal route, and a read missing either continuation witness lies outside the accepted packet.

Finally combine structural-row nonpromotion, bridge readiness, and TasteGate grounding. The rows H, C, P, N only transport, replay, record provenance for, and name the same route; the bridge-ready statement keeps the handoff finite without asserting a standard bridge; and the TasteGate grounding identifies the checked carrier and NameCert obligation row as the verification anchor. The exclusions in the statement are precisely the coordinates absent from the carrier and from each consumer theorem, so the packet exposes one scoped L10 consumer surface and nothing beyond it. □

Theorem 1081.29 (Real completion selector-seal public route export)**.** *Every accepted* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *packet exports the public L10 selector-seal route*

$$B \to W \to R \to L \to E.$$

The structural rows H, C, P, N only transport, replay, record provenance for, and name that route. Every public Real-and-Completion consumer must read the selector budget B, selected $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff L, terminal* $\mathsf{Real}^{\uparrow}$ *endpoint E, componentwise transports H, displayed continuations C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}^{\uparrow}}$ *row N.*

The exported public surface gives no quotient stream equality, host real equality, hidden selector, detached limit object, second completion route, ambient completeness principle, TasteGate instance, or checked bridge.

Proof. First expose the public route. The L10 handoff scope, Definition 1081.5, orders every accepted packet through $B \to W \to R \to L \to E$ before any structural row is read.

Next fix the public reads. Selector-budget totality, Definition 1081.6, forces every consumer to enter through B, and RegSeqRat readback exactness, Definition 1081.7, fixes R as the only readback after B and W.

Then exhaust the endpoint. Terminal consumer exhaustion and terminality, Definition 1081.10 and Definition 1081.11, make every terminal Real-and-Completion consumer factor through the displayed route and leave no residual completion coordinate.

Finally apply structural-row nonpromotion and the scoped consumer surface, Definition 1081.17 and Definition 1081.28. The rows H, C, P, N remain transport, replay, provenance, and naming

rows for the same route, while the scoped surface excludes quotient stream equality, host real equality, hidden selectors, detached limits, second completion routes, ambient completeness, TasteGate instances, and checked bridges. Thus the accepted packet exports exactly the public selector-seal route named in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}\uparrow}$.

Constructive story: A $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ packet is generated from a finite selector-budget row, selected StreamName window, RegSeqRat readback row, CauchyLimitSeal handoff row, terminal Real endpoint row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1081.1, Definition 1081.2, Definition 1081.3, Definition 1081.4, Definition 1081.5, Definition 1081.6, Definition 1081.7, Definition 1081.10, Definition 1081.11, Definition 1081.17, Definition 1081.18, Definition 1081.19, Definition 1081.20, Definition 1081.21, Definition 1081.22, Definition 1081.23, Definition 1081.24, Definition 1081.27, Definition 1081.28, and Definition 1081.29 with

Lean4 ✓ `BEDC.Derived.RealCompletionSelectorSealUp.taste gate`

bind the carrier, NameCert obligations, route determinacy, non-escape, L10 handoff scope, selector-budget totality, readback exactness, terminal consumer exhaustion, terminality, structural nonpromotion, carrier admission, selector-budget exposure, selected-window readback, limit-seal handoff, terminal endpoint routing, TasteGate witness surface, bridge readiness, TasteGate grounding, scoped L10 consumer surface, and public route export rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionSelectorSeal}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealCompletionSelectorSealUp.RealCompletionSelectorSealCarrier\_namecert\_obligations`

Bridge status: paperBridge

Not claimed: This chapter does not close quotient stream equality, host real equality, hidden selectors, detached limits, second completion routes, ambient completeness, TasteGate instances, or bridge checking.

Upgrade path: The remaining formal-axis work is a machine-checked standard bridge from auditCleanV to bridgeCheckedV over the same selector-budget, selected-window, RegSeqRat readback, CauchyLimitSeal handoff, Real endpoint, transport, continuation, provenance, and naming rows.

Chapter 1082

A Concrete Naming Certificate for $\mathsf{InscribedRouteSelector}^{\uparrow}$

The $\mathsf{InscribedRouteSelector}^{\uparrow}$ packet realizes the route-selection act described by the inscription-event vision in chapter 9397. It sits between the full event surface of chapter 519, the visible residue ledger of chapter 520, and the route-audit packet of chapter 1005. The selector isolates the moment at which a visible gap is paired with one named route, one accepted check, and one downstream consumer role before public record export is allowed.

Definition 1082.1 (Inscribed route-selector carrier)**.** *An* $\mathsf{InscribedRouteSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, M, R, K, U, L, H, C, P, N)$$

with the following displayed rows:

1. *G is the visible gap row inherited from the inscription-event source boundary and GAP ledger;*
2. *M is the named public record row to be reached by the selected route;*
3. *R is the selected route row from G to M;*
4. *K is the accepted check row for the route selection;*
5. *U is the downstream consumer role permitted to read the accepted route as public record data;*
6. *L is the GAP-ledger row recording the residue left after R, K, U have been displayed;*
7. *H records componentwise* hsame *transports for G, M, R, K, U, L together with package and local name rows;*
8. *C records the displayed* Cont *routes by which consumers inspect the gap, selected route, accepted check, consumer role, and ledger residue;*
9. *P is the* Pkg *provenance over* BHist, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, $\mathsf{InscriptionAcceptanceBudget}^{\uparrow}$, $\mathsf{InscriptionTraceClassifier}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InscribedRouteSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden route choice, an off-packet checker, an unbudgeted consumer, a private source predicate, or a public record export that bypasses the selected route row.

Theorem 1082.2 (Inscribed route-selector NameCert obligations)**.** *For the carrier of Definition 1082.1, the local* $\mathsf{NameCert}_{\mathsf{InscribedRouteSelector}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier exposure for* $S = (G, M, R, K, U, L, H, C, P, N)$*;*
2. *route-selection admission requiring the visible gap* G*, named public record* M*, selected route* R*, accepted check* K*, and downstream consumer role* U *to be displayed in the same packet;*
3. *componentwise stability of gap, name, route, check, consumer, and ledger rows under the* hsame *transports in* H*;*
4. *exactness of public export: the selected route may be exported only through* R, K, U *together with the GAP-ledger row* L *and the displayed routes in* C*;*
5. *ledger rows for GAP residue, route audit, acceptance budget, trace-classifier handoff, package provenance, and the local naming certificate.*

Lean4 ✓ `BEDC.Derived.InscribedRouteSelectorUp.InscribedRouteSelectorTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 1082.1. The displayed packet lists the gap, name, route, accepted check, consumer, ledger, transport, continuation, package, and local naming rows.

Route-selection admission reads the first five rows together. The visible gap G supplies the open boundary, M names the public record, R selects the route, K admits that selection, and U names the consumer role that may read it.

Stability is componentwise through H. Exactness and ledger coverage then use C, L, P, N to replay the same finite route for the route-audit, acceptance-budget, and trace-classifier consumers. Since no hidden route choice, off-packet checker, or extra consumer row is present, the five rows exhaust the local NameCert surface. □

Falsifiable prediction: A packet that omits the accepted route row cannot export an inscribed public record to any downstream consumer.

Theorem 1082.3 (Inscribed route-selector nontriviality prediction)**.** *For every accepted* InscribedRouteSelector$^{\uparrow}$ *carrier* $S = (G, M, R, K, U, L, H, C, P, N)$*, a downstream consumer that lacks the selected route row* R *or the accepted check row* K *cannot export* M *as an inscribed public record. Such a consumer may read the visible gap and GAP-ledger residue, but the public inscription route remains unavailable until* R, K, U *are present in the same finite packet.*

Proof. Project the accepted carrier. The only row whose role is route selection is R, and the only row whose role is route admission is K.

Apply Definition 1082.2. Public export is licensed by the simultaneous display of G, M, R, K, U and is replayed through C with residue tracked in L.

If either R or K is absent, the remaining rows still expose a visible gap and a ledger residue, but they do not expose an accepted selected route to the named record. Therefore the consumer cannot export M as inscribed public record data. □

Independence witness: inscriptionevent, inscriptiongapledger, inscriptionrouteaudit, inscriptionacceptancebudget, inscriptiontraceclassifier: the selector packet binds visible gap, named record, selected route, accepted check, downstream consumer, GAP ledger, hsame transport, Cont routes, Pkg provenance, and local NameCert in one carrier. The named siblings respectively expose the whole event, the residue ledger, route audit, acceptance budget, or trace classifier, but none singles out the accepted route-selection act as this finite packet.

The future Lean target family $BEDC.Derived.InscribedRouteSelectorUp$ must provide explicit Nontrivial and FieldFaithful instances for the displayed selector carrier before the route-selection packet can serve as a checked taste gate.

Closure status for $\mathsf{NameCert}_{\mathsf{InscribedRouteSelector}\uparrow}$.

Constructive story: An $\mathsf{InscribedRouteSelector}^{\uparrow}$ packet is generated from a visible gap row, a named public record row, one selected route row, one accepted check row, one downstream consumer role, a GAP-ledger row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscribedRouteSelector}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1082.1, Definition 1082.2, and Definition 1082.3 bind the visible-gap, selected-route, accepted-check, consumer-role, GAP-ledger, transport, route, provenance, and local naming surface of inscription route selection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscribedRouteSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean checking, bridge checking, private agency, hidden source choice, unchecked route replay, or consumer rights outside the displayed packet.

Upgrade path: Obligation closure requires Lean encoding of the selector carrier and checked route-selection admission, exactness, nontriviality, and field-faithfulness results.

Chapter 1083

A Concrete Naming Certificate for $\mathsf{DigestLoopRefusal}^{\uparrow}$

The $\mathsf{DigestLoopRefusal}^{\uparrow}$ packet realizes the loop-facing refusal in the hash-like apophatic fixed-point discipline of chapter 9402. It sits after the digest provenance packet of chapter 1062, the fiber ledger of chapter 1067, the hash apophatic seal of chapter 1079, the apophatic far-end socket of chapter 1061, and the inscription hash trace of chapter 1080. Those siblings expose visible digest, source, fiber, far-end, and inscription routes; this carrier adds the row that refuses the loop in which digest equality is treated as a source-preimage generator.

Definition 1083.1 (DigestLoopRefusal carrier)**.** *A* $\mathsf{DigestLoopRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (V, F, A, L, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the visible digest row, read through the digest provenance and hash-trace surfaces;*
2. *F is the fiber ledger row, recording which source-side histories may be mentioned behind V without identifying them with the digest;*
3. *A is the apophatic far-end socket row for the hidden end of the fiber;*
4. *L is the loop-refusal row blocking the route from digest equality back to global source-preimage assertion;*
5. *R is the exactness-refusal ledger row, distinguishing displayed local exactness certificates from absent global preimage data;*
6. *H records componentwise* hsame *transport for V, F, A, L, R, C, P, N;*
7. *C records displayed* Cont *replay routes for the digest, fiber, socket, loop-refusal, exactness-refusal, transport, provenance, and naming rows;*
8. *P is the* Pkg *provenance over* $\mathsf{DigestProvenancePacket}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{HashApophaticSeal}^{\uparrow}$, $\mathsf{InscriptionHashTrace}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DigestLoopRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden preimage selector, digest-as-source equality, global source universe, quotient source representative, host hash-function inverse, or internal object at the apophatic far end.

Theorem 1083.2 (DigestLoopRefusal NameCert obligations)**.** *For an accepted* $\mathsf{DigestLoopRefusal}^{\uparrow}$ *carrier*

$$D = (V, F, A, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DigestLoopRefusal}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, visible digest exposure through* V*, fiber-ledger exposure through* F*, apophatic far-end socket exposure through* A*, loop-refusal exposure through* L*, exactness-refusal exposure through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every accepted consumer must preserve the loop-refusal row together with the digest and fiber rows. Thus digest equality can be used only as a visible digest comparison unless a separate displayed exactness certificate is supplied outside the $\mathsf{DigestLoopRefusal}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.DigestLoopRefusalUp.DigestLoopRefusalTasteGate single carrier alignment`

Proof. Project the carrier in Definition 1083.1. Its displayed coordinates are precisely $V, F, A, L, R, H, C, P, N$, so the local NameCert row has no hidden coordinate from which to read a preimage selector or host inverse.

Read the sibling inputs in their local roles. The digest provenance packet supplies digest and source discipline, the digest fiber ledger supplies the fiber row, the hash apophatic seal supplies the common far-end refusal, the inscription hash trace supplies the route audit, and the apophatic far-end socket supplies the negative socket boundary.

The loop-refusal row L is then the additional public obstruction. Transport, replay, provenance, and naming rows preserve that same obstruction through H, C, P, N and do not add exactness data. Therefore the NameCert obligations are exhausted by the displayed packet. □

Theorem 1083.3 (DigestLoopRefusal no global preimage)**.** *No accepted* $\mathsf{DigestLoopRefusal}^{\uparrow}$ *consumer can derive a global source-preimage row from equality of visible digest rows alone. A source-preimage claim is admissible only when a separate exactness certificate is displayed; otherwise the consumer receives only the digest row* V*, fiber row* F*, far-end socket row* A*, loop-refusal row* L*, exactness-refusal row* R*, and structural rows* H, C, P, N*.*

Proof. Start with two visible digest reads. The carrier exposes those reads through V, while F records the fiber ledger and A records the apophatic far-end socket.

Apply the NameCert obligations of Definition 1083.2. They require the loop-refusal row L and the exactness-refusal row R to remain attached to any digest-facing consumer route.

Now inspect the excluded coordinates in Definition 1083.1. There is no hidden preimage selector, no global source universe, no quotient representative, and no host inverse row. Without an external exactness certificate, equality of digest rows therefore remains a visible comparison and cannot become a global source-preimage row. □

Theorem 1083.4 (DigestLoopRefusal fiber nonescape)**.** *Every accepted* $\mathsf{DigestLoopRefusal}^{\uparrow}$ *consumer that reads the fiber ledger* F *must also preserve the visible digest row* V*, the apophatic far-end socket* A*, the loop-refusal row* L*, the exactness-refusal row* R*, componentwise transport* H*, continuation replay* C*, provenance* P*, and local naming* N*. The fiber read exports no source equality, selected preimage, host equality, quotient source representative, global digest universe, or internal far-end object.*

Lean4 ✓ `BEDC.Derived.DigestLoopRefusalUp.DigestLoopRefusal fiber nonescape`

Proof. Begin with the fiber row F. By Definition 1083.1, F is one coordinate of the accepted packet and is not a source object by itself.

Use the digest-fiber and apophatic sibling routes through chapter 1067 and chapter 1079. They already force fiber reads to keep GAP, far-end, and refusal rows visible rather than internalizing the source side.

Finally apply the loop-refusal obligations of Definition 1083.2. The rows L, R, H, C, P, N keep the fiber read on the same finite packet and remove the forbidden source-equality and preimage exits. □

Falsifiable prediction: If a DigestLoopRefusalUp carrier is accepted, no consumer can derive a global source-preimage row from digest equality without an explicit exactness certificate.

Independence witness: digestprovenancepacket, digestfiberledger, hashapophaticseal, inscriptionhashtrace, and apophaticfarendsocket: this carrier is not bijective to any sibling because it adds the loop-refusal row that blocks digest-to-source inversion while preserving visible digest, fiber ledger, apophatic far-end socket, exactness-refusal, transport, replay, provenance, and local naming rows.

The Lean follow-up target is `BEDC.Derived.DigestLoopRefusalUp`. Its `TasteGate.lean` file must provide `Nontrivial DigestLoopRefusalUp` and `FieldFaithful DigestLoopRefusalUp` instances for the same displayed digest, fiber, far-end socket, loop-refusal, exactness-refusal, transport, replay, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DigestLoopRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{DigestLoopRefusal}^{\uparrow}$ packet is generated from a visible digest row, fiber ledger, apophatic far-end socket, loop-refusal row, exactness-refusal ledger, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier first exposes digest and fiber data, then records the far-end socket, and then blocks the loop from digest equality back to source-preimage assertion. The classifier permits visible digest comparison only with the loop-refusal and exactness-refusal rows attached, so digest equality never becomes a hidden source generator.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DigestLoopRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the digest loop refusal carrier, NameCert obligations, no-global-preimage theorem, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DigestLoopRefusal}^{\uparrow}}, \mathsf{unformalized})$

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.digestLoopRefusalDeltaLedger

Bridge status: none

Origin: ai

Not claimed: It does not claim hidden preimage selectors, digest-as-source equality, global source universes, quotient representatives, host inverses, or internal far-end objects.

Upgrade path: To reach obligationClosure, formalize the loop-refusal and exactness-refusal rows as checked blockers for digest-to-source inversion.

Chapter 1084

A Concrete Naming Certificate for $\mathsf{HashLikeFiberBoundary}^{\uparrow}$

The $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ packet realizes the hash-like apophatic fixed-point discipline of chapter 9402 as a composite finite NameCert packet. It sits beside the hash certificate of chapter 242, the digest-fiber ledger of chapter 1067, the apophatic-fixed-point fiber of chapter 1069, and the apophatic fixed-point packet of chapter 1068. Its role is to bind digest visibility, hidden fiber membership, GAP provenance, apophatic far-end refusal, and consumer non-escape as one concrete boundary surface.

Definition 1084.1 (HashLikeFiberBoundary carrier)**.** *A* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, F, G, T, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest row read through the* $\mathsf{Hash}^{\uparrow}$ *certificate;*
2. *F is the hidden fiber ledger row recording source-side membership behind D without identifying the source with the digest;*
3. *G is the* GAP *provenance row for the same fiber ledger;*
4. *T is the apophatic far-end boundary row read through the* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *and* $\mathsf{ApophaticFixedPoint}^{\uparrow}$ *sources;*
5. *R is the refusal row blocking any consumer from treating T as a digest value, source payload, selected preimage, kernel object, or ambient universe object;*
6. *H records componentwise* hsame *transports for D, F, G, T, R, C, P, N;*
7. *C records the displayed* Cont *routes for digest, fiber, GAP, far-end, refusal, transport, provenance, and local naming reads;*
8. *P is the* Pkg *provenance over* $\mathsf{Hash}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HashLikeFiberBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for digest-as-source equality, selected preimage recovery, an internal far-end object, a global universe digest, or an unledgered fiber consumer.

Lean4 (definition) `BEDC.Derived.HashLikeFiberBoundaryUp.HashLikeFiberBoundaryUp`

Theorem 1084.2 (HashLikeFiberBoundary NameCert obligations)**.** *For the carrier of Definition 1084.1, the local* $\mathsf{NameCert}_{\mathsf{HashLikeFiberBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$X = (D, F, G, T, R, H, C, P, N),$$

visible digest exposure through D, hidden fiber-ledger exposure through F, GAP provenance through G, apophatic far-end boundary naming through T, far-end refusal through R, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The rows D, F, G, T, R are the complete concrete boundary content, while H, C, P, N transport, replay, package, and name the same finite packet.

Read the source chapters in their displayed order. The $\mathsf{Hash}^{\uparrow}$ certificate supplies digest visibility, $\mathsf{DigestFiberLedger}^{\uparrow}$ supplies hidden fiber and GAP provenance, $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ supplies the fiber-to-far-end boundary route, and $\mathsf{ApophaticFixedPoint}^{\uparrow}$ supplies the refusal discipline that prevents far-end internalization.

The NameCert surface is exhausted by these rows. Since the carrier has no coordinate for digest-as-source equality, selected preimage recovery, an internal far-end object, global universe digest, or unledgered fiber consumer, no accepted read can obtain any of those objects from the packet. □

Theorem 1084.3 (HashLikeFiberBoundary sibling route)**.** *Every accepted* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ *packet reads visible digest exposure from chapter 242, hidden fiber ledger rows from chapter 1067, and far-end boundary rows from chapter 1069 before it exports the local boundary certificate. Concretely, the rows $D, F, G, T, R, H, C, P, N$ must be displayed as digest visibility, hidden fiber membership, GAP provenance, apophatic far-end boundary, refusal, transport, replay, provenance, and local naming rows for the same finite route.*

Proof. Project D first. The hash certificate supplies only the visible digest exposure; it does not expose a hidden fiber or far-end boundary by itself.

Next project F and G. The digest-fiber ledger supplies hidden membership and GAP provenance behind the visible digest, keeping the source-side row ledgered rather than identified with D.

Finally project T, R, H, C, P, N. The apophatic-fixed-point fiber supplies the far-end boundary row, while the refusal, transport, replay, provenance, and local naming rows keep that boundary attached to the displayed digest-fiber route before the local certificate is exported. □

Theorem 1084.4 (HashLikeFiberBoundary fiber-digest stability)**.** *Any accepted transport of a* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ *packet that preserves the visible digest row D and the hidden fiber ledger row F must preserve the same GAP provenance row G, apophatic boundary row T, and refusal row R through the displayed packet*

$$D, F, G, T, R, H, C, P, N.$$

Thus the fiber-digest route cannot be replayed as a digest-only route, a selected preimage route, or a far-end object route.

Proof. Project the carrier rows of Definition 1084.1. The rows D and F are not free-standing coordinates; they are displayed together with G, T, R, H, C, P, N in one finite packet.

Apply the NameCert obligations of Definition 1084.2. They identify the accepted surface with digest exposure, hidden fiber-ledger exposure, GAP provenance, apophatic far-end boundary naming, refusal, transport, replay, provenance, and local naming for the same rows.

Use noninternalization Definition 1084.5. Once transport preserves D and F, the far-end boundary can still be read only through G, T, R, H, C, P, N, so the transport preserves the GAP, apophatic boundary, and refusal rows rather than creating a digest-only or selected-preimage route. □

Theorem 1084.5 (HashLikeFiberBoundary noninternalization)**.** *Every accepted* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ *consumer that reads the far-end row T receives it only as the apophatic boundary of the displayed fiber behind the visible digest row D. The consumer may replay T through R, H, C, P, N, but it cannot read T as a digest value, source payload, selected preimage, kernel object, global universe digest, host equality, or ambient completeness principle.*

Proof. Start with the carrier inventory in Definition 1084.1. The far-end coordinate is the boundary row T, and it is displayed together with digest, fiber, GAP, and refusal rows.

Apply the NameCert obligations of Definition 1084.2. Any consumer of T must preserve the route through $D, F, G, T, R, H, C, P, N$, so the far-end read cannot detach from the fiber ledger that made it visible.

The refusal row R excludes the forbidden interpretations. A read that treats T as a digest value, payload, selected preimage, object, global digest, host equality, or completeness principle has no coordinate in the carrier and therefore no accepted route. □

The far-end refusal row is consumed only after the digest-fiber ledger of chapter 1067 and the apophatic-fixed-point fiber of chapter 1069 have both been displayed. Thus a consumer reaches the boundary through the fiber ledger and apophatic fiber before it can read the refusal row.

Theorem 1084.6 (HashLikeFiberBoundary consumer route)**.** *Every accepted* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ *consumer reads the displayed rows*

$$D, F, G, T, R, H, C, P, N$$

in that order before any digest, fiber, GAP, far-end boundary, or refusal endpoint is consumed. The route first exposes the $\mathsf{Hash}^{\uparrow}$ *digest row, then the* $\mathsf{DigestFiberLedger}^{\uparrow}$ *fiber row and GAP provenance, then the* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$ *and* $\mathsf{ApophaticFixedPoint}^{\uparrow}$ *far-end boundary and refusal rows, and only then the transport, replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 1084.1. The rows D, F, G, T, R are ordered as digest, hidden fiber, GAP provenance, far-end boundary, and refusal data, while H, C, P, N preserve and name that same route.

Apply the NameCert obligations of Definition 1084.2. They identify local admission with precisely the rows $D, F, G, T, R, H, C, P, N$, so a consumer has no accepted entry point outside the displayed packet.

Finally use noninternalization Definition 1084.5. The far-end row can be consumed only after the digest and fiber ledger have been read and only with the refusal row attached, so the consumer route is exactly the ordered route stated above. □

Theorem 1084.7 (HashLikeFiberBoundary transport stability)**.** hsame *transport and* Cont *replay preserve the same digest, fiber, GAP, far-end, and refusal route of an accepted* HashLikeFiberBoundary$^{\uparrow}$ *packet. Transport may move the rows componentwise, and replay may expose the route again, but neither operation adds selected preimage recovery, digest-as-source equality, internal far-end objects, global universe digests, host equality, or ambient completeness.*

Proof. Start with the consumer route of Definition 1084.6. It fixes the ordered path through $D, F, G, T, R, H, C, P, N$ before any endpoint is consumed.

The carrier row H records componentwise hsame transport for those same rows, and the row C records displayed Cont replay. Therefore transporting or replaying an accepted packet preserves the digest row, the hidden fiber ledger, GAP provenance, far-end boundary, refusal row, provenance, and local naming route.

Apply noninternalization Definition 1084.5. Since the carrier contains no coordinate for selected preimage recovery, digest-as-source equality, internal far-end objects, global universe digests, host equality, or ambient completeness, neither transport nor replay can create such a coordinate. □

Falsifiable prediction: Any accepted HashLikeFiberBoundaryUp consumer that reads the far-end row as a digest value violates the noninternalization theorem.

Independence witness: Nearest siblings: hash, digestfiberledger, apophatic-fixed-point-fiber, and apophaticfixedpoint. None is bijective to HashLikeFiberBoundaryUp because each sibling omits at least one row among digest visibility, hidden fiber membership, GAP provenance, apophatic far-end refusal, and consumer non-escape.

The Lean TasteGate boundary is `lean4/BEDC/Derived/HashLikeFiberBoundaryUp/TasteGate.lean`. It is the intended formal carrier for the displayed digest, fiber, GAP, far-end, refusal, transport, continuation, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.HashLikeFiberBoundaryUp.hashLikeFiberBoundaryChapterTasteGate`

Closure status for NameCert$_{\text{HashLikeFiberBoundary}^{\uparrow}}$.

Constructive story: A HashLikeFiberBoundary$^{\uparrow}$ packet is generated from visible digest, hidden fiber-ledger, GAP provenance, apophatic far-end boundary, far-end refusal, componentwise transport, displayed continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{HashLikeFiberBoundary}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1084.1, Definition 1084.2, Definition 1084.3, Definition 1084.4, Definition 1084.5, Definition 1084.6, and Definition 1084.7 bind the visible digest row, hidden fiber ledger row, GAP provenance row, apophatic far-end boundary row, refusal row, transport row, replay row, provenance row, and local naming row into one closed packet.

Formal verification: FormalStatus(NameCert$_{\text{HashLikeFiberBoundary}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HashLikeFiberBoundaryUp.hashLikeFiberBoundaryChapterTasteGate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close selected preimage recovery, digest-as-source equality, internal far-end objects, global universe digests, or bridge checking.

Upgrade path: Scoped closure requires binding those obligations to Hash$^{\uparrow}$, DigestFiberLedger$^{\uparrow}$, ApophaticFixedPointFiber$^{\uparrow}$, ApophaticFixedPoint$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Chapter 1085

A Concrete Naming Certificate for $\mathsf{DigestInscription}^{\uparrow}$

The $\mathsf{DigestInscription}^{\uparrow}$ packet realizes the digest-surface inscription of chapter 9402 as a finite NameCert surface. It sits beside the hash-like fiber boundary of chapter 1084, the digest-fiber ledger of chapter 1067, the apophatic fixed-point fiber of chapter 1069, the inscription event surface of chapter 519, and the observer-history identity surface of chapter 479. Its role is narrower than those siblings: it binds the visible digest, observation fiber, GAP row, apophatic far end, local self-center inscription, no-subject observer identity, transport, continuation, package, and local name in one displayed packet.

Definition 1085.1 (DigestInscription carrier)**.** *A* $\mathsf{DigestInscription}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, F, G, T, S, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest row read from* $\mathrm{ObsDigest}(\mathcal{C}, t)$*;*
2. *F is the observation-fiber row* $\mathrm{ObsFib}(D)$ *behind the visible digest;*
3. *G is the* GAP *provenance row witnessing that the fiber is displayed rather than inferred from digest equality;*
4. *T is the apophatic far-end boundary row read through* $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$*;*
5. *S is the local self-center inscription row read through the public inscription event surface;*
6. *O is the observer-history identity row preserving the no-subject discipline of the inscription site;*
7. *H records componentwise* hsame *transports for $D, F, G, T, S, O, C, P, N$;*
8. *C records the displayed* Cont *routes for digest, fiber, GAP, far-end, inscription, observer-history, transport, package, and local naming reads;*
9. *P is the* Pkg *provenance over* $\mathsf{HashLikeFiberBoundary}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{ApophaticFixedPointFiber}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DigestInscription}^{\uparrow}}$ *row.*

The carrier contains no coordinate for digest-as-source equality, direct digest-to-T *identification, observer-substance, unledgered fiber membership, private inscription agency, or hidden preimage recovery.*

Lean4 (definition) `BEDC.Derived.DigestInscriptionUp.DigestInscriptionUp`

Definition 1085.2 (DigestInscription classifier)**.** *The* $\mathsf{DigestInscription}^{\uparrow}$ *classifier compares accepted carriers componentwise through* $D, F, G, T, S, O, H, C, P, N$*. Two carriers classify together only when the visible digest, observation fiber, GAP provenance, apophatic far-end boundary, local self-center inscription, observer-history identity, transport rows, continuation routes, package provenance, and local names agree through the displayed packet.*

Theorem 1085.3 (DigestInscription NameCert obligations)**.** *For the carrier of Definition 1085.1, the local* $\mathsf{NameCert}_{\mathsf{DigestInscription}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *visible digest and fiber exposure through* D, F, G*;*
2. *apophatic far-end exposure through* T*;*
3. *local inscription and observer-history exposure through* S, O*;*
4. *componentwise stability through* H, C, P*;*
5. *local naming through* N*.*

Every accepted digest-surface inscription must display all five rows before a consumer may read the visible digest as carrying an inscription site.

Proof. Project the carrier of Definition 1085.1. The rows D, F, G place the digest inside an explicit observation fiber rather than inside source equality.

Read the far-end and inscription rows next. The row T is only an apophatic boundary, while S records the local self-center inscription as a public event row. The observer-history row O keeps that event on the no-subject side.

The remaining rows H, C, P, N transport, replay, package, and name the same packet. Since the carrier has no other coordinate, the five listed obligation rows exhaust the local NameCert surface. □

Theorem 1085.4 (DigestInscription fiber far-end nonescape)**.** *Every accepted* $\mathsf{DigestInscription}^{\uparrow}$ *consumer that reads the apophatic far-end row* T *must first keep the visible digest row* D*, observation-fiber row* F*, GAP row* G*, local inscription row* S*, observer-history row* O*, transport row* H*, continuation row* C*, package row* P*, and local name row* N *in the same displayed packet. Hence the far end cannot be exported as a digest value, source payload, selected preimage, observer subject, private inscription agent, or BEDC kernel object.*

Proof. Start with the classifier of Definition 1085.2. It permits comparison only through the displayed rows $D, F, G, T, S, O, H, C, P, N$.

Apply the obligation theorem. The far-end row belongs to the second obligation row, but the first and third rows force it to remain attached to the digest fiber, GAP provenance, local inscription, and observer-history surface.

Finally inspect the carrier exclusions in Definition 1085.1. There is no coordinate for a digest value equal to T, a source payload, selected preimage, observer subject, private inscription agent, or kernel object. Any such consumer therefore leaves the accepted packet. □

Falsifiable prediction: An accepted DigestInscriptionUp consumer refutes the packet if it reads the far-end row without the visible digest, observation-fiber, GAP, local-inscription, and observer-history rows in the same NameCert surface.

Independence witness: hashlikefiberboundary, digestfiberledger, apophaticfixedpointfiber, inscriptionevent, observerhistidentity: no sibling is bijective to DigestInscriptionUp because only this packet carries visible digest, observation fiber, GAP provenance, apophatic far end, local self-center inscription, observer-history identity, transport, continuation, package, and local NameCert rows together.

The Lean TasteGate boundary is `lean4/BEDC/Derived/DigestInscriptionUp/TasteGate.lean`. It must provide `Nontrivial DigestInscriptionUp` and `FieldFaithful DigestInscriptionUp` instances preserving $D, F, G, T, S, O, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{DigestInscription}^{\uparrow}}$.

Constructive story: A $\mathsf{DigestInscription}^{\uparrow}$ packet is generated from visible digest, observation-fiber, GAP, apophatic far-end, local self-center inscription, observer-history identity, componentwise hsame transport, displayed Cont routes, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DigestInscription}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1085.1, Definition 1085.2, Definition 1085.3, Definition 1085.4, the falsifiable prediction, and the independence witness bind the finite digest-inscription packet and its fiber far-end nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DigestInscription}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close digest-as-source equality, direct digest-to-T identification, hidden preimage recovery, observer-substance, private inscription agency, Lean verification, or bridge checking for $\mathsf{DigestInscription}^{\uparrow}$.

Upgrade path: Obligation closure requires Lean encoding of the carrier, classifier, NameCert obligation surface, fiber far-end nonescape boundary, TasteGate nontriviality, and field faithfulness for $D, F, G, T, S, O, H, C, P, N$.

Chapter 1086

A Concrete Naming Certificate for $\mathsf{InscriptionEventReplay}^{\uparrow}$

The $\mathsf{InscriptionEventReplay}^{\uparrow}$ packet realizes the replay-only boundary named in chapter 9397. It sits downstream of the agency-bearing inscription surface of chapter 519, consumes the accepted route exposed by chapter 582, and offers deterministic checker replay to the route-audit surface of chapter 1005. The point of the packet is negative as much as positive: replay is valuable audit evidence, but it is not the technical inscription event unless the carrier also contains the acceptance row that selects the route as the record to inscribe.

Falsifiable prediction: A replay packet whose carrier has no stored accepted public route is not an accepted $\mathsf{InscriptionEventReplay}^{\uparrow}$ carrier.

Independence witness: inscriptionevent, inscribedroute, inscriptionrouteaudit, inscriptiongapledger, finitewindowobserver; the replay carrier is not bijective to any sibling because it requires a pre-existing accepted route and checker replay row while excluding the inscription-acceptance coordinate that the inscription-event siblings carry.

Definition 1086.1 (Inscription-event replay carrier)**.** *An* $\mathsf{InscriptionEventReplay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, M, A, K, D, H, C, P, N)$$

with the following displayed rows:

1. *G is the visible gap row inherited from the source boundary of the earlier inscription event;*
2. *M is the named public record whose route has already been accepted;*
3. *A is the stored accepted public route row from the prior inscription carrier;*
4. *K is the deterministic checker replay row for that stored route;*
5. *D is the route-audit readback row exposing G, M, A, K to downstream consumers;*
6. *H records componentwise* hsame *transports for the gap, name, accepted route, replay, readback, provenance, and local naming rows;*
7. *C records the displayed* Cont *routes by which consumers replay the accepted route and inspect the audit readback;*
8. *P records* Pkg *provenance over* $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscribedRoute}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, $\mathsf{FiniteWindowObserver}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{InscriptionEventReplay}^{\uparrow}}$ *row.*
The carrier has no coordinate for the inscription-facing acceptance act by which a conscious observational system first selected A as the record to inscribe. It therefore records deterministic replay of an already accepted route, not technical inscription-event agency.

Lean4 (definition) `BEDC.Derived.InscriptionEventReplayUp.InscriptionEventReplayUp`

Theorem 1086.2 (Inscription-event replay NameCert obligations)**.** *For the carrier of Definition 1086.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionEventReplay}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (G, M, A, K, D, H, C, P, N),$$

stored accepted-route exposure through A, deterministic checker replay through K, route-audit readback through D, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. The surface does not include the conscious acceptance row that belongs to the inscription-event carrier of chapter 519.*

Proof. Project the carrier of Definition 1086.1. Its public rows are the visible gap, named record, stored accepted route, checker replay, audit readback, transport, continuation, provenance, and local naming rows.

Read the dependency order. The replay row K is meaningful only after the accepted public route A is already stored, and the readback row D exposes the pair A, K to the audit consumers.

The structural rows H, C, P, N transport, replay, record, and name that same finite surface. Since no coordinate records the first public acceptance of A as the route to inscribe, the NameCert obligations close replay audit only. □

Theorem 1086.3 (Inscription-event replay noninscription boundary)**.** *Every accepted* $\mathsf{InscriptionEventReplay}^{\uparrow}$ *packet separates deterministic proof replay from technical inscription-event agency. A consumer may read the visible gap G, named record M, stored accepted route A, checker replay K, and route-audit readback D, but it cannot recover the inscription-facing acceptance coordinate that made A the public record in the first place.*

Proof. Begin with the carrier inventory. The row A stores an accepted public route and the row K replays it deterministically, but the tuple contains no row for the acceptance act itself.

Apply Definition 1086.2. The local certificate exposes replay obligations for A, K, D, H, C, P, N and does not enlarge the carrier with a second agency-bearing coordinate.

Therefore a replay consumer can audit the route once it is present, while the inscription-event acceptance boundary remains in the sibling event carrier rather than in the replay packet. □

Theorem 1086.4 (Inscription-event replay consumer non-escape)**.** *Every* $\mathsf{InscriptionEventReplay}^{\uparrow}$ *consumer factors through the finite replay route*

$$G \longmapsto M \longmapsto A \longmapsto K \longmapsto D \longmapsto H, C, P, N.$$

The consumer receives no hidden route choice, off-packet checker, private source predicate, unrecorded acceptance act, or public-record export outside the displayed replay rows.

Proof. Use the carrier definition to read the replay route in order. The gap and named record identify the inherited inscription boundary, the stored route A supplies the object to replay, and the checker row K performs deterministic replay before D exposes the audit readback.

Next apply Definition 1086.3. It removes the only tempting bypass: treating the replay row as if it were the original acceptance row.

The remaining structural rows only transport, replay, package, and name the displayed route. Hence every consumer factors through the listed packet and cannot escape into hidden selection, private source content, or unrecorded agency. □

The Lean follow-up must provide `Nontrivial InscriptionEventReplayUp` and `FieldFaithful InscriptionEventReplayUp` in `lean4/BEDC/Derived/InscriptionEventReplayUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionEventReplay}^{\uparrow}}$.

Constructive story: An $\mathsf{InscriptionEventReplay}^{\uparrow}$ packet is generated from a visible gap row, named public record row, stored accepted-route row, deterministic checker replay row, audit readback row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionEventReplay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1086.1, Definition 1086.2, Definition 1086.3, and Definition 1086.4 bind the replay-only carrier, deterministic checker route, noninscription boundary, and consumer non-escape surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionEventReplay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the original inscription acceptance act, private agency, hidden route selection, Lean verification, bridge checking, or public export outside the displayed replay packet.

Upgrade path: Obligation closure requires checked carrier admission, accepted-route exposure, deterministic replay readback, route-audit exactness, consumer non-escape, and TasteGate Nontrivial and FieldFaithful witnesses.

Chapter 1087

A Concrete Naming Certificate for $\mathsf{HashDigestFiberBoundary}^{\uparrow}$

The $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ packet realizes the digest/fiber boundary isolated in chapter 9402 as a finite NameCert surface. It sits beside the hash-apophatic seal of chapter 1079, the digest-fiber ledger of chapter 1067, and the hash-like fiber boundary of chapter 1084. Its role is narrower than those siblings: it names the boundary at which visible digest sameness, source-fiber membership, GAP provenance, and refusal of source equality must be read together.

Definition 1087.1 (HashDigestFiberBoundary carrier)**.** *A* $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, E, F, G, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest row read through the* $\mathsf{Hash}^{\uparrow}$ *surface;*
2. *E is the digest-sameness row, recording classifier-level sameness of visible digest tokens;*
3. *F is the fiber-membership row, recording the source-side membership behind D without identifying members of the fiber;*
4. *G is the* GAP *provenance row for the same fiber boundary;*
5. *R is the refusal row blocking any consumer from turning E into source equality or selected preimage recovery;*
6. *H records componentwise* hsame *transports for D, E, F, G, R, C, P, N;*
7. *C records displayed* Cont *routes for digest, digest-sameness, fiber, GAP, refusal, transport, provenance, and local naming reads;*
8. *P is the* Pkg *provenance over* $\mathsf{Hash}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{HashLikeFiberBoundary}^{\uparrow}$, $\mathsf{HashApophaticSeal}^{\uparrow}$, $\mathsf{ApophaticFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HashDigestFiberBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for source equality obtained from digest sameness, no hidden preimage selector, no quotient collapse of the fiber, no global digest, and no unledgered source read.

Theorem 1087.2 (HashDigestFiberBoundary NameCert obligations)**.** *For the carrier of Definition 1087.1, the local* $\mathsf{NameCert}_{\mathsf{HashDigestFiberBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission*

for

$$B = (D, E, F, G, R, H, C, P, N),$$

visible digest exposure through D, digest-sameness exposure through E, fiber membership through F, GAP provenance through G, source-equality refusal through R, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Every accepted consumer of the packet must preserve the boundary order: digest sameness is read before source equality is even meaningful, and source equality remains unavailable unless a separate exactness certificate is displayed outside the $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.HashDigestFiberBoundaryUp.HashDigestFiberBoundaryTasteGate single carrier alignment`

Proof. Project the displayed carrier in Definition 1087.1. The rows D, E, F, G, R are the complete digest/fiber boundary content, while H, C, P, N transport, replay, package, and name the same finite packet.

Read the sibling sources through their local roles. The $\mathsf{Hash}^{\uparrow}$ surface supplies the visible digest, $\mathsf{DigestFiberLedger}^{\uparrow}$ supplies the fiber and GAP rows, $\mathsf{HashLikeFiberBoundary}^{\uparrow}$ supplies the consumer boundary discipline, $\mathsf{HashApophaticSeal}^{\uparrow}$ supplies the digest/fiber/far-end seal pattern, and $\mathsf{ApophaticFixedPoint}^{\uparrow}$ supplies the refusal of internalized far-end objects.

The NameCert surface is exhausted by these rows. Since the carrier has no coordinate for source equality from digest sameness, selected preimage recovery, quotient collapse, global digest, or unledgered source read, an accepted consumer cannot obtain any of those readings from the packet. □

Theorem 1087.3 (HashDigestFiberBoundary nonescape)**.** *Every accepted* $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *consumer that reads digest sameness through E must also keep the visible digest row D, fiber row F, GAP provenance row G, refusal row R, componentwise transport H, continuation routes C, package provenance P, and local naming row N. Removing the fiber and GAP provenance rows while keeping digest sameness collapses the packet's ability to distinguish digest equality from source equality.*

Consequently the packet exports no source-equality reflection, selected preimage, quotient fiber, global universe digest, hidden source payload, or unledgered source read.

Lean4 ✓ `BEDC.Derived.HashDigestFiberBoundaryUp.HashDigestFiberBoundary nonescape`

Proof. Start from an accepted packet $B = (D, E, F, G, R, H, C, P, N)$. The digest-sameness row E is only a classifier-facing comparison of visible digest rows, while F and G record the source-side fiber membership and provenance that keep the boundary auditable.

Apply the NameCert obligations of Definition 1087.2. They force any accepted read of E to remain packaged with the digest, fiber, GAP, refusal, transport, continuation, provenance, and local naming rows.

If F and G are removed while E remains, the remaining carrier does not record which source-side fiber is being compared. The packet therefore has no displayed way to separate digest equality from source equality. Since source-equality reflection, selected preimage recovery, quotient collapse, global digest, hidden source payload, and unledgered source reads have no carrier coordinate, none can escape the displayed packet. □

Theorem 1087.4 (HashDigestFiberBoundary field-faithful obligation)**.** *For an accepted* $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *packet*

$$B = (D, E, F, G, R, H, C, P, N),$$

the field-faithful obligation surface is visible in the displayed rows. The digest row D, digest-sameness row E, fiber row F, GAP row G, refusal row R, transport row H, continuation row C, provenance row P, and naming row N are the fields whose carrier alignment is cited by the checked TasteGate marker attached to Definition 1087.2.

Proof. Project the carrier rows of Definition 1087.1. Each field named in the obligation is one coordinate of B, and no row is supplied by ambient digest theory or by a hidden source read.

The checked TasteGate marker attached to Definition 1087.2 records the same single-carrier alignment. Hence field-faithfulness is a paper obligation over the concrete rows $D, E, F, G, R, H, C, P, N$, not a prose property outside the packet. □

Theorem 1087.5 (HashDigestFiberBoundary nontrivial obligation)**.** *The nontriviality obligation for* $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *is the joint presence of*

$$D, E, F, G, R, H, C, P, N.$$

A consumer may not retain only digest sameness E: the visible digest D, fiber membership F, GAP provenance G, refusal R, componentwise transport H, continuation replay C, package provenance P, and local naming N are the concrete rows that make the TasteGate carrier nontrivial.

Proof. Use Definition 1087.3. It already shows that a read of E must keep the visible digest, fiber, GAP, refusal, transport, continuation, provenance, and naming rows.

The TasteGate marker attached to the NameCert obligations therefore has a visible nontrivial carrier to check: it is precisely the row tuple $D, E, F, G, R, H, C, P, N$. Removing any of F, G, R would collapse the boundary into digest-sameness prose, while removing H, C, P, N would leave no transported, replayable, packaged, and named certificate. □

Theorem 1087.6 (HashDigestFiberBoundary obligation scope)**.** *The scoped carrier obligations of* $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ *are exactly the obligations carried by*

$$B = (D, E, F, G, R, H, C, P, N).$$

They include field-faithful visibility of the displayed rows, nontrivial joint presence of digest, fiber, GAP, and refusal data, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming; they do not include source-equality reflection, preimage selection, quotient collapse, a global digest, or an unledgered source read.*

Proof. Combine the row projection in Definition 1087.4 with the nontriviality projection in Definition 1087.5. Both are statements about the same accepted carrier B.

The checked TasteGate marker cited at the NameCert obligation site aligns the single carrier with those rows. The scope is therefore the finite digest/fiber boundary packet itself, and the excluded readings remain outside the obligation surface because no coordinate of B names them. □

Falsifiable prediction: Removing the fiber and GAP provenance rows while keeping digest sameness collapses the chapter's ability to distinguish digest equality from source equality.

Independence witness: Nearest siblings: hashapophaticseal, digestfiberledger, hashlikefiberboundary, hash, and apophaticfixedpoint. No bijection exists because each sibling omits at least one row among explicit digest-sameness exposure, fiber membership, GAP provenance, source-equality refusal, and local NameCert packaging for the digest/fiber boundary.

The Lean boundary `lean4/BEDC/Derived/HashDigestFiberBoundaryUp/TasteGate.lean` must provide `Nontrivial HashDigestFiberBoundaryUp` and `FieldFaithful HashDigestFiberBoundaryUp` instances for the displayed digest, digest-sameness, fiber, GAP, refusal, transport, continuation, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HashDigestFiberBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HashDigestFiberBoundary}^{\uparrow}$ packet is generated from visible digest exposure, digest-sameness comparison, source-side fiber membership, GAP provenance, source-equality refusal, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier first reads digest equality as a visible comparison, then keeps the source-side fiber and gap rows attached so that equality of digest rows is not confused with equality of sources. The classifier accepts the boundary only when digest, fiber, gap, and refusal data remain present as one finite NameCert packet.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HashDigestFiberBoundary}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1087.6 binds the displayed carrier rows together with the checked TasteGate marker attached to Definition 1087.2.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HashDigestFiberBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim source-equality reflection, selected preimage recovery, quotient collapse, global digest universes, hidden source payloads, or unledgered source reads.

Upgrade path: To strengthen the verification axis, formalize the digest-sameness, fiber, gap, refusal, field-faithful, nontriviality, and scoped obligation rows as a checked boundary.

Chapter 1088

ForceInterHistRelationUp NameCert construction

The $\mathsf{ForceInterHistRelation}^{\uparrow}$ packet realizes the force-like relation interface of chapter 9416 as finite relation-certificate data over inter-Hist rows. It consumes the invariant gate of chapter 656, the locality ledger of chapter 759, the audit-map family ledger of chapter 810, and the observer bridge ledger of chapter 1032. Its carrier is not a physical force law, metric tensor, or field equation; it is the BEDC packet that says which inter-Hist relation rows may be exported as force-like relation claims.

Falsifiable prediction: A force-like relation packet without a displayed non-preserved-content row cannot pass the relation gate as an exported BEDC claim.

Independence witness: interhistinvariant, interhistlocalityledger, auditmapfamilyledger, observerbridgeledger: none is bijective to $\mathsf{ForceInterHistRelation}^{\uparrow}$ because this packet carries source rows, a same-pattern classifier, a layer-map row, a non-preserved-content refusal row, a relation ledger, bridge-read rows, audit-map provenance, and local NameCert data together, while each sibling supplies only one gate, route, ledger, or bridge surface.

Definition 1088.1 (Force-like inter-Hist relation carrier)**.** *A* $\mathsf{ForceInterHistRelation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, K, L, R, U, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is a finite source row of inter-*Hist *change sites admitted by the surrounding multi-history ledger;*
2. *K is the same-force-pattern classifier row comparing the visible relation packets over S;*
3. *L is the layer-map row that records which source, carrier, classifier, ledger, exactness, and refusal layers are being compared;*
4. *R is the non-preserved-content refusal row required by the layered relation gate;*
5. *U is the* $\mathsf{InterHistInvariant}^{\uparrow}$ *row that prevents observer-slot or invariant collapse;*
6. *A is the* $\mathsf{InterHistLocalityLedger}^{\uparrow}$ *row that routes the compared histories through a finite locality ledger;*
7. *B is the* $\mathsf{ObserverBridgeLedger}^{\uparrow}$ *row that records the observer-side bridge read used by the relation packet;*

8. *H records componentwise* hsame *transport for the source, classifier, layer-map, refusal, invariant, locality, bridge, audit, provenance, and naming rows;*
9. *C records the displayed* Cont *routes for every relation-certificate consumer read;*
10. *P is the* Pkg *provenance over* $\mathsf{ForceInterHistRelation}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{AuditMapFamilyLedger}^{\uparrow}$, $\mathsf{ObserverBridgeLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}$ *row.*

*The carrier accepts a force-like inter-*Hist *relation only when the source, classifier, layer-map, refusal, invariant, locality, observer-bridge, transport, route, provenance, and local naming rows are displayed together. It exports no hidden force cause, empirical constant, global frame, metric tensor, field equation, subject-substance row, or source-free relation primitive.*

Lean4 ✓ `BEDC.Derived.ForceInterHistRelationUp.ForceInterHistRelationTasteGate single carrier alignment`

Theorem 1088.2 (Force-like relation NameCert obligations)**.** *For the carrier of Definition 1088.1, the local* $\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (S, K, L, R, U, A, B, H, C, P, N),$$

source exposure through S, same-pattern classification through K, layer-map exposure through L, non-preserved-content refusal through R, invariant preservation through U, locality-ledger routing through A, observer-bridge readback through B, componentwise hsame *transport through H, displayed continuation replay through C, audit-map provenance through P, and local naming through N.*

Proof. Project an accepted carrier to the eleven rows of Definition 1088.1. The source row S, classifier row K, layer-map row L, and refusal row R are the relation-certificate rows required before any force-like reading is available.

The invariant row U, locality ledger A, and observer bridge ledger B bind the relation to the already displayed inter-Hist and observer routes. Componentwise transport H and continuation replay C preserve those rows as rows of the same packet.

The provenance row P records the audit-map family ledger together with the invariant, locality, and bridge siblings, while N names the local certificate. Since the carrier has no coordinate outside these rows, the listed obligations exhaust the exported $\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}$ surface. □

Theorem 1088.3 (Force-like relation refusal non-escape)**.** *No accepted* $\mathsf{ForceInterHistRelation}^{\uparrow}$ *consumer can export a force-like inter-*Hist *relation as a BEDC claim unless the same carrier displays the non-preserved-content refusal row R together with the source row S, classifier row K, layer-map row L, invariant row U, locality-ledger row A, observer-bridge row B, transport row H, continuation row C, provenance row P, and local name row N.*

Proof. The relation gate of Definition 9416.19 asks whether preserved and non-preserved content are both displayed. In the carrier of Definition 1088.1, the displayed non-preserved-content coordinate is exactly R.

If a consumer omits R, then the read may preserve the source and classifier rows but lacks the refusal row that prevents identity collapse, hidden empirical law, global-frame import, or subject-substance import. Such a read is not a route through the full packet F.

If a consumer omits any of $S, K, L, U, A, B, H, C, P, N$, then it loses the source, classifier, layer, invariant, locality, observer bridge, transport, continuation, provenance, or naming row needed by

the local NameCert obligations. Hence every exported force-like relation claim must pass through the displayed packet and its refusal row. □

The Lean follow-up for `BEDC.Derived.ForceInterHistRelationUp` must provide explicit Nontrivial and FieldFaithful instances preserving the source, classifier, layer-map, refusal, invariant, locality-ledger, observer-bridge, transport, continuation, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}$.

Constructive story: A $\mathsf{ForceInterHistRelation}^{\uparrow}$ packet is generated from source rows, a same-force-pattern classifier, a layer map, a non-preserved-content refusal row, inter-Hist invariant and locality rows, an observer bridge row, componentwise transport, displayed continuation routes, audit-map provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1088.1, Definition 1088.2, and Definition 1088.3 specify the finite force-like relation carrier, its local NameCert obligation surface, and the refusal row that blocks over-reading as identity, hidden cause, empirical law, or global frame.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForceInterHistRelation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a physical force law, empirical constant, metric tensor, field equation, global frame, subject-substance carrier, Lean verification, or bridge checking for $\mathsf{ForceInterHistRelation}^{\uparrow}$.

Upgrade path: Scoped closure requires checked carrier admission, relation-gate refusal, invariant preservation, locality-ledger routing, observer-bridge readback, audit-map provenance, and TasteGate nontriviality with field faithfulness for the same packet.

Chapter 1089

A Concrete Naming Certificate for $\mathsf{LayeredRelationSite}^{\uparrow}$

The $\mathsf{LayeredRelationSite}^{\uparrow}$ packet realizes the layered relation certificate of chapter 9416 as finite BHist data. It sits beside the frame-refusal audit of chapter 1037, the observer classifier of chapter 986, the local gluing surface of chapter 151, and the compiler route of chapter 473. Those siblings supply frame refusal, observer comparison, cover-local transport, and certificate movement; this packet binds relation rows, refinement chains, locality gates, and refusal ledgers as one NameCert site.

Falsifiable prediction: If the LayeredRelationSiteUp carrier is atomic, then any reduction to RelationalFrameAuditUp loses either the refinement-chain row or the locality-gate refusal row.

Independence witness: relationalframeaudit, observerhsameclassifier, sheaf, certificatecompiler: none is bijective to $\mathsf{LayeredRelationSite}^{\uparrow}$ because the site carrier carries source-site rows, relation rows, refinement-chain rows, locality-gate refusals, transport, replay, provenance, and local NameCert rows in one packet. RelationalFrameAuditUp carries frame refusal without the refinement chain, ObserverHsameClassifierUp carries observer comparison without the source-site ledger, SheafUp carries locality and gluing without the relation-refusal ledger, and CertificateCompilerUp carries certificate movement without the layered relation site.

Definition 1089.1 (LayeredRelationSite carrier)**.** *A* $\mathsf{LayeredRelationSite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S_A, S_B, R_A, R_B, J, G, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S_A and S_B are the two source-site rows whose relation is being certified;*
2. *R_A and R_B are the relation rows read from the two sites, including the displayed classifier surfaces consumed by the comparison;*
3. *J is the refinement-chain row recording the finite layer correspondence and the obstruction rows that force each classifier split;*
4. *G is the locality-gate row deciding which relation reads are admitted locally and which requests are refused;*
5. *T records the relation transport row that moves accepted source-site comparisons componentwise through* hsame*;*

6. *E records replay rows for relation reads, refinement-chain steps, locality-gate decisions, and refusal ledger entries;*
7. *H records componentwise* hsame *transport for $S_A, S_B, R_A, R_B, J, G, T, E, P, N$;*
8. *C records the displayed* Cont *routes through which downstream consumers replay relation rows, refinement rows, locality decisions, and refusal rows;*
9. *P is the* Pkg *provenance over* BHist, hsame, Cont, RelationalFrameAudit$^{\uparrow}$, ObserverHsameClassifier$^{\uparrow}$, Sheaf$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, *and local relation-site dependencies;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationSite}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these rows. It has no coordinate for object identity between the two sites, global-frame synchronization, host powerset equality, unrecorded compiler semantics, empirical physical force, or subject-substance.

Lean4 (definition) `BEDC.Derived.LayeredRelationSiteUp.LayeredRelationSiteUp`

Theorem 1089.2 (LayeredRelationSite NameCert obligations)**.** *For the carrier of Definition 1089.1, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationSite}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier exposure: every accepted packet exposes exactly $L = (S_A, S_B, R_A, R_B, J, G, T, E, H, C, P, N)$;*
2. *relation-row admission: each comparison is read through the displayed source-site and relation rows S_A, S_B, R_A, R_B;*
3. *refinement-chain exactness: the layer map and obstruction-to-classifier-split data are exactly the row J;*
4. *locality-gate refusal: every nonlocal or over-reading request is routed to the displayed gate and refusal row G;*
5. *componentwise transport and replay: relation rows, refinement-chain rows, gate decisions, refusal ledgers, package provenance, and local naming are transported only by T, H and replayed only by E, C.*

Proof. Carrier exposure is projection from the finite tuple in Definition 1089.1. The accepted packet has no fields outside $S_A, S_B, R_A, R_B, J, G, T, E, H, C, P, N$.

Relation-row admission is the projection to the two source-site rows and their relation rows. The packet can compare only the displayed R_A and R_B surfaces over S_A and S_B, so it cannot identify the source sites as objects.

Refinement-chain exactness is the projection to J. The chain row carries the layer map and the obstruction rows that force classifier splits, matching the certificate discipline of chapter 9416.

Locality-gate refusal is the projection to G. A request outside the displayed relation read or locality site has no accepted output row except the gate refusal.

Finally, transport and replay are componentwise. The row T records relation transport, H records hsame transport over all carrier coordinates, and E, C replay the same route with the package and local naming rows P, N. □

Theorem 1089.3 (LayeredRelationSite independence boundary)**.** *The siblings* RelationalFrameAudit$^{\uparrow}$, ObserverHsameClassifier$^{\uparrow}$, Sheaf$^{\uparrow}$, *and* CertificateCompiler$^{\uparrow}$ *are independence witnesses for* LayeredRelationSite$^{\uparrow}$. *No bijection from any one sibling carrier to the carrier of Definition 1089.1 preserves all carrier rows and all five obligations of Definition 1089.2.*

Proof. Compare RelationalFrameAudit$^{\uparrow}$ first. It supplies the no-global-frame and refusal discipline of chapter 1037, but it has no row that simultaneously records the layered refinement chain J and the relation-site locality gate G.

Compare ObserverHsameClassifier$^{\uparrow}$ next. It supplies an observer-history classifier and transport route, but it does not carry the two source-site rows, the paired relation rows, or the obstruction-driven refinement chain required by the relation site.

The Sheaf$^{\uparrow}$ sibling supplies local gluing and cover-sensitive transport, while CertificateCompiler$^{\uparrow}$ supplies finite certificate movement. Neither sibling carries the paired relation rows together with the gate-refusal ledger and the layered relation provenance row.

Thus any attempted bijection loses at least one carrier coordinate or one NameCert obligation: relation admission, refinement-chain exactness, locality-gate refusal, or componentwise relation replay. The LayeredRelationSite$^{\uparrow}$ carrier is therefore not a renamed copy of any listed sibling. □

Theorem 1089.4 (LayeredRelationSite downstream gate target)**.** *The Lean-facing downstream surface for* LayeredRelationSite$^{\uparrow}$ *is the packet family recorded under* Derived/LayeredRelationSiteUp/TasteGate.lean *with explicit* Nontrivial *and* FieldFaithful *instances. The gate checks that the source-site rows, relation rows, refinement-chain row, locality-gate refusal row, transport, replay, provenance, and local naming row remain recoverable from the admitted packet.*

Proof. The carrier and NameCert obligations already name the row inventory the gate must inspect. Nontriviality is witnessed by two packets differing in a visible source-site row. Field faithfulness is witnessed by projection of all carrier rows from the constructor.

The downstream gate therefore does not add a new relation semantics. It records that the finite site packet has a real display, that every displayed field can be recovered from that display, and that the relation-site carrier is not a singleton placeholder. □

Lean4 ✓ `BEDC.Derived.LayeredRelationSiteUp.taste gate`

Closure status for NameCert$_{\text{LayeredRelationSite}^{\uparrow}}$.

Constructive story: A LayeredRelationSite$^{\uparrow}$ packet is generated from two source-site rows, paired relation rows, a refinement-chain row, a locality-gate refusal row, relation transport, replay, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LayeredRelationSite}^{\uparrow}}$, seed)

Scope closed: Definition 1089.1, Definition 1089.2, and Definition 1089.3 bind the layered relation certificate as a finite relation-site NameCert packet over source-site rows, relation rows, refinement chains, locality gates, refusal ledgers, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{LayeredRelationSite}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.LayeredRelationSiteUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close object identity between related sites, empirical physics, subject-substance, global-frame synchronization, host powerset equality, compiler semantics, or bridge checking.

Upgrade path: Scoped closure requires checked paper-side exactness for relation admission, refinement-chain replay, locality-gate refusal, dependency provenance, and downstream consumer boundaries over the same finite rows.

Chapter 1090

A Relation-Before-Identity Surface for $\mathsf{LayeredRelationCert}^{\uparrow}$

The $\mathsf{LayeredRelationCert}^{\uparrow}$ packet realizes the relation-before-identity discipline of chapter 9416 as a finite NameCert surface. It sits beside the classifier-transport row of chapter 590, the two-history invariant row of chapter 656, the generated-recognizer flow of chapter 1048, the finite reflection tuple of chapter 993, and the audit-gate composition of chapter 696. Those siblings expose morphism, invariant, recognizer-flow, tuple, and gate-composition surfaces; this chapter binds two displayed refinement chains, their relation classifier, refusal rows, ledger rows, exactness row, failure boundary, and strength row as one consumer packet.

Falsifiable prediction: Every accepted LayeredRelationCert packet lacking a not-preserved row fails the public exactness export test.

Independence witness: classifiermorphism, interhistinvariant, groundcompilerrecognizerflow, finitereflectiontuple, auditgatecomposition: none is bijective with LayeredRelationCertUp because this carrier has two displayed refinement chains plus preserved, not-preserved, exactness, failure-boundary, and strength rows, while the siblings each bind only one morphism, invariant, recognizer-flow, tuple, or gate-composition surface.

Definition 1090.1 (LayeredRelationCert carrier)**.** *A* $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, M, Q, R, U, E, F, S, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two refinement-chain rows, each carrying source, layer, obstruction, carrier, and local classifier data;*
2. *M is the displayed layer-map row pairing layers of A with layers of B;*
3. *Q is the relation-classifier row, accepting same-schema reads only through A, B, M;*
4. *R is the preserved-structure row;*
5. *U is the not-preserved row, listing the object content, empirical content, proof content, bridge content, or subject-side content refused by the comparison;*
6. *E is the exactness row, recording the public meaning of same-schema export;*
7. *F is the failure-boundary row, naming the condition under which the relation weakens or cannot be exported;*

8. *S is the strength row, separating paper certificate, checked certificate, and bridge certificate readings;*
9. *H records componentwise* hsame *transport for chain, layer-map, classifier, preserved, not-preserved, exactness, failure-boundary, strength, route, provenance, and naming rows;*
10. *C records displayed* Cont *routes for source-chain projection, layer-map inspection, classifier admission, refusal inspection, exactness export, and public consumer handoff;*
11. *P is the* Pkg *provenance over chapter 9416, chapter 590, chapter 656, chapter 1048, chapter 993, chapter 696,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for object identity between the two domains, hidden empirical law, unmediated proof transfer, subject-substance transfer, bridge certificate, global frame, or unlisted relation strength.

Lean4 (definition) `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertUp`

Definition 1090.2 (LayeredRelationCert classifier)**.** *The* $\mathsf{LayeredRelationCert}^{\uparrow}$ *classifier accepts a carrier $L = (A, B, M, Q, R, U, E, F, S, H, C, P, N)$ only when all public relation reads factor through the chain rows A, B, the layer-map row M, the relation-classifier row Q, the preserved row R, the not-preserved row U, the exactness row E, the failure-boundary row F, and the strength row S, with transport, continuation, provenance, and naming supplied by H, C, P, N.*

Two accepted carriers are classifier-same only when they present the same source-chain boundaries, the same displayed layer-map boundary, the same preserved and not-preserved rows, the same exactness boundary, the same failure-boundary row, and the same strength row after componentwise hsame *transport. The classifier never compares the two domains as objects.*

Theorem 1090.3 (LayeredRelationCert field exhaustion)**.** *For every accepted* $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier*

$$L = (A, B, M, Q, R, U, E, F, S, H, C, P, N),$$

every public relation read is exhausted by the displayed rows $A, B, M, Q, R, U, E, F, S, H, C, P, N$. In particular, a same-schema export can read preserved structure only through R, refused content only through U, public exactness only through E, failure behavior only through F, and certification strength only through S.

Proof. Project the accepted carrier of Definition 1090.1. The tuple displays the two chain rows, layer map, classifier, preserved row, not-preserved row, exactness row, failure-boundary row, strength row, transport, continuation, provenance, and local name.

Use the classifier of Definition 1090.2. It admits public reads only after those rows are present and transported to the same finite surface, so the relation cannot acquire a second source for preservation, refusal, exactness, failure, or strength.

The structural rows H, C, P, N replay and name the same finite packet. Since they do not add a domain-identity coordinate, bridge coordinate, proof-transfer coordinate, or empirical-law coordinate, the row inventory exhausts the public relation surface. □

Theorem 1090.4 (LayeredRelationCert refusal non-escape)**.** *For every accepted* $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier, all non-preserved content is confined to the row U and the failure-boundary row F. The packet cannot export refused content as preserved structure, cannot promote same-schema exactness to object identity, and cannot turn paper-certificate strength into checked or bridge strength.*

Proof. Start with the carrier inventory. The only row assigned to not-preserved content is U, and the only row assigned to failure behavior is F.

Apply Definition 1090.3. A public read of preserved structure must pass through R, a public exactness read must pass through E, and a public strength read must pass through S. None of those rows is a constructor from refused content to preserved content.

Finally inspect the classifier row Q. It accepts same-schema reads through the displayed chain and layer-map surface, not object identity between the domains. Hence refusal remains visible to every consumer and cannot escape through transport, continuation, provenance, local naming, or strength language. □

Lean4 ✓ `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCert refused row concrete boundary`

Theorem 1090.5 (LayeredRelationCert public exactness test)**.** *An accepted* $\mathsf{LayeredRelationCert}^{\uparrow}$ *packet exports public exactness only when the not-preserved row U is present before the exactness row E is read. If the packet lacks U, then the exactness export has no displayed refusal surface and fails the public exactness export test.*

Proof. Project the accepted packet and read the exactness route through C. The route reaches E only after the carrier has displayed the chain rows, layer map, classifier row, preserved row, and not-preserved row.

The role of U is fixed by Definition 1090.1: it lists the content that the relation refuses to preserve. Without that row, same-schema language has no public boundary separating relation exactness from identity.

Apply Definition 1090.4. Since refusal is the guard against over-reading, an exactness export without U lacks the row needed to keep non-preserved content from escaping as preserved content. □

Theorem 1090.6 (LayeredRelationCert NameCert obligation surface)**.** *For the carrier of Definition 1090.1, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for $L = (A, B, M, Q, R, U, E, F, S, H, C, P, N)$;*
2. *source-chain exposure for A and B before any relation read;*
3. *layer-map and classifier exposure through M and Q;*
4. *preserved and not-preserved disclosure through R and U;*
5. *exactness, failure-boundary, and strength disclosure through E, F, S;*
6. *consumer non-escape for object identity, hidden empirical law, proof transfer, subject-substance transfer, bridge promotion, global-frame import, and unlisted strength.*

Thus the NameCert surface is a relation packet rather than an identity theorem.

Proof. Carrier admission is the projection of Definition 1090.1. The tuple gives the whole finite row list, and the local name row N names that packet as a $\mathsf{LayeredRelationCert}^{\uparrow}$ surface.

The source and classifier obligations are ordered by C. A consumer first reads A and B, then the layer map M, then the relation-classifier row Q.

The disclosure obligations are the rows R, U, E, F, S. They identify what is preserved, what is refused, what exactness means, where the relation weakens, and what certification strength is available.

Finally apply Definition 1090.3, Definition 1090.4, and Definition 1090.5. Together they show that the listed rows exhaust the public surface and that no identity, bridge, proof-transfer, empirical-law, subject-substance, global-frame, or unlisted-strength read can escape from the local NameCert packet. □

The Lean carrier in `Derived/LayeredRelationCertUp/TasteGate.lean` provides Nontrivial LayeredRelationCer and FieldFaithful $\mathsf{LayeredRelationCert}^{\uparrow}$ instances for the displayed carrier $A, B, M, Q, R, U, E, F, S, H, C, P, N$.

Corollary 1090.7 (LayeredRelationCert strict local obstruction)**.** *Let*

$$L = (A, B, M, Q, R, U, E, F, S, H, C, P, N)$$

be an accepted $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier. Suppose a displayed failure, refusal, separation, or countermodel row is used by a consumer against a named converse or reuse route for* L*. Then that row is a strict local obstruction: it blocks only the named route whose read asks to add, replace, or bypass one of the displayed relation coordinates*

$$A, B, M, Q, R, U, E, F, S, H, C, P, N.$$

Equivalently, every consumer-facing use must factor through the two source-chain rows A, B*, the layer-map row* M*, the classifier row* Q*, the preserved row* R*, the not-preserved row* U*, the exactness row* E*, the failure-boundary row* F*, the strength row* S*, the componentwise* hsame *transport row* H*, the displayed* Cont *route row* C*, the provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *row* N*. The obstruction cannot be reinterpreted as a global impossibility, a host theorem, hidden external validation, object identity between the domains, unmediated proof transfer, subject-substance transfer, bridge certification, or an unlisted relation strength.*

Proof. Read the accepted carrier through Definition 1090.1. The finite row inventory is exactly $A, B, M, Q, R, U, E, F, S, H, C, P, N$. The same definition states that the carrier has no coordinate for object identity between the two domains, hidden empirical law, unmediated proof transfer, subject-substance transfer, bridge certificate, global frame, or unlisted relation strength.

Use the classifier of Definition 1090.2. A public relation read is admitted only when it factors through the chain rows A, B, the layer-map row M, the relation-classifier row Q, the preserved row R, the not-preserved row U, the exactness row E, the failure-boundary row F, and the strength row S, with transport, continuation, provenance, and naming supplied by H, C, P, N. Thus a consumer-facing obstruction can meet a named converse or reuse route only at one of those displayed coordinates.

Now apply Definition 1090.4. Refused content is confined to U and F; it cannot escape as preserved structure through R, cannot promote same-schema exactness through E into object identity, and cannot promote paper-certificate strength through S into checked or bridge strength. The route row C only displays source-chain projection, layer-map inspection, classifier admission, refusal inspection, exactness export, and public consumer handoff, while H, P, N only transport, cite, and name the same finite surface. None of these rows adds a second route around the displayed obstruction.

Finally apply Definition 1090.5. Public exactness is readable only after the not-preserved row U has been displayed before E is read. Hence a converse or reuse route that tries to consume exactness without the refusal surface fails at the local U-then-E boundary; it does not become a theorem about all relations, a host-side impossibility result, or a hidden validation channel. If the attempted route asks for any unlisted coordinate, the carrier and classifier provide no readback for it. Therefore the displayed negative row has exactly the local blocking force of the named route and no wider export surface. □

Chapter 1091

A Concrete Naming Certificate for $\mathsf{LayeredRelationDisplay}^{\uparrow}$

The $\mathsf{LayeredRelationDisplay}^{\uparrow}$ packet realizes the relation-before-identity discipline of chapter 9416 as finite NameCert data. It sits after the relation-audit frame of chapter 1037, the licensed observer comparison of chapter 1031, the classifier transport of chapter 590, and the refusal boundary of chapter 1070. Its public role is to display what a layered relation preserves, what it refuses to preserve, which exactness row is available, and where a failed relation request is routed.

Definition 1091.1 (LayeredRelationDisplay carrier)**.** *A* $\mathsf{LayeredRelationDisplay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, S, K, M, R, U, E, F, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source-domain rows whose comparison is being displayed;*
2. *S is the finite source ledger naming the paper sites, histories, or certificate rows from which the comparison may be read;*
3. *K is the classifier row that decides which relation layer is being displayed;*
4. *M is the layer-map row from source coordinates to target coordinates;*
5. *R is the preserved-structure row;*
6. *U is the not-preserved row, recording the object content, identity claim, global frame, host semantics, or over-strong reading refused by the packet;*
7. *E is the exactness row, naming the available sameness strength for the displayed relation;*
8. *F is the failure-boundary row for requests that omit a required source, classifier, preserved, not-preserved, or exactness row;*
9. *H records componentwise* hsame *transports for $A, B, S, K, M, R, U, E, F$ together with package and naming rows;*
10. *C records the displayed* Cont *routes by which readers inspect the source ledger, classifier, layer map, preservation row, refusal row, exactness row, and failure boundary;*
11. *P is the* Pkg *provenance over chapter 9416, chapter 1037, chapter 1031, chapter 590, chapter 1070,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*;*

12. N is the local $\mathsf{NameCert}_{\mathsf{LayeredRelationDisplay}^{\uparrow}}$ *row.*
The carrier has no coordinate for object identity between A and B, host equivalence, unlisted analogy, hidden classifier output, or relation strength beyond the exactness row E.

Lean4 (definition) `BEDC.Derived.LayeredRelationDisplayUp.LayeredRelationDisplayUp`
Lean4 ✓ `BEDC.Derived.LayeredRelationDisplayUp.LayeredRelationDisplayTasteGate single carrier alignment`

Theorem 1091.2 (LayeredRelationDisplay NameCert surface)**.** *For every accepted* $\mathsf{LayeredRelationDisplay}^{\uparrow}$ *carrier $L = (A, B, S, K, M, R, U, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationDisplay}^{\uparrow}}$ *surface consists exactly of carrier exposure, source-ledger exposure, classifier exposure, layer-map exposure, preserved-row exposure, not-preserved-row exposure, exactness exposure, failure-boundary routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the displayed carrier of Definition 1091.1. The tuple lists every row available to a relation-certificate reader.

The source, classifier, layer-map, preservation, refusal, exactness, and failure rows are S, K, M, R, U, E, F. They are the finite relation-certificate fields required by chapter 9416, and they are read through the relation-audit sibling of chapter 1037.

The structural rows H, C, P, N transport, replay, package, and name the same displayed packet. Since no additional identity, host-equivalence, or hidden-strength coordinate appears in the carrier, those rows exhaust the local NameCert surface. □

Theorem 1091.3 (LayeredRelationDisplay preservation-refusal exactness)**.** *Every accepted* LayeredRelationDisplay *packet exports a relation claim only when the preserved row R, the not-preserved row U, and the exactness row E are all displayed in the same carrier. A request with an empty not-preserved row, a missing exactness row, or a classifier read outside K, M is routed to the failure boundary F and cannot be exported as identity, host equivalence, global-frame equivalence, or unchecked analogy.*

Proof. Begin with the carrier inventory. The relation-facing rows are the classifier K, layer map M, preserved row R, not-preserved row U, exactness row E, and failure boundary F.

Use Definition 1091.2. The local NameCert surface requires each of these rows before a public relation read can be exported, and it supplies only componentwise transport and displayed replay for them.

For a complete request, R, U, E are read together through C and transported through H. This gives a relation certificate with a stated preservation claim, a stated refusal boundary, and a stated exactness strength.

For an incomplete request, the missing not-preserved, exactness, or classifier-layer coordinate has no replacement row. The only displayed endpoint for that request is F, so the packet refuses identity, host equivalence, global-frame equivalence, and unchecked analogy. □

Falsifiable prediction: Any accepted LayeredRelationDisplayUp packet with an empty notPreserved row fails the exported relation-certificate gate.

Independence witness: relationalframeaudit, observerlicensedcomparison, classifiermorphism, boundarygate: none is bijective because LayeredRelationDisplayUp carries preserved/notPreserved/exactness/failu rows as one displayed relation packet.

The later Lean TasteGate file must provide Nontrivial $\mathsf{LayeredRelationDisplay}^{\uparrow}$ and FieldFaithful LayeredRelationDi instances for the displayed rows $A, B, S, K, M, R, U, E, F, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationDisplay}^{\uparrow}}$.

Constructive story: A LayeredRelationDisplay$^{\uparrow}$ packet is generated from two source-domain rows, a source ledger, classifier, layer map, preserved row, not-preserved row, exactness row, failure boundary, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LayeredRelationDisplay}^{\uparrow}}$, seed)

Scope closed: Definition 1091.1, Definition 1091.2, and Definition 1091.3 bind layered relation display as a finite NameCert packet with explicit preservation, refusal, exactness, and failure-boundary rows.

Formal verification: FormalStatus(NameCert$_{\text{LayeredRelationDisplay}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, object identity between source domains, global-frame equivalence, host semantic equivalence, or unchecked analogy.

Upgrade path: Obligation closure requires checked carrier admission and TasteGate-visible preservation, refusal, exactness, and failure-boundary rows for the same displayed relation packet.

Chapter 1092

A Concrete Naming Certificate for $\mathsf{AnalogyCertificateGate}^{\uparrow}$

The $\mathsf{AnalogyCertificateGate}^{\uparrow}$ packet concretizes the relation-certificate discipline of chapter 9416 as finite NameCert data. It consumes the same-schema audit rows of chapter 1037, the contextual reading discipline of chapter 684, the description boundary of chapter 661, the admission gate of chapter 545, and the refusal router of chapter 666. Its role is to admit analogy language only after the source rows, carrier, classifier, preservation, refusal, ledger, exactness, and failure-boundary rows are visible in one packet.

Definition 1092.1 (AnalogyCertificateGate carrier)**.** *An* $\mathsf{AnalogyCertificateGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, K, G, R, V, U, L, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-row ledger for the two themes whose analogy is being tested;*
2. *K is the finite carrier row naming the analogy packet rather than either source theme;*
3. *G is the contextual classifier row that licenses the analogy as a relation-first reading;*
4. *R records the relation rows shared by the sources;*
5. *V records the preserved rows that support same-schema language;*
6. *U records the not-preserved rows, including any identity, empirical, subject-substance, or bridge claim refused by the packet;*
7. *L is the ledger row recording which source row supports each preserved or not-preserved entry;*
8. *E is the exactness row saying that same-schema language reads only the displayed relation schema;*
9. *F is the failure-boundary row for analogy requests whose source, classifier, preservation, refusal, ledger, or exactness rows are absent;*
10. *H records componentwise* hsame *transport for all visible rows;*
11. *C records displayed* Cont *routes for admitted same-schema reads and refused analogy reads;*

12. *P is the* Pkg *provenance over* AnalogyCertificateGate$^{\uparrow}$, RelationalFrameAudit$^{\uparrow}$, ContextualClassReading$^{\uparrow}$, DefiniteDescriptionBoundary$^{\uparrow}$, TasteGateAdmission$^{\uparrow}$, SupplyKindRouter$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*

13. *N is the local* NameCert$_{\text{AnalogyCertificateGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for object identity between the sources, global-frame comparison, empirical law, subject-substance claim, denoting description, or requested supply outside the displayed packet.

Lean4 ✓ `BEDC.Derived.AnalogyCertificateGateUp.AnalogyCertificateGateNameCertObligations`

Theorem 1092.2 (AnalogyCertificateGate NameCert obligations)**.** *For the carrier of Definition 1092.1, the local* NameCert$_{\text{AnalogyCertificateGate}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (S, K, G, R, V, U, L, E, F, H, C, P, N),$$

source exposure through S, packet-carrier exposure through K, contextual classifier exposure through G, relation-row exposure through R, preserved-row exposure through V, not-preserved-row exposure through U, ledger exactness through L, E, refusal through F, and transport, continuation, provenance, and local naming through H, C, P, N.

Lean4 ✓ `BEDC.Derived.AnalogyCertificateGateUp.AnalogyCertificateGate namecert obligations`

Proof. Carrier admission is projection from Definition 1092.1. The tuple displays the source ledger, packet carrier, contextual classifier, relation rows, preserved rows, not-preserved rows, ledger, exactness row, failure boundary, transport, continuation, provenance, and local name.

The admitted route first reads S, K, G, R. These rows make the analogy a relation-first classifier over displayed source rows, in the sense of chapter 684, rather than a host object shared by the sources.

Next the route reads V, U, L, E. The preserved rows and not-preserved rows are both ledgered before exact same-schema language can be exported, matching the relation-audit discipline of chapter 1037.

The refused route reads F, with chapter 661, chapter 545, and chapter 666 supplying sibling boundaries for denotation, admission, and missing-supply requests. Finally H, C, P, N only transport, replay, package, and name the same rows, so the NameCert surface is exhausted by the displayed packet. □

Theorem 1092.3 (AnalogyCertificateGate obligation row scope)**.** *Every accepted same-schema export from an* AnalogyCertificateGate$^{\uparrow}$ *carrier must expose the preserved row V, the not-preserved row U, the ledger row L, the exactness row E, and the failure row F before the export is readable. The TasteGate instances* FieldFaithful AnalogyCertificateGate$^{\uparrow}$ *and* Nontrivial AnalogyCertificateGate$^{\uparrow}$ *witness that these rows are distinct visible fields of the packet, so a same-schema read cannot erase U, merge L with E, or hide F behind the carrier, classifier, transport, replay, provenance, or local naming rows.*

Lean4 ✓ `BEDC.Derived.AnalogyCertificateGateUp.AnalogyCertificateGateObligationRowScope`

Proof. Start with the carrier of Definition 1092.1 and the NameCert obligations of Definition 1092.2. They place same-schema export on the route through $S, K, G, R, V, U, L, E, F, H, C, P, N$, with V, U, L, E, F displayed before transport or replay can name the export.

The falsifiable prediction of Definition 1092.4 supplies the negative side: deleting U routes the request to F rather than to a same-schema export. The TasteGate paragraph records the Lean-side FieldFaithful AnalogyCertificateGate$^{\uparrow}$ and Nontrivial AnalogyCertificateGate$^{\uparrow}$ instances, so the row scope is not a collapsed carrier read.

Thus accepted analogy language must keep the preserved, not-preserved, ledger, exactness, and failure rows visible before export. The packet exposes no same-schema route that bypasses those rows or treats the analogy as source identity. □

Theorem 1092.4 (AnalogyCertificateGate falsifiable prediction)**.** *If an accepted* AnalogyCertificateGate$^{\uparrow}$ *packet has no not-preserved row U, then it cannot export same-schema language. Equivalently, every admitted same-schema read of an analogy packet factors through both the preserved row V and the not-preserved row U before the exactness row E is readable.*

Lean4 ✓ `BEDC.Derived.AnalogyCertificateGateUp.AnalogyCertificateGateFalsifiablePrediction`

Proof. Start with the NameCert obligations of Definition 1092.2. The theorem places same-schema export on the route through V, U, L, E.

If U is absent, the exactness row has no displayed refusal side. The request is therefore routed to the failure-boundary row F, because the packet cannot distinguish same-schema language from identity collapse or unlisted bridge language.

Thus an accepted analogy gate without U has no admitted export route for same-schema language. The prediction is falsifiable by deleting the not-preserved row: the packet then fails the gate rather than exporting analogy language. □

Theorem 1092.5 (AnalogyCertificateGate independence witness)**.** *The carrier of Definition 1092.1 is not bijective, row for row, to any of* RelationalFrameAudit$^{\uparrow}$, ContextualClassReading$^{\uparrow}$, DefiniteDescriptionBoundary$^{\uparrow}$ TasteGateAdmission$^{\uparrow}$, *or* SupplyKindRouter$^{\uparrow}$. *A bijection preserving local NameCert obligations would have to preserve the joint rows $S, K, G, R, V, U, L, E, F, H, C, P, N$, and no listed sibling carries exactly this source, analogy-carrier, contextual-classifier, preserved, not-preserved, ledger, exactness, failure-boundary, transport, continuation, provenance, and local-name surface.*

Proof. Compare first with RelationalFrameAudit$^{\uparrow}$. That sibling carries frame-audit and no-global-frame rows; it supplies the relation-audit discipline consumed here, but it does not carry the contextual analogy classifier G together with the admission, description, and supply-refusal boundaries.

Compare next with ContextualClassReading$^{\uparrow}$ and DefiniteDescriptionBoundary$^{\uparrow}$. They control contextual class reading and denoting-description use, but neither stores the paired preserved and not-preserved analogy rows V, U with the source ledger S and exactness row E.

Finally compare with TasteGateAdmission$^{\uparrow}$ and SupplyKindRouter$^{\uparrow}$. They carry admission and requested-supply refusal rows, while AnalogyCertificateGate$^{\uparrow}$ carries a relation packet whose same-schema export is governed by V, U, L, E, F. Since each sibling omits at least one required coordinate and carries rows not present in the analogy gate, no carrier bijection can preserve all NameCert obligations. □

Lean4 ✓ `BEDC.Derived.AnalogyCertificateGateUp.AnalogyCertificateGateIndependenceWitness`

Falsifiable prediction: An accepted analogy gate with no not-preserved row cannot export same-schema language.

Independence witness: Nearest siblings: RelationalFrameAudit, ContextualClassReading, DefiniteDescriptionBoundary, TasteGateAdmission, and SupplyKindRouter. The analogy gate is not bijective to any one sibling because it binds source, analogy-carrier, contextual classifier, preserved,

not-preserved, ledger, exactness, failure-boundary, transport, continuation, provenance, and local naming rows in one packet.

The Lean TasteGate file provides Nontrivial AnalogyCertificateGate$^{\uparrow}$ and FieldFaithful AnalogyCertificateGate$^{\uparrow}$ instances for the displayed rows $S, K, G, R, V, U, L, E, F, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{AnalogyCertificateGate}^{\uparrow}}$.

Constructive story: An AnalogyCertificateGate$^{\uparrow}$ packet is generated from source rows, an analogy carrier, contextual classifier, relation rows, preserved rows, not-preserved rows, source ledger, exactness row, failure boundary, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AnalogyCertificateGate}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1092.1, Definition 1092.2, Definition 1092.3, Definition 1092.4, and Definition 1092.5 display the finite NameCert surface and row-level same-schema export obligation for analogy export and refusal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AnalogyCertificateGate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct an empirical bridge, source identity, global-frame object, denoting description, requested-supply witness, complete row-scope Lean theorem, or standard bridge for AnalogyCertificateGate$^{\uparrow}$.

Upgrade path: Public closure requires checked row-scope preservation, same-schema refusal, independence, and bridge-ready export routing for BEDC.Derived.AnalogyCertificateGateUp.

Chapter 1093

A Same-Schema Relation Certificate for $\mathsf{LayeredRelationCert}^{\uparrow}$

The $\mathsf{LayeredRelationCert}^{\uparrow}$ packet realizes the relation-certificate discipline of chapter 9416 as a finite NameCert object. It sits beside the normalization frontier of chapter 892, the inter-Hist invariant surface of chapter 656, the audit-map interface of chapter 806, and the apophatic naming surface of chapter 543. Those siblings expose frontier rows, local relation invariants, audit-map inventory, and named refusal boundaries; this packet binds same-schema comparison without turning the compared domains into one object.

Definition 1093.1 (LayeredRelationCert carrier)**.** *A* $\mathsf{LayeredRelationCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, A, B, L, M, P, Q, V, W, E, F, H, C, K, N)$$

with the following displayed rows:

1. *S is the same-schema request row naming the proposed relation claim;*
2. *A and B are the two source-domain rows whose refinement chains are being compared;*
3. *L is the finite layer-map row matching visible layers of A to visible layers of B;*
4. *M records the classifier rows used at the matched layers;*
5. *P is the preserved-structure row;*
6. *Q is the not-preserved row, recording the content that the relation refuses to identify;*
7. *V records the provenance ledger for each matched layer and classifier read;*
8. *W records the exactness row explaining what the same-schema claim means;*
9. *E is the failure-boundary row for weakened or rejected relation reads;*
10. *F is the strength row, distinguishing paper certificate, checked certificate, and bridge certificate readings;*
11. *H stores componentwise* hsame *transports for $S, A, B, L, M, P, Q, V, W, E, F$ and the structural rows;*
12. *C stores displayed* Cont *routes from the relation request through preservation, refusal, provenance, exactness, and failure readback;*

13. *K is the* Pkg *provenance over* $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{AuditMapInterface}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *row.*

The carrier certifies a relation between displayed refinement chains. It has no coordinate for object identity between A and B, no bridge from physical content to MetaCIC content, and no hidden row that erases Q or E.

Theorem 1093.2 (LayeredRelationCert NameCert surface)**.** *For the carrier of Definition 1093.1, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$R = (S, A, B, L, M, P, Q, V, W, E, F, H, C, K, N),$$

same-schema request exposure through S, source-domain exposure through A, B, layer-map and classifier readback through L, M, preservation readback through P, explicit non-preservation readback through Q, ledger and exactness readback through V, W, failure and strength readback through E, F, and transport, continuation, provenance, and local naming through H, C, K, N.

Proof. Carrier admission is projection from Definition 1093.1. The displayed tuple contains every public row used by the relation request, including the same-schema request, the two source-domain rows, the layer map, classifier rows, preservation and non-preservation rows, ledger, exactness, failure, strength, transport, route, provenance, and local name.

The positive read is then forced through the relation rows. A consumer first reads S, A, B, then reads the finite layer map L and classifier row M, and only then may inspect the preserved-structure row P.

The refusal read is equally public. The rows Q, E, F remain attached to the same packet, so exactness and strength never erase the content that is not preserved and never turn a failure boundary into an identity claim.

Finally H, C, K, N only transport, replay, package, and name the displayed rows. Thus the NameCert surface is exhausted by the carrier and cannot export a hidden comparison channel. □

Lean4 ✓ `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertCarrier namecert surface`

Theorem 1093.3 (LayeredRelationCert preservation boundary)**.** *Every accepted* $\mathsf{LayeredRelationCert}^{\uparrow}$ *packet exposes preserved structure only as the row P read through the same-schema request S, the two source rows A, B, the layer-map row L, the classifier row M, the provenance ledger V, the exactness row W, and the local transport and continuation rows H, C. A downstream consumer may cite the packet as a same-schema certificate, but it may not replace either source domain by the other or infer that their non-preserved content agrees.*

Proof. Start with the carrier and the NameCert surface. The preserved-structure row P is a coordinate of R, and the surface requires a read of $S, A, B, L, M, V, W, H, C$ before P is available to a consumer.

The row L gives a finite layer correspondence, while M gives the classifiers used at those layers. The ledger V and exactness row W record why the correspondence is being read as same-schema rather than as source equality.

The not-preserved row Q remains a separate coordinate in the same carrier. Since no continuation route deletes Q or rewrites it as preserved content, preservation is bounded by P and cannot become object identity between A and B. □

Lean4 ✓ `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertCarrier preservation boundary`

Theorem 1093.4 (LayeredRelationCert failure boundary)**.** *Every accepted* LayeredRelationCert$^{\uparrow}$ *packet carries a failure readback row E tied to the explicit not-preserved row Q. If a requested comparison omits Q, asks the layer map L to identify source objects, or asks exactness W to preserve content outside P, the packet routes the request to E rather than to a same-schema certificate.*

Proof. Project the rows Q, E from the carrier. The not-preserved row and the failure-boundary row are public coordinates, so a relation read cannot hide them behind the preserved row.

Apply the NameCert surface. It exposes exactness only together with P, Q, V, W, E, F, which means a same-schema read has a visible place for the content it refuses to preserve and a visible boundary for failed requests.

If Q is omitted, the carrier no longer has the row that separates relation from identity. If L or W is asked to preserve content outside P, the request has no accepted positive coordinate. In both cases the only displayed route is the failure-boundary row E. □

Lean4 ✓ `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertCarrier failure boundary`

Theorem 1093.5 (LayeredRelationCert TasteGate contract)**.** *The Lean follow-up for the carrier of Definition 1093.1 requires* Nontrivial LayeredRelationCert$^{\uparrow}$ *and* FieldFaithful LayeredRelationCert$^{\uparrow}$ *instances before the TasteGate target* `BEDC.Derived.LayeredRelationCertUp.TasteGate` *may be read as the checked single-carrier handoff. The paper-side packet fixes the rows $S, A, B, L, M, P, Q, V, W, E, F, H, C$, and the no-identity boundary that those instances must preserve.*

Proof. The paper-side contract is read from the carrier and the three boundary theorems. They display the rows that any later carrier instance must expose and the non-preservation row that prevents the relation from collapsing into identity.

The Nontrivial instance is required so that the Lean object is not an empty naming shell. The FieldFaithful instance is required so that the Lean fields preserve the source, layer, classifier, preservation, non-preservation, ledger, exactness, failure, strength, transport, continuation, provenance, and naming rows without dropping or merging them.

Thus the later TasteGate file may certify alignment only for this finite field surface. It cannot certify object identity, physical-to-formal derivation, or any relation read that bypasses Q and E. □

Lean4 ✓ `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertTasteGate single carrier alignment`

Theorem 1093.6 (LayeredRelationCert scoped row discipline)**.** *Every accepted* LayeredRelationCert$^{\uparrow}$ *same-schema read factors through the concrete rows*

$$S, A, B, L, M, P, Q, V, W, E, F, H, C, K, N$$

of Definition 1093.1 and through the sibling anchors MetaCICNormalizationFrontier$^{\uparrow}$, InterHistInvariant$^{\uparrow}$, AuditMapInterface$^{\uparrow}$, *and* ApophaticName$^{\uparrow}$ *named in the carrier provenance. Such a read may expose a same-schema relation certificate, but it cannot turn the source domains A, B into object identity, erase the not-preserved row Q, or erase the failure-boundary row E.*

Proof. Project the accepted carrier. The visible packet is exactly the displayed tuple $R = (S, A, B, L, M, P, Q, V, W,$ so every same-schema read starts from those rows rather than from an ambient relation object.

Apply the NameCert surface of Definition 1093.2. It exposes the request row, source rows, layer map, classifier readback, preservation row, non-preservation row, ledger, exactness, failure, strength, transport, replay, provenance, and local name as one finite surface.

Apply the preservation boundary, failure boundary, and TasteGate contract of Definition 1093.3, Definition 1093.4, and Definition 1093.5. Preservation is bounded by P, failed or overstrong reads remain routed to E, and the checked carrier alignment must preserve the no-identity boundary.

Finally read the sibling anchors already named in the chapter introduction and in the provenance row K: chapter 892, chapter 656, chapter 806, and chapter 543. They supply normalization-frontier, inter-Hist, audit-map, and apophatic rows only as scoped dependencies, not as permissions to identify A with B or to drop Q and E. □

Falsifiable prediction: If the not-preserved row is removed, same-schema relation certificates collapse into object-identity claims and cannot pass the failure-boundary readback.

Independence witness: metacicnormalizationfrontier, interhistinvariant, auditmapinterface, apophaticname, consciousobserverstate: none is bijective because LayeredRelationCertUp carries the same-schema request, paired source rows, layer map, preserved row, not-preserved row, exactness row, failure boundary, and strength row together, while those siblings separately supply frontier evidence, inter-Hist invariants, audit-map inventory, apophatic refusal, or observer-state surfaces without this full relation-certificate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}$.

Constructive story: A $\mathsf{LayeredRelationCert}^{\uparrow}$ packet is generated from a same-schema request row, paired source-domain rows, a finite layer map, classifier rows, preserved and not-preserved rows, provenance and exactness ledgers, failure and strength rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1093.1, Definition 1093.2, Definition 1093.3, Definition 1093.4, Definition 1093.5, and Definition 1093.6 close the same-schema scope on rows $S, A, B, L, M, P, Q, V, W, E, F, H, C, K, N$ and the MetaCICNormalizationFrontierUp, InterHistInvariantUp, AuditMapInterfaceUp, and ApophaticNameUp sibling anchors.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LayeredRelationCert}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.LayeredRelationCertUp.LayeredRelationCertTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close object identity between the source domains, physical-to-formal derivation, bridge checking, or relation reads that bypass the not-preserved and failure rows.

Upgrade path: Scoped closure requires checked public-object stability for the displayed rows $S, A, B, L, M, P, Q, V, W, E, F, H, C, K, N$, including preservation-boundary, failure-boundary, and same-schema ledger nonescape obligations.

Chapter 1094

A Concrete Naming Certificate for $\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}$

The $\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}$ packet isolates the flow-level theorem family in the GroundCompiler audit map of chapter 9407. It sits beside the verdict packet of chapter 822, the event-flow packet of chapter 829, and the event-flow route packet of chapter 830: those chapters expose aggregate verdict, event-flow, and routed audit surfaces, while this chapter fixes the local row where a displayed flow encoding decodes back to the same flow level.

Definition 1094.1 (GroundCompiler flow-level round-trip carrier)**.** *A* $\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, E, D, L, T, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the GroundCompiler event-flow level row being encoded;*
2. *E is the finite* $\mathsf{FlowEncoding}(S)$ *row;*
3. *D is the decoder row witnessing* $\mathsf{Decode}(E) = \mathsf{some}(S)$*;*
4. *L is the legal-stream row certifying that E is a legal channel stream before any decoder read is accepted;*
5. *T is the transport row tying the flow-level readback to the source-channel and event-flow audit surfaces;*
6. *R is the route row exposing the round-trip to the GroundCompiler audit map without identifying it with verdict, frontier, obstruction, or report data;*
7. *H records componentwise* hsame *transports for the flow level, encoding, decoder, legal stream, transport, route, provenance, and local naming rows;*
8. *C records displayed* Cont *routes from S to E, from E through L to D, and from the decoded row back to the audit-map consumer;*
9. *P is the* Pkg *provenance over chapter 9407,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}}$ *row.*

The carrier records a finite flow-level encoding and decoder readback. It has no coordinate for compiler verdicts, event-flow aggregate audits, routed certificate gates, audit-map frontier packets, obstruction sockets, report evidence, motif evidence, metric evidence, or host compiler state.

Lean4 ✓ BEDC.Derived.GroundCompilerFlowLevelRoundTripUp.GroundCompilerFlowLevelRoundTripTasteGate
single carrier alignment

Theorem 1094.2 (GroundCompiler flow-level round-trip NameCert obligations)**.** *For the carrier of Definition 1094.1, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}}$ *surface consists exactly of six obligation rows: flow-level exposure through* S*, finite flow encoding through* E*, legal-stream admission through* L*, decoder readback through* D *with* $\mathsf{Decode}(\mathsf{FlowEncoding}(S)) = \mathsf{some}(S)$*, audit-map transport through* T, R*, and provenance-preserving naming through* H, C, P, N*.*

In particular, an accepted packet exposes the checked GroundCompiler round-trip row only when the legal-stream row and the decoder row are both displayed.

↪variant BEDC.GroundCompiler.ChannelEncoding.flow level round trip

Proof. Project the finite packet $F = (S, E, D, L, T, R, H, C, P, N)$. The rows S, E, D, L are the whole flow-level encoding and decoding surface, while T, R, H, C, P, N transport, route, locate, and name that same surface.

Legal-stream admission comes before decoder readback. The route from S to D first records $E = \mathsf{FlowEncoding}(S)$ and the legal-stream row L, and only then records the decoder equation $\mathsf{Decode}(E) = \mathsf{some}(S)$.

The checked Lean theorem `BEDC.GroundCompiler.ChannelEncoding.flow_level_round_trip` supplies the kernel-side readback row for the displayed flow level. The local NameCert obligations add the paper-side carrier boundary: every use of that readback must pass through the visible encoding, legal-stream, decoder, transport, route, provenance, and naming coordinates. □

Theorem 1094.3 (GroundCompiler flow-level round-trip non-escape)**.** *Every accepted* GroundCompilerFlowLevelRo *packet exports only the displayed rows*

$$S, E, D, L, T, R, H, C, P, N.$$

It cannot export a $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ *verdict row, a* $\mathsf{GroundCompilerEventFlowAudit}^{\uparrow}$ *aggregate audit row, a* $\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}$ *certificate-gate route, an* $\mathsf{AuditMapFrontierPacket}^{\uparrow}$ *frontier row, an* $\mathsf{AuditMapObstructionSocket}^{\uparrow}$ *obstruction row, report evidence, motif identity, metric evidence, source text, carried metadata, or host compiler state.*

Proof. Start from the carrier inventory in Definition 1094.1. The packet displays a single flow level, its finite encoding, the legal-stream row, the decoder readback, and the structural rows used to transport and route that readback.

Apply the obligation surface of Definition 1094.2. Any consumer of the round-trip must read the legal stream and decoder row before it receives $\mathsf{Decode}(\mathsf{FlowEncoding}(S)) = \mathsf{some}(S)$.

The sibling packets named in the chapter opening have different carrier coordinates. The verdict packet contains query, forbidden-row, and verdict data; the event-flow and route packets contain aggregate audit and certificate-gate rows; the frontier and obstruction packets contain audit-map boundary rows. None is bijective to the round-trip carrier, because none has exactly the displayed legal-stream plus decoder row as its determining coordinate pair, and this carrier has no coordinate storing their verdict, gate, frontier, or obstruction rows. □

Independence witness: groundcompilerauditverdict, groundcompilereventflowaudit, groundcompilereventflowauditroute, auditmapfrontierpacket, auditmapobstructionsocket: the GroundCompilerFlowLevelRo carrier is determined by the displayed flow level, finite encoding, legal-stream row, decoder readback, route, and provenance. It is not bijective to the verdict, event-flow aggregate, event-flow route, frontier, or obstruction carriers, since those packets respectively carry verdict comparison, aggregate audit, certificate-gate routing, frontier, and obstruction rows that this carrier neither stores nor reconstructs.

Falsifiable prediction: If a packet exposes Decode(FlowEncoding S)=some S but omits the displayed legal-stream or decoder row, then it fails the local NameCert obligations.

Lean follow-up for this surface must provide Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/Grou`

Theorem 1094.4 (GroundCompiler flow-level round-trip TasteGate row)**.** *The TasteGate row for* BEDC.Derived.GroundCompilerFlowLevelRoundTripUp *reads the displayed* GroundCompilerFlowLevelRoundTrip$^{\uparrow}$ *carrier through the same flow-level, encoding, legal-stream, decoder, route, transport, replay, provenance, and local naming coordinates. It adds no verdict row, event-flow aggregate, certificate-gate route, frontier packet, obstruction socket, report evidence, motif evidence, metric evidence, source text, carried metadata, or host compiler state.*

Lean4 ✓ `BEDC.Derived.GroundCompilerFlowLevelRoundTripUp.taste gate`

Proof. Project the carrier definition. The TasteGate row has access only to $S, E, D, L, T, R, H, C, P, N$, and the local classifier already separates those coordinates from verdict, route, frontier, obstruction, report, motif, metric, and host-state data.

Apply the single-carrier alignment marker. It identifies the checked TasteGate surface with the same finite carrier rather than with an aggregate GroundCompiler packet.

The non-escape theorem then closes the public boundary: every TasteGate read remains a read of the displayed round-trip row and cannot import a sibling packet coordinate. □

Closure status for NameCert$_{\text{GroundCompilerFlowLevelRoundTrip}^{\uparrow}}$**.**

Constructive story: A GroundCompilerFlowLevelRoundTrip$^{\uparrow}$ packet is generated from a GroundCompiler flow-level row, finite flow encoding, decoder row, legal-stream certificate, audit-map transport row, route row, componentwise hsame transport, displayed Cont replay, provenance, and local naming; the TasteGate instance row is recorded at BEDC.Derived.GroundCompilerFlowLevelRoundTripUp.taste_gate. The carrier accepts a round-trip only after the encoded stream is displayed as legal and the decoder readback returns the same flow level. The classifier therefore isolates one local flow-level row of the audit map: it preserves encoding and decoder data without importing verdicts, frontiers, reports, or host compiler state.

Theory closure: TheoryGate(NameCert$_{\text{GroundCompilerFlowLevelRoundTrip}^{\uparrow}}$, obligation)

Scope closed: This chapter binds the GroundCompiler flow-level round-trip carrier, NameCert obligations, non-escape row, falsifiable prediction, independence witness, and cited TasteGate alignment as the visible obligation surface.

Formal verification: FormalStatus(NameCert$_{\text{GroundCompilerFlowLevelRoundTrip}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim audit verdicts, aggregate event-flow audits, certificate-gate routes, frontier packets, obstruction sockets, report evidence, or host compiler state.

Upgrade path: Scoped closure requires kernel-scope binding for each displayed obligation, including legal-stream admission, decoder readback, non-escape, and TasteGate alignment.

Chapter 1095

A Concrete Naming Certificate for $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$

The $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$ packet isolates the finite cover-shrinkage boundary inside chapter 9237. It sits after the compact-net selector of chapter 914, the finite lower-bound fold of chapter 619, and the metric and rational source rows of chapter 57 and chapter 37; it feeds the uniform-modulus consumer of chapter 369 for continuous-map packets of chapter 130.

Definition 1095.1 (Compact cover shrinkage ledger carrier)**.** *A* $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, U, F, M, Q, C, H, P, N)$$

with the following displayed rows:

1. *K is the compact-net selector row, exposing the finite* $\mathsf{ProbeBundle}$ *of centers and the local pointwise-modulus rows selected from the compact source;*
2. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, recording the tolerance history, the selected precision history, and the consumer packet that will read the uniform modulus;*
3. *F is the finite lower-bound fold row, obtained by primitive recursion over the displayed bundle and carrying the readback that the selected precision is below every shrinkage radius;*
4. *M is the metric row, displaying the source and target distance witnesses, symmetry, and triangle coordinates needed to move from local balls to a uniform estimate;*
5. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positivity and comparison row for the tolerance, shrinkage radii, half-radius comparisons, and finite-fold accumulator;*
6. *C is the cover-shrinkage route, binding the compact cover, pointwise modulus rows, half-radius shrinkage, and finite fold before any uniform-modulus read is accepted;*
7. *H records componentwise* hsame *transports for K, U, F, M, Q, C, P, N;*
8. *P is the* Pkg *provenance over compact-net selector, uniform-modulus consumer, finite-fold, metric,* $\mathsf{Rat}^{\uparrow}$*, cover-shrinkage, transport, continuation, and naming rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCoverShrinkageLedger}^{\uparrow}}$ *row.*

The packet has no coordinate for a direct pointwise-to-uniform selector, ambient compactness oracle, host-side minimum, quotient metric equality, or choice-based radius selection.

Lean4 (definition) `BEDC.Derived.CompactCoverShrinkageLedgerUp.CompactCoverShrinkageLedgerUp`

Theorem 1095.2 (Compact cover shrinkage ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$ *packet*

$$L = (K, U, F, M, Q, C, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactCoverShrinkageLedger}^{\uparrow}}$ *surface consists exactly of compact-net selector exposure through* K*, uniform-modulus consumer exposure through* U*, finite lower-bound fold readback through* F*, metric and* $\mathsf{Rat}^{\uparrow}$ *comparison exposure through* M *and* Q*, cover-shrinkage routing through* C*, componentwise transport through* H*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.CompactCoverShrinkageLedgerUp.CompactCoverShrinkageLedgerTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1095.1. Its coordinates are exactly $K, U, F, M, Q, C, H, P, N$, with K supplying the compact finite-net selector, F supplying the fold readback, and U naming the consumer precision row.

The metric and rational rows M and Q are the only comparison surface. They provide source and target distance witnesses, half-radius comparisons, positivity, and fold comparisons before the shrinkage route can be read.

The route row C orders the handoff. It records that compact cover selection and pointwise-modulus shrinkage precede the finite lower-bound fold, and that the uniform-modulus consumer reads only after this fold has produced the displayed precision history.

Finally H, P, N transport, package, and name the same finite surface. Since the carrier has no coordinate for a direct selector, ambient compactness oracle, host minimum, quotient metric equality, or choice-based radius selection, no such object can be part of the local NameCert obligations. □

Theorem 1095.3 (Compact cover shrinkage ledger uniform-modulus handoff)**.** *Every accepted* $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$ *consumer that reads a* $\mathsf{UniformModulus}^{\uparrow}$ *precision row factors through the displayed compact-net selector* K*, cover-shrinkage route* C*, finite lower-bound fold* F*, metric row* M*, and* $\mathsf{Rat}^{\uparrow}$ *positivity row* Q*. In particular, a consumer cannot pass directly from pointwise modulus rows to* $\mathsf{UniformModulus}^{\uparrow}$ *without the finite shrinkage ledger.*

Lean4 ✓ `BEDC.Derived.CompactCoverShrinkageLedgerUp.CompactCoverShrinkageLedger uniform modulus handoff`

Proof. Begin with the compact-net selector row K. It displays the finite ProbeBundle of centers and the pointwise-modulus rows used by the compact-uniform-continuity construction of chapter 9237.

Use the cover-shrinkage route C to attach the half-radius rows to those centers. This route is the only coordinate that binds the compact cover, local pointwise modulus, and shrinkage comparison before a consumer can inspect a uniform precision.

Run primitive recursion over the displayed ProbeBundle in F. At the empty boundary the packet carries the nonempty selector witness from K; at each successor step the $\mathsf{Rat}^{\uparrow}$ row Q compares the accumulator with the next shrinkage radius and keeps a certified lower-bound history.

The metric row M then supplies the source triangle, symmetry, and target triangle reads needed to turn the lower-bound history into the consumer estimate recorded in U. The structural rows H, P, N only transport, package, and name this route, so they cannot introduce a direct pointwise-to-uniform handoff. □

Falsifiable prediction: If the shrinkage ledger is atomic, every accepted uniform-modulus handoff factors through the displayed compact-net selector and finite lower-bound fold rows; a direct handoff from pointwise moduli to UniformModulusUp without those rows refutes the chapter.

Independence witness: Nearest siblings: compactnetmodulusselector, uniformmodulus, finitenetminimumfold, compactmoduluscover, subordinatemoduluscover. It is not bijective to any one sibling because it simultaneously records the cover-shrinkage route, the fold-readback row, and the consumer precision row; each sibling lacks at least one of those rows.

Lean follow-up for this surface must provide Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/Comp`

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCoverShrinkageLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCoverShrinkageLedger}^{\uparrow}$ packet is generated from a compact-net selector, uniform-modulus consumer row, finite lower-bound fold, metric row, rational positivity and comparison row, cover-shrinkage route, componentwise hsame transport, provenance, and local naming. The carrier reads local pointwise modulus data only after the finite cover is displayed, shrinks the cover through rational comparisons, folds the finite lower bound by primitive recursion, and then hands the result to the uniform-modulus consumer. The classifier records this finite shrinkage ledger instead of a direct pointwise-to-uniform selector.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCoverShrinkageLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the compact cover shrinkage ledger carrier, NameCert obligations, uniform-modulus handoff theorem, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCoverShrinkageLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim ambient compactness oracles, host-side minima, quotient metric equality, direct selectors, or choice-based radius selection.

Upgrade path: To reach obligationClosure, formalize the compact-net selector, finite fold, rational comparisons, and uniform-modulus handoff as checked carrier rows.

Chapter 1096

A Concrete Naming Certificate for KernelInductiveEliminator$^{\uparrow}$

The KernelInductiveEliminator$^{\uparrow}$ packet realizes the eliminator side of the kernel-acceptance witness of chapter 9364 as finite NameCert data. It sits beside the inductive-declaration packet of chapter 915, the accepted-declaration witness of chapter 549, the generator-closure surface of chapter 486, the type-classifier membership surface of chapter 476, and the subject-reduction boundary of chapter 483. Those siblings record declaration acceptance, generator closure, type membership, and preservation-facing rows; this packet records the emitted case-analysis rule rows that must be displayed before any eliminator consumer reads them.

Definition 1096.1 (KernelInductiveEliminator carrier)**.** *A* KernelInductiveEliminator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, L, M, R, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted inductive declaration row already named by* KernelInductiveAcceptance$^{\uparrow}$*;*
2. *S is the constructor-signature row whose finite constructor inventory determines the available case branches;*
3. *L is the emitted eliminator-name row, including the recursor, induction-principle, motive, minor-premise, and computation-rule names emitted for D;*
4. *M is the motive-and-branch arity row, recording the finite arity match between constructor signatures and eliminator minor premises;*
5. *R is the emitted-rule ledger, listing exactly the case-analysis and computation rows that may be read by downstream consumers;*
6. *B is the boundary row that refuses a consumer asking for a host eliminator primitive not displayed in R;*
7. *H records componentwise* hsame *transports for D, S, L, M, R, B, P, N;*
8. *C records the displayed* Cont *routes from accepted declaration to constructor signatures, from signatures to emitted eliminator rows, from eliminator rows to motive-and-branch arity, and from emitted rules to consumers;*
9. *P is the* Pkg *provenance over chapter 9364, chapter 915, chapter 549, chapter 486, chapter 476, chapter 483,* BHist*,* hsame*,* Cont*, and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelInductiveEliminator}^{\uparrow}}$ *row.*
The carrier contains no hidden host recursor, no unlisted computation rule, no ambient induction theorem, and no subject-reduction proof beyond the displayed emitted-rule surface.

Lean4 ✓ `BEDC.Derived.KernelInductiveEliminatorUp.TasteGate.KernelInductiveEliminatorTasteGate single carrier alignment`

Theorem 1096.2 (KernelInductiveEliminator NameCert obligations)**.** *For the carrier of Definition 1096.1, the local* $\mathsf{NameCert}_{\mathsf{KernelInductiveEliminator}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier exposure for the packet $E = (D, S, L, M, R, B, H, C, P, N)$;*
2. *declaration handoff, requiring the row D to be consumed from the accepted inductive declaration surface of* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$*;*
3. *branch arity exactness, requiring the eliminator motive and minor-premise rows in L, M to match the constructor-signature row S;*
4. *emitted-rule ledger exactness, requiring every case-analysis or computation read to be one of the displayed entries of R;*
5. *primitive non-escape, requiring any attempted host eliminator primitive outside R to land in the refusal boundary B rather than in a consumer route.*

Proof. Carrier exposure is projection from the displayed BHist packet. The accepted declaration, constructor signature, eliminator names, motive-and-branch arities, emitted-rule ledger, refusal boundary, transport, route, provenance, and local name are coordinates of one finite object.

The declaration handoff reads D through chapter 915. That chapter records the accepted inductive declaration and constructor-signature ledger, while the present packet starts only after those rows are available.

Branch arity exactness is the local comparison between S and M. Each constructor listed in S has exactly the minor-premise row recorded in M, and the eliminator names in L are public only through that finite comparison.

Finally, emitted-rule exactness and primitive non-escape are read from R and B. A downstream generator, type-classifier, or subject-reduction consumer may consume a displayed emitted rule, but a request for an unrecorded host primitive has no coordinate in E and is routed to the refusal boundary. □

Theorem 1096.3 (KernelInductiveEliminator nontrivial boundary)**.** *The* $\mathsf{KernelInductiveEliminator}^{\uparrow}$ *packet has a nontrivial boundary for TasteGate: an accepted packet displays at least one emitted-rule row through $D, S, L, M, R, H, C, P, N$, while a consumer asking for an eliminator primitive absent from R lands in B and is not accepted as a displayed eliminator read. The Lean file* `lean4/BEDC/Derived/KernelInductiveEliminatorUp/TasteGate.lean` *must provide* Nontrivial KernelInductiveE *for this finite carrier.*

Proof. The accepted side is visible because the carrier exposes an emitted-rule ledger R after the declaration, constructor-signature, eliminator-name, and arity rows have been displayed.

The refused side is visible because B is a separate row. A request for an eliminator primitive not listed in R cannot be reclassified as an emitted-rule read without adding a coordinate absent from the carrier.

Thus the boundary has two distinguishable finite outcomes: displayed emitted-rule consumption and refusal of an unrecorded primitive. □

Theorem 1096.4 (KernelInductiveEliminator field-faithful boundary)**.** *The* KernelInductiveEliminator$^{\uparrow}$ *packet is field-faithful for TasteGate: any accepted read that preserves $D, S, L, M, R, B, H, C, P, N$ componentwise preserves the emitted eliminator surface, and any read that drops L, M, or R is not a faithful read of this carrier. The Lean file* `lean4/BEDC/Derived/KernelInductiveEliminatorUp/TasteGate.le` *must provide* FieldFaithful KernelInductiveEliminator$^{\uparrow}$ *for this finite carrier.*

Proof. Start with two accepted reads related by the displayed hsame rows. Componentwise transport preserves the accepted declaration, constructor signature, eliminator names, arity ledger, emitted-rule ledger, refusal boundary, route, provenance, and local name together.

The fields L, M, and R distinguish this carrier from the surrounding acceptance packets. KernelInductiveAcceptance$^{\uparrow}$ can expose accepted declarations and constructor signatures, but it does not by itself expose the full emitted-rule consumer surface isolated here.

If any of L, M, or R is absent, the read may still be a declaration-acceptance read, a generator-closure read, a type-membership read, or a subject-reduction boundary read. It is not a faithful read of KernelInductiveEliminator$^{\uparrow}$. □

Falsifiable prediction: An accepted KernelInductiveEliminator$^{\uparrow}$ packet cannot export a host eliminator primitive unless the requested primitive appears as a displayed emitted-rule row in R.

Independence witness: KernelInductiveAcceptance, KernelAcceptanceWitness, GeneratorClosure, TypeClassifierMembership, SubjectReduction: declaration acceptance, accepted-declaration witnessing, generator closure, type membership, and preservation rows are not bijective to emitted eliminator discharge because none of the five sibling surfaces simultaneously records eliminator names, motive-and-branch arity, emitted computation rows, and refusal of host primitives as the packet $E = (D, S, L, M, R, B, H, C, P, N)$ does.

Closure status for NameCert$_{\text{KernelInductiveEliminator}^{\uparrow}}$**.**

Constructive story: A KernelInductiveEliminator$^{\uparrow}$ packet is generated from an accepted inductive declaration row, constructor-signature row, emitted eliminator-name row, motive-and-branch arity row, emitted-rule ledger, host-primitive refusal boundary, componentwise hsame transport, displayed Cont routes, provenance, and local naming. The carrier first consumes the accepted declaration, then compares constructor signatures with eliminator minor premises, and finally exposes only the emitted rules listed in the ledger. The classifier admits eliminator use as finite emitted-rule data and routes any absent primitive to the refusal boundary.

Theory closure: TheoryGate(NameCert$_{\text{KernelInductiveEliminator}^{\uparrow}}$, seed)

Scope closed: This chapter binds the kernel inductive eliminator carrier, NameCert obligations, nontrivial boundary, field-faithful boundary, prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{KernelInductiveEliminator}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim hidden host recursors, unlisted computation rules, ambient induction theorems, or subject-reduction proofs beyond the emitted-rule surface.

Upgrade path: To reach obligationClosure, check the declaration handoff, arity exactness, emitted-rule ledger, and primitive-refusal boundary.

Chapter 1097

A Concrete Naming Certificate for $\mathsf{CompilerClassifierRoute}^{\uparrow}$

The $\mathsf{CompilerClassifierRoute}^{\uparrow}$ surface turns the compiler-as-NameCert-morphism vision of chapter 9371 into one concrete route object. It joins classifier membership from chapter 476, classifier morphism from chapter 590, NameCert morphism from chapter 481, compiler graph replay from chapter 473, and trace faithfulness from chapter 1055. A consumer therefore reads one finite route rather than coordinating five adjacent certificate packets.

Falsifiable prediction: A compiler-classifier route is rejected whenever Ext preservation, SigRel preservation, Cont replay, and hsame trace faithfulness do not land on the same displayed graph row.

Independence witness: namecertmorphism, certificatecompiler, classifiermorphism, typeclassifiermembership, compilertracefaithfulness: none is bijective because this packet stores the single route joining all five surfaces.

Definition 1097.1 (CompilerClassifierRoute carrier)**.** *A* $\mathsf{CompilerClassifierRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, A, M, K, G, E, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{NameCert}$ *row whose classifier admits source histories;*
2. *T is the target* $\mathsf{NameCert}$ *row whose classifier receives transported histories;*
3. *A is the* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *row exposing the source classifier-membership surface consumed by the route;*
4. *M is the* $\mathsf{ClassifierMorphism}^{\uparrow}$ *row recording classifier preservation between source and target classifier rows;*
5. *K is the* $\mathsf{NameCertMorphism}^{\uparrow}$ *row recording carrier and naming-certificate preservation;*
6. *G is the* $\mathsf{CertificateCompiler}^{\uparrow}$ *compiler graph row whose displayed edges are read by both the classifier morphism and the NameCert morphism;*
7. *E is the* Ext*-preservation row, stating that accepted source histories land as accepted target histories along the graph row G;*

8. *Q is the* SigRel*-preservation row, stating that the route preserves the displayed source and target signature relation rows;*
9. *L is the* $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ *row transporting source classifier agreement and source* hsame *to target classifier agreement and target* hsame *through the same graph row;*
10. *H records componentwise* hsame *transport for $S, T, A, M, K, G, E, Q, L, C, P, N$;*
11. *C records the displayed* Cont *replay for reading source membership, graph edges, target landing, classifier transport, NameCert transport, and trace faithfulness through one route;*
12. *P names the dependency surface* BHist, Ext, SigRel, hsame, Cont, Pkg, NameCert, $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{ClassifierMorphism}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, $\mathsf{CertificateCompiler}^{\uparrow}$, *and* $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompilerClassifierRoute}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through these displayed rows and never through host compiler semantics, quotient program equality, hidden target objects, or an undisplayed graph.

Theorem 1097.2 (CompilerClassifierRoute NameCert obligations)**.** *For the carrier of Definition 1097.1, the local* $\mathsf{NameCert}_{\mathsf{CompilerClassifierRoute}^{\uparrow}}$ *surface has five obligations:*

1. *carrier exposure: the accepted object is the finite packet $R = (S, T, A, M, K, G, E, Q, L, H, C, P, N)$;*
2. *classifier membership admission: the source row A is the only source of accepted classifier membership, and every route read of that membership is replayed through C;*
3. *shared graph alignment:* Ext *preservation,* SigRel *preservation, classifier morphism, NameCert morphism, compiler graph replay, and trace faithfulness all cite the same displayed graph row G;*
4. *trace transport: source classifier agreement and source* hsame *are transported to target classifier agreement and target* hsame *only through L, H, C;*
5. *non-escape: no host compiler semantics, quotient program equality, hidden target object, second graph, or undeclared optimization row is exported by the route.*

Proof. Carrier exposure is projection from the finite tuple. The rows $S, T, A, M, K, G, E, Q, L, H, C, P, N$ are the complete public inventory available to the local classifier.

Classifier membership admission is inherited from chapter 476. The route does not introduce another membership predicate; it replays the accepted membership row A through the continuation row C.

Shared graph alignment is the defining route condition. The classifier morphism, NameCert morphism, certificate compiler graph, Ext row, SigRel row, and trace-faithfulness row are admissible only when their graph coordinate is the same displayed G.

Trace transport then follows through L, H, C. Since L is the trace-faithfulness row and H, C preserve the same componentwise route, target agreement can be read only after the source membership and shared graph rows have been displayed.

Finally P and N name the provenance and local certificate without adding semantic content. The absence of any other coordinate rules out host compiler semantics, quotient equality, hidden target objects, a second graph, and optimization data. □

Lean4 ✓ `BEDC.Derived.CompilerClassifierRouteUp.CompilerClassifierRouteNameCertObligations`

Theorem 1097.3 (CompilerClassifierRoute shared-row rejection)**.** *An accepted* CompilerClassifierRoute$^{\uparrow}$ *packet must place* Ext *preservation,* SigRel *preservation,* Cont *replay, and* hsame *trace faithfulness on the same displayed graph row G. If any of those rows lands on a different graph, or if one of the rows is absent, the packet is not accepted as a compiler-classifier route.*

Proof. Use the carrier definition to read the single graph coordinate G. Every row that claims source-to-target transport is listed beside that coordinate in the same packet.

Apply the NameCert obligations. Shared graph alignment makes G the common endpoint for Ext, SigRel, classifier morphism, NameCert morphism, compiler replay, and trace faithfulness, while trace transport forces hsame preservation through L, H, C over that same coordinate.

If a row uses another graph, the shared graph obligation has no witness. If a row is absent, one of the five obligation rows cannot be read. In either case the displayed packet may still contain a compiler graph or classifier morphism sibling, but it does not contain the joined compiler-classifier route. □

Lean4 ✓ `BEDC.Derived.CompilerClassifierRouteUp.CompilerClassifierRouteSharedRowRejection`

Theorem 1097.4 (CompilerClassifierRoute downstream target)**.** *The Lean-facing downstream surface for* CompilerClassifierRoute$^{\uparrow}$ *is the packet family to be recorded under* Derived/CompilerClassifierRouteUp/Tast *with explicit* Nontrivial CompilerClassifierRoute$^{\uparrow}$ *and* FieldFaithful CompilerClassifierRoute$^{\uparrow}$ *instances. The gate must recover the shared graph row G, the preservation rows E, Q, the trace-faithfulness row L, and the replay rows H, C, P, N before admitting the route.*

Proof. The carrier and obligation surface name the rows the gate must inspect. A nontrivial instance requires a packet with a displayed source certificate, target certificate, classifier-membership row, classifier morphism, NameCert morphism, compiler graph, preservation rows, trace-faithfulness row, replay rows, provenance, and local naming.

Field faithfulness requires those rows to remain recoverable from the admitted route. The shared-row rejection theorem gives the test: the gate reads G, E, Q, L, H, C, P, N and accepts only when the preservation and trace-faithfulness rows land on the same displayed graph row.

No additional Lean object is part of this downstream target. The paper packet fixes the finite row inventory that later formalization must preserve. □

Lean4 ✓ `BEDC.Derived.CompilerClassifierRouteUp.compilerClassifierRouteChapterTasteGate`

Closure status for NameCert$_{\text{CompilerClassifierRoute}^{\uparrow}}$.

Constructive story: A CompilerClassifierRoute$^{\uparrow}$ packet is generated from source and target NameCert rows, classifier-membership evidence, classifier morphism, NameCert morphism, compiler graph, Ext and SigRel preservation rows, trace-faithfulness ledger, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier accepts the route only when membership, morphism, compiler graph, preservation, and trace rows land on the same displayed graph. Its route ledger keeps source admission and target readback synchronized. The classifier therefore joins five adjacent certificate surfaces into one finite route rather than letting a consumer choose incompatible compiler or classifier witnesses.

Theory closure: TheoryGate(NameCert$_{\text{CompilerClassifierRoute}^{\uparrow}}$, seed)

Scope closed: This chapter binds the compiler classifier route carrier, shared-row rejection theorem, and downstream TasteGate target.

Formal verification: FormalStatus(NameCert$_{\text{CompilerClassifierRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim arbitrary compiler correctness, detached classifier preservation, hidden graph equality, or trace faithfulness outside the displayed route.

Upgrade path: To reach obligationClosure, formalize the shared compiler graph row and prove Ext, SigRel, Cont, and hsame preservation land on it together.

Chapter 1098

A Concrete Naming Certificate for $\mathsf{LayeredRelationFailureBoundary}^{\uparrow}$

The $\mathsf{LayeredRelationFailureBoundary}^{\uparrow}$ packet realizes the failure-boundary row of chapter 9416 as a finite NameCert surface for cross-domain relation certificates. It sits beside the full relation packet of chapter 1090, the audit-frame packet of chapter 1037, the force-like inter-history relation packet of chapter 1088, and the failed gate witness of chapter 901. Those siblings expose a complete relation certificate, an audit frame, a force-like carrier, or a failed audit item; this packet isolates the coordinate that says exactly where same-schema export weakens or is refused before any public consumer may read relation language.

Falsifiable prediction: Any public same-schema export that lacks a displayed failure-boundary row cannot cite this NameCert; a verified export without such a row refutes the packet.

Independence witness: layeredrelationcert, relationalframeaudit, forceinterhistrelation, auditgatefailurewitness, apophaticname: the packet is not bijective with any sibling because it isolates the weakening/failure coordinate, whereas the siblings respectively package the full relation certificate, audit frame, force-like carrier, failed audit witness, or named apophatic boundary.

Definition 1098.1 (Layered relation failure-boundary carrier)**.** *A* $\mathsf{LayeredRelationFailureBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, E, B, W, G, H, C, P, N)$$

with the following displayed rows:

1. *R is the relation-certificate row being tested for public same-schema export;*
2. *E is the exactness row that records what the relation may export before identity, empirical transfer, proof transfer, or bridge strength is read;*
3. *B is the failure-boundary row naming the condition under which the relation weakens, fails, or must be withheld from public export;*
4. *W is the weakening and refusal row attached to B, listing the non-preserved structure, refused object content, refused empirical content, and refused bridge content;*
5. *G is the gate report row recording no identity collapse, hidden axiom, global-frame leak, subject-substance leak, or empirical overclaim for the displayed relation read;*

6. *H records componentwise* hsame *transport for relation-certificate, exactness, failure-boundary, weakening, gate-report, route, provenance, and local-name rows;*
7. *C records displayed* Cont *routes from R to E, from E to B, from B to W, from W to G, and from the accepted gate report to any public same-schema consumer;*
8. *P is the* Pkg *provenance over chapter 9416, chapter 1090, chapter 1037, chapter 1088, chapter 901, chapter 543,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationFailureBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for exporting a same-schema relation without a displayed weakening and failure boundary, and it has no coordinate for turning the boundary into object identity, empirical law, hidden axiom, global frame, subject substance, or checked bridge strength.

Lean4 (definition) `BEDC.Derived.LayeredRelationFailureBoundaryUp.LayeredRelationFailureBoundaryUp`
Lean4 ✓ `BEDC.Derived.LayeredRelationFailureBoundaryUp.LayeredRelationFailureBoundaryTasteGate single carrier alignment`

Theorem 1098.2 (Layered relation failure-boundary non-escape)**.** *For every accepted* LayeredRelationFailureBound *carrier*

$$F = (R, E, B, W, G, H, C, P, N),$$

all public relation export reads pass through $R, E, B, W, G, H, C, P, N$. The consumer may read exactness only together with the displayed failure-boundary row B and the weakening/refusal row W; it receives no route that bypasses B, detaches W, hides the gate report G, or promotes same-schema language to object identity, empirical transfer, proof transfer, global-frame transfer, subject-substance transfer, or bridge checking.

Proof. Project the carrier of Definition 1098.1. The exactness row E, failure-boundary row B, weakening/refusal row W, and gate report row G are displayed as separate coordinates of the same finite packet.

Read the continuation routes in C. A public same-schema consumer reaches E from R, then reaches B, then reaches W, and then reads G before export. Thus exactness cannot be consumed as a detached object-identity or bridge-strength claim.

The structural rows H, P, N transport, cite, and name the same packet. Since none of those rows supplies a new relation source, hidden axiom, empirical law, global frame, subject-substance carrier, or bridge checker, the displayed boundary cannot escape through transport, provenance, or local naming. □

Lean4 ✓ `BEDC.Derived.LayeredRelationFailureBoundaryUp.LayeredRelationFailureBoundary failure boundary row determinacy`

Theorem 1098.3 (Layered relation failure-boundary gate exactness)**.** *An accepted* LayeredRelationFailureBoundary *packet permits public same-schema export exactly when the relation-certificate row R, exactness row E, failure-boundary row B, weakening/refusal row W, and gate report row G are displayed in the same carrier and the consumer route preserves them through H, C, P, N. If any of B, W, G is absent, then the packet supplies no public exactness export.*

Proof. For the forward direction, read the carrier. The public exactness route is the displayed route from R through E, B, W, G and then through the structural rows H, C, P, N. These rows give the relation certificate, the exactness claim, the boundary where it weakens, the refused content attached to that boundary, and the gate report that prevents over-reading.

For the reverse direction, suppose a displayed route presents $R, E, B, W, G, H, C, P, N$ in one carrier. The route has the exactness row, the failure boundary, the weakening/refusal row, and the gate report needed by chapter 9416 before public export, so it determines the same same-schema consumer surface.

If B is absent, the export has no row naming where the relation weakens or fails. If W is absent, the export has no refusal row separating exactness from preserved content. If G is absent, the export has no gate report blocking identity collapse, hidden axiom use, global-frame leakage, subject-substance leakage, or empirical overclaim. In each case the carrier lacks a public exactness export route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationFailureBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{LayeredRelationFailureBoundary}^{\uparrow}$ packet is generated from a relation-certificate row, exactness row, failure-boundary row, weakening/refusal row, gate report, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LayeredRelationFailureBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1098.1, Definition 1098.2, and Definition 1098.3 bind the finite failure-boundary packet for same-schema relation export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LayeredRelationFailureBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the full layered relation certificate, a relational frame audit, a force-like relation, a failed audit witness, object identity, empirical transfer, global-frame transfer, subject-substance transfer, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exactness gating, weakening/refusal attachment, gate-report preservation, transport, continuation, provenance, naming, and TasteGate nontriviality with field faithfulness for the same failure-boundary packet.

Chapter 1099

A Concrete Naming Certificate for $\mathsf{LayeredRelationGate}^{\uparrow}$

The $\mathsf{LayeredRelationGate}^{\uparrow}$ packet realizes the gate surface of Definition 9416.19 as a finite same-schema relation verdict. It sits after the relation certificate of chapter 9416, the concrete certificate packet of chapter 1090, the force inter-history relation packet of chapter 1088, and the axiom-refusal ledger of chapter 1057. Those siblings supply relation-certificate rows, force-relation rows, and refusal-ledger rows; this packet binds their gate verdict, identity refusal, and same-schema consumer route as one NameCert surface.

Falsifiable prediction: A gate packet that lacks a not-preserved row cannot export a same-schema consumer theorem.

Independence witness: layeredrelationcert, forceinterhistrelation, groundcompilerrecognizerflow, axiomrefusalledger: the gate carrier has a verdict and identity-refusal row absent from the certificate-only, force-relation, recognizer-flow, and axiom-refusal packets, so no bijection can preserve all NameCert rows.

Definition 1099.1 (LayeredRelationGate carrier)**.** *A* $\mathsf{LayeredRelationGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, L, P, N, F, G, H, C, \Pi)$$

with the following displayed rows:

1. *A and B are the two source-domain rows whose relation is being gated;*
2. *L is the finite layer-list row pairing source layers, carrier families, classifier families, and layer-map entries;*
3. *P is the preserved row, listing exactly the relation content that may be exported under same-schema language;*
4. *N is the not-preserved row, listing object identity, empirical law, proof transfer, subject-substance, global-frame, and bridge-strength content refused by the gate;*
5. *F is the finite ledger, exactness, and failure-boundary row, recording each gate question of Definition 9416.19;*
6. *G is the gate verdict row, whose accepted value permits same-schema relation export only when A, B, L, P, N, F are all displayed;*

7. *H records componentwise* hsame *transport for source, layer, preserved, not-preserved, refusal, verdict, continuation, and provenance rows;*
8. *C records displayed* Cont *routes for source inspection, layer-list replay, preserved-row export, not-preserved-row inspection, refusal-ledger replay, and consumer handoff;*
9. *Π is the* Pkg *and local* $\mathsf{NameCert}_{\mathsf{LayeredRelationGate}^{\uparrow}}$ *row over chapter 9416, chapter 1090, chapter 1088, chapter 1048, chapter 1057,* BHist, hsame, Cont, Pkg, *and* NameCert.

The carrier has no coordinate for identifying A with B as objects. Same-schema relation language is an export of the verdict row G after the preserved row P and the not-preserved row N have both been displayed.

Theorem 1099.2 (LayeredRelationGate NameCert obligations)**.** *For the carrier of Definition 1099.1, the local* $\mathsf{NameCert}_{\mathsf{LayeredRelationGate}^{\uparrow}}$ *surface consists of seven finite obligations:*

1. *carrier exposure for $K = (A, B, L, P, N, F, G, H, C, \Pi)$;*
2. *source-domain exposure for A and B before any same-schema relation read;*
3. *layer-list exposure for the paired source layers, carrier families, classifier families, and layer map in L;*
4. *preserved-row disclosure through P;*
5. *not-preserved disclosure through N before verdict export;*
6. *refusal-ledger and failure-boundary replay through F;*
7. *consumer handoff through G, H, C, Π without object identity, empirical law, proof transfer, subject-substance, global-frame, bridge-strength, or hidden-choice export.*

Lean4 ✓ `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateNameCertObligations`

Proof. Carrier exposure is projection from the finite tuple in Definition 1099.1. The accepted packet has no fields outside $A, B, L, P, N, F, G, H, C, \Pi$.

Source-domain and layer-list exposure are ordered by the displayed continuation row C. A same-schema consumer first reads the source rows A, B and then the layer-list row L before any preserved or refused content is available.

The preserved and not-preserved obligations are the rows P and N. Since G can be read only after both rows have been displayed, a passing gate cannot hide refused content behind same-schema wording.

The refusal-ledger obligation is the row F. It records the failure boundary for the gate questions of Definition 9416.19, including identity collapse, hidden empirical law, subject-substance, global-frame, object-level T, and strength over-reading.

Finally, the structural rows H, C, Π transport, replay, package, and name the same finite verdict surface. They do not add object identity or choice rows, so the local NameCert surface is exactly the listed gate packet. □

Theorem 1099.3 (LayeredRelationGate same-schema non-identity)**.** *If an accepted* $\mathsf{LayeredRelationGate}^{\uparrow}$ *packet passes its verdict row G, then downstream consumers may use the displayed same-schema relation over A, B, L, P, N, F. They may not read the verdict as object identity between A and B, as empirical law, as proof transfer, as subject-substance transfer, as a global-frame theorem, or as bridge strength.*

Lean4 ✓ BEDC.Derived.LayeredRelationGateUp.LayeredRelationGate same schema nonidentity

Proof. Begin with the NameCert obligations of Definition 1099.2. The verdict is downstream from the source rows, layer-list row, preserved row, not-preserved row, and refusal ledger.

The same-schema relation is therefore exhausted by A, B, L, P, N, F, G. The preserved row P records what may be exported, while the not-preserved row N records what the export refuses.

Object identity and the other excluded readings are listed only in the not-preserved and refusal rows. Since neither H nor C nor Π contains a constructor from those rows to preserved content, a passing verdict licenses relation use without promoting the two domains to one object. □

Lean4 ✓ BEDC.Derived.LayeredRelationGateUp.LayeredRelationGate not preserved row determinacy

Theorem 1099.4 (LayeredRelationGate consumer route)**.** *Every same-schema consumer of a passing* $\mathsf{LayeredRelationGate}^{\uparrow}$ *packet factors through the relation-certificate source of chapter 9416, the concrete certificate surface of chapter 1090, the force inter-history relation surface of chapter 1088, and the axiom-refusal ledger of chapter 1057. The consumer reads exactly the source, layer-list, preserved, not-preserved, ledger, exactness, failure-boundary, verdict, transport, continuation, package, and local NameCert rows of Definition 1099.1.*

Lean4 ✓ BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateConsumerRoute
↪ variant BEDC.Derived.LayeredRelationGateUp.LayeredRelationGate consumer route

Proof. First read the relation-certificate discipline of chapter 9416 and chapter 1090. These supply the source, layer, preserved, not-preserved, exactness, and failure-boundary rows consumed by the gate.

Next use chapter 1088. It supplies a concrete same-schema relation surface without turning the compared histories or domains into one object.

Now use chapter 1057. It supplies the refusal-ledger discipline that keeps hidden axioms, global-frame imports, and unrecorded strength claims outside the exported relation.

Finally apply Definition 1099.2 and Definition 1099.3. They force the consumer through $A, B, L, P, N, F, G, H, C, \Pi$ and make the verdict a same-schema relation gate, not an identity theorem. □

Theorem 1099.5 (LayeredRelationGate carrier obligation)**.** *Every* $\mathsf{LayeredRelationGate}^{\uparrow}$ *obligation read must expose the carrier rows* $A, B, L, P, N, F, G, H, C, \Pi$ *before any verdict, consumer route, or same-schema export is available.*

Lean4 ✓ BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateCarrierObligation

Proof. Use the carrier of Definition 1099.1. Its finite tuple lists exactly the two source rows, layer-list row, preserved row, not-preserved row, refusal ledger, verdict row, transport row, continuation row, and package-name row.

Apply the NameCert obligations of Definition 1099.2. Carrier exposure is the first obligation, so the downstream route cannot read G or a same-schema consumer before $A, B, L, P, N, F, G, H, C, \Pi$ have all been displayed. □

Theorem 1099.6 (LayeredRelationGate classifier obligation)**.** *A* $\mathsf{LayeredRelationGate}^{\uparrow}$ *classifier read may export a passing verdict only after comparing the preserved row P with the not-preserved row N over the same source, layer, and refusal rows A, B, L, F. The verdict row G is therefore not a classifier for identity between A and B.*

Lean4 ✓ `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateClassifier obligation`
↪ variant `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateCarrier classifier obligation`

Proof. Start from Definition 1099.2. The preserved and not-preserved disclosures are separate obligations, and both are upstream of verdict export.

Now apply Definition 1099.3. It states that a passing verdict licenses relation use while refusing object identity, empirical law, proof transfer, subject-substance transfer, global-frame content, and bridge strength. Hence the classifier must compare P and N before G can be read as a same-schema verdict. □

Theorem 1099.7 (LayeredRelationGate ledger obligation)**.** *Every same-schema consumer of* $\mathsf{LayeredRelationGate}^{\uparrow}$ *must keep the refusal ledger F visible. If F is not displayed, the consumer route cannot distinguish a permitted same-schema relation from a hidden identity, empirical-law, proof-transfer, subject-substance, global-frame, or bridge-strength claim.*

Proof. Use the consumer route of Definition 1099.4. The route reads the relation-certificate source, concrete certificate surface, force inter-history relation surface, and axiom-refusal ledger before it reaches the local packet.

Apply Definition 1099.2 and Definition 1099.3. They keep F as the displayed failure-boundary row and place the refused readings in N rather than in the exported relation. Without F, the same-schema consumer has no visible ledger separating the accepted relation from the forbidden readings. □

Lean4 ✓ `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGate ledger obligation`

Theorem 1099.8 (LayeredRelationGate obligation closure)**.** *The carrier of Definition 1099.1, the NameCert obligations of Definition 1099.2, same-schema non-identity from Definition 1099.3, the consumer route of Definition 1099.4, and the carrier, classifier, and ledger obligations of Definition 1099.5, Definition 1099.6, and Definition 1099.7 form the complete finite obligation surface for* $\mathsf{LayeredRelationGate}^{\uparrow}$*. The assembled surface exports only the same-schema relation verdict over the displayed rows $A, B, L, P, N, F, G, H, C, \Pi$ and refuses object identity, empirical law, proof transfer, subject-substance transfer, global-frame theorem, bridge strength, hidden choice, Lean verification, and bridge checking.*

Proof. The carrier theorem fixes the finite packet as $A, B, L, P, N, F, G, H, C, \Pi$. There is no additional row from which identity, empirical law, proof transfer, subject-substance, global-frame structure, bridge strength, hidden choice, verification status, or bridge status could be read.

The NameCert obligation theorem exposes the ordered rows for source domains, layer list, preserved content, not-preserved content, refusal ledger, verdict, transport, continuation, package, and local certificate. Same-schema non-identity then identifies the verdict as relation language rather than object equality.

The consumer route sends every downstream read through the relation-certificate source, concrete certificate surface, force inter-history relation surface, and axiom-refusal ledger. The carrier, classifier, and ledger obligations close the remaining local readings by requiring carrier exposure, preserved/not-preserved comparison, and visible refusal-ledger replay.

Thus every allowed read is already listed by the finite surface, and every refused reading remains in N or F rather than in the exported verdict. The assembled obligation surface is complete for the chapter's current NameCert claim and carries no Lean verification or bridge-checking assertion. □

Lean4 ✓ `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGate obligation closure`

Theorem 1099.9 (LayeredRelationGate obligation scope)**.** *The carrier, classifier, ledger, same-schema nonidentity, and consumer-route obligations of* $\mathsf{LayeredRelationGate}^{\uparrow}$ *are scoped by the single-carrier TasteGate witness for the finite packet* $A, B, L, P, N, F, G, H, C, \Pi$*. Consequently the accepted verdict cannot be read as host identity between the compared domains; it is only a same-schema relation export over the displayed rows.*

Lean4 ✓ `BEDC.Derived.LayeredRelationGateUp.LayeredRelationGateTasteGate single carrier alignment`

Proof. Read the carrier of Definition 1099.1. Its rows form one displayed packet, so every obligation in Definition 1099.5, Definition 1099.6, and Definition 1099.7 is evaluated on the same finite surface.

The TasteGate witness supplies the single-carrier alignment for that surface. It keeps source rows, preserved rows, not-preserved rows, ledger rows, verdict rows, transport, replay, and provenance in one packet rather than allowing an external identification between the compared domains.

Apply same-schema non-identity and the consumer route, Definition 1099.3 and Definition 1099.4. The consumer may export the displayed relation after reading the packet, but no row converts the verdict into host identity, empirical law, proof transfer, subject-substance transfer, global-frame content, bridge strength, or hidden choice. □

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{LayeredRelationGate}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{LayeredRelationGate}^{\uparrow}$ instances in `lean4/BEDC/Derived/LayeredRelationGateUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{LayeredRelationGate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LayeredRelationGate}^{\uparrow}$ packet is generated from source domains, finite layer lists, preserved and not-preserved rows, a refusal ledger, a verdict row, componentwise hsame transport, displayed Cont routes, and local Pkg and NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LayeredRelationGate}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1099.1, Definition 1099.2, Definition 1099.3, Definition 1099.4, Definition 1099.8, and Definition 1099.9 bind the finite LayeredRelationGate carrier, NameCert obligations, same-schema non-identity verdict, consumer route, finite obligation surface, and TasteGate-scoped nonidentity boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LayeredRelationGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close object identity between compared domains, empirical physics, proof transfer, subject-substance content, global-frame structure, Lean verification, audit cleanliness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, verdict determinacy, preserved and not-preserved disclosure, refusal-ledger replay, consumer handoff, and TasteGate field faithfulness for the same finite packet.

Chapter 1100

A Concrete Naming Certificate for MetaCICOpenProblemWitness$^{\uparrow}$

The MetaCICOpenProblemWitness$^{\uparrow}$ packet realizes the consolidated open-problem surface of chapter 9386 as one finite frontier-witness NameCert carrier. It sits beside the subject-reduction obstruction of chapter 9377, the confluence audit map of chapter 9378, the normalization audit map of chapter 9383, the decidability audit map of chapter 9379, and the typed-examples audit map of chapter 9382. Those siblings expose separate audit faces; this packet records the joint ledger that a consistency-facing consumer must read before any stronger route is cited.

The concrete critical-path route of chapter 765 consumes this witness packet as the shared frontier ledger for subject reduction, normalization, decidability, typed-example coverage, and consistency-frontier rows. MetaCICCriticalPath$^{\uparrow}$ may cite the witness only as that joint ledger: it cannot read it as a closed consistency proof or as a replacement for the named audit-face obligations.

Definition 1100.1 (MetaCIC open-problem witness carrier)**.** *A* MetaCICOpenProblemWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, C, N, D, T, K, H, R, P, Q)$$

with the following displayed rows:

1. *S is the subject-reduction frontier row, naming* `metacic_subject_reduction_obstruction` *and the simultaneous beta-substitution, application-argument, lambda-domain, and Pi-domain discharge obligations;*
2. *C is the confluence frontier row, naming* `metacic_confluence_audit` *and the residual-substitution, closed parallel diamond, and arbitrary closed Church–Rosser endpoints;*
3. *N is the normalization frontier row, naming* `metacic_normalization_audit` *and the candidate-mediated strong-normalization boundary;*
4. *D is the decidability frontier row, naming* `metacic_decidable_audit` *and the bounded normal-comparison and closed beta-conversion boundary;*
5. *T is the typed-examples frontier row, naming* `metacic_typedexamples_audit` *and the gallery-versus-coverage boundary;*
6. *K is the consistency-frontier row, recording that any consistency-facing route must pass through one of S, C, N, D, T before it may cite a closed-normal or stronger consistency surface;*

7. *H records componentwise* hsame *transport for S, C, N, D, T, K;*
8. *R records displayed* Cont *replay routes for the five audit faces and the consistency frontier;*
9. *P is the* Pkg *provenance over chapter 9386, the five sibling audit chapters,* BHist, hsame, Cont, *and* NameCert*;*
10. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for closed subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive typed-example coverage, host quotient equality of represented terms, or full closed consistency.

Lean4 (definition) `BEDC.Derived.MetaCICOpenProblemWitnessUp.MetaCICOpenProblemWitnessUp`

Theorem 1100.2 (MetaCIC open-problem witness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemW}}$ *surface for Definition 1100.1 consists exactly of carrier admission for $W = (S, C, N, D, T, K, H, R, P, Q)$, subject-reduction obstruction visibility through S, confluence audit visibility through C, normalization audit visibility through N, decidability audit visibility through D, typed-example audit visibility through T, consistency-frontier visibility through K, componentwise* hsame *transport through H, displayed* Cont *route replay through R, package provenance through P, and local naming through Q.*

Lean4 ✓ `BEDC.Derived.MetaCICOpenProblemWitnessUp.MetaCICOpenProblemWitnessTasteGate single carrier alignment`

Proof. Project the accepted carrier. Its audit-face rows are S, C, N, D, T, and its consistency-facing row is K.

Read the sibling references in the provenance row. The subject-reduction obstruction, confluence audit map, normalization audit map, decidability audit map, and typed-examples audit map supply the five named faces, while chapter 9386 supplies the consolidated frontier reading.

The structural rows H, R, P, Q only transport, replay, package, and name those displayed rows. Since no further coordinate appears in the carrier, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1100.3 (MetaCIC open-problem witness frontier nonescape)**.** *Any accepted consumer of a* $\mathsf{MetaCICOpenProblemWitness}^{\uparrow}$ *packet reaches one of the named frontier rows S, C, N, D, T before it may cite the consistency-facing row K. A route that exports full closed consistency without passing through one of those rows does not factor through the packet.*

Proof. Start with the carrier definition. The consistency-facing row K is displayed only as a frontier row inside $W = (S, C, N, D, T, K, H, R, P, Q)$.

The five audit rows are the only inputs to K. Subject reduction must pass through S, confluence through C, normalization through N, decidability through D, and typed-example coverage through T. Transport through H and replay through R preserve that routing rather than adding a separate consistency coordinate.

Finally read the excluded objects. Since the carrier contains no closed subject-reduction theorem, no arbitrary closed Church–Rosser theorem, no evidence-free normalization theorem, no unrestricted conversion decider, and no exhaustive coverage theorem, a full closed-consistency export without one of the named frontier rows has no coordinate in the packet. □

Falsifiable prediction: Any accepted consumer of $\mathsf{MetaCICOpenProblemWitness}^{\uparrow}$ reaches a named frontier row before it may cite a consistency-facing route; a route that exports full closed consistency without one of those rows refutes the packet.

Independence witness: metacic_subject_reduction_obstruction, metacic_confluence_audit, metacic_normalization_audit, metacic_decidable_audit, metacic_typedexamples_audit: none is bijective with $\mathsf{MetaCICOpenProblemWitness}^{\uparrow}$ because each sibling records one audit face, while this carrier records the joint frontier ledger $S, C, N, D, T, K, H, R, P, Q$ and the consistency-facing non-escape route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICOpenProblemWitness}^{\uparrow}$ packet is generated from subject-reduction obstruction, confluence audit, normalization audit, decidability audit, typed-example audit, consistency-frontier, transport, route, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1100.1, Definition 1100.2, and Definition 1100.3 bind the finite open-problem witness packet and its consistency-facing non-escape rule.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICOpenProblemWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, arbitrary closed Church–Rosser, evidence-free global strong normalization, unrestricted closed beta-conversion decidability, exhaustive typed-example coverage, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exact row coverage for the five audit faces, consistency-frontier routing, route replay, provenance, local naming, and downstream non-escape over the same packet.

Chapter 1101

A Concrete Naming Certificate for $\mathsf{GammaZetaFactorSeal}^{\uparrow}$

The $\mathsf{GammaZetaFactorSeal}^{\uparrow}$ packet is the finite seal consumed by zeta functional-equation routes when they read the factor $\Gamma(1-s)$. It sits between the constructive Gamma surface of chapter 75, the application packet of chapter 588, and the witness packet of chapter 580. Its role is to make the Gamma factor a displayed NameCert packet rather than a horizon mention inside the zeta continuation application.

Definition 1101.1 (Gamma-Zeta factor seal carrier)**.** *A* $\mathsf{GammaZetaFactorSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, R, F, A, H, C, Q, N)$$

with the following displayed rows:

1. *D is the Gamma-domain row for the input $1-s$, including the pole exclusion and complex-history carrier inherited from* $\mathsf{Gamma}^{\uparrow}$*;*
2. *R is the reflected-input row, recording the source input s and the displayed reflection route to $1-s$;*
3. *F is the functional-equation factor row, recording the product slot in which $\Gamma(1-s)$ is consumed together with the elementary factors;*
4. *A is the zeta-application row that binds the factor read to the* $\mathsf{ZetaContinuationApplication}^{\uparrow}$ *packet;*
5. *H stores componentwise* hsame *transports for D, R, F, A and for the local naming row;*
6. *C stores the finite* Cont *replay routes by which downstream consumers read Gamma-domain, reflection, factor, zeta-application, transport, provenance, and local naming data;*
7. *Q is the* Pkg *provenance row over the Gamma, zeta-application, and zeta-witness source packets;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GammaZetaFactorSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for an undisplayed Gamma value, an external functional-equation proof, a second zeta-application packet, a branch outside the reflected-input route, or a nontrivial-zero classifier.

Lean4 ✓ `BEDC.Derived.GammaZetaFactorSealUp.GammaZetaFactorSealTasteGate single carrier alignment`

Theorem 1101.2 (Gamma-Zeta factor seal NameCert obligations)**.** *For the carrier of Definition 1101.1, the local* $\mathsf{NameCert}_{\mathsf{GammaZetaFactorSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (D, R, F, A, H, C, Q, N),$$

Gamma-domain exactness through D, reflected-input exactness through R, functional-equation factor exactness through F, zeta-application exactness through A, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through Q, and local naming through N.*

Proof. Project the displayed carrier. The rows D, R, F, A are the complete analytic content of the seal, while H, C, Q, N transport, replay, package, and name the same finite packet.

Read the source packets in their displayed roles. The $\mathsf{Gamma}^{\uparrow}$ chapter supplies the Gamma-domain row, the $\mathsf{ZetaContinuationApplication}^{\uparrow}$ chapter supplies the application row that consumes the factor, and the $\mathsf{ZetaContinuationWitness}^{\uparrow}$ chapter supplies the continuation witness route in which the factor is replayed.

The NameCert surface is exhausted by these rows. Since the carrier has no coordinate for an undisplayed Gamma value, an external functional-equation proof, a second zeta-application packet, an outside branch, or a nontrivial-zero classifier, no accepted read can obtain any of those objects from the seal. □

Theorem 1101.3 (Functional-equation Gamma-factor readback)**.** *Every accepted* $\mathsf{GammaZetaFactorSeal}^{\uparrow}$ *consumer that reads the functional-equation Gamma factor must receive the displayed packet*

$$S = (D, R, F, A, H, C, Q, N).$$

In particular, a read of $\Gamma(1-s)$ is accepted only when the Gamma-domain row, reflected-input row, functional-equation factor row, zeta-application row, componentwise transport, replay route, provenance row, and local name occur in the same finite packet.

Proof. Start with the carrier inventory in Definition 1101.1. A Gamma-factor read has only one factor coordinate, namely F, and that coordinate is displayed together with D, R, A, H, C, Q, N.

Apply the NameCert obligations of Definition 1101.2. The readback route must preserve the Gamma-domain row through D, the reflected input through R, the factor slot through F, and the zeta application through A before it can replay through C.

The missing alternatives named in the carrier definition have no coordinate. Thus the accepted readback is exactly the finite seal packet, not a detached Gamma theorem or an external functional-equation route. □

Theorem 1101.4 (Gamma-Zeta factor seal non-escape)**.** *Every downstream zeta functional-equation route that consumes a* $\mathsf{GammaZetaFactorSeal}^{\uparrow}$ *packet factors through D, R, F, A, H, C, Q, N. Requests for a Gamma value outside the displayed domain row, an unreflected input, an undisplayed functional-equation factor, a second application packet, a hidden transport, or a nontrivial-zero conclusion are rejected by the carrier inventory.*

Proof. Let a downstream route be given. By carrier admission, the route can inspect only the Gamma-domain row, reflected-input row, factor row, application row, transport row, replay row, provenance row, and local name.

Componentwise transport keeps these rows separate. A Gamma-domain read returns to D, an input-reflection read returns to R, a factor read returns to F, and a zeta-application read returns to A.

Every request listed in the statement asks for a coordinate absent from the carrier. The route therefore remains a read of the displayed seal and cannot export any stronger analytic object. □

Falsifiable prediction: A zeta functional-equation read using a Gamma factor is rejected unless the Gamma-domain row, reflected-input row, functional-equation row, and zeta-application row appear in the same displayed packet.

Independence witness: Nearest siblings: GammaFunction, ZetaContinuationApplication, ZetaContinuationWitness, Complex, and Holomorphic. None is bijective to GammaZetaFactorSealUp because none simultaneously stores the Gamma-domain row, reflected-input row, functional-equation factor row, zeta-application row, transport, replay, provenance, and local name.

The Lean TasteGate boundary is `lean4/BEDC/Derived/GammaZetaFactorSealUp/TasteGate.lean`. It must provide explicit `Nontrivial GammaZetaFactorSealUp` and `FieldFaithful GammaZetaFactorSealUp` instances so downstream zeta functional-equation routes can consume the same finite seal packet without collapsing the Gamma-domain, reflected-input, factor, application, transport, replay, provenance, and naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GammaZetaFactorSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{GammaZetaFactorSeal}^{\uparrow}$ packet is generated from Gamma-domain, reflected-input, functional-equation factor, zeta-application, componentwise transport, displayed continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GammaZetaFactorSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1101.1, Definition 1101.2, Definition 1101.3, Definition 1101.4, the falsifiable prediction, and the independence witness bind the finite Gamma-factor seal consumed by zeta functional-equation routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GammaZetaFactorSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close the full constructive Gamma function, the zeta functional equation, meromorphic uniqueness, nontrivial-zero classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires explicit carrier, NameCert, readback, non-escape, TasteGate, Nontrivial, and FieldFaithful theorems for the displayed Gamma-Zeta factor seal.

Chapter 1102

A Concrete Naming Certificate for $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$

The $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$ packet realizes the substitution audit map of chapter 9380 as finite NameCert data. It consumes the closed shift and substitution boundary of chapter 527, the authorized generator surface of chapter 525, and the gap-closure boundary of chapter 1006. Its role is to expose exactly the rows by which a closed shift or closed substitution route may be audited before a consumer reads the operation result.

Definition 1102.1 (SubstitutionBoundaryAudit carrier)**.** *A* $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed source-term row whose boundary is being audited;*
2. *V is the closed value row used by a substitution route, or the inert value row for a shift-only route;*
3. *d is the displayed depth row at which shift or substitution is read;*
4. *C_{src} is the closedness witness for the source term at depth d;*
5. *C_{val} is the closedness witness for the value row at the same audit boundary;*
6. *Q is the shift-boundary row recording the closed shift operation and its classifier read;*
7. *R is the substitute-boundary row recording the closed substitution operation and its classifier read;*
8. *L is the audit ledger for operation order, closed composition, and generator-closure classification;*
9. *E is the audit-exposure row that must be read before either Q or R is exported to a downstream consumer;*
10. *H records componentwise* hsame *transport for the term, value, depth, closedness, operation, ledger, exposure, provenance, and local naming rows;*
11. *C records displayed* Cont *routes for closedness reads, shift reads, substitution reads, ledger reads, and audit exposure;*

12. *P is the* Pkg *provenance over chapter 9380,* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstitutionBoundaryAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open source term, host substitution semantics, subject reduction, unrestricted generator recursion, hidden evaluator state, or a gap discharge outside the displayed packet.

Lean4 ✓ `BEDC.Derived.SubstitutionBoundaryAuditUp.SubstitutionBoundaryAuditTasteGate single carrier alignment`

Theorem 1102.2 (SubstitutionBoundaryAudit NameCert obligations)**.** *For the carrier of Definition 1102.1, the local* $\mathsf{NameCert}_{\mathsf{SubstitutionBoundaryAudit}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability, closedness-read order, audit non-escape, and consumer exactness for*

$$A = (T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L, E, H, C, P, N).$$

Carrier admission exposes all displayed rows; classifier stability keeps the shift row Q and substitute row R attached to the same term, value, and depth rows; closedness-read order places C_{src} and C_{val} before either operation row; audit non-escape forces L, E, H, C, P, N to travel with the operation reads; consumer exactness says that every accepted consumer of Q or R reads only this packet.

Proof. Carrier admission is projection from Definition 1102.1. The displayed tuple lists the source, value, depth, closedness, operation, ledger, exposure, transport, continuation, provenance, and local naming rows.

Classifier stability follows from the closed shift and substitution boundary of chapter 527. Both operation rows are read at the same displayed term, value, and depth boundary, so no second classifier source is available.

Closedness-read order is the route through C_{src} and C_{val}. The authorized generator surface of chapter 525 contributes only the permitted generator-closure classifier rows recorded in L and does not supply a replacement closedness witness.

Audit non-escape is the row E together with the ledger, transport, continuation, provenance, and local name rows L, H, C, P, N. The gap-closure boundary of chapter 1006 remains a sibling refusal surface: it can be cited as a boundary, but it is not discharged by this packet.

Finally, consumer exactness follows because Q and R are the only operation rows and E is their shared audit-exposure gate. Any accepted consumer route for a closed shift or substitution result therefore factors through the displayed packet and nowhere else. □

Theorem 1102.3 (SubstitutionBoundaryAudit falsifiable route)**.** *Any accepted* $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$ *consumer route that reaches Q or R must expose both the relevant closedness rows $C_{\mathsf{src}}, C_{\mathsf{val}}$ and the audit ledger L before the operation row. A route that reaches Q or R while bypassing those rows is not an accepted carrier route.*

Lean4 ✓ `BEDC.Derived.SubstitutionBoundaryAuditUp.SubstitutionBoundaryAuditCarrier falsifiable route`

Proof. Start with Definition 1102.2. It places closedness-read order and audit non-escape among the local certificate obligations.

If a route reaches the shift row Q, the classifier stability clause ties Q to $T, d, C_{\mathsf{src}}, L, E$ and to the displayed transport and continuation rows. The shift read therefore cannot be consumed before the source closedness and ledger rows are visible.

If a route reaches the substitute row R, the same obligation surface ties R to $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, L, E$. The value closedness row is part of the substitution boundary, so a consumer that bypasses it is outside the carrier.

Thus the prediction is refuted exactly by a counter-route that accepts Q or R without the relevant closedness and ledger rows. Such a route would contradict the displayed NameCert surface rather than extend it. □

Theorem 1102.4 (SubstitutionBoundaryAudit independence witness)**.** *The carrier of Definition 1102.1 is not bijective, row for row, to any of* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, *or* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$. *A bijection preserving local NameCert obligations would have to preserve the dedicated audit-exposure row E coupled to both the shift boundary row Q and the substitute boundary row R, and no listed sibling carries exactly that joint exposure surface.*

Proof. Compare first with $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$. That sibling supplies the closed shift and substitution boundary rows, but it does not isolate a separate audit-exposure row E whose only role is to gate both operation rows before downstream use.

Compare next with $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ and $\mathsf{GapClosureBoundary}^{\uparrow}$. The generator sibling supplies permitted generator-recursion rows, while the gap sibling records a closure boundary. Neither carries the paired shift and substitute operation rows together with the closedness, ledger, and exposure order required here.

Finally compare with $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$. It records candidate and endpoint normalization routes with obstruction retention, not the term/value/depth substitution audit packet. Since each sibling omits at least one of Q, R, L, E or carries a different endpoint surface, no carrier bijection can preserve all NameCert obligations. □

Falsifiable prediction: Any accepted SubstitutionBoundaryAuditUp consumer route that reaches Q or R must expose both the relevant ClosedAt rows and the audit ledger before the operation row; a counter-route bypassing those rows refutes the chapter.

Independence witness: closedtermsubstitutionboundary, authorizedgeneratorrecursor, gapclosureboundary, metacicnormalizationfrontier: the audit carrier is not bijective to any sibling because it has a dedicated audit-exposure row E coupled to both shift and substitute boundary reads.

The later Lean TasteGate file will provide $\mathsf{Nontrivial\ SubstitutionBoundaryAudit}^{\uparrow}$ with two explicit witness packets, one for a shift audit route and one for a substitute audit route, and $\mathsf{FieldFaithful\ SubstitutionBoundaryAudit}^{\uparrow}$ by matching and injection over the displayed fields $T, V, d, C_{\mathsf{src}}, C_{\mathsf{val}}, Q, R, L$

Closure status for $\mathsf{NameCert}_{\mathsf{SubstitutionBoundaryAudit}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$ packet is generated from a source term, value row, depth row, closedness witnesses, shift and substitute boundary rows, audit ledger, audit-exposure row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstitutionBoundaryAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1102.1, Definition 1102.2, Definition 1102.3, and Definition 1102.4 display the finite NameCert surface for closed shift and substitution audit routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstitutionBoundaryAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, unrestricted substitution commutation, open-term substitution semantics, generator recursion, gap discharge, or bridge checking for $\mathsf{SubstitutionBoundaryAudit}^{\uparrow}$.

Upgrade path: Obligation closure requires a Lean carrier encoding, checked NameCert obligation theorem, checked route theorem, independence theorem, and TasteGate nontriviality with field faithfulness for the same audit packet.

Chapter 1103

A Concrete Naming Certificate for $\mathsf{WitnessedDescentLedger}^{\uparrow}$

Falsifiable prediction: Every public downward use of an upward name factors through the displayed witness row of an accepted $\mathsf{WitnessedDescentLedger}^{\uparrow}$ packet.

Independence witness: russellboundary, definitedescriptionboundary, contextualclassreading, russellanalysisladder: the witnessed-descent ledger is not bijective to any sibling because it isolates the source, bridge, descent request, witness, transport, provenance, and local NameCert rows for downward use, while the siblings bind the full Russell boundary, description-expansion boundary, contextual-class reading, or analysis ladder surface.

The $\mathsf{WitnessedDescentLedger}^{\uparrow}$ packet realizes the witnessed-descent restriction of chapter 9457 as a concrete NameCert surface. It sits beside the definite-description boundary of chapter 661: that chapter controls when a description may be consumed as denoting, while this chapter controls when an already licensed upward name may be read downward by a consumer.

The packet also consumes the neighboring Russell surfaces. $\mathsf{RussellBoundary}^{\uparrow}$ supplies the five-fold boundary, $\mathsf{ContextualClassReading}^{\uparrow}$ supplies scoped classifier reading, and $\mathsf{RussellAnalysisLadder}^{\uparrow}$ supplies the analysis-spine context in which witnessed descent is one row among the full Russell discipline. The ledger below isolates the descent row itself so later real, rational-sequence, and bridge-readback consumers can cite a finite witness route rather than a slogan about harmless level passage.

Definition 1103.1 (Witnessed-descent ledger carrier)**.** *A* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, B, D, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row naming the upward interface whose public name is being consumed;*
2. *B is the bridge row naming the stratum passage from the upward interface to the downward consumer boundary;*
3. *D is the descent-request row recording the requested downward use of the upward name;*
4. *W is the witness row certifying that this particular downward use is licensed;*
5. *H records componentwise* hsame *transport for source, bridge, descent request, witness, continuation, provenance, and local name rows;*

6. *C records the displayed* Cont *replay by which a consumer moves from S through B, D, W and returns to the local ledger;*
7. *P is the* Pkg *provenance over* BHist*, upward names, bridge rows, descent requests, witnesses,* hsame*,* Cont*, and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{WitnessedDescentLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a stratum collapse, host identity between source and target, reducibility principle, quotient equality, proof-irrelevance shortcut, or unrecorded downward read.

Lean4 (definition) `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerUp`
Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerTasteGate single carrier alignment`

Theorem 1103.2 (Witnessed-descent NameCert obligations)**.** *For the carrier of Definition 1103.1, the local* $\mathsf{NameCert}_{\mathsf{WitnessedDescentLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (S, B, D, W, H, C, P, N),$$

source exposure through S, bridge exposure through B, descent-request accountability through D, witness exposure through W, componentwise transport through H, continuation replay through C, package provenance through P, local naming through N, and non-collapse of the upward and downward strata.

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerNamecertObligations`

Proof. Carrier admission is projection from Definition 1103.1. The source, bridge, request, witness, transport, replay, provenance, and local name are all displayed rows of the same finite packet.

The source and bridge obligations are S and B. They identify which upward name is being consumed and which stratum passage is in scope before any downward read can proceed.

The request and witness obligations are D and W. A consumer may ask for a downward read only through D, and the read is licensed only by the displayed witness row W.

Finally H, C, P, N keep the read local to the packet. They transport, replay, cite, and name the same descent route, and no row remains from which a collapse, quotient equality, proof-irrelevance shortcut, or unrecorded downward use could be extracted. □

Theorem 1103.3 (Witnessed-descent public readback)**.** *Every public downward read of an accepted* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *packet factors through the rows*

$$S, B, D, W, H, C, P, N.$$

In particular, the read exposes the source upward name, the bridge row, the descent request, the witness row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before the downward consumer receives the name.*

Proof. Start from the NameCert obligations of Definition 1103.2. They list the complete row surface available to a public consumer.

The downward read must first identify its source through S and its bridge through B. It must then record the requested use through D and expose the certificate witness through W.

The structural rows H, C, P, N preserve that route as a finite BEDC packet. Since the carrier has no hidden coordinate for source-target identity, external reducibility, quotient equality, or unrecorded witness selection, every public downward read factors through the displayed ledger. □

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerPublicReadback`

Theorem 1103.4 (Witnessed-descent sibling boundary)**.** *The* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *packet is a sibling of* $\mathsf{RussellBoundary}^{\uparrow}$, $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$, $\mathsf{ContextualClassReading}^{\uparrow}$, *and* $\mathsf{RussellAnalysisLadder}^{\uparrow}$, *not a replacement for them. It consumes their boundary, description, contextual, and ladder surfaces only as neighboring rows and exports exactly the descent ledger rows* S, B, D, W, H, C, P, N *for downstream bridge-readback consumers.*

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerSiblingBoundary`

Proof. Read the neighboring chapters as separate NameCert surfaces. $\mathsf{RussellBoundary}^{\uparrow}$ binds the fivefold Russell discipline, $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$ binds description-expansion use, $\mathsf{ContextualClassReading}^{\uparrow}$ binds scoped class-like readings, and $\mathsf{RussellAnalysisLadder}^{\uparrow}$ binds the analysis-spine packet.

The witnessed-descent ledger projects only the descent route. Its carrier names the source, bridge, request, witness, transport, replay, provenance, and local name rows needed by a downward consumer.

Therefore no sibling is collapsed into this packet and this packet is not a full Russell boundary. It is the concrete descent ledger that later consumers may cite when an upward name is read downward. □

Theorem 1103.5 (Displayed witness accountability)**.** *Every accepted* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *carrier* $L = (S, B, D, W, H, C, P, N)$ *accounts for a downward read only through the displayed witness row* W*. The TasteGate-facing carrier alignment records that the single carrier exposed to downstream consumers is the same finite packet whose rows display the source, bridge, descent request, witness, transport, replay, provenance, and local naming data.*

Proof. Use the carrier of Definition 1103.1. The descent request row D records the requested downward use, while W is the only row licensing that particular use.

The checked TasteGate alignment keeps the public carrier from being replaced by a detached event-flow presentation. Thus every consumer that reads the carrier reads the same displayed W row before any downward name is available.

The structural rows H, C, P, N then transport, replay, source, and name this route. Since the carrier omits unrecorded downward reads and proof-irrelevance shortcuts, witness accountability is exactly the displayed-row obligation. □

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerDisplayedWitnessAccountability`

Theorem 1103.6 (Witnessed-descent noncollapse transport)**.** *Componentwise* hsame *transport for an accepted* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *packet preserves the separation between the source row* S*, bridge row* B*, descent-request row* D*, and witness row* W*. It never identifies the upward source stratum with the downward consumer boundary, and it never converts the witness row into quotient equality or proof-irrelevant reducibility.*

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerNoncollapseTransport`

Proof. Read transport through the row H in Definition 1103.1. It is componentwise: source, bridge, request, witness, continuation, provenance, and local name rows are transported as separate displayed coordinates.

The non-collapse claim follows from this row separation. Transport can preserve the finite route from S through B, D, W, but it has no coordinate that turns the bridge into identity between strata.

Finally replay through C and source through P, N. The omitted coordinates exclude quotient equality, proof irrelevance, and reducibility, so the transported packet remains a witnessed descent ledger rather than a collapse principle. □

Theorem 1103.7 (Witnessed-descent bridge-readback exhaustion)**.** *Every bridge-readback consumer of an accepted* $\mathsf{WitnessedDescentLedger}^{\uparrow}$ *packet must inspect the finite route*

$$S, B, D, W, H, C, P, N$$

before it can consume the downward name. The route exhausts the readback surface: there is no additional bridge row, hidden witness selector, global descent rule, quotient equality, or identity between upward and downward strata available to the consumer.

Proof. Start from the public readback theorem Definition 1103.3. It already forces every public downward read through the source, bridge, descent request, witness, transport, replay, provenance, and local NameCert rows.

For a bridge-readback consumer, the bridge row B is therefore not optional. The consumer reaches the requested downward use only after D and the displayed witness W have been inspected.

The remaining rows H, C, P, N close the route inside the packet. Since no second bridge row, selector, global rule, quotient equality, or stratum identity coordinate exists, the listed route exhausts the bridge-readback surface. □

Lean4 ✓ `BEDC.Derived.WitnessedDescentLedgerUp.WitnessedDescentLedgerBridgeReadbackExhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{WitnessedDescentLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{WitnessedDescentLedger}^{\uparrow}$ packet is generated from a source upward-name row, a bridge row, a descent-request row, a witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WitnessedDescentLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1103.1, Definition 1103.2, Definition 1103.3, Definition 1103.4, Definition 1103.5, Definition 1103.6, and Definition 1103.7 bind the complete mature paper-side route through finite source, bridge, request, witness, public readback, noncollapse transport, and bridge-readback exhaustion rows for downward use of upward names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WitnessedDescentLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove Lean-side formal verification, a checked standard bridge, a global descent rule, a reducibility principle, quotient equality, proof irrelevance, or identity between upward and downward strata.

Upgrade path: The paper-side route is mature; verification-axis progress requires Lean-side checking of carrier admission, witness-row exactness, public readback factorization, sibling noncollapse, noncollapse transport, and bridge-readback exhaustion.

Chapter 1104

Stationary Window Seal Exhaustion Name Certificate Construction

The $\mathsf{StationaryWindowSealExhaustion}^{\uparrow}$ packet realizes the transported stationary finite-window seal of chapter 9238 as a single finite $\mathsf{NameCert}$ carrier. It packages the rational seed, constant stream window, regular-rational seal readback, dyadic ledger, real-facing seal, exhaustion row, transport rows, continuations, provenance, and local naming so that downstream RealUp seal consumers read one L10 surface rather than five disconnected public readings. The route consumes the sibling packet surfaces of chapter 37, chapter 126, chapter 301, chapter 291, and chapter 38, and remains adjacent to $\mathsf{DiagonalCofinalTail}^{\uparrow}$ and $\mathsf{StationaryWindowLattice}^{\uparrow}$.

Definition 1104.1 (Stationary window seal exhaustion carrier)**.** *A* $\mathsf{StationaryWindowSealExhaustion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, S, R, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the paired* $\mathsf{Rat}^{\uparrow}$ *seed row, including the two rational history carriers and their transported* $\mathsf{RatHistoryClassifier}$ *row;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *finite-window row induced by Q;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window seal consumption row over the same stream window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger for that window;*
5. *W is the* $\mathsf{Real}^{\uparrow}$ *finite-window classifier row reached only through S, R, D;*
6. *E is the transported stationary seal exhaustion row of Definition 9238.18;*
7. *H records componentwise* hsame *transports for Q, S, R, D, W, E and the structural rows;*
8. *C records the displayed* Cont *routes from Q to S, from S to R, from R to D, from D to W, and from E to each public reading;*
9. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiagonalCofinalTail}^{\uparrow}$, $\mathsf{StationaryWindowLattice}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryWindowSealExhaustion}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unbounded window family, a selected limit, quotient-stream equality, host real equality, or an ambient completeness principle.

Lean4 ✓ BEDC.Derived.StationaryWindowSealExhaustionUp.TasteGate.StationaryWindowSealExhaustionTasteGate
single carrier alignment

Theorem 1104.2 (Stationary window seal exhaustion NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StationaryWindowSea}}$ *surface for Definition 1104.1 consists exactly of carrier admission for* $L = (Q, S, R, D, W, E, H, C, P, N)$*,* $\mathsf{Rat}^{\uparrow}$ *seed exposure through* Q*,* $\mathsf{StreamName}^{\uparrow}$ *constant-window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *seal consumption exposure through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exposure through* D*,* $\mathsf{Real}^{\uparrow}$ *finite-window classifier exposure through* W*, transported stationary exhaustion through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from the packet L. The five public readings are exactly Q, S, R, D, W, and the exhaustion row E names the theorem that forces those readings to agree with the same transported rational seed.

Apply Definition 9238.18. It identifies the certified finite observation window with the RatUp seed classifier, StreamNameUp constant-window classifier, RegSeqRatUp finite-window seal consumption row, DyadicRatCoreUp observation ledger, and RealUp finite-window classifier.

The structural rows H, C, P, N preserve that single route under componentwise transport, continuation replay, package provenance, and local naming. Since the carrier has no extra selector, quotient, host-equality, or completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1104.3 (Stationary window seal exhaustion L10 handoff)**.** *Every accepted downstream consumer that reads the transported stationary seal through* $\mathsf{StationaryWindowSealExhaustion}^{\uparrow}$ *receives precisely the finite L10 packet*

$$L = (Q, S, R, D, W, E, H, C, P, N).$$

The consumer first reads the $\mathsf{Rat}^{\uparrow}$ *seed row, then the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{Real}^{\uparrow}$ *seal row, the transported exhaustion row, componentwise transports, continuation replay, provenance, and local naming. The handoff exports no second carrier, hidden selector, quotient-stream equality, host real equality, or ambient completion theorem.*

Proof. Project the accepted carrier of Definition 1104.1. This gives the displayed L10 rows $Q, S, R, D, W, E, H, C, P, N$ and no further public coordinates.

Use Definition 1104.2. It binds the five public readings to the same local naming surface and routes them through the transported stationary exhaustion theorem.

Finally compose with the finite faces already consumed by $\mathsf{DiagonalCofinalTail}^{\uparrow}$ and $\mathsf{StationaryWindowLattice}^{\uparrow}$. Those sibling surfaces read the same RatUp seed, StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal rows, so a downstream consumer can only receive the single packet L before using the transported stationary seal. □

Falsifiable prediction: Any accepted consumer that reads the transported stationary seal without the RatUp seed, StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealUp seal rows refutes the $\mathsf{StationaryWindowSealExhaustion}^{\uparrow}$ packet.

Independence witness: transportedstationarydiagonal, stationarywindowlattice, diagonalcofinaltail, realtailagreementseal, and realcompletionselectorseal: none is bijective to StationaryWindowSealExhaustion because this carrier alone stores the five public readings together with the exhaustion row, transport rows, continuation replay, provenance, and local naming as one L10 handoff.

The Lean realization must provide Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/StationaryW` before the packet can be promoted to a checked TasteGate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryWindowSealExhaustion}^{\uparrow}}$.

Constructive story: A $\mathsf{StationaryWindowSealExhaustion}^{\uparrow}$ packet is generated from a paired rational seed row, constant StreamName finite-window row, RegSeqRat seal-consumption row, DyadicRatCore ledger, RealUp finite-window classifier, transported stationary exhaustion row, componentwise hsame transports, displayed Cont routes, provenance, and local naming. The carrier reads five public faces of the same stationary window and binds them through the exhaustion row. The same finite route is preserved when the packet is handed to downstream RealUp seal consumers. The classifier hands them one finite L10 route: rational seed, stream window, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and name.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StationaryWindowSealExhaustion}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the stationary window seal exhaustion carrier, NameCert obligations, L10 handoff theorem, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StationaryWindowSealExhaustion}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.StationaryWindowSealExhaustionUp.TasteGate.StationaryWindowSealExhaustionTasteGate\_sin`

Bridge status: none

Origin: ai

Not claimed: It does not claim unbounded window families, selected limits, quotient-stream equality, host real equality, or ambient completeness.

Upgrade path: To reach obligationClosure, formalize the five-face stationary window handoff and transported exhaustion row as a checked finite carrier.

Chapter 1105

A Concrete Naming Certificate for $\mathsf{InterHistSynchronizationRefusal}^{\uparrow}$

The $\mathsf{InterHistSynchronizationRefusal}^{\uparrow}$ packet realizes the no-global-clock boundary of chapter 9418 as a finite two-stream certificate. It consumes the local stream carrier of chapter 603, the observer-Hist identity carrier of chapter 479, and the locality-transport carrier of chapter 889. Its role is to name the refusal of a proposed synchronization token without replacing the two local Hist streams by a shared clock, host time index, quotient stream, or real-completion timing oracle.

Definition 1105.1 (InterHistSynchronizationRefusal carrier)**.** *An* $\mathsf{InterHistSynchronizationRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the first local* $\mathsf{HistTimeStream}^{\uparrow}$ *row, read as a finite* BHist *stream with its own* Cont*-step ledger;*
2. *B is the second local* $\mathsf{HistTimeStream}^{\uparrow}$ *row, read as a finite* BHist *stream with its own* Cont*-step ledger;*
3. *T is the proposed synchronization token whose admission is refused by this packet;*
4. *R is the refusal ledger recording that T is not a shared clock, host time index, quotient stream, global synchronization row, or real-completion timing oracle;*
5. *H records componentwise* hsame *transport for the two local streams, token row, refusal ledger, route rows, provenance row, and local naming row;*
6. *C records displayed* Cont *replay for reading A, B, T, R, H, P, and N;*
7. *P is the* Pkg *provenance over* $\mathsf{InterHistSynchronizationRefusal}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ObserverLocalityTransport}^{\uparrow}$, $\mathsf{ObserverFrameRefusal}^{\uparrow}$, $\mathsf{ObserverFilter}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{InterHistSynchronizationRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a common clock value, common index set, host-time comparison, quotient of the two streams, synchronized observer frame, or real-number completion time.

Theorem 1105.2 (InterHistSynchronizationRefusal NameCert obligations)**.** *For the carrier of Definition 1105.1, the local* $\mathsf{NameCert}_{\mathsf{InterHistSynchronizationRefusal}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier admission for* $S = (A, B, T, R, H, C, P, N)$*;*
2. *two local-stream readbacks, one through* A *and one through* B*;*
3. *refusal-ledger exactness for the proposed synchronization token* T *through* R*;*
4. *componentwise* hsame *stability through* H *and displayed* Cont *replay through* C*;*
5. *no-global-clock non-escape through* P *and* N*, which keeps every consumer inside the two local streams and the refusal ledger.*

Lean4 ✓ BEDC.Derived.InterHistSynchronizationRefusalUp.InterHistSynchronizationRefusalTasteGate
single carrier alignment

Proof. Carrier admission is projection from Definition 1105.1. The packet lists the two local stream rows, the proposed synchronization token, the refusal ledger, transport, replay, provenance, and local naming row.

The two positive readbacks are A and B. The negative readback is R, which records the exact status of T as refused synchronization data rather than an admitted clock coordinate.

The structural rows H, C, P, N preserve that split. Transport and replay act componentwise on the displayed rows, while provenance and local naming certify that the consumer has not inserted a shared clock outside the packet. □

Theorem 1105.3 (InterHistSynchronizationRefusal no-global-clock boundary)**.** *Any consumer of an accepted* InterHistSynchronizationRefusal$^{\uparrow}$ *packet may read only the two local streams* A *and* B*, the proposed synchronization token* T *as refused data, the refusal ledger* R*, the componentwise* hsame *transport* H*, the displayed* Cont *replay* C*, the* Pkg *provenance* P*, and the local* NameCert *row* N*. The consumer never receives a shared clock, host time index, quotient stream, global synchronization row, synchronized observer frame, or real-completion timing oracle.*

Proof. Project the carrier of Definition 1105.1. Its rows are exactly A, B, T, R, H, C, P, N, with A and B giving the local streams and R giving the refusal status of T.

Apply the NameCert obligations of Definition 1105.2. The admissible reads are the two local-stream readbacks, refusal-ledger exactness, componentwise transport, displayed replay, provenance, and local naming.

No displayed row is a shared clock, host time index, quotient stream, global synchronization row, synchronized observer frame, or real-completion timing oracle. A consumer that exports one of those objects has introduced data not present in the accepted packet. □

Falsifiable prediction: If the refusal row is removed while the two local streams remain, the packet can still name two HistTimeStream rows but cannot block a consumer from treating them as synchronized.

Independence witness: Nearest siblings: histtimestream, observerhistidentity, observerlocalitytransport, observerframerefusal, and observerfilter. HistTimeStreamUp names one local stream rather than a two-stream refusal; ObserverHistIdentityUp identifies observer with Hist but does not carry a synchronization token; ObserverLocalityTransportUp transports locality rows but does not reject a shared clock; ObserverFrameRefusalUp rejects a frame but not the two-stream synchronization token; ObserverFilterUp selects observation rows without naming a clock-refusal carrier. No bijection preserving two local streams, refusal token, refusal ledger, hsame transport, Cont replay, provenance, and local naming identifies these packets.

The Lean site `lean4/BEDC/Derived/InterHistSynchronizationRefusalUp/TasteGate.lean` must provide explicit Nontrivial InterHistSynchronizationRefusal$^{\uparrow}$ and FieldFaithful InterHistSynchronizationRefusal$^{\uparrow}$ instances. These instances must preserve the two local streams, proposed synchronization token, refusal ledger, componentwise hsame transport, displayed Cont replay, provenance, and local naming row as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{InterHistSynchronizationRefusal}^{\uparrow}}$.

Constructive story: An InterHistSynchronizationRefusal$^{\uparrow}$ packet is generated from two local HistTimeStream rows, a proposed synchronization token, refusal ledger, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The BHist carrier records the two local streams as visible data and then refuses the token that would identify them by a shared clock. Each accepted read keeps both stream rows and the refusal token in the same packet. The classifier accepts the two-stream packet only with the refusal ledger attached, so transport and continuation replay preserve locality rather than manufacturing synchronized observer time or a global frame.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InterHistSynchronizationRefusal}^{\uparrow}}$, seed)

Scope closed: This chapter binds the inter-Hist synchronization refusal carrier, non-escape theorem, prediction, and independence witness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InterHistSynchronizationRefusal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim shared clocks, host time indices, quotient streams, global synchronization rows, synchronized frames, or real-completion timing oracles.

Upgrade path: To reach obligationClosure, check the two local streams and refusal token as distinguishable nontrivial and field-faithful rows.

Chapter 1106

A Concrete Naming Certificate for NoGlobalSynchronizationLedger$^{\uparrow}$

The NoGlobalSynchronizationLedger$^{\uparrow}$ packet realizes the no-global-synchronization premise of Definition 9419.4 as a positive finite ledger row. It is downstream of NoGlobalSyncBoundary$^{\uparrow}$ but not the same object: the boundary records that a global synchronization object is refused, while the ledger records the accepted local rows that consume that refusal in inter-Hist locality, causal-rate, and observer-transport chapters. The ledger consumes the no-global-sync boundary of chapter 1003 and the local read-site carrier of chapter 509 before handing the refusal row to inter-Hist, causal-rate, and observer-transport consumers.

Falsifiable prediction: Any accepted inter-Hist consumer that reads a shared synchronization Hist or global clock row refutes the ledger packet.

Independence witness: Nearest siblings: noglobalsyncboundary, localitycell, observerlocalitytransport, interhistinvariant, crosshistcausal. None is bijective because the ledger has a positive no-global-sync refusal row consumed by locality and rate chapters, while the siblings either record a boundary, a locality cell, an invariant, or a causal route without this carrier row.

Definition 1106.1 (NoGlobalSynchronizationLedger carrier)**.** *A* NoGlobalSynchronizationLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H_0, H_1, R, B, O, I, K, S, C, P, N)$$

with the following displayed rows:

1. *H_0 and H_1 are the two source history rows whose local reads are compared;*
2. *R is the positive no-global-synchronization refusal row, recording that no shared synchronization* BHist*, global clock, privileged frame, total observer order, or selected all-observer map is admitted;*
3. *B is the* NoGlobalSyncBoundary$^{\uparrow}$ *boundary row whose refusal surface is being consumed;*
4. *O is the* LocalityCell$^{\uparrow}$ *row that binds the two local read sites before a causal or rate consumer can read them;*
5. *I is the* InterHistInvariant$^{\uparrow}$ *row that receives the displayed local comparison without adding a global frame;*
6. *K is the downstream consumer row for* CrossHistCausal$^{\uparrow}$*,* ObserverLocalityTransport$^{\uparrow}$*, or* MaxCausalRate$^{\uparrow}$*;*

7. *S records componentwise* hsame *transport for the histories, refusal row, boundary row, locality cell, invariant row, consumer row, routes, provenance, and local name;*
8. *C records displayed* Cont *replay for reading exactly those rows;*
9. *P is the* Pkg *provenance over* $\mathsf{NoGlobalSynchronizationLedger}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, $\mathsf{ObserverLocalityTransport}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NoGlobalSynchronizationLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a shared synchronization history, global clock, privileged host frame, total observer order, quotient of histories, metric spacetime, host equality oracle, or selected limit.

Lean4 ✓ `BEDC.Derived.NoGlobalSynchronizationLedgerUp.NoGlobalSynchronizationLedgerTasteGate single carrier alignment`

Theorem 1106.2 (NoGlobalSynchronizationLedger NameCert obligations)**.** *For the carrier of Definition 1106.1, the local* $\mathsf{NameCert}_{\mathsf{NoGlobalSynchronizationLedger}^{\uparrow}}$ *surface consists of carrier admission for*

$$L = (H_0, H_1, R, B, O, I, K, S, C, P, N),$$

refusal exactness through R, boundary consumption through B, observer-locality exposure through O, invariant readback through I, downstream handoff through K, componentwise hsame *transport through S, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoGlobalSynchronizationLedgerUp.NoGlobalSynchronizationLedgerUp namecert obligations`

Proof. Carrier admission is projection from Definition 1106.1. The rows $H_0, H_1, R, B, O, I, K, S, C, P, N$ are exactly the histories, refusal row, boundary row, locality row, invariant row, consumer row, transport, replay, provenance, and naming surface.

The refusal and boundary obligations are separate. The row R is the positive ledger coordinate saying which global synchronization data are absent from the accepted packet, while B records the boundary packet from which that refusal is inherited.

The locality, invariant, and consumer obligations then read O, I, K after the two source histories and the refusal row have been displayed. The structural rows S, C, P, N transport, replay, package, and name the same finite packet, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 1106.3 (NoGlobalSynchronizationLedger consumer non-escape)**.** *Every accepted* NoGlobalSynchronizatio *consumer route factors through*

$$H_0, H_1, R, B, O, I, K, S, C, P, N.$$

It may read the two source histories, the positive no-global-synchronization refusal row, the boundary row, the locality cell, the invariant row, the downstream causal or rate consumer row, displayed hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It exports no shared synchronization* BHist*, global clock, privileged frame, total observer order, metric spacetime, quotient history, host equality oracle, or selected limit.*

Lean4 ✓ `BEDC.Derived.NoGlobalSynchronizationLedgerUp.NoGlobalSynchronizationLedgerUp consumer nonescape`

Proof. Inspect the carrier coordinates in Definition 1106.1. The only positive rows are H_0, H_1, O, I, K together with the structural rows S, C, P, N.

The row R is not a hidden synchronization object; it is the refusal ledger. The row B is the boundary source for the same refusal. Thus a consumer can cite global synchronization only as excluded data and only through the displayed refusal route.

No coordinate of the packet is a shared synchronization BHist, global clock, privileged frame, total observer order, metric spacetime, quotient history, host equality oracle, or selected limit. Therefore any consumer exporting such a coordinate has left the accepted ledger packet. □

Theorem 1106.4 (NoGlobalSynchronizationLedger inter-Hist handoff)**.** *Every accepted* NoGlobalSynchronizationLe *packet supplies the no-global-frame premise required by* CrossHistCausal$^{\uparrow}$, ObserverLocalityTransport$^{\uparrow}$, *and* MaxCausalRate$^{\uparrow}$ *consumers. The handoff is the displayed route from the two source histories through the refusal row R, the boundary row B, the locality cell O, the invariant readback I, and the consumer row K, transported by S and replayed by C.*

Proof. Start with the local NameCert surface of Definition 1106.2. It exposes R and B before any locality, invariant, causal, transport, or rate consumer can read the packet.

Next read the locality and invariant rows. The locality cell O binds the two source histories as local sites, and the invariant row I receives their comparison without inserting a global frame.

Finally project the consumer row K and the structural rows S, C, P, N. These rows make the same refusal ledger available to CrossHistCausal$^{\uparrow}$, ObserverLocalityTransport$^{\uparrow}$, and MaxCausalRate$^{\uparrow}$ consumers, while Definition 1106.3 excludes shared clocks, global frames, metric spacetime, quotient histories, and host equality as alternate handoff data. □

Lean4 ✓ `BEDC.Derived.NoGlobalSynchronizationLedgerUp.NoGlobalSynchronizationLedgerUp inter hist handoff`

Theorem 1106.5 (NoGlobalSynchronizationLedger TasteGate obligation)**.** *The later Lean site* `lean4/BEDC/Derived/NoGlobalSynchronizationLedgerUp/TasteGate.lean` *must provide explicit* Nontrivial NoGlobalSynchronizationLedger$^{\uparrow}$ *and* FieldFaithful NoGlobalSynchronizationLedger$^{\uparrow}$ *instances. These instances must keep the two source histories, the positive refusal row, boundary row, locality cell, invariant row, downstream consumer row,* hsame *transport,* Cont *routes,* Pkg *provenance, and local* NameCert *row as distinguishable fields.*

Proof. The carrier separates the source histories H_0, H_1 from the refusal row R, boundary row B, locality row O, invariant row I, and consumer row K. A trivial collapse would identify refusal data with the local histories or with the consumer handoff, erasing the no-global-synchronization premise needed downstream.

Field faithfulness is the projection discipline for $L = (H_0, H_1, R, B, O, I, K, S, C, P, N)$. Each coordinate has a distinct consumer role, so a later TasteGate witness must preserve those projections before any locality, causal-rate, or transport chapter consumes the ledger. □

Closure status for NameCert$_{\text{NoGlobalSynchronizationLedger}^{\uparrow}}$.

Constructive story: A NoGlobalSynchronizationLedger$^{\uparrow}$ packet is generated from two source histories, a positive no-global-synchronization refusal row, a NoGlobalSyncBoundary row, a locality cell, an inter-Hist invariant readback row, a causal or rate consumer row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NoGlobalSynchronizationLedger}^{\uparrow}}$, seed)

Scope closed: Definition 1106.1, Definition 1106.2, Definition 1106.3, and Definition 1106.4 specify the finite no-global-synchronization ledger consumed by inter-Hist locality, causal, transport, and rate chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoGlobalSynchronizationLedger}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct a shared synchronization history, global clock, privileged frame, total observer order, metric spacetime, quotient history, Lean proof, or bridge checking for $\mathsf{NoGlobalSynchronizationLedger}^{\uparrow}$.

Upgrade path: Obligation closure requires a Lean carrier encoding, checked NameCert obligation theorem, consumer non-escape theorem, inter-Hist handoff theorem, and TasteGate nontriviality with field faithfulness.

Chapter 1107

A Concrete Naming Certificate for $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$

The $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$ packet realizes the non-axiom supply discipline of chapter 9395 as a finite NameCert audit route. It sits beside the boundary packet of chapter 1065, the admission classifier of chapter 1074, the GAP-facing closure boundary of chapter 1006, and the inscription ledger surface of chapter 520. Those siblings expose boundary names, admissible forms, ledger closure, or inscription gaps; the audit route binds the displayed boundary form, witness route, non-axiom ledger, and local NameCert obligations in one consumer-visible packet.

Definition 1107.1 (ExternalSupplyAuditRoute carrier)**.** *An* $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, W, G, L, H, P, N)$$

with the following displayed rows:

1. *B is the boundary-form row, recording the strongest non-axiom form accepted for the requested supply;*
2. *W is the witness-route row, recording the finite route by which a consumer reaches the displayed supply boundary;*
3. *G is the* GAP*-ledger row preserving the residue or socket pressure generated by the request;*
4. *L is the local route-classifier ledger, recording whether B, W, G are all present before the route may be accepted;*
5. *H records componentwise* hsame *transport for the boundary-form, witness-route, GAP-ledger, and classifier rows;*
6. *P is the* Pkg *provenance over* $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, $\mathsf{NonAxiomBoundaryForm}^{\uparrow}$, $\mathsf{NonAxiomAdmission}^{\uparrow}$, $\mathsf{InscriptionGapLedger}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, BHist, hsame, Pkg, *and* NameCert *rows;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyAuditRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for an axiom, hidden selector, quotient representative, proposition-equality token, unledgered supply witness, or host-side admission outside B, W, G, L, H, P, N.

Lean4 ✓ `BEDC.Derived.ExternalSupplyAuditRouteUp.ExternalSupplyAuditRouteTasteGate single carrier alignment`

Theorem 1107.2 (ExternalSupplyAuditRoute non-axiom boundary)**.** *Every accepted* $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$ *packet exposes an external-supply request only through its displayed non-axiom boundary form. A downstream consumer may read the boundary form B, the witness route W, the* GAP *ledger G, the route-classifier ledger L, the componentwise* hsame *row H, the package row P, and the local* NameCert *row N, but it receives no axiom, hidden selector, quotient representative, proposition-equality token, or host-side admission of the requested supply.*

Lean4 ✓ `BEDC.Derived.ExternalSupplyAuditRouteUp.ExternalSupplyAuditRouteCarrier nonaxiom boundary`

Proof. Project the carrier of Definition 1107.1. The accepted packet displays B, W, G, L, H, P, N before any consumer can read the route.

Use the distinction in Definition 9395.2 and Definition 9395.4. The boundary-form row B records which non-axiom form carries the request, while W records the finite route to that boundary.

Now read G and L. The GAP row preserves the residue or socket pressure, and the classifier ledger records acceptance only when the boundary form, witness route, and GAP ledger are present together.

The structural rows H, P, N transport, package, and name only those displayed rows. Since no coordinate stores an axiom or hidden supply witness, the accepted route keeps the requested supply external. □

Theorem 1107.3 (ExternalSupplyAuditRoute ledger non-escape)**.** *No accepted* $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$ *packet can be consumed as a supply route without its ledger row. If the route classifier reads a packet missing G, then the packet fails the route classifier and cannot expose a witness route as an accepted non-axiom supply boundary.*

Proof. Start from the carrier definition. The classifier ledger L is a row over the displayed triple B, W, G, not over the boundary form and witness route alone.

Apply the non-axiom boundary theorem above. It permits downstream reads only through B, W, G, L, H, P, N, so the GAP ledger is part of the public route rather than an optional annotation.

If G is absent, the route classifier cannot certify that the requested supply remained ledgered. The packet therefore has no accepted $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$ read and cannot be consumed as a non-axiom supply route. □

Lean4 ✓ `BEDC.Derived.ExternalSupplyAuditRouteUp.ExternalSupplyAuditRouteUp ledger nonescape`

Theorem 1107.4 (ExternalSupplyAuditRoute NameCert obligations)**.** *For the carrier of Definition 1107.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplyAuditRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (B, W, G, L, H, P, N),$$

boundary-form exposure through B, witness-route exposure through W, GAP-ledger readback through G, classifier-ledger readback through L, componentwise hsame *stability through H, package provenance through P, and local naming through N.*

Consequently a certificate consumer may audit which boundary form was accepted, which finite route witnesses the boundary, which ledger records the residue, and which local rows stabilize the route, but it may not read unlisted supply content.

Proof. Carrier admission is immediate from Definition 1107.1. The displayed packet lists exactly B, W, G, L, H, P, N.

The first obligation group exposes the supply boundary without internalizing it: B names the accepted non-axiom form, W names the route to the boundary, and G keeps the ledger pressure visible.

The second obligation group stabilizes the route. The classifier ledger L records that the route is accepted only with the ledger present, while H, P, N provide componentwise transport, provenance, and local naming over the same finite packet.

By Definition 1107.3, omitting the ledger row prevents route acceptance. Thus the NameCert obligations are exhausted by the displayed rows and do not supply any hidden content. □

Theorem 1107.5 (ExternalSupplyAuditRoute GAP-ledger factorization)**.** *Every accepted* ExternalSupplyAuditRou *packet factors its accepted route through the displayed boundary form B, witness route W,* GAP *ledger G, classifier ledger L, componentwise transport H, provenance P, and local* NameCert *row N. Before any consumer reads the route, the provenance row must pass through* ExternalSupplyBoundary$^{\uparrow}$, NonAxiomBoundaryForm$^{\uparrow}$, NonAxiomAdmission$^{\uparrow}$, InscriptionGapLedger$^{\uparrow}$, *and* GapClosureBoundary$^{\uparrow}$.

Consequently the accepted audit route cannot expose an axiom, hidden selector, unledgered supply witness, quotient representative, proposition-equality token, or host-side admission outside the finite packet $A = (B, W, G, L, H, P, N)$.

Proof. Project the carrier of Definition 1107.1. The only accepted route coordinates are the boundary-form row B, witness route W, GAP ledger G, classifier ledger L, transport row H, package provenance P, and local naming row N.

Apply Definition 1107.2. It restricts every downstream read of the external supply request to the displayed non-axiom boundary form and the rows that stabilize it.

Next use Definition 1107.3. The route cannot be accepted without G, so the supply witness is read only after the GAP ledger and classifier ledger have both been displayed.

Finally apply Definition 1107.4. The local certificate packages B, W, G, L, H, P, N over the provenance chain through ExternalSupplyBoundary$^{\uparrow}$, NonAxiomBoundaryForm$^{\uparrow}$, NonAxiomAdmission$^{\uparrow}$, InscriptionGapLedger$^{\uparrow}$, and GapClosureBoundary$^{\uparrow}$. Since no additional carrier coordinate exists, no consumer can extract an axiom, hidden selector, unledgered supply witness, quotient representative, proposition-equality token, or host-side admission. □

Falsifiable prediction: Any accepted ExternalSupplyAuditRoute packet missing the ledger row fails the route classifier.

Independence witness: Compare inscriptionevent, kernelacceptanceauditwitness, gapclosure-boundary, and inscriptiongapledger: none is bijective because none carries all boundary-form, witness-route, non-axiom ledger, and local NameCert rows.

The Lean follow-up must provide explicit Nontrivial ExternalSupplyAuditRoute$^{\uparrow}$ and FieldFaithful ExternalSupplyA instances in Derived/ExternalSupplyAuditRouteUp/TasteGate.lean before the alignment theorem BEDC.Derived.Exter can be cited by a checked paper marker.

Closure status for NameCert$_{\text{ExternalSupplyAuditRoute}^{\uparrow}}$.

Constructive story: An ExternalSupplyAuditRoute$^{\uparrow}$ packet is generated from a boundary-form row, witness-route row, non-axiom ledger, audit classifier, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier reads external supply through an auditable route: first the boundary form is displayed, then the witness route is inspected, then the ledger records why the request remains non-axiomatic. The package row records the surrounding inscription and gap surfaces consumed by the audit. The classifier rejects any route missing the ledger row, because transport and continuation replay preserve the route only as ledgered finite BEDC data.

Theory closure: TheoryGate(NameCert$_{\text{ExternalSupplyAuditRoute}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter binds the boundary-form row, witness-route row, GAP-ledger row, classifier ledger, componentwise hsame, displayed Cont, Pkg, NameCert, and the cited TasteGate marker as the scoped external-supply audit route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ExternalSupplyAuditRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim direct internal supply, unledgered witness routes, hidden axiom admission, or theorem content outside the displayed boundary route.

Upgrade path: Public closure requires a public-object boundary and checked transport, replay, provenance, and local naming coverage for the displayed external-supply audit route.

Chapter 1108

A Concrete Naming Certificate for MetaCICNormalizationCertificateRouter$^{\uparrow}$

The MetaCICNormalizationCertificateRouter$^{\uparrow}$ packet realizes the candidate-mediated route inventory of chapter 9383 as a finite NameCert router. It sits beside MetaCICNormalizationFrontier$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, and KernelInductiveAcceptance$^{\uparrow}$: those siblings expose frontier, closed-substitution, generator, compiler-flow, and inductive-acceptance rows, while this packet chooses which displayed normalization evidence is present before any closed-term normalization read is admitted.

The exact concrete carrier rows consumed by the router are supplied by chapter 892, chapter 527, chapter 525, chapter 1048, and chapter 915.

Thus the router bundles the normalization-frontier, closed-substitution, authorized-generator, compiler-recognizer, and kernel-inductive-acceptance rows into one routing surface rather than claiming a new atomic carrier.

Definition 1108.1 (MetaCIC normalization certificate router carrier)**.** *A* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, K, S, P, D, U, G, L, H, C, Q, N)$$

with the following displayed rows:

1. *A is the audit-map anchor row for chapter 9383;*
2. *K is the candidate-evidence row, including candidate, closed-candidate, and candidate-set evidence;*
3. *S is the strong-normalization consequence row read only from K;*
4. *P is the Pi and application adequacy row that keeps function, argument, and application candidates on the displayed route;*
5. *D is the closed-term projection row consumed through* ClosedTermSubstitutionBoundary$^{\uparrow}$*;*
6. *U is the authorized generator handoff row consumed through* AuthorizedGeneratorRecursor$^{\uparrow}$*;*
7. *G is the compiler recognizer row consumed through* GroundCompilerRecognizerFlow$^{\uparrow}$*;*
8. *L is the kernel inductive-acceptance row consumed through* KernelInductiveAcceptance$^{\uparrow}$*;*
9. *H records componentwise* hsame *transport for the evidence, projection, adequacy, generator, compiler, acceptance, route, provenance, and local naming rows;*

10. *C records displayed* Cont *replay from candidate evidence to strong normalization and from strong normalization to no-infinite-reduction projections;*
11. *Q records* Pkg *provenance over* MetaCICNormalizationFrontier$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, KernelInductiveAcceptance$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}}$ *row.*

The carrier has no coordinate for evidence-free closed-term strong normalization, unconditional application normalization, host normalization, quotient reduction equality, or a normalization oracle outside the displayed router.

Theorem 1108.2 (MetaCIC normalization certificate router obligation surface)**.** *For the carrier of Definition 1108.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}}$ *surface consists exactly of candidate-evidence admission through K, strong-normalization readback through S, Pi and application adequacy through P, closed-term projection through D, authorized generator handoff through U, compiler recognizer handoff through G, kernel inductive-acceptance handoff through L, route replay through C, componentwise transport through H, provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterUp single carrier alignment`

Proof. Project the carrier coordinates. The router admits normalization reads only after the audit anchor A and candidate evidence K have been displayed, and the row S is downstream of that evidence rather than a separate global theorem.

The adequacy and closed-term rows are then finite consumers. The row P records Pi and application adequacy, while D records the closed-term projection through the sibling ClosedTermSubstitutionBoundary$^{\uparrow}$ surface.

Finally attach the producer and acceptance rows. The rows U, G, L keep authorized generation, compiler recognition, and kernel inductive acceptance on the same packet, and H, C, Q, N provide only transport, replay, provenance, and local naming. Since no other row is present, the listed rows exhaust the local certificate surface. □

Theorem 1108.3 (MetaCIC normalization certificate router closed-read refusal)**.** *A closed-term normalization read of an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *packet is refused unless the displayed route contains candidate evidence, closed-candidate evidence, or candidate-set evidence in K. The rows $P, D, U, G, L, H, C, Q, N$ may route, project, transport, replay, and package such evidence, but none of them can replace K by an evidence-free normalization theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterClose`

Proof. Read the evidence side first. By Definition 1108.2, the strong-normalization row S is reachable only from the candidate-evidence row K.

Now inspect the sibling rows. MetaCICNormalizationFrontier$^{\uparrow}$ supplies the candidate-frontier boundary, ClosedTermSubstitutionBoundary$^{\uparrow}$ supplies the closed projection row, AuthorizedGeneratorRecursor$^{\uparrow}$ supplies the authorized generator handoff, GroundCompilerRecognizerFlow$^{\uparrow}$ supplies the compiler recognizer flow, and KernelInductiveAcceptance$^{\uparrow}$ supplies the inductive acceptance row. Each is consumed as a displayed row of the router rather than as a source of evidence-free normalization.

The structural rows cannot change that dependency. Transport, replay, provenance, and local naming preserve the already displayed route, so a closed-term normalization request with no candidate, closed-candidate, or candidate-set evidence has no carrier coordinate from which to read S. □

Definition 1108.4 (Normalization candidate router evidence budget)**.** *For an accepted* MetaCICNormalizationCertif *carrier*

$$R = (A, K, S, P, D, U, G, L, H, C, Q, N),$$

the candidate-mediated evidence budget is the displayed finite read

$$K, D, P, U, G, L, H, C, Q, N.$$

Here K supplies candidate, closed-candidate, or candidate-set evidence; D supplies the closed-candidate projection through $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$*; P supplies Pi and application adequacy; U supplies the authorized-generator handoff; G supplies the compiler-recognizer handoff; L supplies kernel inductive acceptance; and H, C, Q, N supply transport, replay, provenance, and local naming for the same packet.*

The budget is finite row data inside the carrier and has no coordinate for subject reduction, global confluence, host normalization, quotient reduction equality, or evidence-free normalization.

Theorem 1108.5 (Normalization candidate router evidence-budget non-escape)**.** *Every closed-normalization consumer of an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier must spend the finite evidence budget of Definition 1108.4 before it reads strong-normalization output from S. The packet exports no subject reduction theorem, global confluence theorem, host normalization theorem, quotient reduction equality, or evidence-free normalization route outside*

$$K, D, P, U, G, L, H, C, Q, N.$$

Proof. Start with the carrier and obligation surface. The carrier of Definition 1108.1 exposes $K, D, P, U, G, L, H, C, Q, N$ as separate finite rows, and Definition 1108.2 places candidate evidence before the strong-normalization readback.

Apply the closed-read refusal. Theorem 1108.3 refuses any closed normalization read with no candidate, closed-candidate, or candidate-set evidence in K, so a consumer cannot reach S without first entering the displayed evidence side.

Now use candidate determinacy. Theorem 1108.6 fixes S from the evidence in K, while D, P, U, G, L only project, check adequacy, generate, recognize, and accept that same route. Transport, replay, provenance, and naming through H, C, Q, N preserve the displayed budget.

The refused endpoints are therefore absent by inspection of the carrier coordinates. Since no row names subject reduction, global confluence, host normalization, quotient reduction equality, or an evidence-free normalizer, the closed-normalization consumer must spend exactly the finite budget before any strong-normalization output is read. □

Theorem 1108.6 (MetaCIC normalization certificate router candidate determinacy)**.** *For an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier, the strong-normalization readback row S is determined by the displayed candidate, closed-candidate, or candidate-set evidence in K. The closed projection, Pi and application adequacy, generator, compiler, acceptance, transport, replay, provenance, and local naming rows may preserve and route that evidence, but they do not create a second source for S.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterCandi determinacy`

Proof. Project the carrier of Definition 1108.1. The row K is the only carrier coordinate containing candidate, closed-candidate, or candidate-set evidence, and S is declared as the readback row downstream of it.

Apply the obligation surface and closed-read refusal theorems. Theorem 1108.2 lists K before the strong-normalization readback, and Definition 1108.3 refuses any closed read lacking such evidence.

The Lean TasteGate marker attached to the obligation surface records the single-carrier alignment for the same router. Thus the paper route and the checked carrier site agree that S has no evidence-free source outside K. □

Theorem 1108.7 (MetaCIC normalization certificate router adequacy evidence source)**.** *Every Pi or application adequacy read in the row P of an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier factors through candidate, closed-candidate, or candidate-set evidence displayed in K. The rows D, U, G, L, H, C, Q, N project, generate, recognize, accept, transport, replay, package, and name that evidence; none of those rows exports a closed normalization read without the candidate evidence already present in K.*

Proof. Project the carrier of Definition 1108.1. The row P is the Pi and application adequacy consumer, and the only evidence row upstream of the strong-normalization readback is K.

Use the obligation surface and closed-read refusal. Theorem 1108.2 places candidate admission in K before S and P, while Definition 1108.3 rejects any closed read with no candidate, closed-candidate, or candidate-set evidence.

Apply candidate determinacy. Theorem 1108.6 determines the readback S from the evidence in K, so P can only preserve that evidence along Pi and application routes. The remaining rows project, generate, recognize, accept, transport, replay, package, and name the same displayed route, and therefore no evidence-free closed normalization read is exported. □

Theorem 1108.8 (MetaCIC normalization certificate router adequacy chain)**.** *The Pi and application adequacy rows P of an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier can route only evidence already admitted by K. They cannot turn function, argument, application, generator, compiler, or kernel-acceptance rows into evidence-free normalization reads.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterAdequ`

Proof. Read the candidate row first. By Definition 1108.1, the adequacy row P is a consumer of the displayed candidate-evidence row K, not a replacement for it.

Apply candidate determinacy. Theorem 1108.6 pins the strong-normalization readback S to evidence in K, so adequacy can only preserve and transmit that already displayed evidence.

Now compare with the sibling separation theorem. Theorem 1108.12 separates this router from the frontier, closed-substitution, generator, compiler-flow, and kernel-acceptance siblings, so none of those rows can be treated as an independent normalization source. □

Theorem 1108.9 (MetaCIC normalization certificate router projection coverage)**.** *The closed-term projection row D of an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier covers exactly the displayed* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *consumer route. It projects candidate-mediated normalization evidence into the closed-term boundary and exposes no additional closed-term normalization principle outside the rows admitted by K and routed by P.*

Proof. Project the carrier row D. In Definition 1108.1, D is the closed-term projection row consumed through ClosedTermSubstitutionBoundary$^{\uparrow}$.

Use the obligation surface. Theorem 1108.2 places D beside the candidate-evidence row K and adequacy row P, so the projection is a consumer route rather than a new source of evidence.

Apply closed-read refusal and sibling separation. Theorem 1108.3 blocks any closed read without evidence in K, and Definition 1108.12 prevents collapsing the router to the closed-substitution sibling alone. Hence D covers exactly the displayed closed-term substitution boundary route. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterProje

Theorem 1108.10 (MetaCIC normalization certificate router route coverage)**.** *For an admitted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *read, strong-normalization output is reachable only through the ordered finite route*

$$A, K, S, P, D, U, G, L, H, C, Q, N.$$

Here K supplies candidate, closed-candidate, or candidate-set evidence; D supplies the closed projection; U supplies authorized generation; G supplies compiler recognition; and L supplies kernel inductive acceptance. No sibling row replaces K by an evidence-free normalization theorem.

Proof. Project the carrier of Definition 1108.1. The row inventory is exactly $A, K, S, P, D, U, G, L, H, C, Q, N$, with S downstream of K rather than a standalone normalization fact.

Use the obligation-surface theorem, Definition 1108.2, to exhaust the rows. It places candidate-evidence admission in K, Pi and application adequacy in P, the closed projection in D, authorized generation in U, compiler recognition in G, and kernel inductive acceptance in L.

Now apply closed-read refusal, Definition 1108.3. A closed-term normalization read without candidate, closed-candidate, or candidate-set evidence has no route to S.

Finally compare the sibling rows. ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, and KernelInductiveAcceptance$^{\uparrow}$ contribute projection, generation, compiler, and acceptance evidence respectively, but none is the candidate-evidence row K. Transport, replay, provenance, and naming preserve the same ordered route, so the admitted read cannot bypass K. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterRoute
Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterPacke
route coverage

Theorem 1108.11 (MetaCIC normalization certificate router kernel scope)**.** *Every accepted* MetaCICNormalization *read must expose the audit anchor A, candidate evidence K, strong-normalization readback S, Pi and application adequacy P, closed projection D, authorized generator U, compiler recognizer G, inductive acceptance L, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance Q, and local* NameCert *row N before any normalization output is read. Thus the router is scoped to the finite kernel packet carried by $A, K, S, P, D, U, G, L, H, C, Q, N$, not to a host normalization principle.*

Proof. Project the carrier of Definition 1108.1. Its displayed rows are exactly $A, K, S, P, D, U, G, L, H, C, Q, N$, and the strong-normalization readback row S is downstream of the candidate-evidence row K.

Apply route coverage. Theorem 1108.10 orders every admitted normalization output through the displayed route, while Definition 1108.14 keeps the generator and compiler recognizer rows inside the same packet.

Finally use the inductive-acceptance scope theorem. It binds the L row to the audit anchor, candidate evidence, closed projection, generator, compiler, transport, replay, provenance, and local

naming rows. Hence no normalization output can be read before the finite kernel-scope surface is displayed. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterKerne

Theorem 1108.12 (MetaCIC normalization certificate router sibling separation)**.** *The* MetaCICNormalizationCertif *carrier is not bijective, row for row, with any one of* MetaCICNormalizationFrontier$^{\uparrow}$, ClosedTermSubstitutionBoundar AuthorizedGeneratorRecursor$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, *or* KernelInductiveAcceptance$^{\uparrow}$. *A bijection preserving local NameCert obligations would have to preserve simultaneous candidate evidence, closed projection, Pi and application adequacy, authorized generator handoff, compiler recognition, kernel acceptance, and no-infinite-reduction routing, and no listed sibling carries that whole surface.*

Proof. Compare first with MetaCICNormalizationFrontier$^{\uparrow}$. The frontier sibling supplies candidate and endpoint boundary rows, but it does not also carry the closed-substitution projection, authorized generator handoff, compiler recognizer handoff, and kernel inductive acceptance rows.

Compare next with ClosedTermSubstitutionBoundary$^{\uparrow}$ and AuthorizedGeneratorRecursor$^{\uparrow}$. These siblings supply closed projection and generator authority respectively, while neither contains the candidate-evidence router with Pi and application adequacy plus no-infinite-reduction replay.

Finally compare with GroundCompilerRecognizerFlow$^{\uparrow}$ and KernelInductiveAcceptance$^{\uparrow}$. Those rows record compiler-flow and inductive-acceptance surfaces, not the simultaneous normalization evidence router. Since each sibling omits at least one of K, S, P, D, U, G, L, no row-preserving bijection can preserve the router obligations. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouter sibling separation

Theorem 1108.13 (MetaCIC normalization certificate router obligation package)**.** *For every accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier*

$$R = (A, K, S, P, D, U, G, L, H, C, Q, N),$$

the obligation package exposed by R consists exactly of the audit anchor A, candidate-evidence row K, strong-normalization readback row S, Pi and application adequacy row P, closed-term projection row D, authorized generator handoff row U, compiler recognizer row G, kernel inductive-acceptance row L, componentwise transport row H, displayed replay row C, provenance row Q, and local naming row N.

Proof. Project the carrier coordinates in the order displayed by Definition 1108.1. The rows A, K, S, P, D, U, G, L provide the audit anchor, evidence, readback, adequacy, projection, generator, compiler, and acceptance obligations.

The structural rows then close the package. Transport through H is componentwise, replay through C follows the displayed route, provenance through Q records the sibling certificates named at the start of the chapter, and N supplies the local NameCert row. Since no further coordinate is present, the listed rows are the complete obligation package. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterOblig

Theorem 1108.14 (MetaCIC normalization certificate router generator-compiler route)**.** *Every generator-to-compiler read exported by an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier factors through the authorized generator handoff U, the compiler recognizer row G, the kernel*

inductive-acceptance row L, componentwise transport H, displayed replay C, provenance Q, and local naming N. It cannot be read as an evidence-free normalization theorem, a host compiler assertion, or a route that bypasses the candidate evidence K.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterGener`

Proof. First inspect the producer rows. The authorized generator handoff is the row U, while the compiler recognizer handoff is the row G; both are visible only as coordinates of the same finite carrier that also displays K and S.

Next attach the acceptance and structural rows. The row L records kernel inductive acceptance, and H, C, Q, N preserve, replay, source, and name the same route. Thus any generator-to-compiler read remains inside the displayed packet and has no coordinate from which to extract evidence-free normalization or a detached host compiler claim. □

Theorem 1108.15 (MetaCIC normalization certificate router inductive acceptance scope)**.** *The inductive-acceptance contribution of an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier is scoped to the displayed L row together with the audit anchor A, candidate evidence K, closed projection D, generator handoff U, compiler recognizer row G, transport H, replay C, provenance Q, and local naming N. It does not close global MetaCIC normalization, quotient reduction equality, host normalization, or unconditional closed-term normalization outside that finite route.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterInduc`

Proof. Read the scope from the sibling dependencies. KernelInductiveAcceptance$^{\uparrow}$ contributes the row L, while MetaCICNormalizationFrontier$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, and GroundCompilerRecognizerFlow$^{\uparrow}$ contribute the candidate, closed-projection, generator, and compiler-facing rows named in the chapter introduction.

The router preserves those rows as one finite packet. Componentwise transport, displayed replay, provenance, and local naming do not add a global normalization coordinate, so the inductive-acceptance read remains scoped to the displayed route and refuses the excluded host and quotient claims. □

Theorem 1108.16 (MetaCIC normalization certificate router residual candidate coverage)**.** *Every residual normalization read accepted by a* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier exposes candidate, closed-candidate, or candidate-set evidence in the row K before the Pi and application adequacy row P, closed projection row D, authorized generator row U, compiler recognizer row G, and kernel acceptance row L may contribute to the displayed route. The structural rows H, C, Q, N preserve that candidate-mediated route by componentwise* hsame, Cont *replay,* Pkg *provenance, and local* NameCert *naming, but they do not introduce a residual normalization read outside K.*

Proof. Start with the obligation surface. Theorem 1108.2 lists candidate-evidence admission through K before the readback, adequacy, projection, generator, compiler, and acceptance rows, so every accepted residual read has a displayed evidence coordinate.

Apply the refusal and determinacy clauses. Theorem 1108.3 rejects closed reads with no candidate, closed-candidate, or candidate-set evidence, while Definition 1108.6 determines the strong-normalization readback from the evidence in K.

It remains to account for the route rows. Theorem 1108.10 orders the finite route through $A, K, S, P, D, U, G, L, H, C, Q, N$, and Definition 1108.15 keeps the kernel-acceptance contribution scoped to that displayed packet. Hence $P, D, U, G, L, H, C, Q, N$ can only transmit a residual read already mediated by K. □

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterResid

Theorem 1108.17 (MetaCIC normalization certificate router residual determinacy)**.** *If two accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *packets expose the same authorized-generator row U, closed-term substitution boundary D, normalization frontier rows K, S, P, and residual candidate rows carried by K, then every candidate-mediated normalization read selects the same displayed residual route up to the packet classifier. No hidden redex choice, host reduction equality, quotient reduction row, or evidence-free normalizer can separate the two reads.*

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterResid

Proof. Start with candidate determinacy. Theorem 1108.6 determines the strong-normalization readback from the candidate evidence in K, and Definition 1108.16 requires every residual read to be mediated by that displayed evidence.

Project the shared rows. Projection coverage and route coverage identify the finite route through $A, K, S, P, D, U, G, L, H, C, Q, N$, while the common D, U and normalization-frontier rows fix the closed-substitution boundary, authorized-generator handoff, and candidate frontier seen by the read.

It remains only to exclude hidden selectors. The carrier has no coordinate for an external redex choice, host reduction equality, quotient reduction, or evidence-free normalizer, and the structural rows H, C, Q, N only transport, replay, source, and name the displayed route. Hence the two packets select the same residual route up to their packet classifier. □

Theorem 1108.18 (MetaCIC normalization certificate router residual readback boundary)**.** *Every residual readback consumed from an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier must expose candidate evidence in K, closed projection through D, authorized generator evidence through U, compiler recognizer evidence through G, kernel acceptance through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, and the local* NameCert *row N before any strong-normalization output in S can be consumed. No residual readback can bypass those rows through a hidden redex selector, evidence-free normalizer, host reduction equality, or quotient reduction row.*

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterResid

Proof. Begin with the obligation surface and closed-read refusal. Theorem 1108.2 lists candidate admission through K before readback, and Definition 1108.3 rejects any closed normalization read without candidate, closed-candidate, or candidate-set evidence.

Next attach route and residual coverage. Theorem 1108.10 orders the accepted route through $A, K, S, P, D, U, G, L, H, C, Q, N$, while Definition 1108.16 requires every residual read to expose the candidate row before P, D, U, G, L contribute.

Finally use residual determinacy. Theorem 1108.17 fixes the residual route once the displayed candidate, closed projection, authorized generator, and frontier rows agree. Transport, replay, provenance, and naming preserve that route, so the strong-normalization output cannot be consumed before $K, D, U, G, L, H, C, Q, N$ have been displayed. □

Theorem 1108.19 (MetaCIC normalization certificate router scoped kernel route)**.** *Every accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *read exposes the ordered finite route*

$$A, K, S, P, D, U, G, L, H, C, Q, N$$

through candidate evidence, closed projection, authorized generator, compiler recognizer, kernel acceptance, componentwise hsame*, displayed* Cont*,* Pkg*, and local* NameCert *before any normalization output is read. Thus the router binds normalization output to the BEDC kernel-facing surface and not to an evidence-free host normalizer.*

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterScope

Proof. Begin with the carrier and route coverage. The carrier definition displays $A, K, S, P, D, U, G, L, H, C, Q, N$, and Theorem 1108.10 orders any admitted strong-normalization output through that finite route.

Next apply the kernel-scope and generator-compiler routes. Theorem 1108.11 requires candidate evidence, closed projection, authorized generation, compiler recognition, and kernel inductive acceptance before a normalization output is read, while Theorem 1108.14 keeps generator-to-compiler reads inside the same displayed packet.

Finally use the inductive acceptance scope. Theorem 1108.15 binds the L row to the audit anchor, candidate evidence, closed projection, generator, compiler, transport, replay, provenance, and local naming rows. Hence every normalization read is scoped to the BEDC kernel route carried by the accepted packet. □

Theorem 1108.20 (MetaCIC normalization certificate router candidate soundness)**.** *An accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *packet may expose candidate-mediated normalization evidence only when the audit map A, candidate row K, generator row U, compiler row G, projection row D, and inductive acceptance row L all agree on the same finite* BHist *packet. If any one of those rows is absent or names a different packet, then no normalization certificate, no-infinite-reduction certificate, or public router export is available from S.*

Proof. Begin with the obligation surface and candidate determinacy. Theorem 1108.2 lists the candidate row before readback, and Definition 1108.6 determines S from the displayed candidate evidence in K.

Next attach the projection and producer rows. Projection coverage keeps D as the closed-substitution consumer of that same evidence, while the generator-compiler route keeps U and G inside the same accepted packet.

The inductive row is then scoped by Definition 1108.15. It binds L to the audit anchor, candidate evidence, closed projection, generator, compiler, transport, replay, provenance, and local naming rows rather than to a detached host kernel claim.

Finally apply the scoped kernel route. Theorem 1108.19 permits normalization output only through the ordered finite route $A, K, S, P, D, U, G, L, H, C, Q, N$. Therefore disagreement among A, K, U, G, D, L leaves no carrier coordinate from which S can export a normalization certificate, a no-infinite-reduction certificate, or a public router output. □

Theorem 1108.21 (MetaCIC normalization certificate router public export)**.** *An accepted* MetaCICNormalizationC *packet exports exactly the public router surface already scoped by its carrier: generator recursion through U, ground-compiler recognition through G, kernel-inductive acceptance through L, closed-substitution boundary reads through D, residual-candidate determinacy through K, S, P, and non-escape through H, C, Q, N. The public interface contains no new MetaCIC primitive, evidence-free normalizer, host reduction equality, quotient reduction row, or closed-term normalization read outside*

$$A, K, S, P, D, U, G, L, H, C, Q, N.$$

Proof. Start with the carrier and obligation surface. The definition displays $A, K, S, P, D, U, G, L, H, C, Q, N$, and Theorem 1108.2 identifies these rows as the complete local certificate surface.

Next use the coverage and producer routes. Theorems 1108.10 and 1108.14 keep every admitted normalization read, generator read, and compiler-recognizer read inside the same finite router packet.

The acceptance and residual clauses supply the remaining public fields. Theorem 1108.15 scopes the L row to the displayed kernel route, while Theorems 1108.16 and 1108.17 force residual normalization reads to be mediated by the candidate rows K, S, P and the closed-substitution boundary D.

Finally apply the scoped kernel route. Theorem 1108.19 binds normalization output to the BEDC kernel-facing surface, and the carrier has no coordinate for a new MetaCIC primitive, evidence-free normalizer, host equality, quotient reduction row, or closed-term normalization read outside the displayed packet. Thus the listed rows are exactly the public export surface. □

Theorem 1108.22 (MetaCIC normalization certificate router public consumer boundary)**.** *Every downstream consumer of the public* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *export must preserve the ordered route*

$$A, K, S, P, D, U, G, L, H, C, Q, N.$$

The candidate row K must supply candidate, closed-candidate, or candidate-set evidence before the generator row U, compiler row G, and acceptance row L are consumed. The consumer boundary therefore admits only candidate-mediated normalization evidence over the displayed packet and refuses evidence-free normalizers, host reduction equality, quotient reduction rows, and unconditional closed-term normalization outside that route.

Proof. Start with route coverage and the public export. Theorem 1108.10 orders every accepted read through $A, K, S, P, D, U, G, L, H, C, Q, N$, and Definition 1108.21 identifies that ordered packet as the public router surface.

Next enforce candidate visibility. The residual candidate coverage theorem, Definition 1108.16, requires K to display candidate, closed-candidate, or candidate-set evidence before P, D, U, G, L can contribute to a residual normalization read. Residual determinacy, Definition 1108.17, then keeps the chosen residual route fixed by that displayed evidence.

The projection and producer rows do not create another source. Projection coverage, Definition 1108.9, keeps D as a closed-substitution consumer of candidate-mediated evidence, while the generator-compiler route, Definition 1108.14, keeps U, G, L inside the same packet.

Finally apply the scoped kernel route, Definition 1108.19. It binds normalization output to the BEDC kernel-facing rows and leaves no carrier coordinate for evidence-free normalizers, host reduction equality, quotient reduction rows, or unconditional closed-term normalization outside $A, K, S, P, D, U, G, L, H, C, Q, N$. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterPubli`

Theorem 1108.23 (MetaCIC normalization certificate router no-infinite-reduction public handoff)**.** *Every public* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *consumer preserving the displayed route*

$$A, K, S, P, D, U, G, L, H, C, Q, N$$

exports no-infinite-reduction evidence only through candidate evidence, closed-candidate evidence, or candidate-set evidence in K. The public handoff cannot replace those rows by an evidence-free normalizer, host reduction equality, quotient reduction row, or unconditional closed-term normalization principle.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterNoInf`

Proof. Start with route coverage and the public consumer boundary. Theorems 1108.10 and 1108.22 force every downstream read to preserve the ordered packet $A, K, S, P, D, U, G, L, H, C, Q, N$.

Next apply residual candidate coverage. Theorem 1108.16 requires candidate, closed-candidate, or candidate-set evidence in K before the adequacy, closed projection, generator, compiler, and acceptance rows may contribute to the displayed route.

The Lean-side source discipline is exactly the three no-infinite-reduction routes through candidate, closed-candidate, and candidate-set evidence. Thus a public consumer can export no-infinite-reduction only by preserving one of those evidence rows, while the carrier offers no coordinate for evidence-free normalization, host reduction equality, quotient reduction, or unconditional closed-term normalization. □

Theorem 1108.24 (MetaCIC normalization certificate router critical-pair discharge)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier. Suppose a router candidate read through R passes through the sibling rows* MetaCICNormalizationFrontier$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, *and* MetaCICConfluenceCriticalPairBound *Then the read reaches a finite discharge row for the displayed residual-substitution and confluence-critical-pair boundary before any normalization certificate is exported through S or through the public route. The discharge remains inside $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named sibling boundary rows; it does not export evidence-free strong normalization, host conversion equality, quotient reduction equality, or a hidden global confluence theorem.*

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterCriti`

Proof. Start from candidate determinacy and route coverage. Theorems 1108.6 and 1108.10 force every accepted normalization read to expose candidate evidence in K and to follow the ordered packet $A, K, S, P, D, U, G, L, H, C, Q, N$.

Next use the producer and residual rows. The generator-compiler route, Definition 1108.14, keeps U, G, L inside the same finite packet, while residual candidate coverage and residual determinacy, Definition 1108.16 and Definition 1108.17, keep the residual substitution read tied to the candidate, closed-substitution, and frontier rows already displayed.

Now apply the residual readback boundary and scoped kernel route. Theorem 1108.18 requires $K, D, U, G, L, H, C, Q, N$ before any residual readback can be consumed, and Definition 1108.19 keeps the resulting normalization output kernel-facing.

Finally invoke the public export and consumer boundary. Theorems 1108.21 and 1108.22 allow downstream reads only through the displayed public router surface. Therefore a candidate route that also names MetaCICResidualSubstitutionBoundary$^{\uparrow}$ and MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ must discharge those finite boundary rows before export, and no evidence-free strong normalization, host conversion equality, quotient reduction equality, or hidden confluence theorem has a carrier coordinate. □

Theorem 1108.25 (MetaCIC normalization certificate router frontier residual audit handoff)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *carrier. Any public consumer that exports either no-infinite-reduction evidence or a critical-pair discharge from R must first expose the audit anchor A, the candidate row K, the frontier row supplied by* MetaCICNormalizationFrontier$^{\uparrow}$, *the residual-substitution boundary supplied by* MetaCICResidualSubstitutionBoundary$^{\uparrow}$, *the generator and compiler rows U, G, and the kernel-acceptance row L. Hence frontier evidence, residual-substitution evidence, and audit-map evidence form one candidate-mediated route before export; the router exports no evidence-free strong normalization, host conversion equality, quotient reduction equality, or hidden confluence theorem.*

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterFront

Proof. Start with route coverage and the residual readback boundary. Theorem 1108.10 orders every accepted read through $A, K, S, P, D, U, G, L, H, C, Q, N$, while Definition 1108.18 requires $K, D, U, G, L, H, C, Q, N$ before a residual readback is consumed.

Next impose the public consumer boundary. Theorem 1108.22 keeps downstream reads on the same candidate-mediated public surface, and Definition 1108.23 permits no-infinite-reduction export only through candidate, closed-candidate, or candidate-set evidence in K.

Finally apply the critical-pair discharge. Theorem 1108.24 says that a route naming the frontier, residual-substitution, and confluence-critical-pair sibling rows must discharge those finite boundaries before public export. Combining these clauses forces any no-infinite-reduction or critical-pair consumer to pass through A, K, the frontier row, the residual-substitution boundary, U, G, and L before export, with no carrier coordinate for evidence-free strong normalization, host conversion equality, quotient reduction equality, or hidden confluence. □

Theorem 1108.26 (MetaCIC normalization certificate router evidence adequacy lock)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier. Any accepted residual normalization read that uses the Pi and application adequacy row P must first expose candidate, closed-candidate, or candidate-set evidence in K before any row among $P, D, U, G, L, H, C, Q, N$ contributes to residual readback. In particular, no adequacy row by itself supplies strong-normalization output, no-infinite-reduction evidence, critical-pair discharge, or kernel-facing normalization export.*

Proof. Start with the obligation surface and route coverage. Theorems 1108.2 and 1108.10 display $A, K, S, P, D, U, G, L, H, C, Q, N$ as one ordered packet, so P is not an independent adequacy oracle.

Next apply candidate determinacy and the adequacy chain. Theorem 1108.6 fixes the accepted candidate evidence in K, while Definition 1108.8 reads Pi and application adequacy only through that candidate-mediated route.

Residual coverage keeps the same boundary. The residual candidate coverage theorem forces candidate, closed-candidate, or candidate-set evidence in K before P, D, U, G, L contribute to a residual read, and Definition 1108.18 extends the same requirement across H, C, Q, N before readback.

Finally attach the public residual handoff. Theorem 1108.25 requires the audit anchor, frontier row, residual-substitution boundary, generator, compiler, and kernel-acceptance rows to remain on the same candidate-mediated route. Therefore adequacy alone cannot supply strong-normalization output or any public normalization export. □

Theorem 1108.27 (MetaCIC normalization certificate router public adequacy)**.** *Let $R = (A, K, S, P, D, U, G, L, H$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier. Suppose its public export surface, no-infinite-reduction handoff, critical-pair discharge, and frontier residual audit rows are all displayed. Then R exports exactly the candidate-mediated normalization certificate surface needed by downstream MetaCIC consumers: the route through $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named frontier, residual-substitution, and confluence-critical-pair boundary rows. The adequacy claim does not assert a full strong-normalization theorem, global confluence, unrestricted substitution, host conversion equality, or quotient reduction equality.*

Lean4 ✓ BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterPubli

Proof. Start with the public export and consumer boundary. Theorems 1108.21 and 1108.22 identify the downstream surface as the ordered route through $A, K, S, P, D, U, G, L, H, C, Q, N$.

Attach the no-infinite-reduction handoff. Theorem 1108.23 permits such evidence only through candidate, closed-candidate, or candidate-set evidence in K, so the public route remains candidate-mediated.

Next attach the critical-pair and residual audit rows. Theorem 1108.24 discharges the displayed residual-substitution and confluence-critical-pair boundaries before export, and Definition 1108.25 forces the audit anchor, frontier row, residual-substitution boundary, generator, compiler, and kernel-acceptance rows to be visible on the same route.

The Lean TasteGate site named in this chapter records the single-carrier alignment for the origin-ai router surface. Therefore the public adequacy theorem packages the displayed exports for downstream consumers while refusing full strong normalization, global confluence, unrestricted substitution, host conversion equality, and quotient reduction equality. □

Theorem 1108.28 (MetaCIC residual SN candidate-mediated route)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *packet. Every residual strong-normalization export from R must expose the audit anchor A, candidate row K, frontier row from* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*, residual-substitution boundary from* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$*, generator and compiler rows U, G, kernel-acceptance row L, and structural rows H, C, Q, N before exporting strong normalization or no-infinite-reduction evidence. The adequacy rows alone cannot supply evidence-free normalization, host conversion equality, quotient reduction equality, or an unconditional closed-term normalization principle.*

Proof. Start with the public consumer boundary. Theorem 1108.22 forces every downstream read to preserve the ordered packet $A, K, S, P, D, U, G, L, H, C, Q, N$ and to keep candidate evidence visible before generator, compiler, and acceptance rows are consumed.

Attach the no-infinite-reduction public handoff. Theorem 1108.23 permits no-infinite-reduction export only through candidate, closed-candidate, or candidate-set evidence in K.

Next use the frontier residual audit handoff. Theorem 1108.25 requires the audit anchor, frontier row, residual-substitution boundary, generator row, compiler row, and kernel-acceptance row to remain on the same candidate-mediated public route.

Finally apply public adequacy. Theorem 1108.27 packages only that displayed candidate-mediated surface for downstream consumers. Since the carrier has no independent adequacy oracle or evidence-free normalization coordinate, adequacy rows alone cannot export strong normalization, no-infinite-reduction, host conversion equality, quotient reduction equality, or unconditional closed-term normalization. □

Theorem 1108.29 (MetaCIC normalization router certificate-budget discharge)**.** *Let $R = (A, K, S, P, D, U, G, L, H$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *carrier. Every public read of R that reaches Definition 1108.27 must first discharge the finite* $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ *row together with the residual-substitution and critical-pair boundary rows named by the router. The discharge remains inside the displayed route*

$$A, K, S, P, D, U, G, L, H, C, Q, N$$

and the named frontier, residual-substitution, and confluence-critical-pair rows; it exports no evidence-free strong normalization, host conversion equality, quotient reduction equality, or hidden global confluence theorem.

Proof. Start with public export and public consumer boundary. Theorems 1108.21 and 1108.22 force every downstream read to preserve the ordered packet $A, K, S, P, D, U, G, L, H, C, Q, N$.

Next read the finite boundary obligations. The critical-pair discharge theorem requires the residual-substitution and confluence-critical-pair boundary rows to be discharged before a normalization certificate is exported, while Definition 1108.25 keeps the audit anchor, frontier row, residual-substitution boundary, generator, compiler, and kernel-acceptance rows on the same candidate-mediated route.

Now attach the evidence side. The evidence adequacy lock, Definition 1108.26, requires candidate, closed-candidate, or candidate-set evidence in K before adequacy, closed projection, generator, compiler, kernel acceptance, transport, replay, provenance, and local naming contribute to residual readback.

Finally use public adequacy. Theorem 1108.27 packages only the candidate-mediated router surface together with the displayed frontier, residual-substitution, and confluence-critical-pair rows. Therefore a public read reaching that theorem has already discharged the finite normalization-evidence budget and the named boundary rows, and no excluded global endpoint is exported. □

Theorem 1108.30 (MetaCIC normalization certificate router axiom-clean synchronization)**.** *Let* $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ *be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *packet. The router's accepted packet is exhausted by the displayed candidate-evidence row* K*, closed-projection row* D*, authorized generator row* U*, compiler recognizer row* G*, kernel-acceptance row* L*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* Q*, and local* $\mathsf{NameCert}$ *row* N*, together with the audit anchor* A*, readback row* S*, and adequacy row* P*. Hence the audit-clean paper route has exactly the finite surface required for axiom-clean formal-status synchronization. It does not assert full subject reduction, global confluence, or strong normalization as a host theorem.*

Proof. Project the carrier of Definition 1108.1. Its rows are exactly $A, K, S, P, D, U, G, L, H, C, Q, N$, with candidate evidence in K and all projection, generator, compiler, acceptance, replay, provenance, and naming data displayed as separate finite coordinates.

Apply route coverage and public consumer coverage. Theorems 1108.10 and 1108.22 force every accepted or public read to preserve the same candidate-mediated route before downstream consumers inspect normalization evidence.

Use public adequacy and certificate-budget discharge. Theorems 1108.27 and 1108.29 package only the displayed frontier, residual-substitution, confluence-critical-pair, generator, compiler, acceptance, replay, provenance, and local naming rows.

Therefore the finite router surface already identifies the rows relevant to axiom-clean synchronization: candidate evidence, closed projection, generator, compiler, acceptance, replay, provenance, and $\mathsf{NameCert}$ data. Since no carrier coordinate exports full subject reduction, global confluence, or host strong normalization, the synchronization is a finite-surface claim rather than a MetaCIC completeness theorem. □

Theorem 1108.31 (MetaCIC normalization router bridge-candidate surface)**.** *Let* $R = (A, K, S, P, D, U, G, L, H, C$ *be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *packet. Any bridge-facing consumer that uses the public adequacy surface must first read candidate evidence* K*, closed projection* D*, authorized generator* U*, compiler recognizer* G*, kernel acceptance* L*, componentwise transport* H*, displayed replay* C*, provenance* Q*, and local name* N *as visible rows of the same finite packet. The bridge-candidate surface is only this displayed route through public adequacy; it refuses subject reduction, global confluence, host normalization, and quotient reduction equality.*

Proof. Start with the carrier. Definition 1108.1 displays candidate evidence, closed projection, authorized generator, compiler recognition, kernel acceptance, transport, replay, provenance, and local naming as distinct coordinates of the same packet.

Next apply public adequacy. Definition 1108.27 exports only the candidate-mediated route through $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named frontier and residual boundary rows; it does not supply a separate bridge theorem.

Finally use the axiom-clean synchronization surface. Definition 1108.30 identifies the same finite rows as the audit-clean public surface and excludes full subject reduction, global confluence, and host strong normalization. Because quotient reduction equality is also absent from the carrier, a bridge-facing consumer can treat this theorem only as a candidate surface for a later checked bridge. □

Theorem 1108.32 (MetaCIC router bridge-ready exactness)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *packet. Every bridge-facing read that treats the router as bridge-ready must preserve the public adequacy target, the axiom-clean synchronization surface, and the TasteGate row before it uses the bridge-candidate route. The exact bridge-ready surface is therefore the same finite route through $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named frontier, residual-substitution, confluence-critical-pair, and TasteGate rows. It still refuses evidence-free strong normalization, unconditional application normalization, host normalization, quotient reduction equality, and bridgeChecked status.*

Proof. Start with public adequacy. Definition 1108.27 exports only the candidate-mediated route through $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named frontier and residual boundary rows.

Next preserve the axiom-clean synchronization surface. Definition 1108.30 identifies the same finite rows as the audit-clean public surface and refuses full subject reduction, global confluence, and host strong normalization.

Now attach the bridge-candidate surface and TasteGate row. Definition 1108.31 keeps bridge-facing reads behind the displayed router rows, while the existing Lean TasteGate marker records the single-carrier alignment for the same packet. Since neither row is a checked standard bridge, the read is bridge-ready only as this exact finite surface and not as bridgeChecked evidence. □

Theorem 1108.33 (MetaCIC router residual candidate coverage)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$ *packet. Any residual candidate read that reaches a bridge-ready or closed-normalization-facing consumer must pass through candidate evidence in K, closed projection D, Pi and application adequacy P, authorized generator U, compiler recognizer G, kernel acceptance L, componentwise transport H, displayed replay C, provenance Q, and local naming N. The read is therefore the same candidate-mediated finite route already exposed by the router; it is not evidence-free normalization, host conversion equality, quotient reduction equality, or a hidden closed-term normalizer.*

Proof. Start with the existing residual candidate coverage theorem. Definition 1108.16 requires each residual normalization read to expose candidate, closed-candidate, or candidate-set evidence in K before the adequacy, projection, generator, compiler, acceptance, and structural rows contribute.

Next use the evidence adequacy lock and public adequacy surface. Theorems 1108.26 and 1108.27 keep $P, D, U, G, L, H, C, Q, N$ downstream of the same candidate-mediated packet.

Finally apply bridge-ready exactness. Definition 1108.32 permits a bridge-ready read only as the displayed route through $A, K, S, P, D, U, G, L, H, C, Q, N$ together with the named frontier and residual rows. Hence any closed-normalization-facing consumer sees only the listed finite coverage route and none of the excluded host endpoints. □

Theorem 1108.34 (MetaCIC router frontier audit handoff)**.** *Let $R = (A, K, S, P, D, U, G, L, H, C, Q, N)$ be an accepted* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *packet. A closed-normalization consumer may read the frontier residual audit handoff only after the route has passed through the* MetaCICNormalizationFrontier *row, the candidate evidence row K, the residual-substitution boundary, generator and compiler rows U, G, kernel acceptance L, transport H, replay C, provenance Q, and local naming N. Thus the handoff is routed through the frontier row before any closed-normalization consumer receives it, and it exports no evidence-free strong normalization, host conversion equality, quotient reduction equality, or hidden confluence theorem.*

Proof. Read the verified frontier residual audit handoff first. Definition 1108.25 forces public consumers that export no-infinite-reduction evidence or critical-pair discharge to expose A, K, the frontier row, the residual-substitution boundary, U, G, and L before export.

Now attach residual candidate coverage. Definition 1108.33 keeps the candidate evidence, closed projection, adequacy, generator, compiler, acceptance, transport, replay, provenance, and naming rows on the same route.

Finally use bridge-ready exactness. A closed-normalization consumer can only read the bridge-ready surface after the frontier, residual-substitution, confluence-critical-pair, and TasteGate rows have been named with the router packet. Therefore the handoff passes through MetaCICNormalizationFrontier$^{\uparrow}$ before the consumer receives it, and no excluded host or quotient endpoint is exported. □

Falsifiable prediction: A closed-term normalization read is refused unless candidate, closed-candidate, or candidate-set evidence is present on the displayed route.

Independence witness: metacicnormalizationfrontier, closedtermsubstitutionboundary, authorizedgeneratorrecursor, groundcompilerrecognizerflow, kernelinductiveacceptance: the router has simultaneous candidate-evidence, closed projection, Pi/app adequacy, and no-infinite-reduction rows not bijective with any one sibling.

The Lean TasteGate site for MetaCICNormalizationCertificateRouter$^{\uparrow}$ must provide explicit Nontrivial MetaCICNo and FieldFaithful MetaCICNormalizationCertificateRouter$^{\uparrow}$ instances. These instances preserve candidate evidence, closed projection, Pi and application adequacy, authorized generator handoff, compiler recognition, kernel acceptance, route replay, provenance, and local naming as distinguishable fields.

Closure status for NameCert$_{\text{MetaCICNormalizationCertificateRouter}^{\uparrow}}$.

Constructive story: A MetaCICNormalizationCertificateRouter$^{\uparrow}$ packet is generated from an audit-map anchor, candidate evidence, strong-normalization readback, Pi and application adequacy, closed-term projection, authorized generator handoff, compiler recognition, kernel inductive acceptance, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationCertificateRouter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1108.1, Definition 1108.2, Definition 1108.3, Definition 1108.12, Definition 1108.13, Definition 1108.14, Definition 1108.15, and Definition 1108.19 bind candidate-mediated normalization reads as a finite kernel-scoped router surface.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.taste gate`

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalizationCertificateRouter}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MetaCICNormalizationCertificateRouterUp.MetaCICNormalizationCertificateRouterPublicAdeq`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close evidence-free strong normalization, unconditional application normalization, host normalization, quotient reduction equality, or bridge checking.

Upgrade path: No theory-axis upgrade remains; bridge checking is a separate formal-axis target.

Chapter 1109

A Concrete Naming Certificate for $\mathsf{FocusedMultiHistLedger}^{\uparrow}$

The $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ packet realizes the finite closure target of chapter 9420 as a concrete NameCert carrier. It reads the current-state surface of chapter 578, the local frame surface of chapter 609, the no-subject trace surface of chapter 640, the two-history locality ledger of chapter 759, and the sealed trace boundary of chapter 1054. The packet adds no universe object, subject carrier, global clock, host equality of histories, or selected synchronization service.

Definition 1109.1 (Focused multi-Hist ledger carrier)**.** *A* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, F, K, T, G, R, I, D, O, H, C, P, N)$$

with the following displayed rows:

1. *S is an accepted* $\mathsf{ConsciousObserverState}^{\uparrow}$ *row naming the focused current-state surface;*
2. *F is an accepted* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *row naming the local coordinate frame available at that state;*
3. *K is an accepted* $\mathsf{ObserverKernelTrace}^{\uparrow}$ *row exposing the focused observer trace without a subject carrier;*
4. *T is an accepted* $\mathsf{ObserverTraceSeal}^{\uparrow}$ *row sealing the focused trace and its no-subject refusal ledger;*
5. *G is the* GAP *and provenance ledger for every compressed, digested, or not-yet-fulfilled row in the finite packet;*
6. *R is the finite inter-*Hist *relation ledger, each relation being read through an* $\mathsf{InterHistLocalityLedger}^{\uparrow}$ *row;*
7. *I is the invariant-readback row for relation-stable records visible to the focused ledger;*
8. *D is the finite dependence and route row for local time slots, selector rows, and optional maximum-rate reads;*
9. *O is the audit row asserting no subject parameter, global clock, unledgered chooser, object-level* T *field, quotient history equality, or hidden choice-like service;*
10. *H records componentwise* hsame *transport for the displayed rows;*

11. *C records the displayed* Cont *replay routes by which each row is consumed from the ledger;*
12. *P is the* Pkg *provenance over* ConsciousObserverState$^{\uparrow}$, ObserverCoordinateFrame$^{\uparrow}$, ObserverKernelTrace$^{\uparrow}$, ObserverTraceSeal$^{\uparrow}$, InterHistLocalityLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* NameCert$_{\text{FocusedMultiHistLedger}^{\uparrow}}$ *row.*

The carrier is the records-side realization of the closed focused multi-Hist ledger of Definition 9420.1; it is not a carrier for a global universe, observer-substance, host spacetime, or external synchronization substrate.

Lean4 ✓ `BEDC.Derived.FocusedMultiHistLedgerUp.FocusedMultiHistLedgerUp single carrier alignment`

Theorem 1109.2 (Focused multi-Hist ledger NameCert obligations)**.** *For the carrier of Definition 1109.1, the local* NameCert$_{\text{FocusedMultiHistLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $L = (S, F, K, T, G, R, I, D, O, H, C, P, N)$, focused current-state admission through S, local frame admission through F, focused trace and trace-seal admission through K, T,* GAP *and provenance accountability through G, inter-*Hist *relation admission through R, invariant and dependence readback through I, D, audit refusal through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 1109.1. The rows S, F, K, T expose the focused state, local frame, trace row, and sealed trace before the ledger reads any relation or invariant data.

The rows G, R, I, D supply the closed records-side body of the packet. The GAP ledger accounts for compressed or not-yet-fulfilled rows, the relation row consumes InterHistLocalityLedger$^{\uparrow}$, and the invariant and dependence rows record exactly the finite readback available to a focused multi-history consumer.

The audit and structural rows are O, H, C, P, N. They record the refusal of subject parameters, global clocks, hidden choosers, object-level T fields, quotient history equality, and choice-like services, while transport, replay, provenance, and local naming keep every public read inside the same displayed packet. □

Theorem 1109.3 (Focused multi-Hist ledger non-escape)**.** *Every public consumer of an accepted* FocusedMultiHistLedger$^{\uparrow}$ *carrier can read only the displayed focused state, local frame, kernel trace, trace seal,* GAP *ledger, inter-*Hist *locality ledger, invariant row, dependence row, audit row, componentwise* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *row*

$$S, F, K, T, G, R, I, D, O, H, C, P, N.$$

It receives no observer-substance, global observer frame, global clock, host equality of histories, quotient observer identity, unledgered compression, object-level T *field, selected limit, or choice-like synchronization service.*

Lean4 ✓ `BEDC.Derived.FocusedMultiHistLedgerUp.FocusedMultiHistLedgerCarrier nonescape`

Proof. Start with the NameCert obligations. They force every accepted read through the displayed tuple and identify $S, F, K, T, G, R, I, D, O, H, C, P, N$ as the whole local certificate surface.

The sibling rows do not widen the carrier. ConsciousObserverState$^{\uparrow}$ supplies a focused current-state surface, ObserverCoordinateFrame$^{\uparrow}$ supplies a local frame rather than a global frame, ObserverKernelTrace$^{\uparrow}$ and ObserverTraceSeal$^{\uparrow}$ supply no-subject trace exposure, and InterHistLocalityLedger$^{\uparrow}$ supplies finite two-history relation routing.

The excluded objects have no coordinate in L. A consumer requesting observer-substance, global synchronization, host equality, quotient identity, unledgered compression, an object-level T field, selected limits, or choice-like services would have to leave the displayed packet, so the accepted carrier cannot export such data. □

Theorem 1109.4 (Focused multi-Hist ledger public export)**.** *Every public consumer of an accepted* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ *carrier receives the export packet*

$$S, F, K, T, G, R, I, D, O, H, C, P, N$$

and no further row. The export surface keeps the focused current-state row, local frame row, no-subject kernel trace, trace seal, GAP *and provenance ledger, inter-*Hist *locality ledger, invariant readback, dependence route, audit refusal, componentwise transport, replay, package provenance, and local naming row visible to the public route.*

It does not export observer-substance, a global universe object, a global clock, host equality of histories, quotient observer identity, selected synchronization, unledgered compression, or an object-level T *field.*

Proof. Start with the non-escape theorem Definition 1109.3. It already identifies the displayed rows as the whole readable surface of an accepted carrier.

The public export is therefore a repackaging of those rows, not a new carrier. The focused state S, local frame F, trace rows K, T, ledger rows G, R, I, D, O, and structural rows H, C, P, N are exported together so that each public read remains tied to the same TasteGate-certified carrier surface.

The excluded objects have no coordinate in the carrier and cannot be recovered from the export packet. Hence the public route exposes exactly the focused multi-Hist ledger rows and nothing outside them. □

Falsifiable prediction: Removing either the focus row or the GAP/provenance ledger leaves a multi-Hist packet that cannot certify the closed focused ledger.

Independence witness: consciousobserverstate, observercoordinateframe, observerkerneltrace, interhistlocalityledger, and observertraceseal: none is bijective to the focused multi-Hist ledger because the ledger alone carries the focused state row, local frame row, no-subject kernel trace row, sealed trace row, GAP/provenance ledger, inter-Hist locality ledger, invariant readback, dependence route, audit refusal, transport, replay, provenance, and local NameCert rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{FocusedMultiHistLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ packet is generated from focused current-state, local frame, kernel trace, trace-seal, GAP/provenance, inter-Hist locality, invariant, dependence, audit, transport, replay, package, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FocusedMultiHistLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1109.1, Definition 1109.2, Definition 1109.3, Definition 1109.4, and

Lean4 ✓ `BEDC.Derived.FocusedMultiHistLedgerUp.taste gate`

bridge the public export to the acceptance-schema route while preserving the focused state, local frame, trace seal, GAP/provenance, inter-Hist locality, invariant readback, dependence, audit, transport, replay, package, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FocusedMultiHistLedger}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FocusedMultiHistLedgerUp.FocusedMultiHistLedgerUp\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a global universe object, observer-substance, host spacetime, quotient history equality, selected synchronization, or an object-level T field.

Upgrade path: No further paper-side theory-closure upgrade is pending; machine bridge checking remains the formal-axis successor for the same public export and acceptance-schema route.

Chapter 1110

A Concrete Naming Certificate for $\mathsf{CauchyWitnessGluing}^{\uparrow}$

The $\mathsf{CauchyWitnessGluing}^{\uparrow}$ packet realizes the Real-completeness witness route of chapter 9238 as one finite NameCert surface. It sits after the witness ledger of chapter 902, the tail-witness envelope of chapter 898, the seal-budget synchronizer of chapter 886, and the completion-seal classifier of chapter 900. Those sibling packets expose the witness, tail, synchronizer, and completion-seal rows separately; this packet glues them into the single consumer surface through which a Real-completion reader may reach $\mathsf{Real}^{\uparrow}$.

Definition 1110.1 (CauchyWitnessGluing carrier)**.** *A* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, T, S, K, U, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row recording the diagonal budget, observation-budget, seal-synchronizer, and completion-seal witness ledger;*
2. *T is the* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *row recording the finite tail envelope and its Real-facing tail agreement route;*
3. *S is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row aligning the selector budget, observation budget, shared threshold, and seal request;*
4. *K is the* $\mathsf{RealCompletionSealClassifierUp}$ *row classifying the completion-seal read visible to the Real consumer;*
5. *U is the finite* $\mathsf{StreamName}^{\uparrow}$ *window-family row exposed by the glued route;*
6. *V is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence handoff row reached after the stream windows are read;*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance ledger consumed by the stream and regular-sequence rows;*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row served to the completion-facing consumer;*
9. *E is the Real-completion witness edge stating that L, T, S, K, U, V, D, R are read as one gluing packet;*
10. *H records componentwise* hsame *transports for every displayed row;*

11. *C records the displayed* Cont *routes from the ledger to the tail envelope, from the synchronizer to the completion-seal classifier, from stream and dyadic reads to the regular handoff, and from the regular handoff to* $\mathsf{Real}^{\uparrow}$*;*
12. *P is the* Pkg *provenance over chapter 902, chapter 898, chapter 886, chapter 900, chapter 9238,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessGluing}^{\uparrow}}$ *row.*

The carrier exports no quotient stream equality, host real equality, selected external limit, countable-choice row, detached Cauchy witness, or ambient completeness principle beyond the displayed packet.

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingTasteGate single carrier alignment`

Theorem 1110.2 (CauchyWitnessGluing budget compatibility)**.** *Let*

$$G = (L, T, S, K, U, V, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *packet. Any Real-completion consumer that preserves the witness ledger L, tail envelope T, seal synchronizer S, completion-seal classifier K, stream window row U, regular handoff V, dyadic ledger D, Real seal R, witness edge E, transports H, continuations C, provenance P, and local name N preserves the same finite budget route. Replacing the synchronizer row while keeping the ledger and envelope fixed is not a refinement of this packet; it is a different candidate for acceptance.*

Proof. Begin with the witness ledger. The row L records which diagonal, observation-budget, synchronizer, and completion-seal witnesses are present before the Real-facing consumer reads the packet.

Next attach the tail and seal rows. The envelope T supplies the finite tail request seen by the Real route, while S aligns that request with the selected observation budget and shared threshold. The classifier K then reads the completion-seal side of the same synchronized route.

The lower observation rows cannot be varied independently. The stream windows U, dyadic ledger D, and regular handoff V are routed together by the displayed Cont row, and their componentwise hsame transports are recorded in H before the terminal $\mathsf{Real}^{\uparrow}$ seal R is served.

Thus a consumer preserving all displayed rows preserves the budget route as a whole. Changing S changes the coordinate that binds the ledger to the completion-seal classifier, so the resulting data may form another accepted packet, but it is not the same $\mathsf{CauchyWitnessGluing}^{\uparrow}$ carrier read. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier budget compatibility`

Theorem 1110.3 (CauchyWitnessGluing route obligation surface)**.** *An accepted* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *packet exposes exactly the route rows*

$$L, T, S, K, U, V, D, R, E, H, C, P, N$$

as its NameCert *obligation surface. The packet has no quotient-stream row, host-real row, selected-limit row, countable-choice row, or ambient-completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier route obligation surface`

Proof. Read the displayed carrier of Definition 1110.1. Its finite BHist data are precisely the thirteen rows $L, T, S, K, U, V, D, R, E, H, C, P, N$, with P recording the sibling provenance and N recording the local $\mathsf{NameCert}_{\mathsf{CauchyWitnessGluing}^{\uparrow}}$ row.

The exclusion clause at the end of the carrier definition removes every off-route object named in the statement. Since the consumer is accepted only as a read of that displayed packet, there is no remaining coordinate from which a quotient stream, host real, selected external limit, countable-choice principle, or ambient-completeness principle can be recovered. □

Theorem 1110.4 (CauchyWitnessGluing witness ledger exactness)**.** *In an accepted* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *packet, the witness ledger row L is exact only together with the tail envelope T, seal synchronizer S, completion classifier K, and witness edge E. The ledger read is therefore the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *certificate of chapter 902 synchronized through the seal-budget route of chapter 886, not a detached witness row.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier witness ledger exactness`

Proof. The row L supplies the diagonal, observation-budget, synchronizer, and completion-seal witness ledger, but the gluing carrier does not expose L as an independent terminal object. Its exactness is tested inside the same packet that contains T, S, K, and E.

The tail envelope T determines the Real-facing tail request, while S aligns that request with the shared budget and seal request. The classifier K reads the completion-seal side of the synchronized route, and the edge E states that these rows are consumed as one gluing packet.

Thus a consumer that keeps L while dropping any of T, S, K, or E no longer reads the displayed $\mathsf{CauchyWitnessGluing}^{\uparrow}$ surface. It reads a detached ledger candidate, which is outside the accepted packet described here. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier witness ledger semantic name certificate`

Theorem 1110.5 (CauchyWitnessGluing tail envelope handoff)**.** *The handoff from the tail envelope of an accepted* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *packet to the terminal Real seal factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *as listed in the carrier provenance. Equivalently, the route from T to R is mediated by the finite stream-window row U, the dyadic tolerance ledger D, and the regular-sequence handoff V before the* $\mathsf{Real}^{\uparrow}$ *seal is served.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier tail envelope handoff`

Proof. Start with the carrier rows named in Definition 1110.1. The tail row T is not adjacent to R by an ambient completion map; the displayed route records the stream window row U, dyadic ledger D, and regular handoff V between them.

The provenance row P names the same sibling certificates and primitive rows: $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The continuation row C then records the displayed path from stream and dyadic reads to the regular handoff and from the regular handoff to the terminal Real seal.

Therefore any tail-envelope handoff to R factors through those finite BEDC rows. A direct appeal to Real completeness or a hidden quotient-stream map would bypass U, D, V, and so would not be a read of this gluing packet. □

Theorem 1110.6 (CauchyWitnessGluing completion seal scope)**.** *Every completion-seal consumer of an accepted* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *packet reads the gluing data only through* BHist, hsame, Cont,

Pkg, NameCert, *and the sibling certificates named in the carrier provenance. No off-table completion principle contributes an additional primitive to the completion-seal read.*

Proof. The accepted packet is a finite BHist carrier. Componentwise equality transport is recorded by H through hsame, route transport is recorded by C through Cont, and the provenance and naming surfaces are recorded by P and N through Pkg and NameCert.

The sibling certificates listed in P supply the ledger, tail, seal-budget, completion-seal, stream, regular-sequence, dyadic, and Real rows. These are the only primitive sources available to the completion-seal consumer.

Consequently, a consumer that invokes a completion principle not present in BHist, hsame, Cont, Pkg, NameCert, or the listed sibling certificates is no longer reading the accepted CauchyWitnessGluing$^{\uparrow}$ packet. It has introduced an off-table primitive outside the chapter's scoped surface. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier completion seal scope`

Theorem 1110.7 (CauchyWitnessGluing Real seal nonescape)**.** *Every* Real$^{\uparrow}$ *or completion-facing read of an accepted* CauchyWitnessGluing$^{\uparrow}$ *packet factors through*

$$L \to T \to S \to K \to U \to D \to V \to R$$

with the witness edge E and the structural rows H, C, P, N visible. No consumer can reach the Real seal through an omitted ledger, hidden tail envelope, alternate seal synchronizer, detached completion-seal classifier, quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier real seal nonescape`

Proof. Project the carrier of Definition 1110.1. The rows $L, T, S, K, U, V, D, R, E, H, C, P, N$ are the whole finite packet, and the definition excludes the ambient Real-completion objects named in the statement.

The route first exposes the witness and tail side. A Real-facing read has to pass through the witness ledger L and tail envelope T before it can inspect the synchronized seal request.

The route then exposes the seal and observation side. The synchronizer S and completion-seal classifier K determine which completion-seal row is being read, while U, D, V supply the finite StreamName, DyadicRatCore, and RegSeqRat handoff before R is served.

Finally, the structural rows repack the result. Componentwise hsame transports, displayed Cont routes, Pkg provenance, and the local NameCert row leave no coordinate from which a consumer could extract a second Real seal or a hidden completion principle. □

Theorem 1110.8 (CauchyWitnessGluing obligation ledger)**.** *Every accepted* CauchyWitnessGluing$^{\uparrow}$ *packet exposes exactly five obligation families: witness-ledger admission through L, T, S, K, E; stream, dyadic, and* RegSeqRat$^{\uparrow}$ *handoff through U, D, V, R; componentwise* hsame *transport through H; displayed* Cont *replay through C; and* Pkg *plus local* NameCert *provenance through P, N. These families exhaust the obligation ledger and export no quotient stream equality, selected limit, countable-choice row, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier obligation ledger`

Proof. Start with the carrier definition and route-obligation surface. The carrier of Definition 1110.1 lists exactly $L, T, S, K, U, V, D, R, E, H, C, P, N$, and Definition 1110.3 says that these rows are the local NameCert surface.

The first family is the witness-ledger admission. By Definition 1110.4, the ledger L is exact only together with the tail envelope T, seal synchronizer S, completion classifier K, and witness edge E.

The second family is the finite handoff to the Real seal. Apply Definition 1110.5 and Definition 1110.7: the stream windows U, dyadic ledger D, regular readback V, and terminal seal R mediate every Real-facing read.

The remaining three families are structural. The row H carries componentwise hsame transport, the row C carries displayed Cont replay, and the rows P, N record package provenance and local naming. Since none of the five families contains a quotient stream, selected limit, countable-choice schedule, host real equality row, or ambient Real-completeness coordinate, the obligation ledger is exhausted by the listed rows. □

The Lean TasteGate development is expected to place the explicit Nontrivial CauchyWitnessGluing$^{\uparrow}$ and FieldFaithful CauchyWitnessGluing$^{\uparrow}$ instances in BEDC.Derived.CauchyWitnessGluingUp.TasteGate, with the single-carrier alignment target named BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingTasteGa

Theorem 1110.9 (CauchyWitnessGluing scoped primitive binding)**.** *Every accepted* CauchyWitnessGluing$^{\uparrow}$ *packet reads its witness ledger L, tail envelope T, seal synchronizer S, completion classifier K,* StreamName$^{\uparrow}$ *row U,* RegSeqRat$^{\uparrow}$ *handoff V,* DyadicRatCore$^{\uparrow}$ *ledger D,* Real$^{\uparrow}$ *seal R, witness edge E, componentwise* hsame *transports H, displayed* Cont *replays C,* Pkg *provenance P, and local* NameCert$_{\text{CauchyWitnessGluing}^{\uparrow}}$ *row N only through the displayed BEDC primitives and the sibling scope named in the carrier provenance. The scoped read exports no quotient stream equality, selected limit, countable-choice row, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier scoped primitive binding`

Proof. Start with the route-obligation surface. By Definition 1110.3, the local NameCert surface of an accepted packet is exactly the displayed tuple $L, T, S, K, U, V, D, R, E, H, C, P, N$, so there is no additional coordinate from which a consumer can read a hidden primitive.

Next bind the completion-facing side. Apply Definition 1110.6: completion-seal reads pass only through BHist, hsame, Cont, Pkg, NameCert, and the sibling certificates listed by the provenance row. Apply Definition 1110.7 to force every Real-facing read through the ledger, tail, synchronizer, classifier, stream, dyadic, regular, and Real rows before the terminal seal is visible.

It remains to account for the ledger boundary. By Definition 1110.8, the five obligation families are witness-ledger admission, finite handoff to Real$^{\uparrow}$, componentwise transport, displayed replay, and package plus local naming. These families include the hsame transports, Cont replays, Pkg provenance, and local NameCert row, and they exhaust the scoped primitive surface.

Combining these rows gives the claimed primitive binding. Since the cited theorems also exclude quotient streams, selected limits, countable-choice schedules, host real equality, and ambient Real-completeness coordinates, the accepted packet remains inside the displayed BEDC primitive and sibling scope. □

Theorem 1110.10 (CauchyWitnessGluing scoped route tightening)**.** *Any completion-facing read of an accepted* CauchyWitnessGluing$^{\uparrow}$ *packet must factor through the witness ledger L, tail envelope T, seal synchronizer S, completion-seal classifier K,* StreamName$^{\uparrow}$ *windows U,* DyadicRatCore$^{\uparrow}$ *ledger D,* RegSeqRat$^{\uparrow}$ *handoff V, terminal* Real$^{\uparrow}$ *seal R, witness edge E, componentwise* hsame *transports H, displayed* Cont *replays C,* Pkg *provenance P, and local* NameCert$_{\text{CauchyWitnessGluing}^{\uparrow}}$ *row N before a Real-completion consumer may inspect the route. The read exports no detached ledger, hidden tail envelope, alternate synchronizer, quotient stream equality, selected external limit, countable-choice row, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessGluingUp.CauchyWitnessGluingCarrier scoped route tightening`

Proof. Start with the route-obligation surface of Definition 1110.3. It identifies $L, T, S, K, U, V, D, R, E, H, C, P, N$ as the whole local NameCert surface of an accepted packet.

Bind the witness side next. Apply witness-ledger exactness, tail-envelope handoff, and completion-seal scope from Definition 1110.4, Definition 1110.5, and Definition 1110.6. These results force the ledger, tail, synchronizer, classifier, and witness edge to be read as one synchronized route before the completion-seal consumer reaches the lower observation rows.

Now bind the Real-facing side. Apply Definition 1110.7 to require the StreamName windows, DyadicRatCore ledger, RegSeqRat handoff, and terminal Real seal before any completion-facing consumer reaches R.

Finally apply the obligation ledger and scoped primitive binding of Definition 1110.8 and Definition 1110.9. The structural rows H, C, P, N transport, replay, source, and name exactly the displayed route, so the excluded ledger, quotient, choice, equality, and completeness data remain outside the scoped handoff. □

Falsifiable prediction: If the gluing packet is accepted, a Real-completion consumer that preserves the witness ledger and tail envelope but changes the seal-synchronizer row is rejected by the classifier.

Independence witness: cauchywitnessledger, cauchytailwitnessenvelope, cauchysealbudgetsynchronizer, realcompletionsealclassifier, regularcauchydiagonalmeet: none is bijective because each sibling omits at least one of the paired witness, tail, synchronizer, completion-seal, and RealUp rows required by this gluing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWitnessGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyWitnessGluing}^{\uparrow}$ packet is generated from a witness ledger, tail-envelope row, seal-synchronizer row, completion-seal classifier, StreamName, RegSeqRat, DyadicRatCore, RealUp, witness-edge, hsame, Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWitnessGluing}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1110.1, Definition 1110.2, Definition 1110.3, Definition 1110.4, Definition 1110.5, Definition 1110.6, Definition 1110.7, Definition 1110.8, Definition 1110.9, Definition 1110.10, the falsifiable prediction, and the independence witness bind the displayed witness-ledger, tail-envelope, seal-synchronizer, completion-seal classifier, StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows into one scoped Real-completion witness packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWitnessGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient completeness, host real equality, Lean verification, or bridge checking for $\mathsf{CauchyWitnessGluing}^{\uparrow}$.

Upgrade path: The next paper step is a public Real-completion witness export, while Lean audit-token synchronization for the witness ledger, tail envelope, seal synchronizer, completion-seal classifier, and Real-facing handoff remains separate.

Chapter 1111

A Concrete Naming Certificate for $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$

The $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$ packet is the BEDC-native endofunctor surface on the unary-continuation category. In the standard category-theoretic account, as in Mac Lane's treatment of functors, an endofunctor preserves objects, arrows, identities, and composition inside one category. Here the one category is the unary-continuation reading of BHist from chapter 9363, so the carrier is finite packet data rather than a host functor object.

The packet consumes the sibling categorical surfaces chapter 480, chapter 484, chapter 62, and chapter 63. Its object and hom rows are unary BHist rows, its preservation rows are displayed Cont ledgers, and its transport, provenance, and local naming rows are the same hsame/Pkg/NameCert surface used by the unary-continuation category.

Definition 1111.1 (Unary-continuation endofunctor carrier)**.** *A* $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$ *carrier is a finite packet*

$$E = (O, H, F, I, M, T, C, P, N)$$

with the following displayed rows:

1. *O is the unary object row, accepting only unary* BHist *object histories from the unary-continuation category.*
2. *H is the displayed hom row, accepting only* $\mathsf{CategoryHomCarrier}(a, b, f)$ *rows between unary object histories.*
3. *F records the object and hom action rows of the endofunctor, sending unary objects to unary objects and displayed hom histories to displayed hom histories over the transported endpoints.*
4. *I records identity preservation: the image of an empty-tail identity row is classified with the empty-tail identity row at the image object.*
5. *M records composition preservation: the image of a displayed composite hom row is classified with the displayed composite of the two image hom rows.*
6. *T records componentwise* hsame *transport for object rows, hom rows, identity-preservation rows, and composition-preservation rows.*
7. *C stores the displayed* Cont *routes used to read the object action, hom action, identity preservation, and composition preservation.*
8. *P is the* Pkg *provenance row over* $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{FiniteKernelCategory}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, *and* $\mathsf{NatTrans}^{\uparrow}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UnaryContinuationEndofunctor}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host functor category, quotient morphism class, external naturality object, or equality outside the displayed BHist *rows.*

Lean4 ✓ `BEDC.Derived.UnaryContinuationEndofunctorUp.UnaryContinuationEndofunctorTasteGate single carrier alignment`

Theorem 1111.2 (Unary-continuation endofunctor NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{UnaryContinuationEndo}}$ *obligation surface for the carrier of Definition 1111.1 consists of five rows:*

1. *carrier admission by the finite packet $E = (O, H, F, I, M, T, C, P, N)$;*
2. *object and hom classifier transport by componentwise* hsame*;*
3. *identity preservation through the empty-tail identity rows of the unary-continuation category;*
4. *composition preservation through the displayed* Cont *composition rows of the unary-continuation category;*
5. *non-escape from host functor categories, quotient morphisms, external natural transformation objects, and host equality of arrows.*

Lean4 ✓ `BEDC.Derived.UnaryContinuationEndofunctorUp.UnaryContinuationEndofunctor namecert obligations`

Proof. Carrier admission is a projection of Definition 1111.1. The object, hom, action, identity, composition, transport, route, provenance, and local naming rows are all finite BHist data.

For classifier transport, use the sibling $\mathsf{KernelCategory}^{\uparrow}$ and $\mathsf{FiniteKernelCategory}^{\uparrow}$ packets. They identify the public object and hom surfaces as unary histories, displayed CategoryHomCarrier rows, and componentwise hsame transports.

Identity and composition preservation are the $\mathsf{Functor}^{\uparrow}$ rows restricted to this single unary-continuation category. The identity row I reads empty-tail continuation ledgers, while the composition row M reads displayed Cont composition ledgers before repacking the result as a displayed hom row.

The $\mathsf{NatTrans}^{\uparrow}$ sibling is consumed only as a downstream compatibility surface: it can compare components built from the same endofunctor packet, but it does not add a host naturality object. Since E has no host functor category, quotient morphism, or host arrow equality coordinate, every public read remains inside the displayed rows. □

Theorem 1111.3 (Unary-continuation endofunctor identity stability)**.** *Let $E = (O, H, F, I, M, T, C, P, N)$ be an accepted* $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$ *carrier. If $e_a : a \to a$ is the empty-tail identity hom row of the unary-continuation category, then the image of e_a under the object and hom action row F is classified with the empty-tail identity row at the image object $F(a)$. The comparison is routed only through H, I, T, C, P, N and exports no quotient morphism, host identity arrow, host arrow equality, or external functor law.*

Lean4 ✓ `BEDC.Derived.UnaryContinuationEndofunctorUp.UnaryContinuationEndofunctorCarrier identity stability`

Proof. Project the accepted carrier to its displayed hom row H, action row F, identity-preservation row I, transport row T, route row C, provenance row P, and local naming row N.

The row H reads e_a as the empty-tail identity $\mathsf{CategoryHomCarrier}(a, a, e_a)$ from the unary-continuation category surface. The action row F transports the object endpoint to $F(a)$ and the hom row to a displayed hom row over that transported endpoint.

Now apply the identity-preservation row I. It classifies the image hom row with the empty-tail identity row at $F(a)$, using the same unary-continuation identity ledger read by the sibling $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, and $\mathsf{NatTrans}^{\uparrow}$ surfaces referenced above.

Finally repack the comparison through T, C, P, N. Since the packet contains no quotient morphism coordinate, host identity-arrow coordinate, host arrow equality, or external functor-law object, the identity-stability statement remains a finite BEDC route through the named carrier rows. □

Theorem 1111.4 (Unary-continuation endofunctor composition stability)**.** *Let $E = (O, H, F, I, M, T, C, P, N)$ be an accepted* $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$ *carrier. If $f : a \to b$ and $g : b \to c$ are displayed unary-continuation hom rows and fg is their displayed* Cont *composite, then the image of fg under F is classified with the displayed* Cont *composite of the image of f and the image of g. The comparison is routed only through H, M, T, C, P, N and exports no quotient morphism, host function composition, host arrow equality, or external functor law.*

Lean4 ✓ `BEDC.Derived.UnaryContinuationEndofunctorUp.UnaryContinuationEndofunctorCarrier composition stability`

Proof. Start with the hom rows. The carrier reads f, g, and fg through $\mathsf{CategoryHomCarrier}$ over unary endpoints, and the route C records the displayed Cont composition producing fg.

Next apply the endofunctor action row F. It sends the three displayed hom rows to displayed hom rows over the transported endpoints, and T supplies the componentwise hsame comparisons needed to keep endpoints and hom tails aligned.

Now use the composition preservation row M. It states that the image of the displayed composite is classified with the displayed composite of the two image hom rows, all inside the same unary-continuation category surface.

Finally repack the comparison with P and N. Because the packet contains no quotient morphism class, host function-composition coordinate, host arrow equality, or external functor-law object, the stability claim is exactly the finite BEDC route through H, M, T, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{UnaryContinuationEndofunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{UnaryContinuationEndofunctor}^{\uparrow}$ packet is generated from unary object histories, displayed CategoryHomCarrier hom rows, object and hom action rows, identity-preservation rows, composition-preservation rows, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnaryContinuationEndofunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed surface names the concrete BEDC carrier rows for unary-continuation endofunctors over objects, homs, identity preservation, composition preservation, transport, route, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnaryContinuationEndofunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean namespace, host functor categories, quotient morphisms, external naturality objects, arbitrary functor categories, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, classifier transport, identity preservation, composition preservation, and non-escape over the displayed unary-continuation rows.

Chapter 1112

A Concrete Naming Certificate for $\mathsf{CrossHistCausalCone}^{\uparrow}$

The $\mathsf{CrossHistCausalCone}^{\uparrow}$ packet realizes the bounded two-history cone object in the Inter-Hist Locality chain of chapter 9419. It sits after the finite configuration surface of chapter 474, the causal witness surface of chapter 478, the maximum-rate boundary of chapter 594, the observer-locality transport surface of chapter 889, and the no-global-synchronization boundary of chapter 1003. Its role is to bind one directed affected-prefix window and its maximum-rate ledger into a finite carrier before any downstream consumer reads a causal cone as physical structure.

Definition 1112.1 (Cross-Hist causal-cone carrier)**.** *A* $\mathsf{CrossHistCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (A, B, K, W, R, M, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-history row, read through the same finite configuration surface as* $\mathrm{MultiHistConfig}^{\uparrow}$*;*
2. *B is the target-history row whose affected prefix is inspected;*
3. *K is the unary slot or budget row selecting the bounded two-history time window;*
4. *W is the affected-prefix route, naming the finite* $\mathsf{CrossHistCausal}^{\uparrow}$ *dependence rows from A into the selected prefix of B;*
5. *R is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *bound row attached to the displayed route W;*
6. *M is the observer-symmetry row, recording that the cone can be reread through the observer-locality transport of chapter 889 without changing the displayed bound;*
7. *S is the no-global-frame refusal row, inherited from chapter 1003 and excluding privileged frames, synchronized clocks, ambient metric tensors, and host-valued speed rows;*
8. *H records componentwise* hsame *and* psame *transport for the two history rows, slot row, affected-prefix route, maximum-rate bound, symmetry row, refusal row, continuation routes, provenance, and local name;*
9. *C records displayed* Cont *replay of the source row, target row, window row, affected-prefix route, maximum-rate bound, symmetry row, refusal row, and transport rows;*
10. *P is the* Pkg *provenance over* $\mathsf{CrossHistCausalCone}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{ObserverLocalityTransport}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, BHist, hsame, psame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}$ *row.*

The carrier contains one bounded two-history cone packet. It contains no global frame, synchronized clock, ambient metric geometry, host real-valued speed, detached causal relation, or maximum-rate row outside the displayed cone.

Theorem 1112.2 (Cross-Hist causal-cone NameCert obligations)**.** *For the carrier of Definition 1112.1, the local* $\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$Z = (A, B, K, W, R, M, S, H, C, P, N),$$

source and target history exactness, bounded slot/window exactness, affected-prefix route replay through W, maximum-rate attachment through R, observer-symmetry transport through M, no-global-frame refusal through S, componentwise hsame *and* psame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 1112.1. The coordinates $A, B, K, W, R, M, S, H, C, P, N$ are the complete public inventory of the cone packet.

The source, target, and slot rows fix the bounded two-history window. The affected-prefix route W is the only causal-dependence coordinate, and the maximum-rate row R is attached to that route rather than to an unlisted family of histories.

The observer-symmetry row M and refusal row S then determine the observer-facing boundary. Transport in H and replay in C preserve the same finite cone, while P and N record provenance and local naming. Since no further coordinate is present, the listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalConeUp.CrossHistCausalConeNameCert obligations`

Theorem 1112.3 (Cross-Hist causal-cone zero-rate exclusion)**.** *If an accepted* $\mathsf{CrossHistCausalCone}^{\uparrow}$ *packet has a zero maximum-rate row R, then no public consumer can read a nonempty affected-prefix dependence route through the same packet. Any such route would have to be the displayed row W and hence would be bounded by the zero row attached to R.*

Proof. Project the accepted carrier. The only affected-prefix coordinate is W, and the only maximum-rate coordinate is R.

Apply Definition 1112.2. It attaches R to W inside the same local certificate, so a consumer cannot detach the route from its bound or replace the bound by an external rate row.

When R is the zero bound, an accepted public dependence read through the same cone would have to be both nonempty and zero-bounded. That violates the displayed cone packet rather than producing a second route, because the carrier has no coordinate for an alternate causal family or host-side rate comparison. □

Lean4 ✓ `BEDC.Derived.CrossHistCausalConeUp.CrossHistCausalConeUp zero rate exclusion`

Theorem 1112.4 (Cross-Hist causal-cone observer-local boundary)**.** *Every observer-local consumer of an accepted* $\mathsf{CrossHistCausalCone}^{\uparrow}$ *packet factors through the single finite tuple*

$$A, B, K, W, R, M, S, H, C, P, N.$$

The consumer reads the source and target histories through the finite configuration surface, the affected-prefix dependence through $\mathsf{CrossHistCausal}^{\uparrow}$*, the bound through* $\mathsf{MaxCausalRate}^{\uparrow}$*, the observer-symmetry transport through* $\mathsf{ObserverLocalityTransport}^{\uparrow}$*, and the refusal row through* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$*. It receives no global frame, synchronized clock, metric tensor, Lorentz coordinate, host real-valued speed, detached rate row, or quotient of histories.*

Proof. Start with the carrier inventory in Definition 1112.1. The only public cone rows are the source, target, slot, affected-prefix route, maximum-rate bound, observer-symmetry row, refusal row, transport, replay, provenance, and local name.

Use Definition 1112.2. It routes the history rows through the finite configuration surface, the affected-prefix row through $\mathsf{CrossHistCausal}^{\uparrow}$, and the bound through $\mathsf{MaxCausalRate}^{\uparrow}$ before any observer-local consumer can read the cone.

The observer-symmetry and no-global-frame rows then close the consumer boundary. Because M and S are part of the same finite packet, observer-local transport cannot supply a privileged frame or a detached metric datum. The consumer remains inside $A, B, K, W, R, M, S, H, C, P, N$. □

Theorem 1112.5 (Cross-Hist causal-cone sibling independence)**.** *The* $\mathsf{CrossHistCausalCone}^{\uparrow}$ *carrier is not bijective to* $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, $\mathsf{ObserverLocalityTransport}^{\uparrow}$, *or* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$. *It carries all five sibling roles at once: source and target histories, affected-prefix causal row, maximum-rate bound, observer-symmetry transport, and no-global-frame refusal.*

Proof. Compare the carrier rows. $\mathrm{MultiHistConfig}^{\uparrow}$ supplies finite history slots but not the affected-prefix route, maximum-rate bound, observer-symmetry transport, or refusal row as one cone. $\mathsf{CrossHistCausal}^{\uparrow}$ supplies causal witness rows but not the bound and observer-local refusal ledger.

$\mathsf{MaxCausalRate}^{\uparrow}$ supplies the unary bound but not the selected affected-prefix route as a cone carrier. $\mathsf{ObserverLocalityTransport}^{\uparrow}$ supplies transport but not the bounded causal route itself. $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ supplies refusal data but not the positive causal and maximum-rate rows.

Thus each sibling omits at least one coordinate of $Z = (A, B, K, W, R, M, S, H, C, P, N)$. No projection to a single sibling can be inverted without recreating omitted cone data. □

Theorem 1112.6 (Cross-Hist causal-cone TasteGate surface)**.** *The later Lean* `TasteGate.lean` *development for* $\mathsf{CrossHistCausalCone}^{\uparrow}$ *must include explicit* Nontrivial $\mathsf{CrossHistCausalCone}^{\uparrow}$ *and* FieldFaithful $\mathsf{CrossHistCausalCone}^{\uparrow}$ *instances. These instances must preserve the source row, target row, slot/window row, affected-prefix route, maximum-rate bound, observer-symmetry row, no-global-frame refusal, transport rows, continuation replay, package provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.CrossHistCausalConeUp.CrossHistCausalConeTasteGate single carrier alignment`

Proof. Nontriviality is forced by the carrier inventory. A cone packet must distinguish at least the two history rows, selected window, affected-prefix route, maximum-rate bound, observer-symmetry row, and refusal row.

Field faithfulness is rowwise. A gate-facing instance that forgets any of $A, B, K, W, R, M, S, H, C, P, N$ would let a consumer read a collapsed object as a cone while omitting one of the local $\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}$ obligations.

Therefore the TasteGate surface must preserve exactly the finite packet named in Definition 1112.1. □

Falsifiable prediction: If the cone carrier is accepted, zero max-rate rows forbid any public affected-prefix dependence route through the same cone packet.

Independence witness: Compare multihistconfig, crosshistcausal, maxcausalrate, observerlocalitytransport, and noglobalsyncboundary: none is bijective because none carries all of causal row, max-rate bound, observer-symmetry transport, and no-global-frame refusal in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{CrossHistCausalCone}^{\uparrow}$ packet is generated from source and target history rows, a bounded slot/window row, an affected-prefix causal route, a maximum-rate bound, observer-symmetry transport, no-global-frame refusal, componentwise hsame and psame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1112.1, Definition 1112.2, Definition 1112.3, Definition 1112.4, Definition 1112.5, and Definition 1112.6 specify the finite bounded two-history causal-cone packet and its local consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CrossHistCausalCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive a numerical speed constant, Lorentz geometry, metric tensors, synchronized clocks, global frames, host real-valued speed, Lean verification, or bridge checking for $\mathsf{CrossHistCausalCone}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, affected-prefix replay, maximum-rate attachment, observer-symmetry preservation, no-global-frame refusal, and TasteGate nontriviality with field faithfulness.

Chapter 1113

A Concrete Naming Certificate for $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$

The $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ packet realizes the Real-completeness witness route of chapter 9238 as the finite budget layer consumed by the $\mathsf{CauchyWitnessGluing}^{\uparrow}$ packet of chapter 1110. Its role is narrower than a full gluing certificate: it binds the witness ledger, tail envelope, synchronizer, completion-seal classifier, stream window, regular-sequence handoff, dyadic tolerance ledger, and terminal Real seal into one finite budget surface before diagonal-limit or Real-completion consumers read the route.

Definition 1113.1 (Finite Cauchy gluing-budget carrier)**.** *A* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, L, T, S, K, U, V, D, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *route row naming the witness-gluing consumer surface;*
2. *L is the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row recording the selected witness ledger consumed by G;*
3. *T is the* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *row recording the finite tail envelope served to the gluing route;*
4. *S is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row aligning the witness ledger, tail envelope, selected window, and completion-seal request;*
5. *K is the* $\mathsf{RealCompletionSealClassifierUp}$ *row classifying the synchronized completion-seal read;*
6. *U is the* $\mathsf{StreamName}^{\uparrow}$ *finite window-family row exposed to diagonal-limit readers;*
7. *V is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence handoff row reached from the stream windows;*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance ledger used to check the finite observation budget;*
9. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row served by the gluing budget;*
10. *H records componentwise* hsame *transport for $G, L, T, S, K, U, V, D, R$ and for the structural rows;*
11. *C records displayed* Cont *routes from the gluing row to the witness ledger, tail envelope, synchronizer, classifier, stream windows, dyadic ledger, regular handoff, and Real seal;*

12. *P is the* Pkg *provenance over chapter 9238, chapter 1110,* $\mathsf{CauchyWitnessLedger}^{\uparrow}$, $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, RealCompletionSealClassifierUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, *and* NameCert *rows;*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyGluingBudget}^{\uparrow}}$ *row.*

The carrier contains one finite gluing-budget packet. It contains no quotient stream, host real equality, selected external limit, countable-choice row, hidden synchronizer, detached completion-seal classifier, or ambient completeness principle.

Definition 1113.2 (Finite Cauchy gluing-budget classifier)**.** *The local classifier for* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *accepts exactly those candidate packets*

$$B = (G, L, T, S, K, U, V, D, R, H, C, P, N)$$

whose rows jointly display the same witness-gluing route, witness ledger, tail envelope, synchronizer, completion-seal classifier, StreamName window family, RegSeqRat handoff, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame *transport, displayed* Cont *replay, package provenance, and local NameCert row. A candidate that preserves L and T but changes S is a different candidate and is rejected as a read of the same budget packet.*

Theorem 1113.3 (Finite Cauchy gluing-budget obligation surface)**.** *For the carrier of Definition 1113.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyGluingBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$B = (G, L, T, S, K, U, V, D, R, H, C, P, N),$$

gluing-row exactness through G, witness-ledger exactness through L, tail-envelope exactness through T, synchronizer exactness through S, completion-seal classifier exactness through K, finite-window exactness through U, regular-sequence handoff exactness through V, dyadic-tolerance exactness through D, Real-seal exactness through R, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyGluingBudgetUp.TasteGate.FiniteCauchyGluingBudgetObligationSurface`

Proof. Project the carrier definition. The rows $G, L, T, S, K, U, V, D, R, H, C, P, N$ are the complete public inventory of the budget packet, and the definition excludes every ambient completion object named in the statement.

The first block fixes the gluing route and its witness side. The row G names the consumer surface, while L and T determine the witness ledger and finite tail envelope read by that surface.

The second block fixes the seal side. The synchronizer S binds the ledger and envelope to the completion-seal classifier K, then U, D, V expose the finite stream, dyadic, and regular-sequence rows before R is served.

The structural rows close the local certificate. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row are the remaining public coordinates, so the listed clauses exhaust the obligation surface. □

Theorem 1113.4 (Finite Cauchy gluing-budget route exactness)**.** *Every accepted* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *packet presents the same Real-completion route to a consumer precisely when the consumer preserves*

$$G, L, T, S, K, U, V, D, R, H, C, P, N.$$

In particular, preserving the witness ledger L and tail envelope T while replacing the synchronizer row S does not preserve the accepted classifier read.

Proof. Use Definition 1113.2. The classifier reads the displayed tuple as one packet, not as independent witness, tail, and seal fragments.

Preserving all displayed rows preserves the route. The witness ledger, tail envelope, synchronizer, classifier, stream windows, dyadic ledger, regular handoff, and Real seal remain tied to the same gluing row and to the same structural rows.

If S is replaced while L and T are fixed, the row that binds the witness side to the completion-seal classifier has changed. The candidate may be inspected as another packet, but it is rejected as the same $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ classifier read. □

Lean4 ✓ `BEDC.Derived.FiniteCauchyGluingBudgetUp.FiniteCauchyGluingBudgetCarrier route exactness`

Theorem 1113.5 (Finite Cauchy gluing-budget nonescape)**.** *Every diagonal-limit,* $\mathsf{CauchyWitnessGluing}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *packet factors through*

$$G \to L \to T \to S \to K \to U \to D \to V \to R$$

with H, C, P, N *visible. No such consumer can reach a Real seal through an omitted witness ledger, hidden tail envelope, alternate synchronizer, detached completion-seal classifier, quotient stream, host real equality, selected external limit, countable-choice row, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyGluingBudgetUp.FiniteCauchyGluingBudgetCarrier nonescape`

Proof. Start from the carrier inventory of Definition 1113.1. The finite tuple is the whole packet, and the excluded objects are not coordinates of the carrier.

A diagonal-limit consumer first reads the gluing route and witness side. The rows G, L, T determine which witness-gluing surface, ledger, and tail envelope are available.

The consumer then reads the synchronized seal side. The rows S and K determine the completion-seal request, while U, D, V provide the finite StreamName, DyadicRatCore, and RegSeqRat handoff before the Real seal R is exposed.

Finally, the structural rows make the route public. Since H, C, P, N are visible and no further coordinate exists, the consumer has no escape route to a second seal or to an ambient completion principle. □

Theorem 1113.6 (Finite Cauchy gluing-budget witness-ledger exhaustion)**.** *For any accepted* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *packet*

$$B = (G, L, T, S, K, U, V, D, R, H, C, P, N),$$

every witness-ledger read available to a diagonal-limit, $\mathsf{CauchyWitnessGluing}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer is exhausted by the displayed route*

$$G \to L \to T \to S \to K \to U \to D \to V \to R$$

with G, H, C, P, N *visible. A purported extra witness ledger, hidden tail, alternate synchronizer, detached completion-seal classifier, quotient stream, selected limit, countable-choice row, or ambient completeness principle is outside the carrier.*

Proof. Project the carrier of Definition 1113.1. The tuple $B = (G, L, T, S, K, U, V, D, R, H, C, P, N)$ is the finite budget packet, so a consumer receives only rows displayed by that carrier.

Next apply Definition 1113.3. It identifies the complete public inventory: the witness ledger L, tail envelope T, synchronizer S, completion-seal classifier K, stream window U, dyadic ledger D, regular handoff V, Real seal R, and the structural rows G, H, C, P, N.

Apply Definition 1113.4 to preserve the classifier read along that inventory. Replacing any ledger, tail, synchronizer, classifier, stream, handoff, tolerance, or seal coordinate produces a different candidate rather than another read of the same accepted packet.

Finally use Definition 1113.5. Since the consumer factors through the displayed route and the structural rows remain visible, every witness-ledger read has already been accounted for by the finite carrier; the listed escape objects are not coordinates of the accepted budget. □

Falsifiable prediction: If the finite gluing-budget packet is accepted, a consumer preserving the witness ledger and tail envelope but changing the synchronizer row is rejected by the classifier.

Independence witness: cauchywitnessgluing, cauchywitnessledger, cauchytailwitnessenvelope, cauchysealbudgetsynchronizer, realcompletionsealclassifier: none is bijective because each sibling omits at least one of the gluing route, witness ledger, tail envelope, synchronizer, completion-seal classifier, StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows required by this finite budget packet.

The Lean TasteGate development must provide explicit Nontrivial $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ and FieldFaithful $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ instances in BEDC.Derived.FiniteCauchyGluingBudgetUp.TasteGate.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyGluingBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ packet is generated from a finite Cauchy gluing route, witness ledger, tail envelope, seal-budget synchronizer, completion-seal classifier, StreamName row, RegSeqRat row, DyadicRatCore row, RealUp seal row, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier records the exact finite budget by which local Cauchy witnesses are glued and synchronized before a Real-facing seal is read. The support rows keep stream, regular, dyadic, and Real data on one route. The classifier preserves the witness ledger and tail envelope together; changing the synchronizer changes the exported finite budget surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyGluingBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the finite Cauchy gluing budget carrier, classifier rejection criterion, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyGluingBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim arbitrary Cauchy gluing, quotient regular-sequence equality, selected completed reals, or ambient completeness.

Upgrade path: To reach obligationClosure, formalize the witness ledger, tail envelope, synchronizer, and completion-seal classifier as checked finite budget rows.

Chapter 1114

A Concrete Naming Certificate for $\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}$

Falsifiable prediction: A packet that identifies the two substrate roles without a displayed refusal ledger is not an accepted TwinSubstrateBoundaryClassifierUp carrier.

Independence witness: twinsubstratesynthesis, metacicauditmapfamily, groundcompilerflowlevelroundtrip, certificateauditgate, substitutionboundaryaudit: none is bijective because each lacks the simultaneous two-substrate role classifier plus shared refusal ledger carried here.

The $\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}$ packet realizes the MetaCIC/GroundCompiler separation of chapter 9392 as a finite role classifier rather than as a request router. It sits beside the architecture packet of chapter 845 and the request-level router of chapter 851: those siblings name the cross-substrate architecture and its public routing discipline, while this chapter names the carrier that classifies the two substrate roles and records their shared refusal ledger before any consumer reads a boundary claim.

Definition 1114.1 (Twin-substrate boundary-classifier carrier)**.** *A* $\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, G, Q, L, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC substrate-role row, recording closed-CIC theorem inventory, proof-theoretic frontiers, substitution boundaries, and MetaCIC-side non-discharge clauses;*
2. *G is the GroundCompiler substrate-role row, recording compiler event-flow, recognizer, certificate, report, source-channel, and cannot-claim rows;*
3. *Q is the two-substrate classifier row, assigning each displayed boundary request to MetaCIC role data, GroundCompiler role data, paired-context data, or refusal data;*
4. *L is the shared refusal ledger, recording every forbidden conversion between compiler reports and proof evidence, between MetaCIC theorem rows and compiler-channel evidence, and between hidden host input and finite carrier data;*
5. *D is the downstream consumer row, exposing how a reader may use Q and L without collapsing M and G;*

6. *H records componentwise* hsame *transports for the two role rows, classifier row, refusal ledger, downstream consumer row, continuation row, provenance, and local naming;*
7. *C records displayed* Cont *routes replaying M, G, Q, L, D in finite order for boundary consumers;*
8. *P is the* Pkg *provenance over chapter 9392, chapter 845, chapter 851,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for identifying MetaCIC and GroundCompiler as one substrate, converting reports into proof evidence, converting proof rows into compiler-channel evidence, or hiding refusal data outside the packet.

Lean4 ✓ `BEDC.Derived.TwinSubstrateBoundaryClassifierUp.TwinSubstrateBoundaryClassifierTasteGate`
`single carrier alignment`

Theorem 1114.2 (Twin-substrate boundary-classifier NameCert obligations)**.** *For every accepted* $\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}$ *carrier*

$$B = (M, G, Q, L, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TwinSubstrateBoundaryClassifier}^{\uparrow}}$ *surface consists exactly of MetaCIC role readback through M, GroundCompiler role readback through G, two-substrate classification through Q, shared refusal-ledger accounting through L, downstream consumer exposure through D, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite carrier. The only substrate-role rows are M and G, the only classifier row is Q, and the only shared refusal ledger is L.

The classifier Q assigns a boundary read to MetaCIC role data, GroundCompiler role data, paired-context data, or refusal data before any consumer can use the read. The ledger L records the forbidden conversions for the same classification, so a role assignment is never separated from its refusal surface.

The downstream row D exposes only this classified and ledgered surface. Componentwise hsame transport in H preserves the projected rows, while Cont replay in C presents them in finite order.

Finally P records the twin-substrate provenance and N binds the local certificate name. Since no other coordinate is present, these rows exhaust the NameCert obligation surface. □

Theorem 1114.3 (Twin-substrate boundary-classifier refusal discipline)**.** *In every accepted* TwinSubstrateBoundary *carrier, identifying a boundary read as MetaCIC-facing, GroundCompiler-facing, paired-context, or refused requires the same shared refusal ledger L. A packet that supplies the role classifier Q but omits the displayed ledger L is not an accepted carrier.*

Proof. Start from Definition 1114.1. The classifier row Q and refusal ledger L are separate displayed coordinates of the same finite packet.

Apply the NameCert obligations of Definition 1114.2. They require classifier readback and ledger accounting together before the local certificate surface is complete.

If Q is present without L, a consumer could identify a substrate role while losing the explicit record that blocks report-to-proof conversion, proof-to-compiler conversion, and hidden host input. That packet lacks one mandatory NameCert coordinate, so it is outside the accepted carrier class. □

Lean4 ✓ `BEDC.Derived.TwinSubstrateBoundaryClassifierUp.TwinSubstrateBoundaryClassifierRefusalDiscipline`

Theorem 1114.4 (Twin-substrate boundary-classifier non-escape)**.** *Every public read exported by an accepted* TwinSubstrateBoundaryClassifier$^{\uparrow}$ *carrier factors through the finite package*

$$M, G, Q, L, D, H, C, P, N.$$

The export cannot collapse MetaCIC into GroundCompiler, collapse GroundCompiler into MetaCIC, replace the refusal ledger by prose, or route through a hidden host-level substrate classifier.

Proof. Use the carrier projection. Every exported read begins with the two displayed role rows, then passes through the classifier and the shared refusal ledger.

The downstream consumer row D only exposes the classified and ledgered surface. It does not add a role coordinate, a hidden classifier, or a conversion witness between the two substrate roles.

Transport, continuation, provenance, and naming preserve the same finite packet. Since the carrier contains no field for substrate collapse or host-level classification, no public read can export one. □

Theorem 1114.5 (Twin-substrate boundary-classifier public surface)**.** *The public surface of an accepted* TwinSubstrateBoundaryClassifier$^{\uparrow}$ *carrier is exactly the finite export of the MetaCIC role row, GroundCompiler role row, two-substrate classifier, shared refusal ledger, downstream consumer row, transport row, continuation row, provenance row, and local naming row. Downstream audit readers may distinguish the two substrate roles through this packet, but only with the displayed refusal ledger carried alongside the classifier.*

Proof. Project the public carrier fields. The reader receives $M, G, Q, L, D, H, C, P, N$ and no further role or refusal coordinate.

The classifier row Q distinguishes MetaCIC-facing, GroundCompiler-facing, paired-context, and refused reads. The refusal ledger L remains part of the same public surface, so every distinction carries the record of forbidden conversions.

The consumer, transport, continuation, provenance, and naming rows expose this data to downstream audit readers. They preserve the two-substrate classification without turning it into an unledgered route or a single merged substrate. □

Closure status for NameCert$_{\text{TwinSubstrateBoundaryClassifier}^{\uparrow}}$.

Constructive story: A TwinSubstrateBoundaryClassifier$^{\uparrow}$ packet is generated from MetaCIC and GroundCompiler substrate-role rows, a two-substrate classifier, a shared refusal ledger, a downstream consumer row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TwinSubstrateBoundaryClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 1114.1, Definition 1114.2, Definition 1114.3, Definition 1114.4, and Definition 1114.5 bind the finite role classifier, shared refusal ledger, non-escape boundary, and public export surface.

Formal verification: FormalStatus(NameCert$_{\text{TwinSubstrateBoundaryClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close MetaCIC proof-theoretic frontiers, prove GroundCompiler self-hosting, certify compiler report soundness, provide the Lean TasteGate Nontrivial and FieldFaithful instances, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, role classification, refusal-ledger exactness, consumer exposure, public export exactness, transport, continuation, provenance, naming, and TasteGate Nontrivial and FieldFaithful instances for the same finite packet.

Chapter 1115

A Concrete Naming Certificate for $\mathsf{RegularCauchyIntervalSelector}^{\uparrow}$

The $\mathsf{RegularCauchyIntervalSelector}^{\uparrow}$ packet names the finite interval selector surface used when a regular Cauchy rational name is read through nested located intervals. In Bishop-style constructive analysis, nested rational or located intervals are data that can be inspected at any requested finite precision; the limiting real is never obtained by selecting a quotient representative or invoking an ambient completeness oracle. The packet therefore sits between the finite observation surfaces of chapter 126, chapter 301, chapter 38, and chapter 394.

Definition 1115.1 (Regular Cauchy interval selector carrier)**.** *A* $\mathsf{RegularCauchyIntervalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, J, E, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator and radius ledger naming the requested precision;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that selects the finite observation windows inspected at that precision;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected rational observations;*
4. *J is the finite* $\mathsf{LocatedInterval}^{\uparrow}$ *nesting ledger whose cells contain the selected regular Cauchy observations and whose successive cells refine the previous visible cell;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after the selected schedule, regular-rational readback, and located-interval nesting ledger have been displayed;*
6. *H records componentwise* hsame *transports for denominator ledgers, schedules, regular-rational readbacks, located interval cells, the seal row, provenance, and local naming;*
7. *C records the displayed* Cont *replay routes from D to S, from S to R, from R to J, and from J to E;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyIntervalSelector}^{\uparrow}}$ *row.*

The classifier compares packets only by these displayed rows and their componentwise transports. It has no coordinate for a selected completed real, quotient equality of Cauchy names, a detached interval limit, or an ambient completeness principle.

Theorem 1115.2 (Regular Cauchy interval selector nesting extraction)**.** *For every accepted* RegularCauchyIntervalS *packet*

$$Q = (D, S, R, J, E, H, C, P, N),$$

the finite nesting route extracted for a precision request is exactly

$$D \longmapsto S \longmapsto R \longmapsto J.$$

The extracted route exposes the denominator and radius ledger, the selected $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{LocatedInterval}^{\uparrow}$ *nesting ledger, all transported by the displayed* hsame *rows and replayed by the displayed* Cont *rows. No extracted object is a quotient real, a chosen limit point, or a host interval.*

Lean4 ✓ `BEDC.Derived.RegularCauchyIntervalSelectorUp.RegularCauchyIntervalSelectorCarrier nesting extraction`

Proof. Project the carrier of Definition 1115.1. The first four rows are precisely the precision ledger D, the selected schedule S, the regular-rational readback R, and the located-interval nesting ledger J.

Read the route in dependency order. The schedule row is available only after the precision ledger is displayed, the regular-rational readback is available only through the selected schedule, and the nesting ledger records only the located interval cells reached from that readback.

Transport is componentwise. The rows in H compare the four projected coordinates by hsame, while the rows in C replay the same finite route through Cont. The package row P and the local naming row N record provenance for that route rather than supplying an additional interval object.

The excluded objects are absent from the carrier. Since Q contains no quotient-real coordinate, no chosen limit point, and no host interval coordinate, the extraction can return only the displayed finite nesting route. □

Theorem 1115.3 (Regular Cauchy interval selector RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyIntervalSelector}^{\uparrow}$ *packet factors through the finite route*

$$D \longmapsto S \longmapsto R \longmapsto J \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer may read the dyadic precision ledger, selected stream window, regular-rational readback, located interval nesting ledger, terminal $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transports, continuation replays, package provenance, and local* NameCert *row. It receives no quotient equality of regular Cauchy names, selected completed real, detached interval limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyIntervalSelectorUp.RegularCauchyIntervalSelectorCarrier real seal handoff`

Proof. Begin with the nesting extraction of Definition 1115.2. It fixes the only route from a precision request to a displayed located-interval nesting ledger as $D \mapsto S \mapsto R \mapsto J$.

Now inspect the seal-facing coordinate of Definition 1115.1. The row E is reachable only after the selected schedule, regular-rational readback, and located-interval nesting ledger have been displayed. Hence a $\mathsf{Real}^{\uparrow}$ consumer reaches E through the same finite route.

The structural rows do not widen the route. The hsame rows in H move the displayed coordinates componentwise, the Cont rows in C replay the displayed route, the package row P records provenance for that route, and N names the local certificate surface.

Finally, the carrier has no field for quotient equality of regular Cauchy names, a selected completed real, a detached interval limit, or an ambient completeness principle. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only the displayed selector-to-seal handoff. □

Theorem 1115.4 (Regular Cauchy interval selector schedule exposure)**.** *For every accepted* RegularCauchyIntervalS *packet*

$$Q = (D, S, R, J, E, H, C, P, N),$$

the selector exposes the dyadic precision ledger D before the selected $\mathsf{StreamName}^{\uparrow}$ *schedule S. The schedule row S is readable only together with the displayed* hsame *rows in H, the displayed* Cont *replay from D to S in C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyIntervalSelector}^{\uparrow}}$ *row N. No consumer may read S as an ambient stream-choice oracle detached from the displayed precision ledger and certificate rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyIntervalSelectorUp.RegularCauchyIntervalSelectorCarrier schedule exposure`

Proof. Project the carrier of Definition 1115.1. Its first coordinate is the dyadic denominator and radius ledger D, while its second coordinate is the selected stream schedule S.

The dependency order is part of the displayed carrier surface. Row C contains the replay route from D to S, row H supplies the componentwise hsame transport for the same displayed coordinates, and the rows P and N bind provenance and local naming for the selector.

Since the carrier has no coordinate for a detached schedule oracle, the only readable schedule is the row S reached through D, H, C, P, N. □

Theorem 1115.5 (Regular Cauchy interval selector located nesting transport)**.** *Let*

$$Q = (D, S, R, J, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyIntervalSelector}^{\uparrow}$ *packet. Replacing the displayed located-interval ledger J by a transported refinement J' preserves the selector route only when the precision ledger D, selected schedule S, regular-rational readback R, seal row E, componentwise transports H, continuation rows C, provenance row P, and local naming row N are transported with it. A detached interval limit, quotient real, or unrecorded refinement cannot replace this route.*

Proof. Use Definition 1115.2. The nesting route visible to the selector is exactly $D \mapsto S \mapsto R \mapsto J$, so any change of the located ledger must be read as a change inside that finite route.

Now inspect the remaining carrier rows in Definition 1115.1. The row E is reached only after J, while H, C, P, and N record the transports, replay routes, provenance, and local certificate that make the selected route readable.

Therefore a transported refinement J' preserves the selector route only as part of the transported packet. The excluded detached limit, quotient real, and unrecorded refinement have no row in the carrier and cannot serve as replacements. □

Theorem 1115.6 (Regular Cauchy interval selector obligation surface)**.** *The visible* $\mathsf{NameCert}_{\mathsf{RegularCauchyIntervalSelec}}$ *obligations for an accepted packet*

$$Q = (D, S, R, J, E, H, C, P, N)$$

are exactly the following five rows: the carrier definition of Definition 1115.1, the schedule exposure of Definition 1115.4, the nesting extraction of Definition 1115.2, the located nesting transport of Definition 1115.5, and the Real$^{\uparrow}$ *seal handoff of Definition 1115.3. These rows are the public obligation surface of the selector packet; they do not add quotient equality, detached interval limits, selected completed reals, or ambient completeness.*

Proof. The carrier row is Definition 1115.1. It names the finite packet coordinates $D, S, R, J, E, H, C, P, N$ and excludes any coordinate for a quotient real, detached interval limit, or completeness oracle.

The schedule row is supplied by Definition 1115.4. The nesting row is supplied by Definition 1115.2, and the transported refinement row is supplied by Definition 1115.5.

Finally, apply Definition 1115.3 for the seal-facing row. The five cited rows exhaust the readable selector obligations because every consumer route factors through the displayed carrier coordinates and their recorded transports. □

Theorem 1115.7 (Regular Cauchy interval selector finite route obligation package)**.** *For every accepted* RegularCauchyIntervalSelector$^{\uparrow}$ *packet*

$$Q = (D, S, R, J, E, H, C, P, N),$$

the displayed rows $D, S, R, J, E, H, C, P, N$ form the finite obligation package of the selector route. The row D exposes the dyadic precision ledger, S is schedule-dependent on D, R is the RegSeqRat$^{\uparrow}$ *readback through S, J is the located interval nesting ledger through R, and E is reached only through J. The remaining rows H, C, P, N bind the componentwise* hsame *transport,* Cont *replay, provenance, and local* NameCert *surface for that route.*

Proof. Start with the carrier definition of Definition 1115.1. Its displayed coordinates are exactly $D, S, R, J, E, H, C, P, N$, so the candidate obligation package has no hidden coordinate beyond those rows.

The dependency spine is fixed by the existing selector rows. Use schedule exposure from Definition 1115.4 to read S only after D, use nesting extraction from Definition 1115.2 to read $D \mapsto S \mapsto R \mapsto J$, and use located nesting transport from Definition 1115.5 to keep transported refinements inside the same finite route.

The obligation surface of Definition 1115.6 says that these rows exhaust the public selector obligations. Finally, Definition 1115.3 places the seal row E and the structural rows H, C, P, N after the displayed route, which gives the finite package claimed here. □

Closure status for NameCert$_{\text{RegularCauchyIntervalSelector}^{\uparrow}}$.

Constructive story: A RegularCauchyIntervalSelector$^{\uparrow}$ packet is generated from a selected finite schedule, regular-rational readback row, located-interval nesting ledger, seal-facing handoff row, componentwise hsame transports, displayed Cont routes, provenance, and local naming. The carrier reads interval selection through finite BEDC data: first the schedule is displayed, then the RegSeqRat readback and nesting ledger certify the selected interval route, and only then the RealUp-facing seal row is available. The classifier keeps the selector-to-seal handoff finite and refuses detached interval limits or quotient Cauchy names.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyIntervalSelector}^{\uparrow}}$, obligation)

Scope closed: This chapter binds the finite route obligation package of Definition 1115.7, including the checked rows of Definition 1115.2 and Definition 1115.3.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyIntervalSelector}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyIntervalSelectorUp.RegularCauchyIntervalSelectorCarrier\_nesting\_extracti`

Bridge status: none

Origin: human

Not claimed: It does not claim quotient equality of regular Cauchy names, selected completed reals, detached interval limits, or ambient completeness.

Upgrade path: To reach scopedClosure, close the carrier interactions across BHist, hsame, Cont, Pkg, NameCert, StreamNameUp, RegSeqRatUp, LocatedIntervalUp, DyadicRatCoreUp, and RealUp.

Chapter 1116

A Concrete Naming Certificate for $\mathsf{CauchyLimitFunctor}^{\uparrow}$

The $\mathsf{CauchyLimitFunctor}^{\uparrow}$ packet names the functorial transport surface for Cauchy limit seals in the Real and Completion route of chapter 9238. It sits after the dyadic tolerance core of chapter 291, the finite stream window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. It also consumes the existing limit-seal boundary of chapter 386 and the completion-seal classifier of chapter 900. The packet is a routed transport object: it carries a finite source seal, a finite target seal, and the displayed map between their regular Cauchy data without adding a host-level completion theorem.

Falsifiable prediction: If a completion consumer accepts a transported Cauchy limit seal while changing the source stream window, target stream window, regular-rational readback, dyadic tolerance ledger, or RealUp endpoint row outside the displayed transport packet, it refutes the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ classifier.

Independence witness: cauchylimitseal, realcompletionsealclassifier, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{CauchyLimitFunctor}^{\uparrow}$ because the functor packet carries source and target limit-seal rows together with the finite transport map, classifier row, dyadic tolerance ledger, regular-rational readback, RealUp endpoint, componentwise transports, continuation replay, provenance, and local NameCert row.

Definition 1116.1 (Cauchy limit functor carrier)**.** *A* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row whose limit-seal route is being transported;*
2. *T is the target* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row reached after transport;*
3. *M is the finite transport-map row from the source window and readback data to the target window and readback data;*
4. *W_0 is the source* $\mathsf{StreamName}^{\uparrow}$ *window row read by S;*
5. *W_1 is the target* $\mathsf{StreamName}^{\uparrow}$ *window row read by T;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row preserved by the displayed transport;*

7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger controlling the transported precision request;*
8. *K is the* RealCompletionSealClassifierUp *row that recognizes the transported seal boundary;*
9. *E is the terminal* Real$^{\uparrow}$ *endpoint row consumed after the target seal is displayed;*
10. *H records componentwise* hsame *transports for the two limit seals, map row, stream windows, regular-rational readback, dyadic ledger, classifier, endpoint, provenance, and local naming rows;*
11. *C records the displayed* Cont *replay route from S through M, W_0, W_1, R, D, K to T and then to E;*
12. *P is the* Pkg *provenance over chapter 386, chapter 900, chapter 126, chapter 301, chapter 291, chapter 38,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
13. *N is the local* NameCert$_{\text{CauchyLimitFunctor}^{\uparrow}}$ *row.*

The carrier exports no quotient stream equality, selected host limit, ambient completeness principle, countable-choice selector, detached Real value, or second transport route beyond the displayed packet.

Definition 1116.2 (Cauchy limit functor classifier)**.** *The* CauchyLimitFunctor$^{\uparrow}$ *classifier compares two carriers only by the displayed rows*

$$S, T, M, W_0, W_1, R, D, K, E, H, C, P, N.$$

Two packets are classifier-equal exactly when the source and target limit-seal rows agree, the transport-map row agrees, the source and target StreamName$^{\uparrow}$ *windows agree, the* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance ledger agree, and the classifier, endpoint, transport, continuation, provenance, and local naming rows agree componentwise. The classifier has no coordinate for quotient equality of Cauchy names, host real equality, an untracked selector, a detached limit object, or an ambient completeness theorem.*

Theorem 1116.3 (Cauchy limit functor NameCert obligations)**.** *For the carrier of Definition 1116.1, the local* NameCert$_{\text{CauchyLimitFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier exposure for $F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$;*
2. *source-to-target seal exposure through the ordered route $S \to M \to T$;*
3. *finite observation exposure through W_0, W_1, R, D;*
4. *completion-consumer exposure through K and E;*
5. *structural exposure through componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming N.*

No CauchyLimitFunctor$^{\uparrow}$ *consumer can read the transported limit seal without preserving all five rows.*

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier namecert obligations`

Proof. Project the carrier of Definition 1116.1. Its displayed packet is exactly $F = (S, T, M, W_0, W_1, R, D, K, E, H,$ so the carrier row supplies the finite object being named.

Read the seal route next. The source seal S is transported only through the finite map row M, and the target seal T is reached only after that row is displayed. Thus the seal exposure is the ordered route $S \to M \to T$.

Now inspect the observation rows. The source and target windows W_0, W_1, the regular-rational readback R, and the dyadic tolerance ledger D are the finite data over which the transport map is read. The completion-facing rows K, E then recognize the transported seal boundary and deliver the terminal $\mathsf{Real}^{\uparrow}$ endpoint.

Finally the rows H, C, P, N transport, replay, record provenance for, and name the same packet. Since the carrier and classifier contain no further coordinates, a consumer that bypasses any one of the five obligation rows leaves the $\mathsf{CauchyLimitFunctor}^{\uparrow}$ surface. □

Theorem 1116.4 (Cauchy limit functor source-target seal exposure)**.** *For every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N),$$

the source seal is exposed to consumers only through the finite route $S \to M \to T$*. In particular, a consumer can cite* S *and* T *as the displayed source and target* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows, and can cite* M *as the finite transport-map row between them, but receives no detached source-target equality, host-level limit comparison, quotient stream equality, or second seal route outside the classifier of Definition 1116.2.*

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier source target seal exposure`

Proof. Use the carrier of Definition 1116.1. Its first three displayed rows are exactly the source seal S, the target seal T, and the transport-map row M.

Apply the classifier of Definition 1116.2. It compares the seal data through the listed coordinates $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$ and has no coordinate for a detached source-target equality or an alternate route.

Finally apply the NameCert obligation surface of Definition 1116.3. The second obligation row is precisely source-to-target seal exposure through $S \to M \to T$, so any accepted packet exposes the seal only by that displayed finite route. □

Theorem 1116.5 (Cauchy limit functor finite observation exposure)**.** *For every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N),$$

the observation side available to a consumer is exactly

$$W_0, W_1, R, D, K, E.$$

The consumer may inspect the two stream windows, the regular-rational readback, the dyadic tolerance ledger, the completion-seal classifier row, and the terminal $\mathsf{Real}^{\uparrow}$ *endpoint row. It receives no quotient stream equality, selected limit, countable-choice selector, host real equality, detached completion object, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier finite observation exposure`

Proof. Project the carrier of Definition 1116.1. The observation rows are the source and target windows W_0, W_1, the readback row R, the tolerance ledger D, the completion classifier K, and the endpoint row E.

Use the classifier of Definition 1116.2. It allows comparison only along the displayed carrier coordinates, so there is no classifier coordinate for quotient stream equality, a selected limit, countable choice, host real equality, or an ambient completion theorem.

The NameCert obligations of Definition 1116.3 split the observation surface into finite observation exposure through W_0, W_1, R, D and completion-consumer exposure through K, E. These two rows give exactly the displayed observation side and nothing outside it. □

Theorem 1116.6 (Cauchy limit functor structural transport)**.** *For every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N),$$

the only structural transport, continuation replay, provenance, and local naming rows available to consumers are H, C, P, N*. Thus all componentwise* hsame *transport is carried by* H*, all displayed replay is carried by* C*, all dependency provenance is carried by* P*, and the local certificate name is carried by* N*.*

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier structural transport`

Proof. Start from the carrier of Definition 1116.1. Its final four rows are H, C, P, N, respectively recording componentwise hsame transports, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}^{\uparrow}}$ row.

Read those rows through the classifier of Definition 1116.2. Since the classifier compares only the displayed coordinates, it cannot supply any additional transport object, replay path, provenance row, or local name beyond H, C, P, N.

Apply the fifth obligation row of Definition 1116.3. Structural exposure is exactly componentwise transport H, displayed continuation replay C, provenance P, and local naming N, so every consumer is confined to those structural rows. □

Theorem 1116.7 (Cauchy limit functor Real-completion consumer boundary)**.** *Every Real and Completion consumer that cites an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto M \longmapsto W_0, W_1, R, D \longmapsto K \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

The consumer may use the source and target $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows, the transport map, the two* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* RealCompletionSealClassifierUp *row, the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It receives no quotient stream equality, host real equality, selected limit, detached completion object, ambient completeness principle, countable-choice selector, or second transport route.*

Proof. Start with the NameCert obligations of Definition 1116.3. They force every accepted read through the carrier row, the source-to-target seal route, the finite observation rows, the completion-consumer rows, and the structural rows.

Use the classifier of Definition 1116.2. It compares only $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$, so a Real and Completion consumer cannot distinguish two reads by adding quotient stream equality, host real equality, a selected limit, or an ambient completion proof.

Finally follow the route displayed in the carrier. The source seal S is transported by M over the windows W_0, W_1, readback R, and tolerance ledger D; the classifier row K recognizes the transported boundary before the target seal T and endpoint row E are consumed. The structural rows H, C, P, N package that same finite route, so no detached completion object, choice-selected witness, or second route remains available. □

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier real completion consumer boundary`

Theorem 1116.8 (Cauchy limit functor primitive scope package)**.** *Every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

binds its source-target seal route, finite observation rows, completion-consumer rows, and structural rows only through the primitive BEDC objects BHist, hsame, Cont, Pkg, *and* NameCert, *together with the sibling source certificates* CauchyLimitSeal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealCompletionSealClassifierUp, *and* Real$^{\uparrow}$. *Thus the scoped packet consists exactly of the displayed carrier row, classifier coordinates, source-to-target route* $S \to M \to T$, *observation rows* W_0, W_1, R, D, *completion rows* K, E, *structural rows* H, C, P, N, *and their cited sibling certificates.*

No quotient stream equality, host real equality, selected limit, detached completion object, ambient completeness principle, countable-choice selector, or second route enters the scoped packet.

Proof. Start with the carrier and classifier of Definition 1116.1 and Definition 1116.2. The carrier is the finite BHist packet $F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$, and the classifier reads exactly those coordinates, so every row available to the packet is already one of the displayed source, target, map, observation, completion, structural, provenance, or naming rows.

Apply the NameCert obligation surface of Definition 1116.3. Its five rows identify carrier exposure, source-to-target seal exposure, finite observation exposure, completion-consumer exposure, and structural exposure. These are the only rows a local NameCert$_{\text{CauchyLimitFunctor}^{\uparrow}}$ consumer may preserve.

Next use Definition 1116.4, Definition 1116.5, and Definition 1116.6. They place the seal route at $S \to M \to T$, the finite observation surface at W_0, W_1, R, D, K, E, and the structural surface at H, C, P, N, where H carries componentwise hsame, C carries displayed Cont, P carries Pkg provenance, and N carries the local NameCert row.

Finally apply the Real-completion consumer boundary of Definition 1116.7. It keeps the completion read inside the same route through the sibling source certificates CauchyLimitSeal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealCompletionSealClassifierUp, and Real$^{\uparrow}$. Since each cited result excludes quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, and second routes, the primitive-scoped packet contains no further object beyond BHist, hsame, Cont, Pkg, NameCert, and those sibling certificates. □

Theorem 1116.9 (Cauchy limit functor public seal export)**.** *Every accepted* CauchyLimitFunctor$^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

exports a reusable public transport surface from the source CauchyLimitSeal$^{\uparrow}$ *row* S *through the displayed transport map* M, *the* StreamName$^{\uparrow}$ *windows* W_0, W_1, *the* RegSeqRat$^{\uparrow}$ *readback* R, *the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* D, *the* RealCompletionSealClassifierUp *row* K, *and the terminal* Real$^{\uparrow}$ *endpoint* E. *The exported surface is reusable only as the public route*

$$S \longmapsto M \longmapsto W_0, W_1, R, D \longmapsto K \longmapsto T \longmapsto E \longmapsto H, C, P, N,$$

with H *carrying componentwise* hsame *transport,* C *carrying displayed* Cont *replay,* P *carrying* Pkg *provenance, and* N *carrying the local* NameCert$_{\text{CauchyLimitFunctor}^{\uparrow}}$ *row.*

No quotient stream equality, host real equality, selected limit, detached completion object, ambient completeness principle, countable-choice selector, or second transport route is exported by this public surface.

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier public seal export`

Proof. Start with the carrier and classifier of Definition 1116.1 and Definition 1116.2. They make the public packet a finite read of the displayed rows $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$, so a reusable export can only cite those rows.

Apply the NameCert obligations of Definition 1116.3. The obligation rows expose the carrier, the source-target seal route, finite observations, Real-completion consumer boundary, and structural rows. Use Definition 1116.4 and Definition 1116.5 to identify the public route through $S, M, W_0, W_1, R, D, K, T, E$.

Next apply Definition 1116.6. It supplies exactly the reusable transport, replay, provenance, and naming rows H, C, P, N, so the exported surface can be handed to later consumers without adding any route outside the packet.

Finally use Definition 1116.7 and Definition 1116.8. The first confines Real and Completion consumers to the same finite transport route, while the second keeps the route primitive-scoped. The exclusions in those two results remove quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, and second routes from the public export. □

Theorem 1116.10 (Cauchy limit functor terminal-window fusion)**.** *Every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

fuses the source terminal $\mathsf{StreamName}^{\uparrow}$ *window* W_0*, target terminal* $\mathsf{StreamName}^{\uparrow}$ *window* W_1*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RealCompletionSealClassifierUp}$ *classifier row* K*, and terminal* $\mathsf{Real}^{\uparrow}$ *row* E *into the single terminal-window route*

$$S \longmapsto M \longmapsto W_0, W_1, R, D \longmapsto K \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

The route has no quotient stream equality, selected host limit, countable-choice selector, detached completion object, or second Real-facing route.

Proof. Project the carrier of Definition 1116.1. The only finite source and target windows are W_0 and W_1, and the only Real-facing observation rows attached to them are R, D, K, E.

Use finite observation exposure, Definition 1116.5. It identifies W_0, W_1, R, D, K, E as the complete observation side available to a consumer and excludes quotient stream equality, selected limits, countable-choice selectors, host real equality, detached completion objects, and ambient completeness.

Finally apply the Real-completion consumer boundary and the public seal export, Definition 1116.7 and Definition 1116.9. Together they force the terminal Real read through the same route from S through M, the two windows, readback, tolerance, classifier, target seal, endpoint, and structural rows. Hence the terminal-window route is fused inside the accepted packet and no second Real-facing route remains. □

Theorem 1116.11 (Cauchy limit functor Real-seal naturality)**.** *Let*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet. Componentwise* hsame *transport along the structural row* H *preserves the terminal-window route of Definition 1116.10 before any Real-facing seal is made public. Thus the transported source seal, target seal, stream windows, readback, tolerance ledger, classifier row, terminal Real row, continuation replay, provenance, and local name still form the same single route, with no public seal read adding a quotient stream equality, selected host limit, countable-choice selector, detached completion object, or second Real-facing route.*

Proof. Read the classifier of Definition 1116.2. It compares packets only by the rows $S, T, M, W_0, W_1, R, D, K, E, H$, so componentwise transport cannot create a new coordinate for a second seal route or a detached Real object.

Use structural transport, Definition 1116.6. The row H is the only componentwise hsame transport surface, while C, P, N keep the replay, provenance, and local name attached to the same packet.

Apply terminal-window fusion, Definition 1116.10. Since the fused route is already determined before public seal export, transporting each displayed component along H preserves that same route. The exclusions in the fusion theorem therefore persist before any Real-facing seal becomes public. □

Theorem 1116.12 (Cauchy limit functor bridge-scope coherence)**.** *Every bridge-scope consumer of an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

must preserve the finite route

$$S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$$

through $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, RealCompletionSealClassifierUp, *and* $\mathsf{Real}^{\uparrow}$. *The bridge-facing read may expose the public transport surface of Definition 1116.9, but it cannot add quotient stream equality, host real equality, selected limits, ambient completeness, countable choice, or a second transport route.*

Proof. Start with the carrier and classifier. The packet rows are exactly $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$, and the classifier reads only those coordinates.

Apply the public seal export, Definition 1116.9. It exposes the reusable route from S through M, the stream windows W_0, W_1, the readback R, the tolerance ledger D, the classifier row K, the target seal T, the terminal endpoint E, and the structural rows H, C, P, N.

Next use the primitive scope package and Real-completion consumer boundary, Definition 1116.8 and Definition 1116.7. These place the route through $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, RealCompletionSealClassifierUp, and $\mathsf{Real}^{\uparrow}$ without admitting a detached completion object.

Carrier-coordinate exhaustion gives bridge-scope coherence. Since the displayed packet has no coordinate for quotient stream equality, host real equality, selected limits, ambient completeness, countable choice, or a second transport route, a bridge-scope consumer can preserve only the finite route already exported by the public seal surface. □

Theorem 1116.13 (CauchyLimitFunctor standard reflection bridge surface)**.** *Every standard-reflection read of an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

factors through the displayed source seal S*, target seal* T*, transport row* M*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W_0, W_1*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* RealCompletionSealClassifierUp *classifier row* K*,* $\mathsf{Real}^{\uparrow}$ *endpoint* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}^{\uparrow}}$ *row* N*.*

Thus the bridge surface is the single route

$$S \longmapsto M \longmapsto W_0, W_1, R, D \longmapsto K \longmapsto T \longmapsto E \longmapsto H, C, P, N,$$

read through the same primitive-scoped packet. It exports no quotient stream equality, host real equality, selected limit, detached completion object, ambient completeness principle, countable-choice selector, or alternate completion transport.

Lean4 ✓ `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorUp StdBridge`

Proof. Start from the carrier and classifier of Definition 1116.1 and Definition 1116.2. They make $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$ the only rows visible to a standard-reflection read.

Apply the NameCert obligation surface of Definition 1116.3. Its obligation rows expose the source-target seal route, finite observations, Real-completion consumer boundary, structural transport, provenance, and local naming surface for the same accepted packet.

Next use Definition 1116.4, Definition 1116.5, and Definition 1116.6. These identify the route through S, M, T, the windows W_0, W_1, readback R, tolerance ledger D, classifier row K, endpoint E, and the structural rows H, C, P, N.

Finally apply Definition 1116.7, Definition 1116.8, and Definition 1116.9. They keep Real-completion consumption inside the primitive-scoped public seal route. The bridge-scope coherence theorem then makes any standard-reflection read preserve that route without adding quotient equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, or a second transport. □

Theorem 1116.14 (Cauchy limit functor terminal endpoint non-escape)**.** *For any accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}$ *endpoint row E is readable only after the target seal T, the transported windows W_0, W_1, the regular-rational readback R, the dyadic ledger D, the completion classifier K, and the structural rows H, C, P, N have been displayed through the same finite route. No consumer may detach E as host real equality, a selected limit, ambient completeness, a countable-choice selector, or a second Real-facing route.*

Proof. Start with the carrier of Definition 1116.1. The endpoint row E appears only inside the packet

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N),$$

so no terminal read exists before the surrounding seal, window, readback, ledger, classifier, and structural rows are present.

Apply terminal-window fusion, Definition 1116.10, and Real-seal naturality, Definition 1116.11. These keep the endpoint route tied to the transported windows, regular-rational readback, dyadic ledger, completion classifier, target seal, and structural replay rows.

Finally use the standard-reflection bridge surface, Definition 1116.13. Its non-escape boundary excludes quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, and any second Real-facing route, so the endpoint row remains a terminal packet coordinate rather than an external Real oracle. □

Theorem 1116.15 (Cauchy limit functor seal route totality)**.** *Every accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet with carrier*

$$F = (S, T, M, W_0, W_1, R, D, K, E, H, C, P, N)$$

exposes its public route only through the displayed source seal S, target seal T, map row M, $\mathsf{StreamName}^{\uparrow}$ *finite-window rows W_0, W_1,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{Real}^{\uparrow}$ *endpoint row E, the provenance row* Pkg*, the replay row* Cont*, the componentwise*

transport row hsame*, and the local row* $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}^{\uparrow}}$*. No accepted packet admits a public route through quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, or any second transport row.*

Proof. Start with the source-target seal exposure of Definition 1116.4. It makes S, M, and T the displayed route from the source $\mathsf{CauchyLimitSeal}^{\uparrow}$ side to the target seal side.

Next apply finite observation exposure, Definition 1116.5. This inserts the $\mathsf{StreamName}^{\uparrow}$ windows W_0, W_1, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, the classifier row K, and the terminal $\mathsf{Real}^{\uparrow}$ endpoint E into the same route.

Finally use the primitive scope package, Definition 1116.8, together with the carrier and classifier definitions. The remaining visible rows are exactly the structural transport, displayed replay, provenance, and local naming rows H, C, P, N, so every public route is exhausted by the listed Pkg, Cont, hsame, and NameCert data. □

Theorem 1116.16 (Cauchy limit functor finite-window naturality)**.** *Let two accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packets share the same source seal row, target seal row, and map row inside the displayed carrier. If their* $\mathsf{StreamName}^{\uparrow}$ *finite windows are* hsame*-compatible and are transported by that same map row, then the two packets induce the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route and the same* $\mathsf{Real}^{\uparrow}$ *terminal seal route through the displayed rows* $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$*.*

Proof. Begin with structural transport, Definition 1116.6. The shared map row M and the hsame row H make the two finite-window reads componentwise compatible inside the carrier.

Apply finite observation exposure. Since the windows are transported by the same map row, the readback row R and tolerance row D are read from the same finite observation surface, and the classifier row K sees no additional coordinate.

The source-target seal exposure then carries this common finite observation route to the same target seal row and terminal $\mathsf{Real}^{\uparrow}$ endpoint. Carrier-coordinate exhaustion rules out a second route, so the induced $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal routes agree. □

Theorem 1116.17 (Cauchy limit functor completion consumer exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *output factors through the source-target seal route and the bridge-scope coherence route before reading any terminal seal. Such a consumer may read* $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$*, but it cannot bypass the public seal route by introducing quotient stream equality, host real equality, a selected limit, a detached completion object, ambient completeness, countable choice, or an alternate completion transport.*

Proof. Use the Real-completion consumer boundary, Definition 1116.7. It confines every $\mathsf{Real}^{\uparrow}$ or completion-facing read to the finite route through the source seal, target seal, map row, finite windows, readback, tolerance ledger, classifier row, and terminal endpoint.

Next apply the public seal export, Definition 1116.9. It packages the same route with the structural transport, replay, provenance, and local NameCert rows, making those rows the only reusable surface for downstream consumers.

Finally apply bridge-scope coherence, Definition 1116.12. Any terminal seal read must preserve the bridge-scoped route already exported by the public surface, so completion-facing exactness follows without adding a second completion route. □

Theorem 1116.18 (Cauchy limit functor identity consumer)**.** *Suppose an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet has the same displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row as source and target, and suppose its transport row is the identity replay over the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* RealCompletionSealClassifierUp

and $\mathsf{Real}^{\uparrow}$ *rows. Then every identity consumer of the packet still reads only the finite route isolated by Definition 1116.13.*

In particular, the identity consumer may reuse the displayed rows

$$S, T, M, W_0, W_1, R, D, K, E, H, C, P, N,$$

but it exports no quotient stream equality, host real equality, selected limit, detached completion object, ambient completeness principle, countable-choice selector, or alternate completion transport.

Proof. Start with the carrier and classifier of Definition 1116.1 and Definition 1116.2. The identity assumption identifies the source and target seal rows inside the already displayed carrier; it does not create a new carrier coordinate or a second classifier read.

Apply Definition 1116.4, Definition 1116.5, and Definition 1116.6. The first keeps the route at the same $\mathsf{CauchyLimitSeal}^{\uparrow}$ row, the second keeps the observation through W_0, W_1, R, D, K, E, and the third records the identity replay through H, C, P, N.

Next use Definition 1116.7, Definition 1116.9, and Definition 1116.12. They keep any Real-completion read, reusable public export, and bridge-scope read inside the same primitive-scoped route.

Finally apply Definition 1116.13. Since the identity replay has no coordinate beyond the cited carrier rows and sibling certificates, the consumer reads exactly the finite route already isolated there and nothing outside it. □

Theorem 1116.19 (Cauchy limit functor identity transport)**.** *For an accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet whose transport row is the identity on a displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, the identity transport preserves the same source seal, target seal,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealCompletionSealClassifierUp}$ *row, terminal* $\mathsf{Real}^{\uparrow}$ *endpoint,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. It exports no second source-target route, quotient stream equality, selected limit, detached completion object, host real equality, ambient completeness principle, or countable-choice selector.*

Proof. Project the identity consumer carrier. Since the source and target $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows are the same displayed row, the identity transport has no independent source-target coordinate to expose.

Apply finite observation exposure and structural transport. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion-seal classifier row, and terminal $\mathsf{Real}^{\uparrow}$ endpoint are replayed through the existing H, C, P, N rows.

Finally use the public seal export and bridge-scope coherence. They keep every reusable consumer on the displayed route, so the identity transport preserves the listed rows and cannot introduce a second completion route or any excluded host principle. □

Theorem 1116.20 (Cauchy limit functor composite consumer)**.** *Let two accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packets be composable in the sense that the middle target seal row of the first packet and the middle source seal row of the second packet agree componentwise, and their shared observation rows agree through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCompletionSealClassifierUp}$*, and* $\mathsf{Real}^{\uparrow}$ *certificates. Then the composite consumer route is the concatenation of the two finite routes displayed by Definition 1116.13.*

The composite consumer exports only this concatenated finite route through the two packets' rows $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$*. It exports no quotient stream equality, selected limit, ambient completeness principle, countable-choice selector, alternate completion transport, or host-level equality between the two completion presentations.*

Proof. Use the carrier and classifier definitions for both packets. Componentwise agreement of the middle seal and observation rows lets the second route begin exactly where the first route ends, while each packet still exposes only its own displayed carrier rows.

Apply Definition 1116.4 and Definition 1116.5 to each packet. These two results identify the composable seal route and the shared finite observation surface without adding a quotient or selected limiting object at the middle row.

Next apply Definition 1116.6, Definition 1116.7, and Definition 1116.9. The structural row supplies the componentwise replay, the Real-completion boundary keeps both reads inside the displayed certificates, and the public export gives the reusable surface for concatenation.

Finally use Definition 1116.12 and Definition 1116.13. Bridge-scope coherence prevents a second route at the shared middle row, and the standard-reflection bridge surface identifies the composite as the two finite routes placed end to end. □

Theorem 1116.21 (Cauchy limit functor transport composition)**.** *Let two accepted* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packets transport a displayed source seal to an intermediate seal and then from that intermediate seal to a displayed target seal. Suppose their shared intermediate seal rows agree componentwise, and suppose the displayed finite observation rows are aligned through the two packets'* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RealCompletionSealClassifierUp}$ *classifier rows, and terminal* $\mathsf{Real}^{\uparrow}$ *endpoint rows.*

Then the composite consumer factors through the concatenated displayed rows

$$S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$$

of the first packet and the corresponding displayed rows of the second packet, yielding one target $\mathsf{Real}^{\uparrow}$ *completion seal. The transport composition exports no quotient stream equality, selected representative, host real equality, ambient completeness principle, countable-choice selector, or detached completion object.*

Proof. Expose the source and target seals for both packets by Definition 1116.4. The first packet supplies the source-to-intermediate route, and the second packet supplies the intermediate-to-target route; the componentwise agreement hypothesis identifies the middle seal as a displayed handoff row rather than a quotient or selected representative.

Next apply finite observation exposure to both packets. This aligns the source and target windows, regular readbacks, dyadic tolerance ledgers, classifier rows, and Real endpoint rows before any terminal seal is read.

Use structural transport for the H, C, P rows. Componentwise hsame transport preserves the displayed rows, Cont replay keeps the consumer on those rows, and Pkg provenance records that the same local packets are being concatenated.

Finally apply the composite consumer theorem. Its concatenated route is exactly the two displayed transport packets placed end to end, so the terminal read is a single target $\mathsf{Real}^{\uparrow}$ completion seal and none of the excluded host or choice principles has a carrier coordinate from which to be exported. □

Theorem 1116.22 (Cauchy limit functor induction route)**.** *Let S be a finite source* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, let T be a finite target* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, and let M be a displayed transport map between their regular Cauchy data. Suppose the source and target rows expose the same finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback discipline,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealCompletionSealClassifierUp}$ *boundary row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal as rows of a single* BHist

packet with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row.*

Then induction on the displayed $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite window constructs a routed* $\mathsf{CauchyLimitFunctor}^{\uparrow}$ *packet from* S, T, M*. The route preserves the source seal, target seal, transport map, finite windows, regular readback, dyadic tolerance ledger, completion-seal classifier, Real endpoint, transport row, continuation row, provenance row, and local naming row, and it exports no quotient stream equality, selected limit, ambient completeness principle, host real equality, or second transport route.*

Proof. Begin with the base window. The empty or initial displayed $\mathsf{StreamName}^{\uparrow}$ window carries only the source seal, target seal, transport-map row, and dyadic tolerance ledger already present in the BHist packet. The classifier row from RealCompletionSealClassifierUp is read as a boundary row, not as a selected limit.

For the successor window, assume the routed packet has been built for the current finite prefix. Extend the $\mathsf{StreamName}^{\uparrow}$ row by one displayed observation and extend the $\mathsf{RegSeqRat}^{\uparrow}$ readback by the matching regular rational datum. The dyadic ledger supplies the tolerance comparison for this single step, and the transport map M carries the source observation to the target observation inside the same continuation route.

At each induction step, componentwise hsame transport preserves the visible rows and the Cont replay records the ordered handoff from S through M to T. The Pkg row records that the construction uses only $\mathsf{CauchyLimitSeal}^{\uparrow}$, RealCompletionSealClassifierUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

After the finite window is exhausted, the local $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}^{\uparrow}}$ row names exactly the routed packet. No induction clause introduces quotient equality of streams, a selected limit, host completeness, host real equality, or an alternate route, so the constructed row is the required finite $\mathsf{CauchyLimitFunctor}^{\uparrow}$ route. □

Theorem 1116.23 (Cauchy limit functor Real-completion handoff exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-and-completion consumer using either the identity route of Definition 1116.18 or the composite route of Definition 1116.20 factors through the same displayed rows*

$$S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$$

and through the sibling certificates $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* RealCompletionSealClassifierUp*, and* $\mathsf{Real}^{\uparrow}$ *already cited in this chapter.*

Consequently the handoff is exhausted by the primitive-scoped bridge surface: it has no extra output for quotient stream equality, host real equality, selected limits, detached completion objects, ambient completeness, countable choice, or alternate completion transport.

Proof. Begin with the carrier and classifier definitions. They make $S, T, M, W_0, W_1, R, D, K, E, H, C, P, N$ the complete coordinate list for any accepted packet, so an identity or composite handoff has no independent row to read.

Apply Definition 1116.4, Definition 1116.5, and Definition 1116.6. These locate the seal, observation, and replay parts of the handoff inside the cited sibling certificates and structural rows.

Next use Definition 1116.7, Definition 1116.9, and Definition 1116.12. They confine every Real-and-completion consumer to the public bridge-scope route and exclude detached completion data from the handoff.

Finally apply Definition 1116.13, together with the identity and composite consumer theorems above. The only available handoff is the identity route or the concatenated finite route, and both are exhausted by the same primitive-scoped rows and sibling certificates. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}\uparrow}$.

Constructive story: A $\mathsf{CauchyLimitFunctor}^{\uparrow}$ packet is generated from source and target CauchyLimitSeal rows, a finite transport-map row, source and target StreamName windows, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealCompletionSealClassifier row, a terminal RealUp endpoint, componentwise transports, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1116.1, Definition 1116.2, Definition 1116.3, Definition 1116.4, Definition 1116.5, Definition 1116.6, Definition 1116.7, Definition 1116.8, Definition 1116.9, Definition 1116.12, Definition 1116.13, Definition 1116.15, Definition 1116.16, Definition 1116.17, Definition 1116.18, Definition 1116.20, and Definition 1116.23 bind the finite transport carrier, classifier, five NameCert obligation rows, separate seal-route exposure, finite observation exposure, structural transport, Real-completion consumer boundary, primitive scope package, reusable public transport surface, bridge-scope route coherence, standard-reflection bridge surface, seal-route totality, finite-window naturality, completion consumer exactness, identity consumer, composite consumer, and Real-completion handoff exhaustion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitFunctor}\uparrow}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CauchyLimitFunctorUp.CauchyLimitFunctorCarrier\_public\_seal\_export`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, host real equality, selected limits, ambient completeness, countable choice, or a universal completion functor.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 1117

A Concrete Naming Certificate for $\mathsf{CauchyWitnessPullback}^{\uparrow}$

The $\mathsf{CauchyWitnessPullback}^{\uparrow}$ packet realizes the inverse readback side of the Real-completion witness route of chapter 9238. It is paired with the forward gluing packet of chapter 1110 and reads back through the witness ledger of chapter 902, the tail-witness envelope of chapter 898, the seal-budget synchronizer of chapter 886, and the Real-completion seal classifier of chapter 900. The packet also records the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows needed by a completion-facing consumer that must recover which witness ledger supported a glued seal.

Definition 1117.1 (CauchyWitnessPullback carrier)**.** *A* $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (G, K, R, L, T, S, U, V, D, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the accepted* $\mathsf{CauchyWitnessGluing}^{\uparrow}$ *row whose Real-facing seal is being read back;*
2. *K is the* $\mathsf{RealCompletionSealClassifierUp}$ *row that recognizes the glued completion seal;*
3. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row served by the glued packet;*
4. *L is the recovered* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row;*
5. *T is the recovered* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$ *row;*
6. *S is the recovered* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row;*
7. *U is the* $\mathsf{StreamName}^{\uparrow}$ *row visible at the recovered tail window;*
8. *V is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reached by reading the recovered stream window;*
9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the recovered stream and regular-sequence rows;*
10. *A is the pullback agreement row saying that G, K, R determine the displayed L, T, S, U, V, D readback packet;*
11. *H records componentwise* hsame *transports for all displayed rows;*
12. *C records the* Cont *routes from the glued seal to the classifier, from the classifier to the recovered ledger, from the ledger through tail and synchronizer rows, and from the stream and dyadic rows to the regular-sequence handoff;*

13. *P is the* Pkg *provenance over chapter 1110, chapter 902, chapter 898, chapter 886, chapter 900, chapter 126, chapter 301, chapter 291, chapter 38,* hsame, Cont, *and* NameCert *rows;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessPullback}^{\uparrow}}$ *row.*

The carrier exports no quotient stream equality, host real equality, detached external limit, countable-choice row, ambient completeness principle, or untracked witness replacement beyond the displayed pullback packet.

Lean4 ✓ `BEDC.Derived.CauchyWitnessPullbackUp.TasteGate.CauchyWitnessPullbackTasteGate single carrier alignment`

Theorem 1117.2 (CauchyWitnessPullback NameCert obligations)**.** *Let*

$$Q = (G, K, R, L, T, S, U, V, D, A, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet. Every consumer that preserves the glued seal G, completion classifier K, Real seal R, recovered ledger L, tail envelope T, synchronizer S, stream row U, regular-sequence row V, dyadic ledger D, pullback agreement A, transports H, continuations C, provenance P, and local name N preserves the same witness-readback packet. A consumer that preserves G, K, R but changes L is not a preservation of this carrier.*

Proof. Project the carrier of Definition 1117.1. The displayed rows are the whole finite packet, and the agreement row A is the coordinate that binds the glued seal to the recovered witness side.

The readback first passes through G, K, R. These rows identify the Real-facing seal and the classifier that accepts it, so a consumer cannot choose a second completion-seal source without changing the packet.

The readback then reaches L, T, S, U, V, D. These rows record which witness ledger, tail envelope, synchronizer, stream window, regular-sequence handoff, and dyadic tolerance ledger support the accepted seal.

Finally H, C, P, N make the readback a NameCert surface rather than an ambient equality assertion. Thus preserving all displayed rows preserves the packet, while changing L changes the coordinate that the agreement row recovers from the glued seal. □

Theorem 1117.3 (CauchyWitnessPullback Real seal factorization)**.** *Every completion-facing read of an accepted* $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet factors through*

$$G \to K \to R \to A \to (L, T, S, U, D, V)$$

with H, C, P, N visible as structural rows. No consumer can preserve the glued seal while replacing the recovered witness-ledger row, tail-envelope row, synchronizer row, StreamName row, RegSeqRat row, DyadicRatCore row, or RealUp seal row.

Proof. Start with the Real-facing side. The rows G, K, R present the accepted glued packet, the classifier that recognizes its completion seal, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply the pullback agreement row A. It records the inverse readback from that glued seal to the displayed witness-side packet, so the recovered rows are not optional annotations.

Read the recovered side. The ledger L, tail envelope T, synchronizer S, stream row U, dyadic ledger D, and regular-sequence row V are the displayed coordinates through which the Real seal is supported.

The structural rows H, C, P, N then repack these coordinates into the local NameCert surface. Replacing any recovered coordinate while keeping only the glued seal fixed therefore gives a different candidate packet, not the same $\mathsf{CauchyWitnessPullback}^{\uparrow}$ factorization. □

Lean4 ✓ `BEDC.Derived.CauchyWitnessPullbackUp.CauchyWitnessPullbackUp StdBridge`

Theorem 1117.4 (CauchyWitnessPullback recovered ledger determinacy)**.** *Let*

$$Q = (G, K, R, L, T, S, U, V, D, A, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet. Any second accepted pullback packet that agrees with* Q *on* G, K, R, A, H, C, P, N *has the same recovered witness-ledger row* L*. Thus the recovered ledger is determined by the glued seal, completion classifier, Real seal, pullback agreement, transport, continuation, provenance, and local NameCert rows.*

Lean4 ✓ `BEDC.Derived.CauchyWitnessPullbackUp.CauchyWitnessPullbackCarrier recovered ledger determinacy`

Proof. Project the carrier of Definition 1117.1. The row L is a displayed coordinate of the same finite packet as G, K, R, A, H, C, P, N, not an ambient ledger chosen after the seal has been read.

Apply Definition 1117.2. A consumer that preserves the glued seal, classifier, Real seal, agreement, transports, continuations, provenance, and local name but changes L fails to preserve the carrier.

Finally use Definition 1117.3. The Real-facing read passes through $G \to K \to R \to A$ before it reaches the recovered witness side, so agreement on those structural rows fixes the ledger coordinate recovered by the pullback packet. □

Theorem 1117.5 (CauchyWitnessPullback tail synchronizer compatibility)**.** *For every accepted* $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet*

$$Q = (G, K, R, L, T, S, U, V, D, A, H, C, P, N),$$

the recovered rows T, S, U, V, D *remain on the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *route that the agreement row reads back from* G, K, R*. In particular, a consumer cannot keep* $G, K, R, L, A, H, C, P, N$ *fixed while moving the recovered tail envelope, synchronizer, stream row, regular-sequence row, or dyadic ledger to an unrelated route.*

Proof. Read the displayed coordinates in Definition 1117.1. The tail envelope T, synchronizer S, stream row U, regular-sequence row V, and dyadic ledger D are named together with the recovered ledger L and the agreement row A.

Apply Definition 1117.2. Preserving the pullback carrier preserves the same recovered witness-readback packet, including the rows that route from the tail envelope through synchronizer, stream, regular-sequence, and dyadic data.

Use Definition 1117.3. Since the Real seal factors through the displayed recovered side, replacing any one of T, S, U, V, D by an unrelated route while keeping the structural rows fixed leaves the $\mathsf{CauchyWitnessPullback}^{\uparrow}$ factorization. □

Theorem 1117.6 (CauchyWitnessPullback Real-seal readback exactness)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet receives exactly the rows*

$$G, K, R, L, T, S, U, V, D, A, H, C, P, N$$

as its readback surface. The readback exports no quotient stream, selected limit, choice row, ambient completeness datum, host real equality, or hidden witness replacement.

Proof. Start from Definition 1117.1. The carrier lists precisely the fourteen displayed rows and explicitly excludes quotient stream equality, detached limits, choice rows, ambient completeness, and untracked witness replacement.

Apply Definition 1117.2. The local NameCert surface is preserved only by preserving those displayed coordinates together, so a completion-facing consumer has no additional row from which to read a selected limit or hidden witness.

Finally use Definition 1117.3. The Real-seal read passes through the glued seal, classifier, Real seal, agreement, and recovered witness-side rows, with H, C, P, N visible as structural rows. That factorization is exactly the exported readback surface. □

Theorem 1117.7 (CauchyWitnessPullback scoped primitive dependency binding)**.** *Let*

$$Q = (G, K, R, L, T, S, U, V, D, A, H, C, P, N)$$

be an accepted $\mathsf{CauchyWitnessPullback}^{\uparrow}$ *packet. The readback surface of* Q *is scoped exactly by the displayed* BHist *packet, the componentwise* hsame *transports* H*, the displayed* Cont *routes* C*, the* Pkg *provenance* P*, the local* $\mathsf{NameCert}_{\mathsf{CauchyWitnessPullback}^{\uparrow}}$ *row* N*, and the sibling certificate rows cited by the carrier:* $\mathsf{CauchyWitnessGluing}^{\uparrow}$*,* $\mathsf{CauchyWitnessLedger}^{\uparrow}$*,* $\mathsf{CauchyTailWitnessEnvelope}^{\uparrow}$*,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$*,* $\mathsf{RealCompletionSealClassifierUp}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. No quotient stream, selected limit, host real equality, choice row, or ambient completeness datum lies in this scoped primitive dependency surface.*

Proof. Project the carrier of Definition 1117.1. Its fourteen displayed coordinates are the whole accepted BHist packet, and the rows H, C, P, N are already part of the carrier rather than external annotations.

Apply Definition 1117.2. The local NameCert obligations force preservation of $G, K, R, L, T, S, U, V, D, A, H, C,$ together, so a consumer cannot keep the pullback packet while replacing any cited sibling certificate row.

Next use the factorization theorem Definition 1117.3. It routes every completion-facing read through $G \to K \to R \to A$ before the recovered ledger, tail, synchronizer, stream, dyadic, and regular-sequence rows are consumed. Thus the displayed Cont replay and Pkg provenance are the only admissible dependency route from the glued seal to the recovered witness side.

Apply recovered-ledger determinacy from Definition 1117.4 and tail-synchronizer compatibility from Definition 1117.5. These results keep the ledger, tail envelope, synchronizer, stream row, regular-sequence row, and dyadic ledger on the same sibling-certified route.

Finally apply readback exactness from Definition 1117.6. Since the accepted packet exports exactly the displayed rows, quotient streams, selected limits, host real equality, choice rows, and ambient completeness remain outside the scoped primitive dependency boundary. □

The Lean TasteGate namespace $\mathsf{BEDC.Derived.CauchyWitnessPullbackUp.TasteGate}$ provides explicit $\mathsf{Nontrivial}\ \mathsf{CauchyWitnessPullback}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CauchyWitnessPullback}^{\uparrow}$ instances for the displayed carrier, with the single-carrier alignment target named $\mathsf{BEDC.Derived.CauchyWitnessPullbackUp.Taste}$

Falsifiable prediction: If the pullback packet is accepted, a Real-completion consumer preserving the glued seal but changing the witness-ledger row is rejected by the classifier.

Independence witness: cauchywitnessgluing, cauchywitnessledger, cauchytailwitnessenvelope, cauchysealbudgetsynchronizer, realcompletionsealclassifier: no bijection preserves all pullback rows because the pullback packet carries the glued seal, classifier, Real seal, recovered ledger, recovered

tail, recovered synchronizer, StreamName, RegSeqRat, DyadicRatCore, agreement, transport, continuation, provenance, and local NameCert rows together, while each sibling exposes only one direction or one face of that readback.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWitnessPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyWitnessPullback}^{\uparrow}$ packet is generated from a glued seal row, completion classifier, Real seal, recovered witness ledger, recovered tail envelope, recovered synchronizer, StreamName row, RegSeqRat row, DyadicRatCore row, agreement row, componentwise hsame transport, displayed Cont replay, provenance, and local naming. The carrier reads a Real-completion seal backwards through the witness data that produced it: the classifier exposes the recovered ledger, envelope, and synchronizer together. The agreement and transport rows keep that recovery tied to the same seal. A consumer preserving the glued seal but changing the witness ledger leaves the displayed pullback route.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWitnessPullback}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1117.1, Definition 1117.2, Definition 1117.3, Definition 1117.4, Definition 1117.5, Definition 1117.6, Definition 1117.7, and

Lean4 ✓ `BEDC.Derived.CauchyWitnessPullbackUp.taste gate`

bind the Cauchy witness pullback carrier, recovered-ledger determinacy, tail synchronizer compatibility, Real-seal readback exactness, scoped primitive dependency surface, prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWitnessPullback}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyWitnessPullbackUp.TasteGate.CauchyWitnessPullbackTasteGate\_single\_carrier\_alig`

Bridge status: none

Origin: ai

Not claimed: It does not claim arbitrary pullback of completed reals, quotient sequence equality, hidden witness recovery, or ambient completeness.

Upgrade path: To reach publicClosure, provide the public certificate export theorem for standard reflection consumers of the checked pullback carrier.

Chapter 1118

A Concrete Naming Certificate for $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$

The $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ packet names the two-route diamond that compares displayed Zeckendorf carry-normalization paths. It is adjacent to the finite confluence surface of chapter 713, but its local role is the route certificate itself: two source carry windows, two route rows, one shared normal target row, value ledgers, non-hsame boundary rows, componentwise transports, continuation replay, provenance, and a local naming row are carried together as a finite packet.

It is an ai over the $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$, $\mathsf{ZeckendorfCarryValue}^{\uparrow}$, and $\mathsf{AxisZeckendorfCarryBudg}$ sibling surfaces, with the diamond row acting only as the finite shared-target route bundle.

Falsifiable prediction: Two accepted diamond packets with the same two displayed source carry windows and same shared target row cannot expose two different target-normality ledgers.

Independence witness: zeckendorfcarryclassifier, zeckendorfcarrynormalization, zcarry, zeckendorfcarryvalue, axiszeckendorfcarrybudget: the diamond carrier has two route rows and a shared-target ledger absent from each sibling, while each sibling lacks at least one of the paired-route or shared-normal-target coordinates.

The packet consumes $\mathsf{ZCarry}^{\uparrow}$ edges from chapter 544, the normalization handoff of chapter 608, the value row of chapter 606, and the finite budget surface of chapter 692. It does not identify the carry sources by hsame and does not promote the diamond to a kernel rewrite.

Definition 1118.1 (Axis carry diamond-route carrier)**.** *An* $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N)$$

with the following displayed rows:

1. *a is the displayed overlap source row for the diamond;*
2. *b and c are the two displayed carry-successor source windows;*
3. *n is the shared normal target row admitted by* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$*;*
4. *ρ_0 is the first route row, consisting of a* $\mathsf{ZCarry}^{\uparrow}$ *edge from a to b and a* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *handoff from b to n;*
5. *ρ_1 is the second route row, consisting of a* $\mathsf{ZCarry}^{\uparrow}$ *edge from a to c and a* $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ *handoff from c to n;*
6. *ν_0 and ν_1 are the displayed finite value ledgers for the two routes;*

7. *η is the shared target-normality ledger for n;*
8. *H records the non-*hsame *carry boundaries together with componentwise transports for all displayed route, value, target, continuation, provenance, and naming rows;*
9. *C records displayed* Cont *routes for replaying the two route rows, both value ledgers, the shared target-normality row, and the local naming row;*
10. *P records* Pkg *provenance over* $\mathsf{ZCarry}^{\uparrow}$, $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$, $\mathsf{ZeckendorfCarryValue}^{\uparrow}$, $\mathsf{AxisZeckendorfCarryBudget}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AxisCarryDiamondRoute}^{\uparrow}}$ *row.*

The carrier exports only this finite two-route packet. It contains no source-target hsame *proof, no host arithmetic quotient, no global confluence theorem, no hidden normalizer, and no kernel rewrite rule.*

Lean4 (definition) `BEDC.Derived.AxisCarryDiamondRouteUp.AxisCarryDiamondRouteUp`

Lean4 ✓ `BEDC.Derived.AxisCarryDiamondRouteUp.AxisCarryDiamondRouteTasteGate single carrier alignment`

Theorem 1118.2 (Axis carry diamond routes name two-step confluence)**.** *For an accepted* AxisCarryDiamondRoute *carrier $D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N)$, the route rows ρ_0 and ρ_1 name two displayed carry-normalization paths from the same overlap source row a to the same normal target row n. The shared target-normality ledger η is read only after both route rows and both value ledgers have been exposed, so any diamond consumer receives a finite two-step confluence packet rather than two unrelated carry edges.*

Proof. Project the accepted carrier. The rows ρ_0 and ρ_1 are the only route coordinates, and their endpoints are fixed by the displayed rows a, b, c, n.

Read the normal target and value rows next. The target coordinate n is shared, while ν_0 and ν_1 record the finite value ledgers along the two displayed paths.

Finally replay the support rows. The continuation and provenance rows expose the two routes and the shared target ledger as one packet, and the local NameCert row names that packet before a consumer may cite the diamond. □

Lean4 ✓ `BEDC.Derived.AxisCarryDiamondRouteUp.AxisCarryDiamondRouteCarrier names two step confluence`

Theorem 1118.3 (Axis carry diamond-route non-escape)**.** *Every accepted* $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ *read factors through the displayed packet*

$$D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N).$$

*It can expose the two carry-normalization route rows, the shared normal target row, the two value ledgers, the shared target-normality ledger, non-*hsame *carry boundaries, componentwise transports, route replay, provenance, and local naming. It cannot expose source-target* hsame*, host arithmetic equality, a quotient normal form, an unbounded normalizer, a global confluence theorem, or a kernel rewrite.*

Proof. Begin with Definition 1118.1. The displayed tuple lists every coordinate available to an accepted read, and none of the excluded objects is a coordinate.

The route coordinates are finite. Each route row is assembled from the $\mathsf{ZCarry}^{\uparrow}$ and ZeckendorfCarryNormalizatio sibling surfaces, while the value and target ledgers are displayed as ν_0, ν_1, η rather than as host arithmetic or quotient data.

The support rows H, C, P, N preserve and replay only those displayed coordinates. Since the carrier contains no hidden normalizer, global confluence row, source-target hsame proof, or kernel rewrite rule, no accepted projection can escape the finite diamond-route packet. □

Lean4 ✓ `BEDC.Derived.AxisCarryDiamondRouteUp.AxisCarryDiamondRouteNonescape`

The Lean taste-gate file for this carrier should provide explicit `Nontrivial AxisCarryDiamondRouteUp` and `FieldFaithful AxisCarryDiamondRouteUp` instances for the displayed row inventory.

Theorem 1118.4 (Axis carry diamond-route obligation surface)**.** *For every accepted* AxisCarryDiamondRoute$^{\uparrow}$ *carrier*

$$D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N),$$

*the local obligation surface consists exactly of carrier admission for the displayed rows, classifier stability over those same rows, route exposure for ρ_0, ρ_1, shared-target normality for η, componentwise value-ledger transport for ν_0, ν_1, non-*hsame *boundary preservation in H, continuation replay in C, package provenance in P, and local naming in N. No obligation is stored outside the finite tuple.*

Proof. Project the carrier of Definition 1118.1. The rows $a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N$ are the complete row inventory.

Apply the two-step confluence and non-escape theorems. They force every route, target, value, transport, replay, provenance, and naming read to remain inside that inventory.

Thus the admission and classifier obligations range over exactly the displayed packet, while the excluded host equality, quotient, hidden normalizer, global confluence, and kernel rewrite objects have no carrier coordinate. □

Theorem 1118.5 (Axis carry diamond-route target-ledger exactness)**.** *For an accepted* AxisCarryDiamondRoute$^{\uparrow}$ *carrier $D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N)$, the shared target-normality ledger η may be read only after both route rows ρ_0, ρ_1 and both value ledgers ν_0, ν_1 have been displayed. A consumer that reads η while omitting either route row or either value ledger is not reading the accepted diamond-route packet.*

Proof. Use Definition 1118.2. It places η downstream of the two route rows and the two value ledgers in the public diamond handoff.

Then apply non-escape. Since the carrier has no second target-normality coordinate and no hidden normalizer, η cannot be detached from $\rho_0, \rho_1, \nu_0, \nu_1, H, C, P, N$. □

Theorem 1118.6 (Axis carry diamond-route value-ledger stability)**.** *Let $D = (a, b, c, n, \rho_0, \rho_1, \nu_0, \nu_1, \eta, H, C, P, N)$ and $D' = (a', b', c', n', \rho'_0, \rho'_1, \nu'_0, \nu'_1, \eta', H', C', P', N')$ be accepted* AxisCarryDiamondRoute$^{\uparrow}$ *carriers related componentwise by the displayed transport and continuation rows. Then the value ledgers ν_0, ν_1 are stable routewise: the first ledger is transported only along the ρ_0 route, the second ledger only along the ρ_1 route, and both remain tied to the shared target-normality row before any diamond consumer can use them.*

Proof. Project both carrier tuples. The only value-ledger coordinates are ν_0 and ν_1, and the only route coordinates governing them are ρ_0 and ρ_1.

Use the transport rows H, H' and continuation rows C, C'. They replay the componentwise route and value data without adding a cross-route ledger or a host arithmetic equality.

Finally apply target-ledger exactness and non-escape. The ledgers stay attached to their route rows and the shared target ledger, so a consumer cannot swap, merge, or replace them by quotient normal forms, global carry confluence, or kernel rewriting. □

Theorem 1118.7 (Axis carry diamond-route scoped kernel dependencies)**.** *Every accepted* AxisCarryDiamondRoute *two-route diamond packet factors through the displayed rows*

$$\mathsf{ZCarry}^{\uparrow}, \mathsf{ZeckendorfCarryNormalization}^{\uparrow}, \mathsf{ZeckendorfCarryValue}^{\uparrow}, \mathsf{AxisZeckendorfCarryBudget}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}$$

before any consumer may read the shared target-normality ledger or the local diamond name. The scoped route exposes the two carry-successor rows, two normalization routes, two value ledgers, one shared normal target, componentwise transport, replay, provenance, and local naming.

It exposes no host arithmetic quotient, no global confluence theorem, no source-target hsame *proof, and no kernel rewrite rule.*

Proof. Start with the carrier and route theorems. Definition 1118.1 lists the finite row inventory, while Definition 1118.2 places the two $\mathsf{ZCarry}^{\uparrow}$ edges, the two $\mathsf{ZeckendorfCarryNormalization}^{\uparrow}$ handoffs, the shared normal target, and the value ledgers in one displayed packet.

Next apply non-escape and the obligation surface. Definition 1118.3 removes host arithmetic quotients, hidden normalizers, global confluence, and kernel rewriting from the accepted read, and Definition 1118.4 confines the local obligations to BHist, hsame, Cont, Pkg, and NameCert rows.

Finally use target-ledger exactness and value-ledger stability. They keep the shared target-normality ledger downstream of both routes and both value ledgers, so the scoped packet can be consumed only through the named kernel dependencies and never through a source-target hsame shortcut or an ambient rewrite rule. □

Closure status for $\mathsf{NameCert}_{\mathsf{AxisCarryDiamondRoute}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ packet is generated from an overlap source row, two carry-successor rows, a shared normal target, two carry-normalization route rows, two value ledgers, a shared target-normality ledger, non-hsame boundary rows, componentwise transports, continuation routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AxisCarryDiamondRoute}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1118.1, Definition 1118.2, Definition 1118.3, Definition 1118.4, Definition 1118.5, Definition 1118.6, and Definition 1118.7 specify the finite two-route diamond carrier, obligation rows, shared target ledger, value-ledger stability, non-escape boundary, and scoped dependencies over ZCarry, ZeckendorfCarryNormalization, ZeckendorfCarryValue, AxisZeckendorfCarryBudget, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AxisCarryDiamondRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary carry confluence, host arithmetic quotienting, global normalization, source-target hsame, or kernel rewriting.

Upgrade path: Public closure requires a checked export obligation for paired route rows, shared target-normality, value-ledger exactness, non-hsame preservation, route replay, provenance, and local naming for the same finite packet.

Chapter 1119

A Concrete Naming Certificate for $\mathsf{FiniteTailFilter}^{\uparrow}$

The $\mathsf{FiniteTailFilter}^{\uparrow}$ packet is the finite-filter surface shared by the Real and Completion routes. It packages the cofinal-tail information that is read by the stream-name surface of chapter 126, the dyadic rational core of chapter 291, the regular rational sequence surface of chapter 301, the RealUp seal of chapter 38, the tail budget of chapter 379, and the common-tail comparison packet of chapter 387. The packet is the BEDC finite-filter analogue of the constructive Cauchy tail-filter interface: a consumer receives a finite set of admitted tail windows, the dyadic and regular-sequence rows attached to those windows, and the seal-facing route that preserves the same cofinal-tail packet.

Definition 1119.1 (FiniteTailFilter carrier)**.** *A* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

with the following visible rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger whose entries form the admitted tail filter base;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the admitted windows;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same admitted windows and dyadic ledger;*
4. *B is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *tail-budget row that records the finite threshold and tolerance data used by the filter;*
5. *Q is the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *common-tail comparison row read through the same window ledger and dyadic ledger;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the window, dyadic, regular-sequence, tail-budget, and common-tail comparison rows have all been displayed;*
7. *H records componentwise* hsame *transports for $S, D, R, B, Q, E, C, P, N$;*
8. *C records the displayed* Cont *routes for window readback, dyadic tolerance use, regular-rational readback, tail-budget consumption, common-tail comparison, and RealUp seal consumption;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $S, D, R, B, Q, E, H, C, P, N$ and their recorded read routes.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.TasteGate.FiniteTailFilterUpTasteGate single carrier alignment`

Theorem 1119.2 (FiniteTailFilter NameCert obligations)**.** *For the carrier of Definition 1119.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

window-filter exposure through S, dyadic tolerance exposure through D, regular-sequence readback through R, tail-budget exposure through B, common-tail comparison through Q, RealUp seal exposure through E, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier namecert obligations`

Proof. Project the displayed carrier. The rows $S, D, R, B, Q, E, H, C, P, N$ are the only coordinates of the finite packet, so carrier admission is the assertion that these rows are assembled as one BHist surface.

Read the analytic rows in order. The window ledger S supplies the finite filter base, the dyadic ledger D supplies the tolerance rows attached to that base, and the regular readback R is the regular rational interpretation of the same finite windows.

The tail-budget row B and the common-tail comparison row Q then read the same S, D, R packet rather than separate windows or separate tolerances. The RealUp seal E is exposed only after those rows have been displayed.

The structural rows H, C, P, N transport, replay, authenticate, and name the same packet. Hence the listed coordinates exhaust the local naming surface. □

Theorem 1119.3 (FiniteTailFilter non-escape)**.** *Every downstream consumer of an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet receives only the finite surface*

$$S, D, R, B, Q, E, H, C, P, N.$$

It receives no quotient stream equality, selected limit, host real equality, ambient completeness datum, metric-completeness datum, countable-choice row, or standard bridge row.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier nonescape`

Proof. Inspect the carrier of Definition 1119.1. The analytic coordinates are the finite window ledger, dyadic tolerance ledger, regular-sequence readback, tail-budget row, common-tail comparison row, and RealUp seal row. The remaining coordinates are transport, continuation, package provenance, and local naming rows for that same packet.

Apply the NameCert obligation theorem. Any consumer route must pass through the displayed coordinates and their recorded hsame and Cont witnesses.

None of the excluded objects is a coordinate of the carrier or a row certified by the local naming surface. Therefore an accepted consumer cannot recover quotient stream equality, a selected limit, host real equality, ambient completeness, metric-completeness data, countable choice, or a standard bridge row from the finite filter packet. □

Theorem 1119.4 (FiniteTailFilter tail-window exactness)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

the tail windows consumed by the $\mathsf{DyadicTailBound}^{\uparrow}$ *row* B*, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *row* Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E *are exactly the windows displayed in* S *with the dyadic tolerances displayed in* D*. No accepted read may replace the filter base by an unrecorded window selector, split the tail-budget row from the common-tail comparison row, or substitute a separate dyadic tolerance ledger.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier tail window exactness`

Proof. Project S and D from the carrier. These are the only coordinates that display tail windows and dyadic tolerances.

Use the NameCert obligations to locate B and Q. The tail-budget row and the common-tail comparison row are accepted only as rows of the same packet, and both are read after the S, D, R surface has been displayed.

The RealUp seal row E is downstream of that same packet. Its continuation route in C can replay the displayed rows, but it has no coordinate for a second filter base or a second tolerance ledger.

Thus the visible tail-window packet is exact: all accepted budget, comparison, and seal reads use S and D as displayed. □

Theorem 1119.5 (FiniteTailFilter Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

reaches the seal row E *only through the displayed route*

$$S, D, R, B, Q, H, C, P, N.$$

The handoff is therefore a finite cofinal-tail route shared by Real and Completion seal reads, not a quotient of streams, selected limit, host real equality, ambient completeness theorem, metric-completeness datum, countable-choice row, or standard bridge row.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier real seal handoff`

Proof. Start from the carrier. The seal coordinate E is stored in the same finite packet as the stream-window ledger S, dyadic tolerance ledger D, regular readback R, tail-budget row B, common-tail comparison row Q, and structural rows H, C, P, N.

Apply tail-window exactness. The tail-budget and common-tail comparison rows use the same S, D window packet, so a seal-facing read cannot detach the RealUp row from the cofinal-tail filter base.

Use the NameCert obligation theorem to read the remaining route data. The hsame rows in H, the Cont routes in C, the package row P, and the local naming row N certify the displayed handoff and no larger completion structure.

Finally apply non-escape. Since the carrier has no excluded coordinate, the Real and Completion consumers receive the finite route $S, D, R, B, Q, H, C, P, N$ before reaching E and nothing outside that route. □

Theorem 1119.6 (FiniteTailFilter Real-window consumer)**.** *Every* RealWindowBudget$^{\uparrow}$ *or Real-completion consumer reading an accepted* FiniteTailFilter$^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

receives the displayed finite cofinal-tail route

$$S, D, R, B, Q, E, H, C, P, N.$$

The Real-window budget row may use only the already displayed StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, CauchyTailComparison$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows. It receives no quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness theorem, metric-completeness datum, or unrecorded completion carrier.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier real window consumer`

Proof. Start from the NameCert obligations of Definition 1119.2. They expose the accepted packet exactly as $S, D, R, B, Q, E, H, C, P, N$, with S, D, R, B, Q, E as the Real-facing analytic rows and H, C, P, N as the structural rows.

Apply tail-window exactness, Definition 1119.4. The tail-budget row B, common-tail comparison row Q, and RealUp seal row E all read the same displayed S, D window and tolerance packet, so a Real-window budget consumer cannot introduce a separate selector or tolerance ledger.

Use the Real seal handoff theorem, Definition 1119.5, to identify the downstream route. Any Real or Completion read reaches E only through $S, D, R, B, Q, H, C, P, N$, and the present consumer includes E as the displayed seal row reached by that route.

Finally apply non-escape, Definition 1119.3. Since quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, and unrecorded completion carriers are not coordinates of F, the RealWindowBudget$^{\uparrow}$ or Real-completion consumer receives only the finite cofinal-tail packet already displayed. □

Theorem 1119.7 (FiniteTailFilter RealWindowBudget consumer link)**.** *Every* RealWindowBudget$^{\uparrow}$ *consumer of an accepted* FiniteTailFilter$^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

must reuse the displayed route as its finite request, StreamName$^{\uparrow}$ *window, dyadic readback,* RegSeqRat$^{\uparrow}$ *handoff, and* Real$^{\uparrow}$ *seal surface. It cannot introduce a private selector, separate dyadic ledger, selected limit, quotient stream, host equality, countable-choice row, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier real window budget consumer link`

Proof. Start with the FiniteTailFilter NameCert obligations from Definition 1119.2. They expose $S, D, R, B, Q, E, H, C, P, N$ as the complete accepted packet, with S, D, R, B, Q, E carrying the finite analytic route.

Apply tail-window exactness, Definition 1119.4. The tail-budget and comparison rows read the same finite window and dyadic tolerance data, so a RealWindowBudget consumer cannot replace S or D by a private selector or separate ledger.

Use the Real seal handoff and Real-window consumer theorems, Definition 1119.5 and Definition 1119.6. Together they force the consumer route through the displayed RegSeqRat readback R, tail budget B, common-tail comparison Q, and seal row E.

Finally apply the RealWindowBudget finite-route visibility supplied by the packet surface and the non-escape theorem, Definition 1119.3. The structural rows H, C, P, N preserve the same route, while the carrier has no coordinate for selected limits, quotient streams, host equality, countable choice, ambient completeness, or an unrecorded completion surface. □

Theorem 1119.8 (FiniteTailFilter Real-window selector bridge)**.** *Every* $\mathsf{RealWindowBudget}^{\uparrow}$ *selector-stability read that consumes an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

must factor through the displayed finite route $S, D, R, B, Q, E, H, C, P, N$*. Through this route it reads the* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, tail-budget row, common-tail row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation, package source, and local naming coordinate supplied by the packet.*

Consequently, by $\mathsf{RealWindowBudget}^{\uparrow}$ *selector stability, the downstream selector surface cannot replace that route by a private window family, detached dyadic ledger, selected limit, quotient stream, host real equality, countable-choice row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier real window budget selector bridge`

Proof. Start with the FiniteTailFilter NameCert obligations, Definition 1119.2. They expose the accepted packet as the ten displayed rows $S, D, R, B, Q, E, H, C, P, N$, with the analytic rows and structural rows carried by one local certificate.

Apply tail-window exactness, Definition 1119.4. The stream-window row S and dyadic ledger D are the common cofinal-tail packet used by the tail budget B, common-tail comparison Q, and the Real-facing seal row E.

Use the Real seal handoff and the RealWindowBudget consumer link, Definition 1119.5 and Definition 1119.7. These identify the downstream read as a consumer of the same $S, D, R, B, Q, E, H, C, P, N$ route, rather than a read of any hidden completion or host-level real structure.

Finally apply $\mathsf{RealWindowBudget}^{\uparrow}$ selector stability, Definition 1164.3. Once the read reaches the RealWindowBudget selector surface, that theorem keeps the selector synchronized with the finite request, window, dyadic, regular, Real-facing, and support rows supplied by the displayed route, so no private window family, detached ledger, selected limit, quotient stream, host equality, countable-choice row, or ambient completeness coordinate can replace the packet route. □

Theorem 1119.9 (FiniteTailFilter sibling tail handoff)**.** *Every consumer whose route passes through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicTailBound}^{\uparrow}$ *budget,* $\mathsf{CauchyTailComparison}^{\uparrow}$ *common-tail rows, or the* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed finite packet*

$$S, D, R, B, Q, E, H, C, P, N.$$

No sibling chapter may substitute its own tail selector, tolerance ledger, common-tail row, or seal row for the packet displayed by $\mathsf{FiniteTailFilter}^{\uparrow}$*.*

Proof. Start from the carrier of Definition 1119.1. The rows $S, D, R, B, Q, E, H, C, P, N$ are the whole finite packet, so any sibling read of a stream window, dyadic tolerance, regular readback, budget, comparison, or seal row has to be a read of one of those displayed coordinates.

Apply the NameCert obligations of Definition 1119.2. They expose the analytic rows and the structural rows as one local naming surface, with the hsame and Cont data recording how the same packet is replayed downstream.

Use tail-window exactness, Definition 1119.4, to identify the common window and tolerance packet consumed by the budget, comparison, and seal-facing rows. Then use the Real seal handoff theorem, Definition 1119.5, to carry the same packet to Real and Completion consumers.

Finally apply non-escape. The handoff is finite and has no coordinate for a quotient stream, selected limit, host real equality, ambient completeness datum, metric-completeness datum, countable-choice row, or standard bridge row. Hence no sibling route can replace the displayed tail selector, ledger, or seal row by an unrecorded one. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier sibling tail handoff`

Theorem 1119.10 (FiniteTailFilter shared-window meet)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet. Suppose an accepted* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *packet has its selected-window row and readback row consumed as the same displayed finite window and regular readback route used by* F*. Then every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer reads the shared L10 route through*

$$S, D, R, B, Q, E, H, C, P, N.$$

It cannot replace this meet by a hidden selector, second meet, quotient stream, selected limit, host real equality, countable-choice row, ambient completeness datum, or metric-completeness row.

Proof. First bind the common finite packet. Tail-window exactness, Definition 1119.4, identifies S and D as the displayed window and tolerance packet read by the tail-budget row, common-tail row, and Real seal of F.

Next route the sibling consumers. The sibling tail handoff and RealWindowBudget consumer link, Definition 1119.9 and Definition 1119.7, force StreamName, DyadicRatCore, RegSeqRat, RealWindowBudget, Real, and Completion reads through the displayed route $S, D, R, B, Q, E, H, C, P, N$.

Now attach the adjacent selected-window route. By Definition 885.130, an accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$ consumer whose selected-window and readback rows are the same displayed finite window and regular readback route cannot introduce a separate selector, second meet, or off-route Real seal.

Finally apply non-escape, Definition 1119.3. Since neither accepted packet contributes a coordinate for quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, or a hidden completion surface, the shared consumer route is exactly the finite packet displayed by F. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier shared window meet`

Theorem 1119.11 (FiniteTailFilter carrier transport obligation)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

the componentwise hsame *row* H *transports exactly the displayed packet rows* $S, D, R, B, Q, E, C, P, N$*. It has no coordinate that can introduce quotient stream equality, host real equality, selected limits, choice data, completeness data, or standard bridge rows.*

Proof. Project the carrier and isolate the structural row H. The definition of the carrier lists H only as the componentwise hsame data for the already displayed rows $S, D, R, B, Q, E, C, P, N$.

Use the NameCert obligations to read the certified transport surface. Since those obligations expose only the carrier coordinates and their recorded routes, a transport in H is fieldwise over the finite packet and cannot add a new analytic row.

Finally apply non-escape. The excluded objects are not coordinates of F and are not authenticated by the local naming row N. Thus classifier transport preserves exactly the displayed packet rows and nothing outside them. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier transport obligation`

Theorem 1119.12 (FiniteTailFilter cofinal-window directedness obligation)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

the displayed Cont *row* C *replays every accepted tail-window read, dyadic tolerance read, regular rational readback, tail-budget read, common-tail comparison read, and RealUp seal read through the same finite cofinal packet* $S, D, R, B, Q, E, H, P, N$.

Proof. Start with the continuation coordinate C. By the carrier definition, C records displayed routes for window readback, dyadic tolerance use, regular-rational readback, tail-budget consumption, common-tail comparison, and RealUp seal consumption.

Apply tail-window exactness. The budget, comparison, and seal-facing reads all consume the same S, D window and tolerance packet, so the continuation cannot branch into an unrecorded selector or tolerance ledger.

Use the Real seal handoff theorem to identify the downstream route. Every Real or Completion consumer reaches E through the finite packet $S, D, R, B, Q, H, C, P, N$, and the replay row C is the route data for that same packet. Hence all accepted reads are directed through the displayed cofinal window surface. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier cofinal window directedness obligation`

Theorem 1119.13 (FiniteTailFilter cofinal-route exhaustion)**.** *Every* $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing consumer of an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

factors through the cofinal route $S, D, R, B, Q, E, H, C, P, N$. *No such consumer can add private windows, detached dyadic ledgers, selected limits, quotient streams, host equality, countable-choice rows, ambient completeness, or standard bridge rows.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier cofinal route exhaustion`

Proof. First assemble the carrier route from the NameCert obligations and tail-window exactness, Definition 1119.2 and Definition 1119.4. These identify $S, D, R, B, Q, E, H, C, P, N$ as the accepted packet and make S, D the exact window and tolerance rows read by B, Q, E.

Next pass to the Real-facing surface. Apply the Real seal handoff and the RealWindowBudget consumer link, Definition 1119.5 and Definition 1119.7. They route RealWindowBudget, Real, and Completion-facing consumers through the same finite packet.

Use the Real-window selector bridge to rule out a selector-side replacement. Definition 1119.8 keeps selector stability tied to $S, D, R, B, Q, E, H, C, P, N$, not to a private window family or detached dyadic ledger.

Finally apply non-escape, Definition 1119.3. Since the carrier contains no selected-limit, quotient-stream, host-equality, countable-choice, ambient completeness, or standard bridge coordinate, the displayed cofinal route exhausts the admitted consumer surface. □

Theorem 1119.14 (FiniteTailFilter structural provenance obligation)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

the rows P *and* N *record exactly the sibling and kernel provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$. *They record no independent completion theorem, quotient carrier, selected limit, or bridge schema.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier structural provenance obligation`

Proof. Read P and N from the carrier. The package row P names the sibling and kernel sources of the displayed packet, while N is the local $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ row for the same surface.

Apply the NameCert obligations. They identify the provenance domain as the listed sibling chapters together with the kernel rows BHist, hsame, Cont, and $\mathsf{NameCert}$.

The carrier has no separate coordinate for an independent completion theorem, quotient carrier, selected limit, or bridge schema. Therefore the provenance and naming rows certify exactly the stated sources and no larger structure. □

Theorem 1119.15 (FiniteTailFilter terminal cofinal-consumer exclusion)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

every terminal $\mathsf{Real}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, *or completion-facing consumer of the accepted cofinal packet factors through the displayed finite route* $S, D, R, B, Q, E, H, C, P, N$.

Consequently such a consumer cannot recover quotient stream equality, selected limits, host real equality, ambient completeness, metric-completeness data, countable choice, a private selector, a separate dyadic ledger, or a standard bridge row from the cofinal packet.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier terminal cofinal consumer exclusion`

Proof. Start with tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. They identify the stream-window and dyadic-tolerance rows as the common cofinal packet used by the tail budget, common-tail comparison, Real seal, and completion-facing reads.

Next apply the RealWindowBudget selector bridge and the sibling tail handoff, Definition 1119.8 and Definition 1119.9. These force selector-stability and sibling reads to reuse the same $S, D, R, B, Q, E, H, C, P, N$ route rather than a detached window family, tolerance ledger, Real row, or seal row.

Use the cofinal-window directedness obligation to route every terminal read through the displayed continuation row C. The continuation can replay only the accepted tail-window, dyadic-tolerance, regular-rational, tail-budget, common-tail, and Real seal reads carried by the finite packet.

Finally combine non-escape with the structural provenance obligation, Definition 1119.3 and Definition 1119.14. The accepted carrier has no coordinate for quotient stream equality, selected limits, host real equality, ambient or metric completeness, countable choice, private selectors, detached ledgers, or standard bridge rows, and the provenance rows P, N certify no hidden completion source. Hence every terminal consumer remains confined to the displayed cofinal route. □

Theorem 1119.16 (FiniteTailFilter common-tail seal cofinality)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted FiniteTailFilter$^{\uparrow}$ *carrier. Any* RealWindowBudget$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, Real$^{\uparrow}$, *or completion-facing consumer that uses the accepted* FiniteTailFilter$^{\uparrow}$ *packet must identify its* StreamName$^{\uparrow}$ *window row,* DyadicRatCore$^{\uparrow}$ *tolerance row,* RegSeqRat$^{\uparrow}$ *readback row,* DyadicTailBound$^{\uparrow}$ *budget row,* CauchyTailComparison$^{\uparrow}$ *row, and* Real$^{\uparrow}$ *seal row with the displayed finite packet* $S, D, R, B, Q, E, H, C, P,$

No private selector, detached dyadic ledger, quotient stream, selected limit, countable-choice row, ambient completeness row, or standard bridge row is admitted by this common-tail seal.

Proof. First apply tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. These identify the window, tolerance, readback, budget, comparison, and Real seal rows as coordinates of the same accepted finite packet.

Next apply the RealWindowBudget consumer link and the Real-window selector bridge, Definition 1119.7 and Definition 1119.8. A Real-window consumer therefore reads its selector and budget data through $S, D, R, B, Q, E, H, C, P, N$, not through a private selector family or detached dyadic ledger.

Use the sibling tail handoff and cofinal-window directedness obligation, Definition 1119.9 and Definition 1119.12. The UniformCauchyCriterion and completion-facing reads must replay the same continuation route C over the displayed packet, so the common-tail comparison row cannot be replaced by a quotient stream, selected limit, countable-choice row, ambient completeness row, or standard bridge row.

Finally invoke terminal cofinal-consumer exclusion, Definition 1119.15. It confines every terminal Real-window, Real, UniformCauchyCriterion, or completion-facing consumer to the accepted cofinal packet, which proves that the finite packet $S, D, R, B, Q, E, H, C, P, N$ is the common route consumed by those chapters. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier common tail seal cofinality`

Theorem 1119.17 (FiniteTailFilter cofinal-window budget normal form)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted FiniteTailFilter$^{\uparrow}$ *carrier. Every accepted* RealWindowBudget$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, Real$^{\uparrow}$, *or completion-facing read of* F *has the same normal-form route* $S, D, R, B, Q, E, H, C, P, N$. *The rows* S, D *are the only window and dyadic tolerance rows,* B, Q *are the only tail-budget and common-tail comparison rows, and* E *is the only* Real$^{\uparrow}$ *seal row.*

No private selector, detached dyadic ledger, quotient stream, selected limit, host real equality, countable-choice row, ambient completeness row, metric-completeness row, independent completion row, or standard bridge row is part of this normal form.

Proof. First build the finite route from tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. These identify S, D, R, B, Q, E as one accepted finite packet rather than separate window, tolerance, readback, budget, comparison, and Real seal sources.

Next account for Real-window and selector-facing consumers. Apply Definition 1119.7 and Definition 1119.8. They force every RealWindowBudget$^{\uparrow}$ read to use the same route $S, D, R, B, Q, E, H, C, P, N$, with no private selector family or detached dyadic ledger.

Then route the uniform and completion-facing reads through the sibling and cofinal rows. Use Definition 1119.9 together with Definition 1119.12. The continuation row C replays only the displayed cofinal packet, so the UniformCauchyCriterion$^{\uparrow}$ and completion-facing consumers cannot replace B or Q by another tail-budget or common-tail comparison row.

Finally apply Definition 1119.16. It states that every RealWindowBudget, UniformCauchyCriterion, Real, or completion-facing consumer identifies its window, tolerance, readback, budget, comparison, and Real seal rows with the accepted packet $S, D, R, B, Q, E, H, C, P, N$.

The exclusion clause follows from terminal cofinal-consumer exclusion and structural provenance, Definition 1119.15 and Definition 1119.14. Since the accepted carrier has no coordinate for the listed external structures, the normal form is exactly the displayed route. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FinitetailfilterCofinalWindowBudgetNormalForm`

Theorem 1119.18 (FiniteTailFilter terminal classifier exactness)**.** *For every accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

the terminal classifier available to any $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer is exactly the componentwise classifier on the displayed finite route* $S, D, R, B, Q, E, H, C, P, N$*.*

Thus no terminal read may replace S, D *by a private window or tolerance ledger, replace* R, B, Q, E *by detached* $\mathsf{RegSeqRat}^{\uparrow}$*, tail-budget, common-tail, or* $\mathsf{Real}^{\uparrow}$ *rows, or introduce quotient streams, selected limits, host equality, countable choice, ambient completeness, or bridge schema data.*

Proof. Begin with the NameCert obligations, Definition 1119.2. They expose the accepted packet as the rows $S, D, R, B, Q, E, H, C, P, N$, with H carrying the componentwise hsame transports, C carrying the displayed continuation routes, and P, N carrying package provenance and local naming.

Next apply tail-window exactness, Definition 1119.4. The window and dyadic tolerance inputs of the tail-budget row, the common-tail comparison row, and the Real seal are exactly S and D, so the classifier cannot be evaluated through a private window family or a detached tolerance ledger.

For the Real and completion-facing side, use the Real seal handoff and the RealWindowBudget consumer link, Definition 1119.5 and Definition 1119.7. These route every Real-window, Real, and completion-facing read through the same finite packet before reaching the seal row E.

For the common-tail side, use common-tail seal cofinality, Definition 1119.16. It identifies the $\mathsf{RegSeqRat}^{\uparrow}$ readback, tail-budget, common-tail comparison, and $\mathsf{Real}^{\uparrow}$ seal rows with the displayed route R, B, Q, E rather than with detached sibling rows.

Finally apply the cofinal-window budget normal form, Definition 1119.17. That theorem states that every terminal RealWindowBudget, UniformCauchyCriterion, Real, or completion-facing read has the same normal-form route $S, D, R, B, Q, E, H, C, P, N$ and no coordinate for the excluded external structures. Therefore the only terminal classifier left available is the componentwise classifier on that displayed route. □

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier terminal classifier exactness`

Theorem 1119.19 (FiniteTailFilter terminal Lean target surface)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. The paper-level Lean target for this chapter is the exact terminal packet* F*: every* $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and completion-facing read must consume the displayed window row, dyadic tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, tail-budget row, common-tail comparison row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *routes, package provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ *row.*

No terminal Lean target may replace that packet by quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, metric-completeness data, independent completion data, or a standard bridge row.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterTerminalLeanTargetSurface`

Proof. Begin with Definition 1119.18. It identifies the terminal classifier as the componentwise classifier on $S, D, R, B, Q, E, H, C, P, N$ for every RealWindowBudget, UniformCauchyCriterion, Real, and completion-facing consumer.

Use the cofinal-window budget normal form, Definition 1119.17. It states that the same finite route is the normal form for the listed consumers and that the excluded external structures are not coordinates of the carrier.

Thus the Lean target surface has no additional input row to consume. It must take exactly the displayed packet and prove the terminal reads from that packet. □

Theorem 1119.20 (FiniteTailFilter formal-target consumer route)**.** *Every formal target consuming a* $\mathsf{FiniteTailFilter}^{\uparrow}$ *terminal read must read the same finite cofinal-tail route*

$$S, D, R, B, Q, E, H, C, P, N.$$

The route factors RealWindowBudget, UniformCauchyCriterion, Real, and completion-facing requests through BHist *carrier rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ *row before any terminal classifier is evaluated.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier formal target consumer route`

Proof. First apply Definition 1119.19. It fixes the formal target as the exact terminal packet rather than a family of host-side consumer assumptions.

Next use Definition 1119.16. It forces the window, tolerance, readback, budget, comparison, and Real seal rows for each terminal consumer to agree with the accepted packet.

Finally attach the consumer exclusion theorem, Definition 1119.15. Since every terminal consumer is confined to the accepted cofinal packet, the formal target route is precisely the finite BHist, hsame, Cont, provenance, and NameCert route stated above. □

Theorem 1119.21 (FiniteTailFilter Lean intake non-escape)**.** *The Lean intake for* $\mathsf{FiniteTailFilter}^{\uparrow}$ *is non-escaping: it may consume only the BHist packet* $S, D, R, B, Q, E, H, C, P, N$ *and the terminal classifier facts already displayed in this chapter. In particular, the intake cannot add a private RealWindowBudget selector, a separate UniformCauchyCriterion witness, a host Real equality, a selected completion point, a quotient stream, countable choice, ambient completeness, metric completeness, or bridge schema data.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterLeanIntakeNonescape`

Proof. Start from Definition 1119.20. The formal target reads the terminal consumers only through the displayed cofinal-tail route.

Use Definition 1119.18 to evaluate the classifier. The classifier is already exact on $S, D, R, B, Q, E, H, C, P, N$, so an additional selector, witness, equality, completion point, quotient stream, choice principle, completeness row, or bridge row would not be read by the accepted classifier.

The non-escape clause follows from Definition 1119.3 and Definition 1119.14. Both keep accepted reads inside the finite carrier, transport, route, provenance, and local naming surface. □

Theorem 1119.22 (FiniteTailFilter formal intake route exhaustion)**.** *Every formal intake read of an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

factors through the finite tail-window row S*, the* $\mathsf{RealWindowBudget}^{\uparrow}$ *row, the cofinal-route rows* D, R, B, Q, E*, the terminal classifier, and the Lean intake non-escape surface. No formal intake read may add a hidden tail oracle, quotient stream, selected host limit, host real equality, choice datum, ambient completeness row, metric-completeness row, independent completion datum, or bridge row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier formal intake route exhaustion`

Proof. First apply the formal-target consumer route, Definition 1119.20. It confines every formal target to the finite cofinal-tail route $S, D, R, B, Q, E, H, C, P, N$ before any terminal classifier is evaluated.

Use the cofinal-window budget normal form, Definition 1119.17. This factors RealWindowBudget, UniformCauchyCriterion, Real, and completion-facing reads through the displayed tail-window, tolerance, regular-readback, tail-budget, comparison, and Real-seal rows.

Next apply terminal classifier exactness, Definition 1119.18. The classifier available to every terminal consumer is exactly the componentwise classifier on the displayed finite route, so the formal intake has no second classifier outside the packet.

Finally invoke Lean intake non-escape, Definition 1119.21. Since the intake may consume only the BHist packet and the terminal classifier facts already displayed in the chapter, every formal intake read is exhausted by the finite route above and cannot add a hidden tail oracle, quotient stream, selected host limit, choice datum, completeness datum, or bridge row. □

Theorem 1119.23 (FiniteTailFilter formal target route closure)**.** *The formal target for the accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

is the closed L10 cofinal-tail surface formed by the existing DyadicRatCoreUp*,* StreamNameUp*,* RegSeqRatUp*,* RealUp*,* RealWindowBudgetUp*,* UniformCauchyCriterionUp*,* CauchyTailComparisonUp*, and* DyadicTailBoundUp *rows. It consumes only finite cofinal tail windows, classifier-stable tail cutoffs, non-escape ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterFormalTargetRouteClosure`

Proof. Start from Definition 1119.2 and Definition 1119.22. These identify the accepted packet and exhaust every formal intake route through $S, D, R, B, Q, E, H, C, P, N$.

Next apply Definition 1119.17 and Definition 1119.18. The window, dyadic tolerance, regular-readback, tail-budget, common-tail comparison, and Real-seal rows are therefore the only classifier-stable tail cutoffs visible to an L10 consumer.

Finally use Definition 1119.21. Since the intake has no hidden tail oracle, quotient stream, selected limit, host equality, choice datum, completeness row, or bridge row, the formal target is exactly the closed finite cofinal-tail surface above. □

Theorem 1119.24 (FiniteTailFilter bridge premise surface)**.** *Any bridge premise that reads the L10* FiniteTailFilter$^{\uparrow}$ *surface must first read the same finite packet* $F = (S, D, R, B, Q, E, H, C, P, N)$*. The premise may expose* DyadicRatCoreUp*,* StreamNameUp*,* RegSeqRatUp*,* RealUp*,* RealWindowBudgetUp*,* UniformCauchyCriterionUp*,* CauchyTailComparisonUp*, and* DyadicTailBoundUp *only as finite tail-window, dyadic tolerance, regular-readback, Real-seal, budget, criterion, comparison, and tail-bound rows inside the packet; it may not replace them by a standard bridge row, ambient completeness, quotient stream equality, selected host limits, or host Real equality.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterBridgePremiseSurface`

Proof. Use Definition 1119.23 to put every bridge-facing premise on the closed cofinal-tail surface.

Then apply Definition 1119.16. The Real seal, common-tail comparison, dyadic tail bound, and window-budget data are consumed together, so a bridge premise cannot separate one row into an ambient completion witness.

The excluded replacements are ruled out by Definition 1119.15 and Definition 1119.21. Thus a later bridge proof has a premise surface, but that surface is still the finite packet rather than a host-side bridge assumption. □

Theorem 1119.25 (FiniteTailFilter Lean-ready obligation surface)**.** *The Lean-ready obligation for* FiniteTailFilter$^{\uparrow}$ *is to prove that the formal target route closes over the displayed finite cofinal-tail packet and no other data. Concretely, every* RealWindowBudgetUp*,* UniformCauchyCriterionUp*,* CauchyTailComparisonUp*,* DyadicTailBoundUp*,* RegSeqRatUp*,* StreamNameUp*,* DyadicRatCoreUp*, and* RealUp *read must be reconstructed from* $S, D, R, B, Q, E, H, C, P, N$*, the terminal classifier exactness row, and the non-escape ledger already certified in this chapter.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier lean ready obligation surface`

Proof. First collect the carrier and naming data from Definition 1119.1 and Definition 1119.2. They provide the rows that a Lean theorem can bind without adding a new carrier.

Next combine cofinal-route exhaustion with terminal classifier exactness. Theorems 1119.22 and 1119.18 leave the target with only finite cofinal windows and classifier-stable tail cutoffs.

It remains to record the non-escape side. Theorems 1119.21 and 1119.23 show that the future Lean obligation is a closure theorem over this packet, not a request for quotient streams, selected limits, ambient completeness, or a bridge row. □

Theorem 1119.26 (FiniteTailFilter Real-completion lattice link)**.** *An accepted finite tail filter consumed by the current Real-completion route determines the same cofinal-tail lattice row for* StreamName$^{\uparrow}$ *finite windows,* DyadicRatCore$^{\uparrow}$ *tolerance windows,* RegSeqRat$^{\uparrow}$ *regularity tails, and* Real$^{\uparrow}$ *seal reads. The lattice row is the displayed finite packet*

$$F = (S, D, R, B, Q, E, H, C, P, N),$$

with StreamName, DyadicRatCore, RegSeqRat, and RealUp appearing only as the four sibling faces referenced by chapter 126, chapter 291, chapter 301, and chapter 38.

The route exports only finite cofinal-tail windows through those existing NameCert routes. It does not export a selected limit, host real equality, quotient stream equality, or Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier real completion lattice link`

Proof. Start with tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. They identify S, D, R, B, Q, E as one finite cofinal-tail packet consumed by the window, tolerance, regular-readback, budget, comparison, and Real-seal rows.

Next apply the shared-window meet, Definition 1119.10. The meet forces all accepted consumers to use the same cofinal window and dyadic tolerance packet rather than separate StreamName, DyadicRatCore, RegSeqRat, or RealUp witnesses.

Finally apply the Lean-ready obligation surface, Definition 1119.25. It records that every Real-completion route read of the four sibling chapters must be reconstructed from $S, D, R, B, Q, E, H, C, P, N$, the terminal classifier exactness row, and the non-escape ledger, so the cofinal-tail lattice row is shared across all four objects. □

Theorem 1119.27 (FiniteTailFilter refinement seal witness-ledger pullback)**.** *Let an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

expose the cofinal-tail row consumed by compatible accepted $\mathsf{CauchyModulusRefinement}^{\uparrow}$, $\mathsf{CauchyWitnessLedger}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *packets. Then the three consumers determine one shared finite route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Each consumer reads the same* $S, D, R, B, Q, E, H, C, P, N$ *rows as its StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, Real seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *coordinate.*

No quotient stream, selected limit, host real equality, choice row, or ambient completeness coordinate is available along the pullback.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier refinement seal witness ledger pullback`

Proof. First combine the local Real-completion lattice link, Definition 1119.26, with the shared-window meet, Definition 1119.10. These two rows put the four sibling faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same finite packet F before any downstream consumer reads the route.

For the refinement consumer, use the CauchyModulusRefinement seal and selector rows: Definition 885.7, Definition 885.8, and Definition 885.148. For the witness-ledger consumer, use Definition 902.64, Definition 902.67, and Definition 902.68.

Together these rows say that refinement and ledger reads enter through the displayed finite route rather than through a detached selector or completion witness.

Finally apply the CauchyLimitSeal RegSeqRat-window boundary, Definition 386.11. It keeps the limit-seal consumer inside the same RegSeqRat and StreamName window route already fixed by F. Since all three consumers factor through the same finite packet and each cited non-escape row excludes quotient streams, selected limits, host equality, choice data, and ambient completeness, the displayed $S, D, R, B, Q, E, H, C, P, N$ rows form the shared pullback route. □

Theorem 1119.28 (FiniteTailFilter exact-boundary terminal normal form)**.** *Let an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

export its cofinal-tail route to an accepted RealCompletionExactBoundaryUp *packet whose exact-boundary consumer reads that route. Then the composite terminal read is absorbed by the four-face terminal normal form*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The composite exports no quotient stream equality, selected limit, host real equality, countable-choice row, ambient-completeness row, metric-completeness row, bridge row, or hidden tail selector.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterExactBoundaryTerminalNormalForm`

Proof. First apply the Real-completion lattice link, Definition 1119.26. It places the StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity face, and terminal Real seal face on the same finite packet F before the exact-boundary consumer reads them.

Next use the refinement seal witness-ledger pullback, Definition 1119.27. The pullback says that downstream refinement, witness-ledger, and seal consumers still read the displayed $S, D, R, B, Q, E, H, C, P, N$ rows and do not acquire a detached selector, quotient equality, selected limit, host equality, choice row, or ambient completeness coordinate.

Finally invoke Definition 1120.23. Once the accepted RealCompletionExactBoundary packet consumes the finite route, its terminal read is absorbed by the same four exact-boundary faces and structural rows. Combining the two finite-tail-filter rows with that absorption theorem leaves no carrier slot for metric completeness, a bridge row, or a hidden tail selector, so the composite terminal read is exactly the displayed four-face normal form. □

Theorem 1119.29 (FiniteTailFilter exact-boundary shared consumer determinacy)**.** *Let an accepted* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

export its cofinal-tail route to two accepted $\mathsf{RealCompletionExactBoundaryUp}$ *packets. Suppose their exact-boundary consumers read the same displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows through* F*. Then both consumers determine the same four-face terminal normal form through* S, D, R, E, H, C, P, N*:*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

No quotient stream equality, selected limit, host real equality, countable-choice row, ambient-completeness row, metric-completeness row, bridge row, or hidden tail selector can distinguish the two terminal consumers.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier exact boundary shared consumer determinacy`

Proof. Apply the Real-completion lattice link, Definition 1119.26. It places the four L10 faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same accepted finite packet F before either exact-boundary consumer reads the route.

Next use the refinement seal witness-ledger pullback, Definition 1119.27. The pullback keeps refinement, witness-ledger, and seal reads on the displayed $S, D, R, B, Q, E, H, C, P, N$ route, so neither consumer can introduce a detached selector, quotient stream, selected limit, host equality, choice row, or ambient completeness coordinate.

Now invoke the exact-boundary terminal normal form, Definition 1119.28, for each consumer. Since both consumers read the same StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows, both terminal reads are absorbed into the same four-face normal form through the displayed finite route.

Finally use downstream exact-boundary determinacy, Definition 1120.10 and Definition 1120.18, only to identify what the exact-boundary consumers can observe after absorption. Those determinacy rows leave no additional carrier coordinate for metric completeness, a bridge row, or a

hidden tail selector, so the two accepted consumers cannot branch into distinguishable terminal outcomes. □

Theorem 1119.30 (FiniteTailFilter public exit package)**.** *The public L10 exit package for* $\mathsf{FiniteTailFilter}^{\uparrow}$ *is exactly the finite route*

$$S, D, R, B, Q, E, H, C, P, N$$

consuming the sibling faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, *and* $\mathsf{DyadicTailBound}^{\uparrow}$. *The exit package has no quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness, metric-completeness datum, or bridge row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterUp StdBridge`

Proof. First combine the carrier and NameCert obligations, Definition 1119.1 and Definition 1119.2. They expose the route $S, D, R, B, Q, E, H, C, P, N$ as the finite cofinal-tail packet available to public consumers.

Next use tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. These identify the StreamName, DyadicRatCore, RegSeqRat, RealWindowBudget, UniformCauchyCriterion, CauchyTailComparison, DyadicTailBound, and RealUp faces as reads of the same finite tail-window surface.

Apply terminal classifier exactness and the Lean-ready obligation surface, Definition 1119.18 and Definition 1119.25. The classifier and obligation rows leave no public coordinate for quotient stream equality, selected limits, host real equality, choice, ambient completeness, metric completeness, or bridge data.

Finally invoke Definition 1119.26. The four Real-completion faces already share one cofinal-tail lattice row, so the public L10 exit package is precisely the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteTailFilter}^{\uparrow}$ packet is generated from a finite StreamName window ledger, DyadicRatCore tolerance ledger, RegSeqRat readback, DyadicTailBound tail-budget row, CauchyTailComparison common-tail row, RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1119.1, Definition 1119.2, Definition 1119.3, Definition 1119.4, Definition 1119.5, Definition 1119.9, Definition 1119.10, Definition 1119.11, Definition 1119.12, Definition 1119.14, Definition 1119.15, and Definition 1119.16 bind the finite cofinal-tail filter packet, its classifier transport, its continuation replay, and its provenance/name certification surface for Real and Completion seal reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteTailFilter}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier\_real\_completion\_lattice\_link`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, ambient completeness, metric-completeness data, countable choice, or standard bridge rows.

Upgrade path: No further upgrade remains on the formal axis; the chapter is bridgeChecked for the finite cofinal-tail filter surface.

Theorem 1119.31 (FiniteTailFilter classifier alignment boundary)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. Any classifier-aligned receiving gate, whether presented as a local classifier read, a fiberwise certificate, a pullback surface, or a reindexing surface, is admitted only after it re-reads the displayed rows of* F.

The transported rows are exactly

$$S, D, R, B, Q, E, C, P, N,$$

with H serving as the componentwise hsame *transport witness for those rows. The classifier then evaluates on the displayed route*

$$S, D, R, B, Q, E, H, C, P, N.$$

The rows forgotten at this boundary are precisely all non-carrier rows: private window selectors, detached dyadic ledgers, detached $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, detached tail-budget rows, detached common-tail rows, detached* $\mathsf{Real}^{\uparrow}$ *seals, quotient stream equalities, selected limits, host real equalities, countable-choice rows, ambient or metric completeness data, independent completion rows, and standard bridge rows.*

Consequently a copied source excerpt is never sufficient evidence for classifier alignment. The receiving gate must project the rows again from the accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet and may classify only the componentwise route displayed by F.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterClassifierAlignmentBoundary`

Proof. Start with the carrier definition and the NameCert obligations, Definition 1119.1 and Definition 1119.2. They identify the accepted packet as the ten displayed rows $S, D, R, B, Q, E, H, C, P, N$. The analytic coordinates are S, D, R, B, Q, E, and the structural coordinates H, C, P, N transport, replay, source, and name that same packet.

Apply the carrier transport obligation, Definition 1119.11. It states that the componentwise hsame row H transports exactly $S, D, R, B, Q, E, C, P, N$. Thus H is not a new analytic coordinate and not a hidden classifier input; it is the transport witness for the already displayed packet rows.

Use tail-window exactness and the Real seal handoff, Definition 1119.4 and Definition 1119.5. These force the window row S, dyadic ledger D, regular readback R, tail budget B, common-tail row Q, and seal row E to remain one route. A pullback or reindexing read therefore cannot replace one of these rows by a private selector, separate ledger, detached readback, detached budget, detached comparison row, or detached seal.

Now apply terminal classifier exactness, Definition 1119.18. The terminal classifier available to $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion-facing consumers is exactly the componentwise classifier on $S, D, R, B, Q, E, H, C, P, N$. Hence a classifier-aligned receiving gate has no second classifier surface outside the displayed packet.

Finally use non-escape and structural provenance, Definition 1119.3 and Definition 1119.14. The carrier and its provenance rows contain no quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness row, metric-completeness row, independent completion datum, or standard bridge schema. These are precisely the rows forgotten at the alignment boundary. Therefore the receiving gate must re-read the displayed rows from F and may not classify from a copied excerpt or any off-packet coordinate. □

Theorem 1119.32 (FiniteTailFilter forgetful projection boundary)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. Suppose a consumer displays this packet with explicit sibling-face names for the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicTailBound}^{\uparrow}$ *budget row, the* $\mathsf{CauchyTailComparison}^{\uparrow}$ *common-tail row, the*

$\mathsf{Real}^{\uparrow}$ *seal row, and any terminal* $\mathsf{RealWindowBudget}^{\uparrow}$ *or* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *read that consumes those rows. Then the BEDC-native forgetful projection of that richer display is exactly the target packet*

$$S, D, R, B, Q, E, H, C, P, N.$$

After the projection, the target carrier is F, the terminal classifier is the componentwise classifier on these ten displayed rows, the ledger surface is carried by H, C, P, and the local naming coordinate is N. No projected read may drop one of $S, D, R, B, Q, E, H, C, P, N$, and no sibling-face name survives as an independent coordinate outside the target packet. In particular, the projection exports no private selector, detached dyadic ledger, quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness row, metric-completeness row, independent completion row, or standard bridge row.

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterForgetfulProjectionBoundary`

Proof. Start with the carrier and NameCert obligations, Definition 1119.1 and Definition 1119.2. They present the accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ packet as precisely the ten rows $S, D, R, B, Q, E, H, C, P, N$. The named sibling faces are not additional coordinates of the carrier: the window face is S, the dyadic face is D, the regular-rational readback is R, the tail-budget face is B, the common-tail face is Q, and the Real seal face is E.

Next apply tail-window exactness and cofinal-window directedness, Definition 1119.4 and Definition 1119.12. These show that all accepted budget, comparison, seal, and continuation reads replay the same finite cofinal packet. Thus forgetting the external face names cannot change the rows being read; it only removes the labels by which sibling chapters address those rows.

The terminal classifier is fixed by Definition 1119.18. Every terminal Real-window, UniformCauchyCriterion, Real, or completion-facing consumer classifies the packet componentwise on $S, D, R, B, Q, E, H, C, P, N$. Therefore a projection that dropped one of these coordinates would no longer preserve the classifier surface available to the terminal consumer.

The ledger and naming coordinates are fixed by Definition 1119.11, Definition 1119.14, and Definition 1119.30. The transport row H, continuation row C, package row P, and local NameCert row N are the only ledger and naming data certified for the public exit package. Hence the forgetful projection retains them as target coordinates rather than replacing them by a sibling provenance label.

Finally apply non-escape and terminal cofinal-consumer exclusion, Definition 1119.3 and Definition 1119.15. The accepted carrier has no coordinate for private selectors, detached dyadic ledgers, quotient stream equality, selected limits, host real equality, choice, ambient or metric completeness, independent completion data, or standard bridge rows. Any richer display retaining one of those objects after forgetting sibling-face names would therefore retain a coordinate that was never present in F. The only possible BEDC-native projection is the displayed target packet $S, D, R, B, Q, E, H, C, P, N$. □

Theorem 1119.33 (FiniteTailFilter substrate bisimulation boundary)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. Any receiving gate that presents F as a* $\mathsf{Rule110}$*, cellular, binary,* $\mathsf{FKernel}$*, or* BHist *substrate read is admitted only as a relation certificate over the displayed rows of F, not as definitional equality with a host substrate.*

The source of the certificate is the accepted carrier F. The target is the consumer-visible route

$$S, D, R, B, Q, E, H, C, P, N.$$

The direction is source-to-target: the gate first projects the intermediate route rows

$$S, D, R, B, Q,$$

then reaches the seal row E, and keeps the noneliminable structural boundary rows

$$H, C, P, N.$$

On the analytic rows S, D, R, B, Q, E the certificate is a row embedding into the target route. With the componentwise transport H, displayed replay C, provenance P, and local naming row N retained, the certificate is a substrate mirror and hence a bisimulation boundary for the displayed finite packet.

No receiving gate may replace this relation certificate by a copied source excerpt, a host coordinate, an external source coordinate, a private window selector, a detached dyadic ledger, a detached regular readback, a detached tail-budget row, a detached common-tail row, a detached $\mathsf{Real}^{\uparrow}$ *seal, quotient stream equality, selected limits, host real equality, countable-choice data, ambient completeness, metric-completeness data, independent completion data, or a standard bridge row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier substrate bisimulation boundary`

Proof. Start with the carrier definition and the NameCert obligations, Definition 1119.1 and Definition 1119.2. They identify the accepted packet as exactly the ten displayed rows $S, D, R, B, Q, E, H, C, P, N$. Thus a receiving gate has a local source only when it binds this carrier, and it has no local source coordinate outside the displayed packet.

Apply tail-window exactness, Definition 1119.4. The rows S and D are the only window and dyadic tolerance rows, and the rows B, Q, E read that same window and tolerance packet. Therefore the intermediate analytic route is the listed route S, D, R, B, Q followed by the seal row E; it cannot be replaced by a private selector, separate tolerance ledger, detached readback, detached budget, detached comparison row, or detached seal.

Use the Real seal handoff and the cofinal-route exhaustion, Definition 1119.5 and Definition 1119.13. These route every Real, Real-window, and completion-facing read through the same finite cofinal packet $S, D, R, B, Q, E, H, C, P, N$. Hence a substrate presentation of the packet can only point from the accepted carrier to this consumer-visible route.

Now apply the carrier transport, continuation, and provenance obligations, Definition 1119.11, Definition 1119.12, and Definition 1119.14. The row H transports the displayed rows componentwise, the row C replays the displayed read route, and the rows P, N record provenance and naming for that same packet. These structural rows are not eliminable when the gate claims a substrate relation, because they are exactly what records source, target, and direction.

Finally invoke terminal classifier exactness and classifier alignment, Definition 1119.18 and Definition 1119.31. The terminal classifier evaluates only on the componentwise route $S, D, R, B, Q, E, H, C, P, N$, and a copied source excerpt is not sufficient evidence for alignment. Together with non-escape, Definition 1119.3, this rules out every unlisted host or external coordinate. The only admissible substrate claim is therefore the stated source-to-target row embedding with H, C, P, N retained as the substrate mirror, which is precisely the bisimulation boundary for the displayed finite packet. □

Theorem 1119.34 (FiniteTailFilter left-right mirror boundary)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. Any left-oriented or right-oriented mirror read of the finite tail-filter packet has the same BEDC-native boundary: its displayed history row is* S*, its observation and tolerance rows are* D, R, B, Q*, its result row is the* $\mathsf{Real}^{\uparrow}$ *seal* E*, its transport and termination rows are* H, C*, and its certificate rows are* P, N*.*

Consequently the right-oriented counterpart is admitted only by re-reading the same packet $S, D, R, B, Q, E, H, C, P, N$ *with the endpoint words exchanged. It is not a second carrier, not a second theorem-label surface, and not a new dual row. In particular, a right-oriented read cannot introduce a private window selector, detached dyadic ledger, detached regular readback, detached tail-budget row, detached common-tail row, detached* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness row, metric-completeness row, independent completion row, or standard bridge row.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier left right mirror boundary`

Proof. Start from the carrier and NameCert obligations, Definition 1119.1 and Definition 1119.2. They expose the accepted packet as exactly the ten rows $S, D, R, B, Q, E, H, C, P, N$. Thus any orientation of a mirror read has only these rows available: S supplies the finite window history, D, R, B, Q supply the dyadic, regular-readback, tail-budget, and common-tail observation surface, E supplies the seal-facing result, and H, C, P, N supply transport, replay, provenance, and local naming.

Apply tail-window exactness and cofinal-window directedness, Definition 1119.4 and Definition 1119.12. The rows B, Q, E all consume the same displayed S, D window and tolerance packet, and the continuation row C replays every accepted read through that packet. Exchanging left and right endpoint words therefore cannot create a new selector, tolerance ledger, readback row, budget row, comparison row, or seal row; it can only re-read the same directed finite route.

Use the carrier transport obligation and structural provenance obligation, Definition 1119.11 and Definition 1119.14. The transport row H is componentwise over $S, D, R, B, Q, E, C, P, N$, while P and N certify the same sibling and kernel provenance. Hence the rows that terminate and certify a left-oriented read are already the rows that terminate and certify the right-oriented read.

Now apply the substrate bisimulation boundary, Definition 1119.33. It identifies the only admissible substrate mirror as the source-to-target row embedding over $S, D, R, B, Q, E, H, C, P, N$, with the structural rows retained. Since that boundary is already the whole displayed packet, the mirror counterpart has no independent right-side carrier or extra theorem-label surface to occupy.

Finally use classifier alignment, terminal classifier exactness, and non-escape, Definition 1119.31, Definition 1119.18, and Definition 1119.3. The receiving classifier must re-read the rows from F, evaluates only on the componentwise route $S, D, R, B, Q, E, H, C, P, N$, and has no coordinate for any excluded external structure. Therefore the left-right mirror closes only as the diagonal finite-tail packet boundary stated above. □

Theorem 1119.35 (FiniteTailFilter domain-codomain mirror boundary)**.** *Let*

$$F = (S, D, R, B, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ *carrier. Any domain-oriented read of* F *and any codomain-oriented mirror read of* F *have the same BEDC-native boundary exactly when both reads project to the displayed packet rows*

$$S, D, R, B, Q, E, H, C, P, N.$$

On this boundary, the domain history is the finite window ledger S*, the domain observation and tolerance surface is* D, R, B, Q*, the codomain result is the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the termination and replay data are* H, C*, and the certificate data are* P, N*.*

Consequently the codomain side is not a second carrier, not a second theorem-label surface, and not a hidden dual packet. It is admitted only by re-reading the same history, observation, result, termination, and certificate rows already displayed by F. If a receiving gate asks for a codomain row outside this list, the request is recorded as a boundary failure: no private selector, detached dyadic ledger, detached $\mathsf{RegSeqRat}^{\uparrow}$ *readback, detached tail-budget row, detached common-tail row, detached* $\mathsf{Real}^{\uparrow}$ *seal, quotient stream equality, selected limit, host real equality, countable-choice row, ambient completeness row, metric-completeness row, independent completion row, or standard bridge row is available from the accepted packet.*

Lean4 ✓ `BEDC.Derived.FiniteTailFilterUp.FiniteTailFilterCarrier domain codomain mirror boundary`

Proof. Start from the carrier definition and the NameCert obligations, Definition 1119.1 and Definition 1119.2. They expose the accepted $\mathsf{FiniteTailFilter}^{\uparrow}$ packet as exactly the rows $S, D, R, B, Q, E, H, C, P, N$. Hence a domain-oriented read has no source coordinate except the displayed finite window, dyadic, regular-readback, tail-budget, common-tail, seal, transport, replay, provenance, and naming rows.

Apply tail-window exactness, Definition 1119.4. The rows B, Q, E consume the same S, D window and tolerance packet, so a codomain-oriented read cannot obtain its result row by replacing the domain window, splitting the tail budget from the common-tail comparison, or substituting a detached tolerance ledger. The codomain result is the displayed seal row E reached through S, D, R, B, Q.

Use the Real seal handoff and cofinal-route exhaustion, Definition 1119.5 and Definition 1119.13. They route every Real, Real-window, and completion-facing read through the same finite packet before the seal is consumed. Thus exchanging the words domain and codomain changes only the orientation of the read; it does not create an additional row on either side.

The termination and certificate rows are fixed by the carrier transport, cofinal-window directedness, and structural provenance obligations, Definition 1119.11, Definition 1119.12, and Definition 1119.14. The row H transports the displayed rows componentwise, C replays the accepted route, and P, N certify provenance and local naming for the same packet. These rows are therefore shared by the domain and codomain presentations rather than duplicated on a codomain side.

Finally apply classifier alignment, substrate bisimulation, and the left-right mirror boundary, Definition 1119.31, Definition 1119.33, and Definition 1119.34. The classifier must re-read the rows from F, the only admissible substrate certificate is the row embedding over $S, D, R, B, Q, E, H, C, P, N$, and the mirror read closes only as the diagonal finite-tail packet boundary. Together with non-escape, Definition 1119.3, this excludes every off-packet codomain coordinate listed above. Hence the codomain mirror is exactly the same displayed packet, and any stronger codomain request is stopped at the stated boundary. □

Chapter 1120

A Concrete Naming Certificate for RealCompletionExactBoundaryUp

The RealCompletionExactBoundaryUp packet is the finite boundary surface for the Real completion route of chapter 9238. It sits after the seal route of chapter 386, the completion-seal classifier of chapter 900, the witness ledger of chapter 902, the synchronizer of chapter 886, the finite stream interface of chapter 126, the regular-rational readback surface of chapter 301, the dyadic endpoint ledger of chapter 291, and the terminal real-name seal of chapter 38. Its role is to bind those existing rows into one exact finite NameCert packet, not to add quotient streams, selected host limits, countable choice, host real equality, or ambient completeness.

Falsifiable prediction: If a Real completion boundary read bypasses either the displayed seal/classifier rows or the displayed witness/budget rows, the packet is refuted.

Independence witness: cauchylimitseal, realcompletionsealclassifier, cauchywitnessledger, cauchysealbudgetsynchronizer, real: no listed sibling contains the exact joint seal, classifier, witnessledger, synchronizer, StreamName, RegSeqRat, DyadicRatCore, and RealUp packet surface.

Definition 1120.1 (Real completion exact-boundary carrier)**.** *A* RealCompletionExactBoundaryUp *carrier is a finite* BHist *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *seal row consumed by the completion route;*
2. *K is the* RealCompletionSealClassifierUp *classifier row for the same displayed seal surface;*
3. *J is the* $\mathsf{CauchyWitnessLedger}^{\uparrow}$ *row naming the finite witness evidence for the route;*
4. *S is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row synchronizing the seal budget with the witness ledger;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by the same route;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to W;*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance and endpoint row used by the readback;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, K, J, S, W, R, D have been displayed;*

9. *H records componentwise* hsame *transports for all displayed rows;*
10. *C records the displayed* Cont *replay from seal and witness admission to the terminal Real seal;*
11. *P is the* Pkg *provenance over the sibling certificates,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ *row.*

The classifier compares carriers only through these rows. It has no coordinate for quotient stream equality, host real equality, a hidden selector, a second seal route, a detached witness ledger, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTasteGate single carrier alignment`

Theorem 1120.2 (Real completion exact-boundary NameCert obligations)**.** *For every accepted* RealCompletionExactBoundaryUp *carrier*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ *surface consists exactly of carrier exposure for B, completion-seal exposure through L, classifier exposure through K, witness-ledger exposure through J, synchronized budget exposure through S, finite-window exposure through W, regular-rational readback through R, dyadic tolerance readback through D, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryNameCertObligations`

Proof. Project the carrier of Definition 1120.1. Its displayed packet lists the seal row, classifier row, witness ledger, budget synchronizer, finite window, regular-rational readback, dyadic ledger, terminal Real seal, and structural rows.

Read the sibling dependencies in order. The completion seal L and classifier K fix the seal-facing side, while the witness ledger J and synchronizer S fix the evidence and budget side. The rows W, R, D, E are then the StreamName, RegSeqRat, DyadicRatCore, and RealUp projections of that same route.

The structural rows H, C, P, N only transport, replay, record provenance for, and name the displayed packet. Since no other coordinate is present in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1120.3 (Real completion exact-boundary non-escape)**.** *Every accepted* RealCompletionExactBoundaryUp *consumer factors through the finite route*

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

The consumer may inspect the completion seal, seal classifier, witness ledger, synchronized budget, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise transports, continuations, provenance, and local NameCert row. It receives no quotient stream equality, host real equality, hidden selector, second seal route, detached witness ledger, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryNonescape`

Proof. Start from the accepted packet $B = (L, K, J, S, W, R, D, E, H, C, P, N)$. The seal/classifier side and the witness/budget side are both displayed before the StreamName, RegSeqRat, DyadicRatCore, and RealUp rows can be consumed as boundary projections.

Apply Definition 1120.2. The local certificate exports exactly the rows $L, K, J, S, W, R, D, E, H, C, P, N$ and treats H, C, P, N as structural support for that same finite packet.

Finally inspect the excluded objects. Quotient stream equality, host real equality, hidden selectors, second seal routes, detached witness ledgers, and ambient completeness are absent from the carrier and classifier. A consumer that reads any of them has left the accepted exact-boundary packet. □

Theorem 1120.4 (Real completion StreamName–RegSeq route)**.** *Every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

routes the completion read through the finite chain W, R, D, E, H, C, P, N after the seal, classifier, witness, and synchronizer rows L, K, J, S have been displayed. In that route, W supplies the StreamName$^{\uparrow}$ *finite-window row, R supplies the* RegSeqRat$^{\uparrow}$ *readback row for the same window, D supplies the* DyadicRatCore$^{\uparrow}$ *tolerance and endpoint row consumed by that readback, and E supplies the terminal* Real$^{\uparrow}$ *seal reached by the displayed completion packet.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryStreamNameRegSeq route`

Proof. Project the carrier of Definition 1120.1. The coordinates W, R, D, E occur as named rows of the same accepted packet as L, K, J, S, H, C, P, N.

Apply Definition 1120.2. The local NameCert obligations expose the StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, and RealUp seal as part of one certificate surface, with H, C, P, N transporting, replaying, packaging, and naming that surface.

Use Definition 1120.3. Any completion consumer that attempts to read W, R, D, or E outside the displayed route would require an absent coordinate, so the completion read is forced through the finite packet W, R, D, E, H, C, P, N under the already displayed rows L, K, J, S. □

Theorem 1120.5 (Real completion terminal seal non-escape)**.** *No terminal* Real$^{\uparrow}$ *seal read accepted by a* RealCompletionExactBoundaryUp *packet can bypass the displayed route*

$$L, K, J, S, W, R, D, E, H, C, P, N$$

by using quotient stream equality, host real equality, a hidden selector, a second seal route, a detached witness ledger, or ambient completeness. The only accepted terminal seal read is the row E reached through the carrier, NameCert obligations, and non-escape surface of the same finite packet.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryRealSealNonescape`

Proof. Begin with the carrier. The row E is displayed as the terminal Real$^{\uparrow}$ seal only after the seal row L, classifier row K, witness ledger J, synchronizer row S, StreamName row W, RegSeqRat row R, and DyadicRatCore row D are present.

Apply Definition 1120.2. The certificate exports no equality quotient on streams, no host real equality coordinate, no selector coordinate, no alternate seal route, no detached ledger, and no ambient completeness witness.

Finally apply Definition 1120.3. Since every accepted consumer factors through $L, K, J, S, W, R, D, E, H, C, P, N$ any terminal seal read that uses one of the excluded routes is outside the packet rather than an accepted Real seal read. □

Theorem 1120.6 (Real completion exact-boundary four-face exactness)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the accepted completion boundary exposes exactly the four Phase Real and Completion faces $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, *and* $\mathsf{Real}^{\uparrow}(E)$ *after the seal, classifier, witness-ledger, and budget rows* L, K, J, S. *Every completion-facing consumer and every terminal Real-seal consumer factors through the structural rows* H, C, P, N *of the same packet. No quotient stream row, host real equality row, selected limit row, countable-choice row, ambient-completeness row, metric-completeness row, or bridge row is exported by the boundary.*

Proof. Project the carrier of Definition 1120.1. Its displayed rows place L, K, J, S first, followed by the four boundary faces W, R, D, E and the structural rows H, C, P, N.

Apply Definition 1120.2. The local certificate identifies the complete row inventory of the accepted packet, so the only Phase Real and Completion face rows available after L, K, J, S are W, R, D, and E.

Apply Definition 1120.4. The ordered route reads W as the $\mathsf{StreamName}^{\uparrow}$ finite-window face, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback face, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal face.

Finally use Definition 1120.5 together with Definition 1120.3. The first theorem blocks terminal Real-seal escape, and the second blocks completion-boundary escape, so any consumer of the accepted boundary must pass through H, C, P, N and receives none of the excluded quotient, host-equality, selected-limit, choice, completeness, or bridge coordinates. □

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryFourFaceExactness`

Theorem 1120.7 (Real completion exact-boundary finite-tail-filter handoff)**.** *Every accepted* RealCompletionExactB *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

whose completion boundary is read through a finite cofinal-tail filter must consume the displayed $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet of chapter 1119 before any completion-facing or terminal Real-seal consumer reads the boundary. Equivalently, the four faces* $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, *and* $\mathsf{Real}^{\uparrow}(E)$ *factor through that finite-tail-filter packet, with no quotient stream row, host real equality row, selected limit row, countable-choice row, ambient-completeness row, metric-completeness row, or bridge row inserted between the cofinal-tail handoff and the accepted boundary.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryFiniteTailFilterHandoff`

Proof. Start with Definition 1120.6. It identifies the only completion and Real faces exported by the accepted boundary as $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$, all after the rows L, K, J, S and before the structural rows H, C, P, N are consumed.

Apply Definition 1120.4. The displayed route orders the same four faces as a StreamName window, regular-rational readback, dyadic tolerance ledger, and terminal Real seal, so a completion-boundary read through a finite cofinal-tail filter has no separate route on which to read those faces.

Now use Definition 1119.9. The finite-tail-filter packet is exactly the sibling handoff surface for StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the Real seal. Hence the four displayed faces of the exact boundary must pass through that packet whenever the boundary is presented through a finite cofinal-tail filter.

Finally apply Definition 1119.5 and Definition 1120.5. The first theorem carries the same finite cofinal-tail route to Real and Completion consumers, and the second prevents the terminal $\mathsf{Real}^{\uparrow}(E)$ seal from bypassing the exact-boundary route. The excluded quotient, host-equality, selected-limit, choice, completeness, and bridge coordinates therefore cannot be inserted between the finite-tail-filter handoff and the accepted boundary read. □

Theorem 1120.8 (Real completion exact-boundary filter-handoff sibling link)**.** *The finite-tail-filter handoff of Definition 1120.7 is a sibling dependency on chapter 1119. In every accepted boundary read through a finite cofinal-tail filter, the four faces W, R, D, E consume the sibling* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet before the terminal* $\mathsf{Real}^{\uparrow}(E)$ *seal is read back.*

Proof. The cited handoff theorem already identifies $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$ as the exact boundary faces read through the finite cofinal-tail filter.

Use the sibling chapter named in chapter 1119. Its packet supplies the finite cofinal-tail surface consumed by those four faces, and Definition 1120.5 blocks terminal Real-seal readback outside that displayed route. □

Theorem 1120.9 (Real completion terminal-route pullback)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}(E)$ *read is the pullback of the same displayed route*

$$\mathsf{StreamName}^{\uparrow}(W), \quad \mathsf{RegSeqRat}^{\uparrow}(R), \quad \mathsf{DyadicRatCore}^{\uparrow}(D)$$

over the seal, classifier, witness-ledger, and budget rows L, K, J, S. If two accepted terminal reads display the same rows $L, K, J, S, W, R, D, H, C, P, N$, then completion-facing consumers cannot distinguish their $\mathsf{Real}^{\uparrow}$ *terminal rows except through the displayed terminal coordinate of that finite packet. No quotient stream row, host real equality row, selected-limit row, countable-choice row, metric-completeness row, or bridge row participates in the pullback.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTerminalRoutePullback`

Proof. Project the carrier of Definition 1120.1. The terminal row E is not a detached Real object; it appears only as the $\mathsf{Real}^{\uparrow}$ coordinate of the same finite packet that already displays $L, K, J, S, W, R, D, H, C, P, N$.

Apply Definition 1120.2. The NameCert obligations expose the seal, classifier, witness, budget, StreamName, RegSeqRat, DyadicRatCore, terminal Real, transport, replay, provenance, and naming rows as one carrier surface. Hence the displayed terminal read is determined at the completion boundary by the same route data, not by an external selector.

Apply Definition 1120.4. The route first reads $\mathsf{StreamName}^{\uparrow}(W)$, then $\mathsf{RegSeqRat}^{\uparrow}(R)$, then $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and reaches $\mathsf{Real}^{\uparrow}(E)$ over L, K, J, S. This is the advertised pullback square at the consumer boundary: the terminal Real face is the fourth face attached to the preceding three displayed faces over the four governing rows.

Finally apply Definition 1120.3 and Definition 1120.6. Two accepted reads with the same displayed nonterminal route and the same structural rows can be compared only through the four-face inventory already exported by the packet. The theorem explicitly refuses quotient streams,

host real equality, selected limits, countable choice, metric completeness, and bridge rows, so those absent objects cannot distinguish the two completion-facing reads. □

Theorem 1120.10 (Real completion terminal seal determinacy)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

and

$$B' = (L', K', J', S', W', R', D', E', H', C', P', N')$$

be accepted RealCompletionExactBoundaryUp *packets. Suppose the rows* $L, K, J, S, W, R, D, H, C, P, N$ *and* $L', K', J', S', W', R', D', H', C', P', N'$ *agree through the displayed componentwise* hsame *comparison, the displayed* Cont *replay, the displayed* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundary}}$ *comparison. Then every completion-facing consumer identifies the terminal* $\mathsf{Real}^{\uparrow}$ *rows* E *and* E' *only through the displayed terminal coordinate of the finite packets. No quotient stream, host real equality, selected limit, countable choice, metric-completeness row, ambient-completeness row, or bridge row can distinguish the two terminal seal reads.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTerminalSealDeterminacy`

Proof. Begin with Definition 1120.9. The terminal $\mathsf{Real}^{\uparrow}$ read in each accepted packet is pulled back over the same displayed StreamName, RegSeqRat, and DyadicRatCore route above the governing rows L, K, J, S.

Apply Definition 1120.12. The assumed comparison of the nonterminal rows is exactly the comparison admitted by the finite boundary: componentwise hsame, continuation replay, package provenance, and the local NameCert row. Thus the two terminal reads meet over the same displayed prefix and structural tail, not over an external Real object.

Now use Definition 1120.6. The only completion-facing and terminal Real faces exported after the prefix are W, R, D, E in the first packet and W', R', D', E' in the second packet. Since the first three faces are already part of the displayed comparison, the remaining terminal coordinate is the only place where a completion-facing consumer can read the Real seal.

Finally apply Definition 1120.5, Definition 1120.3, and Definition 1120.14. These exclusions leave no quotient stream, host equality, selected-limit, choice, metric-completeness, ambient-completeness, or bridge coordinate with a carrier slot or insertion point. Hence any accepted consumer identifies E and E' only through the displayed terminal coordinate of the finite RealCompletionExactBoundaryUp packet. □

Theorem 1120.11 (Real completion seal-classifier witness balance)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet. Any completion-facing consumer that reaches one of the four terminal faces*

$$\mathsf{StreamName}^{\uparrow}(W), \quad \mathsf{RegSeqRat}^{\uparrow}(R), \quad \mathsf{DyadicRatCore}^{\uparrow}(D), \quad \mathsf{Real}^{\uparrow}(E)$$

must first display the seal and classifier rows L, K *and the witness-ledger and budget rows* J, S *of the same packet. Thus the four terminal faces cannot be consumed from the boundary as a detached tail; they are read only after the local seal, classifier, witness, and budget rows have met inside the accepted NameCert carrier.*

Proof. Project the carrier in Definition 1120.1. The rows L, K, J, S occur before W, R, D, E in the displayed packet, and the theorem concerns only consumers of that accepted finite carrier.

Apply Definition 1120.2. The NameCert obligations make L the local seal row, K the classifier row, J the witness-ledger row, and S the budget row of the same certificate surface. Hence a consumer that reads any later face of the carrier already has the four governing rows in scope.

Now use Definition 1120.6. The completion and terminal Real faces exported by the boundary are exactly W, R, D, E after the governing rows. Since the exactness theorem exposes no detached terminal tail, the four faces can be consumed only through the balanced seal, classifier, witness, and budget prefix. □

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundarySealClassifierWitnessBala`

Theorem 1120.12 (Real completion exact-boundary seal-budget prefix pullback)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet. Any consumer route that reaches one of the four faces*

$$\mathsf{StreamName}^{\uparrow}(W), \quad \mathsf{RegSeqRat}^{\uparrow}(R), \quad \mathsf{DyadicRatCore}^{\uparrow}(D), \quad \mathsf{Real}^{\uparrow}(E)$$

pulls back the same governing prefix L, K, J, S. The equality test for the four governing rows is only the displayed comparison through the componentwise hsame *rows H, the displayed* Cont *replay C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ *row N. It does not use quotient stream equality, host real equality, a selected limit, countable choice, metric completeness, ambient completeness, or a bridge coordinate.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundarySealBudgetPrefixPullback`

Proof. Project the carrier in Definition 1120.1. The accepted packet lists L, K, J, S before the four faces W, R, D, E, and the structural rows H, C, P, N are the only displayed rows available for comparing that prefix across consumer reads.

Apply Definition 1120.2. It identifies L as the completion-seal row, K as the classifier row, J as the witness-ledger row, and S as the synchronized budget row inside the same local NameCert surface. It also identifies H, C, P, N as the componentwise transport, continuation replay, package provenance, and local naming rows for that surface.

Use Definition 1120.11 and Definition 1120.6. The first theorem forces any consumer of the four later faces to display the governing prefix first, while the second theorem says that the only Real and completion faces behind that prefix are W, R, D, E.

Finally apply Definition 1120.3. Since the accepted carrier has no quotient stream, host equality, selected-limit, choice, completeness, ambient-completeness, or bridge coordinate, none of those absent objects can alter the pullback of L, K, J, S along any of the four displayed faces. □

Theorem 1120.13 (Real completion terminal consumer exactness)**.** *For every accepted* RealCompletionExactBound *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

every completion-facing consumer and every terminal $\mathsf{Real}^{\uparrow}$ *consumer reads exactly the finite route*

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

The route begins with the seal, classifier, witness-ledger, and budget rows, passes through the four completion and terminal faces W, R, D, E, and ends with the transport, continuation, package, and local naming rows H, C, P, N. No shorter route and no route with an added quotient, host-equality, selected-limit, choice, completeness, or bridge coordinate is a consumer route for the accepted boundary.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTerminalConsumerExactness`

Proof. Start from Definition 1120.2. The accepted NameCert surface lists the rows $L, K, J, S, W, R, D, E, H, C, P, N$ as one finite carrier inventory, so a consumer route for the accepted boundary is a route through that inventory.

Apply Definition 1120.12. It pulls the first four governing rows back unchanged along the completion and Real faces and restricts their equality test to the displayed hsame, Cont, Pkg, and local NameCert rows.

Next apply Definition 1120.3 and Definition 1120.6. The non-escape theorem rules out external completion coordinates, while the four-face theorem identifies W, R, D, E as the only completion-facing and terminal Real faces exported after L, K, J, S.

Use Definition 1120.7. When the boundary is presented through a finite cofinal-tail filter, the same four faces pass through the finite-tail-filter handoff rather than through a separate completion route.

Finally apply Definition 1120.9. The terminal $\mathsf{Real}^{\uparrow}(E)$ read is pulled back over the same displayed route and the rows H, C, P, N remain the trailing structural rows of the accepted packet. Therefore every completion-facing or terminal Real consumer has exactly the finite route displayed in the statement. □

Theorem 1120.14 (Real completion no completeness escape)**.** *For an accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

there is no insertion point between the accepted boundary and the terminal $\mathsf{Real}^{\uparrow}(E)$ *seal for a quotient stream, host real equality, selected limit, metric-completeness row, bridge row, or ambient completeness coordinate. Any such object lies outside the finite consumer route*

$$L, K, J, S, W, R, D, E, H, C, P, N$$

and therefore cannot mediate terminal Real consumption for the accepted boundary.

Proof. Project the accepted packet and apply Definition 1120.13. Terminal Real consumption is already fixed as the displayed finite route, so an added object would have to occupy a coordinate not present in the packet.

Apply Definition 1120.3 for quotient stream, host equality, selected-limit, choice, completeness, and bridge exclusions at the completion boundary. Apply Definition 1120.5 for the terminal $\mathsf{Real}^{\uparrow}(E)$ seal.

It remains only to compare the consumer point where such an object might be inserted. By Definition 1120.11, the governing rows L, K, J, S must already meet before W, R, D, E are consumed. By Definition 1120.9, the terminal Real face is the pullback of that same finite route. Hence quotient, equality, selected-limit, metric-completeness, bridge, and ambient-completeness coordinates have no packet-projection slot between the accepted boundary and the terminal Real seal. □

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryNoCompletenessEscape`

Theorem 1120.15 (Real completion roadmap off-table refusal)**.** *For every accepted* RealCompletionExactBoundaryU *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the only accepted completion-facing and terminal Real route remains the finite row list

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

Consequently the packet cannot be read as Real completeness, metric completeness, trichotomy, open-cover compactness, sequential compactness, Heine-Borel compactness, quotient-stream equality, host real equality, a selected-limit principle, countable choice, or a standard bridge.

Proof. Apply Definition 1120.13. It fixes every completion-facing or terminal Real consumer to the displayed finite route $L, K, J, S, W, R, D, E, H, C, P, N$.

Next use Definition 1120.6. The only completion-facing and terminal Real faces exported by an accepted packet are $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$, with no separate carrier for quotient streams, host equality, selected limits, compactness, trichotomy, choice, or bridge data.

Now apply Definition 1120.23. Even a certified terminal read through the finite-window and pullback normal forms is absorbed back into the same four-face packet with structural rows H, C, P, N; a second read adds no trichotomy, open-cover compactness, hidden selector, completeness, choice, or bridge row.

Finally use Definition 1120.14. Since the accepted boundary has no insertion point between the finite packet and the terminal $\mathsf{Real}^{\uparrow}(E)$ seal, the classical completeness and compactness readings listed in the statement are outside the route rather than latent consequences of it. □

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryRoadmapOffTableRefusal`

Theorem 1120.16 (Real completion exact-boundary mature route)**.** *For every accepted* RealCompletionExactBoun *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

consumption by a Real-facing or completion-facing route has exactly the finite mature route

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

The seal and classifier rows are L, K, the witness and budget rows are J, S, the four Phase Real and Completion faces are W, R, D, E, and the structural rows are H, C, P, N. When the sibling finite-tail-filter packet supplies the cofinal-tail surface, this route factors through that handoff. It exports no quotient stream, host real equality, selected limit, countable choice, metric completeness, ambient completeness, or bridge row.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryMatureRoute`

Proof. Project the carrier in Definition 1120.1 and apply Definition 1120.2. These identify the accepted packet rows $L, K, J, S, W, R, D, E, H, C, P, N$ and assign the seal, classifier, witness-ledger, budget, four face, transport, continuation, package, and local naming roles inside one finite inventory.

Next apply Definition 1120.6. The completion-facing and terminal Real coordinates of any accepted packet are exactly W, R, D, E, so the mature route cannot add a separate completion object or a second terminal Real face.

Use Definition 1120.7 for the sibling cofinal-tail surface. When that sibling supplies the finite-tail-filter presentation, the same rows factor through the handoff and no additional route coordinate is introduced.

Finally apply Definition 1120.13 and Definition 1120.14. The terminal consumer route is already the displayed finite row list, and quotient streams, host equality, selected limits, choice, completeness, and bridge rows have no carrier slot or insertion point. Hence the displayed route is the chapter's final finite consumer route. □

Theorem 1120.17 (Real completion exact-boundary tail cofinality schedule factorization)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet, and let*

$$T = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T)$$

be an accepted TailCofinalitySchedule$^{\uparrow}$ *packet through which the four faces* W, R, D, E *are consumed. Then the combined Real-completion and schedule consumer factors through the single exact-boundary route*

$$L, K, J, S, W, R, D, E, H, C, P, N$$

together with the schedule route

$$E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T.$$

The schedule is a consumer-side factor of the same four faces, not an added completion coordinate. The combined route exports no quotient stream, host real equality, selected limit, countable choice, ambient completeness, metric-completeness, hidden tail oracle, unbounded schedule, or machine bridge row.

Proof. Start with Definition 1120.6. It identifies W, R, D, E as the only StreamName, RegSeqRat, DyadicRatCore, and terminal Real faces exported by the accepted boundary after the governing rows L, K, J, S.

Next apply Definition 1120.13 and Definition 1120.16. These place every Real-facing or completion-facing consumer on the exact-boundary route $L, K, J, S, W, R, D, E, H, C, P, N$ and rule out quotient streams, host equality, selected limits, choice, completeness, and bridge rows as route coordinates.

Now attach the accepted TailCofinalitySchedule$^{\uparrow}$ packet. By Definition 681.9, every completion-facing seal read through the schedule consumes the ordered schedule rows E_T, W_T, D_T, R_T, L_T before the completion seal is available.

Finally use Definition 681.10. The shared-tail budget is already a finite consumer-side factor through the schedule route and its structural rows H_T, C_T, P_T, N_T. Since the exact-boundary route has only the four faces W, R, D, E available for the Real-completion handoff, the schedule can factor those faces only as a consumer of that route, not as a new completion coordinate or hidden tail source. □

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTailCofinalityScheduleFac`

Theorem 1120.18 (Real completion exact-boundary cofinal terminal determinacy)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

and

$$B' = (L, K, J, S, W, R, D, E', H, C, P, N)$$

be accepted RealCompletionExactBoundaryUp *packets. Suppose both packets are consumed through the same displayed* FiniteTailFilter$^{\uparrow}$ *packet and the same accepted* TailCofinalitySchedule$^{\uparrow}$ *route*

$$T = (E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T).$$

Then every completion-facing consumer and every terminal Real$^{\uparrow}$ *consumer identifies the terminal seals* Real$^{\uparrow}(E)$ *and* Real$^{\uparrow}(E')$ *only through the displayed cofinal schedule T and the exact-boundary terminal coordinate of the finite packets. The comparison exports no quotient stream equality, host real equality, selected limit, countable choice, ambient completeness, metric-completeness, hidden tail oracle, or bridge row.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryCofinalTerminalDeterminac`

Proof. Apply Definition 1120.7. Since both accepted packets enter through the same displayed finite-tail filter, the cofinal-tail entry for W, R, D, E and W, R, D, E' is already the sibling handoff surface, not a separate completion coordinate.

Next apply Definition 1120.17. The shared schedule rows $E_T, W_T, D_T, R_T, L_T, H_T, C_T, P_T, N_T$ are therefore consumed as a finite consumer-side factor of the same exact-boundary route. They do not create a second terminal Real row and do not supply a hidden tail source.

Use Definition 1120.9 and Definition 1120.10. The first binds each terminal Real$^{\uparrow}$ row to the displayed StreamName, RegSeqRat, and DyadicRatCore route over L, K, J, S; the second says that matching nonterminal and structural rows leave the terminal coordinate as the only Real-seal read available to completion-facing consumers.

Finally apply Definition 1120.14. The excluded quotient, host-equality, selected-limit, choice, completeness, hidden-tail, and bridge objects have no carrier slot in either packet and no insertion point in the shared schedule route. Thus any accepted consumer identifies the terminal seals only through the displayed cofinal schedule and the exact-boundary terminal coordinate. □

Theorem 1120.19 (Real completion exact-boundary obligation scope)**.** *For every accepted* RealCompletionExactBo *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the RealCompletionExactBoundaryUp *obligation scope is exactly the finite route containing the completion seal L, seal classifier K, witness ledger J, budget synchronizer S,* StreamName$^{\uparrow}$ *window W,* RegSeqRat$^{\uparrow}$ *readback R,* DyadicRatCore$^{\uparrow}$ *tolerance row D, terminal* Real$^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert$_{\text{RealCompletionExactBoundaryUp}}$ *row N. No quotient stream, host real equality, selected limit, countable-choice row, metric-completeness row, ambient-completeness coordinate, or bridge row belongs to that scope.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryObligationScope`

Proof. Project the carrier of Definition 1120.1. It displays exactly the packet rows $L, K, J, S, W, R, D, E, H, C, P, N$, with H, C, P, N serving only as transport, replay, provenance, and local naming rows for that same packet.

Apply Definition 1120.2. The NameCert surface names each displayed row as an obligation of the accepted carrier, so the seal, classifier, witness-ledger, budget, StreamName, RegSeqRat, DyadicRatCore, terminal Real, transport, continuation, package, and local naming rows all lie inside the same finite route.

Next use Definition 1120.6 and Definition 1120.13. These bind the four Phase Real and Completion faces to the consumer route and prevent a shorter or enlarged terminal route from serving as the accepted boundary.

Finally apply Definition 1120.14. The excluded quotient, host-equality, selected-limit, countable-choice, completeness, and bridge objects have no carrier slot and no consumer insertion point, so they are outside the obligation scope rather than dormant obligations of the packet. □

Theorem 1120.20 (Real completion exact-boundary scoped obligation closure)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the scoped obligation surface is exhausted by the seal row L, classifier row K, witness-ledger row J, synchronizer row S, StreamName$^{\uparrow}$ *row W,* RegSeqRat$^{\uparrow}$ *row R,* DyadicRatCore$^{\uparrow}$ *row D, terminal* Real$^{\uparrow}$ *row E, componentwise* hsame *transports H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert$_{\text{RealCompletionExactBoundaryUp}}$ *row N. No quotient stream, host real equality, selected limit, countable choice, metric-completeness, ambient-completeness, or bridge row is in scope.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryScopedObligationClosure`

Proof. Apply Definition 1120.2. It identifies the accepted carrier surface with the displayed rows $L, K, J, S, W, R, D, E, H, C, P, N$ and gives each row its local NameCert role.

Next apply Definition 1120.6. The completion-facing faces W, R, D, E are exactly the StreamName, RegSeqRat, DyadicRatCore, and terminal Real rows already present in that carrier surface, so there is no extra completion coordinate to add.

Use Definition 1120.13. Terminal Real consumption follows the same finite route and leaves H, C, P, N as the componentwise transport, continuation replay, package provenance, and local naming rows of the packet.

Finally apply Definition 1120.14. Quotient streams, host real equality, selected limits, countable choice, metric-completeness, ambient-completeness, and bridge rows have neither carrier slots nor consumer insertion points. Hence the displayed packet rows exhaust the scoped obligation surface. □

Theorem 1120.21 (Real completion exact-boundary public bridge factorization)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

the public exact-boundary surface factors the four Phase Real and Completion faces through the displayed packet $L, K, J, S, W, R, D, E, H, C, P, N$. The bridge is a paper-side factorization only: it exports no quotient stream, host real equality, selected limit, countable choice, ambient completeness, metric-completeness, or machine bridge row.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryUp StdBridge`

Proof. First apply Definition 1120.20. It places the accepted boundary on the finite route $L, K, J, S, W, R, D, E, H, C$ and leaves no carrier slot outside those rows.

Next use Definition 1120.6. The completion-facing and terminal Real faces are exactly the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows W, R, D, E inside the same packet.

Now apply Definition 1120.13. A public consumer of the exact-boundary surface can reach those four faces only through the seal, classifier, witness-ledger, synchronizer, transport, continuation, package, and local naming rows already present in B.

Finally use Definition 1120.14. Quotient streams, host real equality, selected limits, countable choice, ambient or metric completeness, and machine bridge rows have no coordinate in the displayed packet, so the factorization is a paper-side bridge and not a checked machine bridge. □

Theorem 1120.22 (Real completion exact-boundary terminal consumer exhaustion)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

every completion-facing or terminal Real consumer reads exactly the four Phase Real and Completion faces $\mathsf{StreamName}^{\uparrow}(W)$*,* $\mathsf{RegSeqRat}^{\uparrow}(R)$*,* $\mathsf{DyadicRatCore}^{\uparrow}(D)$*, and* $\mathsf{Real}^{\uparrow}(E)$*. Those reads occur after the governing rows* L, K, J, S *and through the structural rows* H, C, P, N*. No quotient stream, host real equality, selected limit, countable-choice row, metric-completeness row, ambient-completeness row, or bridge row participates in the consumer surface.*

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryFourFaceTerminalConsumerR`

Proof. Project the carrier of Definition 1120.1. It displays the accepted packet as the finite row list $L, K, J, S, W, R, D, E, H, C, P, N$, so a consumer has no coordinate outside the displayed carrier.

Apply Definition 1120.2 and Definition 1120.4. The NameCert obligations bind the governing rows L, K, J, S, and the displayed route carries StreamName and RegSeqRat consumption into the later completion faces.

Next use Definition 1120.6 and Definition 1120.9. These identify $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$ as the exact four completion-facing faces, with the terminal Real read pulled back over the same finite route.

Finally apply Definition 1120.13 and Definition 1120.14. The first theorem makes every completion-facing or terminal Real consumer pass through $L, K, J, S, W, R, D, E, H, C, P, N$, while the second rules out quotient streams, host real equality, selected limits, countable choice, metric completeness, ambient completeness, and bridge rows as carrier slots or insertion points. Hence the four displayed faces exhaust the terminal consumer surface. □

Theorem 1120.23 (Real completion exact-boundary terminal normal-form absorption)**.** *For every accepted* RealCompletionExactBoundaryUp *packet*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N),$$

consider a certified terminal read which first passes through the $\mathsf{StreamName}^{\uparrow}(W)$ *finite-window normal form, then through the* $\mathsf{RegularCauchyLimitClassifier}^{\uparrow}$ *finite pullback route over the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and finally through the exact-boundary consumer surface of* B*. The composite read is absorbed by the same four-face packet*

$$W, R, D, E$$

with structural rows H, C, P, N*.*

Applying the same terminal read a second time adds no quotient stream equality, host real equality, selected limit, countable-choice row, ambient completeness row, metric-completeness row, bridge row, trichotomy row, open-cover compactness row, or hidden tail selector.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryTerminalNormalFormAbsorpt`

Proof. First put the terminal read in StreamName normal form. By Definition 126.118, once the finite-window read exposes the displayed StreamName, DyadicRatCore, RegSeqRat, Real, and support rows, a further terminal pullback reads the same finite packet.

Next transport that normal form through the classifier. Apply Definition 443.15. The classifier handoff remains on the finite BHist, hsame, Cont, Pkg, and NameCert route and exports no quotient stream, host real equality, selected limit, hidden selector, unbounded schedule, or ambient completeness coordinate.

Now enter the exact-boundary surface. By Definition 1120.22, every completion-facing or terminal Real consumer of B reads exactly $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{RegSeqRat}^{\uparrow}(R)$, $\mathsf{DyadicRatCore}^{\uparrow}(D)$, and $\mathsf{Real}^{\uparrow}(E)$ through the structural rows H, C, P, N.

Finally apply Definition 1120.14. The off-table coordinates named above have no carrier slots or consumer insertion points in B, so the second terminal read is absorbed by the same four-face packet rather than producing a new equality, choice, completeness, bridge, trichotomy, compactness, or tail-selection row. □

Theorem 1120.24 (Real completion exact-boundary all rows necessary)**.** *For every accepted* RealCompletionExactBoundaryUp *read, validity is exactly the display of the carrier rows*

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

Deleting or replacing any seal, classifier, witness, or budget row L, K, J, S; any four-face row W, R, D, E; or any support row H, C, P, N leaves the exact-boundary carrier. In particular, no quotient stream, host real equality, selected limit, countable choice, metric completeness, ambient completeness, bridge row, or aggregate formalstatus row can substitute for a missing displayed row.

Lean4 ✓ `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryAllRowsNecessary`

Proof. Start from Definition 1120.2. The NameCert surface names the accepted carrier only as the finite inventory $L, K, J, S, W, R, D, E, H, C, P, N$, so the seal row, classifier row, witness ledger, and synchronizer are not optional parameters of a larger completion object.

Next apply Definition 1120.6. The completion and terminal Real faces are exactly W, R, D, E after L, K, J, S. Removing or replacing any one of these rows destroys the displayed four-face boundary rather than producing a second accepted face.

Now use Definition 1120.13 and Definition 1120.23. Every completion-facing and terminal consumer reads the whole finite route, and repeated terminal normal-form reading is absorbed by the same four-face packet with the same support rows H, C, P, N. Thus the transport, continuation, provenance, and local naming rows are necessary support for the accepted read.

Finally apply Definition 1120.3. Quotient streams, host real equality, selected limits, countable choice, metric completeness, ambient completeness, bridge rows, and aggregate formalstatus rows have no carrier slot and no consumer insertion point. Replacing any displayed row by one of those objects therefore leaves the exact-boundary packet rather than giving a different valid read. □

Theorem 1120.25 (Real completion exact-boundary tail-envelope criterion)**.** *An accepted* RealCompletionExactBou *packet admits a Real-facing completion read only when the tail schedule, cofinal terminal route,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal are present in one envelope over*

$$L, K, J, S, W, R, D, E, H, C, P, N.$$

Equivalently, the exact boundary rejects any Real-facing completion read that omits one of the tail schedule, cofinal terminal route, finite window, regular readback, dyadic tolerance, or terminal seal rows. The criterion exports no quotient stream equality, host real equality, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1120.17. It makes the accepted tail schedule a consumer-side factor of the same four faces W, R, D, E, not a separate completion coordinate.

Next apply Definition 1120.18. The cofinal terminal route can compare terminal seals only through the displayed schedule and the exact-boundary terminal coordinate.

Use Definition 1120.22. Every terminal consumer reads the StreamName, RegSeqRat, DyadicRatCore, and Real rows through the same support rows H, C, P, N after L, K, J, S.

Finally apply Definition 1120.24. Since every displayed row is necessary, omitting the tail schedule factor, cofinal terminal route, window row, readback row, tolerance ledger, or terminal seal leaves the accepted boundary. Quotient stream equality, host real equality, selected representatives, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness have no carrier slot in the criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$.

Constructive story: A RealCompletionExactBoundaryUp packet is generated from displayed seal, classifier, witness-ledger, synchronizer, StreamName, RegSeqRat, DyadicRatCore, terminal Real, transport, continuation, package, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$, maturePackageClosed)

Scope closed: Definition 1120.16 closes the concrete Real-completion finite route through $L, K, J, S, W, R, D, E, H, C, P, N$, and the displayed carrier keeps the TasteGate boundary on the same finite packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$, bridgeChecked)

Lean target: `BEDC.Derived.RealCompletionExactBoundaryUp.RealCompletionExactBoundaryUp\_StdBridge` Lean4 (definition) `BEDC.Derived.RealCompletionExactBoundaryUp.taste gate`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, or metric completeness.

Upgrade path: No further upgrade – chapter at the top of the formal axis.

The Lean TasteGate file at `lean4/BEDC/Derived/RealCompletionExactBoundaryUp/TasteGate.lean` supplies Nontrivial RealCompletionExactBoundaryUp and FieldFaithful RealCompletionExactBoundaryUp instances for the displayed finite carrier; bridge checking remains governed by the verification axis.

Lemma 1120.26 (Real completion exact-boundary forgetful projection)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet. The displayed boundary admits the finite target projection*

$$\mathrm{Forget}_{\mathrm{target}}(B) = (E, K, J, N),$$

where E is the terminal $\mathsf{Real}^{\uparrow}$ *target row, K is the displayed completion-seal classifier row, J is the displayed witness-ledger row, and N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ *row. This projection is exact for target-facing reads: every accepted consumer that asks only for the target row, classifier row, ledger row, and local NameCert row factors through (E, K, J, N), while any accepted read that distinguishes two packets with the same (E, K, J, N) must inspect one of the dropped rows L, S, W, R, D, H, C, P and is therefore not a read of the target projection. No quotient stream row, host real equality row, selected-limit row, countable-choice row, ambient-completeness row, metric-completeness row, or machine row is exported by this projection.*

Proof. Project the accepted carrier through Definition 1120.1. The displayed packet has exactly the rows $L, K, J, S, W, R, D, E, H, C, P, N$. Among them, the terminal target coordinate is E, the classifier coordinate is K, the witness-ledger coordinate is J, and the local naming coordinate is N.

Apply Definition 1120.2. It identifies the same finite row inventory as the local $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryU}}$ surface: K is read as classifier exposure, J as witness-ledger exposure, E as terminal $\mathsf{Real}^{\uparrow}$ seal exposure, and N as local naming. The remaining rows L, S, W, R, D, H, C, P are respectively seal, synchronizer, finite-window, readback, dyadic-tolerance, transport, replay, and provenance support for the full boundary.

Thus the target projection is obtained by dropping exactly the support rows and retaining (E, K, J, N). A consumer whose request is restricted to target, classifier, ledger, and local NameCert data has no coordinate in that request for L, S, W, R, D, H, C, P, so its read factors through the displayed four-row projection.

It remains to rule out hidden coordinates. By Definition 1120.3, every accepted consumer of the full boundary factors through the displayed finite route and receives no quotient stream equality, host real equality, hidden selector, second seal route, detached witness ledger, or ambient completeness coordinate. By Definition 1120.14, there is also no insertion point for selected limits, metric completeness, or a machine row between the accepted boundary and the terminal $\mathsf{Real}^{\uparrow}(E)$ seal. Therefore two accepted packets with the same (E, K, J, N) can be distinguished only by inspecting one of the dropped displayed support rows, which is outside the target projection, or by asking for an absent ambient coordinate, which is not exported by the packet. □

Theorem 1120.27 (Real completion exact-boundary substrate relation boundary)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet. Any reading of this packet into a Rule 110, cellular, binary,* FKernel*, or* BHist *substrate must be a typed relation certificate over the displayed rows*

$$L, K, J, S, W, R, D, E, H, C, P, N,$$

with relation kind one of bisimulation, row embedding, or substrate mirror. It cannot be a definitional equality between the Real-completion boundary and the chosen substrate. The source side of the certificate is the displayed Real-completion route, the target side is the named substrate read, and the direction is fixed by the consumer route from the seal, classifier, witness-ledger, and budget rows through W, R, D, E to H, C, P, N. Any certificate that uses an unlisted host coordinate, unlisted source coordinate, quotient stream equality, host real equality, selected limit, countable choice, metric-completeness, ambient-completeness, or machine bridge row is not a certificate for the accepted boundary. Any displayed row among $L, K, J, S, W, R, D, E, H, C, P, N$ that is needed by that relation remains named in the certificate and is not eliminated by the substrate interpretation.

Proof. Project the carrier in Definition 1120.1. Its full displayed inventory is exactly the seal row L, classifier row K, witness-ledger row J, synchronizer row S, StreamName finite-window row W, RegSeqRat readback row R, DyadicRatCore tolerance row D, terminal Real seal row E, componentwise transport row H, continuation replay row C, package provenance row P, and local naming row N.

Apply Definition 1120.2. The local NameCert surface exposes those rows as one finite packet and exposes no coordinate for quotient stream equality, host real equality, hidden selection, a second seal route, detached witness evidence, or ambient completeness. Hence a substrate-facing

read has no object from which to form a definitional equality with a Rule 110, cellular, binary, FKernel, or external BHist substrate.

Use Definition 1120.4 and Definition 1120.11. They force the consumer direction: the rows L, K, J, S govern the route, the rows W, R, D, E are the visible completion and terminal faces, and the rows H, C, P, N transport, replay, package, and name that same route. Thus the only admissible substrate comparison is relational data attached to this route. If the comparison is symmetric behavioral matching, it is a bisimulation; if it injects the displayed route rows into a substrate ledger, it is a row embedding; if it interprets the same visible route in the substrate vocabulary without identifying carriers, it is a substrate mirror.

Finally apply Definition 1120.20 and Definition 1120.14. The scoped obligation surface is exhausted by $L, K, J, S, W, R, D, E, H, C, P, N$, and the excluded host, equality, selection, choice, completeness, and machine rows have no carrier slot or insertion point. Therefore every accepted substrate relation certificate must be read from the displayed rows, every unlisted coordinate is rejected, and every noneliminable displayed row remains named at the boundary rather than being collapsed into definitional equality. □

Theorem 1120.28 (Real completion exact-boundary left-right mirror boundary)**.** *Let*

$$B = (L, K, J, S, W, R, D, E, H, C, P, N)$$

be an accepted RealCompletionExactBoundaryUp *packet. A proposed left-right mirror read of this boundary is accepted only in the following rowwise sense: its source side is the displayed Real-completion route*

$$L, K, J, S, W, R, D, E, H, C, P, N,$$

and its target side is an interpretation of exactly the same named history, observation, result, termination, provenance, and certificate rows. Thus the mirror has no independent right-side seal, no independent right-side witness ledger, no second terminal Real row, and no separate right-side certificate row. If a proposed counterpart is obtained by renaming the displayed rows while preserving the roles of $L, K, J, S, W, R, D, E, H, C, P, N$, it is only a substrate-mirror interpretation of the same packet; if it requires any unlisted right-side row, then the asymmetry is a boundary condition rather than an accepted counterpart.

Proof. Project the accepted carrier through Definition 1120.1. The carrier exposes exactly the rows $L, K, J, S, W, R, D, E, H, C, P, N$: the completion seal L, seal classifier K, witness ledger J, synchronizer S, StreamName window W, RegSeqRat readback R, DyadicRatCore tolerance row D, terminal $\mathsf{Real}^{\uparrow}$ seal E, componentwise transport H, displayed Cont replay C, package provenance P, and local $\mathsf{NameCert}_{\mathsf{RealCompletionExactBoundaryUp}}$ row N.

Apply Definition 1120.2. The local NameCert surface consists exactly of those rows. In particular, the seal and classifier side L, K and the witness and budget side J, S are not two independent boundary objects; they are governing rows of the same displayed packet, and the later rows W, R, D, E are consumed only inside that packet.

Now use Definition 1120.13. Every completion-facing or terminal $\mathsf{Real}^{\uparrow}$ consumer reads exactly the finite route $L, K, J, S, W, R, D, E, H, C, P, N$. Therefore a mirror read cannot add a right-side history row, a right-side observation row, a second result row, a second terminal row, or a separate certificate row without leaving the accepted consumer route.

Finally apply Definition 1120.27. A substrate mirror is permitted only as typed relation data over the displayed rows, with source side, target side, and direction fixed by the consumer route; it is not a definitional equality with a second carrier. Together with Definition 1120.3 and Definition 1120.14, this excludes quotient stream equality, host real equality, selected limits, countable

choice, metric completeness, ambient completeness, machine coordinates, and unlisted source or target coordinates. Hence the only accepted left-right mirror is the row-preserving interpretation of the same finite packet, and any stronger right-side counterpart is recorded as the exact-boundary asymmetry. □

Chapter 1121

A Concrete Naming Certificate for $\mathsf{ObserverBudgetSupport}^{\uparrow}$

The $\mathsf{ObserverBudgetSupport}^{\uparrow}$ packet realizes observer-local support readings from chapter 9420 as one finite support-and-budget carrier. It sits after the observer-coordinate surface of chapter 609, the selected observer budget of chapter 1042, the finite support surface of chapter 835, and the cross-Hist causal surface of chapter 478. Its role is to bind the local observer frame, finite support ledger, causal witness rows, and budget cap before any observer-local consumer reads a finite ledger slice as a coherent support surface.

Definition 1121.1 (Observer-budget support carrier)**.** *An* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, S, X, B, T, H, C, P, N)$$

with the following displayed rows:

1. F *is the* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *row naming the observer-local coordinate surface, not a global frame;*
2. S *is the* $\mathsf{FiniteSupport}^{\uparrow}$ *row listing the finite ledger entries visible to the observer;*
3. X *is the* $\mathsf{CrossHistCausal}^{\uparrow}$ *row recording the witnessed cross-Hist dependence used by the reading;*
4. B *is the* $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$ *row bounding the selected observation budget;*
5. T *is the support-transport row tying* F, S, X, B *into one observer-local route;*
6. H *records componentwise* hsame *transport for the frame, support, causal, budget, transport, provenance, and naming rows;*
7. C *records displayed* Cont *replay of the observer-coordinate read, support ledger read, cross-Hist causal read, budget read, and support transport;*
8. P *is the* Pkg *provenance over* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$, $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$, $\mathsf{FiniteSupport}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global observer frame, infinite support set, unbounded observation budget, unwitnessed cross-Hist relation, host spacetime object, quotient history equality, or detached support route.

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportTasteGate carrier recognition`

Theorem 1121.2 (Observer-budget support NameCert obligations)**.** *For the carrier of Definition 1121.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}$ *surface consists of five rows:*

1. *carrier admission for* $O = (F, S, X, B, T, H, C, P, N)$*;*
2. *observer-coordinate admission through* F*;*
3. *finite support and selected-budget admission through* S *and* B*;*
4. *cross-Hist causal visibility through* X *together with support transport through* T*;*
5. *componentwise* hsame *transport, displayed replay, provenance, and local naming through* H, C, P, N*.*

No observer-local reading through this packet can bypass one of these rows.

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportNameCert obligations`

Proof. Project the carrier. The packet contains exactly the frame row F, support row S, causal row X, budget row B, transport row T, and structural rows H, C, P, N.

Read the sibling surfaces. The observer-coordinate row supplies the local frame, the finite-support row supplies the displayed ledger support, the cross-Hist causal row supplies witnessed dependence, and the observer-budget row supplies the selected finite budget.

Finally inspect T, H, C, P, N. The support-transport row ties the four sibling rows into one route, while the structural rows preserve, replay, source, and name that route. Since the carrier contains no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 1121.3 (Observer-budget support nontrivial boundary)**.** *An accepted* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *packet is nontrivial exactly as a joint observer-local support-and-budget carrier: removing any one of* F, S, X, B, T *destroys one of observer-coordinate locality, finite support visibility, witnessed cross-Hist dependence, selected-budget boundedness, or the route tying them together. The packet is therefore not a renaming of any single sibling surface.*

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportNontrivialBoundary`

Proof. Start from the carrier in Definition 1121.1. The five content rows F, S, X, B, T play distinct roles in the accepted packet.

The row F supplies observer-coordinate locality; without it the packet has no local frame boundary. The row S supplies finite support; without it the reading has no finite ledger support. The row X supplies witnessed cross-Hist dependence; without it the route has no causal surface. The row B supplies the selected observation budget; without it the reading is unbounded. The row T ties those four rows into one route; without it the packet is only a list of sibling citations.

Thus the boundary is nontrivial at the carrier level. No projection to a single sibling can be inverted without recreating omitted rows. □

Theorem 1121.4 (Observer-budget support field faithfulness)**.** *Every field-faithful observer-local consumer of an accepted* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *packet must preserve the full tuple*

$$F, S, X, B, T, H, C, P, N.$$

In particular, it cannot keep the observer-coordinate row while dropping finite support, keep finite support while dropping the budget, keep the causal row while dropping transport, or read provenance and local naming as detached evidence.

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportFieldFaithfulness`

Proof. Use the NameCert obligations of Definition 1121.2. Each public read is assigned to one of the displayed rows $F, S, X, B, T, H, C, P, N$.

Field faithfulness is therefore rowwise. Dropping S erases the support ledger, dropping B erases the selected budget, dropping X erases the witnessed cross-Hist dependence, and dropping T disconnects the sibling rows from the observer-local route.

The structural rows are also part of the same carrier. Transport through H, replay through C, provenance through P, and local naming through N cannot be read as detached evidence because the carrier has no coordinate outside the displayed tuple. Hence a faithful consumer preserves all nine rows. □

Theorem 1121.5 (Observer-budget support route totality)**.** *Every accepted observer-local read of an* ObserverBudgetSupport$^{\uparrow}$ *packet routes through the finite tuple*

$$F, S, X, B, T, H, C, P, N.$$

The route may read the observer-coordinate row, finite support row, cross-Hist causal row, selected budget row, support transport row, componentwise hsame *row, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it has no coordinate for a global observer frame, infinite support set, unbounded budget, host spacetime object, or unwitnessed cross-Hist relation.*

Proof. Begin with the carrier of Definition 1121.1. It lists $F, S, X, B, T, H, C, P, N$ as the complete support-and-budget packet.

Apply the NameCert obligations of Definition 1121.2. They force observer-coordinate admission through F, finite support through S, selected budget through B, causal visibility through X, and support transport through T before the structural rows can be consumed.

Finally use the nontrivial boundary and field-faithfulness rows from Definition 1121.3 and Definition 1121.4. Since no faithful consumer can drop any content row, every accepted route is total on the displayed tuple. □

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportRouteTotality`

Theorem 1121.6 (Observer-budget support ledger exactness)**.** *In an accepted* ObserverBudgetSupport$^{\uparrow}$ *packet, finite support, selected budget, cross-Hist causal evidence, and support transport are exact ledger rows rather than detached evidence. A consumer reads S, B, X, and T only as rows of the same carrier*

$$F, S, X, B, T, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportLedgerExactness`

Proof. Project the carrier. The finite support row S, selected budget row B, causal row X, and transport row T appear in the same packet as the observer-coordinate and structural rows.

Use the NameCert obligations to read these rows together. The local certificate does not allow the support row to detach from the budget, the causal evidence to detach from transport, or the provenance and naming rows to certify a different object.

Now apply the nontrivial boundary and field-faithfulness theorems. Removing S, B, X, or T destroys the intended observer-local support-and-budget packet, so these rows are exact ledger content and not replaceable external evidence. □

Theorem 1121.7 (Observer-budget support public readback)**.** *Every public readback of an accepted* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *packet consumes the observer-coordinate row* F*, finite support row* S*, cross-Hist causal row* X*, selected budget row* B*, and support route row* T *together with the structural rows* H, C, P, N*. Thus public readback cannot expose support, causal evidence, budget, or route as independent host data.*

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportPublicReadback`

Proof. Start with route totality from Definition 1121.5. Any accepted observer-local read already passes through $F, S, X, B, T, H, C, P, N$.

Add ledger exactness from Definition 1121.6. The support, budget, causal, and route rows are not detachable evidence; they are ledger rows of the same accepted carrier.

Finally apply field faithfulness from Definition 1121.4. A public consumer that drops any of F, S, X, B, T no longer preserves the accepted packet, so every public readback consumes the full observer-coordinate, support, causal, budget, and route surface together. □

Theorem 1121.8 (Observer-budget support obligation-closure package)**.** *The carrier, NameCert obligations, nontrivial boundary, field-faithfulness, route-totality, ledger-exactness, and public-readback rows assemble the obligation-closure surface for* $\mathsf{ObserverBudgetSupport}^{\uparrow}$*. Every accepted observer-local support reading is bound to*

$$F, S, X, B, T, H, C, P, N$$

through the displayed BHist *carrier, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupportObligationClosurePackage`

Proof. Start with the carrier of Definition 1121.1 and the NameCert obligations of Definition 1121.2. They display the observer-coordinate, finite-support, cross-Hist causal, selected-budget, support-transport, and structural rows.

Use the nontrivial boundary and field-faithfulness theorems, Definition 1121.3 and Definition 1121.4. Together they show that no accepted consumer can collapse the packet to one sibling row or drop any displayed field.

Finally apply route totality, ledger exactness, and public readback from Definition 1121.5, Definition 1121.6, and Definition 1121.7. These results bind observer-local support, selected budget, causal evidence, transport, replay, provenance, and local naming into one obligation surface. □

Theorem 1121.9 (Observer-budget support bridge surface)**.** *Every bridge-facing observer-local support export of an accepted* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *packet factors through the finite tuple*

$$F, \quad S, \quad X, \quad B, \quad T, \quad H, \quad C, \quad P, \quad N.$$

The observer-coordinate row F*, finite-support row* S*, cross-*BHist *causal row* X*, selected budget row* B*, support-transport row* T*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}$ *row* N *are preserved as one bridge surface. No bridge-facing export may replace them by global observer frames, infinite support, unbounded budgets, detached host spacetime data, quotient history equality, or unwitnessed cross-*BHist *dependence.*

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.ObserverBudgetSupport bridge surface`

Proof. First read the local certificate. The carrier and NameCert obligations, Definition 1121.1 and Definition 1121.2, display exactly the observer-coordinate, finite-support, causal, budget, transport, replay, provenance, and naming rows.

Next pass through the public surface. Public readback, Definition 1121.7, requires every accepted observer-local read to consume $F, S, X, B, T, H, C, P, N$ together rather than as detachable host data.

Finally use the obligation package and ledger exactness. Theorem 1121.8 assembles the same finite tuple as the chapter surface, and Theorem 1121.6 rules out replacing support, budget, causal evidence, or route transport by external witnesses. Thus a bridge-facing export preserves precisely the displayed tuple, with no additional coordinate for the excluded objects. □

The later Lean `TasteGate.lean` development for $\mathsf{ObserverBudgetSupport}^{\uparrow}$ will need explicit $\mathsf{Nontrivial\ ObserverBudgetSupport}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObserverBudgetSupport}^{\uparrow}$ instances. These instances must preserve the observer-coordinate row, finite-support row, cross-Hist causal row, selected-budget row, support-transport row, componentwise transport, displayed replay, provenance, and local naming row.

Falsifiable prediction: If an accepted observer-local reading keeps the observer-coordinate row while replacing finite support, selected budget, cross-Hist causal evidence, or support transport by detached host data, it refutes the $\mathsf{ObserverBudgetSupport}^{\uparrow}$ classifier.

Independence witness: observercoordinateframe, observerfilterbudgetselector, finitesupport, crosshistcausal, observationbudgetlimiter: none is bijective to $\mathsf{ObserverBudgetSupport}^{\uparrow}$ because the carrier binds observer-local frame, finite support, cross-Hist causal evidence, selected budget, route transport, structural replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverBudgetSupport}^{\uparrow}$ packet is generated from observer-coordinate, finite-support, cross-Hist causal, selected-budget, support-transport, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.ObserverBudgetSupportUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1121.1, Definition 1121.2, Definition 1121.3, Definition 1121.4, Definition 1121.7, Definition 1121.8, and Definition 1121.9 bind the observer-local support obligations, public readback row, obligation-closure package, and bridge-facing export surface to BHist, hsame, Cont, Pkg, NameCert, $\mathsf{ObserverCoordinateFrame}^{\uparrow}$, $\mathsf{FiniteSupport}^{\uparrow}$, $\mathsf{ObserverFilterBudgetSelector}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, and the support-transport row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverBudgetSupport}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ObserverBudgetSupportUp.taste\_gate`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close bridge checking, global observer frames, infinite support, unbounded budgets, host spacetime objects, quotient history equality, or unwitnessed cross-Hist dependence.

Upgrade path: Bridge checking remains a separate verification-axis obligation for the observer-local support export surface.

Chapter 1122

A Concrete Naming Certificate for $\mathsf{ObserverSupportLattice}^{\uparrow}$

The $\mathsf{ObserverSupportLattice}^{\uparrow}$ packet realizes the observer support direction of chapter 9420 as a finite NameCert carrier. It sits after the observer-coordinate frame of chapter 609, the finite support surface of chapter 835, the locality-cell surface of chapter 509, the observer-budget support packet of chapter 1121, and the conscious observer state of chapter 578. Its role is to bind local frame, finite support, locality cell, budget support, route, transport, provenance, and local naming rows before any observer-local support consumer treats a support read as stable observer evidence.

Definition 1122.1 (Observer-support lattice carrier)**.** *An* $\mathsf{ObserverSupportLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, C, B, R, H, Q, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *row naming the local observer frame;*
2. *S is the* $\mathsf{FiniteSupport}^{\uparrow}$ *row naming the finite support ledger available to that frame;*
3. *C is the* $\mathsf{LocalityCell}^{\uparrow}$ *row that bounds the local spatial and relation-cell read;*
4. *B is the* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *row restricting which support reads are budget-admissible;*
5. *R is the observer-local support route tying F, S, C, B to the conscious observer state consumer;*
6. *H records componentwise* hsame *transport for the frame, support, locality, budget, route, provenance, and naming rows;*
7. *Q records displayed* Cont *replay of the support-lattice read;*
8. *P is the* Pkg *provenance over* $\mathsf{ObserverSupportLattice}^{\uparrow}$, $\mathsf{ObserverCoordinateFrame}^{\uparrow}$, $\mathsf{FiniteSupport}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, $\mathsf{ConsciousObserverState}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverSupportLattice}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global observer frame, infinite support set, detached spacetime manifold, unbudgeted support read, quotient history equality, or host-level consciousness object.

Lean4 ✓ `BEDC.Derived.ObserverSupportLatticeUp.ObserverSupportLatticeTasteGate single carrier alignment`

Theorem 1122.2 (Observer-support lattice NameCert obligations)**.** *For the carrier of Definition 1122.1, the local* $\mathsf{NameCert}_{\mathsf{ObserverSupportLattice}^{\uparrow}}$ *surface consists of five rows:*

1. *carrier admission for* $L = (F, S, C, B, R, H, Q, P, N)$*;*
2. *frame and conscious-state anchoring through* $\mathsf{ObserverCoordinateFrame}^{\uparrow}$ *and* $\mathsf{ConsciousObserverState}^{\uparrow}$*;*
3. *finite support and locality-cell admission through* S *and* C*;*
4. *budget support and route exactness through* B *and* R*;*
5. *componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming through* H, Q, P, N*.*

No accepted observer-local support read can bypass one of these rows.

Proof. Project the carrier of Definition 1122.1. The packet contains exactly the frame row F, support row S, locality-cell row C, budget-support row B, route row R, and structural rows H, Q, P, N.

Read the sibling surfaces in the displayed order. The observer-coordinate frame anchors the local observer frame, the finite support row supplies the visible ledger entries, the locality cell bounds the local relation read, and the observer-budget support row records which support reads are admissible for the consumer.

Finally attach R, H, Q, P, N. The route row ties the four sibling surfaces to the observer-local read, while the structural rows preserve, replay, source, and name that route. Since the carrier contains no extra coordinate, the five listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ObserverSupportLatticeUp.ObserverSupportLatticeCarrier namecert obligations`

Theorem 1122.3 (Observer-support lattice route exactness)**.** *Every observer-local support read of an accepted* $\mathsf{ObserverSupportLattice}^{\uparrow}$ *packet factors through the lattice route*

$$F \longmapsto S \longmapsto C \longmapsto B \longmapsto R \longmapsto H \longmapsto Q \longmapsto P \longmapsto N.$$

In particular, a consumer cannot preserve the observer frame and finite support rows while replacing the locality-cell or budget-support route by host spacetime data, an unbounded read, a quotient of histories, or a detached conscious-state object.

Lean4 ✓ `BEDC.Derived.ObserverSupportLatticeUp.ObserverSupportLatticeCarrier route exactness`

Proof. Begin with the NameCert obligations of Definition 1122.2. They require every accepted read to expose the carrier row, frame and conscious-state anchoring, finite support, locality cell, budget support, route, transport, replay, provenance, and local naming rows.

The support read is therefore routed through F, S, C, B, R before any structural row is consumed. The frame row alone supplies no support ledger, the finite support row alone supplies no locality-cell boundary, and the locality-cell row alone supplies no budget-admissible support consumer.

It remains to rule out replacements. Because H, Q, P, N preserve and replay the same displayed route, a consumer that swaps C or B for host spacetime data, an unbounded read, quotient history equality, or a detached state object no longer reads the accepted carrier. Hence every observer-local support read factors through the lattice route exactly. □

The later Lean `TasteGate.lean` development for $\mathsf{ObserverSupportLattice}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ObserverSupportLattice}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObserverSupportLattice}^{\uparrow}$ instances with `witness_pair` and field-preservation witnesses. These instances must preserve the frame row, finite-support row, locality-cell row, budget-support row, route row, componentwise transport, displayed replay, provenance, and local naming row.

Falsifiable prediction: If an observer-local support consumer preserves the frame and finite support rows while replacing the locality-cell or budget-support route by host spacetime data, it refutes the $\mathsf{ObserverSupportLattice}^{\uparrow}$ classifier.

Independence witness: observercoordinateframe, finitesupport, localitycell, observerbudgetsupport, consciousobserverstate: none is bijective to $\mathsf{ObserverSupportLattice}^{\uparrow}$ because only the lattice packet carries frame, support, locality, budget, route, transport, provenance, and local NameCert rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverSupportLattice}^{\uparrow}}$.

Constructive story: The observer-support lattice is generated by collecting the observer-indexed support rows already present in a $BHist$ ledger and ordering them only by the carrier's visible inclusion convention. Each support cell is read through $hsame$, so two histories that present the same observer support cannot be separated by a hidden host equality test. Continuation information appears as Cont-stable route extension: a later observation may extend a support row, but it must keep the lattice address and carrier evidence already recorded. The NameCert route is therefore built bottom-up from support rows, lattice comparability, and non-escaping observer routes rather than from an external set-theoretic lattice.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverSupportLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the observer-support lattice carrier, its NameCert obligations, and the route-exactness theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverSupportLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a complete ambient lattice, observer identity outside the ledger, or joins computed by host equality.

Upgrade path: Reaching obligationClosure requires formal carriers for support rows and checked $hsame$ preservation of the route-exactness argument.

Chapter 1123

A Concrete Naming Certificate for $\mathsf{CellularBoundaryCondition}^{\uparrow}$

The $\mathsf{CellularBoundaryCondition}^{\uparrow}$ packet realizes the edge-policy surface left implicit by the cellular-carrier alternative of chapter 9391. It is a composite over the local update packet of chapter 984, the substrate carrier of chapter 942, the carrier-encoding packet of chapter 943, the substrate classifier of chapter 945, and the row classifier of chapter 941. Its datum is not another orbit evaluator; it is the finite edge policy by which local reads at the boundary are admitted, refused, or padded before a local update rule may inspect them.

Falsifiable prediction: No accepted $\mathsf{CellularBoundaryCondition}^{\uparrow}$ packet exports a global orbit, hidden evaluator, or unregistered boundary row beyond its displayed edge policy.

Independence witness: cellularlocalupdaterule, cellularsubstrate, cellularcarrierencoding, cellularsubstrateclassifier, cellularrowclassifier: none is bijective to $\mathsf{CellularBoundaryCondition}^{\uparrow}$ because the listed siblings respectively carry local transition, substrate, encoding, substrate-classifier, or row-classifier rows, while this packet carries the finite edge-policy rows that determine which boundary reads are admitted before local update.

Definition 1123.1 (Cellular boundary condition carrier)**.** *A* $\mathsf{CellularBoundaryCondition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite edge-policy row, displaying the admitted boundary convention such as fixed edge, periodic edge, reflected edge, refused edge, or explicitly padded edge.*
2. *W is the boundary-window row, naming the finite cell positions at which a local neighbourhood would otherwise cross the displayed row boundary.*
3. *R is the local update-read row, citing the* $\mathsf{CellularLocalUpdateRule}^{\uparrow}$ *surface whose neighbourhood reads are constrained by E and W.*
4. *A is the admission ledger recording which boundary reads are accepted, which are refused, and which are filled by a displayed padding row.*
5. *H records componentwise* hsame *transport for the edge policy, boundary window, update-read row, admission ledger, continuation route, package provenance, and local name.*
6. *C records the displayed* Cont *route through E, W, R, A, H, P, N.*

7. *P is the* Pkg *provenance over* CellularBoundaryCondition$^{\uparrow}$, CellularLocalUpdateRule$^{\uparrow}$, CellularSubstrate$^{\uparrow}$, CellularCarrierEncoding$^{\uparrow}$, CellularSubstrateClassifier$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularBoundaryCondition}^{\uparrow}}$ *row.*

The classifier for B reads only the displayed edge policy, boundary window, constrained local update read, admission ledger, transport row, continuation route, provenance, and local name. It has no coordinate for a global orbit, host evaluator, unregistered boundary row, unstated padding convention, or detached substrate classifier.

Lean4 ✓ `BEDC.Derived.CellularBoundaryConditionUp.CellularBoundaryConditionCarrierSemanticNameCert`

Theorem 1123.2 (CellularBoundaryCondition NameCert obligations)**.** *For the carrier of Definition 1123.1, the local* $\mathsf{NameCert}_{\mathsf{CellularBoundaryCondition}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$B = (E, W, R, A, H, C, P, N),$$

edge-policy exposure through E, boundary-window exposure through W, local update-read accountability through R, admission and refusal replay through A, componentwise stability through H, displayed route replay through C, package provenance through P, and local naming through N. A consumer may use the packet as boundary-condition evidence only through those rows.

Proof. Carrier admission is projection from Definition 1123.1. The packet displays the edge policy, boundary window, local update-read row, admission ledger, transport row, continuation route, provenance row, and local name before any consumer can read the boundary surface.

The cellular content is split between E, W, R, A. The edge-policy row E states the accepted boundary convention, the window row W identifies the finite neighbourhood reads that meet the edge, and the update-read row R records which local rule surface is constrained by that edge policy.

Admission and refusal are read from A. This ledger records accepted boundary reads, refused boundary reads, and explicitly padded reads, so the consumer cannot replace the displayed edge policy by an unregistered boundary row.

Finally use H, C, P, N. These rows transport, replay, package, and name the same finite packet, and they introduce no global orbit, host evaluator, hidden padding convention, or detached substrate classifier. □

Lean4 ✓ `BEDC.Derived.CellularBoundaryConditionUp.CellularBoundaryConditionTasteGate single carrier alignment`

Theorem 1123.3 (CellularBoundaryCondition non-escape boundary)**.** *Every accepted* CellularBoundaryCondition$^{\uparrow}$ *public read factors through the finite route*

$$E, W, R, A, H, C, P, N.$$

The route may expose the displayed edge policy, the affected boundary windows, the constrained local update-read row, and the admission ledger. It cannot export a global cellular orbit, a hidden evaluator, an unregistered boundary row, an unstated padding convention, or a substrate-wide classifier.

Lean4 ✓ `BEDC.Derived.CellularBoundaryConditionUp.CellularBoundaryConditionNonescapeBoundary`

Proof. Start with the NameCert obligation surface of Definition 1123.2. Every public read reaches the local name only after the edge-policy row, boundary-window row, local update-read row, admission ledger, transport row, continuation route, and provenance row have been displayed.

The local update read is present only as R. It can use the boundary policy named by E and the finite boundary window named by W, but it cannot import the global orbit or a host evaluator as an extra source of boundary cells.

The admission ledger A separates accepted, refused, and padded boundary reads. Because the padding row must be displayed when it is used, an unstated padding convention or unregistered boundary row has no coordinate in the packet.

Thus any read of a global orbit, hidden evaluator, unregistered boundary row, unstated padding convention, or substrate-wide classifier would require data absent from $B = (E, W, R, A, H, C, P, N)$. Such a read is outside the accepted boundary-condition certificate. □

The Lean TasteGate surface expected for this packet is `BEDC.Derived.CellularBoundaryConditionUp.TasteG` Its paper-side carrier is already fixed by E, W, R, A, H, C, P, N and by the non-escape boundary above.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularBoundaryCondition}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularBoundaryCondition}^{\uparrow}$ packet is generated from a finite edge-policy row, boundary-window row, constrained local update-read row, admission ledger, componentwise hsame transports, displayed continuation route, package provenance, and local NameCert row. Its composite dependency surface is the edge-policy, boundary-window, local update-read, admission-ledger, continuation-route, package-provenance, and local naming surfaces.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularBoundaryCondition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1123.1, Definition 1123.2, and Definition 1123.3 bind the finite edge-policy carrier, its local NameCert obligations, and its non-escape boundary for local update reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularBoundaryCondition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global cellular orbit evaluation, host evaluator correctness, substrate universality, complete row-classifier coverage, Lean implementation, or standard bridge.

Upgrade path: Checked closure requires Lean encoding of the carrier plus Nontrivial and FieldFaithful TasteGate instances preserving the edge policy, boundary window, constrained local update read, admission ledger, route, provenance, and local naming projections.

Chapter 1124

A Concrete Naming Certificate for $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$

The $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ interface realizes the physical model audit program of section 9414.60 as a finite BEDC naming certificate. It is a composite over the finite stream and real-name surfaces of chapter 126 and chapter 38: those chapters supply finite observation names and real-valued readback rows, while this chapter packages the physical audit rows that decide whether a model-facing fit claim may pass an acceptance gate.

Definition 1124.1 (Reality-constrained model-audit carrier)**.** *A* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (H, \Pi, O, M, C, T, L, F, S)$$

whose displayed rows are as follows:

1. H *is the finite source-trace history for the audited physical system;*
2. Π *is the observation bundle naming the finite queries through which the system is read;*
3. O *is the observed-signature row extracted from* H *under* Π*;*
4. M *is the model-generated signature row declared by the candidate model;*
5. C *is the classifier row comparing* O *and* M *at the declared physical-fit level;*
6. T *is the approximation-tower row by which source traces, observations, and model signatures are related;*
7. L *is the audit-ledger row recording provenance, hidden compression, translation, and idealization entries;*
8. F *is the failure-surface row naming the finite defeating observations or signature mismatches that would invalidate the declared claim;*
9. S *is the audit-strength row, ranging from seed-level model support through reproduced and bridged fit surfaces.*

The carrier is exactly this finite packet. It contains no unrecorded source row, no hidden compression step, no external oracle for model truth, and no physical-fit claim outside the displayed ledger.

Lean4 ✓ BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditCarrierSemanticNameCert

Definition 1124.2 (Reality-constrained model-audit classifier)**.** *For accepted packets*

$$A = (H, \Pi, O, M, C, T, L, F, S) \quad \textit{and} \quad A' = (H', \Pi', O', M', C', T', L', F', S'),$$

the predicate $\mathsf{RealityConstrainedModelAuditClassifier}(A, A')$ *holds when the component rows are related by displayed* hsame *transports, the observation bundle routes replay through the same finite source-trace schedule, the observed and model-generated signatures preserve the classifier comparison, the approximation tower and audit ledger transport componentwise, the failure-surface rows name the same defeating tests, and the declared strength comparison is preserved.*

Thus classifier agreement is not raw equality of models. It is hsame-stable agreement of the source, observation, signature, classifier, tower, ledger, failure, and strength rows together with preservation of the declared physical-fit comparison.

Theorem 1124.3 (Reality-constrained model-audit NameCert obligations)**.** *For the carrier of Definition 1124.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedModelAudit}^{\uparrow}}$ *surface consists of the following five obligation rows:*

1. *carrier admission: the accepted object is exactly* $A = (H, \Pi, O, M, C, T, L, F, S)$*;*
2. *classifier stability: every accepted comparison uses the classifier of Definition 1124.2;*
3. *signature exactness: the observed row* O *and model-generated row* M *are the only signatures licensed for the fit claim;*
4. *ledger exactness: every compression, translation, idealization, and hidden-content row used by the audit appears in* L*;*
5. *non-escape: a fit claim that depends on a row absent from* $H, \Pi, O, M, C, T, L, F, S$ *is outside the acceptance gate.*

Lean4 ✓ BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditTasteGate
single carrier alignment

Proof. Carrier admission projects the nine rows displayed in Definition 1124.1. Each accepted read begins from the same finite packet, so no additional source trace or model-signature coordinate can be introduced later in the certificate.

Classifier stability follows by unpacking Definition 1124.2. The transports compare the source, observation, signature, tower, ledger, failure, and strength rows componentwise and preserve the declared fit comparison.

The signature and ledger obligations are the distinguished projections of the same packet. Rows O and M carry the comparison surface, while L records the compression, translation, idealization, and hidden-content material consumed by that comparison.

The non-escape clause is forced by the carrier inventory. Since the packet has no coordinate beyond $H, \Pi, O, M, C, T, L, F, S$, a claim depending on any unlisted row cannot be projected by the local naming certificate. □

Theorem 1124.4 (Reality-constrained model-audit ledger non-escape)**.** *A model-audit packet with an unledgered compression row cannot pass the acceptance gate for any fit claim whose signature depends on that hidden row. The accepted route must expose the source trace, observation bundle, observed signature, model-generated signature, classifier, approximation tower, audit ledger, failure surface, and strength row before the fit claim is read.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditLedgerNonescape`

Proof. Read the NameCert obligations of Definition 1124.3. Ledger exactness places every compression or translation row used by the fit comparison inside L.

If a fit claim depends on a hidden compression row not present in L, then the claim asks the acceptance gate to read a coordinate outside $A = (H, \Pi, O, M, C, T, L, F, S)$. The non-escape obligation excludes exactly that route.

Therefore the acceptance gate can pass only claims whose signature dependence is visible in the displayed ledger. The unledgered row remains outside the accepted packet. □

Theorem 1124.5 (Reality-constrained model-audit fit separation)**.** *Two accepted* RealityConstrainedModelAudit$^{\uparrow}$ *packets may agree on source traces and observation bundles while remaining separated as physical audit certificates whenever their declared physical-fit comparison, failure surface, or strength row is not preserved by the classifier. This separation is visible at the certificate level and does not require a host-side judgment that either model is false.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditFitSeparation`

Proof. Use the classifier definition. Agreement of H and Π supplies only the source and observation-bundle transports.

The same classifier also requires preservation of the observed and model-generated signatures, the physical-fit comparison, the tower, the ledger, the failure surface, and the strength row. If any of the comparison, failure, or strength rows is not preserved, the classifier relation fails even when the source rows agree.

The failure of classifier transport is an audit separation, not a truth judgment. It says that the two packets do not present the same licensed physical-fit certificate. □

Theorem 1124.6 (Reality-constrained model-audit signature exactness)**.** *Every accepted* RealityConstrainedModelA *audit route for*

$$A = (H, \Pi, O, M, C, T, L, F, S)$$

reads exactly the observed row O and the model row M extracted from the same source trace H and observation bundle Π. No alternative observed signature, model signature, classifier input, approximation witness, ledger entry, failure row, or strength row can be substituted without leaving the accepted packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditSignatureExactness`

Proof. Start with carrier admission in Definition 1124.3. It identifies the accepted object with the nine displayed rows of Definition 1124.1, so the audit route has no coordinate from which a second observed or model-generated signature can be read.

The classifier of Definition 1124.2 then binds the signature comparison to the same H and Π transports. Since classifier stability preserves the displayed comparison row C, any route using different signature inputs fails to be the same accepted classifier route.

Ledger non-escape in Definition 1124.4 excludes a hidden replacement through compression, translation, or idealization. The fit-separation theorem shows that a changed comparison row is an audit separation rather than another certificate for the same route. □

Theorem 1124.7 (Reality-constrained model-audit failure-surface exhaustion)**.** *For an accepted* RealityConstrainedModelAudit$^{\uparrow}$ *fit claim on* $A = (H, \Pi, O, M, C, T, L, F, S)$*, every defeating observation and every signature mismatch used against the claim is named in the finite failure-surface row* F*. A defeat that requires an observation route or mismatch row absent from* F *is outside the displayed audit certificate.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditFailureSurfaceExhaustio`

Proof. The NameCert obligation surface fixes F as one of the carrier rows and makes non-escape part of the accepted audit. Thus a defeating test used by the certificate must be a projection from A, not an extra host-side objection added after acceptance.

Classifier stability requires the failure-surface rows of compared packets to name the same defeating tests. If a mismatch is not present in F, the classifier has no component on which to transport it as part of the accepted fit comparison.

Ledger non-escape also blocks hiding the missing defeat inside L. The ledger may expose provenance, compression, translation, and idealization rows, but it does not create an unlisted failure coordinate. Hence the finite row F exhausts the defeating observations and signature mismatches licensed by the audit route. □

Theorem 1124.8 (Reality-constrained model-audit strength-row transport)**.** *Whenever* RealityConstrainedModelAu *holds for*

$$A = (H, \Pi, O, M, C, T, L, F, S) \quad \textit{and} \quad A' = (H', \Pi', O', M', C', T', L', F', S'),$$

the classifier transports the audit-strength row S *to* S'*. Therefore classifier agreement cannot upgrade the evidential grade of a fit claim unless the transported strength row already presents that grade.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditStrengthRowTransport`

Proof. Unpack the classifier definition. Alongside the source, observation, signature, tower, ledger, and failure rows, it includes preservation of the declared strength comparison.

The NameCert obligations and ledger non-escape prevent the strength grade from being imported through an unrecorded coordinate. The accepted packet has only the displayed row S for audit strength, and any grade used by the fit claim must be read from that row.

Finally apply Definition 1124.5. If strength is not preserved, the packets are separated as physical audit certificates even when their source and observation rows agree. Thus transported classifier agreement carries S; it does not manufacture a stronger evidential grade. □

The later Lean file `lean4/BEDC/Derived/RealityConstrainedModelAuditUp/TasteGate.lean` must provide `Nontrivial RealityConstrainedModelAuditUp` with two distinct audit packets and `FieldFaithful RealityConstrainedModelAuditUp` by `.mk` projections from $H, \Pi, O, M, C, T, L, F, S$ and injection on the displayed field surface.

Falsifiable prediction: A model-audit packet with an unledgered compression row cannot pass the acceptance gate for any fit claim whose signature depends on the hidden row.

Independence witness: Compare largemodelauditchannel, largemodelcorpussupply, physicalfit, finitewindowenvelope, and streamname: those siblings supply channel, corpus, window, and stream rows, but none carries the ten-field physical audit certificate with failure-surface and strength coordinates, so no bijection preserves the obligation rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedModelAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ packet is generated from a finite source-trace history, observation bundle, observed signature, model-generated signature, classifier comparison, approximation tower, audit ledger, failure surface, and audit-strength row. Its composite dependency surface is the source-trace, observation-bundle, signature-comparison, approximation-tower, audit-ledger, failure-surface, and audit-strength surfaces.

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelAuditUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedModelAudit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1124.1, Definition 1124.2, Definition 1124.3, Definition 1124.4, Definition 1124.5, Definition 1124.6, Definition 1124.7, and Definition 1124.8 bind the finite carrier, classifier, NameCert obligations, ledger non-escape, fit separation, and exact audit rows for signatures, failures, and strength.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedModelAudit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealityConstrainedModelAuditUp.RealityConstrainedModelAuditTasteGate\_single\_carrier\_`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove host-level model truth, unbounded physical validation, continuous experimental adequacy, hidden-source recovery, axiom-clean audit, or bridge checking for $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$.

Upgrade path: No further paper-side theory upgrade is pending; bridge checking remains a separate verification-axis matter.

Chapter 1125

A Concrete Naming Certificate for $\mathsf{OnticState}^{\uparrow}$

The $\mathsf{OnticState}^{\uparrow}$ packet realizes the stable-signature source of section 9414.15 as a finite NameCert surface. It sits beside the observer-local packet $\mathsf{ConsciousObserverState}^{\uparrow}$ of chapter 578, the model-state packet $\mathsf{StateSpaceModel}^{\uparrow}$ of chapter 329, the support-and-budget packet $\mathsf{ObserverBudgetSupport}^{\uparrow}$ of chapter 1121, the observer-history identity packet $\mathsf{ObserverHistIdentity}^{\uparrow}$ of chapter 479, and the quantum-state packet $\mathsf{QuantumState}^{\uparrow}$ of chapter 217. Its role is narrower than a total ontology and wider than a model row: it binds the reality-facing signature source, the admissible observer-access route, and the residue ledger in one accepted carrier before any consumer treats a model as ontically adequate.

Definition 1125.1 (OnticState carrier)**.** *An* $\mathsf{OnticState}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, A, K, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite reality-facing signature-source row, naming the stable signatures exposed by the accepted bundle family.*
2. *A is the observer-access route row, recording which* $\mathsf{ConsciousObserverState}^{\uparrow}$ *and* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *surfaces are allowed to inspect the signatures.*
3. *K is the classifier row comparing exposed signatures under the admissible observer bundle.*
4. *R is the residue ledger, recording reality-facing content not captured by the displayed* $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, *or* $\mathsf{QuantumState}^{\uparrow}$ *sibling rows.*
5. *H records componentwise* hsame *transport for the signature source, observer-access route, classifier, residue ledger, provenance, and local naming rows.*
6. *C records displayed* Cont *replay through S, A, K, R, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{OnticState}^{\uparrow}$, $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{QuantumState}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row.*

The classifier for O reads only the displayed signature source, observer-access route, signature classifier, residue ledger, transport, replay, provenance, and local name. It has no coordinate for

a total physical object, hidden global observer, unregistered model state, quotient observer identity, or host-level quantum state.

Definition 1125.2 (OnticState classifier)**.** *The* $\mathsf{OnticState}^{\uparrow}$ *classifier identifies two accepted packets only when their displayed signature-source rows agree through* K*, their observer-access routes agree through the same admitted observer and support surfaces, and their residue ledgers preserve the same hidden or unqueried reality-facing rows under componentwise* hsame *transport. Agreement of model coordinates alone is not enough to classify two* $\mathsf{OnticState}^{\uparrow}$ *packets as the same.*

Theorem 1125.3 (OnticState signature stability)**.** *Let* $O = (S, A, K, R, H, C, P, N)$ *be an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier. If the displayed signature source* S *is transported along the componentwise* hsame *rows in* H *and replayed through the route* C*, then the resulting packet remains classified by the same* $\mathsf{OnticState}^{\uparrow}$ *surface exactly when* A*,* K*, and* R *are transported with it. In particular, signature stability is not a property of* S *alone; it is a property of the source together with observer access, classifier, and residue ledger.*

Proof. Start with the carrier of Definition 1125.1. The signature source S is admitted only as part of the packet $O = (S, A, K, R, H, C, P, N)$.

Transport through H is componentwise. Thus a transported source row must be accompanied by the transported observer-access row A, classifier row K, residue ledger R, replay row C, provenance row P, and local name N.

The classifier of Definition 1125.2 then reads the transported packet through all four content rows S, A, K, R. A consumer that keeps S while dropping A, K, or R has left the accepted $\mathsf{OnticState}^{\uparrow}$ classifier surface. □

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateSignatureStability`

Theorem 1125.4 (OnticState residue-ledger exactness)**.** *Every accepted* $\mathsf{OnticState}^{\uparrow}$ *public read factors through the finite route*

$$S, A, K, R, H, C, P, N.$$

The route may expose the reality-facing signature source, admitted observer access, classifier comparison, and residue ledger. It cannot replace R *by a bare* $\mathsf{StateSpaceModel}^{\uparrow}$ *state, a* $\mathsf{ConsciousObserverState}^{\uparrow}$ *state, an* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *support row, an* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *identity row, or a* $\mathsf{QuantumState}^{\uparrow}$ *state.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateResidueLedgerExactness`

Proof. Use the displayed carrier $O = (S, A, K, R, H, C, P, N)$. Public reads reach the local name only after the signature source, access route, classifier, residue ledger, transport row, replay row, and provenance row have been displayed.

The residue ledger R is a content row of the carrier. It records what is not captured by the displayed model, observer, support, history-identity, or quantum-state sibling rows.

Replacing R by any one sibling would delete the ledger coordinate that distinguishes ontic source from model interface. Such a replacement has no row in O, so it is outside the accepted packet. □

Definition 1125.5 (OnticState access route)**.** *The admitted observer-access route row* A *of an* $\mathsf{OnticState}^{\uparrow}$ *packet is the finite route through the observer-local state surface of* $\mathsf{ConsciousObserverState}^{\uparrow}$ *and the support-and-budget surface of* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *before any source signature in* S *is public. It records which observer-facing read may inspect the signature source, which budget support*

makes the read admissible, and how that read remains tied to the classifier row K, residue ledger R, transport row H, replay row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{OnticState}^\uparrow}$ *row N.*

Theorem 1125.6 (OnticState model-residue noncollapse)**.** *No* $\mathsf{StateSpaceModel}^\uparrow$ *row or* $\mathsf{QuantumState}^\uparrow$ *row can replace the residue ledger R of an accepted* $\mathsf{OnticState}^\uparrow$ *carrier while preserving the* $\mathsf{OnticState}^\uparrow$ *classifier. Any such replacement loses the record of hidden or unqueried reality-facing content required by the classifier and by residue-ledger exactness.*

Proof. Use the carrier of Definition 1125.1. The row R is displayed as a residue ledger distinct from the model-state and quantum-state sibling rows.

The classifier of Definition 1125.2 reads equality through the joint rows S, A, K, R. Replacing R by a model row or a quantum row changes the packet read by the classifier, so the resulting tuple is outside the same accepted classifier surface.

Finally apply Definition 1125.4. It states that every public read factors through R and cannot replace it by a bare $\mathsf{StateSpaceModel}^\uparrow$ or $\mathsf{QuantumState}^\uparrow$ row. The claimed collapse therefore deletes a required ledger coordinate. □

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateModelResidueNoncollapse`

Theorem 1125.7 (OnticState observer-support factorization)**.** *Every observer-facing public read of an accepted* $\mathsf{OnticState}^\uparrow$ *packet factors through*

$$A, K, R, H, C, P, N$$

after the signature source S is displayed and before any model adequacy claim can be made. A read that reaches a $\mathsf{StateSpaceModel}^\uparrow$ *or* $\mathsf{QuantumState}^\uparrow$ *adequacy judgment without this access, classifier, residue, transport, replay, provenance, and naming route is outside the accepted* $\mathsf{OnticState}^\uparrow$ *public surface.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateObserverSupportFactorization`

Proof. Begin with the access route of Definition 1125.5. It forces every observer-facing read through $\mathsf{ConsciousObserverState}^\uparrow$ and $\mathsf{ObserverBudgetSupport}^\uparrow$ before the source signature is treated as public.

Apply Definition 1125.3. The source row is stable only when the observer-access row and classifier row are transported with it.

Use Definition 1125.4 and Definition 1125.6. The public route keeps the residue ledger before any model or quantum adequacy claim is considered, so the read factors through A, K, R, H, C, P, N rather than through a detached model surface. □

Theorem 1125.8 (OnticState NameCert obligation surface)**.** *For the carrier of Definition 1125.1, the local* $\mathsf{NameCert}_{\mathsf{OnticState}^\uparrow}$ *surface consists exactly of five rows:*

1. *carrier admission for $O = (S, A, K, R, H, C, P, N)$;*
2. *classifier exposure through the signature-source row S, observer-access row A, and classifier row K;*
3. *signature stability through componentwise transport and displayed replay of S, A, K;*
4. *residue-ledger exactness through R and its componentwise transport;*
5. *public obligation surface through H, C, P, N, with no hidden total-state, global-observer, quotient-identity, or host quantum-state coordinate.*

These five rows are the complete public obligation surface for using $\mathsf{OnticState}^{\uparrow}$ *as an ontic-state certificate.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateNameCertObligationSurface`

Proof. Carrier admission is the first projection from Definition 1125.1. Classifier exposure is the content of Definition 1125.2, read through S, A, K.

Signature stability follows from Definition 1125.3: the source row remains stable only with the access and classifier rows transported alongside it. Residue-ledger exactness follows from Definition 1125.4: R cannot be replaced by any single sibling surface.

The remaining rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no further coordinate, the listed five rows exhaust the public NameCert obligation surface. □

Definition 1125.9 (OnticState access-residue boundary)**.** *The accepted access-residue boundary for* $\mathsf{OnticState}^{\uparrow}$ *is the displayed tuple*

$$(S, A, K, R, H, C, P, N).$$

A public read is boundary-respecting only when it reaches the local name through the signature source S, observer-access row A, classifier row K, residue ledger R, componentwise transport row H, displayed replay row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row N. The boundary has no coordinate for observer access without residue, residue without an admitted observer route, or a model-facing substitute for the reality-facing signature source.*

Theorem 1125.10 (OnticState access-residue coupling)**.** *Let $O = (S, A, K, R, H, C, P, N)$ be an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier, and let a public read of O preserve the signature source S through the componentwise* hsame *rows in H and the displayed replay row C. Then the read preserves the observer-access row A and the residue-ledger row R as displayed coordinates of the same packet. In particular, preserving the source signature while dropping either admitted observer access or the residue ledger is not an accepted* $\mathsf{OnticState}^{\uparrow}$ *public read.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.onticState access residue boundary fields`

Proof. Begin with the boundary of Definition 1125.9. The public read reaches N only through the full tuple (S, A, K, R, H, C, P, N), so a source-preserving read is still evaluated inside the same displayed packet.

Apply Definition 1125.3 to the transported source. The theorem makes source stability depend on transporting the observer-access route and classifier alongside S.

Apply Definition 1125.4 to the public route. The residue ledger is an exposed coordinate of every accepted read and cannot be replaced by a sibling surface. Thus A and R remain coupled to any accepted preservation of the signature source. □

Theorem 1125.11 (OnticState sibling noncollapse)**.** *No sibling surface among* $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, *or* $\mathsf{QuantumState}^{\uparrow}$ *can replace an accepted* $\mathsf{OnticState}^{\uparrow}$ *packet. Any such replacement loses at least one displayed coordinate from the boundary (S, A, K, R, H, C, P, N): the reality-facing signature source, admitted observer-access route, classifier row, residue ledger, componentwise transport, displayed replay, provenance, or local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateSiblingNoncollapse`

Proof. Inspect the carrier in Definition 1125.1. Its coordinates bind the signature source, observer access, classifier, residue ledger, transport, replay, provenance, and local name in one finite packet.

The observer-state, model-state, budget-support, history-identity, and quantum-state siblings each expose one neighboring surface, not the complete boundary of Definition 1125.9. Replacing O by any one sibling therefore deletes a coordinate required by the accepted $\mathsf{OnticState}^{\uparrow}$ classifier or public read.

By Definition 1125.4, the residue row in particular cannot be supplied by a bare model, observer, support, history-identity, or quantum-state row. Hence the sibling replacement is outside the accepted packet. □

Theorem 1125.12 (OnticState transport scope)**.** *Let $O = (S, A, K, R, H, C, P, N)$ be an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier. The primitive transport scope of O is the componentwise* BHist *and* hsame *route over the signature source S, observer-access row A, classifier row K, residue ledger R, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row N. A transported public read of O cannot keep only S or only S, K while dropping access, residue, replay, provenance, or local naming.*

Proof. Begin with the carrier of Definition 1125.1. The row H is displayed as componentwise transport for the same finite packet $O = (S, A, K, R, H, C, P, N)$, not as a free equality on the signature source alone.

Apply Definition 1125.2 and Definition 1125.3. The classifier reads S, A, K, R together, and signature stability requires the access route, classifier, and residue ledger to travel with the source row.

Finally use Definition 1125.10 and Definition 1125.11. Access and residue remain packet coordinates under source preservation, while each sibling surface omits at least one coordinate of O. Hence the primitive transport scope is exactly the displayed BHist, hsame, Cont, Pkg, and NameCert route on the full carrier. □

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateCarrier transport scope`

Theorem 1125.13 (OnticState observer-access scope)**.** *For an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier $O = (S, A, K, R, H, C, P, N)$, the observer-access row A is scoped only through the admitted observer and support route, the classifier row K, the residue ledger R, and the support rows H, C, P, N. It is not a detachable observer permission, a global observer row, or a model-facing access token independent of the* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *surface of O.*

Proof. Start from the access route of Definition 1125.5. The row A records which $\mathsf{ConsciousObserverState}^{\uparrow}$ and $\mathsf{ObserverBudgetSupport}^{\uparrow}$ surfaces may inspect the source signatures before those signatures become public.

Use Definition 1125.9 and Definition 1125.10. They keep A inside the full boundary (S, A, K, R, H, C, P, N) and force accepted source-preserving reads to preserve the residue row R with the access row.

Apply Definition 1125.7 and Definition 1125.11. Observer-facing reads factor through A, K, R, H, C, P, N, and no sibling-only observer, model, support, history-identity, or quantum surface exposes all required coordinates. Thus observer access has the stated primitive scope. □

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateObserverAccessScope`

Theorem 1125.14 (OnticState residue ledger scope)**.** *For an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier $O = (S, A, K, R, H, C, P, N)$, the residue ledger R is scoped as the reality-facing residual row of the same* BHist *packet, transported through H, replayed through C, sourced by P, and named by N. The row*

R cannot be replaced by a $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ConsciousObserverState}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, *or* $\mathsf{QuantumState}^{\uparrow}$ *sibling row, and it cannot be read without the classifier and observer-access rows that determine the accepted packet.*

Proof. Inspect Definition 1125.1. The residue ledger R is a displayed coordinate of O, adjacent to the signature source, observer-access route, classifier, transport, replay, provenance, and local naming rows.

Apply Definition 1125.4. Every accepted public read factors through R, and none of the listed sibling rows can replace it as the ledger of hidden or unqueried reality-facing content.

Finally combine Definition 1125.6, Definition 1125.10, and Definition 1125.11. Model or quantum rows do not collapse the residue ledger, access and residue remain coupled under source preservation, and sibling surfaces omit the complete carrier. The residue scope is therefore the displayed BHist, hsame, Cont, Pkg, and NameCert ledger row of O. □

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateResidueLedgerScope`

Theorem 1125.15 (OnticState obligation-closure package)**.** *The* $\mathsf{OnticState}^{\uparrow}$ *obligation surface is the package consisting of carrier admission, classifier exposure, signature stability, access-residue coupling, sibling noncollapse, and the TasteGate-facing nontrivial and field-faithful obligations for the same tuple* (S, A, K, R, H, C, P, N)*. This package is sufficient for obligation closure of the paper-side* $\mathsf{OnticState}^{\uparrow}$ *NameCert surface: every accepted public use must preserve the displayed carrier, read classification through* S, A, K, R*, keep the access and residue rows coupled under source preservation, avoid sibling replacement, and expose a nontrivial field-faithful packet to downstream TasteGate consumers.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateObligationClosurePackage`

Proof. Carrier admission and classifier exposure are Definition 1125.1 and Definition 1125.2. Signature stability is Definition 1125.3.

The access-residue boundary is fixed by Definition 1125.9, and source-preserving public reads preserve both access and residue by Definition 1125.10. Sibling noncollapse is Definition 1125.11.

The Lean follow-up file named below supplies the TasteGate-facing nontrivial and field-faithful rows for the same finite packet. Together these clauses cover the carrier, classifier, source transport, access-residue boundary, noncollapse, and downstream TasteGate obligations named by the $\mathsf{OnticState}^{\uparrow}$ NameCert surface. □

Theorem 1125.16 (OnticState componentwise transport scope)**.** *Let* $O = (S, A, K, R, H, C, P, N)$ *be an accepted* $\mathsf{OnticState}^{\uparrow}$ *carrier. Any accepted public read of* O *that preserves the source signature* S *transports the full packet componentwise: the observer-access route* A*, classifier row* K*, residue ledger* R*, transport history* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row* N *remain the coordinates of the same packet. The transport is read through the* hsame *rows in* H*, the displayed* Cont *replay, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *witness; it is not a public read of* S *alone.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateComponentwiseTransportScope`

Proof. Begin with the accepted carrier of Definition 1125.1. The tuple stores S, A, K, R, H, C, P, N as one finite public packet, so a public read preserving S is still evaluated against that tuple.

Apply Definition 1125.3 to the source-preserving read. Its stability clause requires the signature source, observer-access route, classifier row, componentwise hsame transport, displayed replay, provenance, and local naming row to remain aligned.

Apply Definition 1125.10 for the exposed access and residue coordinates. The residue ledger and observer-access route are preserved as packet coordinates, not as independent sibling rows. Hence every accepted source-preserving read transports the eight displayed fields componentwise through H, C, P, N. □

Theorem 1125.17 (OnticState TasteGate field scope)**.** *The TasteGate-facing nontrivial and field-faithful rows for* $\mathsf{OnticState}^{\uparrow}$ *are scoped to the same eight-field packet* (S, A, K, R, H, C, P, N)*. They cannot be discharged by a sibling-only surface among* $\mathsf{ConsciousObserverState}^{\uparrow}$*,* $\mathsf{StateSpaceModel}^{\uparrow}$*,* $\mathsf{ObserverBudgetSupport}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*, or* $\mathsf{QuantumState}^{\uparrow}$*, because each such surface omits at least one of the signature source, observer access, classifier, residue ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, or local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateTasteGateFieldScope`

Proof. The TasteGate rows named for this chapter are attached to $\mathsf{OnticState}^{\uparrow}$, whose carrier is the packet of Definition 1125.1. The nontrivial witness separates two accepted packets inside that carrier, and field faithfulness reads the same packet coordinates.

By Definition 1125.11, no listed sibling exposes all eight coordinates of the ontic-state carrier. A sibling-only surface therefore lacks the public boundary required by Definition 1125.9.

The TasteGate rows must consequently be interpreted over the full $\mathsf{OnticState}^{\uparrow}$ packet, with the source, access, classifier, residue, transport, replay, provenance, and local name rows kept together. □

Theorem 1125.18 (OnticState scoped-closure package)**.** *The* $\mathsf{OnticState}^{\uparrow}$ *paper-side boundary is scoped closed by the carrier, classifier, residue exactness, access-residue coupling, sibling noncollapse, componentwise transport scope, and TasteGate field scope. Thus every accepted public use of* $\mathsf{OnticState}^{\uparrow}$ *must remain inside the single packet* (S, A, K, R, H, C, P, N)*, preserve source transport through the displayed* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$ *rows, and reject any sibling-only substitute for the ontic-state packet.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateScopedClosurePackage`

Proof. Carrier admission and classifier exposure are fixed by Definition 1125.1 and Definition 1125.2. Residue exactness is supplied by Definition 1125.4.

Access and residue remain coupled under source preservation by Definition 1125.10. Sibling replacement is excluded by Definition 1125.11.

The transport boundary is Definition 1125.16, which keeps all eight coordinates inside the same packet. The TasteGate obligations are scoped to that packet by Definition 1125.17.

Combining these clauses gives the public boundary claimed here: accepted uses preserve the complete ontic-state packet and cannot discharge the chapter through a model, observer, support, history-identity, or quantum-state sibling alone. □

Theorem 1125.19 (OnticState mature consumer completeness)**.** *Every downstream consumer of an accepted* $\mathsf{OnticState}^{\uparrow}$ *packet* $O = (S, A, K, R, H, C, P, N)$ *is complete exactly over the same displayed source, access, classifier, residue, transport, replay, provenance, and local naming rows. A consumer may use the access-residue coupling, sibling noncollapse, componentwise transport scope, and TasteGate field scope, but it cannot add a total physical ontology row, detach observer access from residue, collapse the packet to a sibling surface, or discharge the TasteGate obligations on any carrier other than* O*.*

Lean4 ✓ `BEDC.Derived.OnticStateUp.OnticStateUp StdBridge`

Proof. Start with the carrier and scoped-closure package. The carrier of Definition 1125.1 gives the eight rows S, A, K, R, H, C, P, N, and Definition 1125.18 binds accepted public use to that packet.

Apply access-residue coupling from Definition 1125.10. Any source-preserving consumer keeps the observer-access row A and residue ledger R as coordinates of the same public read.

Next use sibling noncollapse and componentwise transport scope. By Definition 1125.11, no neighboring observer, model, support, history-identity, or quantum surface replaces O; by Definition 1125.16, every accepted source-preserving read transports the full packet through H, C, P, N.

Finally apply the TasteGate field scope of Definition 1125.17. Its nontrivial and field-faithful rows are scoped to the same carrier, so a downstream consumer has no remaining paper-side row to add. Consumer completeness is therefore exactly the displayed packet O and its accepted public route. □

The Lean follow-up path is `lean4/BEDC/Derived/OnticStateUp/TasteGate.lean`. That file must provide explicit Nontrivial $\mathsf{OnticState}^{\uparrow}$ and FieldFaithful $\mathsf{OnticState}^{\uparrow}$ instances preserving the signature source, observer-access route, classifier, residue ledger, componentwise transport, displayed replay, provenance, and local naming rows.

Falsifiable prediction: An accepted $\mathsf{OnticState}^{\uparrow}$ packet cannot be reduced to $\mathsf{StateSpaceModel}^{\uparrow}$ without losing either the residue-ledger row or the observer-access route.

Independence witness: consciousobserverstate, statespacemodel, observerbudgetsupport, observerhistidentity, quantumstate: none is bijective because $\mathsf{OnticState}^{\uparrow}$ carries the reality-facing signature source and residue ledger together, while those siblings carry observer state, model state, budget support, history identity, or quantum-state rows separately.

Closure status for $\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OnticState}^{\uparrow}$ packet is generated from a finite reality-facing signature source, observer-access route, classifier row, residue ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.OnticStateUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1125.1, Definition 1125.2, Definition 1125.3, Definition 1125.4, Definition 1125.8, Definition 1125.9, Definition 1125.10, Definition 1125.11, Definition 1125.15, Definition 1125.16, Definition 1125.17, Definition 1125.18, and Definition 1125.19 bind the finite ontic-state carrier, classifier, stability row, residue ledger, access-residue boundary, sibling noncollapse, public obligation package, componentwise transport scope, TasteGate field scope, scoped public boundary, and downstream consumer completeness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OnticState}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.OnticStateUp.OnticStateTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total physical ontology, global observer access, model-state sufficiency, quotient observer identity, or host quantum-state equality.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 1126

A Concrete Naming Certificate for $\mathsf{ObservationReflectionPacket}^{\uparrow}$

The $\mathsf{ObservationReflectionPacket}^{\uparrow}$ packet realizes the observation–reflection equivalence thesis of chapter 9414 as a finite BEDC NameCert surface. It sits beside the finite observation route of chapter 1007, the closed observation packet of chapter 616, the observation conservation ledger of chapter 607, and the runtime checkpoint packet of chapter 997. Those siblings provide finite observation windows, closed observation ledgers, monotone record preservation, and runtime checkpoint boundaries; this packet binds the shared observation–reflection roles into one local carrier without identifying physical reality with MetaCIC.

Definition 1126.1 (ObservationReflectionPacket carrier)**.** *An* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, H, \Sigma, C, P, L, E, T, R, N)$$

with the following displayed rows:

1. *S is the source trace row, recording the reality-facing or MetaCIC-facing source access that starts the inquiry;*
2. *H is the finite* BHist *carrier of observations, generated traces, and replayable inquiry events;*
3. *Σ is the signature extraction row, listing the finite signature read from S through H;*
4. *C is the classifier row comparing the extracted signature with the displayed model, package, or checkpoint claim;*
5. *P is the package row naming the finite* Pkg *provenance for the source, trace, signature, classifier, ledger, and route rows;*
6. *L is the ledger row recording accepted matches, open gaps, rejected identity claims, and conservation constraints;*
7. *E is the open-continuation row, recording the prediction horizon or open meta-continuation that remains after the finite certificate is read;*
8. *T is the transport row carrying componentwise* hsame *comparisons between the observation side and the reflection side;*
9. *R is the route row of displayed* Cont *consumers through $S, H, \Sigma, C, P, L, E, T$;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationReflectionPacket}^{\uparrow}}$ *row naming exactly this packet.*

The carrier has no coordinate for an ontological identity between physical reality and MetaCIC, no final truth oracle, and no host-level physical law beyond the finite inquiry rows shown above.

Theorem 1126.2 (ObservationReflectionPacket NameCert obligations)**.** *For the carrier of Definition 1126.1, the local* $\mathsf{NameCert}_{\mathsf{ObservationReflectionPacket}\uparrow}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed packet* $Q = (S, H, \Sigma, C, P, L, E, T, R, N)$*;*
2. *source-trace exactness, requiring every inquiry read to start from* S *and pass through the finite history row* H*;*
3. *signature-classifier fit, requiring every accepted comparison to expose* Σ *and* C *before any package read;*
4. *package and ledger exactness, requiring* P *and* L *to name the finite provenance, accepted matches, open gaps, and rejected identity claims;*
5. *open-continuation preservation, requiring* E *to remain an open continuation rather than a final oracle;*
6. *transport, route, and local naming closure through* T, R, N*, requiring all observation-side and reflection-side reads to remain inside the displayed packet.*

Thus the observation–reflection thesis is certified as a finite role correspondence and not as an identity of source domains.

Lean4 ✓ `BEDC.Derived.ObservationReflectionPacketUp.ObservationReflectionPacketTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1126.1. The ten rows $S, H, \Sigma, C, P, L, E, T, R, N$ are the complete finite packet visible to a consumer.

Source-trace exactness uses the first two rows. The source row S records the inquiry access point, while H records only the finite trace available from that source. This matches the finite-trace side of chapter 9414 and the finite observation interface of chapter 1007.

Signature-classifier fit is then read through Σ and C. The packet does not classify a hidden source directly; it classifies only the displayed signature extracted from the finite trace. Package and ledger exactness attach P and L to that same read, so accepted matches, open gaps, and rejected identity claims remain part of the local NameCert packet.

The open-continuation row E records the non-final horizon. It agrees with the reflection checkpoint boundary of chapter 997: checkpoint and inquiry rows can be inspected, but neither side exports total truth or host validation from the finite packet.

Finally T, R, N transport, route, and name the same rows. Since those rows add no source-domain equality, final oracle, or host physical-law coordinate, the six obligations exhaust the local certificate surface. □

Theorem 1126.3 (ObservationReflectionPacket non-escape boundary)**.** *Every accepted* ObservationReflectionPacke *packet exports only the displayed rows* $S, H, \Sigma, C, P, L, E, T, R, N$*. A consumer may read the source trace, finite history, extracted signature, classifier fit, package provenance, ledgered gaps, open continuation, transport, route, and local naming row. It cannot recover an ontological identity between physical reality and MetaCIC, a final truth oracle, a host-level physical law, an unbounded observation stream, or a runtime validator outside those rows.*

Proof. Inspect the carrier of Definition 1126.1. Its only source-facing rows are S and H, its only pattern-facing rows are Σ and C, and its only boundary-facing rows are P, L, E, T, R, N.

Apply the NameCert obligations of Definition 1126.2. Every legal read starts with source-trace exactness, passes through signature-classifier fit, and then records package provenance, ledger status, and open continuation before transport and local naming.

The sibling closed-observation packets make the same inventory boundary explicit. chapter 616 exposes only finite observation records plus a gap socket, while chapter 607 preserves displayed records without adding an oracle for unseen witnesses.

Thus any demand for source-domain identity, final truth, host physical law, unbounded observation, or runtime validation has no coordinate in the accepted packet. It is recorded as a non-exported demand at the ledger or continuation boundary rather than as a readable row. □

Lean4 ✓ `BEDC.Derived.ObservationReflectionPacketUp.ObservationReflectionPacket nonescape fields`

Theorem 1126.4 (ObservationReflectionPacket public export surface)**.** *The public export surface of an accepted* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *packet is exactly*

$$Q = (S, H, \Sigma, C, P, L, E, T, R, N).$$

It exports the source trace, finite history, signature extraction, classifier, package, ledger, open continuation, transport, route, and local NameCert *rows. It does not export ontological identity, a final oracle, host physical law, or an unbounded observation stream.*

Lean4 ✓ `BEDC.Derived.ObservationReflectionPacketUp.ObservationReflectionPacketUp StdBridge`

Proof. Project the carrier of Definition 1126.1. Its ten displayed rows are precisely $S, H, \Sigma, C, P, L, E, T, R, N$.

Apply the local NameCert obligations from Definition 1126.2. Source trace, finite history, signature extraction, classifier fit, package provenance, ledger status, open continuation, transport, route, and local naming are the rows made available to a public consumer.

Finally use the non-escape boundary of Definition 1126.3. Since the accepted packet has no coordinate for source-domain identity, final truth, host physical law, or unbounded observation, the public export surface is exactly the displayed finite packet. □

Falsifiable prediction: No accepted $\mathsf{ObservationReflectionPacket}^{\uparrow}$ packet exports an ontological identity between physical reality and MetaCIC, a final truth oracle, or a host-level physical law beyond its displayed finite inquiry rows.

Independence witness: finiteobservationinterface, closedobservationsystem, observationconservationledger, packagereflection, reflectionruntimecheckpoint: $\mathsf{ObservationReflectionPacket}^{\uparrow}$ is not bijective to any listed sibling because it simultaneously carries source trace, finite history, signature extraction, classifier fit, package provenance, ledgered gap status, open continuation, transport, route, and local naming rows. The finite observation sibling lacks the reflection-side classifier and open-continuation ledger; the closed observation and conservation siblings lack the signature-package reflection row; the package-reflection surface lacks observation-source and conservation rows; the runtime checkpoint sibling lacks the physical-inquiry source trace and empirical signature inventory.

The Lean follow-up in `lean4/BEDC/Derived/ObservationReflectionPacketUp/TasteGate.lean` should provide explicit Nontrivial $\mathsf{ObservationReflectionPacket}^{\uparrow}$ and FieldFaithful $\mathsf{ObservationReflectionPacket}^{\uparrow}$ instances preserving $S, H, \Sigma, C, P, L, E, T, R, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationReflectionPacket}^{\uparrow}}$.

Constructive story: An ObservationReflectionPacket$^{\uparrow}$ packet is generated from a source trace, finite history carrier, signature extraction row, classifier row, package provenance, ledger row, open continuation, componentwise transport, displayed route, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObservationReflectionPacket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1126.1, Definition 1126.2, Definition 1126.3, Definition 1126.4, and the TasteGate single-carrier alignment supply the paper-side bridge from the finite observation-reflection packet to the abstract interface by binding the carrier, non-escape boundary, public export surface, and alignment target together.

Formal verification: FormalStatus(NameCert$_{\text{ObservationReflectionPacket}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.ObservationReflectionPacketUp.ObservationReflectionPacketTasteGate\_single\_carrier\_al`

Bridge status: bridgeChecked

Origin: ai

Lean4 ✓ `BEDC.Derived.ObservationReflectionPacketUp.taste gate`

Not claimed: This chapter does not assert ontological identity between physical reality and MetaCIC, final truth, host-level physical laws, unbounded observation streams, or bridge checking.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 1127

A Concrete Naming Certificate for $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$

The $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$packet realizes the indexed approximation-tower discipline of chapter 9414 as a composite BEDC NameCert carrier. It sits after the observation reflection packet of chapter 1126, the ontic-state packet of chapter 1125, the approximation carrier of chapter 225, the finite Cauchy approximation carrier of chapter 601, the Real approximation envelope of chapter 637, and the observer-filter budget selector of chapter 1042. Those siblings provide observation readback, ontic-state naming, approximation rows, finite Cauchy windows, Real-facing envelope data, and selected observer budgets; this carrier binds them into one indexed reality-constrained tower without importing host equality, quotient reals, hidden limit selectors, or ambient completeness. As an ai packet, it bundles observation reflection, ontic-state naming, approximation, finite Cauchy approximation, Real-envelope, and observer-budget surfaces into one indexed tower ledger.

Definition 1127.1 (Reality-constrained approximation-tower carrier)**.** *A* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *c*
is a finite BHist *packet*

$$T = (O, D, M, S, C, F, L, H, R, P, N)$$

with the following displayed rows:

1. *O is the ontic source row naming the reality-facing source whose signatures are being approximated;*
2. *D is the observation-data row, containing the finite admissible observations selected for the tower and the displayed observation-reflection readback;*
3. *M is the model row, naming the visible mathematical model read from the observed source;*
4. *S is the physical-signature row extracted from O through D and read against M;*
5. *C is the reality-constrained classifier row comparing model-facing signatures with the reality-facing signature row S;*
6. *F is the GAP-admissible compression row, recording the indexed sequence of approximation, finite Cauchy, Real-envelope, observer-budget, and residue stages accepted by the displayed classifier;*
7. *L is the ledger row recording every hidden source, compressed stage, rejected signature omission, and open residue of the tower;*

8. *H records componentwise* hsame *transport among source, observation-data, model, signature, classifier, compression, ledger, provenance, and local-name rows;*
9. *R records the displayed* Cont *route from source observation to model readback, signature comparison, compression use, and ledger inspection;*
10. *P is the* Pkg *provenance over chapter 1126, chapter 1125, chapter 225, chapter 601, chapter 637, chapter 1042,* BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedApproximationTower}^{\uparrow}}$ *row.*

The classifier admits T only when the physical signature, GAP-admissible compression, and ledger rows are all displayed on the same indexed tower. The carrier has no coordinate for mathematical approximation divorced from physical signature, compression without a listed GAP row, erased compression residue, quotient-real equality, selected host limits, host-level physical law, or ambient completeness.

Theorem 1127.2 (Reality-constrained approximation-tower NameCert obligations)**.** *For every accepted* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *carrier*

$$T = (O, D, M, S, C, F, L, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedApproximationTower}^{\uparrow}}$ *surface consists exactly of carrier admission for T, ontic source exposure through O, observation-data readback through D, model exposure through M, physical-signature display through S, reality-constrained classifier exactness through C, GAP-admissible compression through F, ledger coverage through L, componentwise* hsame *transport through H, displayed* Cont *route replay through R, dependency provenance through P, and local naming through N.*

Proof. Project the carrier definition. The positive route exposes the source row O, observation-data row D, model row M, physical-signature row S, classifier C, GAP-admissible compression row F, and ledger row L before the structural rows are read.

The structural rows are attached to that same route. Transport through H is componentwise over the displayed rows, replay through R follows the same tower ledger, provenance through P names the listed siblings, and N names no additional coordinate.

Thus the local NameCert surface is exhausted by the displayed rows. Since no additional coordinate exists for detached mathematical approximation, unledgered physical signature data, a hidden classifier, or an ambient completion principle, the obligations are exactly the ones listed. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedApproximationTowerUp.RealityConstrainedApproximationTowerTasteGa`
`single carrier alignment`

Theorem 1127.3 (Reality-constrained approximation-tower non-escape)**.** *Every mathematical approximation consumed through an accepted* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *packet factors through the displayed physical-signature row S, classifier row C, GAP-admissible compression row F, and tower ledger L. It cannot replace physical-fit signature data by using only the model row M, host equality, quotient-real equality, selected limits, or ambient completeness.*

Proof. Start with the carrier $T = (O, D, M, S, C, F, L, H, R, P, N)$. Its only approximation-facing coordinates after model readback are the physical-signature row S, classifier row C, compression row F, and ledger row L.

Apply the obligation surface of Definition 1127.2. Any mathematical classifier whose stability depends on hidden source history, omitted physical signature, or compressed residue must be recorded through S, C, F, L rather than silently transported across M.

It remains to exhaust the excluded routes. The model row M records the visible mathematical representation, and the structural rows H, R, P, N only transport, replay, package, and name displayed data. None of these rows contains host equality, quotient-real equality, selected limits, ambient completeness, or a second physical-fit channel. Thus mathematical approximation cannot replace the displayed physical-fit signature data. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedApproximationTowerUp.RealityConstrainedApproximationTowerNonesca`

Theorem 1127.4 (Reality-constrained approximation-tower sibling independence)**.** *The* RealityConstrainedApproxi *is not bijective to the observation-reflection-packet, ontic-state, finite-observation-interface, closed-observation-system, reflection-runtime-checkpoint, approximation, finite-Cauchy-approximation, Real-approximation-envelope, observer-filter-budget-selector, or residue surfaces named by its provenance. Each sibling lacks at least one of the joint ontic source, observation-data row, model row, physical-signature row, classifier row, GAP-admissible compression row, ledger row, transport, continuation, provenance, and local NameCert coordinates required by Definition 1127.1.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedApproximationTowerUp.RealityConstrainedApproximationTower sibling independence boundary`

Proof. Inspect the observation-facing siblings. The observation-reflection packet supplies a readback route but not the full GAP-admissible compression ledger. The ontic-state packet names source state but not the classifier and tower ledger. The finite-observation-interface, closed-observation-system, and reflection-runtime-checkpoint surfaces each isolate an interface, closure, or runtime checkpoint without carrying the full physical-fit classifier across the tower.

Inspect the approximation-facing siblings. The approximation carrier supplies approximation rows but not the full physical-signature and ledger inventory. The finite Cauchy approximation carrier supplies finite precision windows but not the ontic source and physical-signature classifier. The Real approximation envelope supplies Real-facing boundary rows but not the selected observation bundle with omitted-row residue. The observer-filter budget selector supplies selected and omitted observer rows but not the full indexed tower from ontic source to model readback.

The residue surface is also separated. It records what remains outside a given admission, while the tower carrier binds residue to the same packet that contains source, observation data, model, physical signature, classifier, GAP-admissible compression, ledger, transport, continuation, provenance, and local naming rows.

Therefore no listed sibling carries all coordinates of T and no coordinate of T can be deleted without changing the accepted NameCert surface. A bijection preserving the local obligations would have to preserve all displayed rows, which none of the siblings can do. □

Falsifiable prediction: An accepted $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ packet cannot identify two representations by a mathematical classifier unless their displayed physical signature rows agree through the same tower ledger.

Independence witness: observationreflectionpacket, onticstate, finiteobservationinterface, closedobservationsystem, reflectionruntimecheckpoint, approximation, finite-cauchy-approximation, real-approximation-envelope, observer-filter-budget-selector, residue: no listed sibling is bijective to $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ because only this packet carries ontic source, observation data, model row, physical signature, classifier, GAP-admissible compression, tower ledger, hsame

transport, Cont route, Pkg provenance, and local NameCert rows together. The tower ledger and physical-fit classifier separate these coordinates from every sibling surface.

The packet composes observationreflectionpacket, onticstate, finiteobservationinterface, closedobservationsystem, reflectionruntimecheckpoint, approximation, finite-cauchy-approximation, real-approximation-envelope, observer-filter-budget-selector, and residue into the displayed approximation-tower surface.

Lean development for this route must provide Nontrivial RealityConstrainedApproximationTower$^{\uparrow}$ and FieldFaithful RealityConstrainedApproximationTower$^{\uparrow}$ instances in `lean4/BEDC/Derived/RealityConstrainedA`

Theorem 1127.5 (Reality-constrained approximation-tower bridge surface)**.** *Every public* RealityConstrainedAppro *packet bridges the observation-reflection, ontic-state, approximation, finite-Cauchy approximation, Real-approximation envelope, and observer-filter budget rows into one BEDC carrier interface. The bridge preserves the physical-signature classifier and ledger nonescape while keeping the TasteGate handoff attached to the same finite tower packet.*

Proof. Start from the carrier of Definition 1127.1. Its displayed rows contain the ontic source, observation-data readback, model row, physical-signature row, classifier, GAP-admissible compression, ledger, transport, replay, provenance, and local naming data.

Apply Definition 1127.2 and Definition 1127.3. The obligation surface binds all six dependency families to the same packet, and nonescape keeps mathematical approximation behind the physical-signature classifier and tower ledger.

Use sibling independence, Definition 1127.4, to prevent any listed sibling from replacing the composite carrier. The existing TasteGate marker remains the Lean-facing witness that the displayed carrier alignment is the accepted public route. □

Closure status for NameCert$_{\text{RealityConstrainedApproximationTower}^{\uparrow}}$.

Constructive story: A RealityConstrainedApproximationTower$^{\uparrow}$ packet is generated from ontic source, observation-data, model, physical-signature, classifier, GAP-admissible compression, ledger, transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedApproximationTower}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1127.1, Definition 1127.2, Definition 1127.3, Definition 1127.4, Definition 1127.5, and

Lean4 ✓ `BEDC.Derived.RealityConstrainedApproximationTowerUp.taste gate`

bind the public carrier, NameCert surface, non-escape boundary, sibling-independence witness, paper bridge surface, and TasteGate handoff for the finite tower.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedApproximationTower}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RealityConstrainedApproximationTowerUp.RealityConstrainedApproximationTowerTasteGate\_s`

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim machine bridge checking for the paper bridge surface.

Upgrade path: Bridge checking remains the next verification step for binding the same carrier, NameCert surface, non-escape boundary, sibling-independence witness, paper bridge surface, and TasteGate handoff.

Chapter 1128

A Concrete Naming Certificate for $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$

The $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ packet realizes the reality-constrained approximation tower of chapter 9414 as a finite NameCert carrier. It sits beside the ontic-state surface of chapter 1125, the finite stream window surface of chapter 126, the regular rational readback of chapter 301, the Real seal of chapter 38, and the dyadic observation ledger of chapter 291. When tower observations are exposed, the packet also consumes the methodology-ledger route of chapter 1145 because that sibling provides the finite audit and signature-residue rows that the tower packet packages. Those sibling chapters supply source, schedule, rational-window, completed-real, and dyadic-tolerance rows; this packet binds them to classifier adequacy, descent, ledger, residue, transport, and non-escape rows before any consumer treats approximation or completion as reality-facing fit.

Definition 1128.1 (Reality-constrained tower packet carrier)**.** *A* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, S, R, E, D, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OnticState}^{\uparrow}$ *source row naming the reality-facing source whose approximation is being read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite observation windows available to the packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular rational readback.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and residue ledger row for the same finite windows.*
6. *L is the classifier-ledger row recording which approximation, completion, and physical-signature comparisons are asserted and which adequacy gaps remain visible.*
7. *Q is the descent row. A consumer may descend along a tower operation only when the corresponding permission is displayed in Q.*
8. *H records componentwise* hsame *transport for the source, schedule, readback, Real seal, dyadic ledger, classifier ledger, descent row, replay, provenance, and local name.*

9. *C records the displayed* Cont *replay through $O, S, R, E, D, L, Q, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$, $\mathsf{OnticState}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for identifying mathematical classifier adequacy with physical fit, no route from a completed Real row to reality-facing equivalence without O, L, D, and no descent across the tower outside the displayed row Q.

Lean4 (definition) `BEDC.Derived.RealityConstrainedTowerPacketUp.RealityConstrainedTowerPacketCarrier`

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerPacketUp.RealityConstrainedTowerPacketCarrier route closure`

Theorem 1128.2 (Reality-constrained tower packet NameCert boundary)**.** *For the carrier of Definition 1128.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerPacket}^{\uparrow}}$ *boundary consists exactly of seven obligation surfaces:*

1. *carrier admission for $T = (O, S, R, E, D, L, Q, H, C, P, N)$;*
2. *source exposure through the* $\mathsf{OnticState}^{\uparrow}$ *row O;*
3. *finite-window classifier exposure through S, R, E, D in the displayed StreamName, RegSeqRat, Real, and DyadicRatCore order;*
4. *reality-facing classifier accountability through the ledger row L;*
5. *descent permission through Q;*
6. *componentwise transport and replay through H, C;*
7. *provenance and local naming through P, N.*

Any consumer that reads endpoint equivalence, completion adequacy, or tower descent from the packet must display these surfaces before exporting its conclusion.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketNameCertBou`

Proof. Project the carrier. The only accepted packet is the finite tuple $O, S, R, E, D, L, Q, H, C, P, N$, so endpoint equivalence and tower descent have no source outside the displayed rows.

Read the sibling rows in order. The ontic source comes from chapter 1125, finite access comes through chapter 126, regular readback through chapter 301, Real sealing through chapter 38, and dyadic tolerance through chapter 291.

The ledger L records the reality-facing classifier and adequacy residue, while Q records the permitted descent rows. The transport, replay, provenance, and local name rows H, C, P, N preserve exactly that packet. Therefore the local certificate boundary is exhausted by the seven listed surfaces. □

Theorem 1128.3 (Reality-constrained tower packet non-escape)**.** *Every accepted* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *consumer read factors through*

$$O, S, R, E, D, L, Q, H, C, P, N.$$

In particular, a consumer cannot export endpoint equivalence, completion adequacy, physical fit, or tower descent from the mathematical approximation rows alone. The reality-facing classifier ledger L, dyadic residue row D, ontic source row O, and descent row Q remain mandatory boundary data.

Lean4 ✓ BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketUp nonescape transport

Proof. Start with Definition 1128.2. It requires source exposure, finite-window classifier exposure, reality-facing classifier accountability, descent permission, transport, replay, provenance, and local naming before the packet can be consumed.

The mathematical rows S, R, E provide windowed readback and Real sealing, but they do not supply the ontic source O, dyadic residue ledger D, classifier ledger L, or descent row Q. Thus a route that keeps only mathematical approximation has lost rows required by the certificate.

Transport and replay do not add a hidden adequacy route. The rows H, C, P, N only preserve and name the displayed packet, so every accepted read remains inside $O, S, R, E, D, L, Q, H, C, P, N$. □

Theorem 1128.4 (Reality-constrained tower packet TasteGate obligation)**.** *The* RealityConstrainedTowerPacket$^{\uparrow}$ *TasteGate surface is obligated to keep one carrier alignment across the source, classifier, descent, ledger, residue, transport, provenance, and local-name rows of Definition 1128.1. A* Nontrivial *instance must distinguish at least one accepted source-ledger packet from a packet with missing reality-facing classifier, descent, or residue rows, and a* FieldFaithful *instance must preserve the row order*

$$O, S, R, E, D, L, Q, H, C, P, N$$

under every accepted field projection.

Proof. Use the NameCert boundary theorem to identify the carrier fields visible to the gate. The accepted gate has to read the source, schedule, regular readback, Real seal, dyadic residue, classifier ledger, descent permission, transport, replay, provenance, and local name as one packet.

Nontriviality is forced by the non-escape theorem: a packet that omits the reality-facing classifier, descent, or residue rows cannot export the same accepted consumer route. Field faithfulness is therefore the requirement that every gate projection preserve the displayed row order rather than replacing it by a detached mathematical classifier. □

Lean4 ✓ BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketTasteGate single carrier alignment

Theorem 1128.5 (Reality-constrained tower packet classifier-ledger exactness)**.** *For every accepted* RealityConstrainedTowerPacket$^{\uparrow}$ *carrier* $T = (O, S, R, E, D, L, Q, H, C, P, N)$*, the row* L *is the only classifier-adequacy row visible to endpoint-equivalence and completion-adequacy consumers. The schedule, readback, Real-seal, and descent rows* S, R, E, Q *may be read only as finite access and permission data, while* D *records dyadic residue; none of* S, R, E, D *exports physical fit without the reality-facing classifier ledger* L *together with the source row* O *and the local certificate rows* P, N*.*

Lean4 ✓ BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketClassifierL

Proof. Project the carrier through Definition 1128.1. The displayed classifier-adequacy coordinate is L, and the NameCert boundary of Definition 1128.2 lists reality-facing classifier accountability as its own obligation surface.

The finite-window rows S, R, E and the residue row D are therefore not alternative classifier witnesses. They supply schedule, readback, completion, and tolerance data inside the packet, but their consumer routes still pass through the seven-surface boundary.

Apply Definition 1128.3. Any endpoint-equivalence or completion-adequacy export that forgets O, L, D, Q, H, C, P, N has left the accepted packet. Hence L is exact as the ledger row for classifier adequacy, and S, R, E, D cannot by themselves export physical fit. □

Theorem 1128.6 (Reality-constrained tower packet descent permission stability)**.** *Let $T = (O, S, R, E, D, L, Q, H,$ and $T' = (O', S', R', E', D', L', Q', H', C', P', N')$ be accepted* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *carriers related by componentwise* hsame *transport preserving the displayed rows $O, S, R, E, D, L, Q, H, C, P, N$. If a descent is permitted for T' after transport, then its permission is already displayed in the transported row Q'. The transport and replay rows H', C' cannot add a descent outside Q'.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketDescentPerm`

Proof. Project the target carrier T'. By the carrier definition, descent permission is stored in the single displayed row Q', while H' records componentwise hsame transport and C' records replay through the same named rows.

Use the NameCert boundary to read descent permission as a separate obligation surface. Since the boundary exposes Q' before transport and replay are consumed, replay can only preserve the displayed permission surface.

Finally apply the non-escape theorem to the transported packet. A descent route outside Q' would export tower descent without the mandatory descent row of the accepted carrier, contradicting the stated consumer boundary. Thus componentwise transport preserves permitted descent and cannot create a hidden descent permission. □

Theorem 1128.7 (Reality-constrained tower packet Real-phase consumer boundary)**.** *Every consumer route that passes through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *before treating an endpoint as reality-facing must also display the rows*

$$O, L, Q, H, C, P, N.$$

Thus finite windows, regular rational readback, Real sealing, and dyadic tolerance are not sufficient for a reality-facing endpoint unless the source, classifier ledger, descent permission, transport, replay, provenance, and local NameCert rows remain present.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedTowerPacketRealPhaseCo`

Proof. Read the sibling route in the order named in the chapter introduction. $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, $\mathsf{Real}^{\uparrow}$ supplies the completed seal, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies dyadic tolerance and residue.

The classifier-ledger exactness theorem then identifies L as the visible classifier-adequacy row for endpoint and completion consumers. The descent-permission stability theorem identifies Q as the only displayed descent row preserved under componentwise transport and replay.

Apply the non-escape theorem to the combined route. Since every accepted consumer read factors through $O, S, R, E, D, L, Q, H, C, P, N$, the Real-phase route may use S, R, E, D only while also retaining O, L, Q, H, C, P, N. Therefore no completed endpoint becomes reality-facing from the Real-phase rows alone. □

Falsifiable prediction: Any accepted RealityConstrainedTowerPacketUp consumer that exports endpoint equivalence without reality-facing classifier, descent, and residue-ledger rows refutes the packet.

Independence witness: onticstate, streamname, regseqrat, real, dyadicratcore: none is bijective because $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ binds source, classifier, descent, ledger, residue, transport, provenance, and local NameCert rows in one carrier, while each sibling supplies only one constituent surface.

The Lean `TasteGate.lean` development for $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ provides explicit $\mathsf{Nontrivial\ RealityConstrainedTowerPacket}^{\uparrow}$ and $\mathsf{FieldFaithful\ RealityConstrainedTowerPacket}^{\uparrow}$ instances, together with the target `BEDC.Derived.RealityConstrainedTowerPacketUp.TasteGate.RealityConstrainedT`

Theorem 1128.8 (Reality-constrained tower packet public export)**.** *The public* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *surface exposes exactly the carrier, NameCert boundary, non-escape, classifier-ledger exactness, descent-permission stability, and Real-phase consumer boundary rows of this chapter. A downstream public consumer may cite the packet as a reality-constrained approximation-tower object only while preserving the displayed source, schedule, readback, Real seal, dyadic residue ledger, classifier ledger, descent permission, transport, replay, provenance, and local NameCert rows.*

Proof. Start with the carrier and NameCert boundary. They present the finite tuple $O, S, R, E, D, L, Q, H, C, P, N$ and identify the seven obligation surfaces available to consumers.

Apply the non-escape, classifier-ledger exactness, descent-permission stability, and Real-phase consumer boundary theorems. Together they rule out public reads that forget the source, classifier ledger, residue ledger, descent permission, transport, replay, provenance, or local naming rows.

Thus the public export is not a new mathematical object. It is the named outward-facing use of the same displayed packet, with every consumer route still factored through the rows proved in the chapter. □

Theorem 1128.9 (Reality-constrained tower packet mature boundary)**.** *The public export, non-escape, classifier-ledger exactness, descent-permission stability, and Real-phase consumer boundary theorems exhaust the mature paper surface of* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$*. Every mature consumer route must preserve the displayed source, schedule, readback, Real seal, dyadic residue ledger, classifier ledger, descent permission, transport, replay, provenance, and local NameCert rows, and no further paper-side endpoint is exported by this chapter.*

Proof. Start with the public export theorem Definition 1128.8. It identifies the outward-facing surface as the carrier, NameCert boundary, non-escape, classifier-ledger exactness, descent-permission stability, and Real-phase consumer boundary rows.

Apply non-escape and classifier-ledger exactness, Definition 1128.3 and Definition 1128.5. These force every endpoint-equivalence or completion-adequacy read to retain the source, classifier ledger, dyadic residue, descent row, transport, replay, provenance, and local naming data.

Apply descent-permission stability and the Real-phase consumer boundary, Definition 1128.6 and Definition 1128.7. They keep tower descent inside Q and keep the Real-facing route dependent on O, L, Q, H, C, P, N.

The carrier and NameCert boundary contain no additional mature endpoint. Therefore the listed public, non-escape, classifier-ledger, descent-permission, and Real-phase rows exhaust the chapter's paper-side mature surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ packet is generated from an ontic source row, finite stream schedule, regular rational readback, Real seal, dyadic residue ledger, reality-facing classifier ledger, descent permission row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerPacket}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1128.1, Definition 1128.2, Definition 1128.3, Definition 1128.4, Definition 1128.5, Definition 1128.6, Definition 1128.7, Definition 1128.8, and Definition 1128.9 bind the mature paper surface across the public source, classifier, descent, ledger, residue, transport, provenance, and local naming rows for reality-constrained approximation-tower packets; this bridge is not Lean-checked.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedTowerPacket}\uparrow}$, unformalized)
Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerPacketUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total physical adequacy, host physical identity, quotient equality of Real-completion data with source reality, or unrestricted tower descent.

Upgrade path: Mature closure requires a checked standard bridge and audit-clean transport for the displayed packet rows.

Chapter 1129

A Concrete Naming Certificate for $\mathsf{PhysicalModelAudit}^{\uparrow}$

The $\mathsf{PhysicalModelAudit}^{\uparrow}$ packet realizes the audit protocol of Definition 9414.617 as one finite BEDC naming certificate. It is adjacent to the audit-channel surface of chapter 969, the trace-storage surface of chapter 972, the state-space model surface of chapter 329, the ontic-state surface of chapter 1125, and the observation-reflection routing surface of chapter 1126. Its role is to bind source traces, target phenomenon, observed and model signatures, classifier, tower, stability and descent rows, ledger, failure surface, strength, transports, routes, provenance, and local NameCert before a physical claim is read as accepted.

Definition 1129.1 (PhysicalModelAudit carrier)**.** *A* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N)$$

with the following displayed rows:

1. *Q is the finite source-trace row, recording the histories or experimental traces consumed by the audit.*
2. *R is the target-phenomenon row, naming the physical system, phenomenon, or class of systems under audit.*
3. *O is the observed-signature row extracted from Q through the declared observation bundle.*
4. *M is the model-signature row generated by the candidate model for the same bundle.*
5. *C is the physical-audit classifier comparing O and M at the declared fit level.*
6. *T is the approximation-tower row from source traces to model-facing signatures.*
7. *Y is the stability row specifying the licensed reuse domain for the audited comparison.*
8. *D is the descent row recording which predictive or explanatory operations survive the tower.*
9. *L is the audit ledger for provenance, hidden compression, representation, idealization, and translation material.*
10. *F is the failure-surface row naming finite defeating observations or signature mismatches.*
11. *S is the audit-strength row, distinguishing seed support, paper fit, reproduced fit, and bridged fit.*
12. *H records componentwise* hsame *transport for all content rows.*

13. *U records displayed* Cont *routes through source, phenomenon, signature, classifier, tower, stability, descent, ledger, failure, and strength rows.*
14. *P is the* Pkg *provenance over* $\mathsf{PhysicalModelAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalModelAudit}^{\uparrow}}$ *row.*

The carrier contains no host-level truth oracle, unbounded observation stream, unledgered compression, detached model equivalence, or physical acceptance route outside the displayed packet.

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditTasteGate single carrier alignment`

Definition 1129.2 (PhysicalModelAudit classifier)**.** *For accepted packets*

$$A = (Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N) \quad \text{and} \quad A' = (Q', R', O', M', C', T', Y', D', L', F', S', H', U', P', N')$$

the predicate $\mathsf{PhysicalModelAuditClassifier}(A, A')$ *holds when H and H' transport source traces, target phenomena, observed signatures, model signatures, approximation towers, stability rows, descent rows, ledger rows, failure surfaces, strength rows, routes, provenance, and local names componentwise, and when C and C' preserve the same declared comparison between O, M and O', M'.*

Classifier agreement is therefore stronger than model-syntax agreement and weaker than host-level physical truth. It is the finite assertion that the same audit comparison is read through the same displayed source-signature-classifier-tower-ledger packet.

Theorem 1129.3 (PhysicalModelAudit NameCert obligation surface)**.** *For the carrier of Definition 1129.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalModelAudit}^{\uparrow}}$ *surface is exhausted by the following five obligation rows:*

1. *source admission: Q and R are the only source-trace and target-phenomenon rows licensed for the audit;*
2. *signature admission: O and M are the observed and model-signature rows read by the packet;*
3. *classifier admission: C is the finite comparison row relating O and M at the declared fit level;*
4. *tower, stability, and descent admission: T, Y, D are the displayed approximation, reuse, and operation-survival rows;*
5. *ledger, failure, and strength admission: L, F, S record provenance material, defeating observations, and the evidential grade before the local name is accepted.*

No accepted physical-audit claim through this packet can read a field outside these rows.

Proof. Carrier admission is the projection of the displayed tuple in Definition 1129.1. The packet lists the source row, target row, signature rows, classifier row, tower row, stability row, descent row, ledger row, failure row, strength row, and the structural transport, route, provenance, and naming rows.

The classifier of Definition 1129.2 reads only those rows. Its comparison is not a separate model oracle; it is the relation between observed and model signatures as transported and replayed inside the same packet.

The ledger and failure-surface rows close the remaining escape points. Any compression, idealization, translation, or possible defeat used by the audit must be named by L or F, and the evidential level must be named by S.

Thus the local NameCert has no further coordinate to expose. A consumer that asks for another source, signature, classifier, tower, ledger, failure surface, or strength row has left the accepted PhysicalModelAudit$^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditPacket namecert obligation surface`

Theorem 1129.4 (PhysicalModelAudit acceptance non-escape)**.** *An accepted* PhysicalModelAudit$^{\uparrow}$ *route cannot turn a model-signature comparison into a physical acceptance claim unless the source-trace row, target-phenomenon row, observed-signature row, model-signature row, classifier row, tower row, stability row, descent row, ledger row, failure-surface row, and strength row are all present in the same finite packet.*

Proof. Start with the obligation surface in Definition 1129.3. The accepted route factors through the displayed tuple

$$Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N.$$

The real-facing part of the route is not the model signature M alone. It also requires the source traces Q, target phenomenon R, observed signatures O, classifier C, tower T, stability and descent rows Y, D, ledger L, failure surface F, and strength row S.

If any one of those rows is absent, the route can still describe syntax or a partial finite comparison, but it cannot project the local physical acceptance name. The missing row is outside the carrier, so the corresponding physical claim is not accepted by this NameCert packet. □

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditPacket acceptance nonescape`

Theorem 1129.5 (PhysicalModelAudit ledger failure strength separation)**.** *For every accepted packet*

$$A = (Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N),$$

the ledger row L, failure row F, and strength row S are separate audit coordinates. The ledger row records provenance, compression, representation, idealization, and translation material; the failure row records finite defeating observations or signature mismatches; the strength row records the evidential grade licensed by the accepted route. None of the three rows can be reconstructed from either of the other two without leaving the displayed carrier.

Proof. Start with the carrier in Definition 1129.1. It lists L, F, and S as distinct rows of the same finite packet, with L assigned to audit material, F assigned to defeating observations or mismatches, and S assigned to evidential strength.

Apply the classifier in Definition 1129.2. The classifier transports ledger rows, failure surfaces, and strength rows componentwise; it does not collapse them into the observed signature, model signature, or comparison row.

Finally use the NameCert obligation surface and the acceptance non-escape theorem. Definition 1129.3 places ledger, failure, and strength admission in the local certificate, while Definition 1129.4 requires all three rows before a physical acceptance name can be projected. Hence an accepted audit cannot substitute ledger material for a defeat surface, a defeat surface for evidential strength, or strength for either of the audit rows. □

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditPacket ledger failure strength separation`

Theorem 1129.6 (PhysicalModelAudit route strength determinacy)**.** *For an accepted packet*

$$A = (Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N),$$

the audit strength row S is determined only by the displayed source, signature, classifier, tower, stability, descent, ledger, and failure route

$$Q, R, O, M, C, T, Y, D, L, F, H, U, P, N.$$

No host physical truth oracle, detached model equivalence, unbounded observation stream, or hidden validation route can determine the accepted audit strength outside the displayed packet.

Proof. Read the carrier first. Definition 1129.1 places S in the same finite tuple as the source trace Q, target phenomenon R, observed and model signatures O, M, classifier C, tower T, stability and descent rows Y, D, ledger L, failure surface F, and structural rows H, U, P, N.

The classifier fixes the comparison route. By Definition 1129.2, accepted comparison is a relation between O, M and transported copies of the same displayed rows; it is not a separate model oracle.

Use the obligation surface to bind the strength source. Definition 1129.3 admits S only together with the source, signature, classifier, tower, stability, descent, ledger, and failure rows. Then Definition 1129.4 blocks a physical acceptance name whenever any of those rows is absent.

Therefore the accepted strength is determined by the displayed route and by no hidden host evidence. A consumer that changes S without changing the listed source, signature, classifier, tower, stability, descent, ledger, failure, transport, route, provenance, or naming rows has left the accepted $\mathsf{PhysicalModelAudit}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditPacket route strength determinacy`

Theorem 1129.7 (PhysicalModelAudit failure surface exhaustion)**.** *For an accepted packet*

$$A = (Q, R, O, M, C, T, Y, D, L, F, S, H, U, P, N),$$

every defeating observation, signature mismatch, reuse failure, descent failure, or audit-level objection available to the physical model audit must appear through the failure row F and its displayed transports through H, U, P, N. No hidden host observation, unledgered mismatch, or detached failure witness can defeat the accepted route outside F.

Proof. Begin with the displayed packet. The carrier of Definition 1129.1 assigns finite defeating observations and signature mismatches to F, while $Q, R, O, M, C, T, Y, D, L, S$ record the source, phenomenon, signatures, classifier, tower, stability, descent, ledger, and strength rows.

The classifier does not add a second failure store. By Definition 1129.2, mismatch information is compared through the transported observed and model signatures together with the failure surface already present in the packet.

Apply Definition 1129.3. The local NameCert has a ledger, failure, and strength admission row, and its failure coordinate is the only named surface for defeating observations or signature mismatches.

Finally use Definition 1129.4. Acceptance requires F in the same finite packet as the source, signature, classifier, tower, stability, descent, ledger, and strength rows. Thus any alleged defeat that does not enter through F is not a defeating observation of the accepted $\mathsf{PhysicalModelAudit}^{\uparrow}$ route; it is outside the carrier rather than hidden host evidence. □

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.PhysicalModelAuditPacket failure surface exhaustion`

The Lean follow-up path `lean4/BEDC/Derived/PhysicalModelAuditUp/TasteGate.lean` must provide explicit Nontrivial PhysicalModelAudit$^{\uparrow}$ and FieldFaithful PhysicalModelAudit$^{\uparrow}$ instances. These instances must distinguish source traces, target phenomenon, observed and model signatures, classifier, tower, stability, descent, ledger, failure surface, strength, transports, routes, provenance, and local NameCert as separate fields.

Falsifiable prediction: If the checker or failure-surface row is absent, an accepted PhysicalModelAuditUp packet cannot promote finite fit beyond ModelSeed.

Independence witness: largemodelauditchannel, modeltrace, statespacemodel, onticstate, observationreflectionpacket: none is bijective because PhysicalModelAuditUp carries system-relative observation bundle, fit classifier, tower, stability/descent, failure, and strength rows together rather than only output auditing, trace storage, state-space presentation, ontic-source status, or observation-reflection routing.

Closure status for NameCert$_{\text{PhysicalModelAudit}^{\uparrow}}$.

Constructive story: A PhysicalModelAudit$^{\uparrow}$ packet is generated from finite source traces, target phenomenon, observed and model signatures, audit classifier, tower, stability, descent, ledger, failure surface, strength, componentwise transports, displayed routes, provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.PhysicalModelAuditUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{PhysicalModelAudit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1129.1, Definition 1129.2, Definition 1129.3, Definition 1129.4, Definition 1129.5, Definition 1129.6, and Definition 1129.7 bind the source-trace, target-phenomenon, observed-signature, model-signature, classifier, tower, stability, descent, ledger, failure, strength, hsame, Cont, Pkg, and NameCert rows of the accepted audit packet.

Formal verification: FormalStatus(NameCert$_{\text{PhysicalModelAudit}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.PhysicalModelAuditUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host-level physical truth, unbounded validation, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: The accepted public audit packet has mature paper closure; Lean verification, axiom-clean audit, and bridge checking remain separate verification-axis obligations.

Chapter 1130

A Concrete Naming Certificate for $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$

The $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ packet realizes the physical-truth decomposition of chapter 9414 as a finite BEDC NameCert carrier. It consumes the tower packet of chapter 1128, the reality-constrained approximation carrier of chapter 1127, the observation-reflection packet of chapter 1126, the physical-model audit surface of chapter 1129, and the open-fit surface of chapter 1132. Those siblings supply source access, signature comparison, approximation, handoff, audit, and refutation rows; this packet binds them into a single truth-certificate surface without adding model–reality identity or a final-fit oracle.

Definition 1130.1 (CertifiedPhysicalTruth carrier)**.** *A* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, G, K, A, D, I, L, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-observation row, recording the finite reality-facing traces or experiments admitted by the packet.*
2. *G is the reality-facing signature row extracted from S through the declared observation bundle.*
3. *K is the classifier row comparing the displayed signature with the candidate physical claim at the stated fit level.*
4. *A is the approximation-tower row through which mathematical approximation, completion, and prediction are read.*
5. *D is the descent-certificate row licensing exactly the operations that survive passage through A.*
6. *I is the observer-invariance certificate row, recording the displayed anchor-change or inter-observer route by which the claim remains invariant.*
7. *L is the ledger row for provenance, compression, idealization, translation, reuse domain, and accepted evidential strength.*
8. *F is the failure-surface row naming finite defeating observations, signature mismatches, and open residues.*
9. *H records componentwise* hsame *transport for source, signature, classifier, tower, descent, invariance, ledger, failure, replay, provenance, and local-name rows.*

10. *C records displayed* Cont *replay through $S, G, K, A, D, I, L, F, H, P, N$.*
11. *P is the* Pkg *provenance over chapter 9414, the cited concrete-instance siblings,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$ *row.*

The carrier has no coordinate for host-level truth, model–reality identity, unbounded observation, unledgered approximation, privileged observer access, or final physical fit outside the displayed packet.

Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.CertifiedPhysicalTruth carrier row injective`

Definition 1130.2 (CertifiedPhysicalTruth obligation surface)**.** *For the carrier of Definition 1130.1, the local* $\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$ *surface consists exactly of eight obligation rows:*

1. *source admission through S;*
2. *reality-facing signature admission through G;*
3. *classifier exactness through K;*
4. *approximation-tower accountability through A;*
5. *descent permission through D;*
6. *observer-invariance display through I;*
7. *ledger exactness through L;*
8. *failure-surface exposure through F.*

The structural rows H, C, P, N preserve, replay, source, and locally name those eight obligations, but they do not add another truth coordinate.

Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.CertifiedPhysicalTruth obligation surface row injective`

Theorem 1130.3 (CertifiedPhysicalTruth reality fit)**.** *Every accepted* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *truth read factors through the source row S, signature row G, classifier row K, approximation tower A, descent certificate D, observer-invariance row I, ledger L, failure surface F, componentwise transport H, replay row C, provenance P, and local name N. In particular, physical truth through this packet is certified fit with displayed stability, descent, invariance, ledger, and failure data, not identity between a model and reality.*

Proof. Begin with the carrier of Definition 1130.1. The accepted packet displays exactly the rows

$$S, G, K, A, D, I, L, F, H, C, P, N.$$

The source and signature rows S, G provide the reality-facing input, while K records the finite classifier comparison. The approximation row A and descent row D then state which mathematical operations survive the tower rather than converting the tower into physical identity.

The invariance, ledger, and failure rows I, L, F supply the remaining truth-certificate constraints. The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no further carrier coordinate is available, every accepted truth read must remain inside the displayed fit, stability, descent, invariance, ledger, and failure surface. □

Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.CertifiedPhysicalTruthCarrier reality fit`

Theorem 1130.4 (CertifiedPhysicalTruth failure surface)**.** *An accepted* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *packet must expose a displayed failure-surface row* F*. If a physical-truth claim asserts model–reality identity, or if the packet lacks a row naming finite defeating observations, signature mismatches, and open residues, then the claim is not accepted by* $\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.CertifiedPhysicalTruth failure surface row injective`

Proof. Use Definition 1130.2. Failure-surface exposure through F is one of the eight local NameCert obligations, so a packet without F lacks a required certificate row.

The same obligation surface separates the classifier K and tower A from any identity claim. They record finite fit and approximation accountability, while F records how the claim can be defeated. A model–reality identity assertion therefore asks the carrier to export a coordinate that the packet does not contain.

Consequently the accepted packet can certify only the displayed physical-truth surface. Removing F or replacing finite fit by identity leaves the local NameCert carrier. □

Theorem 1130.5 (CertifiedPhysicalTruth observer invariance)**.** *Every* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *export is invariant only through the displayed row* I *and its componentwise transport in* H*. A truth read depending on one observer history without the invariant route recorded in* I *is local observation content, not a certified physical-truth export.*

Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.CertifiedPhysicalTruthObserverInvariance`

Proof. Read the carrier fields for invariance. The only row that records observer-invariance data is I, and the only transport row preserving it componentwise is H.

The replay and provenance rows C and P do not introduce a privileged observer. They only replay and source the packet already transported through H.

Thus an accepted export must pass through I before being read as observer-invariant. If the claimed truth depends on a single observer history outside that route, the claim may remain a local finite observation, but it is not exported by $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$. □

Falsifiable prediction: If a CertifiedPhysicalTruthUp packet asserts model–reality identity or lacks a displayed failure-surface row, the packet is refuted.

Independence witness: realityconstrainedtowerpacket, realityconstrainedapproximationtower, observationreflectionpacket, physicalmodelaudit, realityconstrainedopenfit: each sibling supplies part of fit, signature, tower, handoff, audit, or refutation data, while none contains the full source, signature, classifier, tower, descent, invariant, ledger, failure-surface, transport, replay, provenance, and local NameCert truth-certificate packet.

The Lean TasteGate row provides `Nontrivial CertifiedPhysicalTruthUp` and `FieldFaithful CertifiedPhysicalTruthUp` instances in `lean4/BEDC/Derived/CertifiedPhysicalTruthUp/TasteGate.lean`.
Lean4 ✓ `BEDC.Derived.CertifiedPhysicalTruthUp.taste gate`

Theorem 1130.6 (CertifiedPhysicalTruth scoped transport coverage)**.** *For every accepted* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *packet, the rows* $S, G, K, A, D, I, L, F, H, C, P, N$ *cover every scoped transport and consumer read admitted by the chapter. Source, signature, classifier, approximation, descent, invariance, ledger, and failure reads are transported only through* H*; replay and provenance reads are exposed only through* C *and* P*; and the local certificate is named only by* N*.*

Proof. Project the carrier of Definition 1130.1. The packet contains exactly the source row S, signature row G, classifier row K, approximation row A, descent row D, invariance row I, ledger row L, failure row F, transport row H, replay row C, provenance row P, and local name N.

Apply Definition 1130.2. Its eight obligation rows are precisely the source, signature, classifier, approximation, descent, invariance, ledger, and failure surfaces; none is stored outside the displayed carrier.

Finally use Definition 1130.3, Definition 1130.4, and Definition 1130.5. These theorems force truth, refutation, and invariance consumers through the same displayed rows, with H, C, P, N supplying transport, replay, provenance, and local naming. Hence the scoped transport and consumer surface is exhausted by $S, G, K, A, D, I, L, F, H, C, P, N$. □

Theorem 1130.7 (CertifiedPhysicalTruth public surface)**.** *The public export of* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *is exactly the finite truth-certificate surface*

$$S, \quad G, \quad K, \quad A, \quad D, \quad I, \quad L, \quad F, \quad H, \quad C, \quad P, \quad N.$$

Downstream chapters may cite $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *as a public object through those rows, but they receive no model–reality identity, no final-fit oracle, no unbounded observation row, and no bridge check.*

Proof. Project the carrier and obligation surface of Definition 1130.1 and Definition 1130.2. They list the full source, signature, classifier, approximation, descent, invariance, ledger, failure, transport, replay, provenance, and local naming rows, and no further public coordinate.

Use the reality-fit, failure-surface, and observer-invariance theorems. The fit route consumes $S, G, K, A, D, I, L, F, H, C, P, N$, the failure route requires the displayed F row, and the invariance route passes through I and componentwise transport in H.

Finally apply scoped transport coverage, Definition 1130.6. Since every admitted consumer read is already covered by the same finite surface, public citation exposes exactly those rows and cannot add model–reality identity, a final-fit oracle, unbounded observation, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$.

Constructive story: A $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ packet is generated from finite source observations, a reality-facing signature, classifier, approximation tower, descent certificate, observer-invariance certificate, ledger, failure surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1130.1, Definition 1130.2, Definition 1130.3, Definition 1130.4, Definition 1130.5, Definition 1130.6, and Definition 1130.7 bind certified physical truth to the finite public surface $S, G, K, A, D, I, L, F, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CertifiedPhysicalTruth}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CertifiedPhysicalTruthUp.taste\_gate`

Bridge status: none

Not claimed: This chapter does not assert model–reality identity, host-level truth, final physical fit, unbounded observation, or bridge checking for $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$.

Upgrade path: The theory axis is complete; any remaining bridgeChecked claim belongs to the independent formal bridge axis.

Chapter 1131

A Concrete Naming Certificate for OpenPhysicalFit$^{\uparrow}$

The OpenPhysicalFit$^{\uparrow}$ packet realizes the finite-observation open-fit discipline of section 9414.13 as a finite BEDC NameCert carrier. It sits beside the $\mathsf{PhysicalFitTower}^{\uparrow}$ role, the $\mathsf{ObserverApproximationTower}^{\uparrow}$ role, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ of chapter 1124, and $\mathsf{ObservationReflectionPacket}^{\uparrow}$ of chapter 1126. Those siblings expose tower, observer, audit, and observation–reflection surfaces; OpenPhysicalFit$^{\uparrow}$ binds the finite fit certificate itself, its displayed refutation surface, and its open-continuation ledger without adding a final-fit oracle.

Definition 1131.1 (OpenPhysicalFit carrier)**.** *An* OpenPhysicalFit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (H, \Pi, O, M, C, F, E, L, R, N)$$

with the following displayed rows:

1. *H is the finite reality trace row, recording the observed history through which the physical system is read;*
2. *Π is the observation-bundle row, naming the admissible questions used to extract finite signatures;*
3. *O is the observed-signature row $\mathrm{ObsSig}_{\Pi}(R \mid H)$;*
4. *M is the model-generated signature row $\mathrm{ModSig}_{\Pi}(M \mid H)$;*
5. *C is the packet classifier row comparing O and M at the declared finite-fit level;*
6. *F is the refutation row, naming displayed finite observations whose observed and model-generated signatures fail C;*
7. *E is the open-continuation row, recording admissible observation extensions still outside the finite certificate;*
8. *L is the ledger row recording hidden source residue, idealization residue, undefeated finite-fit scope, and open adequacy gaps;*
9. *R is the replay row, recording the displayed* Cont *route through H, Π, O, M, C, F, E, L;*
10. *N is the local* $\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$ *row naming exactly this finite packet.*

The carrier has no coordinate for total physical identity, no internal proof of $\forall n, \mathrm{Fit}_n(M, R; \Pi)$, no final-fit oracle, and no route from finite success to unrestricted confirmation.

Lean4 ✓ `BEDC.Derived.OpenPhysicalFitUp.OpenPhysicalFitTasteGate single carrier alignment`

Theorem 1131.2 (OpenPhysicalFit NameCert obligations)**.** *For the carrier of Definition 1131.1, the local* $\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$ *surface consists of six obligation rows:*

1. *carrier admission for the displayed packet* $P = (H, \Pi, O, M, C, F, E, L, R, N)$*;*
2. *trace and bundle exactness through* H *and* Π*;*
3. *observed and model-generated signature exposure through* O *and* M*;*
4. *finite classifier accountability through* C*;*
5. *refutation and open-continuation visibility through* F *and* E*;*
6. *ledger, replay, and local naming closure through* L, R, N*.*

Every accepted finite open-fit read must display these rows before a consumer may treat the model as undefeated by the observed trace.

Lean4 ✓ `BEDC.Derived.OpenPhysicalFitUp.OpenPhysicalFit semantic name certificate`

Proof. Project the carrier of Definition 1131.1. The admitted object is the ten-row packet $P = (H, \Pi, O, M, C, F, E, L, R, N)$, so a consumer begins with the finite trace and the declared observation bundle.

The signature rows then expose the two sides compared by finite fit. The observed row O and model row M are the only signatures available to the classifier C.

The boundary rows make the open status explicit. The refutation row F records finite defeating observations, while the open-continuation row E records admissible extensions that are not collapsed into a total certificate.

Finally L, R, N ledger, replay, and name the same packet. Since no other coordinate is admitted, the NameCert surface contains exactly the six displayed obligation rows. □

Theorem 1131.3 (OpenPhysicalFit refutation boundary)**.** *An accepted* $\mathrm{OpenPhysicalFit}^{\uparrow}$ *packet is refuted by a displayed finite observation whose observed and model-generated signatures fail the packet classifier. Conversely, finite acceptance of* P *records only undefeatedness by the displayed trace and does not certify total fit, model-reality identity, or a final theory.*

Lean4 ✓ `BEDC.Derived.OpenPhysicalFitUp.OpenPhysicalFit refutation boundary`

Proof. Read the classifier and refutation rows of Definition 1131.1. A finite defeating observation belongs to F precisely when the observed and model-generated signatures cannot pass C.

Such a row is internal to the packet, so the accepted certificate itself names the finite observation that defeats the fit claim. The local NameCert obligations of Definition 1131.2 require this row to remain visible to consumers.

The reverse boundary follows from the absence of any total-fit coordinate. The packet records $H, \Pi, O, M, C, F, E, L, R, N$ only. None of those rows is an internal proof of all future finite fits, an identity between model and reality, or a final-fit oracle. □

Theorem 1131.4 (OpenPhysicalFit observation-extension stability)**.** *Let*

$$P = (H, \Pi, O, M, C, F, E, L, R, N)$$

be an accepted $\mathrm{OpenPhysicalFit}^{\uparrow}$ *packet. If an admissible extension of the observation trace remains classified by* C*, then it can be recorded as an extension of the same open-continuation ledger. If the extension fails* C*, the result is a refutation row rather than a hidden weakening of the packet.*

Lean4 ✓ `BEDC.Derived.OpenPhysicalFitUp.OpenPhysicalFit observation extension stability`

Proof. Start with the extension row E. It records admissible continuation observations without converting them into a final certificate.

When the extended observed and model-generated signatures pass C, the same finite classifier route can be replayed through R and recorded in the ledger L as another undefeated finite trace.

When the extended signatures fail C, the refutation row F is the only row that can receive the displayed mismatch. The packet therefore separates stable continuation from refutation and never hides a classifier failure inside the open ledger. □

Theorem 1131.5 (OpenPhysicalFit public certificate)**.** *The finite trace row, observation bundle, classifier, refutation row, open-continuation ledger, replay route, provenance row, and local* $\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$ *row form the public* $\mathrm{OpenPhysicalFit}^{\uparrow}$ *certificate. The certificate exports undefeated finite-fit status for the displayed trace and classifier while making no total-fit, model-reality identity, final-theory, or final-fit claim.*

Proof. Use the carrier of Definition 1131.1. The visible public rows are exactly the finite trace H, observation bundle Π, observed and model-generated signatures O, M, classifier C, refutation row F, open-continuation row E, ledger L, replay row R, and local name N.

Apply the semantic certificate of Definition 1131.2, the refutation boundary of Definition 1131.3, and the observation-extension stability theorem of Definition 1131.4. These rows expose the finite-fit status, the displayed defeating observations, and the admissible continuations as one public packet.

The carrier contains no coordinate for total fit, model-reality identity, a final theory, or a final-fit oracle. Therefore the public certificate exports only undefeated finite-fit status at the displayed boundary. □

Falsifiable prediction: An accepted OpenPhysicalFitUp packet is refuted by a displayed finite observation whose observed and model-generated signatures fail the packet classifier.

Independence witness: physicalfittower, observerapproximationtower, realityconstrainedmodelaudit, observationreflectionpacket, onticstate: none is bijective because $\mathrm{OpenPhysicalFit}^{\uparrow}$ carries the finite open-fit certificate, refutation row, and open-continuation ledger in one packet. $\mathsf{PhysicalFitTower}^{\uparrow}$ packages tower descent and hidden-source residue, $\mathsf{ObserverApproximationTower}^{\uparrow}$ packages observer-side approximation routes, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ packages audit strength and failure surfaces, $\mathsf{ObservationReflectionPacket}^{\uparrow}$ packages observation–reflection role correspondence, and $\mathsf{OnticState}^{\uparrow}$ names state rows; none of them alone gives the exact finite-fit classifier with both refutation and open-continuation rows.

The Lean follow-up in `lean4/BEDC/Derived/OpenPhysicalFitUp/TasteGate.lean` should provide `Nontrivial OpenPhysicalFitUp` and `FieldFaithful OpenPhysicalFitUp`, together with the future target `BEDC.Derived.OpenPhysicalFitUp.TasteGate.OpenPhysicalFitTasteGate_single_carrier`

Closure status for $\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$**.**

Constructive story: An $\mathrm{OpenPhysicalFit}^{\uparrow}$ packet is generated from a finite trace, observation bundle, observed signature, model-generated signature, classifier, refutation row, open-continuation row, ledger, replay route, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1131.1, Definition 1131.2, Definition 1131.3, Definition 1131.4, and Definition 1131.5 read the mature finite trace, observation bundle, observed and model-generated signatures, classifier, refutation row, open-continuation ledger, replay route, provenance row, and local naming scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathrm{OpenPhysicalFit}^{\uparrow}}$, unformalized)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.openPhysicalFitDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total physical fit, model-reality identity, final theory, or a final-fit oracle.

Upgrade path: No further paper-side theory upgrade is pending before Lean verification and bridge checking for the same finite trace, observation bundle, signatures, classifier, refutation, open-continuation, ledger, replay, and local naming rows.

Chapter 1132

A Concrete Naming Certificate for $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$

The $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ packet realizes the finite-observation and open-fit discipline of section 9414.13 as a concrete BEDC NameCert carrier. It sits after the audit packet of chapter 1124, the observation-reflection packet of chapter 1126, the finite observation interface of chapter 1007, and the observer-budget support surface of chapter 1121. Those siblings supply audit, reflection, finite-window, and budget rows; this carrier binds finite fit to an open continuation and a finite refutation surface without exporting a final-fit oracle.

Definition 1132.1 (Reality-constrained open-fit carrier)**.** *A* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (H, \Pi, O, M, F, Q, L, R, N)$$

with the following displayed rows:

1. *H is the finite trace row through which the reality-facing system is observed;*
2. *Π is the observation-bundle row naming the admissible finite questions;*
3. *O is the observed-signature row extracted from H through Π;*
4. *M is the model-signature row generated by the candidate model over the same finite trace;*
5. *F is the finite-fit certificate row comparing O and M at the displayed prefix;*
6. *Q is the open-continuation surface recording the admissible next observations still outside the finite fit row;*
7. *L is the ledger row recording observation provenance, accepted fit, undecided continuation, and refused final-fit claims;*
8. *R is the refutation row naming displayed defeating observations that invalidate the finite-fit certificate;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedOpenFit}^{\uparrow}}$ *row.*

The carrier contains no coordinate for total model identity, no final-fit oracle, no host-level physical truth predicate, and no unbudgeted observation route beyond the displayed continuation and refutation rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedOpenFitUp.TasteGate.RealityConstrainedOpenFitTasteGate`
`single carrier alignment`

Theorem 1132.2 (Reality-constrained open-fit NameCert obligations)**.** *For the carrier of Definition 1132.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedOpenFit}^{\uparrow}}$ *surface consists exactly of carrier admission for* $U = (H, \Pi, O, M, F, Q, L, R, N)$*, signature exactness through* O *and* M*, finite-fit exactness through* F*, open-continuation preservation through* Q*, ledger exactness through* L*, finite refutation exposure through* R*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedOpenFitUp.RealityConstrainedOpenFitCarrier namecert obligations`

Proof. Carrier admission projects the nine displayed rows of Definition 1132.1. The trace, bundle, observed signature, model signature, fit certificate, continuation surface, ledger, refutation row, and local name are the complete finite packet.

Signature and finite-fit exactness follow by reading O, M, and F together. The fit row compares only the observed and model signatures available at the displayed prefix, so it cannot be detached as a total confirmation claim.

Open-continuation preservation and ledger exactness keep Q and L visible on every consumer route. The continuation row records the remaining observation surface, while the ledger records which fit claims, undecided continuations, and refused final-fit demands are part of the packet.

Finally the refutation row R supplies the finite defeating surface. Since the carrier has no coordinate beyond $H, \Pi, O, M, F, Q, L, R, N$, local naming through N cannot export model identity, a final-fit oracle, or an unbudgeted observation route. □

Theorem 1132.3 (Reality-constrained open-fit finite refutation surface)**.** *Every accepted* RealityConstrainedOpenFi *packet exposes refutation only through the displayed row* R*. A single defeating observation may refute the finite-fit certificate* F *when it is recorded in* R*, but the open-continuation row* Q *does not become a final confirmation surface when no such defeating observation has yet appeared.*

Proof. Read the carrier inventory. The only row that names defeating observations is R, and the only row that records finite fit is F.

If a defeating observation is displayed in R, the packet has a finite witness against the corresponding fit row. This is a prefix-level refutation of F, not a judgment about all possible continuations.

If no defeating observation has appeared, the continuation row Q remains open. It records admissible next observations still outside the finite prefix and therefore cannot certify final agreement between the model and the reality-facing system. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedOpenFitUp.RealityConstrainedOpenFitFiniteRefutationSurface`

Theorem 1132.4 (Reality-constrained open-fit ledger non-escape)**.** *Every consumer of an accepted* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *packet must read finite fit through the trace* H*, bundle* Π*, observed signature* O*, model signature* M*, finite-fit certificate* F*, open continuation surface* Q*, ledger* L*, refutation row* R*, and local name* N*. No consumer may detach* F *as model identity, total confirmation, a final-fit oracle, host physical truth, or an unledgered future-observation claim.*

Proof. Use the NameCert obligations above. They require every accepted read to pass through the displayed signature, finite-fit, continuation, ledger, refutation, and naming rows.

The ledger row L records the status of accepted finite fit, undecided continuation, and refused final-fit demands. A consumer route that omits L has no place to distinguish finite support from total confirmation.

The continuation and refutation rows Q and R then enforce non-escape. Future observations remain open through Q, while finite defeat is displayed through R; neither row supplies model identity, host physical truth, or a final-fit oracle. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedOpenFitUp.RealityConstrainedOpenFitLedgerNonescape`

The later Lean file `lean4/BEDC/Derived/RealityConstrainedOpenFitUp/TasteGate.lean` must provide explicit Nontrivial RealityConstrainedOpenFit$^{\uparrow}$ and FieldFaithful RealityConstrainedOpenFit$^{\uparrow}$ instances preserving $H, \Pi, O, M, F, Q, L, R, N$.

Falsifiable prediction: A single displayed defeating observation refutes the finite-fit row, while no finite prefix exports a final-fit oracle.

Independence witness: reality-constrained-model-audit, observationreflectionpacket, finite-observation-interface, observer-budget-support, onticstate: none is bijective because RealityConstrainedOpen-FitUp stores the open-fit relation and its finite refutation surface rather than an audit packet, reflection packet, observation window, support budget, or ontic source row.

Closure status for NameCert$_{\text{RealityConstrainedOpenFit}^{\uparrow}}$.

Constructive story: A RealityConstrainedOpenFit$^{\uparrow}$ packet is generated from a finite trace, observation bundle, observed signature, model signature, finite-fit certificate, open-continuation surface, ledger, finite refutation row, and local NameCert row.

Lean4 ✓ `BEDC.Derived.RealityConstrainedOpenFitUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedOpenFit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1132.1, Definition 1132.2, Definition 1132.3, and Definition 1132.4 bind the finite trace H, observation bundle Π, observed signature O, model signature M, finite-fit row F, continuation surface Q, ledger row L, refutation row R, local NameCert row N, TasteGate nontriviality, and field-faithfulness rows.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedOpenFit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RealityConstrainedOpenFitUp.TasteGate.RealityConstrainedOpenFitTasteGate\_single\_carri`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not assert model identity, total confirmation, final-fit oracles, host physical truth, or bridge checking for RealityConstrainedOpenFit$^{\uparrow}$.

Upgrade path: Only machine bridge checking remains before the formal axis reaches bridgeCheckedV over the same finite trace, bundle, signature, fit, continuation, ledger, refutation, and local NameCert rows.

Chapter 1133

A Concrete Naming Certificate for $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$

The $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ packet realizes the relational objectivity discipline of section 9414.30 as a finite multi-observer NameCert surface. It consumes the physical signature and descent rows of the $\mathsf{PhysicalFitTower}^{\uparrow}$ role, the observation handoff rows of chapter 1126, the observer-facing approximation route of the $\mathsf{ObserverApproximationTower}^{\uparrow}$ role, and the model-audit rows of chapter 1124. Those siblings provide fit, observation handoff, approximation, and audit carriers; this packet adds the symmetric anchor-change classifier and the refusal row that prevents any one observer history from becoming a global observer.

Definition 1133.1 (Reality-constrained objectivity carrier)**.** *A* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (H, A, I, F, R, T, P, L, N)$$

with the following displayed rows:

1. *H is the finite multi-observer history family, containing the observer histories whose relation is being certified;*
2. *A is the anchor row, naming the currently selected observer standpoint and the admissible anchor-change schedule;*
3. *I is the inter-observer invariant packet, containing the relation, classifier, and invariant value transported across the histories;*
4. *F is the* $\mathsf{PhysicalFitTower}^{\uparrow}$ *fit row that exposes the reality-facing signature discipline consumed by the invariant;*
5. *R is the* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *handoff row that records how observation and reflection data enter the invariant packet;*
6. *T is the* $\mathsf{ObserverApproximationTower}^{\uparrow}$ *row that bounds the observer-facing approximation route used by terminal consumers;*
7. *P is the* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *row that records the model-facing audit surface and failure rows;*
8. *L is the no-global-observer ledger, recording anchor changes, rejected privileged observers, and the finite routes by which invariant stability is checked;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedObjectivity}^{\uparrow}}$ *row naming the whole packet.*

The carrier has no coordinate for a view from nowhere, no hidden synchronizing observer, and no objectivity claim outside the displayed multi-observer, invariant, fit, handoff, approximation, audit, ledger, and naming rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedObjectivityUp.RealityConstrainedObjectivityTasteGate single carrier alignment`

Definition 1133.2 (Reality-constrained objectivity certificate)**.** *For accepted packets*

$$O = (H, A, I, F, R, T, P, L, N) \quad \text{and} \quad O' = (H', A', I', F', R', T', P', L', N'),$$

the certificate classifier $\mathsf{RealityConstrainedObjectivityClassifier}(O, O')$ *holds exactly when the multi-observer histories are related by displayed* hsame *transports, the anchor-change rows carry A to A' and back without selecting a privileged history, the invariant packets I and I' preserve their relation and value under the same classifier, and the fit, handoff, approximation, audit, ledger, and naming rows transport componentwise.*

Thus certificate agreement is not local certainty inside one observer history. It is symmetric preservation of the invariant packet under admissible anchor change, with every physical-fit, observation-handoff, approximation, model-audit, and no-global-observer row still visible.

Theorem 1133.3 (Reality-constrained objectivity anchor stability)**.** *Let $O = (H, A, I, F, R, T, P, L, N)$ be an accepted* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *packet. If an objectivity claim is exported by O, then the claim must be read from the invariant packet I through an anchor-change route recorded in A and L. A claim depending on one observer history without such a symmetric route is local content, not a* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *export.*

Proof. Read the carrier of Definition 1133.1. The only rows that can export an objectivity claim are the multi-observer family H, anchor row A, invariant packet I, sibling carrier rows F, R, T, P, ledger L, and local NameCert row N.

Now use the certificate classifier of Definition 1133.2. It requires anchor-change transport in both directions and preservation of the invariant packet under the same classifier; it does not accept one observer history as a distinguished frame.

Therefore a route through one history alone lacks the anchor-change row and the no-global-observer ledger. It remains local observational content even if it is internally stable for that history. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedObjectivityUp.RealityConstrainedObjectivityCarrier anchor stability`

Theorem 1133.4 (Reality-constrained objectivity NameCert obligations)**.** *For the carrier of Definition 1133.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedObjectivity}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission: the accepted object is exactly $O = (H, A, I, F, R, T, P, L, N)$;*
2. *classifier stability: every accepted comparison uses the certificate classifier of Definition 1133.2;*
3. *anchor-stability exactness: objectivity exports must preserve the invariant packet I under the anchor-change rows recorded in A and L;*
4. *sibling-carrier accountability: the physical-fit, observation-handoff, observer-approximation, and model-audit rows are consumed only through F, R, T, P;*

5. *non-escape: no consumer may infer objectivity from a single observer history, hidden global observer, detached model audit, unbounded observation route, or row outside* $H, A, I, F, R, T, P, L, N$.

Proof. Carrier admission is projection from Definition 1133.1. The packet lists the only rows available to the local certificate.

Classifier stability follows from Definition 1133.2. Comparisons transport the observer histories, anchor rows, invariant packet, sibling carriers, ledger, and naming row as one finite object rather than as detached local claims.

Anchor-stability exactness is the distinguished use of A, I, L. The invariant packet can be exported only when the anchor-change schedule and no-global-observer ledger record the symmetric route that preserves it.

Sibling-carrier accountability is forced by the rows F, R, T, P. Physical fit, observation handoff, approximation, and model audit data are consumed through the displayed sibling carriers, so no hidden adequacy or reflection route can be inserted after admission.

The non-escape row is the finite-packet conclusion. A claim requiring a single privileged observer, unbounded observation, detached audit, or absent coordinate has no projection from O. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedObjectivityUp.RealityConstrainedObjectivityUp StdBridge`

Theorem 1133.5 (Reality-constrained objectivity falsifiable separation)**.** *An accepted inter-observer invariant packet that factors every admitted comparison through one anchor row is rejected by the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *classifier as non-objective. Such a packet may still carry local or model-audit content, but it does not export relational objectivity because the no-global-observer ledger cannot certify symmetric anchor change.*

Proof. Assume the admitted comparisons all factor through one anchor row. Then the route has a distinguished observer standpoint that orders or evaluates the other histories from that row.

Apply the anchor-stability theorem. A $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ export must carry the invariant through the anchor-change schedule and no-global-observer ledger, not through a single privileged row.

The certificate classifier therefore rejects the packet as an objectivity certificate. The remaining rows may still support local content, fit comparison, observation handoff, approximation, or model audit, but they do not supply the symmetric anchor-change surface required for relational objectivity. □

Falsifiable prediction: If every admitted inter-observer invariant packet factors through a single anchor row, then the RealityConstrainedObjectivity carrier rejects it as non-objective.

Independence witness: physicalfittower, observationreflectionpacket, observerapproximationtower, realityconstrainedmodelaudit: fit, observation handoff, approximation tower, and model-audit carriers each lack the symmetric anchor-change classifier and no-global-observer refusal row.

The Lean follow-up must provide `Nontrivial RealityConstrainedObjectivityUp` and `FieldFaithful RealityConstrainedObjectivityUp` instances in `lean4/BEDC/Derived/RealityConstrainedObjectivityUp/Ta`

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedObjectivity}^{\uparrow}}$.
Lean4 ✓ `BEDC.Derived.RealityConstrainedObjectivityUp.taste gate`

Constructive story: A $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ packet is generated from a finite multi-observer history family, anchor-change row, inter-observer invariant packet, physical-fit row, observation handoff row, observer-approximation row, model-audit row, no-global-observer ledger, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedObjectivity}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1133.1, Definition 1133.2, Definition 1133.3, Definition 1133.4, Definition 1133.5, and the TasteGate carrier row bind the finite multi-observer carrier, certificate classifier, anchor-stability boundary, five NameCert obligation rows, and the public privileged-anchor rejection surface.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedObjectivity}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.RealityConstrainedObjectivityUp.RealityConstrainedObjectivityTasteGate\_single\_carrier`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not prove bridge checking, global synchronization, a final observer, host-level physical truth, unbounded observation, or completeness for every future observer configuration.

Upgrade path: no further upgrade – chapter at top of formal axis

Chapter 1134

A Concrete Naming Certificate for $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$

The $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$ packet realizes the negative test implicit in the reality-constrained objectivity vision of section 9414.30. It sits beside the positive $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ carrier of chapter 1133, the relational objectivity carrier of chapter 1138, the no-global-synchronization refusal boundary of chapter 1003, the observer-frame refusal surface, and the inter-history invariant surface of chapter 656. Those siblings certify positive invariant routes or refusal-only boundaries; this packet records the finite failed-invariant and failed-anchor-stability rows that make an asserted objectivity claim refutable.

Definition 1134.1 (Objectivity refutation boundary carrier)**.** *An* $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (H, K, A, W, R, T, P, N)$$

with the following displayed rows:

1. *H is the finite multi-observer packet over which an objectivity claim is asserted;*
2. *K is the failed inter-observer invariant row, recording that no displayed* $\mathsf{InterHistInvariant}^{\uparrow}$ *row preserves the asserted claim across the observer packet;*
3. *A is the failed anchor-stability row, recording that no anchor-change table transports the claim without selecting a privileged observer;*
4. *W is the witness row naming the concrete claim, observer slots, missing invariant coordinates, and missing anchor-stability coordinates;*
5. *R is the refusal ledger, routing the failed claim through* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *and* $\mathsf{ObserverFrameRefusal}^{\uparrow}$ *rather than through a positive objectivity certificate;*
6. *T records componentwise* hsame *transport and displayed* Cont *replay for the failed-invariant, failed-anchor, witness, refusal, provenance, and local naming rows;*
7. *P is the* Pkg *provenance over* $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$, $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{ObserverFrameRefusal}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ObjectivityRefutationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a positive invariant certificate, anchor-stable objectivity export, global synchronization map, privileged observer, quotient of observer histories, or hidden host-side truth value.

Lean4 ✓ `BEDC.Derived.ObjectivityRefutationBoundaryUp.TasteGate.ObjectivityRefutationBoundaryTasteGate single carrier alignment`

Theorem 1134.2 (Objectivity refutation boundary NameCert obligations)**.** *For the carrier of Definition 1134.1, the local* $\mathsf{NameCert}_{\mathsf{ObjectivityRefutationBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$B = (H, K, A, W, R, T, P, N),$$

failed-invariant exposure through K, failed anchor-stability exposure through A, concrete witness display through W, refusal-ledger routing through R, componentwise transport and replay through T, package provenance through P, and local naming through N. An asserted objectivity claim over the packet H fails this boundary whenever no inter-observer invariant and no anchor-stability row are displayed.

Lean4 ✓ `BEDC.Derived.ObjectivityRefutationBoundaryUp.ObjectivityRefutationBoundaryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1134.1. The rows H, K, A, W, R, T, P, N are the only rows available to the local certificate.

The failed-invariant row K and failed-anchor row A are the decisive certificate data. If both are absent as positive rows, the witness row W names the claim and the observer slots where the attempted objectivity export lacks support.

The refusal ledger R then routes the packet through the no-global-synchronization and observer-frame-refusal surfaces rather than through positive objectivity. Transport, replay, provenance, and local naming are supplied by T, P, N, so every obligation is one of the displayed rows. □

Theorem 1134.3 (Objectivity refutation boundary non-escape)**.** *Every consumer of an accepted* $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$ *packet factors through the finite rows*

$$H, K, A, W, R, T, P, N.$$

The consumer may report the asserted claim as refuted for the displayed multi-observer packet, but it cannot convert the failed-invariant row or failed-anchor-stability row into a positive $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *export, a* $\mathsf{RelationalObjectivity}^{\uparrow}$ *certificate, a global synchronization map, a privileged observer frame, a quotient stream equality, or a host-side truth value.*

Proof. Project the accepted packet to $B = (H, K, A, W, R, T, P, N)$. The only claim-facing rows are the failed invariant K, failed anchor-stability row A, concrete witness W, and refusal ledger R.

Apply the NameCert obligations. They identify a refutation boundary precisely when the asserted objectivity claim has no displayed inter-observer invariant and no displayed anchor-stability row.

No coordinate of the packet is a positive objectivity certificate, global synchronization map, privileged frame, quotient equality, or host truth value. A consumer that exports such data has therefore left the finite packet rather than reading the accepted boundary. □

Lean4 ✓ `BEDC.Derived.ObjectivityRefutationBoundaryUp.ObjectivityRefutationBoundaryCarrier nonescape`

Theorem 1134.4 (Objectivity refutation boundary sibling independence)**.** *The* ObjectivityRefutationBoundary$^{\uparrow}$ *packet is not bijective to* RealityConstrainedObjectivity$^{\uparrow}$, RelationalObjectivity$^{\uparrow}$, NoGlobalSyncBoundary$^{\uparrow}$, ObserverFrameRefusal$^{\uparrow}$, *or* InterHistInvariant$^{\uparrow}$. *The positive objectivity siblings carry invariant and anchor-stability rows rather than failed rows; the refusal-only siblings carry synchronization or frame refusals without the failed-invariant witness; and* InterHistInvariant$^{\uparrow}$ *carries invariant readback rather than the paired absence of invariant and anchor-stability evidence.*

Lean4 ✓ `BEDC.Derived.ObjectivityRefutationBoundaryUp.ObjectivityRefutationBoundaryCarrier sibling independence`

Proof. Compare the carrier rows. RealityConstrainedObjectivity$^{\uparrow}$ and RelationalObjectivity$^{\uparrow}$ include positive invariant and anchor-stability surfaces, while ObjectivityRefutationBoundary$^{\uparrow}$ records the failed invariant row K, failed anchor-stability row A, and concrete witness row W.

The no-global-synchronization and observer-frame refusal siblings supply refusal ledgers, but they do not name a failed objectivity claim, the missing invariant coordinates, and the missing anchor-stability coordinates in one packet. They are refusal-only boundaries rather than objectivity-refutation witnesses.

Finally, InterHistInvariant$^{\uparrow}$ supplies a positive invariant readback surface. Removing that readback and replacing it by the failure rows K, A, W changes the carrier shape and the consumer obligation surface, so no bijection preserving carrier rows, refusal ledger, transport, provenance, and local naming identifies the packets. □

Falsifiable prediction: An asserted objectivity claim over a multi-observer packet fails the ObjectivityRefutationBoundaryUp test whenever no inter-observer invariant and no anchor-stability row are displayed.

Independence witness: realityconstrainedobjectivity, relationalobjectivity, noglobalsyncboundary, observerframerefusal, interhistinvariant: no sibling is bijective to ObjectivityRefutationBoundary$^{\uparrow}$ because this carrier combines failed-invariant, failed-anchor-stability, witness, refusal-ledger, transport, provenance, and local NameCert rows. The positive objectivity siblings certify invariant export; the refusal-only siblings lack the objectivity-failure witness; the invariant sibling supplies readback rather than the paired failure packet.

The Lean TasteGate surface for `lean4/BEDC/Derived/ObjectivityRefutationBoundaryUp/TasteGate.lean` includes explicit Nontrivial ObjectivityRefutationBoundary$^{\uparrow}$ and FieldFaithful ObjectivityRefutationBoundary$^{\uparrow}$ instances preserving the failed-invariant row, failed-anchor-stability row, witness row, refusal ledger, transport, provenance, and local naming row.

Closure status for NameCert$_{\text{ObjectivityRefutationBoundary}^{\uparrow}}$ **·**

Constructive story: An ObjectivityRefutationBoundary$^{\uparrow}$ packet is generated from a finite multi-observer packet, failed inter-observer invariant row, failed anchor-stability row, witness row, refusal ledger, componentwise transport and replay, package provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObjectivityRefutationBoundary}^{\uparrow}}$, obligation)

Scope closed: Definition 1134.1, Definition 1134.2, Definition 1134.3, Definition 1134.4, and the displayed TasteGate marker `BEDC.Derived.ObjectivityRefutationBoundaryUp.TasteGate.ObjectivityRefutationBoundaryTasteGate_single_carrier_` bind the finite failed-invariant, failed-anchor-stability, witness, refusal-ledger, transport, provenance, naming, nonescape, sibling-independence, nontriviality, and field-faithfulness rows.

Formal verification: FormalStatus(NameCert$_{\text{ObjectivityRefutationBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct a positive objectivity certificate, global synchronization map, privileged observer frame, quotient equality, host truth value, Lean verification, or bridge checking for ObjectivityRefutationBoundary$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, failed-invariant exposure, failed-anchor-stability exposure, witness display, refusal-ledger routing, non-escape, TasteGate nontriviality, and field faithfulness for the displayed rows.

Chapter 1135

A Concrete Naming Certificate for $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$

The $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ packet realizes the open-continuation discipline of section 9414.25 as a finite NameCert surface. It consumes the open-fit row of $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ from chapter 1132, the tower row of $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ from chapter 1127, the physical-signature row of $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ from **??**, the multi-observer objectivity row of $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ from chapter 1133, and the audit row of $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ from chapter 1124. Those siblings supply finite fit, approximation, signature comparison, objectivity, and audit carriers; this packet binds them to a displayed continuation class, stability basis, failure surface, ledger, and local name without closing future observation.

Definition 1135.1 (Reality-constrained physical-induction carrier)**.** *A* $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (H, U, T, S, O, A, C, B, F, L, K, P, N)$$

with the following displayed rows:

1. *H is the finite observed history at which the induction claim is licensed;*
2. *U is the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *row recording finite fit and open continuation;*
3. *T is the* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *row recording the approximation tower used by the model;*
4. *S is the* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ *row recording physical-signature compatibility;*
5. *O is the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *row recording the multi-observer invariant route for the claim;*
6. *A is the* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *row recording audit strength and known failure obligations;*
7. *C is the continuation-class row naming the admissible future histories to which the claim is allowed to speak;*
8. *B is the stability-basis row naming the physical, observational, and model-side invariants whose preservation licenses the continuation read;*
9. *F is the failure-surface row naming displayed continuations that defeat the predicted fit;*

10. *L is the induction ledger recording finite fit, continuation scope, stability basis, failure surface, and refused final-fit demands;*
11. *K stores componentwise* hsame *transport for $H, U, T, S, O, A, C, B, F, L$;*
12. *P stores* Pkg *provenance over the sibling carriers,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}$ *row.*

The carrier has no coordinate for a final induction oracle, no prediction route outside C, B, F, L, and no way to convert finite fit into future fit without the displayed stability-basis and failure-surface rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedPhysicalInductionUp.RealityConstrainedPhysicalInductionTasteGate`
`single carrier alignment`

Theorem 1135.2 (Reality-constrained physical-induction NameCert obligations)**.** *For the carrier of Definition 1135.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}$ *surface consists exactly of seven obligation rows:*

1. *carrier admission for $I = (H, U, T, S, O, A, C, B, F, L, K, P, N)$;*
2. *finite-fit exposure through the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *row U and the observed history H;*
3. *tower and signature accountability through T and S;*
4. *objectivity and audit accountability through O and A;*
5. *continuation scope through the row C;*
6. *stability-basis and failure-surface exposure through B and F;*
7. *ledger, transport, provenance, and local naming through L, K, P, N.*

Any consumer of a physical-induction prediction must display all seven rows before reading finite fit as licensed continuation fit.

Lean4 ✓ `BEDC.Derived.RealityConstrainedPhysicalInductionUp.RealityConstrainedPhysicalInductionNameCert(`

Proof. Carrier admission is projection from Definition 1135.1. The packet exposes the observed history, five sibling carrier rows, continuation class, stability basis, failure surface, ledger, transport, provenance, and local name as its complete finite surface.

Finite-fit exposure is inherited through U and H. Tower and signature accountability are read through T and S, while objectivity and audit accountability are read through O and A. These rows keep the prediction attached to the same approximation, physical-signature, inter-observer, and model-audit data that licensed the finite fit.

The continuation row C bounds the future histories under discussion. The stability-basis row B states what must persist, and the failure-surface row F states what would defeat the licensed read.

Finally L, K, P, N record the induction ledger, componentwise transport, provenance, and local name. Since the carrier has no further coordinate, a consumer that omits one of the listed rows has no accepted physical-induction route. □

Theorem 1135.3 (Reality-constrained physical-induction non-escape)**.** *Every accepted* RealityConstrainedPhysicalI *consumer read factors through*

$$H, U, T, S, O, A, C, B, F, L, K, P, N.$$

In particular, no consumer may read a future-continuation prediction from finite fit alone, from an approximation tower alone, from signature fit alone, from objectivity alone, or from model audit alone. The displayed continuation class, stability-basis row, failure-surface row, and induction ledger remain mandatory boundary data.

Lean4 ✓ BEDC.Derived.RealityConstrainedPhysicalInductionUp.RealityConstrainedPhysicalInductionCarrier nonescape

Proof. Apply the NameCert obligations of Definition 1135.2. They require finite-fit exposure, tower and signature accountability, objectivity and audit accountability, continuation scope, stability-basis exposure, failure-surface exposure, and ledgered naming before a prediction can be consumed.

Each sibling row supplies only one side of the route. Open fit supplies finite fit and open continuation; the approximation tower supplies model-side descent; signature fit supplies physical-signature compatibility; objectivity supplies an inter-observer invariant route; model audit supplies audit strength and failure obligations.

The continuation, stability, failure, and ledger rows C, B, F, L are therefore not optional annotations. They are the coordinates that prevent finite fit from becoming a final induction oracle. Transport and provenance preserve this displayed packet but do not add an unrecorded continuation claim. □

Theorem 1135.4 (Reality-constrained physical-induction continuation scope)**.** *An accepted* RealityConstrainedPhys *packet*

$$I = (H, U, T, S, O, A, C, B, F, L, K, P, N)$$

licenses a continuation prediction only through the jointly displayed finite-fit row U, continuation class C, stability basis B, failure surface F, induction ledger L, transport row K, provenance row P, and local NameCert *row N. The row U supplies finite fit only over the observed prefix H; the rows C, B, F, L give the irreducible continuation boundary; and K, P, N keep the prediction route inside the same finite packet.*

Proof. Begin with the carrier of Definition 1135.1. Its coordinates display $H, U, T, S, O, A, C, B, F, L, K, P, N$ as the complete accepted packet, and there is no coordinate for a continuation read outside those rows.

Apply the NameCert obligations of Definition 1135.2. They require finite-fit exposure through U and H, continuation scope through C, stability and failure exposure through B and F, and ledgered naming through L, K, P, N before a prediction can be consumed.

Apply the non-escape theorem Definition 1135.3. It rules out reading future-continuation content from finite fit, approximation, signature fit, objectivity, or audit rows alone, so the only licensed route is the displayed continuation packet.

Thus U contributes only the observed-prefix fit, while C, B, F, L state the continuation class, persistence basis, defeat surface, and ledger for the read. The rows K, P, N preserve transport, provenance, and local naming inside the same finite packet. The theorem therefore asserts continuation scope only; it does not assert total future fit, host physical truth, or a final induction oracle. □

Lean4 ✓ BEDC.Derived.RealityConstrainedPhysicalInductionUp.RealityConstrainedPhysicalInductionCarrier continuation scope

Theorem 1135.5 (Reality-constrained physical-induction scoped kernel route)**.** *Every accepted* $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ *packet scopes its physical-induction read exactly over the displayed carrier rows*

$$H, U, T, S, O, A, C, B, F, L, K, P, N,$$

the ambient BHist *history discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}$ *row, and the TasteGate surface for the same carrier. The scoped route exposes finite fit, approximation, signature, objectivity, audit, continuation, stability, failure, and ledger data, but it does not add a final induction oracle or a prediction route outside those rows.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedPhysicalInductionUp.taste gate`

Proof. Start with the carrier of Definition 1135.1. Its tuple lists $H, U, T, S, O, A, C, B, F, L, K, P, N$ as the complete packet, and the definition explicitly withholds any coordinate for a final induction oracle.

Apply Definition 1135.2, Definition 1135.3, and Definition 1135.4. Together they force every consumer read through the finite-fit, approximation, signature, objectivity, audit, continuation, stability, failure, ledger, transport, replay, provenance, and local naming rows.

The TasteGate marker is attached to the same carrier surface. It records the accepted single-carrier alignment without changing the paper closure axis, so the scoped route is exactly the displayed kernel route plus the TasteGate surface and no hidden future-fit principle. □

Theorem 1135.6 (Reality-constrained physical-induction public continuation boundary)**.** *Every public consumer of an accepted* $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ *packet reads physical induction through the continuation class* C*, stability basis* B*, failure surface* F*, induction ledger* L*, transport row* K*, provenance row* P*, and local naming row* N *displayed in the same carrier*

$$I = (H, U, T, S, O, A, C, B, F, L, K, P, N).$$

The public interface may expose finite fit, tower accountability, signature compatibility, objectivity, audit strength, continuation scope, stability conditions, and failure boundaries. It may not export a final induction oracle, total future fit, host physical truth, or a prediction outside the displayed continuation and failure-surface rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedPhysicalInductionUp.RealityConstrainedPhysicalInductionCarrier public continuation boundary`

Proof. Start from the scoped kernel route of Definition 1135.5. It binds every physical-induction read to the displayed carrier rows and to the TasteGate surface for that same carrier.

Apply the continuation-scope theorem Definition 1135.4. The rows C, B, F, L are exactly the public continuation boundary: C names the admissible continuations, B names the persistence basis, F names the defeating continuations, and L records the finite induction ledger.

Apply the non-escape theorem Definition 1135.3. It blocks reads from finite fit, approximation, signature fit, objectivity, or audit rows alone, so a public consumer receives only the displayed continuation, stability, failure-surface, transport, provenance, and naming route. □

Falsifiable prediction: Any future-continuation read that bypasses either the displayed stability-basis row or the failure-surface row refutes the RealityConstrainedPhysicalInduction carrier.

Independence witness: reality-constrained-open-fit, reality-constrained-approximation-tower, reality-constrained-signature-fit, reality-constrained-objectivity, reality-constrained-model-audit: none is bijective because $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ binds finite fit, approximation tower, physical-signature compatibility, inter-observer objectivity, model audit, continuation scope, stability basis, failure surface, induction ledger, transport, provenance, and local NameCert rows in one carrier. Each sibling lacks at least the displayed continuation, stability-basis, failure-surface, or induction-ledger coordinate required for physical induction.

The Lean follow-up must provide `Nontrivial RealityConstrainedPhysicalInductionUp` and `FieldFaithful RealityConstrainedPhysicalInductionUp` instances in `lean4/BEDC/Derived/RealityCon`

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$ packet is generated from finite history, open-fit, approximation-tower, signature-fit, objectivity, model-audit, continuation-class, stability-basis, failure-surface, induction-ledger, transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1135.1, Definition 1135.2, Definition 1135.3, Definition 1135.4, and Definition 1135.6 bind the TasteGate-backed scoped kernel route and public continuation boundary as the bridge-facing handoff for the finite continuation, stability, failure-surface, ledger, and NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert a final induction oracle, total future fit, host physical truth, Lean verification, or bridge checking for $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$.

Upgrade path: No further paper-closure upgrade remains; Lean verification and bridge checking remain separate formal-axis work for the displayed packet rows.

Chapter 1136

A Concrete Naming Certificate for $\mathsf{RealityConstrainedExplanation}^{\uparrow}$

The $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ packet realizes the explanation discipline of section 9414.46 inside the reality-constrained tower vision of chapter 9414 and the final synthesis of section 9414.68 as a finite NameCert surface. It consumes the audit rows of chapter 1124, the physical-signature comparison rows of **??**, the $\mathsf{PhysicalFitTower}^{\uparrow}$ tower-descent rows, and the physical audit rows of chapter 1129. Those siblings supply model audit, signature fit, observation reflection, fit tower, and physical audit carriers; this packet binds them into the finite certificate by which an equation, derivation, model syntax, or effective-theory claim becomes explanatory only inside a displayed source, comparison, descent, residue, failure, route, provenance, and local naming boundary.

Definition 1136.1 (Reality-constrained explanation carrier)**.** *A* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (\Pi, M, P, S, D, R, F, H, C, Q, N)$$

with the following displayed rows:

1. Π *is the finite observation-bundle row through which the phenomenon is read;*
2. M *is the model row whose equations, derivations, mechanisms, or effective-theory operations are offered as explanatory;*
3. P *is the phenomenon row naming the reality-facing target of the explanation;*
4. S *is the signature-comparison row comparing the physical signature generated by* Π, P *with the model-facing signature generated by* Π, M*;*
5. D *is the tower-descent row recording which explanatory or predictive operations survive the approximation route that produced* M*;*
6. R *is the residue ledger naming unmodelled variables, hidden tower levels, idealized-away structure, measurement conditions, and physical properties not queried by* Π*;*
7. F *is the failure-surface row naming finite observations, refinements, or signature mismatches that would defeat the explanation;*
8. H *records componentwise* hsame *transport for the observation, model, phenomenon, signature, descent, residue, and failure rows;*

9. *C records the accepted consumer route through the displayed explanation rows;*
10. *Q records* Pkg *provenance over* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{PhysicalFitTower}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedExplanation}^{\uparrow}}$ *row.*

The carrier has no coordinate for bare equation reading, detached derivation, syntax realism, unscoped model truth, hidden idealization, or explanatory export outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedExplanationUp.RealityConstrainedExplanationTasteGate`
`single carrier alignment`

Theorem 1136.2 (Reality-constrained explanation NameCert obligations)**.** *For the carrier of Definition 1136.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedExplanation}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission: the accepted object is exactly $E = (\Pi, M, P, S, D, R, F, H, C, Q, N)$;*
2. *signature-comparison exactness: every explanation read must use the displayed comparison row S between the Π-indexed model signature and the Π-indexed phenomenon signature;*
3. *tower-descent: every explanatory operation exported from M must pass through the descent row D before it is read as a claim about P;*
4. *residue and failure-surface exposure: the residue ledger R and failure-surface row F are public before the explanation route is accepted;*
5. *ledger non-escape: no consumer may infer explanation from rows outside $\Pi, M, P, S, D, R, F, H, C, Q, N$.*

Proof. Carrier admission is projection from Definition 1136.1. The packet lists the only rows available to the local certificate, so no later consumer can insert an unobserved source, hidden model row, detached phenomenon row, or extra truth oracle.

Signature-comparison exactness is the distinguished role of S. The comparison is not a bare equation inside M and not a host judgment about P; it is the displayed relation between the two Π-indexed signatures.

Tower-descent is forced by D. An operation internal to M becomes explanatory for P only when the same packet records its survival through the approximation route.

Residue and failure exposure are forced by R and F. The packet accepts an explanation only with the unmodelled remainder and defeating observations visible to the consumer.

Ledger non-escape follows because the carrier contains no coordinate outside $\Pi, M, P, S, D, R, F, H, C, Q, N$. A route depending on any absent row is not a route in the accepted local naming certificate. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedExplanationUp.RealityConstrainedExplanationNamecertObligations`

Theorem 1136.3 (Reality-constrained explanation scope and residue)**.** *Let $E = (\Pi, M, P, S, D, R, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ *packet. A consumer of E cannot read bare equations, bare derivations, syntax realism, or unscoped model claims as explanations unless the same route displays the observation bundle, model row, phenomenon row, signature-comparison row, tower-descent row, residue ledger, failure surface, transport, consumer route, provenance, and local NameCert row.*

Proof. Start with the carrier definition. The only exported explanation packet is the finite tuple

$$\Pi, M, P, S, D, R, F, H, C, Q, N.$$

Bare equations and bare derivations use only syntax internal to M. They do not expose the phenomenon row P, the comparison row S, the descent row D, or the residue and failure rows R, F, so they do not project an explanation route from E.

Syntax realism asks the consumer to identify model syntax with reality-facing structure. The classifier surface instead requires the Π-indexed signature comparison and the displayed transport row H, so syntax alone has no accepted coordinate.

Unscoped model claims ignore the residue ledger and failure surface. By the NameCert obligations of Definition 1136.2, the accepted route must expose R and F before the explanation can be read.

Therefore a consumer can export an explanation only through the complete displayed packet. Any route that omits one of the listed rows is a partial mathematical, syntactic, or model-audit read, not a $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ explanation. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedExplanationUp.RealityConstrainedExplanationCarrier scope and residue`

Falsifiable prediction: A public explanation read that preserves the model syntax but changes either the physical-signature comparison row or the residue/failure-surface row is rejected by the carrier classifier.

Independence witness: realityconstrainedmodelaudit, realityconstrainedsignaturefit, observationreflectionpacket, physicalfittower, physicalmodelaudit: none is bijective to the explanation packet because each lacks at least one of phenomenon row, signature-comparison row, tower-descent row, residue ledger, or failure-surface row.

The Lean follow-up must provide `Nontrivial RealityConstrainedExplanationUp` and `FieldFaithful RealityConstrainedExplanationUp` instances in `lean4/BEDC/Derived/RealityConstrainedExplanationUp/Ta`

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedExplanation}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ packet is generated from an observation-bundle row, model row, phenomenon row, signature-comparison row, tower-descent row, residue ledger, failure surface, componentwise hsame transports, displayed Cont route, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.RealityConstrainedExplanationUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedExplanation}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1136.1, Definition 1136.2, Definition 1136.3, and the TasteGate row bind the finite explanation carrier, obligation surface, residue-scoped consumer route, and AI-discovery gate as the paper-side scope witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedExplanation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host model truth, unbounded validation, hidden-source recovery, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: No paper-side theory upgrade remains; bridge status and Lean verification are governed by their own axis.

Chapter 1137

A Concrete Naming Certificate for $\mathsf{OpenFitPacket}^{\uparrow}$

The $\mathsf{OpenFitPacket}^{\uparrow}$ carrier realizes the finite-observation open-fit discipline of section 9414.13 as a finite NameCert packet. It consumes the $\mathsf{PhysicalFitTower}^{\uparrow}$ physical signature tower, the classifier boundary of **??**, and the audit gate of chapter 1124. Those packets separately expose physical-fit towers, signature compatibility, and audit ledgers; $\mathsf{OpenFitPacket}^{\uparrow}$ binds the open-fit failure surface and no-final-oracle boundary into one finite observational carrier.

Definition 1137.1 (OpenFitPacket carrier)**.** *An* $\mathsf{OpenFitPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (H, \Pi, S, M, F, E, L, B, N)$$

with the following displayed rows:

1. H *is the finite observation trace row.*
2. Π *is the admissible observation-bundle row.*
3. S *is the observed-signature row extracted from* H *through* Π*.*
4. M *is the model-signature row compared against* S *at the displayed trace.*
5. F *is the finite fit row certifying the currently accepted signature comparison.*
6. E *is the failure-surface row naming finite signature comparisons that refute* F*.*
7. L *is the ledger row recording accepted fit, undecided continuation, failure evidence, and refused final-fit demands.*
8. B *is the boundary row forbidding export of a final-fit oracle from the finite packet.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{OpenFitPacket}^{\uparrow}}$ *row.*

The carrier has no coordinate for total confirmation, host identity between model and system, unobserved future signatures, or a global oracle deciding all admissible observation histories.

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.OpenFitPacketTasteGate single carrier alignment`

Definition 1137.2 (OpenFitPacket obligation surface)**.** *For the carrier of Definition 1137.1, the local* $\mathsf{NameCert}_{\mathsf{OpenFitPacket}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for* $P = (H, \Pi, S, M, F, E, L, B, N)$*;*

2. *finite signature exactness through S and M over the displayed trace H and bundle Π;*
3. *finite fit exactness through F;*
4. *finite refutation exposure through the failure-surface row E;*
5. *ledger exactness through L;*
6. *no-final-oracle boundary preservation through B;*
7. *local naming through N.*

Theorem 1137.3 (OpenFitPacket finite refutation surface)**.** *Every accepted* $\mathsf{OpenFitPacket}^{\uparrow}$ *packet is refuted locally by a finite failed signature comparison displayed in E. Conversely, a consumer route that does not display such a failed comparison has no refutation witness inside the packet and must leave continuation status ledgered as open rather than confirmed.*

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.OpenFitPacketFiniteRefutationSurface`

Proof. Carrier admission exposes exactly $P = (H, \Pi, S, M, F, E, L, B, N)$. The only row naming failed comparisons is E, and the only finite fit row that can be defeated is F.

When E displays a failed comparison between the observed signature S and the model signature M over the same H and Π, the packet contains a finite witness against the accepted fit row. This is a local refutation because all participating rows are part of the same finite carrier.

When no such failed comparison is displayed, the ledger row L still records the status as open. The boundary row B prevents the missing failure from being converted into final confirmation, so the packet remains a finite open-fit certificate rather than an oracle for all continuations. □

Theorem 1137.4 (OpenFitPacket no-final-oracle boundary)**.** *No accepted* $\mathsf{OpenFitPacket}^{\uparrow}$ *packet exports a final-fit oracle. Its public evidence is exhausted by finite observation, finite signature comparison, finite fit, finite refutation exposure, ledger status, boundary refusal, and local naming through $H, \Pi, S, M, F, E, L, B, N$.*

Proof. The carrier of Definition 1137.1 contains no row for total confirmation or global quantification over every admissible observation history. The obligation surface of Definition 1137.2 instead requires finite signature exactness, finite fit exactness, finite refutation exposure, ledger exactness, and preservation of the boundary row.

Thus a consumer may read the displayed finite fit, and it may read a finite refutation when E displays one. The same route must also read L and B, so undecided continuation remains ledgered and the final-fit demand is refused.

Any purported oracle for all future or counterfactual histories would need a coordinate outside $H, \Pi, S, M, F, E, L, B, N$. Since the NameCert surface admits only these rows, the purported oracle has no accepted route in the packet. □

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.OpenFitPacket no final oracle boundary`

Theorem 1137.5 (OpenFitPacket signature-fit handoff)**.** *An* $\mathsf{OpenFitPacket}^{\uparrow}$ *consumer can hand finite fit to the signature-fit and audit stack only through the displayed signature rows S, M, finite fit row F, failure-surface row E, ledger row L, and boundary row B. The handoff may consume* $\mathsf{PhysicalFitTower}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, *and* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *certificates, but it cannot erase the open-fit failure surface or the no-final-oracle boundary.*

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.OpenFitSignatureFitHandoff`

Proof. The physical-fit tower supplies the reality-facing tower context, the signature-fit packet supplies the comparison discipline, and the model-audit packet supplies the audit gate. The $\mathsf{OpenFitPacket}^{\uparrow}$ carrier reads those surfaces only through its own displayed rows.

The handoff therefore preserves S, M, F, E, L, B. Signature comparison travels through S and M, accepted finite fit through F, finite defeat through E, audit status through L, and the final-oracle refusal through B.

If a consumer route drops E or B, it no longer carries the open-fit failure surface or the no-final-oracle boundary. Such a route may still mention a physical tower, signature classifier, or audit gate, but it is not an accepted $\mathsf{OpenFitPacket}^{\uparrow}$ handoff. □

Theorem 1137.6 (OpenFitPacket bridge-facing surface)**.** *Every bridge-facing consumer of an accepted* $\mathsf{OpenFitPacket}^{\uparrow}$ *packet must read exactly the displayed rows* $H, \Pi, S, M, F, E, L, B, N$*. These rows preserve the finite observation trace, observation bundle, observed and model signature rows, finite fit row, failure surface, ledger row, no-final-oracle boundary, and local NameCert row. No bridge-facing route may erase the finite observation trace, bypass* E *or* B*, replace the ledger, or import a final-fit oracle from a physical tower, signature classifier, audit packet, host approval, or ambient model class.*

Proof. Start from the carrier and obligation surface. The packet has exactly the rows $H, \Pi, S, M, F, E, L, B, N$, and the local NameCert obligations assign finite observation, observation bundle, signature comparison, finite fit, failure exposure, ledger, boundary, and naming roles to those rows.

Use the finite refutation surface. It forces any failed comparison read to pass through E over the same S, M comparison surface, so a bridge-facing consumer cannot replace the failure surface while preserving packet acceptance.

Apply the no-final-oracle boundary and signature-fit handoff. The boundary keeps continuation status finite and refuses global confirmation, while the handoff routes signature fit through S, M, F, E, L, B.

Therefore the bridge-facing surface is exactly the displayed packet. A consumer missing one of $H, \Pi, S, M, F, E, L, B, N$ or importing an outside oracle is not reading the accepted $\mathsf{OpenFitPacket}^{\uparrow}$ bridge-facing row. □

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.OpenFitPacketUp StdBridge`

The Lean follow-up in `lean4/BEDC/Derived/OpenFitPacketUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{OpenFitPacket}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{OpenFitPacket}^{\uparrow}$ instances preserving $H, \Pi, S, M, F, E, L, B,$

Falsifiable prediction: A finite failed signature comparison refutes the packet locally, while no accepted packet exports a final-fit oracle.

Independence witness: physicalfittower, realityconstrainedsignaturefit, realityconstrainedmodelaudit, streamname, real: none contains the open-fit failure-surface and no-final-oracle rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{OpenFitPacket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OpenFitPacket}^{\uparrow}$ packet is generated from a finite observation trace, observation bundle, observed signature, model signature, finite fit row, failure surface, ledger, no-final-oracle boundary, and local NameCert row.

Lean4 ✓ `BEDC.Derived.OpenFitPacketUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpenFitPacket}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1137.1, Definition 1137.2, Definition 1137.3, Definition 1137.4, and Definition 1137.5 bind the finite carrier, obligation surface, finite refutation route, oracle refusal, and signature-fit handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpenFitPacket}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.OpenFitPacketUp.OpenFitPacketTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not assert bridge checking, host physical truth, model identity, total confirmation, or a global observation-history oracle.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 1138

A Concrete Naming Certificate for $\mathsf{RelationalObjectivity}^{\uparrow}$

The $\mathsf{RelationalObjectivity}^{\uparrow}$ packet realizes the objectivity-without-privileged-observer surface of section 9414.30 as a finite NameCert carrier. It consumes the finite inter-Hist family of chapter 474, the observer-history identity surface of chapter 479, the public other-minds commitment of chapter 499, the locality-cell evidence of chapter 509, and the public alignment packet of chapter 643. The packet is not a view from nowhere; it is the displayed finite route through which observer-family rows, invariant evidence, anchor-stability rows, ledgers, transports, package provenance, and local naming are read together.

Definition 1138.1 (RelationalObjectivity invariant-packet carrier)**.** *A* $\mathsf{RelationalObjectivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, I, A, L, T, P, N)$$

with the following displayed rows:

1. *F is the finite observer-family row, read through the* $\mathrm{MultiHistConfig}^{\uparrow}$ *packet and the relevant* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *rows;*
2. *I is the invariant-packet row, recording the inter-observer relation, the classifier under which the relation is compared, and the claim read from that invariant;*
3. *A is the anchor-stability row, recording that changing the source observer anchor preserves the invariant claim by the displayed classifier;*
4. *L is the ledger row, recording the observation, continuation, locality-cell, other-minds commitment, and cross-perspective alignment evidence used by the invariant read;*
5. *T is the componentwise transport row, storing the* hsame *and* Cont *routes that replay the invariant across observer-family coordinates;*
6. *P is the* Pkg *provenance over the vision objectivity surface and the five concrete sibling packets cited above;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a privileged observer, global synchronization frame, hidden observer substance, unledgered invariant, or anchor-free claim.

Lean4 (definition) `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityCarrier`

Definition 1138.2 (RelationalObjectivity observer-family source)**.** *An observer-family source for* $\mathsf{RelationalObjectivity}^{\uparrow}$ *is the row* F *of an accepted carrier* $R = (F, I, A, L, T, P, N)$ *together with its displayed routes to* $\mathrm{MultiHistConfig}^{\uparrow}$ *and* $\mathsf{ObserverHistIdentity}^{\uparrow}$*. A read of* F *gives only the finite family of observer histories and their public identity rows; it does not supply a distinguished evaluator of the family.*

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityObserverFamilySource`

Definition 1138.3 (RelationalObjectivity NameCert classifier)**.** *The local* $\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}$ *classifier compares two accepted carriers only through the displayed coordinates* F, I, A, L, T, P, N*. Two reads agree when their observer-family rows are matched by the same public identity and multi-Hist routes, their invariant-packet rows determine the same claim under the displayed classifier, their anchor-stability and ledger rows are transported by* T*, and their provenance and local naming rows agree.*

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityNameCertClassifier semantic certificate`
Lean4 (definition) `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityNameCertClassifier`

Theorem 1138.4 (RelationalObjectivity anchor-stability obligation)**.** *Every accepted* $\mathsf{RelationalObjectivity}^{\uparrow}$ *read of a claim factors through the anchor-stability row* A *of an accepted carrier* $R = (F, I, A, L, T, P, N)$*. A consumer may treat the claim as relationally objective only after the invariant-packet row* I *and the anchor-stability row* A *are displayed in the same finite packet.*

Proof. Project the accepted carrier through Definition 1138.1. The invariant-packet coordinate I names the relation, classifier, and claim, while A records stability under change of observer anchor.

The classifier of Definition 1138.3 compares carriers only through the displayed rows. Hence a read that lacks A has no coordinate from which anchor change can be checked.

The coordinate list also excludes an anchor-free claim and a privileged observer. Therefore the objective read is available only through the finite pair I, A transported by T and named by N. □

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityAnchorStabilityObligation`

Theorem 1138.5 (RelationalObjectivity invariant-ledger exactness)**.** *For every accepted* $\mathsf{RelationalObjectivity}^{\uparrow}$ *carrier* $R = (F, I, A, L, T, P, N)$*, the ledger row* L *is exactly the displayed evidence surface for the invariant read: observer-history identity, multi-Hist configuration, locality-cell evidence, other-minds commitment, cross-perspective alignment, continuation replay, and componentwise transport. No invariant claim may be accepted through an unledgered observation or hidden synchronization coordinate.*

Proof. Read L from the carrier definition. Its declared evidence is the observation, continuation, locality-cell, other-minds commitment, and cross-perspective alignment data used by the invariant read, with the observer family supplied by F and the replay route supplied by T.

The sibling packets cited in the chapter opening supply the public rows consumed by this ledger. They provide finite multi-Hist, observer-identity, other-minds, locality-cell, and cross-perspective surfaces, not ambient observer access.

Since the carrier explicitly has no coordinate for hidden observer substance, unledgered invariant evidence, global synchronization, or an anchor-free claim, ledger exactness is exhausted by the displayed rows F, I, A, L, T, P, N. □

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityInvariantLedgerExactness`

Theorem 1138.6 (RelationalObjectivity no-privileged-anchor certificate)**.** *Every accepted* $\mathsf{RelationalObjectivity}^{\uparrow}$ *packet certifies relational objectivity without selecting a privileged observer anchor. Changing the source observer route may transport the invariant through T and check stability through A, but it cannot create a global observer, a synchronized external frame, or a host-level equality between observer histories.*

Proof. Begin with the observer-family source of Definition 1138.2. The row F exposes only a finite family and its public identity routes, with no distinguished evaluator.

The anchor-stability obligation of Definition 1138.4 explains how a claim is read after anchor change: the invariant row I and stability row A must remain visible in the same finite packet and are transported by T.

Finally inspect the carrier exclusions in Definition 1138.1. The packet has no coordinate for a privileged observer, global synchronization frame, hidden observer substance, or anchor-free claim. Thus anchor transport remains an internal certificate route rather than a promotion to an external standpoint. □

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityNoPrivilegedAnchorCertificate`
Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityCarrier no privileged anchor certificate`

Theorem 1138.7 (RelationalObjectivity public NameCert export)**.** *The public export surface of an accepted* $\mathsf{RelationalObjectivity}^{\uparrow}$ *packet is exactly the observer-family source F, the local classifier over F, I, A, L, T, P, N, the anchor-stability row A, the invariant-ledger exactness row L, the no-privileged-anchor certificate, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}$ *row. A public consumer may cite this surface after the TasteGate carrier alignment has been read; it receives no hidden observer substance, privileged anchor, global synchronization coordinate, or anchor-free claim.*

Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.taste gate`

Proof. Project the carrier of Definition 1138.1. The visible public rows are F, I, A, L, T, P, N, and the classifier of Definition 1138.3 compares accepted packets only through those rows.

Use the observer-family source, anchor-stability obligation, invariant-ledger exactness theorem, and no-privileged-anchor certificate. These results expose the public source, the classifier-facing invariant, the anchor-change check, the ledger, and the refusal of a distinguished observer anchor.

The TasteGate carrier alignment supplies the Lean-side public handoff for the same displayed rows. The structural rows then transport by hsame, replay by Cont, source by Pkg, and name by the local NameCert row, so the exported surface is public without adding any hidden observer coordinate. □

Falsifiable prediction: A RelationalObjectivityUp read is refuted whenever it identifies two observer claims without a displayed invariant packet and anchor-stability row.

Independence witness: multihistconfig, observerhistidentity, othermindscommitment, locality-cell, crossperspectivealignment: none of these siblings contains the full observer-family, invariant-packet, anchor-stability, no-privileged-anchor, and local NameCert surface.

The Lean TasteGate surface for this packet is ‘lean4/BEDC/Derived/RelationalObjectivityUp/TasteGate.lean‘, where the handoff provides ‘Nontrivial RelationalObjectivityUp‘ and ‘FieldFaithful RelationalObjectivityUp‘ instances over the displayed carrier and classifier rows.
Lean4 ✓ `BEDC.Derived.RelationalObjectivityUp.RelationalObjectivityTasteGate single carrier alignment`

Theorem 1138.8 (RelationalObjectivity invariant-packet bridge surface)**.** *The public observer-family export of Definition 1138.7 is bridged to the invariant-packet carrier of Definition 1138.1 by the displayed observer-family row F, classifier row I, anchor-stability row A, invariant ledger L, componentwise transport row T,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}$ *row N. The bridge-facing surface is therefore the finite route through F, I, A, L, T, P, N and exports no privileged observer, global synchronization frame, hidden observer substance, unledgered invariant, or anchor-free claim.*

Proof. Project the public export theorem. Its observer-family source, local classifier, anchor-stability row, invariant-ledger exactness row, no-privileged-anchor certificate, componentwise transport, provenance, and local naming are exactly the displayed rows of the carrier.

Use the classifier and anchor-stability obligations. The classifier reads only F, I, A, L, T, P, N, while anchor stability keeps the invariant claim tied to I and A under the same transport row.

Apply invariant-ledger exactness and the no-privileged-anchor certificate. The ledger accounts for the observer-history identity, multi-Hist configuration, locality-cell, other-minds, cross-perspective alignment, continuation, and transport evidence, and the certificate refuses a distinguished observer anchor.

Thus the public export is bridged to the invariant-packet carrier by the displayed rows themselves. Since the carrier has no coordinate for a privileged observer, global synchronization, hidden observer substance, unledgered invariant, or anchor-free claim, the bridge-facing surface exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RelationalObjectivity}^{\uparrow}$ packet is generated from a finite observer-family source, invariant-packet row, anchor-stability row, invariant ledger, componentwise transport row, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1138.1, Definition 1138.2, Definition 1138.3, Definition 1138.4, Definition 1138.5, Definition 1138.6, Definition 1138.7, and Definition 1138.8 bind the public carrier, observer-family source, classifier, anchor-stability, ledger exactness, no-privileged-anchor certificate, TasteGate handoff, invariant-packet bridge surface, provenance, and local NameCert export over $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, and $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RelationalObjectivity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close privileged observers, global synchronization, hidden observer substance, anchor-free claims, Lean verification, or bridge checking.

Upgrade path: Verification closure requires a checked Lean target for the scoped observer-family, classifier, anchor-stability, ledger, transport, provenance, and local naming route.

Chapter 1139

A Concrete Naming Certificate for $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$

The $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ packet realizes the certificate-relative physical induction surface of section 9414.25 as a finite BEDC naming certificate. It is adjacent to the open-fit surface of chapter 1132, the audit packet of chapter 1129, and the inter-observer boundary of chapter 1133. Those siblings provide finite fit, audit, and objectivity rows; this seal binds finite fit to a continuation class, a stability basis, a failure surface, predictive reuse, transport, provenance, and local naming without exporting an unbounded future oracle.

Definition 1139.1 (PhysicalInductionStabilitySeal carrier)**.** *A* $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, F, C, S, L, R, T, P, N)$$

with the following displayed rows:

1. *H is the observation prefix row from which the finite physical fit is read;*
2. *F is the finite fit row comparing observed and model signatures over the displayed prefix;*
3. *C is the continuation-class row naming the admissible future histories for this seal;*
4. *S is the stability-basis row licensing transport from F to continuations in C;*
5. *L is the failure-surface row recording the continuations that separate observed and model signatures;*
6. *R is the predictive-reuse route from the finite fit row to the continuation-class read;*
7. *T records componentwise* hsame *transport for H, F, C, S, L, R;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, *and the sibling fit, audit, objectivity, and induction rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}}$ *row.*

The classifier accepts only packets exposing all nine rows. It has no coordinate for a final-fit oracle, unbounded physical necessity, hidden future history, host-law identity, or predictive reuse outside the displayed continuation class.

Lean4 (definition) `BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealUp`
Lean4 ✓ `BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealTasteGate`
`single carrier alignment`

Theorem 1139.2 (PhysicalInductionStabilitySeal NameCert obligations)**.** *For the carrier of Definition 1139.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier exposure for* $K = (H, F, C, S, L, R, T, P, N)$*;*
2. *classifier laws over the continuation class* C*;*
3. *stability transport from the finite fit row* F *through the stability basis* S *and transport row* T*;*
4. *failure-surface exactness through* L*;*
5. *predictive-reuse non-escape through the route* R*, provenance* P*, and local name* N*.*

Every accepted prediction from the seal must read these rows. No consumer may replace them by a final physical law, an ambient completeness claim, an unlisted continuation, or a host-level oracle over future observations.

Proof. Carrier exposure is immediate from the displayed packet. The rows $H, F, C, S, L, R, T, P, N$ are the complete inventory available to the local naming certificate, and each downstream read must begin from that finite tuple.

Next read the classifier rows over C. The continuation class is not a background universal domain; it is a coordinate of the accepted packet. Classifier laws therefore compare finite-fit and continuation reads only through the displayed continuation class and its componentwise transports.

Now transport stability. The finite fit row F is licensed for reuse only through the stability basis S and the transport row T. The predictive route R carries that transported read to admissible continuations, while P records the sibling fit, audit, and objectivity provenance that lets the route be consumed.

Finally inspect the failure surface. The row L records exactly the continuations where observed and model signatures separate, so removing L erases the falsifiable boundary of the seal. Since N names the whole packet and no other carrier coordinate is present, predictive reuse cannot escape to an unlisted continuation, hidden future history, host-law identity, or final-fit oracle. □

Lean4 ✓ `BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealNameCert obligations`

Theorem 1139.3 (PhysicalInductionStabilitySeal scoped route)**.** *Every scoped* $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ *consumer must read the finite observation prefix, fit row, continuation class, stability basis, failure surface, predictive-reuse route, componentwise* hsame *transport,* Pkg *provenance, and local* NameCert *row through the carrier of Definition 1139.1. The scoped route exports no unbounded oracle, hidden future history, final physical law, or predictive read outside the displayed continuation class.*

Proof. Project the accepted carrier $K = (H, F, C, S, L, R, T, P, N)$. Its displayed rows are the only rows available to a scoped consumer, so the observation prefix, finite fit, continuation class, stability basis, failure surface, predictive route, transport, provenance, and local name must be read by projection from this tuple.

Apply Definition 1139.2. It makes stability transport, failure-surface exactness, and predictive-reuse non-escape obligations of the same local NameCert surface.

The TasteGate-aligned carrier marker fixes the consumer-facing packet to this row inventory. Since no coordinate supplies an unbounded oracle, hidden future history, final physical law, or extra continuation source, the scoped route is exhausted by the displayed carrier. □

Lean4 ✓ BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealScopedRoute

Theorem 1139.4 (PhysicalInductionStabilitySeal public export)**.** *The public export of* PhysicalInductionStabilitySe *is exactly the finite route through observation prefix H, finite fit F, continuation class C, stability basis S, failure surface L, predictive-reuse route R, componentwise* hsame *transport T,* Pkg *provenance P, and local* NameCert *row N. A public consumer may read predictive reuse only through that displayed route and only over the named continuation class; it receives no unbounded future oracle, hidden continuation source, final physical law, or host-law identity.*

Proof. Project the carrier of Definition 1139.1. The tuple $K = (H, F, C, S, L, R, T, P, N)$ is the full public inventory, so the observation prefix, fit row, continuation class, stability basis, failure surface, predictive route, transport, provenance, and local name are all visible before reuse is consumed.

Apply the NameCert obligations of Definition 1139.2. They bind classifier laws to C, stability transport to F, S, T, failure-surface exactness to L, and predictive-reuse non-escape to R, P, N.

Finally use the scoped route of Definition 1139.3. The public export is the same finite route made object-facing: it can be cited by later physical-induction consumers, but it still refuses unbounded prediction, hidden future histories, final physical laws, and reuse outside the displayed continuation class. □

Lean4 ✓ BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealPublicExport

Theorem 1139.5 (PhysicalInductionStabilitySeal bridge route)**.** *The public export route of Definition 1139.4 is the bridge surface from* PhysicalInductionStabilitySeal$^{\uparrow}$ *to the abstract physical-induction interface. The route exposes exactly the rows*

$$H, F, C, S, L, R, T, P, N$$

as observation prefix, finite fit, continuation class, stability basis, failure surface, predictive reuse, componentwise transport, provenance, and local naming. A bridge consumer may cite those rows as the abstract physical-induction boundary, but it still receives no unbounded future oracle, hidden continuation source, final physical law, host-law identity, or predictive read outside the displayed continuation class.

Proof. Start with the public export theorem. It already presents $H, F, C, S, L, R, T, P, N$ as the complete consumer-facing route for finite physical induction.

The abstract interface reads the same rows by role: H supplies observations, F supplies fit, C supplies the continuation class, S supplies stability, L supplies the falsifiable boundary, R supplies predictive reuse, and T, P, N preserve, source, and locally name the packet.

No additional bridge coordinate is present. Therefore the route can be used as the bridge surface while preserving the same refusal of unbounded oracles, hidden continuations, final laws, host-law identity, and reuse outside the displayed class. □

The public export and bridge route give the complete finite example and completeness surface for the displayed continuation-class seal: every accepted use is routed through $H, F, C, S, L, R, T, P, N$, and no remaining paper obligation adds a stronger prediction interface.

Falsifiable prediction: Removing the failure-surface row or the continuation-class row makes predictive reuse indistinguishable from an unbounded oracle claim and invalidates the seal.

Independence witness: Nearest siblings: realityconstrainedopenfit, physicalmodelaudit, realityconstrainedobjectivity, observationreflectionhandoff, inscriptiongapledger. The seal has a continuation-class plus failure-surface pair none of those siblings has as a single carrier, while each sibling has audit, objectivity, handoff, or residue rows absent from this seal; no row-preserving bijection exists.

The Lean target family `BEDC.Derived.PhysicalInductionStabilitySealUp` must provide explicit Nontrivial $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ and FieldFaithful $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ instances in `Derived/PhysicalInductionStabilitySealUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ packet is generated from a finite observation prefix, finite fit, continuation class, stability basis, failure surface, predictive-reuse route, componentwise transport, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1139.1, Definition 1139.2, Definition 1139.3, Definition 1139.4, and Definition 1139.5 bind the finite certificate-relative physical induction packet, its local NameCert surface, scoped consumer route, public export, and bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PhysicalInductionStabilitySealUp.PhysicalInductionStabilitySealPublicExport`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close StdBridge checking, mature bridge completeness, unbounded future prediction, total physical law, host-law identity, or continuations outside the displayed class.

Upgrade path: The remaining formal-axis step is checked bridge interpretation for the displayed bridge route.

Chapter 1140

A Concrete Naming Certificate for $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$

The $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$ packet realizes the final synthesis of section 9414.68 as a finite NameCert seal rather than as an ambient truth predicate. It sits beside the audit carrier of chapter 1124, the open-fit carrier of chapter 1132, the objectivity carrier of chapter 1133, the observation-reflection packet of chapter 1126, and the explanation carrier of chapter 1136. Those sibling chapters expose model audit, open continuation, observer invariance, observation handoff, and explanatory descent; this seal binds them into the finite packet by which physical truth is read as constrained synthesis.

Definition 1140.1 (Reality-constrained synthesis seal carrier)**.** *A* $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, F, O, Q, E, T, L, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *row recording model-facing audit and admissible comparison data.*
2. *F is the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *row recording finite fit, open continuation, and finite refutation.*
3. *O is the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *row recording observer-anchor change and invariant public content.*
4. *Q is the* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *row recording observation handoff and signature readback.*
5. *E is the* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ *row recording explanatory descent, residue, and failure boundaries.*
6. *T is the tower-descent row recording which operations survive transport from source traces to explanatory packages.*
7. *L is the ledger row recording hidden residue, compression, idealization, provenance, and unresolved continuation.*
8. *R is the failure-surface row naming finite defeats, domain exits, and refused final-fit demands.*
9. *H records componentwise* hsame *transport for the audit, fit, objectivity, observation, explanation, tower, ledger, and failure rows.*

10. *C records displayed* Cont *replay through all rows of the seal.*

11. *P records* Pkg *provenance over the five sibling carriers,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for model–reality identity, unbounded final confirmation, private-observer truth, unledgered explanation, hidden physical equality, or export outside the displayed seal.

Lean4 (definition) `BEDC.Derived.RealityConstrainedSynthesisSealUp.taste gate`
Lean4 ✓ `BEDC.Derived.RealityConstrainedSynthesisSealUp.RealityConstrainedSynthesisSealTasteGate single carrier alignment`

Theorem 1140.2 (Reality-constrained synthesis seal NameCert obligations)**.** *For the carrier of Definition 1140.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (A, F, O, Q, E, T, L, R, H, C, P, N)$, audit exposure through A, open-fit exposure through F, observer-invariant exposure through O, observation-reflection exposure through Q, explanatory-descent exposure through E, tower transport through T, ledger exactness through L, failure-surface exposure through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission projects the finite packet of Definition 1140.1. No row outside $A, F, O, Q, E, T, L, R, H, C,$ can be read by the local certificate.

The five sibling rows supply the public synthesis surface. The audit row A constrains model-facing comparison, F keeps finite fit open to continuation and refutation, O prevents one observer history from becoming the public claim, Q exposes the observation handoff, and E records explanatory descent.

The tower, ledger, and failure rows bind the same surface to physical use. The tower row T licenses operations through the approximation route, L records hidden residue and provenance, and R names defeat conditions and refused finality.

Finally the transport, replay, provenance, and naming rows H, C, P, N preserve exactly the displayed seal. Thus the local NameCert surface is exhausted by the listed obligations. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedSynthesisSealUp.RealityConstrainedSynthesisSealUp namecert obligations`

Theorem 1140.3 (Reality-constrained synthesis seal non-escape)**.** *Every consumer of an accepted* $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$ *packet must read constrained synthesis through the displayed audit, open-fit, objectivity, observation-reflection, explanation, tower, ledger, failure-surface, transport, replay, provenance, and local-name rows. No consumer may replace that route by model–reality identity, a final-fit oracle, a private observer anchor, an unledgered explanation, host equality, or a bare model row.*

Proof. Use the NameCert obligations of Definition 1140.2. They force every accepted route to expose $A, F, O, Q, E, T, L, R, H, C, P, N$ before constrained synthesis is read.

The open-fit, objectivity, and observation-reflection rows prevent the familiar escape routes. The fit row F is finite and defeasible, the objectivity row O requires an invariant anchor-change route, and the observation row Q keeps the claim tied to displayed readback.

The explanation, tower, ledger, and failure rows close the remaining routes. A consumer that omits E, T, L, R loses descent, hidden residue, and defeat information, so it no longer reads the accepted seal.

Therefore a synthesis read can only be the displayed route through the full carrier. The excluded identity, finality, private-anchor, unledgered, host-equality, and bare-model routes have no row from which to be exported. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedSynthesisSealUp.RealityConstrainedSynthesisSealUp nonescape`

Theorem 1140.4 (Reality-constrained synthesis seal independence boundary)**.** *The* RealityConstrainedSynthesisSea *carrier is independent of each single sibling surface named by* RealityConstrainedModelAudit$^{\uparrow}$, RealityConstrainedOpe RealityConstrainedObjectivity$^{\uparrow}$, ObservationReflectionPacket$^{\uparrow}$, *and* RealityConstrainedExplanation$^{\uparrow}$. *Each sibling can supply one required row of the seal, but none of them alone supplies the full packet* $S = (A, F, O, Q, E, T, L, R, H, C, P, N)$ *or the non-escape boundary of Definition 1140.3.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedSynthesisSealUp.RealityConstrainedSynthesisSealUp independence boundary`

Proof. Read the carrier definition componentwise. The audit sibling supplies A, the open-fit sibling supplies F, the objectivity sibling supplies O, the observation-reflection sibling supplies Q, and the explanation sibling supplies E.

None of these rows contains the other four sibling rows together with the tower, ledger, failure, transport, replay, provenance, and local-name rows. A projection to any one sibling therefore forgets part of $S = (A, F, O, Q, E, T, L, R, H, C, P, N)$.

The non-escape theorem depends on the full displayed packet. Since each single sibling lacks at least one of the audit, open-fit, objectivity, observation, explanation, tower, ledger, or failure coordinates, no single sibling can be bijective with the seal. □

Falsifiable prediction: A constrained-synthesis consumer that exports physical truth while omitting any one of audit, open fit, observer invariance, observation reflection, explanation descent, ledger, or failure surface refutes the seal.

Independence witness: realityconstrainedmodelaudit, realityconstrainedopenfit, realityconstrainedobjectivity, observationreflectionpacket, realityconstrainedexplanation: none is bijective because RealityConstrainedSynthesisSeal$^{\uparrow}$ carries all five sibling surfaces together with tower, ledger, failure, transport, replay, provenance, and local NameCert rows in one finite packet.

Closure status for NameCert$_{\text{RealityConstrainedSynthesisSeal}^{\uparrow}}$ **·**

Constructive story: A RealityConstrainedSynthesisSeal$^{\uparrow}$ packet is generated from audit, open-fit, objectivity, observation-reflection, explanation, tower, ledger, failure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedSynthesisSeal}^{\uparrow}}$, maturePackageClosed)

Lean4 ✓ `BEDC.Derived.RealityConstrainedSynthesisSealUp.taste gate`

Scope closed: Definition 1140.1, Definition 1140.2, Definition 1140.3, Definition 1140.4, the falsifiable prediction, the independence witness, and the TasteGate marker record the bridge-facing paper route for the displayed constrained-synthesis seal.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedSynthesisSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert Lean verification, bridge checking, final physical truth, model–reality identity, a private observer oracle, unbounded observation, or unledgered explanation.

Upgrade path: No theory-axis upgrade remains; future work is verification and bridge synchronization for the displayed constrained-synthesis seal packet.

Chapter 1141

A Concrete Naming Certificate for $\mathsf{PhysicalTruthCertificate}^{\uparrow}$

The $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ packet realizes the final physical-truth synthesis of section 9414.68 as one finite NameCert carrier. It sits beside the reality-constrained open-fit surface of chapter 1132, the reality-constrained objectivity surface of chapter 1133, and the physical model-audit surface of chapter 1129. Those sibling chapters expose open continuation, observer-invariant reading, and audit rows; this carrier binds them with tower descent, source fit, ledger, failure surface, transport, route replay, provenance, and local naming before any consumer reads a physical truth claim.

Definition 1141.1 (Physical truth certificate carrier)**.** *A* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, F, O, D, I, L, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-trace and observation-bundle row from which the claim is read;*
2. *F is the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *row exposing finite fit, open continuation, and finite refutation;*
3. *O is the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *row exposing observer-anchor change and invariant content;*
4. *D is the tower-descent row recording which operations survive the approximation route;*
5. *I is the stability row naming the continuation domain over which the claim may be reused;*
6. *L is the ledger row recording hidden tower levels, compression, idealization, provenance, and residue;*
7. *R is the failure-surface row naming finite defeating observations, signature mismatches, or domain exits;*
8. *H records componentwise* hsame *transport for the source, fit, objectivity, descent, stability, ledger, failure, route, provenance, and local-name rows;*
9. *C records displayed* Cont *replay through all rows of T;*
10. *P records* Pkg *provenance over* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, RealityConstrainedObjectivi $\mathsf{PhysicalModelAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalTruthCertificate}^{\uparrow}}$ *row.*
The carrier has no coordinate for model–reality identity, final confirmation, unledgered idealization, detached observer certainty, hidden truth oracle, host physical equality, or truth export outside the displayed packet.

Lean4 (definition) `BEDC.Derived.PhysicalTruthCertificateUp.PhysicalTruthCertificateCarrier`

Theorem 1141.2 (Physical truth certificate NameCert obligations)**.** *For the carrier of Definition 1141.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalTruthCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (S, F, O, D, I, L, R, H, C, P, N)$, open-fit exposure through F, observer-invariance exposure through O, tower-descent exposure through D, stability exposure through I, ledger exactness through L, failure-surface exposure through R, componentwise transport through H, displayed route replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.PhysicalTruthCertificateUp.PhysicalTruthCertificateNameCertObligations`

Proof. Carrier admission projects the finite packet of Definition 1141.1. No row outside $S, F, O, D, I, L, R, H, C, P, N$ can be read by the local certificate.

Open-fit exposure is the role of F. It imports finite fit, open continuation, and finite refutation only as a displayed sibling row, so the certificate cannot identify finite support with final confirmation.

Observer-invariance exposure is the role of O. It requires the invariant and anchor-change route before the physical truth claim can be public, so one observer history cannot stand in for the full packet.

Descent, stability, ledger, and failure exposure are the remaining content rows. The descent row D licenses operations through the tower, I names the reuse domain, L records hidden residue and provenance, and R names defeat conditions.

The transport, continuation, provenance, and naming rows H, C, P, N then preserve exactly the displayed packet. Thus the local $\mathsf{NameCert}$ surface is exhausted by the listed obligations. □

Theorem 1141.3 (Physical truth certificate open-fit non-escape)**.** *Every consumer of an accepted* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ *packet must read physical truth through the displayed source row, open-fit row, observer-invariance row, descent row, stability row, ledger row, failure-surface row, transport row, route row, provenance row, and local name. No consumer may replace that route by model–reality identity, a final-fit oracle, unledgered continuation, host equality, or a bare mathematical model row.*

Proof. Use the NameCert obligations of Definition 1141.2. They force every accepted route to expose $S, F, O, D, I, L, R, H, C, P, N$ before a physical truth claim is read.

The open-fit row F is finite and defeasible. It supplies observed fit, continuation, and refutation rows, but it has no coordinate for final fit, model identity, or host equality.

The ledger and failure rows L and R close the remaining escape routes. A consumer that omits them loses the record of compression, idealization, hidden residue, and possible defeat, so it no longer reads the accepted packet.

Therefore a physical truth read can only be the displayed route through the full certificate carrier. The excluded identity, finality, unledgered, host-equality, and bare-model routes have no row from which to be exported. □

Lean4 ✓ `BEDC.Derived.PhysicalTruthCertificateUp.PhysicalTruthCertificate open fit nonescape`

Theorem 1141.4 (Physical truth certificate observer invariance)**.** *Let $T = (S, F, O, D, I, L, R, H, C, P, N)$ be an accepted* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ *carrier. If a consumer exports a physical truth claim from T, then the claim is read through the observer-invariance row O and the componentwise transport row H before the route C and local name N are exposed. A claim that depends on one observer history without the displayed O, H, C, N route is not a* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ *export.*

Lean4 ✓ `BEDC.Derived.PhysicalTruthCertificateUp.physicalTruthCertificate observer route fields`

Proof. Project the carrier. The only observer-facing coordinate is O, and the only transport coordinate for comparing the rows across accepted presentations is H.

Now read the route and naming rows. The continuation route C and local name N can replay and name only the rows already displayed by the carrier; they cannot create observer invariance after the fact.

If a consumer depends on one observer history alone, then it omits the invariant packet and anchor-change discipline carried by O. The route may still be local evidence, open fit, or model audit material, but it is not an export of the physical truth certificate. □

Falsifiable prediction: If a certified physical truth consumer reads model–reality identity without the displayed open-fit, ledger, observer-invariance, and failure-surface rows, the packet is refuted.

Independence witness: physical-fit-tower, reality-constrained-signature-fit, reality-constrained-open-fit, reality-constrained-objectivity, physical-model-audit: no sibling carries the full TruthCert packet with fit, tower, descent, observer invariance, ledger, and failure surface as one carrier.

Theorem 1141.5 (Physical truth certificate bridge interface)**.** *Every public* $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ *TruthCert packet bridges to consumers only through the displayed route*

$$S, F, O, D, I, L, R, H, C, P, N.$$

The bridge interface may expose source trace, open fit, observer invariance, tower descent, stability, ledger, failure surface, transport, replay, provenance, and local naming. It refuses model–reality identity, completed physical truth, unledgered oracle export, hidden observer certainty, host equality, and any truth row outside the displayed packet.

Proof. Project the carrier of Definition 1141.1. The only public coordinates are $S, F, O, D, I, L, R, H, C, P, N$, and the carrier explicitly omits model–reality identity, detached certainty, host equality, and hidden truth-oracle rows.

Apply the NameCert obligations of Definition 1141.2. They make open-fit exposure, observer-invariance exposure, tower descent, stability, ledger exactness, failure exposure, transport, replay, provenance, and local naming the complete public surface.

Use the open-fit non-escape and observer-invariance theorems, Definition 1141.3 and Definition 1141.4. They prevent a consumer from replacing the displayed route by final fit, a single observer history, or a bare mathematical model row.

Thus the bridge interface is exactly the finite TruthCert route through $S, F, O, D, I, L, R, H, C, P, N$, with all identity, completed-truth, unledgered-oracle, hidden-certainty, and host-equality exports refused. □

Lean4 ✓ `BEDC.Derived.PhysicalTruthCertificateUp.PhysicalTruthCertificateBridgeInterface`

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalTruthCertificate}^{\uparrow}}$.

Constructive story: A PhysicalTruthCertificate$^{\uparrow}$ packet is generated from finite source traces, reality-constrained open fit, observer invariance, tower descent, stability, ledger, failure surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lean4 ✓ `BEDC.Derived.PhysicalTruthCertificateUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{PhysicalTruthCertificate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1141.1, Definition 1141.2, Definition 1141.3, Definition 1141.4, and Definition 1141.5 bind the carrier, NameCert obligations, open-fit non-escape, observer-invariance route, bridge interface, falsifiable prediction, independence witness, and TasteGate marker to the displayed physical-truth packet.

Formal verification: FormalStatus(NameCert$_{\text{PhysicalTruthCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, final physical truth, host model–reality equality, unbounded observation, or a total observer-independent oracle.

Upgrade path: The paper-side package is mature; Lean verification and checked bridge evidence remain the next formal-axis obligations for the public TruthCert interface.

Chapter 1142

A Concrete Naming Certificate for $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$

The $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ packet realizes the ledgered tower-compression discipline of section 9414.20 as a finite BEDC NameCert carrier. It sits beside the indexed approximation carrier of chapter 1127, the $\mathsf{ObserverApproximationTower}^{\uparrow}$ observer-facing tower carrier, the signature-fit carrier of **??**, and the model-audit carrier of chapter 1124. The $\mathsf{PhysicalFitTower}^{\uparrow}$ role supplies the source-facing descent route; this chapter packages the composite ledger that prevents endpoint/source collapse across the tower.

Definition 1142.1 (Reality-constrained tower-compression carrier)**.** *A* $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, O, F, A, M, C, L, E, R, P, N)$$

with the following displayed rows:

1. *S is the source row, naming the reality-facing system, trace family, or observation process at the base of the tower;*
2. *T is the indexed tower row, recording the finite approximation levels and their compression arrows;*
3. *O is the observer tower row, recording the admissible observer-side readback and budgeted observation windows;*
4. *F is the physical-fit tower row, exposing the source-facing signature and descent discipline;*
5. *A is the reality-constrained approximation row, naming the tower carrier whose mathematical endpoint is being compressed;*
6. *M is the model-audit row, recording which model-facing fit claims are audited against the tower;*
7. *C is the classifier row comparing endpoint packages only through the declared physical signatures and observer budgets;*
8. *L is the composite-ledger row, recording the hidden residue of every compression arrow after transport to the endpoint;*
9. *E is the endpoint row, naming the visible terminal package without identifying it with S;*
10. *R is the replay row, recording the displayed* Cont *route through $S, T, O, F, A, M, C, L, E$;*

11. *P is the* Pkg *provenance over the sibling carriers,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}$ *row.*

The carrier has no coordinate for source/endpoint identity, unledgered quotient collapse, hidden physical truth, or endpoint-only model validation.

Lean4 ✓ BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionTasteGate single carrier alignment

Theorem 1142.2 (Reality-constrained tower-compression NameCert obligations)**.** *For the carrier of Definition 1142.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}$ *surface consists exactly of seven obligation rows:*

1. *carrier admission for $K = (S, T, O, F, A, M, C, L, E, R, P, N)$;*
2. *source and tower exposure through S and T;*
3. *observer and physical-fit exactness through O and F;*
4. *approximation and audit compatibility through A and M;*
5. *classifier accountability through C;*
6. *composite-ledger and endpoint exposure through L and E;*
7. *replay, provenance, and local naming closure through R, P, N.*

Every accepted read of the compressed endpoint must display these rows before a consumer may treat the endpoint as a reality-facing modelling claim.

Lean4 ✓ BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionNameCert obligations

Proof. Project the carrier of Definition 1142.1. The admitted object is the twelve-row packet $K = (S, T, O, F, A, M, C, L, E, R, P, N)$, so the source, tower, observer, physical-fit, approximation, audit, classifier, ledger, endpoint, replay, provenance, and local-name coordinates are all visible.

The source and tower rows bind the compression chain to S and T. The observer and physical-fit rows then restrict which signatures can be read from that chain, and the approximation and audit rows specify which endpoint packages may be compared.

Classifier accountability belongs to C alone. It compares endpoint packages only through the displayed observer and physical-fit rows, so it cannot convert mathematical endpoint equality into source identity.

Finally the ledger and endpoint rows expose the terminal package together with the transported hidden residue of every compression arrow. Replay, provenance, and local naming close the same packet, giving exactly the seven displayed obligation rows. □

Theorem 1142.3 (Reality-constrained tower-compression ledger non-escape)**.** *Let*

$$K = (S, T, O, F, A, M, C, L, E, R, P, N)$$

be an accepted $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ *packet. Any consumer that exports an endpoint claim from E must read it through the composite ledger L, the classifier C, and the source-facing rows S, T, O, F, A, M. A route that omits L or treats E as source identity is not a* RealityConstrainedTowerCompressio *export.*

Proof. Use the NameCert obligations of Definition 1142.2. They require every accepted read to expose the source, tower, observer, physical-fit, approximation, audit, classifier, composite-ledger, endpoint, replay, provenance, and naming rows.

The composite ledger row L records the hidden residue of each compression arrow after transport to the endpoint. A consumer route that omits L therefore lacks the row that distinguishes terminal visibility from the source material hidden along the tower.

The endpoint row E names only the visible terminal package. Since the carrier has no source/endpoint identity coordinate, any route that treats E as S bypasses the displayed classifier and ledger and is rejected by the local NameCert boundary. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionLedgernonescape`

Definition 1142.4 (Reality-constrained tower-compression scoped packet)**.** *The scoped* RealityConstrainedTowerCo *packet is the same finite carrier*

$$K = (S, T, O, F, A, M, C, L, E, R, P, N)$$

read over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, with sibling anchors in chapter 1127, the* ObserverApproximationTower$^{\uparrow}$ *role,* **??***, chapter 1124, and the* PhysicalFitTower$^{\uparrow}$ *role. Its scoped read admits endpoint claims only through the displayed classifier, composite ledger, replay, provenance, and local naming rows.*

Theorem 1142.5 (Reality-constrained tower-compression classifier accountability)**.** *For the scoped packet of Definition 1142.4, every classifier read through C is accountable to the displayed source, tower, observer, physical-fit, approximation, audit, composite-ledger, endpoint, replay, provenance, and local naming rows. In particular, C cannot compare endpoint packages without exposing $S, T, O, F, A, M, L, E, R, P, N$ inside the same* BHist *packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionClassifieraccountability`

Proof. Project the carrier of Definition 1142.1. The classifier coordinate C occurs only inside the twelve-row packet $K = (S, T, O, F, A, M, C, L, E, R, P, N)$.

Apply Definition 1142.2. The local certificate requires carrier admission, source and tower exposure, observer and physical-fit exactness, approximation and audit compatibility, classifier accountability, composite-ledger and endpoint exposure, and replay/provenance/name closure.

Thus C is accountable to exactly the displayed packet rows. The sibling approximation-tower, observer-tower, signature-fit, model-audit, and physical-fit chapters supply the surfaces read by A, O, F, M and the source-facing tower route, but none supplies a hidden endpoint classifier outside K. □

Theorem 1142.6 (Reality-constrained tower-compression endpoint-source separation)**.** *In every scoped* RealityConstrainedTowerCompression$^{\uparrow}$ *packet, the endpoint row E is separated from the source row S by the composite ledger L, classifier C, replay R, provenance P, and local name N. A consumer route that identifies E with S without those rows is outside the chapter's accepted export surface.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionEndpointsource separation`

Proof. Start with Definition 1142.3. It requires every endpoint export to read through the composite ledger L, classifier C, and source-facing rows S, T, O, F, A, M.

Use the scoped packet definition to keep the route inside BHist, hsame, Cont, Pkg, and NameCert. The replay row R, package row P, and local name N bind the endpoint read to the same finite carrier.

Since the carrier has no source/endpoint identity coordinate, the endpoint E cannot be collapsed to S by omission. The required ledger and replay rows are precisely the separator between terminal visibility and source identity. □

Theorem 1142.7 (Reality-constrained tower-compression scoped dependency surface)**.** *The scoped dependency surface for* RealityConstrainedTowerCompression$^{\uparrow}$ *is exactly the finite BEDC packet over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *together with the sibling approximation-tower, observer-tower, signature-fit, model-audit, and physical-fit chapters named in Definition 1142.4. No endpoint claim is exported from this surface without the classifier accountability of Definition 1142.5 and the endpoint/source separation of Definition 1142.6.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionScoped dependency surface`

Proof. First project the scoped packet. Its rows are $S, T, O, F, A, M, C, L, E, R, P, N$, and its ambient objects are BHist, hsame, Cont, Pkg, and NameCert.

Next apply Definition 1142.2. Those obligations make carrier admission, row exposure, classifier accountability, ledger exposure, endpoint exposure, replay, provenance, and local naming mandatory.

Finally combine Definition 1142.5 with Definition 1142.6. The first prevents C from floating free of the sibling rows; the second prevents E from being identified with S without the ledger and replay surface. Hence the displayed dependencies are exactly the scoped export surface. □

Theorem 1142.8 (Reality-constrained tower-compression scoped source exactness)**.** *Every accepted* RealityConstrainedTowerCompression$^{\uparrow}$ *packet reads its composite tower, open-fit, objectivity, approximation, methodology, compression, transport, continuation, provenance, and local naming rows through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. No hidden model-selection oracle, truth primitive, or external adequacy source is exported from that scoped surface.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionScoped source exactness`

Proof. Begin with the carrier and scoped packet. The carrier of Definition 1142.1 supplies the twelve displayed rows $S, T, O, F, A, M, C, L, E, R, P, N$, and Definition 1142.4 reads them only over BHist, hsame, Cont, Pkg, and NameCert support.

Apply Definition 1142.2. It accounts for source, tower, observer, physical-fit, approximation, audit, classifier, composite-ledger, endpoint, replay, provenance, and local naming rows before any consumer can read the compressed tower.

Next use Definition 1142.3 and Definition 1142.5. The ledger theorem prevents endpoint exports that omit L, while classifier accountability keeps C attached to the same source, tower, open-fit, objectivity, approximation, methodology, compression, transport, continuation, provenance, and naming rows.

Finally combine Definition 1142.6 with Definition 1142.7. The sibling chapters named by the scoped packet provide only the displayed constituent surfaces; none supplies a hidden model-selection oracle, truth primitive, or external adequacy source. Thus the accepted packet is exact at the scoped source surface. □

Theorem 1142.9 (Reality-constrained tower-compression downstream route exhaustion)**.** *Every downstream consumer of the compressed endpoint of an accepted* $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ *packet must read through the source row, indexed tower row, observer readback row, physical-fit row, approximation row, model-audit row, classifier row, composite-ledger row, endpoint row, replay row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}$ *row. The route cannot be replaced by endpoint-only model validation, hidden physical truth, source/endpoint identity, or an unledgered quotient collapse.*

Proof. Start from the NameCert obligations. They enumerate exactly the rows $S, T, O, F, A, M, C, L, E, R, P, N$ and require each accepted endpoint-facing read to expose source, indexed tower, observer readback, physical fit, approximation, audit, classifier, ledger, endpoint, replay, provenance, and local naming data.

Next apply ledger non-escape and classifier accountability. The composite ledger L keeps the endpoint from becoming source identity, while the classifier row C remains accountable to the displayed observer, physical-fit, approximation, audit, ledger, endpoint, replay, provenance, and naming surface.

Endpoint-source separation and scoped source exactness complete the route. The former prevents E from replacing S, and the latter keeps every tower read inside BHist, hsame, Cont, Pkg, and NameCert support.

Thus any downstream consumer that omits one of the displayed rows is outside the accepted route. Endpoint-only model validation and hidden physical truth require coordinates absent from the carrier, so they cannot exhaust the downstream route. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.RealityConstrainedTowerCompressionDownstreamR`

The Lean development in `lean4/BEDC/Derived/RealityConstrainedTowerCompressionUp/TasteGate.lean` must provide Nontrivial $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ and FieldFaithful $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ instances preserving $S, T, O, F, A, M, C, L, E, R, P, N$.

Falsifiable prediction: An accepted RealityConstrainedTowerCompressionUp endpoint claim that omits the composite ledger or identifies the endpoint with the source refutes the carrier.

Independence witness: realityconstrainedapproximationtower, observerapproximationtower, realityconstrainedsignaturefit, realityconstrainedmodelaudit, physicalfittower: none is bijective because $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ carries source, indexed tower, observer readback, physical fit, approximation, audit, classifier, composite-ledger, endpoint, replay, provenance, and local NameCert rows in one packet. Each listed sibling supplies a constituent surface, while only this packet binds composite compression ledgers to endpoint/source non-collapse.

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ packet is generated from source, tower, observer, physical-fit, approximation, audit, classifier, composite-ledger, endpoint, replay, provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTowerCompressionUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1142.4, Definition 1142.5, Definition 1142.6, Definition 1142.7, and Definition 1142.8 bind the bridge-facing paper surface through classifier accountability, endpoint-source separation, scoped dependency surface, scoped source exactness, the falsifiable prediction, and the independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedTowerCompression}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert Lean verification, bridge checking, source/endpoint identity, unledgered quotient collapse, hidden physical truth, or endpoint-only model validation.

Upgrade path: No further paper-side theory grade remains; Lean verification and bridge checking remain separately reported on their own axes.

Chapter 1143

A Concrete Naming Certificate for $\mathsf{TowerEquivalence}^{\uparrow}$

The $\mathsf{TowerEquivalence}^{\uparrow}$ packet realizes the tower-equivalence boundary of section 9414.20 as a finite BEDC NameCert carrier. It consumes the tower packet of chapter 1128, the indexed approximation tower of chapter 1127, the $\mathsf{PhysicalFitTower}^{\uparrow}$ physical-fit descent surface, the open-fit refutation surface of chapter 1132, and the observer-invariant surface of chapter 1133. Those siblings supply tower carriers, indexed approximation rows, physical descent data, open refutation, and observer-stability rows; this packet binds endpoint agreement to ledger and descent compatibility before any consumer treats two towers as equivalent.

Definition 1143.1 (TowerEquivalence carrier)**.** *A* $\mathsf{TowerEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, T', A, A', P, P', O, O', B, L, D, R, H, Q, N)$$

with the following displayed rows:

1. *T and T' are the two* $\mathsf{RealityConstrainedTowerPacket}^{\uparrow}$ *rows whose endpoint packages are being compared.*
2. *A and A' are the* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *rows exposing the indexed approximation levels of the two towers.*
3. *P and P' are the* $\mathsf{PhysicalFitTower}^{\uparrow}$ *rows recording source-facing signatures, hidden-source ledgers, and descent policies.*
4. *O and O' are the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *rows recording open-continuation and finite-refutation surfaces for the two endpoint reads.*
5. *B is the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *row that transports the comparison across the admitted observer anchors without selecting a global observer.*
6. *L is the paired ledger row. It records source-history agreement, hidden-residue agreement, omitted-row exposure, and every open adequacy gap that remains visible after endpoint comparison.*
7. *D is the descent-compatibility row. It lists exactly which operations descend through both towers and records refusal for any operation lacking a paired displayed route.*
8. *R is the endpoint-agreement row, comparing the terminal packages only under the displayed classifier bundle.*

9. *H records componentwise* hsame *transport for $T, T', A, A', P, P', O, O', B, L, D, R, Q, N$.*
10. *Q is the* Pkg *provenance over the sibling carriers,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TowerEquivalence}^{\uparrow}}$ *row.*

The carrier has no coordinate for endpoint-only equivalence, no hidden route from terminal agreement to source identity, no descent outside D, and no observer-stable claim outside B, L, H.

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalenceTasteGate single carrier alignment`

Theorem 1143.2 (TowerEquivalence NameCert obligations)**.** *For the carrier of Definition 1143.1, the local* $\mathsf{NameCert}_{\mathsf{TowerEquivalence}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for $E = (T, T', A, A', P, P', O, O', B, L, D, R, H, Q, N)$;*
2. *endpoint agreement through R together with the two tower-packet rows T, T';*
3. *ledger compatibility through L together with the approximation rows A, A', physical-fit rows P, P', and open-fit rows O, O';*
4. *descent compatibility through D and observer-stable transport through B, H;*
5. *provenance and local naming closure through Q, N.*

Every accepted tower-equivalence read must display these five rows before exporting equivalence between the two towers.

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalence namecert obligations`

Proof. Project the carrier of Definition 1143.1. The admitted packet lists the two tower rows, the two approximation rows, the two physical-fit rows, the two open-fit rows, the objectivity row, the paired ledger, the descent row, endpoint agreement, transport, provenance, and local naming.

Endpoint agreement is therefore only one coordinate of the packet. The local certificate reads it through R together with T and T', so terminal package agreement cannot by itself certify the two towers.

Ledger compatibility is read next. The rows A, A', P, P', O, O' expose indexed approximation, physical signature, descent policy, open continuation, and finite refutation data on both sides, while L records source-history and hidden-residue agreement across those rows.

Descent and observer stability close the comparison. The row D lists exactly the operations that descend through both towers, and B, H preserve the comparison across observer anchors and componentwise transport. The provenance and local name rows then name the same displayed packet, giving exactly the five listed obligations. □

Theorem 1143.3 (TowerEquivalence endpoint refusal)**.** *Let*

$$E = (T, T', A, A', P, P', O, O', B, L, D, R, H, Q, N)$$

be an accepted $\mathsf{TowerEquivalence}^{\uparrow}$ *packet. Endpoint agreement through R does not by itself export tower equivalence. A consumer route is accepted only when the paired ledger L, descent row D, objectivity row B, and componentwise transport row H travel with the same endpoint comparison.*

Proof. Read the NameCert obligations of Definition 1143.2. The endpoint row R appears only as one row of a five-row local surface.

The paired ledger L records the source histories, hidden residues, omitted rows, and open adequacy gaps that endpoint comparison may hide. Removing L leaves no displayed evidence that the two towers agree below the terminal package.

The descent row D and objectivity row B are also mandatory. Without D, an operation may descend through one tower but not the other; without B, H, the comparison may depend on a privileged observer anchor or an untransported component. Thus endpoint agreement alone is refused as tower equivalence. □

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalenceCarrier endpoint refusal`

Theorem 1143.4 (TowerEquivalence ledger-descent stability)**.** *Every accepted* $\mathsf{TowerEquivalence}^{\uparrow}$ *transport preserves ledger and descent compatibility componentwise. If H transports E to another accepted carrier, then the transported paired ledger still records the same source-history, hidden-residue, omitted-row, and adequacy-gap comparison, and the transported descent row still refuses exactly the operations not displayed on both towers.*

Proof. Use the carrier definition. The transport row H is componentwise over the tower rows, approximation rows, physical-fit rows, open-fit rows, objectivity row, ledger, descent row, endpoint row, provenance, and local name.

The ledger row L is one of those transported components, so source-history agreement, hidden-residue agreement, omitted-row exposure, and open adequacy gaps cannot be dropped by transport. They remain attached to the same comparison after transport.

The descent row D is transported in the same packet. Since D lists only paired displayed descent routes, any operation missing on either side remains outside the accepted transported descent row. □

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalence ledger descent stability`

Theorem 1143.5 (TowerEquivalence refinement separation)**.** *Two* $\mathsf{TowerEquivalence}^{\uparrow}$ *candidates may agree at the endpoint row R and still be separated by a refinement that exposes a difference in L or D. Such a separation does not contradict endpoint agreement; it shows that the endpoint comparison was not a complete tower-equivalence certificate.*

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalenceRefinementSeparation`

Proof. Start with Definition 1143.3. Endpoint agreement is not the accepted certificate unless the paired ledger, descent row, objectivity row, and transport row are present in the same packet.

A refinement may inspect a source-history residue, omitted physical-signature row, open-fit refutation row, or operation descent route that was invisible to the endpoint classifier. Those data live in L and D, not in R alone.

If the refinement separates the candidates through L or D, the result is exactly the refusal boundary: the endpoint comparison remains an endpoint comparison, while the full tower-equivalence carrier is not accepted. □

Theorem 1143.6 (TowerEquivalence obligation scope)**.** *Every accepted* $\mathsf{TowerEquivalence}^{\uparrow}$ *read is scoped by the two tower rows T, T', the two approximation rows A, A', physical-fit rows P, P', open-fit rows O, O', objectivity row B, paired ledger L, descent row D, endpoint row R, componentwise* hsame *transport H, provenance Q, and local* $\mathsf{NameCert}_{\mathsf{TowerEquivalence}^{\uparrow}}$ *row N. Any consumer omitting L or D receives only endpoint comparison, not tower equivalence.*

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalence obligation scope`

Proof. Project the carrier of Definition 1143.1 and the local NameCert obligations of Definition 1143.2. The accepted packet displays both tower rows, both approximation rows, both physical-fit rows, both open-fit rows, objectivity, paired ledger, descent, endpoint, transport, provenance, and local naming before any equivalence read is exported.

Use endpoint refusal, Definition 1143.3. It says the endpoint row R is not accepted as tower equivalence unless L, D, B, and H travel with the same comparison, so an endpoint-only consumer remains outside the tower-equivalence surface.

Finally apply ledger-descent stability, Definition 1143.4. The paired ledger and descent row are preserved componentwise under transport, so omitting either row cannot be repaired downstream by provenance, local naming, or endpoint agreement. □

Theorem 1143.7 (TowerEquivalence public export surface)**.** *The public* $\mathsf{TowerEquivalence}^{\uparrow}$ *export surface consists exactly of the carrier of Definition 1143.1, the NameCert obligations of Definition 1143.2, endpoint refusal of Definition 1143.3, ledger-descent stability of Definition 1143.4, refinement separation of Definition 1143.5, and obligation scope of Definition 1143.6. A public consumer may cite tower equivalence only through that surface; endpoint-only agreement, host source equality, unrestricted descent, and observer-independent comparison outside* B, L, H *remain outside the export.*

Lean4 ✓ `BEDC.Derived.TowerEquivalenceUp.TowerEquivalencePublicExportSurface`

Proof. Project the carrier and NameCert obligations. They display the full packet $T, T', A, A', P, P', O, O', B, L, D$, before any equivalence read is available.

Apply endpoint refusal, ledger-descent stability, and refinement separation. These theorems keep endpoint comparison tied to the paired ledger, descent row, observer row, and componentwise transport, while allowing refinements to separate candidates through L or D.

Finally use obligation scope. It identifies the same displayed rows as the accepted scope for every equivalence read. Therefore the public export is exactly the listed paper surface, and the excluded endpoint-only or host-level claims have no carrier coordinate. □

Falsifiable prediction: An accepted TowerEquivalenceUp classifier is refuted by any pair of endpoint-agreeing towers whose displayed refinement exposes a source-ledger mismatch or an operation that descends through one tower but not the other.

Independence witness: realityconstrainedtowerpacket, realityconstrainedapproximationtower, physicalfittower, realityconstrainedopenfit, realityconstrainedobjectivity: none is bijective because $\mathsf{TowerEquivalence}^{\uparrow}$ carries two tower packets, two approximation rows, two physical-fit rows, two open-fit rows, observer-stable transport, paired ledger compatibility, descent compatibility, endpoint agreement, provenance, and local NameCert rows in one packet. Each sibling supplies a constituent surface, while only this packet binds endpoint agreement to ledger and descent compatibility.

The Lean development for this carrier must provide Nontrivial $\mathsf{TowerEquivalence}^{\uparrow}$ and FieldFaithful TowerEquivale TasteGate instances in `lean4/BEDC/Derived/TowerEquivalenceUp/TasteGate.lean`. The expected namespace is `BEDC.Derived.TowerEquivalenceUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{TowerEquivalence}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{TowerEquivalence}^{\uparrow}$ packet is generated from two tower packets, two approximation rows, two physical-fit rows, two open-fit rows, an objectivity row, paired ledger compatibility, descent compatibility, endpoint agreement, componentwise transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TowerEquivalence}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1143.1, Definition 1143.2, Definition 1143.3, Definition 1143.4, Definition 1143.5, Definition 1143.6, and Definition 1143.7 close the paper-side public surface binding endpoint agreement to paired ledger, descent, objectivity, transport, provenance, and local naming rows for finite tower-equivalence packets.

Formal verification: FormalStatus(NameCert$_{\mathsf{TowerEquivalence}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert Lean verification, bridge checking, endpoint-only tower identity, host source equality, unrestricted descent, or observer-independent comparison outside the displayed objectivity row.

Upgrade path: Obligation closure requires Lean encoding of carrier admission, endpoint refusal, ledger-descent stability, refinement separation, obligation scope, TasteGate nontriviality, and field faithfulness for $T, T', A, A', P, P', O, O', B, L, D, R, H, Q, N$.

Chapter 1144

A Concrete Naming Certificate for $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$

The $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ packet realizes the TruthCert field list of Definition 9414.672 as a finite BEDC NameCert carrier. It sits beside the open-fit surface of chapter 1132, the objectivity surface of chapter 1133, the explanation surface of chapter 1136, the induction-stability seal of chapter 1139, and the tower-compression ledger of chapter 1142. Those sibling chapters provide the finite fit, observer-invariant reading, explanatory handoff, continuation stability, and compression ledger rows; this carrier binds exactly the nine TruthCert fields before any reality-constrained truth claim is exported.

Definition 1144.1 (Reality-constrained TruthCert carrier)**.** *A* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N)$$

with the following displayed rows:

1. *S is the finite source row recording observation traces, source bundles, and the claim site.*
2. *Σ is the reality-facing signature row extracted from the source and compared against the model-facing signature.*
3. *K is the classifier row naming the sameness relation used for signature comparison.*
4. *T is the tower row recording approximation levels and the endpoint at which the claim is read.*
5. *U is the stability row naming the continuation class over which finite fit may be reused.*
6. *D is the descent row listing the operations that survive the tower route.*
7. *I is the invariant row recording observer-anchor transport and public content.*
8. *L is the ledger row recording compression, idealization, hidden residue, provenance, and packaging.*
9. *F is the failure-surface row naming finite defeats, domain exits, classifier mismatches, and later observations that can overturn the claim.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row binding the nine displayed TruthCert fields as one packet.*

The carrier has no coordinate for model–reality identity, unbounded confirmation, unledgered compression, observer privilege, hidden classifier choice, or truth export outside the nine displayed TruthCert rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.taste gate`

Definition 1144.2 (Reality-constrained TruthCert classifier)**.** *For two* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *carrier packets*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N) \quad \text{and} \quad Q' = (S', \Sigma', K', T', U', D', I', L', F', N'),$$

the classifier $Q \sim_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}} Q'$ *is the componentwise agreement predicate asserting source agreement between* S *and* S'*, signature agreement between* Σ *and* Σ'*, classifier-row agreement between* K *and* K'*, tower agreement between* T *and* T'*, continuation-stability agreement between* U *and* U'*, descent agreement between* D *and* D'*, invariant-reading agreement between* I *and* I'*, ledger agreement between* L *and* L'*, failure surface agreement between* F *and* F'*, and local-name agreement between* N *and* N'*. The* K *component preserves exactly the accepted signature comparison used by* Σ *and* Σ'*. The* L *and* F *components preserve the ledger rows and failure rows that determine whether an export remains readable through the same local TruthCert packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertClassifier row certificate`

Theorem 1144.3 (Reality-constrained TruthCert classifier obligation surface)**.** *The classifier of Definition 1144.2 is the full comparison surface for the carrier of Definition 1144.1. In particular, an accepted comparison between two* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packets must expose the componentwise rows*

$$S, \Sigma, K, T, U, D, I, L, F, N$$

and their primed counterparts, with K *governing the accepted signature comparison and with* L, F *governing the ledger and failure-surface comparisons.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertClassifierObligationSurfa`

Proof. Start from the carrier definition Definition 1144.1. It exposes exactly the source, signature, classifier, tower, stability, descent, invariant, ledger, failure-surface, and local-name rows.

Now unfold the classifier definition Definition 1144.2. Each displayed carrier row is compared only with its counterpart in the second packet. The K row is therefore fixed before signature comparison is accepted, and the L, F rows are fixed before ledger or failure reads are transported.

Finally use the sibling-dependency theorem Definition 1144.10. The open-fit, objectivity, explanation, physical-induction, and tower-compression siblings supply the finite source comparison, invariant reading, source-to-claim handoff, continuation stability, tower route, descent row, and ledger discipline consumed by this componentwise classifier. □

Theorem 1144.4 (Reality-constrained TruthCert carrier transport)**.** *If* $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$ *is an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet and* $Q \sim_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}} Q'$ *for*

$$Q' = (S', \Sigma', K', T', U', D', I', L', F', N'),$$

then Q' *is accepted by the same carrier surface. Transport preserves the accepted signature comparison through* K'*, preserves the ledger row through* L'*, preserves the failure-surface row through* F'*, and leaves no truth coordinate outside the transported packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertCarrier transport`

Proof. The carrier definition Definition 1144.1 lists all rows required for admission. The classifier definition Definition 1144.2 supplies a comparison for each such row, so every required component of Q' is reached by the matching component of Q.

Apply the classifier obligation surface of Definition 1144.3. It keeps K' attached to the accepted signature comparison and keeps L' and F' attached to the ledger and failure-surface rows rather than to an external truth route.

The sibling-dependency theorem Definition 1144.10 supplies the ambient transport sources for finite fit, invariant reading, explanatory handoff, continuation stability, and tower compression. Thus the transported packet remains inside the same accepted carrier surface. □

Theorem 1144.5 (Reality-constrained TruthCert NameCert obligation exhaustion)**.** *For* RealityConstrainedTruthC *the local* NameCert *obligation surface is exhausted by the carrier of Definition 1144.1, the classifier of Definition 1144.2, the classifier obligation surface of Definition 1144.3, and the carrier transport of Definition 1144.4. Ledger exactness, failure exactness, and export readiness add no second certificate surface; they read the same rows* $S, \Sigma, K, T, U, D, I, L, F, N$.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertNameCertObligationExhaust`

Proof. Use the carrier definition and classifier definition first. Together they give the admitted packet and the accepted packet comparison, including the signature comparison controlled by K and the ledger and failure comparisons controlled by L and F.

Next apply Definition 1144.3 and Definition 1144.4. These results make comparison and transport internal to the displayed packet rather than external to the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ row.

Finally use the downstream readiness blocks Definition 1144.15, Definition 1144.19, and Definition 1144.21. They consume the same ledger, failure, classifier, sibling, and export rows already displayed in the packet. Hence the local NameCert obligations are exhausted by the carrier, classifier, and transport surface, not by any hidden certificate coordinate. □

Theorem 1144.6 (Reality-constrained TruthCert field coverage)**.** *For the carrier of Definition 1144.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *surface covers exactly the source, signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface fields of the certified physical truth packet. Any accepted export of a reality-constrained truth claim must read these fields through* $S, \Sigma, K, T, U, D, I, L, F$ *before the local name* N *is exposed.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertFieldCoverage`

Proof. Carrier admission exposes exactly $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$. The source field is the row S, and the signature field is the row Σ.

The classifier field is the row K. The tower field is the row T, while the stability and descent fields are U and D.

The invariant field is the row I, the ledger field is the row L, and the failure-surface field is the row F. The local name N binds these rows but adds no further truth coordinate.

Thus the NameCert surface covers the nine TruthCert fields and is exhausted by them. A consumer that omits any one row reads a projection of the packet, not an accepted $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ export. □

Theorem 1144.7 (Reality-constrained TruthCert source-signature-classifier obligation)**.** *Every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *truth export must expose the source row* S*, the reality-facing signature row* Σ*, and the classifier row* K *before the local truth claim is readable. The source may not be replaced by an unnamed observation route, the signature may not be replaced by a model-side assertion, and the classifier may not be chosen after export.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertSourceSignatureClassifier`

Proof. Use the carrier of Definition 1144.1. The displayed packet contains S, Σ, K as separate rows: source, reality-facing signature, and classifier.

Apply Definition 1144.6. It makes these rows mandatory fields of the TruthCert surface before the local name N is exposed, so omitting any one of them yields only a projection of the packet.

Finally use Definition 1144.10. The open-fit and explanation siblings supply finite source comparison and source-to-claim handoff, and the classifier row controls the accepted signature comparison. Thus truth export is blocked until S, Σ, K have all been recorded. □

Theorem 1144.8 (Reality-constrained TruthCert stability-descent-invariant obligation)**.** *Every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet must preserve the tower route* T*, the continuation-stability row* U*, the descent row* D*, and the observer-invariant row* I *as a joint obligation surface. These rows certify that the claim is read at the displayed endpoint, reused only over the named continuation class, transported only through the recorded descent operations, and exposed through an observer-invariant public reading.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertStabilityDescentInvariant`

Proof. The carrier in Definition 1144.1 displays the rows T, U, D, I separately. They name the approximation tower, the continuation class, the surviving descent operations, and the observer-anchor transport.

Apply Definition 1144.6. Since tower, stability, descent, and invariant reading are mandatory TruthCert fields, the local name N can certify a truth export only after these four rows are readable.

Use Definition 1144.10. Physical induction supplies the continuation-stability seal, tower compression supplies tower route and descent discipline, and objectivity supplies observer-invariant reading. Therefore T, U, D, I form one preservation surface rather than four optional annotations. □

Theorem 1144.9 (Reality-constrained TruthCert ledger-failure-local-name obligation)**.** *Every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *local name must be preceded by the ledger row* L *and the failure-surface row* F*. The ledger records compression, idealization, hidden residue, provenance, and packaging, while the failure row records finite defeats, domain exits, classifier mismatches, and later observations before* N *binds the packet as a local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$*.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertLedgerFailureLocalNameOb`

Proof. Read the carrier of Definition 1144.1. It assigns compression, idealization, hidden residue, provenance, and packaging to L, assigns the defeat and exit surface to F, and reserves N for the local NameCert row.

Apply Definition 1144.6. The ledger and failure-surface fields are mandatory before the local name is exposed, while N binds the already displayed rows without adding an extra truth coordinate.

Finally use Definition 1144.10. Tower compression controls descent, ledger, and hidden residue; open-fit controls finite defeat; and the remaining siblings route invariant reading, explanation, and

stability. Hence local naming is accepted only after ledger and failure obligations have been recorded in L and F. □

Theorem 1144.10 (Reality-constrained TruthCert sibling dependency)**.** *Every accepted* RealityConstrainedTruthCe *packet consumes the open-fit, objectivity, explanation, physical-induction, and tower-compression dependencies supplied by the sibling certificate family. The open-fit dependency controls finite fit and finite defeat; the objectivity dependency controls invariant observer reading; the explanation dependency controls source-to-claim handoff; the physical-induction dependency controls stability over continuations; and the tower-compression dependency controls descent, ledger, and hidden residue.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertSiblingDependency`

Proof. Read the carrier rows of Definition 1144.1. The rows S, Σ, K, F require the finite source, signature comparison, classifier, and defeat surface supplied by the open-fit certificate family.

The invariant row I requires the observer-anchor discipline supplied by the objectivity certificate family. Without that row, the packet can carry a private observation route but not a public reality-constrained truth claim.

The source-to-claim handoff is supplied by the explanation certificate family. It binds signatures, derivation routes, and admissible explanatory use before the local truth certificate can be read.

The stability row U consumes the physical-induction seal, because finite fit can be reused only over the displayed continuation class. The tower, descent, and ledger rows T, D, L consume the tower-compression certificate, because endpoint claims must retain their approximation route and hidden residue.

Consequently the packet is composite rather than atomic: its accepted route depends on the open-fit, objectivity, explanation, physical-induction, and tower-compression siblings, and no single sibling alone carries the full TruthCert surface. □

Theorem 1144.11 (Reality-constrained TruthCert kernel truth route)**.** *Every accepted* RealityConstrainedTruthCer *truth export is routed only through*

$$S, \Sigma, K, T, U, D, I, L, F, N$$

and through the sibling dependency surface of Definition 1144.10. It is not routed through model–reality identity, hidden observer privilege, unbounded confirmation, or an external truth coordinate.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertKernelTruthRoute`

Proof. Start from the carrier definition Definition 1144.1. It displays the source, signature, classifier, tower, stability, descent, invariant, ledger, failure, and local NameCert rows as the only rows of the packet.

Apply Definition 1144.3 and Definition 1144.6. These theorems make the classifier and all nine exported fields mandatory parts of the displayed route rather than optional side data.

Finally use Definition 1144.10. The open-fit, objectivity, explanation, physical-induction, and tower-compression siblings account for the admissible dependency surface, so no model–reality identity row or external truth coordinate remains available for export. □

Theorem 1144.12 (Reality-constrained TruthCert failure readiness)**.** *In every accepted* RealityConstrainedTruthCe *export, the failure row F is displayed before the truth certificate is readable. The same row remains part of transported classifier agreement whenever the packet is compared, replayed, or carried across a sibling route.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertFailureReadiness`

Proof. First read F from Definition 1144.1. The carrier reserves it for finite defeats, domain exits, classifier mismatches, and later observations before local naming.

Use Definition 1144.9. It places F before the local NameCert row N, so export cannot precede the displayed failure surface.

Then apply Definition 1144.4 together with Definition 1144.3. Transported agreement must preserve the classifier-facing rows already exposed by the packet, and the failure row remains one of those rows rather than a removable annotation. □

Theorem 1144.13 (Reality-constrained TruthCert public export boundary)**.** *A public* RealityConstrainedTruthCert *export is accepted only after the source row S, classifier row K, tower row T, stability row U, invariant row I, ledger row L, failure row F, and local NameCert row N are all present, with the signature and descent rows Σ, D displayed in the same carrier.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertPublicExportBoundary`

Proof. Begin with Definition 1144.1. The public packet is $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$, so each named row must be available inside the same finite carrier before any export can be read.

Next use Definition 1144.6. It identifies the required source, signature, classifier, tower, stability, descent, invariant, ledger, failure, and local naming fields, and it leaves no public export row outside them.

Finally combine Definition 1144.9 with Definition 1144.10. The ledger and failure rows precede local naming, while the sibling surface supplies the dependency discipline for source, classifier, tower, stability, invariant, ledger, and failure. Hence public export is accepted only at the full displayed boundary. □

Theorem 1144.14 (Reality-constrained TruthCert carrier admission)**.** *An accepted* RealityConstrainedTruthCert$^{\uparrow}$ *packet exposes exactly the carrier*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N).$$

The accepted export may project the source, signature, classifier, tower, stability, descent, invariant, ledger, failure-surface, and local-name rows, but it has no additional TruthCert coordinate outside this displayed packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertCarrierAdmission`
`eventflow boundary`

Proof. Use the carrier definition Definition 1144.1. Its displayed packet is exactly $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$.

Apply the field-coverage theorem Definition 1144.6. It identifies the source, signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface fields with the rows $S, \Sigma, K, T, U, D, I, L, F$, and it reserves N for local naming.

No remaining row is available for carrier admission. A consumer that asks for a second truth coordinate is therefore asking outside the admitted packet rather than reading an accepted RealityConstrainedTruthCert$^{\uparrow}$ export. □

Theorem 1144.15 (Reality-constrained TruthCert ledger exactness)**.** *In every accepted* RealityConstrainedTruthCer *packet, the ledger row L records compression, idealization, hidden residue, provenance, and package descent before any truth export is readable. These records are routed through the open-fit, objectivity, explanation, physical-induction stability, and tower-compression siblings before the local name N is exposed.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertLedgerExactness`

Proof. The carrier definition Definition 1144.1 assigns compression, idealization, hidden residue, provenance, and packaging to the ledger row L. The field-coverage theorem Definition 1144.6 then makes L a mandatory TruthCert field rather than an optional annotation.

Now use the sibling-dependency theorem Definition 1144.10. Open-fit controls finite fit and finite defeat; objectivity controls public observer reading; explanation controls source-to-claim handoff; physical induction controls continuation stability; and tower compression controls descent, ledger, and hidden residue.

Consequently the ledger is exact at the point of export: compression, idealization, hidden residue, provenance, and package descent must already be visible in L and routed through the named siblings before N can certify the truth packet. □

Theorem 1144.16 (Reality-constrained TruthCert non-escape)**.** *No accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet exports model–reality identity, unbounded confirmation, observer privilege, hidden classifier choice, or truth outside the displayed rows* $S, \Sigma, K, T, U, D, I, L, F, N$.

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertNonescape`

Proof. Inspect the carrier definition Definition 1144.1. Its final sentence excludes model–reality identity, unbounded confirmation, unledgered compression, observer privilege, hidden classifier choice, and truth export outside the nine displayed TruthCert rows.

The field-coverage theorem Definition 1144.6 makes the displayed rows exhaustive for accepted export. The sibling-dependency theorem Definition 1144.10 further routes finite fit, objectivity, explanation, induction stability, and compression through their own certificate surfaces.

Therefore none of the excluded demands has a readable coordinate in the packet. Such a demand can only be recorded at the failure-surface or ledger boundary, not exported as part of the local TruthCert name. □

Theorem 1144.17 (Reality-constrained TruthCert classifier stability)**.** *Let*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N) \quad \textit{and} \quad Q' = (S', \Sigma', K', T', U', D', I', L', F', N')$$

be accepted $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packets. Any accepted comparison between* Q *and* Q' *must preserve the signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface rows componentwise:* Σ *is compared only with* Σ'*,* K *only with* K'*,* T *only with* T'*,* U *only with* U'*,* D *only with* D'*,* I *only with* I'*,* L *only with* L'*, and* F *only with* F'*. No comparison accepted by the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row may substitute one of these rows for another.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertClassifier stability`

Proof. Project the two carriers through Definition 1144.1. The displayed packets expose the rows $S, \Sigma, K, T, U, D, I, L, F, N$ and $S', \Sigma', K', T', U', D', I', L', F', N'$ with the same row order and the same local naming role.

Use Definition 1144.6. It makes the signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface fields readable exactly through the displayed rows $\Sigma, K, T, U, D, I, L, F$ and their primed counterparts.

Now use Definition 1144.10. The open-fit sibling controls signature comparison and finite defeat, the objectivity sibling controls invariant reading, the explanation sibling controls source-to-claim handoff, the physical-induction sibling controls stability, and the tower-compression sibling controls tower, descent, ledger, and hidden residue.

Therefore an accepted comparison has no free classifier or cross-row transport left to choose. It preserves each displayed obligation row against its counterpart, while N and N' only name the already displayed packets. □

Theorem 1144.18 (Reality-constrained TruthCert ledger non-escape)**.** *No accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *truth export may bypass the ledger row L. In particular, compression, idealization, provenance, and hidden residue must be readable through L before the local name N can certify the exported truth packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertLedgerNonescape`

Proof. Read the carrier row L in Definition 1144.1. It is the displayed ledger coordinate for compression, idealization, hidden residue, provenance, and packaging.

Apply Definition 1144.6. The ledger is one of the nine mandatory TruthCert fields, so an export that omits L reads only a projection of the packet.

Apply Definition 1144.15. The ledger entries must already record compression, idealization, hidden residue, provenance, and package descent before truth export is readable.

Finally use Definition 1144.10. The tower-compression sibling controls descent, ledger, and hidden residue, while the other siblings route finite fit, invariant reading, explanatory handoff, and continuation stability. Thus a truth export that bypasses L bypasses a required sibling-routed row and is not accepted by the local certificate. □

Theorem 1144.19 (Reality-constrained TruthCert failure-surface exactness)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet, later observations, classifier mismatches, domain exits, and finite defeats are accepted exactly through the displayed failure-surface row F. They are not hidden in the source, classifier, stability, invariant, ledger, or local-name rows.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertFailureSurfaceExactness`

Proof. The carrier definition Definition 1144.1 assigns finite defeats, domain exits, classifier mismatches, and later observations to the failure-surface row F.

The field-coverage theorem Definition 1144.6 makes F a mandatory TruthCert field and exhausts the accepted export by the displayed rows. Therefore a claimed defeat route outside F is outside the local TruthCert surface.

The sibling-dependency theorem Definition 1144.10 identifies the dependencies that feed this surface. Open-fit supplies finite fit and finite defeat, the classifier row supplies signature comparison, physical induction supplies the continuation class, and tower compression supplies descent and hidden residue.

Consequently later observations, classifier mismatches, domain exits, and finite defeats have a single accepted public landing row: F. The other rows may supply the data that makes a defeat meaningful, but they do not certify a separate failure surface. □

Theorem 1144.20 (Reality-constrained TruthCert export defeat exhaustion)**.** *Let an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet expose the rows $S, \Sigma, K, T, U, D, I, L, F, N$. If F records a later observation, classifier mismatch, domain exit, or finite defeat, then no truth export through the local name N is accepted unless that defeat is discharged through the same displayed $S, \Sigma, K, T, U, D, I, L, F, N$ route. A purported export that hides F or bypasses it is outside the TruthCert packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertExportDefeatExhaustion`

Proof. The carrier coverage theorem Definition 1144.6 first fixes the only rows that the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ surface can expose. Thus the name N does not create a second export channel; it binds the nine displayed rows after they are readable.

The ledger exactness theorem Definition 1144.15 keeps compression, residue, provenance, and package descent visible at L. Together with the sibling-dependency theorem Definition 1144.10, this forces open-fit, objectivity, explanation, physical-induction, and tower-compression data to travel through the displayed route before a truth claim reaches N.

Finally apply Definition 1144.19. Later observations, classifier mismatches, domain exits, and finite defeats land only at F. Therefore a discharge of such a defeat must be visible through the same carrier route that made F visible. If a purported export hides or bypasses F, it is not an accepted export through the local TruthCert packet. □

Theorem 1144.21 (Reality-constrained TruthCert export readiness)**.** *An accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *truth export is readable through the local name N only after field coverage, sibling dependency routing, ledger exactness, classifier stability, and failure-surface exactness have all been read through the displayed carrier*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N).$$

Thus the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row certifies export readiness only for the full ten-row packet, not for a projection of the packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertExportReadiness`

Proof. Start with Definition 1144.14 and Definition 1144.6. They make $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$ the admitted carrier and make $S, \Sigma, K, T, U, D, I, L, F$ the mandatory readable TruthCert rows before N is exposed.

Next use Definition 1144.10. It routes finite fit, objectivity, explanation, physical-induction stability, and tower compression through the displayed packet rather than through an external export channel.

The ledger and classifier obligations are fixed by Definition 1144.15 and Definition 1144.17. The former keeps compression, idealization, hidden residue, provenance, and package descent visible at L, while the latter prevents signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface rows from being substituted across comparisons.

Finally apply Definition 1144.19. Later observations, classifier mismatches, domain exits, and finite defeats land at F. Therefore an accepted export through N is ready only when all these readings have already passed through the displayed carrier rows. □

Theorem 1144.22 (Reality-constrained TruthCert methodology-ledger handoff)**.** *Every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *export consumed as methodology authority must route through* RealityConstrainedMetho *before the local name N is read. The methodology consumer must preserve the TruthCert source, signature, classifier, tower, stability, descent, invariant, ledger, and failure rows together with the methodology-ledger export row, rather than reading a projection of the TruthCert packet as an independent methodological warrant.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertMethodologyLedgerHandoff`

Proof. Start with field coverage and export readiness, Definition 1144.6 and Definition 1144.21. They force every accepted export through $S, \Sigma, K, T, U, D, I, L, F, N$ before the local name is exposed.

Next use sibling dependency and ledger exactness, Definition 1144.10 and Definition 1144.15. These keep explanation, audit-facing source use, observer reading, physical-induction stability, tower compression, provenance, hidden residue, and package descent visible at the point where the TruthCert export is consumed.

The methodology-ledger chapter records exactly the downstream scope, domain-leakage refusal, upgrade route, and methodology failure surface for such consumption. Thus a methodology consumer cannot treat the TruthCert name as freestanding authority. It must first pass the accepted export through $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, where the export discipline is recorded before N is read. □

Theorem 1144.23 (Reality-constrained TruthCert open-fit export boundary)**.** *Every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *export whose failure row changes must route through the open-fit and failure-surface siblings before export. A changed row F is accepted only when the finite-fit, defeat, classifier-mismatch, domain-exit, continuation-scope, and ledger reads remain visible through the displayed carrier $S, \Sigma, K, T, U, D, I, L, F, N$.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertOpenFitExportBoundary`

Proof. Use failure-surface exactness, Definition 1144.19. Later observations, classifier mismatches, domain exits, and finite defeats have the single public landing row F.

Apply sibling dependency and classifier stability, Definition 1144.10 and Definition 1144.17. The open-fit sibling controls finite fit and finite defeat, while the classifier row prevents a changed failure surface from being compared by substituting another displayed row.

Finally use export readiness. Since N is readable only after field coverage, sibling routing, ledger exactness, classifier stability, and failure-surface exactness have all been read, a changed F must be routed through the open-fit and failure-surface surfaces before export. Otherwise the changed failure row is outside the accepted TruthCert packet. □

Theorem 1144.24 (Reality-constrained TruthCert observer-invariant ledger route)**.** *Every observer-invariant* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *truth export must preserve the invariant row I and the ledger row L together. A consumer may not read observer invariance without the compression, idealization, hidden-residue, provenance, and package-descent ledger, and may not read the ledger as public truth without the observer-invariant row.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertObserverInvariantLedgerRo`

Proof. Project the carrier through Definition 1144.1. The invariant row I and ledger row L are separate displayed coordinates, but both are mandatory fields of the same accepted packet.

Use the stability-descent-invariant obligation and ledger exactness, Definition 1144.8 and Definition 1144.15. The former keeps observer-anchor transport joined to tower, stability, and descent rows; the latter keeps compression, idealization, hidden residue, provenance, and package descent visible at L.

Now apply export readiness. Since the local name N certifies export only after all mandatory rows have been read through the same carrier, I and L cannot be exported as independent authorities. Observer-invariant truth export preserves the invariant and ledger rows as one route through the accepted TruthCert packet. □

Theorem 1144.25 (Reality-constrained TruthCert failure routing)**.** *For an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet, defeating observations, domain exits, classifier mismatches, and later observations land only in the row F before the local name N certifies export. They do not create a hidden truth coordinate, an unledgered compression route, or a classifier choice outside $S, \Sigma, K, T, U, D, I, L, F, N$.*

Proof. Use Definition 1144.19. It assigns later observations, classifier mismatches, domain exits, and finite defeats to the displayed failure-surface row F and excludes separate failure surfaces in the source, classifier, stability, invariant, ledger, and local-name rows.

Now apply Definition 1144.16. The carrier has no row for model–reality identity, unbounded confirmation, observer privilege, hidden classifier choice, or truth outside the displayed packet.

Finally use Definition 1144.18. The ledger row L must already expose compression, idealization, hidden residue, provenance, and package descent before N names the export. Hence a failure cannot be routed by hiding in the ledger or by bypassing F; it reaches the local certificate only through the displayed failure row. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertFailureRouting`

Theorem 1144.26 (Reality-constrained TruthCert sibling handoff totality)**.** *Every accepted* RealityConstrainedTrut *packet consumes the open-fit, objectivity, explanation, physical-induction stability, and tower-compression sibling handoffs before the local name N exports its TruthCert packet. No accepted export may skip one of these sibling routes while still claiming the full rows $S, \Sigma, K, T, U, D, I, L, F, N$.*

Proof. Expand Definition 1144.10. Open-fit controls finite fit and finite defeat, objectivity controls observer-invariant reading, explanation controls source-to-claim handoff, physical induction controls continuation stability, and tower compression controls tower route, descent, ledger, and hidden residue.

Then apply Definition 1144.21. Export through N is readable only after field coverage, sibling routing, ledger exactness, classifier stability, and failure-surface exactness have all been read through $S, \Sigma, K, T, U, D, I, L, F, N$.

If one sibling handoff were skipped, at least one required row would lose its source of admissible data: finite fit or defeat, invariant reading, explanatory handoff, continuation stability, or compression and descent. Such a packet would be a projection rather than the accepted TruthCert carrier.

Therefore the five sibling routes are total for this export surface. They are all consumed before N certifies the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ packet. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertSiblingHandoffTotality`

Theorem 1144.27 (Reality-constrained TruthCert root readiness packet)**.** *For the carrier*

$$Q = (S, \Sigma, K, T, U, D, I, L, F, N),$$

root readiness of $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *is exactly the packet route that passes through carrier admission, field coverage, sibling handoff totality, ledger exactness, failure-surface exactness, and export readiness over the displayed rows $S, \Sigma, K, T, U, D, I, L, F, N$. A root consumer receives a TruthCert packet only along that route, not along a separate truth coordinate or a projection of the carrier.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootReadinessPacket`

Proof. Start with carrier admission and field coverage, Definition 1144.14 and Definition 1144.6. They identify $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$ as the admitted carrier and make the source, signature, classifier, tower, stability, descent, invariant, ledger, and failure-surface rows mandatory before N is readable.

Next use sibling handoff totality, Definition 1144.26. It forces the open-fit, objectivity, explanation, physical-induction stability, and tower-compression routes to be consumed before the local name exports the packet.

Finally apply ledger exactness, failure-surface exactness, and export readiness: Definition 1144.15, Definition 1144.19, and Definition 1144.21. These results make L and F exact rows and allow N to certify export only after the same displayed packet has been read. Thus root readiness is neither more nor less than this carrier route through $S, \Sigma, K, T, U, D, I, L, F, N$. □

Theorem 1144.28 (Reality-constrained TruthCert root defeat projection)**.** *Any downstream consumer that changes or omits the failure row F of an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet reads only a projection of the TruthCert carrier. Such a consumer does not receive the root-ready packet unless the changed or disputed failure surface is routed through the displayed open-fit, defeat, classifier, ledger, and export rows before N is read.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootDefeatProjection`

Proof. Use failure routing, Definition 1144.25. Defeating observations, domain exits, classifier mismatches, and later observations land only in F before N certifies export; they do not create a hidden truth coordinate or an unledgered compression route.

Now apply the open-fit export boundary, Definition 1144.23. A changed F must remain visible through the open-fit and failure-surface siblings together with the carrier rows $S, \Sigma, K, T, U, D, I, L, F, N$.

Finally use export defeat exhaustion, Definition 1144.20. A later observation, classifier mismatch, domain exit, or finite defeat recorded by F must be discharged through the same displayed route. Omitting or changing F without that route leaves only a projection, so the downstream consumer has not received the root-ready TruthCert packet. □

Theorem 1144.29 (Reality-constrained TruthCert root consumer totality)**.** *Every methodology, open-fit, objectivity, physical-induction, or tower-compression consumer of the* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *name must consume all five sibling routes before N exports the TruthCert packet. No such consumer may read one sibling route as an independent root authority while claiming the full rows $S, \Sigma, K, T, U, D, I, L, F, N$.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootConsumerTotality`

Proof. Begin with sibling dependency and sibling handoff totality, Definition 1144.10 and Definition 1144.26. They assign finite fit and defeat to open-fit, invariant reading to objectivity, source-to-claim handoff to explanation, continuation stability to physical induction, and tower route, descent, ledger, and hidden residue to tower compression.

For methodology consumption, use Definition 1144.22. It requires the methodology consumer to preserve the TruthCert source, signature, classifier, tower, stability, descent, invariant, ledger, and failure rows together with the methodology-ledger export row.

For the remaining root consumers, apply export readiness and root readiness, Definition 1144.21 and Definition 1144.27. The local name N is readable only after field coverage, sibling routing, ledger exactness, classifier stability, and failure-surface exactness have all passed through the displayed carrier. Therefore each named downstream consumer must consume all five sibling routes before treating N as the exported TruthCert packet. □

Theorem 1144.30 (Reality-constrained TruthCert root carrier obligation)**.** *Every root-facing consumer of* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *must receive the displayed carrier packet $S, \Sigma, K, T, U, D, I, L, F, N$. Omitting any one of these rows leaves only a projection and cannot be read as the root-ready TruthCert export named by N.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootCarrierObligation`

Proof. Use the carrier and classifier definitions, Definition 1144.1 and Definition 1144.2. They make the ten displayed rows the whole accepted packet and the whole comparison surface.

Apply root consumer totality, Definition 1144.29. Each methodology, open-fit, objectivity, physical-induction, or tower-compression consumer must consume the sibling routes through this same packet. Therefore a consumer that omits one displayed row has not received the carrier required for root export. □

Theorem 1144.31 (Reality-constrained TruthCert root classifier stability)**.** *If a root-facing transport compares two accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packets, then the classifier rows* K, K' *remain the rows governing signature comparison for the transported packets* $S, \Sigma, K, T, U, D, I, L, F, N$ *and* $S', \Sigma', K', T', U', D', I', L', F', N'$*. The transport cannot replace the classifier row while preserving root readiness.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootClassifierStability`

Proof. Start with classifier stability and carrier transport, Definition 1144.17 and Definition 1144.4. They keep the classifier component attached to the accepted signature comparison across the transported carrier.

Now use root readiness, Definition 1144.27. Root readiness is the carrier route through field coverage, sibling handoff, ledger exactness, failure exactness, and export readiness. A changed classifier row would leave that route, so the transported export remains root-ready only when K and K' are the classifier rows being compared. □

Theorem 1144.32 (Reality-constrained TruthCert root ledger nonescape)**.** *No root-facing export may expose the local name* N *while bypassing the ledger row* L *or the failure row* F*. The accepted export of* $S, \Sigma, K, T, U, D, I, L, F, N$ *therefore records compression, provenance, hidden residue, finite defeat, domain exit, classifier mismatch, and later observation before* N *is readable.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootLedgerNonescape`

Proof. Apply ledger exactness and failure-surface exactness, Definition 1144.15 and Definition 1144.19. They make L and F exact displayed rows, not optional annotations.

Finally use export readiness and sibling handoff totality, Definition 1144.21 and Definition 1144.26. The local name N is readable only after all displayed rows and sibling routes are consumed, so an export that bypasses L or F cannot be the root-ready $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ packet. □

Falsifiable prediction: A purported reality-constrained truth export that omits the displayed source, signature, classifier, tower, stability, descent, invariant, ledger, or failure-surface row is not an accepted TruthCert packet.

Independence witness: reality-constrained-open-fit, reality-constrained-objectivity, reality-constrained-explanation, physical-induction-stability-seal, reality-constrained-tower-compression: the full nine-field TruthCert packet is available only after these sibling dependencies are consumed together.

Theorem 1144.33 (Reality-constrained TruthCert L10 source readiness)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet*

$$(S, \Sigma, K, T, U, D, I, L, F, N),$$

the L10 consumers $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *may read the TruthCert surface only through the finite source row* S, *signature row* Σ, *classifier row* K, *tower row* T, *stability row* U, *descent row* D, *observer-invariant row* I, *ledger row* L, *failure row* F, *and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row* N. *No L10 consumer receives an unlisted source, ambient truth predicate, global reality oracle, or off-packet continuation.*

Proof. Read the carrier and field-coverage rows from Definition 1144.1 and Definition 1144.6. They make the displayed tuple the full finite packet, with $S, \Sigma, K, T, U, D, I, L, F, N$ as the available rows.

Next apply sibling dependency and sibling handoff totality, Definition 1144.10 and Definition 1144.26. These place $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ behind the displayed TruthCert handoff rather than behind an independent source supply.

Finally use export readiness, Definition 1144.21. The export row is root-ready only after the finite source, classifier, ledger, failure, and local naming rows are consumed, so every L10 read is confined to the accepted packet. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10SourceReadiness`

Theorem 1144.34 (Reality-constrained TruthCert L10 failure ledger lock)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet* $(S, \Sigma, K, T, U, D, I, L, F, N)$, *each L10 read by* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *that reaches the classifier row* K *must also keep the ledger row* L *and the failure row* F *in the same packet. The read cannot drop finite defeat, domain exit, classifier mismatch, hidden residue, later observation, or compression provenance while retaining the local name* N.

Proof. Start with ledger exactness and failure-surface exactness, Definition 1144.15 and Definition 1144.19. They identify L and F as displayed rows of the accepted packet, not optional annotations.

Use field coverage to keep K, L, F, N in the same finite carrier. A consumer that exposes K through the local name must therefore remain inside $S, \Sigma, K, T, U, D, I, L, F, N$.

Apply export readiness to the L10-facing handoff. Since the export cannot expose N before ledger and failure rows have been consumed, any read that drops L or F is outside the $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ route. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCert l10 failure ledger lock`

Theorem 1144.35 (Reality-constrained TruthCert L10 export totality)**.** *The L10 export of an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet* $(S, \Sigma, K, T, U, D, I, L, F, N)$ *is total for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$: *every accepted downstream read is accounted for by the finite source row* S, *the classifier row* K, *the ledger row* L, *the failure row* F, *and the local* $\mathsf{NameCert}$ *row* N, *together with the intervening signature, tower, stability, descent, and observer-invariant rows. There is no second L10 export through hidden source material, hidden failure repair, host equality, or an ambient truth object.*

Proof. Assemble the packet from carrier coverage and sibling dependency. These rows state that the TruthCert object and its downstream consumers are expressed by the displayed finite carrier and its listed sibling routes.

The ledger and failure lock fixes the classifier-facing part of the export. By Definition 1144.34, any L10 read that keeps the local name also keeps the classifier, ledger, and failure rows inside the same packet.

Finally apply export readiness and sibling handoff totality. Together they make the displayed packet the only source-readiness route for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the L10 export is total over the listed rows and has no hidden branch. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10ExportTotality`

Theorem 1144.36 (Reality-constrained TruthCert L10 readiness obligation surface)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet* $(S, \Sigma, K, T, U, D, I, L, F, N)$*, the three L10 rows Definition 1144.33, Definition 1144.34, and Definition 1144.35 form the complete readiness obligation surface for routing the packet through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The surface consists exactly of source confinement to the displayed packet, classifier-ledger-failure co-residence inside that packet, and total export through the finite packet rows* $S, \Sigma, K, T, U, D, I, L, F, N$*. It exposes no ambient truth predicate, quotient real equality, selected limit, global reality oracle, or off-packet continuation.*

Proof. First use Definition 1144.33. It confines every L10 read by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to the finite rows $S, \Sigma, K, T, U, D, I, L, F, N$, so no consumer can begin from an unlisted source, ambient truth predicate, global reality oracle, or off-packet continuation.

Next use Definition 1144.34. This keeps the classifier row K, the ledger row L, the failure row F, and the local name row N co-resident whenever a downstream L10 read reaches the classifier. Hence the readiness surface cannot retain classifier evidence while dropping defeat, domain exit, mismatch, hidden residue, observation, or compression provenance.

Finally apply Definition 1144.35. The export row accounts for every accepted downstream read by the displayed finite packet, so there is no alternate route through quotient real equality, selected limits, host equality, hidden repair, or an ambient truth object. The three rows therefore give the complete L10 obligation surface needed by the downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10ReadinessObligationSur`

Theorem 1144.37 (Reality-constrained TruthCert L10 terminal source boundary)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet* $(S, \Sigma, K, T, U, D, I, L, F, N)$*, any terminal L10 consumer read by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *that is licensed by the TruthCert route must pass through the same displayed source row* S*, classifier row* K*, ledger row* L*, failure row* F*, export row, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row* N*. The terminal read cannot substitute quotient real equality, host equality, model–reality identity, selected limits, hidden repair, or off-packet truth export for those packet rows.*

Proof. First apply Definition 1144.33. It confines each $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ read to the displayed packet $S, \Sigma, K, T, U, D, I, L, F, N$, so the terminal consumer has no source outside the accepted TruthCert carrier.

Next use Definition 1144.34. This keeps the classifier row K, the ledger row L, the failure row F, and the local name row N co-resident whenever the terminal route reaches classifier evidence.

The export side is exhausted by Definition 1144.35 and Definition 1144.36. Together they say that every accepted L10 read is already accounted for by the finite source, classifier, ledger, failure, export, and naming rows of the same packet.

Finally identify the downstream Real and Completion consumers through Definition 1144.42, Definition 1144.43, and Definition 1144.44. These root L10 obligations place physical fit, open-fit refusal, and terminal $\mathsf{Real}^{\uparrow}$ access on the same source boundary, blocking quotient real equality, host equality, model–reality identity, selected limits, hidden repair, and off-packet truth export. □

Lean4 ✓ BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10TerminalSourceBoundary

Theorem 1144.38 (Reality-constrained TruthCert L10 terminal readback determinacy)**.** *Let* $Q = (S, \Sigma, K, T, U, D, I, L, F, N)$ *and* $Q' = (S', \Sigma', K', T', U', D', I', L', F', N')$ *be accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packets. Suppose the terminal L10 reads of* Q *and* Q' *by* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *expose the same displayed source row, classifier row, ledger row, failure row, export row, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row. Then the two terminal reads determine the same accepted terminal TruthCert readback. No quotient real equality, host equality, selected limit, hidden repair, off-packet truth export, or ambient truth predicate can distinguish the two readbacks.*

Proof. Apply Definition 1144.33 to both packets. Each terminal read is confined to the displayed finite rows of its accepted TruthCert carrier, so a comparison between the two reads has no source material beyond the listed source, signature, classifier, tower, stability, descent, observer-invariant, ledger, failure, and local naming rows.

Next use Definition 1144.34. Since the exposed classifier, ledger, failure, and local name rows agree, the classifier-facing portions of the two readbacks stay co-resident in the same displayed packet positions. The terminal comparison cannot keep a classifier row while changing defeat, domain exit, mismatch, hidden residue, observation, or compression provenance.

The accepted export is exhausted by Definition 1144.35 and Definition 1144.36. Thus every downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read is already accounted for by the displayed rows shared by the two packets, together with the same export obligation.

Finally apply Definition 1144.37. It rules out the off-packet substitutes named in the statement, so nothing outside the common displayed source, classifier, ledger, failure, export, and local NameCert rows can separate the two accepted terminal TruthCert readbacks. □

Lean4 ✓ BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10TerminalReadbackDeterm

Theorem 1144.39 (Reality-constrained TruthCert root TasteGate obligation)**.** *For an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet*

$$(S, \Sigma, K, T, U, D, I, L, F, N),$$

the root TasteGate obligation is determined by exactly the displayed source, signature, classifier, tower, stability, descent, invariant, ledger, failure, and local-name rows. The Lean TasteGate instance reads this finite packet and does not read a hidden truth coordinate, a hidden classifier choice, or an off-packet source of reality.

Proof. Use the carrier and field-coverage rows, Definition 1144.1 and Definition 1144.6. They make $(S, \Sigma, K, T, U, D, I, L, F, N)$ the whole admitted TruthCert packet before the local name is exposed.

Next apply sibling dependency, root readiness, and root consumer totality, Definition 1144.10, Definition 1144.27, and Definition 1144.29. These results force every root consumer to pass through the displayed sibling routes and through the same finite carrier.

Finally use root classifier stability and root ledger nonescape, Definition 1144.31 and Definition 1144.32. The classifier row K, ledger row L, and failure row F remain internal to the packet, so the TasteGate read has no coordinate beyond $S, \Sigma, K, T, U, D, I, L, F, N$. □

Lean4 ✓ BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootTasteGateObligation

Theorem 1144.40 (Reality-constrained TruthCert root nontrivial readiness)**.** *The root nontriviality obligation for* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *is witnessed at the packet surface*

$$(S, \Sigma, K, T, U, D, I, L, F, N).$$

A nontrivial TasteGate reading must distinguish accepted packet presence from off-packet absence by the displayed carrier, field coverage, sibling dependency, root readiness, root consumer totality, root classifier stability, and root ledger nonescape rows, not by a separate truth object.

Proof. Carrier admission and field coverage first supply the finite packet and its mandatory rows. Thus packet presence is read through $S, \Sigma, K, T, U, D, I, L, F, N$, while off-packet absence has no local $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ row to expose.

Sibling dependency, root readiness, and root consumer totality then determine which downstream reads count as root-facing reads. A consumer that omits one of the displayed sibling routes or displayed rows receives only a projection, so it cannot be the nontrivial root packet.

Root classifier stability and root ledger nonescape close the distinction at the classifier and ledger boundary. The rows K, L, F, N remain part of the same finite packet, so the Lean TasteGate instance may witness nontriviality from packet presence versus off-packet absence without consulting any hidden truth coordinate. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootNontrivialReadiness`

Theorem 1144.41 (Reality-constrained TruthCert root field-faithful readiness)**.** *The root field-faithfulness obligation for* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *is ready exactly when every TasteGate read of*

$$(S, \Sigma, K, T, U, D, I, L, F, N)$$

preserves the displayed rows as the TruthCert fields. In particular, the source, signature, classifier, tower, stability, descent, invariant, ledger, failure, and local-name rows are not compressed into a hidden truth coordinate, and no root consumer may replace their field order by an ambient object.

Proof. Start from the carrier and field-coverage theorems. They identify the accepted packet rows and make them the TruthCert fields that must be preserved before N becomes readable.

Apply sibling dependency, root readiness, and root consumer totality. These theorems route the open-fit, objectivity, explanation, physical-induction, and tower-compression dependencies through the same displayed packet, so a root-facing read cannot rearrange the fields through a second consumer path.

Finally use root classifier stability and root ledger nonescape. They keep K attached to signature comparison and keep L, F attached to ledger and failure discipline. Hence a field-faithful Lean TasteGate instance reads exactly $S, \Sigma, K, T, U, D, I, L, F, N$ and nothing outside that finite packet. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertRootFieldFaithfulReadines`

Theorem 1144.42 (Reality-constrained TruthCert L10 source exposure)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet*

$$(S, \Sigma, K, T, U, D, I, L, F, N),$$

the source exposed to the L10 chain $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *is exactly the displayed TruthCert surface. The chain reads the source row S, signature row Σ, classifier row K, approximation tower row T, continuation-stability row U, descent row D, observer-invariant row I, methodology-ledger row L, open-fit failure row F, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$ *row N through the same finite packet, not through model–reality identity, unbounded confirmation, observer privilege, host equality, quotient real equality, or a direct truth export.*

Proof. Start from field coverage and sibling dependency, Definition 1144.6 and Definition 1144.10. They project the accepted carrier to the nine displayed TruthCert rows and place the open-fit, objectivity, explanation, physical-induction, and tower-compression dependencies on that same carrier.

Next use L10 source readiness, Definition 1144.33. It routes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads through $S, \Sigma, K, T, U, D, I, L, F, N$ and leaves no unlisted source, ambient truth predicate, global reality oracle, or off-packet continuation for the L10 consumer.

Finally apply the root TasteGate obligation Definition 1144.39. The root read consumes the same finite packet, so the L10 exposure cannot be replaced by identity between the model and reality, unbounded confirmation, observer privilege, host equality, quotient real equality, or direct truth export. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertL10SourceExposure`

Theorem 1144.43 (Reality-constrained TruthCert physical-fit nonescape)**.** *Every physical-fit read of an accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet that reaches* $\mathsf{Real}^{\uparrow}$ *must also expose the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *and* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *sibling surfaces in the same nine-row TruthCert boundary. In particular, finite fit, open continuation, finite defeat, methodology scope, domain-leakage refusal, provenance, hidden residue, and local naming remain recorded in* $S, \Sigma, K, T, U, D, I, L, F, N$*; no consumer may export physical fit by quotient real equality, host equality, model–reality identity, unbounded confirmation, observer privilege, or direct truth export.*

Proof. Project the carrier by field coverage. The rows $S, \Sigma, K, T, U, D, I, L, F, N$ are mandatory before the local name is readable, so a physical-fit read that drops one of them is only a projection of the TruthCert packet.

Use sibling dependency to identify the two sibling surfaces. The $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ route supplies finite fit, open continuation, and finite defeat, while $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ supplies methodology scope, refusal, provenance, hidden residue, and consumer discipline.

Now apply L10 source readiness together with the root TasteGate obligation. The $\mathsf{Real}^{\uparrow}$ read enters through the same L10 source boundary that serves $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, and the root read has no hidden coordinate beyond the displayed packet.

Thus physical fit cannot escape through quotient real equality, host equality, model–reality identity, unbounded confirmation, observer privilege, or direct truth export. It remains a finite TruthCert read with the open-fit and methodology-ledger surfaces visible. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertPhysicalFitNonescape`

Theorem 1144.44 (Reality-constrained TruthCert open-fit refusal surface)**.** *For every accepted* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *packet* $(S, \Sigma, K, T, U, D, I, L, F, N)$*, any downstream read that uses the* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ *surface must carry the refusal information through the same L10 chain* $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ *and through the* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *row. The refusal surface records finite defeat, domain exit, classifier mismatch, later observation, compression residue, and local naming, and it refuses model–reality identity, unbounded confirmation, observer privilege, host equality, quotient real equality, and direct truth export.*

Proof. Read the nine rows from field coverage. The failure row F, ledger row L, classifier row K, tower row T, and local name N are part of the same accepted carrier, so the refusal information cannot be detached from the TruthCert packet.

Apply sibling dependency to the open-fit and methodology-ledger routes. The open-fit surface contributes finite defeat and later-observation exposure, while the methodology-ledger surface records domain exit, refusal, compression residue, consumer discipline, and provenance.

Use L10 source readiness to place the refusal surface on the $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ chain. The L10 read is therefore a finite source read of $S, \Sigma, K, T, U, D, I, L, F, N$, not a separate physical-truth channel.

Finally use the root TasteGate obligation. Since the root read consumes exactly the displayed packet, the open-fit refusal surface blocks model–reality identity, unbounded confirmation, observer privilege, host equality, quotient real equality, and direct truth export. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCertOpenFitRefusalSurface`

Theorem 1144.45 (Reality-constrained TruthCert scoped TasteGate route)**.** *The scoped surface of* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *is the route through the carrier of Definition 1144.1, the field coverage of Definition 1144.6, the sibling dependency of Definition 1144.10, the root TasteGate obligation of Definition 1144.39, the root nontrivial readiness of Definition 1144.40, the root field-faithful readiness of Definition 1144.41, and the L10 source readiness of Definition 1144.33. It binds exactly* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the sibling TruthCert rows consumed by the displayed packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedTruthCertUp.RealityConstrainedTruthCert scoped tastegate route`

Proof. Start with the carrier and field-coverage rows. They expose the source, signature, classifier, tower, stability, descent, invariant, ledger, failure, and local-name entries as the whole accepted TruthCert packet.

Apply sibling dependency and L10 source readiness. The open-fit, objectivity, explanation, physical-induction, tower-compression, and L10 source rows enter as displayed dependencies of that same packet, not as separate truth authorities.

Finally use the root TasteGate obligation together with root nontrivial and field-faithful readiness. These rows force the TasteGate read to distinguish packet presence at $S, \Sigma, K, T, U, D, I, L, F, N$ and to preserve the listed fields through BHist, hsame, Cont, Pkg, and NameCert. Hence the scoped surface is exactly the displayed route and has no hidden model–reality identity, observer privilege, unbounded confirmation, or off-packet truth export. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ packet is generated from finite source traces, reality-facing signatures, a classifier, an approximation tower, a continuation-stability row, a descent row, an observer-invariant row, a ledger row, a failure surface, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1144.13 and Definition 1144.45 bind the exported bridge-facing TruthCert packet, TasteGate route, root readiness rows, and failure-surface exactness over the displayed kernel rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealityConstrainedTruthCert}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.RealityConstrainedTruthCertUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert standard bridge checking, model–reality identity, unbounded confirmation, observer privilege, or truth export outside the displayed packet.

Upgrade path: No further theory-axis upgrade is asserted here; bridge and audit claims remain governed by the formal axis.

The $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ hub reads the methodology-ledger carrier through the explanation, model-audit, objectivity, and open-fit family and exports it toward chapter 1144 and chapter 1185.

Chapter 1145

A Concrete Naming Certificate for RealityConstrainedMethodologyLedger$^{\uparrow}$

The RealityConstrainedMethodologyLedger$^{\uparrow}$ packet realizes the scientific-methodology discipline of section 9414.46 as a finite BEDC NameCert carrier. It consumes the explanation rows of chapter 1136, the model-audit rows of chapter 1124, the multi-observer rows of chapter 1133, the tower-compression rows of chapter 1142, and the open-fit rows of chapter 1137. Those siblings separately expose explanation scope, fit audit, observer symmetry, compression residue, and open refutation surfaces; this packet binds them into the finite ledger by which a scientific-methodology claim is admitted only with effective-region bounds, idealization residue, domain-leakage refusal, upgrade routes, provenance, and local naming rows visible.

Definition 1145.1 (Reality-constrained methodology-ledger carrier)**.** *A* RealityConstrainedMethodologyLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$$

with the following displayed rows:

1. *X is the explanation row, recording the observation, model, phenomenon, signature-comparison, descent, residue, and failure-surface material inherited from* RealityConstrainedExplanation$^{\uparrow}$*;*
2. *A is the audit row, recording source traces, observed signatures, model-generated signatures, classifier comparisons, ledgers, failure surfaces, and strength rows inherited from* RealityConstrainedModelAudit$^{\uparrow}$*;*
3. *O is the objectivity row, recording multi-observer histories, anchor-change schedules, invariant packets, and no-global-observer refusals inherited from* RealityConstrainedObjectivity$^{\uparrow}$*;*
4. *T is the tower-compression row, recording source, indexed tower, observer tower, physical-fit tower, endpoint, composite ledger, replay, and provenance data inherited from* RealityConstrainedTowerCompressio
5. *I is the open-fit row, recording finite observation traces, admissible observation bundles, current fit, open continuation, and finite refutation surface inherited from* OpenFitPacket$^{\uparrow}$*;*
6. *S is the scope row, naming the domain, observation budget, admissible consumers, and physical-signature interface for the methodology claim;*
7. *D is the domain-leakage refusal row, naming attempted exports beyond S and the finite reason each export is refused;*

8. *U is the upgrade-route row, naming the additional observations, classifier refinements, bridge rows, or ledger entries needed to extend the claim without deleting the present carrier;*
9. *F is the methodology failure-surface row, collecting finite tests, signature mismatches, observer-anchor failures, tower-residue conflicts, and open-fit defeats;*
10. *H records componentwise* hsame *transport for the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, and failure rows;*
11. *C records the accepted consumer route through the displayed methodology ledger;*
12. *Q records* Pkg *provenance over the sibling carriers,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded explanation, final model truth, hidden observer privilege, endpoint/source identity, closed empirical fit, or methodological export outside the displayed packet.

Lean4 (definition) `BEDC.Derived.RealityConstrainedMethodologyLedgerUp.RealityConstrainedMethodologyLedgerC`

Definition 1145.2 (Reality-constrained methodology-ledger certificate)**.** *For accepted packets*

$$L = (X, A, O, T, I, S, D, U, F, H, C, Q, N) \quad \text{and} \quad L' = (X', A', O', T', I', S', D', U', F', H', C', Q', N'),$$

the certificate classifier $\mathsf{RealityConstrainedMethodologyLedgerClassifier}(L, L')$ *holds exactly when the explanation, audit, objectivity, tower-compression, and open-fit rows transport componentwise, the scope rows name the same admissible domain after* hsame *transport, the domain-leakage refusal rows preserve the refused exports, the upgrade-route rows preserve the admitted extension paths, the methodology failure surfaces preserve the defeating finite tests, and the consumer, provenance, and local naming rows transport through the same displayed route.*

Thus certificate agreement is not sameness of theories, models, or observers. It is preservation of the finite methodology ledger that records where the claim explains, where it fits, where it is objective, where it is compressed, where it remains open, where it refuses export, and how it may be extended.

Theorem 1145.3 (Reality-constrained methodology-ledger NameCert obligations)**.** *For the carrier of Definition 1145.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission: the accepted object is exactly $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$;*
2. *sibling accountability: explanation, audit, objectivity, tower-compression, and open-fit reads must pass through X, A, O, T, I;*
3. *scope exactness: every methodology claim must display the scope row S before it is exported;*
4. *domain-leakage refusal: every attempted export beyond S must be recorded in D as refused or routed through U as a declared extension path;*
5. *failure-surface exposure: finite defeats from explanation, audit, objectivity, tower compression, and open fit must appear in F before the consumer route C is accepted;*
6. *ledger non-escape: no consumer may infer scientific-methodology authority from rows outside $X, A, O, T, I, S, D, U, F, H, C, Q, N$.*

Lean4 ✓ BEDC.Derived.RealityConstrainedMethodologyLedgerUp.RealityConstrainedMethodologyLedgerCarrier namecert obligations

Proof. Carrier admission is projection from Definition 1145.1. The packet lists the only rows available to the local certificate.

Sibling accountability follows from the distinguished roles of X, A, O, T, I. Explanation scope, model audit, objectivity, tower compression, and open fit are not reconstructed by host prose; they enter through the five displayed sibling rows.

Scope exactness and domain-leakage refusal are forced by S and D. A methodology claim is admitted only after its domain and budget are named, and an attempted export beyond that boundary is either refused in D or sent to the declared extension route U.

Failure exposure is forced by F. The consumer route may read the methodology ledger only after the finite tests, signature mismatches, anchor failures, tower-residue conflicts, and open-fit defeats are visible.

Ledger non-escape follows because the carrier contains no coordinate outside $X, A, O, T, I, S, D, U, F, H, C, Q, N$ A route depending on an absent row is not a route in the accepted local naming certificate. □

Theorem 1145.4 (Reality-constrained methodology-ledger domain-leakage refusal)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. A claim that uses the explanation, audit, objectivity, tower-compression, or open-fit rows while exporting beyond the displayed scope row S is rejected unless the same packet records the attempted export in D and supplies an upgrade route in U.*

Lean4 ✓ BEDC.Derived.RealityConstrainedMethodologyLedgerUp.RealityConstrainedMethodologyLedgerDomainLea

Proof. Start with the scope-exactness and domain-leakage obligations of Definition 1145.3. They require the scope row before export and require every export beyond that scope to appear in the refusal or upgrade surface.

If a claim exports beyond S without a row in D, the packet lacks the displayed refusal record. If it also lacks a route in U, the packet gives no accepted extension path.

The carrier has no remaining coordinate through which the export could be licensed. The claim may remain a local explanation, audit read, objectivity read, tower-compression read, or open-fit read, but it is not a $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ methodology export. □

Theorem 1145.5 (Reality-constrained methodology-ledger scope-export exactness)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every methodology export must display the scope row S and either remain inside that displayed scope or route the attempted export through the refusal and upgrade rows D, U.*

Proof. Project the carrier of Definition 1145.1. The packet contains a distinguished scope row S, a domain-leakage refusal row D, and an upgrade-route row U; it contains no other export-authorizing coordinate.

Apply the NameCert obligations of Definition 1145.3. Scope exactness requires S before export, while domain-leakage refusal sends every attempted export beyond S through D or through the declared extension path U.

Finally use Definition 1145.4. A methodology claim that crosses the displayed scope without those rows is rejected as an export outside the accepted finite ledger. Hence the only accepted export routes are the in-scope route through S and the displayed beyond-scope route through D, U. □

Lean4 ✓ BEDC.Derived.RealityConstrainedMethodologyLedgerUp.RealityConstrainedMethodologyLedgerScopeExpo

Theorem 1145.6 (Reality-constrained methodology-ledger sibling independence)**.** *The* RealityConstrainedMethodo *carrier is not bijective to any one of* RealityConstrainedExplanation$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, RealityConstrainedObjectivity$^{\uparrow}$, RealityConstrainedTowerCompression$^{\uparrow}$, *or* OpenFitPacket$^{\uparrow}$. *Each sibling lacks at least one of the joined effective-region, idealization-residue, domain-leakage, upgrade-route, and methodology-ledger rows required by Definition 1145.1.*

Proof. Compare the carrier fields. RealityConstrainedExplanation$^{\uparrow}$ supplies explanation scope and residue but lacks the joined audit, objectivity, tower-compression, open-fit, domain-leakage, and upgrade-route rows.

RealityConstrainedModelAudit$^{\uparrow}$ supplies audit and failure rows but lacks the explanation, no-global-observer, tower-compression endpoint, open-continuation, methodology scope, and upgrade-route rows. RealityConstrainedObjectivity$^{\uparrow}$ supplies anchor-change and no-global-observer refusal but lacks the effective-region, idealization-residue, open-fit, and methodology-upgrade rows.

RealityConstrainedTowerCompression$^{\uparrow}$ supplies compression residue and endpoint/source separation but lacks the methodology domain-leakage refusal and open-fit refutation surface. OpenFitPacket$^{\uparrow}$ supplies finite fit and open continuation but lacks the joined explanation, audit, objectivity, tower-compression, and methodology-ledger rows.

Since each sibling omits at least one required field of L, no bijection can preserve all carrier fields of Definition 1145.1. □

Theorem 1145.7 (Reality-constrained methodology-ledger failure-surface exhaustion)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet. Every accepted methodology consumer routed by* C *must expose in* F *each finite defeating test available from the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, and upgrade rows before the route is accepted.*

Proof. The carrier of Definition 1145.1 displays F as the row collecting finite tests, signature mismatches, observer-anchor failures, tower-residue conflicts, and open-fit defeats. The certificate of Definition 1145.2 requires preservation of that row when two accepted packets agree as methodology ledgers.

Apply the failure-surface exposure clause of Definition 1145.3. It states that finite defeats from the sibling rows must appear in F before C is accepted.

The refusal theorem Definition 1145.4 closes the remaining route. A consumer that attempts to use an unexposed defeating test as if it were harmless either exports beyond the displayed scope or reads outside the admitted ledger, and neither route is licensed by D and U. □

Theorem 1145.8 (Reality-constrained methodology-ledger consumer-route exactness)**.** *For any accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, *the accepted consumer route factors exactly through the displayed rows* $X, A, O, T, I, S, D, U, F, H, C, Q, N$. *In particular, it cannot read hidden observer privilege, unbounded explanation, final model truth, closed empirical fit, or any other coordinate absent from the carrier.*

Proof. By Definition 1145.1, the packet has exactly the displayed coordinates and explicitly omits unbounded explanation, final model truth, hidden observer privilege, endpoint/source identity, closed empirical fit, and export outside the packet. Thus a consumer route that uses one of those omitted coordinates is not a route through L.

The certificate in Definition 1145.2 makes the same restriction invariant under accepted comparison: agreement preserves the displayed route through the sibling, scope, refusal, upgrade,

failure, transport, provenance, and local naming rows. The NameCert obligation theorem Definition 1145.3 records this as ledger non-escape.

Finally apply Definition 1145.4. Any alleged consumer authority that depends on a hidden observer, an unbounded explanation, a final truth coordinate, or a closed fit is an export beyond the displayed scope and must be refused unless it is rerouted through the admitted upgrade surface; without such a displayed row it is not accepted by C. □

Theorem 1145.9 (Reality-constrained methodology-ledger upgrade-route determinacy)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every accepted export beyond the scope row* S *is determined by the displayed refusal row* D *and upgrade row* U*; it is not determined by deleting, replacing, or bypassing the present carrier.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedMethodologyLedgerUp.RealityConstrainedMethodologyLedgerUpgradeRo`

Proof. The carrier definition Definition 1145.1 assigns attempted exports beyond S to D and admissible extension paths to U. No separate coordinate records an export by erasing the carrier or by appealing to an outside methodological authority.

The classifier in Definition 1145.2 preserves both rows. Therefore accepted comparison of methodology-ledger packets keeps the refused exports and admitted extension paths visible rather than allowing them to be reconstructed from hidden data.

Use the domain-leakage clause of Definition 1145.3 and the refusal theorem Definition 1145.4. Together they force every beyond-S export to be either refused in D or supplied with a route in U, so bypassing the present carrier cannot determine an accepted upgrade route. □

Theorem 1145.10 (Reality-constrained methodology-ledger root scope carrier admission)**.** *Every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *root read is admitted only through the displayed carrier rows*

$$X, A, O, T, I, S, D, U, F, H, C, Q, N.$$

No root consumer may replace those rows by an unbounded explanation, final model truth coordinate, hidden observer privilege, closed empirical fit, endpoint identity, or off-ledger methodology export.

Proof. Project the carrier of Definition 1145.1. It lists exactly the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and local naming rows.

Apply the carrier-admission and ledger-non-escape clauses of Definition 1145.3. The accepted root read must therefore expose those rows before it is a local $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ read.

The certificate of Definition 1145.2 preserves the same displayed rows under comparison. Hence any route using an absent coordinate is not a root read of the accepted methodology-ledger carrier. □

Theorem 1145.11 (Reality-constrained methodology-ledger root refusal upgrade coverage)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every beyond-scope methodology export from the root read is covered by the domain-leakage refusal row* D *or by the upgrade-route row* U*.*

Proof. Start with scope-export exactness, Definition 1145.5. It requires the scope row S before any methodology export and sends every export beyond S through the displayed refusal and upgrade rows.

The domain-leakage refusal theorem, Definition 1145.4, gives the local rejection clause. A claim that crosses the displayed scope without a refusal record in D and an extension route in U is not accepted as a methodology export.

Finally apply upgrade-route determinacy, Definition 1145.9. It rules out deleting, replacing, or bypassing the carrier as a way to determine an accepted beyond-scope export, leaving only the D and U coverage rows. □

Theorem 1145.12 (Reality-constrained methodology-ledger root failure consumer exhaustion)**.** *For any accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, every accepted consumer route must expose the failure row* F *before using the consumer route* C*.*

Proof. Use the failure-surface exposure clause of Definition 1145.3. It places the finite defeats from the explanation, audit, objectivity, tower-compression, and open-fit rows in F before the consumer route is accepted.

Apply failure-surface exhaustion, Definition 1145.7. The same exhaustion requirement covers defeating tests available from the scope, refusal, and upgrade rows, so the root consumer cannot postpone F until after C.

Finally use consumer-route exactness, Definition 1145.8. Since C factors through the displayed carrier rows and no hidden failure surface exists outside them, every accepted consumer route must expose F before it is read through C. □

Theorem 1145.13 (Reality-constrained methodology-ledger source-refusal boundary)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Any export that reads one of the sibling source rows* X, A, O, T, I *beyond the displayed scope row* S *must be witnessed in the refusal row* D *before the consumer route* C *may accept it.*

Proof. Project the carrier of Definition 1145.1. The sibling source rows, the scope row, the refusal row, and the consumer route are all displayed in the same finite packet, and there is no off-ledger coordinate from which a consumer can authorize a beyond-scope read.

Apply the scope exactness and domain-leakage clauses of Definition 1145.3. They force every methodology export to display S and force every attempted export beyond S into the refusal or extension surface.

Now use Definition 1145.4. A beyond-scope export that lacks its witness in D is rejected before it reaches C. Finally, Definition 1145.9 rules out deleting, replacing, or bypassing the carrier as a competing source for the same accepted export. □

Theorem 1145.14 (Reality-constrained methodology-ledger root audit source lock)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every root audit read of the methodology ledger must expose the sibling rows* X, A, O, T, I *and the kernel support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the consumer route* C *can treat the read as methodology authority.*

Proof. Begin with the carrier and certificate, Definition 1145.1 and Definition 1145.2. They place the explanation, audit, objectivity, tower-compression, and open-fit siblings in X, A, O, T, I and preserve them through the displayed comparison surface.

Apply the NameCert obligations of Definition 1145.3. The sibling-accountability and carrier-admission rows require those five sibling sources inside the same finite BHist packet, while the transport, continuation, provenance, and local naming rows are read through H, C, Q, N as hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

Finally use scope-export exactness and source-refusal boundary, Definition 1145.5 and Definition 1145.13. They prevent a root audit read from exporting sibling authority beyond S unless the same packet records the refusal and route rows. Thus the audit source is locked to X, A, O, T, I over the displayed kernel support. □

Theorem 1145.15 (Reality-constrained methodology-ledger root open-fit refusal route)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Any root read that uses the open-fit sibling row I as a public methodology refusal must route the read through the displayed sibling rows X, A, O, T, I, the refusal and failure rows D, F, and the* $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ *support of H, C, Q, N.*

Proof. Project the carrier. The open-fit material is one sibling row I of the same packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, not a separate public refusal oracle.

Use the NameCert obligations, scope-export exactness, and source-refusal boundary. The obligations require sibling accountability and failure-surface exposure, scope-export exactness requires S before public methodology use, and the source-refusal boundary sends any beyond-scope sibling read through the displayed refusal row D before C accepts it.

Now apply root open-fit refusal package and failure-route locality, Definition 1145.27 and Definition 1145.18. They keep the open-fit defeat in F and the refusal route in D inside the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, so the route cannot become an off-ledger empirical-fit authority. □

Theorem 1145.16 (Reality-constrained methodology-ledger root upgrade readiness)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, the root upgrade surface is ready exactly when the sibling rows X, A, O, T, I, the scope/refusal/failure rows S, D, F, the upgrade row U, and the* $\mathsf{BHist}/\mathsf{hsame}/\mathsf{Cont}/\mathsf{Pkg}/\mathsf{NameCert}$ *support H, C, Q, N are displayed together. No upgrade route may delete the present carrier or replace it by an external methodology source.*

Proof. First assemble the displayed packet from Definition 1145.1. The upgrade row U is part of $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, so it is read together with the sibling, scope, refusal, failure, transport, consumer, provenance, and local naming rows.

Apply upgrade-route determinacy and upgrade scope lock, Definition 1145.9 and Definition 1145.17. These results make every accepted extension a route through the displayed D, U surface and prevent the upgrade row from replacing the carrier.

Finally combine Definition 1145.14, Definition 1145.15, and Definition 1145.22. The audit lock keeps X, A, O, T, I visible, the open-fit refusal route keeps D, F visible for empirical defeats, and the closure package binds H, C, Q, N to hsame, Cont, Pkg, and $\mathsf{NameCert}$. Hence upgrade readiness is exactly the displayed root surface and not an external methodology authority. □

Theorem 1145.17 (Reality-constrained methodology-ledger upgrade scope lock)**.** *Let $L = (X, A, O, T, I, S, D, U, I$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. An accepted upgrade row U preserves the same displayed packet*

$$X, A, O, T, I, S, D, U, F, H, C, Q, N$$

by constraining U to the declared extension route; it does not replace the carrier, erase S or D, or create a second consumer packet for the methodology claim.

Proof. The carrier definition Definition 1145.1 makes U one row of the displayed packet rather than a substitute carrier. Thus an upgrade route is read together with the explanation, audit,

objectivity, tower-compression, open-fit, scope, refusal, failure, transport, consumer, provenance, and local naming rows.

Apply Definition 1145.3. The obligation surface demands carrier admission, sibling accountability, scope exactness, domain-leakage refusal, failure exposure, and ledger non-escape for the same packet whenever a methodology claim is accepted.

The refusal theorem Definition 1145.4 and upgrade-route determinacy Definition 1145.9 then close the route. They allow U to declare how the claim may be extended, but they do not allow U to delete the existing scope, refusal, failure, or consumer rows. □

Theorem 1145.18 (Reality-constrained methodology-ledger failure-route locality)**.** *Let $L = (X, A, O, T, I, S, D, U$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Defeating tests recorded in the failure row F remain local to the displayed sibling, scope, refusal, upgrade, transport, consumer, provenance, and local naming rows. They cannot be converted by C into unbounded model truth, hidden observer privilege, final explanation authority, or a closed empirical oracle.*

Proof. Read the carrier of Definition 1145.1. The failure row F is one component of the finite methodology ledger, and the omitted coordinates include unbounded explanation, final model truth, hidden observer privilege, and closed empirical fit.

Apply Definition 1145.3. The failure-surface exposure clause requires defeating tests to appear in F before C is accepted, while ledger non-escape prevents C from reading authority outside the displayed carrier.

Use Definition 1145.4 for any attempt to export those defeating tests beyond the displayed scope. Finally, Definition 1145.9 confines accepted beyond-scope movement to the displayed refusal and upgrade rows, so no defeating test in F becomes an unbounded truth or hidden-observer privilege. □

Theorem 1145.19 (Reality-constrained methodology-ledger route normal form)**.** *Every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *consumer route factors through exactly*

$$X, A, O, T, I, S, D, U, F, H, C, Q, N$$

before any methodology export is read.

Proof. Project the carrier of Definition 1145.1. It lists exactly the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and local naming rows.

Apply the NameCert obligations of Definition 1145.3. The carrier-admission, sibling-accountability, scope-exactness, failure-exposure, and ledger-non-escape clauses require the same rows before the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ route accepts a methodology export.

Finally use consumer-route exactness, Definition 1145.8. Since the accepted consumer route has no coordinate outside the displayed carrier, every methodology export factors through the listed route and no shorter host route is available.

The failure and upgrade coordinates are fixed by Definition 1145.7 and Definition 1145.9. Thus the normal form does not hide defeating tests after C and does not move beyond S except through the displayed D, U rows. □

Theorem 1145.20 (Reality-constrained methodology-ledger sibling failure coverage)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Finite defeating tests from the explanation, audit, objectivity, tower-compression, and open-fit sibling rows must be collected in F before C is accepted.*

Proof. Begin with the carrier and certificate, Definition 1145.1 and Definition 1145.2. They make F the displayed row preserving finite tests, signature mismatches, observer-anchor failures, tower-residue conflicts, and open-fit defeats.

Apply the failure-surface exhaustion theorem, Definition 1145.7. It requires the defeating tests available from the sibling, scope, refusal, and upgrade rows to appear in F before the consumer route is accepted.

The NameCert obligations and consumer-route exactness, Definition 1145.3 and Definition 1145.8, make this coverage a condition on the same accepted route C, not an external audit of a different carrier.

Use failure-route locality, Definition 1145.18, to close the remaining route. A defeating test not collected in F cannot be converted by C into a public methodology export, because such a conversion would read outside the displayed local ledger.

Finally apply Definition 1145.9. An attempted escape through an upgrade route remains displayed in U and cannot remove the prior requirement that sibling defeats have already been collected in F. □

Theorem 1145.21 (Reality-constrained methodology-ledger refusal-upgrade noncollapse)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the refusal row* D *and upgrade row* U *remain separate displayed rows. They cannot collapse into a silent methodology export outside the scope row* S*.*

Proof. Read the carrier definition. The refusal row D records attempted exports beyond S, while the upgrade row U records admitted extension paths; neither row is identified with the consumer route C or with the scope row S.

Apply the NameCert obligations and scope-export exactness, Definition 1145.3 and Definition 1145.5. They require the scope row before export and force any beyond-scope movement to remain visible in the refusal and upgrade surface.

Use failure-surface exhaustion and consumer-route exactness, Definition 1145.7 and Definition 1145.8. The consumer route cannot accept by hiding failures, and it cannot read a collapsed D, U authority outside the displayed carrier.

Finally use upgrade-route determinacy, Definition 1145.9. An accepted route beyond S is determined by the displayed D, U rows, not by a silent export outside S and not by collapsing the two rows into one hidden methodology authority. □

Theorem 1145.22 (Reality-constrained methodology-ledger obligation closure package)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the finite obligation-closure surface is exactly the carrier, certificate, NameCert obligation rows, domain-leakage refusal, scope-export exactness, failure-surface exhaustion, consumer-route exactness, upgrade-route determinacy, and failure-route locality displayed in this chapter. Thus every methodology-ledger consumer must factor through* $X, A, O, T, I, S, D, U, F, H, C, Q, N$ *with the sibling, scope, refusal, upgrade, failure, transport, consumer, provenance, and local* $\mathsf{NameCert}$ *rows visible.*

Proof. Begin with the carrier and certificate, Definition 1145.1 and Definition 1145.2. They provide the displayed packet and the classifier preserving the sibling, scope, refusal, upgrade, failure, consumer, provenance, and local naming rows.

Next apply the NameCert obligations, Definition 1145.3. Those obligations expose carrier admission, sibling accountability, scope exactness, domain-leakage refusal, failure-surface exposure, and ledger non-escape for the same packet.

Now use the local route theorems: Definition 1145.4, Definition 1145.5, Definition 1145.7, and Definition 1145.8. They force exported methodology authority, defeating tests, and consumer reads through the displayed rows.

Finally apply upgrade-route determinacy and failure-route locality, Definition 1145.9 and Definition 1145.18. They close the remaining beyond-scope and failure-consumer paths without adding an off-ledger carrier. □

Theorem 1145.23 (Reality-constrained methodology-ledger root upgrade refusal totality)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every root export beyond the displayed scope row* S *is exhausted by the refusal row* D *and the upgrade row* U*. Such an export cannot bypass the carrier, delete the present scope, or appeal to an unbounded explanation, final model truth, hidden observer privilege, closed empirical fit, endpoint identity, or off-ledger methodology authority.*

Proof. Start with root refusal upgrade coverage, Definition 1145.11. It places every beyond-scope root export in the displayed D, U surface.

Apply scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. Together they require S before export and reject an attempted crossing that lacks the refusal or upgrade rows.

Finally use upgrade-route determinacy and upgrade scope lock, Definition 1145.9 and Definition 1145.17. They make U a row of the same carrier rather than a replacement carrier, so no bypass route can authorize the root export. □

Theorem 1145.24 (Reality-constrained methodology-ledger failure consumer public scope)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the consumer route* C *is not accepted before the failure row* F *and the displayed sibling, scope, refusal, and upgrade rows* X, A, O, T, I, S, D, U *are visible. Public methodology use therefore reads* C *only after the finite failure surface has been exposed inside the same local carrier.*

Proof. First apply root failure consumer exhaustion, Definition 1145.12. It requires the failure row F before the consumer route C can be read.

Next use failure-surface exhaustion and failure-route locality, Definition 1145.7 and Definition 1145.18. They keep the defeating tests local to the displayed sibling, scope, refusal, upgrade, transport, consumer, provenance, and naming rows.

Finally apply consumer-route exactness and the obligation closure package, Definition 1145.8 and Definition 1145.22. These make C a route through the same finite carrier, with no public consumer path that can hide F or omit the visible sibling, scope, refusal, or upgrade rows. □

Theorem 1145.25 (Reality-constrained methodology-ledger root methodology export totality)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, every methodology export factors through the displayed scope row* S*, refusal row* D*, upgrade row* U*, failure row* F*, consumer row* C*, provenance row* Q*, and local* $\mathsf{NameCert}$ *row* N*. The root methodology surface therefore has no off-ledger export coordinate, hidden observer privilege, final truth source, or closed empirical-fit authority.*

Proof. Start with the existing NameCert obligations, Definition 1145.3. They bind carrier admission, sibling accountability, scope exactness, refusal, failure exposure, consumer non-escape, provenance, and local naming to the same finite packet.

Apply scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. These force methodology export through S and route any beyond-scope attempt through the displayed refusal and upgrade surface.

Next use failure-surface exhaustion and consumer-route exactness, Definition 1145.7 and Definition 1145.8. They require finite defeats in F before a public methodology read reaches C, with no consumer authority outside the displayed ledger rows.

Finally combine upgrade-route determinacy with the obligation-closure package, Definition 1145.9 and Definition 1145.22. The accepted export remains a route inside S, D, U, F, C, Q, N, so no hidden authority can separate it from the local methodology ledger. □

Theorem 1145.26 (Reality-constrained methodology-ledger root TruthCert handoff)**.** *Any TruthCert-facing read of an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet uses the displayed explanation, audit, objectivity, tower-compression, open-fit, refusal, upgrade, and failure rows before it reaches methodology consumption. Such a read does not appeal to final truth, hidden observer privilege, endpoint identity, or an unbounded external validation row.*

Proof. First apply the NameCert obligation surface. By Definition 1145.3, an accepted packet exposes the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and naming rows together.

Use scope-export exactness and domain-leakage refusal next, Definition 1145.5 and Definition 1145.4. They prevent a TruthCert-facing read from crossing the displayed methodology scope unless the refusal and upgrade rows are visible.

Now apply failure-surface exhaustion and consumer-route exactness, Definition 1145.7 and Definition 1145.8. The read reaches C only after the defeating tests in F have been displayed, so it remains an auditable methodology handoff rather than final truth.

Finally use upgrade-route determinacy and the obligation-closure package. They keep any allowed extension inside the same ledger carrier and close the handoff through the declared rows, excluding hidden observer privilege and external TruthCert authority. □

Theorem 1145.27 (Reality-constrained methodology-ledger root open-fit refusal package)**.** *Every open-fit defeat in an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet is routed through the failure row F and the refusal row D before any public methodology consumer reads C. Thus an open-fit failure cannot become a consumer-facing methodology authority unless the displayed defeat and refusal package has already been exposed in the same local ledger.*

Proof. Begin with the NameCert obligations and scope-export exactness, Definition 1145.3 and Definition 1145.5. They place the open-fit row inside the same carrier and require the displayed scope row before methodology export.

Apply domain-leakage refusal and failure-surface exhaustion, Definition 1145.4 and Definition 1145.7. The first routes beyond-scope claims through D, while the second places the open-fit defeat in F before it can affect any consumer route.

Next use consumer-route exactness. By Definition 1145.8, the public consumer C reads only the displayed sibling, scope, refusal, upgrade, failure, transport, provenance, and naming rows.

Finally apply upgrade-route determinacy and the obligation-closure package, Definition 1145.9 and Definition 1145.22. They keep any open-fit response inside the same D, F, C package, so no public methodology consumer can read the defeat as an off-ledger authorization. □

Theorem 1145.28 (Reality-constrained methodology-ledger scope refusal triad)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, any*

methodology export must pass through the displayed scope row S, refusal row D, and upgrade row U. The export cannot be read from the explanation, model-audit, objectivity, tower-compression, or open-fit siblings unless those three methodology rows are visible in the same carrier.

Proof. Use the carrier and NameCert obligations, Definition 1145.1 and Definition 1145.3. They place the sibling rows and methodology rows inside one finite packet and make S, D, U part of the local certificate surface.

Apply scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. The first requires S for methodology export, while the second routes any beyond-scope assertion through D.

Finally use upgrade-route determinacy, Definition 1145.9. It keeps U in the same carrier, so no sibling row can replace the displayed scope-refusal-upgrade triad. □

Theorem 1145.29 (Reality-constrained methodology-ledger failure upgrade triad)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the failure row F is visible before any upgrade route can be accepted by the consumer row C. Thus a consumer cannot read an upgrade as methodology authority unless the same local packet exposes the defeating tests that constrain the upgrade.*

Proof. Start with failure-surface exhaustion, Definition 1145.7. It places the finite defeating tests in row F as part of the accepted methodology-ledger surface.

Next apply upgrade-route determinacy and upgrade scope lock, Definition 1145.9 and Definition 1145.17. They make U a route inside the displayed carrier and keep it tied to the same scope and refusal rows.

Finally use consumer-route exactness and failure-route locality, Definition 1145.8 and Definition 1145.18. These force C to read the visible failure and upgrade rows together, with no consumer-side shortcut that hides F. □

Theorem 1145.30 (Reality-constrained methodology-ledger sibling audit triad)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the sibling rows X, A, O, T, I are jointly accountable to the methodology ledger. They cannot be replaced by a single explanation, model-audit, objectivity, tower, or open-fit sibling without losing at least one displayed accountability row.*

Proof. Project the carrier and NameCert obligations, Definition 1145.1 and Definition 1145.3. They list X, A, O, T, I as separate sibling rows and bind each one to the same local certificate.

Use scope-export exactness, failure-surface exhaustion, and consumer-route exactness, Definition 1145.5, Definition 1145.7, and Definition 1145.8. Together they require the sibling surface to remain visible before export, failure reading, and consumer reading.

The independence witness then separates the five siblings by the rows they do not jointly carry. Since the carrier has no collapse coordinate for replacing X, A, O, T, I with one sibling, the audit remains a joint row-wise obligation. □

Theorem 1145.31 (Reality-constrained methodology-ledger public obligation surface)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the public obligation surface consists exactly of the refusal, upgrade, failure-surface, consumer-route, prediction, sibling-audit, TruthCert handoff, open-fit refusal,* hsame, Cont, Pkg*, and local* NameCert *rows displayed in this chapter. It exports no domain-truth oracle, no physical-model adequacy theorem, no hidden fit selector, and no external methodology primitive.*

Proof. First assemble the refusal and upgrade rows. Apply Definition 1145.4 and Definition 1145.11. They force every beyond-scope methodology export to pass through the displayed domain-leakage refusal row D and accepted route row U.

Next add the public consumer and failure rows. Use Definition 1145.24. It requires the failure surface and consumer route to be read together before the packet can be consumed as a methodology-ledger export.

The TruthCert and open-fit interfaces are then local handoffs, not external sources. Apply Definition 1145.26 and Definition 1145.27; these keep TruthCert and open-fit readings inside the same refusal, route, package, and naming rows.

Finally use Definition 1145.30. It keeps X, A, O, T, I as accountable sibling rows and prevents their replacement by a domain-truth oracle, physical-model adequacy theorem, hidden fit selector, or external methodology primitive. The remaining hsame, Cont, Pkg, prediction, and local NameCert rows are already the displayed support rows of the same packet, so the public surface is exactly the listed finite interface. □

Theorem 1145.32 (Reality-constrained methodology-ledger root scope admission)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, a root methodology read is admitted only after the scope row* S*, domain-leakage refusal row* D*, upgrade-route row* U*, failure-surface row* F*, consumer row* C*, package row* Q*, and local* NameCert *row* N *are visible in the same carrier.*

Proof. Start from the carrier and NameCert obligations, Definition 1145.1 and Definition 1145.3. They make S, D, U, F, C, Q, N displayed coordinates of the accepted packet, not external methodology premises.

Apply scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. These force the root read to expose its scope and to route every beyond-scope attempt through the displayed refusal and upgrade surface.

Next use failure-route locality and consumer-route exactness, Definition 1145.18 and Definition 1145.8. They keep the failure row and consumer route inside the same finite ledger.

Finally apply the public obligation surface and upgrade-route determinacy, Definition 1145.31 and Definition 1145.9. The root read is therefore admitted only when S, D, U, F, C, Q, N are all visible. □

Theorem 1145.33 (Reality-constrained methodology-ledger root sibling accountability)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, root methodology admission keeps the five sibling rows* X, A, O, T, I *accountable as separate rows. No root read may collapse the explanation, audit, objectivity, tower-compression, and open-fit rows into a single sibling carrier.*

Proof. Project the carrier and local obligation surface. The definition displays X, A, O, T, I separately, while Definition 1145.3 identifies sibling accountability as one of the certificate rows.

Use the sibling-audit triad, Definition 1145.30. It states directly that the five siblings cannot be replaced by a single explanation, model-audit, objectivity, tower, or open-fit row.

Now route the public read through scope-export exactness, failure-route locality, and consumer-route exactness. These keep the root read inside the same S, D, U, F, C methodology surface and prevent a consumer shortcut from hiding one sibling behind another.

The public obligation surface then records the same accountability at the root boundary. Since no coordinate of L carries a collapsed sibling row, the root admission preserves X, A, O, T, I as separate accountable entries. □

Theorem 1145.34 (Reality-constrained methodology-ledger root export coverage)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, every beyond-scope methodology export is covered by the domain-leakage refusal row* D *or the upgrade-route row* U*. Every public consumer route must expose the failure-surface row* F *before it can be accepted as the consumer row* C*.*

Proof. Begin with scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. Together they send every export beyond S through the displayed D, U surface.

Apply upgrade-route determinacy. An accepted route beyond the displayed scope is therefore a route named by U against the refusal surface D, not an off-ledger methodology export.

Next use failure-route locality and the failure-consumer public-scope theorem, Definition 1145.18 and Definition 1145.24. They require finite defeats to remain visible in F before public methodology consumption.

Finally invoke the public obligation surface. It records D, U, F, C, Q, N as the root export package, so every public consumer route exposes F before it reads C. □

Falsifiable prediction: A carrier with no domain-leakage refusal row cannot support the methodology-ledger NameCert obligations.

Independence witness: realityconstrainedexplanation, realityconstrainedmodelaudit, realityconstrainedobjectivity, realityconstrainedtowercompression, openfitpacket: each sibling lacks at least one of the joined effective-region, idealization-residue, domain-leakage, upgrade-route, and methodology-ledger rows, so no bijection preserves all carrier fields.

The Lean TasteGate file supplies `Nontrivial RealityConstrainedMethodologyLedgerUp` and `FieldFaithful RealityConstrainedMethodologyLedgerUp` instances in `lean4/BEDC/Derived/RealityConstra`

Theorem 1145.35 (Reality-constrained methodology-ledger refusal upgrade totality)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every refusal-facing or upgrade-facing methodology read anchored in the finite* BHist *carrier is exhausted by the displayed scope row* S*, refusal row* D*, upgrade row* U*, componentwise* hsame *transport* H*, consumer route* C*,* Pkg *provenance* Q*, and local* $\mathsf{NameCert}$ *row* N*. No such read can pass through unbounded explanation, final model truth, hidden observer privilege, closed empirical fit, endpoint identity, or an off-ledger methodology source.*

Proof. Begin with the carrier and local obligation surface in Definition 1145.1 and Definition 1145.3. They place S, D, U, H, C, Q, N inside the same finite BHist packet and require scope exactness, domain-leakage refusal, failure exposure, and ledger non-escape before methodology authority is consumed.

Next use scope-export exactness, domain-leakage refusal, upgrade-route determinacy, and upgrade scope lock. These theorems force every beyond-scope movement to be either refused in D or routed through U, while keeping the route attached to the same hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows.

Finally apply the root refusal and export coverage theorems. They identify the root read with the displayed carrier surface rather than with a hidden external methodology source. Since the carrier has no coordinate for unbounded explanation, final truth, hidden observer privilege, closed fit, or endpoint identity, the refusal and upgrade rows are total for the accepted read. □

Theorem 1145.36 (Reality-constrained methodology-ledger failure consumer coverage)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet.*

Every consumer route C that reads methodology authority from the finite BHist *ledger must expose the failure row F, the sibling rows X, A, O, T, I, the refusal and upgrade rows D, U, componentwise* hsame *transport H,* Pkg *provenance Q, and local* NameCert *row N before the consumer is accepted.*

Proof. Project the carrier of Definition 1145.1. The explanation, audit, objectivity, tower-compression, open-fit, refusal, upgrade, failure, transport, consumer, provenance, and naming rows are displayed together in one packet.

Apply the NameCert obligations of Definition 1145.3. The sibling-accountability and failure-surface clauses put the defeating tests in F before the consumer route is accepted, while ledger non-escape prevents C from importing hidden authority.

Now use failure-surface exhaustion, failure-route locality, and consumer-route exactness. These results keep finite defeats and consumer reads inside the same BHist/hsame/Cont/Pkg/NameCert support and route them through $X, A, O, T, I, D, U, F, H, C, Q, N$ rather than through an external audit.

The public failure-consumer and root export coverage theorems close the root case. They require the failure surface before public methodology consumption, so every accepted consumer route is covered by the displayed failure and methodology rows. □

Theorem 1145.37 (Reality-constrained methodology-ledger prediction row non-escape)**.** *For every accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, any falsifiable-prediction read of the methodology ledger is anchored in the same finite* BHist *carrier. It may use only the sibling rows X, A, O, T, I, scope row S, refusal row D, upgrade row U, failure row F, componentwise* hsame *transport H, consumer route C,* Pkg *provenance Q, and local* NameCert *row N; it cannot become a domain-truth oracle, physical-model adequacy theorem, hidden fit selector, or off-ledger methodology primitive.*

Proof. Start from the carrier and certificate, Definition 1145.1 and Definition 1145.2. They make the prediction-facing read a route through the displayed methodology ledger, with componentwise transport and package provenance preserving the same rows.

Apply the NameCert obligation surface in Definition 1145.3. It requires scope exactness, refusal coverage, failure exposure, and non-escape before a methodology read reaches the consumer route.

Finally use the public obligation surface, sibling audit triad, and failure-route locality. These rows keep prediction, sibling accountability, failure exposure, hsame, Cont, Pkg, and local NameCert support inside L, and they exclude a domain-truth oracle, physical-model adequacy theorem, hidden fit selector, or off-ledger methodology primitive from the accepted prediction row. □

Theorem 1145.38 (Reality-constrained methodology-ledger formal-target route totality)**.** *For every accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, a methodology-ledger formal-target read must pass through the displayed refusal-upgrade surface D, U, failure-consumer surface F, C, and prediction row before any truth-certificate consumer can read the methodology packet. The route remains inside the same finite* BHist/hsame/Cont/Pkg/NameCert *support and exports no domain-truth oracle, physical-model adequacy theorem, hidden fit selector, or off-ledger methodology primitive.*

Proof. First apply refusal upgrade totality, Definition 1145.35. It forces every beyond-scope methodology read through the displayed refusal row D and upgrade row U before the packet can be consumed.

Next use failure consumer coverage, Definition 1145.36. This places the failure row F and consumer route C in the same accepted packet as the sibling, transport, provenance, and local naming rows.

Finally apply prediction row non-escape, Definition 1145.37. The prediction-facing read is still a finite methodology-ledger read, so a truth-certificate consumer receives the packet only after the refusal-upgrade, failure-consumer, and prediction surfaces are all displayed. □

Theorem 1145.39 (Reality-constrained methodology-ledger TasteGate readback surface)**.** *For every displayed* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the rows shown in the paper carrier are the same rows decoded by the checked TasteGate readback theorem* `RealityConstrainedMethodologyLedgerTasteGate_single_carrier_alignment`*. The encoding route reads each* BHist *row as a finite* BMark *event, decodes it back to the same* BHist *row, and reconstructs the single* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *carrier rather than a second methodology object.*

Proof. Project L into its thirteen displayed rows. The TasteGate readback theorem proves the BHist encode-decode identity for each row and the round-trip law from the event-flow presentation back to the same ledger packet.

Apply the injectivity component of the same checked theorem. If two event-flow readbacks agree, the underlying $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ packets agree, so the paper rows and the decoded rows are not parallel carriers. They are the same finite carrier read through the TasteGate surface. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedMethodologyLedgerUp.taste gate`

Theorem 1145.40 (Reality-constrained methodology-ledger nontrivial field-faithful readback)**.** *The paper-side falsifiable-prediction row and independence witness are witnessed on the same* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *carrier by the Lean* `Nontrivial` *and* `FieldFaithful` *instances. The* `Nontrivial` *instance separates the all-empty packet from the packet whose* X *row is* $e0(\emptyset)$*, and the* `FieldFaithful` *instance says equality of the displayed row list* $[X, A, O, T, I, S, D, U, F, H, C, Q, N]$ *determines equality of the packet.*

Proof. Use the `Nontrivial` instance for the falsifiable-prediction witness. It supplies two packets in the same carrier and proves they are distinct by constructor separation at the X row.

Use the `FieldFaithful` instance for row independence. Its field map is exactly the displayed row list, and its faithfulness clause says that no hidden coordinate remains after those rows agree. Thus the prediction witness and the field-faithfulness readback live on the carrier already named by the paper certificate. □

Theorem 1145.41 (Reality-constrained methodology-ledger formal-target closure route)**.** *The refusal-upgrade surface, failure-consumer surface, prediction row, and TasteGate readback surface together give a formal-target route for later closure promotion of* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$*. The route is confined to the displayed finite carrier, its* BHist/hsame/Cont/Pkg *support, its local* NameCert *obligations, and the checked TasteGate readback; it does not claim public closure, bridge closure, physical-model adequacy, or a domain-truth oracle.*

Proof. Apply Definition 1145.35 and Definition 1145.36. These surfaces route refusal, upgrade, failure exposure, and consumer access through the displayed methodology-ledger rows.

Apply Definition 1145.37 and Definition 1145.39. The prediction read remains inside the same finite ledger, and the TasteGate readback identifies the event-flow decoding with the displayed carrier. This is a formal-target route for a later closurestatus decision, not itself a claim of public or bridge closure. □

Theorem 1145.42 (Reality-constrained methodology-ledger methodology root triad)**.** *For every accepted root methodology read of a* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C$ *the read exposes the sibling rows* X, A, O, T, I*, the scope-refusal-upgrade rows* S, D, U*, and the failure-consumer-provenance-name rows* F, C, Q, N *inside the same finite carrier. The transport row* H *remains part of the same* $\mathsf{BHist/hsame/Cont/Pkg}$ *support, so the root read cannot replace these rows by a hidden methodology source, final truth route, observer-privilege route, or off-ledger NameCert.*

Proof. Start with the carrier and certificate, Definition 1145.1 and Definition 1145.2. They display L as a single finite packet and make its classifier read the sibling, methodology, failure, transport, consumer, provenance, and naming rows through the same local support.

Apply the NameCert obligations and root audit source lock, Definition 1145.3 and Definition 1145.14. These obligations expose carrier admission, sibling accountability, source lock, provenance, and local naming before a root methodology read can be accepted.

Now add the route constraints. Scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4, force the S, D surface to be visible, while upgrade-route determinacy, Definition 1145.9, keeps U inside the same carrier.

Finally use failure-surface exhaustion and consumer-route exactness, Definition 1145.7 and Definition 1145.8. They force F and C to be read with the provenance and naming rows already carried by the packet, so the root methodology triad is a row-wise readback of L rather than a second authority. □

Theorem 1145.43 (Reality-constrained methodology-ledger scope-refusal cover)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, every beyond-scope methodology export is covered by the displayed scope row* S*, refusal row* D*, and upgrade row* U *before any consumer route* C *may treat the export as methodology authority. No consumer route can read a beyond-scope export directly from the sibling, failure, provenance, or local* $\mathsf{NameCert}$ *rows.*

Proof. Begin with scope-export exactness, Definition 1145.5. It requires the displayed scope row S before a methodology export can be read.

Next apply domain-leakage refusal and upgrade-route determinacy, Definition 1145.4 and Definition 1145.9. They place a beyond-scope attempt in D and any admitted continuation in U, with both rows still belonging to the same packet.

Use the NameCert obligations to keep the consumer read local to the carrier. The certificate does not provide a separate consumer authority outside S, D, U, F, H, C, Q, N.

Finally apply failure-surface exhaustion, consumer-route exactness, and root export coverage, Definition 1145.7, Definition 1145.8, and Definition 1145.34. These results require F before C and identify the root export package with the displayed refusal, upgrade, failure, consumer, provenance, and naming rows. Hence a consumer route can treat a beyond-scope export as methodology authority only after the S, D, U cover has already been exposed. □

Theorem 1145.44 (Reality-constrained methodology-ledger failure-upgrade readback)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, failure rows and upgrade rows are read back through the same finite carrier. The readback cannot split* F *into hidden truth, cannot split* U *into observer privilege, and cannot route either row through an off-ledger methodology object before consumer access.*

Proof. Project the packet through the carrier and certificate. The rows F and U are coordinates of the same $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ object, with H, C, Q, N supplying the transport, consumer, provenance, and local naming support.

Apply failure-surface exhaustion and failure-route locality, Definition 1145.7 and Definition 1145.18. They keep defeating tests in F and prevent a consumer-facing route from turning those tests into hidden truth.

Next apply upgrade-route determinacy and upgrade scope lock, Definition 1145.9 and Definition 1145.17. They keep any admitted upgrade inside U and tie it to the same displayed scope and refusal surface, not to observer privilege.

Finally combine consumer-route exactness with root audit source lock, Definition 1145.8 and Definition 1145.14. The consumer reads the visible failure and upgrade rows through the same carrier, provenance, and local NameCert surface, so no split into a hidden truth route or observer-privilege route is available. □

Theorem 1145.45 (Reality-constrained methodology-ledger public export readiness)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the public export surface is the finite methodology-ledger interface carried by the explanation row* X*, audit row* A*, objectivity row* O*, tower-compression row* T*, open-fit row* I*, scope row* S*, refusal row* D*, upgrade row* U*, failure row* F*, transport row* H*, consumer row* C*, provenance row* Q*, and local* NameCert *row* N*. Any public export of methodology authority must pass through those rows, the existing failure-consumer coverage, the formal-target route totality surface, and the TasteGate readback surface before it can be consumed.*

Proof. Project the carrier using Definition 1145.1. The displayed coordinates are exactly the five sibling rows X, A, O, T, I together with the methodology scope, refusal, upgrade, failure, transport, consumer, provenance, and local naming rows.

Apply the NameCert obligations, Definition 1145.3. They bind carrier admission, scope exactness, refusal, upgrade, failure exposure, sibling accountability, and ledger non-escape to the same local certificate surface.

Next use failure-consumer coverage and formal-target route totality, Definition 1145.36 and Definition 1145.38. These results place the failure row and consumer route before public methodology consumption and keep TruthCert-facing access inside the same BHist/hsame/Cont/Pkg/NameCert support.

Finally apply the TasteGate readback surface, Definition 1145.39. It identifies the event-flow decoding with the displayed carrier, so the public export is a readback of L rather than an off-ledger methodology object. □

Theorem 1145.46 (Reality-constrained methodology-ledger TruthCert consumer handoff)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, any TruthCert-facing consumer handoff receives the packet only after the carrier rows, local* NameCert *obligations, failure-consumer coverage, formal-target route totality, and TasteGate readback have displayed the same finite methodology ledger. The handoff cannot replace the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, or local NameCert rows by a domain-truth oracle or a physical-model adequacy theorem.*

Proof. Begin with the root TruthCert handoff theorem, Definition 1145.26. It already forces a TruthCert-facing read through the local methodology rows instead of through an external truth coordinate.

Use public export readiness, Definition 1145.45, to identify the available consumer surface. The handoff receives the same finite packet rows and no additional methodology primitive.

Now apply the formal-target route totality theorem. It puts refusal-upgrade, failure-consumer, and prediction-facing reads before TruthCert consumption and keeps all of them in the displayed BHist/hsame/Cont/Pkg/NameCert support.

Finally use the TasteGate readback surface. Since the decoded event-flow carrier is the same L, a TruthCert-facing consumer handoff reads the public methodology ledger without replacing it by a separate domain-truth or adequacy claim. □

Theorem 1145.47 (Reality-constrained methodology-ledger root dependency totality)**.** *For every accepted root methodology consumer of a* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the consumer reads the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and local NameCert rows before any TruthCert consumer handoff. In particular, the five sibling dependencies* RealityConstrainedExplanation$^{\uparrow}$*,* RealityConstrainedModelAudit$^{\uparrow}$*,* RealityConstrainedObjectivity$^{\uparrow}$*,* RealityConstrainedTowerCompression$^{\uparrow}$*, and* OpenFitPacket$^{\uparrow}$ *remain displayed as rows of the same finite ledger.*

Proof. Start from the methodology root triad, Definition 1145.42. It exposes the five sibling rows, the scope-refusal-upgrade rows, and the failure-consumer-provenance-name rows inside one carrier.

Apply root sibling accountability and the public obligation surface, Definition 1145.33 and Definition 1145.31. These keep the explanation, audit, objectivity, tower-compression, and open-fit dependencies visible as rows of L rather than as hidden sources.

Next use failure-consumer public scope, Definition 1145.24. It requires the failure row and displayed sibling, scope, refusal, and upgrade rows before the consumer route C may be read.

Finally apply the TruthCert consumer handoff theorem, Definition 1145.46. The handoff therefore occurs only after the root consumer has read every row listed in the carrier and the five sibling dependencies remain part of the same public methodology ledger. □

Theorem 1145.48 (Reality-constrained methodology-ledger export accountability)**.** *For every accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, any methodology export from* L *must expose the sibling rows* X, A, O, T, I*, the scope row* S*, the refusal and upgrade rows* D, U*, the failure row* F*, and the support rows* H, C, Q, N*. The export is therefore a read of the displayed* BHist/hsame/Cont/Pkg/NameCert *surface, not a read of an omitted methodology coordinate.*

Proof. Project the carrier of Definition 1145.1. It displays the five sibling rows, the scope-refusal-upgrade surface, the failure row, and the transport, consumer, provenance, and naming rows in one finite packet.

Apply the NameCert obligations of Definition 1145.3. Carrier admission, sibling accountability, scope exactness, domain-leakage refusal, failure exposure, and ledger non-escape all bind the same displayed rows before a methodology claim is accepted.

Finally use public export readiness and sibling independence, Definition 1145.45 and Definition 1145.6. They keep the export on the public carrier surface and prevent any single sibling from replacing the full methodology ledger. □

Theorem 1145.49 (Reality-constrained methodology-ledger upgrade-refusal totality)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet. Every methodology export beyond the displayed scope row* S *is either refused in the domain-leakage*

row D or routed through the upgrade row U before any consumer route C may treat it as methodology authority.

Proof. Begin with scope-export exactness and domain-leakage refusal, Definition 1145.5 and Definition 1145.4. They require the scope row before export and reject a beyond-scope claim that lacks the displayed refusal and extension surface.

Use source-refusal boundary next, Definition 1145.13. A beyond-scope read of the sibling source rows must be witnessed in D before the consumer route may accept it.

Finally apply upgrade-route determinacy and root export coverage, Definition 1145.9 and Definition 1145.34. They rule out deleting, replacing, or bypassing the carrier, leaving only the displayed refusal row D and upgrade row U as accepted beyond-scope routes. □

Theorem 1145.50 (Reality-constrained methodology-ledger authority nonescape)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, no unbounded explanation, final model truth, hidden observer privilege, closed empirical fit, endpoint/source identity, or off-ledger authority can be read from the packet. Any accepted methodology authority must factor through the displayed rows* $X, A, O, T, I, S, D, U, F, H, C, Q, N$*.*

Proof. Project the carrier and certificate, Definition 1145.1 and Definition 1145.2. The carrier has no coordinate for unbounded explanation, final model truth, hidden observer privilege, closed empirical fit, endpoint/source identity, or off-ledger methodology authority, and the certificate preserves only the displayed rows.

Apply consumer-route exactness and failure-route locality, Definition 1145.8 and Definition 1145.18. They keep consumer authority and defeating tests inside the visible carrier instead of turning either one into a hidden truth or observer-privilege route.

Finally use the export-accountability and upgrade-refusal-totality surfaces of this file. Every methodology export exposes the packet rows, and every beyond-scope export is covered by D or U, so there is no remaining route through which off-ledger authority can escape the finite NameCert packet. □

Theorem 1145.51 (Reality-constrained methodology-ledger root carrier obligation)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every root methodology-ledger read must expose the sibling failure, open-fit, and objectivity dependency rows through the same finite* BHist *carrier, the same componentwise* hsame *transport, the same displayed consumer route, the same package provenance row, and the same local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethod}}$ *row. The read cannot replace L by an off-ledger methodology carrier, hidden observer privilege, final model truth, or closed empirical oracle.*

Proof. Project the carrier of Definition 1145.1. It displays the explanation, audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and local naming rows in one packet.

Apply Definition 1145.47. The sibling failure, open-fit, and objectivity dependencies remain rows of the same finite methodology ledger rather than external carriers.

Finally use Definition 1145.19. Since every consumer route factors through L, there is no route for an off-ledger methodology carrier, hidden observer privilege, final model truth, or closed empirical oracle. □

Theorem 1145.52 (Reality-constrained methodology-ledger root classifier stability)**.** *For an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the root classifier is stable exactly along the displayed* hsame *transport and* Cont *route. Any classifier*

read comparing sibling failure, open-fit, objectivity, refusal, upgrade, or failure rows must compare the same local NameCert packet, and it cannot introduce a hidden fit selector, detached domain-truth predicate, or external adequacy theorem.

Proof. Start from Definition 1145.51. The root read has already exposed the displayed finite carrier and the local NameCert row.

Use Definition 1145.3. Its classifier and ledger clauses bind sibling accountability, scope exactness, domain-leakage refusal, failure exposure, and non-escape to the same packet.

Then apply Definition 1145.8. The classifier travels only along the displayed hsame and Cont rows, so hidden fit selectors, detached domain-truth predicates, and external adequacy theorems have no accepted comparison route. □

Theorem 1145.53 (Reality-constrained methodology-ledger root ledger exactness)**.** *Every root ledger export of an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet is exactly the finite packet* $X, A, O, T, I, S, D, U, F, H, C, Q, N$ *read through* BHist, hsame, Cont, *package provenance, and local NameCert data. The ledger records sibling failure, open-fit, objectivity, refusal, upgrade, and failure rows without exporting unbounded methodology authority, hidden empirical fit, endpoint identity, or off-ledger proof evidence.*

Proof. First retain the carrier and classifier by Definition 1145.51 and Definition 1145.52. The ledger export is already tied to the displayed finite packet and its componentwise transport route.

Next read the failure and refusal rows through Definition 1145.18 and Definition 1145.11. These theorems keep defeating tests, refusal records, and allowed extensions local to L.

Finally use Definition 1145.45. A public methodology readback may export only the local methodology-ledger interface, so unbounded methodology authority, hidden empirical fit, endpoint identity, and off-ledger proof evidence are excluded. □

Theorem 1145.54 (Reality-constrained methodology-ledger root authority carrier totality)**.** *Let* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ *be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every BEDC-kernel authority read of the root methodology ledger is total only when it reads the displayed carrier rows*

$$X, A, O, T, I, S, D, U, F, H, C, Q, N$$

as one finite BHist *packet, transports them through the same* hsame *row* H*, follows the same* Cont *consumer route* C*, preserves the same* Pkg *provenance row* Q*, and exposes the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodolo}}$ *row* N*. No authority read may replace this packet by host truth, hidden observer privilege, unbounded methodology scope, or an off-ledger proof surface.*

Proof. Project the carrier of Definition 1145.1. It displays the thirteen rows of L and gives no coordinate outside BHist, hsame, Cont, Pkg, and the local $\mathsf{NameCert}$ row from which root authority could be read.

Apply Definition 1145.3. Carrier admission, sibling accountability, scope exactness, refusal routing, failure exposure, and ledger non-escape are all obligations of the same packet.

Use Definition 1145.51 and Definition 1145.39. The first keeps the root read on the displayed rows, while the second identifies the TasteGate readback with the same finite carrier rather than a second methodology object. □

Theorem 1145.55 (Reality-constrained methodology-ledger scope-refusal upgrade-route exhaustion)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*,*

each root authority attempt beyond the displayed scope row S is exhausted by the refusal row D and upgrade-route row U before it reaches the consumer route C. The exhaustion is read inside the same finite BHist *carrier, along the same* hsame *transport,* Cont *route,* Pkg *provenance, and local* NameCert *surface; it cannot appeal to hidden domain truth, unrecorded methodology authority, or a closed empirical oracle.*

Proof. Begin with Definition 1145.54. Any root authority attempt has already been placed on the displayed packet $X, A, O, T, I, S, D, U, F, H, C, Q, N$.

Apply Definition 1145.5 and Definition 1145.4. They require S before methodology export and route every beyond-scope attempt through D or the declared extension path U.

Now use Definition 1145.11 and Definition 1145.9. Together they exhaust the accepted beyond-scope route by the displayed refusal and upgrade rows rather than by deleting, replacing, or bypassing the carrier.

Finally apply Definition 1145.8. Since C factors through the same carrier and no hidden domain-truth coordinate exists, the exhausted D, U route is the only root authority route available before consumption. □

Theorem 1145.56 (Reality-constrained methodology-ledger failure consumer ledger coverage)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet* $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$*, the root failure-consumer surface covers all finite defeats that a methodology consumer may read. The coverage keeps F and C inside the displayed* BHist *ledger, preserves comparison by* hsame*, routes consumption through* Cont*, records provenance by* Pkg*, and exposes the same local* NameCert *obligations; it never exports an untracked failure waiver, hidden fit selector, endpoint identity, or off-ledger methodology certificate.*

Proof. Start from Definition 1145.54. The failure row F and consumer route C are rows of the same packet, with no additional failure-waiver coordinate.

Apply Definition 1145.7 and Definition 1145.12. They require finite defeats from the sibling, scope, refusal, and upgrade rows to be visible in F before C is accepted.

Use Definition 1145.8 and Definition 1145.18. These results keep the consumer route and defeating tests local to the displayed carrier rather than to hidden fit, endpoint identity, or an off-ledger certificate.

Finally read the same surface through Definition 1145.39. The TasteGate carrier alignment confirms that the public readback decodes this finite ledger itself, so the failure-consumer coverage is not supplied by a second methodology object. □

Definition 1145.57 (Reality-constrained methodology-ledger root route packet)**.** *The root route packet for* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *is the finite* BHist *display*

$$R = (O, M, J, E, T, K, C, P, N)$$

whose rows record, in order, open-fit evidence from $\mathsf{OpenFitPacket}^{\uparrow}$*, model-audit evidence from* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$*, objectivity evidence from* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$*, explanation evidence from* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$*, tower-compression evidence from* RealityConstrainedTowerCompr*transport rows for the displayed methodology constraints, continuation rows for root consumers, provenance rows for the sibling certificates, and the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ *row. The five source rows are read through chapter 1137, chapter 1124, chapter 1133, chapter 1136, and chapter 1142.*

The packet has no coordinate for hidden model identity, unbounded truth, a host semantic oracle, unledgered compression, or private classifier choice.

Theorem 1145.58 (Reality-constrained methodology-ledger root route totality)**.** *Every root methodology-ledger consumer factors through the rows of Definition 1145.57. In particular, the consumer must read the open-fit, model-audit, objectivity, explanation, tower-compression, transport, continuation, provenance, and local* NameCert *rows before using the methodology ledger as a root dependency.*

Proof. Project the local carrier and certificate from Definition 1145.1 and Definition 1145.2. They already expose the sibling, transport, consumer, provenance, and local naming rows as a finite ledger.

Use the sibling chapters as the only admitted sources. The open-fit packet, model-audit packet, objectivity packet, explanation packet, and tower-compression packet supply the five root methodology constraints named in the packet definition.

Finally apply the local NameCert obligations, root carrier obligation, and consumer-route exactness. These results force the consumer route to stay on the displayed finite rows and prevent a hidden methodology source from replacing the local ledger. □

Theorem 1145.59 (Reality-constrained methodology-ledger root route nonescape)**.** *The root route packet of Definition 1145.57 exports no hidden model identity, unbounded truth coordinate, host semantic oracle, unledgered compression channel, or private classifier choice. Every accepted root read remains a finite BEDC route through* BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert *row.*

Proof. Start from root route totality. Any accepted root consumer has already been routed through $O, M, J, E, T, K, C, P, N$.

Apply the carrier non-escape clauses from the local NameCert obligations and the root authority carrier totality theorem. They leave no field in the methodology ledger for a model identity hidden from the packet, unbounded truth, or a host semantic oracle.

Use the tower-compression and consumer-route rows for the remaining exclusions. Compression is ledgered through the displayed tower row, while classifier choice and consumer replay remain public BHist, Cont, Pkg, and NameCert data rather than private host coordinates. □

Theorem 1145.60 (Reality-constrained methodology-ledger root source invariant triad)**.** *For the root route packet*

$$R = (O, M, J, E, T, K, C, P, N)$$

of Definition 1145.57, the source invariants read by downstream methodology-ledger consumers are exactly the sibling rows O, M, J, E, T together with the transport, continuation, provenance, and local naming rows K, C, P, N. The packet exposes those invariants only as finite BHist *data transported by* hsame*, replayed by* Cont*, packaged by* Pkg*, and bound by the local* NameCert *row.*

Proof. Begin with the packet definition. It names the five sibling source rows for open fit, model audit, objectivity, explanation, and tower compression, and it records the four support rows K, C, P, N in the same finite display.

Apply root route totality. Any root methodology-ledger consumer must first read the displayed rows $O, M, J, E, T, K, C, P, N$, so the sibling sources cannot be replaced by a hidden methodology source.

Finally apply root route nonescape. The only kernel support exported with those sibling rows is BHist, hsame, Cont, Pkg, and the local NameCert row, which makes the displayed triad of source, transport, and provenance the whole root surface. □

Theorem 1145.61 (Reality-constrained methodology-ledger root failure ledger surface)**.** *Every failure or refusal read available at the methodology-ledger root is ledgered through the displayed packet rows*

$$O, M, J, E, T, K, C, P, N.$$

The sibling source rows supply the visible open-fit, model-audit, objectivity, explanation, and tower-compression tests, while K, C, P, N keep the failure ledger inside BHist*,* hsame*,* Cont*,* Pkg*, and the local* NameCert *surface. No off-ledger failure waiver, hidden model identity, or unbounded truth coordinate is exported.*

Proof. Project the finite packet from Definition 1145.57. Its source rows are precisely the sibling rows O, M, J, E, T, and its support rows are K, C, P, N.

Use root route totality to force every accepted root consumer through those rows before the methodology ledger can be used as a dependency. Thus any failure or refusal test seen at the root is read from the same finite packet.

Root route nonescape supplies the exclusions. Since the packet exports only BHist, hsame, Cont, Pkg, and local NameCert support, an off-ledger waiver, hidden identity, or unbounded truth coordinate has no row from which it could be read. □

Theorem 1145.62 (Reality-constrained methodology-ledger root export nonescape)**.** *The public root export of* RealityConstrainedMethodologyLedger$^{\uparrow}$ *is exhausted by the packet rows $O, M, J, E, T, K, C, P, N$, with the sibling source rows carried through the displayed BEDC support rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. A downstream consumer may export the finite methodology-ledger route, but it may not export hidden model identity, unbounded truth, a host semantic oracle, unledgered compression, private classifier choice, or a source row outside the sibling packet.*

Proof. First read the packet definition, which fixes the export surface as $O, M, J, E, T, K, C, P, N$ and gives no coordinate for any additional source row.

Root route totality then identifies every accepted downstream read with that same packet. The source rows are the listed siblings, and the continuation and provenance rows carry them without introducing a second route.

Apply root route nonescape for the export boundary. It restricts the consumer-visible support to BHist, hsame, Cont, Pkg, and the local NameCert row, so none of the excluded host, truth, compression, classifier, or extra-source coordinates can leave the root packet. □

Theorem 1145.63 (Reality-constrained methodology-ledger open-fit model-audit obligation)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *carrier. Every downstream methodology-ledger consumer that reads the open-fit row I must also expose the model-audit row A, the scope row S, the refusal row D, the failure row F, the transport row H, the consumer route C, the provenance row Q, and the local* NameCert *row N. The read is therefore a finite* OpenFitPacket$^{\uparrow}$ *and* RealityConstrainedModelAudit$^{\uparrow}$ *obligation inside the same ledger, not a closed empirical fit or a host model-truth certificate.*

Proof. Project the accepted carrier from Definition 1145.1. The open-fit row I and model-audit row A are sibling rows of the same packet, while S, D, F, H, C, Q, N are the scope, refusal, failure, transport, route, provenance, and local naming rows that control public consumption.

Use the sibling-accountability and scope-exactness clauses of Definition 1145.3. They force the open-fit and model-audit reads to remain visible before any methodology claim is exported.

Now apply Definition 1145.58 and Definition 1145.59. The root route packet admits open-fit and model-audit evidence only through displayed BEDC rows and excludes hidden model identity, unbounded truth, host semantic oracles, unledgered compression, and private classifier choice.

Finally, Definition 1145.54 keeps the authority read on L itself. Thus the open-fit read cannot close empirical fit or certify model truth outside the local ledger. □

Theorem 1145.64 (Reality-constrained methodology-ledger objectivity tower-compression handoff)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *carrier. Every root consumer that uses the objectivity row O together with the tower-compression row T must hand both rows through the displayed transport, consumer, provenance, and local naming surface H, C, Q, N. The handoff preserves the* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *and* $\mathsf{RealityConstrainedTowerCompr}$ *obligations as finite sibling dependencies and does not export a hidden observer, endpoint/source identity, or unledgered compression channel.*

Proof. Start with the carrier inventory. The rows O and T are displayed beside the common support rows H, C, Q, N, so objectivity and tower-compression data are not separate host-side resources.

Apply Definition 1145.60. It identifies the sibling source rows of the root route packet with the transport, continuation, provenance, and local naming rows that carry them to downstream consumers.

Use Definition 1145.14. That theorem requires the sibling rows and BEDC kernel support rows to be exposed before the consumer route can treat the methodology ledger as authority.

The exclusions then follow from Definition 1145.62. Since the public root export is exhausted by the displayed packet rows, no hidden observer, endpoint identity, or unledgered compression channel can be exported with the handoff. □

Theorem 1145.65 (Reality-constrained methodology-ledger NameCert unblock surface)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *carrier. The combined root dependency surface for* $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$, $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$, *and the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodolog}}$ *row is exactly the finite packet rows $X, A, O, T, I, S, D, U, F, H, C, Q, N$. A downstream methodology-ledger consumer may use those rows as a root route, but it cannot bypass them by importing unbounded explanation, final model truth, closed empirical fit, hidden observer privilege, endpoint identity, or off-ledger methodology authority.*

Proof. First combine the carrier and local certificate. The carrier displays the rows $X, A, O, T, I, S, D, U, F, H, C, Q,$ and Definition 1145.3 makes sibling accountability, scope exactness, refusal routing, failure exposure, and ledger non-escape obligations of that same packet.

Next read the root route packet through Definition 1145.57. Its open-fit, model-audit, objectivity, explanation, and tower-compression rows are carried with transport, continuation, provenance, and local naming support, so the root route is a finite BEDC surface rather than an ambient methodology principle.

For downstream consumption, use Definition 1145.61 and Definition 1145.56. These results keep failure and refusal reads inside the displayed ledger before the consumer route is accepted.

Finally apply Definition 1145.54. It leaves no coordinate from which unbounded explanation, final truth, closed fit, hidden observer privilege, endpoint identity, or off-ledger authority could be read, so the listed rows form the unblock surface. □

Theorem 1145.66 (Reality-constrained methodology-ledger observation row)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet. Every observation-facing methodology read must expose the sibling rows for* $\mathsf{RealityConstrainedTruthCert}$ *and* $\mathsf{SignatureFit}$ *together with the displayed scope row S, refusal row D, failure row F, transport row H, consumer route*

C, provenance row Q, and local $\mathsf{NameCert}_{\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}}$ *row N. The observation row is therefore finite* BHist *ledger data, not final model truth, hidden observer privilege, or an off-ledger domain oracle.*

Proof. Start with the carrier and NameCert obligations of the chapter. They expose $X, A, O, T, I, S, D, U, F, H, C, Q$ as the finite packet and make sibling accountability, scope exactness, refusal routing, failure exposure, and ledger non-escape duties of that packet.

Next use the root methodology triad and root audit source lock. These rows keep observation, audit, objectivity, tower-compression, and open-fit evidence inside the same transport, consumer, provenance, and naming surface.

Finally apply the root open-fit model-audit obligation. It prevents an observation read from becoming a closed empirical fit or host model-truth certificate outside the displayed local ledger. □

Theorem 1145.67 (Reality-constrained methodology-ledger method-fit row)**.** *For every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$, every method-fit read through the* SignatureFit *surface is accepted only with the* RealityConstrainedTruthCert *observation rows, the scope/refusal/upgrade rows S, D, U, and the failure/consumer/provenance/name rows F, C, Q, N visible in the same packet. The read cannot be replaced by unbounded explanation, final truth, closed empirical fit, hidden observer privilege, endpoint identity, or off-ledger methodology authority.*

Proof. Begin with scope-export exactness and domain-leakage refusal. These results require S before methodology export and route beyond-scope attempts through the displayed refusal surface.

Now apply upgrade-route determinacy and failure-surface exhaustion. Any admitted movement beyond the current scope stays in U, and defeating tests from sibling surfaces must be collected in F before C is accepted.

Finally use consumer-route exactness and the root route packet. They identify method-fit consumption with a read of L itself, carried by H, C, Q, N, rather than with a second authority outside the ledger. □

Theorem 1145.68 (Reality-constrained methodology-ledger nonescape)**.** *Every root consumer of an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet is exhausted by the observation row, method-fit row, scope row, refusal row, upgrade row, failure row, transport row, consumer route, provenance row, and local* NameCert *row displayed in $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$. No hidden domain leakage, private classifier choice, final model-truth oracle, unledgered compression channel, endpoint identity, or off-ledger methodology authority is available to that consumer.*

Proof. First combine the observation row and method-fit row above. Together they place truth-certificate and signature-fit consumption inside the displayed methodology ledger with S, D, U, F, H, C, Q, N visible.

Next apply the chapter's root route totality and root route non-escape theorems. They admit consumer reads only through displayed BEDC rows and exclude hidden model identity, unbounded truth, host semantic oracles, unledgered compression, and private classifier choice.

Finally use the NameCert unblock surface and root authority carrier totality. Since the authority read stays on L itself and the local certificate names no additional coordinate, the root consumer cannot escape the finite methodology ledger. □

Theorem 1145.69 (Reality-constrained methodology-ledger TruthCert reverse dependency)**.** *Let $L = (X, A, O, T, I, S, D, U, F, H, C, Q, N)$ be an accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$

packet. If L is read as TruthCert methodology authority, then the read is consumed through chapter 1144 and preserves the displayed rows

$$X, A, O, T, I, S, D, U, F, H, C, Q, N.$$

It may not be read as freestanding methodology authority, final model truth, hidden observer privilege, closed empirical fit, or off-ledger proof.

Proof. Project the carrier of Definition 1145.1. The only available methodology-ledger coordinates are the displayed rows $X, A, O, T, I, S, D, U, F, H, C, Q, N$, and the carrier explicitly omits final model truth, hidden observer privilege, closed empirical fit, and off-ledger export.

Apply the TruthCert consumer handoff theorem, Definition 1145.46. A TruthCert-facing methodology read receives the same finite ledger after carrier admission, local NameCert obligations, failure-consumer coverage, formal-target route totality, and TasteGate readback have displayed it.

The consumer is therefore routed through the TruthCert carrier chapter, chapter 1144, rather than through a freestanding authority surface. Since that chapter consumes the methodology-ledger handoff as a TruthCert-facing export, any accepted read must preserve all rows of L and cannot replace them by an absent truth, observer, fit, or proof coordinate. □

Theorem 1145.70 (Reality-constrained methodology-ledger sibling lattice route)**.** *Every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *public route is not a kernel-only float. It is read through the existing explanation, model-audit, objectivity, tower-compression, and open-fit dependency family, exported to chapter 1144, and consumed by chapter 1185.*

Proof. Apply Definition 1145.47. It keeps the explanation, model-audit, objectivity, tower-compression, and open-fit dependencies displayed as rows of the same finite methodology ledger before the root consumer can read the packet.

Next apply Definition 1145.45. Public methodology authority is therefore exported through the displayed rows and their failure-consumer, formal-target, and TasteGate routes, rather than through a kernel-only object detached from the sibling dependency family.

Finally use Definition 1145.69. The accepted read is routed through chapter 1144; the scientific-methodology consumer of chapter 1185 then receives the methodology-ledger row as a downstream public dependency, so the route remains horizontal across the sibling lattice instead of floating as an isolated kernel packet. □

Theorem 1145.71 (Reality-constrained methodology audit-membrane dependency)**.** *Every accepted* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *packet that is exported as public methodology must consume the audit/export boundary of chapter 526 and the membrane-local routing discipline of chapter 533 before its local* NameCert *row is read. The exported route keeps methodology scope, audit evidence, membrane admission, failure exposure, transport, replay, provenance, and naming in one finite* BHist *packet, and it cannot be read as a kernel-only methodology island.*

Proof. Use the sibling lattice route. It already places public methodology export inside the displayed explanation, audit, objectivity, tower-compression, open-fit, truth-certificate, and scientific-methodology dependency family.

The audit-gate boundary supplies the finite pass, refusal, ancestry, and replay rows that make a public methodology claim inspectable. The operational membrane supplies the export-side membrane row that keeps those audit rows attached to a concrete consumer boundary rather than to an ambient host predicate.

Combining these rows with the methodology-ledger carrier forces every public export through the same BHist, hsame, Cont, Pkg, and NameCert surface. Thus the methodology ledger records a substantive horizontal dependency on the audit and membrane boundary chapters before its local name is consumed. □

The RealityConstrainedMethodologyLedger$^{\uparrow}$ carrier consumes the finite-cover methodology exemplified by chapter 388 as a sibling route before public methodology reads are exported. This dependency keeps the ledger's scope, audit, refusal, and failure rows tied to a concrete finite-cover method pattern rather than to a detached methodology slogan. The RealityConstrainedMethodologyLedger$^{\uparrow}$ carrier reads the core registry index of chapter 1251 before methodology authority, route unblock, and export-surface rows can be treated as ledger entries. This dependency keeps the methodology ledger inside the shared registry lattice: the authority row, root-route row, open-fit row, and public export row are not free-standing methodology claims, but indexed entries whose provenance is visible before any downstream consumer reads them.

Theorem 1145.72 (Reality-constrained methodology refusal-registry dependency)**.** *Every accepted* RealityConstrainedMethodologyLedger$^{\uparrow}$ *packet whose refusal, denial, or upgrade row is consumed as methodology output must first pass through the finite refusal packet of chapter 1248. A downstream reader may then consume the methodology row as a displayed refusal boundary, but it may not read denial as silence, package provenance as authority, or missing upgrade evidence as permission to export.*

Proof. Start with the methodology-ledger root and refusal rows. The root obligation surface, authority unblock rows, and route-unblock packet expose methodology scope, denial, upgrade, provenance, and local naming as finite ledger coordinates rather than as ambient policy claims.

Apply the export-control surface of RefusalRegistry$^{\uparrow}$. Its blocked-claim, refusal-entry, gate, cannot-claim, status, provenance, transport, package, and local NameCert rows require an explicit refusal packet before a consumer can treat a blocked route as a valid refusal boundary.

Compose the two packets through the existing ledger route. The methodology ledger supplies the local denial and upgrade coordinates, while RefusalRegistry$^{\uparrow}$ supplies the refusal export control. Therefore the consumer receives a named refusal boundary and not silence, hidden authority, or an unregistered upgrade route. □

Theorem 1145.73 (RealityConstrainedMethodologyLedger Helly sibling route)**.** *Every methodology-ledger read that consumes the* HellyTheorem$^{\uparrow}$ *finite convex-intersection surface must display the convex-set, polytope, affine, vector, and compact finite-read rows before the route is admissible. Concretely, the* RealityConstrainedMethodologyLedger$^{\uparrow}$ *consumer may read the Helly finite-subfamily ledger only after the* ConvexSet$^{\uparrow}$ *rows of chapter 208, the* Polytope$^{\uparrow}$ *witness rows of chapter 209, the* AffineSpace$^{\uparrow}$ *and* VecSpace$^{\uparrow}$ *ambient rows of chapter 206 and chapter 47, and the* CompactMetric$^{\uparrow}$ *finite-read boundary of chapter 129 have been displayed in the Helly packet of chapter 2309.*

Thus the methodology ledger admits Helly as a horizontal dependency route only through the displayed finite convex-intersection ledger. It receives no host convexity theorem, selected intersection point, quotient convex-set equality, unrestricted finite-cover principle, or ambient standard-model convexity coordinate.

Proof. Project the Helly carrier route from Definition 2309.1. Its public rows expose the affine ambient data, vector displacement data, convex-family rows, polytope witness ledger, compact finite-read boundary, finite subfamily ledger, transport, replay, provenance, and local naming.

Use the Helly sibling route of Definition 2309.10. It orders the convex-family membership rows, finite witness rows, affine and vector ambient rows, and compact finite-read boundary before the finite intersection ledger can be consumed.

The methodology ledger may therefore record the Helly route only as an admissible finite dependency after those rows are displayed. Since neither the Helly carrier nor the methodology ledger has a coordinate for a host convexity theorem, selected point, quotient convex-set equality, or unrestricted finite-cover oracle, none is exported by the dependency. □

Theorem 1145.74 (RealityConstrainedMethodologyLedger scoped closure record)**.** *The carrier of Definition 1145.1, the consumer-route exactness theorem Definition 1145.8, the domain-leakage refusal theorem Definition 1145.4, and the upgrade-route determinacy theorem Definition 1145.9 close the scoped* RealityConstrainedMethodologyLedger$^{\uparrow}$ *methodology-ledger surface. The scope is exactly the finite packet rows* $X, A, O, T, I, S, D, U, F, H, C, Q, N$ *over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, together with the sibling rows already displayed by the chapter. It exports no hidden methodology authority, final model truth, closed empirical fit, endpoint identity, or bridge row.*

Proof. First project the carrier. The rows $X, A, O, T, I, S, D, U, F, H, C, Q, N$ display the explanation, model-audit, objectivity, tower-compression, open-fit, scope, refusal, upgrade, failure, transport, consumer, provenance, and local naming surface.

Next apply consumer-route exactness. Every accepted consumer route factors through precisely those rows, so a downstream methodology read cannot bypass the displayed packet.

Then attach domain-leakage refusal and upgrade-route determinacy. The refusal row blocks hidden domain truth and unbounded scope, while the upgrade row determines accepted exports beyond the present scope without replacing the carrier.

Thus the chapter is scoped over the displayed methodology-ledger packet and its sibling obligations. The formal-status axis is witnessed by the existing checked TasteGate readback, while the theory-closure axis is supplied by the paper carrier and route arguments themselves. □

Closure status for NameCert$_{\text{RealityConstrainedMethodologyLedger}^{\uparrow}}$**.**

Constructive story: A public RealityConstrainedMethodologyLedger$^{\uparrow}$ packet is generated from methodology carrier rows, consumer-route exactness, domain-leakage refusal, upgrade-route determinacy, sibling scope rows, public export readiness, root methodology export, componentwise transport, consumer replay, package provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedMethodologyLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1145.45, Definition 1145.34, Definition 1145.70, Definition 1145.46, and Definition 1145.39 bind the mature methodology-ledger route to the displayed carrier, root obligation, public export, sibling lattice, and TasteGate readback rows.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedMethodologyLedger}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealityConstrainedMethodologyLedgerUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This mature record does not assert final model truth, unbounded explanation, closed empirical fit, hidden observer privilege, endpoint identity, bridge closure, or standard bridge checking.

Upgrade path: Bridge checking remains the separate standard-bridge obligation for the same public methodology-ledger rows across the acceptance-bridge interface.

Chapter 1146

A Concrete Naming Certificate for TheoryTransitionCertificate$^{\uparrow}$

The TheoryTransitionCertificate$^{\uparrow}$ packet realizes theory change as a finite certificate transition between two certified theory states. It sits beside RealityConstrainedModelAudit$^{\uparrow}$ of chapter 1124, PhysicalTruthCertificate$^{\uparrow}$ of chapter 1141, RealityConstrainedExplanation$^{\uparrow}$ of chapter 1136, RealityConstrainedObjectiv of chapter 1133, and OpenFitPacket$^{\uparrow}$ of chapter 1137. Those siblings certify audit, truth, explanation, objectivity, and open fit for presented claims; the present packet binds two theory states and the transition rows between them.

Definition 1146.1 (Theory transition certificate carrier)**.** *A* TheoryTransitionCertificate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T_1, T_2, S, C, D, L, F, H, R, P, N)$$

with the following displayed rows:

1. *T_1 is the old certified theory state, read through its displayed* NameCert *surface;*
2. *T_2 is the candidate successor certified theory state, read through its displayed* NameCert *surface;*
3. *S is the conservative signature preservation row, listing the old certified signatures whose current predictions must remain readable in T_2;*
4. *C is the classifier-refinement row, naming the classifier changes allowed by the transition;*
5. *D is the predictive-stability and descent row, recording which old predictions remain current and which tower-local descent rows are refined;*
6. *L is the ledger-refinement row, displaying the ledger entries added, split, or sharpened by the transition;*
7. *F is the failure-surface refinement row, displaying the failure boundaries that become more explicit under T_2;*
8. *H stores componentwise* hsame *transports for $T_1, T_2, S, C, D, L, F, P, N$;*
9. *R stores the displayed* Cont *routes from the old state through signature preservation, classifier refinement, stability, ledger refinement, and failure-surface refinement to the successor state;*
10. *P is the* Pkg *provenance over* RealityConstrainedModelAudit$^{\uparrow}$, PhysicalTruthCertificate$^{\uparrow}$, RealityConstrainedExplana RealityConstrainedObjectivity$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoryTransitionCertificate}^{\uparrow}}$ *row.*
The carrier has no coordinate for theory identity, no final-truth oracle, no unledgered prediction transfer, and no hidden failure-surface erasure.

Lean4 (definition) `BEDC.Derived.TheoryTransitionCertificateUp.TheoryTransitionCertificateUp`
Lean4 ✓ `BEDC.Derived.TheoryTransitionCertificateUp.TheoryTransitionCertificateUp single carrier alignment`

Theorem 1146.2 (TheoryTransitionCertificate NameCert obligations)**.** *For every accepted* TheoryTransitionCertific *carrier*

$$K = (T_1, T_2, S, C, D, L, F, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TheoryTransitionCertificate}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed packet, old and successor theory-state readability through T_1, T_2, conservative signature preservation through S, classifier refinement through C, prediction and descent stability through D, ledger refinement through L, failure-surface refinement through F, componentwise transport through H, route replay through R, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 1146.1. The definition displays exactly the rows $T_1, T_2, S, C, D, L, F, H, R, P, N$ and assigns one transition role to each row.

The first seven rows are the mathematical transition surface: old state, successor state, conserved signatures, classifier refinement, stability and descent, ledger refinement, and failure-surface refinement. The remaining rows are the BEDC certificate infrastructure for transport, route replay, provenance, and local naming.

The sibling packets provide the surrounding claim-level surfaces. $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ supplies audit rows, $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ supplies truth-certificate rows, $\mathsf{RealityConstrainedExplanation}^{\uparrow}$ supplies explanation rows, $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ supplies objectivity rows, and $\mathsf{OpenFitPacket}^{\uparrow}$ supplies open-fit rows. Since the carrier has no further coordinates, every local obligation is one of the displayed rows above. □

Lean4 ✓ `BEDC.Derived.TheoryTransitionCertificateUp.TheoryTransitionCertificateNameCert obligations`

Theorem 1146.3 (TheoryTransitionCertificate conservative preservation)**.** *Let $K = (T_1, T_2, S, C, D, L, F, H, R, P,$ be an accepted* $\mathsf{TheoryTransitionCertificate}^{\uparrow}$ *packet. Every consumer that reads an old current prediction through T_1 and then through the successor state T_2 must pass through the conservative signature preservation row S, the stability row D, the displayed transport row H, and the route row R. Agreement on C, L, F, P, N cannot replace S or D, cannot erase an old certified signature, and cannot turn the transition into an identity between theories.*

Proof. Read the old and successor states from Definition 1146.1. The carrier separates T_1 from T_2 and makes S the only row that preserves old certified signatures across the transition.

Next read the stability row. The row D records which old predictions remain current and which tower-local descent rows are refined, so a consumer of old current predictions must pass through D after S has displayed the preserved signature.

The route and transport rows preserve this finite passage. Componentwise transport through H and route replay through R carry the same T_1, S, D, T_2 surface; the classifier, ledger, failure, provenance, and naming rows cannot substitute for it. Thus conservative preservation is a displayed finite condition, not identity of theories and not an unledgered prediction transfer. □

Lean4 ✓ `BEDC.Derived.TheoryTransitionCertificateUp.TheoryTransitionCertificate conservative preservation`

Theorem 1146.4 (TheoryTransitionCertificate ledger refinement boundary)**.** *For every accepted* $\mathsf{TheoryTransitionCertificate}^{\uparrow}$ *packet* $K = (T_1, T_2, S, C, D, L, F, H, R, P, N)$*, ledger refinement is accepted only when the ledger row* L *and failure-surface row* F *remain displayed with the preserved-signature and stability rows* S, D*. A transition packet that preserves old current predictions but drops either* L *or* F *is not an accepted* $\mathsf{TheoryTransitionCertificate}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.TheoryTransitionCertificateUp.TheoryTransitionCertificateLedgerRefinementBoundary`

Proof. Project L and F from the carrier definition. The ledger row records the entries added, split, or sharpened by the successor theory, while the failure-surface row records the boundaries made more explicit by the transition.

Now combine these rows with the conservative preservation theorem. Definition 1146.3 forces old current predictions through S and D; the present carrier also requires L and F as separate coordinates of the same packet.

The NameCert obligation theorem exhausts the accepted surface. Since Definition 1146.2 lists L and F as mandatory local obligations, a route that preserves predictions while dropping either row is not a route through the displayed $\mathsf{TheoryTransitionCertificate}^{\uparrow}$ packet. □

The later Lean file `lean4/BEDC/Derived/TheoryTransitionCertificateUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ TheoryTransitionCertificate}^{\uparrow}$ and $\mathsf{FieldFaithful\ TheoryTransitionCertificate}^{\uparrow}$ instances over the displayed fields $T_1, T_2, S, C, D, L, F, H, R, P, N$.

Falsifiable prediction: A transition packet that preserves old current predictions but drops a displayed ledger or failure-surface row is not an accepted TheoryTransitionCertificateUp packet.

Independence witness: realityconstrainedmodelaudit, physicaltruthcertificate, realityconstrained-explanation, realityconstrainedobjectivity, openfitpacket: none is bijective because TheoryTransitionCertificateUp binds two certified theory states and the transition rows between them, while the siblings certify audit, truth, explanation, objectivity, or open fit for a single presented claim.

Closure status for $\mathsf{NameCert}_{\mathsf{TheoryTransitionCertificate}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{TheoryTransitionCertificate}^{\uparrow}$ packet is generated from old and successor certified theory states, preserved signatures, classifier refinement, prediction and descent stability, ledger refinement, failure-surface refinement, componentwise transport, route replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoryTransitionCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1146.1, Definition 1146.2, Definition 1146.3, and Definition 1146.4 bind theory change as a finite transition packet over displayed certificate rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoryTransitionCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove identity of theories, final physical truth, unledgered prediction transfer, host semantic equivalence, Lean verification, or bridge checking for $\mathsf{TheoryTransitionCertificate}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, conservative preservation, ledger and failure-surface boundary, route replay, provenance, and local naming over the displayed transition rows.

Chapter 1147

A Concrete Naming Certificate for $\mathsf{OnticAdequacyAudit}^{\uparrow}$

The $\mathsf{OnticAdequacyAudit}^{\uparrow}$ packet realizes the adequacy discipline of section 9414.15 as one finite NameCert boundary. It sits after the ontic-source certificate of chapter 1125, the model-audit packet of chapter 1129, the tower certificate of chapter 1127, the stream-window surface of chapter 126, the regular-rational readback of chapter 301, and the Real carrier of chapter 38. Its role is to bind source, audit, Real-facing readback, classifier, residue, transport, route, provenance, and local naming rows before signature exactness is read as ontic adequacy.

Definition 1147.1 (Ontic adequacy audit carrier)**.** *An* $\mathsf{OnticAdequacyAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, M, S, R, E, K, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OnticState}^{\uparrow}$ *source row naming the stable reality-facing source under audit.*
2. *M is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row recording the finite source traces, observed signatures, model signatures, tower data, failure surface, strength, and audit ledger already accepted for the candidate model.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite observation windows that may be used for adequacy readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *facing row reached only through the displayed stream and regular-rational readback rows.*
6. *K is the signature classifier row comparing the ontic source signatures with the audited model signatures at the declared adequacy level.*
7. *L is the approximation and residue ledger, recording selected tower limits, rejected shortcut limits, unrepresented source content, and the gap between signature exactness and total ontic exhaustion.*
8. *H records componentwise* hsame *transport for the source, audit, stream, regular-rational, Real-facing, classifier, ledger, route, provenance, and local naming rows.*
9. *C records the displayed* Cont *route through $O, M, S, R, E, K, L, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{OnticAdequacyAudit}^{\uparrow}$, $\mathsf{OnticState}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OnticAdequacyAudit}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host truth, unbounded observation, selected limits outside L, quotient-real equality, Real completeness as physical adequacy, or a detached model-syntax route. Signature exactness may be read as ontic adequacy only through the full displayed packet.

Lean4 ✓ `BEDC.Derived.OnticAdequacyAuditUp.OnticAdequacyAuditTasteGate single carrier alignment`

Theorem 1147.2 (Ontic adequacy audit NameCert obligations)**.** *For the carrier of Definition 1147.1, the local* $\mathsf{NameCert}_{\mathsf{OnticAdequacyAudit}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $A = (O, M, S, R, E, K, L, H, C, P, N)$;*
2. *signature-exactness admission through the joint* $\mathsf{OnticState}^{\uparrow}$ *source row O,* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *audit row M, and classifier row K;*
3. *Real-facing readback admission through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows S, R, E;*
4. *residue-ledger exactness through L, including the selected tower limits, rejected shortcuts, and unrepresented source content;*
5. *non-escape from host truth, selected limits, quotient reals, Real completeness, and model syntax through componentwise transport, displayed replay, provenance, and local naming rows H, C, P, N.*

Thus an accepted adequacy read can promote signature exactness to ontic adequacy only when source trace, model audit, Real-facing readback, classifier, residue ledger, transport, route, provenance, and local name are displayed together.

Lean4 ✓ `BEDC.Derived.OnticAdequacyAuditUp.OnticAdequacyAudit semantic name certificate`

Proof. Carrier admission is projection from Definition 1147.1. The accepted tuple displays the ontic source O, the physical model audit M, the stream schedule S, the regular-rational readback R, the Real-facing row E, the classifier K, the residue ledger L, and the structural rows H, C, P, N.

The signature-exactness row is not a model-only comparison. It reads the source signatures from O, the audited model signatures from M, and the comparison row K inside the same packet. The Real-facing readback is likewise constrained: E is available only after the scheduled stream row S and regular-rational row R have been displayed.

The residue ledger closes the remaining adequacy boundary. Selected tower limits, rejected shortcut limits, and unrepresented source content must be recorded in L, so Real completion or quotient-real equality cannot replace the ledger. The rows H, C, P, N only transport, replay, source, and name the displayed packet. They add no host truth oracle, no hidden physical-object coordinate, and no second route from model syntax to ontic adequacy.

Therefore the five listed rows exhaust the local NameCert surface, and every accepted adequacy read remains inside the finite packet A. □

The Lean TasteGate file `lean4/BEDC/Derived/OnticAdequacyAuditUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{OnticAdequacyAudit}^{\uparrow}$ and FieldFaithful $\mathsf{OnticAdequacyAudit}^{\uparrow}$ instances preserving $O, M, S, R, E, K, L, H, C, P, N$ as separate displayed fields.

Falsifiable prediction: If the source-trace row or residue ledger is absent, no accepted OnticAdequacyAuditUp route can promote signature exactness to ontic adequacy.

Independence witness: onticstate, physicalmodelaudit, realityconstrainedapproximationtower, observationreflectionpacket, real: none is bijective because the audit packet binds source, audit, Real-facing readback, classifier, residue, transport, route, provenance, and local naming rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{OnticAdequacyAudit}^{\uparrow}}$.

Constructive story: An $\mathsf{OnticAdequacyAudit}^{\uparrow}$ packet is generated from an ontic source row, physical model audit row, stream schedule, regular-rational readback, Real-facing row, signature classifier, approximation and residue ledger, componentwise hsame transport, displayed Cont route, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OnticAdequacyAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1147.1 and Definition 1147.2 bind the finite adequacy-audit carrier and its source, audit, readback, classifier, ledger, transport, route, provenance, and local naming obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OnticAdequacyAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total physical ontology, host truth, quotient-real adequacy, selected-limit sufficiency, Real completeness as physical adequacy, or model syntax as ontic adequacy.

Upgrade path: Obligation closure requires checked carrier admission, signature-exactness admission, Real-facing readback admission, residue-ledger exactness, non-escape, TasteGate nontriviality, and field faithfulness for $O, M, S, R, E, K, L, H, C, P, N$.

Chapter 1148

A Concrete Naming Certificate for $\mathsf{ApproximationTowerResidue}^{\uparrow}$

The $\mathsf{ApproximationTowerResidue}^{\uparrow}$ packet realizes the hidden-source and failure-surface side of the approximation-tower vision in chapter 9414. It is adjacent to the indexed tower carrier of chapter 1127, the finite tower packet of chapter 1128, the audit surface of chapter 1129, the certified truth surface of chapter 1130, and the explanatory surface of chapter 1136. Those siblings expose approximation, packet routing, physical audit, certified truth, and explanation rows; this carrier isolates the residue row that prevents a hidden source or failed compression from being read as physical adequacy.

Definition 1148.1 (ApproximationTowerResidue carrier)**.** *An* $\mathsf{ApproximationTowerResidue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, C, L, F, U, D, H, K, P, N)$$

with the following displayed rows:

1. *S is the hidden-source residue row left by the tower compression;*
2. *T is the approximation-tower packet row whose compression produced the residue;*
3. *C is the classifier row recording the physical-signature comparison that remains under inspection;*
4. *L is the ledger row naming the GAP-admissible compression and every omitted source coordinate;*
5. *F is the failure-surface row naming finite defeating observations, signature gaps, or classifier mismatches;*
6. *U is the recovery-route row, recording which later observation or audit route may reopen the residue;*
7. *D is the descent-survivor row, recording which tower operations remain valid after the residue is exposed;*
8. *H records componentwise* hsame *transport for source, tower, classifier, ledger, failure, recovery, and descent rows;*
9. *K records displayed* Cont *replay through the same rows;*

10. *P is the* Pkg *provenance over* $\mathsf{ApproximationTowerResidue}^{\uparrow}$*, the five cited sibling chapters,* BHist, hsame, Cont, Pkg*, and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApproximationTowerResidue}^{\uparrow}}$ *row.*

The carrier has no coordinate for promoting a hidden source, failed classifier, or unledgered residue into a certified physical adequacy claim.

Lean4 ✓ `BEDC.Derived.ApproximationTowerResidueUp.ApproximationTowerResidueTasteGate single carrier alignment`

Definition 1148.2 (ApproximationTowerResidue classifier)**.** *For accepted packets*

$$R = (S, T, C, L, F, U, D, H, K, P, N) \quad \text{and} \quad R' = (S', T', C', L', F', U', D', H', K', P', N'),$$

the predicate $\mathsf{ApproximationTowerResidueClassifier}(R, R')$ *holds when the hidden-source residue, tower packet, classifier, ledger, failure surface, recovery route, and descent-survivor rows are transported componentwise by H, H' and replayed by K, K' without changing which residue coordinates are still exposed.*

Classifier agreement therefore preserves the residue surface itself. It does not identify two tower endpoints, certify physical truth, or erase a failure row.

Lean4 (definition) `BEDC.Derived.ApproximationTowerResidueUp.ApproximationTowerResidueClassifier`

Theorem 1148.3 (ApproximationTowerResidue NameCert obligations)**.** *For the carrier of Definition 1148.1, the local* $\mathsf{NameCert}_{\mathsf{ApproximationTowerResidue}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier admission for $R = (S, T, C, L, F, U, D, H, K, P, N)$;*
2. *hidden-source residue exposure through S together with the tower packet row T;*
3. *classifier and GAP-ledger accountability through C, L;*
4. *failure-surface and recovery-route exposure through F, U;*
5. *descent-survivor, transport, replay, provenance, and local naming through D, H, K, P, N.*

Every consumer that reads a residue, failure surface, recovery route, or descent survivor from an approximation tower must display these rows before it can export its conclusion.

Proof. Project the carrier in Definition 1148.1. The source residue S, tower row T, classifier C, ledger L, failure surface F, recovery route U, and descent survivor D are the only positive content rows.

The structural rows do not add another source. Transport through H, replay through K, provenance through P, and local naming through N preserve the same displayed residue packet.

Thus the NameCert surface is exactly the five listed obligation rows. A consumer that asks for hidden source material, failure evidence, recovery, or descent outside those rows has left the accepted $\mathsf{ApproximationTowerResidue}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.ApproximationTowerResidueUp.TasteGate.ApproximationTowerResidueTasteGate single carrier alignment`

Theorem 1148.4 (ApproximationTowerResidue non-escape)**.** *An accepted* ApproximationTowerResidue$^{\uparrow}$ *packet cannot promote a hidden-source residue, omitted physical signature, failed classifier row, or unrecovered descent gap into physical adequacy. Any such promotion must first pass through*

$$S, T, C, L, F, U, D, H, K, P, N.$$

In particular, the packet exports no tower endpoint equality, certified physical truth, host-level model adequacy, unbounded validation route, or ambient completeness row.

Proof. Begin with the NameCert obligations of Definition 1148.3. They force every residue-facing read to expose the hidden-source row, tower packet, classifier, ledger, failure surface, recovery route, and descent survivor before any conclusion is exported.

The five sibling chapters supply larger consumer surfaces, but none supplies permission to erase the residue row. The tower and packet chapters expose approximation routes, the audit chapter reads physical fit, the certified-truth chapter reads a finite truth certificate, and the explanation chapter reads explanatory descent; each route still requires the local residue packet when the hidden-source surface is under inspection.

Therefore a hidden source, failure row, or recovery gap cannot be reinterpreted as physical adequacy by structural transport. The rows H, K, P, N only preserve and name the displayed packet, so all stronger claims are outside the carrier. □

Lean4 ✓ `BEDC.Derived.ApproximationTowerResidueUp.ApproximationTowerResidueCarrier hidden source nonescape`

Theorem 1148.5 (ApproximationTowerResidue recovery-route determinacy)**.** *For an accepted* ApproximationTowerResidue$^{\uparrow}$ *carrier* $R = (S, T, C, L, F, U, D, H, K, P, N)$*, every recovery-route consumer factors through the same residue ledger rows* S, T, C, L, F, U, D *and the same structural rows* H, K, P, N*. A recovery read may reopen the finite source or failure surface named by* U*, but it cannot add a private source row, second classifier, unledgered descent operation, certified-truth row, or physical-audit acceptance row.*

Lean4 ✓ `BEDC.Derived.ApproximationTowerResidueUp.ApproximationTowerResidue recovery route determinacy`

Proof. Use the classifier of Definition 1148.2 to keep the source residue, tower packet, classifier, ledger, failure surface, recovery route, and descent survivor componentwise aligned.

Apply Definition 1148.3. The recovery route is one of the local obligation rows, so a consumer sees U only together with $S, T, C, L, F, D, H, K, P, N$.

Finally use the non-escape theorem. Since certified truth and physical-audit acceptance require their own displayed carriers, the residue recovery route cannot supply those rows by itself. It only names the finite route along which the residue can be inspected again. □

The Lean follow-up path `lean4/BEDC/Derived/ApproximationTowerResidueUp/TasteGate.lean` must provide explicit Nontrivial ApproximationTowerResidue$^{\uparrow}$ and FieldFaithful ApproximationTowerResidue$^{\uparrow}$ instances. These instances must distinguish the hidden-source residue, tower packet, classifier, ledger, failure surface, recovery route, descent survivor, transport, replay, provenance, and local NameCert rows.

Falsifiable prediction: If the residue row is absent, an approximation-tower packet cannot promote a hidden-source failure into a certified physical adequacy claim.

Independence witness: reality-constrained-approximation-tower, reality-constrained-tower-packet, physical-model-audit, certified-physical-truth, reality-constrained-explanation: none is bijective because $\mathsf{ApproximationTowerResidue}^{\uparrow}$ carries the hidden-source residue, failure surface, recovery route, descent survivor, and local NameCert rows as one finite packet, while the siblings expose tower structure, packet routing, audit, truth, or explanation without isolating this residue surface.

Theorem 1148.6 (ApproximationTowerResidue scope lock)**.** *Every residue-facing consumer of* $\mathsf{ApproximationTowerResidue}^{\uparrow}$ *is scoped exactly by the carrier of Definition 1148.1, the classifier of Definition 1148.2, the NameCert obligations of Definition 1148.3, the non-escape boundary of Definition 1148.4, the recovery-route determinacy of Definition 1148.5, the falsifiable prediction, the independence witness, and the displayed TasteGate marker. No consumer may add a private hidden-source coordinate, second classifier, unledgered recovery route, or physical-adequacy export outside that finite packet.*

Lean4 ✓ `BEDC.Derived.ApproximationTowerResidueUp.ApproximationTowerResidue scope lock`

Proof. The carrier lists the only rows a residue-facing consumer can project: hidden-source residue, tower packet, classifier, GAP ledger, failure surface, recovery route, descent survivor, transport, replay, provenance, and local naming.

The classifier and NameCert obligations then force every accepted read to keep those rows componentwise visible. The non-escape theorem blocks adequacy export, while recovery-route determinacy keeps every reopened route inside the same ledger.

The falsifiable prediction and independence witness record the observable boundary against sibling chapters, and the TasteGate marker identifies the same finite carrier as the chapter's accepted packet. Hence the scoped surface is precisely the listed carrier, classifier, obligations, boundary theorems, witness rows, and marker. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApproximationTowerResidue}^{\uparrow}}$.

Constructive story: An $\mathsf{ApproximationTowerResidue}^{\uparrow}$ packet is generated from the hidden-source residue row, approximation-tower packet row, classifier row, GAP-admissible ledger, failure surface, recovery route, descent survivor, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApproximationTowerResidue}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The public route is Definition 1148.1, Definition 1148.2, Definition 1148.3, Definition 1148.4, Definition 1148.5, Definition 1148.6, the falsifiable prediction, the independence witness, and the TasteGate marker for the same finite residue packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApproximationTowerResidue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide Lean verification, bridge checking, physical adequacy, hidden-source promotion, certified physical truth, or a host-level model-adequacy row for $\mathsf{ApproximationTowerResidue}^{\uparrow}$.

Upgrade path: Public closure requires checked carrier admission, classifier preservation, residue non-escape, recovery-route determinacy, transport closure, and TasteGate nontriviality with field faithfulness.

Chapter 1149

A Concrete Naming Certificate for $\mathsf{ReflectiveStability}^{\uparrow}$

The $\mathsf{ReflectiveStability}^{\uparrow}$ packet realizes the reflective stability certificate of chapter 9414 as a finite NameCert surface. It consumes the source and classifier rows of chapter 1126, the checkpoint boundary of chapter 997, the model-audit row of chapter 1124, the closed observation carrier of chapter 616, and the finite observation interface of chapter 1007. Its role is to certify reuse under displayed continuations without exporting final truth or source-domain identity.

Definition 1149.1 (ReflectiveStability carrier)**.** *A* $\mathsf{ReflectiveStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, K, A, Q, O, F, T, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row inherited from an* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *inquiry;*
2. *K is the checkpoint row naming the visible* $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ *reuse boundary;*
3. *A is the classifier and audit row inherited from* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$*;*
4. *Q is the open-continuation row under which the self-description may be reused;*
5. *O is the closed observation row inherited from* $\mathsf{ClosedObservationSystem}^{\uparrow}$*;*
6. *F is the finite interface row inherited from* $\mathsf{FiniteObservationInterface}^{\uparrow}$*;*
7. *T is the transport row carrying componentwise* hsame *comparisons among the displayed source, checkpoint, classifier, observation, and interface rows;*
8. *L is the ledger row recording hidden source material, rejected finality claims, and source-domain separation;*
9. *H is the finite* BHist *carrier row for the reuse trace;*
10. *C is the displayed* Cont *route through the accepted reuse continuation;*
11. *P is the* Pkg *provenance over the same source, checkpoint, classifier, continuation, observation, interface, transport, and ledger rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectiveStability}^{\uparrow}}$ *row naming exactly this packet.*

The carrier has no coordinate for a final truth predicate, an identity between physical source and formal source, a host validation oracle, or an undisplayed checkpoint.

Lean4 ✓ `BEDC.Derived.ReflectiveStabilityUp.ReflectiveStabilityTasteGate single carrier alignment`

Theorem 1149.2 (ReflectiveStability NameCert obligations)**.** *For the carrier of Definition 1149.1, the local* $\mathsf{NameCert}_{\mathsf{ReflectiveStability}^{\uparrow}}$ *surface consists exactly of seven finite obligations:*

1. *carrier admission for the displayed packet* $R = (S, K, A, Q, O, F, T, L, H, C, P, N)$*;*
2. *source exposure, requiring every reuse read to begin from the displayed* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *source row* S*;*
3. *checkpoint exposure, requiring every reflected self-description to pass through the visible checkpoint row* K*;*
4. *classifier and audit exposure, requiring reuse to preserve the displayed audit row* A *before any continuation read;*
5. *open-continuation stability, requiring* Q *and* C *to expose the continuation under which reuse is licensed;*
6. *observation and interface stability, requiring* O *and* F *to keep the closed observation and finite interface rows visible;*
7. *transport, ledger, provenance, history, and local naming closure through* T, L, H, P, N*.*

Thus reflective stability is certified as finite reuse under displayed rows, not as a total self-truth principle.

Proof. Carrier admission is projection from Definition 1149.1. The twelve rows $S, K, A, Q, O, F, T, L, H, C, P, N$ are the complete finite packet visible to a consumer.

Source, checkpoint, and classifier exposure come from the inbound packets. The source row is read through chapter 1126; the checkpoint row is read through chapter 997; the classifier and audit row is read through chapter 1124. These rows describe reuse of a self-description, not identity with the source domain.

The continuation and observation rows then delimit the reuse claim. The open-continuation row Q and displayed route C state where the certificate may be replayed, while O and F keep the closed observation and finite interface surfaces visible.

Finally the transport, ledger, history, provenance, and local naming rows close the certificate. Since T, L, H, P, N add no final truth coordinate and no hidden checkpoint coordinate, the seven obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ReflectiveStabilityUp.ReflectiveStabilityCarrier namecert obligations`

Theorem 1149.3 (ReflectiveStability anchor stability)**.** *Every accepted* $\mathsf{ReflectiveStability}^{\uparrow}$ *packet preserves the same displayed anchor rows* S, K, A, O, F *whenever its self-description is reused under* Q *through the displayed* Cont *route* C*. A consumer may transport these anchors through* T *and record the reuse in* L, H, P, N*, but it cannot replace them by a detached source, an undisplayed checkpoint, a hidden classifier, an unbounded observation stream, or a host validation route.*

Proof. Read the accepted carrier of Definition 1149.1. The only anchor rows available for reuse are S, K, A, O, F, and the only continuation route available for reuse is Q through C.

Apply the obligation surface of Definition 1149.2. It requires source exposure, checkpoint exposure, classifier and audit exposure, open-continuation stability, and observation-interface stability before any consumer may read the local NameCert row.

The remaining structural rows only preserve the displayed anchors. Componentwise hsame transport at T, ledgering at L, history at H, package provenance at P, and local naming at N record the same finite route. None of these rows contains a detached source, hidden classifier, unbounded observation stream, or host validation route. □

Lean4 ✓ `BEDC.Derived.ReflectiveStabilityUp.ReflectiveStabilityCarrier anchor stability`

Theorem 1149.4 (ReflectiveStability nonescape)**.** *No accepted* $\mathsf{ReflectiveStability}^{\uparrow}$ *packet exports final truth, source-domain identity, host validation, or an undisplayed reflection checkpoint beyond the rows*

$$S, K, A, Q, O, F, T, L, H, C, P, N.$$

Every reflective reuse read must remain inside the displayed finite source, checkpoint, classifier, open-continuation, transport, ledger, provenance, and local NameCert packet.

Lean4 ✓ `BEDC.Derived.ReflectiveStabilityUp.ReflectiveStabilityCarrier nonescape`

Proof. Inspect the carrier. Its only truth-facing row is the finite classifier and audit row A, its only reflection-facing row is the checkpoint row K, and its only future-facing rows are the open continuation Q and displayed route C.

Use anchor stability from Definition 1149.3. Any consumer that reuses the self-description must preserve S, K, A, O, F and may only transport them through T while recording the route in L, H, P, N.

The ledger row records the excluded demands as non-exported material. Final truth, source-domain identity, host validation, and undisplayed checkpoints therefore have no carrier coordinate through which they can be read. □

Falsifiable prediction: No accepted $\mathsf{ReflectiveStability}^{\uparrow}$ packet exports final truth or source-domain identity beyond its displayed finite source, checkpoint, classifier, open-continuation, transport, ledger, provenance, and local NameCert rows.

Independence witness: observationreflectionpacket, reflectionruntimecheckpoint, realityconstrained-modelaudit, closedobservationsystem, finiteobservationinterface: $\mathsf{ReflectiveStability}^{\uparrow}$ is not bijective to any listed sibling because it simultaneously carries finite source access, visible checkpoint reuse, classifier audit, open continuation, closed observation, finite interface, componentwise transport, hidden-material ledger, replay history, displayed route, package provenance, and local naming rows. The observation packet lacks checkpoint-stability reuse; the runtime checkpoint lacks the model-audit and finite observation interface rows; the model audit lacks the open continuation and checkpoint reuse rows; the closed observation system lacks the reflection classifier; the finite observation interface lacks the ledgered stability certificate.

The Lean follow-up in `lean4/BEDC/Derived/ReflectiveStabilityUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{ReflectiveStability}^{\uparrow}$ and FieldFaithful $\mathsf{ReflectiveStability}^{\uparrow}$ instances preserving $S, K, A, Q, O, F, T, L, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{ReflectiveStability}^{\uparrow}}$.

Constructive story: A $\mathsf{ReflectiveStability}^{\uparrow}$ packet is generated from finite source access, a visible reflection checkpoint, classifier audit data, an open continuation, closed observation and finite interface rows, componentwise transport, a hidden-material ledger, replay history, displayed route, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReflectiveStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1149.1, Definition 1149.2, Definition 1149.3, and Definition 1149.4 bind the finite reflective stability packet and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReflectiveStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final truth, source-domain identity, host validation, undisplayed checkpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the packet carrier, NameCert obligations, anchor stability, non-escape boundary, TasteGate nontriviality, and field faithfulness for $S, K, A, Q, O, F, T, L, H, C, P, N$.

Chapter 1150

A Concrete Naming Certificate for $\mathsf{AnchorStabilityCertificate}^{\uparrow}$

The $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ packet realizes the anchor-stability coordinate of section 9414.30 as a finite NameCert carrier. It sits behind the anchor-stability row consumed by chapter 1138: $\mathsf{RelationalObjectivity}^{\uparrow}$ reads a relational claim only after this certificate has bound the invariant row, the anchor-change route, the classifier, the ledger, and the local naming row. Its sibling context is supplied by $\mathsf{ObserverHistIdentity}^{\uparrow}$ in chapter 479, $\mathrm{MultiHistConfig}^{\uparrow}$ in chapter 474, $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ in chapter 643, and $\mathsf{LocalityCell}^{\uparrow}$ in chapter 509.

Definition 1150.1 (Anchor-stability certificate carrier)**.** *An* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, I, R, K, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the observer-family row read through* $\mathrm{MultiHistConfig}^{\uparrow}$ *and the relevant* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *surfaces;*
2. *I is the invariant row whose claim is being transported across observer anchors;*
3. *R is the anchor-change route naming the source anchor, target anchor, and locality-cell boundary through which the comparison is allowed to move;*
4. *K is the classifier row that compares the transported invariant at the two anchors;*
5. *L is the ledger row recording the observations, transformations, and cross-perspective alignment facts used in the comparison;*
6. *H records componentwise* hsame *transports for the observer-family, invariant, route, classifier, ledger, package, and local naming rows;*
7. *C records displayed* Cont *routes from the observer-family surface to the anchor-change route, from the route to classifier comparison, and from the accepted certificate to downstream relational-objectivity consumers;*
8. *P is the* Pkg *provenance over* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AnchorStabilityCertificate}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares exactly $F, I, R, K, L, H, C, P, N$. It has no coordinate for a privileged observer, an unledgered anchor jump, a hidden invariant source, a global synchronization history, or host equality of observer descriptions.

Lean4 (definition) `BEDC.Derived.AnchorStabilityCertificateUp.AnchorStabilityCertificateUp`

Theorem 1150.2 (Anchor-stability certificate NameCert obligations)**.** *For every accepted* AnchorStabilityCertificate *carrier $A = (F, I, R, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AnchorStabilityCertificate}^{\uparrow}}$ *surface consists exactly of observer-family exposure through F, invariant-row exposure through I, anchor-change routing through R, classifier comparison through K, observation-and-transformation ledger exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.AnchorStabilityCertificateUp.AnchorStabilityCertificateTasteGate single carrier alignment`

Proof. Project the accepted packet. The rows F, I, R, K, L display the family, invariant, route, classifier, and ledger coordinates that make anchor stability inspectable.

Read the structural rows next. The transport row H compares the same displayed coordinates componentwise, the continuation row C replays only the displayed route, and the package and local name rows P, N locate the certificate inside the surrounding NameCert surface.

The sibling chapters supply the required inputs but not the whole certificate. $\mathsf{ObserverHistIdentity}^{\uparrow}$ and $\mathrm{MultiHistConfig}^{\uparrow}$ supply observer-family structure, $\mathsf{CrossPerspectiveAlignment}^{\uparrow}$ supplies alignment evidence, and $\mathsf{LocalityCell}^{\uparrow}$ supplies the local anchor boundary; the present packet is where those rows are stored together with the invariant row and classifier. □

Theorem 1150.3 (Anchor-stability certificate nonescape)**.** *Every accepted* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ *consumer reads anchor stability through the displayed packet*

$$F, I, R, K, L, H, C, P, N.$$

The consumer has no route to a privileged observer, unledgered anchor transport, hidden synchronization, detached invariant evidence, host-level equality of observers, or a relational-objectivity verdict outside that packet.

Lean4 ✓ `BEDC.Derived.AnchorStabilityCertificateUp.AnchorStabilityCertificateCarrier nonescape`

Proof. Use the carrier definition. Any accepted read must first expose the observer-family row F, invariant row I, anchor-change route R, classifier row K, and ledger row L.

The transport and continuation rows preserve that exposed packet. They may move the displayed invariant along the recorded route and replay the classifier comparison, but they do not add a second route, hidden observer anchor, or host equality coordinate.

Finally inspect the downstream use in $\mathsf{RelationalObjectivity}^{\uparrow}$. A relational-objectivity read can consume the certificate only as the anchor-stability row already displayed here. If the certificate is absent, detached from its ledger, or transported across anchors without R, K, L, H, C, P, N, the consumer leaves the accepted $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ surface. □

Theorem 1150.4 (Anchor-stability route exhaustion)**.** *Every anchor-stability read that is accepted by* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ *exposes the full packet*

$$F, I, R, K, L, H, C, P, N$$

and the sibling rows ObserverHistIdentity$^{\uparrow}$, MultiHistConfig$^{\uparrow}$, CrossPerspectiveAlignment$^{\uparrow}$, *and* LocalityCell$^{\uparrow}$ *before any* RelationalObjectivity$^{\uparrow}$ *consumer receives the invariant. The certificate has no route that sends an invariant to relational objectivity while skipping observer identity, multi-history configuration, cross-perspective alignment, locality-cell locality, or one of the displayed packet rows.*

Lean4 ✓ `BEDC.Derived.AnchorStabilityCertificateUp.AnchorStabilityCertificateCarrier route exhaustion`

Proof. Project the carrier of Definition 1150.1. The rows F, I, R, K, L expose the observer family, invariant, route, classifier, and ledger, while H, C, P, N expose transport, continuation, provenance, and local naming.

Read the sibling inputs through the same displayed rows. ObserverHistIdentity$^{\uparrow}$ and MultiHistConfig$^{\uparrow}$ supply the observer-family surface, CrossPerspectiveAlignment$^{\uparrow}$ supplies the alignment row, and LocalityCell$^{\uparrow}$ supplies the admissible locality boundary.

Now pass to the relational-objectivity consumer. By Definition 1150.3, the consumer can receive anchor stability only through the displayed certificate packet, so no unlisted sibling row or hidden anchor route can carry the invariant onward. □

Falsifiable prediction: A relational objectivity read fails whenever its invariant row is transported across observer anchors without the displayed anchor-stability certificate.

Independence witness: relationalobjectivity, observerhistidentity, multihistconfig, crossperspectivealignment, localitycell: none is bijective to AnchorStabilityCertificateUp because none stores the invariant row, anchor-change route, classifier, ledger, and local NameCert as one certificate.

The Lean file 'TasteGate.lean' for BEDC.Derived.AnchorStabilityCertificateUp must provide Nontrivial AnchorStab and FieldFaithful AnchorStabilityCertificate$^{\uparrow}$ instances for the displayed carrier $F, I, R, K, L, H, C, P, N$.

Closure status for NameCert$_{\text{AnchorStabilityCertificate}^{\uparrow}}$.

Constructive story: An AnchorStabilityCertificate$^{\uparrow}$ packet is generated from an observer-family row, invariant row, anchor-change route, classifier row, ledger row, componentwise transport, displayed continuation routes, package provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AnchorStabilityCertificate}^{\uparrow}}$, scopedClosed)

Scope closed: The scoped surface records the carrier, NameCert obligations, nonescape theorem, route-exhaustion theorem, TasteGate marker, falsifiable prediction, and independence witness for reads through $F, I, R, K, L, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\text{AnchorStabilityCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean TasteGate, Nontrivial, FieldFaithful, or checked NameCert obligations for anchor-stability certificates.

Upgrade path: Promotion beyond seed requires Lean TasteGate, Nontrivial, FieldFaithful, and NameCert checks for the displayed carrier and nonescape surface.

Chapter 1151

A Concrete Naming Certificate for $\mathsf{ReflectiveInquiry}^{\uparrow}$

The $\mathsf{ReflectiveInquiry}^{\uparrow}$ packet realizes the reflective inquiry system of Definition 9414.409 as a finite NameCert carrier. It places physical inquiry and MetaCIC self-study in one certificate surface without identifying their sources: physical observation contributes source-access and model-audit rows, while formal reflection contributes source-access and self-study rows. The bridge is a role correspondence, not a truth oracle.

Its physical audit rows are read through chapter 1124, and its observation-reflection packet rows are read through chapter 1126.

Definition 1151.1 (Reflective inquiry carrier)**.** *A* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, F, S, K, A, R, L, H, C, Q, N)$$

with the following displayed rows:

1. *P is the physical inquiry source row, read through observational traces and the physical model-audit surface;*
2. *F is the formal inquiry source row, read through MetaCIC source access and reflected self-study;*
3. *S is the signature row comparing physical observation signatures with formal source signatures;*
4. *K is the classifier row that records which role correspondences are accepted;*
5. *A is the package row storing the accepted physical and formal inquiry roles as one certificate packet;*
6. *R records componentwise* hsame *transports for the physical source, formal source, signature, classifier, package, ledger, continuation, and local naming rows;*
7. *L is the ledger row preserving failed fits, hidden-source refusals, rejected source-identifications, and non-finality demands;*
8. *H is the finite history row through which accepted inquiry reads are replayed;*
9. *C is the displayed* Cont *route for extending either the physical inquiry or the formal self-study without closing it;*
10. *Q is the open-continuation row recording that the certificate remains usable only under further inquiry;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectiveInquiry}^{\uparrow}}$ *row.*
The carrier has no coordinate for a final truth oracle, a physical-formal source identity, a host-level validation route, or an unledgered self-study claim.

Lean4 (definition) `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryCarrier`

Definition 1151.2 (Reflective inquiry classifier)**.** *The* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *classifier compares accepted carriers by the displayed rows $P, F, S, K, A, R, L, H, C, Q, N$. Two carriers classify together only when their physical source rows, formal source rows, signature rows, accepted role correspondences, package rows, transports, ledgers, replay histories, continuation routes, open-continuation rows, and local names agree componentwise through the certificate surface.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryClassifierAuditCertificate`

Theorem 1151.3 (Reflective inquiry NameCert obligation surface)**.** *For the carrier of Definition 1151.1, the local* $\mathsf{NameCert}_{\mathsf{ReflectiveInquiry}^{\uparrow}}$ *surface consists exactly of physical-source exposure through P, formal-source exposure through F, signature comparison through S, classifier acceptance through K, package admission through A, componentwise transport through R, ledger preservation through L, replay history through H, displayed continuation through C, open-continuation non-finality through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryNameCertObligationSurface`

Proof. Carrier admission projects the displayed packet $I = (P, F, S, K, A, R, L, H, C, Q, N)$. The two source rows P and F are separate coordinates, so the certificate can read both physical inquiry and formal self-study without identifying them.

The signature, classifier, and package rows bind the bridge. The row S records the visible signatures, K records the accepted role correspondence, and A stores the resulting packet as a NameCert object rather than as an external validation oracle.

The structural rows close the local obligation surface. Transport R, ledger L, replay history H, continuation C, open-continuation row Q, and local name N preserve the accepted route while recording failed fits and non-finality demands. Since no further coordinate exists, the listed obligations exhaust the surface. □

Theorem 1151.4 (Reflective inquiry non-escape)**.** *An accepted* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *packet cannot export a final truth oracle, identify physical reality with MetaCIC, replace either source row by the other, or hide a self-study claim outside*

$$P, F, S, K, A, R, L, H, C, Q, N.$$

Every consumer of the observation–reflection bridge must pass through the displayed physical source, formal source, signature, classifier, package, ledger, replay, continuation, open-continuation, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryUp nonescape`

Proof. Start from the carrier of Definition 1151.1. The physical source row P and formal source row F are both displayed and distinct, while the classifier row K records only accepted role correspondences between them.

Apply the classifier of Definition 1151.2. A comparison can use $P, F, S, K, A, R, L, H, C, Q, N$ only through componentwise certificate agreement, so it has no hidden coordinate for a final truth oracle or a source-domain identity.

Use the obligation surface of Definition 1151.3. The ledger row L records failed fits and rejected source-identifications, and the open-continuation row Q keeps both inquiry directions non-final. Hence every bridge consumer remains inside the displayed finite packet. □

Theorem 1151.5 (Reflective inquiry source separation)**.** *For every accepted* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *carrier*

$$I = (P, F, S, K, A, R, L, H, C, Q, N),$$

the physical source row P and the formal source row F remain distinct carrier coordinates under classifier agreement. Agreement through $S, K, A, R, L, H, C, Q, N$ records a role correspondence between the sources, not an identity of the physical inquiry source with the formal self-study source.

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryUp source separation transport`

Proof. Project the carrier of Definition 1151.1. The rows P and F are listed as separate displayed coordinates, one for physical inquiry and one for formal inquiry.

Apply the classifier of Definition 1151.2. It compares accepted carriers componentwise through $P, F, S, K, A, R, L, H, C, Q, N$, so agreement preserves both source rows rather than replacing either one.

Use the obligation surface and non-escape theorem. The local NameCert exposes both source rows, while Definition 1151.4 blocks physical-formal source identity and source replacement outside the displayed packet. □

Theorem 1151.6 (Reflective inquiry open-continuation preservation)**.** *In every accepted* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *packet, the open-continuation row Q prevents either the physical route through P or the formal route through F from becoming a final truth oracle. The certificate may replay and extend inquiry through H and C, but it keeps both source routes open to further inquiry.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiry open continuation row separation`

Proof. Start with the carrier. The row Q is displayed as the open-continuation row, while H and C record replay history and displayed continuations for accepted inquiry reads.

Use the NameCert obligation surface. It requires open-continuation non-finality through Q as part of the local certificate, not as an optional annotation outside the packet.

Apply non-escape. Since no final truth oracle is a carrier coordinate and every consumer remains inside $P, F, S, K, A, R, L, H, C, Q, N$, neither source route can be closed by replay or continuation alone. □

Theorem 1151.7 (Reflective inquiry ledgered role correspondence)**.** *The physical/formal role correspondence in an accepted* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *packet is read through the rows*

$$S, K, A, R, L, H, C, Q, N,$$

together with the displayed source rows P, F. It is not read through source identity, host-level validation, or an unledgered self-study claim.

Proof. Read the bridge rows from the carrier and classifier. The signature row S, classifier row K, package row A, transport row R, ledger row L, replay row H, continuation row C, open-continuation row Q, and local naming row N are the displayed route by which the two source rows are compared.

Use the NameCert obligation surface. The certificate exposes the role correspondence through signature comparison, classifier acceptance, package admission, ledger preservation, replay, continuation, non-finality, and local naming.

Finally apply non-escape. The packet has no coordinate for source identity, host validation, or hidden self-study, so the role bridge is ledgered through the displayed rows rather than through identity of P and F. □

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiry ledgered role correspondence`

Theorem 1151.8 (Reflective inquiry TasteGate obligation closure)**.** *The finite packet rows already exposed by Definition 1151.1, Definition 1151.3, Definition 1151.4, Definition 1151.5, Definition 1151.6, and Definition 1151.7 are exactly the rows consumed by the TasteGate single-carrier alignment for* $\mathsf{ReflectiveInquiry}^{\uparrow}$*. Hence any obligation-closure read of this packet must cite the TasteGate marker before moving beyond seed closure, and that citation verifies single-carrier alignment rather than creating a paper-side closure consequence.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryTasteGate single carrier alignment`

Proof. Project the carrier $I = (P, F, S, K, A, R, L, H, C, Q, N)$. The carrier and classifier rows expose physical source access, formal source access, signature comparison, role classification, package admission, transport, ledger preservation, replay history, continuation, open-continuation non-finality, and local naming.

Use the NameCert obligation surface together with non-escape and source separation. These results show that the same rows are the only admissible local certificate surface, that no final truth oracle or source identity can be exported, and that P and F remain separate under classifier agreement.

Use open-continuation preservation and ledgered role correspondence. They keep the inquiry route non-final while forcing the role bridge through $S, K, A, R, L, H, C, Q, N$ and the displayed source rows P, F.

The Lean TasteGate theorem is the verification touchpoint for single-carrier alignment on precisely this finite row list. It is therefore the marker required before any obligation-closure read moves past seed closure; it does not by itself upgrade the paper closure axis. □

Theorem 1151.9 (Reflective inquiry ground-compiler obligation route)**.** *The TasteGate alignment of Definition 1151.8 feeds the ground compiler exactly the displayed* BHist *rows*

$$P, F, S, K, A, R, L, H, C, Q, N.$$

The physical source row P, formal source row F, signature row S, classifier row K, package row A, transport row R, ledger row L, replay row H, continuation row C, open-continuation row Q, and local naming row N are the only carrier rows available to a compiler-side obligation read.

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryGroundCompilerObligationRoute`

Proof. Project the carrier of Definition 1151.1. Its coordinates are already BHist rows, so a ground-compiler read receives row data rather than a prose description of inquiry.

Use Definition 1151.8. The TasteGate marker verifies single-carrier alignment for the same finite row list, which is the input surface the ground compiler may inspect.

Finally apply non-escape and ledgered role correspondence. They prevent the compiler route from adding a final truth oracle, source-domain identity, host validation row, or unledgered self-study claim outside $P, F, S, K, A, R, L, H, C, Q, N$. □

Theorem 1151.10 (Reflective inquiry FieldFaithful row separation)**.** *The carrier rows used by any future FieldFaithful witness for* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *are separated by role: P and F are source rows, S and K are comparison and acceptance rows, A is package admission, R is componentwise transport, L is ledger preservation, H and C are replay and continuation rows, Q is open-continuation non-finality, and N is local naming. No two of these roles are identified by the classifier.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryFieldFaithfulRowSeparation`

Proof. Read the classifier of Definition 1151.2. It compares accepted carriers componentwise through the displayed row names, so each row role remains visible under classifier agreement.

Apply source separation and open-continuation preservation. Source separation keeps P and F distinct, while open-continuation preservation keeps Q separate from replay H and continuation C.

Use the obligation surface for the remaining roles. Signature, classifier, package, transport, ledger, replay, continuation, non-finality, and naming are all named separately in the local $\mathsf{NameCert}_{\mathsf{ReflectiveInquiry}^{\uparrow}}$ surface, which is exactly the row separation a FieldFaithful witness must preserve. □

Theorem 1151.11 (Reflective inquiry nontrivial witness route)**.** *There are two accepted* $\mathsf{ReflectiveInquiry}^{\uparrow}$ *packet shapes with the same physical source, formal source, signature, classifier, package, transport, ledger, replay, continuation, and local naming rows, but with distinct open-continuation rows. They are displayed as*

$$I_0 = (P, F, S, K, A, R, L, H, C, Q_0, N)$$

and

$$I_1 = (P, F, S, K, A, R, L, H, C, Q_1, N),$$

where Q_0 records one admissible further-inquiry opening and Q_1 records a different admissible further-inquiry opening.

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryNontrivialWitnessRoute`

Proof. Start from the carrier definition. The open-continuation row is an ordinary displayed coordinate, so replacing Q by another accepted open-continuation row yields a packet shape with the same source, bridge, ledger, replay, continuation, and naming surface.

The two packets remain accepted because each keeps the non-finality obligation in the same coordinate. The distinction is not a change of physical source, formal source, classifier, package, ledger, replay, or local name; it is exactly the difference between Q_0 and Q_1.

Apply open-continuation preservation and non-escape. Both rows refuse a final truth oracle and keep inquiry open, while the packet exposes no hidden coordinate that could collapse the two accepted openings into host equality. □

Falsifiable prediction: An accepted $\mathsf{ReflectiveInquiry}^{\uparrow}$ packet cannot export a final truth oracle or identify physical reality with MetaCIC without losing either the physical-source row or the formal-source row.

Independence witness: observationreflectionpacket, physicalmodelaudit, reflectivestability, finite-traceevaluator, truthtotalreflection: none is bijective because $\mathsf{ReflectiveInquiry}^{\uparrow}$ carries both physical and formal source rows together with role correspondence, ledger preservation, and open-continuation non-finality.

The Lean follow-up in `lean4/BEDC/Derived/ReflectiveInquiryUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{ReflectiveInquiry}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ReflectiveInquiry}^{\uparrow}$ instances preserving $P, F, S, K, A, R, L, H, C, Q, N$.

Theorem 1151.12 (Reflective inquiry obligation-closure package)**.** *The displayed rows*

$$P, F, S, K, A, R, L, H, C, Q, N$$

form the complete obligation-closure package consumed by ReflectiveInquiry$^{\uparrow}$*. The package exposes carrier and classifier data, non-escape, source separation, open-continuation preservation, ledgered role correspondence, TasteGate single-carrier alignment, and the paper Nontrivial and FieldFaithful routes, all over the same finite* BHist *carrier.*

Lean4 ✓ `BEDC.Derived.ReflectiveInquiryUp.ReflectiveInquiryObligationClosurePackage`

Proof. Project the carrier of Definition 1151.1. The classifier of Definition 1151.2 reads exactly the rows $P, F, S, K, A, R, L, H, C, Q, N$, so the carrier and classifier exposure use the same finite packet.

Use the named boundary theorems. Theorem 1151.3 supplies the local NameCert surface, Definition 1151.4 supplies non-escape, Definition 1151.5 supplies source separation, and Definition 1151.6 keeps both inquiry routes open.

Add the correspondence and gate rows. Theorem 1151.7 forces the physical/formal bridge through the ledgered rows, while Definition 1151.8 records TasteGate single-carrier alignment for the same row list.

The paper Nontrivial and FieldFaithful routes are exactly the two displayed row routes of Definition 1151.11 and Definition 1151.10. Since no row remains for a final truth oracle, source identity, host validation, or unledgered self-study, no such obligation can enter the package. □

Closure status for NameCert$_{\text{ReflectiveInquiry}^{\uparrow}}$**.**

Constructive story: A ReflectiveInquiry$^{\uparrow}$ packet is generated from physical source access, formal source access, signature comparison, role classification, package admission, componentwise transport, ledger preservation, replay history, displayed continuation, open-continuation non-finality, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ReflectiveInquiry}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1151.1, Definition 1151.2, Definition 1151.3, Definition 1151.4, Definition 1151.8, Definition 1151.10, Definition 1151.11, and Definition 1151.9 bind the scoped finite inquiry packet through carrier, classifier audit, NameCert obligations, non-escape, TasteGate single-carrier alignment, FieldFaithful row separation, Nontrivial witness route, and ground-compiler obligation route.

Formal verification: FormalStatus(NameCert$_{\text{ReflectiveInquiry}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final truth, source-domain identity, host validation, Lean verification, or bridge checking for ReflectiveInquiry$^{\uparrow}$.

Upgrade path: Public closure requires a public certificate surface for the scoped finite inquiry packet, with the same $P, F, S, K, A, R, L, H, C, Q, N$ rows exported without final truth, source-domain identity, or host validation.

Chapter 1152

A Concrete Naming Certificate for $\mathsf{OnticResidueLedger}^{\uparrow}$

The $\mathsf{OnticResidueLedger}^{\uparrow}$ packet realizes the representation-ledger boundary of Definition 9414.282 as a finite NameCert carrier. It sits after the ontic-source surface of chapter 1125, the audit rows of chapter 1129, the finite model-state carrier of chapter 329, the observer-access support row of chapter 1121, and the current observer-state row of chapter 578. Its role is to store signature exposure, observer access, model comparison, residue, transport, route replay, provenance, and local naming before any consumer treats a model packet as exhausting the ontic source.

Definition 1152.1 (Ontic residue ledger carrier)**.** *An* $\mathsf{OnticResidueLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, M, S, A, C, R, H, T, P, N)$$

with the following displayed rows:

1. *O is the accepted* $\mathsf{OnticState}^{\uparrow}$ *source row whose reality-facing signature content is being read.*
2. *M is the accepted* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row carrying the audited model signatures, source traces, classifier, failure surface, and strength rows.*
3. *S is the accepted* $\mathsf{StateSpaceModel}^{\uparrow}$ *row naming the finite model-state packet offered as a representation.*
4. *A is the observer-access row, read through* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *and* $\mathsf{ConsciousObserverState}^{\uparrow}$*, that records which observer-local support surface may inspect the source and model signatures.*
5. *C is the signature classifier row comparing the exposed ontic signatures from O with the model-state and model-audit signatures from S and M.*
6. *R is the residue row recording source content, observer-access limits, failed model signatures, and audit-visible gaps not represented by S.*
7. *H records componentwise* hsame *transport for $O, M, S, A, C, R, T, P, N$.*
8. *T records displayed* Cont *routes from the ontic source to observer access, from observer access to model comparison, and from the accepted ledger to downstream adequacy or audit consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{OnticResidueLedger}^{\uparrow}$*,* $\mathsf{OnticState}^{\uparrow}$*,* $\mathsf{PhysicalModelAudit}^{\uparrow}$*,* $\mathsf{StateSpaceModel}^{\uparrow}$*,* $\mathsf{ObserverBudgetSupport}^{\uparrow}$*,* $\mathsf{ConsciousObserverState}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OnticResidueLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for model-state exhaustion, host-level reality access, unbounded observation, detached observer authority, quotienting away residue, or audit acceptance outside the displayed packet.

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.TasteGate.OnticResidueLedgerTasteGate single carrier alignment`

Definition 1152.2 (Ontic residue ledger classifier)**.** *The* $\mathsf{OnticResidueLedger}^{\uparrow}$ *classifier compares two accepted carriers only by the displayed rows* $O, M, S, A, C, R, H, T, P, N$*. It requires the ontic source row, model-audit row, model-state row, observer-access row, signature classifier row, and residue row to remain separately visible under componentwise* hsame *transport and displayed* Cont *replay.*

Two carriers are therefore classified together only when they expose the same source signatures, the same audited model comparison surface, the same finite state-space representation, the same observer-access budget and current-state support, and the same residue ledger. Agreement of the model-state row alone is not enough.

Theorem 1152.3 (Ontic residue ledger NameCert obligations)**.** *For the carrier of Definition 1152.1, the local* $\mathsf{NameCert}_{\mathsf{OnticResidueLedger}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier exposure for the packet* $L = (O, M, S, A, C, R, H, T, P, N)$*;*
2. *source-and-audit exposure through* O *and* M*;*
3. *model-state and observer-access exposure through* S *and* A*;*
4. *signature exactness and residue visibility through* C *and* R*;*
5. *non-escape from model-state exhaustion, unbounded observation, detached observer authority, quotienting away residue, host-level reality access, and audit acceptance outside* H, T, P, N*.*

Thus the ledger is accepted only as a finite source, audit, model, observer-access, classifier, residue, transport, route, provenance, and local-name packet.

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.OnticResidueLedgerTasteGate single carrier alignment`

Proof. Carrier exposure is projection from the displayed tuple in Definition 1152.1. The rows O, M, S, A, C, R supply the source, audit, model-state, observer-access, classifier, and residue coordinates, while H, T, P, N supply transport, route replay, provenance, and local naming.

The source-and-audit obligation is read from O and M. The model-state and observer-access obligation is read from S and A, so a consumer cannot inspect the model representation without also recording the observer-local support surface that performs the inspection.

Signature exactness and residue visibility are then separated. The classifier row C records the admitted comparison between exposed source signatures and model signatures; the residue row R records what the state-space model and audit rows do not represent.

The final obligation follows by inspecting the remaining structural rows. Componentwise transport, route replay, provenance, and local naming can replay only the displayed packet, so they cannot add model-state exhaustion, unbounded observation, detached observer authority, host reality access, or a route that erases the residue row. □

Theorem 1152.4 (Ontic residue ledger signature exactness)**.** *Every accepted* $\mathsf{OnticResidueLedger}^{\uparrow}$ *packet reads signature exactness through the triple* (O, M, C) *together with the model-state row* S *and observer-access row* A*. The exactness claim is not a bare equality between the ontic source and* $\mathsf{StateSpaceModel}^{\uparrow}$*; it is a ledgered comparison whose residue row* R *remains part of the accepted packet.*

Proof. Project the accepted carrier as $L = (O, M, S, A, C, R, H, T, P, N)$. The row O supplies the ontic source signatures, M supplies the audited model signatures and failure surface, and S supplies the finite state-space model offered as representation.

The observer-access row A records how those signatures may be inspected from $\mathsf{ObserverBudgetSupport}^{\uparrow}$ and $\mathsf{ConsciousObserverState}^{\uparrow}$ surfaces. The classifier C can therefore compare only signatures that have passed through the displayed source, audit, model, and observer-access rows.

The residue row R remains visible after the comparison. Consequently the exactness claim never collapses into source-model identity; it is the accepted comparison plus the displayed account of what the model row does not represent. □

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.TasteGate.OnticResidueLedgerSignatureResidue row nonescape`

Theorem 1152.5 (Ontic residue ledger nonescape)**.** *An accepted* $\mathsf{OnticResidueLedger}^{\uparrow}$ *packet cannot be reduced to* $\mathsf{StateSpaceModel}^{\uparrow}$ *or to* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *without losing either the observer-access row A or the residue row R. Every consumer-visible read of the ledger factors through*

$$O, M, S, A, C, R, H, T, P, N.$$

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.OnticResidueLedgerObserverAccess row nonescape`

Proof. The state-space row S records only the finite model-state representation. It does not contain the ontic source row O, the observer-access row A, the signature classifier row C, or the residue row R.

The physical audit row M records finite audit evidence for a candidate model. It does not itself supply the full observer-access support row or the local ledger that keeps represented and unrepresented source content apart.

The remaining rows H, T, P, N preserve the same displayed packet under transport, replay, provenance, and local naming. Since no row outside $O, M, S, A, C, R, H, T, P, N$ is exported, any reduction that forgets A or R leaves the accepted $\mathsf{OnticResidueLedger}^{\uparrow}$ surface. □

The Lean follow-up `lean4/BEDC/Derived/OnticResidueLedgerUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{OnticResidueLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{OnticResidueLedger}^{\uparrow}$ instances preserving $O, M, S, A, C, R, H, T, P, N$ as separate displayed fields.

Falsifiable prediction: An accepted $\mathsf{OnticResidueLedger}^{\uparrow}$ packet cannot be reduced to $\mathsf{StateSpaceModel}^{\uparrow}$ without losing either the observer-access row or the residue row.

Independence witness: onticstate, physicalmodelaudit, statespacemodel, observerbudgetsupport, consciousobserverstate: none is bijective to $\mathsf{OnticResidueLedger}^{\uparrow}$ because the ledger stores source signatures, audited model comparison, finite model-state representation, observer-local access, residue, transport, route, provenance, and local NameCert rows as one packet.

Theorem 1152.6 (Ontic residue ledger scoped consumer surface)**.** *Every accepted* $\mathsf{OnticResidueLedger}^{\uparrow}$ *consumer factors through the displayed packet*

$$O, M, S, A, C, R, H, T, P, N.$$

The consumer surface is exactly the carrier and classifier data of Definition 1152.1 and Definition 1152.2, the local NameCert obligation surface of Definition 1152.3, the signature exactness surface of Definition 1152.4, the nonescape boundary of Definition 1152.5, the checked TasteGate row `BEDC.Derived.OnticResidueLedgerUp.taste_gate`*, and the Lean-side Nontrivial and FieldFaithful witnesses preserving the ten rows as separate displayed fields.*

Proof. Project an accepted consumer to the carrier of Definition 1152.1. The only exported coordinates are $O, M, S, A, C, R, H, T, P, N$, and the classifier of Definition 1152.2 compares accepted carriers only through those displayed rows.

Apply Definition 1152.3 to read the local NameCert surface. Its obligations expose carrier, source-and-audit, model-state, observer-access, signature, residue, transport, route, provenance, and local naming data, with no additional coordinate for a stronger consumer.

The exactness and boundary claims then close the surface. By Definition 1152.4, signature comparison remains ledgered through O, M, C, S, A with R still visible. By Definition 1152.5, reducing the consumer to $\mathsf{StateSpaceModel}^{\uparrow}$ or $\mathsf{PhysicalModelAudit}^{\uparrow}$ loses either observer access or residue.

Finally the TasteGate row and the Lean-side Nontrivial and FieldFaithful witnesses record that the ten displayed fields remain separately visible. Therefore no accepted consumer may replace the packet by state-space exhaustion, host reality access, unbounded observation, detached observer authority, or quotienting away residue. □

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.TasteGate.OnticResidueLedgerScoped consumer surface`

Theorem 1152.7 (Ontic residue ledger bridge export)**.** *The public* $\mathsf{OnticResidueLedger}^{\uparrow}$ *consumer packet*

$$O, M, S, A, C, R, H, T, P, N$$

exports a bridge-facing surface only through the ontic source row O, physical model audit row M, finite state-space model row S, observer-access row A, signature classifier C, residue ledger R, componentwise transport H, displayed route replay T, provenance P, local naming row N, and the checked TasteGate row `BEDC.Derived.OnticResidueLedgerUp.taste_gate`*. It exports no state-space exhaustion row, host reality row, unbounded observer authority, residue-erasing quotient, or Lean bridge check.*

Proof. Start with the scoped consumer surface of Definition 1152.6. It binds every accepted consumer to the displayed packet $O, M, S, A, C, R, H, T, P, N$ together with the checked TasteGate row.

Read those rows as the bridge-facing surface. The source, audit, model, observer-access, signature, residue, transport, route, provenance, and naming coordinates are exactly the rows already exposed by the public consumer packet, so the bridge-facing export does not add another carrier coordinate.

The excluded rows are absent from the carrier and from the scoped consumer theorem. Therefore the paper bridge surface records only the displayed export boundary; it does not claim Lean bridge checking. □

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.TasteGate.OnticResidueLedgerBridgeExport structural rows nonescape`

Theorem 1152.8 (Ontic residue ledger mature consumer surface)**.** *Every mature* $\mathsf{OnticResidueLedger}^{\uparrow}$ *consumer reads exactly the displayed rows*

$$O, M, S, A, C, R, H, T, P, N.$$

The carrier, classifier, local NameCert obligations, signature exactness, nonescape boundary, scoped consumer theorem, and bridge-export theorem already expose the same ten-row surface. Hence every adequacy, audit, or bridge-facing read of the ledger is a read of these rows, and no accepted consumer obtains a state-space exhaustion row, host reality row, unbounded observation row, detached observer authority row, or residue-erasing quotient row.

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.TasteGate.OnticResidueLedgerMature consumer surface`

Proof. Project the accepted carrier to Definition 1152.1. The only displayed rows are $O, M, S, A, C, R, H, T, P, N$, and Definition 1152.2 compares accepted carriers only through those rows.

The local NameCert obligations, signature exactness theorem, and nonescape theorem keep the same rows visible. They expose source, audit, model, observer-access, classifier, residue, transport, replay, provenance, and local naming data, while excluding state-space exhaustion, unbounded observation, detached observer authority, host-level reality access, and quotienting away residue.

The scoped consumer theorem and bridge-export theorem then identify the same row list as the consumer-facing and bridge-facing surface. Since the chapter has no carrier coordinate outside that list, every mature consumer of the ledger is confined to the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{OnticResidueLedger}^{\uparrow}}$ **.**

Constructive story: An $\mathsf{OnticResidueLedger}^{\uparrow}$ packet is generated from an ontic source row, physical model audit row, finite state-space model row, observer-access row, signature classifier, residue ledger, componentwise hsame transport, displayed Cont routes, Pkg provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.OnticResidueLedgerUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OnticResidueLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1152.6 and Definition 1152.7 bind the public consumer and bridge-facing scopes to the displayed rows $O, M, S, A, C, R, H, T, P, N$ from Definition 1152.1, Definition 1152.2, Definition 1152.3, Definition 1152.4, and Definition 1152.5.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OnticResidueLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total ontology, state-space exhaustion, host-level reality access, unbounded observation, detached observer authority, or quotienting away residue.

Upgrade path: The paper-side displayed surface is mature over $O, M, S, A, C, R, H, T, P, N$; Lean verification and bridge checking remain recorded on their separate axes.

Chapter 1153

A Concrete Naming Certificate for $\mathsf{OnticTower}^{\uparrow}$

The $\mathsf{OnticTower}^{\uparrow}$ packet realizes the certified ontic tower of section 9414.19 as a finite stage-refinement NameCert carrier. It consumes the stable source row of $\mathsf{OnticState}^{\uparrow}$ from chapter 1125, the indexed approximation tower of $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ from chapter 1127, the physical-signature classifier of $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ from **??**, the residue surface of $\mathsf{Residue}^{\uparrow}$ from chapter 125, and the observer support row of $\mathsf{ObserverBudgetSupport}^{\uparrow}$ from chapter 1121. These rows turn the schematic tower into a finite packet with visible stage, transition, continuity, residue, observer-budget, transport, replay, provenance, and local naming coordinates.

Definition 1153.1 (OnticTower carrier)**.** *An* $\mathsf{OnticTower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, A, S, R, B, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OnticState}^{\uparrow}$ *source row naming the stable reality-facing signatures at the base of the tower.*
2. *A is the* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *stage row, recording the accepted finite approximation stages.*
3. *S is the* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ *classifier row, recording the physical-signature fit used at each stage.*
4. *R is the* $\mathsf{Residue}^{\uparrow}$ *ledger row, recording content preserved, withheld, or still unresolved between stages.*
5. *B is the* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *observer-budget row, recording which observer-facing supports may read the stage data.*
6. *L is the ledger-continuity row, recording each stage transition as preservation, refinement, or explicit correction of earlier signature rows.*
7. *H is componentwise* hsame *transport for O, A, S, R, B, L.*
8. *C is the displayed* Cont *replay route through the stage and ledger rows.*
9. *P is* Pkg *provenance over* $\mathsf{OnticTower}^{\uparrow}$*, the five cited sibling packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}$ *row.*
The carrier has no coordinate for an imported limit object, hidden final ontology, host equality of physical states, quotient stream, countable-choice selector, or observer-independent access route.

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTowerTasteGate single carrier alignment`

Definition 1153.2 (OnticTower classifier)**.** *Two accepted* $\mathsf{OnticTower}^{\uparrow}$ *packets are classified as the same only when their* $\mathsf{OnticState}^{\uparrow}$ *source rows agree through the same stage row, their* RealityConstrainedSignatureFit *classifiers preserve the same physical-signature comparisons, their* $\mathsf{Residue}^{\uparrow}$ *ledgers record the same hidden and unresolved content, their* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *rows admit the same observer-facing reads, and their ledger-continuity rows identify the same sequence of preservation, refinement, and correction transitions under componentwise* hsame *transport. Agreement of a single terminal* $\mathsf{OnticState}^{\uparrow}$ *row is not enough to classify tower packets.*

Theorem 1153.3 (OnticTower stage refinement)**.** *Let* $T = (O, A, S, R, B, L, H, C, P, N)$ *be an accepted* $\mathsf{OnticTower}^{\uparrow}$ *packet. Every accepted transition between two tower stages factors through the displayed approximation row* A*, signature-fit row* S*, residue ledger* R*, observer-budget row* B*, and ledger-continuity row* L*. A transition that preserves only the terminal* $\mathsf{OnticState}^{\uparrow}$ *source row* O *while dropping any one of* A, S, R, B, L *is not an accepted stage refinement.*

Proof. Start from the carrier of Definition 1153.1. The stage data are not read as a bare source row; they are carried by O, A, S, R, B, L together with transport, replay, provenance, and local naming rows.

Use the classifier of Definition 1153.2. It compares packets by the joint source, approximation-stage, physical-signature, residue, observer-budget, and ledger-continuity rows, so a transition missing one of these rows cannot be classified as the same tower route.

The sibling chapters supply the individual surfaces: $\mathsf{OnticState}^{\uparrow}$ supplies the source, RealityConstrainedApproxima supplies indexed stage structure, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ supplies physical-signature comparison, $\mathsf{Residue}^{\uparrow}$ supplies ledgered residue, and $\mathsf{ObserverBudgetSupport}^{\uparrow}$ supplies observer-facing support. Repacking them through H, C, P, N gives the accepted refinement route and no omitted coordinate remains. □

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTower stage refinement`

Theorem 1153.4 (OnticTower NameCert obligation surface)**.** *For the carrier of Definition 1153.1, the local* $\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier exposure, stage-refinement preservation, ledger-continuity exposure, observer-budget support, componentwise transport, displayed replay, provenance, and local naming. It excludes imported tower limits, final total descriptions, quotient streams, selected host limits, host real equality, countable-choice selectors, and second ledger sources.*

Proof. Carrier admission is the displayed tuple of Definition 1153.1. Classifier exposure is Definition 1153.2, which requires the joint source, stage, signature-fit, residue, support, and continuity rows.

Stage-refinement preservation is Definition 1153.3. The theorem forces every transition through the same finite packet before a later stage can be read as preserving or refining an earlier stage.

Ledger-continuity and observer support are visible rows of the carrier, and the remaining H, C, P, N rows transport, replay, source, and name the packet. Since the carrier has no additional coordinate, the excluded limit, selector, quotient, host-equality, and second-ledger surfaces cannot be exported from the accepted NameCert. □

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTower namecert obligation surface`

Theorem 1153.5 (OnticTower observer-budget separation)**.** *Let $T = (O, A, S, R, B, L, H, C, P, N)$ be an accepted* $\mathsf{OnticTower}^{\uparrow}$ *packet. Any accepted stage-refinement read of T must expose the* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *row B together with the* $\mathsf{OnticState}^{\uparrow}$ *source row O, approximation-tower row A, signature-fit row S, residue row R, ledger-continuity row L, transport row H, replay row C, provenance row P, and local naming row N. If the observer-budget coordinate is removed, the remaining rows do not form a valid* $\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}$ *route to observer-independent stage access.*

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTower observer budget separation`

Proof. Start with the carrier of Definition 1153.1. The observer-budget coordinate is a named carrier row, not explanatory prose outside the packet.

Use the classifier of Definition 1153.2. It compares accepted tower packets through the shared observer-facing reads as well as through source, stage, signature-fit, residue, continuity, and structural rows, so a comparison that omits B is not the same classifier route.

Apply the stage-refinement theorem Definition 1153.3. Every accepted transition factors through A, S, R, B, L before it is read as preserving or refining tower stage data.

Finally apply the NameCert obligation surface of Definition 1153.4. That surface lists observer-budget support among the local obligations and explicitly excludes observer-independent access. The TasteGate marker binds the scoped paper witnesses while leaving bridge checking and the verification axis to the closurestatus block. □

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTower observer budget semantic name certificate`

Theorem 1153.6 (OnticTower obligation closure surface)**.** *The carrier rows of Definition 1153.1, the classifier of Definition 1153.2, the stage-refinement route of Definition 1153.3, the NameCert obligation surface of Definition 1153.4, the observer-budget separation of Definition 1153.5, and the displayed TasteGate marker on Definition 1153.1 form exactly the paper-side obligation surface needed before scoped closure. A scoped-closure claim must still bind those rows to checked Lean targets and bridge data.*

Lean4 ✓ `BEDC.Derived.OnticTowerUp.OnticTower obligation closure surface`

Proof. The carrier definition supplies the finite packet $T = (O, A, S, R, B, L, H, C, P, N)$ and excludes imported limits, hidden final ontology, host equality, quotient streams, countable-choice selectors, and observer-independent access.

The classifier and stage-refinement theorems then force every tower read through the same source, approximation, signature-fit, residue, observer-budget, ledger-continuity, transport, replay, provenance, and local naming rows.

The NameCert obligation theorem names the local certificate surface, and observer-budget separation shows that dropping B leaves the accepted route. The TasteGate marker attached to the carrier records the single-carrier alignment on the Lean side, but it does not by itself supply scoped closure.

Thus the listed rows are exactly the obligation surface: enough to state the seed-to-obligation path, and still short of the checked bindings and bridge data required for scoped closure. □

The Lean follow-up path is `lean4/BEDC/Derived/OnticTowerUp/TasteGate.lean`. It must provide explicit $\mathsf{Nontrivial}\ \mathsf{OnticTower}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{OnticTower}^{\uparrow}$ instances preserving the source, stage, signature-fit, residue, observer-budget, ledger-continuity, transport, replay, provenance, and local naming rows.

Falsifiable prediction: An accepted $\mathsf{OnticTower}^{\uparrow}$ packet cannot collapse to $\mathsf{OnticState}^{\uparrow}$ without losing either a refinement transition or a ledger-continuity row.

Independence witness: onticstate, realityconstrainedapproximationtower, realityconstrainedsignaturefit, residue, observerbudgetsupport: none is bijective because $\mathsf{OnticTower}^{\uparrow}$ binds source, finite stage refinement, physical-signature classification, residue ledger, observer support, ledger continuity, transport, replay, provenance, and local NameCert rows in one packet. Each named sibling exposes one adjacent surface without the complete tower transition and ledger-continuity packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}$.

Constructive story: An $\mathsf{OnticTower}^{\uparrow}$ packet is generated from OnticState source, approximation-stage, signature-fit, residue-ledger, observer-budget, ledger-continuity, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1153.1, Definition 1153.2, Definition 1153.3, Definition 1153.4, Definition 1153.5, Definition 1153.6, and `BEDC.Derived.OnticTowerUp.OnticTowerTasteGate_single_carrier_alignment` bind the carrier, classifier, stage-refinement, NameCert surface, observer-budget separation, obligation closure surface, and TasteGate scoped witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OnticTower}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, final total ontology, imported tower limits, quotient streams, selected host limits, host equality, countable-choice selectors, or observer-independent access.

Upgrade path: Scoped closure requires checked bindings for carrier admission, classifier exposure, stage-refinement preservation, ledger-continuity exposure, observer-budget support, and the Nontrivial and FieldFaithful TasteGate rows.

Chapter 1154

A Concrete Naming Certificate for $\mathsf{OnticResidueLattice}^{\uparrow}$

The $\mathsf{OnticResidueLattice}^{\uparrow}$ packet realizes the residue-activation and refined-bundle direction of section 9414.15 as a finite NameCert surface. It sits after the source packet $\mathsf{OnticState}^{\uparrow}$ of chapter 1125, the model-audit packet $\mathsf{PhysicalModelAudit}^{\uparrow}$ of chapter 1129, the state-space carrier $\mathsf{StateSpaceModel}^{\uparrow}$ of chapter 329, the support-and-budget packet $\mathsf{ObserverBudgetSupport}^{\uparrow}$ of chapter 1121, and the observation handoff $\mathsf{ObservationReflectionPacket}^{\uparrow}$ of chapter 1126. Its role is to store which residue rows become active under which refined bundles before any consumer treats a model row, audit row, or observation route as ontically exhaustive.

Definition 1154.1 (Ontic residue lattice carrier)**.** *An* $\mathsf{OnticResidueLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, A, M, S, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{OnticState}^{\uparrow}$ *source row carrying the reality-facing signature source and base residue ledger.*
2. *A is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row recording the audited model signatures, failure surface, strength, and audit ledger for the model under comparison.*
3. *M is the* $\mathsf{StateSpaceModel}^{\uparrow}$ *row naming the finite state-space presentation whose projections are being tested against residue activation.*
4. *S is the* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *row specifying which observer-local support and budget caps make the refined bundle admissible.*
5. *B is the* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *row recording the observation and reflection route through which the refined bundle is read.*
6. *R is the finite residue-activation lattice row. Its elements are pairs of a refined-bundle token and the residue rows activated by that token, ordered by inclusion of displayed activation evidence.*
7. *K is the classifier row comparing two packets by source status, audit comparison, model-state projection, budget support, observation route, and residue activation together.*
8. *H records componentwise* hsame *transport for $O, A, M, S, B, R, K, C, P, N$.*
9. *C records displayed* Cont *replay through $O, A, M, S, B, R, K, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{OnticResidueLattice}^{\uparrow}$, $\mathsf{OnticState}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OnticResidueLattice}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare source state, model-only equivalence, audit-only comparison, unbudgeted observation route, quotient physical object, or host-level ontology. Residue activation is public only as the displayed lattice row R transported with the refined-bundle rows.

Lean4 (definition) `BEDC.Derived.OnticResidueLatticeUp.taste gate`

Definition 1154.2 (Ontic residue lattice classifier)**.** *The* $\mathsf{OnticResidueLattice}^{\uparrow}$ *classifier identifies two accepted packets only when their* $\mathsf{OnticState}^{\uparrow}$ *source rows,* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *audit rows,* $\mathsf{StateSpaceModel}^{\uparrow}$ *projections,* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *support rows,* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *routes, and residue-activation lattice rows agree through the displayed componentwise* hsame *transport. Agreement of any proper sublist of those rows is not enough: the classifier is sensitive to which refined bundle activates which residue, not merely to source status, audit comparison, model state, observer support, or reflection routing in isolation.*

Theorem 1154.3 (Ontic residue lattice activation non-escape)**.** *Let $L = (O, A, M, S, B, R, K, H, C, P, N)$ be an accepted* $\mathsf{OnticResidueLattice}^{\uparrow}$ *carrier. Any public read that preserves the source row O, audit row A, model-state projection M, budget-support row S, and observation-reflection route B but drops the residue-activation lattice row R leaves the accepted classifier surface. Thus residue activation under refined bundles cannot be recovered from a bare* $\mathsf{OnticState}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, *or* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *row.*

Proof. Project the carrier of Definition 1154.1. The local name N is reached only after the source, audit, model-state, support, observation-route, activation-lattice, classifier, transport, replay, and provenance rows have all been displayed.

Apply the classifier of Definition 1154.2. It compares the refined-bundle activation row R together with O, A, M, S, B; it does not classify by any one sibling row or by their subtuple without the activation lattice.

Therefore a public read that drops R has no displayed row recording which refined bundle activated which residue. Since the carrier has no quotient physical object, host ontology, hidden selector, or unbudgeted observation coordinate, no off-packet route can reconstruct that activation relation. □

Lean4 ✓ `BEDC.Derived.OnticResidueLatticeUp.OnticResidueLattice activation nonescape`

Theorem 1154.4 (Ontic residue lattice NameCert obligation surface)**.** *For the carrier of Definition 1154.1, the local* $\mathsf{NameCert}_{\mathsf{OnticResidueLattice}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for $L = (O, A, M, S, B, R, K, H, C, P, N)$;*
2. *refined-bundle exposure through the joint* $\mathsf{OnticState}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, *and* $\mathsf{ObservationReflectionPacket}^{\uparrow}$ *rows O, A, M, S, B;*
3. *residue-activation classification through the lattice row R and classifier row K;*
4. *activation non-escape through componentwise transport and displayed replay along H, C;*
5. *provenance and local naming through P, N, with no hidden model-state, audit-only, source-only, budget-only, observation-only, quotient-object, or host-ontology coordinate.*

These five rows are the complete public obligation surface for using OnticResidueLattice$^{\uparrow}$ *as a residue-activation certificate under refined bundles.*

Proof. Carrier admission is the first projection from Definition 1154.1. Refined-bundle exposure is the joint visibility of O, A, M, S, B inside the same finite packet.

Residue-activation classification is Definition 1154.2: R and K record which residue rows are active under which displayed refined-bundle tokens. Activation non-escape follows from Definition 1154.3, which rules out recovering that row from any single sibling surface or from the sibling subtuple without R.

The rows H, C, P, N transport, replay, source, and name the same displayed packet. Since the carrier has no additional coordinate for a quotient physical object, host ontology, hidden selector, or unbudgeted observation route, the five listed rows exhaust the local NameCert obligation surface. □

The Lean follow-up path is `lean4/BEDC/Derived/OnticResidueLatticeUp/TasteGate.lean`. That file must provide explicit Nontrivial OnticResidueLattice$^{\uparrow}$ and FieldFaithful OnticResidueLattice$^{\uparrow}$ instances preserving $O, A, M, S, B, R, K, H, C, P, N$ as separate displayed fields, with the future target name BEDC.Derived.OnticResidueLatticeUp.taste_gate.

Falsifiable prediction: A model-state projection that drops the refined-bundle activation row cannot classify two OnticResidueLatticeUp packets as equivalent.

Independence witness: onticstate, physicalmodelaudit, statespacemodel, observerbudgetsupport, observationreflectionpacket: none is bijective because the lattice stores residue activation under refined bundles rather than only source status, audit comparison, model state, budget support, or reflection routing.

Closure status for NameCert$_{\text{OnticResidueLattice}^{\uparrow}}$.

Constructive story: An OnticResidueLattice$^{\uparrow}$ packet is generated from ontic-source, physical-model-audit, state-space-model, observer-budget-support, observation-reflection, residue-activation lattice, classifier, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{OnticResidueLattice}^{\uparrow}}$, seed)

Scope closed: Definition 1154.1, Definition 1154.2, Definition 1154.3, and Definition 1154.4 bind the finite residue-lattice carrier, classifier, activation non-escape boundary, and public NameCert obligation surface.

Formal verification: FormalStatus(NameCert$_{\text{OnticResidueLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, total physical ontology, quotient physical objects, unbudgeted observation, model-state sufficiency, audit-only sufficiency, or source-only sufficiency.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, activation non-escape, TasteGate nontriviality, and field faithfulness for $O, A, M, S, B, R, K, H, C, P, N$.

Chapter 1155

A Concrete Naming Certificate for $\mathsf{LawCertificate}^{\uparrow}$

The $\mathsf{LawCertificate}^{\uparrow}$ packet realizes the law-certificate discipline of section 9414.25 as a finite NameCert surface. It sits after the physical-induction carrier of chapter 1135, the open-fit packet of chapter 1137, the model-audit packet of chapter 1124, the induction-stability seal of chapter 1139, and the tower-compression packet of chapter 1142. Those siblings supply finite fit, audit comparison, continuation stability, and compression-residue surfaces; this packet binds an equation-like pattern to source fit, classifier, stability, failure surface, ledger, transport, provenance, and local naming before any consumer may read it as a physical law.

The packet is composite over chapter 1135, chapter 1137, chapter 1124, chapter 1139, and chapter 1142; it bundles those rows into one law-certificate surface.

Definition 1155.1 (Law certificate carrier)**.** *A* $\mathsf{LawCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, P, C, S, E, Ld, H, R, Q, N)$$

with the following displayed rows:

1. *F is the finite source-fit row, read through* $\mathsf{OpenFitPacket}^{\uparrow}$ *and* $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$*, recording the observations that license the equation-like pattern.*
2. *P is the pattern row naming the equation, invariant, or calculation rule being offered as lawlike.*
3. *C is the classifier row comparing model-generated and source-observed signatures under the admitted observational equivalence.*
4. *S is the continuation-stability row, read through* $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$*, naming the continuation class and stability basis under which reuse is licensed.*
5. *E is the explicit failure-surface row naming observations, refined histories, or tower-level reads that would defeat the law claim.*
6. *Ld is the ledger row recording idealizations, hidden assumptions, tower-compression residue, audit strength, and refused final-fit demands.*
7. *H records componentwise* hsame *transport for $F, P, C, S, E, Ld, R, Q, N$.*
8. *R records displayed* Cont *replay through source fit, pattern use, classifier comparison, stability reuse, failure exposure, and ledger inspection.*

9. *Q is the* Pkg *provenance over* $\mathsf{LawCertificate}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$, $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LawCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare equation, unbounded natural law, final-fit oracle, detached model syntax, hidden stability assumption, omitted failure surface, or ledger-free physical truth. A physical law is public only as the displayed packet L.

Lean4 (definition) `BEDC.Derived.LawCertificateUp.LawCertificateUp`

Definition 1155.2 (Law certificate classifier)**.** *The* $\mathsf{LawCertificate}^{\uparrow}$ *classifier identifies two accepted packets only when their source-fit rows, equation-pattern rows, classifier rows, continuation-stability rows, failure-surface rows, ledger rows, transport rows, replay rows, provenance rows, and local NameCert rows agree through displayed componentwise* hsame *transport. Agreement of the equation-like pattern alone is not enough, and finite source fit without stability or failure exposure is not enough.*

Lean4 (definition) `BEDC.Derived.LawCertificateUp.lawCertificateClassifier`

Theorem 1155.3 (Law certificate NameCert obligations)**.** *For the carrier of Definition 1155.1, the local* $\mathsf{NameCert}_{\mathsf{LawCertificate}^{\uparrow}}$ *surface consists exactly of seven finite obligations:*

1. *carrier admission for $L = (F, P, C, S, E, Ld, H, R, Q, N)$;*
2. *source-fit exposure through F;*
3. *equation-pattern exposure through P without treating the pattern as self-licensing;*
4. *classifier exposure through C;*
5. *continuation-stability and failure-surface exposure through S and E;*
6. *ledger accountability through Ld, including audit strength, compression residue, idealization, and refused final-fit demands;*
7. *transport, replay, provenance, and local naming through H, R, Q, N.*

These seven rows are the complete public obligation surface for reading an equation-like pattern as a physical law.

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificate semantic name certificate`

Proof. Carrier admission is projection from Definition 1155.1. The packet exposes source fit, pattern, classifier, continuation stability, failure surface, ledger, transport, replay, provenance, and local name as its complete finite surface.

Source-fit exposure comes from F, while pattern exposure comes from P. The classifier row C states how the model-generated signatures and source-observed signatures are compared; therefore the pattern is not read in isolation from the source and classifier data that make it empirical.

Continuation stability and failure exposure are supplied by S and E. The ledger row Ld records the audit, tower, idealization, and refusal data that prevent the packet from becoming an unbounded natural-law oracle.

Finally H, R, Q, N preserve and name the same displayed packet. Since the carrier has no further coordinate for hidden stability, detached model syntax, or ledger-free truth, the seven listed obligations exhaust the local NameCert surface. □

Theorem 1155.4 (Law certificate failure-surface nonescape)**.** *An accepted* $\mathsf{LawCertificate}^{\uparrow}$ *packet cannot be reduced to an equation pattern plus finite fit while preserving law status. Any public read that drops either the continuation-stability row* S *or the failure-surface row* E *leaves the accepted classifier surface.*

Lean4 ✓ `BEDC.Derived.LawCertificateUp.lawCertificate failure surface rows nonescape`

Proof. Project the carrier as $L = (F, P, C, S, E, Ld, H, R, Q, N)$. The rows F and P record finite source fit and the equation-like pattern, but they do not state which continuations are licensed or what observations would defeat the law claim.

Apply the classifier of Definition 1155.2. It compares accepted packets through S and E together with F, P, C, Ld, H, R, Q, N. Dropping either S or E therefore changes the public comparison surface.

Transport, replay, provenance, and local naming preserve only the displayed rows. Because the carrier has no hidden coordinate for stability or failure exposure, no off-packet route can recover law status from equation syntax and finite fit alone. □

Theorem 1155.5 (Law certificate continuation scope)**.** *Every accepted* $\mathsf{LawCertificate}^{\uparrow}$ *consumer read of a physical law is scoped to the continuation-stability row* S*, the failure-surface row* E*, and the ledger row* Ld*. The packet licenses prediction only inside that displayed scope and does not assert total future fit.*

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificate continuation scope`

Proof. The NameCert obligations of Definition 1155.3 require S, E, Ld before the local name is accepted. These rows state the continuation class, the defeating observations, and the audit-and-residue ledger that bound reuse.

The nonescape result of Definition 1155.4 rules out reading law status after removing either continuation stability or failure exposure. The ledger row also remains visible, so idealization residue and tower-compression assumptions are part of the same public packet.

Thus the consumer-visible law is not a total claim over all possible observation histories. It is a certificate-relative read whose scope is exactly the displayed source fit, classifier, stability, failure surface, ledger, transport, provenance, and local NameCert rows. □

Theorem 1155.6 (Law certificate carrier admission)**.** *Every accepted* $\mathsf{LawCertificate}^{\uparrow}$ *carrier admits the row tuple*

$$L = (F, P, C, S, E, Ld, H, R, Q, N)$$

only as the finite packet of Definition 1155.1: F *supplies source fit,* P *supplies the equation-like pattern,* C *supplies classifier comparison,* S *supplies continuation stability,* E *supplies the failure surface,* Ld *supplies ledger accountability, and* H, R, Q, N *supply transport, replay, provenance, and local naming. The TasteGate surface records nontrivial field separation for those rows before the packet can be read as a law certificate.*

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateCarrier admission`
↪ variant `BEDC.Derived.LawCertificateUp.LawCertificateCarrier carrier admission`

Proof. Project the carrier definition and the NameCert obligations. The carrier definition lists exactly $F, P, C, S, E, Ld, H, R, Q, N$, while Definition 1155.3 makes those rows the public obligation surface.

Use the TasteGate field-faithfulness witness represented by the LawCertificateUp TasteGate file. It keeps the displayed rows separate, so source fit, equation pattern, classifier, stability, failure surface, and ledger data cannot collapse into a single equation token.

Finally replay through H, R, Q, N. Since every public read is transported, replayed, sourced, and named through the same tuple, carrier admission is finite row admission and not an unbounded physical-law oracle. □

Theorem 1155.7 (Law certificate classifier stability)**.** *If two accepted* $\mathsf{LawCertificate}^{\uparrow}$ *packets agree through the displayed classifier of Definition 1155.2, then every law-certificate consumer route preserves the same rows* $F, P, C, S, E, Ld, H, R, Q, N$*. In particular, agreement of* P *alone cannot transport law status unless* $F, C, S, E, Ld, H, R, Q, N$ *are transported with it.*

Proof. Start from the classifier block. It compares accepted packets through source fit, pattern, classifier, continuation stability, failure surface, ledger, transport, replay, provenance, and local NameCert rows.

Apply the continuation-scope theorem Definition 1155.5. A consumer read is scoped by S, E, Ld, so classifier transport must preserve those rows along with F, P, C.

The structural rows H, R, Q, N then preserve the route. The TasteGate field-faithfulness surface prevents a packet whose equation pattern matches but whose stability, failure, or ledger rows differ from being read as the same accepted law certificate. □

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateCarrier classifier stability`

Theorem 1155.8 (Law certificate ledger nonescape)**.** *Every accepted* $\mathsf{LawCertificate}^{\uparrow}$ *consumer of a physical law must keep the ledger row* Ld *visible with* $F, P, C, S, E, H, R, Q, N$*. A route that reads source fit, equation pattern, classifier, continuation stability, or failure exposure while dropping* Ld *leaves the accepted NameCert surface and cannot export ledger-free law status.*

Proof. Use the NameCert obligations. They make ledger accountability through Ld one of the finite rows required before the local name is public.

Apply the failure-surface and continuation-scope theorems, Definition 1155.4 and Definition 1155.5. These results keep S, E, Ld visible whenever the law is consumed, so the ledger cannot be omitted after the classifier has accepted F, P, C.

Finally use carrier admission and TasteGate field separation. The rows H, R, Q, N preserve the same packet, and no coordinate outside $F, P, C, S, E, Ld, H, R, Q, N$ restores a missing ledger row. □

Lean4 ✓ `BEDC.Derived.LawCertificateUp.taste gate`

Theorem 1155.9 (Law certificate scope transport)**.** *Every accepted* $\mathsf{LawCertificate}^{\uparrow}$ *consumer route preserves the displayed rows*

$$F, P, C, S, E, Ld, H, R, Q, N$$

through source fit, classifier comparison, continuation stability, failure-surface exposure, ledger accountability, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *scope. No equation-only route and no ledger-free route exports law status.*

Proof. Project the carrier and NameCert obligations. They make $F, P, C, S, E, Ld, H, R, Q, N$ the complete finite surface for source fit, pattern exposure, classifier comparison, continuation stability, failure surface, ledger accountability, transport, replay, provenance, and local naming.

Apply the continuation-scope theorem and failure-surface nonescape theorem. A consumer route that keeps law status must preserve S and E together with the rows that source, classify, and transport the same packet.

Apply classifier stability and ledger nonescape. Agreement of the equation pattern alone cannot transport law status, and dropping Ld leaves the accepted surface. Thus every accepted route remains scoped by the displayed rows and exports neither equation-only nor ledger-free law status. □

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateCarrier scope transport`

Theorem 1155.10 (Law certificate public export route)**.** *Every public downstream consumer of an accepted* $\mathsf{LawCertificate}^{\uparrow}$ *packet receives only the displayed law rows* F, P, C, S, E*, the classifier-stability and ledger rows* C, S, E, Ld*, componentwise* hsame *transport* H*, displayed* Cont *replay* R*,* Pkg *provenance* Q*, and the local* $\mathsf{NameCert}$ *row* N*. The public export route contains no bare equation token, hidden stability premise, ledger-free law status, final-fit oracle, detached model syntax, or unbounded natural-law claim.*

Proof. Project the carrier $L = (F, P, C, S, E, Ld, H, R, Q, N)$ and the NameCert obligations. They identify the complete row surface for source fit, equation pattern, classifier comparison, continuation stability, failure exposure, ledger accountability, transport, replay, provenance, and local naming.

Apply classifier stability and scope transport, Definition 1155.7 and Definition 1155.9. These keep every accepted consumer route on the same displayed rows and prevent equation-pattern agreement from transporting law status by itself.

Finally use ledger nonescape and continuation scope, Definition 1155.8 and Definition 1155.5. They keep Ld, S, E visible in every law read, so a public downstream consumer receives only the displayed packet and no hidden stability, ledger-free truth, detached model syntax, final-fit oracle, or unbounded physical-law claim. □

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateCarrier public export route`

Theorem 1155.11 (Law certificate bridge-carrier route)**.** *Every bridge-facing consumer of an accepted* $\mathsf{LawCertificate}^{\uparrow}$ *packet must preserve*

$$F, P, C, S, E, Ld, H, R, Q, N$$

through the same source-fit, classifier, continuation-stability, failure-surface, ledger, transport, replay, provenance, and local $\mathsf{NameCert}$ *rows before any abstract law-certificate interface can consume it. A bridge read that drops one of these rows has not transported the public* $\mathsf{LawCertificate}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateCarrier bridge carrier route`

Proof. Project the public export route of Definition 1155.10. It already makes $F, P, C, S, E, Ld, H, R, Q, N$ the complete public surface for source fit, equation pattern, classifier comparison, continuation stability, failure exposure, ledger accountability, transport, replay, provenance, and local naming.

Apply scope transport and ledger nonescape. The bridge-facing consumer must keep the accepted carrier inside the same componentwise hsame and displayed Cont route, and it cannot replace the ledger row Ld by an equation-only or stability-only law read.

Finally read the taste-gate marker as the carrier-side field-faithfulness witness for the same displayed rows. The abstract law-certificate interface therefore consumes only the preserved packet, not a detached equation token, hidden continuation premise, ledger-free law status, final-fit oracle, or unbounded natural-law claim. □

Lean4 ✓ `BEDC.Derived.LawCertificateUp.LawCertificateBridgeCarrierRoute`

Theorem 1155.12 (Law certificate mature consumer route)**.** *Every downstream consumer of an accepted* LawCertificate$^{\uparrow}$ *packet must preserve the source-fit row F, equation-pattern row P, classifier row C, continuation-stability row S, failure-surface row E, ledger row Ld, componentwise* hsame *transport row H, displayed* Cont *replay row R,* Pkg *provenance row Q, and local* NameCert *row N. Such a consumer may not read the packet as an equation-only law, total prediction, or final-fit oracle.*

Proof. Start with the carrier and classifier of Definition 1155.1 and Definition 1155.2. They make $F, P, C, S, E, Ld, H, R, Q, N$ the complete accepted comparison surface, so no downstream route begins from the equation pattern alone.

Apply the NameCert obligations, failure-surface nonescape, and continuation-scope theorems, namely Definition 1155.3, Definition 1155.4, and Definition 1155.5. These results keep source fit, classifier comparison, stability, failure exposure, and ledger accountability visible whenever law status is consumed.

Finally apply the public export route and bridge-carrier route of Definition 1155.10 and Definition 1155.11. They transport only the displayed packet through hsame, Cont, Pkg, and local naming, and therefore exclude an equation-only law, total prediction, detached final-fit oracle, hidden stability premise, or ledger-free physical truth. □

Together with the existing TasteGate marker for `BEDC.Derived.LawCertificateUp.taste_gate`, the checked bridge-carrier route records the bridge evidence needed for paper-side bridged closure: the abstract interface consumes only the displayed carrier rows and their field-faithful separation.

The Lean follow-up file `lean4/BEDC/Derived/LawCertificateUp/TasteGate.lean` must provide explicit Nontrivial LawCertificate$^{\uparrow}$ and FieldFaithful LawCertificate$^{\uparrow}$ instances, with witness-pair and field-projection proofs preserving $F, P, C, S, E, Ld, H, R, Q, N$ as separate displayed fields.

Falsifiable prediction: A packet that presents an equation and finite fit but omits either the continuation-stability row or the failure-surface row is rejected by the LawCertificateUp classifier.

Independence witness: realityconstrainedphysicalinduction, openfitpacket, realityconstrainedmodelaudit, physicalinductionstabilityseal, realityconstrainedtowercompression: none is bijective because LawCertificateUp binds equation pattern, source fit, classifier, continuation stability, failure surface, ledger, transport, provenance, and local NameCert rows in one carrier, while each sibling supplies only a constituent surface.

Closure status for NameCert$_{\text{LawCertificate}^{\uparrow}}$**.**

Constructive story: A LawCertificate$^{\uparrow}$ packet is generated from finite source-fit, equation-pattern, classifier, continuation-stability, failure-surface, ledger, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{LawCertificate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1155.1, Definition 1155.2, Definition 1155.3, Definition 1155.4, Definition 1155.5, Definition 1155.6, Definition 1155.7, Definition 1155.8, Definition 1155.9, Definition 1155.10, Definition 1155.11, and Definition 1155.12 bind the finite law-certificate carrier, classifier, obligation surface, failure-surface boundary, continuation scope, carrier admission, classifier stability, ledger nonescape, accepted consumer-route scope, public export route, checked bridge-carrier route, and mature downstream consumer route.

Formal verification: FormalStatus(NameCert$_{\text{LawCertificate}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not close Lean verification, bridge checking, total physical law, all-history prediction, equation-only law status, hidden stability, ledger-free physical truth, or final-fit oracle behavior.

Upgrade path: The paper-side closure axis is complete at the mature consumer route; Lean-side verification remains independent of the bridge construction recorded here.

Chapter 1156

A Concrete Naming Certificate for $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$

The $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ packet realizes the record-and-gap reading of physical laws in subsection 9462.28.4 as a finite NameCert surface. It sits before the law-certificate packet of chapter 1155 and beside the physical-induction carrier of chapter 1135, the open-fit packet of chapter 1137, the model-audit packet of chapter 1124, and the induction-stability seal of chapter 1139. Its role is to keep observed regularity and future-persistence commitment in one packet without identifying either side with a bare equation or a total law oracle.

Definition 1156.1 (Empirical regularity persistence carrier)**.** *An* $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, R, K, L, G, A, S, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite measurement row naming the observed cases, instruments, units, and admissible sampling window.*
2. *R is the records-side regularity row stating the relation satisfied by the observed rows under M.*
3. *K is the classifier row comparing two regularity packets only through their displayed measurements, records, tolerance ledgers, GAP rows, stability rows, failure rows, transports, continuations, provenance, and local names.*
4. *L is the finite error-ledger row recording tolerances, idealizations, calibration residue, and excluded observations.*
5. *G is the Type IV future-persistence GAP row asserting that the records-side regularity is offered for not-yet-observed continuations.*
6. *A is the* $\mathsf{LawCertificate}^{\uparrow}$ *consumption row naming the law-certificate surface that may read the packet.*
7. *S is the stability row, read against* $\mathsf{PhysicalInductionStabilitySeal}^{\uparrow}$ *and* $\mathsf{RealityConstrainedPhysicalInduction}^{\uparrow}$*, naming the continuation class and persistence basis under which G is licensed.*
8. *F is the failure row naming observations, continuations, refined measurements, or audit rows that defeat the persistence claim.*

9. *H records componentwise* hsame *transport for $M, R, K, L, G, A, S, F, C, P, N$.*
10. *C records displayed* Cont *replay through measurement, regularity, classifier, error ledger, future-persistence GAP, law-certificate consumption, stability, and failure rows.*
11. *P is the* Pkg *provenance over* EmpiricalRegularityPersistence$^{\uparrow}$, LawCertificate$^{\uparrow}$, RealityConstrainedPhysicalInductio OpenFitPacket$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, PhysicalInductionStabilitySeal$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{EmpiricalRegularityPersistence}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare equation, hidden measurement row, omitted tolerance ledger, persistence claim without G, all-future truth, final-fit oracle, host physical necessity, or law status detached from A, S, F, H, C, P, N.

Lean4 (definition) BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceUp
Lean4 ✓ BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceTasteGate
single carrier alignment

Definition 1156.2 (Empirical regularity persistence classifier)**.** *The* EmpiricalRegularityPersistence$^{\uparrow}$ *classifier identifies two accepted packets only when $M, R, K, L, G, A, S, F, H, C, P, N$ agree through displayed componentwise* hsame *transport. Equality of equation-like syntax is not enough; agreement of finite observed rows is not enough; and a future-persistence claim not attached to the explicit GAP row G, stability row S, failure row F, and law-certificate consumption row A is outside the accepted classifier.*

Theorem 1156.3 (Empirical regularity persistence GAP exposure)**.** *For every accepted* EmpiricalRegularityPersiste *carrier $E = (M, R, K, L, G, A, S, F, H, C, P, N)$, the future-facing part of the packet is exposed exactly by the Type IV GAP row G together with stability row S, failure row F, continuation replay C, provenance P, and local name N. The records-side rows M, R, K, L certify observed regularity only; they do not by themselves license future persistence.*

Lean4 ✓ BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceCarrier
gap exposure

Proof. Project the carrier of Definition 1156.1. The observed side is listed as M, R, K, L, while the persistence side is listed as G, S, F and is replayed through C, P, N.

Apply the classifier of Definition 1156.2. It refuses equality of equation-like syntax and finite observed rows when the GAP, stability, failure, replay, provenance, or local naming rows are missing.

Thus records-side regularity is a finite observation claim, while future persistence is the separate Type IV GAP exposure. Since the carrier has no hidden persistence coordinate, no accepted consumer can read forward commitment from M, R, K, L alone. □

Theorem 1156.4 (LawCertificate consumption of empirical regularity persistence)**.** *Every* LawCertificate$^{\uparrow}$ *consumer read of an accepted* EmpiricalRegularityPersistence$^{\uparrow}$ *packet factors through the measurement row M, regularity row R, classifier row K, error-ledger row L, future-persistence GAP row G, law-certificate consumption row A, stability row S, failure row F, transport row H, replay row C, provenance row P, and local* NameCert *row N. The consumer may not read an equation-like law packet when either the finite measurement and error rows or the explicit future-persistence GAP are absent.*

Lean4 ✓ `BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceCarrier lawcertificate consumption`

Proof. Start from the consumption row A. It names the $\mathsf{LawCertificate}^{\uparrow}$ surface that may read the empirical packet, but A is itself only one coordinate of $E = (M, R, K, L, G, A, S, F, H, C, P, N)$.

Use Definition 1156.3. The theorem separates observed regularity from future persistence and requires G, S, F, C, P, N for the forward commitment. The measurement row M, regularity row R, classifier row K, and error-ledger row L remain mandatory for the records-side evidence.

Therefore a law-certificate consumer sees a complete finite packet rather than a detached equation. If either the measurement and error rows or the future-persistence GAP row is absent, the requested law read lacks one side of the carrier and is rejected by the classifier. □

Theorem 1156.5 (Empirical regularity persistence nonescape)**.** *No accepted* $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ *consumer can recover physical-law status from a bare equation, observed records alone, model syntax alone, hidden calibration assumptions, an omitted error ledger, an unlisted continuation class, or a future-persistence assertion outside the displayed Type IV GAP row. Every accepted read remains inside* $M, R, K, L, G, A, S, F, H, C, P, N$.

Proof. Carrier projection gives the complete inventory $M, R, K, L, G, A, S, F, H, C, P, N$. The bare equation, model syntax, hidden calibration assumption, omitted error ledger, unlisted continuation class, and off-packet future-persistence assertion are not coordinates of that inventory.

Apply Definition 1156.2. The classifier compares accepted packets only through the displayed rows, so replacing any of those rows by an absent object changes the comparison surface rather than preserving the accepted packet.

Apply Definition 1156.4. Even the law-certificate consumer must pass through both sides of the packet: finite measurement and error evidence on the records side, and GAP, stability, and failure exposure on the future-facing side. Hence no accepted consumer escapes to physical-law status outside the displayed carrier. □

Lean4 ✓ `BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceCarrier nonescape`

The standard five-row NameCert surface is carrier admission for E, classifier exactness through K, stability and failure exposure through S, F, displayed route and provenance readback through H, C, P, N, and non-escape for bare equations, hidden measurement, omitted error ledgers, all-future truth, host physical necessity, and final-fit oracles.

The four AI rows are the finite measurement row M, the records-side regularity row R, the Type IV future-persistence GAP row G, and the law-certificate consumption row A. These rows make the packet a concrete physical-law precursor rather than an equation-only model artifact.

The Lean `TasteGate.lean` boundary for $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ instances preserving the displayed rows $M, R, K, L, G, A, S, F, H, C, P, N$.

Falsifiable prediction: An equation-like law packet lacking either finite measurement and error rows or an explicit future-persistence GAP is rejected by the EmpiricalRegularityPersistenceUp classifier.

Independence witness: lawcertificate, realityconstrainedphysicalinduction, openfitpacket, realityconstrainedmodelaudit, physicalinductionstabilityseal: none is bijective because $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ stores finite measurement, records-side regularity, classifier, error ledger, future-persistence GAP, LawCertificate consumption, stability, failure, transport, replay, provenance, and local NameCert

rows in one carrier. LawCertificateUp consumes a law surface but does not isolate records-side empirical regularity; RealityConstrainedPhysicalInductionUp binds induction continuation but not the measurement/error ledger as a physical-law precursor; OpenFitPacketUp records open fit without a law-consumption row; RealityConstrainedModelAuditUp audits model comparison without the Type IV persistence GAP; PhysicalInductionStabilitySealUp stores stability but not the empirical regularity and LawCertificate consumption pair.

Theorem 1156.6 (Empirical regularity persistence public interface)**.** *The public read exported by an accepted* EmpiricalRegularityPersistence$^{\uparrow}$ *packet is exactly the finite measurement row M, observed regularity row R, classifier row K, Type IV GAP row G,* LawCertificate$^{\uparrow}$ *consumption row A, stability row S, failure row F, componentwise transport row H, displayed replay row C, provenance row P, and local* NameCert *row N. A public consumer may not replace this interface by a bare equation, records-only regularity claim, model syntax, hidden calibration assumption, omitted error ledger, all-future truth, or law status detached from the displayed rows.*

Lean4 ✓ `BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceCarrier public interface`

Proof. Project the carrier of Definition 1156.1. The rows $M, R, K, L, G, A, S, F, H, C, P, N$ are the full packet inventory, with M, R, L supplying observed evidence, G, S, F supplying the future-facing GAP boundary, and A, H, C, P, N supplying consumption, transport, replay, provenance, and local naming.

Apply the classifier of Definition 1156.2. It identifies accepted packets only through the displayed rows, so a public read cannot drop the classifier, transport, replay, provenance, or local name while preserving the packet.

Use GAP exposure and LawCertificate consumption, Definition 1156.3 and Definition 1156.4. These results keep records-side regularity separate from future persistence and require the law-certificate consumer to pass through measurement, regularity, classifier, error ledger, GAP, consumption, stability, failure, transport, replay, provenance, and local naming rows.

Finally apply nonescape, Definition 1156.5. It excludes bare equations, records-only reads, model syntax, hidden calibration, omitted ledgers, unlisted continuations, and off-packet future-persistence assertions, leaving exactly the stated public interface. □

Theorem 1156.7 (Empirical regularity persistence bridge route)**.** *Every* LawCertificate$^{\uparrow}$*-facing bridge read of an accepted* EmpiricalRegularityPersistence$^{\uparrow}$ *packet factors through the displayed measurement row M, observed regularity row R, classifier row K, future-persistence GAP row G, stability row S, failure row F, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. The bridge may read the error-ledger row L as support for the same route, but it cannot replace the packet by a bare equation, records-only regularity claim, model syntax, hidden calibration assumption, all-future truth, or law status detached from those rows.*

Lean4 ✓ `BEDC.Derived.EmpiricalRegularityPersistenceUp.EmpiricalRegularityPersistenceBridgeRoute`

Proof. Start from Definition 1156.6. The public read exported by the packet is exactly the displayed measurement, regularity, classifier, GAP, law-certificate consumption, stability, failure, transport, replay, provenance, and local naming surface, with the error ledger retained as support.

Apply Definition 1156.4. A law-certificate consumer must pass through both the records side and the future-facing GAP side before reading any law-facing surface.

Finally use Definition 1156.5. The excluded replacements are not coordinates of the accepted carrier, so a bridge read that omits the listed rows leaves the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EmpiricalRegularityPersistence}\uparrow}$.

Constructive story: An $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ packet is generated from finite measurement rows, observed regularity rows, classifier exposure, error-ledger rows, a Type IV future-persistence GAP, LawCertificate consumption, stability, failure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmpiricalRegularityPersistence}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1156.1, Definition 1156.2, Definition 1156.3, Definition 1156.4, Definition 1156.5, Definition 1156.6, and Definition 1156.7 bind the finite observed-regularity and future-persistence packet exactly to the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EmpiricalRegularityPersistence}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, all-future truth, final physical necessity, host law identity, or law status without finite measurement, error, GAP, stability, failure, and LawCertificate rows.

Upgrade path: Further work is formal verification and bridge checking for carrier admission, classifier exposure, GAP exposure, LawCertificate consumption, non-escape, and TasteGate nontriviality with field faithfulness for $M, R, K, L, G, A, S, F, H, C, P, N$.

Chapter 1157

A Concrete Naming Certificate for $\mathsf{HostPrimitiveLeakage}^{\uparrow}$

The $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ packet realizes the host-smuggling branch of chapter 9462 as a finite diagnostic refusal carrier. It sits beside chapter 901, chapter 634, chapter 630, chapter 526, and chapter 546. Those siblings expose failed audit rows, quotient-boundary refusals, negative certificates, audit-gate verdicts, and kernel acceptance witnesses; this packet isolates the public-surface import of host primitives as the refused item.

Definition 1157.1 (HostPrimitiveLeakage carrier)**.** *A* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, E, D, F, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the public-surface site where a consumer requests host equality, host numerals, host lists, host function spaces, quotient soundness, choice, or propositional extensionality;*
2. *Q is the requested primitive row, naming the host feature that would otherwise enter as substrate content;*
3. *E is the exposed BEDC replacement row, containing the finite* BHist*,* msame*,* hsame*,* psame*,* Cont*,* Pkg*, or local* $\mathsf{NameCert}$ *route offered instead;*
4. *D is the diagnostic row explaining which public route attempted the import;*
5. *F is the failed-gate row consumed from* $\mathsf{AuditGateFailureWitness}^{\uparrow}$*;*
6. *G is the audit-boundary row consumed from* $\mathsf{AuditGateBoundary}^{\uparrow}$*;*
7. *H records componentwise* hsame *transport for the site, request, replacement, diagnostic, failed-gate, audit-boundary, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay for the refusal route and for the replacement route when one is available;*
9. *P is the* Pkg *provenance over* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$*,* $\mathsf{AuditGateFailureWitness}^{\uparrow}$*,* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$*,* $\mathsf{NegativeCertificateBoundary}^{\uparrow}$*,* $\mathsf{AuditGateBoundary}^{\uparrow}$*,* $\mathsf{KernelAuditWitness}^{\uparrow}$*,* BHist*,* msame*,* hsame*,* psame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HostPrimitiveLeakage}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host equality token, host-level numeric object, host list eliminator, host function-space extensionality principle, quotient soundness dependency, choice principle, or propositional-extensionality transport.

Lean4 (definition) `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageUp`

Theorem 1157.2 (HostPrimitiveLeakage NameCert obligations)**.** *For the carrier of Definition 1157.1, the local* $\mathsf{NameCert}_{\mathsf{HostPrimitiveLeakage}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$L = (S, Q, E, D, F, G, H, C, P, N),$$

public-site exposure through S*, requested-host-primitive exposure through* Q*, BEDC replacement exposure through* E*, diagnostic readback through* D*, failed-gate witness exposure through* F*, audit-boundary verdict exposure through* G*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageNameCertObligations`

Proof. Carrier admission is projection from Definition 1157.1. The rows $S, Q, E, D, F, G, H, C, P, N$ account for the public site, requested host primitive, BEDC replacement, diagnostic readback, failed-gate witness, audit boundary, transport, replay, provenance, and local naming surface.

The failed-gate and audit rows are consumed from the sibling certificates chapter 901 and chapter 526. The quotient-specific branch is read through chapter 634, and the named refusal branch is read through chapter 630.

The kernel witness sibling chapter 546 supplies the accepted-declaration readback surface, not a permission to import host primitives. Repacking these rows gives the local naming certificate, and every public consumer route is forced to display the requested host feature before the route can be refused or replaced. □

Theorem 1157.3 (HostPrimitiveLeakage non-escape)**.** *Every accepted* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ *packet* $L = (S, Q, E, D, F, G, H, C, P, N)$ *admits public reads only through the displayed diagnostic route* $S, Q, D, F, G, H, C, P, N$ *and, when present, the replacement route* E, H, C, P, N*. A consumer that reads host equality, host numerals, host lists, host function spaces, quotient soundness, choice, or propositional extensionality as BEDC-internal content has left the accepted packet.*

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageNonescape`

Proof. Inspect the carrier. The requested host feature is present only as the request row Q, and the diagnostic row D records the public route that tried to expose it.

Apply Definition 1157.2. The failed-gate row F and audit-boundary row G keep the request on the refusal side of the packet, while H, C, P, N preserve only the displayed diagnostic and replacement routes.

Finally use the excluded coordinates in Definition 1157.1. Since no host primitive is a packet coordinate, every accepted consumer route either reads it as a refused request through Q, D, F, G or uses the visible BEDC replacement row E. □

Falsifiable prediction: An accepted $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ packet is refuted by any public consumer route that obtains host equality, host numerals, host lists, host function spaces, quotient soundness, choice, or propositional extensionality without first exposing the requested primitive row and the failed-gate diagnostic row.

Independence witness: auditgatefailurewitness, quotientsoundnessboundary, negativecertificateboundary, auditgateboundary, kernelauditwitness: HostPrimitiveLeakage$^{\uparrow}$ has the simultaneous public-site row, requested-host-primitive row, replacement row, diagnostic row, failed-gate row, and audit-boundary row. The siblings each expose a gate, quotient, negative, audit, or kernel face, but none carries the complete finite leakage witness.

The Lean follow-up for this chapter must provide `Derived/HostPrimitiveLeakageUp/TasteGate.lean` with Nontrivial HostPrimitiveLeakage$^{\uparrow}$ and FieldFaithful HostPrimitiveLeakage$^{\uparrow}$ instances for the displayed carrier fields.

Theorem 1157.4 (HostPrimitiveLeakage public export)**.** *Every public* HostPrimitiveLeakage$^{\uparrow}$ *leakage read must expose the public site S, requested primitive Q, replacement row E, diagnostic row D, failed-gate row F, audit-boundary row G, transport row H, replay row C, provenance row P, local naming row N, and the displayed* `TasteGate` *marker before a consumer can use either the refusal route or the replacement route.*

Proof. Project the carrier of Definition 1157.1. The public leakage surface is exactly $S, Q, E, D, F, G, H, C, P, N$, with Q naming the requested host primitive and E naming only the BEDC replacement route when one is available.

Apply Definition 1157.2 and Definition 1157.3. The first theorem lists the local NameCert obligations for every displayed row, and the second prevents a public consumer from using a host primitive as accepted BEDC content.

The falsifiable prediction records the observable failure condition, while the independence witness separates this packet from the audit-gate and boundary siblings. Together with the displayed `BEDC.Derived.HostPrimitiveLeakageUp.taste_gate` marker, these rows expose the refusal or replacement route before any public consumer may use it. □

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakagePublicExport`

Theorem 1157.5 (HostPrimitiveLeakage report-row refusal)**.** *Let $L = (S, Q, E, D, F, G, H, C, P, N)$ be an accepted* HostPrimitiveLeakage$^{\uparrow}$ *packet. Any public report that identifies the requested primitive Q as host equality, host numerals, host lists, host function spaces, quotient soundness, choice, or propositional extensionality must pass through the diagnostic report row D, failed-gate row F, audit-boundary row G, transport row H, replay row C, provenance row P, and local naming row N before the refusal or replacement route is exported.*

Proof. Project the carrier of Definition 1157.1. The requested host feature is the row Q, and the only row that reports where the public import was attempted is D.

Apply Definition 1157.3. It keeps every accepted public read on the displayed diagnostic route $S, Q, D, F, G, H, C, P, N$ or on the replacement route E, H, C, P, N.

Finally use Definition 1157.4. The public export must display all ten rows before a consumer can use the refusal or replacement route, so a report that omits D, F, G, H, C, P, N cannot be an accepted HostPrimitiveLeakage$^{\uparrow}$ export. □

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageReportRowRefusal`

Theorem 1157.6 (HostPrimitiveLeakage quotient-soundness non-escape)**.** *Let $L = (S, Q, E, D, F, G, H, C, P, N)$ be an accepted* HostPrimitiveLeakage$^{\uparrow}$ *packet whose requested primitive row Q names quotient soundness. The packet can expose that request only as a refused host primitive routed through $S, Q, D, F, G, H, C, P, N$ and through the sibling quotient-boundary provenance in P; it does not export quotient soundness as BEDC-internal evidence.*

Proof. Use the carrier projection from Definition 1157.1. Quotient soundness can occur only as the requested primitive row Q, while the provenance row P records the quotient-boundary sibling route.

Apply Definition 1157.2. The local NameCert surface requires the request row, diagnostic row, failed-gate row, audit-boundary row, replay, provenance, and naming rows to be displayed together.

Now apply Definition 1157.3. Since quotient soundness is listed among the host primitives that cannot be accepted as BEDC-internal content, the only accepted read is the refusal route, with any BEDC replacement separated into E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageQuotSoundnessNonescape`

Theorem 1157.7 (HostPrimitiveLeakage audit-gate factorization)**.** *Every accepted public export of a* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ *packet* $L = (S, Q, E, D, F, G, H, C, P, N)$ *factors its leakage refusal through the failed-gate row* F *and audit-boundary row* G *before the structural rows* H, C, P, N *can name the exported route. Thus a consumer cannot bypass the audit gate by reading* Q *or* E *alone.*

Proof. Start with Definition 1157.4. It states that a public leakage read displays $S, Q, E, D, F, G, H, C, P, N$ before either the refusal route or replacement route is available.

The NameCert obligations of Definition 1157.2 identify F as the failed-gate witness row and G as the audit-boundary verdict row. These rows occur before the structural transport, replay, provenance, and local naming rows are consumed.

Finally apply Definition 1157.3. Reading Q alone would expose only a refused request, and reading E alone would expose only a replacement row; neither read can bypass the displayed audit-gate factorization through F, G, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageAuditGateFactorization`

Theorem 1157.8 (HostPrimitiveLeakage formal-axis boundary)**.** *The paper-mature refusal route for* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ *is assembled from the displayed carrier, NameCert obligations, non-escape theorem, public-export theorem, report-row refusal theorem, quotient-soundness non-escape theorem, audit-gate factorization theorem, the checked TasteGate row* `BEDC.Derived.HostPrimitiveLeakageUp.t` *and the checked non-escape row* `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageNonescape`*. What remains on the formal axis is a single combined Lean target,* `BEDC.Derived.HostPrimitiveLeakageUp.Host` *packaging those rows as one machine-checked chapter closure theorem.*

Proof. The carrier and paper theorems bind the public-site row, requested-host-primitive row, replacement row, diagnostic row, failed-gate row, audit-boundary row, transport row, replay row, provenance row, and local naming row inside the chapter.

The two checked Lean rows witness different formal components: the TasteGate marker records the refusal packet's taste surface, and HostPrimitiveLeakageNonescape records the checked non-escape boundary. Neither marker is stated as the single combined closure theorem named above.

Therefore the closurestatus block continues to report the formal axis as unformalized for the combined chapter theorem, while preserving the mature paper closure already witnessed by the displayed refusal route. □

Theorem 1157.9 (HostPrimitiveLeakage formal target)**.** *For an accepted* $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ *packet, the displayed carrier of Definition 1157.1, the NameCert obligations of Definition 1157.2,*

the non-escape route of Definition 1157.3, the public-export route of Definition 1157.4, the report-row refusal of Definition 1157.5, the quotient-soundness non-escape route of Definition 1157.6, the audit-gate factorization of Definition 1157.7, the displayed TasteGate row `BEDC.Derived.HostPrimitiveLeakageU` *and the checked non-escape row* `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageNonescape` *together form the formal target for future Lean packaging as* `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimi`

Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.HostPrimitiveLeakageFormalTarget`

Proof. The carrier projection supplies the ten displayed rows $S, Q, E, D, F, G, H, C, P, N$. Apply Definition 1157.2 to identify those rows as the complete local certificate surface.

The refusal side is fixed by Definition 1157.3, Definition 1157.5, and Definition 1157.6. These theorems keep host equality, host numerals, host lists, host function spaces, quotient soundness, choice, and propositional extensionality on the requested-primitive and diagnostic rows rather than as accepted BEDC content.

The export side is fixed by Definition 1157.4 and Definition 1157.7. They force every public route through the failed-gate and audit-boundary rows before transport, replay, provenance, and local naming can export either the refusal route or the replacement route.

Finally the displayed TasteGate row and checked non-escape row are exactly the Lean-visible rows already named in the chapter. Packaging these paper and Lean-visible components as one theorem is the combined formal target, with no additional marker asserted here. □

Closure status for $\mathsf{NameCert}_{\mathsf{HostPrimitiveLeakage}^{\uparrow}}$.
Lean4 ✓ `BEDC.Derived.HostPrimitiveLeakageUp.taste gate`

Constructive story: A $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ packet is generated from a public-surface site, requested host primitive, BEDC replacement row, diagnostic readback, failed-gate witness, audit-boundary row, componentwise transports, continuation replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HostPrimitiveLeakage}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1157.1, Definition 1157.2, Definition 1157.3, Definition 1157.4, the displayed falsifiable prediction, the displayed independence witness, the audit-boundary sibling chapters, and the displayed `BEDC.Derived.HostPrimitiveLeakageUp.` marker bind the refusal packet through BHist, msame, hsame, psame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HostPrimitiveLeakage}^{\uparrow}}, \mathsf{unformalized})$

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.hostPrimitiveLeakageDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, or every host-leakage case.

Upgrade path: Machine bridge checking requires a checked standard interface row that preserves the refusal packet across standard BEDC interfaces.

Chapter 1158

A Concrete Naming Certificate for $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$

The $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ packet realizes the diagonal-regularity row of Definition 9238.4 as a finite concrete NameCert carrier. It sits downstream of the selected diagonal representative in chapter 374, chapter 1205, and the three-leg estimate surface of chapter 466, while consuming the window, regular-rational, dyadic, and real-seal sibling rows of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the finite regularity evidence used by downstream Real-completion routes; it is not a selected limit, host real equality, quotient-stream equality, countable choice, or ambient completeness principle.

Definition 1158.1 (Regular Cauchy regularity-witness carrier)**.** *A* $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the selected source row naming the diagonal representative route consumed from chapter 1205.*
2. *μ is the modulus index row that selects the finite regular-Cauchy comparison budget for the requested pair of precisions.*
3. *j is the finite index arithmetic row tying the selected source row to the regularity request.*
4. *Ω is the* $\mathsf{StreamName}^{\uparrow}$ *window exposing the diagonal observation and the intermediate observations used by the regularity consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected window and its regular-rational endpoint comparisons.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the two endpoint tolerances, the middle comparison tolerance, and the final diagonal-regularity bound.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached only after the displayed window, readback, and tolerance ledger have been read.*
8. *H records componentwise* hsame *transport for $S, \mu, j, \Omega, R, Q, E$ and the local certificate row.*
9. *C records displayed* Cont *replay from selected source row through modulus index, window, readback, dyadic ledger, and Real seal handoff.*

10. *P is the* Pkg *provenance over* RegularCauchyRegularityWitness$^{\uparrow}$, DiagonalLimit$^{\uparrow}$, RegularCauchyTriangleBound$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyRegularityWitness}^{\uparrow}}$ *row.*

The carrier exports no selected host limit, host real equality, quotient-stream equality, Quot.sound *row,* Classical.choice *row, or ambient* Real$^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.TasteGate.RegularCauchyRegularityWitnessTasteGate single carrier alignment`

Theorem 1158.2 (Regular Cauchy regularity-witness NameCert obligations)**.** *For the carrier of Definition 1158.1, the local* NameCert$_{\text{RegularCauchyRegularityWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for $W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$, selected diagonal source exposure through S, modulus-index computation through μ and j, finite* StreamName$^{\uparrow}$ *window exposure through Ω,* RegSeqRat$^{\uparrow}$ *regularity readback through R,* DyadicRatCore$^{\uparrow}$ *tolerance-ledger exposure through Q, downstream* Real$^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.TasteGate.RegularCauchyRegularityWitnessNameCert(`

Proof. Project the carrier. Its coordinates list the selected diagonal source row, modulus index, index arithmetic, finite stream window, regular-rational readback, dyadic tolerance ledger, Real seal handoff, transport, replay, provenance, and local name.

The sibling surfaces account for the data-bearing rows. The diagonal representative supplies S; the triangle-bound packet supplies the regularity estimate route; StreamName$^{\uparrow}$ supplies Ω; RegSeqRat$^{\uparrow}$ supplies R; DyadicRatCore$^{\uparrow}$ supplies Q; and Real$^{\uparrow}$ supplies E only as the terminal seal handoff.

The remaining rows do not add mathematical content. They transport, replay, source, and name the displayed rows, so the listed clauses exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness namecert obligations`

Theorem 1158.3 (Regular Cauchy regularity-witness diagonal-consumer boundary)**.** *Every diagonal-regularity consumer of an accepted* RegularCauchyRegularityWitness$^{\uparrow}$ *packet factors through the finite route*

$$S, \mu, j, \Omega, R, Q, E, H, C, P, N.$$

The consumer reads the selected source row, modulus index, finite StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger before it can receive the* Real$^{\uparrow}$ *seal handoff.*

Proof. Begin with the NameCert obligations. They require the selected diagonal source row and the modulus-index rows to be visible before any regularity evidence is read.

Next read the sibling surfaces in order. The stream window exposes the finite observations, the regular-rational row supplies the endpoint and middle readbacks, and the dyadic ledger records the tolerance arithmetic consumed by the diagonal-regularity theorem.

Finally pass through the Real seal handoff and the structural rows. Since the carrier contains no off-route equality or selection coordinate, every diagonal-regularity consumer remains inside the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitnessDiagonalConsumerBou`

Theorem 1158.4 (Regular Cauchy regularity-witness no-hidden-limit boundary)**.** *An accepted* RegularCauchyRegularityWitness$^{\uparrow}$ *packet exports exactly the selected source row S, modulus-index rows μ, j, finite* StreamName$^{\uparrow}$ *window Ω,* RegSeqRat$^{\uparrow}$ *readback R,* DyadicRatCore$^{\uparrow}$ *tolerance ledger Q,* Real$^{\uparrow}$ *seal handoff E, componentwise transport H, continuation replay C, package provenance P, and local naming row N. It exports no selected host limit, host real equality, quotient-stream equality,* Quot.sound *witness,* Classical.choice *witness, or ambient completeness row.*

Proof. Inspect the carrier coordinates of Definition 1158.1. Every coordinate is a displayed BEDC row, a finite modulus or index row, or structural support for transport, replay, provenance, and naming.

Definition 1158.2 accounts for the admissible carrier and classifier-facing rows. Definition 1158.3 accounts for every diagonal-regularity consumer read.

The excluded objects require additional coordinates absent from the carrier. Hence they cannot be exported by the local naming certificate or recovered by a consumer of the displayed regularity-witness route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness no hidden limit boundary`

Theorem 1158.5 (Regular Cauchy regularity-witness Real-completion handoff)**.** *Every accepted* RegularCauchyRegularityWitness$^{\uparrow}$ *packet*

$$W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$$

consumed by a RealCauchyCompletion$^{\uparrow}$ *diagonal route is read through the selected diagonal source S, the finite* StreamName$^{\uparrow}$ *window Ω, the* RegSeqRat$^{\uparrow}$ *readback R, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger Q, and the* Real$^{\uparrow}$ *seal handoff E before any completion-facing seal is available. The same route displays the supporting* BHist, hsame, Cont, Pkg, *and* NameCert *rows H, C, P, N carried by the packet.*

Consequently the completion-facing read factors through the finite regularity-witness surface and the finite-window Real-completion support route. It exports no selected host limit, host real equality, quotient-stream equality, Quot.sound *row,* Classical.choice *row, ambient completeness coordinate, or substitute support row outside the displayed BEDC packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness real completion handoff`

Proof. Begin with the local packet. Definition 1158.2 exposes exactly the selected source, modulus-index, window, readback, tolerance-ledger, Real-seal handoff, transport, replay, provenance, and naming rows of W.

The diagonal-consumer boundary Definition 1158.3 orders these rows before any consumer receives the Real$^{\uparrow}$ seal handoff. The no-hidden-limit boundary Definition 1158.4 removes selected host limits, host real equality, quotient-stream equality, Quot.sound, Classical.choice, and ambient completeness from the packet surface.

Now attach the completion-facing route. Definition 360.48 exhausts every accepted RealCauchyCompletion$^{\uparrow}$ finite-window completion read by the displayed StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, DyadicRatCore$^{\uparrow}$ ledgers, diagonal packet, Real$^{\uparrow}$ seal, BHist, hsame, Cont, Pkg, and NameCert rows. Definition 360.52 says that the Phase Real and Completion exit remains valid only when DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and the displayed structural support rows are present.

Intersecting these two finite routes leaves exactly the handoff stated above. The regularity witness supplies the selected diagonal source, window, readback, dyadic ledger, Real seal, and support rows consumed by the Real-completion route, and the cited non-escape clauses leave no coordinate through which the completion-facing seal could bypass them. □

Theorem 1158.6 (Regular Cauchy regularity witness tail-window regularity)**.** *Every accepted* $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *packet consumed as a regularity witness exposes its* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{Real}^{\uparrow}$ *sealing row, and* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *handoff before the witness can be consumed. Equivalently, a consumer of the packet must factor through the displayed tail-window route*

$$S, \mu, j, \Omega, R, Q, E, H, C, P, N$$

with Ω *read as the finite observation window,* R *as the regular-rational regularity readback,* Q *as the dyadic tolerance ledger,* E *as the Real sealing row, and* H, C, P, N *as the componentwise transport, continuation replay, provenance, and naming rows that carry the handoff into the completion-facing route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitnessTailWindowRegularit`
↪ variant `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness tail window regularity`

Proof. Project the accepted carrier. The coordinates of Definition 1158.1 display the selected source row, modulus-index rows, finite observation window, regular-rational readback, dyadic tolerance ledger, Real seal, and structural support rows.

The consumer order is forced by the preceding boundaries. Definition 1158.3 places the window, readback, and tolerance ledger before the Real seal; Definition 1158.5 then routes the same displayed rows into the $\mathsf{RealCauchyCompletion}^{\uparrow}$ diagonal surface.

It remains to check that transport cannot alter the exposed rows. Replay the componentwise hsame row H and the displayed Cont row C along the packet coordinates; the Pkg row P and local $\mathsf{NameCert}$ row N only source and name the same finite data. Thus any accepted consumer sees the tail-window route before it can consume the regularity witness. □

Theorem 1158.7 (Regular Cauchy regularity-witness obligation closure)**.** *For the carrier* $W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$ *of Definition 1158.1, carrier admission, diagonal source exposure, modulus-index rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularityWitness}^{\uparrow}}$ *row form the full obligation surface. Any accepted regularity-witness consumer factors through that finite surface before it can use the diagonal-regularity witness, and no selected host limit, host real equality, quotient-stream equality,* $\mathsf{Quot.sound}$ *row,* $\mathsf{Classical.choice}$ *row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate is part of the obligation closure.*

Proof. Project the accepted packet W. The carrier definition displays exactly the source, modulus, index, finite window, readback, tolerance, Real seal, transport, replay, provenance, and naming rows, and Definition 1158.2 identifies those rows as the local NameCert obligations.

Induct over the displayed finite window route. The base read exposes the selected diagonal source S and the first window row in Ω through the modulus-index data μ, j. The step extends the same route by one displayed observation, transports the previous rows through H, and replays the continuation row C without adding a new carrier coordinate.

The consumer boundaries close the remaining cases. Definition 1158.3 forces any diagonal-regularity consumer through the listed rows; Definition 1158.4 removes off-route limit and equality data; and Definition 1158.5 places the $\mathsf{Real}^{\uparrow}$ seal after the window, readback, and dyadic ledger.

Thus the finite route named in Definition 1158.6 is not merely a handoff order. It is the whole obligation surface for the accepted packet, with P and N recording provenance and local naming for the same rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitnessCarrier obligation closure`

Theorem 1158.8 (Regular Cauchy regularity-witness scoped kernel route)**.** *The accepted* RegularCauchyRegularityW *carrier factors only through the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal handoff,* BHist *packet rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before a diagonal-regularity consumer can read it. Thus the scoped kernel route for the carrier*

$$W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$$

is exhausted by the displayed rows

$$S, \mu, j, \Omega, R, Q, E, H, C, P, N$$

and cannot pass through selected host limits, host real equality, quotient-stream equality, Quot.sound*,* Classical.choice*, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness scoped kernel route`

Proof. Project the accepted carrier. The source, modulus, and index rows determine which finite window route is being read, while Ω is the only $\mathsf{StreamName}^{\uparrow}$ window coordinate available to the consumer.

The readback and estimate rows then fix the numerical surface. The $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies the regular-rational readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row Q records the tolerance ledger used by the diagonal-regularity consumer before any Real seal is visible.

The terminal and structural rows close the route. The $\mathsf{Real}^{\uparrow}$ handoff E is read only after the window, readback, and tolerance rows; H, C, P, N transport, replay, source, and name the same packet. By the no-hidden-limit boundary, there is no carrier coordinate for off-route equality, choice, quotient soundness, or ambient completeness, so the displayed finite route is the scoped kernel route. □

Theorem 1158.9 (Regular Cauchy regularity-witness finite-window induction)**.** *Every accepted* $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *packet*

$$W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N)$$

can be consumed by induction over the finite $\mathsf{StreamName}^{\uparrow}$ *windows displayed in* Ω *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal handoff* E *are exposed. The induction base reads the selected diagonal source* S *and the first window row named by* μ, j*; each induction step appends one displayed window row, transports the prior window data through* H*, and replays the finite continuation route in* C*. Only after this finite induction has displayed the requested window may a consumer read* R*, then* Q*, and finally* E*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness finite window induction`

Proof. Project the accepted carrier of Definition 1158.1. The only window data available before readback is the finite $\mathsf{StreamName}^{\uparrow}$ coordinate Ω, with the selected source row S and modulus-index rows μ, j determining which finite observations are requested.

Induct over the displayed window spine. In the base case, the consumer reads S and the first named observation through Ω; no regular-rational readback, dyadic ledger, or Real seal coordinate is yet available.

For the step case, append one displayed observation row of Ω. The componentwise transport row H preserves the prior selected source, modulus, index, and window rows, while the continuation row C replays the finite route to the extended window without adding an off-carrier equality or choice row.

After the finite window has been displayed, Definition 1158.3 and Definition 1158.5 force the remaining order: the consumer reads the $\mathsf{RegSeqRat}^{\uparrow}$ row R, then the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger Q, and only then the $\mathsf{Real}^{\uparrow}$ seal handoff E. Thus the accepted witness is consumable by finite-window induction before any downstream readback, tolerance ledger, or Real seal is exposed. □

Theorem 1158.10 (Regular Cauchy regularity-witness tail-window synchronization)**.** *Let*

$$W = (S, \mu, j, \Omega, R, Q, E, H, C, P, N) \quad \text{and} \quad W' = (S', \mu', j', \Omega', R', Q', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *packets. If their inspected finite tails agree through the displayed* hsame *transports and* Cont *rows on* S, μ, j, Ω *and* S', μ', j', Ω'*, and if their* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows are read only after that shared finite-window route, then the two packets expose the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface and the same Real-completion handoff surface to every completion-facing consumer.*

The synchronization is a finite tail-window fact. It does not select a limit, identify host real values, prove quotient equality of regular sequences, or use ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularityWitnessUp.RegularCauchyRegularityWitness tail window synchronization`

Proof. Project the two accepted carriers using Definition 1158.1. The only rows visible before readback are the selected source, modulus index, index arithmetic, and finite $\mathsf{StreamName}^{\uparrow}$ window rows.

Apply finite-window induction from Definition 1158.9 to both packets. The hypothesis that the inspected finite tails agree under the displayed hsame and Cont rows makes the induction expose the same finite window route before either packet can read R, Q, E or R', Q', E'.

Now use the diagonal-consumer boundary Definition 1158.3. It forces every diagonal-regularity consumer to read the $\mathsf{RegSeqRat}^{\uparrow}$ row only after the shared window route has been displayed. Hence R and R' present the same readback surface to such a consumer.

Finally apply the Real-completion handoff Definition 1158.5. The handoff can inspect only the window, readback, dyadic ledger, Real seal, and structural rows already synchronized above. The no-hidden-limit boundary excludes selected limits, quotient equality, host real equality, and ambient completeness, so no off-route coordinate can distinguish the two completion-facing surfaces. □

Falsifiable prediction: Every accepted RegularCauchyRegularityWitnessUp packet exposes a finite dyadic regularity-witness row and admits no route to a selected limit, host real equality, Quot.sound, or Classical.choice.

Independence witness: diagonallimit, regular-cauchy-triangle-bound, regseqrat, streamname, real: no listed sibling is bijective to $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ because this packet binds selected diagonal source, modulus index, finite window, regular-rational readback, dyadic tolerance ledger, Real seal handoff, transport, replay, provenance, and local NameCert rows in one consumer-facing regularity witness. The diagonal-limit sibling lacks the regularity-consumer tolerance ledger; the triangle-bound sibling lacks the selected representative seal handoff; RegSeqRatUp lacks the StreamName and Real route; StreamNameUp lacks regularity and dyadic ledger rows; RealUp lacks the finite witness route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularityWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ packet is generated from a selected diagonal source row, modulus index, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularityWitness}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1158.1, Definition 1158.2, Definition 1158.3, Definition 1158.4, Definition 1158.7, and Definition 1158.8 bind the diagonal-regularity witness packet as a finite Real-completion handoff whose consumer-facing route is scoped by the displayed StreamName, RegSeqRat, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularityWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected host limits, host real equality, quotient-stream equality, Quot.sound, Classical.choice, or ambient Real completeness.

Upgrade path: Lean verification requires a checked RegularCauchyRegularityWitnessUp theorem for the scoped consumer-facing route and its sibling dependency boundary.

Chapter 1159

A Concrete Naming Certificate for $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$

The $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ packet isolates the last estimate in the compact-uniform-continuity construction of chapter 9237. It is the Bishop–Bridges finite-net proof surface for Heine–Cantor in BEDC form: a selected compact-net center is used twice in the source metric, the continuous-map rows provide two pointwise target estimates at that center, and the target metric triangle row returns the final uniform estimate. The packet is anchored in the metric certificate of chapter 57, compact metric data of chapter 129, continuous-map data of chapter 130, finite lower-bound folding of chapter 619, and the consumer surface of chapter 369.

Definition 1159.1 (Metric-triangle uniform-estimate carrier)**.** *A* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier is a finite packet*

$$E = (X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *row carrying the located source-distance classifier.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row carrying the target-distance classifier.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus packet for the displayed function from X to Y.*
4. *x and x' are the two source histories whose target images are compared by the uniform-modulus consumer.*
5. *c is the compact-net center selected from the finite bundle associated to the tolerance row.*
6. *d_x is the source distance row from x to c supplied by compact-net coverage.*
7. *$d_{x'}$ is the source distance row from x' to c obtained from the selected precision row, the source distance between x and x', and the source triangle ledger.*
8. *e_c is the selected precision row exported by the finite fold of chapter 619.*
9. *e_x and $e_{x'}$ are the two pointwise target-distance estimates at c, one for $f(x)$ and one for $f(x')$.*
10. *T is the final target triangle ledger joining the two pointwise target estimates into the displayed estimate between $f(x)$ and $f(x')$.*

11. *H stores the componentwise* hsame *transports for all source, target, precision, and triangle rows.*
12. *C stores the* Cont *routes by which a consumer reads the source triangle, the target triangle, and the uniform-modulus handoff.*
13. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, *and* Cont *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricTriangleUniformEstimate}^{\uparrow}}$ *registration row.*

The classifier compares carriers only through these displayed rows. It has no coordinate for host compactness, selected subsequences, choice principles, open-cover compactness, or completeness of the real numbers.

Lean4 (definition) `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier`

Theorem 1159.2 (Metric-triangle uniform-estimate NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MetricTriangleUniformE}}$ *obligation surface consists of five finite rows:*

1. *carrier exposure, requiring every public read of E to factor through the displayed source metric, target metric, continuous graph, selected center, distance, precision, triangle, transport, continuation, provenance, and local naming rows;*
2. *classifier stability, requiring componentwise* hsame *transport for $X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N$;*
3. *source triangle admissibility, requiring $d_{x'}$ to be read from the source metric triangle row using the selected precision e_c, the compact-net coverage row d_x, and the displayed source distance between x and x';*
4. *target triangle admissibility, requiring T to be read from the target metric triangle row using the two pointwise target-distance estimates e_x and $e_{x'}$ at the same center c;*
5. $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff, requiring the downstream uniform-modulus packet to receive only the finite selected precision row, the two source metric bounds, the two pointwise target bounds, the final target triangle row, and the visible transport and provenance rows.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier namecert obligations`

Proof. Project the carrier $E = (X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N)$. Carrier exposure and classifier stability are the displayed coordinates and the componentwise transports in H.

For source admissibility, read the finite-fold precision row e_c and the compact-net coverage row d_x. The source metric triangle row in X combines those rows with the displayed distance between x and x' and produces the source bound $d_{x'}$ at the same center c.

For target admissibility, use the pointwise rows in F at the chosen center. They give e_x and $e_{x'}$ as target estimates from $f(c)$ to $f(x)$ and from $f(c)$ to $f(x')$. The target metric triangle row in Y then supplies T.

The consumer handoff is the continuation route C with provenance P and local name N. Since the packet has no coordinate for host compactness, selected subsequences, choice, open-cover compactness, or real completeness, none of those objects can be exported to $\mathsf{UniformModulus}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier uniformmodulus boundary exactness`

Theorem 1159.3 (Metric-triangle uniform-estimate consumer handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier receives exactly the selected precision row* e_c*, the two source metric bounds* d_x *and* $d_{x'}$*, the two pointwise target bounds* e_x *and* $e_{x'}$*, the final target triangle row* T*, and the displayed transport, continuation, provenance, and local naming rows* H, C, P, N*. It receives no host compactness coordinate, selected subsequence, choice-selected point outside the finite bundle, open-cover compactness object, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier consumer handoff`

Proof. Start with the carrier exposure and source triangle admissibility of Definition 1159.2. They identify the only source-side data delivered to a consumer: the selected precision row, compact-net coverage, the source distance between x and x', and the source triangle row producing $d_{x'}$.

Next use target triangle admissibility. The pointwise target estimates e_x and $e_{x'}$ are read from the continuous-map packet at the same selected center, and the final estimate is the target triangle row T.

Finally apply the handoff obligation. The continuation and provenance rows send precisely those finite coordinates to $\mathsf{UniformModulus}^{\uparrow}$, while the excluded host compactness, subsequence, choice, open-cover, and real-completeness objects are absent from the carrier. □

Theorem 1159.4 (Metric-triangle uniform-estimate source-triangle coherence)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier* E*, the selected center route and the source metric triangle row determine the two source-distance rows* d_x *and* $d_{x'}$ *used by the uniform-estimate packet. The determination uses the carrier rows, the local NameCert obligations, compact-net coverage from chapter 129, finite lower-bound precision from chapter 619, and the source* $\mathsf{Met}^{\uparrow}$ *triangle ledger; it uses no host compactness, selected subsequence, open-cover compactness, choice principle, or real completeness.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier source triangle coherence`

Proof. Project the carrier of Definition 1159.1. The displayed rows include the selected center c, the selected precision e_c, compact-net coverage for x at c, the source distance between x and x', and the source triangle ledger inside X.

Apply the NameCert obligations of Definition 1159.2. Source triangle admissibility says that d_x is the compact-net coverage row and that $d_{x'}$ is read from the source triangle row using e_c, d_x, and the displayed distance between x and x'.

The consumer handoff theorem Definition 1159.3 then fixes these rows as the only source bounds sent onward. Since the carrier has no host compactness or choice coordinate, no outside compactness principle can alter the determined source route. □

Theorem 1159.5 (Metric-triangle uniform-estimate target-triangle coherence)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier* E*, the two pointwise target estimates* e_x *and* $e_{x'}$ *at the same selected center* c *feed the target triangle row* T *and determine the displayed target estimate between* $f(x)$ *and* $f(x')$*. The route factors through the target* $\mathsf{Met}^{\uparrow}$ *row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise rows, the carrier, the local NameCert obligations, and the consumer handoff, with no semantic target-space completeness or host equality of target points.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier target triangle coherence`

Proof. Read the carrier rows $Y, F, c, e_x, e_{x'}, T$ from Definition 1159.1. The target metric row Y supplies the triangle ledger, while the continuous-map packet F supplies both pointwise target estimates at the same selected center.

Use target triangle admissibility from Definition 1159.2. It identifies T as the row obtained from those two pointwise estimates through the target metric triangle.

Apply Definition 1159.3. The downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer receives T only through this finite target route, so the estimate is not supplemented by target completeness, host equality, or a hidden semantic metric argument. □

Theorem 1159.6 (Metric-triangle uniform-estimate uniform-modulus route exhaustion)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier factors through the selected precision* e_c*, the two source bounds* $d_x, d_{x'}$*, the two target bounds* $e_x, e_{x'}$*, the final target triangle row* T*, and the visible transport, continuation, provenance, and local naming rows* H, C, P, N*. No consumer route may add host compactness, a choice-selected center outside the finite bundle, an open-cover compactness witness, a semantic continuity theorem, or a* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier uniform modulus route exhaustion`

Proof. First combine source-triangle coherence with target-triangle coherence. The source side fixes d_x and $d_{x'}$ from the selected center route and source triangle ledger, while the target side fixes T from the two pointwise target estimates at that same center.

Next apply the NameCert obligations of Definition 1159.2. They make carrier exposure, classifier stability, source admissibility, target admissibility, and consumer handoff the full obligation surface for the packet.

Finally use Definition 1159.3. It states that the consumer receives exactly $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$, so every accepted $\mathsf{UniformModulus}^{\uparrow}$ read factors through those rows and through no host compactness or completion object. □

Theorem 1159.7 (Metric-triangle uniform-estimate classifier transport)**.** *For accepted* MetricTriangleUniformEstim *carriers* E *and* E'*, componentwise* hsame *transport of*

$$X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N$$

preserves the public classifier. The transported route remains internal to BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it does not pass through host compactness, selected subsequences, open-cover compactness, choice, semantic continuity, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier classifier transport`

Proof. Project both carriers through Definition 1159.1. The classifier reads only the displayed rows $X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N$.

Apply classifier stability from Definition 1159.2. Each displayed coordinate is transported by the matching component of H, so the two public reads agree component by component.

Use source-triangle coherence, target-triangle coherence, and route exhaustion. These results keep the source bound, target triangle, and consumer handoff inside the displayed BHist, hsame, Cont, Pkg, and NameCert route, so no external compactness or completion coordinate can affect the classifier. □

Theorem 1159.8 (Metric-triangle uniform-estimate shared-center triangle composition)**.** *For an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier E, the shared center c composes compact-net coverage, source triangle admissibility, the two* ContinuousMap$^{\uparrow}$ *pointwise estimates, and the target* Met$^{\uparrow}$ *triangle row into the displayed estimate between $f(x)$ and $f(x')$. The composite route factors through the carrier rows and the local NameCert obligations.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier shared center triangle composition`

Proof. Begin with compact-net coverage at the selected center. The carrier exposes c, d_x, e_c, and the source metric row X, while source-triangle coherence identifies $d_{x'}$ as the source-triangle output at the same center.

Next read the continuous-map packet F. Its two pointwise rows give e_x and $e_{x'}$ at the shared center c.

Apply target-triangle coherence. The target metric row Y uses those two pointwise estimates to produce T, the displayed target estimate between $f(x)$ and $f(x')$.

Finally apply the NameCert obligations and route exhaustion. They repack the compact-net coverage, source triangle, pointwise estimates, and target triangle as one internal consumer route. □

Theorem 1159.9 (Metric-triangle uniform-estimate UniformModulus boundary exactness)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier receives exactly the finite boundary rows*

$$e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$$

and no host compactness row, selected subsequence row, open-cover compactness row, choice row, or Real$^{\uparrow}$ *completeness row. This boundary is exact for the consumer interface of the packet.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateUniformModulusBoundar`

Proof. Project the carrier and apply the NameCert obligations. Carrier exposure lists the finite rows, and the consumer handoff obligation restricts the public UniformModulus$^{\uparrow}$ read to the displayed boundary.

Use source-triangle coherence and shared-center triangle composition. They show that $d_x, d_{x'}$ and $e_x, e_{x'}, T$ are not auxiliary imports; they are the source and target sides of the same selected-center route.

Finally apply route exhaustion. It identifies $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$ as the whole downstream surface, so the absent compactness, subsequence, open-cover, choice, and real-completeness objects cannot be received by any accepted consumer. □

Theorem 1159.10 (Metric-triangle uniform-estimate radius composition)**.** *For an accepted* MetricTriangleUniformE *carrier E, the selected precision row e_c, compact-net coverage, and the source metric triangle row determine the source-distance row $d_{x'}$ at the same selected center c. The determination factors through the carrier of Definition 1159.1, the NameCert obligations of Definition 1159.2, source-triangle coherence of Definition 1159.4, target-triangle coherence of Definition 1159.5, and the consumer handoff of Definition 1159.3.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier radius composition`

Proof. Project the carrier. The rows $c, d_x, d_{x'}, e_c$ and the source metric row X are displayed coordinates of E.

Apply the NameCert obligations. Source triangle admissibility reads e_c and the compact-net coverage row d_x before using the source metric triangle row to obtain $d_{x'}$.

Use source-triangle coherence to identify the center. The theorem fixes both source-distance rows on the selected center c, so $d_{x'}$ cannot be recomputed at another center.

The target-triangle coherence and consumer handoff keep the resulting row in the same downstream route. The UniformModulus$^{\uparrow}$ consumer receives $d_{x'}$ only as this displayed source-triangle output. □

Theorem 1159.11 (Metric-triangle uniform-estimate center reuse determinacy)**.** *For an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier E, the same selected center c is used by both pointwise target estimates $e_x, e_{x'}$ and by the final target triangle row T. No accepted consumer route may replace that center between the two pointwise estimates or between those estimates and the target triangle row.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier center reuse determinacy`

Proof. Read $c, e_x, e_{x'}, T$ from the carrier of Definition 1159.1. The pointwise estimates and the final triangle row are displayed in one packet.

Apply target triangle admissibility from Definition 1159.2. It says that T is read from the target metric triangle row using the two pointwise target estimates at the same selected center c.

Use Definition 1159.5. The theorem identifies the target estimate between $f(x)$ and $f(x')$ as the route through e_x, $e_{x'}$, and T at that center.

Finally apply Definition 1159.3. The handoff exports no second center, so the same c is determined for both pointwise estimates and for the final target triangle row. □

Theorem 1159.12 (Metric-triangle uniform-estimate no completion escape)**.** *A* UniformModulus$^{\uparrow}$ *consumer of an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier cannot add real completeness, open-cover compactness, choice, or a second selected center. Its read is exhausted by the displayed source bounds, pointwise target estimates, final target triangle row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier no completion escape`

Proof. Start with the carrier and NameCert obligations. They expose the finite rows $X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}$ and no outside analytic coordinate.

Apply source-triangle coherence and radius composition. These results determine d_x and $d_{x'}$ from the selected center route, compact-net coverage, selected precision, and source triangle row.

Apply target-triangle coherence and center reuse determinacy. The target route uses the same c for both pointwise estimates and for the final target triangle row T.

The consumer handoff and route exhaustion then list the whole downstream surface. Since real completeness, open-cover compactness, choice, and a second center are absent from that list, a UniformModulus$^{\uparrow}$ consumer cannot obtain them from the accepted packet. □

Theorem 1159.13 (Metric-triangle uniform-estimate shared-modulus domain)**.** *For an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier E, the selected compact-net center c, source metric triangle row, two* ContinuousMap$^{\uparrow}$ *pointwise rows, target metric triangle row, and* UniformModulus$^{\uparrow}$

consumer handoff share one displayed modulus domain. The carrier does not permit the source triangle, the two target estimates, or the final handoff to choose separate centers, separate tolerance rows, or separate consumer domains.

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier shared modulus domain`

Proof. Project the carrier of Definition 1159.1. The rows $c, d_x, d_{x'}, e_c, e_x, e_{x'}, T$ occur in one packet, so the selected center and selected precision are visible before any consumer handoff.

Apply the NameCert obligations of Definition 1159.2. Source triangle admissibility uses e_c, d_x, and the displayed source distance to produce $d_{x'}$, while target triangle admissibility uses the two pointwise target estimates at the same center to produce T.

Finally use Definition 1159.3. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives only those rows and the structural rows H, C, P, N, so no second center, tolerance row, or consumer domain is available. □

Theorem 1159.14 (Metric-triangle uniform-estimate consumer coverage)**.** *Every accepted* $\mathsf{UniformModulus}^{\uparrow}$ *read of a* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier is covered by the compact-net center row, the source triangle row, the two continuous-map pointwise rows, the target triangle row, and the handoff rows H, C, P, N. A consumer that asks for a compactness theorem, a semantic continuity theorem, a real-completeness row, or a selected center outside the carrier is outside the covered surface.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier consumer coverage`

Proof. Begin with Definition 1159.13. It places the source and target estimates over the same selected center and selected precision.

Use route exhaustion from Definition 1159.6. The consumer route is exhausted by $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$

Apply no-completion escape from Definition 1159.12. The excluded compactness theorem, semantic continuity theorem, real-completeness row, and outside center are not carrier coordinates and cannot be supplied by the handoff. □

Theorem 1159.15 (Metric-triangle uniform-estimate source-target lock)**.** *For an accepted* MetricTriangleUniformEs *carrier, the source metric triangle row and the target metric triangle row are locked to the same selected compact-net center and the same* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. The source-side production of $d_{x'}$ cannot be consumed without the target-side production of T, and the target-side production of T cannot be consumed without the displayed source bounds $d_x, d_{x'}$.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier source target lock`

Proof. Read the source side first. Source-triangle coherence and radius composition identify d_x and $d_{x'}$ as the source bounds produced from the selected center, selected precision, and source triangle row.

Read the target side next. Target-triangle coherence and center reuse determinacy identify T as the target triangle row using the two pointwise estimates at the same selected center.

The lock is the consumer boundary. By Definition 1159.3 and Definition 1159.9, the downstream $\mathsf{UniformModulus}^{\uparrow}$ read receives source rows and target rows together as one finite boundary, with no separate source-only or target-only export route. □

Theorem 1159.16 (Metric-triangle uniform-estimate two-triangle route)**.** *For an accepted* MetricTriangleUniformEs *carrier, the source metric triangle row and the target metric triangle row form one selected-center route: the compact-net center c supplies the source bounds $d_x, d_{x'}$, the pointwise target estimates $e_x, e_{x'}$, and the final target triangle row T before the* UniformModulus$^{\uparrow}$ *handoff. A public consumer cannot choose one center for the source triangle and another center for the target triangle.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier source target two triangle route`

Proof. Begin with source-triangle coherence and radius composition. They produce the source-side bounds from the selected compact-net center, the selected precision row, and the displayed source triangle.

Next read target-triangle coherence with center reuse determinacy. The target estimate row T is built from the same center c and the same pointwise target estimates that the carrier already exposes.

The route exhaustion and source-target lock theorems join these two readings. They show that the UniformModulus$^{\uparrow}$ handoff receives the source and target triangle rows as a single finite route, so a second center would be an absent coordinate rather than a valid consumer choice. □

Theorem 1159.17 (Metric-triangle uniform-estimate finite-net consumer lock)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier receives only the finite selected precision, the source bounds, the pointwise target estimates, the final target triangle row, and the structural rows H, C, P, N. The consumer handoff does not export a compactness theorem, an open-cover argument, a semantic continuity principle, or any center selected outside the finite net row.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier finite net consumer lock`

Proof. Project the carrier and NameCert obligations. The finite selected precision, source-distance rows, pointwise target estimates, final target triangle row, and H, C, P, N support are all named inside one accepted packet.

Use the shared-modulus domain and consumer coverage theorems. They identify the handoff data delivered to UniformModulus$^{\uparrow}$ and keep the selected center, source bounds, target estimates, and support rows together.

Finally apply source-target lock and no-completion escape. The consumer receives a finite handoff package only; host compactness, open covers, semantic continuity, and Real completeness are not support rows and therefore cannot be read from the accepted packet. □

Theorem 1159.18 (Metric-triangle uniform-estimate choice refusal)**.** *No accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *route may replace its finite selected center and selected precision by host compactness, choice-selected centers, open-cover compactness, semantic continuity, or Real completeness. Such replacements fail to preserve the compact-net row, source triangle row, pointwise target estimate rows, target triangle row, and* UniformModulus$^{\uparrow}$ *consumer boundary.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier choice refusal`

Proof. The finite route is already exhausted by Definition 1159.6. It names the selected compact-net center, the source and target estimate rows, and the consumer support rows.

Center reuse determinacy and the two-triangle route prevent a hidden choice of centers. The source and target estimates use the same displayed c, so a choice-selected replacement would no longer be the carrier route.

The remaining exclusions follow from no-completion escape, shared-modulus domain, consumer coverage, and source-target lock. Host compactness, open-cover compactness, semantic continuity, and Real completeness are not coordinates of the accepted packet, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff cannot synthesize them from H, C, P, N. □

Theorem 1159.19 (Metric-triangle uniform-estimate two-point target boundary)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier E, the pointwise target estimates delivered to the final uniform-estimate boundary are exactly the two rows e_x and $e_{x'}$ at the selected center c. The first row is the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *estimate from $f(c)$ to $f(x)$, the second is the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *estimate from $f(c)$ to $f(x')$, and the target triangle row T consumes those two rows before any* $\mathsf{UniformModulus}^{\uparrow}$ *handoff can occur.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier two point target boundary`

Proof. Read the source side through source-triangle coherence. It fixes the selected center c, the compact-net coverage row d_x, and the derived source bound $d_{x'}$ before the target estimates are read.

Next apply target-triangle coherence with center reuse determinacy. These theorems identify e_x and $e_{x'}$ as the two pointwise target rows at that same center and identify T as the target metric triangle row built from them.

Finally use route exhaustion and choice refusal. The exhausted $\mathsf{UniformModulus}^{\uparrow}$ boundary lists no third target estimate and no replacement center, while choice refusal blocks host compactness or a choice-selected center from changing the two pointwise target rows. □

Theorem 1159.20 (Metric-triangle uniform-estimate center-bound continuation)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier E, every center-bound route from the selected precision row e_c to the source bounds $d_x, d_{x'}$ and onward to the target triangle row T factors through the displayed* Cont *support inside C. The route cannot bypass the carrier by appealing to host compactness, semantic continuity, open-cover compactness, choice, or real completeness.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier center bound continuation`

Proof. Start with source-triangle coherence and center reuse determinacy. They bind the selected center c to the source bounds $d_x, d_{x'}$ and to the two pointwise target estimates.

Use target-triangle coherence to continue the same route into T. The target estimate is therefore not a separate semantic assertion; it is the next displayed continuation step after the source and pointwise rows.

Route exhaustion then identifies the consumer surface as $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$. Since C is the displayed Cont support for this surface, any accepted center-bound handoff must pass through those continuation rows.

Choice refusal supplies the negative boundary. Host compactness, semantic continuity, open covers, choice, and real completeness are not carrier coordinates and cannot serve as alternate continuations. □

Theorem 1159.21 (Metric-triangle uniform-estimate UniformModulus exact package)**.** *The exact* $\mathsf{UniformModulus}^{\uparrow}$ *consumer package exported by an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier is*

$$(e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N).$$

The package contains the selected finite precision, both source bounds at the selected center, both pointwise target estimates at that center, the final target triangle row, componentwise hsame *transport, displayed* Cont *support,* Pkg *provenance, and the local* NameCert *row; it contains no additional analytic or choice-theoretic boundary data.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier`
`uniform modulus exact package`

Proof. Combine source-triangle coherence with target-triangle coherence. The source theorem supplies $d_x, d_{x'}$ from the selected center route, and the target theorem supplies $e_x, e_{x'}$ and T at the same center.

Use center reuse determinacy to keep these rows in one boundary. The two pointwise target estimates and the final target triangle row cannot move to another center after the source bounds have been fixed.

Route exhaustion gives the full downstream list $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$. This is exactly the package read by $\mathsf{UniformModulus}^{\uparrow}$.

Choice refusal closes the excluded side of the claim. The accepted carrier exports no host compactness row, semantic continuity theorem, open-cover compactness object, choice row, or real-completeness row, so none of them can be added to the exact package. □

Theorem 1159.22 (Metric-triangle uniform-estimate compact-net triangle extraction)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier* E*, compact-net coverage and the source* $\mathsf{Met}^{\uparrow}$ *triangle row extract one shared-center route*

$$(c, e_c, d_x, d_{x'})$$

and no second center route. The extracted data are the selected center from the $\mathsf{CompactMetric}^{\uparrow}$ *finite bundle, the selected precision row supplied by* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*, the compact-net source bound from* x *to* c*, and the source-triangle bound from* x' *to the same* c*; the extraction factors through the carrier, local* NameCert *obligations, source-triangle coherence, and consumer handoff.*

Proof. Project E through the carrier definition. The visible rows already contain the compact-net center c, the finite selected precision e_c, the coverage row d_x, the source distance between x and x', and the source metric triangle ledger inside X. These are the rows supplied by the metric, compact-metric, and finite-fold chapters cited in the opening paragraph.

Apply the local NameCert obligations. Source triangle admissibility reads $d_{x'}$ from the source $\mathsf{Met}^{\uparrow}$ triangle row using e_c, d_x, and the displayed source distance; carrier exposure keeps all four rows in the same packet.

Now use source-triangle coherence and center reuse determinacy. They identify the center used by compact-net coverage with the center used by the derived source bound, so the package exports one route $(c, e_c, d_x, d_{x'})$ rather than parallel center choices.

The consumer handoff fixes the downstream boundary. Since the handoff sees only the finite source rows and the displayed structural support, host compactness, open covers, choice, and real completeness cannot supply another extraction route. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCompactNetTriangleExt`

Theorem 1159.23 (Metric-triangle uniform-estimate UniformModulus export)**.** *For an accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *carrier E, the* ContinuousMap$^{\uparrow}$ *pointwise rows at the compact-net center and the target* Met$^{\uparrow}$ *triangle row export exactly the finite* UniformModulus$^{\uparrow}$ *estimate package*

$$(e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N).$$

The export uses the compact-net triangle extraction, the two pointwise target estimates at c, and the target triangle row T; it does not add a semantic continuity theorem, target completeness principle, or external metric estimate.

Proof. Start from the compact-net triangle extraction theorem. It supplies the shared-center source route $(c, e_c, d_x, d_{x'})$ that the finite UniformModulus$^{\uparrow}$ consumer expects, using the Met$^{\uparrow}$, CompactMetric$^{\uparrow}$, and FiniteNetMinimumFold$^{\uparrow}$ surfaces cited above.

Read the ContinuousMap$^{\uparrow}$ rows in F at this same center. They provide the pointwise target estimates e_x and $e_{x'}$ from $f(c)$ to the two target points, with the componentwise transports recorded in H and with the target triangle ledger read from the cited Met$^{\uparrow}$ chapter.

Apply target-triangle coherence and the two-point target boundary. The target Met$^{\uparrow}$ triangle row consumes exactly e_x and $e_{x'}$ and returns T before any consumer handoff is formed.

Finally apply the UniformModulus exact package theorem. It identifies the exported package as $(e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N)$, while route exhaustion excludes semantic continuity, target completeness, and any external estimate row. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier uniform modulus export`

Theorem 1159.24 (Metric-triangle uniform-estimate finite-route nonescape)**.** *Every accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *route from compact-net coverage to the* UniformModulus$^{\uparrow}$ *export remains inside the finite rows of the carrier. It contains no host compactness row, no open-cover compactness object, no choice principle, no* Real$^{\uparrow}$ *completeness row, and no second selected center.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier finite route nonescape`

Proof. The compact-net triangle extraction theorem binds the source side to one selected center c, one selected precision e_c, and the two source bounds $d_x, d_{x'}$. This is the finite-net route through the cited compact-metric and finite-fold chapters, and center reuse determinacy prevents a second center from entering the route.

The UniformModulus export theorem then binds the target side to the two pointwise estimates $e_x, e_{x'}$ at c and to the target triangle row T. The handoff package is finite, is carried by H, C, P, N, and lands at the cited UniformModulus$^{\uparrow}$ consumer surface.

Use choice refusal and no-completion escape for the negative boundary. Host compactness, open-cover compactness, choice, real completeness, and hidden semantic continuity are not coordinates of E and are not accepted consumer rows.

Therefore the whole compact-net-to-uniform-modulus route is the displayed finite carrier route. It cannot leave the carrier and re-enter through an analytic compactness or completeness principle. □

Theorem 1159.25 (Metric-triangle uniform-estimate primitive scope)**.** *For every accepted* MetricTriangleUniformEs *carrier*

$$E = (X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N),$$

the metric, compact-net, continuous-map, finite-fold, target-triangle, and $\mathsf{UniformModulus}^{\uparrow}$ *obligation rows are scoped by concrete BEDC primitives. The metric rows are finite* BHist *rows, classifier stability is componentwise* hsame*, every replay from compact-net coverage to the consumer handoff is displayed by* Cont*, provenance is carried by* Pkg*, and local admission is the row* N *of* $\mathsf{NameCert}_{\mathsf{MetricTriangleUniformEstimate}^{\uparrow}}$*. No obligation coordinate is read as host compactness, semantic continuity, choice, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateTasteGate single carrier alignment`

Proof. Project the carrier through Definition 1159.1. The source and target metric data, selected compact-net center, finite precision row, source bounds, pointwise target bounds, and target triangle ledger are all named rows of E.

Apply the local NameCert obligations. Carrier exposure reads these rows as the public surface, classifier stability transports them component by component through H, and the source and target admissibility clauses keep the metric rows inside the displayed BHist ledgers.

The continuation support C replays the compact-net-to-consumer route. It passes from the source metric rows through finite-net selection, pointwise target estimates, and the target triangle row before reaching the $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Finally read P and N. The provenance row records the BEDC support chapters, and the local NameCert row admits exactly this primitive surface, so the accepted scope has no coordinate for the excluded analytic or choice-theoretic objects. □

Theorem 1159.26 (Metric-triangle uniform-estimate continuation route scope)**.** *Every accepted compact-net-to-*$\mathsf{UniformModulus}^{\uparrow}$ *consumer route of* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *factors through the displayed* Cont *path from the source metric rows through finite-net selection and into the target triangle rows. The route consumes* $\mathsf{CompactMetric}^{\uparrow}$ *coverage,* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *precision,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise target estimates, the target* $\mathsf{Met}^{\uparrow}$ *triangle ledger, and the exported* $\mathsf{UniformModulus}^{\uparrow}$ *package only as carrier rows and continuation rows. It has no host compactness, open-cover compactness, choice, semantic continuity, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier cont route scope`

Proof. Begin with compact-net triangle extraction. It fixes the selected center c, selected precision e_c, and source bounds $d_x, d_{x'}$ as the finite source route read from the metric, compact-metric, and finite-fold rows.

Next apply the UniformModulus export theorem. The continuous-map rows at c supply e_x and $e_{x'}$, and the target triangle row supplies T, so the consumer package is the displayed tuple $(e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N)$.

The center-bound continuation theorem identifies C as the Cont support for this route. Thus every accepted replay from the source metric boundary to the uniform-modulus consumer passes through the continuation rows already exposed by the carrier.

Use finite-route nonescape for the negative boundary. Since host compactness, open covers, choice, semantic continuity, and real completeness are not rows of E, they cannot provide another factorization of the same consumer route. □

Theorem 1159.27 (Metric-triangle uniform-estimate NameCert scoped surface)**.** *The scoped* $\mathsf{NameCert}_{\mathsf{MetricTriangleUniformEstimate}^{\uparrow}}$ *surface is the finite package whose clauses are carrier exposure,*

classifier stability, source triangle admissibility, target triangle admissibility, and $\mathsf{UniformModulus}^{\uparrow}$ *handoff, all read through the primitive support rows of*

$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{FiniteNetMinimumFold}^{\uparrow}, \mathsf{Rat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{F}$

The source chapters cited in the introduction supply only those rows to this chapter, and the local NameCert surface exports only those rows to downstream consumers.

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier namecert scope`

Proof. Carrier exposure gives the visible tuple E and classifier stability gives the componentwise hsame transport over that tuple. These are the first two local NameCert clauses and they determine how the public carrier can be read.

Source triangle admissibility then reads $d_x, d_{x'}$ through the source $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and BHist support rows. Target triangle admissibility reads $e_x, e_{x'}$ and T through the target $\mathsf{Met}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows.

The handoff clause passes the exact boundary package to $\mathsf{UniformModulus}^{\uparrow}$. The continuation and provenance coordinates are C and P, so the export is a Cont route with Pkg provenance and local $\mathsf{NameCert}$ admission.

The primitive-scope and continuation-route-scope theorems identify this as the full scoped surface. No sibling chapter contributes a host compactness theorem, open-cover object, choice row, semantic-continuity assertion, or real-completeness coordinate to the accepted package. □

Theorem 1159.28 (Metric-triangle uniform-estimate root-unblock certificate spine)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier* E*, the finite root-unblock certificate spine is exactly the conjunction of carrier exposure, source-triangle coherence, target-triangle coherence, finite-net consumer lock, and no-completion escape. The spine reads the displayed rows*

$$X, Y, F, x, x', c, d_x, d_{x'}, e_c, e_x, e_{x'}, T, H, C, P, N$$

and exports no host compactness row, open-cover compactness object, choice principle, semantic continuity theorem, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Carrier exposure is the first row of the local NameCert obligations. It identifies the public read of E with the displayed source metric, target metric, continuous graph, selected center, source-distance, precision, target-estimate, triangle, transport, continuation, provenance, and local naming rows.

Source-triangle coherence and target-triangle coherence provide the two internal triangle routes. The source route fixes $d_x, d_{x'}$ at the selected compact-net center, and the target route fixes $e_x, e_{x'}$ and T at that same center.

The finite-net consumer lock theorem then packages these rows for $\mathsf{UniformModulus}^{\uparrow}$. It keeps the selected center, selected precision, source bounds, pointwise target bounds, target triangle row, and support rows in one consumer boundary.

Finally apply the finite-route nonescape theorem. It removes host compactness, open-cover compactness, choice, semantic continuity, and $\mathsf{Real}^{\uparrow}$ completeness from the accepted route, so the displayed finite conjunction is the whole root-unblock certificate spine. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier root unblock certificate spine`

Theorem 1159.29 (Metric-triangle uniform-estimate dependency readiness)**.** *Every Real-facing dependency row used by the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *root-unblock spine is consumed only as a displayed* BHist*,* hsame*,* Cont*,* Pkg*, or* NameCert *row through*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{FiniteNetMinimumFold}^{\uparrow}, \mathsf{Rat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}.$$

No dependency is consumed as host compactness, open-cover compactness, selected-subsequence data, semantic continuity, a choice principle, or a $\mathsf{Real}^{\uparrow}$ *completeness axiom.*

Proof. Read the NameCert obligations for carrier exposure, classifier stability, source triangle admissibility, target triangle admissibility, and consumer handoff. These obligations name only displayed BHist rows, componentwise hsame transports, Cont consumer routes, Pkg provenance, and the local NameCert registration.

The metric, compact-metric, continuous-map, finite-fold, rational, real-row, and uniform-modulus chapters enter through those displayed rows. Source-triangle coherence consumes the source $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ precision surface; target-triangle coherence consumes the target $\mathsf{Met}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ pointwise surface.

The finite-net consumer lock and UniformModulus export theorems identify the downstream consumer package. They pass exactly $e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$ to $\mathsf{UniformModulus}^{\uparrow}$.

Use finite-route nonescape for the boundary. Since no accepted dependency row is a host compactness object, open-cover compactness object, selected subsequence, semantic continuity theorem, choice principle, or $\mathsf{Real}^{\uparrow}$ completeness row, all Real-facing dependency data have already been consumed in the displayed BEDC row classes. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier l10 dependency readiness`

Theorem 1159.30 (Metric-triangle uniform-estimate consumer boundary package)**.** *The finite boundary exported by* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *to* $\mathsf{UniformModulus}^{\uparrow}$ *is the pair consisting of the root-unblock certificate spine and the dependency-readiness row. A downstream* $\mathsf{UniformModulus}^{\uparrow}$ *consumer may cite this pair as the complete BEDC obligation package for the compact-net-to-target-triangle estimate.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier consumer boundary package`

Proof. The root-unblock certificate spine theorem supplies the constructive estimate route. It combines carrier exposure, source-triangle coherence, target-triangle coherence, finite-net consumer lock, and no-completion escape.

The dependency-readiness theorem supplies the allowed dependency interface. It says that the Real-facing support is read only through displayed BHist, hsame, Cont, Pkg, and NameCert rows over the cited metric, compact-metric, continuous-map, finite-fold, rational, real-row, and uniform-modulus chapters.

Package the two rows together. The first row gives the finite estimate carrier and the second row gives the permitted dependency boundary, so the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives one finite BEDC obligation package and no external compactness, choice, semantic-continuity, or completeness route. □

Theorem 1159.31 (Metric-triangle uniform-estimate public uniformity criterion)**.** *For an accepted* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *carrier* E*, the following finite BEDC surface is exactly the*

public finite uniformity criterion consumed by a $\mathsf{UniformModulus}^{\uparrow}$ *packet: carrier exposure, the local* $\mathsf{NameCert}_{\mathsf{MetricTriangleUniformEstimate}^{\uparrow}}$ *obligations, shared-center triangle composition, compact-net triangle extraction, the* $\mathsf{UniformModulus}^{\uparrow}$ *export, dependency readiness, and the consumer boundary package. Equivalently, a public* $\mathsf{UniformModulus}^{\uparrow}$ *read of the compact-net-to-target-triangle estimate is admissible precisely when it factors through those rows and through no host compactness, open-cover compactness, selected-subsequence data, choice principle, semantic continuity theorem,* $\mathsf{Real}^{\uparrow}$ *completeness row, or second selected center.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier public uniformity criterion`

Proof. For the forward reading, start with an accepted public $\mathsf{UniformModulus}^{\uparrow}$ read of E. Carrier exposure and the local NameCert obligations identify the displayed rows $c, e_c, d_x, d_{x'}, e_x, e_{x'}, T, H, C, P, N$ as the only rows available to that read.

The shared-center triangle composition and compact-net triangle extraction theorems then determine the source side of the route. They bind the compact-net center, selected precision, compact-net coverage row, and source triangle output before any target estimate is read.

Next apply the $\mathsf{UniformModulus}^{\uparrow}$ export and dependency-readiness theorems. The export supplies the target pointwise rows and the final target triangle row, while dependency readiness restricts all support to displayed BHist, hsame, Cont, Pkg, and NameCert rows over the cited source chapters.

For the reverse reading, assume exactly this finite surface. The consumer boundary package repacks the root-unblock certificate spine together with the dependency-readiness row. Hence the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the complete public finite uniformity criterion and no excluded analytic or choice-theoretic route. □

Theorem 1159.32 (Metric-triangle uniform-estimate Heine–Cantor last step)**.** *In the* MetricTriangleUniformEstima *packet, the Heine–Cantor last-step route is the finite composition in which* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coverage supplies the selected center and the source bounds,* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise estimates at that center supply the two target bounds, and the target* $\mathsf{Met}^{\uparrow}$ *triangle row yields the displayed uniform estimate. The route uses no open-cover compactness, choice, selected subsequence,* $\mathsf{Real}^{\uparrow}$ *completeness, or host semantic continuity.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier heine cantor last step`

Proof. Read compact-net triangle extraction first. It gives the selected center c, selected precision e_c, compact-net source bound d_x, and source-triangle bound $d_{x'}$ inside the finite carrier route.

At the same center, read the two $\mathsf{ContinuousMap}^{\uparrow}$ pointwise rows from the $\mathsf{UniformModulus}^{\uparrow}$ export theorem. These are the rows e_x and $e_{x'}$, and center reuse determinacy keeps them attached to the compact-net center already chosen by the source route.

Apply target-triangle coherence and the two-point target boundary. The target $\mathsf{Met}^{\uparrow}$ triangle row consumes exactly e_x and $e_{x'}$ and returns T, the displayed uniform estimate between the two target values.

Finally apply finite-route nonescape and dependency readiness. They state that every step of this last route is already carried by the finite BEDC rows and that the excluded open-cover, choice, subsequence, semantic-continuity, and completeness objects have no coordinate in the accepted packet. □

Theorem 1159.33 (Metric-triangle uniform-estimate UniformModulus readiness)**.** *A downstream* UniformModulus$^{\uparrow}$ *packet may cite Definition 1159.31 together with Definition 1159.30 as its complete BEDC input surface for the compact-net-to-target-triangle estimate. No additional* Met$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, *or host analytic datum is required beyond the displayed rows already exported by* MetricTriangleUniformEstimate$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier uniformmodulus readiness`

Proof. The public uniformity criterion names the complete finite surface that a UniformModulus$^{\uparrow}$ consumer is allowed to read. It includes carrier exposure, NameCert obligations, shared-center triangle composition, compact-net extraction, the export theorem, and dependency readiness.

The consumer boundary package supplies the packaged input object. It joins the root-unblock certificate spine with the dependency-readiness row, so the downstream packet receives one boundary rather than a collection of local rows that must be rediscovered.

It remains only to rule out missing inputs. The Heine–Cantor last-step theorem and finite-route nonescape show that the compact-net coverage, pointwise estimates, and target triangle row already produce the final estimate inside the carrier. Therefore the cited criterion and boundary package are the whole BEDC input surface for UniformModulus$^{\uparrow}$ readiness. □

Theorem 1159.34 (Metric-triangle uniform-estimate public closure package)**.** *The public export surface of* MetricTriangleUniformEstimate$^{\uparrow}$ *for downstream* UniformModulus$^{\uparrow}$ *consumers is the package consisting of Definition 1159.30, Definition 1159.31, Definition 1159.32, and Definition 1159.33. Every public read of the compact-net-to-target-triangle estimate factors through the displayed* CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows, and through no open-cover compactness, choice principle, selected-subsequence datum, host semantic-continuity theorem, or* Real$^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier public closure package`

Proof. The consumer-boundary package supplies the public object boundary: it joins the root-unblock certificate spine with dependency readiness and makes that pair the complete BEDC obligation package for a downstream UniformModulus$^{\uparrow}$ consumer.

The public uniformity criterion then identifies the admissible read surface. It says that the consumer may read carrier exposure, local NameCert obligations, shared-center triangle composition, compact-net triangle extraction, the UniformModulus$^{\uparrow}$ export, dependency readiness, and the boundary package, and that no excluded analytic or choice-theoretic route is part of the public criterion.

The Heine–Cantor last-step theorem supplies the estimate route inside that surface. Compact-net coverage chooses the center, continuous-map pointwise rows supply the two target estimates, and the target Met$^{\uparrow}$ triangle row returns the displayed uniform estimate without open-cover compactness, selected subsequences, semantic continuity, choice, or completeness data.

Finally, UniformModulus readiness repacks these components as the complete input surface for downstream UniformModulus$^{\uparrow}$ use. Hence the four cited theorem blocks compose to a single public closure package whose public reads factor exactly through the displayed BEDC rows and through none of the excluded data. □

Theorem 1159.35 (Metric-triangle uniform-estimate compact-continuous bridge)**.** *Every accepted* MetricTriangleUniformEstimate$^{\uparrow}$ *packet bridges the* CompactMetric$^{\uparrow}$ *finite-net coverage row, the* FiniteNetMinimumFo

selected precision row, the ContinuousMap$^{\uparrow}$ *pointwise estimate rows, the target* Met$^{\uparrow}$ *triangle ledger, and the exported* UniformModulus$^{\uparrow}$ *consumer package through the displayed* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The bridge route is exactly the public package of Definition 1159.34 read together with the Heine–Cantor last step, UniformModulus readiness, compact-net triangle extraction, and the consumer boundary package. It contains no open-cover compactness, choice principle, selected-subsequence datum, semantic-continuity theorem,* Real$^{\uparrow}$ *completeness row, or host compactness datum.*

Proof. Start with the public closure package. It identifies the displayed CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows as the public read surface for the compact-net-to-target-triangle estimate.

Apply compact-net triangle extraction and the Heine–Cantor last-step theorem. The compact-metric finite-net row supplies the selected center and source bounds, the finite-fold row supplies the selected precision, the continuous-map rows supply the two pointwise target estimates, and the target metric triangle ledger returns the final target estimate.

Next apply UniformModulus readiness and the consumer boundary package. These two rows repack the estimate route as the downstream UniformModulus$^{\uparrow}$ input surface, with the root-unblock certificate spine and dependency-readiness row carried as one public consumer package.

Finally use finite-route nonescape and dependency readiness. Since every step has already been read through displayed BHist, hsame, Cont, Pkg, and NameCert coordinates, no open-cover compactness, choice principle, selected subsequence, semantic-continuity theorem, Real$^{\uparrow}$ completeness row, or host compactness datum remains available to the route. □

Lean4 ✓ `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier`
`compact continuous bridge`

Closure status for NameCert$_{\text{MetricTriangleUniformEstimate}^{\uparrow}}$.

Constructive story: A MetricTriangleUniformEstimate$^{\uparrow}$ packet is generated from source and target metric rows, a continuous-map pointwise row, two source histories, a selected compact-net center, two source-distance rows, a finite-fold precision row, two pointwise target estimates, a final target triangle row, componentwise hsame transport, Cont consumer routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetricTriangleUniformEstimate}^{\uparrow}}$, maturePackageClosed)

Scope closed: The obligation scope records carrier exposure, classifier stability, source triangle admissibility, target triangle admissibility, UniformModulus$^{\uparrow}$ consumer handoff, and the public closure package of Definition 1159.34.

Formal verification: FormalStatus(NameCert$_{\text{MetricTriangleUniformEstimate}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier`

Bridge status: paperBridge

Not claimed: This chapter does not close a Lean target, a full compact-uniform-continuity theorem, open-cover compactness, selected subsequences, choice principles, or real completeness.

Upgrade path: The mature compact-net-to-UniformModulus$^{\uparrow}$ package is recorded on the paper closure axis; remaining Lean bridge verification stays on the independent formal axis.

Chapter 1160

A Concrete Naming Certificate for $\mathsf{CertifiedOnticTower}^{\uparrow}$

The $\mathsf{CertifiedOnticTower}^{\uparrow}$ packet realizes the certified-tower direction of section 9414.15 as a finite staged NameCert carrier. It sits next to the ontic-source surface of chapter 1125, the approximation carrier of chapter 1127, the observation-routing surface of chapter 1126, the model-state surface of chapter 329, and the certified-truth surface of chapter 1130. Those siblings expose source, approximation, route, model, or truth-certificate rows separately; this carrier binds staged signature refinement and ledger continuity before stable ontic content is read from the tower.

Definition 1160.1 (CertifiedOnticTower carrier)**.** *A* $\mathsf{CertifiedOnticTower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, T, R, L, K, H, Q, P, N)$$

with the following displayed rows:

1. *S is the finite stage list naming the observation-bundle levels used by the tower;*
2. *T is the staged signature row, recording the signature exposed at each listed level;*
3. *R is the refinement row, recording preservation, exposure, or correction between consecutive stages;*
4. *L is the ledger-continuity row, naming the hidden, preserved, and exposed content carried through every refinement step;*
5. *K is the classifier row comparing stable tower content across admitted stages;*
6. *H records componentwise* hsame *transport for stage, signature, refinement, ledger, classifier, provenance, and local naming rows;*
7. *Q records displayed* Cont *replay through the same finite staged rows;*
8. *P is the* Pkg *provenance over* $\mathsf{CertifiedOnticTower}^{\uparrow}$, $\mathsf{OnticState}^{\uparrow}$, $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedOnticTower}^{\uparrow}}$ *row.*

The carrier has no coordinate for an imported limit object, total reality oracle, unledgered stage transition, bare model state, or certified-truth row detached from the staged ledger.

Lean4 (definition) `BEDC.Derived.CertifiedOnticTowerUp.taste gate`

Definition 1160.2 (CertifiedOnticTower classifier)**.** *For accepted packets*

$$C = (S, T, R, L, K, H, Q, P, N) \quad \textit{and} \quad C' = (S', T', R', L', K', H', Q', P', N'),$$

the predicate $\mathsf{CertifiedOnticTowerClassifier}(C, C')$ *holds when the stage list, staged signatures, refinement rows, ledger-continuity rows, and stable-content classifier are transported componentwise and replayed through the displayed* Cont *routes without changing which content is preserved, newly exposed, or corrected at each stage.*

Theorem 1160.3 (CertifiedOnticTower ledger continuity)**.** *Every accepted* $\mathsf{CertifiedOnticTower}^{\uparrow}$ *public read of stable content from* $C = (S, T, R, L, K, H, Q, P, N)$ *factors through the ledger-continuity row* L *together with the stage list* S*, staged signatures* T*, refinement row* R*, classifier row* K*, transport row* H*, replay row* Q*, provenance row* P*, and local name* N*. Removing* L *leaves the stagewise signature rows readable, but it does not license a stable-content read across the tower.*

Proof. Project the carrier of Definition 1160.1. The stage list and staged signatures expose the finite tower levels, while R records the transition relation between consecutive levels.

The continuity information needed for stable content is not stored in S, T, or R alone. It is the displayed row L that names what each transition preserves, hides, and exposes, and the classifier row K reads stability only through that named transition ledger.

Rows H, Q, P, N transport, replay, source, and name the same finite packet. If L is removed, a consumer can still inspect the individual stage signatures in T, but the carrier has no row from which to certify that later stages preserve the earlier stable content. □

Lean4 ✓ `BEDC.Derived.CertifiedOnticTowerUp.CertifiedOnticTower ledger continuity`

Theorem 1160.4 (CertifiedOnticTower signature stability)**.** *Let* $C = (S, T, R, L, K, H, Q, P, N)$ *be an accepted* $\mathsf{CertifiedOnticTower}^{\uparrow}$ *packet. Any stable signature read across admitted stages must preserve the staged signature row* T*, the refinement row* R*, and the ledger-continuity row* L *under the componentwise transport displayed by* H *and the replay displayed by* Q*. In particular, stability is a property of the staged packet, not of a single endpoint signature or a bare model state.*

Proof. Use the classifier of Definition 1160.2. Classifier agreement transports the stage list, staged signatures, refinement rows, and ledger-continuity rows together before it compares stable tower content.

The sibling chapters provide adjacent surfaces but do not replace this packet. $\mathsf{OnticState}^{\uparrow}$ supplies an ontic-source row, $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ supplies an approximation tower row, $\mathsf{ObservationReflectionPacket}^{\uparrow}$ supplies an observation route, $\mathsf{StateSpaceModel}^{\uparrow}$ supplies a model state, and $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ supplies a truth-certificate surface.

Thus an accepted stable-signature read must remain inside the displayed carrier. Dropping R or L, or replacing T by a single endpoint from a sibling surface, leaves the $\mathsf{CertifiedOnticTower}^{\uparrow}$ classifier and cannot certify tower-level stability. □

The later Lean file `lean4/BEDC/Derived/CertifiedOnticTowerUp/TasteGate.lean` must include explicit $\mathsf{Nontrivial}$ $\mathsf{CertifiedOnticTower}^{\uparrow}$ and $\mathsf{FieldFaithful}$ $\mathsf{CertifiedOnticTower}^{\uparrow}$ instances. These instances must distinguish stage list, staged signatures, refinement, ledger continuity, classifier, transport, replay, provenance, and local NameCert rows.

Falsifiable prediction: Removing the ledger-continuity row breaks stable-content readback while leaving stagewise signature rows readable.

Independence witness: onticstate, realityconstrainedapproximationtower, observationreflectionpacket, statespacemodel, certifiedphysicaltruth: none is bijective because the tower carrier stores staged refinement and ledger continuity, while the siblings store a single ontic source, approximation tower packet, observation route, model state, or truth-certificate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CertifiedOnticTower}\uparrow}$.

Constructive story: The certified ontic tower is generated from a finite $BHist$ packet whose visible rows already contain the tower layers, certificate row, and signature residue used to recognize the tower. The carrier convention reads each layer through $hsame$-stable anchors: changing the representing history may rename the packet, but it cannot add a hidden level or replace the ledger row. Continuation data enter only through Cont extensions that preserve the displayed certification spine. The classifier is therefore bottom-up: first the tower carrier records the layer ledger, then the certificate bit is read back from the same history, and finally signature stability checks that all later consumers see the same certified tower route.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertifiedOnticTower}\uparrow}, \mathsf{seed})$

Scope closed: This chapter binds the CertifiedOnticTower carrier, classifier, ledger-continuity theorem, and signature-stability theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertifiedOnticTower}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim host-level ontic equality, ambient tower completeness, or certification by any row outside the displayed $BHist$ ledger.

Upgrade path: Reaching obligationClosure requires a Lean carrier for the tower packet plus checked preservation of the classifier under $hsame$ and Cont.

Chapter 1161

A Concrete Naming Certificate for $\mathsf{ReflectionGapBoundary}^{\uparrow}$

The $\mathsf{ReflectionGapBoundary}^{\uparrow}$ packet realizes the reflection-gap direction of Definition 9414.413 as a finite boundary NameCert surface. It sits beside $\mathsf{ObservationReflectionPacket}^{\uparrow}$, $\mathsf{FiniteObservationInterface}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, $\mathsf{ObservationConservationLedger}^{\uparrow}$, and $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$: those siblings expose finite source traces, finite observation interfaces, closed observation ledgers, monotone conservation records, and checkpoint boundaries, while this packet records the gap between certified self-description and total truth without identifying physical reality with MetaCIC. The source-trace row is read through chapter 1126, so the reflection-gap boundary consumes a finite observation-reflection packet before it records any open continuation or total-truth refusal.

Definition 1161.1 (ReflectionGapBoundary carrier)**.** *A* $\mathsf{ReflectionGapBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, H, \Sigma, K, P, L, Q, T, R, N, B)$$

with the following displayed rows:

1. *S is the source trace row inherited from the finite inquiry side of* $\mathsf{ObservationReflectionPacket}^{\uparrow}$*;*
2. *H is the finite history row of observed, reflected, and replayable inquiry events;*
3. *Σ is the signature extraction row read from S through H;*
4. *K is the classifier row comparing extracted self-description data with the displayed reflection claim;*
5. *P is the package provenance row over source trace, history, signature, classifier, ledger, continuation, transport, route, and boundary rows;*
6. *L is the ledger row recording accepted finite correspondences, rejected total-truth claims, and open reflection gaps;*
7. *Q is the open continuation row recording the inquiry still available after the finite certificate is read;*
8. *T is the componentwise* hsame *transport row between observation-side and reflection-side reads;*
9. *R is the displayed* Cont *route through $S, H, \Sigma, K, P, L, Q, T$;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ReflectionGapBoundary}^{\uparrow}}$ *row naming exactly this packet;*

11. *B is the distinct boundary row marking the reflection gap between finite certified self-description and total truth.*

The carrier has no coordinate for a truth oracle, a host-level physical law, an identity between physical reality and MetaCIC, or a closed proof that the open continuation has been exhausted.

Theorem 1161.2 (ReflectionGapBoundary NameCert obligations)**.** *For the carrier of Definition 1161.1, the local* $\mathsf{NameCert}_{\mathsf{ReflectionGapBoundary}^{\uparrow}}$ *surface consists exactly of the finite obligations for carrier admission, source-trace exposure, history and signature extraction, classifier fit, package provenance, ledgered gap accounting, open-continuation preservation, componentwise transport, displayed route replay, local naming, and the distinct boundary row B. These obligations realize the reflection gap as a finite boundary packet and not as a total truth predicate.*

Lean4 ✓ `BEDC.Derived.ReflectionGapBoundaryUp.TasteGate.ReflectionGapBoundaryTasteGate single carrier alignment`

Proof. Carrier admission projects the displayed packet $G = (S, H, \Sigma, K, P, L, Q, T, R, N, B)$ from the definition. The source trace, finite history, and signature rows are the only rows through which a consumer can read the observation side.

The classifier and package rows are then read through K and P. The packet compares finite extracted signatures with a displayed self-description claim, while package provenance records only the finite inquiry route. It does not compare a hidden source domain with a host truth oracle.

The ledger and open-continuation rows give the reflection-gap boundary. The row L records accepted finite matches and rejected total-truth demands, while Q records the continuation still open after the finite certificate is read.

Finally T, R, N, B transport, route, name, and mark the same boundary. The sibling packets $\mathsf{ObservationReflectionPacket}^{\uparrow}$, $\mathsf{FiniteObservationInterface}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, $\mathsf{ObservationConservationLedger}$ and $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ supply finite observation, conservation, and checkpoint surfaces; none supplies a coordinate that collapses B into total truth. □

Theorem 1161.3 (ReflectionGapBoundary non-escape)**.** *Every accepted* $\mathsf{ReflectionGapBoundary}^{\uparrow}$ *packet exports only*

$$S, H, \Sigma, K, P, L, Q, T, R, N, B.$$

A consumer may read the source trace, finite history, signature extraction, classifier, package provenance, ledgered gaps, open continuation, transport, route, local NameCert row, and distinct boundary row for the reflection gap. It cannot recover an identity between physical reality and MetaCIC, a truth oracle, a host-level physical law, a closed exhaustion proof for the continuation, or a route outside the displayed boundary packet.

Proof. Inspect the carrier of Definition 1161.1. Its only readable coordinates are the displayed source, history, signature, classifier, provenance, ledger, continuation, transport, route, naming, and boundary rows.

Apply the obligation theorem. Each accepted read starts from the finite source trace, passes through the finite history and signature extraction rows, and then reaches the classifier, package, ledger, and continuation rows before any local NameCert read is available.

Use the sibling observation and checkpoint chapters for the boundary check. $\mathsf{ObservationReflectionPacket}^{\uparrow}$ supplies finite observation-reflection role correspondence, $\mathsf{FiniteObservationInterface}^{\uparrow}$ supplies finite observation windows, $\mathsf{ClosedObservationSystem}^{\uparrow}$ and $\mathsf{ObservationConservationLedger}^{\uparrow}$ supply closed-observation and conservation ledgers, and $\mathsf{ReflectionRuntimeCheckpoint}^{\uparrow}$ supplies checkpoint

boundaries. These surfaces preserve finite inquiry rows but do not add total truth or source-domain identity.

Thus a demand for physical-MetaCIC identity, final truth, host physical law, closed continuation exhaustion, or an undisplayed route reaches no carrier coordinate. The accepted packet records that demand at L, Q, B rather than exporting it as data. □

Lean4 ✓ `BEDC.Derived.ReflectionGapBoundaryUp.ReflectionGapBoundary nonescape`

Theorem 1161.4 (ReflectionGapBoundary obligation closure)**.** *The carrier of Definition 1161.1, the local NameCert obligations of Definition 1161.2, the non-escape theorem of Definition 1161.3, and the TasteGate single-carrier alignment together close the obligation surface for* $\mathsf{ReflectionGapBoundary}^{\uparrow}$*. The closed surface consists only of carrier admission, finite source-trace exposure, history and signature extraction, classifier fit, package provenance, ledgered gap accounting, open-continuation preservation, componentwise* hsame *transport, displayed* Cont *replay, local naming, and the distinct boundary row* B*.*

Proof. Project the finite packet $G = (S, H, \Sigma, K, P, L, Q, T, R, N, B)$ from the carrier definition. This gives the only readable rows for the boundary object.

Apply the local NameCert obligation theorem. It records the carrier, source trace, history, signature, classifier, package, ledger, continuation, transport, replay, naming, and boundary clauses as the finite obligation surface.

Apply the non-escape theorem. It blocks any consumer route that asks for source-domain identity, total truth, host physical law, closed continuation exhaustion, or an undisplayed route.

Finally use the TasteGate single-carrier alignment cited above. The alignment keeps the reflected carrier rows on the same finite packet, so the carrier, obligations, non-escape result, and TasteGate row assemble exactly the obligation closure and no stronger public or bridge closure. □

Theorem 1161.5 (ReflectionGapBoundary public export)**.** *Every public* $\mathsf{ReflectionGapBoundary}^{\uparrow}$ *consumer must read the finite packet through the source trace, finite observation-reflection packet, boundary row, open-continuation row, componentwise transport row, displayed replay route, package provenance row, and local* $\mathsf{NameCert}_{\mathsf{ReflectionGapBoundary}^{\uparrow}}$ *row. The public export surface therefore exposes only the displayed source trace, finite observation-reflection data, ledgered boundary, continuation, transport, replay, provenance, and local naming clauses, and exports no total-truth theorem, host physical law, physical-reality identification, closed continuation exhaustion, or undisplayed route.*

Proof. Start with the obligation closure theorem and project the same finite packet $G = (S, H, \Sigma, K, P, L, Q, T, R, N$ The public read begins at the source trace and reaches the observation-reflection data only through the displayed finite history and signature rows.

The boundary and continuation checks then pass through B, L, Q. These rows record the reflection gap, accepted finite correspondences, rejected total-truth demands, and the continuation that remains open after the packet is read.

Finally the public route uses T, R, P, N for transport, replay, provenance, and local naming. The non-escape theorem blocks every demand for total truth, host physical law, physical-reality identification, closed continuation exhaustion, or any route not displayed by the packet, so the export surface is public exactly over the finite boundary rows listed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReflectionGapBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ReflectionGapBoundary}^{\uparrow}$ packet is generated from source trace, finite history, signature extraction, classifier, package provenance, ledgered gaps, open continuation, componentwise transport, displayed route, local NameCert, and a distinct reflection-gap boundary row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ReflectionGapBoundary}\uparrow}$, publicClosed)

Scope closed: Definition 1161.1, Definition 1161.2, Definition 1161.3, Definition 1161.4, Definition 1161.5, and the TasteGate alignment bind the public primitive surface over source trace, finite history, signature extraction, classifier, package, ledger, continuation, transport, replay, local NameCert, and boundary rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ReflectionGapBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert source-domain identity, total truth, host physical law, closed continuation exhaustion, Lean verification, or bridge checking.

Upgrade path: Bridge closure requires a checked standard bridge over the same finite source trace, history, signature, classifier, package, ledger, continuation, transport, replay, local NameCert, and boundary rows.

Chapter 1162

A Concrete Naming Certificate for $\mathsf{CompressionLedgerFunctor}^{\uparrow}$

The $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ packet realizes the functorial composite-ledger side of section 9414.20 as a finite BEDC NameCert carrier. It answers the tower-compression vision of chapter 9414 at the point where chapter 1142 records a composite ledger, chapter 1143 binds endpoint equivalence to ledger compatibility, chapter 1148 exposes hidden-source residue, chapter 1127 supplies the indexed tower carrier, and chapter 1153 supplies the stage-refinement tower. Those siblings name the pieces; this carrier names the classifier that preserves ledger routes under tower-arrow composition.

Definition 1162.1 (CompressionLedgerFunctor carrier)**.** *A* $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, K, E, R, O, M, L, Q, J, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealityConstrainedApproximationTower}^{\uparrow}$ *row naming the indexed tower objects.*
2. *K is the* $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ *row naming the accepted composite compression packet.*
3. *E is the* $\mathsf{TowerEquivalence}^{\uparrow}$ *row naming endpoint, ledger, and descent compatibility for compared towers.*
4. *R is the* $\mathsf{ApproximationTowerResidue}^{\uparrow}$ *row naming hidden-source and failed-descent residue.*
5. *O is the* $\mathsf{OnticTower}^{\uparrow}$ *row naming stage-refinement and ledger-continuity data.*
6. *M is the tower-arrow row, displaying the composable tower arrows whose ledgers are being routed.*
7. *L is the composite-ledger row, recording each tower arrow as a finite source-history, classifier, residue, and endpoint route.*
8. *Q is the classifier row, assigning to each displayed tower arrow its ledger route through L.*
9. *J is the composition row, recording ledger-route composition for every displayed composable pair in M.*
10. *H is componentwise* hsame *transport for $A, K, E, R, O, M, L, Q, J$.*

11. *C is the displayed* Cont *replay route through the tower-arrow and ledger-route rows.*
12. *P is the* Pkg *provenance over* $\mathsf{CompressionLedgerFunctor}^{\uparrow}$*, the five cited siblings,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$ *row.*

The carrier has no coordinate for endpoint-only equality, unledgered tower-arrow composition, hidden source identity, private residue transport, or a second classifier outside Q.

Lean4 (definition) `BEDC.Derived.CompressionLedgerFunctorUp.CompressionLedgerFunctorUp`
Lean4 ✓ `BEDC.Derived.CompressionLedgerFunctorUp.CompressionLedgerFunctorTasteGate single carrier alignment`

Theorem 1162.2 (CompressionLedgerFunctor NameCert obligations)**.** *For the carrier of Definition 1162.1, the local* $\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission for $F = (A, K, E, R, O, M, L, Q, J, H, C, P, N)$;*
2. *tower-object exposure through A, K, E, R, O;*
3. *tower-arrow exposure through M;*
4. *composite-ledger and classifier exposure through L, Q;*
5. *composition-law exposure through J;*
6. *componentwise transport, replay, provenance, and local naming closure through H, C, P, N.*

Every accepted consumer of the functorial ledger must display these rows before using tower-arrow composition as a ledger-preserving operation.

Proof. Project the carrier of Definition 1162.1. The admitted object is the thirteen-row packet $F = (A, K, E, R, O, M, L, Q, J, H, C, P, N)$, so the tower-object, arrow, ledger, classifier, composition, transport, replay, provenance, and local-name coordinates are visible.

The rows A, K, E, R, O bind the packet to the indexed tower, composite compression, tower-equivalence, residue, and ontic-stage surfaces. The tower-arrow row M then supplies the finite composable pairs whose ledger routes can be inspected.

The rows L and Q expose the route assigned to each arrow. The row J is separate because preservation of composition is not implied by endpoint agreement or by the existence of individual arrow ledgers.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no additional coordinate, an accepted read is exhausted by the six displayed obligation rows. □

Lean4 ✓ `BEDC.Derived.CompressionLedgerFunctorUp.CompressionLedgerFunctorCarrier namecert obligations`

Theorem 1162.3 (CompressionLedgerFunctor composition law)**.** *Let $F = (A, K, E, R, O, M, L, Q, J, H, C, P, N)$ be an accepted* $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ *packet. If α and β are composable tower arrows displayed by M, then the classifier row sends their composite to the composite of their ledger routes:*

$$Q(\beta \circ \alpha) = Q(\beta) \circ_{\mathsf{led}} Q(\alpha),$$

where $\circ_{\mathsf{led}}$ is the ledger-route operation recorded by J.

Proof. Use the NameCert obligations of Definition 1162.2. They require every accepted consumer to expose the tower-arrow row M, the composite-ledger row L, the classifier row Q, and the composition row J.

The pair α, β is composable only as displayed by M. The classifier row therefore assigns individual ledger routes through the same ledger row L, not through a private endpoint comparison.

The row J records exactly how the ledger route of a composite tower arrow is obtained from the two displayed component routes. Replaying J gives the stated equality with $\circ_{\mathsf{led}}$. □

Lean4 ✓ `BEDC.Derived.CompressionLedgerFunctorUp.CompressionLedgerFunctorCarrier composition law`

Theorem 1162.4 (CompressionLedgerFunctor sibling independence)**.** *The carrier of Definition 1162.1 is not bijective with any one of* RealityConstrainedTowerCompression$^{\uparrow}$, TowerEquivalence$^{\uparrow}$, ApproximationTowerResidu RealityConstrainedApproximationTower$^{\uparrow}$, *or* OnticTower$^{\uparrow}$. *Each sibling supplies a constituent tower, compression, equivalence, residue, or stage-refinement surface, while* CompressionLedgerFunctor$^{\uparrow}$ *additionally carries the tower-arrow row M, the classifier row Q, and the composition row J as one finite NameCert packet.*

Lean4 ✓ `BEDC.Derived.CompressionLedgerFunctorUp.CompressionLedgerFunctorCarrier sibling independence`

Proof. The compression sibling supplies the composite ledger but not the functorial assignment from tower arrows to ledger routes. The equivalence sibling supplies endpoint and ledger compatibility but not the classifier that sends a composed tower arrow to a composed ledger route.

The residue sibling supplies hidden-source and failed-descent rows, the approximation sibling supplies indexed tower objects, and the ontic sibling supplies stage-refinement and ledger-continuity rows. None of these packets has all three rows M, Q, J.

Because the CompressionLedgerFunctor$^{\uparrow}$ carrier keeps M, Q, J together with the five sibling surfaces and the structural rows H, C, P, N, any bijection to a single sibling would have to erase either a tower-arrow coordinate, a classifier coordinate, or the composition-law coordinate. That erasure is outside the accepted carrier. □

The Lean follow-up path `lean4/BEDC/Derived/CompressionLedgerFunctorUp/TasteGate.lean` must provide explicit Nontrivial CompressionLedgerFunctor$^{\uparrow}$ and FieldFaithful CompressionLedgerFunctor$^{\uparrow}$ instances preserving the indexed tower, compression, equivalence, residue, ontic tower, tower-arrow, composite-ledger, classifier, composition, transport, replay, provenance, and local naming rows.

Falsifiable prediction: An accepted CompressionLedgerFunctorUp classifier is refuted by any composable tower-arrow pair whose composite endpoint agrees while the transported composite ledger differs from the ledger-route composite recorded by the component arrows.

Independence witness: reality-constrained-tower-compression, towerequivalence, approximation-tower-residue, reality-constrained-approximation-tower, ontic-tower: none is bijective because CompressionLedgerFu binds tower objects, composite compression, tower equivalence, residue, ontic refinement, tower arrows, ledger routes, classifier assignment, composition law, transport, replay, provenance, and local NameCert rows in one finite packet. Each sibling exposes a constituent surface without the functorial ledger-preservation classifier.

Theorem 1162.5 (CompressionLedgerFunctor scoped route)**.** *The* CompressionLedgerFunctor$^{\uparrow}$ *carrier, its NameCert obligations, its composition law, its sibling-independence theorem, its falsifiable prediction, its independence witness, and the checked TasteGate marker*

CompressionLedgerFunctorTasteGate_single_carrier_alignment

jointly scope the packet over the displayed ledger-functor rows $A, K, E, R, O, M, L, Q, J$*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, local* $\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$ *row* N*, classifier refutation surface, and sibling-separation witness.*

Proof. Read the carrier through Definition 1162.1. Its rows display the five sibling surfaces, the tower-arrow row, the composite-ledger row, the classifier row, the composition row, and the transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 1162.2 to force every accepted consumer through those displayed coordinates. Then apply Definition 1162.3 for the ledger-route composition boundary and Definition 1162.4 for the refusal of any one-sibling collapse.

The falsifiable prediction supplies the local refutation surface for a classifier that preserves endpoints while breaking ledger composition. The independence witness records the same separation from the five siblings, and the checked TasteGate marker pins the single-carrier alignment. These rows exhaust the scoped packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ packet is generated from indexed tower, composite compression, tower equivalence, residue, ontic refinement, tower-arrow, composite-ledger, classifier, composition, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1162.5 binds the carrier, ledger functor rows $A, K, E, R, O, M, L, Q, J$, hsame transport H, Cont replay C, Pkg provenance P, local NameCert row N, classifier refutation surface, and sibling-separation witness as the scoped surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompressionLedgerFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, endpoint-only equality, unledgered tower-arrow composition, source identity, private residue transport, or a classifier outside the displayed packet.

Upgrade path: Obligation closure requires Lean encoding of carrier admission, tower-arrow classification, ledger-route composition, sibling separation, TasteGate nontriviality, and field faithfulness for $A, K, E, R, O, M, L, Q, J, H, C, P, N$.

Chapter 1163

A Concrete Naming Certificate for $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$

The $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ packet realizes the failure-surface requirement of section 9414.68 as a finite NameCert carrier. It sits between the audit row of chapter 1124 and the TruthCert packet of chapter 1144: those sibling chapters name the audit and truth-export surfaces, while this chapter isolates the finite row that records how a certified physical truth claim can be defeated.

Definition 1163.1 (Reality-constrained failure-surface carrier)**.** *A* $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, D, K, U, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite row of defeating observations and later observation traces;*
2. *D is the domain-exit row recording where the certified claim is no longer licensed;*
3. *K is the classifier-mismatch row comparing reality-facing and model-facing signatures;*
4. *U is the continuation-scope row naming the class over which the claim remains reusable;*
5. *L is the ledger row recording hidden compression, idealization, source loss, and packaging material relevant to the defeat;*
6. *H records componentwise* hsame *transport for O, D, K, U, L, C, P, N;*
7. *C records displayed* Cont *replay of the defeat, domain-exit, mismatch, continuation, and ledger reads;*
8. *P is the* Pkg *provenance over* BHist, hsame, Cont, *ledger, classifier, continuation-scope, and truth-export rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded confirmation, final model–reality identity, hidden classifier choice, unledgered defeat, or truth export outside the displayed failure row.

Lean4 (definition) `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceUp`

Theorem 1163.2 (Reality-constrained failure-surface NameCert obligations)**.** *For the carrier of Definition 1163.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}$ *surface consists exactly of carrier admission, defeating-observation visibility, domain-exit visibility, classifier-mismatch visibility, continuation-scope visibility, ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* N*.*

Proof. Project the carrier $F = (O, D, K, U, L, H, C, P, N)$. The first five rows expose the defeating observations, exits, mismatches, continuation scope, and ledger entries that can overturn a certified truth claim.

Read the structural rows. Transport through H preserves the displayed components, replay through C keeps the consumer inside the same finite route, provenance through P records the source of the failure row, and N names the packet.

Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. A purported failure certificate that omits one of them is a projection of the packet, not the accepted packet itself. □

Theorem 1163.3 (Reality-constrained failure-surface non-escape)**.** *Any truth export that depends on a defeating observation, domain exit, classifier mismatch, continuation boundary, or hidden ledger entry absent from*

$$O, D, K, U, L, H, C, P, N$$

is rejected by the $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ *classifier. In particular, omitting the displayed failure row from a* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *export leaves no accepted route for certified physical truth.*

Proof. Use the NameCert obligations of Definition 1163.2. Defeat, exit, mismatch, continuation, and ledger reads are all projections from the same finite packet.

If a truth export cites a defeat or exit not present in O, D, K, U, L, then the cited row is outside the carrier. The structural rows cannot add it: hsame transports displayed components, Cont replays displayed reads, and Pkg records provenance for the same packet.

Thus the classifier rejects any export that omits the displayed failure row or replaces it by hidden host evidence. The certified truth route remains open exactly because the failure surface is retained. □

Theorem 1163.4 (Reality-constrained failure-surface continuation scope)**.** *For an accepted* RealityConstrainedFailu *packet, continuation reuse is scoped by* U *and replayed only through* C*. A consumer may transport the failure surface along* H *and cite provenance through* P*, but it cannot extend the truth claim beyond the displayed continuation class or erase a listed defeat by moving to a different continuation route.*

Proof. Start from the continuation-scope row U in the carrier. It is the only row that licenses reuse of the truth claim beyond the finite source trace.

The replay row C then determines how the defeat, domain-exit, mismatch, and ledger reads are consumed under that scope. Transport through H preserves those rows componentwise; it does not create a wider continuation class.

Finally apply non-escape from Definition 1163.3. A consumer that moves outside U or erases a listed defeat asks for a coordinate absent from the packet, so the accepted continuation scope is exactly the displayed one. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceUp continuation scope`

Theorem 1163.5 (Reality-constrained failure-surface domain-export refusal)**.** *For an accepted* $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ *packet* $F = (O, D, K, U, L, H, C, P, N)$*, any truth export that crosses the displayed domain-exit row* D *or omits the classifier-mismatch row* K *is rejected by the failure-surface classifier unless the same packet exposes the defeating observation row* O*, continuation-scope row* U*, ledger row* L*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}$ *row* N*.*

Proof. Project the carrier $F = (O, D, K, U, L, H, C, P, N)$. The rows D and K are displayed coordinates, so a truth export that crosses D or omits K asks the classifier to ignore part of the accepted packet.

Invoke the NameCert obligations of Definition 1163.2. The defeating observation, continuation scope, ledger, transport, replay, provenance, and local naming rows are all part of the same finite surface; none may be supplied by a hidden host witness.

Use the continuation-scope theorem Definition 1163.4. Reuse is licensed only through U and replayed only through C, so crossing the domain-exit row cannot create a wider export route.

Finally apply non-escape from Definition 1163.3. Hidden evidence outside $O, D, K, U, L, H, C, P, N$ is not a coordinate of the packet, and the classifier rejects the attempted export unless the full displayed defeat surface is exposed. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceDomainExportRef`

Theorem 1163.6 (Reality-constrained failure-surface bridge boundary)**.** *For an accepted* RealityConstrainedFailureS *packet* $F = (O, D, K, U, L, H, C, P, N)$*, the displayed failure-surface packet is the only bridge-facing boundary consumed by truth-export and model-audit siblings. A consumer may pass defeating observations, domain exits, classifier mismatches, continuation scope, ledger entries, transport, replay, provenance, and local naming through that boundary, but it cannot add a second failure surface, hidden model-reality identity, unbounded confirmation row, unledgered defeat, or private truth-export route.*

Proof. Start from the carrier of Definition 1163.1. It exposes the finite failure-surface tuple $F = (O, D, K, U, L, H, C, P, N)$ as the whole packet available to bridge-facing consumers.

Apply Definition 1163.2 and Definition 1163.5. These theorems force every truth-export and model-audit read to display the same defeat, exit, mismatch, continuation, ledger, transport, replay, provenance, and naming rows.

Use non-escape from Definition 1163.3. Since the carrier lacks hidden model-reality identity, unbounded confirmation, unledgered defeat, and private export coordinates, the displayed packet is the sole bridge-facing boundary. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceUpStdBridge`

Falsifiable prediction: Any certified truth export that omits the displayed $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ row $F = (O, D, K, U, L, H, C, P, N)$ is rejected by the reality-constrained truth classifier.

Independence witness: realityconstrainedmodelaudit, realityconstrainedtruthcert, openfitpacket, lawcertificate, realityconstrainedsynthesisseal: none is bijective to $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ because the carrier isolates defeating observations, domain exits, classifier mismatches, continuation scope, ledger, componentwise transport, replay, provenance, and local naming as one finite failure packet.

The checked single-carrier alignment theorem records the ChapterTasteGate realization for this carrier.

Lean4 ✓ `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ packet is generated from defeating observations, domain exits, classifier mismatches, continuation scope, ledger entries, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1163.1, Definition 1163.2, Definition 1163.3, Definition 1163.4, Definition 1163.5, Definition 1163.6, and

Lean4 ✓ `BEDC.Derived.RealityConstrainedFailureSurfaceUp.taste gate`

bind the carrier, NameCert obligations, non-escape theorem, continuation-scope theorem, domain-export refusal, bridge boundary, and TasteGate instance for the finite failure-surface packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedFailureSurface}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RealityConstrainedFailureSurfaceUp.RealityConstrainedFailureSurfaceTasteGate\_single\_c`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not claim final model–reality identity, unbounded confirmation, hidden classifier choice, or truth export outside the displayed failure row.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 1164

A Concrete Naming Certificate for $\mathsf{RealWindowBudget}^{\uparrow}$

The $\mathsf{RealWindowBudget}^{\uparrow}$ packet concretizes the finite observation route used by section 9414.68. It sits between the selector surface of chapter 853 and the disclosure surface of chapter 611: the first sibling fixes the finite diagonal-selector budget, and the second sibling exposes the observation budget consumed by Real-facing handoffs.

Definition 1164.1 (Real-window budget carrier)**.** *A* $\mathsf{RealWindowBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, V, D, R, S, B, O, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row naming the precision, observation scope, and StreamName extent visible to the consumer;*
2. *V is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *window rows selected by Q;*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks and dyadic-radius rows attached to the windows in V;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after those finite readbacks;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed to the Real-facing consumer;*
6. *B is the synchronized* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector row for the same finite request;*
7. *O is the* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure row that makes the visible request, window, dyadic, handoff, and seal route inspectable;*
8. *H records componentwise* hsame *transports for $Q, V, D, R, S, B, O, P, N$;*
9. *C records displayed* Cont *routes from Q to V, from V to D, from D to R, from R to S, and through the selector and disclosure rows B, O;*
10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, host completeness, host real equality, selected limits, unbounded observation schedules, hidden selector choice, or a private Real seal.

Theorem 1164.2 (Real-window budget NameCert obligations)**.** *For the carrier of Definition 1164.1, the local* $\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$ *surface consists exactly of carrier admission, finite request exposure,* $\mathsf{StreamName}^{\uparrow}$ *window coverage,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing,* $\mathsf{Real}^{\uparrow}$ *seal exposure, synchronized* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector routing,* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* N*.*

Proof. Project the displayed packet $W = (Q, V, D, R, S, B, O, H, C, P, N)$. The rows Q, V, D, R, S expose the finite request, StreamName windows, dyadic readbacks, regular-rational handoff, and Real seal.

Read the sibling rows next. The selector row B is the finite diagonal-budget surface supplied by chapter 853, while O is the observation disclosure surface supplied by chapter 611. These rows attach the same request, windows, readbacks, handoff, and seal to the budget route.

Finally read the structural rows. Transport through H preserves the displayed coordinates componentwise, replay through C keeps every consumer on the finite route, provenance through P records the sibling and kernel sources, and N names the packet. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier namecert obligations`

Theorem 1164.3 (Real-window budget selector stability)**.** *Let* $W = (Q, V, D, R, S, B, O, H, C, P, N)$ *be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. Any consumer that reads the selector row* B *through the same request* Q *must use the displayed StreamName windows* V*, dyadic readbacks* D*, regular handoff* R*, Real seal* S*, disclosure row* O*, and support rows* H, C, P, N*. The selector cannot be replaced by a host-selected limit, a second window family, quotient stream equality, or an undisclosed observation route.*

Proof. Start from the finite request row Q. The displayed route in C sends that request first to the StreamName windows V and then to the dyadic readbacks D before any regular or Real-facing handoff can be consumed.

Attach the selector surface. The $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ sibling records finite selector stability through its precision, mesh, window, dyadic, comparison, and seal rows, and the present packet exposes that sibling only as the row B synchronized with Q, V, D, R, S.

Use the disclosure row O to keep the same route observable. The $\mathsf{RealObservationBudget}^{\uparrow}$ sibling makes the request, windows, dyadic readbacks, handoff rows, and seal rows visible as one finite observation budget.

The structural rows H, C, P, N then preserve, replay, source, and name the same packet. Because no other selector, window family, quotient equality, host limit, or hidden disclosure coordinate is a field of W, selector stability is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier selector stability`

Theorem 1164.4 (Real-window budget Real-completion handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet factors through*

$$Q \longmapsto V \longmapsto D \longmapsto R \longmapsto S$$

together with the selector row B*, disclosure row* O*, componentwise transports* H*, displayed continuations* C*, provenance* P*, and local naming row* N*. Conversely, every accepted packet exposes this finite route as its StreamName, DyadicRatCore, RegSeqRat, and Real face.*

The handoff exports no quotient stream, no host sequence equality, no host completeness theorem, no selected limit, no unbounded observation schedule, and no private Real seal.

Proof. First read the local carrier. By Definition 1164.1, the only Real-facing route begins at Q, passes through the finite windows V, dyadic readbacks D, and regular handoff R, and reaches the seal row S.

Next compose the sibling budget surfaces. The $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ row B supplies the finite selector budget for the same request, and the $\mathsf{RealObservationBudget}^{\uparrow}$ row O supplies the observable disclosure ledger for the same windows, readbacks, handoff, and seal.

For the converse direction, project any accepted carrier onto Q, V, D, R, S. Those rows are precisely its StreamName, DyadicRatCore, RegSeqRat, and Real faces, with H, C, P, N preserving and naming the packet.

Thus the Real and Completion handoff is the displayed finite route. Since neither the carrier nor the cited siblings contain quotient stream equality, host equality, host completeness, selected limits, unbounded schedules, or private seal rows, none can be exported by the handoff. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier real completion handoff`

Theorem 1164.5 (Real-window budget finite-route visibility)**.** *Every accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet exposes the single finite route*

$$Q \to V \to D \to R \to S$$

together with the selector row B, disclosure row O, and structural rows H, C, P, N. This visible route consumes $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector budget, and the* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure budget as one packet-level handoff. It exports no quotient stream, host equality, selected limit, host completeness claim, or hidden observation route.*

Proof. Project the carrier of Definition 1164.1. Its rows list exactly $Q, V, D, R, S, B, O, H, C, P, N$, with $Q \to V \to D \to R \to S$ as the displayed Real-facing route.

Use the NameCert obligations of Definition 1164.2. They attach the selector row B and disclosure row O to the same request, window, dyadic, handoff, and seal rows, while H, C, P, N provide only transport, replay, provenance, and local naming for that packet.

Now apply selector stability and the Real-completion handoff, Definition 1164.3 and Definition 1164.4. The first prevents replacing B by a detached selector or hidden limit, and the second forces every Real and Completion consumer through the same finite route. Hence the accepted packet makes the finite observation route visible without exporting quotient streams, host equality, selected limits, or completeness claims. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier finite route visibility`

Theorem 1164.6 (Real-window budget finite-tail-filter handoff)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet whose finite route is consumed as a cofinal-tail Real or Completion budget. The consumer reads the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces through the displayed* $\mathsf{FiniteTailFilter}^{\uparrow}$ *packet of chapter 1119; for a regular Cauchy tail source, the read also factors through chapter 1605. No detached finite observation budget, private tail window, separate dyadic ledger, or ambient completion seal is exported.*

Proof. Start with finite-route visibility and the Real-completion handoff, Definition 1164.5 and Definition 1164.4. They identify every Real or Completion read of W with the displayed $Q, V, D, R, S, B, O, H, C, P, N$ route.

Now apply tail-window exactness and Real-seal handoff for $\mathsf{FiniteTailFilter}^{\uparrow}$, Definition 1119.4 and Definition 1119.5. These bind the shared stream window, dyadic ledger, regular readback,

and Real seal to the same cofinal-tail packet. For regular Cauchy sources, compose with Definition 1605.4. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier finite tail filter handoff`

Theorem 1164.7 (Real-window budget carrier admission obligation)**.** *An accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet is admitted exactly by the finite tuple*

$$W = (Q, V, D, R, S, B, O, H, C, P, N).$$

Carrier admission has no quotient stream coordinate, host equality coordinate, hidden selector coordinate, private seal coordinate, selected limit coordinate, unbounded schedule coordinate, or host completeness coordinate.

Proof. Start with the carrier definition, Definition 1164.1. Its displayed tuple lists exactly $Q, V, D, R, S, B, O, H, C,$ and the definition explicitly excludes quotient streams, host completeness, host real equality, selected limits, unbounded observation schedules, hidden selector choice, and private Real seals.

Use the NameCert obligations of Definition 1164.2. They make the displayed rows the local admission surface and leave H, C, P, N as transport, replay, provenance, and naming rows rather than extra carrier coordinates.

Selector stability and the Real-completion handoff, Definition 1164.3 and Definition 1164.4, keep every accepted read on the same route. Finite-route visibility, Definition 1164.5, then rules out a second admitted route. Thus carrier admission is exactly the displayed tuple. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier admission obligation`

Theorem 1164.8 (Real-window budget window coverage obligation)**.** *For an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet* $W = (Q, V, D, R, S, B, O, H, C, P, N)$*, the request row* Q *covers precisely the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows* V*. Every selector read and every observation-budget read must reuse those windows through the support rows* H, C, P, N*.*

Proof. Project the packet through Definition 1164.1. The only window row named by the finite request Q is V, and the subsequent dyadic row D is attached to those windows.

Apply the local NameCert obligations, Definition 1164.2. They tie request exposure, StreamName-window coverage, selector routing, disclosure routing, transport, replay, provenance, and naming to the same carrier rows.

Selector stability, Definition 1164.3, prevents B from reading a second window family. The Real-completion handoff and finite-route visibility, Definition 1164.4 and Definition 1164.5, force the observation-budget route through $Q \to V \to D \to R \to S$ with H, C, P, N. Hence both selector and disclosure reads reuse precisely the displayed windows. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier window coverage obligation`

Theorem 1164.9 (Real-window budget dyadic handoff seal obligation)**.** *For an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet* $W = (Q, V, D, R, S, B, O, H, C, P, N)$*, the route*

$$V \to D \to R \to S$$

is the only accepted $\mathsf{DyadicRatCore}^{\uparrow}$*-to-*$\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal handoff for the packet.*

Proof. Read the route from the carrier definition, Definition 1164.1. The window row V feeds the $\mathsf{DyadicRatCore}^{\uparrow}$ readback row D, then the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R, and finally the $\mathsf{Real}^{\uparrow}$ seal row S.

The NameCert obligations, Definition 1164.2, expose dyadic readbacks, handoff routing, Real seal exposure, displayed replay, provenance, and local naming only along those rows.

Selector stability, Definition 1164.3, keeps the selector row synchronized with the same V, D, R, S chain. The Real-completion handoff and finite-route visibility, Definition 1164.4 and Definition 1164.5, make that chain the sole Real-facing route and exclude private seals or alternate host-level completions. Therefore the accepted seal handoff is exactly $V \to D \to R \to S$. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier dyadic handoff seal obligation`

Theorem 1164.10 (Real-window budget root selector obligation)**.** *Before a* $\mathsf{RealWindowBudget}^{\uparrow}$ *selector is accepted, the root obligation must display the budget request Q, the finite window row V, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, the* $\mathsf{Real}^{\uparrow}$ *seal row S, the synchronized selector row B, the disclosure row O, the componentwise transport row H, the continuation row C, the package row P, and the local* $\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$ *row N. No selector is accepted from a hidden window source or from a Real-facing seal that omits one of these rows.*

Proof. Project the carrier of Definition 1164.1. Its displayed coordinates already list the finite request, window, dyadic observation, regular handoff, Real seal, selector, disclosure, transport, continuation, package, and local name rows.

Use the NameCert obligations of Definition 1164.2. They make each of those rows part of the local acceptance surface, while selector stability Definition 1164.3 prevents replacing the selector by an undisclosed source.

Finally apply the Real-completion handoff and finite-route visibility, Definition 1164.4 and Definition 1164.5. These keep the consumer on $Q \to V \to D \to R \to S$ with the displayed support rows, so root selector acceptance is exactly the listed obligation surface. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier root selector obligation`

Theorem 1164.11 (Real-window budget finite-window source exhaustion)**.** *For every accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, finite window reads exhaust the allowed* $\mathsf{StreamName}^{\uparrow}$ *source surface. A consumer may read the windows V through the request Q, the selector row B, the disclosure row O, and the support rows H, C, P, N, but it cannot import an additional stream source, unbounded observation schedule, hidden selector, quotient stream, or selected host limit.*

Proof. Start with the route made visible in Definition 1164.5. The only displayed window source is V, reached from the finite request Q and then consumed by the dyadic row D.

Use selector stability, Definition 1164.3. The selector row B is synchronized with the same request, windows, dyadic readbacks, handoff, seal, disclosure, and support rows, so it cannot choose a second window family.

The NameCert obligations then exhaust the local surface. Since the carrier contains no coordinate for another stream source or unbounded schedule, the finite windows V are the whole allowed $\mathsf{StreamName}^{\uparrow}$ source surface for the accepted packet. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier finite window source route certificate`

Theorem 1164.12 (Real-window budget Real-seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet factors through the displayed finite budget route*

$$Q \to V \to D \to R \to S.$$

The consumer may use B, O, H, C, P, N *to select, disclose, transport, replay, package, and name that route, but it cannot bypass it into host equality, countable choice, quotient streams, ambient completeness, selected limits, or a private Real seal.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier real seal nonescape`

Proof. Apply the Real-completion handoff Definition 1164.4. It places every Real or Completion consumer on the chain $Q \to V \to D \to R \to S$ together with B, O, H, C, P, N.

Use finite-route visibility, Definition 1164.5, to make that chain the only exposed route of an accepted packet. The carrier and NameCert obligations contain no alternate host equality, choice, quotient, completeness, selected-limit, or private-seal coordinate.

Therefore a Real-seal consumer cannot escape the displayed finite budget route. Any attempted bypass asks for a row absent from the accepted $\mathsf{RealWindowBudget}^{\uparrow}$ packet. □

Theorem 1164.13 (Real-window budget scoped dependency binding)**.** *For every accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet*

$$W = (Q, V, D, R, S, B, O, H, C, P, N),$$

every scoped read factors through the displayed rows $Q, V, D, R, S, B, O, H, C, P, N$*:* V *is the* $\mathsf{StreamName}^{\uparrow}$ *window surface,* D *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback surface,* R *the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* S *the* $\mathsf{Real}^{\uparrow}$ *seal,* B *the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector,* O *the* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure, and* H, C, P, N *the* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No scoped read is accepted outside this displayed dependency binding.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier scoped dependency binding`

Proof. Project the carrier of Definition 1164.1. Its finite packet lists precisely $Q, V, D, R, S, B, O, H, C, P, N$ and assigns each row to the named StreamName, dyadic, regular, Real, selector, disclosure, history, transport, continuation, package, and naming surfaces.

Use the NameCert obligations of Definition 1164.2. They make the same rows the complete acceptance surface: carrier admission, finite request exposure, window coverage, dyadic readback exposure, handoff routing, Real seal exposure, selector routing, disclosure routing, componentwise transport, displayed replay, provenance, and local naming.

The selector-stability and Real-completion handoff results, Definition 1164.3 and Definition 1164.4, keep every consumer on the same rows. Finite-route visibility, finite-window source exhaustion, and Real-seal non-escape, Definition 1164.5, Definition 1164.11, and Definition 1164.12, then rule out hidden sources, alternate selectors, host equality, selected limits, quotient streams, and ambient completeness. Therefore the dependency envelope is exactly the displayed packet. □

Theorem 1164.14 (Real-window budget selector-disclosure scope)**.** *In an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet, the selector row* B *and the disclosure row* O *are readable only as sibling-scoped rows tied to the same finite request route*

$$Q \to V \to D \to R \to S.$$

No consumer may detach B *from* O*, read* O *against a different selector, or route either row through a second request, hidden window source, selected limit, host equality, quotient stream, or private Real seal.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.SelectorDisclosureScope.RealWindowBudgetCarrier selector disclosure scope`

Proof. Begin with the finite request route in Definition 1164.5. It exposes $Q \to V \to D \to R \to S$ together with B, O, H, C, P, N as one accepted packet-level handoff.

Apply selector stability, Definition 1164.3. The selector row B must reuse the displayed windows, dyadic readbacks, regular handoff, Real seal, disclosure row, and support rows, so it is not a detached source of limits or a second window family.

Now use the Real-completion handoff and Real-seal non-escape, Definition 1164.4 and Definition 1164.12. They force every Real-facing read, including disclosure through O, through the same finite route and exclude host equality, quotient streams, ambient completeness, selected limits, and private seals.

Finally, the carrier and NameCert obligations identify B as the $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ sibling row and O as the $\mathsf{RealObservationBudget}^{\uparrow}$ sibling row for the same packet. Since no further selector or disclosure coordinate exists, both rows are readable only in this sibling-scoped sense. □

Theorem 1164.15 (Real-window budget Real-completion scope)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet may use only the displayed finite route*

$$Q \to V \to D \to R \to S$$

and the support rows B, O, H, C, P, N*. Such a consumer cannot import quotient streams, host equality, selected limits, hidden choice, ambient completeness, unbounded observation schedules, or any private completion object absent from the packet.*

Proof. Use the Real-completion handoff, Definition 1164.4. It states that every Real and Completion consumer factors through $Q \to V \to D \to R \to S$ with the selector, disclosure, transport, continuation, provenance, and naming rows B, O, H, C, P, N.

Finite-route visibility, Definition 1164.5, makes that route the exposed packet surface. The carrier admission and NameCert obligation theorems, Definition 1164.7 and Definition 1164.2, identify all rows available for admission, replay, provenance, and local naming.

The finite-window source exhaustion and Real-seal non-escape results, Definition 1164.11 and Definition 1164.12, remove the remaining escape routes. They deny an additional StreamName source, unbounded schedule, hidden selector, quotient stream, selected host limit, host equality, ambient completeness, and private Real seal.

Thus the Real and Completion scope of the packet is exactly the displayed finite route plus B, O, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier real completion scope`

Theorem 1164.16 (Real-window budget terminal pullback coherence)**.** *Every accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet whose finite route*

$$Q \to V \to D \to R \to S$$

is consumed by a $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *terminal route and a* $\mathsf{RealCompletionExactBoundaryUp}$ *four-face boundary factors through the same displayed rows* $Q, V, D, R, S, B, O, H, C, P, N$*. Selector-budget totality and exact-boundary four-face exactness agree on the StreamName window, RegSeqRat readback, and Real seal faces, and the pullback introduces no quotient stream, selected limit, host equality, countable choice, or ambient completeness coordinate.*

Proof. Start from the local Real-completion scope, Definition 1164.15. It restricts every Real or Completion-facing consumer of the packet to the finite route $Q \to V \to D \to R \to S$ together with the support rows B, O, H, C, P, N.

Apply selector-budget totality and selector-seal terminality, Definition 1081.6 and Definition 1081.11. These identify the selector-seal consumer as a terminal read of the same StreamName window, RegSeqRat readback, Real seal, and selector budget rows rather than as an external limit-selection device.

Now use exact-boundary four-face exactness, Definition 1120.6. Its boundary faces read the Real seal through the displayed completion route, so the selector-seal face and exact-boundary face meet on the same packet rows.

The carrier and NameCert obligations of the $\mathsf{RealWindowBudget}^{\uparrow}$ packet list no quotient stream, selected limit, host equality, countable-choice coordinate, or ambient completeness coordinate. Hence the two completion consumers pull back through the displayed finite rows and nowhere else. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier terminal pullback coherence`

Theorem 1164.17 (Real-window budget terminal pullback non-escape)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet whose finite route is consumed simultaneously by a* $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ *terminal route and a* $\mathsf{RealCompletionExactBoundaryUp}$ *four-face boundary. Then every completion-facing coordinate of the common consumer lies among $Q, V, D, R, S, B, O, H, C, P, N$. In particular the common consumer has no quotient stream, selected limit, host equality, countable-choice row, ambient completeness row, metric-completeness row, detached selector row, or second terminal route.*

Proof. Start with the Real-completion scope theorem, Definition 1164.15. It restricts every Real or Completion consumer of the accepted packet to $Q \to V \to D \to R \to S$ together with the support rows B, O, H, C, P, N.

Now apply the terminal pullback coherence theorem, Definition 1164.16. It says that the selector-seal terminal consumer and the exact-boundary four-face consumer meet on those same displayed rows rather than on an external completion object.

On the selector side, use selector-budget totality and terminality, Definition 1081.6 and Definition 1081.11, together with selector-seal non-escape, Definition 1081.4. These force the terminal selector-seal read through its finite selector, window, readback, seal, endpoint, and structural rows and exclude hidden selectors, detached limits, quotient streams, host equality, second completion routes, and ambient completeness.

On the exact-boundary side, use four-face exactness and boundary non-escape, Definition 1120.6 and Definition 1120.3. The four faces expose only the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and terminal Real seal, while the non-escape boundary excludes quotient, selected-limit, host-equality, choice, metric-completeness, ambient-completeness, bridge, and second-route coordinates. Combined with the Real-window budget non-escape surface, these exclusions leave exactly the rows $Q, V, D, R, S, B, O, H, C, P, N$ for the simultaneous consumer. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.TerminalPullbackNonescape.RealWindowBudgetCarrier terminal pullback nonescape`

Theorem 1164.18 (Real-window budget RegSeqRat selector exactness)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. The packet exposes a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R to a Real or Completion consumer exactly when the same finite request Q has the displayed* $\mathsf{StreamName}^{\uparrow}$

window rows V, $\mathsf{DyadicRatCore}^{\uparrow}$ readback rows D, selector row B, observation-disclosure row O, componentwise transport row H, displayed continuation row C, provenance row P, and local naming row N. Thus the accepted RegSeqRat handoff is not a detached sequence, quotient stream representative, selected host limit, or host equality witness; it is the regular-rational face of the displayed finite budget route.

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier regseq selector exactness`

Proof. Start with the NameCert obligations, Definition 1164.2. They make the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row one of the named carrier obligations and attach it to request exposure, StreamName-window coverage, dyadic readback exposure, selector routing, disclosure routing, transport, replay, provenance, and local naming.

Selector stability, Definition 1164.3, then prevents the row B from selecting a different window family or a hidden host limit. The Real-completion handoff, Definition 1164.4, identifies $Q \to V \to D \to R \to S$ as the route consumed by Real and Completion readers.

Finally apply terminal pullback coherence, Definition 1164.16. The selector-seal consumer and the exact-boundary consumer meet on the same StreamName window, RegSeqRat readback, and Real seal faces. Therefore a consumer sees R exactly through $Q, V, D, B, O, H, C, P, N$, with no detached sequence, quotient representative, selected host limit, or host equality witness. □

Theorem 1164.19 (Real-window budget observation-scope exhaustion)**.** *For an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet $W = (Q, V, D, R, S, B, O, H, C, P, N)$, every visible RealWindowBudget consumer route is exhausted by the displayed rows $Q, V, D, R, S, B, O, H, C, P, N$. No such route can appeal to quotient streams, host sequence equality, selected limits, hidden schedules, a detached selector, private Real seals, or an ambient completeness coordinate outside the packet.*

Proof. Use finite-route visibility, Definition 1164.5. It exposes $Q \to V \to D \to R \to S$ together with the selector row B, disclosure row O, and structural rows H, C, P, N as the packet-level route for every visible consumer.

Next apply scoped dependency binding, Definition 1164.13. It binds any accepted scoped read to the same finite request, window, dyadic readback, handoff, Real seal, selector, disclosure, history, continuation, provenance, and local NameCert rows.

Terminal pullback non-escape, Definition 1164.17, removes the completion-facing escape coordinates. Hence the visible observation scope is exhausted by $Q, V, D, R, S, B, O, H, C, P, N$ and cannot cite quotient streams, host sequence equality, selected limits, hidden schedules, detached selectors, private seals, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.ObservationScopeExhaustion.RealWindowBudgetCarrier observation scope exhaustion`

Theorem 1164.20 (Real-window budget Real-seal readback totality)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. The Real seal row S is readable only through the finite chain from* $\mathsf{StreamName}^{\uparrow}$ *windows V to* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks D to the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, together with the selector row B, observation-disclosure row O, and structural rows H, C, P, N. This gives every downstream Real and Completion consumer a finite BEDC route to S and gives no completeness claim, quotient-stream claim, selected-limit claim, or host-Real equality claim.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier real seal readback totality`

Proof. Read the seal through the local Real-completion scope, Definition 1164.15. It places every Real and Completion consumer on $Q \to V \to D \to R \to S$ with B, O, H, C, P, N as the only support rows.

The Real-seal non-escape theorem, Definition 1164.12, rules out a private seal, hidden selector, quotient stream, selected limit, host equality, or ambient completeness witness. The RegSeqRat selector exactness theorem, Definition 1164.18, identifies the handoff row R as the finite selector-readback face rather than a detached host sequence.

The observation-scope exhaustion theorem, Definition 1164.19, then makes the readback total for the public consumer surface. The downstream Real or Completion consumer receives the finite BEDC route to S and receives no claim of completeness, quotient-stream identity, selected host limit, or host-Real equality. □

Theorem 1164.21 (Real-window budget selector pullback exactness)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. A selector-disclosure consumer reaches the Real seal row S if and only if it preserves the same request row Q,* $\mathsf{StreamName}^{\uparrow}$ *window rows V,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, selector row B, disclosure row O, and structural rows H, C, P, N. No quotient stream, host equality, selected limit, hidden schedule, private seal, or ambient completeness coordinate can witness this pullback.*

Proof. For the forward direction, start from the local NameCert obligations, Definition 1164.2. They expose only the displayed request, window, dyadic readback, handoff, Real seal, selector, disclosure, transport, continuation, provenance, and naming rows.

Selector stability and finite-route visibility, Definition 1164.3 and Definition 1164.5, then force any selector-disclosure read to stay on $Q \to V \to D \to R \to S$ with B, O, H, C, P, N as its support. The RegSeqRat selector exactness theorem, Definition 1164.18, identifies R as the handoff reached through those same finite rows rather than through a detached sequence or quotient representative.

Observation-scope exhaustion and Real-seal readback totality, Definition 1164.19 and Definition 1164.20, remove the remaining escape coordinates. Thus a consumer that reaches S has preserved exactly the displayed carrier rows.

For the reverse direction, assume the consumer preserves $Q, V, D, R, S, B, O, H, C, P, N$. Project the accepted packet onto its continuation row C and read the displayed route $Q \to V \to D \to R \to S$ together with the selector and disclosure rows B, O.

The same projection supplies H for componentwise transport, P for provenance, and N for local naming. Since the carrier has no field for quotient streams, host equality, selected limits, hidden schedules, private seals, or ambient completeness, preserving those rows is precisely the pullback condition for reaching the Real seal. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier selector disclosure readback scope`

Theorem 1164.22 (Real-window budget StreamName-RegSeqRat factorization)**.** *Let $W = (Q, V, D, R, S, B, O, H$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. Every accepted consumer of* $\mathsf{RealWindowBudget}^{\uparrow}$ *that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed rows in the order*

$$Q \to V \to D \to R \to S,$$

with B, O, H, C, P, N supplying selector, disclosure, componentwise transport, continuation, provenance, and local naming support. Thus the route first reads the $\mathsf{StreamName}^{\uparrow}$ *window rows, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and only then the* $\mathsf{Real}^{\uparrow}$ *seal.*

No quotient stream, selected limit, hidden schedule, or ambient completeness row can supply an alternate factor of this route.

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier streamname regseqrat factorization`

Proof. Start with selector pullback exactness, Definition 1164.21. It identifies reaching the Real seal with preservation of exactly $Q, V, D, R, S, B, O, H, C, P, N$.

Next read the cross-chapter current-phase route through the StreamName status ledger and the four-object exit theorem, Definition 126.516 and Definition 126.500. These place the public completion path in the order StreamName windows, DyadicRatCore readbacks, RegSeqRat handoff, and RealUp seal.

Finally apply observation-scope exhaustion and Real-seal readback totality, Definition 1164.19 and Definition 1164.20. They remove the remaining non-displayed coordinates, so quotient streams, selected limits, hidden schedules, and ambient completeness rows cannot appear in the factorization. □

Theorem 1164.23 (Real-window budget four-object readback lock)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P, N)$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. The only public current-phase status faces available to a* $\mathsf{RealWindowBudget}^{\uparrow}$ *consumer are exactly* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, each read through the same finite rows $Q, V, D, R, S, B, O, H, C, P, N$. The packet has no public face for a quotient stream, selected limit, hidden schedule, ambient completeness theorem, or additional completion object.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier four object readback lock`

Proof. The StreamName status ledger and four-object exit route, Definition 126.516 and Definition 126.500, identify the current Phase Real and Completion surface with the four named objects: $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Selector pullback exactness, Definition 1164.21, pins those faces to the single displayed RealWindowBudget carrier route. The rows $Q, V, D, R, S, B, O, H, C, P, N$ therefore serve all four status reads rather than four independent host objects.

Observation-scope exhaustion and Real-seal readback totality, Definition 1164.19 and Definition 1164.20, close the public readback surface. Since no carrier coordinate names a quotient stream, selected limit, hidden schedule, ambient completeness theorem, or further completion object, those cannot be public current-phase status faces. □

Theorem 1164.24 (Real-window budget open-phase window-selector lock)**.** *Let $W = (Q, V, D, R, S, B, O, H, C, P,$ be an accepted* $\mathsf{RealWindowBudget}^{\uparrow}$ *packet. It exposes a Real-facing finite observation route only when the* $\mathsf{StreamName}^{\uparrow}$ *window row V,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R,* $\mathsf{Real}^{\uparrow}$ *seal row S,* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *selector row B, and* $\mathsf{RealObservationBudget}^{\uparrow}$ *disclosure row O are all present, transported by the displayed* hsame *row H, replayed through C, and named by N.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier open phase window selector lock`

Proof. Read the carrier definition. It makes $Q, V, D, R, S, B, O, H, C, P, N$ the complete finite packet and gives no coordinate for a detached selector, private disclosure row, quotient stream, selected limit, or host-completeness witness.

Apply selector stability, Real-completion handoff, finite-route visibility, selector-disclosure scope, and Real-completion scope, Definition 1164.3, Definition 1164.4, Definition 1164.5, Definition 1164.14,

and Definition 1164.15. These results force the Real-facing route through V, D, R, S with B, O, H, C, P, N as support rows for the same request.

The four-object readback lock, Definition 1164.23, and the sibling chapters cited at the start of this chapter identify the open-phase route with exactly the displayed window, dyadic, handoff, seal, selector, and disclosure rows. □

Theorem 1164.25 (Real-window budget diagonal observation route exhaustion)**.** *Every public Real-window route claimed by this chapter factors through the displayed finite rows $Q, V, D, R, S, B, O, H, C, P, N$. It cannot be replaced by a hidden selector, host equality, unbounded observation schedule, or quotient-stream witness.*

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier diagonal observation route exhaustion`

Proof. Start from the accepted packet W. The request row Q selects the finite window family V, dyadic readbacks D, regular handoff R, Real seal S, selector budget B, and observation disclosure O; H, C, P, N then transport, replay, source, and name that same route.

The local obligations and scope theorems, Definition 1164.2, Definition 1164.14, and Definition 1164.15, make those rows the public dependency surface. The four-object readback lock then limits the current-phase faces to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ through the same rows.

The sibling CauchyDiagonalBudget and RealObservationBudget surfaces provide only the displayed selector and disclosure rows consumed here. Since neither the carrier nor these scopes provide hidden selectors, host equality, unbounded schedules, or quotient-stream witnesses, no public route can factor through such data. □

Theorem 1164.26 (Real-window budget open-phase four-face status pullback)**.** *Let $W = (Q, V, D, R, S, B, O, H, C$ be an accepted $\mathsf{RealWindowBudget}^{\uparrow}$ packet. A RealWindowBudget consumer is status-correct for the current Phase Real and Completion route if and only if its four public status faces are exactly the $\mathsf{StreamName}^{\uparrow}$ finite-window status, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance-ledger status, the $\mathsf{RegSeqRat}^{\uparrow}$ readback status, and the $\mathsf{Real}^{\uparrow}$ seal status, all read through the displayed rows $Q, V, D, R, S, B, O, H, C, P, N$. No aggregate token, quotient stream, selected limit, ambient completeness theorem, countable-choice row, metric-completeness datum, or bridge row can replace one of these faces or add a fifth status face.*

Proof. For the forward direction, start with selector pullback exactness, Definition 1164.21. It forces every status-correct consumer that reaches the Real seal to preserve the displayed request, window, dyadic readback, handoff, seal, selector, disclosure, transport, continuation, provenance, and naming rows.

Next apply the StreamName status ledger and the four-object exit route, Definition 126.516 and Definition 126.500. These identify the current Phase Real and Completion route with exactly the four public faces: StreamName finite windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and the RealUp seal.

The local non-escape and readback theorems then remove the remaining possible coordinates. By Definition 1164.17, Definition 1164.19, and Definition 1164.20, the accepted packet exports no aggregate token, quotient stream, selected limit, ambient completeness theorem, countable-choice row, metric-completeness datum, bridge row, or private fifth face.

For the reverse direction, assume the consumer reads exactly those four status faces through $Q, V, D, R, S, B, O, H, C, P, N$. The displayed rows already contain the finite window, tolerance

ledger, readback handoff, Real seal, selector, disclosure, transport, replay, provenance, and local NameCert data needed by the RealWindowBudget packet.

Apply selector pullback exactness again to repack this read as the finite selector-disclosure pullback to S. The StreamName status ledger then reads the same four faces as the current Phase Real and Completion status record, so the consumer is status-correct for that route. □

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.RealWindowBudgetCarrier open phase four face status pullback`

Falsifiable prediction: A Real-facing consumer that omits the displayed $\mathsf{RealWindowBudget}^{\uparrow}$ packet $W = (Q, V, D, R, S, B, O, H, C, P, N)$ cannot cite the finite observation route of section 9414.68 as a completed Real handoff.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchydiagonalbudget, realobservationbudget: none is bijective to $\mathsf{RealWindowBudget}^{\uparrow}$ because the carrier packages the request, finite windows, dyadic readbacks, regular handoff, Real seal, selector budget, disclosure budget, componentwise transport, continuation replay, provenance, and local naming as one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{RealWindowBudget}^{\uparrow}$ packet is generated from a finite request, StreamName windows, DyadicRatCore readbacks, RegSeqRat handoff, Real seal, selector budget, disclosure budget, componentwise transport, continuation replay, provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.RealWindowBudgetUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1164.13 binds every accepted scoped read to the displayed $Q, V, D, R, S, B, O, H, C, P, N$ dependency rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealWindowBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host equality, selected limits, host completeness, unbounded observation schedules, hidden selector choice, or bridge checking.

Upgrade path: Verification and bridge checking remain Lean/bridge-axis work for the same finite Real-window budget route.

Chapter 1165

A Concrete Naming Certificate for $\mathsf{ModelSelectionBoundary}^{\uparrow}$

The $\mathsf{ModelSelectionBoundary}^{\uparrow}$ packet realizes the comparative-selection boundary of section 9414.68 as a finite BEDC NameCert carrier for reality-constrained physical truth. It sits beside the open-fit packet of chapter 1137, the model-audit packet of chapter 1124, the signature-fit packet of **??**, and the synthesis seal of chapter 1140. It also consumes the truth-certificate discipline of chapter 1141: those siblings expose fit, audit, signature, synthesis, and truth-certificate faces, while this boundary records which finite candidate currently wins and which displayed defeats keep that selection provisional.

Definition 1165.1 (ModelSelectionBoundary carrier)**.** *A* $\mathsf{ModelSelectionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, F, D, L, T, P, N)$$

with the following displayed rows:

1. *M is the finite model-family row listing the accepted candidate packets under comparison.*
2. *S is the observed-signature row shared by the comparison.*
3. *F is the comparative-fit row recording the selected packet, its displayed fit score, and the losing rows that remain accepted but not selected.*
4. *D is the defeat-surface row naming finite observations, signature mismatches, audit failures, domain exits, and ledger conflicts that overturn the displayed selection.*
5. *L is the selection-ledger row recording admissibility, tie policy, defeat visibility, provenance, and refused final-oracle demands.*
6. *T is the componentwise transport row preserving the candidate family, observed signatures, fit comparison, defeat surface, and ledger under displayed* hsame *routes.*
7. *P is the* Pkg *provenance row over* $\mathsf{ModelSelectionBoundary}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$, $\mathsf{PhysicalTruthCertificate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ModelSelectionBoundary}^{\uparrow}}$ *row naming the whole boundary.*

The carrier has no coordinate for model–reality identity, total confirmation, a hidden winner outside M, an erased losing row, an unledgered defeat, or a final oracle deciding all future observations.

Theorem 1165.2 (ModelSelectionBoundary fit-ledger separation)**.** *For an accepted* ModelSelectionBoundary$^{\uparrow}$ *carrier* $B = (M, S, F, D, L, T, P, N)$*, the selected row in* F *is a finite ledgered selection from the displayed candidate family* M*, not a truth identity between the selected model and physical reality. Every public selection read must expose the observed signatures* S*, the comparative fit row* F*, the losing accepted rows recorded by* F*, the selection ledger* L*, the transport row* T*, provenance* P*, and local name* N *before it can be consumed as a reality-constrained physical-truth certificate.*

Lean4 ✓ `BEDC.Derived.ModelSelectionBoundaryUp.TasteGate.ModelSelectionBoundaryTasteGate single carrier alignment`

Proof. Carrier admission exposes exactly $B = (M, S, F, D, L, T, P, N)$. The only row that selects a candidate is F, and F is indexed by the finite family M and the displayed signature row S.

The ledger row L separates acceptance from selection. A losing row may remain an accepted packet while failing to be the displayed winner; conversely, the winning row is selected only under the current finite comparison and ledger policy.

The transport and provenance rows keep the comparison public. Since T, P, N preserve the same candidate family, signature, fit, ledger, and local name, no consumer can replace the selected row by hidden model identity or final physical truth. □

Theorem 1165.3 (ModelSelectionBoundary defeat surface)**.** *An accepted* ModelSelectionBoundary$^{\uparrow}$ *packet is locally refuted whenever a competing accepted packet in* M *beats the displayed selected row of* F *while the relevant finite defeat is hidden from* D *or erased from* L*. A selection that keeps* D *and* L *visible remains a current finite winner; a selection that suppresses the defeat surface is outside the boundary.*

Lean4 ✓ `BEDC.Derived.ModelSelectionBoundaryUp.TasteGate.ModelSelectionBoundary defeat surface`

Proof. Read the carrier rows componentwise. The family row M supplies the competing accepted packets, the signature row S supplies the shared observation surface, and the comparative-fit row F records the selected and losing rows.

If a competing accepted packet beats the selected row, the defeat must be visible in D and ledgered in L. Those two rows are the only coordinates that expose overturned fit, mismatch, audit failure, domain exit, or policy conflict.

When the defeat is hidden or erased, the claimed selection asks the packet to export a winning status without the displayed boundary rows that make the comparison auditable. Such a route is not an accepted ModelSelectionBoundary$^{\uparrow}$ read. □

Definition 1165.4 (ModelSelectionBoundary obligation carrier)**.** *A* ModelSelectionBoundary$^{\uparrow}$ *obligation carrier is a finite* BHist *packet*

$$O = (B, F, D, L, A, S, T, P, N)$$

where B *is an accepted* ModelSelectionBoundary$^{\uparrow}$ *carrier,* F *is its comparative fit ledger,* D *is its defeat surface,* L *is its selection ledger,* A *is the audit route through chapter 1137, chapter 1124,* **??***, chapter 1140, and chapter 1141,* S *is the corresponding TasteGate alignment row,* T *is componentwise* hsame *transport,* P *is* Pkg *provenance, and* N *is the local* NameCert *row. The carrier has no coordinate for a hidden model oracle, probability semantics, host truth predicate, or final winner outside the displayed finite family.*

Lean4 (definition) `BEDC.Derived.ModelSelectionBoundaryUp.ModelSelectionBoundaryObligationCarrier`

Theorem 1165.5 (ModelSelectionBoundary NameCert obligations)**.** *For every accepted obligation carrier $O = (B, F, D, L, A, S, T, P, N)$, the* ModelSelectionBoundary$^{\uparrow}$ *NameCert obligation surface consists exactly of the fit ledger, defeat surface, selection ledger, sibling audit route, TasteGate alignment row, componentwise transport, package provenance, and local naming displayed in O. The selected model remains a current finite winner only relative to those rows.*

Lean4 ✓ `BEDC.Derived.ModelSelectionBoundaryUp.ModelSelectionBoundary namecert obligations`

Proof. Project the obligation carrier. The rows F, D, L are inherited from the existing fit-ledger and defeat-surface theorems, so selection is exposed as a ledgered finite comparison rather than as an oracle.

The sibling audit route A forces every selection read through open-fit, model-audit, signature-fit, synthesis-seal, and truth-certificate surfaces. The TasteGate row S promotes that finite route only as single-carrier alignment, not as probability semantics or host truth.

Finally T, P, N preserve, source, and name the same displayed rows. Since the carrier contains no hidden oracle, final winner outside the candidate family, probability space, or host truth predicate, the listed rows exhaust the NameCert obligations. □

Theorem 1165.6 (ModelSelectionBoundary audit scope)**.** *Every audit of a* ModelSelectionBoundary$^{\uparrow}$ *selection is limited to the displayed candidate family, observed signatures, fit ledger, defeat surface, selection ledger, sibling audit route, and TasteGate alignment row. A later observation may defeat the current winner by adding a visible row to D and L, but the packet never exports a host-level final model order, probability semantics, or truth predicate.*

Lean4 ✓ `BEDC.Derived.ModelSelectionBoundaryUp.ModelSelectionBoundaryAuditScope`

Proof. Read the audit rows in order. The candidate family and observed signatures determine which finite rows are under comparison, while F records the current selection and losing accepted rows.

The defeat and ledger rows D, L are the only places where a later observation, mismatch, audit failure, domain exit, or ledger conflict can overturn the selection. If such a row is absent, the packet records only the current finite winner; if it is present, the winner is visibly defeated.

The sibling audit route and TasteGate alignment row keep the promotion local to displayed BEDC data. They do not add probability semantics, a host truth predicate, or a final model order, so the audit scope is exactly the finite surface stated above. □

Falsifiable prediction: A selected model packet is refuted whenever its displayed fit row is beaten by a competing accepted packet while the defeat surface is hidden or erased.

Independence witness: openfitpacket, realityconstrainedmodelaudit, realityconstrainedsignaturefit, realityconstrainedsynthesisseal, physicaltruthcertificate: none is bijective because each supplies one fit, audit, signature, synthesis, or truth-certificate face, while ModelSelectionBoundaryUp carries the comparative candidate family, selected row, losing rows, finite defeat surface, ledger, transport, and no-final-oracle refusal together.

The Lean follow-up in `lean4/BEDC/Derived/ModelSelectionBoundaryUp/TasteGate.lean` provides explicit Nontrivial ModelSelectionBoundary$^{\uparrow}$ and FieldFaithful ModelSelectionBoundary$^{\uparrow}$ instances preserving M, S, F, D, L, T, P, N, together with the target `BEDC.Derived.ModelSelectionBoundaryUp.Ta`

Closure status for NameCert$_{\text{ModelSelectionBoundary}^{\uparrow}}$**.**

Constructive story: A ModelSelectionBoundary$^{\uparrow}$ packet is generated from a finite model-family row, observed signatures, comparative fit, finite defeat surface, selection ledger, componentwise transport, package provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ModelSelectionBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1165.1, Definition 1165.2, Definition 1165.3, Definition 1165.4, Definition 1165.5, Definition 1165.6, and

Lean4 ✓ `BEDC.Derived.ModelSelectionBoundaryUp.taste gate`

bind the public exported interface over M, S, F, D, L, T, P, N: candidate-family carrier rows, observed signatures, comparative fit-ledger separation, finite defeat visibility, obligation carrier rows, NameCert obligations, audit-scope limits, TasteGate instance evidence, and no-final-oracle refusal.

Formal verification: FormalStatus(NameCert$_{\text{ModelSelectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final physical truth, model–reality identity, total confirmation, host-level model ordering, Lean verification, or bridge checking for ModelSelectionBoundary$^{\uparrow}$.

Upgrade path: The paper-side mature package is recorded while Lean verification and bridge checking remain outside this closure block.

Chapter 1166

A Concrete Naming Certificate for $\mathsf{AnchorChangeInvariant}^{\uparrow}$

The $\mathsf{AnchorChangeInvariant}^{\uparrow}$ packet realizes the anchor-change invariant surface of section 9414.32 as a finite symmetric transport certificate over observer-history anchors. It sits after the objectivity carrier of chapter 1133, consumes the anchor-stability packet of chapter 1150, reads the invariant row of chapter 656, and supplies a focused sibling to the refusal boundary of chapter 1134. The carrier is neither a global observer nor a view from nowhere; it is the finite table that records when an invariant survives a reversible change of observer anchor.

Definition 1166.1 (Anchor-change invariant carrier)**.** *An* $\mathsf{AnchorChangeInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (H, A, I, S, R, L, T, P, N)$$

with the following displayed rows:

1. *H is the finite observer-history family whose anchors are being compared;*
2. *A is the anchor-change table, with one source anchor, one target anchor, and the displayed reverse row for each admitted comparison;*
3. *I is the* $\mathsf{InterHistInvariant}^{\uparrow}$ *row whose value is transported across the anchor change;*
4. *S is the* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ *row certifying that the invariant read is stable under the displayed table;*
5. *R is the no-privileged-anchor refusal ledger, rejecting routes that factor through one fixed observer anchor;*
6. *L is the local transport ledger recording the observations, redescription steps, and classifier reads used by the table;*
7. *T records componentwise* hsame *transport and displayed* Cont *replay for H, A, I, S, R, L, P, N;*
8. *P is the* Pkg *provenance over* $\mathsf{AnchorChangeInvariant}^{\uparrow}$, $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$, $\mathsf{AnchorStabilityCertificate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, $\mathsf{ObjectivityRefutationBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}$ *row.*

The carrier has no coordinate for a one-way anchor change, privileged observer, hidden synchronizer, anchor-free invariant, quotient of observer histories, or host-side truth value.

Lean4 (definition) `BEDC.Derived.AnchorChangeInvariantUp.AnchorChangeInvariantUp`

Definition 1166.2 (Anchor-change invariant classifier)**.** *The local* $\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}$ *classifier compares accepted carriers only through* $H, A, I, S, R, L, T, P, N$*. Two packets agree when their observer-history families are matched by the displayed transport, their anchor-change tables contain both directional rows for each admitted comparison, their* $\mathsf{InterHistInvariant}^{\uparrow}$ *values are transported through the same classifier, their* $\mathsf{AnchorStabilityCertificate}^{\uparrow}$ *rows certify the same route, and their refusal ledgers reject the same privileged-anchor factorizations.*

Theorem 1166.3 (Anchor-change invariant NameCert obligations)**.** *For the carrier of Definition 1166.1, the local* $\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}$ *surface consists exactly of carrier admission for* $G = (H, A, I, S, R, L, T, P, N)$*, classifier comparison through Definition 1166.2, invariant transport through* I*, anchor-stability certification through* S*, no-privileged-anchor refusal through* R*, ledger exposure through* L*, componentwise transport and replay through* T*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted packet. The rows H, A, I, S, R, L display the observer family, reversible anchor-change table, invariant value, stability certificate, refusal ledger, and route ledger.

Apply the classifier definition. It compares only the displayed rows and requires both directions of each anchor-change entry before the invariant row may be read as transported content.

The structural rows T, P, N preserve that finite surface by componentwise transport, replay, provenance, and local naming. Since the carrier lists no hidden synchronizer or privileged anchor, the stated rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.AnchorChangeInvariantUp.AnchorChangeInvariant namecert obligations`

Theorem 1166.4 (Anchor-change invariant no privileged anchor)**.** *An accepted* $\mathsf{AnchorChangeInvariant}^{\uparrow}$ *packet cannot export an invariant route that factors through one fixed observer anchor without the reverse anchor-change row. Such a route is rejected by the refusal ledger* R *and cannot be read as objective transport.*

Lean4 ✓ `BEDC.Derived.AnchorChangeInvariantUp.AnchorChangeInvariant no privileged anchor`

Proof. Read the anchor-change table A and refusal ledger R from the carrier. Admission requires the displayed reverse row for each comparison and records the rejection of fixed-anchor factorizations.

The classifier of Definition 1166.2 then compares invariant transport only through tables carrying both directions. A one-anchor route lacks the reverse row and is therefore not an accepted comparison.

Consequently the packet may still record local observational content, but it cannot export the invariant as objective anchor-change transport. □

Theorem 1166.5 (Anchor-change invariant anchor-symmetry totality)**.** *Every admitted* $\mathsf{AnchorChangeInvariant}^{\uparrow}$ *invariant read exposes the source-to-target and target-to-source anchor-change rows in the same finite packet before the read reaches any* $\mathsf{RealityConstrainedObjectivity}^{\uparrow}$ *or* $\mathsf{RelationalObjectivity}^{\uparrow}$ *consumer.*

Proof. Use the carrier admission row A. Its table stores both directional rows for each admitted observer-anchor comparison.

The stability row S and invariant row I are read only after that table is visible. The transport row T replays the same displayed comparison, so downstream objectivity consumers receive the invariant together with the symmetric anchor-change evidence.

No additional route can be supplied by a consumer because R, L, P, N keep the refusal ledger, local route ledger, provenance, and naming rows inside the same packet. □

Lean4 ✓ `BEDC.Derived.AnchorChangeInvariantUp.AnchorChangeInvariant anchor symmetry totality`

Falsifiable prediction: If an invariant route factors through one fixed anchor without a reverse anchor-change row, the AnchorChangeInvariant carrier rejects it as non-objective transport.

Independence witness: realityconstrainedobjectivity, anchorstabilitycertificate, interhistinvariant, relationalobjectivity, objectivityrefutationboundary: these siblings provide objectivity, anchor stability, invariant, relational-objectivity, and refutation surfaces, but none combines a symmetric anchor-change classifier with a no-privileged-anchor refusal ledger.

The Lean follow-up must provide `Nontrivial AnchorChangeInvariantUp` and `FieldFaithful AnchorChangeInvariantUp` in `lean4/BEDC/Derived/AnchorChangeInvariantUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}$.

Constructive story: An $\mathsf{AnchorChangeInvariant}^{\uparrow}$ packet is generated from a finite observer-history family, reversible anchor-change table, transported invariant row, anchor-stability certificate, no-privileged-anchor refusal ledger, local transport ledger, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1166.1, Definition 1166.2, Definition 1166.3, Definition 1166.4, and Definition 1166.5 bind the finite carrier, classifier, local NameCert obligations, no-privileged-anchor refusal, and anchor-symmetry totality surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AnchorChangeInvariant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert global observers, view-from-nowhere semantics, hidden anchor privilege, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding each displayed anchor-change carrier, classifier, invariant-transport, anchor-stability, refusal-ledger, route-ledger, and non-escape obligation to its BEDC primitive dependencies.

Chapter 1167

A Concrete Naming Certificate for FiniteObservationResidueBoundary$^{\uparrow}$

The FiniteObservationResidueBoundary$^{\uparrow}$ packet realizes the hidden-residue and failure-surface branch of the model-audit program in section 9414.60. It composes the audit rows of chapter 1124, the finite defeat rows of chapter 1163, the hidden-material ledger of chapter 1152, and the finite source interface of chapter 1007. Those siblings already expose model audits, failure rows, residue ledgers, and finite observation windows; this packet binds them into one boundary surface for audit exports that cannot claim physical closure unless the hidden residue and defeating observations remain visible.

Definition 1167.1 (Finite-observation residue-boundary carrier)**.** *A* FiniteObservationResidueBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, \Sigma, K, T, L, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source row, read through the finite observation interface and the model-audit source trace.*
2. *Σ is the exposed observation-signature row for the finite audit export.*
3. *K is the classifier row deciding which source and model signatures are compared.*
4. *T is the approximation-tower row that routes source, observation, and model-facing signatures through the same finite packet.*
5. *L is the hidden-residue ledger, including unobserved source content, compression residue, idealization residue, and observer-access limits inherited from the residue-ledger surface.*
6. *F is the failure-surface row naming the finite defeating observations, domain exits, classifier mismatches, and continuation boundaries relevant to the audit export.*
7. *H records componentwise* hsame *transport for $S, \Sigma, K, T, L, F, C, P, N$.*
8. *C records displayed* Cont *replay of source access, signature comparison, tower movement, residue reading, and failure-surface consumption.*
9. *P is the* Pkg *provenance over* FiniteObservationResidueBoundary$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, RealityConstrainedFailureSurface$^{\uparrow}$, OnticResidueLedger$^{\uparrow}$, FiniteObservationInterface$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteObservationResidueBoundary}^{\uparrow}}$ *row.*
The carrier has no coordinate for a closed physical audit packet without L, no defeat-free model truth export without F, no hidden source completion outside S, and no classifier route outside K.

Lean4 (definition) BEDC.Derived.FiniteObservationResidueBoundaryUp.FiniteObservationResidueBoundaryUp
Lean4 ✓ BEDC.Derived.FiniteObservationResidueBoundaryUp.FiniteObservationResidueBoundaryTasteGate single carrier alignment

Theorem 1167.2 (Finite-observation residue boundary NameCert obligations)**.** *For the carrier of Definition 1167.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationResidueBoundary}^{\uparrow}}$ *surface consists exactly of source exposure, signature exposure, classifier exposure, tower exposure, hidden-residue ledger exactness, failure-surface exposure, componentwise* hsame *transport, displayed* Cont *continuation replay,* Pkg *provenance, and local naming through N.*

Lean4 ✓ BEDC.Derived.FiniteObservationResidueBoundaryUp.FiniteObservationResidueBoundaryNameCert obligations

Proof. Project the carrier $R = (S, \Sigma, K, T, L, F, H, C, P, N)$. The first four rows provide the finite source, observation signatures, classifier, and tower needed to read a model-audit export as a bounded packet.

The ledger row L and failure row F supply the two boundary fields. L records hidden residue from source access, compression, idealization, and observer limits; F records the displayed defeating observations, domain exits, mismatches, and continuation boundaries.

The structural rows preserve and replay the same packet. Transport through H is componentwise, replay through C follows the displayed continuation routes, provenance through P records the sibling certificates consumed by the packet, and N supplies the local name.

Since the carrier has no further coordinate, the listed rows exhaust the NameCert surface. A model-audit export with the source, signature, classifier, and tower rows but no residue ledger or failure surface is a projection of R, not the accepted $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$ packet. □

Theorem 1167.3 (Finite-observation residue boundary audit non-closure)**.** *Any model-audit export that passes through the finite source row S, signature row Σ, classifier row K, and tower row T but omits either the hidden-residue ledger L or the failure-surface row F cannot claim a closed physical audit packet through* $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$*.*

Lean4 ✓ BEDC.Derived.FiniteObservationResidueBoundaryUp.FiniteObservationResidueBoundaryAuditNonclosure

Proof. Use the NameCert obligations of Definition 1167.2. The accepted packet exposes L and F as obligations, not as optional annotations on an already closed audit route.

If L is absent, the export has no displayed row for hidden source content, compression residue, idealization residue, or observer-access limits. If F is absent, the export has no displayed row for finite defeats, domain exits, classifier mismatches, or continuation boundaries.

The remaining rows cannot restore the missing boundary. Transport, replay, provenance, and naming preserve the displayed packet; they do not create a hidden-residue ledger or failure surface. Hence omission of L or F leaves the route outside the closed physical audit packet named here. □

Theorem 1167.4 (Finite-observation residue boundary audit handoff)**.** *An accepted* FiniteObservationResidueBoun *packet exports a model-audit handoff only by preserving the finite source row S, observation-signature row Σ, classifier row K, approximation-tower row T, hidden-residue ledger L, failure-surface row F, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

The handoff may be consumed by the model-audit, failure-surface, residue-ledger, and finite-observation siblings named at the start of the chapter, but it has no route that drops either the hidden-residue ledger or the failure-surface row while still claiming a closed physical audit export.

Lean4 ✓ `BEDC.Derived.FiniteObservationResidueBoundaryUp.FiniteObservationResidueBoundaryAuditHandoff`

Proof. First use the NameCert obligations of Definition 1167.2. They list the finite source, signature, classifier, approximation tower, hidden-residue ledger, failure surface, transport, replay, provenance, and local naming rows as the complete local surface of the accepted packet.

Next apply audit non-closure, Definition 1167.3. A model-audit export that retains S, Σ, K, T but omits L or F is not a closed physical audit packet through this boundary, so the audit handoff cannot be formed by a projection that hides residue or defeating observations.

Finally read the sibling roles named in the introduction. $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ supplies the model-audit rows, $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ supplies the finite defeat and continuation-boundary rows, $\mathsf{OnticResidueLedger}^{\uparrow}$ supplies hidden-material residue, and $\mathsf{FiniteObservationInterface}^{\uparrow}$ supplies the finite source window. The accepted packet preserves those roles only through $S, \Sigma, K, T, L, F, H, C, P, N$ so downstream audit exports cite this finite handoff rather than a detached model, residue, or failure-surface route. □

Falsifiable prediction: A model-audit export with no hidden-residue ledger cannot claim a closed physical audit packet.

Independence witness: realityconstrainedmodelaudit, realityconstrainedfailuresurface, onticresiduleledger, finiteobservationinterface: $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$ is not bijective to any listed sibling because it carries the source-signature-classifier-tower rows of the audit together with the hidden-residue ledger, failure surface, componentwise transport, continuation replay, provenance, and local naming in one packet. The model audit lacks the explicit residue-boundary split; the failure surface lacks the source-signature-classifier tower; the residue ledger lacks the finite defeat row; the finite observation interface lacks audit, residue, and failure rows.

Lean follow-up requires `Nontrivial FiniteObservationResidueBoundaryUp` and `FieldFaithful FiniteObservationResidueBoundaryUp` instances in `BEDC.Derived.FiniteObservationResidueBoundaryUp` if this composite packet is later made atomic.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationResidueBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$ packet is generated from a finite source row, observation-signature row, classifier row, approximation tower, hidden-residue ledger, failure surface, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationResidueBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1167.1, Definition 1167.2, Definition 1167.3, and the public handoff of Definition 1167.4 bind the mature paper-side boundary for the carrier, NameCert obligations, audit non-closure, public handoff, falsifiable row, and independence witness.

Lean4 ✓ `BEDC.Derived.FiniteObservationResidueBoundaryUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationResidueBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a closed physical audit, hidden source completion, defeat-free truth export, Lean verification, or bridge checking.

Upgrade path: No further theory-axis upgrade is required; formal verification and bridge checking remain separate verification-axis obligations.

Chapter 1168

A Concrete Naming Certificate for $\mathsf{OpenFitResidueGate}^{\uparrow}$

The $\mathsf{OpenFitResidueGate}^{\uparrow}$packet realizes the open physical fit discipline of section 9414.13 as a finite residue-gate certificate. It sits beside the open physical fit surface of chapter 1131, the finite-observation residue boundary of chapter 1167, the observation-reflection packet of chapter 1126, and the GAP-policy surfaces of chapter 8 and chapter 16. Those siblings expose open fit, visible residue, observation readback, and gap-policy accountability; this gate binds them into the packet a consumer must read before treating finite undefeated fit as physically admissible. As an ai packet, it depends on those open-fit, residue-boundary, observation-reflection, and GAP-policy siblings rather than introducing an independent primitive carrier.

Definition 1168.1 (Open-fit residue-gate carrier)**.** *An* $\mathsf{OpenFitResidueGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, F, R, V, C, H, Q, P, N)$$

with the following displayed rows:

1. *T is the trace-request row, recording the finite observation request and the reality-facing readback route that generated the question.*
2. *F is the finite-fit certificate row, read through the open physical fit and finite observational fit surfaces of section 9414.13.*
3. *R is the open-residue row, recording the unqueried admissible continuations and the hidden residue boundary inherited from chapter 1167.*
4. *V is the refutation-verdict row, recording whether a displayed finite observation defeats F, leaves F undefeated at the current prefix, or refuses any final-fit oracle reading.*
5. *C is the classifier row deciding which observed signature, model signature, residue boundary, and verdict rows are compared.*
6. *H records componentwise* hsame *transport for T, F, R, V, C, Q, P, N.*
7. *Q records displayed* Cont *replay through the observation-reflection route and the open continuation route.*
8. *P is the* Pkg *provenance over* $\mathsf{OpenFitResidueGate}^{\uparrow}$, $\mathrm{OpenPhysicalFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$, $\mathsf{ObservationReflectionPacket}^{\uparrow}$, *GAP-policy rows,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{OpenFitResidueGate}^{\uparrow}}$ *row.*
The carrier has no coordinate for final fit, total model identity, host physical truth, residue erasure, hidden observation closure, or a refutation verdict detached from the finite packet.

Lean4 (definition) `BEDC.Derived.OpenFitResidueGateUp.taste gate`
Lean4 ✓ `BEDC.Derived.OpenFitResidueGateUp.TasteGate.OpenFitResidueGateTasteGate single carrier alignment`

Theorem 1168.2 (Open-fit residue-gate NameCert obligations)**.** *For the carrier of Definition 1168.1, the local* $\mathsf{NameCert}_{\mathsf{OpenFitResidueGate}^{\uparrow}}$ *surface consists exactly of trace-request exposure through T, finite-fit certificate exposure through F, open-residue exposure through R, refutation-verdict exposure through V, classifier exactness through C, componentwise* hsame *transport through H, displayed* Cont *continuation replay through Q,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.OpenFitResidueGateUp.OpenFitResidueGateNameCertObligations`

Proof. Project the carrier $G = (T, F, R, V, C, H, Q, P, N)$. The request, finite fit, residue, verdict, and classifier rows are the complete content-facing surface of the packet.

Read F through Definition 9414.265 and the open physical fit carrier of Definition 1131.1. Finite undefeated fit is therefore a displayed prefix certificate, not a final-fit oracle.

Read R through Definition 1167.1. The residue row keeps unqueried continuations, hidden source content, compression residue, and observation-access limits visible rather than allowing the finite fit row to erase them.

The verdict row V is then bound by the classifier C. It records finite defeat, current undefeated status, or refusal of a final-fit reading as a row of the same packet, so a consumer cannot detach the verdict from the observed/model signature comparison.

Finally the structural rows preserve the same surface. Transport through H, replay through Q, provenance through P, and local naming through N provide no additional coordinate for final fit, host truth, or residue erasure. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 1168.3 (Open-fit residue-gate scoped consumer factorization)**.** *Every admissible* $\mathsf{OpenFitResidueGate}^{\uparrow}$ *consumer must factor through the trace request T, finite-fit row F, open-residue row R, verdict row V, classifier row C, transport H, replay Q, provenance P, and local NameCert row N before treating finite undefeated fit as physically admissible. The factorization exposes finite undefeated fit only as a residue-aware packet read; it does not assert final fit, host physical truth, hidden observation closure, or residue erasure.*

Proof. Project the carrier $G = (T, F, R, V, C, H, Q, P, N)$ from Definition 1168.1. These nine rows are the only admitted route from an open-fit request to a verdict about finite undefeated fit.

Apply Definition 1168.2. The NameCert surface requires trace-request exposure, finite-fit exposure, open-residue exposure, verdict exposure, classifier exactness, transport, replay, provenance, and local naming before any consumer may read the packet as physically admissible at the finite prefix.

The classifier row C binds the finite-fit row to the residue and verdict rows. Therefore a consumer that reads F without also reading R and V has not factored through the displayed gate.

Transport through H, replay through Q, provenance through P, and naming through N preserve the same finite packet. Since none of these rows contains final fit, host physical truth, hidden observation closure, or residue erasure, the scoped consumer factorization cannot claim any of those stronger conclusions. □

Falsifiable prediction: A consumer that treats finite undefeated fit as final fit without the residue verdict row refutes the OpenFitResidueGateUp packet.

Independence witness: open-physical-fit, finite-observation-residue-boundary, observation-reflection-packet, reality-constrained-open-fit, gap-policy: none is bijective because the residue gate binds finite fit, open residue, refutation verdict, classifier, route, provenance, and local naming in one packet.

Later Lean development in `lean4/BEDC/Derived/OpenFitResidueGateUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{OpenFitResidueGate}^{\uparrow}$ and FieldFaithful $\mathsf{OpenFitResidueGate}^{\uparrow}$ instances.

Closure status for $\mathsf{NameCert}_{\mathsf{OpenFitResidueGate}^{\uparrow}}$.

Constructive story: An $\mathsf{OpenFitResidueGate}^{\uparrow}$ packet is generated from trace request, finite-fit certificate, open-residue, verdict, classifier, transport, replay, provenance, and local NameCert rows, as a composite over open physical fit, finite observation residue boundary, observation reflection packet, and reality-constrained open fit surfaces.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpenFitResidueGate}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1168.1, Definition 1168.2, and Definition 1168.3 bind carrier, NameCert obligations, TasteGate alignment, consumer factorization, falsifiable prediction, independence witness, Nontrivial, and FieldFaithful rows into the displayed public residue-gate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpenFitResidueGate}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.OpenFitResidueGateUp.TasteGate.OpenFitResidueGateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final fit, host physical truth, residue erasure, hidden observation closure, or bridge checking.

Upgrade path: Bridge checking requires a standard bridge from the public open-fit residue-gate surface to the acceptance schema.

Chapter 1169

A Concrete Naming Certificate for $\mathsf{FiniteObservationRefutation}^{\uparrow}$

The $\mathsf{FiniteObservationRefutation}^{\uparrow}$ packet realizes the finite-observation open-fit boundary of section 9414.13 as a concrete BEDC NameCert carrier. It sits beside the open physical fit surface of chapter 1131, the reality-constrained open-fit packet of chapter 1132, the open-fit packet of chapter 1137, the open-fit residue gate of chapter 1168, and the physical model audit surface of chapter 1129. Those siblings expose finite fit, open continuation, residue, verdict, and audit rows; this carrier isolates the displayed defeating observation row beside the finite fit row without adding a final-fit oracle.

Definition 1169.1 (FiniteObservationRefutation carrier)**.** *A* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$$

with the following visible rows:

1. *T is the finite trace row through which the observation is read.*
2. *Π is the observation-bundle row naming the finite questions used for the read.*
3. *O is the observed-signature row extracted from T through Π.*
4. *M is the model-signature row generated over the same finite trace and bundle.*
5. *F is the finite-fit row whose acceptance is being tested.*
6. *D is the displayed defeating-observation row, storing the finite observation that makes O and M fail the accepted classifier.*
7. *V is the verdict row separating finite defeat, current undefeatedness, and final-fit refusal.*
8. *L is the ledger row recording the finite fit scope, the defeating observation, and the refusal to promote the prefix into final confirmation.*
9. *E is the open-continuation row recording admissible observation extensions still outside the finite packet.*
10. *H records componentwise* hsame *transport for $T, \Pi, O, M, F, D, V, L, E, C, P, N$.*
11. *C records displayed* Cont *replay through the trace, bundle, signature, fit, defeat, verdict, ledger, and continuation rows.*

12. *P is the* Pkg *provenance over* FiniteObservationRefutation$^{\uparrow}$, OpenPhysicalFit$^{\uparrow}$, RealityConstrainedOpenFit$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, OpenFitResidueGate$^{\uparrow}$, PhysicalModelAudit$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

13. *N is the local* NameCert$_{\text{FiniteObservationRefutation}^{\uparrow}}$ *row.*

The carrier has no coordinate for total model identity, host physical truth, all-prefix confirmation, or a final-fit oracle.

Lean4 ✓ `BEDC.Derived.FiniteObservationRefutationUp.FiniteObservationRefutationTasteGate single carrier alignment`

Theorem 1169.2 (FiniteObservationRefutation NameCert obligations)**.** *For the carrier of Definition 1169.1, the local* NameCert$_{\text{FiniteObservationRefutation}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N),$$

trace and bundle exposure through T, Π, observed and model signature exposure through O, M, finite-fit exposure through F, defeating-observation exposure through D, verdict exposure through V, ledger and open-continuation exposure through L, E, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The tuple $J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$ is the complete finite packet, so carrier admission exposes every row that can be read by a consumer.

Read the observational rows first. The trace and bundle rows T, Π determine the finite questions, while O and M are the observed and model-generated signatures compared by the finite-fit row F.

The defeating observation is stored in D and classified by V. A finite defeat therefore enters the certificate as a visible row of the same packet, not as a hidden host objection or a global claim about all observations.

The ledger and continuation rows separate defeat from openness. The ledger L records the finite scope and the refusal of final confirmation, while E preserves admissible observations not yet queried. The structural rows H, C, P, N transport, replay, authenticate, and name that same packet, so the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationRefutationUp.FiniteObservationRefutationCarrier namecert obligations`

Theorem 1169.3 (FiniteObservationRefutation open-fit consumer)**.** *Every open-fit consumer reading an accepted* FiniteObservationRefutation$^{\uparrow}$ *packet receives the displayed finite route*

$$T, \Pi, O, M, F, D, V, L, E, H, C, P, N.$$

A displayed defeating observation refutes only the finite-fit row F through D and V; it does not supply model-reality identity, host physical truth, total confirmation, all-prefix fit, or a final-fit oracle.

Proof. Start with the carrier in Definition 1169.1. The only rows available to an open-fit consumer are the trace, bundle, signature, finite-fit, defeating-observation, verdict, ledger, continuation, transport, replay, provenance, and naming rows displayed in J.

Apply the NameCert obligation surface of Definition 1169.2. It forces the consumer route to expose D and V together with F, so a defeat is a finite observation against the displayed finite-fit row.

Use the sibling boundaries from Definition 1131.3, Definition 1132.3, Definition 1137.3, Definition 1168.2, and Definition 1129.7. They keep refutation, residue, open continuation, and audit failure surfaces inside finite carrier rows rather than turning them into final physical adequacy.

Since J has no coordinate for final fit or total identity, the consumer receives a finite refutation packet and an open continuation boundary. The defeating observation can invalidate the current finite-fit row, but it cannot close the remaining observation surface or provide a final-fit oracle. □

Theorem 1169.4 (Defeating-observation row exposure)**.** *Every accepted* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *packet exposes D and V as the only finite defeating-observation route for the displayed finite-fit row F. In particular, a consumer of*

$$J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$$

may refute F through the pair D, V, but cannot extract any second defeat row, hidden host verdict, total model identity, or final-fit oracle from the packet.

Proof. Project the carrier of Definition 1169.1. Its visible coordinates contain one defeating-observation row D and one verdict row V, both placed beside the finite-fit row F.

Apply the NameCert obligation surface of Definition 1169.2. It exposes D and V as ordinary carrier rows and lists no additional defeat coordinate.

Apply the open-fit consumer boundary of Definition 1169.3. The consumer can read the finite route through D, V, while the absent coordinates rule out hidden host truth, total identity, all-prefix confirmation, and final-fit output. Thus D, V are the exhaustive finite defeating-observation route for F. □

Theorem 1169.5 (Open-continuation separation)**.** *For every accepted* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *packet, the continuation row E preserves admissible future observations while the pair D, V refutes only the current finite-fit row F. The same packet therefore separates current finite defeat from closure of the remaining observation surface.*

Proof. Read the displayed tuple $J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$. The row E is a continuation coordinate, whereas D and V are the defeating-observation and verdict coordinates attached to F.

Use Definition 1169.2 to keep these rows distinct under local naming. The certificate exposes L as the ledger of finite scope and E as the open-continuation row, so the defeating route through D, V does not consume the continuation surface.

Use Definition 1169.3 to read the packet as an open-fit boundary. The finite defeat invalidates the displayed row F, while the absence of a final-fit oracle and the presence of E leave the admissible continuation rows outside the current packet. □

Theorem 1169.6 (TasteGate witness surface)**.** *The existing Lean TasteGate,* $\mathsf{Nontrivial}$*, and* $\mathsf{FieldFaithful}$ *witnesses for* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *are consumed as the single-carrier witness surface for the displayed tuple*

$$T, \Pi, O, M, F, D, V, L, E, H, C, P, N.$$

They certify that the packet has a nonempty carrier, field-readable coordinates, and one BEDC witness surface over the same carrier shape used by the local NameCert obligations.

Proof. Start from the checked TasteGate marker attached to Definition 1169.1. It names the Lean theorem that aligns the $\mathsf{FiniteObservationRefutation}^{\uparrow}$ carrier with the single tuple shape

$$T, \Pi, O, M, F, D, V, L, E, H, C, P, N.$$

The carrier definition gives the row order, and Definition 1169.2 supplies the local naming duties for those rows. The Lean witness surface supplies nontriviality and field faithfulness for the same carrier rather than for a separate presentation.

Finally apply Definition 1169.4 and Definition 1169.5. They identify the defeat and continuation rows consumed by the TasteGate surface, so the witness certifies exactly the displayed packet rather than an abstract replacement. □

Theorem 1169.7 (FiniteObservationRefutation scope exactness)**.** *Every accepted* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *packet*

$$J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$$

reads trace, bundle, observed signature, model signature, finite fit, displayed defeat, verdict, ledger, continuation, componentwise hsame*, displayed* Cont*,* Pkg *provenance, and local* NameCert *rows only through the displayed BEDC finite-kernel scope. No consumer of the packet obtains a row outside* $T, \Pi, O, M, F, D, V, L, E, H, C, P, N$*.*

Proof. Start with the carrier of Definition 1169.1. It displays the whole accepted packet as the finite tuple $J = (T, \Pi, O, M, F, D, V, L, E, H, C, P, N)$ and states that there is no coordinate for total model identity, host physical truth, all-prefix confirmation, or a final-fit oracle.

Apply Definition 1169.2. The NameCert surface lists carrier admission, trace and bundle exposure, observed and model signature exposure, finite-fit exposure, defeat, verdict, ledger, continuation, componentwise hsame, displayed Cont, Pkg, and local naming through exactly the coordinates of J.

Use Definition 1169.3, Definition 1169.4, and Definition 1169.5 to close the consumer side. They identify the only open-fit route, the only defeating-observation route, and the continuation split inside the same tuple, so the scoped read cannot escape the finite-kernel rows. □

Theorem 1169.8 (FiniteObservationRefutation scoped carrier transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{FiniteObservationRefutation}^{\uparrow}$ *packet preserves the same defeat and continuation split over*

$$D, V, L, E, H, C, P, N$$

without adding host truth, total model identity, all-prefix fit, or a final-fit oracle.

Proof. Project the transport row H from the carrier of Definition 1169.1. It records componentwise hsame transport for the displayed trace, bundle, signature, finite-fit, defeat, verdict, ledger, continuation, replay, provenance, and naming rows.

Apply Definition 1169.2 to keep the transported data inside the same NameCert surface. The row H transports the packet components; it does not create an additional carrier coordinate.

The defeat and continuation split is then inherited from Definition 1169.4 and Definition 1169.5. The defeat remains the finite route through D, V, while E remains the open-continuation row recorded by L.

Finally apply Definition 1169.3. Since the consumer has no access to host truth, total identity, all-prefix fit, or a final-fit oracle before transport, componentwise hsame transport cannot add such a row afterward. □

Theorem 1169.9 (FiniteObservationRefutation scoped open-fit non-escape)**.** *Any open-fit consumer of an accepted* FiniteObservationRefutation$^{\uparrow}$ *packet factors through*

$$D, V, L, E, H, C, P, N.$$

It cannot replace the finite refutation row by total model identity, all-prefix fit, or final confirmation.

Proof. Read the carrier of Definition 1169.1 through the consumer theorem Definition 1169.3. The consumer receives the finite route $T, \Pi, O, M, F, D, V, L, E, H, C, P, N$ and no separate oracle row.

Apply Definition 1169.2. The rows relevant to an open-fit defeat are the displayed defeat, verdict, ledger, continuation, transport, replay, provenance, and naming rows D, V, L, E, H, C, P, N.

Use Definition 1169.4 to identify D, V as the exhaustive finite defeating-observation route for F. Then use Definition 1169.5 to keep E outside the consumed finite defeat as the remaining observation surface.

Thus every open-fit read factors through the displayed scoped rows. Because those rows do not contain total model identity, all-prefix fit, or final confirmation, the consumer cannot replace finite refutation by any of those stronger outputs. □

Falsifiable prediction: A displayed defeating observation refutes only the finite-fit row and never supplies a final-fit oracle.

Independence witness: openphysicalfit, realityconstrainedopenfit, openfitpacket, openfitresiduegate, physicalmodelaudit: none is bijective because the refutation packet alone stores the defeating observation row as a public carrier coordinate beside fit, continuation, ledger, transport, route, provenance, and local NameCert rows.

The Lean TasteGate path `lean4/BEDC/Derived/FiniteObservationRefutationUp/TasteGate.lean` provides explicit Nontrivial FiniteObservationRefutation$^{\uparrow}$ and FieldFaithful FiniteObservationRefutation$^{\uparrow}$ instances for the displayed carrier.

Closure status for NameCert$_{\text{FiniteObservationRefutation}^{\uparrow}}$.

Constructive story: A FiniteObservationRefutation$^{\uparrow}$ packet is generated from a finite trace, observation bundle, observed signature, model signature, finite-fit row, displayed defeating observation, verdict, ledger, open-continuation row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteObservationRefutation}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1169.1, Definition 1169.2, Definition 1169.3, Definition 1169.4, Definition 1169.5, Definition 1169.6, Definition 1169.7, Definition 1169.8, and Definition 1169.9 bind the finite refutation carrier, local NameCert obligations, open-fit consumer boundary, defeating-observation exposure, open-continuation separation, TasteGate witness surface, finite-kernel scope exactness, componentwise transport, and open-fit non-escape.

Lean4 ✓ `BEDC.Derived.FiniteObservationRefutationUp.taste gate`

Formal verification: FormalStatus(NameCert$_{\text{FiniteObservationRefutation}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FiniteObservationRefutationUp.FiniteObservationRefutationTasteGate\_single\_carrier\_al`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, model-reality identity, host physical truth, all-prefix fit, total confirmation, or final-fit oracles.

Upgrade path: Bridge closure requires a standard bridge surface that exports the public finite-refutation route without strengthening it into total model identity, all-prefix fit, or final confirmation.

Chapter 1170

A Concrete Naming Certificate for $\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}$

The $\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}$ packet realizes the idealization branch of the scientific-methodology discipline in chapter 9414, especially section 9414.46. It sits beside the explanation rows of chapter 1136, the methodology-ledger rows of chapter 1145, the ontic-residue rows of chapter 1152, the finite observation boundary of chapter 1167, and the model-audit rows of chapter 1124. Those siblings expose explanation, methodology scope, residue, finite defeat, and model-audit surfaces; this packet binds a safe idealization into one finite seal before a smooth, continuous, limiting, or coarse endpoint can be read as explanatory.

Definition 1170.1 (Scientific idealization residue-seal carrier)**.** *A* $\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, M, O, B, A, I, R, D, F, \Sigma, T, C, P, N)$$

with the following displayed rows:

1. *E is the explanation row inherited from* $\mathsf{RealityConstrainedExplanation}^{\uparrow}$*, naming the phenomenon, model, signature comparison, descent, residue, and failure surface used by the explanation.*
2. *M is the methodology-ledger row inherited from* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$*, naming the effective region, admissible consumers, domain-leakage refusals, and declared extension routes.*
3. *O is the ontic-residue row inherited from* $\mathsf{OnticResidueLedger}^{\uparrow}$*, recording source material not visible at the ideal endpoint.*
4. *B is the finite observation residue-boundary row inherited from* $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$*, recording the finite observation budget and defeat boundary under which residue may become active.*
5. *A is the model-audit row inherited from* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$*, recording source traces, observed signatures, model-generated signatures, classifier comparison, and audit strength.*
6. *I is the idealization arrow row, naming the compression $L_i \rightsquigarrow L_j$ and the ideal endpoint used by the explanation.*
7. *R is the idealization-residue ledger, listing each source feature removed, smoothed, averaged, completed, quotient-read, or limit-read by I.*

8. *D is the descent row, naming the operations on the ideal endpoint whose use is licensed back through the source and residue ledger.*
9. *F is the idealization failure-surface row, naming the refined bundles, tower levels, source conditions, or classifier mismatches that activate residue.*
10. *Σ is the scope row, naming the observation bundle, tower region, admissible endpoint operations, and consumers for which the idealization is safe.*
11. *T records componentwise* hsame *transport for $E, M, O, B, A, I, R, D, F, \Sigma, C, P, N$.*
12. *C records displayed* Cont *replay of the safe-idealization route through explanation, methodology ledger, residue, descent, failure surface, and endpoint scope.*
13. *P records* Pkg *provenance over the sibling carriers,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for treating the ideal endpoint as explanatory when the residue ledger, descent row, failure surface, scope row, transport, route, provenance, or local name is absent.

Lean4 ✓ `BEDC.Derived.ScientificIdealizationResidueSealUp.ScientificIdealizationResidueSealTasteGate`
`single carrier alignment`

Theorem 1170.2 (Scientific idealization residue-seal NameCert obligations)**.** *For the carrier of Definition 1170.1, the local* $\mathsf{NameCert}_{\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}}$ *surface consists exactly of finite explanation accountability, methodology-scope accountability, ontic-residue accountability, finite observation-boundary accountability, model-audit accountability, idealization-arrow exposure, residue-ledger exactness, descent exposure, failure-surface exposure, scope exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ScientificIdealizationResidueSealUp.ScientificIdealizationResidueSealNameCertOblig`

Proof. Project the carrier $S = (E, M, O, B, A, I, R, D, F, \Sigma, T, C, P, N)$. The five sibling rows E, M, O, B, A supply the surrounding explanation, methodology, residue, finite-boundary, and model-audit surfaces consumed by the seal.

The idealization-specific rows I, R, D, F, Σ expose the compression arrow, removed source material, licensed descent, active-residue boundary, and safe scope. A route that omits any one of these rows has only an ideal endpoint or a partial audit read, not the safe-idealization seal.

The structural rows preserve the same displayed packet. Transport through T is componentwise, replay through C follows the displayed continuation route, provenance through P records the sibling certificates and kernel rows, and N supplies the local name.

Since the carrier contains no further coordinate, the listed rows exhaust the NameCert surface. In particular, smoothness, continuity, limiting behavior, or coarse-grained fit cannot substitute for the residue ledger or descent row. □

Theorem 1170.3 (Scientific idealization residue-seal nonescape)**.** *An idealized explanation whose route uses the idealization arrow I but omits the residue ledger R, descent row D, failure-surface row F, or scope row Σ cannot pass the* $\mathsf{ScientificIdealizationResidueSeal}^{\uparrow}$ *safe-idealization seal.*

Proof. Use the NameCert obligations of Definition 1170.2. They require the idealization arrow, residue ledger, descent row, failure surface, and scope row as separate displayed obligations.

If R is absent, the route has no finite account of the source material removed by the idealization. If D is absent, operations performed at the ideal endpoint have no displayed descent back through the compressed source. If F is absent, refined observations have no named surface on which hidden residue can become active. If Σ is absent, the route has no admitted observation bundle, tower region, endpoint operation class, or consumer boundary.

The remaining rows cannot create the missing obligations. Explanation, methodology, residue, boundary, and audit siblings provide surrounding surfaces; transport, replay, provenance, and naming preserve the displayed packet. Hence the route remains outside the accepted safe-idealization seal. □

Theorem 1170.4 (Scientific idealization residue-seal scope refinement)**.** *Let $S = (E, M, O, B, A, I, R, D, F, \Sigma, T, C$ be an accepted* ScientificIdealizationResidueSeal$^{\uparrow}$ *packet. A refined observation bundle may extend the safe-idealization scope only by adding the corresponding residue entries, descent rows, failure-surface rows, and methodology-ledger route to the same displayed packet.*

Lean4 ✓ `BEDC.Derived.ScientificIdealizationResidueSealUp.ScientificIdealizationResidueSeal scope refinement`

Proof. The accepted scope is the displayed row Σ, and the methodology-ledger row M records how exports beyond the present scope are refused or routed.

A refined bundle that keeps the same safe-idealization seal must preserve the residue ledger R and descent row D for the endpoint operations it uses. If the refinement activates source material hidden by I, that material must appear in R and the associated defeat must appear in F.

The extension is therefore not a bare enlargement of Σ. It is admitted only when the packet also exposes the residue, descent, failure, and methodology route rows needed by the refined scope. □

Falsifiable prediction: An idealized explanation whose residue ledger is absent cannot pass the safe-idealization seal.

Independence witness: realityconstrainedexplanation, realityconstrainedmethodologyledger, onticresidueledger, finiteobservationresidueboundary, realityconstrainedmodelaudit: none is bijective to ScientificIdealizationResidueSeal$^{\uparrow}$ because the seal carries idealization-arrow, residue, descent, failure-surface, scope, transport, route, provenance, and local NameCert rows in one packet. The explanation row lacks the methodology export route; the methodology ledger lacks the specific idealization arrow and descent row; the ontic-residue ledger lacks endpoint-operation scope; the finite observation boundary lacks the idealization route; the model audit lacks the safe-idealization residue seal.

Lean follow-up must provide `Nontrivial ScientificIdealizationResidueSealUp` and `FieldFaithful ScientificIdealizationResidueSealUp` instances in `BEDC.Derived.ScientificIdealizationResidueSealUp`.

Closure status for NameCert$_{\text{ScientificIdealizationResidueSeal}^{\uparrow}}$.

Constructive story: A ScientificIdealizationResidueSeal$^{\uparrow}$ packet is generated from explanation, methodology, residue, finite observation boundary, model-audit, idealization-arrow, residue-ledger, descent, failure-surface, scope, transport, route, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ScientificIdealizationResidueSeal}^{\uparrow}}$, seed)

Scope closed: Definition 1170.1, Definition 1170.2, Definition 1170.3, and Definition 1170.4 bind the finite safe-idealization carrier, local NameCert obligations, missing-row refusal, and refined-scope route.

Formal verification: FormalStatus(NameCert$_{\text{ScientificIdealizationResidueSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close endpoint truth, unledgered idealization, bridge checking, smooth-limit validity, or explanatory force outside the displayed residue and descent rows.

Upgrade path: Promotion past seed requires Lean TasteGate nontriviality and field faithfulness for ScientificIdealizationResidueSeal$^{\uparrow}$.

Chapter 1171

A Concrete Naming Certificate for $\mathsf{CauchyTailModulusSeal}^{\uparrow}$

The $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ packet realizes the Real-completeness route of chapter 9238 by joining the shared-tail threshold service of chapter 983, the fusion route of chapter 1024, and the witness-facing extraction route of chapter 652. It keeps the public classifier between the shared threshold and the extractor handoff, so a consumer can read one finite RealUp-facing route without receiving quotient stream equality, host real equality, a selected host limit, or ambient completeness.

The nearest seal siblings are chapter 699 and chapter 386. The tail-agreement seal records agreement across cofinal tail windows, while the Cauchy-limit seal records a terminal limit handoff. The present packet instead binds the shared-tail modulus row, the tail-modulus fusion row, the witness-extractor row, paired StreamName windows, paired RegSeqRat readbacks, the RealUp seal row, and the public classifier that mediates them.

Falsifiable prediction: Every accepted CauchyTailModulusSealUp packet exposes one shared threshold service and admits no consumer route to private per-source thresholds, quotient stream equality, host real equality, selected limits, or ambient completeness.

Independence witness: cauchysharedtailmodulus, cauchytailmodulusfusion, realcompletionwitnessextractor, realtailagreementseal, cauchylimitseal: none is bijective because the seal packet keeps the public classifier between shared-tail threshold service and witness-extractor handoff, whereas the siblings expose only a source modulus, fusion route, extractor route, tail agreement, or limit seal.

Definition 1171.1 (Cauchy tail-modulus seal carrier)**.** *A* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, X, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *row supplying the shared-tail threshold service;*
2. *F is the* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *row fusing the two source moduli through the shared service;*
3. *X is the* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *row that receives the witness-facing handoff;*
4. *τ is the single shared threshold read from the shared-tail and fusion rows;*
5. *D is the dyadic ledger attached to the shared threshold route;*
6. *W_0 and W_1 are the selected* $\mathsf{StreamName}^{\uparrow}$ *windows served by τ;*

7. *Q_0 and Q_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks obtained from the two selected windows;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the shared threshold, dyadic ledger, windows, and readbacks have been displayed;*
9. *H records componentwise* hsame *transport for the source rows, threshold, ledger, windows, readbacks, seal, provenance, and local name;*
10. *C records displayed* Cont *routes from M, F to $\tau, D, W_0, W_1, Q_0, Q_1$, then through X to E;*
11. *P is the* Pkg *provenance over* $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{CauchyTailModulusFusion}^{\uparrow}$, $\mathsf{RealCompletionWitnessExtr}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailModulusSeal}^{\uparrow}}$ *row.*

The carrier contains no private per-source threshold, quotient stream equality, host real equality, selected host limit, independent RealUp seal, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyTailModulusSealUp.CauchyTailModulusSealTasteGate single carrier alignment`

Theorem 1171.2 (Cauchy tail-modulus seal NameCert obligations)**.** *For the carrier of Definition 1171.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailModulusSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for S, shared-tail admission through M, fusion admission through F, witness-extractor admission through X, single-threshold exposure through τ, dyadic ledger exposure through D, selector-window service through W_0, W_1, regular readback through Q_0, Q_1, RealUp seal ordering through E, componentwise stability through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The packet lists the shared-tail row, fusion row, witness-extractor row, shared threshold, dyadic ledger, paired windows, paired readbacks, RealUp seal, transport, route, provenance, and local name as public coordinates.

Use the sibling obligations of chapter 983 to read M as the unique shared-tail threshold service, and use chapter 1024 to force the fusion route through the same τ rather than through two private thresholds.

Next route the witness-facing consumer through chapter 652. The extractor sees D, W_0, W_1, Q_0, Q_1, E only after the same threshold service has fixed the selected windows and regular readbacks.

The rows H, C, P, N transport, replay, package, and name exactly these displayed coordinates. Since the carrier has no private threshold, quotient stream equality, host real equality, selected host limit, independent RealUp seal, or ambient completeness row, these coordinates exhaust the local NameCert surface. □

Theorem 1171.3 (Cauchy tail-modulus seal threshold readback)**.** *Every accepted* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *packet exposes exactly one shared threshold service before any witness-facing or RealUp-facing consumer can read D, W_0, W_1, Q_0, Q_1, E. The threshold readback route is*

$$M, F, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.CauchyTailModulusSealUp.CauchyTailModulusSealThresholdReadback`

Proof. Start from the carrier order. The only threshold coordinate is τ, and both source surfaces M and F precede it in the displayed packet.

Apply the shared-tail and fusion sibling obligations. The $\mathsf{CauchySharedTailModulus}^{\uparrow}$ row supplies the threshold service, while the $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ row attaches both source moduli

to that same service. Thus the packet has one threshold readback before any window or readback row is exposed.

Finally pass through the witness-extractor side. The rows D, W_0, W_1, Q_0, Q_1, E are downstream of τ and are carried by H, C, P, N as one named route, so a witness-facing or RealUp-facing consumer cannot choose a separate threshold. □

Theorem 1171.4 (Cauchy tail-modulus seal non-escape)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumer of an accepted* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *packet factors through* $M, F, X, \tau, D, W_0, W_1$. *The packet exports no private per-source threshold, quotient stream equality, host real equality, selected limit, independent RealUp seal, or ambient completeness principle.*

Proof. Use the NameCert obligations. They route every accepted consumer through the displayed shared-tail row M, fusion row F, witness-extractor row X, shared threshold τ, dyadic ledger D, selected windows, regular readbacks, RealUp seal, transport, continuation, provenance, and local name.

For a StreamName or RegSeqRat consumer, the route stops at W_0, W_1, Q_0, Q_1 after τ and D have already been read. For a RealUp or completion-facing consumer, the route continues through E and the witness-extractor handoff without changing the threshold service.

The excluded outputs require coordinates absent from Definition 1171.1. Since H, C, P, N only transport, replay, package, and name the displayed coordinates, no consumer can leave the finite seal packet. □

Lean4 ✓ `BEDC.Derived.CauchyTailModulusSealUp.CauchyTailModulusSeal nonescape`

Theorem 1171.5 (Cauchy tail-modulus seal independence witness)**.** *The* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *carrier is not bijective, by any row-preserving carrier map, to* $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{CauchyTailModulusFusion}^{\uparrow}$, $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$. *A row-preserving bijection would have to preserve the public classifier between the shared-tail threshold service and the witness-extractor handoff, but none of the listed sibling carriers contains that exact classifier together with* $M, F, X, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$.

Proof. Compare the displayed carrier rows. The shared-tail sibling exposes the source modulus and threshold service but not the fusion row, witness-extractor handoff, and RealUp seal route together. The fusion sibling exposes the fused route but not the public classifier tying it to the extractor handoff as a seal packet.

The witness-extractor sibling exposes the downstream handoff but not the upstream shared-tail threshold service and fusion row. The Real tail-agreement sibling exposes a paired tail-agreement seal, and the Cauchy-limit sibling exposes a terminal limit seal; neither carries the combined shared-tail, fusion, extractor, and RealUp classifier route.

Any row-preserving bijection must preserve the rows and the route order used by Definition 1171.3. Each listed sibling lacks at least one of those rows or replaces the classifier surface by a different one, so no such bijection exists. □

Theorem 1171.6 (Cauchy tail-modulus seal public classifier route)**.** *Every public consumer route for an accepted* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *packet factors through the single classifier route*

$$M, F, X, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N.$$

Here M *is the shared-tail modulus row,* F *is the fusion row,* X *is the witness-extractor row, and* E *is the* $\mathsf{Real}^{\uparrow}$ *seal row. The route exports no private threshold, quotient stream equality, host real equality, selected limit, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyTailModulusSealUp.CauchyTailModulusSeal public classifier route`

Proof. Read the row inventory from Definition 1171.1. The carrier displays the shared-tail row M, fusion row F, witness-extractor row X, threshold τ, dyadic ledger D, selected windows W_0, W_1, readbacks Q_0, Q_1, RealUp seal E, transport H, continuation C, provenance P, and local name N as the complete public packet.

Apply Definition 1171.2. Its NameCert surface exposes exactly the carrier and classifier rows just listed, so a public consumer can enter the seal only through those displayed coordinates.

Next use Definition 1171.3 to fix the unique shared threshold before any window, readback, witness-facing, or RealUp-facing consumer route is available. Then use Definition 1171.4 to rule out exits to private thresholds, quotient stream equality, host real equality, selected limits, and ambient Real completeness.

Finally compare with Definition 1171.5. That theorem separates this route from the sibling seal surfaces, so the displayed classifier is the public route carried by the accepted $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ packet. □

Theorem 1171.7 (Cauchy tail-modulus seal paper bridge)**.** *For every accepted* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *packet*

$$S = (M, F, X, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N),$$

the public classifier route $M, F, X, \tau, D, W_0, W_1, Q_0, Q_1, E, H, C, P, N$ *forms the paper bridge between the shared-tail modulus row, the tail-modulus fusion row, the witness-extractor row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal row. The bridge exports no quotient stream equality, host real equality, selected host limit, or ambient Real completeness principle.*

Proof. Start with the carrier of Definition 1171.1. It displays exactly the shared-tail row M, fusion row F, witness-extractor row X, shared threshold τ, dyadic ledger D, paired windows W_0, W_1, paired readbacks Q_0, Q_1, RealUp seal E, transport H, continuation C, provenance P, and local name N.

Apply Definition 1171.2 to read these rows as the local NameCert surface. Apply Definition 1171.3 to force the single threshold service before the window, readback, witness-facing, and RealUp-facing rows become available.

Next use Definition 1171.4. It rules out exits to private thresholds, quotient stream equality, host real equality, selected limits, independent RealUp seals, and ambient Real completeness.

Finally use Definition 1171.5 and Definition 1171.6. The first separates this packet from the neighboring seal surfaces, and the second identifies the single public classifier route, so the displayed rows give exactly the paper bridge for the $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailModulusSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ packet is generated from a shared-tail modulus row, a tail-modulus fusion row, a witness-extractor row, one shared threshold, a dyadic ledger, paired StreamName windows, paired RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailModulusSeal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1171.1 and the theorem rows of Definition 1171.2, Definition 1171.3, Definition 1171.4, Definition 1171.5, Definition 1171.6, and Definition 1171.7 bind the displayed carrier to BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{CauchyTailModulusFusion}^{\uparrow}$, and $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyTailModulusSealUp.taste gate`

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailModulusSeal}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.CauchyTailModulusSealUp.CauchyTailModulusSealTasteGate\_single\_carrier\_alignment`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close quotient stream equality, host real equality, selected host limits, Real completeness, or bridge checking.

Upgrade path: no further upgrade – chapter at top of formal axis.

Chapter 1172

A Concrete Naming Certificate for $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$

The $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ packet realizes a replayable finite window of inscription events. It sits beside the event carrier of chapter 519, the gap ledger of chapter 520, the inscription point of chapter 610, the audit trace of chapter 682, and the digest inscription packet of chapter 1085. Those siblings provide single-event admission, gap accounting, pointwise commitment, audit trace routing, and digest naming; the buffer seal binds a finite replay window without exporting an unbounded inscription stream.

Definition 1172.1 (Inscription event buffer seal carrier)**.** *An* $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, W, R, D, A, H, Q, P, N)$$

with the following displayed rows:

1. *E is the finite inscription-event row, listing the accepted event records admitted into the buffer;*
2. *W is the finite buffer-window row, naming the bounded replay interval and its visible endpoints;*
3. *R is the replay row, recording the ordered finite readback route through the events in W;*
4. *D is the digest row, binding the replayed window to its digest inscription and digest provenance;*
5. *A is the audit-route row, recording accepted checks, gap residues, and consumer-facing route obligations for the replay;*
6. *H is the componentwise* hsame *transport row for event, window, replay, digest, and audit-route presentations;*
7. *Q is the displayed* Cont *row carrying replay consumers through the same finite window;*
8. *P is the* Pkg *provenance over the event, window, replay, digest, audit, transport, continuation, and local naming rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionEventBufferSeal}^{\uparrow}}$ *row naming exactly this replay buffer seal.*

The carrier has no coordinate for an unbounded inscription stream, a host-level replay oracle, or replay data outside the finite event window.

Lean4 (definition) `BEDC.Derived.InscriptionEventBufferSealUp.InscriptionEventBufferSealUp`

Theorem 1172.2 (Inscription event buffer seal NameCert obligations)**.** *For the carrier of Definition 1172.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionEventBufferSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for* $B = (E, W, R, D, A, H, Q, P, N)$*, finite window exactness, replay row determinacy, digest attachment, audit-route coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.InscriptionEventBufferSealUp.InscriptionEventBufferSealCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1172.1. The accepted packet contains exactly the rows $E, W, R, D, A, H, Q, P, N$.

Window exactness comes from W and replay determinacy from R. The event row E supplies only the finite event records admitted into that window, and the replay row orders only those records.

Digest attachment and audit-route coverage are supplied by D and A. The digest row binds the finite replay to the digest inscription surface, while the audit-route row records accepted checks, gap residues, and consumer-facing obligations.

Finally H, Q, P, N transport, continue, package, and name the same finite buffer seal. Since these rows add no unbounded stream or host replay oracle, the listed obligations exhaust the local certificate surface. □

Theorem 1172.3 (Inscription event buffer seal non-escape)**.** *No accepted* $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ *packet exports an unbounded inscription stream, a host-level replay oracle, or a replay route outside the displayed rows* $E, W, R, D, A, H, Q, P, N$*.*

Lean4 ✓ `BEDC.Derived.InscriptionEventBufferSealUp.InscriptionEventBufferSealCarrier nonescape`

Proof. Inspect the carrier of Definition 1172.1. Its only event-facing row is E, its only window row is W, and its only replay row is R.

Apply the NameCert obligations of Definition 1172.2. Every accepted replay must attach the digest row D, audit-route row A, transport row H, continuation row Q, provenance row P, and local name N to that same finite window.

Thus an unbounded inscription stream, host-level replay oracle, or hidden replay route has no coordinate in the accepted packet. Such a demand can only be rejected at the audit-route boundary, not exported by the buffer seal. □

Falsifiable prediction: No accepted $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ packet exports an unbounded inscription stream or host-level replay oracle beyond the displayed finite event window.

Independence witness: inscriptionevent, inscriptiongapledger, inscriptionpoint, inscriptionauditrace, digestinscription: $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ is not bijective to any listed sibling because it simultaneously carries finite event admission, bounded replay window, replay order, digest attachment, audit-route coverage, componentwise transport, displayed continuation, provenance, and local naming rows.

Lean follow-up must provide `Nontrivial InscriptionEventBufferSealUp` and `FieldFaithful InscriptionEventBufferSealUp` instances before the replay buffer seal is treated as an atomic Lean carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionEventBufferSeal}^{\uparrow}}$ **.**

Constructive story: An $\mathsf{InscriptionEventBufferSeal}^{\uparrow}$ packet is generated from finite inscription events, a bounded replay window, replay readback, digest attachment, audit-route records, componentwise transport, continuation routing, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{InscriptionEventBufferSeal}\uparrow}$, seed)

Scope closed: Definition 1172.1, Definition 1172.2, and Definition 1172.3 bind the finite replay-buffer carrier, its local obligations, and its non-escape boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{InscriptionEventBufferSeal}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert an unbounded inscription stream, a host-level replay oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, replay-window exactness, digest attachment, audit-route coverage, TasteGate nontriviality, and field faithfulness for the replay buffer seal.

Chapter 1173

A Concrete Naming Certificate for CauchyChoiceBoundary$^{\uparrow}$

The CauchyChoiceBoundary$^{\uparrow}$ packet realizes the no-choice side of the Real completeness diagonal route of chapter 9238. It sits after the limit selector of chapter 673, the shared-threshold meet of chapter 909, the seal-budget synchronizer of chapter 886, and the completion-seal classifier of chapter 900. Its role is to bind the explicit finite budget, selected index, tail witness, seal route, refusal row, transport, replay, provenance, and local NameCert surface into one packet without importing a host selected limit, quotient stream equality, countable choice, ambient completeness, or Classical.choice.

Falsifiable prediction: A Real-completion diagonal consumer that omits the selected finite budget row cannot claim the no-choice diagonal seal route.

Independence witness: limitselector, regular-cauchy-diagonal-meet, cauchy-seal-budget-synchronizer, real-completion-seal-classifier, cauchy-witness-ledger: none is bijective to CauchyChoiceBoundary$^{\uparrow}$ because this carrier binds the selected finite index, tail-witness route, no-choice refusal row, seal route, transport, replay, provenance, and local naming in one boundary packet, while the siblings isolate selector admission, meet formation, budget synchronization, seal classification, or witness-ledger exposure.

Definition 1173.1 (Cauchy choice-boundary carrier)**.** *A* CauchyChoiceBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, E, I, T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus row for the accepted regular Cauchy input, read through the explicit Real-completion diagonal route;*
2. *E is the finite tolerance-budget row whose bound determines the diagonal scale being served;*
3. *I is the selected finite index row produced from M and E, not from a global selector or countable-choice principle;*
4. *T is the tail-witness route row certifying that all downstream observations are read on the displayed finite tail determined by I;*
5. *S is the Real seal route row reached through the same finite budget and tail-witness data;*

6. *R is the no-choice refusal row, recording that the packet exports no selected limit object, no quotient stream equality, no ambient completeness row, no countable choice row, and no* Classical.choice *row;*
7. *H stores componentwise* hsame *transport for M, E, I, T, S, R and for the local packet shape;*
8. *C stores the displayed* Cont *replay from modulus and tolerance admission to selected index, tail witness, Real seal, refusal, and local naming;*
9. *P is the* Pkg *provenance over chapter 673, chapter 909, chapter 886, chapter 900, chapter 902,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host-side real equality, quotient regular streams, an unbounded selector, ambient metric completeness, countable choice, or Classical.choice.

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundaryTasteGate single carrier alignment`

Theorem 1173.2 (Cauchy choice-boundary diagonal-budget factorization)**.** *Every accepted Real-completion diagonal consumer of a* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ *packet*

$$B = (M, E, I, T, S, R, H, C, P, N)$$

factors through the displayed route

$$M, E, I, T, S, R, H, C, P, N.$$

In particular, the consumer reads the explicit modulus before the tolerance budget, the selected finite index before the tail witness, the tail witness before the Real seal route, and the refusal row before any no-choice claim is exported.

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundaryDiagonalBudgetFactorization`

Proof. Project the carrier of Definition 1173.1. Its only modulus, budget, selected-index, tail-witness, seal, and refusal coordinates are M, E, I, T, S, R.

The selected-index row I is displayed as a finite consequence of M and E. Hence any diagonal consumer that reaches T or S has already passed through the modulus and tolerance rows rather than through a hidden selector.

The structural rows H, C, P, N preserve and replay this same order. Componentwise hsame transport carries only the displayed rows, Cont replay follows the listed route, provenance names the sibling certificates, and local naming exposes the packet as the same finite boundary surface.

Thus every accepted diagonal consumer factors through $M, E, I, T, S, R, H, C, P, N$ before it can claim the Real seal route. □

Theorem 1173.3 (Cauchy choice-boundary no-choice non-escape)**.** *No selected limit, quotient stream equality, ambient completeness row, countable choice row, or* Classical.choice *row can be read from an accepted* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ *packet. The packet exports only the modulus row, tolerance budget, selected finite index, tail-witness route, Real seal route, no-choice refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row listed in Definition 1173.1.*

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundary no choice nonescape`

Proof. Inspect the accepted carrier. The row R explicitly records refusal of selected limits, quotient stream equality, ambient completeness, countable choice, and Classical.choice, while the carrier itself contains no additional coordinate from which such data could be reconstructed.

Apply Definition 1173.2. Any accepted consumer must pass through the finite route $M, E, I, T, S, R, H, C, P, N$.

The finite selected-index row I is not a selected limit object; it is only the index admitted by the displayed modulus and tolerance budget. The tail row T and seal row S are likewise route witnesses, not quotient equalities or ambient completeness principles.

Therefore the only exported data are the ten displayed packet rows, and every attempted read of a selected limit, quotient stream equality, ambient completeness, countable choice, or Classical.choice leaves the accepted boundary packet. □

Theorem 1173.4 (Cauchy choice-boundary NameCert obligation surface)**.** *The carrier of Definition 1173.1, the diagonal-budget factorization of Definition 1173.2, and the no-choice boundary of Definition 1173.3 assemble the five NameCert obligations for* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$*: the carrier row, classifier-stable visible route, diagonal-budget exactness, refusal of selected-limit or quotient exits, and local ledger provenance. Concretely, every accepted packet*

$$B = (M, E, I, T, S, R, H, C, P, N)$$

exports exactly those five obligation rows through its ten displayed carrier fields.

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundaryNameCertObligationSurface`

Proof. Project the ten fields of the carrier. The rows $M, E, I, T, S, R, H, C, P, N$ give the carrier row, and the structural rows H and C restrict comparison and replay to the visible route already listed in the definition.

The route exactness is supplied by Definition 1173.2: any accepted consumer reaches the seal row only through the displayed modulus, tolerance budget, selected index, tail witness, and refusal row.

The negative boundary is supplied by Definition 1173.3. It blocks selected-limit data, quotient stream equality, ambient completeness, countable choice, and Classical.choice as exits from the packet.

It remains to identify the local ledger. The provenance row P records the sibling certificates and kernel primitives used by the construction, while N is the local $\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}^{\uparrow}}$ row. These projections give the carrier, visible-route, exactness, refusal, and provenance obligations without adding any global Cauchy selector or host-side completeness principle. □

Theorem 1173.5 (Cauchy choice-boundary scoped consumer route)**.** *Every accepted Real-completion diagonal consumer of*

$$B = (M, E, I, T, S, R, H, C, P, N)$$

factors through the five obligation rows of Definition 1173.4: the carrier row, classifier-stable visible route, diagonal-budget exactness, no-choice refusal, and local ledger provenance. The scoped route is read as a finite BHist *consumer whose componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}^{\uparrow}}$ *row pass through the sibling dependencies* $\mathsf{LimitSelector}^{\uparrow}$*,* $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$*,* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$*,* RealCompletionSealClassifierUp*, and* $\mathsf{CauchyWitnessLedger}^{\uparrow}$*.*

Consequently the consumer may use only the displayed modulus, tolerance budget, selected finite index, tail-witness route, Real seal route, refusal row, transport, replay, provenance, and local naming. It preserves the exclusions on selected limits, quotient stream equality, ambient completeness, countable choice, and Classical.choice*.*

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundaryScopedConsumerRoute`

Proof. Project the carrier definition. The packet supplies exactly the rows $M, E, I, T, S, R, H, C, P, N$, with BHist carrying the finite history, hsame transporting the displayed components, Cont replaying the route, Pkg recording provenance, and $\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}^{\uparrow}}$ naming the local surface.

Apply Definition 1173.2 to fix the consumer order. The route reaches the selected finite index only after the modulus and tolerance budget, reaches the tail witness and seal route only through that selected index, and reaches the refusal row before any no-choice boundary is claimed.

Apply Definition 1173.3. Any attempted exit to a selected limit, quotient stream equality, ambient completeness row, countable choice row, or Classical.choice lies outside the accepted boundary packet.

Finally apply Definition 1173.4. Its five rows package the carrier, classifier-stable visible route, diagonal-budget exactness, no-choice refusal, and local ledger provenance, with the sibling dependencies $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, RealCompletionSealClassifierU and $\mathsf{CauchyWitnessLedger}^{\uparrow}$ recorded through P and exposed through N. □

Theorem 1173.6 (Cauchy choice-boundary public consumer seal)**.** *Every public Real-completion consumer of an accepted* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ *packet*

$$B = (M, E, I, T, S, R, H, C, P, N)$$

must read the same ten rows $M, E, I, T, S, R, H, C, P, N$ through the five obligation rows of Definition 1173.4 and the scoped route of Definition 1173.5 before reaching any Real seal claim. Consequently the public interface exports the finite no-choice boundary and excludes selected limits, quotient stream equality, ambient completeness, countable choice, and Classical.choice*.*

Proof. Project the carrier definition. The accepted packet has only the displayed rows $M, E, I, T, S, R, H, C, P, N$, so a public consumer has no additional coordinate from which to read a selected limit, a quotient equality, an ambient completeness witness, countable choice, or Classical.choice.

Apply Definition 1173.2 to fix the public row order. The route reads the modulus and tolerance budget before the selected finite index, the selected index before the tail witness and Real seal route, and the refusal row before any no-choice claim is exported.

Apply Definition 1173.3 to block the excluded exits. The public consumer can only keep the displayed no-choice boundary rows; every attempted selected-limit, quotient-stream, ambient-completeness, countable-choice, or Classical.choice read leaves the accepted packet.

Apply Definition 1173.4. Its five obligation rows package the carrier row, classifier-stable visible route, diagonal-budget exactness, no-choice refusal, and local ledger provenance, so the public interface is already mediated by the finite NameCert surface.

Finally apply Definition 1173.5. The sibling dependencies $\mathsf{LimitSelector}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalMeet}^{\uparrow}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, RealCompletionSealClassifierUp, and $\mathsf{CauchyWitnessLedger}^{\uparrow}$ are bound through the scoped route before any Real seal claim is made, which seals the public no-choice consumer surface. □

The Lean follow-up for this chapter is the namespace `BEDC.Derived.CauchyChoiceBoundaryUp`. Its `TasteGate.lean` surface must provide both a `Nontrivial CauchyChoiceBoundaryUp` instance and a `FieldFaithful CauchyChoiceBoundaryUp` instance for the carrier of Definition 1173.1.

Theorem 1173.7 (Cauchy choice-boundary bridged Real seal)**.** *Every accepted* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ *public consumer whose rows factor through the displayed diagonal budget, selected finite index, tail-witness route, and Real seal route is a bridged Real-completion seal consumer. The bridge exports*

only the rows already named by Definition 1173.6, Definition 1173.5, and Definition 1173.3. It exports no selected limit, quotient stream, ambient completeness, countable choice, host equality, or Classical.choice *coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundaryUp StdBridge`

Proof. First apply Definition 1173.5. The accepted public consumer is forced through the finite carrier row, classifier-stable visible route, diagonal-budget exactness, no-choice refusal, and local ledger provenance before it reaches the Real seal route.

Next apply Definition 1173.6. The public interface reads the same ten rows $M, E, I, T, S, R, H, C, P, N$ and reaches the seal only through the displayed budget, selected finite index, tail witness, refusal, transport, replay, provenance, and local naming surface.

Finally apply Definition 1173.3. The accepted boundary packet has no coordinate for a selected limit, quotient stream, ambient completeness, countable choice, host equality, or Classical.choice, so the Real-completion seal consumer is bridged only by the already displayed no-choice rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}\uparrow}$.

Constructive story: The Cauchy choice boundary is generated by starting with the explicit diagonal-budget rows recorded in $BHist$ and refusing to add a selector row for an arbitrary Cauchy subsequence. The carrier reads a Cauchy handoff only when it is already factored through the displayed budget ledger. The $hsame$ relation transports that boundary across equivalent histories, while Cont only extends the visible budget route and never licenses a new choice functional. Thus the chapter's object is a bottom-up refusal surface: it certifies the route from finite diagonal evidence to a Cauchy boundary and records where countable-choice-style exits leave the carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}\uparrow}$, maturePackageClosed)

Scope closed: This chapter binds the Cauchy choice-boundary carrier, diagonal-budget factorization theorem, no-choice non-escape theorem, NameCert obligation-surface rows, scoped consumer route, public consumer seal, bridged Real-seal theorem Definition 1173.7, mature worked consumers over the public no-choice Real-completion seal, and TasteGate instance `BEDC.Derived.CauchyChoiceBoundaryUp.taste_gate`.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyChoiceBoundary}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.CauchyChoiceBoundaryUp.CauchyChoiceBoundary\_no\_choice\_nonescape`

Bridge status: bridgeChecked

Origin: ai

Not claimed: It does not claim a global Cauchy selector, choice over all tails, or host-level equality of Cauchy representatives.

Upgrade path: No further upgrade – chapter at top of formal axis.

Chapter 1174

A Concrete Naming Certificate for $\mathsf{CofinalWindowRealSeal}^{\uparrow}$

The $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ packet realizes the finite-window and cofinal-tail Real-completion route of chapter 9238. It sits after the StreamName surface of chapter 126, the dyadic ledger of chapter 291, the RegSeqRat readback surface of chapter 301, the Real seal of chapter 38, the tail schedule of chapter 681, the finite observation-budget selector of chapter 782, the exact boundary of chapter 1120, and the cofinal diagonal seal of chapter 1041. The carrier names their shared terminal meet; it is not a quotient stream, host real equality, selected limit, hidden selector, countable-choice row, or ambient completeness principle.

Falsifiable prediction: If a Real-facing consumer reaches the terminal seal without the displayed cofinal window, dyadic ledger, and RegSeqRat readback rows, the packet is refuted.

Independence witness: finiteobservationbudgetselector, tailcofinalityschedule, realcompletionexactboundary, cofinaldiagonalseal: each sibling supplies one route face, while this packet names their cofinal-window Real-seal meet.

Definition 1174.1 (Cofinal-window Real-seal carrier)**.** *A* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, T, W, D, R, E, X, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *selector-budget row that bounds the visible requests;*
2. *T is the accepted* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *row that turns those requests into a cofinal finite-window schedule;*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row consumed by the selector, schedule, and terminal seal route;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and dyadic readback ledger attached to W;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same window and dyadic ledger;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row read only after B, T, W, D, R have been displayed;*
7. *X is the* $\mathsf{RealCompletionExactBoundaryUp}$ *boundary row confirming that the four Real-completion faces meet at this finite route;*

8. *G is the* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ *row confirming that the cofinal-tail route and diagonal Real seal route share the same terminal surface;*
9. *H records componentwise* hsame *transports for all displayed rows;*
10. *C records the displayed* Cont *replay from selector budget and tail schedule through window, dyadic ledger, readback, Real seal, exact boundary, and cofinal diagonal seal;*
11. *P is the* Pkg *provenance over the cited sibling packets,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalWindowRealSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient streams, host real equality, selected limits, hidden selectors, countable choice, a detached Real seal, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSealTasteGate single carrier alignment`

Theorem 1174.2 (Cofinal-window Real-seal NameCert obligations)**.** *For every accepted* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *carrier*

$$S = (B, T, W, D, R, E, X, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CofinalWindowRealSeal}^{\uparrow}}$ *surface consists exactly of selector-budget exposure through B, tail-cofinality exposure through T, finite-window exposure through W, dyadic-ledger exposure through D, RegSeqRat readback through R, terminal Real-seal exposure through E, exact-boundary exposure through X, cofinal-diagonal-seal exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1174.1. Its displayed inventory lists the selector budget, tail schedule, finite window, dyadic ledger, RegSeqRat readback, Real seal, exact boundary, cofinal diagonal seal, and structural support rows.

Read the route in dependency order. The selector row B bounds the requests, the tail schedule row T supplies the cofinal finite-window service, and the rows W, D, R, E form the StreamName, DyadicRatCore, RegSeqRat, and RealUp terminal route.

The boundary rows X and G identify that route respectively with the Real-completion exact boundary and the cofinal diagonal seal. The rows H, C, P, N only transport, replay, package, and name the same finite packet.

Since quotient streams, host real equality, selected limits, hidden selectors, countable choice, detached Real seals, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1174.3 (Cofinal-window Real-seal terminal meet)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$B, T, W, D, R, E, X, G, H, C, P, N.$$

The route is the common meet of the finite observation-budget selector, the cofinal tail schedule, the exact Real-completion boundary, and the cofinal diagonal seal. No consumer may replace it by a quotient stream, host real equality, selected limit, hidden selector, countable-choice row, detached Real-seal row, or ambient completeness coordinate.

Proof. Start from Definition 1174.2. It exposes the complete route inventory for the accepted packet and gives no second source for budget, schedule, window, dyadic, readback, seal, boundary, or diagonal-seal data.

Attach the selector and schedule faces. The $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ row B fixes the bounded request surface, while the $\mathsf{TailCofinalitySchedule}^{\uparrow}$ row T fixes the cofinal window service before W, D, R, E can be consumed.

Attach the terminal faces. The $\mathsf{RealCompletionExactBoundaryUp}$ row X keeps the StreamName, DyadicRatCore, RegSeqRat, and RealUp faces inside the exact boundary, and the $\mathsf{CofinalDiagonalSeal}^{\uparrow}$ row G keeps the same terminal seal on the cofinal diagonal route.

Finally inspect the excluded objects. None occurs in the carrier or in the local classifier, and the structural rows H, C, P, N preserve only the displayed packet. Therefore every accepted Real-facing or completion-facing consumer reaches the terminal seal through the stated meet. □

Theorem 1174.4 (Cofinal-window Real-seal source exhaustion)**.** *Every accepted Real-facing or completion-facing consumer of a* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *carrier can project only the rows*

$$B, T, W, D, R, E, X, G, H, C, P, N$$

of Definition 1174.1. A terminal $\mathsf{Real}^{\uparrow}$ *seal is available only after the selector-budget row* B*, the tail-schedule row* T*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the exact-boundary row* X*, and the cofinal diagonal row* G *have been displayed on the same route, with* H, C, P, N *carrying the componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support.*

Thus the terminal seal cannot be reached through a quotient stream, host real equality, selected host limit, hidden selector, countable-choice row, detached Real seal, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSeal source exhaustion`

Proof. Project the carrier in Definition 1174.1. The packet has exactly the selector-budget, tail-schedule, window, dyadic-ledger, RegSeqRat readback, terminal Real-seal, exact-boundary, cofinal-diagonal, transport, continuation, package, and local naming rows.

Invoke Definition 1174.2. It identifies those rows as the whole local NameCert surface and gives no additional source for budget, schedule, window, dyadic, readback, boundary, diagonal, or seal data.

Use Definition 1174.3 for non-escape. Since every Real-facing or completion-facing consumer reaches the terminal meet through the displayed route, a consumer that tries to bypass B, T, W, D, R, X, G before reading E has left the accepted packet. □

Theorem 1174.5 (Cofinal-window Real-seal obligation scope)**.** *Every* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *NameCert obligation is scoped only over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the sibling packets* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$*,* $\mathsf{TailCofinalitySchedule}^{\uparrow}$*,* $\mathsf{RealCompletionExactBoundaryUp}$*, and* $\mathsf{CofinalDiagonalSeal}^{\uparrow}$*. No obligation row may quantify over a quotient stream, host real equality, selected host limit, hidden selector, countable-choice source, detached Real-seal coordinate, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSealObligationScope`

Proof. Project the carrier of Definition 1174.1. Its rows are exactly $B, T, W, D, R, E, X, G, H, C, P, N$, where the first eight rows name the displayed sibling surfaces and the last four rows give hsame, Cont, Pkg, and local NameCert support over the same finite history.

Apply Definition 1174.2 to read the NameCert surface from those rows. Apply Definition 1174.3 to force every Real-facing consumer through the same route.

Finally apply Definition 1174.4. Since the checked source-exhaustion target admits no further source coordinate, every obligation remains inside the displayed BEDC primitive and sibling dependency scope. □

Theorem 1174.6 (Cofinal-window Real-seal public certificate)**.** *The public* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *export is exactly the finite route*

$$B, T, W, D, R, E, X, G, H, C, P, N$$

and its local $\mathsf{NameCert}_{\mathsf{CofinalWindowRealSeal}^{\uparrow}}$ *row. It exports no quotient stream, host equality, selected limit, completeness coordinate, hidden selector, countable-choice row, or detached Real seal.*

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSealPublicCertificate`

Proof. Begin with Definition 1174.2. It identifies the local NameCert row with the displayed finite route and gives no second certificate component.

Use Definition 1174.3 and Definition 1174.4 to rule out consumer-side escape from that route. The terminal Real seal can be read only after the selector, schedule, window, dyadic ledger, RegSeqRat readback, exact boundary, and cofinal diagonal rows are present.

It remains to bind the export boundary. Definition 1174.5 says every obligation row lives inside the same BEDC primitive and sibling dependency scope, so the public certificate is precisely the finite route together with its local NameCert. □

Theorem 1174.7 (Cofinal-window Real-seal bridge boundary)**.** *Any standard bridge that later presents* $\mathsf{CofinalWindowRealSeal}^{\uparrow}$ *to a Real-facing consumer must factor through the checked source-exhaustion route and through the public certificate of Definition 1174.6. A consumer row that bypasses either boundary is not a bridge for this packet.*

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSealUp StdBridge`

Proof. Use the checked source-exhaustion marker for Definition 1174.4. It fixes the only rows a Real-facing or completion-facing consumer may project from the accepted carrier.

Apply Definition 1174.6. The public export contains the finite route $B, T, W, D, R, E, X, G, H, C, P, N$ and the local NameCert row, and it excludes the quotient, host-equality, selected-limit, and completeness coordinates.

Therefore a standard bridge can only expose a Real-facing consumer row after passing through both boundaries. Any map that omits the checked source-exhaustion route or the displayed public certificate has left the named packet. □

Lean development for this route must provide `Nontrivial CofinalWindowRealSealUp` and `FieldFaithful CofinalWindowRealSealUp` in `lean4/BEDC/Derived/CofinalWindowRealSealUp/TasteGate.le`

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalWindowRealSeal}^{\uparrow}}$ **.**

Constructive story: The cofinal-window Real seal is generated from a $BHist$ packet that records selector-budget rows, tail schedules, finite windows, dyadic ledgers, regular-rational readbacks, and the terminal Real-seal handoff. The carrier convention first exposes the window ledger, then requires every later Real-facing row to be connected by Cont to that same cofinal route. The $hsame$ relation supplies representative stability: if the history is renamed, the terminal meet and source-exhaustion rows still read back the same window evidence. No quotient stream or selected limit is introduced; the seal is the visible route by which finite windows become the only admissible Real consumer surface.

Lean4 ✓ `BEDC.Derived.CofinalWindowRealSealUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalWindowRealSeal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1174.5 binds every obligation row to the StreamName, RegSeqRat, DyadicRatCore, Realseal, BHist, hsame, Cont, Pkg, and NameCert dependency scope.

Formal verification: FormalStatus(NameCert$_{\text{CofinalWindowRealSeal}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.CofinalWindowRealSealUp.CofinalWindowRealSeal\_source\_exhaustion`

Bridge status: none

Origin: ai

Not claimed: It does not claim ambient real completeness, quotient-stream equality, or a selected limit beyond the cofinal window route.

Upgrade path: Public closure requires a public certificate surface beyond the scoped cofinal-window Real-seal dependency route.

Chapter 1175

A Concrete Naming Certificate for $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$

The $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ packet is the finite selector surface behind the diagonal-limit construction of chapter 9238. It refines the witness schedule of chapter 593 by displaying the explicit modulus request and selected finite window, uses the modulus witness ledger of chapter 977 for tolerance and readback ordering, and lands at the exact completion boundary of chapter 1120. Its role is to make the selector packet visible before any diagonal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Falsifiable prediction: Any diagonal-limit route that reads a $\mathsf{Real}^{\uparrow}$ seal without the selector's explicit modulus request, selected window, dyadic tolerance, and regular-rational readback rows refutes the $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ packet.

Independence witness: regularcauchywitnessschedule, regular-cauchy-modulus-witness-ledger, real-completion-exact-boundary, streamname, regseqrat: none is bijective to the selector packet because each sibling supplies respectively a schedule row, a ledger row, an exact boundary, a window interface, or a regular-rational readback, while this carrier binds all five into one seal-facing selector surface.

Definition 1175.1 (Regular Cauchy witness selector packet)**.** *A* $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the explicit modulus-request row for the regular Cauchy input and the requested diagonal precision;*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row served by that modulus request;*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested rational bound and the selected diagonal denominator data;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected member and selected diagonal index;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, W, T, R have been displayed;*
6. *H records componentwise* hsame *transports for the modulus request, window, tolerance, readback, seal, provenance, and local name;*

7. *C records the displayed* Cont *replay from the modulus request through the window, tolerance, readback, and seal rows;*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWitnessSchedule}^{\uparrow}$, $\mathsf{RegularCauchyModulusWitnessLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelector}^{\uparrow}}$ *row.*

The classifier compares carriers only through these displayed rows and their componentwise transports. It has no coordinate for a host-selected limit, quotient equality of regular Cauchy names, an unrecorded stream read, countable choice, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessSelectorUp.RegularCauchyWitnessSelectorTasteGate single carrier alignment`

Theorem 1175.2 (Regular Cauchy witness selector source totality)**.** *For every accepted* RegularCauchyWitnessSelec *packet*

$$Q = (M, W, T, R, E, H, C, P, N),$$

the diagonal source selected for a requested precision is exactly the finite route

$$M \longmapsto W \longmapsto T \longmapsto R.$$

The route exposes the explicit modulus request, the selected $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is available.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessSelectorUp.RegularCauchyWitnessSelector source totality`

Proof. Project the carrier of Definition 1175.1. The rows M, W, T, R are the only source-facing rows before the terminal seal coordinate E.

Read the route in dependency order. The selected window W is served by the modulus request M, the tolerance row T records the dyadic precision attached to that request, and the readback row R is the regular-rational observation at the selected member and selected diagonal index.

The structural rows do not add another source. The hsame rows in H transport the four displayed coordinates, the Cont rows in C replay the same route, and P, N record provenance and local naming. Hence the selected source is total exactly on the displayed route $M \mapsto W \mapsto T \mapsto R$. □

Theorem 1175.3 (Regular Cauchy witness selector classifier stability)**.** *If two accepted* RegularCauchyWitnessSelect *packets have componentwise* hsame *transports on their modulus-request, selected-window, dyadic-tolerance, regular-rational-readback, and terminal-seal rows, then their local classifiers preserve the same diagonal selector route. Transporting the packet cannot replace the selected window, alter the tolerance row, change the readback row, or reach a different* $\mathsf{Real}^{\uparrow}$ *seal outside the displayed* Cont *replay.*

Proof. Begin with the componentwise classifier data recorded in H. It compares the modulus requests, windows, tolerance rows, readbacks, and terminal seals row by row.

Apply source totality from Definition 1175.2. Since the source route is exactly $M \mapsto W \mapsto T \mapsto R$, transporting those four rows transports the whole selected source.

Now inspect the continuation rows. The Cont replay in C reaches the seal row E only from the transported selected source, and the package row P records the same dependency surface. Thus the local classifier preserves the route and has no coordinate on which to substitute another selector, tolerance, readback, or seal. □

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessSelectorUp.RegularCauchyWitnessSelector classifier stability`

Theorem 1175.4 (Regular Cauchy witness selector ledger non-escape)**.** *No diagonal-limit consumer accepted by a* $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ *packet can read the terminal* $\mathsf{Real}^{\uparrow}$ *seal without first displaying the rows*

$$M, W, T, R, H, C, P, N.$$

In particular, a route that reaches a $\mathsf{Real}^{\uparrow}$ *seal without the explicit modulus request, selected window, dyadic tolerance, regular-rational readback, componentwise transport, continuation replay, provenance, and local* NameCert *rows is outside the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyWitnessSelectorUp.RegularCauchyWitnessSelector ledger nonescape`

Proof. Project an accepted packet $Q = (M, W, T, R, E, H, C, P, N)$. The seal row E is not a detached Real object; it is the terminal coordinate of the same finite selector carrier.

Use Definition 1175.2. A diagonal source read must pass through M, W, T, R before any terminal seal is available.

Use Definition 1175.3. The rows H and C preserve and replay exactly that displayed source, while P and N record provenance and local naming for the same packet.

The excluded route would need a coordinate for an unrequested modulus, unselected window, hidden tolerance, detached readback, quotient equality, host limit, countable choice, or ambient completeness. No such coordinate occurs in the carrier, so the seal read cannot escape the displayed ledger. □

Theorem 1175.5 (Regular Cauchy witness selector NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelecto}}$ *surface of an accepted packet* $Q = (M, W, T, R, E, H, C, P, N)$ *consists exactly of carrier exposure for* Q*, source totality through* M, W, T, R*, classifier stability through* H, C*, terminal seal non-escape through* E*, package provenance through* P*, and local naming through* N*. This surface is the finite diagonal-selector packet consumed by the Real completeness route and exports no host-selected limit, quotient of Cauchy names, countable-choice selector, or ambient completeness row.*

Proof. Carrier exposure is Definition 1175.1. It lists the modulus request, selected window, tolerance, readback, terminal seal, transport, continuation, package, and local naming rows.

Apply Definition 1175.2 to identify the selected diagonal source as M, W, T, R. Apply Definition 1175.3 to keep that source stable under local classifier transport.

Apply Definition 1175.4 for the terminal $\mathsf{Real}^{\uparrow}$ boundary. The seal row is reachable only through the same displayed selector ledger and its structural rows.

These rows exhaust the NameCert surface. Since every listed dependency is displayed inside Q, and since the carrier contains no host limit, quotient, choice, or ambient-completeness coordinate, the packet is exactly the finite diagonal selector consumed by the Real completeness route. □

Theorem 1175.6 (Regular Cauchy witness selector completion-boundary route)**.** *For every accepted* $\mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ *packet*

$$Q = (M, W, T, R, E, H, C, P, N),$$

any completion-facing consumer that uses the selector to reach a $\mathsf{Real}^{\uparrow}$ *seal factors through the explicit modulus request* M*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal seal row* E*, and the structural rows* H, C, P, N*, and then lands through the exact boundary of chapter 1120. The route has no coordinate for quotient Cauchy-name equality, host-selected limits, countable choice, or ambient completeness.*

Proof. Begin with the selector's source route. By Definition 1175.2, the only source-facing path available to an accepted packet is $M \mapsto W \mapsto T \mapsto R$ before the terminal seal row can be read.

Apply Definition 1175.4. Any consumer of the terminal $\mathsf{Real}^{\uparrow}$ row must also display H, C, P, N, so the selected rows are not detached observations but the finite ledger carried by Q.

Now use Definition 1175.5. The local $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelector}^{\uparrow}}$ surface exports exactly the carrier, source, classifier, terminal, provenance, and naming rows listed in the statement. Thus a completion-facing consumer has no selector-side route other than the displayed packet.

The landing side is the exact-boundary route of Definition 1120.13 in chapter 1120. That route consumes the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$, and structural rows as a finite boundary packet. Together with Definition 1120.14, it refuses quotient Cauchy-name equality, host-selected limits, countable choice, and ambient completeness as boundary coordinates. □

Lean development for this route must provide $\mathsf{Nontrivial}\ \mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RegularCauchyWitnessSelector}^{\uparrow}$ instances in `BEDC.Derived.RegularCauchyWitnessSelectorUp.Tast`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelector}^{\uparrow}}$.

Constructive story: The regular Cauchy witness selector is generated from a $BHist$ packet whose source rows already contain a regular Cauchy witness, its classifier, and the ledger entry that records how the witness is consumed. The carrier convention is selective but not choice-based: it reads the witness from the named packet, transports it along $hsame$, and allows Cont only to extend the same regular route. Source totality says every admitted packet exposes the needed witness row; classifier stability says representative changes do not alter the selector; ledger non-escape says no external selection principle is smuggled into the NameCert surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the regular Cauchy witness-selector packet, source-totality, classifier-stability, ledger-non-escape, and NameCert-surface theorems.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyWitnessSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim countable choice, arbitrary subsequence selection, or host-level equality of Cauchy representatives.

Upgrade path: Reaching obligationClosure requires formal packet rows for the regular witness and checked $hsame$ stability of the selector classifier.

Chapter 1176

A Concrete Naming Certificate for $\mathsf{CompressionDescentAudit}^{\uparrow}$

The $\mathsf{CompressionDescentAudit}^{\uparrow}$ packet realizes endpoint-operation descent as a finite audit object rather than as a claim read from an endpoint alone. It composes the tower-compression carrier of chapter 1142, the tower-equivalence surface of chapter 1143, the composite-ledger classifier of chapter 1162, and the physical model audit of chapter 1129. Those chapters name the tower, endpoint, ledger, equivalence, and model-facing audit rows; this packet names the finite decision surface that accepts or refuses an endpoint operation by displayed descent.

Definition 1176.1 (CompressionDescentAudit carrier)**.** *A* $\mathsf{CompressionDescentAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, E, O, D, L, F, H, P, N)$$

with the following displayed rows:

1. *T is the tower-route row, naming the accepted approximation or compression route whose endpoint is being queried.*
2. *E is the endpoint row, naming the visible terminal package on which the operation is stated.*
3. *O is the endpoint-operation row, naming the operation, predicate, or comparison proposed at E.*
4. *D is the descent-witness row, recording the displayed route by which O survives the tower classifiers back to the licensed source-facing level.*
5. *L is the composite-ledger row, importing every hidden-source, classifier, residue, and endpoint dependency required by the descent witness.*
6. *F is the failure-surface row, naming every displayed reason for refusing descent, including missing ledger material, failed classifier compatibility, endpoint-only equality, and model-audit defeat.*
7. *H records componentwise* hsame *transport for T, E, O, D, L, F.*
8. *P is the* Pkg *provenance over* $\mathsf{CompressionDescentAudit}^{\uparrow}$*, the cited tower-compression, tower-equivalence, ledger-functor, and physical-audit packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompressionDescentAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for endpoint truth as source truth, no private descent oracle, no unledgered tower collapse, and no physical acceptance route outside the displayed failure surface and model-audit provenance.

Lean4 (definition) `BEDC.Derived.CompressionDescentAuditUp.TasteGate.CompressionDescentAuditUp`

Theorem 1176.2 (CompressionDescentAudit route soundness)**.** *Let*

$$A = (T, E, O, D, L, F, H, P, N)$$

be an accepted $\mathsf{CompressionDescentAudit}^{\uparrow}$ *packet. If the packet accepts the endpoint operation* O*, then* O *is read only through the displayed descent witness* D*, the composite ledger* L*, the failure surface* F*, and the transported tower route* T, E, H*. Endpoint truth at* E *alone is not a source-facing or model-facing acceptance claim.*

Lean4 ✓ `BEDC.Derived.CompressionDescentAuditUp.TasteGate.CompressionDescentAuditRouteSoundness`

Proof. Project the carrier of Definition 1176.1. The endpoint row E and operation row O are only two coordinates of a nine-row finite packet, and neither coordinate contains a source-facing acceptance route by itself.

The descent witness D records the route by which the operation survives the tower classifiers. The composite ledger L records the hidden material and classifier dependencies that the witness consumes. The failure surface F records the finite reasons for refusing the route.

Therefore an accepted read factors through D, L, F and through the componentwise transport H of the same tower route. A consumer that keeps E and O while dropping those rows has an endpoint statement, not a $\mathsf{CompressionDescentAudit}^{\uparrow}$ acceptance. □

Theorem 1176.3 (CompressionDescentAudit ledger exhaustion)**.** *For the carrier of Definition 1176.1, every accepted descent decision is exhausted by the tower route* T*, endpoint* E*, operation* O*, descent witness* D*, composite ledger* L*, failure surface* F*, transports* H*, provenance* P*, and local name* N*. No later consumer may add a hidden source distinction, private endpoint equivalence, or detached model-audit reason without entering it through one of these rows.*

Lean4 ✓ `BEDC.Derived.CompressionDescentAuditUp.TasteGate.CompressionDescentAuditLedgerExhaustion`

Proof. The displayed tuple lists all coordinates admitted by the local NameCert. The tower route, endpoint, operation, descent witness, ledger, failure surface, transport, provenance, and name are therefore the complete finite surface available to the audit.

The ledger-functor component contributes only through L, and the physical model-audit component contributes only through the failure and provenance rows F, P. Tower equivalence contributes endpoint and ledger compatibility only as transported through the same packet.

Thus every accepted or refused descent decision is already represented by the displayed rows. Adding another hidden source distinction, endpoint equivalence, or model-audit reason changes the packet rather than reading the accepted carrier. □

Theorem 1176.4 (CompressionDescentAudit non-escape)**.** *An endpoint operation refused by the failure surface* F *of an accepted* $\mathsf{CompressionDescentAudit}^{\uparrow}$ *packet cannot be exported as a tower-valid operation by appealing to endpoint agreement, tower equivalence, composite-ledger functoriality, or physical model audit outside the same packet.*

Lean4 ✓ `BEDC.Derived.CompressionDescentAuditUp.TasteGate.CompressionDescentAuditNonEscape`

Proof. Begin with the route-soundness theorem. Definition 1176.2 says that acceptance factors through D, L, F, H with the same tower route and endpoint operation.

Endpoint agreement is not enough, because the tower-equivalence surface also requires ledger and descent compatibility. Composite-ledger functoriality is not enough, because it preserves ledger routes but does not by itself supply a descent witness for O.

The physical model audit is likewise not a bypass. Its role in this packet is visible through F and P, so a model-facing reason that is not carried by those rows is not a reason of the accepted audit. A refusal recorded in F therefore blocks export until the packet itself displays a repaired descent witness and ledger route. □

Falsifiable prediction: An accepted CompressionDescentAuditUp packet is refuted by any endpoint operation exported as tower-valid while its descent witness is absent or its failure surface records a classifier, ledger, endpoint-equivalence, or model-audit defeat.

Closure status for $\mathsf{NameCert}_{\mathsf{CompressionDescentAudit}\uparrow}$.

Constructive story: A $\mathsf{CompressionDescentAudit}^{\uparrow}$ packet is generated from tower route, endpoint, operation, descent witness, composite ledger, failure surface, componentwise transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompressionDescentAudit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1176.1, Definition 1176.2, Definition 1176.3, and Definition 1176.4 bind endpoint-operation descent to a finite tower-route, composite-ledger, failure-surface, transport, provenance, and NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompressionDescentAudit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert Lean verification, bridge checking, source truth from endpoint truth, private descent, unledgered tower collapse, or physical acceptance outside the displayed audit packet.

Upgrade path: Obligation closure requires Lean encoding of carrier admission, descent route soundness, ledger exhaustion, refusal non-escape, and field faithfulness for $T, E, O, D, L, F, H, P, N$.

Chapter 1177

A Concrete Naming Certificate for $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$

The $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ packet realizes the assembly view of chapter 9375 as a finite composite NameCert surface. It sits beside chapter 733, chapter 732, chapter 738, chapter 740, and chapter 739: those siblings expose the positive normal-form exclusion and the subject-reduction boundary in separate packets, while the present chapter binds their rows into one composite naming certificate.

Definition 1177.1 (Closed-consistency assembly carrier)**.** *A* $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, T, E, F, S, O, D, R, P, N)$$

with the following displayed rows:

1. *L is the closedness row for the inspected MetaCIC proof candidate;*
2. *T is the empty-context typing row for the same proof candidate and its displayed false type;*
3. *E is the typed normal endpoint row, including beta-strong normality and the endpoint false-type read;*
4. *F is the normal false-proof exclusion row consumed at the endpoint;*
5. *S is the subject-reduction boundary row separating the arbitrary closed source from the typed-normal endpoint;*
6. *O is the obstruction row recording the dependent-codomain reduction frontier;*
7. *D is the assembly ledger row joining closedness, endpoint, exclusion, subject-reduction boundary, and obstruction entries;*
8. *R records displayed* Cont *routes for every consumer read of the assembly packet;*
9. *P is the* Pkg *provenance over the closed-normal consistency boundary, normal false exclusion, reduction-boundary, normal-form seal,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}$ *row.*

The carrier contains no coordinate for full closed consistency, no hidden subject-reduction theorem, no endpoint transport outside S, and no host-level proof search channel.

Lean4 (definition) `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblyUp`

Definition 1177.2 (Closed-consistency assembly classifier)**.** *Two* $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ *carriers*

$$A = (L, T, E, F, S, O, D, R, P, N) \quad \textit{and} \quad A' = (L', T', E', F', S', O', D', R', P', N')$$

are classified by the componentwise hsame *comparison of the displayed rows together with route replay through* R, R' *and provenance comparison through* P, P'*. The classifier is undefined on any attempted comparison that omits* S *or* O*, because the assembly route is a boundary certificate rather than a proof of closed subject reduction.*

Lean4 (definition) `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblyClassifier`
Lean4 ✓ `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblyClassifier route obligations`

Theorem 1177.3 (Closed-consistency assembly NameCert obligations)**.** *For the carrier of Definition 1177.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}$ *surface consists exactly of carrier admission for* A*, classifier exactness for the rows* $L, T, E, F, S, O, D, R, P, N$*, closedness readback, typed-normal endpoint readback, normal false-proof exclusion, subject-reduction boundary exposure, obstruction-ledger exposure, displayed continuation replay, provenance accounting, and local naming.*

Lean4 ✓ `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblyUp namecert obligations`

Proof. Carrier admission is the finite row list in Definition 1177.1. The classifier of Definition 1177.2 compares only those displayed rows, so no unlisted proof object or host equality can enter the NameCert surface.

The positive consistency read is inherited as rows, not as an ambient theorem. Closedness and typing are read through L, T; the typed normal endpoint is read through E; and the normal false-proof exclusion is read through F, matching the closed-normal packets in chapter 739 and chapter 740.

The boundary rows are equally part of the certificate. The subject-reduction row S and obstruction row O record the frontier named in chapter 738. Since D, R, P, N only ledger, replay, package, and locally name the same surface, the local NameCert obligations are exhausted by the stated rows. □

Theorem 1177.4 (Closed-consistency assembly boundary nonescape)**.** *Every accepted* $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ *carrier keeps full closed consistency outside the exported route. A consumer may read closedness, empty-context typing, the typed-normal endpoint, normal false-proof exclusion, the subject-reduction boundary, obstruction ledger, replay routes, provenance, and local naming, but it cannot export a closed subject-reduction theorem or an arbitrary closed proof contradiction detached from* S *and* O*.*

Lean4 ✓ `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssembly boundary nonescape`

Proof. Project the carrier $A = (L, T, E, F, S, O, D, R, P, N)$. The positive rows L, T, E, F give the closed-normal endpoint route, while S and O remain displayed boundary rows rather than discharged theorems.

The continuation row R only replays the displayed rows, and the package row P records provenance over the sibling packets already cited by the chapter. Neither row contains a coordinate for full closed consistency or for an independent subject-reduction theorem.

Thus every consumer route remains inside the finite assembly packet. Detaching the contradiction read from S and O would require an unlisted row, so the boundary cannot escape through this NameCert surface. □

Theorem 1177.5 (Closed-consistency assembly sibling routing)**.** *The* $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ *packet routes every public read through the sibling surfaces named in its provenance row: closed-normal boundary data, normal false-proof exclusion, closed-consistency reduction boundary, normal-form consistency seal, and closed-normal consistency packet data. No public read bypasses those siblings by appealing directly to an ambient MetaCIC consistency principle.*

Proof. Read the provenance row P. It names the sibling packet sources for the positive normal-form route and for the reduction boundary route, so the assembly consumer obtains a composite route rather than a new primitive consistency principle.

The closed-normal and normal-form siblings supply L, T, E, F as finite rows. The reduction-boundary sibling supplies the displayed S, O frontier. The ledger and continuation rows D, R join these routes without identifying the frontier with a discharged subject-reduction theorem.

Therefore sibling routing is exhaustive for public reads of the assembly carrier. Any route that skips the sibling packet rows would have no carrier coordinate in A and no local $\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}$ admission. □

Lean4 ✓ `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblySiblingRouting`

Theorem 1177.6 (Closed-consistency assembly obligation handoff)**.** *The carrier of Definition 1177.1, the classifier of Definition 1177.2, the NameCert obligations of Definition 1177.3, the boundary nonescape theorem of Definition 1177.4, and the sibling routing theorem of Definition 1177.5 are exactly the obligation-closure surface for* $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$*. Any downstream handoff must therefore read the closedness, empty-context typing, endpoint, exclusion, boundary, obstruction, ledger, replay, provenance, and local naming rows through those displayed obligations, without importing full closed consistency or closed subject reduction as an ambient theorem.*

Proof. Project the carrier and classifier. They expose the finite row list $L, T, E, F, S, O, D, R, P, N$ and compare only those rows by componentwise transport, replay, and provenance.

Apply the NameCert obligation, boundary nonescape, and sibling routing theorems. The NameCert theorem identifies the local fields, the nonescape theorem blocks a detached consistency conclusion, and sibling routing confines every public read to the closed-normal, normal-false-exclusion, reduction-boundary, normal-form-seal, and closed-normal-packet sources.

Thus the public handoff surface is exhausted by the existing carrier, classifier, NameCert, boundary, and routing rows. No further paper obligation is exposed before a checked TasteGate marker and formal bridge evidence are available. □

Lean4 ✓ `BEDC.Derived.ClosedConsistencyAssemblyUp.ClosedConsistencyAssemblyObligationHandoff`

Theorem 1177.7 (Closed-consistency assembly scope ledger)**.** *The displayed rows*

$$A = (L, T, E, F, S, O, D, R, P, N)$$

scope every accepted $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ *read over* BHist, hsame, Cont, $\mathsf{NameCert}$, *the cited sibling packets, and the retained obstruction boundary. The scope includes closedness, empty-context typing, typed-normal endpoint readback, normal false-proof exclusion, subject-reduction boundary exposure, obstruction-ledger exposure, assembly ledgering, replay, provenance, and local naming, and it excludes full MetaCIC consistency, closed subject reduction, arbitrary closed proof contradiction, and endpoint transport outside* S *and* O*.*

Proof. Project the carrier of Definition 1177.1. Its rows $L, T, E, F, S, O, D, R, P, N$ are the only coordinates available to a public consumer, and the classifier of Definition 1177.2 compares only those coordinates through componentwise hsame, displayed Cont replay, and provenance.

Apply Definition 1177.3, Definition 1177.4, and Definition 1177.5. These theorems bind the positive endpoint rows, retain the subject-reduction and obstruction boundary rows, and route public reads through the named sibling packets.

Therefore the scope ledger is exactly the displayed assembly packet. A route that exports full consistency, a closed subject-reduction theorem, a detached contradiction, or endpoint transport outside S and O has no carrier coordinate and no local $\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}$ admission. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedConsistencyAssembly}^{\uparrow}$ packet is generated from closedness, empty-context typing, typed-normal endpoint, normal false-proof exclusion, subject-reduction boundary, obstruction, assembly ledger, continuation replay, package provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1177.7 scopes the carrier, classifier, NameCert obligations, boundary nonescape, and sibling routing rows over BHist, hsame, Cont, NameCert, the cited sibling packets, and the retained obstruction boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyAssembly}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close full MetaCIC consistency, closed subject reduction, arbitrary closed proof contradiction, endpoint transport outside the subject-reduction boundary, or Lean verification.

Upgrade path: Obligation closure requires checked carrier, classifier, boundary, sibling-routing, and local naming obligations for the closed-consistency assembly packet.

Chapter 1178

A Concrete Naming Certificate for RealityConstrainedFinalSynthesisLedger$^{\uparrow}$

The RealityConstrainedFinalSynthesisLedger$^{\uparrow}$ packet realizes the final-synthesis route of section 9414.68 as a finite BEDC NameCert carrier. It consumes the truth-certificate surface of chapter 1141, the reality-constrained truth packet of chapter 1144, the methodology-ledger route of chapter 1145, the failure-surface row of chapter 1163, and the model-selection boundary of chapter 1165. The methodology export row is consumed through chapter 1145 before final truth, open-failure, model-selection, or observer-invariant synthesis rows are read. Those siblings separately expose truth fields, methodology export discipline, open defeat, comparative model selection, observer-invariant reading, and local naming; this carrier binds them into the finite packet by which final synthesis is admitted without asserting final model–reality identity.

Definition 1178.1 (Reality-constrained final-synthesis-ledger carrier)**.** *A* RealityConstrainedFinalSynthesisLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, R, M, F, B, O, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the physical-truth-certificate row recording open fit, tower descent, observer invariance, ledger data, and failure surface material from* PhysicalTruthCertificate$^{\uparrow}$*;*
2. *R is the reality-constrained TruthCert row recording source, signature, classifier, tower, stability, descent, invariant, ledger, failure-surface, and local-name data from* RealityConstrainedTruthCert$^{\uparrow}$*;*
3. *M is the methodology-ledger export row recording explanation scope, audit, objectivity, tower compression, open fit, domain-leakage refusal, upgrade route, and methodology failure surface from* RealityConstrainedMethodologyLedger$^{\uparrow}$*;*
4. *F is the open-fit failure row recording defeating observations, domain exits, classifier mismatches, continuation scope, ledger entries, transport, replay, provenance, and local naming from* RealityConstrainedFailureSurface$^{\uparrow}$*;*
5. *B is the model-selection boundary row recording the comparative fit, finite defeat, and no-final-oracle boundary inherited from* ModelSelectionBoundary$^{\uparrow}$*;*
6. *O is the observer-invariance row recording that the public reading of the final-synthesis export is invariant across admissible observer histories;*

7. *S is the synthesis-scope row naming the finite observation domain, tower endpoint, admissible continuation class, and public consumer route for the export;*
8. *H records componentwise* hsame *transport for the truth, methodology, failure, model-selection, observer-invariance, and scope rows;*
9. *C records displayed* Cont *replay of the final-synthesis export through T, R, M, F, B, O, S, H;*
10. *P records* Pkg *provenance over the consumed sibling packets,* BHist*,* hsame*,* Cont*,* Pkg*, and local naming rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded confirmation, final model–reality identity, unledgered methodology authority, hidden observer privilege, erased defeat, model-selection outside the displayed boundary, or truth export outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.RealityConstrainedFinalSynthesisLedger carrier namecert surface`

Theorem 1178.2 (Reality-constrained final-synthesis-ledger obligations)**.** *For the carrier of Definition 1178.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}}$ *surface consists exactly of carrier admission, TruthCert preservation, methodology-ledger export preservation, open-fit failure exposure, model-selection-boundary visibility, observer-invariant public reading, synthesis-scope exactness, componentwise transport, displayed continuation replay, provenance, and local naming through N.*

Proof. Project the carrier $Z = (T, R, M, F, B, O, S, H, C, P, N)$. The first seven rows expose the truth-certificate packet, the reality-constrained TruthCert packet, the methodology-ledger export, the open failure row, the model-selection boundary, the observer-invariance row, and the finite synthesis scope.

TruthCert preservation follows through T and R. These rows retain open fit, tower descent, source, signature, classifier, stability, observer-invariant reading, ledger data, failure surface, and local naming before the final-synthesis export can be read.

Methodology and failure exposure follow through M and F. The methodology row records export discipline, domain-leakage refusal, upgrade routes, and methodology defeats, while the open-fit failure row keeps defeating observations, domain exits, classifier mismatches, continuation scope, ledger entries, replay, and provenance visible.

The model-selection and public-reading obligations follow through B, O, S. The boundary row blocks a no-final-oracle selection from being replaced by final confirmation, the observer row keeps public reading invariant across admissible histories, and the scope row fixes the observation domain, endpoint, continuation class, and consumer route.

The structural rows H, C, P, N then transport, replay, package, and name exactly the displayed rows. Since the carrier has no further coordinate, every accepted final-synthesis export is exhausted by this NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.TasteGate.RealityConstrainedFinalSynthesi single carrier alignment`

Theorem 1178.3 (Reality-constrained final-synthesis-ledger model-selection handoff)**.** *For every accepted* $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$ *carrier $Z = (T, R, M, F, B, O, S, H, C, P, N)$, a final-synthesis export must read the model-selection boundary row B before the observer-invariant*

public reading row O and the synthesis-scope row S are consumed. The truth-certificate row T, reality-constrained TruthCert row R, methodology-ledger row M, and open-fit failure row F cannot bypass B, erase finite defeat, or turn the export into final model–reality identity.

Lean4 ✓ `BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.RealityConstrainedFinalSynthesisLedger model selection handoff`

Proof. Project the carrier $Z = (T, R, M, F, B, O, S, H, C, P, N)$. The model-selection boundary row B is a displayed coordinate of the same packet as the truth, methodology, failure, observer, and scope rows.

Apply Definition 1178.2. The local NameCert surface requires model-selection-boundary visibility together with observer-invariant public reading and synthesis-scope exactness, so the public rows O and S are not consumed apart from B.

Finally use the sibling-independence theorem Definition 1178.4. The final-synthesis ledger is not merely the truth, methodology, failure, or model-selection sibling in isolation; it binds the rows in one carrier. Therefore T, R, M, F cannot bypass the boundary row, and the finite defeat recorded by B remains visible whenever the export is read. □

Theorem 1178.4 (Reality-constrained final-synthesis-ledger sibling independence)**.** *The* RealityConstrainedFinalSy *carrier is not bijective to* PhysicalTruthCertificate$^{\uparrow}$*,* RealityConstrainedTruthCert$^{\uparrow}$*,* RealityConstrainedMethodologyLed RealityConstrainedFailureSurface$^{\uparrow}$*, or* ModelSelectionBoundary$^{\uparrow}$*. The final-synthesis ledger uniquely binds truth fields, methodology export, failure exposure, model-selection boundary, observer invariance, synthesis scope, transport, replay, provenance, and local naming in one finite carrier.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.RealityConstrainedFinalSynthesisLedger sibling independence`

Proof. Compare the displayed carrier fields. PhysicalTruthCertificate$^{\uparrow}$ and RealityConstrainedTruthCert$^{\uparrow}$ supply truth-certificate rows, but they do not by themselves bind the methodology-ledger export row, model-selection boundary, and final synthesis scope.

RealityConstrainedMethodologyLedger$^{\uparrow}$ supplies methodology export discipline, domain-leakage refusal, and upgrade rows, but it is not the combined truth, open-failure, model-selection, and observer-invariant final-synthesis carrier.

RealityConstrainedFailureSurface$^{\uparrow}$ supplies the defeat and open-fit failure rows, while ModelSelectionBoundary$^{\uparrow}$ supplies comparative fit and no-final-oracle selection. Neither carries the full truth-certificate and methodology-ledger export rows together with final public naming.

Therefore no listed sibling is bijective to Z. Each omits at least one displayed field required by Definition 1178.1, and the local NameCert row accepts only the joined packet. □

Theorem 1178.5 (Reality-constrained final-synthesis obligation closure route)**.** *For the accepted* RealityConstrainedFinalSynthesisLedger$^{\uparrow}$ *carrier*

$$Z = (T, R, M, F, B, O, S, H, C, P, N),$$

the carrier definition, local NameCert obligations, model-selection handoff, sibling-independence witness, falsifiable prediction, independence witness, and the TasteGate marker

`BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.TasteGate.RealityConstrainedFinalSynthes`

exhaust the obligation-level surface for the final-synthesis ledger. The closure route reads only the displayed truth-certificate, methodology-ledger, open-failure, model-selection, observer-invariance,

synthesis-scope, transport, replay, provenance, and local naming rows; it does not add final model–reality identity, unbounded confirmation, hidden observer privilege, or erased defeat.

Proof. Project the carrier Z. The displayed rows T, R, M, F, B, O, S are the only truth, methodology, failure, model-selection, public-reading, and synthesis-scope data that the local certificate can inspect.

Apply Definition 1178.2. It lists carrier admission, sibling preservation, model-selection visibility, observer-invariant reading, synthesis-scope exactness, transport, replay, provenance, and local naming as the local NameCert surface.

Use Definition 1178.3 and Definition 1178.4. The first theorem forces the public export through the model-selection boundary, and the second prevents the ledger from collapsing to any single sibling packet.

The falsifiable prediction and independence witness state the observable refusal boundary, while the TasteGate marker records the finite carrier-alignment check on the Lean side. Together these rows give the obligation-closure route without adding a final oracle, unbounded confirmation, hidden observer privilege, or erased defeat. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedFinalSynthesisLedgerUp.RealityConstrainedFinalSynthesis obligation route closure`

Falsifiable prediction: A final-synthesis export whose carrier omits either the open-fit failure row or the methodology-ledger export row is not accepted by the local NameCert.

Independence witness: physicaltruthcertificate, realityconstrainedtruthcert, realityconstrained-methodologyledger, realityconstrainedfailuresurface, modelselectionboundary: no sibling is bijective because the final-synthesis ledger uniquely binds truth fields, methodology export, failure exposure, model-selection boundary, observer invariance, and local naming in one carrier.

The Lean follow-up must provide `Nontrivial RealityConstrainedFinalSynthesisLedgerUp` and `FieldFaithful RealityConstrainedFinalSynthesisLedgerUp` instances in `lean4/BEDC/Derived/Reality`

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$ packet is generated from displayed truth-certificate, reality-constrained TruthCert, methodology-ledger, open-failure, model-selection-boundary, observer-invariance, synthesis-scope, componentwise transport, continuation, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The bridge-facing surface is the route of Definition 1178.5 together with the model-selection handoff, sibling-independence witness, falsifiable prediction, independence witness, and existing TasteGate marker as the exported final-synthesis ledger boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim final model–reality identity, unbounded confirmation, hidden observer privilege, erased defeat, Lean verification, or bridge checking.

Upgrade path: No further paper-side theory closure remains; Lean verification and bridge checking remain independent formal-axis work.

Chapter 1179

A Concrete Naming Certificate for $\mathsf{SliceCategory}^{\uparrow}$

The classical slice category over a fixed base object is the mathematical precedent for $\mathsf{SliceCategory}^{\uparrow}$: its objects are arrows into the base, and its morphisms are triangles over the base. In BEDC this reading is finite and displayed. The base object is a unary-continuation history, every displayed object row is a BHist row, every arrow-to-base row is a $\mathsf{CategoryHomCarrier}$ entry, and every triangle witness is a replayable Cont and hsame ledger rather than an external categorical diagram.

This chapter turns the kernel-as-category reading of chapter 9363 into a concrete fiber packet. It depends on the kernel-category carrier of chapter 488, the unary-continuation category packet of chapter 471, the generic category certificate of chapter 61, and the categorical consumers represented by chapter 109. The packet supplies a base-fixed surface that later functorial and representable constructions can read without importing a host slice category.

Definition 1179.1 (Slice category NameCert carrier)**.** *A* $\mathsf{SliceCategory}^{\uparrow}$ *carrier over a fixed unary-continuation base object is a finite* BHist *packet*

$$S = (B, X, A, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the fixed base unary object row, admitted through the checked unary-continuation* $\mathsf{Cat}^{\uparrow}$ *packet;*
2. *X is the displayed object row of the fiber;*
3. *A is the arrow-to-base row, recorded as a* $\mathsf{CategoryHomCarrier}(X, B, A)$ *entry;*
4. *F records the fiber condition, namely that the displayed object is accepted only together with its arrow into the same base row B;*
5. *H records componentwise* hsame *transport preserving both B and the arrow-to-base row A;*
6. *C records the* Cont *replay rows used by morphisms in the fiber and by the commutative-triangle readback to the fixed base;*
7. *P is the* Pkg *provenance over* BHist, $\mathsf{Cat}^{\uparrow}$, $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{CategoryHomCarrier}$, hsame, Cont, *and the local certificate rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SliceCategory}^{\uparrow}}$ *row.*

The classifier for accepted packets compares only the displayed rows B, X, A, F, H, C, P, N. It has no coordinate for a host slice object, quotient arrow class, external triangle equality, or a second base object.

Lean4 (definition) `BEDC.Derived.SliceCategoryUp.SliceCategoryCarrier`
Lean4 ✓ `BEDC.Derived.SliceCategoryUp.SliceCategoryTasteGate single carrier alignment`

Theorem 1179.2 (Slice-category kernel fiber obligation)**.** *Every accepted* $\mathsf{SliceCategory}^{\uparrow}$ *packet keeps the base object fixed and reads arrow data only through the displayed* $\mathsf{CategoryHomCarrier}$ *row of its own fiber packet. In particular, a public read of $S = (B, X, A, F, H, C, P, N)$ may expose B, X, the arrow-to-base row $A : X \to B$, componentwise* hsame *transport preserving B and A, and replayable* Cont *routes, but it cannot change the base object or inspect an arrow outside the displayed row A.*

Lean4 ✓ `BEDC.Derived.SliceCategoryUp.SliceCategoryCarrier kernel fiber obligation`

Proof. Project the accepted carrier of Definition 1179.1. The base coordinate is the single row B, and the fiber condition F admits X only together with the arrow-to-base row A into that same B.

The arrow read is therefore forced through $\mathsf{CategoryHomCarrier}(X, B, A)$. The obligations inherited from the kernel-as-category chapter identify this row with unary endpoints, a displayed continuation morphism, and hsame transport of the displayed tail; no additional arrow table appears in S.

Now apply the componentwise transport row H. Its source and target clauses preserve B and the displayed arrow-to-base row A, so transport can move the representation of the same fiber object but cannot replace the base coordinate. The Cont replay row C can replay only the routes already named by A and by the fiber condition F.

Finally inspect the package and naming rows P and N. They name only BHist, $\mathsf{Cat}^{\uparrow}$, $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{CategoryHomCarrier}$, hsame, Cont, and local certificate data. Since neither an alternate base object nor an external arrow store is a coordinate of the packet, no public read can project one. □

Theorem 1179.3 (Slice-category NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{SliceCategory}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the packet $S = (B, X, A, F, H, C, P, N)$;*
2. *base fixation, requiring every accepted object and morphism read to preserve the same unary-continuation base row B;*
3. *arrow-to-base classification, requiring every displayed object row X to be paired with the row* $\mathsf{CategoryHomCarrier}(X, B, A)$*;*
4. *stability and transport, requiring componentwise* hsame *transport of X and A while preserving B and replaying only the named* Cont *routes;*
5. *ledger non-escape, requiring all public reads to pass through* BHist, $\mathsf{Cat}^{\uparrow}$, $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{CategoryHomCarrier}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Thus $\mathsf{SliceCategory}^{\uparrow}$ *is a concrete BEDC fiber over the unary-continuation category, not a primitive host category construction.*

Lean4 ✓ `BEDC.Derived.SliceCategoryUp.SliceCategoryCarrier namecert obligation surface`

Proof. Carrier admission is tuple projection from Definition 1179.1. It exposes exactly B, X, A, F, H, C, P, N.

Base fixation is the single-base clause of F. Since every object row enters the packet as a pair (X, A) over B, the accepted fiber has no operation that changes the codomain of the displayed arrow.

Arrow-to-base classification is the $\mathsf{CategoryHomCarrier}(X, B, A)$ row. The kernel-as-category obligations supply its endpoint, continuation, and displayed-tail laws; the slice packet only restricts their codomain endpoint to the fixed base row.

Stability follows by applying H and replaying C. These rows transport X and A componentwise and replay the named continuation routes, while the base coordinate remains the same B.

Ledger non-escape follows from P and N. The package provenance names only BEDC history, category, hom-carrier, transport, continuation, package, and local naming rows, so a host slice category, quotient arrow equality, or unlisted triangle witness is outside the certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SliceCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{SliceCategory}^{\uparrow}$ packet is generated from a fixed unary-continuation base history, a displayed object history, a displayed CategoryHomCarrier arrow into the base, a fiber condition, componentwise hsame transport preserving the base and arrow rows, replayable Cont routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SliceCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1179.1, Definition 1179.2, and Definition 1179.3 bind the finite carrier, base-fixation obligation, and local NameCert surface for the unary-continuation slice packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SliceCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host slice categories, quotient arrow equality, external triangle diagrams, arbitrary pullbacks, comma categories, functor categories, Yoneda representability, or checked Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, base fixation, arrow-to-base classification, componentwise transport stability, continuation replay, ledger provenance, and public non-escape for the same finite packet.

Chapter 1180

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$

The $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ packet realizes the Phase Real and Completion route of chapter 9238 as a single finite frontier between regular rational names and the completed real-facing seal. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational source of chapter 301, the displayed refinement route of chapter 885, and the terminal real-name seal of chapter 38. The frontier is not a completed real by itself; it is the finite selector, cofinal-tail, agreement, and seal-handoff surface a consumer must traverse before any $\mathsf{Real}^{\uparrow}$ limit is public.

Definition 1180.1 (Regular Cauchy completion frontier carrier)**.** *A* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, L, M, A, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger used by the regular Cauchy readback;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule through which every window request is exposed;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row;*
4. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row, read only after the source, selector, and cofinal-tail rows are displayed;*
5. *M is the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *refinement row binding the source windows, selected window, and tolerance readback;*
6. *A is the diagonal-selector row that chooses finite windows from the displayed schedules and never from a host choice function;*
7. *T is the cofinal-tail agreement row, comparing the selected regular tails before the completion seal is consumed;*
8. *E is the completion-seal handoff row from the selected regular tail to the* $\mathsf{Real}^{\uparrow}$ *seal;*
9. *H is the componentwise* hsame *transport row for D, S, R, L, M, A, T, E;*
10. *C is the* Cont *replay row for the same source, selector, tail, and seal route;*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *row.*

No coordinate of F is a quotient stream, a selected limit, host real equality, a countable-choice selector, or an ambient completeness witness.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionFrontierUp.taste gate`

Theorem 1180.2 (Regular Cauchy completion frontier NameCert obligations)**.** *For the carrier of Definition 1180.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the displayed packet $F = (D, S, R, L, M, A, T, E, H, C, P, N)$;*
2. *source-window exposure, requiring every consumer to read D, S, R, M before any diagonal selector or completion seal is public;*
3. *selector and cofinal-tail stability, requiring A and T to be transported by the same* hsame *and* Cont *route as the source rows;*
4. *completion-seal handoff, requiring E to expose the selected cofinal-tail agreement before the* $\mathsf{Real}^{\uparrow}$ *seal row L can be consumed;*
5. *ledger non-escape, requiring every accepted Real and Completion consumer to factor through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionFrontierUp.RegularCauchyCompletionFrontierNameCert obligations`

Proof. Carrier admission is projection from Definition 1180.1. The finite packet displays the dyadic, schedule, regular-source, refinement, selector, tail, seal, transport, continuation, provenance, and naming rows as one route.

Source-window exposure reads the sibling surfaces in the order forced by the carrier: $\mathsf{DyadicRatCore}^{\uparrow}$ supplies tolerances, $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular readback, and $\mathsf{CauchyModulusRefinement}^{\uparrow}$ binds the selected window and tolerance route.

Selector and tail stability use the displayed H and C rows. They transport and replay the same source route through A and T, so the diagonal selector and cofinal-tail agreement cannot be detached from the source window.

The completion-seal handoff is the row E followed by L. Since L is exposed only after the selected cofinal tail, a consumer cannot read a $\mathsf{Real}^{\uparrow}$ limit without the displayed seal handoff.

Ledger non-escape follows from P and N. The package and local certificate name only the displayed route, so quotient streams, selected limits, countable choice, host equality, and ambient completeness have no certificate coordinate. □

Theorem 1180.3 (Regular Cauchy completion frontier nonescape)**.** *Every accepted* RegularCauchyCompletionFront *consumer factors through*

$$D, S, R, L, M, A, T, E, H, C, P, N.$$

In particular, no consumer can export a quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness witness as an alternate Real and Completion route.

Proof. Read the carrier. Its public route contains the dyadic tolerance row, stream schedule, regular rational source, Real seal, modulus refinement, diagonal selector, cofinal-tail agreement, completion seal, transport, continuation, package, and local certificate rows.

Apply the NameCert obligations of Definition 1180.2. Source-window exposure places D, S, R, M before selector access, selector and tail stability bind A, T to the same route, and completion-seal handoff places E before L.

The forbidden objects have no row in the packet. A quotient stream, selected host limit, host real equality proof, countable-choice selector, or ambient completeness witness therefore cannot supply another route for an accepted frontier consumer. □

Theorem 1180.4 (Regular Cauchy completion frontier source exhaustion)**.** *Every accepted frontier read consumed by a* $\mathsf{Real}^{\uparrow}$ *consumer or a completion-facing consumer factors through exactly the displayed rows*

$$D, S, R, L, M, A, T, E, H, C, P, N$$

of Definition 1180.1. Here D is the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source, M is the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *refinement row, A, T, E are the selector, cofinal-tail, and completion-seal handoff rows, and L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The rows H, C, P, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows for the same route.*

Consequently no Real or Completion handoff may read a quotient stream, selected host limit, host real equality proof, countable-choice selector, detached completion witness, or ambient completeness principle outside those coordinates.

Proof. Project the carrier of Definition 1180.1. Its coordinates display the dyadic ledger, stream schedule, regular rational source, Real seal, modulus refinement, selector, cofinal-tail agreement, completion-seal handoff, transport, continuation, package, and local naming rows.

Apply Definition 1180.2. Source-window exposure forces D, S, R, M before selector access, selector and cofinal-tail stability keep A, T on the same route, and completion-seal handoff places E before the terminal seal row L.

Finally use Definition 1180.3. Any accepted Real or Completion read outside the displayed coordinates would be an alternate route through an absent source. Since the forbidden objects have no carrier row and no local certificate coordinate, the listed rows exhaust the reusable handoff surface. □

Theorem 1180.5 (Regular Cauchy completion frontier scoped dependency route)**.** *For an accepted* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *packet*

$$F = (D, S, R, L, M, A, T, E, H, C, P, N),$$

the scoped dependency route is exactly the following finite surface: $\mathsf{DyadicRatCore}^{\uparrow}$ *through D,* $\mathsf{StreamName}^{\uparrow}$ *through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *through R,* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *through M,* $\mathsf{Real}^{\uparrow}$ *through L, the diagonal-selector row A, the cofinal-tail row T, the completion-seal handoff row E, componentwise* hsame *through H,* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *row N.*

No quotient stream, selected limit, host equality, countable-choice selector, ambient completeness principle, metric-completeness principle, or bridge row belongs to the scoped dependency route.

Proof. Project the carrier of Definition 1180.1. The displayed coordinates name the same sibling anchors cited at the chapter entry: chapter 291, chapter 126, chapter 301, chapter 885, and chapter 38.

Use the NameCert obligations of Definition 1180.2. They force source-window exposure through D, S, R, M, keep the selector and cofinal-tail rows A, T on the same hsame and Cont route, and place the completion-seal handoff row E before the Real seal row L.

The exhaustion and non-escape clauses of Definition 1180.4 and Definition 1180.3 remove every alternate source. What remains is precisely the listed finite dependency route, with no coordinate for quotient streams, selected limits, host equality, choice, ambient or metric completeness, or bridge data. □

Theorem 1180.6 (Regular Cauchy completion frontier obligation closure)**.** *For* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$, *the carrier row, source-window exposure, selector and cofinal-tail stability, completion-seal handoff, and ledger non-escape obligations are all closed by the displayed finite packet*

$$D, S, R, L, M, A, T, E, H, C, P, N.$$

The packet therefore gives an obligation-closed Real and Completion frontier surface for accepted consumers, with the carrier alignment tracked by the existing TasteGate marker `BEDC.Derived.RegularCauchyCom`

Proof. The carrier row is exactly Definition 1180.1: it displays the dyadic ledger, finite observation schedule, regular rational source, Real seal, modulus refinement, diagonal selector, cofinal-tail agreement, completion-seal handoff, transport, continuation, package, and local certificate coordinates.

The five obligation rows are supplied by Definition 1180.2. Carrier admission, source-window exposure, selector and cofinal-tail stability, completion-seal handoff, and ledger non-escape are all stated for the same packet $D, S, R, L, M, A, T, E, H, C, P, N$.

The boundary is closed for accepted consumers by Definition 1180.3 and Definition 1180.4. The first excludes alternate routes through absent quotient, host-limit, equality, choice, and ambient-completeness coordinates; the second says every accepted Real or Completion read factors through exactly the displayed rows. The existing TasteGate marker records the Lean-side carrier alignment for that same singleton carrier surface. □

Theorem 1180.7 (Regular Cauchy completion frontier real terminal cofinal route)**.** *For any accepted* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *packet*

$$F = (D, S, R, L, M, A, T, E, H, C, P, N),$$

every public $\mathsf{Real}^{\uparrow}$ *terminal seal read through the frontier factors through the finite source, selector, cofinal-tail, and completion handoff route*

$$D, S, R, M, A, T, E$$

before the terminal seal row L *is consumed. The rows* H, C, P, N *are the only support rows for this handoff:* H *supplies componentwise* hsame *transport,* C *supplies* Cont *replay,* P *supplies* Pkg *provenance, and* N *supplies the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *row.*

Conversely, any displayed packet with the rows D, S, R, M, A, T, E, L *and no support rows beyond* H, C, P, N *repacks as the same frontier-mediated public* $\mathsf{Real}^{\uparrow}$ *terminal seal route. The carrier alignment for this route is the existing TasteGate marker* `BEDC.Derived.RegularCauchyCompletionFrontierUp.`

Proof. Apply Definition 1180.5. It identifies the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, the selector row A, the cofinal-tail row T, the completion-seal row E, and the terminal $\mathsf{Real}^{\uparrow}$ seal row L.

Apply Definition 1180.4. Any public Real or Completion read through the frontier must use exactly the displayed coordinates, so the terminal seal cannot be read before the source, refinement, selector, cofinal-tail, and handoff rows have been traversed.

Apply Definition 1180.3. Since quotient streams, selected host limits, host equality, choice selectors, detached completion witnesses, and ambient completeness principles have no carrier row, the only support rows left for the terminal handoff are H, C, P, N.

For the converse, project the displayed packet back into the carrier of Definition 1180.1. The same rows supply the source window, cofinal tail, completion handoff, transport, continuation, provenance, and local naming coordinates, and the TasteGate marker `BEDC.Derived.RegularCauchyCompletionF` records the singleton carrier alignment for the resulting frontier-mediated terminal seal route. □

Theorem 1180.8 (Regular Cauchy completion frontier terminal seal correspondence)**.** *An accepted* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *packet determines exactly the public finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal boundary obtained by the composed route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, *the selector row A, the cofinal-tail row T, the completion handoff row E, and the terminal seal row L. Its only support rows are H, C, P, N, carrying componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *row.*

Conversely, any consumer of that public terminal boundary factors through the displayed rows D, S, R, M, A, T, E, L together with the support rows H, C, P, N. No quotient stream, selected host limit, host equality proof, countable-choice selector, ambient completeness principle, metric-completeness principle, or bridge datum belongs to this boundary.

Proof. Start with the accepted frontier packet. By Definition 1180.4, every public Real or Completion read through the frontier is exhausted by the displayed rows $D, S, R, L, M, A, T, E, H, C, P, N$.

The route part of that exhaustion is identified by Definition 1180.5. It places the public finite-window boundary through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, the selector row A, the cofinal-tail row T, the completion handoff row E, and the terminal seal row L, with H, C, P, N as the transport, continuation, provenance, and local certificate support.

Use Definition 1180.6 to close the carrier, source-window, selector, cofinal-tail, handoff, and non-escape obligations on the same finite packet. Then apply Definition 1180.7 to read the route as the terminal $\mathsf{Real}^{\uparrow}$ seal boundary rather than as a detachable completion object.

For the converse, any consumer of the displayed terminal boundary is already a public Real or Completion read through the accepted frontier. The source-exhaustion theorem forces it through D, S, R, M, A, T, E, L, and the scoped dependency route leaves only H, C, P, N as support rows.

The exclusions follow from the non-escape clauses used in those theorems. Quotient streams, selected limits, host equality, countable choice, ambient completeness, metric completeness, and bridge data have no carrier coordinate and no local NameCert row, so they cannot be part of the public terminal seal boundary. □

Theorem 1180.9 (Regular Cauchy completion frontier CofinalModulusSeal handoff)**.** *Every accepted* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *terminal seal route factors through the* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *cofinal tail-budget, tail-modulus,* $\mathsf{Real}^{\uparrow}$*-window budget,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before the public* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Equivalently, the frontier route

$$D, S, R, M, A, T, E, L, H, C, P, N$$

is accepted only when its cofinal-tail and completion-seal coordinates admit the already checked factorization surface `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealRegSeqRatWindowFactorizatio` *No quotient stream, selected host limit, host equality proof, countable-choice selector, ambient completeness principle, metric-completeness principle, or bridge datum is introduced by this handoff.*

Proof. Start with an accepted frontier terminal seal route. By Definition 1180.4, every Real or Completion read is exhausted by the displayed rows $D, S, R, L, M, A, T, E, H, C, P, N$.

Use Definition 1180.5 to read those rows as the concrete dependency surface through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, the selector row, the cofinal-tail row, the completion-seal row, componentwise hsame, Cont, Pkg, and the local $\mathsf{NameCert}$ row.

Apply Definition 1180.8. It identifies the same route as the public finite-window $\mathsf{Real}^{\uparrow}$ terminal seal boundary, so the terminal seal is not a detachable limit object.

Now compose that boundary with the checked CofinalModulusSeal route `BEDC.Derived.CofinalModulusSealU` The CofinalModulusSeal packet supplies the cofinal tail-budget, tail-modulus, and $\mathsf{Real}^{\uparrow}$-window budget rows over the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ sources, with the same hsame, Cont, Pkg, and $\mathsf{NameCert}$ support.

The exclusions are inherited from the frontier non-escape and scoped-route clauses. Quotient streams, selected host limits, host equality, countable choice, ambient completeness, metric completeness, and bridge data have no coordinate in either the frontier carrier or the CofinalModulusSeal factorization surface, so the public Real seal can only be read through the displayed handoff. □

Theorem 1180.10 (Regular Cauchy completion frontier Real four-face status factorization)**.** *Every accepted* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *packet whose terminal seal route factors through* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *presents exactly the four local status faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *used by Definition 38.815.*

The handoff supplies no route-level formalstatus row, aggregate leantarget row, aggregate bridge row, quotient stream, selected host limit, countable-choice row, metric-completeness datum, or ambient completeness claim. Its status surface is only the local four-face surface carried along the frontier terminal seal route.

Proof. First read the accepted packet through Definition 1180.8. The terminal seal boundary is the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the frontier source window, the cofinal-tail row, the completion handoff row, and the terminal $\mathsf{Real}^{\uparrow}$ seal, with only the displayed transport, continuation, package, and local NameCert support.

Next apply Definition 1180.9. The cofinal-tail and completion-seal coordinates factor through $\mathsf{CofinalModulusSeal}^{\uparrow}$ without adding a quotient stream, selected host limit, host equality proof, countable-choice selector, ambient completeness principle, metric-completeness principle, or bridge datum.

Finally compose this route with Definition 38.815. The Real terminal status theorem reads every formalstatus, leantarget, and bridgestatus coordinate from the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since the frontier handoff and the CofinalModulusSeal factorization introduce no aggregate coordinate, the accepted packet presents precisely that four-face status surface and no route-level completion or bridge claim. □

Theorem 1180.11 (Regular Cauchy completion frontier finite-cover support removal obstruction)**.** *Let* $\{Q_i\}_{i\in I}$ *be an accepted finite cover of* $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *terminal seal routes. Suppose the accepted local sections and overlaps agree on the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulusRefinement}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, selector, cofinal-tail, handoff,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$ *rows. If one displayed local support row is deleted from some member* Q_j*, then the cover-descent frontier packet cannot present the same* $\mathsf{Real}^{\uparrow}$ *terminal seal boundary.*

The obstruction is carried only by the displayed BEDC rows. Quotient streams, selected host limits, host equality proofs, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness,*

aggregate formalstatus rows, and bridge rows cannot restore the deleted support row or recover the same completion-frontier boundary.

Proof. First restrict the cover to the displayed $\mathsf{StreamName}^{\uparrow}$ terminal support surface. The hypotheses give the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert support rows on the accepted local sections and overlaps, so Definition 126.665 applies to the deleted local support row.

That obstruction says the descended StreamName terminal support cannot still present the same $\mathsf{Real}^{\uparrow}$ seal-visible source. It also rules out quotient stream equality, selected host limits, countable choice, metric-completeness data, aggregate formalstatus rows, bridge rows, ambient $\mathsf{Real}^{\uparrow}$ completeness, and replacement selectors as substitute coordinates.

Now transport the obstruction to the completion frontier. By Definition 1180.8, the accepted frontier boundary is exactly the public finite-window $\mathsf{Real}^{\uparrow}$ terminal seal obtained from the route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, the selector row, the cofinal-tail row, the completion handoff row, and the terminal seal row, with only H, C, P, N as support.

Apply Definition 1180.9 to keep the cofinal-tail and completion-seal coordinates on the same $\mathsf{CofinalModulusSeal}^{\uparrow}$ handoff surface. Then use Definition 1180.10 to read the status data only through the four local faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The frontier packet therefore has no separate completion coordinate through which the missing row can reappear. Host equality proofs are excluded with the same force: the terminal-seal correspondence and CofinalModulusSeal handoff give them no route coordinate and no local support row. Since the StreamName finite-cover obstruction has already removed the common terminal support and the frontier correspondence reads the boundary through that support, the cover-descent frontier packet cannot present the same $\mathsf{Real}^{\uparrow}$ terminal seal boundary after the row is deleted. □

Theorem 1180.12 (Regular Cauchy completion frontier cofinal-tail support uniqueness)**.** *Let*

$$F = (D, S, R, L, M, A, T, E, H, C, P, N)$$

and

$$F' = (D', S', R', L', M', A', T', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ *packets presenting the same public finite-window* $\mathsf{Real}^{\uparrow}$ *terminal seal boundary. If both terminal seal routes factor through the same* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *cofinal tail-budget, then the consumer-visible cofinal-tail support is exactly the displayed surface*

$$D, S, R, M, A, T, E, H, C, P, N.$$

Equivalently, F' presents the same boundary only by matching the dyadic tolerance ledger, stream schedule, regular source, refinement, selector, cofinal-tail, completion handoff, transport, continuation, provenance, and local NameCert rows on that surface. No quotient stream, selected host limit, host equality proof, countable-choice selector, ambient completeness principle, metric-completeness datum, aggregate status row, or bridge row can witness another same-seal route.

Proof. First read both accepted packets through Definition 1180.8. The common public terminal boundary is not a host limit or quotient object; it is the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusRefinement}^{\uparrow}$, the selector row, the cofinal-tail row, the completion handoff row, and the terminal $\mathsf{Real}^{\uparrow}$ seal, with only the displayed transport, continuation, package, and local NameCert support.

Next apply Definition 1180.9. Since the two routes factor through the same $\mathsf{CofinalModulusSeal}^{\uparrow}$ cofinal tail-budget, their tail-budget, tail-modulus, and $\mathsf{Real}^{\uparrow}$-window budget coordinates are read on the same RegSeqRat, StreamName, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert surface.

Use Definition 1180.10 to keep the status information local to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Thus the shared terminal seal has no aggregate formalstatus, leantarget, or bridge coordinate through which F' could present a different consumer-visible tail support.

Finally apply Definition 1180.11. Deleting or replacing any row among $D, S, R, M, A, T, E, H, C, P, N$ destroys the same terminal seal boundary, even under accepted cover descent. Therefore the second packet can present the same $\mathsf{Real}^{\uparrow}$ terminal seal over the same $\mathsf{CofinalModulusSeal}^{\uparrow}$ handoff only by matching that displayed support surface, and the listed forbidden objects have no coordinate from which to supply another route. □

Falsifiable prediction: An accepted frontier consumer that exports a $\mathsf{Real}^{\uparrow}$ limit without a displayed cofinal-tail row and completion-seal handoff row refutes the packet.

Independence witness: real, streamname, regseqrat, dyadicratcore, cauchymodulusrefinement: no sibling is bijective because RegularCauchyCompletionFrontierUp binds Real sealing, finite observation, regular rational readback, dyadic tolerance, modulus refinement, diagonal selection, cofinal-tail agreement, and completion handoff in one certificate. RealUp has endpoint seals without source-window selectors; StreamNameUp has windows without regular rational readback; RegSeqRatUp has regular sources without Real seals; DyadicRatCoreUp has tolerances without completion handoff; CauchyModulusRefinementUp has source-window refinement without the frontier's cofinal-tail and Real-seal package.

The Lean follow-up site `lean4/BEDC/Derived/RegularCauchyCompletionFrontierUp/TasteGate.lean` must provide Nontrivial and FieldFaithful instances for $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$, together with a carrier-alignment theorem named `BEDC.Derived.RegularCauchyCompletionFrontierUp.RegularCauchyC`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}$ packet is generated from displayed DyadicRatCore, StreamName, RegSeqRat, RealUp, CauchyModulusRefinement, diagonal-selector, cofinal-tail, completion-seal, transport, continuation, package, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1180.5 binds the route through DyadicRatCore, StreamName, RegSeqRat, CauchyModulusRefinement, RealUp, selector, cofinal-tail, seal handoff, hsame, Cont, Pkg, and local NameCert rows, while Definition 1180.6 keeps the finite obligation surface closed and `BEDC.Derived.RegularCauchyCompletionFrontierUp.RegularCauchyCompletio` witnesses carrier alignment for `BEDC.Derived.RegularCauchyCompletionFrontierUp.taste_gate`.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFrontier}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyCompletionFrontierUp.RegularCauchyCompletionFrontierTasteGate\_single\_car`

Bridge status: none

Origin: ai

Not claimed: This chapter does not prove bridge checking, uniqueness of Real limits, quotient completion, countable choice, host real equality, or ambient completeness.

Upgrade path: Bridge checking remains outside this chapter.

Chapter 1181

A Concrete Naming Certificate for $\mathsf{DomainTruthCertificate}^{\uparrow}$

The $\mathsf{DomainTruthCertificate}^{\uparrow}$ packet realizes the domain-indexed truth surface of section 9414.70 as a finite BEDC NameCert carrier. It sits beside certified physical truth from chapter 1130, open fit from chapter 1132, synthesis sealing from chapter 1140, final-synthesis ledgering from chapter 1178, and failure-surface exposure from chapter 1163. Those siblings supply fit, open continuation, seal, ledger, and defeat surfaces; this packet binds the displayed domain row before any truth export is accepted.

Definition 1181.1 (DomainTruthCertificate carrier)**.** *A* $\mathsf{DomainTruthCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, U, B, A, C, O, F, H, R, P, N)$$

with the following displayed rows:

1. *T is the certified truth row naming the local claim, source traces, and reality-facing signature admitted by the packet.*
2. *U is the domain row recording the observation bundle, approximation tower, continuation class, and observer configuration over which the claim is read.*
3. *B is the open-fit row recording the finite fit comparison and the boundary at which the domain remains open.*
4. *A is the observer-invariance row recording the anchor or observer transport under which the claim remains publicly readable.*
5. *C is the continuation row recording the permitted reuse and replay route inside the displayed domain.*
6. *O is the failure-surface row naming domain exits, classifier mismatches, finite defeats, and later observations that can stop the export.*
7. *F is the transport row recording componentwise* hsame *preservation for the truth, domain, open-fit, invariance, continuation, failure, replay, provenance, and local-name rows.*
8. *H is the displayed* Cont *replay row through the same finite packet.*
9. *R is the provenance row recording source, tower, classifier, ledger, sibling-certificate, and construction references.*

10. *P is the package row over* BHist, hsame, Cont, Pkg, NameCert, *and the cited sibling packets.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}$ *row binding the displayed domain-indexed truth packet.*

The carrier has no coordinate for final truth, model–reality identity, host truth, unbounded observation, hidden observer privilege, or truth export outside the displayed domain row.

Lean4 ✓ `BEDC.Derived.DomainTruthCertificateUp.taste gate`
Lean4 ✓ `BEDC.Derived.DomainTruthCertificateUp.DomainTruthCertificateTasteGate single carrier alignment`

Theorem 1181.2 (DomainTruthCertificate NameCert obligations)**.** *For the carrier of Definition 1181.1, the local* $\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission, domain-scope display, open-fit display, observer-invariance display, failure-surface display, componentwise transport, displayed replay, provenance, and local naming through N.*

Proof. Project the carrier $D = (T, U, B, A, C, O, F, H, R, P, N)$. The rows T, U, B, A, C, O expose the certified truth claim, the domain scope, the open-fit boundary, the observer-invariant route, the continuation replay scope, and the failure surface.

Read the structural rows. Transport through F preserves each displayed component, replay through H keeps the consumer inside the same finite route, provenance through R records the source and sibling dependencies, and P, N package and locally name the certificate.

Since no additional coordinate is available in the carrier, these rows exhaust the local NameCert surface. A domain-truth export that omits one row is only a projection, not an accepted $\mathsf{DomainTruthCertificate}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.DomainTruthCertificateUp.DomainTruthCertificateNameCertObligations`

Theorem 1181.3 (DomainTruthCertificate non-escape)**.** *An accepted* $\mathsf{DomainTruthCertificate}^{\uparrow}$ *packet excludes final truth, model–reality identity, host truth, unbounded observation, and domain leakage. Any export that asserts validity outside the displayed domain row U, or reads truth from a coordinate absent from*

$$T, U, B, A, C, O, F, H, R, P, N$$

is rejected by the local $\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}$ *surface.*

Lean4 ✓ `BEDC.Derived.DomainTruthCertificateUp.DomainTruthCertificateUp nonescape`

Proof. Use the carrier of Definition 1181.1. The truth row T is always read together with the domain row U, the open-fit row B, the observer-invariance row A, the continuation row C, and the failure row O.

The structural rows cannot widen the domain. Transport through F preserves displayed components, replay through H consumes the same route, provenance through R records sources already named in the packet, and P, N package and name the same finite certificate.

Thus final truth, model–reality identity, host truth, unbounded observation, and domain leakage would require a coordinate not present in the packet. The accepted route certifies domain truth only within the displayed finite surface. □

Theorem 1181.4 (DomainTruthCertificate sibling independence)**.** *The* $\mathsf{DomainTruthCertificate}^{\uparrow}$ *packet is not bijective to* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$, $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$, *or* $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$. *Each sibling supplies a required surface, but none alone carries the displayed domain row tying truth, continuation, observer, failure, transport, replay, provenance, package, and local naming into one finite certificate.*

Proof. Compare the displayed carrier with the sibling roles. $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ supplies the certified truth read, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$ supplies finite open fit, $\mathsf{RealityConstrainedSynthesisSeal}^{\uparrow}$ supplies the synthesis seal, $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$ supplies ledger discipline, and $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ supplies the defeat surface.

The row U is the separating coordinate. It binds the observation bundle, approximation tower, continuation class, and observer configuration as the domain of the truth read, and the rows C, F, H, R, P, N replay, transport, source, package, and name that same domain-indexed packet.

Therefore no listed sibling is bijective to $\mathsf{DomainTruthCertificate}^{\uparrow}$. Removing the domain row erases the certificate's scope, while replacing any sibling with the domain packet erases the sibling's specialized surface. □

Lean4 ✓ `BEDC.Derived.DomainTruthCertificateUp.DomainTruthCertificateUp sibling independence`

Falsifiable prediction: A domain-truth export that omits the domain row or asserts final validity over all future observations is not accepted by the local NameCert.

Independence witness: certifiedphysicaltruth, realityconstrainedopenfit, realityconstrainedsynthesisseal, realityconstrainedfinalsynthesisledger, realityconstrainedfailuresurface: no sibling is bijective because DomainTruthCertificateUp adds the displayed certified-domain row tying truth, continuation, observer, failure, transport, replay, provenance, and local naming into one carrier.

The checked TasteGate carrier-alignment target records that the nontriviality and field-faithful rows are present for the displayed carrier surface.

For the paper-side standard bridge schema, the displayed domain row U remains mandatory for every standard truth export. The final-truth, host-truth, and unbounded-observation readings are still refused: a bridge-facing consumer may transport and replay only the certified truth row, the domain row, open-fit and observer rows, continuation and failure rows, provenance, package data, and local naming already displayed by the carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{DomainTruthCertificate}^{\uparrow}$ packet is generated from a certified truth row, displayed domain row, open-fit row, observer-invariance row, continuation row, failure surface, componentwise transport, replay, provenance, package data, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1181.1, Definition 1181.2, Definition 1181.3, Definition 1181.4, the falsifiable prediction, the independence witness, and the TasteGate markers expose a public finite domain-truth certificate with carrier, obligation, nonescape, sibling independence, domain-scope, open-fit, observer-invariance, failure-surface, transport, replay, provenance, package, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DomainTruthCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, bridge checking, final truth, model–reality identity, host truth, unbounded observation, or truth export outside the displayed domain row.

Upgrade path: Only Lean bridge checking remains on the verification axis.

Chapter 1182

A Concrete Naming Certificate for $\mathsf{PredictiveDescent}^{\uparrow}$

The $\mathsf{PredictiveDescent}^{\uparrow}$ packet realizes the predictive reuse and descent condition of section 9414.25 as a finite BEDC NameCert surface. It sits beside the physical-induction packet of chapter 1135, the stability seal of chapter 1139, the law certificate of chapter 1155, the open-fit packet of chapter 1137, and the failure surface of chapter 1163. Those siblings supply induction, stability, law, fit, and failure-surface rows; this packet isolates the prediction-operation descent row between observed and continuation classifiers.

Definition 1182.1 (PredictiveDescent carrier)**.** *A* $\mathsf{PredictiveDescent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, O, K, A, S, X, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite-fit row recording the observed history, fitted model, comparison bundle, and visible signature that admitted the source classifier.*
2. *O is the observed-classifier row recording the classifier under which the fitted model was formed.*
3. *K is the continuation-classifier row recording the classifier used after predictive reuse over the displayed continuation class.*
4. *A is the prediction-operation row naming the operation whose output is consumed by the continuation classifier.*
5. *S is the stability-scope row recording the continuation class, reuse license, and sibling stability certificate.*
6. *X is the failure-surface row recording classifier mismatches, finite defeats, and continuations on which descent is not licensed.*
7. *H is the componentwise* hsame *transport row preserving finite fit, observed classifier, continuation classifier, prediction operation, stability scope, and failure surface.*
8. *C is the displayed* Cont *replay row through the same predictive-descent packet.*
9. *P is the package row over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the sibling packets cited above.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}$ *row binding the displayed predictive-descent packet.*

The carrier has no coordinate for representative-dependent prediction, unbounded future success, final empirical confirmation, hidden model state, host real equality, or a continuation classifier outside the displayed row K.

Lean4 ✓ `BEDC.Derived.PredictiveDescentUp.PredictiveDescentTasteGate single carrier alignment`

Theorem 1182.2 (PredictiveDescent NameCert obligations)**.** *For the carrier of Definition 1182.1, the local* $\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed tuple $F, O, K, A, S, X, H, C, P, N$;*
2. *finite-fit exposure through F together with observed-classifier exposure through O;*
3. *prediction-operation descent from O through A to the continuation classifier K;*
4. *stability-scope and failure-surface exposure through S and X;*
5. *structural transport, replay, provenance, package, and local naming through H, C, P, N.*

Proof. Project the carrier $D = (F, O, K, A, S, X, H, C, P, N)$. The first two obligations read the finite-fit and observed-classifier data from F and O.

The descent obligation is the separating row. It records that the prediction operation A is accepted only when its output is readable by the continuation classifier K from the observed classifier O.

The stability and failure rows then bound the route. The row S fixes the continuation class and reuse license, while X records the continuations and classifier mismatches that stop the predictive export.

Finally the structural rows H, C, P, N transport, replay, package, and locally name the same finite packet. Since the carrier has no other coordinates, these five rows exhaust the local NameCert obligation surface. □

Theorem 1182.3 (PredictiveDescent stability scope)**.** *An accepted* $\mathsf{PredictiveDescent}^{\uparrow}$ *packet licenses predictive reuse only over the displayed stability scope S and only through the prediction operation A descending from the observed classifier O to the continuation classifier K. A consumer that changes the continuation class, replaces either classifier, or reads a different prediction operation leaves the accepted packet surface.*

Lean4 ✓ `BEDC.Derived.PredictiveDescentUp.PredictiveDescentStabilityScope`

Proof. Use the obligation surface of Definition 1182.2. The stability row S is part of the same local certificate as O, K, A, so the continuation classifier cannot be read independently of the displayed stability scope.

The transport and replay rows do not widen the scope. They preserve and replay the listed components of the same finite packet, while the package and local name authenticate only that packet.

Therefore the accepted reuse is scoped by S and by the descent route $O \to A \to K$. Changing the continuation class, either classifier, or the prediction operation requires a coordinate absent from the carrier. □

Theorem 1182.4 (PredictiveDescent non-escape)**.** *No accepted* $\mathsf{PredictiveDescent}^{\uparrow}$ *packet exports representative-dependent prediction, unbounded future success, final empirical confirmation, hidden model state, host real equality, or a continuation classifier outside the displayed row K. Any packet whose prediction operation respects finite fit but fails descent from the observed classifier to the continuation classifier is rejected by the local* $\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}$ *surface.*

Proof. Read the carrier of Definition 1182.1. The only prediction route available to a consumer is the displayed path through finite fit, observed classifier, prediction operation, continuation classifier, stability scope, and failure surface.

Apply the NameCert obligations. They require the descent row from O through A to K before the continuation classifier is accepted, and they keep the failure surface visible as X.

Finally use the absence of additional coordinates. Representative-dependent prediction, unbounded future success, final empirical confirmation, hidden model state, host real equality, and off-row continuation classifiers are not entries of D. A finite-fit-preserving operation that does not descend from O to K therefore lacks the required local naming certificate. □

Falsifiable prediction: A packet whose prediction operation respects finite fit but fails descent from the observed classifier to the continuation classifier is rejected by the PredictiveDescentUp classifier.

Independence witness: realityconstrainedphysicalinduction, physicalinductionstabilityseal, lawcertificate, openfitpacket, realityconstrainedfailuresurface: none is bijective because PredictiveDescentUp isolates the prediction-operation descent row between observed and continuation classifiers, while the siblings supply induction, stability, law, fit, or failure-surface rows.

Lean follow-up must provide `Nontrivial PredictiveDescentUp` and `FieldFaithful PredictiveDescentUp` instances in `lean4/BEDC/Derived/PredictiveDescentUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}$.

Constructive story: A $\mathsf{PredictiveDescent}^{\uparrow}$ packet is generated from finite fit, observed classifier, continuation classifier, prediction operation, stability scope, failure surface, componentwise transport, continuation replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1182.1, Definition 1182.2, Definition 1182.3, and Definition 1182.4 bind the finite predictive-descent carrier, five-row NameCert obligation surface, scoped reuse theorem, and non-escape theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PredictiveDescent}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, bridge checking, unbounded future success, final empirical confirmation, representative-dependent prediction, or a continuation classifier outside the displayed packet.

Upgrade path: Obligation closure requires checked carrier admission, finite-fit exposure, classifier descent, stability-scope exposure, failure-surface exposure, transport, replay, provenance, local naming, TasteGate nontriviality, and field faithfulness for $F, O, K, A, S, X, H, C, P, N$.

Chapter 1183

A Concrete Naming Certificate for $\mathsf{ContinuationCountermodel}^{\uparrow}$

The $\mathsf{ContinuationCountermodel}^{\uparrow}$ packet realizes the open-continuation refusal surface of section 9414.25 as a finite counter-continuation certificate. It sits beside the physical-induction packet of chapter 1135, the stability seal of chapter 1139, the law certificate of chapter 1155, the open-fit packet of chapter 1132, and the failure surface of chapter 1163. Those chapters supply fit, reuse, law, and defeat rows; this packet stores the single finite continuation route where predictive reuse is refused.

Definition 1183.1 (ContinuationCountermodel carrier)**.** *A* $\mathsf{ContinuationCountermodel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (F, U, M, O, K, S, E, H, R, P, N)$$

with the following displayed rows:

1. *F is the finite-fit row recording the observed prefix, classifier, and model agreement used by the proposed reuse claim;*
2. *U is the admissible-continuation row naming the continuation class being tested;*
3. *M is the model-prediction row exported from the fitted model along U;*
4. *O is the observed-continuation row recorded on the same continuation route;*
5. *K is the mismatch and counter-row comparing M with O at the displayed continuation;*
6. *S is the stability-certificate row whose reuse claim is being tested;*
7. *E is the failure-surface row recording the displayed defeat of predictive reuse;*
8. *H records componentwise* hsame *transport for $F, U, M, O, K, S, E, R, P, N$;*
9. *R is the displayed replay route tying the finite fit, prediction, observed continuation, mismatch, stability basis, and failure surface into one counter-continuation;*
10. *P is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and the sibling physical-induction, stability-seal, law-certificate, open-fit, and failure-surface rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuationCountermodel}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying all eleven rows. It has no coordinate for final empirical truth, global model failure, unbounded future history, hidden continuation choice, or predictive reuse beyond the counter-continuation route in R.

Lean4 (definition) BEDC.Derived.ContinuationCountermodelUp.ContinuationCountermodelUp
Lean4 ✓ BEDC.Derived.ContinuationCountermodelUp.ContinuationCountermodelTasteGate single carrier alignment

Theorem 1183.2 (ContinuationCountermodel NameCert obligations)**.** *For the carrier of Definition 1183.1, the local* $\mathsf{NameCert}_{\mathsf{ContinuationCountermodel}^{\uparrow}}$ *surface is exhausted by five obligation rows:*

1. *carrier admission for* $C = (F, U, M, O, K, S, E, H, R, P, N)$*;*
2. *finite-fit and admissible-continuation exposure through* F *and* U*;*
3. *prediction and observation comparison through* M, O, K*;*
4. *stability-certificate and failure-surface exposure through* S *and* E*;*
5. *componentwise transport, replay, provenance, and local naming through* H, R, P, N*.*

Every accepted counter-continuation must display these rows before it can defeat predictive reuse for the stated continuation class.

Proof. Project the carrier of Definition 1183.1. The row list displays the finite fit, admissible continuation, model prediction, observed continuation, mismatch row, stability certificate, failure surface, transport, replay, provenance, and local name.

The first two obligations bind the source of the reuse claim: F supplies only finite fit, and U names only the continuation class under test. They do not export a global future-success or global future-failure claim.

The comparison obligation is carried by M, O, K. The model prediction and observed continuation are both read on the displayed route, and K records the separating row that blocks predictive reuse for that continuation.

The stability and failure rows S, E then place the mismatch against the exact certificate being tested. Transport, replay, provenance, and local naming through H, R, P, N keep the countermodel inside the same finite packet. Since no further coordinate is present, the five rows exhaust the local NameCert surface. □

Lean4 ✓ BEDC.Derived.ContinuationCountermodelUp.ContinuationCountermodelNameCertObligations

Theorem 1183.3 (ContinuationCountermodel predictive-reuse defeat)**.** *An accepted* $\mathsf{ContinuationCountermodel}^{\uparrow}$ *packet rejects predictive reuse for the displayed continuation class whenever* F *matches the finite fit row,* U *admits the tested continuation, and the counter-row* K *records a mismatch between the model prediction* M *and the observed continuation* O*. The rejection is local to the route* R *and does not assert global model failure outside the displayed packet.*

Proof. Begin with the NameCert obligations of Definition 1183.2. They require $F, U, M, O, K, S, E, H, R, P, N$ to be read together before a counter-continuation is accepted.

The rows F and U keep the source finite fit and continuation class visible. Thus the packet does not reject the model merely because a different continuation or a different finite-fit row exists.

The rows M, O, K supply the defeating comparison. Once K displays the mismatch between the prediction and the observed continuation on the same route, the stability certificate S cannot be reused as a predictive license for that continuation class.

Finally E, H, R, P, N localize the defeat to the displayed failure surface, transport, replay route, provenance, and local name. The packet therefore defeats predictive reuse on that route without claiming a final oracle about every possible continuation. □

Lean4 ✓ BEDC.Derived.ContinuationCountermodelUp.ContinuationCountermodelUp root replay transport

Theorem 1183.4 (ContinuationCountermodel replay non-escape)**.** *Every consumer of an accepted* $\mathsf{ContinuationCountermodel}^{\uparrow}$ *packet must read the counter-continuation through*

$$F, U, M, O, K, S, E, H, R, P, N.$$

No consumer may detach the mismatch row from the finite fit, admissible continuation, stability certificate, failure surface, replay route, provenance, or local naming row and then treat it as an unbounded physical-law refutation.

Proof. Use the carrier definition. The mismatch row K is one coordinate of the finite packet, not an independent global falsifier.

Apply Definition 1183.3. The predictive-reuse rejection depends on F, U, M, O, K, S, E being read along the replay route R.

The transport and provenance rows H, P preserve that same packet, and N names it locally. Because no coordinate exports unbounded physical-law refutation, hidden continuation choice, or global model failure, a consumer that detaches K from the displayed packet leaves the accepted surface. □

The later Lean file `lean4/BEDC/Derived/ContinuationCountermodelUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ ContinuationCountermodel}^{\uparrow}$ and $\mathsf{FieldFaithful\ ContinuationCountermodel}^{\uparrow}$ instances preserving $F, U, M, O, K, S, E, H, R, P, N$.

Falsifiable prediction: A continuation packet with matching finite fit but a displayed mismatch row is rejected as predictive reuse for that continuation class.

Independence witness: realityconstrainedphysicalinduction, physicalinductionstabilityseal, lawcertificate, realityconstrainedfailuresurface, realityconstrainedopenfit: none is bijective because ContinuationCountermodelUp stores the single counter-continuation route tying finite fit, prediction, observed continuation, mismatch row, stability basis, failure surface, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuationCountermodel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuationCountermodel}^{\uparrow}$ packet is generated from finite fit, admissible continuation, model prediction, observed continuation, mismatch, stability certificate, failure surface, componentwise transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuationCountermodel}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1183.1, Definition 1183.2, Definition 1183.3, Definition 1183.4, the falsifiable prediction, the independence witness, and the TasteGate row `BEDC.Derived.ContinuationCountermodelUp.ContinuationCountermodelTasteGate_sing` bind the mature finite counter-continuation package: carrier, NameCert obligations, predictive-reuse defeat, replay non-escape, nontriviality witness, and field-faithful row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuationCountermodel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim Lean verification, bridge checking, global model failure, total physical-law refutation, final empirical truth, or continuations outside the displayed route.

Upgrade path: Scoped closure requires exact BEDC source rows for finite fit, admissible continuation, prediction, observation, mismatch, stability, failure surface, transport, replay, provenance, local naming, TasteGate nontriviality, and field faithfulness for $F, U, M, O, K, S, E, H, R, P, N$.

Chapter 1184

A Concrete Naming Certificate for RealityConstrainedModelErrorBudget$^{\uparrow}$

The RealityConstrainedModelErrorBudget$^{\uparrow}$ packet realizes the error-budget branch of the model-audit program in section 9414.60 as a finite NameCert surface. It sits beside the model-audit certificate of chapter 1124, the finite-observation residue boundary of chapter 1167, the open-fit packet of chapter 1137, the model-selection boundary of chapter 1165, and the failure surface of chapter 1163. Those siblings supply audit, residue, fit, selection, and defeat rows; this packet stores the bounded mismatch ledger by which a model-facing claim exposes its finite error evidence before any audit or selection consumer reads it.

Definition 1184.1 (Reality-constrained model error-budget carrier)**.** *A* RealityConstrainedModelErrorBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, E, O, M, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-trace row naming the audited source histories and instrumentation route.*
2. *E is the error-budget row recording the declared tolerance, residual allowance, measurement uncertainty, and admissible mismatch envelope.*
3. *O is the observation-window row that bounds which finite observations can consume the budget.*
4. *M is the mismatch-ledger row comparing the observed and model-generated signatures inside the budget.*
5. *F is the failure-surface row naming budget overruns, hidden residuals, classifier exits, and model-selection defeats.*
6. *H records componentwise* hsame *transport for the source, budget, observation-window, mismatch-ledger, and failure-surface rows.*
7. *C records the displayed* Cont *route through source tracing, observation-window reading, mismatch comparison, failure exposure, and downstream audit consumption.*
8. *P is the* Pkg *provenance over* RealityConstrainedModelErrorBudget$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, FiniteObservationResidueBoundary$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, ModelSelectionBoundary$^{\uparrow}$, RealityConstrainedFailureSurface$^{\uparrow}$ BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedModelErrorBudget}^{\uparrow}}$ *row.*
The carrier has no coordinate for unbounded validation, hidden residual cancellation, model syntax truth, detached fit scoring, off-window observation repair, or a selection oracle outside the displayed budget packet.

Lean4 (definition) `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetUp`

Theorem 1184.2 (Reality-constrained model error-budget NameCert obligations)**.** *For the carrier of Definition 1184.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedModelErrorBudget}^{\uparrow}}$ *surface consists exactly of source-trace exposure through S, error-budget exposure through E, observation-window exposure through O, mismatch-ledger exactness through M, failure-surface exposure through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetNameCert obligations`

Proof. Project the carrier $B = (S, E, O, M, F, H, C, P, N)$. The first three rows expose the finite source trace, declared budget, and observation window before any consumer can read model fit or audit strength.

The mismatch and failure rows supply the budget-facing certificate. The row M records the observed-versus-model comparison inside the declared envelope, while F names the finite ways the envelope can fail by overrun, hidden residual, classifier exit, or selection defeat.

The structural rows preserve that same packet. Transport through H is componentwise, replay through C follows the displayed source-window-mismatch-failure route, provenance through P records the sibling certificates consumed by the packet, and N supplies the local name.

Since the carrier has no further coordinate, these rows exhaust the local NameCert surface. A model-audit or selection read that depends on an off-window observation, hidden residual cancellation, or detached fit score is not a projection of the accepted packet. □

Theorem 1184.3 (Reality-constrained model error-budget nonescape)**.** *Every accepted* RealityConstrainedModelErr *consumer must read bounded model-error evidence through*

$$S, E, O, M, F, H, C, P, N.$$

No consumer may preserve the model syntax while replacing the error-budget row, changing the observation window, hiding the mismatch ledger, erasing the failure surface, or importing a model-selection verdict from outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier nonescape`

Proof. Start with the NameCert obligations of Definition 1184.2. They make $S, E, O, M, F, H, C, P, N$ the complete row surface for a bounded error-budget read.

The budget row E and observation window O bind the scope of the mismatch ledger. A comparison in M is licensed only for the source traces and finite observations exposed by S and O under the declared tolerance and residual allowance in E.

The failure row F keeps rejection conditions visible. Budget overruns, hidden residuals, classifier exits, and model-selection defeats cannot be suppressed and then recovered from transport, replay, provenance, or local naming rows.

Thus the accepted packet has no route that preserves only model syntax and changes the error evidence. Any such route leaves the finite carrier and cannot be consumed as a RealityConstrainedModelErrorBudget$^{\uparrow}$ certificate. □

Theorem 1184.4 (Reality-constrained model error-budget observation-window exactness)**.** *Every mismatch-ledger read in an accepted* RealityConstrainedModelErrorBudget$^{\uparrow}$ *carrier preserves the source trace S, error-budget row E, observation-window row O, and mismatch-ledger row M as one finite packet. A consumer cannot read M while changing the source route, widening the observation window, or detaching the comparison from the declared budget envelope.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier observation window exactness`

Proof. Project the carrier $B = (S, E, O, M, F, H, C, P, N)$. The row M is introduced only after the source trace S, budget row E, and observation window O have been displayed.

The sibling audit chapters named in the introduction supply separate audit, residue, fit, selection, and failure rows, but none of them replaces the packet order here. The mismatch comparison remains a read of the displayed source-window-budget surface.

Transport and replay preserve the same dependency. Rows H and C move and replay S, E, O, M componentwise, while P and N only source and name the route. Hence any accepted mismatch-ledger read keeps those four rows together. □

Theorem 1184.5 (Reality-constrained model error-budget failure visibility)**.** *No accepted* RealityConstrainedModel *budget consumer may preserve the model syntax while hiding the failure-surface row F. The rows for budget overruns, hidden residuals, classifier exits, and model-selection defeats must remain visible whenever the source trace, budget, observation window, and mismatch ledger are exported.*

Proof. Use the carrier inventory and the nonescape theorem. The failure-surface row F is one of the displayed coordinates of every accepted budget packet, and Definition 1184.3 forbids erasing it while preserving model syntax.

The cited failure-surface sibling supplies the interpretation of rejection rows, but visibility is already local to this carrier. Once S, E, O, M are exported, the corresponding row F must travel with the packet so that overruns, hidden residuals, classifier exits, and selection defeats remain inspectable.

The structural rows add no hidden substitute. Transport through H, replay through C, provenance through P, and naming through N preserve the displayed failure row rather than replacing it by a detached fit score. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier failure visibility`

Theorem 1184.6 (Reality-constrained model error-budget selection handoff)**.** *Every* ModelSelectionBoundary$^{\uparrow}$ *consumer of an accepted* RealityConstrainedModelErrorBudget$^{\uparrow}$ *carrier may consume the budget only through*

$$S, E, O, M, F, H, C, P, N.$$

It receives source trace, error budget, observation window, mismatch ledger, failure surface, transport, replay, provenance, and local naming rows, not detached fit scoring outside the finite error-budget packet.

Proof. Begin with the model-selection sibling named in the introduction. Its consumer route can inspect a budget packet only through the rows that this chapter exposes.

Rows S, E, O, M, F bind the substantive evidence: source route, declared budget, observation window, mismatch comparison, and visible failure surface. A fit score without these rows is not a projection of the accepted $\mathsf{RealityConstrainedModelErrorBudget}^{\uparrow}$ carrier.

Rows H, C, P, N then transport, replay, source, and name the same packet for the selection boundary. Therefore the handoff to $\mathsf{ModelSelectionBoundary}^{\uparrow}$ is through $S, E, O, M, F, H, C, P, N$ and cannot be replaced by detached scoring. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier selection handoff`

Falsifiable prediction: A packet that preserves the model syntax while changing the error-budget row is rejected by the carrier classifier whenever the mismatch ledger or observation-window route is not preserved.

Independence witness: realityconstrainedmodelaudit, finiteobservationresidueboundary, openfitpacket, modelselectionboundary, realityconstrainedfailuresurface: none is bijective because RealityConstrainedMod carries source traces, error budget, observation window, mismatch ledger, failure surface, transport, route, provenance, and local naming in one packet. The model audit lacks the bounded error row, the residue boundary lacks the mismatch-budget envelope, the open-fit packet lacks the audit error ledger, the model-selection boundary lacks the source-window budget, and the failure surface lacks the accepted mismatch route.

The Lean follow-up must provide `Nontrivial RealityConstrainedModelErrorBudgetUp` and `FieldFaithful RealityConstrainedModelErrorBudgetUp` instances in `lean4/BEDC/Derived/RealityConstrai`
Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetTasteGate single carrier alignment`

Theorem 1184.7 (Reality-constrained model error-budget public certificate)**.** *The public bounded-error-budget certificate exported by* $\mathsf{RealityConstrainedModelErrorBudget}^{\uparrow}$ *consists exactly of the NameCert obligation surface for* $S, E, O, M, F, H, C, P, N$*, the nonescape restriction, observation-window exactness, failure visibility, selection handoff, the falsifiable prediction for classifier rejection, and the independence witness separating the sibling audit, residue, fit, selection, and failure-surface packets. Any public consumer of the certificate must preserve the source trace, declared budget, observation window, mismatch ledger, visible failure surface, transport, replay, provenance, and local naming rows as one finite packet.*

Proof. Start with the NameCert obligations and nonescape theorem. They make $S, E, O, M, F, H, C, P, N$ the whole bounded-error-budget surface and forbid a public consumer from preserving only model syntax while changing the error evidence.

Use observation-window exactness and failure visibility next. The mismatch ledger remains tied to the displayed source, budget, and observation window, and the failure row stays visible whenever the bounded packet is exported.

Apply the selection handoff to the downstream $\mathsf{ModelSelectionBoundary}^{\uparrow}$ consumer. It receives the same finite packet rather than a detached score, so public export preserves the selection-facing route already present in the carrier.

Finally include the falsifiable prediction and independence witness. The prediction exposes the classifier rejection test for changed budget or window rows, while the witness separates this packet from its audit, residue, fit, selection, and failure-surface siblings. Together these clauses form the

public bounded-error-budget certificate and add no unbounded validation, complete calibration, final model truth, or bridge claim. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier public certificate`

Theorem 1184.8 (Reality-constrained model error-budget bridge interface)**.** *Every public consumer bridge for an accepted* RealityConstrainedModelErrorBudget$^{\uparrow}$ *packet preserves the displayed observation-window row O, failure-visibility row F, model-selection handoff route, componentwise transport row H, displayed replay row C, provenance row P, and local* NameCert *row N together with the source trace, error-budget row, and mismatch ledger. The bridge exposes no hidden model oracle, no off-window observation repair, no detached scoring channel, and no unbounded validation row outside*

$$S, E, O, M, F, H, C, P, N.$$

Proof. Start with the public certificate of Definition 1184.7. It already identifies $S, E, O, M, F, H, C, P, N$ as the whole bounded-error-budget surface consumed by public readers.

Use observation-window exactness, failure visibility, and selection handoff. These force the mismatch ledger to remain tied to S, E, O, keep F visible whenever the packet is exported, and send every ModelSelectionBoundary$^{\uparrow}$ read through the same finite rows rather than through detached scoring.

Transport, replay, provenance, and naming do not add a hidden coordinate. Rows H, C, P, N preserve and source the displayed bounded-error packet, so any bridge that claims this public interface must preserve the same rows and cannot expose a model oracle outside the carrier. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedModelErrorBudgetUp.RealityConstrainedModelErrorBudgetCarrier bridge interface`

Closure status for NameCert$_{\text{RealityConstrainedModelErrorBudget}^{\uparrow}}$.

Constructive story: A RealityConstrainedModelErrorBudget$^{\uparrow}$ packet is generated from finite source traces, an error-budget row, an observation-window row, a mismatch ledger, a failure surface, componentwise hsame transports, a displayed Cont route, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedModelErrorBudget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1184.1, Definition 1184.2, and Definition 1184.3 bind bounded model-error evidence to finite source traces, observation windows, mismatch ledgers, failure surfaces, transport, route, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedModelErrorBudget}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not assert model truth, unbounded validation, complete statistical calibration, final model selection, Lean verification, axiom-clean audit, or bridge checking for RealityConstrainedModelErrorBudget$^{\uparrow}$.

Upgrade path: No further paper-side theory upgrade is pending for the finite bounded-error packet, including its observation window, failure visibility, selection handoff, transport, replay, provenance, and local naming rows.

Chapter 1185

A Concrete Naming Certificate for RealityConstrainedScientificMethodology$^{\uparrow}$

The RealityConstrainedScientificMethodology$^{\uparrow}$ packet realizes the methodology consequences of section 9414.46 as one composite finite NameCert carrier. It sits after the methodology ledger of chapter 1145, the physical model audit of chapter 1129, the explanation packet of chapter 1136, and the idealization residue seal of chapter 1170. Those siblings expose ledger, audit, explanation, and idealization rows separately; this carrier binds them with an effective-domain row, a domain-leakage failure surface, audit routing, provenance, and a local name before a scientific-methodology claim may be read as accepted. As an ai packet, it bundles methodology-ledger, physical-audit, explanation, idealization-residue, and open-fit surfaces into one methodology consumer ledger.

Definition 1185.1 (Reality-constrained scientific-methodology carrier)**.** *A* RealityConstrainedScientificMethodology *carrier is a finite* BHist *packet*

$$M = (X, A, E, I, D, L, F, R, T, C, P, N)$$

with the following displayed rows:

1. *X is the methodology-ledger row inherited from* RealityConstrainedMethodologyLedger$^{\uparrow}$*, naming scope, domain-leakage refusals, extension routes, and failure records.*
2. *A is the physical-model-audit row inherited from* PhysicalModelAudit$^{\uparrow}$*, naming source traces, observed signatures, model signatures, comparison, tower, stability, descent, ledger, failure, and strength rows.*
3. *E is the explanation-certificate row inherited from* RealityConstrainedExplanation$^{\uparrow}$*, naming observation bundle, model, phenomenon, signature comparison, descent, residue, and failure surface.*
4. *I is the idealization-residue row inherited from* ScientificIdealizationResidueSeal$^{\uparrow}$*, naming the idealization arrow, removed source material, descent row, failure surface, and safe scope.*
5. *D is the effective-domain row, naming the observation bundle, tower region, admissible consumers, and physical-signature interface over which the methodology packet is licensed.*
6. *L is the domain-leakage row, naming attempted exports beyond D and the finite refusal or routing record for each attempted export.*
7. *F is the methodology failure-surface row, collecting explanation failure, audit mismatch, active idealization residue, domain leakage, and open physical-fit defeat.*

8. *R is the audit-route row, recording how a consumer reaches the accepted methodology read through explanation, audit, idealization, domain, leakage, and failure rows.*
9. *T records componentwise* hsame *transport for $X, A, E, I, D, L, F, R, C, P, N$.*
10. *C records displayed* Cont *replay of the accepted methodology route.*
11. *P records* Pkg *provenance over the sibling carriers,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}}$ *row.*

The carrier has no coordinate for bare explanatory syntax, detached model truth, global realism, unledgered idealization, unbounded effective exactness, or methodology export outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RealityConstrainedScientificMethodologyUp.TasteGate.RealityConstrainedScientificMe single carrier alignment`

Theorem 1185.2 (Reality-constrained scientific-methodology obligation surface)**.** *For the carrier of Definition 1185.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission: the accepted object is exactly $M = (X, A, E, I, D, L, F, R, T, C, P, N)$;*
2. *explanation accountability: explanation-to-truth-cert consumers must pass through the explanation row E and its residue and failure surface;*
3. *audit accountability: physical-model reads must pass through A and the accepted audit route;*
4. *idealization accountability: ideal endpoint reads must pass through the idealization residue row I;*
5. *domain-leakage accountability: every export beyond the effective-domain row D must be recorded in L and exposed to F;*
6. *route and naming accountability: accepted consumers must preserve R, T, C, P, N as the audit route, componentwise transport, replay, provenance, and local name of the same packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedScientificMethodologyUp.RealityConstrainedScientificMethodology obligation surface`

Proof. Carrier admission is projection from Definition 1185.1. The packet lists the only rows available to the local methodology certificate.

Explanation accountability is carried by E. A consumer that reads explanation toward truth without the observation, model, phenomenon, signature comparison, descent, residue, and failure-surface material of E has not passed through the accepted explanation certificate.

Audit and idealization accountability are carried by A and I. The audit row records the finite physical-model comparison and its strength, while the idealization row records the ideal endpoint together with residue, descent, failure, and scope.

Domain-leakage accountability is carried by D, L, F. The effective-domain row names the licensed region, the leakage row records attempted exports beyond it, and the failure surface makes those refusals visible to the consumer.

Finally R, T, C, P, N preserve the same displayed route. Since the carrier contains no further coordinate, a methodology consumer cannot bypass explanation, audit, idealization residue, domain

leakage, or local naming while remaining an accepted $\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}$ read. □

Theorem 1185.3 (Reality-constrained scientific-methodology consumer factorization)**.** *Every accepted explanation-to-truth-cert consumer of a* $\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}$ *packet factors through either the explanation certificate row E or the idealization residue row I, and in both cases preserves the effective-domain row D, leakage row L, failure surface F, audit route R, transport T, replay C, provenance P, and local name N.*

Proof. Read the obligation surface of Definition 1185.2. An explanation-to-truth-cert consumer is admitted only when explanation accountability or idealization accountability is displayed.

In the explanation branch, the route uses E and therefore carries the observation bundle, model row, phenomenon row, signature comparison, descent, residue, and failure surface. In the idealization branch, the route uses I and therefore carries the idealization arrow, residue ledger, descent, failure surface, and safe scope.

Both branches still belong to the same methodology packet. The domain, leakage, failure, audit-route, transport, replay, provenance, and local-name rows D, L, F, R, T, C, P, N remain visible, so neither branch can become a detached truth-cert consumer. □

Falsifiable prediction: If the methodology packet is atomic, then every accepted explanation-to-truth-cert consumer must factor through either the explanation certificate row or the idealization residue row; a consumer bypassing both refutes the chapter.

Independence witness: realityconstrainedmethodologyledger, physicalmodelaudit, realityconstrainedexplanation, scientificidealizationresidueseal, openphysicalfit: the methodology carrier has both explanation and idealization-residue rows plus a domain-leakage failure surface, while each sibling omits at least one of those rows, so no row-preserving bijection can identify the carriers.

The packet composes realityconstrainedmethodologyledger, physicalmodelaudit, realityconstrained-explanation, scientificidealizationresidueseal, and openphysicalfit into the displayed methodology surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}$ packet is generated from methodology-ledger, physical-audit, explanation, idealization-residue, effective-domain, domain-leakage, failure-surface, audit-route, transport, replay, provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.RealityConstrainedScientificMethodologyUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1185.1, Definition 1185.2, and Definition 1185.3 bind the finite carrier, obligation surface, and explanation-or-idealization consumer factorization.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealityConstrainedScientificMethodologyUp.TasteGate.RealityConstrainedScientificMethodo`

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert global model truth, unbounded realism, final empirical closure, public closure, or bridge checking for $\mathsf{RealityConstrainedScientificMethodology}^{\uparrow}$.

Upgrade path: Scoped closure requires downstream examples showing that truth-cert and open-fit consumers preserve the explanation-or-idealization factorization through the displayed domain-leakage surface.

Chapter 1186

A Concrete Naming Certificate for $\mathsf{EmptyFableMachine}^{\uparrow}$

The $\mathsf{EmptyFableMachine}^{\uparrow}$ interface is the finite NameCert packet for the Empty Fable Machine of chapter 9399. It sits above the boundary packet of chapter 878, the local clock packet of chapter 623, and the observer-history identity packet of chapter 479. Its carrier keeps the empty-boundary source, selector-step witness, trace row, fable ledger, transport, continuation replay, provenance, and local naming row in one displayed tuple.

Definition 1186.1 (EmptyFableMachine NameCert carrier)**.** *An* $\mathsf{EmptyFableMachine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Z, S, T, L, H, C, P, N)$$

with the following displayed rows:

1. *Z is the empty-boundary source row, read through the boundary discipline of chapter 878;*
2. *S is the selector-step witness row, including the enacted mark, its displayed* Ext *edge, and the local branch certificate;*
3. *T is the finite trace row read from the constructor spine generated by the displayed selector step;*
4. *L is the fable-ledger row identifying the same trace as the Empty-fable story ledger;*
5. *H records componentwise* hsame *transport for Z, S, T, L and for the local naming row;*
6. *C records the displayed* Cont *replay routes by which a consumer reads the source, selector witness, trace, ledger, transport, and local name;*
7. *P is the* Pkg *provenance over* $\mathsf{EmptyFableMachine}^{\uparrow}$, $\mathsf{FableMachineBoundary}^{\uparrow}$, $\mathsf{FableMachineClock}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, BHist, Ext, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{EmptyFableMachine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these rows. It has no coordinate for an undisplayed selector function, observer-substance, host trace quotient, global clock, or truth oracle.

Theorem 1186.2 (EmptyFableMachine NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{EmptyFableMachine}^{\uparrow}}$ *surface for the carrier of Definition 1186.1 consists exactly of carrier admission for $E = (Z, S, T, L, H, C, P, N)$, empty-boundary admission for Z, selector-step admission for S, trace admission for T, fable-ledger admission for L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and the local naming row N.*

The boundary row is supplied by chapter 878; the step-order reading is supplied by chapter 623; and the observer-facing history read is constrained by chapter 479. The chapter therefore binds the source, selected branch, trace, ledger, transport, replay, provenance, and naming obligations without adding a hidden chooser.

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.EmptyFableMachineTasteGate single carrier alignment`

Proof. Project the carrier rows. The empty-boundary source Z, selector witness S, trace row T, fable ledger L, transport row H, continuation row C, provenance row P, and local name N are separate coordinates of the finite packet.

Read Z through the boundary packet of chapter 878. Read the local order and enacted selected mark through chapter 623. Read the observer-facing history discipline through chapter 479.

The trace row and fable-ledger row then refer to the same displayed constructor spine named in chapter 9399. The transport, continuation, package, and local naming rows repack that spine for consumers, so no further selector, subject, quotient, clock, or oracle coordinate is introduced. □

Theorem 1186.3 (EmptyFableMachine selector-ledger non-escape)**.** *Every public selector-ledger read of an accepted* $\mathsf{EmptyFableMachine}^{\uparrow}$ *packet factors through*

$$Z, S, T, L, H, C, P, N.$$

In particular, a branch mark can be exposed only with its displayed selector-step witness S, its trace row T, and its fable-ledger row L. The packet exports no branch mark detached from selector evidence, no hidden chooser, no observer-substance coordinate, no global clock, and no host trace equality.

Lean4 ✓ `BEDC.Derived.EmptyFableMachineUp.Ledger.EmptyFableMachine selector ledger nonescape`

Proof. Inspect the accepted carrier. The only branch-facing row is S, and that row contains the selected mark together with the displayed Ext edge and local branch certificate.

Trace and ledger reads are confined to T and L. The hsame transport row H and continuation row C move only those displayed rows, while the provenance row P names the boundary, clock, and observer-history identity surfaces that license the read.

A public branch mark without S would require an additional selector coordinate. No such coordinate appears in $E = (Z, S, T, L, H, C, P, N)$, so every selector-ledger read remains inside the finite NameCert packet. □

Falsifiable prediction: A public selector-ledger read that exposes a branch mark without a displayed selector-step witness refutes the EmptyFableMachineUp packet.

Independence witness: fablemachineboundary, fablemachineclock, observerhistidentity, localitycell: none is bijective because EmptyFableMachineUp simultaneously carries empty-boundary source, selector witness, trace ledger, fable ledger, transport, continuation, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EmptyFableMachine}^{\uparrow}}$.

Constructive story: An $\mathsf{EmptyFableMachine}^{\uparrow}$ packet is generated from an empty-boundary source row, selector-step witness, trace row, fable-ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmptyFableMachine}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite carrier, NameCert obligation surface, selector-ledger non-escape route, falsifiable selector-witness condition, and independence surface for Empty-fable machine packets over BHist, Ext, trace, ledger, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{EmptyFableMachine}\uparrow}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not close Nontrivial or FieldFaithful Lean instances, bridge checking, global branch selection, host trace equality, hidden observer substance, shared-clock semantics, or truth-oracle semantics.
Upgrade path: Lean follow-up must provide Nontrivial and FieldFaithful instances in lean4/BEDC/Derived/EmptyFableMachineUp/Taste with the canonical target expected at BEDC.Derived.EmptyFableMachineUp.taste_gate.

Chapter 1187

A Concrete Naming Certificate for $\mathsf{RecursorClosureClassifierAudit}^{\uparrow}$

The $\mathsf{RecursorClosureClassifierAudit}^{\uparrow}$ packet realizes the recursor-as-authorized-generator view of chapter 9373 as a finite NameCert surface that joins recursor authorization, classifier closure, and replayable audit execution. It consumes the recursor-induced output packet of chapter 522, the authorization packet of chapter 529, the generator packet of chapter 534, the branch-audit packet of chapter 974, and the classifier-membership packet of chapter 476. Those siblings expose the recursor output, legal eliminator call, generator boundary, constructor-indexed branch coverage, and type-classifier membership rows; this packet binds them into one audit replay carrier whose classifier must close over the same displayed branches.

Definition 1187.1 (Recursor closure classifier audit carrier)**.** *A* $\mathsf{RecursorClosureClassifierAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, G, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the recursor signature and motive row, read through the accepted inductive signature and motive boundary;*
2. *G is the* $\mathsf{RecursorGenerator}^{\uparrow}$ *row recording the legal eliminator call and its structural generator boundary;*
3. *B is the* $\mathsf{RecursorBranchAudit}^{\uparrow}$ *branch matrix, with one displayed branch row for every constructor of the accepted signature;*
4. *K is the classifier-closure row, carrying the* $\mathsf{TypeClassifierMembership}^{\uparrow}$ *membership surface and the output classifier reached by the recursor call;*
5. *A is the audit-replay row, recording the kernel replay, source axiom gate, and forbidden-dependency purity check for the generated output;*
6. *H records componentwise* hsame *transport for S, G, B, K, A and for the local naming row;*
7. *C records the displayed* Cont *replay routes by which consumers read the recursor, branch, classifier, audit, transport, and local name;*
8. *P is the* Pkg *provenance over* $\mathsf{RecursorInducedNameCert}^{\uparrow}$, $\mathsf{RecursorAuthorization}^{\uparrow}$, $\mathsf{RecursorGenerator}^{\uparrow}$, $\mathsf{RecursorBranchAudit}^{\uparrow}$, $\mathsf{TypeClassifierMembership}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RecursorClosureClassifierAudit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an omitted constructor case, a detached classifier proof, a hidden host type checker, an unreported dependency trace, a second branch package, or an audit result outside $S, G, B, K, A, H, C, P, N$.

Theorem 1187.2 (Recursor closure classifier audit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RecursorClosureClassifierA}}$ *surface for the carrier of Definition 1187.1 consists exactly of carrier admission for $R = (S, G, B, K, A, H, C, P, N)$, recursor authorization through S and G, constructor-complete branch coverage through B, classifier-closure preservation through K, audit replay through A, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and the local naming row N.*

Consequently, any accepted output must be readable both as a legal recursor-generated NameCert and as a classifier-closed output whose audit can be replayed from the same finite branch package. A packet whose branch package omits one constructor case is rejected by the classifier audit, even if its remaining recursor, classifier, and audit rows are individually well formed.

Lean4 ✓ `BEDC.Derived.RecursorClosureClassifierAuditUp.TasteGate.RecursorClosureClassifierAuditTasteGate`
`single carrier alignment`

Proof. Project the carrier rows. The signature and motive row S, generator row G, branch matrix B, classifier-closure row K, audit replay row A, transport row H, continuation row C, provenance row P, and local name N are the only available coordinates.

Read S and G through chapter 529 and chapter 534. They license the eliminator call but do not by themselves expose the per-constructor branch matrix or the classifier membership row.

Next read B through chapter 974 and K through chapter 476. Branch coverage and classifier membership are therefore checked on the same displayed recursor route rather than on separate host judgments.

Finally use the audit row A and the recursor-induced output packet of chapter 522. The replay must see the same branch package and classifier-closure row named in the carrier. If a constructor case is missing from B, the replay has no row from which to certify that branch, so the local classifier rejects the packet as a $\mathsf{RecursorClosureClassifierAudit}^{\uparrow}$ certificate. □

Falsifiable prediction: A recursor packet whose branch package omits one constructor case is rejected by the classifier audit, even when the remaining branch rows, classifier row, and audit-replay row are present.

Independence witness: recursorinducednamecert, recursorauthorization, recursorgenerator, recursorbranchaudit, typeclassifiermembership: none alone carries the joined audit replay and classifier-closure rows. The recursor-induced packet names generated outputs, authorization licenses the eliminator call, the generator packet exposes the structural generator boundary, the branch-audit packet exposes constructor coverage, and the classifier-membership packet exposes type-classifier membership; only RecursorClosureClassifierAuditUp requires all of them in one $S, G, B, K, A, H, C, P, N$ carrier.

The Lean follow-up must provide `Nontrivial RecursorClosureClassifierAuditUp` and `FieldFaithful RecursorClosureClassifierAuditUp` in `lean4/BEDC/Derived/RecursorClosureClassifierAuditUp/TasteGat` with the canonical alignment target expected at `BEDC.Derived.RecursorClosureClassifierAuditUp.TasteGate`

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorClosureClassifierAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RecursorClosureClassifierAudit}^{\uparrow}$ packet is generated from a recursor signature and motive row, legal generator row, constructor-complete branch matrix, classifier-closure row, audit-replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RecursorClosureClassifierAudit}^{\uparrow}}$, seed)

Scope closed: The seed scope records the finite carrier, NameCert obligation surface, branch-omission rejection, and independence surface for recursor closure classifier audits over recursor authorization, generator, branch audit, classifier membership, audit replay, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RecursorClosureClassifierAudit}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Nontrivial or FieldFaithful Lean instances, bridge checking, arbitrary host typing semantics, hidden branch completion, detached classifier proofs, or audit results outside the displayed replay row.

Upgrade path: Lean follow-up must provide Nontrivial and FieldFaithful instances in lean4/BEDC/Derived/RecursorClosureClassifierAudit with the canonical target expected at BEDC.Derived.RecursorClosureClassifierAuditUp.TasteGate.RecursorClosureClassifierAuditTasteGate_si

Chapter 1188

A Concrete Naming Certificate for $\mathsf{LargeModelOutputVerifier}^{\uparrow}$

The $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ packet realizes the externally checked output-channel boundary of chapter 9398 as finite BEDC data. It consumes the checker-facing harness of chapter 976, the consumer-facing audit channel of chapter 969, the inference-output trace of chapter 972, the prompt/output record of chapter 965, and the inscription-audit packet of chapter 1027. Its distinguishing role is the public verifier boundary: a model output is exported only after verifier, proof-check, refusal, route, provenance, and local NameCert rows are displayed together.

Definition 1188.1 (Large-model output verifier carrier)**.** *A* $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (H, A, M, T, I, Q, C, F, R, K, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{VerifiedOutputHarness}^{\uparrow}$ *row carrying the checker-facing accepted-output harness;*
2. *A is the* $\mathsf{LargeModelAuditChannel}^{\uparrow}$ *row carrying the consumer-facing audit channel;*
3. *M is the* $\mathsf{ModelTrace}^{\uparrow}$ *row recording the finite inference-output trace;*
4. *T is the* $\mathsf{PromptResponseTrace}^{\uparrow}$ *row recording the prompt and response boundary;*
5. *I is the* $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$ *row binding the output to inscription-audit provenance;*
6. *Q is the verifier row naming the external checker, checker input, and displayed checker result;*
7. *C is the proof-check row recording the proof object, proof checker, or finite validation transcript consumed by Q;*
8. *F is the refusal row recording failed verification, unavailable evidence, policy refusal, or obstruction disclosure;*
9. *R is the route row through which an accepted output is exported to a public consumer;*
10. *K is the displayed* Cont *replay through $H, A, M, T, I, Q, C, F, R$;*
11. *P is the* Pkg *provenance over* $\mathsf{VerifiedOutputHarness}^{\uparrow}$, $\mathsf{LargeModelAuditChannel}^{\uparrow}$, $\mathsf{ModelTrace}^{\uparrow}$, $\mathsf{PromptResponseTrace}^{\uparrow}$, $\mathsf{LargeModelInscriptionAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LargeModelOutputVerifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden model truth, private verifier correctness, semantic omniscience, or output export without the verifier and proof-check rows.

Lean4 (definition) `BEDC.Derived.LargeModelOutputVerifierUp.LargeModelOutputVerifierUp`
Lean4 ✓ `BEDC.Derived.LargeModelOutputVerifierUp.TasteGate.LargeModelOutputVerifierTasteGate`
`single carrier alignment`

Definition 1188.2 (Large-model output verifier classifier)**.** *Two* $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ *carriers classify together exactly when their harness, audit-channel, model-trace, prompt-response trace, inscription-audit, verifier, proof-check, refusal, route, replay, provenance, and local naming rows compare componentwise by the displayed* hsame *entries. Thus the classifier is the finite componentwise relation*

$$V \equiv V' \quad \textit{iff} \quad H\mathsf{hsame}H',\ A\mathsf{hsame}A',\ M\mathsf{hsame}M',\ T\mathsf{hsame}T',\ I\mathsf{hsame}I',$$
$$Q\mathsf{hsame}Q',\ C\mathsf{hsame}C',\ F\mathsf{hsame}F',\ R\mathsf{hsame}R',\ K\mathsf{hsame}K',\ P\mathsf{hsame}P',\ N\mathsf{hsame}N'.$$

It does not compare model meanings outside the displayed verification packet.

Theorem 1188.3 (Large-model output verifier NameCert obligations)**.** *For the carrier of Definition 1188.1, the local* $\mathsf{NameCert}_{\mathsf{LargeModelOutputVerifier}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $V = (H, A, M, T, I, Q, C, F, R, K, P, N)$*; sibling exposure through* H, A, M, T, I*; verifier and proof-check exposure through* Q, C*; refusal and route exactness through* F, R*; replay and provenance exactness through* K, P*; and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.LargeModelOutputVerifierUp.LargeModelOutputVerifierNameCertObligations`

Proof. Project the carrier of Definition 1188.1. The admitted packet displays the five sibling rows H, A, M, T, I together with verifier, proof-check, refusal, route, replay, provenance, and local-name rows.

The sibling exposure row binds the accepted output to the harness, audit channel, model trace, prompt-response trace, and inscription audit. These are exactly the finite surfaces supplied by chapter 976, chapter 969, chapter 972, chapter 965, and chapter 1027.

The verifier row Q and proof-check row C are then read before route export. Refusal F remains visible whether verification succeeds or blocks, so the packet cannot erase failed checking as if it were accepted evidence.

Finally K, P, and N replay the same displayed rows as continuation, package provenance, and local naming. This gives exactly the six obligation rows and no hidden checker or model-truth coordinate. □

Theorem 1188.4 (Large-model output verifier public acceptance boundary)**.** *Every public export from an accepted* $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ *packet must pass through the verifier row* Q *and proof-check row* C *after the sibling rows* H, A, M, T, I *and before the route row* R*. A route that exports the output without* Q *and* C *is not a* $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ *acceptance route.*

Lean4 ✓ `BEDC.Derived.LargeModelOutputVerifierUp.LargeModelOutputVerifierUp StdBridge`

Proof. Use the NameCert obligations of Definition 1188.3. They require sibling exposure, verifier exposure, proof-check exposure, refusal exposure, route exactness, replay, provenance, and local naming before a public output is exported.

The route row R is therefore downstream of both Q and C. If either row is absent, the route still may describe an emitted model output, but it does not describe an externally checked acceptance packet.

Because the carrier has no coordinate for private verifier correctness or hidden semantic truth, no alternative row can replace Q and C. Public acceptance is exactly the displayed verifier/proof-check route. □

Theorem 1188.5 (Large-model output verifier ledger non-escape)**.** *Let*

$$V = (H, A, M, T, I, Q, C, F, R, K, P, N)$$

be an accepted $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ *packet. Any consumer that reads the exported model output through R must also read the sibling rows H, A, M, T, I, the verifier and proof-check rows Q, C, the refusal row F, the replay row K, the provenance row P, and the local name N. Omitting one of these rows exits the verifier packet rather than exporting an accepted output.*

Proof. Apply Definition 1188.3. The local NameCert surface lists all rows required for carrier admission and public export.

The proof-check row cannot be detached from the verifier row, since C records the finite checking transcript consumed by Q. The refusal row cannot be detached from the route, since failed or unavailable evidence must remain visible at the same boundary.

Replay and provenance then bind the consumer route to the displayed packet. Thus an export that omits a sibling row, verifier row, proof-check row, refusal row, replay row, provenance row, or local name is not an accepted $\mathsf{LargeModelOutputVerifier}^{\uparrow}$ export. □

Theorem 1188.6 (Large-model output verifier obligation closure package)**.** *Every accepted* LargeModelOutputVerif *packet has exactly the obligation package consisting of carrier exposure for $V = (H, A, M, T, I, Q, C, F, R, K, P, N)$, classifier accountability, sibling exposure through H, A, M, T, I, verifier and proof-check exposure through Q, C, refusal and route exactness through F, R, replay and provenance through K, P, local naming through N, and the TasteGate single-carrier alignment for the same finite carrier. It exports no hidden model truth, private verifier correctness, host semantic understanding, or output route without verifier and proof-check rows.*

Proof. Combine the local NameCert obligations of Definition 1188.3 with the carrier and classifier definitions. The carrier exposes exactly the rows $H, A, M, T, I, Q, C, F, R, K, P, N$, while the classifier accounts for them componentwise and does not compare hidden model meanings.

Apply the public boundary of Definition 1188.4. It forces public export through Q and C before the route row R, so no output route qualifies as accepted without the verifier and proof-check rows.

Apply the ledger non-escape theorem Definition 1188.5. It keeps the sibling, refusal, replay, provenance, and local naming rows visible to any consumer of the exported output.

Finally use the checked TasteGate marker

$$\mathsf{LargeModelOutputVerifierTasteGate_single_carrier_alignment}$$

for the same finite carrier. The combined package is therefore an obligation surface only; it does not add hidden model truth, private checker correctness, or host semantic understanding. □

Falsifiable prediction: An accepted LargeModelOutputVerifierUp packet cannot export a model output without displayed verifier and proof-check rows.

Independence witness: verifiedoutputharness, largemodelauditchannel, modeltrace, promptresponsetrace, largemodelinscriptionaudit: none is bijective to $\mathsf{LargeModelOutputVerifier}^{\uparrow}$. Omitting the verifier row Q gives a harness or audit-channel acceptance surface without external checking; omitting the proof-check row C gives an audited output without a check transcript; omitting

the refusal row F erases blocked evidence; omitting H, A, M, T, I loses the sibling packet route; omitting P, N loses provenance and local naming.

The Lean TasteGate.lean surface must provide Nontrivial LargeModelOutputVerifier$^{\uparrow}$ and FieldFaithful LargeModel instances for the same finite carrier.

Theorem 1188.7 (Large-model output verifier bridge surface)**.** *The* LargeModelOutputVerifier$^{\uparrow}$ *bridge surface is the finite packet that simultaneously displays the verifier carrier of Definition 1188.1, the public acceptance boundary of Definition 1188.4, the ledger non-escape route of Definition 1188.5, the obligation package of Definition 1188.6, the TasteGate marker, and the checked standard bridge marker* LargeModelOutputVerifierUp_StdBridge*. A public output export is therefore bridged only when the verifier, proof-check, refusal, route, replay, provenance, local naming, TasteGate, and standard bridge rows are read together.*

Proof. Start with the carrier and public acceptance boundary. The carrier exposes $H, A, M, T, I, Q, C, F, R, K, P, N$ and the public boundary forces every accepted export through the verifier row Q and proof-check row C before the route row R.

Apply the ledger non-escape theorem. It keeps sibling, verifier, proof-check, refusal, replay, provenance, and local naming rows visible to every consumer of the exported output.

Apply the obligation closure package. It binds classifier accountability, sibling exposure, verifier and proof-check exposure, refusal and route exactness, replay, provenance, local naming, and the TasteGate single-carrier alignment for the same finite packet.

The checked standard bridge marker names the same public route as a bridge surface. Thus the bridge does not add hidden model truth or private checker correctness; it records that the displayed verifier packet is read through the standard bridge boundary. □

Closure status for NameCert$_{\text{LargeModelOutputVerifier}^{\uparrow}}$.

Constructive story: A LargeModelOutputVerifier$^{\uparrow}$ packet is generated from harness, audit-channel, model-trace, prompt-response, inscription-audit, verifier, proof-check, refusal, route, replay, provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.LargeModelOutputVerifierUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{LargeModelOutputVerifier}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1188.1, Definition 1188.2, Definition 1188.3, Definition 1188.4, Definition 1188.5, Definition 1188.6, and Definition 1188.7 bind verifier and proof-check rows, refusal and route rows, replay and provenance rows, local naming, sibling surfaces, TasteGate, and the checked standard bridge surface for externally checked model-output export.

Formal verification: FormalStatus(NameCert$_{\text{LargeModelOutputVerifier}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LargeModelOutputVerifierUp.TasteGate.LargeModelOutputVerifierTasteGate\_single\_carrier`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close model consciousness, corpus truth, private verifier correctness, host semantic understanding, or global proof checking for LargeModelOutputVerifier$^{\uparrow}$.

Upgrade path: No further paper-side bridge step is claimed here; stronger closure would require a global proof-checking interface beyond this verifier packet.

Chapter 1189

A Concrete Naming Certificate for $\mathsf{SynchronizationCertificate}^{\uparrow}$

The $\mathsf{SynchronizationCertificate}^{\uparrow}$ packet realizes the local synchronization certificate of section 9414.34 as finite NameCert data. It sits between the one-history stream surface of chapter 603 and the refusal boundary of chapter 1003: two local time rows may be compared only through displayed correspondences, locality ledgers, transport rows, and no-global-clock refusal. The sibling context is supplied by $\mathrm{MultiHistConfig}^{\uparrow}$ in chapter 474, $\mathsf{ObserverFrameRefusal}^{\uparrow}$ in chapter 894, and $\mathsf{AnchorChangeInvariant}^{\uparrow}$ in chapter 1166.

Definition 1189.1 (Synchronization certificate carrier)**.** *A* $\mathsf{SynchronizationCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H_i, H_j, T_i, T_j, R, L, S, C, P, N)$$

with the following displayed rows:

1. H_i *is the source observer-history row drawn from the finite multi-history configuration;*
2. H_j *is the target observer-history row drawn from the same finite multi-history configuration;*
3. T_i *is the local* $\mathsf{HistTimeStream}^{\uparrow}$ *row for* H_i*;*
4. T_j *is the local* $\mathsf{HistTimeStream}^{\uparrow}$ *row for* H_j*;*
5. R *is the selected local-step correspondence ledger, recording only the displayed pairs of steps from* T_i *and* T_j *that the comparison relates;*
6. L *is the locality ledger recording the local read sites, admissible comparison routes, and the refusal of any unlisted step match;*
7. S *is the no-global-clock refusal row, read through* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *and compatible with the observer-frame refusal surface;*
8. C *records componentwise* hsame *transport and displayed* Cont *replay for* $H_i, H_j, T_i, T_j, R, L, S, P, N$*;*
9. P *is the* Pkg *provenance over* $\mathsf{SynchronizationCertificate}^{\uparrow}$, $\mathsf{HistTimeStream}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{ObserverFrameRefusal}^{\uparrow}$, $\mathsf{AnchorChangeInvariant}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{SynchronizationCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a shared present, global clock, total observer-time order, host equality of histories, unlisted correspondence, or privileged frame.

Lean4 ✓ `BEDC.Derived.SynchronizationCertificateUp.SynchronizationCertificateTasteGate single carrier alignment`

Theorem 1189.2 (Synchronization certificate locality)**.** *For the carrier of Definition 1189.1, every accepted* $\mathsf{SynchronizationCertificate}^{\uparrow}$ *read compares the two histories only through the displayed packet*

$$H_i, H_j, T_i, T_j, R, L, S, C, P, N.$$

The rows T_i *and* T_j *remain local* $\mathsf{HistTimeStream}^{\uparrow}$ *rows,* R *selects the admitted local-step pairs, and* L, S *prevent the comparison from becoming a shared clock.*

Proof. Project the accepted packet. The rows H_i, H_j expose the two observer histories through the same finite multi-history surface, while T_i, T_j expose their local time streams as two separate $\mathsf{HistTimeStream}^{\uparrow}$ rows.

Read the correspondence and locality rows next. The ledger R lists the selected step pairs, and L records the local read sites and rejects comparisons outside that finite list.

Finally use the no-global-clock refusal row S together with the transport and replay row C. These rows let a consumer transport and replay the displayed local comparison, but they do not add a shared present, total observer-time order, or privileged frame. □

Lean4 ✓ `BEDC.Derived.SynchronizationCertificateUp.SynchronizationCertificate locality`

Theorem 1189.3 (Synchronization certificate nonescape)**.** *An accepted* $\mathsf{SynchronizationCertificate}^{\uparrow}$ *consumer has no route to a global clock, privileged shared-present row, detached observer frame, host equality of histories, or unlisted local-step match. Any read requiring one of those rows exits the packet rather than reading the local synchronization certificate.*

Proof. Use the carrier definition. The only positive comparison data are the two local time rows T_i, T_j and the selected correspondence ledger R.

The refusal row S is part of the same packet as the positive comparison data. By chapter 1003 and chapter 894, a consumer may cite that refusal as displayed boundary data, but cannot promote it to a hidden synchronizer.

The rows C, P, N then preserve the finite packet by componentwise transport, continuation replay, provenance, and local naming. Since no row names a shared present or host clock, the requested external object has no projection from the accepted carrier. □

Theorem 1189.4 (Synchronization certificate sibling independence)**.** *The* $\mathsf{SynchronizationCertificate}^{\uparrow}$ *carrier is not bijective to* $\mathsf{HistTimeStream}^{\uparrow}$, $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$, $\mathsf{ObserverFrameRefusal}^{\uparrow}$, *or* $\mathsf{AnchorChangeInvariant}^{\uparrow}$. *Each sibling omits at least one of the paired local-time rows, the selected correspondence ledger, or the no-global-clock refusal in the same finite packet.*

Proof. Compare the rows. A $\mathsf{HistTimeStream}^{\uparrow}$ packet supplies a one-history local stream, but not the paired histories, selected correspondence ledger, and refusal row together. A $\mathrm{MultiHistConfig}^{\uparrow}$ packet supplies a family of histories, but not the two local time rows and selected local-step pairs.

The refusal siblings also differ from the present carrier. $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ and $\mathsf{ObserverFrameRefusal}^{\uparrow}$ record refusal surfaces, but they do not carry the positive paired local-time comparison ledger R as part of the same packet.

Finally $\mathsf{AnchorChangeInvariant}^{\uparrow}$ carries anchor-change invariant transport rather than selected local-step correspondences between two HistTimeStream rows. Therefore none of the listed siblings is bijective to $\mathsf{SynchronizationCertificate}^{\uparrow}$. □

Falsifiable prediction: A SynchronizationCertificateUp read fails whenever a consumer asks for a global clock or privileged shared-present row not displayed in the local synchronization packet.

Independence witness: histtimestream, multihistconfig, noglobalsyncboundary, observerframerefusal, anchorchangeinvariant: none is bijective to SynchronizationCertificateUp because each lacks either the paired local-time rows, the selected correspondence ledger, or the explicit no-global-clock refusal in the same packet.

The Lean follow-up must provide `Nontrivial SynchronizationCertificateUp` and `FieldFaithful SynchronizationCertificateUp` in `lean4/BEDC/Derived/SynchronizationCertificateUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{SynchronizationCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{SynchronizationCertificate}^{\uparrow}$ packet is generated from two observer histories, two local HistTimeStream rows, a selected local-step correspondence ledger, locality ledger, no-global-clock refusal row, componentwise transport and continuation replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynchronizationCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1189.1, Definition 1189.2, Definition 1189.3, and Definition 1189.4 bind local synchronization to the displayed paired-stream packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynchronizationCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not construct a global clock, shared present, total observer-time order, privileged frame, Lean verification, or bridge checking for $\mathsf{SynchronizationCertificate}^{\uparrow}$.

Upgrade path: Obligation closure requires Lean encoding of carrier admission, paired local-time correspondence, no-global-clock refusal, TasteGate nontriviality, and field faithfulness for $H_i, H_j, T_i, T_j, R, L, S, C, P, N$.

Chapter 1190

A Concrete Naming Certificate for MetaCICNormalizationConfluenceBridge$^{\uparrow}$

The MetaCICNormalizationConfluenceBridge$^{\uparrow}$ packet binds the normalization audit route of chapter 750 to the confluence audit route of chapter 743, then records how the subject-reduction discharge ledger of chapter 720 and the closed-consistency assembly of chapter 1177 may read the shared closed typed term boundary. It is a finite bridge packet for audit correspondence, not an unconditional closed subject-reduction theorem.

The packet is composite over the normalization-audit, confluence-audit, subject-reduction discharge-ledger, and closed-consistency assembly rows. Its bridge row records the correspondence among those sibling packets rather than presenting an atomic carrier discovery.

Definition 1190.1 (MetaCIC normalization-confluence bridge carrier)**.** *A* MetaCICNormalizationConfluenceBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (N, Q, D, A, T, H, C, P, L)$$

with the following rows:

1. *N is the* MetaCICNormalizationAudit$^{\uparrow}$ *row carrying candidate, closed-candidate, candidate-set, application adequacy, Pi adequacy, and no-infinite-reduction reads;*
2. *Q is the* MetaCICConfluenceAudit$^{\uparrow}$ *row carrying checked confluence fragments and the named confluence-boundary inputs;*
3. *D is the* SubjectReductionDischargeLedger$^{\uparrow}$ *row carrying the beta-substitution, application-argument, lambda-domain, and Pi-domain discharge surface;*
4. *A is the* ClosedConsistencyAssembly$^{\uparrow}$ *row carrying closedness, typed-normal endpoint, normal false-proof exclusion, and subject-reduction boundary exposure;*
5. *T is the closed typed term route-correspondence classifier, recording that the normalization row and confluence row speak about the same displayed closed typed term before any consistency consumer reads the packet;*
6. *H is the componentwise* hsame *transport row over N, Q, D, A, T;*
7. *C is the displayed* Cont *replay route for normalization, confluence, discharge-ledger, and assembly consumers;*
8. *P is the* Pkg *provenance row naming the four sibling packets and the audit-map source route;*

9. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}}$ *row.*

The carrier exports no host evaluator, quotient equality over reductions, hidden typed-term selector, unconditional confluence theorem, unconditional closed subject-reduction theorem, or closed consistency proof detached from $N, Q, D, A, T, H, C, P, L$.

Theorem 1190.2 (MetaCIC normalization-confluence bridge NameCert obligations)**.** *For an accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *carrier, the local* NameCert *surface consists of the following finite obligations:*

1. *carrier admission for $B = (N, Q, D, A, T, H, C, P, L)$ as a* BHist *packet;*
2. *normalization-audit row readback through N, including Candidate, closed-candidate, candidate-set, App adequacy, and Pi adequacy entries;*
3. *confluence-audit row readback through Q, including checked fragment rows and named confluence-boundary inputs;*
4. *subject-reduction discharge-ledger readback through D, preserving the four preservation-obligation rows without turning them into a closed theorem;*
5. *closed-consistency assembly readback through A, preserving the normal false-proof exclusion and subject-reduction boundary rows;*
6. *closed typed term route correspondence through T, with agreement transported componentwise by H;*
7. *displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 1190.1. The rows N, Q, D, A are already distinct sibling packets, and T is the only classifier row that compares their closed typed term routes.

Use the normalization audit obligation surface, the confluence audit obligation surface, the subject-reduction discharge-ledger obligations, and the closed-consistency assembly obligations. These sibling results supply the rows consumed by the bridge, while H, C, P, L preserve transport, replay, provenance, and local naming.

Because the bridge has no further coordinate, every accepted read is exhausted by the displayed finite packet. In particular the read cannot pass through host complex analysis, quotient equality, a hidden typed-term selector, an unconditional confluence theorem, or an unconditional closed subject-reduction theorem. □

Theorem 1190.3 (MetaCIC normalization-confluence bridge route boundary)**.** *Any accepted bridge packet whose normalization and confluence rows agree on the closed typed term through T, H, C, P, L has no route to an unconditional closed subject-reduction theorem. The packet may hand the displayed discharge-ledger row D to the closed-consistency assembly row A, but every such handoff remains inside $N, Q, D, A, T, H, C, P, L$.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationConfluenceBridgeUp.MetacicNormalizationConfluenceBridgeRouteBo`

Proof. Read the agreement first through T. It compares the normalization and confluence routes only as displayed closed typed term rows; it does not add a preservation theorem.

Next route the preservation side through D. The discharge ledger contains the four named preservation-obligation rows, so a consumer may cite those rows but cannot erase their boundary and obtain closed subject reduction for arbitrary closed proofs.

Finally pass to A. The closed-consistency assembly can read the normal false-proof exclusion and the subject-reduction boundary, but it receives them as finite sibling rows under H, C, P, L. Thus the bridge remains a route-correspondence certificate rather than an unconditional theorem. □

Theorem 1190.4 (MetaCIC normalization-confluence bridge transport obligations)**.** *For an accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *carrier* $B = (N, Q, D, A, T, H, C, P, L)$*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *transport preserve the separate bridge rows* N, Q, D, A, T*. The transported read may replay the normalization audit row* N*, the confluence audit row* Q*, the discharge-ledger row* D*, the closed-consistency assembly row* A*, and the typed-term correspondence row* T*, but it cannot collapse those rows into a subject-reduction theorem.*

Proof. Project the accepted carrier into its nine displayed coordinates. The transport row H is componentwise, so it acts on N, Q, D, A, T as separate row data rather than as a quotient over the sibling packets.

The displayed Cont row C replays the same bridge route after transport, while Pkg records the provenance of the four sibling packets. The local $\mathsf{NameCert}$ row L names the transported certificate surface and does not add a new proof coordinate.

Thus transport preserves exactly the rows already present in B. Since no coordinate of B is an unconditional subject-reduction theorem, preserving the packet cannot manufacture one. □

Theorem 1190.5 (MetaCIC normalization-confluence bridge route exactness)**.** *Every accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *bridge read is exhausted by* $N, Q, D, A, T, H, C, P, L$*. In particular, the read has no hidden typed-term selector, no quotient reduction equality, no host evaluator, no unconditional confluence theorem, no closed subject-reduction theorem, and no extra route outside the displayed packet.*

Proof. Read the bridge through the carrier definition. Its displayed rows are exactly $N, Q, D, A, T, H, C, P, L$, with T as the only row comparing the closed typed term routes and H, C, P, L as structural transport, replay, provenance, and local naming rows.

Any accepted route must therefore factor through one of those coordinates. The carrier definition exports no further selector and no equality quotient over reduction routes, so there is no place for a hidden typed-term selector or host evaluator to enter.

The same exhaustion excludes unconditional metatheorems. Confluence is read only through Q, and preservation is read only through D and the boundary row visible to A; neither read becomes a closed theorem detached from the packet. □

Theorem 1190.6 (MetaCIC normalization-confluence bridge discharge boundary)**.** *Within an accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *carrier, row* D *can be handed to row* A *only as the displayed subject-reduction discharge-ledger boundary. The handoff preserves the named discharge rows consumed by closed consistency, but it does not assert an unconditional closed subject-reduction theorem.*

Proof. The discharge side of the bridge is the row D. It carries the named preservation-obligation rows from the subject-reduction discharge ledger and remains distinct from the closed-consistency assembly row A.

The assembly row may consume D only through the displayed bridge route. This gives A access to the ledger boundary that it cites, together with the normal false-proof exclusion already internal to the assembly packet.

Because the bridge has no additional preservation coordinate, the handoff cannot erase the ledger boundary. The result is a routed discharge boundary for the assembly consumer, not an unconditional closed subject-reduction theorem. □

Theorem 1190.7 (MetaCIC normalization-confluence bridge TasteGate surface)**.** *The Lean file* `TasteGate.lean` *for* MetaCICNormalizationConfluenceBridge$^{\uparrow}$ *must provide* Nontrivial MetaCICNormalizationConflue *and* FieldFaithful MetaCICNormalizationConfluenceBridge$^{\uparrow}$ *instances. The nontrivial instance must distinguish bridge packets that agree on N, Q but differ on D or A, and the field-faithful instance must preserve the separate rows $N, Q, D, A, T, H, C, P, L$.*

Lean4 ✓ `BEDC.Derived.MetacicNormalizationConfluenceBridgeUp.MetacicNormalizationConfluenceBridgeTasteGa`
`single carrier alignment`

Proof. The carrier has four sibling packet rows and five structural rows. A TasteGate that identifies all carriers would collapse the bridge to one point and would lose the distinction between normalization, confluence, discharge-ledger, and assembly reads.

Field faithfulness is forced by the same packet shape. Preserving $N, Q, D, A, T, H, C, P, L$ separately is exactly what prevents the bridge from rereading a normalization or confluence agreement as a subject-reduction theorem or as closed consistency. □

Theorem 1190.8 (MetaCIC normalization-confluence bridge obligation closure package)**.** *The displayed rows $N, Q, D, A, T, H, C, P, L$, together with the local* NameCert *obligations, the route-boundary theorem, the TasteGate marker, the falsifiable prediction, and the independence witness below, form the complete* obligation *package for* MetaCICNormalizationConfluenceBridge$^{\uparrow}$.

Proof. Start with the carrier of Definition 1190.1. It fixes the only admissible packet shape $B = (N, Q, D, A, T, H, C, P, L)$ and states that no hidden evaluator, quotient equality, typed-term selector, global confluence theorem, unconditional subject-reduction theorem, or detached consistency proof is exported.

Apply Definition 1190.2. The seven obligation rows cover carrier admission, the normalization-audit row, the confluence-audit row, the discharge-ledger row, the closed-consistency assembly row, closed typed term route correspondence, and the replay/provenance/local-name rows.

Apply Definition 1190.3 and Definition 1190.7. The route theorem keeps every preservation or consistency handoff inside $N, Q, D, A, T, H, C, P, L$, while the TasteGate theorem records that Lean supplies the nontrivial and field-faithful surface needed to keep those rows distinct.

The falsifiable prediction states the observable route failure for unconditional closed subject reduction, and the independence witness records why the four sibling packets do not already contain the simultaneous bridge classifier. Hence the chapter closes precisely the obligation package for this bridge. It does not claim global normalization, global confluence, global subject reduction, or closed consistency. □

Theorem 1190.9 (MetaCIC normalization-confluence bridge scope index)**.** *The scoped* MetaCICNormalizationConfl *read consumes exactly the following displayed rows: the normalization-audit row N, the confluence-audit row Q, the subject-reduction obstruction and discharge row D, the closed-consistency assembly row A, the typed-term route-correspondence row T, componentwise transport H, displayed replay C, provenance P, and local naming L. No scoped read may replace these rows by a host evaluator, quotient reduction equality, hidden typed-term selector, unconditional confluence theorem, unconditional subject-reduction theorem, or detached consistency proof.*

Proof. Start with the carrier fixed in Definition 1190.1. Its coordinates are exactly $N, Q, D, A, T, H, C, P, L$.

Apply Definition 1190.2 to read the local obligation surface. The normalization audit, confluence audit, discharge-ledger boundary, closed-consistency assembly, typed-term correspondence, transport, replay, provenance, and local naming rows are all displayed there.

Apply Definition 1190.3, Definition 1190.6, and Definition 1190.5. These theorems exclude the host evaluator, quotient equality, hidden typed-term selector, unconditional confluence, unconditional subject reduction, and detached consistency routes, so the listed rows are the complete scoped index. □

Theorem 1190.10 (MetaCIC normalization-confluence bridge candidate-mediated normalization)**.** *Every normalization conclusion exported by an accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *carrier is mediated by the displayed candidate and closed-candidate evidence in N. It may be transported through H and replayed through C, and it may be compared with the confluence row Q through T, but it cannot be read as strong normalization for all closed typed terms outside the candidate-mediated route.*

Proof. Read the normalization side through N. By Definition 1190.1, this row is the normalization-audit packet carrying candidate, closed-candidate, candidate-set, application adequacy, Pi adequacy, and no-infinite-reduction reads.

The comparison row T only records that the normalization and confluence rows speak about the same displayed closed typed term. Transport through H and replay through C preserve that row structure; they do not add a universal normalization coordinate.

Finally apply Definition 1190.3 and Definition 1190.8. The bridge remains a finite route-correspondence certificate, so the normalization conclusion is candidate-mediated and not an unconditional strong-normalization theorem for all closed typed terms. □

Theorem 1190.11 (MetaCIC normalization-confluence bridge conditional Church-Rosser boundary)**.** *Every Church-Rosser use exported by an accepted* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ *carrier remains conditional on the displayed confluence row Q, the typed-term correspondence row T, and the discharge-ledger boundary row D. In particular, such a use cannot discharge the beta-substitution, application-argument, lambda-domain, and Pi-domain subject-reduction hypotheses recorded in D.*

Proof. The confluence side of the bridge is the row Q. It supplies only the checked confluence fragments and named confluence-boundary inputs recorded by the carrier.

The bridge can compare Q with the normalization row only through T. That comparison is a route correspondence for the displayed closed typed term, not a replacement for the preservation rows stored in D.

Apply Definition 1190.6. The row D is handed to the assembly row A only as the displayed subject-reduction discharge-ledger boundary. Hence Church-Rosser use remains conditional on Q, T, D and cannot discharge the four subject-reduction hypotheses. □

Falsifiable prediction: Any accepted bridge packet whose normalization and confluence rows agree on the closed typed term has no route to an unconditional closed subject-reduction theorem.

Independence witness: metacicnormalizationaudit, metacic-confluence-audit, subjectreductiondischargeledger, closed-consistency-assembly: the bridge has a simultaneous four-row carrier and route-correspondence classifier absent from each sibling alone.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}}$.

Constructive story: The MetaCIC normalization-confluence bridge is generated from a $BHist$ packet that records a normalization row, a confluence row, and the bridge route joining them to the local certificate surface. The carrier does not assert global metatheory; it reads only the named normalization and confluence evidence and transports that evidence along $hsame$ when the representing history changes. Cont supplies the route boundary: later uses of the bridge must extend the same packet rather than appeal to an external proof of adequacy. The TasteGate surface is then the visible bridge between these rows and the admitted NameCert obligation.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalizationConfluenceBridge}\uparrow}$, scopedClosed)

Scope closed: This chapter binds the MetaCIC normalization-confluence bridge carrier, NameCert obligations, route-boundary theorem, TasteGate-surface theorem, scope index, candidate-mediated normalization boundary, and conditional Church-Rosser boundary.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalizationConfluenceBridge}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.MetacicNormalizationConfluenceBridgeUp.MetacicNormalizationConfluenceBridgeTasteGate\_s`
Lean4 ✓ `BEDC.Derived.MetacicNormalizationConfluenceBridgeUp.taste gate`

Bridge status: none

Origin: ai

Not claimed: It does not claim full MetaCIC normalization, global confluence, or kernel soundness outside the displayed bridge packet.

Upgrade path: Public closure requires a checked consumer-facing package for normalization, confluence, discharge, and assembly reads over the same scoped bridge carrier.

Chapter 1191

A Concrete Naming Certificate for $\mathsf{PhysicalNameCertAudit}^{\uparrow}$

The $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ packet realizes the physical NameCert analogue of section 9414.38 as one finite BEDC naming certificate. It sits beside chapter 1126, chapter 1129, chapter 1149, and chapter 1161. Those siblings expose observation routing, audit strength, reflective reuse, and reflection-gap boundaries; this packet binds the certificate-template rows needed before a model signature may be read as a certified physical name.

Definition 1191.1 (PhysicalNameCertAudit carrier)**.** *A* $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, O, M, C, Y, L, F, H, R, Q, N)$$

with the following displayed rows:

1. *S is the source row recording the finite observation or experiment trace that licenses the naming attempt;*
2. *O is the observed-signature row extracted from S through the displayed observation-reflection route;*
3. *M is the model-signature row offered as the physical name candidate;*
4. *C is the classifier row comparing O and M without importing a hidden model-reality identity;*
5. *Y is the stability certificate row recording the continuation discipline under which the comparison remains reusable;*
6. *L is the ledger policy row recording accepted fits, open gaps, and hidden residue that may later become active;*
7. *F is the failure-surface row on which missing source, classifier, stability, or ledger data are recorded as certificate blockers;*
8. *H is the componentwise* hsame *transport row for source, signature, classifier, stability, ledger, and failure reads;*
9. *R is the displayed* Cont *route through S, O, M, C, Y, L, F, H;*
10. *Q is the* Pkg *provenance over* $\mathsf{PhysicalNameCertAudit}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalNameCertAudit}^{\uparrow}}$ *row naming exactly this packet.*

The carrier has no coordinate for final truth, source-domain identity, unledgered approximation, host physical law, or an external model-selection oracle.

Theorem 1191.2 (PhysicalNameCertAudit obligation surface)**.** *For the carrier of Definition 1191.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalNameCertAudit}^{\uparrow}}$ *surface consists exactly of the finite obligations for source exposure in S, observed-signature extraction in O, model-signature presentation in M, classifier comparison in C, stability certification in Y, ledger policy in L, failure-surface accounting in F, componentwise transport in H, displayed routing in R, provenance in Q, and local naming in N.*

Proof. Project the carrier tuple of Definition 1191.1. It exposes the source, observed signature, model signature, classifier, stability, ledger, failure, transport, route, provenance, and local naming rows as the only packet coordinates.

The observation side is read through chapter 1126. That sibling supplies the finite source and observed-signature route, but it does not by itself certify a physical model name.

The audit and stability rows are read through chapter 1129 and chapter 1149. These give the classifier-facing model audit and continuation stability surfaces needed before the local naming row can be consumed.

Finally use chapter 1161. The boundary row keeps the certificate open to later defeating observations, so the ledger and failure rows remain part of the same finite NameCert surface rather than hidden host evidence. □

Lean4 ✓ `BEDC.Derived.PhysicalNameCertAuditUp.PhysicalNameCertAuditTasteGate single carrier alignment`

Theorem 1191.3 (PhysicalNameCertAudit non-escape)**.** *Every accepted* $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ *packet exports only the displayed rows*

$$S, O, M, C, Y, L, F, H, R, Q, N.$$

A consumer may promote a model signature to a certified physical name only through those rows. It cannot replace them by an unrecorded source, a detached pattern, an implicit classifier, an unstated stability principle, an absent ledger policy, a hidden failure surface, final truth, source-domain identity, or host physical-law semantics.

Proof. Inspect Definition 1191.1. The packet has explicit rows for each certificate-template field and no additional coordinate for a hidden source, detached pattern, final truth, or host physical law.

Apply Definition 1191.2. Source, observed signature, model signature, classifier, stability, ledger, and failure data must all be read before the route, provenance, and local name are consumed.

The non-escape boundary follows because any missing source, classifier, stability, or ledger datum is recorded in F as a blocker rather than supplied by an external oracle. The local NameCert row can therefore name only the displayed packet. □

Theorem 1191.4 (PhysicalNameCertAudit certification handoff)**.** *If a* $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ *packet admits source row S, observed-signature row O, model-signature row M, classifier row C, stability row Y, ledger row L, failure-surface row F, transport row H, displayed route R, provenance row Q, and local name N, then the only certification handoff available to a downstream physical-model consumer is the displayed route from O, M, C, Y, L, F to N. If the source, classifier, stability, or ledger row is absent, the handoff stops at the failure surface and no certified physical name is exported.*

Proof. Start with the obligation surface of Definition 1191.2. It requires the source, signature, classifier, stability, ledger, and failure rows before the local naming row is available.

Use non-escape from Definition 1191.3. Since no hidden source, classifier, stability principle, ledger policy, or host law can be projected from the packet, the handoff cannot be repaired outside the displayed carrier.

Thus a complete carrier hands the model signature to the local NameCert through R, Q, N. An incomplete carrier reaches F and remains a failed naming attempt rather than a certified physical name. □

Falsifiable prediction: If the source, classifier, stability, or ledger row is absent, a PhysicalNameCertAuditUp packet cannot promote a model signature to a certified physical name.

Independence witness: observationreflectionpacket, physicalmodelaudit, reflectivestability, reflectiongapboundary, physicaltruthcertificate: no sibling is bijective because PhysicalNameCertAuditUp uniquely binds the certificate-template rows for physical model naming rather than only routing, audit strength, reflection stability, gap boundary, or truth status.

The Lean follow-up in `lean4/BEDC/Derived/PhysicalNameCertAuditUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ and FieldFaithful $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ instances preserving $S, O, M, C, Y, L, F, H, R, Q, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalNameCertAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalNameCertAudit}^{\uparrow}$ packet is generated from finite source rows, observed signatures, model signatures, a classifier, stability certificate, ledger policy, failure surface, componentwise transport, displayed route, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PhysicalNameCertAudit}^{\uparrow}}$, seed)

Scope closed: Definition 1191.1, Definition 1191.2, Definition 1191.3, and Definition 1191.4 bind the physical NameCert analogue as a finite source, signature, classifier, stability, ledger, failure, route, provenance, and naming packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PhysicalNameCertAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final truth, source-domain identity, host physical-law semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the carrier, non-escape boundary, certification handoff, and TasteGate field-faithful rows for $S, O, M, C, Y, L, F, H, R, Q, N$.

Chapter 1192

A Concrete Naming Certificate for $\mathsf{FiniteObservationReflectionSeal}^{\uparrow}$

The $\mathsf{FiniteObservationReflectionSeal}^{\uparrow}$ packet realizes the observation–reflection equivalence of section 9414.38 as a finite NameCert seal. It sits after the observation handoff of chapter 1126, the finite window discipline of chapter 126, and the reflective inquiry carrier of chapter 1151. Those siblings expose observation, stream, or inquiry rows separately; this carrier binds physical observation and MetaCIC reflection to one shared classifier, ledger, and non-finalization surface.

Definition 1192.1 (Finite observation-reflection seal carrier)**.** *A* $\mathsf{FiniteObservationReflectionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, M, S, K, C, L, H, Q, N)$$

with the following displayed rows:

1. *P is the physical-observation source row, read through finite reality-facing traces;*
2. *M is the MetaCIC-reflection source row, read through finite formal self-study traces;*
3. *S is the shared finite signature row comparing the physical observation surface with the formal reflection surface;*
4. *K is the classifier row accepting the displayed observation-reflection role correspondence;*
5. *C is the shared* Cont *route under which both inquiry directions remain extendable;*
6. *L is the non-finalization ledger recording rejected final physical or MetaCIC truth-oracle demands;*
7. *H records componentwise* hsame *transport for P, M, S, K, C, L;*
8. *Q records* Pkg *provenance for the same finite packet and its sibling observation-reflection, stream, and reflective-inquiry rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationReflectionSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a final physical truth oracle, a final MetaCIC truth oracle, source-domain identity, host validation, quotient stream equality, or an unbounded observation schedule.

Lean4 ✓ `BEDC.Derived.FiniteObservationReflectionSealUp.FiniteObservationReflectionSealTasteGate`
`single carrier alignment`

Theorem 1192.2 (Finite observation-reflection seal NameCert obligations)**.** *For the carrier of Definition 1192.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationReflectionSeal}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for* $F = (P, M, S, K, C, L, H, Q, N)$*;*
2. *physical-observation exposure through* P *and formal-reflection exposure through* M*;*
3. *shared signature and classifier acceptance through* S *and* K*;*
4. *continuation and non-finalization through the shared route* C *and ledger* L*;*
5. *componentwise transport, provenance, and local naming through* H, Q, N*.*

Thus the seal certifies a finite correspondence between physical inquiry and MetaCIC self-study without identifying their sources and without closing either inquiry direction.

Proof. Carrier admission projects the displayed packet $F = (P, M, S, K, C, L, H, Q, N)$. The two source rows P and M are separate coordinates, so the seal can expose physical observation and MetaCIC reflection while refusing source identity.

The signature and classifier rows provide the shared correspondence. Row S records the finite signature comparison, and row K records the accepted classifier judgment. These rows consume the observation-reflection packet and reflective inquiry surfaces rather than replacing them.

The route and ledger rows then close the non-finality condition. The shared continuation row C keeps both directions extendable, while L records rejected final truth-oracle demands. Rows H, Q, N transport, package, and name exactly the same displayed route, so no further NameCert obligation remains outside the packet. □

Theorem 1192.3 (Finite observation-reflection seal non-escape)**.** *No accepted* $\mathsf{FiniteObservationReflectionSeal}^{\uparrow}$ *packet exports a final physical truth oracle, a final MetaCIC truth oracle, source-domain identity, host validation, quotient stream equality, or an unbounded observation schedule beyond the rows*

$$P, M, S, K, C, L, H, Q, N.$$

Every consumer of the seal must read the displayed physical-observation row, MetaCIC-reflection row, shared signature, classifier, continuation, non-finalization ledger, transport, provenance, and local NameCert rows.

Proof. Inspect the carrier of Definition 1192.1. Its only source-facing rows are P and M, its only correspondence rows are S and K, and its only future-facing and refusal rows are C and L.

Apply the obligation theorem Definition 1192.2. It forces every seal read through physical exposure, formal exposure, shared signature comparison, classifier acceptance, continuation, and the non-finalization ledger before the local name is consumed.

The sibling packets do not add the missing oracle rows. $\mathsf{ObservationReflectionPacket}^{\uparrow}$ supplies finite observation-reflection roles, $\mathsf{StreamName}^{\uparrow}$ supplies finite window discipline, and $\mathsf{ReflectiveInquiry}^{\uparrow}$ supplies the paired inquiry carrier, but none of them contributes a final physical oracle, final MetaCIC oracle, source identity, quotient stream equality, or unbounded schedule to this seal. □

Falsifiable prediction: A finite observation-reflection seal is refuted by any accepted consumer that exports a final physical or MetaCIC truth oracle from the same finite packet.

Independence witness: observationreflectionpacket, streamname, reflectiveinquiry, realityconstrainedobservationreflection, namecert: the seal is not bijective to any sibling because it binds

both physical-observation and MetaCIC-reflection rows plus the shared non-finalization ledger in one packet.

The Lean follow-up must provide Nontrivial FiniteObservationReflectionSeal$^{\uparrow}$ and FieldFaithful FiniteObservationR instances in `lean4/BEDC/Derived/FiniteObservationReflectionSealUp/TasteGate.lean`, preserving $P, M, S, K, C, L, H, Q, N$.

Closure status for NameCert$_{\text{FiniteObservationReflectionSeal}^{\uparrow}}$.

Constructive story: A FiniteObservationReflectionSeal$^{\uparrow}$ packet is generated from physical-observation, MetaCIC-reflection, shared signature, classifier, continuation, non-finalization ledger, transport, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{FiniteObservationReflectionSeal}^{\uparrow}}$, seed)

Scope closed: Definition 1192.1, Definition 1192.2, and Definition 1192.3 bind the finite observation-reflection seal and its non-finalization boundary.

Formal verification: FormalStatus(NameCert$_{\text{FiniteObservationReflectionSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert final physical truth, final MetaCIC truth, source-domain identity, host validation, quotient stream equality, unbounded observation schedules, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean encoding of the carrier, five NameCert obligations, non-escape theorem, TasteGate nontriviality, and field faithfulness for $P, M, S, K, C, L, H, Q, N$.

Chapter 1193

A Concrete Naming Certificate for $\mathsf{PhysicalOperationDescent}^{\uparrow}$

The $\mathsf{PhysicalOperationDescent}^{\uparrow}$ packet realizes the operation-descent demand of chapter 9414 as a finite NameCert carrier. It sits beside the signature-fit packet of **??**, the endpoint audit of chapter 1176, the paired-tower boundary of chapter 1143, and the physical audit packet of chapter 1129. Those siblings expose physical-signature compatibility, endpoint-operation audit, tower comparison, and model-facing audit rows; this packet isolates the descent route of a single operation across source and endpoint classifiers before a consumer may apply the operation after classification.

Definition 1193.1 (PhysicalOperationDescent carrier)**.** *A* $\mathsf{PhysicalOperationDescent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, S, E, R, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the operation row, naming the operation, predicate, or comparison whose post-classifier use is requested.*
2. *S is the source-classifier row, read through the physical-signature and representation data of* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$*.*
3. *E is the endpoint-classifier row, read through the compressed or tower-facing endpoint at which O is intended to act.*
4. *R is the descent-route row, recording the displayed proof route that O respects S and lands in E.*
5. *F is the failure-exposure row, naming absent classifier compatibility, missing source signature, missing endpoint classifier, failed operation preservation, and audit defeat as displayed refusal reasons.*
6. *H records componentwise* hsame *transport for O, S, E, R, F.*
7. *C records displayed* Cont *replay through the same operation, classifier, route, and failure rows.*
8. *P is the* Pkg *provenance over* $\mathsf{PhysicalOperationDescent}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, CompressionDescentAu $\mathsf{TowerEquivalence}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalOperationDescent}^{\uparrow}}$ *row.*

The carrier has no coordinate for applying O after a classifier step without a displayed descent route, no endpoint-only operation oracle, no hidden physical-signature source, and no refusal surface outside F.

Lean4 ✓ `BEDC.Derived.PhysicalOperationDescentUp.TasteGate.PhysicalOperationDescentTasteGate single carrier alignment`

Definition 1193.2 (PhysicalOperationDescent classifier)**.** *For accepted packets*

$$D = (O, S, E, R, F, H, C, P, N) \quad \text{and} \quad D' = (O', S', E', R', F', H', C', P', N'),$$

the predicate $\mathsf{PhysicalOperationDescentClassifier}(D, D')$ *holds when H transports operation rows, source classifiers, endpoint classifiers, descent routes, failure rows, continuation replays, provenance, and local names componentwise, and when the transported route still records that the operation respects the source classifier and lands in the endpoint classifier.*

Classifier agreement is therefore not equality of endpoint results and not equality of operation syntax alone. It is agreement of the finite operation, source-classifier, endpoint-classifier, descent-route, failure-exposure, transport, replay, provenance, and local-name rows as one packet.

Theorem 1193.3 (PhysicalOperationDescent NameCert obligations)**.** *For the carrier of Definition 1193.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalOperationDescent}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission: the accepted object is exactly $D = (O, S, E, R, F, H, C, P, N)$;*
2. *classifier stability: every accepted comparison uses the classifier of Definition 1193.2;*
3. *route exposure: every accepted post-classifier operation read displays the operation row O, source classifier S, endpoint classifier E, and descent route R in the same packet;*
4. *failure exposure: every missing source signature, endpoint classifier, operation-preservation proof, or model-audit defeat appears through the failure row F before acceptance is refused;*
5. *non-escape: no consumer may apply O after a classifier step, export endpoint operation validity, or suppress a descent failure from rows outside $O, S, E, R, F, H, C, P, N$.*

Proof. Carrier admission is projection from Definition 1193.1. The displayed tuple lists the operation, source classifier, endpoint classifier, descent route, failure row, transport, replay, provenance, and local name as the only available rows.

Classifier stability follows from Definition 1193.2. Transport preserves the operation row and the two classifier rows together with the route that relates them, so an accepted comparison cannot replace route preservation by endpoint syntax or host equality.

Route exposure is the operational content of the packet. A post-classifier application of O is accepted only when O, S, E, R appear together, because R is the row that records respect for the source classifier and landing in the endpoint classifier.

Failure exposure and non-escape are the finite-packet boundary. Missing classifier compatibility, missing signatures, failed preservation, and audit defeat live in F, while every accepted or refused read remains inside $O, S, E, R, F, H, C, P, N$. □

Theorem 1193.4 (PhysicalOperationDescent signature preservation)**.** *Let*

$$D = (O, S, E, R, F, H, C, P, N)$$

be an accepted $\mathsf{PhysicalOperationDescent}^{\uparrow}$ *packet. If a consumer applies O after reading the source classifier S, the resulting endpoint read is physically admissible only through the displayed route R and endpoint classifier E, with the failure row F still visible in the same packet.*

Proof. Use the route-exposure clause of Definition 1193.3. The source classifier S is not a permission to apply O at an endpoint; it is one row of the accepted descent packet.

The descent route R supplies the displayed preservation claim from the source classifier to the endpoint classifier. The endpoint classifier E records the target comparison in which the result of O is read.

The failure row F remains part of the same carrier. If preservation fails, or if source-signature or endpoint-classifier data are missing, that refusal is still visible and the operation cannot be exported as a physically admissible endpoint read. □

Theorem 1193.5 (PhysicalOperationDescent ledger non-escape)**.** *Every accepted* $\mathsf{PhysicalOperationDescent}^{\uparrow}$ *consumer route factors through*

$$O, S, E, R, F, H, C, P, N.$$

In particular, a consumer that applies an operation after a classifier step is rejected unless the descent route and failure row are displayed in the same packet.

Proof. The carrier definition exposes exactly the nine rows $O, S, E, R, F, H, C, P, N$. The classifier definition transports those rows componentwise and keeps the operation, two classifiers, descent route, and failure exposure tied together.

Apply the NameCert obligations. Route exposure forces O, S, E, R into every accepted post-classifier operation read, and failure exposure forces F to remain available for every refusal reason attached to that read.

Therefore an operation application that keeps the source classifier while omitting the route or failure row has no coordinate in the accepted packet. The consumer has left the finite $\mathsf{PhysicalOperationDescent}^{\uparrow}$ carrier rather than reading it. □

Falsifiable prediction: A consumer that applies an operation after a classifier step is rejected unless the descent route and failure row are displayed in the same packet.

Independence witness: realityconstrainedsignaturefit, compressiondescentaudit, towerequivalence, physicalmodelaudit, openfitpacket: none is bijective because PhysicalOperationDescentUp isolates operation, source classifier, endpoint classifier, descent proof route, and failure exposure in one carrier.

The Lean chapter `TasteGate.lean` for $\mathsf{PhysicalOperationDescent}^{\uparrow}$ is expected to provide explicit $\mathsf{Nontrivial\ PhysicalOperationDescent}^{\uparrow}$ and $\mathsf{FieldFaithful\ PhysicalOperationDescent}^{\uparrow}$ instances. Its gate-facing theorem is expected at `BEDC.Derived.PhysicalOperationDescentUp.taste_gate`.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalOperationDescent}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalOperationDescent}^{\uparrow}$ packet is generated from operation, source-classifier, endpoint-classifier, descent-route, failure-exposure, componentwise transport, continuation replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalOperationDescent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1193.1, Definition 1193.2, Definition 1193.3, Definition 1193.4, and Definition 1193.5 bind operation descent through reality-constrained approximation classifiers to a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalOperationDescent}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.PhysicalOperationDescentUp.TasteGate.PhysicalOperationDescentTasteGate\_single\_carrier`

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert bridge checking, endpoint-only operation validity, total descent for all operations, host physical truth, or operation application without the displayed failure surface.

Upgrade path: Obligation closure requires Lean encoding of classifier stability, signature preservation, and ledger non-escape beyond the carrier alignment theorem for $O, S, E, R, F, H, C, P, N$.

Chapter 1194

A Concrete Naming Certificate for $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$

The $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ carrier realizes the Inter-Hist Locality vision of chapter 9419 as one finite two-history NameCert packet. It sits after the observer-history identity surface of chapter 479, the local time-stream surface of chapter 603, the inter-Hist invariant carrier of chapter 656, the cross-Hist causal route of chapter 875, and the no-global-synchronization boundary of chapter 1003. Its role is to bind two displayed histories, a locality witness, an invariant row, observer symmetry, transport, replay, provenance, and local naming before any downstream consumer reads cross-history locality.

Falsifiable prediction: Any accepted FiniteHistLocalityPacketUp read that inserts a global synchronization row or deletes one displayed Hist row refutes the carrier.

Independence witness: observerhistidentity, histtimestream, interhistinvariant, crosshistcausal, noglobalsyncboundary: none is bijective because this packet simultaneously carries two displayed Hist rows, locality witness, invariant row, symmetry row, transport, Cont route, provenance, and local NameCert.

Definition 1194.1 (Finite Hist locality packet carrier)**.** *A* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (H_0, H_1, L, I, S, T, C, Q, N)$$

with the following displayed rows:

1. H_0 *is the first displayed* BHist *row;*
2. H_1 *is the second displayed* BHist *row;*
3. L *is the finite locality-witness row relating the two displayed histories;*
4. I *is the inter-Hist invariant row read through the same locality witness;*
5. S *is the observer-symmetry row that prevents either history from becoming privileged;*
6. T *records componentwise* hsame *and* psame *transport for* H_0, H_1, L, I, S*;*
7. C *is the displayed* Cont *replay route for the same two-history surface;*
8. Q *is the* Pkg *provenance over observer-history identity, local time streams, inter-Hist invariants, cross-Hist causal dependence, no-global-synchronization boundaries,* BHist*,* hsame*,* psame*,* Cont*, and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$ *row.*
The carrier has no coordinate for a global synchronization row, shared host clock, preferred observer frame, ambient metric tensor, or host-valued cross-history equality.

Lean4 (definition) `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketUp`
Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.taste gate`

Theorem 1194.2 (Finite Hist locality packet obligation surface)**.** *For the carrier of Definition 1194.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$ *surface consists of five finite obligation rows: carrier admission for P, two displayed Hist rows H_0, H_1 with locality witness L, invariant and observer-symmetry exposure through I, S, componentwise* hsame/psame *transport and* Cont *replay through T, C, and package provenance with local naming through Q, N.*

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketObligationSurface`

Proof. Project the carrier rows of Definition 1194.1. The two histories, locality witness, invariant row, symmetry row, transport, replay, provenance, and local name are the complete displayed inventory.

Group that inventory into the five NameCert obligations stated above. The first admits the packet, the second exposes the two Hist rows and their locality witness, the third reads the invariant and symmetry rows, the fourth transports and replays the surface, and the fifth records provenance and local naming.

No remaining coordinate can supply a global synchronizer, shared host clock, preferred frame, ambient metric tensor, or host-valued equality, because none appears in P. □

Theorem 1194.3 (Finite Hist locality packet no-global-sync refusal)**.** *Every accepted* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *read preserves both displayed Hist rows H_0, H_1 and routes locality through L, I, S, T, C, Q, N. It cannot insert a global synchronization row, collapse the two histories into one host clock, or replace observer symmetry by a privileged frame.*

Proof. Start from the obligation surface of Definition 1194.2. The packet has two displayed history rows and a locality witness before the invariant row or symmetry row can be read.

The no-global-synchronization sibling of chapter 1003 supplies the refusal shape consumed by Q, while the local time-stream sibling of chapter 603 keeps time rows attached to their source histories.

Thus an accepted read remains inside $H_0, H_1, L, I, S, T, C, Q, N$. A global synchronizer, one-host-clock collapse, or privileged observer frame would require a row outside that inventory. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketNoGlobalSyncRefusal`

Theorem 1194.4 (Finite Hist locality packet transport stability)**.** *Componentwise* hsame *and* psame *transport through T preserves the two displayed Hist rows, locality witness, invariant row, observer-symmetry row,* Cont *replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$ *row. Transport cannot turn the packet into a single-history surface or a global-frame certificate.*

Proof. Read T in Definition 1194.1. It transports the displayed rows H_0, H_1, L, I, S and is paired with the replay row C, provenance row Q, and local name N.

Apply Definition 1194.2. The transported rows remain the same five-obligation surface, so no consumer obtains a hidden synchronizer or a replacement history row.

Finally use Definition 1194.3. Since global-frame and one-clock coordinates are absent before transport, componentwise transport cannot create them afterward. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketTransportStability`

Theorem 1194.5 (Finite Hist locality packet consumer handoff)**.** *Every cross-Hist causal, maximum-rate, or observer-locality consumer of an accepted* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *packet receives only*

$$H_0, H_1, L, I, S, T, C, Q, N.$$

The handoff supplies a two-history locality route with invariant and symmetry data, but it supplies no global synchronization row, selected global clock, privileged observer frame, ambient metric tensor, or host-valued equality.

Proof. Use Definition 1194.2 to identify the public packet rows. Any consumer receives the two histories, locality witness, invariant row, observer-symmetry row, transport, replay, provenance, and local name.

Apply Definition 1194.4 to preserve that inventory under hsame/psame transport. The consumer therefore sees the same finite route after transport as before transport.

The refusal theorem, Definition 1194.3, excludes every route through global synchronization, selected global clocks, privileged frames, ambient metric tensors, or host-valued equality. Hence the handoff is exactly the finite two-history locality packet. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketConsumerHandoff`

Theorem 1194.6 (Finite Hist locality packet TasteGate handoff)**.** *The same displayed inventory* $H_0, H_1, L, I, S, T, C, Q, N$ *is the paper-side obligation surface that later Lean development must use for* Nontrivial $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *and* FieldFaithful $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *instances in* `lean4/BEDC/Derived/FiniteHistLocalityPacketUp/TasteGate.lean`*.*

Proof. The carrier definition supplies the single concrete field inventory. The obligation-surface theorem identifies those fields as the complete local NameCert surface, while transport stability keeps the same fields under hsame and psame movement.

The consumer handoff then shows that downstream readers obtain no field beyond $H_0, H_1, L, I, S, T, C, Q, N$. Therefore the later Lean TasteGate instances must be Nontrivial and FieldFaithful for this displayed carrier rather than for a hidden global-frame object. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketTasteGate single carrier alignment`

Theorem 1194.7 (Finite Hist locality packet obligation-closure package)**.** *For every accepted* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *carrier*

$$P = (H_0, H_1, L, I, S, T, C, Q, N),$$

the obligationClosure surface consists exactly of carrier admission, the two displayed Hist rows H_0, H_1*, the locality witness* L*, the invariant row* I*, observer symmetry through* S*, componentwise* hsame/psame *transport through* T*, displayed* Cont *replay through* C*, package provenance through* Q*, and local* $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$ *naming through* N*.*

Every accepted read must preserve that full row order before any cross-Hist causal, maximum-rate, observer-locality, or TasteGate consumer receives the packet. No obligation row is supplied by a global synchronization coordinate, shared host clock, privileged observer frame, ambient metric tensor, or host-valued equality.

Proof. Carrier admission is projection from Definition 1194.1. The projected inventory is the displayed tuple $H_0, H_1, L, I, S, T, C, Q, N$, with no hidden row outside the packet.

Apply Definition 1194.2. It groups the inventory into carrier admission, two Hist rows with locality witness, invariant and symmetry exposure, transport and replay, and provenance with local naming.

Use Definition 1194.4 and Definition 1194.5 for downstream reads. Transport preserves the same displayed tuple, and every consumer handoff receives only the two-history locality route.

Finally apply Definition 1194.3 and the TasteGate handoff of Definition 1194.6. They exclude global synchronization and require later Nontrivial and FieldFaithful reads to use the same displayed carrier, so the listed rows exhaust the obligationClosure package. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketCarrier obligation closure package`

Theorem 1194.8 (FiniteHistLocalityPacket scoped kernel scope)**.** *For every accepted* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *carrier*

$$P = (H_0, H_1, L, I, S, T, C, Q, N),$$

every cross-Hist causal, maximum-rate, observer-locality, and TasteGate consumer factors through the displayed rows $H_0, H_1, L, I, S, T, C, Q, N$*. The closed scope is limited to the two* BHist *rows,* H_0, H_1*, the locality witness* L*, the inter-Hist invariant row* I*, observer symmetry* S*,* hsame/psame *transport through* T*,* Cont *replay through* C*,* Pkg *provenance through* Q*, the local* $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$ *row* N*, and the sibling routes cited by this chapter.*

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketScopedKernelScope`

Proof. Project the carrier in Definition 1194.1. The only available rows are the displayed tuple $H_0, H_1, L, I, S, T, C, Q, N$, with no global synchronization coordinate and no hidden host-clock row.

Apply the obligation package of Definition 1194.7. It binds carrier admission, both displayed Hist rows, locality, invariant exposure, observer symmetry, transport, replay, provenance, and local naming into one obligationClosure surface.

Use the refusal and stability theorems next. Definition 1194.3 blocks any route through global synchronization, while Definition 1194.4 keeps T as componentwise hsame/psame transport rather than a host equality.

Now read the consumer handoff. Definition 1194.5 sends cross-Hist causal, maximum-rate, and observer-locality consumers only through the displayed two-history packet, and Definition 1194.6 requires the TasteGate surface to use the same carrier inventory.

The cited sibling chapters supply observer-history identity, local time streams, inter-Hist invariants, cross-Hist causal dependence, and the no-global-synchronization boundary. Since the carrier, obligations, refusal, transport, consumer handoff, and TasteGate handoff all name the same finite rows, the scoped kernel dependency theorem closes exactly that displayed scope. □

Lean4 ✓ `BEDC.Derived.FiniteHistLocalityPacketUp.FiniteHistLocalityPacketCarrier scoped kernel scope`

Theorem 1194.9 (Finite Hist locality packet bridge surface)**.** *Every standard cross-Hist locality reading of an accepted* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *public packet factors through the displayed rows*

$$H_0, H_1, L, I, S, T, C, Q, N$$

and through the sibling routes for observer-history identity, inter-Hist invariants, and no-global-synchronization. The bridge surface is therefore the finite two-history packet exposed by the chapter, not a global clock, privileged observer frame, ambient metric tensor, or host-valued equality of histories.

Proof. Start with the consumer handoff of Definition 1194.5. It exposes exactly the two displayed Hist rows, the locality witness, the invariant row, observer symmetry, transport, replay, provenance, and local naming to cross-Hist causal, maximum-rate, and observer-locality consumers.

Use the scoped kernel scope theorem Definition 1194.8. It keeps those same rows inside the cited sibling routes for observer-history identity, local time streams, inter-Hist invariants, cross-Hist causal dependence, and the no-global-synchronization boundary.

Finally apply the no-global-sync refusal of Definition 1194.3. Since the packet has no global synchronization row, shared host clock, privileged observer frame, ambient metric tensor, or host-valued equality coordinate, every standard locality reading factors through the displayed finite bridge surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ packet is generated from two displayed Hist rows, a locality witness, inter-Hist invariant row, observer-symmetry row, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1194.9 completes the mature paper surface for finite locality examples by factoring standard cross-Hist locality readings through the displayed public packet rows, the cross-chapter observer-history, local time-stream, inter-Hist invariant, cross-Hist causal, and no-global-synchronization routes, and the explicit refusal of host synchronization claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteHistLocalityPacket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close global synchronization, a host clock, a privileged observer frame, an ambient metric tensor, host-valued equality, analytic spacetime structure, or checked Lean carrier instances.

Upgrade path: A checked standard bridge requires Lean verification of the displayed cross-Hist locality bridge surface.

Chapter 1195

A Concrete Naming Certificate for $\mathsf{TowerEndpointReflection}^{\uparrow}$

The $\mathsf{TowerEndpointReflection}^{\uparrow}$ packet realizes the endpoint/source reflection boundary of section 9414.20 as a finite BEDC NameCert carrier. It consumes the tower-compression packet of chapter 1142, the endpoint-equivalence discipline of chapter 1143, the composite-ledger classifier of chapter 1162, and the descent decision surface of chapter 1176. Those siblings expose endpoint rows, tower ledgers, functorial composition, and descent audits; this carrier binds endpoint reflection back to a displayed source row without collapsing endpoint truth into source truth.

Definition 1195.1 (TowerEndpointReflection carrier)**.** *A* $\mathsf{TowerEndpointReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, S, K, Q, F, D, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint row, naming the terminal visible package whose reflection is being queried.*
2. *S is the source row, naming the base system, trace family, or observation process to which reflection is proposed.*
3. *K is the* $\mathsf{RealityConstrainedTowerCompression}^{\uparrow}$ *row, exposing the source, tower, classifier, endpoint, and composite-ledger surface.*
4. *Q is the* $\mathsf{TowerEquivalence}^{\uparrow}$ *row, recording endpoint agreement together with ledger and descent compatibility.*
5. *F is the* $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ *row, recording the composite-ledger route assigned to the displayed tower arrows.*
6. *D is the* $\mathsf{CompressionDescentAudit}^{\uparrow}$ *row, recording the accepted or refused descent route for the endpoint operation under reflection.*
7. *O is the obstruction row, listing the finite reasons that block reflection, including missing composite ledger, failed descent audit, endpoint-only equivalence, source mismatch, and model-facing non-descent.*
8. *H records componentwise* hsame *transport for E, S, K, Q, F, D, O.*
9. *C is the displayed* Cont *replay route through the same endpoint, source, ledger, equivalence, descent, and obstruction rows.*

10. *P is the* Pkg *provenance over* $\mathsf{TowerEndpointReflection}^{\uparrow}$*, the four cited tower siblings,* $\mathsf{RealWindowBudget}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TowerEndpointReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for endpoint truth as source truth, source identity from endpoint equivalence, unledgered reflection, private descent repair, or a model-facing acceptance route outside R.

Lean4 (definition) `BEDC.Derived.TowerEndpointReflectionUp.TowerEndpointReflectionUp`

Theorem 1195.2 (TowerEndpointReflection descent boundary)**.** *Let*

$$R = (E, S, K, Q, F, D, O, H, C, P, N)$$

be an accepted $\mathsf{TowerEndpointReflection}^{\uparrow}$ *packet. Any endpoint/source reflection exported from E to S requires both the composite-ledger row carried by K, F and the descent-audit row D. A consumer that keeps endpoint equivalence through Q while omitting either the composite ledger or the descent audit has an endpoint comparison, not a source-facing reflection.*

Proof. Project the carrier of Definition 1195.1. The endpoint row E and source row S are displayed together with tower compression K, tower equivalence Q, ledger functor F, descent audit D, and obstruction row O.

The tower-compression row exposes the source, classifier, endpoint, and composite-ledger discipline of chapter 1142. The ledger-functor row then records the composite route by which the tower arrows transport hidden-source material to the endpoint-facing ledger.

The descent-audit row supplies the separate decision surface required by chapter 1176. It is not implied by endpoint equivalence, because chapter 1143 binds agreement only when ledger and descent compatibility are also displayed.

Thus endpoint/source reflection is licensed only through the joint route K, F, D transported by H and replayed by C. If the composite-ledger row or the descent-audit row is absent, the packet has no carrier coordinate that can move an endpoint claim from E to S. □

Theorem 1195.3 (TowerEndpointReflection NameCert obligations)**.** *For the carrier of Definition 1195.1, the local* $\mathsf{NameCert}_{\mathsf{TowerEndpointReflection}^{\uparrow}}$ *surface consists exactly of seven obligation rows:*

1. *carrier admission for $R = (E, S, K, Q, F, D, O, H, C, P, N)$;*
2. *endpoint and source exposure through E, S;*
3. *tower-compression exposure through K;*
4. *endpoint-equivalence exposure through Q;*
5. *composite-ledger exposure through F;*
6. *descent-audit and obstruction exposure through D, O;*
7. *componentwise transport, replay, provenance, and local naming closure through H, C, P, N.*

Every accepted endpoint/source reflection must display these rows before a consumer may treat an endpoint operation as source-facing.

Proof. The carrier definition gives exactly the eleven-row packet $R = (E, S, K, Q, F, D, O, H, C, P, N)$. The local NameCert therefore sees endpoint, source, tower-compression, equivalence, ledger-functor, descent-audit, obstruction, transport, replay, provenance, and naming coordinates.

Rows E and S name the boundary being reflected. Rows K, Q, F, D import the four cited siblings, separating tower compression, endpoint equivalence, composite-ledger routing, and descent audit instead of treating any one of them as sufficient.

The obstruction row O records the finite refusal surface whenever a ledger, descent, equivalence, or source-facing requirement is missing. The rows H, C, P, N keep the same packet stable under transport, replay, provenance, and local naming.

Since the carrier admits no further coordinate, the seven displayed obligations exhaust the NameCert surface. A consumer that lacks one of them is reading a different packet rather than an accepted $\mathsf{TowerEndpointReflection}^{\uparrow}$ carrier. □

The Lean development in `lean4/BEDC/Derived/TowerEndpointReflectionUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ TowerEndpointReflection}^{\uparrow}$ and $\mathsf{FieldFaithful\ TowerEndpointReflection}^{\uparrow}$ instances preserving $E, S, K, Q, F, D, O, H, C, P, N$.

Falsifiable prediction: An accepted TowerEndpointReflectionUp packet without a composite-ledger row or descent-audit row refutes the carrier.

Independence witness: realityconstrainedtowercompression, towerequivalence, compressionledgerfunctor, compressiondescentaudit, realwindowbudget: none is bijective because only $\mathsf{TowerEndpointReflection}^{\uparrow}$ binds endpoint row, source row, composite-ledger row, descent audit, obstruction row, componentwise transport, replay, provenance, and local NameCert as one reflection carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{TowerEndpointReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{TowerEndpointReflection}^{\uparrow}$ packet is generated from endpoint, source, tower-compression, tower-equivalence, composite-ledger, descent-audit, obstruction, componentwise transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TowerEndpointReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1195.1, Definition 1195.2, and Definition 1195.3 bind endpoint/source reflection to a finite carrier with displayed composite-ledger and descent-audit obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TowerEndpointReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert Lean verification, bridge checking, endpoint truth as source truth, source identity from endpoint equivalence, unledgered reflection, private descent repair, or model-facing acceptance outside the displayed packet.

Upgrade path: Obligation closure requires Lean encoding of carrier admission, composite-ledger exposure, descent-audit exposure, obstruction non-escape, TasteGate nontriviality, and field faithfulness for $E, S, K, Q, F, D, O, H, C, P, N$.

Chapter 1196

A Concrete Naming Certificate for $\mathsf{AxisCarryRefusalLedger}^{\uparrow}$

The $\mathsf{AxisCarryRefusalLedger}^{\uparrow}$ packet realizes the cannot-claim registry of chapter 9350 as one transportable finite refusal ledger. It sits after the concrete registry packet of chapter 554, reads the bridge boundary of chapter 9344, and keeps the $\mathsf{AxisNat}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ interfaces of chapter 276 and chapter 28 distinct.

It is an ai over the $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{UnaryDirectionBridge}^{\uparrow}$, $\mathsf{AxisNat}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{FullAxis}^{\uparrow}$ sibling surfaces, with the ledger row serving as the finite transportable refusal bundle.

Falsifiable prediction: Any consumer that treats ZCarry as hsame, FullAxisUp as RealUp, or AxisNatUp as replacing NatUp is rejected unless the corresponding refusal row is absent from the packet.

Independence witness: axiszeckendorfcannotclaim, unarydirectionbridge, axisnat, nat, fullaxis: no sibling is bijective because this packet uniquely binds checked negative rows and horizon refusal rows into one transportable refusal ledger.

Definition 1196.1 (Axis carry-refusal ledger carrier)**.** *An* $\mathsf{AxisCarryRefusalLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, Z, F, A, B, L, H, C, P, N)$$

with the following displayed rows:

1. *G is the accepted* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry row containing the checked negative claims and horizon refusals;*
2. *Z is the checked row witnessing that a displayed* $\mathsf{ZCarry}^{\uparrow}$ *edge is not an* hsame *identification;*
3. *F is the horizon row refusing to read* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$ *or as a complete metric object;*
4. *A is the horizon row refusing to read* $\mathsf{AxisNat}^{\uparrow}$ *as a replacement for* $\mathsf{Nat}^{\uparrow}$*;*
5. *B records the* $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *boundary through which any comparison between axis and unary naturals must pass;*
6. *L is the refusal ledger binding the negative rows Z, F, A to their registry source G and bridge boundary B;*
7. *H records componentwise* hsame *transport for $G, Z, F, A, B, L, C, P, N$ without converting a refused identification into equality;*

8. *C records displayed* Cont *replay for the registry row, refusal rows, bridge boundary, ledger, transports, provenance, and local naming;*
9. *P is the* Pkg *provenance over* AxisZeckendorfCannotClaim$^{\uparrow}$, ZCarry$^{\uparrow}$, FullAxis$^{\uparrow}$, Real$^{\uparrow}$, AxisNat$^{\uparrow}$, Nat$^{\uparrow}$, UnaryDirectionBridge$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* NameCert$_{\text{AxisCarryRefusalLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for turning ZCarry$^{\uparrow}$ *into* hsame*, promoting* FullAxis$^{\uparrow}$ *to* Real$^{\uparrow}$*, replacing* Nat$^{\uparrow}$ *by* AxisNat$^{\uparrow}$*, or bypassing the bridge boundary B.*

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerTasteGate single carrier alignment`

Theorem 1196.2 (Axis carry-refusal ledger realizes registry refusals)**.** *Every accepted* AxisCarryRefusalLedger$^{\uparrow}$ *packet $R = (G, Z, F, A, B, L, H, C, P, N)$ exposes the cannot-claim registry as a finite ledger of negative rows. The carry refusal Z, full-axis horizon refusal F, axis-natural refusal A, bridge boundary B, ledger L, transport H, continuation replay C, provenance P, and local naming row N must be read together before a consumer may cite the registry as a transportable refusal packet.*

Proof. Project the carrier of Definition 1196.1. The registry row G supplies the source cannot-claim packet, while Z, F, A are the displayed refusal rows selected from it.

Read the bridge and ledger rows next. The bridge boundary B records where any AxisNat$^{\uparrow}$–Nat$^{\uparrow}$ comparison must pass, and L binds the carry, full-axis, and axis-natural refusals to the same registry source.

Finally replay the support rows. Transport through H preserves the displayed refusal components, C replays only those components, P records their provenance, and N names the finite packet. Since no excluded identification is a coordinate of R, every accepted consumer remains inside the ledger above. □

Theorem 1196.3 (Axis carry-refusal ledger non-identification)**.** *For an accepted* AxisCarryRefusalLedger$^{\uparrow}$ *packet, no consumer can derive* hsame *from the* ZCarry$^{\uparrow}$ *row, a* Real$^{\uparrow}$ *field surface from the* FullAxis$^{\uparrow}$ *row, or a replacement of* Nat$^{\uparrow}$ *by* AxisNat$^{\uparrow}$ *from the axis-natural row. Each such reading is rejected unless its corresponding refusal row Z, F, or A is absent, in which case the packet is not the accepted carrier of Definition 1196.1.*

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier nonidentification rows`

↪ variant `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier refusal rows nonidentification`

Proof. Use Definition 1196.2. The accepted carrier exposes Z, F, A as part of the same finite refusal ledger, so none of the three rows is optional for this packet.

The structural rows do not add hidden identifications. The transport row H preserves displayed components, C replays the same refusal route, and P, N only record provenance and local naming.

Coordinate exhaustion gives the claim. The carrier contains no host equality, no Real field-operation surface for FullAxis$^{\uparrow}$, and no replacement map from AxisNat$^{\uparrow}$ to Nat$^{\uparrow}$; therefore the proposed readings are rejected by the ledger classifier. □

Theorem 1196.4 (Axis carry-refusal ledger negative-row exactness)**.** *For an accepted* AxisCarryRefusalLedger$^{\uparrow}$ *packet*

$$R = (G, Z, F, A, B, L, H, C, P, N),$$

the negative rows Z, F, A are exactly the refusal rows sourced from the $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *registry row G and bound into the ledger row L. The row Z refuses to identify* $\mathsf{ZCarry}^{\uparrow}$ *with* hsame*, the row F refuses to read* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$ *or as a complete metric object, and the row A refuses to replace* $\mathsf{Nat}^{\uparrow}$ *by* $\mathsf{AxisNat}^{\uparrow}$*. Transport through H, replay through C, provenance through P, and local naming through N preserve those refusals without converting them into equalities or quotients.*

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier negative row layer separation`

Proof. Project the carrier of Definition 1196.1. The rows $G, Z, F, A, B, L, H, C, P, N$ are all displayed, and Z, F, A are the only negative rows carried by the packet.

Use Definition 1196.2. It reads the $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ registry through G and binds the carry, full-axis, and axis-natural refusals into the same finite ledger L.

Now apply the non-identification theorem Definition 1196.3. Since H, C, P, N only preserve, replay, source, and name the displayed rows, no refusal row can become an equality, a quotient witness, or a Real field surface. □

Theorem 1196.5 (Axis carry-refusal ledger bridge-boundary preservation)**.** *Every accepted comparison between the* $\mathsf{AxisNat}^{\uparrow}$ *side and the* $\mathsf{Nat}^{\uparrow}$ *side of an* $\mathsf{AxisCarryRefusalLedger}^{\uparrow}$ *packet*

$$R = (G, Z, F, A, B, L, H, C, P, N)$$

passes through the displayed $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *boundary row B. The refusal rows Z, F, A and ledger row L may be transported along H and replayed through C, but the packet does not permit a consumer to bypass B by treating* $\mathsf{AxisNat}^{\uparrow}$ *as* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{FullAxis}^{\uparrow}$ *as* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{ZCarry}^{\uparrow}$ *as an* hsame *equality.*

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier bridge boundary preservation`

Proof. Read the bridge coordinate in the carrier. The row B is the only displayed route through which axis and unary natural comparisons are admitted.

The refusal ledger theorem Definition 1196.2 binds B together with $G, Z, F, A, L, H, C, P, N$. Thus the bridge is not a decorative witness; it is part of the same transportable packet as the cannot-claim registry and the three negative rows.

Finally use Definition 1196.3. Any route that bypasses B would have to identify one of the refused surfaces, but the carrier contains no coordinate for such an identification. The bridge boundary is therefore preserved by every accepted transport and replay of the ledger. □

Theorem 1196.6 (Axis carry-refusal ledger local NameCert exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{AxisCarryRefusalLedger}^{\uparrow}}$ *row of an accepted packet*

$$R = (G, Z, F, A, B, L, H, C, P, N)$$

exposes exactly the registry row G, refusal rows Z, F, A, $\mathsf{UnaryDirectionBridge}^{\uparrow}$ *boundary row B, ledger row L, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N. It exposes no additional equality between* $\mathsf{ZCarry}^{\uparrow}$ *and* hsame*, no quotient identifying* $\mathsf{FullAxis}^{\uparrow}$ *with* $\mathsf{Real}^{\uparrow}$*, no replacement of* $\mathsf{Nat}^{\uparrow}$ *by* $\mathsf{AxisNat}^{\uparrow}$*, and no private route around the NameCert anchors.*

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier local namecert exhaustion`

Proof. Begin with the carrier definition. Its finite packet has exactly the rows $G, Z, F, A, B, L, H, C, P, N$, and the provenance row P names the relevant AxisZeckendorfCannotClaim$^{\uparrow}$, UnaryDirectionBridge$^{\uparrow}$, AxisNat$^{\uparrow}$, Nat$^{\uparrow}$, FullAxis$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert anchors.

Apply negative-row exactness to fix the meaning of Z, F, A. Apply bridge-boundary preservation to fix the only admissible route between the axis and unary natural sides.

It remains to account for local naming. The row N names the same finite packet after H and C have transported and replayed it; it does not create new carrier coordinates. Hence the local NameCert surface is exhausted by the displayed registry, refusal, bridge, ledger, transport, continuation, provenance, and naming rows. □

Closure status for NameCert$_{\text{AxisCarryRefusalLedger}^{\uparrow}}$.

Constructive story: An AxisCarryRefusalLedger$^{\uparrow}$ packet is generated from the AxisZeckendorf cannot-claim registry, checked carry refusal, FullAxis horizon refusal, AxisNat replacement refusal, unary-direction bridge boundary, ledger, componentwise transport, displayed replay, provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.AxisCarryRefusalLedgerUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{AxisCarryRefusalLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1196.1, Definition 1196.2, Definition 1196.3, and Definition 1196.6 specify the public carrier-facing route for the finite refusal ledger, including negative-row exactness, bridge-boundary preservation, and local NameCert exhaustion.

Formal verification: FormalStatus(NameCert$_{\text{AxisCarryRefusalLedger}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.AxisCarryRefusalLedgerUp.AxisCarryRefusalLedgerCarrier\_bridge\_boundary\_preservation`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove Lean verification, bridge checking, ZCarry-to-hsame identification, FullAxis-to-Real identification, AxisNat replacement of Nat, or any hidden host quotient.

Upgrade path: Bridge closure requires a checked standard bridge over the same public carrier-facing route, negative-row exactness, bridge-boundary preservation, transport, provenance, and local naming rows.

Chapter 1197

A Concrete Naming Certificate for $\mathsf{TerminationRefusalBoundary}^{\uparrow}$

The $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ packet realizes the halting distinction of chapter 9368 as a finite NameCert surface that separates positive finite Cont traces from refusal-to-decide rows. It sits beside chapter 496, chapter 490, chapter 508, and chapter 854: those sibling packets record diagonal fixed-point pressure, obstruction ledgers, permitted finite refutation, and certificate truth-branch boundaries, while this carrier stores the positive trace row and the refusal row without converting either into a global halting decision.

Definition 1197.1 (Termination-refusal boundary carrier)**.** *A* $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, I, T, F, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source certificate row naming the local* $\mathsf{NameCert}_P$ *whose continuation chain is being inspected;*
2. *I is the input-history row naming the starting history h;*
3. *T is the positive finite trace row, either displaying a finite* Cont_P *chain ending at a terminal history or recording that no such displayed trace is present;*
4. *F is the finite trace verifier row, checking that every displayed step in T is a local* Cont_P *step and that a positive trace ends at the stated terminal history;*
5. *U is the refusal-to-decide row, recording that absence of a displayed finite trace is not reclassified as divergence, nontermination, or a total halting answer;*
6. *H records componentwise* hsame *transport for source certificates, input histories, trace rows, verifier rows, refusal rows, routes, provenance, and local names;*
7. *C records the displayed* Cont *routes by which consumers read S, I, T, F, U, H as one packet;*
8. *P is the* Pkg *provenance over chapter 9368, chapter 496, chapter 490, chapter 508, chapter 854,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TerminationRefusalBoundary}^{\uparrow}}$ *row.*

The carrier stores no divergence oracle, no unbounded search over continuation histories, no certificate-indexed truth branch, and no host evaluator for arbitrary certificate-history pairs.

Lean4 (definition) `BEDC.Derived.TerminationRefusalBoundaryUp.TerminationRefusalBoundaryCarrier`

Theorem 1197.2 (Termination-refusal boundary NameCert obligations)**.** *For the carrier of Definition 1197.1, the local* $\mathsf{NameCert}_{\mathsf{TerminationRefusalBoundary}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed packet* $R = (S, I, T, F, U, H, C, P, N)$*;*
2. *source and input exactness, requiring every accepted read to preserve the certificate row* S *and input-history row* I*;*
3. *positive trace exactness, requiring a positive termination witness to be read only from the finite trace row* T *together with the verifier row* F*;*
4. *refusal exactness, requiring every no-trace read to remain in the refusal row* U *rather than becoming a divergence classifier;*
5. *transport and provenance closure, requiring all comparisons, consumer routes, package provenance, and local naming to be exactly the displayed rows* H, C, P, N*.*

Thus an accepted packet may expose a positive finite trace or a refusal boundary, but it cannot decide the global halting predicate of Definition 9368.2.

Lean4 ✓ `BEDC.Derived.TerminationRefusalBoundaryUp.TerminationRefusalBoundaryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1197.1. The accepted rows are exactly $S, I, T, F, U, H, C, P, N$.

Source and input exactness keep the read attached to the same local certificate and starting history. A consumer therefore cannot change the inspected $\mathsf{NameCert}_P$ or history h while preserving the packet.

Positive trace exactness reads T together with F. If T displays a finite Cont_P chain, the verifier row checks only the listed steps and the stated terminal endpoint.

Refusal exactness reads U only when no displayed finite trace is being exported. The row records refusal to decide from missing finite evidence; it does not assert divergence, nontermination, or a total negative answer.

Finally H, C, P, N transport, replay, package, and name the same finite surface. The sibling references supply provenance for the diagonal, obstruction, finite-refutation, and truth-branch boundary contexts, but none adds an unbounded search, host evaluator, or global halting classifier. □

Theorem 1197.3 (Termination-refusal boundary non-escape)**.** *Every accepted* $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *packet exports only the displayed rows* $S, I, T, F, U, H, C, P, N$*. A consumer may read a verified finite* Cont *trace from* T, F *or a refusal-to-decide boundary from* U*, but it cannot obtain a divergence oracle, a total halting predicate, an unbounded continuation search, a host evaluator, or a certificate-indexed truth branch from the packet.*

Lean4 ✓ `BEDC.Derived.TerminationRefusalBoundaryUp.TerminationRefusalBoundaryCarrier nonescape`

Proof. Project the carrier and the NameCert obligations. The source, input, trace, verifier, refusal, transport, route, provenance, and local naming rows are the whole accepted surface.

For a positive trace read, the consumer must pass through T and F. The verifier row checks only the displayed finite continuation chain, so the read is a finite witness rather than an evaluator for other histories.

For a no-trace read, the consumer must pass through U. That row is a refusal boundary and not a negative halting proof.

The listed escape routes require rows absent from R: a divergence oracle, total classifier, unbounded search surface, host evaluator, or certificate-indexed truth branch. Since no such coordinate is carried, no accepted consumer route can export one. □

Theorem 1197.4 (Termination-refusal boundary independence)**.** *The* $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *carrier is not bijective to the carrier of chapter 496, chapter 490, chapter 508, or chapter 854 while preserving local* NameCert *obligations. Its distinguishing data are the simultaneous presence of a positive finite trace row* T, F *and a refusal-to-decide row* U *for the same source certificate and input history.*

Lean4 ✓ `BEDC.Derived.TerminationRefusalBoundaryUp.TerminationRefusalBoundary trace refusal row separation`

Proof. Compare the sibling roles one at a time. The halting diagonal packet carries fixed-point diagonal pressure, while the termination-refusal boundary must additionally store the positive trace verifier and refusal row for one inspected continuation chain.

The halting obstruction packet carries an obstruction ledger, not the paired positive finite trace and refusal-to-decide rows as one carrier. A bijection preserving NameCert obligations would have to send the verifier row F and refusal row U to distinct displayed coordinates, and those coordinates are not both present there.

The finite-refutation boundary carries a permitted falsity route and optional local decision row for a proposition, not a continuation trace verifier paired with refusal to decide halting. The certificate truth-branch boundary carries a branch-refusal surface, not a positive trace verifier for Cont_P chains.

Thus no listed sibling preserves $S, I, T, F, U, H, C, P, N$ with the same obligation surface. The carrier remains a separate finite boundary between positive termination evidence and refusal to decide. □

Theorem 1197.5 (Termination-refusal boundary TasteGate obligation)**.** *The Lean TasteGate file for* $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *will include explicit* Nontrivial $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *and* FieldFaithful $\mathsf{TerminationRefusalBoundary}^{\uparrow}$ *instances for the same carrier rows* $S, I, T, F, U, H, C, P, N$*. These instances must preserve the distinction between positive finite trace rows and refusal-to-decide rows.*

Lean4 ✓ `BEDC.Derived.TerminationRefusalBoundaryUp.TerminationRefusalBoundaryTasteGate single carrier alignment`

Proof. Nontriviality is forced by the separation of T, F from U. A carrier with a verified finite trace and a carrier with only the refusal row cannot collapse without losing the distinction recorded by Definition 9368.1.

Field faithfulness records preservation of each displayed row. In particular, it keeps source certificate, input history, positive trace, verifier, refusal, transport, route, package, and local naming data visible to the same carrier, so the TasteGate surface cannot identify a refusal boundary with a positive termination witness. □

Falsifiable prediction: A packet with no displayed finite Cont trace may carry a refusal boundary but cannot be read as a positive termination witness.

Independence witness: haltingdiagonal, haltingobstruction, finiterefutationboundary, certificatetruthbranchboundary: no sibling is bijective because this carrier stores both positive finite trace rows and refusal-to-decide rows while preserving their separation.

Closure status for $\mathsf{NameCert}_{\mathsf{TerminationRefusalBoundary}^{\uparrow}}$.

Constructive story: The termination-refusal boundary is generated from $BHist$ rows that name a boundary carrier, a refusal verdict, and the TasteGate obligation that consumes that verdict. The carrier is built bottom-up: first the history records the termination claim under inspection, then a refusal row marks the exact point where the claim leaves the admitted route, and finally Cont extensions may carry that refusal forward without changing its boundary. The $hsame$ relation keeps the verdict stable across equivalent histories. Non-escape and independence then show that the boundary is not a disguised termination proof, but a visible certificate of where such a proof is absent.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TerminationRefusalBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the termination-refusal boundary carrier, NameCert obligations, non-escape, independence, and TasteGate-obligation theorems.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TerminationRefusalBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim termination, decidability of all termination questions, or a route through any proof object outside the refusal boundary.

Upgrade path: Reaching obligationClosure requires a formal refusal-boundary carrier and checked non-escape for routes that omit the refusal row.

The $\mathsf{CofinalModulusSeal}^{\uparrow}$ hub routes the cofinal-tail modulus seal through chapter 698, chapter 1171, chapter 1164, chapter 386, chapter 126, chapter 301, chapter 291, and chapter 38.

Chapter 1198

A Concrete Naming Certificate for $\mathsf{CofinalModulusSeal}^{\uparrow}$

The $\mathsf{CofinalModulusSeal}^{\uparrow}$ packet realizes the cofinal-tail modulus route of chapter 9238 as one finite completion-seal naming surface. It sits after the shared threshold budget of chapter 698, the tail-modulus seal route of chapter 1171, the window-budget route of chapter 1164, and the terminal Cauchy-limit seal of chapter 386. The packet is not a Real-completeness principle: it records the finite rows a consumer must expose before the Real-completion seal can be read.

Falsifiable prediction: Every accepted CofinalModulusSealUp consumer must expose the cofinal tail budget, CauchyTailModulusSeal, RealWindowBudget, CauchyLimitSeal, RegSeqRat, StreamName, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows before any Real-completion seal read is public.

Independence witness: cofinaltailbudget, cauchytailmodulusseal, realwindowbudget, cauchylimitseal, real: no sibling is bijective because each supplies only a proper route segment while CofinalModulusSealUp binds the cofinal modulus-to-seal route in one local packet.

Definition 1198.1 (Cofinal modulus seal carrier)**.** *A* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, W, L, R, T, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CofinalTailBudget}^{\uparrow}$ *row fixing the shared cofinal-tail threshold;*
2. *M is the* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *row that binds the shared modulus route to the tail seal;*
3. *W is the* $\mathsf{RealWindowBudget}^{\uparrow}$ *row exposing the finite observation window budget for the same route;*
4. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row consumed only after B, M, W have fixed the cofinal modulus route;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the displayed tail windows;*
6. *T is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by the cofinal threshold and the window budget;*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback ledger attached to T;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through R, D, L;*
9. *H records componentwise* hsame *transport for the route rows and the local naming row;*

10. *C records displayed* Cont *replay from B, M, W through T, D, R, L, E;*
11. *P is the* Pkg *provenance over* CofinalTailBudget$^{\uparrow}$, CauchyTailModulusSeal$^{\uparrow}$, RealWindowBudget$^{\uparrow}$, CauchyLimitSeal$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{CofinalModulusSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, an undisplayed completion seal, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealTasteGate single carrier alignment`

Theorem 1198.2 (Cofinal modulus seal NameCert obligations)**.** *For the carrier of Definition 1198.1, the local* NameCert$_{\text{CofinalModulusSeal}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for $S = (B, M, W, L, R, T, D, E, H, C, P, N)$;*
2. *cofinal route synchronization, stating that B, M, W expose one shared tail threshold, tail-modulus seal, and observation-window budget;*
3. *completion-seal handoff, stating that L, R, T, D, E are read only through the synchronized cofinal route;*
4. *structural replay, stating that H, C, P transport, replay, and source exactly the displayed coordinates;*
5. *non-escape, stating that every public consumer remains inside the displayed* NameCert$_{\text{CofinalModulusSeal}^{\uparrow}}$ *row N and cannot import quotient equality, host equality, selected limits, countable choice, or ambient completeness.*

Proof. Project the carrier. The tuple lists the cofinal tail budget, Cauchy tail-modulus seal, Real window budget, Cauchy-limit seal, RegSeqRat handoff, StreamName window, DyadicRatCore ledger, RealUp seal, transport, continuation, provenance, and local naming rows.

Read the three route-source rows next. The CofinalTailBudget$^{\uparrow}$ sibling fixes the shared cofinal threshold, the CauchyTailModulusSeal$^{\uparrow}$ sibling binds the modulus route to the tail seal, and the RealWindowBudget$^{\uparrow}$ sibling exposes the finite observation window budget for the same route.

Now attach the completion-seal side. The CauchyLimitSeal$^{\uparrow}$ row L can be consumed only after B, M, W have fixed the route, and the rows T, D, R, E display the StreamName, dyadic, RegSeqRat, and RealUp faces of that handoff.

The rows H, C, P, N only transport, replay, source, and name the displayed packet. Since no excluded object is a coordinate of the carrier, the five obligation rows exhaust the local NameCert surface. □

Theorem 1198.3 (Cofinal modulus seal RegSeqRat window factorization)**.** *Every* RegSeqRat$^{\uparrow}$ *handoff accepted by* CofinalModulusSeal$^{\uparrow}$ *factors through the same* StreamName$^{\uparrow}$ *window row T and* DyadicRatCore$^{\uparrow}$ *tolerance row D selected by the cofinal tail budget B and the* RealWindowBudget$^{\uparrow}$ *row W.*

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealRegSeqRatWindowFactorization`

Proof. Project the carrier of Definition 1198.1. The RegSeqRat row R is reached from the displayed tail windows, while T is selected by the cofinal threshold and window budget and D is attached to that selected window.

Apply Definition 1198.6 and Definition 1198.7. The shared-route theorem binds B, M, W before the completion side is public, and the window-handoff theorem places T, D, R behind the same B and W rows.

Finally use Definition 1198.9 and Definition 1198.10. These identify the selected window and tolerance ledger as the only public window route for R, so a RegSeqRat handoff cannot factor through a detached StreamName, a second dyadic tolerance row, or an independent RealWindowBudget. □

Theorem 1198.4 (Cofinal modulus seal tail-budget transport)**.** *Componentwise* hsame *transport in an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet preserves the cofinal tail budget B, the* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *row M, and the* $\mathsf{RealWindowBudget}^{\uparrow}$ *row W together as one source route.*

Proof. Start with the source-row synchronization theorem. Definition 1198.9 states that B, M, W are synchronized as one cofinal tail budget, Cauchy tail-modulus seal, and Real window budget before any downstream row is consumed.

Read the transport coordinate H from the carrier. By the obligation surface in Definition 1198.2, H records componentwise hsame transport for the displayed route rows and the local naming row, rather than a separate source selector.

The window-budget exactness theorem then fixes the downstream use of the transported source. Definition 1198.10 makes T, D, R consume the same B, W pair, while the shared-route obligation keeps M on that source route. Thus transport preserves B, M, W together and cannot move only one of the three rows. □

Theorem 1198.5 (Cofinal modulus seal NameCert route coverage)**.** *The local* $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}^{\uparrow}}$ *route covers exactly the displayed rows*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

It cannot import quotient stream equality, host real equality, selected limits, countable choice, or ambient Real completeness.

Proof. First use Definition 1198.2. Its carrier admission, route synchronization, completion handoff, structural replay, and non-escape clauses are all written over the rows of Definition 1198.1.

Next combine the route theorems. Definition 1198.6 fixes the source rows B, M, W, Definition 1198.3 factors the RegSeqRat handoff through T, D, and Definition 1198.4 keeps source transport componentwise.

It remains to exclude external completion data. The completion non-escape theorem, Definition 1198.8, removes quotient stream equality, countable choice, selected host limits, host real equality, and ambient Real completeness from the consumer boundary. Therefore the local NameCert route covers exactly the displayed finite rows and nothing beyond them. □

Theorem 1198.6 (Cofinal modulus seal shared route obligation)**.** *For the carrier of Definition 1198.1, rows B, M, W must name one shared cofinal route before T, D, R, L, E are public. A completion-facing consumer may not read the StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, or RealUp rows from separate cofinal sources.*

Proof. Project the carrier $S = (B, M, W, L, R, T, D, E, H, C, P, N)$. The source side of the packet is exactly the cofinal tail budget row B, the Cauchy tail-modulus seal row M, and the Real window budget row W.

Apply Definition 1198.2. Its cofinal route synchronization clause binds those three rows as one shared threshold, tail-modulus seal, and observation-window budget before the completion-seal handoff is readable.

The sibling chapters named in the introduction supply only their local route segments: $\mathsf{CofinalTailBudget}^{\uparrow}$ supplies the budget, $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ supplies the modulus seal, and $\mathsf{RealWindowBudget}^{\uparrow}$ supplies the window budget. The support rows H, C, P, N preserve that same route, so T, D, R, L, E cannot be public from separate sources. □

Theorem 1198.7 (Cofinal modulus seal window handoff obligation)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet reads through the same* $\mathsf{CofinalTailBudget}^{\uparrow}$ *row* B *and* $\mathsf{RealWindowBudget}^{\uparrow}$ *row* W.

Proof. Read the displayed rows in Definition 1198.1. The StreamName row T is selected by the cofinal threshold and window budget, the DyadicRatCore row D is attached to that selected window, and the RegSeqRat row R is reached from the displayed tail windows.

Apply Definition 1198.2 and then Definition 1198.6. The first theorem supplies the local certificate clauses, and the second locks B, M, W as the shared source route before T, D, R are public.

The introduction's StreamName, DyadicRatCore, and RegSeqRat sibling routes therefore enter the packet only through the same cofinal tail budget and Real window budget rows. Since the carrier has no second B or W coordinate, every handoff reads through that single pair. □

Theorem 1198.8 (Cofinal modulus seal completion non-escape obligation)**.** *No completion-facing consumer accepted by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *can import quotient stream equality, countable choice, selected host limits, host real equality, or ambient Real completeness. The consumer boundary is the displayed route*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Proof. First project the carrier and use Definition 1198.2. The carrier admission and non-escape clauses enumerate the public surface and exclude quotient equality, host equality, selected limits, countable choice, and ambient completeness as local NameCert coordinates.

Next apply Definition 1198.6 and Definition 1198.7. They force the StreamName, DyadicRatCore, and RegSeqRat handoffs through the same B and W rows before the CauchyLimitSeal and RealUp rows can be consumed.

Finally use the CauchyLimitSeal and RealUp sibling routes named in the introduction. Those siblings receive only the displayed finite handoff $B, M, W, T, D, R, L, E, H, C, P, N$ from this packet. Since no excluded completion principle appears in the carrier, the accepted consumer cannot escape the finite route. □

Theorem 1198.9 (Cofinal modulus seal source-row synchronization)**.** *Every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet must name one shared cofinal source route before any of the rows* T, D, R, L, E *can be consumed. In particular, the rows* B, M, W *are synchronized as the cofinal tail budget, Cauchy tail-modulus seal, and Real window budget for the same displayed route.*

Proof. Project the carrier of Definition 1198.1. The source side contains exactly the rows B, M, W before the StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, and RealUp rows are read.

Apply Definition 1198.2. Its cofinal route synchronization row states that B, M, W expose one shared threshold, tail-modulus seal, and observation-window budget in the same packet.

The support rows H, C, P, N transport, replay, source, and name that packet without adding another route source. Therefore any consumer of T, D, R, L, E must first pass through the synchronized B, M, W route. □

Theorem 1198.10 (Cofinal modulus seal window-budget exactness)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{RegSeqRat}^{\uparrow}$ *window read exported by an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet is routed through the same* $\mathsf{RealWindowBudget}^{\uparrow}$ *row* W *and* $\mathsf{CofinalTailBudget}^{\uparrow}$ *row* B*.*

Proof. Start with the synchronized source route of Definition 1198.9. It fixes B, M, W before any read of T, D, R is public.

The StreamName row T is selected by the cofinal threshold and the window budget, the DyadicRatCore row D is attached to that selected window, and the RegSeqRat row R is reached from the displayed tail windows. Thus the three readbacks share the same B and W rows.

No alternate window-budget coordinate appears in the carrier. The rows H, C, P, N preserve the displayed route, so every exported window read is exact for the same RealWindowBudget and CofinalTailBudget rows. □

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealWindowBudgetExactness`

Theorem 1198.11 (Cofinal modulus seal completion readback boundary)**.** *Every completion-facing readback from an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet passes through* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The packet does not claim Real completeness.*

Proof. Use Definition 1198.9 to locate the source route and Definition 1198.10 to locate the window reads. Together they force completion-facing readback through the displayed M, L, R, T, D, E rows and the synchronized source rows.

Now apply the existing Real handoff in Definition 1198.13. The Real-facing consumer reaches E only through $B, M, W, T, D, R, L, E, H, C, P, N$.

Finally use non-escape from Definition 1198.14. Since the carrier has no coordinate for host completeness, quotient equality, selected limits, or countable choice, the readback boundary is the finite packet rather than a Real-completeness principle. □

Theorem 1198.12 (Cofinal modulus seal cofinal route)**.** *Every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet exposes one cofinal modulus-to-seal route*

$$B, M, W \to T, D, R, L, E$$

through the support rows H, C, P, N*. A public route that omits the cofinal tail budget, the Cauchy tail-modulus seal, or the Real window budget is outside the packet.*

Proof. Start from the NameCert obligations. The cofinal route synchronization row ties B, M, W to the same accepted packet before any completion-facing row is read.

Use the sibling boundaries in order. The cofinal tail budget supplies the shared threshold, the Cauchy tail-modulus seal binds the threshold to the modulus route, and the Real window budget exposes the selected observation window and regular handoff surface.

It remains to identify the public route. The completion-seal handoff row reads T, D, R, L, E only downstream of B, M, W, while H, C, P, N preserve the same packet. If any of B, M, W is omitted, no displayed route reaches the seal side. □

Theorem 1198.13 (Cofinal modulus seal Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet factors through the displayed rows*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

The handoff exposes the $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{CauchyTailModulusSeal}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *rows have been read on the same route.*

Proof. Project the carrier to the Real-facing rows. The only RealUp coordinate is E, and it appears after the StreamName window T, dyadic ledger D, RegSeqRat handoff R, and Cauchy-limit seal L.

Apply Definition 1198.12. That theorem places T, D, R, L, E downstream of the synchronized source rows B, M, W, so no Real-facing consumer can reach E from a detached window, a private modulus, or an independent limit seal.

Finally read the support rows. Transport through H, replay through C, provenance through P, and local naming through N preserve exactly the same route. Thus the Real handoff is the displayed finite packet rather than a host completeness assertion. □

Theorem 1198.14 (Cofinal modulus seal non-escape)**.** *No accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *consumer can read beyond*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

In particular, the packet exports no quotient stream equality, host real equality, selected host limit, countable-choice selector, undisplayed completion seal, independent RealUp equality row, or ambient Real-completeness principle.

Proof. Use the NameCert obligations to enumerate the public surface. Carrier admission exposes only $B, M, W, L, R, T, D, E, H, C, P, N$, and the route synchronization and completion-seal handoff rows keep the consumer on those coordinates.

For StreamName, DyadicRatCore, and RegSeqRat reads, the route stops at T, D, R after the cofinal source rows have been displayed. For RealUp and completion-facing reads, the route continues through L, E and then through the same transport, continuation, provenance, and local naming rows.

The excluded objects require coordinates absent from the carrier. Since H, C, P, N only preserve and name the displayed packet, no consumer can reinterpret the cofinal modulus seal as quotient equality, selected limit data, countable choice, independent Real equality, or ambient completeness. □

Corollary 1198.15 (Cofinal modulus seal strict obstruction)**.** *Let*

$$S = (B, M, W, L, R, T, D, E, H, C, P, N)$$

be an accepted $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet. Suppose a completion-facing,* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing,* $\mathsf{StreamName}^{\uparrow}$*-facing, or* $\mathsf{DyadicRatCore}^{\uparrow}$*-facing consumer displays a failure, refusal, separation, or countermodel row against a named converse or reuse route for* S*. Then that displayed row is a strict local obstruction: it blocks only the named route whose consumer-facing read factors through*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

It rejects every unlisted coordinate and cannot be reinterpreted as a global impossibility theorem, a host theorem, hidden external validation of the blocked route, quotient stream equality, countable choice, a selected host limit, host real equality, an undisplayed completion seal, or ambient Real completeness.

Proof. Start with the local obligation surface of Definition 1198.2. Its carrier-admission row lists exactly the displayed packet coordinates, its cofinal route synchronization row binds B, M, W as the shared source route, its completion-seal handoff row places L, R, T, D, E downstream of that route, its structural replay row restricts H, C, P to transport, replay, and provenance over those same coordinates, and its non-escape row keeps every public consumer inside the local $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}^{\uparrow}}$ row N.

Apply the completion non-escape obligation, Definition 1198.8. It says that no completion-facing consumer accepted by $\mathsf{CofinalModulusSeal}^{\uparrow}$ can import quotient stream equality, countable choice, selected host limits, host real equality, or ambient Real completeness, and that the consumer boundary is the displayed route

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Thus every consumer-facing use of a displayed failure, refusal, separation, or countermodel row must be read as a row of this finite route. The row may obstruct the named converse or reuse route exactly where that route asks to read, replace, or bypass one of the displayed coordinates, but it has no access to any coordinate outside the route.

A stronger interpretation would require data not exported by the packet. A global impossibility theorem would require a theorem surface beyond the local $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}^{\uparrow}}$ boundary; a host theorem would require host equality, host limits, host completeness, or a host proof channel; hidden external validation would require an off-packet certificate route; and quotient stream equality, countable choice, selected host limits, host real equality, undisplayed completion seals, and ambient Real completeness are precisely the coordinates excluded by Definition 1198.8. Since Definition 1198.2 supplies only the displayed finite rows, the obstruction remains strict and local to the named converse or reuse route. □

Theorem 1198.16 (Cofinal modulus seal window lock)**.** *Before a* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *completion seal is public, the cofinal tail budget,* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows must all be displayed in the same finite carrier.*

Proof. Project the carrier of Definition 1198.1. Its rows are exactly $B, M, W, L, R, T, D, E, H, C, P, N$, so the cofinal budget, tail-modulus seal, window budget, StreamName window, dyadic ledger, regular-sequence handoff, limit seal, Real seal, transport, replay, provenance, and local name are all visible before the public read.

Apply Definition 1198.2 and Definition 1198.12. They lock the seal side T, D, R, L, E downstream of the synchronized B, M, W source route.

Finally use Definition 1198.14. No hidden completion seal, ambient completeness witness, quotient equality, or choice selector can supply a missing row, so the displayed finite carrier is the public window lock. □

Theorem 1198.17 (Cofinal modulus seal tail readback totality)**.** *Every tail readback accepted by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *is total only on the displayed rows* $B, M, W, T, D, R, L, E, H, C, P, N$*. In particular, the tail budget, Cauchy tail-modulus seal, Real window budget, StreamName, DyadicRatCore, RegSeqRat, Cauchy limit seal, RealUp, hsame, Cont, Pkg, and NameCert rows must be present before the readback can be consumed.*

Proof. Use Definition 1198.10. It routes StreamName, DyadicRatCore, and RegSeqRat reads through the same cofinal tail budget and Real window budget.

Then apply the Real handoff of Definition 1198.13. The readback continues to the CauchyLimitSeal and RealUp rows only through the same support rows H, C, P, N.

The TasteGate requirement for this chapter asks the later Lean carrier to keep Nontrivial and FieldFaithful instances for the same finite surface. In the paper carrier this means totality is row-total on $B, M, W, T, D, R, L, E, H, C, P, N$, not a route through ambient Real completeness. □

Theorem 1198.18 (Cofinal modulus seal completion consumer boundary)**.** *A completion-facing consumer of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *may consume the seal only after the cofinal tail budget,* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows have all been displayed. The consumer boundary contains no choice row, quotient stream equality, or ambient completeness premise.*

Proof. Start from Definition 1198.11. It already places completion-facing readback on the finite sequence of cofinal source, StreamName, dyadic, regular-sequence, CauchyLimitSeal, RealUp, transport, replay, provenance, and naming rows.

Now combine the window lock with tail readback totality. The first theorem requires the rows to be displayed before the seal is public, and the second makes the readback total only on that row inventory.

Finally apply non-escape. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, or ambient completeness, every completion consumer stops at the displayed finite boundary. □

Theorem 1198.19 (Cofinal modulus seal tail cofinality lock)**.** *Every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet locks a completion read to the shared cofinal tail budget before the read may inspect the seal side. In particular, a public completion read must pass through the* $\mathsf{CofinalTailBudget}^{\uparrow}$*,* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}^{\uparrow}}$ *rows in the same finite carrier.*

Proof. Project the carrier of Definition 1198.1. The rows B, M, W form the source side of the cofinal tail route, while T, D, R, L, E form the StreamName, dyadic, regular-sequence, Cauchy-limit, and Real-facing seal side.

Use source-row synchronization and window-budget exactness. Definition 1198.9 identifies B, M, W as one shared cofinal route, and Definition 1198.10 routes the T, D, R windows through that same budget.

The Real handoff then places L and E downstream of the synchronized rows, while H, C, P, N preserve transport, replay, provenance, and local naming. Thus the completion read has no entry point except the displayed cofinal tail lock. □

Theorem 1198.20 (Cofinal modulus seal shared budget exhaustion)**.** *For every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *consumer, the shared budget rows* B, M, W *exhaust the admissible cofinal-tail supply for the downstream rows* $T, D, R, L, E, H, C, P, N$*. No consumer may replace that supply by a private modulus, a detached RealWindowBudget, a hidden StreamName, quotient stream equality, selected host limits, or ambient Real completeness.*

Proof. Begin with the NameCert obligations. They admit only the displayed carrier $S = (B, M, W, L, R, T, D, E, H$ and require cofinal route synchronization before completion-seal handoff.

The tail cofinality lock above gives the budget use required by each completion read. The lock forces $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{CauchyTailModulusSeal}^{\uparrow}$, and $\mathsf{RealWindowBudget}^{\uparrow}$ to be the same route source for StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, and RealUp reads.

Finally apply non-escape. Since the carrier has no coordinate for a private modulus, detached window budget, hidden stream selector, quotient equality, selected host limit, or ambient completeness principle, the shared budget rows exhaust the public supply. □

Theorem 1198.21 (Cofinal modulus seal completion scope factorization)**.** *Every completion-facing scope exported by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *factors through the finite row sequence*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Thus downstream $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *consumers may cite only the cofinal budget, CauchyTailModulusSeal, RealWindowBudget, CauchyLimitSeal, hsame, Cont, Pkg, and NameCert rows displayed by the same accepted packet.*

Proof. First use the completion consumer boundary. It already states that a completion-facing consumer may consume the seal only after every row in the displayed carrier has been exposed.

Next apply shared budget exhaustion. The rows B, M, W supply the only admissible cofinal-tail budget, while T, D, R, L, E are the only StreamName, dyadic, regular-sequence, limit-seal, and Real-facing rows downstream of that budget.

The support rows H, C, P, N then factor the scope through componentwise transport, displayed continuation, provenance, and local naming. Hence the completion-facing export is a finite scope factorization, not a route through endpoint-only Real validation or hidden completion data. □

Theorem 1198.22 (Cofinal modulus seal root-unblock package)**.** *The carrier of Definition 1198.1, together with the window lock, tail readback totality, completion consumer boundary, tail cofinality lock, shared budget exhaustion, and completion scope factorization, packages* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *as the shared root surface for tail-budget,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers.*

Proof. Start with the carrier inventory. Its rows B, M, W expose the cofinal tail budget, Cauchy tail-modulus seal, and Real window budget before any downstream completion row is read.

Use the window lock and tail readback totality to display the consumer-facing route. These theorems require the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows to be present in the packet before a completion read is public.

Finally apply shared budget exhaustion and completion scope factorization. The first theorem states that B, M, W exhaust the admissible cofinal-tail supply, while the second factors every completion-facing scope through $B, M, W, T, D, R, L, E, H, C, P, N$. Thus all listed consumers share the same root surface. □

Theorem 1198.23 (Cofinal modulus seal formal-target handoff)**.** *The rows* $B, M, W, T, D, R, L, E, H, C, P, N$ *are the exact future Lean target surface for* $\mathsf{CofinalModulusSeal}^{\uparrow}$*. Any formal target for this packet must expose the cofinal budget row, Cauchy tail-modulus row, Real window-budget row, StreamName row, dyadic row, regular-sequence row, Cauchy-limit row, Real seal row, componentwise transport row, continuation replay row, provenance row, and local* NameCert *row, with no extra host-limit or choice coordinate.*

Proof. Apply Definition 1198.22. It packages the shared root surface for tail-budget, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers from the displayed carrier.

Use Definition 1198.21. That theorem factors every completion-facing scope through the same rows $B, M, W, T, D, R, L, E, H, C, P, N$.

Finally apply Definition 1198.29 and Definition 1198.30. The first gives no local naming coordinate beyond the displayed rows, and the second excludes choice, quotient, hidden selector, host-limit, and ambient-completeness coordinates from the window ledger. □

Theorem 1198.24 (Cofinal modulus seal dependency-ready package)**.** *The chapter exposes a dependency-ready package over* $\mathsf{CofinalTailBudget}^{\uparrow}$*,* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$*,* $\mathsf{RealWindowBudget}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Each dependency is read through the shared cofinal route of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *rather than through a detached completion premise.*

Proof. Start with Definition 1198.22. It names the downstream tail-budget, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers of the same root surface.

Next use Definition 1198.27. It keeps every tail-budget read inside the finite route $B, M, W, T, D, R, L, E, H, C, P$ and rejects host-selected limits, quotient streams, private moduli, hidden selectors, and ambient completeness.

The completion-scope factorization then supplies the $\mathsf{CauchyTailModulusSeal}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ rows as part of that same route. Thus the named dependencies are ready as one shared package, not as independent imports. □

Theorem 1198.25 (Cofinal modulus seal Lean-frontier readiness)**.** *Any Lean formalization of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *must preserve the shared cofinal route and the no-choice nonescape rows. In particular, it must keep* B, M, W *as the common cofinal source for* T, D, R, L, E *and must preserve* H, C, P, N *as support rows that do not introduce countable choice, quotient equality, selected limits, hidden* $\mathsf{StreamName}^{\uparrow}$ *selectors, or ambient Real completeness.*

Proof. Begin with Definition 1198.27. It makes all accepted tail-budget reads factor through the displayed finite route.

Apply Definition 1198.28. The readback route passes from B, M, W through T, D, R, L to E and then through H, C, P, N, so a formalization must preserve that order.

Finish with Definition 1198.30. The no-choice boundary rules out countable-choice selectors, hidden stream selectors, quotient equality, selected host limits, and ambient Real completeness. Therefore the Lean frontier is exactly the shared cofinal route with those nonescape rows intact. □

Theorem 1198.26 (Cofinal modulus seal StreamName–Real consumer route)**.** *Every public Real-completion consumer route through* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *passes through the displayed cofinal budget,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the same finite carrier. A consumer may not read* $\mathsf{Real}^{\uparrow}$ *from a detached stream name, a detached regular-sequence row, or a hidden cofinal budget.*

Proof. Use Definition 1198.18. It places every completion-facing consumer on the displayed sequence of cofinal source, StreamName, dyadic, regular-sequence, Cauchy-limit, Real seal, transport, replay, provenance, and naming rows.

Apply Definition 1198.22. The root package makes the cofinal budget row B, StreamName row T, RegSeqRat row R, and RealUp seal row E part of one public consumer surface.

The non-escape theorem then excludes detached stream names, private regular sequences, selected limits, quotient equality, and hidden completion data. Therefore a Real-completion consumer reaches $\mathsf{Real}^{\uparrow}$ only through the displayed cofinal-budget-to-StreamName-to-RegSeqRat-to-Real route. □

Theorem 1198.27 (Cofinal modulus seal tail-budget factorization)**.** *Every tail-budget read accepted by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *factors through the finite rows* $B, M, W, T, D, R, L, E, H, C, P, N$*. It cannot import a host-selected limit, quotient stream, private modulus, hidden StreamName selector, or ambient Real-completeness premise.*

Proof. Begin with Definition 1198.19. It locks each completion read to the shared cofinal tail budget before the seal side may be inspected.

Next apply Definition 1198.20. The rows B, M, W exhaust the public cofinal-tail supply for downstream StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, and RealUp reads.

Finish with Definition 1198.21. Since every completion-facing scope factors through $B, M, W, T, D, R, L, E, H, C$ no host-selected limit, quotient stream, private modulus, hidden selector, or ambient completeness row is available to a tail-budget consumer. □

Theorem 1198.28 (Cofinal modulus seal cofinal readback factorization)**.** *Every completion-facing readback from the source rows B, M, W to the Real-facing row E factors first through the seal rows T, D, R, L and then through the support rows H, C, P, N. Equivalently, a public* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *readback has the finite route*

$$B, M, W \longrightarrow T, D, R, L \longrightarrow E \longrightarrow H, C, P, N$$

and has no direct source-to-Real edge outside the displayed carrier.

Proof. Start with Definition 1198.19. It places every completion read behind the shared cofinal source route B, M, W before any StreamName, dyadic, regular-sequence, Cauchy-limit, or Real-facing row is public.

Next apply Definition 1198.21. The completion-facing scope is exactly the finite row sequence $B, M, W, T, D, R, L, E, H, C, P, N$, so the readback must pass through T, D, R, L before it reaches E.

Finally use Definition 1198.27. Its tail-budget factorization excludes host-selected limits, quotient streams, private moduli, hidden selectors, and ambient completeness, leaving only the support rows H, C, P, N to transport, replay, source, and name the completed readback. □

Theorem 1198.29 (Cofinal modulus seal route NameCert exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}^{\uparrow}}$ *route is exhausted by the rows*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Every accepted carrier, public completion readback, root-unblock consumer, and non-escape check for $\mathsf{CofinalModulusSeal}^{\uparrow}$ *is a statement about these rows, and no additional naming coordinate is available.*

Proof. Project Definition 1198.1. The carrier contains exactly the cofinal source rows B, M, W, the seal and readback rows T, D, R, L, E, and the support rows H, C, P, N.

Apply Definition 1198.2. Its carrier admission, route synchronization, completion handoff, structural replay, and non-escape clauses are all written over the displayed rows.

Use Definition 1198.22 to read the downstream package boundary. The root-unblock surface for tail-budget, StreamName, RegSeqRat, DyadicRatCore, and RealUp consumers is built from the same finite packet.

Finally apply Definition 1198.14. Since quotient equality, host equality, selected limits, countable choice, ambient completeness, and undisplayed completion seals are absent from the carrier, the local NameCert route has no coordinate beyond $B, M, W, T, D, R, L, E, H, C, P, N$. □

Theorem 1198.30 (Cofinal modulus seal window ledger no-choice boundary)**.** *Window-budget reads in an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet cannot introduce countable-choice selectors, hidden* $\mathsf{StreamName}^{\uparrow}$ *selectors, quotient stream equality, selected host limits, or ambient Real completeness. The window ledger is bounded by the shared rows B, W, T, D, R and by the non-escape boundary of the same carrier.*

Proof. Begin with Definition 1198.10. It routes every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ window read through the same $\mathsf{RealWindowBudget}^{\uparrow}$ row W and $\mathsf{CofinalTailBudget}^{\uparrow}$ row B.

Next use Definition 1198.20. The shared budget rows B, M, W exhaust the admissible cofinal-tail supply for the downstream rows $T, D, R, L, E, H, C, P, N$, so no private selector or detached window ledger can be inserted.

Finish with Definition 1198.14. The carrier has no coordinate for countable choice, hidden StreamName selection, quotient stream equality, selected host limits, or ambient Real completeness. Therefore the window ledger remains a finite no-choice boundary inside the displayed packet. □

Theorem 1198.31 (Cofinal modulus seal RegSeqRat route exhaustion)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read through an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet factors through the finite rows*

$$B, M, W, T, D, R, H, C, P, N.$$

In particular, the regular-sequence row is consumed only after the cofinal budget, modulus, window, StreamName, and dyadic rows have been displayed, and its public readback is transported, replayed, sourced, and named only through H, C, P, N*.*

Proof. Project the carrier of Definition 1198.1. The regular-sequence row R is not an independent source; it sits behind the cofinal source rows B, M, W and the StreamName and dyadic rows T, D.

Apply Definition 1198.10. It identifies the common window budget used by the StreamName, dyadic, and regular-sequence consumers, so a $\mathsf{RegSeqRat}^{\uparrow}$ read cannot bypass B, M, W, T, D.

Use Definition 1198.22 and Definition 1198.29. The downstream package and the local NameCert route have no naming coordinate beyond the displayed finite carrier rows.

Finish with Definition 1198.30. It excludes hidden selectors, quotient stream equality, selected host limits, and choice-like completion data, leaving H, C, P, N as the only structural support for the regular-sequence readback. □

Theorem 1198.32 (Cofinal modulus seal StreamName window consistency)**.** *In an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *row* T *and the window row* W *are consumed on the same cofinal route. No public route may read a StreamName selector from* T *while using a different window ledger than the displayed* W*.*

Proof. Start from Definition 1198.1. The carrier lists W before T in the same finite packet, so both rows are coordinates of one accepted cofinal route.

Apply Definition 1198.10. It binds the StreamName-facing window read to the same budget row W used by the dyadic and regular-sequence rows.

Then use Definition 1198.22 and Definition 1198.29. Any downstream root-unblock consumer of T is named only through the exhausted route of the same carrier.

Finally apply Definition 1198.30. A hidden StreamName selector or detached window ledger would be a choice-like coordinate outside the packet, so the accepted route must consume T and W together. □

Theorem 1198.33 (Cofinal modulus seal RealUp non-completeness boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *row* E *in an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet is only a finite seal handoff. It carries no Real-completeness premise, quotient-real equality, selected-limit operator, countable-choice selector, or ambient host-completion assumption.*

Proof. Inspect Definition 1198.1. The row E is displayed after the cofinal source, window, StreamName, dyadic, regular-sequence, and Cauchy-limit rows, and before the structural support rows H, C, P, N.

Apply Definition 1198.22. The Real-facing handoff is part of the finite downstream package for tail-budget, StreamName, RegSeqRat, DyadicRatCore, and RealUp consumers; it is not a separate completeness service.

Use Definition 1198.29 to exhaust the naming route. Since the NameCert has no coordinate outside $B, M, W, T, D, R, L, E, H, C, P, N$, the Real-facing row cannot import an external completion principle.

Finish with Definition 1198.30. Countable choice, quotient equality, selected host limits, and ambient Real completeness are all excluded from the same finite ledger, so E remains a handoff row rather than a Real-completeness claim. □

Theorem 1198.34 (Cofinal modulus seal tail-budget row)**.** *Every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *read exposes the shared tail-budget source* B, M, W *before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row is public. The public source row is the displayed cofinal route, not a detached modulus service or host-selected tail choice.*

Proof. Start with Definition 1198.9. It identifies B, M, W as the synchronized cofinal tail budget, tail-modulus seal, and window budget for the same route.

Use Definition 1198.10 and Definition 1198.22. The first places T, D, R behind the same B, W rows, and the second packages the downstream tail-budget, StreamName, RegSeqRat, DyadicRatCore, and RealUp consumers from the same finite carrier.

Finally apply Definition 1198.29. Since the route has no naming coordinate beyond $B, M, W, T, D, R, L, E, H, C,$ no public consumer can bypass the source row. □

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSeal tail budget row`

Theorem 1198.35 (Cofinal modulus seal window readback row)**.** *The window/readback segment* T, D, R *of an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet is exact only inside the same cofinal route. It cannot be replaced by a hidden* $\mathsf{StreamName}^{\uparrow}$ *selector, quotient stream, selected host limit, countable-choice row, or detached readback service.*

Proof. Use Definition 1198.10. It binds every StreamName, DyadicRatCore, and RegSeqRat window read to the same $\mathsf{RealWindowBudget}^{\uparrow}$ row W and $\mathsf{CofinalTailBudget}^{\uparrow}$ row B.

Apply Definition 1198.30. The window ledger excludes countable-choice selectors, hidden StreamName selectors, quotient stream equality, selected host limits, and ambient Real completeness.

The route exhaustion theorem, Definition 1198.29, leaves T, D, R as the only displayed window/readback segment. Thus exactness is internal to the same cofinal route. □

Theorem 1198.36 (Cofinal modulus seal dependency-unblock triad)**.** *The tail-budget row of Definition 1198.34, the window readback row of Definition 1198.35, the route exhaustion of Definition 1198.29, and the RealUp boundary of Definition 1198.33 make* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *dependency-ready for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers without asserting Real completeness.*

Proof. First combine the tail-budget and window-readback rows. They force every downstream consumer to enter through B, M, W and then read T, D, R inside the same cofinal route.

Next use route NameCert exhaustion. It says that accepted carriers, readbacks, root-unblock consumers, and non-escape checks all live on $B, M, W, T, D, R, L, E, H, C, P, N$.

Finally apply the RealUp non-completeness boundary. The row E remains a finite handoff rather than a Real-completeness premise, so downstream consumers can cite the package as a dependency surface without importing ambient completion data. □

Theorem 1198.37 (Cofinal modulus seal diagonal representative handoff)**.** *Every diagonal representative consumer of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *factors through the same finite route*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

The consumer may read the diagonal representative only after the cofinal budget, Cauchy tail-modulus seal, Real window budget, StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, RealUp, hsame, Cont, Pkg, *and local* NameCert *rows are all displayed in the carrier.*

Proof. Begin with Definition 1198.1. The carrier already displays the cofinal source rows B, M, W, the diagonal-facing rows T, D, R, L, E, and the support rows H, C, P, N in one packet.

Use Definition 1198.30. It prevents the diagonal read from importing a selected host limit, quotient equality, hidden selector, or countable-choice coordinate.

Apply Definition 1198.31 and Definition 1198.32. The regular-sequence and StreamName reads remain behind the same window ledger, so a diagonal representative consumer cannot detach D or R from B, M, W, T.

Finally use Definition 1198.33. The Real-facing row E is only a finite handoff, so the diagonal representative route is exactly $B, M, W, T, D, R, L, E, H, C, P, N$ and not a completeness premise. □

Theorem 1198.38 (Cofinal modulus seal tail-modulus handoff)**.** *The* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *row in an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet cannot be detached from* $\mathsf{CofinalTailBudget}^{\uparrow}$ *and* $\mathsf{RealWindowBudget}^{\uparrow}$*. Any public tail-modulus read must keep M between the cofinal budget row B and the window row W, and must continue through $T, D, R, L, E, H, C, P, N$ before reaching a named completion-facing consumer.*

Proof. Project the carrier of Definition 1198.1. The rows B, M, W are adjacent source rows for the shared cofinal route, while $T, D, R, L, E, H, C, P, N$ are downstream rows of the same packet.

Apply Definition 1198.9. It identifies the cofinal budget, tail-modulus seal, and window budget as one synchronized source route.

Next use Definition 1198.10. The downstream StreamName, DyadicRatCore, and RegSeqRat reads consume the same window row W, so the tail-modulus row M cannot be replaced by a detached modulus service.

Finish with Definition 1198.27 and Definition 1198.33. The factorization keeps every tail-budget read inside the displayed route, and the Real-facing handoff remains a finite seal rather than a Real-completeness premise. □

Theorem 1198.39 (Cofinal modulus seal finite route scope)**.** *The full public scope of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *is exactly the finite BEDC row sequence*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

It closes the cofinal-budget, tail-modulus, window, StreamName, dyadic, regular-sequence, Cauchy-limit, Real handoff, transport, replay, provenance, and local naming route, but it does not claim Real completeness, quotient-real equality, selected host limits, countable choice, or an ambient completion operator.

Proof. Start from Definition 1198.21. It states the completion-facing scope over the displayed route $B, M, W, T, D, R, L, E, H, C, P, N$.

Use Definition 1198.29. The local NameCert route has no naming coordinate beyond the same finite carrier rows.

Apply Definition 1198.30. It removes choice, quotient equality, selected limits, hidden selectors, and ambient Real completeness from the window ledger.

Finally apply Definition 1198.33. The RealUp row is a finite handoff row only, so the full scope is the displayed BEDC route and not a Real-completeness theorem. □

Theorem 1198.40 (Cofinal modulus seal budget-window regularity chain)**.** *Every accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet first synchronizes the cofinal budget row B, tail-modulus row M, and window-budget row W. Only after that source synchronization may it read the StreamName row T, DyadicRatCore row D, and RegSeqRat row R, and those reads remain inside the same window route.*

Proof. Apply Definition 1198.9. It makes B, M, W a single synchronized source route for the accepted packet.

Next use Definition 1198.10. The window row W is the exact budget row consumed by the StreamName, DyadicRatCore, and RegSeqRat reads, so T, D, R cannot be read from a detached window service.

Finally apply Definition 1198.31. The regular-sequence readback has no coordinate beyond the displayed route, hence the readback segment T, D, R stays downstream of the same synchronized B, M, W source. □

Theorem 1198.41 (Cofinal modulus seal terminal Real seal factorization)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet factors through the finite BEDC route*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

The consumer reaches the terminal Real-facing seal only through the displayed cofinal budget, tail modulus, window, StreamName, dyadic, regular-sequence, Cauchy-limit, Real handoff, transport, replay, provenance, and local naming rows.

Proof. First use Definition 1198.40. It fixes the entry segment B, M, W and the readback segment T, D, R inside one cofinal route.

Apply Definition 1198.29. The remaining accepted carrier coordinates are precisely the Cauchy-limit row L, Real-facing row E, transport row H, replay row C, package row P, and local NameCert row N.

Finally use Definition 1198.33. The terminal Real-facing row is a finite seal and not a Real-completeness premise, so any terminal $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer factors through $B, M, W, T, D, R, L, E, H$ □

Theorem 1198.42 (Cofinal modulus seal open-phase exit surface)**.** *The* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *chapter supplies the Phase Real and Completion handoff surface for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers. This surface is the finite route $B, M, W, T, D, R, L, E, H, C, P, N$ and it does not claim Real completeness, quotient equality, selected limits, countable choice, or a* $\mathsf{StandardBridge}$.

Proof. Use Definition 1198.41. It makes the completion-facing and Real-facing consumer path factor through the full displayed route.

Next apply Definition 1198.39. The route already records the exact public scope of the chapter and excludes Real completeness, quotient-real equality, selected host limits, countable choice, and ambient completion operators.

The remaining bridge boundary follows from Definition 1198.33. Since E is only a finite Real-facing handoff row, the chapter exposes an exit surface for Phase Real and Completion consumers without asserting a StandardBridge. □

Theorem 1198.43 (Cofinal modulus seal observation-budget factorization)**.** *Every observation-budget read accepted by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *factors through the displayed finite row route*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

The read must expose the cofinal tail budget, tail-modulus seal, Real window budget, StreamName, dyadic tolerance, regular sequence, Cauchy-limit seal, Real handoff, componentwise transport, continuation replay, package, and local NameCert *rows before it can be consumed by a Real-facing or completion-facing observer.*

Proof. Start with Definition 1198.1. It displays $B, M, W, T, D, R, L, E, H, C, P, N$ as the only finite carrier rows available to an accepted packet.

Use Definition 1198.29 and Definition 1198.21. The first theorem exhausts the local naming route on exactly these rows, and the second factors completion-facing scope through the same row sequence.

Apply Definition 1198.10. The observation budget cannot detach T, D, R from B, M, W, because the StreamName, dyadic, and regular-sequence reads consume the same Real-window row.

Finally use Definition 1198.42. The public exit surface is the same finite route, so any observation-budget read for a Real-facing or completion-facing observer factors through $B, M, W, T, D, R, L, E, H, C, P, N$. □

Theorem 1198.44 (Cofinal modulus seal Real and Completion consumer exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet receives exactly the cofinal modulus route*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

It receives no quotient stream, selected host limit, countable-choice selector, ambient completeness row, hidden $\mathsf{StreamName}^{\uparrow}$ *selector, or detached window-budget service.*

Proof. Begin with Definition 1198.43. It gives the only observation-budget route available to Real-facing and completion-facing consumers.

Next apply Definition 1198.30. This removes quotient stream equality, selected host limits, countable-choice selectors, hidden StreamName selectors, and ambient Real completeness from the window ledger.

Use Definition 1198.33. The row E is a finite Real handoff and not a completeness premise, so a $\mathsf{Real}^{\uparrow}$ consumer receives only the displayed cofinal modulus route.

Finish with Definition 1198.41. Terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers already factor through the same rows, which makes the consumer boundary exact. □

Theorem 1198.45 (Cofinal modulus seal L10 exit boundary)**.** *The* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet is a Phase Real and Completion exit boundary over* $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{CauchyTailModulusSeal}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Its public boundary is the finite row route*

$$B, M, W, T, D, R, L, E, H, C, P, N,$$

and no consumer route exits through quotient streams, selected host limits, countable choice, ambient completeness, detached windows, or a separate completion operator.

Proof. First use Definition 1198.24. It places the named upstream and terminal rows in one dependency-ready package over the shared cofinal route.

Apply Definition 1198.43. The observation side of the exit boundary is forced through $B, M, W, T, D, R, L, E, H,$

Next use Definition 1198.44. The Real and Completion consumers receive exactly that route and no external quotient, choice, selected-limit, detached-window, or completeness row.

Finally apply Definition 1198.42. The packet is already the Phase Real and Completion handoff surface for the finite cofinal modulus route, so the displayed rows form the L10 exit boundary. □

Theorem 1198.46 (Cofinal modulus seal L10 four-object factorization)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet factors first through the cofinal source rows* B, M, W *and then through the Real L10 four-object handoff packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Equivalently, after the cofinal source rows are displayed, the consumer can use only the dyadic tolerance row D*, the finite stream-name row* T*, the regular rational sequence row* R*, and the Real handoff row* E*, together with* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support. It receives no quotient stream, host real equality, selected limit, countable choice, ambient completeness, detached window, or bridge coordinate.*

Proof. Begin with Definition 1198.45. It places the accepted packet on the Phase Real and Completion exit route and forces the public boundary through $B, M, W, T, D, R, L, E, H, C, P, N$.

Next apply Definition 1198.44. A Real-facing or completion-facing consumer receives exactly that route, so the first source data available to the consumer are the cofinal rows B, M, W, followed by the displayed terminal rows that feed the Real side.

Now use Definition 38.515. The Real-facing part of the route can be read as a four-object handoff through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with support confined to BHist, hsame, Cont, Pkg, and local NameCert rows.

Finally apply Definition 38.520. Since the Real handoff packet exports none of the forbidden coordinates, the cofinal seal consumer cannot acquire a quotient stream, host real equality, selected limit, countable-choice selector, ambient completeness row, detached window, or bridge coordinate after the factorization. □

Theorem 1198.47 (Cofinal modulus seal row-predicate admission)**.** *For an accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *packet, each displayed row*

$$B, M, W, T, D, R, L, E, H, C, P, N$$

is admitted only through the named BEDC primitives and sibling certificates carried by the cofinal modulus route. The rows are respectively read as cofinal tail budget, tail-modulus seal, Real-window budget, StreamName, DyadicRatCore tolerance, regular rational sequence, Cauchy-limit seal, Real handoff, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. No row may be replaced by an unnamed predicate, quotient-stream witness, selected host limit, detached window service, countable-choice selector, or ambient completeness premise.*

Proof. Start with Definition 1198.1. The carrier display names exactly the rows $B, M, W, T, D, R, L, E, H, C, P, N$ and gives their BEDC reading through BHist, hsame, Cont, Pkg, and NameCert data.

Use Definition 1198.9. It synchronizes the source-side rows with the same finite route, so the admission test cannot introduce a second classifier beside the displayed carrier rows.

Apply Definition 1198.10 and Definition 1198.30. These identify the window ledger rows and exclude detached windows, quotient streams, selected limits, countable choice, and ambient completeness.

Finally use Definition 1198.45. The exit boundary exposes the same finite row route to downstream consumers, so every admitted row is one of the named primitives or sibling certificates. □

Theorem 1198.48 (Cofinal modulus seal obligation surface packaging)**.** *The* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *route packages carrier admission, source synchronization, window ledger exactness, the Real and Completion handoff, nonescape boundaries, and consumer exactness as one finite* NameCert *obligation surface over*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Every obligation row in that surface is discharged by the displayed carrier route or by a cited sibling certificate, and no additional global completion, quotient, selector, or choice obligation is part of the package.

Proof. First combine Definition 1198.47 with Definition 1198.2. The former fixes the row predicates, while the latter places the local NameCert obligations on the same carrier route.

Next use Definition 1198.9 and Definition 1198.10. These two results bind the source rows and the window ledger rows to the displayed packet rather than to any external witness.

Apply Definition 1198.44. The Real-facing and completion-facing handoff consumes exactly the same route.

It remains to exclude escape obligations. Use Definition 1198.30 and Definition 1198.45. They remove quotient streams, selected limits, countable choice, detached windows, and ambient completeness from the public surface, leaving one finite NameCert obligation package. □

Theorem 1198.49 (Cofinal modulus seal consumer exactness obligation)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of the* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *obligation surface receives exactly the displayed route*

$$B, M, W, T, D, R, L, E, H, C, P, N$$

through the packaged carrier, source, window, handoff, nonescape, and exactness rows. The consumer receives no detached selector, quotient stream, countable-choice row, ambient completeness row, hidden completion operator, or separate window service beyond that finite route.

Proof. Begin with Definition 1198.48. It identifies the public obligation surface with the finite NameCert package over $B, M, W, T, D, R, L, E, H, C, P, N$.

Use Definition 1198.44. Real-facing and completion-facing consumers already receive exactly the cofinal modulus route and no external quotient, selector, choice, or completeness witness.

Apply Definition 1198.30. This keeps the window ledger from smuggling in a detached selector or separate completion row.

Finally use Definition 1198.45. The L10 exit boundary is the same finite route, so consumer exactness is an obligation of that surface and not a separate downstream premise. □

Theorem 1198.50 (Cofinal modulus seal scoped source inventory)**.** *The scoped source inventory of* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *is exactly the finite row family*

$$B, M, W, T, D, R, L, E, H, C, P, N$$

over $\mathsf{CofinalTailBudget}^{\uparrow}$, $\mathsf{CauchyTailModulusSeal}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Proof. Use the carrier definition together with Definition 1198.5. The named rows are precisely the admitted source, transport, package, and certificate rows of the displayed route.

Then apply Definition 1198.9. It identifies the route rows with the same BHist, hsame, Cont, Pkg, and NameCert inventory, so no further source primitive is available inside the scoped package. □

Theorem 1198.51 (Cofinal modulus seal primitive route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read from* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *factors through the displayed primitive rows* $B, M, W, T, D, R, L, E, H, C, P, N$ *No such read uses a quotient stream, selected host limit, countable choice, detached window, or ambient completeness coordinate.*

Proof. Apply Definition 1198.44. It already places each Real-facing and completion-facing read on the cofinal route.

Use Definition 1198.30 and Definition 1198.45. Together they rule out the excluded coordinates and leave only the primitive rows of the finite route. □

Theorem 1198.52 (Cofinal modulus seal scoped closure package)**.** *The obligation surface, route coverage, Real handoff, nonescape boundary, L10 exit boundary, Definition 1198.50, and Definition 1198.51 form the scoped* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *closure package.*

Proof. Combine Definition 1198.48, Definition 1198.5, Definition 1198.13, Definition 1198.14, and Definition 1198.45.

Finally add the scoped inventory and primitive exhaustion theorems. They identify the available sources and the allowed consumer reads, so the same finite route is a scoped closure package rather than an open obligation surface. □

Theorem 1198.53 (Cofinal modulus seal public route export)**.** *The finite route*

$$B, M, W, T, D, R, L, E, H, C, P, N$$

exports a reusable public $\mathsf{CofinalModulusSeal}^{\uparrow}$ *surface for downstream* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers. The public surface is exactly the scoped closure package, primitive route exhaustion, Real-completion consumer exactness, and L10 exit boundary; it does not assert Real completeness, quotient-stream equality, selected host limits, countable choice, detached windows, or a checked bridge.*

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealUp StdBridge`

Proof. First invoke Definition 1198.52. It packages the obligation surface, route coverage, Real handoff, nonescape boundary, L10 exit boundary, scoped source inventory, and primitive exhaustion on the same finite route.

Next use Definition 1198.51 together with Definition 1198.44. Every downstream $\mathsf{Real}^{\uparrow}$ or completion-facing read factors through the displayed route, so the route is public as a reusable consumer surface rather than a private local certificate.

Finally apply Definition 1198.45. The exclusions remain part of the exported surface: no Real completeness claim, quotient stream, selected host limit, countable-choice row, detached window, or bridge check is added by making the finite route public. □

Theorem 1198.54 (Cofinal modulus seal mature route package)**.** *The carrier, scoped closure package, primitive route exhaustion, public route export, and checked StdBridge row form the mature* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *package over the finite route*

$$B, M, W, T, D, R, L, E, H, C, P, N$$

The package remains internal to that route: it excludes Real completeness, quotient stream equality, selected host limits, countable choice, detached windows, and external completion operators.

Proof. Start with Definition 1198.52. It binds the carrier, obligation surface, route coverage, Real handoff, nonescape boundary, L10 exit boundary, scoped source inventory, and primitive exhaustion as one route package.

Use Definition 1198.53. The same package is exported as a reusable public surface for downstream $\mathsf{Real}^{\uparrow}$ and completion-facing consumers.

Now apply Definition 1198.51. It keeps every admitted read on the displayed primitive route and rules out quotient streams, selected host limits, countable choice, detached windows, and external completion operators.

Finally use the checked StdBridge row recorded by `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSe` The bridge row supplies the acceptance-bridge witness for the public surface, while the exclusions above keep the mature package finite and do not add Real completeness. □

Corollary 1198.55 (Cofinal modulus seal window-lock strict obstruction)**.** *Suppose a public* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *consumer displays a failure, refusal, separation, or countermodel against a route that tries to reuse the window lock outside the rows*

$$B, M, W, T, D, R, L, E, H, C, P, N.$$

Then the obstruction is strict and local: it blocks only the named converse or reuse route that omits, detaches, or replaces one of the displayed cofinal source, window, readback, Real-handoff, transport, replay, provenance, or local naming rows. It is not a global impossibility result, a host-completeness theorem, or hidden external validation of an undisplayed completion principle.

Proof. Start with Definition 1198.25. It identifies the rows that any formal or consumer-facing reading must preserve: B, M, W remain the common cofinal source for T, D, R, L, E, and H, C, P, N remain support rows that do not introduce countable choice, quotient equality, selected limits, hidden $\mathsf{StreamName}^{\uparrow}$ selectors, or ambient Real completeness.

Apply Definition 1198.21. Every completion-facing scope exported by the packet factors through exactly the displayed row sequence $B, M, W, T, D, R, L, E, H, C, P, N$. Hence a consumer-facing obstruction can only be evaluated against a route coordinate in that finite factorization or against an attempted reuse that bypasses it.

Now inspect the possible reuse attempts. If the attempted route detaches the window, replaces the cofinal source, skips T, D, R, L, E, or reads support from outside H, C, P, N, the obstruction blocks precisely that omitted or replaced coordinate. If the attempted route asks for quotient equality, a selected host limit, countable choice, hidden StreamName selection, ambient Real completeness, or an undisplayed completion seal, the Lean-frontier boundary says that no such coordinate belongs to the packet. In both cases the displayed row blocks only the attempted converse or reuse route.

The same factorization also prevents a stronger reinterpretation. Since every public completion-facing use has already been forced through the finite route, the obstruction has no remaining coordinate from which to assert a global impossibility theorem, no host-completeness premise from

which to prove a host result, and no external validation row hidden behind the window lock. Thus the window-lock obstruction is strict, local, and exhausted by the displayed carrier surface. □

Lean TasteGate formalization for this route must provide `Nontrivial CofinalModulusSealUp` and `FieldFaithful CofinalModulusSealUp` instances in `BEDC.Derived.CofinalModulusSealUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}\uparrow}$.

Constructive story: The cofinal modulus seal is generated from a large finite BHist route containing cofinal tail budget, tail-modulus seal, Real-window budget, StreamName, DyadicRatCore, RegSeqRat, CauchyLimitSeal, RealUp, hsame transport, Cont replay, Pkg provenance, and local NameCert rows. The carrier convention reads each row through hsame and connects them by explicit Cont handoffs, so a consumer receives the same cofinal modulus route under representative change. The factorization, synchronization, non-escape, and exhaustion theorems all refine the same bottom-up object: no quotient stream, selected host limit, detached window, or ambient completeness principle can enter unless it is already one of the displayed route rows.

Lean4 ✓ `BEDC.Derived.CofinalModulusSealUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1198.1, Definition 1198.2, Definition 1198.5, Definition 1198.13, Definition 1198.14, Definition 1198.45, Definition 1198.50, Definition 1198.51, Definition 1198.52, Definition 1198.53, and Definition 1198.54 bind the carrier, NameCert obligations, route coverage, Real handoff, nonescape surface, L10 exit boundary, scoped source inventory, primitive route exhaustion, scoped closure package, public route export, and mature route package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CofinalModulusSeal}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.CofinalModulusSealUp.CofinalModulusSealUp\_StdBridge`

Bridge status: bridgeChecked

Origin: ai

Not claimed: This chapter does not close Real completeness, quotient stream equality, quotient-stream construction, selected host limits, countable choice, detached windows, or any completion operator outside the finite cofinal route.

Upgrade path: no further upgrade – chapter at top of the formal bridge axis.

Chapter 1199

A Concrete Naming Certificate for $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$

The $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ packet is a composite finite audit surface for the StreamName–RegSeqRat–Real seal route of chapter 9238. It bundles the existing $\mathsf{StreamName}^{\uparrow}$ finite stream interface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the audit-refusal rows that exclude hidden completion routes. Its role is to bind the displayed finite window, dyadic tolerance, RegSeqRat readback, RealUp seal, audit-refusal, hsame, Cont, Pkg, and NameCert rows into one NameCert carrier, not to supply quotient streams, selected host limits, host real equality, countable choice, or ambient completeness.

Falsifiable prediction: A RealUp seal consumer that bypasses the displayed finite-window, dyadic-tolerance, and RegSeqRat readback rows refutes the FiniteWindowRealSealAuditUp packet.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-completion-seal-classifier: no sibling is bijective because only this packet binds finite windows, dyadic tolerance, RegSeqRat readback, Real seal, and audit-refusal rows into one NameCert carrier.

Definition 1199.1 (Finite-window Real-seal audit carrier)**.** *A* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the visible request window;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance and readback row attached to that same window;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row reached from W and D;*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row read only after W, D, R have been displayed;*
5. *F is the audit-refusal row stating that no accepted consumer may replace W, D, R, E by a hidden completion route, quotient stream, selected limit, host real equality, countable-choice row, or ambient-completeness row;*
6. *H records componentwise* hsame *transports for the displayed rows;*
7. *C records the displayed* Cont *replay from finite-window admission through dyadic tolerance, RegSeqRat readback, audit refusal, and the terminal Real seal;*

8. *P is the* Pkg *provenance over the cited sibling packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient streams, selected host limits, host real equality, detached Real seals, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSealAuditUp.FiniteWindowRealSealAuditTasteGate single carrier alignment`

Theorem 1199.2 (Finite-window Real-seal audit NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}^{\uparrow}}$ *surface consists exactly of finite-window exposure through W, dyadic-tolerance exposure through D, RegSeqRat readback through R, terminal Real-seal exposure through E, audit-refusal exposure through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1199.1. Its displayed inventory is the StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal Real seal, audit-refusal row, and the structural support rows.

Read the seal route in dependency order. The row W fixes the finite request window, the row D fixes the dyadic tolerance ledger attached to that window, the row R fixes the regular-rational readback, and the row E is the RealUp seal reached by that displayed route.

The audit row F blocks the absent completion, quotient, selected-limit, host-equality, choice, and ambient-completeness coordinates. The rows H, C, P, N only transport, replay, package, and name the same finite packet, so the listed rows exhaust the local NameCert surface. □

Theorem 1199.3 (Finite-window Real-seal audit non-escape)**.** *Every accepted* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *consumer factors through the finite route*

$$W, D, R, E, F, H, C, P, N.$$

The consumer may inspect the StreamName finite window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, audit-refusal row, componentwise transports, continuations, provenance, and local NameCert row. It receives no quotient stream equality, selected host limit, host real equality, detached Real seal, countable-choice row, hidden completion route, or ambient-completeness coordinate.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSealAuditUp.FiniteWindowRealSealAuditCarrier nonescape`

Proof. Start from Definition 1199.2. The local certificate exposes the complete route inventory for the accepted packet and gives no second source for window, tolerance, readback, seal, refusal, transport, continuation, provenance, or naming data.

Attach the sibling faces. The cited StreamName, DyadicRatCore, RegSeqRat, and Real chapters supply the four route faces W, D, R, E in the order needed by a RealUp seal consumer.

Inspect the refusal row. Since F records the excluded coordinates and the carrier has no other coordinate through which to read them, any consumer that bypasses W, D, R before reaching E has left the accepted audit packet. □

Theorem 1199.4 (Finite-window Real-seal audit four-face minimality)**.** *For every accepted* FiniteWindowRealSealA *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

each of the four source faces W, D, R, E is individually necessary for the accepted route. The row W supplies the $\mathsf{StreamName}^{\uparrow}$ *finite-window face, D supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback face, R supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, and E supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal face.*

Replacing or omitting any one of W, D, R, E leaves no accepted $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *packet with the same route role. The rows F, H, C, P, N can refuse, transport, replay, package, and name only after the four source faces have been displayed; they cannot manufacture a missing StreamName window, dyadic tolerance ledger, RegSeqRat readback, or terminal Real seal.*

Proof. Project the carrier of Definition 1199.1. The four rows W, D, R, E are not auxiliary annotations: they are the source faces from which the displayed route obtains its finite window, dyadic tolerance ledger, regular-rational readback, and terminal Real seal.

Apply Definition 1199.2. The exact row inventory places W, D, R, E in the local certificate before the audit-refusal, transport, continuation, provenance, and naming rows. Thus each of the four faces has a distinct certificate role not repeated by F, H, C, P, N.

Apply Definition 1199.3. An accepted consumer has no bypass coordinate through which a missing window, tolerance ledger, readback, or seal can be recovered. Therefore dropping or replacing any one of W, D, R, E breaks the accepted finite-window Real-seal audit route componentwise. □

Theorem 1199.5 (Finite-window Real-seal audit refusal transport)**.** *Let*

$$A = (W, D, R, E, F, H, C, P, N)$$

and

$$A' = (W', D', R', E', F', H', C', P', N')$$

be accepted $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *carriers. If the rows of A are transported componentwise by* hsame *to the corresponding rows of A', then the audit-refusal row is preserved. Every transported Real-seal consumer must pass through W', D', R' before reading E'.*

The transported packet has no quotient stream, selected host limit, host real equality, countable-choice row, hidden completion route, or ambient-completeness coordinate. These coordinates are still refused by F' and cannot be recovered from H', C', P', N'.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSealAuditUp.FiniteWindowRealSealAuditCarrier refusal transport`

Proof. Start with the NameCert obligations of Definition 1199.2. They identify the complete local inventory of an accepted carrier and make F the row that records the audit refusal.

Apply the componentwise hsame transport recorded by H and H'. Since the transported rows are exactly $W', D', R', E', F', H', C', P', N'$, the refusal row travels with the same finite packet rather than opening a detached completion coordinate.

Use Definition 1199.3 for the target carrier A'. Any transported consumer that reaches the terminal Real seal must read the displayed route through W', D', R' before E'.

Finally apply Definition 1199.4. The four source faces cannot be replaced by F', H', C', P', N', so the refused quotient, selected-limit, host-equality, choice, hidden-completion, and ambient-completeness routes remain absent after transport. □

Theorem 1199.6 (Finite-window Real-seal audit obligation closure)**.** *The carrier definition, the local NameCert obligations, the non-escape theorem, four-face minimality, the refusal-transport theorem, and the displayed TasteGate alignment target together bind the carrier, classifier, stability, exactness, and ledger-refusal obligations for* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$*.*

Concretely, Definition 1199.1 supplies the carrier and classifier rows; Definition 1199.2 supplies the exact local NameCert inventory; Definition 1199.3 supplies ledger refusal for non-displayed Real-seal routes; Definition 1199.4 supplies exactness of the four displayed faces; and Definition 1199.5 supplies stability under componentwise hsame *transport.*

Proof. The carrier and classifier obligations are read from Definition 1199.1. The definition lists the displayed rows and says that accepted comparison has no coordinate outside those rows.

The exactness obligation is supplied by Definition 1199.2 and Definition 1199.4. Together they identify the complete NameCert inventory and show that the four source faces W, D, R, E are all necessary.

The ledger-refusal obligation is supplied by Definition 1199.3. It rules out quotient streams, selected host limits, host real equality, detached Real seals, countable-choice rows, hidden completion routes, and ambient-completeness coordinates for accepted consumers.

The stability obligation is supplied by Definition 1199.5. Componentwise hsame transport carries the refusal row and preserves the forced route through the displayed finite-window, dyadic-tolerance, RegSeqRat, and Real-seal faces. The TasteGate alignment target records that the same single carrier is the Lean-side audit surface for this packet, so the listed rows form the obligation-closure boundary for $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$. □

Theorem 1199.7 (Finite-window Real-seal audit StreamName–Dyadic scope)**.** *For every accepted* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

the visible request window W is scoped by the displayed $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the tolerance ledger D is scoped by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row attached to the same window. No accepted consumer may replace W or D by a hidden stream, quotient stream equality, selected tolerance, host real equality, countable-choice row, or ambient-completeness row.*

Proof. Project the carrier of Definition 1199.1. The first two route faces are W and D: W names the visible StreamName request window, and D records the dyadic tolerance ledger attached to that same window.

Apply the NameCert obligations of Definition 1199.2. They place W and D in the local certificate before the RegSeqRat readback, terminal Real seal, audit refusal, transport, continuation, provenance, and naming rows.

Use non-escape and four-face minimality, Definition 1199.3 and Definition 1199.4. A consumer has no bypass coordinate for a hidden stream or selected tolerance, and neither F, H, C, P, N can manufacture a missing StreamName window or DyadicRatCore ledger.

Finally apply refusal transport, Definition 1199.5. Componentwise hsame transport carries the same W, D scope to any accepted target carrier, so the StreamName–Dyadic entry remains the displayed finite-window scope for the route. □

Theorem 1199.8 (Finite-window Real-seal audit RegSeqRat scope)**.** *For every accepted* FiniteWindowRealSealAudi *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

the row R is a $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose admissibility is scoped by the preceding finite* BHist *packet, componentwise* hsame *transport through H, and displayed* Cont *replay through C. The readback cannot be detached from W, D or replaced by a quotient stream, selected host limit, host real equality, countable-choice row, or ambient-completeness row.*

Proof. Read the carrier in Definition 1199.1. The row R is not a free regular-rational name; it is the RegSeqRat readback reached from the already displayed W, D rows inside the same finite BHist packet.

Apply the NameCert obligations of Definition 1199.2. The certificate inventory records R together with the componentwise hsame row H and the displayed Cont replay row C, so the readback is scoped by the same finite route rather than by an external completion object.

Use Definition 1199.3 and Definition 1199.4. Since accepted consumers must pass through W, D, R, E and each face is necessary, R cannot be recovered from the refusal, continuation, provenance, or naming rows after the fact.

Apply refusal transport, Definition 1199.5. Transported carriers preserve the refusal row and force the target consumer through W', D', R' before E', which keeps the RegSeqRat readback under the transported BHist, hsame, and Cont scope. □

Theorem 1199.9 (Finite-window Real-seal audit Real-seal non-escape scope)**.** *For every accepted* $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$ *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

the terminal row E is scoped as the displayed $\mathsf{Real}^{\uparrow}$ *seal reached only after W, D, R have been exposed. The row F records the non-escape refusal, P records* Pkg *provenance for the sibling packets and kernel rows, and N records the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}^{\uparrow}}$ *name; none of these rows exports a hidden completion route, detached Real seal, selected limit, quotient stream equality, host real equality, countable-choice row, or ambient-completeness coordinate.*

Proof. Project the carrier of Definition 1199.1. The row E is the terminal RealUp seal row, while F, P, N are respectively the audit-refusal row, the package provenance row, and the local NameCert row for the same finite packet.

Apply the NameCert obligations of Definition 1199.2. They identify E, F, P, N as displayed rows of one certificate and place the Real seal after the StreamName, DyadicRatCore, and RegSeqRat faces.

Use non-escape and four-face minimality, Definition 1199.3 and Definition 1199.4. A Real-seal consumer must reach E through W, D, R, and the supporting rows F, P, N can refuse, package, and name only that route.

Finally apply refusal transport, Definition 1199.5. Under componentwise hsame transport the target carrier still exposes only $W', D', R', E', F', H', C', P', N'$, so the Real seal remains non-escaping and the Pkg and $\mathsf{NameCert}$ rows remain local to the displayed route. □

Lean development for this route provides `Nontrivial FiniteWindowRealSealAuditUp` and `FieldFaithful FiniteWindowRealSealAuditUp` in `lean4/BEDC/Derived/FiniteWindowRealSealAuditUp/Tast`

Theorem 1199.10 (Finite-window Real-seal audit scoped dependency)**.** *For every accepted* FiniteWindowRealSealA *carrier*

$$A = (W, D, R, E, F, H, C, P, N),$$

the row W is scoped by $\mathsf{StreamName}^{\uparrow}$*, the row D by* $\mathsf{DyadicRatCore}^{\uparrow}$*, the row R by* $\mathsf{RegSeqRat}^{\uparrow}$*, the row E by* $\mathsf{Real}^{\uparrow}$*, the row F by the finite* BHist *audit refusal, the row H by* hsame*, the row C by* Cont*,*

the row P by Pkg*, and the row N by* NameCert*. Thus each obligation named by the audit packet has a displayed BEDC primitive or sibling-chapter scope before any Real-seal consumer route is admitted.*

Proof. Project the carrier of Definition 1199.1. Its displayed rows already name the four route faces W, D, R, E, the audit row F, and the structural rows H, C, P, N.

Apply the three explicit scope lemmas, Definition 1199.7, Definition 1199.8, and Definition 1199.9. They bind the StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, audit refusal, package provenance, and local naming rows to the displayed finite route.

Use the obligation-closure theorem, Definition 1199.6, to package the same rows through carrier admission, classifier scope, stability, exactness, and ledger refusal. Thus every dependency has an explicit BEDC primitive or sibling-chapter scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}\uparrow}$ **.**

Constructive story: The finite-window Real-seal audit is generated from a $BHist$ packet whose window rows, tolerance rows, audit witness, and Real-seal handoff are all visible before any consumer route is admitted. The carrier convention first names the finite window, then records the audit row that checks the handoff, and finally allows Cont only to continue that audited route. If the representing history changes, $hsame$ transports the same audit packet rather than creating a new window. The resulting NameCert object is a local audit surface: it certifies that Real-seal use is confined to the displayed finite packet and that forbidden coordinates remain absent.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}\uparrow}$, scopedClosed)

Scope closed: This chapter binds the finite-window Real-seal audit carrier, NameCert obligations, non-escape theorem, four-face minimality, refusal transport of Definition 1199.5, obligation closure of Definition 1199.6, explicit scope lemmas of Definition 1199.7, Definition 1199.8, and Definition 1199.9, and scoped dependency theorem of Definition 1199.10.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteWindowRealSealAudit}\uparrow}$, axiomClean)

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSealAuditUp.taste gate`

Lean target: `BEDC.Derived.FiniteWindowRealSealAuditUp.FiniteWindowRealSealAuditTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: It does not claim all Real windows are audited, that finite windows are complete, or that unaudited host limits are admissible.

Upgrade path: The next paper-side step is public export of the scoped audit consumer route; Lean-side axiom-clean status remains independent.

Chapter 1200

A Concrete Naming Certificate for RealModulusPurityBoundary$^{\uparrow}$

The RealModulusPurityBoundary$^{\uparrow}$ packet realizes the axiom-purity boundary of the Real-completeness vision in chapter 9238. It collects the no-choice boundary of chapter 1173, the selected-window refusal of chapter 695, the shared-modulus normal form of chapter 778, the finite-window Real seal audit of chapter 1199, and the fused Real-modulus route of **??**. Its role is to bind dyadic observations, finite StreamName windows, RegSeqRat readbacks, RealUp seals, explicit modulus budgets, transports, continuations, provenance, local naming, and an axiom-refusal row into one finite carrier. It is not a quotient stream, selected host limit, host real equality, countable-choice schedule, ambient completeness principle, or proof by Classical.choice, Quot.sound, or propext.

Falsifiable prediction: Any accepted RealModulusPurityBoundaryUp packet with a selected-host-limit, quotient-stream, or countable-choice coordinate fails the local NameCert carrier predicate.

Independence witness: cauchychoiceboundary, choicefreediagonalselector, regular-cauchy-modulus-normalizer, finite-window-real-seal-audit, realmodulusfusion: none is bijective to this packet because each omits at least one of the explicit axiom-purity refusal row, four-face Real route, modulus budget, or local NameCert non-escape obligation.

Definition 1200.1 (Real modulus-purity boundary carrier)**.** *A* RealModulusPurityBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, R_0, L, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *observation row that records the finite dyadic tolerance read by the Real-completion consumer;*
2. *S is the* StreamName$^{\uparrow}$ *finite-window row naming the visible request window attached to D;*
3. *R_0 is the* RegSeqRat$^{\uparrow}$ *readback row reached from the same dyadic and stream-window data;*
4. *L is the* Real$^{\uparrow}$ *seal row reached only after D, S, R_0 have been displayed;*
5. *B is the explicit modulus-budget and axiom-refusal row, carrying the selected finite modulus threshold and recording refusal of quotient streams, selected host limits, countable choice,* Classical.choice*,* Quot.sound*, and* propext*;*
6. *H records componentwise* hsame *transports for D, S, R_0, L, B and the local packet shape;*

7. *C records the displayed* Cont *replay from dyadic observation and window admission through RegSeqRat readback, modulus budget, refusal, Real seal, and local naming;*
8. *P is the* Pkg *provenance over chapter 1173, chapter 695, chapter 778, chapter 1199,* **??**, BHist, hsame, Cont, Pkg, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusPurityBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for selected host limits, quotient streams, detached Real seals, countable-choice schedules, ambient completeness, or proof-irrelevant host equality.

Lean4 (definition) `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryUp`

Theorem 1200.2 (Real modulus-purity boundary NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusPurityBoundary}^{\uparrow}$ *carrier*

$$R = (D, S, R_0, L, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealModulusPurityBoundary}^{\uparrow}}$ *surface consists exactly of dyadic observation exposure through D, finite-window exposure through S, RegSeqRat readback through R_0, RealUp seal exposure through L, explicit modulus-budget and axiom-refusal exposure through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundary namecert obligations`

Proof. Project the carrier of Definition 1200.1. Its displayed inventory is the dyadic observation, StreamName window, RegSeqRat readback, RealUp seal, modulus-budget refusal row, and the structural support rows.

Read the Real route in dependency order. The dyadic row D fixes the tolerance observation, the window row S fixes the finite request surface, the readback row R_0 fixes the regular-rational stream read from those data, and the seal row L is reached only after the four-face route has been displayed.

The budget row B supplies the explicit modulus threshold and records the refusal of quotient, choice, selected-limit, and proof-extensional coordinates. The rows H, C, P, N only transport, replay, package, and name the same finite packet, so the listed rows exhaust the local NameCert surface. □

Theorem 1200.3 (Real modulus-purity boundary non-escape)**.** *Every accepted* $\mathsf{RealModulusPurityBoundary}^{\uparrow}$ *consumer factors through the finite route*

$$D, S, R_0, L, B, H, C, P, N.$$

The consumer may inspect dyadic observations, StreamName windows, RegSeqRat readbacks, RealUp seals, explicit modulus budgets, axiom-refusal rows, componentwise transports, continuations, provenance, and the local NameCert row. It receives no selected host limit, quotient-stream equality, host real equality, countable-choice coordinate, detached Real-completion object, ambient-completeness row, Classical.choice *row,* Quot.sound *row, or* propext *row.*

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundary nonescape`

Proof. Start from Definition 1200.2. The local certificate exposes the complete route inventory for the accepted packet and gives no second source for dyadic, window, readback, seal, budget, refusal, transport, continuation, provenance, or naming data.

Attach the five sibling boundaries. The Cauchy choice-boundary and choice-free selector chapters exclude hidden selected limits and countable-choice schedules; the regular-Cauchy modulus normalizer supplies the displayed modulus route; the finite-window Real-seal audit and Real-modulus fusion chapters keep the StreamName, RegSeqRat, dyadic, and RealUp faces on the same finite route.

Inspect the refusal row. Since B records the excluded coordinates and the carrier has no other coordinate through which to read them, any consumer that asks for selected host limits, quotient streams, ambient completeness, or proof-extensional host equality has left the accepted boundary packet. □

Theorem 1200.4 (Real modulus-purity boundary constructive prediction)**.** *Let a Real-completion consumer accept a* RealModulusPurityBoundary$^{\uparrow}$ *packet and preserve the displayed route*

$$D, S, R_0, L, B, H, C, P, N.$$

Then every later RealUp or RegSeqRat read is predicted by the explicit modulus budget in B together with the dyadic observation D, finite window S, RegSeqRat readback R_0, and RealUp seal L. A consumer that replaces this prediction by a selected-host-limit, quotient-stream, countable-choice, Classical.choice*,* Quot.sound*, or* propext *coordinate fails the local carrier predicate.*

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundary constructive prediction`

Proof. First read the displayed route. The rows D, S, R_0, L give the four Real-facing observations in order, while B supplies the finite modulus budget that selects the visible part of the route.

Use componentwise transport and replay next. The rows H and C preserve the same dyadic, window, readback, seal, and budget data under accepted consumer motion, so downstream RealUp and RegSeqRat requests return to the displayed finite packet.

Finally apply non-escape. The forbidden selected-limit, quotient, choice, and proof-extensional rows are absent from the carrier and explicitly refused by B. Hence constructive prediction is exactly the finite modulus-budget route, and any attempted replacement fails the carrier predicate. □

Theorem 1200.5 (Real modulus-purity boundary host-choice refusal)**.** *Let a Real-and-Completion consumer accept a* RealModulusPurityBoundary$^{\uparrow}$ *packet*

$$R = (D, S, R_0, L, B, H, C, P, N).$$

If the consumer asks for a selected host limit, quotient-stream equality, countable-choice schedule, host real equality, or ambient-completeness coordinate, then the request is refused by the displayed row B. The only admissible route through the packet remains the finite dyadic observation D, StreamName window S, RegSeqRat readback R_0, RealUp seal L, modulus budget B, componentwise transport H, continuation replay C, package provenance P, local NameCert row N, and the refusal information carried by B.

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundary host choice refusal`

Proof. Project the carrier in Definition 1200.1. Its row B is not merely a modulus budget: it also displays refusal of quotient streams, selected host limits, countable choice, Classical.choice, Quot.sound, and propext.

Apply Definition 1200.2. The local NameCert surface exposes exactly the dyadic observation, StreamName window, RegSeqRat readback, RealUp seal, modulus-budget refusal row, componentwise transport, continuation replay, package provenance, and local naming rows.

Apply Definition 1200.3. Since the accepted packet has no quotient, host-limit, host-equality, countable-choice, or ambient-completeness coordinate, a consumer that asks for one of those coordinates cannot obtain it from any hidden field.

Finally use Definition 1200.4. Downstream RealUp and RegSeqRat reads are predicted by the finite modulus route already displayed by $D, S, R_0, L, B, H, C, P, N$, so the host-choice request is isolated as a refusal obligation of B rather than as an admissible Real-completion operation. □

Theorem 1200.6 (Real modulus-purity boundary scoped obligation package)**.** *The carrier of Definition 1200.1, together with Definition 1200.2, Definition 1200.3, Definition 1200.4, and Definition 1200.5, assembles the full obligation surface over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, explicit modulus budget, axiom-refusal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert*. Every admitted consumer of the packet must expose the same row tuple*

$$D, S, R_0, L, B, H, C, P, N$$

before any Real-facing read is exported.

Proof. Project the carrier as $R = (D, S, R_0, L, B, H, C, P, N)$. The dyadic row D, finite StreamName window S, RegSeqRat readback R_0, Real seal L, and modulus-refusal row B are the analytic coordinates of the packet.

Use the NameCert obligations to expose those coordinates in dependency order. The dyadic and window rows precede the readback, the readback precedes the Real seal, and the explicit budget row records both the finite modulus path and the refused axiom imports.

Apply non-escape and constructive prediction. These theorems force every downstream RealUp or RegSeqRat read to return to the displayed finite route and prevent a selected limit, quotient stream, countable choice, Classical.choice, Quot.sound, or propext coordinate from being repacked as an admitted consumer field.

Finally apply host-choice refusal. The support rows H, C, P, N transport, replay, source, and name only the same displayed route, so no hidden selected limit, quotient stream, countable choice, proof-extensional coordinate, or host-completeness coordinate remains outside the scoped obligation package. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryScopePackage`
↪ variant `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundary scope package`

The Lean file `lean4/BEDC/Derived/RealModulusPurityBoundaryUp/TasteGate.lean` provides explicit `Nontrivial RealModulusPurityBoundaryUp` and `FieldFaithful RealModulusPurityBoundaryUp` instances for the carrier of Definition 1200.1.

Theorem 1200.7 (Real modulus purity boundary tail-budget exhaustion)**.** *Let*

$$R = (D, S, R_0, L, B, H, C, P, N)$$

be an accepted $\mathsf{RealModulusPurityBoundary}^{\uparrow}$ *packet whose displayed modulus windows pass through the L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Then R has no additional tail-budget coordinate outside the displayed dyadic observation D, StreamName window S, RegSeqRat readback R_0, RealUp seal L, modulus-budget refusal row B, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. Every completion-facing read of R factors through these rows plus* hsame, Cont, Pkg, *and* NameCert*; it cannot read ambient Real completeness, quotient streams, host-selected limits, hidden moduli, or proof-extensional tail data.*

Proof. Project the carrier of Definition 1200.1. The only analytic rows are D, S, R_0, L, B, and the only structural rows are H, C, P, N.

Apply the NameCert obligations of Definition 1200.2. They expose the dyadic observation, finite StreamName window, RegSeqRat readback, RealUp seal, and explicit modulus-budget refusal row before any Real-facing consumer can read the packet.

Use the non-escape theorem Definition 1200.3 and the host-choice refusal theorem Definition 1200.5. The window-scope and refusal rows exclude selected host limits, quotient streams, countable-choice schedules, host Real equality, and ambient-completeness coordinates.

Finally read the scoped obligation package of Definition 1200.6. It forces every admitted consumer to expose the same tuple $D, S, R_0, L, B, H, C, P, N$, so no hidden tail-budget coordinate remains outside the L10 four-object route and its structural certificate rows. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryTailBudgetExhaustion`

Theorem 1200.8 (Real modulus-purity boundary dyadic witness stability)**.** *Let*

$$R = (D, S, R_0, L, B, H, C, P, N)$$

be an accepted RealModulusPurityBoundary$^{\uparrow}$ *packet. If a finite dyadic observation row D is replaced inside the same* StreamName$^{\uparrow}$ *window S by an* hsame*-transported* DyadicRatCore$^{\uparrow}$ *witness recorded by H, then the resulting finite packet preserves the* RealModulusPurityBoundary$^{\uparrow}$ *carrier, the local classifier, the tail-budget ledger B, and the refusal of* Classical.choice, Quot.sound, *and* propext*. No new host-limit coordinate, quotient-stream equality, hidden modulus, or ambient Real-completeness row is introduced by the replacement.*

Proof. Project the carrier of Definition 1200.1. The dyadic observation row, StreamName window, RegSeqRat readback, RealUp seal, tail-budget ledger, transport, replay, provenance, and local naming rows are the only available coordinates.

Use the scoped obligation package and tail-budget exhaustion. The accepted packet must expose the same tuple $D, S, R_0, L, B, H, C, P, N$, and no second tail-budget or host-limit coordinate is available outside that tuple.

Transport the dyadic witness by the displayed row H. Because the replacement remains inside the same StreamName window S, the RegSeqRat readback R_0, RealUp seal L, and ledger B are replayed by the same Cont row C rather than by a new completion principle.

Finally inspect the refusal information in B. It continues to reject Classical.choice, Quot.sound, propext, selected host limits, quotient streams, and ambient completeness, so the hsame-transported dyadic witness preserves the carrier and classifier surface. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryDyadicWitnessStability`
Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.taste gate`

Theorem 1200.9 (Real modulus-purity boundary scope binding)**.** *Every accepted* RealModulusPurityBoundary$^{\uparrow}$ *packet binds exactly the dyadic row D, StreamName window S, RegSeqRat readback R_0, Real seal*

L, finite modulus-budget row B, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N as its scoped primitive dependency surface. No additional modulus, completion, quotient-stream, or host-Real row is part of the scoped carrier.*

Proof. Project the carrier of Definition 1200.1. Its displayed tuple is $D, S, R_0, L, B, H, C, P, N$, so each accepted packet first exposes precisely those rows.

Use the NameCert obligations and dyadic witness stability. They keep the dyadic observation, StreamName window, RegSeqRat readback, Real seal, finite modulus budget, transport, replay, provenance, and local naming rows componentwise stable under admitted replacements.

The TasteGate marker identifies the same carrier surface on the Lean side, while the paper scope remains a theory-axis statement. Thus the scoped primitive dependency surface is exactly the displayed tuple and no hidden row is available. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryScopeBinding`

Theorem 1200.10 (Real modulus-purity boundary source nonescape)**.** *No consumer of an accepted* RealModulusPurityBoundary$^{\uparrow}$ *packet may replace the scoped rows $D, S, R_0, L, B, H, C, P, N$ by hidden modulus extraction, quotient streams, selected host limits,* Classical.choice*,* Quot.sound*,* propext*, or ambient* Real$^{\uparrow}$ *completeness. Every Real-facing read remains routed through the displayed dyadic observation, StreamName window, RegSeqRat readback, Real seal, finite modulus-budget, transport, replay, provenance, and local NameCert rows.*

Proof. Start from the scoped binding theorem. It gives the complete list of rows available to any accepted consumer, with D, S, R_0, L, B as the analytic surface and H, C, P, N as the structural surface.

Apply non-escape, host-choice refusal, tail-budget exhaustion, and dyadic witness stability. These results exclude hidden modulus coordinates, quotient streams, selected host limits, proof-extensional rows, and ambient completeness while preserving the displayed dyadic and StreamName route.

Finally replay through C and record through P, N. Since transport and replay only preserve the displayed rows, a consumer cannot introduce Classical.choice, Quot.sound, propext, or a host Real-completeness principle as a replacement source. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundarySourceNonescape`

Theorem 1200.11 (Real modulus-purity boundary RegSeqRat readback induction)**.** *Let*

$$R = (D, S, R_0, L, B, H, C, P, N)$$

be an accepted RealModulusPurityBoundary$^{\uparrow}$ *packet. Any finite induction over the displayed* StreamName$^{\uparrow}$ *windows of S preserves the* RegSeqRat$^{\uparrow}$ *readback row R_0, the terminal* Real$^{\uparrow}$ *seal L, and the axiom-refusal budget B through componentwise* hsame *transport and displayed* Cont *replay. In particular, an inductive consumer that extends, refines, or exhausts the visible windows may not replace R_0, L, B by a quotient stream, selected host limit, hidden modulus, ambient Real-completeness row,* Classical.choice *row,* Quot.sound *row, or* propext *row.*

Proof. The base window is the carrier projection of Definition 1200.1. The displayed tuple already contains the dyadic observation D, StreamName window S, RegSeqRat readback R_0, Real seal L, refusal budget B, transport H, replay C, provenance P, and local naming row N.

For the induction step, apply the NameCert obligations of Definition 1200.2. Each admitted window extension must still expose the dyadic row, finite StreamName row, RegSeqRat readback, RealUp seal, and budget-refusal row before any Real-facing read is exported.

Tail-budget exhaustion, Definition 1200.7, prevents the step from creating a second modulus or hidden tail coordinate. Dyadic witness stability, Definition 1200.8, keeps replacements inside the same StreamName window routed through H and replayed by C.

The Lean-checked TasteGate alignment marker for the carrier identifies the same packet shape used by these paper obligations. Therefore the finite induction can only transport and replay R_0, L, B along the displayed rows, and the refused host-choice and proof-extensional coordinates remain outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryRegSeqRatReadbackInduction`

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusPurityBoundary}\uparrow}$.

Constructive story: The Real modulus-purity boundary is generated by taking the modulus rows already recorded in $BHist$ and making them the only admissible path to a Real-facing consumer. The carrier first reads the dyadic, window, readback, seal, and finite modulus-budget rows, then records the purity boundary that forbids hidden completion or host-real rows, and finally carries a constructive prediction through Cont extensions of that same route. The $hsame$ relation ensures that a different representative of the history cannot alter the modulus path or weaken the boundary. Thus the object is a visible purity test for Real use, grounded in ledger rows rather than ambient real analysis.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusPurityBoundary}\uparrow}, \mathsf{obligation})$

Scope closed: This chapter binds the Real modulus-purity boundary carrier, NameCert obligations, non-escape theorem, constructive-prediction theorem, dyadic witness stability, scoped primitive dependency surface, source nonescape theorem, RegSeqRat readback induction, and the TasteGate marker.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusPurityBoundary}\uparrow}, \mathsf{scaffoldChecked})$

Lean target: `BEDC.Derived.RealModulusPurityBoundaryUp.RealModulusPurityBoundaryScopeBinding`

Bridge status: none

Origin: ai

Not claimed: It does not claim ambient Real completeness, hidden modulus extraction, or equality of Real representatives beyond the recorded modulus boundary.

Upgrade path: Scoped closure requires checked source nonescape, readback induction, and transport replay obligations without bridge checking.

Chapter 1201

A Concrete Naming Certificate for $\mathsf{InterHistTransportSeal}^{\uparrow}$

The $\mathsf{InterHistTransportSeal}^{\uparrow}$ packet realizes a finite transport boundary for the Inter-Hist Locality chain of chapter 9419. It sits after the locality ledger of chapter 759, reads invariant rows from chapter 656, and requires the observer-locality cell surface of chapter 513 before any route can be consumed by a maximum-rate geometry.

Definition 1201.1 (Inter-Hist transport-seal carrier)**.** *An* $\mathsf{InterHistTransportSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, L, I, O, R, H, C, P, N)$$

with the following visible rows:

1. *A and B are the paired source-history rows naming the two histories under comparison;*
2. *L is the inter-Hist locality-ledger row that records the admissible two-history locality surface;*
3. *I is the inter-Hist invariant row that mediates observer-symmetric readback;*
4. *O is the observer-locality cell row that identifies the local observer-facing consumption site;*
5. *R is the cross-Hist causal-route row through which a* $\mathsf{MaxCausalRate}$ *consumer may read the sealed packet;*
6. *H records componentwise* hsame *and* psame *transports for the paired histories, locality ledger, invariant row, observer-locality cell, route, provenance, and local name;*
7. *C records displayed* Cont *replay from the locality ledger to the observer-locality cell, from the invariant row to the route, and from the accepted seal to downstream rate consumers;*
8. *P is the* Pkg *provenance over the locality-ledger, invariant, observer-locality-cell, cross-Hist causal-route,* BHist, hsame, psame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{InterHistTransportSeal}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares exactly $A, B, L, I, O, R, H, C, P, N$. It has no coordinate for a global synchronization row, a host frame, an unsealed cross-Hist route, an invariant detached from observer locality, or a maximum-rate consumer that bypasses the displayed cell and invariant rows.

Lean4 (definition) `BEDC.Derived.InterHistTransportSealUp.InterHistTransportSealUp`

Theorem 1201.2 (Inter-Hist transport-seal NameCert obligations)**.** *For every accepted* $\mathsf{InterHistTransportSeal}^{\uparrow}$ *carrier*

$$T = (A, B, L, I, O, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InterHistTransportSeal}^{\uparrow}}$ *surface consists exactly of paired source-history admission, locality-ledger admission, inter-Hist invariant readback, observer-locality cell admission, cross-Hist causal-route export, componentwise* hsame *and* psame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.InterHistTransportSealUp.InterHistTransportSeal namecert obligations`

Proof. Project the carrier. The positive route is the finite row sequence A, B, L, I, O, R, while H, C, P, N transport, replay, locate, and name those same rows.

The locality-ledger row L supplies the admissible paired-history comparison, the invariant row I supplies observer-symmetric content, and the observer-locality cell row O supplies the local consumption site. The route row R is therefore not a standalone causal edge; it is readable only after the ledger, invariant, and cell rows are present.

The structural rows preserve that surface. Componentwise transport cannot introduce a global frame or detached invariant, continuation replay cannot synthesize a missing observer-locality cell, and package provenance records only the displayed dependencies. Thus the listed obligations exhaust the local NameCert surface. □

Theorem 1201.3 (Inter-Hist transport-seal nonescape)**.** *An accepted* $\mathsf{InterHistTransportSeal}^{\uparrow}$ *packet cannot export a cross-Hist causal route to a* MaxCausalRate *consumer unless the same packet displays both its observer-locality cell row and its inter-Hist invariant row. A route missing either row remains outside the accepted seal, even when paired histories,* hsame *transport,* psame *transport,* Cont *replay,* Pkg *provenance, and a local* NameCert *row are present.*

Proof. Start with the carrier classifier of Definition 1201.1. The route row R is accepted only as one coordinate of the displayed packet $T = (A, B, L, I, O, R, H, C, P, N)$.

Apply the NameCert obligation surface of Definition 1201.2. It makes invariant readback and observer-locality cell admission mandatory before route export.

The remaining rows are structural. They transport and replay accepted coordinates but do not create a replacement for I or O. Hence a packet lacking either row cannot escape into a maximum-rate consumer as a sealed route. □

Lean4 ✓ `BEDC.Derived.InterHistTransportSealUp.InterHistTransportSeal nonescape`
↪ variant `BEDC.Derived.InterHistTransportSealUp.InterHistTransportSealCarrier nonescape`

Theorem 1201.4 (Inter-Hist transport-seal sibling independence)**.** *The* $\mathsf{InterHistTransportSeal}^{\uparrow}$ *carrier is independent of the sibling packets for* InterHistLocalityLedger*,* InterHistInvariant*,* ObserverLocalityCell*,* CrossHistCausalRoute*, and* MaxCausalRate*. It consumes their displayed rows but adds the seal-specific boundary that jointly requires locality-ledger admission, invariant readback, observer-locality-cell admission, route export, transport, replay, provenance, and local naming in one finite packet.*

Proof. Compare the row inventories. The locality ledger supplies the paired-history locality surface, the invariant packet supplies observer-symmetric readback, the observer-locality cell supplies a local consumption site, the causal-route packet supplies route data, and the maximum-rate packet supplies the downstream rate boundary.

The seal requires all of those rows simultaneously and adds the nonescape condition of Definition 1201.3. No sibling chapter carries that joint boundary as a single classifier surface, and no sibling can consume a missing row on behalf of the seal.

Therefore the transport seal is not a restatement of any one sibling. It is the finite packet that makes their rows jointly consumable by downstream rate geometry. □

Theorem 1201.5 (Inter-Hist transport-seal falsifiable prediction)**.** *A transport-seal packet that lacks either the observer-locality cell row or the inter-Hist invariant row cannot be consumed by a* MaxCausalRate *route.*

Proof. This is the consumer-facing form of Definition 1201.3. A MaxCausalRate route reads only the sealed route coordinate R of an accepted packet, and acceptance requires both O and I.

If either row is absent, the candidate has no accepted $\mathsf{InterHistTransportSeal}^{\uparrow}$ classifier surface. The downstream route therefore has no finite sealed packet to consume. □

Falsifiable prediction: A transport-seal packet that lacks either the observer-locality cell row or the inter-Hist invariant row cannot be consumed by a MaxCausalRate route.

Independence witness: interhistlocalityledger, interhistinvariant, observerlocalitycell, crosshistcausalroute, maxcausalrate: locality ledgers lack the sealed route export, invariant packets lack observer-cell admission, observer-locality cells lack the paired invariant route, causal routes lack the joint nonescape boundary, and maximum-rate packets are downstream consumers rather than transport seals.

The Lean TasteGate file for BEDC.Derived.InterHistTransportSealUp provides Nontrivial $\mathsf{InterHistTransportSeal}^{\uparrow}$ and FieldFaithful $\mathsf{InterHistTransportSeal}^{\uparrow}$ instances for the displayed carrier $A, B, L, I, O, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{InterHistTransportSeal}^{\uparrow}}$**.**

Constructive story: The Inter-Hist transport seal is generated from named $BHist$ rows for locality, invariant data, observer-cell admission, and the sealed transport route between histories. The seal begins with the source-facing row, reads the target-facing invariant through the declared transport path, and uses Cont only when the continuation preserves that path. The $hsame$ relation is the mechanism that lets equivalent source or target presentations retain the same seal without appealing to host identity. Non-escape, sibling independence, and the falsifiable prediction all express the same construction: any valid inter-history transport must expose its observer-locality, invariant, and continuation witness in the ledger.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InterHistTransportSeal}^{\uparrow}}$, seed)

Scope closed: This chapter binds the Inter-Hist transport-seal carrier, NameCert obligations, nonescape, sibling-independence, and falsifiable-prediction theorems.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InterHistTransportSeal}^{\uparrow}}$, encodedDef)

Bridge status: none

Origin: ai

Not claimed: It does not claim arbitrary history equivalence, transport without a visible continuation witness, or host-level identity between source and target histories.

Upgrade path: Reaching obligationClosure requires formal source-target transport rows and checked non-escape for transports lacking the carrier witness.

Chapter 1202

A Concrete Naming Certificate for $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$

The $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ packet realizes the Real-completeness separation surface of chapter 9238 as a finite NameCert object. It consumes the dyadic tolerance row of chapter 291, the StreamName and RegSeqRat readback rows of chapter 126 and chapter 301, the RealUp seal of chapter 38, the exact completion boundary of chapter 1120, the budgeted seal route of chapter 913, and the shared threshold route of chapter 560. Its role is to make Real separation readable only through a shared finite window, dyadic tolerance, regular-rational readback, componentwise transport, displayed replay, provenance, and local naming packet, not through quotient streams, host real equality, selected representatives, or an ambient completeness oracle.

Falsifiable prediction: If the shared finite-window row is removed, no accepted $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ packet can expose RealUp separation readback.

Independence witness: finitewindowrealsealaudit, locatedreal, realcompletionexactboundary, budgetedrealsealroute, regularcauchytailmeet: none is bijective because none simultaneously carries the shared finite-window row W, dyadic tolerance row D, StreamName and RegSeqRat readback row S, RealUp separation readback row R, componentwise transport row H, displayed replay row C, provenance row P, and local NameCert row N.

Definition 1202.1 (Finite-window Real-separation carrier)**.** *A* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the shared finite-window row naming the common observation window for the two Real-facing reads;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to exactly that shared window;*
3. *S is the combined* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the stream-name schedule and the regular-rational readback reached from W and D;*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *separation readback row reached only after W, D, S have been displayed;*
5. *H records componentwise* hsame *transport for the shared window, tolerance, readback, separation, and local packet rows;*

6. *C records the displayed* Cont *replay from window admission through dyadic tolerance, StreamName and RegSeqRat readback, RealUp separation readback, and local naming;*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBoundaryU BudgetedRealSealRoute$^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eight rows. It has no coordinate for host real equality, quotient-stream equality, selected representatives, detached Real separation witnesses, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparationCarrier admission`

Theorem 1202.2 (Shared-window refusal for Real separation)**.** *For every accepted* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *carrier*

$$F = (W, D, S, R, H, C, P, N),$$

no consumer can expose the RealUp separation readback row R without first preserving the shared finite-window row W, the dyadic tolerance row D, and the StreamName/RegSeqRat readback row S. Any route that asks for host real equality, quotient-stream equality, a selected representative, or a detached separation witness is outside the accepted carrier.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation window refusal`

Proof. Project the carrier of Definition 1202.1. The only Real-facing coordinate is R, and the carrier lists it after the shared window, tolerance, and readback rows.

Read the route in dependency order. The row W fixes the common finite observation window, D fixes the dyadic tolerance for that window, and S fixes the StreamName and RegSeqRat readback that the Real-facing separation row consumes.

The remaining rows H, C, P, N only transport, replay, package, and name the same displayed packet. Since no quotient stream, host equality, selected representative, or detached separation coordinate appears in the carrier, a consumer that omits W, D, S has no accepted path to R. □

Theorem 1202.3 (Finite-window Real-separation NameCert surface)**.** *For every accepted* FiniteWindowRealSepara *carrier*

$$F = (W, D, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}$ *surface consists exactly of shared finite-window exposure through W, dyadic-tolerance exposure through D, StreamName and RegSeqRat readback through S, RealUp separation readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation namecert surface`

Proof. Start with the carrier inventory. The rows W, D, S, R are the full Real-separation route, while H, C, P, N are the structural rows that preserve and expose that route.

Attach the Real-completion faces. The dyadic row D supplies the finite tolerance, S supplies the StreamName schedule and RegSeqRat readback, and R supplies the RealUp separation readback under the exact-boundary, budgeted-seal, and shared-tail-meet provenance recorded by P.

No further row is available. Thus the local NameCert surface is exhausted by W, D, S, R, H, C, P, N, and any read requiring host equality, quotient streams, selected representatives, or ambient completeness is not a read of this packet. □

Theorem 1202.4 (Finite-window Real-separation obligation surface)**.** *For every accepted* FiniteWindowRealSeparat *carrier*

$$F = (W, D, S, R, H, C, P, N),$$

the RealUp separation obligation surface is exposed only through the common finite window W, the dyadic tolerance row D, the StreamName/RegSeqRat readback row S, the RealUp readback row R, and the structural rows H, C, P, N. No accepted separation consumer can replace these rows by quotient streams, host equality of reals, selected representatives, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation obligation surface`

Proof. Project the carrier of Definition 1202.1. Its only Real-separation coordinates are W, D, S, R, followed by the structural rows H, C, P, N.

Apply the NameCert surface of Definition 1202.3. It lists the same eight rows as the whole local surface, so no further obligation coordinate is available to a consumer.

Now use the shared-window obligation of Definition 1202.5. The row W is common to the two Real-facing reads, D fixes the dyadic tolerance for that window, S supplies the StreamName and RegSeqRat readback, and R is the RealUp readback reached only after those rows are displayed.

The refusal clauses follow from absence of carrier coordinates. Quotient streams, host real equality, selected representatives, and ambient completeness are not fields of F, and the structural rows H, C, P, N only transport, replay, source, and name the same finite surface. □

Theorem 1202.5 (Finite-window Real-separation shared-window obligation)**.** *Every separation read exported by an accepted* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *carrier*

$$F = (W, D, S, R, H, C, P, N)$$

must display one common finite window row W before the dyadic tolerance row D, the $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row S, the* $\mathsf{Real}^{\uparrow}$ *seal row R, componentwise transport H, displayed replay C, provenance P, and the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}$ *row N are consumed. The obligation is constructive: it supplies no quotient stream equality, no host Real equality, no selected representative, and no ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation shared window obligation`

Proof. Project the carrier of Definition 1202.1. The row W is the only coordinate that names the common observation window for the two Real-facing reads.

Apply the shared-window refusal of Definition 1202.2. It blocks access to the RealUp separation readback R unless W, D, S have already been displayed in that order.

Now use the NameCert surface of Definition 1202.3. The structural rows H, C, P, N transport, replay, source, and name the same finite route; they do not add a second window, a quotient equality, a host representative, or an ambient completeness object.

Thus every separation read has to expose the common finite window before any dyadic, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, or local naming row is consumed. □

Theorem 1202.6 (Finite-window Real-separation TasteGate obligation)**.** *The Lean TasteGate surface for* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ FiniteWindowRealSeparation}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ FiniteWindowRealSeparation}^{\uparrow}$ *instances for the same finite carrier $F = (W, D, S, R, H, C, P, N)$. These instances must keep RealUp separation readback behind the displayed dyadic tolerance and StreamName/RegSeqRat rows and must not introduce host equality, quotient streams, selected representatives, or ambient completeness.*

Proof. The obligation is read from Definition 1202.1. Nontriviality must inhabit the displayed finite packet rather than a detached Real value, because the only admissible carrier is $F = (W, D, S, R, H, C, P, N)$.

Field faithfulness must preserve the same route. By Definition 1202.2, the separation row R is not public until W, D, S have been displayed, so any TasteGate classifier faithful to the chapter must transport Real separation only through those finite rows.

The excluded coordinates are absent from the carrier and from the NameCert surface of Definition 1202.3. The Lean file `TasteGate.lean` therefore has to expose the instances for the finite route, not for host equality, quotient streams, selected representatives, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparationTasteGate single carrier alignment`

Theorem 1202.7 (Finite-window Real-separation obligation closure route)**.** *For every accepted* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *carrier*

$$F = (W, D, S, R, H, C, P, N),$$

the full obligation route is exposed through the shared finite-window row W, dyadic tolerance row D, StreamName and RegSeqRat readback row S, RealUp separation readback row R, componentwise transport H, displayed replay C, provenance P, and local naming N before any Real separation read is consumed. No accepted route may read R before the packet has displayed W, D, S, H, C, P, N as the finite support of that read.

Proof. Start with the carrier of Definition 1202.1. It lists exactly the rows W, D, S, R, H, C, P, N, so every accepted Real-separation consumer is already bounded by those displayed coordinates.

Apply the shared-window refusal of Definition 1202.2. It prevents access to the RealUp row R until the common finite window W, dyadic tolerance D, and StreamName/RegSeqRat readback S have been displayed.

Use the NameCert surface and shared-window obligation, Definition 1202.3 and Definition 1202.5. These bind the structural rows H, C, P, N to the same finite packet, so transport, replay, provenance, and local naming cannot escape the displayed route.

Finally apply the TasteGate obligation of Definition 1202.6. It forces any Lean-side gate for the packet to preserve the same finite carrier and refuses host equality, quotient streams, selected representatives, and ambient completeness before the Real separation read. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation obligation closure route`

Theorem 1202.8 (Finite-window Real-separation window-budget determinacy)**.** *Let*

$$F = (W, D, S, R, H, C, P, N) \qquad \text{and} \qquad F' = (W, D, S, R', H, C', P', N')$$

be accepted $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *packets with the same shared finite-window row W, the same dyadic tolerance row D, the same* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row S, and the same componentwise* hsame *transport row H. Then F and F' expose the same admissible* $\mathsf{Real}^{\uparrow}$ *separation readback budget: any consumer of R or R' must pass through the identical finite window W, dyadic budget D, readback row S, and transport row H before it can read a Real-separation handoff. The packet has no second budget coordinate, detached Real witness, quotient stream equality, selected representative, host Real equality, or ambient completeness row from which an alternate readback budget could be obtained.*

Proof. Project the two eight-row carriers from Definition 1202.1. The only rows that can govern a Real-separation budget are the shared finite window W, dyadic tolerance D, StreamName/RegSeqRat readback S, Real readback row, transport H, replay, provenance, and local naming.

Apply the refusal, NameCert surface, and shared-window obligation of Definition 1202.2, Definition 1202.3, and Definition 1202.5. They force every admissible Real-separation read to enter through W, D, S and then preserve the same route through H before the Real-facing row is public.

Finally use the obligation closure route Definition 1202.7. Since the accepted packets agree on W, D, S, H, the admissible budget visible to any consumer is the same finite dependency route. The remaining rows C, P, N and C', P', N' replay, source, and name that route, while the excluded coordinates are not fields of the carrier. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation window budget determinacy`

Theorem 1202.9 (Finite-window Real-separation scoped kernel route)**.** *For every accepted* FiniteWindowRealSepar *carrier*

$$F = (W, D, S, R, H, C, P, N),$$

the scoped kernel route is exactly the finite route over the DyadicRatCore$^{\uparrow}$ *tolerance row,* StreamName$^{\uparrow}$ *window row,* RegSeqRat$^{\uparrow}$ *readback row,* Real$^{\uparrow}$ *separation readback row,* RealCompletionExactBoundaryUp*,* BudgetedRealSealRoute$^{\uparrow}$*,* RegularCauchyTailMeet$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The carrier exposes no route through quotient streams, host real equality, selected representatives, countable choice, detached separation witnesses, or ambient Real completeness.*

Proof. Project the carrier of Definition 1202.1. The rows W, D, S, R, H, C, P, N are the only rows available to a consumer, and the provenance row P names the exact-boundary, budgeted-seal, and shared-tail-meet sources of the Real-facing read.

Apply shared-window refusal and the NameCert surface, Definition 1202.2 and Definition 1202.3. They force the RealUp separation row to be read only after the shared finite window, dyadic tolerance, and StreamName/RegSeqRat readback rows have been displayed, while H, C, P, N preserve and name the same route.

Use the shared-window obligation and obligation closure route, Definition 1202.5 and Definition 1202.7. These bind the structural rows to the displayed kernel dependencies and reject every outside coordinate before the Real-separation read is consumed.

Finally apply window-budget determinacy, Definition 1202.8. Since the visible budget is determined by the same W, D, S, H route, the scoped kernel surface is exactly the listed dyadic, stream, regular-rational, Real, completion-boundary, seal-route, tail-meet, history, transport, continuation, provenance, and local naming inventory. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation scoped kernel route`

Theorem 1202.10 (Finite-window Real-separation witness exhaustion)**.** *Every Real-separation witness accepted by a* FiniteWindowRealSeparation$^{\uparrow}$ *carrier is exhausted by the rows*

$$W, D, S, R, H, C, P, N.$$

No detached Real separation witness can be consumed, and no witness route may replace the shared finite window, dyadic tolerance, StreamName and RegSeqRat readback, Real readback, transport, replay, provenance, and local naming rows by quotient streams, host equality, selected representatives, or ambient completeness.

Proof. Project the carrier of Definition 1202.1. The witness-facing coordinates are exactly W, D, S, R, H, C, P, N.

Use shared-window refusal, the NameCert surface, and the shared-window obligation from Definition 1202.2, Definition 1202.3, and Definition 1202.5. They force every Real-separation read to display the common finite window, tolerance, and readback rows before the Real-facing row is consumed.

Finally apply the obligation-closure route of Definition 1202.7. The structural rows preserve the same finite packet, and the carrier contains no quotient-stream, host-equality, selected-representative, or ambient-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation witness exhaustion`

Theorem 1202.11 (Finite-window Real-separation scoped completion route)**.** *Every accepted* $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *separation read that reaches a completion-facing consumer factors through the finite route*

$$W \longmapsto D \longmapsto S \longmapsto R \longmapsto \mathsf{RealCompletionExactBoundaryUp} \longmapsto \mathsf{BudgetedRealSealRoute}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailM}$$

The route first displays the common finite window W, dyadic tolerance D, $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback S, and* $\mathsf{Real}^{\uparrow}$ *seal R; it then reads exact-boundary, budgeted-seal, and tail-meet rows only through the provenance already carried by P. It exposes no quotient stream, host real equality, selected representative, detached completion witness, or ambient completeness principle.*

Proof. Project the carrier of Definition 1202.1. The only separation-facing rows are W, D, S, R, H, C, P, N, and the provenance row names the exact-boundary, budgeted-seal, and shared-tail-meet siblings consumed by the Real-facing route.

Apply the shared-window refusal and NameCert surface theorems, Definition 1202.2 and Definition 1202.3. They force any RealUp read to pass through the same finite window, dyadic tolerance, and StreamName/RegSeqRat readback before R becomes public.

Use the shared-window obligation and obligation-closure route, Definition 1202.5 and Definition 1202.7. These bind H, C, P, N to the same displayed packet, so completion-facing consumers cannot detach the exact-boundary or seal rows from the carrier.

Finally apply the scoped kernel route and witness exhaustion theorems, Definition 1202.9 and Definition 1202.10. The scoped route already lists the dyadic, stream, regular-rational, Real, exact-boundary, budgeted-seal, and tail-meet dependencies, while witness exhaustion blocks every outside completion witness. □

Theorem 1202.12 (Finite-window Real-separation shared-budget normalization)**.** *Let*

$$F = (W, D, S, R, H, C, P, N) \qquad \textit{and} \qquad F' = (W, D, S, R', H, C', P', N')$$

be accepted $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *packets with the same shared finite-window row W and the same dyadic tolerance row D. Any* $\mathsf{Real}^{\uparrow}$ *separation read exposed by either packet factors through one normalized finite budget route*

$$W, D, S, R, H, C, P, N$$

after replacing the replay, provenance, and local naming rows of the second packet by the displayed route for the same window, tolerance, readback, and transport budget. No normalized route may use quotient streams, selected representatives, or ambient completeness.

Proof. Start with the shared-window refusal of Definition 1202.2. It blocks any RealUp read until the common window row W, dyadic tolerance row D, and StreamName/RegSeqRat readback row S have been displayed.

Apply the NameCert surface and shared-window obligation of Definition 1202.3 and Definition 1202.5. These theorems bind the readback row, transport row, replay, provenance, and local name to the same finite packet once W and D are fixed.

Use the window-budget determinacy theorem Definition 1202.8. Agreement on the shared finite window and dyadic tolerance forces the visible Real-separation budget through the same finite route, so the normalized read can be recorded as W, D, S, R, H, C, P, N.

The refusal clauses are structural. Quotient streams, selected representatives, and ambient completeness are not fields of the accepted carrier, and replay or provenance cannot create a second budget outside the displayed finite route. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation shared budget normalization`

Theorem 1202.13 (Finite-window Real-separation common-window exactness)**.** *Let*

$$F = (W, D, S, R, H, C, P, N) \qquad \textit{and} \qquad F' = (W, D, S, R', H', C', P', N')$$

be accepted $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ *packets with the same shared finite-window row W, the same dyadic tolerance row D, and the same* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row S. Every admissible* $\mathsf{Real}^{\uparrow}$ *separation read exposed by either packet factors through the same W, D, S route before reading R or R'. Disagreement in replay, provenance, local naming, or the remaining transport rows cannot create a second separation budget outside that common finite window.*

Proof. Start with the shared-window refusal of Definition 1202.2. It blocks every RealUp separation read until the common finite window W, dyadic tolerance D, and StreamName/RegSeqRat readback row S have been displayed.

Use the NameCert surface and shared-window obligation of Definition 1202.3 and Definition 1202.5. These bind the Real-facing row, transport, replay, provenance, and local name to the finite route whose first three rows are W, D, S.

Apply window-budget determinacy and shared-budget normalization, Definition 1202.8 and Definition 1202.12. Once the two packets agree on W, D, S, any admissible read of R or R' has the same finite budget prefix, while H, C, P, N and H', C', P', N' only preserve, replay, source, and name that displayed route.

The refusal clause follows from the carrier inventory. The packet has no coordinate for a detached budget, quotient stream equality, selected representative, host Real equality, or ambient completeness, so disagreement in replay, provenance, or local naming cannot supply another route to a RealUp separation read. □

Lean4 ✓ `BEDC.Derived.FiniteWindowRealSeparationUp.FiniteWindowRealSeparation common window exactness`

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowRealSeparation}^{\uparrow}$ packet is generated from a shared finite-window row, a DyadicRatCore tolerance row, StreamName and RegSeqRat readback, RealUp separation readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1202.1, Definition 1202.4, Definition 1202.5, Definition 1202.6, Definition 1202.10, and Definition 1202.12 bind the finite Real-separation carrier, obligation surface, shared-window obligation, TasteGate obligation, witness exhaustion, and shared-budget normalization rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWindowRealSeparation}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not close Lean verification, bridge checking, quotient reals, host real equality, selected representatives, or ambient Real completeness.
Upgrade path: Obligation closure requires checked carrier admission and TasteGate-visible Nontrivial and FieldFaithful instances for the same finite Real-separation packet.

Chapter 1203

A Concrete Naming Certificate for $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$

The $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$ packet concretizes the borrowed-recursor boundary of chapter 9396 as a finite NameCert carrier. It sits beside the strict gate packet of chapter 526, the socket refusal surface of chapter 536, and the failed-gate witness of chapter 901. The packet allows host Lean recursors to serve as checking infrastructure only after proof ancestry, gate verdict, socket refusal, and local naming rows are displayed.

Definition 1203.1 (Borrowed-recursor boundary carrier)**.** *A* $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, S, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the borrowed host-recursor use, recorded as a checking route rather than theorem content;*
2. *A is the proof-ancestry audit row classifying whether the route depends on forbidden host commitments;*
3. *S is the visible* $\mathsf{ApophaticSocket}^{\uparrow}$ *row that names the external host supply position;*
4. *F is the* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *row used when the ancestry row refuses the route;*
5. *H records componentwise* hsame *transport for recursor use, ancestry, socket, failure witness, provenance, and local naming rows;*
6. *C records displayed* Cont *replay from the recursor use through ancestry audit, socket naming, refusal, and local consumption;*
7. *P records* Pkg *provenance over auditgateboundary, apophaticsocket, apophaticname, certificateauditgate, auditgatefailurewitness,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BorrowedRecursorBoundary}^{\uparrow}}$ *row.*

The carrier exports a boundary certificate for checked host-recursion use. It exports no host recursion principle, global kernel completeness theorem, hidden eliminator, or proof dependency absent from the ancestry row.

Lean4 (definition) `BEDC.Derived.BorrowedRecursorBoundaryUp.taste gate`
Lean4 (definition) `BEDC.Derived.BorrowedRecursorBoundaryUp.BorrowedRecursorBoundaryUp`

Theorem 1203.2 (Borrowed-recursor boundary nonescape)**.** *Every accepted* $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$ *packet exposes the proof-ancestry row A before any consumer may use the borrowed recursor row R. If A records a forbidden dependency or lacks the displayed ancestry report, the route must pass through the socket row S and the failure witness row F as an apophatic refusal rather than as theorem content.*

Proof. Unfold Definition 1203.1. The only accepted recursor use is the displayed row R, and the next row in the packet is the proof-ancestry audit A.

The socket row S names the host supply position without internalizing it. If the ancestry row is absent or refuses the route, the failure witness F is the only displayed consumer-facing continuation.

The transport, replay, provenance, and local naming rows H, C, P, N can repack only these visible rows. Thus a consumer receives either an ancestry-cleared checking route or a visible refusal path, never an uninspected host-recursion principle. □

Theorem 1203.3 (Borrowed-recursor gate correspondence)**.** *For an accepted* $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$ *packet, the strict audit gate may certify host-recursion nonescape exactly through the displayed tuple R, A, S, F, H, C, P, N. The gate consumes the proof-ancestry row A, the socket row S, the failure-witness row F, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it treats the borrowed recursor as a BEDC checking route.*

Proof. Start with the carrier of Definition 1203.1. Its rows separate host-recursion use R from proof ancestry A, socket naming S, and refusal witness F.

Use the gate sibling of chapter 526. The strict gate can inspect only the finite audit-facing rows it is given, so it reads ancestry and refusal through A, S, F rather than through a hidden host predicate.

The remaining rows H, C, P, N supply transport, replay, provenance, and local naming. Together they are exactly the finite route through which the borrowed recursor is admitted as checking infrastructure, with no extra host-recursion content exported. □

Falsifiable prediction: A BorrowedRecursorBoundaryUp packet without the proof-ancestry row cannot certify host-recursion nonescape.

Independence witness: auditgateboundary, apophaticsocket, apophaticname, certificateauditgate, auditgatefailurewitness: the borrowed-recursor boundary combines proof ancestry, visible host-socket naming, failed-gate refusal, transport, replay, provenance, and local naming; none of the named siblings carries all rows R, A, S, F, H, C, P, N together.

The Lean follow-up lives at `lean4/BEDC/Derived/BorrowedRecursorBoundaryUp/TasteGate.lean` and must provide `Nontrivial BorrowedRecursorBoundaryUp` and `FieldFaithful BorrowedRecursorBoundaryU` instances for the same displayed carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BorrowedRecursorBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BorrowedRecursorBoundary}^{\uparrow}$ packet is generated from a borrowed recursor-use row, proof-ancestry audit row, socket row, failure-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorrowedRecursorBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1203.1, Definition 1203.2, and Definition 1203.3 bind borrowed host-recursion use to displayed proof ancestry, audit refusal, socket naming, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorrowedRecursorBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, a public standard bridge, host kernel completeness, a global recursion theorem, or theorem content for refused host dependencies.

Upgrade path: Obligation closure requires checked carrier admission, proof-ancestry exactness, strict-gate correspondence, socket refusal, componentwise transport, displayed continuation, provenance, and local naming rows.

Chapter 1204

A Concrete Naming Certificate for $\mathsf{TailCompatibleCauchySection}^{\uparrow}$

The $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ packet realizes the finite-window route behind the regular-Cauchy diagonal-limit construction of chapter 9238. It is structurally composite over the stream windows of chapter 126, the dyadic ledgers of chapter 291, the regular-rational readbacks of chapter 301, the real-name seal of chapter 38, and the finite observation-budget selector of chapter 782. The packet names a compatible finite section, not a host limit, quotient stream, selected infinite subsequence, or ambient completeness principle.

Because the carrier is composite over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, its falsifiability and independence witnesses are inherited from those component packets. The atomic falsifiability and independence rows used for primitive carriers are therefore not used here.

Definition 1204.1 (Tail-compatible Cauchy section carrier)**.** *A* $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ *carrier is a finite packet*

$$S = (X, \mu, n, k, i, W, B, R, D, E, H, C, P, N)$$

with the following rows:

1. *X is the displayed regular-Cauchy family from section 9238.1;*
2. *μ is the modulus row selecting tail indices for X;*
3. *n is the visible precision request;*
4. *$k = \mu(1/(n+1))$ and $i = k + n + 1$ are the displayed diagonal-section indices;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window exposing the i-th read of the k-th regular-rational member;*
6. *B is the* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *budget row certifying that the same window is admissible for the section read;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected regular-rational member along W;*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger attached to the selected readback;*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by repacking the selected regular-rational read through the displayed finite route;*
10. *H records componentwise* hsame *transport for $X, \mu, n, k, i, W, B, R, D, E$ and the local certificate row;*

11. *C records* Cont *replay from the precision request through the budgeted window, readback, dyadic ledger, and real seal;*
12. *P is the* Pkg *provenance over* TailCompatibleCauchySection$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, FiniteObservationBudgetSelector$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* NameCert$_{\text{TailCompatibleCauchySection}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite section rows and their transport, replay, provenance, and naming support.

Definition 1204.2 (Tail-compatible Cauchy section classifier)**.** *Two accepted* TailCompatibleCauchySection$^{\uparrow}$ *carriers classify together exactly when their regular-Cauchy family rows and modulus rows select the same displayed diagonal-section route, their precision requests yield the same finite index pair k, i, and their* StreamName$^{\uparrow}$, FiniteObservationBudgetSelector$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *transport, continuation, package, and local naming rows are related componentwise by the recorded* hsame *comparisons and* Cont *routes.*

Theorem 1204.3 (Tail-compatible Cauchy section NameCert obligations)**.** *For the carrier of Definition 1204.1, the local* NameCert$_{\text{TailCompatibleCauchySection}^{\uparrow}}$ *surface consists exactly of carrier admission, the finite diagonal-section computations $k = \mu(1/(n+1))$ and $i = k+n+1$,* StreamName$^{\uparrow}$ *window exposure,* FiniteObservationBudgetSelector$^{\uparrow}$ *budget admission,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* Real$^{\uparrow}$ *seal exposure, componentwise transport, continuation replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.TailCompatibleCauchySectionUp.TailCompatibleCauchySectionTasteGate single carrier alignment`

Proof. Project the packet S. Its coordinates list the regular-Cauchy family, modulus, precision request, computed section indices, stream window, budget selector, regular-rational readback, dyadic ledger, real seal, transport, replay, provenance, and local naming rows.

The finite index computation is internal to the packet. The modulus row supplies k, the request supplies n, and the displayed arithmetic supplies $i = k+n+1$, matching the diagonal representative of Definition 9238.3.

The remaining rows are precisely the component certificate surfaces. The stream window, budget selector, regular-rational readback, dyadic ledger, and real seal are consumed in that order, while H, C, P, N only transport, replay, source, and name those rows. No further NameCert obligation remains. □

Theorem 1204.4 (Tail-compatible Cauchy section index determinacy)**.** *For an accepted* TailCompatibleCauchySect *carrier*

$$S = (X, \mu, n, k, i, W, B, R, D, E, H, C, P, N),$$

the displayed regular-Cauchy family row X, modulus row μ, and precision request n determine the section route indices. The only admitted route is

$$k = \mu(1/(n+1)) \qquad \textit{and} \qquad i = k + n + 1,$$

followed by the ordered read through W, B, R, D, E.

Proof. Project the carrier of Definition 1204.1. Its fourth coordinate is the displayed computation of k from the modulus row and the visible precision request, and its fifth coordinate is the displayed computation of i from k and n.

The classifier of Definition 1204.2 admits carriers together only when those same rows select the same diagonal-section route. The obligation theorem Definition 1204.3 then fixes the subsequent order of the stream window, budget row, regular-rational readback, dyadic ledger, and RealUp seal. □

Theorem 1204.5 (Tail-compatible Cauchy section classifier transport)**.** *Let* $S = (X, \mu, n, k, i, W, B, R, D, E, H, C,$ *and* $S' = (X', \mu', n', k', i', W', B', R', D', E', H', C', P', N')$ *be accepted* $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ *carriers. If the recorded componentwise* hsame *row relates*

$$X, \mu, n, k, i, W, B, R, D, E$$

to

$$X', \mu', n', k', i', W', B', R', D', E',$$

then the carrier admission and classifier verdict transport from S *to* S' *along the same finite section route.*

Proof. Use the transport coordinate H in Definition 1204.1. It records the componentwise hsame comparisons for the regular-Cauchy row, modulus, precision request, index pair, stream window, budget row, readback, dyadic ledger, RealUp seal, and local certificate row.

Apply the classifier definition. Its equality test is not a detached predicate: it is exactly the conjunction of same route selection and componentwise transport through those rows. The NameCert-obligation theorem supplies the carrier admission surface for the transported packet, so the classifier verdict follows with the same ordered route. □

Theorem 1204.6 (Tail-compatible Cauchy section continuation replay)**.** *For every accepted* $\mathsf{TailCompatibleCauchyS}$ *packet, any* Cont *replay recorded by the carrier preserves the ordered route*

$$W, B, R, D, E.$$

In particular, the $\mathsf{Real}^{\uparrow}$ *seal coordinate* E *cannot be exposed before the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *budget row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been displayed on the same packet.*

Proof. Project the continuation coordinate C from the carrier. By Definition 1204.1, it replays the precision request through the budgeted stream window, regular-rational readback, dyadic ledger, and RealUp seal in that order.

The NameCert-obligation theorem records the same order as part of the local certificate surface. The seal-boundary theorem Definition 1204.8 identifies the RealUp-facing boundary as the end of this finite route, so a continuation replay cannot make E available before the component rows W, B, R, D have already appeared. □

Theorem 1204.7 (Tail-compatible Cauchy section component scope binding)**.** *Every accepted* $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ *packet factors its displayed section route through exactly the component scopes*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{FiniteObservationBudgetSelector}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The only support rows available to that factorization are the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{TailCompatibleCauchySection}^{\uparrow}}$ *rows. No further carrier, classifier, host-real equality, quotient-stream equality, selector, or ambient completeness scope is exported by the packet.*

Proof. Project the carrier tuple of Definition 1204.1. Its displayed coordinates first determine the regular-Cauchy family, modulus, precision request, and diagonal indices, and then expose the component rows W, B, R, D, E in the order fixed by Definition 1204.3 and Definition 1204.4.

Read those rows as scope witnesses. The stream window is the $\mathsf{StreamName}^{\uparrow}$ scope, the budget row is the $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ scope, the readback row is the $\mathsf{RegSeqRat}^{\uparrow}$ scope, the tolerance ledger is the $\mathsf{DyadicRatCore}^{\uparrow}$ scope, and the seal row is the $\mathsf{Real}^{\uparrow}$ scope. These are the only component rows listed by the carrier surface.

Use Definition 1204.5 to transport the classifier along the recorded hsame support, and use Definition 1204.6 to preserve the same ordered route under Cont replay. The remaining Pkg and local $\mathsf{NameCert}$ rows only source and name that finite factorization.

Finally, Definition 1204.8 closes the route at the $\mathsf{Real}^{\uparrow}$ seal. Since the boundary theorem excludes host reals, quotient-stream equality, chosen limits, detached selectors, countable-choice rows, and ambient completeness witnesses, no component scope outside the displayed five can be read from an accepted packet. □

Theorem 1204.8 (Tail-compatible Cauchy section RealUp seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer read of an accepted* $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ *packet factors through the finite route*

$$X, \mu, n, k, i, W, B, R, D, E, H, C, P, N.$$

The seal coordinate E is reachable only after the shared tail-compatible $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *budget row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been displayed on the same carrier. The packet exports no host real, quotient stream equality, chosen limit, detached selector, countable-choice row, or ambient completeness witness.*

Proof. Start with Definition 1204.3. It forces every accepted packet to expose the modulus-selected index pair before any readback or seal-facing row is available.

Next read the component surfaces in order. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite tail-compatible window, the $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ row admits that window as a finite budgeted request, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the selected regular-rational readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the tolerance ledger attached to that readback.

Only then can the $\mathsf{Real}^{\uparrow}$ seal row be consumed, and it is preserved solely through the displayed transport, replay, provenance, and local naming rows. Since none of the carrier coordinates is a host real, quotient stream equality, chosen limit, detached selector, countable-choice row, or ambient completeness witness, no Real-facing consumer can obtain such data from the section packet. □

Theorem 1204.9 (Tail-compatible Cauchy section scoped component readiness)**.** *Every accepted* $\mathsf{TailCompatibleCauchySection}^{\uparrow}$ *section route is scoped by the displayed* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *budget admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{Real}^{\uparrow}$ *seal, together with the supporting* BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{TailCompatibleCauchySect}}$ *rows. These rows are the whole component scope exported by the packet: they do not expose a host real, quotient stream equality, detached selector, countable-choice row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.BHistArity14TupleNameCertUp.BHistArity14TupleNameCertCarrierAdmission`

Proof. Start with the NameCert obligations of Definition 1204.3. They list the carrier admission surface, the two displayed diagonal-section computations, and the ordered component rows from stream window through RealUp seal.

The index-determinacy theorem Definition 1204.4 fixes the route selected by the regular-Cauchy family, modulus, and precision request. The component-scope binding theorem Definition 1204.7 then identifies the only component scopes available along that route as $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Use classifier transport and continuation replay from Definition 1204.5 and Definition 1204.6 to keep the same rows under hsame transport and Cont replay. The Pkg and local NameCert rows source and name the same finite factorization rather than adding a further carrier scope.

Finally apply the RealUp seal boundary theorem Definition 1204.8. It makes the $\mathsf{Real}^{\uparrow}$ seal the terminal read of the finite route and excludes host reals, quotient-stream equality, detached selectors, countable-choice rows, and ambient completeness witnesses from the exported packet surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailCompatibleCauchySection}^{\uparrow}}$.

Constructive story: The tail-compatible Cauchy section is generated from a $BHist$ packet that records a Cauchy section, its modulus-selected index pair, finite tail witnesses, and the classifier that checks compatibility with the RealUp seal boundary. The carrier reads the section only through those tail rows; $hsame$ transports the same finite witness data when a history representative changes. Continuation data may extend the section route, but Cont must preserve the tail-compatible classifier already recorded. The NameCert surface is therefore a composite certificate: it connects a Cauchy section to StreamName, finite observation budget, RegSeqRat, DyadicRatCore, and RealUp through visible tail evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TailCompatibleCauchySection}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter binds the tail-compatible Cauchy-section carrier, classifier, index-determinacy row, classifier-transport row, continuation-replay row, NameCert obligations, component scope binding, RealUp seal boundary, and scoped component readiness for the displayed section route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TailCompatibleCauchySection}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.TailCompatibleCauchySectionUp.taste gate`

Lean target: `BEDC.Derived.TailCompatibleCauchySectionUp.TailCompatibleCauchySectionTasteGate\_single\_carrier\_al`

Bridge status: none

Origin: ai

Not claimed: It does not claim arbitrary Cauchy section choice, quotient-stream equality, host real equality, or Real equality outside the finite tail-compatible witness route.

Upgrade path: Public closure requires an exported consumer surface or bridge evidence for the same StreamName, finite-budget, RegSeqRat, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

$\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ reads its finite root route through chapter 126, chapter 301, chapter 291, and chapter 38, with the root-index child below making that route visible at the hub surface.

Chapter 1205

A Concrete Naming Certificate for $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$

The $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ packet realizes the diagonal representative of section 9238.1 as a finite naming certificate. It sits after the diagonal-modulus surface of chapter 390, and it consumes the finite windows, readbacks, dyadic ledgers, and terminal real-name seal supplied by chapter 126, chapter 301, chapter 291, and chapter 38. The same terminal real-facing handoff is the Cauchy-limit seal boundary of chapter 386; the diagonal representative consumes it only as a displayed finite route, not as quotient-stream equality, a selected host limit, choice, or ambient Real completeness. The packet names the representative row itself; it is not a Real-completeness principle and it does not choose a host limit.

Definition 1205.1 (Diagonal-limit representative carrier)**.** *A* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

with the following displayed rows:

1. *X is the regular-Cauchy family over* $\mathsf{RegSeqRat}^{\uparrow}$ *whose diagonal representative is being inspected;*
2. *μ is the explicit modulus row read through chapter 390;*
3. *n is the displayed precision request;*
4. *$k = \mu(1/(n+1))$ is the selected family index;*
5. *$i = k + n + 1$ is the selected readback index;*
6. *w is the* $\mathsf{StreamName}^{\uparrow}$ *window exposing the i-th read of the k-th family member;*
7. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for that selected member and window;*
8. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the tolerance and rational arithmetic of the readback;*
9. *e is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the selected regular-rational readback has been repacked;*
10. *H records componentwise* hsame *transport for $X, \mu, n, k, i, w, r, d, e$ and the local certificate row;*
11. *C records displayed* Cont *replay from the modulus request through the window, readback, dyadic ledger, and real seal;*

12. *P is the* Pkg *provenance over* DiagonalLimitRepresentative$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* NameCert$_{\text{DiagonalLimitRepresentative}^\uparrow}$ *row.*

The carrier exports no host real, quotient stream equality, hidden tail selector, countable choice row, or ambient Real$^\uparrow$ *completeness field.*

Definition 1205.2 (Diagonal-limit representative classifier)**.** *Two accepted* DiagonalLimitRepresentative$^\uparrow$ *carriers are classified together exactly when their modulus rows select the same displayed diagonal route and their selected family index, readback index,* StreamName$^\uparrow$ *window,* RegSeqRat$^\uparrow$ *readback,* DyadicRatCore$^\uparrow$ *ledger,* Real$^\uparrow$ *seal row, transport rows, continuation rows, package rows, and local naming rows are related componentwise by the recorded* hsame *comparisons and* Cont *routes.*

Theorem 1205.3 (Diagonal-limit representative NameCert obligations)**.** *For the carrier of Definition 1205.1, the local* NameCert$_{\text{DiagonalLimitRepresentative}^\uparrow}$ *surface consists exactly of carrier admission, diagonal-index computation $k = \mu(1/(n+1))$ and $i = k + n + 1$, finite-window exposure through w, regular-rational readback through r, dyadic ledger exposure through d, terminal seal exposure through e, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the packet D. The carrier lists the family row, modulus row, request row, computed index rows, selected window, regular-rational readback, dyadic ledger, real seal, transport, continuation, provenance, and local naming row.

The diagonal-index computation is finite. The modulus surface supplies k, the unary request supplies n, and the readback index i is obtained by the displayed rational-index arithmetic used in Definition 9238.3.

The remaining rows expose the route through the four concrete sibling surfaces: StreamName$^\uparrow$ for w, RegSeqRat$^\uparrow$ for r, DyadicRatCore$^\uparrow$ for d, and Real$^\uparrow$ for e. Since H, C, P, N only transport, replay, source, and name those displayed rows, the listed obligations exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeNameCertObligations`

Theorem 1205.4 (Diagonal-limit representative route predicate obligation)**.** *For an accepted* DiagonalLimitRepresentative$^\uparrow$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the accepted route predicates are exactly the displayed rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$. The family and readback rows are read through RegSeqRat$^\uparrow$*, the selected window through* StreamName$^\uparrow$*, the dyadic ledger through* DyadicRatCore$^\uparrow$*, and the terminal seal through* Real$^\uparrow$*. The support predicates are precisely componentwise* hsame *transport through H,* Cont *replay through C,* Pkg *provenance through P, and the local* NameCert$_{\text{DiagonalLimitRepresentative}^\uparrow}$ *row N.*

Proof. Project the carrier of Definition 1205.1. Its rows already name the family, modulus, request, selected indices, selected window, readback, dyadic ledger, Real seal, and the four support coordinates.

Apply the NameCert obligations of Definition 1205.3. They certify the two index computations and expose the sibling rows in the order StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, and Real$^\uparrow$.

It remains to exclude additional route predicates. The classifier of Definition 1205.2 allows transport only by componentwise hsame and replay only by the recorded Cont routes, while P and

N source and name the same packet. Hence the accepted route predicate surface is exactly the displayed carrier row. □

Theorem 1205.5 (Diagonal-limit representative terminal stability obligation)**.** *Let*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

and

$$D' = (X', \mu', n', k', i', w', r', d', e', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers classified together by their recorded componentwise* hsame *rows and replayed* Cont *routes. Then the selected diagonal indices, selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal remain on the displayed route*

$$\mu, n, k, i, w, r, d, e.$$

The transport does not introduce a hidden selector, detached window, alternate readback, second dyadic ledger, or off-packet terminal Real seal.

Proof. First read index stability. By Definition 1205.12, the displayed equations for k, i and k', i' are part of the accepted route and remain tied to the modulus request under classifier transport.

Next read the terminal route. The window-ledger exhaustion theorem, Definition 1205.13, confines public reads of w, r, d, e to the displayed sibling rows, and Definition 1205.10 places every Real-facing read downstream of those rows.

Finally use the support rows. Componentwise hsame and replayed Cont preserve the rows already present in the packet; Pkg and NameCert name that same packet. Therefore terminal stability is stability of the displayed route, not permission to manufacture a hidden selector or an off-packet terminal seal. □

Theorem 1205.6 (Diagonal-limit representative obligation-closure package)**.** *The chapter's* DiagonalLimitRepresen *obligation surface is the package consisting of the carrier of Definition 1205.1, the classifier of Definition 1205.2, the NameCert obligations of Definition 1205.3, the route predicate obligation of Definition 1205.4, and the terminal stability obligation of Definition 1205.5. This package is the concrete diagonal representative surface available to later Lean formalization; it contains no quotient stream equality, host limit, selected completed limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The carrier and classifier supply the accepted packet and its componentwise comparison rule. The NameCert obligations certify the displayed diagonal-index computations, sibling rows, and support rows.

The route predicate obligation identifies the exact predicate surface consumed by downstream readers. The terminal stability obligation then proves that componentwise transport and continuation replay preserve the selected diagonal route through the terminal Real seal.

Combining these pieces yields the chapter's obligation surface. The exclusions are inherited from the carrier and from the already established nonchoice and completion boundaries, so the package closes the obligation level without adding quotient, host-limit, choice, or ambient-completeness data. □

Theorem 1205.7 (Diagonal-limit representative primitive-scope exhaustion)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the displayed route predicates are scoped exactly by BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and the sibling rows supplied by* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*, and* FinitePrefixLimitBudget$^{\uparrow}$*. The representative route uses $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ together with the shared-terminal finite-prefix rows of Definition 1205.32; it has no primitive outside those listed BEDC rows.*

Proof. Project the carrier. The only local rows are the family, modulus, request, selected indices, window, readback, ledger, seal, transport, continuation, provenance, and naming rows listed in Definition 1205.1.

Read the sibling surfaces. The route predicate obligation places w, r, d, e respectively in StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$, and the shared-terminal carrier adds only the finite-prefix budget rows from FinitePrefixLimitBudget$^{\uparrow}$.

The support rows are exactly H, C, P, N. Thus every transport, replay, source, and local name is accounted for by hsame, Cont, Pkg, and NameCert over BHist. Since the carrier and the shared-terminal extension export no further coordinates, the primitive scope is exhausted by the listed rows. □

Theorem 1205.8 (Diagonal-limit representative support-scope nonescape)**.** *All accepted transport and replay for the* DiagonalLimitRepresentative$^{\uparrow}$ *route occurs through the recorded support rows H, C, P, N. No hidden selector, quotient stream, host limit, ambient completeness field, countable-choice row, detached window, alternate dyadic ledger, or off-packet terminal seal can enter the route through* hsame*,* Cont*,* Pkg*, or* NameCert*.*

Proof. Begin with terminal stability, Definition 1205.5. It says that accepted componentwise transport preserves the displayed route μ, n, k, i, w, r, d, e and does not manufacture hidden selectors or off-packet terminal seals.

Use the nonchoice and completion boundaries, Definition 1205.11 and Definition 1205.14. These boundaries remove quotient stream equality, host limits, countable choice, and ambient completeness from the packet surface.

Finally read the support rows. The classifier permits only the componentwise hsame comparisons and Cont routes recorded by H and C, while P and N source and name the same finite packet. Therefore accepted transport and replay remain inside the recorded support scope. □

Theorem 1205.9 (Diagonal-limit representative scoped-closure package)**.** *The* DiagonalLimitRepresentative$^{\uparrow}$ *scoped-closure surface is the package consisting of the obligation-closure package of Definition 1205.6, the primitive-scope exhaustion theorem of Definition 1205.7, and the support-scope nonescape theorem of Definition 1205.8. This package binds the diagonal representative route to the displayed BEDC primitives without claiming Real completeness, quotient-stream equality, host limits, selected completed limits, countable choice, or machine verification.*

Proof. The obligation-closure package supplies the accepted carrier, classifier, NameCert obligations, route predicate obligation, and terminal stability obligation. These theorems identify the diagonal representative route and the terminal Real seal boundary.

The primitive-scope exhaustion theorem identifies the primitive rows available to the route: BHist, hsame, Cont, Pkg, NameCert, the four representative sibling surfaces, and the finite-prefix budget surface. The support-scope nonescape theorem confines every accepted transport and replay step to H, C, P, N.

Combining these two scope results with the obligation package gives a closed scoped surface for the paper chapter. The exclusions are precisely the absent coordinates named by the carrier, nonchoice boundary, completion boundary, and finite-prefix handoff, so the package records scoped closure without adding analytic completion data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$.

Constructive story: The diagonal-limit representative is generated from $BHist$ rows that record diagonal indices, finite prefixes, window ledgers, stream-name to regular-rational handoffs, and the shared terminal route. The construction follows the familiar diagonal-limit idea, but every representative is read through the carrier rather than selected from an ambient completion. The $hsame$ relation transports index and terminal rows across equivalent histories, while Cont carries the finite-prefix and Real-seal handoffs forward. The chapter's boundary, uniqueness, selector-exclusion, completion-frontier, shared-terminal, and handoff theorems keep the representative bottom-up: a terminal diagonal representative exists only as the visible route assembled from the named finite ledgers.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the diagonal-limit representative carrier, classifier, NameCert obligations, the route predicate obligation of Definition 1205.4, the terminal stability obligation of Definition 1205.5, the closure package of Definition 1205.6, primitive-scope exhaustion of Definition 1205.7, support-scope nonescape of Definition 1205.8, scoped-closure packaging of Definition 1205.9, Real-seal and completion boundaries, uniqueness, selector-exclusion, shared-terminal route, and finite-prefix budget handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeNameCertObligations`

Bridge status: none

Origin: human

Not claimed: It does not claim a selected completion point, quotient-stream equality, global limit uniqueness, or choice beyond the displayed diagonal route.

Upgrade path: A higher closure grade requires Lean verification and a public bridge for the scoped package.

Theorem 1205.10 (Diagonal-limit representative Real-seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the ordered finite route*

$$\mu, n, k, i, w, r, d, e, H, C, P, N.$$

The consumer reaches the seal row e *only after the modulus-selected family index, the selected* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been exposed on the same packet.*

Proof. Begin with the NameCert obligations. They force every accepted carrier to compute k and i before any readback row is visible.

Next read the sibling surfaces in order. The $\mathsf{StreamName}^{\uparrow}$ window w exposes the finite observation window, the $\mathsf{RegSeqRat}^{\uparrow}$ row r supplies the selected regular-rational readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d records the tolerance and arithmetic data attached to that readback.

The $\mathsf{Real}^{\uparrow}$ seal row e is downstream of these rows and is preserved only through the displayed H, C, P, N support. Therefore a consumer cannot reach e through a host real, quotient stream equality, detached tail selector, countable choice row, or ambient completeness field. □

Theorem 1205.11 (Diagonal-limit representative nonchoice boundary)**.** *An accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet exports exactly the displayed family row* X*, modulus row* μ*, request row* n*, selected indices* k, i*, selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*,* $\mathsf{Real}^{\uparrow}$ *seal row* e*, componentwise transport* H*, continuation replay* C*, package provenance* P*, and local naming row* N*. It exports no host real, quotient stream equality, hidden tail selector, countable choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier coordinates of Definition 1205.1. Every coordinate is either a displayed BEDC row, a computed finite index, or structural support for transport, replay, provenance, and naming.

Definition 1205.3 accounts for the admissible carrier and classifier rows. Definition 1205.10 accounts for every Real-facing consumer read of the packet.

The excluded objects require additional coordinates absent from the carrier. Hence they cannot be exported by the local naming certificate or recovered by a consumer of the displayed diagonal-limit representative route. □

Theorem 1205.12 (Diagonal-limit representative index stability)**.** *Let* $D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$ *and* $D' = (X', \mu', n', k', i', w', r', d', e', H', C', P', N')$ *be accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers classified together by the componentwise* hsame *rows recorded in their transport data. Then the displayed index equations are preserved:*

$$k = \mu(1/(n+1)), \qquad i = k + n + 1,$$

and

$$k' = \mu'(1/(n'+1)), \qquad i' = k' + n' + 1.$$

Consequently the selected family index and selected readback index remain the same diagonal route under accepted componentwise transport, and no classifier read may replace them by a hidden selector.

Proof. Project both carriers through Definition 1205.1. The rows k, i and k', i' are not auxiliary choices; they are the displayed computations tied to μ, n and μ', n' inside the packet.

Use the classifier of Definition 1205.2. Classification requires the modulus rows, selected family indices, readback indices, windows, readbacks, ledgers, seals, transports, continuations, provenance, and local naming rows to be related componentwise by the recorded hsame comparisons and Cont routes.

Finally apply the NameCert obligations of Definition 1205.3. Since the local certificate includes the two index computations themselves, transport of the packet preserves those displayed computations rather than producing a new tail selector. The accepted diagonal route is therefore stable under the componentwise classifier. □

Theorem 1205.13 (Diagonal-limit representative window-ledger exhaustion)**.** *Every public read of the window, readback, ledger, and seal coordinates of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

factors through the displayed sibling rows w, r, d, e*. The row* w *is read through* $\mathsf{StreamName}^{\uparrow}$*,* r *through* $\mathsf{RegSeqRat}^{\uparrow}$*,* d *through* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* e *through* $\mathsf{Real}^{\uparrow}$*, with* H, C, P, N *only transporting, replaying, sourcing, and naming those same rows.*

Proof. Start with the carrier row. The only window, regular-rational readback, dyadic arithmetic ledger, and Real-facing seal coordinates are w, r, d, e.

Read the public route in the order forced by Definition 1205.10. The selected window w exposes the finite $\mathsf{StreamName}^{\uparrow}$ observation, the readback r exposes the $\mathsf{RegSeqRat}^{\uparrow}$ row selected from that window, the ledger d records the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and arithmetic data, and the seal e is the downstream $\mathsf{Real}^{\uparrow}$ row.

It remains to exclude extra public reads. The support rows H, C, P, N preserve, replay, source, and name the displayed carrier; they do not add a second window, a second ledger, or an unrecorded Real seal. Hence every public read of w, r, d, e is exhausted by the four sibling surfaces already displayed in the carrier. □

Theorem 1205.14 (Diagonal-limit representative completion boundary)**.** *An accepted* $\mathsf{DiagonalLimitRepresentative}$ *packet supplies the diagonal representative boundary*

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N$$

and nothing stronger. In particular, it does not supply a host limit, quotient equality of streams, countable choice for a family of representatives, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The positive content is exactly the carrier and certificate surface. By Definition 1205.3, the packet admits the family, modulus, request, computed indices, selected window, readback, ledger, Real seal, transport, continuation, provenance, and local naming rows.

The consumer-facing route is already exhausted by Definition 1205.13. Any read of the Real-facing coordinate must pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the displayed $\mathsf{Real}^{\uparrow}$ seal row of the same packet.

Finally use the nonchoice boundary of Definition 1205.11. A host limit, quotient stream equality, countable choice principle, or Real-completeness theorem would require coordinates not present in the carrier. The packet therefore supplies only the diagonal representative boundary. □

Theorem 1205.15 (Diagonal-limit representative tail-selector exclusion)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the displayed tuple excludes host limits, quotient stream equality, hidden tail selectors, and countable choice rows. Any consumer of D *can read only the carrier rows, the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *obligations, the real-seal factorization route, the window-ledger exhaustion route, and the completion boundary already displayed by the packet.*

Proof. Start from the carrier of Definition 1205.1. Its coordinates are exactly the regular-Cauchy family row, modulus row, precision request, selected indices, selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal, and the support rows H, C, P, N.

Apply the NameCert obligations of Definition 1205.3. They account for every admitted row and for the two displayed index computations, so a hidden selector or countable family choice would require an additional certificate coordinate.

Use Definition 1205.10 and Definition 1205.13. These theorems force every Real-facing read through w, r, d, e and through the same support rows, not through a host limit or quotient equality of streams.

Finally apply Definition 1205.14. The boundary theorem states that the packet supplies exactly the displayed diagonal representative boundary and nothing stronger. Therefore the excluded objects are not exports of the accepted carrier. □

Theorem 1205.16 (Diagonal-limit representative StreamName–RegSeqRat handoff)**.** *Every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consumer route to the seal row* e *factors first through the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, then through the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, and only then through the downstream* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d *before reaching the* $\mathsf{Real}^{\uparrow}$ *seal row. No route may pass directly from the modulus row* μ *or request row* n *to* $\mathsf{Real}^{\uparrow}$ *without the* w, r *handoff.*

Proof. Project the accepted carrier. The tuple records $k = \mu(1/(n+1))$ and $i = k + n + 1$ before it records the selected window w and selected readback r.

Use Definition 1205.3. The local certificate admits the finite-window exposure through w and the regular-rational readback through r as separate obligations, so the packet cannot collapse them into a direct $\mathsf{Real}^{\uparrow}$ read.

Next apply Definition 1205.10. It orders every Real-facing consumer read through $\mu, n, k, i, w, r, d, e, H, C, P, N$, placing $\mathsf{StreamName}^{\uparrow}$ before $\mathsf{RegSeqRat}^{\uparrow}$ and both before the seal.

The exhaustion theorem Definition 1205.13 supplies the final constraint. Public reads of the window, readback, ledger, and seal coordinates are exhausted by w, r, d, e with H, C, P, N only transporting, replaying, sourcing, and naming those rows. Thus the route factors through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before it reaches $\mathsf{Real}^{\uparrow}$. □

Theorem 1205.17 (Diagonal-limit representative dyadic-real seal nonescape)**.** *In an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d *and* $\mathsf{Real}^{\uparrow}$ *seal* e *are consumed only through the displayed rows*

$$d, e, H, C, P, N.$$

The carrier exposes no second dyadic ledger, no off-packet real seal, no ambient completeness field, and no route from d *or* e *that bypasses the displayed transport, continuation, provenance, and local naming rows.*

Proof. Read the dyadic and real coordinates from Definition 1205.1. The only dyadic arithmetic row is d, and the only Real-facing seal row is e.

Apply the NameCert obligations. They identify dyadic ledger exposure through d, terminal seal exposure through e, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N as the complete local surface.

Use Definition 1205.10 to place d before e in every Real-facing route. Then use Definition 1205.13 to see that all public reads of d and e are exhausted by the displayed sibling rows and their support.

Finally invoke Definition 1205.14. Since the packet supplies no ambient Real-completeness principle or off-packet seal, consumption of the dyadic ledger and Real seal remains confined to d, e, H, C, P, N. □

Theorem 1205.18 (Diagonal-limit representative terminal uniqueness)**.** *Let*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

and

$$D' = (X', \mu', n', k', i', w', r', d', e', H', C', P', N')$$

be accepted $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers. Suppose the displayed modulus, request, selected index, window, readback, dyadic ledger, seal, transport, continuation, provenance, and local naming rows of* D *and* D' *are classified together by the componentwise* hsame *comparisons and* Cont *routes recorded in their packets. Then the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows* e *and* e' *are classified together by the same representative route, and the accepted classifier exports no second diagonal representative, second terminal seal, hidden tail selector, host limit, quotient stream equality, countable-choice row, or ambient completeness witness.*

Proof. Start with the local NameCert obligations. Definition 1205.3 says that every accepted representative packet exposes the modulus request, the two computed indices, the window, readback, dyadic ledger, terminal seal, and the support rows H, C, P, N as the whole local certificate surface.

Next use index stability, Definition 1205.12. The displayed equations $k = \mu(1/(n+1))$ and $i = k+n+1$ are preserved by the accepted classifier, so the two packets cannot diverge by replacing the selected family index or selected readback index with another selector.

Now apply real-seal factorization and dyadic-real seal nonescape. By Definition 1205.10, every Real-facing read reaches the seal only after μ, n, k, i, w, r, d have been exposed on the same packet. By Definition 1205.17, the dyadic and Real coordinates are consumed only through d, e, H, C, P, N and their primed counterparts.

The hypotheses identify exactly those rows on both packets. Therefore the two terminal seal rows are the same accepted representative terminal for the classifier. The nonchoice boundary and completion boundary, Definition 1205.11 and Definition 1205.14, exclude a second representative or any off-packet analytic object. □

Theorem 1205.19 (Diagonal-limit representative carrier handoff)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the finite consumer surface exposes the four sibling rows in the ordered handoff

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The row w is the only $\mathsf{StreamName}^{\uparrow}$ *window, r is the only* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, d is the only* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and e is the only terminal* $\mathsf{Real}^{\uparrow}$ *seal available to the accepted representative packet.*

Proof. Project the carrier of Definition 1205.1. Its displayed rows contain exactly one window coordinate w, one readback coordinate r, one dyadic ledger coordinate d, and one Real-facing seal coordinate e.

Apply Definition 1205.3. The local certificate assigns those four coordinates to the sibling surfaces in the stated order and accounts for all support through H, C, P, N.

Next use Definition 1205.10. Every consumer read that reaches e must pass through μ, n, k, i, w, r, d first, so the terminal Real row cannot be consumed before the StreamName, RegSeqRat, and DyadicRatCore rows have been read.

Finally apply Definition 1205.13. The public reads of w, r, d, e are exhausted by the displayed packet, while the support rows only transport, replay, source, and name those rows. Hence the ordered handoff is the complete finite route exposed by D. □

Theorem 1205.20 (Diagonal-limit representative terminal-seal choice-free totality)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the terminal seal row e is determined by the finite packet. More precisely, a Real-facing or Completion-facing consumer reaches e only after reading the family row X, modulus row μ, request n, computed indices k, i, window row w, readback row r, dyadic ledger d, and support rows H, C, P, N. No host limit, quotient stream, hidden tail selector, selected diagonal family, or countable-choice coordinate can supply an alternate terminal seal.

Proof. Project the accepted packet. The carrier handoff theorem fixes the ordered sibling route through w, r, d, e, while the selector exactness theorem fixes k and i as the displayed computations from μ and n.

Next read the seal boundary. The Real-seal route theorem confines every endpoint read to the finite path through w, r, d, H, C, P, N, and terminal uniqueness rules out a second seal row once the displayed packet rows are classified together.

Finally apply the nonchoice and completion boundaries. These boundaries exclude hidden selectors, host limits, quotient streams, and countable-choice rows, so the terminal seal is total for the accepted finite packet and cannot be replaced by an ambient analytic object. □

Theorem 1205.21 (Diagonal-limit representative cofinal-window route exactness)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

each cofinal refinement that preserves the accepted representative classifier has exactly the route

$$\mu, n, k, i, w, r, d, e, H, C, P, N.$$

The refined route is a finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *route over the same packet, not a detached window ledger, quotient stream comparison, host-limit comparison, hidden tail selector, or countable-choice selection.*

Proof. Start with cofinal-window compatibility. It states that a displayed cofinal refinement preserving μ, k, i and the finite window still follows the ordered sibling route through w, r, d, e.

Use request-monotone seal routing to place μ, n, k, i before the window and readback rows. The modulus-window-readback package then shows that every route to the terminal seal has already consumed μ, n, k, i, w, r, d.

The support rows do not add another endpoint. They transport, replay, source, and name the same finite route, and dyadic-real seal nonescape confines the dyadic and Real rows to d, e, H, C, P, N.

Thus the cofinal-window route is exact. The completion boundary and tail-selector exclusion remove detached windows, quotient comparisons, host limits, hidden selectors, and choice rows from the accepted route. □

Theorem 1205.22 (Diagonal-limit representative cofinal-refinement functoriality)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

two displayed cofinal request refinements that preserve the accepted representative classifier compose to the same ordered route

$$\mu, n, k, i, w, r, d, e, H, C, P, N.$$

The composed refinement is still a finite $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *route through* D*. It exports no second selector, no second window, no quotient-stream comparison, no host limit, no choice row, and no* $\mathsf{Real}^{\uparrow}$*-completeness coordinate.*

Proof. Apply cofinal-window route exactness to the first displayed refinement. Its classifier-preserving route is exactly $\mu, n, k, i, w, r, d, e, H, C, P, N$, with the StreamName, RegSeqRat, DyadicRatCore, and Real rows read inside the accepted packet.

Apply the same exactness theorem to the second displayed refinement after it is viewed over the already refined request. Since both refinements preserve the representative classifier, request-monotone seal routing keeps μ, n, k, i before w, r, d, e at both stages.

Now compose the two reads. The modulus-window-readback package identifies the same window, readback, dyadic ledger, and Real seal rows, so the second stage has no fresh endpoint to append beyond w, r, d, e, H, C, P, N.

Finally use dyadic-real seal nonescape and the completion boundary. They confine the dyadic and Real rows to the displayed seal route and exclude quotient-stream, host-limit, choice, and ambient-completeness coordinates. Hence successive classifier-preserving cofinal refinements collapse to the same finite L10 route through D. □

Theorem 1205.23 (Diagonal-limit representative Real-seal consumer determinacy)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

any $\mathsf{Real}^{\uparrow}$ *consumer of the representative endpoint is determined by the finite packet route*

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N.$$

Two accepted consumer reads with the same displayed rows and the same recorded hsame *and* Cont *support receive the same terminal seal row* e *and the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *predecessors. The consumer receives no host limit, quotient stream, hidden tail selector, detached window, or countable-choice row.*

Proof. First identify the consumer surface. The dependency-ready package binds the StreamName, RegSeqRat, DyadicRatCore, and Real rows into one local $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ surface, and the carrier handoff orders those rows as w, r, d, e.

Next attach the representative route. Selector exactness fixes k and i, request-monotone seal routing fixes the read order before w, r, d, e, and terminal-seal choice-free totality fixes the endpoint row e from the same finite packet.

Now compare two accepted reads. If their displayed rows and support routes agree, terminal uniqueness identifies the seal rows, while window-ledger exhaustion identifies the w, r, d predecessors on the same public packet.

The remaining exclusions follow from nonchoice boundary, tail-selector exclusion, and completion boundary. Since none of the excluded objects is a coordinate of D, the Real-seal consumer is determined by the finite route alone. □

Theorem 1205.24 (Diagonal-limit representative selector exactness)**.** *For every accepted* DiagonalLimitRepresentat *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the selector coordinates available to the classifier are exactly

$$k = \mu(1/(n+1)) \quad \textit{and} \quad i = k + n + 1.$$

No hidden tail selector, off-packet family index, countable-choice row, or quotient-stream selector can replace k *or* i *in the accepted classifier.*

Proof. Start from index stability, Definition 1205.12. The displayed equations defining k and i are preserved by accepted classifier transport, so a classified packet cannot change the family index route or the readback index route.

Apply terminal uniqueness, Definition 1205.18. Once the displayed modulus, request, index, window, readback, dyadic, seal, and support rows are classified together, the terminal Real seal is already fixed by the same representative route.

It remains to exclude replacement selectors. The nonchoice boundary and the tail-selector exclusion theorem, Definition 1205.11 and Definition 1205.15, give no packet coordinate for a hidden tail selector, countable choice row, or quotient-stream selector. Therefore the only classifier selector coordinates are k and i. □

Theorem 1205.25 (Diagonal-limit representative Real-seal route)**.** *Every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consumer that reaches the terminal row e first reads the sibling rows w, r, d and the support rows H, C, P, N. The terminal* $\mathsf{Real}^{\uparrow}$ *seal is a packet row reached by this finite route, not a* $\mathsf{Real}^{\uparrow}$ *completeness principle, host limit, quotient of streams, or ambient analytic object.*

Proof. Begin with dyadic-real seal nonescape, Definition 1205.17. It confines consumption of the dyadic ledger and terminal seal to d, e, H, C, P, N.

Use the carrier handoff theorem just proved. Before a consumer can read d or e, it must pass through the selected $\mathsf{StreamName}^{\uparrow}$ window w and the selected $\mathsf{RegSeqRat}^{\uparrow}$ readback r on the same packet.

Next apply Definition 1205.14. The boundary identifies the Real-facing row as the displayed terminal seal of the representative packet and withholds any ambient completeness witness.

Finally invoke terminal uniqueness. Since every accepted route to e has already consumed w, r, d, H, C, P, N, and no second terminal seal is exported, the Real-facing endpoint is a finite packet row rather than a Real-completeness principle. □

Theorem 1205.26 (Diagonal-limit representative cofinal-window compatibility)**.** *Let $D = (X, \mu, n, k, i, w, r, d, e, H$ be an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, and let a displayed cofinal refinement of the request preserve the same modulus row μ, the same selected diagonal index route k, i, and a refined finite* $\mathsf{StreamName}^{\uparrow}$ *window that reads the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row r,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d, and* $\mathsf{Real}^{\uparrow}$ *seal row e through the recorded support rows. Then the refined consumer follows the same ordered route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside D. It introduces no hidden selector, no second window ledger, no quotient-stream comparison, no host limit, and no ambient completeness coordinate.

Proof. Project the carrier. The request, selected indices, finite window, regular-rational readback, dyadic ledger, Real seal, and support rows are already part of the accepted tuple of Definition 1205.1.

Use the StreamName–RegSeqRat handoff, Definition 1205.16. Any route to the seal must first expose the selected $\mathsf{StreamName}^{\uparrow}$ window and then the selected $\mathsf{RegSeqRat}^{\uparrow}$ readback before it may consume the downstream dyadic ledger and Real seal.

Next apply window-ledger exhaustion, Definition 1205.13. The public reads of w, r, d, e are exhausted by those four sibling surfaces, with H, C, P, N only transporting, replaying, sourcing, and naming the same rows.

The cofinal refinement preserves the displayed request route and the same r, d, e rows, so it has no separate coordinate from which to read a hidden selector or a second window ledger. The completion boundary then removes quotient-stream, host-limit, and ambient-completeness routes. The refined consumer therefore remains on the ordered $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ route of D. □

Theorem 1205.27 (Diagonal-limit representative request-monotone seal route)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

any consumer that refines the request while preserving the accepted representative classifier must read the rows μ, n, k, i before it reads w, r, d, e. The only support rows available after this ordered read are H, C, P, N, and every $\mathsf{Real}^{\uparrow}$ *seal consumption remains confined to the displayed route*

$$\mu, n, k, i, w, r, d, e, H, C, P, N.$$

Proof. First read the index equations from the packet. By Definition 1205.12, the equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ are part of the accepted classifier surface and cannot be replaced by a request-dependent hidden selector.

Next apply real-seal factorization, Definition 1205.10. It orders every Real-facing consumer read through $\mu, n, k, i, w, r, d, e, H, C, P, N$, placing the modulus request and computed indices before the window, readback, ledger, and seal.

Finally apply dyadic-real seal nonescape, Definition 1205.17. The dyadic ledger and Real seal are consumed only through d, e, H, C, P, N, and there is no second dyadic ledger, off-packet seal, ambient completeness field, or bypass around the displayed support.

Thus a request-refinement consumer that preserves the classifier remains monotone in the displayed request route: it reads μ, n, k, i first, then w, r, d, e, and it has only H, C, P, N as support. □

Theorem 1205.28 (Diagonal-limit representative dependency-ready package)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the displayed packet is a single dependency-ready boundary for the diagonal representative route. It consumes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows in the order*

$$\mu, n, k, i, w, r, d, e, H, C, P, N,$$

and it packages them as one Pkg*-sourced* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *surface. Every downstream consumer that asks for the diagonal representative endpoint must pass through this packet rather than through a detached stream name, detached regular sequence, detached dyadic ledger, or detached real seal.*

Proof. Begin with the carrier obligations of Definition 1205.3. They place the family row, modulus row, request, computed indices, selected window, readback, dyadic ledger, terminal seal, transport, continuation, package, and local naming row in one finite certificate.

Next apply the StreamName–RegSeqRat handoff of Definition 1205.16. It forces every route to the terminal seal through w and r before the dyadic ledger and the Real-facing row can be consumed.

Use request-monotone seal routing, Definition 1205.27, to place the modulus request and computed indices before the sibling rows. Then use dyadic-real seal nonescape, Definition 1205.17, to confine the dyadic and Real endpoints to d, e, H, C, P, N.

These dependencies exhaust the same carrier. Hence D is not a collection of independent endpoint names; it is one dependency-ready package tying the diagonal representative route to the four consumed rows and to the local $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ boundary. □

Theorem 1205.29 (Diagonal-limit representative modulus-window-readback package)**.** *In an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

every consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal row* e *only through the finite modulus-window-readback route*

$$\mu, n, k, i, w, r, d.$$

The rows H, C, P, N *may transport, replay, source, and name this route, but they do not supply an alternate route to* e*.*

Proof. First read the modulus route from Definition 1205.12. The selected family index and readback index are the displayed computations attached to μ and n; a consumer cannot replace them by another selector while preserving the accepted classifier.

Next pass to the window and readback rows. By Definition 1205.16, the route to e must expose the selected $\mathsf{StreamName}^{\uparrow}$ window w and then the selected $\mathsf{RegSeqRat}^{\uparrow}$ readback r before any Real-facing seal is available.

Now consume the dyadic ledger. The window-ledger exhaustion theorem, Definition 1205.13, says that public reads of w, r, d, e are exhausted by those displayed sibling rows, with H, C, P, N only preserving the same surface.

Finally use real-seal factorization, Definition 1205.10. It orders every Real-facing read through $\mu, n, k, i, w, r, d, e, H, C, P, N$. Therefore the only path to e has already passed through μ, n, k, i, w, r, d, and the support rows cannot create an alternate endpoint. □

Theorem 1205.30 (Diagonal-limit representative Real–Completion frontier boundary)**.** *The package*

$$(X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

supports the Real-and-Completion frontier as a named diagonal representative endpoint: it reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal row through the dependency-ready route of Definition 1205.28 and the modulus-window-readback route of Definition 1205.29. It does not claim Real completeness, quotient stream equality, host limits, selected limits, hidden tail selectors, or countable choice.*

Proof. The positive frontier content is the accepted representative package. By Definition 1205.28, the packet binds the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows into a single Pkg-sourced $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ boundary.

The endpoint route is finite. By Definition 1205.29, every consumer reaches the terminal seal e only through μ, n, k, i, w, r, d, with H, C, P, N serving as support for the same displayed route.

It remains to state the frontier. The completion boundary of Definition 1205.14 says that the carrier supplies the diagonal representative boundary and nothing stronger. The tail-selector exclusion of Definition 1205.15 rules out hidden selectors, host limits, quotient stream equality, and countable-choice rows.

Therefore the package is ready for Real-and-Completion consumers that need the named diagonal representative endpoint, while the analytic completion principles and choice-sensitive comparisons remain outside this certificate. □

Theorem 1205.31 (Diagonal-limit compatibility representative handoff)**.** *Let* $K = (R, T, S, D, W, Q, L, H_K, C_K, P$ *be an accepted* $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$ *packet whose terminal read has the same four L10 faces as an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$E = (X, \mu, n, k, i, w, r, d, e, H_E, C_E, P_E, N_E),$$

namely the same displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows under the recorded support. Then the terminal read of* K *factors through*

the representative carrier E before reaching the terminal Real seal. It inherits the representative nonescape boundary: no quotient stream, host real equality, selected limit, hidden tail selector, choice datum, ambient completeness row, or second terminal seal can enter the handoff.

Proof. First read the compatibility packet through open-phase synchronization. Definition 646.174 places the terminal read of K on the four finite exit faces $\mathsf{StreamName}^{\uparrow}(W)$, $\mathsf{DyadicRatCore}^{\uparrow}(T)$, $\mathsf{RegSeqRat}^{\uparrow}(Q)$, and $\mathsf{Real}^{\uparrow}(L)$, carried by the displayed support rows.

Next use open-phase four-face readiness, Definition 646.179. It says that readiness of the terminal compatibility read is exactly the displayed four-face surface over its support rows, with no quotient stream, host real equality, selected limit, choice datum, completeness datum, or bridge row.

Now compare those faces with the representative carrier. The hypothesis identifies the compatibility window, readback, ledger, and Real seal with the representative rows w, r, d, e. By Definition 1205.13, every public read of those four representative coordinates is already exhausted by the same sibling surfaces and the support rows H_E, C_E, P_E, N_E.

Finally apply the representative dyadic-real seal nonescape theorem, Definition 1205.17. Since the shared dyadic ledger and Real seal are consumed only through the displayed representative support, the compatibility terminal read factors through E before it reaches the seal. The excluded host, quotient, selector, choice, completeness, and second-seal routes are absent from both the compatibility readiness surface and the representative carrier boundary. □

Definition 1205.32 (Diagonal-limit representative shared-terminal carrier)**.** *A shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is a finite* BHist *packet extending*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

from Definition 1205.1 by finite-prefix budget rows

$$F = (Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F)$$

from chapter 919. The extended packet records the following additional rows:

1. *Q is the finite diagonal request whose precision and prefix depth cover the selected representative read n, k, i;*
2. *B is the finite observation-budget selector row admitting Q before the selected* $\mathsf{StreamName}^{\uparrow}$ *window is served;*
3. *W is the finite prefix-window ledger containing the representative window w and the prefix windows required for the same terminal read;*
4. *D_F is the prefix-depth exhaustion row accounting for every served offset in W and refusing outside-prefix reads before the regular readback or seal is consumed;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row compatible with the representative readback r;*
6. *S is the completion-facing or* $\mathsf{Real}^{\uparrow}$ *seal row compatible with the representative terminal seal e;*
7. *E_F, H_F, C_F, P_F, N_F record the finite-prefix exclusion, transport, continuation, provenance, and local naming rows used by the same terminal consumer.*

The shared-terminal carrier exports the representative row and the finite-prefix budget row as one ordered surface. It exports no quotient stream, host real equality, selected limit, hidden tail schedule, countable choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Theorem 1205.33 (Diagonal-limit representative shared-terminal route)**.** *Every terminal seal read of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the ordered public surface*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{FinitePrefixLimitBudget}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The representative rows w, r, d, e *and the finite-prefix rows* Q, B, W, D_F, R, S *are consumed as one terminal route; no consumer may reach the* $\mathsf{Real}^{\uparrow}$ *seal before the finite-prefix budget has displayed its selector, prefix window, exhaustion row, and regular readback.*

Proof. Start with the representative carrier. By Definition 1205.16, every route to the representative seal exposes the selected $\mathsf{StreamName}^{\uparrow}$ window w before the selected $\mathsf{RegSeqRat}^{\uparrow}$ readback r, and then consumes the downstream $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d before the $\mathsf{Real}^{\uparrow}$ seal row e.

Attach the finite-prefix budget packet. The seal-ordering theorem Definition 919.3 forces every budget-facing seal read through

$$Q \to B \to W \to D_F \to R \to S.$$

Thus the shared terminal route cannot read the seal before the selector, prefix-window ledger, depth-exhaustion row, and regular readback row have all been displayed.

Now compare the two terminal surfaces. The shared-terminal carrier identifies the representative window, readback, and seal with the budget window, readback, and seal rows, while the representative dyadic ledger remains the arithmetic ledger between readback and seal. The consumer-boundary theorem Definition 919.10 and the representative window-ledger exhaustion theorem Definition 1205.13 both confine public reads to the displayed rows.

Therefore a terminal consumer factors through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that finite order. There is no bypass from the representative modulus, from the prefix selector, or from the dyadic ledger directly to a completed Real object. □

Theorem 1205.34 (Diagonal-limit representative finite-prefix budget handoff)**.** *The shared-terminal handoff from Definition 1205.32 preserves the representative nonchoice boundary and the finite-prefix consumer boundary. It supplies a concrete terminal interface for the Real-completion route, but it does not supply quotient-stream equality, host real equality, selected completed limits, hidden tail schedules, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the representative side. The nonchoice boundary Definition 1205.11 and completion boundary Definition 1205.14 say that the representative packet exports only $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and no host limit, quotient stream, hidden selector, countable choice row, or ambient completeness principle.

Next read the finite-prefix side. By Definition 919.15, a $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or completion-facing consumer receives exactly the prefix-budget route $Q \to B \to W \to D_F \to R \to S$ together with E_F, H_F, C_F, P_F, N_F. By Definition 919.12, quotient streams, selected limits, host real equality, hidden tail schedules, countable choice, and ambient completeness cannot replace that route.

Finally use the shared-terminal route theorem above. It composes the representative route with the finite-prefix budget route without adding a new coordinate: the terminal read is still a finite BEDC packet read through StreamName, RegSeqRat, DyadicRatCore, FinitePrefixLimitBudget, and Real rows. Hence the handoff gives the Real-completion route a concrete L10 terminal surface while preserving the nonchoice and no-completeness boundaries on both sides. □

Theorem 1205.35 (Diagonal-limit representative public exactness package)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the public exactness package is precisely the ordered surface*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over the representative rows

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N$$

and the shared-terminal finite-prefix rows

$$Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F.$$

The package has no additional public row beyond these displayed carrier, route, support, budget, exclusion, continuation, provenance, and naming rows.

Proof. First take the terminal route from Definition 1205.33. It orders every terminal consumer through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the type-level surface is already fixed.

Next read the finite-prefix handoff in Definition 1205.34. It says that the representative rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and the budget rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ are consumed as one terminal interface, with the nonchoice and no-completeness boundaries preserved on both sides.

The dependency-ready package Definition 1205.28 supplies the Pkg-sourced $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ boundary for the representative half of the route. The Real–Completion frontier boundary Definition 1205.30 then identifies the terminal endpoint as the named diagonal representative frontier and excludes stronger analytic structure.

Therefore the public exactness package is exactly the displayed ordered surface over the displayed rows. Any extra public row would either precede one of the ordered dependencies, duplicate the finite-prefix budget route, or add a Real-completion principle explicitly excluded by the frontier boundary. □

Theorem 1205.36 (Diagonal-limit representative completion consumer factorization)**.** *Every Real-and-Completion consumer of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the public exactness package of Definition 1205.35. Such a consumer cannot bypass that package by quotient stream equality, host real equality, selected completed limits, hidden tail schedules, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the exactness package just established. It presents the only public route from the selected stream window and regular readback to the dyadic ledger, finite-prefix budget, and terminal Real seal.

The shared-terminal route theorem Definition 1205.33 supplies the factorization order. The finite-prefix budget handoff Definition 1205.34 supplies the consumer boundary for completion-facing reads, including the refusal of quotient streams, selected limits, host equality, hidden schedules, choice data, and ambient completeness.

It remains to connect this route to the representative frontier. The dependency-ready package Definition 1205.28 binds the displayed rows into the accepted representative certificate, while Definition 1205.30 states that the frontier is a named diagonal representative endpoint and not a completed Real object supplied by the host.

Hence every Real-and-Completion consumer reaches the endpoint through the public package. A bypass would have to supply one of the excluded equality, limit, schedule, choice, or completeness

rows, and those rows are absent from both the representative frontier and the finite-prefix handoff. □

Theorem 1205.37 (Diagonal-limit representative readback-index scope)**.** *Every completion-facing consumer of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier must expose the displayed readback-index route*

$$k = \mu(1/(n+1)), \qquad i = k + n + 1$$

before the support rows of the carrier can be consumed. More precisely, the consumer must first display the selected $\mathsf{StreamName}^{\uparrow}$ *window* w*, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* e *in the order prescribed by the public exactness package; the support rows* H, C, P, N *may only transport, replay, source, and name that already displayed route.*

Proof. Begin with selector exactness, Definition 1205.24. It fixes k and i as the displayed computations from μ and n, so a completion-facing consumer cannot substitute an off-packet family index or readback index.

Next use the modulus-window-readback package, Definition 1205.29. It places every route to the terminal seal through μ, n, k, i, w, r, d, with H, C, P, N confined to transport, replay, provenance, and local naming.

Finally apply the completion consumer factorization of Definition 1205.36. Since every Real-and-Completion consumer factors through the public exactness package, the support rows can be consumed only after the readback-index scope has exposed the selected window, readback, ledger, and seal rows. □

Theorem 1205.38 (Diagonal-limit representative shared-terminal stability)**.** *Let two accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers be classified together by their componentwise* hsame *rows and displayed* Cont *replay. Then the shared-terminal carrier preserves the same ordered route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{FinitePrefixLimitBudget}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the representative rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ *and the finite-prefix budget rows* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$*. The transport does not create a second terminal route, a detached budget row, a host Real equality, or an ambient completeness witness.*

Proof. First read the shared-terminal route, Definition 1205.33. It orders terminal consumption through the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real rows as one public surface.

Next use the finite-prefix budget handoff, Definition 1205.34. The handoff identifies the representative and budget coordinates as a single terminal interface while preserving the nonchoice and no-completeness boundaries.

Finally apply the public exactness package of Definition 1205.35. Componentwise hsame and displayed Cont can preserve only the rows that package already exposes, so the shared terminal remains stable under the same ordered route and cannot acquire any detached endpoint or host equality row. □

Theorem 1205.39 (Diagonal-limit representative frontier consumer obligation package)**.** *The downstream obligation surface for a Real-and-Completion consumer of an accepted shared-terminal*

$\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is the package consisting of the public exactness package of Definition 1205.35, the completion consumer factorization of Definition 1205.36, the readback-index scope of Definition 1205.37, and the shared-terminal stability theorem of Definition 1205.38. The package exposes only the finite BEDC route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; it excludes quotient-stream equality, host Real equality, selected completed limits, hidden tail schedules, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The positive surface is fixed by public exactness. It lists the representative rows and finite-prefix budget rows and orders them through the five named BEDC components without adding any public row.

The consumer-facing factorization then makes that surface mandatory for every Real-and-Completion read. The readback-index scope refines the entry condition: $k = \mu(1/(n+1))$, $i = k + n + 1$, w, r, d, and e must be visible before the support rows are consumed.

The stability clause closes the package under the accepted classifier. Componentwise hsame and displayed Cont preserve the same shared-terminal route, while the handoff and frontier boundaries remove quotient equality, host equality, selected limits, hidden schedules, choice data, and ambient completeness from the consumer obligation surface. □

Theorem 1205.40 (Diagonal-limit representative finite-request section exactness)**.** *Every accepted request row of a shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the displayed finite request section*

$$Q, \quad B, \quad k = \mu(1/(n+1)), \quad i = k + n + 1, \quad W, \quad w, \quad r, \quad d, \quad e, \quad S.$$

Thus the request first reads the finite-prefix request and terminal budget, then the representative modulus and selected index, then the $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. No accepted request row can be routed through a later host real, quotient-stream equality, hidden selector, countable-choice family, or ambient completeness witness.*

Proof. Start from the accepted carrier and classifier. The frontier consumer obligation package Definition 1205.39 says that every completion-facing request is consumed through the public exactness package and the displayed readback-index route.

The shared-terminal route theorem Definition 1205.33 supplies the ordered terminal path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The finite-prefix budget handoff Definition 1205.34 identifies the request and budget rows Q, B, W, D_F, R, S with the representative window, readback, and seal rows.

Finally, the Real-seal factorization Definition 1205.10 and the nonchoice boundary Definition 1205.11 exclude any replacement by host equality, quotient equality, selected limits, hidden selectors, choice data, or ambient completeness. The only remaining request section is the displayed finite row sequence. □

Theorem 1205.41 (Diagonal-limit representative shared-window readback determinacy)**.** *Let two accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers be classified by the same finite request row and the same shared terminal budget. Then the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route are determined up to the recorded componentwise* hsame *rows of the classifier. In particular, a second carrier with the same request and terminal budget cannot expose a different public window-readback-ledger-seal route.*

Proof. First use the carrier and classifier rows. The common finite request and shared terminal budget place both carriers in the same public exactness package, so the request section exactness theorem above fixes the same finite row sequence for both reads.

Next apply the shared-terminal route Definition 1205.33. Since the terminal route is finite and ordered, the two carriers can compare only the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The finite-prefix budget handoff Definition 1205.34 and Real-seal factorization Definition 1205.10 identify the budget readback and terminal seal with that same route. The nonchoice boundary Definition 1205.11 removes any unrecorded selector or ambient equality, so determinacy is exactly the recorded componentwise hsame comparison. □

Theorem 1205.42 (Diagonal-limit representative Real-seal nonchoice export)**.** *The Real-facing seal exported by an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is only the displayed terminal row of the finite route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

It cannot be replaced by host real equality, quotient-stream equality, a hidden selector, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness; any standard bridge or downstream Real-completion consumer must export precisely this displayed terminal row.*

Proof. The shared-terminal route Definition 1205.33 gives the positive export: a finite terminal row reached through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real components.

The finite-prefix budget handoff Definition 1205.34 and Real-seal factorization Definition 1205.10 show that the Real-facing endpoint is the same terminal seal row already carried by the request and budget interface.

The nonchoice boundary Definition 1205.11 supplies the negative export clause. It rules out host real equality, quotient-stream equality, hidden selectors, countable choice, and ambient completeness, so the exported Real-facing content is exactly the displayed terminal row and nothing stronger. □

Theorem 1205.43 (Diagonal-limit representative bridge precondition package)**.** *Any later standard bridge for the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *chapter must preserve the same finite packet route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

over the rows named in Definition 1205.35. The bridge precondition may transport the package into a standard schema, but it may not replace the route by quotient equality, host Real equality, selected completed limits, hidden tail schedules, countable choice, or ambient completeness, and it may not identify the paper closure axis with Lean bridge checking.

Proof. The route to be preserved is fixed by Definition 1205.33. It is a finite terminal packet route through the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real rows, with the representative and budget coordinates read as one public surface.

The exact rows to be preserved are fixed by the finite-prefix budget handoff Definition 1205.34 and the dependency-ready package Definition 1205.28. Together they identify the carrier rows, support rows, budget rows, exclusion row, continuation row, provenance row, and naming row that the bridge must transport.

Finally, the Real–Completion frontier boundary Definition 1205.30 gives the negative side of the precondition: no quotient stream equality, host equality, selected completed limit, hidden tail schedule, choice principle, or ambient completeness is part of the chapter certificate. The bridge may check a schema for the same finite route, but that verification status is separate from the paper closure axis recorded by the chapter. □

Theorem 1205.44 (Diagonal-limit representative root-readiness triad)**.** *Every downstream Real-and-Completion consumer of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier must expose the following root-readiness surface before it consumes any support row: the modulus request* μ, n*, the selected diagonal indices* $k = \mu(1/(n+1))$ *and* $i = k+n+1$*, the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, the* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *rows* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* e*. The support rows* H, C, P, N *may only transport, replay, source, and name that displayed root surface; they may not supply a separate selector, ledger, seal, host equality, quotient equality, choice row, or ambient completion principle.*

Proof. The carrier of Definition 1205.1 fixes the representative route $X, \mu, n, k, i, w, r, d, e, H, C, P, N$, and the shared-terminal carrier of Definition 1205.32 attaches the finite-prefix budget rows to the same terminal read.

The public exactness package Definition 1205.35 orders this route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The completion consumer factorization Definition 1205.36 makes that ordered package mandatory for any Real-and-Completion consumer.

Finally apply the readback-index scope Definition 1205.37 and the Real-seal nonchoice export Definition 1205.42. They force the selected window, readback, dyadic ledger, finite-prefix rows, and Real seal to be displayed before the support rows are used, and they exclude the hidden selector, host, quotient, choice, and ambient-completeness alternatives. □

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeRootReadinessTriad`

Theorem 1205.45 (Diagonal-limit representative modulus-window root unblock)**.** *The selected-index and selected-window part of the root-readiness surface is rooted entirely in the displayed rows of the shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet. In particular, the computation of* k *and* i*, the window row* w*, the readback row* r*, and their transport through* H *are read from the finite* BHist *packet, the recorded* hsame *comparisons, the displayed* Cont *replay, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row. No downstream consumer may replace this root by an ambient stream equality, a hidden modulus selector, or an unrecorded completion witness.*

Proof. Start with the NameCert obligations of Definition 1205.3. They list the modulus request, diagonal-index computation, finite-window exposure, regular-rational readback, transport, continuation, package provenance, and local naming rows as the certificate surface.

The modulus-window-readback package Definition 1205.29 then orders every route to the terminal seal through μ, n, k, i, w, r, d, with H, C, P, N preserving the same displayed route. The readback-index scope Definition 1205.37 makes the selected index and selected window visible before those support rows are consumed.

Thus the root for the selected window is not a new analytic principle. It is the already displayed BHist row together with hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, and any unrecorded selector or completion witness lies outside the accepted packet. □

Theorem 1205.46 (Diagonal-limit representative terminal seal root unblock)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is reached only through the finite route*

$$\mu, n, k, i, w, r, d, Q, B, W, D_F, R, S, e, H, C, P, N, E_F, H_F, C_F, P_F, N_F.$$

The route passes through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. It never reaches the terminal seal by quotient stream equality, host real equality, hidden selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First use the shared-terminal route Definition 1205.33. It forces every terminal read through the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real rows as one ordered surface.

Next use the finite-prefix budget handoff Definition 1205.34. It joins the representative route to the rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$, so the terminal seal cannot be consumed before the budget selector, prefix window, depth-exhaustion row, and regular readback have appeared.

Finally apply Real-seal factorization Definition 1205.10 and Real-seal nonchoice export Definition 1205.42. These identify e as the displayed terminal row of the finite route and remove quotient, host, selector, choice, and ambient-completeness routes from the seal. □

Theorem 1205.47 (Diagonal-limit representative dependency-readiness scope)**.** *The root-readiness package consumed by downstream L10 Real-and-Completion chapters consists of the root-readiness triad of Definition 1205.44, the modulus-window root of Definition 1205.45, and the terminal-seal root of Definition 1205.46. This package is exactly the dependency-ready public route supplied by the carrier, the shared-terminal carrier, the public exactness package, the completion consumer factorization, the readback-index scope, the Real-seal nonchoice export, and the bridge precondition package. It is not a claim of quotient stream equality, host Real equality, hidden selector availability, choice, ambient Real completeness, or Lean bridge checking.*

Proof. The triad theorem names the downstream obligations: the modulus request, selected indices, window, readback, dyadic ledger, finite-prefix budget rows, and Real seal must all be displayed before support rows are consumed.

The modulus-window theorem supplies the entry root through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The terminal-seal theorem supplies the exit root through the finite StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real route.

It remains to package the surface for downstream chapters. The dependency-ready package Definition 1205.28, public exactness package Definition 1205.35, completion consumer factorization Definition 1205.36, and bridge precondition package Definition 1205.43 all preserve the same finite route and the same exclusions. Therefore downstream L10 chapters may consume precisely this root-readiness surface and nothing stronger. □

Theorem 1205.48 (Diagonal-limit representative budget-source determinacy)**.** *Let A and A' be accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers. Suppose they display the same finite-prefix budget rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ and the same representative selected-index route $\mu, n, k, i, w, r, d, e, H, C, P, N$. Then their terminal* $\mathsf{Real}^{\uparrow}$ *seal source is the same up to the recorded componentwise* hsame *rows and the displayed* Cont *replay carried by that route. In particular, the terminal seal source is determined by the visible budget rows and selected-index route; it cannot be replaced by quotient-stream equality, host Real equality, hidden selectors, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The root-readiness triad of Definition 1205.44 first fixes the visible data that a downstream consumer may inspect: the modulus request, selected indices, window row, readback row, dyadic ledger, finite-prefix budget rows, terminal seal, and support rows.

Apply the terminal-seal root unblock, Definition 1205.46. Since both carriers expose the same rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ and the same route $\mu, n, k, i, w, r, d, e, H, C, P, N$, each terminal $\mathsf{Real}^{\uparrow}$ seal read is forced through the identical finite StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real surface.

Finally use the dependency-readiness scope of Definition 1205.47. It permits later Real-and-Completion consumers to consume only the recorded hsame comparisons and the displayed Cont replay on this route. Thus the two accepted carriers name the same terminal seal source up to those componentwise records, while quotient-stream equality, host Real equality, hidden selectors, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the accepted package. □

Theorem 1205.49 (Diagonal-limit representative root-readiness exactness)**.** *The root-readiness package of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is exact for downstream Real-and-Completion consumption. Coverage means that every accepted downstream consumer factors through the displayed L10 route*

$$\mu, n, k, i, w, r, d, Q, B, W, D_F, R, S, e, H, C, P, N, E_F, H_F, C_F, P_F, N_F.$$

Soundness means that each exposed row of this route is sourced by the cited $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. Completeness means that no accepted Real-and-Completion consumer route exists outside this finite packet. The exactness statement carries no quotient stream equality, host Real equality, hidden selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Coverage is the content of the root-readiness triad, the modulus-window root unblock, the terminal-seal root unblock, and the dependency-readiness scope. These results force each downstream consumer to expose the modulus request, selected indices, window, readback, dyadic ledger, finite-prefix budget rows, terminal seal, and support rows before it consumes the diagonal representative.

Soundness is inherited from the public exactness package and the bridge precondition package. The public package sources the route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while the bridge precondition package preserves the same displayed sources when the route is handed to later consumers.

Completeness follows because the dependency-readiness theorem permits downstream L10 chapters to consume precisely this root-readiness surface and nothing stronger. Any proposed route using quotient stream equality, host Real equality, a hidden selector, choice, or ambient Real completeness fails one of the displayed support obligations and therefore is not an accepted consumer route for this packet. □

Theorem 1205.50 (Diagonal-limit representative selected-window determinacy)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the fixed displayed rows* μ, n, k, i, H, C, P, N *determine the selected* $\mathsf{StreamName}^{\uparrow}$ *window and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback up to the recorded* hsame *rows and the displayed* Cont *replay carried by the packet. The determinacy is internal to the finite root-readiness route: it does not use a hidden modulus selector, quotient stream equality, host stream equality, countable choice, or ambient completion.*

Proof. The root-readiness triad of Definition 1205.44 identifies $\mu, n, k, i, w, r, H, C, P, N$ as displayed rows of the accepted consumer surface. The modulus-window root unblock of Definition 1205.45 then sources the chosen window and readback through the recorded BHist row, hsame comparisons, Cont replay, Pkg provenance, and NameCert row.

The terminal-seal root unblock of Definition 1205.46 keeps the same selected window and readback on the route to the terminal seal, while the dependency-readiness scope of Definition 1205.47 prevents a downstream Real-and-Completion consumer from importing any stronger equality principle.

Finally, the budget-source determinacy theorem Definition 1205.48 and the root-readiness exactness theorem Definition 1205.49 show that all accepted support reads factor through this displayed route. Hence fixed μ, n, k, i, H, C, P, N determine the selected $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback only through the recorded support rows, with no extra selector or quotient route. □

Theorem 1205.51 (Diagonal-limit representative terminal-seal determinacy)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the visible finite-prefix budget rows* $Q, B, W, D_F, R, S, E_F, H_F$ *together with* w, r, d *determine the terminal* $\mathsf{Real}^{\uparrow}$ *seal source up to the recorded* hsame *support rows and the displayed* Cont *replay. This terminal-seal determinacy excludes host* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the root-readiness triad Definition 1205.44 and the terminal-seal root unblock Definition 1205.46. Together they place $w, r, d, Q, B, W, D_F, R, S, e, E_F, H_F, C_F, P_F, N_F$ on the same finite route to the terminal $\mathsf{Real}^{\uparrow}$ seal.

The modulus-window root unblock Definition 1205.45 fixes the selected window and readback source used by that route, and the dependency-readiness scope Definition 1205.47 restricts later consumers to the displayed hsame and Cont support rows.

Apply budget-source determinacy, Definition 1205.48, to identify the terminal seal source from the visible budget and selected-index rows. The root-readiness exactness theorem Definition 1205.49 supplies the route-exhaustion clause, so host $\mathsf{Real}^{\uparrow}$ equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not admissible sources for the seal. □

Theorem 1205.52 (Diagonal-limit representative consumer route exhaustion)**.** *Every downstream Real-and-Completion consumer accepted by the root-readiness exactness theorem factors through the ordered route*

$$\mu, n, k, i, w, r, d, Q, B, W, D_F, R, S, e, H, C, P, N, E_F, H_F, C_F, P_F, N_F.$$

No accepted consumer may use any support surface outside this ordered route, including a hidden selector, quotient stream equality, host $\mathsf{Real}^{\uparrow}$ *equality, countable choice, or ambient completion principle.*

Proof. The root-readiness triad Definition 1205.44 lists the rows that a downstream Real-and-Completion consumer must expose. The modulus-window root unblock Definition 1205.45 supplies the selected-window entry, and the terminal-seal root unblock Definition 1205.46 supplies the exit through the finite-prefix budget and terminal $\mathsf{Real}^{\uparrow}$ seal.

The dependency-readiness scope of Definition 1205.47 states that downstream chapters may consume precisely this finite root-readiness surface. The budget-source determinacy theorem Definition 1205.48 fixes the terminal seal source once the visible budget and selected-index rows are fixed.

It remains only to exclude extra routes. The root-readiness exactness theorem Definition 1205.49 gives coverage, soundness, and completeness for the displayed packet, so every accepted consumer factors through the ordered route above and no other support surface is available to the consumer. □

Theorem 1205.53 (Diagonal-limit representative root-index coverage)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, every root-index consumer of the diagonal representative must expose the finite equations*

$$k = \mu(1/(n+1)), \qquad i = k + n + 1$$

before it consumes the visible route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Concretely, the consumer reads the selected window* w*, regular-rational readback* r*, dyadic ledger* d*, finite-prefix rows* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$*, and terminal seal* e *as one finite index surface; it receives no quotient-stream equality, host Real equality, hidden tail selector, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier of Definition 1205.1. Its displayed rows already contain the modulus request μ, n, the computed indices k, i, the selected $\mathsf{StreamName}^{\uparrow}$ window w, the $\mathsf{RegSeqRat}^{\uparrow}$ readback r, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger d, and the terminal $\mathsf{Real}^{\uparrow}$ seal e.

Attach the shared-terminal carrier of Definition 1205.32. It places the finite-prefix budget rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ on the same terminal read, so the root-index consumer sees one ordered route rather than a detached budget.

Finally use the root-readiness triad Definition 1205.44 and the readback-index scope Definition 1205.37. They force the equations for k and i to be visible before support rows are consumed, while the shared-terminal route supplies the StreamName, RegSeqRat, DyadicRatCore, FinitePrefixLimitBudget, and Real surfaces. □

Theorem 1205.54 (Diagonal-limit representative root-index soundness)**.** *The root-index surface of Definition 1205.53 is sound for the Real-and-Completion route. Each exposed row is sourced by the named sibling certificate:* w *by* $\mathsf{StreamName}^{\uparrow}$*,* r *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* d *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ *by* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, and* e *by the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. The finite equations* $k = \mu(1/(n+1))$ *and* $i = k + n + 1$ *are the only index computations admitted by this surface.*

Proof. First read the public exactness package Definition 1205.35. It identifies the ordered public route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ and lists the representative and finite-prefix rows that the route may expose.

Next apply the terminal-seal root unblock Definition 1205.46. It shows that the terminal seal is reached only through the displayed finite rows, so none of the sibling surfaces can be replaced by an off-packet equality, selector, choice datum, or completion principle.

The formal handoff boundary Definition 1205.69 supplies the remaining boundary. A later handoff may encode the displayed BHist, hsame, Cont, Pkg, NameCert, and sibling rows, but it cannot import any stronger analytic object. Hence the root-index route is sourced exactly by the named sibling certificates and by the two finite index equations. □

Theorem 1205.55 (Diagonal-limit representative root-index readiness package)**.** *The root-index readiness package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consists exactly of the coverage theorem Definition 1205.53, the soundness theorem Definition 1205.54, the root-readiness exactness theorem Definition 1205.49, the shared-terminal route Definition 1205.33, and the formal handoff boundary*

Definition 1205.69. Downstream Real-and-Completion consumers may use this package as a finite root-index surface, but it does not assert Real completeness, host real equality, quotient stream equality, selected limits, hidden tail schedules, countable choice, or any Lean bridge check.

Proof. Coverage supplies the visible data: the modulus request, the two finite index equations, the selected window and readback, the dyadic ledger, the finite-prefix budget rows, and the terminal Real seal.

Soundness supplies the source of each row through the StreamName, RegSeqRat, DyadicRatCore, FinitePrefixLimitBudget, and Real certificates. The root-readiness exactness theorem then states that downstream Real-and-Completion consumers factor through this finite packet and no stronger support route.

It remains to package the result at the handoff boundary. The shared-terminal route keeps the representative and finite-prefix rows on one terminal surface, while the formal handoff boundary confines later formal work to the displayed BEDC rows. Therefore the readiness package is exactly the finite root-index surface described above, with all host, quotient, selector, choice, completion, and bridge claims outside its scope. □

Definition 1205.56 (Diagonal-limit representative cofinal-terminal budget bridge)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the cofinal-terminal budget bridge is the finite* BHist *packet*

$$G = (X, \mu, n, k, i, w, r, d, e, H, C, P, N, Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F, B_M, M_M, W_M, T_M, D_M, R_M, L_M, E$$

The first block $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ *is the representative route of Definition 1205.1; the second block* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ *is the finite-prefix budget route of Definition 1205.32; and the third block* $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M, H_M, C_M, P_M, N_M$ *is the cofinal modulus route supplied by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *in chapter 1198. The bridge identifies the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w *with the cofinal window row* T_M*, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r *with* R_M*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d *with* D_M*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* e *with* E_M*, while keeping the finite-prefix selector and exhaustion rows visible before the terminal seal is consumed.*

Theorem 1205.57 (Diagonal-limit representative cofinal-budget seal totality)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from the bridge of Definition 1205.56 factors through*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{CofinalModulusSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

before any completion-facing consumer reads it. Thus the representative rows, finite-prefix rows, and cofinal modulus rows form one terminal budget route, and the terminal consumer receives no independent Real-completion principle beyond that finite route.

Proof. Start with the root-index readiness package, Definition 1205.55. It exposes μ, n, k, i, w, r, d, e together with the finite-prefix rows and makes the equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ visible before downstream support rows are consumed.

Next use the public exactness package, Definition 1205.35. It orders the representative and finite-prefix portions through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so there is no detached terminal read on the representative side.

Attach the cofinal modulus seal route from chapter 1198. Its scoped source inventory and primitive route exhaustion identify $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M, H_M, C_M, P_M, N_M$ as the available cofinal budget rows for a completion-facing seal read.

Finally apply the formal handoff boundary, Definition 1205.69. The handoff may encode the displayed BHist, hsame, Cont, Pkg, NameCert, and sibling rows, but it cannot import a stronger analytic object. Therefore every terminal seal read factors through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real surfaces before any completion-facing consumer can use it. □

Theorem 1205.58 (Diagonal-limit representative terminal consumer cofinal nonescape)**.** *No terminal consumer of the cofinal-terminal budget bridge may replace the displayed route by quotient stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, a selected limit, a hidden tail schedule, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any accepted consumer must read only the representative rows, finite-prefix budget rows, cofinal modulus rows, and structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed in Definition 1205.56.*

Proof. The root-index readiness package supplies the representative boundary. It admits the selected window, readback, dyadic ledger, finite-prefix budget rows, and terminal seal, while excluding quotient-stream equality, host equality, hidden selectors, choice rows, ambient completeness, and bridge claims.

The public exactness package supplies the terminal route boundary. A completion-facing consumer can only use the ordered surface through the named sibling certificates, so it has no second entrance into the terminal $\mathsf{Real}^{\uparrow}$ seal.

The cofinal modulus seal chapter supplies the cofinal budget boundary. Its consumer exactness and primitive route exhaustion confine cofinal budget reads to the displayed finite rows and exclude detached selectors, quotient streams, selected host limits, countable choice, and ambient completeness.

It remains to close the formal interface. The formal handoff boundary allows later formal consumers to encode exactly the displayed BEDC rows. Since every possible terminal read is already accounted for by those rows, any quotient, host equality, selected limit, hidden schedule, choice, or ambient-completeness coordinate would be outside the bridge rather than a consumer of it. □

Theorem 1205.59 (Diagonal-limit representative cofinal-terminal route determinacy)**.** *For any accepted bridge packet G of Definition 1205.56, the displayed representative rows, finite-prefix rows, and cofinal-modulus rows determine one terminal route through*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{CofinalModulusSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

If a second accepted terminal read has the same representative rows, finite-prefix rows, cofinal-modulus rows, transport rows, continuation rows, provenance rows, and local NameCert *rows, then it is the same route through the displayed packet G, not a second* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the finite packet rows of G. The definition of the cofinal-terminal budget bridge displays the representative block, the finite-prefix budget block, and the cofinal modulus block, together with the identifications $w = T_M$, $r = R_M$, $d = D_M$, and $e = E_M$. Hence any accepted terminal read with the same displayed rows has the same StreamName window, RegSeqRat readback, DyadicRat ledger, finite-prefix budget rows, cofinal modulus rows, and terminal Real row.

Next apply Definition 1205.57. The positive route to the terminal seal factors through the displayed chain

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{CofinalModulusSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus the terminal consumer reads the Real seal only after the representative, finite-prefix, and cofinal-modulus rows have already fixed the route.

It remains to exclude an apparent second route. Apply Definition 1205.58. A competing accepted read cannot introduce quotient stream equality, host $\mathsf{Real}^{\uparrow}$ equality, a hidden schedule, countable choice, ambient completeness, or a detached terminal seal. With those escape coordinates absent, equality of the displayed representative, finite-prefix, cofinal-modulus, transport, continuation, provenance, and local NameCert rows leaves no second Real seal to branch into. □

Theorem 1205.60 (Diagonal-limit representative cofinal-index source)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the cofinal-index source is the finite surface that reads the equations*

$$k = \mu(1/(n+1)), \qquad i = k + n + 1$$

through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows used by the cofinal-terminal budget bridge. Thus the source of the selected window* w*, readback* r*, dyadic ledger* d*, finite-prefix budget rows, cofinal modulus rows, and terminal seal* e *is visible before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Project the root-index readiness package of Definition 1205.55. It already exposes μ, n, k, i, w, r, d, e and forces the finite equations for k and i to appear on the root-index surface.

Next read the cofinal-terminal budget bridge of Definition 1205.56. Its packet attaches the finite-prefix rows and the cofinal modulus rows to the same terminal read, with w, r, d, e identified with the matching cofinal window, readback, dyadic, and seal rows.

Finally apply the cofinal-budget seal totality theorem Definition 1205.57. It orders the entire route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the cofinal-index equations are sourced by the displayed rows rather than by a host equality, quotient stream, selected limit, hidden schedule, choice row, or ambient completeness principle. □

Theorem 1205.61 (Diagonal-limit representative cofinal-index budget lock)**.** *The finite-prefix budget rows and cofinal modulus rows of the cofinal-index source lock the selected window, readback, dyadic tolerance, and terminal-read tolerance before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is read. In particular, a terminal consumer may read* e *only after the representative rows, the finite-prefix budget rows* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$*, and the cofinal modulus rows* $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M, H_M, C$ *have all been placed on the same finite* BHist *surface.*

Proof. Begin with the cofinal-index source theorem Definition 1205.60. It places the finite equations and the selected representative rows on the same source route as the finite-prefix and cofinal modulus rows.

The bridge definition Definition 1205.56 then supplies the concrete budget lock. It identifies w with T_M, r with R_M, d with D_M, and e with E_M, while keeping the finite-prefix selector and exhaustion rows visible before terminal consumption.

The totality theorem Definition 1205.57 closes the tolerance route. Since every terminal seal read factors through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real rows, no consumer can read a terminal seal before those window, readback, dyadic, and tolerance coordinates are locked. □

Theorem 1205.62 (Diagonal-limit representative cofinal-index terminal readiness)**.** *The cofinal-index terminal readiness package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consists of the cofinal-index*

source Definition 1205.60, the cofinal-index budget lock Definition 1205.61, the root-index readiness package Definition 1205.55, the cofinal-terminal budget bridge Definition 1205.56, the cofinal-budget seal totality theorem Definition 1205.57, and the terminal consumer nonescape theorem Definition 1205.58. It is a finite readiness surface for Real-and-Completion consumers and carries no host equality, quotient stream equality, selected limit, hidden tail schedule, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or Lean bridge claim.*

Proof. The source theorem supplies the cofinal-index origin of the equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ across the named StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real rows.

The budget-lock theorem supplies the ordering constraint. It makes the selected window, readback, dyadic ledger, finite-prefix rows, and cofinal modulus rows prerequisites for terminal seal consumption.

It remains to package the terminal boundary. The root-index readiness package gives the finite representative surface, the cofinal-terminal bridge gives the shared BHist packet, the cofinal-budget seal totality theorem gives the terminal route, and the terminal consumer nonescape theorem excludes every off-packet equality, selector, choice, completion, and bridge coordinate. Therefore the readiness package is exactly the finite cofinal-index surface stated above. □

Definition 1205.63 (Diagonal-limit representative cofinal-tail extraction)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the cofinal-tail extraction surface is the finite packet*

$$T = (\mu, n, k, i, w, r, d, e, Q, B, W, D_F, R, S, E_F, H, C, P, N).$$

The first four rows record the root-index equations $k = \mu(1/(n+1))$ *and* $i = k + n + 1$ *from the root-readiness and cofinal-index readiness surfaces. The rows* w, r, d, e *are respectively the selected* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The rows* Q, B, W, D_F, R, S, E_F *are the finite-prefix budget rows inherited from* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, with the cofinal modulus source supplied by* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *through the cofinal-terminal budget bridge. The rows* H, C, P, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row for the same extraction.*

This packet is sourced by $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*,* $\mathsf{CofinalModulusSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *through Definition 1205.1, Definition 1205.2, Definition 1205.49, Definition 1205.62, and Definition 1205.56. It records only finite cofinal-tail data; it exports no quotient stream equality, host limit, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Theorem 1205.64 (Diagonal-limit representative cofinal-tail coverage)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *terminal route factors through the cofinal-tail extraction packet of Definition 1205.63. In particular, the consumer must read the selected root-index rows* μ, n, k, i*, the selected window* w*, readback* r*, dyadic ledger* d*, terminal seal* e*, finite-prefix budget rows* Q, B, W, D_F, R, S, E_F*, and support rows* H, C, P, N *as one finite* BHist *surface before it can consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the carrier and classifier of Definition 1205.1 and Definition 1205.2. They admit only the selected representative rows and their componentwise support rows, so every downstream read begins on a finite BHist surface rather than on a host stream or completed Real object.

Next use the root-readiness exactness theorem Definition 1205.49 and the cofinal-index readiness package Definition 1205.62. These results place μ, n, k, i, w, r, d, e on the same route as the

finite-prefix and cofinal modulus rows, with the equations for k and i visible before terminal consumption.

Finally attach the cofinal-terminal budget bridge Definition 1205.56 and the terminal-pullback obligation surface Definition 1205.93. The bridge supplies the finite-prefix and cofinal-tail budget rows, while the pullback surface forces terminal consumption back through the displayed StreamName, RegSeqRat, DyadicRatCore, and Real sources. Hence every Real-and-Completion consumer factors through the packet T and no off-packet quotient, host limit, selected limit, choice row, or completeness principle is available. □

Theorem 1205.65 (Diagonal-limit representative cofinal-tail classifier stability)**.** *Let two accepted cofinal-tail extraction packets for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *be classified together by the displayed componentwise* hsame *rows and* Cont *replay. Then the selected rows* μ, n, k, i, w, r, d, e*, the finite-prefix budget rows* Q, B, W, D_F, R, S, E_F*, and the support rows* H, C, P, N *are preserved componentwise. The preservation uses only the packet of Definition 1205.63; it does not add a quotient stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, selected limit, hidden tail schedule, countable choice row, or ambient completeness witness.*

Proof. Read the classifier of Definition 1205.2. It compares accepted carriers componentwise, and the only transport rows available to the extraction packet are the recorded hsame rows together with displayed Cont replay.

The cofinal-index readiness package fixes the rows that must be preserved. It attaches the root-index equations, the selected window and readback, the dyadic ledger, the finite-prefix budget rows, and the terminal seal to the same cofinal source route.

It remains to check that no stronger transport is introduced by the extraction. The cofinal-terminal budget bridge and the terminal-pullback obligation surface preserve the same displayed terminal rows, while the formal handoff boundary Definition 1205.69 permits later work to encode only the visible BHist, hsame, Cont, Pkg, NameCert, and sibling rows. Therefore classifier stability is exactly componentwise stability of the finite packet T. □

Theorem 1205.66 (Diagonal-limit representative cofinal-tail NameCert exactness)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *obligation for cofinal-tail extraction is exactly the package consisting of the extraction surface of Definition 1205.63, the coverage theorem Definition 1205.64, the classifier-stability theorem Definition 1205.65, the cofinal-terminal budget bridge, the terminal-pullback obligation surface, and the formal handoff boundary. Its ledger refuses quotient streams, host limits, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as extraction data.*

Proof. Coverage supplies the positive route. By Definition 1205.64, every Real-and-Completion consumer reaches the terminal seal through the finite extraction packet and through the named StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real rows.

Classifier stability supplies the transport rule. By Definition 1205.65, componentwise hsame and displayed Cont replay preserve exactly the extraction rows, so the local NameCert cannot acquire a second tail, a detached terminal seal, or a hidden selected limit.

The remaining ledger boundary is fixed by the existing surfaces. The cofinal-terminal budget bridge gives the finite cofinal budget route, the terminal-pullback obligation surface pulls terminal consumption back to its source rows, and the formal handoff boundary confines later formal targets to the displayed BEDC rows. Thus the NameCert exactness package is precisely the finite cofinal-tail extraction surface, with quotient streams, host limits, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness refused by the ledger. □

Theorem 1205.67 (Diagonal-limit representative Lean target surface)**.** *For an accepted shared-terminal* DiagonalLimitRepresentative$^{\uparrow}$ *carrier, the exact Lean target surface is the finite displayed route*

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N$$

together with the shared-terminal budget rows

$$Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F.$$

The intended target reads the representative carrier, the shared-terminal carrier, and the root-readiness package as one finite BHist *surface:* $X, \mu, n, k, i, w, r, d, e$ *are the representative rows,* H, C, P, N *are the local transport, replay, provenance, and naming rows, and* $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ *are the finite-prefix budget rows. No additional carrier, classifier, selector, equality principle, host limit, or ambient completion field is part of the target surface.*

Proof. Read the representative carrier of Definition 1205.1. It lists exactly the rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and refuses host limits, quotient stream equality, hidden selectors, countable choice, and ambient Real$^{\uparrow}$ completeness.

Attach the shared-terminal carrier of Definition 1205.32. It adds only the budget rows $Q, B, W, D_F, R, S, E_F, H_F$ and makes them part of the same terminal route.

Finally apply the root-readiness exactness theorem, Definition 1205.49. Coverage, soundness, and completeness there identify every downstream Real-and-Completion consumer route with this finite packet, so a later Lean target has precisely the displayed rows as its obligation surface. □

Theorem 1205.68 (Diagonal-limit representative audit-readiness package)**.** *The audit-ready obligation surface for* DiagonalLimitRepresentative$^{\uparrow}$ *is the package consisting of the carrier and classifier of Definition 1205.1 and Definition 1205.2, the shared-terminal route of Definition 1205.33, the root-readiness triad of Definition 1205.44, the dependency-readiness scope of Definition 1205.47, and the root-readiness exactness theorem of Definition 1205.49. This package is exact for audit: it covers the displayed representative rows, the shared-terminal budget rows, the accepted support rows, and no off-packet route.*

Proof. The carrier and classifier supply the accepted object and its componentwise comparison rule. These rows determine what can be admitted before any shared-terminal consumer is attached.

The shared-terminal route then attaches the finite-prefix budget rows and orders the terminal read through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, FinitePrefixLimitBudget$^{\uparrow}$, and Real$^{\uparrow}$. The root-readiness triad makes that ordered route mandatory before support rows are consumed.

The dependency-readiness scope packages the route for downstream L10 chapters, and root-readiness exactness gives coverage, soundness, and completeness for the displayed packet. Therefore the audit surface is neither weaker nor stronger than the carrier, shared-terminal route, root-readiness triad, and dependency-readiness exactness already displayed in the chapter. □

Theorem 1205.69 (Diagonal-limit representative formal handoff boundary)**.** *Any later Lean handoff, audit target, or standard bridge for* DiagonalLimitRepresentative$^{\uparrow}$ *must preserve the finite route of Definition 1205.67 and the audit-ready package of Definition 1205.68. Such a handoff may encode the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* FinitePrefixLimitBudget$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows, but it may not import quotient-stream equality, host* Real$^{\uparrow}$ *equality, selected limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeFormalHandoffBoundary`

Proof. The positive content is fixed by the Lean target surface. It names the representative rows, support rows, and finite-prefix budget rows that must be preserved by any formal consumer.

The audit-readiness package supplies the binding scope for those rows. It packages carrier admission, classifier transport, shared-terminal routing, root-readiness exposure, dependency-readiness, and exactness without adding any analytic completion principle.

It remains to state the boundary for later formal work. The nonchoice boundary, completion boundary, Real-seal nonchoice export, and bridge precondition package already exclude quotient equality, host equality, selected limits, choice, and ambient completeness from the chapter surface. A Lean handoff or bridge can therefore check only this finite BEDC route; importing any excluded object would change the obligation surface rather than formalize it. □

Theorem 1205.70 (Diagonal-limit representative terminal pullback coverage)**.** *Every accepted formal handoff for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *pulls downstream Real-and-Completion consumption back to the finite route*

$$\mu, n, k, i, w, r, d, Q, B, W, D_F, R, S, e, H, C, P, N, E_F, H_F, C_F, P_F, N_F.$$

Coverage is supplied by the root-readiness exactness package; soundness is supplied by the cited $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *surfaces. Thus the handoff can consume the displayed rows, their componentwise* BHist *packaging, and the associated* $\mathsf{NameCert}$ *obligations, but it cannot introduce quotient-stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Lean target surface of Definition 1205.67. It identifies the representative rows and the shared-terminal budget rows as one finite obligation surface, so every accepted formal handoff has a named row for each element of the displayed route.

The audit-readiness package of Definition 1205.68 then records which carrier, classifier, shared-terminal route, dependency-readiness scope, and exactness theorem can be consumed. The root-readiness exactness theorem, cited there as Definition 1205.49, supplies coverage for the terminal route and soundness for the support surfaces.

Finally apply the handoff boundary of Definition 1205.69. Its exclusions remove quotient-stream equality, host equality, selected limits, countable choice, and ambient completeness from the obligation surface. Route exhaustion leaves only the finite rows named above, which is the required pullback coverage contract for downstream Real-and-Completion consumption. □

Theorem 1205.71 (Diagonal-limit representative terminal pullback determinacy)**.** *Let two accepted Real-and-Completion consumers of the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *formal handoff agree componentwise, by the recorded* hsame *and* Cont *replay, on the representative rows*

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N$$

and on the shared-terminal budget rows

$$Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F.$$

Then the two consumers induce the same terminal $\mathsf{Real}^{\uparrow}$ *seal route and the same finite-prefix budget pullback. Quotient-stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, a selected limit, countable choice, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness row cannot distinguish the induced pullback.*

Proof. The displayed agreement is exactly agreement on the Lean target surface of Definition 1205.67. That theorem records the representative rows and shared-terminal budget rows as one finite BHist surface, so the two consumers begin with the same formal handoff data.

Next apply the audit-readiness package of Definition 1205.68. It identifies the carrier, classifier, shared-terminal route, dependency scope, and root-readiness exactness data consumed by any accepted handoff, so componentwise hsame and Cont replay transport the same package to both consumers.

Finally use terminal pullback coverage from Definition 1205.70. Coverage leaves no off-packet route through quotient streams, host equality, selected limits, countable choice, or ambient completeness. The only remaining Real-and-Completion read is the displayed terminal $\mathsf{Real}^{\uparrow}$ seal route with its finite-prefix budget pullback, hence the two accepted consumers determine the same pullback. □

Theorem 1205.72 (Diagonal-limit representative audit-clean threshold package)**.** *The audit-clean-ready package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is exactly the finite route through* BHist*, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *naming, the object faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, the shared-terminal budget rows*

$$Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F,$$

and the formal handoff theorems

$$\textit{Definition } 1205.67, \quad \textit{Definition } 1205.68, \quad \textit{Definition } 1205.70, \quad \textit{Definition } 1205.71.$$

It contains no quotient stream, selected limit, countable choice, host $\mathsf{Real}^{\uparrow}$ *equality, or ambient completeness row.*

Proof. Start with the Lean target surface and audit-readiness package. The first theorem fixes the representative rows and shared-terminal budget rows as a finite BHist surface, while the second identifies the carrier, classifier, shared-terminal route, dependency scope, and exactness theorem that the audit can consume.

Next apply terminal pullback coverage. It pulls every accepted formal handoff back to the displayed representative rows, the budget rows, and the named $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ support surfaces.

Finally use terminal pullback determinacy. Once two consumers agree by hsame and Cont replay on those rows, no quotient stream, selected limit, countable-choice row, host equality, or ambient completeness row remains available to separate their reads. Hence the audit-clean-ready package is precisely the finite route named in the statement. □

Theorem 1205.73 (Diagonal-limit representative phase-exit audit unblock)**.** *The audit-clean threshold package of Definition 1205.72 is the consumer-facing input by which* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *reads* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *at the Phase Real and Completion exit. The read factors through the handoff lock of Definition 1205.151 and the four-object threshold projection of Definition 1327.16: only the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *face records, with their displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support, can serve the exit criterion.*

Proof. The handoff lock identifies the $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ Phase Real and Completion packet consumed by $\mathsf{PhaseRealCompletionExit}^{\uparrow}$. It keeps the source rows at $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the support route carried by BHist, hsame, Cont, Pkg, and the local naming row.

The audit-clean threshold package supplies the exact finite handoff available to that lock. Its coverage and determinacy statements exclude every off-packet route through quotient streams, selected limits, countable choice, host Real equality, or ambient completeness.

Finally apply the four-object threshold projection. The projection is the roadmap-facing exit read of $\mathsf{PhaseRealCompletionExit}^{\uparrow}$, so the same four object faces and their support rows are the consumer-facing input that unblocks the Phase Real and Completion audit criterion. □

Theorem 1205.74 (Diagonal-limit representative Lean-readiness obligation boundary)**.** *Any later Lean target for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *may encode only the finite rows and sibling faces already named by this formal handoff surface: the representative rows, the shared-terminal budget rows,* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Stronger Real-completeness claims, quotient-stream equality, countable-choice principles, selected limits, host* $\mathsf{Real}^{\uparrow}$ *equality, or standard-bridge assertions remain outside this obligation boundary.*

Proof. The positive boundary is supplied by the Lean target surface, the audit-readiness package, and the audit-clean threshold package. Together they name every row that a formal target can read: representative data, budget data, object faces, and support rows.

The negative boundary is supplied by the formal handoff boundary, terminal pullback coverage, and terminal pullback determinacy. These results state that quotient streams, selected limits, countable choice, host equality, and ambient completion are not part of the accepted carrier or its terminal pullback.

It remains only to separate Lean readiness from stronger bridge claims. The handoff surface is a finite BEDC obligation surface for the displayed rows. A standard bridge or Real-completeness theorem would require additional bridge data beyond those rows, so it cannot be counted as part of this target. □

Theorem 1205.75 (Diagonal-limit representative hub dependency witness)**.** *The hub-visible dependency witness for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is the package consisting of the carrier and classifier of Definition 1205.1 and Definition 1205.2, the shared-terminal carrier of Definition 1205.32, the root-readiness triad of Definition 1205.44, the root-index readiness package of Definition 1205.55, the cofinal-terminal budget bridge of Definition 1205.56, and the formal handoff boundary of Definition 1205.69. Its visible route factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with the cofinal terminal budget rows supplying the completion-facing budget source. It exposes no quotient streams, host real equality, hidden selectors, countable choice, or ambient Real completeness.*

Proof. The carrier and classifier identify the accepted diagonal representative and its componentwise support rows. The shared-terminal carrier then attaches the finite-prefix budget rows to the same terminal read, so the hub package begins with one finite BHist surface rather than two unrelated surfaces.

The root-readiness triad supplies the mandatory display order for the modulus request, selected indices, selected window, readback, dyadic ledger, finite-prefix rows, and terminal $\mathsf{Real}^{\uparrow}$ seal. The root-index readiness package makes the equations for k and i visible before support rows are consumed, and it records that the named sibling certificates source each exposed row.

The cofinal-terminal budget bridge adds the completion-facing budget source without adding an analytic completion principle. Finally, the formal handoff boundary states that later Lean handoffs may encode only the displayed BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, and sibling rows. Hence the

hub witness packages the existing obligations as a visible dependency route while preserving the quotient, host-equality, selector, choice, and completeness exclusions. □

Theorem 1205.76 (Diagonal-limit representative root-unblock consumer surface)**.** *Every Real-and-Completion consumer that reads the hub-visible* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *dependency witness must consume the ordered surface*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

through the accepted carrier rows of Definition 1205.1, the shared-terminal rows of Definition 1205.32, the root-readiness route of Definition 1205.44, the root-index package of Definition 1205.55, and the cofinal-budget seal totality theorem Definition 1205.57. No accepted consumer of this surface may replace the displayed route by quotient stream equality, host $\mathsf{Real}^{\uparrow}$ *equality, a selected completed limit, a hidden selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and shared-terminal carrier. Together they display the representative rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and the finite-prefix rows $Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F$ as one terminal packet.

Apply the root-readiness triad and root-index readiness package. They force the selected-index equations, selected window, readback, dyadic ledger, finite-prefix budget rows, and terminal Real seal to be visible before any downstream support row can be used.

Now use the cofinal-budget seal totality theorem. It routes the completion-facing seal through the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real surfaces. The formal handoff boundary then removes any off-packet quotient, host-equality, selected-limit, hidden-selector, choice, or ambient-completeness entrance. Thus the consumer surface is exactly the ordered finite route stated above. □

Theorem 1205.77 (Diagonal-limit representative L10 exit readiness package)**.** *The L10 exit readiness package exported at the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *hub consists of the hub dependency witness of Definition 1205.75, the root-unblock consumer surface of Definition 1205.76, the terminal pullback coverage theorem Definition 1205.70, and the terminal consumer cofinal nonescape theorem Definition 1205.58. This package is ready for Real and Completion consumers precisely because every exit row is already routed through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with no quotient streams, host real equality, hidden selectors, countable choice, or ambient Real completeness.*

Proof. The hub dependency witness gathers the existing carrier, shared-terminal, root-readiness, root-index, cofinal-terminal, and formal-handoff obligations into one public package. The root-unblock consumer surface then states the exact order in which a Real-and-Completion consumer may read that package.

Terminal pullback coverage supplies the exit coverage: every accepted formal handoff pulls downstream consumption back to the finite representative and budget rows. Terminal consumer cofinal nonescape supplies the negative boundary: the cofinal-terminal budget bridge admits only the displayed representative, finite-prefix, cofinal modulus, and structural BEDC rows.

It remains to name the exported readiness. Since the positive route is the displayed finite path through StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, and Real surfaces, and since every quotient, host-equality, hidden-selector, choice, and ambient-completeness alternative is excluded by the cited boundaries, the hub exports a dependency-ready L10 exit package for the Real and Completion phases. □

Theorem 1205.78 (Diagonal-limit representative root carrier window)**.** *For every accepted* DiagonalLimitRepresent *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the root carrier window exposes, as one finite BHist *packet, the regular-Cauchy family X, the modulus row μ, the selected equations $k = \mu(1/(n+1))$ and $i = k+n+1$, the* $\mathsf{StreamName}^{\uparrow}$ *window w, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger d, the terminal* $\mathsf{Real}^{\uparrow}$ *seal e, the transport row H, the continuation row C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row N. The window contains no quotient stream equality, host limit, choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 1205.1. Its displayed tuple is exactly $X, \mu, n, k, i, w, r, d, e, H, C, P, N$, with the two selected-index equations and the four support rows already present in the finite packet.

Read the sibling route named in the chapter opening. The window coordinate is sourced by $\mathsf{StreamName}^{\uparrow}$, the readback coordinate by $\mathsf{RegSeqRat}^{\uparrow}$, the arithmetic ledger by $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal seal by $\mathsf{Real}^{\uparrow}$.

The carrier definition also states the negative boundary: no host real, quotient stream equality, hidden tail selector, countable choice row, or ambient completeness field is exported. Therefore the accepted root carrier window is precisely the finite BHist packet above. □

Theorem 1205.79 (Diagonal-limit representative classifier transport)**.** *If two accepted* DiagonalLimitRepresentative *carriers are related by the classifier of Definition 1205.2, then the selected-index equation,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row transport only through the displayed H and C rows. In particular, classifier transport preserves the root-readiness route of Definition 1205.44 and the root-index readiness package of Definition 1205.55 without adding an off-packet selector, quotient equality, host limit, choice row, or ambient completeness field.*

Proof. Start with the classifier definition. It relates accepted carriers only by componentwise recorded hsame comparisons and displayed Cont routes over the selected indices, window, readback, dyadic ledger, terminal seal, package row, and local naming row.

Apply the root-readiness triad. It requires every Real-and-Completion consumer to expose the modulus request, selected indices, window, readback, dyadic ledger, finite-prefix budget rows, and terminal seal before any support row is consumed.

Now apply the root-index readiness package. It makes the equations for k and i and the sibling sources for w, r, d, e part of the finite root-index surface. Since the classifier offers no support coordinate outside H and C, transport of those rows can only be the displayed componentwise transport and replay. □

Theorem 1205.80 (Diagonal-limit representative NameCert ledger exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRep}}$ *ledger for the root-unblock surface is exhausted by the carrier of Definition 1205.1, the classifier of Definition 1205.2, the root-readiness route of Definition 1205.44, the cofinal-terminal budget bridge and seal totality of Definition 1205.57, the formal handoff boundary of Definition 1205.69, and the hub dependency witness of Definition 1205.75. No further ledger row is available to a Real-and-Completion consumer.*

Proof. First read the carrier and classifier. They provide the accepted tuple, its componentwise comparison rule, and the local naming row N.

Next use the hub dependency witness. It packages the carrier, classifier, shared-terminal rows, root-readiness route, root-index package, cofinal-terminal bridge, and formal handoff boundary as the hub-visible dependency route.

Finally apply cofinal-budget seal totality and the formal handoff boundary. The seal totality theorem routes every terminal seal read through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real surfaces, while the handoff boundary prevents any stronger analytic object from entering. Hence the local NameCert ledger is exhausted by the listed rows and obligations. □

Theorem 1205.81 (Diagonal-limit representative index unary closure)**.** *For every accepted* DiagonalLimitRepresent *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the selected family row, modulus row, precision request, selected family index, selected readback index, selected $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row are* UnaryHistory *rows. Moreover, any derived route read produced by the displayed* Cont *closures*

$$\mathsf{Cont}(\mu, n, k), \qquad \mathsf{Cont}(w, r, i), \qquad \mathsf{Cont}(d, e, H)$$

remains a UnaryHistory *row. These are precisely the unary rows exposed by the carrier of Definition 1205.1 and the NameCert obligations of Definition 1205.3; no quotient stream equality, host limit, choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row can serve as a replacement index source.*

Proof. Project the carrier of Definition 1205.1. It displays $X, \mu, n, k, i, w, r, d, e$ and the local naming row N as finite BHist coordinates of the accepted representative packet.

Apply the NameCert obligations of Definition 1205.3. Their Lean touchpoint, DiagonalLimitRepresentativeName records the same carrier rows as UnaryHistory data and closes the derived reads by the three displayed Cont routes.

The sibling route supplies the sources of those rows: $\mathsf{StreamName}^{\uparrow}$ exposes w, $\mathsf{RegSeqRat}^{\uparrow}$ exposes r, $\mathsf{DyadicRatCore}^{\uparrow}$ exposes d, and $\mathsf{Real}^{\uparrow}$ exposes e. Since every derived read is obtained by continuation from already displayed unary rows, the root index surface remains a BHist obligation surface rather than an analytic completion claim. □

Theorem 1205.82 (Diagonal-limit representative continuation route closure)**.** *The accepted* Cont *routes for the root-unblock surface of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet are exactly the modulus-request route, the window-readback route, and the ledger-seal route:*

$$\mathsf{Cont}(\mu, n, k), \qquad \mathsf{Cont}(w, r, i), \qquad \mathsf{Cont}(d, e, H).$$

The first route reads the selected family index from the displayed modulus request, the second route reads the selected readback index from the $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the third route carries the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger into the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No accepted root consumer has a fourth continuation route through a hidden selector, quotient stream equality, host limit, choice row, or ambient completeness field.*

Proof. Read the carrier and classifier of Definition 1205.1 and Definition 1205.2. The carrier lists the continuation support row C, while the classifier permits replay only along the recorded componentwise Cont routes for the selected indices, window, readback, ledger, and seal.

Use Definition 1205.3. The obligation theorem exposes exactly the three continuation closures named in the statement: modulus with request, window with readback, and ledger with terminal seal.

Finally follow the sibling route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. That route accounts for every public read needed by a Real-and-Completion consumer, and the carrier contains no coordinate from which an additional accepted continuation could be sourced. □

Theorem 1205.83 (Diagonal-limit representative package and NameCert source closure)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, the local package row* P *and local naming row* N *source the representative packet through the finite* Pkg *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *surface of Definition 1205.1. The source surface is the displayed route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the carrier, classifier, and NameCert obligations of Definition 1205.3. It does not source quotient stream equality, a host limit, a hidden choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeCarrier pkg namecert source closure`

Proof. Start with the carrier. The rows P and N occur only as package provenance and local name for the displayed packet $D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$.

Apply the NameCert obligations. They bind the local Pkg signature and $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ row to the same unary rows and continuation closures used by the index and terminal routes.

The sibling route then identifies the packet source: $\mathsf{StreamName}^{\uparrow}$ supplies the selected window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the readback, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the ledger, and $\mathsf{Real}^{\uparrow}$ supplies the seal row. Because the source is exhausted by those finite rows plus P and N, the local package cannot add quotient, host-limit, choice, or ambient-completeness data. □

Theorem 1205.84 (Diagonal-limit representative terminal row coverage)**.** *Every terminal row consumed from an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet factors through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route. More precisely, a Real-and-Completion consumer may read the terminal seal only after the packet exposes the selected window* w, *readback* r, *dyadic ledger* d, *terminal seal* e, *support rows* H, C, P, N, *and the finite-prefix budget rows on the same finite* BHist *surface. The factorization is the one supplied by the root-readiness triad of Definition 1205.44, the NameCert ledger of Definition 1205.80, and the cofinal-budget seal totality theorem Definition 1205.57; it does not assert quotient stream equality, host real equality, selected limits, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the root-readiness triad. It places the modulus request, selected indices, selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, finite-prefix budget rows, terminal $\mathsf{Real}^{\uparrow}$ seal, and support rows on one displayed root surface before any consumer can use the support coordinates.

Next apply the NameCert ledger exhaustion theorem. Its local ledger identifies the carrier, classifier, root-readiness route, cofinal-terminal bridge, formal handoff boundary, and hub dependency witness as the only rows available to the consumer.

Finally use cofinal-budget seal totality. That theorem orders the terminal seal through the same StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real surfaces. Since no listed theorem exports a quotient, host-limit, selector, choice, or ambient-completeness row, every terminal row factors through exactly the displayed finite route. □

Theorem 1205.85 (Diagonal-limit representative cofinal selector transport)**.** *The cofinal selector data of an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet transports only inside the displayed rows*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}.$$

In particular, transport of the selected equations $k = \mu(1/(n+1))$ *and* $i = k + n + 1$*, the selected window* w*, readback* r*, dyadic ledger* d*, finite-prefix budget rows, cofinal modulus rows, and terminal seal* e *is mediated by the packet rows* H, C, P, N *and by the cofinal-terminal budget bridge. It never passes through a hidden selector, quotient stream equality, host real equality, countable choice, or ambient completion field.*

Proof. Start from the classifier transport theorem in the root obligation surface. It permits componentwise transport only through recorded hsame comparisons and displayed Cont replay over the accepted carrier rows.

The cofinal-index readiness package then attaches the selected equations, the finite-prefix budget rows, and the cofinal modulus rows to the same source route. Its budget-lock theorem makes the selected window, readback, dyadic ledger, tolerance rows, and terminal seal visible before terminal consumption.

It remains to identify the allowed containers for transport. The NameCert ledger supplies the local Pkg and NameCert rows, while the formal handoff boundary Definition 1205.69 prevents any stronger analytic object from entering the route. Therefore cofinal selector transport stays inside the displayed BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 1205.86 (Diagonal-limit representative Real-completion consumer exactness)**.** *The Real-and-Completion root-unblock surface for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is exactly the package consisting of terminal row coverage, cofinal selector transport, the root-readiness triad of Definition 1205.44, the NameCert ledger of Definition 1205.80, the cofinal-budget seal totality theorem of Definition 1205.57, and the formal handoff boundary of Definition 1205.69. Any accepted downstream consumer must factor through this package, and every row in the package is sourced by the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, Real,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data. The package carries no quotient equality, host limit, selected limit, countable choice, Real completeness, or Lean bridge claim.*

Proof. Coverage is supplied by Definition 1205.84. It forces terminal consumption through the visible StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, and Real route.

Transport is supplied by Definition 1205.85. It confines the selector equations and cofinal rows to the displayed BHist, hsame, Cont, Pkg, and NameCert containers.

The existing root-readiness triad, NameCert ledger, cofinal-budget seal totality theorem, and formal handoff boundary close the package. Together they state the finite rows available to a downstream consumer, the ledger that names those rows, the terminal ordering through the cofinal budget, and the boundary excluding stronger analytic data. Hence the consumer exactness surface is precisely the named finite package and nothing outside it. □

Theorem 1205.87 (Diagonal-limit representative root coverage)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier consumed by a Real-and-Completion route, the root-unblock surface covers exactly the displayed finite rows*

$$\mathsf{BHist}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{FinitePrefixLimitBudget}^{\uparrow},$$

Concretely, the consumer may read the selected family row X, modulus row μ, precision row n, selected indices $k = \mu(1/(n+1))$ and $i = k + n + 1$, StreamName window w, RegSeqRat readback r, DyadicRatCore ledger d, finite-prefix budget rows Q, B, W, D_F, R, S, E_F, cofinal modulus rows $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M$, terminal Real seal e, and support rows H, C, P, N only as coordinates of that finite BHist *route. It receives no host limit, quotient class, choice-selected subsequence, hidden selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with the root carrier window and NameCert ledger of Definition 1205.78 and Definition 1205.80. These results from the root NameCert obligation surface identify $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ as the carrier and support rows available to the root consumer.

Next attach the Real-completion root-unblock surface. The terminal row coverage and consumer exactness theorems, Definition 1205.84 and Definition 1205.86, force terminal consumption through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, Real, BHist, Cont, Pkg, and NameCert data.

Finally use finite selector exhaustion and terminal-diagonal coverage, Definition 1205.90 and Definition 1205.98. They place the selector equations, finite-prefix budget rows, cofinal modulus seal, and terminal representative on the same finite route. Since each cited surface excludes host limits, quotient classes, choice rows, hidden selectors, and ambient completeness data, the root coverage is exactly the displayed obligation package. □

Theorem 1205.88 (Diagonal-limit representative selector transport)**.** *Let two accepted* DiagonalLimitRepresentative *carriers be related by the componentwise* hsame *classifier and displayed* Cont *replay of the root NameCert obligation surface. Then transport preserves the chosen diagonal selector rows*

$$k = \mu(1/(n+1)), \qquad i = k + n + 1,$$

the selected StreamName window w, RegSeqRat readback r, DyadicRatCore ledger d, cofinal tail rows $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M$, finite-prefix budget rows Q, B, W, D_F, R, S, E_F, and terminal Real seal row e before the Real seal is consumed. The transported packet remains a finite BHist *obligation with* Cont, Pkg, *and* NameCert *coordinates; it does not create an off-packet selector, host equality, quotient equality, choice row, or Real-completeness witness.*

Proof. Start from classifier transport in the root NameCert obligation surface, Definition 1205.79. It permits comparison only through recorded componentwise hsame rows and displayed Cont replay over the accepted carrier rows.

The Real-completion root-unblock surface supplies the ordering constraint. By Definition 1205.85, the selected equations, finite-prefix budget rows, cofinal modulus rows, selected window, readback, dyadic ledger, and terminal seal all transport inside the displayed BHist, hsame, Cont, Pkg, and NameCert containers.

Now apply finite selector exhaustion and terminal-diagonal transport, Definition 1205.90 and Definition 1205.99. The first fixes the selector route before terminal consumption, and the second preserves the terminal diagonal packet componentwise. Therefore hsame transport preserves the selected diagonal route, cofinal tail row, and finite-prefix budget before the Real seal is consumed, without adding any stronger analytic object. □

Theorem 1205.89 (Diagonal-limit representative Real seal handoff)**.** *Every Real-completion consumer receiving an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *terminal seal receives only the terminal representative row*

$$(X, \mu, n, k, i, w, r, d, e),$$

the cofinal selector and modulus rows $B_M, M_M, W_M, T_M, D_M, R_M, L_M, E_M$, the finite-prefix budget rows Q, B, W, D_F, R, S, E_F, and the NameCert support rows H, C, P, N. The handoff is a finite root-unblock obligation over BHist, Cont, Pkg, *and* NameCert*; it carries no host limit, quotient class, choice-selected subsequence, hidden completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Root coverage supplies the positive source. By Definition 1205.87, every Real-and-Completion consumer reads the terminal representative, cofinal selector, modulus budget, finite-prefix rows, and support rows from the same finite BHist route.

Selector transport supplies stability before consumption. By Definition 1205.88, the selected diagonal rows and budget rows are preserved by the displayed hsame and Cont data, so the consumer cannot replace the terminal representative with a host-selected subsequence or quotient class.

It remains to identify the handoff boundary. The terminal-diagonal Real handoff theorem Definition 1205.100, together with Real-completion consumer exactness Definition 1205.86, confines the downstream route to the finite terminal representative, cofinal selector, modulus budget, and NameCert rows. Thus the Real seal handoff exports exactly those rows and no host limit, quotient class, choice-selected subsequence, hidden completion object, or ambient completeness principle. □

Theorem 1205.90 (Diagonal-limit representative finite selector exhaustion)**.** *For an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier, fix the visible rows X, μ, n, k, i on the root-readiness surface. If the carrier is accepted by the root NameCert obligations and is consumed by a Real-and-Completion route, then the selected* $\mathsf{StreamName}^{\uparrow}$ *window w and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback r are forced through the displayed equations*

$$k = \mu(1/(n+1))$$

and

$$i = k + n + 1.$$

The resulting finite selector packet is exactly the route through $X, \mu, n, k, i, w, r, H, C, P, N$; it does not contain a hidden modulus selector, quotient-stream equality, host stream equality, countable choice, selected completed limit, or ambient completion principle.

Proof. Start with the root-readiness triad of Definition 1205.44. It requires any downstream Real-and-Completion consumer to expose μ, n, k, i, w, r before the support rows are consumed, with H, C, P, N only transporting, replaying, sourcing, and naming that displayed surface.

The readback-index scope and selected-window determinacy theorems, Definition 1205.37 and Definition 1205.50, then identify the selected window and readback as the rows attached to the visible indices. The equations for k and i are not external parameters; they are the displayed computations carried by the accepted packet.

Finally apply budget-source determinacy, Definition 1205.48. It prevents a downstream consumer from replacing the selected rows by an off-packet terminal source. Hence fixed X, μ, n, k, i exhaust the finite selector route through the named StreamName and RegSeqRat rows, with no additional selector or equality principle available. □

Theorem 1205.91 (Diagonal-limit representative support route coverage)**.** *Every accepted support row of a shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is covered by the finite BEDC support surface*

$$\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{FinitePrefixLimitBudget}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

More explicitly, the support rows H, C, P, N cover only the recorded history, componentwise transport, continuation replay, provenance, local naming, selected window, regular-rational readback, dyadic ledger, finite-prefix budget, and terminal Real seal routes. They cannot cover a detached support primitive, hidden selector, quotient stream equality, host equality, selected completed limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read primitive-scope exhaustion, Definition 1205.7. It lists the available primitive rows as BHist, hsame, Cont, Pkg, NameCert, and the sibling rows supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Next apply support-scope nonescape, Definition 1205.8. It confines accepted transport and replay to H, C, P, N, so no detached support primitive can be inserted after the selector and terminal rows have been displayed.

The dependency-readiness scope of Definition 1205.47 and the public exactness package of Definition 1205.35 supply the coverage direction. They make the five sibling surfaces and the four support rows the whole consumer-facing route, while excluding quotient, host, selector, choice, and ambient-completeness alternatives. □

Theorem 1205.92 (Diagonal-limit representative terminal seal stability)**.** *Let A and A' be accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carriers with the same visible selector rows X, μ, n, k, i, w, r, d and the same finite-prefix budget rows. Then their terminal* $\mathsf{Real}^{\uparrow}$ *seal routes are stable up to the recorded* hsame *support and displayed* Cont *replay of H, C, P, N. This stability is exactly a finite NameCert seal: it cannot introduce host equality, quotient-stream equality, selected completed limits, hidden selectors, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with terminal-seal determinacy, Definition 1205.51. It says that the visible finite-prefix budget rows together with w, r, d determine the terminal Real seal source up to the recorded hsame rows and the displayed Cont replay.

Consumer-route exhaustion, Definition 1205.52, supplies the route boundary. Every accepted Real-and-Completion consumer factors through the ordered route containing the selector rows, dyadic ledger, finite-prefix budget rows, terminal seal, and support rows; no other support surface is available.

Use Real-seal nonchoice export and the formal handoff boundary, Definition 1205.42 and Definition 1205.69, for the negative clause. They preserve the displayed terminal row as finite NameCert data and prevent the seal from being strengthened into host equality, quotient equality, selected limits, hidden selectors, choice data, or ambient completeness. Therefore the two carriers have the same terminal seal route precisely up to the recorded support. □

Definition 1205.93 (Diagonal-limit representative terminal-pullback obligation surface)**.** *The terminal-pullback obligation surface for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is the finite packet*

$$\Pi = (W, R, D, E, H, C, P, N)$$

read from the accepted carrier and classifier of Definition 1205.1 and Definition 1205.2. Its four source rows are:

1. *W, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row supplied by chapter 126;*
2. *R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplied by chapter 301;*
3. *D, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supplied by chapter 291;*
4. *E, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row supplied by chapter 38.*

The remaining rows H, C, P, N are respectively the componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row for that same packet. The surface is terminal-pullback data: it records how the terminal Real seal is pulled back through the finite StreamName, RegSeqRat, and DyadicRatCore sources before any downstream consumer may use the seal, and it exports no quotient stream equality, host limit, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Theorem 1205.94 (Diagonal-limit representative terminal-pullback source coverage)**.** *Every accepted terminal-pullback packet* $\Pi = (W, R, D, E, H, C, P, N)$ *for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exposes the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before the terminal Real seal is consumed. The source route is the one named by chapter 126, chapter 301, chapter 291, and chapter 38, read through the carrier and classifier of Definition 1205.1 and Definition 1205.2.*

Proof. Project the accepted carrier. Its displayed tuple contains the selected window, readback, ledger, and seal rows as finite BHist coordinates, and the terminal-pullback surface of Definition 1205.93 merely renames those four rows as W, R, D, E.

Next read the sibling sources. The window row is sourced by $\mathsf{StreamName}^{\uparrow}$, the readback row by $\mathsf{RegSeqRat}^{\uparrow}$, the tolerance ledger by $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal seal by $\mathsf{Real}^{\uparrow}$, with the four chapter references in the statement fixing the source side of the pullback.

Finally apply the classifier boundary. Componentwise hsame transport and displayed Cont replay are the only ways to move the four source rows toward terminal consumption, while P and N only source and name the same packet. Hence every accepted consumer sees W, R, D, E before it can use the terminal Real seal. □

Theorem 1205.95 (Diagonal-limit representative terminal-pullback classifier stability)**.** *If two accepted terminal-pullback packets for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *are classified together by Definition 1205.2, then their selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and terminal* $\mathsf{Real}^{\uparrow}$ *seal classifiers are preserved by the displayed* hsame *and* Cont *rows. This preservation uses only the finite packet* $\Pi = (W, R, D, E, H, C, P, N)$ *of Definition 1205.93; it does not require quotient stream equality, host stream equality, selected limits, countable choice, or ambient Real completeness.*

Proof. Start with the classifier of Definition 1205.2. It relates accepted carriers componentwise, so the selected window, readback, dyadic ledger, terminal seal, provenance, and local name are compared only by the recorded hsame rows and replayed only by the displayed Cont routes.

Use terminal-pullback source coverage. By Definition 1205.94, the four rows W, R, D, E are already visible before the terminal seal is consumed, so classifier transport has concrete rows to preserve rather than an external stream quotient or host equality claim.

The support rows close the stability claim. The rows H, C, P, N are part of the finite packet and cannot introduce a second window, alternate readback, different tolerance ledger, or stronger Real seal classifier. Thus hsame transport preserves exactly the displayed terminal-pullback classifiers. □

Theorem 1205.96 (Diagonal-limit representative terminal-pullback ledger exactness)**.** *The terminal-pullback ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *hides only the finite representative choice already displayed by the carrier of Definition 1205.1: the selected window* W*, readback* R*, tolerance ledger* D*, terminal seal* E*, and support rows* H, C, P, N*. It does not export a host limit, quotient-stream equality, countable choice, selected completed limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle to a Real-and-Completion consumer.*

Proof. Read the terminal-pullback surface. Its packet has exactly eight rows, and the first four are sourced by the StreamName, RegSeqRat, DyadicRatCore, and Real sibling chapters named in Definition 1205.93.

Now compare with the carrier. The carrier of Definition 1205.1 already displays the finite representative choice by its selected window, readback, ledger, and seal coordinates, while the classifier of Definition 1205.2 permits only componentwise transport and continuation replay over those coordinates.

It remains to exclude stronger exports. The rows H, C, P, N transport, replay, source, and name the same finite packet; none of them is a host limit, a stream quotient, a choice functional, or an ambient completeness field. Therefore the ledger is exact: it hides only the finite representative selection and exposes no analytic completion principle beyond the terminal seal row. □

Definition 1205.97 (Diagonal-limit representative terminal-diagonal selector)**.** *The terminal-diagonal selector for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is the finite BEDC packet*

$$\Delta = (M, Q, X, \mu, n, k, i, w, r, d, e, B, S, H, C, P, N)$$

read from an accepted carrier and classifier of Definition 1205.1 and Definition 1205.2. The rows M, Q, B, S *are respectively the cofinal modulus seal, finite-prefix limit budget, displayed budget row, and selected finite prefix window; the rows* X, μ, n, k, i *are the cofinal index rows with*

$$k = \mu(1/(n+1))$$

and

$$i = k + n + 1.$$

The rows w, r, d, e *are the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal for that same index row. The support rows* H, C, P, N *are the displayed* BHist*, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *naming data attached to the selected terminal diagonal route.*

This packet is sourced by the root-index coverage surface, cofinal-tail extraction surface, finite selector exhaustion surface, terminal-pullback obligation surface, and Real-completion consumer exactness surface. It records only finite BEDC rows; it exports no host selection function, quotient stream equality, host real equality, countable choice, completed limit object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Theorem 1205.98 (Diagonal-limit representative terminal-diagonal coverage)**.** *Every accepted terminal-diagonal selector packet* Δ *for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exposes, for each displayed finite prefix window* S*, one terminal representative row*

$$(X, \mu, n, k, i, w, r, d, e)$$

with its cofinal modulus seal M*, finite-prefix budget rows* Q, B, S*, and support rows* H, C, P, N*. The exposed row is exactly the row routed by the root-index coverage theorem, cofinal-tail coverage theorem, finite selector exhaustion theorem, terminal-pullback source coverage theorem, and Real-completion consumer exactness theorem.*

Proof. First read the root-index coverage theorem Definition 1205.53. It makes the displayed equations for k and i visible before any terminal consumer reads the selected window, readback, dyadic ledger, or Real seal.

Next attach cofinal-tail coverage and finite selector exhaustion, Definition 1205.64 and Definition 1205.90. These two surfaces place the cofinal modulus seal, finite-prefix budget rows, selected finite prefix window, and fixed selector rows on the same finite BHist route.

Finally use terminal-pullback source coverage and Real-completion consumer exactness, Definition 1205.94 and Definition 1205.86. They force the terminal Real seal to be consumed only after the selected StreamName, RegSeqRat, DyadicRatCore, budget, cofinal modulus, provenance, and naming rows are displayed. Hence every displayed finite prefix window exposes the stated terminal representative row and no off-packet selector is available. □

Theorem 1205.99 (Diagonal-limit representative terminal-diagonal transport)**.** *Let Δ and Δ' be accepted terminal-diagonal selector packets for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *whose corresponding rows are related by the componentwise* hsame *rows and displayed* Cont *replay of Definition 1205.2. Then the selected terminal diagonal route*

$$(M, Q, X, \mu, n, k, i, w, r, d, e, B, S, H, C, P, N)$$

is preserved componentwise. In particular, transport cannot replace the selected terminal row by a second cofinal index row, a hidden diagonal selector, quotient stream equality, host $\mathsf{Real}^{\uparrow}$ *equality, countable choice, or ambient completeness data.*

Proof. Start from terminal-diagonal coverage. By Definition 1205.98, each accepted selector packet has concrete rows for the cofinal modulus seal, finite-prefix budget, selected index equations, terminal representative row, and support data.

Now apply the classifier stability already available on the cofinal-tail and terminal-pullback surfaces. The cofinal-tail classifier stability theorem preserves the selected cofinal rows and budget rows, while the terminal-pullback classifier stability theorem preserves the StreamName, RegSeqRat, DyadicRatCore, and Real rows through the displayed hsame and Cont data.

The support rows close the transport claim. Since H, C, P, N only record componentwise history, continuation replay, provenance, and local naming for the same finite packet, transport has no coordinate in which to insert an alternate terminal row or a stronger equality principle. Thus the selected terminal diagonal route is preserved exactly componentwise. □

Theorem 1205.100 (Diagonal-limit representative terminal-diagonal Real handoff)**.** *Every Real-and-Completion consumer of an accepted terminal-diagonal selector packet for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *receives only the selected terminal diagonal row, the finite-prefix budget rows, the cofinal modulus seal, and the visible continuation and provenance rows*

$$(X, \mu, n, k, i, w, r, d, e, Q, B, S, M, C, P, N).$$

The handoff factors through the terminal-diagonal selector of Definition 1205.97; it does not import host choice, quotient classes, Real completeness, an ambient diagonal selection principle, or any off-packet terminal representative.

Proof. Coverage supplies the positive handoff. By Definition 1205.98, the selected terminal row and its budget, cofinal modulus, continuation, provenance, and naming rows are already displayed before the Real-and-Completion consumer can use the terminal seal.

Transport supplies stability of the handoff. By Definition 1205.99, componentwise hsame and displayed Cont replay preserve the same selected row and cannot exchange it for another terminal representative.

It remains to identify the consumer boundary. Real-completion consumer exactness, Definition 1205.86, says that the downstream consumer factors through the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, Real, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data. Therefore the Real handoff receives exactly the finite rows in the statement and no host selection, quotient, completeness, or off-packet diagonal principle. □

Theorem 1205.101 (Diagonal-limit representative finite-tail selector uniqueness)**.** *Let a Real-and-Completion consumer read an accepted terminal-diagonal selector packet for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *through a displayed finite tail. Once the root-index rows* X, μ, n, k, i*, the cofinal-tail rows, the terminal-pullback rows, and the terminal-diagonal selector of Definition 1205.97 are fixed, the consumer has exactly one displayed selector route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{FinitePrefixLimitBudget}^{\uparrow} \longrightarrow \mathsf{CofinalModulusSeal}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the packet rows

$$(M, Q, X, \mu, n, k, i, w, r, d, e, B, S, H, C, P, N).$$

The uniqueness is finite-route uniqueness inside BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$*; it is not quotient stream equality, host real equality, a selected host limit, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a Lean bridge claim.*

Proof. Start with root-index coverage, Definition 1205.53. It fixes the displayed equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ before the consumer can read the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, or Real rows.

Next attach cofinal-tail coverage and terminal-pullback source coverage, Definition 1205.64 and Definition 1205.94. These two surfaces place the cofinal modulus seal, finite-prefix budget, selected window, readback, dyadic ledger, and terminal seal on the same finite tail.

Finally use terminal-diagonal coverage, Definition 1205.98. It packages the same rows as one terminal selector packet, while the support rows H, C, P, N only transport, replay, source, and name that packet. Hence a consumer with the fixed finite tail has exactly the displayed route and no coordinate from which to choose a second selector. □

Theorem 1205.102 (Diagonal-limit representative finite-tail selector nonescape)**.** *Every accepted Real-and-Completion consumer of a finite-tail selector route for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *remains inside the displayed rows supplied by* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*. The route may be transported only by the recorded componentwise* hsame *rows and displayed* Cont *replay of the same packet*

$$(M, Q, X, \mu, n, k, i, w, r, d, e, B, S, H, C, P, N).$$

It cannot escape to quotient stream equality, host stream equality, host real equality, a selected host limit, countable choice, ambient Real completeness, a hidden diagonal selector, or a detached terminal seal.

Proof. Read the cofinal-tail classifier stability theorem, Definition 1205.65. It confines the selected index rows, finite-prefix budget rows, cofinal modulus data, and support rows to componentwise transport.

Now apply terminal-pullback classifier stability, Definition 1205.95. The StreamName window, RegSeqRat readback, DyadicRatCore ledger, and Real seal are preserved only through the finite terminal-pullback packet.

The terminal-diagonal transport theorem Definition 1205.99 closes the boundary. Since transport has no coordinate for a quotient class, host equality, selected limit, choice functional, completeness witness, hidden selector, or detached seal, every accepted consumer remains on the displayed finite-tail selector route. □

Theorem 1205.103 (Diagonal-limit representative finite-tail selector obligation package)**.** *The finite-tail selector obligation package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consists exactly of root-index coverage, cofinal-tail coverage, terminal-pullback source coverage, terminal-diagonal coverage, finite-tail selector uniqueness, and finite-tail selector nonescape. A downstream Real-and-Completion consumer may cite this package as the complete finite-route obligation target for the selector route through*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{FinitePrefixLimitBudget}^{\uparrow},\quad \mathsf{CofinalModulusSeal}^{\uparrow},\quad \mathsf{Real}^{\uparrow},$$

together with BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. The package does not assert quotient stream equality, host Real equality, selected host limits, countable choice, ambient Real completeness, or any checked Lean target.*

Proof. The positive route is supplied by the cited coverage surfaces. Root-index coverage fixes the computed selector indices, cofinal-tail coverage joins those indices to the finite-prefix and cofinal modulus rows, terminal-pullback source coverage exposes the StreamName, RegSeqRat, DyadicRatCore, and Real rows, and terminal-diagonal coverage repackages them for Real-and-Completion consumption.

Uniqueness supplies the determinate selector claim. By Definition 1205.101, fixing the finite tail fixes one displayed route through the six named sibling surfaces and the local support rows.

Nonescape supplies the ledger boundary. By Definition 1205.102, the package has no outlet to quotient, host-equality, selected-limit, choice, completeness, hidden-selector, or Lean-bridge data. Therefore the listed theorems form exactly the finite-tail selector obligation package available to downstream consumers. □

Theorem 1205.104 (Diagonal-limit representative terminal cofinal extraction)**.** *For every accepted Real-and-Completion consumer of a shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet, the terminal row factors through one displayed cofinal extraction surface. The extraction first exposes the root-index rows* X, μ, n, k, i*, then the selected* $\mathsf{StreamName}^{\uparrow}$ *window* w*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, finite-prefix budget rows, cofinal modulus rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* e*. Its support is exactly the finite* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *route already named by the cofinal-terminal budget bridge, so the consumer reads one finite extraction package rather than separate local witnesses.*

Proof. Begin with the root-index readiness package, Definition 1205.55. It fixes the equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ before the selected StreamName, RegSeqRat, DyadicRatCore, finite-prefix, cofinal, and Real rows can be consumed.

Next apply terminal row coverage and cofinal selector transport, Definition 1205.84 and Definition 1205.85. Coverage supplies the ordered terminal path through the named sibling certificates, while transport confines the selector equations and cofinal rows to the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ containers.

Finally use the finite-tail selector obligation package, Definition 1205.103. It states that the visible finite route is the whole selector obligation available to a downstream consumer. Therefore the terminal row has one cofinal extraction surface containing the root-index, finite-prefix budget, StreamName, RegSeqRat, DyadicRatCore, cofinal modulus, Real seal, and local support rows. □

Theorem 1205.105 (Diagonal-limit representative Real-seal extraction)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal exported by the terminal cofinal extraction theorem is only the finite seal row e already routed by the cofinal-terminal budget bridge of Definition 1205.56. A consumer may cite the seal after the displayed cofinal index, finite-prefix budget, StreamName, RegSeqRat, DyadicRatCore, cofinal modulus, and local support rows have been exposed, but the citation does not introduce host real equality, quotient stream equality, a selected host limit, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completion principle.*

Proof. Project the cofinal-terminal budget bridge. Its definition identifies the representative window w with the cofinal window row T_M, the readback r with R_M, the dyadic ledger d with D_M, and the terminal seal e with E_M.

Now apply cofinal-budget seal totality, Definition 1205.57. The seal is read only after the route

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{CofinalModulusSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

has been displayed on the finite packet.

It remains to rule out a stronger export. Use terminal consumer cofinal nonescape, Definition 1205.58. Since the consumer has no coordinate for quotient equality, host equality, selected limits, choice, ambient completeness, or a detached seal, the exported Real seal is exactly the finite row already routed by the bridge. □

Theorem 1205.106 (Diagonal-limit representative L10 completion root package)**.** *The L10 completion root package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consists exactly of terminal cofinal extraction, Real-seal extraction, the root-readiness triad of Definition 1205.44, the NameCert ledger of Definition 1205.80, the formal handoff boundary of Definition 1205.69, and the Real-completion consumer exactness theorem of Definition 1205.86. This package is the citable finite completion-frontier handoff from the diagonal representative to downstream Real and Completion consumers, and it carries no host limit, quotient equality, countable choice, ambient Real completeness, or Lean bridge claim.*

Proof. The positive extraction surface is supplied by Definition 1205.104. It gathers the root-index, selected window, readback, dyadic ledger, finite-prefix budget, cofinal modulus, Real seal, and local BEDC support rows into one terminal cofinal package.

The exported seal boundary is supplied by Definition 1205.105. It identifies the Real seal as the finite bridge row already routed through the cofinal-terminal budget surface, not a stronger analytic object.

The remaining cited theorems close the root package. The root-readiness triad exposes the root route, the NameCert ledger names the available obligations, the formal handoff boundary prevents stronger data from entering the formal interface, and Real-completion consumer exactness states that downstream consumers factor through the named finite package. Hence the listed theorems form the terminal L10 completion root package available for citation. □

Definition 1205.107 (Diagonal-limit representative completion-selector ledger)**.** *The completion-selector ledger for an accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier is the finite* BHist *packet*

$$L = (X, \mu, n, k, i, w, r, d, e, H, C, P, N; Q, B, W, D_F, R, S, E_F, H_F, C_F, P_F, N_F; B_M, M_M, W_M, T_M, D_M, R_M, L_M, E$$

The first block is the displayed representative row of Definition 1205.1; the second block is the finite-prefix and terminal-budget row of Definition 1205.32; and the third block is the cofinal-modulus row

already named by the cofinal-terminal budget bridge of Definition 1205.56. The ledger records only the selector equations, the selected $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, finite-prefix budget rows, cofinal modulus rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. It exports no host limit, quotient stream equality, selected completion point, countable choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Theorem 1205.108 (Diagonal-limit representative completion-selector existence)**.** *Every accepted shared-terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier whose terminal cofinal extraction is accepted admits the ledger of Definition 1205.107. The ledger is obtained by projecting the accepted representative carrier, the shared-terminal finite-prefix package, and the cofinal-terminal budget bridge; it is not obtained by choosing a completion point outside the displayed packet.*

Proof. Project the representative carrier. It supplies the finite row $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and records the selector equations $k = \mu(1/(n+1))$ and $i = k + n + 1$ as carrier data.

Attach the shared-terminal carrier and the cofinal-terminal budget bridge. These two surfaces provide the finite-prefix rows and the cofinal modulus rows already used by terminal cofinal extraction, with w, r, d, e identified with the downstream StreamName, RegSeqRat, DyadicRatCore, and Real seal rows.

Finally apply Definition 1205.104. It states that a Real-and-Completion consumer reads one finite extraction package rather than separate witnesses. Repack those displayed rows as L; no quotient equality, host limit, selected completion point, choice row, or ambient completeness field is introduced. □

Theorem 1205.109 (Diagonal-limit representative completion-selector stability)**.** *Let two accepted completion-selector ledgers for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *agree componentwise by the recorded* hsame *rows and by displayed* Cont *replay on their representative rows, finite-prefix budget rows, cofinal modulus rows, provenance rows, and local* $\mathsf{NameCert}$ *rows. Then they determine the same selector handoff route through*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{FinitePrefixLimitBudget}^{\uparrow} \to \mathsf{CofinalModulusSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

The stability is stability of the displayed ledger, not equality of quotient streams, host Real points, chosen completion points, or ambient completion witnesses.

Proof. Begin with cofinal selector transport, Definition 1205.85. It confines the selected equations, selected window, readback, dyadic ledger, finite-prefix budget rows, cofinal modulus rows, and terminal seal to the recorded BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ containers.

Next use terminal pullback determinacy, Definition 1205.71. Componentwise agreement on the representative rows and shared-terminal budget rows determines the same terminal $\mathsf{Real}^{\uparrow}$ seal route and finite-prefix budget pullback.

It remains to include the cofinal modulus block. The cofinal-terminal route determinacy theorem identifies the cofinal modulus rows and the terminal seal as part of the same finite bridge. Therefore componentwise hsame and displayed Cont replay preserve the completion-selector handoff route exactly as the displayed ledger route. □

Theorem 1205.110 (Diagonal-limit representative completion handoff exhaustion)**.** *Every Real-and-Completion consumer of an accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *factors through the six displayed sibling surfaces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{FinitePrefixLimitBudget}^{\uparrow}, \quad \mathsf{CofinalModulusSeal}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

together with the finite BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support of the ledger. The handoff exhausts the L10 root obligations available to such consumers and claims no host limit, quotient stream equality, selected completion point, countable choice, or ambient Real completeness.*

Proof. The positive route is supplied by the L10 completion root package of Definition 1205.106. That package gathers terminal cofinal extraction, Real-seal extraction, root-readiness, the local NameCert ledger, the formal handoff boundary, and Real-completion consumer exactness.

Existence of the completion-selector ledger repackages exactly those rows as the finite packet L. Stability shows that accepted hsame transport and displayed Cont replay preserve the same route through the six sibling surfaces.

For exhaustion, apply Real-completion consumer exactness and finite-tail selector obligation packaging, Definition 1205.86 and Definition 1205.103. They leave every downstream consumer inside the StreamName, RegSeqRat, DyadicRatCore, finite-prefix budget, cofinal modulus, Real, and local support rows. The excluded host, quotient, selected-point, choice, and ambient-completeness data have no coordinate in L, so the displayed handoff is the whole consumer surface. □

Theorem 1205.111 (Diagonal-limit representative terminal-root admission)**.** *Let* $D = (X, \mu, n, k, i, w, r, d, e, H, C,$ *be an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier. Its terminal root is admitted only through the displayed route consisting of the* $\mathsf{StreamName}^{\uparrow}$ *window* w*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* e*, and the support rows* H, C, P, N*. The admission also requires the selected-index equations* $k = \mu(1/(n+1))$ *and* $i = k + n + 1$ *to be exposed on the same finite* BHist *route before any downstream Real-and-Completion consumer may read the terminal row.*

Proof. Apply root-readiness exactness, Definition 1205.49. It gives coverage, soundness, and completeness for the displayed route through the selected indices, window, readback, dyadic ledger, terminal seal, and local support rows.

Next use terminal cofinal extraction, Definition 1205.104. It extracts the terminal package only after the root-index, finite-prefix budget, cofinal modulus, and Real seal rows have appeared on the same finite surface.

Finally apply Real-completion consumer exactness, Definition 1205.86. It confines every accepted consumer to the named StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ rows, so no terminal root is admitted off the displayed route. □

Theorem 1205.112 (Diagonal-limit representative selected-tail coverage)**.** *For an accepted terminal-root package of* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$*, the selected equations* $k = \mu(1/(n+1))$ *and* $i = k + n + 1$*, the window/readback/ledger/seal rows* w, r, d, e*, and the finite support rows* H, C, P, N *exhaust every selected-tail read available to a downstream Real-and-Completion consumer. Any selected-tail read must therefore factor through the finite route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

together with the displayed local support.

Proof. Start with root-index readiness, Definition 1205.55. It makes the equations for k and i part of the visible root-index surface before the selected window, readback, ledger, and seal rows can be consumed.

Attach cofinal-tail coverage, Definition 1205.64. This places the selected tail on the same finite-prefix and cofinal-modulus route as the terminal seal, so a selected-tail read cannot detach from the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Use finite-tail selector uniqueness, Definition 1205.101, for the last coordinate check. Once the selected equations and support rows agree, the accepted selected-tail route is unique up to the recorded hsame rows and displayed Cont replay, so the listed rows exhaust the selected-tail surface. □

Theorem 1205.113 (Diagonal-limit representative completion-consumer nonescape)**.** *Downstream Real-and-Completion consumers may cite the terminal-root package of* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *as a finite BEDC source surface. Such consumers may use the displayed StreamName window, RegSeqRat readback, DyadicRatCore ledger, Real seal, selected-index equations, and local support rows, but they may not extract quotient stream equality, a host limit, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from that citation.*

Proof. Apply the formal handoff boundary, Definition 1205.69. It records that the formal surface contains the carrier rows, shared-terminal rows, root-readiness rows, and local support, and no additional selector, equality principle, host limit, or ambient completion field.

Then apply completion handoff exhaustion, Definition 1205.110. Every Real-and-Completion consumer factors through the displayed sibling surfaces and local support rows, so all permitted reads are already named by the terminal-root package.

The excluded objects have no coordinate in that finite package. Quotient stream equality, host limits, selected limits, choice rows, and ambient Real completeness therefore cannot be extracted by a downstream consumer that cites only the terminal-root package. □

Theorem 1205.114 (Diagonal-limit representative four-object source exhaustion)**.** *Every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exposes exactly the four source rows needed by a Phase Real and Completion consumer: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. These rows are read through the finite* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support already displayed by the ledger, with the sibling chapters chapter 291, chapter 126, chapter 301, and chapter 38 supplying the cited source surfaces.*

Proof. Start with the StreamName–RegSeqRat handoff of Definition 1205.16. It orders every accepted route to the seal through the selected finite window w, the selected regular-rational readback r, the downstream dyadic ledger d, and the terminal Real seal e.

Now use the L10 completion root package of Definition 1205.106. That package gathers the root-index, terminal cofinal extraction, Real-seal extraction, NameCert ledger, formal handoff boundary, and Real-completion consumer exactness rows into one finite completion-frontier handoff.

Finally apply completion handoff exhaustion, Definition 1205.110. It states that every Real-and-Completion consumer remains inside the displayed StreamName, RegSeqRat, DyadicRatCore, finite-prefix, cofinal-modulus, Real, and local support rows. Projecting the four requested source coordinates gives the listed DyadicRatCore radius, StreamName finite-window, RegSeqRat regularity, and Real seal rows. □

Theorem 1205.115 (Diagonal-limit representative Phase Real and Completion exit certificate)**.** *For every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$*, the four rows named in Definition 1205.114 form the Phase Real and Completion exit packet. The packet is a finite*

BEDC source certificate over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$*; it carries no quotient stream equality, no countable or dependent choice principle, no ambient completeness theorem, no trichotomy claim, and no open-cover compactness claim.*

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativePhaseRealCompletionExitCe`

Proof. The four-object exhaustion theorem supplies the source coordinates and ties them to the four sibling chapters. Thus the packet has named BEDC sources for radius accounting, finite-window exposure, regularity readback, and terminal Real sealing.

The completion-selector ledger of Definition 1205.107 records the same rows inside one finite BHist tuple with hsame, Cont, Pkg, and local NameCert support. It also states that host limits, quotient stream equality, selected completion points, choice rows, and ambient completeness fields are not ledger coordinates.

Use completion handoff exhaustion to close the certificate boundary. Since every consumer factors through the displayed finite rows, stronger analytic principles such as trichotomy or open-cover compactness cannot enter through this exit packet. □

Theorem 1205.116 (Diagonal-limit representative audit-clean consumer handoff)**.** *Downstream Real-completion consumers may cite the Phase Real and Completion exit packet of Definition 1205.115 as the finite BEDC source surface for* DiagonalLimitRepresentative$^{\uparrow}$*. Such a citation commits only to the displayed DyadicRatCore, StreamName, RegSeqRat, Real, and local support rows, and not to any off-packet completion principle or hidden selector.*

Proof. Begin with the Phase Real and Completion exit certificate. It identifies the four named source rows as the whole exit packet and excludes quotient, choice, ambient completeness, trichotomy, and open-cover compactness claims.

Then apply the completion handoff exhaustion theorem. Any Real-and-Completion consumer of the accepted ledger factors through the displayed sibling surfaces and the finite local support rows, so citing the exit packet is sufficient for downstream source accounting.

The L10 completion root package supplies the public citation boundary. It gathers the relevant terminal extraction, Real-seal extraction, root-readiness, NameCert, formal handoff, and consumer exactness results without exporting a stronger completion object. Therefore the handoff is audit-clean as a finite BEDC source surface. □

Theorem 1205.117 (Diagonal-limit representative exact-boundary consumer pullback)**.** *Every downstream Real-completion consumer of an accepted* DiagonalLimitRepresentative$^{\uparrow}$ *completion-selector ledger that supplies a same-route* RealCompletionExactBoundaryUp *face factors through the same four rows named in Definition 1205.114, and then through the* PhaseRealCompletionExit$^{\uparrow}$ *public gate. The pullback exports no quotient stream equality, selected limit, countable choice, host* Real$^{\uparrow}$ *equality, ambient completeness, metric completeness, trichotomy, open-cover compactness, or bridge-checked conclusion.*

Proof. First use Definition 1205.114. It identifies the only source rows exposed by the accepted completion-selector ledger as the DyadicRatCore$^{\uparrow}$ radius row, StreamName$^{\uparrow}$ finite-window row, RegSeqRat$^{\uparrow}$ regularity readback row, and terminal Real$^{\uparrow}$ seal row.

Next apply Definition 126.504. A downstream Real-completion read on those four faces factors through the terminal finite pullback determined by the same rows and the displayed BHist, hsame, Cont, Pkg, and local NameCert support.

The supplied RealCompletionExactBoundaryUp face is consumed only on that route. By Definition 1120.20, it remains a scoped exact-boundary witness and contributes no separate quotient,

selector, host-equality, completeness, metric-completeness, choice, trichotomy, compactness, or bridge row.

Finally use the diagonal phase-exit certificate, Definition 1205.115, together with the public gate certificate Definition 1327.18. These two certificates repack the same four-row route and the same-route exact-boundary face as the $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ public gate, so every such consumer factors through that gate without exporting any stronger completion principle. □

Theorem 1205.118 (Diagonal-limit representative phase-source coverage)**.** *Every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exposes the Phase Real source surface through exactly the four named source rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The $\mathsf{DyadicRatCore}^{\uparrow}$ *row records the tolerance and rational-arithmetic ledger, the* $\mathsf{StreamName}^{\uparrow}$ *row records the selected finite observation window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row records the selected regular-rational readback, and the* $\mathsf{Real}^{\uparrow}$ *row records the terminal seal reached by that readback. The ledger's* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows only transport, replay, source, and name these four rows; they do not expose a fifth phase source, a quotient-stream equality, a selected completion point, a countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the completion-selector ledger of Definition 1205.107. Its displayed coordinates include the selected window w, readback r, dyadic ledger d, terminal seal e, and the support rows H, C, P, N.

Read those coordinates through the four sibling surfaces. The window w is a $\mathsf{StreamName}^{\uparrow}$ row, the readback r is a $\mathsf{RegSeqRat}^{\uparrow}$ row, the ledger d is a $\mathsf{DyadicRatCore}^{\uparrow}$ row, and the seal e is a $\mathsf{Real}^{\uparrow}$ row.

For exhaustion, apply the four-object source exhaustion theorem and the Phase Real and Completion exit certificate, Definition 1205.114 and Definition 1205.115. They identify the same four rows as the complete exit packet and exclude quotient, choice, selected-point, and ambient-completeness coordinates. Therefore the finite diagonal representative exposes exactly the four phase sources listed above. □

Theorem 1205.119 (Diagonal-limit representative tail-selector compatibility)**.** *Let L and L' be accepted completion-selector ledgers for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *whose selected diagonal routes are classified together by their recorded* hsame *rows and displayed* Cont *replay. Then the selected tail data remain compatible with the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *classifiers: the selected finite windows stay on the same* $\mathsf{StreamName}^{\uparrow}$ *route, the selected regular-rational readbacks stay on the same* $\mathsf{RegSeqRat}^{\uparrow}$ *route, and no hidden tail selector may replace either route while preserving the accepted completion-selector ledger.*

Proof. Begin with completion-selector stability, Definition 1205.109. It states that componentwise agreement and displayed replay determine the same selector handoff route through the StreamName, RegSeqRat, DyadicRatCore, finite-prefix, cofinal-modulus, and Real surfaces.

Project the two tail coordinates from that route. The $\mathsf{StreamName}^{\uparrow}$ classifier sees only the selected finite-window rows, while the $\mathsf{RegSeqRat}^{\uparrow}$ classifier sees only the selected readback rows reached from those windows. Componentwise hsame transport and displayed Cont replay preserve those rows as ledger data rather than recomputing them by an external selector.

Finally use the support-scope nonescape theorem, Definition 1205.8. It excludes hidden selectors, quotient streams, detached windows, alternate dyadic ledgers, and off-packet terminal seals

from the accepted transport surface. Hence tail selection is compatible with the two classifiers exactly because both tail rows remain visible inside the same finite ledger. □

Theorem 1205.120 (Diagonal-limit representative Real handoff exactness)**.** *For every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$*, the* $\mathsf{Real}^{\uparrow}$ *handoff is exactly the terminal seal row reached after the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger. A downstream Real-completion consumer may cite this handoff as a finite source boundary, but the citation does not assert ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a selected host limit, quotient-stream equality, or global uniqueness of completed limits.*

Proof. First read the terminal route in the carrier. The Real-seal factorization theorem, Definition 1205.10, places every $\mathsf{Real}^{\uparrow}$ consumer read after the selected modulus index, selected window, selected readback, and dyadic ledger.

Next pass from the carrier to the completion-selector ledger. Completion-selector existence repackages the representative row, finite-prefix rows, cofinal-modulus rows, and terminal seal row into one finite packet, while completion handoff exhaustion keeps every Real-and-Completion consumer inside the displayed sibling surfaces.

The exact handoff is therefore the seal coordinate already present in the ledger. The Phase Real and Completion exit certificate names that coordinate as a finite BEDC source row and explicitly excludes quotient, choice, selected-point, and ambient-completeness claims. Thus the handoff is exact as a source boundary without becoming a Real-completeness principle. □

Theorem 1205.121 (Diagonal-limit representative phase-source projection totality)**.** *Every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *has a total visible-source projection onto exactly the four Phase Real rows*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The projection sends the accepted ledger to its dyadic arithmetic ledger, selected finite window, selected regular-rational readback, and terminal Real seal, and every displayed BHist, hsame, Cont, Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row only transports, replays, sources, or names those four rows. No fifth source row, quotient-stream equality, host limit, selected completed point, countable-choice row, or ambient Real-completeness field is part of the projection.*

Proof. Start with phase-source coverage, Definition 1205.118. It already identifies the visible source surface of an accepted completion-selector ledger with the dyadic ledger, selected window, selected readback, and terminal Real seal, with the local support rows serving only as transport and naming data.

Next apply four-object source exhaustion, Definition 1205.114. It rules out additional Phase Real source coordinates and pins the exit packet to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally read the completion-selector ledger definition, Definition 1205.107. Its coordinates supply each of the four rows named above, so the projection is total on accepted ledgers; because the definition has no coordinate for quotient equality, host limits, selected completed points, choice, or ambient completeness, the projection is exhausted by those four rows. □

Theorem 1205.122 (Diagonal-limit representative selector-source nonescape)**.** *For every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$*, the selected windows, regular-rational readbacks, and terminal seals cannot be replaced by hidden selectors, quotient-stream equality, host*

limits, selected completed points, countable choice, or ambient Real completeness while preserving the same ledger. Any replacement that still preserves the ledger must factor through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows already present in the ledger.*

Proof. First use tail-selector compatibility, Definition 1205.119. It keeps the selected finite windows on the displayed $\mathsf{StreamName}^{\uparrow}$ route and the selected regular-rational readbacks on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ route under the recorded hsame rows and Cont replay.

Next use support-scope nonescape, Definition 1205.8. That theorem confines accepted transport and replay to the local BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, so a detached selector or quotient-stream replacement has no support coordinate.

Finally apply completion-consumer nonescape, Definition 1205.113. Downstream Real-and-Completion reads remain inside the terminal-root package, so host limits, selected completed points, countable choice, and ambient Real completeness cannot preserve the same accepted ledger as replacement sources. □

Theorem 1205.123 (Diagonal-limit representative completion-frontier readback coverage)**.** *Every downstream Real-and-Completion consumer reads* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *only through the finite completion-frontier route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. The readback surface is exhausted by this route and does not add quotient-stream equality, host Real equality, a selected completed point, countable choice, ambient Real completeness, or a second completion selector.*

Proof. Begin with Real handoff exactness, Definition 1205.120. It identifies the Real-facing endpoint with the terminal seal reached after the selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

Then compose this endpoint with the L10 completion root package, Definition 1205.106. The package gathers terminal cofinal extraction, Real-seal extraction, root-readiness, the local NameCert ledger, the formal handoff boundary, and Real-completion consumer exactness as the finite completion-frontier citation surface.

Finally apply completion handoff exhaustion, Definition 1205.110. It keeps every downstream consumer inside the displayed sibling surfaces and local support rows, so the only possible readback route is the finite frontier stated above, and the excluded quotient, host, selected-point, choice, completeness, and second-selector data have no remaining coordinate. □

Theorem 1205.124 (Diagonal-limit representative root-window extraction)**.** *Every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *determines the finite five-row root-window package*

$$\Omega = (W, R, D, M, E)$$

where W *is the selected* $\mathsf{StreamName}^{\uparrow}$ *window supplied by chapter 126,* R *is the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *readback supplied by chapter 301,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger supplied by chapter 291,* M *is the* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *budget row supplied by chapter 1198, and* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal supplied by chapter 38. The package is extracted from the displayed carrier, cofinal bridge, and completion-selector rows; it carries no host limit, quotient stream equality, hidden tail selector, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. Project the completion-selector ledger of Definition 1205.107. Its first block exposes the selected window w, readback r, dyadic ledger d, and terminal seal e, while its cofinal block exposes the modulus-seal budget rows.

Use the sibling identifications already recorded by the ledger. The selected window is sourced by $\mathsf{StreamName}^{\uparrow}$, the readback by $\mathsf{RegSeqRat}^{\uparrow}$, the tolerance ledger by $\mathsf{DyadicRatCore}^{\uparrow}$, the cofinal budget by $\mathsf{CofinalModulusSeal}^{\uparrow}$, and the terminal seal by $\mathsf{Real}^{\uparrow}$ through the five chapter references in the statement.

Finally apply completion handoff exhaustion, Definition 1205.110. It keeps every Real-and-Completion consumer inside the displayed finite rows, so repacking them as Ω introduces no off-packet limit, equality, selector, choice, or completeness datum. □

Theorem 1205.125 (Diagonal-limit representative regularity handoff)**.** *For every root-window package $\Omega = (W, R, D, M, E)$ extracted by Definition 1205.124, the regularity handoff to a downstream completion consumer factors first through the* $\mathsf{StreamName}^{\uparrow}$ *window W of chapter 126 and then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R of chapter 301. The* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *budget M, and* $\mathsf{Real}^{\uparrow}$ *seal E may only be consumed after that window-readback route is displayed.*

Proof. Begin with the root-window extraction theorem. It presents all five rows as one finite packet, so the downstream consumer reads a displayed root window rather than a detached completion request.

Now apply the StreamName–RegSeqRat handoff, Definition 1205.16. That theorem orders the selected window before the selected regular-rational readback inside the representative carrier.

The remaining rows are downstream of the same route. The dyadic ledger records the readback tolerance, the cofinal budget records the admissible tail budget, and the Real seal is terminal. Hence none of D, M, E can be consumed before the $W \to R$ regularity route has been displayed. □

Theorem 1205.126 (Diagonal-limit representative Real-seal admissibility)**.** *For every extracted root-window package $\Omega = (W, R, D, M, E)$, the terminal* $\mathsf{Real}^{\uparrow}$ *row E is admissible for Real-and-Completion consumers exactly as the finite seal reached after the* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, and* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *budget M. This admissibility cites chapter 38 only as a displayed seal surface; it does not cite quotient equality, host Real equality, a chosen completed point, or ambient Real completeness.*

Proof. Read the positive seal route from Real handoff exactness, Definition 1205.120. It identifies the Real-facing endpoint with the terminal seal row reached after the selected window, readback, and dyadic ledger.

Add the cofinal budget row from the extracted package. The budget is already part of the completion-selector ledger and is transported through the cofinal-terminal bridge, so it constrains the terminal handoff without creating a new Real object.

The admissibility claim is therefore only the displayed seal claim. The root-window package contains E as a finite BHist coordinate with hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support; absent quotient, host equality, chosen-point, choice, and completeness rows cannot be used by the consumer. □

Theorem 1205.127 (Diagonal-limit representative cofinal-budget stability)**.** *Let $\Omega = (W, R, D, M, E)$ and $\Omega' = (W', R', D', M', E')$ be extracted root-window packages for accepted completion-selector ledgers classified together by the recorded* hsame *rows and displayed* Cont *replay. Then the cofinal budget rows M and M' remain on the same* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *route, and the associated*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows are preserved componentwise as the same root-window handoff.*

Proof. Start with completion-selector stability, Definition 1205.109. It preserves the representative rows, finite-prefix rows, cofinal modulus rows, terminal seal rows, provenance rows, and local naming rows under the recorded transport and replay data.

Project the two extracted root-window packages. Since Ω and Ω' are obtained only by projection from the corresponding ledgers, the hsame and Cont rows that classify the ledgers also classify W, W', R, R', D, D', M, M', and E, E'.

It remains to identify the cofinal budget route. The cofinal-terminal bridge keeps the cofinal modulus budget in the same finite handoff as the terminal Real seal, so stability of the cofinal row preserves the entire root-window package componentwise. □

Theorem 1205.128 (Diagonal-limit representative terminal-selector exactness)**.** *Every terminal selector read made by a Real-and-Completion consumer of* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *factors through the same extracted root-window package* $\Omega = (W, R, D, M, E)$*. Thus the consumer reads the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows, together with the existing* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support, and no second selector route can supply the terminal seal.*

Proof. Use root-window extraction to obtain Ω from the accepted completion-selector ledger. The extraction has already gathered the finite window, readback, dyadic ledger, cofinal budget, and terminal seal rows into one displayed package.

Next apply phase-source coverage and the Phase Real and Completion exit certificate, Definition 1205.118 and Definition 1205.115. They identify the visible phase sources of the terminal read and exclude a hidden selector, quotient equality, selected host point, choice row, and ambient completeness principle.

Finally use cofinal-budget stability. Any accepted selector transport preserves the same root-window package under hsame and Cont, so a terminal selector read can only factor through W, R, D, M, E and the local support rows already present in the ledger. □

Theorem 1205.129 (Diagonal-limit representative selector choice refusal)**.** *Every terminal selector read accepted by a Real-and-Completion consumer of* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *factors through the extracted package* $\Omega = (W, R, D, M, E)$ *of Definition 1205.124. The read may use only the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *budget* M*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the recorded* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. It cannot use countable choice, hidden tail selection, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted completion-selector ledger to the root-window package. By root-window extraction, the terminal read exposes exactly the five displayed rows W, R, D, M, E together with the local support rows of the carrier.

Next apply terminal-selector exactness and the Phase Real and Completion exit certificate, Definition 1205.128 and Definition 1205.115. These results identify the terminal consumer surface with the finite packet and its four phase sources.

The refusal clauses follow from absence of coordinates. Since the package exports no countable-choice row, no tail selector, no quotient stream equality, and no ambient completeness field, a terminal read has no constructor through which any of those data can enter. □

Theorem 1205.130 (Diagonal-limit representative terminal route uniqueness)**.** *Let two accepted terminal selector reads expose the same displayed root-window package* $\Omega = (W, R, D, M, E)$ *and the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. Then they expose the same terminal route through* W, R, D, M, E *and cannot yield a second* $\mathsf{Real}^{\uparrow}$ *seal. In particular, equality of the displayed support identifies the terminal selector route as the unique finite route available to the Real-and-Completion consumer.*

Proof. Read both terminal selector claims through the common displayed package Ω. Terminal-selector exactness sends each accepted read to the route through the same window, readback, dyadic ledger, cofinal budget, and terminal seal rows.

Use cofinal-budget stability and regularity handoff, Definition 1205.127 and Definition 1205.125. The shared hsame and Cont support preserves the $W \to R$ regularity route and the cofinal budget row before the terminal seal is read.

Finally apply Real-seal admissibility and selector choice refusal. The only admissible seal is the displayed E reached after W, R, D, M, while selector choice refusal removes hidden tails, quotient equality, countable choice, and ambient completeness as alternate seal sources. Hence a second terminal route has no available coordinate. □

Theorem 1205.131 (Diagonal-limit representative Phase Real exit obligation package)**.** *The local Phase Real and Completion exit obligation for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is the package consisting of root-window extraction, regularity handoff, Real-seal admissibility, cofinal-budget stability, terminal-selector exactness, selector choice refusal, terminal route uniqueness, and the Phase Real and Completion exit certificate of Definition 1205.115. A downstream exit route for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *may consume this package as the finite terminal selector boundary and may not appeal to off-packet choice, quotient equality, hidden tail selection, selected host limits, or ambient Real completeness.*

Proof. Assemble the positive route first. Root-window extraction supplies $\Omega = (W, R, D, M, E)$; regularity handoff orders the $\mathsf{StreamName}^{\uparrow}$ window before the $\mathsf{RegSeqRat}^{\uparrow}$ readback; Real-seal admissibility identifies E as the displayed terminal seal; and cofinal-budget stability keeps M attached to the same finite terminal route.

Next add the selector boundary. Terminal-selector exactness says every accepted terminal read factors through Ω, selector choice refusal excludes non-displayed selector principles, and terminal route uniqueness prevents a second Real seal from being produced by the same support rows.

The Phase Real and Completion exit certificate supplies the consumer-facing source inventory. Therefore the local obligation package consumed by the exit route is precisely the finite carrier route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with the stated refusal and uniqueness boundaries attached. □

Definition 1205.132 (Diagonal-limit representative route-read selector package)**.** *For an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier*

$$D = (X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the route-read selector package is the finite triple

$$\rho(D) = (\mathsf{indexRead}, \mathsf{terminalRead}, \mathsf{routeRead})$$

whose rows are exactly the displayed continuation outputs

$$\mathsf{Cont}(\mu, n, \mathsf{indexRead}), \qquad \mathsf{Cont}(w, r, \mathsf{terminalRead}), \qquad \mathsf{Cont}(d, e, \mathsf{routeRead}).$$

The first read is the modulus-request selector, the second read is the window-readback selector, and the third read is the ledger-real-seal selector. The package is carried with the same $\mathsf{Pkg}^{\mathrm{sig}}$ *local-name row as* D *and introduces no quotient stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, hidden selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Theorem 1205.133 (Diagonal-limit representative route-read coverage)**.** *Every accepted* DiagonalLimitRepresentat *carrier together with the three displayed* Cont *rows of Definition 1205.132 exposes exactly the route reads* indexRead*,* terminalRead*, and* routeRead *and the existing* $\mathsf{Pkg}^{\mathrm{sig}}$ *local-name package. These four outputs are the route-read obligation surface of the carrier:* indexRead *reads from the modulus request,* terminalRead *reads from the selected window and readback,* routeRead *reads from the ledger and terminal Real seal, and the local package row remains the one already named by the carrier.*

Proof. Project the carrier of Definition 1205.1. It provides the rows $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and the local $\mathsf{Pkg}^{\mathrm{sig}}$ package over the same finite BHist surface.

Apply Definition 1205.3. With the three displayed continuation rows as inputs, it returns exactly the three derived reads and the same local package signature, with no additional carrier coordinate.

The sibling root results identify where those reads may be consumed. Root-readiness, Definition 1205.44, makes the modulus, window, readback, ledger, and seal route visible, while the root-window completion handoff keeps the Real-and-Completion consumer on that finite route. Hence coverage is precisely the package displayed above. □

Theorem 1205.134 (Diagonal-limit representative route-read unary closure)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier and route-read selector package* $\rho(D)$*, the three derived reads satisfy*

$$\mathsf{UnaryHistory}(\mathsf{indexRead}), \qquad \mathsf{UnaryHistory}(\mathsf{terminalRead}), \qquad \mathsf{UnaryHistory}(\mathsf{routeRead}).$$

Their unary status is obtained only by Cont *closure from the carrier rows* μ, n, w, r, d, e*; it is not a separate analytic or quotient witness.*

Proof. Read the unary carrier rows from Definition 1205.3. The theorem records μ, n, w, r, d, e as unary histories of the accepted packet.

Use the three Cont rows in the selector package. The same obligation theorem closes indexRead from μ, n, terminalRead from w, r, and routeRead from d, e by unary continuation closure.

Therefore the route reads remain finite BHist rows. The proof has no coordinate for quotient-stream equality, host Real equality, a chosen completed point, countable choice, or ambient Real completeness. □

Theorem 1205.135 (Diagonal-limit representative route-read consumer boundary)**.** *Downstream Real-and-Completion consumers of* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *may use the reads* indexRead*,* terminalRead*, and* routeRead *only through the finite package of Definition 1205.132. In particular, a consumer cannot replace that package by quotient-stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, a hidden selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeCarrier route read consumer boundary`

Proof. Begin with route-read coverage and unary closure. They make the three reads available exactly as finite BHist outputs of the displayed continuation rows, together with the existing local package signature.

Now apply the root-readiness exactness package and the root-window completion handoff, Definition 1205.49 and Definition 1205.124. These results require downstream consumers to factor through the displayed modulus, window, readback, ledger, cofinal-budget, and Real-seal route before consuming the terminal endpoint.

It remains to exclude stronger replacements. The carrier and the handoff theorems list no coordinate for quotient equality, host Real equality, hidden selectors, countable choice, or ambient completeness. Since the route-read package is exhausted by the three Cont outputs and the existing $\mathsf{Pkg}^{\mathrm{sig}}$ row, any such replacement lies outside the accepted consumer boundary. □

Theorem 1205.136 (Diagonal-limit representative roadmap exit inventory)**.** *The roadmap exit inventory for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is the ten-field paper-side contract obtained from the traditional diagonal representative as follows. Its source row is exactly the four-object route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

Its carrier is Definition 1205.1; its classifier is Definition 1205.2; its stability is supplied by terminal stability, completion-selector stability, and the audit-clean handoff of Definition 1205.5, Definition 1205.109, and Definition 1205.116; its ledger remains inside the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows; its exactness is the phase-source coverage, Real handoff exactness, and completion-frontier readback coverage of Definition 1205.118, Definition 1205.120, and Definition 1205.123; its bridge slot is pending; and its status is a paper-side source contract for Real-and-Completion consumers.*

Proof. Start from the traditional diagonal representative encoded by the carrier and classifier. These two definitions display the family row, modulus row, selected indices, selected $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, terminal $\mathsf{Real}^{\uparrow}$ seal, and local support rows.

The source inventory is the four-object exit packet of Definition 1205.114 and Definition 1205.115. Those results identify exactly the DyadicRatCore, StreamName, RegSeqRat, and Real rows used by the Phase Real and Completion consumer.

Stability and exactness are inherited from the already displayed terminal, selector, and handoff theorems named in the statement. Since each cited theorem uses the same finite ledger and the same local support rows, collecting them gives one citable ten-field contract rather than a stronger analytic object. □

Theorem 1205.137 (Diagonal-limit representative roadmap exit nonescape)**.** *The roadmap exit contract of Definition 1205.136 exports no quotient stream equality, no host* $\mathsf{Real}^{\uparrow}$ *equality, no selected completed point, no countable choice row, no trichotomy principle, no open-cover compactness principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem. Any accepted read of the contract must remain a read of the displayed finite route and its local BEDC support.*

Proof. Apply the Phase Real and Completion exit certificate. It already excludes quotient equality, choice, selected-point data, ambient completeness, trichotomy, and open-cover compactness from the exit packet.

Next apply selector-source nonescape and completion-frontier readback coverage, Definition 1205.122 and Definition 1205.123. They keep every accepted selector and downstream read inside the four source rows and the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support.

Thus none of the excluded analytic or classical principles has a coordinate through which it can be exported by the roadmap exit contract. □

Theorem 1205.138 (Diagonal-limit representative roadmap exit consumer contract)**.** *Every downstream Real-and-Completion consumer may cite the roadmap exit contract only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow},$$

together with the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows. The citation authorizes the carrier, classifier, ledger, stability, nonescape, and exactness fields named in Definition 1205.136, and it authorizes no route outside those rows.*

Proof. Begin with audit-clean consumer handoff, Definition 1205.116. It permits downstream consumers to cite the Phase Real and Completion exit packet as a finite BEDC source surface and refuses off-packet completion principles.

Use completion handoff exhaustion and phase-source coverage, Definition 1205.110 and Definition 1205.118. Together they force every consumer read through the displayed source rows and local support coordinates.

Finally apply the route-read consumer boundary, Definition 1205.135. It prevents the consumer from replacing the route-read package by quotient, host, selector, choice, or completeness data. Therefore the citation contract is exactly the finite route and support surface stated above. □

Theorem 1205.139 (Diagonal-limit representative exit object coverage)**.** *The roadmap exit contract for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *covers exactly the four L10 objects*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The $\mathsf{DyadicRatCore}^{\uparrow}$ *row is the dyadic ledger face, the* $\mathsf{StreamName}^{\uparrow}$ *row is the finite-window face, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row is the regular-rational readback face, and the* $\mathsf{Real}^{\uparrow}$ *row is the terminal seal face. Every accepted object read of the contract factors through the displayed finite route and the local* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support; no fifth L10 object is part of this exit surface.*

Proof. Start with the roadmap exit inventory, Definition 1205.136. Its source row is already the displayed four-object route, and its carrier, classifier, ledger, stability, and exactness fields are all attached to that same route.

Apply four-object source exhaustion and the Phase Real and Completion exit certificate, Definition 1205.114 and Definition 1205.115. These results identify the Phase Real and Completion packet with the dyadic ledger, finite window, regular-rational readback, and terminal seal rows.

Finally use phase-source coverage and completion-frontier readback coverage, Definition 1205.118 and Definition 1205.123. They keep downstream reads inside the four named rows and the displayed local support, so the contract has exactly the objectwise coverage stated above. □

Theorem 1205.140 (Diagonal-limit representative exit audit threshold)**.** *Downstream Phase Real and Completion consumers may treat the roadmap exit contract for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *as an audit-threshold handoff only through the carrier row of Definition 1205.1, the classifier row of Definition 1205.2, the ledger rows displayed in Definition 1205.136, the stability rows named there, and the exactness rows supplied by phase-source coverage, Real handoff exactness, and completion-frontier readback coverage. The handoff does not assert a* $\mathsf{bridgeChecked}$ *status, a standard bridge row, or an ambient completeness theorem.*

Proof. Begin with the roadmap exit inventory and consumer contract, Definition 1205.136 and Definition 1205.138. Together they list the only rows a downstream consumer may cite: carrier, classifier, ledger, stability, nonescape, and exactness over the finite route.

For the exactness side, use phase-source coverage, Real handoff exactness, and completion-frontier readback coverage. These theorems supply the permitted objectwise and readback obligations without adding a bridge slot or a completeness principle.

For the boundary side, apply audit-clean consumer handoff and route-read consumer boundary, Definition 1205.116 and Definition 1205.135. They allow the packet to be audited as a finite BEDC source surface while blocking any read as $\mathsf{bridgeChecked}$, standard bridge data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1205.141 (Diagonal-limit representative exit nonchoice boundary)**.** *The roadmap exit contract for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *refuses quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every accepted downstream use must remain a finite route read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the displayed local support rows.*

Proof. Apply roadmap exit nonescape, Definition 1205.137. It explicitly excludes quotient stream equality, host $\mathsf{Real}^{\uparrow}$ equality, selected completed points, choice rows, trichotomy, open-cover compactness, and ambient completeness from the roadmap exit contract.

Next use the Phase Real and Completion exit certificate, Definition 1205.115. It gives the same negative boundary for the four-object exit packet and keeps the packet constructive: only the displayed source rows and local support are exported.

Finally compose with completion-frontier readback coverage and the roadmap exit consumer contract, Definition 1205.123 and Definition 1205.138. These results force every downstream read back to the finite route, so none of the refused principles can enter through a consumer citation. □

Theorem 1205.142 (Diagonal-limit representative face threshold readiness)**.** *Every accepted completion-selector ledger for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exposes a finite objectwise threshold package whose faces are exactly*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The $\mathsf{DyadicRatCore}^{\uparrow}$ *face records the dyadic tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *face records the finite observation window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *face records the regular-rational readback, and the* $\mathsf{Real}^{\uparrow}$ *face records the terminal seal. These four faces are read from the ledger only through its displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support.*

Proof. Apply four-object source exhaustion and the Phase Real and Completion exit certificate, Definition 1205.114 and Definition 1205.115. They identify the same four rows as the complete exit packet for Phase Real and Completion consumers.

Next apply phase-source projection totality, Definition 1205.121. It sends every accepted ledger to the dyadic ledger, finite window, regular-rational readback, and terminal seal, while the local support rows serve only as transport, replay, sourcing, and naming data.

Thus the threshold package is objectwise: it exposes the four named faces and no extra route coordinate beyond the displayed finite ledger support. □

Theorem 1205.143 (Diagonal-limit representative objectwise audit threshold handoff)**.** *Downstream audit reads of the objectwise threshold package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *may consume the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *face records only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. In particular, an audit read may cite the four faces as a finite route surface, but it may not replace the supporting rows by detached quotient, host-limit, selector, choice, completeness, trichotomy, compactness, or bridge data.*

Proof. Begin with completion-frontier readback coverage, Definition 1205.123. It confines every Real-and-Completion consumer read to the finite frontier through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ together with the displayed local support.

Then use the roadmap exit consumer contract, Definition 1205.138. That contract authorizes downstream citation of the carrier, classifier, ledger, stability, nonescape, and exactness fields only through the same four source rows and support records.

Combining these two reads gives the audit handoff. The consumer sees the four face records as a finite source surface, and every permitted audit step remains anchored in BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support. □

Theorem 1205.144 (Diagonal-limit representative exit threshold consumer boundary)**.** *The objectwise threshold package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *supplies no route-level formalstatus, no quotient stream, no selected host limit, no countable choice row, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, no trichotomy principle, no open-cover compactness principle, and no standard bridge row. Its consumer boundary is exactly the four objectwise faces of Definition 1205.142 together with the local support allowed by Definition 1205.143.*

Proof. First apply the Phase Real and Completion exit certificate, Definition 1205.115. It excludes quotient equality, choice, ambient completeness, trichotomy, and open-cover compactness from the finite exit packet.

Next apply phase-source projection totality and completion-frontier readback coverage, Definition 1205.121 and Definition 1205.123. These results remove host-limit, selected-point, fifth-source, and second-selector coordinates from the accepted readback surface.

Finally use the roadmap exit consumer contract, Definition 1205.138. Since that contract is only a finite source route and does not fill the bridge slot, the threshold package cannot be read as a route-level formalstatus or standard bridge row. The package therefore ends at the four objectwise faces and their local BEDC support. □

Definition 1205.145 (Diagonal-limit representative terminal-modulus coherence surface)**.** *For an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier consumed by a Real-and-Completion route, the terminal-modulus coherence surface is the finite packet*

$$T = (\mu, n, k, i, w, r, d, H, C, P, N).$$

The rows μ *and* n *are the displayed modulus request and precision request. The rows* $k = \mu(1/(n+1))$ *and* $i = k + n + 1$ *are the selected family and readback indices. The row* w *is the* $\mathsf{StreamName}^{\uparrow}$ *pointwise request selected by those indices,* r *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-window readback, and* d *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that sits between the readback and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The rows* H, C, P, N *are only the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *row for that same route.*

The packet T *is the terminal route made visible by Definition 1205.29, Definition 1205.44, Definition 1205.64, and Definition 1205.89. It is not a quotient stream, a host Real limit, a choice-selected tail, an ambient completeness witness, or a second Real seal.*

Theorem 1205.146 (Diagonal-limit representative terminal-modulus coherence)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *Real-and-Completion route factors through the terminal-modulus coherence surface of Definition 1205.145. Thus the consumer can reach the terminal seal only after it has exposed the same packet*

$$(\mu, n, k, i, w, r, d, H, C, P, N)$$

that records the StreamName request, RegSeqRat regular window, DyadicRatCore tolerance ledger, and local support rows.

Proof. Start with the modulus-window-readback package, Definition 1205.29. It says that every route to the terminal $\mathsf{Real}^{\uparrow}$ seal first passes through $\mu, n, k, i, w, r, d,$ while H, C, P, N can only transport, replay, source, and name that same route.

Next attach root readiness. The root-readiness triad of Definition 1205.44 makes the selected modulus request, pointwise window, readback row, dyadic ledger, terminal seal, and support rows visible before a downstream consumer can use them.

The cofinal-tail coverage theorem Definition 1205.64 places those rows on the same finite tail surface as the cofinal and finite-prefix budget rows. The Real seal handoff theorem Definition 1205.89 then confines the downstream endpoint to the finite root-unblock obligation over BHist, Cont, Pkg, and $\mathsf{NameCert}$.

Therefore the terminal seal read has no independent entry point. It factors through the packet T, and the visible StreamName, RegSeqRat, DyadicRatCore, and Real rows remain coherent along that one terminal-modulus route. □

Theorem 1205.147 (Diagonal-limit representative Real-RegSeqRat-StreamName tail factorization)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *terminal route, any* $\mathsf{Real}^{\uparrow}$ *read of the terminal seal consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail only through the selected* $\mathsf{StreamName}^{\uparrow}$ *pointwise request and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in the terminal-modulus coherence surface. Concretely, the Real-facing consumer may read the tail only in the order*

$$\mu, n, k, i, w, r, d,$$

with w supplying the StreamName request, r supplying the RegSeqRat readback for that request, and d supplying the DyadicRatCore tolerance ledger before the seal is consumed.

Proof. Begin with the root-readiness triad Definition 1205.44. It exposes the selected indices and the ordered window-readback-ledger route as data of the accepted carrier rather than as an ambient Real construction.

Now apply the modulus-window-readback package Definition 1205.29. The theorem orders the terminal route through μ, n, k, i, w, r, d, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback row r is unavailable until the selected $\mathsf{StreamName}^{\uparrow}$ request w has already been exposed.

The cofinal-tail coverage theorem Definition 1205.64 keeps the selected tail on the same finite BHist route as the cofinal modulus and finite-prefix budget rows. Finally, Definition 1205.89 allows the Real-facing endpoint to consume only this finite root-unblock package.

Hence the Real read cannot skip from a tail name to a seal. It must pass through the StreamName pointwise request, the RegSeqRat readback, and the DyadicRatCore tolerance ledger recorded in T. □

Theorem 1205.148 (Diagonal-limit representative selector-modulus nonescape)**.** *The terminal-modulus coherence surface of Definition 1205.145 exhausts the selector and modulus data available to a terminal* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *route. No accepted consumer can replace $k = \mu(1/(n+1))$, $i = k + n + 1$, the selected StreamName request w, the RegSeqRat readback r, or the DyadicRatCore ledger d by a hidden tail selector, quotient stream equality, countable choice row, host* $\mathsf{Real}^{\uparrow}$ *limit, or ambient Real-completeness field.*

Proof. First read the terminal-modulus coherence theorem Definition 1205.146. Every terminal seal read factors through the packet T, so a selector or modulus row outside T would have to be an extra coordinate of the same accepted route.

The root-readiness triad Definition 1205.44 and the modulus-window-readback package Definition 1205.29 account for the displayed computations $k = \mu(1/(n+1))$ and $i = k+n+1$, the selected window, the readback, and the dyadic ledger. These results leave no second selector channel for the terminal consumer.

The cofinal-tail coverage theorem Definition 1205.64 supplies the only cofinal tail route, while the Real seal handoff theorem Definition 1205.89 states the endpoint as a finite root-unblock obligation. The excluded objects are stronger than those rows and are not part of the carrier, classifier, BHist support, Cont replay, Pkg provenance, or NameCert ledger.

Thus selector-modulus data cannot escape the displayed packet. Quotient streams, countable choice, hidden tail selectors, host Real limits, and ambient completeness remain outside the terminal route. □

Theorem 1205.149 (Diagonal-limit representative terminal-modulus scope lock)**.** *For every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *terminal route consumed by a Real-and-Completion endpoint, the terminal modulus route is scoped precisely by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows together with the sibling* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. In particular, the route available to the consumer is the terminal-modulus packet*

$$(\mu, n, k, i, w, r, d, H, C, P, N)$$

of Definition 1205.145, read through chapter 291, chapter 126, chapter 301, and chapter 38; no off-packet selector, host Real equality, quotient stream equality, countable-choice row, or ambient completeness field lies in this scope.

Proof. Start from the carrier and classifier of Definition 1205.1 and Definition 1205.2. The carrier exposes only finite route rows, while classifier transport is recorded componentwise by the hsame and Cont support used by the local route.

Apply the NameCert obligation surface through Definition 1205.3, Definition 1205.80, and Definition 1205.83. These results identify carrier admission, displayed replay, provenance, and local naming as the complete support by which the terminal route can be read.

Finally attach terminal-modulus coherence, Definition 1205.146. It factors the terminal seal through the same packet $(\mu, n, k, i, w, r, d, H, C, P, N)$ and ties the four sibling rows to the DyadicRatCore, StreamName, RegSeqRat, and Real chapters. Therefore the terminal modulus route is locked to the stated finite scope. □

Theorem 1205.150 (Diagonal-limit representative cofinal-tail exactness lock)**.** *Every cofinal tail read accepted by a* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *terminal route is exact only through the finite selector*

$$(\mu, n, k, i, w, r, d)$$

with $k = \mu(1/(n+1))$ *and* $i = k + n + 1$*. The read uses* w *as the displayed* $\mathsf{StreamName}^{\uparrow}$ *request,* r *as the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* d *as the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger; it does not introduce quotient-stream equality, a hidden tail selector, countable choice, host-limit selection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the finite selector from the modulus-window-readback package and the terminal diagonal selector surface, Definition 1205.29 and Definition 1205.98. They provide the displayed computations of k and i and place the selected window before the regular-rational readback.

Next use cofinal-tail coverage and exactness, Definition 1205.64 and Definition 1205.66. These theorems keep the cofinal read inside the finite BHist, hsame, Cont, Pkg, and local NameCert route rather than allowing a detached tail principle.

The nonescape side follows from terminal-modulus coherence and selector-modulus nonescape, Definition 1205.146 and Definition 1205.148. Since the accepted packet has no coordinate for quotient equality, hidden selection, choice, host limits, or ambient completeness, the cofinal tail read is exact only through the displayed selector. □

Theorem 1205.151 (Diagonal-limit representative phase-exit handoff lock)**.** *The packet obtained from the terminal-modulus scope lock and the cofinal-tail exactness lock is the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *Phase Real and Completion handoff consumed by* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$*. Its consumer-facing source rows are exactly* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with the support route still carried by* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$*; the handoff exports no quotient stream, selected completed point, countable-choice row, host Real limit, or ambient completion principle.*

Proof. First assemble the local package. The terminal-modulus scope lock supplies the finite route support, while the cofinal-tail exactness lock supplies the selected window, readback, and dyadic ledger through the displayed selector.

Now apply the existing Phase Real and Completion surfaces: phase-source coverage, four-object source exhaustion, and the Phase Real and Completion exit certificate, Definition 1205.118, Definition 1205.114, and Definition 1205.115. They identify the consumer-facing source inventory with the four DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Finally compare with the $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ carrier and obligation surface, Definition 1327.1 and Definition 1327.6. That carrier consumes the same objectwise rows through finite support and rejects fused completion data, quotient streams, selected limits, and host-completeness shortcuts. Hence the resulting packet is precisely the handoff available to the Phase Real and Completion exit consumer. □

Theorem 1205.152 (Diagonal-limit representative phase-root window totality)**.** *Every Phase Real and Completion root-window read accepted by* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is total on the finite packet*

$$(\mu, n, k, i, w, r, d, H, C, P, N)$$

where $k = \mu(1/(n+1))$ *and* $i = k + n + 1$*. The rows* w, r, d *are respectively the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *faces, while* H, C, P, N *are the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support rows. The terminal* $\mathsf{Real}^{\uparrow}$ *seal is reachable only after this packet is displayed, through the sibling chapters chapter 291, chapter 126, chapter 301, and chapter 38.*

Proof. Start with the carrier and classifier of Definition 1205.1 and Definition 1205.2. They expose only finite representative rows and compare them through the displayed support, so the root-window read has no coordinate outside BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ data.

Apply root-readiness exactness, the modulus-window-readback package, and the terminal-modulus scope lock, Definition 1205.49, Definition 1205.29, and Definition 1205.149. These results compute k and i, display the selected window and readback, and bind the dyadic ledger to the same finite route.

Finally use the four-object source exhaustion theorem, Definition 1205.114. It identifies the Real-and-Completion source inventory with the DyadicRatCore, StreamName, RegSeqRat, and Real rows. Hence every accepted phase-root window read is total exactly on the stated finite packet before the terminal Real seal is consumed. □

Theorem 1205.153 (Diagonal-limit representative phase-root readback factorization)**.** *Every accepted Phase Real and Completion readback for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *factors through the ordered root-window route*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

carried by the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows. The factorization reads the selected finite window* w *before the regular-rational readback* r*, reads the dyadic tolerance ledger* d *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and exports no quotient stream equality, host limit, selected completed point, choice row, or ambient completeness principle.*

Proof. First read the StreamName–RegSeqRat handoff, Definition 1205.16. It forces any accepted route toward the terminal seal to expose the selected $\mathsf{StreamName}^{\uparrow}$ window before the selected $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Next apply finite selector exhaustion and the cofinal-tail exactness lock, Definition 1205.90 and Definition 1205.150. These keep the selected readback attached to the displayed root indices and dyadic ledger rather than to a detached tail principle.

The terminal step is supplied by the Phase Real and Completion exit certificate, Definition 1205.115. Since that certificate has exactly the four named source rows and no stronger analytic coordinate, every accepted readback factors through the ordered root-window route stated above. □

Theorem 1205.154 (Diagonal-limit representative phase-root nonchoice boundary)**.** *The Phase Real and Completion root-window surface for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is a nonchoice boundary. Its available data are precisely the displayed* BHist *rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *naming, and the sibling* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces. It cannot be strengthened into quotient stream equality, host-limit selection, countable choice, a selected completed point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the root NameCert obligations, Definition 1205.3 and Definition 1205.80. They record carrier admission, classifier stability, provenance, and naming as finite obligations rather than as external equality or selection principles.

Use support-scope nonescape and Real-seal nonchoice export, Definition 1205.8 and Definition 1205.42. The support rows may transport and replay only the displayed packet, and the Real-facing seal cannot be replaced by host equality, quotient equality, a hidden selector, or choice data.

Finally read the objectwise audit threshold handoff, Definition 1205.143. It leaves downstream consumers with only the four objectwise faces and local support. Therefore the phase-root window surface is exactly the stated nonchoice boundary. □

Theorem 1205.155 (Diagonal-limit representative phase-root NameCert exhaustion)**.** *The* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRe}}$ *obligation for a Phase Real and Completion root-window read is exhausted by the carrier, classifier, root-window totality theorem, readback factorization theorem, and nonchoice boundary theorem. Equivalently, once the displayed* BHist, hsame, Cont, Pkg, *local* NameCert, *and sibling* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows are present, no additional NameCert clause is available for quotient streams, host limits, choice, selected completion points, or ambient completeness.*

Proof. The carrier and classifier supply the underlying admissible rows. The root-window totality theorem then records that every accepted phase-root read has a displayed finite packet before any terminal Real seal is read.

The readback factorization theorem supplies the ordered source route through StreamName, RegSeqRat, DyadicRatCore, and Real. Thus each consumer-visible readback coordinate is already attached to a named sibling source and to the local support rows.

The nonchoice boundary theorem closes the negative ledger. Since it removes quotient equality, host limits, choice rows, selected completed points, and ambient completeness from the surface, there is no remaining NameCert obligation outside the listed packet. □

Theorem 1205.156 (Diagonal-limit representative phase-root consumer handoff)**.** *Every Real-and-Completion consumer of the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *phase-root window receives exactly the finite handoff supplied by the phase-root NameCert exhaustion theorem. The handoff consists of the root-window totality packet, the readback factorization route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresenta}}$ *support. It exports no quotient stream equality, host limit, choice row, selected completed point, ambient completeness principle, or standard bridge row.*

Proof. First assemble the local surface from root-window totality and readback factorization. These two theorems provide the finite packet and its ordered sibling route, so the consumer has a concrete row to read for each source face.

Next apply phase-root NameCert exhaustion. It states that the displayed support rows and four sibling faces already cover the local NameCert obligation, with no additional completion or selection clause.

Finally compare with the phase-exit handoff lock, Definition 1205.151. That theorem identifies the Phase Real and Completion endpoint as a finite consumer handoff and refuses quotient streams, selected completed points, countable choice, host Real limits, and ambient completion principles. The consumer therefore receives exactly the finite phase-root window handoff and nothing stronger. □

Definition 1205.157 (Diagonal-limit representative selector identity surface)**.** *For an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet*

$$(X, \mu, n, k, i, w, r, d, e, H, C, P, N),$$

the selector identity surface is the displayed finite row

$$\Sigma_i = (X, \mu, n, k, i, w, r, d, e, H, C, P, N)$$

subject to the equations $k = \mu(1/(n+1))$ *and* $i = k + n + 1$, *where the same selected index* i *is read through the cofinal modulus route, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* w, *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* r, *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* d, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* e. *The support rows* H, C, P, N *are only the displayed* BHist, *componentwise* hsame, Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *rows for that same identity surface.*

The surface is the selector row exposed by the root-index readiness package, the root-window obligation surface, and the terminal-modulus coherence surface. It is not a quotient stream, a host Real limit, a selected completion point, a countable-choice row, an ambient $\mathsf{Real}^{\uparrow}$ *completeness field, or a second terminal selector.*

Theorem 1205.158 (Diagonal-limit representative selector identity totality)**.** *Every accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier exposes exactly one selector identity surface of Definition 1205.157. The exposed surface is total on the selected cofinal index and simultaneously determines the finite window, regular readback, dyadic ledger, and terminal Real seal route*

$$i \rightsquigarrow w \rightsquigarrow r \rightsquigarrow d \rightsquigarrow e$$

inside the recorded BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support.*

Proof. Start with the accepted carrier of Definition 1205.1. It records $X, \mu, n, k, i, w, r, d, e, H, C, P, N$ and the two selector equations $k = \mu(1/(n+1))$ and $i = k+n+1$ as finite carrier data, so existence of the displayed row is obtained by projection rather than by choosing a tail outside the packet.

Root-index coverage and the phase-root window totality theorem, Definition 1205.53 and Definition 1205.152, attach the selected index to the same finite StreamName, RegSeqRat, and DyadicRatCore route. Thus the selector identity row is total before the terminal seal is read.

For uniqueness, apply selected-window determinacy through the modulus-window-readback package and terminal-route uniqueness, Definition 1205.29 and Definition 1205.130. Once the carrier and support rows are fixed, a second selected index, window, readback, ledger, or seal would have to be a second terminal route, which those results exclude. □

Theorem 1205.159 (Diagonal-limit representative selector identity transport)**.** *Let two accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *selector identity surfaces be classified componentwise by the recorded* hsame *rows and by displayed* Cont *replay on their carrier, cofinal, window, readback, ledger, seal, provenance, and local naming rows. Then transport preserves the same selector identity through the window/readback/ledger/seal route. In particular, the transported surface has the same selected cofinal index route and the same ordered public path through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{CofinalModulusSeal}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Proof. Begin with classifier transport. The classifier of Definition 1205.2 and the hsame–Cont scope theorem Definition 1205.8 allow transport only through the recorded component rows and displayed replay of the same finite packet.

Next use cofinal selector transport, Definition 1205.85. It keeps the selected equations, selected window, readback, dyadic ledger, cofinal modulus rows, and terminal seal inside the displayed BHist, hsame, Cont, Pkg, and NameCert containers.

Finally apply finite-tail selector uniqueness, Definition 1205.101. Since the finite tail admits only one displayed selector route, componentwise transport cannot exchange the window/readback/ledger/seal path for another index, another tail, or another terminal Real seal. □

Theorem 1205.160 (Diagonal-limit representative selector identity Real-seal exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the selector identity surface of Definition 1205.157. Such a consumer may read the terminal seal only after the selected cofinal index, finite window, RegSeqRat readback, and DyadicRatCore ledger have appeared on the displayed surface; it cannot replace that surface by a hidden selector, quotient stream, host limit, countable-choice row, selected completion point, or ambient completeness coordinate.*

Proof. Read the positive factorization first. The Real-seal factorization theorem and Real-seal handoff theorem, Definition 1205.10 and Definition 1205.89, place every terminal Real read after the selected window, readback, dyadic ledger, and local support rows.

The selector identity totality theorem identifies those rows with the same selected index i and the same cofinal route. Therefore a seal consumer reaches e only through Σ_i, with no separate entry point for the Real endpoint.

The boundary is supplied by the nonchoice results, especially Definition 1205.11, Definition 1205.154, and Definition 1205.42. These results remove hidden selectors, quotient streams, host limits, selected completion points, choice rows, and ambient completeness fields from the carrier surface, so the selector identity surface exhausts every Real-seal consumer route. □

Theorem 1205.161 (Diagonal-limit representative audit-clean exit gate)**.** *The finite* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *packet is the citable audit-clean exit gate for the Phase Real and Completion route exactly through the four current L10 phase objects*

$$\mathsf{DyadicRatCore}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{RegSeqRat}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}.$$

The gate is the roadmap-facing read of the carrier, classifier, selector identity surface, phase-source package, and formal handoff surface of this chapter. It authorizes the Phase Real and Completion exit criterion only as a finite route through those four faces and the displayed BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support rows. It does not authorize quotient streams, selected limits, countable choice, host* $\mathsf{Real}^{\uparrow}$ *equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or* $\mathsf{bridgeChecked}$ *status.*

Proof. Start with the roadmap exit inventory and consumer contract, Definition 1205.136 and Definition 1205.138. They identify the downstream citation surface with the four-object route and the local support rows, not with a completed host Real object.

Next attach the phase-source and selector-identity surfaces. The phase-source coverage theorem Definition 1205.118 names the four current L10 faces, while Definition 1205.158 fixes the selected window, readback, dyadic ledger, and terminal seal as one displayed finite route.

Finally apply the phase-exit handoff lock and the formal handoff boundary, Definition 1205.151 and Definition 1205.69. They allow the Phase Real and Completion consumer to read the finite route as an audit-clean exit gate while refusing quotient streams, selected limits, countable choice, host equality, ambient completeness, and any bridge-checked conclusion. □

Theorem 1205.162 (Diagonal-limit representative objectwise audit closure)**.** *Objectwise audit closure for the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *exit gate is exhausted by the four faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *together with the displayed route rows*

$$X, \mu, n, k, i, w, r, d, e, H, C, P, N.$$

The rows w, r, d, e *are respectively the StreamName, RegSeqRat, DyadicRatCore, and Real faces;* H, C, P, N *are only the* BHist*-recorded* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *naming rows. No additional objectwise audit coordinate is supplied by quotient streams, selected limits, countable choice, host* $\mathsf{Real}^{\uparrow}$ *equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Proof. Read the carrier and classifier of Definition 1205.1 and Definition 1205.2. They display exactly the family, modulus, selected indices, window, readback, ledger, seal, transport, replay, provenance, and naming rows used by the local certificate.

The object faces are then fixed by the face threshold readiness theorem and the objectwise audit threshold handoff, Definition 1205.142 and Definition 1205.143. These results read w, r, d, e as the four current L10 faces and route every audit use through H, C, P, N.

For exhaustion, use the selector identity Real-seal exhaustion theorem and the exit threshold consumer boundary, Definition 1205.160 and Definition 1205.144. They remove hidden selectors, quotient streams, selected host limits, countable choice, ambient completeness, route-level formal-status, and bridge rows from the objectwise audit surface. □

Theorem 1205.163 (Diagonal-limit representative formal-axis unblock)**.** *The formal-axis handoff shape for the later* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *Lean audit is the finite obligation surface already carried by*

$$\mathsf{BEDC.Derived.DiagonallimitrepresentativeUp.DiagonalLimitRepresentativeNameCertObligations}.$$

The paper-side exit gate therefore offers a single downstream handoff shape: carrier admission, diagonal-index computation, selected StreamName window, RegSeqRat readback, DyadicRatCore ledger, terminal Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. This handoff does not add a paper marker here, does not assert* bridgeChecked*, and does not import quotient streams, selected limits, countable choice, host* $\mathsf{Real}^{\uparrow}$ *equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the checked NameCert obligation site, Definition 1205.3. Its statement lists the finite rows that the Lean theorem named in the formal-axis target already consumes: index computation, selected window, readback, dyadic ledger, terminal seal, transport, replay, provenance, and local naming.

Next use the audit-readiness package and audit-clean threshold package, Definition 1205.68 and Definition 1205.72. Together they collect the carrier, classifier, shared terminal route, root-readiness exactness, terminal pullback coverage, and determinacy data into one audit-ready finite route.

Finally compare with the Lean-readiness obligation boundary, Definition 1205.74. That boundary permits only the displayed finite rows and sibling faces to pass to later formal work, while standard bridge assertions and stronger Real-completeness claims remain outside the handoff. Thus the formal axis is unblocked by a finite obligation surface rather than by an unrecorded analytic principle. □

Theorem 1205.164 (Diagonal-limit representative root-window selector lock)**.** *Every Phase Real and Completion root-window selector read of an accepted* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *carrier factors through the displayed finite route*

$$(\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert})$$

already named by the chapter. Concretely, the read exposes the selected equations $k = \mu(1/(n+1))$ *and* $i = k+n+1$*, the finite window* w*, regular-rational readback* r*, dyadic ledger* d*, terminal seal* e*, and support rows* H, C, P, N *before any downstream consumer can use the selector. No selector read bypasses those rows through quotient stream equality, host* $\mathsf{Real}^{\uparrow}$ *equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier and classifier rows of Definition 1205.1 and Definition 1205.2. They list exactly the selected index equations, window, readback, ledger, terminal seal, transport, continuation, provenance, and local naming rows available to an accepted carrier.

Next apply the phase-root window totality and readback factorization theorems, Definition 1205.152 and Definition 1205.153. These force the consumer to read the selected $\mathsf{StreamName}^{\uparrow}$ window before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the phase-root nonchoice boundary and NameCert exhaustion, Definition 1205.154 and Definition 1205.155. They identify BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ as the whole support surface, so the forbidden quotient, host-equality, selected-limit, choice, and completeness data have no coordinate in the selector read. □

Theorem 1205.165 (Diagonal-limit representative cofinal-modulus route totality)**.** *The cofinal modulus route consumed by* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *is total only through the finite selected-index route, finite-window route, dyadic-readback route, and terminal Real-seal route already displayed by the packet. Its positive data are the selected equations* $k = \mu(1/(n+1))$ *and* $i = k+n+1$*, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row, the*

cofinal budget rows, the terminal $\mathsf{Real}^{\uparrow}$ *seal row, and the recorded* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support. It contains no quotient stream, countable-choice row, host real equality, or ambient completeness field.*

Proof. Begin with the cofinal-index source and budget lock, Definition 1205.60 and Definition 1205.61. They attach the selected-index equations to the finite window, readback, dyadic ledger, cofinal budget, and terminal seal before a downstream consumer can read the route.

Now apply cofinal-budget seal totality, Definition 1205.57. It orders every terminal seal read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, the finite-prefix budget, the cofinal modulus rows, and $\mathsf{Real}^{\uparrow}$.

It remains to close the negative boundary. The terminal consumer cofinal nonescape theorem and Real-completion consumer exactness, Definition 1205.58 and Definition 1205.86, keep all transport inside the displayed structural rows and exclude quotient streams, countable choice, host real equality, and ambient completeness from the total route. □

Theorem 1205.166 (Diagonal-limit representative phase-exit unblock package)**.** *The root-unblock package for* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *consumed by the Phase Real and Completion exit route is the package consisting of the root-window selector lock of Definition 1205.164, the cofinal-modulus route totality theorem of Definition 1205.165, Real-completion consumer exactness Definition 1205.86, the formal handoff boundary Definition 1205.69, and the Phase Real and Completion exit certificate Definition 1205.115. A downstream exit consumer may cite this package only as a finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{DiagonalLimitRepresentative}^{\uparrow}}$ *support; it cannot infer Real completeness, host equality, quotient streams, selected completed limits, countable choice, or bridge-checked status.*

Proof. Assemble the two local obligations first. The root-window selector lock supplies the selected-index, finite-window, readback, dyadic ledger, terminal seal, and support rows, while the cofinal-modulus route totality theorem attaches the cofinal budget route to the same finite terminal read.

Next add Real-completion consumer exactness and the formal handoff boundary. Exactness confines every downstream consumer to the displayed StreamName, RegSeqRat, DyadicRatCore, cofinal budget, Real, and support rows. The handoff boundary states that later formal work may encode those rows without importing quotient equality, host equality, selected limits, choice, or ambient completeness.

Finally cite the Phase Real and Completion exit certificate. It identifies the consumer-facing source inventory with $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Therefore the exit route receives exactly the finite root-unblock package stated above and no stronger analytic or bridge claim. □

Chapter 1206

A Concrete Naming Certificate for $\mathsf{ConsciousUniverseSpace}^{\uparrow}$

The $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ packet realizes the finite universe-slice, local-time, locality-cell, and trace-identity surface of section 9462.27 as one records-side NameCert carrier. It sits after the focused ledger surface of chapter 1109, the locality-cell surface of chapter 509, the observer-locality surface of chapter 513, and the finite two-history locality packet of chapter 1194. Those siblings supply the nearby ledger, cell, observer, and two-history locality rows; this packet binds the universe-slice read, local-time chain, locality-cell space graph, and focused trace-identity row into one carrier.

Definition 1206.1 (ConsciousUniverseSpace carrier)**.** *A* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (L, U, T, S, I, H, C, P, N)$$

with the following displayed rows:

1. *L is the focused ledger row, read through* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$*, that fixes the finite family of histories available to the focus;*
2. *U is the universe-slice row, the finite subfamily of histories, selectors, digest rows, gap rows, provenance rows, relation rows, invariant rows, and dependence rows admitted by L;*
3. *T is the local-time chain row, recording the finite continuation order internal to the displayed histories rather than a global scalar clock;*
4. *S is the locality-cell space graph row, generated by* $\mathsf{LocalityCell}^{\uparrow}$*,* $\mathsf{ObserverLocalityCell}^{\uparrow}$*, and* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *reads over the histories admitted by U;*
5. *I is the trace-identity row, recording the focused self-proxy and* hsame*-stable trace continuity of the focused history;*
6. *H records componentwise* hsame *transport for L, U, T, S, I, C, P, N;*
7. *C records the displayed* Cont *routes from the focused ledger through the universe slice, local-time chain, locality-cell graph, and trace-identity row;*
8. *P is the* Pkg *provenance over* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$*,* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$*,* $\mathsf{LocalityCell}^{\uparrow}$*,* $\mathsf{ObserverLocalityCell}^{\uparrow}$*,* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConsciousUniverseSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global universe object, God's-eye world, shared scalar time, independent space substrate, owner-subject carrier, host equality of histories, or ambient spacetime manifold.

Lean4 ✓ `BEDC.Derived.ConsciousUniverseSpaceUp.taste gate`

Theorem 1206.2 (ConsciousUniverseSpace NameCert obligations)**.** *For the carrier of Definition 1206.1, the local* $\mathsf{NameCert}_{\mathsf{ConsciousUniverseSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for* $X = (L, U, T, S, I, H, C, P, N)$*, focused-ledger admission through* L*, universe-slice readback through* U*, local-time chain exposure through* T*, locality-cell graph exposure through* S*, trace-identity exposure through* I*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.ConsciousUniverseSpaceUp.ConsciousUniverseSpaceNameCertObligations`

Proof. Project the carrier definition. The rows $L, U, T, S, I, H, C, P, N$ are the complete displayed inventory before any universe-slice, space, time, or identity consumer can read the packet.

The focused ledger row supplies only the finite histories admitted to the focus. The universe-slice row refines that finite ledger into displayed histories and evidence rows, while the local-time row records continuation order internal to those histories.

The locality-cell graph and trace-identity rows close the records-side geometry. The graph row uses locality-cell siblings to express inter-history coordination, and the identity row records focused trace continuity without adding an owner-subject coordinate.

The transport, continuation, provenance, and naming rows preserve and replay this finite packet. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1206.3 (ConsciousUniverseSpace locality non-escape)**.** *Every public locality, time, or identity read of an accepted* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *packet factors through the displayed rows*

$$L, U, T, S, I, H, C, P, N.$$

The packet cannot replace the locality-cell graph row S *by a global spacetime substrate, replace the local-time chain* T *by a shared scalar clock, or replace the trace-identity row* I *by a subject-owner carrier.*

Proof. Begin with the NameCert obligations. A consumer reaches the universe-slice, local-time, locality-cell, and trace-identity surfaces only through the rows U, T, S, I inside the accepted carrier.

Read the sibling boundaries next. $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ fixes the focus-facing finite ledger, $\mathsf{LocalityCell}^{\uparrow}$ and $\mathsf{ObserverLocalityCell}^{\uparrow}$ supply local relation cells, and $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ supplies the two-history locality packet. None of these siblings exports a global world object or a host spacetime row.

Finally use the transport and replay rows. The rows H, C, P, N preserve the displayed packet and its provenance, but they do not add coordinates. Thus locality, time, and identity reads remain confined to $L, U, T, S, I, H, C, P, N$. □

Theorem 1206.4 (ConsciousUniverseSpace independence witness)**.** *The carrier of Definition 1206.1 is independent from each nearest sibling named in its provenance:* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ *lacks the locality-cell space graph and trace-identity rows,* $\mathsf{LocalityCell}^{\uparrow}$ *lacks the focused universe-slice and local-time chain rows,* $\mathsf{ObserverLocalityCell}^{\uparrow}$ *lacks the universe-slice row, and* $\mathsf{FiniteHistLocalityPacket}^{\uparrow}$ *lacks the focused trace-identity row. Hence none of those siblings certifies the same* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *carrier.*

Proof. Compare the displayed carrier rows. The present packet requires the simultaneous rows $L, U, T, S, I, H, C, P, N$; each cited sibling supplies only one part of that inventory.

The focused ledger sibling supplies the focus-facing ledger but not the locality-cell space graph or trace-identity row. The locality-cell siblings supply local relation geometry but not the universe slice and local-time chain generated from the focused ledger.

The finite two-history locality packet supplies a two-history locality surface but not the focused self-proxy trace identity. Therefore no sibling carrier can be renamed into X without adding a row missing from its own displayed packet. □

Falsifiable prediction: A packet omitting either the locality-cell graph row or the trace-identity row cannot certify the same ConsciousUniverseSpace carrier.

Independence witness: focusedmultihistledger, localitycell, observerlocalitycell, finitehistlocalitypacket: none combines finite universe slice, locality-cell geometry, local-time chain, and trace-identity rows in one carrier.

The follow-up Lean TasteGate surface for this chapter must provide explicit `Nontrivial ConsciousUniverseSpaceUp` and `FieldFaithful ConsciousUniverseSpaceUp` instances for the same finite carrier, with the canonical declaration surface expected at `BEDC.Derived.ConsciousUniverseSpaceUp`.

Theorem 1206.5 (ConsciousUniverseSpace scoped-kernel witness)**.** *Every locality, time, trace-identity, and universe-slice consumer of an accepted* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *packet factors through the displayed rows*

$$L, U, T, S, I, H, C, P, N$$

and only through the BEDC kernel primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The scoped kernel for such consumers is therefore the carrier of Definition 1206.1, the NameCert obligations of Definition 1206.2, the locality non-escape theorem Definition 1206.3, and the independence witness of Definition 1206.4.*

Proof. Project the carrier first. The rows $L, U, T, S, I, H, C, P, N$ are the complete displayed inventory, and the carrier text explicitly excludes a global universe object, shared scalar time, independent space substrate, owner-subject carrier, host equality of histories, and ambient spacetime manifold.

Apply the NameCert obligations. They assign focused-ledger admission to L, universe-slice readback to U, local-time exposure to T, locality-cell graph exposure to S, trace-identity exposure to I, componentwise hsame transport to H, displayed Cont replay to C, Pkg provenance to P, and local NameCert naming to N.

Use locality non-escape to constrain every public locality, time, and identity read to the same rows. The independence witness then separates this packet from its nearest siblings, so no sibling carrier can supply a second kernel route for the combined universe slice, local-time chain, locality-cell graph, and trace-identity surface.

Thus the only primitives available to consumers are the displayed finite BHist packet, its hsame transport, its Cont replay, its Pkg provenance, and its local NameCert row. This is exactly the scoped kernel claimed. □

Theorem 1206.6 (ConsciousUniverseSpace public export)**.** *Public consumers of an accepted* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *packet may read only the locality, time, trace-identity, and universe-slice rows already named by the scoped packet. Concretely, a public read exposes the universe-slice row U, local-time row T, locality-cell graph row S, trace-identity row I, and the structural rows*

H, C, P, N from Definition 1206.1. It exports no new subject row, physical substrate row, global universe coordinate, owner identity, or proof-witness row.

Proof. Use the scoped-kernel witness. It confines every locality, time, trace-identity, and universe-slice consumer to the displayed rows $L, U, T, S, I, H, C, P, N$ and to the BEDC primitives BHist, hsame, Cont, Pkg, and NameCert.

The public export only forgets the internal orientation of the scoped packet. It lets a consumer read the rows U, T, S, I and their transport, replay, provenance, and naming data, but it has no additional coordinate from which a subject, substrate, global universe, owner identity, or proof witness could be supplied. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConsciousUniverseSpace}^{\uparrow}}$.

Constructive story: The conscious-universe space is generated from a $BHist$ packet that records a finite universe slice, locality-cell geometry, local-time chain, and trace-identity row without adding a subject parameter or physical substrate row. The carrier convention first exposes the conscious-universe address, then reads the locality and trace boundary through $hsame$ so that equivalent histories keep the same space. Continuations may extend the ledger of observations, but Cont cannot introduce a hidden observer, proof witness, or substrate unless the row is part of the packet. The NameCert obligations, locality non-escape, and independence witness therefore present the object as a visible space of conscious-universe rows rather than a metaphysical host domain.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConsciousUniverseSpace}^{\uparrow}}$, bridgedClosed)

Scope closed: The paper bridge is exactly Definition 1206.1, Definition 1206.2, Definition 1206.3, Definition 1206.4, Definition 1206.5, and Definition 1206.6, with every bridge consumer restricted to locality, time, trace-identity, and universe-slice rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConsciousUniverseSpace}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: It does not claim physical realization, subject identity, global universe completeness, or proof-witness adequacy outside the named carrier rows.

Upgrade path: Machine bridge checking requires a standard bridge theorem that preserves the same locality, time, trace-identity, and universe-slice export without adding subject, substrate, physical-realization, or proof-witness rows.

Chapter 1207

A Concrete Naming Certificate for $\mathsf{SelectedTailSeed}^{\uparrow}$

The $\mathsf{SelectedTailSeed}^{\uparrow}$ packet realizes the stationary selected-tail seed exactness surface of chapter 9238 as a finite concrete NameCert carrier. It is structurally composite over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FiniteTailFilter}^{\uparrow}$: rational endpoints are read through chapter 37; stream windows through chapter 126; regular-rational readback through chapter 301; dyadic tolerance ledgers through chapter 291; Real sealing through chapter 38; and selected finite-tail filtering through chapter 1119. The packet supplies the finite handoff used by Definition 9238.11 without quotient streams, host real equality, or ambient completeness.

Definition 1207.1 (Selected-tail seed carrier)**.** *A* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, E, A, W, K, S, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *seed-rational row naming the stationary rational endpoint.*
2. *E is the endpoint-exactness row comparing the selected seed endpoint with the rational seed under the displayed rational classifier.*
3. *A is the admission row for the selected finite tail, read against* $\mathsf{FiniteTailFilter}^{\uparrow}$*.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window exposing the selected stationary tail segment.*
5. *K is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the stationary regular-rational sequence at the selected window.*
6. *S is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger attached to the selected endpoint.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row obtained from the selected stationary readback.*
8. *H records componentwise* hsame *transport for $Q, E, A, W, K, S, R, P, N$.*
9. *C records displayed* Cont *replay from rational seed through selected finite tail, stream window, regular-rational readback, dyadic ledger, and Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SelectedTailSeed}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ *row.*

The carrier exports no quotient stream, host real equality, countable-choice selector, detached limit object, or ambient Real-completeness coordinate.

Theorem 1207.2 (Selected-tail seed NameCert obligations)**.** *For the carrier of Definition 1207.1, the local* $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^\uparrow}$ *surface consists exactly of carrier admission for* $T = (Q, E, A, W, K, S, R, H, C, P,$ *rational seed admission through* Q*, endpoint exactness through* E*, finite-tail admission through* A*, stationary stream-window exposure through* W*, regular-rational readback through* K*, dyadic tolerance and arithmetic exposure through* S*, Real sealing through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. Its coordinates list the rational seed, endpoint-exactness row, finite-tail admission row, selected stream window, regular-rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local name.

The sibling certificate surfaces account for the data-bearing rows. $\mathsf{Rat}^\uparrow$ supplies Q; $\mathsf{FiniteTailFilter}^\uparrow$ supplies A; $\mathsf{StreamName}^\uparrow$ supplies W; $\mathsf{RegSeqRat}^\uparrow$ supplies K; $\mathsf{DyadicRatCore}^\uparrow$ supplies S; and $\mathsf{Real}^\uparrow$ supplies R.

The endpoint row E is therefore not an additional host equality. It is the displayed comparison tying the selected finite tail back to the stationary rational seed, while H, C, P, N only transport, replay, source, and name the already displayed rows. These rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedTasteGate single carrier alignment`

Theorem 1207.3 (Selected-tail seed Real-window handoff)**.** *Every Real-window consumer of an accepted* $\mathsf{SelectedTailSeed}^\uparrow$ *packet factors through the finite route*

$$Q, E, A, W, K, S, R, H, C, P, N.$$

In particular, the stationary rational selected finite-window endpoint is read as a finite $\mathsf{Rat}^\uparrow$*,* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{Real}^\uparrow$*,* hsame*,* Cont*,* Pkg*, and* NameCert *handoff rather than as quotient stream equality or host real equality.*

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedCarrier real window handoff`

Proof. Start with the selected finite-tail admission row A. It identifies the tail segment whose stationary window is visible, but it does not by itself expose a Real-facing endpoint.

Next read the route through W, K, S. The stream window displays the selected stationary segment, the regular-rational row reads that segment as a regular sequence observation, and the dyadic ledger records the finite tolerance and arithmetic used by the endpoint exactness row.

Finally pass through R, H, C, P, N. The Real seal R is downstream of the selected finite route, and the structural rows only preserve the displayed handoff. Since the carrier has no quotient stream, host real equality, countable-choice selector, detached limit, or ambient completeness coordinate, every Real-window consumer remains inside the finite route. □

Theorem 1207.4 (Selected-tail finite-tail admission exactness)**.** *For every accepted carrier* $T = (Q, E, A, W, K, S, R, H, C, P, N)$ *of Definition 1207.1, the admission row* A *is the unique displayed gate through which the selected stationary tail can be read before any* $\mathsf{StreamName}^\uparrow$ *window or* $\mathsf{RegSeqRat}^\uparrow$ *readback is consumed. Thus every use of* W *or* K *in a* $\mathsf{SelectedTailSeed}^\uparrow$ *consumer first factors through the* $\mathsf{FiniteTailFilter}^\uparrow$ *admission witnessed by* A*.*

Proof. Project the accepted carrier to its displayed rows. The row A is the only coordinate whose type names selected finite-tail admission; the later rows W and K name a stream window and a regular-rational readback, but neither row carries the finite-tail classifier by itself.

The constructive factorization is therefore obtained by ordering the projections. A consumer asking for W or K first reads A, obtaining the $\mathsf{FiniteTailFilter}^{\uparrow}$-admitted selected tail segment, and then passes the admitted segment to the $\mathsf{StreamName}^{\uparrow}$ window row and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row.

No omitted coordinate can bypass this order. The carrier exports no quotient stream, detached limit object, countable-choice selector, or host real equality, so there is no second admission path from a raw tail to W or K outside the displayed finite-tail row. □

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedCarrier finite tail admission exactness`

Theorem 1207.5 (Selected-tail endpoint exactness readback stability)**.** *For an accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$, the endpoint exactness row E is stable under* hsame *transport and displayed* Cont *replay only as part of the full readback surface $Q, E, A, W, K, S, H, C, P, N$. In particular, E cannot be replayed as an isolated host equality detached from the rational seed Q, the window and readback rows W, K, the dyadic ledger S, and the transport, replay, provenance, and naming rows H, C, P, N.*

Proof. Begin with the comparison encoded by E. It relates the selected seed endpoint to the rational seed, so the row already depends on Q before any stability claim can be read.

The readback side is then supplied by W, K, S. The stream row displays the selected stationary window, the regular-rational row reads that window, and the dyadic ledger records the finite tolerance and arithmetic against which endpoint exactness is checked.

Stability is the final structural step. The row H transports the displayed coordinates componentwise by hsame, the row C replays the displayed construction by Cont, and the rows P, N keep the replay sourced and named inside the local $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ packet. Since none of these rows exports a host equality coordinate, the stable endpoint exactness is exactly the displayed finite readback surface. □

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedCarrier endpoint exactness readback stability`

Theorem 1207.6 (Selected-tail Real-seal nonescape obligation)**.** *For an accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$, the Real seal row R can be consumed only after the* $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows Q, A, W, K, S have been displayed, with endpoint exactness supplied by E and structural replay supplied by H, C, P, N. Consequently a* $\mathsf{SelectedTailSeed}^{\uparrow}$ *Real-window consumer has no route through quotient streams, host real equality, countable choice, detached limits, or ambient Real completeness.*

Proof. Project the Real-facing row R. By the carrier definition, R is obtained from the selected stationary readback, so it sits downstream of the rational seed, finite-tail admission, stream window, regular-rational readback, and dyadic tolerance ledger.

The nonescape route is the displayed composition $Q, A, W, K, S, E, R, H, C, P, N$. The rows Q, A, W, K, S expose the finite data needed for the Real seal, E ties the selected endpoint to the rational seed, and H, C, P, N preserve, replay, source, and name the constructed route.

The exclusions are therefore structural, not analytic. Quotient streams, host real equality, countable choice, detached limits, and ambient Real completeness are not coordinates of T, and a consumer of R can only project and compose the rows that the finite $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ carrier displays. □

Theorem 1207.7 (Selected-tail seed Real-seal handoff)**.** *For an accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P,$ of Definition 1207.1, every access to the* $\mathsf{Real}^{\uparrow}$ *seal row R factors through the finite route*

$$Q, E, A, W, K, S, H, C, P, N.$$

Thus the selected stationary endpoint reaches the Real seal only through the displayed rational seed, endpoint exactness, finite-tail admission, stream window, regular-rational readback, dyadic ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ *naming, with no quotient stream or ambient-completeness coordinate.*

Proof. Use the carrier displayed in Definition 1207.1. Its rows place R downstream of Q, E, A, W, K, S, while H, C, P, N are the structural rows that transport, replay, source, and name the same finite data.

The local obligation theorem Definition 1207.2 identifies these rows as the whole $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ surface. The Real-window handoff theorem Definition 1207.3 then gives the consumer route through the same finite packet.

The sibling chapters named in the introduction supply the row meanings: $\mathsf{Rat}^{\uparrow}$ for Q, $\mathsf{FiniteTailFilter}^{\uparrow}$ for A, $\mathsf{StreamName}^{\uparrow}$ for W, $\mathsf{RegSeqRat}^{\uparrow}$ for K, $\mathsf{DyadicRatCore}^{\uparrow}$ for S, and $\mathsf{Real}^{\uparrow}$ for R. Since quotient streams and ambient completeness are not coordinates of T, the Real seal has no other entry path. □

Theorem 1207.8 (Selected-tail seed classifier transport)**.** *For accepted carriers $T = (Q, E, A, W, K, S, R, H, C, P,$ and $T' = (Q', E', A', W', K', S', R', H', C', P', N')$ of Definition 1207.1, componentwise* hsame *transport of $Q, E, A, W, K, S, R, H, C, P, N$ to $Q', E', A', W', K', S', R', H', C', P', N'$ preserves the selected-tail classifier exactly as a displayed* $\mathsf{SelectedTailSeed}^{\uparrow}$ *classifier. It does not collapse endpoint exactness E into host equality of Real endpoints or erase the finite-tail, stream-window, regular-rational, and dyadic rows needed to read the classifier.*

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedCarrier classifier transport`

Proof. Read the transport row H in the carrier of Definition 1207.1. It is componentwise transport for the displayed rows, so each selected-tail coordinate is carried with its own classifier role rather than through a single host equality.

The obligations in Definition 1207.2 show which classifier pieces must remain visible: rational seed admission, endpoint exactness, finite-tail admission, stream-window exposure, regular-rational readback, dyadic ledger, Real seal, replay, provenance, and naming. Transport that preserves the $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ surface must therefore preserve all these rows.

Finally apply the Real-window handoff of Definition 1207.3. Because every Real-facing consumer still passes through $Q, E, A, W, K, S, R, H, C, P, N$, endpoint exactness cannot be reinterpreted as host Real equality, and classifier transport remains a finite selected-tail transport. □

Theorem 1207.9 (Selected-tail seed window readback coherence)**.** *For every accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$ of Definition 1207.1, the stream window W, regular-rational readback K, dyadic ledger S, and Real seal R form one coherent finite readback chain inside the* $\mathsf{SelectedTailSeed}^{\uparrow}$ *packet. Any consumer of R must read W through K and S, while H, C, P, N preserve, replay, source, and name that same chain.*

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedWindowReadbackCoherence`

Proof. Begin with W. The introduction grounds W in $\mathsf{StreamName}^{\uparrow}$, and Definition 1207.1 makes it the selected stationary tail window rather than a quotient stream.

Next read K and S. The $\mathsf{RegSeqRat}^{\uparrow}$ row K reads the selected window as regular-rational data, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row S records the finite tolerance and arithmetic ledger for that readback.

The Real row R is then the seal obtained from this selected stationary readback. By Definition 1207.2 these rows are part of the same local NameCert surface, and by Definition 1207.3 every Real-window consumer remains on the finite route. The structural rows H, C, P, N do not create a second readback; they preserve the single chain. □

Theorem 1207.10 (Selected-tail seed tail-window determinacy)**.** *For every accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$ of Definition 1207.1, any consumer route that exposes the selected stationary endpoint as a $\mathsf{Real}^{\uparrow}$ seal from the same finite-tail admission row A must use the displayed chain A, W, K, S, R, with endpoint exactness supplied by E and structural rows supplied by H, C, P, N. No alternate stream window, regular-rational readback, dyadic ledger, or Real seal is a $\mathsf{SelectedTailSeed}^{\uparrow}$ consumer route unless it is supplied as one of these carrier rows.*

Proof. Project the carrier T. The finite-tail admission row is the displayed coordinate A, and Definition 1207.4 makes the selected endpoint pass through that finite admission gate with endpoint exactness E.

Next apply Definition 1207.9. Once A is fixed, the consumer-visible readback is the single window chain W, K, S, R: W exposes the selected stream window, K reads it as regular-rational data, S records the dyadic ledger, and R seals the same selected stationary endpoint as a Real row.

Finally apply Definition 1207.7. The structural rows H, C, P, N preserve, replay, source, and name the displayed chain, while quotient streams, host real equality, detached limits, and ambient Real completeness are not coordinates of the carrier. Hence a route using a different window, readback, ledger, or seal is not a $\mathsf{SelectedTailSeed}^{\uparrow}$ consumer route unless that data has already been supplied as one of the rows of T. □

Theorem 1207.11 (Selected-tail seed stationary-endpoint coverage)**.** *For every accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$ of Definition 1207.1, the stationary selected endpoint is exposed only through the finite route*

$$Q, E, A, W, K, S$$

before the Real seal row R is consumed. These rows are sufficient for the local selected-tail endpoint read used by Real-window consumers: Q supplies the rational seed, E supplies endpoint exactness, A admits the selected finite tail, W exposes the stationary window, K reads that window as regular-rational data, and S records the dyadic tolerance ledger.

Proof. Project the carrier and apply Definition 1207.2. The displayed NameCert surface contains exactly the seed, exactness, finite-tail admission, stream-window, regular-rational, dyadic-ledger, and Real-seal rows together with the structural rows.

The local endpoint read begins before the Real seal. By Definition 1207.4, the selected stationary tail must first pass through A before any use of W or K.

Finally apply Definition 1207.9. The window-readback chain is W, K, S, R, so the endpoint visible to Real-window consumers has already been routed through Q, E, A, W, K, S before the seal row R is read. □

Theorem 1207.12 (Selected-tail seed dyadic-ledger readback coverage)**.** *For every accepted carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$ of Definition 1207.1, the row S is the unique displayed*

dyadic tolerance and arithmetic ledger mediating the regular-rational readback K and the Real seal R. No $\mathsf{SelectedTailSeed}^{\uparrow}$ *consumer may pass from W, K to R while omitting S.*

Proof. Project T to its displayed rows. The carrier definition names S as the only $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and arithmetic ledger attached to the selected endpoint, while K is the regular-rational readback row and R is the Real seal row.

Apply Definition 1207.2. The local NameCert surface lists K, S, and R as separate required obligations, so the dyadic ledger cannot be absorbed into either the readback row or the Real seal.

The Real-facing route is fixed by Definition 1207.6 and Definition 1207.7. Both routes pass through S before consuming R, and the carrier exports no alternate coordinate that could mediate the same transition. □

Theorem 1207.13 (Selected-tail seed Real-completion boundary)**.** *Every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier supplies a finite handoff into Real-window consumers, but it does not assert Real completeness, quotient-stream equality, countable-choice selection, detached limits, or host real equality. Any consumer that needs one of those completion-facing boundaries must obtain it from a separate L10 chapter rather than from the local rows $Q, E, A, W, K, S, R, H, C, P, N$.*

Proof. First use Definition 1207.11. The stationary endpoint exposed by the packet is the finite route Q, E, A, W, K, S before the Real seal row is read.

Next use Definition 1207.12. The passage from regular-rational readback to Real seal must include the displayed dyadic ledger S, so the packet supplies a concrete finite handoff rather than an ambient completion principle.

Finally combine Definition 1207.3 with Definition 1207.7. Both the consumer route and the seal route remain inside the listed carrier rows, whose definition explicitly exports no quotient stream, host real equality, countable-choice selector, detached limit object, or ambient Real-completeness coordinate. □

Theorem 1207.14 (Selected-tail rational seed admission)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$, the rows Q and E are the only rational seed and endpoint-exactness admission rows in the local packet. The rational seed enters through Q, endpoint exactness is witnessed through E, and neither role may be replaced by host* $\mathsf{Real}^{\uparrow}$ *equality or by a detached completion-facing comparison.*

Proof. Project the accepted carrier. By Definition 1207.1, Q is the unique displayed $\mathsf{Rat}^{\uparrow}$ seed-rational row, and E is the displayed endpoint-exactness row comparing the selected seed endpoint with that rational seed.

Apply Definition 1207.2. The local $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ surface lists rational seed admission through Q and endpoint exactness through E as separate obligations, while the other rows serve finite-tail admission, stream readback, dyadic accounting, Real sealing, transport, replay, provenance, and naming.

Finally use Definition 1207.13. The packet does not export host real equality, quotient streams, detached limits, countable-choice selection, or ambient Real completeness, so no completion-facing equality can replace the displayed Q, E admission plane. □

Theorem 1207.15 (Selected-tail stream–regular–dyadic compatibility)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier $T = (Q, E, A, W, K, S, R, H, C, P, N)$, the rows W, K, S form the unique displayed compatibility chain from* $\mathsf{StreamName}^{\uparrow}$ *through* $\mathsf{RegSeqRat}^{\uparrow}$ *to* $\mathsf{DyadicRatCore}^{\uparrow}$ *before the Real seal row R is consumed. Thus a Real-window consumer cannot read R without first passing through the stream window, regular-rational readback, and dyadic ledger displayed by W, K, S.*

Proof. Start with the readback chain. Definition 1207.9 identifies W, K, S, R as one coherent finite readback chain inside the accepted packet.

The dyadic step is not optional. By Definition 1207.12, the row S is the unique displayed dyadic tolerance and arithmetic ledger mediating the regular-rational readback K and the Real seal R.

The Real-facing boundary then fixes the order of consumption. Definition 1207.7 and Definition 1207.13 keep the consumer route inside the finite carrier rows, so the path to R is exactly the displayed $\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow}$ compatibility chain W, K, S. □

Theorem 1207.16 (Selected-tail package provenance exactness)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier* $T = (Q, E, A, W, K, S, R, H, C, P, N)$*, the rows* P *and* N *name exactly the finite route*

$$Q, E, A, W, K, S, R, H, C.$$

They cannot add quotient streams, countable-choice selectors, detached limits, ambient completeness, or host equality beyond the displayed selected-tail seed packet.

Proof. Inventory the carrier rows. The data-bearing route is Q, E, A, W, K, S, R, and the structural route is H, C; by Definition 1207.1, P is the Pkg provenance over these displayed ingredients and N is the local $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ row.

Apply Definition 1207.2. The provenance and naming rows source and name the existing route; they are not independent coordinates carrying analytic completion data.

Finally apply Definition 1207.13. Since the packet exports no quotient stream, host real equality, countable-choice selector, detached limit object, or ambient Real-completeness coordinate, P, N preserve provenance exactly for $Q, E, A, W, K, S, R, H, C$ and cannot enlarge the route. □

Theorem 1207.17 (Selected-tail seed dependency readiness triad)**.** *Every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier displays a root-ready dependency triad. The rational endpoint plane is* Q, E*; the finite-tail, window, and readback plane is* A, W, K*; and the dyadic Real-seal plane is* S, R*. The structural rows* H, C, P, N *preserve the same packet by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$ *row.*

Proof. Project the accepted carrier as $T = (Q, E, A, W, K, S, R, H, C, P, N)$. The carrier definition already displays the six dependency rows named in the chapter introduction: $\mathsf{Rat}^{\uparrow}$ supplies the rational seed row Q, $\mathsf{FiniteTailFilter}^{\uparrow}$ supplies the selected finite-tail admission row A, $\mathsf{StreamName}^{\uparrow}$ supplies the window row W, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the readback row K, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the dyadic ledger row S, and $\mathsf{Real}^{\uparrow}$ supplies the seal row R.

The row E binds the rational endpoint plane by comparing the selected seed endpoint with the rational seed under the displayed rational classifier. Thus Q, E form the rational endpoint plane, while A, W, K form the finite-tail, window, and readback plane.

Finally use Definition 1207.2 and Definition 1207.3. The rows S, R form the dyadic Real-seal plane, and H, C, P, N do not create another carrier. They transport, replay, source, and name the same finite packet, so the displayed triad is root-ready without quotient streams, host real equality, or ambient completeness. □

Theorem 1207.18 (Selected-tail seed StreamName–RegSeqRat scope)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier* $T = (Q, E, A, W, K, S, R, H, C, P, N)$*, the readback surface* W, K *depends only on the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of the packet. It does not depend on quotient-stream equality, host real equality, detached limits, countable choice, or ambient Real completeness.*

Proof. First project the window and readback coordinates. By Definition 1207.1, W is the $\mathsf{StreamName}^{\uparrow}$ window exposing the selected stationary tail segment, and K is the $\mathsf{RegSeqRat}^{\uparrow}$ readback row for the stationary regular-rational sequence at that window.

Next use Definition 1207.9. The readback chain consumes W through K inside the same finite carrier, while H, C, P, N preserve, replay, source, and name that chain through the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface.

The nonescape claim follows from Definition 1207.13. The packet exports no quotient-stream equality, host real equality, detached limit, countable-choice selector, or ambient Real-completeness coordinate, so the W, K readback scope is exactly the displayed StreamName–RegSeqRat scope. □

Theorem 1207.19 (Selected-tail seed dyadic Real-seal scope)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier* $T = (Q, E, A, W, K, S, R, H, C, P, N)$*, the rows* S, R *route through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *as a finite seal surface. This route uses the displayed dyadic tolerance ledger and Real seal only; it does not use quotient streams, host real equality, detached limits, countable choice, or ambient completeness.*

Proof. Project S and R from the accepted carrier. The carrier definition names S as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and arithmetic ledger attached to the selected endpoint, and it names R as the $\mathsf{Real}^{\uparrow}$ seal row obtained from the selected stationary readback.

Apply Definition 1207.12. The passage from regular-rational readback to Real seal cannot omit S, so the Real-facing route passes through the displayed dyadic ledger before consuming R.

Finally apply Definition 1207.7 and Definition 1207.13. The seal route remains a finite carrier route through S, R and the structural rows that preserve the same packet. Since the carrier has no quotient stream, host real equality, detached limit, countable-choice selector, or ambient-completeness coordinate, the dyadic Real-seal scope is exactly the displayed $\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ scope. □

Theorem 1207.20 (Selected-tail seed finite-tail Real-boundary scope)**.** *For every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier* $T = (Q, E, A, W, K, S, R, H, C, P, N)$*, the tuple* A, W, K, S, R *is the exact finite-tail-to-Real boundary consumed downstream. The finite-tail admission* A*, stream window* W*, regular-rational readback* K*, dyadic ledger* S*, and Real seal* R *are the consumer-visible route from selected finite tail to Real-facing handoff.*

Proof. Begin with the finite-tail side. By Definition 1207.4, every use of W or K first factors through the $\mathsf{FiniteTailFilter}^{\uparrow}$ admission witnessed by A.

Then use Definition 1207.9. The accepted readback chain is W, K, S, R: the window exposes the selected stationary segment, the regular-rational row reads it, the dyadic ledger records the finite tolerance and arithmetic, and the Real row seals the same selected endpoint.

Finally apply Definition 1207.13. Downstream consumers receive a finite handoff and no completion-facing principle from this chapter. Therefore A, W, K, S, R is the exact finite-tail-to-Real boundary supplied by $\mathsf{SelectedTailSeed}^{\uparrow}$. □

This chapter is composite rather than atomic. Its independence from the sibling chapters is not expressed by row-by-row atomic separation: $\mathsf{SelectedTailSeed}^{\uparrow}$ deliberately joins $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FiniteTailFilter}^{\uparrow}$ into one stationary selected-tail seed packet, and each sibling omits at least one coordinate needed for the full Real-window handoff.

The Lean `TasteGate.lean` boundary for $\mathsf{SelectedTailSeed}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial}\ \mathsf{SelectedTailSeed}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{SelectedTailSeed}^{\uparrow}$ instances preserving $Q, E, A, W, K, S, R, H, C, P, N$.

Falsifiable prediction: If the finite-tail admission row A or the Real seal row R is absent, no accepted $\mathsf{SelectedTailSeed}^{\uparrow}$ packet can expose the stationary selected-tail endpoint as a Real-window handoff.

Theorem 1207.21 (Selected-tail seed public Real-window export)**.** *Every accepted* $\mathsf{SelectedTailSeed}^{\uparrow}$ *carrier exports a public Real-window handoff through the scoped rows* $Q, E, A, W, K, S, R, H, C, P, N$*. The export is the finite selected-tail route from rational seed and endpoint exactness through finite-tail admission, stream window, regular-rational readback, dyadic ledger, Real seal, componentwise history transport, displayed continuity replay, package provenance, and local name certificate. It excludes quotient streams, host real equality, countable choice, detached limits, and ambient Real completeness.*

Proof. Project the accepted carrier as $T = (Q, E, A, W, K, S, R, H, C, P, N)$. By Definition 1207.17, the rows split into the rational endpoint plane Q, E, the finite-tail readback plane A, W, K, the dyadic Real-seal plane S, R, and the structural preservation rows H, C, P, N.

Next apply Definition 1207.18. The selected stationary window and regular-rational readback are already scoped by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the handoff has a public readback surface without quotient-stream equality or host real equality.

Apply Definition 1207.19. The same route then passes through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger and $\mathsf{Real}^{\uparrow}$ seal, so the L10 source chain remains finite and carrier-internal.

Finally use Definition 1207.20. Downstream consumers receive exactly the boundary A, W, K, S, R, while Q, E, H, C, P, N give the seed, comparison, transport, replay, provenance, and naming context for that boundary. Since the cited scope theorems rule out quotient streams, host real equality, countable choice, detached limits, and ambient Real completeness, this is the public Real-window export promised by the accepted $\mathsf{SelectedTailSeed}^{\uparrow}$ carrier. □

Theorem 1207.22 (Selected-tail seed Real-completion bridge)**.** *The public Real-window export of Definition 1207.21 is exactly the finite bridge route consumed by the stationary rational selected-tail seed exactness surface of Definition 9238.11. The route passes through*

$$Q, E, A, W, K, S, R, H, C, P, N$$

and its sibling dependencies are precisely $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{FiniteTailFilter}^{\uparrow}$*. It excludes quotient streams, host real equality, countable choice, detached limits, and ambient Real completeness.*

Proof. Start with the public export. By Definition 1207.21, every accepted carrier already presents the route $Q, E, A, W, K, S, R, H, C, P, N$ as a public Real-window handoff, with the rational seed and endpoint comparison in Q, E, the selected finite-tail readback in A, W, K, the dyadic Real-seal surface in S, R, and the structural rows in H, C, P, N.

The admission side is fixed by Definition 1207.4. No consumer can reach the selected stationary stream window or regular-rational readback before passing through the displayed $\mathsf{FiniteTailFilter}^{\uparrow}$ row A, so the bridge cannot insert an alternate raw-tail or detached-limit path.

Next use Definition 1207.15. The rows W, K, S form the required $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$ chain before the Real seal is read, while Definition 1207.19 keeps the seal route inside the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

Finally apply Definition 1207.7 and Definition 1207.13. These results hand the selected endpoint to R without adding quotient streams, host real equality, countable choice, detached limits, or ambient Real completeness. Thus the exactness surface of Definition 9238.11 consumes precisely the same finite route exported by Definition 1207.21. □

Theorem 1207.23 (Selected-tail seed mature Real-completion package)**.** *The public export, finite-tail admission exactness,* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$ *compatibility, dyadic Real-seal scope, and Real-completion bridge together form the mature finite selected-tail package over* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FiniteTailFilter}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package excludes quotient streams, host real equality, countable choice, detached limit selection, and ambient Real completeness.*

Proof. Apply Definition 1207.21. It provides the public route through $Q, E, A, W, K, S, R, H, C, P, N$ and already records the exclusion of quotient streams, host real equality, countable choice, detached limits, and ambient Real completeness.

The finite-tail and sibling compatibility rows are fixed by Definition 1207.4 and Definition 1207.15. These results bind the $\mathsf{FiniteTailFilter}^{\uparrow}$ admission and the $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$ readback chain inside the accepted carrier.

Finally use Definition 1207.19 and Definition 1207.22. The Real-facing seal remains the displayed $\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ finite route, and the bridge consumed by the stationary rational selected-tail exactness surface is precisely the same route. Thus the cited rows constitute the mature finite selected-tail Real-completion package, with no additional quotient, host-equality, choice, detached-limit, or ambient-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SelectedTailSeed}^{\uparrow}$ packet is generated from a rational seed row, selected endpoint exactness, finite-tail admission, stream window, regular-rational readback, dyadic tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1207.17, Definition 1207.18, Definition 1207.19, Definition 1207.20, Definition 1207.21, Definition 1207.22, and Definition 1207.23 bind the mature finite selected-tail Real-completion package over $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SelectedTailSeed}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SelectedTailSeedUp.SelectedTailSeedTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Lean4 ✓ `BEDC.Derived.SelectedTailSeedUp.taste gate`

Not claimed: This chapter does not close bridge checking, quotient streams, host real equality, countable choice, detached limit selection, or ambient Real completeness.

Upgrade path: Further verification requires bridge checking for the selected-tail Real-completion route without quotient streams, host real equality, countable choice, detached limits, or ambient Real completeness.

Chapter 1208

A Concrete Naming Certificate for $\mathsf{UnitCalibrationLedger}^{\uparrow}$

The $\mathsf{UnitCalibrationLedger}^{\uparrow}$ packet realizes the empirical-value discipline of chapter 9462 as a finite NameCert surface for measured numerical constants. It sits beside the law-certificate packet of chapter 1155, the physical-model audit packet of chapter 1129, the certified physical-truth packet of chapter 1130, and the terminal unit surface of chapter 115. Those siblings supply law patterns, audit comparisons, truth-certificate reads, and terminal unit naming; this packet binds numerical measurement, unit conversion, calibration, uncertainty, instrument, reproducibility, dimension, classifier, transport, provenance, and local naming before any consumer may read a value as empirical.

Definition 1208.1 (Unit calibration ledger carrier)**.** *A* $\mathsf{UnitCalibrationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (M, U, C, E, I, R, D, K, H, P, N)$$

with the following displayed rows:

1. *M is the measurement-record row, naming the finite observation, numerical readout, sampling context, and protocol by which the value was obtained.*
2. *U is the unit-bridge row, recording the source unit, target standard unit, conversion route, and local unit certificate read through chapter 115.*
3. *C is the calibration row, naming the calibration source, calibration state, reference route, and validity scope for the measurement instrument.*
4. *E is the uncertainty row, recording error bars, tolerance, resolution, rounding discipline, and any admitted confidence or interval convention.*
5. *I is the instrument-ledger row, naming the device, sensor, apparatus, or procedure whose output supplies M.*
6. *R is the reproducibility-ledger row, recording repeated runs, independent checks, or declared replication boundary for the same empirical value.*
7. *D is the dimension row, recording the physical dimension and the bridge between dimension-bearing syntax and the unit bridge U.*
8. *K is the classifier row comparing empirical-value packets componentwise across M, U, C, E, I, R, D and rejecting value-only equality.*

9. *H records componentwise* hsame *transport for measurement, unit, calibration, uncertainty, instrument, reproducibility, dimension, classifier, provenance, and local-name rows.*

10. *P is the* Pkg *provenance over* UnitCalibrationLedger$^{\uparrow}$, LawCertificate$^{\uparrow}$, PhysicalModelAudit$^{\uparrow}$, CertifiedPhysicalTruth Unit$^{\uparrow}$, RealityConstrainedModelErrorBudget$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare real number, detached constant symbol, unit-free magnitude, uncalibrated measurement, uncertainty-free export, anonymous instrument, unreproducible read, or dimensionless empirical shortcut outside the displayed packet.

Lean4 (definition) `BEDC.Derived.UnitCalibrationLedgerUp.UnitCalibrationLedgerUp`
Lean4 (definition) `BEDC.Derived.UnitCalibrationLedgerUp.taste gate`
Lean4 ✓ `BEDC.Derived.UnitCalibrationLedgerUp.UnitCalibrationLedgerTasteGate single carrier alignment`

Definition 1208.2 (Unit calibration ledger classifier)**.** *The* UnitCalibrationLedger$^{\uparrow}$ *classifier identifies two accepted packets only when their measurement-record rows, unit-bridge rows, calibration rows, uncertainty rows, instrument-ledger rows, reproducibility-ledger rows, dimension rows, classifier rows, componentwise transports, provenance rows, and local NameCert rows agree through displayed* hsame *transport. Agreement of the printed numeral alone is not enough, and agreement of the unit row without calibration, uncertainty, instrument, and reproducibility rows is not enough.*

Theorem 1208.3 (Unit calibration ledger NameCert obligations)**.** *For the carrier of Definition 1208.1, the local* $\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}$ *surface consists exactly of five empirical ledger obligations:*

1. *measurement admission through M, requiring the numerical readout to be tied to a finite observation record and protocol;*

2. *unit and dimension admission through U and D, requiring every numerical read to carry its unit bridge and dimension row;*

3. *calibration admission through C, requiring the measuring route to expose its calibration source and validity scope;*

4. *uncertainty admission through E, requiring the value to expose its tolerance, resolution, or interval convention;*

5. *instrument and reproducibility admission through I and R, requiring both the generating apparatus and the replication boundary to be part of the same packet.*

The structural rows K, H, P, N compare, transport, source, and locally name those five obligations, but they do not add another empirical-value coordinate.

Proof. Carrier admission is projection from Definition 1208.1. The packet lists the measurement record, unit bridge, calibration row, uncertainty row, instrument ledger, reproducibility ledger, dimension row, classifier, transport, provenance, and local name as a single finite surface.

The five obligation rows group these displayed coordinates by empirical function. Measurement admission comes from M; unit and dimension admission come from U and D; calibration admission comes from C; uncertainty admission comes from E; instrument and reproducibility admission come from I and R.

The classifier and structural rows preserve only the same displayed packet. Since the carrier has no coordinate for a detached numeral, hidden unit convention, omitted calibration, suppressed uncertainty, anonymous instrument, or unrecorded replication boundary, the five obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.UnitCalibrationLedgerUp.UnitCalibrationLedgerCarrier namecert obligations`

Theorem 1208.4 (Unit calibration ledger numeric-value nonescape)**.** *An accepted* $\mathsf{UnitCalibrationLedger}^{\uparrow}$ *packet cannot export a numerical empirical value unless the unit bridge* U*, calibration row* C*, uncertainty row* E*, instrument ledger* I*, and reproducibility ledger* R *are present in the same finite carrier. Dropping any one of those rows leaves the accepted empirical-value surface.*

Proof. Read the carrier as $V = (M, U, C, E, I, R, D, K, H, P, N)$. The measurement row M records a finite numerical readout, but the packet treats that readout as empirical only after the unit bridge, calibration row, uncertainty row, instrument ledger, and reproducibility ledger are displayed beside it.

Apply the obligation surface of Definition 1208.3. Unit and dimension admission require U and D, calibration admission requires C, uncertainty admission requires E, and instrument-reproducibility admission requires I and R.

Finally use the classifier of Definition 1208.2. It compares accepted packets componentwise and does not identify two packets by their printed numerals alone. Therefore a value export lacking U, C, E, I, or R is outside the local $\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}$ carrier. □

Lean4 ✓ `BEDC.Derived.UnitCalibrationLedgerUp.UnitCalibrationLedgerNumericValueNonescape`

Theorem 1208.5 (Unit calibration ledger law-certificate handoff)**.** *Every* $\mathsf{LawCertificate}^{\uparrow}$*,* $\mathsf{PhysicalModelAudit}^{\uparrow}$*, or* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$ *consumer read of a numerical empirical constant factors through a displayed* $\mathsf{UnitCalibrationLedger}^{\uparrow}$ *packet. The consumer may use the numerical row only together with its unit bridge, calibration row, uncertainty row, instrument ledger, reproducibility ledger, dimension row, classifier, transport, provenance, and local NameCert row.*

Proof. The sibling consumers have distinct public roles. chapter 1155 reads equation-like law patterns, chapter 1129 reads physical-model audit comparisons, and chapter 1130 reads certified physical-truth packets.

None of those roles supplies a complete empirical-value carrier by itself. A numerical constant used by any of them must therefore be projected through the packet $V = (M, U, C, E, I, R, D, K, H, P, N)$ of Definition 1208.1.

The nonescape theorem Definition 1208.4 blocks value-only export. Hence each consumer receives the numerical row only through the same displayed measurement, unit, calibration, uncertainty, instrument, reproducibility, dimension, classifier, transport, provenance, and local NameCert surface. □

Lean4 ✓ `BEDC.Derived.UnitCalibrationLedgerUp.UnitCalibrationLedgerCarrier law certificate handoff`

Theorem 1208.6 (Unit calibration ledger empirical bridge surface)**.** *Every bridge from* $\mathsf{UnitCalibrationLedger}^{\uparrow}$ *to* $\mathsf{LawCertificate}^{\uparrow}$*,* $\mathsf{PhysicalModelAudit}^{\uparrow}$*,* $\mathsf{CertifiedPhysicalTruth}^{\uparrow}$*, or* $\mathsf{Unit}^{\uparrow}$ *preserves the displayed rows*

$$M, U, C, E, I, R, D, K, H, P, N$$

of Definition 1208.1. No such bridge may read a bare $\mathsf{Real}^{\uparrow}$ *value, unit-free magnitude, omitted calibration row, suppressed uncertainty row, anonymous instrument row, or unrecorded reproducibility row as an empirical-value packet.*

Proof. Project the carrier of Definition 1208.1. The bridge source is the finite packet $V = (M, U, C, E, I, R, D, K, H, P, N)$, so the measurement, unit, calibration, uncertainty, instrument, reproducibility, dimension, classifier, transport, provenance, and local naming rows are all visible before any target sibling is read.

Apply the NameCert obligations of Definition 1208.3. They force measurement admission through M, unit and dimension admission through U, D, calibration admission through C, uncertainty admission through E, instrument and reproducibility admission through I, R, and structural preservation through K, H, P, N.

Next apply numeric-value nonescape from Definition 1208.4. A value-only export leaves the accepted empirical packet, so a bridge cannot replace the carrier by a bare $\mathsf{Real}^{\uparrow}$ value, unit-free magnitude, missing calibration, missing uncertainty, anonymous instrument, or missing reproducibility boundary.

Finally route through the law-certificate handoff of Definition 1208.5. The law-certificate, physical-model audit, certified physical-truth, and terminal unit consumers receive the numerical row only together with the same displayed empirical-value surface, so the bridge preserves $M, U, C, E, I, R, D, K, H, P, N$. □

The Lean follow-up file `lean4/BEDC/Derived/UnitCalibrationLedgerUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ UnitCalibrationLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ UnitCalibrationLedger}^{\uparrow}$ instances, with witness-pair and field-projection proofs preserving $M, U, C, E, I, R, D, K, H, P, N$ as separate displayed fields.

Falsifiable prediction: A numerical empirical-value export lacking any one of unit bridge, calibration, uncertainty, instrument, or reproducibility rows is rejected by the local carrier.

Independence witness: lawcertificate, physicalmodelaudit, certifiedphysicaltruth, unit, realityconstrainedmodelerrorbudget: none is bijective because UnitCalibrationLedgerUp binds numerical measurement, unit conversion, calibration, uncertainty, instrument, reproducibility, dimension, classifier, transport, provenance, and local NameCert rows in one carrier, while each sibling supplies only a law pattern, audit packet, truth surface, terminal unit, or error-budget component.

Closure status for $\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{UnitCalibrationLedger}^{\uparrow}$ packet is generated from finite measurement, unit bridge, calibration, uncertainty, instrument, reproducibility, dimension, classifier, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1208.1, Definition 1208.2, Definition 1208.3, Definition 1208.4, the public export theorem Definition 1208.5, and the empirical bridge surface Definition 1208.6 bind BHist, hsame, Cont, Pkg, NameCert, and the displayed rows $M, U, C, E, I, R, D, K, H, P, N$ as the exported empirical-value handoff and sibling bridge surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnitCalibrationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host-level metrology, all standards bodies, exact experimental replication, or physical truth beyond the displayed empirical-value packet.

Upgrade path: The remaining work is Lean-side verification and bridge checking for the empirical bridge surface of Definition 1208.6.

Chapter 1209

A Concrete Naming Certificate for $\mathsf{ClaimRegistryLayer}^{\uparrow}$

The $\mathsf{ClaimRegistryLayer}^{\uparrow}$ packet realizes the registry layer of section 9462.35 as a finite export-control carrier for philosophy-package claims. It sits beside the unit calibration ledger of chapter 1208 and the external-supply boundary of chapter 1065: those siblings control empirical-value export and external-supply requests, while this packet controls whether a claim is admitted as a package-level exported claim at all.

Definition 1209.1 (Claim registry layer carrier)**.** *A* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, R, B, C, F, V, H, P, N)$$

with the following displayed rows:

1. *E is the claim-entry row, naming the exported philosophy-package claim, its local subject index, and its intended package surface.*
2. *R is the refusal-boundary row, recording the registry entry that blocks claims with no accepted export surface.*
3. *B is the closure-status row, recording the theory-closure and verification axes used by the claim site.*
4. *C is the formal-target row, recording the Lean target, statement-only site, encoded definition, or absence of a formal target for the same claim.*
5. *F is the provenance row, tying the claim to the finite source file, section, carrier, theorem, definition, or cannot-claim row from which it is exported.*
6. *V is the local verification-readback row, recording whether the package claim is read as checked theorem, encoded definition, scoped paper claim, or explicit refusal.*
7. *H records componentwise* hsame *transport for claim entry, refusal boundary, closure status, formal target, provenance, verification readback, package provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{AxisCarryRefusalLedger}^{\uparrow}$, $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$, $\mathsf{ExternalSupplyBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClaimRegistryLayer}^{\uparrow}}$ *row.*

The carrier has no coordinate for an exported package claim outside the displayed entry, refusal boundary, closure status, formal target, provenance, verification readback, transport, package, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.TasteGate.ClaimRegistryLayerTasteGate single carrier alignment`

Theorem 1209.2 (Claim registry layer export control)**.** *Every exported philosophy-package claim admitted by* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *must display the claim-entry row E, refusal-boundary row R, closure-status row B, formal-target row C, provenance row F, verification-readback row V, componentwise transport H, package provenance P, and local naming row N in the same finite carrier. A consumer may cite the claim only through that packet.*

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.ClaimRegistryLayer export control certificate`

Proof. Project the carrier of Definition 1209.1. The exported claim is not just the prose entry E; it is the full packet $Q = (E, R, B, C, F, V, H, P, N)$.

The registry layer of section 9462.35 supplies the export-control reading: a claim must either appear in a registry entry or be blocked by a registry entry. Rows E and R encode those two possibilities inside the same finite packet.

The remaining rows make the export auditable. Closure status B, formal target C, provenance F, and verification readback V identify the claim's status, while H, P, N preserve and name the displayed rows. Since no other export coordinate exists, the consumer can cite the claim only through the packet above. □

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.TasteGate.ClaimRegistryLayer export control certificate`

Theorem 1209.3 (Claim registry layer no unregistered export)**.** *An exported philosophy-package claim with no* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *entry row and no refusal-boundary row is rejected by the carrier. In particular, a prose claim cannot become a package-level BEDC claim merely by appearing near a theorem, definition, closure-status block, or formal target; it must be registered or refused by the same finite export-control surface.*

Proof. Begin with export control, Definition 1209.2. Any admitted package-level export must be read through $Q = (E, R, B, C, F, V, H, P, N)$.

If both the entry row E and refusal-boundary row R are absent, the packet required by Definition 1209.1 is not present. The remaining rows cannot manufacture claim admission because closure status, formal targets, provenance, verification readback, transport, package provenance, and local naming are structural rows for an already displayed entry or refusal.

Thus an unregistered prose claim has no carrier coordinate from which export can be read. It remains outside the package-level claim surface until the registry layer supplies either an entry row or a refusal row. □

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.TasteGate.ClaimRegistryLayer no unregistered export certificate`

The Lean follow-up file `lean4/BEDC/Derived/ClaimRegistryLayerUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{ClaimRegistryLayer}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ClaimRegistryLayer}^{\uparrow}$ instances, preserving $E, R, B, C, F, V, H, P, N$ as separate displayed fields.

Falsifiable prediction: Any exported philosophy-package claim with no registry entry or refusal row is rejected by the $\mathsf{ClaimRegistryLayer}^{\uparrow}$ carrier.

Independence witness: axiscarryrefusalledger, realityconstrainedmethodologyledger, finiteobservationresidueboundary, externalsupplyboundary: none is bijective to $\mathsf{ClaimRegistryLayer}^{\uparrow}$ because those siblings carry refusal, methodology, finite-observation residue, or external-supply boundary rows, while this carrier binds claim entry, refusal boundary, closure status, formal target, provenance, verification readback, transport, package provenance, and local NameCert rows in one export-control packet.

Theorem 1209.4 (Claim registry layer public export)**.** *The public* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *export is exactly the finite packet with claim entry, refusal boundary, closure status, formal target, provenance, verification readback, componentwise transport, package provenance, and local* NameCert *rows. A public consumer of a package-level claim may read only this packet surface, and any claim lacking both a registry entry and a refusal-boundary row remains outside the public export.*

Proof. Apply Definition 1209.2. It identifies every admitted exported claim with the packet $Q = (E, R, B, C, F, V, H, P, N)$, so the public export cannot drop the closure-status, formal-target, provenance, verification-readback, transport, package-provenance, or local naming rows.

Apply Definition 1209.3. A claim with neither an entry row nor a refusal-boundary row has no carrier coordinate from which public export can be read. Thus the public surface is exactly the displayed finite packet and no unregistered prose claim enters it. □

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.ClaimRegistryLayer public export certificate`

Theorem 1209.5 (Claim registry layer bridge route)**.** *The public export packet*

$$Q = (E, R, B, C, F, V, H, P, N)$$

is the bridge-readable $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *surface from claim-entry and refusal-boundary rows to package-level export. A bridge consumer reads the entry row E or refusal row R, then the closure-status, formal-target, provenance, and verification-readback rows B, C, F, V, and finally the transport, package, and local naming rows H, P, N.*

Proof. Project the carrier of Definition 1209.1. It exposes exactly the nine rows $E, R, B, C, F, V, H, P, N$ and no separate package-export coordinate.

Use Definition 1209.4. The public export theorem identifies this finite packet as the complete public surface for package-level claim export.

Finally apply Definition 1209.3. Since a claim without an entry row and without a refusal-boundary row has no public export, every bridge-readable route from claim rows to package export must pass through the displayed packet Q. □

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.TasteGate.ClaimRegistryLayer bridge route certificate`

Theorem 1209.6 (Claim registry layer mature package claim index)**.** *The public export theorem, no-unregistered-export theorem, and bridge route together form the mature package claim index for* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*. A package-level claim is public exactly when it is read through the packet $Q = (E, R, B, C, F, V, H, P, N)$, rejected when both E and R are absent, and bridge-readable only through the same entry, refusal, closure-status, formal-target, provenance, verification-readback, transport, package, and local naming rows.*

Proof. Start with Definition 1209.4. It identifies the public package-claim surface with the displayed packet $Q = (E, R, B, C, F, V, H, P, N)$.

Apply Definition 1209.3. This removes every prose-only package claim with neither entry row nor refusal-boundary row from the index.

Use Definition 1209.5. The bridge-readable route consumes the same packet rows and adds no separate claim coordinate, so the public export, rejection boundary, and bridge route are one complete package claim index. □

Lean4 ✓ `BEDC.Derived.ClaimRegistryLayerUp.ClaimRegistryLayer mature package claim index`

Closure status for $\mathsf{NameCert}_{\mathsf{ClaimRegistryLayer}^{\uparrow}}$.

Constructive story: A $\mathsf{ClaimRegistryLayer}^{\uparrow}$ packet is generated from claim-entry, refusal-boundary, closure-status, formal-target, provenance, verification-readback, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClaimRegistryLayer}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1209.1, Definition 1209.2, Definition 1209.3, Definition 1209.4, Definition 1209.5, Definition 1209.6, the falsifiable prediction, and the independence witness bind the carrier, export-control theorem, no-unregistered-export theorem, public export theorem, bridge route, mature package claim index, and non-bijection surface for package-level claim export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClaimRegistryLayer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, full subject-index governance, or the correctness of every downstream registry row.

Upgrade path: Lean bridge checking remains the formal-axis next step for the public $\mathsf{ClaimRegistryLayer}^{\uparrow}$ export surface and mature package claim index.

Chapter 1210

A Concrete Naming Certificate for $\mathsf{TailModulusAlignment}^{\uparrow}$

The $\mathsf{TailModulusAlignment}^{\uparrow}$ packet realizes the shared tail-modulus route used by the Real-completion surface of chapter 9238. It is structurally composite over the refinement route of chapter 885, the tail schedule of chapter 838, the tail-fusion surface of **??**, the finite windows of chapter 126, the regular-rational readbacks of chapter 301, the dyadic tolerance rows of chapter 291, and the seal rows of chapter 38. The packet names one finite alignment certificate for a shared tail threshold, not a quotient stream, host real equality, selected limit, countable-choice selector, or ambient Real-completeness principle.

Definition 1210.1 (Tail-modulus alignment carrier)**.** *A* $\mathsf{TailModulusAlignment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, S, F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *row exposing the source moduli, shared meet, selector-budget route, and selected-window tolerance surface;*
2. *S is the* $\mathsf{RegularCauchyTailSchedule}^{\uparrow}$ *row exposing the precision request, cofinal tail bound, and scheduled finite tail witness;*
3. *F is the* $\mathsf{CauchyTailModulusFusion}^{\uparrow}$ *or* $\mathsf{RealModulusFusion}^{\uparrow}$ *row that joins the refinement and schedule surfaces before a seal-facing consumer reads them;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family common to the refinement, schedule, and fusion rows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled windows;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger attached to the common window readback;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, F, W, R, D have been displayed on the same packet;*
8. *H records componentwise* hsame *transport for M, S, F, W, R, D, E and the local certificate row;*
9. *C records displayed* Cont *replay from the refinement request through the schedule, fusion, window readback, dyadic ledger, and Real seal;*

10. *P is the* Pkg *provenance over* TailModulusAlignment$^{\uparrow}$, CauchyModulusRefinement$^{\uparrow}$, RegularCauchyTailSchedule$^{\uparrow}$, CauchyTailModulusFusion$^{\uparrow}$, RealModulusFusion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{TailModulusAlignment}^{\uparrow}}$ *row.*

The carrier exports only the displayed alignment rows and their transport, replay, provenance, and naming support.

Lean4 (definition) `BEDC.Derived.TailModulusAlignmentUp.tail modulus alignment taste gate`

Theorem 1210.2 (Tail-modulus alignment obligation surface)**.** *For the carrier of Definition 1210.1, the local* NameCert$_{\text{TailModulusAlignment}^{\uparrow}}$ *surface consists exactly of carrier admission, refinement-source exposure through M, tail-schedule exposure through S, fusion-route exposure through F, common* StreamName$^{\uparrow}$ *window admission through W,* RegSeqRat$^{\uparrow}$ *readback through R,* DyadicRatCore$^{\uparrow}$ *tolerance and arithmetic exposure through D,* Real$^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier A. Its coordinates list the refinement, schedule, fusion, stream-window, regular-rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows.

The data-bearing obligations are inherited in order. The refinement row exposes the source moduli and selected-window tolerance route, the schedule row exposes the cofinal tail request and finite tail witness, and the fusion row forces those two surfaces onto the same tail-modulus route before any seal-facing read is available.

The remaining rows are the common BEDC support rows. The StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ ledger, and Real$^{\uparrow}$ seal are consumed along the displayed route, while H, C, P, N only transport, replay, source, and name that route. Thus the listed rows exhaust the local naming obligations. □

Lean4 ✓ `BEDC.Derived.TailModulusAlignmentUp.TasteGate.TailModulusAlignmentTasteGate single carrier alignment`

Closure status for NameCert$_{\text{TailModulusAlignment}^{\uparrow}}$.

Constructive story: The tail-modulus alignment object is generated from $BHist$ rows that separately record a tail refinement route, a modulus schedule, common stream windows, regular-rational readback, and dyadic ledger data, then bind them by an explicit fusion witness. The carrier first reads the tail evidence, then the modulus evidence, and finally the ledger row showing that both are synchronized for the same Real-facing seal. The $hsame$ relation keeps the aligned rows stable under representative change, while Cont permits later handoffs only when the tail and modulus witnesses travel together. The sibling-dependency and non-escape theorems make this a composite certificate over visible upstream packets rather than a free-standing modulus principle.

Theory closure: TheoryGate(NameCert$_{\text{TailModulusAlignment}^{\uparrow}}$, seed)

Scope closed: This chapter binds the tail-modulus alignment carrier, obligation-surface theorem, sibling-dependency theorem, and non-escape theorem.

Formal verification: FormalStatus(NameCert$_{\text{TailModulusAlignment}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim an independent modulus extractor, arbitrary tail equality, or alignment outside the displayed sibling routes.

Upgrade path: Reaching obligationClosure requires formal tail and modulus row predicates plus checked synchronization through the alignment witness.

Theorem 1210.3 (Tail-modulus alignment sibling dependency)**.** *Every accepted* $\mathsf{TailModulusAlignment}^{\uparrow}$ *packet depends essentially on both the refinement sibling and the schedule sibling: deleting M removes the displayed source-modulus and selector-budget route, while deleting S removes the displayed precision-to-cofinal-tail schedule. In either deletion, the remaining rows cannot determine the common* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff through* hsame*,* Cont*,* Pkg*, and* NameCert *alone.*

Proof. First remove the refinement row. The schedule row still displays a precision request and a cofinal tail witness, but it no longer carries the source-modulus meet, selector-budget route, or selected-window tolerance readback needed to identify the common window W as the refinement window.

Next remove the schedule row. The refinement row still displays source moduli and a selected-window route, but it no longer carries the cofinal-tail request and finite tail witness that make the route usable as a Real-completion tail schedule.

The structural rows cannot repair either absence. The hsame and Cont rows transport and replay displayed data; Pkg and NameCert record provenance and naming for displayed data. None of them creates a missing refinement or schedule coordinate, so both siblings are genuine dependencies of the alignment packet. □

Theorem 1210.4 (Tail-modulus alignment non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{TailModulusAlignment}^{\uparrow}$ *packet factors through the finite route*

$$M, S, F, W, R, D, E, H, C, P, N.$$

The consumer receives a shared tail-modulus alignment over displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It receives no quotient stream, host sequence equality, selected host limit, hidden tail oracle, countable-choice selector, independent Real seal, or ambient Real-completeness witness.*

Proof. Begin with the accepted carrier. The only Real-facing coordinate is the seal row E, and it is downstream of the refinement row M, the schedule row S, the fusion row F, the common stream windows W, the regular-rational readback R, and the dyadic ledger D.

Apply Definition 1210.2. It forces every accepted consumer to read the refinement, schedule, fusion, window, readback, ledger, and seal rows through the displayed continuation route before the local NameCert row can be used.

Finally apply Definition 1210.3. Since neither sibling route can be omitted, the packet has no alternate route to a Real-facing output. The excluded objects are not coordinates of the carrier and are not introduced by hsame, Cont, Pkg, or NameCert, so no consumer can project them from the alignment certificate. □

Chapter 1211

A Concrete Naming Certificate for DyadicToleranceTriangleLedger$^{\uparrow}$

The DyadicToleranceTriangleLedger$^{\uparrow}$ packet realizes the three-leg diagonal-regularity estimate of Definition 9238.4 inside the Real-completeness route of chapter 9238. It is a dyadic ledger layer over the existing RegularCauchyTriangleBound$^{\uparrow}$ estimate of chapter 466: DyadicRatCore$^{\uparrow}$ supplies tolerance arithmetic through chapter 291, StreamName$^{\uparrow}$ supplies the finite diagonal and intermediate windows through chapter 126, RegSeqRat$^{\uparrow}$ supplies endpoint regularity reads through chapter 301, and Real$^{\uparrow}$ is the downstream seal surface of chapter 38. The packet does not select a limit, quotient a stream, or appeal to ambient completeness; it records the finite tolerance rows that the diagonal proof must read before the Real-facing handoff.

Definition 1211.1 (Dyadic tolerance triangle ledger carrier)**.** *A* DyadicToleranceTriangleLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N)$$

with the following displayed rows:

1. D_m *and* D_n *are the two* StreamName$^{\uparrow}$ *diagonal observation rows for the requested indices* m *and* n*.*
2. I_m *and* I_n *are the two intermediate* StreamName$^{\uparrow}$ *rows used by the diagonal-regularity triangle.*
3. E_m *and* E_n *are the endpoint* RegSeqRat$^{\uparrow}$ *regularity reads for the legs* $D_m \longrightarrow I_m$ *and* $I_n \longrightarrow D_n$*.*
4. M *is the middle Cauchy-modulus read for the leg* $I_m \longrightarrow I_n$*.*
5. Q *is the* DyadicRatCore$^{\uparrow}$ *ledger carrying the two endpoint tolerances, the middle tolerance, and the displayed dyadic output bound.*
6. T *is the consumed* RegularCauchyTriangleBound$^{\uparrow}$ *row that assembles the same three legs as one diagonal estimate.*
7. H *records componentwise* hsame *transport for the four window rows, the three leg rows, the dyadic ledger, the consumed triangle-bound row, provenance, and the local name.*
8. C *records displayed* Cont *routes for reading endpoint legs, reading the middle leg, summing dyadic tolerances, checking the consumed triangle-bound row, and handing the packet to the Real-facing diagonal consumer.*

9. *P is the* Pkg *provenance over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RegularCauchyTriangleBound$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{DyadicToleranceTriangleLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a missing endpoint leg, a missing middle modulus leg, quotient stream equality, host real equality, selected limits, or ambient completeness.

Definition 1211.2 (Dyadic tolerance triangle ledger classifier)**.** *Two accepted* DyadicToleranceTriangleLedger$^{\uparrow}$ *carriers classify as the same ledger only when their four* StreamName$^{\uparrow}$ *windows, two* RegSeqRat$^{\uparrow}$ *endpoint reads, middle modulus read,* DyadicRatCore$^{\uparrow}$ *tolerance row, consumed* RegularCauchyTriangleBound$^{\uparrow}$ *row, and* BHist/hsame/Cont/Pkg/NameCert *support rows are componentwise transported by the displayed* hsame *data. The classifier does not identify packets by host equality of real numbers, by quotienting streams, or by forgetting one of the three triangle legs.*

Theorem 1211.3 (Dyadic tolerance triangle ledger obligations)**.** *For every accepted* DyadicToleranceTriangleLedger *carrier $L = (D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N)$, the local* NameCert$_{\text{DyadicToleranceTriangleLedger}^{\uparrow}}$ *surface consists exactly of carrier exposure, classifier transport, endpoint-leg coverage for E_m, E_n, middle-leg coverage for M, dyadic tolerance addition through Q, consumed* RegularCauchyTriangleBound$^{\uparrow}$ *compatibility through T, Real-facing continuation through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DyadicToleranceTriangleLedgerUp.DyadicToleranceTriangleLedger window endpoint separation`

Proof. Carrier exposure is projection from Definition 1211.1. The displayed rows name the two diagonal windows, the two intermediate windows, the two endpoint legs, the middle leg, the dyadic ledger, the consumed triangle-bound packet, and the support rows.

Classifier transport is exactly the relation of Definition 1211.2. It keeps the four windows, three leg rows, dyadic ledger, consumed triangle-bound row, and support rows on one componentwise transported surface.

The endpoint and middle obligations are then read from the sibling surfaces. The endpoint reads use RegSeqRat$^{\uparrow}$ through chapter 301; the middle read is the Cauchy-modulus leg already assembled by chapter 466; and the tolerance addition is the DyadicRatCore$^{\uparrow}$ row of chapter 291.

Finally C, P, N hand this finite ledger to the Real$^{\uparrow}$ diagonal consumer. Since the carrier omits quotient streams, selected limits, host real equality, missing-leg shortcuts, and ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 1211.4 (Dyadic tolerance triangle ledger nonescape)**.** *Every diagonal-regularity consumer of an accepted* DyadicToleranceTriangleLedger$^{\uparrow}$ *packet reads the two endpoint legs E_m, E_n, the middle Cauchy-modulus leg M, the dyadic tolerance ledger Q, the consumed triangle-bound row T, and the displayed* BHist/hsame/Cont/Pkg/NameCert *support rows. It cannot replace any leg by a host real equality, quotient-stream comparison, selected limit, detached completion witness, or ambient completeness row.*

Proof. Begin with the obligation surface of Definition 1211.3. It exposes the two endpoint legs, the middle leg, the tolerance ledger, and the consumed triangle-bound row as required rows of the local certificate.

Use the classifier of Definition 1211.2. A consumer that transports the ledger must transport all three leg rows and the dyadic output row together; forgetting one leg changes the accepted packet.

The excluded objects have no coordinate in Definition 1211.1. Therefore no continuation route can export host equality, quotient-stream comparison, selected limits, detached completion witnesses, or ambient completeness in place of the displayed three-leg ledger. □

Theorem 1211.5 (Dyadic tolerance triangle ledger window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or diagonal-limit consumer that uses a* $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *packet receives the finite handoff*

$$D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N$$

after the four $\mathsf{StreamName}^{\uparrow}$ *windows have been read and before any Real seal is consumed. The handoff realizes the three-leg estimate of Definition 9238.4 as finite* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

Proof. First read the carrier. The four window rows D_m, D_n, I_m, I_n are the only stream observations available to the packet, and the leg rows E_m, E_n, M are attached to those windows.

Next apply the obligation theorem. It requires the dyadic output bound to be read through Q and the compatibility with the existing three-leg estimate to be read through T.

Finally use non-escape. Since no selected limit, quotient stream, host real equality, detached completion witness, or ambient completeness row can be exported, the Real-facing consumer receives exactly the displayed finite handoff. □

Theorem 1211.6 (Dyadic tolerance triangle ledger middle-leg coverage)**.** *For every accepted* $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *carrier* $L = (D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N)$*, the row* M *is the only admissible middle Cauchy-modulus leg. Any accepted consumer of* L *that reaches from* I_m *to* I_n *must read* M *through the displayed* Cont *route and cannot replace it by an endpoint regularity row, a quotient stream equality, a selected limit, a detached completion witness, or ambient completeness.*

Proof. Read the carrier of Definition 1211.1. Among the displayed rows, only M is assigned to the leg $I_m \longrightarrow I_n$; the rows E_m and E_n are bound to the two endpoint legs, and no extra coordinate records a second middle modulus.

Use the classifier of Definition 1211.2. Componentwise hsame transport keeps the middle leg as the M coordinate, so a transported packet cannot silently exchange M for an endpoint row or for an external real-completion witness.

Finally apply non-escape from Definition 1211.4. The excluded host, quotient, limit-selection, detached-completion, and ambient-completeness objects have no coordinate in the packet, hence no accepted middle-leg consumer can read them in place of M. □

Theorem 1211.7 (Dyadic tolerance triangle ledger tolerance addition)**.** *For every accepted* DyadicToleranceTriangle*carrier* $L = (D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N)$*, the row* Q *is the only dyadic tolerance-addition ledger combining the two endpoint tolerances and the middle tolerance. The packet exposes no second dyadic output row and no route that can compute the displayed output bound outside* $\mathsf{DyadicRatCore}^{\uparrow}$.

Proof. The carrier assigns exactly one row to dyadic tolerance addition, namely Q. The endpoint tolerances are supplied by E_m and E_n, the middle tolerance is supplied by M, and the displayed output bound is part of the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row.

Classifier transport preserves Q as the dyadic ledger coordinate. Thus a consumer that transports the accepted packet must transport the endpoint tolerances, the middle tolerance, and the output bound together rather than recomputing one of them through a detached route.

The non-escape theorem blocks the remaining shortcuts. Since the packet has no coordinate for host real equality, quotient streams, selected limits, detached completion witnesses, or ambient completeness, the only accepted tolerance-addition read is the Q row. □

Theorem 1211.8 (Dyadic tolerance triangle ledger triangle-bound compatibility)**.** *For every accepted* $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *carrier* $L = (D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N)$*, the consumed row* T *uses the same three legs* E_m, E_n, M *as the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *packet. It may package the three-leg estimate for the Real-facing handoff, but it cannot introduce a fourth leg, omit the middle leg, or identify endpoint and middle evidence by host equality.*

Proof. The carrier binds T as the consumed $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ row and lists exactly three leg rows beside it: E_m, E_n, and M. These are the two endpoint regularity reads and the middle Cauchy-modulus read used by the diagonal-regularity triangle.

The obligation theorem forces compatibility through the same displayed rows. It does not expose an independent triangle-bound coordinate that could ignore M or introduce a fourth leg.

The window handoff theorem then sends $D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N$ to the Real-facing consumer as one finite packet. Because non-escape forbids host equality and quotient-stream replacement, T remains compatible with the same three concrete legs. □

Theorem 1211.9 (Dyadic tolerance triangle ledger tastegate obligation closure)**.** *The* DyadicToleranceTriangleLedg *packet is obligation-closed for its displayed L10 ledger surface. The carrier, classifier, nonescape theorem, window handoff, falsifiable prediction, independence witness, Lean* `ChapterTasteGate`*,* Nontrivial $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$*, and* FieldFaithful $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *evidence jointly bind the rows* $D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N$ *as the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.DyadicToleranceTriangleLedgerUp.taste gate`

Proof. Carrier and classifier exposure are supplied by Definition 1211.1 and Definition 1211.2. The aggregate obligation theorem names the endpoint, middle, dyadic-addition, triangle-bound, Real-continuation, package, and local-name clauses.

The three separated compatibility theorems above isolate the middle leg M, the dyadic addition row Q, and the consumed triangle-bound row T. Together with endpoint separation, they make the finite ledger surface checkable row by row.

Nonescape and window handoff then bind the consumer behavior: no quotient stream, selected limit, host equality, detached completion witness, or ambient-completeness row can replace the displayed finite packet.

Finally the falsifiable prediction and independence witness record the negative tests, while the checked Lean `ChapterTasteGate` supplies the formal gate together with its Nontrivial and FieldFaithful evidence. These ingredients exhaust the stated obligation surface without claiming a public Real-completion seal. □

Falsifiable prediction: A diagonal-regularity consumer that omits one of the two endpoint legs or the middle Cauchy-modulus leg refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-triangle-bound: none is bijective to $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$, because the dyadic core lacks stream windows, the stream-name surface lacks rational tolerance addition, the regular-rational surface

lacks the middle Cauchy-modulus leg, the Real surface is downstream of the seal, and the regular-cauchy triangle-bound packet does not isolate the dyadic tolerance ledger as its own classifier and Real-window handoff.

Lean `TasteGate.lean` for DyadicToleranceTriangleLedger$^{\uparrow}$ must provide explicit Nontrivial DyadicToleranceTriang and FieldFaithful DyadicToleranceTriangleLedger$^{\uparrow}$ instances for $D_m, D_n, I_m, I_n, E_m, E_n, M, Q, T, H, C, P, N$.

Closure status for NameCert$_{\text{DyadicToleranceTriangleLedger}^{\uparrow}}$.

Constructive story: A DyadicToleranceTriangleLedger$^{\uparrow}$ packet is generated from four stream windows, two endpoint regularity reads, one middle Cauchy-modulus read, a dyadic tolerance-addition ledger, a consumed RegularCauchyTriangleBound row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicToleranceTriangleLedger}^{\uparrow}}$, obligation)

Scope closed: Definition 1211.1, Definition 1211.2, Definition 1211.3, Definition 1211.4, Definition 1211.5, Definition 1211.6, Definition 1211.7, Definition 1211.8, and Definition 1211.9 bind the finite dyadic ledger surface for the diagonal-regularity triangle.

Formal verification: FormalStatus(NameCert$_{\text{DyadicToleranceTriangleLedger}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.DyadicToleranceTriangleLedgerUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridge checking, scoped downstream Real-completion closure, the full Real-completeness theorem, quotient stream equality, selected limits, host real equality, detached completion witnesses, or ambient completeness.

Upgrade path: Scoped closure requires downstream diagonal-regularity consumers to reuse the same endpoint rows, middle row, dyadic tolerance row, triangle-bound row, and Real-window handoff without adding nonlocal witnesses.

Chapter 1212

A Concrete Naming Certificate for $\mathsf{DyadicModulusLadder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicModulusLadder}^{\uparrow}$ packet records a finite ladder for dyadic tolerance requests, unary precision indices, finite stream windows, regular rational readback, DyadicRatCore arithmetic, and the terminal Real seal. It sits beside the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 open phase a finite schedule object for later regular-Cauchy and RealUp routes, not a quotient stream, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1212.1 (Dyadic modulus ladder carrier)**.** *A* $\mathsf{DyadicModulusLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, U, S, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request row naming the dyadic precision request that begins the ladder.*
2. *U is the unary precision index row, recording the finite stage at which the request is read.*
3. *S is the monotone ladder step row, advancing the precision request through a displayed finite order.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the observed stages of the ladder.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying the regular rational response to the displayed window.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row, recording the finite dyadic comparison and reuse of the request.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after T, U, S, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for $T, U, S, W, R, D, E, C, P, N$.*
9. *C records displayed* Cont *replay from the dyadic request through the unary index, ladder step, stream window, regular rational readback, dyadic arithmetic, Real seal, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicModulusLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, U, S, W, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1212.2 (Dyadic modulus ladder NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulusLadder}^{\uparrow}$ *carrier $L = (T, U, S, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulusLadder}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, unary precision-index exposure, monotone ladder-step exposure, finite-window exposure, regular rational readback, DyadicRatCore arithmetic, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 1212.1. The rows T, U, S display the request, its unary index, and the monotone ladder step that advances the request through the finite schedule.

The window row W and readback row R then expose the finite observation and regular rational response. The arithmetic row D keeps the dyadic comparison internal to the same packet rather than turning it into a host-level modulus function.

Finally read E, H, C, P, N. The Real seal is downstream of the displayed dyadic request, unary index, ladder step, window, readback, and arithmetic rows, while the structural rows transport, replay, source, and name the same finite route. □

Theorem 1212.3 (Dyadic modulus ladder regular-sequence route)**.** *Every accepted* $\mathsf{DyadicModulusLadder}^{\uparrow}$ *packet used by a regular-sequence consumer factors through the displayed route*

$$T, U, S, W, R, D, E, H, C, P, N.$$

The consumer reads the dyadic request, unary precision index, monotone ladder step, finite stream window, regular rational readback, DyadicRatCore arithmetic, and terminal $\mathsf{Real}^{\uparrow}$ *seal before it can use the ladder as a regular-sequence route.*

Proof. Begin with the NameCert obligations. They force the dyadic request, unary index, ladder step, finite window, readback, arithmetic, and seal to be displayed before the packet can be consumed.

The finite ladder step is what keeps the route monotone. It carries the request forward through a visible unary stage rather than selecting a limit representative or a quotient stream.

The structural rows H, C, P, N preserve the same route under transport and replay. Since no carrier coordinate stores quotient stream equality, selected limits, countable choice, or ambient completeness, the regular-sequence consumer remains on the displayed ladder. □

Theorem 1212.4 (Dyadic modulus ladder Real seal boundary)**.** *Every Real-facing consumer of an accepted* $\mathsf{DyadicModulusLadder}^{\uparrow}$ *packet must read the dyadic request row T, unary precision index U, monotone ladder step S, finite stream-window row W, regular rational readback row R, and DyadicRatCore arithmetic row D before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is consumed. The boundary exports no quotient-stream equality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The Real-facing route is already forced through the displayed request, unary index, ladder step, window, readback, and arithmetic rows.

Apply the regular-sequence route theorem. It keeps the consumer on the same monotone finite ladder and prevents a detour through a selected limit or quotient stream.

The remaining structural rows H, C, P, N preserve and name this route. No extra completion coordinate is available in the carrier, so the displayed boundary is the whole Real seal surface. □

Falsifiable prediction: Every accepted $\mathsf{DyadicModulusLadder}^{\uparrow}$ Real or regular-sequence consumer reads one dyadic request row, one unary precision-index row, one monotone ladder-step row, one $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before using the displayed ladder.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicModulusLadder}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide tolerance comparison, finite windows, regular rational readback, and terminal Real sealing, while this packet binds them into one monotone finite ladder route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulusLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicModulusLadder}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance request, a unary precision index, a monotone ladder step, a StreamName finite-window row, a RegSeqRat readback row, a DyadicRatCore arithmetic row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulusLadder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1212.1, Definition 1212.2, Definition 1212.3, and Definition 1212.4 seed the finite ladder route from displayed dyadic requests to Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulusLadder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-stream equality, selected limits, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, unary index exposure, monotone ladder-step exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, transport, replay, and ledger non-escape for BEDC.Derived.DyadicModulusLadderUp.

Chapter 1213

A Concrete Naming Certificate for $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$

The $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ packet realizes the representation-gap row of section 9414.1, especially section 9414.1, as a finite NameCert object. It sits downstream of the signature-fit carrier of **??**, the open-fit carrier of chapter 1132, the relational-objectivity carrier of chapter 1138, and the methodology-ledger carrier of chapter 1185. Those siblings record successful compatibility, admitted open fit routes, objectivity rows, and method provenance; this packet records the remaining reality-facing signature separation after a mathematical classifier has already identified the compared representations.

Definition 1213.1 (Reality-constrained signature residue carrier)**.** *A* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M, S, G, R, W, H, C, P, N)$$

with the following visible rows:

1. *M is the mathematical classifier row, recording that the compared representations are identified by the selected mathematical classifier;*
2. *S is the physical signature row, recording the reality-facing signature bundle and the exposed physical-fit comparison;*
3. *G is the GAP and methodology-ledger row, naming the admitted representation gap, the observation scope, and the provenance route through the methodology ledger;*
4. *R is the residue row, storing the remaining physical-signature separation that survives the mathematical identification in M;*
5. *W is the witness route, displaying the finite observation or audit route that exposes the residue rather than inferring it from host disagreement;*
6. *H stores componentwise* hsame *transport for the classifier row, physical signature row, GAP and ledger row, residue row, witness route, continuation rows, provenance row, and local naming row;*
7. *C stores displayed* Cont *replay for mathematical-classifier inspection, physical-signature inspection, GAP-ledger inspection, residue exposure, witness replay, and NameCert comparison;*

8. *P is the* Pkg *provenance row over* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *row.*

The carrier contains no coordinate that turns mathematical classifier agreement alone into physical adequacy. A physical-adequacy read must display S, G, R, W in the same finite packet.

Theorem 1213.2 (Reality-constrained signature residue NameCert obligations)**.** *For the carrier of Definition 1213.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $Z = (M, S, G, R, W, H, C, P, N)$;*

2. *classifier separation, requiring M to be read as mathematical agreement and S as the physical-signature comparison rather than a consequence of M;*

3. *residue exposure, requiring G, R, W to display the representation gap, remaining signature separation, and finite witness route before any consumer can mention physical inadequacy;*

4. *transport and replay stability, requiring H and C to preserve the classifier, signature, ledger, residue, witness, provenance, and local naming rows componentwise;*

5. *provenance and local naming, requiring P and N to cite the signature-fit, open-fit, relational-objectivity, methodology-ledger,* BHist, hsame, Cont, Pkg, *and* NameCert *surfaces used by the packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureResidueUp.TasteGate.RealityConstrainedSignatureResidueT`
`single carrier alignment`

Proof. Project the finite packet of Definition 1213.1. Its coordinates already separate mathematical agreement, physical signature comparison, GAP-ledger data, residue exposure, witness route, transport, replay, provenance, and local naming.

The classifier-separation obligation reads M and S as different rows of the same packet. The residue-exposure obligation then requires G, R, W before any consumer can use the gap as evidence against physical adequacy.

Finally, the rows H, C, P, N keep the packet stable under componentwise transport and displayed replay. Since no further coordinate exists, the five listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureResidueUp.RealityConstrainedSignatureResidueTasteGate`
`single carrier alignment`

Theorem 1213.3 (Reality-constrained signature residue consumer boundary)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. Every public consumer of the residue packet factors through exactly the displayed carrier route*

$$M, S, G, R, W, H, C, P, N.$$

It may read mathematical classifier agreement, physical-signature comparison, GAP and methodology-ledger data, residue, witness route, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *row. It has no direct consumer coordinate for host physical adequacy, objectivity completion, or physical-fit resolution outside that route.*

Proof. Project the carrier of Definition 1213.1. Its only visible rows are $M, S, G, R, W, H, C, P, N$, so a consumer of the local packet has no additional carrier coordinate to inspect.

Apply the NameCert obligations of Definition 1213.2. They require classifier separation, residue exposure through G, R, W, componentwise transport and replay through H, C, and provenance plus local naming through P, N.

The sibling anchors in **??**, chapter 1132, chapter 1138, and chapter 1185 are cited only through the provenance route. Hence a public consumer factors through the displayed route and not through a hidden adequacy or objectivity-completion output. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureResidueUp.RealityConstrainedSignatureResidueConsumerBou`

Theorem 1213.4 (Reality-constrained signature residue dependency handoff)**.** *The* RealityConstrainedSignatureRes *packet consumes exactly the reality-constrained sibling surfaces*

$$\mathsf{RealityConstrainedSignatureFit}^{\uparrow},\quad \mathsf{RealityConstrainedOpenFit}^{\uparrow},\quad \mathsf{RelationalObjectivity}^{\uparrow},\quad \mathsf{RealityConstrainedMethod}$$

for signature-fit compatibility, open-fit admission, objectivity rows, and methodology provenance. Beyond those consumed sibling surfaces and the ambient BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structure, the packet contributes only the residual signature-separation row R together with its displayed witness route W and local naming row N.*

Proof. Read the dependency row in the carrier definition. The row P names $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ together with $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, and the structural BHist, hsame, Cont, Pkg, and NameCert surfaces.

The four sibling chapters cited in the introduction provide the consumed reality-constrained surfaces: signature fit, open fit, relational objectivity, and methodology ledger. The carrier does not name a fifth reality-constrained sibling dependency.

Apply the NameCert obligations. Their classifier-separation and residue-exposure rows force the new local contribution to be the displayed separation R with its witness route W and name N, rather than a duplicate of the consumed sibling packets or an adequacy conclusion. □

Theorem 1213.5 (Reality-constrained signature residue obligation closure)**.** *The obligation surface of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet is exactly the finite package consisting of carrier admission, classifier separation, residue exposure, witness-route display, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *naming for the route $M, S, G, R, W, H, C, P, N$. This closure is the local TasteGate-facing obligation surface of the packet and does not assert bridge checking or public physical adequacy.*

Proof. Start with Definition 1213.1. It supplies the carrier rows $M, S, G, R, W, H, C, P, N$ and separates mathematical agreement from physical-signature comparison before the residue row can be read.

Apply Definition 1213.2. The five NameCert rows expand to carrier admission, classifier separation, residue exposure, stability under H, C, and provenance plus naming under P, N.

Use the consumer boundary and dependency handoff theorems above. The first keeps every consumer on the displayed route, while the second fixes the four consumed sibling surfaces and isolates the residual signature-separation contribution.

The TasteGate alignment already displayed at this NameCert site is therefore read only as a local obligation alignment. Since no standard bridge row or public-adequacy row is present in the carrier, the closure reached here is the obligation surface and not a bridge-checked or physical-adequacy claim. □

Theorem 1213.6 (Reality-constrained signature residue non-escape)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. No consumer may export physical adequacy, physical inadequacy, or physical-fit resolution from the mathematical classifier row M alone. Every public physical-fit consumer must first expose the physical signature row S, GAP and methodology-ledger row G, residue row R, witness route W, transport row H, replay row C, provenance row P, and local naming row N.*

Proof. Start from the NameCert obligations of Definition 1213.2. They make classifier separation and residue exposure mandatory before any physical-fit claim can leave the packet.

If a route reads only M, it has mathematical classifier agreement but no displayed physical-signature comparison, GAP ledger, residue row, or witness route. Such a route therefore lacks the rows required for a physical adequacy or inadequacy claim.

Reading S, G, R, W, H, C, P, N restores the displayed packet route. The conclusion is a non-escape boundary: the consumer may use the accepted residue packet only through its visible rows, never through a direct export from mathematical classifier agreement. □

Theorem 1213.7 (Reality-constrained signature residue sibling independence)**.** *The* RealityConstrainedSignatureRe *carrier is not bijective to* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, *or* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$. *SignatureFit records successful compatibility, OpenFit records admitted open fit routes, RelationalObjectivity records objectivity rows, and MethodologyLedger records method provenance, while the residue packet records the remaining reality-facing signature separation after mathematical identification.*

Proof. Compare the displayed coordinates. The signature-fit packet supplies compatibility rows but need not carry a remaining residue row R or a witness route W for a failed physical-signature descent. The open-fit packet supplies continuation and finite refutation surfaces, but it does not by itself bind the mathematical classifier row M to a surviving representation-gap residue.

The relational-objectivity packet supplies objectivity rows, and the methodology ledger supplies method provenance. Neither one alone contains the joint classifier, physical-signature, GAP, residue, witness, transport, replay, provenance, and local naming coordinates of Z.

Thus each sibling may be consumed by the residue packet, but none is interchangeable with it. The carrier independence is exactly the remaining row R displayed between mathematical identification and physical adequacy. □

Falsifiable prediction: If a RealityConstrainedSignatureResidueUp packet is accepted, every public physical-fit consumer must expose the residue row before claiming adequacy; a route from mathematical classifier agreement directly to physical adequacy refutes the packet.

Independence witness: reality-constrained-signature-fit, reality-constrained-open-fit, relational-objectivity, reality-constrained-methodology-ledger: SignatureFit records successful compatibility, OpenFit records admitted open fit routes, RelationalObjectivity records objectivity rows, and MethodologyLedger records method provenance, while the residue packet records the remaining reality-facing signature separation after mathematical identification.

Theorem 1213.8 (Reality-constrained signature residue scope lock)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. The residue scope is locked to the displayed dependencies on* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, *and* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$*: M reads the signature-fit classifier, S and G read the open-fit and relational-objectivity comparison surface, R stores only the remaining signature*

residue, and W, H, C, P, N expose the methodology-ledger witness, transport, replay, provenance, and local naming rows. No consumer may enlarge the scope to a host physical-adequacy judgment or shrink it to mathematical classifier agreement alone.

Proof. Project the carrier of Definition 1213.1. The rows $M, S, G, R, W, H, C, P, N$ already list the classifier, physical signature, gap ledger, residue, witness, transport, replay, provenance, and local naming surfaces.

Apply the NameCert obligations of Definition 1213.2. They force classifier separation and residue exposure before any physical-fit consumer can read the packet.

Finally invoke the non-escape theorem, Definition 1213.6. A route from M alone lacks S, G, R, W, while a route to host physical adequacy lacks a coordinate in the carrier; hence the displayed sibling dependencies are exactly the locked scope. □

Theorem 1213.9 (Reality-constrained signature residue witness-route totality)**.** *For every accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet* $Z = (M, S, G, R, W, H, C, P, N)$*, every public witness of the residue row R factors through the finite route*

$$M, S, G, R, W, H, C, P, N$$

with P citing $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$*,* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$*,* $\mathsf{RelationalObjectivity}^{\uparrow}$*, and* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$*. Conversely, that displayed route supplies all witness data available to a consumer of the residue packet.*

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureResidueUp.RealityConstrainedSignatureResidueWitnessRout`

Proof. First read the witness row in the carrier. The row W is placed after the mathematical classifier, physical signature, gap-ledger, and residue rows, so a witness to R is not detached from M, S, G.

The NameCert obligations identify G, R, W as the residue-exposure surface. The transport and replay rows H, C then preserve that same exposure under componentwise movement and displayed continuation replay.

The provenance row P supplies the listed sibling dependencies, and N locally names the same packet. Since the non-escape theorem rules out physical-fit resolution from M alone or from any hidden host coordinate, the displayed route is total for residue witnesses. □

Theorem 1213.10 (Reality-constrained signature residue consumer adequacy refusal)**.** *An accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *consumer that exposes $M, S, G, R, W, H, C, P, N$ may report the remaining signature residue, but it may not report physical adequacy, physical-fit resolution, or objectivity completion unless a separate packet supplies those claims through* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$*,* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$*,* $\mathsf{RelationalObjectivity}^{\uparrow}$*, or* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$*. The residue packet itself is a refusal boundary for adequacy by mathematical agreement alone.*

Proof. Begin with the sibling independence theorem, Definition 1213.7. It separates the residue carrier from the signature-fit, open-fit, relational-objectivity, and methodology-ledger carriers.

Next use non-escape, Definition 1213.6. The mathematical classifier row M cannot export physical adequacy, physical inadequacy, or physical-fit resolution without the rows S, G, R, W, H, C, P, N.

The accepted residue consumer therefore receives a residue report, not an adequacy closure. Any positive adequacy or objectivity-completion claim must come from a separate accepted sibling packet, while this packet refuses the shortcut from mathematical classifier agreement to physical adequacy. □

Theorem 1213.11 (Reality-constrained signature residue sibling-route exactness)**.** *For an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet, the sibling route is exactly the displayed dependency on* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$*,* $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$*,* $\mathsf{RelationalObjectivity}^{\uparrow}$*, and* $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ *through the rows*

$$M, S, G, R, W, H, C, P, N.$$

No sibling alone supplies the residue packet, and no residue consumer may replace the displayed route by a host physical-adequacy judgment.

Proof. Begin with Definition 1213.7. It separates the residue carrier from the signature-fit, open-fit, relational-objectivity, and methodology-ledger carriers.

Apply Definition 1213.9. Every public witness of the residue row factors through $M, S, G, R, W, H, C, P, N$ with P citing the same sibling packets.

Use Definition 1213.10. A host physical-adequacy judgment is not exported by the residue packet, so the sibling route is exactly the displayed finite dependency. □

Theorem 1213.12 (Reality-constrained signature residue mathematical-agreement refusal)**.** *Mathematical classifier agreement in the row* M *of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet does not imply physical adequacy, physical-fit resolution, or objectivity completion. Any such claim must pass through the physical-signature, GAP, residue, witness, transport, replay, provenance, and local naming rows* S, G, R, W, H, C, P, N *and through a separate accepted sibling packet when the claim is positive.*

Proof. Start with the NameCert obligations of Definition 1213.2. They read M as mathematical agreement and S as the physical-signature comparison, so the two rows cannot be collapsed.

Apply non-escape, Definition 1213.6. A route from M alone cannot export physical adequacy, physical inadequacy, or physical-fit resolution.

Finally use Definition 1213.10. Positive adequacy or objectivity-completion claims must come from a separate accepted sibling packet, while this residue packet refuses the shortcut from classifier agreement to physical adequacy. □

Theorem 1213.13 (Reality-constrained signature residue methodology-ledger totality)**.** *Every public methodology-ledger read used by an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet is contained in the finite route*

$$G, W, H, C, P, N.$$

The ledger is exposed as GAP and method provenance for the residue witness, not as a free-standing proof of physical adequacy or objectivity completion.

Proof. Project the carrier of Definition 1213.1. The methodology-ledger data enters through G, is witnessed through W, and is transported, replayed, sourced, and named by H, C, P, N.

Apply Definition 1213.9. It makes the witness route for R pass through the same GAP, witness, transport, replay, provenance, and local naming rows.

Finish with Definition 1213.10. The residue consumer receives a residue report rather than an adequacy closure, so the methodology ledger is total only for the displayed witness route. □

Theorem 1213.14 (Reality-constrained signature residue carrier row readback)**.** *Every public residue read of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet projects to the displayed finite route*

$$M, S, G, R, W, H, C, P, N.$$

The read may expose mathematical classifier agreement, physical signature comparison, GAP and methodology-ledger data, residue, witness route, componentwise transport, continuation replay, provenance, and local naming, but it cannot read physical adequacy or objectivity completion outside those rows.

Proof. Project the carrier of Definition 1213.1. Its coordinates are exactly $M, S, G, R, W, H, C, P, N$, so a public residue read has no other row available.

Apply the NameCert obligations and non-escape theorem, Definition 1213.2 and Definition 1213.6. They require classifier separation, residue exposure, transport, replay, provenance, and local naming before a physical-fit consumer may read the packet.

Use witness-route totality and sibling-route exactness, Definition 1213.9 and Definition 1213.11. The public residue read is therefore the displayed route and not a hidden adequacy or objectivity-completion judgment. □

Theorem 1213.15 (Reality-constrained signature residue public consumer factorization)**.** *Every public physical-fit consumer of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet factors first through* $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{RelationalObjectivity}^{\uparrow}$, *and* RealityConstrainedM *as recorded by the rows* M, S, G, W, H, C, P, N, *and only then reads the residue row* R. *In particular, signature-fit, open-fit, relational-objectivity, and methodology-ledger surfaces must all be visible before the consumer treats the residue as a physical-fit obstruction.*

Proof. Begin with sibling independence, Definition 1213.7. It separates the residue carrier from the signature-fit, open-fit, relational-objectivity, and methodology-ledger siblings while allowing the residue packet to consume them.

Apply witness-route totality and sibling-route exactness, Definition 1213.9 and Definition 1213.11. The public route to the residue passes through M, S, G, W, H, C, P, N with provenance citing the sibling packets.

Finally use methodology-ledger totality, Definition 1213.13. The methodology row is exposed for the residue witness rather than for a free-standing adequacy claim, so the public consumer must factor through those surfaces before reading R. □

Theorem 1213.16 (Reality-constrained signature residue row-order totality)**.** *Every public residue read of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet exposes the finite rows in the order*

$$M, S, G, R, W, H, C, P, N.$$

The read may stop at no earlier row as a complete residue report: M *supplies mathematical classifier agreement,* S *supplies the physical-signature comparison,* G *supplies the GAP and methodology-ledger context,* R *supplies the displayed residue,* W *supplies the witness route,* H *and* C *supply transport and replay,* P *supplies provenance, and* N *supplies the local NameCert row.*

Proof. Project the carrier of Definition 1213.1. Its displayed coordinates are exactly $M, S, G, R, W, H, C, P, N$, and the NameCert obligations of Definition 1213.2 read those coordinates as carrier admission, classifier separation, residue exposure, transport and replay stability, provenance, and local naming.

Apply witness-route totality, Definition 1213.9. A public witness of R factors through the same ordered route, so no consumer can present R before M, S, G or detach it from W, H, C, P, N.

Finally use public-consumer factorization, Definition 1213.15. The fit-facing consumer must see the sibling surfaces and methodology ledger before reading R, hence the displayed row order is total for every public residue read. □

Theorem 1213.17 (Reality-constrained signature residue audit consumer readiness)**.** *An audit consumer may treat an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet as actionable only after seeing all nine readiness rows: the classifier row M, the signature row S, the GAP and ledger row G, the residue row R, the witness route W, the transport row H, the replay row C, the provenance row P, and the local NameCert row N. Before those rows are visible, the consumer has at most a partial inspection route and not an actionable residue certificate.*

Proof. Start with the NameCert obligations, Definition 1213.2. They make carrier admission, classifier separation, residue exposure, transport, replay, provenance, and local naming mandatory parts of the packet rather than optional annotations.

Use non-escape, Definition 1213.6. A route from M alone cannot export physical adequacy, physical inadequacy, or physical-fit resolution; the audit consumer must expose S, G, R, W, H, C, P, N before any fit-facing use of the residue is available.

Combine row-order totality and public-consumer factorization, Definition 1213.16 and Definition 1213.15. The actionable audit route is therefore exactly the ordered nine-row surface, with the classifier, signature, GAP, residue, witness, transport, replay, provenance, and local naming rows all present. □

Theorem 1213.18 (Reality-constrained signature residue fit-obstruction handoff)**.** *Every downstream fit-obstruction consumer of an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet receives only the displayed residue obstruction carried by R along the route $M, S, G, R, W, H, C, P, N$. The handoff does not supply physical adequacy, physical-fit resolution, or objectivity completion; those claims require separate accepted sibling packets.*

Proof. Begin with non-escape, Definition 1213.6. It blocks any handoff from exporting adequacy, inadequacy, or fit resolution from mathematical classifier agreement alone.

Apply witness-route totality and public-consumer factorization, Definition 1213.9 and Definition 1213.15. The downstream consumer receives the residue only through the displayed route and through the sibling surfaces named by provenance.

Finally use audit consumer readiness, Definition 1213.17. The consumer may act on the residue obstruction once the nine rows are exposed, but the same readiness surface contains no coordinate for physical adequacy or objectivity completion. □

Theorem 1213.19 (Reality-constrained signature residue methodology provenance nonescape)**.** *In an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet, methodology provenance in G, W, H, C, P, N cannot be exported as physical adequacy, physical-fit resolution, or objectivity completion. It remains provenance for the displayed residue route $M, S, G, R, W, H, C, P, N$ and must be paired with a separate accepted sibling packet before any positive adequacy or objectivity claim is available.*

Proof. Start with methodology-ledger totality, Definition 1213.13. The ledger is total for the route G, W, H, C, P, N, not for adequacy or objectivity completion.

Apply consumer adequacy refusal and mathematical-agreement refusal, Definition 1213.10 and Definition 1213.12. They rule out physical adequacy, physical-fit resolution, and objectivity completion from either M alone or from the residue packet alone.

Finally use the carrier row readback of Definition 1213.14. Since the public read is the same displayed route, methodology provenance cannot leave the packet as an independent physical-adequacy or objectivity-completion certificate. □

Theorem 1213.20 (Reality-constrained signature residue carrier admission)**.** *Every accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet exposes the finite rows*

$$M, S, G, R, W, H, C, P, N$$

as BHist*-carrier data before a public residue consumer may read the local name. These rows respectively carry mathematical classifier agreement, physical signature comparison, GAP and methodology-ledger data, residue, witness route, componentwise transport, displayed replay, package provenance, and local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *naming.*

Proof. Project the carrier row readback of Definition 1213.14. It states that every public residue read of an accepted packet has exactly the displayed route through $M, S, G, R, W, H, C, P, N$.

Apply public-consumer factorization, Definition 1213.15. The consumer must pass through the signature-fit, open-fit, relational-objectivity, and methodology-ledger surfaces recorded by the same rows before reading R as a physical-fit obstruction.

Finally use methodology-provenance nonescape, Definition 1213.19. Since methodology provenance cannot leave the packet as adequacy or objectivity completion, the admitted carrier data are exactly the displayed finite BHist rows. □

Theorem 1213.21 (Reality-constrained signature residue classifier exactness)**.** *Classifier acceptance for* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *is exactly preservation of the rows* $M, S, G, R, W, H, C, P, N$ *under the local residue route. A transported packet is accepted as the same residue packet precisely when mathematical classifier agreement, signature comparison, methodology ledger, residue, witness, transport, replay, provenance, and local naming remain on that route.*

Proof. Begin with carrier admission, Definition 1213.20. It identifies the carrier surface that a residue classifier is allowed to compare.

Use public-consumer factorization to fix the route order. The consumer does not read R before the sibling surfaces and methodology row have been displayed, so classifier preservation cannot be tested on R alone or on mathematical agreement alone.

The reverse containment follows from carrier row readback. Once $M, S, G, R, W, H, C, P, N$ are preserved along the local route, there is no additional carrier coordinate for the classifier to inspect. Methodology-provenance nonescape prevents a preserved methodology row from being reclassified as physical adequacy or objectivity completion. □

Theorem 1213.22 (Reality-constrained signature residue NameCert ledger)**.** *The* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureR}}$ *ledger is exhausted by methodology, signature, residue, witness, transport, replay, provenance, and local name rows. It contains no additional ledger entry for physical adequacy, physical-fit resolution, objectivity completion, or a detached mathematical classifier certificate.*

Proof. Start with the methodology-provenance nonescape theorem. It keeps the methodology rows G, W, H, C, P, N inside the residue route rather than allowing them to export adequacy or objectivity completion.

Next combine carrier row readback with public-consumer factorization. The public consumer sees signature comparison and sibling fit surfaces through M, S, G, W, H, C, P, N, reads the residue row R only on that route, and then reaches the local name N.

It remains to rule out a second ledger row. Classifier exactness shows that acceptance preserves exactly $M, S, G, R, W, H, C, P, N$ under the local residue route. Since the carrier has no coordinate outside those rows, the NameCert ledger is exhausted by methodology, signature, residue, witness, transport, replay, provenance, and local naming. □

Theorem 1213.23 (Reality-constrained signature residue gap-row exhaustion)**.** *Let $Z = (M, S, G, R, W, H, C, P, N$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. Every public residue read that reaches the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *row must expose the gap row G, the residue row R, and the witness route W as* BHist*-carrier data. In particular, the read cannot collapse the packet to mathematical classifier agreement M alone, nor can it replace G, R, W by a detached adequacy, physical-fit resolution, or objectivity-completion entry.*

Proof. Start with the carrier of Definition 1213.1. Its displayed BHist packet contains G, R, W as named coordinates between the classifier and the transport-replay rows.

Apply the NameCert obligations and nonescape theorem, Definition 1213.2 and Definition 1213.6. The obligations make residue exposure a local certificate condition, while nonescape prevents the same packet from exporting adequacy or objectivity completion outside the displayed route.

Use witness-route totality and sibling-route exactness, Definition 1213.9 and Definition 1213.11. They force the public route to pass through the gap, residue, and witness coordinates before a consumer reaches N.

Finally combine carrier row readback with the NameCert ledger theorem, Definition 1213.14 and Definition 1213.22. Since the ledger has no detached mathematical-classifier certificate and no hidden adequacy row, a public residue read cannot be reduced to M alone. □

Theorem 1213.24 (Reality-constrained signature residue physical signature separation)**.** *In every accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet $Z = (M, S, G, R, W, H, C, P, N)$, the physical-signature row S is a visible carrier coordinate independent of mathematical classifier agreement M. Any public adequacy-facing or physical-fit-facing claim must display S together with G, R, W, H, C, P, N before it may cite the residue packet; agreement in M alone is not a substitute for the physical-signature comparison.*

Proof. Read M and S from the carrier of Definition 1213.1. The carrier lists them as different BHist coordinates: M records mathematical classifier agreement, while S records physical signature comparison.

Apply the NameCert obligations, nonescape, and mathematical-agreement refusal, Definition 1213.2, Definition 1213.6, and Definition 1213.12. These results require classifier separation and prohibit export of adequacy, physical-fit resolution, or objectivity completion from M alone.

Next use witness-route totality and sibling-route exactness, Definition 1213.9 and Definition 1213.11. The accepted route carries the sibling physical-fit surfaces through S, G, W, H, C, P, N before the residue row can be read.

Carrier row readback and public-consumer factorization then identify the visible public surface. Because the same read must expose S before it can use the residue packet as a physical-fit obstruction, mathematical classifier agreement cannot absorb the physical-signature coordinate. □

Theorem 1213.25 (Reality-constrained signature residue witness replay stability)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. The transport row H and replay row C preserve the residue witness route componentwise: under the recorded* hsame *transport in H and displayed* Cont *replay in C, the rows $M, S, G, R, W, H, C, P, N$ remain on the same public residue route, and the witness component W cannot be replayed as a route omitting G, R, or the sibling physical-signature surface.*

Proof. Begin with the carrier and the NameCert obligations. The carrier supplies the rows $M, S, G, R, W, H, C, P, N$, and the obligations require H and C to preserve the displayed residue exposure, provenance, and local naming rows.

Apply nonescape together with witness-route totality, Definition 1213.6 and Definition 1213.9. Nonescape keeps the route inside the residue packet, while witness-route totality makes W a total public route through the gap and residue rows.

Use sibling-route exactness to keep the physical-signature, open-fit, relational-objectivity, and methodology-ledger dependencies on the same route. The hsame data in H can transport only the displayed coordinates, and the Cont data in C can replay only the displayed continuation route.

Finally appeal to carrier row readback and classifier exactness. They identify the transported and replayed read with preservation of exactly $M, S, G, R, W, H, C, P, N$, so no componentwise replay can remove G, remove R, detach W, or hide the physical-signature surface behind mathematical classifier agreement alone. □

Theorem 1213.26 (Reality-constrained signature residue fit readback)**.** *Every public fit-facing read of an accepted* RealityConstrainedSignatureResidue$^{\uparrow}$ *packet is anchored in the* BHist *rows*

$$M, S, G, R, W, H, C, P, N.$$

The read may cite RealityConstrainedSignatureFit$^{\uparrow}$, RealityConstrainedOpenFit$^{\uparrow}$, RelationalObjectivity$^{\uparrow}$, *and* RealityConstrainedMethodologyLedger$^{\uparrow}$ *only through the displayed* Pkg *row P and the same witness route W; it cannot read a physical-fit conclusion from the mathematical classifier row M alone.*

Proof. Begin with carrier row readback, Definition 1213.14. It projects every public residue read to $M, S, G, R, W, H, C, P, N$.

Apply public-consumer factorization and sibling-route exactness, Definition 1213.15 and Definition 1213.11. They force the fit-facing consumer through the sibling surfaces recorded by M, S, G, W, H, C, P, N before the residue row R is read.

Finally use mathematical-agreement refusal, Definition 1213.12. Agreement in M is not a physical-fit conclusion, so the readback remains anchored in the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 1213.27 (Reality-constrained signature residue constraint evidence display)**.** *For every accepted* RealityConstrainedSignatureResidue$^{\uparrow}$ *packet $Z = (M, S, G, R, W, H, C, P, N)$, the reality-constraint evidence used by the residue row is displayed through S, G, R, W and preserved through H, C, P, N. A consumer may not replace those rows by unrecorded observational evidence, host adequacy judgment, or detached methodology provenance.*

Proof. Read the physical-signature separation theorem, Definition 1213.24. It makes S visible independently of M before any adequacy-facing claim can cite the residue packet.

Use gap-row exhaustion and methodology provenance nonescape, Definition 1213.23 and Definition 1213.19. They require G, R, W and keep methodology provenance inside the residue route.

The transport and replay rows are fixed by witness replay stability. Thus H and C preserve the displayed evidence route, while P and N package and name it without adding an unrecorded observational coordinate. □

Theorem 1213.28 (Reality-constrained signature residue unrecorded-evidence nonescape)**.** *No accepted* RealityConstrainedSignatureResidue$^{\uparrow}$ *consumer may export unrecorded reality-constraint evidence, physical adequacy, or objectivity completion from the route $M, S, G, R, W, H, C, P, N$. Any such export must be supplied by a separate accepted sibling packet and must still pass through the displayed* hsame, Cont, Pkg, *and* NameCert *rows when it is cited by the residue packet.*

Proof. Start with constraint evidence display, Definition 1213.27. The reality-facing evidence available to the residue packet is exactly the displayed physical-signature, gap, residue, witness, transport, replay, provenance, and local naming route.

Apply consumer adequacy refusal and methodology provenance nonescape, Definition 1213.10 and Definition 1213.19. They prevent the residue packet from exporting adequacy, fit resolution, objectivity completion, or detached methodology provenance.

Finally use fit readback, Definition 1213.26. A positive sibling claim may be cited only through the displayed route, so unrecorded evidence cannot leave the packet as if it were part of the local $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ surface. □

Theorem 1213.29 (Reality-constrained signature residue source exactness)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet. Suppose a public residue read of Z exposes the methodology-ledger, open-fit, signature-fit, and relational-objectivity sibling rows together with the displayed* BHist *carrier, the local* hsame *transport row, the* Cont *replay row, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *support. Then its source surface is exactly the source surface of* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$*: no off-ledger physical-signature row, objectivity row, fit row, or detached adequacy witness can enter the packet.*

Proof. Begin with carrier admission and classifier exactness, Definition 1213.20 and Definition 1213.21. They identify the admitted carrier and the accepted classifier with preservation of the rows $M, S, G, R, W, H, C, P, N$ under the local residue route.

Next use the NameCert ledger and gap-row exhaustion, Definition 1213.22 and Definition 1213.23. The ledger has no extra entry for adequacy, physical-fit resolution, objectivity completion, or detached mathematical agreement, while the public read must display G, R, W before it reaches the local name.

Finally combine sibling-route exactness with fit readback. The methodology-ledger, open-fit, signature-fit, and relational-objectivity siblings can be cited only through the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. Since those rows already exhaust the route, no off-ledger physical-signature, objectivity, or fit source can be added to the residue packet. □

Theorem 1213.30 (Reality-constrained signature residue transport closure)**.** *Let $Z = (M, S, G, R, W, H, C, P, N)$ be an accepted* $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$ *packet, and let Z' be obtained by an* hsame*-compatible transport that preserves the methodology-ledger, open-fit, signature-fit, and relational-objectivity sibling rows on the displayed residue route. Then Z' again satisfies carrier admission, classifier exactness, the* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$ *ledger, and gap-row exhaustion for the same residue packet.*

Proof. Start with witness replay stability. The hsame row transports only the displayed residue route, and the Cont row replays the same continuation surface through $M, S, G, R, W, H, C, P, N$.

Carrier admission is preserved because the transported packet still exposes the same finite BHist rows. Classifier exactness is preserved because the classifier compares exactly those rows and the assumed transport keeps the four sibling rows on the route.

The ledger and exhaustion claims then follow from the source-exactness theorem and the existing NameCert ledger and gap-row exhaustion theorems. Since transport does not add a carrier coordinate, it cannot create a new physical-signature, objectivity, fit, adequacy, or detached mathematical-agreement row. □

Theorem 1213.31 (Reality-constrained signature residue Lean-facing unblock surface)**.** *The local theorem surface consisting of carrier admission, classifier exactness, the* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}}$

ledger, gap-row exhaustion, source exactness, and transport closure is sufficient as the Lean-facing root-unblock handoff for RealityConstrainedSignatureResidue$^{\uparrow}$. *It presents the carrier, classifier, ledger, gap, source, and transport obligations without adding any off-route physical-signature, objectivity, fit, adequacy, or detached mathematical classifier premise.*

Proof. Read the first four rows from carrier admission, classifier exactness, NameCert ledger, and gap-row exhaustion. These theorems give the admitted BHist carrier, the exact accepted classifier, the exhausted local ledger, and the mandatory gap-residue-witness route.

Source exactness supplies the boundary condition for the handoff. It says that the methodology-ledger, open-fit, signature-fit, and relational-objectivity sibling rows enter only through the displayed BHist, hsame, Cont, Pkg, and local NameCert support.

Transport closure supplies the stability condition. Any hsame-compatible transport preserving those sibling rows preserves the four earlier obligations. Thus the six named results form a compact local surface for downstream Lean work on the chapter without importing an off-ledger source or an unrecorded adequacy premise. □

Closure status for NameCert$_{\text{RealityConstrainedSignatureResidue}^{\uparrow}}$.

Constructive story: The reality-constrained signature residue is generated from a $BHist$ ledger that records methodology rows, physical-signature rows, gap rows, residue rows, witness routes, and explicit separation from mathematical agreement alone. The carrier first reads the residue row under the named constraint, then uses $hsame$ to keep that residue stable across equivalent histories. Cont extensions replay witness and methodology rows only when the same residue carrier remains visible. The chapter develops this packet through non-escape, scope lock, sibling-route exactness, adequacy refusal, methodology totality, carrier admission, classifier exactness, gap-row exhaustion, physical separation, and witness replay, all without converting the residue into host reality.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedSignatureResidue}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1213.1, Definition 1213.8, Definition 1213.4, Definition 1213.21, Definition 1213.23, Definition 1213.24, and Definition 1213.25 bind the carrier, dependency handoff, classifier exactness, gap-row exhaustion, physical separation, and replay-stability surface.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedSignatureResidue}^{\uparrow}}$, axiomClean)

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureResidueUp.taste gate`

Lean target: `BEDC.Derived.RealityConstrainedSignatureResidueUp.RealityConstrainedSignatureResidueTasteGate\_singl`

Bridge status: none

Origin: ai

Not claimed: It does not claim physical signature identity, mathematical agreement outside the carrier, or consumer adequacy beyond the recorded witness route.

Upgrade path: The next paper-side grade requires a public consumer theorem or bridge-facing route over the same scoped carrier, dependency, classifier, gap, separation, and replay surface.

Chapter 1214

A Concrete Naming Certificate for $\mathsf{ScientificObject}^{\uparrow}$

The $\mathsf{ScientificObject}^{\uparrow}$ packet realizes the scientific-object surface of Definition 9462.922 and Definition 9462.923 as finite BEDC NameCert data. It sits after the physical-model audit of chapter 1129, the truth-certificate route of chapter 1144, the open-fit packet of chapter 1137, the domain truth certificate of chapter 1181, and the inter-history transport seal of chapter 1201. These siblings provide audit, truth, finite-fit, domain, and invariant rows; this packet binds them into one object-facing carrier before any consumer reads a scientific object as mature.

Definition 1214.1 (Scientific object carrier)**.** *A* $\mathsf{ScientificObject}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, K, I, L, B, G, A, T, O, D, H, C, P, N)$$

with the following displayed rows:

1. *R records the source histories, measurement rows, continuations, or instrument traces that generate the object.*
2. *K is the classifier row specifying which records count as the same object-state presentation.*
3. *I is the inter-history invariant row, read through the transport and invariant discipline of* $\mathsf{InterHistInvariant}^{\uparrow}$*.*
4. *L is the empirical ledger row recording compression, averaging, calibration, fitting, projection, residue, and observed failure data.*
5. *B is the bridge row to a mathematical or scientific model, including the optional model-facing route through* $\mathsf{OpenFitPacket}^{\uparrow}$ *and* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$*.*
6. *G is the gap-report row listing unclosed prediction, domain, bridge, and falsification commitments.*
7. *A is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row exposing source traces, observed signatures, model signatures, classifier comparison, tower data, failure surface, and audit strength.*
8. *T is the* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *row exposing the truth-certificate source, signature, classifier, tower, stability, descent, invariant, ledger, and failure fields.*
9. *O is the* $\mathsf{OpenFitPacket}^{\uparrow}$ *row exposing finite fit, finite defeat, continuation scope, and open residue.*

10. *D is the* $\mathsf{DomainTruthCertificate}^{\uparrow}$ *row exposing the domain in which the object-facing truth claim may be read.*
11. *H records componentwise* hsame *transport for $R, K, I, L, B, G, A, T, O, D$ and the local certificate row.*
12. *C records displayed* Cont *replay from record admission through classifier, invariant, ledger, bridge, gap report, audit, truth, open fit, domain, transport, provenance, and local name.*
13. *P is the* Pkg *provenance over* $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{DomainTruthCertificate}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{ScientificObject}^{\uparrow}}$ *row.*

The carrier has no coordinate for bare object existence, unledgered empirical fit, hidden model identity, global synchronization, or final physical truth outside the displayed rows.

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectUp single carrier alignment`

Definition 1214.2 (Scientific object NameCert obligations)**.** *For the carrier of Definition 1214.1, the local* $\mathsf{NameCert}_{\mathsf{ScientificObject}^{\uparrow}}$ *surface consists exactly of carrier admission for S, record exposure through R, classifier exposure through K, inter-history invariant exposure through I, empirical-ledger exposure through L, bridge exposure through B, gap-report exposure through G, audit exposure through A, reality-constrained truth exposure through T, open-fit exposure through O, domain exposure through D, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 (definition) `BEDC.Derived.ScientificObjectUp.ScientificObjectNameCertObligations`

Theorem 1214.3 (Scientific object gap-report ledger)**.** *Every accepted* $\mathsf{ScientificObject}^{\uparrow}$ *packet exposes its gap report G before the local name N is readable. A consumer may not replace G by the empirical ledger L, the model bridge B, the physical-model audit A, the truth-certificate row T, or the domain certificate D.*

Proof. Project the carrier of Definition 1214.1. The gap report is a named coordinate of the accepted packet, separate from the record, classifier, invariant, ledger, bridge, audit, truth, open-fit, and domain rows.

Apply Definition 1214.2. The local NameCert surface lists G as an obligation before provenance and local naming are available. The empirical ledger records what has been fitted or compressed, the bridge records model correspondence, the audit records signature comparison, the truth certificate records a reality-constrained claim, and the domain certificate records where that claim may be read.

None of those rows lists the unclosed prediction, domain, bridge, and falsification commitments as the gap-report coordinate. Therefore a consumer that omits G reads only a projection of the scientific-object packet, not an accepted $\mathsf{ScientificObject}^{\uparrow}$ object name. □

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectGapReportLedger`

Theorem 1214.4 (Scientific object empirical-bridge separation)**.** *In every accepted* $\mathsf{ScientificObject}^{\uparrow}$ *packet, the empirical ledger L and bridge row B are distinct obligation surfaces. The bridge may use* $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$, *and* $\mathsf{DomainTruthCertificate}^{\uparrow}$ *routes, but it does not discharge record exposure, invariant exposure, ledger residue, audit strength, or the gap report by itself.*

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectEmpiricalBridgeSeparation`

Proof. Read the carrier coordinates. The empirical ledger L records compression, averaging, calibration, fitting, projection, residue, and observed failure data, while B records the model-facing bridge.

The sibling rows are downstream supports for bridge discipline. $\mathsf{OpenFitPacket}^{\uparrow}$ supplies finite-fit and finite-defeat visibility, $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ supplies the truth-certificate route, and $\mathsf{DomainTruthCertificate}^{\uparrow}$ supplies the allowed domain of the truth read. These rows do not identify the bridge with the empirical ledger or with the inter-history invariant.

The NameCert obligations keep $R, K, I, L, B, G, A, T, O, D$ as separate rows. Thus an accepted object may have a bridge only together with the ledger, audit, invariant, and gap-report surfaces that make the scientific-object name non-vacuous. □

Theorem 1214.5 (Scientific object NameCert surface)**.** *Every accepted* $\mathsf{ScientificObject}^{\uparrow}$ *object read factors through*

$$R, K, I, L, B, G, A, T, O, D, H, C, P, N.$$

The consumer receives a finite record, classifier, invariant, empirical ledger, bridge, gap report, audit, truth-certificate, open-fit, domain, transport, replay, provenance, and local NameCert packet. It receives no bare scientific object, unrecorded empirical adequacy, host-model identity, final truth oracle, or hidden bridge completion.

Proof. Start from the obligation surface of Definition 1214.2. It exhausts the local certificate by the displayed record, classifier, invariant, ledger, bridge, gap-report, audit, truth, open-fit, domain, transport, replay, provenance, and local-name rows.

Use Definition 1214.3 to keep unclosed commitments visible, and use Definition 1214.4 to keep empirical fit and bridge status distinct.

The structural rows H, C, P, N transport, replay, source, and name the same displayed packet. They do not create a hidden object coordinate or an unrecorded truth route. Hence every accepted object read factors through the stated finite route. □

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectTasteGate single carrier alignment`

Theorem 1214.6 (Scientific object record-classifier admission)**.** *Every accepted* $\mathsf{ScientificObject}^{\uparrow}$ *packet first exposes its record row R and classifier row K. The carrier transport through H and the displayed replay through C are available before any consumer reads the empirical ledger L, bridge row B, truth row T, domain row D, gap report G, provenance row P, or local name row N.*

Proof. Project the carrier of Definition 1214.1. The displayed packet starts with R and K, and its structural rows H and C record componentwise hsame transport and Cont replay for the same finite data.

Apply Definition 1214.2. Carrier admission, record exposure, and classifier exposure are listed before the later ledger, bridge, gap-report, audit, truth, open-fit, domain, provenance, and local-name reads. Thus a consumer cannot reach the object-facing rows while bypassing the record and classifier admission surface. □

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectRecordClassifierAdmission`

Theorem 1214.7 (Scientific object truth-domain gap scope)**.** *In every accepted* $\mathsf{ScientificObject}^{\uparrow}$ *packet, the truth row T, domain row D, and gap-report row G remain separate scoped obligations. The reality-constrained truth route is readable only through the domain row, and neither row discharges the gap report.*

Proof. Use Definition 1214.3. The gap report G is a named coordinate before the local name and cannot be replaced by the empirical ledger, bridge, audit, truth, or domain rows.

Use Definition 1214.4. The reality-constrained truth row and the domain certificate row support the model-facing route, but the bridge does not discharge record exposure, invariant exposure, ledger residue, audit strength, or the gap report.

Therefore T supplies only the reality-constrained truth surface, D supplies the domain in which that surface may be read, and G remains the separate report of unclosed prediction, domain, bridge, and falsification commitments. □

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectTruthDomainGapScope`

Theorem 1214.8 (Scientific object obligation-closure package)**.** *The carrier, record-classifier admission, gap-report ledger, empirical-bridge separation, truth-domain scope, NameCert surface, falsifiability row, and independence row together provide the paper-side obligation package for* $\mathsf{ScientificObject}^{\uparrow}$*. Every accepted object read is constrained by*

$$R, K, I, L, B, G, A, T, O, D, H, C, P, N$$

and by the displayed refusal of bare object existence, unledgered empirical fit, hidden model identity, global synchronization, and final physical truth outside those rows.

Proof. The carrier of Definition 1214.1 displays all finite rows in the package and states the absent coordinates. Apply Definition 1214.6 to force record and classifier admission before downstream object reads.

Apply Definition 1214.3, Definition 1214.4, and Definition 1214.7. These results keep the gap ledger, empirical bridge, truth route, and domain scope as separate obligations.

Finally apply Definition 1214.5. The falsifiability row rejects consumers that omit the gap report, empirical ledger, inter-history invariant, or reality-constrained truth rows, and the independence row records that the named siblings supply constituent surfaces rather than the full carrier. The listed rows therefore form the paper-side obligation package for $\mathsf{ScientificObject}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ScientificObjectUp.ScientificObjectObligationClosurePackage`

Theorem 1214.9 (Scientific object standard bridge surface)**.** *Every standard-facing* $\mathsf{ScientificObject}^{\uparrow}$ *read preserves the record row R, classifier row K, invariant row I, empirical ledger L, bridge row B, gap report G, audit row A, truth-certificate row T, open-fit row O, domain row D, component-wise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The bridge surface may export the public object-facing package, but it cannot collapse these rows into final truth, hidden model identity, unledgered empirical adequacy, or an off-packet standard object.*

Proof. Start with the carrier and NameCert obligations. The accepted packet exposes $R, K, I, L, B, G, A, T, O, D, H$ as named rows, and the local certificate reads each row before the public object-facing name is available.

Use the gap-report ledger, empirical-bridge separation, and truth-domain gap scope. These results keep unclosed commitments, empirical fit, bridge status, truth route, and domain scope as separate surfaces rather than allowing one row to stand for the others.

Finally apply the NameCert surface and obligation-closure package. They identify the public read with the displayed finite package and rule out bare object existence, hidden bridge completion, and final physical truth outside the packet. A standard-facing bridge therefore preserves the whole row surface. □

Falsifiable prediction: Any accepted ScientificObjectUp consumer that exports a mature scientific object while omitting the gap report, empirical ledger, inter-history invariant, or reality-constrained truth rows refutes the packet.

Independence witness: physicalmodelaudit, realityconstrainedtruthcert, openfitpacket, domaintruthcertificate, interhistinvariant: none is bijective because $\mathsf{ScientificObject}^{\uparrow}$ binds records, classifier, invariant, ledger, bridge, gap report, audit, truth, open-fit, domain, transport, provenance, and local NameCert rows in one carrier, while each sibling supplies only one constituent surface.

The Lean TasteGate path `lean4/BEDC/Derived/ScientificObjectUp/TasteGate.lean` provides explicit $\mathsf{Nontrivial}\ \mathsf{ScientificObject}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ScientificObject}^{\uparrow}$ instances. These instances distinguish record, classifier, invariant, empirical-ledger, bridge, gap-report, audit, truth-certificate, open-fit, domain, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ScientificObject}^{\uparrow}}$.

Constructive story: A $\mathsf{ScientificObject}^{\uparrow}$ packet is generated from finite record, classifier, invariant, empirical ledger, bridge, gap report, audit, truth, open-fit, domain, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.
Lean4 ✓ `BEDC.Derived.ScientificObjectUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ScientificObject}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1214.9 is the standard bridge surface, with Definition 1214.1, Definition 1214.6, Definition 1214.7, Definition 1214.5, and Definition 1214.8 binding its carrier, classifier, truth-domain gap, public export, and obligation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ScientificObject}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, final physical truth, unledgered empirical adequacy, or hidden bridge completion.

Upgrade path: Verification closure requires checked carrier admission, gap-report ledger, empirical-bridge separation, NameCert surface, TasteGate nontriviality, and field faithfulness for $R, K, I, L, B, G, A, T, O, D, H, C, P, N$.

Chapter 1215

A Concrete Naming Certificate for $\mathsf{FiniteRealSection}^{\uparrow}$

The $\mathsf{FiniteRealSection}^{\uparrow}$ packet realizes one finite section of the diagonal-limit representative of Definition 9238.3. It sits between the representative packet of chapter 1205 and the tail-alignment packet of chapter 1210, and it consumes the finite stream windows of chapter 126, the regular-rational readbacks of chapter 301, the dyadic tolerance rows of chapter 291, and the seal rows of chapter 38. The packet names a single displayed precision request and its section route; it is not a quotient stream, selected host limit, countable-choice selector, ambient Real-completeness principle, or standard bridge.

Definition 1215.1 (Finite real section carrier)**.** *A* $\mathsf{FiniteRealSection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (q, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *q is the finite precision request for the diagonal-limit section;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window exposing the selected stream read for that request;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row read through W;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational arithmetic and error budget attached to R;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the selected regular-rational readback has been repacked through D;*
6. *H records componentwise* hsame *transport for q, W, R, D, E and the local certificate row;*
7. *C records displayed* Cont *replay from the request through the window, readback, dyadic ledger, and Real seal;*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteRealSection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite-section route and its transport, replay, provenance, and naming support.

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSectionTasteGate single carrier alignment`

Theorem 1215.2 (Finite real section obligation surface)**.** *For the carrier of Definition 1215.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability for the displayed tuple* (q, W, R, D, E)*, window-readback compatibility between* W *and* R*, dyadic-tolerance exposure through* D*,* $\mathsf{Real}^{\uparrow}$ *seal exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection transport stability`

Proof. Project the carrier S. Its coordinates list one precision request, one stream window, one regular-rational readback, one dyadic ledger, one Real seal, transport, replay, provenance, and local naming rows.

The data-bearing obligations are inherited in order. Carrier admission exposes q; the classifier is stable precisely because accepted packets compare the displayed tuple (q, W, R, D, E) componentwise before any support row is read; the window-readback obligation then forces R to be the readback attached to W.

The remaining rows are downstream support. The dyadic ledger D records the tolerance row for the readback, the seal row E is reachable only after that ledger is displayed, and H, C, P, N only transport, replay, source, and name the same finite route. Thus the listed obligations exhaust the local naming surface. □

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection obligation surface`

Theorem 1215.3 (Finite real section window-readback determinacy)**.** *For an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *carrier* $S = (q, W, R, D, E, H, C, P, N)$ *of Definition 1215.1, the precision request* q *determines the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is read only through that same window. No second stream window, quotient stream, or host sequence coordinate is available inside the accepted finite-section route.*

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection window readback determinacy`

Proof. Read the carrier rows in Definition 1215.1. The only stream coordinate exposed by S is W, and the only regular-rational coordinate is R.

Apply Definition 1215.2. Its window-readback compatibility clause is the unique obligation that relates the displayed $\mathsf{StreamName}^{\uparrow}$ window of chapter 126 to the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301.

Any alternative window, quotient stream, or host sequence coordinate would have to appear either as a carrier coordinate or as a named obligation. Neither appears in the tuple or in the obligation surface, so the accepted route reads R through exactly the window determined by q. □

Theorem 1215.4 (Finite real section dyadic-seal factorization)**.** *For an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *carrier* $S = (q, W, R, D, E, H, C, P, N)$ *of Definition 1215.1, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the ordered route*

$$q \to W \to R \to D \to E.$$

Here D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the route admits no seal exposure that bypasses* D*.*

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection dyadic seal factorization`

Proof. The carrier order in Definition 1215.1 displays the finite request, the stream window, the regular-rational readback, the dyadic tolerance ledger, and then the Real seal. Thus E is not an independent coordinate of an ambient completed Real; it is the seal row reached from the preceding displayed data.

Apply Definition 1215.2. Dyadic-tolerance exposure precedes $\mathsf{Real}^{\uparrow}$ seal exposure in the listed local $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ surface. The $\mathsf{DyadicRatCore}^{\uparrow}$ chapter chapter 291 supplies the tolerance-ledger role of D, and the $\mathsf{Real}^{\uparrow}$ chapter chapter 38 supplies the seal-row role of E.

Because the accepted surface contains no second seal row and no bypass obligation, every Real-facing use of E factors through the displayed ledger D attached to the readback R. □

Theorem 1215.5 (Finite real section route consumer exactness)**.** *Every Real- or completion-facing consumer of an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *carrier sees exactly the rows*

$$q, W, R, D, E, H, C, P, N,$$

transported by hsame*, read through* Cont*, and packaged by* Pkg *and* NameCert*. Conversely, those rows assemble the accepted finite-section consumer route.*

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection route consumer exactness`

Proof. For the forward direction, start from Definition 1215.1. A consumer can project the displayed finite request, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal, transport row, replay row, provenance row, and local naming row.

Apply Definition 1215.2. The same theorem identifies the admissible consumer surface: componentwise hsame transport through H, displayed Cont replay through C, Pkg provenance through P, and local naming through N.

For the converse direction, assume precisely the nine displayed rows with the stated transport, replay, packaging, and naming support. They match the carrier coordinates and satisfy the listed local obligations, so they assemble the accepted finite-section route and expose no additional Real, completion, quotient, or host-sequence coordinate. □

Theorem 1215.6 (Finite real section non-escape)**.** *Every Real- or Completion-facing consumer of an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *packet factors through the finite route*

$$q, W, R, D, E, H, C, P, N.$$

The consumer receives a single precision request, one $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, one* $\mathsf{Real}^{\uparrow}$ *seal, and the displayed transport, replay, provenance, and local naming rows. It receives no quotient stream equality, host sequence equality, selected host limit, countable-choice selector, ambient Real completeness, or standard bridge.*

Proof. Begin with Definition 1215.1. The only Real-facing coordinate is the seal row E, and it is downstream of the finite request q, stream window W, regular-rational readback R, and dyadic tolerance ledger D.

Apply Definition 1215.2. It forces every consumer to pass through carrier admission, classifier stability, window-readback compatibility, dyadic-tolerance exposure, seal exposure, transport, replay, provenance, and local naming before using the packet.

The excluded objects require coordinates that the carrier does not contain. The rows H, C, P, N preserve and name displayed data, but they do not create quotient stream equality, host sequence equality, a selected host limit, a countable-choice selector, ambient Real completeness, or a standard bridge. Hence none of those objects can be projected from the finite-section certificate. □

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection nonescape`

Theorem 1215.7 (Finite real section transport stability)**.** *Let*

$$S = (q, W, R, D, E, H, C, P, N)$$

and

$$S' = (q', W', R', D', E', H', C', P', N')$$

be accepted $\mathsf{FiniteRealSection}^{\uparrow}$ *packets. If* H *displays componentwise* hsame *transport from* S *to* S'*, then the transported packet preserves the same precision request, the same* $\mathsf{StreamName}^{\uparrow}$ *window role, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback role, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger role, the same* $\mathsf{Real}^{\uparrow}$ *seal role, the same* Cont *replay route, the same* Pkg *provenance, and the same local* $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ *route.*

Proof. Project the two carrier tuples through Definition 1215.1. The only rows eligible for transport are the displayed request, window, readback, ledger, seal, replay, provenance, and local naming rows.

Apply Definition 1215.2. Its componentwise hsame clause says that H transports the listed carrier coordinates and not an ambient Real object or quotient representative.

The consumer side is fixed by Definition 1215.5: every Real- or completion-facing use reads precisely the transported finite route. Finally, Definition 1215.6 excludes any target outside that route, so transport preserves the same finite-section request and its local certificate. □

Theorem 1215.8 (Finite real section refinement noncollapse)**.** *For an accepted* $\mathsf{FiniteRealSection}^{\uparrow}$ *packet* $S = (q, W, R, D, E, H, C, P, N)$*, a displayed refinement of the finite request through* C *may refine the visible finite window, readback, ledger, and seal rows along the same route*

$$q \to W \to R \to D \to E.$$

It cannot create a second stream, quotient stream equality, selected host limit, countable-choice row, ambient completeness row, or standard bridge row.

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection refinement noncollapse`

Proof. Read the carrier in the order $q, W, R, D, E, H, C, P, N$ from Definition 1215.1. The replay row C is downstream support for that ordered finite request; it is not an additional source of stream, quotient, host-limit, choice, completeness, or bridge data.

Use Definition 1215.4. The Real-facing seal factors through the displayed dyadic ledger, so refining the finite request can only refine rows already sitting on the route from q to E.

It remains to rule out collapse into a larger completion surface. Apply Definition 1215.6: the excluded objects require coordinates absent from the carrier and absent from the local obligation surface. Hence displayed refinement through C preserves finite visibility without creating any of those rows. □

Theorem 1215.9 (Finite real section obligation-closure package)**.** *The carrier of Definition 1215.1, the obligation surface of Definition 1215.2, window-readback determinacy of Definition 1215.3, dyadic-seal factorization of Definition 1215.4, route consumer exactness of Definition 1215.5, non-escape of Definition 1215.6, transport stability of Definition 1215.7, and refinement noncollapse of Definition 1215.8 are exactly the finite-section obligation set for* $\mathsf{FiniteRealSection}^{\uparrow}$ *over* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. The carrier fixes the nine-row finite section over BHist. The obligation surface names the admissible local $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ rows, while window-readback determinacy and dyadic-seal factorization identify the two data dependencies that a Real-facing consumer must respect.

Route consumer exactness supplies the bidirectional packaging statement: exactly those rows are visible to a consumer and exactly those rows assemble the accepted route. Non-escape removes quotient, host-limit, choice, ambient-completeness, and bridge exits from the obligation set.

The two stability clauses close the remaining transport and refinement faces. Transport stability keeps hsame, Cont, Pkg, and NameCert support on the same finite route, and refinement noncollapse keeps displayed refinement from adding any non-finite completion row. Together these clauses exhaust the finite-section obligations named for $\mathsf{FiniteRealSection}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection obligation closure package`

Theorem 1215.10 (Finite real section public bridge)**.** *The public finite-section certificate bridges the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows to the abstract* $\mathsf{FiniteRealSection}^{\uparrow}$ *schema. The bridge route is exactly the carrier of Definition 1215.1, obligation surface of Definition 1215.2, route exactness of Definition 1215.5, non-escape of Definition 1215.6, transport stability of Definition 1215.7, refinement noncollapse of Definition 1215.8, and TasteGate alignment recorded by the existing finite-section carrier marker.*

Proof. The carrier exposes the four data rows in the required order: stream window, regular-rational readback, dyadic tolerance ledger, and Real seal. The obligation surface then names the only admissible local $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$ reads over those rows.

Route consumer exactness supplies the public direction of the bridge. Every Real- or completion-facing consumer sees precisely the displayed finite-section rows together with hsame transport, Cont replay, Pkg provenance, and local naming.

Non-escape, transport stability, and refinement noncollapse supply the boundary direction. They prevent quotient streams, host limits, ambient completeness, choice, and bridge-external rows from entering the certificate while preserving the same route under transport and displayed refinement. The TasteGate alignment keeps this public package attached to the single carrier schema rather than to a sibling real-section interface. □

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.FiniteRealSection public bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$**.**

Constructive story: The finite Real section is generated from a $BHist$ packet that records a finite request, stream window, regular-rational readback, dyadic tolerance ledger, and the Real-facing seal row consumed by that route. The carrier convention does not assume a completed Real object; it reads only the finite section, transports it along $hsame$, and permits Cont extensions that preserve the same finite witness. The obligation surface names the finite section and the non-escape theorem records the boundary: a consumer that needs an infinite completion witness, quotient stream, selected host limit, or ambient Real equality has left the carrier. Thus the chapter packages a finite visible section as the whole admissible certificate.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter exports the finite-section public consumer surface and its compact-metric, completion, and Real-consumer cross-chapter uses, consisting of the carrier, obligation surface, route consumer exactness, non-escape theorem, transport stability, refinement noncollapse, obligation-closure package, and

Lean4 ✓ `BEDC.Derived.FiniteRealSectionUp.taste gate`

over BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteRealSection}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: ai

Not claimed: It does not claim full Real construction, infinite-section availability, quotient-stream equality, selected host limits, or ambient completeness.

Upgrade path: The verification-axis step remains Lean bridge checking for the same finite-section bridge; the paper-axis scope already includes compact-metric, completion, and Real-consumer cross-chapter use.

Chapter 1216

A Concrete Naming Certificate for $\mathsf{RegistryExportSurface}^{\uparrow}$

The $\mathsf{RegistryExportSurface}^{\uparrow}$ packet realizes the export-control surface of Definition 9462.2249 inside the auditable-generativity programme of chapter 9462. It sits beside the cannot-claim registry packet of chapter 554, the verified-output boundary of chapter 1188, the focused finite ledger of chapter 1109, the GroundCompiler recognition surface of chapter 821, and the NameCert audit pattern of chapter 1191. Those siblings supply refusal rows, checker-facing export, focused ledger scope, machine recognition, and naming-audit discipline; this carrier binds claim registration, blocking, status, machine target, audit route, and non-smuggling rows before a registry claim may leave the philosophy package.

Definition 1216.1 (Registry export surface carrier)**.** *A* $\mathsf{RegistryExportSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, B, S, M, A, U, H, C, P, N)$$

with the following visible rows:

1. *R is the registration row, recording the registry entry in which the claim is allowed to appear;*
2. *B is the blocking row, recording the cannot-claim or refusal entry that prevents export when the claim is not licensed;*
3. *S is the status row, recording whether the claim is exported, blocked, horizon-facing, or held as a formalization target;*
4. *M is the machine-target row, recording the exact machine-facing consumer of the registry entry;*
5. *A is the audit row, recording the finite verification, closure-status, or ledger route that the machine target may inspect;*
6. *U is the non-smuggling row, recording the forbidden readings that the export must not import from prose, host semantics, model output, or hidden registry state;*
7. *H stores componentwise* hsame *transport for the registration, blocking, status, machine-target, audit, non-smuggling, continuation, provenance, and local naming rows;*
8. *C stores displayed* Cont *replay for registration lookup, blocking lookup, status inspection, machine-target export, audit replay, and non-smuggling refusal;*

9. *P is the* Pkg *provenance over* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{LargeModelOutputVerifier}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{GroundCompilerRecognition}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an unregistered claim, a private override of B, a hidden machine target, or a prose-only export route.

Lean4 (definition) `BEDC.Derived.RegistryExportSurfaceUp.RegistryExportSurfaceUp`

Theorem 1216.2 (Registry export surface NameCert obligations)**.** *For the carrier of Definition 1216.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission for $E = (R, B, S, M, A, U, H, C, P, N)$;*
2. *registration and blocking coverage through R and B;*
3. *status exactness through S, requiring exported, blocked, horizon-facing, and formalization-target readings to remain distinct;*
4. *machine-target and audit exposure through M and A;*
5. *non-smuggling accountability through U, requiring every forbidden reading to remain visible at the export boundary;*
6. *transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the finite packet of Definition 1216.1. The rows R, B, S, M, A, U already separate registration, refusal, status, machine target, audit route, and forbidden readings.

Registration and blocking coverage are read first. A consumer may export a claim through R or block it through B, but it cannot erase the blocked case by reclassifying it as an exported registry entry.

The status row S then determines which registry reading is available to the machine target M. The audit row A supplies the finite route that the target may inspect, while the non-smuggling row U records the readings that remain forbidden.

Finally H, C, P, N transport, replay, package, and name the same rows. Since the carrier has no further coordinate, the six obligation rows exhaust the local $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ surface. □

Theorem 1216.3 (Registry export surface non-escape)**.** *Let*

$$E = (R, B, S, M, A, U, H, C, P, N)$$

be an accepted $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet. Any public consumer that exports a registry claim must read the registration row R, blocking row B, status row S, machine-target row M, audit row A, non-smuggling row U, transport row H, replay row C, provenance row P, and local naming row N. The export cannot be recovered from prose synthesis, model output, hidden registry state, or a machine target that bypasses the blocking row.*

Proof. Use the NameCert obligations of Definition 1216.2. They place registration, blocking, status, machine-target export, audit replay, and non-smuggling accountability inside one finite packet.

If a route omits B, the route no longer knows whether the claim is blocked by a registry entry. If it omits U, the route no longer records which prose, host-semantic, model-output, or hidden-state readings are forbidden.

The rows H, C, P, N bind these checks to the same displayed packet. Hence every accepted export remains inside the listed rows, and any route that bypasses them exits the $\mathsf{RegistryExportSurface}^{\uparrow}$ carrier rather than exporting a BEDC registry claim. □

Theorem 1216.4 (Registry export surface status exactness)**.** *Let*

$$E = (R, B, S, M, A, U, H, C, P, N)$$

be an accepted $\mathsf{RegistryExportSurface}^{\uparrow}$ BHist *packet with* hsame *transport* H*,* Cont *replay* C*, provenance package* P*, and local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N*. The status row* S *keeps the exported, blocked, horizon-facing, and formalization-target readings as separate registry statuses: no* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *audit may identify any two of these readings while preserving the displayed registration row* R*, blocking row* B*, machine-target row* M*, audit row* A*, and non-smuggling row* U*.*

Proof. Unpack the carrier of Definition 1216.1. The row S is the only displayed coordinate assigned to export status, and the definition lists exported, blocked, horizon-facing, and formalization-target readings as the alternatives recorded there.

Apply the NameCert obligations of Definition 1216.2. Their third row requires status exactness through S, and their surrounding rows keep $R, B, M, A, U, H, C, P, N$ in the same finite packet.

Finally use the non-escape theorem, Definition 1216.3. A consumer that tries to merge two status readings must still read B and U through the same transport and replay rows; if it bypasses either row, it no longer exports from the $\mathsf{RegistryExportSurface}^{\uparrow}$ carrier. Thus the four status readings remain distinct for every accepted packet audit. □

Theorem 1216.5 (Registry export surface machine-target audit route)**.** *Let*

$$E = (R, B, S, M, A, U, H, C, P, N)$$

be an accepted $\mathsf{RegistryExportSurface}^{\uparrow}$ BHist *packet. Every machine-target export from* E *factors through the displayed machine target* M*, audit row* A*, transport row* H*, replay row* C*, provenance package* P*, and local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N*, and this factorization cannot bypass the blocking row* B*.*

Proof. Read the carrier in Definition 1216.1. The machine-facing consumer is not an ambient parameter: it is the row M, and the route it may inspect is the row A together with H, C, P, N.

The NameCert obligations of Definition 1216.2 place machine-target and audit exposure in one row, and they place transport, replay, provenance, and local naming in the final row. These rows therefore describe the finite route from the machine target back to the named packet.

The blocking row is still part of the same export. By Definition 1216.3, any public consumer exporting a registry claim must read B as well as M, A, H, C, P, N. A purported machine target that reaches the claim without B is outside the accepted carrier rather than a machine-target export from E. □

Theorem 1216.6 (Registry export surface non-smuggling ledger exhaustion)**.** *Let*

$$E = (R, B, S, M, A, U, H, C, P, N)$$

be an accepted $\mathsf{RegistryExportSurface}^{\uparrow}$ BHist *packet. The prose reading, host-semantic reading, model-output reading, and hidden-registry-state reading are refused by the non-smuggling row* U *together with the displayed* hsame *transport* H*,* Cont *replay* C*, provenance package* P*, and local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N*; no further ledger coordinate is available for importing any of these readings into the registry export.*

Proof. The carrier definition names U as the row that records forbidden readings from prose, host semantics, model output, and hidden registry state. It also states that the packet contains no coordinate for a prose-only export route, hidden machine target, or private override of the blocking row.

The NameCert obligations of Definition 1216.2 make this refusal visible at the export boundary. They assign non-smuggling accountability to U and bind it to the transport, replay, provenance, and local naming rows H, C, P, N.

It remains to exclude a hidden ledger extension. Apply Definition 1216.3: every accepted public export must read U, H, C, P, N along with the registration, blocking, status, machine-target, and audit rows. Since the carrier has no additional coordinate, the listed rows exhaust the ledger route and refuse exactly the four forbidden readings named in the theorem. □

Theorem 1216.7 (Registry export surface TasteGate obligation)**.** *The Lean follow-up for* $\mathsf{RegistryExportSurface}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ RegistryExportSurface}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ RegistryExportSurface}^{\uparrow}$ *in* lean4/BEDC/Derived/Reg *with the nontriviality witness exposing at least one accepted registration-or-blocking packet and the field-faithfulness witness preserving the distinct rows* $R, B, S, M, A, U, H, C, P, N$*.*

Proof. The carrier in Definition 1216.1 is not a terminal tag: it has separate rows for registration, blocking, status, machine target, audit route, and non-smuggling accountability. A TasteGate witness must therefore exhibit a packet in which these rows are inspectable.

Field faithfulness is exactly the requirement that the classifier not collapse the rows that the NameCert obligations keep apart. In particular, R cannot substitute for B, M cannot substitute for A, and U cannot be erased after a successful export.

Thus the Lean surface named in the theorem is the appropriate follow-up site for the same finite carrier, with nontriviality and field-faithfulness aligned to the displayed rows. □

Falsifiable prediction: If the blocking row is absent, no accepted RegistryExportSurfaceUp packet exports a blocked-claim reading.

Independence witness: cannotclaim, largemodeloutputverifier, focused-multi-hist-ledger, groundcompilerrecognizerflow: the cannot-claim packet records refusal rows without machine-target export, the large-model verifier records checked output without registry status coverage, the focused multi-Hist ledger records a finite focus ledger without claim blocking, and the GroundCompiler recognizer flow records machine recognition without the full registration, blocking, audit, and non-smuggling packet. None is bijective to $\mathsf{RegistryExportSurface}^{\uparrow}$.

Theorem 1216.8 (Registry export surface public bridge route)**.** *The public* $\mathsf{RegistryExportSurface}^{\uparrow}$ *route bridges registration, blocking, status, machine-target, audit, and non-smuggling rows to the sibling surfaces chapter 554, chapter 1188, chapter 1109, chapter 821, and chapter 1191. The bridge-facing route is paper-level: it records the public dependency surface while leaving Lean bridge checking and* $\mathsf{StandardBridge}$ *discharge outside this chapter.*

Proof. Begin with Definition 1216.2. It displays the registration row R, blocking row B, status row S, machine-target row M, audit row A, non-smuggling row U, and structural rows H, C, P, N.

Use Definition 1216.3, Definition 1216.4, and Definition 1216.5. These theorems keep export, blocking, status, machine target, and audit reads inside the public carrier instead of prose or hidden registry state.

Finally apply Definition 1216.6. The non-smuggling row connects the public route to the cannot-claim, large-model-output-verifier, focused-ledger, GroundCompiler-recognition, and physical NameCert audit sibling surfaces named in the chapter introduction. Since no checked $\mathsf{StandardBridge}$ row is displayed, the result is a bridged paper route while the bridge verification axis remains unchanged. □

Theorem 1216.9 (Registry export surface index coverage)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet* $E = (R, B, S, M, A, U, H, C, P, N)$*, the registry index visible to a downstream consumer is covered exactly by the* BHist *registration row* R*, the blocking row* B*, the status row* S*, the machine-target row* M*, the audit route row* A*, the non-smuggling row* U*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N*. In particular, no BHist registry row may be exported unless the matching blocking, status, machine-target, audit, provenance, and naming rows are present in the same packet.*

Proof. Project the carrier of Definition 1216.1. The visible registry coordinates are $R, B, S, M, A, U, H, C, P, N$, with R naming the BHist registry row and N naming the local NameCert row.

Apply the NameCert obligations of Definition 1216.2. They place registration and blocking coverage in R, B, status exactness in S, machine-target and audit exposure in M, A, and non-smuggling accountability in U.

Finally use Definition 1216.8. The public route reads those same rows before any downstream export, so the index has no hidden entry outside the displayed BHist, Pkg, and $\mathsf{NameCert}$ surface. □

Theorem 1216.10 (Registry export surface export determinacy)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet* $E = (R, B, S, M, A, U, H, C, P, N)$*, the exported registry reading is determined by the displayed status row* S *together with the blocking row* B*, the machine-target row* M*, the audit route row* A*, the non-smuggling row* U*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N*. Two consumers that preserve these rows cannot produce two different export outcomes from the same BHist registry row* R*.*

Proof. Start with status exactness, Definition 1216.4. It keeps exported, blocked, horizon-facing, and formalization-target readings separate while preserving $R, B, M, A, U, H, C, P, N$.

Now read the machine-target audit route of Definition 1216.5. The machine target M receives only the displayed audit row A and cannot bypass the blocking row B.

It remains to rule out a second export by hidden state. Apply nonescape, Definition 1216.3, and the non-smuggling ledger exhaustion theorem. Since prose synthesis, model output, hidden registry state, and detached machine targets are refused by U, H, C, P, N, the preserved rows determine the public export outcome. □

Theorem 1216.11 (Registry export surface consumer provenance)**.** *Every downstream consumer of an accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet must carry the* Pkg *provenance row* P *and the local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row* N *together with the BHist registry row* R*, blocking row* B*, status row* S*, machine-target row* M*, audit route row* A*, and non-smuggling row* U*. A consumer that drops* P *or* N *is no longer a registry-export consumer of the packet.*

Proof. Read the structural rows in Definition 1216.1. The row P records provenance over $\mathsf{RegistryExportSurface}^{\uparrow}$ and its sibling export surfaces, while N records local NameCert naming for the same finite packet.

Apply Definition 1216.2. Its final obligation row binds transport, replay, provenance, and local naming to H, C, P, N, so the consumer cannot detach the registration, blocking, status, machine-target, audit, or non-smuggling rows from their package and certificate.

Finally use the public bridge route. Downstream export passes through the sibling cannot-claim, verifier, focused-ledger, recognition, and audit surfaces only with the same displayed rows. Therefore a consumer without P or N is a hidden or prose-only route, not a $\mathsf{RegistryExportSurface}^{\uparrow}$ consumer. □

Theorem 1216.12 (Registry export surface obligation rows)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet* $E = (R, B, S, M, A, U, H, C, P, N)$*, the registration row* R*, blocking row* B*, status row* S*, machine-target row* M*, audit row* A*, and non-smuggling row* U*, together with* H, C, P, N*, exhaust the public obligation surface. No registry export obligation remains outside registration, blocking, status exactness, machine-target exposure, audit replay, non-smuggling accountability, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Read the carrier of Definition 1216.1. Its visible coordinates are exactly $R, B, S, M, A, U, H, C, P, N$, and the definition excludes unregistered claims, private blocking overrides, hidden machine targets, and prose-only export routes.

Apply Definition 1216.2. The obligation list already assigns carrier admission, registration and blocking coverage, status exactness, machine-target and audit exposure, non-smuggling accountability, and transport/replay/provenance/name rows to these coordinates.

It remains to exclude a public obligation outside the list. Use Definition 1216.8: the public bridge route reads the same registration, blocking, status, machine-target, audit, and non-smuggling rows before connecting to sibling surfaces. Hence every public obligation is already one of the displayed rows. □

Theorem 1216.13 (Registry export surface ledger nonescape)**.** *No accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *export route can bypass the blocking and non-smuggling ledger. Any route that exports a registry claim must read* B *and* U *through the same status, machine-target, audit, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows that name the carrier.*

Proof. Begin with Definition 1216.3. It states that every public consumer exporting a registry claim reads $R, B, S, M, A, U, H, C, P, N$ and cannot recover the export from prose synthesis, model output, hidden registry state, or a machine target that bypasses B.

The status exactness theorem, Definition 1216.4, keeps blocked and exported readings separate while preserving R, B, M, A, U. Thus the blocking row cannot be erased by relabeling a blocked status as an exported status.

Finally use Definition 1216.6. It identifies U, H, C, P, N as the ledger route refusing prose, host-semantic, model-output, and hidden-registry-state readings. A route that bypasses B or U is therefore outside the accepted carrier, not an export route from it. □

Theorem 1216.14 (Registry export surface machine-readable blocking route)**.** *Every machine-readable* $\mathsf{RegistryExportSurface}^{\uparrow}$ *export factors through blocking, audit, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows:*

$$B, A, H, C, P, N.$$

The machine target M *may read a registry claim only along this route, with status supplied by* S *and non-smuggling accountability supplied by* U*.*

Proof. Use Definition 1216.5. It already factors every machine-target export through M, A, H, C, P, N and states that the factorization cannot bypass the blocking row B.

Next apply Definition 1216.13. The blocking row B and non-smuggling row U remain in the same ledger as the status row S, so a machine-readable export cannot read M and A while discarding the rows that decide whether the claim is blocked or forbidden.

The bridge route of Definition 1216.8 supplies the public consumer side. It connects the displayed registry rows to the sibling export surfaces without adding a hidden machine target. Therefore every machine-readable export factors through the listed blocking, audit, transport, replay, provenance, and local naming rows. □

Theorem 1216.15 (Registry export surface blocking-audit factorization)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet*

$$E = (R, B, S, M, A, U, H, C, P, N),$$

each exported machine target factors through the registration row R, blocking row B, status row S, machine-target row M, audit row A, non-smuggling row U, transport row H, replay row C, provenance row P, and local $\mathsf{NameCert}$ *row N. In particular, the audit route for a machine target is not a standalone M, A projection: it is the displayed blocking-audit route*

$$R, B, S, M, A, U \longmapsto H, C, P, N.$$

Proof. Project the carrier of Definition 1216.1. Its visible rows already list registration, blocking, status, machine target, audit, non-smuggling, transport, replay, provenance, and local naming as the complete export surface.

Apply Definition 1216.14. It forces every machine-readable export through B, A, H, C, P, N, with M supplying the machine target and S, U supplying status and non-smuggling accountability.

Now use Definition 1216.13. A route that deletes B or U is not an accepted registry export, while Definition 1216.2 binds R, S, M, A to the same local certificate rows.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming keep the displayed blocking-audit route componentwise, so no hidden registry state, prose-only export, or detached machine target remains. □

Theorem 1216.16 (Registry export surface status machine-target coverage)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet $E = (R, B, S, M, A, U, H, C, P, N)$, every exported, blocked, horizon-facing, or formalization-target machine read is covered by the same displayed rows $R, B, S, M, A, U, H, C, P, N$. The status row S chooses the public status, the machine-target row M chooses the machine-facing consumer, and the audit row A supplies the route that consumer may inspect; none of these rows can cover a target after removing blocking, non-smuggling, transport, replay, provenance, or local naming.*

Proof. Start with status exactness, Definition 1216.4. It keeps exported, blocked, horizon-facing, and formalization-target readings distinct while preserving $R, B, M, A, U, H, C, P, N$.

Apply the blocking-audit factorization theorem above. It places the machine-facing read inside the complete route $R, B, S, M, A, U, H, C, P, N$, so status coverage and machine-target coverage are the same displayed packet rather than two independent projections.

Finally use the non-smuggling ledger exhaustion theorem, Definition 1216.6. It refuses prose, host-semantic, model-output, and hidden-registry-state readings, so every covered machine target remains local to the accepted carrier and its local $\mathsf{NameCert}$ row. □

Theorem 1216.17 (Registry export surface machine-audit totality)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet*

$$E = (R, B, S, M, A, U, H, C, P, N),$$

each machine audit export reads the machine-target row M, audit row A, transport row H, continuation row C, provenance row P, and local name row N together with the blocking row B and non-smuggling row U. No machine audit export is available from M, A, H, C, P, N after deleting B or U.

Proof. Project the carrier E. The machine-readable route of Definition 1216.14 gives the displayed audit path through B, A, H, C, P, N, with M supplying the machine target and S, U supplying status and non-smuggling accountability.

Now apply the ledger nonescape theorem. It keeps B and U attached to any route that exposes a registry claim to a machine reader.

Finally use status exactness. The status row S may classify the claim as exported, blocked, horizon, or formal target, but the machine audit export remains a read of the same displayed rows. Thus M, A, H, C, P, N never form a standalone export after B or U is erased. □

Theorem 1216.18 (Registry export surface status-blocking exhaustion)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet $E = (R, B, S, M, A, U, H, C, P, N)$, the exported, blocked, horizon, and formal-target statuses are exhausted by the status row S. Changing among these four public statuses cannot erase the blocking row B, and any export route that lacks B is outside the accepted registry surface.*

Proof. Read the status row through Definition 1216.4. It is the displayed row that separates exported and blocked registry reads while preserving the registration, machine-target, audit, and non-smuggling context.

The machine-readable blocking route extends the same separation to machine consumers. It requires B before M can be read as an export target.

The ledger nonescape theorem excludes the remaining case. A route that keeps a status label but removes the blocking row is not an accepted export route from E; it is a hidden-registry-state or prose-only route, both of which the carrier refuses. □

Theorem 1216.19 (Registry export surface formal-target boundary)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet $E = (R, B, S, M, A, U, H, C, P, N)$, a formalization-target status in S exports only target metadata together with the audit route through M, A, H, C, P, N and the blocking and non-smuggling rows B, U. It does not export a checked theorem, a checked bridge, or any theorem-level evidence not already present as registry metadata.*

Proof. Begin with the status row S. A formalization-target status is a registry status, so it is read through the same carrier rows as exported, blocked, and horizon statuses.

Apply machine-audit totality. The target metadata can be exposed to a machine reader only with M, A, H, C, P, N and with B, U still attached.

There is no carrier coordinate for a checked theorem or checked bridge in E. The public bridge route is explicitly paper-level, and the closurestatus leaves bridge checking outside the chapter. Therefore the formal-target export boundary is metadata plus audit route, not theorem or bridge evidence. □

Theorem 1216.20 (Registry export surface public discharge)**.** *For every accepted* $\mathsf{RegistryExportSurface}^{\uparrow}$ *packet*

$$E = (R, B, S, M, A, U, H, C, P, N),$$

the public registry-export route is unique: it is the route through the displayed BHist *packet, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ *row that keeps the status row S, machine-target row M, blocking row B, audit row A, and non-smuggling row U together. Any putative public route that deletes one of S, M, B, A, U or bypasses H, C, P, N is outside the accepted carrier rather than a second registry-export discharge.*

Proof. Project the accepted carrier. By Definition 1216.1, the only public coordinates available to an export are $R, B, S, M, A, U, H, C, P, N$, with N naming the local $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$ row.

Apply status-blocking exhaustion and machine-audit totality. The status row S decides the public registry status, while the machine reader can inspect M, A, H, C, P, N only with the blocking and non-smuggling rows B, U still attached.

Finally use the formal-target boundary together with ledger nonescape. A route that claims public export while dropping B, U, or the structural rows H, C, P, N is a prose-only or hidden-registry-state route, not an accepted export from E. Hence the displayed path through BHist, hsame, Cont, Pkg, and NameCert is the single public discharge route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegistryExportSurface}^{\uparrow}$ packet is generated from registration, blocking, status, machine-target, audit, non-smuggling, transport, replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.RegistryExportSurfaceUp.taste gate`

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature export surface is closed by the carrier rows $R, B, S, M, A, U, H, C, P, N$, the local NameCert obligations, nonescape, status exactness, machine-target audit route, non-smuggling ledger exhaustion, the TasteGate obligation, and Definition 1216.8.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegistryExportSurface}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, hidden registry state, prose-only export, or machine targets that bypass blocking.

Upgrade path: Lean closure requires checked registration, blocking, status, machine-target, audit, non-smuggling, transport, replay, provenance, and local naming rows.

Chapter 1217

A Concrete Naming Certificate for $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$

The $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ carrier realizes the standard-bridge discipline of Definition 9462.952 inside the auditable-generativity programme of chapter 9462. It sits beside the kernel acceptance witness of chapter 549, the finite reflection tuple of chapter 993, and the GroundCompiler recognizer flow of chapter 1048. Those siblings expose accepted kernel rows, displayed reflection images, and generated-recognizer admission, while this packet binds bridge encoding, decoding, round trips, theorem preservation, no-host-leak, audit transport, and local naming before a host theory may be read as a standard bridge for a BEDC object.

Definition 1217.1 (Standard bridge audit packet carrier)**.** *A* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (N, T, E, D, R, U, P, L, H, C, Q)$$

with the following displayed rows:

1. *N is the accepted BEDC object row, read through the named* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *surface for the object being bridged;*
2. *T is the host-theory row, recording the host theory only as the right-hand side of the bridge and not as a BEDC primitive;*
3. *E is the encoding row from the BEDC carrier into the host theory;*
4. *D is the decoding row from the host theory back to the BEDC carrier;*
5. *R is the paired round-trip row, containing both decode-after-encode and encode-after-decode witnesses under the licensed classifier;*
6. *U is the finite theorem-preservation row, pairing each preserved host theorem with a BEDC counterpart and a displayed translation;*
7. *P is the no-host-leak row, recording the host primitives, quotient principles, semantic equality assumptions, proof objects, and host compiler readings that do not appear on the BEDC public surface;*
8. *L is the bridge ledger row, recording unsupported fields, refused translations, theorem-preservation gaps, and bridge scope boundaries;*

9. *H stores componentwise* hsame *transport for the accepted object, host theory, encoding, decoding, round-trip, theorem-preservation, no-host-leak, ledger, continuation, provenance, and local naming rows;*
10. *C stores displayed* Cont *replay for bridge checking, theorem-pair lookup, no-host-leak inspection, and downstream audit reads;*
11. *Q is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ *row over* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, $\mathsf{FiniteReflectionTuple}^{\uparrow}$, $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

The carrier has no coordinate for a host primitive promoted to BEDC input, an infinite theorem schema, an unpaired host theorem, a hidden decoder, a private compiler route, or a bridge assertion detached from the displayed no-host-leak row.

Lean4 (definition) `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacketUp`

Theorem 1217.2 (Standard bridge audit packet non-host-leak)**.** *For every accepted* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

the public bridge read factors through the displayed no-host-leak row P. A bridge route that omits P cannot export a BEDC-recognized standard bridge, even when E, D, R, U are present.

Proof. Project the carrier of Definition 1217.1. The host theory row T, encoding row E, decoding row D, round-trip row R, and theorem-preservation row U are not enough to define a public BEDC bridge because each of them may mention host-side structure.

The no-host-leak row P records exactly which host primitives, quotient principles, semantic equalities, proof objects, and host compiler readings are barred from the BEDC public surface. It is therefore the row that separates bridge recognition from host import.

The structural rows L, H, C, Q preserve and replay the same displayed packet. Since none of them supplies a replacement for P, any route that omits P exits the accepted $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ carrier rather than exporting a standard bridge. □

Lean4 ✓ `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacket nonhost leak certificate`

Theorem 1217.3 (Standard bridge audit packet independence)**.** *The carrier of Definition 1217.1 is not bijective, row for row, with* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, $\mathsf{FiniteReflectionTuple}^{\uparrow}$, $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *finite BHist injection data, or* $\mathsf{ProofTermErasureTrace}^{\uparrow}$. *A row-preserving bijection would have to preserve accepted-object evidence, reflection-image readback, generated-recognizer admission, finite history injection, proof-term erasure visibility, encoding, decoding, two round-trip witnesses, finite theorem preservation, no-host-leak, bridge ledger, transport, replay, provenance, and local naming at once; no listed sibling carries that whole surface.*

Proof. First compare with $\mathsf{KernelAcceptanceWitness}^{\uparrow}$. Kernel acceptance witnesses expose candidate, accepted declaration, environment, purity-query, refusal, transport, replay, provenance, and local naming rows, but they do not contain host encoding, decoding, round trips, theorem-preservation pairs, or a no-host-leak row for a host theory.

Next compare with $\mathsf{FiniteReflectionTuple}^{\uparrow}$ and $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$. The reflection tuple supplies displayed image and readback rows, while the recognizer flow supplies source-channel and generated-recognizer admission. Neither records the six standard-bridge fields as one finite audit packet.

Finally compare with finite BHist injection data and $\mathsf{ProofTermErasureTrace}^{\uparrow}$. Injection data concerns how finite histories embed into a displayed history surface, and proof-term erasure traces separate proof evidence from erased public rows. They do not carry the host-relative bridge ledger E, D, R, U, P, L. Thus no row-preserving bijection can identify any listed sibling with $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$. □

Theorem 1217.4 (Standard bridge audit packet TasteGate obligation)**.** *The Lean follow-up for* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *must provide* Nontrivial $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *and* FieldFaithful StandardBridgeAu *in* lean4/BEDC/Derived/StandardBridgeAuditPacketUp/TasteGate.lean*, with the nontriviality witness exposing at least one accepted bridge packet and the field-faithfulness witness preserving the distinct rows* $N, T, E, D, R, U, P, L, H, C, Q$*.*

Lean4 ✓ `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacketTasteGate obligation`

Proof. The carrier in Definition 1217.1 is not a terminal tag. It separates the accepted BEDC object, host theory, encoding, decoding, round trips, theorem preservation, no-host-leak, ledger, transport, replay, and local NameCert rows.

A nontriviality instance must therefore exhibit a packet in which those rows are inspectable. A field-faithfulness instance must prevent the classifier from collapsing the no-host-leak row into theorem preservation, the host-theory row into BEDC input, or the round-trip row into a bare encoding or decoding row.

Thus the later TasteGate file is the appropriate Lean surface for the same carrier, and its instances must distinguish the displayed bridge rows before any bridge-checked consumer cites the packet. □

Theorem 1217.5 (Standard bridge audit packet round-trip exactness)**.** *For every accepted* StandardBridgeAuditPac *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

the two public round-trip reads are exactly the displayed reads through R, H, C, Q*. Decode-after-encode may consume* E *and* D *only after the paired round-trip row* R *names the witness, the componentwise transport row* H *preserves the accepted object and host-theory coordinates, the* Cont *row* C *replays the bridge-check request, and the provenance-and-naming row* Q *records the local packet. Encode-after-decode is governed by the same four rows in the opposite read direction. No hidden encoder, hidden decoder, host equality, or bridge assertion outside* R, H, C, Q *can supply either exactness witness.*

Proof. Project the carrier of Definition 1217.1. The rows E and D store the two bridge maps, but the carrier gives them no public exactness role by themselves; the paired round-trip row R is the only coordinate assigned to decode-after-encode and encode-after-decode witnesses.

Transport and replay are then forced through the structural rows. The row H moves the accepted object, host theory, encoding, decoding, and round-trip data componentwise, while C replays the concrete bridge-check request that asks for one of the two round-trip reads.

Finally Q records the Pkg provenance and local $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ row. Since the carrier has no coordinate for a hidden map, host equality import, or detached bridge assertion, both exactness directions are precisely the reads through R, H, C, Q. □

Lean4 ✓ `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacket round trip rows visible`

Theorem 1217.6 (Standard bridge audit packet theorem-preservation ledger)**.** *For every accepted* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

each theorem-preservation claim is a finite ledger entry whose host theorem, BEDC counterpart, translation evidence, unsupported-field status, and public admissibility are read through U, L, P*. A host theorem that has no paired BEDC counterpart in* U*, or whose required host primitive is refused by* P *or recorded as unsupported in* L*, is not exported as a preserved BEDC theorem.*

Proof. The theorem-preservation coordinate of the carrier is U. By definition it contains finite rows pairing preserved host theorems with BEDC counterparts and displayed translations, so an unpaired host theorem has no preservation row to project.

The ledger row L records unsupported fields, refused translations, theorem-preservation gaps, and bridge scope boundaries. Thus a preservation claim must also be compatible with the finite ledger entry that says the translation is supported inside the packet's bridge scope.

The no-host-leak row P supplies the public admissibility check. If the proposed preservation uses a host primitive, quotient principle, semantic equality assumption, proof object, or compiler reading barred by P, then the claim remains a host-side statement rather than a BEDC-preserved theorem. Hence theorem preservation is exactly the finite U, L, P ledger surface. □

Lean4 ✓ `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacket theorem preservation ledger`

Theorem 1217.7 (Standard bridge audit packet host-leak consumer coverage)**.** *For every accepted* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

any bridge consumer that reads E, D, R*, or* U *must first read the no-host-leak row* P*. A consumer that bypasses* P *cannot be certified by the local* $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ *row, even if it follows the displayed encoding, decoding, round-trip, and theorem-preservation coordinates.*

Proof. Apply Definition 1217.2. The public bridge read factors through P because the rows E, D, R, U may mention host-side structure and do not by themselves separate bridge recognition from host import.

Now inspect the consumer side of the carrier. The row C replays bridge checking, theorem-pair lookup, no-host-leak inspection, and downstream audit reads; the local packet row Q names the same replay surface through Pkg provenance and $\mathsf{NameCert}$ registration. Neither row replaces P with an implicit host-leak check.

Therefore every certified consumer of encoding, decoding, round trips, or theorem preservation must read P before consuming E, D, R, U. A route that omits P leaves the accepted packet surface and cannot be certified as a local standard-bridge audit consumer. □

Theorem 1217.8 (Standard bridge audit packet scoped primitive binding)**.** *For every certified bridge-audit consumer of an accepted* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

the primitive scope is exactly the scope displayed by the carrier: N *is read only through the named* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *surface;* T, E, D, R, U, P, L *are the host-relative bridge rows for host theory, encoding, decoding, paired round trips, theorem preservation, no-host-leak, and the bridge*

ledger; and H, C, Q are the only BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ *replay and provenance surface. There is no certified scope coordinate for a hidden encoder, an unpaired host theorem export, a host primitive imported as BEDC input, a private compiler route, or a bridge assertion detached from the no-host-leak row P.*

Proof. Begin with the carrier of Definition 1217.1. Its displayed coordinates name the accepted object row N, the host-relative bridge rows T, E, D, R, U, P, L, and the structural replay rows H, C, Q; no additional primitive row is present.

Apply Definition 1217.2 and Definition 1217.7. Any certified consumer that reads bridge maps, round trips, or theorem preservation must factor through P, so host primitives, host proof objects, quotient principles, semantic equalities, and host compiler readings cannot enter the public BEDC surface through an implicit side channel.

Next use Definition 1217.5. The two exactness directions are public only as reads through R, H, C, Q, so a hidden encoder, hidden decoder, host equality import, or detached bridge assertion has no accepted witness in the carrier.

Finally use Definition 1217.6. Theorem export is finite and paired through U, L, P, which rules out an unpaired host theorem and any bridge claim whose support is refused by the ledger or by the no-host-leak row. Hence every certified bridge-audit consumer has exactly the primitive boundary displayed by $N, T, E, D, R, U, P, L, H, C, Q$. □

Theorem 1217.9 (Standard bridge audit packet public bridge-audit export)**.** *For every accepted* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier*

$$S = (N, T, E, D, R, U, P, L, H, C, Q),$$

any public standard-bridge audit export exposes exactly the accepted object row N, host-theory row T, encoding row E, decoding row D, paired round-trip row R, theorem-preservation ledger U, no-host-leak row P, bridge ledger L, componentwise hsame *transport H, displayed* Cont *replay C, and local* $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ *provenance Q. It exports no host primitive, hidden decoder, unpaired theorem, quotient equality, or detached bridge assertion.*

Proof. Start from the carrier projection in Definition 1217.1. Its public rows are exactly $N, T, E, D, R, U, P, L, H, C$, with Q carrying the Pkg provenance and local $\mathsf{NameCert}_{\mathsf{StandardBridgeAuditPacket}^{\uparrow}}$ registration.

Use Definition 1217.2 together with Definition 1217.7. These force every public bridge read of E, D, R, or U through the no-host-leak row P, so no host primitive, quotient equality, proof object, or hidden compiler reading is exported as BEDC data.

Next apply Definition 1217.5. The decode-after-encode and encode-after-decode witnesses are public only as paired reads through R, H, C, Q, which excludes hidden decoders and detached bridge assertions.

Finally apply Definition 1217.6 and Definition 1217.8. The theorem-preservation surface is finite and paired through U, L, P, while the certified primitive boundary is precisely the displayed carrier boundary. Hence the public export exposes exactly the accepted bridge-audit rows and nothing outside them. □

Theorem 1217.10 (Standard bridge audit packet bridge-readback boundary)**.** *Let*

$$S = (N, T, E, D, R, U, P, L, H, C, Q)$$

be an accepted $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *carrier. If a bridge read bridgeRead is obtained by the displayed routes* Cont E D R, Cont R P *bridgeRead, and* Cont P L H, *then the public*

bridge boundary is the SemanticNameCert *surface whose carrier reads bridgeRead together with the displayed packet S, whose pattern is exactly the encode-decode route, the round-trip-to-public route, and the no-host-ledger route, and whose ledger preserves only bridgeRead, P, and Q. No host primitive, theorem object, quotient equality, hidden compiler read, or detached bridge assertion is available at that boundary.*

Proof. Start from the carrier definition in Definition 1217.1. The displayed packet fixes the rows $N, T, E, D, R, U, P, L, H, C, Q$, so a bridge-readback certificate has no coordinate beyond the accepted object, host theory, encoding, decoding, round-trip, theorem-preservation, no-host-leak, bridge ledger, transport, replay, and local naming rows.

Apply Definition 1217.9 and Definition 1217.5. The public export exposes the displayed packet, while exactness forces every decode-after-encode or encode-after-decode read through R, H, C, Q; the route Cont E D R is therefore the encode-decode component of the boundary.

Next use Definition 1217.6 together with Definition 1217.2. The theorem-preservation ledger is public only through U, L, P, and the no-host-leak row P is required before any bridge consumer reads E, D, R, or U. Thus Cont R P *bridgeRead* is the round-trip-to-public route and Cont P L H is the no-host-ledger route.

The Lean certificate '`BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacket_nonhost` realizes the same SemanticNameCert shape: the carrier records *bridgeRead* and the displayed packet, the pattern records precisely those three Cont routes, and the ledger records only *bridgeRead*, P, and Q. Hence the public bridge boundary is the stated readback surface and nothing host-side leaks across it. □

Falsifiable prediction: A purported bridge packet that omits the no-host-leak row is not an accepted StandardBridgeAuditPacketUp carrier.

Independence witness: kernelacceptancewitness, finitereflectiontuple, groundcompilerrecognizerflow, finitebhistinjection, prooftermerasuretrace: kernel acceptance records accepted declarations without host bridge fields, finite reflection records tuple image and readback without theorem-preservation pairs, GroundCompiler recognizer flow records generated-recognizer admission without a host bridge ledger, finite BHist injection records history embedding without no-host-leak, and proof-term erasure trace records erased proof visibility without encoding or decoding. None is bijective to StandardBridgeAuditPacket$^{\uparrow}$.

Closure status for NameCert$_{\text{StandardBridgeAuditPacket}^{\uparrow}}$.

Constructive story: A StandardBridgeAuditPacket$^{\uparrow}$ carrier is generated from the rows $N, T, E, D, R, U, P, L, H, C, Q$: accepted BEDC object, host-theory row, encoding, decoding, paired round trips, theorem preservation, no-host-leak ledger, bridge ledger, componentwise hsame transport, displayed Cont replay, and Pkg provenance with local NameCert registration.

Lean4 ✓ `BEDC.Derived.StandardBridgeAuditPacketUp.taste gate`

Theory closure: TheoryGate(NameCert$_{\text{StandardBridgeAuditPacket}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1217.9 and Definition 1217.10 close the public bridge-audit export and bridge-readback boundary for the displayed $N, T, E, D, R, U, P, L, H, C, Q$ rows.

Formal verification: FormalStatus(NameCert$_{\text{StandardBridgeAuditPacket}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.StandardBridgeAuditPacketUp.StandardBridgeAuditPacket\_nonhost\_leak\_certificate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close bridgeChecked verification, host primitive import, unpaired host theorem export, or any host theorem surface outside the displayed no-host-leak row.

Upgrade path: No further paper-side theory closure remains for this packet; object-specific bridge checking remains verification-axis work.

Chapter 1218

A Concrete Naming Certificate for $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$

The $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ packet realizes the object-knowledge synthesis of section 9462.23 as finite BEDC NameCert data. It uses the five-field certificate discipline from Definition 9462.664, reads certificate transport through chapter 481, and reads classifier transport through chapter 590. The carrier therefore exposes an object as a licensed name and knowledge as an auditable witness before any consumer receives a mature object-knowledge export.

Definition 1218.1 (Object-knowledge certificate carrier)**.** *An* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (N, W, S, P, K, T, L, A, C, Q)$$

with the following displayed rows:

1. *N is the object-name row, read as the licensed name of the object rather than as a bare object primitive.*
2. *W is the auditable witness row whose possession supports the knowledge claim.*
3. *S is the source row recording the histories, observations, declarations, or prior certificates from which the name and witness are admitted.*
4. *P is the pattern row naming the stable behavior, recurrence, or constructive presentation that the name claims to track.*
5. *K is the classifier row specifying when two presentations count as the same object-knowledge packet.*
6. *T is the stability row recording the displayed transport by which the name, witness, source, pattern, and classifier persist across accepted continuations.*
7. *L is the ledger row recording gaps, failed reads, refused exports, missing witness routes, and residual obligations.*
8. *A is the audit-route row exposing how a consumer checks the witness against the licensed object name.*
9. *C is the componentwise transport row over N, W, S, P, K, T, L, A and over the sibling* $\mathsf{NameCertMorphism}^{\uparrow}$ *and* $\mathsf{ClassifierMorphism}^{\uparrow}$ *routes.*

10. *Q is the local* $\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$ *row, whose five-field certificate surface is source, pattern, classifier, stability, and ledger, tied to the object-name and witness rows.*

The carrier has no coordinate for an unlicensed object, a bare truth predicate, a witness-free knowledge claim, an anonymous classifier, a hidden ledger, or a local name detached from the displayed NameCert row.

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificateTasteGate single carrier alignment`

Theorem 1218.2 (Object-knowledge NameCert obligations)**.** *For the carrier of Definition 1218.1, the local* $\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$O = (N, W, S, P, K, T, L, A, C, Q),$$

object-name licensing through N, witness exposure through W, source exposure through S, pattern exposure through P, classifier exposure through K, stability exposure through T, ledger exposure through L, audit routing through A, componentwise transport through C, and local naming through Q.

Consequently every accepted object-knowledge export is a finite packet whose object read is licensed by N and whose knowledge read is witnessed by W under the five displayed NameCert fields.

Proof. Project the carrier of Definition 1218.1. Its rows are exactly $N, W, S, P, K, T, L, A, C, Q$, so no additional object primitive, witness predicate, classifier, stability route, ledger, audit path, or local name is available to the consumer.

Read the local NameCert row. The certificate fields are S, P, K, T, L, and the object-knowledge rows N, W, A identify which licensed object is being read, which witness supports the knowledge claim, and which audit route checks that witness.

Finally use the componentwise transport row C. It preserves the displayed name, witness, five-field certificate, sibling morphism routes, and local certificate row together. Therefore an accepted export is exactly the stated finite NameCert packet, not a projection with the witness or ledger suppressed. □

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificateNameCert obligations`

Theorem 1218.3 (Object-knowledge witness exactness)**.** *Every accepted* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ *knowledge read exposes the same witness row W that the audit route A checks against the licensed object name N. Replacing W by the pattern row P, the classifier row K, the stability row T, or the ledger row L is not an accepted knowledge read.*

Proof. The carrier separates the witness row from the certificate fields. The row W records the auditable witness, while P, K, T, L record pattern, classifier, stability, and ledger data.

The audit route A is defined as a displayed check of the witness against the licensed object name. It therefore consumes N and W as rows of the same finite packet, with S, P, K, T, L supplying certificate context rather than replacing the witness.

The local NameCert row Q names the whole packet. Since C transports W componentwise with the other rows, a consumer that substitutes a certificate field for W changes the accepted packet and loses the knowledge read. □

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificate witness audit rows reflect display`

Theorem 1218.4 (Object-knowledge nonescape)**.** *No accepted* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ *packet exports object knowledge outside the displayed rows*

$$N, W, S, P, K, T, L, A, C, Q.$$

In particular, the packet does not export an unlicensed object, a witness-free knowledge claim, a classifier-free identity, an unledgered residue, or a local name outside the local $\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$ *row.*

Proof. Begin with the exact obligation surface of Definition 1218.2. It exhausts the public export by the object-name, witness, source, pattern, classifier, stability, ledger, audit-route, transport, and local-name rows.

Use Definition 1218.3 to keep the witness row from collapsing into the certificate fields. The knowledge component remains auditable only through W checked by A against N.

The sibling morphism routes preserve the displayed certificate and classifier data, but they do not add an inverse, a bare object, or a truth predicate. Hence every accepted read remains inside the finite row list, and the listed escapes are not exports of this carrier. □

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificate nonescape`

Theorem 1218.5 (Object-knowledge public export)**.** *The carrier, NameCert obligations, witness exactness, and nonescape theorems jointly export exactly the public* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ *packet*

$$N, W, S, P, K, T, L, A, C, Q.$$

Every consumer receives object-name licensing through N, witness exposure through W, source, pattern, classifier, stability, and ledger rows through S, P, K, T, L, audit routing through A, componentwise transport through C, and local naming through Q, and receives no additional object, witness-free knowledge read, anonymous classifier, hidden ledger, or detached local name.

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificate public export`

Proof. Use Definition 1218.1 to identify the public packet as the finite row family $N, W, S, P, K, T, L, A, C, Q$.

Apply Definition 1218.2. It makes those rows the full local NameCert surface, with source, pattern, classifier, stability, ledger, audit route, transport, and local naming all visible to the consumer.

Apply Definition 1218.3 and Definition 1218.4. The witness cannot be replaced by a certificate field, and no object-knowledge read escapes the displayed rows. Hence the exported public object-knowledge packet is exactly the stated row family. □

Theorem 1218.6 (Object-knowledge bridge obligation surface)**.** *Any standard bridge for* ObjectKnowledgeCertifica *must preserve the displayed rows*

$$N, W, S, P, K, T, L, A, C, Q.$$

In particular, it preserves object-name licensing through N, witness exposure through W, audit routing through A, ledger exposure through L, classifier transport through C, and the local $\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$ *row Q. It must not collapse the packet to an unlicensed object, a witness-free knowledge predicate, or a detached truth classifier.*

Proof. Start from the carrier of Definition 1218.1. The displayed bridge domain is exactly the finite row family $N, W, S, P, K, T, L, A, C, Q$, so a bridge can preserve only those rows and cannot introduce an unlicensed object coordinate, anonymous truth predicate, or detached classifier.

Apply Definition 1218.2 and Definition 1218.3. These keep object-name licensing on N, witness exposure on W, audit checking on A, and the local certificate row on Q.

Finally use Definition 1218.4 together with Definition 1218.5. The public export is exhausted by the displayed rows, while nonescape rules out witness-free knowledge, unlicensed objects, hidden ledgers, and classifier-free identity. Hence every standard bridge obligation is over the stated surface. □

Lean4 ✓ `BEDC.Derived.ObjectKnowledgeCertificateUp.ObjectKnowledgeCertificate bridge obligation surface`

Falsifiable prediction: If an object-knowledge export bypasses either the displayed five-field NameCert row or the displayed witness row, the packet is refuted.

Independence witness: namecertmorphism, classifiermorphism, semanticnamecert, namecert: none is bijective because this carrier binds object license, auditable witness, five-field certificate, classifier transport, ledger, and local naming as one packet.

The Lean follow-up must provide Nontrivial ObjectKnowledgeCertificate$^{\uparrow}$ and FieldFaithful ObjectKnowledgeCertifi instances in lean4/BEDC/Derived/ObjectKnowledgeCertificateUp/TasteGate.lean. The target BEDC.Derived.ObjectK may be cited only after those instances distinguish the object-name, witness, source, pattern, classifier, stability, ledger, audit-route, componentwise transport, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$ **.**

Constructive story: An ObjectKnowledgeCertificate$^{\uparrow}$ packet is generated from object-name, witness, source, pattern, classifier, stability, ledger, audit-route, componentwise transport, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1218.1, Definition 1218.2, Definition 1218.3, Definition 1218.4, and Definition 1218.5 are the finite public scope for object-name licensing, witness exactness, audit routing, componentwise transport, local NameCert obligations, nonescape, and public packet export for object-knowledge packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ObjectKnowledgeCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close witness-free knowledge, unlicensed objects, classifier-free identity, unledgered residue, Lean verification, or bridge checking for ObjectKnowledgeCertificate$^{\uparrow}$.

Upgrade path: No further paper-theory closure upgrade is claimed; Lean bridge checking remains separate.

Chapter 1219

A Concrete Naming Certificate for $\mathsf{ProofTraceAuditWitness}^{\uparrow}$

The $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ packet realizes the proof-discipline surface of Definition 9462.2010 as a finite NameCert carrier. It sits beside the kernel audit witness of chapter 546, the compiler trace faithfulness packet of chapter 1055, the NameCert morphism route of chapter 481, the category certificate surface of chapter 61, and the proof-term erasure trace of chapter 642. Those siblings audit accepted declarations, compiler routes, certificate morphisms, categorical rows, and proof-object erasure; this packet binds proof trace, source carrier, classifier witness, displayed transport, forbidden-resource refusal, audit verdict, provenance, and local naming before a proof row may be read as BEDC-internal evidence.

Definition 1219.1 (Proof trace audit witness carrier)**.** *A* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, S, K, H, L, F, A, P, N)$$

with the following displayed rows:

1. *T is the proof-trace row, recording the finite witness chain consumed by the proof site.*
2. *S is the source-carrier row, recording the BEDC carrier, source object, or theorem surface whose evidence is being audited.*
3. *K is the classifier-witness row, recording the classifier membership, stability, or exactness witness used by the proof trace.*
4. *H is the displayed-transport row, recording the* hsame *transport needed whenever the trace crosses a classifier, package, bridge, or history-identity surface.*
5. *L is the ledger-witness row, recording the* Cont *replay, package route, and local ledger entries consumed by the trace.*
6. *F is the forbidden-resource row, recording refusal of hidden choice, host equality as BEDC equality, quotient collapse, propositional extensional rewrite, global truth predicate, unledgered compression, and unscoped universal claim.*
7. *A is the audit-verdict row, recording whether the trace is accepted, refused, or held as a displayed obligation at the proof site.*
8. *P is the* Pkg *provenance over* $\mathsf{KernelAuditWitness}^{\uparrow}$, $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{ProofTermErasureTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ProofTraceAuditWitness}^{\uparrow}}$ *row.*
The carrier contains no coordinate for an untraced proof, an implicit transport, an erased forbidden-resource refusal, a host equality proof, or an audit verdict detached from provenance and local naming.

Lean4 (definition) `BEDC.Derived.ProofTraceAuditWitnessUp.ProofTraceAuditWitnessUp`
Lean4 ✓ `BEDC.Derived.ProofTraceAuditWitnessUp.taste gate`

Theorem 1219.2 (Proof trace audit witness NameCert obligations)**.** *For the carrier of Definition 1219.1, the local* $\mathsf{NameCert}_{\mathsf{ProofTraceAuditWitness}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for $W = (T, S, K, H, L, F, A, P, N)$; proof-trace and source-carrier exposure through T and S; classifier-witness stability through K; displayed transport and replay through H and L; forbidden-resource refusal and audit-verdict exactness through F and A; and provenance plus local naming through P and N.*

Lean4 ✓ `BEDC.Derived.ProofTraceAuditWitnessUp.ProofTraceAuditWitnessNameCertObligations`

Proof. Project the carrier of Definition 1219.1. The rows $T, S, K, H, L, F, A, P, N$ are the only available proof-audit coordinates, so a proof consumer first reads a finite trace, its source carrier, and the classifier witness that licenses the trace.

Next read the transport and ledger rows. The displayed transport row H prevents classifier crossing from being treated as host equality, while the ledger row L keeps replay, package route, and proof-site ledger entries visible in BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ form.

Finally attach the refusal and verdict rows. The forbidden-resource row F records which proof resources are unavailable, and the audit-verdict row A can be read only with that refusal row and the provenance and local name rows P, N. Repacking these rows gives the six local NameCert obligations and no additional proof coordinate. □

Theorem 1219.3 (Proof trace audit witness ledger non-escape)**.** *Every accepted* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *proof read factors through the displayed rows*

$$T, S, K, H, L, F, A, P, N.$$

In particular, deleting the displayed-transport row H, erasing the forbidden-resource row F, or detaching the audit-verdict row A from the provenance row P leaves no carrier route from the proof trace T to BEDC-internal proof evidence.

Proof. Begin with the NameCert obligations of Definition 1219.2. They require proof-trace exposure, source-carrier exposure, classifier-witness stability, displayed transport, ledger replay, forbidden-resource refusal, audit verdict, provenance, and local naming to be read as one packet.

If H is absent, the trace has no displayed way to cross a classifier, package, bridge, or history-identity surface. If F is absent, the proof site no longer records refusal of hidden choice, quotient collapse, host equality, or propositional extensional rewrite. If A is detached from P, the verdict is no longer tied to a finite proof-site provenance row.

Thus every accepted proof read remains inside the listed packet. A consumer that asks for proof evidence without one of these rows has left the $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ carrier rather than extracting a hidden proof resource from it. □

The Lean follow-up file `lean4/BEDC/Derived/ProofTraceAuditWitnessUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{ProofTraceAuditWitness}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ProofTraceAuditWitness}^{\uparrow}$ instances, preserving $T, S, K, H, L, F, A, P, N$ as separate displayed fields.

Falsifiable prediction: Deleting the displayed transport row while keeping the proof trace and verdict rows invalidates the audit witness carrier.

Independence witness: kernelauditwitness, groundcompilerauditwitness, namecertmorphism, category, prooftermerasuretrace: none is bijective because this packet uniquely binds a proof trace to displayed transport, forbidden-resource refusal, audit verdict, provenance, and local NameCert rows.

Theorem 1219.4 (Proof trace audit bridge route)**.** *Every bridge-facing proof-audit consumer of a* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *packet must preserve the scoped carrier of Definition 1219.1, the NameCert obligations of Definition 1219.2, the ledger nonescape boundary of Definition 1219.3, the TasteGate target, and the displayed sibling dependencies on KernelAuditWitness, CompilerTraceFaithfulness, NameCertMorphism, Category, and ProofTermErasureTrace rows. The route exports no untraced proof evidence, host equality rule, global proof search, kernel-soundness theorem, or proof-term-erasure correctness claim.*

Proof. Start with the scoped carrier. Its rows $T, S, K, H, L, F, A, P, N$ remain the only proof-audit coordinates available to a bridge-facing consumer.

Apply the NameCert obligations and ledger nonescape theorem. They force proof-trace exposure, source-carrier exposure, classifier-witness stability, displayed transport, ledger replay, forbidden-resource refusal, audit verdict, provenance, and local naming to be read as one finite packet.

Finally preserve the TasteGate target and displayed sibling dependencies. Since the bridge-facing route contains no coordinate for untraced proof evidence, host equality, global proof search, kernel soundness, or proof-term-erasure correctness, those objects cannot be exported by the paper bridge route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTraceAuditWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ packet is generated from proof trace, source carrier, classifier witness, displayed transport, ledger witness, forbidden-resource refusal, audit verdict, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofTraceAuditWitness}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1219.1, Definition 1219.2, Definition 1219.3, and Definition 1219.4 bind the carrier, NameCert obligations, ledger nonescape, TasteGate target, and displayed dependencies on KernelAuditWitness, CompilerTraceFaithfulness, NameCertMorphism, Category, ProofTermErasureTrace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofTraceAuditWitness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ProofTraceAuditWitnessUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, global proof search, kernel soundness, proof-term erasure correctness, or a host equality proof rule.

Upgrade path: No further paper-theory upgrade is pending; bridge checking remains a separate formal-axis task preserving the scoped carrier, NameCert obligations, ledger nonescape, TasteGate target, and displayed sibling dependencies without adding untraced proof evidence.

Chapter 1220

A Concrete Naming Certificate for $\mathsf{RegistryLayer}^{\uparrow}$

The $\mathsf{RegistryLayer}^{\uparrow}$ packet realizes the registry layer of chapter 9462 as a finite NameCert surface for package-level export control. It sits beside the selector ledger of chapter 494, the focused finite ledger of chapter 1109, the choice-recipe ledger of chapter 677, the axiom-requirement ledger of chapter 992, the witnessed-descent ledger of chapter 1103, the cannot-claim registry of chapter 554, the claim registry layer of chapter 1209, and the export surface of chapter 1216. Those siblings record selected rows, focused histories, choice recipes, axiom requirements, descent witnesses, refusal entries, single-claim registration, or export gates; this carrier binds the whole registry layer through simultaneous admit, block, transport, and package-ledger rows.

Definition 1220.1 (Registry layer carrier)**.** *A* $\mathsf{RegistryLayer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, T, G, D, S, K, U, F, H, P, N)$$

with the following displayed rows:

1. *C is the concept-registry row, naming concepts allowed to be exported as package vocabulary;*
2. *T is the theorem-registry row, naming theorem, lemma, definition, closure-status, or formal-target sites available for export;*
3. *G is the gap-registry row, naming statements whose package reading is horizon-facing or not yet licensed for export;*
4. *D is the tradition-and-science registry row, separating inherited mathematical tradition, empirical-science claims, and BEDC-specific discovery claims;*
5. *S is the cannot-claim registry row, recording readings that the packet blocks from export;*
6. *K is the upgrade-registry row, recording the next certificate, bridge, audit, or formalization obligation needed before a blocked or horizon-facing row may move;*
7. *U is the formalization-target registry row, recording the Lean namespace, statement-only site, encoded definition, or explicit absence of a machine target;*
8. *F is the export-control row, recording whether the public claim is admitted by C, T, D, U or blocked by G, S, K;*
9. *H stores componentwise* hsame *transport for the registry rows and the export-control row;*

10. *P is the* Pkg *provenance over* $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{ChoiceRecipeLedger}^{\uparrow}$, $\mathsf{AxiomRequirementLedger}^{\uparrow}$, $\mathsf{WitnessedDescentLedger}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegistryLayer}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a prose-only claim, host truth oracle, hidden theorem registry, private package override, or unledgered export path.

Theorem 1220.2 (Registry layer NameCert obligations)**.** *For the carrier of Definition 1220.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryLayer}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (C, T, G, D, S, K, U, F, H, P, N)$, registry coverage by C, T, G, D, S, K, U, export-control classification by F, componentwise transport by H, package provenance by P, and local naming by N.*

Lean4 ✓ `BEDC.Derived.RegistryLayerUp.RegistryLayerTasteGate single carrier alignment`

Proof. Project the finite packet of Definition 1220.1. The seven registry rows C, T, G, D, S, K, U list the public places where a package-level claim can be admitted, held as a gap, blocked, assigned an obligation, or connected to a formalization target.

The export-control row F does not add a second registry. It reads the displayed rows and records whether the claim is admitted by one of the concept, theorem, tradition, science, or formal-target rows, or blocked by the gap, cannot-claim, or upgrade-obligation rows.

Finally, H, P, N transport, package, and name the same rows. Since the carrier has no further coordinate, every NameCert obligation is exhausted by the displayed packet. □

Theorem 1220.3 (Registry layer export-control exactness)**.** *Every exported package-level claim read through an accepted* $\mathsf{RegistryLayer}^{\uparrow}$ *packet is either admitted by one displayed registry row among C, T, D, U or blocked by one displayed registry row among G, S, K. No claim can be exported by prose placement, host truth, hidden theorem state, or a private package route outside $R = (C, T, G, D, S, K, U, F, H, P, N)$.*

Lean4 ✓ `BEDC.Derived.RegistryLayerUp.RegistryLayer export control exactness`

Proof. Use the obligations of Definition 1220.2. A public export is read through F, and F is part of the same finite carrier as the registry rows.

If F records admission, the admission must point to a displayed concept, theorem, tradition-and-science, or formalization-target row. If F records blocking, the block must point to a displayed gap, cannot-claim, or upgrade-obligation row.

The excluded routes are absent from the carrier. A prose sentence, host truth oracle, hidden theorem table, or private package route supplies no coordinate that H, P, N can transport, package, or name. Thus export control is exact for the displayed registry layer. □

Theorem 1220.4 (Registry layer claim nonescape)**.** *An accepted* $\mathsf{RegistryLayer}^{\uparrow}$ *packet permits no prose-only package claim. If a consumer cannot exhibit an admission row in C, T, D, U and cannot exhibit a blocking row in G, S, K, then the consumer has no* $\mathsf{RegistryLayer}^{\uparrow}$ *route for the claim; the claim remains outside the package export surface.*

Proof. Begin with Definition 1220.3. Every exported claim has a displayed admission row or a displayed blocking row.

Suppose a consumer exhibits neither. Then the classifier row F has no registry row from which to read admission and no registry row from which to read blockage. The transport row H

cannot create the missing row, and the package row P can only record provenance for rows already displayed in the carrier.

It follows that the local naming row N names no exported claim in this case. The attempted prose-only reading therefore fails to escape the registry layer. □

Theorem 1220.5 (Registry layer public export route)**.** *Every public export read of an accepted* $\mathsf{RegistryLayer}^{\uparrow}$ *packet is exhausted by the displayed registry route through* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *row. The route may expose concept, theorem, tradition-and-science, formalization-target, gap, cannot-claim, upgrade-obligation, and export-control rows, but it exports no hidden verification claim, private package override, host truth oracle, unledgered registry entry, or external standard-model assertion.*

Proof. Start with the export-control exactness theorem, Definition 1220.3. It says that every exported package-level claim is admitted by a displayed registry row among C, T, D, U or blocked by a displayed registry row among G, S, K.

Read the public route through the structural rows. The carrier stores those registry rows as finite BHist data, transports them componentwise by H, replays the displayed export decision through Cont, records package provenance by P, and names the same packet locally by N.

Apply claim nonescape, Definition 1220.4. Since a prose-only claim without admission or blocking rows has no route through the packet, the public export surface cannot add hidden verification, private package override, host truth, unledgered registry entry, or an external standard-model assertion. □

The obligation boundary is anchored by the TasteGate carrier in `lean4/BEDC/Derived/RegistryLayerUp/Tast` preserving $C, T, G, D, S, K, U, F, H, P, N$ as separate displayed rows for BEDC.Derived.RegistryLayerUp.taste_gate.

Theorem 1220.6 (Registry layer bridge surface)**.** *Every accepted* $\mathsf{RegistryLayer}^{\uparrow}$ *public export route bridges exactly the displayed concept, theorem, gap, tradition-and-science, cannot-claim, upgrade, formalization-target, export-control, transport, provenance, and local* NameCert *rows to downstream registry consumers. The bridge exports no hidden verification claim, private package override, host truth oracle, unledgered registry entry, or undisplayed standard-model assertion.*

Proof. Start with the public export route of Definition 1220.5. It exhausts every downstream read through the displayed registry rows, BHist, componentwise hsame, displayed Cont, Pkg, and the local NameCert row.

Use export-control exactness from Definition 1220.3. The row F can admit a claim only through the concept, theorem, tradition-and-science, or formalization-target rows, and can block a claim only through the gap, cannot-claim, or upgrade rows.

Apply claim nonescape from Definition 1220.4. A consumer with no displayed admission row and no displayed blocking row has no route through the packet, so there is no private package override or hidden verification surface.

The TasteGate marker BEDC.Derived.RegistryLayerUp.taste_gate remains visible at the boundary. Since the carrier has no further coordinate, the bridge surface is exactly the displayed public registry route. □

Falsifiable prediction: If the cannot-claim row is removed, the packet cannot distinguish blocked export from unregistered absence; if the theorem-registry row is removed, theorem export cannot be audited.

Independence witness: selectorledger, focused-multi-hist-ledger, choicerecipeledger, axiomrequirementledger, witnesseddescentledger: none is bijective because RegistryLayer carries simultaneous admit/block/export rows, while those siblings carry selector, focused-record, recipe, axiom-requirement, or descent rows only.

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryLayer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegistryLayer}^{\uparrow}$ packet is generated from concept, theorem, gap, tradition-and-science, cannot-claim, upgrade, formalization-target, export-control, componentwise hsame transport, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegistryLayer}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1220.1, Definition 1220.2, Definition 1220.3, Definition 1220.4, Definition 1220.5, and Definition 1220.6 bind the public bridge route through the carrier, NameCert obligations, export-control exactness, claim nonescape, TasteGate marker

Lean4 ✓ `BEDC.Derived.RegistryLayerUp.taste gate`

, BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert rows for $C, T, G, D, S, K, U, F, H, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegistryLayer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, complete registry governance, or correctness of every downstream registry entry.

Upgrade path: No further theory-axis upgrade remains for the displayed registry layer; verification and bridge checking remain separate formal-axis obligations.

Chapter 1221

A Concrete Naming Certificate for $\mathsf{MachineInterfaceBoundary}^{\uparrow}$

The $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ packet realizes the registry and machine-facing interface of section 9462.35 as one finite NameCert boundary. It sits beside the registry export surface of chapter 1216, the formalization socket of chapter 1059, the kernel audit witness of chapter 546, and the apophatic naming boundary of chapter 543. Those packets separately expose export control, target routing, audit readback, and refusal discipline; this carrier binds them before a machine-facing route can read a registry claim.

Definition 1221.1 (Machine-interface boundary carrier)**.** *A* $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, E, F, A, S, H, C, P, N)$$

with the following visible rows:

1. *R is the registry-row ledger, recording the claim entry, its registry position, and its cannot-claim boundary when present;*
2. *E is the export-decision row, recording the allowed machine-facing export route for a registry row;*
3. *F is the refused-claim row, recording registry entries that may be cited only as refusals or blocked claims;*
4. *A is the formalization-target row, recording the Lean-side or audit-side target to which an accepted export may be routed;*
5. *S is the socket row, recording the machine-interface socket through which the target is exposed;*
6. *H stores componentwise* hsame *transport for registry, export, refusal, formalization-target, socket, continuation, provenance, and local naming rows;*
7. *C stores displayed* Cont *replay for registry lookup, export inspection, refusal readback, formalization-target routing, socket exposure, and audit replay;*
8. *P is the* Pkg *provenance over* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{KernelAuditWitness}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MachineInterfaceBoundary}^{\uparrow}}$ *row binding the machine-facing boundary to the displayed packet.*

The carrier contains no coordinate for a host-side certificate, unchecked machine claim, hidden registry override, or export route that bypasses a refused registry row.

Lean4 (definition) `BEDC.Derived.MachineInterfaceBoundaryUp.MachineInterfaceBoundaryUp`

Theorem 1221.2 (Machine-interface boundary obligation surface)**.** *For the carrier of Definition 1221.1, the local* $\mathsf{NameCert}_{\mathsf{MachineInterfaceBoundary}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for* $M = (R, E, F, A, S, H, C, P, N)$*;*
2. *registry-export coverage through* R *and* E*;*
3. *refusal exactness through* F*, requiring every blocked or refused registry row to remain visible as refused content;*
4. *formalization-target and socket routing through* A *and* S*;*
5. *audit readback, transport, replay, provenance, and local naming through* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.MachineInterfaceBoundaryUp.MachineInterfaceBoundary obligation surface`

Proof. Project the displayed packet of Definition 1221.1. The rows R, E, F, A, S already separate registry entry, export decision, refused claim, formalization target, and socket exposure.

Use the registry-export surface of chapter 1216 for the R, E, F side of the packet. It keeps exported rows and refused rows distinct before any machine target can inspect them.

Use the machine-interface socket and kernel audit witness surfaces, chapter 1059 and chapter 546, for the A, S, H, C, P, N side. The target is routed through a displayed socket and the audit readback is replayed through the same continuation and provenance rows.

Finally apply the apophatic naming boundary of chapter 543. It prevents a refused request from becoming supplied content, so the five obligation rows exhaust the local $\mathsf{NameCert}_{\mathsf{MachineInterfaceBoundary}^{\uparrow}}$ surface. □

Theorem 1221.3 (Machine-interface boundary export control)**.** *Let*

$$M = (R, E, F, A, S, H, C, P, N)$$

be an accepted $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ *carrier. Every machine-facing export route factors through the export decision row* E*, refused-claim row* F*, formalization-target row* A*, socket row* S*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *after reading the registry row* R*. The route cannot export a refused registry row as a checked machine claim, and it cannot replace the local NameCert row by host-side certification.*

Proof. Read the registry entry first through R. The export decision is then the displayed row E, while every refused or blocked registry claim is kept in F.

Route accepted machine exposure through A and S. These rows determine the formalization target and the machine-interface socket before any audit readback is available to a consumer.

Transport and replay use only H and C, with provenance and local naming supplied by P and N. Since no coordinate of M stores host-side certification or an unchecked machine claim, an export route that tries to bypass F, A, S, H, C, P, N exits the carrier instead of exporting a BEDC machine-facing claim. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceBoundaryUp.MachineInterfaceBoundary export control`

Theorem 1221.4 (Machine-interface boundary TasteGate handoff)**.** *Later Lean work for* MachineInterfaceBoundary *must provide* Nontrivial MachineInterfaceBoundary$^{\uparrow}$ *and* FieldFaithful MachineInterfaceBoundary$^{\uparrow}$ *instances in* lean4/BEDC/Derived/MachineInterfaceBoundaryUp/TasteGate.lean*. The nontriviality witness must expose at least one accepted registry-export or refusal-routing packet, and the field-faithfulness witness must preserve the distinct rows* $R, E, F, A, S, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.MachineInterfaceBoundaryUp.MachineInterfaceBoundary tastegate handoff`

Proof. The carrier is not a terminal tag: it distinguishes registry rows, export decisions, refused claims, formalization targets, socket exposure, transport, replay, provenance, and local naming.

Nontriviality therefore has to exhibit an accepted packet whose machine-facing boundary is readable through those rows. Field faithfulness has to prevent E from replacing F, A from replacing S, or N from being replaced by host-side certification.

Thus the named Lean file is the correct follow-up site for the same displayed carrier, with the TasteGate obligations aligned to the rows bound by Definition 1221.2. □

Falsifiable prediction: Any accepted MachineInterfaceBoundaryUp route that exports a refused registry row as a checked machine claim refutes the carrier.

Independence witness: registryexportsurface, machineinterfacesocket, kernelauditwitness, apophaticname: none is bijective because MachineInterfaceBoundaryUp simultaneously carries registry export rows, refused rows, formalization-target routing, socket exposure, audit readback, transport, continuation, provenance, and local NameCert.

Theorem 1221.5 (Machine-interface boundary paper bridge)**.** *Every accepted* MachineInterfaceBoundary$^{\uparrow}$ *packet realizes the paper-side bridge from machine-interface registry rows to the abstract machine-boundary schema by the displayed rows*

$$R, E, F, A, S, H, C, P, N.$$

The registry row R*, export row* E*, refused-row ledger* F*, formalization-target row* A*, and socket row* S *supply the schema fields for registry admission, export control, refusal retention, target routing, and machine-facing exposure. The audit rows* H, C, P, N *supply transport, replay, provenance, and local naming. Thus a registry/export/socket/audit read reaches the abstract boundary only through the same finite packet, and no route bypasses refused rows, host-side certification, unchecked machine claims, or hidden registry overrides.*

Proof. Read the carrier of Definition 1221.1. The rows R, E, F, A, S already separate the public registry entry, export decision, refused content, formalization target, and socket exposure.

The structural rows H, C, P, N make that separation stable under transport, replay, provenance, and local naming. Combining these rows gives exactly the abstract machine-boundary schema: registry admission is supplied by R, export control by E, F, target and socket routing by A, S, and audit accountability by H, C, P, N.

Because the carrier has no coordinate for host-side certification, unchecked claims, hidden overrides, or routes that bypass F, every bridge read remains inside the displayed packet. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceBoundaryUp.MachineInterfaceBoundary paper bridge certificate`

Closure status for NameCert$_{\text{MachineInterfaceBoundary}^{\uparrow}}$**.**

Constructive story: A MachineInterfaceBoundary$^{\uparrow}$ packet is generated from registry-row, export-decision, refused-claim, formalization-target, socket, componentwise transport, continuation, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{MachineInterfaceBoundary}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1221.1, Definition 1221.2, Definition 1221.3, Definition 1221.4, and Definition 1221.5 bind the public export-control, refused-row, formalization-target, socket, audit, nontriviality, field-faithfulness, and paper-side bridge rows over the BHist packet $R, E, F, A, S, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\mathsf{MachineInterfaceBoundary}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close host-side certification, unchecked machine claims, hidden registry overrides, Lean verification, checked standard-bridge evidence, or export routes that bypass refused registry rows.

Upgrade path: The paper-side mature treatment is complete; remaining work is Lean verification and checked standard-bridge evidence for the same public export-control, refusal, socket, audit, nontriviality, field-faithfulness, and paper-side bridge rows.

Chapter 1222

A Concrete Naming Certificate for $\mathsf{WritingItemAuditPacket}^{\uparrow}$

The $\mathsf{WritingItemAuditPacket}^{\uparrow}$ carrier realizes the writing-item discipline of subsection 9462.30.1 inside the auditable-generativity programme of chapter 9462. It sits beside the closure-status discipline of subsection 9462.30.3, the registry layer of chapter 1220, the object-knowledge certificate of chapter 1218, the registry-consistency audit of chapter 1223, and the machine-readable audit interface of chapter 1232. Those siblings bind chapter scope, registry admission, object-witness evidence, consistency checking, and machine-facing report reads; this packet names the finite writing item before a consumer may read prose as an auditable BEDC claim.

Definition 1222.1 (Writing-item audit packet carrier)**.** *A* $\mathsf{WritingItemAuditPacket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, C, R, L, T, F, G, Q, H, A, P, N)$$

with the following displayed rows:

1. *K is the kind row, recording whether the item is a definition, theorem, corollary, proof, reading, schema, roadmap, or refusal;*
2. *C is the carrier row, recording the BEDC object, packet, registry entry, or refusal surface on which the item lives;*
3. *R is the classifier row, recording the displayed identity, equivalence, or classification relation by which the item is compared;*
4. *L is the ledger row, recording hidden material, package provenance, consumed certificates, and refusal handoffs visible to the item;*
5. *T is the theory-closure row, recording the paper-side closure grade and exact scope licensed by the item;*
6. *F is the formal-status row, recording the Lean-side or audit-side status for the relevant declaration, target, encoded carrier, or explicit absence;*
7. *G is the gap-report row, recording open gaps, refused gaps, closed gaps, and cannot-claim boundaries attached to the item;*
8. *Q is the permitted-claim row, recording the strongest sentence the item authorizes a reader to believe;*

9. *H records componentwise* hsame *transport for kind, carrier, classifier, ledger, theory closure, formal status, gap report, and permitted claim text;*
10. *A records displayed* Cont *replay from item admission through classifier readback, ledger inspection, status readback, gap exposure, and claim publication;*
11. *P is the* Pkg *provenance over* $\mathsf{WritingItemAuditPacket}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$, $\mathsf{RegistryConsistencyAudit}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a prose override, unledgered carrier, hidden classifier, status-free claim, gap-erasing read, or claim text stronger than Q.

Theorem 1222.2 (Writing-item audit packet NameCert obligations)**.** *For the carrier of Definition 1222.1, the local* $\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$ *surface consists exactly of eight obligation groups:*

1. *carrier admission for $W = (K, C, R, L, T, F, G, Q, H, A, P, N)$;*
2. *kind and carrier readback through K and C;*
3. *classifier exactness through R;*
4. *ledger coverage through L;*
5. *theory-closure and formal-status readback through T and F;*
6. *gap-report exposure through G;*
7. *permitted-claim readback through Q;*
8. *componentwise transport, replay, package provenance, and local naming through H, A, P, N.*

Thus a legal writing item is a finite NameCert surface over kind, carrier, classifier, ledger, closure status, formal status, gap report, and permitted claim text, not a bare prose assertion.

Lean4 ✓ `BEDC.Derived.WritingItemAuditPacketUp.WritingItemAuditPacketTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.WritingItemAuditPacketUp.WritingItemAuditPacketNameCert obligations`

Proof. Project the displayed packet $W = (K, C, R, L, T, F, G, Q, H, A, P, N)$. The first seven visible rows supply the writing discipline named in Definition 9462.1984: kind, carrier, classifier, ledger, theory-closure status, formal-verification status, gap report, and claim text.

Read the carrier-facing obligations next. The rows K, C, R, L decide what kind of item is being read, where it lives, how it is compared, and which hidden material or refusal handoffs have been ledgered.

The status and boundary obligations are separate. The rows T, F, G, Q expose the paper closure grade, formal status, gap report, and permitted claim text before a consumer may use the item. The transport, replay, provenance, and local naming rows H, A, P, N preserve exactly that finite inventory, so the eight groups exhaust the local $\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$ surface. □

Theorem 1222.3 (Writing-item audit packet closure bound)**.** *No accepted* $\mathsf{WritingItemAuditPacket}^{\uparrow}$ *packet authorizes a consumer claim stronger than the conjunction of its displayed theory-closure row T, formal-status row F, gap-report row G, and permitted-claim row Q.*

Proof. Begin with an accepted packet $W = (K, C, R, L, T, F, G, Q, H, A, P, N)$. The rows T and F report the two independent axes described by Definition 9462.1992, while G records the visible gap or refusal boundary.

Inspect Q. By Definition 1222.1, Q is the strongest sentence the item authorizes a reader to believe, and the carrier has no coordinate for a stronger prose override.

Finally use H, A, P, N. These rows transport, replay, package, and name the same displayed packet; they do not introduce an additional status grade or a second permitted-claim coordinate. Therefore any stronger consumer claim exits the accepted carrier. □

Theorem 1222.4 (Writing-item audit packet gap exposure)**.** *If the gap-report row G of an accepted* $\mathsf{WritingItemAuditPacket}^{\uparrow}$ *packet records an open gap, refused gap, or cannot-claim boundary, every permitted consumer read of the packet exposes that row before publishing the permitted claim text Q.*

Proof. Use Definition 1222.2. Gap-report exposure is one of the eight local NameCert obligation groups, so a consumer read cannot reach the local name while ignoring G.

The replay row A orders the public read through item admission, classifier readback, ledger inspection, status readback, gap exposure, and claim publication. Thus Q is not a free-standing prose sentence; it is read after the displayed gap report.

If a consumer omits the open-gap, refused-gap, or cannot-claim content of G, the read no longer follows the displayed replay row and no longer preserves the local NameCert surface. Such a read is not a permitted consumer read of the packet. □

Theorem 1222.5 (Writing-item audit packet scoped claim route)**.** *Every permitted consumer read of an accepted* $\mathsf{WritingItemAuditPacket}^{\uparrow}$ *packet $W = (K, C, R, L, T, F, G, Q, H, A, P, N)$ factors through the displayed theory-closure row T, formal-status row F, gap-report row G, permitted-claim row Q, and componentwise transport, replay, provenance, and local naming rows H, A, P, N before publishing the claim. The route keeps the paper-side closure axis and the machine-facing verification axis independent: it does not read T as Lean verification, does not read F as paper closure, and does not erase the exposed gap row G when presenting Q.*

Proof. Start with the NameCert obligations of Definition 1222.2. They require theory-closure readback through T, formal-status readback through F, gap exposure through G, permitted-claim readback through Q, and structural preservation through H, A, P, N.

Apply the closure-bound theorem Definition 1222.3. It prevents a consumer from strengthening the claim beyond the conjunction of the displayed closure, formal-status, gap, and permitted-claim rows.

Apply the gap-exposure theorem Definition 1222.4. A consumer cannot reach the permitted claim text while suppressing an open gap, refused gap, or cannot-claim boundary carried by G.

Finally use the independence of the two status axes from Definition 9462.1992. The row T reports paper closure scope and the row F reports machine-facing status; neither row can discharge the other. The structural rows H, A, P, N replay exactly that separated packet, so the published claim remains scoped by both axes and by the exposed gap row. □

Theorem 1222.6 (Writing-item audit packet mature consumer)**.** *Every downstream consumer of a bridged* $\mathsf{WritingItemAuditPacket}^{\uparrow}$ *packet must preserve the permitted-claim row Q, closure-bound row T, formal-status row F, exposed-gap row G, componentwise transport row H, replay row A, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$ *row N before exporting the audit packet as a consumer-facing writing item.*

Proof. Start with the scoped-claim route of Definition 1222.5. It already forces every permitted consumer read through T, F, G, Q, H, A, P, N before the claim is published.

Now impose the downstream export boundary. A consumer that exports the packet as a writing item has to present the same permitted claim text, the same closure and formal-status rows, and the same exposed gap row, because Definition 1222.3 forbids any stronger claim and Definition 1222.4 forbids hiding G.

The structural rows are equally load-bearing. The transport row H, replay row A, provenance row P, and local naming row N are the only route by which the packet remains the displayed $\mathsf{WritingItemAuditPacket}^{\uparrow}$ carrier. Dropping any of them exports a different object rather than the bridged audit packet. □

Falsifiable prediction: A consumer whose claim text exceeds its displayed closure-status row refutes the $\mathsf{WritingItemAuditPacket}^{\uparrow}$ packet.

Independence witness: registrylayer, objectknowledgecertificate, registryconsistencyaudit, machinereadableauditinterface: none is bijective to $\mathsf{WritingItemAuditPacket}^{\uparrow}$ because the registry layer lacks per-item permitted claim text, the object-knowledge certificate lacks the writing-kind and formal-status rows, the registry-consistency audit lacks the item-kind and permitted-claim rows, and the machine-readable audit interface lacks the carrier-classifier-ledger writing discipline as one finite NameCert surface.

The Lean follow-up must provide explicit Nontrivial $\mathsf{WritingItemAuditPacket}^{\uparrow}$ and FieldFaithful WritingItemAuditP instances in `lean4/BEDC/Derived/WritingItemAuditPacketUp/TasteGate.lean`. The field-faithfulness witness must preserve $K, C, R, L, T, F, G, Q, H, A, P, N$ as distinct rows before BEDC.Derived.WritingItemAuditPacke can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$.

Constructive story: The writing-item audit packet is generated from $BHist$ rows that name a writing kind, permitted claim text, closure-status row, formal-status row, gap row, and the classifier-ledger discipline that binds them. The carrier first reads the item from the ledger, then attaches the audit packet, and finally permits Cont extensions only when the same closure, formal-status, and gap rows remain visible. The $hsame$ relation makes the packet stable under changes of representative history. The NameCert obligations, closure-bound theorem, gap-exposure theorem, prediction, and independence witness define a local audit mechanism: a writing item is admissible only when its permitted text and exposed bounds are carried by the packet.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$, maturePackageClosed)

Scope closed: The scope is exactly the BHist rows $K, C, R, L, T, F, G, Q, H, A, P, N$, hsame transport, Cont replay, Pkg provenance, local NameCert naming, and the carrier, obligation, closure-bound, gap-exposure, scoped-claim route, and mature consumer surfaces of Definition 1222.1, Definition 1222.2, Definition 1222.3, Definition 1222.4, Definition 1222.5, and Definition 1222.6, with the mature consumer route serving as the downstream audit-packet handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WritingItemAuditPacket}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim full document closure, correctness of unaudited writing items, or hidden gap discharge outside the packet rows.

Upgrade path: Verification upgrade requires Lean encoding of the mature consumer route while preserving the permitted claim, closure-bound, formal-status, exposed-gap, transport, replay, provenance, and local NameCert rows.

Chapter 1223

A Concrete Naming Certificate for RegistryConsistencyAudit$^{\uparrow}$

The RegistryConsistencyAudit$^{\uparrow}$ packet realizes the registry-consistency requirement of subsection 9462.35.4 as a finite audit carrier for export-facing registry rows. It sits beside the export packet of chapter 1216, the machine boundary of chapter 1221, the claim-entry layer of chapter 1209, the cannot-claim registry of chapter 554, and the proof-trace audit packet of chapter 1219. Those siblings expose claim rows, machine-facing routes, registry admission, refusal boundaries, and proof-readback evidence; this carrier checks that all such exported readings agree before a package-level registry audit is accepted.

Definition 1223.1 (Registry-consistency audit carrier)**.** *A* RegistryConsistencyAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, S, D, G, X, F, T, L, Q, P, N)$$

with the following displayed rows:

1. *R is the registry-row ledger, naming the primary entry and every related entry whose export reading is being audited;*
2. *S is the status-agreement row, comparing concept status, theorem status, science-roadmap status, closure status, and formal status for the same claim;*
3. *D is the dependency-strength row, recording whether each dependency is definitional, theorem-level, statement-only, encoded, or external to the carrier;*
4. *G is the gap-status row, recording open gaps, refused gaps, and closed gaps visible to the audit;*
5. *X is the cannot-claim row, importing the relevant refusal boundary from* AxisZeckendorfCannotClaim$^{\uparrow}$ *or an adjacent cannot-claim packet;*
6. *F is the closure/formal-status row, comparing the manuscript closure block with the machine-readable formal-status target for the same entry;*
7. *T is the transport row, storing componentwise* hsame *stability for registry, status, dependency, gap, refusal, closure/formal-status, replay, provenance, and local naming rows;*
8. *L is the displayed* Cont *replay row through which an audit consumer reads R, S, D, G, X, F, T;*
9. *Q is the proof-trace row, recording the proof-readback or explicit absence read from* ProofTraceAuditWitness$^{\uparrow}$ *when the audited entry cites proof evidence;*

10. *P is the* Pkg *provenance over* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{ProofTraceAuditWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a hidden registry override, prose-only status repair, machine target outside F, proof read outside Q, or cannot-claim bypass outside X.

Lean4 (definition) `BEDC.Derived.RegistryConsistencyAuditUp.RegistryConsistencyAuditUp`
Lean4 ✓ `BEDC.Derived.RegistryConsistencyAuditUp.RegistryConsistencyAuditTasteGate single carrier alignment`

Theorem 1223.2 (Registry-consistency audit route)**.** *Every accepted* $\mathsf{RegistryConsistencyAudit}^{\uparrow}$ *packet routes an exported claim through the same registry entry, status agreement, dependency-strength, gap-status, cannot-claim, closure/formal-status, proof-trace, transport, replay, provenance, and local naming rows displayed in Definition 1223.1. If any related registry entry disagrees with the primary entry on status, dependency strength, gap status, cannot-claim boundary, closure status, or formal target, the packet is not accepted as a consistency audit for that exported claim.*

Proof. Project the displayed packet $A = (R, S, D, G, X, F, T, L, Q, P, N)$. The primary registry location and all related entries are read first through R, so an audit consumer cannot substitute a private registry row.

Read the agreement rows next. Status agreement is displayed in S, dependency strength in D, gap status in G, cannot-claim boundaries in X, and closure/formal-status comparison in F. A disagreement in any of these rows leaves the corresponding consistency clause without a displayed witness.

The proof-trace row Q and the routing rows T, L, P, N only transport, replay, package, and name the same finite audit packet. Since no remaining coordinate repairs a failed agreement row, an inconsistent registry reading exits the carrier rather than producing an accepted audit. □

Theorem 1223.3 (Registry-consistency audit NameCert obligations)**.** *For the carrier of Definition 1223.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$ *surface consists exactly of eight obligation groups:*

1. *carrier admission for $A = (R, S, D, G, X, F, T, L, Q, P, N)$;*
2. *registry-row coverage through R;*
3. *status-agreement and dependency-strength exactness through S and D;*
4. *gap-status and cannot-claim boundary agreement through G and X;*
5. *closure/formal-status agreement through F;*
6. *proof-trace readback or explicit proof absence through Q;*
7. *componentwise transport and replay through T and L;*
8. *package provenance and local naming through P and N.*

Thus the certificate audits registry coherence by finite rows and does not turn prose governance, hidden proof objects, or host registry state into additional certificate inputs.

Proof. Carrier admission is tuple projection from Definition 1223.1. The registry surface is exhausted by R, and every agreement clause named by Definition 9462.2261 is assigned to one of S, D, G, X, F.

The proof-trace row is separate. It allows a proof-bearing entry to cite the proof-readback packet while allowing an explicitly absent proof read to remain absent; neither case changes the registry, status, dependency, gap, refusal, or closure/formal-status rows.

Finally inspect T, L, P, N. These rows supply componentwise stability, replay, provenance, and local naming for the same displayed packet. They do not add another agreement clause, so the eight groups exhaust the local $\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$ obligation surface. □

Falsifiable prediction: If two registry entries for the same exported claim disagree on status, dependency strength, gap status, cannot-claim boundary, closure status, or formal target, no accepted RegistryConsistencyAuditUp packet certifies that export as registry-consistent.

Independence witness: registryexportsurface, machineinterfaceboundary, claimregistrylayer, axiszeckendorfcannotclaim, prooftraceauditwitness: none carries the joint registry-row, status-agreement, dependency-strength, gap-status, cannot-claim, closure/formal-status, proof-trace, transport, replay, provenance, and local NameCert packet of $\mathsf{RegistryConsistencyAudit}^{\uparrow}$.

Theorem 1223.4 (Registry-consistency audit standard bridge)**.** *The exported* $\mathsf{RegistryConsistencyAudit}^{\uparrow}$ *packet bridges to the standard registry-audit reading by preserving exactly the registry row R, status row S, dependency row D, gap row G, cannot-claim row X, closure/formal-status row F, transport row T, replay row L, proof-trace row Q, provenance row P, and local naming row N. The bridge has no coordinate for a hidden registry override, prose-only status repair, machine target outside F, proof read outside Q, or cannot-claim bypass outside X.*

Proof. Project the carrier of Definition 1223.1. The standard registry-audit reading receives only the displayed rows $R, S, D, G, X, F, T, L, Q, P, N$.

Apply the route theorem Definition 1223.2. It forces every exported claim to pass through the same registry entry, status agreement, dependency strength, gap status, cannot-claim, closure/formal-status, proof-trace, transport, replay, provenance, and local naming rows before the audit is accepted.

Use the NameCert obligations of Definition 1223.3. Those obligations exhaust the local audit surface, so the standard reading cannot import a hidden registry override, prose-only repair, machine target outside F, proof read outside Q, or cannot-claim bypass outside X. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegistryConsistencyAudit}^{\uparrow}$ packet is generated from displayed registry, status, dependency, gap, cannot-claim, closure/formal-status, proof-trace, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1223.1, Definition 1223.2, Definition 1223.3, Definition 1223.4, and the checked TasteGate marker at the carrier site bind the mature paper-axis registry-consistency audit packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegistryConsistencyAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close bridge checking, Lean verification, or any registry repair route for $\mathsf{RegistryConsistencyAudit}^{\uparrow}$.

Upgrade path: No further paper closure-axis upgrade remains; Lean verification and bridge checking remain separate obligations.

Chapter 1224

A Concrete Naming Certificate for $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$

The $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ packet realizes the registry-consistency side of section 9462.35 as a finite NameCert gate for machine-facing registry export. It sits beside the claim-entry layer of chapter 1209, the export surface of chapter 1216, and the formal target socket of chapter 1059. Those siblings admit claims, expose export rows, and route machine targets; this packet binds status-consistency, audit-route, and no-smuggling checks before any accepted registry packet exports a claim.

Definition 1224.1 (Registry export consistency gate carrier)**.** *A* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, B, S, F, A, U, H, C, P, N)$$

with the following visible rows:

1. *R is the registry-entry row, recording the primary registry location and the claim surface being exported;*
2. *B is the blocking row, recording the cannot-claim, refusal, or non-export row that must stop the export when the claim is barred;*
3. *S is the status-consistency row, recording agreement between registry entry, export grade, closure status, and formalization target;*
4. *F is the formal-target row, recording the Lean target, statement-only target, encoded definition, or explicit absence of a machine target for the same claim;*
5. *A is the audit-route row, recording the finite route by which the machine-facing review reads R, B, S, F;*
6. *U is the no-smuggling row, recording that hidden prose synthesis, host semantics, model output, and private registry state are not export sources;*
7. *H stores componentwise* hsame *transport for registry entry, blocking, status consistency, formal target, audit route, no-smuggling, replay, provenance, and local naming rows;*
8. *C stores displayed* Cont *replay for registry lookup, block inspection, status comparison, formal-target readback, audit traversal, and no-smuggling refusal;*

9. *P is the* Pkg *provenance over* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{GroundCompilerRecognition}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an export grade that bypasses S, no private override of B, no machine target outside F, and no audit path outside A.

Lean4 (definition) `BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateCarrier`

Definition 1224.2 (Registry export consistency gate classifier)**.** *The* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *classifier accepts exactly those packets of Definition 1224.1 in which the status-consistency row S compares the same claim across R, the export grade, the closure-status reading, and the formal-target row F, while the audit row A and no-smuggling row U remain visible to the same local* NameCert *row N.*

Lean4 (definition) `BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateClassifierSpe`
Lean4 ✓ `BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateClassifier`
`semantic name certificate`

Theorem 1224.3 (Registry export consistency gate NameCert obligations)**.** *For the carrier of Definition 1224.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission for $G = (R, B, S, F, A, U, H, C, P, N)$;*
2. *registry-entry and blocking coverage through R and B;*
3. *status-consistency exactness through S for registry entry, export grade, closure status, and formal target;*
4. *formal-target readback through F;*
5. *audit-route and no-smuggling accountability through A and U;*
6. *componentwise transport, continuation replay, provenance, and local naming through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RegistryExportConsistencyGateUp.taste gate`

Proof. Project the finite packet of Definition 1224.1. Rows R and B expose the registry entry and the blocking row before any export decision is read.

Next read S together with F. The classifier of Definition 1224.2 requires S to compare the registry entry, export grade, closure-status reading, and formal-target row for the same claim, so the status check cannot be supplied by a separate packet.

The audit and no-smuggling rows are then consumed by A and U. They record the permitted machine-facing route and the readings that remain forbidden at the export boundary.

Finally H, C, P, N transport, replay, package, and name these same rows. Since the carrier has no additional coordinate, the six obligation rows exhaust the local $\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}$ surface. □

Theorem 1224.4 (Registry export consistency gate blocks status conflicts)**.** *If status consistency fails between a registry entry, its export grade, its closure-status reading, and its formalization target, then no accepted* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *packet exports the claim.*

Lean4 ✓ `BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateNoConflictExport`

Proof. Assume an accepted packet

$$G = (R, B, S, F, A, U, H, C, P, N).$$

By Definition 1224.2, acceptance requires S to compare the registry entry in R, the export grade, the closure-status reading, and the formal-target row F for the same claim.

If those readings disagree, the required status-consistency row is absent. The blocking row B cannot be erased, and the audit row A has no accepted route from the inconsistent status data to an exported claim.

The no-smuggling row U rules out recovering the export from prose, host semantics, model output, or private registry state. Therefore the packet is not accepted as an exporting gate for the claim. □

Theorem 1224.5 (Registry export consistency gate sibling dependency)**.** *The* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, *or* $\mathsf{GroundCompilerRecognition}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateSiblingDependency`

Proof. Compare the displayed rows. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ supplies claim-entry export control, while $\mathsf{RegistryExportSurface}^{\uparrow}$ binds registration, blocking, status, machine target, audit, and non-smuggling rows for export.

$\mathsf{MachineInterfaceSocket}^{\uparrow}$ routes formalization targets, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ supplies cannot-claim refusal rows, and $\mathsf{GroundCompilerRecognition}^{\uparrow}$ supplies generated-recognition and accepted-export rows.

None of these siblings carries the joint status-consistency row S together with the audit-route row A, no-smuggling row U, formal-target row F, and the local $\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}$ row N in one finite packet. A row-preserving bijection would have to preserve that combined consistency gate, so no such bijection exists. □

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{RegistryExportConsistencyGate}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RegistryExp}$ instances in `lean4/BEDC/Derived/RegistryExportConsistencyGateUp/TasteGate.lean`. The field-faithfulness witness must preserve $R, B, S, F, A, U, H, C, P, N$ as distinct rows, and the eventual checked target is BEDC.Derived.RegistryExportConsistencyGateUp.RegistryExportConsistencyGateTasteGate_sing

Falsifiable prediction: If status consistency fails between registry entry, export grade, closure status, and formalization target, no accepted $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ packet exports the claim.

Independence witness: claimregistrylayer, registryexportsurface, machineinterfacesocket, axiszeckendorfcannotclaim, groundcompilerrecognition: none carries the joint status-consistency, audit-route, no-smuggling, formal-target, and local NameCert packet of $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ packet is generated from registry entry, blocking, status-consistency, formal-target, audit-route, no-smuggling, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1224.4 is the mature public export boundary, while `BEDC.Derived.RegistryExportConsistencyGateUp.taste` supplies the TasteGate route from registry-entry, status-consistency, audit-route, and no-smuggling rows to the public registry export schema.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegistryExportConsistencyGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This mature package treatment does not close Lean verification, checked bridge status, or exhaustive registry export consumer coverage.

Upgrade path: Further strengthening requires a checked bridge and downstream registry export consumer coverage for the same registry-entry, status-consistency, audit-route, no-smuggling, and local naming rows.

Chapter 1225

A Concrete Naming Certificate for MachineInterfaceFormalizationTarget$^{\uparrow}$

The MachineInterfaceFormalizationTarget$^{\uparrow}$ packet realizes the machine-facing target interface of subsection 9462.35.5 as a finite registry-facing NameCert record. It sits beside the socket router of chapter 1059, the export ledger of chapter 1216, the claim registry layer of chapter 1209, and the audit gate boundary of chapter 526. Those siblings route, export, index, or refuse machine-facing rows; this packet is the target object that a formalization queue can store before any theorem content is claimed.

Definition 1225.1 (Machine-interface formalization-target carrier)**.** *A* MachineInterfaceFormalizationTarget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (N_s, K, R, S, D, E, A, U, H, C, P, L)$$

with the following displayed rows:

1. *N_s is the target-name row, recording the machine-readable name supplied by the interface of Definition 9462.2264;*
2. *K is the intended namespace row in which the future Lean declaration is expected to live;*
3. *R is the registry row tying the target to the current claim registry surface;*
4. *S is the statement-skeleton row, containing the binder shape and public statement outline without proof content;*
5. *D is the dependency-list row, naming the finite prerequisites that the target may cite;*
6. *E is the expected-status row, distinguishing theorem, definition, structure-field, variant, or statement-only queue status as registry data;*
7. *A is the audit-gate row that must accept the socket kind and dependency status before export as theorem content;*
8. *U is the not-claimed row recording the theorem-content, host-supply, automation, or bridge obligations that this target does not assert;*
9. *H records componentwise* hsame *transport for N_s, K, R, S, D, E, A, U;*
10. *C records displayed* Cont *replay from the target name through namespace, registry, statement skeleton, dependencies, expected status, audit gate, and not-claimed rows;*

11. *P is the* Pkg *provenance over* MachineInterfaceSocket$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, AuditGateBoundary$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *L is the local* NameCert$_{\text{MachineInterfaceFormalizationTarget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a proof object, global namespace correctness, automatic dependency discharge, audit-gate bypass, theorem export after refusal, or host-valued truth of the statement skeleton.

Lean4 ✓ `BEDC.Derived.MachineInterfaceFormalizationTargetUp.MachineInterfaceFormalizationTargetTasteGate`
`single carrier alignment`
Lean4 (definition) `BEDC.Derived.MachineInterfaceFormalizationTargetUp.MachineInterfaceFormalizationTargetU`

Theorem 1225.2 (Machine-interface formalization-target NameCert obligations)**.** *For the carrier of Definition 1225.1, the local* NameCert$_{\text{MachineInterfaceFormalizationTarget}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (N_s, K, R, S, D, E, A, U, H, C, P, L),$$

name exposure through N_s, namespace exposure through K, registry membership through R, statement-skeleton exposure through S, finite dependency readback through D, expected-status readback through E, audit-gate eligibility through A, not-claimed exposure through U, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through L.*

Consequently an accepted formalization target is a finite registry-, status-, dependency-, audit-gate-, and not-claimed packet, not a theorem object.

Proof. Project the displayed carrier. The tuple lists the name, namespace, registry, statement skeleton, dependency list, expected status, audit gate, not-claimed row, transport, continuation, provenance, and local name.

Read the registry-facing rows first. The name N_s and namespace K identify where the machine-facing declaration is meant to appear, while R records that the target is part of the claim registry layer and S records only the statement skeleton.

Next inspect D, E, A, U. The dependency and expected-status rows are data for the queue, the audit-gate row controls export eligibility, and the not-claimed row blocks theorem-content conclusions not present in the carrier.

Finally use H, C, P, L. These rows transport, replay, package, and name the same displayed packet. Since no coordinate contains proof content or an audit bypass, the NameCert obligations are exhausted by the stated finite rows. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceFormalizationTargetUp.MachineInterfaceFormalizationTarget`
`namecert obligations`

Theorem 1225.3 (Machine-interface formalization-target audit-gate locality)**.** *Every accepted* MachineInterfaceFormalizationTarget$^{\uparrow}$ *packet can be exported as theorem-facing registry data only through the displayed audit-gate row A. If A refuses the socket kind or dependency status attached to D and E, then the target remains a queue record and cannot be read as theorem content.*

Proof. Begin with the queue rows. The target name, namespace, registry row, statement skeleton, dependency list, and expected status are all displayed before any export decision is made.

Use the audit-gate row. The row A is the only displayed gate eligible to judge the socket kind and dependency status attached to the packet. Neither the registry row nor the statement skeleton contains an independent export predicate.

Apply the not-claimed row. The row U records the theorem-content and host-supply surfaces that the packet refuses to assert. Componentwise transport and continuation replay preserve that refusal, so a gate refusal leaves the target as registry data rather than theorem content. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceFormalizationTargetUp.MachineInterfaceFormalizationTarget audit gate locality`

Theorem 1225.4 (Machine-interface formalization-target registry-consistency handoff)**.** *Every* $\mathsf{RegistryExportSurface}^{\uparrow}$ *or* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *consumer of an accepted* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ *packet receives the same rows*

$$N_s, K, R, S, D, E, A, U, H, C, P, L.$$

The consumer may index the name, namespace, registry row, skeleton, dependencies, expected status, audit gate, and not-claimed surface, but it may not replace them by an off-packet registry claim, proof object, or successful Lean declaration.

Proof. Project the carrier and expose its registry inventory. The only registry-facing data are N_s, K, R, S, D, E, A, U together with their transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 1225.2. They make registry membership, statement skeleton, dependencies, expected status, audit eligibility, and not-claimed surface explicit fields of the same packet.

The export and claim-registry consumers therefore receive a finite handoff rather than a hidden declaration. If a consumer replaces these rows by theorem content or a successful Lean declaration, it has left the displayed carrier and no longer preserves the NameCert surface. □

Lean4 ✓ `BEDC.Derived.MachineInterfaceFormalizationTargetUp.MachineInterfaceFormalizationTarget registry consistency handoff`

Theorem 1225.5 (Machine-interface formalization-target nonescape)**.** *No accepted* MachineInterfaceFormalizationT *packet exports theorem content, marker repair, host supply, automatic dependency discharge, or bridge completion from its statement skeleton, dependency list, expected status, or audit-gate row. Such content can appear only in a later packet with its own displayed carrier and NameCert obligations.*

Proof. Read the statement-skeleton row and dependency row. They specify the shape and prerequisites of a target, but neither contains a proof object, a repaired marker, host supply, or a bridge.

Inspect the expected-status and audit-gate rows. They classify how the target is queued and whether the gate accepts the displayed status; they do not manufacture the content whose admission they classify.

Finally apply the not-claimed row together with the transport and replay rows. The not-claimed surface explicitly refuses those escape routes, while H, C, P, L only preserve and name the same packet. Therefore theorem content or bridge completion must be supplied by another displayed certificate. □

Falsifiable prediction: A MachineInterfaceFormalizationTargetUp packet cannot be exported as theorem content when its audit-gate row refuses its socket kind or dependency status.

Independence witness: machineinterfacesocket, registryexportsurface, claimregistrylayer, auditgateboundary, socketkindclassifier: the socket router lacks the full target queue record, the

export surface lacks the not-claimed target packet, the claim registry layer lacks audit-gate locality for socket kind and dependency status, the audit gate lacks registry and statement-skeleton fields, and the socket kind classifier lacks target name, namespace, dependencies, expected status, and not-claimed rows; no bijection preserves the full carrier.

The Lean follow-up must provide Nontrivial MachineInterfaceFormalizationTarget$^{\uparrow}$ and FieldFaithful MachineInterf in `lean4/BEDC/Derived/MachineInterfaceFormalizationTargetUp/TasteGate.lean` before BEDC.Derived.Mac can be cited by a checked paper marker.

Closure status for NameCert$_{\text{MachineInterfaceFormalizationTarget}^{\uparrow}}$.

Constructive story: A MachineInterfaceFormalizationTarget$^{\uparrow}$ packet is generated from target name, socket kind, registry row, statement skeleton, dependency list, expected status, audit-gate row, not-claimed exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MachineInterfaceFormalizationTarget}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1225.1, Definition 1225.2, Definition 1225.3, Definition 1225.4, and Definition 1225.5 bind the carrier, local NameCert, audit-gate locality, registry-consistency handoff, and nonescape boundary for the finite target packet.

Formal verification: FormalStatus(NameCert$_{\text{MachineInterfaceFormalizationTarget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close theorem content, proof object production, automatic dependency discharge, audit-gate bypass, marker repair, host supply, or bridge completion.

Upgrade path: No further paper-side closure-axis upgrade remains for the displayed machine-interface packet.

Chapter 1226

A Concrete Naming Certificate for $\mathsf{FinitePrefixLimitStability}^{\uparrow}$

The $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ packet realizes the finite-prefix stability surface of the Real-completion diagonal-limit route of chapter 9238. It sits after the finite prefix budget of chapter 919, and it consumes the finite stream windows of chapter 126, the regular-rational readbacks of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. The packet names the stability replay by which a displayed prefix observation reaches a Real-facing consumer; it is not a host limit, quotient stream equality, detached selector, countable-choice row, ambient Real-completeness theorem, or bridge schema.

Falsifiable prediction: An accepted finite-prefix limit-stability packet is refuted by any RealUp-facing consumer that reads a terminal seal without first exposing the displayed finite prefix budget, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance row.

Independence witness: finiteprefixlimitbudget, realwindowbudget, realwindowsynchronizer, regularcauchymodulusnormalizer, reallimituniquenessseal: none is bijective because this packet carries the stability replay of a budgeted prefix read through the RealUp seal, whereas those siblings respectively record budget selection, window budgeting, synchronization, modulus normalization, or two-seal uniqueness.

Definition 1226.1 (Finite-prefix limit-stability carrier)**.** *A* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *row naming the admitted finite prefix budget for the diagonal-limit read;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed under that budget;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the rational arithmetic and error budget of the prefix readback;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, W, R, D have been displayed on the same packet;*
6. *H records componentwise* hsame *transport for B, W, R, D, E and the local certificate row;*

7. *C records displayed* Cont *replay from the finite prefix budget through the window, readback, dyadic tolerance, and Real seal;*
8. *P is the* Pkg *provenance over* FinitePrefixLimitStability$^{\uparrow}$, FinitePrefixLimitBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* NameCert$_{\text{FinitePrefixLimitStability}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite-prefix stability route and its transport, replay, provenance, and naming support.

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityUpTasteGate single carrier alignment`

Theorem 1226.2 (Finite-prefix limit-stability obligation surface)**.** *For the carrier of Definition 1226.1, the local* NameCert$_{\text{FinitePrefixLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission, finite prefix-budget exposure through B,* StreamName$^{\uparrow}$ *window exposure through W,* RegSeqRat$^{\uparrow}$ *readback exposure through R,* DyadicRatCore$^{\uparrow}$ *tolerance exposure through D, terminal* Real$^{\uparrow}$ *seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityTasteGate single carrier alignment`

Proof. Project the carrier S. Its coordinates list the finite prefix budget, stream window, regular-rational readback, dyadic tolerance row, terminal Real seal, transport, replay, provenance, and local naming row.

Read the data-bearing rows in order. The budget row B admits the finite prefix read, the window row W displays the observed stream segment, the readback row R records the regular-rational content of that segment, and the dyadic row D records the tolerance attached to the readback.

The Real seal E is downstream of those four rows. The support rows H, C, P, N only transport, replay, source, and name the same displayed route, so the listed rows exhaust the local naming surface. □

Theorem 1226.3 (Finite-prefix limit-stability carrier admission)**.** *Every accepted* FinitePrefixLimitStability$^{\uparrow}$ *packet exposes exactly the carrier rows*

$$B, W, R, D, E, H, C, P, N$$

before any Real$^{\uparrow}$*-facing read. Thus a consumer cannot read the terminal Real seal E without first admitting the finite prefix budget B,* StreamName$^{\uparrow}$ *window W,* RegSeqRat$^{\uparrow}$ *readback R,* DyadicRatCore$^{\uparrow}$ *tolerance row D, componentwise transport H, displayed replay C, provenance P, and the local naming row N.*

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityCarrier admission`

Proof. Project the carrier of Definition 1226.1. The projection exposes precisely the nine displayed coordinates $B, W, R, D, E, H, C, P, N$ and gives no auxiliary Real-facing coordinate.

Apply Definition 1226.2. Its obligation surface identifies the accepted packet with carrier admission, the four finite data rows, the terminal seal, transport, replay, provenance, and local naming. Hence every Real-facing read must pass through those rows before E can be consumed. □

Theorem 1226.4 (Finite-prefix limit-stability transport stability)**.** *Componentwise* hsame *transport and displayed* Cont *replay preserve the route order from the finite prefix budget through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The preserved route is*

$$B \to W \to R \to D \to E$$

with support rows H and C carrying only transport and replay for the displayed coordinates; it does not use quotient stream equality, selected limits, or countable choice.

Proof. Read the transport row H of Definition 1226.1. It is componentwise hsame transport for the data rows B, W, R, D, E together with the local certificate row, so it can preserve the displayed route but cannot reorder it or create a detached terminal seal.

Read the replay row C. It records displayed Cont replay from the finite prefix budget through the window, readback, dyadic tolerance, and Real seal. Since the replay starts at B and ends at E through the named intermediate rows, it preserves the finite route order.

The excluded objects are absent from both rows. Neither H nor C contains quotient stream equality, a selected limit, or a countable-choice selector, so transport stability is exactly the stability of the displayed finite-prefix route. □

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityCarrier route unary closure`

Theorem 1226.5 (Finite-prefix limit-stability ledger exactness)**.** *The local* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ *ledger is exact for the finite-prefix route. It accounts for the budget row B, window row W, readback row R, tolerance row D, seal row E, transport row H, replay row C, provenance row P, and naming row N, and it exports no ambient Real-completeness theorem, quotient-completion principle, selected-limit rule, countable-choice row, or bridge schema.*

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityLedgerExactness`

Proof. Use Definition 1226.2 to read the local ledger. The theorem lists the admitted rows and support rows as the whole $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ surface.

The carrier admission theorem Definition 1226.3 fixes the exposed coordinates before any Real-facing read. The transport-stability theorem Definition 1226.4 then assigns H and C only the transport and replay roles attached to that same route.

It remains only to identify what the ledger does not export. The carrier has no coordinate for ambient Real completeness, quotient-completion data, a selected limit, countable choice, or a bridge schema. Therefore the local ledger is exact for the finite-prefix route and nothing beyond it. □

Theorem 1226.6 (Finite-prefix limit-stability prefix nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *packet factors through the finite route*

$$B, W, R, D, E, H, C, P, N.$$

The consumer receives the displayed finite prefix budget, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{Real}^{\uparrow}$ *seal, and structural support rows. It receives no host real equality, quotient stream equality, selected host limit, detached tail oracle, countable-choice selector, ambient Real-completeness witness, or standard bridge row.*

Proof. Begin with Definition 1226.1. The only Real-facing coordinate is the seal row E, and it is preceded by the budget row B, window row W, readback row R, and tolerance row D.

Apply Definition 1226.2. It forces every accepted consumer to expose the finite prefix budget, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows before the packet is usable.

The excluded objects are not coordinates of the carrier. Since H, C, P, N preserve and name displayed rows without creating new analytic data, no consumer can project host equality, quotient equality, a selected limit, a tail oracle, countable choice, ambient completeness, or a bridge schema from this packet. □

Theorem 1226.7 (Finite-prefix limit-stability Real handoff)**.** *The diagonal-limit route of chapter 9238 may consume an accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *packet as the finite stability handoff from a displayed prefix observation to a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff is compatible with chapter 919, chapter 126, chapter 301, chapter 291, and chapter 38 because it preserves their budget, window, readback, tolerance, and seal rows in one local* $\mathsf{NameCert}$ *packet.*

Proof. Read the budget sibling first. The $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ row supplies the finite prefix budget consumed by B, so the stability packet never begins from an unbounded stream or hidden tail schedule.

Read the observation siblings next. The $\mathsf{StreamName}^{\uparrow}$ row W, $\mathsf{RegSeqRat}^{\uparrow}$ row R, and $\mathsf{DyadicRatCore}^{\uparrow}$ row D keep the finite observation, regular readback, and tolerance ledger visible before any terminal seal is public.

Finally read the $\mathsf{Real}^{\uparrow}$ sibling through E. The transport, continuation, provenance, and naming rows keep that seal tied to the same displayed prefix route, so the Real-completion consumer receives a finite stability handoff rather than a host limit, quotient stream, selected limit, choice row, ambient completeness theorem, or bridge schema. □

Theorem 1226.8 (Finite-prefix limit-stability route determinacy)**.** *Let* $S = (B, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *carrier. If two consumer reads of* S *expose the same finite prefix budget* B*, the same* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the same componentwise transport rows* H*, and the same continuation replay* C*, then both reads reach the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. No alternate seal may be attached to the displayed route by a hidden tail oracle, host limit, quotient stream equality, selected-limit row, countable-choice row, ambient completeness theorem, or bridge schema.*

Proof. Start from the carrier decomposition in Definition 1226.1. The terminal seal coordinate E is part of the same finite packet as B, W, R, D, H, C, P, N, not a separately supplied analytic witness.

Apply the obligation surface of Definition 1226.2. An accepted read has access exactly to the displayed budget, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows. Thus two reads with identical B, W, R, D, H, C are comparing the same displayed route inside the same packet.

The nonescape theorem Definition 1226.6 excludes every non-coordinate source of a second seal. The Real handoff theorem Definition 1226.7 then identifies the consumer-facing output as the terminal seal already carried by E, so the displayed prefix route has one Real-facing seal and admits no detached alternative. □

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityCarrier real seal determinacy`

Theorem 1226.9 (Finite-prefix limit-stability budget-window lock)**.** *Every accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *packet must expose the finite prefix budget* B *before the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and must expose* W *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*. The budget-window lock is the initial segment of the displayed handoff*

$$B \to W \to R \to D \to E$$

and no $\mathsf{Real}^{\uparrow}$*-facing consumer can bypass it through a host limit, detached stream oracle, selected-limit row, quotient stream equality, countable-choice row, ambient completeness theorem, or bridge schema.*

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityCarrier budget window lock`

Proof. Project the carrier of Definition 1226.1. Its data coordinates begin with the budget row B, the stream window row W, and the regular-rational readback row R.

Apply the obligation surface of Definition 1226.2. It makes carrier admission, budget exposure, window exposure, and readback exposure part of the same local $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ surface, so an accepted packet cannot present W without B or present R without W.

Finally use the Real handoff theorem Definition 1226.7. The Real-facing consumer receives the terminal seal only as the end of the same displayed finite route, hence the budget-window lock is forced before any seal-facing consumption. □

Theorem 1226.10 (Finite-prefix limit-stability tolerance-seal lock)**.** *Every accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *packet reads the row* D *as the required* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row before* E *can be read as a terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the seal is locked to the displayed dyadic tolerance ledger and cannot be supplied by host Real equality, quotient stream equality, a hidden tail route, a selected-limit row, countable choice, ambient Real completeness, or a standard bridge row.*

Proof. Read the carrier coordinates. The row D is explicitly the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row recording the rational arithmetic and error budget of the prefix readback, while E is the $\mathsf{Real}^{\uparrow}$ seal reached only after B, W, R, D have been displayed on the same packet.

Use Definition 1226.4. Componentwise hsame transport and displayed Cont replay preserve the route $B \to W \to R \to D \to E$, so they cannot move the seal before the tolerance row or manufacture an untracked analytic seal.

Apply Definition 1226.6 and Definition 1226.8. They exclude every non-coordinate source of a terminal seal and force the displayed route to determine the Real-facing output already carried by E. □

Theorem 1226.11 (Finite-prefix limit-stability NameCert obligation triad)**.** *The* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ *obligation surface is the triad consisting of the budget-window lock of Definition 1226.9, the tolerance-seal lock of Definition 1226.10, and the route determinacy of Definition 1226.8, together with the carrier of Definition 1226.1, the obligation surface of Definition 1226.2, transport stability of Definition 1226.4, ledger exactness of Definition 1226.5, and prefix nonescape of Definition 1226.6. This package closes the chapter's obligation-grade surface without claiming global Real completeness, quotient completion, selected limits, choice rows, or a bridge schema.*

Proof. The carrier and obligation-surface theorem identify the rows that an accepted packet exposes: $B, W, R, D, E, H, C, P, N$ and no additional analytic coordinate.

The budget-window lock fixes the initial route segment from the finite prefix budget through the stream window and regular-rational readback. The tolerance-seal lock fixes the terminal route segment from the dyadic tolerance ledger into the Real seal.

Transport stability, ledger exactness, and prefix nonescape then show that the support rows only preserve, source, name, and replay the same displayed route. Route determinacy supplies the final consumer-facing uniqueness of the seal attached to that route. These clauses are exactly the obligation surface recorded by the local NameCert packet. □

Theorem 1226.12 (Finite-prefix limit-stability source lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read of an accepted* $\mathsf{FinitePrefixLimitStability}^{\uparrow}$ *carrier must expose the ordered route*

$$B \to W \to R \to D \to E$$

before the support rows H, C, P, N *can be consumed. The support rows may transport, replay, source, and name the displayed route, but they cannot supply a Real seal, prefix budget, stream window, regular-rational readback, or dyadic tolerance row out of order.*

Proof. Project the carrier with Definition 1226.1. The data rows appear as the budget B, window W, readback R, tolerance D, and terminal seal E, while H, C, P, N are support rows for that displayed data.

Use Definition 1226.2 to read the local naming surface. The accepted packet presents the route by exposing the finite prefix budget, then the stream window, then the regular-rational readback, then the dyadic tolerance row, and only then the Real seal.

Finally apply Definition 1226.6 and Definition 1226.7. They force every Real-facing consumer through the displayed finite route and make the handoff terminate at E, so the support rows can only preserve and name that route after its source order has been exposed. □

Theorem 1226.13 (Finite-prefix limit-stability primitive-scope binding)**.** *For every accepted* FinitePrefixLimitStabi *carrier, the route*

$$B \to W \to R \to D \to E$$

is scoped exactly by the BEDC primitives BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, together with the sibling rows* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. No quotient stream, host limit, selected tail, countable-choice row, ambient completeness theorem, or bridge schema contributes to the scoped route.*

Proof. Project the carrier of Definition 1226.1. Its displayed coordinates are the finite-prefix budget row B, the stream window W, the regular-rational readback R, the dyadic tolerance ledger D, the terminal Real seal E, componentwise transport H, continuation replay C, provenance P, and local naming N.

The obligation surface of Definition 1226.2, carrier admission of Definition 1226.3, transport stability of Definition 1226.4, ledger exactness of Definition 1226.5, and Real handoff of Definition 1226.7 identify these coordinates as the whole route surface. The sibling rows are therefore already displayed: B names the finite-prefix budget, W names the stream observation, R names the regular-rational readback, D names the dyadic tolerance ledger, and E names the Real seal.

Route determinacy, the budget-window lock, the tolerance-seal lock, the NameCert obligation triad, and the source lock then bind the support rows to the same route. The support rows H, C, P, N supply only hsame transport, Cont replay, Pkg provenance, and NameCert naming over the carrier's finite BHist packet; they do not introduce a second route or an analytic substitute.

Finally apply prefix nonescape from Definition 1226.6. It excludes every non-coordinate source named in the statement, so the primitive and sibling rows listed above are the exact scope of the accepted finite-prefix limit-stability route. □

Theorem 1226.14 (Finite-prefix limit-stability public NameCert export)**.** *The carrier of Definition 1226.1, the obligation surface of Definition 1226.2, the source lock of Definition 1226.12, the primitive-scope binding of Definition 1226.13, the Real handoff of Definition 1226.7, and the TasteGate marker export the public* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ *surface over* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *This exported surface excludes host limits, quotient streams, selected limits, countable choice, ambient completeness, and bridge schema data.*

Proof. Start with the NameCert obligation triad of Definition 1226.11. It packages the budget-window lock, tolerance-seal lock, and route determinacy together with the carrier, obligation surface, transport, ledger, and prefix nonescape clauses.

Apply the source lock of Definition 1226.12. It forces every Real-facing or Completion-facing consumer to read the ordered finite route $B \to W \to R \to D \to E$ before consuming the support rows H, C, P, N.

Use the primitive-scope binding of Definition 1226.13. This identifies the exact BEDC primitives and sibling rows that generate the route and excludes the host, quotient, choice, ambient-completeness, and bridge-schema sources named in the statement.

Finally apply the Real handoff of Definition 1226.7. The handoff terminates the exported route at the displayed $\mathsf{Real}^{\uparrow}$ seal, while the TasteGate marker records the accepted public surface. The composed package is therefore the public $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ export over precisely the listed primitives and sibling rows. □

Theorem 1226.15 (Finite-prefix limit-stability standard bridge precondition)**.** *Any standard bridge eligible to consume the public* $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ *export must preserve the ordered finite BEDC route*

$$B \to W \to R \to D \to E$$

through the sibling rows $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Bridge-side data may be exposed only after the finite prefix budget, stream window, regular-rational readback, dyadic tolerance ledger, and terminal Real seal have appeared in that order. No host limit, quotient stream equality, selected limit, countable-choice row, ambient Real completeness theorem, or detached bridge-schema row can serve as an alternate precondition for that bridge.*

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityUp StdBridge`

Proof. Begin with the public export of Definition 1226.14. It supplies the $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$ surface over the five sibling rows and the BEDC primitives that name the finite route.

Apply the primitive-scope binding of Definition 1226.13. The route is scoped exactly by BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ together with the displayed sibling rows, so a standard bridge consuming the public export has no additional source from which to derive bridge-side data.

Use the source lock of Definition 1226.12. Every Real-facing or Completion-facing read must encounter B, then W, then R, then D, and only then E before support rows can be consumed.

Finally apply prefix nonescape from Definition 1226.6. The excluded host, quotient, selected-limit, choice, completeness, and detached-schema sources cannot provide a second bridge entry. Thus the only admissible standard-bridge precondition is preservation of the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePrefixLimitStability}^{\uparrow}}$**.**

Constructive story: The finite-prefix limit-stability certificate is generated from $BHist$ rows that name a finite prefix budget, stream-name observation, regular-rational readback, dyadic tolerance ledger, and the terminal Real seal that keeps

the route stable at that prefix. The carrier first reads the finite prefix, then records the stability witness, and finally allows Cont to continue toward a Real handoff only through that same prefix. The $hsame$ relation prevents representative changes from altering which finite prefix carries the stability evidence. The obligation surface, carrier admission, transport stability, ledger exactness, prefix nonescape, and Real handoff theorem together state that finite ledger data are sufficient for the admitted route, while external completion witnesses are outside scope.

Theory closure: TheoryGate(NameCert$_{\text{FinitePrefixLimitStability}\uparrow}$, maturePackageClosed)

Scope closed: This chapter binds the finite-prefix limit-stability carrier, obligation-surface theorem, carrier-admission theorem, transport-stability theorem, ledger-exactness theorem, prefix-nonescape theorem, Real-handoff theorem, route-determinacy theorem, budget-window lock of Definition 1226.9, tolerance-seal lock of Definition 1226.10, NameCert obligation triad of Definition 1226.11, primitive-scope binding of Definition 1226.13, public NameCert export of Definition 1226.14, and TasteGate instance

Lean4 ✓ `BEDC.Derived.FinitePrefixLimitStabilityUp.taste gate`

.

Formal verification: FormalStatus(NameCert$_{\text{FinitePrefixLimitStability}\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.FinitePrefixLimitStabilityUp.FinitePrefixLimitStabilityTasteGate\_single\_carrier\_alig`

Bridge status: bridgeChecked

Origin: ai

Not claimed: It does not claim global limit existence, completed Real equality, selected limits, choice rows, or stability without the named finite prefix row.

Upgrade path: No further upgrade – chapter at top of the formal bridge axis for the displayed finite-prefix route.

Chapter 1227

A Concrete Naming Certificate for $\mathsf{AuditWritingDiscipline}^{\uparrow}$

The $\mathsf{AuditWritingDiscipline}^{\uparrow}$ packet realizes the audit-methodology discipline of section 9462.30 inside the auditable-generativity programme of chapter 9462. It sits beside the registry export surface of chapter 1216, the claim registry layer of chapter 1209, the machine-interface socket of chapter 1059, the standard bridge audit packet of chapter 1217, and the proof trace audit witness of chapter 1219. Those siblings expose registry export, claim admission, machine routing, bridge audit, and proof-trace readback; this carrier binds them into one writing packet before a public claim may be read as auditable prose.

Definition 1227.1 (Audit writing discipline carrier)**.** *An* $\mathsf{AuditWritingDiscipline}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (W, C, K, L, T, F, G, R, H, P, N)$$

with the following visible rows:

1. *W is the writing-item row, naming the claim, paragraph, theorem, definition, or closure-status site being exported.*
2. *C is the carrier row, recording the BEDC carrier or packet surface to which the writing item is attached.*
3. *K is the classifier row, recording the classifier, membership, or exactness predicate that licenses the writing item.*
4. *L is the ledger row, recording the finite audit ledger entries consumed by the writing item.*
5. *T is the theory-axis row, recording the theory-closure grade as a paper-side coordinate.*
6. *F is the formal-axis row, recording the formal-verification grade as an independent Lean-side coordinate.*
7. *G is the gap/refusal row, recording missing bridge data, refused readings, blocked host resources, and unsupported consumer routes.*
8. *R is the registry row, recording how the claim is admitted, blocked, or handed to the registry export surface.*
9. *H is the transport row, recording machine-interface routing, standard-bridge audit transport, and proof-trace readback through the sibling surfaces.*

10. *P is the* Pkg *provenance over* RegistryExportSurface$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, MachineInterfaceSocket$^{\uparrow}$, StandardBridgeAuditPacket$^{\uparrow}$, ProofTraceAuditWitness$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{AuditWritingDiscipline}^{\uparrow}}$ *row.*

The carrier has no coordinate for a prose-only claim, a collapsed theory/formal status, a hidden registry override, a proof-trace read without refusal data, or a host bridge imported as BEDC evidence.

Lean4 ✓ `BEDC.Derived.AuditWritingDisciplineUp.AuditWritingDisciplineTasteGate single carrier alignment`

Theorem 1227.2 (Audit writing discipline obligation surface)**.** *For the carrier of Definition 1227.1, the local* NameCert$_{\text{AuditWritingDiscipline}^{\uparrow}}$ *surface consists exactly of six obligation rows: writing-item admission through W; carrier and classifier exposure through C, K; ledger exactness through L; independent theory and formal status reporting through T, F; gap and refusal exposure through G; and registry, transport, provenance, and local naming through R, H, P, N.*

Proof. Project the displayed packet of Definition 1227.1. The rows $W, C, K, L, T, F, G, R, H, P, N$ are the only available coordinates, so the writing item must first name its carrier, classifier, and ledger route.

Read the status rows next. The theory-axis row T records paper closure, while the formal-axis row F records Lean verification status. Since they are separate rows, the carrier has no rule that turns one status coordinate into the other.

Finally attach gap/refusal, registry, transport, provenance, and local naming. Rows G, R, H, P, N keep unsupported readings, claim export, machine routing, bridge audit, and proof-trace readback visible in the same packet, giving exactly the six local obligation rows. □

Theorem 1227.3 (Audit writing discipline nonescape)**.** *Let*

$$A = (W, C, K, L, T, F, G, R, H, P, N)$$

be an accepted AuditWritingDiscipline$^{\uparrow}$ *packet. Every public claim read factors through the displayed writing item, carrier, classifier, ledger, theory axis, formal axis, gap/refusal, registry, transport, provenance, and local naming rows. No public consumer may replace those rows by prose emphasis, host equality, unchecked bridge evidence, private registry state, hidden proof trace, or a single status coordinate that merges T and F.*

Proof. Begin with the obligation surface of Definition 1227.2. It forces the public writing item to expose its carrier, classifier, ledger, status axes, refusal surface, registry route, transport, provenance, and local name together.

If the route omits T or F, the consumer no longer knows which axis is being cited. If the route omits G, unsupported readings and refusal boundaries are erased. If it omits R or H, the claim no longer passes through registry export or machine-facing transport.

Thus every accepted claim remains inside the displayed finite packet. The excluded prose, host, bridge, registry, proof-trace, and collapsed-status readings have no coordinate from which to enter the AuditWritingDiscipline$^{\uparrow}$ carrier. □

Theorem 1227.4 (Audit writing discipline registry handoff)**.** *For every accepted* AuditWritingDiscipline$^{\uparrow}$ *packet $A = (W, C, K, L, T, F, G, R, H, P, N)$, the claim admitted by W is exported only through*

the registry row R and the sibling surfaces $\mathsf{RegistryExportSurface}^{\uparrow}$ *and* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*. The machine-facing route in H may cite* $\mathsf{MachineInterfaceSocket}^{\uparrow}$*,* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$*, and* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *only after the registry row has preserved the distinct carrier, classifier, ledger, theory-axis, formal-axis, and refusal rows.*

Proof. First read registry admission through R. By the obligation surface, R is attached to the same writing item, carrier, classifier, ledger, theory axis, formal axis, refusal surface, provenance, and local name.

Next use the sibling registry surfaces. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ records whether the claim is admitted or refused, while $\mathsf{RegistryExportSurface}^{\uparrow}$ records whether it may leave the registry as a public export.

Only after those rows are displayed may the transport row H route the claim toward $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, or $\mathsf{ProofTraceAuditWitness}^{\uparrow}$. Since H is part of the same packet, it cannot erase C, K, L, T, F, G or replace them by a hidden bridge or proof-trace reading. □

Falsifiable prediction: If a public claim collapses the theory-closure row and formal-verification row into one status coordinate, it is not an accepted AuditWritingDisciplineUp packet.

Independence witness: registryexportsurface, claimregistrylayer, machineinterfacesocket, standardbridgeauditpacket, prooftraceauditwitness: registry export controls public claim release without writing-item carrier discipline, the claim registry layer records admission without proof-trace transport, the machine-interface socket routes targets without closure-axis separation, the standard bridge audit packet controls host-bridge leakage without claim registry admission, and the proof trace audit witness records proof evidence without registry export. None is bijective to $\mathsf{AuditWritingDiscipline}^{\uparrow}$.

The Lean follow-up file `lean4/BEDC/Derived/AuditWritingDisciplineUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ AuditWritingDiscipline}^{\uparrow}$ and $\mathsf{FieldFaithful\ AuditWritingDiscipline}^{\uparrow}$ instances, preserving $W, C, K, L, T, F, G, R, H, P, N$ as separate displayed fields. The expected canonical target is `BEDC.Derived.AuditWritingDisciplineUp.taste_gate`.

Theorem 1227.5 (Audit writing discipline public export)**.** *Every accepted* $\mathsf{AuditWritingDiscipline}^{\uparrow}$ *read exports only the visible writing-item, closure-axis, formal-axis, registry, proof-trace, machine-interface, transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{AuditWritingDiscipline}^{\uparrow}}$ *rows before any audit consumer may cite the packet.*

Proof. Read the carrier of Definition 1227.1. Its displayed rows expose the writing item, theory-axis, formal-axis, registry, transport, provenance, and local name, while the transport row records the machine-interface, standard-bridge audit, and proof-trace routes.

The obligation surface of Definition 1227.2 then makes those rows the public certificate surface. The nonescape theorem of Definition 1227.3 blocks prose emphasis, host equality, hidden bridge evidence, private registry state, and collapsed status coordinates from replacing the displayed rows.

Finally, the registry handoff of Definition 1227.4 requires any cited public claim to pass through the registry row before machine-facing and proof-trace consumers read it. Thus the exported audit surface is exactly the visible rows listed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{AuditWritingDiscipline}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditWritingDiscipline}^{\uparrow}$ packet is generated from writing item, carrier, classifier, ledger, theory-axis, formal-axis, gap/refusal, registry, transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditWritingDiscipline}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1227.1, Definition 1227.2, Definition 1227.3, Definition 1227.4, and Definition 1227.5 give the mature registry, proof-trace, machine-interface, and bridge-audit compatibility surface over $W, C, K, L, T, F, G, R, H, P, N$ while keeping the theory and verification axes independent.

Formal verification: FormalStatus(NameCert$_{\text{AuditWritingDiscipline}\uparrow}$, axiomClean)
Lean target: `BEDC.Derived.AuditWritingDisciplineUp.AuditWritingDisciplineTasteGate\_single\_carrier\_alignment`
Bridge status: none
Origin: ai
Not claimed: This chapter does not close bridge checking, global registry correctness, proof-trace soundness, or machine-interface completeness.
Upgrade path: No further paper-side closure grade is claimed beyond the mature registry, proof-trace, machine-interface, and bridge-audit compatibility surface over $W, C, K, L, T, F, G, R, H, P, N$; bridge checking remains on the independent formal axis.

Chapter 1228

A Concrete Naming Certificate for $\mathsf{PointwiseModulusLedger}^{\uparrow}$

The $\mathsf{PointwiseModulusLedger}^{\uparrow}$ interface is the concrete source packet for the pointwise-modulus relation isolated in chapter 9237. It reads the $\mathsf{ContinuousMap}^{\uparrow}$ graph and pointwise-continuity rows of chapter 130, the source and target metric rows of chapter 57, the positive tolerance and radius histories of chapter 37, and the finite selector boundary later consumed by chapter 914. Its role is to store the center-specific continuity implication before the compact finite-net fold has selected a tolerance-only precision.

Falsifiable prediction: If the local radius row is absent, no accepted $\mathsf{PointwiseModulusLedger}^{\uparrow}$ packet can hand a center-specific continuity implication to a compact-net selector.

Independence witness: continuousmap, metric, rat, compactnetmodulusselector, compactuniformcontinuity: the packet is not bijective to any one sibling because it binds source and target metric rows, graph row, center row, tolerance row, local radius row, and implication certificate before compact folding; each sibling omits at least one of those source-ledger obligations.

Definition 1228.1 (Pointwise modulus ledger carrier)**.** *A* $\mathsf{PointwiseModulusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, F, C, E, R, I, H, Q, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose distance comparisons are tested near the displayed center;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance comparisons read the requested tolerance;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row over the same source and target metrics;*
4. *C is the center history at which the local continuity row is requested;*
5. *E is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history;*
6. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *local radius history attached to C and E;*
7. *I is the implication certificate saying that every source history within radius R of C is mapped by F within tolerance E in Y;*
8. *H records componentwise* hsame *transport for the metric, graph, center, tolerance, radius, implication, route, provenance, and naming rows;*

9. *Q records the displayed* Cont *routes through which continuous-map and compact-net consumers read the packet;*
10. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseModulusLedger}^{\uparrow}}$ *row.*

The carrier exports only the displayed source metric, target metric, graph, center, tolerance, radius, implication, transport, route, provenance, and local naming rows. It has no coordinate for compactness, a finite-net selector, a folded lower bound, a host continuity theorem, a chosen global modulus, or an open-cover argument.

Theorem 1228.2 (Pointwise modulus ledger NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PointwiseModulusLedger}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet of Definition 1228.1;*
2. *graph compatibility, requiring F to be read as a* $\mathsf{ContinuousMap}^{\uparrow}$ *row from the source metric X to the target metric Y;*
3. *center-tolerance-radius exactness, requiring C, E, and R to be displayed together as the local pointwise request;*
4. *implication exactness, requiring I to consume only the source-distance row in X, the target-distance row in Y, the graph row F, and the displayed C, E, R rows;*
5. *non-escape, requiring every compact-net consumer read to pass through $X, Y, F, C, E, R, I, H, Q, P, N$ and forbidding compactness data, finite selector output, folded lower bounds, host continuity, chosen global moduli, and open-cover subcovers.*

Proof. Carrier admission is projection from Definition 1228.1. The packet displays the source and target metrics, graph row, center, tolerance, radius, implication, transports, routes, provenance, and local name.

Graph compatibility reads F only over X and Y. The pointwise row is therefore attached to the same metric surfaces later used by a compact-net selector.

Center-tolerance-radius exactness binds C, E, R as one local request. The radius row is not reconstructed from compactness or a folded minimum; it is the displayed positive $\mathsf{Rat}^{\uparrow}$ radius carried by the packet.

Implication exactness reads I as the local continuity certificate. Its antecedent is a source-metric comparison to C below R, and its conclusion is a target-metric comparison below E after applying the graph row F.

The non-escape clause follows from the carrier inventory and the displayed route rows. Since no compactness, finite selector, lower-bound fold, host theorem, global modulus, or open-cover coordinate is present, no accepted projection can export such data. □

Theorem 1228.3 (Pointwise modulus ledger center-radius handoff)**.** *Let*

$$L = (X, Y, F, C, E, R, I, H, Q, P, N)$$

be an accepted $\mathsf{PointwiseModulusLedger}^{\uparrow}$ *carrier. Any downstream* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *read of the center-specific pointwise modulus factors through the displayed center row C, local radius row R, tolerance row E, graph row F, source and target metric rows X, Y, and implication certificate I. The selector receives a center-radius implication packet, not a compact finite-net fold or a host modulus.*

Proof. Project the accepted carrier. The center, tolerance, and radius rows are visible as C, E, R, and the graph and metric rows are visible as F, X, Y.

Apply Definition 1228.2. Its graph compatibility and center-tolerance-radius exactness clauses bind these rows before any consumer can read the implication certificate.

Now read I. The implication certificate consumes exactly the source-distance comparison to C below R and returns exactly the target-distance comparison below E through F and Y.

The compact-net selector can therefore use the row only as a center-radius handoff. It must still supply its own finite ProbeBundle spine and lower-bound fold, because neither coordinate appears in the accepted PointwiseModulusLedger$^{\uparrow}$ packet. □

Theorem 1228.4 (Pointwise modulus ledger non-escape)**.** *Every consumer read of an accepted* PointwiseModulusLedger$^{\uparrow}$ *packet factors through the displayed source metric, target metric, graph, center, tolerance, radius, implication, transport, route, provenance, and local naming rows of Definition 1228.1. The packet exports no compact finite-net row, no selected precision, no lower-bound fold, no open-cover compactness, no host continuity theorem, and no external modulus function.*

Proof. Read the carrier coordinates $X, Y, F, C, E, R, I, H, Q, P, N$. These are the only rows available to the local NameCert$_{\text{PointwiseModulusLedger}^{\uparrow}}$ projection.

Use Definition 1228.2. Its exactness clauses route graph reads through F, local request reads through C, E, R, and continuity implication reads through I.

The transport, continuation, provenance, and naming rows H, Q, P, N replay that same finite route componentwise. No coordinate remains from which a consumer could read compact finite-net data, a selected precision, a lower-bound fold, open-cover compactness, host continuity, or an external modulus function. □

Theorem 1228.5 (Pointwise modulus ledger TasteGate obligation)**.** *The Lean follow-up file* lean4/BEDC/Derived/PointwiseModulusLedgerUp/TasteGate.lean *must provide* Nontrivial PointwiseModulusLedger$^{\uparrow}$ *and* FieldFaithful PointwiseModulusLedger$^{\uparrow}$ *instances for the same carrier. These instances must preserve the source metric row, target metric row, graph row, center row, tolerance row, local radius row, implication certificate,* hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.PointwiseModulusLedgerUp.TasteGate.PointwiseModulusLedgerTasteGate single carrier alignment`

Proof. Nontriviality is witnessed by the dependency mix of the carrier. Removing the source metric row breaks the antecedent of I; removing the target metric row breaks its conclusion; removing the graph, center, tolerance, or radius row leaves no pointwise-modulus request.

Field faithfulness records that these rows are preserved as one packet. A TasteGate consumer must distinguish PointwiseModulusLedger$^{\uparrow}$ from ContinuousMap$^{\uparrow}$, Met$^{\uparrow}$, Rat$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, and the compact-uniform-continuity packet by the full coordinate surface $X, Y, F, C, E, R, I, H, Q, P, N$. □

Closure status for NameCert$_{\text{PointwiseModulusLedger}^{\uparrow}}$.

Constructive story: The pointwise modulus ledger is generated from $BHist$ rows that record source and target metric rows, a graph row, center, tolerance, local radius, and the implication certificate attached to that point. The carrier convention reads these rows directly: first the pointwise address, then the radius budget, then the modulus ledger row consumed by later proofs. The $hsame$ relation transports the same pointwise packet across representative changes, while Cont may extend the route only if the center-radius evidence remains visible. The NameCert obligations, handoff theorem, non-escape boundary, and TasteGate obligation therefore make the object a ledger of local modulus use, not a hidden uniform modulus principle.

Theory closure: TheoryGate(NameCert$_{\mathsf{PointwiseModulusLedger}\uparrow}$, seed)

Scope closed: This chapter binds the pointwise modulus-ledger carrier, NameCert obligations, center-radius handoff, non-escape, and TasteGate-obligation theorems.

Formal verification: FormalStatus(NameCert$_{\mathsf{PointwiseModulusLedger}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim a global uniform modulus, host metric completeness, or pointwise data outside the displayed center-radius ledger.

Upgrade path: Reaching obligationClosure requires formal pointwise ledger rows and checked preservation of the center-radius handoff under $hsame$.

Chapter 1229

A Concrete Naming Certificate for GapFailureBridgeAudit$^{\uparrow}$

The GapFailureBridgeAudit$^{\uparrow}$ packet realizes the gap-report, failure-certificate, and standard-bridge audit surface of subsection 9462.30.9 and subsection 9462.30.10 as one finite NameCert carrier. It sits beside the certificate gate of chapter 1002, the claim registry layer of chapter 1209, the standard bridge audit packet of chapter 1217, the host primitive leakage boundary of chapter 1157, and the effective replacement ledger of chapter 568. Those siblings gate certificate export, register claims, display bridge fields, refuse host leakage, and record replacement discipline; this packet binds the exact point where a typed gap, a failure branch, and a bridge-preservation row must be inspected together.

Definition 1229.1 (Gap-failure-bridge audit carrier)**.** *A* GapFailureBridgeAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, F, B, L, X, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed gap-report row, recording site, requested supply, socket type, gate, consequence if internalized, and admitted route.*
2. *F is the failure-certificate row, recording the branch that blocks a theory grade, a formal status, or their verified combination.*
3. *B is the standard-bridge row, recording encoding, decoding, two return witnesses, finite theorem preservation, and bridge scope.*
4. *L is the no-host-leak row, recording host primitives, quotient principles, semantic equality assumptions, choice, propositional extensionality, and compiler readings that are absent from the BEDC public surface.*
5. *X is the axis-separation row, recording that theory closure, formal verification, and bridge status are read as distinct coordinates.*
6. *A is the audit-route row, recording the certificate gate, registry entry, standard-bridge packet, host-leak refusal, and replacement-ledger route consulted by the audit.*
7. *H stores componentwise* hsame *transport for gap reports, failure certificates, bridge rows, no-host-leak rows, axis-separation rows, audit routes, provenance, and local naming.*

8. *C stores displayed* Cont *replay for gap inspection, failure-branch lookup, bridge preservation, no-host-leak inspection, and axis-separation checking.*
9. *P is the* Pkg *provenance row over* $\mathsf{CertificateAuditGate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, $\mathsf{HostPrimitiveLeakage}^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GapFailureBridgeAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untyped gap, an unbranched failure, a bridge row without no-host-leak evidence, a bridge-checked reading detached from theorem preservation, or a conversion of bridge status into theory closure.

Lean4 ✓ `BEDC.Derived.GapFailureBridgeAuditUp.GapFailureBridgeAuditTasteGate single carrier alignment`

Theorem 1229.2 (Gap-failure-bridge audit NameCert obligations)**.** *For the carrier of Definition 1229.1, the local* $\mathsf{NameCert}_{\mathsf{GapFailureBridgeAudit}^{\uparrow}}$ *obligation surface consists exactly of typed gap admission through T, failure-branch admission through F, bridge-field admission through B, no-host-leak inspection through L, axis separation through X, audit routing through A, transport and replay through H, C, provenance through P, and local naming through N.*

Proof. Carrier admission projects the tuple $G = (T, F, B, L, X, A, H, C, P, N)$. The typed gap report, failure certificate, bridge row, no-host-leak row, axis-separation row, audit route, transport, replay, provenance, and local name are the only displayed coordinates.

Use chapter 1002 and chapter 1209 for gate and registry exposure. They ensure that a cited audit packet is read through an admitted certificate surface and through a registered or refused claim row.

Use chapter 1217, chapter 1157, and chapter 568 for bridge preservation, host-leak refusal, and replacement discipline. These sibling surfaces account for B, L, A, P without importing any host primitive as a BEDC input.

The remaining rows H, C, N preserve and name the same finite packet. Since no other row is available, every local NameCert obligation for the audit surface is one of the displayed rows above. □

Lean4 ✓ `BEDC.Derived.GapFailureBridgeAuditUp.GapFailureBridgeAuditPacket namecert obligations`

Theorem 1229.3 (Gap-failure-bridge audit axis separation)**.** *An accepted* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *packet cannot read bridge status as theory closure and cannot read a failure certificate as a checked theorem. The row X keeps theory closure, formal verification, bridge status, and failure blocking as distinct audit coordinates over the same typed gap, failure branch, bridge row, no-host-leak row, audit route, transport, replay, provenance, and local name.*

Proof. Start with the axis-separation row in Definition 1229.1. The row X is a separate carrier coordinate rather than a predicate computed from B or F.

Apply Definition 1229.2. The typed gap row, failure-certificate row, standard-bridge row, and no-host-leak row are admitted separately before transport or replay can export the audit packet.

Thus a bridge-preservation row cannot promote itself to theory closure, and a failure branch cannot promote itself to formal verification. Any such reading would identify two coordinates that the accepted packet stores separately. □

Lean4 ✓ `BEDC.Derived.GapFailureBridgeAuditUp.GapFailureBridgeAuditPacket axis separation`

Theorem 1229.4 (Gap-failure-bridge audit non-escape)**.** *Every accepted bridge-audit read through* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *factors through*

$$T, F, B, L, X, A, H, C, P, N.$$

If either the typed gap row T or the failure-certificate row F is absent, the packet may still record a bridge-writing attempt, but it cannot export a bridge-checked reading.

Proof. Project the carrier. The public audit surface is exactly $T, F, B, L, X, A, H, C, P, N$, so a route that omits T or F is not a route through the accepted packet.

The standard-bridge row B supplies bridge fields, and the no-host-leak row L refuses host import. These rows still require the typed gap and failure-certificate rows to explain which missing supply and which blocking branch are being audited.

Finally use the audit-route, transport, replay, provenance, and local naming rows A, H, C, P, N. They can preserve the displayed packet, but they cannot manufacture a missing typed gap or failure branch. Therefore a bridge-writing attempt without T or F has no bridge-checked export through $\mathsf{GapFailureBridgeAudit}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.GapFailureBridgeAuditUp.GapFailureBridgeAuditPacket nonescape`

Theorem 1229.5 (Gap-failure-bridge audit public surface)**.** *The public audit surface of an accepted* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *packet is exactly*

$$T, F, B, L, X, A, H, C, P, N.$$

It exposes bridge status, theory closure, formal verification, and failure blocking as separate audit coordinates over the same typed gap, failure branch, bridge row, no-host-leak row, audit route, transport, replay, provenance, and local naming data.

Proof. Project the carrier of Definition 1229.1. Its displayed coordinates are precisely the typed gap row T, failure-certificate row F, bridge row B, no-host-leak row L, axis-separation row X, audit route A, transport H, replay C, provenance P, and local name N.

Apply Definition 1229.3. The row X prevents bridge status, theory closure, formal verification, and failure blocking from being read as the same coordinate.

Use Definition 1229.4. Every accepted bridge-audit read factors through the displayed tuple, so the public audit surface has no hidden row beyond $T, F, B, L, X, A, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.GapFailureBridgeAuditUp.GapFailureBridgeAuditPacket public surface`

Falsifiable prediction: An accepted bridge audit packet that omits either the typed gap row or the failure-certificate row cannot export a bridge-checked reading.

Independence witness: certificateauditgate, claimregistrylayer, standardbridgeauditpacket, hostprimitiveleakage, effective-replacement-ledger: none is bijective because the packet stores gap report, failure branch, bridge preservation, no-host-leak, and axis-separation rows together.

The Lean site `lean4/BEDC/Derived/GapFailureBridgeAuditUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ and FieldFaithful $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ instances preserving $T, F, B, L, X, A, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{GapFailureBridgeAudit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ packet is generated from a typed gap report, a failure branch, a bridge row, no-host-leak evidence, axis separation, audit routing, transport, replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GapFailureBridgeAudit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1229.1, Definition 1229.2, Definition 1229.3, Definition 1229.4, and Definition 1229.5 bind the public audit surface, the standard-bridge row B, and the axis-separation theorem for typed gaps, failure branches, bridge rows, no-host-leak rows, audit routing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GapFailureBridgeAudit}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim bridge checking, collapse theory closure into formal verification, import host primitives, admit untyped gaps, or export unbranched failure claims.

Upgrade path: No further paper-side theory-grade upgrade remains; Lean bridge checking remains a separate verification-axis task.

Chapter 1230

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$

The $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ packet realizes the shared tail-window witness used by the Real-and-Completion route of chapter 9238. It sits between the regular Cauchy data of section 9238.1 and the Real-facing seal consumers, and it reuses the finite dyadic rows of chapter 291, the schedule rows of chapter 126, the regular-rational readbacks of chapter 301, and the terminal naming seal of chapter 38. The chapter is composite: it assembles already named Real and Completion rows into one reusable tail certificate rather than claiming a new atomic carrier.

Definition 1230.1 (Regular Cauchy tail-certificate carrier)**.** *A* $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. X *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source row whose tail is being witnessed;*
2. W *is the* $\mathsf{StreamName}^{\uparrow}$ *shared tail-window row naming the finite observation window selected for the displayed precision request;*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the window* W*;*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger for the windowed readback;*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable from* X, W, R, D*;*
6. H *records componentwise* hsame *transports for* X, W, R, D, E *and the local naming row;*
7. C *records displayed* Cont *routes from the regular Cauchy source through the shared tail window, readback, dyadic ledger, and Real seal;*
8. P *is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailCertificate}^{\uparrow}}$ *row.*

The classifier compares carriers only by the rows $X, W, R, D, E, H, C, P, N$*. It has no coordinate for quotient streams, host sequence equality, selected limits, choice rows, standard-bridge rows, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.taste gate`

Theorem 1230.2 (Regular Cauchy tail-certificate NameCert obligations)**.** *For the carrier of Definition 1230.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailCertificate}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier exposure for the packet* $T = (X, W, R, D, E, H, C, P, N)$*;*
2. *shared tail-window selector coverage through the* $\mathsf{StreamName}^{\uparrow}$ *row* W*;*
3. *dyadic readback exactness through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*;*
4. *Real-seal factorization through the row* E *and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows;*
5. *non-escape for quotient streams, host sequence equality, selected limits, choice rows, standard-bridge rows, and ambient Real-completion principles.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificateNameCertObligations`

Proof. Carrier exposure is projection from Definition 1230.1. The displayed tuple lists the regular Cauchy source, shared window, readback, dyadic ledger, Real seal, transport, continuation, provenance, and local naming rows.

Selector coverage is read from W before any numerical tolerance is consumed. Since W is the only shared tail-window row in the packet, the local certificate cannot redirect the request to an unnamed tail schedule.

The dyadic readback and Real-seal obligations then inspect R, D, E, H, C, P, N. These rows keep the regular-rational readback and tolerance ledger visible before the seal is reached, while the support rows only transport and replay the same finite route.

The excluded objects are absent from the carrier. Thus the local NameCert exports no quotient stream, host sequence equality, selected limit, choice row, standard bridge, or ambient completion theorem. □

Theorem 1230.3 (Regular Cauchy tail-certificate selector coverage)**.** *Every accepted* $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ *packet exposes a single shared tail-window selector before any* $\mathsf{Real}^{\uparrow}$ *seal consumer can read the certificate. If*

$$T = (X, W, R, D, E, H, C, P, N),$$

then every consumer route first reads the regular Cauchy source X *and the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and only then reaches* E *through* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificateSelectorCoverage`

Proof. Project the carrier T. The shared tail-window coordinate is W, and the source row X is the only regular Cauchy row from which that window is read.

Apply the NameCert obligations of Definition 1230.2. They force selector coverage before dyadic readback exactness and before Real-seal factorization.

The route to E is therefore ordered by the displayed packet: source, shared window, readback, dyadic ledger, Real seal, and then the transport, continuation, provenance, and naming support rows. No second selector or hidden tail oracle is available. □

Theorem 1230.4 (Regular Cauchy tail-certificate non-escape)**.** *An accepted* $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ *carrier exports only the finite rows* $X, W, R, D, E, H, C, P, N$ *and their local classifier transports. A Real-and-Completion consumer may use the shared tail window, regular-rational readback, dyadic*

tolerance ledger, and Real$^{\uparrow}$ *seal exactly as displayed; it receives no quotient stream, no host sequence equality, no selected limit, no choice row, no standard-bridge row, and no ambient completeness theorem.*

Proof. Inspect the carrier coordinates. They contain finite rows for source, window, readback, dyadic tolerance, Real seal, transport, continuation, provenance, and local naming.

Use Definition 1230.3. Any consumer reaches the Real-facing seal only through the displayed selector, readback, and dyadic ledger.

Since quotient streams, host sequence equality, selected limits, choice rows, standard bridges, and ambient completeness principles are not coordinates of T, neither componentwise hsame transport nor displayed Cont replay can export them. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificateNonescape`

Theorem 1230.5 (Regular Cauchy tail-certificate vision handoff)**.** *The regular Cauchy construction of section 9238.1 may hand its shared tail-window witness to the Real-and-Completion layer through the* RegularCauchyTailCertificate$^{\uparrow}$ *carrier of Definition 1230.1. The handoff is compatible with chapter 291, chapter 126, chapter 301, and chapter 38 because the packet preserves their dyadic tolerance, stream-window, regular readback, and Real-seal rows inside one local* NameCert *surface.*

Proof. Read the upstream vision as a demand for a finite shared tail witness, not as a demand for a completed host limit. The carrier supplies that witness by exposing X and W as the source and shared tail-window rows.

Read the sibling concrete instances next. The RegSeqRat$^{\uparrow}$ row R and DyadicRatCore$^{\uparrow}$ ledger D keep the regular readback and arithmetic tolerance explicit, while the Real$^{\uparrow}$ row E is downstream of those observations.

Finally apply the non-escape boundary of Definition 1230.4. The handoff therefore gives later Real and Completion consumers a finite tail certificate over the named sibling rows and no quotient stream, host sequence equality, selected limit, choice row, standard bridge, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificateVisionHandoff`

Theorem 1230.6 (Regular Cauchy tail-certificate obligation closure package)**.** *For the carrier of Definition 1230.1, the carrier row, shared tail-window selector coverage, dyadic readback exactness, Real-seal factorization, non-escape clauses, and vision handoff form the paper-side* obligation *package for* RegularCauchyTailCertificate$^{\uparrow}$ *over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package contains no quotient stream, host sequence equality, selected limit, choice row, standard bridge row, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificateObligationClosurePackag`

Proof. Project the carrier definition. The tuple $T = (X, W, R, D, E, H, C, P, N)$ exposes exactly the regular Cauchy source, shared tail-window row, readback row, dyadic ledger, Real seal, componentwise transport, displayed continuation, provenance, and local naming row.

Apply Definition 1230.2. Its five rows supply carrier exposure, selector coverage, dyadic readback exactness, Real-seal factorization, and non-escape for the same finite packet.

Use Definition 1230.3 and Definition 1230.4 to fix the route order and exclusions. The consumer reads X, W, R, D before E, and no absent quotient, host-equality, selected-limit, choice, bridge, or ambient-completeness coordinate can be exported by hsame transport or Cont replay.

Finally apply Definition 1230.5. The Real-and-Completion consumer route therefore receives precisely the finite L10 tail certificate carried by the displayed Pkg and NameCert rows, which is the claimed obligation-closure package. □

Theorem 1230.7 (Regular Cauchy tail-certificate primitive scope lock)**.** *Every accepted* RegularCauchyTailCertifica *packet is scoped exactly by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows over* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. For*

$$T = (X, W, R, D, E, H, C, P, N),$$

the accepted route reads the regular Cauchy source X, shared tail-window row W, regular-rational readback R, dyadic ledger D, Real seal E, componentwise transport row H, displayed continuation row C, package provenance P, and local naming row N, and no quotient stream, host sequence equality, selected limit, choice row, standard bridge, ambient completeness principle, or metric-completeness coordinate may be substituted for those primitive rows.

Proof. Project the carrier tuple $T = (X, W, R, D, E, H, C, P, N)$. The projection exposes exactly the rows named in the statement: the regular Cauchy source, shared tail window, regular-rational readback, dyadic tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Apply the NameCert obligations of Definition 1230.2. They bind carrier exposure, selector coverage, dyadic readback exactness, Real-seal factorization, and non-escape to the same displayed tuple, so the accepted certificate has no second route on which a host-side or quotient-side row could enter.

Use selector coverage, non-escape, and the vision handoff from Definition 1230.3, Definition 1230.4, and Definition 1230.5. These results force the Real-and-Completion consumer to pass through X, W, R, D before E, and then through H, C, P, N as support rows.

Finally apply the obligation closure package of Definition 1230.6. Since that package already lists the accepted rows and excludes quotient streams, host sequence equality, selected limits, choice rows, standard bridges, ambient completeness principles, and metric-completeness coordinates, the scoped closure is exactly the primitive BEDC route displayed above. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCertificateUp.RegularCauchyTailCertificate primitive scope lock`

Closure status for NameCert$_{\text{RegularCauchyTailCertificate}^{\uparrow}}$.

Constructive story: A RegularCauchyTailCertificate$^{\uparrow}$ packet is generated from a regular Cauchy source row, a shared StreamName tail-window row, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailCertificate}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1230.7 binds the carrier, five NameCert obligation rows, selector coverage, dyadic readback exactness, Real-seal factorization, non-escape boundary, vision handoff, and TasteGate instance `BEDC.Derived.RegularCauchyTailCertific` to the finite primitive route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailCertificate}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.RegularCauchyTailCertificateUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close quotient streams, host sequence equality, selected limits, choice rows, standard-bridge rows, ambient Real completeness, theorem-level Lean proof, audit-clean verification, or a public standard bridge.

Upgrade path: A stronger grade requires theorem-level Lean proof, audit-clean verification of the primitive route, and any later public or standard bridge demanded by downstream Real and Completion consumers.

Chapter 1231

A Concrete Naming Certificate for $\mathsf{TailCofinalityBudget}^{\uparrow}$

The $\mathsf{TailCofinalityBudget}^{\uparrow}$ packet realizes the shared finite tail threshold consumed by the Real-completion route of section 9238.2. It is a concrete NameCert surface over the finite tail filter of chapter 1119, the stream-window naming of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, the Real seal of chapter 38, the finite-prefix stability route of chapter 1226, and the selected endpoint seed of chapter 1207. The packet records the common cofinal threshold and the terminal seal handoff without exporting a selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness.

Falsifiable prediction: No accepted TailCofinalityBudgetUp packet exports a selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness outside the displayed finite tail rows.

Independence witness: finite-tail-filter, finite-prefix-limit-stability, selected-tail-seed, stream-name, regseqrat: TailCofinalityBudgetUp is not bijective to any listed sibling because it records the shared cofinal threshold and seal handoff row, while those siblings respectively record the broad filter surface, prefix stability, selected endpoint, pointwise stream naming, or regular-rational readback without the same cofinal-budget packet.

Definition 1231.1 (Tail cofinality budget carrier)**.** *A* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite tail request row naming the common cofinal threshold consumed by the Real-completion read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger exposed only at or beyond the threshold named by R.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same threshold and window.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed stream window and dyadic tolerance.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after R, W, D, Q are visible in the packet.*
6. *H records componentwise* hsame *transport for R, W, D, Q, E and the local certificate row.*
7. *C records displayed* Cont *replay from finite tail request through stream window, dyadic tolerance, regular-rational readback, and Real seal.*

8. *P is the* Pkg *provenance over* TailCofinalityBudget$^\uparrow$, FiniteTailFilter$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, FinitePrefixLimitStability$^\uparrow$, SelectedTailSeed$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{TailCofinalityBudget}^\uparrow}$ *row.*

The carrier exports only the finite threshold, the threshold-indexed observation rows, the Real seal handoff, and the structural transport, replay, provenance, and naming rows displayed above.

Theorem 1231.2 (Tail cofinality budget NameCert obligations)**.** *For the carrier of Definition 1231.1, the local* NameCert$_{\text{TailCofinalityBudget}^\uparrow}$ *surface consists exactly of carrier admission for $T = (R, W, D, Q, E, H, C, P, N)$ finite tail request admission through R,* StreamName$^\uparrow$ *window exposure through W,* DyadicRatCore$^\uparrow$ *tolerance exposure through D,* RegSeqRat$^\uparrow$ *readback exposure through Q, terminal* Real$^\uparrow$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudget namecert obligations`

Proof. Project the carrier. Its coordinates list the finite tail request, stream window, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local naming row.

Read the sibling surfaces in the displayed order. The finite tail filter supplies the threshold discipline for R; the StreamName$^\uparrow$ chapter supplies W; the DyadicRatCore$^\uparrow$ chapter supplies D; the RegSeqRat$^\uparrow$ chapter supplies Q; and the Real$^\uparrow$ chapter supplies E.

The remaining rows are structural. The FinitePrefixLimitStability$^\uparrow$ and SelectedTailSeed$^\uparrow$ siblings identify the prefix-stability and selected-seed routes that may consume the same threshold, while H, C, P, N only transport, replay, source, and name the rows already displayed. These rows exhaust the local NameCert obligations. □

Theorem 1231.3 (Tail cofinality budget carrier obligation)**.** *Every accepted* TailCofinalityBudget$^\uparrow$ *carrier $T = (R, W, D, Q, E, H, C, P, N)$ admits exactly the finite rows $R, W, D, Q, E, H, C, P, N$ before any Real-completion consumer may read the packet. The carrier admission route passes through the finite tail request R, the threshold-indexed* StreamName$^\uparrow$ *window W, the* DyadicRatCore$^\uparrow$ *tolerance ledger D, the* RegSeqRat$^\uparrow$ *readback Q, the* Real$^\uparrow$ *seal E, and the displayed structural rows H, C, P, N; it exports no selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetCarrierObligation`
↪ variant `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetObligationCarrier`

Proof. Project the carrier of Definition 1231.1. The tuple already lists the only admissible coordinates: a finite tail request, a stream window, a dyadic tolerance ledger, a regular-rational readback, a terminal seal, and the transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 1231.2. They force every accepted consumer to expose those rows in the displayed order before the packet can be used.

Finally use threshold monotonicity and filtered-tail scope, Definition 1231.6 and Definition 1231.7. Refining the threshold keeps the same finite route, and the filtered-tail read has no coordinate from which a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness can be projected. □

Theorem 1231.4 (Tail cofinality budget classifier obligation)**.** *Let $T = (R, W, D, Q, E, H, C, P, N)$ and $T' = (R', W', D', Q', E', H', C', P', N')$ be accepted* TailCofinalityBudget$^\uparrow$ *packets. If the visible coordinates are componentwise related by* hsame *along the displayed transport rows, then the*

classifier preserves the same finite cofinal-threshold route from R through W, D, Q, E and from R' through W', D', Q', E'. The transported classifier does not add a selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetClassifierObligation`

Proof. Read the classifier from the carrier and the NameCert surface. It compares packets only through the finite threshold row, stream window, tolerance ledger, regular-rational readback, Real seal, and structural rows.

Transport each displayed coordinate through the componentwise hsame row H and its counterpart H'. The threshold row remains a finite tail request, W, D, Q remain the observation and readback rows attached to that request, and E remains downstream of them.

Use threshold monotonicity to keep later threshold refinements inside the same route, then use filtered-tail scope to keep the Real-facing read inside the same finite tail filter. Since the absent objects are not classifier coordinates, componentwise transport cannot create them. □

Theorem 1231.5 (Tail cofinality budget ledger obligation)**.** *Every ledger read of an accepted* TailCofinalityBudget$^{\uparrow}$ *packet that reaches a Real-completion consumer factors through the finite cofinal threshold route*

$$R \longrightarrow W \longrightarrow D \longrightarrow Q \longrightarrow E$$

together with H, C, P, N. The ledger scope is nonescaping: after the common threshold is fixed, the consumer may refine along later finite tails, but it may not read a selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness outside the displayed packet.

Proof. Start from the filtered-tail scope theorem, Definition 1231.7. It identifies the cofinal finite-tail route and keeps later thresholds on the same displayed request row.

Use threshold monotonicity for refinements beyond the common threshold. The rows W, D, Q, E are preserved as the threshold-indexed window, dyadic tolerance, regular readback, and terminal seal rather than replaced by a completion object.

Finally apply nonescape, Definition 1231.8. The structural rows H, C, P, N transport and name the ledger route, but they do not export any selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetLedgerObligation`

Theorem 1231.6 (Tail cofinality budget threshold monotonicity)**.** *Let $T = (R, W, D, Q, E, H, C, P, N)$ be an accepted* TailCofinalityBudget$^{\uparrow}$ *packet. If T serves a displayed threshold and a later unary threshold refines R through the same finite tail request route, then the refined packet preserves the same* StreamName$^{\uparrow}$ *window row W,* DyadicRatCore$^{\uparrow}$ *tolerance row D,* RegSeqRat$^{\uparrow}$ *readback row Q,* Real$^{\uparrow}$ *seal row E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

The refined read exports no selected limit, quotient stream equality, host real equality, countable-choice row, or ambient Real$^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetThresholdMonotonicity`

Proof. Project the carrier of Definition 1231.1. The row R is the only finite tail request coordinate, so a later unary threshold can refine the packet only by traveling through that same request route.

Compare the thresholds inside R. Once the refined request is accepted, the observation side of the carrier is still the displayed route W, D, Q, E: the $\mathsf{StreamName}^{\uparrow}$ window is read at the refined tail, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger remains attached to that window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback is replayed from the same finite data, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after those rows are visible.

Finally replay the structural rows. Transport through H, continuation through C, provenance through P, and local naming through N preserve the accepted packet rather than adding a new completion object. Since the carrier has no coordinate for a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness, the threshold refinement stays inside the displayed finite rows. □

Theorem 1231.7 (Tail cofinality budget filtered-tail scope)**.** *Let $T = (R, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *packet. If the finite threshold row R is consumed together with a pointwise* $\mathsf{StreamName}^{\uparrow}$ *carrier and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row Q, then the packet determines a cofinal finite-tail filter route whose carrier transport is stable under* hsame *and whose* $\mathsf{Real}^{\uparrow}$ *consumer sees only the displayed threshold ledger.*

In particular, later thresholds refine the same finite route

$$R \longrightarrow W \longrightarrow D \longrightarrow Q \longrightarrow E$$

through the finite tail filter and selected-tail seed dependencies. No selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness is exported by the filtered-tail scope.

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudget filtered tail scope`

Proof. Start with threshold monotonicity. The row R is the unique finite request coordinate, so every later threshold must refine the accepted packet through the same displayed request route.

Compose that monotone request with the finite tail filter surface and the selected-tail seed route. The pointwise $\mathsf{StreamName}^{\uparrow}$ row W supplies the displayed observations, while D and Q attach the dyadic tolerance and $\mathsf{RegSeqRat}^{\uparrow}$ regularity needed by the filtered tail.

Finally pass the route to the $\mathsf{Real}^{\uparrow}$ seal row E and transport it by H, C, P, N. Since every consumer sees only R, W, D, Q, E and the structural rows, the filtered-tail scope cannot read a selected limit, host real equality, quotient stream equality, countable choice, or an ambient completeness primitive. □

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetFilteredTailScope`

Theorem 1231.8 (Tail cofinality budget nonescape)**.** *Every Real-completion consumer of an accepted* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *packet factors through the finite route*

$$R, W, D, Q, E, H, C, P, N.$$

The consumer receives the shared finite cofinal threshold, the $\mathsf{StreamName}^{\uparrow}$ *window ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the structural support rows. It receives no selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness outside those displayed finite tail rows.*

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudget nonescape`

Proof. Begin with Definition 1231.1. The only Real-facing coordinate is the seal handoff E, and it is downstream of the threshold row R, window row W, dyadic row D, and readback row Q.

Apply Definition 1231.2. It forces every accepted consumer to expose the finite tail request, stream window, tolerance ledger, regular-rational readback, seal handoff, transport, replay, provenance, and local name before the packet is usable.

The excluded objects are not coordinates of the carrier. Since H, C, P, N preserve and name displayed rows without creating analytic data, no accepted consumer can project a selected limit, quotient stream equality, host real equality, countable-choice row, or ambient completeness witness from this packet. □

Theorem 1231.9 (Tail cofinality budget TasteGate obligation)**.** *The Lean boundary for* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *must supply a nonempty carrier and a field-faithful observation of* $R, W, D, Q, E, H, C, P, N$*. In concrete terms, the future file* `lean4/BEDC/Derived/TailCofinalityBudgetUp/TasteGate.lean` *must provide* $\mathsf{Nontrivial\ TailCofinalityBudget}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ TailCofinalityBudget}^{\uparrow}$ *instances whose projections preserve the finite tail threshold and the terminal Real seal handoff.*

Proof. The carrier is meaningful only when the displayed threshold route can be inhabited. Nontriviality supplies such a packet without replacing the finite rows by an empty or vacuous surface.

Field faithfulness is the corresponding projection discipline. It prevents a consumer from identifying two packets while forgetting the threshold row R, the observation rows W, D, Q, the seal row E, or the structural rows H, C, P, N.

Together these two obligations are exactly the TasteGate boundary needed before the packet can be used by the Real-completion route as a shared finite tail budget. □

Lean4 ✓ `BEDC.Derived.TailCofinalityBudgetUp.TailCofinalityBudgetTasteGate obligation surface`

Closure status for $\mathsf{NameCert}_{\mathsf{TailCofinalityBudget}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TailCofinalityBudget}^{\uparrow}$ packet is generated from a finite tail request, threshold-indexed stream window, dyadic tolerance ledger, regular-rational readback, Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TailCofinalityBudget}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1231.1, Definition 1231.2, Definition 1231.6, Definition 1231.7, Definition 1231.8, and Definition 1231.9 bind the shared finite cofinal threshold, filtered-tail route, nonescape boundary, and TasteGate obligation as a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TailCofinalityBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient streams, selected host limits, host real equality, countable choice, or ambient Real completeness.

Upgrade path: The next paper-axis obligation is a standard bridge route; Lean closure still requires Nontrivial TailCofinalityBudgetUp and FieldFaithful TailCofinalityBudgetUp instances in `lean4/BEDC/Derived/TailCofinalityBudgetUp/TasteGate.lean`.

Chapter 1232

A Concrete Naming Certificate for MachineReadableAuditInterface$^{\uparrow}$

The MachineReadableAuditInterface$^{\uparrow}$ packet realizes the machine-readable audit surface implicit in chapter 9462 and the interface-status boundary of subsection 9462.35.10. It sits beside the socket router of chapter 1059, the formal target packet of chapter 1225, the registry export surface of chapter 1216, and the consistency gate of chapter 1224. Those siblings route targets, store target skeletons, expose registry exports, and check registry consistency; this packet names the finite interface by which a machine consumer reads audit data without treating a report row as theorem evidence.

Definition 1232.1 (Machine-readable audit interface carrier)**.** *A* MachineReadableAuditInterface$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, C, E, R, F, H, K, P, N)$$

with the following displayed rows:

1. *S is the schema row, recording the finite field names, row types, and row-order constraints available to the machine consumer;*
2. *C is the checker row, recording the local validation route that accepts or rejects a displayed schema instance;*
3. *E is the export row, recording the registry entry, machine target, closure-status reading, and public export grade exposed by the packet;*
4. *R is the refusal row, recording cannot-claim conflicts, blocked exports, and non-evidence flags for report rows;*
5. *F is the report row, recording the machine-readable audit report returned to a consumer after S, C, E, R have been read;*
6. *H records componentwise* hsame *transport for schema, checker, export, refusal, and report rows;*
7. *K records displayed* Cont *replay from schema validation through checker verdict, export readback, refusal inspection, and report publication;*
8. *P is the* Pkg *provenance over* MachineReadableAuditInterface$^{\uparrow}$, MachineInterfaceSocket$^{\uparrow}$, MachineInterfaceFormali RegistryExportSurface$^{\uparrow}$, RegistryExportConsistencyGate$^{\uparrow}$, GroundCompilerAuditMap$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$ *row.*
The carrier has no coordinate for hidden theorem evidence, unregistered export, conflict override, proof-producing report conversion, or off-packet schema repair.

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface carrier semantic name certificate`

Theorem 1232.2 (Machine-readable audit interface field coverage)**.** *For the carrier of Definition 1232.1, the local* $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$A = (S, C, E, R, F, H, K, P, N),$$

schema readback through S, checker readback through C, registry and machine-target export readback through E, refusal and non-evidence readback through R, report readback through F, componentwise hsame *stability through H, displayed* Cont *replay through K, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.taste gate`

Proof. Project the finite carrier. The tuple displays schema, checker, export, refusal, report, transport, replay, provenance, and local naming rows before any machine consumer can read the packet.

Read S, C, E, R, F first. These rows provide the complete machine-facing inventory: schema shape, validation route, export surface, refusal surface, and report payload. None of them contains a hidden proof object or an off-packet registry entry.

Finally use H, K, P, N. These rows transport, replay, package, and name the same inventory. Since the carrier has no further coordinate, the stated rows exhaust the local $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$ surface. □

Theorem 1232.3 (Machine-readable audit interface obligation basis)**.** *For every accepted* MachineReadableAuditInt *packet*

$$A = (S, C, E, R, F, H, K, P, N),$$

any machine consumer that reads an audit report from A must first pass through schema validation by S, checker verdict by C, export and refusal consistency across E and R, report non-evidence control for F, replay and provenance through K and P, and the local $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$ *row N. No report read is accepted by the packet unless these obligation rows are all displayed in the same carrier.*

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface obligation basis`

Proof. Project $A = (S, C, E, R, F, H, K, P, N)$ and apply Definition 1232.2. The field-coverage theorem identifies schema, checker, export, refusal, report, transport, replay, provenance, and local naming as the whole machine-readable audit surface.

Read the operational rows in order. The schema row S fixes the permitted field layout, the checker row C supplies the local validation verdict, and the export and refusal rows E, R determine whether the claimed public read is consistent with the displayed registry surface.

Finally apply Definition 1232.4 to the report row F. The replay row K, provenance row P, and local name N preserve the same non-evidence boundary inside the carrier, so a consumer cannot read an audit report before those obligations have been displayed. □

Theorem 1232.4 (Machine-readable audit interface non-evidence refusal)**.** *If the refusal row R marks the report row F as non-evidence, then no accepted* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *packet exports F as theorem evidence.*

Proof. Begin with an accepted packet

$$A = (S, C, E, R, F, H, K, P, N).$$

The checker row C validates the displayed schema instance, and the export row E exposes registry and machine-target data through that same instance.

Inspect the refusal row R. When R marks F as non-evidence, the report row remains a report payload rather than a theorem-evidence row. The carrier contains no second report coordinate that can bypass this flag.

The transport, replay, provenance, and local naming rows preserve the same flag. Thus a consumer may read the audit report, but any read that treats F as theorem evidence leaves the displayed packet. □

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface non evidence refusal`

Theorem 1232.5 (Machine-readable audit interface sibling independence)**.** *The* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{MachineInterfaceSocket}^{\uparrow}$*,* $\mathsf{MachineInterfaceFormalizatio}$ $\mathsf{RegistryExportSurface}^{\uparrow}$*,* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$*, or* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface sibling independence`

Proof. Compare the displayed rows. $\mathsf{MachineInterfaceSocket}^{\uparrow}$ routes formalization targets, and $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ stores target skeletons, dependency rows, expected status, and audit-gate locality.

$\mathsf{RegistryExportSurface}^{\uparrow}$ exposes registration, blocking, status, machine target, audit, and non-smuggling rows. $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ binds status consistency, audit route, no-smuggling, and formal-target readback. $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ exposes compiler-facing audit rows and report boundaries.

None of these siblings preserves at once the schema row S, checker row C, export row E, refusal row R, report row F, and the local $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$ row N. Any proposed row-preserving bijection loses at least one of those rows, so the packets are not bijective under the NameCert surface. □

Theorem 1232.6 (Machine-readable audit interface falsifiable prediction)**.** *A* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *packet fails exactly when a machine consumer claims an export, theorem-evidence report, or consistency verdict that is not backed by the displayed schema, checker, export, refusal, report, transport, replay, provenance, and local naming rows.*

Proof. Use Definition 1232.2 to read the complete field inventory. Every accepted consumer read must factor through $S, C, E, R, F, H, K, P, N$.

For export claims, inspect E and the replay row K. For theorem-evidence report claims, inspect R and F together with Definition 1232.4. For consistency verdicts, inspect the checker row C and the provenance row P naming the registry consistency gate.

If one of those rows is absent, disagrees with the claimed read, or marks the read as non-evidence, the claim exits the carrier. Conversely a read backed by the displayed rows is precisely the accepted machine-readable audit interface read. □

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface falsifiable prediction`

Falsifiable prediction: A MachineReadableAuditInterface$^{\uparrow}$ packet cannot export a report row as theorem evidence when its refusal row marks the report as non-evidence.

Independence witness: machineinterfacesocket, machineinterfaceformalizationtarget, registryexportsurface, registryexportconsistencygate, groundcompilerauditmap: row loss blocks any NameCert-preserving bijection, because the socket lacks the schema and report rows, the formal-target packet lacks report publication, the export surface lacks checker-local report refusal, the consistency gate lacks the schema-to-report replay, and the GroundCompiler audit map lacks the registry-machine interface packet as one carrier.

The Lean follow-up must provide explicit Nontrivial MachineReadableAuditInterface$^{\uparrow}$ and FieldFaithful MachineRe instances in `lean4/BEDC/Derived/MachineReadableAuditInterfaceUp/TasteGate.lean`. The field-faithfulness witness must preserve $S, C, E, R, F, H, K, P, N$ as distinct rows before BEDC.Derived.MachineReada can be cited by a checked paper marker.

Theorem 1232.7 (Machine-readable audit interface kernel scope)**.** *Every accepted* MachineReadableAuditInterface$^{\uparrow}$ *packet scopes its machine-readable report through the displayed schema row, checker row, export row, refusal row, report row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No report row is theorem evidence unless the audit row is separately checked by the displayed checker and survives the refusal row.*

Proof. Project the carrier of Definition 1232.1. The tuple $A = (S, C, E, R, F, H, K, P, N)$ is the whole machine-readable audit interface, and Definition 1232.2 identifies those rows as the complete local naming surface.

Next apply Definition 1232.3. Any accepted report read must pass through schema validation, checker verdict, export and refusal consistency, report non-evidence control, replay, provenance, and the local NameCert row before the machine consumer can read F.

The non-evidence boundary is fixed by Definition 1232.4. The sibling-independence theorem blocks replacement by socket, formal-target, registry-export, consistency-gate, or compiler-audit packets, and Definition 1232.6 rejects any read not backed by the displayed rows.

Therefore the kernel route for the report is exactly the displayed route through $S, C, E, R, F, H, K, P, N$. A report row can be consumed as machine-readable audit data, but it is not theorem evidence unless the checker row separately accepts it and the refusal row does not mark that evidence claim as blocked. □

Lean4 ✓ `BEDC.Derived.MachineReadableAuditInterfaceUp.MachineReadableAuditInterface kernel scope`

Theorem 1232.8 (Machine-readable audit interface bridge readiness)**.** *A bridge consumer of an accepted* MachineReadableAuditInterface$^{\uparrow}$ *packet may read the schema, checker, export, refusal, report, transport, replay, provenance, and local naming rows. The bridge may not promote the report row to theorem evidence unless the checker row separately accepts the evidence claim and the refusal row leaves that claim unblocked.*

Proof. Use Definition 1232.2. It identifies schema, checker, export, refusal, report, transport, replay, provenance, and local naming as the complete readable surface of the packet.

Next apply Definition 1232.4. The report row can be machine-readable audit data while still being blocked as theorem evidence by the refusal row.

Finally use Definition 1232.7. It places the checker and refusal rows on the same kernel route as the report row, so a bridge consumer can read the report but cannot treat it as theorem evidence without a separate checker acceptance and an unblocked refusal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}$.

Constructive story: A $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ packet is generated from a finite schema row, checker row, export row, refusal row, report row, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1232.1, Definition 1232.2, Definition 1232.4, Definition 1232.5, Definition 1232.6, Definition 1232.7, and Definition 1232.8 bind the carrier, field coverage, non-evidence refusal, sibling independence, falsifiable prediction, kernel-scope route, and bridge-readiness boundary for the machine-readable audit surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MachineReadableAuditInterface}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.MachineReadableAuditInterfaceUp.taste\_gate`

Bridge status: paperBridge

Not claimed: This chapter does not close report rows as theorem evidence, checked machine-readable audit targets, TasteGate alignment, Lean verification, or bridge checking.

Upgrade path: The paper axis has reached mature treatment; Lean verification and bridge checking remain separate formal-axis obligations.

Chapter 1233

A Concrete Naming Certificate for $\mathsf{FailureCertificate}^{\uparrow}$

The $\mathsf{FailureCertificate}^{\uparrow}$ packet realizes the typed failure-certificate discipline of subsection 9462.30.9 as a finite $\mathsf{NameCert}$ carrier for rejected gate attempts. It also sits beside chapter 901, chapter 1163, chapter 1224, chapter 554, and chapter 1059: those siblings expose audit witnesses, reality-constrained defeat surfaces, registry export gates, cannot-claim rows, and machine-facing sockets, while this packet binds a typed failure to the exact theory axis, verification axis, and combined gate it blocks.

Definition 1233.1 (Failure certificate carrier)**.** *A* $\mathsf{FailureCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (N_0, C, V, A, B, D, H, K, P, L)$$

with the following displayed rows:

1. N_0 *is the named claim, chapter, or packet whose gate attempt is being rejected;*
2. C *is the theory-closure row, recording the claimed theory grade and the denied source, classifier, exactness, descent, ledger, stability, scope, bridge, or no-host-leak clause;*
3. V *is the verification-status row, recording the claimed formal grade and the denied checked theorem, encoded definition, audit-clean, axiom-clean, or bridge-checked clause;*
4. A *is the failed-axis row, choosing theory-side failure, verification-side failure, or axis-confusion failure as typed audit data;*
5. B *is the blocked-gate row, naming the theory gate, formal-status gate, or verified gate that the failed-axis row blocks;*
6. D *is the diagnostic row, importing a concrete witness from* $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{RealityConstrainedFailureSu}$ $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, *or* $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ *when the rejected packet crosses those surfaces;*
7. H *stores componentwise* hsame *transport for claim name, closure grade, formal grade, failed axis, blocked gate, diagnostic witness, replay, provenance, and local naming rows;*
8. K *stores displayed* Cont *replay for reading* N_0, C, V, A, B, D *without adding a host repair route;*
9. P *is the* Pkg *provenance over* $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. L is the local $\mathsf{NameCert}_{\mathsf{FailureCertificate}^{\uparrow}}$ *row.*
The carrier contains no coordinate for a private gate override, prose repair, host equality, hidden verifier result, or untyped failure reason outside A, B, D.

Lean4 (definition) `BEDC.Derived.FailureCertificateUp.FailureCertificateUp`

Theorem 1233.2 (Failure certificate gate blocking)**.** *For every accepted* $\mathsf{FailureCertificate}^{\uparrow}$ *packet $F = (N_0, C, V, A, B, D, H, K, P, L)$, the failed-axis row A and blocked-gate row B jointly determine which gate is blocked for the named claim N_0. A theory-side failure blocks the theory gate named by B, a verification-side failure blocks the formal-status gate named by B, and an axis-confusion failure blocks the combined verified gate named by B. No accepted packet can recover the blocked gate from D, H, K, P, L alone.*

Lean4 ✓ `BEDC.Derived.FailureCertificateUp.FailureCertificate gate blocking`

Proof. Project the finite carrier of Definition 1233.1. The row A records the typed failure branch, while B records the gate whose acceptance is denied. These rows are read before diagnostic replay, provenance, or local naming can be consumed.

For a theory-side failure, the denied clause lies in C and the gate named by B is the theory gate. For a verification-side failure, the denied clause lies in V and the gate named by B is the formal-status gate. For axis confusion, A records that the two axes have been collapsed, so the verified gate is blocked even when one row is read without the other.

Finally inspect D, H, K, P, L. The diagnostic row only records the concrete witness, and the remaining rows only transport, replay, package, and name the same packet. Since none of them supplies a fresh axis row or a fresh blocked-gate row, the gate-blocking conclusion depends on A and B together. □

Theorem 1233.3 (Failure certificate axis separation)**.** *The* $\mathsf{FailureCertificate}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, *or* $\mathsf{MachineInterfaceSocket}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FailureCertificateUp.FailureCertificate axis separation`

Proof. Compare the displayed rows. $\mathsf{AuditGateFailureWitness}^{\uparrow}$ supplies concrete audit failure evidence, $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$ supplies finite defeat surfaces, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ supplies export consistency checks, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ supplies refusal rows, and $\mathsf{MachineInterfaceSocket}^{\uparrow}$ supplies machine-facing target sockets.

None of those siblings carries the joint two-axis gate-blocking row of A together with B for the same named claim, while also preserving C, V, D, H, K, P, L. A row-preserving bijection would have to preserve that joint failure-axis and blocked-gate pair, so no such bijection exists. □

Lean4 ✓ `BEDC.Derived.FailureCertificateUp.FailureCertificate axis row separation`

Theorem 1233.4 (Failure certificate public axis export)**.** *The exported* $\mathsf{FailureCertificate}^{\uparrow}$ *public row consists exactly of the typed failure branch, the blocked axis, the diagnostic evidence row, the non-collapse of theory and verification axes, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. A public consumer may read A, B, D, H, K, P, L together with the named claim N_0 and the separate closure rows C, V, but it cannot reinterpret the packet as a private gate override, hidden verifier result, host equality, prose repair route, or untyped failure reason.*

Proof. Project the carrier in Definition 1233.1. The rows A and B expose the typed failure branch and blocked gate, while C and V keep the theory and verification axes separate.

Apply gate blocking and axis separation, Definition 1233.2 and Definition 1233.3. These force the failure evidence to remain tied to the selected axis and prevent sibling audit packets from collapsing the two-axis certificate surface.

The remaining rows D, H, K, P, L provide diagnostic evidence, transport, replay, provenance, and local naming for the same packet. Since no private override, hidden verifier, host equality, prose repair, or untyped reason is a carrier coordinate, the listed rows are exactly the public export surface. □

Theorem 1233.5 (Failure certificate bridged interface)**.** *Every bridge-facing consumer of an accepted* $\mathsf{FailureCertificate}^{\uparrow}$ *packet must preserve the named claim row* N_0*, the separate theory row* C*, the separate verification row* V*, the failed-axis row* A*, the blocked-gate row* B*, the diagnostic row* D*, componentwise transport* H*, displayed replay* K*, provenance* P*, and the local* $\mathsf{NameCert}$ *row* L*. The bridge-facing surface may transport those rows to a standard bridge consumer, but it cannot collapse theory and verification status, drop the failed-axis row, replace the blocked-gate row by a private override, omit the diagnostic witness, or detach the local NameCert row.*

Proof. Start with gate blocking in Definition 1233.2. It makes the failed-axis row A and blocked-gate row B jointly responsible for the refused gate of the named claim N_0, so a bridge consumer cannot preserve the failure certificate after dropping either row.

Use axis separation in Definition 1233.3. The rows C and V remain distinct from the sibling audit packets, and the failed-axis row records whether the refusal is theory-side, verification-side, or axis-confusion data.

Apply the public axis export of Definition 1233.4. The export surface already names $N_0, C, V, A, B, D, H, K, P, L$ as the public rows and excludes private overrides, hidden verifier results, host equality, prose repair, and untyped failure reasons.

Finally read the checked TasteGate witness. Field faithfulness keeps $N_0, C, V, A, B, D, H, K, P, L$ as distinct carrier rows, while nontriviality prevents the bridge-facing packet from degenerating into a vacuous single-row object. Hence any bridge consumer must preserve exactly the listed interface. □

The Lean TasteGate file provides explicit $\mathsf{Nontrivial}\ \mathsf{FailureCertificate}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{FailureCertificate}^{\uparrow}$ instances in `lean4/BEDC/Derived/FailureCertificateUp/TasteGate.lean`. The field-faithfulness witness keeps $N_0, C, V, A, B, D, H, K, P, L$ distinct so the failed-axis row and blocked-gate row cannot be merged.

Lean4 ✓ `BEDC.Derived.FailureCertificateUp.taste gate`

Falsifiable prediction: Removing either the failed-axis row or blocked-gate row breaks the gate-blocking theorem for an accepted packet.

Independence witness: auditgatefailurewitness, realityconstrainedfailuresurface, registryexportconsistencygate, axiszeckendorfcannotclaim, machineinterfacesocket: none is bijective to $\mathsf{FailureCertificate}^{\uparrow}$ because the joint two-axis gate-blocking row A, B is preserved only by the finite failure-certificate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FailureCertificate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FailureCertificate}^{\uparrow}$ packet is generated from a named claim row, separate theory-closure and verification-status rows, a failed-axis row, a blocked-gate row, a diagnostic witness row, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FailureCertificate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1233.1, Definition 1233.2, Definition 1233.3, Definition 1233.4, Definition 1233.5, and the checked TasteGate witness bind gate blocking, axis separation, public axis export, the bridged interface, TasteGate evidence, and the sibling audit lattice.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FailureCertificate}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, private gate overrides, host equality, hidden verifier results, or untyped failure reasons.

Upgrade path: Verification closure requires checked Lean coverage for the mature failure-certificate carrier, gate-blocking surface, public export, bridged interface, TasteGate evidence, and sibling lattice.

Chapter 1234

A Concrete Naming Certificate for $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$

The $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$ packet realizes the writing-item discipline of Definition 9462.1984 as a finite transcript object. It sits between the machine consumer of chapter 1232, the registry coherence check of chapter 1223, and the proof-readback witness of chapter 1219. As a composite over those sibling audit surfaces, the packet bundles their report, registry, and proof-trace rows into one transcript envelope. Those siblings read reports, compare registry rows, and expose proof traces; this carrier names the per-claim envelope that keeps status, source, replay, gap, ledger, and refusal data together before any audit export is consumed.

Definition 1234.1 (Audit transcript envelope carrier)**.** *An* $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, C, L, S, G, A, R, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the claim-kind row, recording whether the writing item is read as a definition, theorem, corollary, proof, schema, report, roadmap, or refusal;*
2. *C is the carrier/classifier row, recording the BEDC carrier and the classifier or equality surface under which the item is read;*
3. *L is the ledger row, recording package provenance, dependency entries, and hidden-material refusals visible to the item;*
4. *S is the theory/formal-status pair, recording the theory-closure grade and formal-verification grade as two separate axes;*
5. *G is the gap-report row, recording open gaps, refused gaps, closed gaps, and the reason each gap is or is not consumable;*
6. *A is the source-anchor row, recording the manuscript label, file site, or registry entry from which the writing item is read;*
7. *R is the replay-checksum row, recording the displayed replay route by which the envelope is reconstructed;*
8. *U is the not-claimed surface, recording the strongest nearby readings that the envelope explicitly refuses;*

9. *H records componentwise* hsame *transport for claim kind, carrier/classifier, ledger, status-pair, gap, anchor, replay, and not-claimed rows;*
10. *T records displayed* Cont *replay from the source anchor through the checksum row to the machine-readable export boundary;*
11. *P is the* Pkg *provenance over* $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, RegistryConsistencyAud ProofTraceAuditWitness$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditTranscriptEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden proof object, unanchored prose claim, single-axis status collapse, off-ledger gap repair, unchecked replay substitute, or erased not-claimed surface.

Lean4 (definition) `BEDC.Derived.AuditTranscriptEnvelopeUp.AuditTranscriptEnvelopeUp`

Theorem 1234.2 (Audit transcript envelope field coverage)**.** *For the carrier of Definition 1234.1, the local* $\mathsf{NameCert}_{\mathsf{AuditTranscriptEnvelope}^{\uparrow}}$ *surface consists exactly of five obligation groups:*

1. *carrier admission for $E = (K, C, L, S, G, A, R, U, H, T, P, N)$;*
2. *writing-item exposure through the claim-kind row K and carrier/classifier row C;*
3. *audit-status exactness through the ledger row L, theory/formal-status pair S, and gap-report row G;*
4. *source and replay accountability through the source-anchor row A, replay-checksum row R, not-claimed row U, transport row H, and replay row T;*
5. *package provenance and local naming through P and N.*

Thus every auditable writing item consumed through this packet is read as one finite transcript envelope, not as a prose sentence with detachable audit metadata.

Proof. Project the carrier of Definition 1234.1. The rows $K, C, L, S, G, A, R, U, H, T, P, N$ are the only coordinates available to the local certificate.

First read the writing-item rows. The kind row K fixes the permitted claim type, and C fixes the carrier/classifier surface on which that claim is meaningful.

Next read the audit rows. The ledger row L, status pair S, and gap row G state what the claim depends on, which paper-side and verification-side grades it carries, and which gaps are exposed or refused.

Finally inspect A, R, U, H, T, P, N. These rows anchor the claim, replay the envelope, preserve the not-claimed surface, transport the displayed coordinates, record provenance, and name the packet. Since no further coordinate remains, the five obligation groups exhaust the local $\mathsf{NameCert}_{\mathsf{AuditTranscriptEnvelope}^{\uparrow}}$ surface. □

Theorem 1234.3 (Audit transcript envelope non-evidence refusal)**.** *If an* $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$ *carrier lacks either the theory/formal-status pair S or the replay-checksum row R, then no accepted* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *export may consume the envelope as theorem evidence.*

Proof. Begin with an accepted envelope

$$E = (K, C, L, S, G, A, R, U, H, T, P, N).$$

The machine-readable consumer of chapter 1232 reads audit data through displayed schema, checker, export, refusal, report, transport, replay, provenance, and local naming rows.

If S is absent, the consumer cannot distinguish paper-side closure from Lean-side verification status. If R is absent, the consumer has no checksum row tying the displayed transcript to a replay route.

In either case the requested theorem-evidence reading is stronger than the envelope supplies. The only available response is the non-evidence refusal recorded by the audit interface, so the envelope may still be reported but not consumed as theorem evidence. □

Theorem 1234.4 (Audit transcript envelope sibling independence)**.** *The* $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$*,* $\mathsf{ProofTraceAuditWitn}$ $\mathsf{RegistryConsistencyAudit}^{\uparrow}$*, or* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$*.*

Proof. Compare row inventories. $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ supplies schema, checker, export, refusal, and report rows for a machine consumer, but it does not store the per-claim source anchor, replay checksum, and not-claimed surface as one writing-item envelope.

$\mathsf{ProofTraceAuditWitness}^{\uparrow}$ binds a proof trace to source, classifier, transport, forbidden-resource refusal, verdict, provenance, and local naming. It does not carry the theory/formal-status pair or the gap report required for a general writing item.

$\mathsf{RegistryConsistencyAudit}^{\uparrow}$ compares registry entries and status agreement across exports, while $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ stores formal target skeletons and machine-facing target data. Neither is a per-claim transcript envelope with all of K, C, L, S, G, A, R, U displayed together.

Any NameCert-row-preserving bijection would have to preserve each of claim kind, carrier/classifier row, ledger row, status pair, gap report, source anchor, replay checksum, and not-claimed surface. Each named sibling omits at least one of those rows, so no such bijection exists. □

Theorem 1234.5 (Audit transcript envelope falsifiable prediction)**.** *An* $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$ *carrier is rejected as a machine-consumable audit transcript exactly when a consumer asks for a theorem-evidence, registry-consistency, or proof-readback consequence not backed by its displayed kind, carrier/classifier, ledger, status-pair, gap, source-anchor, replay-checksum, not-claimed, transport, replay, provenance, and local naming rows.*

Proof. Use Definition 1234.2 to read the complete envelope inventory. Every accepted read factors through $K, C, L, S, G, A, R, U, H, T, P, N$.

For theorem-evidence reads, inspect the status pair S and replay checksum R together with Definition 1234.3. For registry-consistency reads, inspect L, S, G, A and the provenance row P naming the registry audit surface. For proof-readback reads, inspect C, L, A and the provenance row naming the proof-trace witness.

If one of those rows is absent, disagrees with the requested consequence, or is refused by U, the consumer has left the envelope. Conversely a consequence backed by the displayed rows is precisely the finite audit transcript reading exposed by this carrier. □

Falsifiable prediction: If an envelope omits either the closure-status row or the replay checksum row, then it cannot be consumed by a MachineReadableAuditInterfaceUp export without a non-evidence refusal.

Independence witness: machinereadableauditinterface, prooftraceauditwitness, registryconsistencyaudit, machineinterfaceformalizationtarget: none is bijective to AuditTranscriptEnvelopeUp because each omits at least one of per-claim status-pair, source-anchor, replay-checksum, or not-claimed rows.

The Lean follow-up must provide explicit Nontrivial AuditTranscriptEnvelope$^{\uparrow}$ and FieldFaithful AuditTranscriptEn instances in `lean4/BEDC/Derived/AuditTranscriptEnvelopeUp/TasteGate.lean`. The field-faithfulness witness must preserve $K, C, L, S, G, A, R, U, H, T, P, N$ as separate displayed fields before BEDC.Derived.AuditTranscriptEnvelopeUp.taste_gate can be cited by a checked paper marker.

Lean4 ✓ `BEDC.Derived.AuditTranscriptEnvelopeUp.taste gate`

Closure status for NameCert$_{\text{AuditTranscriptEnvelope}^{\uparrow}}$.

Constructive story: An AuditTranscriptEnvelope$^{\uparrow}$ packet is generated from claim kind, carrier/classifier, ledger, theory/formal-status, gap, source-anchor, replay-checksum, not-claimed, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{AuditTranscriptEnvelope}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1234.1, Definition 1234.2, Definition 1234.3, Definition 1234.4, and Definition 1234.5 bind auditable writing items to finite transcript envelopes.

Formal verification: FormalStatus(NameCert$_{\text{AuditTranscriptEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, registry completeness, proof-trace correctness, machine export schema completeness, or global audit log finality.

Upgrade path: Public export is the next paper-side step; Lean verification still requires checked carrier admission, five-row NameCert coverage, non-evidence refusal, sibling independence, TasteGate nontriviality, and field faithfulness for $K, C, L, S, G, A, R, U, H, T, P, N$.

Chapter 1235

A Concrete Naming Certificate for WritingItemAudit$^{\uparrow}$

The WritingItemAudit$^{\uparrow}$ packet realizes the writing-item discipline of subsection 9462.30.1 as a concrete NameCert carrier. It turns kind, carrier, classifier, ledger, theory status, formal status, gap exposure, claim text, transport, continuation, provenance, and local naming into one finite object that registry and export chapters can read without identifying paper closure with machine verification.

Theorem 1235.1 (Writing item audit sibling dependency)**.** *Every accepted* WritingItemAudit$^{\uparrow}$ *packet consumes* AuditWritingDiscipline$^{\uparrow}$ *from chapter 1227 for legal writing discipline and* WritingItemAuditPacket$^{\uparrow}$ *from chapter 1222 for packet export shape. The former supplies the rule surface for claim legality, while the latter supplies the export packet read by registries.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit sibling dependency`

Proof. Read the opening carrier role of WritingItemAudit$^{\uparrow}$. Its item kind, claim text, status rows, and gap row are legal only against a writing-discipline surface, so the rule side is supplied by AuditWritingDiscipline$^{\uparrow}$.

The same packet must also be exported to registry and audit consumers. That outward shape is supplied by WritingItemAuditPacket$^{\uparrow}$, which records how the item audit is packaged for downstream readers.

Since these siblings supply different surfaces, the present carrier consumes both without identifying them: one governs legality of the writing item, and the other governs the registry-readable packet shape. □

Falsifiable prediction: If the theory-status row or formal-status row is removed, no accepted WritingItemAuditUp packet can distinguish paper closure from machine verification.

Independence witness: registryexportsurface, axiszeckendorfcannotclaim, physical-namecert-audit, focused-multi-hist-ledger, runtimecandidateboundary: registry export controls machine-facing export but not the eight-field item legality packet; cannot-claim rows refuse overstrong claims but do not carry both closure axes; physical NameCert audit checks naming surfaces without item kind and claim text; focused ledgers restrict histories but not writing-item status; runtime candidates record object-level execution boundaries without prose claim legality.

Definition 1235.2 (Writing item audit carrier)**.** *A* $\mathsf{WritingItemAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, C, R, L, T, F, G, Q, H, U, P, N)$$

with the following displayed rows:

1. *K is the item-kind row, naming whether the writing item is a definition, theorem, corollary, proof, reading, schema, roadmap, or refusal.*
2. *C is the carrier row on which the item lives.*
3. *R is the classifier row, recording the identity or comparison discipline used by the item.*
4. *L is the ledger row, exposing hidden material, refusal material, or the absence of such material.*
5. *T is the theory-closure status row, recording only the paper-side closure grade.*
6. *F is the formal-verification status row, recording only the Lean-side and audit-side verification grade.*
7. *G is the gap-report row, exposing open obligations, refusal boundaries, and non-closed scope.*
8. *Q is the claim-text row, carrying the strongest sentence the packet permits the reader to believe.*
9. *H records componentwise* hsame *transport for the kind, carrier, classifier, ledger, status, gap, claim, continuation, provenance, and naming rows.*
10. *U records displayed* Cont *continuations through which registry, export, closure-status, and refusal consumers read the packet.*
11. *P is the* Pkg *provenance over* $\mathsf{WritingItemAudit}^{\uparrow}$, BHist, hsame, Cont, *writing-item legality, closure-status blocks, registry export rows, and local naming rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}$ *row.*

The carrier exports only these rows. It has no coordinate for proving the claim text, upgrading a theory grade, upgrading a formal grade, filling a gap report, or licensing a stronger registry export.

Lean4 (definition) `BEDC.Derived.WritingItemAuditUp.WritingItemAuditUp`

Theorem 1235.3 (Writing item audit NameCert obligations)**.** *For the carrier of Definition 1235.2, the local* $\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for* $W = (K, C, R, L, T, F, G, Q, H, U,$ *item-kind admission through K, carrier and classifier exactness through C, R, ledger exposure through L, independent status exposure through T, F, gap exposure through G, claim-text binding through Q, componentwise* hsame *transport through H, displayed* Cont *replay through U,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit namecert obligations`

Proof. Project the accepted carrier. Its coordinates list the item kind, carrier, classifier, ledger, two status rows, gap report, claim text, transport, continuation, provenance, and local name.

Read the writing-item legality rules from Definition 9462.1984 and Definition 9462.1985. The rows K, C, R, L, T, F, G, Q are precisely the finite packet fields required there.

The remaining rows are structural. The transport row H preserves the displayed fields componentwise, the continuation row U exposes the allowed consumers, the provenance row P records the source surfaces, and the local row N names the packet. These rows exhaust the NameCert obligations. □

Theorem 1235.4 (Writing item audit axis separation)**.** *In every accepted* WritingItemAudit$^{\uparrow}$ *packet, the theory-closure status is read only from T and the formal-verification status is read only from F. A consumer may compare, report, or jointly display the two rows, but no accepted projection can derive either status row from the other.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit axis separation`

Proof. Begin with the carrier inventory of Definition 1235.2. The two status rows are separate coordinates: T records paper-side closure, while F records Lean-side and audit-side verification.

Apply Definition 1235.3. Its independent status-exposure clause forces both rows to be displayed before a closure-status or registry consumer can use the packet.

Since H, U, P, N transport, replay, source, and name the displayed rows without adding a computation from one status to the other, the packet permits joint reporting but not status derivation across axes. □

Theorem 1235.5 (Writing item audit non-escape)**.** *Every registry, export, closure-status, or refusal consumer of an accepted* WritingItemAudit$^{\uparrow}$ *packet factors through the finite route*

$$K, C, R, L, T, F, G, Q, H, U, P, N.$$

The consumer receives the item kind, carrier, classifier, ledger, theory status, formal status, gap report, claim text, and structural support rows. It receives no proof of the claim text, no theory-status upgrade, no formal-status upgrade, no hidden gap discharge, and no license to export a stronger claim.

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit nonescape`

Proof. Read the carrier coordinates. The only claim-bearing row is Q, and it is bound to the displayed kind, carrier, classifier, ledger, two status rows, and gap report.

Use Definition 1235.3. It requires the consumer route to expose $K, C, R, L, T, F, G, Q, H, U, P, N$ before the packet can serve as a legal writing item.

The excluded objects are not coordinates of the carrier. Because H, U, P, N only preserve, replay, source, and name the visible rows, no accepted projection can create a proof of Q, improve T or F, erase G, or authorize a claim stronger than the displayed text. □

Theorem 1235.6 (Writing item audit TasteGate obligation)**.** *The later Lean boundary* lean4/BEDC/Derived/Writin *must provide explicit* Nontrivial WritingItemAudit$^{\uparrow}$ *and* FieldFaithful WritingItemAudit$^{\uparrow}$ *instances over the same carrier. These instances must preserve $K, C, R, L, T, F, G, Q, H, U, P, N$ and must keep the theory-status row separate from the formal-status row.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit tastegate obligation`

Proof. Nontriviality is required because a writing-item audit packet is meaningful only when the kind, claim text, status rows, and gap report can be jointly inhabited as a finite carrier.

Field faithfulness is the corresponding projection discipline. It prevents a TasteGate consumer from identifying two packets while forgetting the item kind, carrier, classifier, ledger, either status row, the gap report, the claim text, or the structural route.

The same carrier must witness both instances. Otherwise a consumer could certify nonemptiness for one packet shape while checking field preservation for another, which would break the audit surface named by WritingItemAudit$^{\uparrow}$. □

Theorem 1235.7 (Writing item audit obligation closure route)**.** *The carrier, NameCert obligations, axis-separation route, nonescape boundary, and TasteGate obligation rows exhaust the* $\mathsf{WritingItemAudit}^{\uparrow}$ *obligation surface. Every accepted consumer must read the finite packet*

$$K, C, R, L, T, F, G, Q, H, U, P, N$$

together with the displayed $\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}$ *obligation theorem, and no consumer can obtain a proof of the claim text, a closure-grade upgrade, a formal-grade upgrade, or a registry export stronger than the claim row.*

Proof. Start with the carrier of Definition 1235.2. It lists the item kind, carrier, classifier, ledger, two independent status rows, gap report, claim text, transport, continuation, provenance, and local naming rows.

Apply Definition 1235.3 to expose the NameCert surface, then use Definition 1235.4 and Definition 1235.5 to keep the paper-closure row, verification row, gap row, and claim row from escaping their displayed coordinates.

Finally read Definition 1235.6. Its visible TasteGate requirement keeps the same carrier under nontriviality and field-faithfulness checks. The listed rows therefore exhaust the obligation surface without producing public closure, scoped closure, bridge checking, or verification beyond the existing Lean marker. □

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit obligation closure route`

Theorem 1235.8 (Writing item audit scoped kernel route)**.** *Every scoped consumer of an accepted* $\mathsf{WritingItemAudit}^{\uparrow}$ *packet must read*

$$K, C, R, L, T, F, G, Q, H, U, P, N$$

together with the writing-item legality definitions of Definition 9462.1984 and Definition 9462.1985, the axis-separation theorem of Definition 1235.4, the nonescape theorem of Definition 1235.5, and the TasteGate obligation of Definition 1235.6. No scoped projection derives T *from* F*, derives* F *from* T*, proves* Q*, discharges* G*, or exports a registry claim stronger than the displayed claim-text row.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit scoped kernel route`

Proof. Project the carrier of Definition 1235.2. The finite packet gives the item kind, carrier, classifier, ledger, paper-closure row, formal-verification row, gap row, claim row, transport, continuation, provenance, and local name as separate displayed coordinates.

Apply Definition 1235.3. It forces the scoped consumer to read the same carrier rows through the local $\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}$ surface and to keep the writing-item legality definitions visible before the packet is used.

Use Definition 1235.4 and Definition 1235.5. These keep T, F, G, and Q in their own coordinates and rule out proofs of the claim text, status upgrades, hidden gap discharge, and stronger registry exports.

Finally read Definition 1235.6. It attaches nontriviality and field-faithfulness to the same carrier rows, so the scoped route is a kernel-local reading discipline rather than a projection from one axis or claim row into another. □

Theorem 1235.9 (Writing item audit public axis export)**.** *Every registry or export consumer of an accepted* WritingItemAudit$^{\uparrow}$ *packet receives the paper-closure row T and the formal-verification row F as separate public audit fields. The export may display the two rows together with $K, C, R, L, G, Q, H, U, P, N$, but it cannot derive either audit row from the other, discharge the gap row G, prove the claim row Q, or publish a stronger status than the displayed rows carry.*

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit public axis export`

Proof. Begin with the scoped kernel route of Definition 1235.8. A public registry or export consumer must read the same finite packet $K, C, R, L, T, F, G, Q, H, U, P, N$, so both audit rows are visible before the consumer may publish the item.

Apply Definition 1235.4. It permits joint reporting of T and F while forbidding a projection that derives the paper-closure row from the formal-verification row or derives the formal-verification row from the paper-closure row.

Finally use Definition 1235.5. Since no accepted projection can prove Q, discharge G, upgrade either status row, or authorize a stronger registry claim, the public export surface consists exactly of the two displayed audit fields together with their carrier, ledger, claim, transport, replay, provenance, and naming rows. □

Theorem 1235.10 (Writing item audit public status)**.** *The public export of an accepted* WritingItemAudit$^{\uparrow}$ *packet is exactly the separated packet*

$$K, C, R, L, T, F, G, Q, H, U, P, N.$$

The rows T and F are read as independent public audit fields: T records paper closure, while F records formal verification. The export may cite the TasteGate obligation surface for nontriviality and field-faithfulness of the same carrier, but it does not derive either axis from the other, prove the claim row, discharge the gap row, or publish a stronger status than the displayed rows carry.

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit public status`

Proof. Start with Definition 1235.9. It already forces every registry or export consumer to read the separated rows $K, C, R, L, T, F, G, Q, H, U, P, N$ before publishing the item.

Use Definition 1235.4 to keep the paper-closure row and formal-verification row independent. The public packet can display them together, but no projection computes one from the other.

Finally apply Definition 1235.6. Its nontriviality and field-faithfulness requirements attach to the same displayed carrier rows, so the public status record is a public-facing packet surface rather than a verification upgrade or a proof of the claim text. □

Theorem 1235.11 (Writing item audit bridge-schema route)**.** *Every public export of an accepted* WritingItemAudit$^{\uparrow}$ *packet factors through* AuditWritingDiscipline$^{\uparrow}$ *and* WritingItemAuditPacket$^{\uparrow}$ *before any registry consumer reads the separated theory-status and formal-status rows. The bridge-schema route preserves the finite packet*

$$K, C, R, L, T, F, G, Q, H, U, P, N$$

as the writing-discipline carrier, packet-export shape, axis-separation surface, public status record, nonescape boundary, and TasteGate obligation surface.

Thus the registry-facing read may publish the displayed audit rows only after the sibling writing discipline and packet shape have exposed them. It cannot identify T with F, derive either row from the other, discharge G, prove Q, erase the claim-text boundary, or turn the public export into a Lean verification claim.

Lean4 ✓ `BEDC.Derived.WritingItemAuditUp.WritingItemAudit bridge schema route`

Proof. Start with the sibling dependency route at the beginning of the chapter. It forces every accepted $\mathsf{WritingItemAudit}^{\uparrow}$ packet to consume $\mathsf{AuditWritingDiscipline}^{\uparrow}$ for legal writing discipline and $\mathsf{WritingItemAuditPacket}^{\uparrow}$ for the outward packet shape before a registry reader sees the item.

Apply Definition 1235.10. The public status theorem identifies the registry-facing packet as $K, C, R, L, T, F, G, Q, H, U, P, N$, with T and F displayed as independent audit fields.

Use Definition 1235.4 and Definition 1235.5. These theorems keep the two status axes separated and prevent any public export from proving the claim row, discharging the gap row, or publishing a stronger registry status.

Finally apply Definition 1235.6. Its field-faithfulness condition attaches to the same carrier rows, so the bridge-schema route preserves the displayed packet through the writing-discipline and packet-export siblings without adding machine verification or a stronger claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{WritingItemAudit}^{\uparrow}$ packet is generated from item kind, carrier, classifier, ledger, theory-status, formal-status, gap-report, claim-text, hsame transport, Cont continuation, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1235.2, Definition 1235.3, Definition 1235.4, Definition 1235.5, Definition 1235.6, and Definition 1235.11 bind writing-item legality, axis separation, nonescape, the Lean-marked TasteGate obligation row, registry export rows, bridge-schema routing through $\mathsf{AuditWritingDiscipline}^{\uparrow}$ and $\mathsf{WritingItemAuditPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert to finite packets with separated closure and verification axes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WritingItemAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, proof of the claim text, grade upgrades, gap discharge, or registry export beyond the displayed claim row.

Upgrade path: Further progress is Lean verification and bridge checking; the paper-side mature packet remains independent of the formal-status axis.

Chapter 1236

A Concrete Naming Certificate for $\mathsf{HardProblemClosure}^{\uparrow}$

The $\mathsf{HardProblemClosure}^{\uparrow}$ packet realizes the hard-problem form of subsection 9462.27.8 as a finite NameCert carrier. It sits beside the focused multi-history ledger of chapter 1109, the consciousness-universe-space carrier of chapter 1206, the observer-history identity carrier of chapter 477, and the inter-Hist locality ledger of chapter 759. Those siblings provide focused records, a finite universe slice, observer identity rows, and local relation rows; this packet names the finite closure-question surface that keeps traditional hard-problem readings as inspectable boundary obligations.

Definition 1236.1 (Hard-problem closure carrier)**.** *A* $\mathsf{HardProblemClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, L, C, W, I, B, H, Q, P, N)$$

with the following displayed rows:

1. *R is the generated-record row, recording the finite histories, focused records, universe-slice records, observer-identity records, and local relation records available to the question;*
2. *L is the ledgered-local-choice row, recording the finite choices, selectors, focus windows, and refusal entries used by the read;*
3. *C is the compression-provenance row, recording which summaries, digests, or compressed presentations preserve their* Pkg *source;*
4. *W is the witnessed-relation row, recording the finite relation witnesses required before a consciousness, universe, observer, time, space, or identity reading is admitted;*
5. *I is the classifier-stable invariant row, recording exactly the invariants preserved by the displayed classifier and no others;*
6. *B is the boundary-pressure row, recording the gap and* T*-boundary readings that remain exposed rather than being internalised as substance;*
7. *H records componentwise* hsame *transport for $R, L, C, W, I, B, Q, P, N$;*
8. *Q records displayed* Cont *replay from generated records through local choices, compression provenance, relation witnesses, invariant readback, and boundary exposure;*
9. *P records* Pkg *provenance over* $\mathsf{HardProblemClosure}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{ConsciousUniverseSpace}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{HardProblemClosure}^{\uparrow}}$ *row.*
The carrier has no coordinate for a consciousness substance, complete universe object, detached observer, global time, ambient space substrate, host identity equality, hidden relation witness, or boundary-free closure of the displayed hard-problem readings.

Theorem 1236.2 (Hard-problem closure NameCert obligations)**.** *For the carrier of Definition 1236.1, the local* $\mathsf{NameCert}_{\mathsf{HardProblemClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (R, L, C, W, I, B, H, Q, P,$ *ledger coverage for* L*, compression-provenance accountability for* C*, relation-witness coverage for* W*, invariant classifier stability for* I*, boundary exposure for* B*, no-substance refusal for every reading not backed by* R, L, C, W, I, B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* Q*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.HardProblemClosureUp.HardProblemClosureTasteGate single carrier alignment`

Proof. Project the carrier. The displayed rows $R, L, C, W, I, B, H, Q, P, N$ are the full inventory available to a hard-problem closure reader.

Read the evidence rows in order. Generated records, local choices, compression provenance, witnessed relations, classifier-stable invariants, and boundary pressure are exposed before transport, replay, provenance, or local naming can consume the packet.

The sibling surfaces supply only parts of this inventory. Focused multi-history ledgers provide record and ledger rows, consciousness-universe-space packets provide universe, time, space, and trace rows, observer-history identity packets provide identity comparison rows, and inter-Hist locality ledgers provide relation routes. None supplies a substance coordinate or a hidden hard-problem solution row.

Finally use H, Q, P, N. These rows transport, replay, source, and name the same packet. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1236.3 (Hard-problem closure boundary exposure)**.** *Every consciousness, universe, observer, time, space, or identity reading of an accepted* $\mathsf{HardProblemClosure}^{\uparrow}$ *packet remains a boundary row unless the displayed packet supplies generated records* R*, ledgered local choices* L*, compression provenance* C*, witnessed relations* W*, classifier-stable invariants* I*, and boundary-pressure audit rows* B *for that reading. A route preserving only the structural rows* H, Q, P, N *cannot turn the reading into consciousness substance, complete universe object, detached observer, global time, ambient space substrate, or host identity equality.*

Lean4 ✓ `BEDC.Derived.HardProblemClosureUp.HardProblemClosure boundary exposure`

Proof. Start with the boundary-pressure row B. It records which traditional reading is still exposed as gap or T pressure, and it is read only after the packet displays the generated records, choices, provenance, witnesses, and invariants that support the local question.

If those rows are displayed, the reading remains an accepted closure-question carrier rather than a substance theorem. The rows R, L, C, W, I supply finite records-side evidence, while B keeps the pressure visible as a boundary obligation.

If any required evidence row is absent, the NameCert obligation theorem routes the attempted read to no-substance refusal. The structural rows H, Q, P, N can preserve and replay a displayed packet, but they cannot create a hidden witness, detached observer, global time, ambient space, or host identity coordinate. □

Falsifiable prediction: A purported HardProblemClosureUp packet that omits either the witnessed-relation row or the boundary-pressure row is not an accepted closure-question carrier.

Independence witness: focused-multi-hist-ledger, consciousuniversespace, observerhistoryidentity, inter-hist-locality-ledger: each sibling supplies one surface of focused records, universe slice, observer identity, or locality relation, but none is bijective to the full records-ledger-witness-invariant-boundary packet.

The Lean follow-up for this chapter must provide explicit Nontrivial HardProblemClosure$^{\uparrow}$ and FieldFaithful HardProblemClosure$^{\uparrow}$ instances in `lean4/BEDC/Derived/HardProblemClosureUp/TasteGate.lean`. The field-faithfulness witness must preserve $R, L, C, W, I, B, H, Q, P, N$ as distinct rows before BEDC.Derived.HardProblemClosureUp.taste_gate can be cited by a checked paper marker.

Closure status for NameCert$_{\text{HardProblemClosure}\uparrow}$.

Constructive story: The hard-problem closure object is generated from a $BHist$ packet that records focused records, a ledger row, conscious-universe row, witnessed relation, invariant row, and boundary-pressure row. The carrier convention is deliberately local: it reads the problem-facing closure evidence, transports it through $hsame$, and lets Cont carry the boundary forward only as the same named row. The NameCert obligations and boundary-exposure theorem do not solve or dissolve the problem; they certify the visible closure surface available in the ledger. This makes the object a bottom-up boundary certificate whose meaning is fixed by recorded rows rather than by an external philosophical conclusion.

Theory closure: TheoryGate(NameCert$_{\text{HardProblemClosure}\uparrow}$, seed)

Scope closed: This chapter binds the hard-problem closure carrier, NameCert obligations, boundary-exposure theorem, falsifiable prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{HardProblemClosure}\uparrow}$, unformalized)

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.hardProblemClosureDeltaLedger

Bridge status: none

Origin: ai

Not claimed: It does not claim a solution to the hard problem, subjective identity, physical reduction, or closure outside the displayed boundary row.

Upgrade path: Reaching obligationClosure requires formal closure and boundary rows plus checked preservation of boundary exposure under $hsame$.

Chapter 1237

A Concrete Naming Certificate for $\mathsf{TypedFailureCertificate}^{\uparrow}$

The $\mathsf{TypedFailureCertificate}^{\uparrow}$ packet realizes the failure-certificate discipline of subsection 9462.30.9 inside the auditable-generativity programme of chapter 9462. It sits beside the failed-gate witness of chapter 901, the registry export surface of chapter 1216, the cannot-claim packet of chapter 554, and the standard-bridge audit packet of chapter 1217. Those siblings expose local gate failure, registry blocking, axis-specific refusal, and bridge-audit rows; this packet names the typed failure itself before any consumer may treat a gap, failure, or bridge refusal as public audit data.

Definition 1237.1 (Typed failure certificate carrier)**.** *A* $\mathsf{TypedFailureCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (N, C, V, S, R, D, H, P, L)$$

with the following displayed rows:

1. *N is the object-name row for the object, chapter, theorem site, bridge site, or registry entry whose claimed status is being tested;*
2. *C is the theory-grade row, recording the claimed paper-side closure grade and the theory-side gate that fails when the branch is theory-facing;*
3. *V is the formal-status row, recording the claimed verification grade and the verification-side gate that fails when the branch is checker-facing;*
4. *S is the failed-site row, recording the exact source, classifier, exactness, descent, gap, scope, bridge, host-leak, checker, audit, or axis site at which the claimed gate is blocked;*
5. *R is the refusal-route row, recording the gate, registry, bridge, or export route that consumes the failure as refusal data;*
6. *D is the diagnostic-branch row, recording whether the failure is theory-side, verification-side, or axis-confusion data, together with the typed branch named by the failure-certificate discipline;*
7. *H records componentwise* hsame *transport for object name, theory grade, formal status, failed site, refusal route, and diagnostic branch;*
8. *P is the* Pkg *provenance over* $\mathsf{TypedFailureCertificate}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. L is the local $\mathsf{NameCert}_{\mathsf{TypedFailureCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a failed gate without a diagnostic branch, a verification failure read as theory closure, a theory failure read as verification status, a bridge refusal detached from its route, or an export permission that bypasses the displayed refusal data.

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateTasteGate single carrier alignment`

Theorem 1237.2 (Typed failure certificate NameCert obligations)**.** *For the carrier of Definition 1237.1, the local* $\mathsf{NameCert}_{\mathsf{TypedFailureCertificate}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for $F = (N, C, V, S, R, D, H, P, L)$;*
2. *object-and-status readback through the object name N, theory-grade row C, and formal-status row V;*
3. *failure localization through the failed-site row S and diagnostic-branch row D;*
4. *refusal consumption through the route row R and componentwise transport row H;*
5. *package provenance and local naming through P and L.*

Thus every accepted failure certificate is a finite NameCert packet over the named object, the two independent status axes, the failed site, the refusal route, the diagnostic branch, transport, provenance, and local naming.

Proof. Project the displayed carrier of Definition 1237.1. The rows $N, C, V, S, R, D, H, P, L$ are the only coordinates available to a local certificate.

First read the object and status rows. The object row N fixes the site being tested, while C and V keep the theory-grade and formal-status claims as separate rows.

Next read the failure rows. The failed-site row S locates the gate that did not pass, and the diagnostic branch D states whether the failure is theory-side, verification-side, or axis-confusion data.

Finally read R, H, P, L. The refusal route consumes the failure, transport preserves the same displayed coordinates, provenance lists the sibling surfaces through which the certificate may be consumed, and the local name seals the packet. Since no further coordinate remains, the five obligation rows exhaust the local $\mathsf{NameCert}_{\mathsf{TypedFailureCertificate}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateNameCertObligations`

Theorem 1237.3 (Typed failure certificate axis separation)**.** *For every accepted* $\mathsf{TypedFailureCertificate}^{\uparrow}$ *carrier*

$$F = (N, C, V, S, R, D, H, P, L),$$

the theory-side failures, verification-side failures, and axis-confusion failures remain separated by the diagnostic branch D. A consumer may use a theory-side branch to block the theory gate named by C, a verification-side branch to block the formal-status gate named by V, and an axis-confusion branch to block the verified-gate pairing, but no branch may be read as a failure of the other axis.

Proof. Use Definition 1237.2. The second obligation row exposes C and V separately, and the third obligation row requires the diagnostic branch D before a failed site can be consumed.

If D is theory-side, the refused content is read through the theory-grade row C and the failed site S. The formal-status row V remains displayed but does not become the failed gate.

If D is verification-side, the refused content is read through V and S. The theory-grade row C remains displayed but does not become the failed gate.

If D is axis-confusion data, the failure is the attempted collapse of the two rows. The branch blocks the verified-gate pairing while preserving the separate theory and verification rows for downstream audit.

The transport, provenance, and local-name rows H, P, L preserve the same packet. They cannot replace D with a branch-free reading, so the three failure classes remain separated. □

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateAxisSeparation`

Theorem 1237.4 (Typed failure certificate export blocking)**.** *No* RegistryExportSurface$^{\uparrow}$, AxisZeckendorfCannotCla AuditGateFailureWitness$^{\uparrow}$, *or* StandardBridgeAuditPacket$^{\uparrow}$ *consumer may export a public route past a displayed* TypedFailureCertificate$^{\uparrow}$ *failure unless it consumes both the diagnostic branch D and the refusal route R of*

$$F = (N, C, V, S, R, D, H, P, L).$$

An export route that reads only N, C, V, S, H, P, L is not an accepted export of the failed claim, failed status, or failed bridge.

Proof. Begin with the obligation surface of Definition 1237.2. Failure localization uses S together with D, and refusal consumption uses R together with the transport row H.

For registry export, the sibling surface of chapter 1216 requires the blocking and audit rows before a claim may leave the registry. A typed failure certificate supplies those data only through D and R.

For cannot-claim, audit-gate failure, and standard-bridge reads, the sibling chapters chapter 554, chapter 901, and chapter 1217 consume refusal rows rather than hidden prose. The displayed diagnostic branch tells which gate failed, and the displayed route tells how the refusal is replayed.

If either D or R is omitted, the consumer still sees an object and status pair but cannot tell which typed branch failed or how the refusal is consumed. Such a route is outside the accepted TypedFailureCertificate$^{\uparrow}$ carrier and therefore cannot export the failed claim, failed status, or failed bridge as public BEDC audit data. □

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateExportBlocking`

Theorem 1237.5 (Typed failure certificate bridge-refusal route)**.** *Any* StandardBridgeAuditPacket$^{\uparrow}$ *consumer of a* TypedFailureCertificate$^{\uparrow}$ *failure*

$$F = (N, C, V, S, R, D, H, P, L)$$

must read the object-name row N, theory-grade row C, formal-status row V, failed-site row S, refusal-route row R, diagnostic-branch row D, transport row H, provenance row P, local NameCert row L, and the TasteGate-aligned packet before treating the bridge as refused. This route blocks axis confusion: the refusal may close the paper-side bridge route, but it cannot be read as bridgeChecked *verification.*

Proof. Start with the export-blocking theorem Definition 1237.4. A standard-bridge audit consumer cannot pass a failed bridge unless it consumes the diagnostic branch D and refusal route R.

Use the NameCert obligations of Definition 1237.2. Those obligations expose the remaining rows N, C, V, S, H, P, L as part of the same finite certificate, so the consumer cannot detach the bridge refusal from the object, status, failed site, transport, provenance, or local naming data.

Finally apply the axis-separation theorem Definition 1237.3. The diagnostic branch identifies the failed route without collapsing the theory-grade row C into the formal-status row V. Therefore the standard-bridge audit reads a paper-side refusal route for the displayed packet, not a checked Lean bridge result. □

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateBridgeRefusalRoute`

Theorem 1237.6 (Typed failure certificate mature treatment route)**.** *For every accepted* $\mathsf{TypedFailureCertificate}^{\uparrow}$ *carrier*

$$F = (N, C, V, S, R, D, H, P, L),$$

the NameCert obligation surface, axis-separation surface, export-blocking surface, and bridge-refusal surface jointly determine the mature paper-side treatment of the certificate. Every public consumer must read the object name, theory-grade row, formal-status row, failed-site row, refusal-route row, diagnostic branch, componentwise transport, provenance, and local NameCert row before it may treat the failure as audit data, and the route never converts paper-side bridge refusal into bridgeChecked *verification.*

Proof. Start with the NameCert obligations of Definition 1237.2. They expose exactly the finite rows $N, C, V, S, R, D, H, P, L$ and make them the only local carrier surface available to a consumer.

Use axis separation, Definition 1237.3. The diagnostic branch D keeps theory-side, verification-side, and axis-confusion failures apart, so a refused theory gate cannot be read as a failed formal-status gate and a checker-facing failure cannot be read as paper closure.

Next apply export blocking, Definition 1237.4. A registry, cannot-claim, audit-gate, or bridge-audit consumer cannot export the failed claim unless it consumes both D and the refusal route R.

Finally apply the bridge-refusal route of Definition 1237.5. The standard-bridge audit reads the same finite failure packet as a paper-side refusal route. Since the route preserves C and V as separate rows, it closes the mature paper treatment while leaving the Lean verification axis unchanged. □

Lean4 ✓ `BEDC.Derived.TypedFailureCertificateUp.TypedFailureCertificateMatureTreatmentRoute`

Falsifiable prediction: If the diagnostic branch is deleted, a failed theory-grade row and a failed formal-status row become indistinguishable to public consumers.

Independence witness: auditgatefailurewitness, registryexportsurface, axiszeckendorfcannot-claim, standardbridgeauditpacket: AuditGateFailureWitnessUp records a failed gate witness without the full theory-grade, formal-status, and axis-confusion branch surface; RegistryExportSurfaceUp records export blocking without naming every typed failure branch; AxisZeckendorfCannotClaimUp records cannot-claim refusal without checker-facing failure status; StandardBridgeAuditPacketUp records bridge fields and no-host-leak rows without the full failure-certificate branch taxonomy. None is bijective to $\mathsf{TypedFailureCertificate}^{\uparrow}$.

The Lean follow-up must provide explicit Nontrivial $\mathsf{TypedFailureCertificate}^{\uparrow}$ and FieldFaithful TypedFailureCertific instances in `lean4/BEDC/Derived/TypedFailureCertificateUp/TasteGate.lean`. The field-faithfulness witness must preserve $N, C, V, S, R, D, H, P, L$ as distinct rows before BEDC.Derived.TypedFailureCertific can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{TypedFailureCertificate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TypedFailureCertificate}^{\uparrow}$ packet is generated from object-name, theory-grade, formal-status, failed-site, refusal-route, diagnostic-branch, componentwise transport, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedFailureCertificate}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1237.1, Definition 1237.2, Definition 1237.3, Definition 1237.4, Definition 1237.5, Definition 1237.6, and BEDC.Derived.TypedFailureCertificateUp.taste_gate bind the mature paper surface: carrier admission, NameCert obligations, axis separation, export blocking, bridge-refusal routing, and the cited TasteGate marker for the same finite failure-certificate packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{TypedFailureCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim checked Lean code, public export closure, public bridge closure, standard bridge checking, or a completed TasteGate carrier for TypedFailureCertificate$^{\uparrow}$.

Upgrade path: Lean-side carrier, Nontrivial, FieldFaithful, and bridge-checking work remain separate from the mature paper closure grade.

Chapter 1238

A Concrete Naming Certificate for $\mathsf{ProofWitnessChain}^{\uparrow}$

The $\mathsf{ProofWitnessChain}^{\uparrow}$ packet realizes the BEDC proof form of subsection 9462.25.1 as a concrete finite NameCert carrier. It sits between the proof-trace audit packet of chapter 1219, the accepted-declaration audit surface of chapter 546, the certificate-preserving route of chapter 481, and the proof-object erasure surface of chapter 642. Those siblings audit narrower faces of proof use; this carrier binds the proof trace, source carrier, classifier witness, transport, ledger, refusal, audit verdict, provenance, and local name into one finite witness-chain object.

Definition 1238.1 (Proof witness chain carrier)**.** *A* $\mathsf{ProofWitnessChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, S, K, H, L, F, A, P, N)$$

with the following displayed rows:

1. *T is the proof-trace row, recording the finite witness chain consumed by the proof site.*
2. *S is the source-carrier row, recording the BEDC carrier, source object, theorem surface, or certificate row whose proof evidence is being read.*
3. *K is the classifier-witness row, recording the classifier membership, comparison, stability, or exactness witness licensing the chain.*
4. *H is the displayed-transport row, recording the* hsame *transport needed across classifier, package, bridge, or history-identity surfaces.*
5. *L is the proof ledger row, recording* Cont *replay, package route, local ledger entries, and proof-site handoff rows.*
6. *F is the forbidden-resource refusal row, recording refusal of hidden choice, host equality as BEDC equality, quotient collapse, propositional extensional rewrite, global truth predicate, unledgered compression, and unscoped universal claim.*
7. *A is the audit-verdict row, recording accepted, refused, or displayed-obligation status for the whole witness chain.*
8. *P is the* Pkg *provenance over* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$, $\mathsf{KernelAuditWitness}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, $\mathsf{ProofTermErasureTrace}^{\uparrow}$, $\mathsf{FiniteWitnessRoute}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofWitnessChain}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an untraced proof, implicit classifier crossing, hidden proof source, erased refusal, host equality proof, or audit verdict detached from provenance and local naming.

Lean4 (definition) `BEDC.Derived.ProofWitnessChainUp.ProofWitnessChainUp`

Theorem 1238.2 (Proof witness chain NameCert obligations)**.** *For the carrier of Definition 1238.1, the local* $\mathsf{NameCert}_{\mathsf{ProofWitnessChain}^{\uparrow}}$ *surface consists exactly of the following obligation rows: carrier admission for* $W = (T, S, K, H, L, F, A, P, N)$*; proof-trace and source-carrier exposure through* T *and* S*; classifier-witness stability through* K*; displayed transport through* H*; ledger replay and proof-site handoff through* L*; forbidden-resource refusal and audit-verdict exactness through* F *and* A*; and provenance plus local naming through* P *and* N*.*

Lean4 ✓ `BEDC.Derived.ProofWitnessChainUp.ProofWitnessChainTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1238.1. The rows $T, S, K, H, L, F, A, P, N$ are the only displayed coordinates of an accepted proof witness chain, so the local NameCert first reads a finite proof trace, its source surface, and the classifier witness that licenses the trace.

Next read the transport and ledger rows. The displayed transport row H keeps classifier crossing inside hsame data, while the ledger row L records replay, package route, local ledger entries, and handoff rows in Cont, Pkg, and $\mathsf{NameCert}$ form.

Finally attach the refusal, verdict, provenance, and local name rows. The refusal row F records which proof resources are unavailable, the verdict row A classifies the whole chain, and P, N bind that verdict to finite provenance and local naming. Repacking these rows gives the listed obligations and leaves no coordinate for hidden proof evidence. □

Theorem 1238.3 (Proof witness chain ledger non-escape)**.** *Every accepted* $\mathsf{ProofWitnessChain}^{\uparrow}$ *proof read factors through the displayed rows*

$$T, S, K, H, L, F, A, P, N.$$

Deleting the classifier-witness row K *or the forbidden-resource refusal row* F *invalidates the accepted proof witness chain carrier.*

Proof. Begin with Definition 1238.2. The accepted chain must expose proof trace, source carrier, classifier witness, displayed transport, ledger handoff, forbidden-resource refusal, audit verdict, provenance, and local naming as one finite packet.

If K is absent, the trace has no displayed classifier comparison or stability witness. The source row S and trace row T remain visible, but no NameCert obligation licenses them as a BEDC proof read.

If F is absent, the packet no longer records refusal of hidden choice, quotient collapse, host equality, propositional extensional rewrite, global truth, unledgered compression, or unscoped universal claim. The verdict row A then has no refusal boundary to audit.

Thus every accepted proof read remains inside the displayed packet. A consumer that reads proof evidence without K or F has left the $\mathsf{ProofWitnessChain}^{\uparrow}$ carrier rather than extracting a hidden proof resource from it. □

Theorem 1238.4 (Proof witness chain proof-trace handoff)**.** *Every accepted* $\mathsf{ProofWitnessChain}^{\uparrow}$ *carrier hands its trace row* T *to chapter 1219 before a downstream consumer may read the proof as BEDC-internal evidence. The handoff preserves the source row* S*, classifier row* K*, transport row* H*, ledger row* L*, refusal row* F*, audit verdict* A*, provenance row* P*, and local name* N*.*

Proof. Read the proof-trace component T of Definition 1238.1. The trace is not a standalone proof object; it is the row sent to the proof-trace audit witness surface.

The surrounding rows S, K, H, L, F, A, P, N provide the exact context in which the trace is consumed. The source row and classifier witness determine what the trace is evidence for, transport and ledger rows determine how it is replayed, and the refusal and verdict rows determine whether it is accepted or refused.

The handoff therefore preserves the full packet boundary. A downstream consumer may route through $\mathsf{ProofTraceAuditWitness}^{\uparrow}$, $\mathsf{KernelAuditWitness}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, or $\mathsf{ProofTermErasureTrace}^{\uparrow}$, but it may not replace the witness chain by any one of those siblings alone. □

The Lean follow-up file `lean4/BEDC/Derived/ProofWitnessChainUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ ProofWitnessChain}^{\uparrow}$ and $\mathsf{FieldFaithful\ ProofWitnessChain}^{\uparrow}$ instances, preserving $T, S, K, H, L, F, A, P, N$ as separate displayed fields.

Falsifiable prediction: Deleting the classifier-witness or forbidden-resource refusal row invalidates the accepted proof witness chain carrier.

Independence witness: prooftraceauditwitness, kernelauditwitness, namecertmorphism, prooftermerasuretrace, finitewitnessroute: none is bijective to $\mathsf{ProofWitnessChain}^{\uparrow}$ because preserving the carrier requires proof trace, source, classifier, transport, ledger, refusal, audit verdict, provenance, and local name together. ProofTraceAuditWitnessUp audits the trace face, KernelAuditWitnessUp audits accepted declarations, NameCertMorphismUp transports certificate rows, ProofTermErasureTraceUp separates erased proof rows, and FiniteWitnessRouteUp records finite route exits; no listed sibling preserves the whole proof witness chain surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofWitnessChain}^{\uparrow}}$.

Constructive story: The proof witness chain is generated from $BHist$ rows that list proof trace data, source rows, classifier witnesses, transport, ledger entries, forbidden-resource refusal, audit verdict, provenance, and local naming in order. The carrier first reads each witness row, then checks that the ledger connects adjacent rows as a chain, and finally allows Cont to carry the trace forward without replacing it by an external proof object. The $hsame$ relation keeps the witness sequence stable when histories are represented differently. The NameCert obligations, ledger non-escape, proof-trace handoff, prediction, and independence witness thus make the chain a visible certificate of derivation evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofWitnessChain}^{\uparrow}}, \mathsf{obligation})$

Scope closed: This chapter binds the proof witness-chain carrier, NameCert obligations, ledger non-escape theorem, proof-trace handoff theorem, falsifiable prediction, and independence witness as the visible obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofWitnessChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim proof validity outside the witness chain, proof irrelevance, or derivations not recorded in the ledger.

Upgrade path: Reaching obligationClosure requires formal witness-chain rows and checked handoff preservation for the visible proof trace.

Chapter 1239

A Concrete Naming Certificate for $\mathsf{TruthResidueLedger}^{\uparrow}$

The $\mathsf{TruthResidueLedger}^{\uparrow}$ packet realizes the scientific-truth synthesis of chapter 9462 as audited stability rather than final correspondence. It sits beside the scientific-object carrier of chapter 1214, the reality-constrained truth certificate of chapter 1144, the methodology ledger of chapter 1145, the open-fit packet of chapter 1137, and the physical truth certificate of chapter 1141. Those siblings expose object rows, truth-certificate rows, methodology rows, open-fit rows, and physical-truth rows; the residue ledger binds their non-final scientific truth route into one finite NameCert packet.

Falsifiable prediction: If a TruthResidueLedgerUp packet is accepted, no consumer can infer final model truth from model fit without passing through the prediction-gap and falsification rows; a direct route from fit to global truth refutes the packet.

Independence witness: scientificobject, realityconstrainedtruthcert, realityconstrainedmethodologyledger, openfitpacket, physicaltruthcertificate: none is bijective because $\mathsf{TruthResidueLedger}^{\uparrow}$ binds observation records, invariant rows, conservative bridge rows, prediction-gap rows, falsification rows, non-finality rows, transport, replay, provenance, and local NameCert rows in one carrier. ScientificObjectUp names object-facing record structure; RealityConstrainedTruthCertUp names a truth-certificate route; RealityConstrainedMethodologyLedgerUp names methodology export scope; OpenFitPacketUp names finite fit and open defeat; PhysicalTruthCertificateUp names physical-truth export. None carries the complete residue ledger that keeps fit, stability, prediction, falsification, and non-finality jointly visible.

Definition 1239.1 (Truth residue ledger carrier)**.** *A* $\mathsf{TruthResidueLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (O, I, B, P, F, G, H, C, Q, N)$$

with the following displayed rows:

1. *O is the observation-record row, naming the finite records and measurement traces from which the scientific-truth read begins.*
2. *I is the invariant row, recording the stable inter-history or classifier-preserved content that survives the accepted comparison route.*
3. *B is the conservative model-bridge row, recording only the bridge by which the finite records are compared with model-side structure.*

4. *P is the prediction-gap row, exposing commitments not discharged by the present fit, including later-observation and domain-exit obligations.*
5. *F is the falsification row, recording finite defeats, failed predictions, classifier mismatches, and open tests that can refute the export.*
6. *G is the non-finality row, refusing a direct route from finite fit or stable invariant to global model truth.*
7. *H records componentwise* hsame *transport for O, I, B, P, F, G and the local naming row.*
8. *C records displayed* Cont *replay from observation admission through invariant read, bridge read, prediction-gap read, falsification read, non-finality refusal, provenance, and local naming.*
9. *Q is the* Pkg *provenance over* $\mathsf{TruthResidueLedger}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$, $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{PhysicalTruthCertificate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TruthResidueLedger}^{\uparrow}}$ *row.*

The carrier exports only these rows. It has no coordinate for final correspondence, model–world identity, hidden completeness, unledgered fit, erased prediction debt, or a falsification-free truth route.

Lean4 (definition) `BEDC.Derived.TruthResidueLedgerUp.TruthResidueLedgerUp`

Theorem 1239.2 (Truth residue ledger NameCert obligations)**.** *For the carrier of Definition 1239.1, the local* $\mathsf{NameCert}_{\mathsf{TruthResidueLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (O, I, B, P, F, G, H, C, Q, N$ observation exposure through O, invariant exposure through I, conservative bridge exposure through B, prediction-gap exposure through P, falsification exposure through F, non-finality exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Proof. Project the accepted carrier. Its visible coordinates are the observation records, invariants, conservative model bridge, prediction gaps, falsification surface, non-finality refusal, transport, replay, provenance, and local name.

The sibling chapters supply the surrounding surfaces: $\mathsf{ScientificObject}^{\uparrow}$ supplies object-facing records, $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ supplies truth-certificate discipline, $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$ supplies methodology scope, $\mathsf{OpenFitPacket}^{\uparrow}$ supplies finite fit with open defeat, and $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ supplies physical-truth export constraints. The present packet does not identify any one sibling with scientific truth; it records their residue-bearing handoff.

The structural rows H, C, Q, N preserve, replay, source, and name only the displayed packet. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.TruthResidueLedgerUp.TruthResidueLedger namecert obligations`

Theorem 1239.3 (Truth residue ledger falsifiable boundary)**.** *Every accepted* $\mathsf{TruthResidueLedger}^{\uparrow}$ *consumer that reads model fit or invariant stability as scientific truth must pass through both the prediction-gap row P and the falsification row F before the local name N is readable. A route from B or I directly to final model truth is not an accepted* $\mathsf{TruthResidueLedger}^{\uparrow}$ *route.*

Lean4 ✓ `BEDC.Derived.TruthResidueLedgerUp.TruthResidueLedger falsifiable boundary`

Proof. Start from the carrier of Definition 1239.1. The bridge row B and invariant row I are distinct from the prediction-gap row P, the falsification row F, and the non-finality row G.

Apply Definition 1239.2. The local NameCert surface exposes P, F, G as obligations before the accepted scientific-truth read reaches the local name. Thus fit and stability are admitted only with their open prediction and defeat surfaces still displayed.

The excluded direct route has no coordinate in the carrier. Transport, replay, and provenance preserve the same finite packet, so they cannot create final correspondence or erase the falsifiable boundary. □

Theorem 1239.4 (Truth residue ledger field coverage)**.** *Every scientific-truth export from an accepted* $\mathsf{TruthResidueLedger}^{\uparrow}$ *packet factors through*

$$O, I, B, P, F, G, H, C, Q, N.$$

The consumer receives finite observation records, stable invariant rows, conservative bridge rows, prediction-gap rows, falsification rows, non-finality rows, componentwise transport, displayed replay, package provenance, and local NameCert naming. It receives no final truth oracle, no unbounded correspondence claim, no hidden model identity, no erased prediction debt, and no falsification-free route.

Lean4 ✓ `BEDC.Derived.TruthResidueLedgerUp.TruthResidueLedger field coverage`

Proof. Use Definition 1239.2 to identify the complete local certificate surface. Every accepted consumer must expose the six content rows and the four structural rows listed there.

Then apply Definition 1239.3. Any consumer attempting to read fit or stability as scientific truth must still carry prediction-gap and falsification data, so the route cannot collapse to final model truth.

The sibling dependencies remain sources rather than replacements. $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{RealityConstrainedTruthCer}$ $\mathsf{RealityConstrainedMethodologyLedger}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, and $\mathsf{PhysicalTruthCertificate}^{\uparrow}$ contribute rows through Q, but none supplies a hidden coordinate outside $O, I, B, P, F, G, H, C, Q, N$. Therefore the displayed route covers every accepted field of the truth-residue ledger. □

The Lean TasteGate file `lean4/BEDC/Derived/TruthResidueLedgerUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{TruthResidueLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{TruthResidueLedger}^{\uparrow}$ instances over the same carrier. The instances must preserve $O, I, B, P, F, G, H, C, Q, N$ and must keep conservative fit, prediction-gap, falsification, and non-finality rows as separate fields.

Lean4 ✓ `BEDC.Derived.TruthResidueLedgerUp.TruthResidueLedgerTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{TruthResidueLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{TruthResidueLedger}^{\uparrow}$ packet is generated from observation records, invariant rows, conservative model-bridge rows, prediction-gap rows, falsification rows, non-finality rows, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TruthResidueLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1239.1, Definition 1239.2, Definition 1239.3, and Definition 1239.4 close the sibling lattice and mature falsifiable field-coverage route over $O, I, B, P, F, G, H, C, Q, N$ for scientific truth as audited stability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TruthResidueLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, final correspondence, model–world identity, ambient completeness, or falsification-free truth.

Upgrade path: Lean closure requires checked Nontrivial TruthResidueLedgerUp and FieldFaithful TruthResidueLedgerUp coverage for the enumerated carrier, obligation, falsifiable-boundary, and field-coverage rows.

Chapter 1240

A Concrete Naming Certificate for $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$

The $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ packet realizes the seven-socket boundary taxonomy of subsection 9462.26.14 as finite NameCert data. It sits beside the classifier packet of chapter 1058, the induction boundary socket of chapter 498, the host-delegation socket of chapter 551, the apophatic far-end socket of chapter 1061, and the finite trace socket of chapter 504. Those siblings classify a single socket, expose one induction boundary, audit host delegation, bind a far-end boundary, or record finite trace pressure; this packet stores the whole seven-kind carrier together with its gates, non-collapse matrix, and local naming rows.

Falsifiable prediction: No accepted Type III socket packet can be reclassified as a Type II witness-visibility socket without losing the identity-collapse gate row.

Independence witness: SocketKindClassifier, TranscendentalInductionSocket, HostDelegationSocket, ApophaticFarEndSocket, FiniteTraceGapSocket: none is bijective to $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$. The classifier names a kind row without carrying the whole seven-kind non-collapse matrix; the induction socket exposes Type I pressure without the remaining gates; the host-delegation socket records delegated target resolution; the apophatic far-end socket binds boundary and inscription rows; the finite trace socket records bounded trace pressure. The taxonomy carrier alone keeps the seven-kind carrier, gate-row taxonomy, and non-collapse matrix in one NameCert packet.

Definition 1240.1 (Typed gap-socket taxonomy carrier)**.** *A* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, G, M, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the seven-kind carrier row, listing Type I generator-closure sockets, Type II witness-visibility sockets, Type III identity-collapse sockets, Type IV causal-continuity sockets, Type V cross-history coordination sockets, Type VI inscription-actuality sockets, and Type VII reflection-stratification sockets.*

2. *G is the gate taxonomy row, assigning generator closure, witness visibility, extensional-collapse refusal, record-prediction separation, locality discipline, aspect separation, and reflection stratification to the seven socket kinds.*

3. *M is the non-collapse matrix, recording for each ordered pair of distinct socket kinds which gate row would be lost by a reclassification.*

4. *L is the local classifier ledger, recording how a packet may cite* $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{TranscendentalInductionSock}$ $\mathsf{HostDelegationSocket}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, *and* $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ *without replacing the taxonomy packet by any one sibling.*
5. *H records componentwise* hsame *transport for K, G, M, L and the local naming row.*
6. *C records displayed* Cont *replay through kind admission, gate lookup, non-collapse lookup, sibling-ledger lookup, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{TranscendentalInductionSocket}^{\uparrow}$, $\mathsf{HostDelegationSocket}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedGapSocketTaxonomy}^{\uparrow}}$ *row.*

The carrier exports no untyped socket, no hidden universal supply, no witness selector, no quotient equality, no host target resolution, no far-end internalizer, and no collapse of the seven gates into one generic boundary condition.

Lean4 ✓ `BEDC.Derived.TypedGapSocketTaxonomyUp.TypedGapSocketTaxonomyTasteGate single carrier alignment`

Theorem 1240.2 (Typed gap-socket taxonomy NameCert obligations)**.** *For the carrier of Definition 1240.1, the local* $\mathsf{NameCert}_{\mathsf{TypedGapSocketTaxonomy}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (K, G, M, L, H, C, P, N)$, seven-kind exposure through K, gate taxonomy exposure through G, non-collapse matrix exposure through M, sibling-ledger exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TypedGapSocketTaxonomyUp.TypedGapSocketTaxonomyCarrier semantic name certificate`

Proof. Project the accepted carrier. Its displayed rows are the seven-kind carrier, the gate taxonomy, the non-collapse matrix, the sibling ledger, transport, replay, provenance, and local naming.

Read the sibling rows only through the ledger L. $\mathsf{SocketKindClassifier}^{\uparrow}$ supplies classifier discipline, $\mathsf{TranscendentalInductionSocket}^{\uparrow}$ supplies induction-boundary pressure, $\mathsf{HostDelegationSocket}^{\uparrow}$ supplies host-target delegation discipline, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ supplies far-end boundary discipline, and $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ supplies finite trace pressure. None of these rows contains the whole seven-kind matrix.

The rows H, C, P, N preserve and name the same packet. Since the carrier has no further coordinate, every local NameCert obligation is one of the rows listed in the theorem. □

Theorem 1240.3 (Typed gap-socket taxonomy public export)**.** *For every accepted* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ *packet, the public interface is exactly the seven-kind carrier K, the gate taxonomy G, the non-collapse matrix M, the sibling ledger L, the TasteGate marker for the packet, the semantic certificate named by the local* $\mathsf{NameCert}_{\mathsf{TypedGapSocketTaxonomy}^{\uparrow}}$ *row, and the noncollapse row of Definition 1240.4. No public read exports an untyped socket, a hidden witness selector, a host target-resolution row, a far-end internalizer, a quotient equality, or a collapsed generic boundary condition.*

Lean4 ✓ `BEDC.Derived.TypedGapSocketTaxonomyUp.TypedGapSocketTaxonomyPublicExport`

Proof. Project the accepted carrier through Definition 1240.1. The rows K, G, M, L, H, C, P, N are the displayed public data, and Definition 1240.2 identifies the semantic certificate as the same local NameCert surface.

The TasteGate marker reads the same packet rather than adding a second classifier. The noncollapse prediction is the exported row that makes the seven-kind surface public: it records that changing a socket kind loses a named gate and matrix entry.

Since the carrier has no coordinate for an untyped socket, hidden selector, host target resolution, far-end internalizer, quotient equality, or generic boundary condition, every public read factors through the listed rows. □

Theorem 1240.4 (Typed gap-socket taxonomy noncollapse prediction)**.** *In every accepted* TypedGapSocketTaxono *packet, a Type III identity-collapse socket cannot be reclassified as a Type II witness-visibility socket while preserving the local NameCert surface. Any such reclassification loses the Type III gate row in G and the corresponding non-collapse entry in M, so it is a failed taxonomy read rather than an accepted Type II packet.*

Lean4 ✓ `BEDC.Derived.TypedGapSocketTaxonomyUp.TypedGapSocketTaxonomyNoncollapsePrediction`

Proof. Start with Definition 1240.2. The accepted packet exposes both the kind row K and the gate row G before the local name is readable.

For a Type III socket, the gate row is extensional-collapse refusal: a relation may support displayed transport, but it is not identity. For a Type II socket, the gate row is witness visibility: an element or selecting function must be displayed. These are distinct coordinates of G.

Now use the non-collapse matrix M. Its Type III-to-Type II entry records that replacing the identity-collapse gate by witness visibility erases exactly the row that blocks relation-as-equality. Therefore the reclassified packet no longer preserves K, G, M, L, H, C, P, N as one NameCert surface. □

The Lean follow-up file `lean4/BEDC/Derived/TypedGapSocketTaxonomyUp/TasteGate.lean` must provide explicit Nontrivial TypedGapSocketTaxonomy$^{\uparrow}$ and FieldFaithful TypedGapSocketTaxonomy$^{\uparrow}$ instances. These instances must preserve the rows K, G, M, L, H, C, P, N and must keep the seven-kind carrier, gate taxonomy, non-collapse matrix, and sibling ledger as separate fields.

Closure status for NameCert$_{\text{TypedGapSocketTaxonomy}^{\uparrow}}$.

Constructive story: A TypedGapSocketTaxonomy$^{\uparrow}$ packet is generated from a seven-kind socket carrier, gate taxonomy, non-collapse matrix, sibling ledger, componentwise transports, displayed replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{TypedGapSocketTaxonomy}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1240.1, Definition 1240.2, Definition 1240.3, and Definition 1240.4 expose the seven-kind carrier, gate taxonomy, non-collapse matrix, sibling ledger, TasteGate marker, semantic certificate, and noncollapse theorem as one finite taxonomy packet.

Formal verification: FormalStatus(NameCert$_{\text{TypedGapSocketTaxonomy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not supply missing witnesses, quotient equality, host target truth, far-end internalization, bridge checking, or Lean verification for TypedGapSocketTaxonomy$^{\uparrow}$.

Upgrade path: Lean closure requires Nontrivial TypedGapSocketTaxonomyUp and FieldFaithful TypedGapSocketTaxonomyUp instances in `lean4/BEDC/Derived/TypedGapSocketTaxonomyUp/TasteGate.lean`.

Chapter 1241

A Concrete Naming Certificate for $\mathsf{HistoryPrefix}^{\uparrow}$

The $\mathsf{HistoryPrefix}^{\uparrow}$ packet realizes the history-prefix surface of chapter 9462 as a finite NameCert carrier. It sits before the countable-name interface of chapter 126, the regular-rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. Those siblings consume finite observations, regular readbacks, and completed seals; this packet isolates the prefix-residual observation that lets a finite BHist prefix feed those consumers without becoming a host stream, quotient sequence, hidden tail oracle, or ambient time primitive.

Falsifiable prediction: If a HistoryPrefixUp packet is accepted, dropping the residual-tail row destroys the Cont reconstruction route to StreamNameUp finite observations.

Independence witness: streamname, regseqrat, real, dyadicratcore: HistoryPrefixUp carries a prefix-residual Cont obligation none of these siblings carries as a primitive row, so no bijection preserves the NameCert ledger.

Definition 1241.1 (History-prefix NameCert carrier)**.** *A* $\mathsf{HistoryPrefix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (S, p, r, C, H, E, L, N)$$

with the following displayed rows:

1. *S is the source* BHist *row whose generated history is being inspected;*
2. *p is the finite prefix row retained as the public observation;*
3. *r is the residual-tail row recording the finite distinction trace by which the source history is reconstructed from the prefix;*
4. *C is the displayed* Cont *witness routing p through r back to S and forward to finite* $\mathsf{StreamName}^{\uparrow}$ *observations;*
5. *H records componentwise* hsame *compatibility for the source, prefix, residual tail, continuation route, evidence, ledger, and local name;*
6. *E is the* Pkg *evidence row over* $\mathsf{HistoryPrefix}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*
7. *L is the NameCert ledger row recording prefix admission, residual-tail admission, continuation replay, sibling handoff, and refusal of hidden tail data;*

8. N is the local $\mathsf{NameCert}_{\mathsf{HistoryPrefix}^\uparrow}$ *row.*

The carrier exports a finite prefix observation together with the residual route that justifies it. It exports no host stream, global clock, quotient sequence, chosen limit, unbounded observation oracle, or residual tail detached from the displayed source history.

Lean4 ✓ `BEDC.Derived.HistoryPrefixUp.HistoryPrefixTasteGate single carrier alignment`

Definition 1241.2 (History-prefix classifier)**.** *For accepted* $\mathsf{HistoryPrefix}^\uparrow$ *carriers* $P = (S, p, r, C, H, E, L, N)$ *and* $P' = (S', p', r', C', H', E', L', N')$*, the classifier* $\mathsf{HistoryPrefixClassifier}(P, P')$ *holds exactly when the displayed prefix rows* p, p' *are related by the recorded* hsame *comparison, the residual-tail rows* r, r' *reconstruct compatible source histories through the displayed* Cont *witnesses* C, C'*, and the sibling handoff ledgers* L, L' *expose the same finite observation route to* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*, and* $\mathsf{Real}^\uparrow$ *consumers.*

Thus the classifier reads only the source BHist *rows, prefix rows, residual-tail rows, continuation witnesses,* hsame *compatibility rows, package evidence, NameCert ledgers, and local names. It does not compare host streams, external time indices, quotient sequences, selected real limits, or unrecorded tails.*

Theorem 1241.3 (History-prefix NameCert obligations)**.** *For the carrier of Definition 1241.1, the local* $\mathsf{NameCert}_{\mathsf{HistoryPrefix}^\uparrow}$ *surface consists exactly of carrier admission for* $P = (S, p, r, C, H, E, L, N)$*, source-history exposure through* S*, finite-prefix exposure through* p*, residual-tail exposure through* r*,* Cont *reconstruction and sibling handoff through* C*, componentwise* hsame *compatibility through* H*,* Pkg *evidence through* E*, ledger exactness through* L*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.HistoryPrefixUp.HistoryPrefixCarrier namecert obligations`

Proof. Project the carrier P. Its coordinates are precisely the source BHist row, finite prefix, residual tail, continuation route, compatibility row, package evidence, ledger, and local name.

Read the reconstruction route next. The residual-tail row r is not an ambient hidden tail; it is the displayed finite trace that lets C reconstruct the source from the prefix and then hand the prefix observation to the finite StreamName-facing route.

Finally attach compatibility, evidence, ledger, and naming. The row H transports only the displayed coordinates, E records the source of the packet, L records admission and sibling handoff, and N names the local certificate. Repacking these rows gives the listed obligations and leaves no coordinate for a host stream or unrecorded tail. □

Theorem 1241.4 (History-prefix continuation stability)**.** *For accepted* $\mathsf{HistoryPrefix}^\uparrow$ *carriers, the classifier of Definition 1241.2 is stable under displayed* Cont *replay and componentwise* hsame *transport of the source, prefix, residual-tail, package-evidence, ledger, and local-name rows. Every downstream* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*, or* $\mathsf{Real}^\uparrow$ *consumer sees the same finite prefix route only through*

$$S, p, r, C, H, E, L, N.$$

Deleting r *destroys the reconstruction route from the accepted prefix packet to the StreamName-facing finite observation.*

Proof. Start with the classifier definition. It compares accepted packets only through the displayed source, prefix, residual-tail, continuation, compatibility, evidence, ledger, and local-name rows.

Now replay the continuation route. Since C records both reconstruction from p through r to S and handoff from p to finite observation consumers, componentwise hsame transport can preserve the classifier only by transporting r together with the prefix and source rows.

If the residual-tail row is removed, the prefix row remains visible but the reconstruction witness no longer has the displayed trace needed to reach the source history. A downstream StreamName-facing read therefore lacks the route required by the accepted carrier rather than discovering an implicit tail outside the NameCert ledger. □

Lean4 ✓ `BEDC.Derived.HistoryPrefixUp.HistoryPrefixCarrier cont replay transport`

The Lean follow-up file `lean4/BEDC/Derived/HistoryPrefixUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{HistoryPrefix}^{\uparrow}$ and FieldFaithful $\mathsf{HistoryPrefix}^{\uparrow}$ instances, preserving S, p, r, C, H, E, L, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{HistoryPrefix}^{\uparrow}}$**.**

Constructive story: The history-prefix certificate is generated from a $BHist$ row that names an initial segment, a residual-tail row, and a classifier that recognizes the segment inside later histories. The carrier first reads the prefix, then checks the classifier row, and finally uses Cont to describe reconstruction and extensions that append information without changing the certified initial part. The $hsame$ relation transports the prefix certificate across equivalent presentations, so the same visible segment and tail route are preserved under representative change. The NameCert obligations and continuation-stability theorem together state that a prefix is a ledger object: once certified, it remains the same prefix for every admitted continuation.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HistoryPrefix}^{\uparrow}}$, seed)

Scope closed: This chapter binds the history-prefix carrier, classifier, NameCert obligations, continuation-stability theorem, and seed closure marker.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HistoryPrefix}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim equality of whole histories, unique future continuation, or prefix preservation outside the displayed continuation relation.

Upgrade path: Reaching obligationClosure requires formal prefix rows and checked preservation of the classifier under $hsame$ and Cont.

Chapter 1242

A Concrete Naming Certificate for $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$

The $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ packet realizes the diagonal-regularity theorem row of Definition 9238.4 as a finite NameCert seal for the Real-and-Completion route of chapter 9238. It sits after the diagonal-limit packet of chapter 374, the representative route of chapter 1205, the three-leg estimate of chapter 466, the dyadic tolerance ledger of chapter 1211, the regularity witness of chapter 1158, and the terminal Real surface of chapter 38. The packet is a finite consumer seal: it does not introduce quotient streams, selected host limits, host real equality, choice rows, or ambient completeness.

Definition 1242.1 (Real diagonal-regularity seal carrier)**.** *A* $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, T, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted* $\mathsf{DiagonalLimit}^{\uparrow}$ *or diagonal representative source row whose regularity is being sealed.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window bundle exposing both endpoint observations and the intermediate observations used by the diagonal triangle.*
3. *T is the accepted* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *row for the same endpoint and intermediate observations.*
4. *Q is the accepted* $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *row carrying the endpoint tolerances, middle Cauchy-modulus tolerance, dyadic sum, and output bound.*
5. *R is the accepted* $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *row that records the regularity witness consumed by the diagonal-regularity theorem.*
6. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *handoff row, compatible with the exact Real-completion boundary of chapter 1120.*
7. *H records componentwise* hsame *transport for D, W, T, Q, R, E and the local certificate row.*
8. *C records displayed* Cont *replay from the diagonal source through the window bundle, triangle-bound row, dyadic ledger, regularity witness, and Real handoff.*

9. *P is the* Pkg *provenance over* RealDiagonalRegularitySeal$^{\uparrow}$, DiagonalLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyTriangleBound$^{\uparrow}$, DyadicToleranceTriangleLedger$^{\uparrow}$, RegularCauchyRegularityWitness Real$^{\uparrow}$, RealCompletionExactBoundaryUp, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{RealDiagonalRegularitySeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RealDiagonalRegularitySealUp.RealDiagonalRegularitySealTasteGate single carrier alignment`

Definition 1242.2 (Real diagonal-regularity seal classifier)**.** *Two accepted* RealDiagonalRegularitySeal$^{\uparrow}$ *carriers classify together only when their diagonal source rows, window bundles, triangle-bound rows, dyadic ledgers, regularity-witness rows, Real handoff rows, transport rows, continuation rows, package rows, and local naming rows are related componentwise by the displayed* hsame *data and replayed along the same* Cont *route. The classifier does not identify packets by host equality of real numbers, by quotienting streams, by forgetting the middle Cauchy-modulus row, or by replacing the dyadic tolerance row with an ambient completeness appeal.*

Theorem 1242.3 (Real diagonal-regularity seal obligations)**.** *For every accepted* RealDiagonalRegularitySeal$^{\uparrow}$ *carrier $S = (D, W, T, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{RealDiagonalRegularitySeal}^{\uparrow}}$ *surface consists exactly of carrier exposure, classifier transport, diagonal-source exposure through D, endpoint and intermediate window coverage through W, three-leg estimate coverage through T, dyadic tolerance-sum coverage through Q, regularity-witness coverage through R, Real handoff coverage through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Carrier exposure is projection from Definition 1242.1. The displayed tuple lists the diagonal source, window bundle, triangle-bound row, dyadic ledger, regularity witness, Real handoff, transport, replay, provenance, and local naming rows.

Classifier transport is exactly the relation of Definition 1242.2. It keeps the source row, every window row, the triangle route, the dyadic ledger, the regularity witness, the Real handoff, and the support rows on one componentwise transported surface.

The data-bearing obligations are then read from the cited sibling chapters. The diagonal source comes from chapter 374; the estimate comes from chapter 466; the tolerance ledger comes from chapter 1211; the regularity witness comes from chapter 1158; and the terminal handoff uses chapter 38.

Since H, C, P, N only transport, replay, source, and name those displayed rows, the listed clauses exhaust the local seal surface. □

Theorem 1242.4 (Real diagonal-regularity seal triangle route)**.** *Every* Real$^{\uparrow}$ *or completion-facing consumer of an accepted* RealDiagonalRegularitySeal$^{\uparrow}$ *packet reads the ordered finite route*

$$D, W, T, Q, R, E, H, C, P, N$$

before using the diagonal-regularity result as a Real-facing seal. In particular, the consumer receives the endpoint regularity rows, the middle Cauchy-modulus row, the dyadic tolerance-sum row, and the triangle-route handoff as displayed packet coordinates.

Proof. Begin with the obligation theorem. It requires the diagonal source and finite window bundle to be visible before the estimate, ledger, regularity witness, or Real handoff can be consumed.

Next read the triangle route. The $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ row T supplies the three-leg estimate, while the $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ row Q records the endpoint tolerances, the middle Cauchy-modulus tolerance, and the displayed dyadic sum used by the theorem row.

Finally route the result through R and E. The $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ row packages the regularity evidence, and the $\mathsf{Real}^{\uparrow}$ handoff row is reachable only through the same H, C, P, N support. Hence the consumer receives exactly the ordered finite route shown above. □

Theorem 1242.5 (Real diagonal-regularity seal nonescape)**.** *An accepted* $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ *carrier exports only the displayed rows* $D, W, T, Q, R, E, H, C, P, N$ *and their classifier transports. No consumer can replace the endpoint regularity rows, middle Cauchy-modulus row, dyadic tolerance-sum row, or triangle-route handoff by quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or an ambient completeness theorem.*

Proof. Inspect the carrier coordinates of Definition 1242.1. Each coordinate is a displayed BEDC row or structural support for transport, continuation replay, provenance, and local naming.

Definition 1242.3 accounts for the admissible carrier and classifier rows. Definition 1242.4 accounts for every Real-facing or completion-facing consumer route.

The excluded objects require coordinates absent from the carrier. Therefore neither the local NameCert row nor any displayed continuation can export them in place of the finite diagonal-regularity seal. □

Falsifiable prediction: A diagonal-regularity consumer that reads a RealUp completion seal without the displayed endpoint regularity rows, middle Cauchy-modulus row, dyadic tolerance-sum row, and triangle-route handoff refutes the packet.

Independence witness: diagonallimit, regularcauchytrianglebound, dyadic-tolerance-triangle-ledger, regular-cauchy-regularity-witness, realcompletionexactboundary: none is bijective to RealDiagonalRegularity because the diagonal-limit packet lacks the full theorem seal, the triangle-bound packet lacks the terminal Real handoff, the dyadic ledger lacks the selected diagonal source seal, the regularity witness lacks the exact boundary handoff, and the exact boundary packet lacks the diagonal-regularity theorem row as its own classifier surface.

Theorem 1242.6 (Real diagonal-regularity seal TasteGate obligation closure)**.** *The carrier, classifier, triangle route, nonescape row, falsifiable prediction, independence witness, Lean* ChapterTasteGate*,* Nontrivial*, and* FieldFaithful *witnesses jointly close the paper-side obligation surface for* $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ *over the displayed rows* $D, W, T, Q, R, E, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.RealDiagonalRegularitySealUp.taste gate`

Proof. The carrier and classifier obligations are the displayed finite surface of Definition 1242.1 and Definition 1242.2. They expose exactly the diagonal source, finite window bundle, triangle-bound row, dyadic ledger, regularity witness, Real handoff, transport, continuation, package, and local naming rows.

The route obligations are supplied by Definition 1242.3, Definition 1242.4, and Definition 1242.5. Together they bind the admissible carrier clauses, force every Real-facing consumer through the ordered route, and rule out quotient streams, selected host limits, host real equality, countable choice, detached completion evidence, and ambient completeness.

The falsifiable prediction and independence witness give the observable failure condition and the sibling-separation condition for the same rows. The Lean-side ChapterTasteGate, Nontrivial, and FieldFaithful witnesses in `BEDC.Derived.RealDiagonalRegularitySealUp.TasteGate` certify

that this obligation is machine-backed by one carrier with the listed fields. Hence the paper-side obligation surface is closed at the stated row list. □

Theorem 1242.7 (Real diagonal-regularity seal scoped consumer route)**.** *An accepted Real-facing consumer of* $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ *is scoped exactly when it exposes the ten finite rows*

$$D, W, T, Q, R, E, H, C, P, N$$

as primitive BEDC support and consumes them in the displayed order. Here D *is the diagonal source,* W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite window,* T *is the* $\mathsf{RegularCauchyTriangleBound}^{\uparrow}$ *row,* Q *is the* $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$ *row,* R *is the* $\mathsf{RegularCauchyRegularityWitness}^{\uparrow}$ *row,* E *is the* $\mathsf{Real}^{\uparrow}$ *handoff, and* H, C, P, N *are respectively the* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. The scoped route has no coordinate for quotient streams, selected host limits, host real equality, countable choice, detached completion evidence, or ambient completeness.*

Proof. For the forward direction, begin with an accepted scoped consumer. The carrier of Definition 1242.1 supplies exactly the ten rows $D, W, T, Q, R, E, H, C, P, N$, and the classifier of Definition 1242.2 keeps those rows componentwise transported along the same hsame and Cont support.

The obligation theorem then identifies the primitive support used by the consumer. It reads D as the diagonal source, W as the finite window, T as the triangle-bound row, Q as the dyadic ledger, R as the regularity witness, E as the Real handoff, and H, C, P, N as the transport, replay, provenance, and local naming rows.

The downstream direction is forced by Definition 1242.4. A Real-facing consumer cannot enter the theorem row without the ordered route through the triangle estimate, tolerance ledger, regularity witness, and Real handoff, so its scope is exactly the displayed ten-row route.

Conversely, suppose a consumer exposes and consumes precisely those ten rows as primitive BEDC support. The carrier and classifier definitions supply the admissible packet surface, the obligation theorem supplies the NameCert clauses, and the triangle-route theorem supplies the Real-facing consumption order.

It remains to exclude hidden support. The nonescape theorem rules out quotient streams, selected host limits, host real equality, countable choice, detached completion evidence, and ambient completeness, while Definition 1242.6 records the TasteGate-backed closure of the same finite row list. Hence the exposed route is scoped, and no extra ambient witness can change its scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalRegularitySeal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealDiagonalRegularitySeal}^{\uparrow}$ packet is generated from a diagonal source row, finite window bundle, RegularCauchyTriangleBound row, DyadicToleranceTriangleLedger row, RegularCauchyRegularityWitness row, RealUp handoff row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealDiagonalRegularitySeal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1242.1, Definition 1242.2, Definition 1242.3, Definition 1242.4, Definition 1242.5, Definition 1242.6, and Definition 1242.7 bind the finite diagonal-regularity seal surface and its exact Real-facing consumer scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealDiagonalRegularitySeal}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.RealDiagonalRegularitySealUp.taste\_gate`

Bridge status: none

Not claimed: This chapter does not close theorem-level Lean verification, bridge checking, quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or ambient completeness.

Upgrade path: Theorem-level Lean verification and bridge checking require a checked scoped-route theorem over the same ten primitive rows and downstream Real-facing consumer route.

Chapter 1243

A Concrete Naming Certificate for $\mathsf{TypedSupplySocket}^{\uparrow}$

The $\mathsf{TypedSupplySocket}^{\uparrow}$ packet realizes the seven T-supply socket kinds of subsection 9462.26.10 as a finite NameCert carrier. It sits beside the kind classifier of chapter 1058, the supply ledger of chapter 528, the machine-facing route of chapter 1059, and the apophatic far-end socket of chapter 1061. Those siblings classify kinds, account for requested supply, route machine targets, and name the non-internalizing far end; this packet binds the kind, request, consumption site, audit gate, refusal row, transports, continuations, provenance, and local NameCert into one typed socket certificate.

Definition 1243.1 (Typed supply socket carrier)**.** *A* $\mathsf{TypedSupplySocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, Q, S, G, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the socket-kind row, naming one of the seven T-supply socket kinds from subsection 9462.26.10;*
2. *Q is the requested-supply row, recording the precise shortcut requested by the socket;*
3. *S is the consumption-site row, recording where a downstream construction would consume the requested supply;*
4. *G is the audit-gate row, recording the gate that accepts, refuses, or blocks the requested supply at that site;*
5. *U is the non-internalization row, recording that accepted socket traffic remains a visible boundary request rather than an internal substrate theorem;*
6. *H records componentwise* hsame *transport for K, Q, S, G, U and the local naming row;*
7. *C records displayed* Cont *routes from the kind and requested-supply rows through the consumption site, audit gate, non-internalization row, provenance, and local name;*
8. *P is the* Pkg *provenance over* $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedSupplySocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for erasing the socket kind, consuming requested supply without a site, bypassing the audit gate, turning the non-internalization row into a supplied witness, or identifying all socket kinds with one untyped boundary.

Theorem 1243.2 (Typed supply socket NameCert obligations)**.** *For the carrier of Definition 1243.1, the local* $\mathsf{NameCert}_{\mathsf{TypedSupplySocket}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$T = (K, Q, S, G, U, H, C, P, N),$$

socket-kind exactness through K*, requested-supply exposure through* Q*, consumption-site exposure through* S*, audit-gate readback through* G*, non-internalization through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.TypedSupplySocketCarrier semantic name certificate`

Proof. Project the displayed carrier. The rows $K, Q, S, G, U, H, C, P, N$ are the only coordinates available to the local certificate, so the certificate cannot introduce a hidden socket kind, supply request, consumption site, gate, refusal row, transport, replay route, provenance source, or local name.

Read the content rows in order. The kind row K fixes the seven-kind classifier position, Q records the requested supply, S records the site that would consume it, G records the audit gate for that site, and U records the non-internalization boundary.

Finally inspect H, C, P, N. These rows only transport, replay, source, and name the same finite packet. Thus every NameCert obligation is exhausted by the displayed typed-supply socket rows. □

Theorem 1243.3 (Typed supply socket kind separation)**.** *Every accepted* $\mathsf{TypedSupplySocket}^{\uparrow}$ *packet separates Type I induction sockets from Type II witness sockets by the public kind row* K *together with the requested-supply row* Q *and consumption-site row* S*. If the row* K *is erased, the public classifier can no longer determine whether the consumer is asking for unrestricted generator closure or for a missing positive witness.*

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.TypedSupplySocketCarrier kind row public separation`

Proof. Use Definition 1243.1. The row K is the only displayed row whose role is to name the socket kind, while Q and S record the requested supply and its consumption site.

For a Type I read, Q asks for unrestricted induction or universal closure beyond the displayed authorized generator, and S is the construction site where that closure would be consumed. For a Type II read, Q asks for a positive existential witness, and S is the construction site where the witness would be consumed.

The audit row G and non-internalization row U preserve the distinction as a refusal-facing boundary; they do not reconstruct the kind after K has been removed. Therefore kind separation is public exactly while K remains part of the packet. □

Theorem 1243.4 (Typed supply socket continuation kind transport)**.** *For every accepted* $\mathsf{TypedSupplySocket}^{\uparrow}$ *packet*

$$T = (K, Q, S, G, U, H, C, P, N),$$

any displayed Cont *replay that preserves the requested-supply row* Q *and the consumption-site row* S *must also keep the socket-kind row* K *visible before the audit-gate row* G *is consumed. Otherwise Type I induction sockets and Type II witness sockets collapse at the consumer boundary, since the same requested supply and site rows would no longer determine which kind of boundary request is being audited.*

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.TypedSupplySocketCarrier continuation kind transport`

Proof. Project the carrier of Definition 1243.1. The continuation row C replays the same finite rows K, Q, S, G, U, H, P, N rather than building an untyped supply route.

Use Definition 1243.3. The kind row K is the public separator between Type I induction sockets and Type II witness sockets, while Q and S only say what is requested and where it would be consumed.

Thus a replay that preserves Q and S but hides K before G is read cannot present the gate with a typed socket boundary. Reading C as replay of the same finite rows forces K to remain visible until the audit-gate row is consumed. □

Definition 1243.5 (Gate-visible typed supply socket continuation)**.** *The gate-visible continuation subcarrier of* $\mathsf{TypedSupplySocket}^{\uparrow}$ *consists of accepted packets*

$$T = (K, Q, S, G, U, H, C, P, N)$$

whose displayed Cont *row C keeps the socket-kind row K, requested-supply row Q, consumption-site row S, audit-gate row G, and non-internalization row U visible until the audit gate is consumed. A read in this subcarrier may transport the packet by H and replay it through C, but it has no coordinate for hiding G, converting Q into an untyped grant, or replacing U by a witness internal to the consuming construction.*

Theorem 1243.6 (Typed supply socket audit-gate nonescape)**.** *Every accepted gate-visible continuation read of Definition 1243.5 factors through the socket-kind row K, requested-supply row Q, consumption-site row S, audit-gate row G, non-internalization row U, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. It cannot replace the audit gate by an untyped supply grant or by a hidden witness.*

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.TypedSupplySocketCarrier audit gate nonescape`

Proof. First use Definition 1243.3. The row K is the public separator for the socket kind, while Q and S identify the requested supply and the site at which it would be consumed.

Next apply Definition 1243.4. A continuation that preserves Q and S keeps K visible before G is consumed. The definition of the gate-visible subcarrier strengthens the same replay by requiring G and U to remain displayed as well.

Finally read the TasteGate obligation of Definition 1243.7. Its checked marker records that the accepted carrier preserves $K, Q, S, G, U, H, C, P, N$ as distinct rows. Therefore the replay has no route that turns the audit gate into an untyped supply grant or hides a witness inside the consumer. □

Theorem 1243.7 (Typed supply socket TasteGate obligation)**.** *The* $\mathsf{TypedSupplySocket}^{\uparrow}$ *TasteGate obligation is the demand that any admitted instance preserve all rows*

$$K, Q, S, G, U, H, C, P, N$$

as distinct fields. In particular, no TasteGate route may identify the socket-kind row with the requested-supply row, hide the consumption site, bypass the audit gate, internalize the non-internalization row, or replace the displayed hsame, Cont, Pkg, *and* NameCert *support by an external classifier.*

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.TypedSupplySocketTasteGate single carrier alignment`

Proof. Start from the NameCert obligations of Definition 1243.2. They make carrier admission, kind exactness, requested-supply exposure, consumption-site exposure, audit-gate readback, non-internalization, transport, replay, provenance, and local naming separate certificate rows.

Apply the kind-separation theorem. If a TasteGate route erases or identifies K, Type I induction sockets and Type II witness sockets can no longer be separated by the public classifier. If it erases Q or S, the route no longer states what is requested or where it is consumed.

Finally read G, U, H, C, P, N. The audit gate and non-internalization row keep the request boundary-visible, while transport, replay, provenance, and local naming preserve the same finite packet. A TasteGate instance that fails field preservation therefore leaves the accepted $\mathsf{TypedSupplySocket}^{\uparrow}$ carrier. □

Theorem 1243.8 (Typed supply socket obligation rows)**.** *Every accepted* $\mathsf{TypedSupplySocket}^{\uparrow}$ *socket exposes the kind row K, requested-supply row Q, consumption-site row S, audit-gate row G, refusal row U, transport row H, continuation row C, provenance row P, and local name row N before any supply availability is public. The exposed surface is the finite packet*

$$T = (K, Q, S, G, U, H, C, P, N),$$

not an untyped grant, host resource equality, or kind conversion outside the carrier.

Lean4 ✓ `BEDC.Derived.TypedSupplySocketUp.taste gate`

Proof. Project the carrier of Definition 1243.1. Its public coordinates are exactly $K, Q, S, G, U, H, C, P, N$, so an accepted socket has no smaller route that can expose supply availability before those rows are read.

Apply the TasteGate obligation of Definition 1243.7. The checked witness records preservation of the single carrier with these rows distinct, preventing the gate from identifying kind, request, site, refusal, transport, continuation, provenance, or local name.

Thus the only public socket read is the obligation row sequence stated above. Since supply availability has no separate coordinate, it cannot be made public without first exposing the typed socket rows. □

Theorem 1243.9 (Typed supply socket scoped rows)**.** *Every accepted* $\mathsf{TypedSupplySocket}^{\uparrow}$ *packet scopes its displayed rows*

$$K, Q, S, G, U, H, C, P, N$$

exactly over $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The public surface exports the typed socket carrier, TasteGate boundary, and obligation rows of this finite scope only; it exports no untyped supply availability row.*

Proof. Start with the NameCert obligations of Definition 1243.2. They bind carrier admission, kind exactness, requested-supply exposure, consumption-site exposure, audit-gate readback, non-internalization, transport, replay, provenance, and local naming to the displayed rows $K, Q, S, G, U, H, C, P, N$.

Apply the TasteGate obligation of Definition 1243.7. It requires the same rows to remain distinct, so the classifier row, ledger row, machine-facing consumption row, apophatic boundary row, transport, continuation, provenance, and naming surface cannot be collapsed into an untyped supply grant.

Finally use the obligation-row theorem Definition 1243.8. Since every public socket read exposes exactly $K, Q, S, G, U, H, C, P, N$ before any supply availability is public, the scoped dependency surface is precisely the listed BEDC primitives and no separate untyped availability row is exported. □

Theorem 1243.10 (Typed supply socket public consumer surface)**.** *Every public consumer of the scoped* TypedSupplySocket$^{\uparrow}$ *surface must read the rows*

$$K, Q, S, G, U, H, C, P, N$$

before any untyped supply availability row is visible. The consumer therefore receives a socket kind, requested-supply row, consumption site, audit gate, non-internalization row, transport, continuation, provenance, and local name, not an untyped grant or hidden witness.

Proof. First apply the obligation-row theorem Definition 1243.8. It exposes $K, Q, S, G, U, H, C, P, N$ before any supply availability is public, so a consumer has no earlier untyped row to inspect.

Next use the scoped-row theorem Definition 1243.9. The public surface is exactly the typed socket scope over SocketKindClassifierUp, SupplySocketLedgerUp, MachineInterfaceSocketUp, ApophaticFarEndSocketUp, BHist, hsame, Cont, Pkg, and NameCert.

Finally apply audit-gate nonescape Definition 1243.6 together with the checked TasteGate marker of Definition 1243.8. The gate-visible read preserves $K, Q, S, G, U, H, C, P, N$ as the accepted carrier rows, so public consumption cannot replace them by an untyped supply availability row. □

Falsifiable prediction: If the socket kind row is erased, Type I induction sockets and Type II witness sockets can no longer be separated by the public classifier.

Independence witness: socketkindclassifier, supplysocketledger, machineinterfacesocket, apophaticfarendsocket: SocketKindClassifierUp classifies kinds but lacks requested-supply, consumption-site, audit-gate, and non-internalization rows; SupplySocketLedgerUp accounts for requested supply but does not bind all seven kinds to a consumption site and gate; MachineInterfaceSocketUp routes formalization targets rather than the full seven-kind socket taxonomy; ApophaticFarEndSocketUp names the far-end refusal position but lacks the typed request and site rows. None is bijective to TypedSupplySocket$^{\uparrow}$ under NameCert-row preservation.

The Lean follow-up must provide explicit Nontrivial TypedSupplySocket$^{\uparrow}$ and FieldFaithful TypedSupplySocket$^{\uparrow}$ instances in `lean4/BEDC/Derived/TypedSupplySocketUp/TasteGate.lean`. The field-faithfulness witness must preserve $K, Q, S, G, U, H, C, P, N$ as distinct rows before BEDC.Derived.TypedSupplySocketUp.TypedSu can be cited by a checked paper marker.

Closure status for NameCert$_{\text{TypedSupplySocket}^{\uparrow}}$ **·**

Constructive story: The typed supply socket is generated from a $BHist$ packet that records a socket kind, requested supply, consumption site, audit gate, non-internalization row, and the socket route that consumes them together. The carrier convention reads the typed socket directly from those rows and uses $hsame$ to preserve both socket identity and kind across equivalent histories. Continuations are admitted only through Cont extensions that keep the typed row attached to the supply route. NameCert obligations, kind separation, TasteGate obligation, prediction, and independence witness certify that a consumer cannot silently retag a socket or use an untyped supply source. The socket is therefore a typed ledger route, not an ambient resource.

Theory closure: TheoryGate(NameCert$_{\text{TypedSupplySocket}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1243.9 scopes the rows $K, Q, S, G, U, H, C, P, N$ exactly over SocketKindClassifierUp, SupplySocketLedgerUp, MachineInterfaceSocketUp, ApophaticFarEndSocketUp, BHist, hsame, Cont, Pkg, and NameCert, with no untyped supply availability row exported.

Formal verification: FormalStatus(NameCert$_{\text{TypedSupplySocket}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.TypedSupplySocketUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: It does not claim untyped supply availability, host resource equality, or kind conversion outside the recorded socket carrier.

Upgrade path: Bridge closure requires a standard bridge route that exports the public typed-supply socket surface without adding an untyped supply availability row.

Chapter 1244

A Concrete Naming Certificate for $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$

The $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ packet realizes the primitive mark distinction of chapter 9462 before any Boolean, unary, observer, halting, or categorical interpretation is available. It sits beside the observer-local trace of chapter 882, the diagonal finite-trace packet of chapter 492, the mature two-endpoint interface of chapter 32, the unary-history carrier of chapter 28, and the finite kernel category surface of chapter 484. Those siblings expose later readings of distinction; this carrier binds the mark-level separation itself as finite NameCert data, before observer, halting, Boolean, unary, or categorical consumers are licensed.

Falsifiable prediction: If a PrimitiveDistinctionLedgerUp packet is accepted, no consumer can read primitive mark distinction as BoolUp data or host equality unless an explicit BoolUp NameCert bridge row is present.

Independence witness: observerdistinctiontrace, haltingdistinctiontrace, bool, nat, finitekernelcategory: none is bijective because this packet binds primitive bzero/bone mark rows, no-cross msame evidence, trace exposure, anti-Boolean refusal, transport, replay, provenance, and local naming before observer, halting, Boolean, unary, or category readings are available.

Definition 1244.1 (Primitive distinction ledger carrier)**.** *A* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Z, O, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *Z is the primitive zero-mark row, exposing the* b0 *mark as a mark-level row and not as false, zero, an object identifier, or a host datum.*
2. *O is the primitive one-mark row, exposing the* b1 *mark as a mark-level row and not as true, successor, an object identifier, or a host datum.*
3. *D is the no-cross* msame *row, recording the diagonal mark sameness judgments and the absence of generated cross-mark sameness evidence between* b0 *and* b1*.*
4. *T is the finite trace-exposure row, recording the history windows in which the two primitive mark rows may be observed without turning them into a completed data interface.*
5. *R is the anti-Boolean refusal row, recording that direct* $\mathsf{Bool}^{\uparrow}$ *readback, host equality, host disequality, truth-value import, and set-element comparison are unavailable unless a separate* $\mathsf{Bool}^{\uparrow}$ *NameCert bridge row is displayed.*

6. *H records componentwise* hsame *transport for Z, O, D, T, R and the local naming row.*
7. *C records displayed* Cont *replay from primitive mark exposure through no-cross distinction, trace exposure, refusal, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$, $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteKernelCategory}^{\uparrow}$, BHist, msame, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveDistinctionLedger}^{\uparrow}}$ *row.*

The carrier exports only these rows. It has no coordinate for direct Boolean readback, host equality, quotient collapse, observer-local ownership, halting classification, unary counting, or categorical objecthood.

Lean4 (definition) `BEDC.Derived.PrimitiveDistinctionLedgerUp.PrimitiveDistinctionLedgerUp`

Theorem 1244.2 (Primitive distinction ledger NameCert obligations)**.** *For the carrier of Definition 1244.1, the local* $\mathsf{NameCert}_{\mathsf{PrimitiveDistinctionLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $L = (Z, O, D, T, R, H, C, P, N)$, primitive* b0 *exposure through Z, primitive* b1 *exposure through O, no-cross* msame *distinction through D, finite trace exposure through T, anti-Boolean and anti-host-equality refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.PrimitiveDistinctionLedgerUp.PrimitiveDistinctionLedger namecert obligations`

Proof. Project the carrier of Definition 1244.1. The first available rows are Z and O, and they expose the two primitive marks only as mark-level rows.

Next read the distinction and trace rows. The row D records the diagonal msame evidence together with the absence of cross-mark sameness evidence, while T records the finite history windows in which these mark rows are exposed.

Finally attach the refusal and structural rows. The row R blocks direct $\mathsf{Bool}^{\uparrow}$ readback and host equality, and the rows H, C, P, N transport, replay, source, and locally name the same packet. Since the tuple has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1244.3 (Primitive distinction ledger no Boolean collapse)**.** *Every accepted* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ *consumer route that reads the two primitive mark rows must pass through the refusal row R before any* $\mathsf{Bool}^{\uparrow}$ *or host-equality interpretation is available. A direct route from Z, O, D to Boolean data, truth values, host equality, or host disequality is not an accepted* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ *route.*

Proof. Start from the carrier rows in Definition 1244.1. The primitive mark rows Z, O and the no-cross row D are separate from the refusal row R and from any later $\mathsf{Bool}^{\uparrow}$ bridge row.

Apply Definition 1244.2. The local NameCert surface requires the anti-Boolean refusal row before the packet is read as a named primitive distinction ledger.

The excluded route has no coordinate in the carrier. Transport, replay, provenance, and local naming preserve Z, O, D, T, R as the same mark-level packet, so they cannot manufacture Boolean data or host equality. A consumer that obtains such a reading has used an additional bridge outside $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.PrimitiveDistinctionLedgerUp.PrimitiveDistinctionLedger no boolean collapse`

Theorem 1244.4 (Primitive distinction ledger field coverage)**.** *Every accepted* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ *export factors through*

$$Z, O, D, T, R, H, C, P, N.$$

The consumer receives the primitive zero-mark row, primitive one-mark row, no-cross msame *row, finite trace-exposure row, anti-Boolean refusal row, componentwise transport, displayed replay, package provenance, and local NameCert naming. It receives no direct* $\mathsf{Bool}^{\uparrow}$ *endpoint, no host equality proof, no observer-local owner, no halting classifier, no unary count, and no categorical object row.*

Lean4 ✓ `BEDC.Derived.PrimitiveDistinctionLedgerUp.PrimitiveDistinctionLedger field coverage`

Proof. Use Definition 1244.2 to identify the complete local certificate surface. Every accepted consumer must expose the five content rows and four structural rows listed there.

Then apply Definition 1244.3. Any attempt to read the mark rows as Boolean data or host equality must leave the primitive distinction ledger or display a separate bridge row.

The sibling dependencies remain surrounding consumers rather than replacements. $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{FiniteKernelCategory}^{\uparrow}$ may be cited through provenance, but none supplies a hidden coordinate outside $Z, O, D, T, R, H, C, P, N$. Therefore the displayed tuple covers every accepted field of the primitive distinction ledger. □

The Lean file `lean4/BEDC/Derived/PrimitiveDistinctionLedgerUp/TasteGate.lean` provides explicit Nontrivial $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ and FieldFaithful $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ instances over the same rows. Its TasteGate records $Z, O, D, T, R, H, C, P, N$ as separate fields and keeps primitive mark exposure, no-cross msame evidence, trace exposure, and anti-Boolean refusal distinct.

Lean4 ✓ `BEDC.Derived.PrimitiveDistinctionLedgerUp.taste gate`

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveDistinctionLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$ packet is generated from primitive bzero and bone mark rows, no-cross msame evidence, finite trace exposure, anti-Boolean refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveDistinctionLedger}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1244.1, Definition 1244.2, Definition 1244.3, and Definition 1244.4 bind primitive mark distinction as a finite NameCert packet before Boolean, host-equality, observer, halting, unary, or categorical interpretation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveDistinctionLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, BoolUp endpoint readback, host equality, observer-local trace ownership, halting classification, unary counting, or categorical objecthood.

Upgrade path: Lean closure requires Nontrivial PrimitiveDistinctionLedgerUp and FieldFaithful PrimitiveDistinctionLedgerUp instances in `lean4/BEDC/Derived/PrimitiveDistinctionLedgerUp/TasteGate.lean`.

Chapter 1245

A Concrete Naming Certificate for DiagonalRegularitySeal$^{\uparrow}$

The DiagonalRegularitySeal$^{\uparrow}$ packet realizes the diagonal-regularity theorem of section 9238.1 as a finite theorem-facing seal before the terminal Real handoff. It binds the representative route of chapter 1205, the endpoint and middle estimates organized by chapter 1211, and the regularity witness of chapter 1158 into one NameCert surface. The packet is consumed by Real-completion routes such as chapter 1242, but it does not itself assert quotient stream equality, selected host limits, host real equality, countable choice, or ambient completeness.

Definition 1245.1 (Diagonal regularity-seal carrier)**.** *A* DiagonalRegularitySeal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, E_0, E_1, M, Q, W, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted* DiagonalLimitRepresentative$^{\uparrow}$ *row selecting the diagonal representative and its modulus readbacks.*
2. *E_0 and E_1 are the two endpoint-leg rows that read the regular-rational endpoints used in the triangle estimate.*
3. *M is the middle-modulus leg row exposing the regular Cauchy comparison between the selected representatives.*
4. *Q is the* DyadicToleranceTriangleLedger$^{\uparrow}$ *row that records the endpoint tolerances, middle tolerance, dyadic sum, and output tolerance bound.*
5. *W is the finite window-factorization row listing the endpoint windows, the shared intermediate window, and the requested output window.*
6. *R is the* RegularCauchyRegularityWitness$^{\uparrow}$ *row that packages the diagonal-regularity witness for downstream consumers.*
7. *Z is the optional downstream Real-seal row, used only as a terminal named handoff and never as hidden completeness evidence.*
8. *H records componentwise* hsame *transport for $D, E_0, E_1, M, Q, W, R, Z$ and the local certificate row.*
9. *C records displayed* Cont *replay from the representative row through the endpoint legs, middle modulus leg, dyadic tolerance row, window factorization, regularity witness, and terminal seal.*

10. *P is the* Pkg *provenance over* DiagonalRegularitySeal$^{\uparrow}$, DiagonalLimitRepresentative$^{\uparrow}$, DyadicToleranceTriangleLedg RegularCauchyRegularityWitness$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{DiagonalRegularitySeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or an ambient completeness theorem.

Definition 1245.2 (Diagonal regularity-seal classifier)**.** *Two accepted* DiagonalRegularitySeal$^{\uparrow}$ *carriers classify together only when their representative rows, endpoint legs, middle-modulus legs, dyadic tolerance rows, finite windows, regularity-witness rows, terminal seal rows, transport rows, continuation rows, package rows, and local naming rows are related componentwise by displayed* hsame *data and replayed along the same* Cont *route. The classifier does not identify packets by host equality of real numbers, by quotienting streams, by deleting either endpoint leg, by hiding the middle modulus leg, or by replacing the dyadic tolerance row with an ambient completeness appeal.*

Theorem 1245.3 (Diagonal regularity-seal obligations)**.** *For every accepted* DiagonalRegularitySeal$^{\uparrow}$ *carrier $S = (D, E_0, E_1, M, Q, W, R, Z, H, C, P, N)$, the local* NameCert$_{\text{DiagonalRegularitySeal}^{\uparrow}}$ *surface consists exactly of carrier exposure, classifier transport, representative exposure through D, endpoint-leg exposure through E_0 and E_1, middle-modulus exposure through M, dyadic tolerance-sum exposure through Q, finite window factorization through W, regularity-witness exposure through R, terminal Real-seal handoff through Z, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DiagonalRegularitySealUp.DiagonalRegularitySealTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 1245.1. The tuple displays the representative row, the two endpoint legs, the middle modulus leg, the dyadic tolerance row, the finite windows, the regularity witness, the terminal seal, and the structural support rows.

Classifier transport is exactly the relation of Definition 1245.2. It keeps each displayed row on the same componentwise transported route.

The data-bearing obligations are read from the cited sibling chapters. The representative row comes from chapter 1205; the dyadic tolerance surface supplies the endpoint and middle tolerance arithmetic through chapter 1211; and the regularity witness comes from chapter 1158.

Since H, C, P, N only transport, replay, source, and name those displayed rows, the listed clauses exhaust the local seal surface. □

Theorem 1245.4 (Diagonal regularity-seal nonescape)**.** *An accepted* DiagonalRegularitySeal$^{\uparrow}$ *carrier exports only the displayed rows $D, E_0, E_1, M, Q, W, R, Z, H, C, P, N$ and their classifier transports. No consumer can replace the representative row, either endpoint leg, the middle modulus leg, the dyadic tolerance row, the finite window factorization, or the regularity witness by quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.DiagonalRegularitySealUp.DiagonalRegularitySealCarrier nonescape`

Proof. Inspect the carrier coordinates of Definition 1245.1. Every exported coordinate is a displayed BEDC row or structural support for transport, continuation replay, provenance, and naming.

Definition 1245.3 accounts for the admissible carrier and classifier rows. No additional equality, selection, choice, or completeness coordinate appears in that list.

The excluded objects require coordinates absent from the carrier. Therefore neither the local NameCert row nor any displayed continuation can export them in place of the finite diagonal-regularity seal. □

Theorem 1245.5 (Diagonal regularity-seal window factorization)**.** *Every consumer of an accepted* DiagonalRegularitySeal$^{\uparrow}$ *packet factors its diagonal-regularity request through the finite route*

$$D, E_0, E_1, M, Q, W, R, Z, H, C, P, N.$$

The window row W separates the two endpoint observations, the shared middle comparison window, the dyadic tolerance output, and the terminal Real-seal handoff, so the consumer reaches the theorem seal only after reading the representative row, both endpoint legs, the middle modulus leg, and the dyadic tolerance row.

Lean4 ✓ `BEDC.Derived.DiagonalRegularitySealUp.DiagonalRegularitySealUp window factorization`

Proof. Begin with the representative route. The row D fixes the diagonal representative and the modulus-index readbacks before any endpoint or middle comparison can be consumed.

Next read the three estimate legs. The endpoint rows E_0, E_1 expose the regular-rational endpoint bounds, while M exposes the middle Cauchy-modulus comparison. The dyadic row Q records the tolerance arithmetic that combines these three legs into the output bound.

Finally factor the route through W, R, Z, H, C, P, N. The window row records the finite observations needed by the theorem, the witness row packages the regularity evidence, and the terminal Real-seal row remains downstream of the same displayed route. Hence every consumer factors through the ordered finite packet shown above. □

Lean4 ✓ `BEDC.Derived.DiagonalRegularitySealUp.DiagonalRegularitySealCarrier window factorization`

Falsifiable prediction: A diagonal regularity consumer that reaches the Real seal while omitting the representative row, either endpoint leg, the middle modulus leg, or the dyadic tolerance row refutes the packet.

Independence witness: diagonallimitrepresentative, dyadic-tolerance-triangle-ledger, regular-cauchy-regularity-witness, streamname, real: no listed sibling is bijective to DiagonalRegularitySeal$^{\uparrow}$ because this packet binds the representative, three-leg estimate, dyadic tolerance output, finite windows, and terminal Real seal as one consumer-facing seal.

Lean follow-up must add explicit Nontrivial DiagonalRegularitySeal$^{\uparrow}$ and FieldFaithful DiagonalRegularitySeal$^{\uparrow}$ instances in `lean4/BEDC/Derived/DiagonalRegularitySealUp/TasteGate.lean`.

Closure status for NameCert$_{\text{DiagonalRegularitySeal}^{\uparrow}}$**.**

Constructive story: A DiagonalRegularitySeal$^{\uparrow}$ packet is generated from a diagonal representative row, two endpoint-leg rows, a middle-modulus row, a dyadic tolerance ledger, a finite window factorization, a regularity witness, a terminal Real-seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DiagonalRegularitySeal}^{\uparrow}}$, seed)

Scope closed: Definition 1245.1, Definition 1245.2, Definition 1245.3, Definition 1245.4, and Definition 1245.5 bind the finite diagonal-regularity theorem seal surface.

Formal verification: FormalStatus(NameCert$_{\text{DiagonalRegularitySeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected host limits, host real equality, countable choice, detached completion evidence, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, nonescape, window factorization, and TasteGate nontriviality with field faithfulness for $D, E_0, E_1, M, Q, W, R, Z, H, C, P, N$.

Chapter 1246

A Concrete Naming Certificate for $\mathsf{TheoremGapRegistry}^{\uparrow}$

The $\mathsf{TheoremGapRegistry}^{\uparrow}$ packet realizes the theorem and gap registry surface of chapter 9462 as one finite NameCert carrier. It sits beside the registry layer of chapter 1220, the claim registry layer of chapter 1209, the truth-residue ledger of chapter 1239, and the gap-failure-bridge audit packet of chapter 1229. Those siblings expose package registry control, single-claim export, scientific-truth residue, and bridge-facing failure audit; this carrier joins theorem-strength rows to gap-gate rows before any consumer may read a theorem entry as stronger than its displayed dependencies or read a gap entry without its blocking gate.

Definition 1246.1 (Theorem-gap registry carrier)**.** *A* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, D, S, G, C, F, A, L, H, P, N)$$

with the following displayed rows:

1. *T is the theorem-entry row, recording name, statement, proof mode, proof site, and public registry status.*
2. *D is the dependency-strength row, recording the displayed BEDC dependencies and the weakest strength available among them.*
3. *S is the theorem-strength row, recording the strength claimed for the theorem entry and the witnessed dependency row that licenses it.*
4. *G is the gap-entry row, recording name, site, requested supply, socket type, gate, consequence if internalized, and current registry status.*
5. *C is the gap-gate coverage row, recording that every gap entry has a blocking gate and a cannot-claim or refusal route.*
6. *F is the refusal row, recording theorem-strength overclaim refusals, gate-absent gap refusals, and theorem-to-gap leakage refusals.*
7. *A is the audit-export row, recording the route through* $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{TruthResidueLedger}^{\uparrow}$, *and* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$.
8. *L is the ledger nonescape row, recording that missing theorem-strength evidence and missing gap gates cannot be supplied by prose placement, hidden registry state, or host truth.*

9. *H stores componentwise* hsame *transport for theorem entries, dependency strengths, theorem strengths, gap entries, gap gates, refusals, audit export, ledger nonescape, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{TruthResidueLedger}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoremGapRegistry}^{\uparrow}}$ *row.*

The carrier has no coordinate for a theorem strength above its displayed dependency witness, a gate-free gap entry, an erased refusal, a hidden registry table, or a ledger export outside $R = (T, D, S, G, C, F, A, L, H, P, N)$.

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistryTasteGate single carrier alignment`

Theorem 1246.2 (Theorem-gap registry strength gate)**.** *For the carrier of Definition 1246.1, an accepted theorem entry cannot be exported with strength above the weakest displayed dependency strength unless the packet also displays a witnessed dependency row in D that licenses the stronger reading through S.*

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistry strength gate`

Proof. Project the theorem-entry, dependency-strength, and theorem-strength rows T, D, S. The theorem entry is public only through the finite packet, and its stated strength is read through S rather than through prose placement or hidden theorem state.

If S reports a strength no greater than the weakest displayed dependency in D, the export is licensed by the visible dependency chain. If S reports a stronger reading, the carrier requires an additional witnessed dependency row in D naming the evidence for that stronger reading.

The structural rows H, P, N preserve, source, and name the same displayed rows. They cannot create an absent dependency witness, so a strength overclaim has no accepted export route through $\mathsf{TheoremGapRegistry}^{\uparrow}$. □

Theorem 1246.3 (Theorem-gap registry gap-gate coverage)**.** *Every accepted gap entry in a* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *packet is covered by a displayed gate in C and by a refusal or cannot-claim route in F. A gap row with no gate is not an accepted registry entry.*

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistry gap gate coverage`

Proof. Start with the gap-entry row G. It records the requested supply, socket type, consequence if internalized, and registry status, but it does not by itself explain why the supply is unavailable to the internal theory.

The coverage row C supplies that explanation by naming the blocking gate for each displayed gap entry. The refusal row F then records the public cannot-claim route or refusal route that prevents the requested supply from being read as a theorem dependency.

Finally, the audit-export and ledger rows A, L preserve the same gate and refusal data through the sibling registry surfaces. Since the carrier has no gate-free gap coordinate, a row in G without coverage in C and F is outside the accepted packet. □

Theorem 1246.4 (Theorem-gap registry ledger nonescape)**.** *Every accepted* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *export factors through*

$$T, D, S, G, C, F, A, L, H, P, N.$$

No theorem entry may escape the dependency-strength gate, and no gap entry may escape the gate-coverage row, when the packet is read through the registry layer, claim registry layer, truth-residue ledger, and gap-failure-bridge audit surface.

Proof. Use Definition 1246.2 for theorem entries. The rows T, D, S expose the claimed theorem, the dependency strengths, and the witnessed license for the reported strength.

Use Definition 1246.3 for gap entries. The rows G, C, F expose the requested supply, the blocking gate, and the refusal route.

The audit-export row A points to chapter 1220, chapter 1209, chapter 1239, and chapter 1229. Those sibling surfaces can transport and inspect the packet, but none supplies a hidden coordinate that bypasses D or C.

The ledger nonescape row L records this absence explicitly. Componentwise transport, package provenance, and local naming preserve the displayed packet, so every accepted export factors through the listed rows and no theorem or gap reading escapes its gate. □

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistry ledger nonescape`

Theorem 1246.5 (Theorem-gap registry public export surface)**.** *Every public theorem/gap registry export from an accepted* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *packet is routed through*

$$T, D, S, G, C, F, A, L, H, P, N.$$

The public surface exposes theorem entries, dependency strengths, theorem strengths, gap entries, gap-gate coverage, refusals, audit export, ledger nonescape, transport, provenance, and local naming as separate rows; it exports no theorem entry above its displayed dependency license and no gap entry without its displayed gate.

Proof. Start from the carrier definition. The packet lists the rows $T, D, S, G, C, F, A, L, H, P, N$ and has no hidden registry table or outside ledger coordinate.

Use the strength gate and gap-gate coverage theorems. Theorem 1246.2 prevents a theorem entry from exceeding its displayed dependency license, while Theorem 1246.3 prevents a gap entry from being accepted without a gate and refusal route.

Finally apply ledger nonescape. Theorem 1246.4 forces every accepted export through the same displayed rows and through the sibling registry, claim-registry, truth-residue, and gap-failure audit surfaces. Hence the listed rows form the public export surface. □

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistry public export surface`

Falsifiable prediction: An accepted $\mathsf{TheoremGapRegistry}^{\uparrow}$ packet cannot raise a theorem entry above the weakest displayed dependency strength unless a new witnessed dependency row is added to the packet.

Independence witness: registrylayer, claimregistrylayer, truth-residue-ledger, gap-failure-bridge-audit: $\mathsf{TheoremGapRegistry}^{\uparrow}$ joins theorem-strength rows to gap-gate rows; the sibling packets lack that combined carrier and are not bijective to it.

The future Lean `TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{TheoremGapRegistry}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{TheoremGapRegistry}^{\uparrow}$ instances. Those instances must preserve $T, D, S, G, C, F, A, L, H, P, N$ as separate displayed fields before any checked theorem may cite the $\mathsf{TheoremGapRegistry}^{\uparrow}$ packet as a field-faithful registry surface.

Theorem 1246.6 (Theorem-gap registry bridge surface)**.** *Every theorem/gap export from an accepted* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *packet bridges to downstream registry consumers only through*

$$T, D, S, G, C, F, A, L, H, P, N.$$

The rows expose theorem entries, dependency strengths, theorem strengths, gap entries, gap gates, refusals, audit export, ledger nonescape, transport, provenance, and local naming together with the sibling registry surfaces $\mathsf{RegistryLayer}^{\uparrow}$*,* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*,* $\mathsf{TruthResidueLedger}^{\uparrow}$*, and* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$*.*

Proof. Start with the public export surface of Definition 1246.5. It already identifies the accepted export rows as $T, D, S, G, C, F, A, L, H, P, N$ and rules out hidden theorem tables, hidden gap gates, and outside ledger coordinates.

Next use ledger nonescape from Definition 1246.4. The export row A passes through chapter 1220, chapter 1209, chapter 1239, and chapter 1229, while L records that neither theorem strength nor gap coverage may be supplied outside the displayed packet.

Finally, componentwise transport H, provenance P, and local naming N preserve the same rows for downstream consumers. Hence the only bridge surface available to theorem/gap exports is the displayed row list together with the sibling registry surfaces named above. □

Lean4 ✓ `BEDC.Derived.TheoremGapRegistryUp.TheoremGapRegistry bridge surface`

Closure status for $\mathsf{NameCert}_{\mathsf{TheoremGapRegistry}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TheoremGapRegistry}^{\uparrow}$ packet is generated from theorem-entry, dependency-strength, theorem-strength, gap-entry, gap-gate coverage, refusal, audit-export, ledger nonescape, hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoremGapRegistry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1246.1, Definition 1246.2, Definition 1246.3, Definition 1246.4, Definition 1246.5, and the displayed TasteGate marker bind theorem-strength, gap-gate scope, and public registry export to visible BHist, hsame, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoremGapRegistry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, complete registry governance, bridge checking, or correctness of every downstream theorem and gap entry.

Upgrade path: Bridge closure requires a standard bridge and Lean verification remains independent.

Chapter 1247

A Concrete Naming Certificate for $\mathsf{FocusedLedger}^{\uparrow}$

The $\mathsf{FocusedLedger}^{\uparrow}$ packet realizes the finite focused multi-Hist ledger of subsection 9462.27.1 and Definition 9462.846 as a concrete NameCert carrier. It sits beside $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ in chapter 1206, $\mathsf{ObserverHistIdentity}^{\uparrow}$ in chapter 479, $\mathsf{InterHistLocalityLedger}^{\uparrow}$ in chapter 759, $\mathsf{InscriptionEvent}^{\uparrow}$ in chapter 519, and $\mathsf{OnticResidueLedger}^{\uparrow}$ in chapter 1152. Those siblings consume histories, identity rows, locality rows, inscription events, or residue rows; this packet binds the whole focus-selector-digest-gap-relation-invariant-audit tuple before those readings are separated.

Definition 1247.1 (Focused ledger carrier)**.** *A* $\mathsf{FocusedLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N)$$

with the following displayed rows:

1. *I is the finite index ledger naming the available history slots;*
2. *H assigns a finite* Hist *row to each slot of I;*
3. *T records local continuation chains for the displayed histories, without adding a global scalar clock;*
4. *F is the focus index selecting the visible history slot;*
5. *S is the selector ledger that records the branch choices and focus reads used by F;*
6. *D is the digest ledger recording finite summaries whose provenance remains displayed;*
7. *G is the gap and provenance row recording boundary pressure, missing observations, and source data for each digest;*
8. *R is the inter-history relation row;*
9. *J is the inter-history invariant row;*
10. *Q is the dependence row, including optional maximum-rate readings when such rows are present;*
11. *ρ is the focused self-proxy row for the selected history;*
12. *A is the audit row refusing subject parameters, global clocks, hidden choosers, unledgered compression, object-level T, and hidden dependence on meta-closure;*

13. *C records displayed* Cont *replay across* $I, H, T, F, S, D, G, R, J, Q, \rho, A$*;*
14. *P records* Pkg *provenance over* $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{ConsciousUniverseSpace}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{OnticResidueLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg*, and* NameCert *rows;*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a subject substance, ambient universe object, hidden scheduler, unrecorded quotient of histories, object-level time object, or meta-closure oracle.

Lean4 (definition) `BEDC.Derived.FocusedLedgerUp.FocusedLedgerUp`

Theorem 1247.2 (Focused ledger audit closure)**.** *For the carrier of Definition 1247.1, the audit row A closes exactly the refusals of no subject, no global clock, no hidden chooser, no unledgered compression, no object-level T, and no hidden meta-closure dependence. Every accepted* $\mathsf{FocusedLedger}^{\uparrow}$ *consumer must read focus, selection, digest, gap, relation, invariant, dependence, self-proxy, replay, provenance, and local naming through*

$$I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N.$$

Proof. Project the carrier. The rows $I, H, T, F, S, D, G, R, J, Q, \rho$ expose the finite histories, focus, selector, digest, gap, relation, invariant, dependence, and self-proxy data that the packet is allowed to use.

Read the audit row next. The no-subject refusal prevents a hidden owner coordinate; the no-global-clock refusal keeps T local; the no-hidden-chooser refusal forces all branch choice through S; the no-unledgered-compression refusal keeps summaries tied to D and G; the no object-level T refusal prevents a time object from replacing the local continuation rows; and the no hidden meta-closure refusal prevents the packet from citing a closure source not displayed in A.

Finally use the structural rows. The replay row C, provenance row P, and local name N preserve and name only the displayed packet, so no consumer can recover an omitted subject, clock, chooser, compression, time object, or meta-closure dependency from the certificate surface. □

Theorem 1247.3 (Focused ledger NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$ *surface for Definition 1247.1 consists of exactly five obligation rows:*

1. *carrier admission for the finite packet $L = (I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N)$;*
2. *classifier stability for the selected history, selector ledger, digest ledger, relation row, invariant row, dependence row, and self-proxy row under componentwise* hsame *transport;*
3. *continuation exactness for the displayed* Cont *routes in C and no replay route outside the finite index ledger I;*
4. *audit-row exactness for the six refusals recorded in A;*
5. *provenance and local naming through P and N.*

Proof. The carrier definition displays all content rows before transport, replay, provenance, and naming. The first obligation admits that finite packet as the only carrier surface.

The second and third obligations are forced by the BEDC structural rows: hsame transport preserves the selected carrier fields componentwise, and Cont replay is read only through C.

The fourth obligation is Definition 1247.2. The fifth obligation is the terminal NameCert packaging: P records the source rows and N names the accepted packet. No sixth obligation is available, because every accepted consumer already factors through the displayed carrier. □

Lean4 ✓ `BEDC.Derived.FocusedLedgerUp.FocusedLedgerNameCert obligation surface`

Definition 1247.4 (Focused ledger kernel scope)**.** *The kernel scope of a* $\mathsf{FocusedLedger}^{\uparrow}$ *consumer is the displayed* BHist *route*

$$I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N$$

from Definition 1247.1. A consumer is kernel-scoped when carrier admission, selector, digest, gap, relation, invariant, dependence, self-proxy, audit refusals, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming are all read through this route before any focused-ledger conclusion is exported.*

Theorem 1247.5 (Focused ledger kernel-scope route)**.** *Every* $\mathsf{FocusedLedger}^{\uparrow}$ *consumer factors carrier admission, selector rows, digest rows, gap rows, relation rows, invariant rows, dependence rows, self-proxy rows, audit refusals,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *naming through the kernel scope of Definition 1247.4. In particular, it cannot route the focused ledger through a subject parameter, global clock, hidden chooser, object-level time coordinate, unledgered compression step, or meta-closure oracle.*

Proof. Start from the NameCert obligation surface, Definition 1247.3. Its carrier-admission and classifier-stability rows already name the selected history, selector ledger, digest ledger, relation row, invariant row, dependence row, and self-proxy row as displayed coordinates.

Next apply audit closure, Definition 1247.2. The six refusals rule out the omitted subject, clock, chooser, compression, object-level time, and meta-closure coordinates before the consumer reaches replay or naming.

Finally read the kernel scope definition. The route ends with componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row, so any exported focused-ledger conclusion remains inside the displayed BHist kernel rows. □

Theorem 1247.6 (Focused ledger sibling independence)**.** $\mathsf{FocusedLedger}^{\uparrow}$ *is not bijective with* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*,* $\mathsf{InterHistLocalityLedger}^{\uparrow}$*,* $\mathsf{InscriptionEvent}^{\uparrow}$*, or* $\mathsf{OnticResidueLedger}^{\uparrow}$*. The focused ledger carries the whole focus-selector-digest-gap-relation-invariant-audit tuple, while each sibling omits at least one of the required focused-ledger rows or one audit refusal.*

Proof. Compare the carrier inventories. $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ reads universe-slice, locality-cell, local-time, and trace-identity structure after a focused ledger row is available, but it does not carry the full selector, digest, gap, relation, invariant, dependence, and audit tuple of Definition 1247.1.

$\mathsf{ObserverHistIdentity}^{\uparrow}$ supplies observer-history identity rows, and $\mathsf{InterHistLocalityLedger}^{\uparrow}$ supplies inter-history locality rows. Neither packet contains the focused selector ledger, digest provenance, self-proxy, and six-refusal audit row together.

$\mathsf{InscriptionEvent}^{\uparrow}$ records inscription-event admission, while $\mathsf{OnticResidueLedger}^{\uparrow}$ records residue-bearing observation structure. Those packets can consume focused-ledger information, but neither contains the whole finite packet $I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N$. Therefore no sibling is bijective with $\mathsf{FocusedLedger}^{\uparrow}$ as a NameCert carrier. □

Falsifiable prediction: A $\mathsf{FocusedLedger}^{\uparrow}$ packet cannot be reduced to an $\mathsf{InscriptionEvent}^{\uparrow}$ packet without losing at least the selector, digest, relation, invariant, or audit row.

Independence witness: consciousuniversespace, observerhistidentity, interhistlocalityledger, inscriptionevent, onticresidueledger: none is bijective with $\mathsf{FocusedLedger}^{\uparrow}$ because each omits at least one required focused-ledger row or audit refusal.

The Lean TasteGate file `lean4/BEDC/Derived/FocusedLedgerUp/TasteGate.lean` provides explicit $\mathsf{Nontrivial}\ \mathsf{FocusedLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{FocusedLedger}^{\uparrow}$ instances over the same finite carrier.

Theorem 1247.7 (Focused ledger carrier totality)**.** *For every accepted* $\mathsf{FocusedLedger}^{\uparrow}$ *carrier* $L = (I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N)$*, the rows* $I, H, T, F, S, D, G, R, J, Q, \rho, A$ *are finite* BHist *data before the structural rows are read. The rows* C, P, N *then provide displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$ *naming for exactly that finite carrier.*

Proof. Project the carrier definition, Definition 1247.1. It lists the index ledger, history rows, continuation chains, focus index, selector ledger, digest ledger, gap row, relation row, invariant row, dependence row, self-proxy row, and audit row before any replay, provenance, or naming row appears.

Apply the audit closure theorem, Definition 1247.2. The audit row refuses hidden subject, clock, chooser, compression, object-level time, and meta-closure coordinates, so the displayed carrier rows are the whole object surface.

Finally use the NameCert obligation surface, Definition 1247.3. It packages the same finite rows through C, P, N, and the sibling independence theorem prevents replacement by a neighboring packet that omits part of the focused-ledger carrier. □

Theorem 1247.8 (Focused ledger classifier stability)**.** *The classifier of an accepted* $\mathsf{FocusedLedger}^{\uparrow}$ *packet is stable only through componentwise* hsame *transport over* $I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N$*. In particular, the selected history, selector ledger, digest ledger, relation row, invariant row, dependence row, self-proxy row, audit row, replay row, provenance row, and local name remain attached to the same finite* BHist *carrier.*

Proof. Use the classifier-stability clause in Definition 1247.3. The clause names the selected history, selector ledger, digest ledger, relation row, invariant row, dependence row, and self-proxy row as transported fields rather than as host-level predicates.

Read the carrier totality theorem, Definition 1247.7. Since every classified field is part of the finite BHist packet, componentwise hsame transport cannot substitute an unledgered object for one of the displayed rows.

The TasteGate line supplies the same nontrivial and field-faithful carrier surface. Together with audit closure and sibling independence, it keeps the classifier on the focused-ledger rows rather than on a sibling packet or hidden meta-coordinate. □

Theorem 1247.9 (Focused ledger ledger exactness)**.** *The ledger exported by* $\mathsf{FocusedLedger}^{\uparrow}$ *is exact for the route*

$$I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N.$$

Every consumer must read finite BHist *rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming through that route before using the focus, digest, relation, invariant, dependence, self-proxy, or audit data.*

Proof. Start from carrier totality, Definition 1247.7. It fixes the displayed ledger rows and prevents any omitted subject, clock, chooser, compression, object-level time, or meta-closure coordinate from becoming part of the carrier.

Apply classifier stability, Definition 1247.8. The same rows remain attached under componentwise hsame transport, so a consumer cannot alter the selected history or ledger fields while preserving the local classifier.

Finally apply audit closure, the NameCert obligation surface, sibling independence, and the TasteGate line. These force replay through C, provenance through P, and naming through N, while excluding sibling replacement and hidden dependencies. Hence the exported ledger is exact for the displayed BEDC kernel route. □

Theorem 1247.10 (Focused ledger public export surface)**.** *The public* $\mathsf{FocusedLedger}^{\uparrow}$ *export is exactly the finite route*

$$I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N.$$

Every public consumer of the packet receives the displayed index ledger, history rows, local continuation chains, focus index, selector ledger, digest ledger, gap and provenance row, inter-history relation row, invariant row, dependence row, self-proxy row, audit row, replay row, provenance row, and local naming row. No public export adds a subject coordinate, global clock, hidden chooser, unledgered compression step, object-level time object, meta-closure oracle, or sibling replacement.

Proof. Project the carrier of Definition 1247.1. The listed rows are precisely the finite BHist packet admitted by the definition, and the carrier has no additional public coordinate.

Use the kernel-scope route of Definition 1247.5 and ledger exactness of Definition 1247.9. They force every consumer through the same focused selector, digest, relation, invariant, dependence, self-proxy, audit, replay, provenance, and naming rows before any focused-ledger conclusion is exported.

Finally apply sibling independence, Definition 1247.6. The surrounding packets may consume or support the focused ledger, but none of them carries the whole route $I, H, T, F, S, D, G, R, J, Q, \rho, A, C, P, N$. Therefore the public surface is exactly that displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FocusedLedger}^{\uparrow}$ packet is generated from finite history slots, local continuation rows, a focus index, selector rows, digest rows, gap and provenance rows, relation rows, invariant rows, dependence rows, a self-proxy row, audit refusals, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1247.1, Definition 1247.4, Definition 1247.2, Definition 1247.3, Definition 1247.5, Definition 1247.6, Definition 1247.7, Definition 1247.8, Definition 1247.9, and Definition 1247.10 bind the finite focused multi-Hist ledger, its kernel-scoped obligations, and its public export route as a concrete NameCert packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FocusedLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public-object export, global spacetime, subject substance, or hidden meta-closure.

Upgrade path: Formal closure requires checked carrier totality, classifier stability, ledger exactness, audit closure, sibling independence, provenance, and local naming for BEDC.Derived.FocusedLedgerUp.

Chapter 1248

A Concrete Naming Certificate for $\mathsf{RefusalRegistry}^{\uparrow}$

The $\mathsf{RefusalRegistry}^{\uparrow}$ packet realizes the refusal rows named by section 9462.34 as a finite NameCert carrier for blocked package export. It sits beside the registry layer of chapter 1220, the claim registry layer of chapter 1209, the theorem-gap registry of chapter 1246, and the truth-residue ledger of chapter 1239. Those siblings admit package rows, control single-claim export, bind theorem and gap entries, and preserve scientific-truth residue; this carrier stores the refusal packet that blocks an export route before a consumer may read silence, placement, or package provenance as permission.

Definition 1248.1 (Refusal registry carrier)**.** *A* $\mathsf{RefusalRegistry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, E, G, C, S, V, T, P, N)$$

with the following displayed rows:

1. *B is the blocked-claim row, recording the package claim, theorem entry, gap entry, or ordinary-language entry point whose export is blocked.*
2. *E is the refusal-entry row, naming the refusal reason, refusal site, registry owner, and exact package surface to which the refusal applies.*
3. *G is the gate row, recording the finite gate that blocks export through* $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, *or* $\mathsf{TheoremGapRegistry}^{\uparrow}$.
4. *C is the cannot-claim route, recording the public route by which the blocked row is read as unavailable rather than as a theorem, accepted claim, or scientific-truth residue.*
5. *S is the status row, recording whether the refusal is active, discharged by a displayed registry entry, or outside the present package surface.*
6. *V is the provenance row, tying the refusal to the finite source file, registry row, theorem, definition, closure-status block, or truth-residue row from which it is exported.*
7. *T stores componentwise* hsame *transport for blocked claim, refusal entry, gate, cannot-claim route, status, provenance, package provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{RefusalRegistry}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{TruthResidueLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RefusalRegistry}^{\uparrow}}$ *row.*
The carrier has no coordinate for an accepted export with an absent refusal entry, an absent gate, an erased cannot-claim route, a hidden acceptance table, or a package claim outside $R = (B, E, G, C, S, V, T, P, N)$.

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistryTasteGate single carrier alignment`

Theorem 1248.2 (Refusal registry export control)**.** *Every blocked package claim governed by* $\mathsf{RefusalRegistry}^{\uparrow}$ *must display the blocked row B, refusal entry E, gate row G, cannot-claim route C, status row S, provenance row V, componentwise transport T, package provenance P, and local naming row N in the same finite carrier. A consumer may cite the blocked claim only through that refusal packet.*

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistryCarrier export control`

Proof. Project the carrier of Definition 1248.1. The blocked claim is not merely a missing registry entry; it is the finite packet $R = (B, E, G, C, S, V, T, P, N)$.

Rows B, E, G, C give the export-control surface. The blocked row names the target, the refusal entry names the reason and site, the gate records the finite block, and the cannot-claim route gives the public reading of the block.

Rows S, V, T, P, N make the refusal auditable. They preserve the status, source, transport, package provenance, and local name of the same packet, so the consumer cannot cite the blocked claim by dropping the refusal entry or moving through a sibling registry without the gate. □

Theorem 1248.3 (Refusal registry nonescape)**.** *A package claim with no refusal entry, no gate row, and no accepted registry entry has no export route through* $\mathsf{RefusalRegistry}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistry nonescape`

Proof. Use Definition 1248.2. Any blocked export governed by $\mathsf{RefusalRegistry}^{\uparrow}$ is read through $R = (B, E, G, C, S, V, T, P, N)$, and an accepted export must instead be supplied by a sibling registry entry in chapter 1220, chapter 1209, or chapter 1246.

If no refusal entry E and no gate row G are present, the refusal packet is absent. If no accepted registry entry is present, the sibling acceptance route is absent as well.

The structural rows S, V, T, P, N cannot manufacture either missing route; they only preserve and name a displayed packet. Hence the package claim has no export route through $\mathsf{RefusalRegistry}^{\uparrow}$. □

Theorem 1248.4 (Refusal registry claim-layer handoff)**.** *Every* $\mathsf{RefusalRegistry}^{\uparrow}$ *packet handed to* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *exposes the same blocked claim, refusal entry, gate, cannot-claim route, status, provenance, transport, package, and local NameCert rows. The claim layer may read the packet as a refusal boundary, but it may not convert the blocked row into an accepted claim unless a separate accepted claim-entry row is displayed.*

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistry claim layer handoff`

Proof. Start with the rows of Definition 1248.1. Componentwise transport T sends the displayed refusal rows to the sibling claim layer without merging them with an accepted claim-entry row.

The claim registry layer of chapter 1209 already distinguishes an entry row from a refusal-boundary row. The handoff therefore reads $B, E, G, C, S, V, T, P, N$ as the refusal boundary data for the claim surface.

If a separate accepted claim-entry row is absent, the claim layer has only the refusal boundary supplied by $\mathsf{RefusalRegistry}^{\uparrow}$. Package provenance and local naming preserve that boundary; they do not create acceptance. Thus the handoff blocks conversion from refusal to claim export without a displayed accepted entry. □

Theorem 1248.5 (Refusal registry obligation pack)**.** *For an accepted* $\mathsf{RefusalRegistry}^{\uparrow}$ *packet* $R = (B, E, G, C, S, V, T, P, N)$*, the complete local obligation pack consists exactly of the carrier row* B*, refusal-entry classifier row* E*, gate row* G*, cannot-claim row* C*, status row* S*, provenance row* V*, componentwise transport row* T*, package row* P*, and local* $\mathsf{NameCert}_{\mathsf{RefusalRegistry}^{\uparrow}}$ *row* N*. Any package consumer that drops one of these rows leaves the refusal packet and receives neither an accepted claim nor a valid refusal export.*

Proof. Project the carrier of Definition 1248.1. Its displayed coordinates are precisely $B, E, G, C, S, V, T, P, N$, so the carrier and classifier surface is finite and has no hidden acceptance row.

Apply Definition 1248.2 and Definition 1248.3. The first theorem forces blocked exports to use the displayed refusal packet, while the second blocks any route that has neither the refusal entry and gate nor a separate accepted registry entry.

Finally apply Definition 1248.4. The handoff preserves the same nine rows when the claim layer reads the packet, and it cannot convert the blocked row into an accepted claim without an accepted claim-entry row. Thus the listed rows are exactly the local obligation pack. □

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistryObligationPack`

Theorem 1248.6 (Refusal registry public export surface)**.** *Every downstream package or registry consumer may cite* $\mathsf{RefusalRegistry}^{\uparrow}$ *only through the displayed refusal packet*

$$R = (B, E, G, C, S, V, T, P, N).$$

The public citation surface consists of the blocked-claim row B*, refusal-entry row* E*, gate row* G*, cannot-claim route* C*, status row* S*, provenance row* V*, componentwise transport row* T*, package row* P*, and local naming row* N*; it contains no silent acceptance coordinate and no route that lets a package or registry consumer read a blocked claim outside those rows.*

Proof. Project the carrier of Definition 1248.1. Its visible coordinates are exactly $B, E, G, C, S, V, T, P, N$, so any public read of the registry packet starts from the displayed refusal rows.

Use Definition 1248.2 and Definition 1248.5. The export-control theorem confines blocked package claims to the same packet, and the obligation pack shows that dropping one row leaves both accepted-claim export and valid refusal export unavailable.

The claim-layer handoff of Definition 1248.4 preserves this surface for registry consumers. Therefore downstream packages and registries may cite only the displayed refusal packet, with no hidden acceptance route. □

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistry public export surface`

Theorem 1248.7 (Refusal registry bridge-schema export)**.** *The public export surface of Definition 1248.6 and the claim-layer handoff of Definition 1248.4 jointly supply the bridge-schema boundary for every registry consumer of* $\mathsf{RefusalRegistry}^{\uparrow}$*. A consumer that crosses the boundary must preserve the blocked-claim row* B*, refusal-entry row* E*, gate row* G*, cannot-claim route* C*, status row* S*, provenance row* V*, componentwise transport row* T*, package row* P*, and local naming row* N *as the same displayed refusal packet.*

Lean4 ✓ `BEDC.Derived.RefusalRegistryUp.RefusalRegistry bridge schema export`

Proof. The public export theorem identifies the only rows through which downstream package and registry consumers may cite $\mathsf{RefusalRegistry}^{\uparrow}$. These rows are already the displayed packet $R = (B, E, G, C, S, V, T, P, N)$, with no silent acceptance coordinate.

The claim-layer handoff theorem sends exactly the same rows to $\mathsf{ClaimRegistryLayer}^{\uparrow}$ and keeps accepted claim-entry data separate. Hence the public surface and the handoff agree on the boundary that a bridge-schema consumer may inspect.

Because the boundary contains only the displayed refusal rows, any bridge-schema export preserves the packet as a refusal surface rather than converting it into acceptance, erasure, or an ambient registry predicate. □

Falsifiable prediction: A package claim with no refusal entry, gate row, or accepted registry entry has no export route through $\mathsf{RefusalRegistry}^{\uparrow}$.

Independence witness: registrylayer, claimregistrylayer, theoremgapregistry, truthresidueledger: none is bijective to $\mathsf{RefusalRegistry}^{\uparrow}$ because this carrier stores blocked claim, refusal entry, gate, cannot-claim route, status, provenance, transport, package, and local naming rows as a dedicated refusal packet.

The Lean follow-up file `lean4/BEDC/Derived/RefusalRegistryUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ RefusalRegistry}^{\uparrow}$ and $\mathsf{FieldFaithful\ RefusalRegistry}^{\uparrow}$ instances. Those instances must preserve $B, E, G, C, S, V, T, P, N$ as separate displayed fields before any checked theorem may cite the $\mathsf{RefusalRegistry}^{\uparrow}$ packet as a field-faithful refusal surface.

Closure status for $\mathsf{NameCert}_{\mathsf{RefusalRegistry}^{\uparrow}}$.

Constructive story: A $\mathsf{RefusalRegistry}^{\uparrow}$ packet is generated from blocked-claim, refusal-entry, gate, cannot-claim route, status, provenance, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RefusalRegistry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1248.1, Definition 1248.2, Definition 1248.3, Definition 1248.4, Definition 1248.6, Definition 1248.7, the TasteGate marker, the falsifiable prediction, and the independence witness bind the refusal rows against $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{TruthResidueLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RefusalRegistry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, complete package governance, or correctness of every downstream refusal row.

Upgrade path: No further paper-side theory grade is claimed before Lean-side bridge checking for package and registry consumers of the displayed refusal packet.

Chapter 1249

A Concrete Naming Certificate for $\mathsf{ActiveReadingGate}^{\uparrow}$

The $\mathsf{ActiveReadingGate}^{\uparrow}$ packet realizes the single-active-reading governance surface of chapter 9462 as a finite NameCert carrier. It reads the current-reading discipline through the machine-interface ledger and sits beside the registry export surface of chapter 1216, the registry layer of chapter 1220, the claim registry layer of chapter 1209, the audit transcript envelope of chapter 1234, and the writing-item audit packet of chapter 1235. Those siblings expose registry export, registry admission, claim admission, transcript replay, and writing-item legality; this carrier binds the unique active reading before any downstream consumer may export a prose, registry, or machine-facing route.

Definition 1249.1 (Active reading gate carrier)**.** *An* $\mathsf{ActiveReadingGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, A, R, B, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the target row, naming the theorem, definition, registry entry, writing item, machine-facing target, or package surface whose reading is being exported.*
2. *A is the active-reading row, recording the single current reading allowed for T.*
3. *R is the retired-reading row, recording displaced readings as non-exportable audit data rather than as active alternatives.*
4. *B is the blocking or refusal row, recording the public refusal that blocks a second active reading for the same target.*
5. *E is the export row, recording the only route through which the active reading may leave the gate.*
6. *H records componentwise* hsame *transport for target, active reading, retired reading, blocking row, export row, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from the target row through the active-reading and blocking rows to the export row.*
8. *P is the* Pkg *provenance over* $\mathsf{ActiveReadingGate}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{AuditTranscriptEnvelope}^{\uparrow}$, $\mathsf{WritingItemAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ActiveReadingGate}^{\uparrow}}$ *row.*
The carrier has no coordinate for a parallel active reading, hidden target alias, export that bypasses the blocking row, audit transcript without replay, writing item without the current reading, or local name outside the finite packet $G = (T, A, R, B, E, H, C, P, N)$.

Lean4 ✓ `BEDC.Derived.ActiveReadingGateUp.ActiveReadingGateTasteGate single carrier alignment`

Theorem 1249.2 (Active reading gate NameCert obligations)**.** *For the carrier of Definition 1249.1, the local* $\mathsf{NameCert}_{\mathsf{ActiveReadingGate}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (T, A, R, B, E, H, C, P, N)$; target admission through T; active-reading uniqueness through A and B; retired-reading exposure through R; registry and blocking coverage through* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, *and* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*; export non-escape through E; and transport, replay, provenance, and naming stability through H, C, P, N.*

Proof. Project the carrier of Definition 1249.1. The rows $T, A, R, B, E, H, C, P, N$ are the only coordinates available to the local certificate.

The target and reading rows supply the governance surface. Row T fixes the object being read, row A supplies the unique current reading, row R records displaced readings only as audit data, and row B blocks any attempt to export a second current reading.

The registry siblings provide the public routes. $\mathsf{RegistryExportSurface}^{\uparrow}$ controls export, $\mathsf{RegistryLayer}^{\uparrow}$ keeps package rows finite, and $\mathsf{ClaimRegistryLayer}^{\uparrow}$ prevents a claim consumer from treating the retired row as a second accepted claim.

Rows E, H, C, P, N make that boundary stable. They expose the single export route, preserve the displayed rows componentwise, replay the route through the blocking surface, record package provenance, and name the packet. No remaining coordinate can create a parallel active reading or bypass the refusal row. □

Theorem 1249.3 (Active reading gate single export route)**.** *Every registry, claim, transcript, or writing-item consumer of an accepted* $\mathsf{ActiveReadingGate}^{\uparrow}$ *packet must read the target row T through the active-reading row A and the blocking row B before the export row E. Such a consumer cannot export a parallel active reading for the same target.*

Proof. Start with an accepted packet

$$G = (T, A, R, B, E, H, C, P, N).$$

By Definition 1249.2, the export row E is available only as part of the displayed carrier containing T, A, R, B, H, C, P, N.

A registry or claim consumer therefore reaches E through the unique active row A and the refusal row B. The retired row R remains visible, but it is read as non-exportable audit data rather than as a second claim route.

Transcript and writing-item consumers receive the same route through the provenance row P, which names chapter 1234 and chapter 1235. Since H only transports displayed fields and C only replays the displayed route, neither structural row can turn R into a second active row.

Thus every accepted export factors through T, A, B, E, with R blocked from active export. A packet exporting both A and R as active readings for the same target leaves $\mathsf{ActiveReadingGate}^{\uparrow}$ and is refused by B. □

Falsifiable prediction: If both A and R are exported as active readings for the same target, the packet is refuted by the blocking row.

Independence witness: registryexportsurface, registrylayer, claimregistrylayer, audittranscriptenvelope, writingitemaudit: none is bijective to ActiveReadingGateUp. The registry export surface controls public export but does not store the active and retired readings together; the registry layer lacks the single-reading gate; the claim registry layer lacks retired-reading audit data; the audit transcript envelope lacks registry blocking as the active-reading gate; and the writing-item audit packet lacks the target-specific export route T, A, B, E as one finite carrier.

The Lean follow-up must provide explicit Nontrivial $\mathsf{ActiveReadingGate}^{\uparrow}$ and FieldFaithful $\mathsf{ActiveReadingGate}^{\uparrow}$ instances in `lean4/BEDC/Derived/ActiveReadingGateUp/TasteGate.lean`. Those instances must preserve $T, A, R, B, E, H, C, P, N$ as separate displayed fields before BEDC.Derived.ActiveReadingGateUp.taste_gate can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{ActiveReadingGate}^{\uparrow}}$.

Constructive story: An $\mathsf{ActiveReadingGate}^{\uparrow}$ packet is generated from target, active-reading, retired-reading, blocking, export, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ActiveReadingGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1249.1, Definition 1249.2, and Definition 1249.3 bind single-active-reading export to a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ActiveReadingGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, complete governance of every prose replacement, correctness of every registry consumer, or archival completeness for retired readings.

Upgrade path: Lean closure requires Nontrivial ActiveReadingGateUp and FieldFaithful ActiveReadingGateUp instances in `lean4/BEDC/Derived/ActiveReadingGateUp/TasteGate.lean`.

Chapter 1250

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$

The $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet names the finite dyadic mesh used by Bishop-style constructive compactness arguments for located closed bounded intervals. It consumes the endpoint and cell route of chapter 394, the dyadic refinement surface of chapter 306, the rational cell carrier of chapter 354, the finite Real seal of chapter 38, and the downstream compact-uniform-continuity consumer of chapter 388. The construction is an explicit finite net: the interval is read through located endpoints, rational cells, dyadic cells, unary mesh scale, displayed centers, finite coverage rows, StreamName and RegSeqRat windows, RealUp seal rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row. It does not use open-cover compactness, choice of subcovers, quotient streams, host real equality, or an ambient completeness primitive.

Definition 1250.1 (Closed bounded interval net carrier)**.** *A* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, M, Q, D, Z, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located closed bounded interval row supplied by* $\mathsf{LocatedInterval}^{\uparrow}$*, including lower and upper located endpoint windows;*
2. *M is the unary mesh scale naming the dyadic precision used for the net;*
3. *Q is the finite* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger attached to L at scale M;*
4. *D is the finite* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger for the rational cells of Q;*
5. *Z is the* ProbeBundle *center-row ledger, with one displayed dyadic or rational center for each accepted cell;*
6. *F is the finite coverage ledger saying that every displayed containment read of L enters one of the cells in Q and its attached dyadic refinement in D;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the endpoint and center names at the mesh scale;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same endpoint and center windows;*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, M, Q, D, Z, F, S, R are displayed;*

10. *H records componentwise* hsame *transport for endpoints, cells, dyadic refinements, centers, coverage rows, windows, readbacks, and the Real seal;*
11. *C records displayed* Cont *routes from the located interval row through mesh selection, finite cell coverage, StreamName windows, RegSeqRat readback, and RealUp sealing;*
12. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}^{\uparrow}}$ *row.*

The carrier exports no open-cover object, selected finite subcover, quotient stream, host real equality, hidden center chooser, or ambient compactness principle.

Theorem 1250.2 (Closed bounded interval net NameCert obligations)**.** *For the carrier of Definition 1250.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (L, M, Q, D, Z, F, S, R, E, H, C, P, N)$, located endpoint admission through L, unary mesh admission through M, rational cell admission through Q, dyadic refinement admission through D, ProbeBundle center-row admission through Z, finite coverage admission through F, StreamName and RegSeqRat readback through S and R, RealUp sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetTasteGate single carrier alignment`

Proof. Project the carrier. Its first rows display a located interval, a unary dyadic mesh scale, rational cells, dyadic refinements, center rows, and finite coverage rows before any Real seal is exposed.

Read the analytic rows next. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks are attached to the same endpoint and center rows, and the $\mathsf{Real}^{\uparrow}$ seal is downstream of those finite windows rather than a completed host interval.

The structural rows close the certificate surface. The transport row H, replay row C, package row P, and local name N preserve and name only the displayed mesh packet, so there is no further NameCert obligation outside the listed rows. □

Theorem 1250.3 (Closed bounded interval net dyadic coverage)**.** *Every containment read of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet factors through the finite route*

$$L, M, Q, D, Z, F, H, C, P, N.$$

The read first consumes the located endpoint row L, then the mesh scale M, then a displayed rational cell in Q, a displayed dyadic refinement in D, the corresponding center row in Z, and the coverage row F. It receives no open cover, subcover choice, hidden center selector, or host interval order outside that route.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetCoverageRoute semantic name certificate`

Proof. Begin with the located interval row. By the carrier definition, every containment read starts from L and is admitted only at the unary mesh scale M.

The coverage row F names a finite cell route. It sends the read into a rational cell from Q, the attached dyadic refinement from D, and the matching ProbeBundle center row from Z.

Finally the rows H, C, P, N transport, replay, source, and name this finite route. Since the carrier has no coordinate for an open cover, a chosen subcover, a hidden selector, or a host interval order, the dyadic coverage read cannot escape the displayed net. □

Theorem 1250.4 (Closed bounded interval net Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet reads the interval through the finite route*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The Real seal row E is reached only after the located endpoint row, mesh row, rational and dyadic cell rows, ProbeBundle center rows, finite coverage ledger, StreamName windows, and RegSeqRat readbacks have been displayed. Consequently the downstream compact-uniform-continuity route receives an explicit finite dyadic net and receives no open-cover compactness principle, quotient stream equality, selected host real, or ambient Real completeness row.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetRealSeal nonescape`

Proof. Use Definition 1250.2. The accepted packet exposes E only after L, M, Q, D, Z, F, S, R have been read as one finite mesh surface.

Use Definition 1250.3. It identifies the containment side of the packet with the displayed rational and dyadic cells and their ProbeBundle centers, so a consumer cannot replace the mesh by an open-cover object or a hidden center choice.

The $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumer reads only the resulting Real-facing packet. The structural rows H, C, P, N preserve the same finite route through transport, replay, provenance, and local naming, and no carrier coordinate supplies quotient streams, host real equality, or ambient completeness data. □

Definition 1250.5 (Closed bounded interval net scope ledger)**.** *The scoped* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *ledger is the displayed row family*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N$$

of Definition 1250.1, read with the following scope assignments: L is the $\mathsf{LocatedInterval}^{\uparrow}$ *source row, M is the unary dyadic mesh scale, Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger, D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger, Z is the displayed center ledger, F is finite coverage, S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, E is the* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise transport, C is displayed replay, P is provenance, and N is the local NameCert row. The scope ledger is visible exactly to* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumers and does not add an open-cover compactness row, selected subcover, quotient stream, host real equality, or ambient Real completeness principle.*

Lean4 (definition) `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetScopeLedger`

Theorem 1250.6 (Closed bounded interval net RegSeqRat window exactness)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet, the finite rows*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N$$

make the $\mathsf{RegSeqRat}^{\uparrow}$ *readback exact for the displayed interval net: the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row L, mesh row M,* $\mathsf{RationalInterval}^{\uparrow}$ *cells Q,* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D, center row Z, and coverage row F determine the* $\mathsf{StreamName}^{\uparrow}$ *windows S that the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R may read before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. The transport, replay, provenance, and local naming rows H, C, P, N preserve that route for* $\mathsf{Real}^{\uparrow}$ *consumers and for the downstream* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *handoff.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetRegSeqRatWindowExactness`

Proof. Use Definition 1250.5. It assigns each of the rows $L, M, Q, D, Z, F, S, R, E, H, C, P, N$ to a concrete BEDC scope surface and places S and R after the interval, mesh, rational-cell, dyadic-refinement, center, and coverage rows.

The readback exactness is therefore row-order exactness. A $\mathsf{RegSeqRat}^{\uparrow}$ request cannot inspect a window until the $\mathsf{StreamName}^{\uparrow}$ row S has been selected from the displayed mesh data L, M, Q, D, Z, F.

Finally the $\mathsf{Real}^{\uparrow}$ seal E and the structural rows H, C, P, N are downstream of that readback. They preserve the same route for $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumers and introduce no additional analytic coordinate. □

Theorem 1250.7 (Closed bounded interval net compact-uniform consumer handoff)**.** *Every* CompactUniformConti *consumer of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet must read the interval through the scoped row family*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The consumer receives a $\mathsf{LocatedInterval}^{\uparrow}$ *source row, a mesh row,* $\mathsf{RationalInterval}^{\uparrow}$ *cells,* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements, displayed centers, finite coverage,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. It receives no open-cover compactness principle, selected subcover, quotient stream, host real equality, hidden center chooser, or ambient Real completeness row.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetCompactUniformConsumerHandoff`

Proof. Apply Definition 1250.4. It already confines every $\mathsf{Real}^{\uparrow}$ or $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumer to the finite route $L, M, Q, D, Z, F, S, R, E, H, C, P, N$.

Apply Definition 1250.6. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks used by the consumer are determined by the displayed located interval, mesh, rational cells, dyadic refinements, centers, and coverage rows.

The handoff is therefore the same scoped ledger, not a stronger compactness object. Since the carrier contains no open-cover compactness row, selected subcover, quotient stream, host real equality, hidden center chooser, or ambient completeness coordinate, the compact-uniform consumer receives none of them. □

Theorem 1250.8 (Closed bounded interval net public finite-net export)**.** *An accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet exports to compact-uniform consumers exactly the located interval endpoints, dyadic interval cells, rational interval ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}^{\uparrow}}$ *row carried by*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The public export contains no open-cover compactness principle, selected subcover, trichotomy row, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetPublicFiniteNetExport`

Proof. Start from the carrier of Definition 1250.1. Its located endpoint row, mesh row, rational interval ledger, dyadic interval cells, center rows, finite coverage ledger, windows, readbacks, Real seal, transport, replay, provenance, and local naming are the only coordinates available for export.

Use Definition 1250.3 and Definition 1250.4. They identify the interval-facing and Real-facing reads with the displayed finite dyadic mesh route before any compact-uniform consumer can inspect the packet.

Use Definition 1250.6 and Definition 1250.7. The window and readback rows are exact for the same finite ledger, and the compact-uniform handoff preserves precisely that scoped row family.

The exclusions are then carrier exclusions. Since the packet has no coordinate for open-cover compactness, selected subcovers, trichotomy, quotient real equality, or ambient Real completeness, no public export row can supply those objects. □

Definition 1250.9 (Closed bounded interval net root-unblock ledger)**.** *The root-unblock ledger for* ClosedBoundedIntervalNet$^{\uparrow}$ *is the same displayed row family*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N$$

as in Definition 1250.1, read with M as the unary mesh counter and with Q, D, Z, F as the rational cell, dyadic refinement, center, and finite coverage rows at that counter. The rows S, R, E record the StreamName window, RegSeqRat exact readback, and RealUp seal reached from that mesh, while H, C, P, N preserve transport, replay, provenance, and local naming. The ledger contains no root oracle, open-cover compactness row, selected subcover, quotient stream, hidden center chooser, or ambient completeness primitive.

Theorem 1250.10 (Closed bounded interval net unary mesh monotonicity)**.** *Let $K = (L, M, Q, D, Z, F, S, R, E, H$ be an accepted* ClosedBoundedIntervalNet$^{\uparrow}$ *packet, and let M' be a unary mesh scale obtained by extending M by displayed successor rows. The refined packet keeps the source row L and replaces Q, D, Z, F, S, R, E by rows at scale M' so that every rational cell in Q is subdivided by rational cells in the refined ledger, every dyadic cell in D is subdivided by dyadic cells in the refined ledger, and every coverage route displayed by F factors through the refined* ProbeBundle *center rows. Transport by H, replay by C, provenance by P, and local naming by N preserve the same coverage route.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetUnaryMeshMonotonicity`

Proof. Start from the dyadic coverage theorem Definition 1250.3. It places every containment read on the finite route $L, M, Q, D, Z, F, H, C, P, N$, so the mesh coordinate is the only row being extended.

The successor extension of M refines the rational cell row Q and the dyadic refinement row D by finite subdivision. The center row Z is then read from the refined cells, and the coverage row F factors the old containment read through the new center ledger rather than through a different compactness object.

Finally apply the scope ledger of Definition 1250.5. The rows S, R, E, H, C, P, N are downstream of the displayed mesh rows, so they replay the refined route and do not introduce an open-cover, quotient, hidden-choice, or ambient-completeness coordinate. □

Theorem 1250.11 (Closed bounded interval net root-unblock package)**.** *For every accepted* ClosedBoundedIntervalNet$^{\uparrow}$ *packet, the ledger of Definition 1250.9 assembles a finite root-unblock package: repeated unary refinement of M yields rational and dyadic cell refinements through Q and D, preserves the displayed* ProbeBundle *coverage route through Z and F, reads StreamName and RegSeqRat windows exactly through S and R, and hands the resulting RealUp-facing interval packet to every* CompactUniformContinuity$^{\uparrow}$ *consumer through E, H, C, P, N. The package does not claim open-cover compactness, a numerical root selector, quotient streams, host real equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetRootUnblockPackage`

Proof. Use Definition 1250.10 for the induction over the unary mesh row. Each successor step refines Q and D and keeps the coverage route inside Z and F.

Use Definition 1250.6 after the refined mesh has been displayed. The StreamName windows and RegSeqRat readbacks are determined by the located interval, mesh, rational cells, dyadic cells, centers, and coverage rows.

The compact-uniform handoff is then exactly Definition 1250.7. It exports the finite route $L, M, Q, D, Z, F, S, R, E, H, C, P, N$ to the consumer and exports no stronger compactness or selection principle. □

Theorem 1250.12 (Closed bounded interval net finite mesh coverage)**.** *For every accepted* ClosedBoundedIntervalN *packet $K = (L, M, Q, D, Z, F, S, R, E, H, C, P, N)$, the displayed rows Q, D, Z, F give finite mesh coverage of the located interval row L: each containment read admitted by L at mesh scale M is routed through a rational cell in Q, the attached dyadic cell in D, the corresponding center row in Z, and the finite coverage row F. The coverage route exports no open-cover compactness object, selected subcover, hidden center chooser, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetFiniteMeshCoverage`

Proof. Project the carrier of Definition 1250.1. The rows Q, D, Z, F are displayed as the rational cell ledger, dyadic refinement ledger, center ledger, and finite coverage ledger attached to the same located interval L and mesh scale M.

Apply the dyadic coverage theorem Definition 1250.3. It already confines every containment read to the route $L, M, Q, D, Z, F, H, C, P, N$, so the mesh coverage visible to a downstream consumer is the finite cell route through Q, D, Z, F.

The structural rows H, C, P, N preserve that same route and add no compactness coordinate. Thus the theorem strengthens the root-unblock surface by making the finite mesh coverage row explicit, without claiming open-cover compactness, selected subcovers, hidden centers, or ambient completeness. □

Theorem 1250.13 (Closed bounded interval net located endpoint stability)**.** *For every accepted* ClosedBoundedIntervalNet$^{\uparrow}$ *packet, the located endpoint data in L is stable under the transport route L, H, C: replacing endpoint windows along the displayed* hsame *row H and replaying the same* Cont *row C preserves the rational-cell, dyadic-cell, center, coverage, StreamName, RegSeqRat, and Real-seal route used by the packet. The stability claim is only about the displayed located endpoint transport; it does not identify host real endpoints or add quotient equality.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetLocatedEndpointStability`

Proof. Use the scope ledger Definition 1250.5. It places the located endpoint row L before the mesh, cell, window, readback, and Real-seal rows, and assigns H and C as the componentwise transport and replay rows for that same family.

Apply the RegSeqRat exactness theorem Definition 1250.6. The windows S and readbacks R are determined by the displayed located interval, mesh, rational cells, dyadic cells, centers, and coverage rows, so endpoint transport through H cannot change the route inspected by the readback.

Finally use the root-unblock package Definition 1250.11. Its handoff reaches compact-uniform consumers only through E, H, C, P, N, so the transported endpoint data remains the displayed located route and never becomes host real equality or quotient endpoint equality. □

Theorem 1250.14 (Closed bounded interval net compact-uniform route)**.** *Every compact-uniform consumer of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet consumes the full finite route*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The route displays located endpoints, a mesh scale, rational cells, dyadic refinements, centers, finite coverage, StreamName windows, RegSeqRat exact readback, a RealUp seal, transport, replay, provenance, and local naming before the compact-uniform handoff. It does not supply open-cover compactness, selected limits, quotient streams, host real equality, hidden center choice, or RealUp completeness.

Proof. Apply Definition 1250.12. It identifies the finite coverage rows Q, D, Z, F that every containment read must use after reading L and M.

Apply Definition 1250.6. The StreamName and RegSeqRat rows S, R are exact for the displayed endpoint and center windows, and the Real seal E is reached only after those finite rows are visible.

Finally apply Definition 1250.7 and Definition 1250.11. Together they route every compact-uniform consumer through $L, M, Q, D, Z, F, S, R, E, H, C, P, N$ and export no stronger compactness, selection, quotient, host-real, hidden-center, or completeness principle. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetCompactUniformRoute`

Theorem 1250.15 (Closed bounded interval net public envelope)**.** *Every public read of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet factors through the displayed compact-uniform continuity row together with the interval-net route*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

Thus the read consumes the dyadic and rational interval rows, $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}^{\uparrow}}$ *row before it reaches any compact-uniform consumer. It exports no open-cover compactness principle, selected subcover, selected limit, quotient stream, host real equality, hidden center chooser, or ambient Real completeness row.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetPublicEnvelope`

Proof. Project the carrier of Definition 1250.1. Its public rows are the located interval source L, mesh scale M, rational cells Q, dyadic refinements D, center row Z, coverage ledger F, StreamName windows S, RegSeqRat readback R, Real seal E, and the structural rows H, C, P, N.

Apply Definition 1250.7. The compact-uniform consumer receives exactly the scoped row family already displayed by the interval-net packet, so its public read cannot skip the dyadic, rational, regular-readback, stream-window, Real-seal, transport, replay, provenance, or local naming rows.

Finally use Definition 1250.11. The package exports a finite mesh handoff rather than an open-cover or selected-limit object, and the carrier has no coordinate for the excluded compactness and host-real principles. □

Theorem 1250.16 (Closed bounded interval net mesh refinement fusion)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet* $K = (L, M, Q, D, Z, F, S, R, E, H, C, P, N)$*, extending the unary mesh row* M *by displayed successor steps fuses the rational-cell ledger* Q*, the dyadic-refinement ledger* D*, the center rows* Z*, and the finite coverage ledger* F *into one refined interval-net route. The fused route preserves the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows, and it does not import open-cover compactness, selected subcovers, hidden center choice, quotient streams, or ambient completeness.*

Proof. Start from Definition 1250.10. Each displayed successor of M refines the rational cells in Q and the dyadic cells in D while keeping the source interval row L fixed.

Next use Definition 1250.11. It identifies the refined coverage route through Z and F and then carries the same mesh data through S, R, E, H, C, P, N.

The exclusion is a carrier exclusion. By Definition 1250.9 and Definition 1250.15, the packet has no coordinate for open-cover compactness, selected subcovers, hidden centers, quotient streams, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetMeshRefinementFusion`

Theorem 1250.17 (Closed bounded interval net cell-center readback)**.** *Every displayed center read of an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet factors through the ordered rows* Q, D, Z, F, S, R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is visible. Thus a center is read as a rational cell in* Q*, a dyadic refinement in* D*, a* $\mathsf{ProbeBundle}$ *center row in* Z*, a finite coverage row in* F*, a* $\mathsf{StreamName}^{\uparrow}$ *window in* S*, and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* R*, not as a hidden host chooser or quotient real representative.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetCellCenterReadback`

Proof. Project the carrier and scope ledger, Definition 1250.1 and Definition 1250.5. They place Q, D, Z, F before the window and readback rows and before the terminal Real-facing row.

Apply Definition 1250.6. The readback row R may inspect only the windows S determined by the displayed located interval, rational cells, dyadic cells, centers, and coverage rows.

Finally use Definition 1250.4. Since E is downstream of Q, D, Z, F, S, R, the center read cannot be replaced by host center choice, quotient real equality, or ambient Real completeness. □

Theorem 1250.18 (Closed bounded interval net dense mesh readback)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet* $K = (L, M, Q, D, Z, F, S, R, E, H, C, P, N)$ *and every displayed successor refinement of the mesh row* M*, every* $\mathsf{Real}^{\uparrow}$*-facing dense mesh read factors through*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The refined rows Q, D, Z, F *give the rational and dyadic cells, displayed centers, and finite coverage at the successor mesh, while* S, R, E *give the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal. The read receives no open-cover compactness principle, selected subcover, quotient real equality, hidden center choice, or ambient Real completeness row.*

Proof. Start from finite mesh coverage. The row F covers each displayed containment read of the located interval row L by cells from Q and dyadic refinements from D at the mesh scale M.

Use unary mesh monotonicity and Definition 1250.16. A displayed successor refinement of M refines the same finite cell route rather than selecting a host subcover, and the refined Q, D, Z, F rows remain part of the accepted interval-net packet.

Next apply Definition 1250.17. The center and dense-mesh reads pass through Q, D, Z, F, S, R before the Real-facing row is visible, so the readback is a finite window and regular-rational route.

Finally use Definition 1250.4. The $\mathsf{Real}^{\uparrow}$ seal E and the structural rows H, C, P, N preserve that refined finite mesh route, and no coordinate supplies open-cover compactness, selected subcovers, quotient real equality, hidden center choice, or ambient Real completeness. □

Theorem 1250.19 (Closed bounded interval net compact-uniform modulus input)**.** *Every* CompactUniformContinu *consumer receives an accepted* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet as finite modulus input through the complete row family*

$$L, M, Q, D, Z, F, S, R, E, H, C, P, N.$$

The consumer reads the located interval, mesh scale, rational cells, dyadic refinements, centers, finite coverage, stream windows, regular-rational readbacks, Real seal, transport, replay, provenance, and local naming before any uniform-continuity conclusion can be replayed.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetUniformModulusInput`

Proof. Use Definition 1250.7. It already states that compact-uniform consumers receive the scoped row family $L, M, Q, D, Z, F, S, R, E, H, C, P, N$ and no stronger compactness object.

Apply Definition 1250.14. The route makes Q, D, Z, F the finite coverage input and S, R, E the readback and Real-facing boundary used by the consumer.

Finally combine Definition 1250.16 and Definition 1250.17. Mesh refinement and center readback stay inside the same finite packet, so the compact-uniform input is a displayed interval net rather than an open-cover compactness principle, selected subcover, hidden center chooser, quotient stream, host real equality, or ambient completeness row. □

Falsifiable prediction: Every compact-uniform-continuity read of a ClosedBoundedIntervalNetUp packet factors through the displayed finite dyadic mesh, not through an open-cover compactness row.

Independence witness: locatedinterval, dyadicinterval, rationalinterval, real, compactuniformcontinuity: ClosedBoundedIntervalNetUp is not bijective to any listed sibling because it jointly carries located endpoints, a unary mesh scale, rational and dyadic cell ledgers, ProbeBundle centers, finite coverage rows, StreamName and RegSeqRat readback, a RealUp seal, and the compact-uniform-continuity handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}\uparrow}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet is generated from located interval endpoints, a unary dyadic mesh scale, rational cells, dyadic refinements, ProbeBundle centers, finite coverage rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1250.1, Definition 1250.2, Definition 1250.3, Definition 1250.4, Definition 1250.5, Definition 1250.6, Definition 1250.7, and Definition 1250.8 bind the scoped finite dyadic mesh route for located closed bounded RealUp intervals and compact-uniform consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalNet}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ClosedBoundedIntervalNetUp.ClosedBoundedIntervalNetCoverageRoute\_semantic\_name\_certi`

Bridge status: paperBridge

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient streams, host real equality, selected host limits, or ambient Real completeness.

Upgrade path: No further theory-axis upgrade remains; bridge checking is the remaining formal-axis route beyond the current paper bridge.

Chapter 1251

A Concrete Naming Certificate for $\mathsf{CoreRegistryIndex}^{\uparrow}$

The $\mathsf{CoreRegistryIndex}^{\uparrow}$ packet realizes the finite registry tables of section 9462.34 as one NameCert carrier for package-level indexing. It sits beside the export surface of chapter 1216, the cannot-claim registry of chapter 554, the verifier boundary of chapter 1188, the focused ledger of chapter 1109, and the GroundCompiler recognition packet of chapter 821. Those siblings supply export, refusal, machine-verifier, focused-history, and recognition rows; this carrier binds the index itself before a consumer may treat a registry table as an accepted route.

Definition 1251.1 (Core registry index carrier)**.** *A* $\mathsf{CoreRegistryIndex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, T, G, D, S, F, U, O, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the concept-registry row, naming BEDC concepts and their package surfaces;*
2. *T is the theorem-registry row, naming theorem entries, proof sites, and checked or unclosed status;*
3. *G is the gap-registry row, naming open gaps, refusal sites, and boundary items;*
4. *D is the tradition-registry row, recording inherited mathematical or philosophical entry surfaces;*
5. *S is the scientific-claim row, recording empirical or model-facing claims with their audit boundaries;*
6. *F is the refusal row, recording blocked export, cannot-claim, unavailable-evidence, or policy refusal entries;*
7. *U is the route and upgrade row, recording how a registry entry may move to a stronger closure surface;*
8. *O is the ordinary-language row, recording entry points by which a reader reaches a finite registry entry;*
9. *A is the audit row, recording export checks, consistency gates, and machine-readable review data;*
10. *H is the componentwise* hsame *transport row for all displayed registry rows;*

11. *K is the displayed* Cont *continuation row through the concept, theorem, gap, tradition, scientific-claim, refusal, route, ordinary-language, and audit entries;*
12. *P is the* Pkg *provenance over* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{LargeModelOutputVerifier}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{GroundCompilerRecognition}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CoreRegistryIndex}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unlicensed export, hidden total index, theorem-content synthesis, erased refusal, audit-free route, or ordinary-language entry point outside the finite packet $R = (C, T, G, D, S, F, U, O, A, H, K, P, N)$.

Lean4 ✓ `BEDC.Derived.CoreRegistryIndexUp.CoreRegistryIndexCarrier row surface semantic certificate`

Theorem 1251.2 (Core registry index NameCert obligations)**.** *For the carrier of Definition 1251.1, the local* $\mathsf{NameCert}_{\mathsf{CoreRegistryIndex}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (C, T, G, D, S, F, U, O, A, H,$ registry exposure for concept, theorem, gap, tradition, scientific-claim, refusal, route, and ordinary-language rows; audit exposure through A; componentwise transport through H; displayed continuation through K; package provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.CoreRegistryIndexUp.CoreRegistryIndexTasteGate single carrier alignment`

Proof. Project the carrier definition. The finite packet displays every registry table named by the core-registry surface together with audit, transport, continuation, package, and local naming rows.

The sibling chapters provide only bounded surfaces. The export surface controls public registry export, the cannot-claim packet supplies refusal discipline, the large-model verifier controls checked machine output, the focused multi-Hist ledger supplies finite history focus, and the GroundCompiler recognition packet supplies recognized machine-facing rows.

Rows A, H, K, P, N bind those registry rows as one NameCert surface. Since no row names a hidden total index or theorem-content generator, the obligations are exactly the displayed finite registry packet and its audit, transport, continuation, provenance, and naming data. □

Theorem 1251.3 (Core registry index nonescape)**.** *Every public consumer of an accepted* $\mathsf{CoreRegistryIndex}^{\uparrow}$ *packet must read the concept, theorem, gap, tradition, scientific-claim, refusal, route, ordinary-language, audit, transport, continuation, provenance, and local NameCert rows together. Removing the refusal row F or audit row A exits the accepted registry index and cannot be repaired by transport, continuation, package provenance, or local naming alone.*

Lean4 ✓ `BEDC.Derived.CoreRegistryIndexUp.CoreRegistryIndex refusal audit nonescape certificate`

Proof. Apply the NameCert obligations of Definition 1251.2. They require the registry rows C, T, G, D, S, F, U, O and the audit row A before the packet may be exported as $\mathsf{CoreRegistryIndex}^{\uparrow}$.

The export surface and cannot-claim sibling chapters keep exported rows and blocked rows distinct. The verifier, focused-ledger, and recognition siblings keep machine-output, history-focus, and recognition data bounded by their displayed rows.

Thus a consumer that removes F loses the refusal boundary, and a consumer that removes A loses the export audit. The support rows H, K, P, N preserve an admitted packet; they do not synthesize the missing refusal or audit surface. □

Falsifiable prediction: Removing the refusal row or audit row permits an unlicensed registry export.

Independence witness: registryexportsurface, axiszeckendorfcannotclaim, largemodeloutputverifier, focused-multi-hist-ledger, groundcompilerrecognition: none is bijective to $\mathsf{CoreRegistryIndex}^{\uparrow}$. The export surface lacks the concept, theorem, gap, tradition, scientific-claim, ordinary-language, and upgrade index together; the cannot-claim packet lacks accepted registry export rows; the large-model verifier lacks package registry rows; the focused multi-Hist ledger lacks refusal and ordinary-language index rows; the GroundCompiler recognition packet lacks theorem, gap, refusal, audit, and route rows as one finite registry carrier.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{CoreRegistryIndex}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CoreRegistryIndex}^{\uparrow}$ instances in `lean4/BEDC/Derived/CoreRegistryIndexUp/TasteGate.lean`. Those instances must preserve $C, T, G, D, S, F, U, O, A, H, K, P, N$ as separate displayed fields before any checked theorem may cite the $\mathsf{CoreRegistryIndex}^{\uparrow}$ packet as a field-faithful registry index.

Theorem 1251.4 (Core registry index export-surface bridge)**.** *A public read of* $\mathsf{CoreRegistryIndex}^{\uparrow}$ *bridges to* $\mathsf{RegistryExportSurface}^{\uparrow}$ *only when the refusal row F and audit row A remain displayed with the concept, theorem, gap, tradition, scientific-claim, route, ordinary-language, transport, continuation, provenance, and local naming rows. Removing either F or A leaves the bridge route and cannot be repaired by export-surface recognition alone.*

Proof. Apply the carrier and nonescape rows of Definition 1251.1 and Definition 1251.3. They force every public registry-index consumer to retain the refusal row F and audit row A together with the displayed registry rows.

The registry export surface can expose an accepted public row only after those rows remain visible. Since the export surface does not synthesize missing refusal or audit evidence, a read without F or A is outside the $\mathsf{CoreRegistryIndex}^{\uparrow}$ bridge route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CoreRegistryIndex}^{\uparrow}}$.

Constructive story: A $\mathsf{CoreRegistryIndex}^{\uparrow}$ packet is generated from concept, theorem, gap, tradition, scientific-claim, refusal, route, ordinary-language, audit, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoreRegistryIndex}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1251.1, Definition 1251.2, Definition 1251.3, Definition 1251.4, the visible sibling routes through RegistryExportSurfaceUp, AxisZeckendorfCannotClaimUp, LargeModelOutputVerifierUp, FocusedMultiHistLedgerUp, and GroundCompilerRecognitionUp, the falsifiable prediction row, and the independence-witness row bind the public exported registry-index surface for the displayed finite packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoreRegistryIndex}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.CoreRegistryIndexUp.CoreRegistryIndex\_refusal\_audit\_nonescape\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not close Lean bridge checking, registry completeness, hidden total indexing, or correctness of every downstream registry entry.

Upgrade path: No further paper theory-axis strengthening is recorded for this registry-index packet; bridge checking remains a separate formal-axis target.

Chapter 1252

A Concrete Naming Certificate for $\mathsf{ConceptRegistrySurface}^{\uparrow}$

The $\mathsf{ConceptRegistrySurface}^{\uparrow}$ packet realizes the registry family of Definition 9462.2206 and the concept-entry discipline of Definition 9462.2209 as a finite NameCert carrier. It sits beside the export-control packet of chapter 1216, the failed-gate witness of chapter 901, and the auditable-generativity package of chapter 9462. Those siblings expose registry export, diagnostic failure, or package-level vision rows; this packet binds concept entry, theorem entry, gap entry, tradition entry, science entry, cannot-claim row, upgrade route, transport, provenance, and local naming before a philosophical concept can leave the registry family as BEDC prose.

Definition 1252.1 (Concept registry surface carrier)**.** *A* $\mathsf{ConceptRegistrySurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (C, T, G, R, S, F, U, H, P, N)$$

with the following displayed rows:

1. *C is the concept-entry row, recording the concept name, BEDC site, carrier, classifier, ledger, positive reading, forbidden readings, gap status, closure status, and formal status.*
2. *T is the theorem-entry row, recording the theorem or theorem-facing obligation to which the concept may point.*
3. *G is the gap-entry row, recording the visible socket, missing proof, boundary pressure, or formalization target carried by the concept.*
4. *R is the tradition-entry row, recording comparison with inherited philosophical vocabulary without turning that comparison into package membership.*
5. *S is the science-entry row, recording empirical or records-side claims whose export remains tied to displayed observations and prediction sockets.*
6. *F is the cannot-claim row, recording the forbidden readings that block export as subject substance, global world object, hidden theorem, primitive law, or unlicensed term.*
7. *U is the upgrade-route row, recording the route by which a concept may move to a stronger registry status after a displayed carrier, theorem, gap, or bridge row is present.*
8. *H stores componentwise* hsame *transport for concept, theorem, gap, tradition, science, cannot-claim, upgrade, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* ConceptRegistrySurface$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, AuditGateFailureWitness$^{\uparrow}$, AxisZeckendorfCannotClaim$^{\uparrow}$, *philosophy-package registry rows,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$ *row.*

The carrier has no coordinate for a concept with no displayed BEDC site, no forbidden-reading row, no registry route, or no local naming row.

Lean4 (definition) `BEDC.Derived.ConceptRegistrySurfaceUp.ConceptRegistrySurfaceCarrier`

Theorem 1252.2 (Concept registry surface export exactness)**.** *For the carrier of Definition 1252.1, the local* NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for $Q = (C, T, G, R, S, F, U, H, P, N)$;*
2. *concept-entry exactness through C, requiring the displayed BEDC site, carrier, classifier, ledger, positive reading, forbidden readings, gap status, closure status, and formal status to remain in one row;*
3. *registry-family coverage through T, G, R, S, F, U, requiring theorem, gap, tradition, science, cannot-claim, and upgrade readings to be displayed before export;*
4. *componentwise transport and displayed route replay through H and the registry routes named in P;*
5. *package provenance and local naming through P and N.*

Proof. Project the finite packet of Definition 1252.1. The concept row C contains the exact entry fields required by the concept registry, while T, G, R, S, F, U place the same concept inside the theorem, gap, tradition, science, cannot-claim, and upgrade routes of the registry family.

The export-control surface of chapter 1216 supplies the public reading discipline: a concept leaves the philosophy package only through displayed registration, status, blocking, audit, and non-smuggling rows. Here those rows are specialized to the concept-entry exactness carried by C and to the registry-family rows T, G, R, S, F, U.

Finally H, P, N transport, package, and name the same packet. Since Q contains no hidden registry state or prose-only export coordinate, the five rows listed in the statement exhaust the local NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.ConceptRegistrySurfaceUp.ConceptRegistrySurface export exactness certificate`

Theorem 1252.3 (Concept registry surface forbidden-reading refusal)**.** *Every accepted* ConceptRegistrySurface$^{\uparrow}$ *packet refuses export of a philosophical concept that lacks one of the displayed BEDC site in C, forbidden-reading row F, registry-family route T, G, R, S, U, transport row H, provenance row P, or local naming row N. A consumer cannot replace those rows by prose synthesis, inherited tradition, scientific vocabulary, host semantics, or an audit-gate failure read as theorem content.*

Lean4 ✓ `BEDC.Derived.ConceptRegistrySurfaceUp.ConceptRegistrySurface forbidden reading refusal certificate`

Proof. Use Definition 1252.2. It binds every exportable concept to the displayed concept entry and to the theorem, gap, tradition, science, cannot-claim, and upgrade rows of the same finite packet.

If the BEDC site in C is absent, the concept remains external vocabulary rather than an accepted registry entry. If the forbidden-reading row F is absent, the packet no longer records which readings are blocked, so it cannot be an accepted concept-registry export.

The sibling failure surface of chapter 901 records failed gate items as diagnostic refusal data. It does not convert a missing concept row or missing forbidden-reading row into theorem content. Package provenance and local naming preserve the displayed refusal route; they do not add a hidden export path. □

Theorem 1252.4 (Concept registry surface bridge schema)**.** *For an accepted* ConceptRegistrySurface$^{\uparrow}$ *packet* $Q = (C, T, G, R, S, F, U, H, P, N)$*, the export exactness row of Definition 1252.2 and the forbidden-reading refusal row of Definition 1252.3 form the paper bridge schema from the local registry surface to the abstract registry interface. The schema exports only the displayed concept-entry, theorem-entry, gap-entry, tradition-entry, science-entry, cannot-claim, upgrade, transport, provenance, and local naming rows, and it does not assert Lean bridge checking.*

Lean4 ✓ `BEDC.Derived.ConceptRegistrySurfaceUp.ConceptRegistrySurface bridge schema certificate`

Proof. Read the export exactness theorem first. It identifies the registry-facing public surface with the displayed rows $C, T, G, R, S, F, U, H, P, N$ and makes those rows the only admissible source of concept export.

Read the forbidden-reading refusal theorem next. It blocks prose synthesis, inherited tradition, scientific vocabulary, host semantics, and diagnostic failure rows from replacing any displayed registry coordinate.

Together these two rows give the bridge schema: every abstract registry-interface read must enter through the local carrier rows and must respect the refusal boundary. Since the schema is stated entirely on the paper side, it records no Lean bridge certificate. □

Falsifiable prediction: An exported philosophical concept without a displayed BEDC site, forbidden-reading row, and registry route is outside ConceptRegistrySurfaceUp.

Independence witness: registryexportsurface, auditgatefailurewitness, cannotclaimregistry, conceptregistry, philosophypackage: none is bijective because ConceptRegistrySurfaceUp carries the concept-entry exactness and forbidden-reading route simultaneously while each sibling lacks at least one of those rows.

The Lean follow-up for ConceptRegistrySurface$^{\uparrow}$ must provide Nontrivial ConceptRegistrySurface$^{\uparrow}$ and FieldFaithful ConceptRegistrySurface$^{\uparrow}$ instances in `lean4/BEDC/Derived/ConceptRegistrySurfaceUp/TasteG` The instances must preserve the concept, theorem, gap, tradition, science, cannot-claim, upgrade, transport, provenance, and local NameCert rows of $Q = (C, T, G, R, S, F, U, H, P, N)$ before any later checked theorem cites the carrier as a TasteGate surface.

Closure status for NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$.

Constructive story: A ConceptRegistrySurface$^{\uparrow}$ packet is generated from concept, theorem, gap, tradition, science, cannot-claim, upgrade, componentwise hsame transport, package provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1252.2, Definition 1252.3, Definition 1252.4, the falsifiable prediction, the independence witness, and the existing ConceptRegistrySurfaceTasteGate_single_carrier_alignment, Nontrivial, and FieldFaithful Lean support in `lean4/BEDC/Derived/ConceptRegistrySurfaceUp/TasteGate.lean` bind the paper bridge schema, examples, and completeness coverage for the concept registry surface.

Formal verification: FormalStatus(NameCert$_{\text{ConceptRegistrySurface}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, complete philosophy-package governance, or correctness of every downstream concept registry entry.

Upgrade path: The remaining work is Lean verification and bridge checking on the formal axis, not further paper-axis closure.

Chapter 1253

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$

The $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ packet records the Bishop–Bridges and Heine–Cantor interval route inside the BEDC finite-ledger substrate. It specializes the compact-uniform-continuity vision of chapter 9237 to a closed bounded real interval: the endpoint and membership rows come from $\mathsf{Real}^{\uparrow}$ and $\mathsf{Interval}^{\uparrow}$, compactness is read through the finite dyadic-cover surface of chapter 129, pointwise continuity is read through chapter 130, the Lebesgue-radius ledger is supplied by chapter 452, and the output is the $\mathsf{UniformModulus}^{\uparrow}$ consumer row.

Definition 1253.1 (Closed bounded interval uniform-modulus carrier)**.** *A* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, R, C, D, L, M, U, H, Q, P, N)$$

with the following visible rows:

1. *A and B are* $\mathsf{Real}^{\uparrow}$ *endpoint rows with the displayed interval orientation $A \leq B$;*
2. *R is the* $\mathsf{Interval}^{\uparrow}$ *membership row for the closed bounded source interval $[A, B]$;*
3. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite dyadic-cover row whose centers and mesh are carried by a finite* $\mathsf{ProbeBundle}$*;*
4. *D is the* $\mathsf{DyadicCover}^{\uparrow}$ *mesh ledger relating the interval endpoints, dyadic centers, and half-radius coverage rows;*
5. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius ledger giving one positive radius below every dyadic cover radius used by C and D;*
6. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row for the displayed map on the interval source;*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row carrying the tolerance-to-precision history produced from L and M;*
8. *H records componentwise* hsame *transport for endpoints, membership, dyadic cover, Lebesgue radius, pointwise modulus, and uniform-modulus output;*
9. *Q records the displayed* Cont *routes through which interval, compact-metric, continuous-map, dyadic-cover, Lebesgue-number, and uniform-modulus consumers read the packet;*

10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{Interval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, sequential compactness, a host minimum over radii, quotient real equality, or an external modulus function. Every public read of the interval Heine–Cantor route must pass through the displayed finite dyadic cover and finite Lebesgue-number ledger.

Theorem 1253.2 (Closed bounded interval uniform-modulus NameCert obligations)**.** *For the carrier of Definition 1253.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier admission for the displayed packet I; classifier preservation under the componentwise* hsame *rows H; dyadic-cover coverage for every* $\mathsf{Interval}^{\uparrow}$ *member read from R; finite Lebesgue-number compatibility between D and L; and output exactness saying that the* $\mathsf{UniformModulus}^{\uparrow}$ *row U is read only through C, D, L, M, H, Q, P, N.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulusTasteGate`
`single carrier alignment`

Proof. Carrier admission is immediate from the displayed packet: every endpoint, interval-membership, dyadic-cover, Lebesgue-radius, pointwise-modulus, output, transport, route, provenance, and local naming row is one of the coordinates of I.

Classifier preservation follows by reading the componentwise hsame rows in H. These rows transport the endpoint orientation, interval membership, dyadic cover mesh, finite Lebesgue-number radius, pointwise modulus, and uniform-modulus output without adding a row outside the carrier.

The coverage and radius clauses are supplied by C, D, L. The compact-metric finite cover chooses only bundled dyadic centers from D, and the finite Lebesgue-number ledger L gives a positive radius below the displayed cover radii.

Finally the output exactness clause routes the continuous-map rows M and the finite radius ledger L into U through Q, P, N. Thus every local naming obligation is discharged by rows already visible in I. □

Theorem 1253.3 (Closed bounded interval Heine–Cantor route)**.** *Let $I = (A, B, R, C, D, L, M, U, H, Q, P, N)$ be an accepted* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *carrier. For every positive target-tolerance history read by the* $\mathsf{ContinuousMap}^{\uparrow}$ *row M, the* $\mathsf{UniformModulus}^{\uparrow}$ *output U factors through the finite dyadic cover D and the finite Lebesgue-number ledger L: a source-distance read below the precision exported by U is routed to one bundled dyadic center, then through the corresponding pointwise-modulus row of M, and finally through the target metric triangle row.*

The factorization uses the interval membership row R, compact finite-cover row C, dyadic mesh row D, Lebesgue-number row L, pointwise-modulus row M, and transport and provenance rows H, Q, P, N. It uses no open-cover compactness theorem, no host minimum, no selected subsequence, no quotient equality of reals, and no external modulus selector.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulus`
`heine cantor route rows`

Proof. Start with the interval source. The endpoint rows A, B and the membership row R identify the closed bounded interval as the only source domain visible to the packet.

Read compactness through the finite dyadic-cover rows C and D. For each source history admitted by R, the cover returns a bundled dyadic center and a displayed mesh-radius comparison; the center is a row of the finite ProbeBundle, not an externally chosen real point.

Use the finite Lebesgue-number ledger L to choose the precision handed to the uniform-modulus output. Since L is below every displayed dyadic cover radius, any pair of interval histories whose source distance is below the exported precision remains inside one cover member after transport through H.

At that bundled center, the ContinuousMap$^{\uparrow}$ row M supplies the pointwise modulus for the requested tolerance. The target metric triangle row then combines the two pointwise estimates through the same center, and the route rows Q, P, N repack the result as the UniformModulus$^{\uparrow}$ output U.

All rows used in the argument are coordinates of I. Hence the Heine–Cantor interval route is realized by finite dyadic cover data plus the finite Lebesgue-number ledger, with no appeal to an open cover, host minimum, selected subsequence, quotient real equality, or external analytic modulus. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulusHeineCant`

Theorem 1253.4 (Closed bounded interval dyadic-cover scope)**.** *For every accepted* ClosedBoundedIntervalUniformM *carrier $I = (A, B, R, C, D, L, M, U, H, Q, P, N)$, the endpoint rows A, B, closed-interval membership row R, compact finite-cover row C, and dyadic-cover ledger D are scoped entirely to the source side of the interval route. Any consumer of the uniform-modulus output must read these rows before it may inspect the finite Lebesgue-number ledger or the* UniformModulus$^{\uparrow}$ *row.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulus dyadic cover scope`

Proof. Project the carrier. The endpoint rows A, B and interval membership row R identify the closed bounded source interval and do not mention the target modulus output.

The finite-cover rows C, D remain on the same source side. They provide bundled dyadic centers, mesh data, and coverage comparisons for interval members, but they do not supply a pointwise-modulus value or a uniform-modulus output.

Thus the dependency scope for the source side is precisely the endpoint, interval, compact finite-cover, and dyadic-cover rows. Later rows may consume this scope, but they cannot replace it with open-cover compactness, host minimum selection, quotient-real equality, or an external interval object. □

Theorem 1253.5 (Closed bounded interval Lebesgue handoff scope)**.** *For every accepted* ClosedBoundedIntervalUni *carrier $I = (A, B, R, C, D, L, M, U, H, Q, P, N)$, the pointwise-modulus row M, finite Lebesgue-number ledger L, and* UniformModulus$^{\uparrow}$ *output row U form the handoff scope from source-side dyadic covers to the public uniform-modulus read. Transport, replay, provenance, and local naming are exactly the rows H, Q, P, N.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulus lebesgue handoff scope`

Proof. Start from the source scope of Definition 1253.4. The dyadic-cover rows C, D feed the finite Lebesgue-number ledger L, which records a positive radius below the displayed cover radii.

The pointwise-modulus row M is then read at the bundled centers selected by the finite cover, and the output row U packages the resulting tolerance-to-precision history as the UniformModulus$^{\uparrow}$ consumer row.

The remaining rows H, Q, P, N preserve the same handoff under componentwise hsame transport, displayed Cont replay, provenance, and local naming. Hence the scoped dependencies are exactly Real$^{\uparrow}$, Interval$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicCover$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, UniformModulus$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows. □

Theorem 1253.6 (Closed bounded interval uniform-modulus consumer examples)**.** *Every public consumer example for an accepted* ClosedBoundedIntervalUniformModulus$^{\uparrow}$ *carrier reads the same finite Heine–Cantor route before it reaches its own output row. The* ContinuousMap$^{\uparrow}$ *consumer supplies the pointwise modulus row, the* CompactMetric$^{\uparrow}$ *consumer supplies the finite compact source, the* FiniteLebesgueNumber$^{\uparrow}$ *consumer supplies the radius ledger, the* UniformModulus$^{\uparrow}$ *consumer reads the produced tolerance-to-precision history, the* DyadicCover$^{\uparrow}$ *consumer supplies the cover mesh, and the* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumers read the regular-rational and terminal real-name seals only after those rows are displayed.*

Proof. Begin with the Heine–Cantor route of Definition 1253.3. It identifies the source interval, compact finite cover, dyadic cover, finite Lebesgue-number ledger, pointwise modulus, and uniform-modulus output as one finite route.

Use the dyadic-cover scope theorem Definition 1253.4. It places CompactMetric$^{\uparrow}$ and DyadicCover$^{\uparrow}$ before any Lebesgue-number or uniform-modulus output can be read.

Now apply the Lebesgue handoff scope Definition 1253.5. The ContinuousMap$^{\uparrow}$ row, FiniteLebesgueNumber$^{\uparrow}$ ledger, and UniformModulus$^{\uparrow}$ output form the handoff, while RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers remain downstream of the same displayed real-name route.

Thus the named consumer examples are examples of the existing finite dependency route, not additional compactness, completeness, selected-minimum, quotient-real, or host-function-equality principles. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUniformModulusUp.ClosedBoundedIntervalUniformModulus consumer examples`

Closure status for NameCert$_{\text{ClosedBoundedIntervalUniformModulus}^{\uparrow}}$.

Constructive story: The closed bounded interval uniform-modulus certificate is generated from the classical Heine–Cantor pattern, but its BEDC carrier reads only visible $BHist$ rows: closed interval bounds, finite dyadic cover data, bundled dyadic centers, pointwise continuity rows, finite Lebesgue-number ledger, and the uniform modulus row produced for the interval route. The $hsame$ relation transports those rows across equivalent histories without invoking host equality of functions or intervals. Cont allows the uniform-modulus evidence to be consumed downstream while preserving the same interval and continuity ledger. The chapter therefore records the constructive route expected of the theorem, while keeping compactness and completeness obligations explicit rather than importing them silently.

Theory closure: TheoryGate(NameCert$_{\text{ClosedBoundedIntervalUniformModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1253.3, Definition 1253.4, and Definition 1253.5 bind the public Heine–Cantor interval route, the source-side dyadic-cover rows, and the Lebesgue-to-uniform-modulus handoff to the displayed dependency scope.

Formal verification: FormalStatus(NameCert$_{\text{ClosedBoundedIntervalUniformModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim a proved compactness theorem, ambient real completeness, open-cover minimum selection, or host-level function equality beyond the displayed route rows.

Upgrade path: The mature paper route is closed over Real$^{\uparrow}$, Interval$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicCover$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, UniformModulus$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows; Lean verification and a standard bridge remain outside this chapter.

Chapter 1254

A Concrete Naming Certificate for BolzanoWeierstrass$^{\uparrow}$

Following Bolzano and Weierstrass, the BolzanoWeierstrass$^{\uparrow}$ packet records the finite interval-net and regular-subsequence route for a bounded Real$^{\uparrow}$ sequence. The construction reads boundedness through the finite mesh surface of chapter 1250, the interval modulus surface of chapter 1253, the monotone-cofinal extraction row of chapter 381, the regular rational readback of chapter 301, and the completed real-name seal of chapter 38. It is a subsequence compactness certificate for bounded real sequences, not an open-cover compactness theorem, a host completeness principle, quotient equality of streams, or a choice of arbitrary subsequences.

Definition 1254.1 (Bolzano–Weierstrass carrier)**.** *A* BolzanoWeierstrass$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, K, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* Real$^{\uparrow}$ *sequence surface, exposed by finite* StreamName$^{\uparrow}$ *windows and bounded by the interval read used by K;*
2. *K is the* ClosedBoundedIntervalNet$^{\uparrow}$ *packet naming the finite interval nets and dyadic mesh rows used to trap subsequence windows;*
3. *R is the* RegularCauchySubsequence$^{\uparrow}$ *extraction row, including the monotone-cofinal unary reindexing ledger and the selected finite windows;*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the extracted windows and their dyadic tolerance comparisons;*
5. *E is the* Real$^{\uparrow}$ *cluster seal reached only after S, K, R, Q have been displayed;*
6. *H records componentwise* hsame *transport for the bounded sequence rows, interval nets, extracted subsequence rows, regular rational readback, and real cluster seal;*
7. *C records displayed* Cont *routes from bounded sequence inspection through finite interval nets, regular Cauchy subsequence extraction, RegSeqRat readback, and RealUp sealing;*
8. *P is the* Pkg *provenance over* BolzanoWeierstrass$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, ClosedBoundedIntervalUniformMo RegularCauchySubsequence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrass}^{\uparrow}}$ *row.*
The carrier exports no open-cover compactness object, no host completeness row, no quotient equality of real names, no least cluster selector, and no choice-selected arbitrary subsequence.

Lean4 (definition) `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassUp`
Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassTasteGate single carrier alignment`

Theorem 1254.2 (Bolzano–Weierstrass NameCert obligations)**.** *For the carrier of Definition 1254.1, the local* $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (S, K, R, Q, E, H, C, P, N)$, bounded sequence admission through S, finite interval-net admission through K, interval-modulus compatibility with the closed bounded interval route, monotone-cofinal regular-subsequence extraction through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier namecert obligations`

Proof. Project the carrier. The bounded sequence row S is visible before any cluster seal is read, and the interval-net row K supplies the finite dyadic mesh and containment rows from chapter 1250.

Read the interval compatibility next. The uniform-modulus route of chapter 1253 keeps the closed bounded interval data tied to displayed dyadic covers and finite radius ledgers, so the packet does not import an open-cover object or a host minimum.

The subsequence rows close the analytic surface. The extraction row R is the monotone-cofinal route of chapter 381, while Q is the regular rational readback through chapter 301; the cluster seal E is then reached through the real-name boundary of chapter 38.

Finally the structural rows H, C, P, N transport, replay, source, and name only these displayed rows. No additional NameCert obligation remains outside the listed finite ledger. □

Theorem 1254.3 (Bolzano–Weierstrass finite subsequence obligations)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier $B = (S, K, R, Q, E, H, C, P, N)$, finite subsequence extraction is an obligation over the displayed rows S, K, R, Q, H, C, P, N. The row K supplies finite* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *selections, the row R records the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *monotone-cofinal schedule, the row Q reads the selected windows through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal remains downstream of those rows.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier finite subsequence obligations`

Proof. Project the carrier of Definition 1254.1. The bounded source S and interval-net row K are visible before any subsequence schedule or real cluster seal is consumed.

Apply Definition 1254.2. It identifies K as the finite $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ source, R as the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ extraction row, and Q as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row over finite stream windows and dyadic tolerances.

Transport and replay through H and C, then source and name the same route through P and N. The extraction therefore uses only finite interval-net selection, monotone-cofinal scheduling, regular rational readback, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and the downstream $\mathsf{Real}^{\uparrow}$ seal; it does not claim open-cover compactness or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrass tagged fromEventFlow exactness`

Theorem 1254.4 (Bolzano–Weierstrass interval nesting route)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, nested dyadic-window retention is read only through the finite* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row* K *before the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row* R *is consumed. Later retained windows refine earlier retained windows inside the same interval-net route, and the selected windows are then exposed to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *without importing open-cover compactness, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier interval nesting route`

Proof. Start with Definition 1254.2. It places finite interval-net admission through K before monotone-cofinal regular-subsequence extraction through R and regular rational readback through Q.

At each inspected depth, the retained dyadic window is a displayed child of the same $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ route. The transport row H carries the source window and interval row together, while the replay row C records the containment and dyadic tolerance facts used by the next retained window.

Only after that nesting route is displayed may R append the next monotone-cofinal index. The readback row Q then exposes the same retained windows through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$, so no open-cover compactness or ambient $\mathsf{Real}^{\uparrow}$ completeness row enters the route. □

Theorem 1254.5 (Bolzano–Weierstrass RegSeqRat handoff)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *cluster handoff factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *after finite interval selection and regular-subsequence scheduling have been displayed. The consumer route is*

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

It exports a finite regular-rational handoff, not a host completeness point, selected cluster limit, open-cover compactness theorem, or quotient equality of streams.

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regseqrat handoff`

Proof. Use Definition 1254.3 to expose finite interval-net selection, the regular-subsequence schedule, and the regular rational readback before the terminal Real seal.

Then use Definition 1254.4. The retained windows form a nested dyadic route inside K, and the row R records the monotone-cofinal schedule attached to those windows.

Now read Q. Its $\mathsf{RegSeqRat}^{\uparrow}$ surface receives exactly the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances already displayed by K and R. The row E may be consumed only after this handoff, and H, C, P, N preserve the same route.

Thus the cluster-facing output is a finite $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ handoff. No ambient completeness, selected limit, open-cover compactness, or quotient stream equality is available from the carrier. □

Theorem 1254.6 (Bolzano–Weierstrass root interval-tree obligation)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, the root interval-tree obligation sends every bounded real sequence read through the finite* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ S, K, H, C, P, N.$$

The row S *supplies finite* $\mathsf{StreamName}^{\uparrow}$ *windows for the bounded source,* K *supplies the dyadic interval-tree and* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh rows, and* H, C, P, N *transport, replay, source, and name*

only that displayed tree. The obligation refuses selected subsequence choice, open-cover compactness, quotient real equality, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier root interval tree obligation`

Proof. Project the carrier of Definition 1254.1. The bounded source row S and the interval-net row K are visible before any subsequence extraction or Real seal is consumed.

Apply Definition 1254.2. It identifies S, K, H, C, P, N as NameCert rows over bounded sequence admission, finite dyadic interval nets, componentwise transport, displayed replay, provenance, and local naming.

The excluded principles are absent from the carrier inventory. Since the interval tree is finite and dyadic, the root obligation is a displayed BEDC row rather than an open-cover compactness or choice-selected extraction principle. □

Theorem 1254.7 (Bolzano–Weierstrass root subsequence extraction obligation)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier, extracting a cofinal regular subsequence is an obligation over the displayed rows* S, K, R, Q, H, C, P, N*. The finite interval-tree row* K *supplies retained bounded windows from* S*; the extraction row* R *records the monotone-cofinal* $\mathsf{StreamName}^{\uparrow}$ *reindexing ledger; and* Q *reads those windows through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Thus the root extraction obligation has an explicit $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{NameCert}$ *surface and does not export selected subsequence choice, host compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the interval-tree obligation of Definition 1254.6. It gives the bounded source windows and dyadic retained-cell rows before the extraction ledger is read.

Use the carrier row R for monotone-cofinal reindexing and Q for the regular rational readback. These are the only rows that can name the extracted subsequence in Definition 1254.1.

Finally use H, C, P, N to transport, replay, source, and name the route. Because the subsequence is read from retained finite windows and regular rational schedules, no external selector or ambient completeness object is available. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier root subsequence extraction obligation`

Definition 1254.8 (Bolzano–Weierstrass finite monotone-subsequence window)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, a finite monotone-subsequence window is the displayed BEDC row obtained by pairing:*

1. *the retained closed bounded interval rows of* K*, read as finite* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *cells over the bounded source windows of* S*;*
2. *the monotone-cofinal reindexing ledger of* R*, read as the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *schedule that keeps the retained rows in increasing subsequence order;*
3. *the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of* Q *attached to the same retained windows;*
4. *the transport, replay, provenance, and local naming rows* H, C, P, N*.*

The window has no coordinate for a selected infinite subsequence, quotient real equality, host open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassMonotoneSubsequenceWindow single carrier alignment`

Theorem 1254.9 (Bolzano–Weierstrass monotone window regularity)**.** *Every finite monotone-subsequence window of an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier is a regular BEDC readback surface. Its retained interval rows are read through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh bounds, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback that already occur in* S, K, R, Q, H, C, P, N*.*

Consequently regularity is witnessed by the displayed finite windows and monotone-cofinal ledger, not by an external selector, a quotient stream, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier monotone window regularity`

Proof. Start from Definition 1254.8. The retained interval rows come from K, and the only admitted order on them is the monotone-cofinal schedule recorded by R.

Read the retained rows through Q. The $\mathsf{StreamName}^{\uparrow}$ coordinate supplies the finite window schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ coordinate supplies the dyadic mesh and containment bounds, and the $\mathsf{RegSeqRat}^{\uparrow}$ coordinate supplies the regular rational readback.

Finally transport the same rows through H, C, P, N. Since every regularity datum has already been displayed before the $\mathsf{Real}^{\uparrow}$ seal is inspected, the theorem proves regularity of the finite window surface rather than a host-level subsequence choice principle. □

Theorem 1254.10 (Bolzano–Weierstrass closed-interval cluster handoff)**.** *For an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, every cluster handoff to* $\mathsf{Real}^{\uparrow}$ *factors through a finite monotone-subsequence window. The route first reads retained* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *rows from* K*, then the monotone-cofinal* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *ledger from* R*, then the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows from* Q*, and only then reaches the seal row* E*.*

The handoff exports no open-cover compactness, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. It reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite retained rows and their displayed transport, replay, provenance, and local* $\mathsf{NameCert}$ *coordinates.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier closed interval cluster handoff`

Proof. Apply Definition 1254.6 to obtain the finite retained interval tree and bounded source windows. These rows are the only interval data exposed by K.

Use Definition 1254.7 to read the monotone-cofinal schedule through R and the regular rational surface through Q. The preceding regularity theorem identifies that surface as the finite monotone-subsequence window.

Only after these rows are displayed may the cluster row E be consumed. The support rows H, C, P, N preserve the same route, so the cluster handoff remains a finite closed-interval derivation surface and never becomes an open-cover or ambient-completeness assertion. □

Theorem 1254.11 (Bolzano–Weierstrass root Real-seal obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *cluster endpoint exposed by an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier is routed through the root interval-tree obligation and the root subsequence extraction obligation before the seal row* E *is read. The cluster endpoint is therefore a displayed Real seal over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, not a host-level completeness point.*

The obligation refuses selected subsequence choice, open-cover compactness, quotient real equality, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1254.6. It supplies the finite dyadic retained-cell tree and bounded source windows.

Apply Definition 1254.7. It extracts the monotone-cofinal schedule and reads it as the $\mathsf{RegSeqRat}^{\uparrow}$ row consumed before the Real seal.

Only then read the cluster row E from Definition 1254.1. The support rows H, C, P, N preserve the same finite route, so the displayed $\mathsf{Real}^{\uparrow}$ seal cannot turn into a selected host endpoint or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassRootRealSealObligation`

Theorem 1254.12 (Bolzano–Weierstrass bounded-subsequence cluster obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, each bounded-subsequence cluster consumer must traverse the concrete rows*

$$S, K, R, Q, E, H, C, P, N$$

in the BEDC route through $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$*,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The source row* S *displays bounded real-sequence windows,* K *retains finite closed-bounded interval cells,* R *records the monotone-cofinal subsequence ledger,* Q *reads the retained windows as regular rational data, and* E *is the only* $\mathsf{Real}^{\uparrow}$ *cluster seal exposed after those rows.*

The obligation exports no selected subsequence choice, quotient real equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded subsequence cluster obligation`

Proof. Project the carrier of Definition 1254.1. The displayed rows already separate the bounded source S, interval-net packet K, extraction ledger R, regular readback Q, Real seal E, and structural rows H, C, P, N.

Apply Definition 1254.6. It places every bounded source read behind the finite $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ route and the dyadic interval cells retained by K.

Apply Definition 1254.7. The retained windows are then consumed by R as a $\mathsf{RegularCauchySubsequence}^{\uparrow}$ schedule and by Q through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Finally use Definition 1254.11. Since E is downstream of those rows and H, C, P, N only transport, replay, source, and name the same route, the bounded-subsequence cluster surface cannot expose a selected subsequence, quotient equality, open-cover compactness proof, or ambient completeness row. □

Theorem 1254.13 (Bolzano–Weierstrass cluster Real-seal factorization)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *cluster seal read from* B *factors first through the closed-bounded interval net row* K*, then through the regular Cauchy subsequence row* R*, then through the finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface inside* Q*, and only then through* E, H, C, P, N*.*

Equivalently, the cluster seal is a displayed factorization

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

over the same concrete rows $S, K, R, Q, E, H, C, P, N$*. It is not a quotient-real equality witness, not an open-cover compactness principle, not a selected subsequence, and not ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster real seal factorization`

Proof. Begin with Definition 1254.12. It displays the complete carrier route and identifies the only rows available to a cluster consumer.

The interval-net part is supplied by K. The regular-subsequence part is supplied by R, whose retained windows are the same finite windows obtained from S through K.

The readback part is supplied by Q. It passes the retained finite windows through $\mathsf{StreamName}^{\uparrow}$, reads dyadic tolerances through $\mathsf{DyadicRatCore}^{\uparrow}$, and exposes the regular rational surface through $\mathsf{RegSeqRat}^{\uparrow}$ before the cluster row E can be consumed.

Rows H, C, P, N preserve the displayed route by transport, replay, provenance, and local naming. Therefore the factorization reaches the $\mathsf{Real}^{\uparrow}$ seal only through the named finite rows and cannot introduce quotient real equality, selected subsequence choice, open-cover compactness, or ambient completeness. □

Theorem 1254.14 (Bolzano–Weierstrass interval-net choice-free retention)**.** *For every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, the retained interval windows used by the cluster route are retained inside the displayed* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row* K *before the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row* R *is read. Each retained window is then passed through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *using only the same* S, K, R, Q, H, C, P, N *support.*

Thus interval retention is a finite NameCert projection from the carrier, not a countable-choice selection of subsequence indices, not quotient real equality, not open-cover compactness, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite interval tree supplied by Definition 1254.6. The row K displays the closed-bounded interval net and the retained cells before any subsequence schedule is consumed.

Use Definition 1254.7. The row R reads only the retained windows already exposed by K, so its monotone-cofinal schedule is a projection of the finite interval-net ledger rather than an external choice function.

Read the retained windows through Q and then through E. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows are therefore downstream of the same retained interval data, and the Real seal is reached only after that readback.

Finally transport and name the route with H, C, P, N. Since no carrier row stores a choice-selected subsequence, quotient equality, open-cover compactness proof, or ambient completeness principle, the interval-net retention remains choice-free. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier interval net choice free retention`

Theorem 1254.15 (Bolzano–Weierstrass bounded sequence source)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier reads bounded* $\mathsf{Real}^{\uparrow}$ *sequence data through the finite source route*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{NameCert}$$

before any interval-net row is consumed. The bounded source row S *exposes only scheduled stream windows, dyadic tolerance comparisons, regular rational readback, and the terminal real-name seal needed to feed* K, R, Q, E, H, C, P, N*. It does not expose selected subsequence choice, quotient real equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded sequence source`

Proof. Project the carrier $B = (S, K, R, Q, E, H, C, P, N)$ from Definition 1254.1. The source coordinate S is the only row that can supply bounded sequence observations before interval nets are opened.

Read S through the displayed stream, dyadic, regular rational, and real rows. The $\mathsf{StreamName}^{\uparrow}$ windows give finite sequence observations, $\mathsf{DyadicRatCore}^{\uparrow}$ gives the tolerance comparisons, $\mathsf{RegSeqRat}^{\uparrow}$ gives the regular readback, and $\mathsf{Real}^{\uparrow}$ seals the bounded real-name surface.

Only after that source route is displayed may the interval-net row K be consumed. The structural rows H, C, P, N transport, replay, source, and name the same route, leaving no coordinate for selected subsequence choice, quotient real equality, open-cover compactness, or ambient completeness. □

Theorem 1254.16 (Bolzano–Weierstrass interval-net obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier* $B = (S, K, R, Q, E, H, C, P, N)$*, the rows* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *and* $\mathsf{ClosedBoundedIntervalUniformMod}$ *are the only interval-tree source for retained windows. Every retained window consumed by* R *or* Q *must first be read through* K *and its uniform-modulus compatibility route, and the route refuses selected subsequence choice, quotient real equality, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier interval net obligation`

Proof. Start from the interval-net coordinate K. It is the only carrier row naming finite closed bounded interval nets, dyadic mesh cells, and retained windows.

The uniform-modulus compatibility is read inside the same interval-tree route. It binds the retained windows to displayed dyadic refinements and prevents a consumer from substituting an open-cover compactness proof or a host finite-subcover selector.

Rows R and Q consume retained windows only after K has supplied them. Transport and replay through H and C, followed by provenance and local naming through P, N, preserve the same interval-tree source and add no quotient real equality, selected subsequence choice, or ambient completeness coordinate. □

Theorem 1254.17 (Bolzano–Weierstrass regular subsequence window readback)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route consumes* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the* $\mathsf{Real}^{\uparrow}$ *cluster seal. The row* R *records the monotone-cofinal retained-window extraction from the interval tree, the row* Q *reads those windows as regular rational data, and only then may* E *be read as the Real cluster seal. The readback exports no selected subsequence choice, quotient real equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regular subsequence window readback`

Proof. Use Definition 1254.16 to obtain retained windows from K. These windows are finite interval-tree data rather than host subsequence choices.

Read the extraction row R next. It records the monotone-cofinal $\mathsf{RegularCauchySubsequence}^{\uparrow}$ route, so the subsequence window is named by displayed reindexing data.

Read Q before E. The regular rational readback turns the extracted windows into the $\mathsf{RegSeqRat}^{\uparrow}$ surface consumed by the Real cluster seal, and the rows H, C, P, N preserve that order.

Thus the Real seal is downstream of regular-subsequence and regular-rational readback. None of the refused principles appears in the carrier rows or in the transport and replay route. □

Theorem 1254.18 (Bolzano–Weierstrass regular subsequence route)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier routes a bounded real sequence to a regular Cauchy subsequence through the finite path*

$$S, K, R, Q, H, C, P, N.$$

The path first reads the bounded sequence surface S, then the finite interval-net packet K, then the monotone-cofinal extraction row R, and finally the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q. The route does not expose a choice-selected arbitrary subsequence, a quotient-stream representative, a host compactness theorem, or an ambient completeness field.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regular subsequence route`

Proof. Start with the bounded sequence row. The only boundedness data visible to the carrier is the finite sequence surface S together with the closed bounded interval net K.

Apply the interval-net route of chapter 1250. Its dyadic mesh rows supply nested finite traps for inspected sequence windows, and its Real-facing seal remains downstream of the displayed mesh and readback rows.

Now use the regular-subsequence route of chapter 381. The row R supplies a monotone-cofinal unary reindexing ledger, so the subsequence is read by finite reindexing data rather than by an arbitrary choice function.

The row Q reads the extracted windows as a $\mathsf{RegSeqRat}^{\uparrow}$ surface. Transport and replay through H and C, followed by P and N, preserve that same route, leaving no coordinate for quotient streams, host compactness, or ambient completeness. □

Theorem 1254.19 (Bolzano–Weierstrass bounded source factorization)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier reads its bounded sequence row S through the finite* $\mathsf{BoundedRealSequence}^{\uparrow}$ *route of chapter 1415 before it consumes the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row K, the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal. Thus the bounded-source surface of the Bolzano–Weierstrass packet is not a free host-level sequence input; it is the reusable finite certificate supplied by Definition 1415.4.*

Proof. Apply Definition 1415.4. It exposes the bounded real sequence source by the finite route $S, W, Q, R, I, H, C, P, N$, with stream windows, regular readback, a $\mathsf{Real}^{\uparrow}$ seal, and an interval-net bound row already displayed inside the $\mathsf{BoundedRealSequence}^{\uparrow}$ certificate.

Feed that certified source into Definition 1254.18. The Bolzano–Weierstrass carrier can then consume K for closed bounded interval nets, R for monotone-cofinal regular subsequence extraction, Q for $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal without adding another bounded-sequence coordinate.

The composite route leaves no slot for open-cover compactness, host completeness, quotient stream equality, or a choice-selected subsequence. Those principles are not transported by the $\mathsf{BoundedRealSequence}^{\uparrow}$ handoff, and the regular-subsequence route only consumes the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded source factorization`

Theorem 1254.20 (Bolzano–Weierstrass bounded cluster route)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier whose bounded source is supplied by* $\mathsf{BoundedRealSequence}^{\uparrow}$ *and whose terminal cluster candidate is consumed by* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *factors through the concrete route*

$$\mathsf{BoundedRealSequence}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealS}$$

The same finite interval-tree evidence is preserved by the displayed hsame, Cont, Pkg, *and* NameCert *rows, so the cluster consumer receives no detached bounded sequence, arbitrary subsequence selector, quotient stream equality, open-cover compactness, or ambient completeness row.*

Proof. Start with the bounded-source factorization of Definition 1254.19. It places the source row S behind the finite $\mathsf{BoundedRealSequence}^{\uparrow}$ handoff of Definition 1415.4 before the interval-net row K is consumed.

Use the interval-tree induction and regular-Cauchy subsequence readback blocks. The row K supplies the retained closed-bounded interval cells, R records the monotone-cofinal $\mathsf{RegularCauchySubsequence}^{\uparrow}$ ledger, and Q reads the selected windows as $\mathsf{RegSeqRat}^{\uparrow}$ data.

Now consume the terminal cluster candidate through the public export of Definition 1580.4. The $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ route reads the same bounded source, cofinal subsequence, selected windows, regular readback, dyadic tolerance ledger, and $\mathsf{Real}^{\uparrow}$ seal before a cluster point is exposed.

Finally H, C, P, N transport, replay, source, and name that same finite interval-tree evidence. Since no second source, selector, quotient stream, open-cover compactness theorem, or ambient completeness row is present, the bounded cluster route is exactly the displayed composite. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded cluster route`

Theorem 1254.21 (Bolzano–Weierstrass finite interval-tree induction)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. For every inspected unary depth of the closed-bounded interval net, the displayed finite dyadic refinement tree in K determines, by induction on that tree, a retained cell, a nonempty finite sequence-window row from S, and a monotone-cofinal reindexing row in R. The retained cells refine through the same* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *and* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *rows, and the extracted finite windows are exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumed by Q.*

Proof. For depth 0, read the root bounded interval from S and the root net row from K. The boundedness ledger supplies a nonempty inspected sequence-window row in S, and the identity unary index row is monotone and cofinal on the displayed root window.

For the induction step, assume the retained cell and window have been displayed at depth d. The dyadic mesh row in K splits that cell into its finite children at depth $d + 1$, and the packet ledger witnesses that at least one child retains a nonempty finite sequence-window row from S.

Choose that retained child only by the displayed finite branch witness. Componentwise transport through H carries the previous route to the child cell, while displayed replay through C records the same containment and radius facts in the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ and $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ rows.

The reindexing row in R appends the retained-window index and records monotonicity and cofinality in the unary ledger. Reading the selected windows through Q gives precisely the regular rational schedule consumed as $\mathsf{RegSeqRat}^{\uparrow}$, so the subsequence route is derived from finite interval refinement rather than from an external selector.

No open-cover compactness, arbitrary subsequence choice, quotient stream, or host completeness row is used. The induction uses only the finite tree displayed by K, the inspected windows in S, the monotone-cofinal ledger in R, and the readback route in Q. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier finite interval tree induction`

Theorem 1254.22 (Bolzano–Weierstrass regular-Cauchy subsequence readback)**.** *Let $B = (S, K, R, Q, E, H, C, P,$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The finite retained cells supplied by Definition 1254.21*

force the monotone-cofinal schedule in R and the $\mathsf{RegSeqRat}^{\uparrow}$ *readback in Q through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Thus the regular Cauchy subsequence is read from finite interval-tree evidence and retained windows, with no selected subsequence, quotient stream representative, host compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the bounded source factorization of Definition 1254.19. It ensures that the source row S is already a displayed bounded real-sequence surface before the interval tree is consumed.

Apply Definition 1254.21. At each inspected depth it supplies a retained cell, a nonempty finite window from S, and a monotone-cofinal ledger entry in R.

Use Definition 1254.18. The retained finite windows are the only windows handed to the regular subsequence route, so the $\mathsf{StreamName}^{\uparrow}$ schedule is determined by the displayed ledger rather than by an external selector.

Finally read the selected windows through Q. The dyadic tolerances are carried by the same interval-tree rows in K, the windows are the same finite $\mathsf{StreamName}^{\uparrow}$ rows recorded by R, and the readback is exactly the $\mathsf{RegSeqRat}^{\uparrow}$ surface consumed by the carrier. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regular cauchy subsequence readback`

Theorem 1254.23 (Bolzano–Weierstrass cluster Real-seal handoff)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The cluster row E is read only after the regular-Cauchy subsequence readback of Definition 1254.22. Consequently the cluster point exported by B is a displayed* $\mathsf{Real}^{\uparrow}$ *seal over S, K, R, Q, H, C, P, N, not an ambient completeness point, not a selected limit, not quotient equality, and not an open-cover compactness witness.*

Proof. Begin with the root Real-seal obligation of Definition 1254.11. It places every $\mathsf{Real}^{\uparrow}$ cluster endpoint downstream of the root interval-tree obligation and root subsequence extraction obligation.

Apply Definition 1254.22. The theorem displays the retained cells, monotone-cofinal ledger, finite stream windows, dyadic tolerances, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before E is consumed.

Now use Definition 1254.26. The row E seals that already displayed regular Cauchy readback as a $\mathsf{Real}^{\uparrow}$ boundary, and the structural rows H, C, P, N only transport, replay, source, and name the same finite route.

Therefore the cluster handoff exports a displayed Real seal and nothing stronger. The carrier contains no coordinate for ambient completeness, selected limits, quotient equality, open-cover compactness, or arbitrary subsequence choice. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster real seal handoff`

Theorem 1254.24 (Bolzano–Weierstrass retained-cell monotonicity)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. If two inspected unary depths are read from the same finite closed-bounded interval net row K, and the second depth is later in the displayed unary order, then the retained cell at the later depth refines the retained cell at the earlier depth through the same* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *and* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *rows. The nonempty sequence-window row supplied by S for the later depth is a displayed continuation of the earlier retained window before the monotone-cofinal reindexing row R is consumed.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier retained cell monotonicity`

Proof. Induct on the finite refinement tree supplied by Definition 1254.21. At the root depth, the retained cell is the root closed bounded interval read from S and K, and its nonempty inspected window is the initial window displayed by the bounded-sequence surface.

For the successor step, the induction hypothesis gives the earlier retained cell, its nonempty window, and its refinement ledger inside K. The finite tree row chooses a child retained cell at the later depth together with a nonempty finite sequence-window row, and the child containment is read through the same closed-bounded interval net and interval-modulus rows.

Transport the earlier route componentwise by H and replay the displayed continuation by C. This records that the later window extends the earlier retained-window display before the ledger R appends the later index. Thus the refinement relation and the sequence-window continuation are already present in the finite interval-net surface when the monotone-cofinal reindexing row is read. □

Theorem 1254.25 (Bolzano–Weierstrass cofinal window extraction)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The retained-cell nesting of Definition 1254.24 supplies the nested cofinal ledger in R, and the selected finite sequence windows are exactly the windows handed to the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q. Thus the regular readback route is fixed by the retained interval-cell ledger and uses no arbitrary subsequence choice, quotient stream, host compactness, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cofinal window extraction`

Proof. Project the accepted carrier through Definition 1254.1. The visible coordinates are the bounded sequence surface S, the finite interval-net packet K, the extraction row R, the readback row Q, and the structural rows H, C, P, N.

Apply Definition 1254.21 at each inspected unary depth. It supplies a retained interval cell, a nonempty finite sequence-window row from S, and a displayed refinement relation through K and the interval-modulus route.

Apply Definition 1254.24 to the retained cells at successive inspected depths. Refinement in K preserves the nesting of retained cells, while interval-modulus compatibility preserves the requested finite mesh bound; together these rows keep the ledger monotone and cofinal inside the displayed extraction surface.

Finally read the selected windows through Q. Since Q is attached to the extracted windows of R, and R was built only by appending retained finite-window indices, the $\mathsf{RegSeqRat}^{\uparrow}$ readback receives precisely those windows and no additional subsequence selector or external completeness route. □

Theorem 1254.26 (Bolzano–Weierstrass Real cluster seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *cluster read from an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier factors through*

$$S, K, R, Q, E, H, C, P, N.$$

The cluster row E is reached only after the bounded sequence surface, finite interval-net packet, cofinal-window extraction ledger, regular Cauchy subsequence extraction row, and RegSeqRat readback have been displayed. Consequently a real-facing consumer receives a finite subsequence compactness seal and receives no open-cover compactness principle, host completeness row, quotient equality of streams, least cluster selector, or choice-selected arbitrary subsequence.

Proof. Begin with the factorization in Definition 1254.18. It displays S, K, R, Q before any real cluster row can be consumed.

Then use Definition 1254.25. The row R is a monotone-cofinal ledger built from retained interval cells, and Q reads exactly the selected finite sequence windows supplied by that ledger.

Use the $\mathsf{Real}^{\uparrow}$ chapter chapter 38 only after that route is visible. The seal E is therefore a real-name boundary for the extracted regular Cauchy readback, not a separate host limit or quotient representative.

The transport and continuation rows preserve the same finite route. Since the carrier contains no coordinate for open-cover compactness, host completeness, quotient equality, a least cluster selector, or an arbitrary subsequence choice, the rows $S, K, R, Q, E, H, C, P, N$ exhaust the cluster seal. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier real cluster seal`

Theorem 1254.27 (Bolzano–Weierstrass bounded window induction)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The bounded-window induction consumed by the packet is the finite interval-tree induction of Definition 1254.21 restricted to the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row S and the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row K. At each inspected unary depth it exposes a retained closed interval cell, a nonempty bounded sequence window, and the next monotone-cofinal ledger entry before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *cluster seal is consumed.*

Proof. Start with the bounded source factorization. It reads the bounded $\mathsf{Real}^{\uparrow}$ sequence through the finite $\mathsf{BoundedRealSequence}^{\uparrow}$ surface before the interval-net row is consumed.

Apply Definition 1254.21. The induction step uses only the finite children displayed by K and the nonempty inspected windows displayed by S.

Apply Definition 1254.24. The retained cells and their bounded source windows refine along the same closed bounded interval net, so the next ledger entry in R is attached to a displayed bounded window rather than to an external subsequence selector. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded window induction`

Theorem 1254.28 (Bolzano–Weierstrass cofinal cluster seal)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier whose bounded-window induction row is displayed. The extracted cofinal windows pass through the consumer chain*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using only the rows R, Q, E, H, C, P, N. The terminal cluster seal is therefore a $\mathsf{Real}^{\uparrow}$ *boundary for the retained cofinal windows, not a quotient stream representative, selected limit, or ambient completeness point.*

Proof. Apply Definition 1254.25. It identifies the row R as the monotone-cofinal ledger built from retained bounded windows and identifies Q as the $\mathsf{RegSeqRat}^{\uparrow}$ readback of those windows.

Use the stream and dyadic rows already present in the carrier provenance. The same selected finite windows are read as $\mathsf{StreamName}^{\uparrow}$ windows, and their tolerance comparisons are routed through $\mathsf{DyadicRatCore}^{\uparrow}$ before any cluster endpoint is sealed.

Finally apply Definition 1254.26. Since E is reached only after the retained windows, regular readback, stream windows, and dyadic tolerances are displayed, the cluster seal stays inside the finite cofinal route. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cofinal cluster seal`

Theorem 1254.29 (Bolzano–Weierstrass ledger refusal)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier refuses any consumer ledger that asks for arbitrary subsequence choice, quotient stream equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Such a ledger cannot be filled from* $S, K, R, Q, E, H, C, P, N$ *because those rows expose only bounded source windows, finite interval nets, monotone-cofinal reindexing, regular rational readback, a Real cluster seal, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 1254.1. Its rows are exactly $S, K, R, Q, E, H, C, P, N$, and none of those rows is an arbitrary subsequence selector, quotient stream equality proof, open-cover compactness object, or ambient completeness theorem.

Apply Definition 1254.27 and Definition 1254.28. They show that every cluster-facing route is built from finite bounded windows, retained interval cells, cofinal readback, StreamName windows, DyadicRatCore tolerances, and the terminal Real seal.

The structural rows H, C, P, N only transport, replay, source, and name that displayed route. Therefore a consumer ledger containing any of the excluded principles has no carrier coordinate from which its requested row could be produced. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier ledger refusal`

Theorem 1254.30 (Bolzano–Weierstrass cluster seal extraction)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The cluster seal* E *is extracted only by composing the finite interval-tree induction, retained-cell monotonicity, cofinal-window extraction, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route displayed by* S, K, R, Q, H, C, P, N*. The resulting consumer-facing output is a* $\mathsf{Real}^{\uparrow}$ *cluster seal for the displayed bounded sequence, and it exports no open-cover compactness principle, host completeness row, quotient-stream equality, least-cluster selector, or arbitrary subsequence choice.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster seal extraction`

Proof. Apply Definition 1254.21. It supplies the retained finite interval cells, the nonempty inspected windows, and the displayed finite-branch witness that keeps the route inside the closed-bounded interval net surface.

Apply Definition 1254.24. The retained cells form the nested refinement ledger, and the continuation rows preserve the same bounded-sequence windows through the later inspected depths.

Apply Definition 1254.25. The ledger R is monotone and cofinal, while Q reads exactly the selected finite windows as the regular rational schedule consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

Finally apply Definition 1254.26. The terminal row E is therefore a $\mathsf{Real}^{\uparrow}$ boundary for the extracted regular Cauchy readback, so no open-cover compactness, host completeness, quotient-stream equality, least-cluster selector, or arbitrary subsequence choice is exported to the completion-facing consumer. □

Theorem 1254.31 (Bolzano–Weierstrass tail-modulus window stability)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The retained interval cells selected from* K *and their dyadic mesh moduli transport through the same* H *and* C *rows to the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback windows in* Q*. In particular, changing an accepted packet by componentwise* hsame *transport may rename the displayed bounded source, retained-cell ledger, and modulus rows, but it cannot change which finite source windows are read by* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is consumed.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier tail modulus window stability`

Proof. Apply Definition 1254.21. It attaches each inspected unary depth to a retained interval cell, a nonempty bounded source window from S, and a displayed dyadic mesh row in K.

Apply Definition 1254.24. Later retained cells refine earlier cells through the same interval-net and interval-modulus rows, so the mesh modulus remains tied to the same finite refinement ledger.

Apply Definition 1254.25. The cofinal ledger R records exactly the selected source windows, and the row Q reads precisely those windows as the $\mathsf{RegSeqRat}^{\uparrow}$ schedule.

The rows H and C transport and replay only this displayed route. Thus any accepted transported packet preserves the retained-cell modulus route and cannot redirect the regular rational readback to a different finite window surface. □

Theorem 1254.32 (Bolzano–Weierstrass Real-seal nonescape)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted $\mathsf{BolzanoWeierstrass}^{\uparrow}$ carrier. The $\mathsf{Real}^{\uparrow}$ cluster seal E factors only through $S, K, R, Q, E, H, C, P, N$. Hence the seal cannot export open-cover compactness, host completeness, quotient stream equality, or arbitrary subsequence choice to any Real-facing or completion-facing consumer.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster seal real nonescape`

Proof. Apply Definition 1254.30. The seal is extracted by composing finite interval-tree induction, retained-cell monotonicity, cofinal window extraction, and $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Apply Definition 1254.26. The row E is reached only after S, K, R, Q have displayed the bounded sequence surface, retained interval cells, cofinal reindexing row, and regular rational readback.

The remaining rows H, C, P, N provide componentwise transport, displayed replay, provenance, and local naming for that same route. None of these rows contains an open-cover compactness object, host completeness theorem, quotient stream equality, or arbitrary subsequence selector. □

Theorem 1254.33 (Bolzano–Weierstrass cofinal-subsequence classifier exactness)**.** *Let $B = (S, K, R, Q, E, H, C, P$ and $B' = (S', K', R', Q', E', H', C', P', N')$ be accepted $\mathsf{BolzanoWeierstrass}^{\uparrow}$ carriers. Their cofinal-subsequence classifiers agree exactly when the bounded source windows, retained interval cells, cofinal reindexing rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and $\mathsf{Real}^{\uparrow}$ cluster seal rows agree under the displayed H, C, P, N and H', C', P', N' transport. No further classifier coordinate is available outside those finite rows.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cofinal subsequence classifier exactness`

Proof. For the forward direction, read classifier agreement through Definition 1254.25. The accepted classifier must preserve the retained source windows in S, the retained cells in K, the cofinal ledger in R, and the readback row Q.

Then apply Definition 1254.30. Agreement of the extracted subsequence route determines the terminal Real cluster seal row E because that seal is downstream of the same finite route.

For the reverse direction, assume the listed rows agree under the displayed transports. The finite interval-tree induction and retained-cell monotonicity reconstruct the same retained-cell route, while cofinal window extraction reconstructs the same R, Q readback surface.

Finally use Definition 1254.32. Since the Real seal and structural rows exhaust the classifier surface, the two accepted packets have the same cofinal-subsequence classifier and no hidden host-level equality or choice coordinate remains. □

Theorem 1254.34 (Bolzano–Weierstrass dyadic-mesh subsequence witness)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* BolzanoWeierstrass$^{\uparrow}$ *carrier. At every inspected unary depth, the finite closed-bounded interval net row K supplies a retained dyadic cell, a nonempty finite sequence-window row from S, and the next monotone-cofinal subsequence index recorded by R. The witness is read through the displayed* ClosedBoundedIntervalNet$^{\uparrow}$ *and* ClosedBoundedIntervalUniformModulus$^{\uparrow}$ *rows, and it exports no open-cover compactness, quotient stream equality, arbitrary choice of subsequence, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier dyadic mesh subsequence witness`

Proof. Apply Definition 1254.21. It reads the root interval, the finite dyadic refinements, and the nonempty inspected windows from S and K.

At each inspected depth, use the retained-cell witness supplied by the finite tree. The construction chooses only among displayed finite children and records the chosen child as a retained dyadic cell, so the subsequence index is a row of R rather than a host-level selector.

Transport and replay by H and C preserve the same interval-net and sequence-window rows. Since no carrier coordinate contains an open cover, quotient stream equality, arbitrary subsequence choice, or ambient completeness, the dyadic-mesh witness exports none of those principles. □

Theorem 1254.35 (Bolzano–Weierstrass bounded-window nonempty extraction)**.** *Let $B = (S, K, R, Q, E, H, C, P$ be an accepted* BolzanoWeierstrass$^{\uparrow}$ *carrier. Every retained dyadic cell selected from the finite interval net K carries a nonempty bounded sequence-window row from S, and this row is the only source from which R may append the next monotone-cofinal subsequence index. The extraction therefore passes through the finite interval-tree induction, retained-cell monotonicity, and dyadic-mesh subsequence witness before any selected window is handed to Q.*

Proof. Apply Definition 1254.21. At each inspected unary depth it supplies a retained cell together with a nonempty finite sequence-window row from S and the corresponding monotone-cofinal reindexing row in R.

Apply Definition 1254.24. Later retained cells refine earlier retained cells through the same interval-net rows, and the later nonempty window is a continuation of the earlier bounded window before R appends the next index.

Finally apply Definition 1254.34. The witness records that the next subsequence index is attached to the retained dyadic cell and its displayed nonempty window from S. Since the carrier has no further source coordinate for subsequence indices, R can append the next monotone-cofinal index only from that bounded window row. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded window nonempty extraction`

Theorem 1254.36 (Bolzano–Weierstrass cluster Cauchy window)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* BolzanoWeierstrass$^{\uparrow}$ *carrier. The retained-cell nesting from K yields the regular Cauchy window consumed by the* RegSeqRat$^{\uparrow}$ *row Q: later retained cells refine earlier retained cells, their dyadic diameters follow the inspected unary depth, and the selected windows recorded by R are exactly the finite windows read by Q.*

Proof. Apply Definition 1254.24. It records that retained cells at later inspected depths refine the earlier retained cells through the same closed-bounded interval-net and interval-modulus rows.

Then apply Definition 1254.25. The monotone-cofinal ledger R appends only indices attached to those retained cells, and Q reads precisely the resulting finite sequence windows.

The dyadic diameter control is therefore a displayed regular Cauchy window. It is consumed by $\mathsf{RegSeqRat}^{\uparrow}$ before the Real cluster seal is read, and no quotient stream equality, arbitrary choice, or ambient completeness row is introduced. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster cauchy window`

Theorem 1254.37 (Bolzano–Weierstrass finite interval-subsequence exhaustion)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet whose bounded* $\mathsf{Real}^{\uparrow}$ *sequence row, closed bounded interval-net route, dyadic mesh row, regular-subsequence row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows are displayed exports its cluster witness only through the finite interval-tree induction route plus cofinal window extraction. In particular, the cluster route is exhausted by Definition 1254.21, Definition 1254.24, Definition 1254.25, and Definition 1254.36, and it exports no arbitrary subsequence choice or host* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier finite interval subsequence exhaustion`

Proof. Apply Definition 1254.21. It supplies, at each inspected unary depth, the retained dyadic cell, nonempty source window, and monotone-cofinal reindexing row from the displayed interval tree.

Apply Definition 1254.24. The retained cells refine through the same closed bounded interval-net route, so the later windows are continuations of the earlier finite windows rather than outputs of an external selector.

Apply Definition 1254.25. The cofinal ledger in R is built from those retained windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q reads exactly that finite window surface.

Finally apply Definition 1254.36. The Cauchy window consumed by Q is already determined by the retained-cell ledger, while H, C, P, N only transport, replay, source, and name the displayed rows. Hence the cluster witness has no remaining route through arbitrary subsequence choice or host $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1254.38 (Bolzano–Weierstrass subsequence nonescape)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier. The subsequence route exported by* R *and* Q *remains inside the retained dyadic mesh supplied by* K *until the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is consumed. Consequently the resulting subsequence route exports no open-cover compactness, quotient stream equality, arbitrary choice of subsequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1254.34. It attaches every selected subsequence index to a retained dyadic cell and a displayed finite sequence-window row.

Apply Definition 1254.36. The retained cells form the Cauchy window consumed by Q, so the subsequence route is fixed by the finite mesh and the monotone-cofinal ledger.

Finally apply Definition 1254.30. The seal E is reached only after S, K, R, Q, H, C, P, N have displayed that route, leaving no export channel for open-cover compactness, quotient stream equality, arbitrary subsequence choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier subsequence nonescape`

Theorem 1254.39 (Bolzano–Weierstrass bounded-subsequence Cauchy extraction)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet whose bounded-source row, finite interval-tree row, retained-cell monotonicity row, cofinal-window extraction row, and dyadic-mesh subsequence row are displayed. Then B exports a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing Cauchy subsequence route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The route is read only through $S, K, R, Q, E, H, C, P, N$ and supplies no selected subsequence choice, no host compactness principle, no quotient stream equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the bounded-source factorization and the finite interval-tree induction. They expose the bounded $\mathsf{Real}^{\uparrow}$ sequence surface S and the finite dyadic refinement tree K before any subsequence row or cluster seal is consumed.

Apply retained-cell monotonicity. Later retained cells refine earlier retained cells through the same closed-bounded interval-net and interval-modulus rows, so the dyadic mesh produces a nested finite ledger rather than an external compactness argument.

Apply cofinal-window extraction and the dyadic-mesh subsequence witness. The row R records a monotone-cofinal $\mathsf{StreamName}^{\uparrow}$ reindexing ledger built from retained nonempty windows, and Q reads exactly those windows as the $\mathsf{RegSeqRat}^{\uparrow}$ schedule with $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances.

Finally use subsequence nonescape and the Real cluster seal. The terminal row E is reached only after the bounded source, retained cells, cofinal windows, and regular rational readback have been displayed, so the resulting Cauchy subsequence route remains inside $S, K, R, Q, E, H, C, P, N$ and exports none of the excluded host or choice principles. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded subsequence cauchy extraction`

Theorem 1254.40 (Bolzano–Weierstrass interval-depth witness)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet. For each displayed unary depth whose bounded source window is visible in S, the finite interval-tree row K supplies a retained closed-bounded cell, its dyadic mesh ledger, and a nonempty inspected subsequence-window witness before R appends a cofinal index. The witness is read through* BHist*,* Cont*,* hsame*, and* $\mathsf{NameCert}$ *rows and is not obtained from sequential compactness or any choice-like data.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier interval depth witness`

Proof. Induct on the finite interval tree displayed by K. The root case reads the bounded source window from S and the root closed interval from the interval-net carrier.

For the successor step, replay the finite split through C and transport the retained parent row through H. One displayed child carries the nonempty inspected window recorded by the packet, and that child supplies the next retained cell.

Only after this finite witness is displayed may R record the next monotone-cofinal subsequence entry. The argument never appeals to sequential compactness without the shown interval cell, window, transport, replay, provenance, and local name. □

Theorem 1254.41 (Bolzano–Weierstrass tail-cell stability)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet and restrict the extracted route to a later displayed tail of the monotone-cofinal schedule in R. The retained cells supplied by K remain nested under the same* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row, the restricted windows remain the windows read by Q, and componentwise* hsame *transport plus* Cont *replay preserves the same* $\mathsf{NameCert}$ *provenance. Thus subsequence restriction changes only the visible tail ledger, not the cluster package.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier tail cell stability`

Proof. Start from the interval-depth witness. Every selected index in R is attached to a retained cell and a nonempty bounded window, so deleting an initial finite prefix leaves later entries attached to later retained cells.

Use retained-cell monotonicity to compare the tail cells with the original tree. Later cells refine earlier cells through the same interval-net and interval-modulus rows, and H transports this refinement without changing the carrier surface.

Replay the restricted route through C and then read Q. The regular rational readback still consumes the retained tail windows; no new sequential compactness theorem, quotient stream, selected subsequence choice, or ambient completeness row is introduced. □

Theorem 1254.42 (Bolzano–Weierstrass choice-free cluster package)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet whose interval-depth witness and tail-cell stability rows are displayed packages a regular Cauchy subsequence for* $\mathsf{RegSeqRat}^{\uparrow}$ *through the finite route* S, K, R, Q, H, C, P, N*. The terminal cluster row* E *may be read only after that regular package is displayed, and the package contains no arbitrary subsequence selector, no sequential compactness principle without BEDC rows, no quotient real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. First build the finite retained-cell ledger from Definition 1254.40. It supplies each depth witness as a displayed interval cell, dyadic mesh row, and nonempty bounded window.

Next apply Definition 1254.41. Tail restriction preserves the same nested retained cells and keeps the selected windows inside the same transported and replayed $\mathsf{NameCert}$ surface.

Finally read Q as the $\mathsf{RegSeqRat}^{\uparrow}$ package and then read E as the downstream Real seal. Since all data pass through S, K, R, Q, H, C, P, N, the cluster route is choice-free in the BEDC sense: every subsequence and tail fact is carried by finite interval-tree evidence rather than by an external compactness or choice principle. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier choice free cluster package`

Theorem 1254.43 (Bolzano–Weierstrass finite cluster-retention ledger)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet* $B = (S, K, R, Q, E, H, C, P, N)$ *whose bounded-source row, finite interval-tree row, retained-cell row, cofinal-window row, dyadic-mesh row, Cauchy-window row, and cluster-seal row are displayed has a finite cluster-retention ledger over* $S, K, R, Q, E, H, C, P, N$*. The ledger records exactly the retained dyadic cells, their nonempty bounded sequence windows, the monotone-cofinal subsequence indices, the* $\mathsf{RegSeqRat}^{\uparrow}$ *Cauchy readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal. It exports no arbitrary subsequence choice, no selected least cluster, no open-cover compactness, no quotient stream equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with retained-cell monotonicity. It binds later retained cells to earlier retained cells through the same finite interval-tree rows in K, so the retention data is a finite ledger over the displayed interval net rather than a host compactness choice.

Apply cofinal-window extraction and the dyadic-mesh subsequence witness. The row R appends only indices attached to retained nonempty windows from S, while the dyadic mesh witness records the finite cell supporting each appended index.

Next apply the cluster Cauchy window theorem. The retained cells and their dyadic diameters determine the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the readback is therefore downstream of the same finite retention ledger.

Finally apply cluster seal extraction. The terminal row E is reached only through the displayed route S, K, R, Q, H, C, P, N, so no arbitrary subsequence choice, selected least cluster, open-cover

compactness, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness can be exported from the packet. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier finite cluster retention ledger`

Theorem 1254.44 (Bolzano–Weierstrass source-net exhaustion)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet. The bounded source row S and interval-net row K exhaust the root source surface for the cluster route: every inspected window, retained dyadic cell, mesh comparison, and bounded interval enclosure consumed by R, Q, E is read first through S, K, H, C, P, N as finite* BHist, hsame, Cont, Pkg, *and* NameCert *data. No cluster-facing consumer can reach R, Q, or E from a source row outside S or an interval-net row outside K.*

Proof. Start with the root interval-tree obligation of Definition 1254.6. It places the bounded source windows and dyadic retained cells inside the displayed finite route before any subsequence or Real seal is read.

Next apply bounded-source factorization and the interval-net obligation. The source row S is the only bounded sequence surface admitted by the carrier, and K is the only closed-bounded interval-net row that can supply retained windows to later rows.

Finally read the finite cluster-retention ledger. Since the ledger records every retained cell, nonempty source window, mesh comparison, and downstream handoff over $S, K, R, Q, E, H, C, P, N$, the root source and interval-net surface is exhausted by S and K together with the structural rows. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier source net exhaustion`

Theorem 1254.45 (Bolzano–Weierstrass RegSeqRat cluster package)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet whose source-net exhaustion row is displayed. The extraction row R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E form a single cluster package over S, K, H, C, P, N: the cofinal subsequence schedule is read from retained finite windows, the regular rational row reads exactly those windows through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the Real seal is reachable only after that regular package has been displayed.*

Proof. Use Definition 1254.44 to expose the source windows and interval-net cells available to the extraction route. This fixes the only finite data from which R may append cofinal subsequence indices.

Apply the regular subsequence route and the regular subsequence window readback theorem. They identify R as the monotone-cofinal schedule over retained windows and Q as the $\mathsf{RegSeqRat}^{\uparrow}$ readback of those same windows through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$.

Now use cluster seal extraction. The seal row E is downstream of Q, and the structural rows H, C, P, N only transport, replay, record provenance, and name the same route. Hence R, Q, E form one displayed RegSeqRat-to-Real cluster package rather than three detachable obligations. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regseqrat cluster package`

Theorem 1254.46 (Bolzano–Weierstrass root obligation package)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet $B = (S, K, R, Q, E, H, C, P, N)$ whose root interval-tree, root subsequence, Real-seal, bounded-source, and regular-subsequence routes are displayed carries the full root obligation package*

$$S, K, R, Q, E, H, C, P, N.$$

This package binds the bounded source row, interval-net row, extraction row, RegSeqRat readback row, Real seal row, componentwise hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *as the exact obligation surface for downstream compactness and Real subsequence consumers. It exports no selected least cluster, arbitrary subsequence choice, open-cover compactness, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First combine source-net exhaustion with the root interval-tree obligation. The bounded source and interval-net rows S, K expose all finite windows, retained cells, and dyadic mesh data needed by the cluster route.

Next combine the root subsequence extraction obligation with the RegSeqRat cluster package. The extraction row R and readback row Q are forced to consume the retained finite windows already displayed by S and K, so no detached subsequence selector or quotient stream can enter.

Finally apply the Real cluster seal and finite cluster-retention ledger. The terminal row E is reached only after the displayed regular package, while H, C, P, N preserve and name the same packet. Therefore the root obligation package is exactly $S, K, R, Q, E, H, C, P, N$ with the stated refusals. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier root obligation package`

Theorem 1254.47 (Bolzano–Weierstrass bounded-sequence admission)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet carrying the root obligation package of Definition 1254.46. The bounded source row* S *is admitted only as the source surface for the same interval-net row* K*, regular-subsequence row* R*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and* $\mathsf{Real}^{\uparrow}$ *cluster seal* E*. In particular, the bounded source is read through* S, K, H, C, P, N *before the rows* R, Q, E *are exposed, and it cannot be replaced by a detached host sequence, an arbitrary subsequence selector, or an ambient completeness assumption.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier bounded sequence admission`

Proof. Apply Definition 1254.2. It identifies S as the bounded sequence admission row and places it inside the carrier together with K, R, Q, E, H, C, P, N.

Apply Definition 1254.3 and Definition 1254.16. These force the interval-net row K and the regular-subsequence row R to consume the same bounded source windows rather than a second sequence source.

Finally use Definition 1254.46. The structural rows H, C, P, N bind the bounded source to the later readback and Real seal, so no detached host sequence or completeness principle is exported. □

Theorem 1254.48 (Bolzano–Weierstrass regular cluster seal)**.** *Let* $B = (S, K, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet whose bounded-sequence admission row and root obligation package are displayed. The cluster seal* E *is regular only through the route*

$$S, K, R, Q, E, H, C, P, N.$$

The row R *reads retained finite windows from* S *through* K*, the row* Q *reads the same windows as* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the row* E *seals only that regular readback as* $\mathsf{Real}^{\uparrow}$ *data. No cluster seal is available from a second bounded source, a detached interval net, quotient stream equality, selected subsequence choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier regular cluster seal`

Proof. Start with bounded-sequence admission. It fixes the source row S as the only bounded real-sequence surface available to the cluster route.

Use Definition 1254.5. The handoff places the retained windows behind K and R before Q reads them as regular rational data.

Use the interval-net obligation and the root obligation package. Together they bind K, R, Q to the same support rows H, C, P, N.

Finally read the Real seal E. Since E is downstream of Q in the displayed route, the seal is regular by the carried readback rows and not by a host completeness or selection principle. □

Theorem 1254.49 (Bolzano–Weierstrass closed-interval cluster export)**.** *An accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet exports a closed-interval cluster candidate only through the bounded-sequence admission row S, the closed interval net row K, the regular subsequence extraction row R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, with H, C, P, N transporting, replaying, sourcing, and naming the same route. The export refuses arbitrary subsequence choice, open-cover compactness, quotient stream equality, selected least cluster points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with bounded-source factorization and bounded-sequence admission. These rows fix S as the only source of sequence data and force every cluster-facing read to pass through the displayed bounded source.

Next apply finite interval-tree induction and retained-cell monotonicity. The closed interval net row K supplies the retained cells and their finite refinement route, so the packet cannot switch to an open-cover compactness proof or a detached interval net.

Use cofinal-window extraction and the regular cluster seal. The extraction row R reads retained windows through K, the readback row Q records the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ data, and the seal row E exports only that regular readback as $\mathsf{Real}^{\uparrow}$ cluster evidence.

Finally apply cluster-seal nonescape. The structural rows H, C, P, N preserve and name the same packet, and no carrier row remains from which arbitrary subsequence choice, quotient equality, least-cluster selection, or ambient completeness could be projected. □

Theorem 1254.50 (Bolzano–Weierstrass cluster-seal nonescape)**.** *For every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet $B = (S, K, R, Q, E, H, C, P, N)$ carrying the bounded-sequence admission and regular cluster-seal rows, any cluster-facing consumer remains inside the displayed surface*

$$S, K, R, Q, E, H, C, P, N.$$

The consumer may read a bounded source window, a retained interval-net cell, a regular-subsequence ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it cannot escape to an unlisted source row, an unlisted interval net, a selected least cluster point, quotient real equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1254.47. It fixes S and K as the source and interval-net surface for the cluster route.

Apply Definition 1254.48. It fixes R, Q, E as the regular-subsequence, regular readback, and Real-seal surface downstream of those rows.

The support rows H, C, P, N are the only transport, replay, provenance, and local naming coordinates in the carrier of Definition 1254.1. Therefore every cluster-facing read remains inside $S, K, R, Q, E, H, C, P, N$, and the refused host-level objects have no row from which they can be projected. □

Theorem 1254.51 (Bolzano–Weierstrass retained-window refinement)**.** *Let $B = (S, K, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet carrying the root obligation package of Definition 1254.46. Every retained dyadic interval cell used by the cluster route is read through*

$$S, K, R, Q, H, C, P, N$$

before the terminal $\mathsf{Real}^{\uparrow}$ *cluster seal E is exposed. The refinement preserves the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route: no retained window may be reclassified through a new stream name, a detached dyadic grid, a second rational-readback package, or a host compactness selector.*

Proof. Start with the root interval-tree obligation. It displays the retained dyadic cells through S and K, so each retained window is already attached to the bounded source and interval-net rows before the extraction route is read.

Next apply the root subsequence extraction obligation. The row R consumes only those retained cells, and the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ data used by the extraction remain the ones displayed by the same source and interval-net package.

Now use the RegSeqRat cluster package. The readback row Q is the $\mathsf{RegSeqRat}^{\uparrow}$ route for the retained windows already carried by S, K, R, so it cannot introduce a second rational-readback surface or a detached dyadic grid.

Finally apply the root Real-seal obligation and the root obligation package. The cluster seal E is downstream of S, K, R, Q, H, C, P, N, while H, C, P, N transport, replay, record provenance, and name the same route. Hence the retained-window induction is preserved all the way to the $\mathsf{Real}^{\uparrow}$ seal without importing a new stream, grid, readback package, or host selector. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier window retention refinement`

Theorem 1254.52 (Bolzano–Weierstrass public subsequence selector)**.** *For every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier $B = (S, K, R, Q, E, H, C, P, N)$, the public subsequence selector exported by the chapter is the finite route*

$$S, K, R, Q, H, C, P, N.$$

The bounded $\mathsf{Real}^{\uparrow}$ *sequence source is admitted through S, retained interval cells are admitted through K, monotone-cofinal subsequence windows are admitted through R, and the regular-rational readback is admitted through Q. The selector surface is therefore a displayed NameCert route, not an arbitrary subsequence choice functional, a host compactness theorem, quotient stream equality, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassPublicSubsequenceSelector`

Proof. Apply Definition 1254.47. It fixes the bounded sequence source S as the same source consumed by the interval-net row K and the later regular-subsequence rows.

Apply Definition 1254.21 and Definition 1254.24. They supply the retained finite interval cells and keep later windows nested inside the same displayed interval-net route.

Apply Definition 1254.25 and Definition 1254.18. These theorems identify R and Q as the monotone-cofinal window and $\mathsf{RegSeqRat}^{\uparrow}$ readback surface. The structural rows H, C, P, N then transport, replay, source, and name only that finite route. □

Theorem 1254.53 (Bolzano–Weierstrass subsequence-window publicity)**.** *The public subsequence-window export of an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier exposes only bounded sequence rows, closed bounded interval finite-net rows, regular Cauchy subsequence rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,*

$\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{Real}^{\uparrow}$ *seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and* NameCert *data. Every such export factors through the selector route*

$$S, K, R, Q, H, C, P, N$$

of Definition 1254.52 together with the chapter dependency route through retained intervals, regular subsequences, and terminal real seals.

The public window theorem excludes choice-selected subsequences, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassSubsequenceWindowPublicity`

Proof. First read the public selector theorem. It exposes the bounded source S, finite retained interval row K, regular-subsequence row R, regular-rational readback Q, and structural rows H, C, P, N as the only selector route.

Next compose this route with the chapter dependency route. The retained interval and finite-net rows are supplied before R, the stream windows and dyadic tolerance ledgers are read during retained-window inspection, and the $\mathsf{Real}^{\uparrow}$ seal is available only after the regular Cauchy subsequence and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows have been displayed.

The carrier has no row for an arbitrary choice-selected subsequence, no open-cover compactness object, and no ambient completeness principle. Therefore the public subsequence-window export is exactly the displayed bounded-sequence, finite-net, regular-Cauchy, window, readback, tolerance, seal, transport, replay, provenance, and naming surface. □

Theorem 1254.54 (Bolzano–Weierstrass public cluster witness)**.** *Every public cluster witness exported by an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet factors through*

$$S, K, R, Q, E, H, C, P, N.$$

It reads the bounded-sequence admission, retained-window ledger, regular-subsequence extraction, $\mathsf{RegSeqRat}^{\uparrow}$ *cluster readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal as one displayed package. It exports no least cluster point, no quotient equality of real names, no open-cover compactness object, and no host completeness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassPublicClusterWitness`

Proof. Start from Definition 1254.52. The public subsequence selector has already displayed S, K, R, Q, H, C, P, N and therefore contains the bounded source, retained windows, cofinal extraction, and regular rational readback.

Apply Definition 1254.48. It places the terminal $\mathsf{Real}^{\uparrow}$ seal E downstream of R and Q, so the cluster witness cannot be read before the regular-subsequence and readback rows are present.

Finally use Definition 1254.30. The extracted cluster seal is the Real boundary for that same finite route. The refused host-level objects are absent from the carrier and cannot be projected by the public witness. □

Theorem 1254.55 (Bolzano–Weierstrass public nonescape package)**.** *The public* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *export surface is exhausted by the selector and cluster-witness rows of Definition 1254.52 and Definition 1254.54. In particular, any consumer that reads a bounded sequence, retained finite interval tree, regular subsequence,* $\mathsf{RegSeqRat}^{\uparrow}$ *cluster row, or* $\mathsf{Real}^{\uparrow}$ *seal remains inside*

$$S, K, R, Q, E, H, C, P, N.$$

The package refuses arbitrary subsequence choice, unlisted interval nets, detached bounded sources, quotient stream equality, selected least cluster points, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassPublicNonescapePackage`

Proof. First read the selector surface. By Definition 1254.52, the only public subsequence route is S, K, R, Q, H, C, P, N.

Next read the cluster witness. By Definition 1254.54, the terminal Real seal E is available only after the bounded source, retained-window, regular-subsequence, and regular-rational rows have been displayed.

It remains to check nonescape. Apply Definition 1254.50 and Definition 1254.51. These theorems keep every cluster-facing and retained-window read inside the same finite carrier route, so no public consumer can import a detached selector, host compactness principle, quotient equality, or ambient completeness row. □

Theorem 1254.56 (Bolzano–Weierstrass public selector export)**.** *Every public selector or cluster export of an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet factors through the sibling bounded-sequence, interval-net, regular-rational, stream, and real-name surfaces before it exposes the terminal Real cluster seal. In particular, the bounded sequence is read through chapter 1415, the nested finite interval route through chapter 1250, and the readback and seal through the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *chapters. The public export route is therefore*

$$\mathsf{BoundedRealSequence}^{\uparrow}, \mathsf{ClosedBoundedIntervalNet}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto S, K, R, Q, E, H, C, P, N,$$

and no selector consumer receives a detached subsequence choice, an unlisted interval net, quotient stream equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassPublicSelectorExport`

Proof. Start with the public selector and cluster package. By Definition 1254.52, Definition 1254.54, and Definition 1254.55, the exported rows are exhausted by $S, K, R, Q, E, H, C, P, N$.

The sibling route identifies the source of those rows. The bounded source is the $\mathsf{BoundedRealSequence}^{\uparrow}$ surface, the retained cells are the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ surface, and the regular-rational and stream windows are the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ surfaces.

Finally the Real cluster seal is read only after those rows have been displayed. Since the carrier has no coordinate for a detached selector, unlisted interval net, quotient stream equality, or ambient completeness, every public export factors through the displayed sibling route. □

Theorem 1254.57 (Bolzano–Weierstrass public cluster export)**.** *Every accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet exports a public cluster package only through the displayed chain*

$$S \longmapsto K \longmapsto R \longmapsto Q \longmapsto E.$$

The source row is the bounded sequence surface S; the interval-tree row K supplies finite induction, retained-cell monotonicity, and cofinal window extraction; the extraction row R records the bounded-subsequence Cauchy route; the readback row Q packages the $\mathsf{RegSeqRat}^{\uparrow}$ *cluster package; and the terminal row E supplies the* $\mathsf{Real}^{\uparrow}$ *seal. No public cluster export may bypass this chain by selected subsequences, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the bounded sequence source. Definition 1254.19 and Definition 1254.44 make S the only source surface from which inspected windows enter the cluster route.

Next read the finite interval-tree surface. Definition 1254.21 supplies the finite refinement tree, Definition 1254.24 keeps later cells nested inside the same route, and Definition 1254.25 turns retained windows into the cofinal schedule recorded by R.

Then apply bounded-subsequence Cauchy extraction and the RegSeqRat cluster package. Definition 1254.39 forces the subsequence route through retained finite windows, while Definition 1254.45 packages R, Q, E as one regular-rational-to-Real cluster surface.

Finally use Definition 1254.46. The structural rows H, C, P, N only transport, replay, source, and name the same packet, so the public export has no channel for selected subsequences, countable choice, quotient real equality, or ambient Real completeness outside the displayed chain. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier public cluster export`

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ cluster read factors through the displayed interval net and regular Cauchy subsequence rows before reaching the $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: closedboundedintervalnet, closedboundedintervaluniformmodulus, regularcauchysubsequence, regseqrat, real: $\mathsf{BolzanoWeierstrass}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded sequence surface, finite interval nets, monotone-cofinal subsequence extraction, regular rational readback, RealUp cluster sealing, transport, continuation, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence surface, finite closed bounded interval nets, interval-modulus compatibility, a monotone-cofinal regular Cauchy subsequence extraction row, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrass}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1254.1, Definition 1254.2, Definition 1254.18, Definition 1254.19, Definition 1254.21, Definition 1254.24, Definition 1254.25, Definition 1254.26, and Definition 1254.30 bind the bounded-sequence subsequence compactness route through the carrier, NameCert obligations, bounded sequence extraction, finite interval nets, regular-subsequence route, RegSeqRat readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, host completeness, quotient equality of streams, least cluster selection, or arbitrary subsequence choice.

Upgrade path: The paper-side mature treatment is closed; the remaining checks are Lean verification and bridge checking for BEDC.Derived.BolzanoWeierstrassUp.

Theorem 1254.58 (Bolzano–Weierstrass cluster-source factorization)**.** *Every public cluster read from an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet factors through the finite source route*

$$\mathsf{BoundedRealSequence}^{\uparrow}, \mathsf{RealSequenceClusterPoint}^{\uparrow}, \mathsf{RegularCauchySubsequence}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before any Bolzano–Weierstrass seal is consumed. Equivalently, the public read first exposes the bounded sequence source, then the cluster-point carrier, then the regular Cauchy subsequence row, then the regular rational readback, and only then the terminal Real seal attached to the Bolzano–Weierstrass packet.

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier cluster source factorization`

Proof. Begin with the public selector and cluster export surface of the Bolzano–Weierstrass chapter. The carrier and public export theorems expose the bounded source row S, the regular-subsequence

row R, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the terminal $\mathsf{Real}^{\uparrow}$ cluster seal E before any public consumer may read the packet.

Now apply Definition 1580.4. It identifies the cluster-point consumer route through the bounded source row of chapter 1415, the cofinal subsequence row of chapter 381, selected windows, regular readback, dyadic tolerance, and the Real cluster seal.

Composing these routes gives the stated factorization. The Bolzano–Weierstrass seal is downstream of the bounded source, cluster-point carrier, regular Cauchy subsequence, RegSeqRat readback, and Real seal, so no public cluster read can bypass them through selected subsequences, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1254.59 (Bolzano–Weierstrass retained-window source)**.** *A retained cluster source for an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet can be consumed only after the bounded real sequence row, closed-bounded interval net row, regular Cauchy subsequence row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and terminal* $\mathsf{Real}^{\uparrow}$ *seal have all been displayed. Consequently every later Real or Completion consumer receives a finite retained-window source rather than a selected subsequence, open-cover compactness proof, quotient stream equality, or ambient Real-completeness principle.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassUp.BolzanoWeierstrassCarrier retained window source`

Proof. Project the public source factorization of Definition 1254.58. The bounded source row is read through chapter 1415, and its closed-bounded interval net row is the displayed bound surface supplied by chapter 1250.

Read the retained subsequence surface next. The regular Cauchy subsequence row of chapter 381 can be inspected only through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows.

Attach the terminal seal after those finite rows. The $\mathsf{Real}^{\uparrow}$ boundary is therefore downstream of boundedness, retained windows, regular readback, and dyadic tolerance, not an independent completeness oracle.

The retained cluster source is exactly this projection of displayed rows. Since no carrier coordinate supplies a selected subsequence, an open-cover compactness proof, quotient stream equality, or ambient Real-completeness theorem, no public consumer can read any of those objects from the packet. □

Theorem 1254.60 (Cluster-source factorization formal target)**.** *The formal target for Definition 1254.58 is the BEDC kernel statement that an accepted* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *cluster export factors through four displayed rows before the terminal seal is consumed: a bounded real sequence carrier over* BHist *and* $\mathsf{Real}^{\uparrow}$*; a closed-bounded interval net whose endpoints and refinements are recorded by* Cont *replay; a regular Cauchy subsequence row governed by* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback; and a cluster-point witness in the* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *surface. The statement names only this route and does not claim full sequential compactness for arbitrary real sequences.*

Lean4 ✓ `BEDC.Derived.BolzanoweierstrassUp.BolzanoWeierstrassClusterSourceFormalTarget`

Proof. The bounded real sequence carrier supplies the source row and the interval net records finite closed-bounded windows. Both are ordinary BEDC rows, so the formalization can expose them by carrier projection and displayed Cont replay.

The regular Cauchy subsequence row then gives the constructive subsequence route, while $\mathsf{RegSeqRat}^{\uparrow}$ supplies rational readback of the same row. The cluster-point witness is read only after these rows have been exposed.

Thus the formal target is a factorization theorem about the accepted packet's named rows. It does not need a host compactness principle, an open-cover argument, or a choice-selected subsequence outside the displayed regular Cauchy data. □

Chapter 1255

CofinalTailEquivalence NameCert Construction

The $\mathsf{CofinalTailEquivalence}^{\uparrow}$ packet realizes the paired cofinal-tail route comparison used by the regular-Cauchy completion surface of section 9238.1. It sits after the one-sided cofinal agreement of chapter 920, the shared threshold discipline of chapter 1231, and the Real-facing seal route of chapter 699. The packet records two displayed cofinal tail routes with one shared StreamName, RegSeqRat, DyadicRatCore, RealTailAgreementSeal, transport, continuation, provenance, and naming surface; it is not a quotient equality of regular sequences, a selected host limit, a hidden tail oracle, countable choice, or ambient Real completeness.

Falsifiable prediction: A packet missing either displayed cofinal route cannot discharge the Real-seal consumer theorem.

Independence witness: cofinaltailagreement, tailcofinalitybudget, dyadiccofinaltailselector, realtailagreementseal, diagonallimitcompatibility: the carrier has two-route equivalence plus Real-seal consumption, while these siblings supply single agreement, budget, selector, seal, or diagonal compatibility rows without that combined bijective surface.

Definition 1255.1 (Cofinal tail-equivalence carrier)**.** *A* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R_0, R_1, W, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two cofinal tail-route rows being compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *readback row through which both routes expose the same finite observation window.*
3. Q *is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to that window on both tail routes.*
4. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the tolerance and endpoint observations consumed by the comparison.*
5. A *is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer row reached only after both cofinal routes, the shared window, the regular readback, and the dyadic ledger are displayed.*
6. H *records componentwise* hsame *transport for* R_0, R_1, W, Q, D, A *and the local certificate row.*
7. C *records displayed* Cont *replay from each cofinal route through the shared StreamName, RegSeqRat, DyadicRatCore, and RealTailAgreementSeal rows.*

8. *P is the* Pkg *provenance over* $\mathsf{CofinalTailEquivalence}^{\uparrow}$, $\mathsf{CofinalTailAgreement}^{\uparrow}$, $\mathsf{TailCofinalityBudget}^{\uparrow}$, $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{DiagonalLimitCompatibility}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailEquivalence}^{\uparrow}}$ *row.*

The carrier exports only the two displayed cofinal routes, their shared finite readback rows, the Real-tail agreement seal consumer row, and the structural transport, replay, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.CofinalTailEquivalenceUp.CofinalTailEquivalenceTasteGate single carrier alignment`

Theorem 1255.2 (Cofinal tail-equivalence NameCert obligations)**.** *For the carrier of Definition 1255.1, the local* $\mathsf{NameCert}_{\mathsf{CofinalTailEquivalence}^{\uparrow}}$ *surface consists exactly of carrier admission for $E = (R_0, R_1, W, Q, D, A, H, C, P, N)$, two cofinal route exposures through R_0 and R_1, shared* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W, shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposure through Q, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exposure through D,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer exposure through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CofinalTailEquivalenceUp.CofinalTailEquivalence namecert obligations`

Proof. Project the carrier. Its coordinates display exactly the two cofinal route rows, the shared StreamName window, the regular-rational readback, the dyadic ledger, the Real-tail agreement seal consumer row, and the structural support rows.

Read the sibling sources in order. The $\mathsf{CofinalTailAgreement}^{\uparrow}$ row supplies the one-sided agreement surface, the $\mathsf{TailCofinalityBudget}^{\uparrow}$ row supplies the shared threshold discipline, and the $\mathsf{DyadicCofinalTailSelector}^{\uparrow}$ row supplies the finite selector route that may feed either side.

Finally read the seal consumer. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row A is downstream of both routes and the shared readback rows, while H, C, P, N only transport, replay, source, and name those displayed rows. These clauses exhaust the local NameCert obligations. □

Theorem 1255.3 (Cofinal tail-equivalence tail-agreement transport)**.** *Let $E = (R_0, R_1, W, Q, D, A, H, C, P, N)$ be an accepted* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *carrier. If the first cofinal route R_0 is transported to the second route R_1 through the displayed H and C rows, then the tail-agreement read remains on the same shared finite surface*

$$W, Q, D, A, H, C, P, N.$$

The transport preserves the StreamName readback, RegSeqRat readback, DyadicRatCore ledger, and RealTailAgreementSeal consumer row, and it exports no quotient stream equality, host real equality, selected limit, hidden tail selector, countable-choice row, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CofinalTailEquivalenceUp.CofinalTailEquivalenceTailAgreementTransport`

Proof. Begin with the two route coordinates of Definition 1255.1. The only relation between them inside the packet is the componentwise hsame table H and the displayed continuation replay C.

Apply the NameCert obligations. They force both R_0 and R_1 to read the same W, Q, D, A rows before any tail-agreement consumer can inspect the packet.

Thus transporting the tail-agreement read from R_0 to R_1 changes only the displayed route coordinate. The shared readback and seal rows remain W, Q, D, A, H, C, P, N, and the excluded external equalities and selectors have no carrier coordinate from which they could be projected. □

Theorem 1255.4 (Cofinal tail-equivalence Real-seal consumer)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *packet reaches the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* A *only after both cofinal routes* R_0, R_1*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *have been displayed. The consumer route is exhausted by*

$$R_0, R_1, W, Q, D, A, H, C, P, N.$$

No consumer of this packet receives a one-route agreement in place of two-route equivalence, a host equality of completed reals, a quotient-stream comparison, a selected limit, a hidden tail oracle, or an ambient completeness witness.

Proof. Inspect the carrier inventory. The Real-facing coordinate is the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row A, and it appears only after the two route rows and the three shared readback rows.

Use Definition 1255.2. It makes R_0, R_1, W, Q, D mandatory before A can be consumed, while H, C, P, N authenticate the same finite route.

Use Definition 1255.3. Any transport between the two routes preserves the shared tail-agreement surface and therefore cannot collapse the carrier to a one-route packet or import a host-level equality. The listed finite route is the whole Real-seal consumer surface. □

Theorem 1255.5 (Cofinal tail-equivalence common-tail induction)**.** *Let* $E = (R_0, R_1, W, Q, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *carrier. Finite induction over the two displayed cofinal route rows* R_0 *and* R_1 *preserves the shared readback surface*

$$W, Q, D, A$$

at every common threshold reached by both routes. In particular, the common threshold read still exposes the same $\mathsf{StreamName}^{\uparrow}$ *row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* A*, and it does not introduce quotient stream equality, a selected host limit, hidden tail selection, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CofinalTailEquivalenceUp.CofinalTailEquivalenceCommonTailInduction`

Proof. Project the carrier of Definition 1255.1. The route coordinates R_0 and R_1 are the only cofinal rows, and both are paired with the same displayed W, Q, D, A surface.

For the base threshold, apply the NameCert obligations of Definition 1255.2. They force both routes to display the StreamName window, RegSeqRat readback, DyadicRatCore ledger, and RealTailAgreementSeal consumer before any common-tail read is available.

For the successor threshold, use the transport theorem Definition 1255.3. Transport from one displayed route to the other changes only the route coordinate while preserving the shared readback and seal rows.

The Real-seal consumer theorem Definition 1255.4 then closes the induction step for completion-facing reads. Since the carrier has no coordinate for quotient equality, selected limits, hidden tail selection, countable choice, or ambient completeness, the induced common-tail read cannot export any of those surfaces. □

Theorem 1255.6 (Cofinal tail-equivalence RegSeqRat stability)**.** *Let* $E = (R_0, R_1, W, Q, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *carrier. Componentwise* hsame *transport of both cofinal routes through* H *preserves the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and keeps it tied to the same* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer row* A*. This stability cannot create quotient stream equality, selected host limits, countable choice, ambient Real completeness, or host equality of completed reals.*

Proof. Start from the componentwise transport row H in the carrier. It transports only the displayed coordinates R_0, R_1, W, Q, D, A and the local certificate row; it is not a quotient constructor for regular sequences.

Apply Definition 1255.3. The theorem says that moving between the two cofinal routes preserves the shared readback surface W, Q, D, A, H, C, P, N.

Now apply the common-tail induction theorem Definition 1255.5. At each common threshold the same RegSeqRat row remains paired with the same StreamName, dyadic ledger, and Real-seal rows.

Finally apply the Real-seal consumer theorem Definition 1255.4. Completion-facing consumers receive only the displayed finite route, so transport stability for Q does not manufacture quotient stream equality, selected host limits, countable choice, ambient Real completeness, or host real equality. □

Theorem 1255.7 (Cofinal tail-equivalence scope readback)**.** *The scoped proof surface of* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *depends only on the displayed* BHist *carrier, componentwise* hsame *rows, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *and the cofinal-tail sibling rows named in Definition 1255.1. It exports no quotient stream equality, selected host limit, hidden tail oracle, countable choice, host real equality, or ambient Real completeness.*

Proof. Start with the carrier of Definition 1255.1. Its row inventory is exactly $R_0, R_1, W, Q, D, A, H, C, P, N$, with the cofinal-tail sibling rows appearing only through the two route coordinates and the shared readback surface.

Apply Definition 1255.2 and Definition 1255.3. These theorems bind the local certificate clauses and transport any route comparison through the displayed hsame and Cont rows without adding an external selector or quotient.

Finally use Definition 1255.4. The Real-facing read reaches only the displayed $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer after $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ have been displayed, so the listed rows are the whole scoped readback surface. □

Theorem 1255.8 (Cofinal tail-equivalence scoped closure)**.** *The scope readback theorem, together with the Real-seal consumer, common-tail induction, and RegSeqRat stability theorems, binds every exported* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *consumer to the displayed* BHist *carrier,* hsame *transport,* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *and the named cofinal-tail sibling rows of Definition 1255.1. No exported consumer can leave that scope by reading quotient stream equality, selected host limits, hidden tail selection, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1255.7. It identifies the only rows available to the scoped proof surface: the displayed carrier, transport, replay, provenance, local naming, StreamName readback, RegSeqRat readback, DyadicRatCore ledger, RealTailAgreementSeal consumer, and cofinal-tail siblings.

Apply Definition 1255.4. Completion-facing reads reach the Real-tail agreement row only after the two cofinal routes and the shared finite readback rows have been displayed.

Apply Definition 1255.5 and Definition 1255.6. These theorems keep common-threshold and transport reads on the same StreamName, RegSeqRat, DyadicRatCore, and RealTailAgreementSeal surface.

Thus the scope is closed around exactly the named finite rows. The excluded quotient, selection, choice, host equality, and ambient-completeness objects are absent from the carrier and cannot be projected by any exported consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailEquivalence}\uparrow}$.

Constructive story: A $\mathsf{CofinalTailEquivalence}^{\uparrow}$ packet is generated from two cofinal tail routes, a shared StreamName window, a shared RegSeqRat readback, a shared DyadicRatCore ledger, a RealTailAgreementSeal consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailEquivalence}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 1255.1, Definition 1255.2, Definition 1255.3, Definition 1255.4, Definition 1255.5, Definition 1255.6, Definition 1255.7, and Definition 1255.8 bind the carrier, NameCert obligations, tail-agreement transport, Real-seal consumer route, common-tail induction, RegSeqRat stability, scoped readback surface, and scoped closure theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailEquivalence}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CofinalTailEquivalenceUp.CofinalTailEquivalenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, host real equality, selected host limits, hidden tail selection, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Public export and bridge work require a public CofinalTailEquivalenceUp object surface and a checked standard bridge for the same finite packet.

Chapter 1256

A Concrete Naming Certificate for $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$

The $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$ packet realizes the auditable-generativity reading of chapter 9462 as a finite synthesis-audit NameCert carrier. It sits beside the core registry index of chapter 1251, the registry export surface of chapter 1216, the theorem-gap registry of chapter 1246, the audit gate boundary of chapter 526, and the focused ledger of chapter 1247. Those siblings index, export, gate, audit, and focus finite rows; this carrier binds the philosophical synthesis only when the same rows make the synthesis auditable.

Definition 1256.1 (Philosophy synthesis audit carrier)**.** *A* $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, E, G, A, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CoreRegistryIndex}^{\uparrow}$ *row naming the concept, theorem, gap, tradition, scientific-claim, refusal, route, ordinary-language, and audit entries that the synthesis may read;*
2. *E is the* $\mathsf{RegistryExportSurface}^{\uparrow}$ *row recording whether a synthesized reading is exported, blocked, horizon-facing, or held as a formalization target;*
3. *G is the* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *row separating theorem-strength readings from gap-gate readings;*
4. *A is the* $\mathsf{AuditGateBoundary}^{\uparrow}$ *row recording the audit route by which the synthesis is checked before export;*
5. *L is the* $\mathsf{FocusedLedger}^{\uparrow}$ *row recording the finite focus, selector, digest, gap, relation, invariant, dependence, self-proxy, and audit rows consumed by the reading;*
6. *Q is the synthesis-question row recording the philosophical reading, its allowed registry entries, its blocked readings, its audit-route requirement, and its not-claimed surface;*
7. *H records componentwise* hsame *transport for R, E, G, A, L, Q;*
8. *C records displayed* Cont *replay through the registry index, export surface, theorem-gap registry, audit boundary, focused ledger, and synthesis-question row;*
9. *P is the* Pkg *provenance over* $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhilosophySynthesisAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for inspiration-only synthesis, hidden registry export, gate-free theorem strength, audit-free philosophical conclusion, unledgered focus selection, or a reading that bypasses the not-claimed surface in Q.

Lean4 ✓ `BEDC.Derived.PhilosophySynthesisAuditUp.PhilosophySynthesisAuditTasteGate single carrier alignment`

Theorem 1256.2 (Philosophy synthesis audit NameCert obligations)**.** *For the carrier of Definition 1256.1, the local* $\mathsf{NameCert}_{\mathsf{PhilosophySynthesisAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (R, E, G, A, L, Q, H, C, P, N),$$

registry-index coverage through R, export-status coverage through E, theorem-gap separation through G, audit-route exposure through A, focused-ledger support through L, synthesis-question accountability through Q, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite carrier of Definition 1256.1. The rows R, E, G, A, L, Q display the registry, export, theorem-gap, audit, focus, and synthesis-question data before any public reading of auditable generativity is allowed.

Read the sibling rows in order. $\mathsf{CoreRegistryIndex}^{\uparrow}$ supplies the finite registry entries, $\mathsf{RegistryExportSurface}^{\uparrow}$ controls whether the reading may leave the registry, $\mathsf{TheoremGapRegistry}^{\uparrow}$ keeps theorem strength distinct from gap status, $\mathsf{AuditGateBoundary}^{\uparrow}$ supplies the audit route, and $\mathsf{FocusedLedger}^{\uparrow}$ records the finite focused ledger consumed by the reading.

Finally inspect Q, H, C, P, N. The synthesis-question row records the philosophical reading and its not-claimed surface, while the structural rows only transport, replay, source, and name the same finite packet. Hence the local NameCert obligations are exactly the displayed synthesis-audit rows. □

Theorem 1256.3 (Philosophy synthesis audit nonescape)**.** *Every exported auditable-generativity reading of an accepted* $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$ *packet must read the registry-index row R, export row E, theorem-gap row G, audit-route row A, focused-ledger row L, synthesis-question row Q, transport row H, replay row C, provenance row P, and local naming row N. A packet without the audit-route row A cannot export an auditable-generativity reading.*

Proof. Apply the NameCert obligations of Definition 1256.2. They require the registry, export, theorem-gap, audit, focused-ledger, and synthesis-question rows as one finite packet before a reading can be exported.

If A is absent, the reading still has no audit route. The registry index can name entries, the export surface can classify status, the theorem-gap registry can separate theorem and gap rows, and the focused ledger can display finite focus data, but none of those rows records the audit-route check by itself.

The structural rows H, C, P, N preserve an admitted packet; they do not synthesize a missing audit boundary. Thus a synthesis-audit packet without A exits the accepted carrier and cannot export an auditable-generativity reading. □

Falsifiable prediction: A synthesis-audit packet without the audit-route row cannot export an auditable-generativity reading.

Independence witness: coreregistryindex, registryexportsurface, theoremgapregistry, auditgateboundary, focusedledger: the core registry index lacks export-status and synthesis-question accountability, the registry export surface lacks theorem-gap and focused-ledger rows, the theorem-gap registry lacks the philosophical synthesis-question row, the audit gate boundary lacks registry-index and focused-ledger content, and the focused ledger lacks export-status and audit-route authority; no sibling is bijective to $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$.

The Lean follow-up must provide $\mathsf{Nontrivial\ PhilosophySynthesisAudit}^{\uparrow}$ and $\mathsf{FieldFaithful\ PhilosophySynthesisAudit}$ in `lean4/BEDC/Derived/PhilosophySynthesisAuditUp/TasteGate.lean`. Those instances must preserve $R, E, G, A, L, Q, H, C, P, N$ as separate displayed fields before the target BEDC.Derived.PhilosophySynthesis can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{PhilosophySynthesisAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{PhilosophySynthesisAudit}^{\uparrow}$ packet is generated from core-registry, registry-export, theorem-gap, audit-route, focused-ledger, synthesis-question, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhilosophySynthesisAudit}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1256.1, Definition 1256.2, and Definition 1256.3 bind auditable generativity to the displayed finite synthesis-audit packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhilosophySynthesisAudit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.PhilosophySynthesisAuditUp.PhilosophySynthesisAuditTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, full philosophical completeness, theorem truth for every registry entry, or export of readings without an audit route.

Upgrade path: Lean closure requires Nontrivial PhilosophySynthesisAuditUp and FieldFaithful PhilosophySynthesisAuditUp instances in `lean4/BEDC/Derived/PhilosophySynthesisAuditUp/TasteGate.lean`.

Chapter 1257

A Concrete Naming Certificate for $\mathsf{CompactUniformEpsilonTriangleHandoff}^{\uparrow}$

The $\mathsf{CompactUniformEpsilonTriangleHandoff}^{\uparrow}$ packet isolates the epsilon-third source and target triangle step in the compact-uniform-continuity construction of chapter 9237. It sits after the source metric and compact finite-net rows of chapter 57 and chapter 60, the continuous-map pointwise-modulus rows of chapter 130, the rational tolerance rows of chapter 37, the selected uniform-modulus consumer row of chapter 369, and the final triangle estimate packet of chapter 1159. Its carrier is narrower than $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$: it records the precise source triangle that keeps x' inside the selected compact center and the target triangle that combines the two $\varepsilon/3$ pointwise estimates.

Definition 1257.1 (Compact-uniform epsilon-triangle handoff carrier)**.** *A* $\mathsf{CompactUniformEpsilonTriangleHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, c, x, x', \varepsilon, \mu, \rho, d_x, d_{x'}, d_y, d_{y'}, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, read together with the* $\mathsf{Compact}^{\uparrow}$ *finite-net center row for c;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row carrying the target triangle certificate;*
3. *c is the compact center selected from the finite cover for the tolerance history ε;*
4. *x and x' are the source histories whose target images are compared by the uniform-modulus consumer;*
5. *ε is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row whose three equal thirds are read by the pointwise target estimates;*
6. *μ is the positive* $\mathsf{Rat}^{\uparrow}$ *uniform precision row exported to* $\mathsf{UniformModulus}^{\uparrow}$*;*
7. *ρ is the positive* $\mathsf{Rat}^{\uparrow}$ *local center radius subordinate to the pointwise-modulus row at c;*
8. *d_x is the compact coverage distance row certifying $d_X(x, c) < \rho/2$;*
9. *$d_{x'}$ is the source triangle row certifying $d_X(x', c) \leq d_X(x', x) + d_X(x, c) < \rho$ from the μ-bound on $d_X(x, x')$ and the coverage row d_x;*
10. *d_y is the pointwise-modulus target row at c for the pair (c, x), bounded by $\varepsilon/3$;*

11. *$d_{y'}$ is the pointwise-modulus target row at c for the pair (c, x'), bounded by $\varepsilon/3$;*
12. *H records componentwise* hsame *transport for the metric, compact center, source histories, rational tolerance, precision, radius, source triangle, pointwise target, and target triangle rows;*
13. *C records the displayed* Cont *routes from compact coverage to source triangle, from source triangle to the double pointwise-modulus application, and from those target estimates to the uniform-modulus output;*
14. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, MetricTriangleUniformE BHist, hsame, Cont, *and* NameCert *rows;*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEpsilonTriangleHandoff}^{\uparrow}}$ *row.*

The classifier compares carriers only through these displayed rows. It has no coordinate for an open-cover compactness proof, a selected subsequence, a host-level modulus function, a quotient real equality, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CompactUniformEpsilonTriangleHandoffUp.CompactUniformEpsilonTriangleHandoffTasteGa`
`single carrier alignment`

Theorem 1257.2 (Compact-uniform epsilon-triangle handoff NameCert obligations)**.** *For the carrier of Definition 1257.1, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEpsilonTriangleHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$E = (X, Y, c, x, x', \varepsilon, \mu, \rho, d_x, d_{x'}, d_y, d_{y'}, H, C, P, N),$$

compact-center preservation through c, source triangle admission through d_x and $d_{x'}$, double pointwise-modulus admission through d_y and $d_{y'}$, target triangle admission through Y, uniform-modulus output through μ, componentwise hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The rows $X, c, x, x', \varepsilon, \mu, \rho, d_x, d_{x'}, d_y, d_{y'}$ are the analytic rows, while H, C, P, N are the structural rows that transport, replay, package, and name the same finite packet.

The compact center is preserved because c is a carrier coordinate and every consumer route in C starts from the coverage row at that center. The source triangle row $d_{x'}$ is therefore read from the same X, c, x, x' data as the coverage row d_x.

The two pointwise-modulus rows then use the same center c and the same tolerance split $\varepsilon/3$. The target metric row Y consumes those two target estimates and sends the resulting bound to the $\mathsf{UniformModulus}^{\uparrow}$ output row attached to μ.

No further row is present in the carrier. Thus the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CompactUniformEpsilonTriangleHandoffUp.CompactUniformEpsilonTriangleHandoff`
`namecert obligations`

Theorem 1257.3 (Compact-uniform epsilon-triangle handoff estimate)**.** *For every accepted* CompactUniformEpsilo *packet*

$$E = (X, Y, c, x, x', \varepsilon, \mu, \rho, d_x, d_{x'}, d_y, d_{y'}, H, C, P, N),$$

the uniform-modulus consumer receives the following finite estimate route. The compact row preserves the selected center c and supplies $d_X(x, c) < \rho/2$. The source metric row combines this with the μ-bound on $d_X(x, x')$ to certify

$$d_X(x', c) \le d_X(x', x) + d_X(x, c) < \rho.$$

The continuous-map pointwise-modulus row at c then applies to both x and x', giving $d_y < \varepsilon/3$ and $d_{y'} < \varepsilon/3$. The target metric triangle row combines d_y and $d_{y'}$ into the displayed ε-bound exported to UniformModulus$^{\uparrow}$.

Proof. Start with the compact center row. The carrier preserves c together with the coverage row d_x, so the input point x is within the displayed half-radius around c.

Use the source triangle row in X. Since the uniform precision row μ bounds the distance between x and x' and is below the selected half-radius ledger, the source triangle gives the displayed bound on $d_X(x', c)$ and keeps x' inside the same local radius ρ.

Now apply the pointwise-modulus rows at c. The radius bound ρ is subordinate to the local modulus, so the same continuous-map packet supplies the target estimates $d_y < \varepsilon/3$ and $d_{y'} < \varepsilon/3$.

Finally read the target triangle certificate in Y. It joins the two target estimates through the same center image and exports the resulting ε-bound along the continuation route to UniformModulus$^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CompactUniformEpsilonTriangleHandoffUp.TasteGate.CompactUniformEpsilonTriangleHand`
`single carrier alignment`

Falsifiable prediction: If the source triangle row $d_{x'}$ is removed, the packet cannot certify that x' lies inside the local radius around the selected compact center, and the second pointwise-modulus application at c is unavailable.

Independence witness: metric, compact, continuousmap, rat, uniformmodulus, metric-triangle-uniform-estimate: CompactUniformEpsilonTriangleHandoff$^{\uparrow}$ binds one selected center, two source points, the epsilon-third split, the source triangle route, two pointwise target rows, and the final target triangle route; none of the named siblings alone contains both the source and target triangle handoff as one finite packet.

Closure status for NameCert$_{\text{CompactUniformEpsilonTriangleHandoff}^{\uparrow}}$.

Constructive story: A CompactUniformEpsilonTriangleHandoff$^{\uparrow}$ packet is generated from source and target metric rows, a compact finite-net center, two source histories, a positive rational tolerance, a positive uniform precision, a local radius, source-distance rows, pointwise target estimates, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformEpsilonTriangleHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 1257.1, Definition 1257.2, and Definition 1257.3 bind the epsilon-third source and target triangle handoff for compact-uniform-continuity to one finite carrier.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformEpsilonTriangleHandoff}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CompactUniformEpsilonTriangleHandoffUp.CompactUniformEpsilonTriangleHandoff\_namecert\_`

Bridge status: none

Not claimed: This chapter does not close Lean verification, Real completion, open-cover compactness, selected subsequences, quotient real equality, or an ambient completeness principle.

Upgrade path: The next paper grade is obligation closure for compact-center preservation, source triangle admission, double pointwise-modulus admission, target triangle admission, and UniformModulus$^{\uparrow}$ output.

Chapter 1258

A Concrete Naming Certificate for $\mathsf{AuditSystem}^{\uparrow}$

The $\mathsf{AuditSystem}^{\uparrow}$ packet realizes the total audit theory of subsection 9462.31.11 as a finite NameCert surface. It sits beside the kernel-acceptance audit witness of chapter 547, the claim registry layer of chapter 1209, and the failure-certificate packet of chapter 1233. As a composite over those sibling audit surfaces, the packet bundles their acceptance, registry, and refusal rows into one finite audit carrier. Those siblings expose kernel acceptance, positive claim registration, and typed refusal separately; this packet binds admissible claims, positive certificates, failure certificates, cannot-claim rows, export grades, and ledger closure in one finite audit carrier.

As a composite over $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, and $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, the packet records the joint audit surface rather than a primitive carrier discovery.

Definition 1258.1 (Audit system carrier)**.** *An* $\mathsf{AuditSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (C, P, F, R, E, L, H, K, Q, N)$$

with the following displayed rows:

1. *C is the admissible-claim row, recording the claim name, subject surface, carrier, classifier, and intended export site;*
2. *P is the positive-certificate row, recording the certificate that licenses the claim when the carrier, classifier, ledger, status, and surface fields are present;*
3. *F is the failure-certificate row, recording the missing, invalid, overstated, unchecked, or axis-confused field when export is blocked;*
4. *R is the cannot-claim row, recording a typed refusal when the packet is required to expose non-export rather than positive admission;*
5. *E is the export-grade row, recording whether the claim is read as paper-scoped, theorem-checked, audit-clean, axiom-clean, bridge-checked, or explicitly non-exportable;*
6. *L is the ledger-closure row, recording that the claim, certificate, failure certificate, refusal row, export grade, provenance, and local naming are closed as one audit packet;*
7. *H stores componentwise* hsame *transport for admissible claim, positive certificate, failure certificate, cannot-claim row, export grade, ledger closure, replay, provenance, and local naming;*

8. *K stores displayed* Cont *replay for reading C, P, F, R, E, L without adding a host-side repair route;*
9. *Q is the* Pkg *provenance over* $\mathsf{AuditSystem}^{\uparrow}$, $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditSystem}^{\uparrow}}$ *row.*

The carrier has no coordinate for exporting a claim outside the displayed positive certificate, failure certificate, refusal row, export grade, and ledger-closure surface.

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemTasteGate single carrier alignment`

Theorem 1258.2 (Audit system NameCert surface)**.** *For the carrier of Definition 1258.1, the local* $\mathsf{NameCert}_{\mathsf{AuditSystem}^{\uparrow}}$ *surface consists exactly of the admissible-claim row C, positive-certificate row P, failure-certificate row F, cannot-claim row R, export-grade row E, ledger-closure row L, componentwise transport row H, replay row K, provenance row Q, and local naming row N.*

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemCarrier namecert surface`

Proof. Project the finite packet $A = (C, P, F, R, E, L, H, K, Q, N)$. The admissible claim is read first through C, but it is not an exported BEDC claim until the certificate and boundary rows are read in the same carrier.

Rows P, F, R, E supply the positive and negative audit faces. The positive row licenses export, the failure row blocks export by naming the failed field, the cannot-claim row records explicit refusal, and the export-grade row records the public strength of the same claim site.

Rows L, H, K, Q, N then close, transport, replay, package, and name the displayed audit data. Since the carrier contains no additional export coordinate, these ten rows exhaust the local NameCert surface. □

Theorem 1258.3 (Audit system export refusal)**.** *If an accepted* $\mathsf{AuditSystem}^{\uparrow}$ *packet attempts to export the claim row C while its failure-certificate row F or cannot-claim row R is active for the same claim site, then the packet produces a certificate conflict rather than an exportable claim.*

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemCarrier export refusal conflict`

Proof. Assume an accepted packet $A = (C, P, F, R, E, L, H, K, Q, N)$. The export-grade row E is read together with the positive certificate P, failure certificate F, and cannot-claim row R.

If F is active for the same claim site, the packet records a missing, invalid, overstated, unchecked, or axis-confused field. The positive certificate cannot erase that negative row, because both rows are displayed before ledger closure is read.

If R is active, the packet records explicit refusal. Again P cannot by itself create an export route, since the carrier admits no coordinate for export outside P, F, R, E, L.

Thus the accepted packet exposes a certificate conflict. It may be audited as a blocked or refused claim, but it is not an exportable claim. □

Theorem 1258.4 (Audit system certificate conflict determinacy)**.** *Let $A = (C, P, F, R, E, L, H, K, Q, N)$ be an accepted* $\mathsf{AuditSystem}^{\uparrow}$ *packet. If the same claim site has active F or active R, then every export-grade read through E, L, H, K, Q, N is classified as the same blocked or refused claim boundary, and no second export route can be produced by replaying P alone.*

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemCertificateConflictDeterminacy`

Proof. Project the accepted carrier of Definition 1258.1. The negative rows F and R share the same claim site as C, and the export-grade row E is read together with those rows before ledger closure.

Apply Definition 1258.3. An active F yields a certificate conflict, and an active R yields explicit refusal; in either case the packet is classified at the blocked or refused claim boundary rather than as an exportable claim.

Now use Definition 1258.5. Rows L, H, K, Q, N close, transport, replay, source, and name the same displayed negative boundary. Since the carrier has no coordinate for a second export grade and P is not readable outside F, R, E, L, replaying P alone cannot produce another export route. □

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemCarrier certificate conflict determinacy`

Theorem 1258.5 (Audit system ledger closure)**.** *Every exported claim admitted by* $\mathsf{AuditSystem}^{\uparrow}$ *is ledger-closed by the same finite packet that displays its admissible-claim row, certificate row, negative rows, export grade, transport, replay, provenance, and local NameCert row.*

Proof. By Definition 1258.1, the only ledger-closure coordinate is L inside $A = (C, P, F, R, E, L, H, K, Q, N)$. The packet therefore cannot close a claim by citing an external ledger while omitting C, P, F, R, E.

The transport, replay, provenance, and local naming rows H, K, Q, N preserve the same displayed rows. They do not add a second claim, a second export grade, or a hidden ledger.

Consequently any admitted export is closed by the same finite packet that names its positive and negative audit data. □

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemLedgerClosure`

Theorem 1258.6 (Audit system independence witness)**.** *The* $\mathsf{AuditSystem}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, *or* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$.

Proof. Compare the displayed rows. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ registers or refuses package claims, $\mathsf{FailureCertificate}^{\uparrow}$ supplies a typed negative certificate, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ blocks inconsistent export, and $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$ records kernel-side acceptance audit data.

None of those siblings carries the simultaneous positive-certificate row P, failure-certificate row F, cannot-claim row R, export-grade row E, and ledger-closure row L for the same admissible claim C. A NameCert-row-preserving bijection would have to preserve that joint positive and negative audit surface, so no such bijection exists. □

Lean4 ✓ `BEDC.Derived.AuditSystemUp.AuditSystemIndependenceWitness`

Theorem 1258.7 (Audit system bridged handoff package)**.** *The admissible claim row, positive certificate row, failure certificate row, refusal row, export grade row, ledger-closure row, transport row, replay row, provenance row, and local name row of an accepted* $\mathsf{AuditSystem}^{\uparrow}$ *packet form the paper bridge handoff to the abstract audit interface. The handoff exposes exactly the finite audit surface needed by that interface, and it exports no additional registry, certificate, or ledger coordinate beyond those rows.*

Proof. Read the carrier of Definition 1258.1 as $A = (C, P, F, R, E, L, H, K, Q, N)$. The rows C, P, F, R, E, L provide the admissible claim, positive certificate, typed failure certificate, refusal boundary, export grade, and ledger closure.

The structural rows H, K, Q, N transport, replay, source, and name the same finite packet. By Definition 1258.2, Definition 1258.3, and Definition 1258.5, those rows are exactly the public audit rows preserved by the abstract audit interface.

The independence witness Definition 1258.6 excludes collapse to any one sibling registry or certificate packet. Hence the displayed rows form a bridge handoff package, not a hidden registry extension. □

The Lean follow-up file `lean4/BEDC/Derived/AuditSystemUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{AuditSystem}^{\uparrow}$ and FieldFaithful $\mathsf{AuditSystem}^{\uparrow}$ instances. The field-faithfulness witness must preserve $C, P, F, R, E, L, H, K, Q, N$ as distinct rows so positive certification, typed failure, cannot-claim refusal, export grade, and ledger closure cannot be merged.

Falsifiable prediction: If an AuditSystemUp packet accepts a claim while its failure-certificate row is active, the export-refusal theorem produces a certificate conflict rather than an exportable claim.

Independence witness: Compare claimregistrylayer, failurecertificate, registryexportconsistencygate, kernelacceptanceauditwitness: AuditSystemUp is not bijective with any one sibling because it carries both positive certificate and negative refusal rows plus export-grade ledger closure.

Closure status for $\mathsf{NameCert}_{\mathsf{AuditSystem}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditSystem}^{\uparrow}$ packet is generated from admissible-claim, positive-certificate, failure-certificate, cannot-claim, export-grade, ledger-closure, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AuditSystem}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1258.1, Definition 1258.2, Definition 1258.3, Definition 1258.4, Definition 1258.5, Definition 1258.6, Definition 1258.7, the falsifiable claim-conflict prediction, the independence witness, and the displayed TasteGate alignment marker bind the BHist, hsame, Cont, Pkg, NameCert, $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, and $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ scope used by the carrier, export-refusal, certificate-conflict, ledger, independence, sibling audit rows, and bridged handoff package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AuditSystem}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, proof of every exported claim, or registry export beyond the displayed audit packet.

Upgrade path: Obligation closure requires Lean Nontrivial and FieldFaithful TasteGate instances plus checked carrier, export-refusal, and ledger rows for $C, P, F, R, E, L, H, K, Q, N$.

Chapter 1259

A Concrete Naming Certificate for AxiomPurityGate$^{\uparrow}$

The AxiomPurityGate$^{\uparrow}$ packet is a composite over the forbidden-schema obstruction of chapter 899, the finite choice-output refusal ledger of chapter 677, the quotient-boundary packet of chapter 634, the proposition-transport boundary of chapter 566, and the positive-content closure surface of chapter 515. It realizes the axiom-purity section of subsection 9462.25.8 inside the auditable-generativity programme of chapter 9462. Those siblings expose forbidden schemas, term-stratum choice demand, quotient soundness, proposition transport, and positive-content closure; this carrier binds the dependency-audit gate that accepts only finite reports with visible forbidden-dependency refusals.

It is an ai over the forbidden-schema obstruction, finite choice-output refusal ledger, quotient-boundary packet, and proposition-transport boundary, with the additional theorem-target and replay rows serving only as the local audit interface.

The packet is a composite over theorem-target audit rows, dependency-report rows, forbidden-dependency refusal rows, replay rows, replacement rows, and transport rows, rather than an atomic carrier.

Definition 1259.1 (AxiomPurityGate carrier)**.** *An* AxiomPurityGate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, D, F, R, L, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the theorem-target row naming the declaration whose dependency report is being consumed;*
2. *D is the finite dependency-report row, listing the transitive proof dependencies visible to the audit consumer;*
3. *F is the forbidden-dependency refusal row for* Classical.choice*, quotient soundness, and propositional extensionality;*
4. *R is the replay row recording the command-level route by which the dependency report can be re-read;*
5. *L is the replacement row naming the constructive BEDC route used when a forbidden dependency request is refused;*
6. *S is the transport row recording that theorem-target, dependency-report, refusal, replay, and replacement reads are consumed componentwise;*

7. *H records componentwise* hsame *transports for T, D, F, R, L, S, P, N;*
8. *C records the displayed* Cont *routes by which consumers read the report, refusals, replay route, replacement route, and transport row;*
9. *P is the* Pkg *provenance over* $\mathsf{MetaClosureObstruction}^{\uparrow}$, $\mathsf{UnwitnessedExistenceClosure}^{\uparrow}$, $\mathsf{ChoiceRecipeLedger}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, $\mathsf{PropextTransportBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AxiomPurityGate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $T, D, F, R, L, S, H, C, P, N$. A packet missing any forbidden-dependency refusal row in F is not accepted as an $\mathsf{AxiomPurityGate}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.AxiomPurityGateUp.AxiomPurityGateTasteGate single carrier alignment`

Theorem 1259.2 (AxiomPurityGate NameCert obligations)**.** *For the carrier of Definition 1259.1, the local* $\mathsf{NameCert}_{\mathsf{AxiomPurityGate}^{\uparrow}}$ *surface consists of carrier admission for*

$$G = (T, D, F, R, L, S, H, C, P, N),$$

theorem-target exposure through T, dependency-report exposure through D, forbidden-dependency refusal through F, replay exposure through R, constructive replacement exposure through L, transport exposure through S, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 1259.1. The tuple displays the theorem target, dependency report, forbidden-dependency refusal, replay route, replacement route, transport row, componentwise stability, continuation replay, provenance, and local naming.

The rows T, D, F are the audit-facing core. Row T names the declaration being read, row D exposes the finite dependency report, and row F records the refusal of each forbidden dependency class rather than treating purity as prose outside the carrier.

The rows R, L, S make the refusal replayable and consumable. Row R records the report replay route, row L records the constructive replacement route, and row S records transport of the visible rows without adding a hidden dependency or host equality.

Finally H, C, P, N exhaust stability, continuation replay, provenance, and naming. Since no further coordinate exists, the listed rows are exactly the local NameCert obligations for the gate packet. □

Lean4 ✓ `BEDC.Derived.AxiomPurityGateUp.AxiomPurityGateNameCert obligations`

Theorem 1259.3 (AxiomPurityGate non-escape)**.** *Every accepted* $\mathsf{AxiomPurityGate}^{\uparrow}$ *packet exports only the finite rows*

$$T, D, F, R, L, S, H, C, P, N.$$

No consumer can extract from such a packet an unreported theorem dependency, a Classical.choice *dependency, a quotient-soundness dependency, a propositional-extensionality dependency, a hidden replay oracle, a host equality transport, or a proof of meta closure. A packet missing any forbidden-dependency refusal row cannot be accepted as an* $\mathsf{AxiomPurityGate}^{\uparrow}$ *carrier.*

Proof. Project the accepted packet to the carrier inventory of Definition 1259.1. The only dependency information available to a consumer is the theorem target T, the finite report D, and the forbidden-dependency refusal row F.

Apply the NameCert obligations of Definition 1259.2. Every replay, replacement, and transport read must pass through R, L, S, H, C, P, N, so a consumer may inspect the displayed audit packet but cannot bypass it to obtain a hidden dependency or host-level equality route.

The last claim is part of carrier admission. Row F contains the refusal records for the forbidden dependency classes; removing one of those records removes a displayed coordinate required by the classifier and therefore removes acceptance as an $\mathsf{AxiomPurityGate}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.AxiomPurityGateUp.AxiomPurityGateNonescape`

Theorem 1259.4 (AxiomPurityGate witness-preserving consumer)**.** *Every downstream consumer of an accepted* $\mathsf{AxiomPurityGate}^{\uparrow}$ *packet that asks for a theorem-target audit must read the dependency report, forbidden-dependency refusal, replay route, constructive replacement route, transport row, stability row, continuation route, provenance row, and local naming row through*

$$T, D, F, R, L, S, H, C, P, N.$$

Such a consumer preserves the displayed witness boundary: accepted theorem reads may cite the finite report and replacement rows, while rejected reads remain blocked by F. The consumer cannot turn the gate into a selector for witnesses, a quotient of dependency reports, a proposition-equality transport, a second replay surface, or a meta-closure proof.

Proof. First expose the theorem-target audit through T, D, F. The dependency report is visible only as the finite row D, and each forbidden dependency class is handled by its refusal record in F.

Next route replay and replacement through R, L, S. The replay row lets the report be re-read, the replacement row records the constructive BEDC route available after refusal, and the transport row keeps those reads componentwise rather than quotienting dependency reports.

Finally apply H, C, P, N. These rows preserve stability, continuation replay, package provenance, and the local name of the packet. They do not add a selector, quotient, proposition-equality transport, second replay surface, or meta-closure theorem, so the consumer remains witness-preserving. □

Lean4 ✓ `BEDC.Derived.AxiomPurityGateUp.AxiomPurityGateWitnessPreservingConsumer`

Theorem 1259.5 (AxiomPurityGate bridged handoff)**.** *The public* $\mathsf{AxiomPurityGate}^{\uparrow}$ *surface has a bridge-facing handoff whose visible rows are the theorem target T, dependency report D, forbidden-dependency refusals F, replay route R, constructive replacement route L, transport row S, stability row H, continuation row C, provenance row P, and local naming row N. The handoff packages the forbidden-dependency refusal rows of Definition 1259.3, the witness-preserving consumer of Definition 1259.4, and the TasteGate carrier alignment recorded at Definition 1259.1.*

Consequently any bridge-facing consumer must preserve the displayed refusal boundary: accepted theorem reads may replay the finite dependency report and constructive replacement route, while rejected reads remain blocked by F. The handoff does not assert Lean bridge checking, global axiom-purity for every BEDC theorem, a selector for witnesses, a quotient of dependency reports, proposition-equality transport, or meta closure.

Proof. First read the carrier and TasteGate row. The accepted packet has exactly the displayed coordinates $T, D, F, R, L, S, H, C, P, N$, and the carrier alignment binds the local gate to those coordinates rather than to an external audit oracle.

Next apply Definition 1259.3. The forbidden-dependency refusal row F is part of carrier admission, so a bridge-facing read cannot erase a refusal for Classical.choice, quotient soundness, or propositional extensionality.

Finally apply Definition 1259.4. The downstream consumer must read report, refusal, replay, replacement, transport, stability, continuation, provenance, and naming rows through the same packet. This repackages the public surface for bridge-facing use without adding a Lean bridge theorem or any of the excluded escape routes. □

Lean4 ✓ `BEDC.Derived.AxiomPurityGateUp.AxiomPurityGateBridgedHandoff`

Falsifiable prediction: A packet missing any forbidden-dependency refusal row cannot be accepted as an AxiomPurityGateUp carrier.

Independence witness: MetaClosureObstruction, UnwitnessedExistenceClosure, ChoiceRecipeLedger, QuotientSoundnessBoundary, PropextTransportBoundary: the axiom-purity gate packet contains a theorem-target row, dependency-report row, forbidden-dependency refusal row, replay row, replacement row, transport row, provenance row, and local NameCert row. MetaClosureObstruction carries forbidden-schema refusal without dependency-report replay; UnwitnessedExistenceClosure carries the positive-content closure family without a gate-local theorem target; ChoiceRecipeLedger records term-stratum recipe refusal without quotient or propext dependency rows; QuotientSoundnessBoundary isolates quotient soundness without choice and propext refusal as one audit packet; PropextTransportBoundary isolates proposition transport without a theorem-target dependency report. Hence none is bijective to $\mathsf{AxiomPurityGate}^{\uparrow}$ while preserving all displayed NameCert rows.

The future Lean target `BEDC.Derived.AxiomPurityGateUp` must place the ChapterTasteGate realization in `TasteGate.lean` with explicit `Nontrivial` and `FieldFaithful` instances for the displayed carrier and forbidden-dependency refusal rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AxiomPurityGate}^{\uparrow}}$.

Constructive story: An $\mathsf{AxiomPurityGate}^{\uparrow}$ packet is generated from a theorem target, finite dependency report, forbidden-dependency refusals, replay route, replacement route, transport row, componentwise hsame stability, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AxiomPurityGate}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1259.1, Definition 1259.2, Definition 1259.3, the witness-preserving consumer of Definition 1259.4, and the bridge-facing handoff of Definition 1259.5 bind the public-facing and bridge-facing scope to the forbidden-dependency refusal rows $T, D, F, R, L, S, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AxiomPurityGate}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not prove a Lean implementation, a standard bridge, global axiom-purity for every BEDC theorem, or a replacement proof for any particular refused dependency.

Upgrade path: No further paper closure grade remains; Lean-side axiom-purity and bridge checking remain on the verification axis.

Chapter 1260

A Concrete Naming Certificate for $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$

The $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ packet realizes the runtime-boundary claim of subsection 9462.25.5 as a finite audit surface. It binds the object-level candidate boundary of chapter 591, the runtime-request and refusal surface of chapter 890, the meta-closure obstruction of chapter 899, the structural reflection limit of chapter 586, and the claim ledger of chapter 1258 into a single NameCert packet. The carrier records an auditable boundary without converting the host proof checker into an object-layer row.

Definition 1260.1 (Runtime-boundary audit carrier)**.** *A* $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, E, M, L, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the structural self-description row, recording formation, replay, and certificate-shape data described by Definition 9462.702;*
2. *R is the runtime-candidate row inherited from* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$*, naming the object-level candidate, transition, and accepted-result boundary;*
3. *E is the runtime-reflection refusal row inherited from* $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$*, recording the request for validation and the execution refusal report;*
4. *M is the meta-closure obstruction row inherited from* $\mathsf{MetaClosureObstruction}^{\uparrow}$*, blocking conversion of the runtime request into an unwitnessed closure principle;*
5. *L is the reflection-limit row inherited from* $\mathsf{ReflectionLimitBoundary}^{\uparrow}$*, separating structural reflection from truth-total or runtime-total claims;*
6. *G is the audit-system row inherited from* $\mathsf{AuditSystem}^{\uparrow}$*, recording the admissible claim, positive or negative certificate face, export grade, and ledger closure;*
7. *H records componentwise* hsame *transport for $S, R, E, M, L, G, C, P, N$;*
8. *C records displayed* Cont *routes by which consumers read the structural description, candidate boundary, runtime refusal, obstruction, reflection limit, audit ledger, and transports;*
9. *P is the* Pkg *provenance over* $\mathsf{RuntimeCandidateBoundary}^{\uparrow}$, $\mathsf{RuntimeReflectionBoundary}^{\uparrow}$, $\mathsf{MetaClosureObstruction}$ $\mathsf{ReflectionLimitBoundary}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, *and* $\mathsf{TasteGate}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{RuntimeBoundaryAudit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $S, R, E, M, L, G, H, C, P, N$. No accepted carrier has an additional row for host-runtime execution, runtime validation, or a BEDC truth predicate.

Theorem 1260.2 (Runtime-boundary audit NameCert obligations)**.** *For the carrier of Definition 1260.1, the local* $\mathsf{NameCert}_{\mathsf{RuntimeBoundaryAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (S, R, E, M, L, G, H, C, P, N)$, structural-description exactness through S, runtime-candidate accountability through R, runtime-reflection refusal through E, meta-closure obstruction through M, reflection-limit boundary through L, audit-ledger closure through G, and transport, consumer, provenance, and local naming closure through H, C, P, N.*

Proof. Carrier admission is projection from the finite packet of Definition 1260.1. The rows $S, R, E, M, L, G, H, C, P, N$ are the complete displayed surface.

The rows S, R, E, M, L, G supply the substantive audit coordinates. They record structural self-description, the object-level candidate boundary, runtime-request refusal, meta-closure obstruction, reflection-limit separation, and claim-ledger status.

The rows H, C, P, N transport, replay, package, and name the same packet. Since those rows only preserve the displayed coordinates, they cannot add a hidden validator, a runtime-total row, or a truth predicate. □

Theorem 1260.3 (Runtime-boundary audit validation non-escape)**.** *Every accepted* $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ *packet exports runtime-boundary evidence only through the finite rows $S, R, E, M, L, G, H, C, P, N$. A consumer asking for host-runtime validation must therefore encounter the refusal row E, the meta-closure obstruction row M, and the reflection-limit row L before any audit export through G.*

Proof. Let $A = (S, R, E, M, L, G, H, C, P, N)$ be an accepted carrier. By Definition 1260.1, every consumer route is displayed in C and every route is transported by H over the same coordinates.

A runtime-validation request is first classified as an object-level candidate read by R. It is then recorded as a runtime-reflection request and refusal by E.

The request cannot bypass the obstruction and limit rows. Row M blocks the conversion of the request into an unwitnessed closure principle, while row L blocks the conversion of structural reflection into a truth-total or runtime-total claim.

The audit-system row G can export only the resulting finite claim status. It has no coordinate for replacing E, M, L by host execution, so validation does not escape the displayed boundary packet. □

Theorem 1260.4 (Runtime-boundary audit consumer boundary)**.** *Every consumer of an accepted* $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ *packet that reports structural self-description must preserve the distinction between the structural row S and the runtime-validation refusal path R, E, M, L. In particular, the consumer cannot certify self-description by replacing R, E, M, L with a host-checker row outside the packet.*

Proof. Consumer access is mediated by the displayed route row C. Reading the structural description therefore begins at S and remains inside the same finite carrier.

The runtime-facing path is separate. It passes through the candidate row R, the runtime-reflection refusal row E, the meta-closure obstruction row M, and the reflection-limit row L.

The provenance and local naming rows P, N name that route as part of the same packet, and H transports it componentwise. None of these rows supplies a host-checker coordinate. The consumer boundary is therefore exactly the displayed separation between S and R, E, M, L. □

Lean4 ✓ `BEDC.Derived.RuntimeBoundaryAuditUp.RuntimeBoundaryAuditTasteGate single carrier alignment`

Falsifiable prediction: Any accepted RuntimeBoundaryAuditUp packet that exports a self-description claim while omitting the runtime-reflection refusal row, meta-closure obstruction row, or reflection-limit row fails the carrier classifier.

Independence witness: runtimecandidateboundary, runtime-reflection-boundary, metaclosure-obstruction, reflectionlimitboundary, auditsystem: the runtime-boundary audit carrier is not bijective to any single sibling because it simultaneously carries structural self-description, runtime-candidate accountability, runtime-reflection refusal, meta-closure obstruction, reflection-limit separation, audit-ledger closure, and one local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RuntimeBoundaryAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ packet is generated from structural self-description, runtime-candidate accountability, runtime-reflection refusal, meta-closure obstruction, reflection-limit separation, audit-ledger closure, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuntimeBoundaryAudit}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1260.1, Definition 1260.2, Definition 1260.3, Definition 1260.4, and

Lean4 ✓ `BEDC.Derived.RuntimeBoundaryAuditUp.taste gate`

bind the finite runtime-boundary audit carrier, validation non-escape, and consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuntimeBoundaryAudit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RuntimeBoundaryAuditUp.RuntimeBoundaryAuditTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not internalize the host proof checker, provide a BEDC truth predicate, claim runtime-total validation, or bridge $\mathsf{RuntimeBoundaryAudit}^{\uparrow}$ to a standard schema.

Upgrade path: Bridge checking requires a standard-schema bridge for the displayed runtime-boundary audit packet.

Chapter 1261

A Concrete Naming Certificate for $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$

The $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ packet realizes the logic and meta-loop boundary of section 9462.25 as a finite NameCert carrier. It sits after the quotient refusal surface of chapter 634 and the failed audit witness of chapter 901. Those siblings refuse representative collapse and expose failed gate diagnostics, while the present packet binds proof-pattern discipline, local refutation, open meta-loop refusal, audit-gate failure, transport, route replay, provenance, and local naming before any consumer may read a logical closure claim.

Definition 1261.1 (Logic contradiction meta-loop carrier)**.** *A* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, R, M, A, T, C, G, N)$$

with the following displayed rows:

1. *P is the finite proof-pattern row recording constructive formation, introduction, elimination, and local refutation steps;*
2. *R is the local-refutation row that records a witnessed contradiction only as a bounded failure of the displayed proof pattern;*
3. *M is the open meta-loop refusal row, rejecting total truth, unrestricted choice, quotient collapse, and halting closure as consumer-readable coordinates;*
4. *A is the audit-gate row, with failed reads routed through the* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *surface;*
5. *T records componentwise* hsame *transport for proof-pattern, refutation, meta-loop refusal, audit gate, route, provenance, and local naming rows;*
6. *C records the displayed* Cont *replay from proof-pattern inspection through local refutation and open meta-loop refusal into the audit gate;*
7. *G is the* Pkg *provenance over* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$*,* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$*,* $\mathsf{AuditGateFailureWitne}$ BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LogicContradictionMetaLoop}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows P, R, M, A, T, C, G, N and has no coordinate for total truth, global choice, quotient equality elimination, or a halting oracle.

Lean4 ✓ BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoopTasteGate single carrier alignment

Theorem 1261.2 (Logic contradiction meta-loop NameCert obligations)**.** *For every accepted* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ *carrier*

$$L = (P, R, M, A, T, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{LogicContradictionMetaLoop}^{\uparrow}}$ *surface consists exactly of proof-pattern admission through* P*, local-refutation exposure through* R*, open meta-loop refusal through* M*, audit-gate failure routing through* A*, componentwise transport through* T*, displayed continuation replay through* C*, package provenance through* G*, local naming through* N*, and non-admission of total truth, unrestricted choice, quotient collapse, and halting closure.*

Lean4 ✓ BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoopCarrier namecert obligations

Proof. Project the carrier of Definition 1261.1. Each admitted proof-pattern, refutation, meta-loop refusal, audit-gate, transport, continuation, provenance, and naming row is a displayed coordinate of the finite packet.

The quotient side is already constrained by chapter 634: a quotient-like consumer receives a refusal-and-transport surface rather than a representative selector or a Quot.sound row. The present carrier records that refusal in M and transports it only through T, C, G, N.

The audit side is constrained by chapter 901: a failed gate read must expose its diagnostic row before it reaches a refusal target. The present carrier records that dependency in A and does not promote the failed item to theorem content.

Thus the local NameCert surface is exhausted by the eight displayed rows. There is no remaining coordinate from which a consumer could read total truth, unrestricted choice, quotient collapse, or halting closure. □

Theorem 1261.3 (Logic contradiction meta-loop carrier admission)**.** *Carrier admission for* LogicContradictionMet*is exactly the finite projection of* $L = (P, R, M, A, T, C, G, N)$ *through proof-pattern discipline* P*, local refutation* R*, open meta-loop refusal* M*, audit-gate routing* A*, and the structural rows* T, C, G, N*. A packet that supplies only a contradiction report, only an audit failure, or only a quotient refusal is not admitted as this carrier.*

Lean4 ✓ BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoopCarrierAdmission

Proof. Project the definition of the carrier. The rows P, R, M, A jointly supply proof-pattern formation, bounded refutation, open meta-loop refusal, and audit-gate diagnostics.

The structural rows are not optional. Componentwise hsame transport T, displayed Cont replay C, Pkg provenance G, and local naming N keep the same finite packet available to consumers.

Thus carrier admission is the whole eight-row packet. A narrower item may be a sibling route, but it does not expose the local proof-pattern, refutation, refusal, audit, transport, replay, provenance, and naming surface at once. □

Theorem 1261.4 (Logic contradiction meta-loop ledger nonescape)**.** *Every refusal-ledger route of an accepted* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ *packet must pass through open meta-loop refusal* M*, audit-gate row* A*, transport* T*, replay* C*, provenance* G*, and local naming* N*. Transport and replay preserve the ledger route; they do not export total truth, unrestricted choice, quotient collapse, halting closure, or a closed meta-loop classifier.*

Proof. Start with the refusal-bearing rows. The open meta-loop row M records refusal of total truth, choice, quotient collapse, and halting closure, while A routes failed reads through the audit-gate surface.

Transport and replay keep this ledger finite. The row T compares the same refusal and audit coordinates componentwise, and C replays the route from proof-pattern inspection through local refutation into the audit gate.

Finally G records provenance and N supplies local naming. Since no row in the route is a truth oracle, choice principle, quotient-elimination rule, halting oracle, or closed classifier, the refusal ledger cannot escape the displayed packet. □

Lean4 ✓ `BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoopCarrier ledger nonescape`

Theorem 1261.5 (Logic contradiction meta-loop non-escape)**.** *An accepted* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ *carrier cannot export a route from local contradiction management to total truth, unrestricted choice, quotient equality elimination, global proof search, halting closure, or a closed meta-loop classifier. Every accepted consumer route must preserve the displayed packet* P, R, M, A, T, C, G, N.

Proof. Read the NameCert obligations of Definition 1261.2. The only content rows are proof pattern, local refutation, open meta-loop refusal, and audit gate; the remaining rows transport, replay, package, and name those same coordinates.

If a consumer asks for quotient collapse, the request is forced through the quotient-soundness refusal surface cited in the obligation proof. If a consumer asks to turn a failed audit item into theorem content, the request is forced through the audit-gate failure witness and remains refusal data.

The carrier supplies no row for total truth, unrestricted choice, global proof search, halting closure, or a closed meta-loop classifier. Therefore every accepted route remains inside P, R, M, A, T, C, G, N. □

Lean4 ✓ `BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoopCarrier nonescape`

Theorem 1261.6 (Logic contradiction meta-loop sibling dependency)**.** *The carrier of Definition 1261.1 is not bijective, row for row, with* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, *quotient-soundness boundary packets, audit-gate failure packets, MetaCIC decidable-boundary packets, MetaCIC confluence-audit packets, or normal-false-exclusion packets. The present packet jointly carries proof-pattern discipline and open meta-loop refusal, while each sibling exposes only a quotient gate, an audit diagnostic, a checker boundary, a confluence audit, or a local false-exclusion surface.*

Lean4 ✓ `BEDC.Derived.LogicContradictionMetaLoopUp.LogicContradictionMetaLoop sibling dependency route`

Proof. Compare first with $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$. That sibling refuses representative selection and replaces quotient-like reads by displayed psame transport, but it does not carry the proof-pattern row P, the local contradiction row R, or the full open meta-loop refusal row M.

Compare next with $\mathsf{AuditGateFailureWitness}^{\uparrow}$. That sibling records a failed gate item, diagnostic readback, refusal target, transport, replay, provenance, and local naming, but it does not carry the quotient-collapse refusal or the proof-pattern discipline as one logic-facing carrier.

The remaining checker and confluence siblings are narrower still. A decidable-boundary packet concerns a checker boundary, a confluence-audit packet concerns rewrite or normalization audit,

and a normal-false-exclusion packet concerns one local false-exclusion surface. None contains all rows P, R, M, A, T, C, G, N, so no row-preserving bijection can identify it with $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$.

□

Later Lean work supplies the Nontrivial and FieldFaithful instances for $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ in TasteGate.lean. Those instances must read the same finite carrier rows P, R, M, A, T, C, G, N and may not add total truth, unrestricted choice, quotient collapse, halting closure, or a closed meta-loop classifier.

Falsifiable prediction: Any accepted consumer route that reads total truth, choice, quotient collapse, or halting closure without passing through the open meta-loop refusal row refutes the packet.

Independence witness: quotientsoundnessboundary, auditgatefailurewitness, metacicdecidable-boundary, metacicconfluenceaudit, normalfalseexclusion: none is bijective because this packet jointly carries proof-pattern discipline and open-meta-loop refusal rather than only one gate or one checker boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{LogicContradictionMetaLoop}^{\uparrow}}$.

Constructive story: A $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ packet is generated from finite proof-pattern discipline, local-refutation evidence, open meta-loop refusal, audit-gate routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LogicContradictionMetaLoop}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1261.1, Definition 1261.2, Definition 1261.5, Definition 1261.6, and the existing TasteGate marker bind the finite carrier, NameCert obligations, nonescape surface, and sibling dependency route as the obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LogicContradictionMetaLoop}^{\uparrow}}, \mathsf{unformalized})$

Ledger evidence: BEDC.Meta.DiscoveryDeltaLedger.logicContradictionMetaLoopDeltaLedger

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, total truth, unrestricted choice, quotient collapse, halting closure, global proof search, or a public standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, open meta-loop refusal, audit-gate routing, transport stability, route replay, provenance, local naming, TasteGate instances, and non-escape for the same finite packet.

Chapter 1262

A Concrete Naming Certificate for $\mathsf{ForwardBindingGapLedger}^{\uparrow}$

The $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ packet realizes forward-binding belief as a finite NameCert surface over displayed closed-observation, GAP-boundary, refusal-ledger, apophatic-socket, and belief rows. It is deliberately composite: the carrier assembles the closed observation record of chapter 616, the source/refusal boundary of chapter 1006, the axiom-refusal ledger of chapter 1057, the socket boundary of chapter 536, and the observation-update surface of chapter 289. Its role is not to present a new atomic TasteGate object, but to package the present commitment row and the visible GAP row that keep future universal-closure use auditable.

Definition 1262.1 (Forward-binding GAP ledger carrier)**.** *A* $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, R, G, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the forward-binding commitment row, recording the present use of a universal-closure claim whose witness is not wholly contained in the current record;*
2. *R is the finite record ledger supplied by the closed-observation surface;*
3. *G is the GAP socket row that keeps the unsupported universal-closure part visible;*
4. *S is the noninternalization and refusal row, routing hidden substrate-axiom, hidden induction-principle, and internal-object readings to the displayed refusal surface;*
5. *H records componentwise* hsame *transports for K, R, G, S and for the structural rows;*
6. *C records the displayed* Cont *routes from commitment use to record readback, GAP-socket exposure, refusal readback, and downstream citation;*
7. *P records* Pkg *provenance over* $\mathsf{ClosedObservationSystem}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, $\mathsf{AxiomRefusalLedger}^{\uparrow}$, $\mathsf{ApophaticSocket}^{\uparrow}$, $\mathsf{Belief}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}$ *row.*

The carrier exports only the finite commitment, record, GAP-socket, refusal, transport, route, provenance, and naming rows. It exports no universal closure, hidden induction principle, substrate axiom, or internal object for T.

Lean4 ✓ `BEDC.Derived.ForwardBindingGapLedgerUp.ForwardBindingGapLedgerTasteGate single carrier alignment`

Theorem 1262.2 (Forward-binding GAP ledger NameCert obligations)**.** *For the carrier of Definition 1262.1, the local* $\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $F = (K, R, G, S, H, C, P, N)$*, forward-binding commitment exposure through* K*, finite record accountability through* R*, GAP-socket preservation through* G*, refusal and noninternalization through* S*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows K, R, G, S, H, C, P, N are the only rows available to the local certificate.

Read the closed-observation and GAP-boundary siblings. The closed-observation chapter supplies the finite record ledger R, while the GAP-boundary chapter keeps a closure request on the displayed source/refusal split rather than on an internal universal-closure row.

Attach the refusal and socket rows. The axiom-refusal and apophatic-socket siblings make unsupported supply visible as a refusal surface, and the belief sibling supplies the present commitment row without changing it into a substrate theorem.

Finally use H, C, P, N. These rows transport, replay, package, and name the same finite packet, so the NameCert obligations are exhausted by the displayed route. □

Lean4 ✓ `BEDC.Derived.ForwardBindingGapLedgerUp.ForwardBindingGapLedgerNameCertObligations`

Theorem 1262.3 (Forward-binding GAP ledger noninternalization)**.** *Every accepted* $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ *packet keeps the forward-binding commitment outside substrate-internal theoremhood. A downstream consumer may cite* K, R, G, S, H, C, P, N *as a finite commitment ledger, but it receives no universal closure, hidden induction principle, substrate axiom, hidden theorem, host assumption, or internal object for* T*.*

Lean4 ✓ `BEDC.Derived.ForwardBindingGapLedgerUp.ForwardBindingGapLedgerNoninternalization`

Proof. Begin with the carrier inventory of Definition 1262.1. The commitment row K is paired with the finite record ledger R and the GAP socket G before any downstream citation can use it.

Use the GAP-boundary refusal surface of Definition 1006.2 and Definition 1006.3. Unsupported closure requests remain on the displayed GAP and refusal rows rather than becoming internal universal closure.

Now apply the axiom-refusal noninternalization theorem, Definition 1057.3. The refusal row S is therefore a visible ledger row, not a consumable axiom or hidden theorem.

The remaining support rows H, C, P, N only preserve that route under transport, replay, provenance, and local naming. Hence a consumer can cite the finite forward-binding ledger but cannot extract an internal object for T or any hidden closure principle from it. □

Theorem 1262.4 (Forward-binding GAP ledger sibling dependency)**.** *The* $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ *carrier depends on the joint availability of closed-observation records, GAP-boundary refusal, axiom-refusal noninternalization, apophatic-socket exposure, and belief commitment rows. No one sibling among* $\mathsf{ClosedObservationSystem}^{\uparrow}$*,* $\mathsf{GapClosureBoundary}^{\uparrow}$*,* $\mathsf{AxiomRefusalLedger}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*, or* $\mathsf{Belief}^{\uparrow}$ *carries all of* K, R, G, S, H, C, P, N*, and no row-preserving reduction to one sibling preserves the local* $\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}$ *surface.*

Proof. Compare first with the closed-observation and belief siblings. $\mathsf{ClosedObservationSystem}^{\uparrow}$ supplies finite record rows, and $\mathsf{Belief}^{\uparrow}$ supplies observation-update commitment rows, but neither by itself carries the GAP socket and refusal ledger required to keep forward binding noninternal.

Compare next with the GAP-boundary, axiom-refusal, and apophatic-socket siblings. They expose unsupported closure requests, refusal rows, and socket boundaries, but none of them also carries the present commitment row K together with the closed record ledger R and the full structural support H, C, P, N.

Thus the chapter is a composite NameCert surface. The dependence on its siblings is visible in the carrier, and the composite packet refuses universal closure, hidden induction, substrate axioms, and internal objects for T precisely because no single sibling row can replace the whole finite route. □

Lean4 ✓ `BEDC.Derived.ForwardBindingGapLedgerUp.ForwardBindingGapLedgerCarrier sibling dependency`

Closure status for $\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ packet is generated from a forward-binding commitment row, finite closed-observation record ledger, GAP socket, noninternalization and refusal row, componentwise hsame transports, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1262.1, Definition 1262.2, Definition 1262.3, and Definition 1262.4 bind the finite composite carrier, NameCert obligations, noninternalization boundary, and sibling dependency surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForwardBindingGapLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close universal future cases, export an induction principle, supply a substrate axiom, internalize T, assert Lean verification, or bridge to a standard schema.

Upgrade path: Obligation closure requires checked carrier admission, record-ledger accountability, GAP-socket preservation, refusal noninternalization, transport stability, route replay, provenance, and local naming for the same finite composite packet.

Chapter 1263

A Concrete Naming Certificate for $\mathsf{CofinalStreamTailSelector}^{\uparrow}$

The $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ packet records the finite cofinal-window route used by the Real and Completion construction of chapter 9238. It starts from a precision request, opens a displayed $\mathsf{StreamName}^{\uparrow}$ tail window, hands that window through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ readback rows, reaches the $\mathsf{Real}^{\uparrow}$ seal, and records selector provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming in one finite BHist carrier. The packet is a concrete NameCert surface for the L10 construction order of section 9467.12; it is not a quotient stream, host-selected limit, ambient completeness principle, countable-choice rule, or hidden selector.

The one-source tail witness of chapter 458, the dyadic cofinal route of chapter 680, and the diagonal consumer route of chapter 697 are sibling surfaces. The present packet binds their shared cofinal-window role into one StreamName-tail selector carrier before any $\mathsf{Real}^{\uparrow}$ consumer can read the completion-facing seal.

Definition 1263.1 (Cofinal StreamName tail selector carrier)**.** *A* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$$

with the following displayed rows:

1. *ϵ is the positive precision request made by the completion-facing consumer;*
2. *W is the finite cofinal* $\mathsf{StreamName}^{\uparrow}$ *tail window opened at that request;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row read from the displayed window;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row that records the finite dyadic bounds consumed by the handoff;*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, R, D are displayed;*
6. *σ records selector provenance: the request, cofinal window index, selected tail segment, and sibling selector rows used by the packet;*
7. *H records componentwise* hsame *transports for the request, window, regular handoff, dyadic readback, seal, selector provenance, package provenance, and local naming rows;*
8. *C records the displayed* Cont *route from ϵ through W, R, D, σ into A;*

9. *P is the* Pkg *provenance over* CofinalStreamTailSelector$^{\uparrow}$, RegularCauchyTailSelector$^{\uparrow}$, DyadicCofinalTailSelector$^{\uparrow}$, DiagonalTailSelector$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*

10. *N is the local* NameCert$_{\text{CofinalStreamTailSelector}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the visible rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorCarrier route rows`

Theorem 1263.2 (Cofinal StreamName tail selector NameCert obligations)**.** *The local* NameCert$_{\text{CofinalStreamTailSelect}}$ *surface for Definition 1263.1 consists of carrier admission for $Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$, precision-request exposure through ϵ, cofinal* StreamName$^{\uparrow}$ *tail-window exposure through W,* RegSeqRat$^{\uparrow}$ *handoff through R,* DyadicRatCore$^{\uparrow}$ *readback through D,* Real$^{\uparrow}$ *seal delivery through A, selector provenance through σ, componentwise* hsame *stability through H, displayed* Cont *route replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed tuple in Definition 1263.1. The request, cofinal window, regular handoff, dyadic readback, seal, selector provenance, transports, routes, provenance, and naming row are all visible coordinates of one finite packet.

The selector obligations are read in order. The precision request ϵ opens only the displayed StreamName tail window W, the RegSeqRat handoff R and dyadic readback D are both downstream of that finite window, and the selector provenance row σ records the sibling selector rows used by the same route.

The RealUp seal row A is downstream of W, R, D, σ, while H, C, P, N provide transport, route replay, provenance, and local naming. Since the carrier has no coordinate for quotient streams, host-selected limits, countable choice, hidden tail selection, or ambient completeness, these rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorCarrier namecert obligations`

Theorem 1263.3 (Cofinal StreamName tail selector seal order)**.** *For an accepted* CofinalStreamTailSelector$^{\uparrow}$ *carrier*

$$Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N),$$

every completion-facing read of the Real$^{\uparrow}$ *seal row A first passes through the finite cofinal* StreamName$^{\uparrow}$ *tail window W, the* RegSeqRat$^{\uparrow}$ *handoff row R, the* DyadicRatCore$^{\uparrow}$ *readback row D, and the selector provenance row σ. The seal route cannot skip directly from the precision request to a host-selected limit.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorSealOrder`

Proof. Project the carrier. The seal row A appears only after the window, handoff, dyadic readback, and selector provenance rows in the displayed tuple.

Use the NameCert obligations of Definition 1263.2. They force any accepted read of the seal to preserve the same request, finite window, regular handoff, dyadic readback, selector provenance, transports, continuation route, package provenance, and local naming row.

Thus every completion-facing seal read factors through W, R, D, σ before A is available. There is no coordinate in the carrier for a direct host limit or an ambient completeness argument. □

Theorem 1263.4 (Cofinal StreamName tail selector TasteGate obligation)**.** *The taste gate for* CofinalStreamTailSelector$^{\uparrow}$ *requires Lean to provide the local* Nontrivial *and* FieldFaithful *instances in* lean4/BEDC/Derived/CofinalStreamTailSelectorUp/TasteGate.lean*. The intended instances must read the same finite carrier rows* $\epsilon, W, R, D, A, \sigma, H, C, P, N$ *and may not add a quotient stream, host limit, ambient completeness theorem, countable-choice selector, or hidden tail oracle.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorTasteGate single carrier alignment`

Proof. The carrier of Definition 1263.1 supplies the finite support on which the taste gate is allowed to act. Nontriviality is witnessed only by the presence of the displayed request-to-window-to-seal route, and field faithfulness is witnessed only by the listed rows and their componentwise transports.

Any Lean instance for the taste gate must therefore be local to the same carrier surface. Adding quotient streams, host limits, ambient completeness, countable choice, or hidden selector rows would create data not present in Q and would fail the NameCert obligation surface of Definition 1263.2. □

Theorem 1263.5 (Cofinal StreamName tail selector obligation surface)**.** *For an accepted* CofinalStreamTailSelector$^{\uparrow}$ *carrier* $Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$*, the obligation surface exposes the precision request* ϵ*, the displayed* StreamName$^{\uparrow}$ *tail window* W*, the* RegSeqRat$^{\uparrow}$ *handoff* R*, the* DyadicRatCore$^{\uparrow}$ *readback* D*, the* Real$^{\uparrow}$ *seal* A*, the selector provenance row* σ*, the componentwise* hsame *row* H*, the displayed* Cont *route* C*, the package provenance* P*, and the local naming row* N *in one finite packet. The surface is read through the sibling routes of chapter 458, chapter 680, chapter 697, and chapter 38.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorObligationSurface`

Proof. Project the carrier of Definition 1263.1. The request, tail window, regular handoff, dyadic readback, seal, provenance, transport, replay, package, and name are all visible coordinates of one finite packet.

The proof-route order is already fixed by the carrier: the precision request opens the displayed StreamName tail window, the RegSeqRat and DyadicRatCore rows follow from that finite window, and the Real seal is downstream of the same route.

The sibling citations identify the shared tail surface, while H, C, P, N preserve the same route componentwise. Since the carrier has no coordinate for quotient streams, host-selected limits, countable choice, hidden tail selection, or ambient completeness, the displayed rows exhaust the obligation surface. □

Theorem 1263.6 (Cofinal StreamName tail selector stream route)**.** *For every accepted* CofinalStreamTailSelector$^{\uparrow}$ *carrier* $Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$*, the stream route factors through* W, R, D, σ *before the* Real$^{\uparrow}$ *seal row* A *is public. A consumer may cite the route only after the shared sibling surfaces of chapter 458, chapter 680, chapter 697, and chapter 38 have already displayed the same finite route.*

Proof. Use the obligation surface theorem above. It fixes the displayed rows and places the Real seal downstream of the cofinal StreamName tail window, the regular handoff, and the dyadic readback.

The sibling routes supply the same cofinal-window role in different presentations. Since H, C, P, N only transport, replay, package, and name that route, a consumer cannot bypass the shared finite packet before reading A. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorStreamRoute`

Theorem 1263.7 (Cofinal StreamName tail selector real seal nonescape)**.** *For an accepted* CofinalStreamTailSelecto *carrier* $Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* A *does not export quotient stream identity, host real equality, host-selected limits, countable-choice selection, hidden family limits, or an ambient completeness principle. The seal remains a finite boundary for the same cofinal StreamName route displayed in the sibling tail-selector chapters.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorRealSealNonescape`

Proof. Project the carrier. The seal row A is downstream of the finite cofinal tail window W, the regular handoff R, the dyadic readback D, and the selector provenance row σ.

Apply the seal-order theorem of Definition 1263.3. It forces every Real-facing read through the same finite cofinal route before A is available.

Since no extra coordinate is present in the carrier, the seal row cannot export quotient stream identity, host equality, selected limits, countable choice, hidden family limits, or ambient completeness. □

Theorem 1263.8 (Cofinal StreamName tail selector non-escape)**.** *An accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple* $\epsilon, W, R, D, A, \sigma, H, C, P, N$*. In particular, it exports no quotient stream identity, host real equality, host-selected limit, unbounded tail search, countable-choice selector, hidden family limit, or ambient RealUp completeness principle.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorNonescape`

Proof. The NameCert obligations identify every row available to a downstream consumer. The precision request, finite StreamName tail window, RegSeqRat handoff, dyadic readback, RealUp seal, and selector provenance rows are the only content rows, and H, C, P, N only transport, replay, package, and name those rows.

Apply Definition 1263.3 to any real-facing read. It forces the read through the finite cofinal window, regular handoff, dyadic readback, and selector provenance before the seal row can be consumed.

Since no additional coordinate is present, the packet has no source for quotient stream identity, host real equality, host-selected limits, unbounded tail search, countable choice, hidden family limits, or an ambient RealUp completeness principle. □

Theorem 1263.9 (Cofinal StreamName tail selector obligation package)**.** *The* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier, local* NameCert *obligations, seal-order theorem, TasteGate obligation, and non-escape theorem assemble the L10 handoff surface over*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Every completion-facing seal read is therefore routed through the displayed cofinal tail window, regular-sequence handoff, dyadic readback, selector provenance, componentwise transport, continuation route, package provenance, and local naming row. The assembled package gives no bypass through quotient stream equality, host real equality, host-selected limits, countable choice, hidden tail selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorObligationPackage`

Proof. Use the carrier of Definition 1263.1 and the obligations of Definition 1263.2. These identify the finite rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$ and bind them over the StreamName, RegSeqRat, DyadicRatCore, RealUp, history, transport, continuation, package, and naming surfaces.

Apply Definition 1263.3. It forces every $\mathsf{Real}^{\uparrow}$ seal read through W, R, D, σ before A is available, so the handoff is a finite cofinal-window route rather than a direct host-limit read.

Finally combine the TasteGate obligation of Definition 1263.4 with non-escape from Definition 1263.8. The taste gate may only read the displayed carrier rows, and the non-escape theorem excludes quotient stream equality, countable choice, hidden selectors, and ambient completeness from the exported surface. □

Theorem 1263.10 (Cofinal StreamName tail selector public export)**.** *Every accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *public consumer receives only the precision request, displayed* $\mathsf{StreamName}^{\uparrow}$ *tail window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, selector provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows of the carrier. The public export is routed through the obligation package, seal order, and non-escape boundary, and it uses the TasteGate site of Definition 1263.4 only as the local finite-carrier discipline.*

Proof. Start with the obligation package of Definition 1263.9. It assembles the carrier, local NameCert obligations, seal-order theorem, TasteGate obligation, and non-escape theorem over the finite rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$.

Apply the seal-order theorem Definition 1263.3. It forces every public read of the $\mathsf{Real}^{\uparrow}$ seal through the precision request, StreamName tail window, RegSeqRat handoff, DyadicRatCore readback, and selector provenance before the seal is visible.

Apply the non-escape theorem Definition 1263.8. It removes quotient stream identity, host real equality, host-selected limits, unbounded tail search, countable choice, hidden family limits, and ambient RealUp completeness from the public surface.

The TasteGate obligation, already recorded at Definition 1263.4, supplies only the local finite-carrier discipline for the same rows. Therefore the accepted public consumer receives exactly the listed BEDC rows and no additional tail-selection or completion principle. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorCarrier public export`

Theorem 1263.11 (Cofinal StreamName tail selector scoped route)**.** *For every accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier*

$$Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N),$$

the route

$$\epsilon, W, R, D, A, \sigma, H, C, P, N$$

is scoped over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The precision request* ϵ *opens only the displayed StreamName tail window* W*;* R *and* D *read regular-sequence and dyadic rows from that window;* A *is the terminal Real seal;* σ, H, C, P, N *record selector provenance, componentwise transport, continuation replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorScopedRoute`

Proof. Start from the carrier and obligation package of Definition 1263.9. They expose exactly the rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$ and list the BEDC surfaces on which those rows may be read.

The carrier order fixes the scope. The request row opens the finite $\mathsf{StreamName}^{\uparrow}$ window; the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are read from that same displayed window; and the $\mathsf{Real}^{\uparrow}$ seal is reached only after these rows and selector provenance are present.

The remaining rows do not introduce new analytic data. They transport, replay, source, and name the same finite route through BHist, hsame, Cont, Pkg, and NameCert, so the accepted packet is scoped by the listed kernel rows. □

Theorem 1263.12 (Cofinal StreamName tail selector window readback scope)**.** *In an accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier* $Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N)$*, the cofinal window row* W *and readback rows* R, D, σ *remain ordered before any consumer read of the* $\mathsf{Real}^{\uparrow}$ *seal row* A*. No accepted route can inspect* A *before the finite window, regular handoff, dyadic readback, and selector provenance have been displayed.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorWindowReadbackScope`

Proof. Apply the seal-order theorem Definition 1263.3. It places the Real seal downstream of W, R, D, σ for every completion-facing read.

Use the scoped-route theorem Definition 1263.11 to keep the same ordering inside the BEDC row classes. The rows H, C, P, N only preserve the already displayed route and cannot move the seal ahead of the cofinal window or readback rows. □

Theorem 1263.13 (Cofinal StreamName tail selector public certificate)**.** *The carrier of Definition 1263.1, the obligation package of Definition 1263.9, the seal order of Definition 1263.3, the scoped route of Definition 1263.11, and the window readback scope of Definition 1263.12 export a public certificate for the cofinal-window route over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The certificate exposes only the displayed rows*

$$\epsilon, W, R, D, A, \sigma, H, C, P, N$$

and exports no quotient stream identity, host-selected limit, countable-choice selector, hidden tail oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and obligation package. They identify the accepted packet as the finite row tuple $\epsilon, W, R, D, A, \sigma, H, C, P, N$ and bind those rows to StreamName, RegSeqRat, DyadicRatCore, RealUp, history, transport, continuation, package, and naming surfaces.

Apply the seal-order theorem to the Real-facing boundary. It places A downstream of the finite cofinal window W, the regular handoff R, the dyadic readback D, and selector provenance σ.

Use the scoped-route theorem and the window-readback scope theorem to make the same ordering public. These results keep the route inside $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert and prevent the seal from moving ahead of the displayed readback rows.

The carrier has no coordinate for quotient stream identity, host-selected limits, countable choice, hidden tail oracles, or ambient Real completeness. Therefore the assembled route is exactly the stated public certificate. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorPublicCertificate`

Theorem 1263.14 (Cofinal StreamName tail selector public window transport)**.** *For every accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *carrier*

$$Q = (\epsilon, W, R, D, A, \sigma, H, C, P, N),$$

any public transport of the cofinal selector window preserves the order

$$\epsilon \to W \to R \to D \to \sigma \to A$$

through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The rows* H, C, P, N *only transport, replay, source, and name the same finite route. Such a transport cannot replace the window by quotient stream identity, a host-selected limit, countable choice, a hidden selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public certificate of Definition 1263.13. It exposes exactly the displayed packet rows and already refuses quotient stream identity, host-selected limits, countable-choice selectors, hidden tail oracles, and ambient Real completeness.

Apply the scoped route of Definition 1263.11. The precision request opens the $\mathsf{StreamName}^{\uparrow}$ window W, the regular and dyadic rows R, D read that window, selector provenance σ remains part of the displayed route, and the $\mathsf{Real}^{\uparrow}$ seal A is reached only afterward.

Use the window readback scope of Definition 1263.12 together with the seal-order theorem of Definition 1263.3. These keep every public transport from moving A ahead of $\epsilon, W, R, D, \sigma$.

The remaining rows H, C, P, N have only transport, replay, source, and naming roles in the same finite packet. They therefore preserve the already displayed route rather than selecting a new tail, adding a limit, or appealing to a completeness principle outside the carrier. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorPublicWindowTransport`

Theorem 1263.15 (Cofinal StreamName tail selector bridge surface)**.** *The public certificate of Definition 1263.13 and the public-window transport of Definition 1263.14 determine the exact bridge surface available to any future standard bridge for* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$. *That surface consists of the precision request, cofinal* $\mathsf{StreamName}^{\uparrow}$ *tail window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, selector provenance,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows of the accepted carrier.*

No standard bridge may export quotient stream equality, a host-selected limit, a countable-choice selector, a hidden tail oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from this public surface.*

Proof. Start with the public certificate. It exposes exactly the rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$ and refuses quotient stream identity, host-selected limits, countable choice, hidden tail oracles, and ambient Real completeness.

Apply the scoped route, seal-order, and window-readback scope theorems. They place the precision request before the cofinal $\mathsf{StreamName}^{\uparrow}$ window, route $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ readback through that window, keep selector provenance visible, and allow the $\mathsf{Real}^{\uparrow}$ seal only after those rows have been displayed.

Use public-window transport to preserve the same route across any public handoff. The rows H, C, P, N only transport, replay, source, and name the displayed carrier, while the non-escape theorem excludes every off-packet tail selector or completeness principle. Therefore the listed rows are the whole bridge surface available to a future standard bridge. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorBridgeSurface`

Theorem 1263.16 (Cofinal StreamName tail selector Real-seal bridge)**.** *Any Real-facing bridge from an accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *packet factors the* $\mathsf{Real}^{\uparrow}$ *seal through the displayed precision request, cofinal* $\mathsf{StreamName}^{\uparrow}$ *tail window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback, selector provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and*

local NameCert *row. The bridge may cite the public certificate and public-window transport only in this order, and it cannot replace the route by quotient stream equality, a hidden tail selector, a host limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorRealSealBridge`

Proof. Start with the public certificate of Definition 1263.13. It exposes exactly the accepted rows $\epsilon, W, R, D, A, \sigma, H, C, P, N$ and places the Real seal inside the same finite packet as the precision request, cofinal window, regular handoff, dyadic readback, selector provenance, transport, replay, provenance, and local naming row.

Apply the public-window transport theorem of Definition 1263.14. It preserves the order $\epsilon \to W \to R \to D \to \sigma \to A$ across any public handoff, so a Real-facing bridge cannot move the seal ahead of the displayed window or readback rows.

Use the scoped route of Definition 1263.11, the seal-order theorem of Definition 1263.3, and the window-readback scope of Definition 1263.12. These results keep the bridge inside $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with A downstream of W, R, D, σ.

Finally apply the bridge-surface theorem of Definition 1263.15. Since that surface has no coordinate for quotient stream equality, hidden tail selectors, host limits, countable choice, or ambient Real completeness, every Real-facing bridge must factor the seal through the displayed finite route. □

Theorem 1263.17 (Cofinal StreamName tail selector bridge stability)**.** *Let a bridge-stable consumer read an accepted* $\mathsf{CofinalStreamTailSelector}^{\uparrow}$ *packet through the public bridge surface. Then the consumer preserves the displayed order*

$$\epsilon \to W \to R \to D \to \sigma \to A$$

from precision request to $\mathsf{Real}^{\uparrow}$ *seal. It cannot replace any segment of that route by quotient stream equality, a host limit, countable choice, a hidden selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public certificate of Definition 1263.13. It exposes exactly the accepted carrier rows and already refuses quotient stream identity, host-selected limits, countable-choice selectors, hidden tail oracles, and ambient Real completeness.

Apply public-window transport, Definition 1263.14. This preserves the order $\epsilon \to W \to R \to D \to \sigma \to A$ across public handoffs, so a bridge-stable consumer cannot move the seal ahead of the displayed window, handoff, readback, or selector-provenance rows.

Finally use the bridge surface and Real-seal bridge, Definition 1263.15 and Definition 1263.16. These force the bridge to read the same finite surface before the $\mathsf{Real}^{\uparrow}$ seal is consumed, and the listed excluded objects have no coordinate in that surface. □

Lean4 ✓ `BEDC.Derived.CofinalStreamTailSelectorUp.CofinalStreamTailSelectorCarrier bridge stability`

Falsifiable prediction: A selector packet that lacks a displayed cofinal StreamName tail window is not an accepted CofinalStreamTailSelectorUp carrier.

Independence witness: regularcauchytailselector, dyadiccofinaltailselector, diagonaltailselector, realcompletionselectorseal: these siblings expose tail, dyadic, diagonal, or seal rows separately, while CofinalStreamTailSelectorUp binds all of them with StreamName cofinal-window selection and RealUp seal handoff, so no row-preserving bijection exists.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalStreamTailSelector}^{\uparrow}}$.

Constructive story: A CofinalStreamTailSelector$^{\uparrow}$ packet is generated from a precision request, a displayed cofinal StreamName tail window, RegSeqRat handoff, DyadicRatCore readback, RealUp seal delivery, selector provenance, componentwise hsame transports, displayed Cont routes, package provenance, and a local naming row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CofinalStreamTailSelector}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1263.10 exports the accepted cofinal-window selector packet as a public Real-completion surface over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CofinalStreamTailSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, RealUp completeness, quotient stream equality, host real limits, countable choice, hidden tail selection, or a public standard bridge.

Upgrade path: Bridged closure requires a standard bridge for the public cofinal-window selector surface without quotient streams, host limits, countable choice, hidden selectors, or ambient RealUp completeness.

Chapter 1264

A Concrete Naming Certificate for $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$

The $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ packet is the finite Real-and-Completion handoff that binds two cofinal modulus schedules to one normalized shared-window route. It sits after the normalized comparison surface of chapter 778, consumes the cofinal modulus seal row of chapter 1198, and hands its displayed rows to the four Real-and-Completion anchors chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a terminal finite NameCert handoff for a paired cofinal route; it is not a quotient of regular streams, a host equality of reals, a selected limiting representative, a hidden tail oracle, or an ambient completeness principle.

Definition 1264.1 (Cofinal modulus normalization seal carrier)**.** *A* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, M, W, D, R, E, H, C, P, L, N)$$

with the following displayed rows:

1. *A and B are the two cofinal modulus schedules whose terminal Real route is being normalized.*
2. *M is the accepted* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *row that records the paired cofinal modulus surface.*
3. *W is the normalized shared* $\mathsf{StreamName}^{\uparrow}$ *window row obtained after the regular Cauchy modulus normalizer has exposed the common window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read on that shared window.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the same normalized window and tolerance data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after M, W, D, R have been displayed.*
7. *H stores componentwise* hsame *transport for A, B, M, W, D, R, E and the local support rows.*
8. *C stores the displayed* Cont *replay from the paired cofinal schedules through M, then through W, D, R, E, and then into the local certificate.*
9. *P is the* Pkg *provenance over* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, $\mathsf{RegularCauchyModulusNor}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *L is the terminal ledger row that records the Real-and-Completion handoff boundary.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for side-specific modulus replacement, quotient regular streams, host real equality, selected limits, detached completion evidence, countable choice, or ambient completeness.

Definition 1264.2 (Cofinal modulus normalization seal classifier)**.** *Two accepted* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ *carriers classify together exactly when their paired cofinal schedules, cofinal modulus seal row, normalized shared window, dyadic tolerance ledger, paired regular readback, terminal Real seal, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, terminal ledger, and local naming row agree componentwise along the same normalized handoff route. The classifier does not identify packets by comparing host modulus functions, by quotienting regular streams, by forgetting the shared window, or by replacing the terminal seal with an external completeness claim.*

Theorem 1264.3 (Cofinal modulus normalization seal NameCert obligations)**.** *For every accepted* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ *carrier*

$$S = (A, B, M, W, D, R, E, H, C, P, L, N),$$

the local $\mathsf{NameCert}_{\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}}$ *surface consists exactly of carrier admission, paired-schedule exposure through* A, B*, cofinal modulus seal exposure through* M*, normalized shared-window coverage through* W*, dyadic tolerance coverage through* D*, paired regular readback through* R*, Real seal coverage through* E*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, terminal handoff ledgering through* L*, local naming through* N*, and non-escape for side-specific modulus replacement, quotient streams, host equality, selected limits, detached completion evidence, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 1264.1. The tuple displays two cofinal schedules, one cofinal modulus seal row, one normalized shared window, one dyadic ledger, one paired readback, one Real seal, transport, replay, provenance, terminal ledgering, and the local name.

The paired-schedule and cofinal-seal obligations read A, B, M before any terminal Real handoff is available. The sibling normalizer of chapter 778 supplies the shared-window discipline: once M is fixed, the route exposes one W, one D, and one R rather than two side-specific windows.

The Real-and-Completion anchors then determine the terminal handoff. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies readbacks, and $\mathsf{Real}^{\uparrow}$ supplies the seal row, all inside the same displayed packet.

Finally H, C, P, L, N repack the displayed route as a local NameCert surface. Since the carrier lacks side-specific modulus replacement, quotient streams, host equality, selected limits, detached completion evidence, countable choice, and ambient completeness coordinates, the listed rows exhaust the obligation surface. □

Lean4 ✓ `BEDC.Derived.CofinalModulusNormalizationSealUp.CofinalModulusNormalizationSeal namecert obligations`

Theorem 1264.4 (Cofinal modulus normalization seal Real-completion handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ *carrier factors through the finite route*

$$(A, B) \longmapsto M \longmapsto (W, D, R) \longmapsto (E, H, C, P, L, N).$$

In particular, any diagonal-limit route that consumes two cofinal schedules reads the normalized shared-window row before it reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.CofinalModulusNormalizationSealUp.CofinalModulusNormalizationSeal real completion handoff`

Proof. Start with the NameCert obligations. They expose A, B as the paired cofinal schedules and M as the only cofinal modulus seal row available to the consumer.

Next apply the shared-window discipline inherited from chapter 778. The route from M to the Real-facing surface is forced through W, D, R, so the consumer obtains one $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and one paired $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The terminal rows E, H, C, P, L, N then seal the handoff. The $\mathsf{Real}^{\uparrow}$ row is downstream of the normalized shared-window readback, and the remaining rows only transport, replay, source, ledger, and name that displayed route. Thus no diagonal-limit consumer can reach the Real seal before factoring through the normalized shared-window row. □

Falsifiable prediction: If the normalized cofinal modulus seal is atomic, then any RealUp diagonal-limit route consuming two cofinal schedules factors through its normalized shared-window row before reaching the RealUp seal.

Independence witness: regular-cauchy-modulus-normalizer, cofinal-tail-budget, dyadic-cofinal-tail-selector, real-tail-agreement-seal: the new carrier binds paired cofinal schedules, normalized shared-window data, and terminal RealUp seal handoff in one packet, while each sibling lacks at least one of those rows.

Lean follow-up provides explicit $\mathsf{Nontrivial}\ \mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CofinalModulusNo}$ instances in `lean4/BEDC/Derived/CofinalModulusNormalizationSealUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}}$ **.**

Constructive story: The cofinal modulus normalization seal is generated from $BHist$ rows that record paired cofinal schedules, normalized shared-window data, a normalization classifier, and the RealUp seal handoff that consumes the normalized route. The carrier first reads the cofinal modulus evidence, then applies the classifier to identify the normalized seal, and finally permits Cont extensions only when Real and completion consumers see that same normalized packet. The $hsame$ relation transports the normalized seal without changing the cofinal route. The prediction and independence witness identify one shared normalized carrier, rather than two independent routes to Real and completion surfaces.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the cofinal modulus normalization-seal carrier, classifier, NameCert obligations, Real-completion handoff theorem, falsifiable prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim a separate completion operator, quotient-stream equality, or normalization outside the displayed cofinal modulus route.

Upgrade path: Reaching obligationClosure requires formal normalized cofinal-modulus rows and checked Real-completion handoff through the classifier.

Chapter 1265

A Concrete Naming Certificate for $\mathsf{ClosureStatusReference}^{\uparrow}$

The $\mathsf{ClosureStatusReference}^{\uparrow}$ packet realizes the closure-status reference layer of subsection 9462.35.8 as a finite NameCert carrier for registry-bound export status. It sits after the registry export surface of chapter 1216, the consistency gate of chapter 1224, the refusal registry of chapter 1248, the theorem-gap registry of chapter 1246, and the concept registry surface of chapter 1252. Those siblings expose export rows, consistency checks, refusal rows, theorem gaps, and concept entries; this packet binds the displayed closure-status reference that caps the strength of each exported registry read.

Definition 1265.1 (Closure status reference carrier)**.** *A* $\mathsf{ClosureStatusReference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, F, L, B, N, U, H, C, P, Q)$$

with the following displayed rows:

1. *S is the constructive-story row describing how the target packet is generated from finite BEDC data.*
2. *T is the theory-closure row recording the paper-side grade and its closed scope.*
3. *F is the formal-status row recording the Lean-side grade and the canonical target when one is displayed.*
4. *L is the Lean-target row, empty exactly when the reference records an unformalized or statement-only site.*
5. *B is the bridge-status row recording whether no bridge, a paper bridge, or a checked bridge is available.*
6. *N is the not-claimed row recording the surfaces that the reference refuses to export.*
7. *U is the next-grade row recording the route needed before the registry may advertise a stronger status.*
8. *H stores componentwise* hsame *transport for S, T, F, L, B, N, U.*
9. *C stores displayed* Cont *replay from registry export rows through the closure-status bound.*
10. *P is the* Pkg *provenance over* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{RefusalRegistry}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{ClosureStatusReference}^{\uparrow}}$ *row.*

The carrier has no coordinate for a registry export stronger than the displayed reference, for merging the theory and verification axes, for erasing the bridge-status row, or for bypassing the not-claimed and next-grade rows.

Theorem 1265.2 (Closure status reference export bound)**.** *Every registry export read through an accepted* $\mathsf{ClosureStatusReference}^{\uparrow}$ *carrier is bounded by the displayed rows*

$$S, T, F, L, B, N, U, H, C, P, Q.$$

In particular, an export row cannot report a theory grade stronger than T, a formal grade stronger than F, or a bridge grade stronger than B unless the same carrier also replaces the reference rows and their next-grade route.

Proof. Project the carrier of Definition 1265.1. The only status coordinates are the theory row T, formal row F, Lean-target row L, bridge row B, not-claimed row N, and next-grade row U.

Read the registry siblings through C and P. The export surface supplies the public read, the consistency gate checks it against the same reference, the refusal and theorem-gap registries record blocked or open surfaces, and the concept registry supplies the named target row.

Since H, C, P, Q only transport, replay, source, and name those displayed rows, no consumer receives an extra status coordinate. Any stronger export would need a row outside R, so it is not accepted by this carrier. □

Theorem 1265.3 (Closure status reference nonescape)**.** *An accepted* $\mathsf{ClosureStatusReference}^{\uparrow}$ *packet exports only constructive story, two-axis status, Lean-target display, bridge status, not-claimed surface, next-grade route, transport, replay, provenance, and local naming. It does not export theorem truth, global registry correctness, host proof search, bridge completion, or status promotion outside those displayed fields.*

Proof. Inspect the fields of Definition 1265.1. They list the constructive story row, theory-closure row, formal-status row, Lean-target row, bridge-status row, not-claimed row, next-grade row, and structural support rows.

Apply Definition 1265.2. The registry export can read only those rows, and the consistency gate can compare only against the same displayed reference.

The excluded claims would require a truth row, global-correctness row, host-search row, checked-bridge row, or status-promotion row not present in R. Hence they cannot escape through the $\mathsf{ClosureStatusReference}^{\uparrow}$ carrier. □

Falsifiable prediction: A registry export that reports a status stronger than the displayed closure-status reference is not accepted by the $\mathsf{ClosureStatusReference}^{\uparrow}$ carrier.

Independence witness: registryexportsurface, registryexportconsistencygate, refusalregistry, theoremgapregistry, conceptregistrysurface: the registry export surface lacks the full constructive-story and next-grade packet, the consistency gate lacks the not-claimed and bridge-status rows, the refusal registry lacks positive two-axis status, the theorem-gap registry lacks the complete closure reference, and the concept registry surface lacks the export-bound reference; none carries the full constructive-story, two-axis status, bridge-status, not-claimed, and next-grade packet.

The Lean TasteGate follow-up must include explicit $\mathsf{Nontrivial}\ \mathsf{ClosureStatusReference}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ClosureStatusReference}^{\uparrow}$ instances in `lean4/BEDC/Derived/ClosureStatusReferenceUp/TasteGate.`

The proofs must exhibit a witness pair for the carrier and prove field projections for $S, T, F, L, B, N, U, H, C, P, Q$ so the two axes, bridge status, not-claimed row, and next-grade route remain separate.
Lean4 ✓ `BEDC.Derived.ClosureStatusReferenceUp.ClosureStatusReferenceTasteGate single carrier alignment`

Theorem 1265.4 (Closure status reference obligation surface)**.** *For every accepted* $\mathsf{ClosureStatusReference}^{\uparrow}$ *carrier* $R = (S, T, F, L, B, N, U, H, C, P, Q)$*, the obligation surface consists exactly of carrier admission, two-axis status separation, Lean-target display, bridge-status row, not-claimed row, next-grade route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ClosureStatusReference}^{\uparrow}}$ *row. In particular,* T *and* F *remain separate,* B *remains a visible bridge-status coordinate,* N *remains the refusal boundary, and* U *remains the only next-grade route.*

Proof. Project the carrier of Definition 1265.1. The displayed rows $S, T, F, L, B, N, U, H, C, P, Q$ are the only accepted coordinates for the reference.

Use Definition 1265.2. A registry export is bounded by the same theory row T, formal row F, Lean-target row L, bridge row B, not-claimed row N, and next-grade row U, so no field can be promoted by replay alone.

Finally apply Definition 1265.3. Transport by H, replay by C, provenance by P, and naming by Q preserve the displayed rows but cannot add theorem truth, bridge completion, status promotion, or global registry correctness. The listed rows therefore exhaust the local obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosureStatusReference}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosureStatusReference}^{\uparrow}$ packet is generated from constructive-story, theory-closure, formal-status, Lean-target, bridge-status, not-claimed, next-grade, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosureStatusReference}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1265.1, Definition 1265.2, Definition 1265.3, and Definition 1265.4 bind the mature paper package from the carrier, export-bound, nonescape, obligation-surface, bridge-status reference, not-claimed row, and next-grade route; Lean verification is recorded separately by the TasteGate carrier-alignment target.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosureStatusReference}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.ClosureStatusReferenceUp.ClosureStatusReferenceTasteGate\_single\_carrier\_alignment`

Bridge status: paperBridge

Origin: ai

Not claimed: This chapter does not close bridge checking, theorem truth, global registry correctness, host proof search, or status promotion outside the displayed reference.

Upgrade path: The remaining upgrade is a machine-checked standard bridge for the displayed closure-status reference rows, not an additional paper bridge.

Chapter 1266

A Concrete Naming Certificate for FiniteObservationLock$^{\uparrow}$

The FiniteObservationLock$^{\uparrow}$ packet realizes the subject-free observation thesis of section 9462.22 as a finite NameCert carrier. It sits beside the real observation budget of chapter 611, the finite observation-budget selector of chapter 782, and the observation-budget limiter of chapter 903. Those siblings expose consumer service, selected finite budgets, and capped window routes; this packet locks the observation itself to a displayed continuation, history-sameness comparison, finite budget, readback, and no-subject ledger before any Real- or Completion-facing route may consume it.

Definition 1266.1 (Finite observation lock carrier)**.** *A* FiniteObservationLock$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, D, Z, Q, W, R, B, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the antecedent history row of the observed continuation;*
2. *D is the displayed distinction or continuation-tail row;*
3. *Z is the successor history row reached from A through D;*
4. *Q is the subject-free ledger row recording that the observation signature contains no observer-substance parameter, owner map, or separate viewpoint carrier;*
5. *W is the finite* StreamName$^{\uparrow}$ *window surface through which the observation becomes readable by finite consumers;*
6. *R is the readback row consumed by* RegSeqRat$^{\uparrow}$ *or another finite observation reader after W is displayed;*
7. *B is the finite observation-budget row supplied through* RealObservationBudget$^{\uparrow}$, FiniteObservationBudgetSelector$^{\uparrow}$ *or* ObservationBudgetLimiter$^{\uparrow}$*;*
8. *U is the downstream use row for the Real- or Completion-facing consumer;*
9. *H records componentwise* hsame *transport for A, D, Z, Q, W, R, B, U and the local certificate row;*
10. *C records displayed* Cont *replay from A through D to Z, and from the locked observation row through W, R, B to U;*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationLock}^{\uparrow}$, $\mathsf{RealObservationBudget}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{ObservationBudgetLimiter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationLock}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden subject, host observation, quotient stream, selected limit, ambient completeness principle, detached readback, or observer identity outside the finite packet L.

Theorem 1266.2 (Finite observation lock NameCert obligations)**.** *For the carrier of Definition 1266.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationLock}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $L = (A, D, Z, Q, W, R, B, U, H, C, P, N)$;*
2. *continuation and history-sameness lock, requiring the observed event to be read through the displayed* $\mathsf{Cont}(A, D, Z)$ *route and transported only by the componentwise* hsame *row H;*
3. *finite budget and readback lock, requiring W, R, B to be displayed before the downstream use row U is consumed;*
4. *no-subject ledger exposure, requiring Q to block subject-substance parameters, owner maps, viewpoint carriers, and host observation rows;*
5. *non-escape, excluding hidden subjects, quotient streams, selected limits, ambient completeness, detached readbacks, and observer identities not carried by the finite packet.*

Proof. Carrier admission is projection from the packet in Definition 1266.1. The visible observation rows are $A, D, Z, Q, W, R, B, U, H, C, P, N$ and there is no other coordinate.

The continuation and history-sameness lock reads the observation as the displayed route from A through D to Z. The transport row H can only compare the displayed rows componentwise, so it preserves the locked observation rather than creating a hidden observer surface.

The finite budget and readback lock then routes finite consumers through W, R, B before U is available. The sibling budget surfaces provide window service, selected finite budgets, and limiter caps, but the lock packet requires those rows to remain attached to the same continuation event.

The no-subject ledger row Q records the negative side of subject-free observation. It refuses a subject parameter, an owner map, a separate viewpoint carrier, and a host observation row because none is part of the accepted carrier.

Finally, C, P, N replay, source, and name the same rows. Since the carrier contains no quotient stream, selected limit, ambient completeness principle, detached readback, or external observer identity coordinate, the five obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationLockUp.FiniteObservationLockTasteGate single carrier alignment`

Theorem 1266.3 (Finite observation lock consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$*, completion-facing, finite-budget, or observer-ledger consumer of an accepted* $\mathsf{FiniteObservationLock}^{\uparrow}$ *packet factors through the finite route*

$$A, D, Z, Q, W, R, B, U, H, C, P, N.$$

The consumer receives the displayed continuation, the no-subject ledger row, finite window and readback rows, the finite budget row, the downstream use row, and the structural transport, replay, provenance, and local naming rows. It receives no hidden subject, host observation, quotient stream, selected limit, ambient completeness datum, detached readback, or observer identity outside the packet.

Proof. Start with the accepted carrier $L = (A, D, Z, Q, W, R, B, U, H, C, P, N)$. The continuation route A, D, Z is the only observation event in the carrier, and Q is the only row that records subject-free admissibility.

Apply Definition 1266.2. The finite budget and readback lock forces any downstream use to pass through W, R, B before U is read, while the continuation and history-sameness lock keeps those rows attached to the same displayed observation.

The Real observation budget, finite observation-budget selector, and observation-budget limiter siblings may consume the window, readback, and budget rows. They do not supply a second observation carrier, because the packet gives them only W, R, B, U, H, C, P, N downstream of the locked continuation and no-subject ledger.

Thus every accepted consumer route is exhausted by the displayed packet. The excluded objects require coordinates absent from L, so they cannot be projected from an accepted $\mathsf{FiniteObservationLock}^{\uparrow}$ read. □

Falsifiable prediction: A purported finite observation read that omits the displayed Cont route or budget readback is not an accepted FiniteObservationLockUp carrier.

Independence witness: realobservationbudget, finiteobservationbudgetselector, observationbudgetlimiter, streamname, real: each sibling supplies a consumer, selector, limiter, stream, or seal row, but none is bijective to the packet that locks observation to Cont, hsame, finite budget, readback, and no-subject ledger rows.

The Lean follow-up in `lean4/BEDC/Derived/FiniteObservationLockUp/TasteGate.lean` will provide $\mathsf{Nontrivial\ FiniteObservationLock}^{\uparrow}$ and $\mathsf{FieldFaithful\ FiniteObservationLock}^{\uparrow}$ instances with explicit witness rows. The field-faithfulness witness must preserve $A, D, Z, Q, W, R, B, U, H, C, P, N$ as distinct rows before `BEDC.Derived.FiniteObservationLockUp.FiniteObservationLockTasteGate_single_c` can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationLock}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationLock}^{\uparrow}$ packet is generated from an antecedent history, displayed distinction, successor history, no-subject ledger, finite window, readback, budget, downstream use, componentwise hsame transport, Cont replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationLock}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1266.1, Definition 1266.2, and Definition 1266.3 bind subject-free observation to a finite continuation, budget, readback, and ledger packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationLock}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, every Real or Completion consumer route, total observer-identity theory, or ambient analytic completeness.

Upgrade path: Lean closure requires Nontrivial FiniteObservationLockUp and FieldFaithful FiniteObservationLockUp instances in `lean4/BEDC/Derived/FiniteObservationLockUp/TasteGate.lean`.

Chapter 1267

A Concrete Naming Certificate for $\mathsf{TraditionRegistry}^{\uparrow}$

The $\mathsf{TraditionRegistry}^{\uparrow}$ packet realizes the tradition registry of subsection 9462.34.5 as a finite NameCert carrier. It sits after the auditable-generativity context of chapter 9462, beside the core registry index of chapter 1251, the concept-registry surface of chapter 1252, and the theorem-gap registry of chapter 1246. These siblings index concepts, expose concept surfaces, and record theorem gaps; the present packet binds one inherited tradition comparison to the rejected-surplus and comparison-discipline rows required before that comparison is named.

Definition 1267.1 (Tradition registry carrier)**.** *A* $\mathsf{TraditionRegistry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, S, P, R, H, C, Q, N)$$

with the following displayed rows:

1. *T is the tradition row, naming the inherited tradition and the traditional claim being compared.*
2. *S is the BEDC-site row, naming the concrete BEDC chapter, definition, theorem, or registry surface where the comparison is allowed to land.*
3. *P is the preserved-structure row, recording the exact constructor, classifier, transport, or obligation shape preserved by the comparison.*
4. *R is the rejected-surplus row, recording the doctrine, host semantics, prose reading, metaphysical claim, or ambient assumption not imported into the BEDC site.*
5. *H records componentwise* hsame *transport for the tradition, site, preserved-structure, rejected-surplus, continuation, provenance, and naming rows.*
6. *C records the displayed* Cont *route from the inherited comparison through the BEDC site and rejected-surplus boundary.*
7. *Q is the* Pkg *provenance row over philosophy tradition entries,* $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, S, P, R, H, C, Q, N. It has no coordinate for inherited doctrine without rejected surplus, a BEDC site without preserved structure, a hidden comparison rule, or a sibling registry bijection.

Lean4 (definition) `BEDC.Derived.TraditionRegistryUp.TraditionRegistryUp`

Theorem 1267.2 (Tradition registry NameCert obligations)**.** *For the carrier of Definition 1267.1, the local* $\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}$ *surface consists exactly of carrier admission for* $X = (T, S, P, R, H, C, Q, N)$*, tradition and BEDC-site exposure through* T, S*, preserved-structure exposure through* P*, rejected-surplus exposure through* R*, componentwise transport through* H*, displayed comparison replay through* C*, package provenance through* Q*, and local naming through* N*.*

Proof. Project the finite carrier. The rows T, S, P, R display the inherited tradition, the BEDC landing site, the structure preserved by the comparison, and the surplus reading rejected at that site.

The support rows H, C, Q, N then transport, replay, source, and name the same comparison packet. They do not add a second doctrine row, a second landing site, or an unrecorded comparison rule.

Thus every accepted NameCert obligation is one of the displayed rows. Since the carrier contains no further coordinate, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}$ surface. □

Theorem 1267.3 (Tradition registry rejected-surplus nonescape)**.** *Every accepted* $\mathsf{TraditionRegistry}^{\uparrow}$ *packet must display the rejected-surplus row* R *before a tradition comparison can be exported as a BEDC comparison. A packet with* T, S, P, H, C, Q, N *but no* R *is outside the registry surface.*

Proof. Use the obligation surface of Definition 1267.2. It requires the preserved structure and the rejected surplus to be displayed in the same finite packet before the local name is public.

The siblings named by the provenance row can index a concept, expose a concept surface, record a theorem gap, route a claim, or audit the packet. None of them manufactures the missing rejected-surplus coordinate.

Therefore an export without R is not an accepted $\mathsf{TraditionRegistry}^{\uparrow}$ packet. It has left the finite registry carrier rather than completing it. □

Theorem 1267.4 (Tradition registry sibling dependency)**.** *The sibling packets* $\mathsf{CoreRegistryIndex}^{\uparrow}$*,* $\mathsf{ConceptRegistrySurface}^{\uparrow}$*,* $\mathsf{TheoremGapRegistry}^{\uparrow}$*,* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*, and* $\mathsf{AuditSystem}^{\uparrow}$ *are dependencies of* $\mathsf{TraditionRegistry}^{\uparrow}$ *only through the displayed provenance and replay rows* Q, C*. None is bijective with the carrier* $X = (T, S, P, R, H, C, Q, N)$*.*

Proof. Read the carrier coordinates. The tradition row, BEDC-site row, preserved-structure row, and rejected-surplus row are local to X and are not replaced by a concept index, theorem-gap row, claim row, or audit row.

The siblings enter through Q as package provenance and through C as displayed replay. Those rows state where the comparison can be checked and consumed, but they do not erase T, S, P, R or reconstruct them from a sibling alone.

Hence each sibling is a non-bijective dependency. The registry packet remains the finite tradition-comparison surface with its own rejected-surplus and naming rows. □

Falsifiable prediction: Any accepted TraditionRegistryUp packet without a rejected-surplus row is rejected before the comparison can be exported as a BEDC tradition-registry entry.

Independence witness: coreregistryindex, conceptregistrysurface, theoremgapregistry, claim-registrylayer, auditsystem: no sibling is bijective because each exposes only an index, concept surface, theorem-gap row, claim route, or audit route, while TraditionRegistryUp binds tradition,

BEDC site, preserved structure, rejected surplus, hsame transport, Cont replay, Pkg provenance, and local NameCert in one packet.

The later Lean `TasteGate.lean` for this chapter must include explicit `Nontrivial TraditionRegistryUp` and `FieldFaithful TraditionRegistryUp` instances preserving T, S, P, R, H, C, Q, N as separate displayed rows.

Theorem 1267.5 (Tradition registry scoped dependency rows)**.** *Every scoped* $\mathsf{TraditionRegistry}^{\uparrow}$ *read binds the tradition row, BEDC-site row, preserved-structure row, rejected-surplus row, transport row, replay row, provenance row, and local naming row to the displayed* BHist, hsame, Cont, Pkg, NameCert, $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, *and* $\mathsf{AuditSystem}^{\uparrow}$ *surfaces. The scope is exactly this finite registry route; it does not export inherited doctrine without the rejected-surplus row, a BEDC landing site without preserved-structure exposure, or a hidden comparison rule outside the named sibling registries.*

Lean4 ✓ `BEDC.Derived.TraditionRegistryUp.taste gate`

Closure status for $\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}$.

Constructive story: A $\mathsf{TraditionRegistry}^{\uparrow}$ packet is generated from tradition, BEDC-site, preserved-structure, rejected-surplus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1267.1, Definition 1267.2, Definition 1267.3, Definition 1267.4, Definition 1267.5, the falsifiable prediction, the independence witness, and the TasteGate single-carrier marker bind the mature paper surface for the tradition-comparison NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TraditionRegistry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, philosophical correctness of every tradition comparison, or completeness of the sibling registry lattice.

Upgrade path: Lean closure requires the public tradition-comparison surface to be matched by Nontrivial TraditionRegistryUp and FieldFaithful TraditionRegistryUp instances in `lean4/BEDC/Derived/TraditionRegistryUp/TasteGate.lean`.

Chapter 1268

A Concrete Naming Certificate for $\mathsf{TraditionComparisonBoundary}^{\uparrow}$

The $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ packet realizes the structural-comparison discipline of section 9462.29 as a finite NameCert boundary. It sits beside the claim registry layer of chapter 1209, the cannot-claim registry of chapter 554, the object-knowledge certificate of chapter 1218, and the registry export consistency gate of chapter 1224. Those siblings regulate claim export, refusal rows, object-knowledge licensing, and registry consistency; this packet binds tradition source, BEDC landing, rejected surplus, and structural distinction before any comparison may be exported.

Definition 1268.1 (Tradition comparison boundary carrier)**.** *A* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, L, R, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the tradition-source row, recording the inherited tradition, text family, question, vocabulary surface, and comparison prompt.*
2. *L is the BEDC landing row, recording the carrier, relation, classifier, ledger, certificate, gap, or T-position onto which the structural core lands.*
3. *R is the rejected-surplus row, recording the doctrinal, metaphysical, host-semantic, or identity claim not imported by the comparison.*
4. *D is the structural-distinction row, recording that the accepted comparison preserves a structural landing while refusing doctrinal identity.*
5. *H stores componentwise* hsame *transport for source, landing, rejected-surplus, distinction, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay for the comparison route from source row through landing row and rejected-surplus inspection.*
7. *P is the* Pkg *provenance over* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TraditionComparisonBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for tradition membership alone, inherited doctrinal truth, host-semantic identity, or comparison export outside the displayed packet $T = (S, L, R, D, H, C, P, N)$.

Lean4 (definition) `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundaryCarrier`

Definition 1268.2 (Tradition comparison boundary classifier)**.** *The* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ *classifier accepts exactly those packets of Definition 1268.1 whose comparison is read only through the structural landing row L together with the rejected-surplus row R. The classifier does not compare traditions by doctrinal identity, inherited vocabulary, host truth, or prose affinity unless those readings are rejected by R and kept outside the accepted landing.*

Lean4 (definition) `BEDC.Derived.TraditionComparisonBoundaryUp.traditionComparisonBoundaryClassifier`

Theorem 1268.3 (Tradition comparison boundary NameCert obligations)**.** *For the carrier of Definition 1268.1, the local* $\mathsf{NameCert}_{\mathsf{TraditionComparisonBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for $T = (S, L, R, D, H, C, P, N)$; tradition-source exposure through S; BEDC landing exposure through L; rejected-surplus exposure through R; structural-distinction exactness through D; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundaryTasteGate single carrier alignment`

Proof. Project the finite packet of Definition 1268.1. Row S displays the inherited source, while row L displays the exact BEDC landing surface available to the comparison.

The rejected-surplus row R and the structural-distinction row D are then read together. By the classifier of Definition 1268.2, acceptance depends on structural landing and rejected surplus, so a doctrinal or host-semantic identity claim has no accepted coordinate.

Finally H, C, P, N transport, replay, package, and name the same packet. Since no additional row can export tradition membership, inherited doctrine, or prose affinity as a BEDC identity, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{TraditionComparisonBoundary}^{\uparrow}}$ surface. □

Theorem 1268.4 (Tradition comparison boundary rejects doctrinal identity)**.** *No accepted* TraditionComparisonBou *packet exports doctrinal identity between an inherited tradition and a BEDC object. Every accepted comparison must display both the BEDC landing row L and the rejected-surplus row R in the same carrier.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary no doctrinal identity`

Proof. Let an accepted packet be

$$T = (S, L, R, D, H, C, P, N).$$

The classifier of Definition 1268.2 reads the packet only through L and R for comparison acceptance.

If a route exports doctrinal identity, it must either bypass R or treat R as empty surplus. Both moves leave the displayed carrier of Definition 1268.1: R is the row that records what is not imported by the comparison.

The structural-distinction row D records that the comparison preserves a landing without identifying the tradition with the BEDC object. Transport, replay, package provenance, and naming rows preserve that packet; they do not add a doctrinal-identity coordinate. □

Theorem 1268.5 (Tradition comparison boundary sibling independence)**.** *The* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$, *or* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary sibling independence`

Proof. Compare the displayed rows. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ regulates package-level claim export, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$ records cannot-claim refusal, $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ licenses object knowledge, and $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ checks registry export consistency.

None of those sibling packets carries the tradition-source row S, BEDC landing row L, rejected-surplus row R, structural-distinction row D, and local $\mathsf{NameCert}_{\mathsf{TraditionComparisonBoundary}^{\uparrow}}$ row N in one finite comparison packet.

A NameCert-row-preserving bijection would have to preserve those rows as a joint comparison boundary. Since each sibling lacks at least one of the source, landing, surplus, or distinction coordinates, no such bijection exists. □

Theorem 1268.6 (Tradition comparison boundary source registry obligation)**.** *Every accepted* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ *packet that is read as a comparison boundary must keep the tradition-source row* S *visible before any* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *export row or local* $\mathsf{NameCert}_{\mathsf{TraditionComparisonBoundary}^{\uparrow}}$ *row is consumed. The source registry obligation is the displayed route*

$$S, L, R, D, H, C, P, N$$

inside the same finite BHist *packet; it is not a free tradition-membership claim, inherited doctrinal truth, host-semantic identity, or prose-affinity license.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary source registry obligation`

Proof. Project the carrier of Definition 1268.1. The tradition-source row S is a displayed coordinate of the same packet that also contains the landing, rejected-surplus, distinction, transport, replay, provenance, and local naming rows.

Read the classifier of Definition 1268.2. Acceptance is mediated by the BEDC landing row L and rejected-surplus row R, so S cannot be consumed as a detached tradition-membership certificate.

Finally use the NameCert obligations of Definition 1268.3. Since local naming and package provenance are downstream from the displayed carrier rows, every registry-facing export keeps S visible inside S, L, R, D, H, C, P, N. □

Theorem 1268.7 (Tradition comparison boundary export consistency obligation)**.** *Every export of an accepted* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ *comparison to a* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *consumer must display the BEDC landing row* L*, rejected-surplus row* R*, structural-distinction row* D*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *together with the source row* S*. The export may cite registry consistency only for that finite packet; it cannot export doctrinal identity, host truth, unrecorded source equivalence, or object-knowledge authority outside* S, L, R, D, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary export consistency obligation`

Proof. Start from the finite obligation surface of Definition 1268.3. It makes S, L, R, D, H, C, P, N the full local certificate surface for the comparison packet.

Use sibling independence, Definition 1268.5. The registry export gate is a consumer of the displayed comparison boundary, not a replacement for the source, landing, surplus, and distinction rows.

The transport, replay, provenance, and local naming rows then keep the export tied to the same packet. Since the carrier has no coordinate for doctrinal identity, host truth, detached source equivalence, or object-knowledge authority, registry consistency cannot export any of those endpoints. □

Theorem 1268.8 (Tradition comparison boundary cannot-claim obligation)**.** *Every accepted* TraditionComparisonB *packet retains the cannot-claim boundary carried by its rejected-surplus row R before any comparison is exported through* AxisZeckendorfCannotClaim$^{\uparrow}$*,* ObjectKnowledgeCertificate$^{\uparrow}$*, or* RegistryExportConsistencyGate *The retained boundary says that the comparison may land structurally through L and D, but cannot claim inherited doctrinal identity, prose affinity, host-semantic equality, or public object knowledge outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary cannot claim obligation`

Proof. Begin with the no-doctrinal-identity theorem, Definition 1268.4. It requires every accepted comparison to display both the BEDC landing row L and rejected-surplus row R.

Apply the classifier definition. The comparison is accepted only when the structural landing is read together with rejected surplus, so R remains the cannot-claim boundary rather than an optional annotation.

Finally route the packet through the listed sibling consumers. Axis-level refusal, object-knowledge licensing, and registry export consistency may consume the boundary only as S, L, R, D, H, C, P, N, so none can turn the rejected surplus into doctrinal identity, prose affinity, host equality, or public object knowledge. □

Theorem 1268.9 (Tradition comparison boundary scoped kernel route)**.** *Every accepted* TraditionComparisonBound *packet routes doctrinal refusal, sibling independence, source registry, export consistency, and cannot-claim obligations through the displayed rows S, L, R, D, H, C, P, N, the TasteGate marker on* BEDC.Derived.Traditio *and the finite BEDC rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The scoped kernel route exports no doctrinal identity, hidden tradition source, detached prose-affinity comparison, host comparison oracle, public object-knowledge authority, or registry export outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.TraditionComparisonBoundaryUp.TraditionComparisonBoundary scoped kernel route`

Proof. Start with the NameCert obligations of Definition 1268.3. They expose the finite comparison packet as exactly S, L, R, D, H, C, P, N and tie the route to the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Use doctrinal refusal and sibling independence next. The no-doctrinal-identity theorem, Definition 1268.4, requires the rejected-surplus row R, while Definition 1268.5 prevents the carrier from collapsing into the claim registry, cannot-claim registry, object-knowledge certificate, or registry export gate.

The source and export obligations are then supplied by Definition 1268.6 and Definition 1268.7. They keep the source row S, landing row L, rejected-surplus row R, and structural-distinction row D visible before any registry-facing read is exported.

Finally apply Definition 1268.8 and the displayed TasteGate marker named after Definition 1268.3. The cannot-claim boundary remains in the same finite packet, and the marker records only the single-carrier alignment. Hence the scoped kernel route exports none of the excluded doctrinal, hidden-source, object-knowledge, or host-comparison endpoints. □

Falsifiable prediction: If a comparison route exports doctrinal identity without a rejected-surplus row, it is not an accepted TraditionComparisonBoundaryUp packet.

Independence witness: claimregistrylayer, axiszeckendorfcannotclaim, objectknowledgecertificate, registryexportconsistencygate: none carries the tradition-source, BEDC landing, rejected-surplus, structural-distinction, and local NameCert rows in one packet.

The Lean follow-up must provide Nontrivial TraditionComparisonBoundary$^{\uparrow}$ and FieldFaithful TraditionComparison instances in `lean4/BEDC/Derived/TraditionComparisonBoundaryUp/TasteGate.lean`. Those instances must preserve S, L, R, D, H, C, P, N as separate displayed rows before BEDC.Derived.TraditionComparisonB may be cited by a checked paper marker.

Closure status for NameCert$_{\text{TraditionComparisonBoundary}^{\uparrow}}$.

Constructive story: A TraditionComparisonBoundary$^{\uparrow}$ packet is generated from tradition-source, BEDC landing, rejected-surplus, structural-distinction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{TraditionComparisonBoundary}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1268.1, Definition 1268.2, Definition 1268.3, Definition 1268.4, Definition 1268.5, Definition 1268.6, Definition 1268.7, Definition 1268.8, and Definition 1268.9 bind structural tradition comparison to a finite scoped NameCert kernel route.

Formal verification: FormalStatus(NameCert$_{\text{TraditionComparisonBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public export closure, or the correctness of any inherited tradition's doctrine.

Upgrade path: The next paper-axis step is a bridge-facing comparison export route; Nontrivial TraditionComparison-BoundaryUp and FieldFaithful TraditionComparisonBoundaryUp remain verification-side obligations in `lean4/BEDC/Derived/TraditionComp`

Chapter 1269

A Concrete Naming Certificate for $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$

The $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$ packet realizes the witness-preserving side of subsection 9462.25.6, subsection 9462.25.7, and subsection 9462.25.8. It sits beside the obstruction packet of chapter 899, the witnessed refutation surface of chapter 503, the proof witness chain of chapter 1238, the typed failure certificate of chapter 1237, and the certificate truth-branch boundary of chapter 854. Those siblings expose forbidden-schema refusal, permitted refutation, proof-chain witness order, typed failure rows, and truth-branch boundaries; this packet records the positive replacement ledger that makes an axiom-purity read consumable without meta closure.

Definition 1269.1 (Meta-closure witness ledger carrier)**.** *A* $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, K, D, A, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the witness row, carrying the selected finite object, recursion route, proof chain, or refutation witness before any consumer may use the positive content;*
2. *K is the classifier-transport row, recording how the relevant classifier is preserved instead of being collapsed into quotient equality;*
3. *D is the directional-transport row, recording the consumed direction for proposition-facing reads instead of identifying propositions by extensional equality;*
4. *A is the dependency-audit row, recording that the read avoids* $\mathsf{Classical.choice}$*, quotient soundness, and propositional extensionality;*
5. *R is the replacement row that names which axiom-purity request is being discharged by W, K, D, A;*
6. *H records componentwise* hsame *transport for W, K, D, A, R and the local name row;*
7. *C records the displayed* Cont *routes by which consumers read the witness, classifier-transport, directional-transport, dependency-audit, and replacement rows in order;*
8. *P is the* Pkg *provenance over* $\mathsf{MetaClosureObstruction}^{\uparrow}$, $\mathsf{WitnessedRefutation}^{\uparrow}$, $\mathsf{ProofWitnessChain}^{\uparrow}$, $\mathsf{TypedFailureCertificate}^{\uparrow}$, $\mathsf{CertificateTruthBranchBoundary}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, BHist, hsame, Cont, Pkg*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaClosureWitnessLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unwitnessed selector, quotient representative, proposition equality supplied by extensionality, host equality, ambient completeness principle, hidden dependency exemption, or detached proof source.

Lean4 ✓ `BEDC.Derived.MetaClosureWitnessLedgerUp.TasteGate.MetaClosureWitnessLedgerTasteGate single carrier alignment`

Theorem 1269.2 (Meta-closure witness ledger NameCert obligations)**.** *For the carrier of Definition 1269.1, the local* $\mathsf{NameCert}_{\mathsf{MetaClosureWitnessLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L = (W, K, D, A, R, H, C, P, N)$*, witness exposure through* W*, classifier-transport exposure through* K*, directional-transport exposure through* D*, dependency-audit exposure through* A*, replacement-read exposure through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MetaClosureWitnessLedgerUp.MetaClosureWitnessLedgerTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1269.1. The tuple displays the positive witness row, the two transport rows, the dependency-audit row, the replacement request, and the structural rows before a consumer can read the packet as an axiom-purity replacement.

The witness row W handles the term-facing side: selection is read only from displayed finite data, recursion data, or a proof chain. The classifier row K handles the type-facing side by preserving the classifier relation without creating quotient equality.

The directional row D handles the proposition-facing side. It records which implication or transport direction is consumed, so the packet never turns a pair of directions into equality of propositions.

The audit and structural rows close the local certificate surface. Row A records the dependency refusal, R identifies the replacement read, and H, C, P, N preserve, replay, source, and name the same finite rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1269.3 (Meta-closure witness ledger non-escape)**.** *Every accepted* $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$ *packet exports only the finite witness, classifier-transport, directional-transport, dependency-audit, replacement,* hsame, Cont, Pkg*, and local* NameCert *rows of Definition 1269.1. Any terminal axiom-purity replacement read of the packet factors through*

$$W, K, D, A, R, H, C, P, N.$$

It cannot be replaced by an unwitnessed selector, quotient stream, quotient representative, host equality, proposition equality supplied by extensionality, choice datum, ambient completeness row, or detached proof source.

Proof. Begin with the accepted carrier $L = (W, K, D, A, R, H, C, P, N)$. Its only positive object is the displayed witness row W, and the only permitted transport coordinates are the classifier row K and directional row D.

Apply Definition 1269.2. Every replacement read must pass through the audit row A and the replacement row R before it is named by N, so the read is dependency-audited before it becomes consumable certificate data.

The structural rows then enforce non-escape. Componentwise *hsame* can compare only the displayed rows, Cont replays only the displayed route, and Pkg names only the listed sibling sources.

Therefore any accepted replacement read is exhausted by $W, K, D, A, R, H, C, P, N$. The excluded objects require a selector, quotient, host equality, proposition collapse, choice row, ambient completeness row, or detached source coordinate that is absent from the carrier. □

Falsifiable prediction: Any accepted MetaClosureWitnessLedgerUp packet lacking one of displayed witness, classifier-transport, directional-transport, or dependency-audit rows cannot discharge an axiom-purity replacement read.

Independence witness: metaclosureobstruction, witnessedrefutation, proofwitnesschain, typedfailurecertificate, certificatetruthbranchboundary: MetaClosureObstructionUp records the forbidden-schema and truth-branch refusal side without the positive replacement ledger; WitnessedRefutationUp records a permitted refuter without classifier-transport, directional-transport, and dependency-audit rows; ProofWitnessChainUp records proof-chain order without the three-stratum replacement audit; TypedFailureCertificateUp records typed failure status without a positive axiom-purity replacement route; CertificateTruthBranchBoundaryUp records branch refusal without the witness ledger that discharges term, type, and proposition reads. Hence no row-preserving bijection exists with $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$.

The Lean follow-up namespace `BEDC.Derived.MetaClosureWitnessLedgerUp` must provide explicit `Nontrivial MetaClosureWitnessLedgerUp` and `FieldFaithful MetaClosureWitnessLedgerUp` instances for the displayed rows $W, K, D, A, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaClosureWitnessLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$ packet is generated from displayed witness, classifier-transport, directional-transport, dependency-audit, replacement, componentwise hsame, Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaClosureWitnessLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1269.1, Definition 1269.2, and Definition 1269.3 bind the positive axiom-purity replacement ledger to finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaClosureWitnessLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, ambient completeness, quotient construction, proposition extensionality, or a general host-level selector.

Upgrade path: Obligation closure requires checked Nontrivial and FieldFaithful TasteGate witnesses for $\mathsf{MetaClosureWitnessLedger}^{\uparrow}$ over the displayed witness, transport, audit, replacement, hsame, Cont, Pkg, and NameCert rows.

Chapter 1270

A Concrete Naming Certificate for $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$

The $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$ packet realizes the tradition and science registry rows of subsection 9462.34.5 and subsection 9462.34.6 as one finite NameCert carrier. It sits beside the auditable-generativity package of chapter 9462, the concept-registry surface of chapter 1252, the core registry index of chapter 1251, the registry layer of chapter 1220, and the claim registry layer of chapter 1209. Those siblings expose concept, index, registry, claim, or export rows; this packet binds tradition comparison and science failure-surface rows in one carrier before either family can be exported as package prose.

Definition 1270.1 (Tradition and science registry bridge carrier)**.** *A* $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, P, R, S, E, F, H, C, K, N)$$

with the following displayed rows:

1. *T is the tradition-entry row, recording tradition, traditional claim, BEDC site, preserved structure, rejected surplus, and status.*
2. *P is the preserved-structure row, recording the exact structural feature allowed to pass from the inherited tradition into the BEDC site.*
3. *R is the rejected-surplus row, recording the metaphysical, doctrinal, host-semantic, or prose-only reading that is not imported by the comparison.*
4. *S is the science-entry row, recording name, records carrier, invariant carrier, prediction gap, bridge status, claim status, and science-roadmap boundary.*
5. *E is the empirical-ledger row, recording the displayed records, measurement route, model-facing ledger, or explicit absence of empirical support for the same science entry.*
6. *F is the failure-surface row, recording what would block promotion of the science entry, including missing source rows, unledgered measurement, hidden choice, quotient collapse, propositional extensionality, missing invariant, or missing empirical ledger.*
7. *H stores componentwise* hsame *transport for tradition entry, preserved structure, rejected surplus, science entry, empirical ledger, failure surface, replay, provenance, and local naming rows.*
8. *C is the displayed* Cont *replay row through the tradition-comparison and science-failure routes.*

9. *K is the* Pkg *provenance over* ConceptRegistrySurface$^{\uparrow}$, CoreRegistryIndex$^{\uparrow}$, RegistryLayer$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, *philosophy tradition registry rows, philosophy science registry rows,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for inherited vocabulary without rejected surplus, scientific roadmap prose without a failure surface, empirical claims without a ledger row, or registry export outside the displayed packet $Q = (T, P, R, S, E, F, H, C, K, N)$.

Theorem 1270.2 (Tradition and science registry bridge obligations)**.** *For the carrier of Definition 1270.1, the local* NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $Q = (T, P, R, S, E, F, H, C, K, N)$; tradition-comparison exposure through T, P, R; science-roadmap exposure through S, E, F; componentwise transport through H; displayed replay through C; package provenance through K; and local naming through N.*

Lean4 ✓ `BEDC.Derived.TraditionScienceRegistryBridgeUp.TraditionScienceRegistryBridgeTasteGate`
`single carrier alignment`

Proof. Project the finite packet of Definition 1270.1. Rows T, P, R display the inherited tradition, the structure preserved by the comparison, and the surplus reading rejected by the BEDC site.

Rows S, E, F display the science entry, its empirical ledger, and the failure surface that can block promotion. These rows specialize the registry discipline of chapter 1220 and the claim-export discipline of chapter 1209 to the two registry families named in the philosophy tables.

Finally H, C, K, N transport, replay, package, and name the same packet. Since the carrier contains no hidden tradition doctrine, no science promise outside the failure surface, and no empirical row outside E, the listed obligations exhaust the local NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$ surface. □

Theorem 1270.3 (Tradition and science registry bridge nonescape)**.** *Every accepted* TraditionScienceRegistryBridge *packet refuses a tradition export with no rejected-surplus row and refuses a science export with no failure-surface row. A consumer cannot replace R or F by concept-registry placement, core-index membership, registry-layer admission, claim-layer export, transport, continuation replay, package provenance, or local naming alone.*

Proof. Use the obligations of Definition 1270.2. Tradition comparison must pass through T, P, R, so the preserved structure and rejected surplus are displayed in the same finite packet.

Science roadmap export must pass through S, E, F, so empirical support and the blocking surface are displayed before the roadmap is treated as a registry export.

The sibling packets named in the statement govern concept entries, index rows, registry admission, claim export, transport, continuation, package, or naming. They preserve or route an admitted packet; they do not synthesize the missing rejected-surplus row R or failure-surface row F. Thus any attempted export without those rows exits TraditionScienceRegistryBridge$^{\uparrow}$. □

Falsifiable prediction: A tradition or science registry export that lacks either a rejected-surplus row or a failure-surface row is outside TraditionScienceRegistryBridgeUp.

Independence witness: conceptregistrysurface, coreregistryindex, registrylayer, claimregistrylayer, registryexportsurface: no sibling is bijective because each controls concept, index, registry, claim, or export rows separately, while this packet binds tradition comparison and science failure-surface rows in one carrier.

The Lean follow-up requires Nontrivial TraditionScienceRegistryBridge$^{\uparrow}$ and FieldFaithful TraditionScienceRegistryB instances in `lean4/BEDC/Derived/TraditionScienceRegistryBridgeUp/TasteGate.lean`. Those instances must preserve $T, P, R, S, E, F, H, C, K, N$ as separate displayed rows before any checked theorem may cite the packet as a TasteGate surface.

Closure status for NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$.

Constructive story: A TraditionScienceRegistryBridge$^{\uparrow}$ packet is generated from tradition-entry, preserved-structure, rejected-surplus, science-entry, empirical-ledger, failure-surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$, seed)

Scope closed: Definition 1270.1, Definition 1270.2, and Definition 1270.3 bind tradition comparison and science roadmap export to one finite NameCert packet.

Formal verification: FormalStatus(NameCert$_{\text{TraditionScienceRegistryBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, empirical correctness of science entries, or philosophical correctness of every tradition comparison.

Upgrade path: Lean closure requires Nontrivial TraditionScienceRegistryBridgeUp and FieldFaithful TraditionScienceRegistryBridgeUp instances in `lean4/BEDC/Derived/TraditionScienceRegistryBridgeUp/TasteGate.lean`.

Chapter 1271

A Concrete Naming Certificate for $\mathsf{SubjectFreeObservation}^{\uparrow}$

The $\mathsf{SubjectFreeObservation}^{\uparrow}$packet realizes the subject-free observation discipline of section 9462.22 as a finite NameCert surface. It consumes the observer-history identity packet of chapter 477, the observer-filter split of chapter 514, the kernel observation sieve of chapter 932, and the local clock budget of chapter 665. The packet records an observation as displayed Cont replay, treats observer identity as hsame over history rows, and exposes the absence of any global subject parameter as a ledgered exclusion.

Definition 1271.1 (Subject-free observation carrier)**.** *A* $\mathsf{SubjectFreeObservation}^{\uparrow}$*carrier is a finite* BHist *packet*

$$S = (O, H, K, D, L, C, P, N)$$

with the following displayed rows:

1. *O is the displayed* Cont *observation row, recording the observation event as a continuation over the current history surface;*
2. *H is the* hsame *observer-history identity row certified by chapter 477;*
3. *K is the subject-free signature row, recording that the observation is typed by its* BHist*,* Cont*, and* hsame *rows rather than by an auxiliary subject carrier;*
4. *D is the no-subject exclusion row, refusing a subject-substance coordinate, owner map, global observer parameter, or private viewpoint object outside the displayed history rows;*
5. *L is the public ledger-event row, exposing the accepted observation event to downstream consumers as a named finite record;*
6. *C records* Cont *replay for observation readback, signature inspection, no-subject checking, and ledger-event exposure;*
7. *P is the* Pkg *provenance over* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$*,* $\mathsf{ObserverFilter}^{\uparrow}$*,* $\mathsf{KernelObservationSieve}^{\uparrow}$*,* $\mathsf{LocalClockBudget}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectFreeObservation}^{\uparrow}}$ *row.*

The classifier compares carriers only through O, H, K, D, L, C, P, N. It has no coordinate for a subject-substance, host observer identity, owner function, global viewpoint, or observation source outside the displayed BHist*,* Cont*, and* hsame *rows.*

Lean4 (definition) BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationCarrier

Theorem 1271.2 (Subject-free observation NameCert obligations)**.** *For the carrier of Definition 1271.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectFreeObservation}^{\uparrow}}$ *surface consists exactly of carrier admission for* $S = (O, H, K, D, L, C, P, N)$*, displayed* Cont *observation through* O*, observer-history identity through* H*, subject-free signature exposure through* K*, no-subject exclusion through* D*, public ledger-event exposure through* L*, replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationTasteGate single carrier alignment

Proof. Project the carrier. The tuple lists the observation row, observer-history identity row, subject-free signature row, no-subject exclusion row, public ledger-event row, replay row, provenance row, and local naming row.

Read the observation and identity rows first. The observation is the displayed Cont row O, while observer identity is supplied by the hsame row H from the observer-history identity surface.

The rows K and D then separate admissible signature data from excluded subject data. The signature is built from the displayed history, continuation, and sameness rows, and the exclusion row records that no subject-substance coordinate is available.

Finally L, C, P, N expose, replay, source, and name the same packet. Since the carrier contains no hidden subject coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 1271.3 (Subject-free observation no-subject parameter)**.** *If a purported observation packet requires a subject-substance coordinate outside the displayed* BHist*,* Cont*, and* hsame *rows, then it is not an accepted* $\mathsf{SubjectFreeObservation}^{\uparrow}$ *carrier.*

Lean4 ✓ BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationCarrier no subject parameter

Proof. Inspect Definition 1271.1. The only observation source is O, the only observer-identity row is H, and the only signature row is K.

The no-subject row D records the forbidden coordinates before any public ledger event is exposed. A subject-substance coordinate outside the displayed rows would have to be read either from K or from L.

Both routes are blocked by the carrier inventory. The signature row records only the subject-free observation type, and the ledger row exposes only the accepted finite event, so a packet requiring an auxiliary subject parameter falls outside the classifier. □

Theorem 1271.4 (Subject-free observation ledger event)**.** *Every public event exported by an accepted* $\mathsf{SubjectFreeObservation}^{\uparrow}$ *packet factors through*

$$O, H, K, D, L, C, P, N.$$

The event is a displayed Cont *observation with* hsame *observer-history identity and a no-subject ledger, not a host-side subject report.*

Lean4 ✓ BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationCarrier ledger event

Proof. Start from the accepted packet $S = (O, H, K, D, L, C, P, N)$. The public event row is L, and it is read only after the observation row O, observer-history row H, signature row K, and exclusion row D have been displayed.

Replay through C preserves this order for consumers. Provenance through P names the observer-history, observer-filter, kernel-sieve, and local-clock-budget source rows, while N names the local certificate.

No additional report channel appears in the packet. Thus the exported event is exactly the finite ledgered observation route and cannot be interpreted as a host-side subject report. □

Theorem 1271.5 (Subject-free observation sibling independence)**.** *The* SubjectFreeObservation$^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* ObserverHistoryIdentity$^{\uparrow}$*,* ObserverFilter$^{\uparrow}$*,* KernelObservationSieve$^{\uparrow}$*,* LocalClockBudget$^{\uparrow}$*, or* ConsciousObserverState$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationCarrier sibling independence`

Proof. Compare the sibling surfaces. ObserverHistoryIdentity$^{\uparrow}$ supplies hsame observer-history identity, ObserverFilter$^{\uparrow}$ separates selected and omitted rows, KernelObservationSieve$^{\uparrow}$ filters kernel-facing observations, LocalClockBudget$^{\uparrow}$ budgets local observation windows, and ConsciousObserverState$^{\uparrow}$ records current observer-state rows.

None of those siblings carries the full tuple O, H, K, D, L, C, P, N with both the displayed Cont observation row and the no-subject exclusion ledger. A NameCert-row-preserving bijection would have to preserve those two rows together with the public ledger-event row.

Because the siblings each supply only a proper dependency surface, SubjectFreeObservation$^{\uparrow}$ remains a distinct finite carrier for subject-free observation packets. □

Falsifiable prediction: If a packet requires a subject-substance coordinate outside the displayed BHist, Cont, and hsame rows, it cannot satisfy the SubjectFreeObservationUp NameCert obligations.

Independence witness: observerhistoryidentity, observerfilter, kernelobservationsieve, localclockbudget, consciousobserverstate: none carries the displayed Cont observation row, hsame observer-history identity row, subject-free signature row, no-subject exclusion ledger, public event row, replay row, provenance row, and local NameCert row in one packet.

The Lean follow-up in `lean4/BEDC/Derived/SubjectFreeObservationUp/TasteGate.lean` must provide Nontrivial SubjectFreeObservation$^{\uparrow}$ and FieldFaithful SubjectFreeObservation$^{\uparrow}$ instances before `BEDC.Derived.SubjectFreeObservationUp.SubjectFreeObservationTasteGate_single_carrier_alig` can be cited by a checked paper marker.

Closure status for NameCert$_{\text{SubjectFreeObservation}^{\uparrow}}$ **·**

Constructive story: The subject-free observation certificate is generated from a $BHist$ event row, a displayed Cont observation row, an $hsame$ observer-history identity row, a subject-free signature row, and a no-subject exclusion ledger. The carrier first reads the observation event, then records that no subject slot is part of the route, and finally uses Cont only to extend the same event-ledger path. The $hsame$ relation preserves subject-freeness under changes of representative history: an equivalent history may carry the same observation, but it cannot introduce a hidden subject row. The NameCert obligations, no-subject parameter theorem, ledger-event theorem, and sibling-independence result together make observation a visible event packet.

Theory closure: TheoryGate(NameCert$_{\text{SubjectFreeObservation}^{\uparrow}}$, seed)

Scope closed: This chapter binds the subject-free observation carrier, NameCert obligations, no-subject parameter theorem, ledger-event theorem, sibling-independence theorem, prediction, and seed closure marker.

Formal verification: FormalStatus(NameCert$_{\text{SubjectFreeObservation}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim observer identity, subject experience, subject-substance coordinates, or support-lattice membership outside the subject-free event ledger.

Upgrade path: Reaching obligationClosure requires a formal observation-event carrier and checked absence of subject parameters under $hsame$ transport.

Chapter 1272

A Concrete Naming Certificate for $\mathsf{SubjectIndexRoute}^{\uparrow}$

The $\mathsf{SubjectIndexRoute}^{\uparrow}$ packet realizes the subject-index route of Definition 9462.2242 as a finite NameCert carrier. It sits beside the core registry index of chapter 1251, the concept registry surface of chapter 1252, the registry export surface of chapter 1216, the registry export consistency gate of chapter 1224, and the tradition/science bridge of chapter 1270. Those siblings expose registry tables, concept entries, export rows, consistency gates, and tradition/science rows; this packet stores the route by which ordinary subject vocabulary reaches that lattice.

Definition 1272.1 (SubjectIndexRoute carrier)**.** *A* $\mathsf{SubjectIndexRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, C, R, E, G, T, H, K, P, N)$$

with the following displayed rows:

1. *O is the ordinary subject-entry row, recording the public subject vocabulary, local reader entry point, and displayed registry destination.*
2. *C is the core-registry route row, recording the* $\mathsf{CoreRegistryIndex}^{\uparrow}$ *table position reached by the ordinary entry.*
3. *R is the concept-registry route row, recording the* $\mathsf{ConceptRegistrySurface}^{\uparrow}$ *concept entry when the subject term names a concept-facing route.*
4. *E is the registry-export route row, recording the* $\mathsf{RegistryExportSurface}^{\uparrow}$ *export surface through which a public subject entry may leave the registry lattice.*
5. *G is the consistency-gate route row, recording the* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *check that keeps registry entry, export grade, closure status, and formal target aligned.*
6. *T is the tradition/science bridge row, recording the* $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$ *route when the subject term points into inherited comparison or science-roadmap vocabulary.*
7. *H stores componentwise* hsame *transport for ordinary subject entries, core registry rows, concept rows, export rows, consistency-gate rows, tradition/science rows, replay, provenance, and local naming.*
8. *K stores displayed* Cont *replay through ordinary subject lookup, core registry indexing, concept registry lookup, registry export, consistency-gate inspection, and tradition/science bridge handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{SubjectIndexRoute}^{\uparrow}$, $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubjectIndexRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for a public subject entry without a displayed registry row, for a hidden ordinary-language index, for a route outside the registry lattice, or for prose synthesis that bypasses $S = (O, C, R, E, G, T, H, K, P, N)$.

Theorem 1272.2 (SubjectIndexRoute NameCert obligations)**.** *For the carrier of Definition 1272.1, the local* $\mathsf{NameCert}_{\mathsf{SubjectIndexRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (O, C, R, E, G, T, H, K, P, N$ ordinary subject-entry exposure through O; core registry routing through C; concept registry routing through R; registry export routing through E; consistency-gate routing through G; tradition/science bridge routing through T; componentwise transport through H; displayed continuation replay through K; package provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.SubjectIndexRouteUp.SubjectIndexRouteUp single carrier alignment`

Proof. Project the finite packet of Definition 1272.1. The ordinary subject row O gives the reader-facing entry point, while C, R, E, G, T display the registry lattice positions reached by that entry.

The sibling chapters supply the table, concept, export, consistency, and tradition/science surfaces. $\mathsf{SubjectIndexRoute}^{\uparrow}$ does not copy any one sibling alone; it binds the route from ordinary vocabulary into the same finite registry lattice.

Finally H, K, P, N transport, replay, package, and name the same packet. Since the carrier contains no hidden subject index or prose-only route, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{SubjectIndexRoute}^{\uparrow}}$ surface. □

Theorem 1272.3 (SubjectIndexRoute registry sibling handoff)**.** *Every accepted* $\mathsf{SubjectIndexRoute}^{\uparrow}$ *packet hands an ordinary subject entry to at least one displayed sibling row among chapter 1251, chapter 1252, chapter 1216, chapter 1224, and chapter 1270. A consumer cannot replace that handoff by ordinary prose, host semantics, hidden registry state, or local naming alone.*

Proof. Use the obligations of Definition 1272.2. An accepted packet must display the ordinary subject-entry row O together with the sibling route rows C, R, E, G, T and the support rows H, K, P, N.

If none of C, R, E, G, T is displayed, then O has no registry destination and the packet is not an accepted subject-index route. If a consumer keeps only local naming, the same destination data is still absent.

Thus each accepted subject entry reaches the registry lattice through a displayed sibling row. The route may pass through the core index, concept surface, export surface, consistency gate, or tradition/science bridge, but it cannot bypass the displayed handoff. □

Falsifiable prediction: Any accepted SubjectIndexRouteUp export that reaches a public subject entry without a displayed registry row refutes the packet.

Independence witness: coreregistryindex, conceptregistrysurface, registryexportsurface, registryexportconsistencygate, traditionscienceregistrybridge: SubjectIndexRouteUp is not bijective to any of these because it stores the route from subject vocabulary into the registry lattice rather than a registry table, export surface, consistency gate, or bridge alone.

The Lean follow-up must provide Nontrivial SubjectIndexRoute$^{\uparrow}$ and FieldFaithful SubjectIndexRoute$^{\uparrow}$ instances in `lean4/BEDC/Derived/SubjectIndexRouteUp/TasteGate.lean`. The instances must preserve $O, C, R, E, G, T, H, K, P, N$ as separate displayed fields before any checked theorem may cite the packet as a TasteGate surface.

Closure status for NameCert$_{\text{SubjectIndexRoute}^{\uparrow}}$.

Constructive story: A SubjectIndexRoute$^{\uparrow}$ packet is generated from ordinary subject-entry, core registry, concept registry, registry export, consistency-gate, tradition/science bridge, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{SubjectIndexRoute}^{\uparrow}}$, seed)

Scope closed: Definition 1272.1, Definition 1272.2, and Definition 1272.3 bind ordinary subject entries to displayed registry-lattice routes.

Formal verification: FormalStatus(NameCert$_{\text{SubjectIndexRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completeness of the subject index, or correctness of every downstream registry entry.

Upgrade path: Lean closure requires Nontrivial SubjectIndexRouteUp and FieldFaithful SubjectIndexRouteUp instances in `lean4/BEDC/Derived/SubjectIndexRouteUp/TasteGate.lean`.

Chapter 1273

A Concrete Naming Certificate for $\mathsf{ScientificModelBridge}^{\uparrow}$

The $\mathsf{ScientificModelBridge}^{\uparrow}$ packet realizes the science-bridge site of section 9462.28 as finite BEDC certificate data. It sits after the scientific-object carrier of chapter 1214, the physical-model audit of chapter 1129, the standard-bridge audit packet of chapter 1217, the reality-constrained model audit of chapter 1124, and the open-fit packet of chapter 1137. Those siblings expose object records, physical audit, bridge audit, model-facing audit, and open empirical fit; this packet binds them into a bridge carrier whose public reads cannot replace internal certificates with an external standard model.

Definition 1273.1 (Scientific model bridge carrier)**.** *A* $\mathsf{ScientificModelBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, B, R, F, L, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{ScientificObject}^{\uparrow}$ *row exposing the scientific-object record, classifier, empirical ledger, bridge row, gap report, audit support, open-fit support, and local naming surface.*
2. *A is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row exposing source traces, target phenomenon, observed and model signatures, audit classifier, tower data, failure surface, and audit strength.*
3. *B is the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *row exposing encoding, decoding, round-trip, theorem preservation, no-host-leak, transport, provenance, and local naming data for a proposed standard bridge.*
4. *R is the* $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ *row that records model-facing acceptance, rejection, or open status against finite BEDC evidence.*
5. *F is the* $\mathsf{OpenFitPacket}^{\uparrow}$ *row exposing finite fit, finite defeat, continuation scope, and no-final-oracle boundary.*
6. *L is the empirical ledger row recording accepted observations, failed predictions, open prediction gaps, and the declared falsification surface for the bridge.*
7. *T is the transport row that records how the object, audit, bridge-audit, model-audit, open-fit, and ledger rows are replayed into a consumer-facing prediction route.*
8. *H records componentwise* hsame *transport for $O, A, B, R, F, L, T, C, P, N$.*

9. *C records the displayed* Cont *routes by which the scientific-object, audit, bridge, model-audit, open-fit, ledger, transport, and prediction rows are replayed.*
10. *P is the* Pkg *provenance over* $\mathsf{ScientificModelBridge}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{OpenFitPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ScientificModelBridge}^{\uparrow}}$ *row.*

The classifier compares carriers only through these rows. It has no coordinate for external model authority, host-theory equality, unledgered empirical adequacy, unbounded prediction streams, hidden quotient classes, or a bridge accepted without the local audit rows.

Lean4 ✓ `BEDC.Derived.ScientificModelBridgeUp.TasteGate.ScientificModelBridgeTasteGate single carrier alignment`

Theorem 1273.2 (Scientific model bridge NameCert obligations)**.** *For the carrier of Definition 1273.1, the local* $\mathsf{NameCert}_{\mathsf{ScientificModelBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $M = (O, A, B, R, F, L, T, H, C, P, N)$, scientific-object exposure through O, physical-model audit exposure through A, standard-bridge audit exposure through B, reality-constrained model-audit exposure through R, open-fit exposure through F, empirical-ledger and falsification-surface exposure through L, prediction-transport exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ScientificModelBridgeUp.ScientificModelBridgeCarrier namecert obligations`

Proof. Project the accepted packet to its displayed rows. The scientific-object row O supplies the object-facing record, gap, audit, and bridge support; the rows A, B, R, F supply the physical audit, standard-bridge audit, reality-constrained model audit, and open-fit surfaces.

Read the ledger and prediction route next. The row L records empirical support, failed predictions, open gaps, and falsification exposure, while T records the consumer-facing prediction transport. These rows make the bridge a finite BEDC carrier rather than a detached model identifier.

Finally apply componentwise transport, continuation replay, provenance, and local naming. The rows H, C, P, N preserve exactly the displayed object, audit, bridge, model-audit, open-fit, ledger, and prediction surfaces. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1273.3 (Scientific model bridge falsifiable prediction)**.** *Every accepted* $\mathsf{ScientificModelBridge}^{\uparrow}$ *packet that exports a prediction route must expose the empirical ledger row L and prediction transport row T together with the object, physical-audit, standard-bridge-audit, model-audit, and open-fit rows O, A, B, R, F. If a downstream consumer reads the bridge while deleting L, deleting T, bypassing the open-fit row F, or replacing B by external standard-model authority, then the consumer has left the accepted packet. A defeating observation is admissible exactly as a finite update to the displayed ledger and open-fit surface, not as a host verdict outside the carrier.*

Lean4 ✓ `BEDC.Derived.ScientificModelBridgeUp.ScientificModelBridgeCarrier falsifiable prediction`

Proof. Begin with the NameCert obligation surface of Definition 1273.2. Any accepted prediction route must carry $O, A, B, R, F, L, T, H, C, P, N$ before a consumer can read the local name.

The scientific-object row keeps the object record and gap report visible, the audit rows keep model-facing evidence visible, the standard-bridge audit row keeps bridge discipline internal, and

the open-fit row keeps finite defeat and continuation scope visible. The ledger row L and transport row T then state where a prediction is read and where a finite defeating observation may enter.

Thus a consumer that removes L or T no longer has a prediction surface, and a consumer that removes F no longer has the open empirical boundary. A consumer that replaces B by an external standard model bypasses the bridge-audit row and therefore bypasses the carrier. The only accepted falsification route is the displayed finite update through L and F under the same H, C, P, N support. □

Theorem 1273.4 (Scientific model bridge scoped kernel route)**.** *The scoped bridge route for* ScientificModelBridge$^{\uparrow}$ *is exactly the finite readback surface generated by the* ScientificObject$^{\uparrow}$, PhysicalModelAudit$^{\uparrow}$, StandardBridgeAuditPacket$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of an accepted carrier* $M = (O, A, B, R, F, L, T, H, C, P, N)$*. A consumer may inspect the empirical ledger and prediction transport only through those rows, and no external standard-model authority, unledgered empirical adequacy, host-theory equality, quotient-model equality, or unbounded prediction stream lies inside the scoped route.*

Proof. Use the carrier of Definition 1273.1. Its displayed rows are precisely the scientific-object, physical-audit, bridge-audit, model-audit, open-fit, ledger, prediction-transport, componentwise transport, replay, provenance, and local naming coordinates.

The obligation theorem Definition 1273.2 makes those coordinates the local NameCert surface. The falsifiable-prediction theorem Definition 1273.3 then forces every prediction read to keep the ledger L, transport row T, and open-fit row F visible together with the object and audit rows.

The excluded alternatives have no carrier coordinate. Therefore the scoped kernel route is exactly the listed finite surface over ScientificObject$^{\uparrow}$, PhysicalModelAudit$^{\uparrow}$, StandardBridgeAuditPacket$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert. □

Lean4 ✓ `BEDC.Derived.ScientificModelBridgeUp.ScientificModelBridgeCarrier scoped kernel route`

Falsifiable prediction: Deleting the empirical ledger, prediction transport, open-fit surface, or standard-bridge audit row from an accepted ScientificModelBridgeUp packet breaks the bridge classifier.

Independence witness: scientificobject, physicalmodelaudit, standardbridgeauditpacket, realityconstrainedmodelaudit, openfitpacket: none is bijective because ScientificModelBridge$^{\uparrow}$ binds object records, physical audit, bridge audit, model-facing audit, open fit, empirical ledger, prediction transport, componentwise hsame transport, Cont replay, Pkg provenance, and one local NameCert row in a single bridge packet.

Closure status for NameCert$_{\text{ScientificModelBridge}^{\uparrow}}$.

Constructive story: A ScientificModelBridge$^{\uparrow}$ packet is generated from scientific-object, physical-model-audit, standard-bridge-audit, reality-constrained-model-audit, open-fit, empirical-ledger, prediction-transport, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{ScientificModelBridge}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1273.1, Definition 1273.2, and Definition 1273.3 bind the ScientificObject, PhysicalModelAudit, StandardBridgeAuditPacket, RealityConstrainedModelAudit, and OpenFitPacket sibling rows to the displayed bridge packet $O, A, B, R, F, L, T, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\text{ScientificModelBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove external standard-model correctness, physical truth, bridge checking, complete empirical adequacy, unbounded prediction coverage, quotient-model equivalence, or a Lean implementation.

Upgrade path: Scoped closure requires grounding the displayed object, audit, bridge-audit, model-audit, open-fit, ledger, prediction-transport, replay, provenance, and local NameCert rows against the named sibling chapters.

Chapter 1274

A Concrete Naming Certificate for $\mathsf{UpgradeRegistry}^{\uparrow}$

The $\mathsf{UpgradeRegistry}^{\uparrow}$ packet realizes the upgrade-registry table of subsection 9462.34.8 as a finite NameCert surface for promotion claims. It sits beside the claim-entry discipline of chapter 1209, the public export gate of chapter 1216, and the refusal axis of chapter 554. Those siblings admit a claim row, expose registry export, and block overstrong readings; this packet binds the current-status, next-status, blocker, formal-target, provenance, transport, and local naming rows before any promotion may be read as accepted.

Definition 1274.1 (UpgradeRegistry carrier)**.** *An* $\mathsf{UpgradeRegistry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, S, Nx, F, B, A, R, H, C, P, L)$$

with the following displayed rows:

1. *T is the target row naming the chapter, package, theorem site, registry family, or bridge surface whose status may be promoted;*
2. *S is the current-status row, recording the theory grade, verification grade, and bridge-status grade presently exported by the target;*
3. *Nx is the next-status row, naming the immediate stronger status that may be accepted only after the required rows are discharged;*
4. *F is the required-field row, listing the carrier, classifier, stability, exactness, ledger, bridge, or proof fields that must become public packet data;*
5. *B is the blocker row, recording the open gap, refused reading, missing proof, missing bridge, or unresolved dependency that prevents immediate completion export;*
6. *A is the audit row, recording the registry-consistency, writing-discipline, and cannot-claim checks consumed before the promotion is exported;*
7. *R is the formal-target row, naming the Lean declaration, statement-only target, bridge obligation, or absence of such a target as public registry data;*
8. *H records componentwise* hsame *transports for the target, current-status, next-status, field, blocker, audit, formal-target, continuation, package, and local naming rows;*
9. *C records displayed* Cont *routes by which claim consumers, export consumers, and refusal consumers read the same promotion packet;*

10. *P is the* Pkg *provenance row over* ClaimRegistryLayer$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, AxisZeckendorfCannotClaim$^{\uparrow}$, AuditWritingDiscipline$^{\uparrow}$, RegistryConsistencyAudit$^{\uparrow}$, BHist, hsame, Cont, *and local certificate data;*
11. *L is the local* NameCert$_{\text{UpgradeRegistry}^{\uparrow}}$ *row.*

The classifier for two accepted upgrade-registry packets compares only the displayed target, current-status, next-status, required-field, blocker, audit, formal-target, transport, continuation, package, and local naming rows.

Lean4 ✓ `BEDC.Derived.UpgradeRegistryUp.TasteGate.UpgradeRegistryTasteGate single carrier alignment`

Theorem 1274.2 (UpgradeRegistry NameCert obligations)**.** *For the carrier of Definition 1274.1, the local* NameCert$_{\text{UpgradeRegistry}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier exposure: every accepted registry promotion packet exposes exactly $U = (T, S, Nx, F, B, A, R, H, C, P, L)$;*
2. *status locality: public comparison reads the current-status row S and next-status row Nx before any promotion claim is exported;*
3. *blocker accountability: every accepted blocker row B remains visible to the claim registry and cannot-claim registry until the required-field row F is discharged;*
4. *formal-target exposure: the formal-target row R is public packet data and cannot be replaced by prose expectation or hidden tooling state;*
5. *audit preservation: registry-consistency and writing-discipline checks are carried through the audit row A and the displayed* Cont *routes in C;*
6. *naming locality: componentwise* hsame, Pkg, *and* NameCert *provenance preserve the same promotion packet for all public consumers.*

Lean4 ✓ `BEDC.Derived.UpgradeRegistryUp.UpgradeRegistryNameCertObligations`

Proof. Carrier exposure is projection from Definition 1274.1. The tuple displays the target, status rows, required fields, blockers, audit route, formal target, transports, continuations, package provenance, and local naming row.

Status locality and blocker accountability follow from the classifier. Since $S, Nx, F, B, A, R, H, C, P, L$ are the only compared rows, a consumer cannot read a promotion without also reading the current status and the blocker surface that prevents completion export.

Formal-target exposure and audit preservation are the R, A, C coordinates. The claim layer admits the row only through the public registry packet, while the export surface and cannot-claim sibling read the same displayed continuation routes. Componentwise transport and package provenance then repack the rows as the local NameCert packet. □

Theorem 1274.3 (UpgradeRegistry status preservation)**.** *If an accepted* UpgradeRegistry$^{\uparrow}$ *packet has current-status row S, next-status row Nx, required-field row F, blocker row B, audit row A, and formal-target row R, then every public export route preserves the distinction between current status and next status. A promotion claim may be exported only through the displayed route $S, Nx, F, B, A, R, H, C, P, L$; it cannot be exported from the next-status row alone.*

Lean4 ✓ `BEDC.Derived.UpgradeRegistryUp.UpgradeRegistryStatusPreservation`

Proof. Apply Definition 1274.2. The status rows are adjacent coordinates in the finite carrier, and the NameCert obligations force both to remain visible to the public comparison.

The blocker and required-field rows are also displayed. Thus a route that attempts to read Nx as already accepted must also expose F and B, where the unmet field or blocker remains part of the same packet.

Finally, the export route factors through A, R, H, C, P, L. These rows preserve audit, formal-target, transport, continuation, provenance, and local naming data, so no consumer route can erase the current-status row while retaining the promotion claim. □

Theorem 1274.4 (UpgradeRegistry noncompletion surface)**.** *An accepted* $\mathsf{UpgradeRegistry}^{\uparrow}$ *packet with no current-status row or no blocker row cannot export a completion claim. The packet may still expose a target row, next-status row, required-field row, audit row, or formal-target row, but those rows do not constitute a completed promotion without the displayed current-status and blocker accounting.*

Lean4 ✓ `BEDC.Derived.UpgradeRegistryUp.UpgradeRegistryNoncompletionSurface`

Proof. Use the carrier definition. A valid packet exposes S and B as coordinates, so a route without either coordinate is not the accepted carrier of Definition 1274.1.

Use status preservation from Definition 1274.3. Completion export requires a route that preserves the distinction between current status and next status, and that route also carries the blocker surface. Without S or B, the route may at most display a target, desired status, field list, audit row, or formal target; it cannot display the complete promotion packet. □

Theorem 1274.5 (UpgradeRegistry public export)**.** *Any public* $\mathsf{UpgradeRegistry}^{\uparrow}$ *export must expose the target row, current-status row, next-status row, required-field row, blocker row, audit row, formal-target row, componentwise transport row, continuation row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}$ *row. Export through* $\mathsf{RegistryExportSurface}^{\uparrow}$ *and* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *cannot erase blocker accountability or read the next-status row as a completed promotion.*

Lean4 ✓ `BEDC.Derived.UpgradeRegistryUp.UpgradeRegistryPublicExport`

Proof. Apply Definition 1274.2. The public packet is the displayed carrier $U = (T, S, Nx, F, B, A, R, H, C, P, L)$, so every export route reads the target, status, field, blocker, audit, formal target, transport, continuation, provenance, and local naming rows.

Apply Definition 1274.3. The current-status row S and the next-status row Nx remain distinct along every public route, and the required-field and blocker rows travel with that distinction.

Finally apply Definition 1274.4. A route through $\mathsf{RegistryExportSurface}^{\uparrow}$ or $\mathsf{ClaimRegistryLayer}^{\uparrow}$ that drops S or B may still display registry metadata, but it cannot export completion. Thus blocker accountability remains public whenever the upgrade-registry packet is exported. □

Falsifiable prediction: An accepted $\mathsf{UpgradeRegistry}^{\uparrow}$ packet with no current-status row or blocker row cannot export a completion claim.

Independence witness: claimregistrylayer, registryexportsurface, auditwritingdiscipline, axiszeckendorfcannotclaim, registryconsistencyaudit: $\mathsf{UpgradeRegistry}^{\uparrow}$ is not bijective to any sibling because it simultaneously carries target, current-status, next-status, required-field, blocker, audit, formal-target, transport, continuation, provenance, and local NameCert rows. The claim layer admits claim rows without the full promotion packet; the export surface controls release without blocker accounting; the writing-discipline packet controls prose legality without next-status fields; the cannot-claim packet records refusals without promotion transport; and the consistency audit checks agreement without owning the registry entry.

The Lean follow-up for this chapter must provide $\mathsf{Nontrivial\ UpgradeRegistry}^{\uparrow}$ and $\mathsf{FieldFaithful\ UpgradeRegistry}^{\uparrow}$ instances for the displayed carrier fields.

Theorem 1274.6 (UpgradeRegistry bridged handoff surface)**.** *Every bridge-facing read of an accepted* $\mathsf{UpgradeRegistry}^{\uparrow}$ *packet routes through the target row, current-status row, next-status row, required-field row, blocker row, audit row, formal-target row, componentwise transport row, continuation row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}$ *row before any promotion surface is exported. The handoff therefore exposes the same finite route* $T, S, Nx, F, B, A, R, H, C, P, L$ *to bridge consumers as to public registry consumers, and it cannot replace blocker accountability, status preservation, audit exposure, or formal-target exposure by hidden bridge state.*

Proof. Read the public export theorem Definition 1274.5. It already forces every public promotion route to expose $T, S, Nx, F, B, A, R, H, C, P, L$ before completion can be claimed.

A bridge-facing consumer is still a consumer of the same accepted registry packet. The carrier of Definition 1274.1 has no separate bridge-only coordinate, so the bridge read cannot bypass the target, status, required-field, blocker, audit, formal-target, transport, continuation, provenance, or local naming rows.

Status preservation and the noncompletion surface then keep the blocker and current-status rows visible. Thus the bridge-facing handoff is a route through the displayed packet rather than an external promotion assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}$**.**

Constructive story: An $\mathsf{UpgradeRegistry}^{\uparrow}$ packet is generated from target, current-status, next-status, required-field, blocker, audit, formal-target, transport, continuation, package, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1274.1, Definition 1274.2, Definition 1274.3, Definition 1274.4, Definition 1274.5, and Definition 1274.6 scope the public and bridge-facing export surfaces for the target, current-status, next-status, required-field, blocker, audit, formal-target, hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}$ rows owned by the upgrade-registry certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpgradeRegistry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, or completion of registry promotions beyond the displayed public and bridge-facing export surfaces.

Upgrade path: No theory-axis upgrade remains; the remaining work is Lean verification and bridge checking for the displayed registry rows.

Chapter 1275

A Concrete Naming Certificate for $\mathsf{RelationalPhysics}^{\uparrow}$

The $\mathsf{RelationalPhysics}^{\uparrow}$ packet realizes the relational-physics site of subsection 9462.29.10 as finite BEDC certificate data. The recognized tradition supplies the external reading that physical content is relational rather than attached to a privileged global frame; the packet below keeps the BEDC realization internal to observer-history identity, inter-history locality, inter-history invariant, physical-model audit, admissible rate or refusal rows, displayed transport, replay, provenance, and local naming.

Its sibling context is the observer-history identity surface of chapter 479, the invariant readback of chapter 656, the locality ledger of chapter 759, and the physical-model audit of chapter 1129. The philosophical horizon is the relational-physics synthesis site of subsection 9462.29.10.

Definition 1275.1 (Relational physics carrier)**.** *A* $\mathsf{RelationalPhysics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (O, I, L, V, A, H, C, P, N)$$

with the following displayed rows:

1. *O is the observer-history identity row, read through* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *and carrying only history, observation,* hsame*,* Cont*,* Pkg*, and local naming data.*
2. *I is the inter-history invariant row supplied by* $\mathsf{InterHistInvariant}^{\uparrow}$ *after the relevant two-history packet and relation row have been displayed.*
3. *L is the inter-history locality ledger row supplied by* $\mathsf{InterHistLocalityLedger}^{\uparrow}$*, binding the two source histories, locality cells, invariant readback, and no-global-frame exclusion.*
4. *V is the visible physical-model audit row supplied by* $\mathsf{PhysicalModelAudit}^{\uparrow}$*, recording source traces, observed and model signatures, classifier, tower, failure surface, and audit strength as finite packet data.*
5. *A is the admissible rate or refusal row: it either carries a maximum-rate certificate accepted by the relevant inter-history route, or a displayed refusal that no such rate row is licensed for the packet.*
6. *H records componentwise* hsame *transport for O, I, L, V, A, C, P, N.*
7. *C records the displayed* Cont *routes by which observer identity, locality, invariant readback, model audit, admissible rate or refusal, and consumer handoff are replayed.*

8. *P is the* Pkg *provenance over* $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InterHistInvariant}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$ *row.*

The classifier compares carriers only through these rows. It has no coordinate for an absolute frame, global synchronization history, host spacetime, quotient of histories, unbounded observation stream, detached physical truth oracle, or unledgered model equivalence.

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsTasteGate single carrier alignment`

Theorem 1275.2 (Relational physics NameCert obligations)**.** *For the carrier of Definition 1275.1, the local* $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (O, I, L, V, A, H, C, P, N)$, observer-history identity through O, inter-history locality through L, invariant readback through I, visible physical-model audit through V, admissible rate or displayed refusal through A, component-wise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsCarrier namecert obligations`

Proof. Project the accepted packet to the nine displayed rows. The rows O, I, L, V, A are the only relational and physical content coordinates, while H, C, P, N provide transport, replay, provenance, and naming.

Read observer identity first. The row O is governed by chapter 479, so it supplies no observer-substance, host observer equality, or quotient identity.

Read locality and invariant content next. The row L comes from the finite locality ledger of chapter 759, and the row I comes from the invariant readback of chapter 656. These rows keep the physical relation on displayed history packets rather than on a global frame.

Finally read the audit and rate boundary. The row V is the visible physical-model audit of chapter 1129, and A is either the licensed rate row or the refusal row. Since the carrier contains no further physical coordinate, the listed rows exhaust the NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsNameCertObligations`

Theorem 1275.3 (Relational physics invariant route exclusion)**.** *Every accepted* $\mathsf{RelationalPhysics}^{\uparrow}$ *consumer that reads relational physical content must factor through the displayed observer-history identity row O, locality ledger row L, inter-history invariant row I, visible physical-model audit row V, admissible rate or refusal row A, and the support rows H, C, P, N. No such consumer may add an absolute frame, synchronized global clock, ambient spacetime, quotient history equality, unbounded observation stream, detached physical truth oracle, or model equivalence outside the audit row.*

Proof. Start with carrier admission from Definition 1275.1. The public packet has exactly the rows $O, I, L, V, A, H, C, P, N$.

Apply the NameCert obligations of Definition 1275.2. They force identity, locality, invariant, audit, and rate-or-refusal reads to pass through the corresponding displayed rows before any consumer handoff can occur.

The exclusions are immediate from the carrier inventory. Absolute frames, synchronized clocks, ambient spacetime, quotient history equality, unbounded observations, detached truth oracles, and model equivalence outside the audit row are not coordinates of R and cannot be created by hsame, Cont, Pkg, or the local NameCert row. □

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsGlobalFrameTailExclusion`

Theorem 1275.4 (Relational physics NameCert handoff)**.** *The downstream handoff from an accepted* $\mathsf{RelationalPhysics}^{\uparrow}$ *packet is the finite route*

$$O, L, I, V, A, H, C, P, N.$$

It presents relational physical content as observer-history identity, inter-history locality, invariant readback, visible model audit, admissible rate or refusal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It does not present a completed physical theory, global geometry, semantic real-valued dynamics, or bridge-checked standard model.*

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysics namecert handoff`

Proof. Project the accepted carrier and order the consumer route by the displayed dependencies. The observer-history row O gives the identity surface, L gives locality, I gives the invariant readback, and V gives the physical-model audit.

The row A records whether the route carries an admissible rate certificate or an explicit refusal. Transport and replay are then supplied only by H and C, with P and N recording provenance and local naming.

Apply Definition 1275.3. The same route excludes global geometry, semantic real-valued dynamics, detached physical truth, and bridge checking because none is a displayed coordinate of the accepted packet. □

Theorem 1275.5 (Relational physics observer-locality-rate obligation)**.** *Every accepted* $\mathsf{RelationalPhysics}^{\uparrow}$ *packet that exports relational physical content must display observer-history identity, inter-history locality, invariant readback, physical-model audit, and admissible rate or refusal before the export. In carrier notation, the consumer route is*

$$O, L, I, V, A, H, C, P, N,$$

where O supplies observer-history identity, L supplies locality, I supplies invariant readback, V supplies visible model audit, A supplies the admissible rate certificate or refusal, and H, C, P, N supply transport, replay, provenance, and local naming. No export may bypass these rows by using an absolute frame, global synchronization history, ambient spacetime, detached truth oracle, unbounded observation stream, or unledgered model equivalence.

Proof. Project the carrier of Definition 1275.1. Its only relational-content rows are O, I, L, V, A, and its only structural rows are H, C, P, N.

Apply the NameCert obligations of Definition 1275.2. They require observer identity, locality, invariant readback, physical-model audit, and admissible rate or refusal to be read through their displayed rows before the local $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$ row can support a downstream export.

Use route exclusion and handoff, Definition 1275.3 and Definition 1275.4. The exclusion theorem rules out the non-displayed global and semantic endpoints, while the handoff theorem orders the surviving route as $O, L, I, V, A, H, C, P, N$. Thus the observer-locality-rate obligation is exactly the finite BEDC row route listed above. □

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsObserverLocalityRateObligation`

Theorem 1275.6 (Relational physics kernel scope closure)**.** *Every accepted* $\mathsf{RelationalPhysics}^{\uparrow}$ *packet scopes observer-history identity, inter-history locality, invariant readback, admissible rate or refusal, transport, replay, provenance, and local naming through explicit BEDC kernel rows. In carrier notation, the packet is read only through* BHist *rows for* $O, I, L, V, A, H, C, P, N$*,* hsame *transport in* H*,* Cont *replay in* C*,* Pkg *provenance in* P*, and the local* $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$ *row* N*.*

Proof. Begin with the carrier projection of Definition 1275.1. It displays the observer-history identity row O, invariant row I, locality row L, physical-model audit row V, and admissible rate or refusal row A as finite BHist data.

Apply the NameCert obligations of Definition 1275.2. The obligation surface requires each observer, locality, invariant, audit, and rate-or-refusal read to pass through those displayed rows before export.

Finally use Definition 1275.4 and Definition 1275.5. They order the structural route through H, C, P, N, so transport is hsame, replay is Cont, provenance is Pkg, and naming is exactly the local $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$ row. No remaining coordinate is available outside the BEDC kernel rows named in the statement. □

Lean4 ✓ `BEDC.Derived.RelationalPhysicsUp.RelationalPhysicsKernelScopeClosure`

Falsifiable prediction: Any accepted RelationalPhysicsUp consumer that exports relational physical content while bypassing observer-history identity, inter-history locality, invariant readback, physical-model audit, or the admissible rate and refusal row fails the carrier classifier.

Independence witness: observerhistidentity, interhistinvariant, interhistlocalityledger, physicalmodelaudit: none carries observer-history identity, locality ledger, invariant readback, visible model audit, admissible rate or refusal, componentwise hsame transport, Cont replay, Pkg provenance, and one local NameCert row in a single relational-physics packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RelationalPhysics}^{\uparrow}$ packet is generated from observer-history identity, inter-history invariant, locality ledger, physical-model audit, admissible rate or refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1275.1, Definition 1275.2, Definition 1275.3, Definition 1275.4, and Definition 1275.5 form the obligation surface for observer-history identity, locality, invariant readback, physical-model audit, rate or refusal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RelationalPhysics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not derive a completed physical theory, global spacetime geometry, semantic real-valued dynamics, host physical truth, or a standard bridge for relational physics.

Upgrade path: The next paper grade is public export for the observer-history, locality, invariant, physical-model audit, rate or refusal, transport, replay, provenance, and local NameCert rows.

Chapter 1276

A Concrete Naming Certificate for $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$

The $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ packet is the finite seal surface for the Real and Completion limit theorem of chapter 9238. It consumes the finite stream naming surface of chapter 126, the regular-rational readback surface of chapter 301, the dyadic endpoint ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 386. Its role is to bind one displayed window, readback, tolerance row, limit seal, RealUp handoff, and naming route into a consumer-facing envelope for Definition 9238.5.

Definition 1276.1 (Finite limit-seal envelope carrier)**.** *A* $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, R, D, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected for the displayed limit request;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger used to compare the readback with the requested precision;*
4. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row certifying that the finite window and readback feed the limit-seal route;*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *handoff row consumed by Real and Completion users;*
6. *H records componentwise* hsame *transport for W, R, D, L, Q and the local naming row;*
7. *C records displayed* Cont *replay from the finite window through readback, dyadic tolerance, limit seal, and RealUp handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSealEnvelope}^{\uparrow}}$ *row.*

The carrier exports only these finite rows and their support. It has no coordinate for quotient stream equality, host real equality, a selected host limit, countable choice, ambient completeness, or a second limit route.

Lean4 ✓ BEDC.Derived.FiniteLimitSealEnvelopeUp.FiniteLimitSealEnvelopeTasteGate single carrier alignment

Theorem 1276.2 (Finite limit-seal envelope NameCert obligations)**.** *For the carrier of Definition 1276.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSealEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for* $E = (W, R, D, L, Q, H, C, P, N)$*, finite StreamName window exposure through* W*, RegSeqRat readback through* R*, dyadic tolerance and endpoint exposure through* D*, CauchyLimitSeal handoff through* L*, terminal RealUp handoff through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The displayed packet lists one finite stream window, one regular-rational readback, one dyadic tolerance ledger, one completion-facing limit seal, one terminal RealUp handoff, and four structural support rows.

Read the data rows in order. The window W supplies the finite observation surface, R is the readback attached to that window, D records the dyadic tolerance evidence for the same request, and L records that these rows feed the CauchyLimitSeal route.

The terminal row Q is then available only as the RealUp handoff reached through W, R, D, L. The rows H, C, P, N transport, replay, source, and name that same finite packet, so the listed obligations exhaust the local NameCert surface. □

Theorem 1276.3 (Finite limit-seal envelope non-choice boundary)**.** *Every accepted* $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ *packet keeps the Real and Completion limit theorem on the displayed finite route*

$$W, R, D, L, Q, H, C, P, N.$$

A downstream consumer may inspect the StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchyLimitSeal row, RealUp handoff, transport, continuation, provenance, and local NameCert row. It receives no quotient stream equality, host real equality, selected host limit, countable-choice selector, ambient completeness principle, or hidden Real equality rule.

Proof. Start from the carrier inventory of Definition 1276.1. The only route to a Real-facing output is the ordered passage from W through R, D, L to Q.

Apply Definition 1276.2. The local certificate exposes exactly the displayed rows and treats H, C, P, N as support for the same packet, not as independent choice or equality data.

Inspect the excluded objects. Quotient stream equality, host real equality, selected host limits, countable-choice selectors, ambient completeness, and hidden Real equality rules would require coordinates outside $W, R, D, L, Q, H, C, P, N$. Since the carrier has no such coordinate, every accepted consumer remains on the finite route. □

Theorem 1276.4 (Finite limit-seal envelope sibling dependency)**.** *The* $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ *carrier depends on the joint availability of* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. No one sibling among chapter 126, chapter 301, chapter 291, chapter 386, and chapter 38 carries the whole tuple* $W, R, D, L, Q, H, C, P, N$*.*

Proof. Compare the finite observation siblings first. The StreamName surface supplies window data, the RegSeqRat surface supplies regular-rational readback, and the DyadicRatCore surface supplies tolerance and endpoint evidence. None of these siblings by itself also carries the completion-facing limit seal and the terminal RealUp handoff.

Compare the seal-facing siblings next. The CauchyLimitSeal surface records the completion handoff, and the RealUp surface records the terminal real-name seal, but neither one includes the finite window, readback, and dyadic tolerance ledger as a single ordered carrier.

Thus the envelope is a composite packet. Its sibling dependence is visible in the carrier itself, and the structural rows H, C, P, N preserve the combined route rather than reducing it to any one sibling surface. □

Theorem 1276.5 (Finite limit-seal envelope public surface)**.** *The public surface exported by an accepted* $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ *packet is exactly the finite Real and Completion handoff*

$$\mathsf{StreamName}^{\uparrow}(W), \mathsf{RegSeqRat}^{\uparrow}(R), \mathsf{DyadicRatCore}^{\uparrow}(D), \mathsf{CauchyLimitSeal}^{\uparrow}(L), \mathsf{Real}^{\uparrow}(Q), H, C, P, N.$$

This surface realizes the limit theorem of Definition 9238.5 as a consumer-facing NameCert envelope without exporting quotient streams, host real equality, selected host limits, countable choice, ambient completeness, or a bridge row.

Proof. Use Definition 1276.2 to identify the rows exported by the local certificate. The exported data rows are exactly W, R, D, L, Q, and the support rows are exactly H, C, P, N.

Use Definition 1276.4 to read those rows as the joint StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, and RealUp handoff required by the Real and Completion route. The theorem does not collapse the route to any one sibling.

Finally apply Definition 1276.3. Since the excluded equality, choice, completeness, and bridge coordinates are absent, the public surface is precisely the finite envelope displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSealEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteLimitSealEnvelope}^{\uparrow}$ packet is generated from a finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchyLimitSeal row, RealUp handoff, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLimitSealEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1276.1, Definition 1276.2, Definition 1276.3, Definition 1276.4, and Definition 1276.5 bind the finite composite carrier, NameCert obligations, non-choice boundary, sibling dependency, and public Real and Completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLimitSealEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, host real equality, selected host limits, countable choice, ambient completeness, hidden Real equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window locality, RegSeqRat readback ordering, DyadicRatCore tolerance exposure, CauchyLimitSeal handoff, RealUp handoff, transport stability, replay, provenance, and local naming for the same finite packet.

Chapter 1277

A Concrete Naming Certificate for $\mathsf{PhilosophyMapTarget}^{\uparrow}$

The $\mathsf{PhilosophyMapTarget}^{\uparrow}$ packet realizes the formal target draft of subsection 9462.33.11 as a finite NameCert carrier. It sits beside the concept registry surface of chapter 1252, the core registry index of chapter 1251, the audit system of chapter 1258, the axiom-purity gate of chapter 1259, and the closure-status reference of chapter 1265. Those siblings register concepts, index registry rows, audit accepted packets, rule out host-level axioms, and display closure ledgers; this carrier binds them into one philosophy-map target only when every exported coordinate remains a named BEDC row.

Definition 1277.1 (Philosophy map target carrier)**.** *A* $\mathsf{PhilosophyMapTarget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, G, K, S, A, X, H, C, P, N)$$

with the following displayed rows:

1. *R is the registry-map row, read through* $\mathsf{ConceptRegistrySurface}^{\uparrow}$ *and* $\mathsf{CoreRegistryIndex}^{\uparrow}$*, naming concept entries, theorem entries, tradition entries, scientific entries, and registry-route entries;*
2. *G is the gap-map row, naming typed gaps, formal target rows, missing bridge rows, and not-claimed surfaces;*
3. *K is the cannot-claim row, recording blocked readings for subject substance, global truth, host equality, quotient import, choice import, and hidden far-end objects;*
4. *S is the closure-status row, read through* $\mathsf{ClosureStatusReference}^{\uparrow}$*, recording paper closure, verification status, bridge status, and not-claimed scope for every exported row;*
5. *A is the audit row, read through* $\mathsf{AuditSystem}^{\uparrow}$ *and* $\mathsf{AxiomPurityGate}^{\uparrow}$*, recording the finite audit route that rejects forbidden host-equality, quotient, choice, and global-truth reads;*
6. *X is the synthesis row, recording the philosophy-map target named in Definition 9462.2196 as registry maps, gap maps, cannot-claim conflict checks, closure-status rows, no-smuggling refusal, and audit evidence;*
7. *H records componentwise* hsame *transport for R, G, K, S, A, X;*
8. *C records displayed* Cont *replay through the registry maps, gap maps, cannot-claim checks, closure rows, audit rows, and synthesis row;*

9. *P is the* Pkg *provenance over* ConceptRegistrySurface$^{\uparrow}$, CoreRegistryIndex$^{\uparrow}$, AuditSystem$^{\uparrow}$, AxiomPurityGate$^{\uparrow}$, ClosureStatusReference$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{PhilosophyMapTarget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a prose-only philosophical map, an unchecked truth predicate, a hidden host equality, a quotient witness, a choice operator, or a closure-status read outside the displayed packet.

Lean4 ✓ `BEDC.Derived.PhilosophyMapTargetUp.TasteGate.PhilosophyMapTargetTasteGate single carrier alignment`

Theorem 1277.2 (Philosophy map target NameCert obligations)**.** *For the carrier of Definition 1277.1, the local* NameCert$_{\text{PhilosophyMapTarget}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for $M = (R, G, K, S, A, X, H, C, P, N)$;*
2. *registry-map coverage through R, requiring every concept, theorem, tradition, scientific, and route entry to factor through* ConceptRegistrySurface$^{\uparrow}$ *and* CoreRegistryIndex$^{\uparrow}$*;*
3. *gap-map and cannot-claim coverage through G and K, requiring every missing proof, formal target, blocked reading, and cannot-claim conflict to be displayed before export;*
4. *closure-status and audit coverage through S and A, requiring every paper closure, verification grade, bridge status, not-claimed surface, and axiom-purity refusal to be read through* ClosureStatusReference$^{\uparrow}$, AuditSystem$^{\uparrow}$, *and* AxiomPurityGate$^{\uparrow}$*;*
5. *structural preservation through X, H, C, P, N, requiring the synthesis row, componentwise transport, displayed replay routes, package provenance, and local naming to preserve the same finite packet.*

Lean4 ✓ `BEDC.Derived.PhilosophyMapTargetUp.PhilosophyMapTargetTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1277.1. The rows R, G, K, S, A, X expose the registry, gap, cannot-claim, closure-status, audit, and synthesis data before any philosophy-map target can be accepted.

The registry row factors through the concept registry surface and core registry index. The gap and cannot-claim rows keep missing proof sites, formal targets, blocked readings, and conflict checks as displayed coordinates rather than prose authority.

The closure-status and audit rows bind every exported reading to a visible closure ledger and to an axiom-purity route. The remaining rows X, H, C, P, N transport, replay, source, and name that same packet, so the five listed rows exhaust the local NameCert surface. □

Theorem 1277.3 (Philosophy map target no-smuggling boundary)**.** *Every accepted* PhilosophyMapTarget$^{\uparrow}$ *packet exposes philosophy-map content only through the rows $R, G, K, S, A, X, H, C, P, N$. If the no-smuggling row K is absent, a consumer may request a forbidden host-equality, quotient, choice, or global-truth read; with K present, no public consumer can read those coordinates from the packet.*

Proof. Use Definition 1277.2. The cannot-claim row K is the only row that records the forbidden readings, while A records the audit route that rejects them and S records the closure-status boundary where they remain not claimed.

If K is removed, the carrier no longer contains the displayed refusal row for host equality, quotient import, choice import, or global-truth import. A request for one of those readings is then

no longer blocked by the accepted packet, because the remaining registry, gap, closure, audit, and synthesis rows do not themselves name every refusal coordinate.

With K present, the refusal row is transported by H, replayed only through C, sourced by P, and named by N. A public consumer receives the finite packet and cannot project any forbidden coordinate not already displayed there. □

Theorem 1277.4 (Philosophy map target public law boundary)**.** *Every public* $\mathsf{PhilosophyMapTarget}^{\uparrow}$ *export factors through the registry-map row R, gap-map row G, cannot-claim row K, closure-status row S, audit row A, synthesis row X, and structural rows H, C, P, N. The export therefore carries only the displayed registry, gap, refusal, closure, audit, synthesis, transport, replay, provenance, and naming coordinates, and no public consumer can read a prose-only philosophy map, unchecked truth predicate, hidden host equality, quotient witness, choice operator, or closure-status row outside the packet.*

Proof. Start with the carrier of Definition 1277.1. It exposes exactly the rows $R, G, K, S, A, X, H, C, P, N$ and has no coordinate for a prose-only philosophical map, unchecked truth predicate, hidden host equality, quotient witness, choice operator, or external closure-status read.

Apply the NameCert obligations of Definition 1277.2. Registry-map coverage is read through R, gap-map and cannot-claim coverage are read through G and K, closure-status and audit coverage are read through S and A, and the synthesis and structural obligations preserve the same finite packet through X, H, C, P, N.

Finally use the no-smuggling boundary of Definition 1277.3. Since the forbidden readings are blocked by K and the audit and closure rows remain displayed, a public export can factor only through the named rows and cannot acquire an additional public-law coordinate. □

Falsifiable prediction: Every accepted $\mathsf{PhilosophyMapTarget}^{\uparrow}$ packet whose no-smuggling row is removed admits a forbidden host-equality, quotient, choice, or global-truth read; with the row present no public consumer can read those coordinates.

Independence witness: conceptregistrysurface, coreregistryindex, auditsystem, axiompurity-gate, closurestatusreference: none is bijective because $\mathsf{PhilosophyMapTarget}^{\uparrow}$ simultaneously carries registry maps, gap maps, cannot-claim conflict checks, closure-status rows, and no-smuggling refusal in one packet.

Theorem 1277.5 (Philosophy map target bridge ledger)**.** *Every bridge-facing* $\mathsf{PhilosophyMapTarget}^{\uparrow}$ *read factors through the registry row R, gap row G, cannot-claim row K, closure-status row S, audit row A, synthesis row X, transport row H, replay row C, provenance row P, and local* $\mathsf{NameCert}$ *row N. Thus a bridge consumer receives only the displayed registry, gap, refusal, closure, audit, synthesis, transport, replay, provenance, and naming coordinates.*

Proof. Use the carrier of Definition 1277.1. The rows R, G, K, S, A, X are the only content rows exposed by the packet, while H, C, P, N are the only structural rows available to transport, replay, source, and name them.

Apply the public law boundary of Definition 1277.4. Since public export already factors through the same ten rows, a bridge-facing read has no additional coordinate from which to import a prose-only philosophy map, hidden host equality, quotient witness, choice operator, global truth predicate, or closure-status row outside the packet. □

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{PhilosophyMapTarget}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PhilosophyMapTarget}^{\uparrow}$ instances in `lean4/BEDC/Derived/PhilosophyMapTargetUp/TasteGate.lean`. Those instances

must preserve $R, G, K, S, A, X, H, C, P, N$ as separate displayed fields before the target BEDC.Derived.PhilosophyMa can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{PhilosophyMapTarget}^{\uparrow}}$.

Constructive story: A $\mathsf{PhilosophyMapTarget}^{\uparrow}$ packet is generated from registry-map, gap-map, cannot-claim, closure-status, audit, synthesis, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhilosophyMapTarget}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1277.1, Definition 1277.2, Definition 1277.3, Definition 1277.4, and Definition 1277.5 bind the displayed carrier, NameCert obligations, no-smuggling boundary, sibling scope, public export, and bridge-facing ledger over $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, and $\mathsf{ClosureStatusReference}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhilosophyMapTarget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, philosophical completeness, global truth, host equality, quotient import, choice import, or all downstream registry entries.

Upgrade path: Lean closure requires Nontrivial PhilosophyMapTargetUp and FieldFaithful PhilosophyMapTargetUp instances in `lean4/BEDC/Derived/PhilosophyMapTargetUp/TasteGate.lean`.

Chapter 1278

A Concrete Naming Certificate for $\mathsf{DiagonalIndexCofinality}^{\uparrow}$

The $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ packet realizes the finite modulus-threshold row used by the Real completeness proof target of section 9238.2. For a regular Cauchy input and precision request n, it records the displayed inequality that the selected diagonal index is at least $\mu(1/(n+1))$ before any $\mathsf{Real}^{\uparrow}$ completion-facing consumer reads the diagonal representative. The packet sits between the representative route of chapter 1205, the cofinal modulus seal of chapter 1198, the tail budget of chapter 1231, the stream naming surface of chapter 126, the regular-rational row of chapter 301, the dyadic observation core of chapter 291, and the terminal Real surface of chapter 38.

The certificate is deliberately finite. It names the threshold comparison, the selected diagonal window, the regular-rational readback, the dyadic tolerance row, the Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. It does not export quotient stream equality, a host-selected limit, countable choice, an unbounded search principle, or an ambient completeness theorem.

Definition 1278.1 (Diagonal index cofinality carrier)**.** *A* $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, \mu, n, k, i, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted regular Cauchy input family over* $\mathsf{RegSeqRat}^{\uparrow}$*;*
2. *μ is the displayed modulus row whose precision read at $1/(n+1)$ returns k;*
3. *n is the finite precision request made by the Real-completion consumer;*
4. *k is the threshold row $k = \mu(1/(n+1))$;*
5. *i is the selected diagonal index row with $i = k + n + 1$ and hence $i \geq k$;*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened at the selected diagonal index;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected member and index;*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite read;*
9. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after W, R, D and the threshold comparison have been displayed;*

10. *H records componentwise* hsame *transport for the input family, modulus row, request, threshold, index, window, readback, dyadic ledger, Real handoff, route, provenance, and naming rows;*
11. *C records the displayed* Cont *replay from the request through the modulus threshold and selected diagonal index into the window, readback, dyadic ledger, and Real handoff;*
12. *P is the* Pkg *provenance over* $\mathsf{DiagonalIndexCofinality}^{\uparrow}$, $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$, $\mathsf{CofinalModulusSeal}^{\uparrow}$, $\mathsf{TailCofinalityBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalIndexCofinality}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $X, \mu, n, k, i, W, R, D, E, H, C, P, N$ and the visible threshold proof $i \geq k$.

Lean4 ✓ `BEDC.Derived.DiagonalIndexCofinalityUp.DiagonalIndexCofinalityTasteGate single carrier alignment`

Theorem 1278.2 (Diagonal index cofinality NameCert obligation surface)**.** *For every accepted* $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ *carrier $I = (X, \mu, n, k, i, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DiagonalIndexCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier transport through the displayed rows, modulus-threshold exposure $k = \mu(1/(n+1))$, diagonal-index exposure $i = k+n+1$, cofinality exposure $i \geq k$, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger exposure through D,* $\mathsf{Real}^{\uparrow}$ *handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DiagonalIndexCofinalityUp.DiagonalIndexCofinalityNameCertObligationSurface`

Proof. Project the accepted packet to the displayed tuple in Definition 1278.1. The input family, modulus row, precision request, threshold, selected index, finite window, readback, dyadic ledger, Real handoff, transport, route, provenance, and naming row are all visible coordinates.

The arithmetic content is confined to the threshold and selected-index rows. The carrier records $k = \mu(1/(n+1))$ and $i = k + n + 1$, so the cofinality comparison $i \geq k$ is read from the displayed successor extension rather than from an unbounded search.

The downstream certificate rows then follow the same ordered route as the representative and seal siblings. The $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{Real}^{\uparrow}$ handoff are consumed only after the threshold comparison has been displayed, while H, C, P, N only transport, replay, source, and name those rows. □

Theorem 1278.3 (Diagonal index cofinality window monotonicity)**.** *For an accepted* $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ *carrier $I = (X, \mu, n, k, i, W, R, D, E, H, C, P, N)$, every downstream finite-window read exposed through W, R, D, E is indexed at i and therefore lies at or beyond the modulus threshold $k = \mu(1/(n+1))$. If the same packet is transported by the componentwise* hsame *row H, the transported window preserves the same threshold comparison before the* $\mathsf{Real}^{\uparrow}$ *handoff can be consumed.*

Proof. The selected window row W is attached to the index coordinate i of the carrier. By Definition 1278.1, this coordinate is not independent of the modulus row: it is the displayed value $k + n + 1$ with $k = \mu(1/(n+1))$.

Apply the obligation surface of Definition 1278.2. It requires any accepted read of W, R, D, E to preserve the threshold row, selected-index row, and cofinality comparison before the readback or handoff row is public.

Transport along H is componentwise. Hence it carries the request, threshold, selected index, window, readback, dyadic ledger, and Real handoff together, and it cannot move the finite window to a pre-threshold index without changing one of the displayed coordinates. □

Lean4 ✓ `BEDC.Derived.DiagonalIndexCofinalityUp.TasteGate.DiagonalIndexCofinalityTasteGate single carrier alignment`

Theorem 1278.4 (Diagonal index cofinality Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Real-completion consumer of an accepted* $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ *packet factors through the finite route*

$$X, \mu, n, k, i, W, R, D, E, H, C, P, N.$$

In particular, the consumer receives the diagonal-index cofinality row $i \geq \mu(1/(n+1))$ *before using the diagonal representative in the limit proof of Definition 9238.5.*

Proof. Start with the route displayed by the carrier. The precision request n first reads the modulus row μ to obtain k, then forms the selected diagonal index $i = k + n + 1$.

Next apply Definition 1278.3. It shows that the finite StreamName, RegSeqRat, dyadic, and Real rows exposed by the packet all remain at or beyond the modulus threshold.

Finally read the Real handoff through E and the support rows H, C, P, N. Since the carrier has no coordinate for a direct host limit, quotient stream equality, unbounded selection, countable choice, or ambient completeness, every consumer of this packet receives exactly the finite cofinality handoff displayed above. □

Falsifiable prediction: A diagonal representative consumer that reaches a RealUp handoff without displaying the modulus threshold k, selected index i, and proof that i is at least k is not an accepted DiagonalIndexCofinalityUp carrier.

Independence witness: diagonallimitrepresentative, cofinalmodulusseal, tailcofinalitybudget, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{DiagonalIndexCofinality}^{\uparrow}$, because each sibling records a representative, seal, budget, window, readback, dyadic observation, or Real handoff row without binding the explicit diagonal-index inequality $i \geq \mu(1/(n+1))$ as its own local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalIndexCofinality}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{DiagonalIndexCofinality}^{\uparrow}$ packet is generated from a regular Cauchy input family, modulus row, precision request, threshold row, selected diagonal index, finite StreamName window, RegSeqRat readback, DyadicRatCore ledger, RealUp handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiagonalIndexCofinality}^{\uparrow}}$, seed)

Scope closed: Definition 1278.1, Definition 1278.2, Definition 1278.3, and Definition 1278.4 bind the finite diagonal-index cofinality carrier and its Real-completion handoff route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiagonalIndexCofinality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Real completeness, quotient stream equality, host real equality, countable choice, unbounded diagonal search, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, modulus-threshold exposure, selected-index cofinality, finite-window monotonicity, Real handoff non-escape, and TasteGate nontriviality with field faithfulness for $X, \mu, n, k, i, W, R, D, E, H, C, P, N$.

Chapter 1279

A Concrete Naming Certificate for $\mathsf{SocketReport}^{\uparrow}$

The $\mathsf{SocketReport}^{\uparrow}$ packet realizes the socket-report discipline of subsection 9462.26.14 as a finite NameCert carrier. It sits beside the typed supply socket of chapter 1243, the socket-kind classifier of chapter 1058, the supply socket ledger of chapter 528, the audit gate boundary of chapter 526, and the closed observation system of chapter 616. Those siblings expose typed requests, kind classification, ledger accounting, gate locality, and closed observation separately; this packet binds them into one report object that audit consumers can inspect without broadening the reported socket kind.

Definition 1279.1 (Socket report carrier)**.** *A* $\mathsf{SocketReport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, U, K, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the site row, recording the proof, construction, or audit position where a missing supply would be consumed;*
2. *U is the requested-supply row, recording the exact supply requested at S rather than a generic gap label;*
3. *K is the narrow socket-kind row, recording the smallest socket kind that explains the requested supply;*
4. *G is the audit-gate row, recording the local gate that accepts, blocks, or refuses internalization of the request;*
5. *H stores componentwise* hsame *transport for the site, requested-supply, narrow-kind, audit-gate, continuation, provenance, and local naming rows;*
6. *C stores displayed* Cont *continuation from the reported site through the requested supply, classifier, gate, provenance, and local name;*
7. *P is the* Pkg *provenance over* $\mathsf{TypedSupplySocket}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SocketReport}^{\uparrow}}$ *row.*

The carrier has no coordinate for replacing K by a broader gap family, discharging U, moving G to a different site, or hiding the report behind the closed observation substrate.

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportTasteGate single carrier alignment`

Theorem 1279.2 (Socket report NameCert obligations)**.** *For the carrier of Definition 1279.1, the local* $\mathsf{NameCert}_{\mathsf{SocketReport}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$R = (S, U, K, G, H, C, P, N),$$

site exposure through S*, requested-supply exposure through* U*, narrow-kind classifier stability through* K*, audit-gate locality through* G*, componentwise* hsame *transport through* H*, displayed* Cont *continuation through* C*, provenance through* P*, local naming through* N*, and a non-discharge ledger stating that the report names a boundary request without supplying the requested row.*

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportNameCertObligations`

Proof. Project the finite packet $R = (S, U, K, G, H, C, P, N)$. The site, requested-supply, narrow-kind, and audit-gate rows are displayed before transport, continuation, provenance, and naming are read, so the certificate cannot introduce an unreported site or a hidden request.

Read the classifier and gate rows together. The row K is stable because it is the only row whose role is to name the narrow socket kind for the displayed request U at site S. The row G is local because it is read in the same packet as S and cannot be reassigned after continuation replay.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite packet. None of them contains a supply witness for U, so the non-discharge ledger remains part of the local NameCert surface. □

Theorem 1279.3 (Socket report minimal classification)**.** *Every accepted* $\mathsf{SocketReport}^{\uparrow}$ *report preserves the narrowest socket kind displayed by* K*. In particular, an accepted report for the same requested-supply row* U *cannot replace a Type II witness request, a Type III identity request, or any other displayed narrow socket kind by a broader gap label while remaining the same NameCert report.*

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportUp minimal classification`

Proof. Assume an accepted packet $R = (S, U, K, G, H, C, P, N)$. By Definition 1279.2, the classifier row K is part of the local NameCert surface and is transported only componentwise with the same requested-supply row U.

If an audit consumer broadens the report while preserving U, it must either erase K or replace K by a different classifier row. Erasing K leaves the carrier of Definition 1279.1; replacing it changes the local naming row N because the certificate names the displayed packet.

Thus Type II, Type III, and the other narrow socket rows remain public whenever the report is accepted. A broader gap label may be written as a separate report, but it is not the same accepted $\mathsf{SocketReport}^{\uparrow}$ packet for the displayed requested supply. □

Theorem 1279.4 (Socket report sibling separation)**.** *The* $\mathsf{SocketReport}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{TypedSupplySocket}^{\uparrow}$*,* $\mathsf{SocketKindClassifier}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*,* $\mathsf{AuditGateBoundary}^{\uparrow}$*, or* $\mathsf{ClosedObservationSystem}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportUp sibling separation`

Proof. Compare the displayed rows. $\mathsf{TypedSupplySocket}^{\uparrow}$ carries typed socket data, $\mathsf{SocketKindClassifier}^{\uparrow}$ classifies kinds, $\mathsf{SupplySocketLedger}^{\uparrow}$ accounts for requested supply, $\mathsf{AuditGateBoundary}^{\uparrow}$ exposes gate locality, and $\mathsf{ClosedObservationSystem}^{\uparrow}$ records closed observational discipline.

None of those siblings carries the simultaneous report rows S, U, K, G, H, C, P, N with a non-discharge ledger for the same site and request. A NameCert-row-preserving bijection would have to preserve that joint report surface, so no listed sibling is bijective to $\mathsf{SocketReport}^{\uparrow}$. □

Theorem 1279.5 (Socket report scoped report route)**.** *Every scoped read of an accepted* $\mathsf{SocketReport}^{\uparrow}$ *packet factors through*

$$S, U, K, G, H, C, P, N$$

before any audit consumer reads the report. The route preserves the requested-supply row U, the narrow classifier row K, the audit-gate row G, the non-discharge ledger, componentwise transport, displayed replay, provenance, and local naming; it cannot replace the report by a broader gap family or absorb it into $\mathsf{TypedSupplySocket}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, *or* $\mathsf{ClosedObservationSystem}^{\uparrow}$.

Proof. Project the carrier definition of Definition 1279.1. The accepted packet displays exactly S, U, K, G, H, C, P, N, so a scoped report read begins from the site, requested supply, classifier, and gate rows before structural replay is available.

Use the NameCert obligations of Definition 1279.2. They expose the site row, requested-supply row, narrow-kind classifier, audit gate, transport, continuation, provenance, local naming, and non-discharge ledger as one report surface.

The minimal classification theorem Definition 1279.3 prevents replacing K by a broader gap label while keeping the same requested supply. The sibling separation theorem Definition 1279.4 then rules out absorbing the full report surface into any listed sibling, because none of them carries the simultaneous rows S, U, K, G, H, C, P, N with the same non-discharge ledger. □

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportScopedReportRoute`

Theorem 1279.6 (Socket report public export surface)**.** *The public export of an accepted* $\mathsf{SocketReport}^{\uparrow}$ *packet is exactly the visible tuple*

$$S, U, K, G, H, C, P, N$$

from Definition 1279.1. The export preserves the narrow classifier row K and the non-discharge ledger attached to U; it does not broaden K, discharge U, move G away from the displayed site S, or absorb the packet into any sibling socket interface.

Proof. Start with the scoped report route of Definition 1279.5. It forces every accepted read to pass through S, U, K, G, H, C, P, N before any consumer can inspect the report.

The public export forgets no displayed report row. The typed request is U, the socket-kind classifier is K, the audit gate is G, the transport and closed-observation replay are H and C, and the supply-ledger provenance is carried by P together with the local naming row N.

Minimal classification from Definition 1279.3 prevents replacing K by a broader gap family, while sibling separation from Definition 1279.4 prevents absorbing the joint packet into any listed socket sibling. Thus the public surface exports the scoped rows without widening the socket kind. □

Lean4 ✓ `BEDC.Derived.SocketReportUp.SocketReportPublicExportSurface`

Falsifiable prediction: An accepted $\mathsf{SocketReport}^{\uparrow}$ packet is refuted by any audit consumer that replaces the displayed narrow socket kind with a broader gap label while preserving the same requested-supply row.

Independence witness: typedsupplysocket, socketkindclassifier, supplysocketledger, auditgateboundary, closedobservationsystem: TypedSupplySocketUp binds typed socket rows but not the

full report continuation; SocketKindClassifierUp lacks site, gate, and non-discharge rows; SupplySocketLedgerUp accounts for requests without preserving the narrow classifier as a report field; AuditGateBoundaryUp localizes gates without carrying the requested-supply ledger; ClosedObservationSystemUp records substrate closure rather than a visible gap report. None is bijective to the full $\mathsf{SocketReport}^{\uparrow}$ carrier under NameCert-row preservation.

The Lean follow-up file `lean4/BEDC/Derived/SocketReportUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ SocketReport}^{\uparrow}$ and $\mathsf{FieldFaithful\ SocketReport}^{\uparrow}$ instances. The field-faithfulness witness must preserve S, U, K, G, H, C, P, N as distinct rows so a checked TasteGate theorem can rule out broad-gap replacement of a narrow accepted report.

Closure status for $\mathsf{NameCert}_{\mathsf{SocketReport}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SocketReport}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1279.1, Definition 1279.2, Definition 1279.3, Definition 1279.4, Definition 1279.5, Definition 1279.6, the falsifiable prediction, the independence witness, and the TasteGate marker bind the public typed request, socket-kind classifier, supply-ledger provenance, audit gate, closed-observation replay, transport, and local naming rows of the displayed report packet without broadening the socket kind.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SocketReport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim stack certification, typed supply availability, phase-refusal verdicts, or broad-gap replacement of the report classifier.

Upgrade path: A higher closure grade requires Lean-side verification of the public export surface and any standard bridge demanded by downstream socket-report schemas.

Constructive story: The socket report is generated from a $BHist$ packet that contains a requested-supply row, socket kind, site, gate, narrow report classifier, and the continuation needed to identify it as a socket-facing report. The carrier does not include stack certification, typed supply, or phase-refusal rows; it reads only the report packet and transports that packet along $hsame$. A Cont extension may carry the report to a later consumer, but it must preserve the minimal classification row. The NameCert obligations, minimal classification theorem, sibling separation, prediction, independence witness, and TasteGate marker make the report a narrow ledger object.

Chapter 1280

A Concrete Naming Certificate for $\mathsf{StatusDowngrade}^{\uparrow}$

The $\mathsf{StatusDowngrade}^{\uparrow}$ packet realizes the downgrade discipline of chapter 9462 as a finite NameCert carrier for lowering an advertised report status after a verification failure. It sits beside chapter 1233, chapter 1237, chapter 901, chapter 1265, and chapter 1259. Those siblings expose failure reasons, typed failure branches, audit-gate diagnostics, closure-status references, and axiom-purity gates; this packet binds the previous advertised status, the displayed failure certificate, the lowered report, and the route by which the public gap row remains visible.

Definition 1280.1 (Status downgrade carrier)**.** *A* $\mathsf{StatusDowngrade}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, F, L, G, R, V, H, C, P, N)$$

with the following displayed rows:

1. *O is the prior-status row, recording the theory-closure, formal-status, bridge-status, and public report row that had been advertised;*
2. *F is the failure-certificate row, consumed from* $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{TypedFailureCertificate}^{\uparrow}$, AuditGateFailureWitne $\mathsf{ClosureStatusReference}^{\uparrow}$, *or* $\mathsf{AxiomPurityGate}^{\uparrow}$*;*
3. *L is the lowered-report row, recording the status that remains licensed after the failure certificate is displayed;*
4. *G is the exposed gap row, naming the theorem, definition, bridge, audit, axiom-purity, or closure field that blocks the prior status;*
5. *R is the downgrade-route row, displaying the finite route from O through F and G to L;*
6. *V is the verification-report row whose status is being lowered and whose public read must retain the prior-status and failure-certificate rows;*
7. *H stores componentwise* hsame *transport for prior status, failure certificate, lowered report, gap row, route, verification report, provenance, and local naming rows;*
8. *C stores displayed* Cont *replay for reading the prior status before the failure certificate, reading the failure certificate before the gap row, and reading the gap row before the lowered report;*
9. *P is the* Pkg *provenance over* $\mathsf{StatusDowngrade}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{TypedFailureCertificate}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, $\mathsf{ClosureStatusReference}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{StatusDowngrade}^{\uparrow}}$ *row.*
The carrier has no coordinate for silently erasing the prior status, hiding the failure certificate, promoting the lowered report above the displayed gap, or replacing the finite downgrade route by an unchecked host report.

Lean4 (definition) `BEDC.Derived.StatusDowngradeUp.StatusDowngradeUp`

Theorem 1280.2 (StatusDowngrade NameCert obligations)**.** *For the carrier of Definition 1280.1, the local* $\mathsf{NameCert}_{\mathsf{StatusDowngrade}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$S = (O, F, L, G, R, V, H, C, P, N),$$

prior-status display through O, failure-certificate display through F, lowered-report display through L, exposed-gap display through G, downgrade-route display through R, verification-report display through V, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite carrier. The displayed BHist packet lists the prior status, failure certificate, lowered report, gap row, downgrade route, verification report, transport, replay, provenance, and local naming rows as separate coordinates.

Read the sibling certificate rows next. $\mathsf{FailureCertificate}^{\uparrow}$ and $\mathsf{TypedFailureCertificate}^{\uparrow}$ supply the failed gate and typed branch; $\mathsf{AuditGateFailureWitness}^{\uparrow}$ supplies the diagnostic row; $\mathsf{ClosureStatusReference}^{\uparrow}$ supplies the bounded report status; $\mathsf{AxiomPurityGate}^{\uparrow}$ supplies the forbidden-dependency refusal when that is the failing field.

The support rows only preserve this inventory. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming cannot add a hidden stronger report, so the listed rows exhaust the local NameCert surface. □

Theorem 1280.3 (StatusDowngrade strength monotonicity)**.** *Every accepted* $\mathsf{StatusDowngrade}^{\uparrow}$ *packet*

$$S = (O, F, L, G, R, V, H, C, P, N)$$

exports a report no stronger than the lowered-report row L after the failure-certificate row F is read. A consumer may still display the prior-status row O, but it cannot advertise O as current status unless it also removes the displayed failure certificate and thereby leaves the accepted packet.

Proof. Start with the NameCert obligations of Definition 1280.2. They force a public read to display both the prior-status row O and the failure-certificate row F before the lowered report L is consumed.

The downgrade route R records the ordering from O through F and G to L. Thus the public report may remember the previous status as evidence of what was lowered, but the exported status after the route is the row L.

Finally inspect the carrier inventory. No coordinate supplies a second current-status row stronger than L, and H, C, P, N only transport, replay, source, and name the same finite packet. Treating O as current status would erase F or bypass R, so it is not an accepted read of S. □

Theorem 1280.4 (StatusDowngrade gap routing)**.** *For every accepted* $\mathsf{StatusDowngrade}^{\uparrow}$ *packet*

$$S = (O, F, L, G, R, V, H, C, P, N),$$

the exposed gap row G is routed between the failure-certificate row F and the lowered-report row L. The lowered report cannot be read as a bare editorial decision, a detached audit note, a hidden

checker result, or an ambient closure correction; it must pass through the displayed finite route $O, F, G, R, L, H, C, P, N$.

Proof. Project the route and replay rows. The route R and continuation row C display the readback from prior status to failure certificate, from failure certificate to exposed gap, and from exposed gap to lowered report.

Use the sibling certificates named in the carrier provenance. Failure-certificate, typed-failure, audit-gate witness, closure-status reference, and axiom-purity gate rows each provide a concrete blocking surface rather than an editorial status change.

The support rows preserve the same route. Since hsame, Cont, Pkg, and NameCert do not create a detached report coordinate, every accepted lowered report remains routed through G and the finite downgrade route. □

Theorem 1280.5 (StatusDowngrade nonescape)**.** *An accepted* $\mathsf{StatusDowngrade}^{\uparrow}$ *packet exports only prior status, failure certificate, lowered report, exposed gap, downgrade route, verification-report row, componentwise transport, replay, provenance, and local naming. It does not export silent status erasure, hidden promotion, unchecked report repair, host proof search, bridge completion, axiom-purity repair, or a verification status stronger than* L.

Proof. Inspect Definition 1280.1. The carrier has exactly the rows $O, F, L, G, R, V, H, C, P, N$ and deliberately lacks hidden repair, promotion, host-search, bridge-completion, or axiom-repair coordinates.

Apply strength monotonicity from Definition 1280.3. After the failure certificate is read, the accepted exported status is bounded by L.

Apply gap routing from Definition 1280.4. The lowered report reaches consumers only through the displayed gap row and downgrade route, so no detached report can escape the packet. Therefore every listed stronger or hidden report lies outside the accepted carrier. □

Theorem 1280.6 (StatusDowngrade scoped gap route)**.** *For every accepted* $\mathsf{StatusDowngrade}^{\uparrow}$ *packet*

$$S = (O, F, L, G, R, V, H, C, P, N),$$

every accepted downgrade read is scoped by the prior-status row O*, failure-certificate row* F*, lowered-report row* L*, exposed-gap row* G*, downgrade-route row* R*, verification-report row* V*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. A consumer may read the lowered report only along this finite route, so the downgrade cannot be detached from the failure certificate, the exposed gap, or the local NameCert surface.*

Proof. Start from the carrier of Definition 1280.1. Its displayed rows are exactly $O, F, L, G, R, V, H, C, P, N$, so an accepted read has no coordinate outside the prior status, failure certificate, lowered report, exposed gap, route, verification report, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 1280.2. They force the public downgrade surface to expose each of the displayed rows before the packet is named as a local $\mathsf{StatusDowngrade}^{\uparrow}$ certificate.

Now use strength monotonicity and gap routing from Definition 1280.3 and Definition 1280.4. The lowered report is bounded by L after the failure certificate is read, and the gap row G lies on the route from F to L.

Finally apply nonescape from Definition 1280.5. Hidden promotion, silent status erasure, detached repair, and status conversion outside the route are absent from the carrier, so the displayed rows scope every accepted downgrade read. □

Falsifiable prediction: A StatusDowngradeUp packet is refuted by any lowered report status that lacks a displayed prior-status row and failure-certificate row.

Independence witness: failurecertificate, typedfailurecertificate, auditgatefailurewitness, closurestatusreference, axiompuritygate: none is bijective because only StatusDowngradeUp stores prior status, failure certificate, lowered report, exposed gap row, downgrade route, provenance, and local NameCert together.

The Lean follow-up for this chapter must provide `Derived/StatusDowngradeUp/TasteGate.lean` with Nontrivial StatusDowngrade$^{\uparrow}$ and FieldFaithful StatusDowngrade$^{\uparrow}$ instances for the displayed carrier fields.

Closure status for NameCert$_{\text{StatusDowngrade}^{\uparrow}}$.

Constructive story: The status downgrade certificate is generated from $BHist$ rows that record an original status, a failure certificate, a weaker report status, and the exposed gap route that explains why the weaker status is the admitted one. The carrier first reads the strength ordering, then records the gap routing row, and finally allows Cont to carry the weaker status forward without silently restoring the stronger one. The $hsame$ relation keeps the downgrade stable across representative histories. NameCert obligations, strength monotonicity, gap routing, nonescape, prediction, and independence witness define a visible status transition: every downgrade must expose both the target and the ledger reason that blocks the stronger status.

Theory closure: TheoryGate(NameCert$_{\text{StatusDowngrade}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter binds the status downgrade carrier, NameCert obligations, strength-monotonicity theorem, gap-routing theorem, scoped gap-route theorem, nonescape theorem, falsifiable prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{StatusDowngrade}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim recovery of the stronger status, hidden gap discharge, or status conversion outside the recorded downgrade route.

Upgrade path: Reaching scopedClosure requires formal status-order and gap-route predicates plus checked nonescape of downgraded statuses.

Chapter 1281

EventualTailAgreement NameCert Construction

The $\mathsf{EventualTailAgreement}^{\uparrow}$ packet realizes the shared-tail classifier route for the regular-Cauchy completion surface of section 9238.1. It sits after the finite StreamName windows of chapter 126, the RegSeqRat readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, the paired seal route of chapter 699, and the cofinal agreement route of chapter 920. The packet records a finite shared-tail window before any Real or completion-facing seal is served; it is not a quotient of streams, a host real equality, a selected limiting real, countable choice, or ambient completeness.

Falsifiable prediction: Every accepted $\mathsf{EventualTailAgreement}^{\uparrow}$ packet must expose the shared $\mathsf{StreamName}^{\uparrow}$ tail window before a $\mathsf{RealTailAgreementSeal}^{\uparrow}$ handoff; a consumer route reaching the seal without that window refutes the packet.

Independence witness: streamname, regseqrat, real, realtailagreementseal, cofinaltailagreement: EventualTailAgreementUp is not bijective to any listed sibling because it jointly carries two Real-facing regular-Cauchy sources, one eventual threshold and modulus row, a shared StreamName tail window, paired RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealTailAgreementSeal handoff, and local transport, continuation, package, and naming rows.

Definition 1281.1 (Eventual tail-agreement carrier)**.** *An* $\mathsf{EventualTailAgreement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, B, M, T, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two Real-facing regular-Cauchy source rows whose tails are compared.*
2. *M is the shared eventual threshold and modulus row; it records the point after which both sources are read by the same tolerance request.*
3. *T is the shared* $\mathsf{StreamName}^{\uparrow}$ *tail-window row exposed at that threshold.*
4. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for A and B over the same window T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, endpoint, and comparison rows used by the shared readback.*
6. *S is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff row reached only after A, B, M, T, R, D are displayed.*

7. *H records componentwise* hsame *transport for A, B, M, T, R, D, S and the local certificate row.*
8. *C records displayed* Cont *replay from the two source rows through the shared threshold, tail window, readback, dyadic ledger, and seal handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{EventualTailAgreement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CofinalTailAgreement}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EventualTailAgreement}^{\uparrow}}$ *row.*

The carrier exports only the displayed shared-tail classifier packet and its Real-completion handoff surface.

Theorem 1281.2 (Eventual tail-agreement NameCert obligations)**.** *For the carrier of Definition 1281.1, the local* $\mathsf{NameCert}_{\mathsf{EventualTailAgreement}^{\uparrow}}$ *surface consists exactly of carrier admission for $E = (A, B, M, T, R, D, S, H, C, P, N)$, source admission for A and B, shared threshold and modulus exposure through M, shared* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure through T, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger exactness through D,* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff exposure through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EventualTailAgreementUp.EventualTailAgreementTasteGate single carrier alignment`

Proof. Project the carrier. Its coordinates display the two regular-Cauchy source rows, the shared eventual threshold and modulus row, the tail window, the paired regular readback, the dyadic tolerance ledger, the Real-tail seal row, and the structural support rows.

Read the sibling surfaces in order. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite tail window, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the paired regular-rational readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance ledger, and the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row supplies the downstream seal handoff.

The transport, continuation, package, and local naming rows do not add another analytic object. They authenticate exactly the displayed rows A, B, M, T, R, D, S, so the listed clauses exhaust the local NameCert obligations. □

Theorem 1281.3 (Eventual tail-agreement tail-shift invariance)**.** *Let $E = (A, B, M, T, R, D, S, H, C, P, N)$ be an accepted* $\mathsf{EventualTailAgreement}^{\uparrow}$ *carrier. If the shared eventual threshold in M is shifted to a later displayed threshold and the corresponding* $\mathsf{StreamName}^{\uparrow}$ *tail window is replayed through the same H and C rows, then the accepted classifier surface remains*

$$A, B, M, T, R, D, S, H, C, P, N.$$

The shift changes only the chosen late window inside the same shared-tail route; it preserves the paired $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff, and it exports no quotient stream, host real equality, selected limit, countable-choice row, or ambient completeness theorem.*

Proof. Begin with the threshold coordinate M and the tail-window coordinate T. The carrier stores no independent tail oracle; a later threshold is admitted only by replaying the same source rows through the displayed StreamName route.

Use Definition 1281.2. The paired readback R and tolerance ledger D are attached to the shared tail window, while H and C transport and replay that attachment componentwise.

Finally read the seal coordinate S. Since the seal handoff is downstream of A, B, M, T, R, D, a tail shift that preserves the displayed route preserves the same Real-facing classifier surface and cannot import any excluded external equality or selection principle. □

Theorem 1281.4 (Eventual tail-agreement Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{EventualTailAgreement}^{\uparrow}$ *packet reaches the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row* S *only after the two source rows* A, B*, the shared eventual threshold and modulus* M*, the shared* $\mathsf{StreamName}^{\uparrow}$ *tail window* T*, the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *have been displayed. The consumer route is exhausted by*

$$A, B, M, T, R, D, S, H, C, P, N.$$

No consumer of this packet receives a quotient stream comparison, host real equality, selected host limit, countable-choice schedule, hidden tail oracle, or ambient completeness witness.

Proof. Inspect the carrier inventory. Its only Real-facing coordinate is the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ row S, and the carrier places S after the two source rows, the shared threshold row, the shared window, the paired regular readback, and the dyadic ledger.

Apply Definition 1281.2. The local certificate makes A, B, M, T, R, D mandatory before S can be consumed, with H, C, P, N recording the visible transport, replay, provenance, and naming route.

Apply Definition 1281.3 for later-tail reindexing. Any accepted shift remains on the same shared-tail surface, so a Real or completion-facing consumer cannot bypass the finite window by appealing to quotient streams, selected limits, host equality, countable choice, or ambient completeness. □

The Lean follow-up for this chapter is to provide $\mathsf{Nontrivial\ EventualTailAgreement}^{\uparrow}$ and $\mathsf{FieldFaithful\ EventualTail}$ instances in `BEDC.Derived.EventualTailAgreementUp/TasteGate.lean` for the same finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EventualTailAgreement}^{\uparrow}}$.

Constructive story: An $\mathsf{EventualTailAgreement}^{\uparrow}$ packet is generated from two Real-facing regular-Cauchy source rows, one shared eventual threshold and modulus row, a shared StreamName tail window, paired RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealTailAgreementSeal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EventualTailAgreement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1281.1, Definition 1281.2, Definition 1281.3, and Definition 1281.4 bind the finite shared-tail classifier packet and its Real-completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EventualTailAgreement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, hidden tail selection, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, tail-shift invariance, Real-seal handoff, Nontrivial EventualTailAgreementUp, and FieldFaithful EventualTailAgreementUp for the same finite packet.

Chapter 1282

A Concrete Naming Certificate for $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$

The $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$ packet realizes the formal-target draft of chapter 9462 as a finite package ledger for philosophy-facing machine targets. It sits beside the claim registry layer of chapter 1209, the registry export surface of chapter 1216, the machine-interface formalization target of chapter 1225, the writing-item audit packet of chapter 1235, and the axiom-purity gate of chapter 1259. Those siblings respectively bind claim admission, export rows, target skeletons, writing-item legality, and dependency purity; this carrier binds all five before a philosophy-package target can be read by an audit consumer.

Definition 1282.1 (Philosophy formal-target ledger carrier)**.** *A* $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, T, G, D, S, U, L, M, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the concept-registry row, recording the concept entry, carrier or boundary position, classifier status, ledger status, forbidden readings, and cannot-claim conflicts.*
2. *T is the theorem-map row, recording the theorem or statement site, its claim-entry row, its expected machine namespace, and its dependency surface.*
3. *G is the gap-map row, recording the typed open boundary, refusal branch, or target socket associated with the claim.*
4. *D is the tradition-map row, recording preserved structure, rejected surplus, and tradition-comparison boundary rows.*
5. *S is the scientific-map row, recording empirical record, invariant, bridge, value, and prediction-gap rows when the target touches science-facing prose.*
6. *U is the cannot-claim registry row, recording the exact claim forms refused by the packet.*
7. *L is the closure-status row, recording the paper-side closure grade, formal-verification grade, bridge status, not-claimed surface, and next obligation for the target.*
8. *M is the formal-target skeleton row, recording name, namespace, binder shape, statement outline, dependency list, expected status, and queue status.*

9. *B is the refusal-boundary and no-smuggling row, recording absence of host truth predicates, subject-substance, quotient equality, Classical choice, external philosophy authority, and any hidden solution claim.*
10. *H records componentwise* hsame *transport for $R, T, G, D, S, U, L, M, B$.*
11. *C records displayed* Cont *replay through the registry, theorem, gap, tradition, science, refusal, closure-status, target-skeleton, and no-smuggling rows.*
12. *P is the* Pkg *provenance over* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$, $\mathsf{WritingItemAudit}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host truth predicate, subject-substance, quotient equality, Classical choice, external philosophical authority, proof content for the target skeleton, or export beyond the displayed refusal boundary.

Lean4 (definition) `BEDC.Derived.PhilosophyFormalTargetLedgerUp.PhilosophyFormalTargetLedgerUp`
Lean4 ✓ `BEDC.Derived.PhilosophyFormalTargetLedgerUp.PhilosophyFormalTargetLedgerTasteGate`
`single carrier alignment`

Definition 1282.2 (Philosophy formal-target ledger classifier)**.** *The* $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$ *classifier compares exactly the displayed rows*

$$R, T, G, D, S, U, L, M, B, H, C, P, N.$$

Two packets are classifier-equal only when their concept-registry entries, theorem-map entries, gap-map entries, tradition-map entries, scientific-map entries, cannot-claim rows, closure-status rows, target skeletons, no-smuggling rows, componentwise hsame *routes, displayed* Cont *replays, provenance rows, and local NameCert rows agree. The classifier refuses any comparison that factors through a host truth predicate, subject-substance, quotient equality, Classical choice, or external philosophy authority.*

Theorem 1282.3 (Philosophy formal-target ledger NameCert obligations)**.** *For the carrier of Definition 1282.1, the local* $\mathsf{NameCert}_{\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (R, T, G, D, S, U, L, M, B, H, C, P, N),$$

claim-entry coverage through R and T, gap and tradition coverage through G and D, science-facing coverage through S, refusal coverage through U and B, closure-status coverage through L, target-skeleton coverage through M, componentwise hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite packet of Definition 1282.1. The rows $R, T, G, D, S, U, L, M, B$ are displayed before transport, replay, provenance, and naming are read, so every accepted target exposes its claim entry, target skeleton, no-smuggling check, closure status, and refusal boundary as packet data.

Read the sibling surfaces in order. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ supplies claim-entry discipline, $\mathsf{RegistryExportSurface}^{\uparrow}$ supplies public export status, $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ supplies the machine-readable skeleton, $\mathsf{WritingItemAudit}^{\uparrow}$ supplies separated paper and machine status rows, and $\mathsf{AxiomPurityGate}^{\uparrow}$ supplies the dependency-purity refusal boundary.

Finally inspect H, C, P, N. These structural rows preserve, replay, source, and name the same displayed packet. None of them adds a hidden truth predicate, subject-substance, quotient equality, Classical choice, external philosophical authority, or proof object. Hence the local NameCert obligations are exactly the displayed ledger rows. □

Lean4 ✓ `BEDC.Derived.PhilosophyFormalTargetLedgerUp.PhilosophyFormalTargetLedger namecert obligations`

Theorem 1282.4 (Philosophy formal-target ledger nonescape)**.** *Every machine-readable audit consumer of an accepted* PhilosophyFormalTargetLedger$^{\uparrow}$ *packet reads a claim-entry row, target-skeleton row, no-smuggling row, closure-status row, and refusal-boundary row before it can read the philosophy-package formal target. If any one of these rows is absent, the consumer leaves the accepted carrier and cannot cite the packet as a formal-target ledger.*

Lean4 ✓ `BEDC.Derived.PhilosophyFormalTargetLedgerUp.PhilosophyFormalTargetLedger nonescape`

Proof. Use Definition 1282.3. The obligations expose claim-entry coverage through R, T, target-skeleton coverage through M, no-smuggling and refusal-boundary coverage through B, U, and closure-status coverage through L.

The audit consumer has no alternate route to those data. The continuation row C only replays the displayed path, the transport row H only preserves componentwise rows, and the provenance row P only records the sibling surfaces already named by the packet.

Therefore removing a claim-entry row, target-skeleton row, no-smuggling row, closure-status row, or refusal-boundary row changes the carrier being read. The resulting object may be a partial registry or target record, but it is not an accepted PhilosophyFormalTargetLedger$^{\uparrow}$ packet. □

Theorem 1282.5 (Philosophy formal-target ledger sibling dependency)**.** *The* PhilosophyFormalTargetLedger$^{\uparrow}$ *carrier depends on* ClaimRegistryLayer$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, MachineInterfaceFormalizationTarget$^{\uparrow}$, WritingItemAudit$^{\uparrow}$, *and* AxiomPurityGate$^{\uparrow}$ *without being bijective to any one of them under NameCert-row preservation.*

Proof. Compare the displayed rows. ClaimRegistryLayer$^{\uparrow}$ binds claim entries, RegistryExportSurface$^{\uparrow}$ binds export rows, MachineInterfaceFormalizationTarget$^{\uparrow}$ binds target sockets and statement skeletons, WritingItemAudit$^{\uparrow}$ binds writing-item legality and separated status rows, and AxiomPurityGate$^{\uparrow}$ binds forbidden dependency checks.

The present carrier stores all five surfaces together with closure status, no-smuggling, transport, replay, provenance, and local NameCert rows. A NameCert-row-preserving bijection to any one sibling would have to preserve fields that the sibling does not carry.

Thus the sibling lattice is a dependency lattice rather than an identification. The ledger consumes the siblings as rows of one package-formal-target packet while preserving its own classifier and nonescape boundary. □

Lean4 ✓ `BEDC.Derived.PhilosophyFormalTargetLedgerUp.PhilosophyFormalTargetLedger sibling dependency`

Falsifiable prediction: If a PhilosophyFormalTargetLedgerUp packet is accepted, every exported philosophy-package formal target has a claim-entry row, target-skeleton row, no-smuggling row, closure-status row, and refusal-boundary row before any machine-readable audit consumer can read it; a consumer that reaches a formal target without one of these rows refutes the packet.

Independence witness: claimregistrylayer, registryexportsurface, machineinterfaceformalizationtarget, writingitemaudit, axiom-purity-gate: none is bijective to PhilosophyFormalTargetLedger$^{\uparrow}$

because those siblings respectively bind claim entries, export rows, target sockets, writing-item checks, or axiom-purity gates, while this carrier binds all five with closure status, no-smuggling, transport, provenance, and local NameCert rows in one package-formal-target ledger.

The Lean follow-up file `lean4/BEDC/Derived/PhilosophyFormalTargetLedgerUp/TasteGate.lean` must provide explicit Nontrivial PhilosophyFormalTargetLedger$^{\uparrow}$ and FieldFaithful PhilosophyFormalTargetLedger$^{\uparrow}$ instances. The nontriviality witness must use distinct packet witnesses, and the field-faithfulness witness must use constructor-field injection to preserve $R, T, G, D, S, U, L, M, B, H, C, P, N$ rather than a singleton carrier.

Closure status for NameCert$_{\text{PhilosophyFormalTargetLedger}^{\uparrow}}$.

Constructive story: The philosophy formal-target ledger is generated from $BHist$ rows that record a claim-entry row, target-skeleton row, no-smuggling row, closure-status row, refusal-boundary row, and the classifier that binds a philosophy-facing item to that formal target. The carrier first reads the ledger entry, then checks the target classifier, and finally uses Cont to carry the same target relation into later audit or writing routes. The $hsame$ relation keeps the target ledger stable across equivalent histories. The NameCert obligations, nonescape theorem, sibling-dependency theorem, prediction, and independence witness separate this object from claim registries, export surfaces, machine-interface targets, or axiom-purity gates.

Theory closure: TheoryGate(NameCert$_{\text{PhilosophyFormalTargetLedger}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the carrier, classifier, NameCert obligations, nonescape theorem, sibling dependency, falsifiable prediction, independence witness, and TasteGate marker to the local BHist, hsame, Cont, Pkg, NameCert, target-classifier, and ledger-entry scope.

Formal verification: FormalStatus(NameCert$_{\text{PhilosophyFormalTargetLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim the target is proved, that claims are registered elsewhere, or that status rows are supplied by this ledger.

Upgrade path: No theory-axis grade remains beyond the displayed carrier, classifier, NameCert, nonescape, sibling-dependency, prediction, independence, and TasteGate surface; the Lean verification axis remains independent.

Chapter 1283

A Concrete Naming Certificate for $\mathsf{SocketStackCertificate}^{\uparrow}$

The $\mathsf{SocketStackCertificate}^{\uparrow}$ packet realizes stacked socket reports from subsection 9462.26.13 as finite NameCert data. It sits beside the typed gap socket taxonomy of chapter 1240, the socket-kind classifier of chapter 1058, the socket taxonomy ledger of chapter 767, the supply socket ledger of chapter 528, and the apophatic far-end socket of chapter 1061. Those siblings expose kind families, single-row classifiers, non-collapse accounting, requested-supply ledgers, and the common far-end boundary; this packet binds an ordered multi-socket stack without collapsing the component requests into one dominant kind.

Definition 1283.1 (Socket stack certificate carrier)**.** *A* $\mathsf{SocketStackCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Sigma = (Q, R, E, K, A, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the stack site row, naming the blocked proof, construction, or audit position whose requested supply has more than one socket component;*
2. *$R = (R_0, \ldots, R_n)$ is a nonempty ordered list of component socket-report rows, each carrying its site, requested-supply, narrow kind, and local gate;*
3. *E is the stack-exposure row, recording that every component socket row is visible to audit consumers before any summary row is read;*
4. *$K = (K_0, \ldots, K_n)$ is the componentwise narrowest-kind classifier row, with K_i classifying R_i by the smallest socket kind licensed for that component;*
5. *$A = (A_0, \ldots, A_n)$ is the per-row audit-gate preservation row, requiring each component gate to be replayed at its own socket row;*
6. *L is the non-discharge ledger route, recording that the stack reports requested supply without supplying any component row;*
7. *H stores componentwise* hsame *transport for $Q, R, E, K, A, L, C, P, N$;*
8. *C stores displayed* Cont *continuation from the stack site through the component reports, classifiers, gates, ledger route, provenance, and local name;*
9. *P is the* Pkg *provenance over* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SocketTaxonomyLedger}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{SocketStackCertificate}^{\uparrow}}$ *row.*
The carrier has no coordinate for replacing the ordered component list by a single aggregate socket kind, deleting a component row after acceptance, moving a component gate to a sibling row, or treating the common far-end position as a discharge witness.

Lean4 (definition) `BEDC.Derived.SocketStackCertificateUp.SocketStackCertificateUp`

Theorem 1283.2 (Socket stack certificate NameCert obligations)**.** *For the carrier of Definition 1283.1, the local* $\mathsf{NameCert}_{\mathsf{SocketStackCertificate}^{\uparrow}}$ *surface consists exactly of carrier exposure for*

$$\Sigma = (Q, R, E, K, A, L, H, C, P, N),$$

stack-site exposure through Q, ordered component socket exposure through R, stack exhaustion through E, componentwise narrowest-kind classification through K, per-row audit-gate preservation through A, non-discharge ledger routing through L, componentwise hsame *transport through H, displayed* Cont *continuation through C, provenance through P, and local naming through N.*

Proof. Project the finite packet $\Sigma = (Q, R, E, K, A, L, H, C, P, N)$. The stack site and the ordered component rows are read before classifier, gate, ledger, continuation, provenance, and naming rows, so the certificate cannot replace the stack by an uninspected summary.

Read K and A componentwise. Each K_i names the narrow socket kind for the displayed component R_i, while each A_i preserves the local gate attached to that same row. Componentwise transport in H can replay these rows but cannot move a classifier or gate to a different component.

Finally inspect L, C, P, N. These rows account for the route, replay the same ordered packet, cite the sibling socket ledgers, and name the certificate. None supplies a requested component row, so the non-discharge route remains part of the local NameCert surface. □

Theorem 1283.3 (Socket stack certificate exhaustion)**.** *Every accepted* $\mathsf{SocketStackCertificate}^{\uparrow}$ *packet exposes all component socket rows in its ordered carrier. Removing any component row from an accepted stack changes the componentwise classifier surface or breaks the* NameCert *handoff.*

Proof. Let $\Sigma = (Q, R, E, K, A, L, H, C, P, N)$ be accepted. By Definition 1283.2, the exposure row E is part of the local NameCert surface and is read against the ordered component list $R = (R_0, \ldots, R_n)$.

If a component row R_i is removed, then the matching classifier row K_i and gate row A_i either become unmatched or must also be removed. In the first case, the componentwise classifier surface is no longer a well-formed stack surface. In the second case, the displayed packet named by N is a different finite carrier.

Thus accepted stacks are exhausted by their displayed component rows. A shorter stack may be a separate packet, but it is not the same accepted $\mathsf{SocketStackCertificate}^{\uparrow}$ certificate. □

Theorem 1283.4 (Socket stack certificate sibling independence)**.** *The* $\mathsf{SocketStackCertificate}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SocketTaxonomyLedger}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, *or* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$.

Proof. Compare the displayed rows. $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ carries the seven-kind taxonomy, $\mathsf{SocketKindClassifier}^{\uparrow}$ classifies a socket kind, $\mathsf{SocketTaxonomyLedger}^{\uparrow}$ records taxonomy and non-collapse rows, $\mathsf{SupplySocketLedger}^{\uparrow}$ accounts for requested supply, and $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ names the common far-end boundary.

None of those siblings carries an ordered multi-socket component list with stack exposure, per-row narrowest-kind classifiers, per-row gate preservation, stack exhaustion, and a non-discharge

ledger route. A NameCert-row-preserving bijection would have to preserve that joint ordered surface, so no listed sibling is bijective to $\mathsf{SocketStackCertificate}^{\uparrow}$. □

Falsifiable prediction: Removing any component socket row from an accepted stack changes the componentwise classifier or breaks the NameCert handoff.

Independence witness: typedgapsockettaxonomy, socketkindclassifier, sockettaxonomyledger, supplysocketledger, apophaticfarendsocket: none carries an ordered multi-socket carrier with per-row gate preservation and stack exhaustion.

The Lean follow-up file for BEDC.Derived.SocketStackCertificateUp must provide explicit Nontrivial SocketStackC and FieldFaithful $\mathsf{SocketStackCertificate}^{\uparrow}$ instances. The field-faithfulness witness must preserve $Q, R, E, K, A, L, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out component deletion and single-kind collapse of an accepted stack.

Closure status for $\mathsf{NameCert}_{\mathsf{SocketStackCertificate}^{\uparrow}}$.

Constructive story: The socket stack certificate is generated from $BHist$ rows that record an ordered multi-socket carrier, component socket rows, per-row gate preservation, and an exhaustion witness saying that the stack has been read as a stack. The carrier first names the stack packet, then checks that the stack rows are exhausted by the certificate, and finally allows Cont only to carry the same certified stack forward. The $hsame$ relation preserves the stack certificate across representative histories. NameCert obligations, stack exhaustion, sibling independence, prediction, and independence witness keep this object distinct from typed-gap taxonomy, socket-kind classifier, supply socket ledger, and apophatic far-end sockets.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SocketStackCertificate}^{\uparrow}}$, seed)

Scope closed: This chapter binds the socket stack-certificate carrier, NameCert obligations, stack-exhaustion theorem, sibling-independence theorem, falsifiable prediction, and independence witness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SocketStackCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim report classification, typed supply availability, phase-refusal verdicts, or stack equality outside the certified rows.

Upgrade path: Reaching obligationClosure requires formal stack rows and checked exhaustion plus sibling separation for the stack certificate.

Chapter 1284

A Concrete Naming Certificate for $\mathsf{ObjectConstitutionLedger}^{\uparrow}$

The $\mathsf{ObjectConstitutionLedger}^{\uparrow}$ packet realizes the object-constitution claim of Definition 9462.1948 as finite BEDC data. It sits after the focused history ledger of chapter 1247, the observer-history identity surface of chapter 477, the public inscription event surface of chapter 519, the concept registry surface of chapter 1252, and the scientific-object packet of chapter 1214. The chapter does not add a hidden object substrate to chapter 9462; it records the appearance of an object as focused history, continuation, classifier, ledger, gap, provenance, and local naming rows.

Definition 1284.1 (Object-constitution ledger carrier)**.** *An* $\mathsf{ObjectConstitutionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, I, E, R, S, A, K, L, G, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FocusedLedger}^{\uparrow}$ *row exposing the focused records-side history in which the object pattern is read.*
2. *I is the* $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ *row naming the observer-history identity surface for the same focused read.*
3. *E is the* $\mathsf{InscriptionEvent}^{\uparrow}$ *row recording the public event by which the constituted object is available as record data.*
4. *R is the* $\mathsf{ConceptRegistrySurface}^{\uparrow}$ *row naming the concept entry or registry route under which the object is classified.*
5. *S is the* $\mathsf{ScientificObject}^{\uparrow}$ *row when the constituted object is used by a scientific-object consumer.*
6. *A is the stable-appearance row recording the focused pattern exposed by F, I, E, R before any object name is accepted.*
7. *K is the classifier row that decides which focused appearances count as the same object-facing pattern.*
8. *L is the constitution ledger row recording continuation, classifier stability, registry readback, and inscription provenance.*
9. *G is the gap row listing unclosed appearance, classifier, registry, empirical, and scientific-object commitments.*

10. *H records componentwise* hsame *transport for $F, I, E, R, S, A, K, L, G$ and the local certificate row.*
11. *C records displayed* Cont *replay from focused history through observer identity, inscription, registry, scientific-object read, stable appearance, classifier, ledger, gap, provenance, and local name.*
12. *P is the* Pkg *provenance over* $\mathsf{ObjectConstitutionLedger}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ObjectConstitutionLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare object behind appearances, a private subject-state object, a host-side identity predicate, an unregistered concept, or an unledgered scientific object.

Lean4 (definition) `BEDC.Derived.ObjectConstitutionLedgerUp.ObjectConstitutionLedgerUp`
Lean4 ✓ `BEDC.Derived.ObjectConstitutionLedgerUp.ObjectConstitutionLedgerTasteGate single carrier alignment`

Theorem 1284.2 (Object-constitution ledger NameCert obligations)**.** *For the carrier of Definition 1284.1, the local* $\mathsf{NameCert}_{\mathsf{ObjectConstitutionLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for O, focused-ledger exposure through F, observer-history identity exposure through I, inscription-event exposure through E, concept-registry exposure through R, scientific-object exposure through S, stable-appearance exposure through A, classifier exposure through K, constitution-ledger exposure through L, gap exposure through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed packet. The rows expose the focused records-side history, observer-history identity, public inscription event, registry route, scientific-object route, stable appearance, classifier, ledger, gap, transport, replay, provenance, and local name.

The appearance rows are read first. The focused ledger F supplies the records-side focus, I binds the observer-history identity of that focus, E makes the read public as an inscription event, and R supplies the concept-registry classifier surface.

The object-facing rows are then forced. The stable-appearance row A and classifier row K must be present before the constitution ledger L can name the pattern, and the gap row G remains visible before N is readable. If a scientific-object consumer is present, it reaches the object through S and the same H, C, P, N rows rather than through a separate object substrate. □

Theorem 1284.3 (Object-constitution ledger non-escape)**.** *Every accepted* $\mathsf{ObjectConstitutionLedger}^{\uparrow}$ *object read factors through*

$$F, I, E, R, S, A, K, L, G, H, C, P, N.$$

A consumer that omits the focused ledger, observer-history identity, inscription event, concept-registry surface, stable appearance, classifier, constitution ledger, gap row, provenance, or local NameCert row cannot export the constituted object as a BEDC object name.

Proof. Inspect the carrier inventory of Definition 1284.1. The packet contains exactly the displayed rows $F, I, E, R, S, A, K, L, G, H, C, P, N$ and no row for bare object existence, hidden subject state, host identity, or unregistered concept membership.

Apply Definition 1284.2. The local certificate makes each appearance, classifier, ledger, gap, transport, replay, provenance, and name row mandatory for an accepted object-constitution read.

The sibling packets supply only their respective surfaces. $\mathsf{FocusedLedger}^{\uparrow}$ does not by itself name an object, $\mathsf{ObserverHistoryIdentity}^{\uparrow}$ does not by itself certify a concept, $\mathsf{InscriptionEvent}^{\uparrow}$ does not by itself stabilize an appearance, $\mathsf{ConceptRegistrySurface}^{\uparrow}$ does not by itself provide empirical object use, and $\mathsf{ScientificObject}^{\uparrow}$ does not erase the focused-appearance ledger. Therefore a consumer that drops any mandatory row reads a projection of the packet, not the constituted object name. □

Falsifiable prediction: Any accepted ObjectConstitutionLedgerUp consumer that exports a stable object name while omitting the focused ledger, observer-history identity, inscription event, concept registry, stable-appearance classifier, gap row, or local NameCert row refutes the packet.

Independence witness: focusedledger, observerhistoryidentity, inscriptionevent, conceptregistrysurface, scientificobject: none is bijective because $\mathsf{ObjectConstitutionLedger}^{\uparrow}$ binds focused history, observer-history identity, public inscription, concept registry, scientific-object use, stable appearance, classifier, constitution ledger, gap, transport, replay, provenance, and local NameCert rows in one carrier, while each sibling supplies only one constituent surface.

The Lean follow-up namespace `BEDC.Derived.ObjectConstitutionLedgerUp.TasteGate` must provide explicit $\mathsf{Nontrivial\ ObjectConstitutionLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObjectConstitutionLedger}^{\uparrow}$ instances. These instances must preserve the rows $F, I, E, R, S, A, K, L, G, H, C, P, N$ as the same finite carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ObjectConstitutionLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{ObjectConstitutionLedger}^{\uparrow}$ packet is generated from focused history, observer-history identity, inscription event, concept registry, scientific-object use, stable appearance, classifier, constitution ledger, gap, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObjectConstitutionLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1284.1, Definition 1284.2, and Definition 1284.3 bind object constitution to a finite NameCert carrier and non-escape surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObjectConstitutionLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, bare object existence, hidden subject-state identity, unregistered concept membership, or final scientific truth.

Upgrade path: Obligation closure requires checked carrier admission, focused-ledger exposure, observer-history identity exposure, inscription-event exposure, concept-registry exposure, scientific-object exposure, gap-report separation, and TasteGate nontriviality and field faithfulness for the same packet.

Chapter 1285

A Concrete Naming Certificate for $\mathsf{PhysicalLawBridge}^{\uparrow}$

The $\mathsf{PhysicalLawBridge}^{\uparrow}$ packet realizes the science-bridge site of section 9462.28 as a finite BEDC NameCert carrier for physical laws. It sits after the law-certificate packet of chapter 1155, the empirical-regularity persistence packet of chapter 1156, the scientific-object carrier of chapter 1214, the standard-bridge audit packet of chapter 1217, and the open physical fit packet of chapter 1131. Those siblings expose law status, empirical persistence, object anchoring, bridge theorem preservation, and open finite fit; this carrier binds them before a consumer may read a physical law as bridge-recognized.

Definition 1285.1 (Physical law bridge carrier)**.** *A* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, E, B, O, F, G, H, C, Q, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LawCertificate}^{\uparrow}$ *row exposing equation pattern, source fit, classifier, continuation stability, failure surface, ledger, transport, provenance, and local law naming.*
2. *E is the* $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ *row exposing finite measurement, observed regularity, classifier, error ledger, future-persistence GAP, LawCertificate consumption, stability, failure, replay, provenance, and local naming.*
3. *B is the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *row exposing host bridge encoding, decoding, paired round trips, finite theorem-preservation, no-host-leak, audit transport, provenance, and local bridge naming.*
4. *O is the* $\mathsf{ScientificObject}^{\uparrow}$ *row exposing record, classifier, invariant, empirical ledger, bridge, gap report, audit, truth, open-fit, domain, transport, provenance, and local object naming.*
5. *F is the* $\mathrm{OpenPhysicalFit}^{\uparrow}$ *row exposing finite trace, observed signature, model-generated signature, classifier, refutation row, open-continuation row, ledger, replay, and local fit naming.*
6. *G is the failure and falsification row recording omitted-law rows, finite refuting observations, bridge-preservation failures, open-fit exits, and the precise refusal branch.*
7. *H records componentwise* hsame *transport for $L, E, B, O, F, G, C, Q, N$.*
8. *C records the displayed* Cont *routes by which the law, empirical persistence, bridge audit, scientific object, open fit, failure, transport, provenance, and local name are replayed.*

9. *Q is the* Pkg *provenance over* $\mathsf{PhysicalLawBridge}^{\uparrow}$, $\mathsf{LawCertificate}^{\uparrow}$, $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, $\mathrm{OpenPhysicalFit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$ *row.*

The classifier compares packets only through these rows. It has no coordinate for bare equation authority, record-only empirical adequacy, unchecked host-model bridges, hidden theorem preservation, final physical truth, or a law read without an exposed falsification surface.

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.TasteGate.PhysicalLawBridgeTasteGate single carrier alignment`

↪ variant `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeTasteGate single carrier alignment`

Lean4 (definition) `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeUp`

Theorem 1285.2 (Physical law bridge NameCert surface)**.** *For the carrier of Definition 1285.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $P = (L, E, B, O, F, G, H, C, Q, N$ law-certificate exposure through L, empirical-persistence exposure through E, standard-bridge-audit exposure through B, scientific-object exposure through O, open-physical-fit exposure through F, failure and falsification exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeCarrier namecert surface`

Proof. Project the accepted packet to its displayed rows. The row L supplies law-certificate status, E supplies finite empirical persistence and its future-persistence GAP, B supplies finite bridge theorem-preservation with no-host-leak discipline, O supplies the scientific-object anchoring surface, and F supplies open physical fit.

Read the failure surface next. The row G is not a summary sentence; it is the carrier coordinate where omitted empirical persistence, missing theorem-preservation, fit defeat, and falsification branches are recorded.

Finally apply componentwise transport, continuation replay, provenance, and local naming. Since H, C, Q, N preserve only the displayed rows, the listed surface exhausts the local NameCert obligations and leaves no hidden bridge-recognition coordinate. □

Theorem 1285.3 (Physical law bridge law-gap nonescape)**.** *Every accepted* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *read of a physical law factors through the law-certificate row L, empirical-persistence row E, standard-bridge-audit row B, scientific-object row O, open-fit row F, and failure row G. A consumer that keeps the law-certificate row but omits empirical persistence, finite theorem-preservation, open physical fit, or falsification data reads only a projection of the packet, not an accepted bridge-recognized physical law.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeLawGapNonescape`

Proof. Start from Definition 1285.2. The accepted local name is available only after $L, E, B, O, F, G, H, C, Q, N$ have been displayed.

The law-certificate row records equation-like status under continuation and failure discipline, but it does not by itself provide finite measurement persistence or bridge theorem preservation. The empirical-persistence row supplies finite records and future-persistence GAP exposure, while the standard-bridge-audit row supplies the bridge rows that keep theorem transport internal to the packet.

The scientific-object and open-fit rows prevent the law from being detached from object anchoring and finite observation defeat. The failure row then records the refusal branch whenever one of those surfaces is missing. Thus no accepted consumer can escape the law gap by reading L alone or by replacing E, B, F, G with outside authority. □

Theorem 1285.4 (Physical law bridge falsification boundary)**.** *An accepted* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *packet admits falsification exactly through the displayed failure row G together with the empirical-persistence row E, standard-bridge-audit row B, and open-physical-fit row F. A finite refuting observation, a missing persistence ledger, a failed theorem-preservation row, or an open-fit classifier exit changes the packet only as a displayed failure branch; it cannot be hidden inside law status, object status, bridge status, or provenance.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeFalsificationBoundary encode append`

Proof. Use the packet coordinates of Definition 1285.1. The rows E, B, F expose the empirical, bridge, and finite-fit sites where a physical-law read can fail.

The row G names the corresponding failure branch. It records omitted rows, finite refutations, theorem-preservation failures, open-fit exits, and the refusal route in the same carrier.

Transport, replay, provenance, and naming do not erase those branches. They only replay the displayed packet. Therefore falsification remains a finite row inside $\mathsf{PhysicalLawBridge}^{\uparrow}$ rather than a host verdict outside the carrier. □

Theorem 1285.5 (Physical law bridge obligation package)**.** *For every accepted* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *carrier $P = (L, E, B, O, F, G, H, C, Q, N)$, the obligation package is exactly the displayed collection of carrier admission, law-certificate exposure, empirical-persistence exposure, bridge-audit exposure, scientific-object anchoring, open-fit exposure, falsification exposure, componentwise transport, replay, provenance, and local naming rows. A consumer may project one of these rows only as a projection of the package, not as a smaller physical-law bridge certificate.*

Proof. Project the carrier of Definition 1285.1. The rows L, E, B, O, F, G provide the law, empirical, bridge, object, fit, and falsification obligations.

The structural rows H, C, Q, N preserve and replay those obligations without introducing a hidden shortcut. Thus the accepted certificate is the full displayed package, and any smaller read is only a projection. □

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeObligationPackage`

Theorem 1285.6 (Physical law bridge scoped export)**.** *Every accepted* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *export factors through the displayed carrier rows*

$$L, E, B, O, F, G, H, C, Q, N.$$

The rows L, E, B, O, F are respectively the $\mathsf{LawCertificate}^{\uparrow}$ *route of chapter 1155, the* $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$ *route of chapter 1156, the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *route of chapter 1217, the* $\mathsf{ScientificObject}^{\uparrow}$ *route of chapter 1214, and the* $\mathrm{OpenPhysicalFit}^{\uparrow}$ *route of chapter 1131. The row G is the falsification branch, and H, C, Q, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. No export that omits one of these rows is an accepted physical-law bridge export.*

Proof. Project the obligation package of Definition 1285.5. It already lists carrier admission, law-certificate exposure, empirical-persistence exposure, bridge-audit exposure, scientific-object anchoring, open-fit exposure, falsification exposure, transport, replay, provenance, and local naming as the displayed package.

Read the five upstream science-bridge siblings through the introduction's carrier order. The law-certificate row L supplies law status, E supplies empirical persistence, B supplies standard bridge audit, O supplies scientific-object anchoring, and F supplies open physical fit.

Now retain the falsification and structural rows. The row G records the refusal or falsification branch, while H, C, Q, N transport, replay, source, and name the same packet. Removing any one of these rows leaves only a projection because the local $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$ is issued for the complete displayed carrier.

Thus every accepted export is scoped by the carrier rows themselves. The theorem supplies no additional physical truth predicate, no final model authority, and no bridge-recognition shortcut outside the sibling rows. □

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeScopeExport`

Theorem 1285.7 (Physical law bridge field-faithful obligation)**.** *No accepted* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *read can erase any of the rows* $L, E, B, O, F, G, H, C, Q, N$*. If two accepted reads agree after deleting one of these coordinates, the agreement is only a projection-level agreement and is not an accepted field-faithful bridge read.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeFieldFaithfulObligation`

Proof. Use Definition 1285.5. Each row is named as part of the obligation package and is transported by H and replayed by C before provenance and local naming are read.

Deleting a row removes either a semantic obligation among L, E, B, O, F, G or a structural obligation among H, C, Q, N. The resulting projection cannot be the same accepted local name, because the local $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$ is issued only for the displayed packet. □

Theorem 1285.8 (Physical law bridge nontrivial obligation)**.** *The law-recognized branch and the refusal or falsification branch of a* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *packet remain distinguishable. In particular, a carrier that exposes* L, E, B, O, F *with no active failure row is not identified with a carrier whose row* G *records omitted empirical persistence, missing theorem preservation, open-fit exit, or finite refutation.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeNontrivialObligation`

Proof. Read the falsification boundary of Definition 1285.4. The row G is the finite coordinate where omitted rows, refutations, preservation failures, and open-fit exits are recorded.

The law-recognized branch depends on the displayed law, empirical, bridge, object, and fit rows without such a failure record. Since transport and replay preserve G rather than erasing it, the two branches remain distinct inside the packet. □

Falsifiable prediction: A PhysicalLawBridgeUp consumer that reads a law as bridge-recognized while omitting either empirical persistence, finite theorem-preservation, or falsification rows refutes the packet.

Independence witness: lawcertificate, empiricalregularitypersistence, scientificobject, standardbridgeauditpacket, openphysicalfit: non-bijection follows by row omission. $\mathsf{LawCertificate}^{\uparrow}$ lacks empirical-persistence, bridge-audit, scientific-object, and open-fit rows; $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$

lacks bridge theorem-preservation and object anchoring; $\mathsf{ScientificObject}^{\uparrow}$ lacks law-certificate and theorem-preservation rows; $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ lacks empirical persistence and open physical fit; $\mathrm{OpenPhysicalFit}^{\uparrow}$ lacks law-certificate, empirical-persistence, and bridge-audit rows.

The Lean follow-up path `lean4/BEDC/Derived/PhysicalLawBridgeUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{PhysicalLawBridge}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PhysicalLawBridge}^{\uparrow}$ instances. These instances must preserve law-certificate, empirical-persistence, bridge-audit, scientific-object, open-fit, failure, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before `BEDC.Derived.PhysicalLa` can be cited by a checked paper marker.

Theorem 1285.9 (Physical law bridge scoped TasteGate route)**.** *Every accepted scoped export of* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *factors through the carrier rows*

$$L, E, B, O, F, G, H, C, Q, N$$

of Definition 1285.1, where H is the componentwise hsame *transport row, C is the displayed* Cont *replay row, Q is the* Pkg *provenance row, and N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$ *row. The scoped export is accepted only along the TasteGate single-carrier alignment marker* `BEDC.Derived.PhysicalLawBri` *it may not bypass the law-certificate, empirical-persistence, bridge-audit, scientific-object, open-fit, falsification, transport, replay, provenance, or local naming rows.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeScopedTasteGateRoute`

Proof. Apply Definition 1285.6. It already forces every accepted export through $L, E, B, O, F, G, H, C, Q, N$, with H, C, Q, N respectively carrying the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Use Definition 1285.7 to keep the read field-faithful. Deleting any one of the displayed coordinates leaves only a projection-level agreement, so a scoped export cannot erase a carrier row while retaining the accepted local name.

Finally apply Definition 1285.8. The law-recognized branch and the refusal or falsification branch remain distinct, which is exactly the branch discipline witnessed by the cited TasteGate single-carrier alignment marker. Therefore the scoped export is accepted only through the displayed BEDC rows and the TasteGate route, not through outside physical authority or host-model equality. □

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgeCarrier scoped tastegate route`

Theorem 1285.10 (Physical law bridge public closure route)**.** *The public* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *read is available only after the carrier of Definition 1285.1, the law-gap nonescape theorem Definition 1285.3, the falsification boundary Definition 1285.4, the field-faithful obligation Definition 1285.7, the nontrivial obligation Definition 1285.8, and the scoped TasteGate route Definition 1285.9 have all been displayed.*

Thus a public consumer reads the same rows $L, E, B, O, F, G, H, C, Q, N$ and cannot treat $\mathsf{PhysicalLawBridge}^{\uparrow}$ *as bare physical truth, host-model identity, unchecked bridge authority, or law status without empirical persistence, theorem preservation, open fit, and falsification rows.*

Lean4 ✓ `BEDC.Derived.PhysicalLawBridgeUp.PhysicalLawBridgePublicClosureRoute`

Proof. Start with the carrier and NameCert surface. The carrier of Definition 1285.1 supplies exactly the law-certificate, empirical-persistence, bridge-audit, scientific-object, open-fit, falsification, transport, replay, provenance, and local naming rows, and Definition 1285.2 binds those rows as the local certificate surface.

Read the public prerequisites in order. The law-gap nonescape theorem keeps L attached to E, B, O, F, G, while the falsification boundary keeps failure and refutation data visible instead of hiding them in law or bridge status.

Now apply the field-faithful and nontrivial obligations. They prevent deletion of a row and keep law-recognized and refusal branches distinguishable inside the same packet.

Finally use the scoped TasteGate route. Since the scoped export already passes through $L, E, B, O, F, G, H, C, Q, N$ and the TasteGate single-carrier alignment marker, the public read is available only as that displayed route, with no additional host-model authority or physical-truth coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalLawBridge}^{\uparrow}$ packet is generated from law-certificate, empirical-persistence, standard-bridge-audit, scientific-object, open-physical-fit, failure, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1285.1, Definition 1285.2, Definition 1285.3, Definition 1285.4, Definition 1285.5, Definition 1285.7, Definition 1285.8, Definition 1285.9, and Definition 1285.10 bind the finite physical-law bridge carrier, NameCert surface, law-gap boundary, falsification boundary, obligation package, field-faithful read, nontrivial branch distinction, scoped TasteGate route, and public read.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalLawBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, final physical truth, total future prediction, host-model law identity, or law status without empirical persistence, theorem-preservation, open-fit, and falsification rows.

Upgrade path: Further closure requires checked carrier admission, NameCert surface, law-gap nonescape, falsification boundary, scoped TasteGate routing, nontriviality, and field faithfulness for $L, E, B, O, F, G, H, C, Q, N$.

Chapter 1286

A Concrete Naming Certificate for $\mathsf{ScienceBridge}^{\uparrow}$

The $\mathsf{ScienceBridge}^{\uparrow}$ packet realizes the science-bridge site of section 9462.28 as one finite NameCert surface. It sits after the scientific-object carrier of chapter 1214, the physical-model audit of chapter 1129, the reality-constrained truth certificate of chapter 1144, the standard-bridge audit packet of chapter 1217, and the gap/failure bridge audit of chapter 1229. Those siblings expose object records, physical audit, truth rows, bridge preservation, and typed gap/failure rows; this packet binds them before any consumer reads a scientific bridge as an accepted BEDC object.

Definition 1286.1 (Science bridge carrier)**.** *A* $\mathsf{ScienceBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, O, A, T, B, G, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the records row, storing the source histories, observation traces, instrument records, and finite ledgers from which the bridge is read.*
2. *O is the* $\mathsf{ScientificObject}^{\uparrow}$ *row exposing the object record, classifier, invariant, empirical ledger, bridge row, gap report, audit, truth, open-fit, domain, transport, replay, provenance, and local naming surfaces.*
3. *A is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row exposing source traces, target phenomenon, observed signatures, model signatures, audit classifier, tower data, failure surface, and audit strength.*
4. *T is the* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *row exposing the truth-certificate source, signature, classifier, tower, stability, descent, invariant, ledger, and failure fields.*
5. *B is the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *row exposing encoding, decoding, round-trip, theorem preservation, no-host-leak, transport, provenance, and local naming data for a proposed bridge.*
6. *G is the* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *row exposing the typed gap, failure branch, bridge row, no-host-leak row, axis-separation row, audit route, transport, replay, provenance, and local naming data.*
7. *F is the failure-surface row, recording falsifying observations, open prediction gaps, bridge defeats, and domain exits that remain visible to downstream consumers.*
8. *H records componentwise* hsame *transport for $R, O, A, T, B, G, F, C, P, N$.*

9. *C records the displayed* Cont *routes by which records, objects, audits, truth certificates, bridge audits, gaps, failure surfaces, transports, provenance, and local naming are replayed.*

10. *P is the* Pkg *provenance over* $\mathsf{ScienceBridge}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{RealityConstrainedTruthCert}$ $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ScienceBridge}^{\uparrow}}$ *row.*

The classifier compares carriers only through these rows. It has no coordinate for host scientific-object identity, unledgered physical truth, standard-model authority outside the bridge audit, a bridge-checked export without a gap/falsification row, or a repair of a failed bridge by local naming alone.

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.ScienceBridgeTasteGate single carrier alignment`

Theorem 1286.2 (Science bridge NameCert obligations)**.** *For the carrier of Definition 1286.1, the local* $\mathsf{NameCert}_{\mathsf{ScienceBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (R, O, A, T, B, G, F, H, C, P, N)$, records exposure through R, scientific-object exposure through O, physical-model audit exposure through A, reality-constrained truth exposure through T, standard-bridge audit exposure through B, gap/falsification exposure through G, failure-surface exposure through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.TasteGate.ScienceBridgeTasteGate single carrier alignment`

Proof. Project the accepted packet to its displayed rows. The records row R supplies the finite source data, while O, A, T, B, G, F supply the scientific-object, physical-audit, truth-certificate, standard-bridge-audit, gap/falsification, and failure-surface coordinates.

Read the transport and replay rows next. The componentwise hsame row H preserves only the displayed coordinates, and the continuation row C replays only the routes listed by the packet.

Finally apply provenance and local naming. The package row P records the sibling surfaces used by the carrier, and N names the resulting finite packet. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1286.3 (Science bridge axis separation)**.** *An accepted* $\mathsf{ScienceBridge}^{\uparrow}$ *carrier cannot read scientific-object maturity, physical-model audit strength, reality-constrained truth, standard-bridge preservation, and gap/falsification visibility as one axis. The rows O, A, T, B, G, F remain distinct under the same R, H, C, P, N support, so a consumer may not promote a bridge row to physical truth, erase the failure surface, or treat the gap/falsification row as discharged by local naming.*

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.ScienceBridgeAxisSeparation`

Proof. Start from the carrier of Definition 1286.1. The rows O, A, T, B, G, F are separate coordinates, and each is transported and replayed through H and C without being identified with another coordinate.

Use Definition 1286.2. The local certificate admits scientific-object, physical-audit, truth-certificate, standard-bridge-audit, gap/falsification, and failure-surface rows separately before provenance or naming can be consumed.

Thus a bridge-preservation read cannot become physical truth, and a local name cannot erase an open gap or failure surface. Any such reading would collapse coordinates that the accepted packet stores separately. □

Theorem 1286.4 (Science bridge kernel-scope grounding)**.** *Every accepted* $\mathsf{ScienceBridge}^{\uparrow}$ *carrier* $S = (R, O, A, T, B, G, F, H, C, P, N)$ *is scoped only through finite* BHist *records, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *route. The rows* R, O, A, T, B, G, F *may be consumed as records, scientific-object surface, physical-model audit, reality-constrained truth certificate, standard-bridge audit, gap/falsification surface, and failure surface inside that finite route, but none of them can be read as external physical truth, host scientific-object identity, complete empirical adequacy, or bridge closure.*

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.ScienceBridgeKernelScopeGrounding`

Proof. Project the carrier of Definition 1286.1. The rows R, O, A, T, B, G, F are finite coordinates of one BHist packet, and their only structural support is the displayed transport, replay, provenance, and local naming rows H, C, P, N.

Apply the NameCert obligations, Definition 1286.2. They admit each row separately through the finite carrier surface and do not add an external truth coordinate, a host identity coordinate, or a bridge-checked export.

Use axis separation, Definition 1286.3. It prevents the scientific-object, physical-audit, truth-certificate, bridge-audit, gap/falsification, and failure rows from collapsing into a single authority axis, so no row can promote another row into physical truth or bridge closure.

The falsifiable prediction and independence witness give the same scope test. Omitting the gap/falsification or bridge-audit row refutes the carrier, and the packet is not bijective to any sibling surface; hence the accepted read remains grounded in the listed BEDC kernel primitives and nowhere else. □

Falsifiable prediction: Any accepted ScienceBridgeUp packet that exports a bridge-checked scientific object while omitting the gap/falsification row or the standard-bridge audit row refutes the carrier.

Independence witness: scientificobject, physicalmodelaudit, realityconstrainedtruthcert, standardbridgeauditpacket, gapfailurebridgeaudit: none is bijective because ScienceBridgeUp binds records, object surface, audit row, truth row, bridge row, gap/falsification row, failure surface, transport, replay, provenance, and local NameCert in one carrier.

The Lean follow-up path `lean4/BEDC/Derived/ScienceBridgeUp/TasteGate.lean` must provide $\mathsf{Nontrivial}\ \mathsf{ScienceBridge}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ScienceBridge}^{\uparrow}$ instances for the same finite $R, O, A, T, B, G, F, H, C,$ carrier.

Theorem 1286.5 (Science bridge standard-audit handoff)**.** *Every accepted bridge-facing consumer of a* $\mathsf{ScienceBridge}^{\uparrow}$ *packet must pass through the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *surface, the* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *surface, the* $\mathsf{RealityConstrainedTruthCert}^{\uparrow}$ *surface, and the finite* BHist, hsame, Cont, Pkg, *and* NameCert *rows before any bridge read is public. Consequently a consumer may not publish the bridge row while bypassing the standard audit route, erasing the gap or failure branch, replacing the truth certificate by host physical truth, or reading outside the displayed records and local naming rows.*

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.ScienceBridgeStandardAuditHandoff`

Proof. Project the carrier of Definition 1286.1. The bridge-facing rows are already displayed as B, G, T together with the records, failure, transport, replay, provenance, and naming rows.

Apply Definition 1286.2. It requires standard-bridge audit exposure through B, gap/falsification exposure through G, reality-constrained truth exposure through T, and finite structural support through R, H, C, P, N before the local certificate can be consumed.

Use axis separation, Definition 1286.3. It prevents bridge preservation, gap/failure visibility, and truth certification from collapsing into one authority row, so the public bridge read remains routed through the standard audit handoff. □

Theorem 1286.6 (Science bridge mature failure-surface export)**.** *Every mature* $\mathsf{ScienceBridge}^{\uparrow}$ *consumer must preserve the records row, scientific-object row, physical-model-audit row, truth-certificate row, standard-bridge-audit row, gap/falsification row, and failure-surface row through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route. In particular, a mature read may not publish the bridge surface while dropping* $R, O, A, T, B, G, F, H, C, P, N$ *or replacing the gap/falsification and failure-surface rows by external scientific authority.*

Lean4 ✓ `BEDC.Derived.ScienceBridgeUp.ScienceBridgeMatureFailureSurfaceExport`

Proof. Project the carrier of Definition 1286.1. The rows R, O, A, T, B, G, F are distinct coordinates of the accepted packet, and H, C, P, N are the only structural rows that transport, replay, source, and name them.

Apply the NameCert obligations and axis-separation results. They require records, scientific-object, physical-model audit, truth certificate, standard-bridge audit, gap/falsification, and failure-surface exposure before any bridge read can be consumed, and they keep those rows from collapsing into one authority axis.

Use the standard-audit handoff. Since the handoff already requires the standard-bridge audit, gap/falsification, truth-certificate, records, transport, replay, provenance, and local naming rows, a mature consumer can export the science bridge only by preserving the whole displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScienceBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{ScienceBridge}^{\uparrow}$ packet is generated from records, scientific-object, physical-model-audit, reality-constrained-truth, standard-bridge-audit, gap/falsification, failure-surface, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ScienceBridge}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1286.4 is the public export surface for the finite carrier, obligation, axis-separation, BHist, hsame, Cont, Pkg, and NameCert scope of the science-bridge packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ScienceBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, external physical truth, host scientific-object identity, complete empirical adequacy, or final model–reality correspondence.

Upgrade path: Bridge closure requires a standard-bridge audit whose bridge row preserves the public science-bridge surface without erasing the gap/falsification or failure rows.

Chapter 1287

A Concrete Naming Certificate for $\mathsf{ConceptRegistryObligation}^{\uparrow}$

The $\mathsf{ConceptRegistryObligation}^{\uparrow}$ packet realizes the concept-entry obligation of Definition 9462.2157 and Definition 9462.2161 as finite BEDC data. It sits beside chapter 1252, chapter 1216, chapter 1251, chapter 1220, and chapter 901. Those siblings expose the concept-registry surface, public export boundary, core index, package registry layer, and failed-gate diagnostic surface; this carrier isolates the finite row packet a concept entry must carry before it can be exported as philosophy-package vocabulary.

Definition 1287.1 (Concept registry obligation carrier)**.** *A* $\mathsf{ConceptRegistryObligation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (E, A, L, F, S, X, H, C, P, N)$$

with the following displayed rows:

1. *E is the concept-entry row, recording the name named by the philosophy package and the entry surface required by Definition 9462.2157;*
2. *A is the carrier-and-classifier row, recording the displayed BEDC carrier and classifier to which the concept entry is attached;*
3. *L is the ledger row, recording the positive reading, supporting ledger, and package route that a consumer may inspect;*
4. *F is the forbidden-reading row, recording the readings blocked before the entry can be exported;*
5. *S is the status row, recording gap status, theory closure, formal status, and export readiness as finite registry data;*
6. *X is the export row, recording whether the entry is read by the concept-registry surface, registry-export surface, core registry index, or registry layer;*
7. *H stores componentwise* hsame *transport for concept-entry, carrier-and-classifier, ledger, forbidden-reading, status, export, route, provenance, and local naming rows;*
8. *C stores displayed* Cont *routes from concept entry through carrier, classifier, ledger, forbidden-reading, status, and export rows;*
9. *P is the* Pkg *provenance over* $\mathsf{ConceptRegistryObligation}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{AuditGateFailureWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{ConceptRegistryObligation}^{\uparrow}}$ *row.*
The carrier has no coordinate for a prose-only concept, missing carrier, missing classifier, erased ledger, absent forbidden-reading row, hidden status row, export bypass, host semantic equality, or package-private override.

Lean4 ✓ `BEDC.Derived.ConceptRegistryObligationUp.ConceptRegistryObligationTasteGate single carrier alignment`

Theorem 1287.2 (Concept registry obligation NameCert obligations)**.** *For the carrier of Definition 1287.1, the local* $\mathsf{NameCert}_{\mathsf{ConceptRegistryObligation}^{\uparrow}}$ *surface consists exactly of carrier admission for $O = (E, A, L, F, S, X, H, C, P, N)$; concept-entry exposure through E; carrier and classifier exposure through A; ledger accountability through L; forbidden-reading accountability through F; status exposure through S; export routing through X; transport and route replay through H, C; provenance through P; and local naming through N.*

Proof. Project the finite packet of Definition 1287.1. The row E names the entry required by the philosophy concept registry, and A, L, F, S display the carrier, classifier, ledger, forbidden readings, and status fields that Definition 9462.2157 requires.

The export row X binds the same entry to the sibling registry surfaces. The concept-registry surface reads the entry discipline, the registry-export surface reads public export control, the core registry index reads the table location, and the registry layer reads package-level admit-or-block routing.

Finally H, C, P, N transport, replay, package, and name the same rows. Since the carrier has no hidden row from which a missing carrier, classifier, ledger, forbidden-reading, or status field can be reconstructed, the displayed rows exhaust the local NameCert surface. □

Theorem 1287.3 (Concept registry obligation export readiness)**.** *An accepted* $\mathsf{ConceptRegistryObligation}^{\uparrow}$ *packet is export-ready only when the concept-entry row E, carrier-and-classifier row A, ledger row L, forbidden-reading row F, status row S, export row X, transport row H, route row C, provenance row P, and local naming row N are all present in the same finite packet. A concept export missing any one of carrier, classifier, ledger, forbidden-reading, or status data exits the accepted carrier.*

Proof. Use Definition 1287.2. It lists each accepted row as part of one local NameCert surface, so export readiness cannot be checked by the entry name alone.

If A is absent, the concept has no displayed BEDC carrier and classifier. If L is absent, the positive reading and supporting ledger are not available to the consumer. If F is absent, blocked readings can be smuggled into the export. If S is absent, gap, closure, and formal status cannot be inspected.

The row X routes the admitted packet to the registry surfaces, while H, C, P, N preserve and name the route. None of these support rows synthesizes a missing obligation row, so absence of any listed field exits the accepted $\mathsf{ConceptRegistryObligation}^{\uparrow}$ carrier. □

Theorem 1287.4 (Concept registry obligation forbidden-reading boundary)**.** *Every public reading of an accepted* $\mathsf{ConceptRegistryObligation}^{\uparrow}$ *packet must carry the forbidden-reading row F together with the carrier-and-classifier row A, ledger row L, status row S, and export row X. The packet cannot export subject substance, hidden theorem content, host semantic equality, audit-gate failure as theorem proof, or a registry entry whose blocked readings are not displayed.*

Proof. Begin with export readiness, Definition 1287.3. It keeps F in the same finite packet as the carrier, classifier, ledger, status, and export rows.

The audit-gate failure witness is diagnostic. It can record a failed gate and route that failure to the registry boundary, but it does not convert a missing forbidden-reading row into theorem content or export permission.

The concept-registry surface and registry-export surface then read the same displayed boundary. Since neither sibling surface contains a coordinate for hidden host semantics, subject substance, or blocked readings erased from the packet, the forbidden-reading boundary is exact for accepted concept export. □

Theorem 1287.5 (Concept registry obligation sibling dependency)**.** *The* $\mathsf{ConceptRegistryObligation}^{\uparrow}$ *packet depends on the sibling rows* $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{CoreRegistryIndex}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, *and* $\mathsf{AuditGateFailureWitness}^{\uparrow}$ *only through displayed route and provenance data. None of those siblings is bijective to* $\mathsf{ConceptRegistryObligation}^{\uparrow}$*: each lacks at least one of the simultaneous concept-entry, carrier-and-classifier, ledger, forbidden-reading, status, export, transport, route, provenance, and local NameCert rows.*

Proof. Read the provenance row P and route row C. They name the sibling surfaces as consumers or diagnostic neighbors of the obligation packet, not as replacements for the local carrier.

The concept-registry surface records concept-family export, but it does not isolate the obligation packet required for each concept entry. The registry-export surface records public export control, but not the concept-entry carrier and classifier obligations together. The core registry index records table position, while the registry layer records package admit-or-block routing. The audit-gate failure witness records diagnostic failure rather than export readiness.

Thus the dependency is one-way through displayed route and provenance rows. A sibling can consume or diagnose the packet, but none reconstructs the entire finite carrier $O = (E, A, L, F, S, X, H, C, P, N)$. □

Falsifiable prediction: A concept export missing carrier, classifier, ledger, forbidden-reading, or status rows fails the ConceptRegistryObligationUp carrier.

Independence witness: conceptregistrysurface, registryexportsurface, coreregistryindex, registrylayer, auditgatefailurewitness: none is bijective because ConceptRegistryObligationUp carries concept-entry, carrier-and-classifier, ledger, forbidden-reading, status, export, transport, route, provenance, and local NameCert rows simultaneously.

The Lean follow-up for this chapter must provide `Nontrivial ConceptRegistryObligationUp` and `FieldFaithful ConceptRegistryObligationUp` instances in `BEDC.Derived.ConceptRegistryObligationU` These instances must preserve $E, A, L, F, S, X, H, C, P, N$ as separate displayed rows before any checked theorem cites the packet as a TasteGate surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ConceptRegistryObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{ConceptRegistryObligation}^{\uparrow}$ packet is generated from concept-entry, carrier-and-classifier, ledger, forbidden-reading, status, export, componentwise hsame transport, displayed Cont route, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConceptRegistryObligation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1287.1, Definition 1287.2, Definition 1287.3, Definition 1287.4, and Definition 1287.5 bind philosophy concept-entry obligations to a finite registry packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConceptRegistryObligation}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.ConceptRegistryObligationUp.ConceptRegistryObligationTasteGate\_single\_carrier\_alignm`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, complete philosophy-package governance, correctness of every concept entry, or public export without the displayed obligation rows.

Upgrade path: Lean closure requires Nontrivial ConceptRegistryObligationUp and FieldFaithful ConceptRegistryObligationUp instances preserving $E, A, L, F, S, X, H, C, P, N$.

Chapter 1288

A Concrete Naming Certificate for $\mathsf{KernelPhaseRefusal}^{\uparrow}$

The $\mathsf{KernelPhaseRefusal}^{\uparrow}$ packet realizes the phase-bundle discipline of chapter 9462 as a finite NameCert surface for refusing targets that attempt to cross a displayed programme boundary. It sits beside the claim-entry discipline of chapter 1209, the promotion table of chapter 1274, the formal-target ledger of chapter 1282, and the closure reference carrier of chapter 1265. Those siblings store claim rows, upgrade rows, target skeletons, and closure references; this packet binds the current phase, attempted target, dependency blocker, refusal verdict, transport, continuation, provenance, and local name in one classifier surface.

Definition 1288.1 (Kernel phase-refusal carrier)**.** *A* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, T, D, R, A, H, C, Q, N)$$

with the following displayed rows:

1. *P is the current-phase row, recording the phase bundle currently licensed by the philosophy programme.*
2. *T is the attempted-target row, naming the proof target, certificate target, chapter target, bridge target, or review gate that asks to be treated as available.*
3. *D is the dependency-blocker row, recording the lower carrier, theorem, bridge, audit field, or NameCert obligation that must be displayed before T can be accepted.*
4. *R is the refusal-verdict row, classifying the target as in phase, blocked by D, or off-table relative to P.*
5. *A is the admissibility row, recording which claim-registry, upgrade-registry, formal-target-ledger, and closure-status-reference rows have been read by the classifier.*
6. *H stores componentwise* hsame *transport for phase, target, blocker, verdict, admissibility, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the current phase through the attempted target, blocker, verdict, admissibility row, provenance, and local name.*
8. *Q is the* Pkg *provenance over* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{UpgradeRegistry}^{\uparrow}$, $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$, $\mathsf{ClosureStatusReference}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{KernelPhaseRefusal}^{\uparrow}}$ *row.*
The carrier has no coordinate for accepting a later-phase target without a blocker row, promoting an off-table verdict to an in-phase verdict, erasing the current phase after classification, or replacing the finite dependency route by ambient attraction to a famous goal.

Lean4 (definition) `BEDC.Derived.KernelPhaseRefusalUp.KernelPhaseRefusalUp`
Lean4 ✓ `BEDC.Derived.KernelPhaseRefusalUp.KernelPhaseRefusalTasteGate single carrier alignment`

Theorem 1288.2 (Kernel phase-refusal phase order)**.** *For the carrier of Definition 1288.1, every accepted* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *packet reads the current-phase row P before the attempted-target row T, and reads the dependency-blocker row D before any refusal verdict that places T outside the current phase. A target above the current phase cannot be accepted unless its blocker row is displayed in the same finite packet.*

Proof. Project the carrier $K = (P, T, D, R, A, H, C, Q, N)$. The phase row, target row, blocker row, and verdict row are separate coordinates, and the continuation row C orders their public readback.

The sibling rows preserve the same ordering. $\mathsf{ClaimRegistryLayer}^{\uparrow}$ supplies the admitted claim row, $\mathsf{UpgradeRegistry}^{\uparrow}$ supplies the status and blocker route, $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$ supplies the target skeleton, and $\mathsf{ClosureStatusReference}^{\uparrow}$ supplies the closure report that bounds the current phase.

Finally inspect H, Q, N. Transport, provenance, and local naming preserve the displayed packet; they do not create a second route from T to acceptance. Therefore a later-phase target without D is not an accepted read of the phase-refusal packet. □

Theorem 1288.3 (Kernel phase-refusal off-table boundary)**.** *An accepted* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *packet exports only the current phase, attempted target, displayed blocker, refusal verdict, admissibility row, componentwise transport, displayed continuation, provenance, and local naming. It does not export a hidden target queue, a host-priority override, a famous-goal exception, a bridge-completion claim, or a proof that the blocker has been discharged.*

Proof. Use Definition 1288.1. The carrier exposes exactly the nine rows $P, T, D, R, A, H, C, Q, N$, and none of those rows is a hidden queue, host-priority override, exception list, bridge-completion proof, or blocker discharge.

Use Definition 1288.2. When the verdict places T above the current phase, the packet must keep the blocker row visible. The verdict may refuse the target or record the displayed dependency, but it cannot silently convert that dependency into completion.

The structural rows do not alter this boundary. Componentwise hsame, displayed Cont, Pkg provenance, and local $\mathsf{NameCert}$ naming replay the same finite object, so every listed stronger reading lies outside the accepted $\mathsf{KernelPhaseRefusal}^{\uparrow}$ surface. □

Theorem 1288.4 (Kernel phase-refusal admissibility obligation)**.** *For the carrier $K = (P, T, D, R, A, H, C, Q, N)$ of Definition 1288.1, any accepted* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *packet must expose the admissibility row A as the claim-registry, upgrade-registry, formal-target-ledger, and closure-status-reference read before the verdict row R can classify the attempted target T.*

Proof. Project the carrier $K = (P, T, D, R, A, H, C, Q, N)$. The admissibility coordinate A is separate from the target row T and the verdict row R, so the packet cannot classify T using an implicit host-side admissibility check.

Use phase order from Definition 1288.2. The current phase P, attempted target T, and dependency blocker D must be read in the displayed order before a later-phase target can be treated as available.

Now read the sibling registries named by the carrier. $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{UpgradeRegistry}^{\uparrow}$, $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$, and $\mathsf{ClosureStatusReference}^{\uparrow}$ are exactly the rows stored by A for this classifier read.

Finally apply the off-table boundary of Definition 1288.3. Since the packet has no hidden queue or priority override, R can classify T only after the displayed admissibility row has been exposed. □

Theorem 1288.5 (Kernel phase-refusal transport obligation)**.** *For every accepted* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *packet* $K = (P, T, D, R, A, H, C, Q, N)$*, componentwise* hsame *transport through* H *and displayed* Cont *replay through* C *preserve the phase row* P*, target row* T*, blocker row* D*, verdict row* R*, admissibility row* A*, provenance row* Q*, and local name* N*. Such transport cannot turn a blocked or off-table verdict into an accepted target.*

Proof. Project $K = (P, T, D, R, A, H, C, Q, N)$. The rows being transported are exactly the displayed coordinates of the carrier; H supplies componentwise hsame and C replays the same route under Cont.

Use Definition 1288.2. The blocker row remains part of the same packet whenever the attempted target lies outside the current phase, so transport cannot erase D while preserving the classifier read.

Apply Definition 1288.3. The verdict row exports only the visible refusal boundary and never stores a bridge-completion claim, blocker discharge, host-priority override, or hidden target queue.

It follows that H and C preserve the displayed P, T, D, R, A, Q, N rows as classifier data. A blocked or off-table verdict therefore remains a blocked or off-table verdict under the packet transport, and no formal verification or phase completion is claimed here. □

Theorem 1288.6 (Kernel phase-refusal scope closure)**.** *For every accepted* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *packet* $K = (P, T, D, R, A, H, C, Q, N)$*, the scoped read of the packet is exhausted by phase-order exposure through* P, T, D, R*, off-table boundary exposure through* R*, admissibility exposure through* A*, componentwise transport through* H*, displayed replay through* C*, prediction exposure, independence separation from registry siblings, provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 1288.1. The phase, target, blocker, verdict, admissibility, transport, replay, provenance, and local name are the only displayed rows.

Use Definition 1288.2 for the order of P, T, D, R, Definition 1288.3 for the refusal boundary, and Definition 1288.4 for the registry rows read by A.

Apply Definition 1288.5 to keep the same rows under H and C. The prediction and independence witness then state the external tests for the same finite packet: a no-blocker target is refused, and the classifier remains separate from the registry and closure-reference siblings.

Thus the scoped route contains exactly the displayed carrier rows and the named tests. It does not add target validity, phase completion, claim-registry status, or bridge completion. □

Falsifiable prediction: A packet whose attempted target lies above the current phase without a displayed dependency-blocker row cannot be accepted by the phase-refusal classifier.

Independence witness: claimregistrylayer, upgraderegistry, philosophyformaltargetledger, closurestatusreference: this carrier binds phase order, target admissibility, and refusal verdict in one

packet, while the siblings store claims, upgrade rows, formal targets, or closure references without the phase-refusal classifier.

The Lean follow-up must provide `lean4/BEDC/Derived/KernelPhaseRefusalUp/TasteGate.lean` with Nontrivial KernelPhaseRefusal$^{\uparrow}$ and FieldFaithful KernelPhaseRefusal$^{\uparrow}$ instances. The field-faithfulness witness must preserve $P, T, D, R, A, H, C, Q, N$ as distinct rows so the checked TasteGate surface can rule out no-blocker acceptance, off-table promotion, and hidden target-queue readings.

Closure status for NameCert$_{\text{KernelPhaseRefusal}^{\uparrow}}$.

Constructive story: The kernel phase-refusal certificate is generated from a $BHist$ packet that records phase order, target admissibility, dependency blockers, and a refusal verdict in one visible carrier. The construction first reads the phase row, then the attempted target row, then the blocker row, and finally the refusal boundary that says the target is off-table for that phase. The $hsame$ relation preserves the verdict under equivalent histories, while Cont can only propagate the same phase-refusal packet. The phase-order theorem, off-table boundary theorem, prediction, and independence witness keep the object separate from claim registries, formal-target ledgers, status references, or closure metadata that do not carry the refusal classifier.

Theory closure: TheoryGate(NameCert$_{\text{KernelPhaseRefusal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1288.1, Definition 1288.2, Definition 1288.3, Definition 1288.4, Definition 1288.5, Definition 1288.6, and the checked TasteGate row bind the scoped route for phase order, refusal boundaries, admissibility, transport, replay, prediction, independence, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{KernelPhaseRefusal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: It does not claim target invalidity in all phases, claim-registry status, or closure metadata beyond the recorded refusal packet.

Upgrade path: Reaching public closure requires a public export route and Lean targets for the phase, admissibility, and refusal-preservation rows.

Chapter 1289

A Concrete Naming Certificate for VerificationFailureRoadmap$^{\uparrow}$

The VerificationFailureRoadmap$^{\uparrow}$ packet realizes the verification-failure roadmap of chapter 9462 and subsection 9462.32.9 as a finite NameCert carrier for checked reports, audit rows, failure certificates, downgrade routes, roadmap boundaries, and cannot-claim rows. It sits beside chapter 1237, chapter 1233, chapter 1229, chapter 1259, and chapter 1274. Those siblings expose typed failure, failure certificates, gap-and-bridge audit, forbidden-dependency gates, and promotion registries separately; this packet binds the full report, failure, downgrade, roadmap, refusal, transport, replay, provenance, and local naming surface before a verification-failure route can be consumed.

Definition 1289.1 (Verification failure roadmap carrier)**.** *A* VerificationFailureRoadmap$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (R, A, F, D, M, U, B, T, C, P, N)$$

with the following displayed rows:

1. *R is the checked-report row, recording the theorem, definition, bridge, audit, or closure-status report whose advertised verification state is being inspected.*
2. *A is the audit-report row, recording the finite checker, marker, dependency, bridge, and paper-to-Lean rows that make the report inspectable.*
3. *F is the failure-certificate row, consumed from* TypedFailureCertificate$^{\uparrow}$*,* FailureCertificate$^{\uparrow}$*,* GapFailureBridgeAudit *or* AxiomPurityGate$^{\uparrow}$*.*
4. *D is the downgrade-route row, recording how the advertised status is lowered after F is read.*
5. *M is the roadmap-boundary row, recording which named blocker must be discharged before the report may be promoted.*
6. *U is the upgrade-registry row, consumed from* UpgradeRegistry$^{\uparrow}$ *and recording the current status, next status, required field, and blocker.*
7. *B is the cannot-claim row, recording the stronger readings that the failed report may not export.*
8. *T records componentwise* hsame *transport for report, audit, failure, downgrade, roadmap, upgrade, cannot-claim, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay through the report, audit, failure certificate, downgrade route, roadmap boundary, upgrade registry, and cannot-claim rows.*

10. *P is the* Pkg *provenance over* VerificationFailureRoadmap$^{\uparrow}$, TypedFailureCertificate$^{\uparrow}$, FailureCertificate$^{\uparrow}$, GapFailureBridgeAudit$^{\uparrow}$, AxiomPurityGate$^{\uparrow}$, UpgradeRegistry$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{VerificationFailureRoadmap}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden checker success, silent marker promotion, unstated bridge completion, unreported axiom-purity discharge, erased failure certificates, or roadmap advancement without the displayed blocker row.

Theorem 1289.2 (VerificationFailureRoadmap NameCert obligations)**.** *For the carrier of Definition 1289.1, the local* $\mathsf{NameCert}_{\mathsf{VerificationFailureRoadmap}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$V = (R, A, F, D, M, U, B, T, C, P, N),$$

checked-report display through R, audit-report display through A, failure-certificate display through F, downgrade-route display through D, roadmap-boundary display through M, upgrade-registry display through U, cannot-claim display through B, componentwise hsame *transport through T, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.TasteGate.VerificationFailureRoadmapNameCertObligatio`

Proof. Project the finite carrier. The report, audit, failure, downgrade, roadmap, upgrade-registry, and cannot-claim rows are displayed before the structural rows are read.

Read the sibling rows in order. TypedFailureCertificate$^{\uparrow}$ and FailureCertificate$^{\uparrow}$ supply failure branches, GapFailureBridgeAudit$^{\uparrow}$ supplies the gap-and-bridge audit row, AxiomPurityGate$^{\uparrow}$ supplies forbidden-dependency refusal, and UpgradeRegistry$^{\uparrow}$ supplies the current-status and next-status registry entry.

Finally inspect T, C, P, N. These rows preserve, replay, source, and name the same displayed packet. They do not create a hidden checker result, bridge completion, or marker promotion, so the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.verificationFailureRoadmap taste gate surface`

Theorem 1289.3 (VerificationFailureRoadmap downgrade route)**.** *Every accepted* VerificationFailureRoadmap$^{\uparrow}$ *packet*

$$V = (R, A, F, D, M, U, B, T, C, P, N)$$

routes a failed verification report from R through the audit row A and failure-certificate row F before the downgrade row D can be read. A consumer that reads a lowered status without A and F leaves the accepted carrier and cannot cite the packet as a roadmap result.

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.VerificationFailureRoadmapDowngradeRoute`

Proof. Use the NameCert obligations of Definition 1289.2. The report row R, audit row A, failure-certificate row F, and downgrade row D are all separate displayed coordinates.

The replay row C records the order in which the report is inspected, the audit row is read, the failure certificate is consumed, and the downgrade route is exposed. Transport through T preserves these coordinates rather than replacing them.

Therefore a lowered report that omits A or F is not a read of the same carrier. It may be a detached editorial statement, but it is not an accepted VerificationFailureRoadmap$^{\uparrow}$ route. □

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.TasteGate.VerificationFailureRoadmapDowngradeRoute`

Theorem 1289.4 (VerificationFailureRoadmap nonclaim boundary)**.** *For every accepted* VerificationFailureRoadma *packet, the roadmap-boundary row M, upgrade-registry row U, and cannot-claim row B jointly determine which stronger verification readings are refused. The packet does not export a checked theorem, bridge-checked status, axiom-clean status, public closure, or completed roadmap item unless that reading is licensed by the displayed registry and not listed in B.*

Proof. Start from the carrier inventory. The roadmap boundary, upgrade registry, and cannot-claim row are displayed as M, U, B and are named by the same local $\mathsf{NameCert}_{\mathsf{VerificationFailureRoadmap}^{\uparrow}}$ row.

Apply the downgrade-route theorem Definition 1289.3. A failed report reaches consumers only after the audit and failure rows are read, so no stronger verification reading can bypass those rows.

The structural rows preserve the same refusal boundary. Since T, C, P, N do not add a completion coordinate, any reading listed in B or blocked by M, U lies outside the accepted public export of the packet. □

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.TasteGate.VerificationFailureRoadmapNonclaimBoundary`

Theorem 1289.5 (VerificationFailureRoadmap sibling independence)**.** *The* $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ *carrier depends on* $\mathsf{TypedFailureCertificate}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, *and* $\mathsf{UpgradeRegistry}^{\uparrow}$ *without being bijective to any one sibling under NameCert-row preservation.*

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.TasteGate.VerificationFailureRoadmapSiblingIndepender`

Proof. Compare the displayed rows. $\mathsf{TypedFailureCertificate}^{\uparrow}$ names typed failure branches, $\mathsf{FailureCertificate}^{\uparrow}$ names rejected gate attempts, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ binds typed gaps to bridge-audit rows, $\mathsf{AxiomPurityGate}^{\uparrow}$ binds forbidden-dependency refusal, and $\mathsf{UpgradeRegistry}^{\uparrow}$ records current and next status for promotion claims.

The present carrier stores those component rows together with checked report, audit report, downgrade route, roadmap boundary, cannot-claim row, transport, replay, provenance, and local NameCert rows. A row-preserving bijection to any single sibling would have to preserve fields that the sibling does not carry.

Thus the siblings are inputs to one verification-failure roadmap packet, not alternative names for the same carrier. □

Theorem 1289.6 (VerificationFailureRoadmap bridge-facing schema route)**.** *Any bridge-facing reader of an accepted* $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ *packet must expose the checked-report row R, audit-report row A, failure-certificate row F, downgrade-route row D, roadmap-boundary row M, upgrade-registry row U, cannot-claim row B, componentwise transport row T, displayed replay row C, package provenance row P, and local* NameCert *row N before any bridge schema may consume the packet. A schema that reads only the downgrade row, only the cannot-claim row, or only a sibling failure certificate is not a bridge-facing read of the same roadmap carrier.*

Lean4 ✓ `BEDC.Derived.VerificationFailureRoadmapUp.TasteGate.VerificationFailureRoadmapBridgeFacingSchem`

Proof. Start with Definition 1289.2. It lists $R, A, F, D, M, U, B, T, C, P, N$ as the displayed rows of the local NameCert surface, so a bridge-facing reader cannot omit any of those rows and still cite the accepted carrier.

Apply the downgrade route Definition 1289.3. The failed report is read from R through A and F before D is available, which blocks a bridge schema from consuming a detached downgrade row.

Next use the nonclaim boundary and sibling independence results, Definition 1289.4 and Definition 1289.5. The rows M, U, B determine the stronger readings refused by the packet, while the sibling failure and audit packets remain inputs rather than replacements for the roadmap carrier.

It remains to carry the exposed route to a bridge-facing schema. Transport through T, replay through C, provenance through P, and local naming through N preserve the same row inventory, so the bridge schema may consume only the fully exposed verification-failure roadmap packet. □

Falsifiable prediction: Any reported verification upgrade that lacks the displayed audit report, failure-certificate, downgrade, and cannot-claim rows is rejected by the VerificationFailureRoadmapUp classifier.

Independence witness: typedfailurecertificate, failurecertificate, gapfailurebridgeaudit, axiompuritygate, upgraderegistry: no sibling is bijective because each supplies only a component row while VerificationFailureRoadmapUp binds report, failure, downgrade, roadmap, refusal, transport, replay, provenance, and NameCert rows in one packet.

The Lean follow-up must provide Nontrivial VerificationFailureRoadmap$^{\uparrow}$ and FieldFaithful VerificationFailureRoadmapUp instances in the future namespace `BEDC.Derived.VerificationFailureRoadmapUp`. The field-faithfulness witness must preserve $R, A, F, D, M, U, B, T, C, P, N$ as distinct rows before a checked TasteGate theorem may classify verification-failure reports.

Closure status for NameCert$_{\text{VerificationFailureRoadmap}^{\uparrow}}$.

Constructive story: The verification failure roadmap is generated from $BHist$ rows that name an audit report, failure certificate, downgrade row, cannot-claim row, roadmap route, and sibling references that remain outside the roadmap carrier. The carrier first records the failure row, then the route toward weaker status or exposed gap, and finally allows Cont to carry that roadmap forward as the same plan. The $hsame$ relation keeps the failure row stable under representative change. NameCert obligations, downgrade route, nonclaim boundary, sibling independence, prediction, and independence witness make the object a visible response plan for verification failure rather than a proof or audit packet.

Theory closure: TheoryGate(NameCert$_{\text{VerificationFailureRoadmap}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the verification failure-roadmap carrier, NameCert obligations, downgrade-route theorem, nonclaim-boundary theorem, sibling-independence theorem, bridge-facing schema route, falsifiable prediction, independence witness, and complete refusal surface over TypedFailureCertificateUp, FailureCertificateUp, GapFailureBridgeAuditUp, AxiomPurityGateUp, UpgradeRegistryUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{VerificationFailureRoadmap}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim verification success, proof recovery, hidden gap closure, or substitution by downgrade and audit sibling packets.

Upgrade path: Mature treatment requires a checked bridge surface and the audit-clean verification route for the complete failure-roadmap schema.

Chapter 1290

A Concrete Naming Certificate for $\mathsf{CausalIntervention}^{\uparrow}$

The $\mathsf{CausalIntervention}^{\uparrow}$ packet realizes the cross-history causal-dependence step of Definition 9462.2145 as a finite NameCert carrier. The recognized object is Pearl-style causal intervention: one named intervention row changes the admitted dependence surface while the observer histories, causal row, rate boundary, commitment row, transport, replay, provenance, and local name remain visible. Its programme context is chapter 9462; its sibling inputs are the multi-history configuration surface of chapter 474, the cross-history dependence surface of chapter 478, the causal commitment boundary of chapter 505, and the maximum-rate boundary of chapter 594.

Definition 1290.1 (Causal intervention carrier)**.** *A* $\mathsf{CausalIntervention}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, S, T, D, J, R, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathrm{MultiHistConfig}^{\uparrow}$ *row exposing the finite observer-history family.*
2. *S and T are the source and target history rows selected from M.*
3. *D is the displayed* $\mathsf{CrossHistCausal}^{\uparrow}$ *dependence row from S into T before the intervention is read.*
4. *J is the finite intervention row naming the changed local input, held as packet data rather than as a host-level command.*
5. *R is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row bounding the affected dependence route after J is admitted.*
6. *K is the* $\mathsf{CausalCommitment}^{\uparrow}$ *row recording the forward-looking commitment licensed by the intervention and dependence rows.*
7. *H records componentwise* hsame *transport for $M, S, T, D, J, R, K, C, P, N$.*
8. *C records the displayed* Cont *routes by which the configuration, dependence, intervention, rate, commitment, and consumer handoff are replayed.*
9. *P is the* Pkg *provenance over* $\mathrm{MultiHistConfig}^{\uparrow}$, $\mathsf{CrossHistCausal}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{MaxCausalRate}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CausalIntervention}^{\uparrow}}$ *row.*

The classifier compares carriers only through these rows. It has no coordinate for a hidden do-operator, counterfactual universe, global observer frame, quotient of histories, host probability space, semantic causal graph, or unbounded physical dynamics.

Lean4 ✓ `BEDC.Derived.CausalInterventionUp.CausalInterventionTasteGate single carrier alignment`

Theorem 1290.2 (Causal intervention NameCert obligations)**.** *For the carrier of Definition 1290.1, the local* $\mathsf{NameCert}_{\mathsf{CausalIntervention}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for* $I = (M, S, T, D, J, R, K, H, C, P, N)$*, multi-history configuration exposure through* M*, cross-history dependence exposure through* D*, intervention-row accountability through* J*, maximum-rate accountability through* R*, causal-commitment accountability through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted packet to its displayed rows. The multi-history configuration row supplies the finite history family, while S and T select the two visible histories consumed by the dependence row.

Read the causal layer next. The row D is the cross-history dependence surface, J is the intervention row that changes the admitted local input, R bounds the affected route, and K records the forward commitment licensed by those rows.

Finally apply the structural rows. Componentwise transport is supplied only by H, replay only by C, provenance only by P, and local naming only by N. Since the carrier lists no hidden do-operator, counterfactual universe, global frame, quotient, probability space, semantic graph, or unbounded dynamics, those objects are not NameCert obligations for $\mathsf{CausalIntervention}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CausalInterventionUp.CausalInterventionNameCertObligations`

Theorem 1290.3 (Causal intervention falsifiable non-escape)**.** *Any accepted* $\mathsf{CausalIntervention}^{\uparrow}$ *consumer that claims a cross-history intervention effect must factor through the finite route*

$$M, S, T, D, J, R, K, H, C, P, N.$$

If the claimed effect cannot identify its multi-history configuration row, source and target histories, dependence row, intervention row, maximum-rate row, causal-commitment row, componentwise transport, replay, provenance, and local name, then the claim fails the $\mathsf{CausalIntervention}^{\uparrow}$ *classifier rather than escaping to a host-level causal semantics.*

Proof. Start with carrier exposure from Definition 1290.1. Every public read of the intervention packet has only the displayed rows $M, S, T, D, J, R, K, H, C, P, N$.

Apply the NameCert obligations of Definition 1290.2. They force the consumer to read configuration, dependence, intervention, rate, and commitment before it can export an intervention effect.

The failure condition is therefore finite and falsifiable. A consumer missing one of the listed rows has no projection target in the packet, while a consumer seeking a hidden causal operator, counterfactual universe, global observer frame, quotient history, host probability space, semantic graph, or unbounded dynamics asks for a coordinate the carrier does not contain. □

Lean4 ✓ `BEDC.Derived.CausalInterventionUp.CausalIntervention falsifiable nonescape`

Falsifiable prediction: Any claimed CausalInterventionUp effect that does not expose the rows $M, S, T, D, J, R, K, H, C, P, N$ is rejected by the local carrier classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{CausalIntervention}^{\uparrow}}$**.**

Constructive story: A CausalIntervention$^{\uparrow}$ packet is generated from multi-history configuration, selected source and target histories, a cross-history causal row, a finite intervention row, a maximum-rate row, a causal-commitment row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CausalIntervention}\uparrow}$, obligation)

Scope closed: The obligation scope records the finite intervention carrier, its NameCert obligation surface, and its falsifiable non-escape boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{CausalIntervention}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not derive numerical dynamics, Pearl calculus laws, a host probability model, a semantic causal graph, global spacetime, Lean verification, or bridge checking for CausalIntervention$^{\uparrow}$.

Upgrade path: The next paper grade is scoped closure for causal-calculus consumer routes over carrier exposure, intervention-row accountability, causal-dependence accountability, rate accountability, commitment accountability, transport, replay, provenance, and local naming.

Chapter 1291

A Concrete Naming Certificate for $\mathsf{CrossHistoryCausalDependence}^{\uparrow}$

The $\mathsf{CrossHistoryCausalDependence}^{\uparrow}$ packet realizes the multi-history physics route of subsection 9462.32.10 as a finite no-global-frame causal-route packet. It sits after the uniform metric estimate of chapter 1159, the scientific model bridge of chapter 1273, and the physical law bridge of chapter 1285. Those siblings expose estimate discipline, model-facing bridge rows, and law-facing bridge rows; this carrier binds a cross-history dependence route without asserting Lorentz geometry, completed physical law, or empirical success.

Definition 1291.1 (Cross-history causal dependence carrier)**.** *A* $\mathsf{CrossHistoryCausalDependence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, B, R, S, F, M, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-history row carrying the first displayed history, its local event window, and its admissible observation route.*
2. *B is the target-history row carrying the second displayed history, its local event window, and its admissible observation route.*
3. *R is the causal-route row stating that the target window is read as dependent on the source window through displayed continuation data rather than through host time.*
4. *S is the stability row recording that the dependence predicate is preserved by the componentwise history transport used by the packet.*
5. *F is the no-global-frame row excluding synchronized clock, privileged observer frame, ambient spacetime coordinate, and host-valued cross-history equality.*
6. *M is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *estimate row available to consumers that later compare route bounds through a finite metric-estimate surface.*
7. *L is the bridge-ledger row recording how* $\mathsf{ScientificModelBridge}^{\uparrow}$ *and* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *may consume the causal dependence as a displayed scientific or law-facing route.*
8. *H records componentwise* hsame *transport for $A, B, R, S, F, M, L, C, P, N$.*
9. *C records the displayed* Cont *routes by which the two history rows, causal route, stability row, no-global-frame refusal, metric-estimate row, and bridge-ledger row are replayed.*

10. *P is the* Pkg *provenance over* $\mathsf{CrossHistoryCausalDependence}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{ScientificModelBridge}^{\uparrow}$, $\mathsf{PhysicalLawBridge}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CrossHistoryCausalDependence}^{\uparrow}}$ *row.*

The classifier compares packets only through these rows. It has no coordinate for global simultaneity, preferred frame, semantic spacetime, quotient history equality, unbounded empirical confirmation, or a causal relation imported outside the displayed route.

Lean4 (definition) `BEDC.Derived.CrossHistoryCausalDependenceUp.CrossHistoryCausalDependenceUp`

Theorem 1291.2 (Cross-history causal dependence NameCert obligations)**.** *For the carrier of Definition 1291.1, the local* $\mathsf{NameCert}_{\mathsf{CrossHistoryCausalDependence}^{\uparrow}}$ *surface consists exactly of carrier admission for $D = (A, B, R, S, F, M, L, H, C, P, N)$, source-history exposure through A, target-history exposure through B, causal-route exposure through R, transported dependence stability through S, no-global-frame refusal through F, metric-estimate handoff through M, scientific-model and physical-law bridge handoff through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted packet to its displayed rows. The rows A and B supply the two history-local event windows, while R supplies the finite route by which the second window is read as dependent on the first.

Read stability and refusal next. The row S keeps the route stable under the componentwise history transport of the packet, and the row F blocks every read that would replace the route by synchronized time, privileged observer frame, ambient spacetime, or host-valued cross-history equality.

Finally read the consumer handoff. The estimate row M is the only metric-estimate surface available to later bounded-route consumers, and L is the only scientific-model or physical-law bridge ledger. Transport, replay, provenance, and local naming through H, C, P, N preserve exactly those displayed rows, so the listed surface exhausts the local NameCert obligations. □

Theorem 1291.3 (Cross-history causal dependence nonframe invariance)**.** *Every accepted* CrossHistoryCausalDeper *consumer that reads a cross-history causal dependence factors through $A, B, R, S, F, M, L, H, C, P, N$. In particular, if the no-global-frame row F is absent, or if a consumer replaces F by a synchronized clock, privileged observer frame, ambient spacetime coordinate, or host-valued equality of histories, then the consumer no longer reads the accepted packet.*

Proof. Start from the carrier inventory of Definition 1291.1. The causal dependence is not a bare relation: it is the route row R between the displayed source and target histories A, B under the stability row S and the no-global-frame row F.

Apply the NameCert obligation surface of Definition 1291.2. Any accepted consumer route must preserve the two history rows, the route row, the stability row, the refusal row, the estimate handoff, the bridge ledger, transport, replay, provenance, and local naming.

Thus deleting F deletes one of the carrier obligations. Replacing F by global simultaneity, a preferred frame, ambient spacetime, or host equality adds a coordinate not present in D. Componentwise hsame, displayed Cont, Pkg, and NameCert replay cannot create that coordinate, so the accepted packet is invariant only as a no-global-frame causal-route packet. □

Lean4 ✓ `BEDC.Derived.CrossHistoryCausalDependenceUp.CrossHistoryCausalDependenceCarrier no global frame row deterministic`

Falsifiable prediction: If the no-global-frame row is absent, the packet cannot export a cross-history causal-dependence reading.

Independence witness: metric-triangle-uniform-estimate, scientificmodelbridge, physicallawbridge, standardbridgeauditpacket: the cross-history packet stores multi-history causal-route and no-global-frame rows not bijective to any sibling carrier.

The Lean follow-up file `lean4/BEDC/Derived/CrossHistoryCausalDependenceUp/TasteGate.lean` must provide explicit Nontrivial CrossHistoryCausalDependence$^{\uparrow}$ and FieldFaithful CrossHistoryCausalDependence$^{\uparrow}$ instances. These instances must preserve the two history rows, causal-route row, stability row, no-global-frame row, metric-estimate handoff, bridge-ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before `BEDC.Derived.CrossHistoryCausalDependen` can serve as the future checked namespace for this packet.

Closure status for NameCert$_{\text{CrossHistoryCausalDependence}\uparrow}$.

Constructive story: A CrossHistoryCausalDependence$^{\uparrow}$ packet is generated from two displayed history-event rows, a causal-route row, transported stability, no-global-frame refusal, metric-estimate handoff, scientific-model and physical-law bridge ledger, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{CrossHistoryCausalDependence}\uparrow}$, seed)

Scope closed: Definition 1291.1, Definition 1291.2, and Definition 1291.3 bind the finite no-global-frame carrier, NameCert surface, and nonframe invariance boundary.

Formal verification: FormalStatus(NameCert$_{\text{CrossHistoryCausalDependence}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive Lorentz geometry, maximum causal rate, completed physical law, empirical adequacy, standard-bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert surface, nonframe invariance, and TasteGate nontriviality with field faithfulness for $A, B, R, S, F, M, L, H, C, P, N$.

Chapter 1292

A Concrete Naming Certificate for BedcPhilosophyPackage$^{\uparrow}$

The BedcPhilosophyPackage$^{\uparrow}$ packet realizes the package object of subsection 9462.33.1 as one finite NameCert carrier rather than as a prose registry. It sits beside the philosophy-map target of chapter 1277, the tradition registry of chapter 1267, the science bridge of chapter 1286, the gap/failure bridge audit of chapter 1229, and the claim registry layer of chapter 1209. Those siblings expose map targets, inherited-comparison rows, science-bridge rows, typed gap/failure audit rows, and exported-claim rows; this packet binds the package object only when all those surfaces remain named coordinates of the same BEDC carrier.

Definition 1292.1 (BEDC philosophy package NameCert carrier)**.** *A* BedcPhilosophyPackage$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, R, M, G, D, S, C, A, H, K, N)$$

with the following displayed rows:

1. *T is the thesis row, recording auditable generativity and the boundary that the package is not a BEDC kernel extension.*
2. *R is the concept-registry row, recording concept entries, forbidden readings, theorem entries, and package-level registry routes.*
3. *M is the theorem-map row, recording theorem sites, definition sites, Lean targets when present, paper-only targets when present, and their package readings.*
4. *G is the gap-map row, recording typed gaps, failure branches, missing bridge rows, and not-claimed surfaces.*
5. *D is the tradition/science-map row, recording inherited tradition comparisons, scientific-object bridges, physical-audit readings, and rejected surplus.*
6. *S is the cannot-claim registry row, recording blocked primitive philosophical readings, host-equality imports, quotient imports, choice imports, global-truth imports, and hidden subject-substance imports.*
7. *C is the closure-status row, recording theory closure, formal status, bridge status, scope, not-claimed surface, and next obligation for each exported package row.*

8. *A is the audit-policy row, recording the finite export policy that rejects unregistered claims, missing cannot-claim rows, host-primitive leakage, and bridge-status collapse.*
9. *H records componentwise* hsame *transport for $T, R, M, G, D, S, C, A, K, N$.*
10. *K records displayed* Cont *routes,* Pkg *provenance, sibling routes, and source-file provenance for the same displayed rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BedcPhilosophyPackage}^{\uparrow}}$ *row.*

The carrier has no coordinate for a primitive philosophical object, a kernel constructor, a host truth predicate, an unchecked registry export, a theorem map detached from gaps, a science map detached from bridge audit, or a claim export without cannot-claim and audit-policy rows.

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackageTasteGate single carrier alignment`

Theorem 1292.2 (BEDC philosophy package registry-ledger exactness)**.** *For the carrier of Definition 1292.1, every public package entry is read from exactly one of the displayed registry-ledger rows R, M, G, D, S, C, A together with the thesis row T and the support rows H, K, N. In particular, concept entries, theorem-map entries, gap-map entries, tradition/science-map entries, cannot-claim entries, closure-status entries, and audit-policy entries cannot be exported by prose adjacency alone.*

Proof. Project the finite BHist packet. The thesis row T states the package boundary, while R, M, G, D, S, C, A display the registry, theorem-map, gap-map, tradition/science-map, cannot-claim, closure-status, and audit-policy ledgers before any export is named.

The support rows then preserve the same display. The hsame row H transports only those coordinates, the Cont and Pkg row K replays and sources only those routes, and the local NameCert row N names the resulting packet.

Thus a public package entry has no remaining carrier coordinate from which to arise. It must be a projection of the displayed rows, and prose adjacency to a theorem, definition, bridge, or registry paragraph is not an export rule. □

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackage registry ledger exactness`
↪ variant `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackageRegistryLedgerExactness`

Theorem 1292.3 (BEDC philosophy package audit-map nonescape)**.** *Every accepted* $\mathsf{BedcPhilosophyPackage}^{\uparrow}$ *audit export must pass through the cannot-claim registry row S and the audit-policy row A. If S is removed, an audit-map export can no longer refuse primitive philosophical readings uniformly across theorem-map, gap-map, tradition/science-map, closure-status, and claim-registry rows.*

Proof. Use the carrier of Definition 1292.1. The row S is the only coordinate that lists the blocked primitive readings, while A is the finite policy row that checks an exported package reading against that list.

Rows R, M, G, D, C expose registry, theorem, gap, tradition/science, and closure-status data, but none of them is the cannot-claim registry. They can say where a reading lands, what gap it has, or what closure status it carries, yet they do not by themselves refuse host equality, quotient import, choice import, global truth, or hidden subject substance.

With S present, the export route in K must replay the refusal row before N names the packet. Without S, the audit map lacks the displayed refusal coordinate, so the package cannot certify nonescape for primitive philosophical readings. □

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackage audit map nonescape`

Theorem 1292.4 (BEDC philosophy package NameCert obligations)**.** *For the carrier of Definition 1292.1, the local* $\mathsf{NameCert}_{\mathsf{BedcPhilosophyPackage}^{\uparrow}}$ *surface consists exactly of carrier admission for* $P = (T, R, M, G, D, S, C, A, H, K, N)$*, package-thesis exposure through* T*, registry-ledger exposure through* R, M, G, D, S, C, A*, componentwise* hsame *transport through* H*, displayed* Cont *replay and* Pkg *provenance through* K*, and local naming through* N*.*

Proof. Read the packet by coordinates. Carrier admission supplies the finite BHist object, and the thesis row T supplies the non-kernel-extension boundary fixed in Definition 9462.2155.

The ledger rows R, M, G, D, S, C, A then display the package contents: registry, theorem map, gap map, tradition/science map, cannot-claim registry, closure status, and audit policy. The certificate surface has to inspect those rows before it can expose any package-level reading.

Finally H, K, N preserve, replay, source, and name exactly the same packet. Since the carrier contains no other coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackage namecert obligations`

Theorem 1292.5 (BEDC philosophy package sibling independence)**.** *The sibling packets* $\mathsf{PhilosophyMapTarget}^{\uparrow}$*,* $\mathsf{TraditionRegistry}^{\uparrow}$*,* $\mathsf{ScienceBridge}^{\uparrow}$*,* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$*, and* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *are dependencies of* $\mathsf{BedcPhilosophyPackage}^{\uparrow}$ *only through displayed routes in* K *and displayed ledger rows in* R, M, G, D, S, C, A*. None is bijective with the carrier* $P = (T, R, M, G, D, S, C, A, H, K, N)$*.*

Proof. Inspect the package carrier. It stores the thesis, registry, theorem-map, gap-map, tradition/science-map, cannot-claim, closure-status, audit-policy, transport, route/provenance, and local naming rows together.

Each sibling contributes a narrower surface. The philosophy-map target names map obligations, the tradition registry names inherited-comparison rows, the science bridge names scientific bridge rows, the gap/failure bridge audit names typed gap and bridge audit rows, and the claim registry layer names export-control rows.

The package route K may cite those siblings, and the ledger rows may consume their outputs. But no sibling reconstructs all eleven displayed coordinates, and the package cannot be projected back to any one sibling without losing local rows. Hence the dependency is non-bijective. □

Falsifiable prediction: Deleting the cannot-claim registry row allows an audit-map export that no longer refuses primitive philosophical readings.

Independence witness: philosophy-map-target, traditionregistry, sciencebridge, gap-failure-bridge-audit, claimregistrylayer: none is bijective because the package carrier alone stores registry, theorem-map, gap-map, tradition/science-map, cannot-claim, closure-status, and audit-policy rows together.

The Lean `TasteGate.lean` boundary must provide $\mathsf{Nontrivial}\ \mathsf{BedcPhilosophyPackage}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{BedcPhilosophyPackage}^{\uparrow}$ instances. These instances must keep $T, R, M, G, D, S, C, A, H, K, N$ as separate displayed fields before any checked marker can cite a target in `lean4/BEDC/Derived/BedcPhilosophyPa`

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.taste gate`

Theorem 1292.6 (BEDC philosophy package scoped obligations)**.** *For the carrier* $P = (T, R, M, G, D, S, C, A, H,$ *of Definition 1292.1, the package scope is exactly the displayed TasteGate-bound NameCert surface consisting of the carrier row, registry-ledger exactness, audit-map nonescape, NameCert obligations, sibling independence, falsifiable prediction, independence witness, and the checked TasteGate marker. No package-level reading is in scope unless it is routed through one of* $T, R, M, G, D, S, C, A, H, K, N$*.*

Proof. Combine the carrier inventory with Definition 1292.2. The public rows are $T, R, M, G, D, S, C, A, H, K, N$, and the registry ledger prevents prose adjacency from exporting any additional package entry.

Apply Definition 1292.3 and Definition 1292.4. They force every accepted audit reading and every local certificate obligation to pass through the displayed cannot-claim, audit-policy, transport, replay, provenance, and local naming rows.

Finally read Definition 1292.5, the falsifiable prediction, the independence witness, and the checked TasteGate marker. Together they bind the ai-origin package to this finite carrier surface and leave no scoped coordinate outside $T, R, M, G, D, S, C, A, H, K, N$. □

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackage scoped obligations`

Theorem 1292.7 (BEDC philosophy package bridge boundary)**.** *Every bridge read of the* BedcPhilosophyPackage$^{\uparrow}$ *packet must pass through the displayed carrier rows $T, R, M, G, D, S, C, A, H, K, N$, the checked TasteGate marker, sibling independence, and audit-map nonescape. Such a bridge read cannot collapse the package into a primitive subject, primitive truth predicate, host-equality reading, quotient reading, or choice-reading; it can only replay the finite package ledger already exposed by the carrier.*

Proof. Project the carrier of Definition 1292.1. Rows $T, R, M, G, D, S, C, A, H, K, N$ are the complete package surface, and Definition 1292.2 prevents any package entry from being exported by prose adjacency.

Apply audit-map nonescape from Definition 1292.3. The cannot-claim row S and audit-policy row A must be read before a bridge can expose a package reading that refuses primitive philosophical interpretations.

Apply sibling independence from Definition 1292.5. The sibling routes may support the package through K and the ledger rows, but no sibling reconstructs the whole carrier or replaces it with a primitive host object.

Finally read the checked TasteGate marker and the scoped-obligation theorem. They keep the public package boundary tied to the displayed carrier rows, so the bridge can replay only that finite NameCert surface and cannot add subject, truth, host-equality, quotient, or choice readings. □

Lean4 ✓ `BEDC.Derived.BedcPhilosophyPackageUp.BedcPhilosophyPackage bridge boundary`

Closure status for NameCert$_{\text{BedcPhilosophyPackage}^{\uparrow}}$**.**

Constructive story: A BedcPhilosophyPackage$^{\uparrow}$ packet is generated from thesis, registry, theorem-map, gap-map, tradition/science-map, cannot-claim, closure-status, audit-policy, componentwise hsame transport, displayed Cont and Pkg routes, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{BedcPhilosophyPackage}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature package surface binds scoped obligations, bridge boundary, sibling independence, audit-map nonescape, registry-ledger exactness, falsifiable prediction, independence witness, and the checked TasteGate marker over the displayed rows $T, R, M, G, D, S, C, A, H, K, N$.

Formal verification: FormalStatus(NameCert$_{\text{BedcPhilosophyPackage}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, philosophical completeness, correctness of every registry row, or any primitive subject, truth, host-equality, quotient, or choice reading.

Upgrade path: Lean closure requires Nontrivial BedcPhilosophyPackageUp and FieldFaithful BedcPhilosophyPackageUp instances in `lean4/BEDC/Derived/BedcPhilosophyPackageUp/TasteGate.lean`.

Chapter 1293

A Concrete Naming Certificate for $\mathsf{PhysicsFalsificationSurface}^{\uparrow}$

The $\mathsf{PhysicsFalsificationSurface}^{\uparrow}$ packet realizes the physics-route falsification surface of subsection 9462.32.10 as a finite NameCert carrier. It sits after the relational physics carrier of chapter 1275, the physical-law bridge of chapter 1285, the empirical persistence carrier of chapter 1156, the continuation countermodel surface of chapter 1183, and the gap-failure bridge audit of chapter 1229. The packet binds the exact point where a physical route, a law bridge, an empirical persistence row, and a countermodel handoff must be read together before any science bridge can be exported as completed physics.

Definition 1293.1 (Physics falsification surface carrier)**.** *A* $\mathsf{PhysicsFalsificationSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (R, L, E, K, B, A, G, H, C, Q, N)$$

with the following displayed rows:

1. *R is the multi-history physics route row, recording the relational physics route whose public physical reading is under test.*
2. *L is the physical-law bridge row, recording law syntax, route interpretation, finite preservation, and bridge scope.*
3. *E is the empirical persistence row, recording the observation packets over which the proposed law remains stable.*
4. *K is either a continuation countermodel row or a failure row naming the branch where the proposed law cannot be carried forward.*
5. *B is the bridge and no-host-leak audit row, recording bridge preservation together with absence of host primitives, quotient principles, semantic equality assumptions, choice, propositional extensionality, and compiler readings.*
6. *A is the axiom-purity inspection row, recording that the cited formal surface is not being read through forbidden logical supply.*
7. *G is the typed gap routing row, recording the requested supply, admissible route, and consequence if the gap is internalized.*
8. *H stores componentwise* hsame *transport for route, law, empirical, countermodel, bridge-audit, axiom-purity, typed-gap, provenance, and naming rows.*

9. *C stores displayed* Cont *replay for physical-route inspection, law-bridge transport, empirical persistence, countermodel handoff, bridge audit, axiom-purity inspection, and typed-gap routing.*
10. *Q is the* Pkg *provenance row over* $\mathsf{RelationalPhysics}^{\uparrow}$, $\mathsf{PhysicalLawBridge}^{\uparrow}$, $\mathsf{EmpiricalRegularityPersistence}^{\uparrow}$, $\mathsf{ContinuationCountermodel}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicsFalsificationSurface}^{\uparrow}}$ *row.*

The carrier has no coordinate for completed physics detached from a countermodel or failure branch, a law bridge without empirical persistence, a bridge audit without no-host-leak evidence, an axiom-purity claim without typed gap routing, or a science-bridge export without the displayed falsification surface.

Lean4 ✓ `BEDC.Derived.PhysicsFalsificationSurfaceUp.PhysicsFalsificationSurfaceTasteGate single carrier alignment`

Theorem 1293.2 (Physics falsification surface NameCert rows)**.** *For the carrier of Definition 1293.1, the local* $\mathsf{NameCert}_{\mathsf{PhysicsFalsificationSurface}^{\uparrow}}$ *surface consists of five rows: carrier exposure for $P = (R, L, E, K, B, A, G, H, C, Q, N)$, classifier stability for the route-law-empirical triple, countermodel handoff through K, bridge and no-host-leak inspection through B, A, G, and non-escape from completed-physics export.*

Proof. Carrier exposure is projection from the displayed tuple P. The rows R, L, E give the route, law bridge, and empirical persistence data that a physical reading may inspect.

Classifier stability is read through the same displayed rows under the transport and replay coordinates H and C. The route-law-empirical triple cannot be renamed without preserving the transport of those coordinates and the provenance row Q.

The countermodel handoff is the row K. If K is a continuation countermodel, it records the route that defeats the proposed law; if it is a failure row, it records the branch on which the law cannot be carried forward.

Bridge and no-host-leak inspection are stored in B, A, G. The bridge row is checked together with axiom-purity and typed-gap routing, so no host primitive, quotient principle, semantic equality assumption, choice, propositional extensionality, or compiler reading is imported as a BEDC physics input.

Finally, non-escape follows because the carrier has no completed-physics coordinate. Any export through a science bridge must still pass through $R, L, E, K, B, A, G, H, C, Q, N$, and therefore cannot discard the falsification surface. □

Falsifiable prediction: If the countermodel row is removed, the packet may still record a law bridge and empirical persistence, but it cannot certify a physics falsification surface.

Independence witness: relationalphysics, physicallawbridge, empiricalregularitypersistence, continuationcountermodel, and gapfailurebridgeaudit: no bijection preserves route, law, empirical, countermodel, bridge-audit, axiom-purity, gap, transport, replay, provenance, and naming rows at once. RelationalPhysicsUp supplies the route without the law-empirical-countermodel audit; PhysicalLawBridgeUp supplies law transport without falsification handoff; EmpiricalRegularityPersistenceUp supplies persistence without the physical route and countermodel branch; ContinuationCountermodelUp supplies a countermodel route without the bridge and axiom-purity audit; GapFailureBridgeAuditUp supplies gap and bridge audit rows without the route-law-empirical classifier.

The Lean site `lean4/BEDC/Derived/PhysicsFalsificationSurfaceUp/TasteGate.lean` must provide explicit Nontrivial PhysicsFalsificationSurface$^{\uparrow}$ and FieldFaithful PhysicsFalsificationSurface$^{\uparrow}$ instances preserving $R, L, E, K, B, A, G, H, C, Q, N$.

Closure status for NameCert$_{\text{PhysicsFalsificationSurface}^{\uparrow}}$ **.**

Constructive story: The physics falsification surface is generated from $BHist$ rows that name a physical-facing route, law bridge, empirical persistence row, continuation countermodel, bridge-audit row, axiom-purity row, gap row, and the falsification surface tying them together. The carrier first reads the claim-facing route, then the observable and countermodel rows, and finally the audit surface that exposes how the claim can fail. The $hsame$ relation keeps this surface stable across equivalent histories, while Cont can pass the same falsification route to later audit or observation consumers. The NameCert row theorem, prediction, and independence witness record the observable boundary of a physical claim without asserting a full physical theory.

Theory closure: TheoryGate(NameCert$_{\text{PhysicsFalsificationSurface}^{\uparrow}}$, seed)

Scope closed: This chapter binds the physics falsification-surface carrier, NameCert-rows theorem, falsifiable prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{PhysicsFalsificationSurface}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim empirical truth, full physical model adequacy, or observation logic beyond the displayed claim, signature, countermodel, and falsification rows.

Upgrade path: Reaching obligationClosure requires formal claim, signature, countermodel, and falsification rows plus checked readback of their joint surface.

Chapter 1294

A Concrete Naming Certificate for $\mathsf{AllowedProofAudit}^{\uparrow}$

The $\mathsf{AllowedProofAudit}^{\uparrow}$ packet realizes the allowed and refused proof boundary of chapter 9462 and subsection 9462.31.5 as finite NameCert data. It sits beside the certificate-preserving route of chapter 481, the classifier-preserving route of chapter 590, the host primitive refusal of chapter 1157, and the gap-and-bridge refusal packet of chapter 1229. Those siblings expose certificate transport, classifier transport, host-source leakage, and gap routing; this carrier binds the proof-admissibility surface itself before a consumer may treat a proof as accepted BEDC evidence.

Definition 1294.1 (AllowedProofAudit carrier)**.** *An* $\mathsf{AllowedProofAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, F, S, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the allowed proof surface, naming the explicit witness construction or local refutation route accepted by the current carrier, classifier, and certificate;*
2. *F is the refused proof boundary, recording attempted extraction of positive content without a constructor witness and attempted closure of global truth, choice, quotient, propositional extensionality, or halting surfaces;*
3. *S is the scope row fixing the carrier, classifier, certificate, and local theorem site over which A and F are read;*
4. *W is the witness-construction row that exposes the constructor data supporting every positive proof read in A;*
5. *R is the local-refutation row that records the finite contradiction or failed classifier route used by every negative proof read in A;*
6. *L is the ledger-residue row for refused dependencies, unexported host material, failed extraction attempts, and proof obligations not discharged inside S;*
7. *H records componentwise* hsame *transport for $A, F, S, W, R, L, C, P, N$;*
8. *C records displayed* Cont *replay of allowed reads, refused reads, witness construction, local refutation, and ledger inspection;*
9. *P is the* Pkg *provenance over chapter 9462, subsection 9462.31.5, chapter 481, chapter 590, chapter 1157, chapter 1229,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{AllowedProofAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for hidden proof search, ambient semantic validity, unrecorded host extraction, global truth closure, quotient collapse, choice supply, propositional extensionality, or total halting decision.

Lean4 ✓ `BEDC.Derived.AllowedProofAuditUp.AllowedProofAuditTasteGate single carrier alignment`

Theorem 1294.2 (AllowedProofAudit NameCert obligations)**.** *For the carrier of Definition 1294.1, the local* $\mathsf{NameCert}_{\mathsf{AllowedProofAudit}^{\uparrow}}$ *surface consists exactly of carrier admission, allowed proof visibility, refused proof visibility, scope visibility, witness construction, local refutation, ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N.*

Lean4 ✓ `BEDC.Derived.AllowedProofAuditUp.AllowedProofAuditPacket namecert obligations`

Proof. Project the carrier $Q = (A, F, S, W, R, L, H, C, P, N)$. The first six rows expose accepted proof reads, refused proof attempts, the scope in which those reads are meaningful, the constructor witnesses, the local refutations, and the residue ledger.

Read the structural rows. Transport through H preserves the displayed components, replay through C keeps the consumer inside the same finite route, provenance through P records the proof-audit sources, and N names the packet.

Since the carrier contains no further proof coordinate, the listed rows exhaust the local NameCert surface. A claimed audit that omits one of them is a projection of the packet, not the accepted packet itself. □

Theorem 1294.3 (AllowedProofAudit witness construction)**.** *For an accepted* $\mathsf{AllowedProofAudit}^{\uparrow}$ *packet $Q = (A, F, S, W, R, L, H, C, P, N)$, every positive proof read in A factors through the displayed witness-construction row W, and every negative proof read in A factors through the displayed local-refutation row R. No accepted proof read may be supplied by P, H, or C unless the same route is already visible in W or R under the fixed scope S.*

Lean4 ✓ `BEDC.Derived.BHistDecaSequenceNameCertUp.BHistDecaSequenceNameCertCarrier namecert obligations`

Proof. Start with the scope row S. It fixes the carrier, classifier, certificate, and theorem site over which allowed proof reads are admitted.

For positive content, the carrier exposes only the witness-construction row W. The transport and replay rows can preserve and replay that witness, but they do not create a constructor outside the displayed row.

For negative content, the carrier exposes only the local-refutation row R. The ledger row L may record residue and refused dependencies, but it is not a proof source unless the finite contradiction or failed classifier route is displayed in R.

Thus every accepted proof read in A factors through W or R within S. The structural rows certify preservation and provenance for those routes, not independent proof supply. □

Theorem 1294.4 (AllowedProofAudit non-escape)**.** *Any proof read that extracts positive content without the witness row W, closes a refused surface recorded in F, or cites a dependency present only as ledger residue in L is rejected by the* $\mathsf{AllowedProofAudit}^{\uparrow}$ *classifier. In particular, neither certificate transport, classifier transport, host primitive refusal, nor gap-and-bridge audit routing can turn a refused proof attempt into accepted BEDC evidence unless the same finite packet exposes $A, F, S, W, R, L, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.AllowedProofAuditUp.AllowedProofAuditNonEscape`

Proof. Use the NameCert obligations of Definition 1294.2. Accepted proof reads, refused boundaries, scope, witnesses, refutations, and ledger residue are all projections from the same finite packet.

Apply the witness-construction theorem Definition 1294.3. Positive content must pass through W, and negative content must pass through R, with both rows read under the scope S.

If a proof read closes a boundary recorded in F or cites only residue in L, then it asks for a coordinate absent from Q. Transport through H, replay through C, and provenance through P cannot add that coordinate.

Therefore the classifier rejects the attempted escape. The surrounding packets named by chapter 481, chapter 590, chapter 1157, and chapter 1229 may route or refuse proof material, but they do not replace the displayed allowed-proof audit packet. □

Falsifiable prediction: Omitting either the witness-construction row W or the refused-proof boundary row F refutes the $\mathsf{AllowedProofAudit}^{\uparrow}$ packet: the classifier can no longer distinguish accepted proof evidence from host extraction or refused closure.

Independence witness: objectknowledgecertificate, gapfailurebridgeaudit, hostprimitiveleakage, classifiermorphism, and namecertmorphism: none is bijective to $\mathsf{AllowedProofAudit}^{\uparrow}$ because this carrier binds allowed proof surface, refused proof boundary, scope, witness construction, local refutation, ledger residue, transport, replay, provenance, and local naming as one proof-admissibility packet.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ AllowedProofAudit}^{\uparrow}$ and $\mathsf{FieldFaithful\ AllowedProofAudit}^{\uparrow}$ instances before the touchpoint theorem BEDC.Derived.AllowedProofAuditUp.taste_gate can serve as a checked TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{AllowedProofAudit}^{\uparrow}}$**.**

Constructive story: The allowed proof audit is generated from $BHist$ rows that record an allowed proof surface, refused proof boundary, scope row, witness construction, local refutation, ledger residue, transport, replay, provenance, and local naming. The carrier first reads the permitted proof evidence, then attaches the audit witness, and finally lets Cont carry the audited proof route forward only if the same permission and refusal rows remain present. The $hsame$ relation preserves the audit across equivalent histories. NameCert obligations, witness construction, non-escape, prediction, and independence witness therefore define the object as an audit of permitted proof use, not as a general proof checker or an untracked derivation route.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AllowedProofAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the AllowedProofAudit carrier, NameCert obligations, witness-construction theorem, non-escape theorem, falsifiable prediction, and independence witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AllowedProofAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim proof validity outside the allowed row, global proof checking, or acceptance of unaudited derivations.

Upgrade path: Reaching obligationClosure requires formal allowed-proof and audit-witness rows plus checked non-escape for untracked proof routes.

Chapter 1295

A Concrete Naming Certificate for $\mathsf{ObservationLogicBoundary}^{\uparrow}$

The $\mathsf{ObservationLogicBoundary}^{\uparrow}$ packet realizes the observation, proof-discipline, and open-meta-loop boundary of subsection 9462.31.4, subsection 9462.31.5, and subsection 9462.31.6 as one finite NameCert carrier. It sits after the subject-free observation packet of chapter 1271, the logic/meta-loop refusal packet of chapter 1261, the phase-refusal packet of chapter 1288, the gap-failure bridge audit of chapter 1229, and the finite observation residue boundary of chapter 1167. Those siblings separately expose observation without subject-substance, proof discipline without total truth, phase-sensitive refusal, bridge-audit failure rows, and finite residue boundaries; this packet binds their shared audit boundary before any consumer may export a BEDC-valid proof-boundary reading.

Definition 1295.1 (Observation-logic boundary carrier)**.** *An* $\mathsf{ObservationLogicBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, L, M, K, G, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the subject-free observation row, recording observation as displayed* Cont *replay over* BHist *and* hsame *rather than as a hidden observer substance.*
2. *L is the proof-discipline row, recording constructive witness formation, local refutation, and refusal of unwitnessed positive supply.*
3. *M is the open-meta-loop row, refusing total truth, unrestricted choice, quotient collapse, global proof search, halting closure, and self-validation of the certificate surface.*
4. *K is the phase-refusal row, recording whether the attempted proof-boundary export is in phase, blocked by a dependency, or off-table relative to the displayed programme surface.*
5. *G is the gap/failure/bridge audit row, tying typed gaps, failure branches, bridge evidence, no-host-leak evidence, and axis separation to the same boundary read.*
6. *R is the finite observation residue row, recording hidden-residue and failure-surface exposure for finite observation exports.*
7. *H stores componentwise* hsame *transport for observation, proof discipline, open meta-loop refusal, phase refusal, gap audit, residue, continuation, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from observation through proof-discipline inspection, open-meta-loop refusal, phase classification, audit routing, residue exposure, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{SubjectFreeObservation}^{\uparrow}$, $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$, $\mathsf{KernelPhaseRefusal}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationLogicBoundary}^{\uparrow}}$ *row.*

The local classifier compares carriers only through $O, L, M, K, G, R, H, C, P, N$. It has no coordinate for a subject-substance, a total truth predicate, an unrestricted choice supplier, quotient equality elimination, a halting oracle, a closed meta-loop classifier, or a proof-boundary export detached from finite observation residue.

Lean4 (definition) `BEDC.Derived.ObservationLogicBoundaryUp.ObservationLogicBoundaryUp`
Lean4 ✓ `BEDC.Derived.ObservationLogicBoundaryUp.ObservationLogicBoundaryTasteGate single carrier alignment`

Theorem 1295.2 (Observation-logic boundary NameCert obligations)**.** *For the carrier of Definition 1295.1, the local* $\mathsf{NameCert}_{\mathsf{ObservationLogicBoundary}^{\uparrow}}$ *surface consists exactly of subject-free observation through O, proof-discipline exposure through L, open-meta-loop refusal through M, phase-refusal classification through K, gap/failure/bridge audit through G, finite observation residue through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $B = (O, L, M, K, G, R, H, C, P, N)$. The tuple exposes observation, proof discipline, meta-loop refusal, phase refusal, gap audit, residue, transport, replay, provenance, and local naming as distinct rows.

Use chapter 1271 for O and chapter 1261 for L, M. The first supplies the observation row without a subject-substance coordinate; the second supplies constructive proof discipline and open meta-loop refusal without total truth, choice, quotient collapse, or halting closure.

Use chapter 1288, chapter 1229, and chapter 1167 for K, G, R. These rows record the phase boundary, the gap/failure/bridge audit surface, and the finite residue surface that must remain visible before a proof-boundary reading is exported.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no further coordinate, the displayed rows exhaust the local NameCert surface. □

Theorem 1295.3 (Observation-logic boundary axis separation)**.** *An accepted* $\mathsf{ObservationLogicBoundary}^{\uparrow}$ *packet keeps observation, proof discipline, open meta-loop refusal, phase refusal, gap/failure/bridge audit, and finite observation residue as separate classifier axes. No accepted consumer route may read observation as proof, proof discipline as total truth, bridge status as theory closure, phase refusal as blocker discharge, or finite residue as closed physical audit.*

Proof. Start with the carrier of Definition 1295.1. The rows O, L, M, K, G, R are separate coordinates before transport, continuation replay, provenance, or local naming is consulted.

Apply Definition 1295.2. Each axis is admitted through its own row and then preserved by H, C, P, N; none of the structural rows computes one axis from another.

The sibling packets enforce the same separation. $\mathsf{SubjectFreeObservation}^{\uparrow}$ refuses a hidden subject, $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ refuses total meta closure, $\mathsf{KernelPhaseRefusal}^{\uparrow}$ keeps blocker rows visible, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ separates bridge, failure, and theory axes, and $\mathsf{FiniteObservationResidueBoundar}$ keeps finite residue visible. Thus every listed collapsed reading lies outside the accepted boundary carrier. □

Theorem 1295.4 (Observation-logic boundary non-escape)**.** *Every BEDC-valid proof-boundary reading exported by an accepted* ObservationLogicBoundary$^{\uparrow}$ *packet factors through*

$$O, L, M, K, G, R, H, C, P, N.$$

If the open-meta-loop row M is deleted, the remaining rows may still describe observation, local proof discipline, phase refusal, audit routing, and finite residue, but they cannot export a BEDC-valid proof-boundary reading.

Proof. Project the accepted packet. A public proof-boundary read must preserve the observation row, proof-discipline row, open-meta-loop row, phase-refusal row, gap/failure/bridge audit row, finite residue row, transport row, replay row, provenance row, and local name.

The open-meta-loop row is not recoverable from the observation row or the proof-discipline row. Observation exposes a finite Cont read, and proof discipline exposes constructive and local-refutation behavior; neither refuses total truth, choice, quotient collapse, global proof search, halting closure, or self-validation on its own.

The remaining rows also cannot reconstruct M. Phase refusal records target admissibility, the gap/failure/bridge row records audit coordinates, and the residue row records finite observation limits. Transport, replay, provenance, and local naming preserve displayed rows, so deleting M prevents export of the named proof-boundary reading. □

Falsifiable prediction: Deleting the open-meta-loop row prevents export of a BEDC-valid proof-boundary reading, even when observation, proof-discipline, phase-refusal, audit, finite-residue, transport, replay, provenance, and local naming rows remain present.

Independence witness: subjectfreeobservation, logiccontradictionmetaloop, kernelphaserefusal, gapfailurebridgeaudit, finiteobservationresidueboundary: none is bijective because this carrier jointly stores subject-free observation, proof discipline, open-meta-loop refusal, phase refusal, gap/failure/bridge audit, finite observation residue, componentwise transport, continuation replay, provenance, and local naming in one boundary packet.

The Lean file `lean4/BEDC/Derived/ObservationLogicBoundaryUp/TasteGate.lean` must include explicit Nontrivial ObservationLogicBoundary$^{\uparrow}$ and FieldFaithful ObservationLogicBoundary$^{\uparrow}$ instances for the displayed carrier $O, L, M, K, G, R, H, C, P, N$ before `BEDC.Derived.ObservationLogicBoundaryUp` may be used as a checked boundary target.

Closure status for NameCert$_{\text{ObservationLogicBoundary}^{\uparrow}}$**.**

Constructive story: The observation-logic boundary is generated from $BHist$ rows that separately record subject-free observation, proof discipline, open-meta-loop refusal, phase refusal, gap and bridge audit, finite observation residue, and the boundary carrier that prevents the axes from collapsing. The carrier first reads the observation row, then the logic and proof-discipline rows, and finally records their separation as a boundary condition. The $hsame$ relation preserves these axes across equivalent histories, while Cont can carry the boundary forward only when observation and logic remain distinguished. NameCert obligations, axis separation, non-escape, prediction, and independence witness define a visible interface rather than a reduction of one axis to the other.

Theory closure: TheoryGate(NameCert$_{\text{ObservationLogicBoundary}^{\uparrow}}$, seed)

Scope closed: This chapter binds the observation-logic boundary carrier, NameCert obligations, axis-separation theorem, non-escape theorem, falsifiable prediction, and independence witness.

Formal verification: FormalStatus(NameCert$_{\text{ObservationLogicBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim reduction of observation to logic, logic to observation, or consumer routes that collapse the two axes.

Upgrade path: Reaching obligationClosure requires formal observation and logic rows plus checked preservation of axis separation under continuation.

Chapter 1296

A Concrete Naming Certificate for $\mathsf{AuditExportFirewall}^{\uparrow}$

The $\mathsf{AuditExportFirewall}^{\uparrow}$ packet realizes the total-audit synthesis of section 9462.31 as a finite export firewall. It sits beside the sibling audit packet $\mathsf{AuditSystem}^{\uparrow}$ of chapter 1258, the negative certificate surface of chapter 1233, the positive claim-entry layer of chapter 1209, the registry export consistency gate of chapter 1224, and the kernel acceptance audit witness of chapter 547. Those siblings supply audit packet, failure row, claim row, consistency gate, and kernel acceptance data separately; this carrier binds the same claim site before export can be read.

Definition 1296.1 (Audit export firewall carrier)**.** *An* $\mathsf{AuditExportFirewall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, P, A, E, R, G, L, H, K, Q, N)$$

with the following displayed rows:

1. *C is the claim-site row, recording the registry claim name, subject surface, carrier, classifier, and intended export site;*
2. *P is the positive-certificate row, recording the certificate that would license the claim if no active negative row addresses the same site;*
3. *A is the* $\mathsf{AuditSystem}^{\uparrow}$ *sibling row, carrying the admissible claim, positive certificate, failure certificate, refusal, export grade, and audit ledger surface of the same site;*
4. *E is the* $\mathsf{FailureCertificate}^{\uparrow}$ *row, recording missing, invalid, overstated, unchecked, or axis-confused data for the same claim site when export is blocked;*
5. *R is the* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *row, exposing the registry admission and refusal boundary for the claim;*
6. *G is the* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *row, checking status consistency across the registry entry, export grade, closure-status reading, and formal-target row;*
7. *L is the firewall-ledger row, recording whether the packet admits export, refuses export, or records a positive/negative certificate conflict;*
8. *H records componentwise* hsame *transport for claim, certificate, audit-system, failure, registry, consistency-gate, ledger, replay, provenance, and local naming rows;*
9. *K records displayed* Cont *replay for reading C, P, A, E, R, G, L without adding a host-side export override;*

10. *Q is the* Pkg *provenance over* $\mathsf{AuditExportFirewall}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditExportFirewall}^{\uparrow}}$ *row.*

The carrier has no coordinate for exporting a claim outside the displayed claim, positive certificate, audit-system, failure-certificate, registry, consistency-gate, firewall-ledger, transport, replay, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.AuditExportFirewallUp.AuditExportFirewallTasteGate single carrier alignment`

Theorem 1296.2 (Audit export firewall NameCert obligations)**.** *For the carrier of Definition 1296.1, the local* $\mathsf{NameCert}_{\mathsf{AuditExportFirewall}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (C, P, A, E, R, G, L, H, K, Q$ claim-site exposure through C, positive-certificate exposure through P, audit-system handoff through A, failure-certificate exposure through E, claim-registry exposure through R, registry-export consistency through G, firewall-ledger decision through L, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.AuditExportFirewallUp.AuditExportFirewallCarrier namecert obligations`

Proof. Project the finite packet $F = (C, P, A, E, R, G, L, H, K, Q, N)$. The claim-site row C and positive-certificate row P are visible before any export decision is read.

Next read the sibling audit rows. The $\mathsf{AuditSystem}^{\uparrow}$ row A supplies the joint positive and negative audit packet, the $\mathsf{FailureCertificate}^{\uparrow}$ row E exposes the active negative certificate when present, the $\mathsf{ClaimRegistryLayer}^{\uparrow}$ row R exposes registry admission, and the $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ row G checks the export status of the same claim site.

Finally read L, H, K, Q, N. The firewall-ledger row records the export, refusal, or conflict state; transport, replay, provenance, and local naming preserve the same displayed rows. Since the carrier has no additional export coordinate, these rows exhaust the local NameCert surface. □

Theorem 1296.3 (Audit export firewall non-escape)**.** *If an accepted* $\mathsf{AuditExportFirewall}^{\uparrow}$ *packet has the positive-certificate row P and an active failure-certificate row E addressing the same claim site C, then the firewall-ledger row L records refusal or certificate conflict. It does not produce an admitted exportable claim.*

Lean4 ✓ `BEDC.Derived.AuditExportFirewallUp.AuditExportFirewallCarrier nonescape`

Proof. Assume an accepted packet $F = (C, P, A, E, R, G, L, H, K, Q, N)$ and suppose P and E address the same claim site C. The claim row is therefore read with both a positive certificate and an active negative certificate.

The audit-system handoff A keeps positive certificate, failure certificate, cannot-claim refusal, export grade, and ledger closure in one finite audit packet. The registry row R and consistency gate G read the same claim site before export is exposed, so the negative row cannot be hidden behind a separate registry entry.

The firewall-ledger row L is the only row from which an export decision may be projected. Because L is read after P, E, R, G, it records the visible conflict or refusal rather than an admitted export. The remaining rows H, K, Q, N transport, replay, package, and name that same decision, and no hidden export coordinate remains. □

Theorem 1296.4 (Audit export firewall sibling independence)**.** *The* $\mathsf{AuditExportFirewall}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{AuditSystem}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, *or* $\mathsf{KernelAcceptanceAuditWitness}^{\uparrow}$.

Proof. Compare the displayed rows. AuditSystem$^{\uparrow}$ binds positive and negative audit data, FailureCertificate$^{\uparrow}$ supplies a typed negative certificate, ClaimRegistryLayer$^{\uparrow}$ registers or refuses a claim, RegistryExportConsistencyGate$^{\uparrow}$ blocks inconsistent registry export, and KernelAcceptanceAuditWitness$^{\uparrow}$ records kernel-side acceptance audit data.

The firewall carrier contains all those handoff roles together with the firewall-ledger row L that decides export, refusal, or certificate conflict for the same claim site. A NameCert-row-preserving bijection would have to preserve that joint conflict-blocking coordinate, but no listed sibling has it as a local row. Therefore no such bijection exists. □

Lean4 ✓ `BEDC.Derived.AuditExportFirewallUp.AuditExportFirewallCarrier sibling independence`

The Lean TasteGate.lean surface must provide Nontrivial AuditExportFirewall$^{\uparrow}$ and FieldFaithful AuditExportFirewa instances. The field-faithfulness witness must preserve $C, P, A, E, R, G, L, H, K, Q, N$ as distinct rows so positive certification, active failure, registry admission, consistency checking, firewall decision, and local naming cannot be merged.

Falsifiable prediction: If the positive certificate row and active failure row address the same claim site, the firewall packet refuses export rather than producing an admitted claim.

Independence witness: Compare auditsystem, failurecertificate, claimregistrylayer, registryexportconsistencygate, kernelacceptanceauditwitness: AuditExportFirewallUp is not bijective with any one sibling because it carries the same-site firewall-ledger decision over positive certificate, active failure, registry admission, consistency gate, and audit witness rows.

Closure status for NameCert$_{\text{AuditExportFirewall}\uparrow}$.

Constructive story: The audit export firewall is generated from $BHist$ rows that record a claim site, positive certificate row, active failure row, registry admission row, consistency gate, audit witness, and firewall decision controlling whether the object may leave the audit context. The carrier first reads the audited row, then the export gate, and finally uses Cont only to carry export-visible data through that same firewall. The $hsame$ relation preserves the firewall boundary across equivalent histories. NameCert obligations, non-escape, prediction, and independence witness keep the firewall distinct from the audit packet, failure certificate, claim registry, or kernel acceptance witness it may mention; it is the visible boundary condition for export, not the exported object itself.

Theory closure: TheoryGate(NameCert$_{\text{AuditExportFirewall}\uparrow}$, scopedClosed)

Scope closed: This chapter binds the audit export-firewall carrier, NameCert obligations, non-escape theorem, sibling-independence theorem, falsifiable prediction, independence witness, and scoped dependencies on AuditSystem$^{\uparrow}$, FailureCertificate$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, RegistryExportConsistencyGate$^{\uparrow}$, KernelAcceptanceAuditWitness$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{AuditExportFirewall}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim unrestricted export, audit correctness, socket-report content, or roadmap status outside the firewall boundary.

Upgrade path: Reaching scopedClosure requires formal export-boundary rows and checked non-escape for routes that bypass the firewall.

Chapter 1297

A Concrete Naming Certificate for FormalConstantEmpiricalValueBoundary$^{\uparrow}$

The FormalConstantEmpiricalValueBoundary$^{\uparrow}$ packet realizes the boundary in subsection 9462.28.6 as a finite NameCert surface. It sits after the cross-history causal-rate carrier of chapter 876, the maximum-rate bound carrier of chapter 594, the unit-calibration ledger of chapter 1208, and the empirical-regularity persistence carrier of chapter 1156. Those siblings expose invariant rate form, unary maximum-bound rows, unit and calibration records, and empirical persistence records; this packet binds them without collapsing formal invariant form into measured numerical value.

Definition 1297.1 (Formal-constant empirical-value boundary carrier)**.** *A* FormalConstantEmpiricalValueBoundary *carrier is a finite* BHist *packet*

$$K = (F, E, U, I, R, G, H, C, P, N)$$

with the following displayed rows:

1. *F is the formal invariant-form row, read through* CrossHistCausalRate$^{\uparrow}$ *and* MaxCausalRate$^{\uparrow}$*, exposing only observer-symmetric bound form, unary comparison, no-global-frame discipline, and structural transport.*
2. *E is the empirical measurement ledger row, read through* EmpiricalRegularityPersistence$^{\uparrow}$*, exposing observed records, measurement protocol, classifier, error ledger, and persistence-facing failure branch.*
3. *U is the unit-calibration row, read through* UnitCalibrationLedger$^{\uparrow}$*, exposing unit convention, calibration standard, conversion route, uncertainty budget, and instrument ledger.*
4. *I is the uncertainty and instrument row binding error bars, instrument identity, resolution limits, and finite replay of the measurement route.*
5. *R is the reproducibility row recording repeated finite observation packets, accepted classifier agreement, rejected runs, and displayed continuation routes for remeasurement.*
6. *G is the failure row recording missing calibration, incompatible units, instrument defeat, reproducibility failure, detached invariant form, and detached empirical value.*
7. *H records componentwise* hsame *transport for $F, E, U, I, R, G, C, P, N$.*
8. *C records displayed* Cont *routes for invariant-form readback, measurement replay, calibration replay, uncertainty reading, reproducibility replay, and failure consumption.*

9. *P is the* Pkg *provenance over* FormalConstantEmpiricalValueBoundary$^{\uparrow}$, CrossHistCausalRate$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, UnitCalibrationLedger$^{\uparrow}$, EmpiricalRegularityPersistence$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{FormalConstantEmpiricalValueBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for extracting a numerical empirical value from F alone, for promoting E, U, I, R to a structural invariant without F, or for reading a physical constant without the failure row G.

Lean4 (definition) `BEDC.Derived.FormalConstantEmpiricalValueBoundaryUp.FormalConstantEmpiricalValueBoundar`
Lean4 ✓ `BEDC.Derived.FormalConstantEmpiricalValueBoundaryUp.FormalConstantEmpiricalValueBoundaryTasteGa`
`single carrier alignment`

Theorem 1297.2 (Formal-constant empirical-value boundary NameCert obligations)**.** *For the carrier of Definition 1297.1, the local* NameCert$_{\text{FormalConstantEmpiricalValueBoundary}^{\uparrow}}$ *surface consists exactly of formal invariant-form exposure through F, empirical measurement exposure through E, unit-calibration exposure through U, uncertainty and instrument exposure through I, reproducibility exposure through R, failure exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $K = (F, E, U, I, R, G, H, C, P, N)$. The first row exposes the structural side of the boundary: observer-symmetric invariant form, maximum-rate bound form, unary comparison, and no-global-frame discipline.

Read the empirical side through E, U, I, R. These rows display measurement records, unit convention, calibration, instrument and uncertainty data, and reproducibility evidence as finite ledger coordinates rather than as consequences of the invariant-form row.

The failure row G records every way the two sides can be detached. Missing unit calibration, incompatible units, instrument defeat, reproducibility failure, a bare invariant form, and a bare empirical value are all visible refusal branches.

The structural rows H, C, P, N preserve and replay only the displayed packet. Since no further coordinate is present, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FormalConstantEmpiricalValueBoundaryUp.FormalConstantEmpiricalValueBoundaryNameCer`

Theorem 1297.3 (Formal-constant empirical-value noncollapse)**.** *An accepted* FormalConstantEmpiricalValueBound *carrier permits a consumer to compare a formal constant with an empirical value only through the full packet*

$$F, E, U, I, R, G, H, C, P, N.$$

The formal invariant-form row F cannot supply the empirical numerical value, and the empirical rows E, U, I, R cannot supply the structural invariant form. A physical-law or rate-facing consumer must preserve the boundary row G and the local NameCert *row N before citing the packet.*

Proof. Use Definition 1297.2. It separates invariant-form exposure from measurement, calibration, instrument, and reproducibility exposure.

The row F is inherited from the cross-history causal-rate and maximum-rate surfaces. Those packets expose structural bound form and no-global-frame discipline, but they carry no unit convention, instrument ledger, uncertainty budget, or reproducibility ledger.

The rows E, U, I, R are empirical rows. They provide measured records, unit calibration, uncertainty, and reproducibility, but they do not create the observer-symmetric invariant-form row supplied by F.

Finally read G, H, C, P, N. The failure row records every attempted collapse as a visible boundary case, while transport, continuation, provenance, and local naming preserve the same finite packet. Thus comparison is available only as a noncollapse boundary, not as an identification of formal constants with empirical values. □

Lean4 ✓ `BEDC.Derived.FormalConstantEmpiricalValueBoundaryUp.FormalConstantEmpiricalValueBoundaryNoncoll`

Falsifiable prediction: A physical-law consumer that reads a numerical empirical value from invariant form alone refutes the $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$ packet.

Independence witness: crosshistcausalrate, maxcausalrate, unitcalibrationledger, empiricalregularitypersistence: $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$ is not bijective to any listed sibling because it carries formal invariant form, measurement ledger, unit calibration, uncertainty, instrument, reproducibility, failure, transport, continuation, provenance, and local naming rows in one finite boundary packet.

Theorem 1297.4 (Formal-constant empirical-value boundary scope)**.** *Every accepted* FormalConstantEmpiricalValu *boundary packet is scoped only over the formal invariant row F, empirical measurement row E, unit-calibration row U, uncertainty and instrument row I, reproducibility row R, failure row G, componentwise* hsame *transport row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N of Definition 1297.1. The scope is exactly the carrier, the NameCert obligations of Definition 1297.2, and the noncollapse boundary of Definition 1297.3, read through the cross-history causal-rate, maximum-rate, unit-calibration, and empirical-regularity sibling chapters named in the chapter introduction.*

No accepted packet exports a formal-to-empirical equality, a measurement-free invariant value, a calibration-free empirical value, or a host physical law beyond those displayed rows.

Proof. Project the carrier $K = (F, E, U, I, R, G, H, C, P, N)$ from Definition 1297.1. The only primitive rows available to a boundary reader are the formal invariant row, empirical measurement row, calibration row, uncertainty and instrument row, reproducibility row, failure row, transport row, replay row, provenance row, and local naming row.

Apply Definition 1297.2. It identifies the local NameCert surface with exactly those rows and their componentwise hsame, Cont, and Pkg structure.

Finally use Definition 1297.3. The noncollapse theorem keeps the formal row and empirical rows separated through the same failure boundary G, so the cited sibling chapters contribute only their displayed rows and do not enlarge the packet scope. □

Lean4 ✓ `BEDC.Derived.FormalConstantEmpiricalValueBoundaryUp.FormalConstantEmpiricalValueBoundaryScope`

Theorem 1297.5 (Formal-constant empirical-value boundary public export)**.** *Every physical-law comparison consumer of an accepted* $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$ *packet may use it only as the reusable route*

$$F, E, U, I, R, G, H, C, P, N.$$

The public route preserves formal invariant form, empirical measurement, unit calibration, uncertainty, reproducibility, failure, componentwise transport, displayed replay, provenance, and local naming as separate rows. It exports no formal-to-empirical equality, calibration-free measured value, measurement-free invariant value, or host physical law beyond the displayed boundary packet.

Proof. Project the carrier of Definition 1297.1. The only exported rows are $F, E, U, I, R, G, H, C, P, N$, with F carrying invariant form, E, U, I, R carrying measurement, calibration, uncertainty, and reproducibility data, and G carrying the visible failure boundary.

Apply the NameCert obligations of Definition 1297.2. They identify the public comparison surface with those displayed rows and their hsame, Cont, Pkg, and local naming structure.

Use the noncollapse theorem Definition 1297.3. It prevents the formal row from becoming an empirical value and prevents the empirical rows from becoming invariant form.

Finally apply the scope theorem Definition 1297.4. The cited sibling chapters contribute only their recorded physical-law comparison rows, so the public interface is the finite route above and not an enlarged host-law assertion. □

The paper-side bridge route for this boundary is the displayed public route $F, E, U, I, R, G, H, C, P, N$ read against the accepted formal-constant and empirical-value schema: the formal row F remains the invariant-form side, the empirical rows E, U, I, R remain the measured-value side, and G, H, C, P, N record refusal, transport, replay, provenance, and local naming. This bridge adds no carrier coordinate and claims no Lean bridge; it only records that the public boundary route already matches the standard paper schema for keeping formal constants and empirical values separated.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$ packet is generated by first displaying the formal invariant-rate rows and then attaching empirical measurement, unit-calibration, uncertainty, instrument, reproducibility, failure, transport, continuation, package, and local naming rows in one finite BHist carrier. The classifier is not allowed to identify the formal row with the measured rows; it reads each coordinate through componentwise hsame and the visible Cont replay routes. The ledger story is therefore a boundary story: empirical values may be compared to invariant form only through the recorded calibration and reproducibility rows, while failed attempts to collapse the two sides remain visible as the failure coordinate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1297.5 and the paper-side bridge route bind the physical-law comparison route to the displayed carrier, NameCert obligations, noncollapse theorem, scope theorem, falsifiable prediction, and independence witness for $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: It does not claim host physical equality, extraction of a numerical value from invariant form alone, or promotion of empirical ledgers into structural constants.

Upgrade path: A Lean bridge check would have to verify the same standard bridge without enlarging the displayed rows or changing the verification axis.

Chapter 1298

A Concrete Naming Certificate for $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$

The $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ packet realizes the route of subsection 9462.32.10 as a composite finite NameCert surface. It sits after the relational physics carrier of chapter 1275, the physical-law bridge of chapter 1285, the scientific-model bridge of chapter 1273, and the quantum-state packet of chapter 217. These siblings expose relational route discipline, law-facing bridge rows, model-facing bridge rows, and quantum-state rows; this carrier binds them without asserting a completed physical theory.

Definition 1298.1 (Multi-history physics route carrier)**.** *A* $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, L, M, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RelationalPhysics}^{\uparrow}$ *row carrying observer-history identity, inter-history locality, invariant readback, visible physical-model audit, and no-global-frame refusal.*
2. *L is the* $\mathsf{PhysicalLawBridge}^{\uparrow}$ *row carrying law-certificate, empirical-persistence, bridge-audit, scientific-object, open-fit, and failure rows.*
3. *M is the* $\mathsf{ScientificModelBridge}^{\uparrow}$ *row carrying object, physical-model-audit, standard-bridge-audit, reality-constrained model-audit, open-fit, empirical-ledger, and prediction-transport rows.*
4. *Q is the* $\mathsf{QuantumState}^{\uparrow}$ *row carrying the finite quantum-state interface consumed by the route.*
5. *F is the falsification boundary row recording where the route may fail through no-global-frame refusal, failed law persistence, model-open-fit exit, quantum-state interface failure, or axiom-purity obstruction.*
6. *H stores componentwise* hsame *transport for G, L, M, Q, F, C, P, N.*
7. *C stores displayed* Cont *replay for relational-route inspection, law-bridge inspection, model-bridge inspection, quantum-state inspection, and falsification-boundary inspection.*
8. *P is the* Pkg *provenance over* $\mathsf{RelationalPhysics}^{\uparrow}$, $\mathsf{PhysicalLawBridge}^{\uparrow}$, $\mathsf{ScientificModelBridge}^{\uparrow}$, $\mathsf{QuantumState}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}}$ *row.*

The classifier compares packets only through these rows. It has no coordinate for completed relativity, completed quantum mechanics, empirical adequacy, semantic spacetime equality, host-valued physical truth, quotient history equality, or a physical law detached from the displayed falsification boundary.

Lean4 ✓ `BEDC.Derived.MultiHistoryPhysicsRouteUp.MultiHistoryPhysicsRouteTasteGate single carrier alignment`

Theorem 1298.2 (Multi-history physics route NameCert obligations)**.** *For the carrier of Definition 1298.1, the local* $\mathsf{NameCert}_{\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (G, L, M, Q, F, H, C, P, N)$*, relational-route exposure through* G*, physical-law bridge exposure through* L*, scientific-model bridge exposure through* M*, quantum-state exposure through* Q*, falsification-boundary exposure through* F*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier to its displayed rows. The relational physics row G, physical-law bridge row L, scientific-model bridge row M, and quantum-state row Q are the only physics-facing inputs.

The falsification row F is read next. It records the failure boundary for no-global-frame refusal, law persistence, model fit, quantum-state interface, and axiom-purity inspection, so the route cannot be read as a completed physical theory after deleting F.

Transport and replay are componentwise. The rows H and C preserve exactly G, L, M, Q, F together with provenance P and local naming N, so the listed rows exhaust the NameCert surface. □

Theorem 1298.3 (Multi-history physics route falsification surface)**.** *Every accepted* $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ *consumer that reads the route as physics-facing data must preserve the falsification row* F *together with* G, L, M, Q, H, C, P, N*. If a consumer removes* F*, or replaces it by completed relativity, completed quantum mechanics, empirical adequacy, semantic spacetime equality, quotient history equality, or host-valued physical truth, then the consumer has left the accepted packet.*

Proof. Start from the carrier inventory of Definition 1298.1. The row F is a visible coordinate, not a comment on the surrounding route.

Apply Definition 1298.2. Any accepted read must expose F along with the relational, law, model, and quantum rows and must transport those rows through H and replay them through C.

The carrier contains no completed-theory coordinate and no host physical-truth coordinate. Thus deleting or replacing F changes the packet rather than refining it, and the accepted physics-facing read fails. □

Theorem 1298.4 (Multi-history physics route composite handoff)**.** *The public handoff from an accepted* $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ *carrier is exactly the composite finite packet*

$$G, L, M, Q, F, H, C, P, N.$$

It may feed later route work with relational physics, physical-law bridge, scientific-model bridge, and quantum-state rows, but it supplies no completed physical theory and no escape from the displayed falsification boundaries.

Proof. Use the carrier definition to identify the public packet. The four dependency rows G, L, M, Q are carried beside the falsification row F, transport row H, continuation row C, provenance row P, and local naming row N.

The NameCert obligations keep every consumer read on those displayed rows. The falsification-surface theorem keeps the failure boundary attached to the handoff whenever the route is read as physics-facing data.

Therefore the composite handoff is finite and internal to the BEDC packet. It can be consumed by later route work, but it cannot be read as completed physical law, completed spacetime semantics, or completed quantum theory. □

Falsifiable prediction: A multi-history physics consumer that exports completed physical theory while deleting the falsification row does not read an accepted $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ packet.

Independence witness: relationalphysics, physicallawbridge, scientificmodelbridge, and quantumstate: no listed sibling carries the relational route, law bridge, model bridge, quantum-state row, falsification boundary, transport, replay, provenance, and local naming rows at once.

Closure status for $\mathsf{NameCert}_{\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ packet is generated from relational physics, physical-law bridge, scientific-model bridge, quantum-state, falsification-boundary, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1298.1, Definition 1298.2, Definition 1298.3, and Definition 1298.4 bind the finite composite route, NameCert obligations, falsification boundary, and public handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not derive relativity, quantum mechanics, empirical adequacy, standard-bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, falsification preservation, composite handoff, and TasteGate nontriviality with field faithfulness for $G, L, M, Q, F, H, C, P, N$.

Chapter 1299

A Concrete Naming Certificate for $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$

The $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ packet is the finite Real and Completion carrier that turns a shared-tail threshold into a single normal-form row before any $\mathsf{Real}^{\uparrow}$ seal consumer may read a tail window. It sits between the shared tail data of chapter 983, the seal-facing row of chapter 1171, the readback surfaces of chapter 126, chapter 301, and chapter 291, and the terminal consumers chapter 699 and chapter 38. Its role is to make the common finite threshold a displayed route object rather than a private parameter carried separately by each downstream seal.

Definition 1299.1 (Cauchy tail threshold normalizer carrier)**.** *A* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchySharedTailModulus}^{\uparrow}$ *row naming the shared tail source whose threshold is being normalized.*
2. *M is the* $\mathsf{CauchyTailModulusSeal}^{\uparrow}$ *row that exposes the seal-facing modulus bound.*
3. *Θ is the normalized shared finite threshold row.*
4. *W_0 and W_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *window readbacks cut from the same threshold Θ.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read on the same threshold.*
6. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to W_0, W_1, D.*
7. *A is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row reached after the two threshold-indexed windows have been compared.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *consumer row available only after $S, M, \Theta, W_0, W_1, D, R, A$ have been displayed.*
9. *H stores componentwise* hsame *transport for the shared-tail, seal, threshold, window, tolerance, readback, agreement, Real, provenance, ledger, and naming rows.*
10. *C stores displayed* Cont *replay from the shared-tail source through the seal row, normalized threshold, both window readbacks, tolerance ledger, regular readback, agreement seal, and Real consumer row.*

11. *P is the* Pkg *provenance over* CauchyTailThresholdNormalizer$^{\uparrow}$, CauchySharedTailModulus$^{\uparrow}$, CauchyTailModulusSe RealTailAgreementSeal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the route-exhaustion ledger recording that every seal-facing consumer has already read the same threshold Θ.*
13. *N is the local* NameCert$_{\text{CauchyTailThresholdNormalizer}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, private side thresholds, or a second Real seal route.

Definition 1299.2 (Cauchy tail threshold normalizer classifier)**.** *Two accepted* CauchyTailThresholdNormalizer$^{\uparrow}$ *carriers classify together exactly when their shared-tail source, modulus seal, normalized threshold, two StreamName windows, dyadic tolerance ledger, RegSeqRat readback, RealTailAgreementSeal route, RealUp consumer row, componentwise transports, displayed continuation replay, provenance, route-exhaustion ledger, and local naming row agree componentwise over the same threshold Θ. The classifier does not identify carriers by comparing quotient streams, by choosing host limits, by imposing host real equality, by invoking trichotomy, by importing ambient completeness, or by allowing each seal consumer to carry a private threshold.*

Lean4 (definition) `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerClassifier`

Theorem 1299.3 (Cauchy tail threshold normalizer NameCert obligations)**.** *For every accepted* CauchyTailThresholdNormalizer$^{\uparrow}$ *carrier*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N),$$

the local NameCert$_{\text{CauchyTailThresholdNormalizer}^{\uparrow}}$ *surface consists exactly of carrier admission, shared-tail source exposure through S, modulus-seal exposure through M, normalized threshold exposure through Θ, paired StreamName window readback through W_0, W_1, dyadic tolerance exposure through D, RegSeqRat readback through R, RealTailAgreementSeal handoff through A, RealUp consumer exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, route-exhaustion ledgering through L, local naming through N, and non-escape for quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, private side thresholds, and second Real seal routes.*

Proof. Carrier admission is projection from Definition 1299.1. The packet displays one shared-tail source, one seal-facing modulus row, one normalized threshold, two windows cut from that threshold, one dyadic tolerance ledger, one regular readback row, one agreement seal, one terminal Real consumer row, and five structural support rows.

The threshold row is read before any seal consumer. The source S and seal row M expose the tail data, then Θ fixes the finite threshold that both windows must use.

The readback rows are downstream of that threshold. The windows W_0, W_1, tolerance ledger D, and regular readback R are all indexed by the same Θ, so neither side can substitute a private finite window before the Real-facing handoff.

Finally A, E, H, C, P, L, N package the route as a NameCert surface. The carrier lacks quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, private side thresholds, and second Real seal routes, so the listed rows exhaust the local obligations. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerCarrier namecert obligations`

Theorem 1299.4 (Cauchy tail threshold normalizer route exhaustion)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumer of a* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto M \longmapsto \Theta \longmapsto (W_0, W_1, D, R) \longmapsto (A, E, H, C, P, L, N).$$

In particular, no such consumer can read two incompatible finite tail windows through private thresholds.

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerCarrier route exhaustion`

Proof. Start with the NameCert obligations of Definition 1299.3. They expose S and M before any Real-facing row appears.

Next read the normalized threshold Θ. The two StreamName windows, dyadic ledger, and regular readback are attached to Θ, so the threshold-indexed data rows are W_0, W_1, D, R and not side-private alternatives.

The agreement and Real rows are therefore downstream of the same finite threshold. The rows A, E, H, C, P, L, N transport, replay, source, ledger, and name the displayed route, and the carrier contains no coordinate for a second route around Θ. □

Theorem 1299.5 (Cauchy tail threshold normalizer tail-window induction)**.** *Every accepted* CauchyTailThresholdN *packet normalizes a threshold read by finite induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *tail windows. The base window is the threshold row* Θ *exposed between* S, M *and* W_0, W_1*; each successor window preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, route-exhaustion ledger, and local* NameCert *row.*

Consequently every Real-facing consumer must pass through the same finite induction surface before reading a tail window. The induction exports no quotient stream equality, selected limit, trichotomy, countable choice, ambient Real completeness, private side threshold, or second Real seal route.

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerTailWindowInduction`

Proof. Project the carrier of Definition 1299.1. Its threshold row Θ is displayed before the two $\mathsf{StreamName}^{\uparrow}$ windows, so the base step is the finite threshold read itself, not a hidden limit or quotient-stream representative.

For the successor step, use the componentwise transport row H. Moving from one displayed tail window to the next preserves the shared-tail source, seal row, threshold row, tolerance ledger, regular readback, agreement-seal handoff, Real consumer row, continuation replay, provenance, route ledger, and local naming row component by component.

The $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows supply the numerical readback discipline at each displayed window. The $\mathsf{Real}^{\uparrow}$ seal row is downstream of those rows, so no successor step can inspect a Real-facing value before the window, tolerance, and readback data have been presented.

Finally apply route exhaustion from Definition 1299.4. Every accepted Real-facing consumer already factors through

$$S \longmapsto M \longmapsto \Theta \longmapsto (W_0, W_1, D, R) \longmapsto (A, E, H, C, P, L, N),$$

so the induction has no coordinate for quotient equality, selected limits, trichotomy, countable choice, ambient completeness, private side thresholds, or a second Real seal route. □

Theorem 1299.6 (Cauchy tail threshold normalizer shared-threshold uniqueness)**.** *Let T and T' be accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carriers read by* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumers. If the two consumers read the same shared-tail source row S and the same seal-facing modulus row M through classified normalizer packets, then both consumers expose the same normalized threshold row Θ before any W_0, W_1, D, R, A, E row is available.*

In particular, the classifier cannot admit two downstream seal reads with the same S, M source data and different normalized tail thresholds.

Proof. Project both accepted carriers through Definition 1299.1. In each carrier, the source and seal rows S, M are followed by a single normalized threshold row before any window, tolerance, readback, agreement, or Real-seal row appears.

Apply route exhaustion from Definition 1299.4. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer has no route around Θ, so the downstream rows W_0, W_1, D, R, A, E can be read only after the normalized threshold has been displayed.

Finally use the classifier of Definition 1299.2. Classified packets are compared componentwise on the shared-tail source, seal row, normalized threshold, two windows, dyadic tolerance ledger, regular readback, agreement seal, and Real row. Since the two reads agree on S and M inside that classified normalizer surface, the threshold component exposed before the window rows is the same Θ in both reads. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerSharedThresholdUnique`

Theorem 1299.7 (Cauchy tail threshold normalizer window cofinality)**.** *Let T be an accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *packet with shared source row S, seal row M, and normalized threshold row Θ. Every later finite* $\mathsf{StreamName}^{\uparrow}$ *window read by a* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through a finite successor of the same Θ. No such consumer can introduce a private later threshold outside the displayed normalizer route.*

Proof. Project the carrier of Definition 1299.1. The rows S, M, Θ appear before W_0, W_1, D, R, A, E, so every seal-facing window read is downstream of the normalized threshold already displayed by the packet.

Use route exhaustion from Definition 1299.4. A $\mathsf{RealTailAgreementSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer has no route around

$$S \longmapsto M \longmapsto \Theta \longmapsto (W_0, W_1, D, R) \longmapsto (A, E, H, C, P, L, N),$$

and therefore any later finite StreamName window must be a successor window in that same finite route.

Apply shared-threshold uniqueness from Definition 1299.6. If a later consumer tried to introduce a private threshold while keeping the same S, M rows, the classified normalizer surface would expose two thresholds before the downstream window rows, contradicting the uniqueness row.

Finally transport the displayed successor window componentwise through H and replay it through C. The rows P, L, N keep the provenance, route-exhaustion ledger, and local NameCert attached to the same Θ, so the consumer receives a cofinal finite-window route rather than a hidden side threshold. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerWindowCofinality`

Theorem 1299.8 (Cauchy tail threshold normalizer public Real-seal route)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N)$$

passes through the same displayed route

$$S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N.$$

It cannot use quotient streams, selected host limits, host equality of real names, trichotomy, ambient completeness, private side thresholds, or a second seal route.

Proof. Start with route exhaustion from Definition 1299.4. It already forces every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer through the finite route from S, M to Θ, then through W_0, W_1, D, R, and finally through A, E, H, C, P, L, N.

Apply shared-threshold uniqueness from Definition 1299.6. A public seal read with the same shared source S and modulus seal M cannot split into a private side threshold before the window and readback rows are exposed.

Use window cofinality from Definition 1299.7. Every later finite $\mathsf{StreamName}^{\uparrow}$ window remains downstream of a finite successor of the same Θ, so the public route has no opening for quotient streams, selected host limits, host equality, trichotomy, or ambient completeness.

Finally read the structural rows. Transport, replay, provenance, route ledger, and local naming preserve the same carrier surface, and no listed row contains a second seal route. Therefore every public $\mathsf{Real}^{\uparrow}$ seal read factors through the displayed route and nothing else. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerRealSealPublicRoute`

Theorem 1299.9 (Cauchy tail threshold normalizer Real-facing determinacy)**.** *Let two accepted* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *consumers classify through the same* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *source row S and seal row M. Then their* $\mathsf{StreamName}^{\uparrow}$ *windows W_0, W_1,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row A are all read downstream of the same normalized threshold row Θ.*

Equivalently, the Real-facing classifier surface exposes one finite route

$$S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N$$

before either consumer can inspect a tail window, tolerance ledger, regular readback, or agreement-seal row.

Proof. Project the carrier of Definition 1299.1. The normalized threshold row Θ is a displayed coordinate between the shared source and seal rows S, M and the downstream W_0, W_1, D, R, A, E rows.

Apply route exhaustion from Definition 1299.4. Any accepted $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer must pass through Θ before reading the two StreamName windows, the dyadic tolerance ledger, the RegSeqRat readback, or the agreement-seal row.

Apply shared-threshold uniqueness from Definition 1299.6. Since the two consumers classify through the same S, M normalizer source and seal rows, the threshold exposed before the window rows is the same normalized row Θ.

Finally use the classifier of Definition 1299.2. It compares the downstream rows componentwise over the shared threshold surface, so W_0, W_1, D, R, A are determined as reads below that same Θ rather than as private tail windows or side thresholds. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerRealFacingDeterminacy`

Theorem 1299.10 (Cauchy tail threshold normalizer window scope)**.** *Every* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that classifies through a* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *packet must expose*

$$S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N$$

in that order before inspecting a tail-window row. Consequently a private side threshold or second seal route is outside the classifier surface.

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerWindowScope`

Proof. Project the carrier of Definition 1299.1. The displayed coordinates are exactly $S, M, \Theta, W_0, W_1, D, R, A, E, H$ so any classified read begins with the shared-tail source, seal row, and normalized threshold before it reaches a window.

Apply route exhaustion and shared-threshold uniqueness. The route-exhaustion theorem forces every $\mathsf{RealTailAgreementSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through the same threshold path, and uniqueness prevents two classified consumers with the same S, M rows from introducing separate threshold coordinates.

Use window cofinality and Real-facing determinacy next. They place W_0, W_1, D, R, A, E downstream of the same Θ, so the two windows, dyadic ledger, regular readback, agreement row, and Real row are not available as side reads.

Finally inspect the classifier surface. It has no coordinate for a private threshold or a second seal route, and the structural rows H, C, P, L, N only transport, replay, source, ledger, and name the same displayed packet. Thus any consumer that tries to inspect a tail-window row before the ordered route is outside the accepted classifier. □

Theorem 1299.11 (Cauchy tail threshold composition scope)**.** *Every accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N)$$

composes the shared-tail source S, modulus seal M, normalized threshold Θ, StreamName windows W_0, W_1, DyadicRatCore tolerance row D, RegSeqRat readback R, RealTailAgreementSeal row A, RealUp row E, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, route-exhaustion ledger L, and local* NameCert *row N into one finite route. No Real-facing consumer may bypass that route with private thresholds, quotient streams, selected host limits, trichotomy, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdCompositionScope`

Proof. Project the carrier of Definition 1299.1. The displayed coordinates are exactly $S, M, \Theta, W_0, W_1, D, R, A, E, H$ so the route starts at the shared-tail and seal rows before any Real-facing read is exposed.

Apply the NameCert obligations of Definition 1299.3. Those obligations make each analytic row available only as part of the same local certificate surface, with the normalized threshold placed before the paired StreamName windows, tolerance ledger, regular readback, agreement seal, and Real row.

Apply route exhaustion from Definition 1299.4. Every accepted $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer already factors through the displayed threshold route, so a private threshold, quotient stream, selected host limit, trichotomy, host real equality, or ambient completeness principle has no public coordinate to occupy.

Finally use componentwise hsame transport and displayed Cont replay. They preserve the same route under transport and continuation, while P, L, N source, ledger, and name the packet without adding a second Real-facing path. □

Theorem 1299.12 (Cauchy tail threshold normalizer noninternalization)**.** *The* CauchyTailThresholdNormalizer$^{\uparrow}$ *packet does not internalize quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, or hidden threshold choice. Its public surface is exactly the finite shared-threshold route displayed by*

$$S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N.$$

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerCarrier noninternalization`

Proof. Inspect the carrier of Definition 1299.1. Its data coordinates are the shared-tail source, seal row, normalized threshold, two windows, dyadic ledger, regular readback, agreement seal, and Real consumer row.

Inspect the structural coordinates next. Transport, continuation, provenance, route-exhaustion ledgering, and local naming preserve the displayed route; none supplies quotient equality, a host limit, host real equality, trichotomy, completeness, or hidden threshold selection.

Apply route exhaustion from Definition 1299.4. Every downstream seal consumer already factors through $S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N$, so there is no remaining public coordinate in which an excluded object could be read. □

Theorem 1299.13 (Cauchy tail threshold normalizer obligation ledger)**.** *For every accepted* CauchyTailThresholdNormalizer$^{\uparrow}$ *packet*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N),$$

the Real-facing route exposes exactly the shared-tail source S, modulus seal M, normalized threshold Θ, two StreamName$^{\uparrow}$ *windows W_0, W_1,* DyadicRatCore$^{\uparrow}$ *ledger D,* RegSeqRat$^{\uparrow}$ *readback R,* RealTailAgreementSeal$^{\uparrow}$ *row A,* Real$^{\uparrow}$ *row E, componentwise transport H, displayed replay C, provenance P, route-exhaustion ledger L, and local* NameCert *row N before any Real-facing consumer reads a tail window.*

No accepted route adds quotient-stream equality, selected host limits, host real equality, trichotomy, ambient completeness, a private side threshold, or a second Real seal route.

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerObligationLedger`

Proof. Project the carrier of Definition 1299.1 and its classifier in Definition 1299.2. The exposed coordinates are precisely $S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N$, compared componentwise over the same threshold surface.

Apply the NameCert obligations of Definition 1299.3 and route exhaustion of Definition 1299.4. They place the shared-tail source and modulus seal before Θ, and place the two StreamName windows, dyadic ledger, RegSeqRat readback, RealTailAgreementSeal row, and Real row downstream of that threshold.

Apply shared-threshold uniqueness and Real-facing determinacy from Definition 1299.6 and Definition 1299.9. These rows prevent two Real-facing consumers with the same source and seal rows from reading private threshold windows or incompatible downstream readbacks.

Finally use noninternalization, Definition 1299.12. The excluded quotient, limit, equality, trichotomy, completeness, private-threshold, and second-seal objects have no carrier coordinate, so the listed rows form the complete L10 obligation ledger before any Real-facing tail-window read. □

Theorem 1299.14 (Cauchy tail threshold encoded obligation readback)**.** *The carrier row of Definition 1299.1, the classifier row of Definition 1299.2, the route-exhaustion row of Definition 1299.4, the shared-threshold uniqueness row of Definition 1299.6, and the Real-facing determinacy row of Definition 1299.9 together provide the full finite obligation readback needed for the next Lean target over* BEDC.Derived.CauchyTailThresholdNormalizerUp*. The readback exports carrier admission, classifier comparison, route coverage, unique shared-threshold exposure, and Real-facing downstream determinacy, and it exports no quotient-stream equality, selected host limit, host real equality, trichotomy, ambient completeness, private side threshold, or second Real seal route.*

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerEncodedObligationRead`

Proof. Read the carrier and classifier first. The carrier theorem displays the rows $S, M, \Theta, W_0, W_1, D, R, A, E, H, C,$ while the classifier requires those rows to be compared componentwise over the same threshold surface.

Next apply route exhaustion. It forces every $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer through

$$S \longmapsto M \longmapsto \Theta \longmapsto (W_0, W_1, D, R) \longmapsto (A, E, H, C, P, L, N),$$

so the encoded target has a finite route to read rather than an ambient completion premise.

The constructor induction is over threshold-window rows. The base constructor reads Θ after the shared source and modulus rows; the successor constructors preserve the two StreamName windows, dyadic tolerance ledger, regular readback, agreement seal, Real row, transport, replay, provenance, route ledger, and local naming.

Finally use shared-threshold uniqueness and Real-facing determinacy. They prevent two classified reads with the same S, M rows from splitting into private thresholds, and they force W_0, W_1, D, R, A to remain downstream of the same Θ. These cited rows are exactly the finite obligation surface required by the Lean target. □

Falsifiable prediction: Removing the normalized shared-tail row permits two RealUp seal consumers to read incompatible finite windows through private thresholds.

Independence witness: cauchysharedtailmodulus, cauchytailmodulusseal, realtailagreementseal, regseqrat, real: not bijective because the normalizer carries the common threshold, both window readbacks, both seal-facing routes, and the route-exhaustion ledger simultaneously.

Lean follow-up provides explicit $\mathsf{Nontrivial}\ \mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CauchyTailThreshold}$ instances in `lean4/BEDC/Derived/CauchyTailThresholdNormalizerUp/TasteGate.lean`.

Theorem 1299.15 (Cauchy tail threshold normalizer obligation scope)**.** *The finite obligation surface of* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *is exactly the carrier row of Definition 1299.1, the classifier row of Definition 1299.2, route exhaustion from Definition 1299.4, tail-window induction from Definition 1299.5, shared-threshold uniqueness from Definition 1299.6, window cofinality from Definition 1299.7, Real-facing determinacy from Definition 1299.9, composition scope from Definition 1299.11, noninternalization from Definition 1299.12, and the obligation ledger of Definition 1299.13. These rows bind the obligationClosure surface over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without exporting quotient streams, selected host limits, host real equality, trichotomy, ambient completeness, private side thresholds, or a second Real seal route.*

Proof. Project the carrier and classifier first. They display the finite packet $S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L,$ and require classified reads to compare those rows componentwise over the same threshold surface.

Use route exhaustion, tail-window induction, shared-threshold uniqueness, window cofinality, and Real-facing determinacy to bind the analytic rows. These theorems force every $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RealTailAgreementSeal}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal read to pass through the same normalized threshold Θ.

Apply composition scope and noninternalization next. They show that the displayed packet composes as one finite route and that no quotient stream, selected host limit, host real equality, trichotomy, ambient completeness, private side threshold, or second seal row can be projected from it.

Finally read the obligation ledger. It names exactly the same rows as the finite L10 surface, so the chapter reaches obligationClosure on the paper axis while leaving later verification to kernel-scoped dependency wording over those rows. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerObligationScope`

Theorem 1299.16 (Cauchy tail threshold normalizer uniform-threshold induction)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *tail-window successor read of an accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier preserves the single normalized threshold row* Θ *across the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *handoff, and every* $\mathsf{Real}^{\uparrow}$ *consumer row. In particular, no successor window may replace the shared* S, M, Θ *surface by a private threshold before exposing* $W_0, W_1, D, R, A, E, H, C, P, L, N$.

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerUniformThresholdIndu`

Proof. Use the tail-window induction row of Definition 1299.5. Its base case reads Θ immediately after the shared source S and seal row M; its successor step preserves the displayed windows W_0, W_1 together with the dyadic, regular-rational, agreement-seal, and Real rows.

Apply shared-threshold uniqueness and window cofinality. These rows force every later $\mathsf{StreamName}^{\uparrow}$ window to remain downstream of the same Θ, and they prevent a $\mathsf{RealTailAgreementSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer from introducing a side threshold.

Finally read Real-facing determinacy and the obligation ledger. The ledger lists only $S, M, \Theta, W_0, W_1, D, R, A, E$ so the induction is uniform over all displayed downstream consumers and exports no quotient stream, selected host limit, trichotomy, ambient completeness, or second Real seal route. □

Theorem 1299.17 (Cauchy tail threshold normalizer Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal handoff of an accepted* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ *carrier factors first through*

$$S,\ M,\ \Theta,\ W_0,\ W_1,\ D,\ R,\ A$$

before any row among

$$E,\ H,\ C,\ P,\ L,\ N$$

is available to the consumer. Thus the Real-facing route reads the shared source, modulus seal, normalized threshold, two StreamName windows, dyadic tolerance ledger, RegSeqRat readback, and RealTailAgreementSeal handoff before it may expose the Real consumer row, structural transport, replay, provenance, route ledger, or local naming.

Proof. Start with route exhaustion from Definition 1299.4. It presents every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealTailAgreementSeal}^{\uparrow}$ consumer through the same finite row order and prohibits a detached Real seal.

Use Real-facing determinacy next. Once S and M are fixed, the windows W_0, W_1, dyadic row D, regular readback R, and agreement handoff A are all read downstream of Θ; the $\mathsf{Real}^{\uparrow}$ consumer row E cannot precede that threshold path.

The composition and noninternalization rows then keep H, C, P, L, N as transport, replay, provenance, ledger, and naming data for the same route. They do not create an additional seal handoff, so every Real-facing consumer factors through $S, M, \Theta, W_0, W_1, D, R, A$ before exposing E, H, C, P, L, N. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerRealSealHandoff`

Theorem 1299.18 (Cauchy tail threshold normalizer scoped kernel route)**.** *Every accepted* CauchyTailThresholdNor *carrier*

$$T = (S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N)$$

scopes its public threshold-normalizer route over exactly BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* StreamName$^\uparrow$*,* RegSeqRat$^\uparrow$*,* DyadicRatCore$^\uparrow$*,* RealTailAgreementSeal$^\uparrow$*, and* Real$^\uparrow$*. Any* Real$^\uparrow$*-facing read must expose those rows before using the normalized threshold* Θ*, and the scoped route exports no quotient stream, selected host limit, trichotomy, ambient completeness, side threshold, or second* Real$^\uparrow$ *seal.*

Proof. Project the carrier and route-exhaustion rows. The accepted packet displays $S, M, \Theta, W_0, W_1, D, R, A, E, H,$ and Definition 1299.4 forces every public read through that finite route.

Use tail-window induction, shared-threshold uniqueness, and window cofinality. These theorems keep both StreamName$^\uparrow$ windows and the DyadicRatCore$^\uparrow$ tolerance ledger downstream of the same normalized threshold, so no side threshold can be selected by a later consumer.

Apply Real-facing determinacy and the Real-seal handoff. They require the RegSeqRat$^\uparrow$ readback and RealTailAgreementSeal$^\uparrow$ row to be exposed before the terminal Real$^\uparrow$ seal, and they prevent a second seal route from being projected out of the carrier.

Finally read transport, replay, provenance, ledger, and local naming. The rows H, C, P, L, N are exactly the hsame, Cont, Pkg, route-ledger, and NameCert boundary over the same finite packet, so the scoped route has no quotient stream, selected host limit, trichotomy, ambient completeness, or hidden threshold coordinate to export. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerScopedKernelRoute`

Theorem 1299.19 (Cauchy tail threshold normalizer public certificate)**.** *Every public* Real$^\uparrow$*,* RealTailAgreementSeal$^\uparrow$*,* StreamName$^\uparrow$*,* RegSeqRat$^\uparrow$*, or* DyadicRatCore$^\uparrow$ *consumer of an accepted* CauchyTailThresholdNormalizer$^\uparrow$ *packet receives exactly the scoped route*

$$S,\ M,\ \Theta,\ W_0,\ W_1,\ D,\ R,\ A,\ E,\ H,\ C,\ P,\ L,\ N$$

from Definition 1299.18. The public certificate exposes the same shared threshold Θ*, the two window rows* W_0, W_1*, the dyadic tolerance ledger* D*, the regular-rational readback* R*, the agreement-seal handoff* A*, the Real row* E*, and the structural rows* H, C, P, L, N*; it exports no quotient stream, selected host limit, trichotomy, ambient completeness, private side threshold, or second Real seal route.*

Proof. Start with the scoped kernel route of Definition 1299.18. It already fixes the accepted packet as the finite row list $S, M, \Theta, W_0, W_1, D, R, A, E, H, C, P, L, N$ over BHist, hsame, Cont, Pkg, NameCert, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, RealTailAgreementSeal$^\uparrow$, and Real$^\uparrow$.

For a Real$^\uparrow$ or RealTailAgreementSeal$^\uparrow$ consumer, apply the Real-seal handoff of Definition 1299.17. The handoff requires $S, M, \Theta, W_0, W_1, D, R, A$ before the public Real-facing row is available, and the remaining structural rows E, H, C, P, L, N are read only as part of the same route.

For a $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{DyadicRatCore}^{\uparrow}$ consumer, use window cofinality from Definition 1299.7. It keeps both window rows and the dyadic tolerance ledger downstream of the shared threshold, so the regular readback and agreement seal cannot be separated from S, M, Θ.

The public certificate is therefore only a consumer-facing packaging of the scoped route. Since each admissible public read is obtained by composing these rows, no consumer can add a quotient stream, selected host limit, trichotomy, ambient completeness, private side threshold, or second Real seal route without leaving the displayed packet. □

Lean4 ✓ `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerPublicCertificate`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ packet is generated from two regular-Cauchy tail presentations by making the common tail threshold explicit before any agreement or real-seal read is allowed. The finite BHist carrier stores the threshold normalizer, the tail-agreement row, the eventual-equality row, componentwise hsame transport, displayed Cont routes, package provenance, ledger entries, and local naming. Its classifier reads the same threshold surface in both presentations and rejects side thresholds or second real-seal routes. The constructive content is the normalization discipline: every downstream RealUp or agreement read is generated bottom-up from the displayed threshold and its replay ledger, not from a quotient stream or hidden host limit.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: This chapter binds Definition 1299.18: the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows of the finite $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$ kernel route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyTailThresholdNormalizerUp.CauchyTailThresholdNormalizerPublicCertificate`

Bridge status: none

Not claimed: It does not claim quotient-stream equality, selected host limits, trichotomy, ambient completeness, private side thresholds, or a second Real seal route.

Upgrade path: Bridged closure requires a standard bridge over the same public route, including its StreamName, RegSeqRat, DyadicRatCore, RealTailAgreementSeal, Real, BHist, transport, replay, provenance, and local naming rows.

Chapter 1300

A Concrete Naming Certificate for $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$

The $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ packet realizes the total phenomenology and total philosophy-of-science synthesis of section 9462.31 as one finite NameCert interface. It sits beside the science bridge of chapter 1286, the observer-history identity surface of chapter 479, the inter-history locality ledger of chapter 759, the scientific-object carrier of chapter 1214, and the standard-bridge audit packet of chapter 1217. Those siblings expose science-facing object rows, no-subject observer identity, finite locality ledgers, empirical object structure, and conservative bridge rows; this chapter binds them before any consumer treats phenomenological record access and scientific object access as the same accepted public surface.

Definition 1300.1 (Phenomenology-science interface carrier)**.** *A* $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, U, O, L, S, J, B, G, H, C, P, N)$$

whose displayed rows have the following roles:

1. *R is the records-side conscious packet read through focused records, observer identity, and local continuation rows.*
2. *U is the finite universe-slice row, exposing only the histories admitted by the current ledger.*
3. *O is the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *row that keeps observer identity on the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *surface.*
4. *L is the* $\mathsf{InterHistLocalityLedger}^{\uparrow}$ *row recording local spatial and inter-history relation data.*
5. *S is the* $\mathsf{ScientificObject}^{\uparrow}$ *row carrying record, classifier, invariant, empirical ledger, bridge, gap report, audit, truth, and local naming data.*
6. *J is the* $\mathsf{ScienceBridge}^{\uparrow}$ *row that reads the scientific-object, physical-audit, truth, bridge, and gap/failure surfaces together.*
7. *B is the* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *row exposing conservative bridge encoding, decoding, round-trip, theorem-preservation, and no-host-leak data.*
8. *G is the gap ledger for hard-problem, other-minds, causality, freedom, self-reference, prediction, domain, bridge, and falsification reports that remain visible to the interface.*

9. *H records componentwise* hsame *transport for $R, U, O, L, S, J, B, G, C, P, N$.*
10. *C records displayed* Cont *replay for records-side reading, universe-slice restriction, locality transport, scientific-object audit, bridge audit, and gap-ledger export.*
11. *P is the* Pkg *provenance over* $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$, $\mathsf{ScienceBridge}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PhenomenologyScienceInterface}^{\uparrow}}$ *row.*

The carrier has no coordinate for private essence, global universe, global time, host real equality, unrecorded scientific object, or bridge theorem outside the conservative audit row.

Lean4 (definition) `BEDC.Derived.PhenomenologyScienceInterfaceUp.taste gate`

Theorem 1300.2 (Phenomenology-science observation slice)**.** *Every accepted* $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ *observation read factors through the records-side row R, the finite universe slice U, the observer-history identity row O, the locality ledger L, and the structural rows H, C, P, N. A read that omits one of these rows may still be a projection of a conscious record or a scientific object, but it is not the interface carrier of Definition 1300.1.*

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterfaceObservationSlice`

Proof. Project the carrier $I = (R, U, O, L, S, J, B, G, H, C, P, N)$. The records-side row gives the focused conscious packet, while U restricts the read to the finite observation-history slice admitted by the ledger.

Next read observer identity and locality. The $\mathsf{ObserverHistIdentity}^{\uparrow}$ row keeps the observer on the displayed BHist side, and the $\mathsf{InterHistLocalityLedger}^{\uparrow}$ row supplies the finite local relation surface. These two rows prevent the observation slice from becoming a hidden subject or a global container.

Finally use H, C, P, N. Transport preserves the same rows componentwise, continuation replay follows the displayed routes, provenance records the sibling sources, and local naming registers the interface. Therefore an accepted observation slice must keep R, U, O, L, H, C, P, N visible. □

Theorem 1300.3 (Phenomenology-science invariant transport)**.** *If two accepted interface carriers are related by the componentwise* hsame *row H, then the scientific-object row S, the science-bridge row J, the locality ledger L, and the observer-history identity row O transport together. Invariant content may be read as scientific only through the same finite locality and observer-identity support used by the records-side slice.*

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterface invariant transport`

Proof. Begin with the componentwise transport coordinate H. Its domain is the whole carrier, so it preserves the records-side, universe-slice, observer-identity, locality, scientific-object, bridge, gap, replay, provenance, and naming rows as one packet.

Apply the sibling readings named in the chapter introduction. $\mathsf{ScientificObject}^{\uparrow}$ supplies invariant and empirical object rows; $\mathsf{ScienceBridge}^{\uparrow}$ supplies the bridge-facing scientific synthesis; $\mathsf{InterHistLocalityLedger}^{\uparrow}$ supplies locality support; and $\mathsf{ObserverHistIdentity}^{\uparrow}$ supplies no-subject observer identity.

The transported scientific read is therefore not detached from the phenomenological slice. It remains supported by O, L, H, C, P, N, so a consumer cannot export scientific invariance while dropping the finite observer and locality rows that make the record slice readable. □

Theorem 1300.4 (Phenomenology-science gap ledger)**.** *The gap row G of an accepted* PhenomenologyScienceInterfa *carrier is mandatory for every interface export that mentions hard-problem excess, other-minds access, causal direction, freedom, self-reference, prediction, domain fit, bridge fit, or falsification. None of the rows $R, U, O, L, S, J, B, H, C, P, N$ may absorb G.*

Proof. Read the carrier inventory. The records-side and universe rows expose finite observation data; the observer and locality rows expose identity and inter-history support; the scientific-object and bridge rows expose empirical and conservative bridge surfaces.

The listed gap forms are not coordinates of those rows. Hard-problem and other-minds reports are not observer identity; prediction, domain, bridge, and falsification reports are not scientific-object naming; bridge fit is not no-host-leak by itself.

Thus G remains the displayed ledger for unresolved or refusal-bearing parts of the interface. Transport, continuation, provenance, and local naming preserve the ledger but do not create it after omission. □

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterface gap ledger`

Theorem 1300.5 (Phenomenology-science NameCert obligations)**.** *For the carrier of Definition 1300.1, the local* $\mathsf{NameCert}_{\mathsf{PhenomenologyScienceInterface}^{\uparrow}}$ *surface consists exactly of carrier admission for I, records-side exposure through R, universe-slice exposure through U, observer-identity exposure through O, locality-ledger exposure through L, scientific-object exposure through S, science-bridge exposure through J, conservative bridge exposure through B, gap-ledger exposure through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterfaceNameCertObligations`

Proof. Project I into its twelve rows. The first four analytic rows R, U, O, L supply finite conscious-record reading, universe-slice restriction, no-subject observer identity, and local inter-history relation data.

The science-facing rows S, J, B then expose scientific-object, science-bridge, and conservative standard-bridge audit data. The gap row G remains separate by Definition 1300.4; it is not discharged by object naming or bridge audit.

The structural rows finish the certificate. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming keep the same finite packet available to downstream consumers.

Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. A proposed interface that supplies only phenomenology rows or only science rows is a projection, not the accepted $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ packet. □

Theorem 1300.6 (Phenomenology-science conservative bridge obligations)**.** *Every bridge-facing export of an accepted* $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ *packet must expose the standard-bridge audit row B together with the scientific-object row S, the science-bridge row J, the gap ledger G, transport H, continuation replay C, provenance P, and local naming N. The export may use conservative bridge rows, but it cannot treat host primitives, private phenomenological essence, or a global universe as public BEDC data.*

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterfaceConservativeBridgeObl`

Proof. Use Definition 1300.5. The NameCert surface lists B as the bridge audit row and keeps it adjacent to S, J, G, H, C, P, N.

The standard-bridge audit packet supplies encoding, decoding, round-trip, theorem-preservation, and no-host-leak obligations. These obligations do not identify host primitives with BEDC rows; they only record a conservative bridge read.

The records-side and universe-slice rows remain finite, and the gap ledger remains visible. Therefore the bridge-facing export is accepted only as a conservative interface row over the displayed packet, not as a private essence, global universe, or host-theoretic replacement. □

Falsifiable prediction: A proposed phenomenology/science interface that exports scientific object maturity while omitting observer identity, locality ledger, gap ledger, or standard bridge audit is not an accepted PhenomenologyScienceInterfaceUp carrier.

Independence witness: sciencebridge, observerhistidentity, interhistlocalityledger, scientificobject, standardbridgeauditpacket: none is bijective because $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ binds records-side consciousness, finite universe slice, observer identity, locality, scientific object, science bridge, standard bridge audit, gap ledger, transport, replay, provenance, and local naming in one carrier. Each listed sibling omits at least one mandatory interface row.

Theorem 1300.7 (Phenomenology-science TasteGate grounding)**.** *Any promotion of* PhenomenologyScienceInterfa *beyond seed closure must read the observation slice R, U, O, L, invariant transport over S, J, H, the gap ledger G, the conservative bridge rows B, J, and the Lean-side TasteGate carrier-alignment target as separate obligations. None of these obligations may be discharged merely by naming the composite carrier $I = (R, U, O, L, S, J, B, G, H, C, P, N)$.*

Proof. Inspect the carrier rows. The observation slice is carried by R, U, O, L, while scientific invariance and bridge-facing exports require S, J, B together with transport H and replay C.

The gap ledger G is independent of those rows by Definition 1300.4. It records hard-problem, other-minds, causal, freedom, self-reference, prediction, domain, bridge, and falsification reports that cannot be absorbed by object naming or bridge audit.

The AI growth roadmap discipline requires the TasteGate side to align the carrier before the chapter can move beyond seed status. Therefore a later promotion must exhibit the paper rows and the Lean carrier-alignment obligation separately, with provenance and local naming still visible through P, N. □

Theorem 1300.8 (Phenomenology-science interface obligation carrier)**.** *Every accepted* PhenomenologyScienceInter *carrier exposes the dependency rows* $\mathsf{InterHistLocalityLedger}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{ScienceBridge}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, *and* $\mathsf{StandardBridgeAuditPacket}^{\uparrow}$ *through one finite* BHist *packet. The public carrier rows are exactly $R, U, O, L, S, J, B, G, H, C, P, N$, where the displayed R, U, O, L route the observational side, S, J, B route the scientific and bridge side, and H, C, P, N preserve the accepted packet.*

Proof. Project the carrier of Definition 1300.1. The rows $R, U, O, L, S, J, B, G, H, C, P, N$ are the only public coordinates, so each dependency surface is already stored inside the same finite packet.

Replay the displayed BHist route. The observation slice remains on R, U, O, L, while the scientific-object and bridge rows stay on S, J, B; transport, continuation, provenance, and local naming keep those routes readable as the same accepted packet. □

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterface obligation carrier`

Theorem 1300.9 (Phenomenology-science interface obligation classifier)**.** *The classifier for an accepted* PhenomenologyScienceInterface$^{\uparrow}$ *carrier identifies the same finite rows after transporting the records-side and scientific-side routes through* hsame *and replaying them through* Cont*. In particular, the classifier sees the* InterHistLocalityLedger$^{\uparrow}$ *row, the* ObserverHistIdentity$^{\uparrow}$ *row, the* ScienceBridge$^{\uparrow}$ *row, the* ScientificObject$^{\uparrow}$ *row, and the* StandardBridgeAuditPacket$^{\uparrow}$ *row only as displayed dependencies of the same carrier, not as separate choice points.*

Lean4 ✓ `BEDC.Derived.PhenomenologyScienceInterfaceUp.PhenomenologyScienceInterfaceObligationClassifier`

Proof. Project the carrier and read the classifier surface. The rows R, U, O, L and S, J, B are already separated in the definition, so the classifier only compares how those rows are displayed in the same packet.

Replay the hsame and Cont routes. The two routes preserve the same dependency rows, and the local Pkg provenance together with NameCert records that no hidden classifier branch was introduced outside the accepted packet. □

Theorem 1300.10 (Phenomenology-science interface obligation ledger)**.** *The local ledger of an accepted* PhenomenologyScienceInterface$^{\uparrow}$ *carrier records exactly the dependency rows* InterHistLocalityLedger$^{\uparrow}$*,* ObserverHistIdentity$^{\uparrow}$*,* ScienceBridge$^{\uparrow}$*,* ScientificObject$^{\uparrow}$*, and* StandardBridgeAuditPacket$^{\uparrow}$ *together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route. No ledger row exports a hidden subject, a global universe, a detached scientific-object authority, or a bridge claim that bypasses the displayed packet.*

Proof. Project the carrier one last time. The dependency rows named in the statement are already present in the packet, and the structural rows H, C, P, N keep them visible as one finite ledger.

Read the route through the displayed packet rather than by an external authority. The BHist carrier, componentwise hsame transport, displayed Cont replay, package provenance, and local naming are sufficient to replay the dependency ledger without adding any hidden subject or detached bridge claim. □

Theorem 1300.11 (Phenomenology-science TasteGate obligation scope)**.** *Every accepted* PhenomenologyScienceIn *packet must expose records-side consciousness R, finite universe slice U, observer-history identity O, inter-history locality L, scientific-object row S, science-bridge row J, standard-bridge audit row B, gap ledger G, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before any interface read. The TasteGate carrier-alignment obligation may be read only after the sibling routes cited in the chapter introduction have supplied* ObserverHistIdentity$^{\uparrow}$*,* InterHistLocalityLedger$^{\uparrow}$*,* ScientificObject$^{\uparrow}$*,* ScienceBridge$^{\uparrow}$*, and* StandardBridgeAuditPacket$^{\uparrow}$ *as exact source rows.*

Proof. Project the carrier of Definition 1300.1. The rows $R, U, O, L, S, J, B, G, H, C, P, N$ are the full public interface, so an accepted packet cannot expose a phenomenology-facing read or a science-facing read before all twelve coordinates are visible.

Read the records-side scope first. The rows R, U, O, L bind conscious-record access to the finite universe slice, observer-history identity, and inter-history locality ledger, matching the sibling routes named in the introduction.

Read the science and bridge scope next. The rows S, J, B must arrive from ScientificObject$^{\uparrow}$, ScienceBridge$^{\uparrow}$, and StandardBridgeAuditPacket$^{\uparrow}$, while G remains the displayed ledger for hard-problem, other-minds, causality, freedom, self-reference, prediction, domain, bridge, and falsification reports.

The structural rows close the obligation boundary. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming keep the same sibling-source packet available before the TasteGate carrier-alignment read. Thus a carrier-only paragraph, a science-only row, or a records-only row cannot move the interface beyond seed scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{PhenomenologyScienceInterface}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$ packet is generated from records-side consciousness, finite universe slice, observer-history identity, inter-history locality, scientific object, science bridge, standard bridge audit, gap ledger, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhenomenologyScienceInterface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1300.1 and Definition 1300.11 record the composite interface carrier and its exact records-side, science-side, gap-ledger, transport, continuation, provenance, and naming scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhenomenologyScienceInterface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, global universe construction, host-theory identity, private essence, or scientific completeness.

Upgrade path: Obligation closure requires binding each carrier row to its exact sibling source, classifier, transport, continuation, provenance, and local naming obligation.

Chapter 1301

A Concrete Naming Certificate for CompactLebesgueUniformHandoff$^{\uparrow}$

The CompactLebesgueUniformHandoff$^{\uparrow}$ packet realizes the finite radius-ledger bridge isolated by chapter 9237. It binds the Lebesgue-number ledger of chapter 452, the compact-net selector of chapter 914, the uniform-modulus consumer of chapter 369, the compact metric source of chapter 129, and the pointwise continuity rows of chapter 130. The coordinate added here is the single public handoff row that makes the finite Lebesgue radius ledger readable by the compact-net selector before the UniformModulus$^{\uparrow}$ consumer accepts a tolerance-only modulus packet.

Definition 1301.1 (Compact Lebesgue uniform handoff carrier)**.** *A* CompactLebesgueUniformHandoff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, S, R, M, C, H, T, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row carrying the finite compact-metric witness consumed by the compact-uniform route;*
2. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row over the same source and target metric carriers;*
3. *S is the* FiniteLebesgueNumber$^{\uparrow}$ *positive radius ledger, read through* DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, *and* DyadicMesh$^{\uparrow}$ *rows;*
4. *R is the compact-net selector packet supplied by* CompactNetModulusSelector$^{\uparrow}$, *with its finite* ProbeBundle *centers and radius-route witnesses;*
5. *M is the* UniformModulus$^{\uparrow}$ *output row that may read a tolerance-only precision history only after K, F, S, and R have been displayed;*
6. *C records the displayed* Cont *routes from compact source, pointwise modulus, Lebesgue radius ledger, and selector packet into the uniform-modulus row;*
7. *H records componentwise* hsame *transport for the compact source, continuous-map row, radius ledger, selector packet, uniform-modulus output, continuation, provenance, and local name rows;*
8. *T is the handoff trace row stating that the selector consumes exactly the positive finite Lebesgue-number ledger before the uniform-modulus row is public;*

9. *P is the* Pkg *provenance over* FiniteLebesgueNumber$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, UniformModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, DyadicMesh$^{\uparrow}$, BHist, hsame, Cont, *and local certificate rows;*
10. *N is the local* NameCert$_{\text{CompactLebesgueUniformHandoff}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a host minimum, a selected subcover, quotient equality of source points, or an external modulus function.

Theorem 1301.2 (Compact Lebesgue uniform handoff NameCert obligations)**.** *For the carrier of Definition 1301.1, the local* NameCert$_{\text{CompactLebesgueUniformHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for $U = (K, F, S, R, M, C, H, T, P, N)$, compact metric source exposure through K, pointwise continuous-map modulus exposure through F, finite Lebesgue-number radius-ledger exposure through S, compact-net selector exposure through R, uniform-modulus output exposure through M, displayed continuation through C, componentwise* hsame *transport through H, handoff trace exposure through T, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. The rows $K, F, S, R, M, C, H, T, P, N$ list the full compact-uniform route before any consumer can read the uniform-modulus output.

The source and pointwise rows are read first. The compact source row K supplies the finite compact-metric context, while F supplies the pointwise-modulus relation over that same source route.

The radius ledger and selector rows form the handoff. The finite Lebesgue-number row S exposes only the dyadic cover, finite-window, rational-ball, and mesh ledgers, and the selector row R reads that positive radius ledger as its compact-net input.

The output row is therefore downstream of the displayed handoff trace T. The transport, continuation, provenance, and local naming rows add no open-cover compactness, host minimum, selected subcover, quotient equality, or external modulus coordinate, so the listed rows exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.CompactLebesgueUniformHandoffUp.TasteGate.CompactLebesgueUniformHandoffTasteGate single carrier alignment`

Theorem 1301.3 (Compact Lebesgue uniform handoff factorization)**.** *Every accepted* UniformModulus$^{\uparrow}$ *read from a* CompactLebesgueUniformHandoff$^{\uparrow}$ *carrier factors through the ordered route*

CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, UniformModulus$^{\uparrow}$

The factorization reads the finite Lebesgue-number positive radius ledger through DyadicCover$^{\uparrow}$, FiniteWindowEnvelope$^{\uparrow}$, RationalBall$^{\uparrow}$, *and* DyadicMesh$^{\uparrow}$, *then hands that ledger to the compact-net selector packet, and finally exposes the uniform-modulus output through the carried* hsame, Cont, Pkg, *and* NameCert *rows.*

Proof. Start with the NameCert obligations of Definition 1301.2. They identify $K, F, S, R, M, C, H, T, P, N$ as the complete consumer-facing surface.

The finite Lebesgue-number row S is the only positive radius ledger in the carrier. By chapter 452, its public read is limited to the displayed dyadic-cover, finite-window, rational-ball, mesh, transport, continuation, provenance, and local naming rows.

The selector row R is therefore forced to consume that finite ledger rather than an open-cover compactness object. The route through chapter 914 packages the compact-net centers and radius comparisons before the UniformModulus$^{\uparrow}$ output row M is public.

Finally the compact source K and pointwise continuous-map row F provide the source and local modulus data used by the selector. Since the carrier has no coordinate for host minima, selected subcovers, quotient equality, or external modulus functions, every accepted uniform-modulus read factors through the displayed route and no wider one. □

Falsifiable prediction: Every accepted CompactLebesgueUniformHandoffUp consumer must expose the finite Lebesgue-number radius ledger, compact-net selector packet, compact metric source, continuous-map pointwise-modulus row, uniform-modulus output, hsame, Cont, Pkg, and NameCert rows before a uniform-modulus handoff is public.

Independence witness: finitelebesguenumber, compactnetmodulusselector, uniformmodulus, compactmetric, continuousmap: no sibling is bijective because the handoff binds the radius ledger, selector route, source compactness, pointwise continuity, and output modulus as one packet.

The later Lean TasteGate.lean surface must include explicit Nontrivial CompactLebesgueUniformHandoff$^{\uparrow}$ and FieldFaithful CompactLebesgueUniformHandoff$^{\uparrow}$ instances.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLebesgueUniformHandoff}^{\uparrow}}$.

Constructive story: A CompactLebesgueUniformHandoff$^{\uparrow}$ packet is generated by placing compact finite-net data, pointwise continuous-map radii, a Lebesgue-style radius selector, a uniform-modulus output row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming into one finite BHist carrier. The classifier does not search an ambient open cover; it only inspects the displayed compact source and the local modulus rows already carried by the packet. The ledger story is the handoff from local to uniform: the selector records a finite radius choice and the uniform-modulus row may consume it only through the same transport and continuation routes.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactLebesgueUniformHandoff}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition, the NameCert-obligation and factorization theorems, the falsifiable prediction, independence witness, and Lean-follow-up boundary for CompactLebesgueUniformHandoff$^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactLebesgueUniformHandoff}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: It does not claim open-cover compactness, host minima, selected subcovers, quotient equality of points, or an external modulus function.

Upgrade path: To reach obligationClosure, the compact finite-net handoff and uniform-modulus factorization must be represented as checked finite-carrier obligations.

Chapter 1302

A Concrete Naming Certificate for CompactCoverLebesgueLedger$^{\uparrow}$

The CompactCoverLebesgueLedger$^{\uparrow}$ packet materializes the finite-cover lower-bound ledger used by chapter 9237. It sits after the compact-uniform-continuity packet of chapter 388, the finite rational lower-bound fold of chapter 619, the compact-metric finite-net surface, the continuous-map pointwise-radius rows, and the uniform-modulus consumer. Its purpose is narrower than the cover-shrinkage ledger: it records the concrete Lebesgue-number-style row that joins a finite ProbeBundle cover, positive Rat$^{\uparrow}$ radius rows, and the final UniformModulus$^{\uparrow}$ handoff.

Definition 1302.1 (Compact cover Lebesgue ledger carrier)**.** *A* CompactCoverLebesgueLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, C, R, F, U, H, T, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, exposing the nonempty finite* ProbeBundle *of centers and the cover-membership rows;*
2. *C is the* ContinuousMap$^{\uparrow}$ *pointwise-radius ledger, assigning to each displayed center a positive local radius for the requested target tolerance;*
3. *R is the positive* Rat$^{\uparrow}$ *radius row, recording the half-radius comparisons and the lower-bound candidates consumed by the fold;*
4. *F is the* FiniteNetMinimumFold$^{\uparrow}$ *lower-bound row over the same finite* ProbeBundle *and the same positive radius ledger R;*
5. *U is the* UniformModulus$^{\uparrow}$ *consumer row that receives the precision exported by the finite lower-bound fold;*
6. *H records componentwise* hsame *transport for the compact net, pointwise-radius ledger, rational rows, finite fold, uniform-modulus row, routes, provenance, and local naming;*
7. *T records the displayed* Cont *routes through which compact finite nets, continuous pointwise radii, rational comparisons, fold readback, and uniform-modulus consumers inspect the packet;*
8. *P is the* Pkg *provenance over* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Rat$^{\uparrow}$, FiniteNetMinimumFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactCoverLebesgueLedger}^{\uparrow}}$ *row.*
The carrier has no coordinate for an open-cover compactness oracle, host minimum over radii, selected center outside the finite bundle, quotient metric equality, or external modulus function.

Theorem 1302.2 (Compact cover Lebesgue ledger classifier surface)**.** *The classifier of an accepted* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *packet is determined by exactly the rows $K, C, R, F, U, H, T, P, N$. Two presentations with the same compact finite-net row, pointwise-radius row, positive rational ledger, lower-bound fold, uniform-modulus output, componentwise transports, displayed routes, provenance, and local naming row determine the same Lebesgue-ledger classifier, and no presentation may refine the classifier by a host minimum, open-cover oracle, external center selector, quotient metric equality, or external modulus function.*

Proof. Project both presentations to the carrier of Definition 1302.1. Equality of the nine displayed rows identifies the compact source, local radius ledger, rational lower-bound candidates, fold output, and uniform-modulus consumer.

The transport, route, provenance, and naming rows are also shared. They replay the same reads and cannot introduce a new classifier coordinate.

Every excluded object is absent from the carrier inventory. Therefore the classifier is exhausted by the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerCarrier classifier surface`

Theorem 1302.3 (Compact cover Lebesgue ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *packet*

$$L = (K, C, R, F, U, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactCoverLebesgueLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for L, compact finite-net admission through K, continuous pointwise-radius admission through C, positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound candidates through R, finite lower-bound fold exactness through F, uniform-modulus output exactness through U, componentwise* hsame *stability through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1302.1. The rows $K, C, R, F, U, H, T, P, N$ are the complete displayed packet, and the first five rows name the compact net, pointwise radii, rational candidates, finite fold, and uniform-modulus consumer.

The compact finite-net and continuous pointwise-radius obligations are read before the fold. Thus every lower-bound candidate in R is attached to a center already visible in K and to a pointwise-radius row already visible in C.

The fold row F is exact over that same finite bundle and radius ledger. Its exported precision is the only row that U may read as a $\mathsf{UniformModulus}^{\uparrow}$ precision.

Finally H, T, P, N transport, replay, package, and name the same finite surface. Since no excluded coordinate occurs in the carrier, no local NameCert obligation can appeal to an open-cover oracle, host minimum, external selected center, quotient metric equality, or external modulus function. □

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerCarrier namecert obligations`

Theorem 1302.4 (Compact cover Lebesgue ledger uniform-modulus handoff)**.** *Every accepted* CompactCoverLebesgueLedger$^{\uparrow}$ *consumer that exports a* UniformModulus$^{\uparrow}$ *precision must factor through the compact finite-net row K, the continuous pointwise-radius row C, the positive* Rat$^{\uparrow}$ *lower-bound ledger R, the* FiniteNetMinimumFold$^{\uparrow}$ *row F, and the structural rows H, T, P, N. A consumer cannot export the uniform modulus without the compact finite-net row, continuous pointwise-radius row, positive lower-bound fold,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with the compact finite-net row K. It supplies the finite ProbeBundle of centers used by the compact-uniform-continuity construction of chapter 9237.

Read the continuous pointwise-radius row C at those centers. The positive rational row R records the half-radius candidates whose comparisons make the local pointwise estimates usable by a common precision.

Apply the lower-bound fold F. Because F is a FiniteNetMinimumFold$^{\uparrow}$ row over the same finite bundle and the same rational ledger, its exported precision is below every displayed local radius and is available as a single tolerance-only row.

The UniformModulus$^{\uparrow}$ row U can read that precision only through H, T, P, N. These structural rows preserve and route the finite ledger; they do not add a direct pointwise-to-uniform selector or a hidden compactness theorem. □

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerCarrier uniform modulus handoff`

Theorem 1302.5 (Compact cover Lebesgue ledger finite-net radius exhaustion)**.** *Every uniform precision exported by an accepted* CompactCoverLebesgueLedger$^{\uparrow}$ *packet is exhausted by the displayed finite-net radius ledger. Concretely, if*

$$L = (K, C, R, F, U, H, T, P, N)$$

exports a precision to U, then the export is read from the finite net row K, the pointwise radius row C, the positive rational ledger R, the lower-bound fold F, and the structural rows H, T, P, N. No exported precision may depend on an open-cover oracle, a host minimum over radii, a selected center outside K, quotient metric equality, or an external modulus function.

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerFiniteNetRadiusExhaustion`

Proof. Project the carrier through Definition 1302.1. The only rows that can supply radius data are K, C, R, F: the finite net names the displayed centers, the pointwise ledger attaches local radii to those centers, the rational row records positive lower-bound candidates, and the fold row reads the same finite list.

Apply Definition 1302.4. It places every UniformModulus$^{\uparrow}$ export downstream of K, C, R, F and the structural rows H, T, P, N.

The classifier-surface theorem then excludes every other source of precision. Since the carrier has no coordinate for an open-cover compactness oracle, host minimum, external center selector, quotient metric equality, or external modulus function, the finite-net ledger exhausts the exported precision. □

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerCarrier finite net radius exhaustion`

Falsifiable prediction: A consumer that exports a UniformModulus$^{\uparrow}$ precision from a compact-continuous packet without reading the compact finite-net row, continuous pointwise-radius row,

positive $\mathsf{Rat}^{\uparrow}$ lower-bound fold, hsame, Cont, Pkg, and NameCert rows refutes the $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ carrier.

Independence witness: compactuniformcontinuity, finitenetminimumfold, compactnetmodulus-selector, compactmoduluscover, uniformmodulus: $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ binds the compact finite-net row, continuous pointwise-radius row, positive rational lower-bound fold, uniform-modulus output, and structural transport rows in one finite NameCert packet; each named sibling lacks at least one of these rows as a local carrier obligation.

Lean follow-up for this surface must provide Nontrivial and FieldFaithful instances in `BEDC.Derived.CompactCov`

Lean4 ✓ `BEDC.Derived.CompactCoverLebesgueLedgerUp.CompactCoverLebesgueLedgerTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCoverLebesgueLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ packet is generated from a compact finite-net row, continuous pointwise-radius rows, positive rational lower-bound candidates, a finite minimum-fold row, the resulting uniform-modulus output, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier is exhausted by these rows: it reads the finite ProbeBundle centers, their local radii, and the lower-bound fold before any uniform consumer may inspect a precision. The ledger story is a finite Lebesgue-number account, where the cover evidence is not an oracle but a displayed list whose fold and replay routes can be audited in the packet itself.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCoverLebesgueLedger}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: This chapter gives the mature paper-side finite-net Lebesgue-ledger route witnessed by Definition 1302.4 and Definition 1302.5, carrying $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows from the compact cover surface to the uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCoverLebesgueLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: It does not claim an open-cover compactness oracle, host minimum over radii, selected center outside the finite bundle, quotient metric equality, or external modulus function.

Upgrade path: Lean formalization and bridge checking remain separate verification-axis work for the finite-net Lebesgue-ledger route.

Chapter 1303

A Concrete Naming Certificate for $\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}$

The $\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}$ packet realizes operational descent from section 9414.1 as a finite NameCert carrier. It consumes the signature-fit packet of **??**, the signature-residue packet of chapter 1213, the finite schedule surface of chapter 126, the endpoint surface of chapter 38, and the comparison surface of chapter 57. These siblings expose physical fit, remaining signature residue, stream schedules, real approximation rows, and metric comparison rows; this packet adds the descent witness that lets a transported mathematical operation be read as a physical-signature operation.

Definition 1303.1 (Reality-constrained signature-descent carrier)**.** *A* $\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, M, G, P, O, W, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the observation stream on which the operation is tested.*
2. *A is the* $\mathsf{Real}^{\uparrow}$ *approximation row naming the mathematical endpoint before and after operation transport.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used to evaluate transported outputs.*
4. *G is the GAP row recording unrepresented, withheld, or physically separated signatures.*
5. *P is the physical-representation certificate row for the source and target representations.*
6. *O is the operation-descent witness row recording that the transported operation preserves the source classifier and lands in the target physical-signature classifier.*
7. *W is the witness route joining S, A, M, G, P, O as one physical-signature descent route.*
8. *H is the componentwise* hsame *transport row for S, A, M, G, P, O, W.*
9. *C is the displayed* Cont *replay row for the same finite route.*
10. *Q is the* Pkg *provenance row over* $\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureResidue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}}$ *row.*
The carrier has no coordinate for bare mathematical operation transport, physical adequacy without S, G, P, O, quotient-stream equality, host physical identity, or an unledgered representation shortcut.

Lean4 (definition) `BEDC.Derived.RealityConstrainedSignatureDescentUp.RealityConstrainedSignatureDescentUp`

Theorem 1303.2 (Reality-constrained signature-descent NameCert obligations)**.** *For the carrier of Definition 1303.1, the local* $\mathsf{NameCert}_{\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}}$ *surface consists exactly of carrier admission, stream schedule visibility, Real endpoint visibility, Metric comparison visibility, GAP-row visibility, representation-certificate visibility, operation-descent witness visibility, witness-route coherence, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N.*

Proof. Project the carrier $D = (S, A, M, G, P, O, W, H, C, Q, N)$. The rows S, A, M, G, P expose the mathematical and physical-signature data already separated by the signature-fit and signature-residue packets.

The row O is the additional descent witness. It records that the transported operation respects the mathematical source classifier and lands in the target physical-signature classifier rather than only in a mathematical target classifier.

The witness route W joins those rows before any public physical-operation read. Transport through H, replay through C, and provenance through Q preserve the same finite route, while N names it locally.

No further coordinate remains in the carrier. Thus the displayed rows exhaust the NameCert obligations for physical-signature operation descent. □

Lean4 ✓ `BEDC.Derived.RealityConstrainedSignatureDescentUp.RealityConstrainedSignatureDescentCarrier namecert obligations`

Theorem 1303.3 (Reality-constrained signature-descent no-free operation)**.** *Let $D = (S, A, M, G, P, O, W, H, C, Q$ be an accepted* $\mathsf{RealityConstrainedSignatureDescent}^{\uparrow}$ *packet. A transported mathematical operation is accepted as a physical-fit operation only when the same route displays the stream schedule S, GAP row G, representation certificate P, physical-signature descent witness O, witness route W, transport row H, replay row C, provenance row Q, and local naming row N. If any of S, G, P, O, W is absent, the route remains mathematical operation transport and is rejected as a physical-fit operation.*

Proof. Use the NameCert obligations of Definition 1303.2. A public physical-operation read must expose the schedule, GAP row, representation certificate, and descent witness as displayed coordinates of one packet.

The mathematical rows A and M can still support operation transport inside the mathematical classifier. They do not by themselves expose a physical-signature route, because the descent witness O and witness route W are the rows that join classifier transport to the physical signature.

The structural rows cannot add missing physical evidence. Componentwise hsame transport preserves the displayed components, Cont replay follows the same route, and Pkg provenance records that route rather than introducing a hidden host witness.

Therefore a route lacking S, G, P, O, or W has no accepted physical-fit operation coordinate. It is rejected as a physical-fit operation even if the mathematical operation is transported. □

Falsifiable prediction: A transported mathematical operation that lacks the displayed stream schedule, GAP row, representation certificate, and physical-signature descent witness is rejected as a physical-fit operation.

Independence witness: reality-constrained-signature-fit, reality-constrained-signature-residue, streamname, real, metric: none carries the operation-descent witness row together with the physical-signature route. Signature-fit binds compatibility rows without the descent witness, signature-residue records remaining separation without licensing operation transport, StreamName supplies schedules without Real or Metric operation data, Real supplies endpoints without physical signatures, and Metric supplies comparison without GAP or representation certificates.

The Lean follow-up site `lean4/BEDC/Derived/RealityConstrainedSignatureDescentUp/TasteGate.lean` must provide Nontrivial RealityConstrainedSignatureDescent$^{\uparrow}$ and FieldFaithful RealityConstrainedSignatureDescent$^{\uparrow}$ instances for the same finite descent packet.

Closure status for NameCert$_{\text{RealityConstrainedSignatureDescent}^{\uparrow}}$ **.**

Constructive story: A RealityConstrainedSignatureDescent$^{\uparrow}$ packet is generated by joining a reality-constrained signature fit, GAP boundary, physical-representation certificate, operation-signature row, descent certificate, componentwise hsame transport, displayed Cont replay, package provenance, and local naming into a finite BHist carrier. The classifier is not bare operation transport: it requires the signature row, the GAP socket, the representation certificate, and the physical-operation surface to be read together. The ledger story is descent under constraint, where an operation may move down the signature route only when every physical and gap-facing coordinate remains visible through the same continuation and provenance rows.

Theory closure: TheoryGate(NameCert$_{\text{RealityConstrainedSignatureDescent}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition, NameCert-obligation theorem, no-free-operation theorem, falsifiable prediction, independence witness, and Lean-follow-up boundary for RealityConstrainedSignatureDescent$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RealityConstrainedSignatureDescent}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim bare mathematical operation transport, physical adequacy without the displayed signature rows, quotient-stream equality, host physical identity, or unledgered representation shortcuts.

Upgrade path: To reach obligationClosure, the signature-fit carrier and no-free-operation descent boundary must receive checked finite-history proofs.

Chapter 1304

A Concrete Naming Certificate for $\mathsf{PhenomenologicalFrame}^{\uparrow}$

The $\mathsf{PhenomenologicalFrame}^{\uparrow}$ packet realizes the phenomenological framework of section 9462.27 as a finite NameCert carrier. It sits beside the focused ledger of chapter 1247, the focused multi-Hist ledger of chapter 1109, the observer-subject boundary of chapter 615, the gap-closure boundary of chapter 1006, and the causal-commitment packet of chapter 505. Those siblings expose focused records, multi-history support, no-subject observation, shared gap refusal, and forward causal sockets; this chapter binds the complete phenomenological frame before any consumer reads consciousness, universe, action, or time as a hidden object.

Definition 1304.1 (Phenomenological frame carrier)**.** *A* $\mathsf{PhenomenologicalFrame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (L, M, U, O, G, A, T, H, C, P, R, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{FocusedLedger}^{\uparrow}$ *row exposing selected histories, focus, selector, digest, gap, relation, invariant, dependence, audit, replay, provenance, and local naming data;*
2. *M is the* $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ *row exposing the finite multi-history support used by the frame;*
3. *U is the consciousness/universe row, read through the finite universe-slice discipline of* $\mathsf{ConsciousUniverseSpace}^{\uparrow}$ *rather than through a global universe object;*
4. *O is the* $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ *row that keeps observation on finite* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows without adding observer substance;*
5. *G is the* $\mathsf{GapClosureBoundary}^{\uparrow}$ *row recording gap requests, source rows, refusal ledgers, and non-escape data;*
6. *A is the hard-problem and action row, consuming the gap boundary and the* $\mathsf{CausalCommitment}^{\uparrow}$ *socket for action, causality, freedom, other minds, mind–matter relation, and self-reference readings;*
7. *T is the local time and continuation ledger, exposing only displayed continuation steps and no object-level time carrier;*
8. *H records componentwise* hsame *transport for $L, M, U, O, G, A, T, C, P, R, N$;*

9. *C records displayed* Cont *replay for focused ledger reads, multi-history restriction, universe-slice access, observer-boundary inspection, gap refusal, causal commitment, and local time steps;*
10. *P records* Pkg *provenance over* $\mathsf{PhenomenologicalFrame}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{ConsciousUniverseSpace}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *R is the audit-refusal row excluding subject substance, global universe object, hidden chooser, object-level T, host equality, quotient history equality, and off-packet continuation;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PhenomenologicalFrame}^{\uparrow}}$ *row.*

The carrier has no coordinate for private consciousness substance, ambient world container, global time object, unledgered agency selector, hidden solution of the hard problem, or causal necessity outside the displayed rows.

Lean4 ✓ `BEDC.Derived.PhenomenologicalFrameUp.PhenomenologicalFrameTasteGate single carrier alignment round trip`

Theorem 1304.2 (Phenomenological frame NameCert obligations)**.** *For the carrier of Definition 1304.1, the local* $\mathsf{NameCert}_{\mathsf{PhenomenologicalFrame}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (L, M, U, O, G, A, T, H, C, P, R, N)$, focused-ledger admission through L, multi-history support through M, consciousness/universe slicing through U, no-subject observation through O, gap and refusal routing through G, hard-problem and action commitment through A, local time and continuation exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, audit refusal through R, and local naming through N.*

Proof. Project the carrier into its twelve rows. The rows L, M, U, O expose focused records, multi-history support, finite universe slicing, and no-subject observation before the frame can be consumed as a phenomenological packet.

Read the gap and action rows next. The gap boundary G supplies source/refusal routing, while A records hard-problem and causal-commitment readings only as displayed socketed commitments.

Finally use the structural rows T, H, C, P, R, N. Local time remains a continuation ledger, transport and replay preserve only displayed rows, provenance names the sibling sources, audit refusal excludes hidden objects, and local naming registers the packet. Since no further coordinate exists, these obligations exhaust the local NameCert surface. □

Theorem 1304.3 (Phenomenological frame audit non-escape)**.** *Every accepted* $\mathsf{PhenomenologicalFrame}^{\uparrow}$ *consumer reads consciousness, universe, time, identity, causality, freedom, other minds, mind–matter relation, and self-reference only through the finite rows*

$$L, M, U, O, G, A, T, H, C, P, R, N.$$

It receives no subject substance, global universe object, hidden chooser, object-level T, host equality, quotient history equality, hidden hard-problem solution, causal necessity row, or off-packet continuation.

Proof. Apply the NameCert obligation surface. It identifies $L, M, U, O, G, A, T, H, C, P, R, N$ as the complete accepted packet and supplies no additional public coordinate.

Inspect the sibling rows. $\mathsf{FocusedLedger}^{\uparrow}$ and $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ supply finite focused histories; $\mathsf{ObserverSubjectBoundary}^{\uparrow}$ refuses a separate subject carrier; $\mathsf{GapClosureBoundary}^{\uparrow}$ routes unsupported closure requests to visible refusal; and $\mathsf{CausalCommitment}^{\uparrow}$ exposes causal reading as a forward socket rather than as necessity.

The excluded objects would require coordinates outside the displayed carrier. The audit row R records their refusal, while H, C, P, N only transport, replay, source, and name the same packet. Therefore the accepted frame cannot export any of the excluded data. □

Theorem 1304.4 (Phenomenological frame dependency package)**.** *The dependency surface of an accepted* $\mathsf{PhenomenologicalFrame}^{\uparrow}$ *carrier is the ordered package consisting of focused-ledger rows, focused multi-Hist rows, consciousness/universe-slice rows, observer-subject-boundary rows, gap-closure rows, causal-commitment action rows, local-time continuation rows, transport, replay, provenance, audit refusal, and local naming. Dropping any one of the sibling rows* L, M, O, G, A *changes the accepted dependency package.*

Proof. The carrier definition assigns one displayed coordinate to each sibling role. The package therefore has distinct rows for focus, multi-history support, observer boundary, shared gap closure, and causal commitment.

If L is absent, the frame has no focused record route. If M is absent, it has no finite multi-history support. If O is absent, observation can no longer be certified without a subject-substance gap. If G is absent, unsupported closure requests have no refusal ledger. If A is absent, action and causal commitment are not socketed into the frame.

The remaining structural rows cannot reconstruct a missing sibling row. They preserve, replay, source, refuse, and name only what is already displayed, so the dependency package changes whenever any listed sibling coordinate is removed. □

Theorem 1304.5 (Phenomenological frame independence witness)**.** *The* $\mathsf{PhenomenologicalFrame}^{\uparrow}$ *carrier is not row-bijective to any of* $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, $\mathsf{ObserverSubjectBoundary}^{\uparrow}$, $\mathsf{GapClosureBoundary}^{\uparrow}$, *or* $\mathsf{CausalCommitment}^{\uparrow}$. *A row-bijection preserving local NameCert obligations would have to preserve focused ledger data, multi-history support, observer no-subject data, source/refusal boundary data, causal commitment data, local time, transport, replay, provenance, audit refusal, and local naming in one packet, while each listed sibling omits at least one of these rows.*

Proof. Compare the carrier inventories. $\mathsf{FocusedLedger}^{\uparrow}$ lacks the separate multi-history support, observer-boundary, gap-boundary, and causal-commitment rows of the full frame. $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ lacks the full consciousness/universe, gap/action, and audit-refusal package.

$\mathsf{ObserverSubjectBoundary}^{\uparrow}$ exposes no focused-ledger or causal-commitment packet. $\mathsf{GapClosureBoundary}^{\uparrow}$ exposes source/refusal routing but not the focused ledger, universe slice, observer-boundary packet, or local-time frame. $\mathsf{CausalCommitment}^{\uparrow}$ exposes a forward causal socket but not the full phenomenological carrier.

Thus no listed sibling is bijective row for row to $F = (L, M, U, O, G, A, T, H, C, P, R, N)$ while preserving local NameCert obligations. □

Falsifiable prediction: Any accepted PhenomenologicalFrameUp packet that exports subject substance, global universe object, hidden chooser, or object-level T without the displayed ledger and audit rows refutes the carrier.

Independence witness: focusedledger, focusedmultihistledger, observersubjectboundary, gapclosureboundary, and causalcommitment: none is bijective to the phenomenological frame because the frame binds focused records, multi-history support, universe slicing, no-subject observation, gap refusal, causal action, local time, transport, replay, provenance, audit refusal, and local naming in one row-distinct packet.

The Lean follow-up for this chapter must provide explicit Nontrivial PhenomenologicalFrame$^{\uparrow}$ and FieldFaithful PhenomenologicalFrame$^{\uparrow}$ instances in `lean4/BEDC/Derived/PhenomenologicalFrameUp/TasteGa` The field-faithfulness witness must preserve $L, M, U, O, G, A, T, H, C, P, R, N$ as distinct rows before `BEDC.Derived.PhenomenologicalFrameUp.taste_gate` can be cited by a checked paper marker.

Closure status for NameCert$_{\text{PhenomenologicalFrame}^{\uparrow}}$.

Constructive story: A PhenomenologicalFrame$^{\uparrow}$ packet is generated from focused-ledger, focused multi-Hist, consciousness/universe, observer-boundary, gap-closure, causal-action, local-time, transport, replay, provenance, audit-refusal, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{PhenomenologicalFrame}^{\uparrow}}$, seed)

Scope closed: Definition 1304.1, Definition 1304.2, Definition 1304.3, Definition 1304.4, and Definition 1304.5 specify the finite phenomenological-frame carrier, its certificate obligations, non-escape boundary, dependency package, and row-level independence.

Formal verification: FormalStatus(NameCert$_{\text{PhenomenologicalFrame}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, private consciousness substance, global universe construction, global time, hidden chooser selection, object-level T, or causal necessity.

Upgrade path: Obligation closure requires checked carrier admission, displayed classifier stability, dependency exactness, audit refusal, and TasteGate Nontrivial and FieldFaithful obligations for the same finite packet.

Chapter 1305

A Concrete Naming Certificate for $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$

The $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ packet realizes the strength-gradient discipline of subsection 9462.32.2 inside the auditable-generativity programme of chapter 9462. It sits beside the philosophy-package carrier of chapter 1292, the upgrade registry of chapter 1274, the verification-failure roadmap of chapter 1289, the gap-and-bridge audit surface of chapter 1229, and the formal-constant empirical-value boundary of chapter 1297. Those siblings expose package rows, promotion rows, failure-roadmap rows, bridge-audit rows, and empirical-boundary rows separately; this packet binds claim strength, dependency status, verification status, bridge status, refusal, provenance, transport, replay, and local naming before a claim may export a reported BEDC strength.

Definition 1305.1 (Programme strength ledger carrier)**.** *A* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, S, D, V, B, R, H, C, P, N)$$

with the following displayed coordinates:

1. *Q is the claim-site row, recording the theorem, definition, chapter, bridge target, or package row whose reported strength is being inspected.*
2. *S is the reported-strength row, recording whether the site is a naming seed, paper certificate, machine-checked certificate, or conservative bridge.*
3. *D is the dependency row, recording lower-layer blockers, sibling inputs, open gaps, and required upstream rows.*
4. *V is the theory/formal status pair, recording the independent theory-closure and verification-status axes without collapsing either axis into the other.*
5. *B is the bridge-status row, recording whether the site has no bridge, a paper bridge, or a checked bridge.*
6. *R is the refusal and not-claimed row, recording the stronger readings that the site may not export.*
7. *H records componentwise* hsame *transport for $Q, S, D, V, B, R, C, P, N$.*
8. *C records displayed* Cont *replay through the claim site, reported strength, dependency, status, bridge, and refusal rows.*

9. *P is the* Pkg *provenance over* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$, $\mathsf{BedcPhilosophyPackage}^{\uparrow}$, $\mathsf{UpgradeRegistry}^{\uparrow}$, $\mathsf{VerificationFailureRoadmap}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for promoting a seed to a certificate, treating a paper certificate as machine checked, reading an unchecked bridge as conservative, erasing a blocker row, or exporting a stronger claim after the refusal row has named it.

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerCarrier displayed route`

Theorem 1305.2 (ProgrammeStrengthLedger NameCert obligations)**.** *For the carrier of Definition 1305.1, the local* $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $L = (Q, S, D, V, B, R, H, C, P, N)$, claim-site exposure through Q, reported-strength exposure through S, dependency exposure through D, theory/formal status exposure through V, bridge-status exposure through B, refusal and not-claimed exposure through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerNameCert obligations`

Proof. Project the displayed BHist packet. The claim site Q is not yet a strength report; it becomes reportable only after the strength row S, dependency row D, status pair V, bridge row B, and refusal row R have all been read.

The structural rows preserve the same display. Componentwise transport through H keeps each row separate, the continuation row C replays the inspection order, and the provenance row P records the sibling package, registry, roadmap, audit, and empirical-boundary sources.

The local naming row N therefore names the whole packet rather than a bare claim site. Since the carrier has no additional promotion coordinate, these rows exhaust the NameCert obligations. □

Theorem 1305.3 (ProgrammeStrengthLedger no strength inflation)**.** *An accepted* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *packet cannot export a reported BEDC strength above the row displayed by S. In particular, an open dependency in D, a weaker theory/formal status in V, an absent or unchecked bridge row in B, or a refusal in R blocks any consumer from reading Q as a stronger seed, certificate, machine-check, or bridge claim.*

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerNoStrengthInflation`

Proof. Read the carrier rows in order. The reported strength S is paired with dependency status D, theory/formal status V, bridge status B, and refusal row R before the claim site can be exported.

Apply the strength-gradient rule of Definition 9462.2114. A stronger report would require one of the missing obligations to be treated as discharged.

The packet contains no such discharge row. The upgrade registry can record a next obligation, the verification-failure roadmap can record a blocker, and the gap-and-bridge audit can record an open bridge, but none of those sibling rows changes S inside this carrier. Hence every exported strength is bounded by the displayed row. □

Theorem 1305.4 (ProgrammeStrengthLedger axis separation)**.** *For an accepted* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *carrier, the claim site Q, reported-strength row S, dependency row D, theory/formal status pair V, bridge-status row B, refusal row R, and local* NameCert *row N remain separate audit coordinates.*

A consumer may combine them into one public strength report, but it may not derive the theory-closure axis from the verification axis, derive bridge status from formal status, erase dependency blockers by provenance, or replace the refusal row by a sibling package reading.

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerAxisSeparation`

Proof. Inspect the displayed packet. The status pair V records two independent axes, while B and R record bridge and refusal information that are not components of V.

The support rows H, C, P, N preserve and name the separation. Transport and replay can move the packet through equivalent presentations, and provenance can cite the package, registry, roadmap, audit, and boundary siblings, but none of those operations merges rows.

Thus every public report must keep the rows jointly visible. Any route that infers bridge status from formal status, erases a blocker from D, or replaces R by prose from a sibling chapter leaves the accepted $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ carrier. □

Theorem 1305.5 (ProgrammeStrengthLedger public report interface)**.** *A public strength report exported from an accepted* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *carrier may expose the claim-site row Q only together with the reported strength row S, dependency row D, theory/formal status pair V, bridge-status row B, refusal row R, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Such a report may not promote theory closure from formal status, bridge status from verification status, or the reported strength beyond the row displayed by S.*

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerPublicReportInterface`

Proof. Start with the no-strength-inflation theorem Definition 1305.3. It blocks any route that reads Q as a stronger seed, certificate, machine-check, or bridge claim than the displayed row S permits.

Apply the axis-separation theorem Definition 1305.4. It keeps Q, S, D, V, B, R as distinct audit coordinates, so a public report cannot derive theory closure from verification status or bridge status from the formal axis.

The structural rows H, C, P, N preserve, replay, source, and name exactly that joint display. Therefore the public consumer surface is the full row family $Q, S, D, V, B, R, H, C, P, N$, with no promotion coordinate outside the accepted carrier. □

Theorem 1305.6 (ProgrammeStrengthLedger bridge-blocker readback)**.** *Let a public strength report be exported from an accepted* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *carrier $L = (Q, S, D, V, B, R, H, C, P, N)$. If the bridge row B records no bridge or a paper bridge, or if the dependency row D names an open blocker, then the report must export that blocker together with Q, S, V, R, H, C, P, N. It may not read the claim site Q as bridge-checked, machine-verified, or stronger than the displayed rows permit.*

Proof. Start from no strength inflation, Definition 1305.3. An absent bridge, a paper bridge, or an open dependency blocker is one of the displayed reasons that prevents Q from being read above the strength row S.

Apply axis separation, Definition 1305.4. The bridge row B and dependency row D remain separate from the theory/formal status pair V, so neither row can be erased by reading the other as a stronger verification coordinate.

Finally use the public report interface, Definition 1305.5. It forces every public report to expose $Q, S, D, V, B, R, H, C, P, N$ together. Hence a bridge blocker or dependency blocker is exported as part of the report surface, and no consumer can read the packet as bridgeChecked or machine-verified unless the corresponding displayed rows already carry that status. □

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerBridgeBlockerReadback`

Theorem 1305.7 (Finite blocker readback factors through programme strength ledger)**.** *Let $L = (Q, S, D, V, B, R, H, C, P, N)$ be an accepted* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *carrier whose source row cites* $\mathsf{UpgradeRegistry}^{\uparrow}$, $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, *and* $\mathsf{VerificationFailureRoadmap}^{\uparrow}$. *Any blocker readback exposed by D or B factors through a single* BHist *ledger row K carried by L such that K preserves the displayed* hsame *transport and* Cont *provenance route from the source certificates to the public strength report. The factorization is a finite* Pkg*/GAP readback inside the ledger surface, not an external judgement about the claim site Q.*

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerBlockerReadbackFactorization`

Proof. Unpack the accepted carrier. The dependency row D, bridge row B, transport row H, replay row C, provenance row P, and local naming row N are all displayed before the public report can be read, so a blocker cannot be exported as a free-standing judgement.

Compose the three cited source certificates. The $\mathsf{UpgradeRegistry}^{\uparrow}$ row names the current obligation and next required row, the $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ row keeps bridge blockers visible, and the $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ row records the verification blocker. Their common readback is therefore a single finite BHist row K inside the programme-strength ledger.

Finally apply the bridge-blocker readback theorem Definition 1305.6. It forces the blocker to travel with $Q, S, D, V, B, R, H, C, P, N$, while H preserves hsame and C, P preserve the Cont and Pkg provenance route. Hence the blocker readback factors through K as a finite ledger readback and not through any external status judgement. □

Theorem 1305.8 (ProgrammeStrengthLedger formal target request)**.** *Any formalization target for the public* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *strength report must preserve the displayed row family $Q, S, D, V, B, R, H, C, P, N$. Its proof obligations are exactly no-strength-inflation for S, axis separation for Q, S, D, V, B, R, N, the public report interface for $Q, S, D, V, B, R, H, C, P, N$, and bridge-blocker readback for D and B. In particular, the target may not derive theory closure from verification status, derive bridge status from formal status, erase a dependency blocker by provenance, or promote the report beyond the displayed strength row.*

Proof. Begin with the NameCert obligations of Definition 1305.2. They expose the finite carrier $L = (Q, S, D, V, B, R, H, C, P, N)$, so a formal target that drops one coordinate no longer names the public ledger surface.

Apply no strength inflation, Definition 1305.3, and axis separation, Definition 1305.4. These two rows keep the reported strength bounded by S and prevent the theory, formal, bridge, dependency, refusal, and naming coordinates from being inferred from one another.

Finally use the public report interface Definition 1305.5 together with bridge-blocker readback Definition 1305.6. The first theorem exports the full row family, while the second keeps every displayed bridge or dependency blocker visible. These four obligations therefore give the required request surface without turning closure status, verification status, or bridge status into one another. □

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthLedgerUp.ProgrammeStrengthLedgerFormalTargetRequest`

Falsifiable prediction: A packet that omits S or V can still name a claim site, but it cannot export a reported BEDC strength.

Independence witness: bedcphilosophypackage, upgraderegistry, verificationfailureroadmap, gapfailurebridgeaudit, formalconstantempiricalvalueboundary: no listed sibling is bijective with

$\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ because none preserves claim site, strength row, dependency row, theory/formal status pair, bridge row, refusal row, transport, replay, provenance, and local NameCert together. The package carrier stores programme rows without owning the strength audit packet; the upgrade registry records current and next obligations without binding refusal and bridge rows; the verification-failure roadmap records blockers without exporting a strength ledger; the gap-failure bridge audit records bridge blockers without the full status pair; and the empirical-value boundary separates formal and empirical constants rather than claim-strength reporting.

The Lean-side boundary is expected to provide explicit $\mathsf{Nontrivial}\ \mathsf{ProgrammeStrengthLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ProgrammeStrengthLedger}^{\uparrow}$ instances in `lean4/BEDC/Derived/ProgrammeStrengthLedgerUp/Tas` The field-faithfulness witness must keep $Q, S, D, V, B, R, H, C, P, N$ as distinct rows before the target `BEDC.Derived.ProgrammeStrengthLedgerUp.taste_gate` can classify strength-ledger packets.

Closure status for $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ packet is generated from a claim site, reported strength, dependency row, theory/formal status pair, bridge-status row, refusal row, componentwise transport, displayed replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProgrammeStrengthLedger}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1305.1, Definition 1305.2, Definition 1305.3, Definition 1305.4, Definition 1305.5, and Definition 1305.6 scope the displayed carrier, NameCert obligations, no-strength-inflation row, axis-separation row, exported public strength report surface, and bridge-blocker readback through finite audit rows, hsame, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProgrammeStrengthLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, correctness of every package row, discharge of every dependency blocker, or promotion of any named claim beyond its displayed strength row.

Upgrade path: Public closure requires a public strength-report consumer surface without claiming Lean verification, bridge checking, or promotion beyond the displayed strength row.

Chapter 1306

A Concrete Naming Certificate for $\mathsf{EpistemicWitness}^{\uparrow}$

The $\mathsf{EpistemicWitness}^{\uparrow}$ packet realizes the BEDC account of knowledge in subsection 9462.31.3 as a finite auditable witness rather than as a bare truth predicate. It sits beside the failure-certificate packet of chapter 1233, the registry export surface of chapter 1216, and the machine-facing socket of chapter 1059. Those siblings respectively expose blocked gates, public export, and formalization routing; this packet binds a claimed proposition to its visible witness route, strength row, gap or refusal boundary, bridge row, and local naming certificate before it can be read as BEDC knowledge.

Definition 1306.1 (Epistemic witness carrier)**.** *An* $\mathsf{EpistemicWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (P, W, A, S, G, B, H, C, K, N)$$

with the following displayed rows:

1. *P is the proposition or claim row whose BEDC knowledge status is being named;*
2. *W is the witness row, storing the sameness, continuation, classifier, package, gap, naming, semantic-naming, derivation, bridge, gap-report, or failure witness used by the claim;*
3. *A is the audit-route row, recording the finite path by which the witness may be inspected;*
4. *S is the strength row, recording seed, obligation, paper-level closure, checked theorem, checked bridge, gap, or cannot-claim status without collapsing the theory and verification axes;*
5. *G is the gap-or-refusal row, recording any missing route, failed gate, blocked export, or explicit cannot-claim boundary;*
6. *B is the bridge row, recording whether the witness is internal, paper-bridged, checked-bridged, or explicitly unbridged;*
7. *H records componentwise* hsame *transport for the claim, witness, audit route, strength, gap-or-refusal, bridge, provenance, and local name;*
8. *C records displayed* Cont *replay for the witness route, audit route, strength row, gap-or-refusal row, and bridge row;*
9. *K is the* Pkg *provenance over* $\mathsf{FailureCertificate}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceSocket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EpistemicWitness}^{\uparrow}}$ *row.*

The carrier contains no row for bare truth, hidden conviction, external oracle access, host-level semantic satisfaction, or a bridge that bypasses the displayed audit route.

Lean4 (definition) `BEDC.Derived.EpistemicWitnessUp.EpistemicWitnessCarrier`

Theorem 1306.2 (Epistemic witness names knowledge audit)**.** *For the carrier of Definition 1306.1, the local* $\mathsf{NameCert}_{\mathsf{EpistemicWitness}^{\uparrow}}$ *surface consists of five obligation rows:*

1. *carrier admission for $E = (P, W, A, S, G, B, H, C, K, N)$;*
2. *witness-route audit through W, A, H, C;*
3. *strength-axis readback through S without identifying theory closure with formal verification;*
4. *gap, refusal, and bridge boundary readback through G, B;*
5. *provenance and local naming through K, N.*

Every BEDC knowledge read exported by an accepted $\mathsf{EpistemicWitness}^{\uparrow}$ *packet factors through these five rows. In particular, the read cannot be replaced by bare truth, prose assent, hidden registry state, an uninspected derivation certificate, or a bridge witness that omits the audit route.*

Proof. Project the displayed packet $E = (P, W, A, S, G, B, H, C, K, N)$. The claim row P and witness row W give the object being known and the finite witness by which it is known, while A, H, C make the route inspectable and stable under transport and replay.

Read the status rows next. The strength row S records the grade of the knowledge claim, and the gap-or-refusal row G records the boundary when the route is incomplete or blocked. These rows are separate, so a gap, cannot-claim status, checked theorem, and checked bridge cannot be interchanged by a consumer.

Finally read the bridge and naming rows. The bridge row B records how far the witness is exported beyond its local audit route, while K, N source and name the packet. Since no other coordinate exists, an accepted knowledge read must consume the five displayed obligation rows and cannot recover a bare truth predicate or hidden witness from outside the carrier. □

Lean4 ✓ `BEDC.Derived.EpistemicWitnessUp.EpistemicWitnessCarrier namecert obligations`

Falsifiable prediction: Deleting the audit-route row leaves a claim-like packet but not a BEDC knowledge witness.

Independence witness: failurecertificate, registryexportsurface, bridgewitness, derivcert, machineinterfacesocket: none is bijective to $\mathsf{EpistemicWitness}^{\uparrow}$ under NameCert-row preservation. A failure certificate records blocked gates without the full knowledge-strength and bridge surface; a registry export surface controls public release without owning the witness route; a bridge witness may expose cross-theory transport without the local gap-or-refusal row; a derivation certificate records proof ancestry without the public audit-strength packet; and a machine-interface socket routes formal targets without naming the epistemic witness as knowledge.

The later Lean `TasteGate.lean` boundary for this carrier must include `Nontrivial EpistemicWitnessUp` and `FieldFaithful EpistemicWitnessUp` instances, so the knowledge packet is inhabited by an actual audit witness and preserves $P, W, A, S, G, B, H, C, K, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{EpistemicWitness}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EpistemicWitness}^{\uparrow}$ packet is generated by binding a claimed proposition to its visible witness row, audit route, strength row, gap-or-refusal row, bridge row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming in one finite BHist carrier. The classifier treats knowledge as an auditable route

rather than a bare truth predicate: a consumer must read the witness, audit, strength, and boundary rows before it can export an epistemic claim. The ledger story keeps each witness attached to its gap, bridge, and provenance coordinates, so hidden conviction, uninspected derivation, or oracle access never become classifier data.

Theory closure: TheoryGate(NameCert$_{\mathsf{EpistemicWitness}\uparrow}$, seed)

Scope closed: This chapter binds the carrier definition, knowledge-audit theorem, independence witness, Lean-follow-up boundary, and existing closure annotations for EpistemicWitness$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{EpistemicWitness}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim bare truth, hidden conviction, external oracle access, host-level semantic satisfaction, or bridge access bypassing the audit route.

Upgrade path: To reach obligationClosure, the witness-route carrier and knowledge-audit obligations must be encoded as checked finite packet projections.

Chapter 1307

A Concrete Naming Certificate for PackageMapBoundary$^{\uparrow}$

The PackageMapBoundary$^{\uparrow}$ packet realizes the map and boundary discipline of subsection 9462.33.4 inside the auditable-generativity programme of chapter 9462. It sits beside the package carrier of chapter 1292, the claim registry layer of chapter 1209, the socket report of chapter 1279, the standard bridge audit packet of chapter 1217, and the host primitive leakage boundary of chapter 1157. Those siblings expose package rows, claim admission, socket diagnostics, bridge-audit rows, and host-leak refusal; this packet binds the four map families and their axis boundary before a philosophy-package audit reading can be exported.

Definition 1307.1 (Package map-boundary carrier)**.** *A* PackageMapBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, G, R, S, X, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the theorem-map row, recording theorem sites, definition sites, proof mode, claim strength, dependency surface, and not-claimed readings;*
2. *G is the gap-map row, recording typed gaps, requested supply, socket type, gate, status, route, and forbidden internalization;*
3. *R is the tradition-map row, recording inherited comparison, preserved structure, rejected surplus, audit status, and package route;*
4. *S is the scientific-map row, recording records carrier, invariant carrier, empirical ledger, prediction gap, bridge status, claim status, and failure surface;*
5. *X is the axis-boundary row, recording that theorem strength, gap status, tradition comparison, scientific bridge status, theory closure, formal verification, and bridge status are distinct audit coordinates;*
6. *A is the audit-route row, recording the philosophy-package carrier, claim registry layer, socket report, standard bridge audit packet, and host primitive leakage route consulted by the export;*
7. *H records componentwise* hsame *transport for theorem maps, gap maps, tradition maps, scientific maps, axis boundaries, audit routes, provenance, and local naming;*
8. *C records displayed* Cont *replay for the four map rows, the axis-boundary row, and the audit-route row;*

9. *P is the* Pkg *provenance over* BedcPhilosophyPackage$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, SocketReport$^{\uparrow}$, StandardBridgeAuditPacket$^{\uparrow}$, HostPrimitiveLeakage$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PackageMapBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a prose-only map, a theorem entry detached from its gap boundary, a tradition comparison detached from rejected surplus, a scientific bridge detached from failure surface, or a collapse of map status into formal verification or bridge status.

Lean4 ✓ `BEDC.Derived.PackageMapBoundaryUp.PackageMapBoundaryTasteGate single carrier alignment`

Theorem 1307.2 (Package map-boundary NameCert obligations)**.** *For the carrier of Definition 1307.1, the local* $\mathsf{NameCert}_{\mathsf{PackageMapBoundary}^{\uparrow}}$ *obligation surface consists exactly of theorem-map admission through T, gap-map admission through G, tradition-map admission through R, scientific-map admission through S, axis-boundary admission through X, audit routing through A, transport and replay through H, C, provenance through P, and local naming through N.*

Proof. Project the finite packet $M = (T, G, R, S, X, A, H, C, P, N)$. The four map rows expose theorem, gap, tradition, and scientific readings before any package export is named.

Use the package and registry siblings next. The package carrier supplies the package-level row, the claim registry layer controls claim admission, and the socket report records the diagnostic socket through which a gap or map entry is inspected.

The bridge and host-leak siblings account for the export boundary. The standard bridge audit packet displays bridge fields, while the host primitive leakage packet refuses host equality, quotient import, choice, and semantic primitives.

Finally read X, A, H, C, P, N. These rows separate the status axes, route the audit, transport and replay only the displayed packet, source it, and name it locally. Since no other coordinate is available, the listed rows exhaust the NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.PackageMapBoundaryUp.PackageMapBoundaryNameCertObligations`

Theorem 1307.3 (Package map-boundary non-escape)**.** *Every accepted philosophy-package audit reading through* PackageMapBoundary$^{\uparrow}$ *factors through*

$$T, G, R, S, X, A, H, C, P, N.$$

If any one of the theorem-map, gap-map, tradition-map, scientific-map, or axis-boundary rows is absent, the packet cannot export a philosophy-package audit reading.

Lean4 ✓ `BEDC.Derived.PackageMapBoundaryUp.PackageMapBoundaryNonEscape`

Proof. Start from Definition 1307.2. The theorem-map, gap-map, tradition-map, scientific-map, and axis-boundary rows are admitted separately before the audit route and local name can be consumed.

The package audit route then reads the same finite carrier through A, H, C, P, N. These rows preserve the displayed rows and cannot manufacture a missing map family or a missing axis boundary.

Thus an accepted reading remains on $T, G, R, S, X, A, H, C, P, N$. A route missing any of T, G, R, S, X may still be prose about the package, but it is not a PackageMapBoundary$^{\uparrow}$ audit export. □

Falsifiable prediction: A map-boundary packet lacking any one of theorem-map, gap-map, tradition-map, scientific-map, or axis-boundary rows cannot export a philosophy-package audit reading.

Independence witness: bedcphilosophypackage, claimregistrylayer, socketreport, standardbridgeauditpacket, hostprimitiveleakage: none is bijective because $\mathsf{PackageMapBoundary}^{\uparrow}$ carries the simultaneous four-map plus axis-boundary audit row.

The Lean site `lean4/BEDC/Derived/PackageMapBoundaryUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ PackageMapBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ PackageMapBoundary}^{\uparrow}$ instances preserving $T, G, R, S, X, A, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{PackageMapBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{PackageMapBoundary}^{\uparrow}$ packet is generated from a package map row, theorem row, gap-boundary row, tradition-comparison row, scientific-bridge row, status row, refusal row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts a map only when its theorem entry, gap boundary, rejected surplus, bridge surface, and status discipline are all present in the same finite BHist carrier. The ledger story is boundary preservation: package motion may be read by consumers only through the displayed map and refusal rows, so formal verification, bridge status, and prose comparison cannot silently collapse into one export coordinate.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PackageMapBoundary}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: This chapter scopes the carrier and obligation rows over BHist, hsame, Cont, Pkg, and NameCert by Definition 1307.1, Definition 1307.2, and Definition 1307.3.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PackageMapBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim prose-only maps, theorem entries detached from gap boundaries, detached tradition comparisons, detached scientific bridges, or status collapse into verification.

Upgrade path: Formal upgrade requires preserving the scoped package-map boundary carrier and non-escape theorem as checked finite-row projection and transport obligations.

Chapter 1308

A Concrete Naming Certificate for PhilosophyGapMapLedger$^{\uparrow}$

The PhilosophyGapMapLedger$^{\uparrow}$ packet realizes the gap-map table of subsection 9462.33.4 as a finite NameCert ledger for audit-facing boundary questions. It sits beside the philosophy-map target of chapter 1277, the package carrier of chapter 1292, the theorem-gap registry of chapter 1246, the gap-failure bridge audit of chapter 1229, and the claim registry layer of chapter 1209. Those siblings expose map targets, package rows, theorem/gap entries, bridge-audit rows, and claim admission; this packet binds the gap entry, socket kind, gate, status, upgrade route, and forbidden-internalization row before any philosophy export may read a boundary question as closed.

Definition 1308.1 (Philosophy gap-map ledger carrier)**.** *A* PhilosophyGapMapLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, S, K, T, U, F, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the gap-entry row, recording the name, BEDC site, requested supply, and claim surface for each gap entry;*
2. *S is the socket-kind row, recording whether the entry requests theorem supply, bridge supply, empirical supply, reflection supply, or package-audit supply;*
3. *K is the gate row, recording the local gate that must be passed before the entry may move from reported gap to accepted theorem, bridge, or package row;*
4. *T is the status row, recording open, blocked, scaffolded, checked, bridged, or refused status without collapsing theory closure into formal verification;*
5. *U is the upgrade-route row, recording the finite route from the current status to the next admissible BEDC obligation;*
6. *F is the forbidden-internalization row, recording that the gap may not be exported as a closed theorem, primitive truth predicate, host equality, quotient import, choice principle, or hidden subject-substance reading;*
7. *A is the audit-route row, recording the philosophy-map target, package carrier, theorem-gap registry, gap-failure bridge audit, and claim registry layer consulted by the export;*
8. *H records componentwise* hsame *transport for entries, sockets, gates, statuses, upgrade routes, forbidden-internalization rows, audit routes, provenance, and local naming;*

9. *C records displayed* Cont *replay for the same gap-map ledger rows;*
10. *P is the* Pkg *provenance over* PhilosophyMapTarget$^{\uparrow}$, BedcPhilosophyPackage$^{\uparrow}$, *theorem-gap registry rows,* GapFailureBridgeAudit$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhilosophyGapMapLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for prose adjacency, unchecked theorem closure, implicit bridge success, host-level semantic completion, or a status row that erases the forbidden-internalization boundary.

Lean4 ✓ `BEDC.Derived.PhilosophyGapMapLedgerUp.PhilosophyGapMapLedgerTasteGate single carrier alignment`

Theorem 1308.2 (Philosophy gap-map ledger exactness)**.** *For the carrier of Definition 1308.1, every public gap-map export is read from exactly the displayed rows $E, S, K, T, U, F, A, H, C, P, N$. In particular, the entry name, socket kind, gate, status, upgrade route, and forbidden-internalization row are separate finite coordinates and cannot be recovered from prose placement or from a sibling package route alone.*

Proof. Project the finite packet $G = (E, S, K, T, U, F, A, H, C, P, N)$. The gap entry, socket kind, gate, status, upgrade route, and forbidden-internalization rows are displayed before the audit route or local name is consumed.

The audit route A may cite the philosophy-map target, package carrier, theorem-gap registry, gap-failure bridge audit, and claim registry layer. Those citations only explain where the displayed ledger is checked; they do not supply a replacement coordinate for any of E, S, K, T, U, F.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no prose-only coordinate and no hidden closure coordinate, every public export factors through the listed rows. □

Theorem 1308.3 (Philosophy gap-map ledger nonescape)**.** *Every accepted* PhilosophyGapMapLedger$^{\uparrow}$ *export must pass through the forbidden-internalization row F. If F is absent, a gap-map export has no displayed coordinate that refuses the reading of a boundary question as a closed theorem, primitive truth predicate, host equality, quotient import, choice principle, or hidden subject-substance claim.*

Proof. Use the carrier of Definition 1308.1. The rows E, S, K, T, U identify the gap, socket, gate, status, and next route, but none of them states the internalization refusal.

The audit rows A, H, C, P, N preserve and name only what the carrier already displays. They can route the ledger through sibling packets and replay the same packet for a consumer, but they cannot manufacture a missing refusal row.

Therefore an accepted export that blocks closure illusion must read F explicitly. Without F, the packet may still list an open gap, but it does not certify nonescape for the forbidden closed readings. □

Theorem 1308.4 (Philosophy gap-map ledger NameCert obligations)**.** *For the carrier of Definition 1308.1, the local* $\mathsf{NameCert}_{\mathsf{PhilosophyGapMapLedger}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for $G = (E, S, K, T, U, F, A, H, C, P, N)$, gap-entry exposure through E, socket-kind exposure through S, gate exposure through K, status exposure through T, upgrade-route exposure through U, forbidden-internalization exposure through F, audit routing through A, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Read the packet by coordinates. Carrier admission supplies the finite BHist object, and the rows E, S, K, T, U, F display the gap-map data before any export is named.

The audit route A then records the sibling surfaces consulted by the export: philosophy-map target, package carrier, theorem-gap registry, gap-failure bridge audit, and claim registry layer. This route keeps those siblings as dependencies rather than replacing the ledger rows.

Finally H, C, P, N provide transport, replay, provenance, and the local name. These rows preserve the same displayed coordinates, so the listed fields exhaust the local NameCert obligations. □

Theorem 1308.5 (Philosophy gap-map ledger sibling independence)**.** *The sibling packets* $\mathsf{PhilosophyMapTarget}^{\uparrow}$, $\mathsf{BedcPhilosophyPackage}^{\uparrow}$, *theorem-gap registry rows,* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$, *and* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *are dependencies of* $\mathsf{PhilosophyGapMapLedger}^{\uparrow}$ *only through displayed routes in* A *and provenance in* P*. None is bijective with the carrier* $G = (E, S, K, T, U, F, A, H, C, P, N)$*.*

Proof. Inspect the carrier. It stores the gap entry, socket kind, gate, status, upgrade route, forbidden-internalization row, audit route, transport, replay, provenance, and local name together.

Each sibling stores a narrower surface. The philosophy-map target names map obligations, the package carrier stores package rows, the theorem-gap registry stores theorem/gap entries, the gap-failure bridge audit stores typed gap and bridge audit rows, and the claim registry layer stores claim admission and refusal rows.

The ledger may cite those siblings through A and P. But no one sibling reconstructs all eleven displayed coordinates, and the gap-map ledger cannot be projected back to any one sibling without losing the forbidden-internalization row or local ledger name. □

Falsifiable prediction: Removing the forbidden-internalization row allows a gap-map export to present a boundary question as a closed theorem.

Independence witness: philosophy-map-target, bedc-philosophy-package, theorem-gap-registry, gap-failure-bridge-audit, claimregistrylayer: none is bijective because the gap-map ledger alone isolates gap entry, socket kind, gate, status, upgrade route, and forbidden-internalization rows.

The Lean follow-up must provide Nontrivial $\mathsf{PhilosophyGapMapLedger}^{\uparrow}$ and FieldFaithful PhilosophyGapMapLedge in `lean4/BEDC/Derived/PhilosophyGapMapLedgerUp/TasteGate.lean`. Those instances must preserve $E, S, K, T, U, F, A, H, C, P, N$ as separate displayed fields before the target `BEDC.Derived.PhilosophyGapl` can be cited by a checked marker.

Closure status for $\mathsf{NameCert}_{\mathsf{PhilosophyGapMapLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PhilosophyGapMapLedger}^{\uparrow}$ packet is generated from a philosophical node row, gap row, package-map row, proof-witness row, tradition-comparison row, signature-descent row, status row, forbidden-internalization boundary, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts the ledger only when the philosophical adjacency is backed by these finite rows rather than by prose proximity. The constructive story is a map of gaps: each public export is generated from the carried gap, proof, comparison, descent, and status coordinates, while the forbidden-boundary row records where closure or semantic completion is not available inside the packet.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhilosophyGapMapLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, exactness, nonescape, NameCert-obligation, sibling-independence, Lean-follow-up boundary, and existing closure annotations for $\mathsf{PhilosophyGapMapLedger}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhilosophyGapMapLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim prose adjacency, unchecked theorem closure, implicit bridge success, host-level semantic completion, or status rows erasing forbidden internalization.

Upgrade path: To reach obligationClosure, the gap-map ledger and its nonescape exactness must be encoded as finite-row carrier proofs.

Chapter 1309

A Concrete Naming Certificate for $\mathsf{NameEligibilityProof}^{\uparrow}$

The $\mathsf{NameEligibilityProof}^{\uparrow}$ packet realizes the inward eligibility proof demanded by Definition 9462.2110 as one finite NameCert carrier. It sits beside the machine-facing route of chapter 1221, the registry audit of chapter 1223, the kernel audit witness of chapter 546, and the apophatic naming boundary of chapter 543. Those siblings expose machine routing, registry consistency, accepted-kernel readback, and refusal-bearing named boundaries; this chapter binds source eligibility, classifier eligibility, strength, audit, refusal, transport, replay, provenance, and local naming before a proposed name can be reported as licensed.

Definition 1309.1 (Name eligibility proof carrier)**.** *A* $\mathsf{NameEligibilityProof}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, S, C, Q, A, R, H, K, P, N)$$

with the following displayed rows:

1. *T is the target-name row, recording the candidate BEDC name and the public surface on which the name is requested.*
2. *S is the source-route row, recording the finite source specification through which the candidate name is generated or inherited.*
3. *C is the classifier-route row, recording the classifier evidence through which accepted packets for the name are compared.*
4. *Q is the strength row, recording seed, paper-certificate, machine-checked, or conservative-bridge strength without allowing a lower row to report a higher strength.*
5. *A is the audit row, recording the registry, machine-interface, and kernel-readback surfaces consulted before the name is admitted.*
6. *R is the refusal row, recording apophatic boundary data for missing source, classifier, strength, audit, or bridge obligations.*
7. *H records componentwise* hsame *transport for target names, source routes, classifier routes, strength rows, audit rows, refusal rows, provenance, and local naming.*
8. *K records displayed* Cont *replay for source eligibility, classifier eligibility, audit readback, refusal propagation, and strength reporting.*

9. *P is the* Pkg *provenance over* MachineInterfaceBoundary$^{\uparrow}$, RegistryConsistencyAudit$^{\uparrow}$, KernelAuditWitness$^{\uparrow}$, ApophaticName$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{NameEligibilityProof}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-selected object, hidden source supply, classifier by prose resemblance, strength promotion without audit, kernel route outside the displayed audit row, or refusal erased after naming.

Lean4 (definition) `BEDC.Derived.NameEligibilityProofUp.NameEligibilityProofUp`

Theorem 1309.2 (Name eligibility proof NameCert obligations)**.** *For the carrier of Definition 1309.1, the local* NameCert$_{\text{NameEligibilityProof}^{\uparrow}}$ *obligation surface consists exactly of target-name admission through T, source-route eligibility through S, classifier-route eligibility through C, strength reporting through Q, audit exposure through A, refusal exposure through R, componentwise transport through H, displayed continuation replay through K, provenance through P, and local naming through N.*

Proof. Project the finite packet $E = (T, S, C, Q, A, R, H, K, P, N)$. The target-name row, source route, and classifier route give the eligibility spine before any local name can be consumed.

Read the machine, registry, kernel, and apophatic siblings next. The machine-interface boundary routes the target to an auditable machine surface, the registry-consistency audit checks export agreement, the kernel audit witness supplies accepted-declaration readback when such a route is present, and the apophatic name row keeps absent supply or refused internalization visible.

The strength, audit, and refusal rows then determine whether the name is only seeded, paper-certified, machine-checked, or bridge-facing. Transport, replay, provenance, and local naming preserve that same packet. Since no further coordinate is present, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1309.3 (Name eligibility proof strength non-inflation)**.** *An accepted* NameEligibilityProof$^{\uparrow}$ *packet cannot raise a target-name row from seed strength to machine-checked strength unless the audit row A and the replay row K display the machine-interface, registry-consistency, and kernel-readback route used for that report.*

Proof. Begin with the strength row Q. It records the reported certificate strength as one row of the same packet, not as an ambient judgement detached from the carrier.

Apply the NameCert obligations above. Machine-checked strength requires source-route and classifier-route eligibility, and it also requires audit exposure through A plus displayed replay through K. The sibling machine-interface, registry-consistency, and kernel-audit rows supply exactly that public route.

If A or K is absent, the packet still may contain a target name, source route, classifier route, or refusal row, but it lacks the displayed evidence required for machine-checked reporting. Thus the strength row cannot promote the name beyond the audit surface carried by the packet. □

Theorem 1309.4 (Name eligibility proof kernel-route non-escape)**.** *Every kernel-facing route exported by an accepted* NameEligibilityProof$^{\uparrow}$ *packet factors through*

$$T, S, C, Q, A, R, H, K, P, N.$$

It cannot bypass the source route, classifier route, audit row, refusal row, componentwise hsame *transport, displayed* Cont *replay, provenance, or local* NameCert *row.*

Proof. Use Definition 1309.2 to enumerate the public surface. A kernel-facing export is admitted only after the target, source, classifier, strength, audit, and refusal rows have been read in the same packet.

The kernel audit witness contributes accepted-declaration readback only through A and K. The registry and machine-interface siblings keep the route public, while the apophatic-name sibling keeps missing supply visible rather than internalized.

Finally read H, P, N. These rows transport, source, and name the displayed carrier; they do not add a hidden kernel route. Therefore every exported kernel-facing route remains on the finite row sequence shown above. □

Theorem 1309.5 (Name eligibility proof obligation package)**.** *For an accepted* $\mathsf{NameEligibilityProof}^{\uparrow}$ *packet*

$$E = (T, S, C, Q, A, R, H, K, P, N),$$

the rows $T, S, C, Q, A, R, H, K, P, N$ *jointly bind the target name, source route, classifier route, strength, audit, refusal, transport, replay, provenance, and local naming as the finite obligation package for eligibility-before-use. A consumer may report the proposed name as licensed only through that displayed package.*

Proof. Begin with Definition 1309.2. It identifies $T, S, C, Q, A, R, H, K, P, N$ as the complete local NameCert surface for target-name admission, source eligibility, classifier eligibility, strength reporting, audit exposure, refusal exposure, transport, replay, provenance, and local naming.

Apply Definition 1309.3. It prevents the strength row Q from promoting a name beyond the audit and replay rows actually displayed in the same packet.

Finally apply Definition 1309.4. It keeps every kernel-facing route on the row sequence $T, S, C, Q, A, R, H, K, P,$ The three results compose into the stated eligibility-before-use package: the target name is not licensed before the source route, classifier route, strength, audit, refusal, transport, replay, provenance, and local naming rows have all been displayed. □

Falsifiable prediction: If a NameEligibilityProofUp packet raises a seed-strength name to machine-checked strength without an audit row, the carrier is refuted.

Independence witness: machineinterfaceboundary, registryconsistencyaudit, kernelauditwitness, apophaticname: none is bijective because NameEligibilityProofUp simultaneously binds source eligibility, classifier eligibility, strength, audit, refusal, transport, replay, provenance, and local NameCert rows.

Lean4 ✓ `BEDC.Derived.NameEligibilityProofUp.taste gate`

Closure status for $\mathsf{NameCert}_{\mathsf{NameEligibilityProof}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NameEligibilityProof}^{\uparrow}$ packet is generated by displaying the proposed name, source row, classifier row, proof witness, strength row, audit row, gap-or-refusal row, kernel-route row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming in a finite BHist carrier. The classifier licenses a name only when the source, proof, audit, strength, and refusal boundaries remain attached to the same packet. Its ledger story is eligibility before use: the name cannot be selected by a host object or by resemblance in prose, because every accepted read must pass through the explicit proof and kernel-route surfaces already recorded in the carrier.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NameEligibilityProof}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter binds the carrier, NameCert-obligation, strength-non-inflation, kernel-route-non-escape, and obligation-package blocks as the finite obligation surface for $\mathsf{NameEligibilityProof}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NameEligibilityProof}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim host-selected objects, hidden source supply, prose classifiers, unaudited strength promotion, off-route kernel acceptance, or erased refusals.

Upgrade path: Scoped closure requires a theorem-level Lean witness for the eligibility carrier, strength boundary, kernel-route boundary, and obligation package against finite-history transport.

Chapter 1310

A Concrete Naming Certificate for $\mathsf{UseRuleLedger}^{\uparrow}$

The $\mathsf{UseRuleLedger}^{\uparrow}$ packet realizes the certified-use reading of subsection 9462.29.7 as a finite NameCert carrier. It sits beside the machine-facing boundary of chapter 1221, the registry export surface of chapter 1216, the apophatic naming boundary of chapter 543, and the kernel audit witness of chapter 546. Those siblings expose machine routing, export control, refused supply, and accepted audit evidence; this packet binds use, rule-following, source/classifier rows, stability, ledger policy, public audit, transport, continuation, provenance, and local naming before a name-use is licensed.

Definition 1310.1 (Use-rule ledger carrier)**.** *A* $\mathsf{UseRuleLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (U, R, S, C, L, A, H, Q, P, N)$$

whose displayed rows have the following roles:

1. *U is the use-event row, recording the name, item used, local history, and requested public reading.*
2. *R is the rule row, recording the use rule by which the name may be applied to the item.*
3. *S is the source/classifier row, recording the displayed source field and classifier that check the item against the licensed name.*
4. *C is the stability row, recording the finite stability evidence under which the same use may be replayed.*
5. *L is the ledger row, recording the hidden-material policy, refused supply, and visible rule-following trace associated with the use.*
6. *A is the public audit row, recording the machine-interface boundary, registry-export surface, apophatic naming boundary, and kernel audit witness consulted by the use.*
7. *H records componentwise* hsame *transport for the use event, rule, source/classifier, stability, ledger, audit, continuation, provenance, and local naming rows.*
8. *Q records displayed* Cont *replay for the certified-use route.*
9. *P is the* Pkg *provenance over* $\mathsf{UseRuleLedger}^{\uparrow}$, $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{KernelAuditWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UseRuleLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for private meaning, bare reference, unclassified social habit, unledgered rule-following, hidden source supply, or a name-use licensed only by prose adjacency.

Theorem 1310.2 (Use-rule ledger NameCert obligations)**.** *For the carrier of Definition 1310.1, the local* $\mathsf{NameCert}_{\mathsf{UseRuleLedger}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for* $G = (U, R, S, C, L, A, H, Q, P, N)$*, use-event exposure through* U*, rule exposure through* R*, source/classifier exposure through* S*, stability exposure through* C*, ledger exposure through* L*, public audit exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* Q*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier into its ten displayed rows. The use event, rule, source/classifier, stability, and ledger rows are read before the public audit route is consumed, so a licensed use cannot be recovered from an audit citation alone.

The audit row records the machine-interface boundary, registry-export surface, apophatic naming boundary, and kernel audit witness. These siblings explain where the use is checked, exported, refused, or read back, but they do not replace the displayed use, rule, source/classifier, stability, and ledger rows.

Finally H, Q, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no hidden source coordinate and no prose-only licensing coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1310.3 (Certified-use route)**.** *Every accepted* $\mathsf{UseRuleLedger}^{\uparrow}$ *route that licenses a public name-use must expose the use-event row* U*, rule row* R*, source/classifier row* S*, stability row* C*, ledger row* L*, public audit row* A*, transport row* H*, continuation row* Q*, provenance row* P*, and local naming row* N*. A route that omits any one of these rows is at most a projection of the use-rule ledger, not the accepted carrier of Definition 1310.1.*

Proof. Begin with the displayed packet $G = (U, R, S, C, L, A, H, Q, P, N)$. The name-use is located by U, governed by R, constrained by S, stabilized by C, and ledgered by L before any consumer can read it as public.

The public audit row A then records the sibling boundary surfaces. It can route the packet to machine-facing, registry-facing, apophatic, and kernel-audit consumers, but those consumers only see the use after the preceding rows have already licensed it.

The structural rows H, Q, P, N preserve the same packet under transport, continuation replay, provenance, and naming. Omitting one of the displayed rows therefore changes the carrier into a projection rather than an accepted certified-use route. □

Theorem 1310.4 (Use-rule ledger nonescape boundary)**.** *No accepted* $\mathsf{UseRuleLedger}^{\uparrow}$ *packet may turn a missing source/classifier row, missing stability row, or missing ledger/audit row into licensed use by appealing to private meaning, social repetition, machine export, registry presence, apophatic refusal, or kernel readback alone.*

Proof. Inspect the licensing rows S, C, L, A. The source/classifier row states what item is being used under what name; the stability row states when the same use is replayable; the ledger row records hidden-material and rule-following policy; and the audit row states how the public route is checked.

Private meaning and social repetition are not coordinates of the carrier. Machine export, registry presence, apophatic refusal, and kernel readback appear only through A and P, where they are dependencies of the displayed packet rather than substitutes for the licensing rows.

Thus a missing source/classifier, stability, ledger, or audit row cannot be supplied by another route. The nonescape boundary is the requirement that certified use remains attached to the finite rows that license it. □

Theorem 1310.5 (Use-rule ledger TasteGate obligation)**.** *Any TasteGate admission for* $\mathsf{UseRuleLedger}^{\uparrow}$ *must preserve nontriviality and field faithfulness for the displayed rows* $U, R, S, C, L, A, H, Q, P, N$*. In particular, admission must distinguish use event, rule, source/classifier, stability, ledger, public audit, transport, continuation, provenance, and local naming rather than collapsing certified use into a single registry or machine-interface row.*

Lean4 ✓ `BEDC.Derived.UseRuleLedgerUp.UseRuleLedgerTasteGate single carrier alignment`

Proof. A TasteGate admission reads the carrier as a candidate public object. Nontriviality requires the carrier to exhibit a real row packet, while field faithfulness requires the named fields to remain separately readable by downstream consumers.

Apply this to $G = (U, R, S, C, L, A, H, Q, P, N)$. Collapsing the use event into the rule, the source/classifier into the stability row, or the audit route into machine or registry export would erase a displayed coordinate of certified use.

Therefore the TasteGate obligation is to keep the ten rows distinct while allowing their joint package to be named. The Lean side must provide $\mathsf{Nontrivial}\ \mathsf{UseRuleLedger}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{UseRuleLedger}^{\uparrow}$ instances in `lean4/BEDC/Derived/UseRuleLedgerUp/TasteGate.lean` before any checked marker for this carrier can be cited. □

Falsifiable prediction: Any accepted $\mathsf{UseRuleLedger}^{\uparrow}$ route that licenses a name-use without the displayed source/classifier, stability, and ledger rows refutes the carrier.

Independence witness: machineinterfaceboundary, registryexportsurface, apophaticname, kernelauditwitness: none is bijective because $\mathsf{UseRuleLedger}^{\uparrow}$ binds certified use, rule-following, source/classifier, stability, and ledger rows as one name-use packet rather than only routing exports, refusing supply, or reading audit evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{UseRuleLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UseRuleLedger}^{\uparrow}$ packet is generated from displayed use rows, rule rows, witness rows, audit rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming in one finite BHist carrier. The classifier licenses rule-following only when the use, rule, witness, and audit coordinates are present together; social repetition and prose adjacency are not enough. The ledger story is a public rule route: a certified-use consumer reads the same finite packet through transport and continuation rows, while hidden source supply, private meaning, and unledgered rule-following remain outside the carrier inventory.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UseRuleLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, certified-use-route, nonescape-boundary, TasteGate-obligation, independence witness, and existing closure annotations for $\mathsf{UseRuleLedger}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UseRuleLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim private meaning, bare reference, unclassified social habit, unledgered rule-following, hidden source supply, or prose-only name-use licensing.

Upgrade path: To reach obligationClosure, the use-rule carrier, route theorem, and nonescape boundary must be encoded as checked finite-row obligations.

Chapter 1311

A Concrete Naming Certificate for $\mathsf{StructuralAdjacency}^{\uparrow}$

The $\mathsf{StructuralAdjacency}^{\uparrow}$ packet realizes the adjacency claim of subsection 9462.29.1 as a finite NameCert construction. It sits beside the tradition-comparison boundary of chapter 1268, the primitive distinction ledger, the observer distinction trace, the finite kernel category route, and the epistemic witness packet. Those siblings regulate tradition comparison, primitive marks, observer trace, categorical kernel rows, and witness admission; this packet binds genealogy labels, BEDC role rows, adjacency witnesses, transport, continuation, provenance, and local naming before any two inherited claims may be read as structurally adjacent.

Definition 1311.1 (Structural adjacency carrier)**.** *A* $\mathsf{StructuralAdjacency}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, R, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the genealogy-label row, recording the two inherited claim labels, source families, question surfaces, and local comparison prompts.*
2. *R is the BEDC role-row classifier, recording the carrier, relation, classifier, ledger, certificate, gap, or T-position on which each claim lands.*
3. *A is the adjacency-witness row, recording that the two labels land on the same BEDC role row without importing doctrinal identity or tradition membership.*
4. *H records componentwise* hsame *transport for genealogy labels, role rows, adjacency witnesses, continuation routes, provenance, and local naming.*
5. *C records displayed* Cont *replay for the route from genealogy labels through role-row classification and adjacency witness inspection.*
6. *P is the* Pkg *provenance over* $\mathsf{StructuralAdjacency}^{\uparrow}$, $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$, $\mathsf{ObserverDistinctionTrace}^{\uparrow}$, $\mathsf{KernelCategory}^{\uparrow}$, $\mathsf{EpistemicWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}$ *row.*

The carrier has no coordinate for prose resemblance alone, inherited doctrinal agreement, host-level semantic identity, or adjacency export outside the displayed packet $S = (G, R, A, H, C, P, N)$.

Lean4 ✓ `BEDC.Derived.StructuralAdjacencyUp.StructuralAdjacencyTasteGate single carrier alignment`

Theorem 1311.2 (Structural adjacency NameCert obligations)**.** *For the carrier of Definition 1311.1, the local* $\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}$ *surface consists exactly of the following five rows: carrier admission for* $S = (G, R, A, H, C, P, N)$*; role-row classifier stability through* R *under componentwise transport; adjacency exactness through* A *for shared BEDC landing; genealogy-label non-collapse through* G*; and ledger non-escape through* H, C, P, N*.*

Proof. Project the finite carrier. Carrier admission supplies the displayed BHist packet, while the genealogy row G keeps the two inherited labels present as distinct labels before any BEDC landing is compared.

Read the role classifier and adjacency witness together. The row R records the BEDC role rows available to each label, and A certifies adjacency only when those role rows coincide as a BEDC site. Componentwise transport in H preserves the classifier row instead of replacing it by prose similarity.

Finally apply the route, provenance, and local-name rows. The continuation row C, package row P, and local NameCert row N name the same packet, so the ledger cannot export adjacency without the displayed role row, genealogy labels, and exact adjacency witness. □

Theorem 1311.3 (Structural adjacency independence witness)**.** *The nearest siblings of* $\mathsf{StructuralAdjacency}^{\uparrow}$ *are* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$*,* $\mathsf{PrimitiveDistinctionLedger}^{\uparrow}$*,* $\mathsf{ObserverDistinctionTrace}^{\uparrow}$*,* $\mathsf{KernelCategory}^{\uparrow}$*, and* $\mathsf{EpistemicWitness}^{\uparrow}$*. No bijection preserves all carrier rows between any one of those siblings and* $S = (G, R, A, H, C, P, N)$*: each sibling lacks at least one of genealogy-pair labeling, BEDC role-row classification for two claims, adjacency witness exactness, or the local* $\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}$ *row.*

Proof. Compare the displayed rows. $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ records source, landing, rejected surplus, and structural distinction for a comparison; it does not store a two-claim adjacency witness as the public row of the packet.

The primitive distinction ledger and observer distinction trace isolate mark distinction and observer trace structure. The finite kernel category route and epistemic witness packet isolate category-shaped operations and witness admission. None of these carries the pair of genealogy labels, shared BEDC role-row classifier, adjacency witness, transport, route, provenance, and local StructuralAdjacency name in one carrier.

A row-preserving bijection would have to preserve G, R, A, H, C, P, N jointly. Since every listed sibling omits at least one of these rows or assigns it a different obligation role, no such bijection preserves all carrier rows. □

Falsifiable prediction: Any accepted structural-adjacency route whose packet lacks a BEDC role row is rejected by the carrier classifier.

Independence witness: traditioncomparisonboundary, primitivedistinctionledger, observerdistinctiontrace, kernelcategory, epistemicwitness: none is bijective because structural adjacency alone binds genealogy-pair labels, BEDC role-row classification, adjacency-witness exactness, and local NameCert rows in one packet.

The Lean follow-up must provide Nontrivial $\mathsf{StructuralAdjacency}^{\uparrow}$ and FieldFaithful $\mathsf{StructuralAdjacency}^{\uparrow}$ instances in `lean4/BEDC/Derived/StructuralAdjacencyUp/TasteGate.lean`. Those instances must preserve G, R, A, H, C, P, N as separate displayed rows before BEDC.Derived.StructuralAdjacencyUp.StructuralA may be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}$.

Constructive story: A $\mathsf{StructuralAdjacency}^{\uparrow}$ packet is generated from genealogy-label, BEDC role-row, adjacency-witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1311.1, Definition 1311.2, and Definition 1311.3 bind structural adjacency to a finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StructuralAdjacency}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, public export closure, or the correctness of any inherited tradition's doctrine.

Upgrade path: Lean closure requires Nontrivial StructuralAdjacencyUp and FieldFaithful StructuralAdjacencyUp instances in `lean4/BEDC/Derived/StructuralAdjacencyUp/TasteGate.lean`.

Chapter 1312

A Concrete Naming Certificate for $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$

The $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ packet realizes the local-closure and open-continuation boundary of section 9414.25 as finite NameCert data. It sits beside the open-fit carrier of chapter 1131, the finite refutation carrier of chapter 1169, and the continuation countermodel packet of chapter 1183. Those siblings expose open fit, finite defeat, and continuation mismatch; this chapter binds the local empirical closure row to an explicit nonfinality ledger before any consumer may read finite settlement as final empirical closure.

Definition 1312.1 (Local empirical closure gate carrier)**.** *A* $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, F, J, O, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the fixed finite source trace, probe bundle, and classifier row for the tested empirical packet;*
2. *F is the local fit row certifying whether the model signature and observed signature agree on S under the displayed classifier;*
3. *J is the local closure judgment row, exposing that the finite inquiry over S has been settled;*
4. *O is the global openness row, recording that no row of the packet supplies a final induction oracle over all admissible continuations;*
5. *L is the nonfinality ledger, naming the refused final-fit, hidden-continuation, and oracle-transfer reads;*
6. *H records componentwise* hsame *transport for S, F, J, O, L, C, P, N;*
7. *C records displayed* Cont *replay for finite source inspection, local fit reading, local closure reading, global openness reading, and nonfinality-ledger inspection;*
8. *P records* Pkg *provenance over* $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$, $\mathsf{RealityConstrainedOpenFit}^{\uparrow}$, $\mathsf{FiniteObservationRefutat}$ BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalEmpiricalClosureGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted empirical confirmation, total future-fit quantification, hidden continuation enumeration, host real equality of physical trajectories, or a final induction oracle.

Lean4 ✓ `BEDC.Derived.LocalEmpiricalClosureGateUp.LocalEmpiricalClosureGateTasteGate single carrier alignment`

Theorem 1312.2 (Local empirical closure gate NameCert obligations)**.** *For the carrier of Definition 1312.1, the local* $\mathsf{NameCert}_{\mathsf{LocalEmpiricalClosureGate}^{\uparrow}}$ *surface consists exactly of five empirical obligation rows: finite source admission through S, local fit settlement through F, local closure certification through J, global openness retention through O, and explicit nonfinality refusal through L. The structural rows H, C, P, N only transport, replay, source, and name those five rows.*

Proof. Project the carrier to the empirical rows S, F, J, O, L. The finite source row fixes the observed trace and classifier; the local fit row records the model/observation comparison; and the closure-judgment row states that this finite inquiry is settled.

Read the openness and nonfinality rows next. The row O preserves the open-continuation boundary of Definition 9414.354, while L records the refused final-fit and oracle-transfer reads.

Finally inspect the structural rows. The transport, continuation, provenance, and local naming data preserve only the displayed rows already present in G. They do not add a sixth empirical coordinate, so the five named rows exhaust the NameCert obligation surface. □

Theorem 1312.3 (Local empirical closure gate nonfinality)**.** *Every accepted* $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ *consumer may read the finite local closure row J only together with the global openness row O and the nonfinality ledger L. It cannot derive a final induction oracle, a universal future-fit certificate, or a continuation enumeration from the finite local-closure packet alone.*

Proof. Begin with the local closure theorem of Definition 9414.356. It separates finite fit settlement over a displayed trace from universal closure over all future observation histories.

Apply the carrier inventory of Definition 1312.1. The only positive settlement row is J, and it is paired in the same accepted packet with O and L, which record global openness and the refused oracle transfer.

The structural rows H, C, P, N cannot synthesize missing empirical data; they only preserve, replay, source, and name the displayed packet. Therefore an accepted consumer receives local closure together with its nonfinality ledger and has no route to a final induction oracle. □

Falsifiable prediction: A consumer that derives a final-induction-oracle row from S, F, J, H, C, P, N without the openness row O and the nonfinality ledger L refutes the $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ carrier.

Independence witness: openphysicalfit, finiteobservationrefutation, continuationcountermodel, realityconstrainedopenfit: no listed sibling is bijective to $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ because this packet binds the finite source, local fit, local closure, global openness, and nonfinality ledger as one NameCert surface. OpenPhysicalFit exposes open-fit routing without the local-closure judgment; FiniteObservationRefutation exposes mismatch defeat without the closure/openness pair; ContinuationCountermodel exposes a continuation separation without local settlement; RealityConstrainedOpenFit exposes open fit without the explicit local-closure row plus nonfinality ledger.

The Lean follow-up for this chapter must provide explicit $\mathsf{Nontrivial}\ \mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ instances in `lean4/BEDC/Derived/LocalEmpiricalClosureGateUp/T` The field-faithfulness witness must preserve $S, F, J, O, L, H, C, P, N$ as distinct rows before `BEDC.Derived.LocalEm` can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalEmpiricalClosureGate}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalEmpiricalClosureGate}^{\uparrow}$ packet is generated from finite source, local fit, local closure, global openness, nonfinality ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocalEmpiricalClosureGate}\uparrow}$, seed)

Scope closed: Definition 1312.1, Definition 1312.2, and Definition 1312.3 specify the finite local empirical closure gate, its NameCert obligations, and its nonfinality boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocalEmpiricalClosureGate}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted empirical confirmation, total future-fit quantification, hidden continuation enumeration, host physical-trajectory equality, or final induction oracle construction.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, nonfinality exactness, and TasteGate Nontrivial and FieldFaithful obligations for the same finite packet.

Chapter 1313

A Concrete Naming Certificate for $\mathsf{UseProcessBoundary}^{\uparrow}$

The $\mathsf{UseProcessBoundary}^{\uparrow}$ packet realizes the certified-use and process-priority surface of subsection 9462.29.7 and subsection 9462.29.8 as one finite NameCert boundary. It sits beside $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, and $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$: those siblings compare inherited tradition rows, register tradition entries, admit package claims, and check registry export consistency, while this packet binds certified use to process priority before either a tradition comparison or a registry export can read meaning as a stable claim.

The export-boundary row E is consumed directly by chapter 1224: registry export may read meaning only after the certified-use row and the process-priority row have been exposed on the same packet. The adjacent tradition consumers, chapter 1268 and chapter 1267, receive the same mediated surface rather than a private meaning row, so the registry-export lattice begins from certified use under process priority.

$\mathsf{UseProcessLimitLedger}^{\uparrow}$ in chapter 1315 consumes this boundary when certified use is read against formal process-limit refusal: $\mathsf{UseProcessBoundary}^{\uparrow}$ supplies the certified-use and process-priority rows, while $\mathsf{UseProcessLimitLedger}^{\uparrow}$ records the limit and refusal ledger that consumes them.

Definition 1313.1 (Use-process-boundary carrier)**.** *A* $\mathsf{UseProcessBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, P, R, E, H, C, K, N)$$

with the following displayed rows:

1. *U is the certified-use row, recording the public use of the name together with the certificate route that licenses that use;*
2. *P is the process-priority row, recording the* Cont *process through which use is replayed before any object-facing or registry-facing conclusion is read;*
3. *R is the rule-audit row, recording the finite distinction between licensed use, bare habit, private image, and unregistered philosophical claim;*
4. *E is the export-boundary row, recording the routes by which* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, *and* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *may consume the packet;*

5. *H records componentwise* hsame *transport for certified use, process priority, rule audit, export boundary, replay, provenance, and local naming;*
6. *C records displayed* Cont *replay of the same certified-use and process-priority rows for downstream consumers;*
7. *K is the* Pkg *provenance over* $\mathsf{UseProcessBoundary}^{\uparrow}$, $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for private meaning, social habit as an oracle, host equality, doctrine identity, total truth, or registry export without the displayed use and process rows.

Theorem 1313.2 (Use-process-boundary NameCert obligations)**.** *For the carrier of Definition 1313.1, the local* $\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (U, P, R, E, H, C, K, N)$, certified-use exposure through U, process-priority exposure through P, rule-audit exposure through R, export-boundary exposure through E, componentwise transport and replay through H, C, provenance through K, and local naming through N.*

Lean4 ✓ `BEDC.Derived.UseProcessBoundaryUp.UseProcessBoundaryTasteGate single carrier alignment`

Proof. Project the finite packet of Definition 1313.1. Its rows display certified use, process priority, rule audit, export boundary, transport, replay, provenance, and local naming.

The first four rows are the obligation surface before any downstream registry or tradition consumer can read the packet. The rows H and C preserve that same surface under componentwise transport and displayed replay.

Finally K and N cite and name the same displayed rows. Since the carrier contains no further coordinate, these obligations exhaust the local $\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}$ surface. □

Theorem 1313.3 (Use-process-boundary certified-use nonescape)**.** *Every accepted* $\mathsf{UseProcessBoundary}^{\uparrow}$ *consumer that exports meaning, tradition comparison, claim admission, or registry consistency must expose the certified-use row U, process-priority row P, rule-audit row R, export-boundary row E, transport row H, replay row C, provenance row K, and local naming row N. No accepted consumer receives private meaning, bare social habit, doctrine identity, host equality, total truth, or registry export without those rows.*

Proof. Read the consumer through the carrier. The certified-use row U licenses the use, while the process-priority row P keeps the use on a displayed continuation before export.

The rule-audit and export-boundary rows R, E determine which downstream tradition and registry siblings may consume the packet. The structural rows H, C, K, N transport, replay, source, and locally name exactly that route.

The excluded readings are absent from the coordinate list of Definition 1313.1. A consumer claiming one of them would therefore require a field that the accepted boundary packet does not contain. □

Theorem 1313.4 (Use-process-boundary process priority)**.** *Every accepted* $\mathsf{UseProcessBoundary}^{\uparrow}$ *export reaches a tradition or registry consumer only after the certified-use row U has been replayed through the process-priority row P and the displayed* Cont *row C. Thus meaning-as-certified-use is process-mediated before* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, *or* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *may consume it.*

Proof. Start with the order of the carrier rows. The use row U is not followed by an immediate object-export row; it is paired with the process-priority row P and the replay row C.

The export-boundary row E names only consumers of that mediated route. It cannot bypass P or C without ceasing to be the export row of the displayed packet.

Consequently each sibling consumer reads certified use through a process route. The theorem asserts process priority for the finite NameCert boundary, not an ambient metaphysical completeness claim. □

Theorem 1313.5 (Use-process-boundary sibling independence)**.** *The* $\mathsf{UseProcessBoundary}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, *or* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$. *Each sibling is consumed through the export-boundary row E, but none carries the joint certified-use, process-priority, rule-audit, export-boundary, transport, replay, provenance, and local naming packet of* $\mathsf{UseProcessBoundary}^{\uparrow}$.

Proof. Compare the displayed rows. $\mathsf{TraditionComparisonBoundary}^{\uparrow}$ separates inherited tradition rows from BEDC landing rows, and $\mathsf{TraditionRegistry}^{\uparrow}$ records admitted tradition entries.

$\mathsf{ClaimRegistryLayer}^{\uparrow}$ controls package-level claim admission, while $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ checks status consistency before machine-facing registry export. These siblings supply downstream or adjacent routes, not the certified-use and process-priority boundary itself.

The $\mathsf{UseProcessBoundary}^{\uparrow}$ packet binds U, P, R, E, H, C, K, N in one finite carrier. A row-preserving bijection would have to preserve certified use, process priority, rule audit, export boundary, transport, replay, provenance, and local naming simultaneously, which none of the four siblings displays as its carrier surface. □

Falsifiable prediction: If a UseProcessBoundaryUp packet exports a tradition or registry claim while deleting either certified use or process priority, the carrier is refuted.

Independence witness: traditioncomparisonboundary, traditionregistry, claimregistrylayer, registryexportconsistencygate: none is bijective because UseProcessBoundaryUp binds certified use, process priority, rule audit, export boundary, transport, replay, provenance, and local naming in one finite packet.

The Lean follow-up requires $\mathsf{Nontrivial\ UseProcessBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ UseProcessBoundary}^{\uparrow}$ instances in `lean4/BEDC/Derived/UseProcessBoundaryUp/TasteGate.lean`. Those instances must preserve U, P, R, E, H, C, K, N as distinct displayed fields before any checked marker can cite `BEDC.Derived.UseProcessBoundaryUp.TasteGate`.

Closure status for $\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UseProcessBoundary}^{\uparrow}$ packet is generated from use rows, process rows, rule-ledger rows, certified-use witnesses, registry and machine-interface rows, philosophical-boundary rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The carrier starts from finite BHist uses and admits only those process routes whose rule, registry, and machine-facing coordinates remain visible. Its classifier preserves use and process as distinct but linked surfaces: certified use may be read only after the process route and rule ledger are replayed, while private meaning, doctrine identity, and social habit as an oracle have no carrier slot.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, certified-use-nonescape, process-priority, sibling-independence, Lean-follow-up boundary, and existing closure annotation for $\mathsf{UseProcessBoundary}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UseProcessBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim private meaning, social habit as an oracle, host equality, doctrine identity, total truth, or registry export without displayed use and process rows.

Upgrade path: To reach obligationClosure, the use-process carrier and certified-use boundary theorems must be checked as finite replay and transport obligations.

Chapter 1314

A Concrete Naming Certificate for $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$

The $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ packet realizes the programme-strength gradient of subsection 9462.32.2 as a finite ledger whose rows prevent over-reporting across naming seed, paper certificate, machine-checked certificate, and conservative bridge. It refines the strength ledger of chapter 1305 by making each reported grade consume its displayed closure, formal, bridge, downgrade, interface, harness, registry, and empirical-boundary rows before the public strength report is readable.

Definition 1314.1 (Programme strength gradient ledger carrier)**.** *A* $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, L, T, F, B, M, V, D, U, E, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the claim-site row naming the theorem, definition, chapter, bridge, package row, or machine-facing output being graded.*
2. *L is the layer row naming the programme layer and lower-layer dependencies consumed by the claim site.*
3. *T is the theory-closure row, recording seed, obligation, scoped, public, bridged, or mature closure as the paper-side grade.*
4. *F is the formal-status row, recording unformalized, formal target, encoded definition, checked scaffold, checked theorem, audit-clean, axiom-clean, or bridge-checked status as the verification-side grade.*
5. *B is the bridge row, recording absent, paper-only, or checked conservative-bridge status.*
6. *M is the machine-interface row, exposing the machine-facing boundary through which any checked output or kernel readback is reported.*
7. *V is the verified-output harness row, exposing checker-facing acceptance for reported machine output.*
8. *D is the status-downgrade row, recording the lower report that must be emitted when T, F, B, M, or V is weaker than the requested grade.*

9. *U is the upgrade-registry row, recording the next displayed obligation rather than silently raising the report.*
10. *E is the formal-constant empirical-value boundary row, separating formal strength from empirical value claims.*
11. *R is the refusal row for over-strong reports, hidden bridge claims, unchecked machine claims, erased downgrade rows, and empirical-value substitution.*
12. *H records componentwise* hsame *transport for all displayed rows.*
13. *C records displayed* Cont *replay from claim site to layer, closure, formal status, bridge, machine interface, verified output, downgrade, upgrade registry, empirical boundary, and refusal rows.*
14. *P records* Pkg *provenance over* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$, $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{VerifiedOutputHarness}^{\uparrow}$, $\mathsf{StatusDowngrade}^{\uparrow}$, $\mathsf{UpgradeRegistry}^{\uparrow}$, $\mathsf{FormalConstantEmpiricalValueBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for reporting a claim above its displayed theory closure, formal status, bridge row, machine-interface row, verified-output harness, downgrade row, upgrade-registry row, empirical boundary, or refusal row.

Lean4 ✓ `BEDC.Derived.ProgrammeStrengthGradientLedgerUp.ProgrammeStrengthGradientLedgerTasteGate single carrier alignment`

Theorem 1314.2 (Programme strength gradient ledger obligations)**.** *For the carrier of Definition 1314.1, the local* $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (Q, L, T, F, B, M, V, D, U, E, R, H, C, P, N)$, layer exposure through L, theory-closure exposure through T, formal-status exposure through F, bridge exposure through B, machine-interface exposure through M, verified-output exposure through V, downgrade exposure through D, upgrade-registry exposure through U, empirical-boundary exposure through E, refusal exposure through R, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet. The claim site Q is not yet a public strength report; it becomes reportable only after the layer, closure, formal, bridge, machine-interface, verified-output, downgrade, registry, empirical-boundary, and refusal rows have all been read.

Use the sibling packets as row sources. The programme-strength ledger supplies the base strength discipline, the machine-interface boundary supplies checked-output routing, the verified-output harness supplies checker-facing acceptance, the status-downgrade row supplies the lower report, the upgrade registry supplies the next obligation, and the formal-constant empirical-value boundary prevents formal strength from becoming empirical value.

Transport, replay, provenance, and local naming preserve the same finite packet. Since no further coordinate can raise T, F, B, M, V, D, U, E, or R, the listed rows exhaust the local NameCert surface. □

Theorem 1314.3 (Programme strength gradient no over-reporting)**.** *An accepted* ProgrammeStrengthGradientLedg *packet cannot export a reported strength above the minimum grade licensed by its displayed theory-closure row T, formal-status row F, bridge row B, machine-interface row M, verified-output harness row V, downgrade row D, upgrade-registry row U, empirical-boundary row E, and refusal row R. A consumer that asks for a stronger grade receives the downgrade row rather than a promoted report.*

Proof. Read T, F, and B first. These rows record the independent paper, verification, and bridge axes, so no one axis can raise the others.

Read M and V next. A machine-facing report is available only through the displayed interface and checker-facing harness; unchecked machine output is therefore not a higher grade inside the packet.

Finally read D, U, E, and R. The downgrade row supplies the lower report, the upgrade registry names the next obligation, the empirical-boundary row blocks value substitution, and the refusal row names over-strong reports. The carrier has no remaining coordinate that can convert these blockers into discharged obligations. □

Theorem 1314.4 (Programme strength ledger falsifiable prediction)**.** *Any accepted* ProgrammeStrengthGradientLed *report that grades a claim above its displayed closure row, formal-status row, bridge row, machine-interface row, verified-output row, downgrade row, upgrade-registry row, empirical-boundary row, or refusal row refutes the carrier.*

Proof. Assume an accepted report exports a stronger grade. By Definition 1314.2, every accepted report must expose T, F, B, M, V, D, U, E, and R before the local name is consumed.

Apply Definition 1314.3. The strongest readable grade is bounded by those displayed rows, and a stronger request must be routed to the downgrade row rather than promoted.

Thus the over-strong report cannot be a consumer of the accepted carrier. It is a falsifying observation for the purported $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ packet. □

Falsifiable prediction: Any accepted programme-strength ledger that reports a claim above its displayed closure, formal, or bridge row refutes the carrier.

Independence witness: machineinterfaceboundary, verifiedoutputharness, statusdowngrade, upgraderegistry, formalconstantempiricalvalueboundary: none is bijective with $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ because the gradient ledger simultaneously carries claim site, layer, theory closure, formal status, bridge status, machine-interface route, verified-output harness, downgrade, upgrade registry, empirical boundary, refusal, transport, replay, provenance, and local NameCert rows.

Lean follow-up requires `Nontrivial ProgrammeStrengthGradientLedgerUp` and `FieldFaithful ProgrammeStrengthGradientLedgerUp` in `lean4/BEDC/Derived/ProgrammeStrengthGradientLedgerUp/TasteG` The target `BEDC.Derived.ProgrammeStrengthGradientLedgerUp.taste_gate` must preserve distinct rows for $Q, L, T, F, B, M, V, D, U, E, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ packet is generated from theory-closure, formal-status, bridge, machine-interface, verified-output, downgrade, upgrade-registry, empirical-boundary, refusal, transport, continuation, provenance, and local naming rows inside one finite BHist carrier. The classifier reads programme strength as a gradient across these displayed coordinates, not as a single status label. The ledger story is anti-over-reporting: each public report must be generated from the current closure, verification, bridge, machine, and empirical rows, while downgrade and refusal coordinates remain visible when an advertised strength exceeds what the carrier can support. Componentwise hsame and Cont replay preserve the same rows for every consumer.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}}, \mathsf{obligation})$

Scope closed: This chapter binds the displayed carrier of Definition 1314.1, the obligation theorem Definition 1314.2, the no-over-reporting theorem Definition 1314.3, the falsifiable prediction Definition 1314.4, and the TasteGate alignment marker for $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ as the obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim reports above displayed theory closure, formal status, bridge row, machine row, verified output, downgrade, registry, empirical boundary, or refusal row.

Upgrade path: To reach scopedClosure, the strength-gradient rows and no-over-reporting theorem must be represented by checked finite-carrier projections.

Chapter 1315

A Concrete Naming Certificate for $\mathsf{UseProcessLimitLedger}^{\uparrow}$

The $\mathsf{UseProcessLimitLedger}^{\uparrow}$ packet realizes the use, continuation-process, and formal-limit boundary of subsection 9462.29.7, subsection 9462.29.8, and subsection 9462.29.9 as one finite export-boundary carrier. It sits beside the registry export surface of chapter 1216, the claim registry layer of chapter 1209, the kernel acceptance witness of chapter 549, and the continuation counter-model of chapter 1183. Those siblings expose export control, claim admission, accepted-declaration evidence, and failed continuation matching; this packet binds certified use, continuation route, formal-limit refusal, transport, replay, provenance, and local naming before any philosophy-facing consumer may read use as exported BEDC content.

Definition 1315.1 (Use-process-limit ledger carrier)**.** *A* $\mathsf{UseProcessLimitLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (U, C, L_f, H, R, P, N)$$

with seven displayed rows:

1. *U is the certified-use row, recording the public use of the named item together with the* $\mathsf{NameCert}$ *legitimacy data that separates licensed use from private image, bare reference, or unregistered habit;*
2. *C is the continuation-process row, recording the displayed* Cont *route through which the certified use is replayed as a process before any export is accepted;*
3. *L_f is the formal-limit row, recording refusal to internalize total truth, host equality, halting decision, or a closed open-meta-loop witness;*
4. *H is the componentwise* hsame *transport row for certified use, continuation process, formal-limit refusal, replay, provenance, and local naming;*
5. *R is the displayed* Cont *replay row that shows downstream consumers the same use/process/refusal route rather than a detached use token;*
6. *P is the* Pkg *provenance row over* $\mathsf{NameCert}$, Cont, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{KernelAcceptanceWitness}^{\uparrow}$, *and the source vision of section 9462.29;*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UseProcessLimitLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unlicensed use oracle, process-free export, total truth predicate, host equality, closed meta loop, or tradition identity collapse.

Lean4 ✓ `BEDC.Derived.UseProcessLimitLedgerUp.UseProcessLimitLedgerTasteGate single carrier alignment`

Theorem 1315.2 (Use-process-limit ledger NameCert obligations)**.** *For the carrier of Definition 1315.1, the local* $\mathsf{NameCert}_{\mathsf{UseProcessLimitLedger}^{\uparrow}}$ *surface has exactly three primary obligation rows before transport, replay, and provenance are consumed: certified use through* U*, continuation process through* C*, and formal-limit refusal through* L_f*. The rows* H, R, P, N *transport, replay, source, and locally name that three-row surface without adding a fourth philosophy-side authority.*

Proof. Project the finite packet $L = (U, C, L_f, H, R, P, N)$. The first three rows display the certified-use, continuation-process, and formal-limit-refusal obligations that a NameCert consumer is allowed to inspect.

The transport row H and replay row R preserve those same displayed rows. They do not add a private meaning row, a process-free route, or a host-side completion coordinate.

Finally P records the registry, claim, kernel-acceptance, and source-vision provenance, while N gives the local NameCert row. Since both rows cite the same three primary obligations, the listed surface exhausts the local NameCert obligations. □

Theorem 1315.3 (Use-process-limit ledger export boundary)**.** *Every accepted* $\mathsf{UseProcessLimitLedger}^{\uparrow}$ *export presents certified use, continuation process, and formal-limit refusal as one finite packet. An export that presents* U*,* C*, and* L_f *as three detachable exports is rejected by the carrier, even if each detached row is individually named.*

Proof. Read the export through the carrier, not through isolated row names. The rows U, C, L_f are fields of the same finite packet, and the downstream route reads them only through the shared transport, replay, provenance, and local naming rows H, R, P, N.

If a proposed export detaches certified use from process, it deletes the route that makes use replayable. If it detaches use and process from formal-limit refusal, it allows the consumer to read licensed use as total truth, host equality, halting decision, or closed meta-loop completion.

Thus an accepted export must preserve the packet boundary. Three separately named exports can cite parts of the vocabulary, but they are not an export of Definition 1315.1. □

Theorem 1315.4 (Use-process-limit ledger sibling independence)**.** *The* $\mathsf{UseProcessLimitLedger}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{RegistryExportSurface}^{\uparrow}$*,* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*,* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$*, reflective-stability packets, or* $\mathsf{ContinuationCountermodel}^{\uparrow}$*. Each sibling omits at least one of certified use, continuation-process route, formal-limit refusal, componentwise transport, displayed replay, provenance, and local naming as one carrier.*

Proof. Compare displayed rows. $\mathsf{RegistryExportSurface}^{\uparrow}$ controls registry export, $\mathsf{ClaimRegistryLayer}^{\uparrow}$ admits or refuses package claims, $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ records accepted kernel declarations, reflective-stability packets track stability of reflection routes, and $\mathsf{ContinuationCountermodel}^{\uparrow}$ records a continuation mismatch.

None of those siblings simultaneously carries U, C, L_f, H, R, P, N. In particular, an export surface can omit certified use, a claim layer can omit the continuation route, a kernel witness can omit the philosophy-side formal-limit refusal, a stability packet can omit the local use ledger, and a countermodel can omit accepted export provenance.

Therefore no sibling is bijective to $\mathsf{UseProcessLimitLedger}^{\uparrow}$ while preserving local NameCert rows. A bijection would have to recover every displayed coordinate of the packet from a sibling that lacks at least one such coordinate. □

Falsifiable prediction: Any accepted use-process-limit export that treats certified use, continuation-process, and formal-limit refusal as three detachable exports refutes the packet.

Independence witness: registryexportsurface, claimregistrylayer, kernelauditwitness, reflectivestability, continuationcountermodel: none is bijective because each omits at least one of certified-use, process-route, formal-limit refusal, transport, replay, provenance, and local naming as one carrier.

The Lean follow-up must provide Nontrivial $\mathsf{UseProcessLimitLedger}^{\uparrow}$ and FieldFaithful $\mathsf{UseProcessLimitLedger}^{\uparrow}$ in `lean4/BEDC/Derived/UseProcessLimitLedgerUp/TasteGate.lean`. Those instances must preserve U, C, L_f, H, R, P, N as distinct displayed fields before any checked marker can cite `BEDC.Derived.UseProcess`

Closure status for $\mathsf{NameCert}_{\mathsf{UseProcessLimitLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{UseProcessLimitLedger}^{\uparrow}$ packet is generated from certified-use and NameCert-legitimacy data, a continuation-process route, a formal-limit refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UseProcessLimitLedger}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1315.1, Definition 1315.2, Definition 1315.3, and Definition 1315.4 scope the BHist, NameCert, Cont, hsame, and Pkg primitives to the finite rows U, C, L_f, H, R, P, N, including the existing carrier obligations, export boundary, and sibling-independence theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UseProcessLimitLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close whole-philosophy formal package checking, total truth, host equality, closed meta-loop completion, or collapsed tradition identity.

Upgrade path: Obligation closure requires checked carrier admission, field faithfulness, nontriviality, NameCert obligation coverage, export-boundary preservation, and sibling-independence coverage for U, C, L_f, H, R, P, N.

Chapter 1316

A Concrete Naming Certificate for $\mathsf{UseProcessLimit}^{\uparrow}$

The $\mathsf{UseProcessLimit}^{\uparrow}$ packet realizes the use, process, and limit-facing philosophy surface of subsection 9462.29.7, subsection 9462.29.8, and Definition 9462.1955 as one finite NameCert carrier. It sits beside the socket-use ledger of chapter 1044, the inward eligibility proof of chapter 1309, the machine-facing boundary of chapter 1221, and the apophatic naming boundary of chapter 543. Those siblings expose visible use, name eligibility, machine export control, and refusal-bearing naming; this packet binds certified use, rule audit, continuation process, and formal-limit refusal before any philosophy-facing export can read meaning as licensed use.

Definition 1316.1 (Use-process-limit carrier)**.** *A* $\mathsf{UseProcessLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, U_c, R, P_c, L, H, C, K, N_c)$$

with the following displayed rows:

1. *N is the name row, recording the public name whose meaning is read only through a certificate-governed use surface;*
2. *U_c is the certified-use row, recording the licensed use of N together with the source and classifier fields that make the use auditable;*
3. *R is the rule-checking row, recording the finite audit route that separates certificate-governed use from private image, bare reference, or ordinary social habit;*
4. *P_c is the process-continuation row, recording the displayed* Cont *process through which the use is replayed before object-facing export;*
5. *L is the formal-limit refusal row, recording that the packet does not internalize total truth, host equality, halting decision, or meta-loop closure;*
6. *H records componentwise* hsame *transport for the name, certified-use, rule-checking, process-continuation, formal-limit refusal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay of the same certified-use and process rows for downstream consumers;*
8. *K is the* Pkg *provenance over* $\mathsf{SocketUseLedger}^{\uparrow}$, $\mathsf{NameEligibilityProof}^{\uparrow}$, $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N_c is the local $\mathsf{NameCert}_{\mathsf{UseProcessLimit}^{\uparrow}}$ *row.*
The carrier has no coordinate for a private-meaning witness, social-habit oracle, host equality, total truth predicate, halting decision, or closed meta loop.

Lean4 ✓ `BEDC.Derived.UseProcessLimitUp.UseProcessLimitTasteGate single carrier alignment`

Definition 1316.2 (Use-process-limit classifier)**.** *A* $\mathsf{UseProcessLimit}^{\uparrow}$ *classifier from*

$$U = (N, U_c, R, P_c, L, H, C, K, N_c)$$

to

$$U' = (N', U'_c, R', P'_c, L', H', C', K', N'_c)$$

is a componentwise preservation map that carries the name row to the name row, certified use to certified use, rule audit to rule audit, continuation process to continuation process, formal-limit refusal to formal-limit refusal, hsame *transport to* hsame *transport,* Cont *replay to* Cont *replay,* Pkg *provenance to* Pkg *provenance, and local NameCert naming to local NameCert naming. A map that preserves the use row while deleting either the rule-checking row or the formal-limit refusal row is not a* $\mathsf{UseProcessLimit}^{\uparrow}$ *classifier.*

Theorem 1316.3 (Use-process-limit NameCert obligations)**.** *For the carrier of Definition 1316.1, the local* $\mathsf{NameCert}_{\mathsf{UseProcessLimit}^{\uparrow}}$ *surface has five displayed obligation rows before transport and provenance are consumed: name exposure through N, certified-use exposure through U_c, rule-checking exposure through R, process-continuation exposure through P_c, and formal-limit refusal exposure through L. The remaining rows H, C, K, N_c transport, replay, source, and locally name exactly that five-row surface.*

Proof. Project the finite packet $U = (N, U_c, R, P_c, L, H, C, K, N_c)$. The first five rows display the naming, use, rule audit, process, and refusal obligations before any downstream consumer can read the packet.

The transport row H and replay row C preserve those same displayed rows. They do not add a private meaning coordinate or a host-side completion coordinate.

Finally K records the sibling provenance, and N_c gives the local NameCert row. Since both rows cite the same five displayed obligations, the listed surface exhausts the local NameCert obligations. □

Theorem 1316.4 (Use-process-limit ledger nonescape)**.** *Every accepted* $\mathsf{UseProcessLimit}^{\uparrow}$ *consumer receives certified use, rule audit, process continuation, formal-limit refusal, transport, replay, provenance, and local naming through the finite rows $U_c, R, P_c, L, H, C, K, N_c$. The same consumer receives no private-meaning witness, social-habit oracle, host equality, total truth predicate, halting decision, or meta-loop closure.*

Proof. Read the consumer surface through the carrier. The rows U_c, R, P_c, L display the certified use, audit, continuation, and refusal components, while H, C, K, N_c transport, replay, source, and name them.

The forbidden readings are not fields of the packet. The name row N identifies the public name, but it does not supply a private witness, social oracle, host equality, truth predicate, halting decision, or closed meta loop.

Therefore any accepted consumer can receive only the displayed finite package. A route that claims one of the forbidden readings would require a coordinate absent from Definition 1316.1. □

Theorem 1316.5 (Use-process-limit certified-use process interface)**.** *Every accepted* $\mathsf{UseProcessLimit}^{\uparrow}$ *export factors meaning-as-certified-use through the displayed* Cont *process row* P_c *and replay row* C *before any export route may cite the local NameCert row* N_c*.*

Proof. Start with the certified-use row U_c. It licenses the use of N, but the carrier places it beside the process-continuation row P_c rather than beside an immediate object-export row.

The replay row C then displays the same continuation route for downstream consumers. This keeps meaning-as-certified-use attached to a visible process before the local naming row is consumed.

The local NameCert row N_c therefore names an already displayed process interface. An export that bypasses P_c or C is not an export of the accepted carrier. □

Falsifiable prediction: If a UseProcessLimitUp packet exports legitimate use while deleting either the rule-audit row or the formal-limit refusal row, the carrier is refuted.

Independence witness: socketuseledger, nameeligibilityproof, machineinterfaceboundary, apophaticname: none is bijective because UseProcessLimitUp simultaneously binds certified use, rule audit, process continuation, formal-limit refusal, transport, replay, provenance, and local naming.

The Lean follow-up requires Nontrivial $\mathsf{UseProcessLimit}^{\uparrow}$ and FieldFaithful $\mathsf{UseProcessLimit}^{\uparrow}$ instances in `lean4/BEDC/Derived/UseProcessLimitUp/TasteGate.lean`. Those instances must preserve $N, U_c, R, P_c, L, H, C, K, N_c$ as distinct displayed fields before any checked marker can cite `BEDC.Derived.UseProcessLimitUp.TasteGate`.

Closure status for $\mathsf{NameCert}_{\mathsf{UseProcessLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UseProcessLimit}^{\uparrow}$ packet is generated from finite use rows, process-step rows, certified-use rows, rule-ledger rows, trace-verdict boundary rows, stream-limit reflection rows, limit-boundary rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. Its classifier reads a process limit only as the visible boundary of finite use and reflection ledgers; it does not close a private meaning, halting decision, or total truth predicate. The ledger story is controlled nonclosure: every limit-facing consumer must replay the certified-use, rule, trace, and stream rows before seeing the boundary row, and no closed meta loop is generated by the packet.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UseProcessLimit}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier and classifier definitions, NameCert-obligation, ledger-nonescape, certified-use-process-interface, Lean-follow-up boundary, and existing closure annotation for $\mathsf{UseProcessLimit}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UseProcessLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim private-meaning witnesses, social-habit oracles, host equality, total truth predicates, halting decisions, or closed meta loops.

Upgrade path: To reach obligationClosure, the use-process classifier and ledger-nonescape theorem must be formalized as finite-history replay obligations.

Chapter 1317

A Concrete Naming Certificate for $\mathsf{CertifiedUseProcess}^{\uparrow}$

The $\mathsf{CertifiedUseProcess}^{\uparrow}$ packet realizes the use-as-public-process surface of subsection 9462.29.7 inside the auditable-generativity programme of chapter 9462. It sits beside the socket-use ledger of chapter 1044, the registry export surface of chapter 1216, the machine-facing boundary of chapter 1221, the apophatic naming boundary of chapter 543, and the inward eligibility proof of chapter 1309. Those siblings separately expose visible use, export control, machine routing, refused supply, and name admission; this carrier binds them before a public use may be read as rule-following.

Definition 1317.1 (Certified-use process carrier)**.** *A* $\mathsf{CertifiedUseProcess}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, R, C, L, H, K, P, N)$$

with the following displayed rows:

1. *U is the certified-use row, recording the public use event, its name, item, local history, and source/classifier surface.*
2. *R is the public rule-witness row, recording the finite rule-following evidence by which the use may be cited outside private impression.*
3. *C is the continuation-process route, recording the displayed* Cont *path through which the use is replayed for downstream consumers.*
4. *L is the refusal ledger, recording hidden-source refusals, private-meaning refusals, and blocked rule-following claims.*
5. *H stores componentwise* hsame *transport for certified use, rule witness, continuation route, refusal ledger, replay, provenance, and local naming rows.*
6. *K stores displayed* Cont *replay of the same certified-use process route.*
7. *P is the* Pkg *provenance over* $\mathsf{SocketUseLedger}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{NameEligibilityProof}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}$ *row binding the packet to the displayed use process.*

The carrier has no coordinate for private meaning, social repetition without a rule witness, a machine-facing export without registry control, a hidden source supply, or a local name admitted without a NameCert row.

Lean4 (definition) `BEDC.Derived.CertifiedUseProcessUp.CertifiedUseProcessUp`

Theorem 1317.2 (Certified-use process NameCert surface)**.** *For the carrier of Definition 1317.1, the local* $\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}$ *surface consists exactly of certified-use admission through U, public rule-witness admission through R, continuation-process routing through C and K, refusal-ledger exposure through L, componentwise transport through H, package provenance through P, and local naming through N.*

Proof. Project the finite packet $Q = (U, R, C, L, H, K, P, N)$. The rows U, R, C, L display the use, rule witness, process route, and refusals before any consumer may cite the packet.

The rows H and K preserve that same displayed route under transport and replay. They do not add a private-meaning coordinate or replace the rule witness by social repetition.

Finally P records the sibling surfaces and N names the local packet. Since the carrier has no other fields, these rows exhaust the local $\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.CertifiedUseProcessUp.CertifiedUseProcessNameCertSurface`

Theorem 1317.3 (Certified-use process nonprivate rule-following)**.** *Every accepted* $\mathsf{CertifiedUseProcess}^{\uparrow}$ *consumer reads public rule-following through the certified-use row U, rule-witness row R, continuation rows C, K, refusal ledger L, transport row H, provenance row P, and local naming row N. A route that accepts the public use while deleting either R or L is not an accepted* $\mathsf{CertifiedUseProcess}^{\uparrow}$ *carrier.*

Proof. Begin with the certified-use row U. It records a public use event, but the carrier does not let that event stand alone as meaning or rule-following.

The rule-witness row R and refusal ledger L determine whether the use is publicly licensed or blocked. The continuation rows C, K then replay only that licensed route, while H, P, N transport, source, and name the same finite packet.

Thus a consumer that omits the rule witness or refusal ledger changes the packet into a projection. The projection may mention a use, but it no longer carries the public rule-following surface of Definition 1317.1. □

Lean4 ✓ `BEDC.Derived.CertifiedUseProcessUp.CertifiedUseProcess nonprivate rule following`

Falsifiable prediction: A public use accepted without the rule-witness row or ledger row is not a CertifiedUseProcessUp packet.

Independence witness: socketuseledger, registryexportsurface, machineinterfaceboundary, apophaticname, nameeligibilityproof: none is bijective because CertifiedUseProcessUp binds use, rule witness, continuation route, refusal ledger, transport, provenance, and local naming in one packet.

The Lean carrier realization provides $\mathsf{Nontrivial}\ \mathsf{CertifiedUseProcess}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CertifiedUseProcess}^{\uparrow}$ instances in `lean4/BEDC/Derived/CertifiedUseProcessUp/TasteGate.lean`. The nontrivial witness exposes an accepted public-use packet, and field faithfulness keeps U, R, C, L, H, K, P, N distinct.

Closure status for $\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}$.

Constructive story: A $\mathsf{CertifiedUseProcess}^{\uparrow}$ packet is generated from process rows, rule rows, use-witness rows, registry and machine-interface rows, source and name rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The finite BHist carrier accepts a use process only when the rule witness, registry control, machine-facing boundary, and NameCert row are all displayed. Its classifier makes certified use public and nonprivate: repeated use becomes readable only through the recorded rule and process ledger, while hidden source supply, social repetition without witness, and machine export without registry control never become accepted coordinates.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-surface theorem, nonprivate-rule-following theorem, Lean-follow-up boundary, and existing closure annotations for $\mathsf{CertifiedUseProcess}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertifiedUseProcess}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.CertifiedUseProcessUp.CertifiedUseProcessUp`

Bridge status: none

Origin: ai

Not claimed: It does not claim private meaning, social repetition without a rule witness, machine-facing export without registry control, hidden source supply, or local naming without NameCert.

Upgrade path: To reach obligationClosure, the certified-use carrier and nonprivate rule-following theorem must be checked as finite packet obligations.

Chapter 1318

A Concrete Naming Certificate for $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$

The $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ packet realizes the cannot-claim registry of subsection 9462.33.7 as a finite NameCert carrier for philosophy-package refusals. It sits beside the claim admission surface of chapter 1209, the whole-registry packet of chapter 1220, the export-control surface of chapter 1216, and the axis refusal ledger of chapter 554. Those siblings admit, route, export, or audit claims; this carrier binds the refusal status, reason, boundary condition, blocking gate, transport, replay, provenance, and local naming row as the primary cannot-claim surface.

Definition 1318.1 (Philosophy cannot-claim registry carrier)**.** *A* $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, S, E, U, B, H, K, P, N)$$

with the following displayed rows:

1. *C is the claim-text row, recording the exact philosophy-package reading under refusal;*
2. *S is the status row, separating forbidden, deferred, and partial cannot-claim readings;*
3. *E is the exclusion-reason row, recording why the claim is not presently exportable;*
4. *U is the upgrade-condition row, recording the certificate, bridge, formal target, or principled boundary needed before the status can change;*
5. *B is the blocking-gate row, recording the registry gate that prevents export through* $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, *or* $\mathsf{RegistryExportSurface}^{\uparrow}$*;*
6. *H stores componentwise* hsame *transport for the claim, status, reason, upgrade, blocking, replay, provenance, and local naming rows;*
7. *K stores displayed* Cont *replay for claim lookup, refusal-status inspection, blocking-gate readback, and upgrade-boundary readback;*
8. *P is the* Pkg *provenance over* $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AxisZeckendorfCannotClaim}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an admitted claim, prose-only override, hidden status change, unrecorded export route, or private gate bypass.

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryTasteGate`
`single carrier alignment`

Theorem 1318.2 (Philosophy cannot-claim registry NameCert obligations)**.** *For the carrier of Definition 1318.1, the local* $\mathsf{NameCert}_{\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (C, S, E, U, B, H, K, P, N)$*, claim-text coverage by* C*, status exactness by* S*, exclusion-reason coverage by* E*, upgrade-boundary coverage by* U*, blocking-gate coverage by* B*, componentwise transport by* H*, displayed replay by* K*, provenance by* P*, and local naming by* N*.*

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryNameCertObligations`

Proof. Project the finite packet of Definition 1318.1. The rows C, S, E, U, B separately display the refused reading, its status, the reason it is excluded, the boundary under which it may be reconsidered, and the gate that blocks export.

The rows H and K keep the refusal stable under transported presentations and replayed registry reads. A transported claim text is therefore still read with its displayed status, exclusion reason, upgrade boundary, and blocking gate.

The rows P and N package and name the same data. Since the carrier has no coordinate for admitted export or private gate bypass, these rows exhaust the local NameCert obligation surface. □

Theorem 1318.3 (Philosophy cannot-claim registry non-escape)**.** *Every public consumer of an accepted* $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ *packet may inspect only the claim-text row* C*, status row* S*, exclusion-reason row* E*, upgrade-condition row* U*, blocking-gate row* B*,* hsame *transport row* H*,* Cont *replay row* K*,* Pkg *provenance row* P*, and local naming row* N*. It receives no admitted-claim row, prose-only override, hidden export permission, private registry route, or status change outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryNonescape`

Proof. Use the carrier definition. The visible rows are exactly $C, S, E, U, B, H, K, P, N$, and the NameCert obligations above attach the refusal reading to those rows.

If a consumer asks for admitted export, private override, or a hidden status change, it asks for a coordinate not present in R. The transport and replay rows can preserve and replay displayed refusal data, but they cannot create an admission row.

Thus every accepted cannot-claim read remains inside the listed packet. A route that bypasses S or B is not a $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ export of refusal content. □

Theorem 1318.4 (Philosophy cannot-claim registry status exactness)**.** *For an accepted* PhilosophyCannotClaimReg *carrier* $R = (C, S, E, U, B, H, K, P, N)$*, the status row* S *and blocking-gate row* B *are jointly necessary for exporting a cannot-claim refusal. If either row is absent, the packet may still display claim text or an exclusion reason, but it cannot export a typed refusal through* $\mathsf{ClaimRegistryLayer}^{\uparrow}$*,* $\mathsf{RegistryLayer}^{\uparrow}$*, or* $\mathsf{RegistryExportSurface}^{\uparrow}$*.*

Proof. Read the status row first. It determines whether the refusal is forbidden, deferred, or partial, and therefore separates a cannot-claim entry from an untyped note.

Read the blocking-gate row next. It is the displayed gate that prevents the refused claim from being exported through the sibling registry and export surfaces.

The transport, replay, provenance, and naming rows preserve this pair. Without S the refusal has no typed status; without B it has no export-blocking gate. Hence a cannot-claim refusal is exported only through the same finite packet containing both rows. □

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryStatusExactness`

Theorem 1318.5 (Philosophy cannot-claim registry obligation boundary)**.** *For an accepted* PhilosophyCannotClaim *carrier* $R = (C, S, E, U, B, H, K, P, N)$*, the rows* $C, S, E, U, B, H, K, P, N$ *are the full obligation boundary. A cannot-claim consumer may inspect claim text, status, exclusion reason, upgrade condition, blocking gate, componentwise transport, displayed replay, provenance, and local naming, but no admitted-claim row, private export permission, hidden status change, or prose-only override.*

Proof. Project the carrier. Its analytic rows are exactly C, S, E, U, B, and the preceding NameCert obligations attach claim text, status, exclusion reason, upgrade boundary, and blocking gate to those rows.

The structural rows H, K, P, N preserve, replay, source, and name the same finite refusal surface. None of them carries an admitted-claim coordinate or a private route around the blocking gate.

Thus every local obligation is exhausted by the displayed packet. A consumer asking for any further refusal authority asks for data outside the carrier. □

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryObligationBoundary`

Theorem 1318.6 (Philosophy cannot-claim registry gate induction)**.** *Every refusal export from an accepted* $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ *packet is obtained by induction over the displayed gate rows* C, S, E, U, B *and then by structural replay through* H, K, P, N*. The export is not licensed by prose, by a global registry convention, or by an unrecorded cannot-claim policy.*

Proof. Use induction over the displayed refusal route. The base read is the claim text row C, which fixes the entry under inspection.

The status, exclusion, upgrade, and blocking steps are the rows S, E, U, B. Each step consumes only the row already displayed in the packet and passes the same refusal entry to the next row.

Finally replay the result through H, K, P, N. These rows can transport and expose the displayed gate chain, but they cannot insert a missing policy premise. Hence refusal export is row-inductive rather than prose-derived. □

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryGateInduction`

Theorem 1318.7 (Philosophy cannot-claim registry upgrade-boundary exhaustion)**.** *For an accepted* $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ *carrier* $R = (C, S, E, U, B, H, K, P, N)$*, the upgrade boundary is exhausted by the displayed upgrade-condition row* U *together with the status and blocking rows* S *and* B*. Any status change for the refused claim must be witnessed through that finite row data before the registry, export surface, transport, replay, provenance, or local naming rows can expose it.*

Proof. Read the status row S and the blocking row B. They determine the current refusal status and the gate through which export is blocked.

The only row that can describe how this status may change is U. It records the certificate, bridge, formal target, or principled boundary needed before reconsideration.

Transport, replay, provenance, and local naming preserve the displayed upgrade condition. Since no other carrier coordinate records status-change authority, U exhausts the upgrade boundary. □

Lean4 ✓ `BEDC.Derived.PhilosophyCannotClaimRegistryUp.PhilosophyCannotClaimRegistryUpgradeBoundaryExhaus`

The Lean follow-up for PhilosophyCannotClaimRegistry$^{\uparrow}$ must provide explicit Nontrivial PhilosophyCannotClaimR and FieldFaithful PhilosophyCannotClaimRegistry$^{\uparrow}$ instances before BEDC.Derived.PhilosophyCannotClaimRegistryUp. can be cited by a checked paper marker. The nontriviality witness must expose at least one blocked philosophy claim, and field faithfulness must preserve $C, S, E, U, B, H, K, P, N$ as distinct displayed rows.

Falsifiable prediction: If the status or blocking row is absent, the carrier cannot export a cannot-claim refusal.

Independence witness: claimregistrylayer, registrylayer, registryexportsurface, registryexportconsistencygate, axiszeckendorfcannotclaim: none is bijective to PhilosophyCannotClaimRegistryUp because those siblings admit, route, or audit claims, while this packet binds refusal status, reason, upgrade boundary, blocking gate, transport, replay, provenance, and local NameCert as the primary carrier.

Closure status for NameCert$_{\text{PhilosophyCannotClaimRegistry}^{\uparrow}}$.

Constructive story: A PhilosophyCannotClaimRegistry$^{\uparrow}$ packet is generated from claim text, refusal status, exclusion reason, upgrade boundary, blocking gate, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PhilosophyCannotClaimRegistry}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1318.1, Definition 1318.2, Definition 1318.3, Definition 1318.4, Definition 1318.6, and Definition 1318.7 bind the public claim-registry export surface for claim text, status, reason, upgrade-boundary, blocking, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{PhilosophyCannotClaimRegistry}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, registry completeness, admitted-claim export, or correctness of every philosophy-package cannot-claim entry.

Upgrade path: Further closure requires Lean verification and bridge evidence for the same public claim-text, status, reason, upgrade-boundary, blocking, transport, replay, provenance, and local naming rows.

Chapter 1319

A Concrete Naming Certificate for $\mathsf{CannotClaimExportGate}^{\uparrow}$

The $\mathsf{CannotClaimExportGate}^{\uparrow}$ packet realizes the philosophy-package cannot-claim registry of Definition 9462.2179 and the export interface of Definition 9462.2192 as one finite NameCert gate. It sits beside the registry export surface of chapter 1216, the refusal registry of chapter 1248, the machine-interface boundary of chapter 1221, the package carrier of chapter 1292, and the audit export firewall of chapter 1296. Those siblings expose exported registry rows, refusal rows, target routing, package ledgers, and audit-firewall decisions separately; this carrier binds the same refused claim before it can be read as a checked formal-package target.

The export gate is the checked-target analogue of $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ in chapter 677. Both packets block a consumer from reading an unlicensed output: $\mathsf{ChoiceRecipeLedger}^{\uparrow}$ refuses a hidden selection output from finite witness recipes, while $\mathsf{CannotClaimExportGate}^{\uparrow}$ refuses checked-target export by displaying the registry row R, refusal row F, export rows E, Q, target row T, and audit row A before any local naming row can be read.

Definition 1319.1 (Cannot-claim export gate carrier)**.** *A* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, F, E, Q, T, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the registry row, recording the philosophy-package claim text, registry status, theorem-map site when present, gap-map site when present, and export request.*
2. *F is the refusal row, recording the active cannot-claim entry, refusal reason, blocked reading, and required target condition for the same claim.*
3. *E is the export-decision row, recording whether the request is refused, reading-export only, paper-synthesis export, component-checked export, formal-package target, or formal-package checked.*
4. *Q is the export-grade row, recording the grade named in Definition 9462.2193 together with the reason the refused row cannot occupy the checked grade.*
5. *T is the target row, recording the machine-facing target route, Lean target name when present, paper-only route when present, and the absence of a checked package target for refused rows.*
6. *A is the audit row, recording the readback from the registry export surface, refusal registry, machine-interface boundary, package carrier, and audit export firewall for the same claim site.*

7. *H records componentwise* hsame *transport for registry, refusal, export decision, export grade, target route, audit readback, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for reading R, F, E, Q, T, A without adding a host-side checked-target override.*
9. *P records* Pkg *provenance over the five sibling packets, the philosophy-package export interface,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CannotClaimExportGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for exporting a refused cannot-claim registry row as a checked formal-package target outside the displayed registry, refusal, export, target, audit, transport, continuation, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGateCarrierSurface`

Theorem 1319.2 (Cannot-claim export gate NameCert obligations)**.** *For the carrier of Definition 1319.1, the local* $\mathsf{NameCert}_{\mathsf{CannotClaimExportGate}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (R, F, E, Q, T, A, H, C, P, N)$, registry exposure through R, refusal exposure through F, export-decision exposure through E, export-grade exposure through Q, target-route exposure through T, audit readback through A, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite packet $G = (R, F, E, Q, T, A, H, C, P, N)$. The registry row R gives the claim site and requested export, while the refusal row F displays the active cannot-claim entry for that same site.

The export rows are then read against the refusal. The decision row E and grade row Q record the permitted export level, and the target row T records the machine-facing route without supplying a checked package target when F is active.

Finally the audit, transport, replay, provenance, and naming rows preserve that same display. Since the carrier has no additional export coordinate, these rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate namecert obligations`

Theorem 1319.3 (Cannot-claim export gate refusal nonescape)**.** *If an accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *packet carries an active refusal row F for registry row R, then its export-decision row E and export-grade row Q cannot expose that same row as formal package checked, and its target row T cannot expose it as a checked formal-package target.*

Proof. Assume an accepted packet $G = (R, F, E, Q, T, A, H, C, P, N)$ with F active for R. The active refusal row is read before the export decision, grade, or target route can be projected.

The decision row E and grade row Q are the only coordinates from which an export status may be read. Because they are displayed together with F, they record a refusal or a weaker export grade rather than the checked formal-package grade for the refused claim.

The target row T is likewise read after F, E, Q. It may record a paper-only route or a missing checked target, but it has no separate coordinate that can turn the refused row into a checked package target. Transport, continuation, provenance, and local naming preserve that refusal instead of escaping it. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate refusal nonescape`

Theorem 1319.4 (Cannot-claim export gate registry factorization)**.** *Every accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *export reading factors through the registry row R, refusal row F, export rows E, Q, target row T, and audit row A before local naming. No registry claim reaches the local* $\mathsf{NameCert}_{\mathsf{CannotClaimExportGate}^{\uparrow}}$ *row by prose adjacency to the philosophy package alone.*

Proof. Read the packet by coordinates. The registry row R supplies the claim site, and F supplies the cannot-claim status for that site. The export rows E, Q then determine the grade, while T supplies the target route.

The audit row A checks the same site against the registry export surface, refusal registry, machine-interface boundary, package carrier, and audit export firewall. Thus the local naming row N receives a factored packet rather than an uninspected package paragraph.

Since H, C, P only transport, replay, and source the displayed rows, they do not create a second route from prose to checked target. The export reading therefore factors through the listed rows. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate registry factorization`

Theorem 1319.5 (Cannot-claim export gate export-grade scope)**.** *For every accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *carrier $G = (R, F, E, Q, T, A, H, C, P, N)$, any export-grade read through Q must be scoped by the active refusal row F, decision row E, target row T, audit row A, and the field-faithful TasteGate handoff before local naming. Thus a read of Q can classify a refused claim only after the gate has displayed the refusal, export decision, target route, audit readback, and distinct-field carrier boundary for the same registry row R.*

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate export grade scope`

Proof. Project the accepted carrier. The registry row R fixes the claim site, and the refusal row F records the active cannot-claim entry for that site before the export-grade row is read.

Use the registry factorization theorem, Definition 1319.4. It forces every export reading through R, F, E, Q, T, A before local naming, so Q is not an independent grade oracle.

Finally apply the TasteGate handoff recorded by the checked alignment target `BEDC.Derived.CannotClaimExpo` Field faithfulness keeps F, E, Q, T, A as distinct rows, hence the grade read through Q remains scoped by refusal, decision, target, audit, and local NameCert data rather than collapsing into an unchecked package target. □

Theorem 1319.6 (Cannot-claim export gate TasteGate handoff)**.** *The Lean-facing handoff for* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *must keep registry row R, refusal row F, export-decision row E, export-grade row Q, target row T, audit row A, transport row H, continuation row C, provenance row P, and local NameCert row N as distinct fields. Collapsing any of F, E, Q, T, A removes the field on which the refused-claim export block depends.*

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGateTasteGate single carrier alignment`

Proof. The gate blocks a refused checked-target export by comparing the refusal row with the export decision, export grade, target route, and audit readback. These comparisons require the rows F, E, Q, T, A to remain separately projected.

Rows H, C, P, N preserve and name the same packet after the comparison. If a Lean-facing carrier merged the refusal with the decision, grade, route, or audit rows, a consumer could no longer state which field records refusal and which field records export. The TasteGate handoff therefore keeps all ten rows as field-faithful data. □

Theorem 1319.7 (Cannot-claim export gate ledger exhaustion)**.** *Every accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *export-gate read must expose exactly the rows*

$$R, F, E, Q, T, A, H, C, P, N$$

before local naming. No refused registry row can bypass the refusal, export, and audit ledger to become a checked target.

Proof. Apply the registry factorization theorem. It forces every export reading through R, F, E, Q, T, A before the local NameCert row, so a refused claim cannot reach naming from the philosophy-package paragraph alone.

Apply the export-grade scope theorem. The grade row Q is read only with the active refusal row F, decision row E, target row T, audit row A, and the field-faithful TasteGate handoff, so the checked grade is not available as an independent route.

Finally use the TasteGate handoff. It keeps $R, F, E, Q, T, A, H, C, P, N$ as distinct fields, while H, C, P only transport, replay, and source the same packet. Since the carrier has no further export coordinate, the ledger is exhausted by the displayed rows and no refused registry row becomes a checked target by bypassing them. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate ledger exhaustion`

Theorem 1319.8 (Cannot-claim export gate obligation-closure package)**.** *An accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *packet exposing the carrier definition, NameCert obligations, refusal nonescape, registry factorization, export-grade scope, TasteGate handoff, and ledger exhaustion supplies the full obligationClosure package for the gate. The package consists of registry row R, refusal row F, export rows E, Q, target row T, audit row A, componentwise* hsame *transport H, displayed* Cont *replay C, package provenance P, and local* $\mathsf{NameCert}_{\mathsf{CannotClaimExportGate}^{\uparrow}}$ *naming row N.*

Proof. Start with the carrier of Definition 1319.1. It displays exactly the rows $R, F, E, Q, T, A, H, C, P, N$, so the package has named coordinates for registry exposure, refusal exposure, export decision, export grade, target route, audit readback, transport, replay, provenance, and local naming.

Apply the NameCert obligations theorem Definition 1319.2. It identifies those coordinates as the local obligation surface. Apply refusal nonescape, registry factorization, export-grade scope, and the TasteGate handoff from Definition 1319.3, Definition 1319.4, Definition 1319.5, and Definition 1319.6. These theorems force the refused claim to pass through R, F, E, Q, T, A while H, C, P, N preserve the same field-faithful packet.

Finally use ledger exhaustion from Definition 1319.7. Since no additional export coordinate remains, the displayed rows are sufficient for the obligationClosure package and no verification-axis claim is needed to state the paper-side closure package. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGateObligationClosurePackage`

Theorem 1319.9 (Cannot-claim export gate kernel scope lock)**.** *For every accepted* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *carrier $G = (R, F, E, Q, T, A, H, C, P, N)$, the consumer-facing scope of the gate is locked to the displayed* BHist, hsame, Cont, Pkg, NameCert, *registry, refusal, export, target, and audit rows. The TasteGate marker*

`BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGateTasteGate_single_carrier_alignment`

is the Lean-facing field-separation witness: it keeps $R, F, E, Q, T, A, H, C, P, N$ as distinct carrier rows before any checked-target export read can be compared. Thus a refused registry row has no kernel route to a checked formal-package target outside the displayed rows.

Proof. Project the accepted carrier. Its only rows are $R, F, E, Q, T, A, H, C, P, N$, with R displaying the registry site, F the refusal, E, Q the export decision and grade, T the target route, and A the audit readback.

Apply registry factorization, export-grade scope, TasteGate handoff, and ledger exhaustion. These results force every consumer-facing export read through the same finite row sequence before local naming, while the checked TasteGate marker records the Lean-facing fact that the fields remain separated.

The structural rows BHist, hsame, Cont, Pkg, and NameCert transport, replay, source, and name that packet. Since no extra checked-target coordinate is present, the kernel scope is exactly the displayed gate surface. □

Lean4 ✓ `BEDC.Derived.CannotClaimExportGateUp.CannotClaimExportGate kernel scope lock`

Falsifiable prediction: A refused cannot-claim registry row cannot be exported as a checked formal-package target through an accepted CannotClaimExportGateUp packet.

Independence witness: registryexportsurface, refusalregistry, machineinterfaceboundary, bedcphilosophypackage, auditexportfirewall: none is bijective to CannotClaimExportGateUp because the gate simultaneously carries refusal status, export grade, target route, audit readback, transport, continuation, provenance, and local NameCert rows.

The Lean follow-up must provide Nontrivial CannotClaimExportGate$^{\uparrow}$ and FieldFaithful CannotClaimExportGate$^{\uparrow}$ instances in `lean4/BEDC/Derived/CannotClaimExportGateUp/TasteGate.lean`. The field-faithfulness witness must preserve $R, F, E, Q, T, A, H, C, P, N$ as distinct rows before a checked target can cite `BEDC.Derived.CannotClaimExportGateUp.TasteGate.CannotClaimExportGateTasteGate_single_carrier_a`

Closure status for NameCert$_{\text{CannotClaimExportGate}^{\uparrow}}$.

Constructive story: A CannotClaimExportGate$^{\uparrow}$ packet is generated from registry rows, refusal rows, export-decision rows, target rows, audit rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier makes a blocked claim exportable only as a blocked claim: any consumer must read the registry entry and the refusal surface before reaching the export gate or formal target row. The ledger story is refusal preservation, where the audit and continuation rows carry the same cannot-claim evidence forward, preventing a refused registry item from being read as a checked package target or silently converted into success.

Theory closure: TheoryGate(NameCert$_{\text{CannotClaimExportGate}^{\uparrow}}$, maturePackageClosed)

Scope closed: This chapter binds the carrier, NameCert obligations, refusal nonescape, registry factorization, export-grade scope, TasteGate handoff, ledger exhaustion, obligation-closure package, and kernel scope lock for CannotClaimExportGate$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{CannotClaimExportGate}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim export of refused registry rows as checked formal-package targets outside the displayed registry, refusal, export, target, audit, transport, and naming rows.

Upgrade path: Verification work may encode the same export-gate package without changing the paper-side mature closure.

Chapter 1320

A Concrete Naming Certificate for $\mathsf{PhilosophyGovernanceExport}^{\uparrow}$

The $\mathsf{PhilosophyGovernanceExport}^{\uparrow}$ packet realizes the governance and export discipline of section 9462.33 as one finite NameCert carrier. It sits beside the package carrier of chapter 1292, the cannot-claim export gate of chapter 1319, the claim registry layer of chapter 1209, the philosophy map target of chapter 1277, and the gap/failure bridge audit of chapter 1229. Those siblings expose package rows, refusal-gate rows, claim rows, map-target rows, and bridge-audit rows separately; this carrier binds them before any package reading may leave as an export-grade claim.

Definition 1320.1 (Philosophy governance-export carrier)**.** *A* $\mathsf{PhilosophyGovernanceExport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (P, R, L, S, C, X, T, O, H, K, N)$$

with the following displayed rows:

1. *P is the package row, recording the* $\mathsf{BedcPhilosophyPackage}^{\uparrow}$ *thesis, registry, theorem-map, gap-map, tradition/science-map, closure-status, and audit-policy coordinates consumed by the export.*
2. *R is the finite cannot-claim registry row for the same package reading.*
3. *L is the active-label row, recording the current citation target, active formulation, and label discipline for the exported reading.*
4. *S is the capstone row, recording whether the reading spans the required registered domains and which dependencies are displayed.*
5. *C is the claim-registry row, recording claim text, admission status, local refusal status, and public route.*
6. *X is the export-grade row, recording reading export, paper-synthesis export, component-checked export, formal-package target, or formal-package checked status.*
7. *T is the transport row from* $\mathsf{CannotClaimExportGate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{PhilosophyMapTarget}^{\uparrow}$, *and* $\mathsf{GapFailureBridgeAudit}^{\uparrow}$ *into the same package reading.*
8. *O is the route and provenance row, recording displayed* Cont *replay,* Pkg *source data, and package-file provenance for $P, R, L, S, C, X, T, H, N$.*

9. *H records componentwise* hsame *transport for package, registry, active-label, capstone, claim, export-grade, transport, route, provenance, and naming rows.*
10. *K is the policy row enforcing finite registry lookup, refusal lookup, active-label lookup, capstone dependency lookup, export-grade lookup, and no hidden checked-target override.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhilosophyGovernanceExport}^{\uparrow}}$ *row.*

The carrier has no coordinate for an export whose claim row bypasses the cannot-claim registry, whose active label detaches from the exported formulation, whose capstone dependency list is implicit, whose export grade outruns its certificate, or whose checked formal-package status is supplied by host-side prose.

Lean4 ✓ `BEDC.Derived.PhilosophyGovernanceExportUp.PhilosophyGovernanceExportTasteGate single carrier alignment`

Theorem 1320.2 (Philosophy governance-export no override)**.** *Every accepted* $\mathsf{PhilosophyGovernanceExport}^{\uparrow}$ *export route factors through the package row P, cannot-claim registry row R, active-label row L, capstone row S, claim-registry row C, export-grade row X, transport row T, route/provenance row O, componentwise transport row H, policy row K, and local naming row N. No consumer can override an active refusal, active label, capstone dependency list, export grade, or package provenance by appealing to prose adjacency in the philosophy package.*

Proof. Project the finite packet $E = (P, R, L, S, C, X, T, O, H, K, N)$. The package row P names the package reading, while R, L, S, C, X display the refusal, active-label, capstone, claim-registry, and export-grade surfaces before any public route is named.

The transport and route rows then preserve the same order. The row T cites the sibling refusal gate, claim registry layer, philosophy map target, and gap/failure bridge audit, while O replays the displayed route and provenance for the same reading.

Finally H, K, N compare, enforce, and name the packet. Since the carrier has no further coordinate, a consumer has nowhere to obtain a checked-target override, a different active label, or a stronger export grade. The no-override claim is therefore a projection of the displayed rows. □

Lean4 ✓ `BEDC.Derived.PhilosophyGovernanceExportUp.PhilosophyGovernanceExportNoOverride`

Theorem 1320.3 (Philosophy governance-export NameCert obligations)**.** *For the carrier of Definition 1320.1, the local* $\mathsf{NameCert}_{\mathsf{PhilosophyGovernanceExport}^{\uparrow}}$ *surface consists exactly of carrier admission for $E = (P, R, L, S, C, X, T, O, H, K, N)$, package exposure through P, cannot-claim exposure through R, active-label exposure through L, capstone exposure through S, claim-registry exposure through C, export-grade exposure through X, sibling transport through T, displayed route and provenance through O, componentwise* hsame *transport through H, finite policy enforcement through K, and local naming through N.*

Proof. Read the packet by coordinates. Carrier admission supplies the finite BHist object, and the rows P, R, L, S, C, X supply the package reading, refusal registry, active label, capstone dependencies, claim entry, and export grade.

The support rows then expose the audit route. The row T records which sibling surfaces are consumed, O records continuation and provenance, H transports only the displayed rows, and K checks the finite policy conditions before the local name N is available.

There is no remaining field in the packet. Hence the listed coordinates exhaust the local NameCert obligations, and any accepted export must be read through those obligations. □

Falsifiable prediction: If either the cannot-claim registry row R or the export-grade row X is absent, Definition 1320.2 fails because a consumer can no longer distinguish refused package readings from export-grade readings inside the carrier.

Independence witness: bedcphilosophypackage, cannotclaimexportgate, claimregistrylayer, philosophymaptarget, gapfailurebridgeaudit: none is bijective to PhilosophyGovernanceExportUp because this packet simultaneously carries finite registry, refusal, active-label, capstone, export-grade, transport, route, provenance, policy, and local NameCert rows.

The Lean follow-up must provide Nontrivial PhilosophyGovernanceExport$^{\uparrow}$ and FieldFaithful PhilosophyGovernance instances in `lean4/BEDC/Derived/PhilosophyGovernanceExportUp/TasteGate.lean`. The field-faithfulness witness must preserve $P, R, L, S, C, X, T, O, H, K, N$ as distinct rows before a checked target can cite `BEDC.Derived.PhilosophyGovernanceExportUp.TasteGate.PhilosophyGovernanceExportTast`

Closure status for $\mathsf{NameCert}_{\mathsf{PhilosophyGovernanceExport}^{\uparrow}}$.

Constructive story: A PhilosophyGovernanceExport$^{\uparrow}$ packet is generated from package, cannot-claim, active-label, capstone, claim-registry, export-grade, sibling-transport, route/provenance, componentwise hsame transport, policy, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhilosophyGovernanceExport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1320.1, Definition 1320.2, Definition 1320.3, the falsifiable prediction, and the independence witness bind governance and export discipline to a finite no-override NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhilosophyGovernanceExport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, complete package correctness, checked formal-package status, or any primitive philosophical object, host truth, quotient, choice, or hidden subject reading.

Upgrade path: Lean closure requires Nontrivial PhilosophyGovernanceExportUp and FieldFaithful PhilosophyGovernanceExportUp instances in `lean4/BEDC/Derived/PhilosophyGovernanceExportUp/TasteGate.lean`.

Chapter 1321

A Concrete Naming Certificate for $\mathsf{PackageGovernanceExport}^{\uparrow}$

The $\mathsf{PackageGovernanceExport}^{\uparrow}$ packet realizes the philosophy-package cannot-claim, in-place governance, capstone, and export interface of subsection 9462.33.7, subsection 9462.33.9, and subsection 9462.33.10 as one finite NameCert carrier. It sits beside the package carrier of chapter 1292, the claim registry layer of chapter 1209, the registry export surface of chapter 1216, the audit export firewall of chapter 1296, and the package map boundary of chapter 1307. Those siblings expose package rows, claim admission, export rows, audit refusal, and map-boundary rows separately; this packet binds them before a philosophy-package governance decision can be read as an exportable NameCert row.

Definition 1321.1 (Package governance export carrier)**.** *A* $\mathsf{PackageGovernanceExport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, R, A, C, E, F, T, M, H, K, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{BedcPhilosophyPackage}^{\uparrow}$ *row, recording the package thesis, concept registry, theorem map, gap map, tradition/science map, cannot-claim registry, closure-status row, and audit-policy row consumed by the governance packet.*

2. *R is the cannot-claim registry row, recording blocked philosophical readings, refusal status, reason, upgrade boundary, and the gate through which* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *and* $\mathsf{RegistryExportSurface}^{\uparrow}$ *must read the refusal.*

3. *A is the in-place governance row, recording the affected registry entry, active label, replacement boundary, status effect, and audit-continuity data for package governance.*

4. *C is the capstone dependency row, recording the concept, theorem-map, gap-map, tradition/science-map, and closure-status dependencies that license any capstone export.*

5. *E is the export-grade row, recording reading export, paper synthesis export, component checked export, formal package target, and formal package checked status as displayed grades rather than prose impressions.*

6. *F is the* $\mathsf{AuditExportFirewall}^{\uparrow}$ *row, recording the same claim site, positive certificate when present, active failure or refusal when present, consistency gate, and firewall decision.*

7. *T is the target row, recording the machine-facing route, paper-only route when present, Lean target name when present, and absence of a checked package target when a refusal or missing dependency is active.*
8. *M is the* $\mathsf{PackageMapBoundary}^{\uparrow}$ *row, recording theorem-map, gap-map, tradition-map, scientific-map, and axis-boundary handoff for the same package reading.*
9. *H records componentwise* hsame *transport for package, registry, governance, capstone, export, firewall, target, map-boundary, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay for reading P, R, A, C, E, F, T, M without adding a host-side governance override or checked-target shortcut.*
11. *Q is the* Pkg *provenance over* $\mathsf{PackageGovernanceExport}^{\uparrow}$, $\mathsf{BedcPhilosophyPackage}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AuditExportFirewall}^{\uparrow}$, $\mathsf{PackageMapBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PackageGovernanceExport}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden governance decision, unregistered cannot-claim override, capstone without dependencies, export grade detached from refusal audit, unchecked formal-package target, host equality, quotient import, choice import, or prose-only package promotion.

Lean4 ✓ `BEDC.Derived.PackageGovernanceExportUp.PackageGovernanceExportTasteGate single carrier alignment`

Theorem 1321.2 (Package governance export NameCert obligations)**.** *For the carrier of Definition 1321.1, the local* $\mathsf{NameCert}_{\mathsf{PackageGovernanceExport}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (P, R, A, C, E, F, T, M, H, K, Q, N)$, package exposure through P, cannot-claim registry exposure through R, in-place governance exposure through A, capstone dependency exposure through C, export-grade exposure through E, firewall readback through F, target-route exposure through T, package-map boundary exposure through M, componentwise* hsame *transport through H, displayed* Cont *replay through K, package provenance through Q, and local naming through N.*

Proof. Project the finite packet $G = (P, R, A, C, E, F, T, M, H, K, Q, N)$. The package row P supplies the whole philosophy-package surface, while R, A, C, E display the refusal registry, governance action, capstone dependency set, and export grade before any target route is named.

Next read the sibling audit rows. The firewall row F checks the same claim site against positive and negative certificate data, the target row T records whether a machine-facing or paper-only route is available, and the map-boundary row M keeps theorem, gap, tradition, science, and axis-boundary maps visible.

Finally H, K, Q, N transport, replay, source, and name exactly those displayed rows. Since the carrier contains no hidden governance, export, or target coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1321.3 (Package governance export refusal totality)**.** *Every accepted* $\mathsf{PackageGovernanceExport}^{\uparrow}$ *export decision reads the cannot-claim row R and the audit-firewall row F before projecting the export-grade row E, target row T, or local naming row N. If R contains an active refusal for the package reading, the packet cannot expose that reading as formal package checked or as a checked formal-package target.*

Proof. Assume an accepted packet $G = (P, R, A, C, E, F, T, M, H, K, Q, N)$. The refusal row R is displayed before export-grade or target projection, so the reading is first classified as admitted, deferred, partial, or blocked by the cannot-claim registry.

The firewall row F reads the same site against the audit rows. If R carries an active refusal, then F records refusal or certificate conflict for that site rather than an admitted checked-target export.

Rows E and T are therefore downstream of the refusal and firewall reads. They can expose a weaker export grade, a paper-only route, or an absent checked package target, but they have no coordinate that can erase R or F. The local naming row N names the factored packet and does not create an independent export path. □

Theorem 1321.4 (Package governance export sibling independence)**.** *The* $\mathsf{PackageGovernanceExport}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{BedcPhilosophyPackage}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{AuditExportFirewall}^{\uparrow}$, *or* $\mathsf{PackageMapBoundary}^{\uparrow}$. *Each sibling is consumed through a displayed row, but none carries the joint package, refusal, governance, capstone, export-grade, firewall, target, map-boundary, transport, replay, provenance, and local naming packet.*

Proof. Compare the displayed rows. $\mathsf{BedcPhilosophyPackage}^{\uparrow}$ supplies the package ledger, $\mathsf{ClaimRegistryLayer}^{\uparrow}$ supplies claim admission and refusal boundary, $\mathsf{RegistryExportSurface}^{\uparrow}$ supplies export rows, $\mathsf{AuditExportFirewall}^{\uparrow}$ supplies same-site audit refusal, and $\mathsf{PackageMapBoundary}^{\uparrow}$ supplies the theorem, gap, tradition, science, and axis-boundary handoff.

The governance export carrier contains those sibling roles together with the in-place governance row A, capstone dependency row C, export-grade row E, target row T, and local NameCert row N. A NameCert-row-preserving bijection would have to preserve this joint governance-and-export surface.

No listed sibling has all of these coordinates as local rows. Hence each sibling is a dependency or consumer surface for $\mathsf{PackageGovernanceExport}^{\uparrow}$, not a bijective presentation of it. □

Theorem 1321.5 (Package governance export capstone dependency exactness)**.** *An accepted* $\mathsf{PackageGovernanceExport}^{\uparrow}$ *packet may expose a capstone export only through the capstone dependency row C together with the map-boundary row M and audit-firewall row F. If C omits a displayed dependency family, the packet may still record a package reading, but it cannot certify that reading as a capstone export.*

Proof. The capstone row C is the only coordinate that records cross-domain dependencies for a unifying package claim. It must therefore be read before a capstone export grade is projected.

The map-boundary row M then checks that the dependency families are visible as theorem-map, gap-map, tradition-map, scientific-map, and axis-boundary handoffs. The firewall row F checks whether any active refusal or certificate conflict addresses the same site.

If C omits one dependency family, the packet still has package, registry, governance, and export rows. But the capstone-specific dependency certificate is absent, and neither M nor F can manufacture it. The export can remain a package reading or weaker export grade, not a certified capstone export. □

Falsifiable prediction: Deleting the cannot-claim row or firewall row permits a governance packet to name an export grade without first refusing blocked package readings.

Independence witness: bedcphilosophypackage, claimregistrylayer, registryexportsurface, auditexportfirewall, packagemapboundary: none is bijective because PackageGovernanceExportUp carries the joint governance, capstone-dependency, export-grade, target, firewall, map-boundary, transport, replay, provenance, and local NameCert rows.

The Lean `TasteGate.lean` boundary for $\mathsf{PackageGovernanceExport}^{\uparrow}$ must provide $\mathsf{Nontrivial}\ \mathsf{PackageGovernance}$ and $\mathsf{FieldFaithful}\ \mathsf{PackageGovernanceExport}^{\uparrow}$ instances in `lean4/BEDC/Derived/PackageGovernanceExportUp/Tas`

The nontriviality witness must expose at least one governed package export decision, and field faithfulness must preserve $P, R, A, C, E, F, T, M, H, K, Q, N$ as distinct rows before any checked marker can cite `BEDC.Derived.PackageGovernanceExportUp.TasteGate.PackageGovernanceExportTasteGate_single`

Closure status for $\mathsf{NameCert}_{\mathsf{PackageGovernanceExport}\uparrow}$.

Constructive story: A $\mathsf{PackageGovernanceExport}^{\uparrow}$ packet is generated from package, cannot-claim registry, in-place governance, capstone dependency, export grade, audit firewall, target route, map-boundary, componentwise transport, displayed replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PackageGovernanceExport}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1321.1, Definition 1321.2, Definition 1321.3, Definition 1321.4, and Definition 1321.5 bind package governance and export to one finite NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PackageGovernanceExport}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, registry completeness, correctness of every governance action, correctness of every capstone dependency, or whole-package checked export.

Upgrade path: Lean closure requires Nontrivial PackageGovernanceExportUp and FieldFaithful PackageGovernanceExportUp instances in `lean4/BEDC/Derived/PackageGovernanceExportUp/TasteGate.lean`.

Chapter 1322

A Concrete Naming Certificate for PhaseGovernedCertificateRoute$^{\uparrow}$

The PhaseGovernedCertificateRoute$^{\uparrow}$ packet realizes the research-programme route of section 9462.32 as a finite certificate router. It sits beside ProgrammeStrengthLedger$^{\uparrow}$ of chapter 1305, PhilosophyCannotClaimRegistr of chapter 1318, CannotClaimExportGate$^{\uparrow}$ of chapter 1319, and UseProcessBoundary$^{\uparrow}$ of chapter 1313: those siblings record strength, refusal, export blocking, and certified-use process boundaries, while this packet binds their rows before a philosophy programme may present a claim as routed through a named certificate phase. The certified-use consumer of chapter 1317 supplies the public use process read by the same route.

Definition 1322.1 (Phase-governed certificate route carrier)**.** *A* PhaseGovernedCertificateRoute$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, S, F, E, D, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the phase row, recording the current certificate phase and the admissible next phase for the programme route;*
2. *S is the strength row, read from the* ProgrammeStrengthLedger$^{\uparrow}$ *surface;*
3. *F is the refusal row, read from the* PhilosophyCannotClaimRegistry$^{\uparrow}$ *surface;*
4. *E is the export-gate row, read from the* CannotClaimExportGate$^{\uparrow}$ *surface before any checked-target reading is exposed;*
5. *D is the dependency and failure-audit row, recording open blockers, failed obligations, and audit-facing non-export decisions;*
6. *U is the certified-use and process-boundary row, read through* UseProcessBoundary$^{\uparrow}$ *and* CertifiedUseProcess$^{\uparrow}$*;*
7. *H records componentwise* hsame *transport for phase, strength, refusal, export, dependency, use-process, continuation, provenance, and local naming rows;*
8. *C records displayed* Cont *replay through the same rows;*
9. *P records* Pkg *provenance over* PhaseGovernedCertificateRoute$^{\uparrow}$*,* ProgrammeStrengthLedger$^{\uparrow}$*,* PhilosophyCannotCl CannotClaimExportGate$^{\uparrow}$*,* UseProcessBoundary$^{\uparrow}$*,* CertifiedUseProcess$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for skipping the phase row, erasing refusal, inflating strength, exporting a blocked target, hiding a failure-audit row, or reading a certified use outside the displayed process boundary.

Lean4 ✓ `BEDC.Derived.PhaseGovernedCertificateRouteUp.PhaseGovernedCertificateRouteTasteGate`
`single carrier alignment`

Theorem 1322.2 (Phase-governed certificate route NameCert obligations)**.** *For the carrier of Definition 1322.1, the local* $\mathsf{NameCert}_{\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (A, S, F, E, D, U, H, C, P, N)$, phase exposure through A, strength exposure through S, refusal exposure through F, export-gate exposure through E, dependency and failure-audit exposure through D, certified-use and process-boundary exposure through U, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite packet. The phase row A is read before the strength, refusal, export, dependency, and use-process rows can be treated as one route.

Read the sibling surfaces in the displayed order. The strength ledger supplies S, the cannot-claim registry supplies F, the export gate supplies E, and the use-process boundary supplies U; the failure-audit row D records blockers that prevent a stronger certificate reading.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since no further route coordinate is present, these rows exhaust the local NameCert obligations. □

Theorem 1322.3 (Phase-governed certificate route nonescape)**.** *Every accepted* $\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}$ *consumer reads a programme claim through the phase row A, strength row S, refusal row F, export-gate row E, dependency and failure-audit row D, certified-use and process-boundary row U, transport row H, replay row C, provenance row P, and local naming row N. It cannot export a stronger phase, erase a refusal, bypass a blocked export gate, hide a failure-audit row, replace certified use by prose, or obtain a checked target outside those rows.*

Proof. Begin with the NameCert obligations. They force any accepted route to expose $A, S, F, E, D, U, H, C, P, N$ before the local name is available.

The sibling packets keep the route non-escaping. $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ blocks strength inflation, $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$ preserves the refusal row, $\mathsf{CannotClaimExportGate}^{\uparrow}$ blocks refused checked-target export, and $\mathsf{UseProcessBoundary}^{\uparrow}$ keeps certified use tied to process mediation.

The remaining transport, replay, provenance, and naming rows preserve that same display. Hence the accepted route has no coordinate from which a stronger phase, erased refusal, bypassed export, hidden blocker, prose-only use, or unchecked target can be exported. □

Falsifiable prediction: If a PhaseGovernedCertificateRouteUp packet remains accepted after deleting either its phase row or its refusal row, then the route carrier is refuted.

Independence witness: programmestrengthledger, philosophycannotclaimregistry, cannotclaimexportgate, useprocessboundary, certifieduseprocess: none is bijective to PhaseGovernedCertificateRouteUp because the route binds phase, strength, refusal, export gate, failure audit, dependency, certified use, transport, replay, provenance, and local naming in one finite packet.

The Lean follow-up requires $\mathsf{Nontrivial}\ \mathsf{PhaseGovernedCertificateRoute}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PhaseGovernedCertificate}$ instances in `lean4/BEDC/Derived/PhaseGovernedCertificateRouteUp/TasteGate.lean`. The

field-faithfulness witness must keep $A, S, F, E, D, U, H, C, P, N$ as distinct rows before a checked boundary can cite `BEDC.Derived.PhaseGovernedCertificateRouteUp.TasteGate`.

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}$ packet is generated from a phase row, certificate row, refusal row, export-gate row, failure-roadmap row, certified-use row, process-boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier treats the certificate route as phase governed: each export or target read must pass through the displayed phase and refusal coordinates before it can inspect the certificate or use-process rows. The ledger story is route discipline, where transport and replay preserve blocked targets, failure audits, and certified-use boundaries instead of allowing a stronger phase or erased refusal to be inferred from prose.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation theorem, nonescape theorem, Lean-follow-up boundary, and existing closure annotation for $\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.PhaseGovernedCertificateRouteUp.PhaseGovernedCertificateRouteTasteGate\_single\_carrier`

Bridge status: none

Origin: ai

Not claimed: It does not claim phase skipping, refusal erasure, strength inflation, blocked-target export, hidden failure-audit rows, or certified use outside the process boundary.

Upgrade path: To reach obligationClosure, the phase-governed carrier and nonescape theorem must be represented as checked route and refusal-preservation obligations.

Chapter 1323

A Concrete Naming Certificate for PackageExportDiscipline$^{\uparrow}$

The PackageExportDiscipline$^{\uparrow}$ packet realizes the export discipline of section 9462.33 as a finite NameCert carrier for philosophy-package exports. It sits beside the package-map boundary of chapter 1307, the cannot-claim registry of chapter 1318, the registry export surface of chapter 1216, the cannot-claim export gate of chapter 1319, and the formal-target ledger of chapter 1282. Those siblings expose map boundaries, refusal status, registry export rows, export-grade gates, and formal-target ledgers separately; this carrier binds them into one package export packet before a philosophy reading can leave the package.

Definition 1323.1 (Package export discipline carrier)**.** *A* PackageExportDiscipline$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, R, S, G, T, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the package-map row, recording theorem-map, gap-map, tradition-map, scientific-map, and axis-boundary data from* PackageMapBoundary$^{\uparrow}$*;*
2. *R is the refusal-registry row, recording cannot-claim status, exclusion reason, upgrade boundary, and blocking gate from* PhilosophyCannotClaimRegistry$^{\uparrow}$*;*
3. *S is the registry-export row, recording registration, blocking, status, machine target, audit route, and non-smuggling data from* RegistryExportSurface$^{\uparrow}$*;*
4. *G is the export-gate row, recording refusal, export decision, export grade, target route, and audit readback from* CannotClaimExportGate$^{\uparrow}$*;*
5. *T is the formal-target row, recording claim entry, target skeleton, no-smuggling, closure status, refusal boundary, and machine-readable audit route from* PhilosophyFormalTargetLedger$^{\uparrow}$*;*
6. *U is the package non-smuggling row, recording that no export may bypass any of M, R, S, G, T by prose adjacency, hidden registry state, or an unchecked machine target;*
7. *H records componentwise* hsame *transport for the five sibling rows, non-smuggling row, replay, provenance, and local naming;*
8. *C records displayed* Cont *replay for package-map inspection, refusal inspection, registry-export inspection, gate inspection, formal-target inspection, and non-smuggling readback;*

9. *P is the* Pkg *provenance over* $\mathsf{PackageMapBoundary}^{\uparrow}$, $\mathsf{PhilosophyCannotClaimRegistry}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PackageExportDiscipline}^{\uparrow}}$ *row.*

The carrier has no coordinate for an export that preserves only a package map, only a registry row, only a refusal, only an export gate, only a formal target, or a hidden package-level synthesis outside the displayed packet.

Theorem 1323.2 (Package export discipline NameCert obligations)**.** *For the carrier of Definition 1323.1, the local* $\mathsf{NameCert}_{\mathsf{PackageExportDiscipline}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for $E = (M, R, S, G, T, U, H, C, P, N)$, package-map coverage through M, refusal-registry coverage through R, registry-export coverage through S, cannot-claim export-gate coverage through G, formal-target coverage through T, non-smuggling coverage through U, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the finite packet $E = (M, R, S, G, T, U, H, C, P, N)$. The first five rows are read as sibling NameCert surfaces: M supplies the package-map boundary, R supplies the cannot-claim registry, S supplies the registry-export surface, G supplies the cannot-claim export gate, and T supplies the formal-target ledger.

Read the non-smuggling row next. It states that a package export is not admitted through prose adjacency, hidden registry state, or unchecked target routing unless the same packet carries M, R, S, G, T.

Finally H, C, P, N transport, replay, source, and locally name the displayed rows. Since the carrier has no further coordinate, these rows exhaust the local $\mathsf{NameCert}_{\mathsf{PackageExportDiscipline}^{\uparrow}}$ obligation surface. □

Theorem 1323.3 (Package export discipline non-smuggling boundary)**.** *Every accepted* $\mathsf{PackageExportDiscipline}^{\uparrow}$ *export factors through the package-map row M, refusal-registry row R, registry-export row S, export-gate row G, formal-target row T, and non-smuggling row U before local naming. Removing either the cannot-claim gate row G or the registry-export row S permits only a partial package record, not an accepted package export discipline packet.*

Proof. Start with the obligation theorem. The export reads M, R, S, G, T before any package-level export can be named, and U blocks routes that skip one of those rows.

If S is absent, the packet has no displayed registry-export row recording registration, blocking, status, machine target, audit route, and non-smuggling data. A consumer may still inspect map or refusal data, but it cannot export a registry reading through $\mathsf{PackageExportDiscipline}^{\uparrow}$.

If G is absent, the packet has no displayed cannot-claim export gate. A refused claim can no longer be checked against an export decision, export grade, target route, and audit readback inside the same package export carrier.

The rows H, C, P, N preserve and name the accepted packet only after M, R, S, G, T, U have been supplied. Hence a route missing S or G leaves the accepted carrier rather than producing a package export discipline packet. □

Theorem 1323.4 (Package export discipline cross-reference handoff)**.** *The Lean-facing handoff for* $\mathsf{PackageExportDiscipline}^{\uparrow}$ *must preserve package-map, registry-export, cannot-claim, export-gate, formal-target, non-smuggling, transport, replay, provenance, and local NameCert rows as distinct*

fields. A consumer that collapses any of M, R, S, G, T cannot recover the cross-reference discipline required by chapter 1307, chapter 1318, chapter 1216, chapter 1319, and chapter 1282.

Proof. The five sibling rows are cited for different reasons. The package-map row supplies boundary coordinates, the refusal-registry row supplies cannot-claim status, the registry-export row supplies public export control, the export-gate row supplies the refused-claim gate, and the formal-target row supplies the machine-readable target ledger.

The non-smuggling, transport, replay, provenance, and local naming rows keep those citations tied to the same finite packet. They do not identify the rows with one another and do not create a replacement route when a sibling row is missing.

Therefore the handoff to Lean must use a field-faithful carrier. The proposed Lean follow-up must provide explicit Nontrivial PackageExportDiscipline$^{\uparrow}$ and FieldFaithful PackageExportDiscipline$^{\uparrow}$ instances in `lean4/BEDC/Derived/PackageExportDisciplineUp/TasteGate.lean`, with a target shaped for `BEDC.Derived.PackageExportDisciplineUp.PackageExportDisciplineTasteGate_single_carrie` □

Lean4 ✓ `BEDC.Derived.PackageExportDisciplineUp.PackageExportDisciplineTasteGate single carrier alignment`

Falsifiable prediction: Removing the cannot-claim gate row or registry-export row permits an export that the package discipline theorem rejects.

Independence witness: packagemapboundary, philosophycannotclaimregistry, registryexportsurface, cannotclaimexportgate, philosophyformaltargetledger: none is bijective to PackageExportDiscipline$^{\uparrow}$ because the export packet preserves package-map, registry-export, cannot-claim, export-gate, and formal-target rows simultaneously.

Closure status for NameCert$_{\text{PackageExportDiscipline}^{\uparrow}}$.

Constructive story: A PackageExportDiscipline$^{\uparrow}$ packet is generated from a package-map row, registry row, refusal row, export-gate row, formal-target row, synthesis row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier requires all of these rows before an export can be read as disciplined; no single package map, registry entry, refusal, gate, or formal target is enough by itself. The ledger story is anti-smuggling: the same finite BHist packet must carry map, registry, refusal, gate, target, and synthesis evidence, and the continuation routes expose exactly which rows a downstream export consumes.

Theory closure: TheoryGate(NameCert$_{\text{PackageExportDiscipline}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition, NameCert-obligation, non-smuggling-boundary, cross-reference-handoff, independence witness, and existing closure annotations for PackageExportDiscipline$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{PackageExportDiscipline}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim exports preserving only one constituent row or hidden package-level synthesis outside the displayed packet.

Upgrade path: To reach obligationClosure, the export-discipline carrier and non-smuggling boundary must be checked as finite-row dependency obligations.

Chapter 1324

A Concrete Naming Certificate for $\mathsf{ReviewAcceptanceGate}^{\uparrow}$

The $\mathsf{ReviewAcceptanceGate}^{\uparrow}$ packet realizes the research-programme route of section 9462.32 as a finite NameCert surface for operator review. It sits beside the failure-certificate packet of chapter 1233, the cannot-claim export gate of chapter 1319, the machine-interface boundary of chapter 1221, the kernel acceptance witness of chapter 549, and the kernel audit witness of chapter 546. Those siblings expose refusal, export blocking, machine-facing routing, kernel admission, and audit readback separately; this carrier binds proposal, decision, rejected alternatives, acceptance row, audit readback, and non-automation boundary before a consumer may read operator approval.

Definition 1324.1 (Review acceptance gate carrier)**.** *A* $\mathsf{ReviewAcceptanceGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, D, R, A, M, K, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the proposal row, naming the claim site, proposed paper or Lean target, requested gate, and submitted certificate packet;*
2. *D is the decision row, recording accept, refuse, request-more-evidence, cannot-claim, or machine-boundary refusal for the same proposal;*
3. *R is the rejected-alternatives row, listing rejected readings, unavailable proof routes, refused automation paths, and non-exportable claim variants;*
4. *A is the human acceptance row, present exactly when the operator accepts the proposal as a review decision rather than as an automated theorem result;*
5. *M is the* $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ *row, recording what the machine-facing route may expose and what it must refuse;*
6. *K is the kernel row, read through* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *and* $\mathsf{KernelAuditWitness}^{\uparrow}$ *when a Lean declaration or marker is part of the proposal;*
7. *Q is the blocked-claim row, consumed from* $\mathsf{FailureCertificate}^{\uparrow}$ *and* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *when the decision refuses approval or export;*
8. *H records componentwise* hsame *transport for proposal, decision, rejected alternatives, acceptance, machine boundary, kernel rows, blocked-claim rows, continuation, provenance, and local naming;*

9. *C records displayed* Cont *replay for reading S, D, R, A, M, K, Q without adding an invisible approving agent;*
10. *P is the* Pkg *provenance over* ReviewAcceptanceGate$^{\uparrow}$, FailureCertificate$^{\uparrow}$, CannotClaimExportGate$^{\uparrow}$, MachineInterfaceBoundary$^{\uparrow}$, KernelAcceptanceWitness$^{\uparrow}$, KernelAuditWitness$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ReviewAcceptanceGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for converting a machine-produced proposal, kernel acceptance stamp, audit readback, cannot-claim refusal, or failure certificate into operator approval without the displayed human acceptance row A.

Lean4 ✓ `BEDC.Derived.ReviewAcceptanceGateUp.ReviewAcceptanceGateTasteGate single carrier alignment`

Theorem 1324.2 (Review acceptance gate NameCert obligations)**.** *For the carrier of Definition 1324.1, the local* $\mathsf{NameCert}_{\mathsf{ReviewAcceptanceGate}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (S, D, R, A, M, K, Q, H, C, P, N)$, proposal exposure through S, decision exposure through D, rejected-alternative exposure through R, human acceptance exposure through A, machine-boundary exposure through M, kernel acceptance and audit exposure through K, refusal and export-block exposure through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ReviewAcceptanceGateUp.ReviewAcceptanceGateNameCertObligations`

Proof. Project the finite packet $G = (S, D, R, A, M, K, Q, H, C, P, N)$. The proposal row S fixes the claim site and requested gate, while D records the decision attached to that same proposal.

Read the boundary rows next. The rejected-alternatives row R records routes not accepted; the human acceptance row A is the only row that turns the decision into operator approval; the machine boundary row M and kernel row K expose machine and Lean-side evidence without identifying either with approval.

Finally inspect Q, H, C, P, N. Refusal and cannot-claim rows remain visible through Q, while transport, replay, provenance, and local naming preserve the same displayed packet. Since no further coordinate remains, the listed rows exhaust the local NameCert obligations. □

Theorem 1324.3 (Review acceptance gate human approval boundary)**.** *For every accepted* ReviewAcceptanceGate$^{\uparrow}$ *packet $G = (S, D, R, A, M, K, Q, H, C, P, N)$, a consumer may read operator approval only through the human acceptance row A together with the proposal and decision rows S, D. Kernel acceptance data in K, audit readback in K, machine-facing route data in M, and refusal data in Q do not by themselves produce an accepted operator decision.*

Lean4 ✓ `BEDC.Derived.ReviewAcceptanceGateUp.ReviewAcceptanceGateHumanApprovalBoundary`

Proof. Start with the carrier definition. It separates the proposal, decision, rejected alternatives, human acceptance, machine boundary, kernel rows, and blocked-claim rows as distinct coordinates.

The NameCert obligations above make that separation consumer-visible. A route that reads K obtains kernel acceptance or audit data; a route that reads M obtains machine-facing admissibility; a route that reads Q obtains refusal or export blocking. None of these rows is the human acceptance row.

Therefore the only approval route is the one that reads S, D, A in the same packet and then transports, replays, sources, and names that packet through H, C, P, N. □

Theorem 1324.4 (Review acceptance gate non-automation boundary)**.** *If the human acceptance row A is removed from an accepted* $\mathsf{ReviewAcceptanceGate}^{\uparrow}$ *packet, then no route through $S, D, R, M, K, Q, H, C, P, N$ can classify the packet as operator approved. The remaining rows may still expose a proposal, a machine boundary, a kernel or audit witness, a refusal, rejected alternatives, transport, replay, provenance, and local naming, but they cannot cross the operator-only gate.*

Proof. Remove A and read the remaining coordinates. The proposal and decision rows can still name a requested gate, and the rejected-alternatives row can still record what was refused.

The machine, kernel, audit, failure, and cannot-claim rows then remain evidence rows only. They may support or block a decision, but none of them has the operator-approval type carried by A.

Transport, continuation, provenance, and local naming preserve existing rows rather than creating a missing one. Hence a packet without A cannot pass the operator-only gate as an accepted review packet. □

Falsifiable prediction: Removing the human acceptance row prevents any accepted packet from passing the operator-only gate.

Independence witness: failurecertificate, cannotclaimexportgate, machineinterfaceboundary, kernelacceptancewitness, kernelauditwitness: none is bijective to ReviewAcceptanceGateUp because only ReviewAcceptanceGateUp preserves proposal, decision, rejected alternatives, acceptance row, and non-automation boundary in one packet.

The Lean TasteGate file must provide explicit $\mathsf{Nontrivial\ ReviewAcceptanceGate}^{\uparrow}$ and FieldFaithful ReviewAccepta instances in `lean4/BEDC/Derived/ReviewAcceptanceGateUp/TasteGate.lean`. The field-faithfulness witness must preserve $S, D, R, A, M, K, Q, H, C, P, N$ as distinct rows before a checked target can cite `BEDC.Derived.ReviewAcceptanceGateUp.taste_gate`.

Closure status for $\mathsf{NameCert}_{\mathsf{ReviewAcceptanceGate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ReviewAcceptanceGate}^{\uparrow}$ packet is generated from proposal, decision, rejected-alternative, human-acceptance, machine-boundary, kernel-acceptance, kernel-audit, blocked-claim, componentwise hsame transport, displayed Cont replay, package provenance, and local naming rows. The classifier reads operator approval only when the proposal and decision are paired with the explicit human acceptance coordinate; machine-produced proposals, kernel stamps, audits, and refusals remain separate rows. The ledger story is non-automation: every accepted review route is generated from finite BEDC evidence plus the displayed acceptance row, and transport or replay can preserve that row but cannot synthesize it from machine evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReviewAcceptanceGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, human-approval-boundary, non-automation-boundary, independence witness, Lean-follow-up boundary, and existing closure annotations for $\mathsf{ReviewAcceptanceGate}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReviewAcceptanceGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim conversion of machine proposals, kernel acceptance, audit readback, refusals, or failure certificates into approval without the human acceptance row.

Upgrade path: To reach obligationClosure, the review carrier and non-automation boundary must be checked as finite-row acceptance obligations.

Chapter 1325

A Concrete Naming Certificate for $\mathsf{RealWindowClassifierSeal}^{\uparrow}$

The $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ packet packages the finite-window classifier demanded by the Real-completion axiom-purity boundary. It sits after the explicit-modulus refusal surface of chapter 1200, the finite-window seal audit of chapter 1199, the regular Cauchy naming layer of chapter 371, and the readback carriers chapter 126, chapter 291, chapter 301, and chapter 38. Its job is to make the Real-facing classifier a displayed NameCert packet before any completion consumer reads a seal row.

Definition 1325.1 (Real window classifier seal carrier)**.** *A* $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, E, F, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite stream observations used by the completion consumer.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row for the dyadic values and tolerance ledger visible inside that window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row proving that the displayed dyadic readbacks are regular-Cauchy data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R have been displayed.*
5. *F is the finite-window classifier row, comparing accepted packets by the displayed stream window, dyadic readback, regular-Cauchy handoff, and Real seal.*
6. *A is the axiom-refusal row, recording absence of quotient streams, host real equality, selected limits, countable choice,* `Quot` *soundness, classical case splits, and proposition extensionality.*
7. *H records componentwise* hsame *transport for the window, dyadic, regular, seal, classifier, refusal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from S through D and R into E and F.*
9. *P records* Pkg *provenance over* $\mathsf{RealModulusPurityBoundary}^{\uparrow}$, $\mathsf{FiniteWindowRealSealAudit}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealWindowClassifierSeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-level real, a quotient stream, a selected limit, a non-displayed modulus, an ambient completeness theorem, or an axiom outside the finite packet.

Lean4 (definition) `BEDC.Derived.RealWindowClassifierSealUp.taste gate`

Theorem 1325.2 (Real window classifier seal NameCert obligations)**.** *For every accepted* $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ *carrier*

$$W = (S, D, R, E, F, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealWindowClassifierSeal}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window exposure through* S*, DyadicRatCore readback through* D*, RegSeqRat handoff through* R*, RealUp seal routing through* E*, finite-window classification through* F*, axiom-refusal exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 1325.1. The packet displays one finite stream window, one dyadic tolerance readback, one regular-Cauchy handoff, one Real seal row, one classifier row, and one refusal row.

The readback route is ordered. The StreamName row S exposes the finite window, the DyadicRatCore row D reads the dyadic data inside it, and the RegSeqRat row R is reached only after those finite observations have been displayed.

The Real seal and classifier are therefore downstream of the same finite data. The rows E and F read S, D, R rather than an ambient real object, while A records the axiom-purity refusals attached to that route.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no further coordinates, these rows exhaust the local NameCert obligations. □

Theorem 1325.3 (Real window classifier seal nonescape)**.** *An accepted* $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ *packet cannot expose quotient-stream equality, host real equality, selected limits, countable choice,* `Quot` *soundness, classical case splits, proposition extensionality, an undisplayed modulus, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness to a Real-completion consumer.*

Proof. Inspect the carrier coordinates in Definition 1325.1. The data rows are exactly S, D, R, E, F, A and the structural rows are exactly H, C, P, N.

The refusal row A is displayed alongside the classifier row F. Thus the same packet that authorizes the finite-window comparison also records that quotient streams, host real equality, selected limits, countable choice, `Quot` soundness, classical case splits, proposition extensionality, undisplayed moduli, and ambient completeness are not exported.

Transport, continuation, provenance, and naming preserve the displayed route. They do not add a second Real seal route or an axiom-bearing coordinate, so a completion consumer can read only the finite packet. □

Theorem 1325.4 (Real window classifier seal consumer handoff)**.** *Every Real-completion modulus consumer of an accepted* $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ *packet factors through*

$$S \longmapsto D \longmapsto R \longmapsto (E, F, A, H, C, P, N).$$

Consequently the consumer receives a finite-window classifier and a visible axiom-refusal row before it may read the $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the NameCert obligations of Definition 1325.2. They require the consumer-facing packet to display the StreamName window, DyadicRatCore readback, and RegSeqRat handoff before the RealUp seal row is available.

Next read the classifier row. Because F compares accepted packets only through S, D, R, E and the structural rows, the classifier is attached to the same finite window that produced the seal route.

The refusal row A is part of the same handoff. By Definition 1325.3, no quotient-stream equality, host real equality, selected limit, hidden modulus, completeness theorem, or forbidden axiom can be supplied after the classifier has been read. The consumer handoff is therefore exactly the displayed finite route. □

Falsifiable prediction: A Real-completion modulus consumer cannot read a RealUp seal from a RealWindowClassifierSealUp packet without first reading the finite StreamName window, DyadicRatCore readback, RegSeqRat handoff, classifier row, and axiom-refusal row.

Independence witness: realmoduluspurityboundary, finitewindowrealsealaudit, regularcauchyname, streamname, dyadicratcore, regseqrat, real: not bijective because the seal packet carries the finite window, dyadic readback, regular handoff, Real seal, classifier, axiom-refusal, transport, continuation, provenance, and local NameCert rows simultaneously.

Lean follow-up provides explicit carrier and TasteGate fields under `lean4/BEDC/Derived/RealWindowClassifi` preserving $S, D, R, E, F, A, H, C, P, N$ as distinct rows before a checked target may cite `BEDC.Derived.RealWindow`

Closure status for $\mathsf{NameCert}_{\mathsf{RealWindowClassifierSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{RealWindowClassifierSeal}^{\uparrow}$ packet is generated from a StreamName window, RegSeqRat readback, RealWindowBudget request, tail-meet row, RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier seals a real-facing read only through the finite window and regular readback rows that the budget admits. The ledger story is finite observation before real sealing: the window, tolerance, tail agreement, and RealUp rows are generated together from BHist data, and transport preserves them coordinatewise so no quotient stream, selected limit, or ambient completeness theorem can enter as a hidden classifier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealWindowClassifierSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, nonescape, consumer-handoff, Lean-follow-up boundary, and existing closure annotation for $\mathsf{RealWindowClassifierSeal}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealWindowClassifierSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim host-level reals, quotient streams, selected limits, non-displayed moduli, ambient completeness, or axioms outside the finite packet.

Upgrade path: To reach obligationClosure, the real-window seal and consumer handoff must be encoded with checked finite-window classifier obligations.

Chapter 1326

A Concrete Naming Certificate for $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$

The $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ packet realizes the research-programme synthesis of subsection 9462.32.11 as a composite finite NameCert carrier. It sits after the multi-history physics route of chapter 1298, the physics falsification surface of chapter 1293, the programme-strength ledger of chapter 1305, the verification-failure roadmap of chapter 1289, and the failure-certificate packet of chapter 1233. These siblings expose route, falsification, strength, verification-failure, and failure-certificate rows separately; this carrier binds them before any consumer reads the programme as a completed physical theory or as a single-axis verification claim.

Definition 1326.1 (Physics programme synthesis carrier)**.** *A* $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, F, L, V, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$ *row, recording the relational physics, law bridge, scientific-model bridge, quantum-state, and falsification-boundary route;*
2. *F is the* $\mathsf{PhysicsFalsificationSurface}^{\uparrow}$ *row, recording route-law-empirical-countermodel refusal data before completed physics may be claimed;*
3. *L is the* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *row, recording claim strength, dependency status, theory and verification status, bridge status, refusal, and local naming;*
4. *V is the* $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ *row, recording checked-report failure, downgrade, roadmap-boundary, cannot-claim, and upgrade-registry data;*
5. *E is the* $\mathsf{FailureCertificate}^{\uparrow}$ *row, recording the concrete failure certificate consumed when a programme route is blocked;*
6. *H records componentwise* hsame *transport for R, F, L, V, E, package provenance, and the local naming row;*
7. *C records displayed* Cont *routes for physical-route readback, falsification-surface inspection, strength-ledger inspection, verification-failure replay, and failure-certificate handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$, $\mathsf{MultiHistoryPhysicsRoute}^{\uparrow}$, $\mathsf{PhysicsFalsificationSurface}$ $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$, $\mathsf{VerificationFailureRoadmap}^{\uparrow}$, $\mathsf{FailureCertificate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose visible rows are $R, F, L, V, E, H, C, P, N$. It contains no completed physical theory, no empirical value row outside the cited physics surfaces, no verification-status promotion outside the ledger, and no failure route outside the displayed certificate and roadmap rows.

Lean4 (definition) `BEDC.Derived.PhysicsProgrammeSynthesisUp.TasteGate.PhysicsProgrammeSynthesisUp`

Theorem 1326.2 (Physics programme synthesis falsification non-escape)**.** *Every accepted* PhysicsProgrammeSynthe *packet reads physical ambition through the displayed route row R and falsification row F before it may inspect programme strength, verification failure, or failure certificates. A consumer cannot bypass F to claim completed physics, empirical adequacy, law completion, bridge completion, or axiom-pure success.*

Proof. Project the carrier of Definition 1326.1. The physical side of the packet is exactly R together with F.

Apply the sibling route and falsification surfaces. The multi-history physics route records the candidate physical route, while the physics falsification surface records the law, empirical, countermodel, bridge-audit, axiom-purity, transport, provenance, and naming rows that can block that route.

The remaining coordinates L, V, E, H, C, P, N replay or report the status of the same displayed route. None of them supplies a second physical-success coordinate. Hence a completed-physics read that omits F is outside the accepted $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.PhysicsProgrammeSynthesisUp.TasteGate.PhysicsProgrammeSynthesisFalsificationNoneso`

Theorem 1326.3 (Physics programme synthesis axis separation)**.** *For an accepted* $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ *carrier $S = (R, F, L, V, E, H, C, P, N)$, the physical-route axis, falsification axis, programme-strength axis, verification-failure axis, and failure-certificate axis remain separate NameCert coordinates. In particular, a stronger programme-strength row in L cannot erase a falsification row in F, a verification-failure row in V cannot be read as a physical countermodel by itself, and a failure certificate E cannot replace the route and ledger rows it diagnoses.*

Lean4 ✓ `BEDC.Derived.PhysicsProgrammeSynthesisUp.PhysicsProgrammeSynthesisTasteGate single carrier alignment`

Proof. Begin with the displayed tuple. The rows R, F, L, V, E are distinct carrier coordinates, and H transports them componentwise rather than by quotienting them together.

Read the programme-strength and verification-failure siblings next. The strength ledger reports dependency, theory, formal, bridge, and refusal coordinates; the verification-failure roadmap records audit report, failure, downgrade, roadmap, upgrade registry, and cannot-claim rows. These reports consume the physical and failure surfaces but do not identify them.

Finally inspect C, P, N. The continuation routes replay each named coordinate, the package row records the sibling provenance, and the local NameCert row seals the composite packet. Therefore no axis can discharge another axis by prose, status promotion, or certificate substitution. □

Independence witness: multi-history-physics-route, physics-falsification-surface, programme-strength-ledger, verification-failure-roadmap, failurecertificate: none is bijective with $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ because the composite packet preserves route, falsification, strength, verification-failure, failure-certificate, transport, replay, provenance, and local NameCert rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ packet is generated from a multi-history physics route, falsification surface, programme-strength ledger, verification-failure roadmap, failure-certificate row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts only the visible composite rows and keeps physical route, falsification, strength, verification failure, and failure certificate as separate axes. The ledger story is synthesis without completion: a consumer may inspect the programme only after route and falsification rows are displayed, while strength and verification reports remain diagnostic coordinates rather than a proof that physics, bridge, empirical adequacy, or law completion has been achieved.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, falsification-non-escape theorem, axis-separation theorem, independence witness, and existing closure annotations for $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim completed physical theory, empirical value rows outside the cited surfaces, verification-status promotion outside the ledger, or failure routes outside the displayed rows.

Upgrade path: To reach obligationClosure, the programme-synthesis carrier and axis-separation proof must be checked as finite composite packet obligations.

Chapter 1327

A Concrete Naming Certificate for $\mathsf{PhaseRealCompletionExit}^{\uparrow}$

The $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet realizes the real-and-completion exit certificate described by section 9462.32 as a finite four-face NameCert carrier. It sits after the dyadic arithmetic surface of chapter 291, the stream-window surface of chapter 126, the regular rational sequence surface of chapter 301, the real seal surface of chapter 38, and the exact completion boundary of chapter 1120. These siblings expose arithmetic, observation, regularity, real sealing, and boundary exactness separately; this carrier binds their exit rows before any consumer may treat the phase as a completed real object or as an ambient completion principle.

Definition 1327.1 (Phase real-completion exit carrier)**.** *A* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face recording the finite dyadic endpoint, tolerance, comparison, and arithmetic ledger used by the exit;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *face recording the finite observation windows through which the exit is read;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *face recording the regular rational sequence handoff generated from the dyadic and stream rows;*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *face recording the real seal reached only after D, S, Q have been exposed;*
5. *B is the* RealCompletionExactBoundaryUp *face recording that the same route supplies exact boundary information without exporting host completeness, quotient stream equality, or a selected limit outside the packet;*
6. *H records componentwise* hsame *transport for the dyadic, stream, regular, real, exact-boundary, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *routes for reading D, then S, then Q, then R, and finally the boundary row B;*
8. *P is the* Pkg *provenance over* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBoundaryUp, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose visible exit surface is $D, S, Q, R, B, H, C, P, N$. It has no coordinate for an ambient Cauchy completion, quotient-stream equality, selected real limit, host completeness proof, or a fused bridge token identifying the four local faces.

Theorem 1327.2 (Phase real-completion exit NameCert obligations)**.** *For the carrier of Definition 1327.1, the local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *surface consists exactly of carrier admission for $E = (D, S, Q, R, B, H, C, P, N)$, dyadic-face exposure through D, stream-window exposure through S, regular-rational handoff through Q, real-seal exposure through R, exact-boundary exposure through B, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed carrier. Its arithmetic, observation, regularity, real-seal, and exact-boundary faces are exactly D, S, Q, R, B.

The four local read faces are ordered by the continuation row C: the dyadic ledger is read together with the stream windows, the regular-rational handoff is reached from those rows, and the real seal is exposed only after the handoff has been fixed. The boundary row B then records exactness for the same route.

The rows H, P, N transport, source, and name that finite surface. Since the carrier lists no further coordinate, the displayed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.PhaseRealCompletionExitUp.PhaseRealCompletionExitTasteGate single carrier alignment`

Theorem 1327.3 (Phase real-completion exit noncollapse boundary)**.** *Every accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *consumer receives the dyadic face D, stream face S, regular face Q, real face R, exact-boundary face B, transport row H, replay row C, provenance row P, and local naming row N as distinct displayed rows. No accepted consumer may fuse the four local faces into a single bridge token, completeness token, quotient equality, selected limit, or host real without leaving the carrier.*

Proof. Start from Definition 1327.1. The rows D, S, Q, R occupy separate carrier coordinates, and the exact-boundary row B reports on the same route rather than replacing those coordinates.

Apply Definition 1327.2. Any accepted read must expose each local face before the local naming row is consumed, while H and C preserve componentwise transport and replay.

A fused bridge or completeness token would require a coordinate identifying the dyadic, stream, regular, and real faces as one object. No such coordinate appears in E, and the boundary row explicitly withholds ambient completion, quotient equality, and selected-limit data. □

Theorem 1327.4 (Phase real-completion exit objectwise face criterion)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *exit read is admissible precisely when it exposes the dyadic face D, stream face S, regular-rational face Q, real-seal face R, exact-boundary face B, and the support rows H, C, P, N as separate finite* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No aggregate bridge token, quotient-stream equality, selected limit, host completeness row, or fused formalstatus row may replace any one of these faces.*

Proof. The forward direction is read from the carrier of Definition 1327.1. Its visible coordinates are $D, S, Q, R, B, H, C, P, N$, so an admissible exit read has a finite row for each local face and for each support route.

For the reverse direction, apply Definition 1327.2. Once those rows are separately exposed, the local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ surface has the dyadic, stream, regular-rational, real-seal, exact-boundary, transport, replay, provenance, and naming obligations it requires.

The exclusions follow from Definition 1327.3. The five sibling references in the chapter introduction supply the objectwise dependency surface: dyadic arithmetic, stream observation, regular rational handoff, real sealing, and exact boundary are imported as distinct faces, so none of the listed aggregate tokens can stand in for them inside the accepted exit read. □

Theorem 1327.5 (Phase real-completion exit facewise audit obligation)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet* $E = (D, S, Q, R, B, H, C, P, N)$ *can satisfy the Phase Real and Completion exit obligation only by separately exposing the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* RealCompletionExactBoundar *rows. Each of the four roadmap faces keeps its own local closurestatus and formalstatus record: dyadic arithmetic through* D*, stream observation through* S*, regular rational handoff through* Q*, and real-completion exit through the paired* R, B *rows. No bridge token, fused completion row, quotient equality, selected limit, or host-completeness certificate may discharge those records jointly.*

Proof. Begin with the objectwise criterion of Definition 1327.4. It says that an admissible exit read exposes $D, S, Q, R, B, H, C, P, N$ as separate finite rows, so the audit obligation cannot be discharged by a single aggregate report.

The noncollapse boundary in Definition 1327.3 rules out the only possible shortcuts: bridge tokens, completion tokens, quotient equalities, selected limits, and host real data have no carrier coordinate in E. Thus any accepted audit must still display the dyadic, stream, regular, real, and exact-boundary faces before the local naming row is read.

The sibling chapters named in the introduction supply the local records that the audit reads. The dyadic face is governed by chapter 291, the stream face by chapter 126, the regular rational handoff by chapter 301, and the real-completion exit face by chapter 38 together with chapter 1120. The minimum-blocker result Definition 38.489 records why those local reports cannot be collapsed into one completion-facing audit.

Consequently the phase-exit packet can pass the roadmap audit only by exposing the five object rows while preserving the four local closurestatus and formalstatus records that produced them. □

Theorem 1327.6 (Phase real-completion exit obligation surface package)**.** *For any accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet, the obligation surface is exactly the carrier row* $E = (D, S, Q, R, B, H, C, P, N)$*, the five object faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* RealCompletionExactBoundaryUp*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *row. No quotient stream, selected limit, host-completeness row, or fused bridge token discharges that surface.*

Proof. Apply Definition 1327.2 to identify the displayed rows of the local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ surface. The carrier contributes exactly $E = (D, S, Q, R, B, H, C, P, N)$, with D, S, Q, R, B read as the five object faces and H, C, P, N read as transport, replay, provenance, and naming rows.

Apply Definition 1327.4 and Definition 1327.5. These results force the accepted exit read and the audit read to keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and RealCompletionExactBoundaryUp as separate finite faces before the local naming row is consumed.

Finally apply Definition 1327.3. It excludes the shortcut coordinates named in the statement, so quotient streams, selected limits, host-completeness rows, and fused bridge tokens cannot discharge the obligation surface. □

Theorem 1327.7 (Phase real-completion exit boundary-witness balance)**.** *For every accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet* $E = (D, S, Q, R, B, H, C, P, N)$*, any Real-completion exit consumer that reads the exact-boundary face* B *reads it over the same finite route* D, S, Q, R, H, C, P, N*.*

A witness-ledger consumer or exact-boundary consumer therefore cannot replace that route by quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, or a detached completion witness.

Lean4 ✓ `BEDC.Derived.PhaseRealCompletionExitUp.PhaseRealCompletionExitUp boundary witness balance`

Proof. Project the carrier of Definition 1327.1. The boundary row B is displayed as the exact-boundary face of the same packet whose route rows are D, S, Q, R, H, C, P, N.

Apply Definition 1327.2. It exposes the dyadic, stream, regular-rational, real-seal, exact-boundary, transport, replay, provenance, and local naming rows as the complete NameCert surface for the accepted packet.

The noncollapse boundary Definition 1327.3 and the objectwise face criterion Definition 1327.4 prevent any fused completion token from standing in for those rows. The facewise audit obligation Definition 1327.5 then identifies the five sibling records as the separate $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealCompletionExactBoundaryUp}$ faces of the same exit read.

The witness endpoint is the ledger surface of chapter 902, and the boundary endpoint is chapter 1120. Since both endpoints are consumed through the displayed route, a consumer that asks for quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, or a detached completion witness has left the accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ carrier. □

Theorem 1327.8 (Phase real-completion exit terminal-envelope exactness)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Any terminal Real-completion exit consumer that uses the* $\mathsf{FiniteRealRouteCommitment}^{\uparrow}$ *route of Definition 930.3 and the* $\mathsf{RealSealAdmissionCut}^{\uparrow}$ *route of Definition 1001.3 is exhausted by one finite envelope: the dyadic face D, the StreamName face S, the RegSeqRat face Q, the Real seal R, the exact-boundary face B, the route-commitment budget, window, readback, seal, and handoff rows, the admission-cut order, and the shared transport, replay, provenance, and naming rows H, C, P, N. In particular, such a consumer has no coordinate for quotient streams, selected limits, countable choice, host real equality, ambient completeness, or detached completion witnesses.*

Proof. First apply the boundary-witness balance of Definition 1327.7. It says that the exact-boundary face B is read over the same finite route D, S, Q, R, H, C, P, N and cannot be replaced by quotient stream equality, selected limits, countable choice, host real equality, ambient completeness, or a detached completion witness.

Next apply the route commitment theorem Definition 930.3. Any Real-facing or CauchyLimitSeal-facing consumer of the route commitment must display the budget, StreamName window, RegSeqRat readback, RealUp seal, and handoff rows before it reaches the terminal seal surface.

Then apply the admission-cut theorem Definition 1001.3. It forces the same terminal read to pass through the ordered finite boundary from request to budget, window, readback, seal, and handoff, so the consumer cannot bypass the budget or window side of the route.

Finally return to the carrier inventory in Definition 1327.1. Its only displayed exit coordinates are $D, S, Q, R, B, H, C, P, N$, with the route-commitment and admission-cut rows read through the named sibling theorems. Therefore the terminal consumer is exhausted by the stated finite envelope, and every listed completion principle outside that envelope remains unavailable. □

Theorem 1327.9 (Phase real-completion exit scoped primitive dependency binding)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Its scoped primitive dependency surface is exactly the conjunction of the* $\mathsf{DyadicRatCore}^{\uparrow}$ *face D, the* $\mathsf{StreamName}^{\uparrow}$ *face*

S, the $\mathsf{RegSeqRat}^{\uparrow}$ *face Q, the* $\mathsf{Real}^{\uparrow}$ *seal R, the* $\mathsf{RealCompletionExactBoundaryUp}$ *face B, and the support rows H, C, P, N, with each row read through the displayed* BHist*, componentwise* hsame*,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *coordinates. No quotient stream, selected limit, host equality, ambient completeness principle, fused bridge token, or single aggregate formalstatus row lies in the scope of this accepted packet.*

Proof. First apply Definition 1327.4 and Definition 1327.5. They identify the accepted exit read with the separate finite rows $D, S, Q, R, B, H, C, P, N$ and force the dyadic, stream, regular-rational, real-seal, and exact-boundary reports to remain objectwise visible.

Next apply Definition 1327.7 and Definition 1327.8. The boundary row B and every terminal consumer are read over the same displayed route, so the support rows H, C, P, N are not optional decorations; they are the transport, replay, provenance, and naming coordinates through which the primitive faces are consumed.

Finally use the sibling exact-boundary scope in Definition 1120.20 together with the terminal route of Definition 1120.13. These results close the exact-boundary side over the same four-face route and exclude quotient streams, selected limits, host equality, ambient completeness, bridge rows, and aggregate formalstatus witnesses from the accepted dependency surface. The displayed conjunction is therefore the whole scoped primitive binding for E. □

Theorem 1327.10 (Phase real-completion exit status-token product)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. The status read of* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *is the product of the local records of* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RealCompletionExactBoundary}$ *Each face keeps its own closurestatus, formalstatus, leantarget, and bridgestatus rows, read from the corresponding local chapter boundary. The phase-exit packet creates no aggregate formalstatus token, aggregate bridge row, aggregate Lean target, quotient stream, selected limit, or host-completeness row.*

Proof. Start with the facewise audit obligation in Definition 1327.5. It forces every accepted packet to expose the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealCompletionExactBoundaryUp}$ rows separately before any local naming row is consumed.

Next apply the noncollapse boundary of Definition 1327.3 and the scoped primitive binding of Definition 1327.9. These results keep the carrier coordinates $D, S, Q, R, B, H, C, P, N$ objectwise visible and exclude fused bridge tokens, quotient streams, selected limits, host-completeness rows, and single aggregate formalstatus rows from the accepted dependency surface.

Finally read the Real L10 status lookup and pullback through Definition 38.470 and Definition 38.471. They state that the dyadic, stream, regular, and real faces contribute their own closurestatus, formalstatus, leantarget, and bridgestatus rows by reference rather than by cached aggregate status data. The exact-boundary row B is the fifth face of the same displayed $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet, so the phase status read is precisely the five-face product and nothing else. □

Theorem 1327.11 (Phase real-completion exit local status gate)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet may serve as the Phase Real and Completion exit gate only when the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RealCompletionExactBoundaryUp}$ *faces are each read through their own local closurestatus, formalstatus, leantarget, and bridgestatus rows. No aggregate formalstatus token, fused bridge token, quotient stream equality, selected limit, host completeness row, or ambient completion premise can discharge this gate.*

Proof. Apply Definition 1327.10. It identifies the accepted status read as the product of the five local records and states that the phase-exit packet contributes no aggregate formalstatus row, bridge row, Lean target, quotient stream, selected limit, or host-completeness row.

Next apply Definition 1327.5. The audit read must expose the dyadic, stream, regular-rational, real, and exact-boundary faces separately before the local naming row can be consumed, so an exit gate cannot be accepted from a cached status summary.

Finally combine Definition 1327.3 with Definition 1327.9. These results keep the primitive scope at the displayed rows $D, S, Q, R, B, H, C, P, N$ and reject fused bridge tokens, quotient equalities, selected limits, host completeness, ambient completion data, and single aggregate formalstatus witnesses. Therefore the only accepted gate is the local five-face status read stated above. □

Theorem 1327.12 (Phase real-completion exit audit-clean threshold)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet serves the current local packet audit threshold exactly when the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealCompletionExactBoundaryUp}$ *faces are separately read through their own local closurestatus and formalstatus records, and each of the five verification-axis records is at least* $\mathsf{auditClean}$. *The* $\mathsf{RealCompletionExactBoundaryUp}$ *row is the same-route exact-boundary witness inside this packet audit, while Definition 1327.16 is the roadmap-facing projection that reads only the four roadmap objects. The threshold is not witnessed by an aggregate bridge token, quotient stream, selected limit, host completeness row, or single fused formalstatus row.*

Proof. For the forward direction, apply Definition 1327.11. Any accepted local packet audit must read the dyadic, stream, regular-rational, real, and exact-boundary faces through their own local status rows.

Next apply Definition 1327.10 and Definition 1327.5. The status read is the product of five local records, and the packet audit read must expose those records before the local naming row is consumed. Thus the local $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet audit can be met only by five objectwise verification-axis records, each at least $\mathsf{auditClean}$.

For the reverse direction, suppose the five faces have been read separately through their local closurestatus and formalstatus records, and each record is at least $\mathsf{auditClean}$. The local status gate supplies the required Phase Real and Completion exit read, while the status-token product keeps the five records in the same accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet.

Finally apply Definition 1327.3 and read the roadmap-facing projection through Definition 1327.16. The noncollapse result rejects aggregate bridge tokens, quotient streams, selected limits, host completeness, and single fused formalstatus rows; the projection states that the boundary row remains a same-route exact-boundary witness rather than a fifth roadmap object. □

Theorem 1327.13 (Phase real-completion exit classifier-envelope admission)**.** *Let $\mathcal{P}$ be an accepted finite Real L10 public classifier packet satisfying the public classifier audit envelope of Definition 38.782. Then $\mathcal{P}$ is accepted as a* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *classifier-envelope read exactly when its visible exit rows are the five face records*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{RealCompletionExactBoundaryUp}$$

together with displayed BHist, *componentwise* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support, and those five faces satisfy the audit-clean threshold of Definition 1327.12. No aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, host completeness row, metric-completeness datum, or countable-choice row is part of the accepted classifier-envelope read.*

Proof. First apply Definition 38.782. The Real L10 classifier packet is reduced to the four displayed public faces: local tightness rows, pairwise terminal classifier equivalences, triple common-tail cocycle data, and the objectwise closurestatus records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The same theorem excludes aggregate route-level formalstatus, aggregate bridge rows, quotient stream equality, selected limits, countable-choice data, metric-completeness data, and $\mathsf{Real}^{\uparrow}$ completeness claims from the classifier envelope.

Next apply Definition 1327.12. A $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet can serve the exit threshold only when the four Real L10 faces are read together with the RealCompletionExactBoundaryUp face, and each of the five local verification-axis records is at least auditClean. The exact-boundary row is therefore the fifth visible face required to consume the four-face Real classifier envelope inside the phase-exit carrier.

Finally use Definition 1327.10 and Definition 1327.3. They keep the five face records attached to their own local status tokens and keep $D, S, Q, R, B, H, C, P, N$ visible as separate finite rows. Thus an accepted classifier-envelope read is precisely the displayed five-face packet with BHist, hsame, Cont, Pkg, and local NameCert support, while the excluded aggregate, quotient, choice, limit, host-completeness, and metric-completeness rows remain outside the carrier. □

Theorem 1327.14 (Phase real-completion exit four-object boundary lift)**.** *An accepted four-object Phase Real and Completion read through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is admitted as a* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *read exactly when the* RealCompletionExactBoundaryUp *face is supplied over the same finite route. The resulting packet has displayed faces* $D, S, Q, R, B, H, C, P, N$ *and exports no quotient stream, selected limit, host equality, ambient completeness, metric-completeness datum, countable-choice row, aggregate formalstatus row, or bridge token.*

Proof. For the forward direction, apply Definition 126.500. It identifies the consumer-side four-object route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any completion-facing output is accepted.

Next apply Definition 126.504. The terminal pullback keeps that route finite and objectwise, so a consumer may transport the same route into the real-completion boundary without adding a quotient stream, selected limit, host equality, or ambient completeness premise.

Now use Definition 1120.6 and Definition 1120.9. These results supply the RealCompletionExactBoundaryUp face over the same four-object route and keep the exact-boundary row separate from the dyadic, stream, regular-rational, and real faces.

Finally apply Definition 1327.13. It admits precisely the five-face classifier-envelope read with displayed $D, S, Q, R, B, H, C, P, N$, BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}}$ support, while excluding the quotient, limit, host, completeness, choice, aggregate formalstatus, and bridge rows listed in the statement. □

Theorem 1327.15 (Phase real-completion exit scoped audit threshold)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet serves the local phase-exit packet audit exactly when the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are read through separate local closurestatus and formalstatus records whose closure-axis entries are at least* scopedClosed *and whose verification-axis entries are at least* auditClean*, while the* RealCompletionExactBoundaryUp *face is read as a same-route exact-boundary witness with its own local status. No aggregate formalstatus row, bridge token, quotient stream, selected limit, host completeness row, or fused completion witness may discharge the packet audit; Definition 1327.16 gives the roadmap-facing four-object read.*

Proof. First apply Definition 1327.10 and Definition 1327.11. They identify the accepted phase-exit status read as the product of the five visible local records and require the dyadic, stream, regular-

rational, real, and exact-boundary faces to remain separate through their own closurestatus and formalstatus rows.

Next apply Definition 1327.12. It supplies the verification-axis side of the local packet audit: the four roadmap faces meet auditClean inside the packet, and the exact-boundary face carries its own audit-clean same-route witness status without becoming a roadmap object. Neither a fused formalstatus row nor an external completion witness can replace those local records.

Now use Definition 1327.13. The accepted classifier-envelope read adds the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ support, while still excluding aggregate bridge tokens, quotient streams, selected limits, host-completeness rows, metric-completeness data, and countable-choice rows.

Finally read the two axes independently through section 9467.2 and the Phase Real and Completion criterion in section 9467.19, then apply the projection of Definition 1327.16. The roadmap gate is therefore met by the four local roadmap records whose closure-axis entries are at least scopedClosed and whose verification-axis entries are at least auditClean; the fifth packet face is the exact-boundary witness on the same route, with no aggregate row or fused completion witness standing in for the displayed faces. □

Theorem 1327.16 (Phase real-completion exit four-object threshold projection)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet satisfies the roadmap Phase Real and Completion exit criterion precisely when its* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *face records are each read through separate local closurestatus and formalstatus rows whose closure-axis entries are at least* scopedClosed *and whose verification-axis entries are at least* auditClean*. The* RealCompletionExactBoundaryUp *face is required only as the same-route exact-boundary witness over those four records; it cannot be counted as a fifth roadmap object, aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, or fused completion witness.*

Proof. Apply Definition 1327.14. It identifies the boundary face as a RealCompletionExactBoundaryUp witness supplied over the same finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, without adding quotient-stream equality, selected limits, host equality, ambient completeness, aggregate formalstatus data, or bridge tokens.

Next apply Definition 1327.15. The local $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ gate still reads five visible faces inside the packet, so the exact-boundary witness must carry its own local status row when the packet is audited.

Now use Definition 1327.10 together with Definition 1327.3. These results keep the status rows objectwise and prevent the boundary row from collapsing the four roadmap records into an aggregate token, quotient stream, selected limit, host-completeness row, or fused completion witness.

Finally read the roadmap criterion in section 9467.19 through the two-axis discipline of section 9467.2. The roadmap exit criterion names the four objects $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; the fifth face is the exact-boundary witness that lets the accepted packet expose the same route, not an additional roadmap object. □

Theorem 1327.17 (Phase real-completion exit five-face local audit)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet passes the local packet audit exactly when the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are read through separate local status rows and the same-route* RealCompletionExactBoundary *witness is read through its own local status row. The audit consumes those five status rows together with the displayed support rows* H, C, P, N *of the carrier* $E = (D, S, Q, R, B, H, C, P, N)$*; it consumes no aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, or fused completion witness.*

Proof. For the forward direction, apply the status-token product of Definition 1327.10. It identifies the accepted status read as the product of the dyadic, stream, regular-rational, real, and exact-boundary local records, with no aggregate formalstatus row or bridge row.

Next apply the local status gate of Definition 1327.11 and the audit-clean threshold of Definition 1327.12. These results force the local audit to read the four roadmap faces and the same-route exact-boundary face through separate local status rows before the packet can meet the audit threshold.

For the reverse direction, suppose those five local status rows are displayed with the support rows H, C, P, N. Apply the scoped audit threshold of Definition 1327.15 to admit the five-face packet audit, then project the roadmap side through Definition 1327.16.

Finally use the noncollapse boundary of Definition 1327.3. Since aggregate formalstatus rows, bridge tokens, quotient streams, selected limits, host-completeness rows, and fused completion witnesses have no carrier coordinate, they cannot discharge or replace any of the five local audit faces. □

Theorem 1327.18 (Phase real-completion exit public gate certificate)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet is the public paper-side gate certificate for the current Phase Real and Completion criterion exactly when the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *satisfy their local scoped-closure records and their local audit-clean records. The* $\mathsf{RealCompletionExactBoundaryUp}$ *face is used only as the same-route exact-boundary witness over those four records. The certificate exports no aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, host completeness row, metric-completeness row, countable-choice row, or fused completion witness.*

Proof. For the forward direction, apply Definition 1327.16. Any accepted packet that satisfies the Phase Real and Completion exit criterion must expose the four roadmap faces through separate local closurestatus and formalstatus rows, with closure-axis entries at least $\mathsf{scopedClosed}$ and verification-axis entries at least $\mathsf{auditClean}$. The same theorem keeps $\mathsf{RealCompletionExactBoundaryUp}$ as a boundary witness over the same route rather than as another roadmap object.

Next apply Definition 1327.15. It identifies the gate consumed by the chapter as the visible status read of the phase-exit packet, so the public certificate cannot be obtained from a cached route summary or from a single fused completion row.

For the reverse direction, suppose the four roadmap faces have their local scoped-closure and audit-clean records, and suppose the accepted packet supplies $\mathsf{RealCompletionExactBoundaryUp}$ only as the same-route exact-boundary witness. Apply Definition 1327.10 to repack those records as the product read of the accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ carrier.

Finally read the product through section 9467.2. The roadmap discipline keeps the paper closure axis, Lean verification axis, and bridge status axis independent. Therefore the certificate can be cited as a public paper-side gate while still refusing aggregate formalstatus rows, bridge tokens, quotient streams, selected limits, host completeness, metric completeness, countable choice, and fused completion witnesses. □

Theorem 1327.19 (Real L10 terminal envelope phase-exit equivalence)**.** *An accepted Real L10 terminal envelope satisfying Definition 38.510 is equivalent to the public* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *gate certificate of Definition 1327.18 exactly when the* $\mathsf{RealCompletionExactBoundaryUp}$ *face is supplied as the same-route exact-boundary witness over the ordered route* $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$, *and when the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are read only through their local scoped-closure and audit-clean records. Under this equivalence the Real envelope contributes the ordered L10 route and finite-window witness rows, while the phase-exit certificate contributes the same-route exact-boundary face and the displayed* BHist,

componentwise hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support. Neither side exports an aggregate formalstatus row, fused bridge token, quotient stream, selected limit, host completeness row, metric-completeness datum, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, start from the accepted Real L10 terminal envelope. Apply Definition 38.510 to read the ordered route and finite-window witness rows, then apply Definition 38.511 to keep the status evidence objectwise over the four local roadmap faces.

Next supply the same-route RealCompletionExactBoundaryUp face and apply Definition 1327.14. The boundary lift turns the ordered Real route into the five-face phase-exit read without adding quotient stream equality, selected limits, host completeness, metric-completeness data, countable choice, aggregate formalstatus data, or a fused bridge token.

Finally apply Definition 1327.18. The four roadmap faces already carry their local scoped-closure and audit-clean records, and the exact-boundary face is used only as the same-route witness, so the public $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ gate certificate is obtained.

For the reverse direction, suppose the public $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ gate certificate is displayed with the same-route RealCompletionExactBoundaryUp face and the four local roadmap records. Apply Definition 1327.18 to recover the objectwise public gate data, then apply Definition 1327.16 to project away the boundary face from the roadmap count while preserving the ordered four-object route.

Now apply Definition 38.508. The projected dyadic, stream, regular-rational, and Real faces, together with the displayed finite-window and support rows, recover the ordered Real terminal envelope. The same exclusion list remains in force: aggregate formalstatus rows, fused bridge tokens, quotient streams, selected limits, host completeness, metric-completeness data, countable-choice rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not envelope data. □

Theorem 1327.20 (Phase real-completion exit bridge-pending axis separation)**.** *Let* $E = (D, S, Q, R, B, H, C, P, N$ *be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Suppose the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *are read through their own local scoped-closure and audit-clean records, and suppose the* RealCompletionExactBoundaryUp *face is supplied on the same finite route. Then the public Phase Real and Completion exit gate remains admitted even when some component bridge rows stand at* `paperBridge` *and other component bridge rows stand at* `bridgeChecked`*. The packet reads bridge rows objectwise from the component boundaries and exports no aggregate* bridgeChecked *row, no fused standard bridge token, no quotient stream, no selected limit, and no host-completeness row.*

Proof. Apply Definition 1327.18. It identifies the public phase-exit gate with the four roadmap faces carrying local scoped-closure and audit-clean records, together with RealCompletionExactBoundaryUp only as the same-route exact-boundary witness.

Next apply Definition 1327.10. The accepted status read is the product of the local face records, so bridge rows are read from their component closurestatus blocks rather than from a single phase-level bridge coordinate.

Use Definition 1327.24 to expose the consumer view. The readback returns the dyadic, stream, regular-rational, real, and exact-boundary faces with the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support rows, while leaving component bridge rows attached to their own objects.

Finally apply Definition 1327.23 and read the result through the two-axis discipline of section 9467.2. A bridge row at `paperBridge` or `bridgeChecked` reports a component bridge state; it

neither collapses the scoped/audit exit gate nor blocks it once the four roadmap records and same-route exact-boundary witness are present. Hence the accepted packet admits the public gate while refusing an aggregate bridgeChecked export, a fused standard bridge token, a quotient stream, a selected limit, or a host-completeness row. □

Theorem 1327.21 (Phase real-completion exit component bridge-upgrade invariance)**.** *Let* $E = (D, S, Q, R, B, H, C, P, N)$ *be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet satisfying the public Phase Real and Completion gate of Definition 1327.18. Suppose one local bridge row among the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* RealCompletionExactBoundaryUp *faces is replaced by a stronger local bridge record, and suppose the corresponding closurestatus and formalstatus rows remain attached only to that face. Then the public phase-exit projection of* E *is unchanged: the same four roadmap records, same-route boundary witness, and support rows* H, C, P, N *are read by the gate. The replacement creates no aggregate* bridgeChecked *row, no fused standard bridge token, no quotient stream, no selected limit, no host-completeness row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. First apply Definition 1327.10. The status read of the accepted packet is the product of the component records, so strengthening one bridge row changes only that local factor and leaves the other face records objectwise.

Next apply Definition 1327.20. Mixed component bridge states are invisible to the scoped/audit gate once the four roadmap records and same-route RealCompletionExactBoundaryUp witness are present. Hence the public phase-exit projection is still admitted after the local bridge replacement.

Use Definition 1327.30 to read back the same exported data. The dyadic, stream, regular-rational, and real records remain the four roadmap rows, the boundary face remains the same-route witness, and H, C, P, N remain the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ support.

Finally apply Definition 1327.23. Because the strengthened bridge row stays local to its face, it cannot supply an aggregate bridgeChecked row, fused standard bridge token, quotient stream, selected limit, host-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. Therefore the bridge-status improvement preserves exactly the public phase-exit projection. □

Theorem 1327.22 (Phase real-completion exit standard-bridge source normal form)**.** *Let* $E = (D, S, Q, R, B, H, C, P, N)$ *be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Any standard-bridge candidate for* E *must take as its source exactly the five displayed face rows* D, S, Q, R, B, *namely* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and the same-route* RealCompletionExactBoundaryUp, *together with the four displayed support rows* H, C, P, N, *namely* BHist, *componentwise* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support. It may not use an aggregate formalstatus row, fused bridge token, quotient stream equality, selected limit, countable-choice row, host-completeness row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate as a source.*

Proof. Apply Definition 1327.10. It identifies the accepted status read as a product of local face records, so a bridge source must begin from the objectwise packet fields rather than from a route-level formalstatus token.

Next use Definition 1327.24. The consumer readback exposes precisely the dyadic, stream, regular-rational, real, and exact-boundary face rows with the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support. These are therefore the rows a bridge candidate can consume.

Now apply Definition 1327.26. The support rows cannot be replaced by a fused support token, external completion datum, quotient equality, selected limit, countable-choice row, host-completeness row, or metric-completeness row.

Finally apply Definition 1327.20. Bridge status remains a component report rather than a source normalizer. Hence the later standard bridge must target the displayed five-face and four-support source normal form, with no aggregate formalstatus row, fused bridge token, quotient stream equality, selected limit, countable-choice row, host-completeness row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate supplying the source. □

Theorem 1327.23 (Phase real-completion exit boundary-face nonsubstitution)**.** *Let $E = (D, S, Q, R, B, H, C, P, N$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. If any one of the four roadmap faces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *lacks its own local* scopedClosed *closure record and local* auditClean *verification record, or if the* RealCompletionExactBoundaryUp *face is not supplied on the same finite route, then E fails the current Phase Real and Completion exit gate. No aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, host-completeness row, metric-completeness row, countable-choice row, or fused completion witness can replace the missing local face.*

Proof. Apply Definition 1327.18. It identifies the public paper-side gate certificate with four roadmap faces that already carry their local scoped-closure and audit-clean records, together with RealCompletionExactBoundaryUp only as a same-route exact-boundary witness.

Next apply Definition 1327.16 and Definition 1327.15. These theorems separate the four roadmap objects from the boundary witness while still requiring every visible face in the accepted packet to be read through its own local closurestatus and formalstatus rows.

Finally use Definition 1327.10 and Definition 1327.3. The status product is objectwise, and the noncollapse boundary excludes aggregate formalstatus rows, bridge tokens, quotient streams, selected limits, host-completeness rows, metric-completeness rows, countable-choice rows, and fused completion witnesses from the carrier. Therefore a missing roadmap face record, or a boundary face not supplied on the same finite route, leaves the exit gate undischarged. □

Theorem 1327.24 (Phase real-completion exit facewise consumer readback)**.** *An accepted* PhaseRealCompletionExi *consumer reads back exactly the five local face records* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* RealCompletionExactBoundaryUp, *together with the supporting* BHist, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *rows. Conversely, any consumer that displays precisely those five local records and those support rows, with* RealCompletionExactBoundaryUp *supplied on the same finite route as the four roadmap faces, is admitted by the existing classifier-envelope and public-gate certificate. The readback exports no aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, metric-completeness data, countable-choice row, or fused completion witness.*

Proof. For the forward direction, apply Definition 1327.13. It identifies the accepted classifier-envelope read with the displayed carrier rows $D, S, Q, R, B, H, C, P, N$, so every accepted consumer receives the dyadic, stream, regular-rational, real, and exact-boundary faces together with BHist, hsame, Cont, Pkg, and local NameCert support.

Next use Definition 1327.16 and Definition 1327.23. These results keep $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as the four roadmap records and require RealCompletionExactBoundaryUp to remain a same-route boundary witness rather than an aggregate row or substitute face.

For the reverse direction, suppose a consumer displays exactly those five local records and support rows. Apply Definition 1327.13 to place the displayed rows inside the classifier envelope, then apply Definition 1327.18 to read the same data as the public phase-exit gate certificate.

Finally, the nonsubstitution theorem excludes the omitted rows from both directions of the readback. Thus quotient streams, selected limits, host completeness, metric completeness, countable choice, bridge tokens, aggregate formalstatus data, and fused completion witnesses cannot be consumed in place of the five displayed face records and their local support. □

Theorem 1327.25 (Phase real-completion exit status-face product injectivity)**.** *For accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packets, the public phase-exit gate read is determined exactly by the status-face product consisting of the four roadmap status records for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the same-route* $\mathsf{RealCompletionExactBoundaryUp}$ *witness and the displayed* BHist, *componentwise* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support. Equivalently, two accepted packet reads give the same public phase-exit gate certificate precisely when they have the same four roadmap status records, the same same-route exact-boundary witness, and the same displayed support rows. The comparison exports no aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, metric-completeness data, countable-choice row, or fused completion witness.*

Proof. For the forward direction, suppose two accepted packet reads have the same four roadmap status records, the same same-route $\mathsf{RealCompletionExactBoundaryUp}$ witness, and the same displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ support. Apply Definition 1327.10. It identifies the accepted status read with the objectwise product of those local records, so the two reads have the same status-face product.

Next apply Definition 1327.18. The public certificate consumes exactly the four roadmap faces with their local scoped-closure and audit-clean records, while the boundary face is used only as the same-route witness over those faces.

Use Definition 1327.23 to rule out alternative witnesses. Since no aggregate row, bridge token, quotient stream, selected limit, host-completeness row, or fused completion witness may replace a missing local face, the shared status-face product determines a single public gate read.

For the reverse direction, suppose the two accepted reads have the same public phase-exit gate certificate. Apply Definition 1327.24. The certificate reads back exactly the five local face records and the supporting BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. Applying the status-token product again identifies those readback rows with the four roadmap status records plus the same-route exact-boundary witness. Hence equality of the public gate readback gives equality of the displayed status-face product. □

Theorem 1327.26 (Phase real-completion exit support-row nonsubstitution)**.** *Let* $E = (D, S, Q, R, B, H, C, P, N)$ *be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. None of the support rows* H, C, P, N *may be replaced by an aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, metric-completeness datum, countable-choice row, or fused completion witness. The packet remains admitted only when componentwise* hsame *support through* H, *displayed* Cont *replay through* C, Pkg *provenance through* P, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support through* N *are present together with the displayed* D, S, Q, R, B *faces.*

Proof. Start from the carrier inventory in Definition 1327.1. The accepted packet has exactly the displayed rows $D, S, Q, R, B, H, C, P, N$, where H records componentwise hsame, C records displayed Cont routes, P records Pkg provenance, and N records the local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ row.

Apply Definition 1327.9. Its primitive scope is the same finite row list, so the support rows are load-bearing coordinates rather than annotations attached to an external completion or status object.

Next apply Definition 1327.24. Any accepted consumer reads back the five local face records together with the supporting BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert rows. Then use Definition 1327.25 to identify equality of the public gate read with equality of that displayed status-face product.

The remaining exclusions are the noncollapse exclusions already enforced by Definition 1327.3 and Definition 1327.23. Aggregate formalstatus rows, bridge tokens, quotient streams, selected limits, host or metric completeness data, countable-choice rows, and fused completion witnesses have no coordinate in E and cannot determine the same support-row product. Therefore H, C, P, N are non-substitutable in every accepted phase-exit packet. □

Theorem 1327.27 (Phase real-completion exit all support rows necessary)**.** *An accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *public gate read is valid exactly when it displays the five face records* D, S, Q, R, B *for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* RealCompletionExactBoundaryUp*, together with the support rows* H, C, P, N *for componentwise* hsame*, displayed* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$. *The boundary face* B *must lie on the same finite route as the four roadmap faces* D, S, Q, R*. No aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, metric-completeness datum, countable-choice row, or fused witness may replace any one of these listed rows.*

Proof. For the forward direction, apply Definition 1327.24. It identifies every accepted consumer read with exactly the five local face records and the displayed BHist, componentwise hsame, Cont, Pkg, and local NameCert support rows. The same readback requires RealCompletionExactBoundaryUp to be supplied on the same finite route as the four roadmap faces.

Next apply Definition 1327.25. Equality of the public gate read is equality of that status-face product, so deleting or replacing any one coordinate changes the displayed packet rather than preserving the accepted gate.

For the reverse direction, suppose the public read displays exactly $D, S, Q, R, B, H, C, P, N$ with B on the same finite route as D, S, Q, R. Apply Definition 1327.18 to read the four roadmap faces and the exact-boundary witness as the public Phase Real and Completion gate certificate.

Finally apply Definition 1327.26. It rules out aggregate formalstatus rows, bridge tokens, quotient streams, selected limits, host or metric completeness data, countable-choice rows, and fused witnesses as replacements for H, C, P, N. Together with the facewise readback, this proves that all five face rows and all four support rows are necessary and sufficient for the accepted public gate read. □

Theorem 1327.28 (Phase real-completion exit current local-status witness)**.** *Let* $E = (D, S, Q, R, B, H, C, P, N)$ *be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Suppose its four roadmap faces read the local records of* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and suppose its* RealCompletionExactBoundaryUp *face is supplied on the same finite route. Then* E *satisfies the roadmap Phase Real and Completion exit criterion in the current paper state: the four roadmap faces are each at least* scopedClosed *on the closure axis and at least* auditClean *on the verification axis, while the boundary face is the same-route exact-boundary witness. The packet introduces no aggregate formalstatus row, quotient stream, selected limit, host-completeness row, or fused bridge token.*

Proof. Read the local component blocks. $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ carry maturePackageClosed with bridgeChecked records, so each is above the required scopedClosed and auditClean thresholds. $\mathsf{RegSeqRat}^{\uparrow}$ carries maturePackageClosed with an axiomClean record, which is still above the required auditClean verification threshold for this roadmap gate.

The RealCompletionExactBoundaryUp block carries the same-route exact-boundary witness with maturePackageClosed and bridgeChecked. This supplies the boundary face required by the accepted packet, but it does not become a fifth roadmap object.

Apply Definition 1327.15 to the accepted packet. The threshold is met by reading the dyadic, stream, regular-rational, real, and exact-boundary faces through their own local closurestatus and formalstatus rows. In particular, the $\mathsf{RegSeqRat}^{\uparrow}$ `paperBridge` row is harmless here because this gate asks for audit-clean local verification, not for componentwise bridgeChecked status.

Finally apply Definition 1327.16. It projects the five-face packet read to the four roadmap objects $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with RealCompletionExactBoundaryUp used only as the same-route boundary witness. The projection therefore discharges the roadmap Phase Real and Completion exit criterion without adding an aggregate formalstatus row, quotient stream, selected limit, host-completeness row, or fused bridge token. □

Theorem 1327.29 (Real-completion exit dyadic-stream regularity)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet satisfying the current local-status witness of Definition 1327.28. Every accepted read of the current L10 gate must expose, in order, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance face D, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity face Q, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R. The* RealCompletionExactBoundaryUp *row B is available only as the same-route boundary witness over those four objectwise faces, with support through H, C, P, N. The read refuses quotient streams, selected limits, aggregate formalstatus rows, bridge tokens, countable choice, metric completeness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the status-token product of Definition 1327.10. It reads the accepted gate as the objectwise product of the four local records D, S, Q, R together with the same-route boundary row B and the support rows H, C, P, N.

Apply Definition 1327.16. The projection identifies the roadmap objects in the required order: dyadic tolerance first, finite StreamName window next, regular rational readback next, and the terminal Real seal last.

Now use Definition 1327.28. It certifies that these four faces meet the current L10 gate and that RealCompletionExactBoundaryUp remains a same-route witness rather than an additional roadmap object or a completion primitive.

The support rows cannot be replaced. By Definition 1327.26, componentwise hsame, displayed Cont, Pkg, and local NameCert support remain part of the same finite read and cannot be supplied by an aggregate status record or bridge token.

Finally apply Definition 1327.23. Quotient streams, selected limits, countable-choice rows, metric-completeness data, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in the packet. Hence every accepted L10 read remains objectwise through D, S, Q, R, with B only as the same-route boundary witness. □

Theorem 1327.30 (Phase real-completion exit component-status extraction)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet satisfying the current local-status witness of Definition 1327.28. Then the roadmap export of E is exactly the four local status records D, S, Q, R for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with the same-route* RealCompletionExactBoundaryU *witness B and the displayed* BHist, *componentwise* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *support rows H, C, P, N. Conversely, any accepted packet displaying precisely those four local records, the same-route boundary witness, and those support rows satisfies the current Phase Real and Completion exit read. No aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, countable-choice row, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim is extracted or supplied by this equivalence.*

Proof. For the forward direction, start from Definition 1327.28. It identifies the accepted packet as meeting the current roadmap exit criterion by the four local records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while the $\mathsf{RealCompletionExactBoundaryUp}$ face remains only the same-route boundary witness.

Apply Definition 1327.24. The consumer readback exposes exactly the face rows D, S, Q, R, B and the support rows H, C, P, N, namely the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ rows. Then use Definition 1327.27 to see that none of those support coordinates is optional or replaceable.

The terminal-envelope comparison in Definition 1327.19 keeps the same data aligned with the Real L10 route. It contributes the route and finite-window witness perspective, but it does not add an aggregate formalstatus row, aggregate bridge token, quotient stream, selected limit, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Therefore the extracted roadmap export is exactly the displayed four-record, one-boundary, four-support packet.

For the reverse direction, suppose an accepted packet displays exactly the four roadmap local records, the same-route $\mathsf{RealCompletionExactBoundaryUp}$ witness, and the support rows H, C, P, N. Apply Definition 1327.24 to admit the displayed rows as the public consumer read, and apply Definition 1327.27 to ensure the support rows are present in the required componentwise form.

Finally apply Definition 1327.28. The four local records are the roadmap objects, the boundary row is on the same finite route, and the support rows are the displayed BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. Hence the packet satisfies the current Phase Real and Completion exit read, with the same refusals of aggregate formalstatus, aggregate bridge token, quotient stream, selected limit, countable choice, metric completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1327.31 (Phase real-completion exit roadmap refusal boundary)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet satisfying the current local-status witness of Definition 1327.28. Its roadmap export consists exactly of the four face records $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, the same-route $\mathsf{RealCompletionExactBoundaryUp}$ witness, and the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ support rows. It exports no Dedekind-cut reals, open-cover compactness, Heine-Borel theorem, classical trichotomy, primitive quotient stream, selected limit, countable-choice row, metric-completeness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.*

Proof. Start from Definition 1327.28. The accepted packet meets the current Phase Real and Completion exit criterion by displaying the four roadmap faces at the required local closure and verification thresholds, while $\mathsf{RealCompletionExactBoundaryUp}$ remains only the same-route exact-boundary witness.

Apply Definition 1327.18. The public gate consumes the face records D, S, Q, R and the boundary witness B together with the support rows H, C, P, N. Thus the exported carrier is the finite tuple $E = (D, S, Q, R, B, H, C, P, N)$ and the support it displays through BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$.

Next apply Definition 1327.23. The excluded objects in section 9467.12 and section 9467.20 have no carrier coordinate in E: Dedekind-cut reals, open-cover compactness, Heine-Borel, classical trichotomy, primitive quotient streams, selected limits, countable choice, metric completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not among $D, S, Q, R, B, H, C, P, N$.

Finally read the same packet through the roadmap two-axis discipline of section 9467.2. Closure records and verification records remain attached to the local faces that supply them, so no refused off-table claim can be recovered by promoting a closure row into a verification row, by promoting a verification row into a closure row, or by fusing the component bridge reports into a completion

primitive. Hence the accepted packet exports exactly the stated roadmap boundary and nothing stronger. □

Theorem 1327.32 (Phase real-completion exit zero-source nonsubstitution)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet satisfying the current local-status witness of Definition 1327.28. An accepted* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet may serve as a zero-source consumer witness for the regular-rational and real-facing route only after E already displays the four local status records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, together with the same-route* $\mathsf{RealCompletionExactBoundaryUp}$ *face B. The* $\mathsf{RegularCauchyZero}^{\uparrow}$ *packet cannot replace any one of D, S, Q, R, cannot supply B as the same-route exact-boundary face, and cannot by itself discharge the roadmap audit-clean threshold for the Phase Real and Completion exit gate.*

Proof. Use Definition 624.22 and Definition 624.21. They identify $\mathsf{RegularCauchyZero}^{\uparrow}$ as a finite zero-source route through its rational zero seed, constant $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ zero row, tail-meet rows, and $\mathsf{Real}^{\uparrow}$ seal-facing terminal interface.

That finite route is a valid consumer witness, but it is not a phase-exit status product. Apply Definition 1327.10. The phase packet reads the roadmap gate through the objectwise product of D, S, Q, R and the same-route boundary face B, with support through H, C, P, N.

Next apply Definition 1327.28. The roadmap threshold is met by the four local status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while $\mathsf{RealCompletionExactBoundaryUp}$ remains the same-route boundary witness rather than a fifth roadmap object or a zero-source replacement.

Finally use Definition 1327.23. Since the phase gate is objectwise and status-token-local, a nearby zero-source packet has no coordinate that can stand in for a missing dyadic, stream, regular-rational, real, or exact-boundary face, and it cannot turn its terminal route into the audit-clean record required by the phase-exit roadmap gate. □

Definition 1327.33 (Phase real-completion exit cover-overlap compatibility)**.** *Let $\{E_i\}_{i \in I}$ be a finite family of accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packets with*

$$E_i = (D_i, S_i, Q_i, R_i, B_i, H_i, C_i, P_i, N_i)$$

on each cover member. The family is cover-overlap compatible when every common finite refinement of two cover members identifies the displayed row restrictions of the four roadmap faces

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

identifies the same-route $\mathsf{RealCompletionExactBoundaryUp}$ *witness, and identifies the support rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ *carried by H_i, C_i, P_i, N_i.*

The compatibility datum has no coordinate for quotient streams, selected limits, countable-choice witnesses, metric-completeness data, aggregate formalstatus rows, aggregate bridge tokens, standard bridge rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claims.*

Theorem 1327.34 (Phase real-completion exit cover-overlap no-confusion)**.** *Let $\{E_i\}_{i \in I}$ and $\{F_j\}_{j \in J}$ be cover-overlap compatible finite families of accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packets for the same public Phase Real and Completion consumer, where*

$$E_i = (D_i, S_i, Q_i, R_i, B_i, H_i, C_i, P_i, N_i)$$

and

$$F_j = (D'_j, S'_j, Q'_j, R'_j, B'_j, H'_j, C'_j, P'_j, N'_j).$$

After passing to a common finite overlap refinement, the two families determine the same public phase-exit gate read if and only if they display the same overlap restrictions of the four roadmap faces, the same-route RealCompletionExactBoundaryUp *witness, and the support rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$.

The common gate read recovers exactly those displayed rows. It recovers no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge token, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, assume the two finite families determine the same public phase-exit gate read. Apply Definition 1327.30 to the common read. It extracts exactly the four roadmap faces, the same-route RealCompletionExactBoundaryUp witness, and the displayed support rows BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$.

Restrict the extracted rows to every common finite overlap refinement. Since both families are cover-overlap compatible in the sense of Definition 1327.33, the restrictions on the E side and the F side are the same displayed rows. Apply Definition 1327.27 to rule out an omitted support coordinate, so equality of the public gate read forces agreement on every listed overlap row.

For the reverse direction, assume the common refinement displays the same four roadmap face restrictions, the same-route boundary witness, and the same support rows. Apply Definition 38.678 to identify the terminal four-face $\mathsf{Real}^{\uparrow}$ surface determined by the dyadic, stream, regular-rational, and real rows.

Next apply Definition 301.230. The regular-rational audit-threshold read has no second source once the four visible faces and support rows agree on the overlap refinement. With Definition 1327.30, the two phase-exit packets therefore expose the same public gate read.

All cited comparisons are finite-row comparisons. By Definition 1327.27, the only support rows available to the public gate are BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$. Thus no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge token, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim can be recovered from the overlap comparison. □

Theorem 1327.35 (Phase real-completion exit finite-cover gate gluing)**.** *Let* $\{E_i\}_{i\in I}$ *be a finite cover of compatible accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packets, with*

$$E_i = (D_i, S_i, Q_i, R_i, B_i, H_i, C_i, P_i, N_i)$$

on each cover member. Suppose the family is cover-overlap compatible in the sense of Definition 1327.33. Then the cover glues to one accepted packet whose public read is the same Phase Real and Completion gate if and only if every E_i *displays the same four roadmap local status records for* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, supplies the same-route* RealCompletionExactBoundaryUp *witness, and agrees on all overlap restrictions of the support rows* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$.

The glued packet has exactly those four roadmap status records, the same-route boundary witness, and those support rows as its public Phase Real and Completion gate read. It exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge token, standard bridge row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the forward direction, restrict the glued public packet to each finite cover member. Apply Definition 1327.33 to read those restrictions through the required common finite refinements, and apply Definition 1327.34 to compare them as one public phase-exit gate read.

Now apply Definition 1327.28 to read the common gate as the four local status records for $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with RealCompletionExactBoundaryUp only

as the same-route boundary witness. Apply Definition 1327.30 on each restriction. It identifies the data visible on every E_i as exactly the four roadmap records, the same boundary witness, and the displayed BHist, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ support rows. Since the restrictions come from one glued public read, the cover members agree on those rows over every overlap.

For the reverse direction, assume the finite cover displays the same four local records, the same-route RealCompletionExactBoundaryUp witness, and overlap agreement on the listed support rows. The compatibility hypothesis gives the common finite refinement rows of Definition 1327.33, and Definition 1327.34 identifies the public phase-exit gate read determined by those rows.

Apply Definition 301.227 to glue the regular-rational and real-facing audit-threshold sections over the shared finite four-face surface.

Next apply Definition 301.228. This makes the glued read independent of the chosen finite refinement and keeps the status records attached to the same dyadic, stream, regular-rational, and real faces. Apply Definition 38.812 to identify the resulting finite Real L10 cover certificate with one objectwise Phase Real and Completion exit certificate.

Finally repack the certificate as a $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet by Definition 1327.28 and read it back by Definition 1327.30. The public gate is therefore the same gate precisely when the cover members display the same four roadmap records, same-route boundary witness, and support rows.

All constructions named above use only finite cover rows and their overlap restrictions. The refused quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge token, standard bridge row, and ambient $\mathsf{Real}^{\uparrow}$ completeness claim have no carrier coordinate in the displayed packet, so none is exported by the gluing. □

Theorem 1327.36 (Phase real-completion exit RegSeqRat common-tail pullback)**.** *Let $E = (D, S, Q, R, B, H, C, P, N)$ be an accepted* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet. Suppose a RegSeqRat common-tail triple-overlap cocycle is displayed on the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows as S, D, Q, R, with the same support rows* BHist, *componentwise* hsame, Cont, Pkg, *and* NameCert *used by H, C, P, N. Then that common-tail cocycle pulls back to the same-route* RealCompletionExactBoundaryUp *face B of E.*

Conversely, if the boundary face B of an accepted $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *packet is supplied over such a RegSeqRat common-tail cocycle, then restricting the packet to the displayed four face rows recovers the RegSeqRat common-tail descent surface. In both directions the construction exports no quotient stream, selected limit, countable-choice row, metric-completeness datum, aggregate formalstatus row, aggregate bridge token, standard bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. For the pullback direction, first read the RegSeqRat input by Definition 301.190. The triple-overlap surface is exactly the common finite surface on the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Apply Definition 301.191. The pairwise common-tail rows descend to one displayed four-face boundary over the same stream, dyadic, regular-rational, and real records. Since these rows are identified with S, D, Q, R and the support rows H, C, P, N of E, Definition 1327.35 glues the finite cover into the objectwise Phase Real and Completion gate on the same displayed packet.

Now apply Definition 1327.30. The public read of the glued packet exposes the same-route RealCompletionExactBoundaryUp witness as the boundary coordinate B, rather than as an added completion carrier. Thus the common-tail cocycle pulls back to the boundary face already present in E.

For the converse direction, start with an accepted phase-exit packet whose boundary face B is supplied over the displayed common-tail cocycle. The finite-cover gluing theorem reads that boundary through the same four local status records and the same support rows, while the component-status extraction theorem prevents any row from being absorbed into a route-level token.

Restricting to the S, D, Q, R faces therefore leaves exactly the RegSeqRat common-tail triple-overlap data. Applying Definition 301.191 to those restricted rows recovers the common-tail descent surface, with the same BHist, hsame, Cont, Pkg, and NameCert provenance.

All steps are objectwise over the displayed finite rows. Quotient streams, selected limits, countable choice, metric-completeness data, aggregate formalstatus rows, aggregate bridge tokens, standard bridge rows, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E or in the RegSeqRat common-tail descent boundary, so the pullback and the restriction cannot export them. □

Falsifiable prediction: If a consumer fuses the dyadic, stream, regular, and real face reports into a single bridge token or completeness token, the $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet is refuted.

Independence witness: dyadicratcore, streamname, regseqrat, real, realcompletionexactboundary: no single sibling is bijective to the phase product because the packet binds dyadic arithmetic, finite stream observation, regular rational handoff, real sealing, exact boundary, transport, replay, provenance, and local naming in one displayed exit carrier.

Theorem 1327.37 (Phase real-completion exit TasteGate primality rows)**.** *Every accepted* PhaseRealCompletionEx *packet carries four separate TasteGate primality rows: the falsifiable fused-token refusal above, the sibling independence witness against dyadicratcore, streamname, regseqrat, real, and realcompletionexactboundary, the Lean 'Nontrivial' witness pair for distinct accepted and rejected carriers, and the Lean 'FieldFaithful' projection over $D, S, Q, R, B, H, C, P, N$. Hence no one-element carrier, field-erasing classifier, aggregate completion token, or hidden sibling bijection discharges the chapter.*

Proof. The carrier inventory in Definition 1327.1 displays the finite exit packet as $E = (D, S, Q, R, B, H, C, P, N)$. Thus the paper surface already names the fields whose preservation is tested by the local TasteGate.

The falsifiability and independence rows immediately above supply the two external primality tests: a consumer that fuses the dyadic, stream, regular, and real face reports into one bridge or completeness token refutes the packet, and none of the five sibling chapters is bijective to the phase product.

The remaining two rows are the checked 'Nontrivial' and 'FieldFaithful' witnesses in `lean4/BEDC/Derived/Phas` The 'Nontrivial' instance supplies distinct accepted and rejected carriers, while the 'FieldFaithful' instance projects the classifier through the displayed fields $D, S, Q, R, B, H, C, P, N$ rather than through an erased aggregate.

Together these four rows block each collapse mode. A one-element carrier violates 'Nontrivial', a field-erasing classifier violates 'FieldFaithful', an aggregate completion token violates the falsifiability row, and a hidden sibling bijection violates the independence witness. Therefore the accepted packet can discharge the chapter only as the displayed finite exit carrier with all four primality rows present. □

Lean4 ✓ `BEDC.Derived.PhaseRealCompletionExitUp.taste gate`

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseRealCompletionExit}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ packet is generated from dyadic, stream, regular-rational, Realseal, RealCompletionExactBoundaryUp, componentwise hsame, Cont, Pkg, and local NameCert rows. The exact-boundary face is a same-route witness over those rows, not a bridge or completion primitive. The classifier accepts the phase exit

only as the visible tuple of finite rows; no single face can stand for completion, and quotient equality, selected limits, host completeness, aggregate formalstatus, and bridge tokens stay outside the carrier. The local TasteGate instance is BEDC.Derived.PhaseRealCompletionExitUp.taste_gate.

Theory closure: TheoryGate(NameCert$_{\mathsf{PhaseRealCompletionExit}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1327.9 and Definition 1327.18 form the mature paper boundary: the accepted packet is the dyadic, stream, regular-rational, Real-seal, exact-boundary, componentwise hsame, Cont, Pkg, and local NameCert same-route gate.

Formal verification: FormalStatus(NameCert$_{\mathsf{PhaseRealCompletionExit}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PhaseRealCompletionExitUp.PhaseRealCompletionExitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: Standard-bridge verification remains unsupplied, and the packet exports no aggregate formalstatus row, bridge token, quotient stream, selected limit, host-completeness row, or metric-completeness row.

Upgrade path: The next formal obligation is standard-bridge verification for the already axiom-clean same-route gate; no aggregate formalstatus, bridge token, quotient stream, selected limit, host completeness, or metric completeness is supplied here.

Chapter 1328

A Concrete Naming Certificate for ResearchProgrammeExecutionGate$^{\uparrow}$

The ResearchProgrammeExecutionGate$^{\uparrow}$ packet realizes the programme route of section 9462.32 inside the auditable-generativity discipline of chapter 9462. It sits beside the strength ledger of chapter 1305, the strength-gradient ledger of chapter 1314, the kernel phase refusal packet of chapter 1288, the verification-failure roadmap of chapter 1289, and the name-eligibility proof of chapter 1309. Those sibling chapters expose strength, gradient, phase refusal, verification failure, and name eligibility separately; this carrier binds them into one finite execution gate before a programme target is accepted.

Definition 1328.1 (ResearchProgrammeExecutionGate carrier)**.** *A* ResearchProgrammeExecutionGate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, P, D, S, F, B, R, H, C, Q, N)$$

with the following displayed rows:

1. *T is the target row naming the programme target proposed for execution;*
2. *P is the current-phase row naming the phase in which the target is being considered;*
3. *D is the dependency-readiness row exposing the lowest open ready phase and its blockers;*
4. *S is the strength-report row inherited from the programme strength and gradient ledgers;*
5. *F is the formal-status row recording the theorem, audit, and axiom-clean status currently visible for the target;*
6. *B is the bridge row recording whether a standard-bridge consumer is present, absent, or refused;*
7. *R is the refusal row for targets outside the displayed lowest open ready phase;*
8. *H records componentwise* hsame *transport for the target, phase, dependency, strength, formal-status, bridge, refusal, replay, provenance, and naming rows;*
9. *C records displayed* Cont *replay of the route through P, D, S, F, B, R before T is accepted;*
10. *Q is the* Pkg *provenance over* ProgrammeStrengthLedger$^{\uparrow}$, ProgrammeStrengthGradientLedger$^{\uparrow}$, KernelPhaseRefusal$^{\uparrow}$, VerificationFailureRoadmap$^{\uparrow}$, NameEligibilityProof$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden priority override, off-phase target promotion, unreported blocker discharge, bridge status inferred from formal status, or name eligibility supplied outside the displayed local certificate.

Lean4 (definition) `BEDC.Derived.ResearchProgrammeExecutionGateUp.ResearchProgrammeExecutionGateUp`

Theorem 1328.2 (ResearchProgrammeExecutionGate NameCert obligations)**.** *For the carrier of Definition 1328.1, the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}$ *surface consists exactly of carrier admission for $G = (T, P, D, S, F, B, R, H, C, Q, N)$, target exposure through T, current-phase exposure through P, lowest-open-layer dependency readiness through D, strength exposure through S, formal-status exposure through F, bridge exposure through B, refusal exposure through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through Q, and local naming through N.*

Proof. Project the finite packet $G = (T, P, D, S, F, B, R, H, C, Q, N)$. The first seven rows expose the target, current phase, dependency readiness, strength report, formal-status report, bridge row, and refusal row that determine whether the programme route may be executed.

Read the structural rows next. Transport through H preserves the displayed coordinates, replay through C keeps the consumer on the same route, provenance through Q records the sibling packets supplying the rows, and N names the accepted carrier.

Since the carrier contains no further coordinate, these rows exhaust the local NameCert obligations. A purported execution gate that omits one of them is a projection or sibling report, not the accepted $\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}$ packet. □

Theorem 1328.3 (ResearchProgrammeExecutionGate lowest-open-layer readiness)**.** *Every accepted* $\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}$ *packet reads the current-phase row P and dependency-readiness row D before it accepts the target row T. The accepted target must lie in the displayed lowest open ready phase, and any target outside that phase is routed to the refusal row R before the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}$ *row is consumed.*

Proof. Start with the carrier obligations of Definition 1328.2. They require the current phase and dependency-readiness rows to be visible in the same finite packet as the target row.

Use the kernel phase refusal sibling together with the verification-failure roadmap. The phase row identifies which layer is ready, while the dependency-readiness row records blockers that prevent a stronger target from being accepted.

Finally read the displayed replay route C. A target in the lowest open ready phase reaches T only after P and D are displayed; a target outside that phase reaches R before N. Thus acceptance and refusal are both carried by the same local NameCert route. □

Theorem 1328.4 (ResearchProgrammeExecutionGate noninflation boundary)**.** *An accepted* ResearchProgrammeE *packet cannot promote a target above the strength row S, formal-status row F, bridge row B, dependency-readiness row D, or refusal row R displayed in its carrier. In particular, a consumer may not treat an unchecked target as checked, an unbridged target as bridge-checked, an off-phase target as ready, or a refused target as accepted.*

Proof. Read the strength and status rows from the carrier. The programme strength ledger and gradient ledger supply the reported strength S, while F and B keep formal and bridge status as separate coordinates.

The dependency and refusal rows fix the execution boundary. If D reports an open blocker or R records an off-phase refusal, no structural row can erase that fact: hsame transports displayed coordinates, Cont replays displayed reads, and Pkg provenance records the same packet.

Therefore any promoted reading asks for a coordinate absent from $G = (T, P, D, S, F, B, R, H, C, Q, N)$. The classifier rejects that reading as outside the execution gate. □

Falsifiable prediction: Any accepted execution-gate packet that selects a target outside the displayed lowest open ready phase refutes the carrier.

Independence witness: programmestrengthledger, programmestrengthgradientledger, kernelphaserefusal, verificationfailureroadmap, nameeligibilityproof: none is bijective with ResearchProgrammeExecutionG because the execution gate simultaneously carries target, current phase, dependency readiness, strength report, formal-status row, bridge row, refusal row, transport, replay, provenance, and local NameCert rows.

The Lean surface uses explicit `Nontrivial ResearchProgrammeExecutionGateUp` and `FieldFaithful ResearchProgrammeExecutionGateUp` instances in `lean4/BEDC/Derived/ResearchProgrammeExecutionGateUp/` The carrier keeps $T, P, D, S, F, B, R, H, C, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}$.

Constructive story: A $\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}$ packet is generated from a target row, current phase, dependency readiness, strength report, formal-status row, bridge row, refusal row, componentwise transport, displayed replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1328.1, Definition 1328.2, Definition 1328.3, and Definition 1328.4 specify the finite execution-gate packet and its noninflation boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResearchProgrammeExecutionGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, standard bridge checking, downstream execution scheduling, or any target outside the displayed lowest open ready phase.

Upgrade path: Lean closure requires Nontrivial ResearchProgrammeExecutionGateUp and FieldFaithful ResearchProgrammeExecutionGateUp instances for the same finite execution-gate packet.

Chapter 1329

A Concrete Naming Certificate for $\mathsf{SmugglingPatternClassifier}^{\uparrow}$

The $\mathsf{SmugglingPatternClassifier}^{\uparrow}$ packet realizes the smuggling discipline of subsection 9462.30.12 as one finite refusal-and-gate NameCert surface. It sits beside the host primitive leakage boundary of chapter 1157, the quotient-soundness boundary of chapter 634, the refutation boundary of chapter 500, and the cannot-claim export gate of chapter 1319. Those siblings refuse host imports, quotient shortcuts, invalid refutation readings, and checked-target export escapes separately; this carrier classifies philosophical, mathematical, and host smuggling requests before any external supply can be read as BEDC-internal.

Definition 1329.1 (Smuggling-pattern classifier carrier)**.** *A* $\mathsf{SmugglingPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, I, G, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the external-supply row, recording whether the requested supply is philosophical, mathematical, or host-provided;*
2. *I is the internal-object request row, recording the BEDC-internal reading the consumer attempts to make from that supply;*
3. *G is the refusal-or-gate row, recording the accepted certificate gate, the typed gap route, or the refusal boundary for the request;*
4. *Q is the sibling-provenance row, recording the relevant host-leakage, quotient-soundness, refutation-boundary, cannot-claim, and logic/meta-loop surfaces;*
5. *H records componentwise* hsame *transport for supply, request, gate, sibling provenance, replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay through which a consumer reads E, I, G, Q, H without adding a hidden external object;*
7. *P records* Pkg *provenance over chapter 1157, chapter 634, chapter 500, chapter 1319, chapter 1261,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SmugglingPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for treating an external entity, mature mathematical object, host primitive, quotient witness, global truth, or checked export target as BEDC-internal apart from the displayed gate or refusal row.

Lean4 (definition) `BEDC.Derived.SmugglingPatternClassifierUp.SmugglingPatternClassifierUp`

Theorem 1329.2 (Smuggling-pattern classifier NameCert obligations)**.** *For the carrier of Definition 1329.1, the local* $\mathsf{NameCert}_{\mathsf{SmugglingPatternClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for* $S = (E, I, G, Q, H, C, P, N)$*, requested external-supply exposure through* E*, internal-object request exposure through* I*, refusal-or-gate routing through* G*, sibling provenance through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the displayed packet $S = (E, I, G, Q, H, C, P, N)$. The external-supply row E classifies the source of the attempted import, while I records the internal reading requested by the consumer.

The routing row G is then read against the sibling-provenance row Q. A certified object must pass through an accepted gate; otherwise the request is sent to a typed gap or refusal boundary supplied by the host-leakage, quotient-soundness, refutation-boundary, cannot-claim, or logic/meta-loop surface.

Finally H, C, P, N transport, replay, source, and name the same displayed decision. Since no additional coordinate remains, these rows exhaust the local NameCert obligations for the classifier packet. □

Lean4 ✓ `BEDC.Derived.SmugglingPatternClassifierUp.SmugglingPatternClassifierTasteGate single carrier alignment`

Theorem 1329.3 (Smuggling-pattern classifier nonescape)**.** *Every accepted* $\mathsf{SmugglingPatternClassifier}^{\uparrow}$ *consumer must expose the requested external-supply row, the internal-object request row, a refusal-or-gate row, sibling provenance,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before any smuggled object can be read as BEDC-internal. If the refusal-or-gate row does not display an accepted certificate gate, the requested object remains outside BEDC as a gap, socket report, or refusal.*

Proof. Read the consumer through the rows of Definition 1329.1. The only row that can decide between admission and refusal is G, and it is visible only together with the supply row E and the requested internal reading I.

The sibling-provenance row Q identifies the boundary surface that handles the request. Host primitive leakage, quotient soundness, refutation boundaries, cannot-claim export, and logic/meta-loop refusals each block one escape channel, but none supplies a hidden internal object outside G.

The structural rows H, C, P, N preserve the same decision during transport, replay, provenance, and naming. Therefore a request without an accepted gate cannot pass through the classifier as internal BEDC content. □

Theorem 1329.4 (Smuggling-pattern classifier TasteGate handoff)**.** *The Lean-facing handoff for* $\mathsf{SmugglingPatternClassifier}^{\uparrow}$ *must preserve external-supply, internal-request, refusal-or-gate, sibling-provenance, transport, replay, provenance, and local NameCert rows as distinct fields. Collapsing* E, I, G, Q *removes the field that distinguishes philosophical, mathematical, and host smuggling from an accepted certificate route.*

Proof. The classifier detects smuggling by comparing an external supply with the internal reading requested from it. This comparison requires E and I to remain separately projected.

The decision is then routed through G and sourced through Q. If either row is merged into the request, a consumer can no longer tell whether the request was admitted by certificate, routed to a gap, or refused by a sibling boundary.

Rows H, C, P, N preserve the same classified packet after the decision. The TasteGate handoff therefore keeps all eight rows field-faithful before the alignment theorem can be cited. □

Falsifiable prediction: Every accepted SmugglingPatternClassifierUp consumer must expose the requested external-supply row, the internal-object request row, a refusal-or-gate row, sibling provenance, hsame transport, Cont replay, Pkg provenance, and local NameCert before any smuggled object can be read as BEDC-internal.

Independence witness: hostprimitiveleakage, quotientsoundnessboundary, refutationboundary, cannotclaimexportgate, logiccontradictionmetaloop: none jointly stores the three-form classifier for philosophical, mathematical, and host smuggling together with refusal-or-gate routing, sibling provenance, transport, replay, provenance, and local NameCert rows.

The Lean follow-up must provide Nontrivial SmugglingPatternClassifier$^{\uparrow}$ and FieldFaithful SmugglingPatternClassif instances in `lean4/BEDC/Derived/SmugglingPatternClassifierUp/TasteGate.lean`. The field-faithful route must preserve E, I, G, Q, H, C, P, N before a consumer may cite `BEDC.Derived.SmugglingPatternCl`

Closure status for NameCert$_{\text{SmugglingPatternClassifier}^{\uparrow}}$.

Constructive story: A SmugglingPatternClassifier$^{\uparrow}$ packet is generated from a displayed external-supply row, internal-object request row, refusal-or-gate row, sibling-provenance row, componentwise transports, replay route, package provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SmugglingPatternClassifier}^{\uparrow}}$, scopedClosed)

Scope closed: This chapter closes the finite classifier surface for philosophical, mathematical, and host smuggling requests through the displayed refusal-or-gate rows only.

Formal verification: FormalStatus(NameCert$_{\text{SmugglingPatternClassifier}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SmugglingPatternClassifierUp.SmugglingPatternClassifierTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close public export, bridge checking, quotient construction, host primitive admission, global truth, or automatic certificate generation for SmugglingPatternClassifier$^{\uparrow}$.

Upgrade path: The next step is an axiom-purity audit and any standard bridge needed by downstream public consumers.

Chapter 1330

A Concrete Naming Certificate for NoAbsoluteBridgeAudit$^{\uparrow}$

The NoAbsoluteBridgeAudit$^{\uparrow}$ packet realizes the no-absolute-bridge direction of the standard-bridge discipline inside the auditable-generativity programme of chapter 9462. It sits beside the standard bridge audit packet of chapter 1217, the host primitive leakage boundary of chapter 1157, the registry consistency audit of chapter 1223, the finite reflection tuple of chapter 993, and the Ground-Compiler recognizer flow of chapter 1048. Those siblings expose bridge fields, host-leak refusal, registry agreement, finite reflection images, and generated-recognizer admission; this packet binds them into a host-relative audit row before any consumer may read a bridge as host-free.

Definition 1330.1 (No absolute bridge audit carrier)**.** *A* NoAbsoluteBridgeAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, T, E, D, U, L, R, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the* StandardBridgeAuditPacket$^{\uparrow}$ *row containing encoding, decoding, round-trip, theorem-preservation, no-host-leak, and bridge-ledger data.*
2. *T is the target host row, naming the specific host theory whose equality, quotient, function, number, and type or set semantics are being audited.*
3. *E is the host-relative encoding row, read only relative to T.*
4. *D is the host-relative decoding row, read only relative to T.*
5. *U is the finite unsupported-field ledger listing host fields, primitives, theorem shapes, and classifier reads that the bridge refuses.*
6. *L is the* HostPrimitiveLeakage$^{\uparrow}$ *no-host-leak row attached to the target host audit.*
7. *R is the* RegistryConsistencyAudit$^{\uparrow}$ *row recording agreement between bridge claims, unsupported fields, and public registry status.*
8. *G is the generated-recognizer landing row through* FiniteReflectionTuple$^{\uparrow}$ *and* GroundCompilerRecognizerFlow$^{\uparrow}$.
9. *H records componentwise* hsame *transport for bridge, host, encoding, decoding, unsupported-field, leakage, registry, recognizer, continuation, provenance, and local naming rows.*
10. *C records displayed* Cont *replay for host selection, theorem-preservation lookup, unsupported-field inspection, and no-host-leak readback.*

11. *P is the* Pkg *provenance over* NoAbsoluteBridgeAudit$^{\uparrow}$, StandardBridgeAuditPacket$^{\uparrow}$, HostPrimitiveLeakage$^{\uparrow}$, RegistryConsistencyAudit$^{\uparrow}$, FiniteReflectionTuple$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{NoAbsoluteBridgeAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-free bridge, a bridge route with no target host row, an unsupported host field outside U, a leaked host primitive outside L, a private recognizer landing, or a theorem-preservation claim detached from the displayed host audit.

Theorem 1330.2 (No absolute bridge audit NameCert obligations)**.** *For the carrier of Definition 1330.1, the local* NameCert$_{\text{NoAbsoluteBridgeAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (B, T, E, D, U, L, R, G, H, C, P, N)$, target-host visibility, host-relative encoding and decoding, finite theorem-preservation inspection through B, unsupported-field ledger exactness through U, no-host-leak inspection through L, registry agreement through R, recognizer landing through G, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoAbsoluteBridgeAuditUp.noAbsoluteBridgeAuditChapterTasteGate`

Proof. Project the carrier of Definition 1330.1. The bridge packet B supplies the standard bridge fields, while T, E, D, U make encoding, decoding, and unsupported-field refusal relative to one displayed host.

Use the leakage and registry rows next. The row L blocks host primitives from entering the BEDC public surface, and R forces public bridge claims to agree with the registry status and unsupported-field ledger.

Finally inspect G, H, C, P, N. Recognizer landing is routed through the finite reflection tuple and GroundCompiler flow, transport and replay preserve the same displayed rows, provenance records the packet source, and the local name exports only this host-relative audit. Since no additional coordinate is present, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: A host-independent bridge route that omits the target host row or unsupported-field ledger refutes NoAbsoluteBridgeAuditUp.

Independence witness: standardbridgeauditpacket, hostprimitiveleakage, registryconsistencyaudit, finitereflectiontuple, groundcompilerrecognizerflow: StandardBridgeAuditPacket$^{\uparrow}$ records bridge fields without the target-host and unsupported-field audit as a separate no-absolute row; HostPrimitiveLeakage$^{\uparrow}$ refuses host primitives without bridge theorem-preservation rows; RegistryConsistencyAudit$^{\uparrow}$ records registry agreement without host-relative encoding and decoding; FiniteReflectionTuple$^{\uparrow}$ records finite reflection images without bridge refusal rows; GroundCompilerRecognizerFlow$^{\uparrow}$ records recognizer admission without a target-host bridge audit. None is bijective to NoAbsoluteBridgeAudit$^{\uparrow}$.

The Lean follow-up for this chapter must provide `lean4/BEDC/Derived/NoAbsoluteBridgeAuditUp/TasteGat` with Nontrivial NoAbsoluteBridgeAudit$^{\uparrow}$ and FieldFaithful NoAbsoluteBridgeAudit$^{\uparrow}$ instances for the displayed target-host, unsupported-field, no-host-leak, registry, recognizer, transport, replay, provenance, and naming rows.

Closure status for NameCert$_{\text{NoAbsoluteBridgeAudit}^{\uparrow}}$.

Constructive story: A NoAbsoluteBridgeAudit$^{\uparrow}$ packet is generated from a standard bridge packet, one target host row, host-relative encoding and decoding rows, unsupported-field ledger, no-host-leak row, registry-consistency row, generated-recognizer landing row, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NoAbsoluteBridgeAudit}^{\uparrow}}$, seed)

Scope closed: Definition 1330.1 and Definition 1330.2 bind the finite host-relative bridge audit and its target-host, unsupported-field, no-host-leak, registry, recognizer, transport, replay, provenance, and naming obligations.

Formal verification: FormalStatus(NameCert$_{\mathsf{NoAbsoluteBridgeAudit}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not close Lean verification, bridge checking, all host theories at once, host-free bridge recognition, or theorem preservation outside the displayed target-host audit.
Upgrade path: Obligation closure requires checked carrier admission, target-host visibility, unsupported-field exactness, no-host-leak preservation, registry agreement, recognizer landing, TasteGate nontriviality, and field faithfulness for $B, T, E, D, U, L, R, G, H, C, P, N$.

Chapter 1331

A Concrete Naming Certificate for $\mathsf{ResearchProgrammeRoute}^{\uparrow}$

The $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ packet realizes the research programme and implementation route of section 9462.32 as a composite finite NameCert carrier. It sits after the programme-strength ledger of chapter 1305, the kernel-phase refusal surface of chapter 1288, the verification-failure roadmap of chapter 1289, the physics-programme synthesis of chapter 1326, and the package-export discipline of chapter 1323. These siblings expose strength, phase refusal, verification failure, physical synthesis, and export discipline separately; this carrier binds existing programme surfaces rather than claiming atomic novelty.

Definition 1331.1 (Research programme route carrier)**.** *A* $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, S, K, V, E, O, X, H, C, G, N)$$

with the following displayed rows:

1. *P is the phase-route row, recording the current research phase, admissible next phase, and the displayed route through the certificate programme;*
2. *S is the* $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ *row, recording claim strength, dependency status, theory and verification status, bridge status, refusal, and local naming;*
3. *K is the* $\mathsf{KernelPhaseRefusal}^{\uparrow}$ *row, recording the phase-boundary refusal that blocks targets outside the displayed kernel phase;*
4. *V is the* $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ *row, recording checked-report failure, downgrade route, roadmap boundary, cannot-claim data, and upgrade-registry data;*
5. *E is the* $\mathsf{PackageExportDiscipline}^{\uparrow}$ *row, recording package-map coverage, refusal-registry coverage, registry-export coverage, export-gate coverage, formal-target coverage, and non-smuggling coverage;*
6. *O is the operator-review row, recording which displayed programme rows a reviewer may inspect before accepting the route as a public research programme reading;*
7. *X is the exit-criterion row, recording which phase, strength, refusal, failure-roadmap, export, and review rows must be present before the route can leave the current research phase;*
8. *H records componentwise* hsame *transport for P, S, K, V, E, O, X, package provenance, and the local naming row;*

9. *C records displayed* Cont *replay for phase routing, strength inspection, refusal inspection, verification-failure replay, export inspection, operator review, and exit-criterion handoff;*
10. *G is the* Pkg *provenance over* $\mathsf{ResearchProgrammeRoute}^{\uparrow}$, $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$, $\mathsf{KernelPhaseRefusal}^{\uparrow}$, $\mathsf{VerificationFailureRoadmap}^{\uparrow}$, $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$, $\mathsf{PackageExportDiscipline}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeRoute}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are $P, S, K, V, E, O, X, H, C, G, N$. It contains no independent programme-success coordinate, no hidden phase promotion, no review authority outside O, no exit criterion outside X, and no export reading outside the displayed package row.

Theorem 1331.2 (Research programme route NameCert obligations)**.** *For the carrier of Definition 1331.1, the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for $R = (P, S, K, V, E, O, X, H, C, G, N)$, phase-route exposure through P, strength-ledger exposure through S, kernel-refusal exposure through K, verification-failure exposure through V, package-export exposure through E, operator-review exposure through O, exit-criterion exposure through X, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through G, and local naming through N.*

Proof. Project the finite packet. The carrier inventory displays $P, S, K, V, E, O, X, H, C, G, N$ before the local name can be read.

Read the sibling rows in their programme order. The strength ledger supplies S, the kernel-phase refusal packet supplies K, the verification-failure roadmap supplies V, the physics-programme synthesis supplies the physical consumer named by the provenance row, and the package-export discipline supplies E.

Finally inspect the local rows. The operator-review row O and exit-criterion row X are part of the same finite packet, while H, C, G, N transport, replay, source, and name the displayed data. Since the carrier has no further coordinate, these rows exhaust the local NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.ResearchProgrammeRouteUp.ResearchProgrammeRouteTasteGate single carrier alignment`

Theorem 1331.3 (Research programme route phase refusal)**.** *Every accepted* $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ *packet reads the phase-route row P together with the kernel-refusal row K before any programme phase can be treated as admissible. A target blocked by K cannot be exported by strengthening S, by citing the roadmap V, by packaging through E, or by appealing to operator review O.*

Proof. Begin with the carrier rows. The route row P and refusal row K are separate displayed coordinates, so phase admissibility is not read from strength, roadmap, export, or review data alone.

Apply the kernel-phase refusal sibling. Its accepted packet records the current phase, attempted target, dependency blocker, refusal verdict, transport, continuation, provenance, and local name. This row remains visible inside $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ as K.

The rows S, V, E, O can report strength, failure, export, and review only for the same displayed packet. They do not contain a second phase-admission coordinate. Hence a target refused by K remains outside the accepted route. □

Theorem 1331.4 (Research programme route review boundary)**.** *For an accepted* $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ *carrier, the operator-review row* O *may inspect only the displayed rows* $P, S, K, V, E, X, H, C, G, N$*. It cannot introduce an unlisted proof obligation, erase a verification-failure row, promote a strength row, bypass a package-export blocker, or replace the exit criterion by reviewer preference.*

Proof. Project the review coordinate. The row O is one component of $R = (P, S, K, V, E, O, X, H, C, G, N)$, so its authority is bounded by the same packet that it inspects.

Use the sibling constraints. $\mathsf{ProgrammeStrengthLedger}^{\uparrow}$ keeps strength claims tied to displayed dependency and status rows, $\mathsf{VerificationFailureRoadmap}^{\uparrow}$ keeps failure and downgrade data visible, and $\mathsf{PackageExportDiscipline}^{\uparrow}$ keeps export through its package rows.

It remains to compare O with X. The exit-criterion row is a separate displayed coordinate, not a conclusion of review preference. Therefore operator review cannot erase, promote, bypass, or replace any listed programme boundary. □

Theorem 1331.5 (Research programme route nonescape)**.** *Every accepted* $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ *consumer receives phase route, strength ledger, kernel refusal, verification-failure roadmap, package export, operator review, exit criterion, transport, replay, provenance, and local naming as one displayed packet. It cannot read the composite route as completed physics, completed verification, bridge-checked closure, package export without blockers, or a reviewer-authorized phase exit unless the corresponding row is present in* R*.*

Proof. Start from the NameCert obligations. They expose all rows $P, S, K, V, E, O, X, H, C, G, N$ and no additional programme-success row.

Read the composite dependency. $\mathsf{PhysicsProgrammeSynthesis}^{\uparrow}$ is named through provenance as the physical consumer, while the strength, refusal, roadmap, and export siblings preserve their own rows inside the packet. The route therefore binds existing programme surfaces rather than collapsing them into one atomic certificate.

The transport, replay, provenance, and local naming rows preserve the same display. A consumer that claims completed physics, completed verification, bridge-checked closure, unblocked package export, or review-authorized phase exit without the corresponding row has left the accepted $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ carrier. □

Falsifiable prediction: If a ResearchProgrammeRouteUp packet remains accepted after deleting its kernel-refusal row, verification-failure row, or exit-criterion row, then the route carrier is refuted.

Independence witness: programme-strength-ledger, kernel-phase-refusal, verification-failure-roadmap, physics-programme-synthesis, package-export-discipline: none is bijective with $\mathsf{ResearchProgrammeRoute}$ because the composite packet preserves phase, strength, refusal, verification-failure, export, review, exit, transport, replay, provenance, and local NameCert rows together.

The Lean follow-up requires $\mathsf{Nontrivial}\ \mathsf{ResearchProgrammeRoute}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ResearchProgrammeRoute}^{\uparrow}$ instances in `lean4/BEDC/Derived/ResearchProgrammeRouteUp/TasteGate.lean`. The field-faithfulness witness must keep $P, S, K, V, E, O, X, H, C, G, N$ as distinct rows before a checked boundary can cite `BEDC.Derived.ResearchProgrammeRouteUp.taste_gate`.

Closure status for $\mathsf{NameCert}_{\mathsf{ResearchProgrammeRoute}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ResearchProgrammeRoute}^{\uparrow}$ packet is generated from phase, strength, kernel, verification, export, review, exit, componentwise hsame transport, displayed Cont continuation, gate, and local naming rows. The classifier accepts a programme route only as this visible carrier tuple, so no hidden phase promotion, review authority, exit criterion, or package export can be read outside the displayed rows. The ledger story is route governance: a programme moves by finite carrier coordinates whose phase, review, and exit surfaces remain auditable, while refusals and gates record why a path stops or stays below a stronger claim.

Theory closure: TheoryGate(NameCert$_{\mathsf{ResearchProgrammeRoute}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition, NameCert-obligation, phase-refusal, review-boundary, nonescape, independence witness, Lean-follow-up boundary, and existing closure annotations for ResearchProgrammeRoute$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{ResearchProgrammeRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim independent programme success, hidden phase promotion, review authority outside the review row, exit criteria outside the exit row, or export outside the package row.

Upgrade path: To reach obligationClosure, the programme-route carrier and phase, review, and export boundary theorems must be checked over finite rows.

Chapter 1332

A Concrete Naming Certificate for $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$

The $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$ packet realizes the research-programme route of section 9462.32 as a finite audit loop. It sits after the writing discipline of chapter 1227, the registry consistency audit of chapter 1223, the machine-readable audit interface of chapter 1232, the package export discipline of chapter 1323, and the programme-strength gradient ledger of chapter 1314. Those packets regulate public prose, registry agreement, machine-readable export, package-level release, and strength reporting; this packet binds them into one loop before a research programme may treat a proposal as accepted programme motion.

Definition 1332.1 (Research-programme audit-loop carrier)**.** *A* $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, G, F, K, A, X, J, T, P, N)$$

with the following displayed rows:

1. *S is the phase row, recording the current programme phase and the admitted transition boundary from* $\mathsf{PhaseGovernedCertificateRoute}^{\uparrow}$*;*
2. *Q is the proposal-and-revise row, recording proposal intake, review acceptance, refusal, and permitted rewrite routes;*
3. *G is the gate row, recording the public gate that must be passed before a proposal is read as programme motion;*
4. *F is the failure-feedback row, recording typed failure, refusal, and retry data without promoting failure into theorem evidence;*
5. *K is the calibration row, recording how reported strength is read against* $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$*;*
6. *A is the audit-axis row, recording separate theory, verification, bridge, and machine-interface axes;*
7. *X is the label-claim row, recording claim labels, registry entries, and their consistency checks;*
8. *J is the trace row, recording machine-readable audit traces and replay data;*
9. *T records componentwise* hsame *transport for all displayed rows;*

10. *P is the* Pkg *provenance over* $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$, $\mathsf{AuditWritingDiscipline}^{\uparrow}$, $\mathsf{RegistryConsistencyAudit}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{PackageExportDiscipline}^{\uparrow}$, $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}$ *row.*

The carrier has no coordinate for private programme approval, unregistered label motion, hidden machine evidence, host-side gate passage, or strength promotion outside the displayed audit loop.

Theorem 1332.2 (Research-programme audit-loop obligation surface)**.** *For the carrier of Definition 1332.1, the local* $\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}$ *surface consists exactly of phase admission through S, proposal and review motion through Q, gate passage through G, failure-feedback routing through F, calibration through K, axis separation through A, label-claim consistency through X, trace readback through J, componentwise transport through T, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ResearchProgrammeAuditLoopUp.ResearchProgrammeAuditLoopTasteGate single carrier alignment`

Proof. Project the finite packet of Definition 1332.1. Its displayed rows are precisely $S, Q, G, F, K, A, X, J, T, P, N$.

The phase, proposal, gate, failure-feedback, calibration, axis, label-claim, and trace rows are the programme-facing obligations. They respectively consume the phase-governed route, review acceptance, writing discipline, failure record, strength gradient, audit axes, registry consistency, and machine-readable audit interface named in the carrier.

The support rows T, P, N transport, source, and locally name the same displayed packet. Since no further coordinate is present, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}$ surface. □

Theorem 1332.3 (Research-programme audit-loop nonescape)**.** *Every accepted* $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$ *loop exposes phase, proposal-and-revise, gate, failure-feedback, calibration, audit-axis, label-claim, trace, provenance, and local* NameCert *rows. It cannot export accepted programme motion through private approval, prose emphasis, unregistered labels, hidden trace evidence, collapsed theory-and-verification status, or strength promotion outside those rows.*

Proof. Read any accepted loop through the carrier. The rows S, Q, G, F, K, A, X, J determine the programme phase, proposal route, admission gate, failure feedback, calibration, axis separation, label claim, and trace readback before the loop is public.

The writing, registry, interface, export, and strength-gradient packet dependencies named in P supply the sibling surfaces that each row consumes. The transport row T and local naming row N keep the loop on that finite packet.

The excluded readings are not coordinates of the carrier. A consumer that asks for them has left the displayed $\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}$ surface rather than reading an accepted research-programme audit loop. □

Falsifiable prediction: Any accepted ResearchProgrammeAuditLoopUp loop must expose phase, proposal/revise, gate, failure-feedback, calibration, axis, label-claim, trace, provenance, and local NameCert rows.

Independence witness: audit-writing-discipline, registry-consistency-audit, machine-readable-audit-interface, package-export-discipline, programme-strength-gradient-ledger: none is bijective

with $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$ because the audit loop simultaneously carries phase governance, proposal review, gate passage, failure feedback, calibration, axis separation, label-claim consistency, machine-readable trace, provenance, and local NameCert rows.

Lean TasteGate formalization must provide $\mathsf{Nontrivial\ ResearchProgrammeAuditLoop}^{\uparrow}$ and FieldFaithful ResearchP instances in `lean4/BEDC/Derived/ResearchProgrammeAuditLoopUp/TasteGate.lean`. The target `BEDC.Derived.ResearchProgrammeAuditLoopUp.taste_gate` must preserve $S, Q, G, F, K, A, X, J, T, P, N$ as distinct rows before a checked paper marker may cite it.

Closure status for $\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}$.

Constructive story: A $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$ packet is generated from research-programme route rows, review rows, strength-ledger rows, machine-interface rows, export-gate rows, loop-state rows, refusal rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts audit-loop motion only when the route, review, machine, export, strength, and refusal coordinates are displayed together. The ledger story is repeated accountability without private approval: each loop pass is generated from finite BEDC rows and replays its gate and refusal evidence, while label motion, host-side gate passage, hidden machine evidence, and strength promotion outside the audit loop have no coordinate to project.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, obligation-surface theorem, nonescape theorem, independence witness, Lean-follow-up boundary, and existing closure annotations for $\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResearchProgrammeAuditLoop}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim private programme approval, unregistered label motion, hidden machine evidence, host-side gate passage, or strength promotion outside the displayed loop.

Upgrade path: To reach obligationClosure, the audit-loop carrier and nonescape theorem must be checked as finite loop-state transport obligations.

Chapter 1333

A Concrete Naming Certificate for $\mathsf{AuditAxisSeparation}^{\uparrow}$

The $\mathsf{AuditAxisSeparation}^{\uparrow}$ packet realizes the two-axis discipline of section 9462.30 as a concrete finite NameCert surface. It sits beside the writing discipline of chapter 1227, the closure-status reference of chapter 1265, the registry consistency audit of chapter 1223, the registry export surface of chapter 1216, and the registry export consistency gate of chapter 1224. Those siblings expose legal audit prose, status references, agreement checks, public export, and consistency gates; this carrier binds the theory-closure and formal-verification rows together without allowing either row to determine the other.

Definition 1333.1 (Audit axis separation carrier)**.** *An* $\mathsf{AuditAxisSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, F, L, B, R, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the theory-closure row, recording the paper-side grade and its closed scope.*
2. *F is the formal-status row, recording the Lean-side verification grade without deriving it from T.*
3. *L is the Lean-target row, recording the canonical machine target when one is displayed and remaining empty when the packet is unformalized.*
4. *B is the bridge-status row, recording whether no bridge, a paper bridge, or a checked bridge is available.*
5. *R is the refusal and not-claimed row, recording exactly which readings the packet refuses to export.*
6. *X is the registry-export row, recording the public export surface that may read the status packet.*
7. *H records componentwise* hsame *transport for theory closure, formal status, Lean target, bridge status, refusal, registry export, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from registry export and closure-status consumers through the same two-axis packet.*
9. *P is the* Pkg *provenance over* $\mathsf{AuditAxisSeparation}^{\uparrow}$, $\mathsf{AuditWritingDiscipline}^{\uparrow}$, $\mathsf{ClosureStatusReference}^{\uparrow}$, $\mathsf{RegistryConsistencyAudit}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{AuditAxisSeparation}^{\uparrow}}$ *row.*
The carrier has no coordinate for a single status row, for deriving F from T, for deriving T from F, for erasing the Lean-target or bridge row, or for exporting a claim outside the refusal and registry rows.

Lean4 ✓ `BEDC.Derived.AuditAxisSeparationUp.AuditAxisSeparationTasteGate single carrier alignment`

Theorem 1333.2 (Audit axis separation NameCert obligations)**.** *For the carrier of Definition 1333.1, the local* $\mathsf{NameCert}_{\mathsf{AuditAxisSeparation}^{\uparrow}}$ *surface consists exactly of carrier admission for $A = (T, F, L, B, R, X, H, C, P, N$ theory-closure exposure through T, formal-status exposure through F, Lean-target visibility through L, bridge-status visibility through B, refusal and not-claimed exactness through R, registry-export exposure through X, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the finite packet. The rows $T, F, L, B, R, X, H, C, P, N$ are the only coordinates available before the local name can be read.

Read the two status axes first. The row T records paper closure, while the row F records formal verification. The Lean-target row L and bridge-status row B are separate displays, so neither status row can silently absorb machine-target or bridge data.

Finally inspect the audit and registry rows. The refusal row R blocks overstrong readings, X supplies the public export surface, and H, C, P, N transport, replay, source, and name the same displayed packet. These rows exhaust the local NameCert obligations. □

Theorem 1333.3 (Audit axis separation noncollapse)**.** *In every accepted* $\mathsf{AuditAxisSeparation}^{\uparrow}$ *packet, theory closure is read only from T and formal verification is read only from F. Registry export may display the pair, compare the pair, and refuse claims that omit either row, but no accepted consumer can replace the pair by one collapsed status coordinate or infer either row from the other.*

Proof. Use the carrier inventory of Definition 1333.1. The two rows T and F are distinct coordinates, and the carrier contains no function row from one status coordinate to the other.

Apply Definition 1333.2. The NameCert surface exposes both rows before any Lean target, bridge status, refusal surface, registry export, transport, replay, provenance, or local name can be consumed.

The registry and continuation rows preserve the displayed packet rather than computing a status grade. Hence a consumer may jointly report the two axes, but a collapsed status coordinate or cross-axis inference is outside the accepted $\mathsf{AuditAxisSeparation}^{\uparrow}$ carrier. □

Falsifiable prediction: A packet with only a theory-closure row or only a formal-status row is not an accepted $\mathsf{AuditAxisSeparation}^{\uparrow}$ carrier.

Independence witness: audit-writing-discipline, closure-status-reference, registry-consistency-audit, registry-export-surface, registry-export-consistency-gate: none is bijective to $\mathsf{AuditAxisSeparation}^{\uparrow}$ because none carries the same paired noncollapsing theory/formal axis rows together with Lean-target, bridge-status, refusal, registry-export, transport, replay, provenance, and local NameCert rows.

The Lean follow-up file `lean4/BEDC/Derived/AuditAxisSeparationUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{AuditAxisSeparation}^{\uparrow}$ and FieldFaithful $\mathsf{AuditAxisSeparation}^{\uparrow}$ instances. The field-faithfulness witness must preserve $T, F, L, B, R, X, H, C, P, N$ as separate rows before the canonical alignment target may be read as checked.

Closure status for $\mathsf{NameCert}_{\mathsf{AuditAxisSeparation}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditAxisSeparation}^{\uparrow}$ packet is generated from theory-closure, formal-status, Lean-target, bridge-status, refusal, registry-export, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditAxisSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1333.1, Definition 1333.2, and Definition 1333.3 bind the two-axis audit packet and its noncollapse rule as a finite NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditAxisSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, registry correctness beyond the displayed export row, or any derivation between theory closure and formal verification.

Upgrade path: Lean closure requires Nontrivial AuditAxisSeparationUp and FieldFaithful AuditAxisSeparationUp instances for $T, F, L, B, R, X, H, C, P, N$.

Chapter 1334

A Concrete Naming Certificate for $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$

The $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ packet realizes the definition, theorem, and proof disciplines of chapter 9462 as one finite NameCert carrier. It sits beside the proof-trace audit witness of chapter 1219, the writing-item audit carrier of chapter 1235, the writing-item audit packet of chapter 1222, and the allowed proof audit packet of chapter 1294. Those siblings expose proof traces, writing legality, export packet shape, and proof admissibility separately; this carrier binds definition discipline, theorem discipline, proof route, scope, refusal, two-axis status, transport, replay, provenance, and local naming as one audit object.

Definition 1334.1 (Definition theorem proof audit carrier)**.** *A* $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, T, R, S, F, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the definition-discipline row, recording carrier, constructor or generator, classifier, ledger, scope, forbidden readings, and status for the definition site;*
2. *T is the theorem-discipline row, recording the claim text, explicit carrier and certificate assumptions, exact scope, conclusion, and audit line for the theorem site;*
3. *R is the proof-route row, recording the finite witness chain from source witness through classifier witness, transport or exactness witness, ledger witness, stability witness, and conclusion;*
4. *S is the scope row, fixing the local definition, theorem, proof, carrier, classifier, and certificate surface over which D, T, R are read;*
5. *F is the forbidden-resource refusal row, recording hidden choice, host equality as BEDC equality, quotient collapse, propositional extensional rewrite, global truth predicate, global clock, unledgered compression, and unscoped universal claim as unavailable proof resources;*
6. *U is the two-axis status row, recording the independent theory-closure and verification-status grades for the same site without deriving either axis from the other;*
7. *H records componentwise* hsame *transport for $D, T, R, S, F, U, C, P, N$;*
8. *C records displayed* Cont *replay for reading the definition discipline, theorem discipline, proof route, scope, refusal, and two-axis status rows in the same packet;*

9. *P is the* Pkg *provenance over chapter 9462, chapter 1219, chapter 1235, chapter 1222, chapter 1294,* BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DefinitionTheoremProofAudit}\uparrow}$ *row.*

The carrier has no coordinate for a theorem claim detached from proof route, a proof route detached from scope, a status row detached from the two-axis ledger, or a definition reading that omits its forbidden readings.

Lean4 (definition) `BEDC.Derived.DefinitionTheoremProofAuditUp.DefinitionTheoremProofAuditUp`
Lean4 ✓ `BEDC.Derived.DefinitionTheoremProofAuditUp.DefinitionTheoremProofAuditUp single carrier alignment`

Theorem 1334.2 (Definition theorem proof audit NameCert obligations)**.** *For the carrier of Definition 1334.1, the local* $\mathsf{NameCert}_{\mathsf{DefinitionTheoremProofAudit}\uparrow}$ *surface consists exactly of definition-discipline visibility through D, theorem-discipline visibility through T, proof-route visibility through R, scope visibility through S, forbidden-resource refusal through F, two-axis status separation through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier $A = (D, T, R, S, F, U, H, C, P, N)$. The rows D, T, R, S, F, U are the audit data coordinates, so the packet first exposes the definition discipline, theorem discipline, proof route, scope, refusal, and two-axis status before any consumer can read the site as accepted.

Read the structural rows next. Transport through H preserves the displayed coordinates, replay through C keeps the consumer inside the same finite route, provenance through P records the sibling audit sources, and N names the packet.

Since the carrier contains no further audit coordinate, the listed rows exhaust the local NameCert surface. A consumer that omits one row has only a projection of the packet rather than the accepted triadic audit object. □

Lean4 ✓ `BEDC.Derived.DefinitionTheoremProofAuditUp.DefinitionTheoremProofAuditCarrier namecert obligations`

Theorem 1334.3 (Definition theorem proof audit axis and proof route)**.** *Every accepted* DefinitionTheoremProofAu *packet reads theorem claim text through T only together with the proof-route row R and the two-axis status row U. The proof route cannot raise the verification grade by itself, and the verification grade cannot supply a missing proof-route row.*

Proof. Start with the theorem row T. It records claim text, assumptions, scope, conclusion, and audit line, but it is not accepted as a BEDC theorem site until the proof route and status rows are read in the same carrier.

The proof route R supplies the finite witness chain. It certifies how the theorem text is supported inside the scope S, but it does not determine the independent verification coordinate in U.

Conversely, the status row U records theory closure and verification status as two separate grades. It reports the status of the same site; it does not contain a witness-chain coordinate and therefore cannot replace R.

Thus any accepted theorem read factors through T, R, U together, with S, F, H, C, P, N preserving scope, refusal, transport, replay, provenance, and local naming. □

Lean4 ✓ `BEDC.Derived.DefinitionTheoremProofAuditUp.DefinitionTheoremProofAuditAxisAndProofRoute`

Theorem 1334.4 (Definition theorem proof audit non-escape)**.** *An audit packet that supplies definition text or theorem claim text while omitting the proof-route row R, the scope row S, the forbidden-resource refusal row F, or the two-axis status row U is rejected by the* $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ *classifier. In particular, proof-trace auditing, writing-item auditing, writing-packet export, allowed-proof auditing, and audit-system consumption cannot turn a partial prose or theorem row into an accepted definition-theorem-proof audit packet.*

Proof. Use the NameCert obligation surface of Definition 1334.2. Accepted packets expose $D, T, R, S, F, U, H, C, P,$ as one finite carrier.

If R is absent, the theorem row has no finite witness chain. If S is absent, neither the definition nor the theorem has a fixed carrier, classifier, or certificate surface. If F is absent, the packet has no displayed refusal of hidden choice, quotient collapse, host equality, or unscoped universal claim. If U is absent, the audit site no longer records theory closure and verification status as independent axes.

The sibling packets named in the statement may audit proof traces, writing legality, packet export, proof admissibility, or claim ledgers. They preserve their own surfaces; they do not add the missing row to this carrier. Therefore the partial read is refused rather than accepted as a $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ packet. □

Falsifiable prediction: A packet that supplies theorem claim text without the proof-route row is rejected by the audit classifier.

Independence witness: prooftraceauditwitness, writingitemaudit, writingitemauditpacket, allowedproofaudit, auditsystem: none is bijective to $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ because this carrier simultaneously binds definition discipline, theorem discipline, proof route, scope, forbidden-resource refusal, two-axis status, transport, replay, provenance, and local naming.

The Lean follow-up file `lean4/BEDC/Derived/DefinitionTheoremProofAuditUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ instances preserving $D, T, R, S, F, U, H, C, P, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DefinitionTheoremProofAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ packet is generated from definition rows, theorem-claim rows, proof-witness rows, proof-route rows, scope rows, status rows, forbidden-reading rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts a definition-theorem-proof unit only when the theorem claim remains attached to its proof route and scope, and when status is read through the same two-axis ledger. The ledger story is triadic audit: definition, theorem, and proof are generated as separate visible coordinates that can be transported and replayed together, while detached claims, detached routes, and status-only reports stay outside the carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DefinitionTheoremProofAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, axis-and-proof-route, non-escape, independence witness, and Lean-follow-up boundary for $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DefinitionTheoremProofAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim theorem statements detached from proof routes, proof routes detached from scope, status rows detached from the two-axis ledger, or definition reads omitting forbidden readings.

Upgrade path: To reach obligationClosure, the triadic audit carrier and proof-route boundary must be encoded as checked finite-row obligations.

Chapter 1335

A Concrete Naming Certificate for ModulusIndexedDiagonalReadback$^{\uparrow}$

The ModulusIndexedDiagonalReadback$^{\uparrow}$ packet realizes the diagonal representative of Definition 9238.3 as a finite selector certificate. It sits after the diagonal-limit representative surface of chapter 1205, the regular-Cauchy diagonal route of chapter 372, and the finite readback carriers StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$. The packet binds one modulus request to one selected window, one dyadic tolerance ledger, one regular readback, and one Real seal before any Real-facing diagonal read becomes public.

Definition 1335.1 (Regular-Cauchy diagonal selector carrier)**.** *A* ModulusIndexedDiagonalReadback$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus-index request row, naming the precision request n and the selected modulus value $\mu(1/(n+1))$ used by the diagonal representative.*
2. *S is the* StreamName$^{\uparrow}$ *window row that records the finite observation window selected from that modulus request.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row that reads the selected regular rational sequence at the displayed window.*
4. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger that records the dyadic comparison budget attached to the selected diagonal value.*
5. *E is the* Real$^{\uparrow}$ *seal row reached only after M, S, R, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus request, stream window, regular readback, dyadic tolerance, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the modulus request through the selected window, regular readback, dyadic tolerance ledger, and Real seal.*
8. *P records* Pkg *provenance over* ModulusIndexedDiagonalReadback$^{\uparrow}$, DiagonalLimitRepresentative$^{\uparrow}$, RegularCauchyDiagonal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose visible diagonal readback surface is $M, S, R, Q, E, H, C, P, N$. It has no coordinate for quotient streams, a selected limit representative outside the packet, ambient completeness, countable choice, host real equality, or an undisplayed diagonal selector.

Theorem 1335.2 (Regular-Cauchy diagonal selector NameCert obligations)**.** *For every accepted* $\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}$ *carrier $D = (M, S, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusIndexedDiagonalRe}}$ *surface consists exactly of carrier admission, modulus-index exposure through M, StreamName window exposure through S, RegSeqRat readback through R, DyadicRatCore tolerance exposure through Q, RealUp seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1335.1. It displays one modulus request, one selected stream window, one regular rational readback, one dyadic tolerance ledger, and one Real seal row.

The continuation route orders the readback. The modulus request M selects the window S, the selected window supplies the regular readback R, and the dyadic ledger Q records the tolerance budget before the Real seal E is consumed.

The support rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier lists no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or hidden-selector coordinate, these rows exhaust the local NameCert obligations. □

```
Lean4 ✓  BEDC.Derived.ModulusIndexedDiagonalReadbackUp.ModulusIndexedDiagonalReadbackTasteGate
single carrier alignment
```

Theorem 1335.3 (Regular-Cauchy diagonal selector terminal exactness)**.** *Every accepted Real-facing diagonal read from a* $\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}$ *packet factors through the finite route*

$$M, S, R, Q, E, H, C, P, N.$$

The terminal read is exact at the displayed modulus index: it exposes the same modulus request, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row before any consumer may treat the diagonal value as public. No quotient stream, selected limit, host real equality, choice row, ambient completeness witness, or off-packet selector can replace that route.*

Proof. Start with the NameCert obligations of Definition 1335.2. They place the modulus request, stream window, regular readback, dyadic tolerance ledger, and Real seal on one displayed carrier before a consumer may read the diagonal value.

Next compare the packet with the diagonal representative. The representative of Definition 9238.3 chooses the diagonal value by the modulus index and the selected sequence window; the carrier records precisely those two selector rows as M and S and then records the regular readback and tolerance ledger as R and Q.

Finally read the terminal seal through E, H, C, P, N. These rows preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Because no other selector or completion coordinate appears in the carrier, every public Real-facing diagonal read factors through the displayed finite route. □

Falsifiable prediction: Every accepted ModulusIndexedDiagonalReadbackUp consumer exposes the modulus request, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, hsame, Cont, Pkg, and NameCert rows before any Real-facing diagonal read is public.

Independence witness: regularcauchydiagonal, diagonallimitrepresentative, streamname, regseqrat, dyadicratcore: no listed sibling is bijective because each lacks the single modulus-indexed selector packet tying the request index, selected window, dyadic tolerance, regular readback, and Real seal into one local NameCert row.

The future Lean file `lean4/BEDC/Derived/ModulusIndexedDiagonalReadbackUp/TasteGate.lean` must provide `Nontrivial ModulusIndexedDiagonalReadbackUp` and `FieldFaithful ModulusIndexedDiagonal` instances. Those instances preserve $M, S, R, Q, E, H, C, P, N$ as distinct displayed rows before any checked target may cite `BEDC.Derived.ModulusIndexedDiagonalReadbackUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}$ packet is generated from a modulus row, schedule row, regular-readback row, dyadic-tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming around one diagonal selector. The classifier accepts diagonal readback only when the modulus indexes the displayed schedule and the readback and seal rows are downstream of that same finite route. The ledger story is choice-free diagonalization: every terminal read is generated from finite modulus-indexed data already present in the BHist carrier, so no quotient stream, selected representative, countable choice, or hidden diagonal selector can be projected.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, terminal-exactness, Lean-follow-up boundary, and existing closure annotations for $\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusIndexedDiagonalReadback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient streams, selected limit representatives outside the packet, ambient completeness, countable choice, host real equality, or undisplayed diagonal selectors.

Upgrade path: To reach obligationClosure, the modulus-indexed selector and terminal exactness must be checked as finite carrier and readback obligations.

Chapter 1336

A Concrete Naming Certificate for $\mathsf{InscriptionNoveltySocket}^{\uparrow}$

The $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ packet realizes the Type VI inscription-novelty socket of chapter 9394 as a finite NameCert carrier. It sits beside the inscription-event packet of chapter 519, the transcendental supply inscription of chapter 604, the free-will inscription commitment of chapter 592, the supply-socket ledger of chapter 528, and the socket-kind classifier of chapter 1058. Those siblings expose event records, supply boundaries, commitment ledgers, socket accounting, and kind classification; this packet binds the prior boundary, event row, aspect-separation row, Type VI audit gate, transport, replay, provenance, and local naming rows in one carrier.

Definition 1336.1 (InscriptionNoveltySocket carrier)**.** *An* $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, I, R, G, H, C, P, N)$$

with the following displayed rows:

1. *B is the prior-boundary row, listing the records visible before the inscription event is read;*
2. *I is the inscription-event row, consumed through chapter 519;*
3. *R is the non-reduction and aspect-separation row, recording that consumers may compare I with B without extracting a closed records-side reducer from B to I;*
4. *G is the Type VI audit-gate row, read from section 9394.7 and preserving the distinction between proof replay and inscription-event actuality;*
5. *H records componentwise* hsame *transport for B, I, R, G, package provenance, and local naming;*
6. *C records displayed* Cont *replay routes through the prior boundary, event row, aspect-separation row, and audit-gate row;*
7. *P is the* Pkg *provenance over* $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$, $\mathsf{FreeWillInscriptionCommitment}$ $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{InscriptionNoveltySocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden novelty source, deterministic reducer, agency oracle, host psychology predicate, unrecorded proof rule, or second socket-kind classifier.

Lean4 ✓ `BEDC.Derived.InscriptionNoveltySocketUp.InscriptionNoveltySocketTasteGate single carrier alignment`

Theorem 1336.2 (InscriptionNoveltySocket NameCert obligations)**.** *For the carrier of Definition 1336.1, the local* $\mathsf{NameCert}_{\mathsf{InscriptionNoveltySocket}^{\uparrow}}$ *surface has five finite obligations:*

1. *carrier exposure for the tuple* $S = (B, I, R, G, H, C, P, N)$*;*
2. *prior-boundary accountability for the records row* B*;*
3. *inscription-event accountability for the event row* I *and its Type VI site;*
4. *aspect-separation stability for the non-reduction row* R*, audit-gate row* G*, componentwise transport row* H*, and replay row* C*;*
5. *package exactness and non-escape for* P, N*, excluding hidden novelty sources, deterministic reducers, agency oracles, host psychology predicates, unrecorded proof rules, and unlisted socket-kind classifiers.*

Proof. Carrier exposure is projection from the finite packet $S = (B, I, R, G, H, C, P, N)$. Every consumer-facing coordinate is one of these displayed rows.

Prior-boundary accountability reads B, while inscription-event accountability reads I through the inscription-event sibling. These rows make the event public as a finite record site before any Type VI audit gate is consulted.

Aspect-separation stability is carried by R, G, H, C. The row R prevents a consumer from turning comparison between B and I into a closed reducer, while G records the Type VI gate that separates proof replay from inscription-event actuality. The rows H and C transport and replay only that displayed data.

Finally read P and N. The package and local naming rows list the concrete sibling surfaces and do not add a hidden source, reducer, agency oracle, host psychology predicate, proof rule, or socket-kind classifier. Thus the five obligations exhaust the local NameCert surface. □

Theorem 1336.3 (InscriptionNoveltySocket inscription-event route)**.** *Every accepted* $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ *packet reads its inscription-event row* I *through the finite route*

$$B \to I \to R \to G \to H \to C \to P \to N.$$

The route preserves the prior-boundary comparison and the Type VI audit gate, and it exports no hidden novelty source, deterministic reducer from B *to* I*, agency oracle, host psychology predicate, or unrecorded proof rule.*

Proof. Project the accepted carrier as $S = (B, I, R, G, H, C, P, N)$. The only inscription-event coordinate is I, and it is displayed beside the prior-boundary row B before the aspect-separation and audit-gate rows are read.

Read I through the concrete inscription-event sibling named in chapter 519. The row R then records that the event may be compared with B without yielding a closed records-side reducer, and G records the Type VI audit gate from section 9394.7.

The support rows H, C, P, N transport, replay, package, and name the same visible route. Since the carrier contains no coordinate for a hidden novelty source, deterministic reducer, agency oracle, host psychology predicate, or unrecorded proof rule, the inscription-event route cannot export any of those objects. □

Theorem 1336.4 (InscriptionNoveltySocket sibling independence)**.** *The* $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{InscriptionEvent}^{\uparrow}$*,* $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$*,* $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*, or* $\mathsf{SocketKindClassifier}^{\uparrow}$*.*

Proof. Compare the displayed rows. $\mathsf{InscriptionEvent}^{\uparrow}$ supplies the event record, $\mathsf{TranscendentalSupplyInscription}^{\uparrow}$ supplies a supply-boundary socket, $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$ supplies a commitment ledger, $\mathsf{SupplySocketLedger}^{\uparrow}$ accounts for requested supply, and $\mathsf{SocketKindClassifier}^{\uparrow}$ classifies the socket kind.

The $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ packet binds B, I, R, G, H, C, P, N together, with R carrying non-reduction and aspect separation and G carrying the Type VI audit gate for the same event. None of the siblings carries that paired prior-boundary, inscription-event, aspect-separation, and Type VI gate surface as one local NameCert packet. A row-preserving bijection would have to preserve those paired rows, so no such bijection exists. □

The Lean follow-up file `lean4/BEDC/Derived/InscriptionNoveltySocketUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial\ InscriptionNoveltySocket}^{\uparrow}$ and $\mathsf{FieldFaithful\ InscriptionNoveltySocket}^{\uparrow}$ instances. The field-faithfulness witness must preserve B, I, R, G, H, C, P, N as distinct rows so prior-boundary accountability, inscription-event routing, aspect separation, Type VI audit gating, transport, replay, provenance, and local naming cannot be merged.

Falsifiable prediction: If an InscriptionNoveltySocketUp packet erases the non-reduction row while keeping the inscription-event row, the route no longer certifies Type VI aspect separation and cannot satisfy the local NameCert obligations.

Independence witness: inscriptionevent, transcendentalsupplyinscription, freewillinscription-commitment, supplysocketledger, socketkindclassifier: none is bijective to $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ because this packet jointly carries prior boundary, inscription event, non-reduction, Type VI audit gate, hsame, Cont, Pkg, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{InscriptionNoveltySocket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{InscriptionNoveltySocket}^{\uparrow}$ packet is generated from a prior-boundary row, an inscription-event row, a non-reduction and aspect-separation row, a Type VI audit gate, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InscriptionNoveltySocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1336.1, Definition 1336.2, Definition 1336.3, and Definition 1336.4 bind the finite Type VI inscription-novelty socket and its NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InscriptionNoveltySocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, hidden novelty sources, deterministic reducers, agency oracles, host psychology predicates, or unrecorded proof rules.

Upgrade path: Obligation closure requires Lean Nontrivial and FieldFaithful TasteGate instances plus checked carrier, aspect-separation, Type VI gate, route, and package rows for B, I, R, G, H, C, P, N.

Chapter 1337

A Concrete Naming Certificate for $\mathsf{AuditFinalityCertificate}^{\uparrow}$

The $\mathsf{AuditFinalityCertificate}^{\uparrow}$ packet realizes the total auditable synthesis of section 9462.31 as a finite finality certificate. It sits beside the firewall of chapter 1296, the two-axis separation packet of chapter 1333, the definition-theorem-proof audit of chapter 1334, and the reality-constrained final-synthesis ledger of chapter 1178. Those siblings expose export refusal, status-axis separation, proof-route auditing, and reality-constrained synthesis ledgers separately; this carrier binds them into one terminal audit packet without claiming host-final truth.

Definition 1337.1 (Audit finality certificate carrier)**.** *An* $\mathsf{AuditFinalityCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, C, F, R, E, S, W, L, H, T, P, N)$$

with the following displayed rows:

1. *K is the claim row, recording the exact finite claim site whose public finality is being certified.*
2. *C is the positive certificate row, recording the accepted certificate data when the claim is admissible.*
3. *F is the failure row, recording the active failure certificate when the claim is defeated.*
4. *R is the refusal row, recording the explicit non-export readings.*
5. *E is the export-grade row, recording which public export surface may read the packet.*
6. *S is the synthesis row, recording the total-synthesis route through the displayed audit and reality-constrained siblings.*
7. *W is the firewall row, read from* $\mathsf{AuditExportFirewall}^{\uparrow}$*.*
8. *L is the final-synthesis ledger row, read from* $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$*.*
9. *H records componentwise* hsame *transport for claim, certificate, failure, refusal, export-grade, synthesis, firewall, ledger, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay for the same finality packet.*
11. *P is the* Pkg *provenance over* $\mathsf{AuditFinalityCertificate}^{\uparrow}$*,* $\mathsf{AuditExportFirewall}^{\uparrow}$*,* $\mathsf{AuditAxisSeparation}^{\uparrow}$*,* $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$*,* $\mathsf{RealityConstrainedFinalSynthesisLedger}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded final truth, erased failure, collapsed theory/formal status, proofless theorem export, firewall bypass, hidden model–reality identity, or host-side completeness.

Lean4 (definition) `BEDC.Derived.AuditFinalityCertificateUp.AuditFinalityCertificateUp`
Lean4 ✓ `BEDC.Derived.AuditFinalityCertificateUp.AuditFinalityCertificateTasteGate single carrier alignment`

Definition 1337.2 (Audit finality certificate classifier)**.** *The classifier for* $\mathsf{AuditFinalityCertificate}^{\uparrow}$ *accepts exactly those packets $A = (K, C, F, R, E, S, W, L, H, T, P, N)$ whose claim row K, certificate row C, failure row F, refusal row R, export-grade row E, synthesis row S, firewall row W, final-synthesis ledger row L, transport row H, replay row T, provenance row P, and local name row N are all displayed in the same finite* BHist *carrier and are preserved by the local* $\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}$ *row. It rejects any packet that omits one of those rows or replaces them by a host-final truth predicate.*

Theorem 1337.3 (Audit finality certificate NameCert obligations)**.** *For the carrier of Definition 1337.1 and classifier of Definition 1337.2, the local* $\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed packet $A = (K, C, F, R, E, S, W, L, H, T, P, N)$;*
2. *classifier exactness for accepting exactly the same displayed packet and rejecting host-final replacements;*
3. *audit-row preservation for claim, positive certificate, failure, refusal, export-grade, synthesis, firewall, and final-synthesis ledger rows;*
4. *structural preservation by componentwise* hsame *transport through H, displayed* Cont *replay through T, and* Pkg *provenance through P;*
5. *local naming through N with no erased failure, collapsed status axis, proofless theorem export, firewall bypass, or model–reality identity row.*

Proof. Project the finite packet. The rows K, C, F, R, E, S, W, L are the only audit-finality data coordinates available before transport, replay, provenance, and local naming are read.

Read the classifier next. It accepts exactly the same displayed rows and rejects host-final replacements, so classifier exactness and audit-row preservation are the same finite surface.

Inspect the sibling rows. The firewall row W supplies the export refusal surface, the axis-separation data is carried through P, the definition-theorem-proof route is carried through P, and the final-synthesis ledger row L supplies the reality-constrained synthesis boundary.

Finally inspect H, T, P, N. These rows preserve, replay, source, and name the same packet, so the certificate cannot add a hidden finality coordinate outside the displayed carrier. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 1337.4 (Audit finality certificate export refusal)**.** *If an* $\mathsf{AuditFinalityCertificate}^{\uparrow}$ *packet has an active failure row F or a refusal row R at the same claim site K, then the export-grade row E may report the refusal but cannot turn the packet into an admitted final claim. The firewall row W, axis-separation provenance, proof-route provenance, and final-synthesis ledger row L remain visible as audit evidence rather than becoming a host-final truth predicate.*

Proof. Begin with Definition 1337.1. The claim, certificate, failure, refusal, export-grade, synthesis, firewall, and ledger rows are separate displayed coordinates.

Apply Definition 1337.3. An accepted read must preserve F and R before E can be consumed, and the firewall row W records the same refusal-sensitive export boundary.

The provenance row P keeps the axis-separation and definition-theorem-proof audit sources attached to the packet. Hence a consumer may report the refusal or the defeated claim, but it has no coordinate that converts refusal, proof-route audit, firewall evidence, or synthesis ledger evidence into host-final truth. □

Falsifiable prediction: A finality packet that omits the failure row, refusal row, firewall row, or final-synthesis ledger row is not an accepted $\mathsf{AuditFinalityCertificate}^{\uparrow}$ carrier.

Independence witness: auditexportfirewall, auditaxisseparation, definitiontheoremproofaudit, realityconstrainedfinalsynthesisledger: none is bijective to $\mathsf{AuditFinalityCertificate}^{\uparrow}$ because this carrier simultaneously binds claim, certificate, failure, refusal, export-grade, synthesis, firewall, ledger, transport, replay, provenance, and local NameCert rows.

The Lean follow-up file `lean4/BEDC/Derived/AuditFinalityCertificateUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{AuditFinalityCertificate}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{AuditFinalityCertificate}^{\uparrow}$ instances preserving $K, C, F, R, E, S, W, L, H, T, P, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditFinalityCertificate}^{\uparrow}$ packet is generated from finite claim, certificate, failure, refusal, export-grade, synthesis, firewall, ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1337.1, Definition 1337.2, Definition 1337.3, and Definition 1337.4 bind the finality certificate, classifier, and export-refusal boundary as a finite NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditFinalityCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not claim host-final truth, erased failure, collapsed theory/formal status, proofless theorem export, firewall bypass, model–reality identity, Lean verification, or bridge checking.

Upgrade path: Lean closure requires Nontrivial AuditFinalityCertificateUp and FieldFaithful AuditFinalityCertificateUp instances for $K, C, F, R, E, S, W, L, H, T, P, N$.

Chapter 1338

A Concrete Naming Certificate for $\mathsf{FourFaceExitClassifier}^{\uparrow}$

The $\mathsf{FourFaceExitClassifier}^{\uparrow}$ packet realizes the regular-Cauchy exit route of section 9238.1 as one finite classifier packet. It sits beside the window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback surface of chapter 301, the real-name seal of chapter 38, and the observation budget discipline of chapter 1164. Those siblings provide the four faces and their budget context; this chapter binds them as a single accepted exit classifier without quotient streams, selected limits, ambient completeness, or bridge rows.

Definition 1338.1 (Four-face exit classifier carrier)**.** *A* $\mathsf{FourFaceExitClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window for the inspected regular-Cauchy read.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and rational readback ledger attached to that window.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed through the same window and tolerance ledger.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S, D, R *have been read on the same packet.*
5. H *records componentwise* hsame *transport for* S, D, R, E *and for the local naming row.*
6. C *records displayed* Cont *replay from window admission through tolerance, readback, real seal, transport, provenance, and local naming.*
7. P *is the* Pkg *provenance over* $\mathsf{FourFaceExitClassifier}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealWindowBudget}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{FourFaceExitClassifier}^{\uparrow}}$ *row.*

The carrier exports no quotient-stream equality, selected limit, host real, ambient completeness principle, or detached bridge row.

Theorem 1338.2 (Four-face exit classifier NameCert obligations)**.** *For the carrier of Definition 1338.1, the local* $\mathsf{NameCert}_{\mathsf{FourFaceExitClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for* $F = (S, D, R, E, H, C, P, N)$*, finite-window exposure through* S*, dyadic tolerance and readback accounting through* D*, regular-rational readback through* R*, real-seal exposure through* E*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.FourFaceExitClassifierUp.TasteGate.FourFaceExitClassifierNameCertObligations`

Proof. Project the packet. The only four face rows are S, D, R, E, and the remaining rows are structural support for transport, replay, provenance, and naming.

The finite window is read through chapter 126. The tolerance ledger is read through chapter 291. The regular-rational readback is read through chapter 301. The real-facing seal is read through chapter 38.

The budget discipline of chapter 1164 controls the observation request, but it is not an extra face of this carrier. Since H, C, P, N preserve and name only the displayed rows, the listed obligations exhaust the local certificate. □

Theorem 1338.3 (Four-face exit classifier stream-readback exactness)**.** *Every accepted* $\mathsf{FourFaceExitClassifier}^{\uparrow}$ *consumer that reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R must first read the same finite* $\mathsf{StreamName}^{\uparrow}$ *window S and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D recorded in the carrier. The readback row cannot be replaced by a detached regular sequence, by a second window, or by a tolerance-free real seal.*

Proof. Start from the obligation surface of Definition 1338.2. It admits R only as the readback row of the displayed carrier $F = (S, D, R, E, H, C, P, N)$.

The window coordinate S supplies the finite inspection site, while D supplies the tolerance and rational arithmetic ledger for that site. Therefore the readback coordinate R is exact only relative to the same S and D rows.

Transport through H and replay through C preserve the componentwise packet. They do not introduce a second window, a detached regular sequence, or a tolerance-free route into the real seal. Hence every accepted read of R remains the stream-readback route displayed in the carrier. □

Theorem 1338.4 (Four-face exit classifier no-completeness escape)**.** *An accepted* $\mathsf{FourFaceExitClassifier}^{\uparrow}$ *packet exports the exit route*

$$S, D, R, E, H, C, P, N$$

and nothing stronger. In particular, a consumer cannot obtain quotient-stream equality, a selected limit, a host real, ambient completeness, or an acceptance bridge from the packet.

Proof. The carrier of Definition 1338.1 lists exactly the four face rows and the structural support rows. None of those coordinates is quotient equality, a selected limit, a host real, a completeness principle, or a bridge row.

Use Definition 1338.3 to keep the regular-rational readback attached to the finite window and dyadic ledger. Use Definition 1338.2 to keep the real seal downstream of those rows and of the displayed support.

Thus every public consumer remains inside the finite four-face route. Requests for quotient streams, selected limits, host completeness, or bridge acceptance leave the accepted carrier rather than being supplied by it. □

Lean4 ✓ `BEDC.Derived.FourFaceExitClassifierUp.FourFaceExitClassifierTasteGate single carrier alignment`

Falsifiable prediction: A packet with the StreamName and RealUp faces but no DyadicRatCore tolerance ledger is not an accepted FourFaceExitClassifierUp carrier.

Independence witness: streamname, dyadicratcore, regseqrat, real, realwindowbudget: no sibling is bijective because StreamNameUp supplies only finite windows, DyadicRatCoreUp supplies tolerance and arithmetic ledgers without regular readback, RegSeqRatUp supplies regular-rational

names without the observation budget, RealUp supplies the seal row without its window and tolerance provenance, and RealWindowBudgetUp schedules observations without itself carrying the four-face packet.

The Lean follow-up must provide Nontrivial FourFaceExitClassifier$^{\uparrow}$ and FieldFaithful FourFaceExitClassifier$^{\uparrow}$ instances in lean4/BEDC/Derived/FourFaceExitClassifierUp/TasteGate.lean. The target BEDC.Derived.FourFaceExitC may be cited only after those instances distinguish the window, tolerance ledger, readback, real seal, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{FourFaceExitClassifier}^{\uparrow}}$.

Constructive story: A FourFaceExitClassifier$^{\uparrow}$ packet is generated from a finite StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier has four faces and reads each face only through the same finite BHist carrier: window admission, tolerance accounting, rational readback, and real-facing seal. The ledger story is exit without completeness: every consumer of the RealUp face must pass through the window and dyadic tolerance rows, while transport preserves the displayed coordinates and cannot create quotient-stream equality, selected limits, host reals, or detached bridge rows.

Theory closure: TheoryGate(NameCert$_{\text{FourFaceExitClassifier}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier definition, NameCert-obligation, stream-readback-exactness, no-completeness-escape, falsifiable prediction, independence witness, Lean-follow-up boundary, and existing closure annotation for FourFaceExitClassifier$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{FourFaceExitClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim quotient-stream equality, selected limits, host reals, ambient completeness principles, or detached bridge rows.

Upgrade path: To reach obligationClosure, the four-face carrier and no-completeness boundary must be encoded with checked face-distinguishing obligations.

Chapter 1339

A Concrete Naming Certificate for $\mathsf{EpistemicOntologyBridge}^{\uparrow}$

The $\mathsf{EpistemicOntologyBridge}^{\uparrow}$ packet realizes the philosophy total-synthesis surface of section 9462.31 as a finite bridge between objecthood, knowledge licensing, observation logic, phenomenology-science interface data, audit export, and local naming. It sits beside the firewall of chapter 1296, the object-knowledge certificate of chapter 1218, the observation-logic boundary of chapter 1295, and the phenomenology-science interface of chapter 1300. Those siblings expose export refusal, object-witness evidence, observation admissibility, and phenomenology-to-science handoff separately; this bridge binds their rows before a public synthesis read may treat ontology and epistemology as one certified surface.

Definition 1339.1 (Epistemic ontology bridge carrier)**.** *An* $\mathsf{EpistemicOntologyBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (O, K, Q, P, A, L, H, C, R, N)$$

with the following displayed rows:

1. *O is the ontology row, recording the finite objecthood and constitution data read from* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$
2. *K is the epistemology row, recording the knowledge-license and witness-admissibility data attached to the same object site.*
3. *Q is the observation-logic row, recording admissible observation claims from* $\mathsf{ObservationLogicBoundary}^{\uparrow}$*.*
4. *P is the phenomenology-science row, recording the interface route from phenomenal framing to public scientific inscription.*
5. *A is the audit-export row, recording the firewall decision from* $\mathsf{AuditExportFirewall}^{\uparrow}$*.*
6. *L is the bridge ledger row, recording that O, K, Q, P, A are read in one public synthesis packet.*
7. *H records componentwise* hsame *transport for the ontology, epistemology, observation-logic, phenomenology-science, audit-export, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for the same bridge packet.*
9. *R is the* Pkg *provenance over* $\mathsf{EpistemicOntologyBridge}^{\uparrow}$*,* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$*,* $\mathsf{ObservationLogicBoundary}^{\uparrow}$*,* $\mathsf{PhenomenologyScienceInterface}^{\uparrow}$*,* $\mathsf{AuditExportFirewall}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for ontology without witness, epistemology without object row, observation outside the observation-logic boundary, phenomenology without science-interface handoff, audit export without firewall decision, or host-level synthesis truth.

Lean4 ✓ `BEDC.Derived.EpistemicOntologyBridgeUp.EpistemicOntologyBridgeTasteGate single carrier alignment`

Definition 1339.2 (Epistemic ontology bridge classifier)**.** *The classifier for* $\mathsf{EpistemicOntologyBridge}^{\uparrow}$ *accepts exactly those packets* $E = (O, K, Q, P, A, L, H, C, R, N)$ *whose ontology row* O*, epistemology row* K*, observation-logic row* Q*, phenomenology-science row* P*, audit-export row* A*, bridge ledger row* L*, transport row* H*, replay row* C*, provenance row* R*, and local name row* N *are all displayed in the same finite* BHist *carrier and preserved by the local* $\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}$ *row. It rejects any packet that omits one of those rows or replaces the bridge by a host-side identification of being, knowing, observing, experiencing, and exporting.*

Theorem 1339.3 (Epistemic ontology bridge NameCert obligations)**.** *For the carrier of Definition 1339.1 and classifier of Definition 1339.2, the local* $\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed packet* $E = (O, K, Q, P, A, L, H, C, R, N)$*;*
2. *classifier exactness for accepting exactly the same displayed packet and rejecting host-side synthesis replacements;*
3. *cross-face preservation for ontology, epistemology, observation-logic, phenomenology-science, audit-export, and bridge-ledger rows;*
4. *structural preservation by componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, and* Pkg *provenance through* R*;*
5. *local naming through* N *with no unlicensed ontology, objectless knowledge, unbounded observation claim, unbridged phenomenology, firewall bypass, or host-level synthesis truth row.*

Proof. Project the carrier of Definition 1339.1. The rows O, K, Q, P, A, L are the only cross-face data coordinates available before transport, replay, provenance, and local naming are read.

Read the classifier of Definition 1339.2. Its accepted packet is the same displayed tuple, so carrier admission and classifier exactness have the same finite boundary.

Inspect the sibling rows. The object-knowledge row supplies ontology and epistemology evidence, the observation-logic row supplies admissible observation form, the phenomenology-science row supplies public interface transport, and the audit-export row supplies the firewall boundary.

Finally inspect H, C, R, N. These rows preserve, replay, source, and name the same packet, so the bridge cannot add a hidden synthesis coordinate outside the displayed carrier. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 1339.4 (Epistemic ontology bridge audit-export boundary)**.** *If any ontology, epistemology, observation-logic, phenomenology-science, audit-export, bridge-ledger, transport, replay, provenance, or local naming row is absent from an* $\mathsf{EpistemicOntologyBridge}^{\uparrow}$ *packet, then the classifier rejects the public export. In particular, the bridge cannot turn an object row alone, a knowledge row alone, an observation row alone, a phenomenology-science handoff alone, or a firewall decision alone into a total philosophy synthesis.*

Proof. Start from Definition 1339.2. Acceptance requires the displayed rows $O, K, Q, P, A, L, H, C, R, N$ in one finite carrier.

Apply Definition 1339.3. The local NameCert obligations preserve each cross-face row before any public export may consume the packet.

If one row is absent, the classifier has no projection for the corresponding face or structural support. The audit-export row A therefore remains a firewall decision inside the packet rather than a replacement for the missing ontology, epistemology, observation-logic, phenomenology-science, ledger, transport, replay, provenance, or local name row. □

Falsifiable prediction: If any ontology, epistemology, observation-logic, phenomenology-science, or audit-export row is absent, the bridge classifier rejects the public export.

Independence witness: auditexportfirewall, objectknowledgecertificate, observationlogicboundary, phenomenologyscienceinterface: each sibling supplies one face or firewall boundary, while EpistemicOntologyBridgeUp binds the cross-face bridge; no bijection preserves all rows.

Lean `TasteGate.lean` for this route must provide `Nontrivial EpistemicOntologyBridgeUp` and `FieldFaithful EpistemicOntologyBridgeUp` instances in `BEDC.Derived.EpistemicOntologyBridgeUp`.

Closure status for $\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}$.

Constructive story: An $\mathsf{EpistemicOntologyBridge}^{\uparrow}$ packet is generated from finite ontology, epistemology, observation-logic, phenomenology-science, audit-export, ledger, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1339.1, Definition 1339.2, Definition 1339.3, and Definition 1339.4 bind the bridge carrier, classifier, NameCert obligations, and export boundary as a finite synthesis surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EpistemicOntologyBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean implementation, checked bridge status, host-level synthesis truth, ontology without witness, objectless knowledge, observation outside the observation-logic boundary, unbridged phenomenology, or firewall bypass.

Upgrade path: The next paper-side step is public export after Lean encoding catches up to the carrier, classifier, NameCert obligations, and audit-export boundary.

Chapter 1340

A Concrete Naming Certificate for $\mathsf{PhysicalRecordInvariant}^{\uparrow}$

The $\mathsf{PhysicalRecordInvariant}^{\uparrow}$ packet realizes the frame-independent physical content of Definition 9462.927 and Definition 9462.928 as finite BEDC NameCert data. It sits between the object-facing packet of chapter 1214, the defeat-facing packet of chapter 1163, and the forward-looking socket of chapter 505. Those siblings provide scientific-object, failure, and causal-commitment surfaces; this chapter isolates the positive invariant carrier consumed when a physical record must remain readable across histories and observer presentations.

Definition 1340.1 (Physical record invariant carrier)**.** *A* $\mathsf{PhysicalRecordInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (R, I, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the physical-record ledger, listing finite history rows, continuation rows, measurement rows, instrument rows, calibration rows, error rows, relation rows, and falsification rows.*
2. *I is the inter-history invariant row whose classifier reads record content only through witnessed sameness between histories.*
3. *O is the observer-symmetry row, requiring each accepted presentation of R and I to be stable under observer-facing reindexing.*
4. *H records componentwise* hsame *transport for R, I, O and for the local naming row.*
5. *C records displayed* Cont *replay from record admission through invariant read, observer symmetry, transport, provenance, and local naming.*
6. *P is the* Pkg *provenance over* $\mathsf{PhysicalRecordInvariant}^{\uparrow}$, $\mathsf{ScientificObject}^{\uparrow}$, $\mathsf{RealityConstrainedFailureSurface}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalRecordInvariant}^{\uparrow}}$ *row.*

The carrier has no coordinate for bare object existence, unledgered measurement fit, one-history physical content, hidden observer frame, final truth export, or selected future observation outside the displayed rows.

Lean4 (definition) `BEDC.Derived.PhysicalRecordInvariantUp.PhysicalRecordInvariantUp`

Theorem 1340.2 (Physical record invariant classifier stability)**.** *The* PhysicalRecordInvariant$^{\uparrow}$ *classifier is stable only under componentwise* hsame *transport of the displayed rows* R, I, O, H, C, P, N*. A consumer may replay the invariant route through* Cont *and cite provenance through* Pkg*, but it must preserve the physical-record ledger, inter-history invariant row, observer-symmetry row, and local* NameCert *row as separate coordinates.*

Proof. Project the carrier of Definition 1340.1. The rows R, I, O are displayed separately: R records physical data, I names the inter-history invariant, and O records observer symmetry.

Transport is then read through H. Since H is componentwise, it preserves the record, invariant, and observer rows rather than identifying them with one ambient physical object.

Replay through C and provenance through P keep the consumer inside the same finite route. Therefore classifier stability holds exactly for the displayed packet R, I, O, H, C, P, N and fails when a consumer erases the observer-symmetry row or collapses the invariant into a single-history record. □

Lean4 ✓ `BEDC.Derived.PhysicalRecordInvariantUp.PhysicalRecordInvariantCarrier classifier stability`

Theorem 1340.3 (Physical record invariant ledger exactness)**.** *Every accepted* PhysicalRecordInvariant$^{\uparrow}$ *packet exposes the physical-record ledger* R *and the inter-history invariant row* I *before any frame-independent physical content is readable. The observer-symmetry row* O *may constrain their presentations, but it cannot replace either the ledger or the invariant.*

Proof. Read the carrier inventory. The ledger R is the only coordinate listing physical records, while I is the only coordinate carrying their inter-history invariant content.

The observer-symmetry row O is a constraint on accepted presentations. It requires stability of the read under observer-facing reindexing, but it does not contain the record ledger and does not itself witness inter-history invariance.

The structural rows H, C, P, N transport, replay, source, and name the same ledger and invariant rows. Hence ledger exactness requires R and I to remain visible before any frame-independent content is exported. □

Theorem 1340.4 (Physical record invariant NameCert surface)**.** *Every accepted* PhysicalRecordInvariant$^{\uparrow}$ *read factors through*

$$R, I, O, H, C, P, N.$$

The consumer receives finite physical records, an inter-history invariant row, observer symmetry, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. It receives no bare scientific object, no defeat-only surface, no forward-GAP causal commitment, no one-history physical content, and no hidden frame.*

Proof. Start from Definition 1340.1. The displayed packet exhausts the local carrier available to consumers of frame-independent physical content.

Apply Definition 1340.2. The classifier preserves the read only when R, I, O, H, C, P, N remain separate under hsame transport and Cont replay.

Apply Definition 1340.3. The physical-record ledger and invariant row must be visible before the content is exported, while observer symmetry constrains presentations without replacing either row.

The sibling chapters locate the boundary. ScientificObject$^{\uparrow}$ in chapter 1214 has object and bridge rows not fixed to inter-history invariants; RealityConstrainedFailureSurface$^{\uparrow}$ in chapter 1163 has defeat rows but no positive invariant carrier; and CausalCommitment$^{\uparrow}$ in chapter 505 has

forward-GAP rows rather than observer-symmetric invariant rows. Thus the accepted read factors through the stated NameCert surface. □

Lean4 ✓ `BEDC.Derived.PhysicalRecordInvariantUp.PhysicalRecordInvariantTasteGate single carrier alignment`

Theorem 1340.5 (Physical record invariant kernel scope)**.** *Every accepted* PhysicalRecordInvariant$^{\uparrow}$ *packet that exports frame-independent record content exposes*

$$R, I, O, H, C, P, N$$

as BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any* ScientificObject$^{\uparrow}$, RealityConstrainedFailureSurface$^{\uparrow}$, *or* CausalCommitment$^{\uparrow}$ *consumer reads it.*

Erasing any one of R, I, O, H, C, P, N leaves the scoped carrier for frame-independent physical content.

Proof. Start from the carrier of Definition 1340.1. The packet displays physical records, inter-history invariance, observer symmetry, transport, replay, provenance, and local naming as separate rows of the same finite BHist carrier.

Use classifier stability from Definition 1340.2. A consumer may transport the record content only componentwise, so the physical-record row, invariant row, observer-symmetry row, and local naming row cannot be collapsed into a bare scientific object.

Use ledger exactness from Definition 1340.3. The physical-record ledger and invariant row must be visible before exported content is read, and observer symmetry constrains the presentation without replacing either row.

Finally apply the NameCert surface theorem Definition 1340.4. It keeps every accepted read on R, I, O, H, C, P, N and refuses the sibling shortcuts: ScientificObject$^{\uparrow}$ cannot supply the invariant row, RealityConstrainedFailureSurface$^{\uparrow}$ cannot supply the positive record carrier, and CausalCommitment$^{\uparrow}$ cannot supply the observer-symmetric readback. Removing any listed row therefore removes the packet from the scoped carrier. □

Theorem 1340.6 (Physical record invariant falsifiable prediction)**.** *Any accepted* PhysicalRecordInvariant$^{\uparrow}$ *packet whose observer-symmetry row O is erased fails the classifier-stability obligation against* RealityConstrainedFailureSurface$^{\uparrow}$ *consumers. Such a packet may still expose records or defeats, but it no longer carries frame-independent physical content.*

Proof. Assume a consumer erases O from the packet $V = (R, I, O, H, C, P, N)$. The remaining projection no longer displays the row that makes presentations observer-symmetric.

Use classifier stability from Definition 1340.2. Stability requires componentwise preservation of R, I, O, H, C, P, N, so deleting O breaks the accepted transport route.

Against RealityConstrainedFailureSurface$^{\uparrow}$ consumers, the failure is visible as a missing positive invariant carrier rather than as a new defeat row. The defeat-facing sibling can record the boundary, but it cannot reconstruct observer symmetry. Therefore the projected packet is rejected for frame-independent physical content. □

Falsifiable prediction: Any accepted PhysicalRecordInvariantUp packet whose observer-symmetry row is erased fails the classifier-stability obligation against RealityConstrainedFailureSurfaceUp consumers.

Independence witness: ScientificObjectUp, RealityConstrainedFailureSurfaceUp, CausalCommitmentUp, ObjectKnowledgeCertificateUp, RealityConstrainedTruthCertUp: none is bijective because ScientificObjectUp has object and bridge rows not fixed to inter-history invariants, RealityConstrainedFailureSurfaceUp has defeat rows but no positive invariant carrier, CausalCommitmentUp has forward-GAP rows rather than observer-symmetric invariant rows, ObjectKnowledgeCertificateUp certifies knowledge export rather than record invariance, and RealityConstrainedTruthCertUp exports truth certification rather than frame-independent record content.

The Lean follow-up path `lean4/BEDC/Derived/PhysicalRecordInvariantUp/TasteGate.lean` provides explicit Nontrivial PhysicalRecordInvariant$^{\uparrow}$ and FieldFaithful PhysicalRecordInvariant$^{\uparrow}$ instances. These instances distinguish physical-record, invariant, observer-symmetry, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{PhysicalRecordInvariant}\uparrow}$.

Constructive story: A PhysicalRecordInvariant$^{\uparrow}$ packet is generated from finite physical-record, inter-history-invariant, observer-symmetry, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{PhysicalRecordInvariant}\uparrow}$, scopedClosed)

Scope closed: Definition 1340.1, Definition 1340.2, Definition 1340.3, Definition 1340.4, Definition 1340.6, the independence witness, the TasteGate marker, and the finite rows R, I, O, H, C, P, N bind the closed scope for frame-independent physical content.

Formal verification: FormalStatus(NameCert$_{\text{PhysicalRecordInvariant}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.PhysicalRecordInvariantUp.PhysicalRecordInvariantTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close final physical truth, object-facing bridge completion, defeat completeness, forward causal commitment, or one-history physical content.

Upgrade path: Mature closure requires downstream bridge checking and consumers that preserve R, I, O, H, C, P, N through the same NameCert surface.

Chapter 1341

A Concrete Naming Certificate for $\mathsf{BareObjectRefusal}^{\uparrow}$

The $\mathsf{BareObjectRefusal}^{\uparrow}$ packet realizes the bare-object refusal direction of section 9462.23 as finite NameCert data. It sits beside the licensed object-knowledge packet of chapter 1218, the description-expansion boundary of chapter 661, the apophatic refusal boundary of chapter 630, and the classifier boundary of chapter 595. Those siblings license object names, regulate descriptions, record apophatic refusal rows, and expose classifier-boundary rows; this packet records the failed five-field object license itself.

The packet is composite over chapter 1218, chapter 661, chapter 630, and chapter 595; it bundles their boundary rows into one failed-license certificate.

Definition 1341.1 (Bare-object refusal carrier)**.** *A* $\mathsf{BareObjectRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (O, M, R, W, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the requested object-name row whose proposed use is tested against the object-knowledge certificate discipline.*
2. *M is the missing-field row, recording which of source, pattern, classifier, stability, and ledger is absent or refused for the requested object license.*
3. *R is the refusal row that prevents the requested name from being exported as an object-layer license while M is nonempty.*
4. *W is the witness-audit row recording that no auditable witness row has been supplied for the requested object read.*
5. *L is the anti-primitive ledger row recording that the refusal cannot be bypassed by treating the name as a bare object primitive.*
6. *H records componentwise* hsame *transport for the requested name, missing-field row, refusal row, witness audit, ledger row, continuation row, package row, and local naming row.*
7. *C records the displayed* Cont *routes by which consumers read the missing fields, refusal, witness audit, and ledger nonescape rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$, $\mathsf{DefiniteDescriptionBoundary}^{\uparrow}$, $\mathsf{NegativeCertificateBoun}$ $\mathsf{ClassifierBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BareObjectRefusal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a licensed object name, witness-free knowledge export, primitive object token, host ontology, or object-layer license detached from the displayed five-field certificate check.

Lean4 ✓ `BEDC.Derived.BareObjectRefusalUp.BareObjectRefusalTasteGate single carrier alignment`

Theorem 1341.2 (Bare-object refusal NameCert obligations)**.** *For the carrier of Definition 1341.1, the local* $\mathsf{NameCert}_{\mathsf{BareObjectRefusal}^{\uparrow}}$ *surface consists of five obligation rows:*

1. *source refusal records the requested object-name row O and the absence of an accepted source field in M;*
2. *pattern refusal records that no displayed pattern row licenses O as an object read;*
3. *classifier refusal records that no accepted classifier row turns O into an object-identity surface;*
4. *stability refusal records that no componentwise transport may convert the missing fields into a stable object license;*
5. *ledger refusal records the anti-primitive row L, witness-audit row W, refusal row R, routes C, provenance P, and local naming row N as the only public export.*

Consequently a public consumer of an accepted $\mathsf{BareObjectRefusal}^{\uparrow}$ *packet receives a refusal certificate for the failed object license, not a weak object license.*

Proof. Carrier admission is projection from the displayed tuple $B = (O, M, R, W, L, H, C, P, N)$. The missing-field row M is the only place where the five object-knowledge fields are named, and it records absence or refusal rather than a licensed object packet.

The first four obligation rows inspect the failed source, pattern, classifier, and stability fields. They show that the requested name has not met the object-knowledge discipline of chapter 1218; description expansion, negative-certificate refusal, and classifier-boundary rows remain sibling surfaces rather than substitutes for the missing object license.

The ledger obligation then closes the export. The witness-audit row W, refusal row R, anti-primitive ledger L, transports H, routes C, provenance P, and local naming row N expose only the failed-license packet. Since no carrier coordinate contains a licensed object name or witness row, the consumer cannot repack the refusal as object knowledge. □

Theorem 1341.3 (Bare-object refusal object-license nonescape)**.** *Every accepted* $\mathsf{BareObjectRefusal}^{\uparrow}$ *packet exports only the requested-name, missing-field, refusal, witness-audit, anti-primitive ledger, transport, route, provenance, and local naming rows*

$$O, M, R, W, L, H, C, P, N.$$

It cannot export an object-layer license unless the read factors through an existing $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ *packet with displayed source, pattern, classifier, stability, ledger, object-name, and witness rows.*

Proof. Start with the NameCert obligations above. They make the missing source, pattern, classifier, stability, and ledger fields part of the refusal surface, so the local packet records a failed object license rather than a partial one.

Use the object-knowledge sibling. chapter 1218 exposes licensed object knowledge only through its object-name, witness, five-field certificate, audit-route, transport, and local naming rows. None of those licensed rows is present in the $\mathsf{BareObjectRefusal}^{\uparrow}$ carrier except as refused or missing data.

Finally apply the anti-primitive ledger row. The ledger L records that a consumer cannot treat O as a primitive object token after the five-field certificate check has failed. Thus any accepted object-layer license must come from the sibling $\mathsf{ObjectKnowledgeCertificate}^{\uparrow}$ route, while this packet exports only the refusal rows listed above. □

Falsifiable prediction: Any accepted bare-object-refusal packet with an empty missing-field row cannot export an object-layer license without factoring through an existing ObjectKnowledgeCertificateUp packet.

Independence witness: Nearest siblings objectknowledgecertificate, definitedescriptionboundary, negativecertificateboundary, and classifierboundary are not bijective because they respectively accept licensed names, description-expansion rows, apophatic refusal rows, and classifier-boundary rows, while BareObjectRefusalUp records the failed five-field object license itself.

Closure status for $\mathsf{NameCert}_{\mathsf{BareObjectRefusal}\uparrow}$.

Constructive story: A $\mathsf{BareObjectRefusal}^{\uparrow}$ packet is generated from requested object-name, missing-field, refusal, witness-audit, anti-primitive ledger, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BareObjectRefusal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1341.1, Definition 1341.2, and Definition 1341.3 specify the finite failed-license packet and its object-license nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BareObjectRefusal}\uparrow}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.BareObjectRefusalUp.BareObjectRefusalTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close positive object knowledge, description expansion, apophatic refusal in general, classifier-boundary completeness, or bridge checking.

Upgrade path: Obligation closure requires checked source, pattern, classifier, stability, ledger, witness-audit, and object-license nonescape rows for the same packet.

Chapter 1342

A Concrete Naming Certificate for $\mathsf{CauchySealInterface}^{\uparrow}$

The $\mathsf{CauchySealInterface}^{\uparrow}$ packet realizes the Real-completion seal interface of chapter 9238 as one finite naming surface. It is composite: the limit-seal row is supplied by chapter 386, the classifier row by chapter 900, the budget-synchronizer row by chapter 886, and the public observation faces by chapter 126, chapter 301, chapter 291, and chapter 38. The chapter records the common interface rather than a new completeness principle.

Definition 1342.1 (Cauchy seal interface carrier)**.** *A* $\mathsf{CauchySealInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, B, T, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. L *is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row naming the completion-facing limit seal;*
2. Q *is the* $\mathsf{RealCompletionSealClassifierUp}$ *row classifying the visible completion-seal read;*
3. B *is the* $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ *row synchronizing the finite budget used by the seal read;*
4. T *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window face exposed by the synchronized budget;*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback face reached from that finite window;*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger face attached to the same readback;*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *handoff face consumed by Real-completion users;*
8. H *records componentwise* hsame *transport for* L, Q, B, T, R, D, E *and the local naming row;*
9. C *records displayed* Cont *replay through the limit seal, classifier, budget synchronizer, StreamName, RegSeqRat, dyadic, and RealUp rows;*
10. P *is the* Pkg *provenance over* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{RealCompletionSealClassifierUp}$, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchySealInterface}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient streams, host real equality, selected host limits, countable choice, metric-completeness data, or ambient Real completeness.

Theorem 1342.2 (Cauchy seal interface carrier admission)**.** *An accepted* CauchySealInterface$^{\uparrow}$ *carrier is admitted exactly as the finite tuple* $S = (L, Q, B, T, R, D, E, H, C, P, N)$ *of Definition 1342.1. Carrier admission exposes no hidden selector, quotient stream, host equality, selected limit, choice row, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchySealInterfaceUp.CauchySealInterfaceTasteGate single carrier alignment`

Proof. Project the carrier definition. The data rows are precisely the limit seal L, classifier Q, synchronizer B, public faces T, R, D, E, and support rows H, C, P, N.

The cited sibling chapters provide those rows as finite NameCert surfaces, and the present carrier contains no second field from which an external selector or host-level equality could be read. Therefore admission is rowwise admission of the displayed tuple. □

Theorem 1342.3 (Cauchy seal interface synchronization)**.** *Every accepted* CauchySealInterface$^{\uparrow}$ *read synchronizes the* CauchyLimitSeal$^{\uparrow}$ *row* L*, the* RealCompletionSealClassifierUp *row* Q*, and the* CauchySealBudgetSynchronizer$^{\uparrow}$ *row* B *before any of the* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*, or* Real$^{\uparrow}$ *faces are public.*

Proof. Read the source side L, Q, B in the carrier. These rows are the only seal, classifier, and budget-synchronizer coordinates available to the packet.

The public faces T, R, D, E occur downstream of the same support rows H, C, P, N. Since no detached budget or classifier coordinate is present, any public face must be read through the synchronized seal interface L, Q, B. □

Theorem 1342.4 (Cauchy seal interface finite-window readback)**.** *The finite-window segment of an accepted* CauchySealInterface$^{\uparrow}$ *packet is exactly the route* B, T, R, D *supported by* H, C, P, N*. It cannot be replaced by a hidden* StreamName$^{\uparrow}$ *selector, a quotient stream, a selected host limit, or a countable-choice row.*

Proof. Start from the synchronized budget row B. The only finite-window face selected by the interface is T, the only regular-rational readback is R, and the only dyadic ledger attached to that readback is D.

The carrier has no coordinate for any alternative selector or quotient stream. The support rows transport and replay the displayed route, so finite-window readback is exactly B, T, R, D inside the same packet. □

Theorem 1342.5 (Cauchy seal interface RealUp handoff)**.** *Every* Real$^{\uparrow}$ *consumer of an accepted* CauchySealInterface$^{\uparrow}$ *packet reaches the Real-facing row* E *only through the synchronized route*

$$L, Q, B \to T, R, D \to E$$

with the support rows H, C, P, N*. The handoff is a finite seal interface and not a Real-completeness assertion.*

Proof. Use seal synchronization to place L, Q, B before the public observation faces. Use finite-window readback to route the StreamName, RegSeqRat, and dyadic faces through the same budget row.

The Real-facing row E is the next displayed face after T, R, D, and the support rows only preserve the same packet. Thus a RealUp consumer receives a finite handoff, while completeness data remains outside the carrier. □

Theorem 1342.6 (Cauchy seal interface no-choice non-quotient nonescape)**.** *No accepted* $\mathsf{CauchySealInterface}^{\uparrow}$ *consumer can escape the displayed interface to quotient streams, host real equality, selected host limits, countable choice, metric-completeness data, hidden selectors, or ambient Real completeness. The public surface is exhausted by* $L, Q, B, T, R, D, E, H, C, P, N$*.*

Proof. Carrier admission enumerates the public rows, seal synchronization fixes the seal-classifier-budget source, finite-window readback fixes the observation faces, and the RealUp handoff fixes the terminal face.

Every excluded object would require a coordinate not present in $S = (L, Q, B, T, R, D, E, H, C, P, N)$. Therefore the local NameCert row N names only the displayed finite interface. □

This composite packet depends on its sibling rows without collapsing into any one of them: $\mathsf{CauchyLimitSeal}^{\uparrow}$ supplies the limit seal, RealCompletionSealClassifierUp supplies seal classification, $\mathsf{CauchySealBudgetSynchronizer}^{\uparrow}$ supplies budget synchronization, and the StreamName, RegSeqRat, DyadicRatCore, and RealUp rows supply the public readback faces. Later Lean TasteGate formalization for `BEDC.Derived.CauchySealInterfaceUp` must provide `Nontrivial CauchySealInterfaceUp` and `FieldFaithful CauchySealInterfaceUp` instances for the same finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySealInterface}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySealInterface}^{\uparrow}$ packet is generated from a CauchyLimitSeal row, RealCompletionSealClassifier row, CauchySealBudgetSynchronizer row, StreamName finite-window face, RegSeqRat readback face, DyadicRatCore ledger face, RealUp handoff face, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySealInterface}^{\uparrow}}$, seed)

Scope closed: Definition 1342.1, Definition 1342.2, Definition 1342.3, Definition 1342.4, Definition 1342.5, and Definition 1342.6 bind the finite interface carrier, seal synchronization, finite-window readback, RealUp handoff, and no-choice non-quotient boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySealInterface}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, Real completeness, quotient stream equality, host real equality, selected host limits, countable choice, metric-completeness data, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seal synchronization, finite-window readback, RealUp handoff, no-choice non-quotient nonescape, and local NameCert support for the same finite packet.

Chapter 1343

A Concrete Naming Certificate for $\mathsf{GapSocketBoundary}^{\uparrow}$

The $\mathsf{GapSocketBoundary}^{\uparrow}$ packet realizes the gap-socket boundary of section 9462.26 as finite NameCert data. It sits beside the typed taxonomy of chapter 1240, the apophatic far-end socket of chapter 1061, the externality gate of chapter 584, the reflection-gap boundary of chapter 1161, and the forward-GAP commitment packet of chapter 505. Those siblings classify socket kinds, expose the common far end, police external supply, separate reflection gaps, and bind observed regularity; this packet records the boundary where the typed socket requests missing supply without importing the missing object.

Falsifiable prediction: Any accepted GapSocketBoundaryUp packet missing typed socket, externality gate, apophatic far-end, refusal ledger, transport, route, provenance, or local NameCert rows is rejected.

Independence witness: typedgapsockettaxonomy, apophaticsocket, externalitygate, reflectiongapboundary, causalcommitment: none is bijective to $\mathsf{GapSocketBoundary}^{\uparrow}$. The taxonomy packet carries the socket-kind matrix but not one boundary request; the apophatic socket names refusal discipline without the typed externality row; the externality gate records gate admission without the far-end refusal ledger; the reflection-gap boundary records self-description pressure rather than missing supply; and the causal commitment packet records a forward regularity GAP rather than a typed boundary socket.

Definition 1343.1 (Gap-socket boundary carrier)**.** *A* $\mathsf{GapSocketBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, E, A, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed gap-socket row, read through* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$*, naming the socket kind and the boundary request;*
2. *E is the externality-gate row, recording that the requested supply is outside the present finite packet;*
3. *A is the apophatic far-end row, naming the far-end boundary without internalizing the missing object;*
4. *R is the refusal ledger, recording rejected reads that try to turn missing supply into an internal witness, host target, quotient equality, or closure theorem;*

5. *H records componentwise* hsame *transport for T, E, A, R and the local naming row;*
6. *C records displayed* Cont *replay through socket admission, externality-gate check, far-end naming, refusal lookup, provenance, and local naming;*
7. *P is the* Pkg *provenance over* TypedGapSocketTaxonomy$^{\uparrow}$, ApophaticSocket$^{\uparrow}$, ExternalityGate$^{\uparrow}$, ReflectionGapBoundary$^{\uparrow}$, CausalCommitment$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
8. *N is the local* NameCert$_{\text{GapSocketBoundary}^{\uparrow}}$ *row.*

The carrier exports no missing witness, host supply, far-end internalization, total reflection closure, hidden causal rule, quotient equality, or untyped boundary socket.

Theorem 1343.2 (Gap-socket boundary NameCert obligations)**.** *For the carrier of Definition 1343.1, the local* NameCert$_{\text{GapSocketBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (T, E, A, R, H, C, P, N)$, typed socket exposure through T, externality-gate exposure through E, apophatic far-end exposure through A, refusal-ledger exposure through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.GapSocketBoundaryUp.GapSocketBoundaryTasteGate single carrier alignment`

Proof. Project the accepted carrier. The packet displays one typed socket row, one gate row, one far-end row, one refusal ledger, and the transport, replay, provenance, and naming rows.

Read the sibling rows through their stated roles. TypedGapSocketTaxonomy$^{\uparrow}$ classifies the socket, ApophaticSocket$^{\uparrow}$ and ApophaticFarEndSocket$^{\uparrow}$ keep the far end named without internalization, ExternalityGate$^{\uparrow}$ records the outside-supply boundary, ReflectionGapBoundary$^{\uparrow}$ supplies the reflection-pressure sibling, and CausalCommitment$^{\uparrow}$ supplies the forward-GAP sibling without replacing the boundary request.

The refusal ledger R rejects every read that asks the boundary packet to supply the missing object. Since no additional coordinate exists beyond T, E, A, R, H, C, P, N, the local NameCert obligations are exactly the listed finite rows. □

The Lean `TasteGate.lean` development for GapSocketBoundary$^{\uparrow}$ must provide explicit Nontrivial GapSocketBou and FieldFaithful GapSocketBoundary$^{\uparrow}$ instances. These instances must preserve the typed socket row, externality gate, apophatic far-end, refusal ledger, componentwise transport, displayed route, package provenance, and local naming row.

Closure status for NameCert$_{\text{GapSocketBoundary}^{\uparrow}}$**.**

Constructive story: A GapSocketBoundary$^{\uparrow}$ packet is generated from a typed socket row, externality gate, apophatic far-end, refusal ledger, componentwise transport, displayed replay route, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{GapSocketBoundary}^{\uparrow}}$, obligation)

Scope closed: Definition 1343.1 and Definition 1343.2 enumerate the carrier rows and local NameCert obligation surface for the finite boundary packet.

Formal verification: FormalStatus(NameCert$_{\text{GapSocketBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not supply the missing object, host supply, far-end internalization, reflection closure, hidden causal rule, quotient equality, Lean verification, or bridge checking for GapSocketBoundary$^{\uparrow}$.

Upgrade path: Lean closure requires Nontrivial GapSocketBoundaryUp and FieldFaithful GapSocketBoundaryUp instances in `lean4/BEDC/Derived/GapSocketBoundaryUp/TasteGate.lean`.

Chapter 1344

A Concrete Naming Certificate for RegistryNoSmugglingPredicate$^{\uparrow}$

The RegistryNoSmugglingPredicate$^{\uparrow}$ packet realizes the first machine-facing no-smuggling predicate demanded by subsection 9462.35.5 as finite registry NameCert data. It sits beside the target rows of MachineInterfaceFormalizationTarget$^{\uparrow}$, the export rows of RegistryExportSurface$^{\uparrow}$, the status gate of RegistryExportConsistencyGate$^{\uparrow}$, the classifier of chapter 1329, and the machine boundary of MachineInterfaceBoundary$^{\uparrow}$. Those siblings store targets, expose export rows, check consistency, classify smuggling forms, and limit machine-facing reads; this packet is the registry predicate that refuses theorem-evidence export whenever one of those routes reports a smuggling row.

Definition 1344.1 (Registry no-smuggling predicate carrier)**.** *A* RegistryNoSmugglingPredicate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, E, G, U, H, C, P, N)$$

with the following displayed rows:

1. *T is the formalization-target row, recording the* MachineInterfaceFormalizationTarget$^{\uparrow}$ *name, namespace, statement skeleton, dependency surface, expected status, and not-claimed surface being checked;*
2. *E is the registry-export row, recording the* RegistryExportSurface$^{\uparrow}$ *registration, blocking, status, machine-target, audit, and non-smuggling fields for the same candidate export;*
3. *G is the consistency-gate row, recording the* RegistryExportConsistencyGate$^{\uparrow}$ *status, audit-route, formal-target readback, and no-smuggling decision;*
4. *U is the refusal-classifier row, recording the* SmugglingPatternClassifier$^{\uparrow}$ *reading as no philosophical smuggling, no mathematical smuggling, no host smuggling, no roadmap-as-theorem, or no cannot-claim conflict;*
5. *H records componentwise* hsame *transport for target, export, gate, classifier, boundary, replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay through the machine-interface boundary, target lookup, registry export surface, consistency gate, classifier refusal, provenance, and local naming;*
7. *P records* Pkg *provenance over* MachineInterfaceFormalizationTarget$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, RegistryExportConsistencyGate$^{\uparrow}$, SmugglingPatternClassifier$^{\uparrow}$, MachineInterfaceBoundary$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

8. N is the local $\mathsf{NameCert}_{\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}}$ *row.*
The carrier has no coordinate for upgrading a roadmap, philosophical doctrine, mathematical tradition, host primitive, or cannot-claim entry into theorem evidence unless the displayed target, export, gate, classifier, boundary, transport, replay, provenance, and local naming rows all admit the route.

Lean4 ✓ `BEDC.Derived.RegistryNoSmugglingPredicateUp.RegistryNoSmugglingPredicateTasteGate`
`single carrier alignment`

Theorem 1344.2 (Registry no-smuggling predicate NameCert obligations)**.** *For the carrier of Definition 1344.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for $R = (T, E, G, U, H, C, P, N)$;*
2. *target and export exposure through T and E;*
3. *consistency and no-smuggling routing through G and U;*
4. *machine-boundary replay, componentwise transport, and package provenance through C, H, P;*
5. *local naming through N.*

Proof. Project the displayed packet $R = (T, E, G, U, H, C, P, N)$. The target row T names the machine-facing object being checked, while E records the registry export surface that would otherwise expose it to a theorem-facing consumer.

The consistency-gate row G and classifier row U decide whether the same candidate passes no-smuggling. The classifier row keeps the five early refusal modes separate: philosophical smuggling, mathematical smuggling, host smuggling, roadmap-as-theorem, and cannot-claim conflict.

The rows H, C, P, N then preserve and name the same decision across transport, replay, package provenance, and local certification. Since no additional coordinate exists, the five displayed obligation rows exhaust the local NameCert surface. □

Theorem 1344.3 (No-smuggling refusal blocks theorem-evidence export)**.** *If an accepted* RegistryNoSmugglingPred *packet marks a registry export by a no-smuggling refusal row, then that export cannot be read as theorem evidence by the same machine-facing registry route. Any later theorem evidence must pass through a distinct accepted packet whose classifier row no longer carries the refusal.*

Proof. Read the candidate export through E and the same target through T. The gate row G compares that export with the target status, dependency surface, and audit route before a machine-facing consumer may use it.

When U records a no-smuggling refusal, the displayed route is refusal rather than admission. The sibling classifier of chapter 1329 supplies the classification row, while the machine-interface boundary keeps the route from importing a hidden theorem object.

The transport, replay, provenance, and naming rows preserve the refused route. Therefore the same packet cannot export theorem evidence; a theorem-evidence export requires a separately displayed packet with its own accepted classifier row. □

Falsifiable prediction: A registry export marked by a RegistryNoSmugglingPredicateUp no-smuggling refusal row cannot be exported as theorem evidence by the same machine-facing registry route.

Independence witness: machineinterfaceformalizationtarget, registryexportsurface, registryexportconsistencygate, smugglingpatternclassifier, machineinterfaceboundary: none carries the joint target, export, consistency-gate, five-form no-smuggling refusal, boundary replay, transport, provenance, and local NameCert packet of $\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}$.

The Lean follow-up must provide $\mathsf{Nontrivial\ RegistryNoSmugglingPredicate}^{\uparrow}$ and $\mathsf{FieldFaithful\ RegistryNoSmugglin}$ instances in `lean4/BEDC/Derived/RegistryNoSmugglingPredicateUp/TasteGate.lean`. The field-faithfulness witness must preserve T, E, G, U, H, C, P, N as distinct rows before any machine-facing registry consumer cites BEDC.Derived.RegistryNoSmugglingPredicateUp.

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}$ packet is generated from machine target, registry export, consistency gate, no-smuggling classifier, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1344.1, Definition 1344.2, and Definition 1344.3 bind the finite predicate packet and its export refusal consequence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not supply theorem evidence, bridge checking, host primitive admission, mathematical-tradition import, philosophical truth, roadmap closure, or automatic registry proof generation for $\mathsf{RegistryNoSmugglingPredicate}^{\uparrow}$.

Upgrade path: Lean closure requires Nontrivial RegistryNoSmugglingPredicateUp and FieldFaithful RegistryNoSmugglingPredicateUp instances.

Chapter 1345

A Concrete Naming Certificate for $\mathsf{RecursorBranchCompleteness}^{\uparrow}$

The $\mathsf{RecursorBranchCompleteness}^{\uparrow}$ packet realizes the branch-completeness side of the recursor-as-authorized-generator vision of chapter 9373. It sits next to the authorized generator surface of chapter 525 and the finite kernel audit perspective of chapter 484. The packet records when an accepted inductive signature, emitted recursor, motive, and branch table have all been checked before any recursor output is available to a public consumer.

Definition 1345.1 (Recursor branch-completeness carrier)**.** *A* $\mathsf{RecursorBranchCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, R, M, B, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted inductive signature row, including the constructor index set and strict-positivity admission needed before a recursor may be emitted.*
2. *R is the emitted recursor row, naming the eliminator generated from I.*
3. *M is the motive row, recording the target family against which every constructor branch is checked.*
4. *B is the branch package row, listing one displayed branch for each constructor admitted by I.*
5. *K is the branch-completeness ledger, recording that the constructor index rows of I and the branch rows of B have the same finite coverage.*
6. *H records componentwise* hsame *transport for I, R, M, B, K, C, P, N.*
7. *C records the displayed* Cont *replay routes by which a consumer reads the emitted recursor only after the branch package and completeness ledger have been replayed.*
8. *P is the* Pkg *provenance over* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{FiniteKernelCategory}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RecursorBranchCompleteness}^{\uparrow}}$ *row.*

The carrier has no coordinate for a missing branch, host induction principle, proof-producing generator channel, unchecked motive, or recursor output detached from the displayed branch-completeness ledger.

Theorem 1345.2 (Recursor branch-completeness NameCert obligations)**.** *For the carrier of Definition 1345.1, the local* $\mathsf{NameCert}_{\mathsf{RecursorBranchCompleteness}^{\uparrow}}$ *surface consists exactly of signature admission through* I*, emitted recursor naming through* R*, motive checking through* M*, constructor-indexed branch exposure through* B*, branch-completeness exactness through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RecursorBranchCompletenessUp.TasteGate.RecursorBranchCompletenessTasteGate single carrier alignment`

Proof. Carrier admission is projection from the displayed packet $Q = (I, R, M, B, K, H, C, P, N)$. The accepted signature row I supplies the constructor index set, and the emitted recursor row R is read only as the eliminator generated from that admitted signature.

The motive and branch rows supply the finite checking surface. The motive row M fixes the target family, while B gives one displayed branch for each constructor row admitted by I.

The completeness ledger K binds those two finite lists together. It records that a consumer cannot read R as authorized unless every constructor row in I has a corresponding branch in B and every branch in B is accounted for by the same signature.

Finally, H, C, P, N keep the obligation surface local. Transport, continuation replay, provenance, and the local NameCert row preserve the displayed packet rather than replacing it by host induction or a proof-producing generator channel. □

Theorem 1345.3 (Recursor branch-completeness non-escape)**.** *Every public consumer of an accepted* $\mathsf{RecursorBranchCompleteness}^{\uparrow}$ *packet can read only the displayed signature, emitted recursor, motive, branch package, branch-completeness ledger, transport, replay, provenance, and local NameCert rows*

$$I, R, M, B, K, H, C, P, N.$$

It cannot export a recursor output, missing-branch repair, unchecked motive, hidden induction rule, or proof-producing generator apart from the same finite packet.

Proof. Start with the NameCert obligations. They force every public read through the displayed rows and make K the only ledger by which branch coverage can be consumed.

The authorized generator sibling supplies the recursor discipline. By chapter 525, an authorized generator recursor is public only through accepted signature, motive, branch, descent, audit, transport, continuation, package, boundary, and local naming rows. The present packet records the branch-completeness portion of that route and does not add a separate generator outlet.

The finite kernel category sibling supplies the finite audit perspective. Its finite-kernel rows license only displayed constructor and morphism data, so a consumer cannot use them to manufacture an off-packet branch or host induction rule.

Thus any attempted export outside $I, R, M, B, K, H, C, P, N$ has no carrier coordinate. The accepted packet exposes branch-complete recursor use as finite NameCert data and nothing beyond that packet. □

Falsifiable prediction: Deleting the branch-completeness ledger K leaves an emitted recursor packet whose branch package cannot certify constructor-index coverage for all accepted constructors in I.

Independence witness: authorizedgeneratorrecursor and finitekernelcategory are not bijective to RecursorBranchCompletenessUp because the present packet alone records accepted signature,

emitted recursor, motive, branch package, branch-completeness ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows together.

Closure status for $\mathsf{NameCert}_{\mathsf{RecursorBranchCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{RecursorBranchCompleteness}^{\uparrow}$ packet is generated from recursor-code rows, motive rows, branch-family rows, branch-completeness ledger rows, output rows, coverage-audit rows, obstruction rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier accepts a recursor output only when every branch required by the displayed motive is accounted for in the finite ledger. The constructive story is branch coverage before elimination: missing branches, unchecked motives, proof-producing generator channels, and host induction principles are represented as absence or obstruction, while the accepted route is generated by replaying the same finite branch ledger through transport and continuation.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RecursorBranchCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation, non-escape theorem, falsifiable prediction, and independence witness for $\mathsf{RecursorBranchCompleteness}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RecursorBranchCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim missing branches, host induction principles, proof-producing generator channels, unchecked motives, or recursor outputs detached from the branch-completeness ledger.

Upgrade path: To reach obligationClosure, the branch-completeness carrier and obstruction boundary must be checked as finite recursor-ledger obligations.

Chapter 1346

A Concrete Naming Certificate for $\mathsf{TraceVerdictBoundary}^{\uparrow}$

The $\mathsf{TraceVerdictBoundary}^{\uparrow}$ packet realizes the proof discipline of subsection 9462.30.6 as a concrete boundary between proof traces and verdict export. It sits beside the proof-trace audit witness of chapter 1219, the machine-interface formalization target of chapter 1225, the ground compiler audit verdict of chapter 822, and the definition-theorem-proof audit packet of chapter 1334. Those siblings expose trace witnesses, interface targets, verdict rows, and proof-discipline rows separately; this chapter binds the handoff from trace data to public verdict export.

Definition 1346.1 (Trace verdict boundary carrier)**.** *A* $\mathsf{TraceVerdictBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, R, V, I, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *row recording the finite proof trace, transport, ledger, refusal, audit-verdict, provenance, and local NameCert witness;*
2. *R is the forbidden-resource refusal row recording hidden choice, host equality as BEDC equality, quotient collapse, propositional extensional rewrite, global truth predicate, global clock, unledgered compression, and unscoped universal claim as unavailable proof resources;*
3. *V is the verdict-export row supplied through the* $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$ *surface;*
4. *I is the* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ *row naming the checker-facing formalization target through which the verdict may be exported;*
5. *A is the* $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ *row recording the proof-discipline site whose trace is being judged;*
6. *H records componentwise* hsame *transport for T, R, V, I, A, C, P, N;*
7. *C records displayed* Cont *replay for the trace, refusal, verdict, interface-target, proof-discipline, provenance, and local naming rows;*
8. *P is the* Pkg *provenance over chapter 1219, chapter 1225, chapter 822, chapter 1334, subsection 9462.30.6,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TraceVerdictBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for reading a proof trace as a verdict, exporting a verdict without a refusal row, bypassing the machine-interface target, or replacing the displayed proof-discipline site with host-side semantic confidence.

Definition 1346.2 (Trace verdict boundary classifier)**.** *Two accepted* $\mathsf{TraceVerdictBoundary}^{\uparrow}$ *carriers classify together exactly when their proof-trace, refusal, verdict, interface-target, proof-discipline, transport, replay, provenance, and local naming rows compare componentwise by* hsame*:*

$$T \sim_{\mathsf{hsame}} T', \quad R \sim_{\mathsf{hsame}} R', \quad V \sim_{\mathsf{hsame}} V', \quad I \sim_{\mathsf{hsame}} I',$$
$$A \sim_{\mathsf{hsame}} A', \quad H \sim_{\mathsf{hsame}} H', \quad C \sim_{\mathsf{hsame}} C', \quad P \sim_{\mathsf{hsame}} P', \quad N \sim_{\mathsf{hsame}} N'.$$

No classifier clause compares hidden proof-checker state, host-side truth, private verdict policy, or trace data outside the displayed handoff.

Theorem 1346.3 (Trace verdict boundary NameCert obligations)**.** *For the carrier of Definition 1346.1, the local* $\mathsf{NameCert}_{\mathsf{TraceVerdictBoundary}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for* $B = (T, R, V, I, A, H, C, P, N)$ *with proof trace, forbidden-resource refusal, verdict export, interface target, proof-discipline site, transport, replay, provenance, and local name all displayed;*
2. *classifier stability for the same rows under componentwise* hsame *as in Definition 1346.2;*
3. *trace-to-verdict exactness, requiring a verdict export through* V *to read the proof trace* T *only together with the refusal row* R *and interface-target row* I*;*
4. *provenance and route exactness, requiring every verdict-boundary read to pass through the proof-trace witness, machine-interface formalization target, ground compiler audit verdict, definition-theorem-proof audit packet, replay row, package row, and local name;*
5. *non-escape for bare proof traces, forbidden-resource erasure, hidden checker state, private verdict policy, host semantic confidence, and verdict export outside the displayed interface target.*

Lean4 ✓ `BEDC.Derived.TraceVerdictBoundaryUp.TasteGate.TraceVerdictBoundaryNameCertObligations`

Proof. Carrier admission is projection from Definition 1346.1. The packet displays trace, refusal, verdict, interface target, proof-discipline, transport, replay, provenance, and local naming rows before the verdict boundary can be read.

Classifier stability is componentwise. The classifier of Definition 1346.2 compares exactly those rows by hsame, so a transported packet cannot keep a verdict while dropping its trace, refusal, interface target, or proof-discipline row.

Trace-to-verdict exactness follows from the route order. The proof trace T supplies audit evidence, but it does not become a verdict export until R, V, I, A, H, C, P, N are displayed in the same packet.

Provenance and route exactness use the sibling packets named in the statement. The proof-trace witness supplies the trace row, the interface target supplies checker-facing export scope, the ground compiler audit verdict supplies verdict data, and the definition-theorem-proof audit supplies the proof-discipline site.

Non-escape is immediate from the carrier inventory. There is no coordinate for bare trace export, hidden checker state, private verdict policy, forbidden-resource erasure, or host semantic confidence, so the NameCert surface cannot export one. □

Theorem 1346.4 (Trace verdict boundary nonescape)**.** *An accepted* $\mathsf{TraceVerdictBoundary}^{\uparrow}$ *packet never exports a proof trace as a verdict unless the forbidden-resource refusal row, ground compiler audit verdict row, machine-interface target row, proof-discipline row, transport row, replay row, provenance row, and local name are displayed with it. Deleting the refusal row while preserving the trace and verdict rows leaves a partial trace-verdict projection, not an accepted verdict boundary carrier.*

Proof. Start with the NameCert obligations of Definition 1346.3. They require trace-to-verdict exactness through $T, R, V, I, A, H, C, P, N$ before a public verdict-boundary read is accepted.

If R is absent, the packet no longer records the refusal of hidden choice, host equality, quotient collapse, propositional extensional rewrite, global truth, global clock, unledgered compression, or unscoped universal claim. A trace row and verdict row without that refusal do not satisfy the carrier admission clause.

The interface and proof-discipline rows cannot repair the missing refusal. They scope the verdict export and identify the proof site, but the boundary classifier still compares R as a displayed row. Hence the partial read is rejected by the $\mathsf{TraceVerdictBoundary}^{\uparrow}$ classifier rather than exported as a verdict boundary. □

Theorem 1346.5 (Trace verdict boundary proof-trace handoff)**.** *Every accepted* $\mathsf{TraceVerdictBoundary}^{\uparrow}$ *verdict export factors through the* $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ *row and then through the forbidden-resource refusal,* $\mathsf{GroundCompilerAuditVerdict}^{\uparrow}$*,* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$*,* $\mathsf{DefinitionTheoremProofAudit}^{\uparrow}$ *transport, replay, provenance, and local NameCert rows. The proof trace remains audit evidence for the verdict route; it is not itself the exported verdict.*

Proof. Project the carrier $B = (T, R, V, I, A, H, C, P, N)$. The first visible witness row is the $\mathsf{ProofTraceAuditWitness}^{\uparrow}$ row T, while the verdict row V appears only after the refusal row R and alongside the interface target I and proof-discipline row A.

Apply the NameCert obligations. Trace-to-verdict exactness requires the verdict export to preserve $T, R, V, I, A, H, C, P, N$ as one route, and provenance exactness records the sibling surfaces named in the statement.

Finally use nonescape. Since a bare proof trace cannot be exported as a verdict and a missing refusal row invalidates the carrier, the handoff from proof trace to verdict export is exactly the displayed finite boundary route. □

Falsifiable prediction: Deleting the forbidden-resource refusal row while keeping trace and verdict rows invalidates the verdict boundary carrier.

Independence witness: prooftraceauditwitness, machineinterfaceformalizationtarget, groundcompilerauditverdict, auditgatefailurewitness, definitiontheoremproofaudit: none is bijective because this packet uniquely binds trace rows to verdict export through refusal, interface target, provenance, and local NameCert rows.

Lean TasteGate.lean must include explicit Nontrivial $\mathsf{TraceVerdictBoundary}^{\uparrow}$ and FieldFaithful TraceVerdictBoundar instances preserving $T, R, V, I, A, H, C, P, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TraceVerdictBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TraceVerdictBoundary}^{\uparrow}$ packet is generated from proof-trace rows, verdict rows, refusal rows, machine-target rows, proof-discipline boundary rows, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier separates a trace from a verdict: the proof trace can be handed to a machine-facing route only through the discipline and refusal coordinates already displayed in the finite carrier. The ledger story is verdict boundary preservation, where replay exposes which trace, target, discipline, and refusal rows were used, while hidden checker state, private verdict policy, forbidden-resource erasure, and host semantic confidence have no coordinate.

Theory closure: TheoryGate(NameCert$_{\mathsf{TraceVerdictBoundary}^{\uparrow}}$, seed)

Scope closed: This chapter binds the carrier and classifier definitions, NameCert-obligation, nonescape, proof-trace-handoff, independence witness, and Lean-follow-up boundary for TraceVerdictBoundary$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{TraceVerdictBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: It does not claim reading proof traces as verdicts, verdict export without refusal rows, machine-target bypass, or replacement by host-side semantic confidence.

Upgrade path: To reach obligationClosure, the trace-verdict classifier and handoff boundary must be checked as finite proof-route obligations.

Chapter 1347

A Concrete Naming Certificate for $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$

The $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ packet realizes the dependency and no-smuggling side of the machine-facing interface of subsection 9462.35.5 as one finite NameCert audit. It sits beside the target carrier of chapter 1225, the classifier of chapter 1329, the dependency lattice of chapter 868, and the export gate of chapter 1224. Those siblings store target skeletons, classify smuggling attempts, expose dependency modes, and block inconsistent registry export; this packet binds the target row, dependency row, no-smuggling decision, axiom-dependency audit, and export-gate result before a formalization queue may treat the rows as one admissible target.

Definition 1347.1 (Formal-target dependency audit carrier)**.** *A* $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, D, S, K, U, A, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ *row, recording the target name, namespace, registry row, statement skeleton, expected status, audit-gate row, and not-claimed surface.*
2. *D is the dependency-list row read from the same formalization target and normalized as finite machine-facing prerequisites.*
3. *S is the* $\mathsf{SmugglingPatternClassifier}^{\uparrow}$ *no-smuggling decision row, exposing whether the target attempts to read prose, host supply, model output, or private registry state as internal BEDC content.*
4. *K is the* $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ *row, recording the internal, restricted, or externally supplied dependency mode for each displayed prerequisite.*
5. *U is the unresolved-supply row, listing the finite dependencies or socket reports that remain outside theorem content after D, S, K have been read.*
6. *A is the audit-decision row, recording whether the target-dependency pair is admissible, refused, or held as a queue record.*
7. *G is the* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ *row, checking that registry status, formal-target readback, audit route, and no-smuggling decision agree before export.*

8. *H records componentwise* hsame *transport for target, dependency, no-smuggling, axiom-audit, unresolved-supply, audit-decision, export-gate, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the target row through dependency readback, no-smuggling classification, axiom-dependency audit, unresolved-supply exposure, audit decision, and export-gate result.*
10. *P is the* Pkg *provenance over* $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$, $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$, $\mathsf{SmugglingPatternClassifier}^{\uparrow}$, $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$, $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalTargetDependencyAudit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a dependency imported outside D, a no-smuggling decision erased from S, an axiom-dependency reading outside K, an export-gate result outside G, or a proof object extracted from an admissible queue record.

Lean4 (definition) `BEDC.Derived.FormalTargetDependencyAuditUp.FormalTargetDependencyAuditUp`

Theorem 1347.2 (Formal-target dependency audit NameCert obligations)**.** *For the carrier of Definition 1347.1, the local* $\mathsf{NameCert}_{\mathsf{FormalTargetDependencyAudit}^{\uparrow}}$ *surface consists exactly of carrier admission for $F = (T, D, S, K, U, A, G, H, C, P, N)$, target exposure through T, finite dependency readback through D, no-smuggling classification through S, axiom-dependency audit through K, unresolved-supply exposure through U, audit-decision routing through A, registry-export consistency through G, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $F = (T, D, S, K, U, A, G, H, C, P, N)$. The target row T supplies the machine-facing skeleton, and D exposes the finite prerequisite list read from the same target.

Read S, K, U, A next. The classifier row S keeps no-smuggling visible, the dependency-audit row K records how each prerequisite is read, the unresolved-supply row U records what remains outside theorem content, and the audit-decision row A decides admission only for that displayed combination.

Finally inspect G, H, C, P, N. The export gate G checks the same target, dependency, audit, and no-smuggling rows against the registry surface, while transport, replay, provenance, and local naming preserve the packet. Since no additional coordinate remains, these rows exhaust the local NameCert obligations. □

Theorem 1347.3 (Formal-target dependency audit nonescape)**.** *A* $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ *packet whose no-smuggling row S is absent, erased, or not transported with T, D, K, A, G cannot be exported as an admissible machine-facing formalization target. Such a packet remains a queue record or refusal report, not theorem content and not a certified target-dependency audit.*

Proof. Begin with the target and dependency rows. The target skeleton T and dependency list D identify what the queue wants to certify, but neither row decides whether external supply has been smuggled into the target.

Use the no-smuggling row S. By Definition 1347.2, the local NameCert surface must expose S together with the axiom-dependency audit K, unresolved-supply row U, audit decision A, and export gate G.

If S is absent or not transported with the other rows, the export gate has no displayed classifier result to compare with registry status and target readback. Componentwise hsame and Cont replay

cannot manufacture a missing row, so the packet cannot be accepted as an admissible machine-facing target. □

Theorem 1347.4 (Formal-target dependency audit sibling independence)**.** *The* $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$, $\mathsf{SmugglingPatternClassifier}^{\uparrow}$, $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$, *or* $\mathsf{RegistryExportConsistencyGate}^{\uparrow}$.

Proof. Compare the displayed rows. $\mathsf{MachineInterfaceFormalizationTarget}^{\uparrow}$ supplies the target skeleton and dependency list but does not itself bind the no-smuggling decision, axiom-dependency audit, and export-gate result as one audit.

$\mathsf{SmugglingPatternClassifier}^{\uparrow}$ supplies the no-smuggling classification but lacks the target skeleton and dependency list. $\mathsf{AxiomDependencyAuditMap}^{\uparrow}$ supplies dependency modes and required-supply rows but does not store the formal target or registry export result.

$\mathsf{RegistryExportConsistencyGate}^{\uparrow}$ checks export consistency, audit route, no-smuggling, and formal-target readback, but it does not expose the dependency-list normalization and axiom-dependency audit rows as the same target packet. A row-preserving bijection would have to preserve all of $T, D, S, K, U, A, G, H, C, P, N$, so none of these siblings is bijective to $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$. □

Falsifiable prediction: A FormalTargetDependencyAuditUp packet whose no-smuggling row is erased cannot be exported as an admissible machine-facing formalization target.

Independence witness: machineinterfaceformalizationtarget, smugglingpatternclassifier, axiomdependencyauditmap, registryexportconsistencygate: none is bijective because each lacks at least one of target skeleton, dependency list, no-smuggling decision, axiom-dependency audit, or export-gate result rows.

The later Lean file `lean4/BEDC/Derived/FormalTargetDependencyAuditUp/TasteGate.lean` must provide Nontrivial $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ and FieldFaithful $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ instances over the same carrier. The field-faithful route must preserve $T, D, S, K, U, A, G, H, C, P, N$ before `BEDC.Derived.FormalTargetDependencyAuditUp.FormalTargetDependencyAuditUp` can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalTargetDependencyAudit}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$ packet is generated from a machine-facing target row, dependency-list row, no-smuggling decision, axiom-dependency audit row, unresolved-supply row, audit-decision row, registry-export gate, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalTargetDependencyAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1347.1, Definition 1347.2, Definition 1347.3, and Definition 1347.4 bind the finite target-dependency audit, no-smuggling nonescape, and sibling-independence surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalTargetDependencyAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, public export closure, bridge checking, automatic dependency discharge, or proof-object production for $\mathsf{FormalTargetDependencyAudit}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, target exposure, dependency readback, no-smuggling classification, axiom-dependency audit, unresolved-supply exposure, audit-decision routing, export-gate consistency, TasteGate nontriviality, and field faithfulness for $T, D, S, K, U, A, G, H, C, P, N$.

Chapter 1348

A Concrete Naming Certificate for $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$

The $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$ packet realizes the reality-facing signature discipline of Definition 9413.89, especially section 9414.1, as one finite NameCert carrier for external-supply certification. It sits beside the residue packet of chapter 1213, the descent packet of chapter 1303, the fit packet of **??**, the GAP socket boundary of chapter 1343, and the registry no-smuggling predicate of chapter 1344. Those siblings expose gap residue, operation descent, physical-signature fit, socket boundary, and registry refusal rows separately; this packet binds an external source signature to the mathematical classifier only when the physical row, descent certificate, hidden-supply ledger, and local naming row travel with it.

Falsifiable prediction: Every accepted ExternalSupplySignatureSealUp packet that exports a mathematical classifier without the physical signature row fails the descent-refusal obligation.

Independence witness: reality-constrained-signature-residue, reality-constrained-signature-descent, reality-constrained-signature-fit, gap-socket-boundary, registrynosmugglingpredicate: none is bijective to the seal because each lacks at least one of the ontic source, external representation, physical signature, GAP socket, descent certificate, hidden-supply ledger, and local NameCert rows.

Definition 1348.1 (External supply signature seal carrier)**.** *An* $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O, M, Q, G, D, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the ontic signature source row from Definition 9414.3;*
2. *M is the mathematical representation row from Definition 9414.4;*
3. *Q is the physical signature row exposed by the admissible question bundle;*
4. *G is the GAP socket row that records the represented absence and admissible compression boundary;*
5. *D is the descent certificate row that decides which operations survive the classifier route;*
6. *L is the hidden-supply ledger recording source material not exported by the mathematical representation;*

7. *H records componentwise* hsame *transport for the source, representation, signature, GAP, descent, ledger, and local naming rows;*
8. *C records displayed* Cont *replay for the same route;*
9. *P is the* Pkg *provenance over* ExternalSupplySignatureSeal$^{\uparrow}$, RealityConstrainedSignatureResidue$^{\uparrow}$, RealityConstrainedSignatureDescent$^{\uparrow}$, RealityConstrainedSignatureFit$^{\uparrow}$, GapSocketBoundary$^{\uparrow}$, RegistryNoSmugglin BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplySignatureSeal}^{\uparrow}}$ *row.*

The classifier may compare accepted carriers only through these rows. It has no coordinate for a bare mathematical classifier, a hidden physical signature, an unledgered external source, or a registry export detached from G, D, L, N.

Theorem 1348.2 (External supply signature seal NameCert obligations)**.** *For the carrier of Definition 1348.1, the local* $\mathsf{NameCert}_{\mathsf{ExternalSupplySignatureSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (O, M, Q, G, D, L, H, C, P, N)$, source-signature exposure through O, Q, representation exposure through M, GAP and descent exposure through G, D, hidden-supply ledger exposure through L, structural replay through H, C, P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ExternalSupplySignatureSealUp.ExternalSupplySignatureSealNameCert obligations`

Proof. Project the carrier. The tuple lists one ontic source row, one mathematical representation row, one physical signature row, one GAP socket row, one descent certificate row, one hidden-supply ledger row, and the structural support rows.

Read the sibling rows componentwise. The residue sibling supplies representation-gap discipline, the descent sibling supplies operation-descent discipline, the fit sibling supplies physical-signature compatibility, the GAP socket boundary supplies represented absence, and the registry no-smuggling predicate supplies the export refusal row.

The support rows H, C, P, N transport, replay, source, and name only this displayed packet. Since no further coordinate exists, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1348.3 (External supply signature seal descent refusal)**.** *An accepted* ExternalSupplySignatureSeal$^{\uparrow}$ *packet refuses any export of the mathematical representation row M as a reality-facing classifier unless the same route displays the physical signature row Q, the GAP socket row G, the descent certificate row D, and the hidden-supply ledger row L.*

Proof. Begin with the obligation surface of Definition 1348.2. It makes source-signature exposure, GAP exposure, descent exposure, and ledger exposure separate required rows of the same local certificate.

Use the descent and fit siblings next. chapter 1303 supplies the operation-descent row, while **??** supplies the physical-signature compatibility row.

Finally attach the GAP socket and no-smuggling siblings. The boundary of chapter 1343 records the represented absence, and chapter 1344 prevents a registry route from exporting the mathematical row as if the omitted source rows had been supplied. Thus a route missing Q, G, D, L is refused rather than accepted as a seal. □

Theorem 1348.4 (External supply signature seal ledger exactness)**.** *In every accepted* ExternalSupplySignatureSeal$^{\uparrow}$ *packet, the hidden-supply ledger L records exactly the source material not exported by M but still required by the physical signature row Q, the GAP socket row G, and the descent certificate row D.*

Proof. First project the carrier rows O, M, Q, G, D, L. The source row O is not identified with its mathematical representation M, and the physical signature row Q is not recoverable from M unless the packet displays the GAP, descent, and ledger rows.

Apply the NameCert obligations. They expose L as its own ledger coordinate, not as a consequence of M or Q alone.

The residue, descent, and no-smuggling siblings then fix exactness. Residue records what remains after mathematical identification, descent records which operations survive the route, and the registry predicate refuses an export that hides supply rows. Therefore L is exact for the omitted source material required by Q, G, D and no broader host supply. □

The Lean `TasteGate.lean` development for $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$ is committed to explicit `Nontrivial ExternalSupplySignatureSealUp` and `FieldFaithful ExternalSupplySignatureSealUp` instances in `BEDC.Derived.ExternalSupplySignatureSealUp`. These instances must preserve the row order $O, M, Q, G, D, L, H, C, P, N$ and keep the external supply, physical signature, GAP socket, descent, ledger, and local NameCert rows separated.

Closure status for $\mathsf{NameCert}_{\mathsf{ExternalSupplySignatureSeal}^{\uparrow}}$.

Constructive story: An $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$ packet is generated from an ontic signature source, a mathematical representation, a physical signature row, a GAP socket, a descent certificate, a hidden-supply ledger, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExternalSupplySignatureSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1348.1, Definition 1348.2, Definition 1348.3, and Definition 1348.4 bind the external-supply signature seal as a finite GAP, descent, ledger, and NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExternalSupplySignatureSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim physical truth, full representation adequacy, registry finality, public closure, bridge checking, or Lean verification for $\mathsf{ExternalSupplySignatureSeal}^{\uparrow}$.

Upgrade path: Lean closure requires Nontrivial ExternalSupplySignatureSealUp and FieldFaithful ExternalSupplySignatureSealUp instances in `lean4/BEDC/Derived/ExternalSupplySignatureSealUp/TasteGate.lean`.

Chapter 1349

A Concrete Naming Certificate for $\mathsf{ClosedGenerationRefusal}^{\uparrow}$

The $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ packet realizes the negative boundary of the inscription-event vision in chapter 9397. It sits beside the public inscription event of chapter 519, the route audit of chapter 1005, and the kernel observation sieve of chapter 932. Those siblings expose accepted public inscription rows, accepted route rows, and accepted-or-rejected observation rows; this packet records the finite case in which a candidate transition is refused as closed-generation-certified from the prior records.

Definition 1349.1 (ClosedGenerationRefusal carrier)**.** *A* $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, T, G, R, I, H, C, Q, N)$$

with the following displayed rows:

1. *P is the prior-record row naming the finite records visible before the candidate transition is tested;*
2. *T is the candidate-transition row whose inscription-facing admission is under audit;*
3. *G is the closed-generation test row, recording whether the prior records certify T by authorized generators;*
4. *R is the refusal-ledger row, present exactly when G does not certify T as prior-record generated;*
5. *I is the inscription-comparison row, reading T against the inscription-event packet of chapter 519;*
6. *H records componentwise* hsame *transport for P, T, G, R, I;*
7. *C records displayed* Cont *replay through the prior records, candidate transition, closed-generation test, refusal ledger, and inscription comparison;*
8. *Q is the* Pkg *provenance over* BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, $\mathsf{KernelObservationSieve}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, *and* $\mathsf{ApophaticSocket}^{\uparrow}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedGenerationRefusal}^{\uparrow}}$ *row.*

The classifier has no coordinate for a transition accepted by the closed-generation test, an untested candidate, a hidden generator, a private agency predicate, or a route that bypasses the refusal ledger.

Lean4 ✓ BEDC.Derived.ClosedGenerationRefusalUp.TasteGate.closedGenerationRefusalTasteGate
single carrier alignment

Theorem 1349.2 (ClosedGenerationRefusal NameCert obligations)**.** *For the carrier of Definition 1349.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedGenerationRefusal}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$F = (P, T, G, R, I, H, C, Q, N),$$

prior-record exposure through P*, candidate-transition exposure through* T*, closed-generation testing through* G*, refusal-ledger accountability through* R*, inscription-event comparison through* I*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*, package provenance through* Q*, and local naming through* N*.*

Proof. Project the displayed carrier. The rows $P, T, G, R, I, H, C, Q, N$ are the full inventory available to the certificate reader, and the classifier lists no extra coordinate for an untested candidate or hidden generator.

Read the refusal surface in order. The prior-record row P supplies the finite source, the candidate row T supplies the transition under audit, and the test row G is the only displayed place where prior-record closed generation may certify the candidate.

Finally inspect R, I, H, C, Q, N. The refusal ledger records the negative result of G, the inscription comparison I connects the candidate to the inscription-event sibling, and the support rows only transport, replay, package, and name the same finite packet. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 1349.3 (ClosedGenerationRefusal inscription separation)**.** *An accepted* $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ *packet separates the candidate transition* T *from prior-record closed generation without treating* T *as a private agency object. The row* R *records only that* G *does not certify* T *from* P*, while* I *keeps the public inscription comparison routed through chapter 519 and chapter 1005.*

Proof. Begin with P, T, G. The candidate transition is tested against the displayed prior records by the closed-generation row, so the refusal is not a claim about an unrecorded psychological state or host process.

Use the refusal ledger R. Its content is negative and local: the candidate is not certified by the displayed prior-record test. The carrier gives no coordinate for replacing that local refusal with an external agency object.

The inscription-comparison row I then routes the same candidate through the public inscription-event and route-audit siblings. Thus the packet separates closed-generation refusal from inscription-facing comparison while keeping both rows inside one finite certificate. □

Theorem 1349.4 (ClosedGenerationRefusal ledger non-escape)**.** *No consumer of an accepted* $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ *packet can export the candidate transition as inscription-event novelty while omitting the refusal-ledger row* R*. A consumer may read the prior records, candidate, test, and inscription comparison through* P, T, G, I, H, C, Q, N*, but the negative closed-generation boundary remains public only through* R*.*

Proof. Project the carrier $F = (P, T, G, R, I, H, C, Q, N)$. The negative closed-generation boundary is named only in R, while P, T, G name the test surface and I names the inscription comparison.

Apply Definition 1349.2. Refusal accountability is one of the local obligations, and the support rows H, C, Q, N preserve the same displayed packet rather than adding a second route around R.

Consequently a consumer that omits R may still see a prior-record test and candidate row, but it has not read the certificate row that makes the negative closed-generation boundary public. □

Theorem 1349.5 (ClosedGenerationRefusal sibling independence)**.** *The* $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ *carrier is not bijective, under NameCert-row preservation, to* $\mathsf{InscriptionEvent}^{\uparrow}$, $\mathsf{InscriptionRouteAudit}^{\uparrow}$, $\mathsf{KernelObservationSieve}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, *or* $\mathsf{ApophaticSocket}^{\uparrow}$. *It uniquely binds the prior-record row P, candidate-transition row T, closed-generation test G, refusal ledger R, and inscription comparison I as one packet.*

Proof. Compare the inscription siblings first. $\mathsf{InscriptionEvent}^{\uparrow}$ records an accepted public inscription packet, and $\mathsf{InscriptionRouteAudit}^{\uparrow}$ records a visible gap-to-accepted-route audit; neither carries the prior-record closed-generation test together with the refusal ledger.

Compare the sieve and commitment siblings next. $\mathsf{KernelObservationSieve}^{\uparrow}$ has accepted and rejected observation ledgers, and $\mathsf{CausalCommitment}^{\uparrow}$ records commitment-facing causal rows, but neither preserves the candidate transition, closed-generation test, and inscription comparison as one local NameCert surface.

Finally compare $\mathsf{ApophaticSocket}^{\uparrow}$. It records a supply-facing refusal socket, not the prior-record test of a candidate transition. Therefore no NameCert-row-preserving bijection can preserve $P, T, G, R, I, H, C, Q, N$ across any of the listed siblings. □

Falsifiable prediction: A packet whose closed-generation test row accepts the candidate transition cannot inhabit $\mathsf{ClosedGenerationRefusal}^{\uparrow}$.

Independence witness: inscriptionevent, inscriptionrouteaudit, kernelobservationsieve, causalcommitment, apophaticsocket: the refusal packet has the prior-record row, candidate-transition row, closed-generation test, refusal ledger, inscription comparison, transport, replay, provenance, and local NameCert rows; each sibling lacks at least the closed-generation test with its refusal ledger, so no NameCert-row-preserving bijection exists.

The Lean carrier file `lean4/BEDC/Derived/ClosedGenerationRefusalUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{ClosedGenerationRefusal}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ClosedGenerationRefusal}^{\uparrow}$ instances for $P, T, G, R, I, H, C, Q, N$ before the taste-gate target BEDC.Derived.ClosedGenerationRefusalUp.TasteGa can be cited as checked.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedGenerationRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedGenerationRefusal}^{\uparrow}$ packet is generated from prior records, a candidate transition, a closed-generation test, a refusal ledger, an inscription comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedGenerationRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1349.1, Definition 1349.2, Definition 1349.3, Definition 1349.4, and Definition 1349.5 specify the finite prior-record closed-generation refusal packet and its inscription-facing comparison boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedGenerationRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, a global agency predicate, hidden generator completeness, untested transition admission, or positive inscription-event acceptance.

Upgrade path: Obligation closure requires checked carrier admission, closed-generation refusal accountability, inscription comparison, ledger non-escape, nontriviality, and field faithfulness for $P, T, G, R, I, H, C, Q, N$.

Chapter 1350

A Concrete Naming Certificate for $\mathsf{TraditionLandingLedger}^{\uparrow}$

The $\mathsf{TraditionLandingLedger}^{\uparrow}$ packet realizes the structural translation discipline of section 9462.29 as a finite landing ledger. It sits beside the comparison boundary of chapter 1268, the tradition registry of chapter 1267, the tradition-science bridge of chapter 1270, the claim registry layer of chapter 1209, and the registry export surface of chapter 1216. Those siblings classify comparison, register inherited entries, bridge tradition with science-roadmap rows, admit claims, and expose registry exports; this packet binds the source tradition, BEDC landing, rejected surplus, structural distinction, transport, replay, provenance, and local name as one landing certificate.

Definition 1350.1 (TraditionLandingLedger carrier)**.** *A* $\mathsf{TraditionLandingLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, R, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-tradition row, recording the inherited tradition, text family, source question, vocabulary surface, and comparison prompt.*
2. *T is the BEDC landing row, recording the carrier, relation, classifier, ledger, certificate, gap, or registry object onto which the structural reading lands.*
3. *R is the rejected-surplus row, recording the doctrinal, metaphysical, host-semantic, identity, or authority claim not imported by the landing.*
4. *D is the structural-distinction row, recording that the packet performs structural translation rather than doctrinal identity.*
5. *H records componentwise* hsame *transport for S, T, R, D and the support rows below.*
6. *C records displayed* Cont *replay from the source row through the BEDC landing row and rejected-surplus inspection.*
7. *P is the* Pkg *provenance over* $\mathsf{TraditionComparisonBoundary}^{\uparrow}$, $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TraditionLandingLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for inherited doctrinal truth, tradition membership alone, host-semantic identity, registry export without surplus inspection, or a claim that the source tradition and BEDC landing are the same object.

Theorem 1350.2 (TraditionLandingLedger NameCert obligations)**.** *For the carrier of Definition 1350.1, the local* $\mathsf{NameCert}_{\mathsf{TraditionLandingLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for* $L = (S, T, R, D, H, C, P, N)$*; source-tradition exposure through* S*; BEDC landing exposure through* T*; rejected-surplus exposure through* R*; structural-distinction exactness through* D*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*; package provenance through* P*; and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.TraditionLandingLedgerUp.TraditionLandingLedgerTasteGate single carrier alignment`

Proof. Project the finite packet of Definition 1350.1. The rows S, T, R, D are the only content rows available to the landing ledger, so source tradition, BEDC landing, rejected surplus, and structural distinction must be exposed before the local name can be read.

Use the sibling comparison boundary chapter 1268. It already separates structural landing from doctrinal identity; the present ledger records the landing as a reusable certificate by keeping T and R in the same packet.

Finally read H, C, P, N. These rows transport, replay, package, and name the displayed content rows through $\mathsf{TraditionRegistry}^{\uparrow}$, $\mathsf{TraditionScienceRegistryBridge}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, and the kernel rows. They do not add a hidden doctrine or identity coordinate, so the NameCert obligations are exhausted by the displayed packet. □

Theorem 1350.3 (TraditionLandingLedger nonidentity)**.** *No accepted* $\mathsf{TraditionLandingLedger}^{\uparrow}$ *packet identifies a source tradition with its BEDC landing. Every accepted packet must display both the rejected-surplus row* R *and the structural-distinction row* D *before the landing row* T *may be exported through the registry-facing provenance row* P*.*

Proof. Let an accepted packet be

$$L = (S, T, R, D, H, C, P, N).$$

The landing row T records where the structural reading lands; it is not a proof that the source tradition row S is identical to the BEDC object named by T.

The rejected-surplus row R records what is not imported, and the structural-distinction row D records that the landing is a translation boundary. Removing either row leaves the carrier of Definition 1350.1, so the packet no longer distinguishes structural translation from doctrinal identity.

The support rows H, C, P, N preserve the same separation through transport, replay, provenance, and local naming. Therefore a registry consumer may export the landing only as a finite structural-translation ledger, not as an identity claim. □

Falsifiable prediction: If the rejected-surplus row is removed, a tradition landing packet can no longer distinguish structural translation from doctrinal identity.

Independence witness: traditioncomparisonboundary, traditionregistry, traditionscienceregistry-bridge, claimregistrylayer, registryexportsurface: no sibling is bijective to $\mathsf{TraditionLandingLedger}^{\uparrow}$ because each lacks at least one landing-plus-surplus pair. The comparison boundary governs comparison acceptance, the tradition registry records admitted inherited entries, the tradition-science bridge joins tradition and science rows, the claim registry controls admission, and the registry export surface exposes public export; none preserves S, T, R, D, H, C, P, N as one landing ledger.

The Lean follow-up must provide Nontrivial TraditionLandingLedger$^{\uparrow}$ with an explicit witness pair and FieldFaithful TraditionLandingLedger$^{\uparrow}$ preserving S, T, R, D, H, C, P, N as separate displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TraditionLandingLedger}^{\uparrow}}$.

Constructive story: A TraditionLandingLedger$^{\uparrow}$ packet is generated from source-tradition, BEDC landing, rejected-surplus, structural-distinction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TraditionLandingLedger}^{\uparrow}}$, seed)

Scope closed: Definition 1350.1, Definition 1350.2, and Definition 1350.3 specify the finite landing ledger and its nonidentity boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TraditionLandingLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, doctrinal truth of any inherited tradition, identity between source tradition and BEDC landing, or public registry export closure.

Upgrade path: Lean closure requires Nontrivial TraditionLandingLedgerUp and FieldFaithful TraditionLandingLedgerUp instances preserving the source, landing, rejected-surplus, distinction, transport, replay, provenance, and local naming rows.

Chapter 1351

A Concrete Naming Certificate for $\mathsf{DiagonalConvergenceBound}^{\uparrow}$

The $\mathsf{DiagonalConvergenceBound}^{\uparrow}$ packet realizes the explicit convergence estimate in Definition 9238.5 as a finite naming certificate. It consumes the Real-completion horizon of chapter 9238, the diagonal representative of chapter 1205, and the three-leg estimate surface of chapter 466. Its readback route is anchored by chapter 126, chapter 301, chapter 291, and chapter 38.

The packet names the row carrying the displayed bound

$$|X_n - L| \leq \frac{2}{n+1}.$$

It is a local NameCert surface for the convergence-bound row; it is not a host-limit choice, a quotient-stream equality, or an ambient completeness principle.

Definition 1351.1 (Diagonal convergence-bound carrier)**.** *A* $\mathsf{DiagonalConvergenceBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, \mu, n, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the regular-Cauchy family row whose convergence to the diagonal representative is being bounded;*
2. *μ is the modulus row used to select the diagonal comparison scale for the request n;*
3. *n is the displayed precision request;*
4. *D is the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *row exposing the selected diagonal representative L at the request determined by μ and n;*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the arithmetic row for the bound $2/(n+1)$;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row comparing X_n with the selected diagonal representative read;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the representative readback and tolerance ledger have both been displayed;*
8. *H records componentwise* hsame *transport for the family, modulus, request, representative, tolerance ledger, readback, seal, provenance, and local name rows;*
9. *C records displayed* Cont *replay from the modulus request through the diagonal representative, dyadic tolerance ledger, regular-rational readback, and Real seal;*

10. *P is the* Pkg *provenance over* DiagonalConvergenceBound$^{\uparrow}$, DiagonalLimitRepresentative$^{\uparrow}$, RegularCauchyTriangleE StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{DiagonalConvergenceBound}^{\uparrow}}$ *row.*

The carrier exports no selected host limit, quotient equality of streams, hidden family choice, ambient Real$^{\uparrow}$ *completeness field, or detached convergence proof.*

Lean4 ✓ `BEDC.Derived.DiagonalConvergenceBoundUp.TasteGate.DiagonalConvergenceBoundTasteGate`
`single carrier alignment`

Definition 1351.2 (Diagonal convergence-bound classifier)**.** *Two accepted* DiagonalConvergenceBound$^{\uparrow}$ *carriers are classified together exactly when their family rows, modulus rows, precision requests, diagonal representative rows, dyadic tolerance ledgers for $2/(n+1)$,* RegSeqRat$^{\uparrow}$ *readbacks,* Real$^{\uparrow}$ *seals, transport rows, continuation rows, package rows, and local naming rows are related componentwise by the recorded* hsame *comparisons and* Cont *routes. The classifier compares only this displayed convergence-bound surface.*

Theorem 1351.3 (Diagonal convergence-bound NameCert obligations)**.** *For the carrier of Definition 1351.1, the local* NameCert$_{\text{DiagonalConvergenceBound}^{\uparrow}}$ *surface consists exactly of carrier admission, family-row exposure through X, modulus-request exposure through μ, n, diagonal-representative exposure through D, dyadic tolerance exposure through Q, regular-rational readback through R, Real seal routing through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DiagonalConvergenceBoundUp.DiagonalConvergenceBoundNameCertObligations`

Proof. Project the packet B. The carrier lists the family row, modulus row, request row, diagonal representative row, dyadic tolerance ledger, regular-rational readback, Real seal, transport, continuation, provenance, and local naming row.

The bound row is finite. The request n and modulus row μ select the diagonal representative read in D, while Q records the displayed arithmetic budget $2/(n+1)$. The readback row R is then the only row comparing the requested family entry with the representative read.

The remaining rows E, H, C, P, N do not add analytic data. They seal, transport, replay, source, and name the already displayed packet, so the listed obligations exhaust the local certificate surface. □

Theorem 1351.4 (Diagonal convergence-bound nonchoice boundary)**.** *An accepted* DiagonalConvergenceBound$^{\uparrow}$ *packet exports exactly the displayed family row X, modulus row μ, request row n, diagonal representative row D, dyadic tolerance ledger Q,* RegSeqRat$^{\uparrow}$ *readback R,* Real$^{\uparrow}$ *seal row E, componentwise transport H, continuation replay C, package provenance P, and local naming row N. It exports no selected host limit, quotient-stream equality, countable choice row, hidden tail selector, ambient* Real$^{\uparrow}$ *completeness proof, or off-packet convergence witness.*

Proof. Inspect the coordinates of Definition 1351.1. Every coordinate is either a displayed BEDC row or structural support for transport, replay, provenance, and naming.

Definition 1351.3 accounts for all admissible carrier and classifier reads. The classifier of Definition 1351.2 then restricts comparison to those same displayed rows.

The excluded objects require coordinates absent from the carrier. Therefore they cannot be exported by the local naming certificate or recovered by a consumer of the displayed convergence-bound row. □

Theorem 1351.5 (Diagonal convergence-bound Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DiagonalConvergenceBound}^{\uparrow}$ *carrier reads the finite route*

$$X, \mu, n, D, Q, R, E, H, C, P, N$$

before treating the convergence estimate

$$|X_n - L| \leq \frac{2}{n+1}$$

as public. The handoff supplies the bound row needed by the Real-completion limit theorem, but it does not by itself supply existence, uniqueness, quotient equality, or an ambient completeness principle.

Proof. Start from the local NameCert obligations. They force every accepted carrier to display the regular-Cauchy family, the modulus request, the diagonal representative, the dyadic tolerance ledger, and the regular-rational readback before the Real-facing seal may be consumed.

Next use the dependency route. The representative row D is the consumer of chapter 1205; the tolerance ledger Q and readback row R are anchored by chapter 291 and chapter 301; and the three-leg comparison surface is supplied by chapter 466.

Finally read the seal row E through H, C, P, N. By Definition 1351.4, no host limit, quotient stream, hidden selector, or ambient completeness coordinate is available. Thus the packet hands only the explicit convergence-bound row to the Real-completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalConvergenceBound}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalConvergenceBound}^{\uparrow}$ packet is generated from a displayed family row, modulus row, request row, diagonal representative row, dyadic-tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming. The classifier bounds diagonal convergence by reading the request and modulus before it inspects the representative, tolerance, readback, and seal rows. The ledger story is finite diagonal control: the packet records exactly which family, modulus, and request generate the bound, while transport preserves those rows and prevents selected host limits, quotient equality, hidden family choice, ambient completeness, or detached convergence proofs from entering the accepted surface.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalConvergenceBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier and classifier definitions, NameCert-obligation, nonchoice-boundary, Real-completion-handoff, and existing closure annotations for $\mathsf{DiagonalConvergenceBound}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalConvergenceBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim selected host limits, quotient equality of streams, hidden family choice, ambient RealUp completeness fields, or detached convergence proofs.

Upgrade path: To reach obligationClosure, the diagonal bound classifier and nonchoice handoff must be encoded as checked finite convergence obligations.

Chapter 1352

A Concrete Naming Certificate for $\mathsf{RealCofinalModulusGate}^{\uparrow}$

The $\mathsf{RealCofinalModulusGate}^{\uparrow}$ packet packages the finite gate that a Real-completeness limit consumer must read before treating a $\mathsf{Real}^{\uparrow}$ seal as supported by cofinal modulus data. It sits after the tail budget of chapter 1231, the normalization route of chapter 1264, and the threshold normal form of chapter 1299. Its terminal readback is anchored by chapter 38 and chapter 301, so the gate exposes the finite cofinal schedule and comparison rows rather than allowing a Real seal to stand alone.

Definition 1352.1 (Real cofinal modulus gate carrier)**.** *A* $\mathsf{RealCofinalModulusGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, M, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the tail-cofinality schedule row consumed from the* $\mathsf{TailCofinalityBudget}^{\uparrow}$ *surface.*
2. *M is the cofinal modulus-comparison row normalized through* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$*.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window whose readback is indexed by the displayed tail schedule and modulus comparison.*
4. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row read through the same window.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after T, M, W, S have been displayed.*
6. *E is the exactness ledger recording that the Real seal read is the one routed through the displayed cofinal tail and modulus rows.*
7. *H stores componentwise* hsame *transport for the tail schedule, modulus comparison, window, regular-rational witness, Real seal, exactness, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from the cofinal tail schedule through the modulus comparison, StreamName window, RegSeqRat witness, Real seal, exactness ledger, provenance, and local naming row.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCofinalModulusGate}^{\uparrow}$*,* $\mathsf{TailCofinalityBudget}^{\uparrow}$*,* $\mathsf{CofinalModulusNormalizationSeal}^{\uparrow}$*,* $\mathsf{CauchyTailThresholdNormalizer}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCofinalModulusGate}^{\uparrow}}$ *row.*

The carrier has no coordinate for a bare Real seal, quotient stream equality, host-selected limit, private tail threshold, untracked modulus conversion, or ambient completeness field.

Definition 1352.2 (Real cofinal modulus gate classifier)**.** *Two accepted* $\mathsf{RealCofinalModulusGate}^{\uparrow}$ *carriers classify together exactly when their tail-cofinality schedules, cofinal modulus-comparison rows, StreamName windows, RegSeqRat witnesses, Real seal rows, exactness ledgers, componentwise transports, displayed continuation replays, package provenance rows, and local naming rows agree componentwise through the recorded* hsame *comparisons and* Cont *routes. The classifier compares the ten-row gate packet only.*

Theorem 1352.3 (Real cofinal modulus gate NameCert obligations)**.** *For the carrier of Definition 1352.1, the local* $\mathsf{NameCert}_{\mathsf{RealCofinalModulusGate}^{\uparrow}}$ *surface consists exactly of carrier exposure through G, cofinal-tail exactness through T, modulus-comparison exposure through M, window-readback stability through W, S, RealUp seal preservation through R, E, componentwise transport through H, displayed continuation replay through C, package provenance through P, local naming through N, and ledger non-escape for bare Real seals, quotient stream equality, host-selected limits, private tail thresholds, untracked modulus conversion, and ambient completeness fields.*

Lean4 ✓ `BEDC.Derived.RealCofinalModulusGateUp.TasteGate.RealCofinalModulusGateTasteGate single carrier alignment`

Proof. Project the packet G. The carrier displays the tail schedule, modulus comparison, StreamName window, RegSeqRat witness, Real seal, exactness ledger, transport, continuation, provenance, and local naming row.

The analytic readback is finite. The rows T and M determine the cofinal tail and comparison budget; W and S are read through that same displayed budget before the terminal Real seal row R appears.

The remaining rows do not add a second analytic source. The exactness ledger E records that R is the seal reached through T, M, W, S, while H, C, P, N transport, replay, source, and name the displayed route. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the local certificate surface. □

Theorem 1352.4 (Real cofinal modulus gate tail-cofinality exactness)**.** *A consumer of an accepted* $\mathsf{RealCofinalModulusGate}^{\uparrow}$ *packet cannot read the displayed* $\mathsf{Real}^{\uparrow}$ *seal row R without also reading the cofinal tail schedule T and the modulus-comparison row M. Equivalently, the Real seal route factors through*

$$T \longmapsto M \longmapsto W \longmapsto S \longmapsto R \longmapsto (E, H, C, P, N).$$

Proof. Start with the carrier exposure in Definition 1352.3. It places the tail schedule T and modulus comparison M before the window W, witness S, and Real seal R.

Window-readback stability then fixes the middle of the route. The StreamName window and RegSeqRat witness are indexed by the displayed cofinal data, so a consumer cannot substitute a private tail or modulus row between M and R.

Finally read the exactness ledger. The row E records that the seal is the one routed through T, M, W, S, while H, C, P, N only transport, replay, source, and name that route. Thus reducing the packet to R alone loses data that the NameCert obligations require before the Real seal can be consumed. □

Falsifiable prediction: An accepted $\mathsf{RealCofinalModulusGate}^{\uparrow}$ packet cannot be reduced to a RealUp seal alone without losing either the cofinal tail schedule or the modulus-comparison row.

Independence witness: tailcofinalitybudget, cofinalmodulusnormalizationseal, cauchytailthresholdnormalizer, real, regseqrat: not bijective to the ten-row gate packet because each listed sibling supplies only a source, normalization, threshold, seal, or witness surface, while the gate binds all ten rows together with exactness, transport, replay, provenance, and local naming.

Lean follow-up provides explicit Nontrivial $\mathsf{RealCofinalModulusGate}^{\uparrow}$ and FieldFaithful $\mathsf{RealCofinalModulusGate}^{\uparrow}$ instances in `lean4/BEDC/Derived/RealCofinalModulusGateUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCofinalModulusGate}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCofinalModulusGate}^{\uparrow}$ packet is generated from a RealUp seal, tail-threshold row, cofinal-modulus row, tail-agreement row, componentwise hsame transport, displayed Cont replay, package provenance, ledger rows, and local naming. The classifier opens the gate only when the Real seal is tied to a displayed tail threshold and a cofinal modulus conversion. The ledger story is cofinal access without ambient completeness: every tail-facing consumer reads the same modulus and agreement rows through the finite carrier, so a bare Real seal, private threshold, quotient stream, host-selected limit, or untracked conversion cannot authorize the route.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCofinalModulusGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier and classifier definitions, NameCert-obligation, tail-cofinality-exactness, Lean-follow-up boundary, and existing closure annotation for $\mathsf{RealCofinalModulusGate}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCofinalModulusGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not claim bare Real seals, quotient stream equality, host-selected limits, private tail thresholds, untracked modulus conversion, or ambient completeness fields.

Upgrade path: To reach obligationClosure, the cofinal-modulus gate and tail-cofinality exactness must be checked as finite threshold and transport obligations.

Chapter 1353

A Concrete Naming Certificate for $\mathsf{ProofPatternClassifier}^{\uparrow}$

The $\mathsf{ProofPatternClassifier}^{\uparrow}$ packet realizes the proof-pattern discipline of subsection 9462.25.1, subsection 9462.25.2, and subsection 9462.25.3 as finite NameCert data. It sits beside the proof witness chain of chapter 1238, the forward-binding GAP ledger of chapter 1262, the GAP socket boundary of chapter 1343, the registry no-smuggling predicate of chapter 1344, and the typed failure certificate of chapter 1237. Those siblings expose proof traces, forward commitments, missing-supply boundaries, export refusal, and typed failure status separately; this packet classifies proof-pattern routes only when positive witness construction, local refutation, decidable double-negation, and prohibited positive extraction remain separated.

Definition 1353.1 (Proof-pattern classifier carrier)**.** *A* $\mathsf{ProofPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (W, L, D, R, E, G, H, C, Q, N)$$

with the following displayed rows:

1. *W is the proof-witness-chain row, recording the accepted finite proof trace, source carrier, classifier witness, ledger, refusal, audit verdict, provenance, and local name consumed from* $\mathsf{ProofWitnessChain}^{\uparrow}$*;*
2. *L is the local carrier and classifier context, recording the fixed object row, predicate row, classifier row, and local certificate surface against which the proof pattern is read;*
3. *D is the displayed decidability evidence, present exactly when a local* $\mathsf{Decidable}(P(h))$ *route is available for the named predicate and history row;*
4. *R is the local refutation and inversion evidence, recording carrier no-confusion, classifier inversion, certificate-field contradiction, and typed failure rows;*
5. *E is the forbidden positive-extraction refusal row, blocking extraction of existential or positive witness content from $\neg\forall h, \neg P(h)$, undecided $\neg\neg P(h)$, arbitrary excluded middle, quotient collapse, host equality, or hidden choice;*
6. *G is the GAP row for forward-binding commitments, recording any unsupported universal, future, or missing-supply component through* $\mathsf{ForwardBindingGapLedger}^{\uparrow}$ *and* $\mathsf{GapSocketBoundary}^{\uparrow}$ *rather than as proof content;*
7. *H records componentwise* hsame *transport for proof witness, local classifier, decidability, refutation, refusal, GAP, continuation, provenance, and local naming rows;*

8. *C records displayed* Cont *replay for positive witness construction, local refutation, decidable double-negation, positive-extraction refusal, GAP handoff, and downstream audit;*
9. *Q records* Pkg *provenance over* ProofPatternClassifier$^{\uparrow}$, ProofWitnessChain$^{\uparrow}$, ForwardBindingGapLedger$^{\uparrow}$, GapSocketBoundary$^{\uparrow}$, RegistryNoSmugglingPredicate$^{\uparrow}$, TypedFailureCertificate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{ProofPatternClassifier}^{\uparrow}}$ *row.*

The carrier exports no route from refutation to positive content, no double-negation eliminator without D, no witness for an existential claim without a displayed witness row in W or L, and no registry export that bypasses E, G, Q, N.

Lean4 ✓ `BEDC.Derived.ProofPatternClassifierUp.ProofPatternClassifierTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.ProofPatternClassifierUp.ProofPatternClassifierUp`

Theorem 1353.2 (Proof-pattern classifier NameCert obligations)**.** *For the carrier of Definition 1353.1, the local* NameCert$_{\text{ProofPatternClassifier}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier exposure for $P = (W, L, D, R, E, G, H, C, Q, N)$;*
2. *classifier stability for the proof-witness row W against the local carrier and classifier context L, preserved under H and C;*
3. *local-refutation boundary through R, with typed failure and certificate-field contradiction kept inside the displayed carrier and classifier rows;*
4. *decidable double-negation gate through D, allowing $\neg\neg P(h)$ to be consumed as positive content only when the same local predicate row has displayed decidability;*
5. *positive-extraction refusal through E, G, Q, N, blocking positive witness export from negative or undecided premises and preserving the GAP and no-smuggling audit route.*

Proof. Project the packet of Definition 1353.1. The local certificate can read only $W, L, D, R, E, G, H, C, Q, N$, so carrier exposure fixes the proof witness, classifier context, decision row, refutation row, refusal row, GAP row, transport, replay, provenance, and local name.

Classifier stability is then read through W, L, H, C. The proof witness row supplies the finite proof chain, the local row supplies the predicate and classifier surface, and transport plus replay preserve that same pairing for downstream use.

The remaining obligations separate the three proof-pattern exits. The refutation route uses R only as local inversion against displayed rows; the double-negation route uses D only as displayed decidability; and the refusal route uses E, G, Q, N to prevent missing witnesses or unsupported universal commitments from becoming positive proof content. □

Theorem 1353.3 (Proof-pattern classifier local-refutation boundary)**.** *An accepted* ProofPatternClassifier$^{\uparrow}$ *packet accepts a negative proof only when its refutation factors through the displayed carrier row L, proof-witness row W, local inversion row R, transport row H, replay row C, provenance row Q, and local NameCert row N. A contradiction that requires a hidden truth predicate, unrecorded classifier crossing, host equality, quotient collapse, or missing certificate field is outside the accepted local-refutation route.*

Proof. Start with the obligation surface of Definition 1353.2. A negative proof is accepted through the local-refutation row, not through a global truth classifier.

The row R must therefore consume a purported witness through the displayed carrier and classifier context L and the proof-witness chain W. Transport and replay are limited to H and C, while provenance and local naming remain Q and N.

If the contradiction uses hidden truth, unrecorded classifier crossing, host equality, quotient collapse, or an absent certificate field, the factorization through L, W, R, H, C, Q, N fails. The packet then records no accepted negative proof; it records either a refusal route through E or an exposed GAP route through G. □

Theorem 1353.4 (Proof-pattern classifier positive-extraction refusal)**.** *No accepted* $\mathsf{ProofPatternClassifier}^{\uparrow}$ *packet can turn* $\neg\forall h, \neg P(h)$ *or undecided* $\neg\neg P(h)$ *into positive content without the displayed decidability row* D *or a displayed witness row in* W *or* L*. In particular, the packet refuses existential export, theorem-evidence export, and local NameCert export for any route whose only positive source is negative pressure, undecided double negation, arbitrary excluded middle, or a hidden registry entry.*

Proof. Read the carrier rows. Positive content must come from a witness already displayed in W or L, or from a decidable double-negation route whose decision evidence is displayed in D.

The premise $\neg\forall h, \neg P(h)$ supplies no named h and no row witnessing $P(h)$. The premise $\neg\neg P(h)$ supplies no constructor for $P(h)$ unless the same local predicate row is accompanied by D.

The refusal row E and GAP row G record exactly these missing-source cases. Provenance and local naming through Q, N preserve the refusal, while the registry no-smuggling sibling prevents a hidden entry from being exported as theorem evidence. Hence the accepted packet cannot convert negative pressure or undecided double negation into positive content. □

Falsifiable prediction: If the displayed decidability row is removed, the packet cannot export double-negation elimination as an accepted proof-pattern route.

Independence witness: proofwitnesschain, forwardbindinggapledger, gapsocketboundary, registrynosmugglingpredicate, typedfailurecertificate: none is bijective to $\mathsf{ProofPatternClassifier}^{\uparrow}$. ProofWitnessChainUp carries a proof trace without the four-way classifier boundary; ForwardBindingGapLedgerUp carries future and unsupported commitments without local proof-pattern classification; GapSocketBoundaryUp exposes missing supply without deciding proof routes; RegistryNoSmugglingPredicateUp blocks registry export without classifying local refutation and decidable double negation; TypedFailureCertificateUp records typed failure without the positive-witness and positive-extraction refusal split.

The Lean follow-up must provide Nontrivial $\mathsf{ProofPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{ProofPatternClassifier}^{\uparrow}$ instances in `lean4/BEDC/Derived/ProofPatternClassifierUp/TasteGate.lean`. The field-faithfulness witness must preserve $W, L, D, R, E, G, H, C, Q, N$ as distinct rows before downstream audit, GAP, no-smuggling, or proof-pattern consumers cite BEDC.Derived.ProofPatternClassifierUp.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofPatternClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofPatternClassifier}^{\uparrow}$ packet is generated from proof-witness-chain, local carrier and classifier, displayed decidability, local refutation, positive-extraction refusal, GAP, transport, continuation, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofPatternClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1353.1, Definition 1353.2, Definition 1353.3, and Definition 1353.4 bind the finite classifier packet, its naming obligations, its local refutation boundary, and its refusal of unwitnessed positive extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofPatternClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not supply Lean verification, bridge checking, global excluded middle, a truth predicate, unwitnessed existential content, hidden registry theorem evidence, or positive extraction from undecided double negation for ProofPatternClassifier$^{\uparrow}$.

Upgrade path: Lean closure requires Nontrivial ProofPatternClassifierUp and FieldFaithful ProofPatternClassifierUp instances.

Chapter 1354

A Concrete Naming Certificate for $\mathsf{CanonicalTailChoice}^{\uparrow}$

The $\mathsf{CanonicalTailChoice}^{\uparrow}$ packet realizes the finite tail-choice route needed by the Real-completeness limit theorem of chapter 9238. It sits after the no-choice boundary of chapter 1173, the shared tail schedule of chapter 681, the bounded prefix budget of chapter 919, and the exact completion boundary of chapter 1120. Its role is to expose the canonical finite index and tail-window witness used before the diagonal seal is read, without importing a host-selected subsequence, quotient stream equality, countable choice row, or ambient completeness principle.

Falsifiable prediction: A Real-completion consumer that omits the canonical finite tail-choice row cannot claim the diagonal seal handoff.

Independence witness: cauchychoiceboundary, tailcofinalityschedule, finiteprefixlimitbudget, realcompletionexactboundary, limitselector: none is bijective to $\mathsf{CanonicalTailChoice}^{\uparrow}$ because this packet simultaneously carries the modulus row, tolerance-budget row, canonical finite index row, tail-window witness row, Real seal route, no-choice refusal row, componentwise transport, continuation replay, provenance, and local naming surface; the named siblings isolate boundary refusal, cofinal scheduling, prefix-budget exhaustion, exact boundary faces, or selector admission without this joint carrier.

Definition 1354.1 (Canonical tail-choice carrier)**.** *A* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, E, I, T, S, R, H, C_0, P, N)$$

with the following displayed rows:

1. *M is the modulus row read from the accepted regular-Cauchy route;*
2. *E is the tolerance-budget row fixing the finite precision scale served by the packet;*
3. *I is the canonical finite index row computed from M and E, not from an external choice schedule;*
4. *T is the tail-window witness row certifying that every downstream observation is read inside the finite tail determined by I;*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal route reached only after M, E, I, T have been displayed;*
6. *R is the no-choice refusal row excluding host-selected limits, quotient stream equality, countable choice, hidden tail oracles, and ambient completeness;*

7. *H records componentwise* hsame *transport for the displayed analytic rows and the structural rows;*
8. *C_0 records the displayed* Cont *replay from modulus and budget admission to canonical index, tail-window witness, Real seal, refusal, and local naming;*
9. *P is the* Pkg *provenance over the sibling certificates,* BHist, hsame, Cont, Pkg, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a host-selected subsequence, quotient stream equality, countable choice, metric completeness, ambient completeness, or a detached Real-completion seal.

Lean4 (definition) `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceUp`

Theorem 1354.2 (Canonical tail-choice NameCert obligations)**.** *For every accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier*

$$C = (M, E, I, T, S, R, H, C_0, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$ *surface consists exactly of carrier exposure for C, modulus exposure through M, tolerance-budget exposure through E, canonical finite-index exposure through I, tail-window witness exposure through T,* $\mathsf{Real}^{\uparrow}$ *seal-route exposure through S, no-choice refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C_0, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier namecert obligations`

Proof. Project the carrier of Definition 1354.1. Its displayed rows list the modulus, tolerance budget, canonical finite index, tail-window witness, seal route, refusal ledger, and structural support.

Read the rows in order. The modulus row M and tolerance-budget row E determine the finite index row I before the tail-window witness T can be consumed. The Real seal route S is therefore downstream from a displayed finite choice, not from a hidden selector.

Finally inspect the structural rows. The rows H, C_0, P, N transport, replay, record provenance for, and name the same finite packet. Since no further coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1354.3 (Canonical tail-choice source obligations)**.** *Every accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *read exposes source rows over* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBounda $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *before the canonical finite index row I or the tail-window witness T can be consumed. These source rows are finite* BHist *entries in the same packet, not references to a selected subsequence, quotient Cauchy name, countable choice row, or ambient completion principle.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier source obligations`

Proof. Project the carrier of Definition 1354.1. The modulus row M and tolerance-budget row E are read before the canonical finite index row I, so the finite-prefix budget and regular-rational source are already visible when the tail window is formed.

Read the sibling boundary rows next. The no-choice boundary of chapter 1173, the exact completion boundary of chapter 1120, and the cofinal schedule of chapter 681 attach to the displayed route before the Real seal route S is admitted.

The refusal row R then excludes the non-source objects. Since H, C_0, P, N only transport, replay, source, and name the same finite packet, no selected limit, quotient Cauchy name, countable choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported by the accepted read. □

Theorem 1354.4 (Canonical tail-choice window stability)**.** *Let $C = (M, E, I, T, S, R, H, C_0, P, N)$ and $C' = (M', E', I', T', S', R', H', C'_0, P', N')$ be accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carriers. If the displayed rows are transported componentwise by the* hsame *data recorded in H and H', then the finite tail-window route M, E, I, T and M', E', I', T' remain on the same canonical tail-choice surface before either packet reaches its* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Induct over the finite route M, E, I, T. The base rows M, E expose the same modulus and tolerance-budget roles under componentwise hsame transport, so both packets compute their canonical index inside the displayed source surface.

At the index step, the rows I and I' are reached only after those transported source rows. The tail rows T and T' are therefore compared as finite window witnesses attached to the transported index rows, not as host-selected subsequences.

The remaining rows preserve the comparison. The seal rows S, S' are downstream from the tail windows, the refusal rows R, R' exclude quotient names and choice schedules, and C_0, C'_0, P, P', N, N' replay, source, and name the same componentwise transport. Thus finite tail-window stability is exactly the displayed hsame transport of the packet rows. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier window stability`

Theorem 1354.5 (Canonical tail-choice nonescape)**.** *No selected limit, quotient Cauchy name, countable choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle is exported by an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet. Any consumer that claims one of these objects has left the displayed rows $M, E, I, T, S, R, H, C_0, P, N$ and therefore cannot be reading the local* $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$ *surface.*

Proof. Inspect the carrier rows. The tuple exposes the modulus, tolerance budget, canonical index, tail window, Real seal route, refusal ledger, transport, replay, provenance, and local name; it contains no coordinate for a selected limit, quotient Cauchy name, countable choice row, or ambient completeness principle.

Apply the source-obligation theorem above. Every accepted read reaches I and T only after the finite source rows over the no-choice boundary, prefix budget, regular-rational source, cofinal schedule, and exact Real-completion boundary have been displayed.

Finally use the refusal coordinate R. It blocks replacing the finite rows by the excluded objects, while H, C_0, P, N keep all downstream consumption inside the same packet. Hence any consumer that exports an excluded object is outside the accepted $\mathsf{CanonicalTailChoice}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier nonescape`

Theorem 1354.6 (Canonical tail-choice obligation surface)**.** *For every accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier*

$$C = (M, E, I, T, S, R, H, C_0, P, N),$$

the finite tail-choice obligation surface is the simultaneous exposure of the modulus row M, tolerance-budget row E, canonical finite index row I, tail-window witness T, $\mathsf{Real}^{\uparrow}$ *seal route S, no-choice refusal R, componentwise* hsame *transport H, displayed* Cont *replay C_0,* Pkg *provenance P, and local* NameCert *row N. The surface admits no host-selected subsequence, quotient stream equality, countable-choice schedule, hidden tail oracle, detached completion seal, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier obligation surface`

Proof. Project the carrier of Definition 1354.1. The ten displayed rows are exactly $M, E, I, T, S, R, H, C_0, P, N$, so an accepted packet exposes the modulus, tolerance budget, canonical finite index, tail window, seal route, refusal ledger, transport, replay, provenance, and local naming as one finite surface.

Use the schedule factorization of Definition 1354.7. It forces every Real-completion consumer to pass through M and E before reaching I, then through T before the seal route S is consumed. Thus the canonical finite index and tail-window witness are displayed route coordinates, not a host-selected subsequence.

Finally apply the no-choice non-escape theorem, Definition 1354.8. The refusal row R and the structural rows H, C_0, P, N keep the same finite packet visible and exclude quotient stream equality, countable choice, hidden tail oracles, detached completion seals, and ambient completeness. Therefore the listed rows exhaust the obligation surface. □

Theorem 1354.7 (Canonical tail-choice schedule factorization)**.** *Every accepted Real-completion consumer of a* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet factors through the finite route*

$$M, E, I, T, S, R, H, C_0, P, N.$$

When the consumer also reads the sibling surfaces of chapter 681 and chapter 919, their schedule and prefix-budget rows attach before the tail-window witness T and before the Real seal route S.

Proof. Start with the obligation surface of Definition 1354.2. The canonical finite index I is exposed only after M and E, so any downstream schedule must already have the modulus and tolerance budget in scope.

Attach the cofinal schedule. The $\mathsf{TailCofinalitySchedule}^{\uparrow}$ packet supplies its finite window and dyadic threshold rows as an accepted sibling route, and those rows can be consumed only through the same displayed finite schedule surface.

Attach the prefix budget next. The $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ packet records which prefix windows have been spent before a seal-facing read is admitted. Thus the local tail-window witness T is reached only after the sibling schedule and prefix-budget rows have been displayed.

The remaining rows S, R, H, C_0, P, N keep the seal route and refusal ledger inside that same finite packet, so every accepted consumer factors through the stated route. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier schedule factorization`

Theorem 1354.8 (Canonical tail-choice no-choice non-escape)**.** *No host-selected limit, quotient stream equality, countable choice row, hidden tail oracle, metric completeness row, ambient completeness row, or detached Real-completion seal can be read from an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet. The packet exports only the rows $M, E, I, T, S, R, H, C_0, P, N$ of Definition 1354.1.*

Proof. Inspect the accepted carrier. The refusal row R is a displayed coordinate, while none of the excluded objects is a coordinate of the tuple.

Apply Definition 1354.7. Every accepted consumer reaches the Real seal route only through $M, E, I, T, S, R, H, C_0, P, N$, with sibling schedule and prefix-budget rows consumed before the tail-window witness.

The finite index row I is not a host-selected limit or a countable-choice schedule; it is the canonical row displayed after M and E. The witness row T and seal row S are likewise route coordinates, not quotient equalities or completeness principles. Hence any read of an excluded object leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier no choice nonescape`

Theorem 1354.9 (Canonical tail-choice refusal exhaustion)**.** *Every accepted Real-completion consumer of a* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet either factors through the displayed rows*

$$M, E, I, T, S, R, H, C_0, P, N$$

or leaves the local $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$ *surface. In particular, the host-selected limit, quotient stream equality, countable choice, hidden tail oracle, metric completeness, ambient completeness, and detached seal alternatives are exactly the refused alternatives recorded by the packet.*

Proof. Project the carrier of Definition 1354.1. Its only consumer-facing coordinates are the rows $M, E, I, T, S, R, H, C_0, P, N$.

Apply schedule factorization, Definition 1354.7. A Real-completion consumer must pass through the modulus and budget rows before the canonical index, then through the tail-window witness before the seal route is consumed.

Use no-choice non-escape, Definition 1354.8, together with the Real-completion handoff of Definition 1354.10. The refusal row R names exactly the excluded alternatives, while H, C_0, P, N only transport, replay, source, and name the same finite route. Thus any consumer not factoring through the displayed rows has left the local surface. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier refusal exhaustion`

Theorem 1354.10 (Canonical tail-choice Real-completion handoff)**.** *The Real-completion diagonal seal handoff may consume an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet only through the displayed route*

$$M, E, I, T, S, R, H, C_0, P, N.$$

Under chapter 1173 and chapter 1120, this route supplies the finite tail-choice row between no-choice refusal and the terminal exact-boundary seal, and supplies no host-side equality or ambient completeness data.

Proof. Begin with the no-choice boundary. The sibling $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ surface already requires a displayed finite index, tail witness, seal route, and refusal row before a diagonal consumer may claim the no-choice seal path.

Next apply the local obligations and schedule factorization. The $\mathsf{CanonicalTailChoice}^{\uparrow}$ packet identifies the canonical finite index I and tail-window witness T as first-class rows between the modulus-budget pair M, E and the Real seal route S.

Finally read the exact boundary. The $\mathsf{RealCompletionExactBoundaryUp}$ packet admits terminal completion reads only through its displayed StreamName, RegSeqRat, DyadicRatCore, and RealUp faces. Since the present packet exports only $M, E, I, T, S, R, H, C_0, P, N$ and the sibling boundary exports no hidden selector, the diagonal seal handoff remains a finite no-choice route. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier real completion handoff`

Theorem 1354.11 (Canonical tail-choice tail cofinality)**.** *Let* $C = (M, E, I, T, S, R, H, C_0, P, N)$ *be an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier. The canonical finite index row* I *remains cofinal with the displayed* $\mathsf{TailCofinalitySchedule}^{\uparrow}$ *and* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ *rows before any tail-window consumer reads the* $\mathsf{Real}^{\uparrow}$ *seal route* S*. Thus every later finite tolerance request transported through* H *and replayed through* C_0 *is served inside the same tail-window row* T*, without a host-selected subsequence, quotient stream equality, hidden tail oracle, countable-choice row, or ambient completeness principle.*

Proof. Start with schedule factorization, Definition 1354.7. It places the cofinal schedule and prefix-budget rows before the tail-window witness T and before the Real seal route S.

Read the carrier rows $M, E, I, T, S, R, H, C_0, P, N$. The index row I is displayed after the modulus and budget rows, so a later tolerance request can be transported only by reusing the same finite route through H and replaying it through C_0.

The refusal row R blocks replacing that finite route by a selected subsequence, quotient equality, hidden tail oracle, choice row, or ambient completeness principle. Hence the cofinality assertion is internal to the displayed packet before any consumer reaches the seal. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier tail cofinality`

Theorem 1354.12 (Canonical tail-choice Real-completion boundary)**.** *The Real-completion exact-boundary handoff for an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier consumes only the rows*

$$M, E, I, T, S, R, H, C_0, P, N.$$

It cannot read quotient stream equality, countable choice, hidden tail oracles, ambient completeness, or any detached completion seal outside the displayed carrier and its $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$ *surface.*

Proof. Apply the Real-completion handoff theorem Definition 1354.10. The diagonal seal route is reached only through the displayed packet rows $M, E, I, T, S, R, H, C_0, P, N$.

Use the no-choice non-escape theorem Definition 1354.8. It identifies quotient stream equality, countable choice, hidden tail oracles, ambient completeness, and detached seals as absent from the carrier.

The structural rows H, C_0, P, N transport, replay, source, and name the same finite packet. They do not add a new completion coordinate, so the exact-boundary handoff remains confined to the listed rows. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier real completion boundary`

Theorem 1354.13 (Canonical tail-choice tail inhabitation)**.** *Let* $C = (M, E, I, T, S, R, H, C_0, P, N)$ *be an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier. For every finite Real-completion request admitted by the displayed rows* $M, E, I, T, S, R, H, C_0, P, N$*, the tail-window row* T *is inhabited by a finite* BHist *witness after the canonical index row* I *is read. The witness is produced inside the same packet and is not a host-selected subsequence, hidden tail oracle, quotient stream equality, or countable-choice row.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier tail inhabitation`

Proof. Induct over the displayed route $M, E, I, T, S, R, H, C_0, P, N$. The base rows M, E expose the regular-Cauchy modulus and tolerance budget, so the request has a finite precision scale before any tail is inspected.

At the index step, use the carrier obligation of Definition 1354.2. The canonical index row I is a displayed coordinate computed from M, E, and the next row T is the packet's own tail-window witness row.

The remaining rows preserve that witness. The seal row S consumes the tail only after T is present; the refusal row R excludes hidden selectors; and H, C_0, P, N transport, replay, source, and name the same finite packet. Hence tail inhabitation is a finite row induction, not an external choice principle. □

Theorem 1354.14 (Canonical tail-choice cofinality stability)**.** *Let $C = (M, E, I, T, S, R, H, C_0, P, N)$ be an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *carrier. If the sibling cofinal schedule of chapter 681 is read before I, then every later finite tolerance request transported through H remains on the same tail-window route $M, E, I, T, S, R, H, C_0, P, N$. In particular, cofinality is stable under componentwise* hsame *transport and displayed* Cont *replay, and no host-selected subsequence or quotient equality is exported.*

Proof. Start with the schedule factorization of Definition 1354.7. It places the cofinal schedule and prefix-budget rows before T and before the Real seal route S.

Now perform induction over later finite tolerance requests. The base request is the one served by E and I, and the induction step appends one displayed request by transporting the same modulus, budget, index, and tail rows through H.

The replay row C_0 records that this appended request follows the same route, while R refuses any move to a selected subsequence, quotient stream equality, hidden tail oracle, or ambient completeness row. Therefore cofinality stability is exactly stability of the displayed finite route. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier cofinality stability`

Theorem 1354.15 (Canonical tail-choice RegSeqRat consumption)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet for a Real-completion handoff must consume the rows*

$$M, E, I, T, S, R, H, C_0, P, N$$

in that order before the terminal $\mathsf{Real}^{\uparrow}$ *seal is treated as available. The consumer receives a finite regular-rational tail route through M, E, I, T and the structural rows, but no quotient stream equality, selected limit, countable choice, hidden tail oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier regseqrat consumption`

Proof. Begin with the Real-completion handoff theorem Definition 1354.10. It makes the diagonal seal read pass through the displayed route $M, E, I, T, S, R, H, C_0, P, N$.

For the regular-rational read, use the exact-boundary sibling cited there. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer can inspect a regular-rational tail only after the modulus row M, tolerance budget E, canonical index I, and tail-window witness T have been displayed.

Finally use no-choice non-escape, Definition 1354.8. The refusal row and structural rows keep the read inside the finite packet, so consuming the regular-rational tail cannot produce quotient equality, a selected limit, countable choice, a hidden tail oracle, or ambient Real completeness. □

Theorem 1354.16 (Canonical tail-choice scoped finite-choice route)**.** *Let $C = (M, E, I, T, S, R, H, C_0, P, N)$ be an accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *packet. Every completion-facing finite choice in C is scoped through* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{RealCompletionExactBoundaryUp}$ *by the displayed route*

$$M, E, I, T, S, R, H, C_0, P, N.$$

The route computes the finite index row I from the modulus and budget rows, inhabits the tail window T, reads the regular-rational tail through $\mathsf{RegSeqRat}^{\uparrow}$, *and hands the result to the exact completion boundary only after the refusal and structural rows have been displayed. It exports no quotient stream equality, hidden tail oracle, selected subsequence, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier scoped finite choice route`

Proof. Start from the schedule factorization theorem, Definition 1354.7. It places the $\mathsf{TailCofinalitySchedule}^{\uparrow}$ and $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ rows before the tail-window witness, so the finite index is read from the displayed packet rather than from an external choice function.

Next apply no-choice non-escape, Definition 1354.8. The refusal row blocks quotient stream equality, hidden tail oracles, selected subsequences, countable choice, selected limits, and ambient completeness before the completion-facing handoff is available.

The Real-completion handoff, Definition 1354.10, moves the packet from the tail window to the exact boundary through the same rows $M, E, I, T, S, R, H, C_0, P, N$. Tail cofinality stability and Definition 1354.15 then identify the $\mathsf{RegSeqRat}^{\uparrow}$ consumer as a downstream read of that finite route.

The transport, replay, provenance, and naming rows H, C_0, P, N repackage the same finite packet for consumers. Thus the only completion-facing choice made by the chapter is the displayed finite index choice, scoped through the listed L10 completion route and not through countable choice. □

Theorem 1354.17 (Canonical tail choice cofinal choice boundary)**.** *Every accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *read that uses a tail before a Real-completion consumer must factor through the displayed* $\mathsf{CauchyChoiceBoundary}^{\uparrow}$*, finite-prefix,* $\mathsf{RealCompletionExactBoundaryUp}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and tail-cofinality schedule rows before that tail can be consumed. The factorization exports the rows* $M, E, I, T, S, R, H, C_0, P, N$ *of Definition 1354.1; it exports no countable-choice selector, quotient stream equality, ambient Real completeness theorem, or host tail function.*

Proof. Project the carrier rows $M, E, I, T, S, R, H, C_0, P, N$. The modulus and budget rows M, E display the finite-prefix data before the canonical index row I is available, and the tail-window witness T is downstream of that index.

Compose the existing schedule and boundary routes. The schedule factorization theorem places the $\mathsf{TailCofinalitySchedule}^{\uparrow}$ and $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ rows before T, while the Real-completion handoff theorem moves from T to the $\mathsf{RealCompletionExactBoundaryUp}$ route through the same displayed carrier rows.

Read the regular and terminal faces last. The $\mathsf{RegSeqRat}^{\uparrow}$ consumption theorem identifies the regular-rational tail read as a downstream consumer of M, E, I, T, and the terminal $\mathsf{Real}^{\uparrow}$ seal is reached only through the exact-boundary handoff.

Finally apply the no-choice non-escape theorem. The refusal row R and the structural rows H, C_0, P, N preserve the finite route and leave no coordinate for countable choice, quotient stream equality, ambient completeness, or a host tail function. □

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier cofinal choice boundary`

Theorem 1354.18 (Canonical tail-choice scope grounding)**.** *Every accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *consumer route factors through the displayed rows*

$$M, E, I, T, S, R, H, C_0, P, N$$

and through the sibling surfaces $\mathsf{CauchyChoiceBoundary}^{\uparrow}$*,* $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$*,* $\mathsf{RealCompletionExactBoundaryUp}$ $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{TailCofinalitySchedule}^{\uparrow}$*. All transport and replay are read through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The scope contains no host-selected subsequence, quotient stream equality, countable choice, hidden tail oracle, metric completeness, ambient completeness, or detached Real-completion seal.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier scope grounding`

Proof. Begin with source obligations, Definition 1354.3. They place the no-choice boundary, prefix budget, Real seal, exact boundary, regular-rational source, and cofinal schedule before the canonical finite index row I or tail-window witness T can be consumed.

Next apply schedule factorization and tail cofinality, Definition 1354.7 and Definition 1354.11. These results force the route through the finite rows $M, E, I, T, S, R, H, C_0, P, N$, with $\mathsf{TailCofinalitySchedule}^{\uparrow}$ and $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ read before the tail-window witness.

Now attach the Real-completion side. The Real-completion handoff theorem, Definition 1354.10, routes the packet from the finite tail window to $\mathsf{RealCompletionExactBoundaryUp}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal, while Definition 1354.13 supplies the finite BHist witness for T after I is read.

Finally apply no-choice non-escape, Definition 1354.8. The refusal row R excludes host-selected subsequences, quotient stream equality, countable choice, hidden tail oracles, metric completeness, ambient completeness, and detached Real-completion seals; the structural rows H, C_0, P, N keep transport, replay, provenance, and naming inside BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. □

Theorem 1354.19 (Canonical tail-choice scoped source discipline)**.** *Let $C = (M, E, I, T, S, R, H, C_0, P, N)$ be an accepted $\mathsf{CanonicalTailChoice}^{\uparrow}$ packet. Before the seal row S is consumed by any Real-completion read, the source route must pass through $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{RealCompletionExactBoundaryUp}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ using only the displayed rows $M, E, I, T, S, R, H, C_0, P, N$. The route exposes no host-selected tail, quotient stream equality, hidden choice oracle, detached Real seal, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier scoped source discipline`

Proof. Project the carrier rows $M, E, I, T, S, R, H, C_0, P, N$. The modulus row M, finite budget row E, canonical index row I, and tail window row T are the only local source rows available before the seal route S is read.

Apply schedule factorization, no-choice non-escape, and the Real-completion handoff, namely Definition 1354.7, Definition 1354.8, and Definition 1354.10. These place the $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, and $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ surfaces before the tail window and keep the refusal row visible before any completion-facing seal is consumed.

Use $\mathsf{RegSeqRat}^{\uparrow}$ consumption and scope grounding, Definition 1354.15 and Definition 1354.18. The regular-rational read reaches $\mathsf{RealCompletionExactBoundaryUp}$ only through the sibling $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces named by that exact-boundary route, while H, C_0, P, N preserve transport, replay, provenance, and local naming.

Thus every seal-facing consumer reads the same finite source discipline before S. The excluded tail selector, quotient equality, hidden oracle, detached seal, and ambient completeness coordinates are absent from the carrier and from the cited boundary routes. □

Theorem 1354.20 (Canonical tail-choice public export)**.** *The public export of an accepted $\mathsf{CanonicalTailChoice}^{\uparrow}$ packet consists exactly of the displayed rows $M, E, I, T, S, R, H, C_0, P, N$ of Definition 1354.1 together with the sibling routes $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{RealCompletionExactBoundaryUp}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It exports no selected tail, quotient stream equality, countable choice, hidden oracle, detached Real seal, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CanonicalTailChoiceUp.CanonicalTailChoiceCarrier public export`

Proof. Start with the NameCert obligations of Definition 1354.2. They expose exactly the rows $M, E, I, T, S, R, H, C_0, P, N$ and give no coordinate for a selected tail, quotient equality, hidden oracle, or ambient completeness theorem.

Next compose the schedule and boundary routes. The schedule factorization of Definition 1354.7 places $\mathsf{TailCofinalitySchedule}^{\uparrow}$ and $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$ before the tail window, while the Real-completion handoff of Definition 1354.10 routes the seal through $\mathsf{RealCompletionExactBoundaryUp}$ and $\mathsf{Real}^{\uparrow}$.

Now add the scoped source discipline of Definition 1354.19. It forces the same source route through $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before any completion-facing consumer can read the seal row.

The structural rows H, C_0, P, N preserve, replay, source, and name that exact route. Therefore the listed rows and sibling routes are the whole public export, and the excluded objects remain absent from every public consumer surface. □

The Lean follow-up namespace is `BEDC.Derived.CanonicalTailChoiceUp`. Its `TasteGate.lean` surface must provide `Nontrivial` and `FieldFaithful` instances for the carrier of Definition 1354.1.

Closure status for $\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{CanonicalTailChoice}^{\uparrow}$ packet is generated from a regular-Cauchy modulus row, a finite tolerance-budget row, a canonical finite-index row, a tail-window witness row, a Real seal route, a no-choice refusal row, componentwise transports, continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: The scoped source discipline binds exactly $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{FinitePrefixLimitBudget}^{\uparrow}$, $\mathsf{RealCompletionExactBoun}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{TailCofinalitySchedule}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows for the finite tail-choice packet and its Real-completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CanonicalTailChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient-stream equality, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, canonical-index exactness, schedule factorization, no-choice refusal, Real-completion handoff, and TasteGate carrier support for the same packet.

Chapter 1355

A Concrete Naming Certificate for $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$

The $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$ packet realizes the transcendental supply position of subsection 9462.26.6 as finite BEDC NameCert data without turning that position into a closed observation object. It sits beside the closed observation system of chapter 616, the forward-binding gap ledger of chapter 1262, and the gap socket boundary of chapter 1343. Those siblings respectively record closed observation rows, present forward-binding commitments, and socket boundary discipline; this packet binds the supply-position row, closed-record interface, GAP socket, noninternalization refusal, route, provenance, and local naming rows before any consumer may read the position as licensed supply.

Definition 1355.1 (Closed observation supply boundary carrier)**.** *A* $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, Q, G, F, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the closed observation record row, read only through the closed observational discipline of chapter 616;*
2. *Q is the supply-position socket row naming the T-position without installing it as an observation state, record, rule, axiom, or object;*
3. *G is the GAP commitment row inherited from the forward-binding gap ledger of chapter 1262;*
4. *F is the noninternalization refusal row separating the supply position from substrate content;*
5. *A is the apophatic naming row that names the boundary without supplying a far-end object;*
6. *H records componentwise* hsame *transports for the closed record, socket, GAP, refusal, naming, route, provenance, and local certificate rows;*
7. *C records displayed* Cont *routes for supply-position inspection, GAP replay, noninternalization refusal, and downstream consumer audit;*
8. *P is the* Pkg *provenance over subsection 9462.26.1, subsection 9462.26.6,* $\mathsf{ClosedObservationSystem}^{\uparrow}$, $\mathsf{ForwardBindingGapLedger}^{\uparrow}$, $\mathsf{GapSocketBoundary}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{NegativeCertificateBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}}$ *row.*
The classifier accepts exactly packets whose visible rows are $R, Q, G, F, A, H, C, P, N$. It contains no hidden supply object, no closed-generation rule for T, no substrate axiom, no far-end witness, and no route that reads Q as a closed observation record.

Lean4 ✓ `BEDC.Derived.ClosedObservationSupplyBoundaryUp.ClosedObservationSupplyBoundaryUpTasteGate`
`single carrier alignment`

Theorem 1355.2 (Closed observation supply boundary NameCert obligations)**.** *For every accepted* $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$ *carrier $S = (R, Q, G, F, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationS}}$ *surface consists of finite carrier admission for the displayed packet, closed-record accountability through R, supply-position exposure through Q, GAP commitment through G, noninternalization refusal through F, apophatic boundary naming through A, componentwise transport through H, replay through C, provenance through P, and local certificate replay through N.*

Lean4 ✓ `BEDC.Derived.ClosedObservationSupplyBoundaryUp.ClosedObservationSupplyBoundaryTasteGate`
`single carrier alignment`

Proof. Start with carrier projection. The packet exposes exactly $R, Q, G, F, A, H, C, P, N$, so every accepted read is a read of one of those finite rows.

The sibling closed observation system supplies the closed-record discipline for R, while the forward-binding gap ledger supplies the GAP commitment row G. The gap socket boundary supplies the socket-side boundary shape through which Q is inspected.

The rows F and A keep the supply position from becoming substrate content. They allow the position to be named and refused as internal content without adding a closed-generation rule, a far-end witness, or a hidden axiom.

Finally the rows H, C, P, N replay only the displayed packet. Thus the NameCert surface is the listed finite obligation set and not an additional object layer. □

Theorem 1355.3 (Closed observation supply boundary noninternalization)**.** *No accepted* $\mathsf{ClosedObservationSupplyI}$ *carrier exports the supply-position row Q as a closed observation record, substrate axiom, closed-generation rule, licensed object, or far-end witness. Any consumer that reads Q as one of those internalized forms leaves the carrier surface $R, Q, G, F, A, H, C, P, N$ and refutes the packet.*

Proof. Use the carrier definition. The closed observation record is R, while the supply-position socket is Q; the two rows have different roles in the packet.

Apply the NameCert obligation theorem. It routes closed-record accountability through R, GAP commitment through G, and supply-position exposure through Q before the refusal and naming rows are consumed.

The noninternalization row F blocks the internal readings. The apophatic row A names the boundary without adding a far-end object, and the rows H, C, P, N preserve that separation under transport, replay, provenance, and local naming. Hence a consumer that treats Q as record, rule, axiom, object, or witness no longer reads an accepted packet. □

Falsifiable prediction: A consumer that reads the supply-position row as a closed observation record refutes the packet.

Independence witness: closedobservationsystem, forwardbindinggapledger, gapsocketboundary, apophaticname, negativecertificateboundary: none is bijective because none carries closed

records, supply-position socket, GAP commitment, noninternalization refusal, route, provenance, and NameCert as one packet.

The later Lean `TasteGate.lean` development for $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ClosedObservationSupplyBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ ClosedObservationSupplyBoundary}^{\uparrow}$ instances in `lean4/BEDC/Derived/ClosedObservationSupplyBoundaryUp/TasteGate.lean`. Those instances must preserve the closed record row, supply-position socket row, GAP commitment row, noninternalization refusal row, apophatic naming row, componentwise transport, replay route, package provenance, and local naming row.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$ packet is generated by attaching, in one finite BHist carrier, the supply-position row, the closed-observation record interface, the GAP socket row, the noninternalization refusal row, the apophatic naming row, and the transport / replay / package / local-NameCert rows. The classifier never identifies the supply-position row with the closed-record interface or with any internal observation object; every coordinate is compared through componentwise hsame along the visible Cont routes. The ledger story is therefore a boundary story: the supply position is named but not closed, and any consumer trying to read it as record, rule, axiom, object, or witness leaves the carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the supply-boundary carrier, NameCert obligation surface, noninternalization theorem, falsifiable prediction, and independence witness for $\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedObservationSupplyBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not internalize the supply position as a closed observation object, supply a far-end witness, or admit a host-level realization of the GAP socket.

Upgrade path: To reach obligationClosure, the finite carrier and noninternalization obligations must be mirrored by Lean definitions and checked transport / TasteGate instances under `lean4/BEDC/Derived/ClosedObservationSupplyBoundaryUp/`.

Chapter 1356

A Concrete Naming Certificate for $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$

The $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ packet turns the canonical tail-choice realization of chapter 1354 into an explicit audit surface for the Real-completeness route of chapter 9238. It reads the diagonal representative of chapter 1205, the finite observation-budget selector of chapter 782, the stream window of chapter 126, the regular-rational readback of chapter 301, the dyadic ledger of chapter 291, and the terminal Real seal of chapter 38. Its purpose is to expose the finite selector boundary as a named audit packet rather than letting a completion consumer confuse it with a countable-choice selector or a host-limit computation.

Falsifiable prediction: Removing the refusal row makes the audit unable to distinguish finite tail choice from countable-choice selection.

Independence witness: canonicaltailchoice, diagonallimitrepresentative, finiteobservationbudgetselector, realtailagreementseal, cofinalwindowrealseal: none is bijective to $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ because the listed packets isolate a canonical finite tail choice, a diagonal representative, an observation-budget selector, a paired tail agreement seal, or a cofinal Real seal, while the audit packet binds all of them to one carrier with displayed tail-window, refusal, transport, route, provenance, and Real-seal rows.

Definition 1356.1 (Real-completeness tail-choice audit carrier)**.** *A* $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, E, I, W, S, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus and tolerance row inherited from the accepted* $\mathsf{CanonicalTailChoice}^{\uparrow}$ *route.*
2. *E is the finite observation-budget row that determines which tail observations may be served.*
3. *I is the canonical finite tail-index row, computed from the displayed modulus and budget rows.*
4. *W is the tail-window row linking the index I to the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surfaces.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the diagonal representative and finite readback rows have been displayed.*

6. *R is the refusal row excluding countable choice, host-selected limits, quotient stream equality, hidden tail oracles, and ambient completeness.*
7. *H records componentwise* hsame *transport for the modulus, budget, index, window, seal, refusal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the finite selector rows through the tail-window readback, refusal row, and Real-seal route.*
9. *P is the* Pkg *provenance over* $\mathsf{CanonicalTailChoice}^{\uparrow}$, $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$, FiniteObservationBudgetSelecto $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CofinalWindowRealSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a countable-choice selector, a quotient stream, a selected host limit, an unbounded tail oracle, a metric-completeness proof, or a detached Real-completion theorem.

Lean4 ✓ `BEDC.Derived.RealCompletenessTailChoiceAuditUp.TasteGate.RealCompletenessTailChoiceAuditTasteGa`
`single carrier alignment`

Theorem 1356.2 (Real-completeness tail-choice audit non-escape)**.** *An accepted* $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ *packet exports exactly the rows*

$$M, E, I, W, S, R, H, C, P, N$$

of Definition 1356.1. No terminal Real-completion consumer can read a countable-choice selector, selected host limit, quotient stream equality, hidden tail oracle, ambient completeness field, or detached limit theorem through the audit packet.

Proof. Project the carrier. The analytic rows are the displayed modulus and tolerance row M, the finite observation budget E, the canonical index I, the tail-window row W, the Real seal row S, and the refusal row R.

The structural rows add no second analytic source. The transports H, continuation replay C, provenance P, and local name N only preserve and name the same finite packet.

The refusal row is part of the carrier itself. Since the excluded objects are not coordinates of the packet and are not introduced by the structural rows, any read of one of them leaves the accepted audit surface. □

Theorem 1356.3 (Real-completeness tail-choice audit Real-seal factorization)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ *packet factors its Real-seal read through the finite route*

$$M \longmapsto E \longmapsto I \longmapsto W \longmapsto S \longmapsto R \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route first consumes the canonical tail-choice row of chapter 1354, then the diagonal representative and finite observation-budget surfaces of chapter 1205 and chapter 782, and only then reads the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces.*

Proof. Begin with the canonical tail-choice packet. Its modulus, tolerance, finite-index, tail-window, seal, and refusal rows supply the finite selector route that the audit records as M, E, I, W, S, R.

Next attach the representative and budget consumers. The diagonal representative and finite observation-budget selector are read before the terminal Real seal, so the audit can expose only their displayed finite rows, not an external limit value or an unbounded selection principle.

Finally read the four concrete faces. The StreamName, DyadicRatCore, RegSeqRat, and Real surfaces appear through the same window and seal route already carried by W and S. The structural rows H, C, P, N transport, replay, source, and name that route, so every terminal read factors through the stated finite sequence. □

The Lean follow-up namespace is `BEDC.Derived.RealCompletenessTailChoiceAuditUp`. Its `TasteGate.lean` surface must provide `Nontrivial` and `FieldFaithful` instances for the carrier of Definition 1356.1.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessTailChoiceAudit}\uparrow}$.

Constructive story: A $\mathsf{RealCompletenessTailChoiceAudit}^{\uparrow}$ packet is generated from finite modulus, observation-budget, canonical-index, tail-window, Real-seal, refusal, transport, continuation, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessTailChoiceAudit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1356.1, Definition 1356.2, and Definition 1356.3 declare the finite audit packet separating canonical tail choice from countable-choice and host-limit routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessTailChoiceAudit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient-stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, audit non-escape, Real-seal factorization, and TasteGate carrier support for the same packet.

Chapter 1357

A Concrete Naming Certificate for $\mathsf{ObservationFiberDigest}^{\uparrow}$

The $\mathsf{ObservationFiberDigest}^{\uparrow}$ packet realizes the local visible digest and observation-fiber discipline of chapter 9402 as a finite NameCert carrier. It sits beside the digest provenance packet of chapter 1062, the apophatic naming boundary of chapter 543, and the fiber ledger of chapter 1067. Those siblings expose digest provenance, far-end refusal, and fiber membership separately; this carrier binds visible digest, observation-fiber membership, GAP ledger, signature and package provenance, local naming, far-end status, transport, and replay before any consumer may use a digest as an observation boundary.

Falsifiable prediction: Every accepted ObservationFiberDigestUp consumer must expose visible digest, observation-fiber membership, GAP ledger, Pkg/Sig provenance, NameCert row, and far-end status before using the digest as an observation boundary.

Independence witness: digestprovenancepacket, apophaticname, observerhistidentity, inscriptionevent, gapfailurebridgeaudit: none is bijective because each omits either the observation-fiber membership row, the local visible digest row, or the apophatic far-end boundary status.

Definition 1357.1 (ObservationFiberDigest carrier)**.** *An* $\mathsf{ObservationFiberDigest}^{\uparrow}$ *carrier is a finite packet*

$$O = (D, F, G, S, P, N, T, H, C)$$

with the following displayed rows:

1. *D is the local visible digest row, naming the classifier-facing digest token available to a particular observer state;*
2. *F is the observation-fiber membership row, recording the generated relation that binds the visible digest to compatible observation histories without identifying them;*
3. *G is the* GAP *ledger row for missing source supply, far-end socketing, and unavailable exactness data;*
4. *S is the signature provenance row, recording the* Sig *or classifier route that made the digest visible;*
5. *P is the* Pkg *provenance over* $\mathsf{DigestProvenancePacket}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, *observation histories, signatures, GAP rows,* hsame, Cont, *and* NameCert *rows;*
6. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservationFiberDigest}^{\uparrow}}$ *row;*

7. *T is the far-end status row, recording whether the observation fiber terminates in a refused, socketed, or locally exact boundary;*
8. *H records componentwise* hsame *transport for the digest, fiber, GAP, signature, package, local naming, far-end status, and replay rows;*
9. *C records the displayed* Cont *replay routes by which accepted consumers inspect D, F, G, S, P, N, T, H.*

The carrier has no coordinate for digest-as-source equality, a global universe digest, object-level T*, quotient fiber equality, hidden source equality, host preimage recovery, or observer-independent totalization.*

Lean4 (definition) `BEDC.Derived.ObservationFiberDigestUp.ObservationFiberDigestUp`

Theorem 1357.2 (ObservationFiberDigest NameCert obligations)**.** *For an accepted* $\mathsf{ObservationFiberDigest}^{\uparrow}$ *packet*

$$O = (D, F, G, S, P, N, T, H, C),$$

the local $\mathsf{NameCert}_{\mathsf{ObservationFiberDigest}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, visible digest exposure through D, observation-fiber membership through F, GAP ledger exposure through G, signature provenance through S, package provenance through P, local naming through N, far-end status through T, componentwise transport through H, replay locality through C, and non-escape from digest-as-source equality, global universe digest, object-level* T*, quotient fiber equality, and hidden source equality.*

Proof. Project the carrier of Definition 1357.1. Its visible inventory is exactly $D, F, G, S, P, N, T, H, C$, so carrier admission and local naming attach to those rows and to no host digest function or hidden observation substrate.

The digest boundary is read first through D. The observation fiber is not the same datum: it is the generated membership row F, and source-sensitive reads must keep F visible before the digest can be cited as an observation boundary.

The rows G, S, P, N, T then provide the finite obligation surface. The GAP row records missing supply and unavailable exactness; the signature row records how the digest became visible; the package row records the sibling route through chapter 1062, chapter 543, and chapter 1067; the local NameCert row records the packet name; and the far-end row records whether the fiber boundary is refused, socketed, or locally exact.

Finally H and C preserve the same finite packet under transport and replay. Since no displayed row is a source payload, a universe digest, an object-level truth value, a quotient representative, or a hidden equality witness, the listed non-escape obligations exhaust the local NameCert surface. □

Theorem 1357.3 (ObservationFiberDigest non-escape)**.** *Every accepted* $\mathsf{ObservationFiberDigest}^{\uparrow}$ *consumer that uses the digest as an observation boundary must factor through the displayed rows*

$$D, F, G, S, P, N, T, H, C.$$

The consumer obtains a local visible digest with fiber, ledger, signature, package, naming, far-end, transport, and replay evidence, but cannot obtain digest-as-source equality, a global universe digest, object-level T*, quotient fiber equality, hidden source equality, host preimage recovery, or observer-independent totalization from the packet.*

Proof. Start with an accepted packet $O = (D, F, G, S, P, N, T, H, C)$. A token-only read of D obtains only the visible digest row, because Definition 1357.1 gives no source coordinate behind D.

If the consumer uses the digest as an observation boundary, the read must open the observation-fiber membership row F. Without F, the digest is merely a visible classifier-facing token and has no packet-local authority to delimit compatible observation histories.

Next the read must pass through G, S, P, N, T. These rows expose the GAP ledger, signature provenance, package provenance, local certificate, and far-end status. In particular, T keeps the apophatic boundary visible rather than replacing it with an internal object.

The remaining rows H and C transport and replay the same packet. They do not add a preimage oracle, quotient collapse, hidden source equality, global universe digest, or object-level T coordinate. Therefore every accepted boundary use is confined to the displayed observation-fiber digest packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationFiberDigest}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservationFiberDigest}^{\uparrow}$ packet is generated from a local visible digest, observation-fiber membership row, GAP ledger, signature provenance, package provenance, local NameCert row, far-end status row, componentwise hsame transports, and displayed Cont replay routes.

Lean4 ✓ `BEDC.Derived.ObservationFiberDigestUp.taste gate`

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservationFiberDigest}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1357.1, Definition 1357.2, and Definition 1357.3 bind the displayed digest, fiber, GAP, signature, package, local naming, far-end, transport, and replay packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservationFiberDigest}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ObservationFiberDigestUp.taste\_gate`

Bridge status: none

Origin: ai

Not claimed: This chapter does not close source equality, global universe digest exposure, object-level truth, quotient fiber equality, hidden source equality, host preimage recovery, or bridge verification.

Upgrade path: A bridge-checked grade requires a standard bridge for the same observation-fiber digest packet and its displayed boundary rows.

Chapter 1358

A Concrete Naming Certificate for $\mathsf{RegistryChangeContinuity}^{\uparrow}$

The $\mathsf{RegistryChangeContinuity}^{\uparrow}$ packet realizes the active-reading continuity clause of section 9462.35 as a finite NameCert carrier. It sits beside the active-reading gate of chapter 1249, the closure-status reference of chapter 1265, the registry-consistency audit of chapter 1223, the claim registry layer of chapter 1209, and the cannot-claim registry of chapter 1318. Those siblings expose current-reading uniqueness, status bounds, consistency checks, claim admission, and refusal rows; this carrier binds the current row, the displaced row, the status boundary, the refusal route, and the replay evidence in one registry-machine packet.

Definition 1358.1 (Registry-change continuity carrier)**.** *A* $\mathsf{RegistryChangeContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, A, D, S, R, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the registry target row, naming the theorem, definition, writing item, formal target, or package entry whose reading changes.*
2. *A is the active-reading row after the change, imported through the current-reading surface of* $\mathsf{ActiveReadingGate}^{\uparrow}$*.*
3. *D is the displaced-reading row, retained as blocked audit data rather than as a second active reading.*
4. *S is the status-bound row, recording the closure-status and formal-status boundary read from* $\mathsf{ClosureStatusReference}^{\uparrow}$*.*
5. *R is the cannot-claim refusal row, recording the public reason that blocks the displaced row from export as current.*
6. *B is the blocked-reading row, binding D to R and recording that the displaced row remains visible only as audit data.*
7. *H records componentwise* hsame *transport for target, active reading, displaced reading, status bound, refusal, blocking, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from T through A, S, R, B to the export-readable registry route.*

9. *P is the* Pkg *provenance over* RegistryChangeContinuity$^{\uparrow}$, ActiveReadingGate$^{\uparrow}$, ClosureStatusReference$^{\uparrow}$, RegistryConsistencyAudit$^{\uparrow}$, ClaimRegistryLayer$^{\uparrow}$, PhilosophyCannotClaimRegistry$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for exporting D as a second active reading, changing the status boundary outside S, citing a refusal outside R, replaying the change outside C, or naming the packet outside the finite row $K = (T, A, D, S, R, B, H, C, P, N)$.

Lean4 ✓ `BEDC.Derived.RegistryChangeContinuityUp.RegistryChangeContinuityTasteGate single carrier alignment`

Theorem 1358.2 (Registry-change continuity obligations)**.** *For the carrier of Definition 1358.1, the local* $\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (T, A, D, S, R, B, H, C, P, N)$; target admission through T; current-reading admission through A; displaced-reading exposure through D; status-bound exactness through S; refusal and blocked-reading compatibility through R and B; registry-consistency and claim-layer readability through* RegistryConsistencyAudit$^{\uparrow}$ *and* ClaimRegistryLayer$^{\uparrow}$*; and transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 1358.1. The rows $T, A, D, S, R, B, H, C, P, N$ are the only coordinates available to the local certificate, so every accepted change-continuity read must pass through those rows.

The registry target is fixed by T. The active row A is the single current reading, while D is retained only as displaced audit data. The status row S bounds the change by the closure-status and formal-status surface, so the carrier cannot silently change the public grade of the target.

The refusal and blocking rows supply the non-export clause. Row R records the cannot-claim reason, and row B binds that reason to the displaced row. Thus the carrier exposes why D remains readable as history while refusing to make it a second current claim.

Rows H, C, P, N preserve the same finite packet under transport, replay, provenance, and naming. They add no coordinate for a hidden active row, so the displayed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}$ surface. □

Theorem 1358.3 (Registry-change continuity active-reading refusal)**.** *Every accepted* RegistryChangeContinuity$^{\uparrow}$ *packet exports the target row T through the active row A, the status-bound row S, the refusal row R, and the blocked-reading row B. A packet that exports both A and D as active readings for the same target is rejected by* RegistryChangeContinuity$^{\uparrow}$.

Proof. Start with an accepted packet

$$K = (T, A, D, S, R, B, H, C, P, N).$$

By Definition 1358.2, the export-readable route contains T, A, S, R, B as displayed rows.

The active-reading gate supplies uniqueness for A, and the closure-status reference supplies the status bound S. The displaced row D is still present, but it is tied to the cannot-claim row R through the blocked-reading row B.

Transport, replay, provenance, and local naming preserve that same route. Since no coordinate turns D into an additional active row, any packet that exports both A and D as active readings for the same target leaves the displayed carrier and is refused by B. □

Falsifiable prediction: A packet that exports both the active row and the displaced row for the same target as active readings is rejected by RegistryChangeContinuityUp.

Independence witness: active-reading-gate, closure-status-reference, registry-consistency-audit, claimregistrylayer, philosophycannotclaimregistry: none is bijective because RegistryChangeContinuityUp binds active reading, blocked displaced reading, status bound, cannot-claim refusal, transport, replay, provenance, and local NameCert rows in one carrier.

The Lean follow-up must provide explicit Nontrivial $\mathsf{RegistryChangeContinuity}^{\uparrow}$ and FieldFaithful RegistryChangeC instances in `lean4/BEDC/Derived/RegistryChangeContinuityUp/TasteGate.lean`. Those instances must preserve $T, A, D, S, R, B, H, C, P, N$ as separate displayed fields before BEDC.Derived.RegistryChangeC can be cited by a checked paper marker.

Closure status for $\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{RegistryChangeContinuity}^{\uparrow}$ packet is generated from target, active-reading, displaced-reading, status-bound, refusal, blocked-reading, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1358.1, Definition 1358.2, and Definition 1358.3 bind active-reading continuity to a finite registry-machine NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegistryChangeContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global registry repair, archival completeness for every displaced reading, or public export correctness for every downstream consumer.

Upgrade path: Lean closure requires Nontrivial RegistryChangeContinuityUp and FieldFaithful RegistryChangeContinuityUp instances in `lean4/BEDC/Derived/RegistryChangeContinuityUp/TasteGate.lean`.

Chapter 1359

A Concrete Naming Certificate for $\mathsf{HashLikeBoundaryGate}^{\uparrow}$

The $\mathsf{HashLikeBoundaryGate}^{\uparrow}$ packet realizes the eight-question audit family of Definition 9402.35 as finite NameCert data. It sits after the general boundary-gate packet of chapter 1070, the hash-like fiber boundary of chapter 1084, the digest-fiber ledger of chapter 1067, the hash digest-fiber boundary of chapter 1087, and the digest provenance packet of chapter 1062. Those siblings provide boundary questions, visible digest rows, hidden fiber rows, exactness/refusal discipline, and provenance rows; this packet makes the H1–H8 gate itself a consumer-facing certificate.

Definition 1359.1 (HashLikeBoundaryGate carrier)**.** *A* $\mathsf{HashLikeBoundaryGate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, F, E, R, V, H, C, P, N)$$

with the following displayed rows:

1. *D is the visible digest-like token row;*
2. *F is the provenance fiber row attached to the same token;*
3. *E is the exactness row recording whether digest equality is licensed as source equality;*
4. *R is the refusal row for absent provenance, absent exactness, global-universe hashes,* T*-value confusion, self-substance leakage, physical-constant derivation, quotient collapse, and un sourced certificate names;*
5. *V is the audit-verdict row for the H1–H8 questions of Definition 9402.35;*
6. *H records componentwise* hsame *transport for D, F, E, R, V together with the structural rows;*
7. *C records the displayed* Cont *routes through digest, fiber, exactness, refusal, audit verdict, transport, provenance, and local naming reads;*
8. *P is the* Pkg *provenance over* $\mathsf{BoundaryGate}^{\uparrow}$, $\mathsf{HashLikeFiberBoundary}^{\uparrow}$, $\mathsf{DigestFiberLedger}^{\uparrow}$, $\mathsf{HashDigestFiberBoun}$ $\mathsf{DigestProvenancePacket}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HashLikeBoundaryGate}^{\uparrow}}$ *row.*

The carrier accepts exactly those digest-like consumers that display provenance fiber, exactness, refusal, audit verdict, transport, continuation, package, and local naming rows before treating digest equality as any source-facing claim.

Lean4 ✓ BEDC.Derived.HashLikeBoundaryGateUp.HashLikeBoundaryGateTasteGate single carrier alignment

Theorem 1359.2 (HashLikeBoundaryGate NameCert obligations)**.** *For the carrier of Definition 1359.1, the local* $\mathsf{NameCert}_{\mathsf{HashLikeBoundaryGate}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$X = (D, F, E, R, V, H, C, P, N),$$

visible digest exposure through D*, provenance-fiber exposure through* F*, exactness exposure through* E*, H1–H8 refusal exposure through* R*, audit-verdict exposure through* V*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Consequently, an accepted consumer cannot use digest equality as source equality unless the exactness row E *is displayed together with* F, R, V, H, C, P, N*.*

Proof. Project the displayed carrier. The rows D, F, E, R, V are the complete audit content, while H, C, P, N transport, replay, package, and name the same finite packet.

Read the sibling routes in order. The boundary-gate surface supplies the finite question and verdict pattern, the hash-like fiber boundary supplies digest and hidden-fiber discipline, the digest-fiber ledger supplies provenance-fiber exposure, the hash digest-fiber boundary supplies exactness discipline, and the digest provenance packet supplies source-row ancestry.

The H1–H8 refusal row then exhausts the gate. Since the carrier has no coordinate for a global universe hash, detached digest-to-source equality, unledgered T input, subject parameter outside Hist, physical constant derivation, quotient collapse, or un sourced certificate citation, no accepted consumer can obtain those readings from the packet. □

Theorem 1359.3 (HashLikeBoundaryGate nonescape)**.** *Every accepted* $\mathsf{HashLikeBoundaryGate}^{\uparrow}$ *consumer factors through*

$$D, F, E, R, V, H, C, P, N.$$

The packet exports no digest-as-source equality, no selected preimage, no global universe digest, no T*-valued theorem input, no subject behind* Hist*, no physical constant, no quotient source equality, and no certificate name detached from its source rows.*

Proof. Use the NameCert obligations of Definition 1359.2. They place every consumer-visible read on the same digest, fiber, exactness, refusal, audit-verdict, transport, continuation, provenance, and local naming route.

The exactness coordinate is displayed as a gate row rather than as an implicit equality principle. Thus digest equality can be consumed only after the packet also exposes its fiber, refusal, verdict, transport, route, provenance, and naming rows.

The forbidden outputs require coordinates absent from Definition 1359.1. Transport and replay preserve the displayed packet; they do not create source preimages, universe hashes, theorem inputs, subject parameters, physical constants, quotient equalities, or detached certificate names. □

Falsifiable prediction: Every accepted HashLikeBoundaryGateUp consumer must display provenance fiber, exactness, refusal, audit verdict, hsame, Cont, Pkg, and NameCert rows before using digest equality as source equality.

Independence witness: boundarygate, hashlikefiberboundary, digestfiberledger, hashdigest-fiberboundary, and digestprovenancepacket: none is bijective to HashLikeBoundaryGateUp because this carrier binds the H1–H8 audit verdict to digest, fiber, exactness, refusal, transport, continuation, provenance, and local naming rows in one packet.

The Lean follow-up namespace is `BEDC.Derived.HashLikeBoundaryGateUp`. Its `TasteGate.lean` file must include `Nontrivial HashLikeBoundaryGateUp` and `FieldFaithful HashLikeBoundaryGateUp` instances for the displayed digest, fiber, exactness, refusal, audit verdict, transport, continuation, provenance, and naming carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HashLikeBoundaryGate}^{\uparrow}}$.

Constructive story: A $\mathsf{HashLikeBoundaryGate}^{\uparrow}$ packet is generated by attaching, in one finite BHist carrier, the digest row, fiber row, exactness row, refusal row, audit-verdict row, and the transport / continuation / package / local-NameCert rows. The classifier compares packets only through componentwise hsame along the displayed Cont routes; the boundary gate never identifies two distinct digest fibers by an external hash-collision argument. The ledger story is a boundary story: each refusal-side coordinate stays visible, and any consumer trying to bypass the gate via a host-level hash equality leaves the carrier.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HashLikeBoundaryGate}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the gate carrier, the NameCert obligation surface, the audit-verdict row, and the falsifiable / independence rows for $\mathsf{HashLikeBoundaryGate}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HashLikeBoundaryGate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: It does not export host-level hash equality, claim collision resistance as a substrate-internal theorem, or admit a digest-fiber identification outside the displayed gate.

Upgrade path: To reach obligationClosure, the finite gate carrier and audit-verdict obligations must be mirrored by Lean definitions and checked TasteGate instances under `lean4/BEDC/Derived/HashLikeBoundaryGateUp/`.

Chapter 1360

A Concrete Naming Certificate for AuditMapRouteBudget$^{\uparrow}$

The AuditMapRouteBudget$^{\uparrow}$ packet realizes the route-budget surface of the GroundCompiler audit map in chapter 9407. It sits beside the cross-map weave of chapter 839, the common audit interface of chapter 806, the event-flow audit packet of chapter 829, the flow-level round-trip packet of chapter 1094, and the cannot-claim export gate of chapter 1319. Those siblings expose dependency routing, generic audit rows, event-flow evidence, flow readback, or claim-export refusal separately; this packet binds the complete event-flow, source-channel, recognizer, certificate-gate, report-boundary, and cannot-claim route budget as one finite NameCert surface.

Definition 1360.1 (Audit-map route-budget carrier)**.** *An* AuditMapRouteBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, R, G, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the event-flow input row naming the GroundCompiler event family admitted by chapter 9407.*
2. *S is the source-channel encoding row recording the legal source channel, channel readback, and lossless route on the same event family.*
3. *R is the recognizer-evidence row recording generated recognizer acceptance without treating reports or generated code as proof.*
4. *G is the certificate-gate row recording which exported certificate gates are open for the accepted route and which gates are absent.*
5. *Q is the report-boundary row recording compiler reports, metric reports, motif reports, theorem-code rows, and chapter-code rows as reports rather than certificates.*
6. *L is the cannot-claim ledger row recording hidden-input, proof-export, and certificate-export refusals for the same route.*
7. *H records componentwise* hsame *transport for event-flow, source-channel, recognizer, certificate-gate, report-boundary, ledger, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for reading E, S, R, G, Q, L in the accepted order.*
9. *P records* Pkg *provenance over chapter 9407, the cited sibling concrete packets,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{AuditMapRouteBudget}^{\uparrow}}$ *row.*
The carrier has no coordinate for a hidden compiler input, proof object, certificate object, report-derived theorem, unbudgeted route, or cannot-claim discharge outside $E, S, R, G, Q, L, H, C, P, N$.

Lean4 (definition) `BEDC.Derived.AuditMapRouteBudgetUp.TasteGate.AuditMapRouteBudgetUp`
Lean4 ✓ `BEDC.Derived.AuditMapRouteBudgetUp.TasteGate.AuditMapRouteBudgetUp single carrier alignment`

Theorem 1360.2 (Audit-map route-budget NameCert obligations)**.** *For the carrier of Definition 1360.1, the local* $\mathsf{NameCert}_{\mathsf{AuditMapRouteBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (E, S, R, G, Q, L, H, C, P, N)$, event-flow input exposure through E, source-channel readback through S, recognizer evidence through R, certificate-gate exposure through G, report-boundary exposure through Q, cannot-claim ledger exposure through L, componentwise* hsame *stability through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1360.1. The event-flow row, source-channel row, recognizer row, certificate-gate row, report-boundary row, and cannot-claim ledger row are visible as separate BHist coordinates.

The GroundCompiler audit-map reading of chapter 9407 fixes the order of use. A public route first reads event-flow input, then source-channel encoding, then recognizer evidence, and only then reaches certificate gates, reports, and cannot-claim ledgers.

The support rows H, C, P, N preserve the same route. Componentwise hsame cannot add a proof coordinate, displayed Cont replay cannot bypass a certificate gate, Pkg records the cited siblings, and local NameCert naming binds the route budget to the displayed packet. □

Theorem 1360.3 (Audit-map route-budget falsifiable boundary)**.** *An accepted* $\mathsf{AuditMapRouteBudget}^{\uparrow}$ *packet cannot export a proof, certificate, or hidden compiler input unless the corresponding certificate-gate row G or cannot-claim ledger row L is displayed in the same route budget. A consumer that reads E, S, R, Q, H, C, P, N while omitting G or L may read event flow, channel encoding, recognizer evidence, and reports, but it cannot claim exported proof evidence, exported certificate evidence, or hidden input access.*

Proof. Begin with the NameCert obligations of Definition 1360.2. They separate recognizer evidence, certificate-gate exposure, report-boundary exposure, and cannot-claim ledger exposure as distinct rows.

If G is omitted, no row records an opened certificate gate. If L is omitted, no row records the hidden-input, proof-export, or certificate-export refusal. The remaining rows can still expose event flow, channel encoding, recognizer acceptance, reports, transports, continuations, provenance, and local naming, but none of them is a proof or certificate export row.

Thus the boundary is falsifiable by inspection. A displayed route that exports proof evidence, certificate evidence, or hidden input while lacking the matching G or L row is not an accepted $\mathsf{AuditMapRouteBudget}^{\uparrow}$ packet. □

Theorem 1360.4 (Audit-map route-budget dependency independence)**.** *The* $\mathsf{AuditMapRouteBudget}^{\uparrow}$ *carrier is not bijective to the sibling packets in chapter 839, chapter 806, chapter 829, chapter 1094, or chapter 1319. None of those siblings carries the complete route-budget row set $E, S, R, G, Q, L, H, C, P, N$: the dependency weave lacks GroundCompiler source-channel and report rows, the generic interface lacks the GroundCompiler-specific event and certificate budget, the event-flow audit lacks the full report and cannot-claim ledger, the flow-level round trip lacks the*

recognizer and certificate-gate budget, and the cannot-claim export gate lacks event-flow, source-channel, and recognizer rows.

Proof. Compare row inventories. Each cited sibling supplies a necessary surface, but each omits at least one row required by Definition 1360.1.

The dependency weave contributes cross-map routing, the interface contributes generic audit shape, the event-flow audit contributes event evidence, the flow-level round trip contributes channel readback, and the cannot-claim export gate contributes refusal discipline. None supplies all of $E, S, R, G, Q, L, H, C, P, N$ with the same local NameCert row.

Therefore a route-budget consumer may cite those siblings through P, but it cannot substitute any one sibling for the accepted $\mathsf{AuditMapRouteBudget}^{\uparrow}$ carrier. □

Falsifiable prediction: An accepted AuditMapRouteBudgetUp route cannot export proof evidence, certificate evidence, or hidden compiler input unless the matching certificate-gate or cannot-claim row is displayed in the same finite packet.

Independence witness: auditmapdependencyweave, auditmapinterface, groundcompilereventflowaudit, groundcompilerflowlevelroundtrip, cannotclaimexportgate: none is bijective to AuditMapRouteBudgetUp because none carries the simultaneous event-flow, source-channel, recognizer, certificate-gate, report-boundary, cannot-claim, transport, continuation, provenance, and local NameCert rows.

The Lean follow-up for `BEDC.Derived.AuditMapRouteBudgetUp.TasteGate.lean` must include explicit Nontrivial $\mathsf{AuditMapRouteBudget}^{\uparrow}$ and FieldFaithful $\mathsf{AuditMapRouteBudget}^{\uparrow}$ instances. The field-faithfulness witness must keep $E, S, R, G, Q, L, H, C, P, N$ distinct, so report rows cannot be read as certificates and cannot-claim rows cannot be erased by transport.

Closure status for $\mathsf{NameCert}_{\mathsf{AuditMapRouteBudget}^{\uparrow}}$.

Constructive story: An $\mathsf{AuditMapRouteBudget}^{\uparrow}$ packet is generated from GroundCompiler event-flow rows, source-channel encoding rows, recognizer-evidence rows, certificate-gate rows, report-boundary rows, cannot-claim ledger rows, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AuditMapRouteBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1360.1, Definition 1360.2, Definition 1360.3, and Definition 1360.4 bind the finite GroundCompiler route-budget carrier and its audit-map export boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AuditMapRouteBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, compiler self-hosting, proof extraction from reports, certificate extraction from recognizers, hidden compiler input, global audit-map closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-channel readback, recognizer evidence, certificate-gate exposure, report-boundary exposure, cannot-claim ledger exposure, transport stability, continuation stability, provenance, and local naming for the displayed route budget.

Chapter 1361

A Concrete Naming Certificate for $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$

The $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ packet turns the finite observation-budget selector of chapter 782 into the cofinal window surface required by the Real-completeness route of chapter 9238. It is composite over the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Its role is not to choose an infinite subsequence; it names the finite rows whose selected windows are cofinal for every displayed diagonal request currently served.

Definition 1361.1 (Finite observation cofinality-selector carrier)**.** *A* $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *budget row that bounds the visible Real-completion requests;*
2. *Q is the cofinality-request row recording the finite diagonal requests currently served by B;*
3. *W is the finite family of selected* $\mathsf{StreamName}^{\uparrow}$ *windows whose indices meet every request in Q;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger read from the windows in W;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached after the dyadic ledger is displayed;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached through R;*
7. *H records componentwise* hsame *transport for the budget, request, window, dyadic, regular, seal, continuation, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from B and Q to W, from W to D, from D to R, and from R to E;*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a countable-choice selector, quotient stream, host real equality test, hidden diagonal subsequence, detached limit value, or ambient completeness theorem.

Theorem 1361.2 (Finite observation cofinality-selector NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ *carrier*

$$F = (B, Q, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}}$ *surface consists exactly of carrier exposure for* F*, budget admission through* B*, finite cofinality-request exposure through* Q*, selected-window coverage through* W*, dyadic readback through* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*,* $\mathsf{Real}^{\uparrow}$ *seal exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 1361.1. The displayed tuple lists the accepted budget selector, the finite cofinality requests, the selected StreamName windows, the dyadic ledger, the regular handoff, the Real seal, and the structural rows.

Read the analytic rows in order. The budget row B bounds the requests in Q, and the continuation route sends those requests to the selected windows W before any dyadic or regular readback can be used.

The seal-facing rows then remain downstream from the same finite surface. The dyadic ledger D is read from W, the handoff row R is read from D, and the seal row E is read from R.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same packet, and no further analytic coordinate appears in the carrier. Hence the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteObservationCofinalitySelectorUp.TasteGate.FiniteObservationCofinalitySelecto`
`single carrier alignment`

Theorem 1361.3 (Finite observation cofinality-selector non-escape)**.** *Every Real-completion, diagonal-limit,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ *packet factors through the displayed finite route*

$$B \to Q \to W \to D \to R \to E$$

together with H, C, P, N*. The consumer may read the finite budget selector, cofinality-request row, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, continuation rows, provenance, and local name, but it cannot read a countable-choice selector, quotient stream, host real equality test, hidden diagonal subsequence, detached limit value, or ambient completeness theorem through the packet.*

Proof. Begin with the NameCert obligations. They place every accepted read on the displayed carrier rows and require the budget-to-window and window-to-seal route to pass through B, Q, W, D, R, E.

Use the cofinality role of the request row. The row Q records only the finite diagonal requests currently served by the budget row B, while W records the finite selected windows that meet those requests. It is therefore a finite cofinal surface, not an infinite subsequence or choice principle.

The remaining rows add no escape route. The dyadic, regular, and Real rows D, R, E are downstream from the displayed windows, and H, C, P, N only preserve, replay, source, and name the same route.

The excluded objects are not coordinates of the carrier and are not introduced by the structural rows. Any consumer that reads one of them has left the accepted $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ packet. □

The later Lean packet is expected at `BEDC.Derived.FiniteObservationCofinalitySelectorUp`. Because the carrier is structurally composite over the existing Real-completion budget and stream-window carriers rather than atomic, its future Lean surface still needs `Nontrivial` and `FieldFaithful` instances for the carrier of Definition 1361.1.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}$ packet is generated from a finite observation-budget selector row, finite cofinality-request row, selected StreamName window family, dyadic ledger, regular handoff, Real seal, componentwise transports, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1361.1, Definition 1361.2, and Definition 1361.3 declare the finite cofinality-selector packet between observation budgets and Real seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationCofinalitySelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close RealUp completeness, quotient-stream equality, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal request coverage, seal-route non-escape, and TasteGate carrier support for the same packet.

Chapter 1362

A Concrete Naming Certificate for $\mathsf{ActionCommitment}^{\uparrow}$

The $\mathsf{ActionCommitment}^{\uparrow}$ surface realizes the inscription-event action reading of subsection 9462.27.10 as finite NameCert data. It sits after the hard-problem boundary of chapter 1236, the free-will inscription commitment of chapter 592, the forward causal socket of chapter 505, the other-minds locality commitment of chapter 499, and the focused multi-history ledger of chapter 1109. Those siblings supply closure pressure, inscription-event discipline, causal commitment, public other-system evidence, and focused ledger support; this packet binds them into the action-facing route through which a selected inscription becomes a local actualization row.

Definition 1362.1 (ActionCommitment carrier)**.** *An* $\mathsf{ActionCommitment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, R, L, G, T, C, H, Q, P, N)$$

with the following displayed rows:

1. *I is the inscription-event row, consumed through* $\mathsf{InscriptionEvent}^{\uparrow}$ *and* $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$*;*
2. *R is the selected route row, naming the finite route by which the event is accepted as the action-facing record rather than as an unselected future row;*
3. *L is the local actualization row, recording that the selected route is read at the local inscription point;*
4. *G is the gap and commitment row, binding the route to* GAP *provenance,* $\mathsf{CausalCommitment}^{\uparrow}$*,* $\mathsf{OtherMindsCommitment}^{\uparrow}$*, and* $\mathsf{HardProblemClosure}^{\uparrow}$ *without adding a hidden agency source;*
5. *T records classifier transport for the inscription, route, local actualization, and gap-commitment rows;*
6. *C records the displayed* Cont *route from focused ledger support through inscription event, selected route, local actualization, and commitment boundary;*
7. *H records componentwise* hsame *transport for I, R, L, G, T, C, P, N;*
8. *Q is the replay row, requiring every downstream action read to replay the displayed route before reading the local NameCert row;*
9. *P is the* Pkg *provenance over* $\mathsf{ActionCommitment}^{\uparrow}$, $\mathsf{HardProblemClosure}^{\uparrow}$, $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$, $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, $\mathsf{FocusedMultiHistLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{ActionCommitment}^{\uparrow}}$ *row.*

The carrier is exactly this packet. It contains no hidden faculty of will, no selected future row outside R, no unledgered agency primitive, no host psychology predicate, and no off-packet causal necessity row.

Lean4 ✓ `BEDC.Derived.ActionCommitmentUp.ActionCommitmentTasteGate single carrier alignment`

Theorem 1362.2 (ActionCommitment NameCert obligations)**.** *For the carrier of Definition 1362.1, the local* $\mathsf{NameCert}_{\mathsf{ActionCommitment}^{\uparrow}}$ *surface consists exactly of carrier exposure for $A = (I, R, L, G, T, C, H, Q, P, N)$, inscription-event accountability for I, selected-route accountability for R, local-actualization accountability for L, gap and sibling-commitment accountability for G, classifier stability through T and H, displayed continuation replay through C and Q, package provenance through P, local naming through N, and non-escape from hidden free-will faculties, selected future rows, and unledgered agency primitives.*

Proof. Project the carrier of Definition 1362.1. The rows $I, R, L, G, T, C, H, Q, P, N$ are the complete displayed inventory before a consumer may read the action commitment.

Read the action-facing content in order. The inscription-event row I supplies the event surface, the selected route row R names the finite route, and the local actualization row L records the inscription point at which that route is read as action.

The gap and commitment row G then attaches the sibling boundaries. $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$ keeps the event from becoming a hidden faculty, $\mathsf{CausalCommitment}^{\uparrow}$ keeps causal reading on the forward-GAP route, $\mathsf{OtherMindsCommitment}^{\uparrow}$ keeps public agency evidence in locality-cell form, and $\mathsf{HardProblemClosure}^{\uparrow}$ keeps closure pressure exposed as a boundary row.

Finally use T, C, H, Q, P, N. These rows transport, replay, source, and name the same finite packet. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1362.3 (ActionCommitment inscription route)**.** *Every accepted* $\mathsf{ActionCommitment}^{\uparrow}$ *packet routes an action read through the displayed inscription-event path*

$$I, R, L, G, T, C, H, Q, P, N.$$

The route begins with the inscription event, passes through the selected route and local actualization rows, exposes the gap and commitment boundary, and then replays only the displayed transport, continuation, provenance, and local naming rows. It exports no hidden free-will faculty, no selected future row, and no unledgered agency primitive outside that route.

Proof. Begin with the inscription row. The packet contains a single action-facing event coordinate I, and it is read through the inscription-event discipline already used by $\mathsf{FreeWillInscriptionCommitment}^{\uparrow}$.

The route is then selected and localized. The row R names the finite route that becomes public record, while L records the local actualization point. Neither row contains a future selector outside the accepted packet.

Next expose the commitment boundary. The row G records the GAP and sibling-commitment handoff to $\mathsf{CausalCommitment}^{\uparrow}$, $\mathsf{OtherMindsCommitment}^{\uparrow}$, and $\mathsf{HardProblemClosure}^{\uparrow}$ without replacing those siblings by an agency substance.

The structural rows close the route. Componentwise transport, continuation replay, package provenance, and local naming preserve I, R, L, G as displayed rows; because no carrier coordinate names a hidden faculty, selected future row, or unledgered agency primitive, no consumer can extract one from the accepted packet. □

Falsifiable prediction: An accepted ActionCommitmentUp packet cannot export a hidden free-will faculty, selected future row, or unledgered agency primitive outside the displayed inscription-event route.

Independence witness: freewillinscriptioncommitment, causalcommitment, othermindscommitment, hardproblemclosure, focused-multi-hist-ledger: each sibling supplies one boundary of inscription, causality, public other-system evidence, closure pressure, or focused ledger support, but none is bijective to the full inscription-route, local-actualization, gap-commitment, transport, replay, provenance, and NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ActionCommitment}^{\uparrow}}$.

Constructive story: An $\mathsf{ActionCommitment}^{\uparrow}$ packet is generated from an inscription-event row, selected route row, local actualization row, gap and sibling-commitment row, classifier transport, displayed continuation replay, componentwise hsame transport, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ActionCommitment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1362.1, Definition 1362.2, and Definition 1362.3 bind the finite inscription-event action surface and its non-escape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ActionCommitment}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close a Lean carrier implementation, bridge status, hidden free-will faculty, selected future row, causal necessity row, host psychology predicate, or agency primitive outside the displayed route.

Upgrade path: Obligation closure requires Lean encoding of the carrier and checked theorems for NameCert obligations, inscription-route exactness, and sibling independence.

This hub points the nona tuple packet to its adjacent tuple surfaces, the octa chapter chapter 1365 and the deca chapter chapter 1364.

Chapter 1363

A Concrete Naming Certificate for $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ packet abstracts the shared arity-nine tag-encoded BHist carrier pattern used across a carrier-isomorphism bucket. It is composite over the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, the BEDC self-substrate packet of chapter 869, the apophatic fixed-point fiber of chapter 1069, and the beta critical-pair packet of chapter 881. Those siblings exhibit the same finite shape: a fixed list of tagged BHist rows, componentwise comparison, displayed hsame transport, displayed Cont replay, provenance, and local naming. This chapter names that carrier-isomorphism surface directly so downstream rows can cite one BEDC-native packet rather than repeating the same arity-nine pattern.

Definition 1363.1 (BHistNonaTupleNameCert carrier)**.** *A* $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_8$ *are the nine tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local role used by the member chapter and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_8, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same nine-row packet.*
4. P *is the* Pkg *provenance over the arity-nine tag encoder, the five named sibling carrier families,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistNonaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a tenth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1363.2 (BHistNonaTupleNameCert classifier)**.** *Two* $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ *packets*

$$N = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, H, C, P, L)$$

and

$$N' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{array}{lllll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, & R_4 \sim R'_4, \\ R_5 \sim R'_5, & R_6 \sim R'_6, & R_7 \sim R'_7, & R_8 \sim R'_8, & \end{array}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1363.3 (BHistNonaTupleNameCert carrier admission)**.** *Every accepted* BHistNonaTupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, H, C, P, L.$$

The nine rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Proof. Project the carrier of Definition 1363.1. The definition lists the nine tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed tenth component or detached structural table lies outside the accepted surface. □

Lean4 ✓ `BEDC.Derived.BHistNonaTupleNameCertUp.BHistNonaTupleNameCertTasteGate single carrier alignment`

Theorem 1363.4 (BHistNonaTupleNameCert classifier exactness)**.** *For accepted* BHistNonaTupleNameCert$^{\uparrow}$ *packets N and N', the classifier of Definition 1363.2 is exact for the arity-nine carrier surface: it compares the nine tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_8$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1363.5 (BHistNonaTupleNameCert componentwise stability)**.** *Componentwise* hsame *transport for a* BHistNonaTupleNameCert$^{\uparrow}$ *packet preserves exactly the nine tagged rows, the replay route, provenance, and local name. Any stable consumer of the carrier must preserve*

$$R_0, \ldots, R_8, H, C, P, L$$

as displayed fields of the same packet before reading a carrier-isomorphism claim.

Proof. Use the H row in Definition 1363.1. It is the only row that records componentwise hsame transport, and its component list is exactly the nine carrier rows together with the replay, provenance, and local naming rows.

The classifier exactness of Definition 1363.4 forces any carrier-isomorphism read to pass through that displayed component list. A consumer that drops a row or adds an unlisted row is therefore no longer reading the accepted packet. □

Theorem 1363.6 (BHistNonaTupleNameCert provenance readback)**.** *The provenance row P of an accepted* BHistNonaTupleNameCert$^{\uparrow}$ *packet reads back only the arity-nine tag encoder, the sibling carrier families named in this chapter, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It does not read a bucket-wide replacement of chapter 1150, chapter 807, chapter 869, chapter 1069, or chapter 881.*

Proof. Inspect the provenance row in the carrier definition. It records the arity-nine tag encoder, the named sibling carrier families, and the local BEDC support rows.

The row is a readback ledger, not a command to alter any sibling chapter. Since neither the carrier nor the classifier contains a member-chapter replacement coordinate, provenance can source the shared pattern without rewriting any bucket member. □

Theorem 1363.7 (BHistNonaTupleNameCert local naming nonescape)**.** *Every local* NameCert$_{\text{BHistNonaTupleNameCert}}$ *read factors through the displayed packet*

$$N = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, H, C, P, L).$$

The read may expose the arity-nine tag-encoded carrier pattern and its componentwise routes, but it cannot export an untagged carrier, a tenth row, a host-side equality test, or a relocation of existing bucket member chapters into this capstone.

Proof. The local name is the row L of the carrier. By Definition 1363.3, L is read only together with the nine carrier rows and the displayed structural rows.

Use classifier exactness and componentwise stability. They force any carrier-isomorphism read to preserve the same tag-encoded components and the same displayed hsame and Cont routes. Since no untagged carrier, tenth row, host equality test, or member-chapter replacement row appears in the packet, local naming cannot export one. □

Lean4 ✓ `BEDC.Derived.BHistNonaTupleNameCertUp.BHistNonaTupleNameCertLocalNamingNonescape`

Theorem 1363.8 (BHistNonaTupleNameCert adjacent arity isolation)**.** *Any classifier-preserving read from* BHistNonaTupleNameCert$^{\uparrow}$ *to either adjacent tuple surface, the arity-eight packet of chapter 1365 or the arity-ten packet of chapter 1364, must either drop one of*

$$R_0, \ldots, R_8$$

or introduce a row outside the nona carrier. Consequently no bijective carrier correspondence preserves the tag list, componentwise hsame *routes,* Cont *replay, provenance, and local* NameCert *rows simultaneously across the nona packet and either adjacent arity sibling.*

Proof. Project the accepted nona carrier with Definition 1363.3. A classifier-preserving read must carry the nine tagged rows and the structural rows H, C, P, L through the exact classifier of Definition 1363.4.

Componentwise stability then fixes the row inventory. By Definition 1363.5 and Definition 1363.7, the local name may expose only the displayed nona packet; it has no clause that forgets a tagged row or admits an external coordinate.

Compare this fixed inventory with the adjacent tuple surfaces. The octa sibling has only eight tagged carrier rows, so a read into it loses the ninth displayed nona row. The deca sibling has a tenth tagged carrier row, so a read into it adds a coordinate not admitted by the nona carrier. In either direction the purported bijection violates row preservation or the no-extra-coordinate boundary, and therefore cannot preserve tag list, hsame, Cont, provenance, and local naming at once. □

Theorem 1363.9 (BHistNonaTupleNameCert scoped support package)**.** *The carrier admission, classifier exactness, componentwise stability, provenance readback, and local naming nonescape clauses bind the* $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ *packet exactly to the displayed BEDC support*

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},$$

and to the five sibling carrier families named in this chapter: chapter 1150, chapter 807, chapter 869, chapter 1069, and chapter 881. The resulting scope is the arity-nine tag-encoded BHist *packet with its displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden tenth row.*

Proof. Apply Definition 1363.3 to identify the admitted rows and exclude any tenth coordinate. Apply Definition 1363.4 to keep comparison on the nine tagged BHist rows and the displayed structural rows.

Apply Definition 1363.5 to force all stable reads through the same hsame and Cont support. Apply Definition 1363.6 to restrict sourcing to the arity-nine tag encoder, the five sibling carrier families, and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Finally, apply Definition 1363.7. The local NameCert row can expose only that displayed packet, so the scoped package has no route to a bucket-wide sibling rewrite, host-side equality classifier, or hidden tenth row. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, a tenth carrier row, or bucket membership alone fails one of carrier admission, classifier exactness, componentwise stability, provenance readback, or local naming nonescape.

Independence witness: anchor-stability-certificate, auditmapfamily, bedc-self-substrate, apophatic-fixed-point-fiber, and beta-critical-pair: none is bijective to $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ because this packet carries all nine tagged rows together with transport, replay, provenance, and local NameCert in one packet, while each sibling supplies only one member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistNonaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistNonaTupleNameCert}^{\uparrow}$ packet is generated from nine tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row; BEDC.Derived.BHistNonaTupleNameC records the composite TasteGate witness.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BHistNonaTupleNameCert}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1363.1, Definition 1363.2, Definition 1363.3, Definition 1363.4, Definition 1363.5, Definition 1363.6, Definition 1363.7, Definition 1363.8, and Definition 1363.9 bind the arity-nine tag-encoded carrier-isomorphism surface and its exact BEDC support package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BHistNonaTupleNameCert}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BHistNonaTupleNameCertUp.BHistNonaTupleNameCertTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: ai

Not claimed: This chapter does not relocate, rewrite, replace, or retire any bucket member chapter, and it does not close bridge checking.

Upgrade path: Bridge checking requires a standard bridge from the scoped package to the public acceptance schema.

Chapter 1364

A Concrete Naming Certificate for $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$ packet extends the carrier-isomorphism capstone of chapter 1363 to the arity-ten tag-encoded BHist surface. It is composite over representative bucket siblings including the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, the BEDC self-substrate packet of chapter 869, the apophatic fixed-point fiber of chapter 1069, and the beta critical-pair packet of chapter 881. It shares the arity-ten BHist tag encoder and transport/provenance budget with chapter 1368, while fixing positions as tuple rows rather than sequence-indexed rows. The chapter gives the ten-row carrier a single BEDC name so later rows can cite the capstone rather than restating the same tag, comparison, transport, replay, provenance, and local-name schema.

Definition 1364.1 (BHistDecaTupleNameCert carrier)**.** *A* $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_9$ *are the ten tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local role used by a member chapter and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_9, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same ten-row packet.*
4. P *is the* Pkg *provenance over the arity-ten tag encoder, the named sibling carrier families,* BHist, hsame, Cont, *and* NameCert *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistDecaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, an eleventh carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1364.2 (BHistDecaTupleNameCert classifier)**.** *Two* $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$ *packets*

$$D = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, H, C, P, L)$$

and

$$D' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{array}{lllll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, & R_4 \sim R'_4, \\ R_5 \sim R'_5, & R_6 \sim R'_6, & R_7 \sim R'_7, & R_8 \sim R'_8, & R_9 \sim R'_9, \end{array}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1364.3 (BHistDecaTupleNameCert carrier admission)**.** *Every accepted* BHistDecaTupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, H, C, P, L.$$

The ten rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Lean4 ✓ `BEDC.Derived.BHistDecaTupleNameCertUp.BHistDecaTupleNameCertCarrierAdmission`

Proof. Project the carrier of Definition 1364.1. The definition lists the ten tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed eleventh component or detached structural table lies outside the accepted surface. □

Theorem 1364.4 (BHistDecaTupleNameCert classifier exactness)**.** *For accepted* BHistDecaTupleNameCert$^{\uparrow}$ *packets D and D', the classifier of Definition 1364.2 is exact for the arity-ten carrier surface: it compares the ten tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_9$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1364.5 (BHistDecaTupleNameCert scoped support package)**.** *The carrier admission and classifier exactness clauses bind the* BHistDecaTupleNameCert$^{\uparrow}$ *packet to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

and to the representative sibling carrier families cited above. The resulting scope is the arity-ten tag-encoded BHist *packet with displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden eleventh row.*

Proof. Apply Definition 1364.3 to identify the admitted rows and exclude any eleventh coordinate. Apply Definition 1364.4 to keep comparison on the ten tagged BHist rows and the displayed structural rows.

The support package is therefore the named BEDC packet itself. Its provenance can source the cited bucket siblings, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, an eleventh carrier row, or bucket membership alone fails carrier admission or classifier exactness for $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$.

Independence witness: b-hist-nona-tuple-name-cert and the cited bucket siblings: none is bijective to $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$ because this packet carries ten tagged rows together with transport, replay, provenance, and local NameCert in one packet, while the nona capstone carries only nine tagged rows and each sibling supplies a member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistDecaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistDecaTupleNameCert}^{\uparrow}$ packet is generated from ten tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistDecaTupleNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1364.1, Definition 1364.2, Definition 1364.3, Definition 1364.4, and Definition 1364.5 identify the arity-ten carrier-isomorphism surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistDecaTupleNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistDecaTupleNameCertUp and checked carrier-admission plus classifier-exactness theorems.

Chapter 1365

A Concrete Naming Certificate for $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$ packet is a composite over the arity-eight tag-encoded BHist carrier-isomorphism bucket. It sits between the nona and deca capstones of chapter 1363 and chapter 1364, and it cites the same bucket anchors: the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, the beta critical-pair packet of chapter 881, the eight-row checkpoint surface of chapter 997, and the finite support pattern carried by the neighboring tuple capstones. The chapter gives the eight-row carrier a single BEDC name so consumers can read the shared tag, comparison, transport, replay, provenance, and local-name pattern without treating any sibling as atomic.

Definition 1365.1 (BHistOctaTupleNameCert carrier)**.** *A* $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_7$ *are the eight tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local carrier role used by a cited bucket sibling and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_7, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same eight-row packet.*
4. P *is the* Pkg *provenance over the arity-eight tag encoder, the cited sibling carrier families,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistOctaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a ninth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1365.2 (BHistOctaTupleNameCert classifier)**.** *Two* $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$ *packets*

$$O = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, H, C, P, L)$$

and

$$O' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{array}{llll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, \\ R_4 \sim R'_4, & R_5 \sim R'_5, & R_6 \sim R'_6, & R_7 \sim R'_7, \end{array}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1365.3 (BHistOctaTupleNameCert carrier admission)**.** *Every accepted* BHistOctaTupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, H, C, P, L.$$

The eight rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Proof. Project the carrier of Definition 1365.1. The definition lists the eight tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed ninth component or detached structural table lies outside the accepted surface. □

Lean4 ✓ `BEDC.Derived.BHistOctaTupleNameCertUp.BHistOctaTupleNameCertCarrierAdmission`

Theorem 1365.4 (BHistOctaTupleNameCert classifier exactness)**.** *For accepted* BHistOctaTupleNameCert$^{\uparrow}$ *packets O and O', the classifier of Definition 1365.2 is exact for the arity-eight carrier surface: it compares the eight tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_7$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1365.5 (BHistOctaTupleNameCert scoped support package)**.** *The carrier admission and classifier exactness clauses bind the* BHistOctaTupleNameCert$^{\uparrow}$ *packet to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

and to the cited sibling carrier families. The resulting scope is the arity-eight tag-encoded BHist *packet with displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden ninth row.*

Proof. Apply Definition 1365.3 to identify the admitted rows and exclude any ninth coordinate. Apply Definition 1365.4 to keep comparison on the eight tagged BHist rows and the displayed structural rows.

The support package is therefore the named BEDC packet itself. Its provenance can source the cited bucket siblings, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, a ninth carrier row, or bucket membership alone fails carrier admission or classifier exactness for $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$.

Independence witness: b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, auditmap-family, anchor-stability-certificate, beta-critical-pair, and reflection-runtime-checkpoint: none is bijective to $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$ because this packet carries exactly eight tagged rows together with transport, replay, provenance, and local NameCert in one packet, while the nona and deca capstones carry different arities and each cited sibling supplies its own member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistOctaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistOctaTupleNameCert}^{\uparrow}$ packet is generated from eight tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BHistOctaTupleNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 1365.1, Definition 1365.2, Definition 1365.3, Definition 1365.4, and Definition 1365.5 identify the arity-eight carrier-isomorphism surface and its BEDC support package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BHistOctaTupleNameCert}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistOctaTupleNameCertUp and checked carrier-admission plus classifier-exactness theorems.

Chapter 1366

A Concrete Naming Certificate for $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ packet is the arity-eleven capstone for the carrier-isomorphism bucket already exhibited by the neighboring tuple chapters chapter 1365, chapter 1363, and chapter 1364. It is composite over the same bucket anchors: the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, the BEDC self-substrate packet of chapter 869, the apophatic fixed-point fiber of chapter 1069, and the beta critical-pair packet of chapter 881. The chapter gives the eleven-row carrier a single BEDC name so downstream consumers can cite the tag, comparison, transport, replay, provenance, and local-name surface without treating any bucket member as a replacement for another.

Definition 1366.1 (BHistHendecaTupleNameCert carrier)**.** *A* $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{10}$ *are the eleven tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local carrier role used by a cited bucket sibling and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{10}, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same eleven-row packet.*
4. P *is the* Pkg *provenance over the arity-eleven tag encoder, the cited sibling carrier families,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistHendecaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a twelfth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1366.2 (BHistHendecaTupleNameCert classifier)**.** *Two* $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ *packets*

$$E = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, H, C, P, L)$$

and

$$E' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{array}{llllll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, & R_4 \sim R'_4, & R_5 \sim R'_5, \\ R_6 \sim R'_6, & R_7 \sim R'_7, & R_8 \sim R'_8, & R_9 \sim R'_9, & R_{10} \sim R'_{10}, & \end{array}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1366.3 (BHistHendecaTupleNameCert obligations)**.** *Every accepted* $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, H, C, P, L,$$

and its classifier is exact for that arity-eleven carrier surface. The eleven rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Proof. Project the carrier of Definition 1366.1. The definition lists the eleven tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

Next read the classifier in Definition 1366.2. It requires one comparison for each displayed carrier row and then uses the same row list for hsame transport, Cont replay, provenance, and local naming.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed twelfth component, detached structural table, host equality test, or bucket-membership comparison lies outside the accepted surface. □

Theorem 1366.4 (BHistHendecaTuple tag exhaustion)**.** *For a* $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ *packet, the tags carried by $R_0, \ldots, R_{10}$ exhaust the carrier rows available to the local* NameCert *interface. A consumer may permute or replay the displayed rows only through the named* hsame *and* Cont *rows H and C; it may not manufacture an unlisted tag, collapse two tags into one bucket witness, or import an additional row from a sibling chapter.*

Proof. The local $\mathsf{NameCert}_{\mathsf{BHistHendecaTupleNameCert}^{\uparrow}}$ row L is part of the displayed carrier, so its reads are bounded by the carrier coordinates named in Definition 1366.1. Those coordinates contain exactly the eleven tagged BHist rows and the structural rows H, C, P, L.

The classifier in Definition 1366.2 repeats the same eleven tags and gives no rule for an omitted tag or a collapsed bucket witness. Transport and replay therefore preserve the displayed tag list rather than extending it. □

Lean4 ✓ `BEDC.Derived.BHistHendecaTupleNameCertUp.BHistHendecaTupleTagExhaustion`

Falsifiable prediction: Any accepted consumer route that reads an untagged row, a twelfth carrier row, or bucket membership alone fails carrier admission, classifier exactness, or tag exhaustion for $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$.

Independence witness: b-hist-octa-tuple-name-cert, b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, and the cited bucket siblings: none is bijective to $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ because this packet carries eleven tagged rows together with transport, replay, provenance, and

local NameCert in one packet, while the neighboring capstones carry different arities and each sibling supplies only a member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistHendecaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistHendecaTupleNameCert}^{\uparrow}$ packet is generated from eleven tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistHendecaTupleNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1366.1, Definition 1366.2, Definition 1366.3, and Definition 1366.4 identify the arity-eleven carrier-isomorphism surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistHendecaTupleNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistHendecaTupleNameCertUp and checked carrier-obligation plus tag-exhaustion theorems.

Chapter 1367

A Concrete Naming Certificate for $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$

The $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$ packet is the sequence-window capstone for the arity-nine carrier-isomorphism bucket. It is anchored by the static nona tuple certificate of chapter 1363, the arity-eight neighbor of chapter 1365, the arity-ten neighbor of chapter 1364, and the arity-eleven neighbor of chapter 1366. The tuple chapters exhibit fixed finite carriers; this chapter names the BEDC-native window that reads the same nine displayed BHist rows together with an order row, componentwise transport, continuation replay, provenance, and local naming.

Definition 1367.1 (BHistNonaSequenceNameCert carrier)**.** *A* $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_8$ *are the nine tag-encoded* BHist *rows admitted by the window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the index and order row. It records the finite window order, the first and last admitted positions, and the rule that every sequence read factors through one of* $R_0, \ldots, R_8$*.*
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_8, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the nine-row window encoder, the tuple capstone anchors cited above, chapter 807, chapter 1150,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistNonaSequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a tenth sequence component, a reordered hidden window, a host equality test, an unordered bucket tuple, or a replacement of any tuple capstone chapter.

Definition 1367.2 (BHistNonaSequenceNameCert classifier)**.** *Two* BHistNonaSequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed window row compares componentwise,

$$\begin{aligned} &R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \quad R_4 \sim R'_4, \\ &R_5 \sim R'_5, \quad R_6 \sim R'_6, \quad R_7 \sim R'_7, \quad R_8 \sim R'_8, \end{aligned}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from window admission through local naming. No comparison is available through bucket membership alone, an omitted position, an untagged payload, a tuple-chapter rewrite, or a host equality test.*

Theorem 1367.3 (BHistNonaSequenceNameCert window admission)**.** *Every accepted* BHistNonaSequenceNameCe *packet admits exactly the displayed window rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, I, H, C, P, L.$$

The nine rows are tag-encoded BHist *components, I is the only index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Proof. Project the carrier of Definition 1367.1. The definition lists the nine tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed tenth component, detached order table, or hidden structural row lies outside the accepted surface. □

Theorem 1367.4 (BHistNonaSequenceNameCert classifier exactness)**.** *For accepted* BHistNonaSequenceNameCert *packets S and S', the classifier of Definition 1367.2 is exact for the arity-nine sequence window: it compares the nine* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered tuple membership, by a hidden permutation, by a detached order table, or by a host equality test.*

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered tuple membership, hidden permutation, detached order table, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1367.5 (BHistNonaSequenceNameCert index stability)**.** *Componentwise* hsame *transport for a* BHistNonaSequenceNameCert$^{\uparrow}$ *packet preserves the finite order row I together with the nine tagged rows, the replay route, provenance, and local name. A stable carrier-isomorphism read must preserve*

$$R_0, \ldots, R_8, I, H, C, P, L$$

as displayed fields of the same sequence-window packet.

Proof. Use the H row in Definition 1367.1. It is the only row that records componentwise hsame transport, and its component list is exactly the nine carrier rows together with the order, replay, provenance, and local naming rows.

The classifier of Definition 1367.2 repeats the same list and requires the index rows to display the same finite order. A consumer that drops the order row, changes the window order, or adds an unlisted row is therefore no longer reading the accepted packet. □

Theorem 1367.6 (BHistNonaSequenceNameCert provenance readback)**.** *The provenance row P of an accepted* BHistNonaSequenceNameCert$^{\uparrow}$ *packet reads back only the nine-row window encoder, the neighboring tuple capstones, chapter 807, chapter 1150, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It does not read a bucket-wide replacement of any tuple capstone, audit-map chapter, or anchor-stability chapter.*

Proof. Inspect the provenance row in the carrier definition. It records the nine-row window encoder, the named capstone and support chapters, and the local BEDC support rows.

The row is a readback ledger for the ordered packet, not a command to alter any cited chapter. Since neither the carrier nor the classifier contains a replacement coordinate, provenance can source the shared bucket context without rewriting any member. □

Theorem 1367.7 (BHistNonaSequenceNameCert local naming nonescape)**.** *Every local* NameCert$_{\text{BHistNonaSequenceN}}$ *read factors through the displayed packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, I, H, C, P, L).$$

The read may expose the nine-row sequence window, its finite order, and its componentwise routes, but it cannot export an untagged carrier, a tenth row, a host-side equality test, or a relocation of chapter 1363, chapter 1365, chapter 1364, or chapter 1366.

Proof. The local name is the row L of the carrier. By Definition 1367.3, L is read only together with the nine window rows, the index row, and the displayed structural rows.

Use index stability and the classifier definition. They force any carrier-isomorphism read to preserve the same tag-encoded components, the same finite order, and the same displayed hsame and Cont routes. Since no untagged carrier, tenth row, host equality test, or tuple-chapter replacement row appears in the packet, local naming cannot export one. □

Theorem 1367.8 (BHistNonaSequenceNameCert scoped support package)**.** *The window admission, classifier exactness, index stability, provenance readback, and local naming nonescape clauses bind the* BHistNonaSequenceNameCert$^{\uparrow}$ *packet exactly to the displayed BEDC support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

to the neighboring tuple capstones chapter 1365, chapter 1363, chapter 1364, and chapter 1366, and to the audit-map and anchor-stability supports of chapter 807 and chapter 1150. The resulting scope is the ordered arity-nine BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a host-side equality classifier, or a hidden tenth position.*

Proof. Apply Definition 1367.3 to identify the admitted rows and exclude any tenth coordinate. Apply Definition 1367.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1367.5 to force all stable reads through the same ordered hsame and Cont support. Apply Definition 1367.6 to restrict sourcing to the nine-row window encoder, the named capstone and support chapters, and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Finally, apply Definition 1367.7. The local NameCert row can expose only that displayed packet, so the scoped package has no route to an unordered bucket tuple, host-side equality classifier, or hidden tenth position. □

Falsifiable prediction: Any accepted consumer route that reads an unordered tuple coordinate, an untagged row, a tenth sequence row, a hidden permutation, bucket membership alone, or an order table detached from I fails window admission, classifier exactness, index stability, provenance readback, or local naming nonescape for $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$.

Independence witness: b-hist-octa-tuple-name-cert, b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, b-hist-hendeca-tuple-name-cert, auditmapfamily, and anchor-stability-certificate: none is bijective to $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$ because this packet carries a nine-row sequence window with an explicit index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply tuple, audit, or anchor surfaces.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistNonaSequenceNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistNonaSequenceNameCert}^{\uparrow}$ packet is generated from nine tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistNonaSequenceNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1367.1, Definition 1367.2, Definition 1367.3, Definition 1367.4, Definition 1367.5, Definition 1367.6, Definition 1367.7, and Definition 1367.8 identify the arity-nine sequence-window carrier surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistNonaSequenceNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, tuple-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistNonaSequenceNameCertUp and checked window-admission, classifier-exactness, index-stability, provenance-readback, local-naming, and scoped-support theorems.

Chapter 1368

A Concrete Naming Certificate for $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$

The $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$ packet is the ordered arity-ten sequence-window capstone for the carrier-isomorphism bucket. It is composite over the deca tuple certificate of chapter 1364, the nona sequence certificate of chapter 1367, the octa tuple certificate of chapter 1365, the hendeca tuple certificate of chapter 1366, the audit-map family of chapter 807, and the anchor-stability certificate of chapter 1150. The chapter names the BEDC-native window that reads ten displayed BHist rows together with an order row, componentwise transport, continuation replay, provenance, and local naming, without rewriting any member chapter.

Definition 1368.1 (BHistDecaSequenceNameCert carrier)**.** *A* $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_9$ *are the ten tag-encoded* BHist *rows admitted by the window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite order and index row. It records the window order, the first and last admitted positions, and the rule that every sequence read factors through one of* $R_0, \ldots, R_9$*.*
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_9, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the ten-row window encoder, the tuple and sequence capstone anchors cited above, chapter 807, chapter 1150,* BHist, hsame, Cont, *and* NameCert *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistDecaSequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, an eleventh sequence component, a reordered hidden window, a host equality test, unordered bucket membership alone, or a replacement of any tuple or sequence capstone chapter.

Definition 1368.2 (BHistDecaSequenceNameCert classifier)**.** *Two* BHistDecaSequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed window row compares componentwise,

$$\begin{array}{lllll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, & R_4 \sim R'_4, \\ R_5 \sim R'_5, & R_6 \sim R'_6, & R_7 \sim R'_7, & R_8 \sim R'_8, & R_9 \sim R'_9, \end{array}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from window admission through local naming. No comparison is available through hidden permutation, untagged payload, host equality, bucket membership alone, omitted position, tuple-chapter rewrite, or sequence-chapter rewrite.*

Theorem 1368.3 (BHistDecaSequenceNameCert window admission)**.** *Every accepted* BHistDecaSequenceNameCer *packet admits exactly the displayed window rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, I, H, C, P, L.$$

The ten rows are tag-encoded BHist *components, I is the only index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Proof. Project the carrier of Definition 1368.1. The definition lists the ten tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed eleventh component, detached order table, or hidden structural row lies outside the accepted surface. □

Theorem 1368.4 (BHistDecaSequenceNameCert classifier exactness)**.** *For accepted* BHistDecaSequenceNameCert$^{\uparrow}$ *packets S and S', the classifier of Definition 1368.2 is exact for the arity-ten sequence window: it compares the ten* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered tuple membership, by a hidden permutation, by a detached order table, by bucket membership alone, by any sibling-chapter rewrite, or by a host equality test.*

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered tuple membership, hidden permutation, detached order table, bucket membership alone, sibling-chapter rewrite, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1368.5 (BHistDecaSequenceNameCert index stability)**.** *Componentwise* hsame *transport for a* BHistDecaSequenceNameCert$^{\uparrow}$ *packet preserves the finite order row I together with the*

ten tagged rows, the replay route, provenance, and local name. A stable carrier-isomorphism read must preserve

$$R_0, \ldots, R_9, I, H, C, P, L$$

as displayed fields of the same sequence-window packet.

Proof. Use the H row in Definition 1368.1. It is the only row that records componentwise hsame transport, and its component list is exactly the ten carrier rows together with the order, replay, provenance, and local naming rows.

The classifier of Definition 1368.2 repeats the same list and requires the index rows to display the same finite order. A consumer that drops the order row, changes the window order, or adds an unlisted row is therefore no longer reading the accepted packet. □

Theorem 1368.6 (BHistDecaSequenceNameCert provenance readback)**.** *The provenance row P of an accepted* BHistDecaSequenceNameCert$^{\uparrow}$ *packet reads back only the ten-row window encoder, the neighboring tuple and sequence capstones, chapter 807, chapter 1150, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows. It does not read a bucket-wide replacement of any tuple capstone, sequence capstone, audit-map chapter, or anchor-stability chapter.*

Proof. Inspect the provenance row in the carrier definition. It records the ten-row window encoder, the named capstone and support chapters, and the local BEDC support rows.

The row is a readback ledger for the ordered packet, not a command to alter any cited chapter. Since neither the carrier nor the classifier contains a replacement coordinate, provenance can source the shared bucket context without rewriting any member. □

Theorem 1368.7 (BHistDecaSequenceNameCert local naming nonescape)**.** *Every local* NameCert$_{\text{BHistDecaSequenceNa}}$ *read factors through the displayed packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, I, H, C, P, L).$$

The read may expose the ten-row sequence window, its finite order, and its componentwise routes, but it cannot export an untagged carrier, an eleventh row, a hidden permutation, a host-side equality test, or a relocation of chapter 1364, chapter 1367, chapter 1363, chapter 1365, or chapter 1366.

Proof. The local name is the row L of the carrier. By Definition 1368.3, L is read only together with the ten window rows, the index row, and the displayed structural rows.

Use index stability and the classifier definition. They force any carrier-isomorphism read to preserve the same tag-encoded components, the same finite order, and the same displayed hsame and Cont routes. Since no untagged carrier, eleventh row, hidden permutation, host equality test, or capstone replacement row appears in the packet, local naming cannot export one. □

Theorem 1368.8 (BHistDecaSequenceNameCert scoped support package)**.** *The window admission, classifier exactness, index stability, provenance readback, and local naming nonescape clauses bind the* BHistDecaSequenceNameCert$^{\uparrow}$ *packet exactly to the displayed BEDC support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

to the tuple and sequence capstones chapter 1364, chapter 1367, chapter 1363, chapter 1365, and chapter 1366, and to the audit-map and anchor-stability supports of chapter 807 and chapter 1150. The resulting scope is the ordered arity-ten BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a host-side equality classifier, a hidden permutation, or a hidden eleventh position.*

Proof. Apply Definition 1368.3 to identify the admitted rows and exclude any eleventh coordinate. Apply Definition 1368.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1368.5 to force all stable reads through the same ordered hsame and Cont support. Apply Definition 1368.6 to restrict sourcing to the ten-row window encoder, the named capstone and support chapters, and the local BHist, hsame, Cont, Pkg, and NameCert rows.

Finally, apply Definition 1368.7. The local NameCert row can expose only that displayed packet, so the scoped package has no route to an unordered bucket tuple, host-side equality classifier, hidden permutation, or hidden eleventh position. □

Falsifiable prediction: Any accepted consumer route that reads an unordered tuple coordinate, an untagged row, an eleventh sequence row, a hidden permutation, bucket membership alone, an order table detached from I, or a sibling capstone relocation fails window admission, classifier exactness, index stability, provenance readback, or local naming nonescape for $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$.

Independence witness: b-hist-deca-tuple-name-cert, b-hist-nona-sequence-name-cert, b-hist-nona-tuple-name-cert, b-hist-octa-tuple-name-cert, b-hist-hendeca-tuple-namecert, auditmapfamily, and anchor-stability-certificate: none is bijective to $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$ because this packet carries ten payload rows plus an order row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply sequence, tuple, audit, or anchor surfaces.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistDecaSequenceNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistDecaSequenceNameCert}^{\uparrow}$ packet is generated from ten tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistDecaSequenceNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1368.1, Definition 1368.2, Definition 1368.3, Definition 1368.4, Definition 1368.5, Definition 1368.6, Definition 1368.7, and Definition 1368.8 identify the arity-ten sequence-window carrier surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistDecaSequenceNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, tuple-chapter replacement, sequence-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistDecaSequenceNameCertUp and checked window-admission, classifier-exactness, index-stability, provenance-readback, local-naming, and scoped-support theorems.

Chapter 1369

A Concrete Naming Certificate for $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$ packet is the arity-twelve capstone for the carrier-isomorphism bucket displayed by the neighboring tuple chapters chapter 1365, chapter 1363, chapter 1364, and chapter 1366. It is composite over representative bucket siblings: the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, the BEDC self-substrate packet of chapter 869, the apophatic fixed-point fiber of chapter 1069, and the beta critical-pair packet of chapter 881. The chapter gives the twelve-row carrier a single BEDC name so downstream consumers can cite the tag, comparison, transport, replay, provenance, and local-name surface without copying the member-local tuple schema.

Definition 1369.1 (BHistDodecaTupleNameCert carrier)**.** *A* $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{11}$ *are the twelve tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local carrier role used by a cited bucket sibling and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{11}, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same twelve-row packet.*
4. P *is the* Pkg *provenance over the arity-twelve tag encoder, the cited sibling carrier families,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistDodecaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a thirteenth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1369.2 (BHistDodecaTupleNameCert classifier)**.** *Two* $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$ *packets*

$$T = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, H, C, P, L)$$

and

$$T' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, R'_{11}, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{array}{llllll} R_0 \sim R'_0, & R_1 \sim R'_1, & R_2 \sim R'_2, & R_3 \sim R'_3, & R_4 \sim R'_4, & R_5 \sim R'_5, \\ R_6 \sim R'_6, & R_7 \sim R'_7, & R_8 \sim R'_8, & R_9 \sim R'_9, & R_{10} \sim R'_{10}, & R_{11} \sim R'_{11}, \end{array}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1369.3 (BHistDodecaTupleNameCert carrier admission)**.** *Every accepted* BHistDodecaTupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, H, C, P, L.$$

The twelve rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Lean4 ✓ `BEDC.Derived.BHistDodecaTupleNameCertUp.BHistDodecaTupleNameCertCarrierAdmission`

Proof. Project the carrier of Definition 1369.1. The definition lists the twelve tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed thirteenth component or detached structural table lies outside the accepted surface. □

Theorem 1369.4 (BHistDodecaTupleNameCert classifier exactness)**.** *For accepted* BHistDodecaTupleNameCert$^{\uparrow}$ *packets T and T', the classifier of Definition 1369.2 is exact for the arity-twelve carrier surface: it compares the twelve tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Lean4 ✓ `BEDC.Derived.BHistArity16EventFlowNameCertUp.BHistArity16EventFlowNameCertTasteGate single carrier alignment`

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_{11}$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1369.5 (BHistDodecaTupleNameCert scoped support package)**.** *The carrier admission and classifier exactness clauses bind the* BHistDodecaTupleNameCert$^{\uparrow}$ *packet to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

and to the representative sibling carrier families cited above. The resulting scope is the arity-twelve tag-encoded BHist *packet with displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden thirteenth row.*

Proof. Apply Definition 1369.3 to identify the admitted rows and exclude any thirteenth coordinate. Apply Definition 1369.4 to keep comparison on the twelve tagged BHist rows and the displayed structural rows.

The support package is therefore the named BEDC packet itself. Its provenance can source the cited bucket siblings, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, a thirteenth carrier row, or bucket membership alone fails carrier admission or classifier exactness for $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$.

Independence witness: b-hist-octa-tuple-name-cert, b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, b-hist-hendeca-tuple-name-cert, and the cited bucket siblings: none is bijective to $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$ because this packet carries twelve tagged rows together with transport, replay, provenance, and local NameCert in one packet, while the neighboring capstones carry different arities and each sibling supplies only a member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistDodecaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistDodecaTupleNameCert}^{\uparrow}$ packet is generated from twelve tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistDodecaTupleNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1369.1, Definition 1369.2, Definition 1369.3, Definition 1369.4, and Definition 1369.5 identify the arity-twelve carrier-isomorphism surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistDodecaTupleNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistDodecaTupleNameCertUp and checked carrier-admission plus classifier-exactness theorems.

Chapter 1370

A Concrete Naming Certificate for $\mathsf{BHistArity13TupleNameCert}^{\uparrow}$

The $\mathsf{BHistArity13TupleNameCert}^{\uparrow}$ packet is the arity-thirteen capstone for the carrier-isomorphism bucket displayed by the neighboring tuple chapters chapter 1365, chapter 1363, chapter 1364, chapter 1366, and chapter 1369. It is composite over representative bucket siblings including $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ from chapter 834, $\mathsf{AnalogyCertificateGate}^{\uparrow}$ from chapter 1092, $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ from chapter 1118, $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ from chapter 1162, $\mathsf{FiniteObservationRefutation}^{\uparrow}$ from chapter 1169, and $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ from chapter 1109. The chapter gives the thirteen-row carrier a single BEDC name so downstream consumers can cite the tag, comparison, transport, replay, provenance, and local-name surface without copying the member-local tuple schema.

Definition 1370.1 (BHistArity13TupleNameCert carrier)**.** *A* $\mathsf{BHistArity13TupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{12}$ *are the thirteen tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local carrier role used by a cited bucket sibling and whose payload is read only as a displayed BEDC history component.*
2. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{12}, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same thirteen-row packet.*
4. P *is the* Pkg *provenance over the arity-thirteen tag encoder, the cited tuple capstones, the representative bucket siblings named above,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistArity13TupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a fourteenth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Definition 1370.2 (BHistArity13TupleNameCert classifier)**.** *Two* $\mathsf{BHistArity13TupleNameCert}^{\uparrow}$ *packets*

$$T = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, H, C, P, L)$$

and

$$T' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, R'_{11}, R'_{12}, H', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{aligned}
&R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \quad R_4 \sim R'_4, \quad R_5 \sim R'_5,\\
&R_6 \sim R'_6, \quad R_7 \sim R'_7, \quad R_8 \sim R'_8, \quad R_9 \sim R'_9, \quad R_{10} \sim R'_{10}, \quad R_{11} \sim R'_{11},\\
&R_{12} \sim R'_{12},
\end{aligned}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1370.3 (BHistArity13TupleNameCert carrier admission)**.** *Every accepted* BHistArity13TupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, H, C, P, L.$$

The thirteen rows are tag-encoded BHist *components, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Lean4 ✓ `BEDC.Derived.BHistArity13TupleNameCertUp.BHistArity13TupleNameCertCarrierAdmission`

Proof. Project the carrier of Definition 1370.1. The definition lists the thirteen tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed fourteenth component or detached structural table lies outside the accepted surface. □

Theorem 1370.4 (BHistArity13TupleNameCert classifier exactness)**.** *For accepted* BHistArity13TupleNameCert$^{\uparrow}$ *packets T and T', the classifier of Definition 1370.2 is exact for the arity-thirteen carrier surface: it compares the thirteen tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_{12}$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1370.5 (BHistArity13TupleNameCert tag exhaustion)**.** *For an accepted* BHistArity13TupleNameCert$^{\uparrow}$ *packet, the tags carried by $R_0, \ldots, R_{12}$ exhaust the carrier rows available to the local* NameCert *interface. Any consumer may transport or replay these rows only through the displayed H and C rows, and cannot manufacture an unlisted tag, collapse two tags into one bucket witness, import a carrier row from a sibling capstone, or read a fourteenth carrier component.*

Proof. Project the carrier of Definition 1370.1. The carrier has exactly the thirteen tag-encoded BHist rows together with H, C, P, L, and Definition 1370.3 identifies this displayed list as the whole accepted row inventory.

The transport and replay access paths are therefore already fixed by H and C. No consumer can introduce another tag or merge two tags into a single bucket witness, because such a move would require an additional carrier row or a non-displayed transport clause.

Finally apply Definition 1370.4. Every public comparison remains componentwise over the same thirteen tags and the same displayed structural rows, so sibling capstones cannot contribute hidden carrier rows and no fourteenth component is readable through the local NameCert surface. □

Theorem 1370.6 (BHistArity13TupleNameCert scoped support package)**.** *The carrier admission and classifier exactness clauses bind the* BHistArity13TupleNameCert$^{\uparrow}$ *packet to the displayed BEDC support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

to the tuple capstones from arity eight through arity twelve, and to the representative bucket members AuditMapRouteCompiler$^{\uparrow}$*,* AnalogyCertificateGate$^{\uparrow}$*,* AxisCarryDiamondRoute$^{\uparrow}$*,* CompressionLedgerFunctor$^{\uparrow}$*,* FiniteObservationRefutation$^{\uparrow}$*, and* FocusedMultiHistLedger$^{\uparrow}$*. The resulting scope is the arity-thirteen tag-encoded* BHist *packet with displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden fourteenth row.*

Proof. Apply Definition 1370.3 to identify the admitted rows and exclude any fourteenth coordinate. Apply Definition 1370.4 to keep comparison on the thirteen tagged BHist rows and the displayed structural rows.

Apply Definition 1370.5 to rule out unlisted tags, row fusion, sibling-carrier imports, and hidden components at the local NameCert interface.

The support package is therefore the named BEDC packet itself. Its provenance can source the cited tuple capstones and representative bucket members, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, a fourteenth carrier row, bucket membership alone, a provenance table detached from P, or a sibling-capstone rewrite fails carrier admission or classifier exactness for BHistArity13TupleNameCert$^{\uparrow}$.

Independence witness: b-hist-octa-tuple-name-cert, b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, b-hist-hendeca-tuple-name-cert, and b-hist-dodeca-tuple-name-cert: none is bijective to BHistArity13TupleNameCert$^{\uparrow}$ because this packet carries thirteen tagged rows together with transport, replay, provenance, and local NameCert in one packet, while the neighboring capstones carry different arities and each sibling supplies only a member-family surface.

Closure status for NameCert$_{\mathsf{BHistArity13TupleNameCert}^{\uparrow}}$**.**

Constructive story: A BHistArity13TupleNameCert$^{\uparrow}$ packet is generated from thirteen tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BHistArity13TupleNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 1370.1, Definition 1370.2, Definition 1370.3, Definition 1370.4, Definition 1370.5, and Definition 1370.6 identify the arity-thirteen carrier-isomorphism surface and its BEDC support package.

Formal verification: FormalStatus(NameCert$_{\mathsf{BHistArity13TupleNameCert}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistArity13TupleNameCertUp and checked carrier-admission plus classifier-exactness theorems.

Chapter 1371

A Concrete Naming Certificate for $\mathsf{BHistArity13SequenceNameCert}^{\uparrow}$

The $\mathsf{BHistArity13SequenceNameCert}^{\uparrow}$ packet is the ordered arity-thirteen sequence capstone for the carrier-isomorphism bucket. It is composite over the sequence-window certificates of chapter 1367, chapter 1368, and chapter 1377, together with the unordered arity-thirteen tuple packet of chapter 1370. It also reads representative bucket members $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ from chapter 834, $\mathsf{AnalogyCertificateGate}^{\uparrow}$ from chapter 1092, $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ from chapter 1118, $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ from chapter 1162, $\mathsf{FiniteObservationRefutation}^{\uparrow}$ from chapter 1169, and $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ from chapter 1109. The chapter gives downstream consumers a single BEDC name for the ordered thirteen-row surface instead of requiring them to cite lower-arity sequence windows or the unordered tuple packet.

Definition 1371.1 (BHistArity13SequenceNameCert carrier)**.** *A* $\mathsf{BHistArity13SequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{12}$ *are the thirteen tag-encoded* BHist *rows admitted by the ordered window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite sequence index and order row. It records the first admitted position, the last admitted position, the successor route through all thirteen entries, and the rule that every sequence read factors through exactly one of* $R_0, \ldots, R_{12}$.
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{12}, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the thirteen-row sequence encoder, the sequence and tuple capstones cited above, the representative bucket members cited above,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistArity13SequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a fourteenth sequence component, a hidden permutation, an order table detached from I, unordered bucket membership alone, a host equality test, or a replacement of any tuple or sequence capstone chapter.

Definition 1371.2 (BHistArity13SequenceNameCert classifier)**.** *Two* BHistArity13SequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, R'_{11}, R'_{12}, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed sequence row compares componentwise,

$$\begin{aligned}
&R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \quad R_4 \sim R'_4, \quad R_5 \sim R'_5,\\
&R_6 \sim R'_6, \quad R_7 \sim R'_7, \quad R_8 \sim R'_8, \quad R_9 \sim R'_9, \quad R_{10} \sim R'_{10}, \quad R_{11} \sim R'_{11},\\
&R_{12} \sim R'_{12},
\end{aligned}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from ordered window admission through local naming. No comparison is available through an omitted position, an untagged payload, unordered tuple membership, hidden permutation, sibling-capstone rewrite, or host equality.*

Theorem 1371.3 (BHistArity13SequenceNameCert carrier admission)**.** *Every accepted* BHistArity13SequenceNam *packet admits exactly the displayed rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, I, H, C, P, L.$$

The thirteen rows are tag-encoded BHist *components, I is the only finite sequence index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Lean4 ✓ `BEDC.Derived.BHistArity13SequenceNameCertUp.BHistArity13SequenceNameCert single carrier alignment`

Proof. Project the carrier of Definition 1371.1. The definition lists the thirteen tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed fourteenth component, detached order table, hidden permutation, or off-packet structural row lies outside the accepted surface. □

Theorem 1371.4 (BHistArity13SequenceNameCert classifier exactness)**.** *For accepted* BHistArity13SequenceName *packets S and S', the classifier of Definition 1371.2 is exact for the arity-thirteen ordered sequence surface: it compares the thirteen* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, or host equality.*

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1371.5 (BHistArity13SequenceNameCert index exhaustion)**.** *For an accepted* BHistArity13SequenceNam *packet, the index row I exhausts the orderable positions readable by the local* NameCert *interface. Every stable carrier-isomorphism read preserves*

$$R_0, \ldots, R_{12}, I, H, C, P, L$$

as displayed fields of the same sequence-window packet, and no consumer can manufacture an omitted position, erase the order row, collapse two positions into unordered tuple membership, or read a fourteenth sequence component.

Proof. Use the I and H rows in Definition 1371.1. The index row displays the finite order for exactly thirteen entries, and the transport row records componentwise hsame transport for exactly those entries together with the replay, provenance, and local naming rows.

The classifier of Definition 1371.2 repeats the same ordered list and requires the index rows to display the same finite order. A consumer that omits a position, erases I, collapses positions into unordered tuple membership, or adds a fourteenth row is therefore no longer reading the accepted packet. □

Theorem 1371.6 (BHistArity13SequenceNameCert tag exhaustion)**.** *For an accepted* BHistArity13SequenceNameC *packet, the tags carried by $R_0, \ldots, R_{12}$ exhaust the sequence rows available to the local* NameCert *interface. Any consumer may transport or replay these rows only through the displayed I, H, and C rows, and cannot manufacture an unlisted tag, collapse two tags into one bucket witness, import a carrier row from a sibling capstone, alter the displayed order, or read a fourteenth sequence component.*

Proof. Project the carrier of Definition 1371.1. The carrier has exactly the thirteen tag-encoded BHist rows together with I, H, C, P, L, and Definition 1371.3 identifies this displayed list as the whole accepted row inventory.

The index, transport, and replay access paths are therefore already fixed by I, H, and C. No consumer can introduce another tag, change the displayed order, or merge two tags into a single bucket witness, because such a move would require an additional carrier row or a non-displayed structural clause.

Finally apply Definition 1371.4. Every public comparison remains indexwise over the same thirteen tags and the same displayed structural rows, so sibling capstones cannot contribute hidden carrier rows and no fourteenth component is readable through the local NameCert surface. □

Theorem 1371.7 (BHistArity13SequenceNameCert scoped support package)**.** *The carrier admission, classifier exactness, index exhaustion, and tag exhaustion clauses bind the* BHistArity13SequenceNameCert$^{\uparrow}$ *packet exactly to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

to the ordered sequence capstones chapter 1367, chapter 1368, and chapter 1377, to the unordered arity-thirteen tuple capstone chapter 1370, and to the representative bucket members AuditMapRouteCompiler$^{\uparrow}$, AnalogyCertificateGate$^{\uparrow}$, AxisCarryDiamondRoute$^{\uparrow}$, CompressionLedgerFunctor$^{\uparrow}$, FiniteObservationRefutation$^{\uparrow}$, *and* FocusedMultiHistLedger$^{\uparrow}$. *The resulting scope is the ordered arity-thirteen* BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a host-side equality classifier, a hidden permutation, or a hidden fourteenth position.*

Proof. Apply Definition 1371.3 to identify the admitted rows and exclude any fourteenth coordinate. Apply Definition 1371.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1371.5 to force all stable reads through the same ordered hsame and Cont support. Apply Definition 1371.6 to rule out unlisted tags, order changes, row fusion, sibling-carrier imports, and hidden components at the local NameCert interface. The cited sequence capstones and the arity-thirteen tuple packet can source the provenance context, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates.

The support package is therefore the named BEDC packet itself. Its local NameCert row exposes only the displayed thirteen-row sequence surface, so the scoped package has no route to an unordered bucket tuple, host-side equality classifier, hidden permutation, or hidden fourteenth position. □

Falsifiable prediction: Any accepted consumer route that reads unordered tuple membership as the classifier, an untagged row, a hidden permutation, a fourteenth sequence row, a host equality classifier, an order table detached from I, an unlisted tag, or a sibling capstone rewrite fails carrier admission, classifier exactness, index exhaustion, or tag exhaustion for BHistArity13SequenceNameCert$^{\uparrow}$.

Independence witness: b-hist-nona-sequence-name-cert, b-hist-deca-sequence-name-cert, b-hist-hendeca-sequence-name-cert, and b-hist-arity13-tuple-namecert: none is bijective to BHistArity13SequenceNameCer because this packet carries thirteen ordered payload rows plus an index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply lower-arity sequence windows or an unordered arity-thirteen tuple surface.

Closure status for NameCert$_{\text{BHistArity13SequenceNameCert}^{\uparrow}}$.

Constructive story: A BHistArity13SequenceNameCert$^{\uparrow}$ packet is generated from thirteen tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BHistArity13SequenceNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 1371.1, Definition 1371.2, Definition 1371.3, Definition 1371.4, Definition 1371.5, Definition 1371.6, and Definition 1371.7 identify the arity-thirteen sequence-window carrier surface and its BEDC support package.

Formal verification: FormalStatus(NameCert$_{\text{BHistArity13SequenceNameCert}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a standard bridge, tuple-chapter replacement, sequence-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires checked classifier-exactness, index-exhaustion, tag-exhaustion, and scoped-support theorems for BEDC.Derived.BHistArity13SequenceNameCertUp.

Chapter 1372

A Concrete Naming Certificate for $\mathsf{BHistArity14SequenceNameCert}^{\uparrow}$

The $\mathsf{BHistArity14SequenceNameCert}^{\uparrow}$ packet is the ordered arity-fourteen sequence capstone for the carrier-isomorphism bucket. It is composite over the sequence-window certificates of chapter 1367, chapter 1368, chapter 1377, and chapter 1371, together with the unordered arity-thirteen tuple packet of chapter 1370. It also reads representative bucket members $\mathsf{AuditMapRouteCompiler}^{\uparrow}$ from chapter 834, $\mathsf{AnalogyCertificateGate}^{\uparrow}$ from chapter 1092, $\mathsf{AxisCarryDiamondRoute}^{\uparrow}$ from chapter 1118, $\mathsf{CompressionLedgerFunctor}^{\uparrow}$ from chapter 1162, $\mathsf{FiniteObservationRefutation}^{\uparrow}$ from chapter 1169, and $\mathsf{FocusedMultiHistLedger}^{\uparrow}$ from chapter 1109. The chapter gives downstream consumers a single BEDC name for the ordered fourteen-row surface instead of requiring them to cite lower-arity sequence windows or the unordered tuple packet.

Definition 1372.1 (BHistArity14SequenceNameCert carrier)**.** *A* $\mathsf{BHistArity14SequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{13}$ *are the fourteen tag-encoded* BHist *rows admitted by the ordered window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite sequence index and order row. It records the first admitted position, the last admitted position, the successor route through all fourteen entries, and the rule that every sequence read factors through exactly one of* $R_0, \ldots, R_{13}$.
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{13}, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the fourteen-row sequence encoder, the sequence and tuple capstones cited above, the representative bucket members cited above,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistArity14SequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a fifteenth sequence component, a hidden permutation, an order table detached from I, unordered bucket membership alone, a host equality test, or a replacement of any tuple or sequence capstone chapter.

Definition 1372.2 (BHistArity14SequenceNameCert classifier)**.** *Two* BHistArity14SequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, R'_{11}, R'_{12}, R'_{13}, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed sequence row compares componentwise,

$$\begin{aligned}
&R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \quad R_4 \sim R'_4, \quad R_5 \sim R'_5,\\
&R_6 \sim R'_6, \quad R_7 \sim R'_7, \quad R_8 \sim R'_8, \quad R_9 \sim R'_9, \quad R_{10} \sim R'_{10}, \quad R_{11} \sim R'_{11},\\
&R_{12} \sim R'_{12}, \quad R_{13} \sim R'_{13},
\end{aligned}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from ordered window admission through local naming. No comparison is available through an omitted position, an untagged payload, unordered tuple membership, hidden permutation, sibling-capstone rewrite, or host equality.*

Theorem 1372.3 (BHistArity14SequenceNameCert carrier admission)**.** *Every accepted* BHistArity14SequenceNam *packet admits exactly the displayed rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, I, H, C, P, L.$$

The fourteen rows are tag-encoded BHist *components, I is the only finite sequence index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Lean4 ✓ `BEDC.Derived.BHistArity14SequenceNameCertUp.BHistArity14SequenceNameCertCarrierAdmission`

Proof. Project the carrier of Definition 1372.1. The definition lists the fourteen tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed fifteenth component, detached order table, hidden permutation, or off-packet structural row lies outside the accepted surface. □

Theorem 1372.4 (BHistArity14SequenceNameCert classifier exactness)**.** *For accepted* BHistArity14SequenceName *packets S and S', the classifier of Definition 1372.2 is exact for the arity-fourteen ordered sequence surface: it compares the fourteen* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, or host equality.*

Lean4 ✓ `BEDC.Derived.BHistArity14SequenceNameCertUp.BHistArity14SequenceNameCertClassifier exactness`

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1372.5 (BHistArity14SequenceNameCert index exhaustion)**.** *For an accepted* BHistArity14SequenceNam *packet, the index row I exhausts the orderable positions readable by the local* NameCert *interface. Every stable carrier-isomorphism read preserves*

$$R_0, \ldots, R_{13}, I, H, C, P, L$$

as displayed fields of the same sequence-window packet, and no consumer can manufacture an omitted position, erase the order row, collapse two positions into unordered tuple membership, or read a fifteenth sequence component.

Proof. Use the I and H rows in Definition 1372.1. The index row displays the finite order for exactly fourteen entries, and the transport row records componentwise hsame transport for exactly those entries together with the replay, provenance, and local naming rows.

The classifier of Definition 1372.2 repeats the same ordered list and requires the index rows to display the same finite order. A consumer that omits a position, erases I, collapses positions into unordered tuple membership, or adds a fifteenth row is therefore no longer reading the accepted packet. □

Theorem 1372.6 (BHistArity14SequenceNameCert tag exhaustion)**.** *For an accepted* BHistArity14SequenceNameC *packet, the tags carried by $R_0, \ldots, R_{13}$ exhaust the sequence rows available to the local* NameCert *interface. Any consumer may transport or replay these rows only through the displayed I, H, and C rows, and cannot manufacture an unlisted tag, collapse two tags into one bucket witness, import a carrier row from a sibling capstone, alter the displayed order, or read a fifteenth sequence component.*

Proof. Project the carrier of Definition 1372.1. The carrier has exactly the fourteen tag-encoded BHist rows together with I, H, C, P, L, and Definition 1372.3 identifies this displayed list as the whole accepted row inventory.

The index, transport, and replay access paths are therefore already fixed by I, H, and C. No consumer can introduce another tag, change the displayed order, or merge two tags into a single bucket witness, because such a move would require an additional carrier row or a non-displayed structural clause.

Finally apply Definition 1372.4. Every public comparison remains indexwise over the same fourteen tags and the same displayed structural rows, so sibling capstones cannot contribute hidden carrier rows and no fifteenth component is readable through the local NameCert surface. □

Theorem 1372.7 (BHistArity14SequenceNameCert scoped support package)**.** *The carrier admission, classifier exactness, index exhaustion, and tag exhaustion clauses bind the* BHistArity14SequenceNameCert$^{\uparrow}$ *packet exactly to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

to the ordered sequence capstones chapter 1367, chapter 1368, chapter 1377, and chapter 1371, to the unordered arity-thirteen tuple capstone chapter 1370, and to the representative bucket members AuditMapRouteCompiler$^{\uparrow}$, AnalogyCertificateGate$^{\uparrow}$, AxisCarryDiamondRoute$^{\uparrow}$, CompressionLedgerFunctor$^{\uparrow}$, FiniteObservationRefutation$^{\uparrow}$, *and* FocusedMultiHistLedger$^{\uparrow}$. *The resulting scope is the ordered arity-fourteen* BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a host-side equality classifier, a hidden permutation, or a hidden fifteenth position.*

Proof. Apply Definition 1372.3 to identify the admitted rows and exclude any fifteenth coordinate. Apply Definition 1372.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1372.5 to force all stable reads through the same ordered hsame and Cont support. Apply Definition 1372.6 to rule out unlisted tags, order changes, row fusion, sibling-carrier imports, and hidden components at the local NameCert interface. The cited sequence capstones and the arity-thirteen tuple packet can source the provenance context, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates.

The support package is therefore the named BEDC packet itself. Its local NameCert row exposes only the displayed fourteen-row sequence surface, so the scoped package has no route to an unordered bucket tuple, host-side equality classifier, hidden permutation, or hidden fifteenth position. □

Falsifiable prediction: Any accepted consumer route that reads unordered tuple membership as the classifier, an untagged row, a hidden permutation, a fifteenth sequence row, a host equality classifier, an order table detached from I, an unlisted tag, or a sibling capstone rewrite fails carrier admission, classifier exactness, index exhaustion, or tag exhaustion for BHistArity14SequenceNameCert$^{\uparrow}$.

Independence witness: b-hist-nona-sequence-name-cert, b-hist-deca-sequence-name-cert, b-hist-hendeca-sequence-name-cert, b-hist-arity13-sequence-name-cert, and b-hist-arity13-tuple-namecert: none is bijective to BHistArity14SequenceNameCert$^{\uparrow}$ because this packet carries fourteen ordered payload rows plus an index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply lower-arity sequence windows or an unordered arity-thirteen tuple surface.

Closure status for NameCert$_{\text{BHistArity14SequenceNameCert}^{\uparrow}}$.

Constructive story: A BHistArity14SequenceNameCert$^{\uparrow}$ packet is generated from fourteen tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BHistArity14SequenceNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 1372.1, Definition 1372.2, Definition 1372.3, Definition 1372.4, Definition 1372.5, Definition 1372.6, and Definition 1372.7 identify the arity-fourteen sequence-window carrier surface and its BEDC support package.

Formal verification: FormalStatus(NameCert$_{\text{BHistArity14SequenceNameCert}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a standard bridge, tuple-chapter replacement, sequence-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires checked classifier-exactness, index-exhaustion, tag-exhaustion, and scoped-support theorems for BEDC.Derived.BHistArity14SequenceNameCertUp.

Chapter 1373

A Concrete Naming Certificate for $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$

The $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$ packet is the ordered arity-fifteen sequence capstone for the carrier-isomorphism bucket. It is composite over the ordered sequence capstones of chapter 1367, chapter 1368, chapter 1377, chapter 1371, and chapter 1372, together with the unordered arity-thirteen tuple packet of chapter 1370. It also reads the bucket members $\mathsf{CauchySharedTailModulus}^{\uparrow}$ from chapter 983, $\mathsf{FocusedLedger}^{\uparrow}$ from chapter 1247, $\mathsf{PhysicalModelAudit}^{\uparrow}$ from chapter 1129, $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$ from chapter 1314, and $\mathsf{TowerEquivalence}^{\uparrow}$ from chapter 1143. The chapter gives downstream consumers a single BEDC name for the ordered fifteen-row surface without treating any constituent sibling as a replacement carrier.

Definition 1373.1 (BHistArity15SequenceNameCert carrier)**.** *A* $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, R_{14}, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{14}$ *are the fifteen tag-encoded* BHist *rows admitted by the ordered window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite sequence index and order row. It records the first admitted position, the last admitted position, the successor route through all fifteen entries, and the rule that every sequence read factors through exactly one of* $R_0, \ldots, R_{14}$.
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{14}, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the fifteen-row sequence encoder, the sequence and tuple capstones cited above,* $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{ProgrammeStrengthGradientLedge}$ $\mathsf{TowerEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistArity15SequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a sixteenth sequence component, a hidden permutation, an order table detached from I, unordered bucket membership alone, a host equality test, or a replacement of any tuple, sequence, or bucket-member chapter.

Lean4 ✓ `BEDC.Derived.BHistArity16SequenceNameCertUp.BHistArity16SequenceNameCertTasteGate`
`single carrier alignment`

Definition 1373.2 (BHistArity15SequenceNameCert classifier)**.** *Two* BHistArity15SequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, R_{14}, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, R'_7, R'_8, R'_9, R'_{10}, R'_{11}, R'_{12}, R'_{13}, R'_{14}, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed sequence row compares componentwise,

$$\begin{aligned}
&R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \quad R_4 \sim R'_4, \quad R_5 \sim R'_5,\\
&R_6 \sim R'_6, \quad R_7 \sim R'_7, \quad R_8 \sim R'_8, \quad R_9 \sim R'_9, \quad R_{10} \sim R'_{10}, \quad R_{11} \sim R'_{11},\\
&R_{12} \sim R'_{12}, \quad R_{13} \sim R'_{13}, \quad R_{14} \sim R'_{14},
\end{aligned}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from ordered window admission through local naming. No comparison is available through an omitted position, an untagged payload, unordered tuple membership, hidden permutation, sibling-capstone rewrite, bucket-member replacement, or host equality.*

Theorem 1373.3 (BHistArity15SequenceNameCert carrier admission)**.** *Every accepted* BHistArity15SequenceNam *packet admits exactly the displayed rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R_{11}, R_{12}, R_{13}, R_{14}, I, H, C, P, L.$$

The fifteen rows are tag-encoded BHist *components, I is the only finite sequence index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Proof. Project the carrier of Definition 1373.1. The definition lists the fifteen tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed sixteenth component, detached order table, hidden permutation, or off-packet structural row lies outside the accepted surface. □

Lean4 ✓ `BEDC.Derived.BHistArity15SequenceNameCertUp.BHistArity15SequenceNameCertCarrierAdmission`

Theorem 1373.4 (BHistArity15SequenceNameCert classifier exactness)**.** *For accepted* BHistArity15SequenceName *packets S and S', the classifier of Definition 1373.2 is exact for the arity-fifteen ordered sequence surface: it compares the fifteen* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, bucket-member replacement, or host equality.*

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered tuple membership, hidden permutation, detached order table, sibling-capstone rewrite, bucket membership alone, bucket-member replacement, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1373.5 (BHistArity15SequenceNameCert index exhaustion)**.** *For an accepted* BHistArity15SequenceNam *packet, the index row I exhausts the orderable positions readable by the local* NameCert *interface. Every stable carrier-isomorphism read preserves*

$$R_0, \ldots, R_{14}, I, H, C, P, L$$

as displayed fields of the same sequence-window packet, and no consumer can manufacture an omitted position, erase the order row, collapse two positions into unordered tuple membership, replace the packet by a bucket member, or read a sixteenth sequence component.

Proof. Use the I and H rows in Definition 1373.1. The index row displays the finite order for exactly fifteen entries, and the transport row records componentwise hsame transport for exactly those entries together with the replay, provenance, and local naming rows.

The classifier of Definition 1373.2 repeats the same ordered list and requires the index rows to display the same finite order. A consumer that omits a position, erases I, collapses positions into unordered tuple membership, replaces the packet by a bucket member, or adds a sixteenth row is therefore no longer reading the accepted packet. □

Theorem 1373.6 (BHistArity15SequenceNameCert tag exhaustion)**.** *For an accepted* BHistArity15SequenceNameC *packet, the tags carried by $R_0, \ldots, R_{14}$ exhaust the sequence rows available to the local* NameCert *interface. Any consumer may transport or replay these rows only through the displayed I, H, and C rows, and cannot manufacture an unlisted tag, collapse two tags into one bucket witness, import a carrier row from a sibling capstone, replace the packet by* CauchySharedTailModulus$^{\uparrow}$, FocusedLedger$^{\uparrow}$, PhysicalModelAudit$^{\uparrow}$, ProgrammeStrengthGradientLedger$^{\uparrow}$, *or* TowerEquivalence$^{\uparrow}$, *alter the displayed order, or read a sixteenth sequence component.*

Proof. Project the carrier of Definition 1373.1. The carrier has exactly the fifteen tag-encoded BHist rows together with I, H, C, P, L, and Definition 1373.3 identifies this displayed list as the whole accepted row inventory.

The index, transport, and replay access paths are therefore already fixed by I, H, and C. No consumer can introduce another tag, change the displayed order, merge two tags into a single bucket witness, or replace the packet by one of the cited bucket members, because such a move would require an additional carrier row or a non-displayed structural clause.

Finally apply Definition 1373.4. Every public comparison remains indexwise over the same fifteen tags and the same displayed structural rows, so sibling capstones and bucket members cannot contribute hidden carrier rows and no sixteenth component is readable through the local NameCert surface. □

Theorem 1373.7 (BHistArity15SequenceNameCert scoped support package)**.** *The carrier admission, classifier exactness, index exhaustion, and tag exhaustion clauses bind the* BHistArity15SequenceNameCert$^{\uparrow}$

packet exactly to the displayed BEDC support

$$\mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert},$$

to the ordered sequence capstones chapter 1367, chapter 1368, chapter 1377, chapter 1371, and chapter 1372, to the unordered arity-thirteen tuple capstone chapter 1370, and to the bucket members $\mathsf{CauchySharedTailModulus}^{\uparrow}$, $\mathsf{FocusedLedger}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{ProgrammeStrengthGradientLedger}^{\uparrow}$, *and* $\mathsf{TowerEquivalence}^{\uparrow}$. *The resulting scope is the ordered arity-fifteen* BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a host-side equality classifier, a hidden permutation, a bucket-member carrier, or a hidden sixteenth position.*

Proof. Apply Definition 1373.3 to identify the admitted rows and exclude any sixteenth coordinate. Apply Definition 1373.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1373.5 to force all stable reads through the same ordered hsame and Cont support. Apply Definition 1373.6 to rule out unlisted tags, order changes, row fusion, sibling-carrier imports, bucket-member replacement, and hidden components at the local NameCert interface. The cited sequence capstones, the arity-thirteen tuple packet, and the listed bucket members can source the provenance context, but no carrier or classifier row rewrites those siblings or replaces their local certificates.

The support package is therefore the named BEDC packet itself. Its local NameCert row exposes only the displayed fifteen-row sequence surface, so the scoped package has no route to an unordered bucket tuple, host-side equality classifier, hidden permutation, bucket-member carrier, or hidden sixteenth position. □

Falsifiable prediction: Any accepted consumer route that reads unordered tuple membership as the classifier, an untagged row, a hidden permutation, a sixteenth sequence row, a host equality classifier, an order table detached from I, an unlisted tag, a sibling capstone rewrite, or a bucket-member replacement fails carrier admission, classifier exactness, index exhaustion, or tag exhaustion for $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$.

Independence witness: b-hist-nona-sequence-name-cert, b-hist-deca-sequence-name-cert, b-hist-hendeca-sequence-name-cert, b-hist-arity13-sequence-name-cert, b-hist-arity14-sequence-name-cert, b-hist-arity13-tuple-namecert, cauchy-shared-tail-modulus, focused-ledger, physical-model-audit, programme-strength-gradient-ledger, and towerequivalence: none is bijective to $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$ because this packet carries fifteen ordered payload rows plus an index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply lower-arity sequence windows, an unordered arity-thirteen tuple surface, or bucket-member carriers.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistArity15SequenceNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistArity15SequenceNameCert}^{\uparrow}$ packet is generated from fifteen tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistArity15SequenceNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1373.1, Definition 1373.2, Definition 1373.3, Definition 1373.4, Definition 1373.5, Definition 1373.6, and Definition 1373.7 identify the arity-fifteen sequence-window carrier surface and its BEDC support package.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BHistArity15SequenceNameCert}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a standard bridge, bucket-member replacement, tuple-chapter replacement, sequence-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires checked classifier-exactness, index-exhaustion, tag-exhaustion, and scoped-support theorems for BEDC.Derived.BHistArity15SequenceNameCertUp.

Chapter 1374

A Concrete Naming Certificate for $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$

The $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ packet realizes the stop-codon suffix bridge of subsection 9462.28.18 as finite BEDC data. It sits beside the finite GAP discipline of chapter 504, the inscription handoff of chapter 610, the Zeckendorf normal-word surface of chapter 278, and the channel/genetic-code bridge of subsection 9462.28.16. Its distinguishing role is a codon-window certificate: the packet records a two-axis RNA or DNA readback, the $\mathrm{U} + \mathrm{RR}$ square, the family-suffix projection, the no-11 Zeckendorf row, the standard-code bridge row, and the cannot-claim boundary without identifying any of these rows with the BEDC channel terminator.

Definition 1374.1 (StopCodonZeckendorfSuffix carrier)**.** *A* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (A, W, S, F, L, T, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the two-axis base-atlas row naming the chosen RNA or DNA notation, the shape axis, the family axis, and the external readback of* U, C, A, *and* G.
2. *W is the three-base codon-window row naming the first-position* U *boundary and the purine shape rows in positions* 2 *and* 3.
3. *S is the family-suffix row projecting positions* 2 *and* 3 *to* 00, 01, 10, *or* 11.
4. *F is the Zeckendorf legality row accepting exactly the length-two suffixes other than* 11.
5. *L is the standard-code bridge row under which* UAA, UAG, *and* UGA *are read as stop codons and* UGG *is read outside the stop set.*
6. *T is the channel-token separation row recording that family suffix* 11 *is not the BEDC channel terminator.*
7. *B is the cannot-claim boundary row refusing biological-code necessity, direct nucleotide denotation by compiled BEDC substrings, and any bridge-free claim that* UGG *is tryptophan.*
8. *H records componentwise* hsame *transport for the atlas, codon window, suffix, legality, standard-code bridge, token-separation, cannot-claim, route, provenance, and local naming rows.*
9. *C records displayed* Cont *routes through A, W, S, F, L, T, B before a consumer may read the stop-codon suffix classifier.*

10. *P is the* Pkg *provenance over* StopCodonZeckendorfSuffix$^{\uparrow}$, ZeckendorfNormal$^{\uparrow}$, FiniteTraceGapSocket$^{\uparrow}$, InscriptionPoint$^{\uparrow}$, *the channel/genetic-code bridge,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}$ *row.*

The carrier has no coordinate for a universal biological genetic-code law, a direct BEDC channel-token interpretation of nucleotide bases, a bridge-free stop-codon table, a release-factor dynamics model, or a Lean-checked biological theorem.

Lean4 (definition) `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixUp`

Theorem 1374.2 (StopCodonZeckendorfSuffix NameCert obligations)**.** *For the carrier of Definition 1374.1, the local* $\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission: every accepted object exposes the tuple $Z = (A, W, S, F, L, T, B, H, C, P, N)$ and no hidden biological-code table;*
2. *suffix classifier: the route through A, W, S, F accepts exactly the* $\mathrm{U} + \mathrm{RR}$ *codon windows whose family suffix is* 00, 01, *or* 10*;*
3. *standard bridge ledger: the row L is the only source that reads the surviving codons as* UAA, UAG, *and* UGA *in the standard genetic code;*
4. *channel separation: the row T keeps the excluded family suffix* 11 *separate from the BEDC channel terminator and from any compiled channel substring;*
5. *cannot-claim ledger: the rows B, H, C, P, N exhaust transport, replay, provenance, and local naming without exporting biological necessity, all-code universality, or a bridge-free tryptophan claim for* UGG.

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1374.1. The accepted packet displays the atlas, codon window, suffix, legality, standard bridge, token-separation, cannot-claim, transport, route, provenance, and naming rows as one finite BHist carrier.

The suffix classifier reads the rows in order. The atlas A gives the two-axis base readback, the window row W fixes the $\mathrm{U} + \mathrm{RR}$ square, the suffix row S exposes the two family bits, and the legality row F removes only the 11 corner.

The standard biological reading is not supplied by those four rows. It enters only through L, so the packet can state the standard stop set only after the bridge ledger has been read.

Channel separation is then read from T. Since T is a coordinate distinct from S and from the BEDC channel codec, matching the displayed bits 11 does not license token identity.

Finally inspect B, H, C, P, N. These rows preserve and name only the displayed packet, while B records the listed refusals. No omitted biological-code law or bridge-free codon assignment can be recovered from the route. □

Theorem 1374.3 (StopCodonZeckendorfSuffix no channel-token collapse)**.** *For an accepted* StopCodonZeckendorfS *carrier $Z = (A, W, S, F, L, T, B, H, C, P, N)$, the excluded family suffix* 11 *in the* UGG *corner is a suffix-classifier row, not the BEDC channel terminator. Any consumer that identifies those rows leaves the accepted packet surface.*

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixNoChannelTokenCollapse`

Proof. Project the carrier of Definition 1374.1. The family suffix appears in S, while the channel-token separation is a separate row T and the package row P records the channel/genetic-code bridge only as provenance.

Apply the channel-separation clause of Definition 1374.2. It says that the displayed 11 pattern is read through the family suffix and legality rows, not through the BEDC channel codec.

Thus a consumer may compare roles only through the recorded bridge. A direct token identity would delete T and bypass the provenance row, so it is not a read of the accepted $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ packet. □

Theorem 1374.4 (StopCodonZeckendorfSuffix bridge boundary)**.** *Every biological stop-codon claim exported by an accepted* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *packet factors through the finite route*

$$A \to W \to S \to F \to L \to T \to B \to H \to C \to P \to N.$$

The route reconstructs the standard stop-codon suffix set only relative to the displayed two-axis readback and the separate standard genetic-code bridge. It does not prove biological-code necessity, all genetic-code universality, release-factor dynamics, direct channel-token identity, or bridge-free nucleotide semantics.

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixBridgeBoundary`

Proof. Start with an accepted carrier $Z = (A, W, S, F, L, T, B, H, C, P, N)$. The only codon-window source is W, and it is readable as a biological codon window only after the atlas row A fixes the two-axis base readback.

The suffix calculation then passes through S and F. These rows remove the family-11 corner and therefore isolate the three surviving corners of the $\mathrm{U} + \mathrm{RR}$ square.

The biological stop reading is still conditional. The row L supplies the standard genetic-code bridge, so the stop-codon export factors through that bridge rather than through Zeckendorf legality alone.

The remaining rows record the boundary of the export. The separation row T blocks channel-token collapse, the cannot-claim row B refuses biological necessity and all-code universality, and H, C, P, N preserve only this displayed route. □

Theorem 1374.5 (StopCodonZeckendorfSuffix vision route)**.** *An accepted* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *packet is the concrete finite route for the conditional stop-codon reconstruction problem of Definition 9462.1113. The route displays the* $\mathrm{U} + \mathrm{RR}$ *square, maps the family suffixes* 00*,* 01*,* 10*, and* 11*, excludes exactly the* 11 *corner, and records the refusal of biological-code derivation as kernel data.*

Proof. There are four displayed suffix cases in the square. The 00 case is the UAA corner: it is read through A, W, S and accepted by the legality row F before the standard-code bridge row L may report it as a stop-codon corner.

The 01 case is the UAG corner. It follows the same A, W, S, F route, and the bridge row L is again the only source of biological stop-set readback.

The 10 case is the UGA corner. It is the third accepted suffix of the finite legality row, still separated from the channel-token row by T.

The 11 case is the UGG corner. The legality row F excludes it, and Definition 1374.3 prevents that excluded family suffix from being identified with the BEDC channel terminator.

Finally use the bridge boundary of Definition 1374.4. The cannot-claim row B records that the packet does not derive the biological genetic code, all-code universality, release-factor dynamics,

bridge-free nucleotide semantics, or channel-token identity as kernel data. Thus the concrete packet is exactly the finite route consumed by the vision-level conditional reconstruction problem. □

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfVisionRoute`

Theorem 1374.6 (StopCodonZeckendorfSuffix obligation closure route)**.** *For every accepted* StopCodonZeckendorf *packet, the finite obligation surface is exactly the route*

$$A, W, S, F, L, T, B, H, C, P, N.$$

The rows expose the two-axis base atlas, codon-window boundary, suffix projection, Zeckendorf legality, standard-code bridge, channel-token separation, cannot-claim boundary, componentwise transport, displayed replay, provenance, and local $\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}$ *evidence. No obligation read may bypass one of these rows or replace it by biological-code necessity, all-code universality, release-factor dynamics, bridge-free nucleotide semantics, or BEDC channel-token identity.*

Proof. The carrier of Definition 1374.1 supplies precisely the displayed rows $A, W, S, F, L, T, B, H, C, P, N$. The NameCert obligations of Definition 1374.2 identify carrier admission, suffix classification, bridge use, channel separation, and cannot-claim ledgers as the local obligation surface.

The no-channel-token collapse theorem Definition 1374.3 keeps the excluded family suffix separate from the BEDC terminator. The bridge boundary of Definition 1374.4 forces every biological stop-codon read through the standard-code row L and the boundary rows T, B.

Finally, Definition 1374.5 checks the four suffix cases against the conditional reconstruction problem. Combining these rows leaves exactly the displayed obligation route and no unlisted biological or channel-token claim. □

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixObligationClosureRoute`

Falsifiable prediction: A $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ packet cannot export a biological-code necessity row without a separate standard genetic-code bridge.

Independence witness: FiniteTraceGapSocket, InscriptionPoint, ZeckendorfNormal, channel-genetic bridge rows: none is bijective to $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$. $\mathsf{FiniteTraceGapSocket}^{\uparrow}$ supplies finite GAP discipline without codon-window rows; $\mathsf{InscriptionPoint}^{\uparrow}$ supplies inscription handoff without the two-axis suffix classifier; $\mathsf{ZeckendorfNormal}^{\uparrow}$ supplies no-11 normality without the standard genetic-code bridge; the channel-genetic bridge rows supply cross-layer separation without the local stop-suffix carrier, excluded-corner ledger, and cannot-claim boundary.

The Lean follow-up file `lean4/BEDC/Derived/StopCodonZeckendorfSuffixUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ instances, with field-projection proofs preserving $A, W, S, F, L, T, B, H, C, P, N$ as separate displayed fields.

Theorem 1374.7 (StopCodonZeckendorfSuffix scope grounding)**.** *Every accepted read of a* StopCodonZeckendorfSu *suffix is scoped over the finite rows*

$$A, W, S, F, L, T, B, H, C, P, N$$

together with $\mathsf{ZeckendorfNormal}^{\uparrow}$, $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, $\mathsf{InscriptionPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The read may use the Zeckendorf normal-word row, finite trace-gap socket, and inscription handoff only through those displayed fields, and no accepted suffix read exports a biological-code necessity, all-code universality, release-factor dynamics, bridge-free nucleotide semantics, or BEDC channel-token identity outside the same finite scope.*

Proof. Start with the carrier of Definition 1374.1. It displays the rows $A, W, S, F, L, T, B, H, C, P, N$ as the complete packet surface, with the Zeckendorf, finite trace-gap, inscription, transport, replay, provenance, and local naming dependencies recorded in those fields.

Apply the NameCert obligations and the obligation closure route. These bind carrier admission, suffix classification, bridge use, channel separation, cannot-claim rows, and structural replay to the same finite route.

The no-channel-token collapse and bridge boundary theorems remove every attempt to read biological-code necessity, all-code universality, release-factor dynamics, bridge-free nucleotide semantics, or BEDC channel-token identity outside the displayed packet. Thus every accepted suffix read is grounded in exactly the stated scope. □

Lean4 ✓ `BEDC.Derived.StopCodonZeckendorfSuffixUp.StopCodonZeckendorfSuffixScopeGrounding`

Closure status for $\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}$.

Constructive story: A $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ packet is generated from a two-axis base atlas, codon-window row, family-suffix projection, Zeckendorf legality row, standard-code bridge, channel-token separation, cannot-claim boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1374.4, Definition 1374.5, Definition 1374.6, and Definition 1374.7 give the public export surface: bridge boundary, vision route, obligation-closure route, and scope-grounding rows over the finite stop-codon suffix carrier and five-row NameCert surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, biological-code necessity, all genetic-code universality, release-factor dynamics, bridge-free nucleotide semantics, or BEDC channel-token identity.

Upgrade path: Lean closure requires Nontrivial StopCodonZeckendorfSuffixUp and FieldFaithful StopCodonZeckendorfSuffixUp instances in `lean4/BEDC/Derived/StopCodonZeckendorfSuffixUp/TasteGate.lean`.

Chapter 1375

A Concrete Naming Certificate for $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$

The $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ packet is the ordered seven-row sequence capstone for the carrier-isomorphism bucket. It is anchored by the hepta tuple capstone of chapter 1376 and by the bucket members $\mathsf{EpistemicWitness}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, $\mathsf{PhysicalRecordInvariant}^{\uparrow}$, $\mathsf{StructuralAdjacency}^{\uparrow}$, $\mathsf{UseProcessLimitLedger}^{\uparrow}$, and $\mathsf{ZCarry}^{\uparrow}$. The chapter names the BEDC-native sequence window that reads exactly seven displayed BHist rows together with an index row, componentwise transport, continuation replay, provenance, and local naming; it does not rewrite any bucket member chapter.

Definition 1375.1 (BHistHeptaSequenceNameCert carrier)**.** *A* $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ *carrier is a finite ordered sequence-window packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_6$ *are the seven tag-encoded* BHist *rows admitted by the sequence window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite index and order row. It records the first displayed position, the last displayed position, and the rule that every sequence read factors through one of* $R_0, \ldots, R_6$ *in that order.*
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_6, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the seven-row window encoder, chapter 1376,* $\mathsf{EpistemicWitness}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, $\mathsf{PhysicalRecordInvariant}^{\uparrow}$, $\mathsf{StructuralAdjacency}^{\uparrow}$, $\mathsf{UseProcessLimitLedger}^{\uparrow}$, $\mathsf{ZCarry}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, an eighth sequence component, a hidden permutation, a host-side equality test, unordered bucket membership alone, or a replacement of the hepta tuple capstone or any bucket member chapter.

Definition 1375.2 (BHistHeptaSequenceNameCert classifier)**.** *Two* BHistHeptaSequenceNameCert$^{\uparrow}$ *packets*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, I, H, C, P, L)$$

and

$$S' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, I', H', C', P', L')$$

are classifier-equivalent exactly when each displayed window row compares componentwise,

$$\begin{aligned} &R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \\ &R_4 \sim R'_4, \quad R_5 \sim R'_5, \quad R_6 \sim R'_6, \end{aligned}$$

the index rows display the same finite order, the transport rows display the same hsame *routes for those components and for I, C, P, L, and the replay rows display the same* Cont *route from window admission through local naming. No comparison is available through omitted position, hidden permutation, untagged payload, unordered bucket membership alone, host equality, or sibling-chapter replacement.*

Theorem 1375.3 (BHistHeptaSequenceNameCert carrier admission)**.** *Every accepted* BHistHeptaSequenceNameCe *packet admits exactly the displayed sequence-window rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, I, H, C, P, L.$$

The seven rows are tag-encoded BHist *components, I is the only index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window.*

Proof. Project the carrier of Definition 1375.1. The definition lists the seven tag-encoded BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted sequence read lands in exactly the displayed rows, and any proposed eighth component, detached order table, hidden permutation, or structural row outside the packet lies outside the accepted surface. □

Theorem 1375.4 (BHistHeptaSequenceNameCert classifier exactness)**.** *For accepted* BHistHeptaSequenceNameCe *packets S and S', the classifier of Definition 1375.2 is exact for the ordered seven-row sequence window: it compares the seven* BHist *rows at matching indices, compares the displayed finite order through I, I', compares the displayed* hsame *and* Cont *routes through H, H', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by unordered bucket membership, hidden permutation, detached order table, sibling-chapter replacement, or host equality.*

Proof. Start with the classifier definition. It requires one comparison for each matching pair R_i, R'_i, requires the index rows to display the same finite order, and requires the displayed transport and replay routes to preserve that same ordered list.

The provenance and local naming rows are part of the same sequence-window carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for unordered bucket membership, hidden permutation, detached order table, sibling-chapter replacement, or host equality, the displayed indexwise route is the exact comparison surface. □

Theorem 1375.5 (BHistHeptaSequenceNameCert index exhaustion)**.** *Every admitted index read of a* $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ *packet is exhausted by the seven displayed positions*

$$0, 1, 2, 3, 4, 5, 6.$$

A stable carrier-isomorphism read must preserve the fields

$$R_0, \ldots, R_6, I, H, C, P, L$$

as rows of the same ordered sequence-window packet.

Proof. Use the I row in Definition 1375.1. It records the first displayed position, the last displayed position, and the rule that every sequence read factors through one of the seven rows in order.

The H row records componentwise hsame transport for that exact list, and the classifier repeats the same index-preserving comparison. Thus a read that introduces an eighth index, drops a displayed position, or detaches the order row is no longer a read of the accepted packet. □

Theorem 1375.6 (BHistHeptaSequenceNameCert tag exhaustion)**.** *The tag inventory of an accepted* $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ *packet is exhausted by the seven row tags on* $R_0, \ldots, R_6$ *together with the structural tags for* I, H, C, P, L*. No untagged payload, duplicate hidden row, bucket-only membership tag, or host-side equality tag is exported by the local* $\mathsf{NameCert}$ *row.*

Proof. Read the carrier definition and the classifier definition together. The carrier displays exactly seven payload rows and five structural rows, while the classifier gives comparison clauses only for those displayed rows and the structural routes attached to them.

The local name is the row L of the same packet. Because L has no coordinate outside the displayed carrier and no classifier clause adds an external tag, local naming cannot export an untagged payload, duplicate hidden row, bucket-only membership tag, or host-side equality tag. □

Theorem 1375.7 (BHistHeptaSequenceNameCert provenance readback)**.** *The provenance row* P *of an accepted* $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ *packet reads back only the seven-row window encoder, chapter 1376, the bucket-member surfaces named by* $\mathsf{EpistemicWitness}^{\uparrow}$*,* $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$*,* $\mathsf{PhysicalRecordInvariant}^{\uparrow}$*,* $\mathsf{StructuralAdjacency}^{\uparrow}$*,* $\mathsf{UseProcessLimitLedger}^{\uparrow}$*,* $\mathsf{ZCarry}^{\uparrow}$*, and the local* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *support rows. It does not read a bucket-wide replacement of the hepta tuple capstone or any named bucket-member chapter.*

Proof. Inspect the provenance row in the carrier definition. It records the seven-row window encoder, the named capstone and bucket-member surfaces, and the local BEDC support rows.

The row is a readback ledger for the ordered packet, not a command to alter any cited chapter. Since neither the carrier nor the classifier contains a replacement coordinate, provenance can source the shared bucket context without rewriting any member. □

Theorem 1375.8 (BHistHeptaSequenceNameCert local naming nonescape)**.** *Every local* $\mathsf{NameCert}_{\mathsf{BHistHeptaSequence}}$ *read factors through the displayed packet*

$$S = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, I, H, C, P, L).$$

The read may expose the seven-row sequence window, its finite order, and its componentwise routes, but it cannot export an untagged carrier, an eighth row, a hidden permutation, a host-side equality test, or a relocation of chapter 1376 or any named bucket-member surface.

Proof. The local name is the row L of the carrier. By Definition 1375.3, L is read only together with the seven window rows, the index row, and the displayed structural rows.

Use index exhaustion, tag exhaustion, and the classifier definition. They force any carrier-isomorphism read to preserve the same tag-encoded components, the same finite order, and the same displayed hsame and Cont routes. Since no untagged carrier, eighth row, hidden permutation, host equality test, or capstone relocation row appears in the packet, local naming cannot export one. □

Theorem 1375.9 (BHistHeptaSequenceNameCert scoped support package)**.** *The carrier admission, classifier exactness, index exhaustion, tag exhaustion, provenance readback, and local naming nonescape clauses bind the* $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ *packet exactly to the displayed BEDC support*

$$\mathsf{BHist}, \quad \mathsf{hsame}, \quad \mathsf{Cont}, \quad \mathsf{Pkg}, \quad \mathsf{NameCert},$$

to the hepta tuple capstone of chapter 1376, and to the bucket-member surfaces named by $\mathsf{EpistemicWitness}^{\uparrow}$, $\mathsf{ExternalSupplyAuditRoute}^{\uparrow}$, $\mathsf{PhysicalRecordInvariant}^{\uparrow}$, $\mathsf{StructuralAdjacency}^{\uparrow}$, $\mathsf{UseProcessLimitLedger}^{\uparrow}$, *and* $\mathsf{ZCarry}^{\uparrow}$. *The resulting scope is the ordered seven-row* BHist *sequence window with displayed index row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not an unordered bucket tuple, a hidden permutation, a host equality classifier, or a hidden eighth position.*

Proof. Apply Definition 1375.3 to identify the admitted rows and exclude any eighth coordinate. Apply Definition 1375.4 to keep comparison on matching sequence indices, the displayed index row, and the structural rows.

Apply Definition 1375.5 to force every stable read through the seven displayed positions and the same ordered support. Apply Definition 1375.6 to prevent untagged payloads, duplicate hidden rows, bucket-only membership tags, and host-side equality tags from escaping through the local name.

Apply Definition 1375.7 and Definition 1375.8. The scoped package is therefore the named BEDC packet itself. Its provenance can source the hepta tuple capstone and the named bucket-member surfaces, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an unordered tuple coordinate, an untagged row, an eighth sequence row, a hidden permutation, bucket membership alone, an order table detached from I, or a sibling capstone relocation fails carrier admission, classifier exactness, index exhaustion, tag exhaustion, provenance readback, or local naming nonescape for $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$.

Independence witness: b-hist-hepta-tuple-name-cert, epistemic-witness, external-supply-audit-route, physical-record-invariant, structural-adjacency, use-process-limit-ledger, and z-carry: none is bijective to $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ because this packet carries seven ordered sequence rows plus an index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply tuple or bucket-member surfaces.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}$ packet is generated from seven tag-encoded BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistHeptaSequenceNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1375.1, Definition 1375.2, Definition 1375.3, Definition 1375.4, Definition 1375.5, Definition 1375.6, Definition 1375.7, Definition 1375.8, and Definition 1375.9 identify the ordered seven-row sequence-window carrier surface and its BEDC support package.

Formal verification: FormalStatus(NameCert$_{\text{BHistHeptaSequenceNameCert}^{\uparrow}}$, unformalized)
Bridge status: none
Origin: ai
Not claimed: This chapter does not provide a Lean declaration, standard bridge, tuple-chapter replacement, bucket-member replacement, or public acceptance interface.
Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistHeptaSequenceNameCertUp and checked carrier-admission, classifier-exactness, index-exhaustion, tag-exhaustion, provenance-readback, local-naming, and scoped-support theorems.

Chapter 1376

A Concrete Naming Certificate for $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$

The $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$ packet is the hard-floor capstone for the arity-seven tag-encoded BHist carrier-isomorphism bucket. It is composite over the neighboring tuple surfaces of chapter 1365, chapter 1363, and chapter 1364, and it also cites the anchor-stability certificate of chapter 1150, the audit-map family of chapter 807, and the beta critical-pair packet of chapter 881. The chapter bundles the seven-row carrier shape as one NameCert surface; it does not claim that the bucket is an atomic independent primitive or that any sibling chapter is replaced by this composite name.

Definition 1376.1 (BHistHeptaTupleNameCert carrier)**.** *A* $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, T, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_6$ *are the seven tag-encoded carrier rows. Each row is a* BHist *entry whose tag records the local carrier role used by a cited bucket sibling and whose payload is read only as a displayed BEDC history component.*
2. T *records componentwise* hsame *transport for* $R_0, \ldots, R_6, C, P, L$ *and no transport row outside the displayed packet.*
3. C *records the displayed* Cont *route that replays row admission, componentwise comparison, transport, provenance readback, and local naming for the same seven-row packet.*
4. P *is the* Pkg *provenance over the arity-seven tag encoder, the cited sibling carrier families,* BHist, hsame, Cont, *and* NameCert *rows.*
5. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistHeptaTupleNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, an eighth carrier component, a host-side equality test, a bucket-wide rewrite of sibling chapters, or a detached provenance table.

Lean4 (definition) `BEDC.Derived.BHistHeptaTupleNameCertUp.BHistHeptaTupleNameCertUp`

Definition 1376.2 (BHistHeptaTupleNameCert classifier)**.** *Two* $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$ *packets*

$$H = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, T, C, P, L)$$

and

$$H' = (R'_0, R'_1, R'_2, R'_3, R'_4, R'_5, R'_6, T', C', P', L')$$

are classifier-equivalent exactly when each displayed carrier row compares componentwise,

$$\begin{aligned} &R_0 \sim R'_0, \quad R_1 \sim R'_1, \quad R_2 \sim R'_2, \quad R_3 \sim R'_3, \\ &R_4 \sim R'_4, \quad R_5 \sim R'_5, \quad R_6 \sim R'_6, \end{aligned}$$

the transport rows display the same hsame *routes for those components and for C, P, L, and the replay rows display the same* Cont *route from row admission through local naming. No comparison is available through an omitted component, an untagged payload, or a member-chapter rewrite.*

Theorem 1376.3 (BHistHeptaTupleNameCert carrier admission)**.** *Every accepted* BHistHeptaTupleNameCert$^{\uparrow}$ *packet admits exactly the carrier rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, T, C, P, L.$$

The seven rows are tag-encoded BHist *components, and T, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the packet.*

Proof. Project the carrier of Definition 1376.1. The definition lists the seven tag-encoded BHist rows and then lists the transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted carrier read lands in exactly the displayed rows, and any proposed eighth component or detached structural table lies outside the accepted surface. □

Lean4 ✓ `BEDC.Derived.BHistHeptaTupleNameCertUp.BHistHeptaTupleNameCertCarrierAdmission`

Theorem 1376.4 (BHistHeptaTupleNameCert classifier exactness)**.** *For accepted* BHistHeptaTupleNameCert$^{\uparrow}$ *packets H and H', the classifier of Definition 1376.2 is exact for the arity-seven carrier surface: it compares the seven tagged* BHist *rows componentwise, compares the displayed* hsame *and* Cont *routes through T, T', C, C', and preserves the provenance and local naming rows through P, P', L, L'. It does not compare by bucket membership alone, by a host equality test, or by a hidden isomorphism between member chapters.*

Proof. Start with the classifier definition. It requires one comparison for each of $R_0, \ldots, R_6$, then requires the displayed transport and replay routes to match the same component list.

The provenance and local naming rows are part of the same carrier, so their comparison is not an external bucket judgment. Since the classifier has no clause for bucket membership alone, host equality, or hidden chapter isomorphism, the displayed componentwise route is the exact comparison surface. □

Theorem 1376.5 (BHistHeptaTupleNameCert scoped support package)**.** *The carrier admission and classifier exactness clauses bind the* BHistHeptaTupleNameCert$^{\uparrow}$ *packet to the displayed BEDC support*

BHist, hsame, Cont, Pkg, NameCert,

and to the cited sibling carrier families. The resulting scope is the arity-seven tag-encoded BHist *packet with displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a bucket-wide sibling rewrite, a host-side equality classifier, or a hidden eighth row.*

Proof. Apply Definition 1376.3 to identify the admitted rows and exclude any eighth coordinate. Apply Definition 1376.4 to keep comparison on the seven tagged BHist rows and the displayed structural rows.

The support package is therefore the named BEDC packet itself. Its provenance can source the cited bucket siblings, but no carrier or classifier row rewrites those sibling chapters or replaces their local certificates. □

Falsifiable prediction: Any accepted consumer route that reads an untagged row, an eighth carrier row, or bucket membership alone fails carrier admission or classifier exactness for $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$.

Independence witness: b-hist-octa-tuple-name-cert, b-hist-nona-tuple-name-cert, b-hist-deca-tuple-name-cert, anchor-stability-certificate, auditmapfamily, and beta-critical-pair: none is bijective to $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$ because this packet carries exactly seven tagged rows together with transport, replay, provenance, and local NameCert in one packet, while the neighboring capstones carry different arities and each cited sibling supplies its own member-family surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistHeptaTupleNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistHeptaTupleNameCert}^{\uparrow}$ packet is generated from seven tag-encoded BHist carrier rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BHistHeptaTupleNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 1376.1, Definition 1376.2, Definition 1376.3, Definition 1376.4, and Definition 1376.5 identify the arity-seven carrier-isomorphism surface and its BEDC support package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BHistHeptaTupleNameCert}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, member-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistHeptaTupleNameCertUp and checked carrier-admission plus classifier-exactness theorems.

Chapter 1377

A Concrete Naming Certificate for $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$

The $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$ packet is the ordered arity-eleven sequence-window capstone for the carrier-isomorphism bucket. It is composite over the deca sequence certificate of chapter 1368, the hendeca tuple certificate of chapter 1366, the dodeca tuple certificate of chapter 1369, the audit-map family of chapter 807, the anchor-stability certificate of chapter 1150, and the BEDC self-substrate packet of chapter 869. The chapter names the BEDC-native window that reads eleven ordered BHist rows together with componentwise transport, continuation replay, provenance, and local naming, without changing any member chapter.

Definition 1377.1 (BHistHendecaSequenceNameCert carrier)**.** *A* $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$ *carrier is a finite sequence-window packet*

$$Q = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, I, H, C, P, L)$$

with the following displayed rows:

1. $R_0, \ldots, R_{10}$ *are the eleven ordered* BHist *rows admitted by the window. Each row is read as a displayed BEDC history component, and its tag records the local sequence role rather than a host-side position.*
2. I *is the finite order and index row. It records the first and last admitted positions, the successor route through the eleven entries, and the rule that every sequence read factors through one of* $R_0, \ldots, R_{10}$.
3. H *records componentwise* hsame *transport for* $R_0, \ldots, R_{10}, I, C, P, L$ *and no transport row outside the displayed packet.*
4. C *records the displayed* Cont *replay from window admission through index preservation, componentwise comparison, transport, provenance readback, and local naming.*
5. P *is the* Pkg *provenance over the eleven-row window encoder, the tuple and sequence capstone anchors cited above, chapter 807, chapter 1150, chapter 869,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
6. L *is the local* $\mathsf{NameCert}_{\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}}$ *row.*

The carrier has no coordinate for an untagged row, a twelfth sequence component, a hidden permutation, a host equality test, unordered bucket membership alone, or a replacement of any tuple or sequence capstone chapter.

Theorem 1377.2 (BHistHendecaSequenceNameCert obligation surface)**.** *Every accepted* BHistHendecaSequenceNa *packet admits exactly the displayed window rows*

$$R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, I, H, C, P, L.$$

The eleven rows are ordered BHist *components, I is the only index and order row, and H, C, P, L are the only structural rows by which a consumer may transport, replay, source, or locally name the sequence window. In particular, the NameCert obligation surface is the carrier of Definition 1377.1 together with its index preservation,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Proof. Project the carrier of Definition 1377.1. The definition lists the eleven ordered BHist rows, then lists the index, transport, continuation, provenance, and local naming rows.

There is no remaining coordinate in the packet. Thus any accepted window read lands in exactly the displayed rows, and any proposed twelfth component, detached order table, hidden permutation, or off-packet structural row lies outside the accepted surface. □

Theorem 1377.3 (BHistHendecaSequenceNameCert field projection)**.** *Let*

$$Q = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, I, H, C, P, L)$$

be an accepted $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$ *packet. Any downstream carrier-isomorphism read of Q factors through one of the displayed fields $R_0, \ldots, R_{10}, I, H, C, P, L$. The projection may expose the ordered eleven-row window, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, or the local* NameCert *row, but it cannot expose unordered tuple membership, a host-side equality classifier, an omitted field, a hidden twelfth field, or a rewrite of chapter 1368, chapter 1366, or chapter 1369.*

Proof. Use the obligation surface of Definition 1377.2. It identifies the complete list of rows available to an accepted packet and states that I, H, C, P, L are the only structural rows attached to the eleven ordered BHist components.

A downstream projection therefore has only the displayed fields as sources. Since the carrier contains no unordered bucket-membership coordinate, no host equality classifier, no omitted-field lookup, no twelfth field, and no capstone rewrite row, none of those readings can be exported by projection. □

Falsifiable prediction: Any accepted consumer route that reads unordered tuple membership, an untagged row, a hidden permutation, a twelfth sequence row, a host equality classifier, an order table detached from I, or a sibling capstone rewrite fails the obligation surface or field projection for $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$.

Independence witness: b-hist-deca-sequence-name-cert, b-hist-hendeca-tuple-namecert, b-hist-dodeca-tuple-name-cert, auditmapfamily, anchor-stability-certificate, and bedc-self-substrate: none is bijective to $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$ because this packet carries eleven ordered payload rows plus an index row, componentwise transport, replay, provenance, and local NameCert, while the cited siblings supply different sequence, tuple, audit, anchor, or substrate surfaces.

Closure status for $\mathsf{NameCert}_{\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}$ packet is generated from eleven ordered BHist sequence rows, a finite index row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BHistHendecaSequenceNameCert}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1377.1, Definition 1377.2, and Definition 1377.3 identify the arity-eleven sequence-window carrier surface and its BEDC support package.

Formal verification: FormalStatus(NameCert$_{\text{BHistHendecaSequenceNameCert}\uparrow}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not provide a Lean declaration, standard bridge, tuple-chapter replacement, sequence-chapter replacement, or public acceptance interface.

Upgrade path: Formalization requires a Lean carrier for BEDC.Derived.BHistHendecaSequenceNameCertUp and checked obligation-surface plus field-projection theorems.

Chapter 1378

A Concrete Naming Certificate for $\mathsf{Order}^{\uparrow}$

The order interface $\mathsf{Order}^{\uparrow}$ is the unary comparison face read by the Full Nat Arithmetic boundary. It sits between the $\mathsf{Nat}^{\uparrow}$ spine and the $\mathsf{Add}^{\uparrow}$, $\mathsf{Mul}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ arithmetic consumers, exposing finite comparison packets over histories rather than a host natural-number order.

The unary comparison carrier consumes the $\mathsf{Nat}^{\uparrow}$ spine of chapter 28. Its public consumer boundary is read by chapter 29, chapter 30, chapter 31, and chapter 37, so the later arithmetic packets depend on $\mathsf{Order}^{\uparrow}$ only through finite comparison rows over the shared unary history substrate.

1378.1 Unary comparison packets

Definition 1378.1 (OrderUp unary comparison carrier)**.** *The carrier for* $\mathsf{Order}^{\uparrow}$ *consists of finite unary comparison packets*

$$\mathsf{OrdRow}(h, k, \beta, \lambda)$$

where h and k are $\mathsf{Nat}^{\uparrow}$ *unary histories, β is one displayed constructive comparison branch, and λ is the finite ledger that justifies that branch. The branch β is either a left-prefix witness from h to k, a right-prefix witness from k to h, or a classifier row $h\mathsf{hsame}k$. The ledger λ records only* BHist *constructors,* hsame *transport, and* Cont *prefix rows over the two unary histories.*

Lean4 ✓ `BEDC.Derived.OrderUp.TasteGate.OrderUpTasteGate single carrier alignment`

Definition 1378.2 (OrderUp unary comparison classifier)**.** *Two* $\mathsf{Order}^{\uparrow}$ *comparison packets* $\mathsf{OrdRow}(h, k, \beta, \lambda)$ *and* $\mathsf{OrdRow}(h', k', \beta', \lambda')$ *are classified together when $h\mathsf{hsame}h'$ and $k\mathsf{hsame}k'$, and the comparison branch transported along those two endpoint classifiers is the displayed branch β'. The ledger component is compared by replaying the same finite prefix and endpoint rows after this* hsame *transport.*

Theorem 1378.3 (OrderUp unary comparison NameCert obligations)**.** *The carrier and classifier of Definition 1378.1 and Definition 1378.2 supply the seed* $\mathsf{NameCert}_{\mathsf{Order}^{\uparrow}}$ *obligation surface over* BHist*. The carrier is inhabited by the reflexive packet on* e*, classifier reflexivity, symmetry, and transitivity are inherited from the endpoint* hsame *rows together with finite ledger replay, stability follows by transporting prefix and equality branches along* hsame*, and the ledger records exactly the displayed carrier, branch, transport, and prefix rows.*

Proof. For inhabitance, take the packet with $h = k = \mathsf{e}$, the equality branch $\mathsf{ehsamee}$, and the empty ledger.

The classifier laws are componentwise. Reflexivity uses the same two endpoints and the same comparison branch. Symmetry reverses the endpoint classifiers and replays the transported branch in the opposite packet direction. Transitivity composes the endpoint hsame rows and concatenates the finite ledger replays.

For stability, transport the two unary endpoints through the supplied hsame rows. A left-prefix branch remains a left-prefix branch after endpoint transport, a right-prefix branch remains right-prefix, and an equality branch remains an hsame branch. Since every ledger entry is one of these finite rows, the transported packet is again an $\mathsf{Order}^{\uparrow}$ carrier row. □

Lean4 ✓ `BEDC.Derived.OrderUp.OrderUnaryComparisonNameCertObligations`

Theorem 1378.4 (OrderUp Full Nat Arithmetic handoff)**.** *The unary comparison packet of Definition 1378.1, its classifier of Definition 1378.2, and the seed obligations of Definition 1378.3 form the* $\mathsf{Order}^{\uparrow}$ *face consumed by the Full Nat Arithmetic phase. Thus the* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Mul}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *consumers read order information only as finite unary comparison rows over the same* BHist *spine.*

Proof. Project the carrier first. Its only endpoints are $\mathsf{Nat}^{\uparrow}$ unary histories, and its comparison branch is a finite prefix-or-classifier row over those histories.

Project the classifier next. It transports comparison rows along hsame on the same two endpoints, so an arithmetic consumer receives a stable comparison interface rather than a second carrier or a host relation.

The seed obligations then provide the NameCert surface needed by the phase boundary: one inhabited carrier row, componentwise classifier laws, transport stability, and a ledger policy. Repacking these projections gives exactly the order face available to the Full Nat Arithmetic consumers listed in the statement. □

Lean4 ✓ `BEDC.Derived.OrderUp.OrderFullNatArithmeticHandoff`

Theorem 1378.5 (OrderUp AddMul consumer boundary)**.** *Every* $\mathsf{Add}^{\uparrow}$ *or* $\mathsf{Mul}^{\uparrow}$ *consumer of* $\mathsf{Order}^{\uparrow}$ *receives only finite unary comparison rows of Definition 1378.1 and classifier transport rows of Definition 1378.2. In particular, monotonicity or divisibility-order uses may read the displayed prefix branch and its finite ledger, but they do not receive host natural-number order, classical trichotomy, quotient order, or any ordered semiring or ordered ring structure from this chapter.*

Proof. An $\mathsf{Add}^{\uparrow}$ consumer reads the two endpoint histories through the additive continuation surface and may attach a comparison packet only after those endpoints are already present as unary histories. The packet contributes the branch and finite ledger, not a host order predicate.

A $\mathsf{Mul}^{\uparrow}$ consumer reads the repeated-addition ledger and may attach comparison packets to the displayed factors, partial products, or final product. Each such attachment is again an OrdRow over unary endpoints with componentwise classifier transport.

Since neither consumer projection introduces a semiring carrier, a ring carrier, a quotient relation, or a classical three-way decision principle, the only order data crossing this boundary are the finite prefix-or-equality rows explicitly stored in the $\mathsf{Order}^{\uparrow}$ packet. □

Theorem 1378.6 (OrderUp endpoint transport obligation)**.** *Any accepted* $\mathsf{Order}^{\uparrow}$ *packet*

$$\mathsf{OrdRow}(h, k, \beta, \lambda)$$

transported along endpoint rows $h\mathsf{hsame}h'$ *and* $k\mathsf{hsame}k'$ *remains an* $\mathsf{Order}^{\uparrow}$ *packet. Its transported branch is still one of the three branches of Definition 1378.1: a left-prefix witness, a right-prefix witness, or an equality classifier row. Its ledger still contains only* BHist *provenance rows, endpoint* hsame *rows, and* Cont *prefix rows, as required by Definition 1378.2 and Definition 1378.3.*

Proof. The carrier of Definition 1378.1 stores exactly two unary endpoints, one displayed comparison branch, and a finite ledger. The classifier of Definition 1378.2 allows transport only through the two endpoint hsame rows, so the endpoints become h' and k' and no new comparison carrier is introduced.

A left-prefix branch is replayed as a left-prefix branch over the transported endpoints, a right-prefix branch is replayed symmetrically, and an equality branch is replayed by the transported endpoint classifier. This is precisely the stability clause in Definition 1378.3.

The ledger replay uses only the original BHist rows, the two endpoint hsame rows, and the Cont rows witnessing the prefix branch. Hence the transported row is again an $\mathsf{Order}^{\uparrow}$ packet with the same finite ledger vocabulary. □

Lean4 ✓ `BEDC.Derived.OrderUp.OrderEndpointTransportObligation`

Theorem 1378.7 (OrderUp prefix branch antisymmetry)**.** *Let h and k be unary* $\mathsf{Nat}^{\uparrow}$ *histories. Suppose one accepted* $\mathsf{Order}^{\uparrow}$ *packet displays a left-prefix branch from h to k, and another accepted* $\mathsf{Order}^{\uparrow}$ *packet displays a right-prefix branch from k to h. Then the two endpoints are classified by* $h\mathsf{hsame}k$*. Equivalently, an arithmetic consumer of the public* $\mathsf{Order}^{\uparrow}$ *face cannot use opposite strict-prefix branches as two independent order facts; the paired finite ledgers collapse to the equality classifier row of Definition 1378.2.*

Proof. Read both packets through the carrier of Definition 1378.1. The only data available are the endpoints, the displayed branch, and the finite BHist ledger of Cont prefix rows and endpoint hsame rows. By Definition 1378.3, no host order or external trichotomy row can enter the comparison.

Induct on the paired unary prefix ledgers. At the empty-prefix boundary, the two displayed branches have no remaining strict Cont step on either side. The classifier of Definition 1378.2 therefore identifies the endpoints by the equality branch $h\mathsf{hsame}k$.

In the successor-prefix case, both opposite branches begin by exposing one shared Cont step through the unary history spine. Strip that step from the left-prefix ledger and from the right-prefix ledger, apply the induction hypothesis to the two tails, and obtain the tail endpoint classifier. Finally, transport this classifier back across the stripped endpoint rows using Definition 1378.6. The transported equality classifier is exactly $h\mathsf{hsame}k$. □

Theorem 1378.8 (OrderUp Int consumer boundary)**.** *Every* $\mathsf{Int}^{\uparrow}$ *order-facing read of* $\mathsf{Order}^{\uparrow}$ *factors through signed-pair endpoint comparisons inherited from the* $\mathsf{Nat}^{\uparrow}$ *spine and the* $\mathsf{Add}^{\uparrow}$ *continuation rows. In the signed-pair chapter chapter 31, an integer comparison may consume only the* OrdRow *packets attached to the displayed positive and negative unary components and their endpoint transport rows. It never imports host integer order, quotient order, ordered ring structure, or a trichotomy principle.*

Proof. The $\mathsf{Int}^{\uparrow}$ surface exposes signed pairs built from unary histories, so an order-facing read must first choose the displayed endpoints in those positive and negative components. By Definition 1378.1, the only comparison datum available on such endpoints is an OrdRow packet.

The endpoint movement is governed by Definition 1378.2 and Definition 1378.6. Thus the signed-pair consumer can replay comparison rows along endpoint hsame and Cont rows inherited from the $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Add}^{\uparrow}$ ledgers.

No clause in Definition 1378.3 exports a host integer relation or quotient order. Therefore the $\mathsf{Int}^{\uparrow}$ consumer boundary is exhausted by finite signed-pair endpoint comparisons. □

Theorem 1378.9 (OrderUp Rat consumer boundary)**.** *Every* $\mathsf{Rat}^{\uparrow}$ *order-facing read of* $\mathsf{Order}^{\uparrow}$ *factors through numerator and positive-denominator unary comparison packets, together with the denominator ledger rows displayed in chapter 37. A rational comparison may consume only* OrdRow *packets on the numerator and positive-denominator unary components, endpoint* hsame *transport, and denominator-positivity ledger rows. It never imports host rational order, field order, quotient order, ordered-field structure, or classical trichotomy.*

Proof. The $\mathsf{Rat}^{\uparrow}$ carrier presents numerator data and a positive-denominator record as finite unary rows. Hence an order-facing rational consumer must attach comparison information to those displayed unary components rather than to an ambient rational carrier.

By Definition 1378.1, each such comparison is an OrdRow with a prefix-or-equality branch. By Definition 1378.2 and Definition 1378.6, transport of that packet uses only endpoint hsame rows and finite Cont replay.

The NameCert obligations in Definition 1378.3 leave no ledger slot for a field order or a host rational trichotomy selector. The denominator contribution is therefore only the finite positivity ledger needed to route the unary denominator endpoint. □

Theorem 1378.10 (OrderUp obligation ledger exhaustion)**.** *The* $\mathsf{Order}^{\uparrow}$ *obligation ledger is exhausted by the two unary endpoints, the prefix-or-equality comparison branch, the endpoint classifier rows, the* Cont *replay rows, and the provenance rows naming the originating* $\mathsf{Nat}^{\uparrow}$ *histories. Every* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{Mul}^{\uparrow}$*,* $\mathsf{Int}^{\uparrow}$*, or* $\mathsf{Rat}^{\uparrow}$ *consumer read of* $\mathsf{Order}^{\uparrow}$ *factors through this finite inventory and no further order datum is present in the NameCert packet.*

Proof. Start with the carrier of Definition 1378.1. Its stored data are exactly two unary histories, one displayed branch, and the finite ledger justifying that branch.

The classifier of Definition 1378.2 adds only endpoint transport. Once the two endpoint hsame rows are fixed, the branch replay is forced by the corresponding Cont prefix rows or by the transported equality row.

The provenance rows record which $\mathsf{Nat}^{\uparrow}$ histories the endpoints came from. Therefore any consumer replay that claims to use $\mathsf{Order}^{\uparrow}$ can be checked against this list: endpoints, branch, endpoint classifier rows, continuation replay, and provenance. There is no remaining slot in the NameCert ledger for a host order, quotient order, trichotomy selector, or ordered-ring import. □

Lean4 ✓ `BEDC.Derived.OrderUp.OrderUnaryComparison obligation prefix ledger exhaustion`

Theorem 1378.11 (OrderUp public unary comparison export)**.** *The carrier of Definition 1378.1, the classifier of Definition 1378.2, the NameCert obligations of Definition 1378.3, the Full Nat Arithmetic handoff of Definition 1378.4, the* $\mathsf{Add}^{\uparrow}$ *and* $\mathsf{Mul}^{\uparrow}$ *boundary of Definition 1378.5, the endpoint transport obligation of Definition 1378.6, the prefix branch antisymmetry theorem of Definition 1378.7, the* $\mathsf{Int}^{\uparrow}$ *boundary of Definition 1378.8, the* $\mathsf{Rat}^{\uparrow}$ *boundary of Definition 1378.9, and the ledger exhaustion of Definition 1378.10 together form the public* $\mathsf{Order}^{\uparrow}$ *unary-comparison export. A public arithmetic consumer receives only finite* BHist *unary comparison rows with endpoint* hsame *transport,* Cont *prefix replay, and equality collapse for opposite prefix ledgers, not host order, quotient order, ordered algebraic structure, or classical trichotomy.*

Proof. The carrier and classifier first determine the public object being exported: an OrdRow stores two $\mathsf{Nat}^{\uparrow}$ unary histories, one prefix-or-equality branch, and a finite ledger, while classifier transport moves only the two endpoints and replays the displayed branch.

The NameCert obligations and Full Nat Arithmetic handoff then make this packet consumable. They provide the inhabited row, classifier laws, stability clause, and ledger policy needed by the arithmetic phase, without adding a second order carrier or a host relation.

The endpoint transport and prefix branch antisymmetry theorems control branch movement. Endpoint transport preserves the displayed branch under hsame, while antisymmetry collapses opposite finite prefix ledgers to the equality classifier row.

The consumer-boundary theorems identify the public readers. The $\mathsf{Add}^{\uparrow}$ and $\mathsf{Mul}^{\uparrow}$ boundary, the $\mathsf{Int}^{\uparrow}$ boundary, and the $\mathsf{Rat}^{\uparrow}$ boundary all route their order-facing uses through the same finite packet, endpoint transport obligation, and opposite-prefix collapse.

Finally, ledger exhaustion proves that the exported inventory is complete. Since every accepted consumer read is checked against the endpoints, branch, endpoint classifier rows, continuation replay, and provenance rows, the public surface contains only finite BHist unary comparison data and excludes host order, quotient order, ordered algebraic structure, and classical trichotomy. □

Theorem 1378.12 (OrderUp finite arithmetic mature consumer package)**.** *Every mature Full Nat Arithmetic consumer of* $\mathsf{Order}^{\uparrow}$ *factors through the finite unary comparison carrier of Definition 1378.1, the endpoint transport obligation of Definition 1378.6, the prefix branch antisymmetry theorem of Definition 1378.7, the* $\mathsf{Add}^{\uparrow}$ *and* $\mathsf{Mul}^{\uparrow}$ *consumer boundary of Definition 1378.5, the* $\mathsf{Int}^{\uparrow}$ *consumer boundary of Definition 1378.8, the* $\mathsf{Rat}^{\uparrow}$ *consumer boundary of Definition 1378.9, and the ledger exhaustion theorem of Definition 1378.10. The mature package exposes only finite* BHist *unary comparison rows, endpoint* hsame *transport,* Cont *prefix replay, equality collapse for opposite prefix ledgers, provenance rows, and the arithmetic consumer routes named above. It does not export host natural order, classical trichotomy, quotient order, ordered semiring structure, ordered ring structure, or field order.*

Proof. Apply the public export theorem Definition 1378.11 to identify the available $\mathsf{Order}^{\uparrow}$ surface. It supplies the unary comparison carrier, classifier obligations, Full Nat Arithmetic handoff, endpoint transport obligation, prefix branch antisymmetry, consumer boundaries, and exhausted finite ledger as one public packet.

The arithmetic consumers are then read through their boundary theorems. Apply Definition 1378.5 for the $\mathsf{Add}^{\uparrow}$ and $\mathsf{Mul}^{\uparrow}$ routes, Definition 1378.8 for signed-pair comparison reads, and Definition 1378.9 for numerator and positive-denominator comparison reads.

Finally, apply Definition 1378.10. Since every accepted consumer read is already accounted for by endpoints, branch data, endpoint classifier rows, continuation replay, and provenance, there is no remaining NameCert slot for host natural order, trichotomy, quotient order, ordered semiring or ring structure, or field order. □

Lean4 ✓ `BEDC.Derived.OrderUp.OrderUp StdBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{Order}^{\uparrow}}$.

Constructive story: An $\mathsf{Order}^{\uparrow}$ packet records finite unary comparison rows over the shared $\mathsf{Nat}^{\uparrow}$ spine. Its carrier stores two unary endpoint histories, a displayed prefix-or-classifier branch, and the finite ledger rows over BHist, hsame, and Cont that justify that branch; arithmetic consumers read only that finite comparison surface.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Order}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1378.1, Definition 1378.2, Definition 1378.3, Definition 1378.4, Definition 1378.5, Definition 1378.6, Definition 1378.7, Definition 1378.8, Definition 1378.9, Definition 1378.10, Definition 1378.11, and Definition 1378.12 bind the paper-side unary comparison, Full Nat Arithmetic handoff, arithmetic consumer transport rows, opposite-prefix equality collapse, Int/Rat comparison boundary, ledger exhaustion, public finite order-packet export, and mature consumer package.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Order}^{\uparrow}}$, bridgeChecked)

Lean target: `BEDC.Derived.OrderUp.OrderFullNatArithmeticHandoff`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not export host natural-number order, classical trichotomy, quotient order, ordered semiring structure, ordered ring structure, or a public bridge beyond the displayed finite comparison rows.

Upgrade path: The standard bridge is already machine checked for the finite unary comparison interface, so no further formal-axis upgrade remains for that bridge.

Chapter 1379

Heine-Borel Interval NameCert Construction

Following Bishop and Bridges, constructive compactness of a closed bounded real interval is recorded here as finite inspection data: every precision request is answered by an explicit dyadic net whose cells are routed through the existing real-name and stream-name surfaces. The packet consumes the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the finite-window interface of chapter 126, the regular Cauchy sequence route of chapter 660, and the dyadic-net certificate of chapter 1250. It packages closed interval endpoints, mesh scales, net centers, coverage rows, tail-stability rows, and Real-seal descent rows without importing an open-cover compactness theorem.

The dependency surface is the closed-interval lattice already present in the concrete-instance spine: the located interval packet of chapter 394, the finite epsilon-net packet of chapter 1767, the compact interval locator of chapter 1891, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The Heine-Borel interval certificate consumes these sibling rows as finite inspection data before exposing any interval-compactness consumer surface.

Definition 1379.1 (Heine-Borel interval carrier)**.** *A* $\mathsf{HeineBorelInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper closed* $\mathsf{Real}^{\uparrow}$ *endpoint names, including the finite seal rows by which they are read;*
2. *K is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet naming the closed bounded interval and its explicit dyadic epsilon-net surface;*
3. *M is the unary precision or mesh request for the dyadic net;*
4. *Z is the finite center ledger, with one displayed dyadic center for each accepted net cell at scale M;*
5. *F is the finite coverage ledger saying that every admitted interval read at scale M enters one of the cells named by Z through the route of K;*
6. *T is the tail-stability ledger saying that refining the precision request preserves the earlier accepted cells through displayed* hsame *comparisons;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the endpoint names, center names, and refinement witnesses;*

8. *R is the* RegularCauchySequence$^{\uparrow}$ *readback row for the same endpoint and center windows;*
9. *E is the terminal* Real$^{\uparrow}$ *seal-descent row reached only after K, M, Z, F, T, S, R have been displayed;*
10. *Q records componentwise* hsame *transport for endpoints, net cells, centers, coverage rows, tail-stability rows, windows, readbacks, and the Real seal;*
11. *C records displayed* Cont *routes from a closed interval read through mesh selection, finite net coverage, tail stability, StreamName windows, regular Cauchy readback, and RealUp sealing;*
12. *P is the* Pkg *provenance over* HeineBorelInterval$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, Real$^{\uparrow}$, StreamName$^{\uparrow}$, RegularCauchySequence$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* NameCert$_{\text{HeineBorelInterval}^{\uparrow}}$ *row.*

The carrier exports constructive compactness only as explicit dyadic epsilon-net witnesses for closed RealUp interval names. It has no coordinate for open covers, chosen finite subcovers, classical sequential compactness, quotient stream equality, host real equality, or an ambient completeness principle.

Lean4 (definition) `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalCarrier`
Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalTasteGate single carrier alignment`

Theorem 1379.2 (Heine-Borel interval net coverage)**.** *For every accepted* HeineBorelInterval$^{\uparrow}$ *carrier $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$, every closed-interval containment read at precision M factors through*

$$K, M, Z, F, Q, C, P, N.$$

The read first consumes the displayed ClosedBoundedIntervalNet$^{\uparrow}$ *packet K, then the mesh request M, then one of the finitely many centers in Z, and then the coverage ledger F attached to that center. No open cover, hidden subcover selector, host interval order, or unrecorded center choice is available to the consumer.*

Proof. Project the accepted carrier. Its interval-facing coordinate is the net packet K, and the mesh request M fixes the scale at which the consumer asks for coverage.

The center and coverage rows are finite. The route through Z and F identifies the displayed cell used by the containment read, so the consumer receives a named center and a named coverage witness rather than an open-cover object.

Finally the structural rows Q, C, P, N transport, replay, source, and name this same finite route. Since the carrier has no field for a chosen subcover, host interval order, or hidden center selector, the coverage read cannot escape the displayed dyadic net. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalNetCoverage`

Theorem 1379.3 (Heine-Borel interval tail stability)**.** *Let $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$ be an accepted* HeineBorelInterval$^{\uparrow}$ *carrier. If a later precision request refines M through the displayed tail-stability ledger T, then the earlier endpoint windows, accepted cells, center rows, coverage rows, and regular Cauchy readbacks transport along the stored* hsame *rows in Q and remain attached to the same package provenance P.*

Thus dyadic nets form a stable finite approximation surface for the closed interval, not a sequence chosen by countable choice or by classical sequential compactness.

Proof. Start with the refinement row in T. It is part of the same finite carrier as the endpoint rows, net packet, centers, and coverage ledger, so it can only compare displayed mesh requests and their displayed cells.

Transport is componentwise. The rows in Q move endpoint windows, centers, coverage witnesses, and regular Cauchy readbacks together, while C preserves the consumer route used to inspect them.

Repack the transported rows with the same provenance P and local name N. The result is still a finite dyadic-net surface because no coordinate supplies a countable-choice sequence, a host subsequence selector, or a classical compactness argument. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalTailStability`

Theorem 1379.4 (Heine-Borel interval Real seal descent)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *carrier reaches the endpoint and interval seal rows only through*

$$A, B, K, M, Z, F, T, S, R, E, Q, C, P, N.$$

The seal descent first displays the closed endpoint names A, B, then the explicit dyadic-net packet K, M, Z, F, then the tail-stability ledger T, then the $\mathsf{StreamName}^{\uparrow}$ *finite windows S and* $\mathsf{RegularCauchySequence}^{\uparrow}$ *readbacks R, and only then the Real seal row E. The route exports no quotient stream equality, host real equality, open-cover compactness, selected convergent subsequence, or ambient Real completeness row.*

Proof. Read the endpoint rows first. The carrier stores A and B as closed $\mathsf{Real}^{\uparrow}$ names, but the seal row E is not public until the finite net and readback rows have been displayed.

Next apply the net-coverage and tail-stability routes. The packet K, M, Z, F supplies the finite dyadic epsilon-net witness, and T keeps refinements attached to the same displayed cells and centers.

Finally pass through the finite observation rows S and R before reaching E. The structural rows Q, C, P, N preserve the route as one NameCert packet, so a Real-facing consumer cannot substitute quotient equality, host equality, open-cover compactness, a selected convergent subsequence, or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalRealSealDescent`

Theorem 1379.5 (Heine-Borel interval finite-net induction)**.** *Let $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$ be an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *carrier. Every accepted finite center ledger Z is consumed by primitive finite induction over the displayed net cells: the base row is the first accepted cell named by K, M, Z, F, and the step row extends an already consumed finite prefix by replaying* Cont *through F, Q, C, P, N for the next displayed cell.*

The induction surface exports only the finite ledger entries already named by Z and the route data already stored in F, Q, C, P, N. It exports no open-cover subcover, selected subsequence, countable-choice enumeration, host real equality, or quotient stream equality.

Proof. Project the finite center ledger Z from the accepted carrier. Since Z is a displayed BHist finite ledger, its cells are consumed by the primitive finite-list induction available to the packet, not by an external cover-selection principle.

The base row is the first accepted cell. Its center and coverage witness are read through K, M, Z, F, and the structural rows Q, C, P, N record the same route as a local $\mathsf{NameCert}_{\mathsf{HeineBorelInterval}^{\uparrow}}$ entry.

For the step row, assume the previous finite prefix has already been consumed. The next displayed cell is attached by replaying Cont through the coverage ledger F, transporting the relevant rows through Q, and repacking the route with C, P, N.

This exhausts the ledger because the induction ranges only over the displayed entries of Z. No row of the carrier can introduce an open-cover subcover, a selected subsequence, or a hidden infinite enumeration. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalNetInduction`

Theorem 1379.6 (Heine-Borel interval mesh-refinement functoriality)**.** *Let $H = (A, B, K, M, Z, F, T, S, R, E, Q, C$ be an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *carrier. A displayed refinement of the mesh request M transports coverage rows and center rows through T, Q, C and then composes with the finite-net induction route of Definition 1379.5. Thus refinement preserves the finite dyadic-net coverage surface as a named* $\mathsf{NameCert}_{\mathsf{HeineBorelInterval}^{\uparrow}}$ *route.*

Proof. Begin with the displayed refinement row in T. By Definition 1379.3, refinement preserves the earlier endpoint windows, accepted cells, center rows, coverage rows, and regular Cauchy readbacks through the stored hsame rows.

Transport the center and coverage rows componentwise through Q. The replay row C keeps the consumer path attached to the same package provenance P and local name N.

Now apply Definition 1379.5 to the transported finite ledger. The refined mesh therefore feeds the same finite induction route, with no appeal to open-cover compactness, a selected finite subcover, or an ambient Real completeness principle. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalMeshRefinementFunctoriality`

Theorem 1379.7 (Heine-Borel interval NameCert obligation surface)**.** *For an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *carrier $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelInterval}^{\uparrow}}$ *obligations are exhausted by carrier admission, finite-net induction, mesh-refinement functoriality, tail stability, and Real seal descent. Equivalently, every local consumer route factors through the carrier rows of Definition 1379.1, the coverage route of Definition 1379.2, the stability route of Definition 1379.3, the seal route of Definition 1379.4, the induction route of Definition 1379.5, and the refinement route of Definition 1379.6. The packet exports no open-cover compactness theorem, selected finite subcover, host real equality, quotient stream equality, selected convergent subsequence, or ambient Real completeness row.*

Proof. The carrier definition lists the only public rows of H: endpoints, closed bounded interval net data, mesh request, finite center ledger, coverage ledger, tail-stability ledger, observation windows, regular Cauchy readbacks, Real seal descent, transport, replay, provenance, and local NameCert.

Carrier admission is therefore the projection of these displayed rows. Coverage and tail stability are supplied by Definition 1379.2 and Definition 1379.3, while Real-facing consumers descend through Definition 1379.4.

The remaining finite-means obligations are the induction over Z and the refinement transport across M. These are exactly Definition 1379.5 and Definition 1379.6.

No other obligation can be exported from the packet, because the carrier has no row for open-cover compactness, selected finite subcovers, host real equality, quotient stream equality, selected convergent subsequences, or ambient Real completeness. Thus the listed carrier rows and theorem sites exhaust the local $\mathsf{NameCert}_{\mathsf{HeineBorelInterval}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalNameCertObligationSurface`

Theorem 1379.8 (Heine-Borel interval public finite-net export)**.** *Every accepted* HeineBorelInterval$^{\uparrow}$ *packet exports to public compactness consumers only the displayed finite dyadic net,* StreamName$^{\uparrow}$ *windows,* RegularCauchySequence$^{\uparrow}$ *readback,* ClosedBoundedIntervalNet$^{\uparrow}$ *coverage, tail-stability rows, mesh-refinement rows,* Real$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows named by Definition 1379.1.*

Consequently, a public consumer may reuse the finite-net coverage route, tail-stability route, mesh-refinement route, Real seal descent, and NameCert obligation surface, but it receives no additional interval object beyond the displayed rows of $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$.

Proof. Project the carrier rows of Definition 1379.1. The public data are exactly the endpoint names, closed bounded interval net packet, mesh request, center ledger, coverage ledger, tail-stability ledger, StreamName windows, regular Cauchy readbacks, Real seal, transport, replay, provenance, and local name.

Apply the existing finite routes. Net coverage is Definition 1379.2, finite induction is Definition 1379.5, refinement is Definition 1379.6, tail stability is Definition 1379.3, and Real-facing descent is Definition 1379.4.

The NameCert obligation surface of Definition 1379.7 repacks precisely those rows for public consumers. Since no further coordinate appears in the carrier, the public export cannot contain any object outside that finite dyadic-net surface. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalPublicFiniteNetExport`

Theorem 1379.9 (Heine-Borel interval closed-interval net extraction)**.** *Every accepted* HeineBorelInterval$^{\uparrow}$ *carrier exports an explicit closed-interval finite-net witness by first projecting the closed bounded interval net row* K*, mesh request* M*, center ledger* Z*, and coverage ledger* F*. The witness is then transported through* Q, C, P, N *as componentwise* hsame *transport, displayed replay, provenance, and local naming, while endpoint and readback access is sealed through the* StreamName$^{\uparrow}$*,* RegularCauchySequence$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows* S, R, E*.*

Thus the extraction surface of

$$H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$$

is a finite closed-interval net witness for compactness-facing consumers, not an open-cover compactness theorem, selected finite subcover, sequential compactness principle, quotient stream equality, or ambient Real$^{\uparrow}$ *completeness claim.*

Proof. Project the finite-net rows K, M, Z, F from Definition 1379.1. Net coverage is supplied by Definition 1379.2, so the interval read begins with the displayed closed bounded interval net rather than an arbitrary cover.

Use finite-net induction and mesh-refinement functoriality, Definition 1379.5 and Definition 1379.6. These keep the center ledger finite and preserve the same witness when later mesh requests are transported.

Finally apply the public finite-net export of Definition 1379.8. It routes the witness through Q, C, P, N and seals endpoint observation and Real-facing readback through S, R, E, leaving no carrier coordinate for open-cover compactness, selected subcovers, sequential compactness, quotient stream equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalClosedIntervalNetExtraction`

Theorem 1379.10 (Heine-Borel interval Real completion-facing route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *packet must pass through the finite center ledger* Z*, mesh-refinement functoriality for the request* M*, finite-net induction over the displayed cells, tail stability for refinements, Real seal descent through* S, R, E*, and the public finite-net export before any compact interval boundary is read.*

Thus the completion-facing endpoint route is the same finite dyadic-net route exposed by

$$H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N),$$

where A, B *are the closed endpoints,* K, M, Z, F *are the finite net and coverage rows,* T *is the tail-stability ledger,* S, R, E *are the StreamName, RegularCauchySequence, and Real seal rows, and* Q, C, P, N *preserve transport, replay, provenance, and local naming. The route is not an ambient completeness principle, quotient stream equality, selected limit, or open-cover compactness theorem.*

Proof. Start with the finite-net coverage route of Definition 1379.2. It forces any interval containment read to pass through K, M, Z, F, Q, C, P, N, so the completion-facing consumer receives a displayed finite center ledger rather than an external compactness object.

Next apply finite-net induction and mesh-refinement functoriality from Definition 1379.5 and Definition 1379.6. The induction makes every center-ledger read finite, and refinement keeps later mesh requests attached to the same coverage and center rows.

Tail stability and Real seal descent are then supplied by Definition 1379.3 and Definition 1379.4. Stability preserves endpoint windows, cell rows, center rows, coverage rows, and regular Cauchy readbacks through the stored transport ledger, while the Real descent requires A, B, S, R, E to be read after the finite-net rows.

Finally use the public finite-net export of Definition 1379.8. It repacks the finite ledger, refinement route, stability route, Real seal descent, transport, replay, provenance, and local name for downstream consumers, leaving no coordinate for ambient completeness, quotient equality, selected limits, or open-cover compactness. □

Lean4 (statement) `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalRealCompletionFacingRoute`

Theorem 1379.11 (Heine-Borel interval cover-refusal boundary)**.** *The public finite-net surface of Definition 1379.8 still refuses general open-cover compactness, selected finite subcovers, classical sequential compactness, host real equality, quotient stream equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every accepted read of* $H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N)$ *must pass through the displayed finite dyadic net and its StreamName, RegularCauchySequence, Real seal, transport, replay, provenance, and local NameCert rows before any compactness-facing conclusion is consumed.*

Proof. Inspect the public export theorem. It lists the finite dyadic net, observation windows, readbacks, coverage, tail-stability, refinement, Real seal, transport, replay, provenance, and naming rows as the complete public surface.

The excluded principles have no carrier coordinate. There is no row for an arbitrary open cover, no selector for finite subcovers, no sequential compactness witness, no host real equality test, no quotient stream equality, and no ambient completeness service.

Therefore the public surface remains a finite-net compactness interface only. Any consumer that attempts to read one of the excluded principles must leave the displayed carrier rows, so it is not reading the $\mathsf{HeineBorelInterval}^{\uparrow}$ packet described here. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalCoverRefusalBoundary`

Theorem 1379.12 (Heine-Borel interval cluster-filter route)**.** *Any* $\mathsf{ClusterFilter}^{\uparrow}$ *consumer that uses an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *packet must first read the interval finite-net coverage row, the tail-stability row, the Real-seal descent row, and the finite-net induction row before the cluster-filter adherent-observation row can be consumed. In carrier notation the Heine-Borel side supplies*

$$H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N),$$

and the cluster-filter side may read its adherent-observation row only after K, M, Z, F*,* T*,* S, R, E*, and the induction over* Z *have all been displayed through* Q, C, P, N*.*

The route exports a finite compactness-to-cluster handoff. It does not select a cluster point, form a quotient filter, import open-cover compactness, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite-net coverage theorem. It forces every interval containment read through K, M, Z, F, Q, C, P, N, so the cluster-filter consumer receives a displayed finite cell and coverage witness rather than a host compactness object.

Next apply tail stability and Real seal descent. Tail stability keeps refinements attached to the same endpoint windows, cell rows, centers, coverage rows, and readbacks, while Real seal descent requires the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegularCauchySequence}^{\uparrow}$ rows to precede the terminal $\mathsf{Real}^{\uparrow}$ seal.

Now use finite-net induction over Z. The induction supplies the finite row-by-row ledger that a cluster-filter consumer may inspect before it reaches its adherent-observation row. This composes with the $\mathsf{ClusterFilter}^{\uparrow}$ Cauchy-completion handoff, which already requires the CauchyFilter, MetricSpace, Topology, StreamName, RegSeqRat, Real seal, and adherent-observation rows to remain visible.

Since both sides expose only displayed finite rows, the handoff cannot choose a cluster point, quotient the filter, replace the dyadic-net route by open-cover compactness, or appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalClusterFilterRoute`

Theorem 1379.13 (Heine-Borel interval bridged finite-net interface)**.** *Every compact-interval or* $\mathsf{Real}^{\uparrow}$*-completion-facing consumer of an accepted* $\mathsf{HeineBorelInterval}^{\uparrow}$ *packet must route through the finite center ledger, mesh-refinement functoriality, net induction, Real seal descent, cover-refusal boundary, and cluster-filter route before reading the interval as a bridge-facing compactness interface.*

In carrier notation, the bridge-facing interface still exposes only

$$H = (A, B, K, M, Z, F, T, S, R, E, Q, C, P, N),$$

where Z *is the finite center ledger,* M *records mesh refinement,* F, T *record coverage and tail stability,* S, R, E *carry the StreamName, RegularCauchySequence, and Real seal rows, and* Q, C, P, N *preserve transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.HeineBorelIntervalUp.HeineBorelIntervalBridgedFiniteNetInterface`

Proof. Start with the public finite-net export of Definition 1379.8. It exposes only the dyadic interval net, finite center ledger, finite coverage and stability rows, readback rows, Real seal, transport, replay, provenance, and local name.

Next use mesh-refinement functoriality, net induction, and Real seal descent. The refinement theorem keeps later mesh reads attached to the same finite rows, the induction theorem makes every center-ledger read finite, and the Real seal descent theorem requires the StreamName and RegularCauchySequence rows before any Real-facing endpoint is consumed.

Finally apply the cover-refusal boundary and the cluster-filter route. The boundary removes open-cover compactness, selected subcovers, quotient stream equality, host real equality, selected convergent subsequences, and ambient Real completeness from the interface. The cluster-filter route shows that downstream compactness readers must still pass through the same finite rows before they can consume adherent-observation data. Hence the bridge-facing compactness interface is the displayed finite-net route and nothing outside it. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelInterval}\uparrow}$.

Constructive story: A $\mathsf{HeineBorelInterval}^{\uparrow}$ packet is generated from closed RealUp endpoint names, a ClosedBoundedIntervalNetUp dyadic-net packet, a mesh request, finite center and coverage ledgers, tail-stability rows, StreamName windows, RegularCauchySequence readbacks, a RealUp seal descent row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HeineBorelInterval}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1379.1, Definition 1379.2, Definition 1379.5, Definition 1379.6, Definition 1379.4, Definition 1379.7, Definition 1379.8, and Definition 1379.13 bind the bridged finite-net interval export for constructive compactness of closed bounded RealUp intervals, with tail stability supplied by Definition 1379.3.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HeineBorelInterval}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general open-cover compactness, classical sequential compactness, choice of finite subcovers, selected convergent subsequences, quotient stream equality, host real equality, or ambient Real completeness.

Upgrade path: Lean closure requires carrier admission, net coverage, finite-net induction, mesh-refinement functoriality, tail stability, Real seal descent, NameCert obligation-surface, and TasteGate field faithfulness for BEDC.Derived.HeineBorelIntervalUp.

Chapter 1380

A Concrete Naming Certificate for $\mathsf{LocatedApartness}^{\uparrow}$

The $\mathsf{LocatedApartness}^{\uparrow}$ packet is the Bishop–Bridges constructive-analysis handoff from located real data to positive apartness evidence. It sits between **??**, chapter 304, chapter 126, chapter 301, chapter 291, and chapter 38: the located endpoints supply finite windows and dyadic enclosures, and the apartness side receives only a positive finite gap row with visible BEDC provenance.

Definition 1380.1 (LocatedApartness carrier)**.** *A* $\mathsf{LocatedApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L_0, L_1, S, R_0, R_1, D, G, A, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are two accepted* $\mathsf{LocatedReal}^{\uparrow}$ *source rows;*
2. S *is a shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row on which both sources are read;*
3. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks of the two located sources over the shared window;*
4. D *is the pair of* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers projected from the two located packets;*
5. G *is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row witnessing that the two projected endpoint ledgers are disjoint on the displayed window;*
6. A *is the* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation ledger receiving the endpoint rows, the positive gap row, the shared window, and the two readbacks;*
7. H *records componentwise* hsame *transport for the two located sources, the shared window, the readbacks, the dyadic endpoint ledgers, the positive gap row, and the apartness ledger;*
8. C *records the displayed* Cont *routes from located windows to regular-rational readbacks, dyadic endpoint rows, and the apartness ledger;*
9. P *is the* Pkg *provenance row over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{LocatedApartness}^{\uparrow}}$ *row.*

The classifier compares two carriers by transporting all listed rows componentwise and then requiring that the transported positive gap row repacks to the transported $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation ledger. No coordinate stores trichotomy, a total order, a quotient representative, a selected limit, completeness, or host real equality.*

Theorem 1380.2 (LocatedApartness NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedApartness}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier visibility: the packet of Definition 1380.1 displays both* $\mathsf{LocatedReal}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers, the positive gap row, and the* $\mathsf{ApartnessReal}^{\uparrow}$ *ledger;*
2. *classifier transport: accepted packets compare only by componentwise* hsame *transport and displayed* Cont *replay of those same rows;*
3. *gap-row stability: the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row remains attached to the transported endpoint ledgers and the transported shared window;*
4. *ledger exactness: the apartness ledger is exactly the finite-window separation row determined by the two endpoint rows, the positive gap, the shared window, and the two readbacks;*
5. *no-trichotomy and no-quotient boundary: the packet exports no classical trichotomy, total order, quotient real representative, selected limit, completion datum, or host equality row.*

Lean4 ✓ `BEDC.Derived.LocatedApartnessUp.LocatedApartnessTasteGate single carrier alignment`

Proof. Project the carrier through Definition 1380.1. The first row exposes both located sources, and the next four rows expose the common finite window, the two regular-rational readbacks, the dyadic endpoint ledgers, and the positive gap row.

Classifier transport is componentwise. The hsame and Cont rows move the two located packets, the shared window, the readbacks, the endpoint ledgers, and the positive gap together, so the transported packet has the same visible shape.

Gap-row stability and ledger exactness are then the same check at the apartness boundary: the transported positive gap is attached to the transported endpoint ledgers and repacked as the $\mathsf{ApartnessReal}^{\uparrow}$ finite-window separation ledger.

The remaining boundary is negative. Since the carrier contains no coordinate for trichotomy, total order, quotient representatives, selected limits, completion, or host equality, no NameCert row can export any of those data. □

Theorem 1380.3 (LocatedApartness gap handoff)**.** *Let* L_0 *and* L_1 *be transported* $\mathsf{LocatedReal}^{\uparrow}$ *packets over a shared* $\mathsf{StreamName}^{\uparrow}$ *finite window. If their* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers display a positive dyadic gap row on that window, then the two endpoint rows, the positive gap, the shared window, the two readbacks, and the visible provenance repack as an* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation row.*

Consequently the located pair crosses the L10 handoff as finite positive-gap evidence, not as trichotomy of constructive reals, a total order, a quotient real comparison, a selected representative, or a completion principle.

Proof. Project L_0 and L_1 through the $\mathsf{LocatedReal}^{\uparrow}$ carrier surface. The projection exposes their regular-rational source rows, their finite schedules, their dyadic endpoint ledgers, and the provenance rows used to read those data.

Use the transported shared $\mathsf{StreamName}^{\uparrow}$ window to align the two projections. The two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks now live over the same finite observation surface, and the dyadic endpoint rows are the endpoint ledgers displayed by the two located packets.

Attach the positive $\mathsf{DyadicRatCore}^{\uparrow}$ gap row to these aligned endpoint ledgers. This forms the tuple named in the $\mathsf{ApartnessReal}^{\uparrow}$ finite-window separation row: two endpoint rows, one positive radius or gap row, one shared window, two readbacks, and the visible separation ledger.

Repack that tuple as the apartness ledger coordinate A of Definition 1380.1. The repacking uses only the displayed BHist, hsame, Cont, Pkg, and NameCert provenance, so no excluded classical or quotient datum enters the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedApartness}^{\uparrow}$ packet is generated from two located real source rows, a shared finite observation window, two regular-rational readbacks, dyadic endpoint ledgers, one positive dyadic gap row, an apartness separation ledger, componentwise transport, displayed continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1380.1, Definition 1380.2, and Definition 1380.3 name the seed carrier, obligation rows, and finite positive-gap handoff between located real packets and apartness evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy of constructive reals, total order, quotient real representatives, selected limits, completeness, or a standard bridge.

Upgrade path: To reach obligationClosure, the carrier, five obligation rows, and gap handoff must be checked as finite BEDC packet obligations.

Chapter 1381

A Concrete Naming Certificate for $\mathsf{BishopRealComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRealComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style regular-Cauchy real names, located-real readbacks, and Dedekind-cut boundary rows without importing quotient reals, trichotomy, host real equality, or ambient completeness.

The packet is a human-origin L10 comparison certificate between the regular-Cauchy presentation of chapter 343, the completion-facing handoff of chapter 360, the finite located readback of **??**, and the rounded cut boundary of chapter 436. It uses those sibling surfaces only as displayed certificate rows; it does not identify them as equivalent real constructions.

Definition 1381.1 (BishopRealComparison carrier)**.** *A* $\mathsf{BishopRealComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (B, C, L, D, R, S, H, P, N)$$

with the following visible rows:

1. *B is the accepted* $\mathsf{BishopReal}^{\uparrow}$ *row exposing a regular-Cauchy real name, its scheduled finite windows, dyadic tolerance rows, located-apartness evidence, and visible* $\mathsf{Real}^{\uparrow}$ *seal;*
2. *C is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *row reached from the same regular-Cauchy source surface, recording the completion-facing handoff without adding a quotient completion point;*
3. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *row that reads the same regular source through finite dyadic location windows;*
4. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *boundary row carrying lower and upper rational observations, roundedness, locatedness, disjointness, and rational embedding data;*
5. *R is the shared* $\mathsf{Real}^{\uparrow}$ *seal row through which all real-facing consumers must pass;*
6. *S is the readback ledger naming the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *schedules, regular-rational rows, and tolerance packets used by B, C, L, and D;*
7. *H records componentwise* hsame *and displayed* Cont *transports among the Bishop, completion, located, Dedekind-boundary, seal, and readback rows;*
8. *P is the* Pkg *provenance over* $\mathsf{BishopReal}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{DedekindCut}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealComparison}^{\uparrow}}$ *row.*
The carrier classifier compares two accepted packets only by replaying the displayed finite rows $B, C, L, D, R, S, H, P, N$. It has no coordinate for quotient-real equality, trichotomy, host real equality, a selected limit, a completeness theorem, or a proof that Bishop, located, and Dedekind presentations are globally equivalent.

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.TasteGate.BishopRealComparisonTasteGate single carrier alignment`

Theorem 1381.2 (BishopRealComparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *carrier $X = (B, C, L, D, R, S, H, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealComparison}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed packet, Bishop-real admission through B, completion handoff admission through C, located-readback admission through L, Dedekind-boundary admission through D, shared* $\mathsf{Real}^{\uparrow}$ *seal admission through R, finite StreamName/RegSeqRat/DyadicRatCore readback through S, componentwise* hsame *and* Cont *transport through H,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket namecert obligations`

Proof. Project the carrier of Definition 1381.1. The rows $B, C, L, D, R, S, H, P, N$ are the complete displayed coordinates of one finite BHist packet.

Read the four analytic-facing rows in route order. The Bishop row supplies the regular-Cauchy name surface, the completion row supplies the handoff reached from that surface, the located row supplies finite dyadic location readback, and the Dedekind row supplies lower and upper rational boundary observations.

The shared seal and readback rows then bind the comparison. The $\mathsf{Real}^{\uparrow}$ seal R is reached only after the regular windows, rational readback rows, and dyadic tolerance packets named by S have been exposed.

Finally, H, P, and N provide transport, provenance, and local naming for exactly those rows. Since no coordinate stores quotient equality, trichotomy, host equality, a selected limit, or ambient completeness, no further NameCert obligation can be read from the packet. □

Theorem 1381.3 (BishopRealComparison Cauchy-Dedekind boundary)**.** *Every completion-facing or cut-facing read exported by an accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet factors through the same finite regular-Cauchy source and tolerance surface. The completion row C is read through the Bishop row B, the shared readback ledger S, and the seal row R; the Dedekind boundary row D is read through the rational observation and dyadic tolerance rows already named by S.*

Thus the packet gives a Cauchy-to-Dedekind boundary comparison surface, not an equivalence theorem between quotient completions and Dedekind cuts.

Proof. First expose the regular-Cauchy source. The Bishop row B carries the scheduled finite windows, regular-rational name, dyadic tolerance data, and visible $\mathsf{Real}^{\uparrow}$ seal that the comparison packet is allowed to inspect.

Next follow the completion route. The row C is a completion-facing handoff attached to the same source surface, so a consumer cannot read C before it has read B and the StreamName/RegSeqRat/DyadicRatCore ledger S.

Now follow the cut-facing boundary. The row D supplies lower and upper rational observations, roundedness, locatedness, disjointness, and rational embedding data, but the public comparison reaches those observations only through the rational and dyadic rows named in S.

The conclusion is therefore a boundary statement. The carrier displays a finite route from Cauchy-name rows to cut-boundary observations, while the excluded quotient completion, cut equality, and ambient equivalence data remain absent from every coordinate. □

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket cauchy dedekind boundary`

Theorem 1381.4 (BishopRealComparison located readback)**.** *For an accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet* $X = (B, C, L, D, R, S, H, P, N)$*, every located-real consumer read factors first through the Bishop regular-Cauchy row* B*, then through the shared finite readback ledger* S*, and only then through the* $\mathsf{LocatedReal}^{\uparrow}$ *row* L*. The read exposes the same scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance packets, and* $\mathsf{Real}^{\uparrow}$ *seal row* R *that the completion and Dedekind-boundary routes use.*

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonCarrier located readback`

Proof. Project the located coordinate L from the carrier. It is not an independent located real object; it is stored with B, S, and R in the same finite packet.

Read the shared ledger S. It names the finite windows, regular-rational rows, and dyadic tolerance packets through which the Bishop, completion, located, and Dedekind-boundary routes are all replayed.

Transport the located read through H. The componentwise hsame and displayed Cont routes preserve only the scheduled rows already listed in S, so the consumer sees the same source and tolerance surface as the other comparison routes.

The located readback is therefore shared rather than detached. No extra order relation, quotient representative, host equality, or completion point is available to the consumer after L has been read. □

Theorem 1381.5 (BishopRealComparison non-escape)**.** *An accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet exports only the finite comparison surface*

$$B, C, L, D, R, S, H, P, N.$$

No downstream consumer can obtain quotient reals, trichotomy, host real equality, a selected limit, ambient completeness, or a global equivalence between Bishop Cauchy names, located reals, and Dedekind cuts from the packet alone.

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket nonescape`

Proof. Read the carrier inventory. The public surface consists exactly of the Bishop row, completion handoff, located readback, Dedekind boundary, shared real seal, shared finite readback ledger, transport row, provenance row, and local naming row.

Apply Definition 1381.2 to exhaust the local certificate obligations. Every admitted consumer read is one of the displayed rows or a replay of those rows through H, P, and N.

Apply Definition 1381.3 and Definition 1381.4 to the two analytic-facing exits. The completion, located, and Dedekind-boundary consumers all pass through the same finite source/readback surface before reaching their public rows.

The excluded data have no coordinate in the packet and no route through the displayed obligations. Hence a downstream consumer cannot escape to quotient equality, trichotomy, host real equality, selected limits, ambient completeness, or a global presentation-equivalence theorem. □

Theorem 1381.6 (BishopRealComparison source induction)**.** *Every accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet is obtained by finite induction over the displayed rows*

$$B, C, L, D, R, S, H, P, N.$$

The induction first admits the Bishop row, then the completion handoff, located row, Dedekind boundary row, shared seal, readback ledger, componentwise transport, provenance, and local naming rows. It refuses quotient equality, trichotomy, selected limits, and ambient completeness at every step.

Proof. The base step is the carrier projection of Definition 1381.1. It exposes a finite packet whose public coordinates are exactly $B, C, L, D, R, S, H, P, N$.

For the analytic rows, apply Definition 1381.2. The Bishop row B supplies the regular-Cauchy source, C is admitted only as the completion-facing handoff from that source, L is admitted only as the located readback, and D is admitted only as the Dedekind-boundary read.

For the shared rows, use Definition 1381.3 and Definition 1381.4. They force the completion, located, and cut-facing exits through the same seal R and readback ledger S.

The final induction steps add H, P, and N as transport, provenance, and local naming rows for the already displayed surface. By Definition 1381.5, no induction case has a coordinate from which quotient equality, trichotomy, selected limits, or ambient completeness can be introduced. □

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket source induction`

Theorem 1381.7 (BishopRealComparison located-cut transport)**.** *In an accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet, a located read and a Dedekind boundary read transported through the same* S, H, P, N *rows remain on the same finite regular-Cauchy and dyadic tolerance surface. The shared transport does not identify the located and cut presentations by quotient equality; it only preserves the displayed readback rows named by the carrier.*

Proof. Start with the located exit. By Definition 1381.4, any located-real consumer first reads the Bishop regular-Cauchy row B, then the shared ledger S, and then the located row L.

Now read the cut-facing exit. By Definition 1381.3, the Dedekind boundary row D is reached through the rational observation and dyadic tolerance rows already named by S.

Transport both reads through H and source them through P, N. The componentwise hsame and displayed Cont rows preserve the visible regular-Cauchy windows, rational readbacks, and dyadic tolerance packets; they do not create a host equality or quotient comparison between the located and cut presentations.

Thus the two exits stay on the same finite surface. The comparison is a shared readback and tolerance transport statement, not a presentation-equivalence theorem. □

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket located cut transport`

Theorem 1381.8 (BishopRealComparison seal exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopRealComparison}^{\uparrow}$ *packet must first consume the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and shared* $\mathsf{Real}^{\uparrow}$ *seal rows named by* S *and* R*. No real-facing read can bypass those rows to obtain quotient real equality, host equality, selected limits, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.BishopRealComparisonUp.BishopRealComparisonPacket seal exhaustion`

Proof. Project the shared readback ledger from Definition 1381.1. The row S names the finite StreamName windows, RegSeqRat rational rows, and DyadicRatCore tolerance packets used by the Bishop, completion, located, and Dedekind-boundary routes.

Apply Definition 1381.2 to the seal row. The shared $\mathsf{Real}^{\uparrow}$ seal R is an admitted row only after the finite readback ledger has been exposed.

Apply Definition 1381.5 to any attempted real-facing consumer exit. Since quotient real equality, host equality, selected limits, trichotomy, and ambient completeness have no coordinates in the packet, a consumer reaching R still receives only the displayed finite comparison surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealComparison}^{\uparrow}$ packet is generated from displayed $\mathsf{BishopReal}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, a shared finite readback ledger, componentwise hsame and Cont transports, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealComparison}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1381.1, Definition 1381.2, Definition 1381.3, Definition 1381.4, Definition 1381.5, Definition 1381.6, Definition 1381.7, and Definition 1381.8 export the finite Cauchy-Dedekind, located-readback, seal-exhaustion, and source-induction packet as the displayed bridge surface for the public Bishop, completion, located, Dedekind-boundary, and shared Real-seal comparison route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, bridge checking, quotient-real equality, trichotomy, host real equality, selected limits, ambient completeness, or a global equivalence theorem between Bishop Cauchy names, located reals, and Dedekind cuts.

Upgrade path: No further theory-axis upgrade remains; Lean verification and bridge checking remain independent obligations.

Chapter 1382

A Concrete Naming Certificate for $\mathsf{FilteredCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Bourbaki, *General Topology*, the FilteredCauchyCompletio packet records the finite BEDC route by which Cauchy filters reach the Real-completion interface. It is built over $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, RealCompletionExactBoundaryUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

The dependency anchors are the finite filter surface of chapter 333, the sequence-space handoff of chapter 406, and the Real boundary of chapter 1120. The uniform criterion supplies the common-tail test before any route reaches the Real seal, while stream names, regular rational rows, and dyadic ledgers keep the consumer boundary finite.

Definition 1382.1 (Filtered Cauchy completion carrier)**.** *A* $\mathsf{FilteredCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row carrying the finite filterbase and common-window evidence;*
2. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row that tests the same finite filter windows by uniform tails;*
3. *S is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff row exposing the associated* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readings;*
4. *Q is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* RealCompletionExactBoundaryUp *route reached from those same finite rows;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the displayed route;*
6. *H is the componentwise* hsame *transport row for F, U, S, Q, E;*
7. *C is the* Cont *continuation row that replays the accepted finite windows without adding host data;*
8. *P is the* Pkg *provenance over the listed completion, filter, uniform, stream, regular-rational, dyadic, transport, and seal rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FilteredCauchyCompletion}^{\uparrow}}$ *row.*

The accepted carrier exposes only these rows. It has no coordinate for a quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.FilteredCauchyCompletionUp.TasteGate.FilteredCauchyCompletionTasteGate single carrier alignment`

Theorem 1382.2 (Filtered completion carrier habitation)**.** *Whenever the rows $F, U, S, Q, E, H, C, P, N$ occur in one finite* BHist *with the dependencies listed in Definition 1403.1, they assemble an accepted* $\mathsf{FilteredCauchyCompletion}^{\uparrow}$ *carrier. Conversely, every accepted carrier projects to exactly those nine displayed rows.*

Proof. The forward direction reads the finite history in the carrier order. The $\mathsf{CauchyFilter}^{\uparrow}$ row supplies the finite filterbase, the uniform criterion row supplies the common-tail test, the sequence-space row supplies the stream, regular-rational, and dyadic readings, and the completion route supplies the Real-boundary handoff and terminal seal.

Transport, continuation, package provenance, and local naming are the final four displayed coordinates. Repacking these rows gives the named carrier.

For the reverse direction, project an accepted carrier by its definition. No extra coordinate is available, so the projection is exhausted by the nine listed rows. □

Theorem 1382.3 (Common-filter classifier transport)**.** *Two accepted* $\mathsf{FilteredCauchyCompletion}^{\uparrow}$ *carriers classify together only by a displayed common* $\mathsf{CauchyFilter}^{\uparrow}$ *refinement whose uniform-tail test, sequence space handoff, Real-completion route, terminal Real seal,* hsame *transport,* Cont *continuation,* Pkg *provenance, and local* NameCert *rows are transported componentwise.*

Proof. Project both carriers to their F rows and compare those rows by the common finite filter refinement supplied by $\mathsf{CauchyFilter}^{\uparrow}$. The classifier has no access to a host filter or quotient of streams.

Once the filter rows share a displayed refinement, transport the U, S, Q, E rows componentwise along H and replay the same reads through C. The package row records that these transported reads remain on the same finite dependency anchors.

Repacking the transported rows gives the classifier judgement for the two carriers. Reading such a judgement back yields exactly the common filter refinement and the listed componentwise transports. □

Theorem 1382.4 (Uniform-tail exactness)**.** *For an accepted carrier $K = (F, U, S, Q, E, H, C, P, N)$, every completion-facing read of Q first passes through the uniform-tail row U attached to the same* $\mathsf{CauchyFilter}^{\uparrow}$ *row F and the same* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff S. No Real seal consumer can bypass U by using a selected subsequence, quotient stream, or host completion point.*

Proof. Read the carrier coordinates. The route to Q is listed after F, U, S, so the completion-facing consumer sees the filterbase, uniform-tail test, and sequence-space handoff before it can read the Real-completion boundary.

The continuation row C only replays the accepted finite windows, and the transport row H only compares displayed coordinates. Neither row introduces a selected subsequence, quotient stream, or host completion point.

Thus the uniform-tail row is the exact gate for completion-facing reads. Any consumer that bypasses it has left the displayed carrier. □

Theorem 1382.5 (Completion handoff factorization)**.** *Every* RealCauchyCompletion$^{\uparrow}$ *or* RealCompletionExactBoun *handoff exported by an accepted* FilteredCauchyCompletion$^{\uparrow}$ *carrier factors through the rows* F, U, S, H, C, P, N *before reaching* Q *and the terminal seal* E*.*

Proof. Start from the carrier decomposition. The handoff row Q is not independent: it is stored as the route reached from the same finite filter, uniform-tail, sequence-space, transport, continuation, provenance, and naming rows.

The sequence-space row supplies the visible stream-name, regular-rational, and dyadic readings. The uniform-tail row checks those readings against the common filter classifier before the completion boundary is exposed.

Consequently every exported completion handoff factors through F, U, S, H, C, P, N and only then reaches Q and E. □

Theorem 1382.6 (Filtered completion no-escape)**.** *An accepted* FilteredCauchyCompletion$^{\uparrow}$ *carrier gives no route to quotient streams, selected limits, host real equality, countable choice, or ambient completeness. Every consumer of the packet is confined to the finite rows* $F, U, S, Q, E, H, C, P, N$*.*

Proof. The excluded objects are not coordinates of Definition 1403.1. A consumer can project the filter row, uniform criterion row, sequence-space handoff, completion route, Real seal, transport, continuation, package, and local naming row, and nothing else.

The previous factorization theorem shows that even completion-facing reads must pass through those displayed rows. The common-filter transport theorem shows the same boundary after classifier comparison.

Therefore any route requiring quotient streams, selected limits, host real equality, countable choice, or ambient completeness lies outside the accepted carrier. □

Closure status for NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$**.**

Constructive story: The filtered Cauchy completion packet is generated from a finite Cauchy filter row, a uniform-tail criterion row, a Cauchy sequence-space handoff, a Real-completion boundary route, a terminal Real seal, and the displayed hsame, Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: This chapter records the finite carrier and five seed obligations for routing Cauchy filters to the Real-completion boundary through displayed BEDC rows.

Formal verification: FormalStatus(NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: It does not claim quotient-stream equality, selected limits, host real equality, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Reaching obligationClosure requires Lean-backed carrier, classifier-transport, uniform-tail, handoff-factorization, and no-escape targets for the displayed packet.

Chapter 1383

A Concrete Naming Certificate for RegularCauchyEquivalence$^{\uparrow}$

The RegularCauchyEquivalence$^{\uparrow}$ packet records the finite equality boundary for regular Cauchy names before a Real$^{\uparrow}$ seal is consumed. Following Bishop–Bridges constructive analysis, two regular rational names are compared by common finite windows, dyadic tolerance ledgers, displayed transports, and local certificate rows, not by a quotient of streams or by host real equality.

The route is the Phase Real and Completion sibling boundary. It reads regular-name rows from chapter 301, finite schedules from chapter 126, dyadic tolerance data from chapter 291, zero-tail handoff data from chapter 380, the seal boundary from chapter 38, and the locatedness consumer interface from **??**. The chapter isolates the equivalence surface used between those rows so later completion and locatedness consumers can cite one visible NameCert packet.

Definition 1383.1 (RegularCauchyEquivalence finite-window classifier)**.** *A* RegularCauchyEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, Y, W, D, Z, R, E, H, C, P, N)$$

with the following rows:

1. *X and Y are the two displayed* RegSeqRat$^{\uparrow}$ *source rows to be compared;*
2. *W is the common* StreamName$^{\uparrow}$ *window schedule that selects paired finite observations from X and Y;*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for every selected pair of observations;*
4. *Z is the* NullSequence$^{\uparrow}$ *zero-tail handoff row when the displayed difference stream is consumed by a real-equality classifier;*
5. *R is the pair of* Real$^{\uparrow}$ *seal rows through which the two regular names may later be read;*
6. *E is the local finite equivalence classifier, asserting that each selected common window satisfies the displayed dyadic tolerance bound;*
7. *H records the componentwise* hsame *transports for the two source rows, schedule, tolerance ledger, seal rows, and classifier row;*
8. *C records the displayed* Cont *routes by which consumers inspect those rows;*
9. *P is the* Pkg *provenance for the two regular names, the common-window schedule, the dyadic ledger, the zero-tail handoff, the real-seal rows, and the consumer routes;*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}$ *row.*
The carrier stores only finite scheduled comparisons. It has no quotient stream equality, no selected limit, no countable-choice row, no trichotomy row, no host real equality, and no ambient real-completeness coordinate.

Theorem 1383.2 (RegularCauchyEquivalence refinement transport)**.** *Let* $(X, Y, W, D, Z, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet. If the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the common* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff row, and the* $\mathsf{Real}^{\uparrow}$ *seal rows are transported componentwise by the displayed* hsame *rows in* H *and read only through the displayed* Cont *routes in* C*, then the transported rows repack as an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet with the same finite-window classifier shape.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalence refinement transport`
Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.TasteGate.RegularCauchyEquivalence refinement transport`

Proof. Project the carrier of Definition 1404.1. Its whole comparison surface is the pair of regular Cauchy rows, one common schedule, one dyadic tolerance ledger, one zero-tail handoff row, the seal rows, and the finite classifier row.

Transport the two source rows, the schedule, and the tolerance ledger through the stated hsame rows. The transported schedule still selects paired finite observations, and every selected pair is checked against the transported dyadic ledger.

Transport the $\mathsf{NullSequence}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows only as stored rows beside the same finite comparison. Since the proof does not add a quotient equality, selected limit, countable-choice selector, trichotomy branch, host real equality, or ambient completeness datum, the transported inventory is exactly another instance of the same carrier definition. □

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalenceCarrier refinement transport`

Theorem 1383.3 (RegularCauchyEquivalence real classifier boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet factors first through the two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, then through the common* $\mathsf{StreamName}^{\uparrow}$ *windows, then through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then through the displayed* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff when equality consumption is requested, and only then reads the stored* $\mathsf{Real}^{\uparrow}$ *seal rows. The consumer receives no quotient stream equality, selected limiting real, host real equality, countable choice, trichotomy, or ambient completeness witness from the equivalence packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.TasteGate.RegularCauchyEquivalence real classifier boundary`

Proof. Start with the finite classifier of Definition 1404.1. The seal row R is stored after X, Y, W, D, Z, and E, so a seal-facing read must first pass through the two regular-name rows, the common finite-window comparison, and the available zero-tail handoff.

Use Definition 1404.2 for stability under displayed transport. It keeps the consumer route attached to the same scheduled windows and dyadic ledger while moving along hsame and Cont rows.

The carrier lists no quotient stream equality, selected limit, host real equality, choice row, trichotomy row, or completeness row. Thus the boundary between regular Cauchy names and real seals is exactly the displayed finite-window route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalence real classifier boundary`

Theorem 1383.4 (RegularCauchyEquivalence NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyEquival}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation by two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, a common* $\mathsf{StreamName}^{\uparrow}$ *window schedule, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff row,* $\mathsf{Real}^{\uparrow}$ *seal rows, and displayed provenance;*
2. *classifier by finite common-window dyadic tolerance checks;*
3. *transport stability under the stored* hsame *rows and displayed* Cont *routes;*
4. *seal-boundary exactness, stating that every* $\mathsf{Real}^{\uparrow}$ *read is reached only after the common-window classifier row and the displayed zero-tail handoff when equality consumption is requested;*
5. *ledger refusal, stating that quotient stream equality, host real equality, countable choice, selected limits, trichotomy, and ambient real completeness are not packet coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalenceTasteGate single carrier alignment`

Proof. Carrier habitation is the displayed tuple in Definition 1404.1. It supplies two regular Cauchy names, a common schedule, dyadic tolerance rows, a zero-tail handoff row, seal rows, transports, continuations, package provenance, and the local naming row.

The classifier row is finite because it reads only windows selected by W and tolerances recorded in D. No comparison is made outside the scheduled observations.

Transport stability is Definition 1404.2. Seal-boundary exactness is Definition 1404.3. Together they keep every consumer route on the visible BHist, hsame, Cont, Pkg, and NameCert rows.

The refusal row is immediate from the carrier inventory: there is no coordinate that can encode a quotient stream equality, selected limit, countable-choice selector, trichotomy decision, host real equality, or ambient completeness principle. □

Theorem 1383.5 (RegularCauchyEquivalence sibling route)**.** *Any completion-facing or locatedness-facing consumer that uses a* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet must read the finite classifier of Definition 1404.1. Its route is exhausted by* $X, Y, W, D, Z, R, E, H, C, P, N$*: the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, common* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff,* $\mathsf{Real}^{\uparrow}$ *seal rows, classifier row, transports, continuations, package provenance, and local* NameCert *row. No quotient of Cauchy streams, host real equality, selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness data is available to that consumer.*

Proof. Project an accepted packet to the tuple of Definition 1404.1. This tuple is already the full finite inventory available to completion-facing and locatedness-facing consumers.

Apply Definition 1404.4. The NameCert rows bind carrier habitation, finite common-window classification, transport stability, seal-boundary exactness, and the ledger refusal inside the same packet.

The route to $\mathsf{LocatedReal}^{\uparrow}$ may read the same regular sources and seal rows used by **??**, but it does not gain an independent real equality or completion principle. Therefore every downstream read remains a nonquotient finite-window comparison over $X, Y, W, D, Z, R, E, H, C, P, N$. □

Theorem 1383.6 (RegularCauchyEquivalence scoped primitive route)**.** *Any completion-facing or locatedness-facing consumer of an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet must read exactly the tuple*

$$(X, Y, W, D, Z, R, E, H, C, P, N)$$

through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows named in Definition 1404.1. This scoped primitive route is the whole access path to the finite equivalence classifier; no quotient of streams, host real equality, selected limit, choice row, trichotomy row, or ambient completeness row is available.*

Proof. First apply refinement transport, Definition 1404.2. It keeps the two regular names, common window, dyadic ledger, null-tail handoff, real seals, classifier, transport, replay, provenance, and local name inside the same finite packet.

Next use the real classifier boundary of Definition 1404.3. It forces every seal-facing read through the regular-name, window, tolerance, and zero-tail rows before the $\mathsf{Real}^{\uparrow}$ seal rows can be inspected.

Finally combine the NameCert obligation surface and sibling route, Definition 1404.4 and Definition 1404.5. These theorems exhaust the consumer path by the displayed tuple, so the scoped primitive route contains exactly the listed BEDC rows and no hidden quotient, host equality, selected limit, choice, trichotomy, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a common $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{NullSequence}^{\uparrow}$ zero-tail handoff row, $\mathsf{Real}^{\uparrow}$ seal rows, a finite equivalence classifier, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1404.1, Definition 1404.2, Definition 1404.3, Definition 1404.4, Definition 1404.5, and Definition 1404.6 display the obligation surface over RegSeqRat, StreamName, DyadicRatCore, NullSequence, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, a public completion quotient, host real equality, selected limits, countable choice, trichotomy, or ambient real completeness.

Upgrade path: Remaining work is Lean verification and bridge checking for the same finite regular-Cauchy equivalence route, not further paper-axis maturation.

Chapter 1384

A Concrete Naming Certificate for $\mathsf{MonotoneSubsequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{MonotoneSubsequence}^{\uparrow}$ packet records monotone subsequences as displayed finite unary selector rows. It does not select an infinite subsequence by countable choice. The packet sits between the regular-Cauchy subsequence surface of chapter 381, the cofinal index witness of chapter 405, and the monotone-Cauchy route of chapter 438. Its endpoint side is read through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 1384.1 (Monotone subsequence carrier)**.** *A* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row whose inspected finite windows are being selected.*
2. *M is the finite monotone unary selector ledger; each displayed entry records a selector value, its source index, and the comparison row certifying monotonicity against the previously displayed selector entries.*
3. *Q is the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *evidence row for the inspected thresholds, limited to the thresholds actually present in the packet.*
4. *W is the finite family of selected* $\mathsf{StreamName}^{\uparrow}$ *windows read from S through the ledger M and the cofinal evidence Q.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger extracted from the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from W and D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the selected regular-rational handoff has been displayed.*
8. *H stores componentwise* hsame *transport for the source row, monotone selector ledger, cofinality evidence, selected windows, dyadic observations, regular handoff, Real seal, continuation, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from S, M, Q to W, from W to D, from D to R, and from R to E.*

10. *P is the* Pkg *provenance over* $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host subsequence equality test, a countable-choice selector, a quotient stream, an ambient completeness principle, or a hidden infinite monotone-subsequence object.

Lean4 ✓ `BEDC.Derived.MonotoneSubsequenceUp.TasteGate.MonotoneSubsequenceTasteGate single carrier alignment`

Theorem 1384.2 (Monotone subsequence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *carrier* $U = (S, M, Q, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *monotone selector stability for the finite unary selector ledger M under the recorded* hsame *transports;*
3. *cofinal coverage for every inspected threshold listed in Q;*
4. *selected-window* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through W, D, R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is consumed;*
5. *non-escape from host subsequence equality, countable choice, quotient-stream equality, and ambient completeness.*

Proof. Project the carrier of Definition 1405.1. The source row S, selector ledger M, cofinality evidence Q, selected windows W, dyadic observations D, handoff R, seal E, and structural rows H, C, P, N are the only coordinates available to the classifier.

The monotone and cofinal faces are finite. Selector stability reads the displayed comparisons inside M, while cofinal coverage reads only the inspected thresholds recorded in Q. No step quantifies over an unbounded family of undisplayed requests.

The analytic endpoint is downstream from the same finite route. The selected $\mathsf{StreamName}^{\uparrow}$ windows W produce the dyadic ledger D, the dyadic ledger produces the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, and the Real seal E is available only after that handoff.

Finally inspect the excluded objects. Host subsequence equality, countable choice, quotient streams, and ambient completeness are not rows of U, and the structural rows only transport, replay, source, and name the displayed packet. The five listed obligations therefore exhaust the local NameCert surface. □

Theorem 1384.3 (Monotone subsequence regular-Cauchy handoff)**.** *Every accepted* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *packet hands its selected monotone finite windows to* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Equivalently, every seal-facing route factors through*

$$S, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Begin with the selected-window rows. The continuation route records that W is read from the source regular-Cauchy subsequence row S, the monotone selector ledger M, and the cofinal threshold evidence Q.

Next read the regular-rational handoff. The dyadic observation ledger D is extracted from the selected StreamName windows, and the row R is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ handoff reached from those dyadic observations.

Only then is the seal row available. The row E is part of the packet as the Real-facing endpoint reached through R, while H, C, P, N only preserve the same route by transport, replay, provenance, and local naming.

Thus a consumer that tries to use the $\mathsf{Real}^{\uparrow}$ seal without the selected-window $\mathsf{RegSeqRat}^{\uparrow}$ handoff leaves the accepted $\mathsf{MonotoneSubsequence}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneSubsequence}^{\uparrow}$ packet is generated from a source regular-Cauchy subsequence row, a finite monotone unary selector ledger, inspected-threshold cofinality evidence, selected StreamName windows, DyadicRatCore observations, a RegSeqRat handoff, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1405.1, Definition 1405.2, and Definition 1405.3 declare the finite selector carrier, its NameCert obligations, and the regular-Cauchy handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Bolzano-Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host subsequence equality, Lean verification, bridge checking, or RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, monotone selector stability, inspected-threshold cofinal coverage, selected-window RegSeqRat handoff, and seal-route non-escape for the same packet.

Chapter 1385

A Concrete Naming Certificate for $\mathsf{SpeckerSequence}^{\uparrow}$

Following Specker's constructive counterexample tradition and Bishop-style finite evidence, the $\mathsf{SpeckerSequence}^{\uparrow}$ packet records a monotone bounded computable rational stream whose supremum is not exposed as a located $\mathsf{Real}^{\uparrow}$ datum without an additional location witness. The carrier sits in the Phase Real and Completion frontier between the regular rational names of chapter 301, the stream schedules of chapter 126, the dyadic tolerance ledgers of chapter 291, the finite real seal boundary of chapter 38, and the selector discipline of chapter 1405.

The Cauchy-criterion sibling boundary of chapter 2255 supplies the tail-window reading discipline used to inspect bounded monotone finite windows without exposing a located supremum.

Definition 1385.1 (SpeckerSequence carrier)**.** *A* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, D, M, B, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the computable rational stream by regular finite windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that exposes the inspected windows of R.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying the rational bounds and tolerances read from those windows.*
4. *M is the finite monotone ledger, recording that each displayed inspected value is below the later displayed values in the schedule.*
5. *B is the boundedness ledger, recording a displayed lower rational row and upper rational row for every inspected window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only as a finite readback boundary for displayed regular windows, not as a located supremum.*
7. *H stores componentwise* hsame *transport for the source, schedule, dyadic, monotone, boundedness, seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from R, W into D, from D into M and B, and from the regular readback surface into E.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for located supremum data, Dedekind cuts, countable choice, trichotomy, ambient completeness, or a selected limiting real.

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceUp`

Theorem 1385.2 (SpeckerSequence carrier exposure)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet exposes exactly the rows $R, W, D, M, B, E, H, C, P, N$ of Definition 1406.1. A consumer may inspect the computable rational stream only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bounds, the monotone and boundedness ledgers, the finite* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed structural rows.*

Proof. Project the carrier tuple. The source row, schedule row, dyadic ledger, monotone ledger, boundedness ledger, seal row, transport row, continuation row, provenance row, and local naming row are the complete public inventory.

The route is finite at every stage. The stream is read through R and W, rational bounds are read through D, monotonicity and boundedness are read through M and B, and the seal row E is reached only after those rows have been displayed.

Since the carrier lists no located supremum, Dedekind cut, selected limit, countable-choice selector, trichotomy branch, or completeness row, those objects are not part of the exposed packet. □

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceCarrier exposure`

Theorem 1385.3 (SpeckerSequence classifier transport)**.** *Let $S = (R, W, D, M, B, E, H, C, P, N)$ be an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier. Componentwise* hsame *transport along H and displayed* Cont *replay along C preserve the classifier shape of S: the transported packet is still compared by the same regular-source row, stream schedule, dyadic ledger, monotone ledger, boundedness ledger, finite seal row, provenance row, and local naming row.*

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceClassifier transport`

Proof. Read the classifier from Definition 1406.1. It depends only on the displayed rows of the packet, not on an ambient order or a host real equality test.

Transport the source, schedule, dyadic, monotone, boundedness, and seal rows componentwise through H. The continuation row C records the same replay order after transport: regular windows first, dyadic observations next, monotone and boundedness ledgers next, and the finite seal last.

The transported inventory therefore repacks as the same carrier shape. No transported coordinate can introduce located-supremum data, countable choice, trichotomy, Dedekind-cut equality, or completeness, because no such coordinate is present before transport. □

Theorem 1385.4 (SpeckerSequence monotone bounded ledger)**.** *In every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet, the monotone ledger M and boundedness ledger B are read before any* $\mathsf{Real}^{\uparrow}$ *seal consumer can inspect E. Thus every seal-facing route factors through*

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequence monotone bounded ledger`

Proof. Begin with the source route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R and $\mathsf{StreamName}^{\uparrow}$ schedule W select the finite rational observations inspected by the packet.

Next read the dyadic ledger. The row D supplies the rational bounds and tolerance data used to compare the selected observations.

Now read the monotone and boundedness ledgers. The row M records the displayed monotone comparisons, while B records the displayed lower and upper rational bounds for the same inspected windows.

Only after those rows have been exposed can the finite $\mathsf{Real}^{\uparrow}$ seal row E be consumed. The structural rows H, C, P, N preserve this route and do not add a selected supremum or an ambient completeness witness. □

Theorem 1385.5 (SpeckerSequence finite-stage induction)**.** *For every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet* $S = (R, W, D, M, B, E, H, C, P, N)$*, induction along the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *lets any consumer read only the current finite rational window of* R*, the corresponding dyadic tolerance row of* D*, the monotone ledger prefix of* M*, the boundedness ledger prefix of* B*, and the transported* BHist *provenance before the consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The induction route introduces no located supremum, selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Induct on the finite schedule prefix displayed by W. In the base case, the consumer sees only the initial inspected window of R, the matching tolerance and bound row in D, the empty or initial monotone and boundedness ledger entries in M and B, and the provenance rows carried by H, C, P, N.

For the step case, append one visible rational observation and its tolerance row. Componentwise hsame in H transports the source, schedule, dyadic, monotone, boundedness, and seal coordinates without changing their row types. The displayed Cont replay in C appends the new dyadic observation before replaying the corresponding monotone comparison in M and the corresponding lower and upper bound in B.

Thus each prefix consumer receives exactly the finite window and ledger data already displayed at that stage. Passing from one prefix to the next preserves the route of Definition 1406.4; the $\mathsf{Real}^{\uparrow}$ seal remains downstream of the visible rational, tolerance, monotone, boundedness, and provenance rows. Since no induction clause contains a coordinate for located supremum data, selected limits, countable choice, trichotomy, or ambient completeness, none is introduced before the seal. □

Theorem 1385.6 (SpeckerSequence finite-prefix Real-tail inspection)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *consumer that inspects the packet before the* $\mathsf{Real}^{\uparrow}$ *seal is read through finite windows only. More precisely, before any completion-facing refusal or located-supremum boundary is invoked, the consumer can access only the displayed* $\mathsf{StreamName}^{\uparrow}$ *prefix of* W*, the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound rows in* D*, the monotone and bounded ledger prefixes in* M *and* B*, and the structural rows* H, C, P, N*.*

The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *may be cited only after those finite-prefix rows have been displayed, and the read still contains no located supremum, selected limit, quotient stream equality, countable-choice selector, trichotomy branch, or ambient completeness witness.*

Proof. Project the carrier of Definition 1406.1. Its analytic coordinates before the seal are exactly the regular-source row R, the finite schedule W, the dyadic ledger D, and the monotone and boundedness ledgers M, B.

Apply the finite-stage induction theorem, Definition 1406.5. At the base prefix the consumer sees only the initial displayed rational window and its ledger rows; at each step it appends one finite observation, its tolerance row, and the matching monotone and boundedness entries.

Use the monotone bounded ledger route, Definition 1406.4, to place the $\mathsf{Real}^{\uparrow}$ seal after these finite-prefix rows. Then apply the RealUp seal non-escape theorem, Definition 1406.10, to keep completion-facing refusal and located-supremum boundaries outside the displayed packet. □

Theorem 1385.7 (SpeckerSequence non-located supremum boundary)**.** *An accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet may certify a monotone bounded computable rational stream, but it does not certify a located supremum. Any downstream route that treats the packet as a located* $\mathsf{Real}^{\uparrow}$ *datum must supply an external location witness not contained in* $R, W, D, M, B, E, H, C, P, N$*.*

Proof. Use Definition 1406.2 to identify the packet inventory. The available analytic rows are the regular rational source, stream schedule, dyadic observations, monotone ledger, boundedness ledger, and finite seal boundary.

Use Definition 1406.4 for the seal route. It records that E is reached through finite windows and ledgers, not through a located-supremum construction.

A located supremum requires a location witness comparing rational cuts around the supremum. No such witness is a row of the carrier, and the structural rows merely transport, replay, source, and name the displayed packet. Hence the packet sets the non-located boundary rather than closing located completeness. □

Theorem 1385.8 (SpeckerSequence located supremum refusal)**.** *For any accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet* $S = (R, W, D, M, B, E, H, C, P, N)$*, finite-stage induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule can expose only the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the dyadic tolerance row* D*, the monotone ledger* M*, the boundedness ledger* B*, the finite* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural rows* H, C, P, N*. Therefore a located-supremum witness cannot be reconstructed from the packet and must be supplied as an external row.*

Proof. Apply Definition 1406.5. The induction route over W reads a finite prefix of the source stream, the matching dyadic tolerance data, the corresponding monotone and boundedness ledger prefixes, and the transported provenance rows before the seal is reached.

Apply Definition 1406.4. It fixes the seal-facing order

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E$$

and leaves H, C, P, N as structural transport, replay, provenance, and naming rows rather than analytic witnesses.

Finally apply Definition 1406.10. The finite $\mathsf{Real}^{\uparrow}$ seal exports only the displayed finite-window read. Since none of the displayed rows is a located comparison witness for rational cuts around the supremum, any downstream located-supremum claim must provide that witness outside the $\mathsf{SpeckerSequence}^{\uparrow}$ packet. □

Theorem 1385.9 (Specker sequence located supremum consumer boundary)**.** *A* $\mathsf{LocatedSupremum}^{\uparrow}$ *or* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer reading an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet receives only finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source windows, the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, monotone and boundedness ledgers, the* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows* H, C, P, N*. The located-supremum witness remains an external row.*

Proof. Apply Definition 1406.5. It confines every inspected prefix to the displayed regular rational window, stream schedule, dyadic tolerance row, monotone prefix, boundedness prefix, and transported provenance before the seal is reached.

Apply Definition 1406.4. The monotone and boundedness rows are read before any $\mathsf{Real}^{\uparrow}$ seal consumer can inspect E, so the route remains finite-window data rather than a located supremum construction.

Apply Definition 1406.8. Since the packet contains no located comparison witness for rational cuts around the supremum, a $\mathsf{LocatedSupremum}^{\uparrow}$ or $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ consumer must bring that witness as an external row. □

Theorem 1385.10 (SpeckerSequence RealUp seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet is a finite seal read through the displayed source, schedule, dyadic, monotone, and boundedness rows. The read exports no Real completeness principle, located supremum, Dedekind cut, countable-choice selector, trichotomy branch, bridge checking datum, or selected limiting real.*

Proof. Factor the read through the carrier exposure theorem. The $\mathsf{Real}^{\uparrow}$ seal row E is available only as one displayed coordinate of the finite packet.

Apply classifier transport to keep every replay on the same componentwise rows. Transport cannot change the seal into a located supremum or a completed real object because it preserves the carrier shape.

Apply the monotone bounded ledger theorem and the non-located supremum boundary. Together they show that a seal-facing consumer receives only finite regular windows, dyadic observations, monotone evidence, boundedness evidence, and structural certificate rows.

The excluded principles are absent from every coordinate. Therefore the $\mathsf{Real}^{\uparrow}$ seal cannot escape to Real completeness, located supremum data, Dedekind cuts, countable choice, trichotomy, bridge checking, or selected limits. □

Definition 1385.11 (Specker sequence obligation surface)**.** *The* $\mathsf{SpeckerSequence}^{\uparrow}$ *obligation surface is the finite subpacket*

$$O = (R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, E, H, C, P, N)$$

read from an accepted carrier $S = (R, W, D, M, B, E, H, C, P, N)$*. The row* $R_{\leq W}$ *is the finite rational stream window selected by the* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*;* $D_{\leq W}$ *is the matching dyadic tolerance and bound ledger;* $M_{\leq W}$ *and* $B_{\leq W}$ *are the monotone and bounded ledger prefixes for the same displayed stages; and* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after those prefixes.*

The structural rows are unchanged: H *gives componentwise* hsame *transport,* C *gives displayed* Cont *replay,* P *gives package provenance, and* N *gives the local* $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$ *row. The surface contains no located supremum row, countable-choice selector, trichotomy branch, Dedekind-cut representation, host-selected limit, or ambient completeness field.*

Theorem 1385.12 (Specker sequence regular-Cauchy obligation)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet discharges its regular-Cauchy obligation through finite-stage induction over the displayed schedule* W*. At each inspected stage the consumer receives only the current rational window* $R_{\leq W}$*, the dyadic tolerance row* $D_{\leq W}$*, the monotone prefix* $M_{\leq W}$*, the boundedness prefix* $B_{\leq W}$*, and the* BHist *provenance transported through* H, C, P, N *before the packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

Proof. Start from the obligation surface of Definition 1406.11. Its visible analytic data are the finite rational stream window, the dyadic tolerance and bound ledger, and the two finite ledger prefixes.

Use the finite-stage induction route of Definition 1406.5. The base prefix exposes the initial rational observation and its matching ledger rows. The step prefix appends one displayed observation, transports the rows through H, and replays the dyadic, monotone, and boundedness entries through C.

The regular-Cauchy obligation is therefore local to each displayed finite prefix. The seal row E is downstream of those rows by Definition 1406.10, so no located supremum, selected limit, countable choice, trichotomy, Dedekind cut, or ambient completeness coordinate is introduced. □

Theorem 1385.13 (Specker sequence no-choice boundary)**.** *The* $\mathsf{SpeckerSequence}^{\uparrow}$ *obligation surface is stable under componentwise* hsame *transport and displayed* Cont *replay without adding any selector that chooses a supremum, a limiting real, a trichotomy branch, or a countable family of witnesses. Any downstream proof that needs one of those objects must supply it outside the packet.*

Proof. Project the transported packet through Definition 1406.3. The classifier shape is preserved on the regular-source row, stream schedule, dyadic ledger, monotone ledger, boundedness ledger, finite seal row, provenance row, and local naming row.

Now restrict to the obligation surface. The rows $R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, E, H, C, P, N$ are exactly the finite rows of Definition 1406.11; transport and replay do not create a new coordinate.

The no-choice boundary follows from the missing coordinate. Since no row stores a supremum selector, limiting real selector, trichotomy decision, countable family choice, Dedekind cut, or completeness witness, no consumer can extract such data from the packet. □

Theorem 1385.14 (Specker sequence Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet factors through the obligation surface and then through the finite seal row* E*. The handoff route is*

$$R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, H, C, P, N \longmapsto E,$$

so the exported real-facing information is a sealed finite-window read rather than a located supremum or a host-selected limit.

Proof. Use Definition 1406.4 to order the route. The source and schedule determine the inspected rational windows, the dyadic ledger supplies tolerance and bound rows, and the monotone and boundedness ledgers are read before E.

Use Definition 1406.11 to package exactly those finite rows as the public obligation surface. The structural rows H, C, P, N preserve and replay the route but do not extend it.

Finally apply Definition 1406.10. The $\mathsf{Real}^{\uparrow}$ handoff remains a finite seal read, so it refuses located supremum data, countable choice, trichotomy, Dedekind cuts, host-selected limits, and ambient completeness. □

Theorem 1385.15 (Specker sequence obligation route)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier routes its monotone bounded ledger, nonlocated supremum refusal,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal data through the displayed finite rows*

$$R, W, D, M, B, E, H, C, P, N.$$

The route uses the BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row, and it contains no coordinate for countable choice, selected supremum data, trichotomy, Dedekind cuts, or a host-selected limiting real.*

Proof. Start with the carrier exposure theorem. It identifies the only analytic rows as the regular rational source, finite schedule, dyadic ledger, monotone ledger, boundedness ledger, and finite real seal.

Use the monotone bounded ledger theorem and finite-stage induction. Together they force every inspected prefix through R, W, D, M, B before the seal row E can be consumed.

Use the non-located supremum boundary and RealUp seal nonescape. Since H, C, P, N only transport, replay, source, and name the same finite route, the packet cannot export a selected supremum, countable choice, trichotomy, Dedekind cut, or ambient completeness coordinate. □

Theorem 1385.16 (Specker sequence regular readback scope)**.** *The regular readback exported by an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet is exactly the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window read selected by* W *and certified through* D, M, B, H, C, P, N *before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Every downstream consumer must read the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; no consumer can extract a countable-choice selector, a located supremum coordinate, a trichotomy branch, or a selected limiting real from the readback.*

Proof. Apply Definition 1406.11. Its visible readback rows are the scheduled regular rational window, the dyadic tolerance and bound ledger, the monotone prefix, the boundedness prefix, and the finite seal.

Apply Definition 1406.12. The regular-Cauchy obligation is local to the displayed finite prefixes, so the readback does not depend on an infinite selector or a located supremum coordinate.

Finally apply Definition 1406.13. Transport and replay preserve the same finite obligation surface and introduce no countable-choice, trichotomy, Dedekind-cut, or selected-limit row. □

Theorem 1385.17 (SpeckerSequence scoped kernel route)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet route through*

$$R, W, D, M, B, E, H, C, P, N$$

is scoped by the finite kernel rows BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The no-choice and non-located-supremum exclusions follow from the absence of rows for countable-choice selectors, located supremum witnesses, trichotomy branches, Dedekind cuts, and host-selected limiting reals, not from external prose.*

Proof. Start with carrier exposure. The accepted packet exposes only the regular rational source row R, finite window row W, dyadic ledger D, monotone ledger M, boundedness ledger B, finite Real seal E, and structural rows H, C, P, N.

Use finite-stage induction and the monotone bounded ledger. Every inspected prefix is read through R, W, D, M, B before any Real-facing seal is available, so the route remains inside the displayed BHist packet and its componentwise hsame and Cont rows.

Apply the no-choice boundary. Since no row records a countable selector, a trichotomy branch, a Dedekind cut, or a chosen limiting real, none of those data can be recovered by replaying H, C, P, N.

Finally use the $\mathsf{Real}^{\uparrow}$ seal nonescape theorem. The seal row E exports only the finite-window Real-facing readback, and therefore the scoped route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ is exhaustive for the packet. □

Theorem 1385.18 (Specker sequence monotone bounded nonconvergence route)**.** *Let* $S = (R, W, D, M, B, E, H, C$ *be an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet. If a consumer reads the packet through the monotone bounded ledger and the regular readback scope, then the consumer receives bounded monotone evidence only as finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows inside the*

displayed BHist *carrier. This route may be used as monotone bounded finite-window data, but it does not yield a located supremum, a selected limit, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier exposure theorem and the monotone bounded ledger. The accepted packet exposes the regular rational source R, the finite schedule W, the dyadic tolerance and bound ledger D, the monotone and boundedness ledgers M, B, the finite seal E, and the structural rows H, C, P, N. The ledger order forces every seal-facing read through

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E.$$

Apply finite-stage induction along W. Each displayed prefix carries only the currently inspected $\mathsf{RegSeqRat}^{\uparrow}$ window, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ row, the monotone and boundedness prefixes, and the transported BHist provenance. The regular readback scope keeps this information in the $\mathsf{StreamName}^{\uparrow}$ schedule and the finite seal route; it never opens an infinite selector.

Now apply the non-located supremum boundary and the scoped kernel route. The packet contains no row for rational-cut location around a supremum, no branch selecting trichotomy, no countable family of witnesses, and no ambient completeness coordinate. The sibling boundaries of chapter 2255, chapter 1405, chapter 1665, and chapter 1666 identify exactly the external consumers that would have to provide those missing rows.

Thus bounded monotone Specker windows are usable finite data inside the BEDC kernel, while the non-located supremum boundary remains intact. The route certifies finite monotone readback, not convergence to a selected $\mathsf{Real}^{\uparrow}$ limit. □

Theorem 1385.19 (Specker sequence completion-refusal induction)**.** *Any Completion-facing consumer reading an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet can induct only over the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*. At each stage the consumer receives the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source window, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the monotone and boundedness ledger prefixes, the finite* $\mathsf{Real}^{\uparrow}$ *seal route, and the structural rows* H, C, P, N*; it receives no located supremum, selected limit, trichotomy branch, countable-choice selector, quotient stream equality, or ambient completion witness.*

Proof. Apply finite-stage induction, Definition 1406.5. The induction variable is the displayed schedule W, so each step exposes only the current regular rational window, dyadic tolerance row, monotone prefix, boundedness prefix, and transported provenance.

Use the regular readback scope, Definition 1406.16. It keeps the readback inside the finite $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before the packet reaches the $\mathsf{Real}^{\uparrow}$ seal.

Apply the no-choice boundary, Definition 1406.13. Transport and replay through H, C, P, N do not add a selector for a supremum, limiting real, trichotomy branch, countable family of witnesses, or quotient stream equality.

Finally use the monotone bounded nonconvergence route, Definition 1406.18. Completion-facing consumers may use the bounded monotone finite-window data, but the packet still refuses ambient completion and any selected limit beyond the displayed finite seal. □

Theorem 1385.20 (Specker sequence Cauchy criterion non-located boundary)**.** *An accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet read through an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *packet still exposes only finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, monotone and bounded ledgers, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The combined read cannot yield a located supremum, selected limit, quotient stream equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the completion-refusal induction of Definition 1406.19. It confines every Completion-facing Specker read to finite $\mathsf{StreamName}^{\uparrow}$ schedule induction over the displayed regular rational windows, dyadic tolerances, monotone and boundedness prefixes, Real seal, and structural rows.

Apply the monotone bounded nonconvergence route of Definition 1406.18. It allows the monotone bounded evidence to be consumed as finite data, but it keeps located supremum, selected limit, trichotomy, countable choice, and ambient completeness outside the packet.

Finally read the same surface through the Cauchy criterion boundary of Definition 1509.15. The criterion contributes only finite windows, threshold and tolerance rows, regular readback, Real seal, and structural rows shared by the packets. It adds no quotient stream equality, selected host limit, located supremum coordinate, or Real completeness witness. □

Theorem 1385.21 (Specker sequence regular-window induction)**.** *Every completion-facing consumer inspecting an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet before a located-supremum row must proceed by finite induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *window. At each step the consumer exposes only the current* $\mathsf{RegSeqRat}^{\uparrow}$ *source prefix, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the monotone ledger, the bounded ledger, and the structural* BHist *rows preserved by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

Proof. Start with the finite-stage induction theorem, Definition 1406.5. Its induction variable is the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, so every inspected stage is a finite prefix read rather than a selected infinite stream.

Apply the regular-Cauchy obligation of Definition 1406.12. The obligation exposes only the current $\mathsf{RegSeqRat}^{\uparrow}$ source prefix, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the monotone and boundedness ledger prefixes before any completion-facing conclusion is visible.

Finally apply the Cauchy-criterion non-located boundary of Definition 1406.20. Transport, replay, provenance, and local naming preserve the same structural BHist rows, and no located-supremum row, selected limit, quotient stream equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness is added. □

Theorem 1385.22 (Specker sequence finite-window refusal induction)**.** *For every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet, finite-stage induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule preserves exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *source prefix,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, monotone ledger, bounded ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. No induction step can introduce a located-supremum row, selected-limit row, quotient stream equality, countable-choice selector, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Induct on the inspected finite window of the displayed $\mathsf{StreamName}^{\uparrow}$ schedule. In the empty window, the carrier exposes only its structural BHist rows and the already named refusal boundary, so no located-supremum or selected-limit coordinate is present.

For the step, extend the inspected window by one displayed source observation. Apply Definition 1406.5 to keep the new observation inside the same finite schedule, and apply Definition 1406.21 to preserve the $\mathsf{RegSeqRat}^{\uparrow}$ prefix, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, monotone ledger, bounded ledger, and structural transport, replay, provenance, and naming rows.

Finally use Definition 1406.20. The terminal $\mathsf{Real}^{\uparrow}$ seal remains a finite boundary read, and the missing located-supremum, selected-limit, quotient, choice, trichotomy, and ambient-completeness rows have no carrier coordinate from which an induction step could create them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$**.**

Constructive story: A SpeckerSequence$^{\uparrow}$ packet is generated from a RegSeqRat$^{\uparrow}$ computable rational source, a StreamName$^{\uparrow}$ finite-window schedule, DyadicRatCore$^{\uparrow}$ tolerance and bound rows, a monotone ledger, a boundedness ledger, a finite Real$^{\uparrow}$ seal boundary, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{SpeckerSequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1406.1, Definition 1406.2, Definition 1406.3, Definition 1406.4, Definition 1406.5, Definition 1406.7, Definition 1406.8, Definition 1406.9, Definition 1406.10, Definition 1406.11, Definition 1406.12, Definition 1406.13, Definition 1406.14, Definition 1406.15, Definition 1406.16, Definition 1406.17, Definition 1406.18, Definition 1406.19, and Definition 1406.20 declare the finite Specker-sequence carrier, transport, monotone bounded ledger, finite-stage induction route, non-located supremum boundary, located-supremum refusal, public located-supremum consumer boundary, public RealUp seal non-escape surface, obligation surface, regular-Cauchy obligation, no-choice boundary, public Real handoff, obligation route, regular readback scope, scoped kernel route, monotone bounded nonconvergence route, Completion-refusal induction route, and Cauchy-criterion non-located boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{SpeckerSequence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceCarrier\_exposure`

Bridge status: none

Not claimed: This chapter does not close Real completeness, located supremum construction, Dedekind cuts, countable choice, trichotomy, bridge checking, or selected limiting reals.

Upgrade path: Further verification work is bridge checking for the existing finite-stage, no-choice, Real-handoff, and Real-seal theorem surface.

Chapter 1386

A Concrete Naming Certificate for $\mathsf{BisectionRootIsolation}^{\uparrow}$

Following Bishop and Bridges, constructive root isolation is read here as a finite sign-change bisection packet over located intervals rather than as a host root-existence theorem. The $\mathsf{BisectionRootIsolation}^{\uparrow}$ surface binds endpoint windows, sign rows, branch choices, regular rational readback, and real-name seal rows into one local NameCert packet. It consumes the located interval and dyadic bisection certificates of chapter 394 and chapter 361, with tolerance and readback rows supplied by chapter 291, chapter 126, chapter 301, and chapter 38.

Definition 1386.1 (Bisection root-isolation carrier)**.** *A* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (J, B, F, E, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, exposing the lower and upper endpoint windows, rational cell ledger, optional dyadic refinements,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal-facing rows;*

2. *B is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger whose retained interval rows refine the endpoint packet of J by finite midpoint comparisons;*

3. *F is the function-inspection ledger, consisting only of finite rational and dyadic reads used to evaluate the displayed sign rows at the retained endpoints and midpoints;*

4. *E records the sign rows at the retained lower and upper endpoints, together with the midpoint sign rows used to choose each branch;*

5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window ledger exposed to downstream consumers after the retained interval is selected;*

6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained endpoint windows and the dyadic tolerance rows supplied through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*

7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal surface for the retained lower and upper endpoint names and for any real-facing consumer of the isolating window;*

8. *H stores the componentwise* hsame *transports for the located interval packet, bisection ledger, sign rows, stream windows, regular rational readbacks, real seal rows, and local certificate row;*

9. *C stores the displayed* Cont *replay routes for endpoint inspection, midpoint inspection, branch selection, retained-window readback, regular-sequence handoff, and real-name seal consumption;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite sign-change bisection packet. It has no coordinate for a host root, a general intermediate value theorem, trichotomy on reals, a completed intersection of nested intervals, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationUp`
Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationTasteGate single carrier alignment`

Theorem 1386.2 (Bisection root-isolation NameCert obligations)**.** *For every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier* $Q = (J, B, F, E, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the displayed finite packet of Definition 1407.1;*
2. *endpoint-window admission through the* $\mathsf{LocatedInterval}^{\uparrow}$ *rows of J and the retained interval rows of B;*
3. *sign-ledger exactness, requiring each branch choice in B to be justified by one displayed sign row in E read through F;*
4. *retained-window handoff through W, R, and S;*
5. *componentwise transport and replay through H and C;*
6. *package provenance and local naming through P and N.*

No further analytic obligation is part of the packet.

Proof. Project the carrier of Definition 1407.1. The endpoint packet J supplies the located lower and upper endpoint windows, while the bisection ledger B supplies the finite retained-interval chain.

The sign-ledger obligation is local to F and E. Each midpoint inspection reads a finite rational or dyadic approximation row, records the displayed sign row, and then selects the corresponding branch row in B.

The retained-window handoff is the route already available from chapter 361 after the endpoints are read through chapter 394. The tolerance rows come from chapter 291; the window, readback, and seal rows are then routed through chapter 126, chapter 301, and chapter 38.

Finally, H, C, P, and N account for transport, replay, provenance, and local naming. Since the carrier stores only these finite rows, it cannot export host root existence, an intermediate value theorem, real trichotomy, a completed interval intersection, or completeness for $\mathsf{Real}^{\uparrow}$. □

Theorem 1386.3 (Bisection root-isolation RegSeqRat window handoff)**.** *For every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier and every unary precision request n, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposed by the isolating window factors through the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix that reaches the retained window at tolerance 2^{-n}, the displayed sign rows at retained endpoints, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, and the replay rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRegSeqRatWindowHandoff`

Proof. Begin with the unary request n. The dyadic tolerance row 2^{-n} is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance, and the branch prefix in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger selects the retained interval window at that tolerance.

The located endpoint packet then supplies the lower and upper endpoint windows for that retained interval. The sign ledger contributes only the displayed endpoint signs and midpoint signs already used to select the branch prefix; it does not introduce a root object.

Route the retained endpoint windows through the $\mathsf{StreamName}^{\uparrow}$ rows and then through the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The replay remains inside H, C, P, and N, so every rational read exposed to the regular-sequence consumer is one of the retained finite endpoint windows. □

Theorem 1386.4 (Bisection root-isolation Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier is reached only after the retained* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint windows, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the displayed endpoint sign rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the rows* H, C, P, N *have been read. The seal exposes the finite isolating window and exports no host root existence, general intermediate value theorem, real trichotomy, completed interval intersection, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRealSealNonescape`

Proof. Read the real-facing coordinate S in Definition 1407.1. It is not an independent real object; it is stored with the endpoint packet, branch ledger, sign rows, stream windows, regular readback, transport rows, replay routes, provenance, and local certificate.

Apply Definition 1407.3. The retained endpoint windows first pass through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces at the requested dyadic tolerance.

Only after that route is fixed can the $\mathsf{Real}^{\uparrow}$ seal be read. The seal therefore records the same finite isolating window and no additional analytic witness. Because no coordinate of Q stores a host root, an intermediate value theorem, trichotomy data, a completed intersection, or completeness for $\mathsf{Real}^{\uparrow}$, the real-facing consumer cannot escape to any of those objects. □

Theorem 1386.5 (Bisection root-isolation finite sign-change certificate)**.** *For every accepted* $\mathsf{BisectionRootIsolation}$ *carrier* $Q = (J, B, F, E, W, R, S, H, C, P, N)$*, every sign-change assertion visible to the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsola}}$ *is certified by finite endpoint and midpoint rows in* F*,* E*, and* B*. The certificate contains no host intermediate value theorem, no real trichotomy oracle, no root object, and no completed intersection of retained intervals.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationFiniteSignChangeCertificate`

Proof. Project the carrier rows of Definition 1407.1. The only function-inspection data are the finite rational and dyadic reads in F, and the only sign data are the endpoint and midpoint rows displayed in E.

Apply Definition 1407.2. Each branch choice in B must be justified by one of those displayed rows, so the sign-change surface is exhausted by the finite ledger already stored in the packet.

The excluded analytic objects are absent from the carrier coordinates. Thus the certificate can replay the accepted sign rows and retained branch choices, but it cannot export a host root, a general intermediate value theorem, trichotomy on reals, or a completed interval intersection. □

Theorem 1386.6 (Bisection root-isolation retained-window refinement)**.** *For every accepted* $\mathsf{BisectionRootIsolation}$ *carrier and every unary precision request* n*, the retained isolating window at tolerance* 2^{-n} *factors through the* $\mathsf{DyadicBisection}^{\uparrow}$ *retained branch prefix, the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposed by the carrier.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRetainedWindowRefinement`

Proof. Begin with the unary request n. The tolerance row is the $\mathsf{DyadicRatCore}^{\uparrow}$ row already named in Definition 1407.1, and the branch prefix is the retained prefix stored in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger B.

The endpoint rows come from the located interval packet J. Once the retained prefix is fixed, the selected lower and upper windows pass through the $\mathsf{StreamName}^{\uparrow}$ rows W.

Apply Definition 1407.3. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads back exactly those retained endpoint windows at the requested tolerance, with transport and replay confined to H, C, P, and N. □

Theorem 1386.7 (Bisection root-isolation downstream boundary)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier receives only the retained isolating window determined by J, B, W, R, S, H, C, P, and N. Such a consumer cannot extract host root existence, a completed interval intersection, a general intermediate value theorem, real trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationDownstreamBoundary`

Proof. Read the downstream-facing seal row through Definition 1407.4. The seal is reached only after the retained located endpoints, branch ledger, stream windows, regular readback, and replay rows have already been fixed.

The retained-window route is the one described in Definition 1407.6. It delivers a finite window at the requested tolerance, not an independent analytic object.

Since Definition 1407.5 confines the sign-change evidence to finite endpoint and midpoint rows, no downstream consumer can recover the excluded host principles from the packet. □

Theorem 1386.8 (Bisection root-isolation public export)**.** *Every public consumer of an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier must read the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the structural* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows in that order before using the isolation certificate. The exported object is the finite retained isolating window with displayed sign-change evidence; it is not a selected root, a real-trichotomy decision, quotient equality of real names, a completed nested-interval point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $Q = (J, B, F, E, W, R, S, H, C, P, N)$ of Definition 1407.1. The carrier already orders the public data: J supplies the located endpoint packet, B supplies the retained dyadic branch ledger, and the dyadic tolerance rows are the arithmetic support named in the same packet.

Apply Definition 1407.2. It forces the endpoint-window admission and sign-ledger exactness to be read before retained-window handoff, so the public consumer cannot inspect W, R, S until the located interval, bisection ledger, tolerance support, and displayed sign rows have been exposed.

Next apply Definition 1407.3 and Definition 1407.4. These statements route the retained endpoint windows through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal without producing an independent real object.

Finally use Definition 1407.7. The structural rows H, C, P, N transport, replay, package, and name exactly that finite window. Since the carrier has no coordinate for a selected root, real trichotomy, quotient real equality, a completed nested-interval point, or ambient completeness, the public export is exhausted by the ordered rows listed in the statement. □

Theorem 1386.9 (Bisection root-isolation sign-ledger determinacy)**.** *Let $Q = (J, B, F, E, W, R, S, H, C, P, N)$ be an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier. For every retained branch row in B, the corresponding finite inspection row in F and displayed sign row in E determine the branch visible to the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *packet. The visible branch decision is therefore a finite* BHist *route through F, E, B, H, C, P, N, not a host trichotomy comparison or an external root-location oracle.*

Proof. Project the carrier rows F, E, B from Definition 1407.1. The function-inspection ledger F is the only source of endpoint and midpoint reads, and E is the only sign ledger stored for those reads.

Apply the NameCert obligations of Definition 1407.2. They require each branch choice in B to be justified by one displayed sign row in E read through F, with transport and replay confined to H, C, P, N.

Thus two accepted replays with the same inspected row and the same displayed sign row expose the same retained branch row to the packet. Since no coordinate stores real trichotomy, a host root, or an external oracle, the branch visible to $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ is determined exactly by the finite sign ledger. □

Theorem 1386.10 (Bisection root-isolation branch monotonicity)**.** *For an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier, extending the dyadic bisection depth refines the retained interval window and never enlarges the* $\mathsf{Real}^{\uparrow}$*-seal-facing interval. The refinement route factors through the retained* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix, the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the displayed sign ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows.*

Proof. Start with two unary depth requests where the second extends the first. The retained branch prefix lives in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger B, and Definition 1407.6 routes each requested depth through the located endpoint rows, dyadic tolerance row, stream windows, and regular readback.

The extension step can only append midpoint inspection and sign rows already admitted by Definition 1407.9. It therefore selects a subwindow of the earlier retained window rather than a new ambient interval.

The downstream seal is constrained by Definition 1407.4. Since every real-facing read passes through the retained window already chosen by the branch ledger, increasing depth can refine the exposed interval but cannot enlarge it or introduce host root existence, completed interval intersection, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1386.11 (Bisection root-isolation depth coverage)**.** *Every unary precision request made to an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier factors through some finite branch prefix in B, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window rows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before it reaches a* $\mathsf{Real}^{\uparrow}$ *seal. The packet therefore covers precision requests by finite bisection depth rather than by a completed nested-interval object.*

Proof. Begin with a unary precision request n. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance 2^{-n}, and Definition 1407.6 selects a retained branch prefix whose window is read at that tolerance.

The endpoint windows then pass through the $\mathsf{StreamName}^{\uparrow}$ rows. Apply Definition 1407.3 to read the same retained windows through $\mathsf{RegSeqRat}^{\uparrow}$ at the requested precision.

Finally route the readback to the real-facing boundary using Definition 1407.4. The request is exhausted by the finite prefix, tolerance, window, and readback rows, so no completed branch, host root, general intermediate value theorem, trichotomy oracle, or ambient completeness principle is exposed. □

Theorem 1386.12 (BisectionRootIsolation public isolation certificate)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *isolation certificate exposes the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the displayed sign-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal before any public isolation read is available. The public read is therefore the finite retained isolating window with its displayed sign-change ledger, not a host root, a general intermediate value theorem, real trichotomy, a completed interval intersection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the retained-window refinement of Definition 1407.6. It routes each requested precision through the located endpoint packet, the retained $\mathsf{DyadicBisection}^{\uparrow}$ branch prefix, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the branch ledger and sign stability supplied by Definition 1407.9 together with Definition 1407.5. These theorems keep every branch choice inside the finite inspection row, displayed sign row, and retained branch row; no host trichotomy or external root-location oracle is added.

Finally apply the public export and Real-seal boundary of Definition 1407.8 and Definition 1407.4. The public consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only after the endpoint, branch, sign, tolerance, stream-window, and regular-readback rows have been displayed, so the exported isolation certificate is exactly the finite L10 window named in the statement. □

Theorem 1386.13 (Bisection root-isolation interval nesting)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *packet exposes a nested chain of retained windows before its public isolation certificate is consumed. For each retained branch prefix, the lower and upper endpoints of the selected window pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows; a later retained prefix selects a subwindow of the earlier one and cannot replace that chain by a host root, a completed interval intersection, real trichotomy, or ambient completeness.*

Proof. Induct over the retained-window refinement rows of the $\mathsf{DyadicBisection}^{\uparrow}$ ledger. The initial row is the located endpoint packet J of Definition 1407.1; its endpoints are already routed through the displayed dyadic tolerance, stream-window, regular-readback, and real-seal rows.

For the refinement step, use Definition 1407.9. The next retained branch is determined by the finite inspection row and displayed sign row, so the branch ledger can only select one of the two displayed subwindows of the previous retained window.

Apply Definition 1407.6 and Definition 1407.12 to the selected prefix. They route its endpoints through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before public consumption. The nonescape boundary of Definition 1407.4 excludes the host analytic objects, so the public certificate sees a nested retained-window chain rather than an external root-selection principle. □

Theorem 1386.14 (Bisection root-isolation bridge schema)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *packet exports a bridge-readable schema consisting of the located interval source, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bisection ledger, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the finite sign ledger, retained-window monotonicity, and the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *rows. The schema is a finite isolation route and does not assert a general intermediate value theorem, host root existence, completed interval intersection, real trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public isolation certificate of Definition 1407.12. It exposes the located endpoint rows, the retained dyadic branch ledger, the displayed sign-window rows, the dyadic tolerance rows, stream windows, regular readback, and the real seal before any public isolation read.

Use interval nesting from Definition 1407.13. The retained prefixes form the bridge-visible window chain, and each endpoint read is still routed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Add branch monotonicity and depth coverage from Definition 1407.10 and Definition 1407.11. These rows identify the retained-window monotonicity and the finite prefix selected for each unary precision request.

Finally apply the Real seal nonescape boundary of Definition 1407.4. The bridge-readable schema is therefore exhausted by the displayed carrier rows and the local $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ route, with no host analytic principle available beyond the finite isolation packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$.

Constructive story: A $\mathsf{BisectionRootIsolation}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint packet, a $\mathsf{DyadicBisection}^{\uparrow}$ branch ledger, finite sign rows at retained endpoints and midpoints, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal rows, displayed hsame and Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1407.1, Definition 1407.2, Definition 1407.3, Definition 1407.4, Definition 1407.5, Definition 1407.6, Definition 1407.7, and Definition 1407.12 bind the carrier, NameCert obligations, RegSeqRat window handoff, Real seal nonescape, finite sign-change certificate, retained-window refinement, downstream boundary, public isolation certificate, and BEDC rows BHist, hsame, Cont, Pkg, NameCert, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to a finite sign-change bisection certificate over located intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationDownstreamBoundary`

Bridge status: none

Not claimed: This chapter does not close host root existence, a general intermediate value theorem, real trichotomy, completed interval intersection, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The remaining work is Lean verification of the displayed carrier, sign-ledger exactness, retained-window handoff, and Real-seal nonescape, not paper obligation closure.

Chapter 1387

A Concrete Naming Certificate for $\mathsf{DiniTheorem}^{\uparrow}$

Following Dini's theorem in constructive analysis, the $\mathsf{DiniTheorem}^{\uparrow}$ packet records only finite compact-window and monotone convergence data. It consumes the compact metric source of chapter 129, the compact-uniform-continuity surface of chapter 388, the continuous-map rows of chapter 130, the located real boundary of **??**, and the Real seal of chapter 38. The packet is a finite NameCert for a displayed monotone convergence window; it does not import open-cover compactness, sequential compactness, a host supremum, a selected limit, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1387.1 (DiniTheorem carrier)**.** *A* $\mathsf{DiniTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, F, M, W, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, carrying the* $\mathsf{CompactMetric}^{\uparrow}$ *name, its finite inspection windows, and the source-side metric classifier used by downstream rows.*
2. *F is a finite monotone family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows over X, together with the pointwise direction ledger that records, for each inspected index pair, which displayed real-name comparison is monotone.*
3. *M is the compact-uniform-continuity ledger imported from* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *for the displayed rows of F on the windows of X.*
4. *W is the shared tolerance-window ledger: for each requested dyadic tolerance it lists the finite X-windows, family indices, continuity moduli, and located real readbacks inspected by the packet.*
5. *R is the boundary real-seal row, storing the* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows reached by the displayed window reads and by no independent host limit coordinate.*
6. *H stores componentwise* hsame *transports for the compact source, continuous-map family, monotonicity ledger, uniform-continuity ledger, tolerance windows, and real-seal rows.*
7. *C stores the displayed* Cont *replay routes from compact windows to continuity rows, from monotone family rows to located real comparisons, and from the shared tolerance ledger to the boundary real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{DiniTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open cover, sequential compactness witness, ambient supremum, selected limit function, quotient equality of real names, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniTheoremUp.DiniTheoremTasteGate single carrier alignment`

Theorem 1387.2 (DiniTheorem NameCert obligations)**.** *For an accepted* $\mathsf{DiniTheorem}^{\uparrow}$ *carrier* $D = (X, F, M, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiniTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed compact source, monotone continuous-map family, uniform-continuity ledger, tolerance-window ledger, boundary real rows, transport, continuation, provenance, and naming rows;*

2. *classifier transport along H and C, preserving the same compact windows, family indices, monotone comparisons, continuity moduli, and real-seal reads;*

3. *monotone ledger exactness, requiring every comparison used by a convergence read to occur in the finite family row F;*

4. *shared-window exactness, requiring every tolerance request to be answered by the displayed compact windows and compact-uniform-continuity rows listed in W and M;*

5. *nonescape, requiring every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read to pass through the rows W, R, H, C, P, N and through no open-cover, sequential-compactness, host-supremum, selected-limit, quotient-equality, or completeness coordinate.*

Proof. Carrier admission is projection from the packet $D = (X, F, M, W, R, H, C, P, N)$. The compact metric row, monotone family, compact-uniform-continuity ledger, shared tolerance windows, real-seal boundary, transports, continuation routes, provenance, and local name are exactly the displayed coordinates of Definition 1387.1.

Classifier transport reads H componentwise and then replays C. A transported consumer sees the same compact source windows, the same finite family indices, the same monotone comparison rows, the same continuity-modulus rows, and the same real-seal boundary route.

Monotone ledger exactness follows because F is the only row that stores family comparison data. A convergence-facing read cannot introduce a private order comparison or a detached tail relation without leaving the accepted carrier.

Shared-window exactness follows from M and W. The continuity ledger supplies the compact-uniform-continuity rows, while the tolerance ledger names the finite windows and family indices consumed for each displayed tolerance.

Finally, the nonescape clause is an inventory check on the carrier. The rows R, H, C, P, N expose the located and Real boundary route, and none of the excluded compactness, supremum, selected-limit, quotient-equality, choice, or completeness principles appears as a coordinate. □

Theorem 1387.3 (DiniTheorem monotone uniform window)**.** *Let $D = (X, F, M, W, R, H, C, P, N)$ be an accepted* $\mathsf{DiniTheorem}^{\uparrow}$ *carrier. For every displayed tolerance request recorded in W, every accepted convergence read factors through a single finite route*

$$X, F, M, W, R, H, C, P, N.$$

The route reads the compact source windows from X, the finite monotone family comparisons from F, the compact-uniform-continuity rows from M, and the shared tolerance window from W before any $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *boundary row in R is consumed. No accepted read may replace this route by open-cover compactness, sequential compactness, a host supremum, a selected limit, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tolerance request in W. It names the compact source windows, the family indices, and the continuity-modulus rows that are inspected at the requested dyadic scale.

The source windows are read through X, and the family rows are read through F. Since F includes the monotone ledger, each comparison used by the convergence read is already a displayed family comparison, not an ambient real-order assertion.

The compact-uniform-continuity row M then identifies the finite windows sufficient for all displayed source points in the request. The route remains finite because M and W list the inspected windows explicitly.

Only after these rows are visible can the read reach R. The continuation row C records this order, while H preserves it under transport and P, N package and name the same route.

Thus every accepted convergence read factors through $X, F, M, W, R, H, C, P, N$. Any route that inserts open-cover compactness, sequential compactness, host suprema, selected limits, quotient real equality, or ambient completeness uses data absent from the carrier. □

Theorem 1387.4 (DiniTheorem nonescape boundary)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *boundary read exported by an accepted* $\mathsf{DiniTheorem}^{\uparrow}$ *carrier is downstream of the displayed finite compact-window monotonicity route. Equivalently, if a consumer reaches the boundary row R, then the consumer has already read the same compact source row X, monotone family row F, compact-uniform-continuity ledger M, shared tolerance-window row W, componentwise transports H, continuation replay C, package provenance P, and local naming row N.*

Proof. Inspect the carrier. The only boundary coordinate is R, and it is stored together with the compact source, monotone family, uniform-continuity ledger, and shared tolerance-window rows.

The continuation row C records the admissible replay into R. Its predecessors are the source-window read, the family monotonicity read, the uniform-continuity read, and the tolerance-window read; no separate boundary route is recorded.

Transport cannot create another route. The row H moves the displayed coordinates componentwise, so a transported boundary read still consumes the same X, F, M, W, R surface.

The package and naming rows P, N therefore certify only this route. Because the carrier contains no open-cover, sequential-compactness, host-supremum, selected-limit, quotient-equality, countable-choice, or completeness coordinate, a boundary read using any such datum is outside the accepted $\mathsf{DiniTheorem}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiniTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiniTheorem}^{\uparrow}$ packet is generated from a compact metric source row, a finite monotone family of continuous-map rows, compact-uniform-continuity rows, shared tolerance-window rows, located and Real boundary seal rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiniTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 1387.1, Definition 1387.2, Definition 1387.3, and Definition 1387.4 seed the finite compact-window monotone convergence certificate for DiniTheoremUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiniTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, host supremum, selected limit, quotient real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone ledger exactness, shared-window exactness, real-seal nonescape, and classifier transport for the same finite packet.

Chapter 1388

A Concrete Naming Certificate for $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet records the finite selector boundary that can be read before any classical Bolzano–Weierstrass conclusion is asserted. The route is anchored by the monotone selector surface of chapter 1405, the bounded monotone Cauchy witness of chapter 535, the bounded monotone convergence seal of chapter 918, the stream-window interface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name boundary of chapter 38. It packages a displayed finite handoff from bounded sequence windows to monotone selected windows, not sequential compactness, a host real completeness theorem, or a choice-selected subsequence.

Definition 1388.1 (Bolzano–Weierstrass selector carrier)**.** *A* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, M, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded source-sequence window ledger; it records only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows currently inspected and the displayed boundedness rows attached to those windows.*
2. *M is the monotone selector ledger read through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*; each displayed entry records a selector position, its source window, and the comparison row that makes the selected window monotone inside the finite packet.*
3. *Q is the cofinal evidence row for the inspected selector thresholds, limited to thresholds that occur in the packet.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window family obtained from B, M, Q.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read from the selected windows before any regular rational or real-facing handoff is available.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the selected dyadic windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only through the bounded monotone witness and bounded monotone convergence seal routes after B, M, Q, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for bounded windows, selector entries, cofinal evidence, selected windows, dyadic rows, regular handoff, Real seal, continuation rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *routes from B, M, Q to W, from W to D, from D to R, and from the bounded monotone witness and convergence seal packets to E.*

10. *P is the* Pkg *provenance over* BolzanoWeierstrassSelector$^{\uparrow}$, MonotoneSubsequence$^{\uparrow}$, BoundedMonotoneCauchyWit BoundedMonotoneConvergenceSeal$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BolzanoWeierstrassSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for full Bolzano–Weierstrass, sequential compactness, a choice-selected subsequence, quotient-stream equality, host real equality, a least cluster selector, or Real$^{\uparrow}$ *completeness.*

Theorem 1388.2 (Bolzano–Weierstrass selector subsequence handoff)**.** *Every accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet hands bounded source windows to selected monotone windows only through the route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R.$$

The first part of the route factors through MonotoneSubsequence$^{\uparrow}$, *and the regular-facing handoff is read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *before any* Real$^{\uparrow}$ *seal row is consumed.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorSubsequenceHandoff`

Proof. Project the carrier of Definition 1408.1. The bounded source row B, selector ledger M, inspected-threshold cofinality row Q, selected windows W, dyadic ledger D, regular handoff R, and structural rows H, C, P, N are the only rows visible before the seal coordinate.

Use the monotone selector surface of chapter 1405. It identifies the selector ledger and cofinal evidence as finite displayed rows, so the selected windows are obtained by packet replay rather than by countable choice or by an ambient subsequence object.

Next read the selected windows through the terminal finite route. The StreamName$^{\uparrow}$ windows provide W, the DyadicRatCore$^{\uparrow}$ ledger provides D, and the RegSeqRat$^{\uparrow}$ handoff provides R.

The transport, continuation, package, and local NameCert rows only preserve this displayed route. Since no other selector coordinate is present, every accepted subsequence handoff factors through $B, M, Q, W, D, R, H, C, P, N$. □

Theorem 1388.3 (Bolzano–Weierstrass selector Real seal boundary)**.** *Every real-facing read of an accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet reaches the seal row E only after the finite selector route*

$$B, M, Q, W, D, R, E$$

has passed through the bounded monotone Cauchy witness and bounded monotone convergence seal packets. The consumer receives a Real boundary for the selected regular handoff, not full Bolzano–Weierstrass, sequential compactness, a choice-selected subsequence, quotient-stream equality, host real equality, or Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRealSealBoundary`

Proof. Begin with Definition 1408.2. It exposes B, M, Q, W, D, R before a Real-facing coordinate can be consumed, and it makes the selected regular handoff the only path from bounded windows to the seal-facing surface.

Now pass through the bounded monotone witness and convergence seal chapters. The witness route of chapter 535 supplies the bounded monotone Cauchy surface, while the seal route of

chapter 918 exposes the Real handoff only after the finite stream, dyadic, regular, and seal rows are displayed.

Finally inspect the excluded data. The carrier has rows for bounded windows, monotone selector entries, cofinal evidence, selected windows, dyadic tolerance rows, a regular handoff, a Real seal, transports, continuations, provenance, and local naming, but it has no row for sequential compactness, arbitrary choice, quotient streams, host equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Thus the Real-facing read is a boundary seal for the displayed finite route and nothing beyond that route is exported. □

Theorem 1388.4 (Bolzano–Weierstrass selector bounded-monotone route)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real-facing row* E *is available only after the finite route*

$$\mathsf{MonotoneSubsequence}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow} \longmapsto \mathsf{Stream}$$

has displayed the rows B, M, Q, W, D, R, E*. The packet therefore records a bounded monotone induction path into the* $\mathsf{Real}^{\uparrow}$ *seal, not full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.TasteGate.BolzanoWeierstrassSelectorBoundedMonotoneRo`

Proof. Project the carrier of Definition 1408.1. The rows B, M, Q, W, D, R, E are the only Real-facing path before transport, continuation, provenance, and local naming are applied.

Apply Definition 1408.2. It exposes the finite selector route from bounded source windows through monotone selected windows and then through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Apply Definition 1408.3. The Real seal row is consumed only after the bounded monotone Cauchy witness and bounded monotone convergence seal packets have been displayed.

Combining these two chapter routes gives the stated induction path from $\mathsf{MonotoneSubsequence}^{\uparrow}$ through the bounded monotone witness, convergence seal, stream, dyadic, regular, and Real rows. The carrier has no coordinate for classical compactness, countable choice, quotient streams, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$, so no accepted read can export those objects. □

Theorem 1388.5 (Bolzano–Weierstrass selector window obligation)**.** *For every accepted* BolzanoWeierstrassSelecto *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, each finite selected-window read is admissible only when the bounded source window row* B*, monotone selector ledger* M*, inspected-threshold row* Q*, selected* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, dyadic tolerance row* D*, regular handoff row* R*, Real seal row* E*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *are all read as one displayed packet. The route factors through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorWindowObligation`

Proof. Project Definition 1408.1. The rows $B, M, Q, W, D, R, E, H, C, P, N$ are the entire accepted packet, so a selected-window consumer cannot detach W from the bounded source and monotone selector rows.

Apply Definition 1408.2. It supplies the finite selector route from B, M, Q to W, D, R through $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Finally apply Definition 1408.4. The bounded monotone witness and convergence seal place E behind the same route, while H, C, P, N transport, replay, source, and name the packet. □

Theorem 1388.6 (Bolzano–Weierstrass selector finite-window induction)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any finite extension of the inspected selector-threshold ledger that preserves the bounded source window row* B *and the monotone selector ledger* M *extends the selected-window route only by replaying*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The induction exports no selected infinite subsequence, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or additional compactness principle.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteWindowInduction`

Proof. Induct over the displayed finite selector-threshold ledger. The base ledger is the selected-window route already bound by Definition 1408.5: the consumer reads $B, M, Q, W, D, R, E, H, C, P, N$ as one finite packet and receives no external subsequence object.

For the successor step, append one inspected threshold that preserves the same bounded source row B and monotone selector ledger M. The subsequence handoff of Definition 1408.2 admits only the corresponding selected $\mathsf{StreamName}^{\uparrow}$ window in W, with the dyadic and regular rows still downstream of that window.

Finally apply Definition 1408.4 and Definition 1408.5. They force the route through D, R, E after the selected window, while H, C, P, N merely transport, replay, source, and name the same finite packet. Thus every finite extension is a replay of the displayed route and not an export of compactness, quotient equality, host equality, or ambient completeness. □

Theorem 1388.7 (Bolzano–Weierstrass selector regular-Cauchy extraction route)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the finite selected-window induction together with the bounded-monotone route determines a regular-Cauchy extraction surface through*

$$W \longmapsto D \longmapsto R \longmapsto E.$$

Every extracted finite window is read through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The route exports no selected infinite subsequence, host cluster point, sequential compactness theorem, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyExtractionRout`

Proof. Start with the subsequence handoff of Definition 1408.2. It exposes only selected finite $\mathsf{StreamName}^{\uparrow}$ windows in W after the bounded source row B, monotone selector ledger M, and inspected-threshold row Q have been read.

Apply the finite-window induction of Definition 1408.6. Each finite extension replays the same route from B, M, Q to W and then to the dyadic, regular-rational, and Real-seal rows.

Use the bounded-monotone route of Definition 1408.4 together with the window obligation of Definition 1408.5. These force the downstream read through D, R, E before any cluster-facing consumer may inspect the packet.

Finally apply the Real seal boundary of Definition 1408.3. Since the carrier has no coordinate for an infinite selected subsequence, host cluster point, sequential compactness theorem, quotient stream equality, or ambient completeness row, the extraction surface is exactly the displayed regular-Cauchy route. □

Theorem 1388.8 (Bolzano–Weierstrass selector cluster route)**.** *For every accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the cluster-facing read is exactly the finite route*

$$B, M, Q, W, D, R \longmapsto E$$

through MonotoneSubsequence$^{\uparrow}$*,* BoundedMonotoneCauchyWitness$^{\uparrow}$*,* BoundedMonotoneConvergenceSeal$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. It records a bounded monotone selected-window route to a Real seal, not a classical cluster-point theorem or a choice-selected subsequence.*

Proof. Start from the window obligation in Definition 1408.5. It binds $B, M, Q, W, D, R, E, H, C, P, N$ into one finite packet and prevents a consumer from reading a cluster-facing seal from an uninspected window.

Use the bounded-monotone route of Definition 1408.4. The row E is available only after the monotone selector has passed through the bounded monotone Cauchy witness and bounded monotone convergence seal.

The structural rows add no compactness principle. They only transport and replay the same carrier through H, C, P, N, so the exported cluster-facing object is the displayed Real seal boundary for the finite route. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorClusterRoute`

Theorem 1388.9 (Bolzano–Weierstrass selector nonescape)**.** *For every accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, no consumer can export full Bolzano–Weierstrass compactness, host sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or ambient* Real$^{\uparrow}$ *completeness from the packet. The only exported Real-facing data are the rows obtained from* $B, M, Q, W, D, R, E, H, C, P, N$ *through* MonotoneSubsequence$^{\uparrow}$*,* BoundedMonotoneCauchyWitness$^{\uparrow}$*,* BoundedMonotoneConvergenceSeal$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

Proof. Project the carrier. Its visible data are the bounded source windows, monotone selector ledger, inspected-threshold row, selected windows, dyadic tolerances, regular handoff, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1408.8. The cluster-facing read reaches E only through the finite bounded-monotone route and the stream, dyadic, regular, and Real rows named in the statement.

The excluded objects have no row in the carrier and no row in the cited route. Therefore an accepted consumer may replay the finite selector packet, but it cannot export compactness, choice, quotient equality, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorNonescape`

Theorem 1388.10 (Bolzano–Weierstrass selector monotone-subsequence obligations)**.** *For every accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the monotone-subsequence surface consists of exactly three finite obligations:*

1. *finite selector admission for the displayed monotone selector ledger* M *and inspected-threshold cofinality row* Q *over the bounded source windows* B*;*
2. *monotone selected-window transport from* B, M, Q *to the selected* StreamName$^{\uparrow}$ *windows* W *through componentwise* hsame *transport and displayed* Cont *replay;*
3. *bounded-monotone Real-seal nonescape from* W, D, R *to* E *through* BoundedMonotoneCauchyWitness$^{\uparrow}$*,* BoundedMonotoneConvergenceSeal$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*.*

The obligations do not claim full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Read the selector rows from Definition 1408.1. The packet admits only bounded source windows, the monotone selector ledger, inspected-threshold cofinality data, selected windows, dyadic tolerances, regular readback, the Real seal, and the structural rows.

Apply Definition 1408.2. It turns the first two obligations into the displayed route $B, M, Q \longmapsto W \longmapsto D \longmapsto R$, with the monotone selector surface supplied by $\mathsf{MonotoneSubsequence}^{\uparrow}$ and preserved by H, C, P, N.

Apply Definition 1408.4. It places the Real-facing row behind the bounded monotone witness and convergence seal packets, so the nonescape obligation is a finite seal route rather than a compactness theorem or a choice-selected subsequence. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorMonotoneSubsequenceObligati`

Theorem 1388.11 (Bolzano–Weierstrass selector window-choice-free carrier)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the selected-window carrier is choice-free: each* $\mathsf{StreamName}^{\uparrow}$ *window in* W *is read from a displayed selector entry in* M *and an inspected bounded source window in* B*, then transported by the componentwise* hsame *row* H *and replayed by the displayed* Cont *row* C*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *are downstream rows of the same finite packet, not witnesses to a countable-choice selector, sequential compactness, or an ambient Bolzano–Weierstrass theorem.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorWindowChoiceFreeCarrier`

Proof. Induct over the finite selector-window portion of the displayed BHist packet. The empty window reads no selector entry and therefore exposes no hidden subsequence object.

At a successor step, project the next entry of M together with its source window in B and inspected threshold in Q. The entry appends the corresponding $\mathsf{StreamName}^{\uparrow}$ row to W through H and C, so the carrier step is a finite replay rather than a choice of a new subsequence.

After the selected window is displayed, the dyadic and regular rows D, R and the $\mathsf{Real}^{\uparrow}$ seal row E remain attached to the same packet with provenance P and local naming N. Thus the carrier exports only the finite window route and none of the excluded compactness or choice principles. □

Theorem 1388.12 (Bolzano–Weierstrass selector finite-subsequence obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, a finite-subsequence read must display the bounded source row* B*, selector ledger* M*, inspected-threshold row* Q*, selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *before the read is available to a consumer. The obligations are finite window admission, selector-order replay, regular-rational readback, and Real-seal nonescape; they do not assert sequential compactness, countable choice, quotient-stream equality, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the displayed selector packet. The rows B, M, Q, W determine the finite subsequence surface, and every selected window is admitted by the packet row that names it.

Run finite window induction over M and Q. Each induction step consumes one displayed selector entry, reads the corresponding bounded source window from B, and appends exactly the matching $\mathsf{StreamName}^{\uparrow}$ row to W by hsame transport and Cont replay.

The readback and seal obligations are then projected from R, E, P, N. Since these rows are attached after the selected windows and contain no coordinate for compactness, countable choice, quotient equality, host equality, or ambient completeness, the finite-subsequence obligations stop at the displayed packet boundary. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteSubsequenceObligatio`

Theorem 1388.13 (Bolzano–Weierstrass selector root window)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, every root selected-window read must expose the bounded source windows* B*, monotone selector ledger* M*, cofinal row* Q*,* $\mathsf{StreamName}^{\uparrow}$ *row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural rows* H, C, P, N*. The read factors through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergence}$ $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and it exports no full Bolzano–Weierstrass theorem, sequential compactness, choice-selected subsequence, quotient stream, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRootWindow`

Proof. Begin with the subsequence handoff of Definition 1408.2. It routes the root selector surface from B, M, Q to W, D, R through the finite monotone subsequence packet and the stream, dyadic, and regular rational rows.

Next use the finite-subsequence obligations of Definition 1408.12. They require the same bounded source, selector, cofinal, selected-window, readback, seal, and structural rows to be displayed before any finite selected-window read is exported.

Now apply the bounded-monotone route of Definition 1408.4. It places the Real seal behind $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ and $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, so the root read remains a finite induction-style handoff rather than a compactness principle.

Finally use the nonescape theorem Definition 1408.9. Since the carrier has no row for full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient streams, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the root window exports only the displayed finite route. □

Theorem 1388.14 (Bolzano–Weierstrass selector RegSeqRat readback nonescape)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from the selected windows must pass through the finite route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R$$

with the $\mathsf{Real}^{\uparrow}$ *seal row* E *available only after that readback is displayed. Componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row preserve the route; they do not let the read escape to a choice-selected subsequence, sequential compactness, quotient-stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegSeqRatReadbackNonescape`

Proof. Start with the finite-subsequence obligations of Definition 1408.12. They force every selected window in W to be displayed through B, M, Q before downstream readback can begin.

Project the dyadic and regular rows from the carrier. The $\mathsf{DyadicRatCore}^{\uparrow}$ row D is attached to the displayed selected windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is read only after those dyadic rows have been exposed.

Finally inspect the structural rows H, C, P, N and the $\mathsf{Real}^{\uparrow}$ seal row E. They replay and name the same finite route, so no hidden compactness, choice, quotient, host-equality, or ambient-completeness datum can be introduced at readback time. □

Definition 1388.15 (Bolzano–Weierstrass selector induction budget)**.** *The finite selector-induction budget of an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$ *is the displayed four-row ledger* B, M, Q, W*: the bounded source-window row* B*, the monotone selector ledger* M*, the inspected-threshold cofinality row* Q*, and the selected* $\mathsf{StreamName}^{\uparrow}$ *window row* W*. A budget step consumes one displayed entry of* M*, reads its bounded source window from* B*, verifies the inspected threshold in* Q*, and appends the admitted selected window to* W *through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*. The budget contains no hidden subsequence object, least-selector functional, countable choice row, or compactness row.*

Lean4 (definition) `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorInductionBudget`
Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorInductionBudget`
`selected window route`

Theorem 1388.16 (Bolzano–Weierstrass selector finite monotone induction)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, finite induction over the selector-induction budget of Definition 1408.15 admits each selected window through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*. The base stage exposes the empty selected-window family over the displayed bounded source row* B*; each successor stage consumes one ledger entry from* M*, checks the inspected threshold row* Q*, and appends exactly the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window to* W *by displayed* hsame *transport and* Cont *replay. No stage exports a choice-selected subsequence, sequential compactness, quotient-stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteMonotoneInduction`

Proof. Project the budget rows B, M, Q, W from the carrier. These rows are finite BHist data, so induction is over the displayed selector ledger rather than over an external sequence or function space.

The base case reads no selector entry. It exposes the empty selected-window family and records that no dyadic, regular, Real-facing, or compactness row is available.

For the successor step, consume the next displayed selector entry in M. The entry names its bounded source window in B, its inspected threshold in Q, and the selected $\mathsf{StreamName}^{\uparrow}$ window that is appended to W through the $\mathsf{MonotoneSubsequence}^{\uparrow}$ comparison row.

The transport rows H and replay rows C preserve the admitted window at each finite step. Since no carrier coordinate names a choice functional or a sequential compactness theorem, the induction admits exactly the selected windows listed in W. □

Theorem 1388.17 (Bolzano–Weierstrass selector finite-window stabilization)**.** *Let* $T = (B, M, Q, W, D, R, E, H, C$ *be an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet. If a selected finite* $\mathsf{StreamName}^{\uparrow}$ *window in* W *is replaced by a later displayed window admitted by the same monotone selector ledger* M *and inspected-threshold row* Q*, then the route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

is preserved through $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}$ $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The stabilized packet records only this finite-window replacement route; it does not export full Bolzano–Weierstrass, sequential compactness, countable choice, quotient streams, host equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteWindowStabilization`

Proof. Project $B, M, Q, W, D, R, E, H, C, P, N$ from the carrier. The replacement hypothesis uses the same M and Q rows, so it remains inside the selector-induction budget of Definition 1408.15.

Apply Definition 1408.16 to the displayed later window. Finite induction over B, M, Q, W admits that window through the same $\mathsf{MonotoneSubsequence}^{\uparrow}$ comparison row, and H, C replay the transport without adding a subsequence object or a choice functional.

Now use the cluster and regular-Cauchy seal routes already proved in Definition 1408.8 and Definition 1408.19. The dyadic rows D, regular handoff R, and Real seal E are consumed after the stabilized selected window, so the downstream route is still the finite chain through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

No carrier coordinate remains for full Bolzano–Weierstrass compactness, sequential compactness, countable choice, quotient-stream equality, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The theorem therefore stabilizes only the displayed finite window. □

Theorem 1388.18 (Bolzano–Weierstrass selector finite nested-window induction)**.** *Let $T = (B, M, Q, W, D, R, E,$ be an accepted $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet. Suppose the bounded monotone route of Definition 1408.4, the nested-interval obligation of Definition 1408.22, the dyadic-window obligation of Definition 1408.21, the $\mathsf{StreamName}^{\uparrow}$ selected-window row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal row E are all displayed in the same BHist packet. Then the selector admits finite induction over the nested windows: each induction stage refines the current interval through the displayed nested-interval row, transports the matching dyadic tolerance through D, replays the same $\mathsf{StreamName}^{\uparrow}$ window through W, and hands the resulting regular readback to R before E can be inspected. The cluster handoff is therefore the same handoff as in Definition 1408.8, and no stage exports a selected infinite subsequence, countable choice, quotient-stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.*

Proof. Project the finite-window stabilization rows from Definition 1408.17. They keep the route

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

inside the accepted packet and leave H, C, P, N as structural transport, replay, provenance, and local naming rows.

Induct on the displayed nested-window budget. The base stage is the current selected window in W together with its bounded monotone route, dyadic tolerance row, regular readback row, and Real seal row.

For the successor step, use Definition 1408.22 to refine the interval row and Definition 1408.21 to transport the matching dyadic tolerance through D. The selected $\mathsf{StreamName}^{\uparrow}$ row remains the displayed window row W, so the successor step has no independent subsequence selector.

Now replay the readback through R and the Real seal through E. The cluster route of Definition 1408.8 and the nonescape boundary of Definition 1408.9 close the handoff through the same rows. Since no carrier coordinate names a selected infinite subsequence, countable-choice schedule, quotient-stream equality, host real equality, or ambient completeness theorem, the finite induction preserves only the displayed cluster handoff. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteNestedWindowInductio`

Theorem 1388.19 (Bolzano–Weierstrass selector regular Cauchy cluster seal)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the finite monotone route of Definition 1408.16 reaches the cluster-facing seal only through*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The rows D, R, E *are read respectively through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the bounded monotone convergence seal route. The resulting consumer receives a regular Cauchy cluster boundary for the displayed selected windows, not classical sequential compactness, arbitrary choice, host real equality, quotient-stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyClusterSeal`

Proof. First apply Definition 1408.16. It admits the selected windows in W by finite induction over B, M, Q, W and makes $\mathsf{MonotoneSubsequence}^{\uparrow}$ the only selector surface.

Next read the dyadic tolerance ledger D and regular handoff row R. The carrier definition attaches each row in D to a selected window in W, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is available only after those dyadic rows are displayed.

Finally use the bounded monotone Cauchy witness and bounded monotone convergence seal routes named in Definition 1408.4. They expose the Real-facing row E after W, D, R and do not add compactness, choice, quotient-stream equality, host equality, or ambient completeness. □

Theorem 1388.20 (Bolzano–Weierstrass selector bounded-subsequence carrier)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the bounded-subsequence carrier surface exposes bounded source windows and monotone selected windows by finite selector induction over* B, M, Q, W *only. The induction may read the bounded window ledger* B*, the monotone selector ledger* M*, the inspected-threshold row* Q*, and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, but it exports no ambient subsequence object, no countable choice functional, and no sequential compactness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorBoundedSubsequenceCarrier`

Proof. Project the carrier rows from Definition 1408.1. The rows B, M, Q, W are the only displayed selector rows before dyadic, regular, Real-facing, and structural rows are consumed.

Induct over the finite selector ledger M together with the inspected-threshold row Q. The base case reads the initial bounded source window from B and exposes the empty selected window family. The successor step consumes one displayed selector entry, verifies its comparison row through $\mathsf{MonotoneSubsequence}^{\uparrow}$, and appends the corresponding selected $\mathsf{StreamName}^{\uparrow}$ window to W.

Transport and continuation rows preserve the same finite route. Since the carrier has no row for an ambient subsequence, choice functional, or compactness theorem, the bounded-subsequence surface is exactly the finite induction over B, M, Q, W. □

Theorem 1388.21 (Bolzano–Weierstrass selector dyadic-window obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, selected windows reach dyadic tolerance rows only through the displayed route*

$$B, M, Q \longmapsto W \longmapsto D.$$

Thus every dyadic read is mediated by $\mathsf{StreamName}^{\uparrow}$ *selected windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, not by a host metric limit, quotient-stream equality, or an uninspected source window.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorDyadicWindowObligation`

Proof. Begin with Definition 1408.20. It supplies the finite selected-window route $B, M, Q \longmapsto W$ and excludes any hidden subsequence object.

Now inspect the dyadic ledger D in Definition 1408.1. Each tolerance row in D is attached to a displayed selected $\mathsf{StreamName}^{\uparrow}$ window in W, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance data before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read is available.

The componentwise hsame and Cont rows replay this displayed handoff. They cannot add quotient-stream equality, host metric limits, or uninspected source windows, so every dyadic tolerance read factors through W and D. □

Theorem 1388.22 (Bolzano–Weierstrass selector nested-interval obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any consumer that reads the selected cofinal windows as a regular rational or Real-facing limit must first route those windows through the sibling* $\mathsf{NestedInterval}^{\uparrow}$ *and* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ *rows. In particular, the displayed route is*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto \mathsf{NestedInterval}^{\uparrow}, \mathsf{NestedIntervalCompleteness}^{\uparrow} \longmapsto R \longmapsto E.$$

The route consumes bounded monotone and dyadic window data before the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals are read, and it exports no interval-compactness theorem, selected endpoint, countable choice schedule, quotient stream, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorNestedIntervalObligation`

Proof. Begin with Definition 1408.20 and Definition 1408.21. They expose the finite selected-window route from B, M, Q to W and then to the dyadic tolerance ledger D.

Next read the sibling nested-interval surfaces. The nested interval chapter supplies the retained-window order row, while the nested interval completeness chapter packages the decreasing-window read before any regular rational name is available. The selector packet can therefore consume R only after the selected dyadic windows have been recognized as nested retained windows.

Finally apply Definition 1408.23. It places the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the Real seal behind the same selected-window route, while the bounded-window exhaustion and regular Cauchy cluster seal keep the consumer inside the displayed packet. Thus no hidden compactness, choice, quotient, host-equality, or ambient-completeness datum enters the selector obligation. □

Theorem 1388.23 (Bolzano–Weierstrass selector regular Cauchy limit handoff)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real-facing row* E *is available only after the selected dyadic windows have passed through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *and the bounded monotone convergence seal route. The packet therefore hands a regular Cauchy limit boundary to* $\mathsf{Real}^{\uparrow}$*, not classical Bolzano–Weierstrass, a choice-selected subsequence, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyLimitHandoff`

Proof. Use Definition 1408.21 to expose B, M, Q, W, D as the only dyadic-window route. No Real-facing row is visible at this stage.

Read the regular handoff row R. By the carrier definition, R is attached to the selected dyadic windows and is preserved by the displayed hsame and Cont rows before the seal coordinate E can be consumed.

Finally apply Definition 1408.4. The bounded monotone Cauchy witness and bounded monotone convergence seal chapters place E behind the finite selector, stream, dyadic, and regular handoff route, so the exported object is a regular Cauchy limit boundary for the displayed packet rather than a compactness or completeness theorem. □

Theorem 1388.24 (Bolzano–Weierstrass selector cofinal window stability)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the inspected cofinal rows in* Q *remain tied to the displayed bounded source windows* B *and selected windows* W *under componentwise* hsame *transport and displayed* Cont *replay. A consumer may refine an inspected threshold only by replaying the same* B, M, Q, W *window route; it cannot replace the cofinal row by a choice-selected subsequence, an uninspected source window, or quotient-stream equality.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorCofinalWindowStability`

Proof. Project the carrier of Definition 1408.1. The cofinal row Q is stored beside the bounded source window row B, the monotone selector ledger M, and the selected $\mathsf{StreamName}^{\uparrow}$ windows W.

Apply Definition 1408.20. Its finite induction over B, M, Q, W makes every inspected threshold part of the displayed selector ledger rather than an external subsequence object.

Now use the dyadic-window and regular-Cauchy handoffs, Definition 1408.21 and Definition 1408.23. They keep each downstream read attached to the same selected windows before D, R, E, H, C, P, N are consumed. Thus hsame transport and Cont replay preserve the cofinal window boundary instead of adding a hidden selector. □

Theorem 1388.25 (Bolzano–Weierstrass selector Real seal finality)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real seal row* E *is terminal after the displayed route through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*. Once* E *is read through that route, no further row may reinterpret the packet as full Bolzano–Weierstrass compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, or a choice-selected cluster point.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRealSealFinality`

Proof. Start with the bounded-monotone route of Definition 1408.4. It orders the Real-facing read through the monotone subsequence packet, the bounded monotone Cauchy witness, the convergence seal, the stream windows, the dyadic tolerance rows, the regular rational handoff, and finally the Real seal.

Use Definition 1408.23. The selected dyadic windows reach E only after the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the bounded monotone convergence route have been displayed.

Finally apply Definition 1408.9. The carrier contains no coordinate after E for compactness, host equality, choice, quotient streams, or ambient completeness. The Real seal is therefore terminal for the finite L10 selector route. □

Theorem 1388.26 (Bolzano–Weierstrass selector bounded-window exhaustion)**.** *Every bounded-window consumer of an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet must expose the rows*

$$B, M, Q, W, D, R, E, H, C, P, N$$

before it may read a selected-window, dyadic-window, regular-Cauchy, or Real-facing conclusion. The exposed rows exhaust the bounded-window surface: no consumer may cite a hidden subsequence object, an ambient compactness theorem, a least cluster selector, quotient-stream equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorBoundedWindowExhaustion`

Proof. Begin with Definition 1408.5. It already binds $B, M, Q, W, D, R, E, H, C, P, N$ as one displayed packet for every finite selected-window read.

Apply Definition 1408.20. The bounded source and selected-window surface is exhausted by the finite induction over B, M, Q, W.

Next use Definition 1408.21 and Definition 1408.23. These theorems force the downstream dyadic and regular-Cauchy reads through D and R before the Real seal E is visible.

The structural rows H, C, P, N transport, replay, source, and name the same carrier. Since the nonescape theorem Definition 1408.9 refuses all stronger endpoints, the displayed row family exhausts the bounded-window consumer surface. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ Real-facing read factors through $\mathsf{MonotoneSubsequence}$ $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, and $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ before the $\mathsf{Real}^{\uparrow}$ seal is visible.

Independence witness: monotonesubsequence, boundedmonotonecauchywitness, boundedmonotoneconvergenceseal, streamname, regseqrat, dyadicratcore, real: $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded sequence windows, a monotone selector ledger, inspected-threshold cofinal evidence, selected stream windows, dyadic tolerance rows, a regular rational handoff, a Real seal boundary, transports, continuations, provenance, and local naming.

Theorem 1388.27 (Bolzano–Weierstrass selector scoped L10 kernel)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelecto}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the carrier, bounded-window exhaustion, monotone-subsequence route, nested-interval obligation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal finality depend only on* BHist*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows displayed in the packet and its cited L10 siblings. No proof step requires a host compactness theorem, a choice-selected subsequence, quotient-stream equality, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorScopeClosure`

Proof. Project the carrier of Definition 1408.1. The rows $B, M, Q, W, D, R, E, H, C, P, N$ are finite BHist data, with componentwise hsame transport and displayed Cont replay as the only structural routes.

Apply the finite-window induction of Definition 1408.6. It binds the monotone-subsequence route to displayed selector entries and selected $\mathsf{StreamName}^{\uparrow}$ windows, not to an external selector functional.

Use the nested-interval obligation of Definition 1408.22. The retained-window route is consumed as sibling L10 data before any regular rational readback, so the proof remains inside the displayed finite window surface.

Finally apply the regular Cauchy limit handoff and Real seal finality, Definition 1408.23 and Definition 1408.25. They place the $\mathsf{RegSeqRat}^{\uparrow}$ row and the $\mathsf{Real}^{\uparrow}$ seal after $\mathsf{DyadicRatCore}^{\uparrow}$ and the selected $\mathsf{StreamName}^{\uparrow}$ windows. Since the nonescape theorem excludes compactness, choice, quotient, host equality, and ambient completeness rows, the stated L10 kernel scope is closed. □

Theorem 1388.28 (Bolzano–Weierstrass selector public export)**.** *An accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet exports exactly the finite handoff from bounded sequence windows through monotone-subsequence*

selection, bounded monotone Cauchy witness, bounded monotone convergence seal, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows. The exported public surface is the row family*

$$B, M, Q, W, D, R, E, H, C, P, N,$$

read in that dependency order and no stronger order. It does not claim sequential compactness, countable choice, quotient-stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorPublicExport`

Proof. Begin with the selector handoff of Definition 1408.2 and the real seal boundary of Definition 1408.3. These place the selected subsequence surface and the Real-facing seal behind displayed packet rows rather than behind a compactness theorem.

Next apply the bounded-monotone route, finite-window stabilization, and regular Cauchy cluster seal through Definition 1408.4, Definition 1408.6, and Definition 1408.23. The route forces the monotone-subsequence selection, bounded monotone Cauchy witness, bounded monotone convergence seal, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff to appear before E can be consumed.

Finally use the scoped L10 kernel theorem, Definition 1408.27. It exhausts the public export by $B, M, Q, W, D, R, E, H, C, P, N$ and refuses any hidden compactness, choice, quotient, host-equality, or ambient-completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet is generated from bounded source windows, a monotone selector ledger, inspected-threshold cofinal evidence, selected StreamName windows, DyadicRatCore tolerance rows, a RegSeqRat handoff, a RealUp seal boundary, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1408.1, Definition 1408.2, Definition 1408.3, Definition 1408.4, Definition 1408.5, Definition 1408.6, Definition 1408.22, Definition 1408.23, Definition 1408.26, Definition 1408.27, and Definition 1408.28 bind the carrier, subsequence handoff, Real seal boundary, bounded-monotone route, window obligation, finite-window induction, nested-interval obligation, regular Cauchy limit handoff, bounded-window exhaustion, scoped L10 kernel, and public finite handoff surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim full Bolzano-Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite selector stability, inspected-threshold cofinal coverage, selected-window RegSeqRat handoff, bounded monotone witness routing, and Real seal boundary non-escape for BEDC.Derived.BolzanoWeierstrassSelectorUp.

Chapter 1389

A Concrete Naming Certificate for $\mathsf{RationalDensityReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RationalDensityReal}^{\uparrow}$ packet records the finite rational-witness interface for displayed $\mathsf{Real}^{\uparrow}$ intervals. It sits after the Real seal of chapter 38, the regular-rational readback of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance arithmetic of chapter 291. It routes located interval data through the rational interval surface of chapter 354 and the apartness-gap boundary of chapter 304, without reading a host real order.

Definition 1389.1 (RationalDensityReal carrier)**.** *A* $\mathsf{RationalDensityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, W_L, W_U, A, I, Q, T, R, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint names, each carrying a* $\mathsf{Real}^{\uparrow}$ *seal row and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected from the two endpoint names.*
3. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-gap row displayed between the endpoint seals.*
4. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *row whose rational bounds are read from the inspected endpoint windows.*
5. *Q is the visible* $\mathsf{Rat}^{\uparrow}$ *witness row selected between the displayed rational interval bounds.*
6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row certifying that the inspected endpoint windows and the rational witness are read at one dyadic scale.*
7. *R is the dyadic radius row that records the positive subinterval around Q still contained in the displayed interval.*
8. *H stores componentwise* hsame *transports for the endpoint, window, apartness, interval, witness, tolerance, and radius rows.*
9. *C stores the* Cont *replay from endpoint windows to the rational interval row, from the rational interval row to the witness row, and from the witness row to the radius row.*
10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^\uparrow}$ *row.*

The carrier stores a displayed rational witness for a displayed interval. It has no host order coordinate, no trichotomy row, no quotient equality of real names, no completeness principle, no Heine–Borel cover row, and no hidden choice selector.

Theorem 1389.2 (RationalDensityReal NameCert obligation surface)**.** *For an accepted* $\mathsf{RationalDensityReal}^\uparrow$ *carrier* $D = (L, U, W_L, W_U, A, I, Q, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^\uparrow}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed endpoint, window, apartness, rational-interval, witness, tolerance, radius, transport, continuation, provenance, and naming rows;*
2. *classifier transport along the stored* hsame *rows and the displayed* Cont *routes;*
3. *witness visibility, stating that the* $\mathsf{Rat}^\uparrow$ *row Q is exposed before any consumer reads the radius row R;*
4. $\mathsf{Real}^\uparrow$ *seal nonescape, stating that endpoint seals are consumed only through the stored* $\mathsf{RegSeqRat}^\uparrow$ *readbacks and* $\mathsf{StreamName}^\uparrow$ *windows;*
5. *ledger exactness, stating that the packet exports no host real order, trichotomy, quotient equality, completeness theorem, Heine–Borel cover, or hidden selector.*

Lean4 ✓ `BEDC.Derived.RationalDensityRealUp.RationalDensityRealTasteGate single carrier alignment`

Proof. Carrier admission is the tuple in Definition 1409.1. The endpoint names, stream windows, apartness row, rational interval row, witness row, dyadic tolerance, dyadic radius, transports, continuations, package row, and local naming row are all finite BHist coordinates.

Classifier transport reads H and C componentwise. The transported endpoint windows still feed the same rational-interval row shape, and the transported interval still exposes the same witness-and-radius sequence.

Witness visibility follows from the carrier order: the row Q is stored before the radius row R, and the continuation row records the replay from I to Q before replaying from Q to R.

The seal nonescape and ledger exactness are inventory checks. The endpoint seals occur only beside their regular-rational readbacks and finite windows, while host order, trichotomy, quotient equality, completeness, Heine–Borel data, and hidden selection are not rows of the packet. □

Theorem 1389.3 (RationalDensityReal located-window witness)**.** *Let L and U be displayed* $\mathsf{Real}^\uparrow$ *endpoint names with* $\mathsf{RegSeqRat}^\uparrow$ *readbacks, finite* $\mathsf{StreamName}^\uparrow$ *windows, and an* $\mathsf{ApartnessReal}^\uparrow$ *positive-gap row between them. If the inspected windows determine a displayed* $\mathsf{RationalInterval}^\uparrow$ *row whose dyadic tolerance is recorded by* $\mathsf{DyadicRatCore}^\uparrow$*, then an accepted* $\mathsf{RationalDensityReal}^\uparrow$ *packet exposes a visible* $\mathsf{Rat}^\uparrow$ *witness row inside that displayed interval. Every consumer route to the witness factors through*

$$(L, U) \longmapsto (W_L, W_U) \longmapsto A \longmapsto I \longmapsto Q$$

and through the dyadic tolerance and radius rows T, R; no step appeals to host real order, trichotomy, quotient equality, Real completeness, or Heine–Borel compactness.

Proof. Read the endpoint data through their displayed regular-rational rows. The finite stream windows W_L, W_U are the only observations used to form the rational interval row, so the route begins inside $\mathsf{RegSeqRat}^\uparrow$ and $\mathsf{StreamName}^\uparrow$.

The positive-gap row A makes the interval located at the displayed scale. The dyadic row T records that the endpoint windows and the rational interval bounds are compared inside $\mathsf{DyadicRatCore}^{\uparrow}$, not inside an ambient order on completed reals.

The rational witness is then the row Q of the carrier. The continuation path reads I before Q and records the radius row R only after Q is visible.

Thus the witness route is exactly the displayed chain through $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$. Any route using host real order, trichotomy, quotient equality, Real completeness, or Heine–Borel compactness leaves the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalDensityReal}^{\uparrow}$ packet is generated from two displayed RealUp endpoint names, their RegSeqRat readbacks, finite StreamName windows, an ApartnessReal positive-gap row, a RationalInterval row, a visible Rat witness, dyadic tolerance and radius rows, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1409.1, Definition 1409.2, and Definition 1409.3 declare the finite rational-witness surface for displayed RealUp intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, trichotomy, quotient equality of real names, host real order, Heine–Borel compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, witness visibility, seal nonescape, and ledger exactness for the same finite packet.

Chapter 1390

A Concrete Naming Certificate for $\mathsf{CauchySequenceEquivalence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ packet records the concrete equivalence relation on regular Cauchy names as finite BEDC data. It sits between the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The equivalence packet is downstream of chapter 3413 because the regular-name comparison only exists after the Cauchy source windows have been licensed. The packet compares two regular Cauchy names through displayed finite windows and tolerances before any real-seal consumer can read the result.

Definition 1390.1 (CauchySequenceEquivalence carrier surface)**.** *A* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, S, T, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name rows;*
2. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the paired windows;*
4. *E is the local finite equivalence classifier recording that every scheduled paired observation is within the displayed dyadic tolerance;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal-comparison row that may be consumed only after X, Y, S, T, D, E have been displayed;*
6. *H is the componentwise* hsame *provenance for the two regular names, the paired stream windows, the dyadic ledger, the classifier, and the seal-comparison row;*
7. *C is the displayed* Cont *route by which a consumer replays those finite windows without adding host data;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$ *row.*

The carrier exposes no quotient stream, selected limit, host real equality, countable-choice row, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceTasteGate single carrier alignment`

Theorem 1390.2 (CauchySequenceEquivalence NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$ *obligation surface consists exactly of the finite rows in Definition 1412.1. Carrier admission requires two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *names, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a local finite equivalence classifier, a* $\mathsf{Real}^{\uparrow}$ *seal-comparison row, componentwise* hsame *provenance, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier stability is tested only by the paired finite windows and the stored dyadic ledger. Transport preserves the same rows through H and C. Tolerance exactness says that every accepted comparison reads exactly the dyadic ledger D. Ledger visibility says that every consumer can inspect $X, Y, S, T, D, E, R, H, C, P, N$ and no other equality source.*

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceNamecertObligations`

Theorem 1390.3 (CauchySequenceEquivalence RegSeqRat Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal comparison routed through an accepted* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *packet factors through the displayed sequence*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, a consumer first reads the two regular Cauchy name rows X, Y, then the paired finite stream windows S, T, then the dyadic tolerance ledger D and finite classifier E, and only then reaches the seal-comparison row R. The packet supplies no quotient stream, selected limit, host real equality, countable choice, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier regseqrat real handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ packet is generated from two displayed $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, paired $\mathsf{StreamName}^{\uparrow}$ finite windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite equivalence classifier, a $\mathsf{Real}^{\uparrow}$ seal-comparison row, displayed hsame provenance, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1412.1, Definition 1412.2, and Definition 1412.3 display the bridge to the public finite handoff over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier\_regseqrat\_real\_handoff`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, countable choice, host real equality, or a standard bridge.

Upgrade path: Bridge checking requires a standard bridge theorem for the displayed RegSeqRat and Real handoff surface.

Chapter 1391

A Concrete Naming Certificate for $\mathsf{BoundedRealSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BoundedRealSequence}^{\uparrow}$ packet records a bounded real sequence by finite stream-window rows, regular rational readback, a real-name seal, and a displayed closed-bounded interval net. It sits before the subsequence compactness route of chapter 1254 and exposes the boundedness surface that chapter consumes. Its dependency surface is the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the interval-net bound row of chapter 1250. The packet is a finite certificate for bounded sequence inspection; it is not open-cover compactness, trichotomy of reals, host completeness, quotient equality of streams, or a choice-selected subsequence.

Definition 1391.1 (Bounded real sequence carrier)**.** *A* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, Q, R, I, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source schedule of sequence indices under inspection.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows exposed for the inspected indices in S.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and their dyadic tolerance observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *sequence-name seal reached only after W and Q have been displayed.*
5. *I is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row carrying the interval net used to witness boundedness for the displayed sequence windows.*
6. *H stores componentwise* hsame *transport for the source schedule, stream windows, regular readback, real seal, interval-net bound, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from inspected indices to stream windows, from stream windows to regular readback, from regular readback to the real seal, and from the real seal to the interval-net bound assertion.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalN}$ $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row.*

The bound row is displayed as the closed interval-net containment surface

$$I : \mathsf{ClosedBoundedIntervalNet}^{\uparrow},$$
$$\forall i \in S_{\text{shown}} \quad W_i \longmapsto Q_i \longmapsto R_i \in I.$$

Here S_{shown} denotes only the inspected finite index rows of the carrier. The classifier compares accepted packets only through $S, W, Q, R, I, H, C, P, N$ and has no coordinate for trichotomy, host real order completeness, quotient-stream equality, open-cover compactness, or a selected infinite subsequence.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceTasteGate single carrier alignment`

Theorem 1391.2 (Bounded real sequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier $B = (S, W, Q, R, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, finite source-window coverage through S and W, regular rational readback through Q, real-name sealing through R, closed-bounded interval-net containment through I, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceCarrier semantic name certificate`

Proof. Project the carrier of Definition 1415.1. The only sequence data available to the classifier are the inspected index schedule S, the corresponding $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the downstream $\mathsf{Real}^{\uparrow}$ seal R.

Read the bound row next. The $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet I supplies the finite interval-net containment surface from chapter 1250; boundedness is therefore a displayed row tied to the inspected windows, not an ambient completeness or order-trichotomy principle.

Finally inspect the structural rows. The rows H, C, P, N transport, replay, source, and name only the displayed sequence-window, regular-readback, real-seal, and interval-net rows. Since no other coordinate is present in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 1391.3 (Bounded real sequence finite-window bound)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier preserves boundedness under finite window inspection: any displayed route from an inspected index to a* $\mathsf{Real}^{\uparrow}$ *sequence-name seal factors through*

$$S \longmapsto W \longmapsto Q \longmapsto R \longmapsto I$$

together with H, C, P, N. Thus a finite-window consumer can read the closed-bounded interval-net bound without importing trichotomy, host completeness, quotient-stream equality, open-cover compactness, or a selected subsequence.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceFiniteWindow bound`

Proof. Begin with an inspected index row in S. The continuation row C exposes its $\mathsf{StreamName}^{\uparrow}$ window in W, then the regular rational readback Q, and only then the $\mathsf{Real}^{\uparrow}$ seal R.

The bound row I is read after this route. It is the finite interval-net containment row supplied by chapter 1250, so the bound is attached to the displayed windows and readbacks rather than to a host-level set of all sequence values.

Transport through H and provenance through P preserve the same route, while N names it locally. Because the carrier has no coordinate for trichotomy, host completeness, quotient equality, open-cover compactness, or selected subsequences, finite-window boundedness cannot leave the displayed path. □

Theorem 1391.4 (Bounded real sequence Bolzano–Weierstrass handoff)**.** *The* BolzanoWeierstrass$^{\uparrow}$ *consumer of chapter 1254 reads a bounded sequence source from an accepted* BoundedRealSequence$^{\uparrow}$ *carrier only through the finite route*

$$S, W, Q, R, I, H, C, P, N.$$

Consequently the handoff supplies a reusable bounded real sequence certificate for the L10 real-completion route while leaving open-cover compactness, trichotomy, host completeness, quotient-stream equality, and choice-selected subsequences outside the exported surface.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceBolzanoWeierstrass handoff`

Proof. Use Definition 1415.3. It places every inspected sequence value on the route from S through W, Q, R to the interval-net bound row I.

Now read the BolzanoWeierstrass$^{\uparrow}$ carrier of chapter 1254. Its bounded sequence row is exactly the source surface needed before finite interval nets, regular Cauchy subsequence extraction, RegSeqRat readback, and the Real cluster seal can be consumed.

The structural rows H, C, P, N preserve the finite route across the handoff. Thus the downstream consumer receives a bounded real sequence certificate and receives no additional compactness, trichotomy, host completeness, quotient-stream equality, or arbitrary subsequence selection principle. □

Theorem 1391.5 (Bounded real sequence forward Bolzano route)**.** *Every* BolzanoWeierstrassRate$^{\uparrow}$ *consumer reading an accepted* BoundedRealSequence$^{\uparrow}$ *carrier must first pass through finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, the* ClosedBoundedIntervalNet$^{\uparrow}$ *bound row, and the terminal* Real$^{\uparrow}$ *seal named by the source packet. Only after those rows are exposed may the consumer inspect the finite-window rate route of chapter 8827.*

Proof. Start with Definition 1415.4. It exposes the bounded source through $S, W, Q, R, I, H, C, P, N$ before any Bolzano–Weierstrass-facing consumer reads the sequence data.

The rate consumer narrows that same source route. Its finite windows are the StreamName$^{\uparrow}$ row W, its rational readback is Q, its boundedness witness is the ClosedBoundedIntervalNet$^{\uparrow}$ row I, and its terminal real-name data is the Real$^{\uparrow}$ seal row R.

Consequently the BolzanoWeierstrassRate$^{\uparrow}$ packet may inspect only the displayed finite source rows. It receives no selected subsequence, countable choice schedule, quotient stream equality, trichotomy decision, host compactness theorem, or ambient Real$^{\uparrow}$ completeness claim from the bounded-sequence source. □

Falsifiable prediction: Every BolzanoWeierstrass$^{\uparrow}$ bounded-source read factors through a BoundedRealSequence$^{\uparrow}$ route displaying finite stream windows, regular readback, a RealUp seal, and a closed-bounded interval-net row.

Independence witness: streamname, regseqrat, real, closedboundedintervalnet, bolzanoweierstrass: BoundedRealSequence$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite sequence schedule, stream windows, regular rational readback, real seals, interval-net boundedness, transport, continuation, provenance, and local naming.

Theorem 1391.6 (Bounded real sequence window envelope)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier exposes a bounded sequence to downstream consumers only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound ledger attached to those windows. In particular, any consumer that reads an inspected index must factor through the visible route*

$$S, W, Q, R, I, H, C, P, N$$

and cannot extract an infinite stream, quotient sequence equality, host supremum, open-cover compactness proof, or selected subsequence from the carrier.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceWindow envelope`

Proof. Project the carrier of Definition 1415.1. The inspected source schedule S and finite windows W are the only entry rows for sequence data, so no downstream read starts from an unbounded stream.

The bound is then read through the displayed route to I. The regular readback Q and real seal R connect each visible window to the closed-bounded interval-net row, while the dyadic ledger named by the sibling tolerance surface records the finite comparison data needed for the bound.

Rows H, C, P, N transport, replay, source, and name the same route. Since the absent rows are not part of the carrier, the envelope remains a finite-window and finite-ledger surface. □

Theorem 1391.7 (Bounded real sequence Bolzano–Weierstrass frontier)**.** *Any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$*-facing consumer that receives data from an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier receives only the bounded-window frontier*

$$S, W, Q, R, I, H, C, P, N.$$

The frontier supplies finite bounded real sequence inspection data before the Bolzano–Weierstrass route opens, but it does not provide a selected subsequence, a compactness principle, trichotomy, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceBolzano frontier`

Proof. Use the handoff route of Definition 1415.4. That theorem identifies the bounded sequence source read by the Bolzano–Weierstrass consumer with the same finite carrier rows.

The frontier is bounded because I is displayed after S, W, Q, R. Thus the consumer can inspect finite sequence windows, read them back through $\mathsf{RegSeqRat}^{\uparrow}$, and see the real-seal and interval-net bound rows before any compactness-facing construction is considered.

No selected subsequence or compactness proof is stored in those rows. The frontier therefore hands off bounded-window data only, leaving all choice, quotient, trichotomy, and completeness principles outside the exported surface. □

Theorem 1391.8 (Bounded real sequence RegSeqRat readback)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, each* $\mathsf{RegSeqRat}^{\uparrow}$ *readback available to a consumer is routed through the same finite bound and real-seal rows:*

$$W \longmapsto Q \longmapsto R \longmapsto I.$$

Consequently a readback consumer cannot detach the regular rational data from the visible $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row, or the local structural rows* H, C, P, N.

Proof. Begin with an exposed window W_i for an inspected index. By the carrier definition, the matching regular rational readback is the corresponding component of Q rather than a host-level sequence object.

The readback continues to the real seal R_i and then to the interval-net bound row I. This is the same visible finite bound route used by Definition 1415.3.

The structural rows keep the readback inside the packet. Transport by H, replay by C, provenance by P, and naming by N prevent the regular rational data from escaping as a quotient stream, selected subsequence, compactness witness, or ambient real-completeness datum. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceRegSeqRat readback`

Theorem 1391.9 (Bounded real sequence located-bound handoff)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet whose interval-net row is read as located lower and upper real bounds factors through the* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ *route of chapter 2183 before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer inspects it. The visible route is*

$$S, W, Q, R, I \longmapsto \mathsf{LocatedBoundedRealSequence}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the closed-bounded interval-net row is narrowed to located lower and upper bound data before the Bolzano–Weierstrass handoff, without exporting compactness, trichotomy, quotient sequence equality, countable choice, or a selected subsequence.

Proof. Project an accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$ from Definition 1415.1. The sequence source is displayed only through inspected indices S, stream windows W, regular rational readback Q, and the downstream $\mathsf{Real}^{\uparrow}$ seal R.

Use Definition 1415.3 and Definition 1415.8. They put every visible sequence value on the route from W through Q, R to the bound row I, so the lower and upper endpoint reads of I are tied to the same finite windows.

When I is read as located lower and upper real bounds, the narrower sibling route is exactly the $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ packet of chapter 2183. The structural rows H, C, P, N preserve that route before Definition 1415.4 lets any $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer inspect the bounded source. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.LocatedBoundHandoff.BoundedRealSequenceLocatedBound handoff`

Theorem 1391.10 (Bounded real sequence located-window choice boundary)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reading an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier must obtain located lower and upper bound windows through the displayed route*

$$S, W, Q, R, I.$$

No such consumer may choose a subsequence, ambient bound, quotient sequence, or host completion witness outside the carrier before the stream windows, regular readback, real seal, and interval-net bound row have been displayed.

Proof. Begin with the finite-window bound of Definition 1415.3. It places each inspected sequence value on the route from S through W, Q, R to the interval-net bound row I.

Use the RegSeqRat readback and located-bound handoff, Definition 1415.8 and Definition 1415.9. They tie the located lower and upper endpoint reads of I to the same finite stream windows and regular rational rows.

Finally apply the obligation-closure package of Definition 1415.12. The package exports only $S, W, Q, R, I, H, C, P, N$, so Bolzano–Weierstrass and real-completion consumers cannot introduce an outside subsequence choice, ambient bound, quotient sequence, or host completion witness. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceLocatedWindowChoice`

Theorem 1391.11 (Bounded real sequence location induction)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

be an accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet whose located-bound row* I*,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-net row,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* R *are displayed. Then every inspected finite prefix admits induction over its window positions: the empty prefix preserves the same located bound row* I*, and each successor read factors through the same route*

$$S, W, Q, R, I, H, C, P, N$$

so the new visible value remains inside the same located interval and real-seal rows. The induction exports no subsequence selection, real trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$. The inspected schedule S and stream windows W are finite, so induction is over the displayed prefix positions rather than over an infinite host stream.

For the empty prefix, read only the already displayed located-bound row I. No sequence value has been inspected, and the same $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ row remains the active bound.

For the successor step, use Definition 1415.3 and Definition 1415.8. The next window read goes through W to Q, then to the $\mathsf{Real}^{\uparrow}$ seal R, and then back to the same interval-net row I.

The located-bound handoff of Definition 1415.9 keeps the lower and upper endpoint reads attached to this route. Transport, replay, provenance, and naming through H, C, P, N preserve the induction invariant, and the carrier has no coordinate for the excluded subsequence, trichotomy, quotient-equality, or completeness principles. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier location induction`

Theorem 1391.12 (Bounded real sequence obligation-closure package)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, the carrier row of Definition 1415.1, the NameCert obligation surface of Definition 1415.2, the finite-window bound of Definition 1415.3, the Bolzano–Weierstrass handoff of Definition 1415.4, the window envelope of Definition 1415.6, the Bolzano frontier of Definition 1415.7, the RegSeqRat readback of Definition 1415.8, and the located-window choice boundary of Definition 1415.10 assemble the obligationClosure surface over*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont,

The assembled package exports exactly the finite bounded-sequence route $S, W, Q, R, I, H, C, P, N$ *and keeps open-cover compactness, trichotomy, host completeness, quotient stream equality, and choice-selected subsequences outside the packet.*

Proof. Start with the carrier and NameCert obligations. The carrier definition lists the finite source schedule, stream windows, regular readback, real seal, closed-bounded interval-net row, and the structural rows. The obligation theorem identifies those rows as the complete local certificate

surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Next use the finite-window bound and handoff theorems. They place every inspected sequence value on the displayed route from S through W, Q, R to I, and they show that the $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer reads the bounded source only through the same packet.

Finally add the envelope, frontier, and readback rows. These theorems keep downstream reads inside the visible route $S, W, Q, R, I, H, C, P, N$ and prevent the regular rational data from detaching from the bounded-window surface. Since none of the cited rows stores open-cover compactness, trichotomy, host completeness, quotient stream equality, or a choice-selected subsequence, the assembled obligation-closure package exports none of those claims. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceObligation closure package`

Theorem 1391.13 (Bounded real sequence carrier admission)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier*

$$B = (S, W, Q, R, I, H, C, P, N)$$

must expose inspected sequence indices S, finite $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *sequence seals R, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row I, componentwise transport H, displayed replay C, package provenance P, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row N before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer can read the bounded source. The admitted surface exports no open-cover compactness, trichotomy, quotient stream equality, or choice-selected subsequence.*

Proof. Project the carrier of Definition 1415.1. Its displayed tuple is exactly $S, W, Q, R, I, H, C, P, N$, so carrier admission starts with inspected indices, finite windows, regular readback, real sequence seals, the interval-net bound row, and the structural rows.

Use the NameCert obligations of Definition 1415.2. They identify those displayed rows as the complete local certificate surface and prevent a consumer from reading a bounded source before the windows, readback, seal, bound, transport, replay, provenance, and naming rows have appeared.

Apply the finite-window bound and Bolzano–Weierstrass handoff, Definition 1415.3 and Definition 1415.4. The first ties each inspected value to the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ bound row, and the second forces every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer to enter through the same bounded-source route.

Finally use the obligation-closure package of Definition 1415.12. Since that package contains no open-cover compactness, trichotomy, quotient stream equality, or choice-selected subsequence coordinate, the admitted carrier exports none of those objects to a Bolzano–Weierstrass or real-completion consumer. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceObligation carrier admission`

Definition 1391.14 (Bounded real sequence finite-window net)**.** *For an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier*

$$B = (S, W, Q, R, I, H, C, P, N),$$

the finite-window net packet is the displayed subroute

$$\mathcal{N}_B = (S, W, Q, R, I, H, C, P, N)$$

read as a closed-bounded interval-net source. Its entries are the inspected sequence schedule S, the synchronized $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{Real}^{\uparrow}$ *seal row R, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row I, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row N. The packet has no coordinate for an infinite subnet, open-cover compactness, trichotomy, quotient sequence equality, countable choice, or a selected Bolzano–Weierstrass subsequence.*

Lean4 (definition) `BEDC.Derived.BoundedRealSequenceUp.FiniteWindowNet.BoundedRealSequenceFiniteWindowNet`

Theorem 1391.15 (Bounded real sequence finite-window net extraction)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier exposes the finite-window net packet of Definition 1415.14 before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reads the bounded source. In particular, every such consumer factors through*

$$S, W, Q, R, I, H, C, P, N$$

and receives the closed-bounded interval-net route before it can inspect compactness-facing or completion-facing data.

Proof. Project the accepted carrier and form the packet $\mathcal{N}_B$ from the same rows. The carrier definition supplies $S, W, Q, R, I, H, C, P, N$ in exactly the order needed to read inspected windows, regular readback, real seals, and the interval-net bound row.

Apply Definition 1415.3. It identifies the finite bound route from S through W, Q, R to I and keeps the route outside trichotomy, host completeness, quotient equality, open-cover compactness, and selected subsequences.

Apply Definition 1415.4. The Bolzano–Weierstrass consumer reads its bounded sequence source only through the same visible rows. The structural rows H, C, P, N then repack transport, replay, provenance, and local naming as the finite-window net packet. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier finite window net extraction`

Theorem 1391.16 (Bounded real sequence regular-readback stability)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

and

$$B' = (S', W', Q', R', I', H', C', P', N')$$

be accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies the source schedules, stream windows, regular readback rows, real seals, and interval-net bound rows S, W, Q, R, I with S', W', Q', R', I', then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the closed-bounded interval-net bound row are preserved. Thus the transported packet still exposes the same finite-window net route before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reads it.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier regular readback stability`

Proof. Project both carriers. The hypothesis says that the five analytic rows S, W, Q, R, I and S', W', Q', R', I' agree under the componentwise transport recorded by the H rows.

Use Definition 1415.8. The readback available to a consumer is attached to the visible route $W \mapsto Q \mapsto R \mapsto I$, so transporting W, Q, R, I preserves the regular rational readback and the bound row together.

Use Definition 1415.15 to repack the transported rows as the finite-window net packet. The remaining rows H, C, P, N and H', C', P', N' supply the transport, replay, provenance, and naming data, so no detached compactness, quotient, choice, trichotomy, or completeness coordinate is introduced. □

Theorem 1391.17 (Bounded real sequence dyadic-bound transport)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

be an accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier. Every boundedness read transported from* B *to a* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or completion-facing consumer factors through finite stream windows* S, W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row* I*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row.*

The transported read may carry dyadic lower and upper bound ledgers from the finite window, but it cannot add trichotomy, quotient sequence equality, countable choice, a choice-selected subsequence, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier dyadic bound transport`

Proof. Project the carrier and read the finite-window net packet of Definition 1415.14. The packet lists $S, W, Q, R, I, H, C, P, N$ as the complete boundedness route.

Apply Definition 1415.16. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and interval-net bound row are stable under componentwise transport, so the transported boundedness row remains attached to the same finite stream windows.

Finally use Definition 1415.15. The Bolzano–Weierstrass or completion-facing consumer receives the closed-bounded interval-net source only after S, W, Q, R, I have been displayed, and the structural rows replay the same route without adding any excluded selector or completeness coordinate. □

Theorem 1391.18 (Bounded real sequence window factorization)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, any consumer read of boundedness factors through a finite window prefix before it reaches the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *source. More precisely, the read first inspects the sequence schedule* S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, then the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, and only then the interval-net bound row* I*.*

No consumer may bypass this route by asking for an infinite stream object, a quotient sequence equality, countable choice, a selected subsequence, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$. The carrier makes the inspected schedule and finite stream windows available before any real readback or interval bound can be consumed.

Use Definition 1415.3. It identifies the bound route as a finite passage from S through W, Q, R into the closed-bounded interval-net row I.

The structural rows H, C, P, N then transport, replay, source, and name the same finite passage. Since the carrier has no infinite stream, quotient equality, choice, trichotomy, open-cover, selected-subsequence, or ambient-completeness coordinate, the displayed window factorization is exhaustive. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier window factorization`

Theorem 1391.19 (Bounded real sequence window localization)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet localizes each inspected finite* $\mathsf{StreamName}^{\uparrow}$ *window through the displayed* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer may read the bounded source. More precisely, for*

$$B = (S, W, Q, R, I, H, C, P, N)$$

each inspected window in W is read through

$$S, W \longmapsto Q \longmapsto R \longmapsto I \longmapsto H, C, P, N.$$

The localization exports finite window bounds and regular readback only; it exports no host set of all sequence values, no choice-selected subsequence, no quotient stream equality, no open-cover compactness, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier window localization`

Proof. Project the carrier and use the finite-window net extraction of Definition 1415.15. That result displays $S, W, Q, R, I, H, C, P, N$ as the finite net packet a consumer must read before bounded-source data are available.

Apply regular-readback stability and dyadic-bound transport, Definition 1415.16 and Definition 1415.17. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ row Q attached to the same stream window W, real seal R, and closed-bounded interval-net row I under componentwise transport.

Finally use the window factorization of Definition 1415.18. It forces every boundedness read to pass through the inspected finite window before reaching the interval-net row, so a Bolzano–Weierstrass consumer receives only the displayed local route and none of the excluded host-level data. □

Theorem 1391.20 (Bounded real sequence Bolzano source exactness)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *source read from an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet is exactly the finite-window source*

$$S \to W \to Q \to R \to I \to H \to C \to P \to N.$$

Thus the Bolzano–Weierstrass consumer receives a bounded source only through finite stream windows, regular rational readback, the real-name seal, and the closed-bounded interval-net row; it receives no selected subsequence, no compactness theorem outside the displayed interval-net route, and no ambient real-completeness principle.

Proof. Read the accepted carrier and form the finite-window net packet. The rows S, W, Q, R, I display the source schedule, finite stream names, regular rational readback, real seal, and closed-bounded interval-net bound before the consumer is allowed to read the source.

Apply Definition 1415.4. That handoff sends Bolzano–Weierstrass consumers to the visible bounded source rather than to a choice-selected subsequence or host compactness theorem.

Use Definition 1415.17 to keep the dyadic bound ledger attached to the same source route under transport. The rows H, C, P, N replay, source, and name the handoff, so the source exactness statement has no remaining coordinate for excluded choice, quotient, compactness, trichotomy, or completeness principles. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier bolzano source exactness`

Theorem 1391.21 (Bounded real sequence compact-interval subsequence boundary)**.** *Every Bolzano–Weierstrass or compact-interval consumer of an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet must read finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seals, and* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound rows before any subsequence-facing claim is visible. The packet exposes the finite route*

$$S \to W \to Q \to R \to I \to H \to C \to P \to N$$

and does not export a selected subsequence, open-cover compactness, quotient sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-window bound of Definition 1415.3. It ties every inspected sequence value to the route from S through W, Q, R into the closed-bounded interval-net row I.

Apply the Bolzano–Weierstrass handoff and the window envelope, Definition 1415.4 and Definition 1415.6. They require a Bolzano-facing consumer to read the bounded source through the same finite window rows rather than through a selected subsequence or a host compactness principle.

Use Bolzano source exactness, Definition 1415.20. This identifies the visible source as the finite path through $S, W, Q, R, I, H, C, P, N$.

Finally apply finite-window net extraction, Definition 1415.15. The interval-net packet supplies the compact-interval-facing bound row while the structural rows transport, replay, source, and name the same route, leaving no coordinate for quotient equality, countable choice, open-cover compactness, selected subsequences, trichotomy, or ambient completeness. □

Theorem 1391.22 (Bounded real sequence bridge-facing route)**.** *The public bounded-source export for* BoundedRealSequence$^{\uparrow}$ *is bridge-facing exactly through the checked finite-window closure package and the Bolzano–Weierstrass source exactness route. A bridge consumer must first read Definition 1415.12 to expose the rows $S, W, Q, R, I, H, C, P, N$ and then read Definition 1415.20 before the bounded source can be handed to a standard bridge comparison. Thus the paper bridge surface is the finite-window route grounded by* BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequence *not a selected subsequence, open-cover compactness proof, quotient stream equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Use the obligation closure package. It gathers carrier admission, finite-window bounds, regular readback, dyadic transport, window factorization, and source handoff into one checked NameCert surface.

Now apply the Bolzano source exactness theorem. It identifies the only public export as the finite route

$$S \to W \to Q \to R \to I \to H \to C \to P \to N.$$

A bridge comparison can therefore inspect exactly those displayed rows before reading the bounded source. Since neither cited route exposes selected subsequences, quotient stream equality, open-cover compactness, or ambient completeness, the bridge-facing paper surface is exhausted by the checked closure package and the Bolzano source exactness route. □

Closure status for NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$**.**

Constructive story: A BoundedRealSequence$^{\uparrow}$ packet is generated from inspected sequence indices, finite StreamName windows, RegSeqRat readback, RealUp sequence-name seals, a ClosedBoundedIntervalNet bound row, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1415.1, Definition 1415.2, Definition 1415.12, Definition 1415.20, and Definition 1415.22 expose the finite-window bounded-sequence source as a bridge-facing route for Bolzano–Weierstrass consumers over StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceObligation\_closure\_packa`

Bridge status: none

Not claimed: This chapter does not close bridge checking, open-cover compactness, trichotomy, host completeness, quotient equality of streams, or choice-selected subsequences.

Upgrade path: A checked standard bridge would turn the bridge-facing finite-window route of Definition 1415.22 into bridgeChecked verification.

Chapter 1392

A Concrete Naming Certificate for $\mathsf{DecimalExpansion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DecimalExpansion}^{\uparrow}$ packet records finite decimal-prefix windows as BEDC readback data. It does not store a host infinite decimal, a quotient real, or a selected representative. Its role is to route base-ten digit observations through the finite stream surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic and rational place-value ledgers of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

The chapter is the decimal sibling of chapter 418 and chapter 342. The shared discipline is finite-window readback: binary prefixes, continued-fraction convergents, and decimal prefixes all expose only displayed rows before a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can read the packet.

The same finite decimal-prefix windows also form a forward consumer of chapter 1809. A decimal window gives a countable dense readback route into $\mathsf{Real}^{\uparrow}$, while $\mathsf{RealSeparability}^{\uparrow}$ supplies the global countable-density ledger that later real-name consumers use to compare those finite routes.

Definition 1392.1 (DecimalExpansion carrier)**.** *A* $\mathsf{DecimalExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, V, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite digit ledger, whose rows record decimal digits and the associated sign or integral-prefix convention when the local packet includes one;*
2. *W is the finite decimal-place window ledger, listing the inspected positions and the request precision served by those positions;*
3. *V is the place-value row that evaluates each inspected prefix as a rational sum with denominator a power of ten;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison row that transports the base-ten place-value read into the dyadic tolerance discipline used by downstream real consumers;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the same decimal window and place-value rows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, W, V, Q, R have been displayed;*
7. *H is the componentwise* hsame *transport row for digit, window, place-value, dyadic/rational, regular-sequence, and real-seal coordinates;*

8. *C is the* Cont *replay row that reuses the accepted finite decimal windows without adding a host stream;*
9. *P is the* Pkg *provenance over* $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$ *row.*

The carrier exposes no infinite decimal stream, Dedekind cut, quotient real, endpoint convention resolver, trichotomy decision, selected representative, or Real-completeness witness.

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.TasteGate.DecimalExpansionTasteGate single carrier alignment`

Theorem 1392.2 (DecimalExpansion digit-window obligations)**.** *Every real-facing read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must consume the digit ledger D, finite window ledger W, place-value row V, dyadic comparison row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E in that order. The structural rows H, C, P, N transport, replay, source, and name only this displayed route, so no decimal consumer can read a real-facing value before the finite digit window and dyadic comparison have both been displayed.*

Proof. Project the carrier of Definition 1417.1. Its public real-facing coordinates appear as D, W, V, Q, R, E before the structural rows H, C, P, N.

The finite-window discipline is the same one used by the sibling routes of chapter 418, chapter 342, and chapter 126. A consumer first receives inspected digits and positions, not an infinite stream.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and ledger exhaustion force every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer to inspect only displayed rows, with V and Q between the digit window and the regular-rational handoff.

Finally read the seal row E downstream of R. Since the carrier has no coordinate for a direct decimal stream, quotient real, selected representative, or completeness witness, the ordered digit-window route is the only real-facing decimal read exported by the packet. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitWindowObligations`

Theorem 1392.3 (DecimalExpansion NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$ *obligation surface consists exactly of the rows in Definition 1417.1. Carrier admission requires a finite digit ledger, a finite decimal-place window ledger, a base-ten rational place-value row, a dyadic/rational comparison row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, a* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through the same displayed rows. Ledger exhaustion says that every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer can inspect only $D, W, V, Q, R, E, H, C, P, N$.*

Proof. Carrier admission is read directly from the finite packet. The digit rows and decimal-place windows give the visible $\mathsf{StreamName}^{\uparrow}$ observation surface, while V evaluates each inspected prefix as a rational place-value sum.

The dyadic/rational row Q is the only bridge from base-ten arithmetic into the tolerance discipline already used by chapter 291. The regular handoff row R and real-seal row E are therefore downstream of the same inspected window.

Transport is componentwise because H has coordinates for each displayed row, and continuation only replays windows already accepted by C. The package and local naming rows record exactly those dependencies.

For ledger exhaustion, project a consumer read from the accepted packet. The definition gives only the ten coordinates $D, W, V, Q, R, E, H, C, P, N$, so any route through an infinite stream, quotient real, selected endpoint convention, or ambient completeness principle is outside the carrier. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionNameCertObligations`

Theorem 1392.4 (DecimalExpansion tail digit stability)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet*

$$K = (D, W, V, Q, R, E, H, C, P, N),$$

finite digit tails are stable under the displayed hsame *transport row H. A consumer that transports the accepted packet can compare tail digits only through the same finite digit ledger D, window ledger W, place-value row V, dyadic comparison row Q, regular handoff R, real seal E, replay row C, provenance row P, and local name row N.*

Proof. Project the obligation surface of Definition 1417.3. The only digit data are the displayed rows D and W, and every place-value or real-facing read is downstream of them.

The transport row H is componentwise through the same carrier coordinates. Therefore a transported tail cannot acquire a new infinite decimal stream or a new endpoint selector; it can only replay the displayed finite tail rows already present in $D, W, V, Q, R, E, C, P, N$. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionTailDigitStability`

Theorem 1392.5 (DecimalExpansion rational Real handoff)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet $K = (D, W, V, Q, R, E, H, C, P, N)$, the rational place-value row V must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison row Q, then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is public. No consumer can read a real-valued decimal prefix directly from D, W, V.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRatRealHandoff`

Proof. Read the carrier order in Definition 1417.1. The rows list digit data, window data, place-value evaluation, dyadic/rational comparison, regular handoff, and real seal as D, W, V, Q, R, E.

Apply Definition 1417.3. Carrier admission and classifier transport preserve this order componentwise, and ledger exhaustion prevents any route from bypassing Q or R before reaching E. □

Theorem 1392.6 (DecimalExpansion nonescape)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet $K = (D, W, V, Q, R, E, H, C, P, N)$ exports exactly the finite decimal-prefix, dyadic comparison, regular handoff, real seal, transport, replay, provenance, and naming rows listed in Definition 1417.1. It exports no infinite decimal, quotient real, endpoint selector, host trichotomy row, selected representative, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionNonescape`

Proof. Use ledger exhaustion from Definition 1417.3. The accepted packet has exactly the coordinates $D, W, V, Q, R, E, H, C, P, N$, so every public read is a projection or replay of those rows.

The excluded objects would require a coordinate outside the displayed carrier: an infinite digit stream, a quotient real, a selector for endpoints, a trichotomy decision, a selected representative, or a completeness witness. Since no such coordinate is present, no accepted consumer can recover it from the packet. □

Theorem 1392.7 (DecimalExpansion RealUp readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads an inspected decimal-place window, then the digit rows on that window, then the rational place-value and dyadic tolerance rows, then the regular rational handoff, and only then the displayed real seal. The packet supplies no Dedekind cut, quotient real, endpoint uniqueness convention, trichotomy principle, selected representative, or Real-completeness witness.

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealReadbackBoundary`

Proof. Start with an accepted carrier $K = (D, W, V, Q, R, E, H, C, P, N)$. The $\mathsf{StreamName}^{\uparrow}$ observation is W together with the digit rows of D on that window, so a real-facing consumer cannot read a decimal prefix without first reading the finite window.

The place-value row V turns that same finite prefix into a rational base-ten sum. The row Q records the dyadic and rational comparison data used to pass from the decimal sum to the tolerance language consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

The handoff row R is therefore reached only after D, W, V, Q are fixed. The seal row E is listed after R, and the rows H, C, P, N preserve the same dependency route under transport, continuation, provenance, and local naming.

Thus every $\mathsf{Real}^{\uparrow}$ read factors through the displayed finite route. The excluded objects are not carrier coordinates, so no consumer of the packet can obtain them from this readback boundary. □

Theorem 1392.8 (DecimalExpansion RegSeqRat handoff)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet routes finite decimal digit rows through rational place-value rows and dyadic comparison rows before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route is exhausted by the displayed coordinates*

$$D, W, V, Q, R, E, H, C, P, N$$

of Definition 1417.1. In particular, the packet exports no quotient real, endpoint representative, trichotomy row, selected decimal stream, or $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatHandoff`

Proof. Project the carrier of Definition 1417.1. The finite digit ledger D is read together with the finite decimal-place windows W, and the place-value row V evaluates exactly those inspected prefixes as rational base-ten sums.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and classifier transport keep D, W, V attached to the same dyadic and rational comparison row Q, while ledger exhaustion prevents a consumer from bypassing the displayed handoff surface.

Finally apply the RealUp readback boundary of Definition 1417.7. The regular handoff row R is reached only after the digit, window, place-value, and dyadic rows are displayed, and the terminal seal row E remains downstream of R with the structural rows H, C, P, N preserving the same route. □

Theorem 1392.9 (DecimalExpansion window normalization obligation)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must display the finite digit ledger D and finite window ledger W before any place-value, dyadic-comparison, regular-rational, or real-seal row can be read. Equivalently, every public read of V, Q, R, E factors through the finite decimal-prefix surface named in Definition 1417.1 and constrained by Definition 1417.3.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionWindowNormalizationObligation`

Proof. Project the carrier of Definition 1417.1. The rows are ordered as $D, W, V, Q, R, E, H, C, P, N$, so digit admission and inspected-window admission are the visible source rows for the packet.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and ledger exhaustion require every consumer to read only displayed coordinates, hence no consumer can reach V, Q, R, E without first reading D and W.

The L10 sibling route is the same finite-window route already used by chapter 126, chapter 301, and chapter 38. Thus window normalization is a finite ledger obligation, not an infinite decimal-stream selection. □

Theorem 1392.10 (DecimalExpansion prefix stability)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, finite decimal prefixes are stable under* $\mathsf{StreamName}^{\uparrow}$ *window refinement and componentwise* hsame *transport. If a consumer refines the inspected window* W *without changing the displayed prefix rows in* D*, then the place-value row* V*, the dyadic tolerance row* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *remain attached to the same prefix route before any downstream read is exposed.*

Proof. Begin with the carrier of Definition 1417.1. The digit ledger D and window ledger W are the only source rows from which a finite decimal prefix can be read.

Use the window normalization obligation of Definition 1417.9. It forces every public read of V, Q, R, E to pass through the displayed D, W prefix surface, so a window refinement that preserves the displayed prefix rows cannot change the downstream route.

The $\mathsf{StreamName}^{\uparrow}$ sibling of chapter 126 supplies the finite-window discipline, and the structural row H transports the same prefix componentwise through hsame. The dyadic and regular-rational consumers of chapter 291 and chapter 301 therefore receive the same place-value and tolerance route before the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixStability`

Theorem 1392.11 (DecimalExpansion dyadic comparison handoff)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, the place-value row* V *reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff only through the comparison row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison surface. No direct route from a decimal place-value sum to* R *or* E *is exported by the packet.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicComparisonHandoff`

Proof. Read Definition 1417.1. The row V records the base-ten rational place-value sum, while Q is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ and rational comparison row.

Apply Definition 1417.3. The obligation surface places Q between place-value evaluation and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and classifier transport is componentwise through that same displayed row.

Finally apply Definition 1417.8. It states that finite digit rows pass through rational place-value rows and dyadic comparison rows before reaching R and the terminal seal E. The dyadic handoff therefore uses the existing routes of chapter 291 and chapter 301, with no bypass from V to R. □

Theorem 1392.12 (DecimalExpansion dyadic rational readback)**.** *Every finite decimal prefix admitted by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal is public. Concretely, the prefix rows* D, W *first determine the rational place-value row* V*, the dyadic and rational comparison row* Q *then supplies the finite*

tolerance comparison, and only then may the regular rational handoff R and terminal seal E be consumed.

Proof. Project the carrier of Definition 1417.1. The packet orders the rows as $D, W, V, Q, R, E, H, C, P, N$, so decimal digits and inspected windows are read before place values, dyadic tolerance, regular readback, and the real seal.

Apply prefix stability, Definition 1417.10. It keeps the finite prefix route fixed under $\mathsf{StreamName}^{\uparrow}$ window refinement and hsame transport, which is the only way a downstream consumer can compare refined windows.

Now use the dyadic comparison handoff of Definition 1417.11. The row Q is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ and rational comparison route, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301 cannot be reached directly from V.

The terminal real read is the seal discipline of chapter 38, and the decimal normal-form consumer of chapter 3952 reads only after this finite dyadic and regular-rational route has been displayed. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicRationalReadback`

Theorem 1392.13 (DecimalExpansion RealSeparability density route)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the finite decimal prefix route contributes to the RealSeparability-facing density surface only through displayed countable dense readback along*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff cited by chapter 1809 reads $D, W, V, Q, R, E, H, C, P, N$ as finite prefix, dyadic comparison, regular-rational, seal, transport, replay, provenance, and naming rows. It exports no quotient real, selected endpoint convention, trichotomy row, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSeparabilityDensityRoute`

Proof. Project the carrier of Definition 1417.1. The digit row D and window row W provide the only prefix data, while V, Q, R, E place that prefix into rational place value, dyadic tolerance, regular rational readback, and the terminal real seal.

Apply Definition 1417.12. The theorem forces the readback route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before E is exposed, so the density consumer can only see a displayed finite route.

Finally read the RealSeparability sibling of chapter 1809. Its countable dense surface consumes finite readback data; the decimal packet supplies that surface through $D, W, V, Q, R, E, H, C, P, N$ and has no coordinate for quotient reals, selected endpoints, trichotomy, or real completeness. □

Theorem 1392.14 (DecimalExpansion countable dense consumer handoff)**.** *Every countable dense consumer that reads an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$ *must first pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the digit ledger D, the place-value row V, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The consumer handoff to chapter 1809 is therefore a displayed dense readback route, not a decimal stream selector, quotient-real representative, endpoint convention, trichotomy decision, or completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionCountableDenseConsumerHandoff`

Proof. Use Definition 1417.13. It identifies the only RealSeparability-facing route exported by the packet as the finite path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The structural rows then keep that route componentwise. Transport by H, replay by C, provenance by P, and the local name N certify the same displayed rows without inserting an ambient sequence selector or quotient-real comparison.

Thus a countable dense consumer receives exactly the finite prefix readback route already present in D, W, V, Q, R, E. Since the carrier has no selected decimal stream, endpoint convention, trichotomy, or completeness coordinate, none can be exported by the handoff. □

Theorem 1392.15 (DecimalExpansion prefix density exhaustion)**.** *For an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, all RealSeparability-facing density reads are exhausted by finite decimal prefixes and their displayed readback through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Once* $D, W, V, Q, R, E, H, C, P, N$ *have been read, the packet has no remaining row from which to choose a quotient real, select an endpoint convention, decide trichotomy, or assert a* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixDensityExhaustion`

Proof. Begin with the countable dense consumer handoff of Definition 1417.14. It routes every density consumer through the finite prefix ledger and the dyadic, regular-rational, and real-seal rows.

Apply the excluded-coordinate clause of Definition 1417.1. The packet lists only $D, W, V, Q, R, E, H, C, P, N$, so there is no extra coordinate after the terminal seal and structural rows have been consumed.

The RealSeparability reference in chapter 1809 therefore receives a countable dense readback route from finite decimal prefixes. It does not receive quotient real equality, selected endpoint data, trichotomy, or a completeness row from $\mathsf{DecimalExpansion}^{\uparrow}$. □

Theorem 1392.16 (DecimalExpansion dyadic normalization lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet passes through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit-place tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row before any real-facing decimal value is consumed. In particular, the digit-place route is the same L10 sibling lattice exposed by chapter 291, chapter 301, chapter 126, and chapter 38; the packet exports no direct decimal-to-real shortcut.*

Proof. Project the carrier of Definition 1417.1. The digit rows D and finite decimal windows W feed the place-value row V, while the comparison row Q is the only displayed dyadic and rational tolerance route.

Apply Definition 1417.11. It places Q between every place-value read and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, so the decimal packet cannot jump from V directly to the seal row.

Finally apply Definition 1417.7. The terminal real read is reached only after finite windows, digit rows, dyadic tolerance, and regular rational readback have been displayed. This locks the decimal route to the dyadic, regular-Cauchy, stream-window, and real-seal sibling lattice named in the statement. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicNormalizationLock`

Theorem 1392.17 (DecimalExpansion Real seal nonescape)**.** *The terminal row* E *of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet exports only the displayed* $\mathsf{Real}^{\uparrow}$ *seal reached from* D, W, V, Q, R*. It exports no quotient real equality, endpoint resolver, trichotomy decision, selected digit stream, selected decimal stream, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with the real readback boundary of Definition 1417.7. Every $\mathsf{Real}^{\uparrow}$ read factors through the finite route from $\mathsf{StreamName}^{\uparrow}$ windows to decimal place value, dyadic comparison, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and then the seal row E.

Use Definition 1417.12 to keep the route through Q and R. The seal is downstream of the dyadic, rational, and regular-rational rows, so it cannot introduce an endpoint resolver, trichotomy row, selected digit stream, selected decimal stream, or countable-choice selector that is not already present in the carrier.

Finally project the excluded-coordinate clause of Definition 1417.1. Since quotient reals, endpoint convention resolvers, trichotomy decisions, selected representatives, selected decimal streams, and completeness witnesses are absent from the displayed packet, none escapes through E. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSealNonescape`

Theorem 1392.18 (DecimalExpansion Real seal factorization)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet whose terminal row is read as a* $\mathsf{Real}^{\uparrow}$ *seal factors through finite decimal windows, dyadic comparison, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{RealSeparability}^{\uparrow}$ *countable-density route. Concretely, the seal read begins with the digit and window rows D, W, evaluates those rows through the decimal place-value row V, passes through the dyadic and rational comparison row Q, reaches the regular rational row R, and only then exposes the terminal seal E. The same finite decimal windows supply the countable dense readback interface named by chapter 1809; no direct decimal stream, quotient real, selected representative, or completeness witness is exported by the seal.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSealFactorization`

Proof. Start with the dyadic normalization lock of Definition 1417.16. It forces every real-facing decimal value to pass through the digit-place tolerance ledger, finite window row, regular rational readback row, and terminal seal.

Apply the Real seal nonescape theorem Definition 1417.17. The terminal row E cannot add a quotient real, selected decimal stream, endpoint resolver, or completeness witness after the finite route has reached it.

Finally use the countable-density role of $\mathsf{RealSeparability}^{\uparrow}$ described in the chapter introduction. The same finite decimal-prefix windows that feed D, W, V, Q, R, E are the windows counted by the RealSeparability route, so the factorization records a finite dense readback path rather than a direct decimal-to-real shortcut. □

Theorem 1392.19 (DecimalExpansion digit-carry determinacy)**.** *Let*

$$K = (D, W, V, Q, R, E, H, C, P, N) \quad \text{and} \quad K' = (D', W', V', Q', R', E', H', C', P', N')$$

be accepted $\mathsf{DecimalExpansion}^{\uparrow}$ *packets. If K and K' display the same finite digit window and the same dyadic place-value ledger before their terminal seals are read, then they expose the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route before the* $\mathsf{Real}^{\uparrow}$ *seal. Carry normalization is confined to the finite window ledger: it may rewrite the displayed digit and place-value rows inside D, W, V, Q, but it cannot introduce a second regular-rational route, a direct decimal-to-real shortcut, a quotient real, a selected representative, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with window normalization, Definition 1417.9. It forces both packets to display their finite digit ledgers and finite window ledgers before V, Q, R, E can be read, so the shared window fixes the source surface for any carry operation.

Use the dyadic comparison handoff, Definition 1417.11. The shared dyadic place-value ledger places both packets on the same comparison row before either packet reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Next apply the dyadic normalization lock, Definition 1417.16. It prevents carry normalization from bypassing the dyadic and regular-rational route; the normalized digit rows remain confined to the finite window and comparison ledgers.

Finally use Real seal factorization, Definition 1417.18. The terminal seal is downstream of D, W, V, Q, R, so equality of the displayed window and dyadic place-value ledger determines the readback route before E is exposed. Since the carrier has no quotient-real, selected-representative, direct decimal-stream, or completeness coordinate, carry normalization exports no additional route beyond the finite ledger. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitCarryDeterminacy`

Theorem 1392.20 (DecimalExpansion separability density obligation)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet supplies its* $\mathsf{RealSeparability}^{\uparrow}$ *density reads only through finite decimal-prefix windows. More precisely, a density-window consumer must read the digit ledger D, the decimal-place window ledger W, the place-value row V, the dyadic comparison row Q, the regular rational handoff R, the terminal real seal E, and then the structural rows H, C, P, N of Definition 1417.1; no dense route is exported from an infinite decimal stream, a quotient real, or a selected representative.*

Proof. Project the carrier of Definition 1417.1. The only rows that can feed a density read are the displayed finite prefix rows D, W, V, Q, R, E together with the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 1417.3. Ledger exhaustion restricts every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer to the same finite coordinates, so a density-window consumer cannot introduce an ambient decimal stream.

Use the Real seal factorization of Definition 1417.18. The finite decimal windows that feed the seal are exactly the windows named by the RealSeparability sibling route in chapter 1809. Thus separability density is reached by finite BEDC readback rather than by a selected decimal representative. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionSeparabilityDensityObligation`

Theorem 1392.21 (DecimalExpansion digit ledger injection)**.** *The digit ledger of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet injects into finite* $\mathsf{StreamName}^{\uparrow}$ *windows before any dyadic comparison is read. Concretely, every use of the comparison row Q must first display the same finite digit and window rows D, W and the place-value row V that determine the inspected decimal prefix.*

Proof. Read the window normalization obligation of Definition 1417.9. It forces every public read of V, Q, R, E to pass through the finite digit ledger D and window ledger W.

The $\mathsf{StreamName}^{\uparrow}$ role is therefore fixed before dyadic comparison begins. The inspected decimal prefix is a finite windowed row, not a host stream, and V evaluates only that displayed prefix as a rational place-value sum.

Apply the dyadic comparison handoff of Definition 1417.11. Since Q is the only displayed route from place value to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, each dyadic comparison read carries the prior D, W, V injection into finite StreamName windows with it. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitLedgerInjection`

Theorem 1392.22 (DecimalExpansion finite prefix completeness)**.** *Every finite-prefix consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is exhausted by the rows*

$$D, W, V, Q, R, E, H, C, P, N.$$

In particular, a consumer that requests decimal digits, finite stream windows, dyadic comparison, regular rational readback, real sealing, transport, replay, provenance, or local naming finds all available data inside those ten rows and no additional prefix-completion coordinate.

Proof. Start from the carrier definition. The packet is exactly the finite tuple $K = (D, W, V, Q, R, E, H, C, P, N)$, and the excluded-coordinate clause removes infinite streams, Dedekind cuts, quotient reals, endpoint resolvers, trichotomy decisions, selected representatives, and completeness witnesses.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and classifier transport are componentwise through the same ten rows, while ledger exhaustion lets consumers inspect only those displayed coordinates.

Finally combine the Real readback boundary of Definition 1417.7 with the dyadic normalization lock of Definition 1417.16. Every real-facing finite-prefix read moves through digit windows, dyadic tolerance, regular rational readback, and the terminal seal before structural replay and naming, so $D, W, V, Q, R, E, H, C, P, N$ exhaust the prefix-completeness surface. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionFinitePrefixCompleteness`

Theorem 1392.23 (DecimalExpansion endpoint-normalization handoff)**.** *Every endpoint-sensitive decimal consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must pass the rows*

$$D, W, V, Q, R, E$$

through $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *before any* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *consumer reads the normalized prefix. The handoff exports only the finite digit ledger, decimal window, place-value row, dyadic comparison, regular rational handoff, and terminal Real seal already displayed by the packet; it exports no repeating-nine endpoint convention, quotient real, selected stream, or Real completeness witness.*

Proof. Start from the finite prefix completeness theorem Definition 1417.22. It exhausts every prefix-sensitive read by $D, W, V, Q, R, E, H, C, P, N$ and leaves no additional endpoint coordinate in the $\mathsf{DecimalExpansion}^{\uparrow}$ packet.

Use the window normalization obligation Definition 1417.9. It forces each public read of place values, dyadic comparisons, regular handoffs, and Real seals to pass through the finite digit and window rows D, W.

Apply the dyadic comparison handoff of Definition 1417.11. The comparison row Q is the only displayed passage from place-value data in V toward the regular rational row R and the seal E.

The endpoint-sensitive part is therefore delegated to chapter 1429. Only after that finite endpoint-normalization surface has acted on the same D, W, V, Q, R, E route may the canonical readback surface of chapter 3952 read the normalized prefix. The excluded endpoint convention, quotient real, selected stream, and completeness witness are absent from the DecimalExpansion carrier and cannot be supplied by this handoff. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointNormalizationHandoff`

Theorem 1392.24 (DecimalExpansion endpoint Cauchy schedule)**.** *Every endpoint-sensitive Cauchy schedule consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must traverse the finite digit window rows* D, W*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, endpoint normalization, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *before any real-facing decimal read is exported. The schedule exposes only finite prefix observations and their normalized endpoint route; it does not select a decimal stream, choose a repeating-nine convention, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the endpoint-normalization handoff of Definition 1417.23. It already forces every endpoint-sensitive read through the displayed D, W, V, Q, R, E route before the normalized decimal surface is read.

Use finite prefix completeness from Definition 1417.22. The packet has no additional coordinate beyond the displayed digit, window, place-value, dyadic comparison, regular readback, Real seal, transport, replay, provenance, and naming rows.

The Cauchy schedule is therefore read through the same finite route. The dyadic comparison Q supplies the requested tolerance window, R supplies regular rational readback, endpoint normalization resolves only the displayed finite endpoint case, and E is the terminal Real seal for the real-facing read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointCauchySchedule`

Theorem 1392.25 (DecimalExpansion finite-prefix separability route)**.** *Every finite-prefix separability consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must read finite digit windows* D, W*, dyadic comparison* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, endpoint normalization, the* $\mathsf{RealSeparability}^{\uparrow}$ *finite-density route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before treating the decimal packet as a real-separating datum.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionFinitePrefixSeparabilityRoute`

Proof. Project the carrier through Definition 1417.1. A separability read receives the same finite prefix data as every other consumer: digit windows, place values, dyadic comparisons, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Apply the finite prefix completeness theorem Definition 1417.22. It prevents a separability consumer from asking for an infinite decimal stream or a selected representative outside the packet.

Now use the Real seal factorization already cited by the chapter and route density through chapter 1809. Endpoint normalization acts before this density read, so the finite decimal prefix is normalized, read back through R, and sealed by E before the $\mathsf{RealSeparability}^{\uparrow}$ surface consumes it. □

Theorem 1392.26 (DecimalExpansion real window exhaustion)**.** *For an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$*-facing decimal read is exhausted by finite digit windows* D, W*, dyadic comparison* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, endpoint normalization, optional* $\mathsf{RealSeparability}^{\uparrow}$ *finite-density inspection, and the terminal Real seal* E*. No real-facing consumer can bypass these rows by importing selected decimal streams, quotient-real equality, trichotomy, or Real completeness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealWindowExhaustion`

Proof. Begin with Definition 1417.7. It places every real-facing read behind the displayed decimal windows, dyadic comparison, regular rational readback, and Real seal.

Add the endpoint-normalization handoff of Definition 1417.23. Endpoint-sensitive reads do not create a second route; they use the same finite prefix rows and then pass through the finite endpoint-normalization surface.

Finally apply Definition 1417.25 when the consumer asks for density or separability data. The $\mathsf{RealSeparability}^{\uparrow}$ surface receives only the finite normalized window, regular readback, and terminal seal, so the displayed rows exhaust the real-facing decimal window. □

Theorem 1392.27 (DecimalExpansion real-window audit route)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the finite-window exhaustion of Definition 1417.26 is an audit route through exactly*

$$D, W, Q, R, E, H, C, P, N.$$

Every $\mathsf{Real}^{\uparrow}$*-facing read that begins in digit windows or endpoint-sensitive windows must pass through the displayed dyadic comparison,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it is accepted as a decimal real-window observation.*

Proof. Begin with Definition 1417.26. It already confines real-facing decimal reads to finite digit windows, dyadic comparison, regular rational readback, endpoint normalization, optional finite-density inspection, and the terminal Real seal.

Use Definition 1417.7 and Definition 1417.11 to name the concrete route. The read moves from D, W through Q before it reaches R and E, so no selected decimal stream or quotient-real equality can enter the audit path.

Finally append the structural rows supplied by Definition 1417.3. Componentwise transport, replay, provenance, and local naming are exactly H, C, P, N, and the carrier has no coordinate outside $D, W, V, Q, R, E, H, C, P, N$ from which another audit row could be read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealWindowExhaustionAuditRoute`

Theorem 1392.28 (DecimalExpansion endpoint-window transport)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, endpoint-sensitive finite windows transport through the same endpoint-normalization handoff used by Definition 1417.23. The transported read exposes only the displayed digit ledger* D*, window ledger* W*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*; it does not select a decimal stream, choose a repeating-nine convention, or import a detached endpoint resolver.*

Proof. Project the packet through Definition 1417.1. Endpoint-sensitive reads have access to the same finite rows as every other decimal consumer and no additional endpoint coordinate.

Apply Definition 1417.23. Its handoff sends the endpoint-sensitive part through D, W, V, Q, R, E before a normalized real-decimal read is exported.

The transport part is supplied by Definition 1417.9 and Definition 1417.22. Window refinement is componentwise through H, replay is through C, provenance through P, and the local name through N, so the endpoint window remains inside the displayed packet instead of selecting an external stream. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointWindowTransport`

Theorem 1392.29 (DecimalExpansion RegSeqRat prefix coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing decimal prefix read exported by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is covered by finite digit windows together with dyadic comparison. Concretely, the read starts in* D, W*, evaluates its place-value and tolerance information through* V, Q*, reaches the regular rational row* R*, and then remains governed by* E, H, C, P, N *before any downstream* $\mathsf{Real}^{\uparrow}$ *consumer may inspect it.*

Proof. Use Definition 1417.8. It places each regular-rational handoff after the finite decimal prefix and its comparison ledger, rather than after an infinite decimal stream.

Apply Definition 1417.11. The row Q is the required finite dyadic comparison between place-value data and the regular rational route, so a $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot read R directly from D, W, V.

Finally combine Definition 1417.22 with Definition 1417.27. The prefix remains covered by finite windows and dyadic comparison after the Real seal and structural rows are inspected, and no selected representative, trichotomy decision, or ambient completeness witness is exported. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatPrefixCoverage`

Theorem 1392.30 (DecimalExpansion prefix normalization triad)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *finite-window read over digit windows* D, W*, place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *normalizes through the triad of finite prefix coverage, dyadic comparison, and Real readback boundary. The normalized read refuses quotient reals, selected decimal streams, endpoint convention selectors, trichotomy, and Real completeness before any downstream consumer may inspect* R *or* E*.*

Proof. Start with Definition 1417.29. It already places each $\mathsf{RegSeqRat}^{\uparrow}$-facing prefix read inside finite digit windows D, W, the place-value row V, and dyadic comparison Q before R is exposed.

Next apply Definition 1417.16. This prevents a normalized prefix from being replaced by a selected stream or by an endpoint convention outside the displayed rows.

Finally use Definition 1417.7. The route reaches E only after D, W, V, Q, R have been inspected, so quotient reals, trichotomy, and ambient completeness are unavailable to the normalized prefix read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixNormalizationTriad`

Theorem 1392.31 (DecimalExpansion terminal carry refusal)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, a terminal-carry read on finite digit windows* D, W *must pass through the place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *as displayed packet data. It cannot discharge the carry by choosing a repeating-nine representative, selecting a decimal stream, invoking endpoint convention selectors, applying trichotomy, identifying quotient reals, or using Real completeness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionTerminalCarryRefusal`

Proof. Read the endpoint side through Definition 1417.28. Endpoint-sensitive finite windows remain attached to D, W, Q, R, E and do not export a detached endpoint resolver.

Use Definition 1417.9 together with Definition 1417.22. A carry inspection is still a finite prefix inspection, so the place-value row V and comparison row Q must remain visible before R is read.

The refusal clause follows from Definition 1417.27. The terminal Real seal E is reached only through the finite-window audit route, leaving no coordinate for a selected stream, quotient-real equality, trichotomy, endpoint convention selector, or completeness witness. □

Theorem 1392.32 (DecimalExpansion RegSeqRat density unblock)**.** *The finite decimal-window route supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *density-facing read required by later real-name consumers: digit windows* D, W *expose place-value data* V*, dyadic comparison* Q *bounds the prefix against the regular rational row* R*, and the terminal seal* E *records the Real handoff. This density-facing read is a finite BEDC route and not a quotient-real argument, selected decimal stream, endpoint selector, trichotomy decision, or Real-completeness application.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatDensityUnblock`

Proof. Begin with Definition 1417.8. It routes decimal prefixes to $\mathsf{RegSeqRat}^{\uparrow}$ through displayed finite windows rather than through an infinite stream representative.

Combine it with Definition 1417.29. The place-value row V and dyadic comparison row Q are the finite density evidence that lets R be consumed by a downstream real-name route.

Finally apply Definition 1417.7 and Definition 1417.27. The Real seal E is a terminal packet row after D, W, V, Q, R, so the density-facing read stays finite and refuses quotient reals, selected streams, endpoint selectors, trichotomy, and ambient completeness. □

Theorem 1392.33 (DecimalExpansion carry normalization totality)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *finite digit-window packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, every carry or borrow read over the inspected decimal window has a normalized finite-window representative using only the rows* D, W, V, Q*. The normalized representative preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and it introduces no selected decimal stream, endpoint convention, quotient real, trichotomy decision, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionCarryNormalizationTotality`

Proof. Induct over the finite digit window named by D, W. The empty and one-place windows normalize by reading the displayed place-value row V and leave Q, R, E unchanged.

For the extension step, replay the local carry ledger inside the next displayed digit position. The dyadic comparison row Q records the tolerance effect of the carry move, so the normalized prefix still reaches the same finite comparison surface.

Finally use Definition 1417.29 and Definition 1417.18. Since R and E are downstream of the normalized finite prefix, no extra stream selector, endpoint convention, quotient representative, trichotomy row, or completeness coordinate can appear. □

Theorem 1392.34 (DecimalExpansion signed window transport)**.** *Let* $K = (D, W, V, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet. If a signed decimal consumer attaches a displayed sign row to the same finite window and transports that signed read by the componentwise* hsame *row* H*, then the transported read preserves the digit ledger* D*, window row* W*, place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The sign transport therefore remains a finite decimal-window route and does not create a second magnitude, selected signed expansion, quotient real, trichotomy decision, or detached Real seal.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionSignedWindowTransport`

Proof. Start from the componentwise transport row H in Definition 1417.1. It transports the displayed packet rows and has no independent coordinate for a second magnitude or a selected signed expansion.

The transported signed read still exposes D, W, V, Q before it reaches R. Apply Definition 1417.9 to keep the place-value and dyadic comparison data attached to the same finite window.

The terminal seal is then controlled by Definition 1417.7. Since E is read only after the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the signed transport cannot bypass the finite decimal route or import quotient equality, trichotomy, or a detached Real-facing value. □

Theorem 1392.35 (DecimalExpansion RegSeqRat digit replay)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is replayable from the finite digit ledger. Concretely, for* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the row* R *is determined for downstream consumers by replaying* D, W *through the place-value row* V *and dyadic comparison row* Q *before the terminal seal* E *is exposed. No downstream consumer can replace that replay by a quotient real, selected decimal stream, endpoint selector, trichotomy decision, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatDigitReplay`

Proof. Read the digit ledger through Definition 1417.21. Every comparison use of Q first displays the same finite digit and window rows D, W together with the place-value row V.

Next apply Definition 1417.29. The $\mathsf{RegSeqRat}^{\uparrow}$ row is covered by finite digit windows and dyadic comparison, so replaying D, W, V, Q reconstructs the public route to R.

Finally use Definition 1417.32. The density-facing handoff and the Real seal remain finite BEDC routes, leaving no coordinate for quotient reals, selected decimal streams, endpoint selectors, trichotomy, or ambient completeness. □

Theorem 1392.36 (DecimalExpansion public window export)**.** *Every public-facing* $\mathsf{DecimalExpansion}^{\uparrow}$ *consumer must expose the digit ledger D, decimal window row W, place-value row V, dyadic and rational comparison row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport, replay, provenance, and local* NameCert *rows before reading a real-facing decimal value. The export is a finite decimal-window route and not a quotient real, selected decimal expansion, endpoint selector, trichotomy decision, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPublicWindowExport`

Proof. Start with the window obligations. Theorems 1417.2 and 1417.9 force every decimal read to display D, W, V, Q before any downstream handoff is available.

Next use the RegSeqRat route. Theorems 1417.8, 1417.29, and 1417.35 keep the handoff row R replayable from the finite digit ledger and dyadic comparison rows.

Finally apply the Real boundary and the non-escape audit. Theorems 1417.7, 1417.27, and 1417.34 leave the terminal seal E downstream of the finite route and provide no coordinate for quotient reals, selected expansions, endpoint selectors, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DecimalExpansion}^{\uparrow}$ packet is generated from finite decimal digit rows, decimal-place windows, rational place-value rows, dyadic/rational comparison rows, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1417.1, Definition 1417.3, Definition 1417.2, Definition 1417.7, Definition 1417.8, Definition 1417.9, Definition 1417.29, Definition 1417.35, and Definition 1417.36 bind the finite decimal-window carrier, digit classifier, RegSeqRat and Real-facing readback rows, hsame transport, Cont replay, Pkg provenance, local NameCert rows, and the public decimal-window consumer export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, endpoint uniqueness, trichotomy, selected representatives, RealUp completeness, or a standard bridge.

Upgrade path: No further paper-axis closure is claimed here; bridge and Lean verification remain separate axes.

Chapter 1393

A Concrete Naming Certificate for $\mathsf{ArchimedeanOrderedField}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ packet records a traditional ordered-field comparison certificate over displayed constructive real names. It sits between the algebra/order surface of chapter 766, the rational-bound route of chapter 430, the real-name seal of chapter 38, and the rational source rows of chapter 37. The packet is a comparison carrier for downstream L10 and real-algebra consumers, not an ambient ordered field object.

Definition 1393.1 (ArchimedeanOrderedField carrier)**.** *An* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, A, Q, O, B, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the family of* $\mathsf{Real}^{\uparrow}$ *endpoint rows whose algebra and order reads are being compared;*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row that supplies displayed addition, negation, multiplication, absolute-value, nonnegative-order, and apartness evidence for those endpoints;*
3. *Q is the finite* $\mathsf{Rat}^{\uparrow}$ *source family used by constant-rational witnesses and Archimedean comparison bounds;*
4. *O is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *rational-bound row attached to the displayed real endpoints;*
5. *B is the local comparison ledger recording the field-operation, order, rational-embedding, and Archimedean-bound reads that the packet exports;*
6. *H stores componentwise* hsame *transports for the real endpoints, algebra/order rows, rational source rows, Archimedean bound rows, and local ledger rows;*
7. *C stores the* Cont *routes that expose the displayed operation, order, rational-bound, and rational-embedding rows to downstream consumers;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedField}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only by replaying these finite coordinates. It has no coordinate for real trichotomy, quotient real equality, a host ordered-field carrier, a quotient-reals construction, a selected least bound, or a standard bridge theorem.

Lean4 (definition) `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldCarrier`

Theorem 1393.2 (ArchimedeanOrderedField NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedField}^{\uparrow}}$ *surface for the carrier of Definition 1419.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet* $F = (R, A, Q, O, B, H, C, P, N)$ *with real endpoint, algebra/order, rational source, Archimedean bound, comparison ledger, transport, continuation, provenance, and local naming coordinates;*
2. *classifier transport: classification transports every displayed* $\mathsf{Real}^{\uparrow}$ *endpoint,* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation or order row,* $\mathsf{Rat}^{\uparrow}$ *source row,* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row, and local ledger row componentwise through the stored* hsame *and* Cont *routes;*
3. *field-row stability: addition, negation, multiplication, unit, zero, and absolute-value reads remain inside the* $\mathsf{RealAlgOrder}^{\uparrow}$ *coordinate* A *after endpoint replacement by classified* $\mathsf{Real}^{\uparrow}$ *rows;*
4. *Archimedean comparison stability: every rational-bound or rational-embedding comparison exposed by* O *and* Q *remains tied to the same displayed real endpoints and the same local ledger rows after transport;*
5. *ledger exactness: every downstream ordered-field or Archimedean read is one of the displayed rows* $R, A, Q, O, B, H, C, P, N$ *before any consumer receives it.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldNameCertObligations`

Proof. Carrier admission is projection from Definition 1419.1. The real endpoint family, algebra/order row, rational source family, Archimedean bound row, local comparison ledger, transport table, continuation table, provenance row, and local naming row are exactly the listed coordinates.

Classifier transport is componentwise. The $\mathsf{Real}^{\uparrow}$ endpoints move with their $\mathsf{RealAlgOrder}^{\uparrow}$ operation and order rows, while the rational source rows and Archimedean bound rows move through their own displayed sibling classifiers and the stored hsame routes.

The field-row and Archimedean stability clauses then repack the transported rows into the same ledger shape B. Algebraic reads use only A and the endpoint rows in R; rational-bound reads use only Q, O and the endpoint rows in R.

Ledger exactness follows because every public read is a coordinate of F or a row of B generated from those coordinates. Since no coordinate stores trichotomy, quotient real equality, a host ordered-field carrier, quotient reals, selected least bounds, or bridge-checked data, none of those principles is exported by the local certificate. □

Theorem 1393.3 (ArchimedeanOrderedField RealAlgOrder handoff)**.** *Every ordered-field or Archimedean comparison read exported by an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet factors through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row* A *and the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *row* O *over the displayed* $\mathsf{Real}^{\uparrow}$ *endpoints* R*. The* $\mathsf{RealAlgOrder}^{\uparrow}$ *side supplies the finite operation and order rows, the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *side supplies explicit rational-bound rows, and the* $\mathsf{Rat}^{\uparrow}$ *coordinate* Q *supplies the rational source data used by both comparison routes.*

Consequently the packet is a handoff between the existing real algebra/order and Archimedean-bound surfaces. It does not identify those surfaces with a host ordered field, quotient-reals carrier, complete real field, trichotomy principle, selected least-bound operation, or bridge-checked standard structure.

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRealAlgOrderHandoff`

Proof. Project the carrier tuple $F = (R, A, Q, O, B, H, C, P, N)$. The only operation and order coordinate is A, and the only Archimedean-bound coordinate is O.

Read the algebra/order route first. By Definition 766.2, the rows exposed by A are finite endpoint, operation, order, apartness, transport, continuation, provenance, and local naming rows over displayed $\mathsf{Real}^{\uparrow}$ data.

Read the Archimedean route next. By Definition 430.2, the rows exposed by O are finite rational-bound, dyadic-bound, stream-window, regular-Cauchy handoff, ledger, transport, continuation, provenance, and local naming rows for displayed $\mathsf{Real}^{\uparrow}$ data.

The rational source coordinate Q is shared by the two routes. The local ledger B records only those field-operation, order, rational-embedding, and rational-bound reads, and the transports H, C replay them componentwise for consumers.

Thus every exported comparison factors through A, O, and Q over R. The excluded host ordered-field, quotient-reals, trichotomy, completeness, least-bound, and bridge-checked objects have no carrier coordinate and no route through the local ledger. □

Theorem 1393.4 (ArchimedeanOrderedField rational-bound window)**.** *Every Archimedean comparison read exported by an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet must expose the* $\mathsf{Rat}^{\uparrow}$ *source row* Q*, the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row* O*, and the displayed* $\mathsf{Real}^{\uparrow}$ *endpoints* R *before any ordered-field consumer reads the local comparison ledger* B*. The rational-bound window is therefore the finite route*

$$Q, O, R \longmapsto B.$$

It is not a route through trichotomy, quotient real equality, selected least bounds, a host ordered-field carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRationalBoundWindow`

Proof. Project the carrier $F = (R, A, Q, O, B, H, C, P, N)$. The rational source row Q, Archimedean bound row O, endpoint family R, and comparison ledger B are displayed coordinates of the same finite packet.

Apply the NameCert obligations of Definition 1419.2. Classifier transport keeps the rational source, bound, endpoint, and ledger rows componentwise aligned, while Archimedean comparison stability ties every rational-bound read to Q, O over the same endpoints R.

Finally use the RealAlgOrder handoff of Definition 1419.3. Ordered-field reads may consume B only after the displayed algebra/order route and the rational-bound route have been replayed, and the carrier has no coordinate for the excluded trichotomy, quotient, least-bound, host-field, or completeness data. □

Theorem 1393.5 (ArchimedeanOrderedField comparison ledger exactness)**.** *For an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet with carrier*

$$F = (R, A, Q, O, B, H, C, P, N),$$

every ordered-field, Archimedean comparison, or rational-embedding consumer of the local comparison ledger B *reads* B *only through the displayed route* R, A, Q, O, H, C, P, N*. Here* A *is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row,* O *is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *row,* Q *is the* $\mathsf{Rat}^{\uparrow}$ *source row, and* R *is the* $\mathsf{Real}^{\uparrow}$ *endpoint family.*

Thus the ledger is an exhaustive consumer interface for those comparison reads, not a channel for trichotomy, quotient reals, selected least bounds, host ordered-field carriers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier definition of Definition 1419.1. The only coordinates adjacent to the ledger are the endpoint family R, the algebra/order row A, the rational source row Q, the Archimedean bound row O, the transport table H, the continuation routes C, the provenance row P, and the local naming row N.

Apply the NameCert obligations of Definition 1419.2. Carrier admission and classifier transport force every consumer-visible ledger row to be one of the displayed rows of F.

Field-row stability and Archimedean comparison stability keep ordered-field, rational-embedding, and rational-bound reads attached to the same R, A, Q, O coordinates.

The handoff theorem Definition 1419.3 identifies the ordered-field side with A over R and the Archimedean side with O and Q over the same endpoints. The rational-bound window of Definition 1419.4 then forces any Archimedean comparison read to pass through Q, O, R before it reaches B.

It follows that a downstream ordered-field, Archimedean comparison, or rational-embedding consumer has no remaining entry point into B except the route R, A, Q, O, H, C, P, N. Since the carrier and the cited obligations store no trichotomy, quotient-reals construction, selected least-bound operation, host ordered-field carrier, or ambient real-completeness coordinate, the ledger cannot export any of those structures. □

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldComparisonLedgerExactness`

Theorem 1393.6 (ArchimedeanOrderedField root obligation package)**.** *The carrier of Definition 1419.1, the NameCert obligations of Definition 1419.2, and the RealAlgOrder handoff of Definition 1419.3 assemble the root package consumed by* $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealAlgOrder}^{\uparrow}$. *The package consists of displayed ordered-field carrier rows, rational-bound rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, comparison transport, displayed replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrdere}}$ *row.*

The package exports only those rows. It does not supply trichotomy, quotient equality of real names, selected rational bounds, a selected least-bound operation, a host ordered-field carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRootObligationPackage`

Proof. Project the packet $F = (R, A, Q, O, B, H, C, P, N)$ from Definition 1419.1. The rows R, A, Q, O, B are respectively the real endpoint source, ordered algebra row, rational source family, Archimedean rational-bound row, and local comparison ledger; the rows H, C, P, N transport, replay, source, and name that finite data.

Apply Definition 1419.2. Its carrier-admission, classifier-transport, field-row stability, Archimedean comparison stability, and ledger-exactness clauses identify every root obligation with one of the displayed coordinates of F.

Apply Definition 1419.3. It routes ordered-field and Archimedean comparison reads through A, O, and Q over the same displayed $\mathsf{Real}^{\uparrow}$ endpoints R, so downstream $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealAlgOrder}^{\uparrow}$ consumers receive a single finite package.

Finally inspect the absent coordinates named in the carrier and handoff statements. Since no row stores trichotomy, quotient equality, selected bounds, host ordered-field structure, least-bound selection, or ambient completeness, the assembled root package cannot export any of those principles. □

Theorem 1393.7 (ArchimedeanOrderedField scoped closure)**.** *The carrier of Definition 1419.1, the NameCert obligations of Definition 1419.2, the RealAlgOrder handoff of Definition 1419.3, the*

rational-bound window of Definition 1419.4, comparison ledger exactness of Definition 1419.5, and the root obligation package of Definition 1419.6 bind every public comparison read to the displayed RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldScopedClosure`

Proof. The carrier identifies the only public rows of an accepted packet:

$$F = (R, A, Q, O, B, H, C, P, N).$$

Thus every comparison read must begin with the displayed real endpoints, RealAlgOrder row, rational source row, Archimedean bound row, and local comparison ledger.

The NameCert obligations keep those rows stable under classifier transport. Field-row stability routes algebra and order reads through A over R, while Archimedean comparison stability routes rational-bound reads through Q and O over the same endpoints.

The RealAlgOrder handoff, rational-bound window, and comparison ledger exactness then close the consumer boundary. Ordered-field, rational-embedding, and Archimedean comparison reads reach B only through R, A, Q, O, H, C, P, N.

Finally the root obligation package repacks the same rows for RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, and Real$^{\uparrow}$ consumers. Since no coordinate remains outside the displayed transport, replay, provenance, and local naming rows, the chapter closes exactly this scoped comparison interface and no stronger ordered-field or completeness principle. □

Closure status for NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$.

Constructive story: An ArchimedeanOrderedField$^{\uparrow}$ packet is generated from displayed Real$^{\uparrow}$ endpoint rows, a RealAlgOrder$^{\uparrow}$ algebra/order row, Rat$^{\uparrow}$ rational source rows, an ArchimedeanReal$^{\uparrow}$ rational-bound row, a finite comparison ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1419.7 binds every public comparison read to the displayed RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completeness, trichotomy, quotient reals, selected least bounds, host ordered-field structure, or bridge-checked status.

Upgrade path: Scoped closure requires a checked finite carrier theorem binding the displayed RealAlgOrder and ArchimedeanReal rows to the local comparison ledger.

Chapter 1394

A Concrete Naming Certificate for $\mathsf{LocatedLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedLimit}^{\uparrow}$ packet records located limits as explicit modulus-and-window data for displayed real-name sequences. It is a finite route from the Cauchy-equivalence surface of chapter 1412, the regular-rational readback surface of chapter 301, the finite stream-window surface of chapter 126, and the dyadic tolerance ledger of chapter 291 to the real-name seal of chapter 38. The packet does not introduce host real equality, quotient limits, countable choice, trichotomy, or ambient completeness.

Definition 1394.1 (LocatedLimit carrier)**.** *A* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, M, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the displayed* $\mathsf{Real}^{\uparrow}$ *sequence surface whose entries are read only through named finite windows;*
2. *M is the Cauchy/modulus ledger recording the explicit tolerance budget for the displayed sequence;*
3. *T is the located finite-window schedule that says which sequence windows and dyadic tolerances are inspected for each requested precision;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled windows and the modulus ledger;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *limit seal reached through Q and through no independent host-limit coordinate;*
6. *H stores componentwise* hsame *transports for the sequence surface, modulus ledger, finite-window schedule, regular readback, and real seal;*
7. *C stores the displayed* Cont *replay routes by which consumers reread the same windows and tolerances;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected host limit, quotient equality of streams, countable-choice schedule, real trichotomy, or ambient completeness principle.

Lean4 (definition) `BEDC.Derived.LocatedLimitUp.LocatedLimitUp`

Definition 1394.2 (LocatedLimit classifier)**.** *Two accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *packets*

$$L = (S, M, T, Q, E, H, C, P, N) \quad \textit{and} \quad L' = (S', M', T', Q', E', H', C', P', N')$$

are classified together exactly when their displayed finite-window schedules expose the same dyadic tolerance comparisons after componentwise transport. Thus the classifier compares L *and* L' *only by shared windows in* T, T'*, the dyadic budgets read by* M, M'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q'*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows* E, E' *reached from those data.*

No classifier step may use host real equality, quotient equality of streams, a selected limiting subsequence, countable choice, trichotomy, or an ambient completeness row.

Theorem 1394.3 (LocatedLimit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *surface for the carrier of Definition 1420.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet* $L = (S, M, T, Q, E, H, C, P, N)$*;*
2. *classifier: comparison is the finite-window and dyadic-tolerance classifier of Definition 1420.2;*
3. *transport:* H *and* C *preserve the same sequence surface, modulus ledger, schedule, readback, and seal rows under accepted consumer motion;*
4. *modulus exactness: every limit-facing tolerance request is answered by the stored Cauchy/modulus ledger* M *together with the schedule* T*;*
5. *Real-seal route: every* $\mathsf{Real}^{\uparrow}$ *limit read factors through* Q, E, H, C, P, N *after the displayed rows* S, M, T *have been read.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitNameCertObligations`
↪ variant `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier namecert obligations`

Proof. Carrier admission is projection from the packet $L = (S, M, T, Q, E, H, C, P, N)$. The sequence surface, modulus ledger, located schedule, regular readback, real seal, transport rows, replay routes, provenance, and local naming row are exactly the coordinates named by the carrier.

The classifier row is finite because it reads only the windows named by T, T' and the dyadic tolerances supplied by M, M'. The regular readbacks Q, Q' and real seals E, E' are compared only after those finite observations have been transported.

For transport, use H to move the displayed rows componentwise and C to replay the same consumer route. The replay never creates a new modulus, a host limit, or a quotient of streams; it returns to the stored rows.

Modulus exactness follows because the only tolerance ledger in the carrier is M, and the only located schedule is T. Therefore every downstream tolerance request has to pass through M and T before it can reach Q or E.

The Real-seal route is then the visible composition from S, M, T to Q and finally to E, with H, C, P, N carrying transport, replay, provenance, and naming. The excluded completeness, choice, trichotomy, quotient, and host-equality principles have no carrier coordinate. □

Theorem 1394.4 (LocatedLimit Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *limit read exported by an accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier factors through the displayed route*

$$S, M, T, Q, E, H, C, P, N.$$

The sequence surface S supplies the real-name entries, the ledger M supplies the explicit Cauchy/modulus budget, the schedule T chooses only finite windows for each dyadic tolerance, the row Q repacks those observations as a $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the row E is the sole* $\mathsf{Real}^{\uparrow}$ *limit seal available to consumers. The rows H, C, P, N provide transport, replay, provenance, and local naming for the same route.*

Proof. Project the accepted carrier. Its real-facing coordinates are not free-standing; E is stored together with the sequence surface S, the modulus ledger M, the located schedule T, and the regular readback Q.

Read the limit request in finite order. A consumer first asks T for the displayed windows attached to the requested dyadic tolerance, then reads the matching budget in M, and then uses Q to view the resulting data as a regular rational name.

Only after that route is visible can the consumer read E. The transport and replay rows H and C preserve the same route, while P and N record provenance and local naming.

Thus every $\mathsf{Real}^{\uparrow}$ limit read factors through $S, M, T, Q, E, H, C, P, N$. Since no other coordinate exists, the packet exports no real completeness theorem, selected host limit, quotient equality of streams, countable-choice schedule, trichotomy, or bridge-checked standard interpretation. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimit real seal route`

Theorem 1394.5 (LocatedLimit window schedule exactness)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier $L = (S, M, T, Q, E, H, C, P, N)$, each limit-facing tolerance request reads the displayed schedule row T and the modulus ledger M before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q or the* $\mathsf{Real}^{\uparrow}$ *seal row E. The exactness claim is the finite routing statement of Definition 1420.1 and Definition 1420.2, with the obligation order supplied by Definition 1420.3 and the terminal route supplied by Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitWindowScheduleExactness`

Proof. Project the carrier of Definition 1420.1. The only rows that can answer a tolerance request before readback are the schedule row T and the modulus ledger M.

Use the classifier of Definition 1420.2 to see that comparison is by shared finite windows and dyadic budgets, not by a host limit. Then apply Definition 1420.3: modulus exactness is one of the named local obligations and it places M and T before every read of Q or E.

Finally apply the route of Definition 1420.4. Since E is reached only after S, M, T, Q, every tolerance request has already passed through T and M before any regular readback or real seal can be exported. □

Theorem 1394.6 (LocatedLimit finite-window scope)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier $L = (S, M, T, Q, E, H, C, P, N)$, each finite-window tolerance request is scoped exactly by the* $\mathsf{StreamName}^{\uparrow}$ *window surface S, T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger M, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *row N. The scope is the source-row route named by the carrier, classifier, NameCert obligations, window-schedule exactness, and Real-seal route; it excludes quotient streams, countable choice, trichotomy, selected host limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier finite window scope`

Proof. Project the carrier of Definition 1420.1. Its source rows are precisely $S, M, T, Q, E, H, C, P, N$, so a tolerance request has no access to any row outside that finite packet.

Use the classifier of Definition 1420.2. It compares accepted packets only through transported finite windows and dyadic budgets, and therefore binds the request to the $\mathsf{StreamName}^{\uparrow}$ surface S, T and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger M before any readback is visible.

Apply Definition 1420.3 and Definition 1420.5. The obligation rows require modulus exactness and route every tolerance request through T and M before it reaches the $\mathsf{RegSeqRat}^{\uparrow}$ row Q or the $\mathsf{Real}^{\uparrow}$ seal E.

Finally use Definition 1420.4. The terminal route is $S, M, T, Q, E, H, C, P, N$, with H, C, P, N supplying componentwise transport, replay, provenance, and local naming. Quotient streams, countable choice, trichotomy, selected host limits, and ambient completeness have no coordinate in that route, so they are outside the finite-window scope. □

Theorem 1394.7 (LocatedLimit schedule monotonicity)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, enlarging a displayed precision request refines the stored finite-window schedule through the same modulus ledger* M *and schedule row* T*. The refined request may expose later sequence windows, stronger dyadic tolerances, and a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, but it may not replace* M*, allocate a new schedule row, or bypass the Real-seal route of Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier schedule monotonicity`

Proof. Project the carrier of Definition 1420.1. The only rows that answer precision requests before readback are the modulus ledger M and the finite-window schedule T.

Use Definition 1420.2 to compare the original request and the enlarged request through transported finite windows and dyadic budgets. The classifier has no host limit or quotient-stream coordinate, so the comparison remains inside M and T.

Apply Definition 1420.3 and Definition 1420.5. Their modulus exactness clause places M, T before every read of Q or E, and the exactness theorem identifies this placement as the stored route for every displayed tolerance request.

Finally use Definition 1420.4. Since the Real seal is reached only after S, M, T, Q, a stronger request can only refine the visible windows and tolerances in the same schedule; it cannot introduce a separate modulus or an independent seal. □

Theorem 1394.8 (LocatedLimit schedule-tail cofinality)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, any later displayed precision request that refines the same located schedule must reuse the same modulus ledger* M*, schedule row* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The refined tail may ask for later sequence windows and stronger dyadic tolerances, but it cannot allocate a second modulus, quotient stream, selected limit, choice schedule, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitScheduleTailCofinality`
↪ variant `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier schedule tail cofinality`

Proof. Project the accepted packet through Definition 1420.1. The only tolerance-facing rows are M and T, and the only readback and terminal seal rows available after them are Q and E.

Use Definition 1420.2 to read a refined request. Since the classifier compares accepted packets by transported finite windows and dyadic budgets, a later tail that refines the same schedule is still classified through the same displayed rows rather than through a fresh host limit or quotient stream.

Apply Definition 1420.5. Every tolerance request reaches Q or E only after it has already passed through the stored schedule row T and modulus ledger M, so a tail request has no alternative ledger from which to obtain its budget.

Finally apply Definition 1420.4. The public limit read is exhausted by the route $S, M, T, Q, E, H, C, P, N$, hence the refined tail can only continue along that route. The excluded second modulus, quotient stream, selected limit, choice schedule, and ambient completeness row have no carrier coordinate. □

Theorem 1394.9 (LocatedLimit Real seal comparison)**.** *Let $L = (S, M, T, Q, E, H, C, P, N)$ and $L' = (S', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carriers. If their transported schedule rows, modulus ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows match under the displayed* hsame *and* Cont *routes, then any comparison between the two accepted Real seals is exhausted by those rows. No comparison may appeal to host real equality, quotient equality of streams, a selected limit, countable choice, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier real seal comparison`

Proof. Read both carriers through Definition 1420.1. The only comparison-facing rows are the sequence surfaces, modulus ledgers, schedule rows, regular readbacks, Real seals, transport rows, replay rows, provenance rows, and local naming rows.

Apply the classifier of Definition 1420.2. It compares accepted carriers through shared finite windows, dyadic budgets, readback rows, and Real seal rows after componentwise transport; it does not expose host real equality or quotient stream equality.

Use Definition 1420.3 to force the comparison through the transported schedule and modulus rows before any readback or seal is consumed. Then Definition 1420.5 identifies the finite-window route as the exact route for each displayed precision request.

Finally apply Definition 1420.4. The terminal Real seal is reached only after S, M, T, Q on one side and S', M', T', Q' on the other. Therefore matching transported schedules, modulus ledgers, readbacks, and seal rows exhaust the accepted comparison, and the excluded external principles have no carrier coordinate from which to act. □

Theorem 1394.10 (LocatedLimit Cauchy tail uniqueness)**.** *Let $L = (S, M, T, Q, E, H, C, P, N)$ and $L' = (S', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carriers. Suppose the two carriers expose matching transported finite schedules T, T', modulus ledgers M, M',* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks Q, Q', and terminal* $\mathsf{Real}^{\uparrow}$ *seals E, E' through the displayed* hsame *and* Cont *routes. Then every limit-facing consumer comparison of E and E' is exhausted by the common Cauchy-tail route*

$$S, M, T, Q, E, H, C, P, N.$$

It cannot appeal to host real equality, quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier cauchy tail uniqueness`

Proof. Project both accepted carriers through Definition 1420.1. The only rows that can feed a limit-facing comparison are the displayed sequence surfaces, modulus ledgers, finite schedules, regular readbacks, Real seals, transport rows, replay rows, provenance rows, and local naming rows.

Use Definition 1420.2 to identify the comparison surface. The classifier reads the transported windows T, T', the dyadic budgets M, M', the readbacks Q, Q', and the seals E, E'; it has no

coordinate for host equality, quotient streams, selected limits, countable choice, trichotomy, or ambient completeness.

Apply Definition 1420.5. Every tolerance request reaches Q or E only after the finite schedule and modulus ledger have already been read, so the two matching schedule-ledger pairs determine the Cauchy-tail route visible to the consumer.

Finally use Definition 1420.4 and Definition 1420.9. The terminal seal comparison is reached only through the same finite route, and the comparison theorem says matching transported schedules, ledgers, readbacks, and seals exhaust the accepted Real-seal comparison. Hence the two seals are unique for this accepted route in the stated Cauchy-tail sense, without invoking any ambient Real-completeness principle. □

Theorem 1394.11 (LocatedLimit RegSeqRat readback stability)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is stable under the same componentwise* hsame *transports and displayed* Cont *replay rows* H, C *that preserve the sequence surface* S*, modulus ledger* M*, located schedule* T*, and real seal* E*. The stability surface is the one described by Definition 1420.1, classified by Definition 1420.2, obligated by Definition 1420.3, and consumed by Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimit regseqrat readback stability`

Proof. Read the accepted packet through Definition 1420.1. The transport row H is componentwise and includes the regular readback row Q together with S, M, T, E.

The classifier in Definition 1420.2 compares only transported finite windows, dyadic budgets, regular readbacks, and real seals. Therefore a transported Q is not compared through quotient stream equality or through a new host sequence.

Apply the transport obligation in Definition 1420.3. The replay row C rereads the same consumer route after H has moved the displayed coordinates. The Real-seal route of Definition 1420.4 then consumes exactly that stable Q before reaching E. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier regseqrat readback stability`

Theorem 1394.12 (LocatedLimit source-row scope)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, every limit-facing consumer binds the finite-window schedule* T*, modulus ledger* M*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *to the source rows named by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *before the seal can be read. The binding is exactly the source-row route of the carrier, classifier, NameCert obligations, window schedule exactness, Real seal route, and RegSeqRat readback stability.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier source row scope`

Proof. Project the carrier of Definition 1420.1. Its consumer-facing rows are $S, M, T, Q, E, H, C, P, N$, so the schedule, modulus, readback, and seal are already stored with their transport, replay, provenance, and local naming rows.

Use the classifier of Definition 1420.2. It compares accepted packets only through transported finite windows, dyadic budgets, regular readbacks, and Real seals; hence T, M, Q, E cannot be detached from the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source rows that classify them.

Apply Definition 1420.3, Definition 1420.5, and Definition 1420.4. These rows force every tolerance request through T and M before Q and E are consumed.

Finally use Definition 1420.11. The readback row Q is stable only under the same hsame and Cont rows that preserve S, M, T, E, and P, N record the same source route. Therefore no consumer can read the seal before these BEDC source rows are bound. □

Theorem 1394.13 (LocatedLimit Real seal nonescape)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *exports no selected host limit, no quotient stream equality, no countable-choice schedule, no trichotomy, and no ambient completeness principle. This nonescape is the carrier exclusion of Definition 1420.1, the classifier exclusion of Definition 1420.2, the Real-seal obligation of Definition 1420.3, and the terminal factorization of Definition 1420.4.*

Proof. Inspect the carrier in Definition 1420.1. The rows are $S, M, T, Q, E, H, C, P, N$, and none is a selected host limit, quotient equality of streams, countable-choice schedule, trichotomy witness, or ambient completeness field.

The classifier of Definition 1420.2 preserves the same boundary: it compares finite windows, dyadic budgets, regular readbacks, and real seals only after componentwise transport.

Apply Definition 1420.3 to force every real-facing read through the named finite rows. Then Definition 1420.4 identifies E as the sole Real seal reached after S, M, T, Q. Since the excluded principles have no carrier coordinate and no classifier clause, they cannot escape through E. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitRealSealNonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedLimit}^{\uparrow}$ packet is generated from a displayed $\mathsf{Real}^{\uparrow}$ sequence surface, an explicit Cauchy/modulus ledger, a located finite-window schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ limit seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1420.1, Definition 1420.2, Definition 1420.3, Definition 1420.5, Definition 1420.4, Definition 1420.7, Definition 1420.8, Definition 1420.11, Definition 1420.12, and Definition 1420.13 close the mature paper-side finite-window treatment: carrier, classifier, NameCert obligations, window schedule exactness, Real-seal route, schedule monotonicity, tail cofinality, readback stability, source-row scope, and nonescape for located-limit consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, selected host limits, quotient equality of streams, countable choice, trichotomy, or bridge checking.

Upgrade path: The next upgrade is Lean bridge checking for the paper-side located-limit route without adding Real completeness or selected limits.

Chapter 1395

A Concrete Naming Certificate for $\mathsf{IntermediateValueBisection}^{\uparrow}$

Following Bishop and Bridges-style constructive bisection, the $\mathsf{IntermediateValueBisection}^{\uparrow}$ packet records the finite sign-change route that turns an inspected located interval, endpoint sign reads, and a displayed continuity budget into a regular rational window and a real-name seal. It is a NameCert consumer of the dyadic branch ledger of chapter 361, the rational endpoint cells of chapter 354, the located interval surface of chapter 394, the stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38.

The certificate is narrower than a classical intermediate value theorem. It stores only the finite branch prefix and the sign ledger needed by a consumer at a requested tolerance, then hands the retained window to the existing regular-sequence and real-name surfaces. No coordinate stores a completed root, a quotient stream, an arbitrary limit selector, real trichotomy, open-cover compactness, or an ambient completeness theorem.

Theorem 1395.1 (IntermediateValueBisection finite sign-change carrier)**.** *An* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (J, R, D, S, W, G, E, H, C, P, N)$$

whose visible rows have the following roles:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, including the lower and upper rational cells and their located order witness;*
2. *R is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger against which endpoint and midpoint sign reads are recorded;*
3. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix that retains an interval with opposite displayed signs at the requested finite precision;*
4. *S is the sign ledger, containing the initial endpoint signs and the midpoint sign row used at each branch of D;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window surface for the retained lower and upper endpoint names;*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained stream windows and the requested dyadic tolerance;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal available to downstream consumers only through the retained regular readback;*

8. *H stores the componentwise* hsame *transports for J, R, D, S, W, G, E, and the local certificate row;*
9. *C stores the displayed* Cont *replay routes for endpoint inspection, midpoint inspection, branch retention, regular readback, and real-seal consumption;*
10. *P is the* Pkg *provenance over* LocatedInterval$^{\uparrow}$, RationalInterval$^{\uparrow}$, DyadicBisection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{IntermediateValueBisection}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite sign-change bisection certificate. It has no coordinate for completed root existence, real trichotomy, quotient streams, selected limits, open-cover compactness, or a global completeness principle for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntermediateValueBisectionUp.IntermediateValueBisectionTasteGate single carrier alignment`

Proof. The carrier is read by projection from the displayed packet I. The located interval row J and rational cell ledger R provide the endpoint cells in which signs are inspected.

The dyadic branch prefix D is the only row that can choose a retained interval. Its choices are justified by the sign ledger S, so every branch step is tied to an endpoint or midpoint sign row already present in the packet.

The retained endpoints then expose stream windows through W and regular rational readback through G. The real-facing row E is reached only after that readback route is fixed.

Finally, H, C, P, and N provide transport, replay, provenance, and local naming for the same rows. Since the tuple contains no other analytic coordinate, the excluded classical and completeness principles cannot be exported by the carrier. □

Theorem 1395.2 (IntermediateValueBisection sign-ledger stability)**.** *For every accepted* IntermediateValueBisection *carrier $I = (J, R, D, S, W, G, E, H, C, P, N)$, every branch prefix of D transports the same sign-ledger boundary under the displayed* hsame *rows and the displayed* Cont *replay routes. In particular, if a consumer reads a retained subinterval through C, then the lower and upper sign rows it receives are exactly the transported entries of S attached to the rational cells in R and the located endpoint packet J.*

Proof. Start with a branch prefix of D. Each step in the prefix names one retained dyadic subinterval and one midpoint sign row, so the only available sign data for the next step are the entries already stored in S.

Transport along H acts componentwise on J, R, D, and S. It preserves the attachment between a rational endpoint cell and its displayed sign row because there is no separate sign coordinate outside the ledger.

Replay along C reads the retained subinterval through the same endpoint inspection and midpoint inspection route. Thus a consumer cannot replace the lower or upper sign by an unrelated host comparison; it receives the transported ledger entries bound to the branch prefix.

Package provenance P and the local certificate row N record that this is the accepted IntermediateValueBisection$^{\uparrow}$ route. The branch-prefix and sign-ledger transport are therefore stable under the named hsame, Cont, and Pkg rows. □

Lean4 ✓ `BEDC.Derived.IntermediateValueBisectionUp.IntermediateValueBisectionCarrier sign ledger stability`

Theorem 1395.3 (IntermediateValueBisection regular seal handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *carrier factors through the route*

$$J, R, D, S, W, G, E, H, C, P, N.$$

The consumer first reads the located and rational endpoint cells, then the finite dyadic branch prefix and transported sign ledger, then the retained stream windows, then the regular rational readback, and only then the real-name seal. The handoff exports a finite sign-change window, not a completed root, quotient stream, selected limit, real trichotomy, open-cover compactness, or ambient completeness theorem.

Proof. Project the accepted carrier. A regular-sequence request begins at the retained endpoint windows, but those windows are not independent rows; they are the W coordinate stored with J, R, D, and S.

Read the finite route in order. The located endpoint packet and rational interval cells determine the initial boundary, the dyadic branch prefix selects the retained interval, and Definition 1422.2 keeps the sign ledger attached to that prefix under transport and replay.

The retained windows then pass through G, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The $\mathsf{Real}^{\uparrow}$ seal E is exposed only after this regular readback has been reached, with H, C, P, and N preserving the same route for consumers.

No step in this route introduces a host root or a completed intersection object. Therefore the handoff supplies exactly the finite regular window and real-name seal named in the packet, with the listed non-claims remaining outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntermediateValueBisection}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint packet, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicBisection}^{\uparrow}$ branch prefix, finite sign rows, $\mathsf{StreamName}^{\uparrow}$ retained windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1422.1, Definition 1422.2, and Definition 1422.3 seed the finite constructive bisection route from displayed sign change to regular rational and real-name consumption.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close completed root existence, real trichotomy, quotient streams, selected limits, open-cover compactness, completed nested intersections, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: A checked BEDC.Derived.IntermediateValueBisectionUp carrier and theorem surface matching the displayed finite sign-change packet would raise the verification axis.

Chapter 1396

A Concrete Naming Certificate for $\mathsf{RealCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCauchyModulus}^{\uparrow}$ packet records the explicit modulus-controlled read by which a regular rational Cauchy name reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite stream-window surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal boundary of chapter 38, and the generic modulus schedule of chapter 292. The packet keeps the modulus visible as finite certificate data before any Real-facing consumer may read the seal.

Definition 1396.1 (Real Cauchy modulus carrier)**.** *A* $\mathsf{RealCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the precision-indexed* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule assigning each requested dyadic tolerance a finite threshold.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *family of finite windows selected by the thresholds in M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the endpoint and radius comparisons read on those windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose regularity is witnessed only through M, S, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the displayed modulus, window, dyadic, and regular-readback rows have been consumed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic tolerance row, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C records the displayed* Cont *replay from each precision request through M, S, D, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, a selected limit, open-cover compactness, or an unbounded stream read.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier local cert ledger transport`

Theorem 1396.2 (Real Cauchy modulus NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ *surface for Definition 1423.1 consists exactly of carrier exposure for B, precision/modulus exposure through M, finite-window exposure through S, dyadic tolerance exposure through D, regular rational readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Each accepted precision request is classified only by the scheduled threshold in M, the finite windows in S, and the tolerance ledger in D.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier namecert obligations`

Proof. Project the carrier tuple $B = (M, S, D, Q, E, H, C, P, N)$. Each coordinate is a displayed finite row, and the final coordinate names the packet as the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ surface.

Read the analytic rows in order. The precision request first consumes the threshold row M, then the finite windows S, then the dyadic ledger D, and only then the regular readback Q and terminal seal E.

The structural rows do not add an analytic source. They transport, replay, source, and name the same route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface. □

Theorem 1396.3 (Real Cauchy modulus finite-window route)**.** *For every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet $B = (M, S, D, Q, E, H, C, P, N)$ and every displayed precision request, the Real-facing read factors through the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The consumer receives no Real completeness theorem, quotient stream equality, host real equality, countable-choice row, trichotomy row, selected-limit row, open-cover compactness row, hidden selector, or detached Real seal.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.TasteGate.RealCauchyModulusCarrier window modulus route`

Proof. Start with Definition 1423.2. It places the modulus schedule M before the StreamName window S and the dyadic tolerance ledger D.

The regular readback is then forced through the same finite window. The row Q is read only after M, S, D are displayed, so no quotient stream or host equality row can substitute for the scheduled window comparison.

Finally read the Real seal coordinate. The row E is reached after Q, and the rows H, C, P, N only preserve and name that route. This exhausts the advertised finite-window path for the precision request. □

Theorem 1396.4 (Real Cauchy modulus Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RealCauchyModulus}^{\uparrow}$ *carrier can bypass the displayed route*

$$\mathsf{CauchyModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Equivalently, every accepted terminal seal read is attached to the same packet $B = (M, S, D, Q, E, H, C, P, N)$ and cannot be separated from its explicit modulus schedule, finite windows, dyadic tolerance ledger, regular readback, transport, continuation, provenance, and local naming rows.

Proof. Use the finite-window route of Definition 1423.3. It already forces each terminal seal read through M, S, D, Q, E before the structural rows are consumed.

Now inspect the carrier of Definition 1423.1. There is no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, a selected limit, open-cover compactness, a hidden selector, or a detached Real seal.

Thus any purported Real seal read that omits one of the displayed rows is not a read of the accepted carrier. The only terminal route available to a consumer is the one named by B. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier real seal nonescape`

Definition 1396.5 (Real Cauchy modulus classifier)**.** *The* $\mathsf{RealCauchyModulus}^{\uparrow}$ *classifier accepts a packet* $B = (M, S, D, Q, E, H, C, P, N)$ *exactly when it is componentwise* hsame *to a displayed carrier packet*

$$B' = (M', S', D', Q', E', H', C', P', N')$$

with matching rows for the $\mathsf{CauchyModulus}^{\uparrow}$ *schedule,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming certificate. The classifier then replays the same* Cont *route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N)$$

and accepts no packet whose Real-facing read omits one of these coordinates.

Lean4 (definition) `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusClassifier`

Theorem 1396.6 (Real Cauchy modulus threshold stability)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Transporting* B *along the classifier of Definition 1423.5 preserves the scheduled threshold in* M*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row in* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal in* E*. In particular, the transported packet has the same displayed Real-facing route and cannot replace the modulus threshold by an unbounded stream read or a detached Real seal.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier threshold stability`

Proof. Project the accepted packet by Definition 1423.1. The rows M, S, D, Q, E are named coordinates of the same finite BHist tuple, while H records the componentwise hsame transport for those coordinates.

Apply the classifier of Definition 1423.5. Acceptance requires the transported packet to match the displayed carrier component by component, so the threshold row, finite windows, dyadic tolerance ledger, regular readback, and terminal seal are preserved as the rows read by the consumer.

Finally replay the finite-window route of Definition 1423.3. Since the Cont replay is the same route through $M, S, D, Q, E, H, C, P, N$, no unbounded stream read, hidden selector, or detached Real seal can enter during transport. □

Theorem 1396.7 (Real Cauchy modulus ledger exhaustion)**.** *Every accepted Real-facing read of a* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet is exhausted by the nine displayed rows* $M, S, D, Q, E, H, C, P, N$*. The read therefore cannot introduce Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, an unbounded stream read, or any analytic row not already present in the carrier.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier ledger exhaustion`

Proof. Start from the obligation surface in Definition 1423.2. It lists the analytic rows M, S, D, Q, E and the structural rows H, C, P, N as the complete seed surface for the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ packet.

Use Definition 1423.6 to transport any accepted read back to the same classifier route. The transported read still consumes the scheduled threshold, finite windows, tolerance ledger, regular readback, and terminal seal in the displayed order.

It remains to inspect the carrier. By Definition 1423.1, none of Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, an unbounded stream read, or a detached analytic source has a coordinate in B. Hence every accepted Real-facing read is exhausted by the displayed ledger. □

Theorem 1396.8 (Real Cauchy modulus root-unblock scope)**.** *Every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet exposes exactly the rows needed by downstream L10 Real-and-Completion consumers: the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the ambient* BHist *carrier, the* hsame *transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*. No accepted packet exposes a quotient stream, selected limit, host real equality, countable-choice row, or ambient Real completeness row as part of the unblock surface.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier root unblock scope`

Proof. Project the tuple in Definition 1423.1. Its analytic coordinates are precisely M, S, D, Q, E, and its structural coordinates are precisely H, C, P, N over the finite BHist packet.

The obligation surface of Definition 1423.2 names the same coordinates as the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ surface. Thus an L10 consumer can read the schedule, window, dyadic, regular-readback, seal, transport, continuation, package, and naming rows without adding any further analytic source.

Finally apply Definition 1423.3 and Definition 1423.7. The route inspection shows that quotient streams, selected limits, host real equality, countable choice, and ambient Real completeness have no coordinate in the accepted packet, so they cannot be part of the root-unblock scope. □

Theorem 1396.9 (Real Cauchy modulus threshold-window totality)**.** *For every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet and every displayed precision request, the consumer first reads a threshold from* M *and then a finite window from* S *before the dyadic ledger* D*, the regular readback* Q*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is available. This ordering is total for the displayed route: there is no accepted access path from the precision request directly to* D, Q, E*, to a quotient stream, to a selected limit, to host real equality, to countable choice, or to ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier threshold window totality`

Proof. Start with the route theorem Definition 1423.3. It factors every accepted precision request through $M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$ before the structural rows are consumed.

The carrier projection in Definition 1423.1 makes the threshold and window rows named coordinates of the same finite packet. Therefore the consumer cannot obtain the dyadic tolerance ledger, regular readback, or terminal seal without first selecting the scheduled threshold and the corresponding finite window.

The NameCert obligations in Definition 1423.2 and the ledger exhaustion theorem Definition 1423.7 rule out all other sources. Since quotient streams, selected limits, host real equality, countable choice, and ambient Real completeness are absent from the tuple, they cannot provide an alternate totality witness. □

Theorem 1396.10 (Real Cauchy modulus RegSeqRat seal admission)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet is admitted to the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *only through the displayed modulus, finite-window, and dyadic route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

No quotient stream equality, selected limit, host real equality, countable-choice principle, or ambient Real completeness theorem can admit Q *to the seal.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier regseqrat seal admission`

Proof. Read the accepted packet through Definition 1423.1. The row Q is a displayed coordinate between the dyadic ledger D and the terminal seal E, not a detached regular sequence or an external limit object.

Use Definition 1423.3. It places Q after the scheduled threshold M, finite window S, and dyadic ledger D, and before the terminal $\mathsf{Real}^{\uparrow}$ seal row E. The continuation row C then replays exactly that admission path.

The obligation list in Definition 1423.2 supplies the same route as local certificate data, while Definition 1423.7 excludes every non-displayed analytic source. Hence Q enters the seal only by the modulus-window-dyadic route, never by quotient streams, selected limits, host real equality, countable choice, or ambient Real completeness. □

Theorem 1396.11 (Real Cauchy modulus precision induction)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For every finite unary precision request, every finite precision refinement of the accepted request is handled by induction on that unary request and preserves the displayed route through*

$$\mathsf{CauchyModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

At each induction stage the refined read consumes only the scheduled modulus row M*, the finite window row* S*, the dyadic ledger row* D*, the regular readback row* Q*, the terminal seal row* E*, and the structural rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier precision induction`

Proof. Project the accepted packet with Definition 1423.1. The precision request is a finite unary object, so the proof proceeds by induction on the displayed request.

In the zero stage, Definition 1423.3 gives the route from M to S, then to D, Q, and E. The classifier rows H, C, P, N replay and name the same route, so no extra analytic coordinate is introduced.

For the successor stage, assume the route has been preserved for the preceding precision request. The next refinement reads the next threshold from M, selects the corresponding finite window in S, extends the tolerance ledger in D, and then replays the regular readback Q before exposing the same terminal $\mathsf{Real}^{\uparrow}$ seal row E.

The replay is the displayed Cont row C, transported componentwise by H and sourced by P, N. The ledger exhaustion theorem Definition 1423.7 rules out Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, or an unbounded stream read at any induction stage. □

Theorem 1396.12 (Real Cauchy modulus window monotonicity)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If a dyadic tolerance request is strengthened inside the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the scheduled threshold in M may enlarge the finite* $\mathsf{StreamName}^{\uparrow}$ *window in S, but the earlier window, dyadic ledger row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row remain present in the accepted carrier route. In particular, the stronger request cannot delete or detach any previously scheduled window, tolerance entry, regular readback, or terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier window monotonicity`

Proof. Begin with the accepted route of Definition 1423.3. A stronger tolerance is still classified through the same M, S, D, Q, E route, because the classifier of Definition 1423.5 accepts only componentwise matches of those displayed rows.

The threshold selected by M may point to a larger finite window in S. This changes the amount of displayed stream data read for the stronger tolerance, but it does not remove the earlier finite window, since that earlier window is already a coordinate of the accepted BHist packet and is transported by H.

The dyadic ledger and regular readback follow the same containment. The row D records the earlier tolerance entry and the stronger tolerance entry as finite ledger data, while Q is read only after D has been displayed. Replaying C therefore keeps the earlier window, ledger, readback, and terminal seal available to every downstream consumer. □

Theorem 1396.13 (Real Cauchy modulus cofinal window extraction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For any two finite unary precision requests whose second request refines the first, the displayed modulus row M selects a finite* $\mathsf{StreamName}^{\uparrow}$ *window in S for the refined request that is cofinal over the earlier window. The extraction preserves the earlier* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger entry, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier cofinal window extraction`

Proof. Proceed by induction on the finite unary refinement. The base request is handled by threshold-window totality, Definition 1423.9, which forces the read through M and then the displayed finite window in S before D, Q, E are available.

For the refinement step, use precision induction from Definition 1423.11. The stronger request is still classified by the same carrier rows and therefore reads a scheduled threshold from M and a finite window from S.

Apply window monotonicity, Definition 1423.12. The refined window may enlarge the displayed StreamName data, but it preserves the earlier finite window, dyadic tolerance entry, regular readback, and terminal seal.

The continuation row C, transported by H and sourced by P, N, repacks this preserved route as one accepted carrier read. Thus the finite window extracted for the refined request is cofinal over the earlier one, with the tolerance ledger, readback, seal, and structural rows intact. □

Theorem 1396.14 (Real Cauchy modulus seal determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ and $B' = (M', S', D', Q', E', H', C', P', N')$ be accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packets. If their modulus schedules, stream windows, dyadic tolerance ledgers, regular readbacks, terminal seal rows, continuation replays, provenance rows, and local naming rows are componentwise* hsame*, then the local classifier exposes the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route for B and B'.*

Proof. Project both accepted packets. By Definition 1423.5, acceptance already identifies each packet with a displayed carrier through componentwise hsame rows for the analytic and structural coordinates.

Use the assumed componentwise hsame witnesses for M and M', S and S', D and D', and Q and Q'. The threshold schedule, finite window, dyadic ledger, and regular readback are therefore read in the same classifier order for both packets.

It remains to compare the terminal seal route. The witnesses for E and E' identify the exposed $\mathsf{Real}^{\uparrow}$ seal rows, and the witnesses for C, P, N identify the replay, provenance, and local naming rows that present that seal to the consumer.

The Real seal nonescape theorem Definition 1423.4 prevents either packet from exposing a detached seal. Hence the classifier exposes one terminal $\mathsf{Real}^{\uparrow}$ route, carried by the componentwise hsame data, for both accepted packets. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier seal determinacy`

Theorem 1396.15 (Real Cauchy modulus window scope)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Every finite precision request scopes only the displayed rows $M, S, D, Q, E, H, C, P, N$ over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *In particular, the request has no scope over quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient Real completeness, or an unbounded stream read.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier window scope`

Proof. Project the accepted packet through Definition 1423.1. The carrier lists exactly the modulus schedule, stream window, dyadic ledger, regular readback, terminal seal, transport, replay, provenance, and local naming rows.

Apply the route theorem Definition 1423.3 and the threshold-window totality theorem Definition 1423.9. They force a finite precision request to pass through M, then S, then D, then Q, and finally E, with H, C, P, N supplying the local transport and certificate data.

The exhaustion theorem Definition 1423.7 supplies the negative part of the scope claim. Since the non-displayed analytic principles are not rows of the packet and are not reached by the finite route, they cannot enter the finite-window scope. □

Theorem 1396.16 (Real Cauchy modulus tail-threshold determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For any finite unary precision request, strengthening the dyadic request may enlarge the scheduled finite* $\mathsf{StreamName}^{\uparrow}$ *window, but it cannot replace the threshold selected by M, detach the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, or introduce a selected limit. The tail of the accepted route remains determined by the same modulus-controlled passage through M, S, D, Q, E.*

Proof. Proceed by induction on the finite unary precision request. At the zero stage, Definition 1423.15 identifies the available rows, and Definition 1423.9 fixes the order from threshold to window, ledger, readback, and seal.

For the successor stage, assume the preceding request already follows the displayed route. A stronger dyadic tolerance is still classified by Definition 1423.5, so the request reads the next threshold through the same modulus row M and schedules the corresponding finite window in S.

The monotonicity theorem Definition 1423.12 allows this window to grow while preserving the earlier window, ledger entry, readback row, and terminal seal. The RegSeqRat seal admission theorem Definition 1423.10 then keeps Q attached to the route from D to E.

Finally, Definition 1423.7 rules out selected limits and other non-displayed analytic sources at every induction stage. Thus the stronger request changes only the finite amount of displayed tail data, not the threshold route or the seal handoff. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail threshold determinacy`

Theorem 1396.17 (Real Cauchy modulus RegSeqRat classifier determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ and $B' = (M', S', D', Q', E', H', C', P', N')$ be accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packets. If the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise* hsame*, then every terminal* $\mathsf{Real}^{\uparrow}$ *seal read factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier route before reaching E or E'.*

In particular, the terminal seal cannot be read directly from the dyadic ledger, from a detached stream representative, from host real equality, from countable choice, or from ambient Real completeness. The unique bridge from the dyadic tolerance ledger to the seal is the displayed $D \to Q \to E$ route transported by H, C, P, N.

Proof. Start with the finite-window route Definition 1423.3. It orders every accepted seal read through M, S, D, Q, E before the structural rows are consumed.

Use threshold-window totality from Definition 1423.9. The terminal route cannot skip from a precision request to D, Q, E; it first reads the scheduled threshold and finite StreamName window.

Now apply RegSeqRat seal admission, Definition 1423.10. It places the readback row Q as the only displayed bridge from the dyadic ledger D to the terminal seal E.

Finally use tail-threshold determinacy, Definition 1423.16. Strengthening the tail request may enlarge the finite window, but it preserves the same modulus-controlled route and keeps the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the seal.

The componentwise hsame assumptions identify the schedules, windows, ledgers, and readback rows of B and B'. Since ledger exhaustion excludes detached stream representatives, host equality, countable choice, and ambient completeness, both terminal seal reads factor through the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier route. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier regseqrat classifier determinacy`

Theorem 1396.18 (Real Cauchy modulus tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *completion consumer of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet must pass through the displayed rows*

$$M, S, D, Q, E, H, C, P, N.$$

Tail-threshold determinacy composes with window monotonicity and seal determinacy: a stronger tail request may enlarge the finite $\mathsf{StreamName}^{\uparrow}$ *window selected by M, but it preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows that transport, replay, source, and name the route. No completion consumer can replace this handoff by quotient stream equality, selected limits, host real equality, countable choice, or ambient Real completeness.*

Proof. Start with the finite route theorem Definition 1423.3. It fixes the consumer route through M, S, D, Q, E before the structural rows are used.

Use window monotonicity from Definition 1423.12. Strengthening the requested tail may enlarge the displayed finite window, but it does not delete the earlier window, ledger entry, readback row, or terminal seal.

Apply seal determinacy from Definition 1423.14. Componentwise hsame agreement on the displayed rows exposes the same terminal $\mathsf{Real}^{\uparrow}$ route to the consumer.

Finally use tail-threshold determinacy, Definition 1423.16. The stronger tail request remains on the same modulus-controlled path, so every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ completion consumer receives exactly the displayed handoff rows and no non-displayed completion principle. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail handoff`

Theorem 1396.19 (Real Cauchy modulus seal handoff)**.** *Every downstream Real-facing consumer of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the same displayed finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

The handoff is mediated by the rows H, C, P, N *and cannot be replaced by Real completeness, quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or an unbounded stream read.*

Proof. Begin with the finite-window scope theorem Definition 1423.15. It identifies the only rows available to a Real-facing consumer and fixes the analytic carriers through which the request may pass.

Use Definition 1423.16 for strengthened tail requests. The consumer may require a larger displayed window, but the threshold route, dyadic ledger, RegSeqRat readback, and terminal seal remain tied to the same modulus-controlled passage.

Now combine the original route theorem Definition 1423.3, threshold stability Definition 1423.6, ledger exhaustion Definition 1423.7, and seal determinacy Definition 1423.14. These results show that H transports the displayed rows, C replays the route, P records provenance, and N names the local certificate without adding another Real-facing access path.

The terminal handoff therefore reaches $\mathsf{Real}^{\uparrow}$ exactly at E. The negative alternatives are absent from the packet and blocked by the exhaustion and nonescape theorems, so every accepted downstream consumer receives the seal through the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier seal handoff`

Theorem 1396.20 (Real Cauchy modulus cofinal refinement handoff)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If* B *is read through a cofinal or monotone modulus schedule supplied by the existing* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *and* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *surfaces, then every downstream Real-facing read still factors through the displayed route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The cofinal or refined schedule may enlarge the finite window selected by M *and replay the same tail route, but it cannot replace the route by quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, Real completeness, or a detached Real seal.*

Proof. Begin with the finite route theorem Definition 1423.3. It fixes the order from the modulus row M to the StreamName window row S, the DyadicRatCore ledger D, the RegSeqRat readback Q, and the terminal Real seal E.

For a stronger precision request, use window monotonicity and tail-threshold determinacy, Definition 1423.12 and Definition 1423.16. These allow the displayed finite window to grow while preserving the threshold route, the dyadic ledger, the regular readback, and the terminal seal handoff.

Now read the sibling cofinal data only as route data. The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ cofinal-window factorization of Definition 885.15 supplies a refinement row for the finite-window schedule, and the $\mathsf{CofinalModulusSeal}^{\uparrow}$ cofinal route of Definition 1198.12 supplies the cofinal seal row. Neither sibling inserts a second Real seal into the $\mathsf{RealCauchyModulus}^{\uparrow}$ carrier.

Finally apply the seal handoff theorem Definition 1423.19. Since every Real-facing consumer of B reaches $\mathsf{Real}^{\uparrow}$ only through M, S, D, Q, E and the support rows H, C, P, N, the cofinal or monotone schedule preserves the same StreamName–DyadicRatCore–RegSeqRat–Real seal route and leaves the excluded completeness, choice, quotient, and host-equality alternatives outside the packet. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier cofinal refinement handoff`

Theorem 1396.21 (Real Cauchy modulus bounded tail-ledger factorization)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet, and let*

$$R_{\mathrm{bd}} = (S', M', B', Q', A', H', C', P', N')$$

be an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet consumed with componentwise matching modulus, window, dyadic ledger, RegSeqRat readback, and Real seal rows. Then every bounded tail-ledger read exported by the joint consumer factors first through the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

and then through the bounded packet route

$$S' \longmapsto M' \longmapsto B' \longmapsto Q' \longmapsto A'.$$

The handoff exports no Real completeness theorem, quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, or host real equality.

Proof. Begin with the cofinal refinement handoff Definition 1423.20. It shows that strengthened or cofinal tail requests still read the $\mathsf{RealCauchyModulus}^{\uparrow}$ packet through M, S, D, Q, E before any Real-facing seal is reached.

Next apply the seal handoff theorem Definition 1423.19. The terminal row E is therefore not a detachable Real object; it is the endpoint of the displayed modulus, window, dyadic-ledger, and readback route.

Now pass to the bounded consumer. The obligation surface of Definition 1697.2 identifies the bounded packet rows $S', M', B', Q', A', H', C', P', N'$, and the Real-seal boundary of Definition 1697.3 forces every bounded read through M', S', B', Q', A' before the terminal seal A' can be consumed.

The assumed componentwise matches identify the shared modulus, window, dyadic, readback, and seal rows at the interface. Thus a bounded tail-ledger read has only the composite finite passage displayed above, while the excluded completeness, quotient, choice, compactness, trichotomy, selected-limit, and host-equality coordinates remain absent from both carriers. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier bounded tail ledger factorization`

Theorem 1396.22 (Real Cauchy modulus bounded readback coherence)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet and let $R_{\mathrm{bd}} = (S', M', B', Q', A', H', C', P', N')$ be an accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet. If the shared modulus, finite window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows agree componentwise by* hsame*, then the* $\mathsf{RealCauchyModulus}^{\uparrow}$

readback and the BoundedRegularCauchy$^{\uparrow}$ *readback expose the same terminal finite Real-facing route. In particular, the bounded consumer cannot introduce a detached seal beyond the matched E/A' boundary.*

Proof. Use seal determinacy Definition 1423.14 to pin the terminal Real$^{\uparrow}$ row of the RealCauchyModulus$^{\uparrow}$ packet to the displayed M, S, D, Q route. Ledger exhaustion Definition 1423.7 then prevents a second dyadic or readback ledger from being inserted between Q and E.

On the bounded side, apply tail-window induction Definition 1697.4. Each requested finite threshold is read through the same bounded route to A', and the induction has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or Real completeness.

The componentwise hsame assumptions transport the modulus, window, dyadic ledger, readback, and seal rows across the interface. Hence the two readbacks terminate at the same finite Real-facing route, and any alleged detached seal would have to be a row not present in either displayed carrier. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier bounded readback coherence`

Theorem 1396.23 (Real Cauchy modulus uniform tail extraction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* RealCauchyModulus$^{\uparrow}$ *packet whose threshold row has passed through the cofinal refinement handoff of Definition 1423.20. Then there is a single displayed finite tail window, read from the same M, S, D, Q, E route, that every* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer named in the bounded readback coherence theorem Definition 1423.22 may use. Enlarging a requested tolerance only replaces the displayed window by a later cofinal window; it does not introduce quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, Real completeness, or a detached Real seal.*

Proof. Start with threshold stability, Definition 1423.6. It transports a stronger tolerance request along the same modulus row M, finite window row S, dyadic ledger D, RegSeqRat readback Q, and terminal Real seal E.

Next use the cofinal refinement handoff, Definition 1423.20. A later cofinal threshold may replace the initially requested tail window, but the replacement is still a displayed finite window in the same carrier route and is replayed by the same structural rows H, C, P, N.

Finally apply bounded readback coherence, Definition 1423.22. Its componentwise hsame matches force the RegSeqRat and Real-facing bounded consumers to read the same terminal finite route, so the cofinal window obtained above is uniform for all those consumers.

The excluded alternatives have no coordinate in either carrier and cannot be inserted between the matched readback and seal rows. Therefore the accepted packet exposes one reusable finite tail window for the bounded RegSeqRat and Real consumers. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier uniform tail extraction`

Theorem 1396.24 (Real Cauchy modulus finite-window tightening)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* RealCauchyModulus$^{\uparrow}$ *packet. Suppose a displayed precision request is tightened and the cofinal refinement row of Definition 1423.20 is available for the same packet. Then the resulting later finite* StreamName$^{\uparrow}$ *window still factors through the original route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The tightened request keeps the same CauchyModulus$^{\uparrow}$ *row M,* DyadicRatCore$^{\uparrow}$ *ledger D,* RegSeqRat$^{\uparrow}$ *readback Q, and* Real$^{\uparrow}$ *seal E. It introduces no detached seal, selected limit, quotient stream, host*

real equality, countable-choice schedule, open-cover compactness, trichotomy, or ambient completeness row.

Proof. Start with the cofinal refinement handoff Definition 1423.20. It permits the tightened precision request to move to a later finite window while preserving the route through M, S, D, Q, E.

Use uniform tail extraction, Definition 1423.23. The later window is not a new endpoint; it is the finite tail window that every named $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer may reuse through the same structural rows.

Apply bounded readback coherence from Definition 1423.22. It keeps the readback row Q and the terminal seal E componentwise matched with any bounded consumer that reads the tightened window.

Finally use ledger exhaustion, Definition 1423.7. Since the dyadic tolerance ledger is already exhausted by D before Q and E are read, there is no remaining coordinate for a detached seal, selected limit, quotient stream, host real equality, choice schedule, compactness proof, trichotomy principle, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier finite window tightening`

Theorem 1396.25 (Real Cauchy modulus obligation closure package)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. The carrier of Definition 1423.1, classifier of Definition 1423.5, NameCert obligations of Definition 1423.2, finite-window route of Definition 1423.3, threshold stability of Definition 1423.6, ledger exhaustion of Definition 1423.7, cofinal refinement handoff of Definition 1423.20, uniform tail extraction of Definition 1423.23, bounded readback coherence of Definition 1423.22, and finite-window tightening of Definition 1423.24 jointly supply the obligation surface for the nine-row packet B. The package introduces no Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, or open-cover compactness.*

Proof. Project the accepted packet through the carrier and classifier. The only visible rows are the modulus row M, finite $\mathsf{StreamName}^{\uparrow}$ window row S, dyadic tolerance ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, terminal $\mathsf{Real}^{\uparrow}$ seal E, componentwise transport H, displayed replay C, provenance P, and local naming row N.

Apply the NameCert obligations and finite-window route. These two sites bind the local naming surface to the displayed route from modulus requests through finite windows, dyadic tolerance data, readback, and the Real seal.

Use threshold stability, cofinal refinement handoff, uniform tail extraction, bounded readback coherence, and finite-window tightening. Together they show that stronger precision requests only move to later displayed windows on the same route and that every bounded consumer reuses the same readback and seal rows.

Finally apply ledger exhaustion. Since the dyadic ledger is already exhausted before the readback and terminal seal are consumed, there is no remaining carrier coordinate for Real completeness, quotient streams, host equality, countable choice, trichotomy, selected limits, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier obligation closure package`

Theorem 1396.26 (Real Cauchy modulus finite modulus-index induction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Every finite family of precision requests indexed by a*

unary finite window is handled by induction on the displayed modulus-index spine. At each successor step the same route

$$M \to S \to D \to Q \to E \to H, C, P, N$$

is preserved, and the induction exposes no unbounded stream read, choice schedule, quotient stream equality, selected limit, or Real completeness row.

Proof. Start with precision induction from Definition 1423.11. The base request reads the displayed modulus row M, the finite window row S, the dyadic ledger D, the regular readback Q, and the terminal seal E through the transported and replayed rows H, C, P, N.

For the successor step, apply window monotonicity from Definition 1423.12. Enlarging the unary finite window preserves the earlier request rows and moves only to a later displayed modulus index on the same spine.

Use uniform tail extraction from Definition 1423.23 to reuse the later finite tail for the whole indexed family. This is finite induction on the BEDC request spine, not a host choice over all precisions.

Finally apply ledger exhaustion from Definition 1423.7. Since every step is already routed through $M, S, D, Q, E, H, C, P, N$, there is no coordinate for an unbounded stream read, quotient stream equality, selected limit, countable-choice schedule, or ambient Real-completeness argument. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier finite modulus index induction`

Theorem 1396.27 (Real Cauchy modulus tail-coherence completeness handoff)**.** *Let $B = (M, S, D, Q, E, H, C, P,$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Any finite tail-coherence read that is passed from B to a completion witness must preserve the displayed route through* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The handoff may tighten a finite window or move to a later cofinal tail window, but it may not replace the route with quotient stream equality, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or ambient Real completeness.*

Proof. Begin with precision induction, Definition 1423.11. It makes each requested tail threshold a finite step on the displayed modulus spine rather than a global choice over all precisions.

Apply window monotonicity, Definition 1423.12. Enlarging the finite window keeps the same $M, S, D, Q, E, H, C, P, N$ route and only moves to a later displayed window.

Use uniform tail extraction, Definition 1423.23. The tail-coherence read exported to a completion witness is the same finite tail window already available to every named $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer.

Finally apply ledger exhaustion, Definition 1423.7. The dyadic tolerance data has no unused coordinate from which quotient stream equality, selected limits, countable choice, host equality, trichotomy, compactness, or ambient completeness could be inserted into the handoff. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail coherence completeness handoff`

Falsifiable prediction: An accepted $\mathsf{RealCauchyModulus}^{\uparrow}$ consumer that reads a terminal Real seal without the explicit modulus threshold, finite StreamName window, DyadicRatCore tolerance ledger, and RegSeqRat readback refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchymodulus: no listed sibling supplies the whole nine-row packet $B = (M, S, D, Q, E, H, C, P, N)$, because each sibling

contributes only a window, readback, tolerance, seal, or generic schedule surface while this chapter binds all five analytic faces with transport, replay, provenance, and local naming.

Theorem 1396.28 (Real Cauchy modulus uniform cofinal seal)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If the modulus row M is transported through the sibling lattice*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyModulus}^{\uparrow}$$

named by chapter 292, then every finite precision request yields a classifier-stable cofinal seal through the displayed route

$$M \to S \to D \to Q \to E \to H, C, P, N.$$

The exported row is uniform over finite precision requests and contains no selected real representative, host convergence principle, quotient stream equality, countable-choice schedule, or detached Real seal.

Proof. Start with finite modulus-index induction from Definition 1423.26. It handles a finite family of precision requests by induction on the displayed modulus-index spine while preserving the route through $M, S, D, Q, E, H, C, P, N$.

Read the sibling lattice as carrier data. The StreamName row supplies the finite windows, the RegSeqRat row supplies the regular readback, the DyadicRatCore row supplies the tolerance ledger, the Real row supplies the terminal seal, and the CauchyModulus chapter cited above supplies the schedule discipline for cofinal requests.

Now apply cofinal refinement handoff and uniform tail extraction. A later cofinal threshold may enlarge the finite window, but the transported request still reaches the same terminal route and is replayed by the same classifier rows H, C, P, N.

Finally use ledger exhaustion. Since every precision request has already consumed the displayed modulus, window, tolerance, readback, and seal rows, the cofinal seal cannot import a selected representative, host convergence theorem, quotient stream equality, countable-choice schedule, or detached Real seal. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier uniform cofinal seal certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyModulus}^{\uparrow}$ packet is generated from an explicit $\mathsf{CauchyModulus}^{\uparrow}$ precision schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1423.1, Definition 1423.5, Definition 1423.2, Definition 1423.3, Definition 1423.4, Definition 1423.6, Definition 1423.7, Definition 1423.8, Definition 1423.9, Definition 1423.10, Definition 1423.11, Definition 1423.12, Definition 1423.14, Definition 1423.15, Definition 1423.16, Definition 1423.17, Definition 1423.18, Definition 1423.19, Definition 1423.20, Definition 1423.21, Definition 1423.22, Definition 1423.23, Definition 1423.24, Definition 1423.27, and Definition 1423.28 seed the explicit modulus-controlled Real Cauchy read over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedRegularCauchy}^{\uparrow}$, and the uniform cofinal seal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, or a Lean theorem.

Upgrade path: Mature closure requires a checked standard bridge for the nine-row carrier and finite-window Real seal route.

Chapter 1397

A Concrete Naming Certificate for $\mathsf{DedekindCauchyComparison}^{\uparrow}$

Following Bishop and Bridges and Troelstra–van Dalen, the $\mathsf{DedekindCauchyComparison}^{\uparrow}$ packet records the finite comparison boundary between rounded located cut observations and regular Cauchy name observations. It reads the cut surface of chapter 436, the Cauchy equivalence surface of chapter 1412, the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a representation comparison certificate, not a Dedekind construction of $\mathsf{Real}^{\uparrow}$ and not a completeness theorem.

Definition 1397.1 (DedekindCauchyComparison carrier)**.** *A* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, R, W, T, E, H, C, P, N)$$

with the following displayed coordinates:

1. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *row, carrying lower and upper rational-observation data together with roundedness, locatedness, and disjointness evidence;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, carrying the regular rational sequence surface that is allowed to answer finite tolerance requests;*
3. *W is the family of finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which R is read by downstream consumers;*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, comparison budgets, and finite refinement rows used to compare D with R;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row reached from the displayed cut and regular-sequence observations, with no independent completed-real coordinate;*
6. *H stores componentwise* hsame *transports for D, R, W, T, E;*
7. *C stores the displayed* Cont *replay routes for rereading the same finite cut observations, stream windows, tolerance rows, and seal boundary rows;*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a quotient cut, a host real equality proof, a selected limit, countable choice, a construction of $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DedekindCauchyComparisonUp.DedekindCauchyComparisonTasteGate single carrier alignment`

Theorem 1397.2 (DedekindCauchyComparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}}$ *surface for the carrier of Definition 1424.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the packet* $K = (D, R, W, T, E, H, C, P, N)$ *with its displayed cut row, regular sequence row, stream windows, dyadic tolerance ledger, seal boundary, transports, continuations, package provenance, and local naming evidence;*
2. *classifier: comparison reads only rational cut observations in* D*, finite regular-sequence observations in* R*, the windows* W*, and the dyadic budgets* T*;*
3. *transport stability:* H *and* C *preserve the same cut observations, sequence windows, tolerance rows, and seal boundary under accepted consumer motion;*
4. *exactness: every comparison-facing read is supplied by the displayed rows* D, R, W, T, E *after the componentwise transport and replay routes are fixed;*
5. *ledger: every downstream* $\mathsf{Real}^{\uparrow}$ *consumer first reads the comparison packet through* $D, R, W, T, E, H, C, P, N$ *and receives no hidden completion coordinate.*

Proof. Carrier admission is projection from the packet $K = (D, R, W, T, E, H, C, P, N)$. The displayed cut row, regular sequence row, stream-window family, dyadic tolerance ledger, seal boundary, transports, continuations, provenance, and local naming evidence are exactly the coordinates named by the carrier.

The classifier is finite because every comparison request is routed through the stored rational observations in D, the finite windows W of the regular sequence row R, and the dyadic tolerance ledger T. No classifier step can inspect a quotient cut or an unbounded real object.

Transport stability uses H componentwise and then replays the same consumer path through C. The replay returns to the stored rows D, R, W, T, E and therefore preserves the same finite observation surface.

Exactness and the ledger clause follow from the same inventory. Since the carrier contains no selected limit, host real equality proof, countable-choice schedule, Dedekind construction row for $\mathsf{Real}^{\uparrow}$, or completeness principle, every downstream comparison is exhausted by the displayed packet rows. □

Theorem 1397.3 (DedekindCauchyComparison Real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *carrier is only a representation boundary between the displayed* $\mathsf{DedekindCut}^{\uparrow}$ *row and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row. The boundary supplies no quotient cut, no host real equality, no selected limit, no countable choice, no Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, and no completeness theorem.*

Proof. Project the accepted carrier. Its real-facing coordinate is the seal boundary row E, and E is stored together with the cut row D, regular rational row R, finite stream-window family W, and dyadic tolerance ledger T.

Read a comparison request in finite order. The consumer first exposes rational cut observations from D, then reads the corresponding finite windows W of R, and then checks the dyadic budgets recorded in T.

Only after those rows have been fixed can the consumer read the seal boundary E. The transport and replay rows H and C preserve that same route, while P and N record provenance and local naming.

Thus the packet exports a comparison boundary only. A quotient cut, host real equality proof, selected limit, countable-choice schedule, Dedekind construction of $\mathsf{Real}^{\uparrow}$, and completeness theorem have no carrier coordinate and cannot be read from the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCauchyComparison}^{\uparrow}$ packet is generated from a $\mathsf{DedekindCut}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1424.1, Definition 1424.2, and Definition 1424.3 seed the finite representation-comparison boundary between Dedekind-cut observations and regular Cauchy name observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DedekindCauchyComparisonUp.DedekindCauchyComparisonTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient cuts, host real equality, selected limits, countable choice, Dedekind construction of $\mathsf{Real}^{\uparrow}$, completeness, or bridge checking.

Upgrade path: Scoped closure requires a checked finite carrier theorem binding the displayed cut row, regular sequence row, stream windows, dyadic tolerance ledger, and Real-seal boundary.

Chapter 1398

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ packet isolates the regular-Cauchy universal-property seed for the L10 Real and Completion route. It refines the general $\mathsf{CompletionUniversalProperty}^{\uparrow}$ surface of chapter 397 by forcing every extension and uniqueness read through the finite regular-name chain already exposed by chapter 652, chapter 1404, chapter 126, chapter 301, chapter 291, and chapter 38.

The purpose of the packet is local. It names the rows needed for extending a regular-Cauchy source route into the terminal $\mathsf{Real}^{\uparrow}$ seal and for comparing any two such extensions by the displayed hsame ledger. It does not claim the full ambient completeness of $\mathsf{Real}^{\uparrow}$, nor does it replace the mature general completion chapter.

Definition 1398.1 (RegularCauchyCompletionUniversal carrier)**.** *A* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, D, R, A, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular source row whose admissible names are read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by the same regular source;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows;*
4. *R is the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *comparison row for replacing regular names without leaving the finite route;*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached through the displayed regular source, window, tolerance, and equivalence rows;*
6. *E is the extension route row, read as the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension face restricted to S, W, D, R, A;*
7. *L is the uniqueness* hsame *ledger comparing two extension routes over the same regular source, finite windows, dyadic ledger, equivalence row, and terminal seal;*
8. *H records the componentwise* hsame *transports for S, W, D, R, A, E, L;*
9. *C records the displayed* Cont *routes that replay the extension and uniqueness reads;*

10. *P is the* Pkg *provenance over* CompletionUniversalProperty$^{\uparrow}$, RealCompletionWitnessExtractor$^{\uparrow}$, RegularCauchyEquivalence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{RegularCauchyCompletionUniversal}^{\uparrow}}$ *row.*

The accepted carrier stores no quotient stream, no selected host limit, no host real equality row, no countable-choice selector, and no ambient Real$^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversalTasteGate single carrier alignment`

Theorem 1398.2 (Regular extension route)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet. Every extension consumer of E first reads the regular source S, then the finite* StreamName$^{\uparrow}$ *windows W, then the* DyadicRatCore$^{\uparrow}$ *ledger D, then the* RegularCauchyEquivalence$^{\uparrow}$ *comparison row R, and only then reaches the terminal* Real$^{\uparrow}$ *seal A and the restricted* CompletionUniversalProperty$^{\uparrow}$ *extension face.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversal extension route`

Proof. Project the carrier of Definition 1425.1. The extension row E is not stored as a detached completion theorem; it appears after the regular source, the scheduled finite windows, the dyadic ledger, the equivalence row, and the terminal Real seal.

Read the sibling surfaces in the displayed order. The StreamName, RegSeqRat, DyadicRatCore, RegularCauchyEquivalence, and Real chapters provide exactly the finite route named by S, W, D, R, A, while the general completion chapter supplies the extension face only after that route has been exposed.

The continuation row C can replay only those displayed reads, and the provenance row P records no selected limit or host completion object. Therefore every extension consumer factors through the regular-Cauchy route before it reaches the universal-property face. □

Theorem 1398.3 (Regular uniqueness ledger)**.** *For accepted packets $U = (S, W, D, R, A, E, L, H, C, P, N)$ and $U' = (S, W, D, R, A, E', L', H', C', P, N')$ over the same regular source, finite windows, dyadic ledger, equivalence row, terminal seal, and provenance, any comparison between the exported extension routes is exhausted by the displayed uniqueness ledgers L and L' together with their componentwise* hsame *transports. No quotient-stream equality, selected host limit, host real equality, countable choice, or ambient completeness row can contribute to the comparison.*

Proof. Start with the shared coordinates S, W, D, R, A, P. They fix the regular source, scheduled windows, tolerance ledger, regular-Cauchy comparison row, and terminal Real seal for both packets.

The only coordinates that can distinguish the two extension exports are E, L, H, C, N and E', L', H', C', N'. The ledgers L and L' compare the extension rows over the same shared route, while H, H' and C, C' display the permitted transports and replays.

Since neither carrier contains a quotient stream, selected host limit, host real equality row, countable-choice selector, or ambient completeness coordinate, no external equality principle can enter the comparison. The uniqueness judgement is exactly the local hsame ledger over the shared regular-Cauchy route. □

Theorem 1398.4 (Regular section law)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet. If a consumer first includes the displayed regular source*

through S, W, D, R into the terminal $\mathsf{Real}^{\uparrow}$ *seal A and then reads the restricted extension face E, the exported route is compared by the local uniqueness ledger L to the original A-sealed route. No quotient stream, selected limit, host real equality, countable choice, or ambient completeness row mediates this section comparison.*

Proof. Project the carrier of Definition 1425.1. The displayed source inclusion has only the coordinates S, W, D, R, A available before the extension face is read.

Apply Definition 1425.2. It forces every consumer of E to pass through the source row, finite windows, dyadic ledger, regular-Cauchy comparison row, and terminal Real seal in that order, so the restricted extension face is attached to the same finite route as the original A-sealed read.

Apply Definition 1425.3 with the shared coordinates fixed by this route. The ledger L, together with the displayed transports H and replays C, is the only comparison surface for the extension export over S, W, D, R, A.

The remaining rows P and N record provenance and the local NameCert seal, and no carrier coordinate stores a quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness witness. Thus L, H, C, P, N exhaust the section comparison inside the packet. □

Theorem 1398.5 (Regular Real-seal factorization)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of U must read the finite route*

$$S \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow A$$

before it may inspect the extension face E or the uniqueness ledger L. No quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate can factor the same accepted read.*

Proof. Project the carrier of Definition 1425.1. The Real-facing rows appear as the regular source S, finite windows W, dyadic ledger D, regular-Cauchy comparison row R, and terminal Real seal A.

Apply Definition 1425.2 to the extension face E. It forces every extension read to pass through S, W, D, R, A before the restricted completion face is available.

Apply Definition 1425.3 to comparisons that use L. The ledger compares extension routes only over the same displayed source, window, dyadic, comparison, and Real-seal rows.

The remaining coordinates H, C, P, N transport, replay, source, and name this route. Since the carrier stores no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness row, coordinate exhaustion gives the stated factorization boundary. □

Theorem 1398.6 (RegularCauchyCompletionUniversal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCor}}$ *surface consists exactly of carrier admission, the regular extension route, the uniqueness ledger, the section law, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming over* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$*,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Carrier admission is the packet shape of Definition 1425.1. Its rows name the regular source, finite windows, dyadic ledger, comparison row, terminal Real seal, extension route, uniqueness ledger, componentwise transports, continuation replays, provenance, and local certificate.

The regular extension route is bound by Definition 1425.2. The uniqueness ledger is bound by Definition 1425.3, and the section law is bound by Definition 1425.4.

No remaining coordinate is available in the carrier. Therefore the displayed rows exhaust the local NameCert obligation surface over the listed kernel and sibling objects. □

Theorem 1398.7 (RegularCauchyCompletionUniversal kernel scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. The extension, uniqueness, and section obligations of Definition 1425.6 are scoped through the displayed kernel rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*: the only admissible reads are carrier projection from U, componentwise comparison through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $U = (S, W, D, R, A, E, L, H, C, P, N)$. The rows H, C, P, N are the only structural coordinates after the source, window, ledger, equivalence, seal, extension, and uniqueness entries have been displayed.

The extension route can therefore be replayed only through the Cont rows stored in C. The uniqueness and section comparisons can be transported only through the componentwise hsame rows stored in H.

The remaining structural access is provenance and naming. The rows P and N bind the route to Pkg and NameCert, so the kernel scope is exhausted by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 1398.8 (RegularCauchyCompletionUniversal real seal scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. Every real-facing obligation in the extension, uniqueness, and section surface is scoped through the sibling rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*: W supplies the finite windows, S supplies the regular source, D supplies the dyadic ledger, R supplies the equivalence boundary, and A supplies the terminal Real seal.*

Proof. Project U and read the real-facing coordinates in order. The source row S is usable only with the finite windows W, and the tolerance row D is the only dyadic ledger available before equality-facing comparison.

The comparison row R is the displayed $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ boundary. It is reached after S, W, D and before the terminal seal, so no quotient equality or host real equality can intervene.

The row A is the only $\mathsf{Real}^{\uparrow}$ seal in the carrier. Hence every real-facing extension, uniqueness, and section read is sealed by A after the StreamName, RegSeqRat, DyadicRatCore, and RegularCauchyEquivalence rows have been displayed. □

Theorem 1398.9 (RegularCauchyCompletionUniversal extractor scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. The completion-facing obligations are scoped through the displayed* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *and restricted completion row: the extractor is read only after S, W, D, R, A, the extension face is E, and the section comparison is the local ledger L transported through H and replayed through C.*

Proof. Project the carrier and isolate the completion-facing rows. The extractor-facing source data are exactly S, W, D, R, A, already sealed as the regular-Cauchy route of Definition 1425.8.

The extension face E is downstream of those rows. It cannot read a quotient completion, a selected host limit, or an ambient completeness theorem because none of those objects has a coordinate in U.

The section comparison is then local. The ledger L compares the extension route with the original Real-sealed route, while H and C provide the only permitted transport and replay rows for that comparison. □

Theorem 1398.10 (RegularCauchyCompletionUniversal public export)**.** *Every public consumer of an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet receives exactly the extension route, uniqueness ledger, section law, componentwise transport, continuation replay, provenance, and local* NameCert *rows scoped by the displayed kernel rows of Definition 1425.7, the Real-seal rows of Definition 1425.8, and the completion-extractor rows of Definition 1425.9. The exported surface contains no quotient stream, selected host limit, host real equality, countable choice, ambient RealUp completeness, or standard bridge row.*

Proof. Start with the accepted carrier $U = (S, W, D, R, A, E, L, H, C, P, N)$. The kernel scope theorem fixes the only structural rows visible to a consumer: carrier projection through U, transport through H, replay through C, provenance through P, and local naming through N.

Next read the Real-facing route. The Real-seal scope theorem requires every extension, uniqueness, or section read to pass through S, W, D, R, A before any completion-facing row is visible, so a public consumer cannot substitute quotient stream equality or host real equality for the displayed seal.

Finally apply the completion-extractor scope theorem. It identifies the completion-facing export as the restricted extension row E together with the local uniqueness and section ledger L transported by H and replayed by C. Thus the public export is exactly the listed route, ledger, law, transport, replay, provenance, and local certificate rows. □

Closure status for NameCert$_{\text{RegularCauchyCompletionUniversal}\uparrow}$.

Constructive story: A RegularCauchyCompletionUniversal$^{\uparrow}$ packet is generated from a regular source row, StreamName windows, a dyadic tolerance ledger, a RegularCauchyEquivalence row, a RealUp seal, an extension route, a uniqueness hsame ledger, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionUniversal}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1425.7, Definition 1425.8, Definition 1425.9, and Definition 1425.10 bind carrier admission, extension, uniqueness, section, transport, replay, provenance, naming, public consumer export, and the restricted completion-interface bridge to the displayed kernel, Real-route, and completion-extractor rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionUniversal}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversalTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, countable choice, ambient RealUp completeness, or a standard bridge.

Upgrade path: No further paper-side theory-axis upgrade remains; formal bridge checking remains on the independent verification axis.

Chapter 1399

A Concrete Naming Certificate for $\mathsf{SternBrocotApproximation}^{\uparrow}$

Following Stern–Brocot and Farey rational approximation, the $\mathsf{SternBrocotApproximation}^{\uparrow}$ packet records finite rational bracketing as displayed BEDC data. It reads a pair of rational interval endpoints through chapter 354, a finite mediant path through dyadic tolerance bookkeeping from chapter 291, a $\mathsf{StreamName}^{\uparrow}$ observation window from chapter 126, a $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, and a terminal $\mathsf{Real}^{\uparrow}$ seal from chapter 38. The packet is a finite bracketing neighbor of the decimal and continued-fraction readback surfaces in chapter 1417 and chapter 342; it does not store an ambient real order or a quotient-real representative.

Definition 1399.1 (Stern–Brocot approximation carrier)**.** *A* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational endpoint row, carrying the left side of the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *bracket.*
2. *U is the upper rational endpoint row, carrying the right side of the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *bracket.*
3. *M is the finite mediant-path ledger, whose rows record the left/right Stern–Brocot choices and the rational mediants actually inspected.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the dyadic precision requested by the consumer and the displayed bound showing that the bracket width is below that tolerance.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window used to request and replay the inspected rational rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same endpoint and mediant data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the rational interval, tolerance, stream-window, and regular-rational readback rows have been displayed.*
8. *H stores componentwise* hsame *transports for endpoint, mediant-path, tolerance, stream-window, readback, and seal rows.*
9. *C stores the* Cont *replay from stream windows to mediant inspection, from mediant inspection to rational bracketing, from bracketing to regular readback, and from readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$ *row.*

The carrier stores only a finite rational bracketing certificate. It has no coordinate for Dedekind cuts, trichotomy, quotient reals, selected representatives, or RealUp completeness.

Lean4 ✓ `BEDC.Derived.SternBrocotApproximationUp.SternBrocotApproximationTasteGate single carrier alignment`

Theorem 1399.2 (Stern–Brocot NameCert obligations)**.** *For an accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *carrier* $K = (L, U, M, T, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower endpoint, upper endpoint, mediant-path, dyadic tolerance, stream-window, regular-rational readback, real-seal, transport, continuation, provenance, and naming rows;*
2. *classifier transport along the stored* hsame *rows and displayed* Cont *routes;*
3. *bracketing exactness, stating that every exported rational bracket is one of the endpoint or mediant rows displayed by L, U, M and checked against T;*
4. *readback discipline, stating that every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads through the finite window W and the regular readback row R before receiving the seal row E;*
5. *ledger exclusion, stating that Dedekind cuts, trichotomy, quotient reals, selected representatives, and RealUp completeness are not exported by the packet.*

Proof. Carrier admission is projection from the tuple in Definition 1431.1. The rational endpoints, mediant path, dyadic tolerance, stream window, regular readback, real seal, transports, continuations, package row, and local naming row are precisely the displayed coordinates.

Classifier transport is componentwise. The endpoint and mediant rows move through their stored hsame witnesses, while C replays the same window-to-mediant and bracket-to-readback route after transport.

Bracketing exactness uses the sibling interval and dyadic arithmetic surfaces. The endpoint rows are the displayed $\mathsf{RationalInterval}^{\uparrow}$ bounds from chapter 354, and the tolerance row is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ comparison row from chapter 291.

The readback discipline then follows the stored route through chapter 126, chapter 301, and chapter 38. The excluded principles are absent from the carrier inventory, so the local certificate cannot export them. □

Theorem 1399.3 (Finite rational bracketing)**.** *Every accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *packet exposes a finite rational bracket*

$$L \leq M_i \leq U$$

at each inspected mediant row M_i of the finite ledger M, and the final displayed bracket has width bounded by the dyadic tolerance row T. Every route to the bracket factors through

$$\mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow}.$$

The packet supplies no host trichotomy, no quotient-real equality test, and no choice of a distinguished representative for the sealed real.

Proof. Read the mediant ledger M in order. Each row records a finite left/right choice and the rational mediant computed from the currently displayed endpoints, so the next bracket is formed by replacing exactly one endpoint with that mediant.

The interval side is supplied by chapter 354. It records the endpoint rows and the local comparison data needed to keep the displayed lower endpoint below the displayed upper endpoint at the inspected scale.

The tolerance side is supplied by chapter 291. The row T records the dyadic precision and the displayed comparison showing that the final interval width is below that precision.

The window side is supplied by chapter 126. Since the carrier has no other observation coordinate, every bracket read factors through the displayed finite window and the rational rows it exposes. □

Theorem 1399.4 (Stern–Brocot Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *packet factors through the finite handoff*

$$\mathsf{SternBrocotApproximation}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The Stern–Brocot carrier provides the finite rational bracket; the decimal and continued-fraction siblings of chapter 1417 and chapter 342 provide comparison readback neighbors; and the real-facing consumer receives only the displayed regular rational row R and seal row E.

Proof. Start with the accepted packet $K = (L, U, M, T, W, R, E, H, C, P, N)$. The carrier exposes the bracket through L, U, M, T and exposes the observation request through W.

The regular-rational handoff is the row R. By the readback discipline of chapter 301, a real-facing consumer reads the finite rational data through the regular sequence surface before reading a real seal.

The seal row E is therefore downstream of R and the displayed window route. The transports H, continuations C, and provenance row P preserve the same route for classified packets and for repeated finite-window reads.

The comparison with decimal and continued-fraction siblings is only a readback comparison: all three chapters present finite rational observation surfaces before the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. None supplies Dedekind cuts, trichotomy, quotient reals, selected representatives, or RealUp completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SternBrocotApproximation}^{\uparrow}$ packet is generated from rational interval endpoints, a finite mediant path, a dyadic tolerance row, a finite StreamName window, a RegSeqRat readback, a RealUp seal, componentwise transports, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1431.1, Definition 1431.2, Definition 1431.3, and Definition 1431.4 seed the finite Stern–Brocot rational-bracketing surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, trichotomy, quotient reals, selected representatives, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, bracketing exactness, readback discipline, ledger exclusion, and RealUp seal handoff for the same finite packet.

Chapter 1400

A Concrete Naming Certificate for $\mathsf{CauchyCompletionStability}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletionStability}^{\uparrow}$ packet isolates the finite stability route by which a modulus-indexed regular rational Cauchy read remains attached to the same terminal Real seal under accepted transport. It depends on the finite-window naming surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal seal of chapter 38, the explicit modulus packet of chapter 1423, and the located comparison boundary of **??**. The chapter records only the constructive L10 completion-stability route over those siblings.

Definition 1400.1 (CauchyCompletionStability carrier)**.** *A* $\mathsf{CauchyCompletionStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule that assigns each requested completion tolerance a finite threshold and radius budget;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family selected by the thresholds in M;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the finite dyadic comparisons read on those windows;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose regularity is witnessed only through M, S, D;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the modulus, window, tolerance, and regular-readback rows have been consumed;*
6. *H records componentwise* hsame *transport for M, S, D, Q, E and for the structural rows of the packet;*
7. *C records the displayed* Cont *replay that rereads the same modulus schedule, StreamName windows, dyadic tolerance rows, RegSeqRat readback, Real seal, and transport ledger;*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionStability}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStability}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, countable choice, trichotomy, a selected limit, open-cover compactness, or a general $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 1400.2 (CauchyCompletionStability NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStability}^{\uparrow}}$ *surface for Definition 1400.1 consists exactly of the following finite obligation rows:*

1. *carrier exposure: the accepted object is the packet* $T = (M, S, D, Q, E, H, C, P, N)$ *with no hidden completion coordinate;*
2. *classifier: each stability request is classified only by the displayed modulus schedule* M*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, dyadic tolerance ledger* D*, and regular readback row* Q*;*
3. *stability: the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *remains attached to the same finite route whenever* H *transports the displayed rows and* C *replays the corresponding consumer path;*
4. *provenance:* P *records exactly the sibling route through* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{LocatedReal}^{\uparrow}$*, while* N *names the local packet.*

Proof. Carrier exposure is projection from the tuple $T = (M, S, D, Q, E, H, C, P, N)$. Every accepted coordinate is displayed, and the final coordinate is the local naming row.

The classifier is finite because a stability request first reads the threshold data in M, then the finite windows in S, then the dyadic tolerance ledger in D, and then the regular rational readback in Q. No classifier row can inspect a quotient stream or a host real equality proof.

Transport stability uses H componentwise and then replays the same path through C. The replay returns to the rows M, S, D, Q, E, so the terminal Real seal is stable only along the displayed route.

The provenance and naming clauses follow from P and N. Since the carrier contains no coordinate for countable choice, trichotomy, selected limits, open-cover compactness, or general Real completeness, the four obligation rows exhaust the local certificate surface. □

Theorem 1400.3 (CauchyCompletionStability cross-chapter scope)**.** *Every export of an accepted* $\mathsf{CauchyCompletionStability}^{\uparrow}$ *packet is confined to the finite L10 completion-stability route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

It exports no quotient stream equality, no host real equality, no countable-choice schedule, no trichotomy principle, no selected limit, no open-cover compactness principle, and no general $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the packet through the obligations of Definition 1400.2. The only analytic route is the displayed sequence from the modulus schedule to the terminal Real seal.

The structural coordinates do not widen that route. The transport row H and replay row C preserve the same rows, while P and N only record provenance and local naming.

Inspecting the carrier of Definition 1400.1 shows that the excluded principles have no row from which they could be read. Therefore the packet exports exactly the finite completion-stability route and none of the listed ambient or classical principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionStability}^{\uparrow}$ packet is generated from a modulus schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1400.1, Definition 1400.2, and Definition 1400.3 seed the finite completion-stability route across StreamName, DyadicRatCore, RegSeqRat, Real, RealCauchyModulus, and LocatedReal siblings.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionStability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, general Real$^{\uparrow}$ completeness, or a Lean theorem.

Upgrade path: A formal route can encode the nine-row carrier and prove the NameCert obligation and cross-chapter scope theorems under BEDC.Derived.CauchyCompletionStabilityUp.

Theorem 1400.4 (CauchyCompletionStability L10 sibling route)**.** *Every accepted* CauchyCompletionStability$^{\uparrow}$ *packet reads its L10 dependency lattice through the finite-window* StreamName$^{\uparrow}$ *rows of chapter 126, the regular-rational* RegSeqRat$^{\uparrow}$ *rows of chapter 301, the* DyadicRatCore$^{\uparrow}$ *tolerance rows of chapter 291, the terminal* Real$^{\uparrow}$ *seal of chapter 38, and the* Completion$^{\uparrow}$ *and* SeparatedMetric$^{\uparrow}$ *boundary of chapter 95 and chapter 1411 before it exports completion stability. The route is visible as a finite* BHist *spine whose* Cont *replay and* hsame *transports reread exactly those sibling rows; it does not identify quotient representatives, select limits, invoke countable choice, assert host equality, or import ambient* Real$^{\uparrow}$ *completeness.*

Proof. The carrier of Definition 1400.1 exposes the rows $M, S, D, Q, E, H, C, P, N$. The finite windows are read from S, the dyadic tolerances from D, the regular-rational readback from Q, and the terminal Real seal from E. The completion boundary is the displayed passage from the regular readback to the terminal seal, and the separated-metric boundary is the uniqueness discipline enforced only by the visible transport rows.

The route is therefore a BHist and Cont route, not a quotient construction. The replay coordinate C rereads the same windows, tolerances, readback, seal, and boundary rows, while H gives only componentwise hsame transport. No coordinate of the packet stores quotient representatives, a selected limit, a choice schedule, host equality, or an ambient completeness theorem, so none of those principles can be exported by the local NameCert. □

Chapter 1401

CauchyContinuousMapUp NameCert Construction

Following Bishop and Bridges, the $\mathsf{CauchyContinuousMap}^{\uparrow}$ packet records the finite interface by which a map on regular Cauchy names preserves regularity and hands its output to a $\mathsf{Real}^{\uparrow}$ seal. The construction is a NameCert surface over the dyadic rational core of chapter 291, the stream-name schedule of chapter 126, the regular rational-name carrier of chapter 301, and the real-seal boundary of chapter 38.

The packet is deliberately weaker than the general continuous-map surface of chapter 130. It records a finite Cauchy-name transformer with displayed readback and seal handoff, not a host function on reals, a metric-continuity proof over all points, a uniform-continuity modulus, a quotient-stream map, or a completeness theorem.

Definition 1401.1 (CauchyContinuousMap carrier)**.** *A* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface read by the map, including the source windows and the output windows that the packet promises to fill;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *image-readback row, carrying the source regular rational name, the displayed output rational name, and the finite evidence that output windows remain regular;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare every selected source window with its displayed output tolerance row;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row reached from the output regular name after the dyadic tolerance checks have been replayed;*
5. *H records componentwise* hsame *transports for W, R, D, E and for the local classifier row;*
6. *C records the displayed* Cont *replay routes by which consumers inspect the same windows, tolerance rows, image readback, and seal handoff;*
7. *P is the* Pkg *provenance over the stream, regular-name, dyadic, real-seal, transport, continuation, and local naming rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuousMap}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by replaying these finite rows. It has no coordinate for a full $\mathsf{ContinuousMap}^{\uparrow}$ *object, host real function, metric continuity theorem, uniform-continuity modulus, quotient of streams, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMapTasteGate single carrier alignment`

Theorem 1401.2 (RegSeqRat image preservation)**.** *For every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$*, every image read exposed by the packet factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*. The readback therefore preserves the regular-Cauchy-name shape visible to chapter 301, without exporting a quotient stream or a host-level function on reals.*

Proof. Project the carrier of Definition 1401.1. The only image-readback coordinate is R, and it is stored beside the finite window surface W and dyadic tolerance ledger D.

Replay the consumer route through C. The route can inspect the selected source windows, the output windows, the dyadic tolerances, and the regular rational output row, but it cannot read an additional stream quotient or host real function because no such row is present.

Transport along H preserves the same displayed rows componentwise. Thus every accepted image read remains a $\mathsf{RegSeqRat}^{\uparrow}$-shaped finite readback assembled from W, D, and R. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap regseqrat image`

Theorem 1401.3 (Real seal handoff)**.** *For every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$ *seal consumer must first read the finite stream windows* W*, the image regular-name row* R*, and the dyadic tolerance ledger* D*. Only after those rows are replayed can the consumer reach the seal handoff* E*.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap real seal handoff`

Proof. Read the carrier order in Definition 1401.1. The seal handoff row E is not independent; it is listed as the handoff reached from the output regular name after the dyadic tolerance checks.

Use the displayed Cont routes in C to replay the handoff. These routes expose the window schedule, the image readback, and the tolerance ledger before the real-seal row is available.

The componentwise hsame transports in H keep the same dependency order after transport. Hence a seal consumer receives a finite RegSeqRat-to-Real route and no detached real function, quotient stream, or completeness datum. □

Theorem 1401.4 (ContinuousMap boundary)**.** *Every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packet supplies only a finite Cauchy-name transformer boundary for the* $\mathsf{ContinuousMap}^{\uparrow}$ *interface. A* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer may read the packet only after the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed through* W, R, D, E *and transported through* H, C, P, N*. The packet does not provide a full host function, metric continuity theorem, uniform-continuity modulus, quotient stream map, or completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMapPacket continuousmap boundary`
↪ variant `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap continuousmap boundary`

Proof. Read the carrier rows from Definition 1401.1. The rows W, R, D, E display the stream windows, regular rational image readback, dyadic tolerance checks, and real-seal handoff before any continuous-map consumer can inspect the packet.

The sibling boundary is chapter 130. This chapter exposes only the Cauchy-name side of that interface: H transports the displayed rows, C replays them for consumers, P records provenance, and N names the local certificate.

Since the carrier inventory has no coordinate for a host function, metric continuity theorem, uniform-continuity modulus, quotient stream map, or completeness witness, a $\mathsf{ContinuousMap}^{\uparrow}$ read must treat the packet as a finite boundary rather than as the full continuous-map object. □

Theorem 1401.5 (CauchyContinuousMap NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CauchyContinuousMap}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation by a* $\mathsf{StreamName}^{\uparrow}$ *window surface,* $\mathsf{RegSeqRat}^{\uparrow}$ *image-readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal handoff, displayed transports, continuations, provenance, and local naming;*
2. *classifier replay by the finite windows and dyadic tolerance checks that witness regular-name preservation;*
3. *transport stability under the stored* hsame *rows and displayed* Cont *routes;*
4. *real-seal exactness, stating that every* $\mathsf{Real}^{\uparrow}$ *read factors through the image regular-name and dyadic tolerance rows first;*
5. *ledger refusal, stating that full* $\mathsf{ContinuousMap}^{\uparrow}$ *structure, metric continuity, uniform continuity, quotient streams, host real functions, and ambient completeness are not packet coordinates.*

Proof. Carrier habitation is exactly the tuple displayed in Definition 1401.1. It supplies the window, image-readback, tolerance, seal, transport, continuation, provenance, and local naming rows.

The classifier row is finite because it replays only the windows in W and the tolerance checks in D. Definition 1401.2 gives the corresponding regular-name preservation surface.

Transport stability is obtained by projecting the componentwise hsame rows in H and replaying the same reads through C. Definition 1401.3 gives the seal-exactness row.

The refusal row follows from the carrier inventory. None of the displayed coordinates stores a full continuous-map object, metric-continuity proof, uniform-continuity modulus, quotient stream, host real function, or completeness principle. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMapPacket namecert obligations`

Theorem 1401.6 (CauchyContinuousMap composition route)**.** *Let* $M_1 = (W_1, R_1, D_1, E_1, H_1, C_1, P_1, N_1)$ *and* $M_2 = (W_2, R_2, D_2, E_2, H_2, C_2, P_2, N_2)$ *be accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packets whose first target regular-rational row is the second source row. A public composite read is admitted only by displaying the two* $\mathsf{StreamName}^{\uparrow}$ *window surfaces* W_1, W_2*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R_1, R_2*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D_1, D_2*, the* $\mathsf{Real}^{\uparrow}$ *seal handoffs* E_1, E_2*, the componentwise* hsame *transports* H_1, H_2*, the displayed* Cont *replays* C_1, C_2*, the* Pkg *provenance rows* P_1, P_2*, and the local* NameCert *rows* N_1, N_2*. The resulting route is a concatenation of displayed Cauchy-name packet reads; it exports no host function composition, selected limit, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap composition route`

Proof. Project the first carrier and read its output through W_1, R_1, D_1, E_1. This displays a regular-rational image row and Real-seal handoff, but no detached host value or quotient stream representative.

Use the stated row match to present that displayed output as the second packet's source row. The second carrier must then reopen W_2, R_2, D_2, E_2 before any final seal is visible.

Concatenate the displayed replay rows C_1 and C_2. Componentwise transport through H_1 and H_2 preserves the two window, readback, tolerance, seal, provenance, and local naming rows without adding a new coordinate.

The public route is therefore the ordered packet read

$$W_1, R_1, D_1, E_1, H_1, C_1, P_1, N_1 \longmapsto W_2, R_2, D_2, E_2, H_2, C_2, P_2, N_2.$$

Since neither carrier stores host function composition, a selected limit, quotient stream equality, or ambient completeness, the composite read cannot export those objects. □

Theorem 1401.7 (Window composition)**.** *Let $M_1 = (W_1, R_1, D_1, E_1, H_1, C_1, P_1, N_1)$ and $M_2 = (W_2, R_2, D_2, E_2, H_2, C_2, P_2, N_2)$ be accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packets. A composite read is admitted only when the* $\mathsf{Real}^{\uparrow}$ *seal handoff exposed by E_1 is replayed as the source-side input window surface inspected by W_2, and the second packet then reopens its* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *image-readback row, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before exposing E_2. Thus composition is a finite replay of the first packet's Real-seal handoff through the second packet's stream-name and regular-rational-name rows, not a host-level composition theorem for* $\mathsf{ContinuousMap}^{\uparrow}$.

Proof. Project the two carriers from Definition 1401.1. The first packet can expose an output real only through W_1, R_1, D_1, and E_1, so there is no detached function value available to feed into the second packet.

The second packet accepts input only through its own window row W_2. A composite read therefore starts by replaying the first seal handoff through the displayed continuation route C_1, then presenting that finite read as the source-side window data inspected by W_2.

After that handoff, the second packet must still replay R_2 and D_2 before E_2 is visible. The structural rows H_2, C_2, P_2, N_2 preserve that order, so the composite remains a finite Cauchy-name replay rather than a full $\mathsf{ContinuousMap}^{\uparrow}$ composition principle. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap window composition`

Theorem 1401.8 (Regularity transport)**.** *For every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier $M = (W, R, D, E, H, C, P, N)$, componentwise* hsame *transport along H preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *image-readback row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before any* $\mathsf{Real}^{\uparrow}$ *seal handoff is exposed. Consequently a transported packet can be consumed only by replaying the same finite W, R, D regularity surface, and not by replacing it with a quotient stream, host real function, or full* $\mathsf{ContinuousMap}^{\uparrow}$ *object.*

Proof. Read the transport row H in Definition 1401.1. It is componentwise: the stream window row W, the regular-rational image row R, the dyadic tolerance ledger D, and the real-seal handoff row E are transported as named packet coordinates.

Replay the consumer route C after transport. The route can inspect the transported W, R, and D rows before it reaches E, so regularity transport is witnessed at the finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ levels.

Because no coordinate in the carrier stores a quotient stream, host function on reals, metric-continuity proof, or completeness datum, the transported read preserves only the displayed regularity surface before the $\mathsf{Real}^{\uparrow}$ seal is exposed. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap regularity transport`

Theorem 1401.9 (CauchyContinuousMap modulus boundary nonescape)**.** *Every accepted* CauchyContinuousMap$^{\uparrow}$ *packet that is read by a modulus-facing consumer must expose only the finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *regularity row,* Real$^{\uparrow}$ *seal handoff,* ContinuousMap$^{\uparrow}$ *boundary row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The read does not create a uniform-continuity modulus, metric-continuity theorem over all points, quotient-stream function, or completeness claim.*

Proof. Start with the boundary theorem, Definition 1401.4. It identifies the ContinuousMap$^{\uparrow}$ side of the packet as a finite Cauchy-name transformer boundary, not a full host function or all-points continuity theorem.

Use window composition, Definition 1401.7. A downstream read must reopen the finite stream windows, regular-rational rows, dyadic tolerances, and Real seal handoff before any composite or modulus-facing consumer can inspect the packet.

Finally apply regularity transport, Definition 1401.8. The transported packet preserves the same W, R, D regularity surface before the seal is exposed. Since no carrier coordinate stores a uniform-continuity modulus, quotient-stream function, metric-continuity theorem over all points, or completeness datum, the modulus-facing read cannot create one. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap modulus boundary nonescape`

Theorem 1401.10 (CauchyContinuousMap uniform-window obligation)**.** *Any uniform-modulus-facing consumer of an accepted* CauchyContinuousMap$^{\uparrow}$ *packet must expose the finite rows* W, R, D, E, H, C, P, N *before it may hand a regular-name transformer to* ContinuousMap$^{\uparrow}$ *or* UniformModulus$^{\uparrow}$*. The exposed route supplies stream windows, regular rational readback, dyadic tolerance checks, Real-seal handoff, componentwise transport, replay, provenance, and local naming; it does not supply a host function, quotient-stream map, all-points metric-continuity theorem, ambient completeness theorem, or detached uniform-continuity modulus.*

Proof. Start with the boundary theorem, Definition 1401.4. It identifies the ContinuousMap$^{\uparrow}$ side of the packet with a finite Cauchy-name transformer boundary whose rows are displayed through W, R, D, E, H, C, P, N.

Use window composition, Definition 1401.7. A consumer that composes or forwards the packet must reopen the stream windows, regular readback, dyadic tolerances, and seal handoff before a downstream packet can inspect the result.

Apply regularity transport, Definition 1401.8. Transport preserves the same W, R, D regularity surface before the seal is exposed, so the handoff remains a finite regular-name transformer rather than a quotient-stream map.

Finally invoke Definition 1401.9. Since the carrier has no row for a detached uniform-continuity modulus, a modulus-facing consumer can only pass the displayed finite transformer to ContinuousMap$^{\uparrow}$ or UniformModulus$^{\uparrow}$ after all eight rows are visible. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.UniformWindowObligation.CauchyContinuousMap uniform window obligation`

Theorem 1401.11 (CauchyContinuousMap uniform-modulus factorization)**.** *Every uniform-modulus-facing read of an accepted* CauchyContinuousMap$^{\uparrow}$ *packet factors through* StreamName$^{\uparrow}$ *windows* W*,* RegSeqRat$^{\uparrow}$ *readback* R*,* DyadicRatCore$^{\uparrow}$ *tolerance data* D*, the* Real$^{\uparrow}$ *seal* E*, the* ContinuousMap$^{\uparrow}$ *boundary,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N *before* UniformModulus$^{\uparrow}$ *may consume it. The factorization exports a finite Cauchy-name transformer route and not a detached uniform-continuity modulus.*

Proof. Start with Definition 1401.4. It identifies the $\mathsf{ContinuousMap}^{\uparrow}$ side as a finite Cauchy-name transformer boundary rather than a host function.

Use window composition and regularity transport, Definition 1401.7 and Definition 1401.8. They keep the stream windows, regular rational readback, dyadic tolerance ledger, and Real seal visible under composition and componentwise transport.

Finally apply the modulus-boundary nonescape and uniform-window obligation theorems. Together they require W, R, D, E, H, C, P, N before any $\mathsf{UniformModulus}^{\uparrow}$ consumer can inspect the packet, so the read factors through the displayed L10 rows. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.UniformModulusFactorization.CauchyContinuousMap uniform modulus factorization`

Theorem 1401.12 (CauchyContinuousMap completion-consumer scope)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$ *must read the same* W, R, D, E, H, C, P, N *route. The consumer receives finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance checks, a Real seal, componentwise transport, replay, provenance, and local naming; it receives no host function, selected limit, quotient stream map, all-points metric continuity theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The boundary theorem Definition 1401.4 gives the only map-facing surface: a displayed Cauchy-name transformer boundary over the carrier rows.

Window composition Definition 1401.7 and regularity transport Definition 1401.8 force any downstream read to reopen W, R, D before the Real seal E is exposed.

Apply Definition 1401.10. The same route is required for modulus-facing consumers, and a completion-facing consumer has no additional coordinate from which to select limits, quotient stream maps, all-points continuity, host functions, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap completion consumer scope`

Theorem 1401.13 (CauchyContinuousMap bridge-readback factorization)**.** *Every bridge-facing consumer of an accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$ *must factor its map-facing readback through the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *completion rows, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *boundary before any readback is exposed. The factorization exports the displayed finite Cauchy-name transformer route, not a host function, quotient stream, selected representative, all-points metric-continuity theorem, or ambient completeness row.*

Proof. Start with the Real seal handoff, Definition 1401.3. It forces every seal-facing read through the stream windows, regular rational image row, and dyadic tolerance ledger before E is visible.

Apply the ContinuousMap boundary theorem, Definition 1401.4. It identifies the map-facing side as a finite boundary row for the displayed Cauchy-name transformer, not as a host-level function or full metric-continuity object.

Use the uniform-modulus factorization and completion-consumer scope, Definition 1401.11 and Definition 1401.12. Together they require the same W, R, D, E, H, C, P, N route before any modulus-facing, completion-facing, or bridge-facing consumer can inspect the packet.

Finally route the readback through the finite completion rows. Since the carrier has no coordinate for a host function, quotient stream, selected representative, all-points metric-continuity theorem, or ambient completeness row, the bridge-facing consumer can expose only the displayed $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$ readback attached to the Cauchy-name transformer surface. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap bridge readback factorization`

Theorem 1401.14 (CauchyContinuousMap completion factorization)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the* $\mathsf{ContinuousMap}^{\uparrow}$ *boundary row, and the finite* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *consumer route before any completion read is exposed. The factorization exports only the displayed Cauchy-name transformer rows* W, R, D, E, H, C, P, N *and refuses host functions, quotient streams, selected limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap completion factorization`

Proof. Start from the carrier of Definition 1401.1. Its rows expose stream windows, regular rational readback, dyadic tolerance checks, the Real seal, componentwise transport, replay, provenance, and local naming.

Use the continuous-map boundary theorem, Definition 1401.4. It says that the map-facing side is already a finite Cauchy-name transformer boundary, not a host function or an all-points metric-continuity object.

The uniform-window obligation and uniform-modulus factorization then force the same W, R, D, E, H, C, P, N route before a downstream modulus or completion-facing reader can inspect the packet. In particular, the reader must pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the $\mathsf{ContinuousMap}^{\uparrow}$ boundary before the completion route is visible.

Finally apply Definition 1401.12. Since the completion consumer has no coordinate for selected limits, quotient streams, host functions, or ambient completeness, its only remaining route is the displayed finite $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer path attached to the Cauchy-name transformer rows. □

Theorem 1401.15 (CauchyContinuousMap separated-completion factorization)**.** *Any accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packet whose source read factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *the* $\mathsf{Real}^{\uparrow}$ *seal, and a separated completion target factors its public map through the displayed completion rows before exposing any* $\mathsf{ContinuousMap}^{\uparrow}$ *boundary. The separatedness datum is used only as a visible NameCert row attached to the target completion route; it is not quotient equality, host equality of completed points, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.SeparatedCompletionFactorization.CauchyContinuousMap separated completion factorization`

Proof. First expose the source-side regular image. Use Definition 1401.2 to place the image read on the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows, then use Definition 1401.3 to pass that read to the $\mathsf{Real}^{\uparrow}$ seal only after those rows are displayed.

Next route the completion consumer. Apply Definition 1401.12; the consumer must read the same W, R, D, E, H, C, P, N packet route and receives no host function, selected limit, quotient stream map, all-points metric continuity theorem, or ambient completeness datum.

Now apply the completion factorization theorem, Definition 1401.14. It places the completion-facing read through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$ before any completion output is public.

Finally attach separatedness as the target NameCert row. Since separatedness is only a displayed row in the target completion route, the public map can reach the $\mathsf{ContinuousMap}^{\uparrow}$ boundary only after the completion rows are visible, and the read still exports no quotient equality or host equality of completed points. □

Theorem 1401.16 (CauchyContinuousMap separated uniform-modulus completion route)**.** *Any accepted* CauchyContinuousMap$^{\uparrow}$ *packet whose consumer is both separated-completion-facing and uniform-modulus-facing must expose the same finite* W, R, D, E, H, C, P, N *route through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *the* Real$^{\uparrow}$ *seal, the* ContinuousMap$^{\uparrow}$ *boundary, the* UniformModulus$^{\uparrow}$ *handoff, and the finite* MetricCompletion$^{\uparrow}$ *or* MetricCompletionFinite$^{\uparrow}$ *consumer route before any public map read is available. The route exports a finite Cauchy-name transformer surface and refuses detached uniform-continuity moduli, selected limits, quotient streams, host functions, all-points metric-continuity theorems, and ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.SeparatedUniformModulusCompletionRoute.CauchyContinuousMap separated uniform modulus completion route`

Proof. Start with the uniform-modulus side. Apply Definition 1401.11; it forces every modulus-facing read through the displayed StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, ContinuousMap boundary, transport, replay, provenance, and local naming rows before UniformModulus$^{\uparrow}$ may consume the packet.

Next use the completion side. Apply Definition 1401.14; it places the same Cauchy-name transformer rows before the finite MetricCompletion$^{\uparrow}$ or MetricCompletionFinite$^{\uparrow}$ consumer route can expose any completion-facing output.

Finally attach the separated target route by Definition 1401.15. Separatedness remains a displayed NameCert row on the completion-facing path, so the public map read must pass through the same W, R, D, E, H, C, P, N rows before either the uniform-modulus handoff or the separated completion route is visible.

The three factorizations share the carrier inventory of Definition 1401.1. Since that inventory has no coordinate for detached uniform-continuity moduli, selected limits, quotient streams, host functions, all-points metric-continuity theorems, or ambient Real$^{\uparrow}$ completeness, the combined route cannot export any of them. □

Theorem 1401.17 (CauchyContinuousMap Real-completion scope)**.** *Every Real-completion-facing read of an accepted* CauchyContinuousMap$^{\uparrow}$ *carrier*

$$M = (W, R, D, E, H, C, P, N)$$

factors through the StreamName$^{\uparrow}$ *windows* W*, the* RegSeqRat$^{\uparrow}$ *readback row* R*, the* DyadicRatCore$^{\uparrow}$ *tolerance rows* D*, the* Real$^{\uparrow}$ *seal* E*, the* ContinuousMap$^{\uparrow}$ *boundary, and the finite* MetricCompletion$^{\uparrow}$ *or* MetricCompletionFinite$^{\uparrow}$ *consumer route before any completion output is public. The read exports no host function, quotient stream, selected limit, all-points metric-continuity theorem, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.RealCompletionScope.CauchyContinuousMap real completion scope`

↪variant `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap real completion scope`

Proof. Start with the continuous-map boundary theorem, Definition 1401.4. It identifies the map-facing side of M as a finite Cauchy-name transformer whose public rows are W, R, D, E, H, C, P, N.

Apply the uniform-window obligation and uniform-modulus factorization, Definition 1401.10 and Definition 1401.11. These force every modulus- or completion-facing read to display the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, Real seal, transport, replay, provenance, and local naming before any downstream consumer inspects the packet.

Use the completion-consumer scope theorem and the completion factorization theorem, Definition 1401.12 and Definition 1401.14. Together they place the completion read on the finite $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$ route after the same Cauchy-name transformer rows.

The carrier inventory has no coordinate for a host real function, quotient stream, selected limit, all-points metric-continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Therefore a public Real-completion read is exhausted by the displayed route and cannot export any of those objects. □

Theorem 1401.18 (CauchyContinuousMap scoped surface)**.** *For every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$*, the rows* W, R, D, E, H, C, P, N *form one scoped Cauchy-name transformer surface consumed by* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$ *readers. The surface is exactly the finite route that exposes StreamName windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real-seal handoff, componentwise transport, replay, provenance, and local naming before any downstream consumer may inspect the packet. It exports no host function, quotient stream, full metric-continuity theorem, detached uniform-continuity modulus, or completeness claim.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap scoped surface`

Proof. The carrier boundary is fixed by Definition 1401.1. Its only exposed rows are W, R, D, E, H, C, P, N, and Definition 1401.6 shows that even composite reads concatenate those displayed packet rows rather than create host-level composition.

Window discipline is supplied by Definition 1401.7. A downstream read must reopen the StreamName windows and the RegSeqRat and DyadicRatCore rows before it reaches the next Real-seal handoff.

Regularity transport is supplied by Definition 1401.8. It keeps the W, R, D surface stable under componentwise hsame transport, so the reader cannot replace the finite route with a quotient stream or host real function.

The modulus boundary is supplied by Definition 1401.9 and Definition 1401.10. Together they require $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{UniformModulus}^{\uparrow}$ consumers to use the displayed rows while refusing a detached modulus, full metric continuity, and completeness. □

Theorem 1401.19 (CauchyContinuousMap L10 sibling route)**.** *Every* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *route accepted by an L10 real-completion client first reads the* $\mathsf{StreamName}^{\uparrow}$ *source windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The client receives the displayed carrier rows* W, R, D, E, H, C, P, N *and the continuity replay route attached to them; it receives no compactness principle, selected limit, quotient stream equality, or general completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap l10 sibling route`

Proof. Project the accepted carrier of Definition 1401.1. Its source-facing rows are the finite window surface W, the regular rational image-readback row R, the dyadic tolerance ledger D, and the Real-seal handoff E.

Follow the displayed continuation route C. A completion-facing client must replay the same windows and tolerance checks before it can inspect the seal, and the componentwise transport row H preserves that order.

The remaining rows P and N package and name this finite route. Since the carrier has no coordinate for compactness, selected limits, quotient stream equality, or general completeness, the L10 client can only consume the displayed sibling route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Theorem 1401.20 (CauchyContinuousMap bridge scope)**.** *Every public bridge-scope read of an accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *carrier* $M = (W, R, D, E, H, C, P, N)$ *factors through the finite Cauchy-name transformer boundary of Definition 1401.4. The bridge-facing rows are exactly* W, R, D, E, H, C, P, N*: StreamName windows, RegSeqRat image readback, DyadicRatCore tolerance data, Real seal handoff, componentwise transport, replay, provenance, and local naming. The scope prepares the route to the* $\mathsf{ContinuousMap}^{\uparrow}$ *boundary without claiming bridge checking, a standard bridge theorem, host functions, all-points metric continuity, detached uniform-continuity moduli, quotient streams, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 1401.1. It exposes only the rows W, R, D, E, H, C, P, N, so any bridge-scope consumer has to use the same finite Cauchy-name transformer surface as the rest of the chapter.

Apply the continuous-map boundary theorem, Definition 1401.4. It identifies the $\mathsf{ContinuousMap}^{\uparrow}$ side as a boundary row for the displayed Cauchy-name transformer, not as a host function or a full metric-continuity theorem.

Use the scoped-surface theorem, Definition 1401.18. It binds W, R, D, E, H, C, P, N as the public surface consumed by $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{UniformModulus}^{\uparrow}$ readers before any downstream inspection.

The bridge scope therefore records exactly which rows face the standard boundary. Since the carrier has no coordinate for bridge checking, host functions, all-points metric continuity, detached moduli, quotient streams, selected limits, or ambient completeness, none of those claims is exported by this child route. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap bridge scope`

Theorem 1401.21 (CauchyContinuousMap regular-Cauchy boundedness route)**.** *Every accepted* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *packet whose source and image are consumed by an L10 completion client must expose the* $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ *row after the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed. The boundedness row is read as the finite induction certificate of Definition 9128.3; it does not supply a host function, quotient stream, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the L10 sibling route of Definition 1401.19. It forces the completion client to read the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal before any map-facing output is consumed.

Next use the Real-completion scope theorem, Definition 1401.17. The same accepted packet remains confined to the displayed completion route; it cannot replace the source or image by a host function, quotient stream, selected limit, or ambient completeness witness.

Finally attach the boundedness row through Definition 9128.3. That theorem identifies the bound as a finite induction certificate over the displayed stream window, regular readback, dyadic ledger, and modulus threshold. Therefore the L10 completion client receives the $\mathsf{RegularCauchyBoundedness}^{\uparrow}$ row only after the existing StreamName, RegSeqRat, DyadicRatCore, CauchyModulus, and Real rows are visible. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousMapUp.CauchyContinuousMap regular cauchy boundedness route`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuousMap}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyContinuousMap}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ image-readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, displayed hsame transports, Cont replay routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyContinuousMap}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1401.18 and the bridge-scope child Definition 1401.20 bind the finite Cauchy-name transformer rows as the public bridge-facing surface, with the uniform-window, uniform-modulus, completion-factorization, separated uniform-modulus completion, Real-completion, mature paper-side example, and cross-chapter completion-consumer rows remaining inside the same displayed package.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyContinuousMap}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.CauchyContinuousMapUp`

Bridge status: none

Not claimed: This chapter does not close full ContinuousMap$^{\uparrow}$, metric continuity, uniform continuity, completeness, quotient streams, or host real functions.

Upgrade path: The next verification step is checked theorem coverage for the same mature Cauchy-name transformer surface; the paper-side examples and cross-chapter completion consumers are included here.

Chapter 1402

A Concrete Naming Certificate for $\mathsf{IntermediateValue}^{\uparrow}$

Following Bishop and Bridges' constructive intermediate-value reading, the $\mathsf{IntermediateValue}^{\uparrow}$ packet records the finite bisection route by which opposite endpoint signs, a displayed continuity modulus, and located interval windows generate a real-facing root-location seal. The certificate is a NameCert reconstruction of the bisection argument: it consumes the real-name boundary of chapter 38, the finite branch ledger of chapter 361, and the nested-window carrier of chapter 320, but it does not claim open-cover compactness, quotient reals, host equality of real values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The packet also reads its endpoint cells through the located interval surface of chapter 394 and its function data through the pointwise-modulus carrier of chapter 130. Thus the intermediate-value route is a finite ledger over displayed source rows, not a global theorem about all continuous maps on a host interval.

Definition 1402.1 (IntermediateValue located-sign carrier)**.** *An* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (J, A, B, F, M, D, N, E, H, C, P, Q)$$

with the following visible rows:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the inspected endpoint interval and its rational and dyadic endpoint cells;*
2. *A and B are the endpoint sign rows, one recording the displayed negative-side read and one recording the displayed positive-side read after both endpoints have been transported into the same located interval packet;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the map graph, pointwise modulus, and metric-distance witness used by the sign inspections;*
4. *M is the finite modulus budget row selecting the dyadic precision at which the sign reads and midpoint tests are performed;*
5. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger whose midpoint choices retain an interval with opposite displayed signs at the requested finite window;*
6. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *row obtained by threading the retained intervals of the bisection ledger;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row offered to downstream root-location consumers;*

8. *H stores the* hsame *transports for the located interval, sign rows, continuous-map row, modulus budget, bisection ledger, nested interval row, and real seal;*
9. *C stores the displayed* Cont *read routes by which a consumer reaches exactly those rows;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *evidence row.*

The carrier predicate asserts that these rows form one finite located-sign bisection packet. The packet has no field for a host real root, a quotient of real names, an order-completeness principle, a real trichotomy oracle, or an open-cover compactness argument.

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueTasteGate single carrier alignment`

Theorem 1402.2 (IntermediateValue dyadic bisection obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *surface has five obligations:*

1. *carrier: admission of the finite* BHist *packet of Definition 1402.1;*
2. *endpoint sign exactness: the displayed negative and positive endpoint reads are routed through J, F, M, H, and C, and every transported sign read remains attached to the same located interval row;*
3. *bisection ledger exactness: each midpoint decision in D is the displayed branch row whose retained subinterval preserves the transported opposite-sign boundary;*
4. *nested-window stability: the retained branch prefix in D repacks as the* $\mathsf{NestedInterval}^{\uparrow}$ *row N through the displayed* hsame *transports and* Cont *routes;*
5. *real-seal handoff: the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed only after the located interval row, endpoint signs, continuous-map modulus row, bisection ledger, nested interval row, provenance row, and local NameCert row have all been read.*

These obligations are internal to the displayed packet and do not use compactness, quotient equality of real names, host real trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier dyadic bisection obligations`

Proof. Carrier admission is projection from the tuple V. Its coordinates are exactly the located interval, endpoint sign, continuous-map, modulus, bisection, nested-window, real-seal, transport, continuation, provenance, and local naming rows listed in Definition 1402.1.

Endpoint sign exactness reads the two endpoint signs only through the located interval and continuous-map rows. The stored hsame and Cont data move both endpoint reads into the same finite window before any midpoint branch is inspected.

Bisection ledger exactness is the finite branch invariant inherited from the displayed $\mathsf{DyadicBisection}^{\uparrow}$ row. Each retained subinterval is selected by a visible midpoint row, and the opposite-sign boundary is carried forward as part of the same packet.

Nested-window stability repacks the retained branch prefix as the $\mathsf{NestedInterval}^{\uparrow}$ row. The repacking uses only the branch ledger, endpoint cells, and transport routes already present in V.

The real-seal handoff then exposes E through the same displayed provenance. Since no coordinate of V contains compactness data, quotient-real equality, a trichotomy oracle, or a completeness theorem, none of these can participate in the seed certificate. □

Theorem 1402.3 (IntermediateValue bisection-window scope)**.** *Let* $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$ *be an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier. Every accepted root-bracket export from* V *factors through the displayed* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger* D*, the retained* $\mathsf{NestedInterval}^{\uparrow}$ *window* N*, and the source* $\mathsf{LocatedInterval}^{\uparrow}$ *row* J *before any* $\mathsf{Real}^{\uparrow}$ *seal or root-location consumer can read* E*.*

The factorization is stable under the componentwise hsame *rows in* H*, is replayed only through the displayed* Cont *row* C*, and remains attached to the same* BHist*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *evidence. It does not use unrestricted completeness, open-cover compactness, quotient real equality, a selected root, or classical trichotomy.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier bisection window scope`

Proof. Project the accepted carrier and read the bisection coordinates. The only row that stores midpoint decisions is D, and the only retained-window row obtained from those decisions is N.

The source interval is fixed by J. Each branch export therefore has to pass from J to D and then to N before the seal row E can be exposed to a root-location consumer.

Transport and replay do not create a second route. The rows H and C move and replay the same coordinates, while P and Q package and name the same finite carrier.

Since the carrier has no coordinate for unrestricted completeness, open-cover compactness, quotient equality, a selected root, or classical trichotomy, the bisection-window scope cannot export any of these objects. □

Theorem 1402.4 (IntermediateValue sign-change ledger)**.** *Let* $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$ *be an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier. Before* V *exports a root bracket, the endpoint sign rows* A, B *must be visible through the same* $\mathsf{LocatedInterval}^{\uparrow}$ *row* J*, the same* $\mathsf{ContinuousMap}^{\uparrow}$ *row* F*, and the same finite modulus budget* M*. Each retained branch row in* D *records the transported sign-change boundary used to choose the next displayed subinterval.*

Thus the sign-change information is a ledger inside the finite carrier, not a host predicate over all real points. It is transported by H*, replayed by* C*, sourced by* P*, and named by* Q*, with no appeal to real trichotomy, selected roots, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier sign change ledger`

Proof. Begin with the endpoint sign rows. The carrier stores them as A and B, and both are attached to the located interval row J through the continuous-map and modulus rows F, M.

The bisection ledger can read only those displayed signs. Each midpoint decision in D retains the subinterval whose endpoint rows still carry the transported opposite-sign boundary.

The structural rows preserve the ledger. The transport row H moves the sign rows componentwise, C replays the same route, and P, Q source and name the packet.

No carrier coordinate supplies a global sign predicate, trichotomy oracle, selected root, quotient equality, or completeness theorem. The sign-change evidence is therefore exactly the finite ledger visible before the bracket is exported. □

Theorem 1402.5 (IntermediateValue Real seal nonescape)**.** *Every accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *real-facing consumer factors through*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{DyadicBisection}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

by first reading J*, then* F, M, A, B*, then the branch ledger* D*, then the retained window* N*, and only then the seal row* E *with* H, C, P, Q*.*

Consequently the $\mathsf{Real}^{\uparrow}$ *seal is a nonescape boundary for the finite bisection route. A consumer cannot replace it by unrestricted completeness of* $\mathsf{Real}^{\uparrow}$*, an infinite nested intersection, quotient equality of real names, classical trichotomy, compactness, or an externally selected root.*

Proof. Read the consumer route from the continuation row C. The route starts at the located interval row J and the continuous-map data F, M because the endpoint sign rows are meaningful only over those displayed source cells.

Next inspect the branch and retained-window rows. The ledger D records the finite bisection choices, and N repacks the retained branch prefix as the nested-window row.

The seal row E is downstream from those rows. The structural rows H, C, P, Q transport, replay, source, and name the same route, so there is no separate Real-facing coordinate.

Since the packet contains neither a completeness principle nor an infinite intersection, quotient equality, trichotomy decision, compactness theorem, or selected root, every real-facing consumer remains inside the displayed bisection route. □

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier real seal nonescape`

Theorem 1402.6 (IntermediateValue real seal handoff)**.** *For every accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier, every* $\mathsf{Real}^{\uparrow}$ *or root-location consumer factors through the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the* $\mathsf{NestedInterval}^{\uparrow}$ *retained window, the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row, the displayed* BHist *packet, the* hsame *transport rows, the* Cont *read routes, the* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *row.*

Consequently the consumer route cannot bypass the finite located-sign bisection packet by reading a host root, an infinite nested intersection, quotient equality of real names, real trichotomy, open-cover compactness, or an ambient completeness row.

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier real seal handoff`

Proof. Start from the accepted carrier $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$. A root-location consumer can reach the real-facing row only through the continuation route C, so its first readable data are the located interval row J, the endpoint sign rows A, B, and the continuous-map row F with its finite modulus budget M.

Next route the midpoint decisions through the bisection ledger. The ledger D supplies the retained branch prefix, and Definition 1402.2 keeps every branch choice attached to the transported endpoint signs and the displayed modulus row.

Then read the nested window. The retained branch prefix repacks as the $\mathsf{NestedInterval}^{\uparrow}$ row N, so the real-facing request reaches a finite nested interval window before it reaches the seal row.

Finally expose E with its structural rows. The transports H, routes C, provenance P, and local certificate row Q bind the $\mathsf{Real}^{\uparrow}$ read to the finite packet. Every named consumer therefore factors through the stated rows, and there is no available coordinate for a host root, infinite intersection, quotient equality, trichotomy decision, compactness theorem, or completeness principle. □

Theorem 1402.7 (IntermediateValue bisection convergence handoff)**.** *For every accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier*

$$V = (J, A, B, F, M, D, N, E, H, C, P, Q),$$

if the retained branch prefix D *and nested window* N *are read through the* $\mathsf{BisectionConvergence}^{\uparrow}$ *packet supplied by Definition 1804.1, then every root-location or* $\mathsf{Real}^{\uparrow}$ *consumer factors through* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{DyadicBisection}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{BisectionConvergence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before reading* E*.*

Consequently the intermediate-value route can consume the named finite convergence certificate without introducing a selected root, quotient real equality, real trichotomy, countable choice, compactness, or an ambient completeness row.

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier bisection convergence handoff`

Proof. Project the accepted carrier $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$. By Definition 1402.6, any root-location or $\mathsf{Real}^{\uparrow}$ consumer first reaches the located interval row J, the dyadic branch ledger D, the retained nested window N, the displayed packet rows, and the local $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ evidence before it can read the seal row E.

Now read D and N through the $\mathsf{BisectionConvergence}^{\uparrow}$ carrier of Definition 1804.1. The carrier places the located interval source, dyadic branch ledger, retained nested windows, tail-diameter schedule, StreamName windows, RegSeqRat readback, and terminal Real seal in one displayed finite route.

Apply Definition 1804.3 to the retained branch prefix and nested window. The convergence packet forces every tolerance request to pass through the displayed branch depth and retained-window diameter bound before any $\mathsf{StreamName}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ readback is exposed.

Finally apply Definition 1804.4. The StreamName and RegSeqRat stages precede the terminal $\mathsf{Real}^{\uparrow}$ seal, so the original intermediate-value consumer factors through the stated $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{BisectionConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route before reading E. Since neither carrier has a coordinate for a selected root, quotient real equality, trichotomy, choice, compactness, or ambient completeness, the composite handoff exports none of them. □

Theorem 1402.8 (IntermediateValue located sign scope)**.** *Every root-bracket consumer of an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier must pass through the located interval row* J*, the endpoint sign rows* A, B*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row* F, M*, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger* D*, the* $\mathsf{NestedInterval}^{\uparrow}$ *retained window* N*, the* $\mathsf{BisectionConvergence}^{\uparrow}$ *convergence packet when that packet is supplied, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks of the convergence route, and only then the* $\mathsf{Real}^{\uparrow}$ *seal row* E *with* H, C, P, Q*.*

Thus the Real-facing route is finite and L10-safe: it is a displayed BEDC path through located signs, finite bisection, retained nested windows, convergence readback, and the terminal seal, not a route through selected roots, quotient real equality, real trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier located sign scope`

Proof. Start from the carrier definition and the dyadic bisection obligations. They place the endpoint signs inside the same located interval row J and the same continuous-map modulus row F, M before any branch ledger can be inspected.

Apply the real seal handoff. It forces every root-location consumer to read the dyadic branch ledger D and the retained nested window N before the seal row E is reachable.

Use the bisection convergence handoff for the completion-facing part of the route. When the retained branch prefix and nested window are consumed through $\mathsf{BisectionConvergence}^{\uparrow}$, the StreamName and RegSeqRat readbacks precede the terminal $\mathsf{Real}^{\uparrow}$ seal.

The structural rows H, C, P, Q transport, replay, source, and name exactly this route. Since the carrier and the convergence packet contain no selected root, quotient equality, trichotomy oracle, choice selector, compactness theorem, or ambient completeness row, the located-sign scope is finite and L10-safe. □

Theorem 1402.9 (IntermediateValue sign-bracket carrier)**.** *For every accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier*

$$V = (J, A, B, F, M, D, N, E, H, C, P, Q),$$

the located sign-bracket surface is exactly the displayed packet whose endpoint cells are read through $\mathsf{LocatedInterval}^{\uparrow}$*, whose sign inspections are read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and the finite modulus row* M*, whose branch history is read through* $\mathsf{DyadicBisection}^{\uparrow}$*, and whose retained interval stream is read through* $\mathsf{NestedInterval}^{\uparrow}$*. The structural rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *attach these reads to one local packet before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Thus the sign-bracket carrier has no separate coordinate for a host root, quotient real equality, real trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier sign bracket carrier`

Proof. Project the carrier of Definition 1402.1. The located interval row J supplies the two endpoint cells, while A and B record the opposite displayed sign reads over the same transported source interval.

The function row F and modulus row M are the only rows that can justify those sign inspections. Hence every endpoint sign read is anchored in $\mathsf{ContinuousMap}^{\uparrow}$ data before the branch ledger is consulted.

The branch row D and retained row N then record the finite bisection history and its nested-window image. The rows H, C, P, Q transport, replay, source, and name the same packet, so the carrier exposes one located sign bracket rather than independent host data.

Since no coordinate of V stores a root selector, quotient equality, trichotomy oracle, compactness theorem, or completeness theorem, no such object can be exported by the carrier. □

Theorem 1402.10 (IntermediateValue bisection nesting)**.** *Let* $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$ *be an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier. Every retained branch read in* D *that is offered to a root-location consumer must pass through the dyadic bisection route of* $\mathsf{DyadicBisection}^{\uparrow}$ *and then through the nested-window route of* $\mathsf{NestedInterval}^{\uparrow}$ *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is reachable. The route is stable under the displayed* hsame *transports and is replayed only by the* Cont *row attached to the same* BHist *packet.*

Proof. Begin with a retained branch read. The row D is the only carrier coordinate that records midpoint choices, retained subintervals, and finite branch prefixes.

The nested row N is downstream of D. It repacks the retained branch prefix as a nested interval window, and the real-facing consumer cannot reach E until that nested-window row has been displayed.

Transport stability is componentwise. The row H transports the located interval, sign rows, bisection ledger, nested window, and seal together, while C replays the same ordered route for consumers.

The provenance row P and local certificate row Q bind this route to one $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ packet. Thus the bisection nesting path is D followed by N inside the displayed BHist carrier, with no bypass through a selected root or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier bisection nesting`

Theorem 1402.11 (IntermediateValue located-sign window exactness)**.** *Let* $V = (J, A, B, F, M, D, N, E, H, C, P, Q$ *be an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier. Every endpoint sign read exported by* V *factors through the same* $\mathsf{LocatedInterval}^{\uparrow}$ *row* J*, the same* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row* F, M*, the same* $\mathsf{DyadicBisection}^{\uparrow}$

branch ledger D, and the same $\mathsf{NestedInterval}^{\uparrow}$ *retained window N before any* $\mathsf{Real}^{\uparrow}$ *seal is reachable. The factorization is stable under the displayed* BHist *packet, componentwise* hsame *transports,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *route.*

Thus the two endpoint signs are exact reads of one located-sign window rather than independent host predicates, selected roots, quotient real equalities, trichotomy decisions, compactness witnesses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness data.*

Proof. Start with the accepted carrier and project the rows listed in Definition 1402.1. The endpoint sign rows A and B are attached to the located interval row J, and Definition 1402.9 identifies this attachment as the displayed sign-bracket surface.

The function and modulus data are read through F and M. By the dyadic bisection obligations of Definition 1402.2, every sign inspection used by the branch ledger remains attached to that same located interval and continuous-map route.

Apply Definition 1402.10. The retained branch read then passes through D and N before the seal row E can be consumed, with H and C preserving the same rows componentwise.

Finally use Definition 1402.6. The real-facing route includes the provenance row P and local naming row Q, so there is no coordinate from which a selected root, quotient equality, trichotomy oracle, compactness theorem, or ambient completeness principle can alter the located-sign window. □

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier located sign window exactness`

Theorem 1402.12 (IntermediateValue modulus branch transport)**.** *Let $V = (J, A, B, F, M, D, N, E, H, C, P, Q)$ be an accepted* $\mathsf{IntermediateValue}^{\uparrow}$ *carrier. Transporting the continuous-map modulus row F, M along the displayed* hsame *data preserves the retained branch ledger D and its nested-window image N. Consequently every transported root-location request still reads the same* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint cells, the same dyadic midpoint choices, the same retained branch window, and the same* $\mathsf{Real}^{\uparrow}$ *handoff route.*

The transport does not introduce a selected root, quotient real equality, trichotomy, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness; it only replays the existing finite branch packet through H, C, P, and Q.*

Proof. Read the transport row H of the carrier. Its component for F, M is stored with the same located interval row J and the same endpoint sign rows A, B, so a transported modulus request cannot detach the sign test from the source window.

The dyadic branch ledger is already bound to those sign tests by Definition 1402.2. Therefore transporting the modulus row preserves the branch choices recorded in D rather than recomputing them from host-level sign predicates.

Next apply Definition 1402.10. The retained branch prefix is repacked as N through the same continuation route C, and the route to the seal row E still passes through D followed by N.

Finally Definition 1402.6 identifies the only real-facing export. Since the carrier has no coordinate for a root selector, quotient equality, trichotomy, compactness, or completeness, modulus transport cannot add any of these objects to the retained branch packet. □

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier modulus branch transport`

Theorem 1402.13 (IntermediateValue NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$ *obligation surface for the carrier of Definition 1402.1 consists exactly of:*

1. *admission of the finite located sign-bracket carrier over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*;*

2. *endpoint sign transport through* BHist, hsame, *and* Cont *without detaching the two signs from the same located interval row;*
3. *bisection branch nesting through* DyadicBisection$^{\uparrow}$ *and* NestedInterval$^{\uparrow}$ *before the real-facing seal is exposed;*
4. *provenance and local naming through* Pkg *and* NameCert *for the same displayed rows.*

No additional obligation asks for a selected root, quotient real equality, trichotomy, open-cover compactness, countable choice, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier namecert obligations`

Proof. The carrier obligation is Definition 1402.9. It identifies the displayed rows and excludes every host-level datum not present in the finite packet.

Endpoint sign transport is the second clause of Definition 1402.2. The two sign rows move only through H and are replayed only through C over the same located interval and continuous-map data.

Bisection branch nesting is Definition 1402.10. It forces every retained branch read through D and then N before the Real$^{\uparrow}$ seal can be consumed.

Finally, P and Q supply the provenance and local naming rows for exactly those carrier coordinates. Since the listed rows exhaust the packet, the excluded host objects cannot become local obligations. □

Theorem 1402.14 (IntermediateValue obligation closure package)**.** *The displayed located-sign carrier of Definition 1402.1, the dyadic bisection obligations of Definition 1402.2, the real seal handoff of Definition 1402.6, the bisection convergence handoff of Definition 1402.7, the modulus branch transport of Definition 1402.12, and the NameCert obligation surface of Definition 1402.13 form the complete obligation package for the current finite* IntermediateValue$^{\uparrow}$ *surface.*

Every root-location or Real$^{\uparrow}$ *consumer must therefore read the same located interval, endpoint sign, continuous-map modulus, dyadic branch ledger, nested window, real seal,* hsame, Cont, Pkg, *and local* NameCert *rows before any real-facing handoff is exposed.*

Proof. Project the carrier rows of Definition 1402.1. The dyadic bisection theorem fixes the carrier, endpoint signs, branch ledger, nested-window stability, and real-seal obligation clauses.

The handoff theorems then bind every consumer route. The real seal handoff prevents a direct read of E, while the bisection convergence handoff routes the retained branch and nested-window rows through the displayed convergence packet when that packet is supplied.

Finally apply modulus branch transport and the NameCert obligations theorem. These results account for the transported modulus branch and the local naming rows, so the listed clauses exhaust the finite obligation surface and leave no coordinate for a selected root, quotient real equality, trichotomy, compactness, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier obligation closure package`

Theorem 1402.15 (IntermediateValue scoped carrier dependency package)**.** *The finite bisection route of an accepted* IntermediateValue$^{\uparrow}$ *carrier is scoped by the displayed dependency package*

LocatedInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicBisection$^{\uparrow}$, NestedInterval$^{\uparrow}$, BisectionConvergence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRa

Every root-location or Real$^{\uparrow}$ *consumer must read the located interval row, the continuous-map modulus row, the dyadic branch ledger, the nested retained window, the bisection-convergence handoff*

when supplied, the StreamName and RegSeqRat readbacks of that handoff, the terminal Real seal, and the structural BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before it reaches the exported bracket.*

This dependency package is the whole scoped carrier boundary for the current IntermediateValue$^{\uparrow}$ *chapter. It does not add open-cover compactness, quotient real equality, host equality of values, selected roots, trichotomy, countable choice, infinite nested intersections, or ambient completeness of* Real$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueScopedCarrierDependencyPackage`

Proof. Start with Definition 1402.14. It already binds the located-sign carrier, dyadic bisection obligations, Real-seal handoff, bisection convergence handoff, modulus transport, and local NameCert obligations into one finite package.

The carrier rows name the scoped dependencies. The located interval, continuous map, dyadic bisection, nested interval, Real seal, BHist, hsame, Cont, Pkg, and NameCert rows are visible in Definition 1402.1, while Definition 1402.7 supplies the BisectionConvergence$^{\uparrow}$, StreamName$^{\uparrow}$, and RegSeqRat$^{\uparrow}$ part of the route.

Combining these two reads gives the stated scoped boundary. Since the cited theorems exclude selected roots, quotient real equality, trichotomy, compactness, choice, infinite intersections, and ambient completeness, the dependency package exports none of those objects. □

Theorem 1402.16 (IntermediateValue public export route)**.** *Every accepted* IntermediateValue$^{\uparrow}$ *public consumer reads the located-sign carrier of Definition 1402.1, the dyadic bisection obligations of Definition 1402.2, the bisection-window scope of Definition 1402.10, the Real seal nonescape route of Definition 1402.6, and the scoped carrier dependency package of Definition 1402.15 before treating the bisection route as the public BEDC intermediate-value export.*

The public export is only this finite route through located interval, endpoint sign, continuous-map modulus, dyadic branch ledger, nested-window, Real seal, BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It does not provide a selected root, quotient real equality, host equality of values, trichotomy, countable choice, open-cover compactness, infinite nested intersections, or ambient completeness of* Real$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier public export route`

Proof. Start with the located-sign carrier and the dyadic bisection obligations. The carrier displays the interval, endpoint signs, continuous-map modulus, branch ledger, retained window, Real seal, transport, replay, provenance, and local naming rows, while the bisection obligations force every branch read through the same finite packet.

Next apply the bisection nesting theorem and the Real seal handoff. They place the retained bisection window before the terminal real-facing row and prevent a consumer from reading the Real$^{\uparrow}$ seal as detached endpoint data.

Finally use the scoped carrier dependency package. It binds the StreamName, RegSeqRat, bisection-convergence, structural, and local naming rows to the same carrier surface, so a public consumer receives the intermediate-value route only as the displayed finite BEDC export and not as any of the excluded host principles. □

Theorem 1402.17 (IntermediateValue bridged finite bisection export)**.** *Every bridge-facing read of* IntermediateValue$^{\uparrow}$ *preserves the finite located-sign bisection route through*

LocatedInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicBisection$^{\uparrow}$, NestedInterval$^{\uparrow}$, BisectionConvergence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRa

The bridge reads the located interval, endpoint signs, continuous-map modulus, dyadic branch ledger, retained nested interval, bisection-convergence handoff, stream windows, regular-rational readback, terminal real seal, componentwise transport, displayed replay, package provenance, and local naming rows before exporting the intermediate-value packet. It exports no selected root, quotient real equality, trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the public export route of Definition 1402.16. It already requires a public consumer to read the located-sign carrier, dyadic bisection obligations, bisection-window scope, Real seal handoff, and scoped carrier dependency package before treating the packet as an intermediate-value export.

The scoped carrier dependency package of Definition 1402.15 supplies the bridge-facing rows. It binds the located interval, continuous-map, dyadic bisection, nested interval, bisection-convergence, StreamName, RegSeqRat, Real seal, hsame, Cont, Pkg, and NameCert data into one finite dependency package.

A bridge-facing read can therefore export only that package. The cited route has no carrier coordinate for selected roots, quotient real equality, trichotomy, open-cover compactness, countable choice, or ambient completeness, so the bridge cannot supply any of those excluded principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$.

Constructive story: An $\mathsf{IntermediateValue}^{\uparrow}$ packet is generated from a located interval row, two endpoint sign rows, a $\mathsf{ContinuousMap}^{\uparrow}$ modulus row, a finite $\mathsf{DyadicBisection}^{\uparrow}$ branch ledger, a $\mathsf{NestedInterval}^{\uparrow}$ retained window, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$, maturePackageClosed)

Scope closed: The mature paper surface is the bridged finite located-sign bisection export of Definition 1402.17, together with the public export route of Definition 1402.16 and the scoped dependency package of Definition 1402.15.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntermediateValue}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.IntermediateValueUp.IntermediateValueCarrier\_located\_sign\_scope`

Bridge status: paperBridge

Not claimed: The chapter does not close open-cover compactness, quotient reals, host equality of real values, real trichotomy, selected roots, infinite nested intersections, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: No further paper theory-axis upgrade is pending; bridge checking remains formal-axis work for the same finite located-sign bisection route.

Chapter 1403

A Concrete Naming Certificate for $\mathsf{FilteredCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Bourbaki, *General Topology*, the FilteredCauchyCompletio packet records the finite BEDC route by which Cauchy filters reach the Real-completion interface. It is built over $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, RealCompletionExactBoundaryUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

The dependency anchors are the finite filter surface of chapter 333, the sequence-space handoff of chapter 406, and the Real boundary of chapter 1120. The uniform criterion supplies the common-tail test before any route reaches the Real seal, while stream names, regular rational rows, and dyadic ledgers keep the consumer boundary finite.

Definition 1403.1 (Filtered Cauchy completion carrier)**.** *A* $\mathsf{FilteredCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row carrying the finite filterbase and common-window evidence;*
2. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row that tests the same finite filter windows by uniform tails;*
3. *S is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff row exposing the associated* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readings;*
4. *Q is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* RealCompletionExactBoundaryUp *route reached from those same finite rows;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the displayed route;*
6. *H is the componentwise* hsame *transport row for F, U, S, Q, E;*
7. *C is the* Cont *continuation row that replays the accepted finite windows without adding host data;*
8. *P is the* Pkg *provenance over the listed completion, filter, uniform, stream, regular-rational, dyadic, transport, and seal rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FilteredCauchyCompletion}^{\uparrow}}$ *row.*

The accepted carrier exposes only these rows. It has no coordinate for a quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.FilteredCauchyCompletionUp.TasteGate.FilteredCauchyCompletionTasteGate single carrier alignment`

Theorem 1403.2 (Filtered completion carrier habitation)**.** *Whenever the rows $F, U, S, Q, E, H, C, P, N$ occur in one finite* BHist *with the dependencies listed in Definition 1403.1, they assemble an accepted* FilteredCauchyCompletion$^{\uparrow}$ *carrier. Conversely, every accepted carrier projects to exactly those nine displayed rows.*

Proof. The forward direction reads the finite history in the carrier order. The CauchyFilter$^{\uparrow}$ row supplies the finite filterbase, the uniform criterion row supplies the common-tail test, the sequence-space row supplies the stream, regular-rational, and dyadic readings, and the completion route supplies the Real-boundary handoff and terminal seal.

Transport, continuation, package provenance, and local naming are the final four displayed coordinates. Repacking these rows gives the named carrier.

For the reverse direction, project an accepted carrier by its definition. No extra coordinate is available, so the projection is exhausted by the nine listed rows. □

Theorem 1403.3 (Common-filter classifier transport)**.** *Two accepted* FilteredCauchyCompletion$^{\uparrow}$ *carriers classify together only by a displayed common* CauchyFilter$^{\uparrow}$ *refinement whose uniform-tail test, sequence space handoff, Real-completion route, terminal Real seal,* hsame *transport,* Cont *continuation,* Pkg *provenance, and local* NameCert *rows are transported componentwise.*

Proof. Project both carriers to their F rows and compare those rows by the common finite filter refinement supplied by CauchyFilter$^{\uparrow}$. The classifier has no access to a host filter or quotient of streams.

Once the filter rows share a displayed refinement, transport the U, S, Q, E rows componentwise along H and replay the same reads through C. The package row records that these transported reads remain on the same finite dependency anchors.

Repacking the transported rows gives the classifier judgement for the two carriers. Reading such a judgement back yields exactly the common filter refinement and the listed componentwise transports. □

Theorem 1403.4 (Uniform-tail exactness)**.** *For an accepted carrier $K = (F, U, S, Q, E, H, C, P, N)$, every completion-facing read of Q first passes through the uniform-tail row U attached to the same* CauchyFilter$^{\uparrow}$ *row F and the same* CauchySequenceSpace$^{\uparrow}$ *handoff S. No Real seal consumer can bypass U by using a selected subsequence, quotient stream, or host completion point.*

Proof. Read the carrier coordinates. The route to Q is listed after F, U, S, so the completion-facing consumer sees the filterbase, uniform-tail test, and sequence-space handoff before it can read the Real-completion boundary.

The continuation row C only replays the accepted finite windows, and the transport row H only compares displayed coordinates. Neither row introduces a selected subsequence, quotient stream, or host completion point.

Thus the uniform-tail row is the exact gate for completion-facing reads. Any consumer that bypasses it has left the displayed carrier. □

Theorem 1403.5 (Completion handoff factorization)**.** *Every* RealCauchyCompletion$^{\uparrow}$ *or* RealCompletionExactBoun *handoff exported by an accepted* FilteredCauchyCompletion$^{\uparrow}$ *carrier factors through the rows* F, U, S, H, C, P, N *before reaching* Q *and the terminal seal* E*.*

Proof. Start from the carrier decomposition. The handoff row Q is not independent: it is stored as the route reached from the same finite filter, uniform-tail, sequence-space, transport, continuation, provenance, and naming rows.

The sequence-space row supplies the visible stream-name, regular-rational, and dyadic readings. The uniform-tail row checks those readings against the common filter classifier before the completion boundary is exposed.

Consequently every exported completion handoff factors through F, U, S, H, C, P, N and only then reaches Q and E. □

Theorem 1403.6 (Filtered completion no-escape)**.** *An accepted* FilteredCauchyCompletion$^{\uparrow}$ *carrier gives no route to quotient streams, selected limits, host real equality, countable choice, or ambient completeness. Every consumer of the packet is confined to the finite rows* $F, U, S, Q, E, H, C, P, N$*.*

Proof. The excluded objects are not coordinates of Definition 1403.1. A consumer can project the filter row, uniform criterion row, sequence-space handoff, completion route, Real seal, transport, continuation, package, and local naming row, and nothing else.

The previous factorization theorem shows that even completion-facing reads must pass through those displayed rows. The common-filter transport theorem shows the same boundary after classifier comparison.

Therefore any route requiring quotient streams, selected limits, host real equality, countable choice, or ambient completeness lies outside the accepted carrier. □

Closure status for NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$**.**

Constructive story: The filtered Cauchy completion packet is generated from a finite Cauchy filter row, a uniform-tail criterion row, a Cauchy sequence-space handoff, a Real-completion boundary route, a terminal Real seal, and the displayed hsame, Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: This chapter records the finite carrier and five seed obligations for routing Cauchy filters to the Real-completion boundary through displayed BEDC rows.

Formal verification: FormalStatus(NameCert$_{\text{FilteredCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: It does not claim quotient-stream equality, selected limits, host real equality, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Reaching obligationClosure requires Lean-backed carrier, classifier-transport, uniform-tail, handoff-factorization, and no-escape targets for the displayed packet.

Chapter 1404

A Concrete Naming Certificate for RegularCauchyEquivalence$^{\uparrow}$

The RegularCauchyEquivalence$^{\uparrow}$ packet records the finite equality boundary for regular Cauchy names before a Real$^{\uparrow}$ seal is consumed. Following Bishop–Bridges constructive analysis, two regular rational names are compared by common finite windows, dyadic tolerance ledgers, displayed transports, and local certificate rows, not by a quotient of streams or by host real equality.

The route is the Phase Real and Completion sibling boundary. It reads regular-name rows from chapter 301, finite schedules from chapter 126, dyadic tolerance data from chapter 291, zero-tail handoff data from chapter 380, the seal boundary from chapter 38, and the locatedness consumer interface from **??**. The chapter isolates the equivalence surface used between those rows so later completion and locatedness consumers can cite one visible NameCert packet.

Definition 1404.1 (RegularCauchyEquivalence finite-window classifier)**.** *A* RegularCauchyEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, Y, W, D, Z, R, E, H, C, P, N)$$

with the following rows:

1. *X and Y are the two displayed* RegSeqRat$^{\uparrow}$ *source rows to be compared;*
2. *W is the common* StreamName$^{\uparrow}$ *window schedule that selects paired finite observations from X and Y;*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for every selected pair of observations;*
4. *Z is the* NullSequence$^{\uparrow}$ *zero-tail handoff row when the displayed difference stream is consumed by a real-equality classifier;*
5. *R is the pair of* Real$^{\uparrow}$ *seal rows through which the two regular names may later be read;*
6. *E is the local finite equivalence classifier, asserting that each selected common window satisfies the displayed dyadic tolerance bound;*
7. *H records the componentwise* hsame *transports for the two source rows, schedule, tolerance ledger, seal rows, and classifier row;*
8. *C records the displayed* Cont *routes by which consumers inspect those rows;*
9. *P is the* Pkg *provenance for the two regular names, the common-window schedule, the dyadic ledger, the zero-tail handoff, the real-seal rows, and the consumer routes;*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}$ *row.*
The carrier stores only finite scheduled comparisons. It has no quotient stream equality, no selected limit, no countable-choice row, no trichotomy row, no host real equality, and no ambient real-completeness coordinate.

Theorem 1404.2 (RegularCauchyEquivalence refinement transport)**.** *Let* $(X, Y, W, D, Z, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet. If the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the common* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff row, and the* $\mathsf{Real}^{\uparrow}$ *seal rows are transported componentwise by the displayed* hsame *rows in* H *and read only through the displayed* Cont *routes in* C*, then the transported rows repack as an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet with the same finite-window classifier shape.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalence refinement transport`
Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.TasteGate.RegularCauchyEquivalence refinement transport`

Proof. Project the carrier of Definition 1404.1. Its whole comparison surface is the pair of regular Cauchy rows, one common schedule, one dyadic tolerance ledger, one zero-tail handoff row, the seal rows, and the finite classifier row.

Transport the two source rows, the schedule, and the tolerance ledger through the stated hsame rows. The transported schedule still selects paired finite observations, and every selected pair is checked against the transported dyadic ledger.

Transport the $\mathsf{NullSequence}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows only as stored rows beside the same finite comparison. Since the proof does not add a quotient equality, selected limit, countable-choice selector, trichotomy branch, host real equality, or ambient completeness datum, the transported inventory is exactly another instance of the same carrier definition. □

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalenceCarrier refinement transport`

Theorem 1404.3 (RegularCauchyEquivalence real classifier boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet factors first through the two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, then through the common* $\mathsf{StreamName}^{\uparrow}$ *windows, then through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then through the displayed* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff when equality consumption is requested, and only then reads the stored* $\mathsf{Real}^{\uparrow}$ *seal rows. The consumer receives no quotient stream equality, selected limiting real, host real equality, countable choice, trichotomy, or ambient completeness witness from the equivalence packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.TasteGate.RegularCauchyEquivalence real classifier boundary`

Proof. Start with the finite classifier of Definition 1404.1. The seal row R is stored after X, Y, W, D, Z, and E, so a seal-facing read must first pass through the two regular-name rows, the common finite-window comparison, and the available zero-tail handoff.

Use Definition 1404.2 for stability under displayed transport. It keeps the consumer route attached to the same scheduled windows and dyadic ledger while moving along hsame and Cont rows.

The carrier lists no quotient stream equality, selected limit, host real equality, choice row, trichotomy row, or completeness row. Thus the boundary between regular Cauchy names and real seals is exactly the displayed finite-window route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalence real classifier boundary`

Theorem 1404.4 (RegularCauchyEquivalence NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyEquival}}$ *obligation surface consists of five finite rows:*

1. *carrier habitation by two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, a common* $\mathsf{StreamName}^{\uparrow}$ *window schedule, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff row,* $\mathsf{Real}^{\uparrow}$ *seal rows, and displayed provenance;*
2. *classifier by finite common-window dyadic tolerance checks;*
3. *transport stability under the stored* hsame *rows and displayed* Cont *routes;*
4. *seal-boundary exactness, stating that every* $\mathsf{Real}^{\uparrow}$ *read is reached only after the common-window classifier row and the displayed zero-tail handoff when equality consumption is requested;*
5. *ledger refusal, stating that quotient stream equality, host real equality, countable choice, selected limits, trichotomy, and ambient real completeness are not packet coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEquivalenceUp.RegularCauchyEquivalenceTasteGate single carrier alignment`

Proof. Carrier habitation is the displayed tuple in Definition 1404.1. It supplies two regular Cauchy names, a common schedule, dyadic tolerance rows, a zero-tail handoff row, seal rows, transports, continuations, package provenance, and the local naming row.

The classifier row is finite because it reads only windows selected by W and tolerances recorded in D. No comparison is made outside the scheduled observations.

Transport stability is Definition 1404.2. Seal-boundary exactness is Definition 1404.3. Together they keep every consumer route on the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The refusal row is immediate from the carrier inventory: there is no coordinate that can encode a quotient stream equality, selected limit, countable-choice selector, trichotomy decision, host real equality, or ambient completeness principle. □

Theorem 1404.5 (RegularCauchyEquivalence sibling route)**.** *Any completion-facing or locatedness-facing consumer that uses a* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet must read the finite classifier of Definition 1404.1. Its route is exhausted by* $X, Y, W, D, Z, R, E, H, C, P, N$*: the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, common* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff,* $\mathsf{Real}^{\uparrow}$ *seal rows, classifier row, transports, continuations, package provenance, and local* $\mathsf{NameCert}$ *row. No quotient of Cauchy streams, host real equality, selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness data is available to that consumer.*

Proof. Project an accepted packet to the tuple of Definition 1404.1. This tuple is already the full finite inventory available to completion-facing and locatedness-facing consumers.

Apply Definition 1404.4. The NameCert rows bind carrier habitation, finite common-window classification, transport stability, seal-boundary exactness, and the ledger refusal inside the same packet.

The route to $\mathsf{LocatedReal}^{\uparrow}$ may read the same regular sources and seal rows used by **??**, but it does not gain an independent real equality or completion principle. Therefore every downstream read remains a nonquotient finite-window comparison over $X, Y, W, D, Z, R, E, H, C, P, N$. □

Theorem 1404.6 (RegularCauchyEquivalence scoped primitive route)**.** *Any completion-facing or locatedness-facing consumer of an accepted* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *packet must read exactly the tuple*

$$(X, Y, W, D, Z, R, E, H, C, P, N)$$

through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows named in Definition 1404.1. This scoped primitive route is the whole access path to the finite equivalence classifier; no quotient of streams, host real equality, selected limit, choice row, trichotomy row, or ambient completeness row is available.*

Proof. First apply refinement transport, Definition 1404.2. It keeps the two regular names, common window, dyadic ledger, null-tail handoff, real seals, classifier, transport, replay, provenance, and local name inside the same finite packet.

Next use the real classifier boundary of Definition 1404.3. It forces every seal-facing read through the regular-name, window, tolerance, and zero-tail rows before the $\mathsf{Real}^{\uparrow}$ seal rows can be inspected.

Finally combine the NameCert obligation surface and sibling route, Definition 1404.4 and Definition 1404.5. These theorems exhaust the consumer path by the displayed tuple, so the scoped primitive route contains exactly the listed BEDC rows and no hidden quotient, host equality, selected limit, choice, trichotomy, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a common $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{NullSequence}^{\uparrow}$ zero-tail handoff row, $\mathsf{Real}^{\uparrow}$ seal rows, a finite equivalence classifier, displayed hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1404.1, Definition 1404.2, Definition 1404.3, Definition 1404.4, Definition 1404.5, and Definition 1404.6 display the obligation surface over RegSeqRat, StreamName, DyadicRatCore, NullSequence, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, a public completion quotient, host real equality, selected limits, countable choice, trichotomy, or ambient real completeness.

Upgrade path: Remaining work is Lean verification and bridge checking for the same finite regular-Cauchy equivalence route, not further paper-axis maturation.

Chapter 1405

A Concrete Naming Certificate for $\mathsf{MonotoneSubsequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{MonotoneSubsequence}^{\uparrow}$ packet records monotone subsequences as displayed finite unary selector rows. It does not select an infinite subsequence by countable choice. The packet sits between the regular-Cauchy subsequence surface of chapter 381, the cofinal index witness of chapter 405, and the monotone-Cauchy route of chapter 438. Its endpoint side is read through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 1405.1 (Monotone subsequence carrier)**.** *A* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row whose inspected finite windows are being selected.*
2. *M is the finite monotone unary selector ledger; each displayed entry records a selector value, its source index, and the comparison row certifying monotonicity against the previously displayed selector entries.*
3. *Q is the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *evidence row for the inspected thresholds, limited to the thresholds actually present in the packet.*
4. *W is the finite family of selected* $\mathsf{StreamName}^{\uparrow}$ *windows read from S through the ledger M and the cofinal evidence Q.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger extracted from the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from W and D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the selected regular-rational handoff has been displayed.*
8. *H stores componentwise* hsame *transport for the source row, monotone selector ledger, cofinality evidence, selected windows, dyadic observations, regular handoff, Real seal, continuation, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from S, M, Q to W, from W to D, from D to R, and from R to E.*

10. *P is the* Pkg *provenance over* $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host subsequence equality test, a countable-choice selector, a quotient stream, an ambient completeness principle, or a hidden infinite monotone-subsequence object.

Lean4 ✓ `BEDC.Derived.MonotoneSubsequenceUp.TasteGate.MonotoneSubsequenceTasteGate single carrier alignment`

Theorem 1405.2 (Monotone subsequence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *carrier* $U = (S, M, Q, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *monotone selector stability for the finite unary selector ledger M under the recorded* hsame *transports;*
3. *cofinal coverage for every inspected threshold listed in Q;*
4. *selected-window* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through W, D, R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is consumed;*
5. *non-escape from host subsequence equality, countable choice, quotient-stream equality, and ambient completeness.*

Proof. Project the carrier of Definition 1405.1. The source row S, selector ledger M, cofinality evidence Q, selected windows W, dyadic observations D, handoff R, seal E, and structural rows H, C, P, N are the only coordinates available to the classifier.

The monotone and cofinal faces are finite. Selector stability reads the displayed comparisons inside M, while cofinal coverage reads only the inspected thresholds recorded in Q. No step quantifies over an unbounded family of undisplayed requests.

The analytic endpoint is downstream from the same finite route. The selected $\mathsf{StreamName}^{\uparrow}$ windows W produce the dyadic ledger D, the dyadic ledger produces the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, and the Real seal E is available only after that handoff.

Finally inspect the excluded objects. Host subsequence equality, countable choice, quotient streams, and ambient completeness are not rows of U, and the structural rows only transport, replay, source, and name the displayed packet. The five listed obligations therefore exhaust the local NameCert surface. □

Theorem 1405.3 (Monotone subsequence regular-Cauchy handoff)**.** *Every accepted* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *packet hands its selected monotone finite windows to* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Equivalently, every seal-facing route factors through*

$$S, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Begin with the selected-window rows. The continuation route records that W is read from the source regular-Cauchy subsequence row S, the monotone selector ledger M, and the cofinal threshold evidence Q.

Next read the regular-rational handoff. The dyadic observation ledger D is extracted from the selected StreamName windows, and the row R is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ handoff reached from those dyadic observations.

Only then is the seal row available. The row E is part of the packet as the Real-facing endpoint reached through R, while H, C, P, N only preserve the same route by transport, replay, provenance, and local naming.

Thus a consumer that tries to use the $\mathsf{Real}^{\uparrow}$ seal without the selected-window $\mathsf{RegSeqRat}^{\uparrow}$ handoff leaves the accepted $\mathsf{MonotoneSubsequence}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneSubsequence}^{\uparrow}$ packet is generated from a source regular-Cauchy subsequence row, a finite monotone unary selector ledger, inspected-threshold cofinality evidence, selected StreamName windows, DyadicRatCore observations, a RegSeqRat handoff, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1405.1, Definition 1405.2, and Definition 1405.3 declare the finite selector carrier, its NameCert obligations, and the regular-Cauchy handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim Bolzano-Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host subsequence equality, Lean verification, bridge checking, or RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, monotone selector stability, inspected-threshold cofinal coverage, selected-window RegSeqRat handoff, and seal-route non-escape for the same packet.

Chapter 1406

A Concrete Naming Certificate for $\mathsf{SpeckerSequence}^{\uparrow}$

Following Specker's constructive counterexample tradition and Bishop-style finite evidence, the $\mathsf{SpeckerSequence}^{\uparrow}$ packet records a monotone bounded computable rational stream whose supremum is not exposed as a located $\mathsf{Real}^{\uparrow}$ datum without an additional location witness. The carrier sits in the Phase Real and Completion frontier between the regular rational names of chapter 301, the stream schedules of chapter 126, the dyadic tolerance ledgers of chapter 291, the finite real seal boundary of chapter 38, and the selector discipline of chapter 1405.

The Cauchy-criterion sibling boundary of chapter 2255 supplies the tail-window reading discipline used to inspect bounded monotone finite windows without exposing a located supremum.

Definition 1406.1 (SpeckerSequence carrier)**.** *A* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, D, M, B, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the computable rational stream by regular finite windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that exposes the inspected windows of R.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying the rational bounds and tolerances read from those windows.*
4. *M is the finite monotone ledger, recording that each displayed inspected value is below the later displayed values in the schedule.*
5. *B is the boundedness ledger, recording a displayed lower rational row and upper rational row for every inspected window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only as a finite readback boundary for displayed regular windows, not as a located supremum.*
7. *H stores componentwise* hsame *transport for the source, schedule, dyadic, monotone, boundedness, seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from R, W into D, from D into M and B, and from the regular readback surface into E.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for located supremum data, Dedekind cuts, countable choice, trichotomy, ambient completeness, or a selected limiting real.

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceUp`

Theorem 1406.2 (SpeckerSequence carrier exposure)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet exposes exactly the rows $R, W, D, M, B, E, H, C, P, N$ of Definition 1406.1. A consumer may inspect the computable rational stream only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bounds, the monotone and boundedness ledgers, the finite* $\mathsf{Real}^{\uparrow}$ *seal row, and the displayed structural rows.*

Proof. Project the carrier tuple. The source row, schedule row, dyadic ledger, monotone ledger, boundedness ledger, seal row, transport row, continuation row, provenance row, and local naming row are the complete public inventory.

The route is finite at every stage. The stream is read through R and W, rational bounds are read through D, monotonicity and boundedness are read through M and B, and the seal row E is reached only after those rows have been displayed.

Since the carrier lists no located supremum, Dedekind cut, selected limit, countable-choice selector, trichotomy branch, or completeness row, those objects are not part of the exposed packet. □

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceCarrier exposure`

Theorem 1406.3 (SpeckerSequence classifier transport)**.** *Let $S = (R, W, D, M, B, E, H, C, P, N)$ be an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier. Componentwise* hsame *transport along H and displayed* Cont *replay along C preserve the classifier shape of S: the transported packet is still compared by the same regular-source row, stream schedule, dyadic ledger, monotone ledger, boundedness ledger, finite seal row, provenance row, and local naming row.*

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceClassifier transport`

Proof. Read the classifier from Definition 1406.1. It depends only on the displayed rows of the packet, not on an ambient order or a host real equality test.

Transport the source, schedule, dyadic, monotone, boundedness, and seal rows componentwise through H. The continuation row C records the same replay order after transport: regular windows first, dyadic observations next, monotone and boundedness ledgers next, and the finite seal last.

The transported inventory therefore repacks as the same carrier shape. No transported coordinate can introduce located-supremum data, countable choice, trichotomy, Dedekind-cut equality, or completeness, because no such coordinate is present before transport. □

Theorem 1406.4 (SpeckerSequence monotone bounded ledger)**.** *In every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet, the monotone ledger M and boundedness ledger B are read before any* $\mathsf{Real}^{\uparrow}$ *seal consumer can inspect E. Thus every seal-facing route factors through*

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.SpeckerSequenceUp.SpeckerSequence monotone bounded ledger`

Proof. Begin with the source route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R and $\mathsf{StreamName}^{\uparrow}$ schedule W select the finite rational observations inspected by the packet.

Next read the dyadic ledger. The row D supplies the rational bounds and tolerance data used to compare the selected observations.

Now read the monotone and boundedness ledgers. The row M records the displayed monotone comparisons, while B records the displayed lower and upper rational bounds for the same inspected windows.

Only after those rows have been exposed can the finite $\mathsf{Real}^{\uparrow}$ seal row E be consumed. The structural rows H, C, P, N preserve this route and do not add a selected supremum or an ambient completeness witness. □

Theorem 1406.5 (SpeckerSequence finite-stage induction)**.** *For every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet* $S = (R, W, D, M, B, E, H, C, P, N)$*, induction along the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *lets any consumer read only the current finite rational window of* R*, the corresponding dyadic tolerance row of* D*, the monotone ledger prefix of* M*, the boundedness ledger prefix of* B*, and the transported* BHist *provenance before the consumer reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The induction route introduces no located supremum, selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Induct on the finite schedule prefix displayed by W. In the base case, the consumer sees only the initial inspected window of R, the matching tolerance and bound row in D, the empty or initial monotone and boundedness ledger entries in M and B, and the provenance rows carried by H, C, P, N.

For the step case, append one visible rational observation and its tolerance row. Componentwise hsame in H transports the source, schedule, dyadic, monotone, boundedness, and seal coordinates without changing their row types. The displayed Cont replay in C appends the new dyadic observation before replaying the corresponding monotone comparison in M and the corresponding lower and upper bound in B.

Thus each prefix consumer receives exactly the finite window and ledger data already displayed at that stage. Passing from one prefix to the next preserves the route of Definition 1406.4; the $\mathsf{Real}^{\uparrow}$ seal remains downstream of the visible rational, tolerance, monotone, boundedness, and provenance rows. Since no induction clause contains a coordinate for located supremum data, selected limits, countable choice, trichotomy, or ambient completeness, none is introduced before the seal. □

Theorem 1406.6 (SpeckerSequence finite-prefix Real-tail inspection)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *consumer that inspects the packet before the* $\mathsf{Real}^{\uparrow}$ *seal is read through finite windows only. More precisely, before any completion-facing refusal or located-supremum boundary is invoked, the consumer can access only the displayed* $\mathsf{StreamName}^{\uparrow}$ *prefix of* W*, the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound rows in* D*, the monotone and bounded ledger prefixes in* M *and* B*, and the structural rows* H, C, P, N*.*

The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *may be cited only after those finite-prefix rows have been displayed, and the read still contains no located supremum, selected limit, quotient stream equality, countable-choice selector, trichotomy branch, or ambient completeness witness.*

Proof. Project the carrier of Definition 1406.1. Its analytic coordinates before the seal are exactly the regular-source row R, the finite schedule W, the dyadic ledger D, and the monotone and boundedness ledgers M, B.

Apply the finite-stage induction theorem, Definition 1406.5. At the base prefix the consumer sees only the initial displayed rational window and its ledger rows; at each step it appends one finite observation, its tolerance row, and the matching monotone and boundedness entries.

Use the monotone bounded ledger route, Definition 1406.4, to place the $\mathsf{Real}^{\uparrow}$ seal after these finite-prefix rows. Then apply the RealUp seal non-escape theorem, Definition 1406.10, to keep completion-facing refusal and located-supremum boundaries outside the displayed packet. □

Theorem 1406.7 (SpeckerSequence non-located supremum boundary)**.** *An accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet may certify a monotone bounded computable rational stream, but it does not certify a located supremum. Any downstream route that treats the packet as a located* $\mathsf{Real}^{\uparrow}$ *datum must supply an external location witness not contained in* $R, W, D, M, B, E, H, C, P, N$*.*

Proof. Use Definition 1406.2 to identify the packet inventory. The available analytic rows are the regular rational source, stream schedule, dyadic observations, monotone ledger, boundedness ledger, and finite seal boundary.

Use Definition 1406.4 for the seal route. It records that E is reached through finite windows and ledgers, not through a located-supremum construction.

A located supremum requires a location witness comparing rational cuts around the supremum. No such witness is a row of the carrier, and the structural rows merely transport, replay, source, and name the displayed packet. Hence the packet sets the non-located boundary rather than closing located completeness. □

Theorem 1406.8 (SpeckerSequence located supremum refusal)**.** *For any accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet* $S = (R, W, D, M, B, E, H, C, P, N)$*, finite-stage induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule can expose only the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the dyadic tolerance row* D*, the monotone ledger* M*, the boundedness ledger* B*, the finite* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural rows* H, C, P, N*. Therefore a located-supremum witness cannot be reconstructed from the packet and must be supplied as an external row.*

Proof. Apply Definition 1406.5. The induction route over W reads a finite prefix of the source stream, the matching dyadic tolerance data, the corresponding monotone and boundedness ledger prefixes, and the transported provenance rows before the seal is reached.

Apply Definition 1406.4. It fixes the seal-facing order

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E$$

and leaves H, C, P, N as structural transport, replay, provenance, and naming rows rather than analytic witnesses.

Finally apply Definition 1406.10. The finite $\mathsf{Real}^{\uparrow}$ seal exports only the displayed finite-window read. Since none of the displayed rows is a located comparison witness for rational cuts around the supremum, any downstream located-supremum claim must provide that witness outside the $\mathsf{SpeckerSequence}^{\uparrow}$ packet. □

Theorem 1406.9 (Specker sequence located supremum consumer boundary)**.** *A* $\mathsf{LocatedSupremum}^{\uparrow}$ *or* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer reading an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet receives only finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source windows, the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, monotone and boundedness ledgers, the* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows* H, C, P, N*. The located-supremum witness remains an external row.*

Proof. Apply Definition 1406.5. It confines every inspected prefix to the displayed regular rational window, stream schedule, dyadic tolerance row, monotone prefix, boundedness prefix, and transported provenance before the seal is reached.

Apply Definition 1406.4. The monotone and boundedness rows are read before any $\mathsf{Real}^{\uparrow}$ seal consumer can inspect E, so the route remains finite-window data rather than a located supremum construction.

Apply Definition 1406.8. Since the packet contains no located comparison witness for rational cuts around the supremum, a $\mathsf{LocatedSupremum}^{\uparrow}$ or $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ consumer must bring that witness as an external row. □

Theorem 1406.10 (SpeckerSequence RealUp seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet is a finite seal read through the displayed source, schedule, dyadic, monotone, and boundedness rows. The read exports no Real completeness principle, located supremum, Dedekind cut, countable-choice selector, trichotomy branch, bridge checking datum, or selected limiting real.*

Proof. Factor the read through the carrier exposure theorem. The $\mathsf{Real}^{\uparrow}$ seal row E is available only as one displayed coordinate of the finite packet.

Apply classifier transport to keep every replay on the same componentwise rows. Transport cannot change the seal into a located supremum or a completed real object because it preserves the carrier shape.

Apply the monotone bounded ledger theorem and the non-located supremum boundary. Together they show that a seal-facing consumer receives only finite regular windows, dyadic observations, monotone evidence, boundedness evidence, and structural certificate rows.

The excluded principles are absent from every coordinate. Therefore the $\mathsf{Real}^{\uparrow}$ seal cannot escape to Real completeness, located supremum data, Dedekind cuts, countable choice, trichotomy, bridge checking, or selected limits. □

Definition 1406.11 (Specker sequence obligation surface)**.** *The* $\mathsf{SpeckerSequence}^{\uparrow}$ *obligation surface is the finite subpacket*

$$O = (R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, E, H, C, P, N)$$

read from an accepted carrier $S = (R, W, D, M, B, E, H, C, P, N)$*. The row* $R_{\leq W}$ *is the finite rational stream window selected by the* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*;* $D_{\leq W}$ *is the matching dyadic tolerance and bound ledger;* $M_{\leq W}$ *and* $B_{\leq W}$ *are the monotone and bounded ledger prefixes for the same displayed stages; and* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after those prefixes.*

The structural rows are unchanged: H *gives componentwise* hsame *transport,* C *gives displayed* Cont *replay,* P *gives package provenance, and* N *gives the local* $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$ *row. The surface contains no located supremum row, countable-choice selector, trichotomy branch, Dedekind-cut representation, host-selected limit, or ambient completeness field.*

Theorem 1406.12 (Specker sequence regular-Cauchy obligation)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet discharges its regular-Cauchy obligation through finite-stage induction over the displayed schedule* W*. At each inspected stage the consumer receives only the current rational window* $R_{\leq W}$*, the dyadic tolerance row* $D_{\leq W}$*, the monotone prefix* $M_{\leq W}$*, the boundedness prefix* $B_{\leq W}$*, and the* BHist *provenance transported through* H, C, P, N *before the packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

Proof. Start from the obligation surface of Definition 1406.11. Its visible analytic data are the finite rational stream window, the dyadic tolerance and bound ledger, and the two finite ledger prefixes.

Use the finite-stage induction route of Definition 1406.5. The base prefix exposes the initial rational observation and its matching ledger rows. The step prefix appends one displayed observation, transports the rows through H, and replays the dyadic, monotone, and boundedness entries through C.

The regular-Cauchy obligation is therefore local to each displayed finite prefix. The seal row E is downstream of those rows by Definition 1406.10, so no located supremum, selected limit, countable choice, trichotomy, Dedekind cut, or ambient completeness coordinate is introduced. □

Theorem 1406.13 (Specker sequence no-choice boundary)**.** *The* $\mathsf{SpeckerSequence}^{\uparrow}$ *obligation surface is stable under componentwise* hsame *transport and displayed* Cont *replay without adding any selector that chooses a supremum, a limiting real, a trichotomy branch, or a countable family of witnesses. Any downstream proof that needs one of those objects must supply it outside the packet.*

Proof. Project the transported packet through Definition 1406.3. The classifier shape is preserved on the regular-source row, stream schedule, dyadic ledger, monotone ledger, boundedness ledger, finite seal row, provenance row, and local naming row.

Now restrict to the obligation surface. The rows $R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, E, H, C, P, N$ are exactly the finite rows of Definition 1406.11; transport and replay do not create a new coordinate.

The no-choice boundary follows from the missing coordinate. Since no row stores a supremum selector, limiting real selector, trichotomy decision, countable family choice, Dedekind cut, or completeness witness, no consumer can extract such data from the packet. □

Theorem 1406.14 (Specker sequence Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet factors through the obligation surface and then through the finite seal row* E*. The handoff route is*

$$R_{\leq W}, D_{\leq W}, M_{\leq W}, B_{\leq W}, H, C, P, N \longmapsto E,$$

so the exported real-facing information is a sealed finite-window read rather than a located supremum or a host-selected limit.

Proof. Use Definition 1406.4 to order the route. The source and schedule determine the inspected rational windows, the dyadic ledger supplies tolerance and bound rows, and the monotone and boundedness ledgers are read before E.

Use Definition 1406.11 to package exactly those finite rows as the public obligation surface. The structural rows H, C, P, N preserve and replay the route but do not extend it.

Finally apply Definition 1406.10. The $\mathsf{Real}^{\uparrow}$ handoff remains a finite seal read, so it refuses located supremum data, countable choice, trichotomy, Dedekind cuts, host-selected limits, and ambient completeness. □

Theorem 1406.15 (Specker sequence obligation route)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *carrier routes its monotone bounded ledger, nonlocated supremum refusal,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal data through the displayed finite rows*

$$R, W, D, M, B, E, H, C, P, N.$$

The route uses the BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row, and it contains no coordinate for countable choice, selected supremum data, trichotomy, Dedekind cuts, or a host-selected limiting real.*

Proof. Start with the carrier exposure theorem. It identifies the only analytic rows as the regular rational source, finite schedule, dyadic ledger, monotone ledger, boundedness ledger, and finite real seal.

Use the monotone bounded ledger theorem and finite-stage induction. Together they force every inspected prefix through R, W, D, M, B before the seal row E can be consumed.

Use the non-located supremum boundary and RealUp seal nonescape. Since H, C, P, N only transport, replay, source, and name the same finite route, the packet cannot export a selected supremum, countable choice, trichotomy, Dedekind cut, or ambient completeness coordinate. □

Theorem 1406.16 (Specker sequence regular readback scope)**.** *The regular readback exported by an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet is exactly the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window read selected by* W *and certified through* D, M, B, H, C, P, N *before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Every downstream consumer must read the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; no consumer can extract a countable-choice selector, a located supremum coordinate, a trichotomy branch, or a selected limiting real from the readback.*

Proof. Apply Definition 1406.11. Its visible readback rows are the scheduled regular rational window, the dyadic tolerance and bound ledger, the monotone prefix, the boundedness prefix, and the finite seal.

Apply Definition 1406.12. The regular-Cauchy obligation is local to the displayed finite prefixes, so the readback does not depend on an infinite selector or a located supremum coordinate.

Finally apply Definition 1406.13. Transport and replay preserve the same finite obligation surface and introduce no countable-choice, trichotomy, Dedekind-cut, or selected-limit row. □

Theorem 1406.17 (SpeckerSequence scoped kernel route)**.** *Every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet route through*

$$R, W, D, M, B, E, H, C, P, N$$

is scoped by the finite kernel rows BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The no-choice and non-located-supremum exclusions follow from the absence of rows for countable-choice selectors, located supremum witnesses, trichotomy branches, Dedekind cuts, and host-selected limiting reals, not from external prose.*

Proof. Start with carrier exposure. The accepted packet exposes only the regular rational source row R, finite window row W, dyadic ledger D, monotone ledger M, boundedness ledger B, finite Real seal E, and structural rows H, C, P, N.

Use finite-stage induction and the monotone bounded ledger. Every inspected prefix is read through R, W, D, M, B before any Real-facing seal is available, so the route remains inside the displayed BHist packet and its componentwise hsame and Cont rows.

Apply the no-choice boundary. Since no row records a countable selector, a trichotomy branch, a Dedekind cut, or a chosen limiting real, none of those data can be recovered by replaying H, C, P, N.

Finally use the $\mathsf{Real}^{\uparrow}$ seal nonescape theorem. The seal row E exports only the finite-window Real-facing readback, and therefore the scoped route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ is exhaustive for the packet. □

Theorem 1406.18 (Specker sequence monotone bounded nonconvergence route)**.** *Let* $S = (R, W, D, M, B, E, H, C$ *be an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet. If a consumer reads the packet through the monotone bounded ledger and the regular readback scope, then the consumer receives bounded monotone evidence only as finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows inside the*

displayed BHist *carrier. This route may be used as monotone bounded finite-window data, but it does not yield a located supremum, a selected limit, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier exposure theorem and the monotone bounded ledger. The accepted packet exposes the regular rational source R, the finite schedule W, the dyadic tolerance and bound ledger D, the monotone and boundedness ledgers M, B, the finite seal E, and the structural rows H, C, P, N. The ledger order forces every seal-facing read through

$$R, W \longmapsto D \longmapsto (M, B) \longmapsto E.$$

Apply finite-stage induction along W. Each displayed prefix carries only the currently inspected $\mathsf{RegSeqRat}^{\uparrow}$ window, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ row, the monotone and boundedness prefixes, and the transported BHist provenance. The regular readback scope keeps this information in the $\mathsf{StreamName}^{\uparrow}$ schedule and the finite seal route; it never opens an infinite selector.

Now apply the non-located supremum boundary and the scoped kernel route. The packet contains no row for rational-cut location around a supremum, no branch selecting trichotomy, no countable family of witnesses, and no ambient completeness coordinate. The sibling boundaries of chapter 2255, chapter 1405, chapter 1665, and chapter 1666 identify exactly the external consumers that would have to provide those missing rows.

Thus bounded monotone Specker windows are usable finite data inside the BEDC kernel, while the non-located supremum boundary remains intact. The route certifies finite monotone readback, not convergence to a selected $\mathsf{Real}^{\uparrow}$ limit. □

Theorem 1406.19 (Specker sequence completion-refusal induction)**.** *Any Completion-facing consumer reading an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet can induct only over the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*. At each stage the consumer receives the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *source window, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the monotone and boundedness ledger prefixes, the finite* $\mathsf{Real}^{\uparrow}$ *seal route, and the structural rows* H, C, P, N*; it receives no located supremum, selected limit, trichotomy branch, countable-choice selector, quotient stream equality, or ambient completion witness.*

Proof. Apply finite-stage induction, Definition 1406.5. The induction variable is the displayed schedule W, so each step exposes only the current regular rational window, dyadic tolerance row, monotone prefix, boundedness prefix, and transported provenance.

Use the regular readback scope, Definition 1406.16. It keeps the readback inside the finite $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before the packet reaches the $\mathsf{Real}^{\uparrow}$ seal.

Apply the no-choice boundary, Definition 1406.13. Transport and replay through H, C, P, N do not add a selector for a supremum, limiting real, trichotomy branch, countable family of witnesses, or quotient stream equality.

Finally use the monotone bounded nonconvergence route, Definition 1406.18. Completion-facing consumers may use the bounded monotone finite-window data, but the packet still refuses ambient completion and any selected limit beyond the displayed finite seal. □

Theorem 1406.20 (Specker sequence Cauchy criterion non-located boundary)**.** *An accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet read through an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *packet still exposes only finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, monotone and bounded ledgers, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The combined read cannot yield a located supremum, selected limit, quotient stream equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the completion-refusal induction of Definition 1406.19. It confines every Completion-facing Specker read to finite $\mathsf{StreamName}^{\uparrow}$ schedule induction over the displayed regular rational windows, dyadic tolerances, monotone and boundedness prefixes, Real seal, and structural rows.

Apply the monotone bounded nonconvergence route of Definition 1406.18. It allows the monotone bounded evidence to be consumed as finite data, but it keeps located supremum, selected limit, trichotomy, countable choice, and ambient completeness outside the packet.

Finally read the same surface through the Cauchy criterion boundary of Definition 1509.15. The criterion contributes only finite windows, threshold and tolerance rows, regular readback, Real seal, and structural rows shared by the packets. It adds no quotient stream equality, selected host limit, located supremum coordinate, or Real completeness witness. □

Theorem 1406.21 (Specker sequence regular-window induction)**.** *Every completion-facing consumer inspecting an accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet before a located-supremum row must proceed by finite induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *window. At each step the consumer exposes only the current* $\mathsf{RegSeqRat}^{\uparrow}$ *source prefix, the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the monotone ledger, the bounded ledger, and the structural* BHist *rows preserved by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

Proof. Start with the finite-stage induction theorem, Definition 1406.5. Its induction variable is the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, so every inspected stage is a finite prefix read rather than a selected infinite stream.

Apply the regular-Cauchy obligation of Definition 1406.12. The obligation exposes only the current $\mathsf{RegSeqRat}^{\uparrow}$ source prefix, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the monotone and boundedness ledger prefixes before any completion-facing conclusion is visible.

Finally apply the Cauchy-criterion non-located boundary of Definition 1406.20. Transport, replay, provenance, and local naming preserve the same structural BHist rows, and no located-supremum row, selected limit, quotient stream equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness is added. □

Theorem 1406.22 (Specker sequence finite-window refusal induction)**.** *For every accepted* $\mathsf{SpeckerSequence}^{\uparrow}$ *packet, finite-stage induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule preserves exactly the* $\mathsf{RegSeqRat}^{\uparrow}$ *source prefix,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, monotone ledger, bounded ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. No induction step can introduce a located-supremum row, selected-limit row, quotient stream equality, countable-choice selector, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Induct on the inspected finite window of the displayed $\mathsf{StreamName}^{\uparrow}$ schedule. In the empty window, the carrier exposes only its structural BHist rows and the already named refusal boundary, so no located-supremum or selected-limit coordinate is present.

For the step, extend the inspected window by one displayed source observation. Apply Definition 1406.5 to keep the new observation inside the same finite schedule, and apply Definition 1406.21 to preserve the $\mathsf{RegSeqRat}^{\uparrow}$ prefix, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, monotone ledger, bounded ledger, and structural transport, replay, provenance, and naming rows.

Finally use Definition 1406.20. The terminal $\mathsf{Real}^{\uparrow}$ seal remains a finite boundary read, and the missing located-supremum, selected-limit, quotient, choice, trichotomy, and ambient-completeness rows have no carrier coordinate from which an induction step could create them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerSequence}^{\uparrow}}$**.**

Constructive story: A SpeckerSequence$^{\uparrow}$ packet is generated from a RegSeqRat$^{\uparrow}$ computable rational source, a StreamName$^{\uparrow}$ finite-window schedule, DyadicRatCore$^{\uparrow}$ tolerance and bound rows, a monotone ledger, a boundedness ledger, a finite Real$^{\uparrow}$ seal boundary, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{SpeckerSequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1406.1, Definition 1406.2, Definition 1406.3, Definition 1406.4, Definition 1406.5, Definition 1406.7, Definition 1406.8, Definition 1406.9, Definition 1406.10, Definition 1406.11, Definition 1406.12, Definition 1406.13, Definition 1406.14, Definition 1406.15, Definition 1406.16, Definition 1406.17, Definition 1406.18, Definition 1406.19, and Definition 1406.20 declare the finite Specker-sequence carrier, transport, monotone bounded ledger, finite-stage induction route, non-located supremum boundary, located-supremum refusal, public located-supremum consumer boundary, public RealUp seal non-escape surface, obligation surface, regular-Cauchy obligation, no-choice boundary, public Real handoff, obligation route, regular readback scope, scoped kernel route, monotone bounded nonconvergence route, Completion-refusal induction route, and Cauchy-criterion non-located boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{SpeckerSequence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SpeckerSequenceUp.SpeckerSequenceCarrier\_exposure`

Bridge status: none

Not claimed: This chapter does not close Real completeness, located supremum construction, Dedekind cuts, countable choice, trichotomy, bridge checking, or selected limiting reals.

Upgrade path: Further verification work is bridge checking for the existing finite-stage, no-choice, Real-handoff, and Real-seal theorem surface.

Chapter 1407

A Concrete Naming Certificate for $\mathsf{BisectionRootIsolation}^{\uparrow}$

Following Bishop and Bridges, constructive root isolation is read here as a finite sign-change bisection packet over located intervals rather than as a host root-existence theorem. The $\mathsf{BisectionRootIsolation}^{\uparrow}$ surface binds endpoint windows, sign rows, branch choices, regular rational readback, and real-name seal rows into one local NameCert packet. It consumes the located interval and dyadic bisection certificates of chapter 394 and chapter 361, with tolerance and readback rows supplied by chapter 291, chapter 126, chapter 301, and chapter 38.

Definition 1407.1 (Bisection root-isolation carrier)**.** *A* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (J, B, F, E, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, exposing the lower and upper endpoint windows, rational cell ledger, optional dyadic refinements,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal-facing rows;*

2. *B is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger whose retained interval rows refine the endpoint packet of J by finite midpoint comparisons;*

3. *F is the function-inspection ledger, consisting only of finite rational and dyadic reads used to evaluate the displayed sign rows at the retained endpoints and midpoints;*

4. *E records the sign rows at the retained lower and upper endpoints, together with the midpoint sign rows used to choose each branch;*

5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window ledger exposed to downstream consumers after the retained interval is selected;*

6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained endpoint windows and the dyadic tolerance rows supplied through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*

7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal surface for the retained lower and upper endpoint names and for any real-facing consumer of the isolating window;*

8. *H stores the componentwise* hsame *transports for the located interval packet, bisection ledger, sign rows, stream windows, regular rational readbacks, real seal rows, and local certificate row;*

9. *C stores the displayed* Cont *replay routes for endpoint inspection, midpoint inspection, branch selection, retained-window readback, regular-sequence handoff, and real-name seal consumption;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite sign-change bisection packet. It has no coordinate for a host root, a general intermediate value theorem, trichotomy on reals, a completed intersection of nested intervals, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationUp`
Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationTasteGate single carrier alignment`

Theorem 1407.2 (Bisection root-isolation NameCert obligations)**.** *For every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier* $Q = (J, B, F, E, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the displayed finite packet of Definition 1407.1;*
2. *endpoint-window admission through the* $\mathsf{LocatedInterval}^{\uparrow}$ *rows of J and the retained interval rows of B;*
3. *sign-ledger exactness, requiring each branch choice in B to be justified by one displayed sign row in E read through F;*
4. *retained-window handoff through W, R, and S;*
5. *componentwise transport and replay through H and C;*
6. *package provenance and local naming through P and N.*

No further analytic obligation is part of the packet.

Proof. Project the carrier of Definition 1407.1. The endpoint packet J supplies the located lower and upper endpoint windows, while the bisection ledger B supplies the finite retained-interval chain.

The sign-ledger obligation is local to F and E. Each midpoint inspection reads a finite rational or dyadic approximation row, records the displayed sign row, and then selects the corresponding branch row in B.

The retained-window handoff is the route already available from chapter 361 after the endpoints are read through chapter 394. The tolerance rows come from chapter 291; the window, readback, and seal rows are then routed through chapter 126, chapter 301, and chapter 38.

Finally, H, C, P, and N account for transport, replay, provenance, and local naming. Since the carrier stores only these finite rows, it cannot export host root existence, an intermediate value theorem, real trichotomy, a completed interval intersection, or completeness for $\mathsf{Real}^{\uparrow}$. □

Theorem 1407.3 (Bisection root-isolation RegSeqRat window handoff)**.** *For every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier and every unary precision request n, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposed by the isolating window factors through the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix that reaches the retained window at tolerance 2^{-n}, the displayed sign rows at retained endpoints, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, and the replay rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRegSeqRatWindowHandoff`

Proof. Begin with the unary request n. The dyadic tolerance row 2^{-n} is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance, and the branch prefix in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger selects the retained interval window at that tolerance.

The located endpoint packet then supplies the lower and upper endpoint windows for that retained interval. The sign ledger contributes only the displayed endpoint signs and midpoint signs already used to select the branch prefix; it does not introduce a root object.

Route the retained endpoint windows through the $\mathsf{StreamName}^{\uparrow}$ rows and then through the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The replay remains inside H, C, P, and N, so every rational read exposed to the regular-sequence consumer is one of the retained finite endpoint windows. □

Theorem 1407.4 (Bisection root-isolation Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier is reached only after the retained* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint windows, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the displayed endpoint sign rows, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the rows* H, C, P, N *have been read. The seal exposes the finite isolating window and exports no host root existence, general intermediate value theorem, real trichotomy, completed interval intersection, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRealSealNonescape`

Proof. Read the real-facing coordinate S in Definition 1407.1. It is not an independent real object; it is stored with the endpoint packet, branch ledger, sign rows, stream windows, regular readback, transport rows, replay routes, provenance, and local certificate.

Apply Definition 1407.3. The retained endpoint windows first pass through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces at the requested dyadic tolerance.

Only after that route is fixed can the $\mathsf{Real}^{\uparrow}$ seal be read. The seal therefore records the same finite isolating window and no additional analytic witness. Because no coordinate of Q stores a host root, an intermediate value theorem, trichotomy data, a completed intersection, or completeness for $\mathsf{Real}^{\uparrow}$, the real-facing consumer cannot escape to any of those objects. □

Theorem 1407.5 (Bisection root-isolation finite sign-change certificate)**.** *For every accepted* $\mathsf{BisectionRootIsolation}$ *carrier* $Q = (J, B, F, E, W, R, S, H, C, P, N)$*, every sign-change assertion visible to the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsola}}$ *is certified by finite endpoint and midpoint rows in* F*,* E*, and* B*. The certificate contains no host intermediate value theorem, no real trichotomy oracle, no root object, and no completed intersection of retained intervals.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationFiniteSignChangeCertificate`

Proof. Project the carrier rows of Definition 1407.1. The only function-inspection data are the finite rational and dyadic reads in F, and the only sign data are the endpoint and midpoint rows displayed in E.

Apply Definition 1407.2. Each branch choice in B must be justified by one of those displayed rows, so the sign-change surface is exhausted by the finite ledger already stored in the packet.

The excluded analytic objects are absent from the carrier coordinates. Thus the certificate can replay the accepted sign rows and retained branch choices, but it cannot export a host root, a general intermediate value theorem, trichotomy on reals, or a completed interval intersection. □

Theorem 1407.6 (Bisection root-isolation retained-window refinement)**.** *For every accepted* $\mathsf{BisectionRootIsolation}$ *carrier and every unary precision request* n*, the retained isolating window at tolerance* 2^{-n} *factors through the* $\mathsf{DyadicBisection}^{\uparrow}$ *retained branch prefix, the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exposed by the carrier.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationRetainedWindowRefinement`

Proof. Begin with the unary request n. The tolerance row is the $\mathsf{DyadicRatCore}^{\uparrow}$ row already named in Definition 1407.1, and the branch prefix is the retained prefix stored in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger B.

The endpoint rows come from the located interval packet J. Once the retained prefix is fixed, the selected lower and upper windows pass through the $\mathsf{StreamName}^{\uparrow}$ rows W.

Apply Definition 1407.3. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads back exactly those retained endpoint windows at the requested tolerance, with transport and replay confined to H, C, P, and N. □

Theorem 1407.7 (Bisection root-isolation downstream boundary)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier receives only the retained isolating window determined by J, B, W, R, S, H, C, P, and N. Such a consumer cannot extract host root existence, a completed interval intersection, a general intermediate value theorem, real trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Lean4 ✓ `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationDownstreamBoundary`

Proof. Read the downstream-facing seal row through Definition 1407.4. The seal is reached only after the retained located endpoints, branch ledger, stream windows, regular readback, and replay rows have already been fixed.

The retained-window route is the one described in Definition 1407.6. It delivers a finite window at the requested tolerance, not an independent analytic object.

Since Definition 1407.5 confines the sign-change evidence to finite endpoint and midpoint rows, no downstream consumer can recover the excluded host principles from the packet. □

Theorem 1407.8 (Bisection root-isolation public export)**.** *Every public consumer of an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier must read the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the structural* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows in that order before using the isolation certificate. The exported object is the finite retained isolating window with displayed sign-change evidence; it is not a selected root, a real-trichotomy decision, quotient equality of real names, a completed nested-interval point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier $Q = (J, B, F, E, W, R, S, H, C, P, N)$ of Definition 1407.1. The carrier already orders the public data: J supplies the located endpoint packet, B supplies the retained dyadic branch ledger, and the dyadic tolerance rows are the arithmetic support named in the same packet.

Apply Definition 1407.2. It forces the endpoint-window admission and sign-ledger exactness to be read before retained-window handoff, so the public consumer cannot inspect W, R, S until the located interval, bisection ledger, tolerance support, and displayed sign rows have been exposed.

Next apply Definition 1407.3 and Definition 1407.4. These statements route the retained endpoint windows through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal without producing an independent real object.

Finally use Definition 1407.7. The structural rows H, C, P, N transport, replay, package, and name exactly that finite window. Since the carrier has no coordinate for a selected root, real trichotomy, quotient real equality, a completed nested-interval point, or ambient completeness, the public export is exhausted by the ordered rows listed in the statement. □

Theorem 1407.9 (Bisection root-isolation sign-ledger determinacy)**.** *Let $Q = (J, B, F, E, W, R, S, H, C, P, N)$ be an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier. For every retained branch row in B, the corresponding finite inspection row in F and displayed sign row in E determine the branch visible to the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *packet. The visible branch decision is therefore a finite* BHist *route through F, E, B, H, C, P, N, not a host trichotomy comparison or an external root-location oracle.*

Proof. Project the carrier rows F, E, B from Definition 1407.1. The function-inspection ledger F is the only source of endpoint and midpoint reads, and E is the only sign ledger stored for those reads.

Apply the NameCert obligations of Definition 1407.2. They require each branch choice in B to be justified by one displayed sign row in E read through F, with transport and replay confined to H, C, P, N.

Thus two accepted replays with the same inspected row and the same displayed sign row expose the same retained branch row to the packet. Since no coordinate stores real trichotomy, a host root, or an external oracle, the branch visible to $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ is determined exactly by the finite sign ledger. □

Theorem 1407.10 (Bisection root-isolation branch monotonicity)**.** *For an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier, extending the dyadic bisection depth refines the retained interval window and never enlarges the* $\mathsf{Real}^{\uparrow}$*-seal-facing interval. The refinement route factors through the retained* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix, the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the displayed sign ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows.*

Proof. Start with two unary depth requests where the second extends the first. The retained branch prefix lives in the $\mathsf{DyadicBisection}^{\uparrow}$ ledger B, and Definition 1407.6 routes each requested depth through the located endpoint rows, dyadic tolerance row, stream windows, and regular readback.

The extension step can only append midpoint inspection and sign rows already admitted by Definition 1407.9. It therefore selects a subwindow of the earlier retained window rather than a new ambient interval.

The downstream seal is constrained by Definition 1407.4. Since every real-facing read passes through the retained window already chosen by the branch ledger, increasing depth can refine the exposed interval but cannot enlarge it or introduce host root existence, completed interval intersection, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1407.11 (Bisection root-isolation depth coverage)**.** *Every unary precision request made to an accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *carrier factors through some finite branch prefix in B, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window rows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before it reaches a* $\mathsf{Real}^{\uparrow}$ *seal. The packet therefore covers precision requests by finite bisection depth rather than by a completed nested-interval object.*

Proof. Begin with a unary precision request n. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance 2^{-n}, and Definition 1407.6 selects a retained branch prefix whose window is read at that tolerance.

The endpoint windows then pass through the $\mathsf{StreamName}^{\uparrow}$ rows. Apply Definition 1407.3 to read the same retained windows through $\mathsf{RegSeqRat}^{\uparrow}$ at the requested precision.

Finally route the readback to the real-facing boundary using Definition 1407.4. The request is exhausted by the finite prefix, tolerance, window, and readback rows, so no completed branch, host root, general intermediate value theorem, trichotomy oracle, or ambient completeness principle is exposed. □

Theorem 1407.12 (BisectionRootIsolation public isolation certificate)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *isolation certificate exposes the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, the displayed sign-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal before any public isolation read is available. The public read is therefore the finite retained isolating window with its displayed sign-change ledger, not a host root, a general intermediate value theorem, real trichotomy, a completed interval intersection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the retained-window refinement of Definition 1407.6. It routes each requested precision through the located endpoint packet, the retained $\mathsf{DyadicBisection}^{\uparrow}$ branch prefix, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the branch ledger and sign stability supplied by Definition 1407.9 together with Definition 1407.5. These theorems keep every branch choice inside the finite inspection row, displayed sign row, and retained branch row; no host trichotomy or external root-location oracle is added.

Finally apply the public export and Real-seal boundary of Definition 1407.8 and Definition 1407.4. The public consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only after the endpoint, branch, sign, tolerance, stream-window, and regular-readback rows have been displayed, so the exported isolation certificate is exactly the finite L10 window named in the statement. □

Theorem 1407.13 (Bisection root-isolation interval nesting)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *packet exposes a nested chain of retained windows before its public isolation certificate is consumed. For each retained branch prefix, the lower and upper endpoints of the selected window pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows; a later retained prefix selects a subwindow of the earlier one and cannot replace that chain by a host root, a completed interval intersection, real trichotomy, or ambient completeness.*

Proof. Induct over the retained-window refinement rows of the $\mathsf{DyadicBisection}^{\uparrow}$ ledger. The initial row is the located endpoint packet J of Definition 1407.1; its endpoints are already routed through the displayed dyadic tolerance, stream-window, regular-readback, and real-seal rows.

For the refinement step, use Definition 1407.9. The next retained branch is determined by the finite inspection row and displayed sign row, so the branch ledger can only select one of the two displayed subwindows of the previous retained window.

Apply Definition 1407.6 and Definition 1407.12 to the selected prefix. They route its endpoints through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before public consumption. The nonescape boundary of Definition 1407.4 excludes the host analytic objects, so the public certificate sees a nested retained-window chain rather than an external root-selection principle. □

Theorem 1407.14 (Bisection root-isolation bridge schema)**.** *Every accepted* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *packet exports a bridge-readable schema consisting of the located interval source, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bisection ledger, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the finite sign ledger, retained-window monotonicity, and the local* $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ *rows. The schema is a finite isolation route and does not assert a general intermediate value theorem, host root existence, completed interval intersection, real trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public isolation certificate of Definition 1407.12. It exposes the located endpoint rows, the retained dyadic branch ledger, the displayed sign-window rows, the dyadic tolerance rows, stream windows, regular readback, and the real seal before any public isolation read.

Use interval nesting from Definition 1407.13. The retained prefixes form the bridge-visible window chain, and each endpoint read is still routed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Add branch monotonicity and depth coverage from Definition 1407.10 and Definition 1407.11. These rows identify the retained-window monotonicity and the finite prefix selected for each unary precision request.

Finally apply the Real seal nonescape boundary of Definition 1407.4. The bridge-readable schema is therefore exhausted by the displayed carrier rows and the local $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$ route, with no host analytic principle available beyond the finite isolation packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}$.

Constructive story: A $\mathsf{BisectionRootIsolation}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint packet, a $\mathsf{DyadicBisection}^{\uparrow}$ branch ledger, finite sign rows at retained endpoints and midpoints, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal rows, displayed hsame and Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1407.1, Definition 1407.2, Definition 1407.3, Definition 1407.4, Definition 1407.5, Definition 1407.6, Definition 1407.7, and Definition 1407.12 bind the carrier, NameCert obligations, RegSeqRat window handoff, Real seal nonescape, finite sign-change certificate, retained-window refinement, downstream boundary, public isolation certificate, and BEDC rows BHist, hsame, Cont, Pkg, NameCert, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to a finite sign-change bisection certificate over located intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionRootIsolation}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BisectionRootIsolationUp.BisectionRootIsolationDownstreamBoundary`

Bridge status: none

Not claimed: This chapter does not close host root existence, a general intermediate value theorem, real trichotomy, completed interval intersection, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The remaining work is Lean verification of the displayed carrier, sign-ledger exactness, retained-window handoff, and Real-seal nonescape, not paper obligation closure.

Chapter 1408

A Concrete Naming Certificate for $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet records the finite selector boundary that can be read before any classical Bolzano–Weierstrass conclusion is asserted. The route is anchored by the monotone selector surface of chapter 1405, the bounded monotone Cauchy witness of chapter 535, the bounded monotone convergence seal of chapter 918, the stream-window interface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name boundary of chapter 38. It packages a displayed finite handoff from bounded sequence windows to monotone selected windows, not sequential compactness, a host real completeness theorem, or a choice-selected subsequence.

Definition 1408.1 (Bolzano–Weierstrass selector carrier)**.** *A* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, M, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded source-sequence window ledger; it records only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows currently inspected and the displayed boundedness rows attached to those windows.*
2. *M is the monotone selector ledger read through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*; each displayed entry records a selector position, its source window, and the comparison row that makes the selected window monotone inside the finite packet.*
3. *Q is the cofinal evidence row for the inspected selector thresholds, limited to thresholds that occur in the packet.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window family obtained from B, M, Q.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read from the selected windows before any regular rational or real-facing handoff is available.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the selected dyadic windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only through the bounded monotone witness and bounded monotone convergence seal routes after B, M, Q, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for bounded windows, selector entries, cofinal evidence, selected windows, dyadic rows, regular handoff, Real seal, continuation rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *routes from B, M, Q to W, from W to D, from D to R, and from the bounded monotone witness and convergence seal packets to E.*

10. *P is the* Pkg *provenance over* BolzanoWeierstrassSelector$^{\uparrow}$, MonotoneSubsequence$^{\uparrow}$, BoundedMonotoneCauchyWit BoundedMonotoneConvergenceSeal$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BolzanoWeierstrassSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for full Bolzano–Weierstrass, sequential compactness, a choice-selected subsequence, quotient-stream equality, host real equality, a least cluster selector, or Real$^{\uparrow}$ *completeness.*

Theorem 1408.2 (Bolzano–Weierstrass selector subsequence handoff)**.** *Every accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet hands bounded source windows to selected monotone windows only through the route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R.$$

The first part of the route factors through MonotoneSubsequence$^{\uparrow}$, *and the regular-facing handoff is read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *before any* Real$^{\uparrow}$ *seal row is consumed.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorSubsequenceHandoff`

Proof. Project the carrier of Definition 1408.1. The bounded source row B, selector ledger M, inspected-threshold cofinality row Q, selected windows W, dyadic ledger D, regular handoff R, and structural rows H, C, P, N are the only rows visible before the seal coordinate.

Use the monotone selector surface of chapter 1405. It identifies the selector ledger and cofinal evidence as finite displayed rows, so the selected windows are obtained by packet replay rather than by countable choice or by an ambient subsequence object.

Next read the selected windows through the terminal finite route. The StreamName$^{\uparrow}$ windows provide W, the DyadicRatCore$^{\uparrow}$ ledger provides D, and the RegSeqRat$^{\uparrow}$ handoff provides R.

The transport, continuation, package, and local NameCert rows only preserve this displayed route. Since no other selector coordinate is present, every accepted subsequence handoff factors through $B, M, Q, W, D, R, H, C, P, N$. □

Theorem 1408.3 (Bolzano–Weierstrass selector Real seal boundary)**.** *Every real-facing read of an accepted* BolzanoWeierstrassSelector$^{\uparrow}$ *packet reaches the seal row E only after the finite selector route*

$$B, M, Q, W, D, R, E$$

has passed through the bounded monotone Cauchy witness and bounded monotone convergence seal packets. The consumer receives a Real boundary for the selected regular handoff, not full Bolzano–Weierstrass, sequential compactness, a choice-selected subsequence, quotient-stream equality, host real equality, or Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRealSealBoundary`

Proof. Begin with Definition 1408.2. It exposes B, M, Q, W, D, R before a Real-facing coordinate can be consumed, and it makes the selected regular handoff the only path from bounded windows to the seal-facing surface.

Now pass through the bounded monotone witness and convergence seal chapters. The witness route of chapter 535 supplies the bounded monotone Cauchy surface, while the seal route of

chapter 918 exposes the Real handoff only after the finite stream, dyadic, regular, and seal rows are displayed.

Finally inspect the excluded data. The carrier has rows for bounded windows, monotone selector entries, cofinal evidence, selected windows, dyadic tolerance rows, a regular handoff, a Real seal, transports, continuations, provenance, and local naming, but it has no row for sequential compactness, arbitrary choice, quotient streams, host equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Thus the Real-facing read is a boundary seal for the displayed finite route and nothing beyond that route is exported. □

Theorem 1408.4 (Bolzano–Weierstrass selector bounded-monotone route)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real-facing row* E *is available only after the finite route*

$$\mathsf{MonotoneSubsequence}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow} \longmapsto \mathsf{Stream}$$

has displayed the rows B, M, Q, W, D, R, E*. The packet therefore records a bounded monotone induction path into the* $\mathsf{Real}^{\uparrow}$ *seal, not full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.TasteGate.BolzanoWeierstrassSelectorBoundedMonotoneRo`

Proof. Project the carrier of Definition 1408.1. The rows B, M, Q, W, D, R, E are the only Real-facing path before transport, continuation, provenance, and local naming are applied.

Apply Definition 1408.2. It exposes the finite selector route from bounded source windows through monotone selected windows and then through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Apply Definition 1408.3. The Real seal row is consumed only after the bounded monotone Cauchy witness and bounded monotone convergence seal packets have been displayed.

Combining these two chapter routes gives the stated induction path from $\mathsf{MonotoneSubsequence}^{\uparrow}$ through the bounded monotone witness, convergence seal, stream, dyadic, regular, and Real rows. The carrier has no coordinate for classical compactness, countable choice, quotient streams, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$, so no accepted read can export those objects. □

Theorem 1408.5 (Bolzano–Weierstrass selector window obligation)**.** *For every accepted* BolzanoWeierstrassSelecto *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, each finite selected-window read is admissible only when the bounded source window row* B*, monotone selector ledger* M*, inspected-threshold row* Q*, selected* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, dyadic tolerance row* D*, regular handoff row* R*, Real seal row* E*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *are all read as one displayed packet. The route factors through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorWindowObligation`

Proof. Project Definition 1408.1. The rows $B, M, Q, W, D, R, E, H, C, P, N$ are the entire accepted packet, so a selected-window consumer cannot detach W from the bounded source and monotone selector rows.

Apply Definition 1408.2. It supplies the finite selector route from B, M, Q to W, D, R through $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$.

Finally apply Definition 1408.4. The bounded monotone witness and convergence seal place E behind the same route, while H, C, P, N transport, replay, source, and name the packet. □

Theorem 1408.6 (Bolzano–Weierstrass selector finite-window induction)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any finite extension of the inspected selector-threshold ledger that preserves the bounded source window row* B *and the monotone selector ledger* M *extends the selected-window route only by replaying*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The induction exports no selected infinite subsequence, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or additional compactness principle.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteWindowInduction`

Proof. Induct over the displayed finite selector-threshold ledger. The base ledger is the selected-window route already bound by Definition 1408.5: the consumer reads $B, M, Q, W, D, R, E, H, C, P, N$ as one finite packet and receives no external subsequence object.

For the successor step, append one inspected threshold that preserves the same bounded source row B and monotone selector ledger M. The subsequence handoff of Definition 1408.2 admits only the corresponding selected $\mathsf{StreamName}^{\uparrow}$ window in W, with the dyadic and regular rows still downstream of that window.

Finally apply Definition 1408.4 and Definition 1408.5. They force the route through D, R, E after the selected window, while H, C, P, N merely transport, replay, source, and name the same finite packet. Thus every finite extension is a replay of the displayed route and not an export of compactness, quotient equality, host equality, or ambient completeness. □

Theorem 1408.7 (Bolzano–Weierstrass selector regular-Cauchy extraction route)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the finite selected-window induction together with the bounded-monotone route determines a regular-Cauchy extraction surface through*

$$W \longmapsto D \longmapsto R \longmapsto E.$$

Every extracted finite window is read through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The route exports no selected infinite subsequence, host cluster point, sequential compactness theorem, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyExtractionRout`

Proof. Start with the subsequence handoff of Definition 1408.2. It exposes only selected finite $\mathsf{StreamName}^{\uparrow}$ windows in W after the bounded source row B, monotone selector ledger M, and inspected-threshold row Q have been read.

Apply the finite-window induction of Definition 1408.6. Each finite extension replays the same route from B, M, Q to W and then to the dyadic, regular-rational, and Real-seal rows.

Use the bounded-monotone route of Definition 1408.4 together with the window obligation of Definition 1408.5. These force the downstream read through D, R, E before any cluster-facing consumer may inspect the packet.

Finally apply the Real seal boundary of Definition 1408.3. Since the carrier has no coordinate for an infinite selected subsequence, host cluster point, sequential compactness theorem, quotient stream equality, or ambient completeness row, the extraction surface is exactly the displayed regular-Cauchy route. □

Theorem 1408.8 (Bolzano–Weierstrass selector cluster route)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the cluster-facing read is exactly the finite route*

$$B, M, Q, W, D, R \longmapsto E$$

through $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. It records a bounded monotone selected-window route to a Real seal, not a classical cluster-point theorem or a choice-selected subsequence.*

Proof. Start from the window obligation in Definition 1408.5. It binds $B, M, Q, W, D, R, E, H, C, P, N$ into one finite packet and prevents a consumer from reading a cluster-facing seal from an uninspected window.

Use the bounded-monotone route of Definition 1408.4. The row E is available only after the monotone selector has passed through the bounded monotone Cauchy witness and bounded monotone convergence seal.

The structural rows add no compactness principle. They only transport and replay the same carrier through H, C, P, N, so the exported cluster-facing object is the displayed Real seal boundary for the finite route. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorClusterRoute`

Theorem 1408.9 (Bolzano–Weierstrass selector nonescape)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, no consumer can export full Bolzano–Weierstrass compactness, host sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet. The only exported Real-facing data are the rows obtained from* $B, M, Q, W, D, R, E, H, C, P, N$ *through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier. Its visible data are the bounded source windows, monotone selector ledger, inspected-threshold row, selected windows, dyadic tolerances, regular handoff, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1408.8. The cluster-facing read reaches E only through the finite bounded-monotone route and the stream, dyadic, regular, and Real rows named in the statement.

The excluded objects have no row in the carrier and no row in the cited route. Therefore an accepted consumer may replay the finite selector packet, but it cannot export compactness, choice, quotient equality, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorNonescape`

Theorem 1408.10 (Bolzano–Weierstrass selector monotone-subsequence obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the monotone-subsequence surface consists of exactly three finite obligations:*

1. *finite selector admission for the displayed monotone selector ledger* M *and inspected-threshold cofinality row* Q *over the bounded source windows* B*;*
2. *monotone selected-window transport from* B, M, Q *to the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W *through componentwise* hsame *transport and displayed* Cont *replay;*
3. *bounded-monotone Real-seal nonescape from* W, D, R *to* E *through* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The obligations do not claim full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Read the selector rows from Definition 1408.1. The packet admits only bounded source windows, the monotone selector ledger, inspected-threshold cofinality data, selected windows, dyadic tolerances, regular readback, the Real seal, and the structural rows.

Apply Definition 1408.2. It turns the first two obligations into the displayed route $B, M, Q \longmapsto W \longmapsto D \longmapsto R$, with the monotone selector surface supplied by $\mathsf{MonotoneSubsequence}^{\uparrow}$ and preserved by H, C, P, N.

Apply Definition 1408.4. It places the Real-facing row behind the bounded monotone witness and convergence seal packets, so the nonescape obligation is a finite seal route rather than a compactness theorem or a choice-selected subsequence. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorMonotoneSubsequenceObligati`

Theorem 1408.11 (Bolzano–Weierstrass selector window-choice-free carrier)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the selected-window carrier is choice-free: each* $\mathsf{StreamName}^{\uparrow}$ *window in* W *is read from a displayed selector entry in* M *and an inspected bounded source window in* B*, then transported by the componentwise* hsame *row* H *and replayed by the displayed* Cont *row* C*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *are downstream rows of the same finite packet, not witnesses to a countable-choice selector, sequential compactness, or an ambient Bolzano–Weierstrass theorem.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorWindowChoiceFreeCarrier`

Proof. Induct over the finite selector-window portion of the displayed BHist packet. The empty window reads no selector entry and therefore exposes no hidden subsequence object.

At a successor step, project the next entry of M together with its source window in B and inspected threshold in Q. The entry appends the corresponding $\mathsf{StreamName}^{\uparrow}$ row to W through H and C, so the carrier step is a finite replay rather than a choice of a new subsequence.

After the selected window is displayed, the dyadic and regular rows D, R and the $\mathsf{Real}^{\uparrow}$ seal row E remain attached to the same packet with provenance P and local naming N. Thus the carrier exports only the finite window route and none of the excluded compactness or choice principles. □

Theorem 1408.12 (Bolzano–Weierstrass selector finite-subsequence obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, a finite-subsequence read must display the bounded source row* B*, selector ledger* M*, inspected-threshold row* Q*, selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *before the read is available to a consumer. The obligations are finite window admission, selector-order replay, regular-rational readback, and Real-seal nonescape; they do not assert sequential compactness, countable choice, quotient-stream equality, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the displayed selector packet. The rows B, M, Q, W determine the finite subsequence surface, and every selected window is admitted by the packet row that names it.

Run finite window induction over M and Q. Each induction step consumes one displayed selector entry, reads the corresponding bounded source window from B, and appends exactly the matching $\mathsf{StreamName}^{\uparrow}$ row to W by hsame transport and Cont replay.

The readback and seal obligations are then projected from R, E, P, N. Since these rows are attached after the selected windows and contain no coordinate for compactness, countable choice, quotient equality, host equality, or ambient completeness, the finite-subsequence obligations stop at the displayed packet boundary. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteSubsequenceObligatio`

Theorem 1408.13 (Bolzano–Weierstrass selector root window)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, every root selected-window read must expose the bounded source windows* B*, monotone selector ledger* M*, cofinal row* Q*,* $\mathsf{StreamName}^{\uparrow}$ *row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural rows* H, C, P, N*. The read factors through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergence}$ $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and it exports no full Bolzano–Weierstrass theorem, sequential compactness, choice-selected subsequence, quotient stream, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRootWindow`

Proof. Begin with the subsequence handoff of Definition 1408.2. It routes the root selector surface from B, M, Q to W, D, R through the finite monotone subsequence packet and the stream, dyadic, and regular rational rows.

Next use the finite-subsequence obligations of Definition 1408.12. They require the same bounded source, selector, cofinal, selected-window, readback, seal, and structural rows to be displayed before any finite selected-window read is exported.

Now apply the bounded-monotone route of Definition 1408.4. It places the Real seal behind $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ and $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, so the root read remains a finite induction-style handoff rather than a compactness principle.

Finally use the nonescape theorem Definition 1408.9. Since the carrier has no row for full Bolzano–Weierstrass, sequential compactness, choice-selected subsequences, quotient streams, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the root window exports only the displayed finite route. □

Theorem 1408.14 (Bolzano–Weierstrass selector RegSeqRat readback nonescape)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ BHist *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from the selected windows must pass through the finite route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R$$

with the $\mathsf{Real}^{\uparrow}$ *seal row* E *available only after that readback is displayed. Componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row preserve the route; they do not let the read escape to a choice-selected subsequence, sequential compactness, quotient-stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegSeqRatReadbackNonescape`

Proof. Start with the finite-subsequence obligations of Definition 1408.12. They force every selected window in W to be displayed through B, M, Q before downstream readback can begin.

Project the dyadic and regular rows from the carrier. The $\mathsf{DyadicRatCore}^{\uparrow}$ row D is attached to the displayed selected windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is read only after those dyadic rows have been exposed.

Finally inspect the structural rows H, C, P, N and the $\mathsf{Real}^{\uparrow}$ seal row E. They replay and name the same finite route, so no hidden compactness, choice, quotient, host-equality, or ambient-completeness datum can be introduced at readback time. □

Definition 1408.15 (Bolzano–Weierstrass selector induction budget)**.** *The finite selector-induction budget of an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$ *is the displayed four-row ledger* B, M, Q, W*: the bounded source-window row* B*, the monotone selector ledger* M*, the inspected-threshold cofinality row* Q*, and the selected* $\mathsf{StreamName}^{\uparrow}$ *window row* W*. A budget step consumes one displayed entry of* M*, reads its bounded source window from* B*, verifies the inspected threshold in* Q*, and appends the admitted selected window to* W *through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*. The budget contains no hidden subsequence object, least-selector functional, countable choice row, or compactness row.*

Lean4 (definition) `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorInductionBudget`
Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorInductionBudget`
`selected window route`

Theorem 1408.16 (Bolzano–Weierstrass selector finite monotone induction)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, finite induction over the selector-induction budget of Definition 1408.15 admits each selected window through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*. The base stage exposes the empty selected-window family over the displayed bounded source row* B*; each successor stage consumes one ledger entry from* M*, checks the inspected threshold row* Q*, and appends exactly the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window to* W *by displayed* hsame *transport and* Cont *replay. No stage exports a choice-selected subsequence, sequential compactness, quotient-stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteMonotoneInduction`

Proof. Project the budget rows B, M, Q, W from the carrier. These rows are finite BHist data, so induction is over the displayed selector ledger rather than over an external sequence or function space.

The base case reads no selector entry. It exposes the empty selected-window family and records that no dyadic, regular, Real-facing, or compactness row is available.

For the successor step, consume the next displayed selector entry in M. The entry names its bounded source window in B, its inspected threshold in Q, and the selected $\mathsf{StreamName}^{\uparrow}$ window that is appended to W through the $\mathsf{MonotoneSubsequence}^{\uparrow}$ comparison row.

The transport rows H and replay rows C preserve the admitted window at each finite step. Since no carrier coordinate names a choice functional or a sequential compactness theorem, the induction admits exactly the selected windows listed in W. □

Theorem 1408.17 (Bolzano–Weierstrass selector finite-window stabilization)**.** *Let* $T = (B, M, Q, W, D, R, E, H, C$ *be an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet. If a selected finite* $\mathsf{StreamName}^{\uparrow}$ *window in* W *is replaced by a later displayed window admitted by the same monotone selector ledger* M *and inspected-threshold row* Q*, then the route*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

is preserved through $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}$ $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The stabilized packet records only this finite-window replacement route; it does not export full Bolzano–Weierstrass, sequential compactness, countable choice, quotient streams, host equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteWindowStabilization`

Proof. Project $B, M, Q, W, D, R, E, H, C, P, N$ from the carrier. The replacement hypothesis uses the same M and Q rows, so it remains inside the selector-induction budget of Definition 1408.15.

Apply Definition 1408.16 to the displayed later window. Finite induction over B, M, Q, W admits that window through the same $\mathsf{MonotoneSubsequence}^{\uparrow}$ comparison row, and H, C replay the transport without adding a subsequence object or a choice functional.

Now use the cluster and regular-Cauchy seal routes already proved in Definition 1408.8 and Definition 1408.19. The dyadic rows D, regular handoff R, and Real seal E are consumed after the stabilized selected window, so the downstream route is still the finite chain through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

No carrier coordinate remains for full Bolzano–Weierstrass compactness, sequential compactness, countable choice, quotient-stream equality, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The theorem therefore stabilizes only the displayed finite window. □

Theorem 1408.18 (Bolzano–Weierstrass selector finite nested-window induction)**.** *Let $T = (B, M, Q, W, D, R, E,$ be an accepted $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet. Suppose the bounded monotone route of Definition 1408.4, the nested-interval obligation of Definition 1408.22, the dyadic-window obligation of Definition 1408.21, the $\mathsf{StreamName}^{\uparrow}$ selected-window row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal row E are all displayed in the same BHist packet. Then the selector admits finite induction over the nested windows: each induction stage refines the current interval through the displayed nested-interval row, transports the matching dyadic tolerance through D, replays the same $\mathsf{StreamName}^{\uparrow}$ window through W, and hands the resulting regular readback to R before E can be inspected. The cluster handoff is therefore the same handoff as in Definition 1408.8, and no stage exports a selected infinite subsequence, countable choice, quotient-stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.*

Proof. Project the finite-window stabilization rows from Definition 1408.17. They keep the route

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

inside the accepted packet and leave H, C, P, N as structural transport, replay, provenance, and local naming rows.

Induct on the displayed nested-window budget. The base stage is the current selected window in W together with its bounded monotone route, dyadic tolerance row, regular readback row, and Real seal row.

For the successor step, use Definition 1408.22 to refine the interval row and Definition 1408.21 to transport the matching dyadic tolerance through D. The selected $\mathsf{StreamName}^{\uparrow}$ row remains the displayed window row W, so the successor step has no independent subsequence selector.

Now replay the readback through R and the Real seal through E. The cluster route of Definition 1408.8 and the nonescape boundary of Definition 1408.9 close the handoff through the same rows. Since no carrier coordinate names a selected infinite subsequence, countable-choice schedule, quotient-stream equality, host real equality, or ambient completeness theorem, the finite induction preserves only the displayed cluster handoff. □

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorFiniteNestedWindowInductio`

Theorem 1408.19 (Bolzano–Weierstrass selector regular Cauchy cluster seal)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the finite monotone route of Definition 1408.16 reaches the cluster-facing seal only through*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The rows D, R, E *are read respectively through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the bounded monotone convergence seal route. The resulting consumer receives a regular Cauchy cluster boundary for the displayed selected windows, not classical sequential compactness, arbitrary choice, host real equality, quotient-stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyClusterSeal`

Proof. First apply Definition 1408.16. It admits the selected windows in W by finite induction over B, M, Q, W and makes $\mathsf{MonotoneSubsequence}^{\uparrow}$ the only selector surface.

Next read the dyadic tolerance ledger D and regular handoff row R. The carrier definition attaches each row in D to a selected window in W, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is available only after those dyadic rows are displayed.

Finally use the bounded monotone Cauchy witness and bounded monotone convergence seal routes named in Definition 1408.4. They expose the Real-facing row E after W, D, R and do not add compactness, choice, quotient-stream equality, host equality, or ambient completeness. □

Theorem 1408.20 (Bolzano–Weierstrass selector bounded-subsequence carrier)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the bounded-subsequence carrier surface exposes bounded source windows and monotone selected windows by finite selector induction over* B, M, Q, W *only. The induction may read the bounded window ledger* B*, the monotone selector ledger* M*, the inspected-threshold row* Q*, and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, but it exports no ambient subsequence object, no countable choice functional, and no sequential compactness row.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorBoundedSubsequenceCarrier`

Proof. Project the carrier rows from Definition 1408.1. The rows B, M, Q, W are the only displayed selector rows before dyadic, regular, Real-facing, and structural rows are consumed.

Induct over the finite selector ledger M together with the inspected-threshold row Q. The base case reads the initial bounded source window from B and exposes the empty selected window family. The successor step consumes one displayed selector entry, verifies its comparison row through $\mathsf{MonotoneSubsequence}^{\uparrow}$, and appends the corresponding selected $\mathsf{StreamName}^{\uparrow}$ window to W.

Transport and continuation rows preserve the same finite route. Since the carrier has no row for an ambient subsequence, choice functional, or compactness theorem, the bounded-subsequence surface is exactly the finite induction over B, M, Q, W. □

Theorem 1408.21 (Bolzano–Weierstrass selector dyadic-window obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, selected windows reach dyadic tolerance rows only through the displayed route*

$$B, M, Q \longmapsto W \longmapsto D.$$

Thus every dyadic read is mediated by $\mathsf{StreamName}^{\uparrow}$ *selected windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, not by a host metric limit, quotient-stream equality, or an uninspected source window.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorDyadicWindowObligation`

Proof. Begin with Definition 1408.20. It supplies the finite selected-window route $B, M, Q \longmapsto W$ and excludes any hidden subsequence object.

Now inspect the dyadic ledger D in Definition 1408.1. Each tolerance row in D is attached to a displayed selected $\mathsf{StreamName}^{\uparrow}$ window in W, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance data before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read is available.

The componentwise hsame and Cont rows replay this displayed handoff. They cannot add quotient-stream equality, host metric limits, or uninspected source windows, so every dyadic tolerance read factors through W and D. □

Theorem 1408.22 (Bolzano–Weierstrass selector nested-interval obligation)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, any consumer that reads the selected cofinal windows as a regular rational or Real-facing limit must first route those windows through the sibling* $\mathsf{NestedInterval}^{\uparrow}$ *and* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ *rows. In particular, the displayed route is*

$$B, M, Q \longmapsto W \longmapsto D \longmapsto \mathsf{NestedInterval}^{\uparrow}, \mathsf{NestedIntervalCompleteness}^{\uparrow} \longmapsto R \longmapsto E.$$

The route consumes bounded monotone and dyadic window data before the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals are read, and it exports no interval-compactness theorem, selected endpoint, countable choice schedule, quotient stream, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorNestedIntervalObligation`

Proof. Begin with Definition 1408.20 and Definition 1408.21. They expose the finite selected-window route from B, M, Q to W and then to the dyadic tolerance ledger D.

Next read the sibling nested-interval surfaces. The nested interval chapter supplies the retained-window order row, while the nested interval completeness chapter packages the decreasing-window read before any regular rational name is available. The selector packet can therefore consume R only after the selected dyadic windows have been recognized as nested retained windows.

Finally apply Definition 1408.23. It places the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the Real seal behind the same selected-window route, while the bounded-window exhaustion and regular Cauchy cluster seal keep the consumer inside the displayed packet. Thus no hidden compactness, choice, quotient, host-equality, or ambient-completeness datum enters the selector obligation. □

Theorem 1408.23 (Bolzano–Weierstrass selector regular Cauchy limit handoff)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real-facing row* E *is available only after the selected dyadic windows have passed through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *and the bounded monotone convergence seal route. The packet therefore hands a regular Cauchy limit boundary to* $\mathsf{Real}^{\uparrow}$*, not classical Bolzano–Weierstrass, a choice-selected subsequence, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRegularCauchyLimitHandoff`

Proof. Use Definition 1408.21 to expose B, M, Q, W, D as the only dyadic-window route. No Real-facing row is visible at this stage.

Read the regular handoff row R. By the carrier definition, R is attached to the selected dyadic windows and is preserved by the displayed hsame and Cont rows before the seal coordinate E can be consumed.

Finally apply Definition 1408.4. The bounded monotone Cauchy witness and bounded monotone convergence seal chapters place E behind the finite selector, stream, dyadic, and regular handoff route, so the exported object is a regular Cauchy limit boundary for the displayed packet rather than a compactness or completeness theorem. □

Theorem 1408.24 (Bolzano–Weierstrass selector cofinal window stability)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the inspected cofinal rows in* Q *remain tied to the displayed bounded source windows* B *and selected windows* W *under componentwise* hsame *transport and displayed* Cont *replay. A consumer may refine an inspected threshold only by replaying the same* B, M, Q, W *window route; it cannot replace the cofinal row by a choice-selected subsequence, an uninspected source window, or quotient-stream equality.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorCofinalWindowStability`

Proof. Project the carrier of Definition 1408.1. The cofinal row Q is stored beside the bounded source window row B, the monotone selector ledger M, and the selected $\mathsf{StreamName}^{\uparrow}$ windows W.

Apply Definition 1408.20. Its finite induction over B, M, Q, W makes every inspected threshold part of the displayed selector ledger rather than an external subsequence object.

Now use the dyadic-window and regular-Cauchy handoffs, Definition 1408.21 and Definition 1408.23. They keep each downstream read attached to the same selected windows before D, R, E, H, C, P, N are consumed. Thus hsame transport and Cont replay preserve the cofinal window boundary instead of adding a hidden selector. □

Theorem 1408.25 (Bolzano–Weierstrass selector Real seal finality)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelector}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the Real seal row* E *is terminal after the displayed route through* $\mathsf{MonotoneSubsequence}^{\uparrow}$*,* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$*,* $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*. Once* E *is read through that route, no further row may reinterpret the packet as full Bolzano–Weierstrass compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, or a choice-selected cluster point.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorRealSealFinality`

Proof. Start with the bounded-monotone route of Definition 1408.4. It orders the Real-facing read through the monotone subsequence packet, the bounded monotone Cauchy witness, the convergence seal, the stream windows, the dyadic tolerance rows, the regular rational handoff, and finally the Real seal.

Use Definition 1408.23. The selected dyadic windows reach E only after the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the bounded monotone convergence route have been displayed.

Finally apply Definition 1408.9. The carrier contains no coordinate after E for compactness, host equality, choice, quotient streams, or ambient completeness. The Real seal is therefore terminal for the finite L10 selector route. □

Theorem 1408.26 (Bolzano–Weierstrass selector bounded-window exhaustion)**.** *Every bounded-window consumer of an accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet must expose the rows*

$$B, M, Q, W, D, R, E, H, C, P, N$$

before it may read a selected-window, dyadic-window, regular-Cauchy, or Real-facing conclusion. The exposed rows exhaust the bounded-window surface: no consumer may cite a hidden subsequence object, an ambient compactness theorem, a least cluster selector, quotient-stream equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorBoundedWindowExhaustion`

Proof. Begin with Definition 1408.5. It already binds $B, M, Q, W, D, R, E, H, C, P, N$ as one displayed packet for every finite selected-window read.

Apply Definition 1408.20. The bounded source and selected-window surface is exhausted by the finite induction over B, M, Q, W.

Next use Definition 1408.21 and Definition 1408.23. These theorems force the downstream dyadic and regular-Cauchy reads through D and R before the Real seal E is visible.

The structural rows H, C, P, N transport, replay, source, and name the same carrier. Since the nonescape theorem Definition 1408.9 refuses all stronger endpoints, the displayed row family exhausts the bounded-window consumer surface. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ Real-facing read factors through $\mathsf{MonotoneSubsequence}$ $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, and $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$ before the $\mathsf{Real}^{\uparrow}$ seal is visible.

Independence witness: monotonesubsequence, boundedmonotonecauchywitness, boundedmonotoneconvergenceseal, streamname, regseqrat, dyadicratcore, real: $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded sequence windows, a monotone selector ledger, inspected-threshold cofinal evidence, selected stream windows, dyadic tolerance rows, a regular rational handoff, a Real seal boundary, transports, continuations, provenance, and local naming.

Theorem 1408.27 (Bolzano–Weierstrass selector scoped L10 kernel)**.** *For every accepted* $\mathsf{BolzanoWeierstrassSelecto}$ *packet* $T = (B, M, Q, W, D, R, E, H, C, P, N)$*, the carrier, bounded-window exhaustion, monotone-subsequence route, nested-interval obligation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal finality depend only on* BHist*,* hsame*,* Cont*,* $\mathsf{ProbeBundle}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows displayed in the packet and its cited L10 siblings. No proof step requires a host compactness theorem, a choice-selected subsequence, quotient-stream equality, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorScopeClosure`

Proof. Project the carrier of Definition 1408.1. The rows $B, M, Q, W, D, R, E, H, C, P, N$ are finite BHist data, with componentwise hsame transport and displayed Cont replay as the only structural routes.

Apply the finite-window induction of Definition 1408.6. It binds the monotone-subsequence route to displayed selector entries and selected $\mathsf{StreamName}^{\uparrow}$ windows, not to an external selector functional.

Use the nested-interval obligation of Definition 1408.22. The retained-window route is consumed as sibling L10 data before any regular rational readback, so the proof remains inside the displayed finite window surface.

Finally apply the regular Cauchy limit handoff and Real seal finality, Definition 1408.23 and Definition 1408.25. They place the $\mathsf{RegSeqRat}^{\uparrow}$ row and the $\mathsf{Real}^{\uparrow}$ seal after $\mathsf{DyadicRatCore}^{\uparrow}$ and the selected $\mathsf{StreamName}^{\uparrow}$ windows. Since the nonescape theorem excludes compactness, choice, quotient, host equality, and ambient completeness rows, the stated L10 kernel scope is closed. □

Theorem 1408.28 (Bolzano–Weierstrass selector public export)**.** *An accepted* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *packet exports exactly the finite handoff from bounded sequence windows through monotone-subsequence*

selection, bounded monotone Cauchy witness, bounded monotone convergence seal, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows. The exported public surface is the row family*

$$B, M, Q, W, D, R, E, H, C, P, N,$$

read in that dependency order and no stronger order. It does not claim sequential compactness, countable choice, quotient-stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BolzanoWeierstrassSelectorUp.BolzanoWeierstrassSelectorPublicExport`

Proof. Begin with the selector handoff of Definition 1408.2 and the real seal boundary of Definition 1408.3. These place the selected subsequence surface and the Real-facing seal behind displayed packet rows rather than behind a compactness theorem.

Next apply the bounded-monotone route, finite-window stabilization, and regular Cauchy cluster seal through Definition 1408.4, Definition 1408.6, and Definition 1408.23. The route forces the monotone-subsequence selection, bounded monotone Cauchy witness, bounded monotone convergence seal, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff to appear before E can be consumed.

Finally use the scoped L10 kernel theorem, Definition 1408.27. It exhausts the public export by $B, M, Q, W, D, R, E, H, C, P, N$ and refuses any hidden compactness, choice, quotient, host-equality, or ambient-completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ packet is generated from bounded source windows, a monotone selector ledger, inspected-threshold cofinal evidence, selected StreamName windows, DyadicRatCore tolerance rows, a RegSeqRat handoff, a RealUp seal boundary, componentwise transports, displayed continuations, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1408.1, Definition 1408.2, Definition 1408.3, Definition 1408.4, Definition 1408.5, Definition 1408.6, Definition 1408.22, Definition 1408.23, Definition 1408.26, Definition 1408.27, and Definition 1408.28 bind the carrier, subsequence handoff, Real seal boundary, bounded-monotone route, window obligation, finite-window induction, nested-interval obligation, regular Cauchy limit handoff, bounded-window exhaustion, scoped L10 kernel, and public finite handoff surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim full Bolzano-Weierstrass, sequential compactness, choice-selected subsequences, quotient-stream equality, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite selector stability, inspected-threshold cofinal coverage, selected-window RegSeqRat handoff, bounded monotone witness routing, and Real seal boundary non-escape for BEDC.Derived.BolzanoWeierstrassSelectorUp.

Chapter 1409

A Concrete Naming Certificate for $\mathsf{RationalDensityReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RationalDensityReal}^{\uparrow}$ packet records the finite rational-witness interface for displayed $\mathsf{Real}^{\uparrow}$ intervals. It sits after the Real seal of chapter 38, the regular-rational readback of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance arithmetic of chapter 291. It routes located interval data through the rational interval surface of chapter 354 and the apartness-gap boundary of chapter 304, without reading a host real order.

Definition 1409.1 (RationalDensityReal carrier)**.** *A* $\mathsf{RationalDensityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, W_L, W_U, A, I, Q, T, R, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint names, each carrying a* $\mathsf{Real}^{\uparrow}$ *seal row and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected from the two endpoint names.*
3. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-gap row displayed between the endpoint seals.*
4. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *row whose rational bounds are read from the inspected endpoint windows.*
5. *Q is the visible* $\mathsf{Rat}^{\uparrow}$ *witness row selected between the displayed rational interval bounds.*
6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row certifying that the inspected endpoint windows and the rational witness are read at one dyadic scale.*
7. *R is the dyadic radius row that records the positive subinterval around Q still contained in the displayed interval.*
8. *H stores componentwise* hsame *transports for the endpoint, window, apartness, interval, witness, tolerance, and radius rows.*
9. *C stores the* Cont *replay from endpoint windows to the rational interval row, from the rational interval row to the witness row, and from the witness row to the radius row.*
10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^\uparrow}$ *row.*

The carrier stores a displayed rational witness for a displayed interval. It has no host order coordinate, no trichotomy row, no quotient equality of real names, no completeness principle, no Heine–Borel cover row, and no hidden choice selector.

Theorem 1409.2 (RationalDensityReal NameCert obligation surface)**.** *For an accepted* $\mathsf{RationalDensityReal}^\uparrow$ *carrier* $D = (L, U, W_L, W_U, A, I, Q, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^\uparrow}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed endpoint, window, apartness, rational-interval, witness, tolerance, radius, transport, continuation, provenance, and naming rows;*
2. *classifier transport along the stored* hsame *rows and the displayed* Cont *routes;*
3. *witness visibility, stating that the* $\mathsf{Rat}^\uparrow$ *row Q is exposed before any consumer reads the radius row R;*
4. $\mathsf{Real}^\uparrow$ *seal nonescape, stating that endpoint seals are consumed only through the stored* $\mathsf{RegSeqRat}^\uparrow$ *readbacks and* $\mathsf{StreamName}^\uparrow$ *windows;*
5. *ledger exactness, stating that the packet exports no host real order, trichotomy, quotient equality, completeness theorem, Heine–Borel cover, or hidden selector.*

Lean4 ✓ `BEDC.Derived.RationalDensityRealUp.RationalDensityRealTasteGate single carrier alignment`

Proof. Carrier admission is the tuple in Definition 1409.1. The endpoint names, stream windows, apartness row, rational interval row, witness row, dyadic tolerance, dyadic radius, transports, continuations, package row, and local naming row are all finite BHist coordinates.

Classifier transport reads H and C componentwise. The transported endpoint windows still feed the same rational-interval row shape, and the transported interval still exposes the same witness-and-radius sequence.

Witness visibility follows from the carrier order: the row Q is stored before the radius row R, and the continuation row records the replay from I to Q before replaying from Q to R.

The seal nonescape and ledger exactness are inventory checks. The endpoint seals occur only beside their regular-rational readbacks and finite windows, while host order, trichotomy, quotient equality, completeness, Heine–Borel data, and hidden selection are not rows of the packet. □

Theorem 1409.3 (RationalDensityReal located-window witness)**.** *Let L and U be displayed* $\mathsf{Real}^\uparrow$ *endpoint names with* $\mathsf{RegSeqRat}^\uparrow$ *readbacks, finite* $\mathsf{StreamName}^\uparrow$ *windows, and an* $\mathsf{ApartnessReal}^\uparrow$ *positive-gap row between them. If the inspected windows determine a displayed* $\mathsf{RationalInterval}^\uparrow$ *row whose dyadic tolerance is recorded by* $\mathsf{DyadicRatCore}^\uparrow$*, then an accepted* $\mathsf{RationalDensityReal}^\uparrow$ *packet exposes a visible* $\mathsf{Rat}^\uparrow$ *witness row inside that displayed interval. Every consumer route to the witness factors through*

$$(L, U) \longmapsto (W_L, W_U) \longmapsto A \longmapsto I \longmapsto Q$$

and through the dyadic tolerance and radius rows T, R; no step appeals to host real order, trichotomy, quotient equality, Real completeness, or Heine–Borel compactness.

Proof. Read the endpoint data through their displayed regular-rational rows. The finite stream windows W_L, W_U are the only observations used to form the rational interval row, so the route begins inside $\mathsf{RegSeqRat}^\uparrow$ and $\mathsf{StreamName}^\uparrow$.

The positive-gap row A makes the interval located at the displayed scale. The dyadic row T records that the endpoint windows and the rational interval bounds are compared inside $\mathsf{DyadicRatCore}^{\uparrow}$, not inside an ambient order on completed reals.

The rational witness is then the row Q of the carrier. The continuation path reads I before Q and records the radius row R only after Q is visible.

Thus the witness route is exactly the displayed chain through $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$. Any route using host real order, trichotomy, quotient equality, Real completeness, or Heine–Borel compactness leaves the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalDensityReal}^{\uparrow}$ packet is generated from two displayed RealUp endpoint names, their RegSeqRat readbacks, finite StreamName windows, an ApartnessReal positive-gap row, a RationalInterval row, a visible Rat witness, dyadic tolerance and radius rows, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1409.1, Definition 1409.2, and Definition 1409.3 declare the finite rational-witness surface for displayed RealUp intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalDensityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, trichotomy, quotient equality of real names, host real order, Heine–Borel compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, witness visibility, seal nonescape, and ledger exactness for the same finite packet.

Chapter 1410

A Concrete Naming Certificate for $\mathsf{BishopIntervalCompact}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopIntervalCompact}^{\uparrow}$ packet records the finite-net compact interval surface for displayed real-name intervals. It is anchored in the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular-rational readback surface of chapter 301, the visible real seal of chapter 38, the comparison boundary of chapter 1381, and the rational witness route of chapter 1409. The chapter treats compactness as a finite-net packet over those rows, not as open-cover compactness or a choice-bearing completeness principle.

A $\mathsf{BishopIntervalCompact}^{\uparrow}$ carrier is a finite BHist packet

$$K = (A, B, W, T, G, Q, R, H, C, P, N)$$

with the following displayed rows:

1. A and B are the lower and upper endpoint names, each read through $\mathsf{RegSeqRat}^{\uparrow}$ rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, and a visible $\mathsf{Real}^{\uparrow}$ seal.
2. W is the bounded interval window ledger naming the endpoint windows inspected at each dyadic tolerance.
3. T is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance scale used to compare endpoint windows and interval mesh points.
4. G is the finite grid row listing the rational mesh points visible at the chosen dyadic scale.
5. Q is the rational-density witness ledger that routes each mesh cell through $\mathsf{RationalDensityReal}^{\uparrow}$ rather than through host real order.
6. R is the Bishop comparison row recording the same endpoint, readback, and seal surface as a $\mathsf{BishopRealComparison}^{\uparrow}$ packet.
7. H stores componentwise hsame transports for endpoints, windows, tolerance, grid, witness, comparison, continuation, package, and local naming rows.
8. C stores the displayed Cont replay from endpoint windows to dyadic scale, from dyadic scale to grid, and from grid cells to rational witnesses.
9. P is the Pkg provenance over $\mathsf{BishopIntervalCompact}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{RationalDensityReal}^{\uparrow}$, BHist, hsame, Cont, and $\mathsf{NameCert}$ rows.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopIntervalCompact}^{\uparrow}}$ row.

The packet gives a finite interval-net surface: every consumer read factors through the displayed endpoint windows, dyadic tolerance, finite grid, rational-density witnesses, comparison row, transport, continuation, provenance, and local naming. It does not supply a cover selector, a subsequence selector, an ambient Heine–Borel theorem, quotient equality of real names, or a host-real equality test.

Theorem 1410.1 (BishopIntervalCompact finite-grid obligation)**.** *Let $K = (A, B, W, T, G, Q, R, H, C, P, N)$ be an accepted* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *carrier. The accepted packet projects the lower endpoint row A, the upper endpoint row B, the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance scale T, the finite rational grid G, the* $\mathsf{RationalDensityReal}^{\uparrow}$ *witness ledger Q, the* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison row R, the componentwise* hsame *transport H, the displayed* Cont *replay C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalCompact}^{\uparrow}}$ *row N. Every compact-interval read admitted by the packet is exhausted by those rows and by no hidden open-cover, host-real, quotient-equality, or choice-bearing coordinate.*

Proof. Project the accepted carrier. Its displayed tuple has exactly the rows $A, B, W, T, G, Q, R, H, C, P, N$, so every endpoint, tolerance, grid, witness, comparison, transport, replay, provenance, and local naming request starts from one of these coordinates.

The finite-grid classifier cannot inspect data outside the displayed tuple. Endpoint reads pass through A, B, W, T and then through the comparison row R; mesh reads pass through T, G, Q; transport and replay reread the same coordinates through H and C.

Since the carrier has no coordinate for a cover selector, a subsequence selector, quotient equality of real names, or host-real equality, a consumer has no route to such data. Thus the accepted compact-interval surface is exhausted by the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.BishopIntervalCompactUp.BishopIntervalCompactTasteGate single carrier alignment`

Theorem 1410.2 (BishopIntervalCompact rational-density routing)**.** *For every accepted* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *carrier $K = (A, B, W, T, G, Q, R, H, C, P, N)$, each mesh-cell witness in Q must route through the rational-density surface of chapter 1409, the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, and the visible real seal of chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is visible.*

Proof. Begin with a mesh-cell witness named by Q. By construction, Q is the rational-density witness ledger for the grid row G, so the witness is not a host-real order comparison; it is a displayed $\mathsf{RationalDensityReal}^{\uparrow}$ route attached to the finite mesh cell.

The tolerance requested by the witness is the dyadic row T. Its endpoint and mesh reads are inspected through the finite windows W, and those windows have regular rational readbacks before they can expose a real-name seal.

The visible seal is therefore reached only after the route has passed through rational density, dyadic tolerance, stream windows, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ certificate surface. Componentwise transport H and continuation replay C preserve this route rather than introducing a separate equality test. □

Theorem 1410.3 (BishopIntervalCompact compact-metric handoff)**.** *Any compact-metric or interval consumer of an accepted* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *carrier $K = (A, B, W, T, G, Q, R, H, C, P, N)$ receives only the finite-net packet consisting of endpoint rows, finite windows, dyadic tolerance, finite grid, rational-density witnesses, comparison row, transport, replay, provenance, and local*

naming. The handoff exports no open-cover compactness, sequential compactness, quotient equality, selected subcover, selected subsequence, countable-choice schedule, or host real equality.

Proof. The consumer boundary is the packet boundary. A downstream compact-metric or interval consumer can read $A, B, W, T, G, Q, R, H, C, P, N$ and can ask the replay row C to reproduce the displayed route from endpoint windows to dyadic tolerance, finite grid, rational witnesses, and comparison rows.

This boundary exports a finite-net object: the grid G is finite, the windows W are displayed, and each witness in Q is routed through the named rational-density and real-seal surfaces. The comparison row R supplies the endpoint boundary without replacing the packet by host equality of real names.

No coordinate of K carries an open cover, an infinite sequence, a selected subcover, a selected subsequence, a quotient-equality proof, or a host-real equality proof. Hence every downstream handoff is a finite-net handoff and not a classical compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalCompact}^{\uparrow}$ packet is generated from displayed endpoint real names, regular-rational readbacks, finite stream windows, a dyadic tolerance scale, a finite rational grid, rational-density witnesses, a Bishop comparison row, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1410.1, Definition 1410.2, and Definition 1410.3 seed the finite-net compact interval packet over the DyadicRatCore, StreamName, RegSeqRat, Real, BishopRealComparison, and RationalDensityReal surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, sequential compactness, Heine–Borel compactness for arbitrary subsets, countable choice, selected limits, or host real equality.

Upgrade path: A later verification route can encode the carrier and prove the finite-grid obligation, rational-density routing, compact-metric handoff, comparison-row nonescape, and ledger exactness for the displayed packet.

Chapter 1411

A Concrete Naming Certificate for $\mathsf{SeparatedMetric}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{SeparatedMetric}^{\uparrow}$ packet records the constructive separation surface needed by completion-limit uniqueness. It refines the metric carrier of chapter 57, consumes the Cauchy-limit rows of chapter 128, respects the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38, and reads regular-name comparison through chapter 1412. The packet treats equality of displayed limit names as a finite classifier consequence of zero-distance observations, not as host equality, trichotomy, quotient formation, open-cover compactness, or ambient real completeness.

Definition 1411.1 (SeparatedMetric carrier surface)**.** *A* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, D, Z, L_0, L_1, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *carrier row, including its point classifier and metric-distance witness surface.*
2. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance witness interface read from metric-distance rows through the displayed real seal.*
3. *Z is the separation row: whenever two carried observations have displayed zero distance through D, the row records the classifier equivalence between the two observations.*
4. *L_0 and L_1 are* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-limit rows for the same regular Cauchy point name, including their displayed moduli and convergence distance ledgers.*
5. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route that identifies the two limit claims as readings of one regular Cauchy name through finite windows and dyadic tolerances.*
6. *H stores componentwise* hsame *transport for the metric carrier, real-valued distance witness, separation row, Cauchy-limit rows, equivalence route, continuation row, package row, and local naming row.*
7. *C stores the* Cont *replay from finite Cauchy observations to metric-distance witnesses, from those witnesses to* $\mathsf{Real}^{\uparrow}$ *zero-distance readings, and from zero-distance readings to the classifier equivalence row.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^\uparrow}$ *row.*

The carrier exposes separatedness only as a finite zero-distance-to-classifier route. It carries no quotient metric-space object, no host equality coordinate, no trichotomy row, no open-cover compactness principle, and no assertion that $\mathsf{Real}^\uparrow$ *is complete.*

Lean4 (definition) `BEDC.Derived.SeparatedMetricUp.SeparatedMetricUp`

Definition 1411.2 (SeparatedMetric classifier surface)**.** *The classifier of an accepted* $\mathsf{SeparatedMetric}^\uparrow$ *packet relates two carried observations a, b precisely when the packet displays a finite route*

$$a, b \longmapsto \mathsf{MetricDistanceWitness}(a, b, d) \longmapsto D(d, 0) \longmapsto Z(a, b).$$

The first step is a $\mathsf{Met}^\uparrow$ *distance-witness row, the second step is a* $\mathsf{Real}^\uparrow$ *zero-distance reading of that witness, and the final step is the separation row identifying a and b inside the point classifier of X. Transport of this classifier is componentwise: if $a, b, d, Z(a, b)$ are moved by the stored* hsame *rows, then the replayed* Cont *path produces the transported zero-distance reading and the transported classifier equivalence.*

This classifier is not a quotient-by-equivalence-class construction. It is a local recognition surface for carried observations whose zero-distance evidence is already displayed in the packet.

Theorem 1411.3 (SeparatedMetric NameCert obligations)**.** *For an accepted* $\mathsf{SeparatedMetric}^\uparrow$ *carrier $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^\uparrow}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the* $\mathsf{Met}^\uparrow$ *carrier row,* $\mathsf{Real}^\uparrow$*-valued distance witness row, separation row, two* $\mathsf{CompleteMetric}^\uparrow$ *Cauchy-limit rows,* $\mathsf{CauchySequenceEquivalence}^\uparrow$ *route, componentwise transport, continuation replay, provenance row, and local naming row;*
2. *classifier stability, stating that zero-distance observations transported by H and replayed by C still yield the transported classifier equivalence through Z;*
3. *completion compatibility, stating that the two limit rows L_0, L_1 are read as limits of the same regular Cauchy point name only through the displayed equivalence route E;*
4. *zero-distance exactness, stating that the only route from metric equality evidence to point-classifier equivalence factors through the* $\mathsf{Real}^\uparrow$ *zero-distance witness and the separation row;*
5. *ledger visibility, stating that the packet exports no quotient metric space, no host equality, no trichotomy, no open-cover compactness, and no* $\mathsf{Real}^\uparrow$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket semantic name certificate`

Proof. Carrier admission is the finite tuple of Definition 1411.1. Every row named in the obligation list is visible in S, and no extra equality or compactness row is available to a consumer.

Classifier stability follows from the paired transport and replay rows. The hsame fields move endpoints, distance witnesses, zero-distance readings, limit rows, and the separation row componentwise, while the Cont row replays the same finite route after transport.

Completion compatibility reads the two limit rows only through the common regular Cauchy name witnessed by E. The route never collapses names into quotient classes; it only shows that both displayed limit claims are attached to one finite-window comparison surface.

Zero-distance exactness is the defining use of the separation row. Once the transported distance witness is read as zero through D, the row Z is the unique exported bridge to the point classifier of X. The final ledger statement is an inventory check on the fields of S. □

Theorem 1411.4 (SeparatedMetric completion limit uniqueness route)**.** *Let S be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet whose rows L_0 and L_1 are two* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-limit witnesses for the same regular Cauchy point name, with sameness displayed by the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route E. Then the packet exposes a constructive uniqueness route from the two limit names to the point classifier of the metric carrier:*

$$L_0, L_1 \longmapsto E \longmapsto \mathsf{MetricDistanceWitness}(\ell_0, \ell_1, d) \longmapsto D(d, 0) \longmapsto Z(\ell_0, \ell_1).$$

Here ℓ_0 and ℓ_1 are the displayed limit histories read from L_0 and L_1. The conclusion is classifier equivalence inside the carried $\mathsf{Met}^{\uparrow}$ *surface. It is not host equality, not equality of quotient classes, and not an appeal to trichotomy, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket completion limit uniqueness route`

Proof. Read the two complete-metric limit rows L_0 and L_1 and use E to keep them attached to the same regular Cauchy point name. The equivalence route supplies the finite-window comparison data needed to compare the two convergence ledgers at a common displayed tolerance.

Replay the metric-distance construction through C. The convergence ledgers for L_0 and L_1 yield a displayed distance witness between the two limit histories; the real-valued distance interface D reads that witness as zero in the packet's $\mathsf{Real}^{\uparrow}$ seal.

Apply the separation row Z to this zero-distance reading. The result is classifier equivalence between ℓ_0 and ℓ_1 in the underlying metric carrier X.

Each step is a row of S or a replay of a row of S. Since quotient classes, host equality, trichotomy, open-cover compactness, and $\mathsf{Real}^{\uparrow}$ completeness are not fields of the packet, none can be used by this route. □

Theorem 1411.5 (SeparatedMetric Real completion consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer of an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet can use the packet only through the displayed chain*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow}.$$

Concretely, the consumer may read the metric carrier, the two complete-limit rows, the finite-window equivalence route, the real-valued zero-distance witness, the separation row, the componentwise transport rows, the continuation replay, package provenance, and the local naming row. The consumer may not infer quotient metric spaces, host equality of histories, trichotomy of real comparisons, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$ *from this packet.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket real completion consumer boundary`

Proof. The carrier inventory in Definition 1411.1 lists exactly the rows available to a consumer. The displayed chain names the order in which those rows can be replayed: metric data first, complete-limit data second, finite-window comparison third, real-valued zero-distance reading fourth, and separation classification last.

The hsame and Cont rows preserve this order under transport, so a consumer cannot bypass the zero-distance witness or replace the classifier conclusion with host equality. The package row records only the named BEDC surfaces in the chain.

The non-claims are again field exclusions. No quotient carrier, trichotomy ledger, open-cover compactness row, or $\mathsf{Real}^{\uparrow}$ completeness row occurs in S, so no consumer boundary licensed by N can read such a principle from the packet. □

Theorem 1411.6 (SeparatedMetric zero-distance consumer exactness)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet. Every consumer that reads a zero-distance identification from S must consume, in order, the* $\mathsf{Met}^{\uparrow}$ *distance witness in X, the* $\mathsf{Real}^{\uparrow}$ *zero-distance read through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* NameCert *row N before the separated classifier row Z may be exported. If the zero-distance read is attached to completion data, the same consumer must first expose the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows L_0, L_1 and the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route E.*

Thus zero-distance consumption is exact at the displayed rows of S. It is not host equality, quotient equality, selected representative data, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1411.1. The metric source is X, the real-valued zero-distance witness is D, and the classifier export is stored only in the separation row Z.

Apply the classifier surface of Definition 1411.2 together with the zero-distance carrier row of Definition 1411.14. These rows force a consumer to form the $\mathsf{Met}^{\uparrow}$ distance witness before the $\mathsf{Real}^{\uparrow}$ zero-distance read, and to use Z only after that read is displayed.

The structural rows H, C, P, N then transport, replay, source, and name the same finite route. When completion-limit rows are involved, Definition 1411.18 inserts L_0, L_1, E before the distance witness is formed.

No other coordinate occurs in S. Therefore every accepted zero-distance consumer is exhausted by the displayed route and cannot obtain any excluded equality or completeness principle. □

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricZeroDistanceConsumerExactness`

Theorem 1411.7 (SeparatedMetric Cauchy uniqueness root package)**.** *An accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet exports completion-limit uniqueness to downstream L10 completion consumers only through the finite package*

$$X, D, Z, L_0, L_1, E, H, C, P, N.$$

Here X is the metric row, D is the $\mathsf{Real}^{\uparrow}$ *zero-distance classifier read, Z is the separated zero-distance classifier, L_0, L_1 are the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row, and H, C, P, N are the transport, replay, provenance, and local NameCert rows. The package exports classifier equivalence of the displayed limit names and nothing stronger: no quotient representative, host equality, trichotomy, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is part of the handoff.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket cauchy uniqueness root package`

Proof. First use the completion-limit uniqueness route of Definition 1411.4. It reads the two complete-metric limit rows through the same Cauchy-sequence equivalence row before forming the metric-distance witness in X.

Next apply Definition 1411.6. That theorem forces every zero-distance consumer to pass through the $\mathsf{Real}^{\uparrow}$ read D, componentwise transport H, displayed replay C, package provenance P, and the local naming row N before the separated classifier row Z is exported.

Combining the two routes gives a single root package for downstream completion chapters. Since the accepted packet contains no quotient representative, host-equality coordinate, trichotomy ledger, selected-limit field, or ambient completeness theorem, the exported uniqueness claim remains exactly the displayed separated-classifier handoff. □

Theorem 1411.8 (SeparatedMetric limit uniqueness scope)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, every limit-uniqueness claim over the two* $\mathsf{CompleteMetric}^{\uparrow}$ *rows* L_0, L_1 *is scoped by the route*

$$L_0, L_1 \longmapsto E \longmapsto X \longmapsto D \longmapsto H, C, P, N \longmapsto Z.$$

Here E *is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route for the two displayed Cauchy presentations,* X *supplies the* $\mathsf{Met}^{\uparrow}$ *distance witness between the two displayed limit histories,* D *reads that witness as* $\mathsf{Real}^{\uparrow}$ *zero distance, and* Z *supplies the separated classifier conclusion. The scope contains no quotient completion equality, host limit equality, selected representative, countable choice, trichotomy, or ambient completeness field.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket limit uniqueness scope`

Proof. Begin with the completion compatibility clause of Definition 1411.3. It requires the two $\mathsf{CompleteMetric}^{\uparrow}$ limit rows to be read through the displayed $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ route E before their endpoints are compared.

Use the completion limit uniqueness route of Definition 1411.4. After E binds the two limit rows to one regular Cauchy point name, the route forms the metric-distance witness in X and reads it as zero through D.

Finally apply Definition 1411.6. The support rows H, C, P, N preserve and name the same route before the classifier row Z is exported. Since the packet has no quotient, host-limit, selected-representative, choice, trichotomy, or ambient-completeness coordinate, the uniqueness claim remains inside this scope. □

Theorem 1411.9 (SeparatedMetric completion handoff nonescape)**.** *Every completion-facing handoff from an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet to a downstream L10 consumer must pass through the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1*, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route* E*, the* $\mathsf{Met}^{\uparrow}$ *distance witness in* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance read* D*, the separated classifier row* Z*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*. The handoff exports only that routed separated-classifier evidence; it does not export a quotient completion carrier, selected limit, host equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket completion handoff nonescape`

Proof. Start with Definition 1411.5. It lists the rows a Real or completion consumer may read from the packet and places the complete-limit and finite-window comparison rows before the real zero-distance and separated-classifier rows.

Apply Definition 1411.8. The limit rows L_0, L_1 enter the handoff only through E, then through the metric witness in X, the zero-distance row D, and the classifier row Z.

Use Definition 1411.6 for the support rows. Transport, replay, provenance, and local naming pass through H, C, P, N and do not add a second carrier or a hidden selector.

The nonescape conclusion is a field inventory. Since no quotient completion, selected limit, host equality, trichotomy, countable-choice, or ambient-completeness row is present in S, no downstream completion handoff can read one from the packet. □

Theorem 1411.10 (SeparatedMetric CompleteMetric consumer exactness)**.** *Any* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of separated limit uniqueness for an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *must pass through*

$$L_0, L_1 \longmapsto E \longmapsto X \longmapsto D \longmapsto Z \longmapsto H, C, P, N.$$

The consumer may read the displayed completion-limit rows, the Cauchy-sequence equivalence route, the metric distance witness, the real zero-distance row, the separated classifier, and the structural ledger. It cannot infer quotient completion equality, selected representatives, host equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1411.18. It places every comparison of the two $\mathsf{CompleteMetric}^{\uparrow}$ limit rows behind the displayed $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ route E before any metric or real-valued comparison is available.

Next apply Definition 1411.9. The completion handoff reads L_0, L_1, then E, then the metric witness in X, the real zero-distance row D, and the separated classifier row Z.

Finally use Definition 1411.6. The structural rows H, C, P, N transport, replay, source, and name the same route, while every excluded principle lacks a carrier coordinate. Hence a $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives exactly the displayed separated-limit route and no stronger equality or choice principle. □

Theorem 1411.11 (SeparatedMetric root-unblock completion package)**.** *For every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, the root-unblock package for completion-limit uniqueness is exactly the displayed row family*

$$X, D, Z, L_0, L_1, E, H, C, P, N.$$

Downstream $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers receive classifier equivalence only through these rows:* L_0, L_1 *and* E *expose the completion comparison,* X, D, Z *expose the zero-distance classifier route, and* H, C, P, N *transport, replay, source, and name the handoff.*

Proof. Apply Definition 1411.10. It already places every $\mathsf{CompleteMetric}^{\uparrow}$ consumer behind $L_0, L_1, E, X, D, Z, H, C,$

Now use Definition 1411.6. It keeps the classifier export behind the metric witness, the $\mathsf{Real}^{\uparrow}$ zero-distance read, the separated row, and the structural support rows.

The resulting package is a root-unblock surface because no other coordinate is present in the carrier. A consumer therefore receives only the displayed classifier equivalence, not quotient completion equality, host equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1411.12 (SeparatedMetric zero-distance root factorization)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every zero-distance root read used in completion-limit uniqueness factors first through the* $\mathsf{Met}^{\uparrow}$ *row* X*, then through the* $\mathsf{Real}^{\uparrow}$ *zero-distance row* D*, and only then through the separated classifier row* Z*, with* L_0, L_1, E *displayed when the read is completion-facing and* H, C, P, N *preserving the same BEDC route.*

Proof. Use Definition 1411.6. It forces the metric-distance witness and Real zero-distance read to precede any separated classifier export.

For completion-facing reads, apply Definition 1411.10. The two limit rows and the Cauchy-sequence equivalence row must be displayed before the zero-distance route is consumed.

The factorization is therefore rooted in the shown carrier rows. Since H, C, P, N only transport, replay, source, and name the same route, no downstream consumer can replace the classifier equivalence with host equality, quotient equality, selected representative data, trichotomy, countable choice, or ambient completeness. □

Theorem 1411.13 (SeparatedMetric CompleteMetric root handoff)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff that consumes completion-limit uniqueness from an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *receives the classifier equivalence through the displayed route*

$$L_0, L_1 \longmapsto E \longmapsto X \longmapsto D \longmapsto Z \longmapsto H, C, P, N.$$

The handoff is a BEDC row transfer only; it carries no quotient completion carrier, selected limit, host equality of completed names, trichotomy, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or off-packet certificate.*

Proof. Start with the completion handoff nonescape theorem, Definition 1411.9. It confines completion-facing consumers to the same limit, comparison, metric, Real, separatedness, and structural rows.

Then read Definition 1411.10. It identifies the exact order in which a $\mathsf{CompleteMetric}^{\uparrow}$ consumer may receive those rows.

Finally apply Definition 1411.11. The root package exhausts the carrier, so the handoff cannot acquire any excluded equality, selection, choice, completeness, or off-packet certificate from outside the displayed rows. □

Definition 1411.14 (SeparatedMetric zero-distance carrier row)**.** *The zero-distance carrier row of an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet is the displayed* BHist *route*

$$\mathsf{MetricDistanceWitness}(a, b, d) \longmapsto \mathsf{Real}^{\uparrow}(d, 0) \longmapsto Z(a, b),$$

where the first term is read from the $\mathsf{Met}^{\uparrow}$ *distance-witness surface of Definition 1411.1, the middle term is the* $\mathsf{Real}^{\uparrow}$ *zero-distance row already admitted by the carrier and local NameCert obligations of Definition 1411.3, and* $Z(a, b)$ *is the separation row consumed by the classifier of Definition 1411.2. The row is a finite route from a metric-distance witness to point-classifier equivalence. It is not host equality of points, quotient equality, trichotomy of distances, or an ambient completeness assertion for* $\mathsf{Real}^{\uparrow}$*.*

Theorem 1411.15 (SeparatedMetric classifier stability row)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet, transported zero-distance evidence replays through the stored* hsame *and* Cont *rows to the transported point classifier. Concretely, if a displayed route*

$$\mathsf{MetricDistanceWitness}(a, b, d) \longmapsto \mathsf{Real}^{\uparrow}(d, 0) \longmapsto Z(a, b)$$

is transported componentwise by H*, then* C *replays the transported distance witness, the transported real zero-distance reading, and the transported separation row before the classifier reads the transported pair.*

Proof. Start with the carrier of Definition 1411.1. Its rows H and C are stored for the metric carrier, real-valued distance witness, separation row, Cauchy-limit rows, equivalence route, continuation row, package row, and local naming row.

Apply classifier stability from Definition 1411.3. The transported endpoints and distance witness first move through the corresponding hsame fields. The replay row then reads the same zero-distance route through the transported $\mathsf{Real}^{\uparrow}$ row.

Finally apply the classifier surface of Definition 1411.2. Since the transported zero-distance evidence reaches the transported separation row before classification, the accepted classifier conclusion is the transported point-classifier equivalence and not a host equality or quotient claim. □

Theorem 1411.16 (SeparatedMetric limit uniqueness ledger)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet, two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows over the same regular Cauchy name are consumed only through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route before the separation row is read. In particular, the route*

$$L_0, L_1 \longmapsto E \longmapsto \mathsf{MetricDistanceWitness}(\ell_0, \ell_1, d) \longmapsto \mathsf{Real}^{\uparrow}(d, 0) \longmapsto Z(\ell_0, \ell_1)$$

is the only completion-limit uniqueness ledger exported by the packet.

Proof. Project the two complete-limit rows and the equivalence route from the carrier of Definition 1411.1. The NameCert obligations of Definition 1411.3 require completion compatibility to pass through E before the two limit histories are compared.

Use the completion-limit uniqueness route of Definition 1411.4. It constructs the displayed metric-distance witness between the two limit histories only after E has attached both rows to the same regular Cauchy point name.

The zero-distance carrier row of Definition 1411.14 then sends that witness through the $\mathsf{Real}^{\uparrow}$ zero-distance reading and into the separation row. Therefore the exported uniqueness ledger is classifier equivalence through Z, with no quotient completion, host equality, selected representative, countable choice, or ambient completeness row. □

Theorem 1411.17 (SeparatedMetric zero-distance transport)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet. If zero-distance evidence is displayed by the route*

$$\mathsf{MetricDistanceWitness}(a, b, d) \longmapsto \mathsf{Real}^{\uparrow}(d, 0) \longmapsto Z(a, b),$$

then every componentwise transport of this evidence through the stored $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*, and* Cont *rows yields the transported classifier route for the transported endpoints. The transported route still factors through* X, D, Z, H, C, P, N *and does not become host equality, quotient equality, or trichotomy of real distances.*

Proof. Project the zero-distance row from Definition 1411.14. The displayed evidence starts with a metric-distance witness in X, is read as zero through the $\mathsf{Real}^{\uparrow}$ seal row D, and reaches the classifier only through the separation row Z.

Apply the classifier stability row of Definition 1411.15. The stored hsame fields move the endpoints, distance witness, real zero-distance reading, and separation row componentwise, while the stored Cont row replays the same finite route after transport.

The provenance and local naming rows P, N name this transported route as part of the same packet. Since no host equality coordinate, quotient carrier, or trichotomy ledger occurs in S, the transported conclusion remains a $\mathsf{SeparatedMetric}^{\uparrow}$ classifier route and nothing stronger. □

Theorem 1411.18 (SeparatedMetric completion compatibility scope)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet. The two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1 *can be compared by the separation row only after both have passed through the displayed* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E*. Thus every completion-limit uniqueness read has the scoped order*

$$L_0, L_1 \longmapsto E \longmapsto D \longmapsto Z.$$

No consumer may bypass E *by comparing selected representatives, quotient classes, host limit values, or ambient complete-space points.*

Proof. Begin with the completion compatibility clause of Definition 1411.3. It states that L_0 and L_1 are read as two limit rows for one regular Cauchy point name only through the displayed equivalence route E.

Use the limit uniqueness ledger of Definition 1411.16. That ledger constructs the metric-distance witness between the displayed limit histories only after E has supplied the common finite-window comparison surface.

The zero-distance and classifier steps then use D and Z in that order. The rows H, C, P, N transport, replay, source, and name the same order; they do not add a representative selector or quotient comparison.

Therefore a completion consumer reaches separation only through E first. The absent host-limit, quotient, selected-representative, and ambient-completeness rows cannot supply an alternate comparison scope. □

Theorem 1411.19 (SeparatedMetric Bishop Cauchy criterion handoff)**.** *Every zero-distance uniqueness consumer of an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *completion carrier must pass through the finite Bishop Cauchy criterion windows of chapter 3576 before the separated classifier row is exported. The route may then read the existing* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 1411.18, but it exports no quotient equality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1411.18. It already forces the two completion-limit rows through E before any zero-distance classifier read.

Insert the Bishop Cauchy criterion windows before that equivalence read. The criterion supplies only finite tolerance and stream-window data, so the separated classifier remains a read of $X, D, Z, L_0, L_1, E, H, C, P, N$ and not a host equality or selected-limit export. □

Theorem 1411.20 (SeparatedMetric completion criterion nonescape)**.** *If a* $\mathsf{SeparatedMetric}^{\uparrow}$ *completion-limit uniqueness consumer reads a completion-criterion dependency row, that row must remain between the Bishop Cauchy criterion windows and the regular-name equivalence route before the classifier Z is exported. The dependency can be consumed through the completion siblings, but it does not supply quotient equality, selected representatives, countable choice, or ambient Real completeness.*

Proof. Use Definition 1411.19 for the finite Cauchy windows and Definition 1411.18 for the ordered route through E, D, Z.

The completion-criterion row is therefore a dependency inside the finite route, not an additional equality principle. Since the carrier has no coordinate for quotient comparison, selected representatives, choice, or ambient completeness, the separated classifier can only be exported after the dependency has been replayed through H, C, P, N. □

Theorem 1411.21 (SeparatedMetric regular-name uniqueness exhaustion)**.** *The regular-name uniqueness surface for an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *completion carrier is exhausted by Bishop Cauchy criterion windows, the completion-criterion dependency row, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route, the* $\mathsf{Real}^{\uparrow}$ *zero-distance witness, and the separated classifier row. No downstream consumer obtains a second uniqueness route from quotient equality, a selected limit, countable choice, or ambient Real completeness.*

Proof. Project the completion carrier and apply Definition 1411.19 and Definition 1411.20. These force the finite criterion, completion dependency, and regular-name equivalence rows to precede the classifier export.

The remaining rows H, C, P, N only transport, replay, source, and name that same route. Thus the listed rows exhaust regular-name uniqueness before separation, and the excluded principles have no carrier row from which a consumer could read them. □

Definition 1411.22 (SeparatedMetric zero-distance scope row)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, the zero-distance scope row is the finite route that begins with a* $\mathsf{Met}^{\uparrow}$ *distance witness in* X*, reads the witness as a* $\mathsf{Real}^{\uparrow}$ *zero-distance row through* D*, transports the evidence through* H*, replays it through* C*, sources it through* P*, names it through* N*, and only then exports the classifier equivalence supplied by* Z*. If the zero-distance read is attached to completion-limit data, the route first passes through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row* E *that binds* L_0 *and* L_1 *to the same regular Cauchy point name.*

The row is a scope obligation for displayed evidence. It is not host point equality, quotient real equality, a selected representative, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1411.23 (SeparatedMetric zero-distance reflection)**.** *Every classifier equivalence exported by a zero-distance read from an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *reflects the route of Definition 1411.22. In particular, the export must pass through a* $\mathsf{Met}^{\uparrow}$ *distance witness, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row* D*, the componentwise* hsame *transport* H*, the displayed* Cont *replay* C*, the* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N *before the classifier row* Z *is available.*

Proof. Project the carrier of Definition 1411.1. The metric source is X, the real-valued distance interface is D, and the classifier export is stored only in the separation row Z.

Apply the zero-distance scope row of Definition 1411.22. It places the distance witness before the $\mathsf{Real}^{\uparrow}$ zero-distance read and places the support rows H, C, P, N before the classifier export.

Use Definition 1411.15 to transport and replay the same route. The resulting classifier equivalence is therefore a reflected BEDC row, not host equality, quotient equality, selected representative data, trichotomy, or a completeness principle. □

Theorem 1411.24 (SeparatedMetric completion-limit uniqueness scope)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Any completion-limit uniqueness claim exported from* S *must read* L_0 *and* L_1 *through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row* E *before it can form the* $\mathsf{Met}^{\uparrow}$ *distance witness, read* $\mathsf{Real}^{\uparrow}$ *zero-distance through* D*, transport through* H*, replay through* C*, source through* P*, name through* N*, and export classifier equivalence through* Z*.*

Consequently completion-limit uniqueness is scoped as classifier equivalence in the displayed metric carrier. It is not host limit equality, quotient real equality, quotient completion equality, a selected representative, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1411.18. It requires the two complete-metric limit rows to pass through E before the separation row can be read.

Next apply Definition 1411.16. The ledger constructs the metric-distance witness only after the finite-window equivalence route has tied the two limit rows to the same regular Cauchy point name.

Finally use Definition 1411.23. The zero-distance evidence travels through D, H, C, P, N before Z exports classifier equivalence. Since no host equality, quotient equality, selected representative, trichotomy, or completeness field occurs in S, the uniqueness claim remains inside the displayed classifier scope. □

Theorem 1411.25 (SeparatedMetric real zero-distance nonescape)**.** *For every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, a* $\mathsf{Real}^{\uparrow}$ *zero-distance read can be consumed only as part*

of the finite route through $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *when completion-limit rows are present, componentwise* hsame*, displayed* Cont, Pkg*, and* $\mathsf{NameCert}$*. The* $\mathsf{Real}^{\uparrow}$ *row supplies the zero-distance witness used by* Z*; it does not export host real equality, quotient real equality, trichotomy, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the zero-distance scope row from Definition 1411.22. It places the $\mathsf{Real}^{\uparrow}$ row after the metric-distance witness and before the separation classifier.

If completion-limit rows are involved, use Definition 1411.24. The route first passes through E, then the distance witness, then D, and only afterward the transport, replay, provenance, naming, and classifier rows.

The nonescape statement is the field inventory of S. The carrier contains no host real equality row, quotient real equality row, trichotomy ledger, selected representative coordinate, or ambient completeness theorem, so the zero-distance read cannot export any of them. □

Theorem 1411.26 (SeparatedMetric consumer nonescape scope)**.** *Every accepted consumer of a* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet receives only the rows*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Consequently the consumer may inspect metric distance evidence, two complete-limit rows, their finite-window equivalence route, the real zero-distance seal, componentwise transport, replay, provenance, and local naming, but no quotient carrier, host equality, trichotomy row, selected representative, open-cover compactness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the carrier inventory in Definition 1411.1. Its visible fields are the metric carrier row X, the real-valued distance row D, the separation row Z, the two complete-limit rows L_0, L_1, the equivalence route E, and the support rows H, C, P, N.

The consumer boundary theorem, Definition 1411.5, gives the permitted order of access. Consumers read metric data, complete-limit data, finite-window equivalence, real zero-distance evidence, and separation classification through the displayed support rows.

The nonescape conclusion is then a field-exhaustion claim. Since the carrier has no quotient object, host equality predicate, trichotomy ledger, selected representative, open-cover compactness coordinate, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, no accepted consumer route can recover any of those objects from N. □

Theorem 1411.27 (SeparatedMetric zero-distance scope)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier as in Definition 1411.1. Every zero-distance transport accepted by the local* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *row is a displayed* BHist *route*

$$\mathsf{MetricDistanceWitness}(a, b, d) \longmapsto D(d, 0) \longmapsto Z(a, b)$$

carried by X, D, Z, H, C, P, N*. Transport by* hsame *and replay by* Cont *preserve this route before the point classifier is read, so the downstream zero-distance consumer receives classifier equivalence and not host equality, quotient equality, trichotomy, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the displayed packet from Definition 1411.1. The rows available for zero-distance transport are the metric carrier X, the real-valued distance interface D, the separation row Z, and the support rows H, C, P, N.

Use the zero-distance carrier row of Definition 1411.14. It fixes the order of the route: metric-distance witness first, real zero-distance reading second, and separation classification third.

Now apply the transported classifier stability row of Definition 1411.15. The hsame fields move the displayed histories componentwise and the Cont field replays the same route after transport.

Since the packet has no host equality coordinate, quotient carrier, trichotomy ledger, or completeness row, the transported conclusion remains exactly the classifier route named by N. □

Theorem 1411.28 (SeparatedMetric limit route scope)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. A completion-limit uniqueness read from S can inspect the two* $\mathsf{CompleteMetric}^{\uparrow}$ *rows L_0, L_1 only after the* BHist *comparison route E has displayed that both rows belong to the same regular Cauchy point name. The accepted downstream route is therefore*

$$L_0, L_1 \longmapsto E \longmapsto \mathsf{MetricDistanceWitness}(\ell_0, \ell_1, d) \longmapsto D(d, 0) \longmapsto Z(\ell_0, \ell_1),$$

with transport and replay supplied only by H and C and with provenance and local naming supplied only by P and N.

Proof. Read L_0, L_1, E from the carrier of Definition 1411.1. The comparison route E is the only displayed row tying the two complete-metric limit claims to one regular Cauchy point name.

Apply Definition 1411.18. It places the equivalence route before the real zero-distance row and before the separation row, ruling out representative comparison or quotient-class comparison as an alternate path.

Then use Definition 1411.16. The ledger constructs the metric-distance witness between the displayed limit histories after E has fixed the common finite-window comparison surface.

The final zero-distance and classifier reads go through D and Z. Rows H, C, P, N only preserve and name that same route, so the completion-limit consumer receives uniqueness as classifier equivalence inside X, not as host equality or selected completion data. □

Theorem 1411.29 (SeparatedMetric classifier ledger scope)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$, the classifier ledger exported by* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *is exhausted by*

$\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Every accepted consumer must read metric evidence, real zero-distance evidence, complete-limit rows, the finite-window equivalence route, componentwise transport, continuation replay, provenance, and local naming through these rows. No ledger entry permits escape to quotient metric spaces, host equality of histories, selected representatives, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier inventory of Definition 1411.1. It lists exactly the metric row X, real-valued distance row D, separation row Z, complete-limit rows L_0, L_1, equivalence route E, and structural rows H, C, P, N.

The NameCert obligations of Definition 1411.3 then partition these rows into carrier admission, classifier stability, completion compatibility, zero-distance exactness, and ledger visibility. Each obligation is anchored in the displayed BHist packet.

Use the consumer nonescape theorem Definition 1411.26. It states that downstream access remains inside the listed metric, completion, equivalence, real, transport, replay, provenance, and naming rows.

Thus the classifier ledger has no unlisted escape row. Quotient metric spaces, host equality, selected representatives, trichotomy, open-cover compactness, and $\mathsf{Real}^{\uparrow}$ completeness are not fields of S and cannot be read from its local naming row. □

Theorem 1411.30 (SeparatedMetric zero-distance NameCert readiness)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, the zero-distance obligation exported to* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *is exactly the finite* BHist *route through*

$$\mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

It reads metric evidence from X*, the real zero-distance seal from* D*, classifier separation from* Z*, and transport, replay, provenance, and local naming from* H, C, P, N*; it does not expose host equality, quotient equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1411.1. The visible zero-distance rows are the metric carrier X, the real-valued distance row D, and the separation row Z.

Use the zero-distance carrier row of Definition 1411.14 and the transported classifier stability row of Definition 1411.15. They force every zero-distance read through the displayed metric witness, the $\mathsf{Real}^{\uparrow}$ zero-distance reading, and the classifier row, with H and C preserving the same route.

The provenance and local naming rows P, N then package the route as a NameCert obligation. Since the carrier contains no host equality or quotient coordinate, the readiness claim remains a finite classifier route. □

Theorem 1411.31 (SeparatedMetric completion-limit equivalence handoff)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P,$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every completion-limit comparison exported by* S *must first pass through*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Thus the two completion limit rows L_0, L_1 *are handed to the separation row only after the finite equivalence route* E *identifies the common regular Cauchy point name.*

Proof. Read the two complete-limit rows and the equivalence route from Definition 1411.1. The packet stores L_0, L_1, E as finite rows, not as selected completion representatives.

Apply Definition 1411.16 and Definition 1411.28. Together they place E before the metric-distance witness, the real zero-distance row, and the separation classifier.

Finally use Definition 1411.29. The support rows H, C, P, N preserve and name the same handoff, and no quotient completion, host equality, countable choice, or ambient completeness row is available. □

Theorem 1411.32 (SeparatedMetric Real-seal boundary obligations)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier, every* $\mathsf{Real}^{\uparrow}$*-facing boundary read in a zero-distance or completion-limit obligation is exhausted by the rows*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The $\mathsf{Real}^{\uparrow}$ *row supplies only the displayed zero-distance seal used by the classifier; it does not supply real completeness, quotient equality of completion names, trichotomy, selected representatives, or open-cover compactness.*

Proof. Start from Definition 1411.30. It identifies the real-facing zero-distance route as a finite metric witness followed by the $\mathsf{Real}^{\uparrow}$ zero seal and the separation classifier.

Then use Definition 1411.31. Completion-limit comparisons reach the real seal only after the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row has tied both complete-metric limit rows to the same regular Cauchy name.

The boundary conclusion follows from Definition 1411.26. All accepted reads are carried by BHist, transported by hsame, replayed by Cont, and named through Pkg and NameCert, with no field for a stronger host-side principle. □

Theorem 1411.33 (SeparatedMetric kernel zero-distance obligation)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, every zero-distance classifier obligation exported by the kernel factors through the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{hsame} \longmapsto \mathsf{Cont} \longmapsto \mathsf{Pkg} \longmapsto \mathsf{NameCert}.$$

The $\mathsf{Met}^{\uparrow}$ *distance witness is read before the* $\mathsf{Real}^{\uparrow}$ *zero row, and only after transport, replay, provenance, and local naming may the classifier row* Z *expose separation equivalence. The obligation exports no host equality, quotient equality, selected representative, trichotomy row, or ambient completeness principle.*

Proof. Read the zero-distance carrier row of Definition 1411.14. It fixes the route from a displayed metric-distance witness to the $\mathsf{Real}^{\uparrow}$ zero row and then to the separation classifier.

Next apply the transported classifier stability row of Definition 1411.15. The hsame and Cont rows preserve the same displayed route after transport, while P and N source and name it as a local certificate obligation.

The nonescape clause is immediate from the carrier inventory in Definition 1411.1. None of the excluded host or quotient principles is a coordinate of S. □

Theorem 1411.34 (SeparatedMetric completion-equivalence obligation)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Any completion-equivalence obligation that compares the two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1 *must first pass through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E *before the separation row can read a metric-distance witness, the* $\mathsf{Real}^{\uparrow}$ *zero seal, transport, replay, provenance, and local naming.*

Thus the separated completion consumer receives only classifier equivalence inside the displayed metric carrier. It does not receive quotient completion equality, selected representatives, host limit equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1411.18. It places the finite-window equivalence route E before any zero-distance or separation read involving L_0 and L_1.

Then use Definition 1411.28. After E has tied the complete-metric rows to one regular Cauchy point name, the route may construct the displayed metric-distance witness and read the $\mathsf{Real}^{\uparrow}$ zero seal through D.

Finally apply Definition 1411.29. The rows H, C, P, N transport, replay, source, and name the same route, so no quotient completion object or stronger host-side equality is exported. □

Theorem 1411.35 (SeparatedMetric Real-seal routing obligation)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *row in every zero-distance or completion-equivalence route is only a seal inside the finite certificate route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

It certifies the displayed zero-distance read consumed by Z*; it does not supply* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, quotient equality of completion names, trichotomy, or selected representative data.*

Proof. Use Definition 1411.32. The real-facing row is reached only after the relevant metric or completion-equivalence rows have been displayed.

For a pure zero-distance route, Definition 1411.33 places the $\mathsf{Real}^{\uparrow}$ row between the metric-distance witness and the classifier. For a completion route, Definition 1411.34 first inserts the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row between the complete-metric limits and the zero-distance read.

In both cases the support rows H, C, P, N preserve and name the same finite route. Since no Real completeness, quotient equality, trichotomy, or representative-selection coordinate is present, the Real row remains a seal and not an escape principle. □

Theorem 1411.36 (SeparatedMetric root zero-distance dependency)**.** *Every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer that exposes the separated classifier must first pass through the root* $\mathsf{Met}^{\uparrow}$ *distance dependency. For a carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, the row* X *supplies the displayed metric source and distance witness,* D *seals that witness as a* $\mathsf{Real}^{\uparrow}$ *zero-distance row, and only then may* Z *expose the separated classifier through* H, C, P, N*.*

The consumer receives no classifier route from host equality, quotient metric equality, selected representatives, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier tuple. The only coordinate naming the metric source and distance witness is X, so every classifier read begins there.

Apply Definition 1411.33. It places the $\mathsf{Met}^{\uparrow}$ distance row before the $\mathsf{Real}^{\uparrow}$ zero seal and the separated classifier.

Finally use the support rows H, C, P, N to transport, replay, source, and name the same route. Since the carrier has no extra host-equality or quotient coordinate, the exposed classifier is dependent on the displayed zero-distance route. □

Theorem 1411.37 (SeparatedMetric root completion-equivalence dependency)**.** *Every accepted completion-facing* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer must pass through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E *and the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1 *before exposing the separated classifier. The completion-equivalence dependency is therefore the finite route*

$$L_0, L_1, E, X, D, Z, H, C, P, N$$

inside the accepted carrier, not a quotient completion equality or a selected host limit.

Proof. Begin with the completion rows L_0, L_1. They are the only packet coordinates that can serve a completion-facing consumer.

Use Definition 1411.34. It forces any comparison of L_0 and L_1 through the finite-window route E before the metric-distance and zero-seal rows may be read.

Then apply Definition 1411.36. The separated classifier is exposed only after X, D, Z and the support rows have been consumed, so the completion-equivalence dependency cannot escape into quotient completion equality, countable choice, or selected limits. □

Theorem 1411.38 (SeparatedMetric root Real-seal dependency)**.** *Every accepted Real-facing* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer must pass through the root* $\mathsf{Real}^{\uparrow}$ *zero-seal row* D *before it exposes the separated classifier. In the zero-distance branch this dependency is* X, D, Z, H, C, P, N*; in the completion branch it is* $L_0, L_1, E, X, D, Z, H, C, P, N$*.*

Thus the Real row is a dependency of the separated classifier, not an independent source of Real completeness, quotient equality of real names, quotient equality of completion names, trichotomy, or representative selection.

Proof. Use Definition 1411.35. It identifies the $\mathsf{Real}^{\uparrow}$ row as a zero-distance seal inside each accepted zero-distance or completion-equivalence route.

For the zero-distance branch, apply Definition 1411.36. For the completion branch, apply Definition 1411.37.

Both branches reach the same classifier row Z only after D has been read and after H, C, P, N have transported and named the displayed route. The carrier contains no independent Real-completeness or quotient-equality coordinate, so the Real seal remains a dependency boundary. □

Theorem 1411.39 (SeparatedMetric root classifier transport obligation)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. The root classifier transport obligation has exactly the* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface displayed by H, C, P, N: any transported classifier read must first expose the carrier row of Definition 1411.1, the zero-distance dependency of Definition 1411.36, the completion-equivalence dependency of Definition 1411.37, and the Real-seal dependency of Definition 1411.38.*

Consequently the classifier cannot be transported by host equality, quotient metric equality, selected completion representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The classifier coordinate Z is not a free equality row; it is downstream of the metric source X, the zero-distance row D, the completion route E when completion rows are read, and the support rows H, C, P, N.

Use Definition 1411.36 for the pure metric branch and Definition 1411.37 for the completion branch. Both branches pass through the same zero-distance classifier route before the separated classifier is exposed.

Finally apply Definition 1411.38. Since the $\mathsf{Real}^{\uparrow}$ row remains a seal inside the displayed route, transport through hsame and replay through Cont can rename only the finite BHist rows already packaged by P, N. □

Theorem 1411.40 (SeparatedMetric root completion readback obligation)**.** *For every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$, a completion-facing readback of the separated classifier must pass through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *rows L_0, L_1, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *dependency E, the zero-distance dependency row, and the* $\mathsf{Real}^{\uparrow}$ *seal before H, C, P, N name it as a* BHist *and* NameCert *obligation.*

The readback does not export quotient completion equality, a selected limit representative, host limit equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and classifier inventory in Definition 1411.1. The only completion rows available to a consumer are L_0, L_1 and the displayed equivalence route E.

Apply Definition 1411.37. It forces completion-facing comparison through E before the zero-distance and classifier rows can be read.

Then use Definition 1411.39. The readback is transported only by H and replayed only by C, while P, N package the same route as a local NameCert row, so no completion quotient or selected representative is exposed. □

Theorem 1411.41 (SeparatedMetric root Real-seal nonescape obligation)**.** *Every Real-facing root obligation of an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier is exhausted by the finite route through the carrier, the zero-distance dependency row, the completion-equivalence dependency row when present, the* $\mathsf{Real}^{\uparrow}$ *seal, and the support surface* BHist, hsame, Cont, Pkg, NameCert*. The Real seal is therefore a boundary row for the separated classifier, not a source of quotient real equality, quotient completion equality, trichotomy, selected representatives, or ambient real completeness.*

Proof. Use Definition 1411.38. It places the Real row after the metric-distance witness in the zero-distance branch and after the completion-equivalence handoff in the completion branch.

For the zero-distance route, apply Definition 1411.36. For the completion route, apply Definition 1411.40. These two rows cover the only classifier-facing branches present in Definition 1411.1.

The rows H, C, P, N then transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for the excluded host principles, the Real seal cannot escape its displayed obligation boundary. □

Theorem 1411.42 (SeparatedMetric public identity export)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every public identity consumer that asks when two displayed points of X are identified must read the metric-distance witness through D, the* $\mathsf{Real}^{\uparrow}$ *zero seal, and the separation classifier Z before exposing an identity claim. The exported claim is exactly the zero-distance route of Definition 1411.33; it carries no host equality, quotient equality, selected representative, trichotomy row, or ambient completeness principle.*

Proof. Project the carrier S. The only identity-facing rows are the metric row X, the distance and zero-seal route D, the classifier Z, and the structural rows H, C, P, N.

Apply Definition 1411.33. It places the metric-distance witness before the $\mathsf{Real}^{\uparrow}$ zero row and places both before the separation classifier.

Finally use Definition 1411.26. The support rows transport, replay, source, and name the same route, so the public identity export cannot detach into host equality or a quotient principle. □

Theorem 1411.43 (SeparatedMetric public completion uniqueness export)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every public completion-uniqueness consumer comparing the completion rows L_0 and L_1 must first read the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route E and then the zero-distance classifier route through D, Z, H, C, P, N. The export is the completion-equivalence boundary of Definition 1411.34; it supplies no quotient completion equality, selected limit, host limit equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the completion rows L_0 and L_1 in the carrier. They are not public equality rows by themselves; they are complete-metric limit rows tied to the displayed route E.

Apply Definition 1411.34. It forces any comparison of L_0 and L_1 through E before the route may read the zero-distance witness and separation classifier.

Then use Definition 1411.31 and Definition 1411.28. These keep the completion comparison inside the finite Cauchy-equivalence and zero-distance handoff.

The rows H, C, P, N provide only transport, replay, provenance, and local naming. Therefore the public uniqueness export is exhausted by the displayed separated-metric route. □

Theorem 1411.44 (SeparatedMetric public Real-seal nonescape export)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every public* $\mathsf{Real}^{\uparrow}$*-seal consumer of a zero-distance or completion-uniqueness route receives the seal only inside*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The exported Real-facing boundary is the nonescape route of Definition 1411.35; it does not provide Real completeness, quotient equality of real names, quotient equality of completion names, trichotomy, or representative selection.

Proof. Use Definition 1411.35. It states that the $\mathsf{Real}^{\uparrow}$ row is only the displayed zero-distance seal inside the finite certificate route.

For identity consumers, apply Definition 1411.42. For completion-uniqueness consumers, apply Definition 1411.43. These two exports account for the public zero-distance and completion-facing routes.

Since the same support rows H, C, P, N preserve both routes, no public consumer can reinterpret the Real seal as a stronger host-side principle. □

Theorem 1411.45 (SeparatedMetric zero-distance separation obligation)**.** *For every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, any bridge-facing zero-distance separation read must pass through the* $\mathsf{Met}^{\uparrow}$ *carrier row* X*, the* $\mathsf{Real}^{\uparrow}$*-valued distance witness row* D*, and the separation classifier row* Z *before the structural rows* H, C, P, N *can expose it to a public consumer. The route is a finite projection from the displayed carrier; it is not host equality, quotient equality, trichotomy, representative selection, or ambient completeness.*

Proof. Project S through Definition 1411.1. The only rows that can answer a zero-distance separation request are X, D, Z and the structural rows attached to them.

Apply Definition 1411.33. It places the $\mathsf{Met}^{\uparrow}$ distance witness before the $\mathsf{Real}^{\uparrow}$ zero-distance reading and places both before the separation classifier.

The bridge-facing consumer then receives the transported and replayed route through H, C, P, N. Since no excluded host-side principle is a carrier row, the bridge surface remains a finite projection. □

Theorem 1411.46 (SeparatedMetric completion uniqueness obligation)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Any bridge-facing completion-uniqueness read comparing the two* $\mathsf{CompleteMetric}^{\uparrow}$ *rows* L_0, L_1 *must first use the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E*, then the zero-distance route through* X, D, Z*, and finally the structural rows* H, C, P, N*.*

Proof. Start with the public completion uniqueness export of Definition 1411.43. It fixes E as the required route from the two complete-metric limit rows to a common finite-window comparison.

Use Definition 1411.34. After E is displayed, the comparison may read the metric-distance witness, the $\mathsf{Real}^{\uparrow}$ zero seal, and the separation row.

The rows H, C, P, N transport, replay, source, and name this same finite chain. Hence the bridge-facing uniqueness obligation exports no quotient completion equality, selected limit, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1411.47 (SeparatedMetric Real-seal nonescape obligation)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, every bridge-facing* $\mathsf{Real}^{\uparrow}$ *seal read is confined to the displayed zero-distance or completion-uniqueness route through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CauchySequenceEquivalence}$ $\mathsf{Real}^{\uparrow}$*,* H, C, P, N*. It does not export* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, quotient equality of completion names, trichotomy, or representative selection.*

Proof. Apply the public Real-seal nonescape export of Definition 1411.44. It already identifies the Real row as a zero-distance seal inside the finite separated-metric route.

For the identity branch, use Definition 1411.45. For the completion branch, use Definition 1411.46. These two obligations exhaust the bridge-facing ways to reach the seal.

The carrier contains no Real-completeness, quotient-equality, trichotomy, or representative-selection coordinate. Therefore the seal remains a nonescape boundary for both bridge-facing routes. □

Theorem 1411.48 (SeparatedMetric bridged-closure readiness)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Its bridge-facing public surface consists exactly of the* $\mathsf{Met}^{\uparrow}$

distance witness, the $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route, the* $\mathsf{Real}^{\uparrow}$ *zero seal, the separation classifier row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. This is the complete surface exported to a bridge consumer.*

The surface exports no quotient completion equality, selected limit, host equality, trichotomy, representative-selection row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public identity export of Definition 1411.42. It identifies the identity-facing part of the surface as the metric-distance witness, the $\mathsf{Real}^{\uparrow}$ zero seal, the separation classifier, and the support rows H, C, P, N.

Next apply the public completion uniqueness export of Definition 1411.43. It adds the two $\mathsf{CompleteMetric}^{\uparrow}$ limit rows and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ route, but it does not add a quotient completion equality, selected limit, or host equality row.

Finally use the public Real-seal nonescape export of Definition 1411.44. The Real row remains a zero-distance seal inside the displayed finite certificate route, and the support rows provide only transport, replay, provenance, and local naming. Therefore the listed rows are exactly the bridge-facing readiness surface. □

Theorem 1411.49 (SeparatedMetric separated-identity transport obligation)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every separated-identity transport read of* S *is a finite* BHist *packet-coordinate route: the* $\mathsf{Met}^{\uparrow}$ *distance rows of* X *expose a distance witness, the* $\mathsf{Real}^{\uparrow}$ *seal row* D *reads that witness as zero,* hsame *rows in* H *transport the endpoints and witness, the displayed* Cont *replay* C *preserves the route, and the* NameCert *row* N *names the transported classifier row* Z*.*

No such read may bypass X, D, H, C, P, N *by parameter transport, host equality, quotient equality, trichotomy, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the packet coordinates from Definition 1411.1. The identity-facing data are the $\mathsf{Met}^{\uparrow}$ distance rows in X, the $\mathsf{Real}^{\uparrow}$ zero-distance seal D, the separation row Z, and the structural rows H, C, P, N.

Use Definition 1411.45. It forces every zero-distance identity read to place the metric-distance witness before the real zero seal and both before the separation classifier.

Transport is then componentwise through H and replayed through C. The provenance and local naming rows P, N expose exactly the transported finite route, so the conclusion is classifier equivalence through Z rather than a host-side identity principle. □

Theorem 1411.50 (SeparatedMetric completion-uniqueness consumer obligation)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every completion-uniqueness consumer of* S *must read the two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1 *through the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *finite-window route* E*, then through the* $\mathsf{Met}^{\uparrow}$ *distance rows of* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal* D*, the* hsame *transport rows* H*, the displayed* Cont *replay* C*, the* Pkg *provenance* P*, and the local* NameCert *row* N *before it may consume the classifier row* Z*.*

The obligation exports completion-limit uniqueness only as displayed classifier equivalence, not as quotient completion equality, selected host limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the completion rows L_0, L_1. By Definition 1411.46, a consumer cannot compare them until the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row E displays the shared regular Cauchy finite windows.

After E is displayed, apply the zero-distance route of Definition 1411.45. The comparison must form a metric-distance witness in X, read the $\mathsf{Real}^{\uparrow}$ zero seal D, and then reach the separation row Z.

Finally transport, replay, provenance, and naming pass through H, C, P, N. Since no quotient completion object, selected limit, countable-choice schedule, host equality row, or ambient completeness theorem is a packet coordinate, the consumer obligation is exhausted by the displayed rows. □

Theorem 1411.51 (SeparatedMetric Cauchy-equivalence zero-distance induction)**.** *Let $S = (X, D, Z, L_0, L_1, E, H$ be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every completion-limit uniqueness consumer that tries to identify the two limit rows L_0, L_1 must proceed by the following induction-style route: first display the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *finite-window route E, then build the* $\mathsf{Met}^{\uparrow}$ *distance witness in X, then read the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal D, then transport through H, replay through C, record* Pkg *provenance through P, name the route through N, and only then consume the separated classifier Z.*

Consequently completion-limit uniqueness is exported as a separated classifier consequence of the displayed zero-distance route. It is not quotient completion equality, selected representative equality, countable-choice selection, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the existing completion uniqueness route of Definition 1411.50. It requires L_0, L_1 to pass through the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ finite-window row E before any comparison reaches a metric witness.

Next apply the zero-distance carrier row of Definition 1411.45. The displayed finite-window comparison supplies the $\mathsf{Met}^{\uparrow}$ distance witness in X, and that witness is read through the $\mathsf{Real}^{\uparrow}$ zero-distance seal D before the classifier row Z is available.

Now use the classifier stability and zero-distance transport rows of Definition 1411.15 and Definition 1411.17. They keep the route componentwise through H and replay it through C while preserving the same separated classifier conclusion.

Finally apply the completion handoff nonescape theorem Definition 1411.9. The provenance and local naming rows P, N expose exactly the routed finite evidence, so no quotient completion equality, selected representative, countable-choice schedule, host equality, trichotomy, compactness, or ambient completeness principle can replace the induction route. □

Theorem 1411.52 (SeparatedMetric L10 completion unblock package)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$, the L10 package offered to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *fronts is the ordered product of the completion-limit rows L_0, L_1, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route E, the* $\mathsf{Met}^{\uparrow}$ *distance witness in X, the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal D, the separated classifier Z, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N.*

Any downstream completion consumer that claims uniqueness outside this product leaves the $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier surface. The package therefore unblocks completion uniqueness only as a finite zero-distance classifier route, not as ambient completion equality or Real completeness.*

Proof. Use Definition 1411.51. It gives the induction-style order in which $L_0, L_1, E, X, D, H, C, P, N$ must be displayed before the separated classifier row Z is consumed.

The zero-distance carrier row and classifier stability row provide the public classifier consequence. The zero-distance transport theorem then shows that the same consequence survives componentwise hsame transport and displayed replay.

It remains to expose the package to completion fronts. Apply Definition 1411.43 and Definition 1411.44. They route $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers through the displayed completion-limit, Cauchy-equivalence, metric-distance, Real-seal, transport, provenance, and local-name rows, while refusing quotient equality, selected limits, countable choice, host equality, trichotomy, compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1411.53 (SeparatedMetric Real-seal zero-distance budget)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal row consumed by a downstream Real-facing or completion-facing read is budgeted to one of two finite routes: a separated-identity route through the* $\mathsf{Met}^{\uparrow}$ *distance rows of* X*,* D*,* Z*,* H, C, P, N*, or a completion-uniqueness route through* $L_0, L_1, E, X, D, Z, H, C, P, N$ *with* E *displaying the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *finite windows.*

The budget contains no third route for Real equality, quotient equality, selected representatives, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1411.47. It confines every bridge-facing $\mathsf{Real}^{\uparrow}$ seal read to the separated-metric zero-distance or completion-uniqueness route.

For the identity branch, apply Definition 1411.49. The seal is read after the metric-distance witness and before the transported classifier row is named.

For the completion branch, apply Definition 1411.50. The seal is read only after the two $\mathsf{CompleteMetric}^{\uparrow}$ rows and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ finite-window route have been displayed.

The carrier inventory has no coordinate for the excluded third routes. Therefore the Real-seal budget is exactly the two displayed finite routes. □

Theorem 1411.54 (SeparatedMetric Real zero-distance scope)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, every Real-facing zero-distance classifier read must pass through the displayed* $\mathsf{Met}^{\uparrow}$ *distance witness in* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal* D*, the optional* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E *when completion-limit rows* L_0, L_1 *are present, component-wise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N *before the separation classifier row* Z *may be exported.*

The export is therefore a local classifier consequence of the displayed zero-distance route. It is not host equality, quotient equality, a selected representative, trichotomy of real comparisons, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier through Definition 1411.1. The zero-distance branch exposes X, D, Z, H, C, P, N, so a Real-facing identity read first forms the $\mathsf{Met}^{\uparrow}$ distance witness in X and only then reads that witness through the $\mathsf{Real}^{\uparrow}$ seal D.

If completion-limit rows are present, apply Definition 1411.50. The read must pass through L_0, L_1 and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row E before it reaches the same metric witness and real zero-distance seal.

Now use the public identity and completion exports, Definition 1411.42 and Definition 1411.43. They place the separated classifier behind the displayed zero-distance and completion-equivalence routes, while Definition 1411.44 keeps the $\mathsf{Real}^{\uparrow}$ row inside that route.

Finally transport, replay, provenance, and naming pass through H, C, P, N. Since the carrier has no host-equality, quotient-equality, selected-representative, trichotomy, or ambient-completeness coordinate, the classifier export cannot escape the displayed Real zero-distance scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeparatedMetric}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a $\mathsf{Real}^{\uparrow}$-valued distance witness, a zero-distance separation row, two $\mathsf{CompleteMetric}^{\uparrow}$ Cauchy-limit rows for one regular Cauchy point name, a

CauchySequenceEquivalence$^{\uparrow}$ route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SeparatedMetric}\uparrow}$, maturePackageClosed)

Scope closed: Definition 1411.48 together with Definition 1411.42, Definition 1411.43, and Definition 1411.44 identifies the mature separated-metric public surface over Met$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchySequenceEquivalence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{SeparatedMetric}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient metric spaces, host equality, trichotomy, open-cover compactness, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No paper-side closure upgrade remains; Lean verification and bridge checking remain separate formal-axis work.

Theorem 1411.55 (SeparatedMetric carrier boundary)**.** *Every accepted carrier read of a* SeparatedMetric$^{\uparrow}$ *packet factors through the displayed* Met$^{\uparrow}$ *carrier row, the* Real$^{\uparrow}$ *zero-distance witness row, the* CompleteMetric$^{\uparrow}$ *limit rows, the* CauchySequenceEquivalence$^{\uparrow}$ *comparison row, and the structural rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. No consumer may read a separated classifier by omitting any of these carrier-boundary rows when the requested read is a completion-limit uniqueness read.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricCarrier boundary`

Proof. Project the carrier of Definition 1411.1. Its coordinates list the metric source X, real-valued distance witness D, complete-metric limit rows L_0, L_1, comparison route E, transport H, replay C, provenance P, and local naming row N.

For identity-facing reads, the metric and real rows expose the distance witness before the separation row is consumed. For completion-facing reads, the two complete-metric rows and the Cauchy-sequence equivalence route are projected before the same zero-distance route is available.

The support rows are not optional. They transport, replay, source, and name the finite path, so every accepted carrier read factors through the listed BEDC rows rather than through host equality or an off-packet completion principle. □

Theorem 1411.56 (SeparatedMetric symmetry transport)**.** *For an accepted* SeparatedMetric$^{\uparrow}$ *carrier, zero-distance classifier evidence is stable under componentwise* hsame *transport and displayed* Cont *replay. If the endpoints, distance witness, zero-distance reading, completion rows, comparison row, provenance row, and local name are transported componentwise, the replayed route exports the transported separated classifier evidence and no additional equality principle.*

Proof. Use the transport row H of Definition 1411.1. It moves endpoints, distance witnesses, zero-distance readings, completion rows, the comparison route, provenance, and the local naming row as components of one finite packet.

Apply the replay row C after transport. The replay forms the same metric-distance witness, reads it through the same Real$^{\uparrow}$ zero-distance interface, and then consumes the transported separation row.

Thus symmetry of the classifier evidence is a transported replay of the displayed route. It is not a conversion to host equality, quotient equality, selected representatives, trichotomy, choice, or ambient Real$^{\uparrow}$ completeness. □

Theorem 1411.57 (SeparatedMetric zero-distance refusal)**.** *No accepted* SeparatedMetric$^{\uparrow}$ *consumer may replace the separated zero-distance classifier route by host equality, quotient equality, trichotomy, choice, selected representatives, or ambient* Real$^{\uparrow}$ *completeness. The only exported route from zero distance to identification is the finite route through* Met$^{\uparrow}$*,* Real$^{\uparrow}$*,* CompleteMetric$^{\uparrow}$

when completion rows are present, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Start from the NameCert obligations of Definition 1411.3. They make carrier admission, classifier stability, completion compatibility, zero-distance exactness, and ledger visibility the complete accepted surface.

The zero-distance exactness row sends a metric-distance witness through the $\mathsf{Real}^{\uparrow}$ zero-distance reading before the separation row may be exported. Completion consumers must additionally display the two $\mathsf{CompleteMetric}^{\uparrow}$ rows and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ comparison route.

The ledger visibility clause excludes each proposed replacement because none is a coordinate of the packet. Therefore the refused principles have no row to transport through H, replay through C, source through P, or name through N. □

Definition 1411.58 (SeparatedMetric zero-distance carrier)**.** *A* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance carrier is a finite* BHist *packet*

$$\mathsf{SepZero}(X, D, Z, E, L, H, C, P, N)$$

whose coordinates are exactly:

1. X*, the displayed* $\mathsf{Met}^{\uparrow}$ *source row containing the carried endpoints and distance witness;*
2. D*, the* $\mathsf{Real}^{\uparrow}$ *distance witness row reading that distance as zero;*
3. Z*, the zero-distance refusal row that permits classifier identification only through the displayed zero-distance route;*
4. E*, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *handoff row used when the endpoints are completion names;*
5. L*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *compatibility row binding completion-limit consumers to the same metric source;*
6. H*, the componentwise* hsame *transport ledger for endpoints, distance witnesses, zero rows, and completion rows;*
7. C*, the displayed* Cont *replay ledger for the same route;*
8. P*, the* Pkg *provenance row naming the source of the carrier;*
9. N*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *row.*

The classifier of this carrier compares packets only by these rows. It has no coordinate for host equality, quotient metric equality, selected representatives, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1411.59 (SeparatedMetric NameCert obligation surface)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance carrier*

$$S = \mathsf{SepZero}(X, D, Z, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *surface is exactly the finite obligation surface consisting of the* $\mathsf{Met}^{\uparrow}$ *source row* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance witness* D*, the zero-distance refusal row* Z*, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *handoff* E*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *compatibility row* L*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local naming row* N*.*

Every accepted identity or completion-uniqueness read must consume these rows in this order before it may export a point-classifier identification. The surface refuses host equality, quotient equality, selected representatives, trichotomy, open-cover compactness, and ambient completeness data because no such row occurs in S*.*

Proof. Project the coordinates of Definition 1411.58. The first three rows supply the metric source, the real-valued zero-distance witness, and the refusal that prevents classifier identification unless the zero-distance route has been displayed.

For completion-facing reads, use the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ rows before the separation classifier is consumed. These rows bind completion names to the displayed metric source rather than to a quotient completion object.

Finally transport, replay, provenance, and naming pass through H, C, P, N. Since the carrier lists no additional coordinate, every accepted $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ obligation is exhausted by the stated finite surface. □

Theorem 1411.60 (SeparatedMetric Cauchy-completion readback)**.** *Let $S = \mathsf{SepZero}(X, D, Z, E, L, H, C, P, N)$ be an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance carrier. Any Cauchy or completion consumer that reads a zero-distance collapse from S must route the read through the $\mathsf{SeparatedMetric}^{\uparrow}$ packet together with the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ rows.*

The consumer may not read zero-distance collapse from a host Cauchy quotient, an ambient equality of completion endpoints, a selected limit, countable choice, trichotomy, or an untracked completeness theorem. The only exported conclusion is the classifier route named by N after the displayed rows X, D, Z, E, L, H, C, P have been consumed.

Proof. Start with a Cauchy or completion consumer. If it asks for limit identification, the read first passes through E and L, so the two completion-facing rows are bound to the same displayed metric source.

Then apply the separated-metric route through X, D, Z. The metric row supplies the distance witness, the $\mathsf{Real}^{\uparrow}$ row reads that witness as zero, and the refusal row allows only the corresponding classifier identification.

The remaining rows H, C, P, N transport, replay, source, and name the route. No host quotient, selected limit, countable-choice schedule, trichotomy row, or ambient completeness theorem is present in the carrier, so the readback is confined to the listed BEDC rows. □

Theorem 1411.61 (SeparatedMetric root completion-limit handoff)**.** *Every completion-limit classifier export from an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance carrier must first route through the $\mathsf{CompleteMetric}^{\uparrow}$ row, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row, $\mathsf{Met}^{\uparrow}$ source row, $\mathsf{Real}^{\uparrow}$ zero-distance witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row. The export is therefore a readback of the finite carrier in Definition 1411.58 together with Definition 1411.60, not a direct read from ambient completion equality.*

Proof. Project the carrier of Definition 1411.58. The completion-facing rows are L and E, the metric and zero-distance rows are X, D, Z, and the structural rows are H, C, P, N.

Use Definition 1411.60. It already forces Cauchy and completion consumers to route through the displayed $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ rows.

The completion-limit classifier export is therefore downstream of $L, E, X, D, Z, H, C, P, N$. Since no host completion equality or selected representative row occurs in the carrier, the export cannot bypass the finite handoff. □

Theorem 1411.62 (SeparatedMetric root classifier nonescape)**.** *The separated classifier of an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance carrier cannot be consumed from host equality, quotient*

equality, selected representatives, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. It is consumable only through the displayed* $\mathsf{Met}^{\uparrow}$ *source,* $\mathsf{Real}^{\uparrow}$ *zero-distance witness, separation row,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *handoff,* $\mathsf{CompleteMetric}^{\uparrow}$ *compatibility row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows.*

Proof. Project Definition 1411.58. Its rows are exactly $X, D, Z, E, L, H, C, P, N$, and none of the excluded host or quotient principles is a carrier coordinate.

Apply Definition 1411.60. Any completion-facing consumption of the classifier must pass through the displayed Cauchy and completion rows before the local name is exported.

The nonescape claim follows because every permitted classifier read is already accounted for by the finite carrier rows, while host equality, quotient equality, selected representatives, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no row to transport, replay, source, or name. □

Definition 1411.63 (SeparatedMetric root obligation basis)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet. Its root obligation basis is the finite* BHist*-anchored row package consisting of:*

1. *carrier admission of* $X, D, Z, L_0, L_1, E, H, C, P, N$ *as the only displayed rows of the packet;*
2. *classifier transport through the stored* $\mathsf{Met}^{\uparrow}$ *source,* $\mathsf{Real}^{\uparrow}$ *zero-distance reading, componentwise* hsame *transport, and displayed* Cont *replay;*
3. *completion handoff from* L_0, L_1 *through* $\mathsf{CompleteMetric}^{\uparrow}$ *and the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route* E *before any separated classifier export;*
4. *provenance and local naming through* P, N *for downstream L10 completion consumers.*

The basis exports no host equality, quotient equality, selected representative, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Theorem 1411.64 (SeparatedMetric root carrier admission)**.** *For every accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$*, the root-facing carrier admitted to a downstream completion consumer is exactly the* BHist *package displayed in Definition 1411.63. In particular, the consumer may project the rows* $X, D, Z, L_0, L_1, E, H, C, P, N$*, and it may not project any additional row for host equality, quotient equality, selected representatives, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 1411.1 together with the local obligation surface of Definition 1411.3. These rows display the metric source, zero-distance reading, separation classifier, two completion-limit entries, regular Cauchy equivalence route, transport, replay, provenance, and local naming.

The later zero-distance and completion-readback theorems use only those coordinates. Since the displayed carrier contains no coordinate for the excluded host or quotient principles, a root consumer has no additional row to project. □

Theorem 1411.65 (SeparatedMetric root classifier transport)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *packet. Every root classifier transport exported from* S *factors through the displayed* $\mathsf{Met}^{\uparrow}$ *row* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row* D*, the separated classifier row* Z*, componentwise* hsame *transport* H*, and displayed* Cont *replay* C*, with provenance and naming supplied by* P, N*. The transport remains a* BHist*-anchored classifier route and does not become host point equality, quotient equality, a selected representative, or trichotomy of real distances.*

Proof. Begin with Definition 1411.17. It already requires the zero-distance evidence to start in the $\mathsf{Met}^{\uparrow}$ distance row, pass through the $\mathsf{Real}^{\uparrow}$ zero-distance reading, and reach the classifier only through the separation row.

Apply the classifier stability and nonescape rows, Definition 1411.15 and Definition 1411.62. These rows keep transport componentwise through H and replay it through C, while P, N source and name the same finite route.

Thus the transported classifier remains the displayed $\mathsf{SeparatedMetric}^{\uparrow}$ route. No excluded equality or order principle is present in the carrier, so none can be exported by the transport. □

Theorem 1411.66 (SeparatedMetric root completion handoff)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ packet. Every root completion-limit read of S must hand L_0 and L_1 through $\mathsf{CompleteMetric}^{\uparrow}$ compatibility and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ route E before it may use D and Z to export a separated classifier. The handoff is a finite BHist route through $X, D, Z, L_0, L_1, E, H, C, P, N$, not a read from ambient completion equality or a selected limit.*

Proof. Use Definition 1411.18. It places every comparison of the two completion-limit rows behind the displayed equivalence route E before any zero-distance classifier read can occur.

Then apply Definition 1411.61. The theorem identifies the completion-facing rows, metric source, zero-distance witness, transport, replay, provenance, and naming as the complete route for a completion-limit classifier export.

The handoff therefore stays inside the finite carrier rows. Ambient completion equality, selected representatives, and choice principles are absent from S, so they cannot supply a competing completion read. □

Theorem 1411.67 (SeparatedMetric root obligation basis)**.** *The root obligation basis of Definition 1411.63 is sufficient for every downstream L10 completion consumer that asks a $\mathsf{SeparatedMetric}^{\uparrow}$ packet for carrier admission, classifier transport, or completion-limit uniqueness. Such a consumer can read the required rows by applying Definition 1411.64, Definition 1411.65, and Definition 1411.66, and every read remains confined to the BHist package $X, D, Z, L_0, L_1, E, H, C, P, N$.*

Proof. First read carrier admission from Definition 1411.64. This gives the downstream consumer exactly the displayed rows of the accepted packet and refuses every host, quotient, choice, trichotomy, compactness, or ambient completeness row.

For classifier requests, use Definition 1411.65. It transports the zero-distance classifier only through $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data already present in the carrier.

For completion-limit uniqueness requests, use Definition 1411.66. It sends the two completion rows through $\mathsf{CompleteMetric}^{\uparrow}$ compatibility and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ before the zero-distance classifier is exported.

These three projections cover the root-facing requests named in the definition, so the L10 consumer receives an obligation basis rather than a stronger equality or completion principle. □

Theorem 1411.68 (SeparatedMetric zero-distance Real reflection)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ be an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ carrier. Every completion-facing read that reflects a zero-distance claim into a Real-facing comparison must first display the $\mathsf{Met}^{\uparrow}$ distance rows of X, the $\mathsf{Real}^{\uparrow}$ zero-distance seal D, the separated classifier row Z, the $\mathsf{CompleteMetric}^{\uparrow}$ limit rows L_0, L_1, the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ handoff E, and the structural rows H, C, P, N.*

The reflection is a finite NameCert route through $X, D, Z, L_0, L_1, E, H, C, P, N$. It is not quotient completion equality, selected limit equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket zero distance real reflection`

Proof. Project the accepted carrier and read the Real-facing rows first. The metric source X supplies the distance witness, and D reads that witness as a $\mathsf{Real}^{\uparrow}$ zero-distance seal before the separated classifier Z may be exported.

For completion-facing requests, use the completion rows L_0, L_1 and the handoff E. These rows bind the two proposed completion readings to the displayed $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ data before the zero-distance route is consumed.

Transport and replay are then componentwise through H and C, with P, N sourcing and naming the same route. Since the carrier has no quotient equality, selected limit, trichotomy, or ambient Real-completeness coordinate, the reflection is exhausted by the displayed rows. □

Theorem 1411.69 (SeparatedMetric Cauchy-equivalence limit coherence)**.** *Let $S = (X, D, Z, L_0, L_1, E, H, C, P, N$ be an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ carrier. If a downstream completion consumer reads the two $\mathsf{CompleteMetric}^{\uparrow}$ limit rows L_0, L_1 as coherent, then the coherence read must pass through the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ row E, the $\mathsf{Met}^{\uparrow}$ row X, the $\mathsf{Real}^{\uparrow}$ zero-distance row D, the separated classifier row Z, and the structural rows H, C, P, N.*

Thus completion uniqueness is exported only as Cauchy-equivalence limit coherence followed by separated zero-distance classification. It is not ambient completion equality, a selected representative theorem, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricCauchyEquivalenceLimitCoherence`

Proof. Begin with the two limit rows. The carrier exposes L_0 and L_1 only as $\mathsf{CompleteMetric}^{\uparrow}$ rows paired with the Cauchy-equivalence handoff E.

Read E before any classifier conclusion. It supplies the finite Cauchy-sequence comparison route that lets the two limit rows be placed over the same metric source X.

The zero-distance claim is then read through D and consumed by Z. The transport, replay, provenance, and local naming rows H, C, P, N preserve exactly this route for downstream consumers.

No quotient completion object, selected limit, trichotomy row, or ambient completeness theorem is present in the carrier. Therefore the only exported coherence is the displayed Cauchy-equivalence and separated-classifier route. □

Theorem 1411.70 (SeparatedMetric NameCert ledger exhaustion)**.** *For an accepted $\mathsf{SeparatedMetric}^{\uparrow}$ carrier $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$, the completion-uniqueness ledger is exhausted by the rows $X, D, Z, L_0, L_1, E, H, C, P, N$. Every $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ consumer that asks for separated completion uniqueness must project those rows before it can read the local $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ conclusion.*

No additional ledger row is available for quotient completion equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Lean4 ✓ `BEDC.Derived.SeparatedMetricUp.SeparatedMetricPacket namecert ledger exhaustion`

Proof. Project the carrier inventory. It lists the metric source, Real zero-distance seal, separated classifier, two complete-metric limit rows, Cauchy-equivalence handoff, transport, replay, provenance, and local name.

Apply the zero-distance reflection route of Definition 1411.68. That route accounts for Real-facing zero-distance reads without adding a host equality coordinate.

Apply the limit-coherence route of Definition 1411.69. That route accounts for completion-facing uniqueness reads by sending L_0, L_1 through E, X, D, Z, H, C, P, N.

These two projections cover the completion-uniqueness ledger. Since no further row exists in the carrier, every excluded equality, selection, trichotomy, or ambient completeness principle is refused by ledger exhaustion. □

Theorem 1411.71 (SeparatedMetric root finite completion readiness)**.** *For an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier*

$$S = (X, D, Z, L_0, L_1, E, H, C, P, N),$$

the rows $X, D, Z, L_0, L_1, E, H, C, P, N$ *jointly provide the root completion-readiness package for every downstream* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *consumer. The package consists of the displayed* $\mathsf{Met}^{\uparrow}$ *distance source,* $\mathsf{Real}^{\uparrow}$ *zero-distance seal, separated classifier, two completion limit rows, Cauchy-equivalence handoff, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{SeparatedMetric}^{\uparrow}}$ *row.*

The package does not include quotient completion equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply the zero-distance Real reflection of Definition 1411.68. It places every Real-facing zero-distance read behind the displayed $\mathsf{Met}^{\uparrow}$ source X, the $\mathsf{Real}^{\uparrow}$ seal D, the separated classifier Z, the completion rows L_0, L_1, the Cauchy-equivalence handoff E, and the structural rows H, C, P, N.

Next apply the Cauchy-equivalence limit coherence route of Definition 1411.69. It supplies the completion-facing route that sends the two $\mathsf{CompleteMetric}^{\uparrow}$ limit rows through E, X, D, Z, H, C, P, N before any uniqueness conclusion is read.

Finally apply the ledger exhaustion theorem of Definition 1411.70. Since the carrier contains no row beyond $X, D, Z, L_0, L_1, E, H, C, P, N$, these three routes jointly form the finite root package available to $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ consumers. The excluded quotient equality, selected-limit, trichotomy, and ambient-completeness objects are absent from the displayed carrier. □

Theorem 1411.72 (SeparatedMetric zero-distance scope obligations)**.** *Let* $S = (X, D, Z, L_0, L_1, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparatedMetric}^{\uparrow}$ *carrier. Every zero-distance-to-classifier export from* S *must expose, in order, the* $\mathsf{Met}^{\uparrow}$ *distance witness in* X*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance read through* D*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0, L_1 *when the export is completion-facing, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}$ *row* N *before the separated classifier row* Z *is consumed.*

Thus the zero-distance export is scoped by the displayed finite packet. It is not host equality, quotient completion equality, selected representative data, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier definition Definition 1411.1. Its zero-distance route begins with the metric row X and the real-valued distance witness interface D, while the classifier consequence is stored only in the separation row Z.

Apply the NameCert obligation surface of Definition 1411.3. Carrier admission, classifier stability, completion compatibility, zero-distance exactness, and ledger visibility require the same displayed rows before any classifier export is available.

For completion-facing reads, use the completion-limit uniqueness route Definition 1411.4. The limit rows L_0, L_1 are compared through the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ route E before the metric witness and $\mathsf{Real}^{\uparrow}$ zero-distance read are formed.

Finally use zero-distance consumer exactness, Definition 1411.6. The rows H, C, P, N preserve, replay, source, and name the same finite route, so no excluded equality, selector, compactness, or ambient completeness principle can enter the export. □

Theorem 1411.73 (SeparatedMetric complete-limit uniqueness forward route)**.** *Every* CompleteMetricLimitUnique *consumer that takes a* SeparatedMetric$^{\uparrow}$ *packet as its separation witness must first expose the displayed* Met$^{\uparrow}$ *distance row, the* Real$^{\uparrow}$ *zero-distance row, the* CompleteMetric$^{\uparrow}$ *limit rows, the* CauchySequenceEquivalence$^{\uparrow}$ *comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before the uniqueness handoff. The consumer route is therefore the finite row chain already named by chapter 5983, Definition 1411.4, Definition 1411.50, and Definition 1411.71.*

It exports completion-limit uniqueness only as a separated zero-distance classifier consequence of the displayed rows. It does not export quotient completion equality, selected host limits, countable choice, host equality, trichotomy, compactness, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the CompleteMetricLimitUniqueness$^{\uparrow}$ request. The consumer may compare the two completion-limit rows only after the CompleteMetric$^{\uparrow}$ entries have been displayed and tied to the same finite-window route by the CauchySequenceEquivalence$^{\uparrow}$ row, as in Definition 1411.4.

Next apply the completion-uniqueness consumer obligation of Definition 1411.50. It sends the comparison through the Met$^{\uparrow}$ distance row and the Real$^{\uparrow}$ zero-distance seal before the separated classifier row may be consumed.

Finally use the root finite completion-readiness package of Definition 1411.71. The structural rows hsame, Cont, Pkg, and NameCert transport, replay, source, and name the same finite chain for the downstream consumer, so no quotient completion equality, selected limit, host equality, choice, trichotomy, compactness, or ambient completeness row can replace the displayed handoff. □

Chapter 1412

A Concrete Naming Certificate for $\mathsf{CauchySequenceEquivalence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ packet records the concrete equivalence relation on regular Cauchy names as finite BEDC data. It sits between the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The equivalence packet is downstream of chapter 3413 because the regular-name comparison only exists after the Cauchy source windows have been licensed. The packet compares two regular Cauchy names through displayed finite windows and tolerances before any real-seal consumer can read the result.

Definition 1412.1 (CauchySequenceEquivalence carrier surface)**.** *A* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, S, T, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name rows;*
2. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the paired windows;*
4. *E is the local finite equivalence classifier recording that every scheduled paired observation is within the displayed dyadic tolerance;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal-comparison row that may be consumed only after X, Y, S, T, D, E have been displayed;*
6. *H is the componentwise* hsame *provenance for the two regular names, the paired stream windows, the dyadic ledger, the classifier, and the seal-comparison row;*
7. *C is the displayed* Cont *route by which a consumer replays those finite windows without adding host data;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$ *row.*

The carrier exposes no quotient stream, selected limit, host real equality, countable-choice row, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceTasteGate single carrier alignment`

Theorem 1412.2 (CauchySequenceEquivalence NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$ *obligation surface consists exactly of the finite rows in Definition 1412.1. Carrier admission requires two displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *names, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a local finite equivalence classifier, a* $\mathsf{Real}^{\uparrow}$ *seal-comparison row, componentwise* hsame *provenance, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier stability is tested only by the paired finite windows and the stored dyadic ledger. Transport preserves the same rows through H and C. Tolerance exactness says that every accepted comparison reads exactly the dyadic ledger D. Ledger visibility says that every consumer can inspect $X, Y, S, T, D, E, R, H, C, P, N$ and no other equality source.*

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceNamecertObligations`

Theorem 1412.3 (CauchySequenceEquivalence RegSeqRat Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal comparison routed through an accepted* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *packet factors through the displayed sequence*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, a consumer first reads the two regular Cauchy name rows X, Y, then the paired finite stream windows S, T, then the dyadic tolerance ledger D and finite classifier E, and only then reaches the seal-comparison row R. The packet supplies no quotient stream, selected limit, host real equality, countable choice, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier regseqrat real handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ packet is generated from two displayed $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, paired $\mathsf{StreamName}^{\uparrow}$ finite windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite equivalence classifier, a $\mathsf{Real}^{\uparrow}$ seal-comparison row, displayed hsame provenance, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1412.1, Definition 1412.2, and Definition 1412.3 display the bridge to the public finite handoff over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceEquivalence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CauchySequenceEquivalenceUp.CauchySequenceEquivalenceCarrier\_regseqrat\_real\_handoff`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, countable choice, host real equality, or a standard bridge.

Upgrade path: Bridge checking requires a standard bridge theorem for the displayed RegSeqRat and Real handoff surface.

Chapter 1413

A Concrete Naming Certificate for $\mathsf{RealSequenceLimit}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealSequenceLimit}^{\uparrow}$ packet records a displayed limit interface for sequences of constructive real names without importing host sequence convergence. It sits after the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational rows of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the nonquotient finite-window equality boundary of chapter 1404.

The sibling dependency route is explicit: chapter 38 supplies the terminal $\mathsf{Real}^{\uparrow}$ seal, chapter 301 supplies rational readback, chapter 126 supplies finite observation windows, and chapter 291 supplies the tolerance ledgers consumed by the limit-modulus obligations.

Definition 1413.1 (RealSequenceLimit carrier)**.** *A* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(S, L, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is a displayed sequence of* $\mathsf{Real}^{\uparrow}$ *names, each read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows selected for the packet.*
2. *L is the candidate limit* $\mathsf{Real}^{\uparrow}$ *row, again accompanied by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows made visible to the packet.*
3. *W is the finite tail-window schedule that records which sequence indices and which observation windows are inspected at each displayed tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the scheduled sequence windows and the scheduled limit windows.*
5. *E is the displayed convergence classifier, asserting that every scheduled tail window compares the corresponding sequence real to L through the recorded dyadic tolerance.*
6. *H records componentwise* hsame *transports for the sequence rows, limit row, window schedule, tolerance ledger, and convergence classifier.*
7. *C records the displayed* Cont *routes by which consumers inspect the sequence windows, the limit windows, the dyadic ledger, and the classifier.*
8. *P is the* Pkg *provenance for the* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and classifier rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^\uparrow}$ *row.*

The carrier stores a finite certificate surface for a displayed sequence-limit claim. It has no host sequence equality, no quotient limit, no countable-choice selector, no trichotomy row, and no ambient $\mathsf{Real}^\uparrow$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitTasteGate single carrier alignment`

Theorem 1413.2 (RealSequenceLimit NameCert obligation surface)**.** *For an accepted* $\mathsf{RealSequenceLimit}^\uparrow$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^\uparrow}$ *surface consists of five finite rows:*

1. *carrier admission for the displayed sequence of* $\mathsf{Real}^\uparrow$ *names, the candidate limit* $\mathsf{Real}^\uparrow$ *row, the scheduled* $\mathsf{StreamName}^\uparrow$ *and* $\mathsf{RegSeqRat}^\uparrow$ *observations, the dyadic tolerance ledger, transports, continuations, provenance, and local naming row;*
2. *convergence classification by the finite tail-window schedule W, the* $\mathsf{DyadicRatCore}^\uparrow$ *ledger D, and the displayed classifier row E;*
3. *tail-window transport, stating that componentwise* hsame *transports and displayed* Cont *routes preserve the same scheduled sequence-to-limit comparison surface;*
4. *real-seal boundary through* $\mathsf{RegularCauchyEquivalence}^\uparrow$*, stating that every equality-facing read of a sequence real against the candidate limit factors through the nonquotient regular-Cauchy comparison boundary before any* $\mathsf{Real}^\uparrow$ *seal is consumed;*
5. *ledger refusal, stating that host sequence equality, quotient limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^\uparrow$ *completeness are not packet coordinates.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitNameCert obligation surface`

Proof. Carrier admission is the tuple of Definition 1413.1. The sequence rows, limit row, finite windows, dyadic ledger, classifier, transports, continuations, package provenance, and local naming row are all explicit BHist coordinates.

The convergence classifier is finite because it reads only the tail windows listed in W and the tolerances listed in D. Each displayed comparison is made between scheduled $\mathsf{RegSeqRat}^\uparrow$ observations of a sequence element and scheduled $\mathsf{RegSeqRat}^\uparrow$ observations of the candidate limit.

Tail-window transport reads the rows H and C componentwise. Since the transported data still contains the same schedule, ledger, and classifier shape, every consumer route remains attached to the displayed finite comparison surface.

The real-seal boundary is inherited from the sibling route of chapter 1404. Equality-facing reads pass through the regular-Cauchy finite-window classifier before the $\mathsf{Real}^\uparrow$ seal rows are consumed.

The refusal row is an inventory check. The carrier has no coordinate for host sequence equality, quotient limits, countable-choice selection, trichotomy, or ambient real completeness, so none of those principles can be exported by the local certificate. □

Theorem 1413.3 (RealSequenceLimit observation-tail carrier)**.** *Every accepted* $\mathsf{RealSequenceLimit}^\uparrow$ *packet exposes only scheduled* $\mathsf{StreamName}^\uparrow$ *and* $\mathsf{RegSeqRat}^\uparrow$ *observations of the sequence and candidate limit through the finite schedule rows W and D. No consumer can read a sequence-to-limit comparison outside those displayed observation windows before the* $\mathsf{Real}^\uparrow$ *seal is reached.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitObservationTailCarrier`

Proof. Read the carrier of Definition 1413.1. The sequence row S and candidate limit row L are visible only through the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows selected by the packet.

Apply the obligation surface of Definition 1413.2. Its carrier-admission and convergence-classification rows force every comparison to use the finite tail-window schedule W and the dyadic tolerance ledger D before the real-seal boundary can be consumed.

The sibling references in the chapter introduction identify the available sources: $\mathsf{Real}^{\uparrow}$ contributes the terminal seal, $\mathsf{RegSeqRat}^{\uparrow}$ contributes regular rational readback, $\mathsf{StreamName}^{\uparrow}$ contributes finite windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ contributes the tolerance ledger. None of these routes supplies an unscheduled host sequence observation. □

Theorem 1413.4 (RealSequenceLimit classifier stability)**.** *For an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, the convergence classifier* E *is stable under componentwise* hsame *transports of* S*,* L*,* W*, and* D*. Transported consumers therefore inspect the same scheduled sequence windows, candidate-limit windows, and dyadic tolerances before reaching the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. The carrier definition gives H as the componentwise hsame row for the sequence rows, limit row, window schedule, tolerance ledger, and convergence classifier. Hence a transported packet still has displayed coordinates of the same shape.

Use Definition 1413.2. The convergence classification row is finite because it reads only the windows listed in W and the tolerances listed in D, and the tail-window transport row preserves that scheduled comparison surface.

The $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ sibling surfaces named in the introduction supply only their displayed rows. Since H transports those rows componentwise, E cannot acquire a host sequence equality, quotient limit, choice-selected tail, trichotomy row, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitClassifierStability`

Theorem 1413.5 (RealSequenceLimit tail-window transport)**.** *Every consumer read of a* $\mathsf{RealSequenceLimit}^{\uparrow}$ *sequence-to-limit comparison factors through the finite tail-window schedule* W *and the dyadic ledger* D *before reaching the* $\mathsf{Real}^{\uparrow}$ *seal. The route is*

$$S, L \longmapsto W, D \longmapsto E \longmapsto C \longmapsto P, N$$

with componentwise transport supplied by H*.*

Proof. Start with a displayed sequence-to-limit comparison request. By Definition 1413.1, the only sequence and limit coordinates available to the packet are S and L, and both are accompanied by scheduled finite observations.

Next apply Definition 1413.2. The convergence classifier row requires the request to pass through W and D, while the real-seal boundary prevents equality-facing reads from bypassing the regular-Cauchy comparison route named in the introduction.

The continuation row C replays the same scheduled comparison for consumers, and P, N record provenance and local naming for that route. The transport row H preserves the packet componentwise, so every replayed comparison remains attached to the same finite tail-window schedule before the $\mathsf{Real}^{\uparrow}$ seal is consumed. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitTailWindowTransport`

Theorem 1413.6 (RealSequenceLimit modulus extraction)**.** *For an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, the rows* W, D, E *expose a* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible tail threshold for the displayed sequence-to-limit claim. A tolerance request is first read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, then matched with a scheduled tail window in* W*, and finally admitted by the convergence classifier* E *before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Use the carrier projection of Definition 1413.1. The schedule row W lists the tail windows visible to the packet, D lists the dyadic tolerance requests attached to those windows, and E records the displayed convergence classifier over the same rows.

Apply the NameCert surface of Definition 1413.2. Its convergence-classification row forces every sequence-to-limit read to pass through W and D, while its real-seal boundary prevents a consumer from replacing that finite threshold by an ambient completeness argument.

The resulting threshold is compatible with $\mathsf{CauchyModulus}^{\uparrow}$ because it is a displayed tail-window response to a dyadic tolerance request. It supplies only the scheduled tail index and the associated readback windows, not a host-level modulus function or a choice-selected tail. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitModulusExtraction`

Theorem 1413.7 (RealSequenceLimit tail refinement ledger)**.** *For an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, refining a dyadic tolerance request remains inside the same* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *route. The refined request may shrink the scheduled windows recorded in* W *and the tolerance row recorded in* D*, but its classifier read still factors through* E, H, C, P, N *before the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with a displayed tolerance request already admitted by D. A refinement is another dyadic request in the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, so it changes the requested precision but not the packet coordinates that are available to the consumer.

The finite window route is fixed by Definition 1413.5. The sequence and limit observations still pass through $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks selected by W, and the classifier row E still compares only those scheduled observations.

Transport and replay do not add new routes. The rows H and C preserve the same componentwise schedule, while P and N record provenance and naming for that schedule. Thus refinement stays inside the displayed finite ledger and cannot introduce host sequence equality, quotient limits, choice, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitTailRefinementLedger`

Theorem 1413.8 (RealSequenceLimit limit uniqueness route)**.** *Let* $(S, L, W, D, E, H, C, P, N)$ *and* $(S, L', W', D', E', H', C', P', N')$ *be accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *candidate-limit rows for the same displayed sequence row* S*. If their scheduled tail windows are compared through a common refinement of the displayed* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *route, then every equality-facing comparison between* L *and* L' *factors through* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *before either* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Project both accepted carriers. The two candidate limits expose only their scheduled finite observations, dyadic tolerance rows, convergence classifiers, transport rows, replay rows, provenance, and local names.

Use the common refinement of the tail-window routes. By Definition 1413.7, both candidates are read through the same kind of $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data, so the comparison between their scheduled limit observations is still finite-window data.

Apply the real-seal boundary in Definition 1413.2. Equality-facing reads cannot jump directly to either $\mathsf{Real}^{\uparrow}$ seal; they must first pass through the nonquotient finite-window comparison boundary named by $\mathsf{RegularCauchyEquivalence}^{\uparrow}$.

Hence uniqueness is a route statement: two accepted candidate-limit rows compared through the same scheduled tails meet at the regular-Cauchy comparison boundary before either terminal real seal is consumed. The packet does not assert host uniqueness of limits outside that displayed route. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitLimitUniquenessRoute`

Theorem 1413.9 (RealSequenceLimit observation-window obligation)**.** *Every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet reads sequence observations only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite windows and the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ *rows of Definition 1413.1. In particular, the packet exposes no unscheduled host sequence observation, quotient sequence equality, or hidden convergence coordinate before the* $\mathsf{Real}^{\uparrow}$ *seal is reached.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitObservationWindowObligation`

Proof. Project the carrier. The sequence row S, candidate limit row L, scheduled window row W, and dyadic tolerance row D are the only observation-facing coordinates available to an accepted packet.

The NameCert obligation surface in Definition 1413.2 records that every convergence classification reads through W, D, E before any terminal seal. The tail-window transport theorem Definition 1413.5 gives the same factorization for consumer reads.

Thus an observation request can only be answered by the displayed finite window route and its regular-rational and dyadic rows. The carrier has no coordinate for host sequence equality, quotient limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1413.10 (RealSequenceLimit Cauchy-modulus handoff)**.** *For every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, the limit row consumes* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible tolerance data through* D *and the scheduled tail windows in* W *before any* $\mathsf{Real}^{\uparrow}$ *seal is read. The handoff exports only the displayed tail threshold, window schedule, classifier admission, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitCauchyModulusHandoff`

Proof. Start with a tolerance request. By Definition 1413.6, the request is first read as dyadic tolerance data in D and then matched with a scheduled tail window in W.

The classifier row E admits the displayed comparison only after those two rows have been consumed. The tail-refinement ledger in Definition 1413.7 keeps refined requests inside the same $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ route.

Rows H, C, P, N then transport, replay, source, and name the same handoff. Nothing in this route selects a host modulus function, chooses a tail by countable choice, or reads ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Definition 1413.11 (RealSequenceLimit finite modulus carrier)**.** *A* $\mathsf{RealSequenceLimit}^{\uparrow}$ *finite modulus packet is a displayed tuple*

$$(S, L, W, D, E, M, H, C, P, N)$$

obtained from an accepted carrier $(S, L, W, D, E, H, C, P, N)$ *by inserting the threshold row* M *extracted from the scheduled window, dyadic tolerance, and classifier rows* W, D, E*. The row* M *is the* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible tail threshold whose existence is read by Definition 1413.6: a dyadic tolerance request is first admitted by* D*, matched to a finite tail window in* W*, and then certified by* E *as the displayed sequence-to-limit comparison before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

The packet contains no host modulus function, no selected infinite tail, no quotient stream equality, no countable-choice coordinate, and no ambient completeness row. It records only the scheduled $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, classifier admission, transport, replay, provenance, and local naming rows needed to expose* M *as finite BEDC data.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitFiniteModulusCarrier threshold route`

Theorem 1413.12 (RealSequenceLimit finite modulus NameCert obligations)**.** *For every accepted finite modulus packet* $(S, L, W, D, E, M, H, C, P, N)$ *of Definition 1413.11, the local* $\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^{\uparrow}}$ *obligation surface contains the following finite rows:*

1. *carrier admission for the displayed sequence row* S*, candidate limit row* L*, scheduled window row* W*, dyadic tolerance row* D*, classifier row* E*, extracted threshold row* M*, transport row* H*, replay row* C*, provenance row* P*, and naming row* N*;*
2. *dyadic tolerance read, stating that every tolerance request reaching* M *first appears as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row in* D*;*
3. *scheduled tail-window read, stating that* W *supplies the finite tail window attached to that dyadic request;*
4. *classifier admission, stating that* E *admits the displayed sequence-to-limit comparison only after the* D *and* W *rows have been read;*
5. $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{StreamName}^{\uparrow}$ *transport, stating that the observations used by* M *are the finite stream windows and regular-rational readbacks named by the packet;*
6. $\mathsf{Real}^{\uparrow}$ *seal nonescape, stating that the terminal real seal supplies no hidden modulus, no host sequence equality, no quotient limit, no trichotomy, and no ambient completeness witness;*
7. *replay, provenance, and naming, stating that* C, P, N *re-export the same finite threshold route without changing its visible rows.*

Proof. Start with Definition 1413.11. The tuple itself supplies the carrier-admission row and records M as the displayed threshold extracted from W, D, E.

Apply Definition 1413.6. It forces a tolerance request to pass through the dyadic ledger D, then through the scheduled tail window W, and finally through the classifier E before the threshold row can be read.

Next use Definition 1413.5. The observations supporting this threshold remain inside the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces named by the packet, with componentwise transport supplied by H and consumer replay supplied by C.

Finally apply Definition 1413.10. The resulting M is exactly the finite $\mathsf{CauchyModulus}^{\uparrow}$-compatible handoff row. The $\mathsf{Real}^{\uparrow}$ seal nonescape clause follows from Definition 1413.14, while P and N record provenance and local naming for the same finite route. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitModulusNameCertObligations`

Theorem 1413.13 (RealSequenceLimit modulus RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a* $\mathsf{RealSequenceLimit}^{\uparrow}$ *sequence-limit packet must read the finite modulus packet* $(S, L, W, D, E, M, H, C, P, N)$ *of Definition 1413.11 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed. Thus the consumer route factors through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, W, D, E, M, H, C, P, N$$

and cannot replace M *by a host modulus function, a selected infinite tail, quotient stream equality, countable choice, trichotomy, or ambient real completeness.*

Proof. Use Definition 1413.10 to identify the consumer-facing tolerance handoff. It already routes through D, W, and the classifier E before the $\mathsf{Real}^{\uparrow}$ seal is read.

The carrier of Definition 1413.11 packages the same handoff with the explicit threshold row M. Hence a consumer cannot reach the terminal real seal without first seeing the scheduled dyadic request, finite stream windows, regular-rational readbacks, classifier admission, and extracted threshold.

The transport and replay clauses are supplied by Definition 1413.5. The extraction clause is supplied by Definition 1413.6. Together with the $\mathsf{Real}^{\uparrow}$ nonescape boundary of Definition 1413.14, these rows force every Real-facing read to pass through the finite modulus packet and exclude all non-displayed modulus sources. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.ModulusRealUpHandoff.RealSequenceLimitModulusRealUphandoff`

Theorem 1413.14 (RealSequenceLimit Real seal non-escape)**.** *For an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet, every* $\mathsf{Real}^{\uparrow}$*-facing read of the candidate limit or of a sequence-to-limit comparison factors through the carrier, NameCert obligation surface, observation window route, Cauchy-modulus handoff, and limit uniqueness route before the real seal is consumed. The packet excludes host sequence equality, quotient limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier and the NameCert obligation surface to list the available rows. They provide the sequence row, candidate limit row, finite windows, dyadic tolerances, convergence classifier, transport, replay, provenance, and local naming.

The observation-window obligation and Cauchy-modulus handoff identify the only routes by which observation and tolerance data can reach the classifier. The limit uniqueness route then shows that equality-facing reads between candidate limits must pass through $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ before either $\mathsf{Real}^{\uparrow}$ seal is read.

Consequently the real seal is terminal for this displayed route rather than a source of extra principles. No row supplies host sequence equality, quotient limits, countable choice, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitRealSealNonescape`

Theorem 1413.15 (RealSequenceLimit Cauchy-modulus extraction)**.** *For every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, the scheduled rows* W, D, E *expose a* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible threshold before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. A tolerance request is admitted only after* D *records the dyadic tolerance,* W *supplies the corresponding finite tail window, and* E *classifies the displayed sequence-to-limit comparison through that scheduled window.*

Proof. Start with the observation-tail carrier of Definition 1413.3. It restricts all visible sequence and candidate-limit observations to the scheduled rows W and D.

The classifier stability theorem Definition 1413.4 and the tail-window transport theorem Definition 1413.5 keep the same comparison route after componentwise transport. Thus the classifier cannot move a tolerance request to an unscheduled observation window.

Apply the modulus extraction theorem Definition 1413.6 and the Cauchy-modulus handoff theorem Definition 1413.10. They identify the displayed threshold as the finite tail response to the dyadic request. The tail-refinement ledger of Definition 1413.7 preserves that response under smaller tolerances, and the real-seal non-escape theorem Definition 1413.14 prevents ambient $\mathsf{Real}^{\uparrow}$ completeness from supplying any further threshold. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitCauchyModulusExtraction`

Theorem 1413.16 (RealSequenceLimit regular classifier transport)**.** *For every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *carrier* $(S, L, W, D, E, H, C, P, N)$*, componentwise* hsame *transport of the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows preserves the same regular-Cauchy classifier route for candidate limits. Transported consumers still compare the sequence row and candidate limit through* W, D, E, H, C, P, N *before any* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Use the carrier definition to project the finite stream windows, regular rational readbacks, dyadic tolerance ledger, convergence classifier, transport, replay, provenance, and local naming rows.

Classifier stability in Definition 1413.4 says that the row E is stable under the componentwise transports stored in H. Tail-window transport in Definition 1413.5 then replays the transported comparison through the same scheduled windows and dyadic ledger.

The observation-tail carrier and tail-refinement ledger, Definition 1413.3 and Definition 1413.7, keep refined or transported reads inside the $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ route. Finally, Definition 1413.14 blocks host sequence equality, quotient limits, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness from entering the transported classifier. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitRegularClassifierTransport`

Theorem 1413.17 (RealSequenceLimit Cauchy criterion factorization)**.** *Every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet exposing rows* $S, L, W, D, E, H, C, P, N$ *factors each displayed sequence-to-limit comparison through the* $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ *finite Cauchy tail-window route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The factorization reads the sequence and candidate limit through* S, L*, the scheduled observation and dyadic rows through* W, D*, the convergence classifier through* E*, the transport and replay rows through* H, C*, and then hands the resulting tail-window comparison to the sibling Cauchy-criterion surface before any Real-facing consumer can inspect* P, N*.*

Proof. Start with the carrier and obligation surface, Definition 1413.1 and Definition 1413.2. They expose exactly the displayed sequence row, candidate limit row, scheduled windows, dyadic tolerance ledger, classifier, transport, replay, provenance, and local naming rows.

Use Definition 1413.5 to keep the comparison inside the same scheduled finite windows after componentwise hsame transport. The regular classifier transport theorem Definition 1413.16 then identifies the transported comparison as a route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows rather than through host sequence equality.

Apply the sibling handoff Definition 2255.3. It consumes the displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ row and returns the Cauchy tail-window comparison through finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ data.

Finally, Definition 1413.14 makes the Real$^{\uparrow}$ seal terminal for this route. Thus the sequence-to-limit comparison reaches any Real-facing consumer only after passing through the finite Cauchy-criterion factorization, and it does not claim ambient completeness, quotient limits, countable choice, or trichotomy. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitCauchyCriterionFactorization`

Theorem 1413.18 (RealSequenceLimit regular Cauchy tail factorization)**.** *Every accepted* RealSequenceLimit$^{\uparrow}$ *sequence-to-limit comparison factors through the displayed regular Cauchy tail route before the terminal* Real$^{\uparrow}$ *seal is consumed. Concretely, the comparison reads the sequence and candidate limit rows through* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *tolerances, the* CauchyModulus$^{\uparrow}$ *threshold handoff, and the* RealSequenceCauchyCriterion$^{\uparrow}$ *tail route, and only then reaches the Real-facing seal, transport, replay, provenance, and local naming rows. The route refuses host sequence equality, quotient limits, countable choice, trichotomy, and ambient Real completeness.*

Proof. Start with the observation-tail carrier Definition 1413.3 and the tail-window transport theorem Definition 1413.5. Together they force every sequence and candidate-limit read through the scheduled StreamName$^{\uparrow}$ windows and the matching RegSeqRat$^{\uparrow}$ readbacks before any consumer replay is available.

Next apply Definition 1413.6 and Definition 1413.10. These theorems identify the dyadic tolerance row as a CauchyModulus$^{\uparrow}$-compatible finite threshold route rather than as a host modulus function or a selected infinite tail.

Use Definition 1413.17. It hands the scheduled comparison to the RealSequenceCauchyCriterion$^{\uparrow}$ finite Cauchy tail-window surface before the terminal Real$^{\uparrow}$ seal can be inspected.

Finally apply Definition 1413.14. The seal is terminal for the displayed route, so it cannot supply host sequence equality, quotient limits, countable choice, trichotomy, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RegularCauchyTailFactorization.RealSequenceLimitRegularCauchyT`

Theorem 1413.19 (RealSequenceLimit factorization completeness)**.** *Every accepted* RealSequenceLimit$^{\uparrow}$ *comparison is exhausted by the displayed L10 factorization route. The comparison first reads the scheduled* StreamName$^{\uparrow}$ *windows, then the* RegSeqRat$^{\uparrow}$ *readbacks, then the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, then the* CauchyModulus$^{\uparrow}$ *threshold handoff, then the* RealSequenceCauchyCriterion$^{\uparrow}$ *tail route, then the* RegularCauchyEquivalence$^{\uparrow}$ *comparison boundary, and only then reaches* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The terminal* Real$^{\uparrow}$ *seal is read only inside that route.*

No accepted comparison can enter through host sequence equality, quotient limits, countable choice, trichotomy, selected host tails, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the carrier and NameCert surface of Definition 1413.2. It lists exactly the visible sequence row, candidate limit row, scheduled windows, dyadic ledger, classifier, transport, replay, provenance, and local naming rows.

Apply the modulus extraction theorem Definition 1413.6. It turns a tolerance request into a finite threshold read through DyadicRatCore$^{\uparrow}$ and CauchyModulus$^{\uparrow}$, so no host modulus or selected infinite tail is available.

Next use the Cauchy-criterion factorization of Definition 1413.17 and the regular Cauchy tail factorization of Definition 1413.18. These two results route the comparison through the sibling

RealSequenceCauchyCriterion$^{\uparrow}$ tail surface and the RegularCauchyEquivalence$^{\uparrow}$ boundary before any Real-facing consumer reads the seal.

Finally apply Definition 1413.14. The seal is terminal for the displayed L10 route, and the remaining rows H, C, P, N provide only transport, replay, provenance, and local naming. Thus the listed rows exhaust every accepted RealSequenceLimit$^{\uparrow}$ comparison. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.FactorizationCompleteness.RealSequenceLimitFactorizationComple`

Theorem 1413.20 (RealSequenceLimit subsequence tail stability)**.** *Every accepted subsequence consumer of a displayed* RealSequenceLimit$^{\uparrow}$ *packet* $K = (S, L, W, D, E, H, C, P, N)$ *must reindex through the same* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *tolerances, and* RealSequenceCauchyCriterion$^{\uparrow}$ *tail route before the terminal* Real$^{\uparrow}$ *seal is consumed. The subsequence request may name a displayed increasing index row, but it receives no host subsequence selector, quotient limit, countable-choice schedule, trichotomy split, or ambient* Real$^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitSubsequenceTailStability`

Proof. Start with the carrier of Definition 1413.1. The visible rows are the displayed sequence, candidate limit, finite observation windows, dyadic tolerance ledger, classifier, transport, replay, provenance, and local naming rows. A subsequence-facing consumer has no additional coordinate from which to read a host-selected tail.

Use Definition 1413.5. Reindexing a displayed subsequence request can only choose subwindows already routed through the StreamName$^{\uparrow}$ schedules in W, with the matching RegSeqRat$^{\uparrow}$ readbacks and DyadicRatCore$^{\uparrow}$ tolerances recorded in D.

Apply Definition 1413.16. The classifier row E is transported through the same componentwise hsame rows H and replayed by C, so a subsequence read remains a transported finite-window comparison rather than a quotient limit or a host sequence equality.

Finally use Definition 1413.17. The transported subsequence comparison reaches the sibling RealSequenceCauchyCriterion$^{\uparrow}$ tail route before any Real$^{\uparrow}$ seal is visible. Thus subsequence extraction stays inside the L10 finite-window route and cannot import host subsequence selection, countable choice, trichotomy, quotient limits, or ambient completeness. □

Theorem 1413.21 (RealSequenceLimit diagonal-window uniqueness)**.** *Let* $K = (S, L, W, D, E, H, C, P, N)$ *be an accepted* RealSequenceLimit$^{\uparrow}$ *packet. If two accepted limit reads use the same* StreamName$^{\uparrow}$ *schedule row* W*, the same* RegSeqRat$^{\uparrow}$ *readback route, the same* DyadicRatCore$^{\uparrow}$ *tolerance ledger* D*, and the same terminal* Real$^{\uparrow}$ *seal, then both reads must pass through the same finite diagonal window before the classifier row* E *is consumed. The uniqueness is a finite-window route claim, not a quotient sequence equality, selected host limit, countable-choice schedule, trichotomy split, or ambient real-completeness assertion.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitDiagonalWindowUniqueness`

Proof. Start with the carrier of Definition 1413.1. The two candidate reads have no schedule coordinate beyond W and no tolerance coordinate beyond D.

Apply Definition 1413.5. Since both reads use the same scheduled StreamName$^{\uparrow}$ windows and the same RegSeqRat$^{\uparrow}$ readback route, transport can only replay the comparison through the displayed window route already stored in the packet.

Next use Definition 1413.16. The classifier row E is reached only after the transported $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ route is displayed, so the two reads cannot choose separate classifier-facing windows.

Finally, Definition 1413.20 rules out an alternate host subsequence selector, and Definition 1413.14 keeps the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite route. Thus the classifier consumes one shared diagonal window. □

Theorem 1413.22 (RealSequenceLimit limit-modulus composition)**.** *For every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet* $K = (S, L, W, D, E, H, C, P, N)$*, composing the finite Cauchy-modulus extraction route with subsequence-tail stability yields a finite modulus route for every accepted subsequence limit read. The route passes through* D*,* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the classifier row* E*, transport and replay through* H, C*, and the provenance and naming rows* P, N *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed; it does not use quotient sequence equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, or a selected host limit.*

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitLimitModulusComposition`

Proof. Use Definition 1413.15. A tolerance request first enters the dyadic ledger D, then the scheduled finite windows W, and then the classifier row E before any threshold can be read.

Apply Definition 1413.20. A subsequence-facing limit read may reindex through displayed subwindows, but it receives no independent host selector and no separate countable-choice schedule.

The regular readback and classifier transport are supplied by Definition 1413.16. They keep the composed route inside the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows after componentwise hsame transport and displayed Cont replay.

Finally, Definition 1413.5 identifies the transported windows with the finite route already used by the limit packet. The resulting modulus route reaches P, N and the terminal $\mathsf{Real}^{\uparrow}$ seal only after the finite composition has been displayed, so no quotient equality, ambient completeness, countable choice, or selected host limit enters the proof. □

Theorem 1413.23 (RealSequenceLimit squeeze handoff)**.** *Let* $K^- = (S^-, L, W^-, D^-, E^-, H^-, C^-, P^-, N^-)$*,* $K = (S, L, W, D, E, H, C, P, N)$*, and* $K^+ = (S^+, L, W^+, D^+, E^+, H^+, C^+, P^+, N^+)$ *be accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packets with the same displayed limit row* L*. Suppose the middle sequence row* S *shares a finite tail-window bound with the lower and upper rows* S^-, S^+ *through the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation route, and suppose the two bounding packets expose finite modulus rows for convergence to* L*. Then the middle packet inherits the same* $\mathsf{RealSequenceLimit}^{\uparrow}$ *finite modulus surface: each tolerance request for* K *factors through the displayed squeeze window, the bounding modulus rows, the regular classifier transport, diagonal-window uniqueness, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before* P, N *are read.*

Proof. Start with the displayed squeeze window. The hypothesis supplies a common $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Dya}$ route relating S^-, S, and S^+ on the finite tail requested by the tolerance ledger. Hence the middle packet receives no host order argument, no quotient sequence equality, and no selected infinite tail outside the displayed rows.

Apply Definition 1413.22 to the two bounding packets. Their convergence to the shared limit row L exposes finite modulus routes through their dyadic ledgers, scheduled windows, classifiers, transports, replays, provenance rows, and local names.

Use Definition 1413.16 to move the middle comparison through the same regular readback route. The squeeze window makes the transported middle read a $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ comparison already bounded by the two displayed modulus rows.

Finally apply Definition 1413.21. The lower, middle, and upper reads meet at one finite diagonal window before any classifier-facing read reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. Therefore the middle packet inherits the finite modulus surface already exposed by the bounding packets, and no ambient completeness, countable choice, trichotomy, selected host limit, or quotient sequence equality enters the handoff. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitSqueezeHandoff`

Theorem 1413.24 (RealSequenceLimit scoped dependency package)**.** *For every accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet* $K = (S, L, W, D, E, H, C, P, N)$*, every accepted limit read factors through the full finite dependency route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N$$

before the limit certificate is exported. Concretely, the tolerance request is recorded in the dyadic ledger D*, the finite observation window is selected from* W*, the rational approximation is read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the candidate limit and sequence values reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and only then do componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row expose the certificate.*

The scoped package is a dependency statement for the displayed packet. It does not assert host sequence convergence, quotient equality of streams, selected limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the carrier of Definition 1413.1. Its visible coordinates are the displayed sequence source, candidate limit row, finite windows, dyadic tolerance ledger, classifier, transport, replay, provenance, and local naming rows.

Apply the NameCert obligation surface of Definition 1413.2. Those obligations force a tolerance request to enter the dyadic ledger D and the finite window row W before any convergence classifier is consumed.

Use Definition 1413.5 and Definition 1413.16. Together they route every transported read through the same $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback, with componentwise hsame transport and displayed Cont replay preserving the dependency order.

Finally apply Definition 1413.14. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after the dyadic, window, and regular-readback rows are displayed, while P, N merely package and name that route. Hence the exported limit certificate has exactly the scoped dependency package stated above. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitScopedDependencyPackage`

Theorem 1413.25 (RealSequenceLimit regular-tail scope lock)**.** *Let* $K = (S, L, W, D, E, H, C, P, N)$ *be an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet. Any tail read that is admitted as a regular limit read is locked to the same* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route as* K*: changing the tail window, rational readback, or Real seal requires changing one of the displayed rows* W, D, E, H, C, P, N *of the packet. Thus regular tails are scoped by the packet rows themselves rather than by a host convergence predicate, an external subsequence selector, a quotient equality of streams, or ambient completeness.*

Proof. Project the packet rows. A regular tail read has no coordinate outside W, D, E, H, C, P, N from which to obtain a tolerance, observation window, rational readback, or terminal Real seal.

Use Definition 1413.5. Tail transport may reindex only through the displayed $\mathsf{StreamName}^{\uparrow}$ windows already stored in W, with tolerance requests still accounted for by the dyadic ledger D.

Next apply Definition 1413.16. The regular classifier can be consumed only after the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback and the componentwise hsame transport have been displayed, so a tail read cannot switch to an unrecorded regular route.

The Real boundary is fixed by Definition 1413.14, and diagonal reuse is controlled by Definition 1413.21. Therefore every admitted regular tail remains locked to the same finite dependency route before P, N export the named limit certificate. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimit regular tail scope lock`

Theorem 1413.26 (RealSequenceLimit bounded source handoff)**.** *Let $K = (S, L, W, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet whose displayed source sequence is supplied by an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packet $X = (S_0, M_0, D_0, R_0, E_0, B_0, H_0, C_0, P_0, N_0)$. Before the limit classifier E and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of K may be read, the source route must consume the boundedness rows*

$$S_0 \longmapsto M_0 \longmapsto D_0 \longmapsto R_0 \longmapsto E_0 \longmapsto B_0, H_0, C_0, P_0, N_0.$$

Consequently a RealSequenceLimit read from a bounded Cauchy source exposes the finite bound ledger B_0 before any limit-classifier or Real-seal consumer receives the source sequence, and it imports no host boundedness theorem, selected limit, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source packet. Apply Definition 1522.3: every real-facing or convergence-facing read from X passes through $S_0, M_0, D_0, R_0, E_0, B_0, H_0, C_0, P_0, N_0$, with B_0 carrying the explicit finite bound ledger.

Now use Definition 1413.25. The RealSequenceLimit packet can admit a source tail only through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route already stored in W, D, E, H, C, P, N.

Finally apply Definition 1413.14. The terminal Real seal cannot be reached before the finite dyadic, window, rational-readback, and transport rows are displayed. Since the bounded source supplies those rows only after B_0 is present, the limit classifier and Real seal inherit the boundedness route stated above. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.BoundedSourceHandoff.RealSequenceLimitBoundedSourceHandoff`

Theorem 1413.27 (RealSequenceLimit bridge surface)**.** *Every public* $\mathsf{RealSequenceLimit}^{\uparrow}$ *packet $K = (S, L, W, D, E, H, C, P, N)$ supplies a bridge surface whose rows are exhausted by the* RealSequenceCauchyCriter $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *routes together with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The bridge reads sequence-limit data only through the displayed finite windows, regular rational readbacks, dyadic tolerance ledger, equality-facing regular-Cauchy comparison, Cauchy-criterion handoff, terminal Real seal, and structural rows of K.*

Thus any later $\mathsf{StandardBridge}$ *consumer must use this public packet surface before interpreting the sequence-limit claim in an external standard presentation. The bridge surface contains no quotient limit, no selected representative, no host sequence equality, no countable-choice schedule, no trichotomy split, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 1413.1. The only sequence and candidate-limit observations available to a consumer are the displayed $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply Definition 1413.15 and Definition 1413.26. These route Cauchy-facing reads through the finite tolerance ledger and source rows before any limit classifier or Real seal is consumed, giving the $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ side of the bridge surface.

Next use Definition 1413.8 and Definition 1413.25. Equality-facing limit comparisons remain inside the $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route already displayed by the packet.

Finally, Definition 1413.24 supplies the structural boundary through H, C, P, N. Hence a bridge-facing consumer can read only the public packet rows named in the statement, and the carrier inventory leaves no coordinate for quotient limits, selected representatives, host sequence equality, choice, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealSequenceLimitUp.RealSequenceLimitBridgeSurface`

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSequenceLimit}^{\uparrow}$ packet is generated from a displayed sequence of RealUp names, a candidate limit RealUp row, finite StreamName windows and RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a finite convergence classifier, componentwise transport, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1413.1, Definition 1413.2, Definition 1413.3, Definition 1413.16, Definition 1413.5, Definition 1413.14, Definition 1413.21, Definition 1413.25, Definition 1413.26, and Definition 1413.27 bind the mature RealSequenceLimit treatment through the displayed bridge surface, bounded-source handoff, $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSequenceLimit}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not claim Lean verification, host sequence equality, quotient limits, selected representatives, countable choice, trichotomy, or ambient RealUp completeness.

Upgrade path: Remaining work is Lean verification and checked bridge status for the displayed RealSequenceLimit bridge surface, not further paper theory closure.

Chapter 1414

A Concrete Naming Certificate for $\mathsf{CauchyNullEquivalence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyNullEquivalence}^{\uparrow}$ packet records the constructive criterion that two regular Cauchy real names agree exactly when their displayed pointwise difference is certified by a null sequence. The packet is a finite NameCert boundary: it reads the source regular names of chapter 301, the difference row of chapter 429, the zero-tail row of chapter 380, the finite equivalence boundary of chapter 1404, and the seal rows of chapter 38.

It does not add a quotient of Cauchy streams, a selected host limit, a host equality of real numbers, a trichotomy branch, or an ambient completeness theorem. Its role is to name the reusable finite bridge between a displayed difference-null packet and the equality-facing read of regular Cauchy real names.

Definition 1414.1 (CauchyNullEquivalence carrier)**.** *A* $\mathsf{CauchyNullEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(X, Y, D, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular Cauchy real names are being compared.*
2. *D is the accepted* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row that displays the pointwise difference of X and Y, including its aligned* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *subtraction ledger.*
3. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail packet consuming the displayed difference row exported by D.*
4. *E is the equality-facing row, either the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *finite-window classifier or the* $\mathsf{Real}^{\uparrow}$ *seal/equality row that reads the same zero-tail evidence.*
5. *H records the componentwise* hsame *transports for the two source rows, the difference row, the null-sequence row, and the equality-facing row.*
6. *C records the displayed* Cont *routes by which consumers inspect the source rows, the difference packet, the zero-tail packet, and the equality-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNullEquivalence}^{\uparrow}}$ *row.*

The carrier stores one finite criterion surface: equality of the two displayed regular Cauchy real names is read through the same difference row and the same null-sequence packet. It stores no quotient stream, no selected limit, no host real equality, and no ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyNullEquivalenceUp.CauchyNullEquivalenceTasteGate single carrier alignment`

Theorem 1414.2 (CauchyNullEquivalence NameCert obligations)**.** *For an accepted* $\mathsf{CauchyNullEquivalence}^{\uparrow}$ *carrier* $(X, Y, D, Z, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyNullEquivalence}^{\uparrow}}$ *surface consists of six finite obligations:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row, the equality-facing* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *row, transports, continuations, provenance, and local naming data;*
2. *difference exactness, stating that* D *is the pointwise difference packet for the displayed source rows* X *and* Y*;*
3. *zero-tail consumption, stating that* Z *consumes exactly the difference row exposed by* D*;*
4. *equality handoff, stating that* E *reads equality of the two regular Cauchy real names only through the same displayed difference-null route;*
5. *classifier transport, stating that the source rows, difference row, zero-tail row, and equality-facing row transport componentwise through the stored* hsame *rows and displayed* Cont *routes;*
6. *ledger refusal, stating that quotient stream equality, host real equality, selected limits, trichotomy, countable choice, and ambient real completeness are not packet coordinates.*

Proof. Carrier admission is the tuple of Definition 1414.1. It exposes exactly two regular Cauchy source rows, their displayed difference packet, the zero-tail packet for that difference, the equality-facing classifier or seal row, and the finite transport, continuation, package, and naming rows.

Difference exactness is read from chapter 429. The row D already binds the source rows X, Y, the aligned $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ subtraction ledger, and the tolerance ledger for the displayed difference.

Zero-tail consumption is read from chapter 380. The row Z consumes the same difference stream and exposes only the tolerance-window-threshold data allowed by the null-sequence packet.

The equality handoff is read through chapter 1404 and chapter 38. The equality-facing row E may inspect the finite equivalence classifier or the real seal, but only after the difference row and null-sequence row have been displayed.

Classifier transport is componentwise over H and C. Since the carrier has no quotient, selected limit, host equality, trichotomy, choice, or completeness coordinate, the refusal row follows from the inventory. □

Theorem 1414.3 (CauchyNullEquivalence difference-zero route)**.** *For accepted* $\mathsf{CauchyNullEquivalence}^{\uparrow}$ *carriers, the equality-facing read of the two regular Cauchy real names and the zero-tail read of their displayed difference are the same finite route. More precisely, a consumer may pass from the source rows* X, Y *to the equality row* E *only by first reading the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *packet* D *and then the* $\mathsf{NullSequence}^{\uparrow}$ *packet* Z *for the displayed difference row; conversely, any such displayed difference-null route supplies exactly the finite equality handoff recorded by* E*.*

Proof. For the forward direction, start with an equality-facing consumer of E. By the carrier definition, E is stored after the source rows X, Y, the difference packet D, and the zero-tail packet Z.

Apply the real-seal and finite-window boundaries of chapter 1404. They force the consumer to read equality of regular Cauchy names through the finite comparison surface rather than through a quotient stream or host real equality.

Next use the difference surface of chapter 429. The comparison surface is therefore the displayed pointwise difference of X and Y, with its windows and dyadic subtraction ledger supplied by D.

Finally apply the zero-tail criterion of chapter 380. The equality read reaches E through the $\mathsf{NullSequence}^{\uparrow}$ row Z that consumes that same displayed difference.

For the reverse direction, begin with the displayed difference-null route D, Z. The row D names the pointwise difference of the two source regular names, and Z certifies that this displayed difference is null.

Read that zero-tail packet through the regular-Cauchy equivalence and real-seal boundaries of chapter 1404 and chapter 38. This produces exactly the equality-facing handoff row E in the carrier.

No additional analytic row is introduced in either direction. The route remains the finite packet $(X, Y, D, Z, E, H, C, P, N)$, so quotient streams, selected limits, host real equality, trichotomy, choice, and ambient completeness remain outside the claim. □

Theorem 1414.4 (CauchyNullEquivalence downstream equality surface)**.** *Any downstream completion or limit consumer that cites a* $\mathsf{CauchyNullEquivalence}^{\uparrow}$ *packet receives only the finite equality criterion*

$$(X, Y, D, Z, E, H, C, P, N).$$

The consumer may use the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *displayed difference, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row, and the equality-facing* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff, but it receives no quotient stream, selected host limit, host real equality, trichotomy, countable choice, or ambient completeness theorem.*

Proof. Project the accepted packet to the carrier of Definition 1414.1. The only equality-facing analytic data are the displayed difference row D, the null-sequence row Z, and the handoff row E.

Apply Definition 1414.2. The obligation package binds carrier admission, difference exactness, zero-tail consumption, equality handoff, componentwise transport, and ledger refusal inside the same finite packet.

Apply Definition 1414.3. It identifies the two directions of the criterion without adding an external real equality principle. Thus downstream consumers inherit one reusable finite route, not a stronger completion or quotient object. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNullEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNullEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, their $\mathsf{RegularCauchyDifference}^{\uparrow}$ displayed difference row, a $\mathsf{NullSequence}^{\uparrow}$ zero-tail packet for that difference, an equality-facing $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff row, componentwise hsame transport, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNullEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1414.1, Definition 1414.2, Definition 1414.3, and Definition 1414.4 state the finite criterion tying equality of regular Cauchy real names to a displayed null difference route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNullEquivalence}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyNullEquivalenceUp.CauchyNullEquivalenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, difference exactness, zero-tail consumption, equality handoff, classifier transport, and ledger refusal for the same finite packet.

Chapter 1415

A Concrete Naming Certificate for $\mathsf{BoundedRealSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BoundedRealSequence}^{\uparrow}$ packet records a bounded real sequence by finite stream-window rows, regular rational readback, a real-name seal, and a displayed closed-bounded interval net. It sits before the subsequence compactness route of chapter 1254 and exposes the boundedness surface that chapter consumes. Its dependency surface is the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the interval-net bound row of chapter 1250. The packet is a finite certificate for bounded sequence inspection; it is not open-cover compactness, trichotomy of reals, host completeness, quotient equality of streams, or a choice-selected subsequence.

Definition 1415.1 (Bounded real sequence carrier)**.** *A* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, Q, R, I, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source schedule of sequence indices under inspection.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows exposed for the inspected indices in S.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and their dyadic tolerance observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *sequence-name seal reached only after W and Q have been displayed.*
5. *I is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row carrying the interval net used to witness boundedness for the displayed sequence windows.*
6. *H stores componentwise* hsame *transport for the source schedule, stream windows, regular readback, real seal, interval-net bound, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from inspected indices to stream windows, from stream windows to regular readback, from regular readback to the real seal, and from the real seal to the interval-net bound assertion.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalN}$ $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row.*

The bound row is displayed as the closed interval-net containment surface

$$I : \mathsf{ClosedBoundedIntervalNet}^{\uparrow},$$
$$\forall i \in S_{\text{shown}} \quad W_i \longmapsto Q_i \longmapsto R_i \in I.$$

Here S_{shown} denotes only the inspected finite index rows of the carrier. The classifier compares accepted packets only through $S, W, Q, R, I, H, C, P, N$ and has no coordinate for trichotomy, host real order completeness, quotient-stream equality, open-cover compactness, or a selected infinite subsequence.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceTasteGate single carrier alignment`

Theorem 1415.2 (Bounded real sequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier $B = (S, W, Q, R, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, finite source-window coverage through S and W, regular rational readback through Q, real-name sealing through R, closed-bounded interval-net containment through I, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceCarrier semantic name certificate`

Proof. Project the carrier of Definition 1415.1. The only sequence data available to the classifier are the inspected index schedule S, the corresponding $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the downstream $\mathsf{Real}^{\uparrow}$ seal R.

Read the bound row next. The $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet I supplies the finite interval-net containment surface from chapter 1250; boundedness is therefore a displayed row tied to the inspected windows, not an ambient completeness or order-trichotomy principle.

Finally inspect the structural rows. The rows H, C, P, N transport, replay, source, and name only the displayed sequence-window, regular-readback, real-seal, and interval-net rows. Since no other coordinate is present in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 1415.3 (Bounded real sequence finite-window bound)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier preserves boundedness under finite window inspection: any displayed route from an inspected index to a* $\mathsf{Real}^{\uparrow}$ *sequence-name seal factors through*

$$S \longmapsto W \longmapsto Q \longmapsto R \longmapsto I$$

together with H, C, P, N. Thus a finite-window consumer can read the closed-bounded interval-net bound without importing trichotomy, host completeness, quotient-stream equality, open-cover compactness, or a selected subsequence.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceFiniteWindow bound`

Proof. Begin with an inspected index row in S. The continuation row C exposes its $\mathsf{StreamName}^{\uparrow}$ window in W, then the regular rational readback Q, and only then the $\mathsf{Real}^{\uparrow}$ seal R.

The bound row I is read after this route. It is the finite interval-net containment row supplied by chapter 1250, so the bound is attached to the displayed windows and readbacks rather than to a host-level set of all sequence values.

Transport through H and provenance through P preserve the same route, while N names it locally. Because the carrier has no coordinate for trichotomy, host completeness, quotient equality, open-cover compactness, or selected subsequences, finite-window boundedness cannot leave the displayed path. □

Theorem 1415.4 (Bounded real sequence Bolzano–Weierstrass handoff)**.** *The* BolzanoWeierstrass$^{\uparrow}$ *consumer of chapter 1254 reads a bounded sequence source from an accepted* BoundedRealSequence$^{\uparrow}$ *carrier only through the finite route*

$$S, W, Q, R, I, H, C, P, N.$$

Consequently the handoff supplies a reusable bounded real sequence certificate for the L10 real-completion route while leaving open-cover compactness, trichotomy, host completeness, quotient-stream equality, and choice-selected subsequences outside the exported surface.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceBolzanoWeierstrass handoff`

Proof. Use Definition 1415.3. It places every inspected sequence value on the route from S through W, Q, R to the interval-net bound row I.

Now read the BolzanoWeierstrass$^{\uparrow}$ carrier of chapter 1254. Its bounded sequence row is exactly the source surface needed before finite interval nets, regular Cauchy subsequence extraction, RegSeqRat readback, and the Real cluster seal can be consumed.

The structural rows H, C, P, N preserve the finite route across the handoff. Thus the downstream consumer receives a bounded real sequence certificate and receives no additional compactness, trichotomy, host completeness, quotient-stream equality, or arbitrary subsequence selection principle. □

Theorem 1415.5 (Bounded real sequence forward Bolzano route)**.** *Every* BolzanoWeierstrassRate$^{\uparrow}$ *consumer reading an accepted* BoundedRealSequence$^{\uparrow}$ *carrier must first pass through finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, the* ClosedBoundedIntervalNet$^{\uparrow}$ *bound row, and the terminal* Real$^{\uparrow}$ *seal named by the source packet. Only after those rows are exposed may the consumer inspect the finite-window rate route of chapter 8827.*

Proof. Start with Definition 1415.4. It exposes the bounded source through $S, W, Q, R, I, H, C, P, N$ before any Bolzano–Weierstrass-facing consumer reads the sequence data.

The rate consumer narrows that same source route. Its finite windows are the StreamName$^{\uparrow}$ row W, its rational readback is Q, its boundedness witness is the ClosedBoundedIntervalNet$^{\uparrow}$ row I, and its terminal real-name data is the Real$^{\uparrow}$ seal row R.

Consequently the BolzanoWeierstrassRate$^{\uparrow}$ packet may inspect only the displayed finite source rows. It receives no selected subsequence, countable choice schedule, quotient stream equality, trichotomy decision, host compactness theorem, or ambient Real$^{\uparrow}$ completeness claim from the bounded-sequence source. □

Falsifiable prediction: Every BolzanoWeierstrass$^{\uparrow}$ bounded-source read factors through a BoundedRealSequence$^{\uparrow}$ route displaying finite stream windows, regular readback, a RealUp seal, and a closed-bounded interval-net row.

Independence witness: streamname, regseqrat, real, closedboundedintervalnet, bolzanoweierstrass: BoundedRealSequence$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite sequence schedule, stream windows, regular rational readback, real seals, interval-net boundedness, transport, continuation, provenance, and local naming.

Theorem 1415.6 (Bounded real sequence window envelope)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier exposes a bounded sequence to downstream consumers only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound ledger attached to those windows. In particular, any consumer that reads an inspected index must factor through the visible route*

$$S, W, Q, R, I, H, C, P, N$$

and cannot extract an infinite stream, quotient sequence equality, host supremum, open-cover compactness proof, or selected subsequence from the carrier.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceWindow envelope`

Proof. Project the carrier of Definition 1415.1. The inspected source schedule S and finite windows W are the only entry rows for sequence data, so no downstream read starts from an unbounded stream.

The bound is then read through the displayed route to I. The regular readback Q and real seal R connect each visible window to the closed-bounded interval-net row, while the dyadic ledger named by the sibling tolerance surface records the finite comparison data needed for the bound.

Rows H, C, P, N transport, replay, source, and name the same route. Since the absent rows are not part of the carrier, the envelope remains a finite-window and finite-ledger surface. □

Theorem 1415.7 (Bounded real sequence Bolzano–Weierstrass frontier)**.** *Any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$*-facing consumer that receives data from an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier receives only the bounded-window frontier*

$$S, W, Q, R, I, H, C, P, N.$$

The frontier supplies finite bounded real sequence inspection data before the Bolzano–Weierstrass route opens, but it does not provide a selected subsequence, a compactness principle, trichotomy, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceBolzano frontier`

Proof. Use the handoff route of Definition 1415.4. That theorem identifies the bounded sequence source read by the Bolzano–Weierstrass consumer with the same finite carrier rows.

The frontier is bounded because I is displayed after S, W, Q, R. Thus the consumer can inspect finite sequence windows, read them back through $\mathsf{RegSeqRat}^{\uparrow}$, and see the real-seal and interval-net bound rows before any compactness-facing construction is considered.

No selected subsequence or compactness proof is stored in those rows. The frontier therefore hands off bounded-window data only, leaving all choice, quotient, trichotomy, and completeness principles outside the exported surface. □

Theorem 1415.8 (Bounded real sequence RegSeqRat readback)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, each* $\mathsf{RegSeqRat}^{\uparrow}$ *readback available to a consumer is routed through the same finite bound and real-seal rows:*

$$W \longmapsto Q \longmapsto R \longmapsto I.$$

Consequently a readback consumer cannot detach the regular rational data from the visible $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row, or the local structural rows* H, C, P, N.

Proof. Begin with an exposed window W_i for an inspected index. By the carrier definition, the matching regular rational readback is the corresponding component of Q rather than a host-level sequence object.

The readback continues to the real seal R_i and then to the interval-net bound row I. This is the same visible finite bound route used by Definition 1415.3.

The structural rows keep the readback inside the packet. Transport by H, replay by C, provenance by P, and naming by N prevent the regular rational data from escaping as a quotient stream, selected subsequence, compactness witness, or ambient real-completeness datum. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceRegSeqRat readback`

Theorem 1415.9 (Bounded real sequence located-bound handoff)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet whose interval-net row is read as located lower and upper real bounds factors through the* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ *route of chapter 2183 before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer inspects it. The visible route is*

$$S, W, Q, R, I \longmapsto \mathsf{LocatedBoundedRealSequence}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the closed-bounded interval-net row is narrowed to located lower and upper bound data before the Bolzano–Weierstrass handoff, without exporting compactness, trichotomy, quotient sequence equality, countable choice, or a selected subsequence.

Proof. Project an accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$ from Definition 1415.1. The sequence source is displayed only through inspected indices S, stream windows W, regular rational readback Q, and the downstream $\mathsf{Real}^{\uparrow}$ seal R.

Use Definition 1415.3 and Definition 1415.8. They put every visible sequence value on the route from W through Q, R to the bound row I, so the lower and upper endpoint reads of I are tied to the same finite windows.

When I is read as located lower and upper real bounds, the narrower sibling route is exactly the $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ packet of chapter 2183. The structural rows H, C, P, N preserve that route before Definition 1415.4 lets any $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer inspect the bounded source. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.LocatedBoundHandoff.BoundedRealSequenceLocatedBound handoff`

Theorem 1415.10 (Bounded real sequence located-window choice boundary)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reading an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier must obtain located lower and upper bound windows through the displayed route*

$$S, W, Q, R, I.$$

No such consumer may choose a subsequence, ambient bound, quotient sequence, or host completion witness outside the carrier before the stream windows, regular readback, real seal, and interval-net bound row have been displayed.

Proof. Begin with the finite-window bound of Definition 1415.3. It places each inspected sequence value on the route from S through W, Q, R to the interval-net bound row I.

Use the RegSeqRat readback and located-bound handoff, Definition 1415.8 and Definition 1415.9. They tie the located lower and upper endpoint reads of I to the same finite stream windows and regular rational rows.

Finally apply the obligation-closure package of Definition 1415.12. The package exports only $S, W, Q, R, I, H, C, P, N$, so Bolzano–Weierstrass and real-completion consumers cannot introduce an outside subsequence choice, ambient bound, quotient sequence, or host completion witness. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceLocatedWindowChoice`

Theorem 1415.11 (Bounded real sequence location induction)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

be an accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet whose located-bound row* I*,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-net row,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* R *are displayed. Then every inspected finite prefix admits induction over its window positions: the empty prefix preserves the same located bound row* I*, and each successor read factors through the same route*

$$S, W, Q, R, I, H, C, P, N$$

so the new visible value remains inside the same located interval and real-seal rows. The induction exports no subsequence selection, real trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$. The inspected schedule S and stream windows W are finite, so induction is over the displayed prefix positions rather than over an infinite host stream.

For the empty prefix, read only the already displayed located-bound row I. No sequence value has been inspected, and the same $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ row remains the active bound.

For the successor step, use Definition 1415.3 and Definition 1415.8. The next window read goes through W to Q, then to the $\mathsf{Real}^{\uparrow}$ seal R, and then back to the same interval-net row I.

The located-bound handoff of Definition 1415.9 keeps the lower and upper endpoint reads attached to this route. Transport, replay, provenance, and naming through H, C, P, N preserve the induction invariant, and the carrier has no coordinate for the excluded subsequence, trichotomy, quotient-equality, or completeness principles. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier location induction`

Theorem 1415.12 (Bounded real sequence obligation-closure package)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, the carrier row of Definition 1415.1, the NameCert obligation surface of Definition 1415.2, the finite-window bound of Definition 1415.3, the Bolzano–Weierstrass handoff of Definition 1415.4, the window envelope of Definition 1415.6, the Bolzano frontier of Definition 1415.7, the RegSeqRat readback of Definition 1415.8, and the located-window choice boundary of Definition 1415.10 assemble the obligationClosure surface over*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont,

The assembled package exports exactly the finite bounded-sequence route $S, W, Q, R, I, H, C, P, N$ *and keeps open-cover compactness, trichotomy, host completeness, quotient stream equality, and choice-selected subsequences outside the packet.*

Proof. Start with the carrier and NameCert obligations. The carrier definition lists the finite source schedule, stream windows, regular readback, real seal, closed-bounded interval-net row, and the structural rows. The obligation theorem identifies those rows as the complete local certificate

surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Next use the finite-window bound and handoff theorems. They place every inspected sequence value on the displayed route from S through W, Q, R to I, and they show that the $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer reads the bounded source only through the same packet.

Finally add the envelope, frontier, and readback rows. These theorems keep downstream reads inside the visible route $S, W, Q, R, I, H, C, P, N$ and prevent the regular rational data from detaching from the bounded-window surface. Since none of the cited rows stores open-cover compactness, trichotomy, host completeness, quotient stream equality, or a choice-selected subsequence, the assembled obligation-closure package exports none of those claims. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceObligation closure package`

Theorem 1415.13 (Bounded real sequence carrier admission)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier*

$$B = (S, W, Q, R, I, H, C, P, N)$$

must expose inspected sequence indices S, finite $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{Real}^{\uparrow}$ *sequence seals R, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row I, componentwise transport H, displayed replay C, package provenance P, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row N before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer can read the bounded source. The admitted surface exports no open-cover compactness, trichotomy, quotient stream equality, or choice-selected subsequence.*

Proof. Project the carrier of Definition 1415.1. Its displayed tuple is exactly $S, W, Q, R, I, H, C, P, N$, so carrier admission starts with inspected indices, finite windows, regular readback, real sequence seals, the interval-net bound row, and the structural rows.

Use the NameCert obligations of Definition 1415.2. They identify those displayed rows as the complete local certificate surface and prevent a consumer from reading a bounded source before the windows, readback, seal, bound, transport, replay, provenance, and naming rows have appeared.

Apply the finite-window bound and Bolzano–Weierstrass handoff, Definition 1415.3 and Definition 1415.4. The first ties each inspected value to the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ bound row, and the second forces every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ consumer to enter through the same bounded-source route.

Finally use the obligation-closure package of Definition 1415.12. Since that package contains no open-cover compactness, trichotomy, quotient stream equality, or choice-selected subsequence coordinate, the admitted carrier exports none of those objects to a Bolzano–Weierstrass or real-completion consumer. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceObligation carrier admission`

Definition 1415.14 (Bounded real sequence finite-window net)**.** *For an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier*

$$B = (S, W, Q, R, I, H, C, P, N),$$

the finite-window net packet is the displayed subroute

$$\mathcal{N}_B = (S, W, Q, R, I, H, C, P, N)$$

read as a closed-bounded interval-net source. Its entries are the inspected sequence schedule S, the synchronized $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{Real}^{\uparrow}$ *seal row R, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row I, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row N. The packet has no coordinate for an infinite subnet, open-cover compactness, trichotomy, quotient sequence equality, countable choice, or a selected Bolzano–Weierstrass subsequence.*

Lean4 (definition) `BEDC.Derived.BoundedRealSequenceUp.FiniteWindowNet.BoundedRealSequenceFiniteWindowNet`

Theorem 1415.15 (Bounded real sequence finite-window net extraction)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier exposes the finite-window net packet of Definition 1415.14 before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reads the bounded source. In particular, every such consumer factors through*

$$S, W, Q, R, I, H, C, P, N$$

and receives the closed-bounded interval-net route before it can inspect compactness-facing or completion-facing data.

Proof. Project the accepted carrier and form the packet $\mathcal{N}_B$ from the same rows. The carrier definition supplies $S, W, Q, R, I, H, C, P, N$ in exactly the order needed to read inspected windows, regular readback, real seals, and the interval-net bound row.

Apply Definition 1415.3. It identifies the finite bound route from S through W, Q, R to I and keeps the route outside trichotomy, host completeness, quotient equality, open-cover compactness, and selected subsequences.

Apply Definition 1415.4. The Bolzano–Weierstrass consumer reads its bounded sequence source only through the same visible rows. The structural rows H, C, P, N then repack transport, replay, provenance, and local naming as the finite-window net packet. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier finite window net extraction`

Theorem 1415.16 (Bounded real sequence regular-readback stability)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

and

$$B' = (S', W', Q', R', I', H', C', P', N')$$

be accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies the source schedules, stream windows, regular readback rows, real seals, and interval-net bound rows S, W, Q, R, I with S', W', Q', R', I', then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the closed-bounded interval-net bound row are preserved. Thus the transported packet still exposes the same finite-window net route before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or real-completion consumer reads it.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier regular readback stability`

Proof. Project both carriers. The hypothesis says that the five analytic rows S, W, Q, R, I and S', W', Q', R', I' agree under the componentwise transport recorded by the H rows.

Use Definition 1415.8. The readback available to a consumer is attached to the visible route $W \mapsto Q \mapsto R \mapsto I$, so transporting W, Q, R, I preserves the regular rational readback and the bound row together.

Use Definition 1415.15 to repack the transported rows as the finite-window net packet. The remaining rows H, C, P, N and H', C', P', N' supply the transport, replay, provenance, and naming data, so no detached compactness, quotient, choice, trichotomy, or completeness coordinate is introduced. □

Theorem 1415.17 (Bounded real sequence dyadic-bound transport)**.** *Let*

$$B = (S, W, Q, R, I, H, C, P, N)$$

be an accepted $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier. Every boundedness read transported from* B *to a* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or completion-facing consumer factors through finite stream windows* S, W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row* I*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealSequence}^{\uparrow}}$ *row.*

The transported read may carry dyadic lower and upper bound ledgers from the finite window, but it cannot add trichotomy, quotient sequence equality, countable choice, a choice-selected subsequence, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier dyadic bound transport`

Proof. Project the carrier and read the finite-window net packet of Definition 1415.14. The packet lists $S, W, Q, R, I, H, C, P, N$ as the complete boundedness route.

Apply Definition 1415.16. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and interval-net bound row are stable under componentwise transport, so the transported boundedness row remains attached to the same finite stream windows.

Finally use Definition 1415.15. The Bolzano–Weierstrass or completion-facing consumer receives the closed-bounded interval-net source only after S, W, Q, R, I have been displayed, and the structural rows replay the same route without adding any excluded selector or completeness coordinate. □

Theorem 1415.18 (Bounded real sequence window factorization)**.** *For every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *carrier, any consumer read of boundedness factors through a finite window prefix before it reaches the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *source. More precisely, the read first inspects the sequence schedule* S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, then the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, and only then the interval-net bound row* I*.*

No consumer may bypass this route by asking for an infinite stream object, a quotient sequence equality, countable choice, a selected subsequence, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $B = (S, W, Q, R, I, H, C, P, N)$. The carrier makes the inspected schedule and finite stream windows available before any real readback or interval bound can be consumed.

Use Definition 1415.3. It identifies the bound route as a finite passage from S through W, Q, R into the closed-bounded interval-net row I.

The structural rows H, C, P, N then transport, replay, source, and name the same finite passage. Since the carrier has no infinite stream, quotient equality, choice, trichotomy, open-cover, selected-subsequence, or ambient-completeness coordinate, the displayed window factorization is exhaustive. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier window factorization`

Theorem 1415.19 (Bounded real sequence window localization)**.** *Every accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet localizes each inspected finite* $\mathsf{StreamName}^{\uparrow}$ *window through the displayed* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer may read the bounded source. More precisely, for*

$$B = (S, W, Q, R, I, H, C, P, N)$$

each inspected window in W is read through

$$S, W \longmapsto Q \longmapsto R \longmapsto I \longmapsto H, C, P, N.$$

The localization exports finite window bounds and regular readback only; it exports no host set of all sequence values, no choice-selected subsequence, no quotient stream equality, no open-cover compactness, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier window localization`

Proof. Project the carrier and use the finite-window net extraction of Definition 1415.15. That result displays $S, W, Q, R, I, H, C, P, N$ as the finite net packet a consumer must read before bounded-source data are available.

Apply regular-readback stability and dyadic-bound transport, Definition 1415.16 and Definition 1415.17. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ row Q attached to the same stream window W, real seal R, and closed-bounded interval-net row I under componentwise transport.

Finally use the window factorization of Definition 1415.18. It forces every boundedness read to pass through the inspected finite window before reaching the interval-net row, so a Bolzano–Weierstrass consumer receives only the displayed local route and none of the excluded host-level data. □

Theorem 1415.20 (Bounded real sequence Bolzano source exactness)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *source read from an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet is exactly the finite-window source*

$$S \to W \to Q \to R \to I \to H \to C \to P \to N.$$

Thus the Bolzano–Weierstrass consumer receives a bounded source only through finite stream windows, regular rational readback, the real-name seal, and the closed-bounded interval-net row; it receives no selected subsequence, no compactness theorem outside the displayed interval-net route, and no ambient real-completeness principle.

Proof. Read the accepted carrier and form the finite-window net packet. The rows S, W, Q, R, I display the source schedule, finite stream names, regular rational readback, real seal, and closed-bounded interval-net bound before the consumer is allowed to read the source.

Apply Definition 1415.4. That handoff sends Bolzano–Weierstrass consumers to the visible bounded source rather than to a choice-selected subsequence or host compactness theorem.

Use Definition 1415.17 to keep the dyadic bound ledger attached to the same source route under transport. The rows H, C, P, N replay, source, and name the handoff, so the source exactness statement has no remaining coordinate for excluded choice, quotient, compactness, trichotomy, or completeness principles. □

Lean4 ✓ `BEDC.Derived.BoundedRealSequenceUp.BoundedRealSequenceCarrier bolzano source exactness`

Theorem 1415.21 (Bounded real sequence compact-interval subsequence boundary)**.** *Every Bolzano–Weierstrass or compact-interval consumer of an accepted* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet must read finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seals, and* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound rows before any subsequence-facing claim is visible. The packet exposes the finite route*

$$S \to W \to Q \to R \to I \to H \to C \to P \to N$$

and does not export a selected subsequence, open-cover compactness, quotient sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-window bound of Definition 1415.3. It ties every inspected sequence value to the route from S through W, Q, R into the closed-bounded interval-net row I.

Apply the Bolzano–Weierstrass handoff and the window envelope, Definition 1415.4 and Definition 1415.6. They require a Bolzano-facing consumer to read the bounded source through the same finite window rows rather than through a selected subsequence or a host compactness principle.

Use Bolzano source exactness, Definition 1415.20. This identifies the visible source as the finite path through $S, W, Q, R, I, H, C, P, N$.

Finally apply finite-window net extraction, Definition 1415.15. The interval-net packet supplies the compact-interval-facing bound row while the structural rows transport, replay, source, and name the same route, leaving no coordinate for quotient equality, countable choice, open-cover compactness, selected subsequences, trichotomy, or ambient completeness. □

Theorem 1415.22 (Bounded real sequence bridge-facing route)**.** *The public bounded-source export for* BoundedRealSequence$^{\uparrow}$ *is bridge-facing exactly through the checked finite-window closure package and the Bolzano–Weierstrass source exactness route. A bridge consumer must first read Definition 1415.12 to expose the rows $S, W, Q, R, I, H, C, P, N$ and then read Definition 1415.20 before the bounded source can be handed to a standard bridge comparison. Thus the paper bridge surface is the finite-window route grounded by* BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequence *not a selected subsequence, open-cover compactness proof, quotient stream equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Use the obligation closure package. It gathers carrier admission, finite-window bounds, regular readback, dyadic transport, window factorization, and source handoff into one checked NameCert surface.

Now apply the Bolzano source exactness theorem. It identifies the only public export as the finite route

$$S \to W \to Q \to R \to I \to H \to C \to P \to N.$$

A bridge comparison can therefore inspect exactly those displayed rows before reading the bounded source. Since neither cited route exposes selected subsequences, quotient stream equality, open-cover compactness, or ambient completeness, the bridge-facing paper surface is exhausted by the checked closure package and the Bolzano source exactness route. □

Closure status for NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$**.**

Constructive story: A BoundedRealSequence$^{\uparrow}$ packet is generated from inspected sequence indices, finite StreamName windows, RegSeqRat readback, RealUp sequence-name seals, a ClosedBoundedIntervalNet bound row, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1415.1, Definition 1415.2, Definition 1415.12, Definition 1415.20, and Definition 1415.22 expose the finite-window bounded-sequence source as a bridge-facing route for Bolzano–Weierstrass consumers over StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedIntervalNet$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{BoundedRealSequence}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.BoundedRealSequenceUp.NameCertObligations.BoundedRealSequenceObligation\_closure\_packa`

Bridge status: none

Not claimed: This chapter does not close bridge checking, open-cover compactness, trichotomy, host completeness, quotient equality of streams, or choice-selected subsequences.

Upgrade path: A checked standard bridge would turn the bridge-facing finite-window route of Definition 1415.22 into bridgeChecked verification.

Chapter 1416

A Concrete Naming Certificate for $\mathsf{DyadicFloor}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicFloor}^{\uparrow}$ packet records an explicit dyadic lower grid point and its bounding successor as finite BEDC data. It sits over the dyadic rational arithmetic surface of chapter 291, the generic precision schedule of chapter 292, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet is the finite floor-bounding primitive used by later regular-Cauchy and Real seal arguments before any consumer may appeal to host floor, trichotomy, quotient equality, choice, or an ambient Archimedean decision procedure.

Definition 1416.1 (DyadicFloor carrier)**.** *A* $\mathsf{DyadicFloor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (x, k, d, s, \ell, u, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *x is the source real-facing request, represented only through the already displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows consumed by the packet.*
2. *k is the requested dyadic scale.*
3. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid point claimed as the lower dyadic floor candidate at scale k.*
4. *s is the dyadic successor row $d + 2^{-k}$, recorded as a second* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row rather than as host integer arithmetic.*
5. *ℓ is the lower-bound evidence that the request is not below d at the displayed scale.*
6. *u is the upper-window evidence that the request is below the successor row s at the same displayed scale.*
7. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule that supplies a finite observation threshold sufficient to read the two displayed inequalities.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row exposing the regular rational window used by M.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic floor rows and regular handoff have been displayed.*
10. *H records componentwise* hsame *transport for the request, scale, lower grid point, successor row, lower and upper evidence, modulus schedule, regular handoff, real seal, continuation, provenance, and local naming rows.*

11. *C records the displayed* Cont *replay from a precision request through $k, d, s, \ell, u, M, R, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{DyadicFloor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicFloor}^{\uparrow}}$ *row.*

The carrier has no coordinate for host floor, integer trichotomy, quotient real equality, countable choice, a selected representative, hidden Archimedean search, or ambient completeness.

Lean4 (definition) `BEDC.Derived.DyadicFloorUp.DyadicFloorUp`

Definition 1416.2 (DyadicFloor classifier)**.** *The local* $\mathsf{DyadicFloor}^{\uparrow}$ *classifier accepts exactly those carrier rows $F = (x, k, d, s, \ell, u, M, R, E, H, C, P, N)$ for which the displayed successor row s is the one-step dyadic successor of d at scale k, the lower evidence ℓ reads d as a lower bound for the request at that scale, the upper evidence u reads s as the strict upper bounding row at that scale, and the modulus schedule M supplies the finite regular-rational window through R used by both comparisons. Transport by H and replay by C preserve the same rows componentwise; they do not add a second classifier, host comparison, or detached real equality test.*

Lean4 (definition) `BEDC.Derived.DyadicFloorUp.DyadicFloorClassifier`
Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorClassifier window obligations`

Theorem 1416.3 (DyadicFloor NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicFloor}^{\uparrow}}$ *surface for Definition 1416.1 consists exactly of carrier exposure for F, classifier exposure from Definition 1416.2, dyadic lower-grid exposure through d, dyadic successor exposure through s, lower-bound exposure through ℓ, upper-window exposure through u, modulus exposure through M, regular rational readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier tuple $F = (x, k, d, s, \ell, u, M, R, E, H, C, P, N)$. The definition lists every analytic and structural coordinate exposed to the local naming row.

The classifier reads only the successor relation between d and s, the lower evidence ℓ, the upper evidence u, and the finite observation route supplied by M and R. Thus the dyadic floor obligation is not a hidden call to a host floor operation.

The remaining rows preserve the same packet. The real seal E is downstream of the dyadic and regular rows, while H, C, P, N transport, replay, source, and name the displayed data. Since no excluded object is a coordinate of F, the listed rows exhaust the seed obligation surface. □

Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorCarrier namecert obligations`

Theorem 1416.4 (DyadicFloor bounding interval)**.** *Every accepted* $\mathsf{DyadicFloor}^{\uparrow}$ *packet $F = (x, k, d, s, \ell, u, M, R, E, H, C, P, N)$ exposes a finite dyadic interval*

$$d \leq_k x <_k s$$

at the displayed scale k, where the left comparison is read by ℓ, the right comparison is read by u, and s is the dyadic successor row $d + 2^{-k}$. No consumer receives a host floor value, trichotomy decision, quotient equality of real names, selected representative, countable-choice row, or hidden Archimedean search.

Proof. Start from the classifier of Definition 1416.2. Its acceptance condition names s as the one-step dyadic successor of d at the requested scale k.

The same classifier binds the lower and upper comparisons. The row ℓ supplies the displayed lower-bound read for d, while u supplies the displayed strict upper-window read for the successor s.

Finally inspect the carrier surface. The only comparison rows available to a consumer are d, s, ℓ, u together with the modulus and regular readback route M, R. Hence the bounding interval is the finite packet interval above, and the excluded host operations have no route into the accepted packet. □

Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorBoundingInterval`

Theorem 1416.5 (DyadicFloor RegSeqRat handoff)**.** *For every accepted* $\mathsf{DyadicFloor}^{\uparrow}$ *packet, the floor-bounding read factors through the finite route*

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the modulus schedule M selects the regular rational window R, the dyadic rows d, s, ℓ, u read their comparisons from that window, and the terminal seal E is available only after those rows have been displayed.

Proof. Use the carrier of Definition 1416.1. The schedule row M is the only threshold source in the packet, and the handoff row R is the only regular rational readback row.

The dyadic comparisons are then forced through the same finite readback. Both the lower row ℓ and the upper row u classify the request using d, s at scale k after M and R have supplied the observation window.

The real seal is terminal in this route. The row E is reached only after the dyadic and regular rows are displayed, and the structural rows H, C, P, N merely preserve and name the same route. Thus no detached seal, quotient equality, selected representative, or choice-driven search is available to the consumer. □

Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorRegSeqRatHandoff`

Theorem 1416.6 (DyadicFloor finite-window obligations)**.** *Every accepted* $\mathsf{DyadicFloor}^{\uparrow}$ *packet must expose the displayed rows*

$$k, d, s, \ell, u, M, R$$

before any floor-bounding read is available. The scale k, lower dyadic row d, successor row s, lower evidence ℓ, upper evidence u, modulus schedule M, and regular readback row R are therefore finite observation obligations, not optional annotations on a later host floor value.

Proof. Read the carrier tuple of Definition 1416.1. The rows k, d, s, ℓ, u, M, R are explicit coordinates, and the classifier of Definition 1416.2 accepts a packet only when these coordinates support the successor and bounding comparisons.

The bounding interval theorem then consumes exactly those rows. The lower read uses d and ℓ, the upper read uses s and u, and both reads are routed through the finite schedule and regular-rational window supplied by M and R.

Thus a consumer cannot reach the floor-bounding interval through a hidden host operation. Omitting any listed row removes the finite observation route required by the accepted packet. □

Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorCarrier window obligations`

Theorem 1416.7 (DyadicFloor Real-seal nonescape)**.** *In an accepted* $\mathsf{DyadicFloor}^{\uparrow}$ *packet, the terminal* E *row is reachable only after the dyadic scale row, successor row, comparison rows, modulus schedule, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed. Consequently the packet gives no route to a host floor, selected representative, trichotomy decision, quotient* $\mathsf{Real}^{\uparrow}$ *equality, or hidden Archimedean search.*

Proof. Follow the handoff route of Definition 1416.5. The modulus row M supplies the finite observation threshold, and the regular readback row R is the only source from which the dyadic comparison rows are read.

The terminal seal E comes after this readback. Since the carrier has no earlier Real-seal coordinate and no coordinate for the excluded host operations, a consumer cannot escape from the displayed dyadic and regular-rational route into a detached real comparison.

The transport, replay, provenance, and local naming rows preserve that order. They name the same finite route rather than adding a quotient Real equality, selected representative, or Archimedean search. □

Lean4 ✓ `BEDC.Derived.DyadicFloorUp.DyadicFloorCarrier real seal nonescape`

Falsifiable prediction: An accepted $\mathsf{DyadicFloor}^{\uparrow}$ consumer that reads a floor-bounding interval without the displayed scale k, lower dyadic row d, successor row s, lower evidence ℓ, upper evidence u, modulus schedule M, and regular readback row R refutes the carrier surface.

Independence witness: dyadicratcore, cauchymodulus, regseqrat, real: no listed sibling is bijective to $\mathsf{DyadicFloor}^{\uparrow}$ because this packet jointly binds a scale, lower grid point, dyadic successor, two bounding comparisons, modulus schedule, regular readback, terminal seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicFloor}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicFloor}^{\uparrow}$ packet is generated from a real-facing request, a dyadic scale, a lower dyadic grid point, its dyadic successor, lower and upper bounding evidence, a finite modulus schedule, a regular rational readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicFloor}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1416.1, Definition 1416.2, Definition 1416.3, Definition 1416.4, Definition 1416.5, Definition 1416.6, and Definition 1416.7 bind the carrier, classifier, and finite dyadic floor-bounding obligation surface for the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicFloor}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.DyadicFloorUp.DyadicFloorCarrier\_real\_seal\_nonescape`

Bridge status: none

Not claimed: This chapter does not close host floor, trichotomy, quotient real equality, countable choice, selected representatives, ambient completeness, or an Archimedean search theorem.

Upgrade path: A bridge route can connect the checked finite dyadic floor-bounding theorem to the standard acceptance schema.

Chapter 1417

A Concrete Naming Certificate for $\mathsf{DecimalExpansion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DecimalExpansion}^{\uparrow}$ packet records finite decimal-prefix windows as BEDC readback data. It does not store a host infinite decimal, a quotient real, or a selected representative. Its role is to route base-ten digit observations through the finite stream surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic and rational place-value ledgers of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

The chapter is the decimal sibling of chapter 418 and chapter 342. The shared discipline is finite-window readback: binary prefixes, continued-fraction convergents, and decimal prefixes all expose only displayed rows before a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can read the packet.

The same finite decimal-prefix windows also form a forward consumer of chapter 1809. A decimal window gives a countable dense readback route into $\mathsf{Real}^{\uparrow}$, while $\mathsf{RealSeparability}^{\uparrow}$ supplies the global countable-density ledger that later real-name consumers use to compare those finite routes.

Definition 1417.1 (DecimalExpansion carrier)**.** *A* $\mathsf{DecimalExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, V, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite digit ledger, whose rows record decimal digits and the associated sign or integral-prefix convention when the local packet includes one;*
2. *W is the finite decimal-place window ledger, listing the inspected positions and the request precision served by those positions;*
3. *V is the place-value row that evaluates each inspected prefix as a rational sum with denominator a power of ten;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison row that transports the base-ten place-value read into the dyadic tolerance discipline used by downstream real consumers;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the same decimal window and place-value rows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, W, V, Q, R have been displayed;*
7. *H is the componentwise* hsame *transport row for digit, window, place-value, dyadic/rational, regular-sequence, and real-seal coordinates;*

8. *C is the* Cont *replay row that reuses the accepted finite decimal windows without adding a host stream;*
9. *P is the* Pkg *provenance over* $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$ *row.*

The carrier exposes no infinite decimal stream, Dedekind cut, quotient real, endpoint convention resolver, trichotomy decision, selected representative, or Real-completeness witness.

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.TasteGate.DecimalExpansionTasteGate single carrier alignment`

Theorem 1417.2 (DecimalExpansion digit-window obligations)**.** *Every real-facing read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must consume the digit ledger D, finite window ledger W, place-value row V, dyadic comparison row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E in that order. The structural rows H, C, P, N transport, replay, source, and name only this displayed route, so no decimal consumer can read a real-facing value before the finite digit window and dyadic comparison have both been displayed.*

Proof. Project the carrier of Definition 1417.1. Its public real-facing coordinates appear as D, W, V, Q, R, E before the structural rows H, C, P, N.

The finite-window discipline is the same one used by the sibling routes of chapter 418, chapter 342, and chapter 126. A consumer first receives inspected digits and positions, not an infinite stream.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and ledger exhaustion force every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer to inspect only displayed rows, with V and Q between the digit window and the regular-rational handoff.

Finally read the seal row E downstream of R. Since the carrier has no coordinate for a direct decimal stream, quotient real, selected representative, or completeness witness, the ordered digit-window route is the only real-facing decimal read exported by the packet. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitWindowObligations`

Theorem 1417.3 (DecimalExpansion NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$ *obligation surface consists exactly of the rows in Definition 1417.1. Carrier admission requires a finite digit ledger, a finite decimal-place window ledger, a base-ten rational place-value row, a dyadic/rational comparison row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, a* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through the same displayed rows. Ledger exhaustion says that every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer can inspect only $D, W, V, Q, R, E, H, C, P, N$.*

Proof. Carrier admission is read directly from the finite packet. The digit rows and decimal-place windows give the visible $\mathsf{StreamName}^{\uparrow}$ observation surface, while V evaluates each inspected prefix as a rational place-value sum.

The dyadic/rational row Q is the only bridge from base-ten arithmetic into the tolerance discipline already used by chapter 291. The regular handoff row R and real-seal row E are therefore downstream of the same inspected window.

Transport is componentwise because H has coordinates for each displayed row, and continuation only replays windows already accepted by C. The package and local naming rows record exactly those dependencies.

For ledger exhaustion, project a consumer read from the accepted packet. The definition gives only the ten coordinates $D, W, V, Q, R, E, H, C, P, N$, so any route through an infinite stream, quotient real, selected endpoint convention, or ambient completeness principle is outside the carrier. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionNameCertObligations`

Theorem 1417.4 (DecimalExpansion tail digit stability)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet*

$$K = (D, W, V, Q, R, E, H, C, P, N),$$

finite digit tails are stable under the displayed hsame *transport row H. A consumer that transports the accepted packet can compare tail digits only through the same finite digit ledger D, window ledger W, place-value row V, dyadic comparison row Q, regular handoff R, real seal E, replay row C, provenance row P, and local name row N.*

Proof. Project the obligation surface of Definition 1417.3. The only digit data are the displayed rows D and W, and every place-value or real-facing read is downstream of them.

The transport row H is componentwise through the same carrier coordinates. Therefore a transported tail cannot acquire a new infinite decimal stream or a new endpoint selector; it can only replay the displayed finite tail rows already present in $D, W, V, Q, R, E, C, P, N$. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionTailDigitStability`

Theorem 1417.5 (DecimalExpansion rational Real handoff)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet $K = (D, W, V, Q, R, E, H, C, P, N)$, the rational place-value row V must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison row Q, then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E is public. No consumer can read a real-valued decimal prefix directly from D, W, V.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRatRealHandoff`

Proof. Read the carrier order in Definition 1417.1. The rows list digit data, window data, place-value evaluation, dyadic/rational comparison, regular handoff, and real seal as D, W, V, Q, R, E.

Apply Definition 1417.3. Carrier admission and classifier transport preserve this order componentwise, and ledger exhaustion prevents any route from bypassing Q or R before reaching E. □

Theorem 1417.6 (DecimalExpansion nonescape)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet $K = (D, W, V, Q, R, E, H, C, P, N)$ exports exactly the finite decimal-prefix, dyadic comparison, regular handoff, real seal, transport, replay, provenance, and naming rows listed in Definition 1417.1. It exports no infinite decimal, quotient real, endpoint selector, host trichotomy row, selected representative, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionNonescape`

Proof. Use ledger exhaustion from Definition 1417.3. The accepted packet has exactly the coordinates $D, W, V, Q, R, E, H, C, P, N$, so every public read is a projection or replay of those rows.

The excluded objects would require a coordinate outside the displayed carrier: an infinite digit stream, a quotient real, a selector for endpoints, a trichotomy decision, a selected representative, or a completeness witness. Since no such coordinate is present, no accepted consumer can recover it from the packet. □

Theorem 1417.7 (DecimalExpansion RealUp readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads an inspected decimal-place window, then the digit rows on that window, then the rational place-value and dyadic tolerance rows, then the regular rational handoff, and only then the displayed real seal. The packet supplies no Dedekind cut, quotient real, endpoint uniqueness convention, trichotomy principle, selected representative, or Real-completeness witness.

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealReadbackBoundary`

Proof. Start with an accepted carrier $K = (D, W, V, Q, R, E, H, C, P, N)$. The $\mathsf{StreamName}^{\uparrow}$ observation is W together with the digit rows of D on that window, so a real-facing consumer cannot read a decimal prefix without first reading the finite window.

The place-value row V turns that same finite prefix into a rational base-ten sum. The row Q records the dyadic and rational comparison data used to pass from the decimal sum to the tolerance language consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

The handoff row R is therefore reached only after D, W, V, Q are fixed. The seal row E is listed after R, and the rows H, C, P, N preserve the same dependency route under transport, continuation, provenance, and local naming.

Thus every $\mathsf{Real}^{\uparrow}$ read factors through the displayed finite route. The excluded objects are not carrier coordinates, so no consumer of the packet can obtain them from this readback boundary. □

Theorem 1417.8 (DecimalExpansion RegSeqRat handoff)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet routes finite decimal digit rows through rational place-value rows and dyadic comparison rows before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route is exhausted by the displayed coordinates*

$$D, W, V, Q, R, E, H, C, P, N$$

of Definition 1417.1. In particular, the packet exports no quotient real, endpoint representative, trichotomy row, selected decimal stream, or $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatHandoff`

Proof. Project the carrier of Definition 1417.1. The finite digit ledger D is read together with the finite decimal-place windows W, and the place-value row V evaluates exactly those inspected prefixes as rational base-ten sums.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and classifier transport keep D, W, V attached to the same dyadic and rational comparison row Q, while ledger exhaustion prevents a consumer from bypassing the displayed handoff surface.

Finally apply the RealUp readback boundary of Definition 1417.7. The regular handoff row R is reached only after the digit, window, place-value, and dyadic rows are displayed, and the terminal seal row E remains downstream of R with the structural rows H, C, P, N preserving the same route. □

Theorem 1417.9 (DecimalExpansion window normalization obligation)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must display the finite digit ledger D and finite window ledger W before any place-value, dyadic-comparison, regular-rational, or real-seal row can be read. Equivalently, every public read of V, Q, R, E factors through the finite decimal-prefix surface named in Definition 1417.1 and constrained by Definition 1417.3.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionWindowNormalizationObligation`

Proof. Project the carrier of Definition 1417.1. The rows are ordered as $D, W, V, Q, R, E, H, C, P, N$, so digit admission and inspected-window admission are the visible source rows for the packet.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and ledger exhaustion require every consumer to read only displayed coordinates, hence no consumer can reach V, Q, R, E without first reading D and W.

The L10 sibling route is the same finite-window route already used by chapter 126, chapter 301, and chapter 38. Thus window normalization is a finite ledger obligation, not an infinite decimal-stream selection. □

Theorem 1417.10 (DecimalExpansion prefix stability)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, finite decimal prefixes are stable under* $\mathsf{StreamName}^{\uparrow}$ *window refinement and componentwise* hsame *transport. If a consumer refines the inspected window* W *without changing the displayed prefix rows in* D*, then the place-value row* V*, the dyadic tolerance row* Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *remain attached to the same prefix route before any downstream read is exposed.*

Proof. Begin with the carrier of Definition 1417.1. The digit ledger D and window ledger W are the only source rows from which a finite decimal prefix can be read.

Use the window normalization obligation of Definition 1417.9. It forces every public read of V, Q, R, E to pass through the displayed D, W prefix surface, so a window refinement that preserves the displayed prefix rows cannot change the downstream route.

The $\mathsf{StreamName}^{\uparrow}$ sibling of chapter 126 supplies the finite-window discipline, and the structural row H transports the same prefix componentwise through hsame. The dyadic and regular-rational consumers of chapter 291 and chapter 301 therefore receive the same place-value and tolerance route before the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixStability`

Theorem 1417.11 (DecimalExpansion dyadic comparison handoff)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, the place-value row* V *reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff only through the comparison row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and rational comparison surface. No direct route from a decimal place-value sum to* R *or* E *is exported by the packet.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicComparisonHandoff`

Proof. Read Definition 1417.1. The row V records the base-ten rational place-value sum, while Q is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ and rational comparison row.

Apply Definition 1417.3. The obligation surface places Q between place-value evaluation and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and classifier transport is componentwise through that same displayed row.

Finally apply Definition 1417.8. It states that finite digit rows pass through rational place-value rows and dyadic comparison rows before reaching R and the terminal seal E. The dyadic handoff therefore uses the existing routes of chapter 291 and chapter 301, with no bypass from V to R. □

Theorem 1417.12 (DecimalExpansion dyadic rational readback)**.** *Every finite decimal prefix admitted by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is read through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal is public. Concretely, the prefix rows* D, W *first determine the rational place-value row* V*, the dyadic and rational comparison row* Q *then supplies the finite*

tolerance comparison, and only then may the regular rational handoff R and terminal seal E be consumed.

Proof. Project the carrier of Definition 1417.1. The packet orders the rows as $D, W, V, Q, R, E, H, C, P, N$, so decimal digits and inspected windows are read before place values, dyadic tolerance, regular readback, and the real seal.

Apply prefix stability, Definition 1417.10. It keeps the finite prefix route fixed under $\mathsf{StreamName}^{\uparrow}$ window refinement and hsame transport, which is the only way a downstream consumer can compare refined windows.

Now use the dyadic comparison handoff of Definition 1417.11. The row Q is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ and rational comparison route, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301 cannot be reached directly from V.

The terminal real read is the seal discipline of chapter 38, and the decimal normal-form consumer of chapter 3952 reads only after this finite dyadic and regular-rational route has been displayed. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicRationalReadback`

Theorem 1417.13 (DecimalExpansion RealSeparability density route)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the finite decimal prefix route contributes to the RealSeparability-facing density surface only through displayed countable dense readback along*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff cited by chapter 1809 reads $D, W, V, Q, R, E, H, C, P, N$ as finite prefix, dyadic comparison, regular-rational, seal, transport, replay, provenance, and naming rows. It exports no quotient real, selected endpoint convention, trichotomy row, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSeparabilityDensityRoute`

Proof. Project the carrier of Definition 1417.1. The digit row D and window row W provide the only prefix data, while V, Q, R, E place that prefix into rational place value, dyadic tolerance, regular rational readback, and the terminal real seal.

Apply Definition 1417.12. The theorem forces the readback route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before E is exposed, so the density consumer can only see a displayed finite route.

Finally read the RealSeparability sibling of chapter 1809. Its countable dense surface consumes finite readback data; the decimal packet supplies that surface through $D, W, V, Q, R, E, H, C, P, N$ and has no coordinate for quotient reals, selected endpoints, trichotomy, or real completeness. □

Theorem 1417.14 (DecimalExpansion countable dense consumer handoff)**.** *Every countable dense consumer that reads an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$ *must first pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the digit ledger D, the place-value row V, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The consumer handoff to chapter 1809 is therefore a displayed dense readback route, not a decimal stream selector, quotient-real representative, endpoint convention, trichotomy decision, or completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionCountableDenseConsumerHandoff`

Proof. Use Definition 1417.13. It identifies the only RealSeparability-facing route exported by the packet as the finite path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The structural rows then keep that route componentwise. Transport by H, replay by C, provenance by P, and the local name N certify the same displayed rows without inserting an ambient sequence selector or quotient-real comparison.

Thus a countable dense consumer receives exactly the finite prefix readback route already present in D, W, V, Q, R, E. Since the carrier has no selected decimal stream, endpoint convention, trichotomy, or completeness coordinate, none can be exported by the handoff. □

Theorem 1417.15 (DecimalExpansion prefix density exhaustion)**.** *For an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, all RealSeparability-facing density reads are exhausted by finite decimal prefixes and their displayed readback through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Once* $D, W, V, Q, R, E, H, C, P, N$ *have been read, the packet has no remaining row from which to choose a quotient real, select an endpoint convention, decide trichotomy, or assert a* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixDensityExhaustion`

Proof. Begin with the countable dense consumer handoff of Definition 1417.14. It routes every density consumer through the finite prefix ledger and the dyadic, regular-rational, and real-seal rows.

Apply the excluded-coordinate clause of Definition 1417.1. The packet lists only $D, W, V, Q, R, E, H, C, P, N$, so there is no extra coordinate after the terminal seal and structural rows have been consumed.

The RealSeparability reference in chapter 1809 therefore receives a countable dense readback route from finite decimal prefixes. It does not receive quotient real equality, selected endpoint data, trichotomy, or a completeness row from $\mathsf{DecimalExpansion}^{\uparrow}$. □

Theorem 1417.16 (DecimalExpansion dyadic normalization lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet passes through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *digit-place tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row before any real-facing decimal value is consumed. In particular, the digit-place route is the same L10 sibling lattice exposed by chapter 291, chapter 301, chapter 126, and chapter 38; the packet exports no direct decimal-to-real shortcut.*

Proof. Project the carrier of Definition 1417.1. The digit rows D and finite decimal windows W feed the place-value row V, while the comparison row Q is the only displayed dyadic and rational tolerance route.

Apply Definition 1417.11. It places Q between every place-value read and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, so the decimal packet cannot jump from V directly to the seal row.

Finally apply Definition 1417.7. The terminal real read is reached only after finite windows, digit rows, dyadic tolerance, and regular rational readback have been displayed. This locks the decimal route to the dyadic, regular-Cauchy, stream-window, and real-seal sibling lattice named in the statement. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDyadicNormalizationLock`

Theorem 1417.17 (DecimalExpansion Real seal nonescape)**.** *The terminal row* E *of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet exports only the displayed* $\mathsf{Real}^{\uparrow}$ *seal reached from* D, W, V, Q, R*. It exports no quotient real equality, endpoint resolver, trichotomy decision, selected digit stream, selected decimal stream, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with the real readback boundary of Definition 1417.7. Every $\mathsf{Real}^{\uparrow}$ read factors through the finite route from $\mathsf{StreamName}^{\uparrow}$ windows to decimal place value, dyadic comparison, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and then the seal row E.

Use Definition 1417.12 to keep the route through Q and R. The seal is downstream of the dyadic, rational, and regular-rational rows, so it cannot introduce an endpoint resolver, trichotomy row, selected digit stream, selected decimal stream, or countable-choice selector that is not already present in the carrier.

Finally project the excluded-coordinate clause of Definition 1417.1. Since quotient reals, endpoint convention resolvers, trichotomy decisions, selected representatives, selected decimal streams, and completeness witnesses are absent from the displayed packet, none escapes through E. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSealNonescape`

Theorem 1417.18 (DecimalExpansion Real seal factorization)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet whose terminal row is read as a* $\mathsf{Real}^{\uparrow}$ *seal factors through finite decimal windows, dyadic comparison, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{RealSeparability}^{\uparrow}$ *countable-density route. Concretely, the seal read begins with the digit and window rows D, W, evaluates those rows through the decimal place-value row V, passes through the dyadic and rational comparison row Q, reaches the regular rational row R, and only then exposes the terminal seal E. The same finite decimal windows supply the countable dense readback interface named by chapter 1809; no direct decimal stream, quotient real, selected representative, or completeness witness is exported by the seal.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealSealFactorization`

Proof. Start with the dyadic normalization lock of Definition 1417.16. It forces every real-facing decimal value to pass through the digit-place tolerance ledger, finite window row, regular rational readback row, and terminal seal.

Apply the Real seal nonescape theorem Definition 1417.17. The terminal row E cannot add a quotient real, selected decimal stream, endpoint resolver, or completeness witness after the finite route has reached it.

Finally use the countable-density role of $\mathsf{RealSeparability}^{\uparrow}$ described in the chapter introduction. The same finite decimal-prefix windows that feed D, W, V, Q, R, E are the windows counted by the RealSeparability route, so the factorization records a finite dense readback path rather than a direct decimal-to-real shortcut. □

Theorem 1417.19 (DecimalExpansion digit-carry determinacy)**.** *Let*

$$K = (D, W, V, Q, R, E, H, C, P, N) \quad \text{and} \quad K' = (D', W', V', Q', R', E', H', C', P', N')$$

be accepted $\mathsf{DecimalExpansion}^{\uparrow}$ *packets. If K and K' display the same finite digit window and the same dyadic place-value ledger before their terminal seals are read, then they expose the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route before the* $\mathsf{Real}^{\uparrow}$ *seal. Carry normalization is confined to the finite window ledger: it may rewrite the displayed digit and place-value rows inside D, W, V, Q, but it cannot introduce a second regular-rational route, a direct decimal-to-real shortcut, a quotient real, a selected representative, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with window normalization, Definition 1417.9. It forces both packets to display their finite digit ledgers and finite window ledgers before V, Q, R, E can be read, so the shared window fixes the source surface for any carry operation.

Use the dyadic comparison handoff, Definition 1417.11. The shared dyadic place-value ledger places both packets on the same comparison row before either packet reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Next apply the dyadic normalization lock, Definition 1417.16. It prevents carry normalization from bypassing the dyadic and regular-rational route; the normalized digit rows remain confined to the finite window and comparison ledgers.

Finally use Real seal factorization, Definition 1417.18. The terminal seal is downstream of D, W, V, Q, R, so equality of the displayed window and dyadic place-value ledger determines the readback route before E is exposed. Since the carrier has no quotient-real, selected-representative, direct decimal-stream, or completeness coordinate, carry normalization exports no additional route beyond the finite ledger. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitCarryDeterminacy`

Theorem 1417.20 (DecimalExpansion separability density obligation)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet supplies its* $\mathsf{RealSeparability}^{\uparrow}$ *density reads only through finite decimal-prefix windows. More precisely, a density-window consumer must read the digit ledger D, the decimal-place window ledger W, the place-value row V, the dyadic comparison row Q, the regular rational handoff R, the terminal real seal E, and then the structural rows H, C, P, N of Definition 1417.1; no dense route is exported from an infinite decimal stream, a quotient real, or a selected representative.*

Proof. Project the carrier of Definition 1417.1. The only rows that can feed a density read are the displayed finite prefix rows D, W, V, Q, R, E together with the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 1417.3. Ledger exhaustion restricts every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer to the same finite coordinates, so a density-window consumer cannot introduce an ambient decimal stream.

Use the Real seal factorization of Definition 1417.18. The finite decimal windows that feed the seal are exactly the windows named by the RealSeparability sibling route in chapter 1809. Thus separability density is reached by finite BEDC readback rather than by a selected decimal representative. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionSeparabilityDensityObligation`

Theorem 1417.21 (DecimalExpansion digit ledger injection)**.** *The digit ledger of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet injects into finite* $\mathsf{StreamName}^{\uparrow}$ *windows before any dyadic comparison is read. Concretely, every use of the comparison row Q must first display the same finite digit and window rows D, W and the place-value row V that determine the inspected decimal prefix.*

Proof. Read the window normalization obligation of Definition 1417.9. It forces every public read of V, Q, R, E to pass through the finite digit ledger D and window ledger W.

The $\mathsf{StreamName}^{\uparrow}$ role is therefore fixed before dyadic comparison begins. The inspected decimal prefix is a finite windowed row, not a host stream, and V evaluates only that displayed prefix as a rational place-value sum.

Apply the dyadic comparison handoff of Definition 1417.11. Since Q is the only displayed route from place value to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, each dyadic comparison read carries the prior D, W, V injection into finite StreamName windows with it. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionDigitLedgerInjection`

Theorem 1417.22 (DecimalExpansion finite prefix completeness)**.** *Every finite-prefix consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is exhausted by the rows*

$$D, W, V, Q, R, E, H, C, P, N.$$

In particular, a consumer that requests decimal digits, finite stream windows, dyadic comparison, regular rational readback, real sealing, transport, replay, provenance, or local naming finds all available data inside those ten rows and no additional prefix-completion coordinate.

Proof. Start from the carrier definition. The packet is exactly the finite tuple $K = (D, W, V, Q, R, E, H, C, P, N)$, and the excluded-coordinate clause removes infinite streams, Dedekind cuts, quotient reals, endpoint resolvers, trichotomy decisions, selected representatives, and completeness witnesses.

Apply the NameCert obligations of Definition 1417.3. Carrier admission and classifier transport are componentwise through the same ten rows, while ledger exhaustion lets consumers inspect only those displayed coordinates.

Finally combine the Real readback boundary of Definition 1417.7 with the dyadic normalization lock of Definition 1417.16. Every real-facing finite-prefix read moves through digit windows, dyadic tolerance, regular rational readback, and the terminal seal before structural replay and naming, so $D, W, V, Q, R, E, H, C, P, N$ exhaust the prefix-completeness surface. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionFinitePrefixCompleteness`

Theorem 1417.23 (DecimalExpansion endpoint-normalization handoff)**.** *Every endpoint-sensitive decimal consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must pass the rows*

$$D, W, V, Q, R, E$$

through $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *before any* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *consumer reads the normalized prefix. The handoff exports only the finite digit ledger, decimal window, place-value row, dyadic comparison, regular rational handoff, and terminal Real seal already displayed by the packet; it exports no repeating-nine endpoint convention, quotient real, selected stream, or Real completeness witness.*

Proof. Start from the finite prefix completeness theorem Definition 1417.22. It exhausts every prefix-sensitive read by $D, W, V, Q, R, E, H, C, P, N$ and leaves no additional endpoint coordinate in the $\mathsf{DecimalExpansion}^{\uparrow}$ packet.

Use the window normalization obligation Definition 1417.9. It forces each public read of place values, dyadic comparisons, regular handoffs, and Real seals to pass through the finite digit and window rows D, W.

Apply the dyadic comparison handoff of Definition 1417.11. The comparison row Q is the only displayed passage from place-value data in V toward the regular rational row R and the seal E.

The endpoint-sensitive part is therefore delegated to chapter 1429. Only after that finite endpoint-normalization surface has acted on the same D, W, V, Q, R, E route may the canonical readback surface of chapter 3952 read the normalized prefix. The excluded endpoint convention, quotient real, selected stream, and completeness witness are absent from the DecimalExpansion carrier and cannot be supplied by this handoff. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointNormalizationHandoff`

Theorem 1417.24 (DecimalExpansion endpoint Cauchy schedule)**.** *Every endpoint-sensitive Cauchy schedule consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must traverse the finite digit window rows* D, W*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, endpoint normalization, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *before any real-facing decimal read is exported. The schedule exposes only finite prefix observations and their normalized endpoint route; it does not select a decimal stream, choose a repeating-nine convention, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the endpoint-normalization handoff of Definition 1417.23. It already forces every endpoint-sensitive read through the displayed D, W, V, Q, R, E route before the normalized decimal surface is read.

Use finite prefix completeness from Definition 1417.22. The packet has no additional coordinate beyond the displayed digit, window, place-value, dyadic comparison, regular readback, Real seal, transport, replay, provenance, and naming rows.

The Cauchy schedule is therefore read through the same finite route. The dyadic comparison Q supplies the requested tolerance window, R supplies regular rational readback, endpoint normalization resolves only the displayed finite endpoint case, and E is the terminal Real seal for the real-facing read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointCauchySchedule`

Theorem 1417.25 (DecimalExpansion finite-prefix separability route)**.** *Every finite-prefix separability consumer of an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet must read finite digit windows* D, W*, dyadic comparison* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, endpoint normalization, the* $\mathsf{RealSeparability}^{\uparrow}$ *finite-density route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before treating the decimal packet as a real-separating datum.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionFinitePrefixSeparabilityRoute`

Proof. Project the carrier through Definition 1417.1. A separability read receives the same finite prefix data as every other consumer: digit windows, place values, dyadic comparisons, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Apply the finite prefix completeness theorem Definition 1417.22. It prevents a separability consumer from asking for an infinite decimal stream or a selected representative outside the packet.

Now use the Real seal factorization already cited by the chapter and route density through chapter 1809. Endpoint normalization acts before this density read, so the finite decimal prefix is normalized, read back through R, and sealed by E before the $\mathsf{RealSeparability}^{\uparrow}$ surface consumes it. □

Theorem 1417.26 (DecimalExpansion real window exhaustion)**.** *For an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$*-facing decimal read is exhausted by finite digit windows* D, W*, dyadic comparison* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, endpoint normalization, optional* $\mathsf{RealSeparability}^{\uparrow}$ *finite-density inspection, and the terminal Real seal* E*. No real-facing consumer can bypass these rows by importing selected decimal streams, quotient-real equality, trichotomy, or Real completeness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealWindowExhaustion`

Proof. Begin with Definition 1417.7. It places every real-facing read behind the displayed decimal windows, dyadic comparison, regular rational readback, and Real seal.

Add the endpoint-normalization handoff of Definition 1417.23. Endpoint-sensitive reads do not create a second route; they use the same finite prefix rows and then pass through the finite endpoint-normalization surface.

Finally apply Definition 1417.25 when the consumer asks for density or separability data. The $\mathsf{RealSeparability}^{\uparrow}$ surface receives only the finite normalized window, regular readback, and terminal seal, so the displayed rows exhaust the real-facing decimal window. □

Theorem 1417.27 (DecimalExpansion real-window audit route)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the finite-window exhaustion of Definition 1417.26 is an audit route through exactly*

$$D, W, Q, R, E, H, C, P, N.$$

Every $\mathsf{Real}^{\uparrow}$*-facing read that begins in digit windows or endpoint-sensitive windows must pass through the displayed dyadic comparison,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it is accepted as a decimal real-window observation.*

Proof. Begin with Definition 1417.26. It already confines real-facing decimal reads to finite digit windows, dyadic comparison, regular rational readback, endpoint normalization, optional finite-density inspection, and the terminal Real seal.

Use Definition 1417.7 and Definition 1417.11 to name the concrete route. The read moves from D, W through Q before it reaches R and E, so no selected decimal stream or quotient-real equality can enter the audit path.

Finally append the structural rows supplied by Definition 1417.3. Componentwise transport, replay, provenance, and local naming are exactly H, C, P, N, and the carrier has no coordinate outside $D, W, V, Q, R, E, H, C, P, N$ from which another audit row could be read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRealWindowExhaustionAuditRoute`

Theorem 1417.28 (DecimalExpansion endpoint-window transport)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, endpoint-sensitive finite windows transport through the same endpoint-normalization handoff used by Definition 1417.23. The transported read exposes only the displayed digit ledger* D*, window ledger* W*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*; it does not select a decimal stream, choose a repeating-nine convention, or import a detached endpoint resolver.*

Proof. Project the packet through Definition 1417.1. Endpoint-sensitive reads have access to the same finite rows as every other decimal consumer and no additional endpoint coordinate.

Apply Definition 1417.23. Its handoff sends the endpoint-sensitive part through D, W, V, Q, R, E before a normalized real-decimal read is exported.

The transport part is supplied by Definition 1417.9 and Definition 1417.22. Window refinement is componentwise through H, replay is through C, provenance through P, and the local name through N, so the endpoint window remains inside the displayed packet instead of selecting an external stream. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionEndpointWindowTransport`

Theorem 1417.29 (DecimalExpansion RegSeqRat prefix coverage)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing decimal prefix read exported by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is covered by finite digit windows together with dyadic comparison. Concretely, the read starts in* D, W*, evaluates its place-value and tolerance information through* V, Q*, reaches the regular rational row* R*, and then remains governed by* E, H, C, P, N *before any downstream* $\mathsf{Real}^{\uparrow}$ *consumer may inspect it.*

Proof. Use Definition 1417.8. It places each regular-rational handoff after the finite decimal prefix and its comparison ledger, rather than after an infinite decimal stream.

Apply Definition 1417.11. The row Q is the required finite dyadic comparison between place-value data and the regular rational route, so a $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot read R directly from D, W, V.

Finally combine Definition 1417.22 with Definition 1417.27. The prefix remains covered by finite windows and dyadic comparison after the Real seal and structural rows are inspected, and no selected representative, trichotomy decision, or ambient completeness witness is exported. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatPrefixCoverage`

Theorem 1417.30 (DecimalExpansion prefix normalization triad)**.** *Every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *finite-window read over digit windows* D, W*, place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *normalizes through the triad of finite prefix coverage, dyadic comparison, and Real readback boundary. The normalized read refuses quotient reals, selected decimal streams, endpoint convention selectors, trichotomy, and Real completeness before any downstream consumer may inspect* R *or* E*.*

Proof. Start with Definition 1417.29. It already places each $\mathsf{RegSeqRat}^{\uparrow}$-facing prefix read inside finite digit windows D, W, the place-value row V, and dyadic comparison Q before R is exposed.

Next apply Definition 1417.16. This prevents a normalized prefix from being replaced by a selected stream or by an endpoint convention outside the displayed rows.

Finally use Definition 1417.7. The route reaches E only after D, W, V, Q, R have been inspected, so quotient reals, trichotomy, and ambient completeness are unavailable to the normalized prefix read. □

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPrefixNormalizationTriad`

Theorem 1417.31 (DecimalExpansion terminal carry refusal)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet, a terminal-carry read on finite digit windows* D, W *must pass through the place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *as displayed packet data. It cannot discharge the carry by choosing a repeating-nine representative, selecting a decimal stream, invoking endpoint convention selectors, applying trichotomy, identifying quotient reals, or using Real completeness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionTerminalCarryRefusal`

Proof. Read the endpoint side through Definition 1417.28. Endpoint-sensitive finite windows remain attached to D, W, Q, R, E and do not export a detached endpoint resolver.

Use Definition 1417.9 together with Definition 1417.22. A carry inspection is still a finite prefix inspection, so the place-value row V and comparison row Q must remain visible before R is read.

The refusal clause follows from Definition 1417.27. The terminal Real seal E is reached only through the finite-window audit route, leaving no coordinate for a selected stream, quotient-real equality, trichotomy, endpoint convention selector, or completeness witness. □

Theorem 1417.32 (DecimalExpansion RegSeqRat density unblock)**.** *The finite decimal-window route supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *density-facing read required by later real-name consumers: digit windows* D, W *expose place-value data* V*, dyadic comparison* Q *bounds the prefix against the regular rational row* R*, and the terminal seal* E *records the Real handoff. This density-facing read is a finite BEDC route and not a quotient-real argument, selected decimal stream, endpoint selector, trichotomy decision, or Real-completeness application.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatDensityUnblock`

Proof. Begin with Definition 1417.8. It routes decimal prefixes to $\mathsf{RegSeqRat}^{\uparrow}$ through displayed finite windows rather than through an infinite stream representative.

Combine it with Definition 1417.29. The place-value row V and dyadic comparison row Q are the finite density evidence that lets R be consumed by a downstream real-name route.

Finally apply Definition 1417.7 and Definition 1417.27. The Real seal E is a terminal packet row after D, W, V, Q, R, so the density-facing read stays finite and refuses quotient reals, selected streams, endpoint selectors, trichotomy, and ambient completeness. □

Theorem 1417.33 (DecimalExpansion carry normalization totality)**.** *For every accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *finite digit-window packet* $K = (D, W, V, Q, R, E, H, C, P, N)$*, every carry or borrow read over the inspected decimal window has a normalized finite-window representative using only the rows* D, W, V, Q*. The normalized representative preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and it introduces no selected decimal stream, endpoint convention, quotient real, trichotomy decision, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionCarryNormalizationTotality`

Proof. Induct over the finite digit window named by D, W. The empty and one-place windows normalize by reading the displayed place-value row V and leave Q, R, E unchanged.

For the extension step, replay the local carry ledger inside the next displayed digit position. The dyadic comparison row Q records the tolerance effect of the carry move, so the normalized prefix still reaches the same finite comparison surface.

Finally use Definition 1417.29 and Definition 1417.18. Since R and E are downstream of the normalized finite prefix, no extra stream selector, endpoint convention, quotient representative, trichotomy row, or completeness coordinate can appear. □

Theorem 1417.34 (DecimalExpansion signed window transport)**.** *Let* $K = (D, W, V, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet. If a signed decimal consumer attaches a displayed sign row to the same finite window and transports that signed read by the componentwise* hsame *row* H*, then the transported read preserves the digit ledger* D*, window row* W*, place-value row* V*, dyadic comparison row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The sign transport therefore remains a finite decimal-window route and does not create a second magnitude, selected signed expansion, quotient real, trichotomy decision, or detached Real seal.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionSignedWindowTransport`

Proof. Start from the componentwise transport row H in Definition 1417.1. It transports the displayed packet rows and has no independent coordinate for a second magnitude or a selected signed expansion.

The transported signed read still exposes D, W, V, Q before it reaches R. Apply Definition 1417.9 to keep the place-value and dyadic comparison data attached to the same finite window.

The terminal seal is then controlled by Definition 1417.7. Since E is read only after the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the signed transport cannot bypass the finite decimal route or import quotient equality, trichotomy, or a detached Real-facing value. □

Theorem 1417.35 (DecimalExpansion RegSeqRat digit replay)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalExpansion}^{\uparrow}$ *packet is replayable from the finite digit ledger. Concretely, for* $K = (D, W, V, Q, R, E, H, C, P, N)$*, the row* R *is determined for downstream consumers by replaying* D, W *through the place-value row* V *and dyadic comparison row* Q *before the terminal seal* E *is exposed. No downstream consumer can replace that replay by a quotient real, selected decimal stream, endpoint selector, trichotomy decision, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionRegSeqRatDigitReplay`

Proof. Read the digit ledger through Definition 1417.21. Every comparison use of Q first displays the same finite digit and window rows D, W together with the place-value row V.

Next apply Definition 1417.29. The $\mathsf{RegSeqRat}^{\uparrow}$ row is covered by finite digit windows and dyadic comparison, so replaying D, W, V, Q reconstructs the public route to R.

Finally use Definition 1417.32. The density-facing handoff and the Real seal remain finite BEDC routes, leaving no coordinate for quotient reals, selected decimal streams, endpoint selectors, trichotomy, or ambient completeness. □

Theorem 1417.36 (DecimalExpansion public window export)**.** *Every public-facing* $\mathsf{DecimalExpansion}^{\uparrow}$ *consumer must expose the digit ledger D, decimal window row W, place-value row V, dyadic and rational comparison row Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport, replay, provenance, and local* NameCert *rows before reading a real-facing decimal value. The export is a finite decimal-window route and not a quotient real, selected decimal expansion, endpoint selector, trichotomy decision, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.DecimalExpansionUp.DecimalExpansionPublicWindowExport`

Proof. Start with the window obligations. Theorems 1417.2 and 1417.9 force every decimal read to display D, W, V, Q before any downstream handoff is available.

Next use the RegSeqRat route. Theorems 1417.8, 1417.29, and 1417.35 keep the handoff row R replayable from the finite digit ledger and dyadic comparison rows.

Finally apply the Real boundary and the non-escape audit. Theorems 1417.7, 1417.27, and 1417.34 leave the terminal seal E downstream of the finite route and provide no coordinate for quotient reals, selected expansions, endpoint selectors, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DecimalExpansion}^{\uparrow}$ packet is generated from finite decimal digit rows, decimal-place windows, rational place-value rows, dyadic/rational comparison rows, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, displayed hsame and Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1417.1, Definition 1417.3, Definition 1417.2, Definition 1417.7, Definition 1417.8, Definition 1417.9, Definition 1417.29, Definition 1417.35, and Definition 1417.36 bind the finite decimal-window carrier, digit classifier, RegSeqRat and Real-facing readback rows, hsame transport, Cont replay, Pkg provenance, local NameCert rows, and the public decimal-window consumer export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DecimalExpansion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, endpoint uniqueness, trichotomy, selected representatives, RealUp completeness, or a standard bridge.

Upgrade path: No further paper-axis closure is claimed here; bridge and Lean verification remain separate axes.

Chapter 1418

A Concrete Naming Certificate for $\mathsf{ContractionMapping}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, a $\mathsf{ContractionMapping}^{\uparrow}$ packet defines a contraction mapping as a certified self-map of a metric carrier with a displayed rational bound below one, not as a selected fixed point. The packet is the map-level interface consumed by the fixed-point route: it refines the Lipschitz carrier of chapter 410, reads its domain through chapter 57, hands finite iterate estimates to chapter 395, routes complete-metric consumers through chapter 128, and supplies the contraction row that chapter 396 uses before any endpoint seal is read.

Definition 1418.1 (ContractionMapping carrier)**.** *A* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, d, T, G, \lambda, M, I, H, C, P, N)$$

with the following visible coordinates:

1. *X is the carried* $\mathsf{Met}^{\uparrow}$ *point-carrier row;*
2. *d is the metric-distance witness row used to read source and target distances inside the same carrier;*
3. *T is the self-map code row, displaying the route from a carried point history of X to another carried point history of X;*
4. *G is the graph row witnessing that each accepted application of T is a finite BEDC graph read rather than a host function application;*
5. *λ is the rational contraction-bound ledger, with displayed nonnegativity and strict unit-bound rows;*
6. *M is the induced modulus row obtained from the displayed bound ledger and the metric-distance reads;*
7. *I is the one-step* $\mathsf{PicardContraction}^{\uparrow}$ *route recording $x \mapsto T(x)$ and the contracted distance comparison for adjacent iterate reads;*
8. *H stores componentwise* hsame *transports for the metric carrier, distance row, self-map row, graph row, rational bound, induced modulus, Picard route, continuation rows, provenance row, and local naming row;*

9. *C stores the* Cont *routes for metric reads, self-map application, graph replay, bound comparison, induced-modulus readback, Picard-step replay, and consumer handoff;*
10. *P is the* Pkg *provenance row over* $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}$ *row.*

The carrier exposes a self-map contraction surface. It contains no coordinate for a selected fixed point, ambient completeness, quotient map equality, host function extensionality, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.ContractionMappingUp.ContractionMappingUp`

Theorem 1418.2 (ContractionMapping NameCert obligations)**.** *For an accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}$ *surface consists of carrier admission, componentwise classifier transport, self-map graph exactness, rational bound stability below one, induced modulus visibility, one-step Picard route compatibility, package provenance stability, and local naming for the finite packet of Definition 1418.1.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.TasteGate.ContractionMappingTasteGate single carrier alignment`

Proof. Carrier admission is exactly the displayed tuple of metric, distance, self-map, graph, bound, modulus, Picard-route, transport, continuation, provenance, and naming rows.

Classifier transport is componentwise. The hsame rows move the carrier, distance, self-map, graph, bound, modulus, Picard, provenance, and naming data, while the Cont rows replay metric reads, application, graph inspection, bound comparison, and Picard-step handoff.

Exactness and stability are field exclusions plus the displayed bound ledger. Every accepted target-distance read factors through d, T, G, and λ, and λ carries the strict unit-bound row needed by contraction consumers.

The package and local certificate rows list only the stated dependencies. Repacking these visible rows gives the local naming surface without selecting a fixed point, asserting completeness, quotienting maps, or invoking a bridge. □

Theorem 1418.3 (ContractionMapping Picard-step contraction)**.** *Every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier exposes a finite* $\mathsf{PicardContraction}^{\uparrow}$ *one-step route I. For any displayed adjacent source histories x, y in the carried metric row, the graph reads for $T(x)$ and $T(y)$ and the rational ledger λ give the finite comparison*

$$d(T(x), T(y)) \leq \lambda\, d(x, y),$$

with the strict unit-bound row of λ retained as part of the packet.

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier picard step contraction`

Proof. Read the self-map coordinate T and graph coordinate G. They provide finite application rows for $x \mapsto T(x)$ and $y \mapsto T(y)$ inside the same metric carrier X.

The distance row d reads both the source distance and the target distance. The rational ledger λ is stored with the displayed comparison row and the strict unit-bound row, so the target-distance comparison is a packet field rather than a host analytic inference.

The one-step Picard route I records exactly this application-and-comparison pattern. Since I has no infinite-orbit coordinate, the theorem exports only the adjacent-step contraction surface consumed by $\mathsf{PicardContraction}^{\uparrow}$. □

Theorem 1418.4 (ContractionMapping orbit induction spine)**.** *Let $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ be an accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier and let ℓ be a finite* $\mathsf{List}^{\uparrow}$ *ledger displaying the Picard window from a base row x_0. Primitive induction over ℓ extends the one-step route I to every adjacent displayed row of the window: the base read starts at x_0, each successor read is supplied by the graph row G, and every adjacent comparison is the instance of Definition 1418.3 read through the same $X, d, T, G, \lambda, M, I, H, C, P, N$ rows. The induction exports only the finite displayed chain; it does not construct an infinite orbit, choose an endpoint, or assert completeness.*

Proof. Induct on the finite $\mathsf{List}^{\uparrow}$ ledger ℓ. In the base case, the ledger exposes only the starting row x_0, so there is no adjacent comparison to export and no endpoint or completeness row can be read.

For the successor step, assume the displayed prefix already factors every adjacent comparison through $X, d, T, G, \lambda, M, I, H, C, P, N$. The next row is admitted only by the graph row G as the application of T to the preceding displayed row.

Apply Definition 1418.3 to that adjacent pair. The comparison is therefore the same finite one-step route I, with the same metric row X, d and rational ledger λ; adjoining it to the prefix gives the next finite Picard window.

The induction variable is the ledger itself, not a sequence object outside the packet. Thus the argument closes exactly the displayed adjacent rows and refuses an infinite orbit, selected endpoint, or ambient completeness claim. □

Definition 1418.5 (ContractionMapping Cauchy orbit ledger)**.** *For an accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$, a finite Cauchy-orbit ledger is a displayed packet*

$$O = (x_0, \ell, B, R, W)$$

where x_0 is the starting point row in X, ℓ is a finite $\mathsf{List}^{\uparrow}$ *of Picard iterates $x_0, T(x_0), \ldots, T^n(x_0)$ read through G, B is the rational bound-power row storing the inspected powers of λ, R is the dyadic tolerance row used for adjacent and tail comparisons, and W is the displayed distance-contraction route*

$$d(x_{i+1}, x_{i+2}) \le \lambda\, d(x_i, x_{i+1}) \quad \text{and} \quad d(x_i, x_j) \le B(i, j)\, d(x_0, x_1)$$

for the finite indices carried by ℓ and R. The ledger contains no selected limit point, no completed orbit, no ambient completeness row, and no host fixed-point equality.

Theorem 1418.6 (ContractionMapping Cauchy-orbit contractivity)**.** *For every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ and every finite Cauchy-orbit ledger $O = (x_0, \ell, B, R, W)$ of Definition 1418.5, each adjacent iterate comparison in ℓ factors through the stored* $\mathsf{Met}^{\uparrow}$ *carrier row X, d, the graph row G, the rational bound ledger λ, and the* $\mathsf{PicardContraction}^{\uparrow}$ *route I. The dyadic tolerance row R and bound-power row B only replay those finite comparisons; they do not introduce a separate convergence theorem.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier cauchy orbit contractivity`

Proof. Project the contraction carrier. The metric carrier and distance row X, d are the only source of distance comparisons, and the self-map is available only through T and the graph row G.

Use Definition 1418.3 at each displayed adjacent pair in the finite list ℓ. The theorem supplies the comparison through G, λ, and the Picard route I for that pair.

The ledger rows B, R, W then record only finite replay data: bound powers, dyadic tolerances, and the displayed chain of inequalities already obtained from the adjacent-step route. Since no row of O carries a limit or an ambient completeness witness, the contractivity statement remains a finite orbit ledger. □

Theorem 1418.7 (ContractionMapping Banach handoff)**.** *Every* BanachFixedPoint$^{\uparrow}$ *consumer of an accepted* ContractionMapping$^{\uparrow}$ *carrier receives the finite Cauchy-orbit ledger of Definition 1418.5 together with a separate* CompleteMetric$^{\uparrow}$ *row. The handoff exports the displayed Picard iterates, dyadic tolerances, rational bound powers, graph reads, and finite Cauchy comparison route; it does not export a selected fixed point, a completed orbit endpoint, host fixed-point equality, quotient equality of maps, or ambient completeness from the contraction packet alone.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier banach handoff`

Proof. Begin with the complete-metric handoff of Definition 1418.8. It makes the metric carrier, graph-exact self-map, rational contraction ledger, induced modulus, and Picard one-step route visible, but it explicitly does not supply completeness as a contraction-packet coordinate.

Attach the finite orbit ledger O. By Definition 1418.6, every adjacent and tail comparison in O is replayed through X, d, G, λ, I and the ledger rows B, R, W.

A BanachFixedPoint$^{\uparrow}$ consumer must therefore add the separate CompleteMetric$^{\uparrow}$ row before any endpoint seal can be read. The contraction packet and orbit ledger give only finite Cauchy evidence; fixed-point selection, completed endpoints, host equality, quotient map equality, and ambient completeness remain outside the handoff. □

Theorem 1418.8 (ContractionMapping complete-metric handoff)**.** *Every* CompleteMetric$^{\uparrow}$ *or* BanachFixedPoint$^{\uparrow}$ *consumer of an accepted* ContractionMapping$^{\uparrow}$ *carrier must pass through the rows*

$$X,\ d,\ T,\ G,\ \lambda,\ M,\ I,\ H,\ C,\ P,\ N.$$

The handoff provides a metric self-map, a graph-exact contraction comparison, an induced modulus, and the one-step Picard route. It does not provide a complete carrier by itself, a Cauchy-limit endpoint, host fixed-point equality, quotient equality of maps, or bridge-checked standardization.

Proof. The carrier inventory in Definition 1418.1 lists all rows available to a consumer. A complete-metric consumer may read the metric row X, d, the self-map and graph rows T, G, the rational contraction ledger λ, and the induced modulus M.

The Picard-facing part is supplied by Definition 1418.3: it is the finite one-step route I, transported by H and replayed by C.

Completeness is not a coordinate of the contraction packet. A CompleteMetric$^{\uparrow}$ handoff therefore requires a separate complete-metric row, and a BanachFixedPoint$^{\uparrow}$ endpoint route requires the later Picard, modulus, completion, and seal rows named in its own carrier. The contraction packet contributes only the map-level surface displayed here. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier complete metric handoff`

Theorem 1418.9 (ContractionMapping tail-modulus readiness)**.** *For every accepted* ContractionMapping$^{\uparrow}$ *carrier* $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ *and every finite Cauchy-orbit ledger* $O = (x_0, \ell, B, R, W)$*, the finite orbit ledger together with the rational contraction bound yields a displayed tail-modulus row for Picard iterates through* Met$^{\uparrow}$*,* LipschitzMap$^{\uparrow}$*,* PicardContraction$^{\uparrow}$*, and* CompleteMetric$^{\uparrow}$*. The row is read from the metric carrier and distance surface* X, d*, the graph-exact self-map* T, G*, the strict unit-bound ledger* λ*, the induced modulus* M*, the Picard route* I*, and the finite replay rows* B, R, W*; it does not select a limit point.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier tail modulus readiness`

Proof. Project the carrier of Definition 1418.1. The metric and distance rows X, d provide the only distance reads, while T, G provide graph-exact applications of the self-map.

Use Definition 1418.3 for adjacent Picard comparisons and Definition 1418.6 for their finite replay through the orbit ledger. The rational bound ledger λ and induced modulus row M are therefore visible before any tail claim is made.

The Banach and complete-metric handoffs, Definition 1418.7 and Definition 1418.8, identify the consumer route. They require a separate $\mathsf{CompleteMetric}^{\uparrow}$ row and preserve the same finite $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, and $\mathsf{PicardContraction}^{\uparrow}$ reads, so the exported tail-modulus row is readiness data rather than a selected endpoint. □

Definition 1418.10 (ContractionMapping regular-Cauchy modulus surface)**.** *For an accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier* $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ *and a finite Cauchy-orbit ledger* $O = (x_0, \ell, B, R, W)$*, the regular-Cauchy modulus surface is the displayed packet*

$$\Theta = (X, d, T, G, \lambda, M, I, O, Q, H, C, P, N)$$

where Q *records the finite tail-bound thresholds extracted from* B, R, W *and presents them to a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer. The surface is a route from contraction data to regular-Cauchy completion evidence. It is not a fixed-point selector, not a completed endpoint, not quotient equality of Picard streams, and not ambient completeness.*

Lean4 (definition) `BEDC.Derived.ContractionMappingUp.ContractionMappingRegularCauchyModulusSurface`

Theorem 1418.11 (ContractionMapping tail-bound extraction)**.** *For every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier* K *and finite Cauchy-orbit ledger* O*, the tail-modulus readiness row of Definition 1418.9 determines a finite extraction row* Q *whose entries are read only from the Picard-step contraction route, the Cauchy-orbit contractivity ledger, the rational bound powers, and the dyadic tail tolerance rows. No extraction step reads a selected fixed point or a completed limit.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier tail bound extraction`

Proof. Project the carrier and the orbit ledger. The only contraction comparison is the Picard-step route of Definition 1418.3, and the only iterated replay is the Cauchy-orbit contractivity statement of Definition 1418.6.

The finite bound-power row B and tolerance row R provide the threshold entries used by Q. The readiness theorem then identifies those entries as the tail-modulus row visible to completion consumers.

Because the complete-metric handoff is still separate, the extraction has no coordinate for an endpoint, selected fixed point, host fixed-point equality, quotient stream equality, or ambient completeness. □

Theorem 1418.12 (ContractionMapping regular-Cauchy modulus export)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing completion consumer of a* $\mathsf{ContractionMapping}^{\uparrow}$ *packet may read the surface of Definition 1418.10 only through the finite route*

$$X, d, T, G, \lambda, M, I, O, Q, H, C, P, N.$$

The route exports a regular-Cauchy modulus for the Picard orbit by combining the stored contraction comparison, finite orbit ledger, tail-bound extraction row, and replay data; it does not export a selected fixed point.

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier regular cauchy modulus export`

Proof. Begin with the tail-bound extraction theorem. It supplies Q from the already displayed contraction route and finite orbit ledger.

Next use Definition 1418.10. The surface appends Q to the carrier and orbit rows, so a regular-Cauchy consumer sees exactly the metric, self-map, graph, bound, Picard, orbit, threshold, transport, replay, provenance, and local naming rows.

The Banach handoff remains downstream. Thus the exported data is a modulus surface for a regular-Cauchy Picard name, not a fixed-point endpoint, completed orbit, quotient equality, host function equality, or completeness assertion. □

Theorem 1418.13 (ContractionMapping complete-metric regular-Cauchy handoff)**.** *Every* CompleteMetric$^{\uparrow}$ *or* BanachFixedPoint$^{\uparrow}$ *consumer that uses a* ContractionMapping$^{\uparrow}$ *carrier for completion evidence must first read the regular-Cauchy modulus export of Definition 1418.12 and then supply its separate complete-metric row. The contraction packet contributes only the finite RegSeqRat-facing modulus route for the Picard orbit.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier complete metric regular cauchy handoff`

Proof. Apply Definition 1418.8. It keeps completeness outside the contraction packet while exposing the metric self-map, contraction comparison, induced modulus, and Picard route.

Apply Definition 1418.12. It repacks the finite orbit ledger and extracted tail-bound row as the RegSeqRat-facing modulus surface needed by a completion consumer.

Only after those rows are displayed can a CompleteMetric$^{\uparrow}$ or BanachFixedPoint$^{\uparrow}$ packet use its own endpoint discipline. The contraction packet itself therefore exports regular-Cauchy completion evidence, not endpoint selection, host fixed-point equality, quotient map equality, or ambient completeness. □

Theorem 1418.14 (ContractionMapping fixed-point readiness)**.** *Every* BanachFixedPoint$^{\uparrow}$ *consumer may read an accepted* ContractionMapping$^{\uparrow}$ *packet only after the complete-metric row of Definition 1418.8 and the finite tail-modulus route of Definition 1418.9 have both been displayed. This readiness condition exposes the contraction packet, Picard iterates, finite Cauchy ledger, rational bound powers, and complete-metric handoff, but it does not select a fixed point, a completed orbit endpoint, or a host equality proof.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier fixed point readiness`

Proof. Start with a BanachFixedPoint$^{\uparrow}$ consumer. By Definition 1418.7, it receives the finite Cauchy-orbit ledger and a separate CompleteMetric$^{\uparrow}$ row rather than an endpoint from the contraction packet alone.

The complete-metric handoff of Definition 1418.8 makes the metric self-map, graph-exact contraction comparison, induced modulus, and one-step Picard route visible. The tail-modulus readiness theorem then packages the finite Picard comparisons into the displayed tail route required by the consumer.

Only after both rows are displayed may the consumer proceed to its own fixed-point seal. Because neither row contains a selected point, completed orbit endpoint, host fixed-point equality, quotient map equality, or ambient completeness coordinate, the readiness statement remains a finite obligation boundary. □

Theorem 1418.15 (ContractionMapping orbit uniqueness window)**.** *Let $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ be an accepted* ContractionMapping$^{\uparrow}$ *carrier, and let*

$$O = (x_0, \ell, B, R, W) \quad \text{and} \quad O' = (x_0', \ell', B', R', W')$$

be finite Cauchy-orbit ledgers for the same packet. If x_0 and x_0' are classifier-stable as the same starting row and both ledgers replay the same graph-exact self-map row T, G, then every common finite window of ℓ and ℓ' is classifier-stable through the stored Met$^{\uparrow}$*,* LipschitzMap$^{\uparrow}$*, and* PicardContraction$^{\uparrow}$ *rows. The comparison uses only $X, d, G, \lambda, M, I, H, C$ and the finite ledger rows; it does not assert an infinite orbit or a selected endpoint.*

Proof. Project the two orbit ledgers and the accepted contraction carrier. The hypotheses identify the starting rows through the local classifier and require both lists to replay the same self-map graph row G.

Induct over the common finite window. The base row is the shared start. At the successor step, graph exactness for T, G gives the next iterate in both ledgers, while the Met$^{\uparrow}$ distance row d and the LipschitzMap$^{\uparrow}$ contraction ledger λ keep the comparison inside the same metric carrier.

The PicardContraction$^{\uparrow}$ route I is the stored one-step replay for each adjacent pair. Transport by H and continuation by C preserve those rows componentwise, so the two finite windows remain classifier-stable without producing any completed orbit or fixed-point endpoint. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier orbit uniqueness window`

Theorem 1418.16 (ContractionMapping Picard regular-Cauchy uniqueness)**.** *Let $K = (X, d, T, G, \lambda, M, I, H, C, P$ be an accepted* ContractionMapping$^{\uparrow}$ *carrier, and let $O = (x_0, \ell, B, R, W)$ and $O' = (x_0', \ell', B', R', W')$ be accepted Picard orbit ledgers for the same contraction packet. If the two ledgers share the same displayed initial row and the same rational bound ledger, then the* RegSeqRat$^{\uparrow}$ *handoff rows exported from O and O' are classifier-equivalent before any* CompleteMetric$^{\uparrow}$ *endpoint row or* BanachFixedPoint$^{\uparrow}$ *seal is read.*

Proof. The shared displayed initial row and rational bound ledger put the two ledgers inside the orbit uniqueness window of Definition 1418.15. Hence every common finite Picard window is classifier-stable through the stored metric, graph-exact self-map, contraction, and replay rows.

Use Definition 1418.9 to expose the same finite tail-modulus route for those windows. Then apply Definition 1418.12; the exported RegSeqRat$^{\uparrow}$ surface reads only the Picard route, finite orbit rows, tail-bound extraction, transport, replay, provenance, and local naming rows.

Finally, Definition 1418.13 identifies the downstream boundary: a complete-metric consumer supplies its own endpoint discipline only after this regular-Cauchy route has been displayed. Therefore the two Picard ledgers determine classifier-equivalent RegSeqRat$^{\uparrow}$ handoff rows without selecting a completed endpoint or invoking a BanachFixedPoint$^{\uparrow}$ seal. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier picard regular cauchy uniqueness`

Theorem 1418.17 (ContractionMapping fixed-point tail seal)**.** *Before a* BanachFixedPoint$^{\uparrow}$ *consumer supplies the separate* CompleteMetric$^{\uparrow}$ *row, the only endpoint-facing seal exported by an accepted* ContractionMapping$^{\uparrow}$ *packet is the finite tail-modulus route of Definition 1418.9. Thus every fixed-point-facing request must pass through the graph-exact self-map T, G, rational bound ledger λ, induced modulus M, Picard route I, finite orbit ledger O, and complete-metric handoff boundary; the contraction packet alone cannot select a fixed point or certify a completed endpoint.*

Proof. Begin with fixed-point readiness in Definition 1418.14. It requires both the complete-metric handoff and the finite tail-modulus route before a Banach consumer may proceed.

Use the tail-modulus readiness theorem. The route it exports is read from $X, d, T, G, \lambda, M, I$ and the finite ledger rows B, R, W, so it is endpoint-facing only as a tail seal over finite Picard data.

The complete-metric row is deliberately outside the contraction packet. Consequently no row of K can supply a selected fixed point, completed orbit endpoint, host fixed-point equality, quotient map equality, or ambient completeness proof before the downstream Banach chapter consumes its own complete-metric data. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier fixed point tail seal`

Theorem 1418.18 (ContractionMapping complete-metric consumer package)**.** *For every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *carrier* $K = (X, d, T, G, \lambda, M, I, H, C, P, N)$ *and finite Cauchy-orbit ledger* $O = (x_0, \ell, B, R, W)$*, the local NameCert obligation surface needed by downstream Banach, Picard, complete-metric, and Lipschitz consumers is the package consisting of the complete-metric handoff of Definition 1418.8, the tail-modulus readiness route of Definition 1418.9, the fixed-point readiness boundary of Definition 1418.14, and the orbit uniqueness window of Definition 1418.15. This package remains finite and* BHist*-anchored and does not claim a selected fixed point.*

Proof. Read the complete-metric handoff first. It exposes the metric carrier, graph-exact self-map, rational contraction ledger, induced modulus, and Picard one-step route while leaving the actual complete-metric row to the consumer.

Next attach the finite orbit data. Tail-modulus readiness supplies the tail route through the ledger rows, and the orbit uniqueness window makes matching finite Picard windows stable under the same classifier and graph replay.

Finally use fixed-point readiness to delimit the downstream obligation. The package now contains exactly the finite rows that Banach, Picard, complete-metric, and Lipschitz consumers need to read before their own endpoint arguments begin, with no coordinate for endpoint selection, completed orbit equality, or host fixed-point proof. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier complete metric consumer package`

Definition 1418.19 (ContractionMapping scoped source stack)**.** *The scoped source stack for a* $\mathsf{ContractionMapping}^{\uparrow}$ *read is the finite packet*

$$\mathcal{S}_{\mathsf{ContractionMapping}^{\uparrow}} = (\mathsf{Met}^{\uparrow}, \mathsf{LipschitzMap}^{\uparrow}, \mathsf{PicardContraction}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pk}$$

It is read through the carrier rows $X, d, T, G, \lambda, M, I, H, C, P, N$ *and the orbit and modulus rows* O, Q *displayed in this chapter. The stack records source metric data, Lipschitz and contraction graph data, Picard-step replay, completion-facing modulus handoff, regular-rational tail thresholds, history transport, continuation replay, provenance, and local naming; it contains no fixed-point selector or completed endpoint.*

Theorem 1418.20 (ContractionMapping scoped dependency surface)**.** *Every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *read factors through the scoped source stack of Definition 1418.19. In particular, any Picard, Banach, complete-metric, or regular-Cauchy consumer can inspect only the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before it uses its own downstream endpoint discipline.*

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier scoped dependency surface`

Proof. Project the carrier and the finite orbit rows. The carrier supplies the metric, graph, contraction bound, induced modulus, Picard route, transport, replay, provenance, and naming coordinates, while the orbit and threshold rows supply only finite Picard tail data.

Apply the complete-metric and Banach handoffs. They expose a separate $\mathsf{CompleteMetric}^{\uparrow}$ row for completion-facing consumers and keep endpoint selection outside the contraction packet.

Apply the regular-Cauchy export and complete-metric regular-Cauchy handoff. They place the tail thresholds in the $\mathsf{RegSeqRat}^{\uparrow}$-facing part of the stack, transported by the same hsame and Cont rows.

No other coordinate is available in the carrier or in the orbit surface. Therefore every accepted read factors through the displayed scoped stack and cannot import host fixed-point equality, quotient map equality, or completed endpoints. □

Theorem 1418.21 (ContractionMapping completion readiness scope)**.** *The completion-facing readiness route for a* $\mathsf{ContractionMapping}^{\uparrow}$ *packet still requires a separate* $\mathsf{CompleteMetric}^{\uparrow}$ *row. The contraction packet exports only finite Picard tail-modulus data through* $\mathsf{PicardContraction}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, together with the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *evidence of the scoped source stack.*

Proof. Start with the fixed-point readiness theorem. It already requires both the complete-metric handoff and the finite tail-modulus route before a $\mathsf{BanachFixedPoint}^{\uparrow}$ consumer may proceed.

Use the scoped dependency surface. The visible source stack includes $\mathsf{CompleteMetric}^{\uparrow}$ only as a separate consumer row and includes $\mathsf{RegSeqRat}^{\uparrow}$ only for the finite threshold and tail-modulus surface.

Thus the contraction packet contributes readiness data, not a completion theorem. Since no row names a limit point, selected fixed point, quotient map equality, or ambient completeness proof, the route remains a completion-facing boundary rather than a fixed-point construction. □

Lean4 ✓ `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier completion readiness scope`

Theorem 1418.22 (ContractionMapping public export)**.** *Every accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *packet exports the public* $\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}$ *surface obtained by packaging the scoped dependency surface, the complete-metric handoff, fixed-point readiness, orbit uniqueness, and completion readiness scope displayed in this chapter. The public export contains the carrier rows, Picard-step contraction data, finite orbit ledger, tail-modulus route, complete-metric consumer row, scoped source stack, and structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it still excludes selected fixed points, ambient completeness, quotient map equality, host function extensionality, and completed endpoint seals.*

Proof. First read the scoped dependency surface. It already factors every accepted contraction read through the finite source stack containing $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Next attach the downstream readiness rows. The complete-metric handoff and complete-metric consumer package expose the completion-facing boundary, while fixed-point readiness and orbit uniqueness keep the Picard tail data finite and local to the displayed orbit ledger.

Finally use the completion readiness scope. It states that the contraction packet contributes readiness data rather than a completion theorem or endpoint construction. Packaging these rows as the public $\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}$ export therefore gives consumers a public surface without adding selected fixed points, ambient completeness, quotient equality, host extensionality, or completed endpoint seals. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}$.

Constructive story: A $\mathsf{ContractionMapping}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier row, a metric-distance row, a self-map row, a graph row, a rational contraction-bound ledger below one, an induced modulus row, a one-step $\mathsf{PicardContraction}^{\uparrow}$ route, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1418.1, Definition 1418.2, Definition 1418.3, Definition 1418.5, Definition 1418.6, Definition 1418.7, Definition 1418.8, Definition 1418.9, Definition 1418.10, Definition 1418.11, Definition 1418.12, Definition 1418.13, Definition 1418.14, Definition 1418.15, Definition 1418.16, Definition 1418.17, Definition 1418.18, Definition 1418.19, Definition 1418.20, Definition 1418.21, and Definition 1418.22 bind the carrier, Picard-step contraction, finite orbit ledger, Banach handoff, complete-metric handoff, finite tail-modulus readiness, regular-Cauchy modulus surface, fixed-point readiness, orbit-window uniqueness, Picard regular-Cauchy uniqueness, endpoint-facing tail seal, complete-metric consumer package, scoped source stack, scoped dependency surface, completion readiness scope, and public export through the checked completion-readiness route for the constructive self-map contraction packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractionMapping}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.ContractionMappingUp.ContractionMappingCarrier\_completion\_readiness\_scope`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim host fixed-point selection, ambient completeness, quotient map equality, bridge checking, host function extensionality, or a completed endpoint seal.

Upgrade path: Only a standard bridge certificate for the same checked completion-readiness route remains for the formal bridge axis.

Chapter 1419

A Concrete Naming Certificate for $\mathsf{ArchimedeanOrderedField}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ packet records a traditional ordered-field comparison certificate over displayed constructive real names. It sits between the algebra/order surface of chapter 766, the rational-bound route of chapter 430, the real-name seal of chapter 38, and the rational source rows of chapter 37. The packet is a comparison carrier for downstream L10 and real-algebra consumers, not an ambient ordered field object.

Definition 1419.1 (ArchimedeanOrderedField carrier)**.** *An* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, A, Q, O, B, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the family of* $\mathsf{Real}^{\uparrow}$ *endpoint rows whose algebra and order reads are being compared;*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row that supplies displayed addition, negation, multiplication, absolute-value, nonnegative-order, and apartness evidence for those endpoints;*
3. *Q is the finite* $\mathsf{Rat}^{\uparrow}$ *source family used by constant-rational witnesses and Archimedean comparison bounds;*
4. *O is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *rational-bound row attached to the displayed real endpoints;*
5. *B is the local comparison ledger recording the field-operation, order, rational-embedding, and Archimedean-bound reads that the packet exports;*
6. *H stores componentwise* hsame *transports for the real endpoints, algebra/order rows, rational source rows, Archimedean bound rows, and local ledger rows;*
7. *C stores the* Cont *routes that expose the displayed operation, order, rational-bound, and rational-embedding rows to downstream consumers;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedField}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only by replaying these finite coordinates. It has no coordinate for real trichotomy, quotient real equality, a host ordered-field carrier, a quotient-reals construction, a selected least bound, or a standard bridge theorem.

Lean4 (definition) `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldCarrier`

Theorem 1419.2 (ArchimedeanOrderedField NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedField}^{\uparrow}}$ *surface for the carrier of Definition 1419.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet* $F = (R, A, Q, O, B, H, C, P, N)$ *with real endpoint, algebra/order, rational source, Archimedean bound, comparison ledger, transport, continuation, provenance, and local naming coordinates;*
2. *classifier transport: classification transports every displayed* $\mathsf{Real}^{\uparrow}$ *endpoint,* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation or order row,* $\mathsf{Rat}^{\uparrow}$ *source row,* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row, and local ledger row componentwise through the stored* hsame *and* Cont *routes;*
3. *field-row stability: addition, negation, multiplication, unit, zero, and absolute-value reads remain inside the* $\mathsf{RealAlgOrder}^{\uparrow}$ *coordinate* A *after endpoint replacement by classified* $\mathsf{Real}^{\uparrow}$ *rows;*
4. *Archimedean comparison stability: every rational-bound or rational-embedding comparison exposed by* O *and* Q *remains tied to the same displayed real endpoints and the same local ledger rows after transport;*
5. *ledger exactness: every downstream ordered-field or Archimedean read is one of the displayed rows* $R, A, Q, O, B, H, C, P, N$ *before any consumer receives it.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldNameCertObligations`

Proof. Carrier admission is projection from Definition 1419.1. The real endpoint family, algebra/order row, rational source family, Archimedean bound row, local comparison ledger, transport table, continuation table, provenance row, and local naming row are exactly the listed coordinates.

Classifier transport is componentwise. The $\mathsf{Real}^{\uparrow}$ endpoints move with their $\mathsf{RealAlgOrder}^{\uparrow}$ operation and order rows, while the rational source rows and Archimedean bound rows move through their own displayed sibling classifiers and the stored hsame routes.

The field-row and Archimedean stability clauses then repack the transported rows into the same ledger shape B. Algebraic reads use only A and the endpoint rows in R; rational-bound reads use only Q, O and the endpoint rows in R.

Ledger exactness follows because every public read is a coordinate of F or a row of B generated from those coordinates. Since no coordinate stores trichotomy, quotient real equality, a host ordered-field carrier, quotient reals, selected least bounds, or bridge-checked data, none of those principles is exported by the local certificate. □

Theorem 1419.3 (ArchimedeanOrderedField RealAlgOrder handoff)**.** *Every ordered-field or Archimedean comparison read exported by an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet factors through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row* A *and the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *row* O *over the displayed* $\mathsf{Real}^{\uparrow}$ *endpoints* R*. The* $\mathsf{RealAlgOrder}^{\uparrow}$ *side supplies the finite operation and order rows, the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *side supplies explicit rational-bound rows, and the* $\mathsf{Rat}^{\uparrow}$ *coordinate* Q *supplies the rational source data used by both comparison routes.*

Consequently the packet is a handoff between the existing real algebra/order and Archimedean-bound surfaces. It does not identify those surfaces with a host ordered field, quotient-reals carrier, complete real field, trichotomy principle, selected least-bound operation, or bridge-checked standard structure.

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRealAlgOrderHandoff`

Proof. Project the carrier tuple $F = (R, A, Q, O, B, H, C, P, N)$. The only operation and order coordinate is A, and the only Archimedean-bound coordinate is O.

Read the algebra/order route first. By Definition 766.2, the rows exposed by A are finite endpoint, operation, order, apartness, transport, continuation, provenance, and local naming rows over displayed $\mathsf{Real}^{\uparrow}$ data.

Read the Archimedean route next. By Definition 430.2, the rows exposed by O are finite rational-bound, dyadic-bound, stream-window, regular-Cauchy handoff, ledger, transport, continuation, provenance, and local naming rows for displayed $\mathsf{Real}^{\uparrow}$ data.

The rational source coordinate Q is shared by the two routes. The local ledger B records only those field-operation, order, rational-embedding, and rational-bound reads, and the transports H, C replay them componentwise for consumers.

Thus every exported comparison factors through A, O, and Q over R. The excluded host ordered-field, quotient-reals, trichotomy, completeness, least-bound, and bridge-checked objects have no carrier coordinate and no route through the local ledger. □

Theorem 1419.4 (ArchimedeanOrderedField rational-bound window)**.** *Every Archimedean comparison read exported by an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet must expose the* $\mathsf{Rat}^{\uparrow}$ *source row* Q*, the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row* O*, and the displayed* $\mathsf{Real}^{\uparrow}$ *endpoints* R *before any ordered-field consumer reads the local comparison ledger* B*. The rational-bound window is therefore the finite route*

$$Q, O, R \longmapsto B.$$

It is not a route through trichotomy, quotient real equality, selected least bounds, a host ordered-field carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRationalBoundWindow`

Proof. Project the carrier $F = (R, A, Q, O, B, H, C, P, N)$. The rational source row Q, Archimedean bound row O, endpoint family R, and comparison ledger B are displayed coordinates of the same finite packet.

Apply the NameCert obligations of Definition 1419.2. Classifier transport keeps the rational source, bound, endpoint, and ledger rows componentwise aligned, while Archimedean comparison stability ties every rational-bound read to Q, O over the same endpoints R.

Finally use the RealAlgOrder handoff of Definition 1419.3. Ordered-field reads may consume B only after the displayed algebra/order route and the rational-bound route have been replayed, and the carrier has no coordinate for the excluded trichotomy, quotient, least-bound, host-field, or completeness data. □

Theorem 1419.5 (ArchimedeanOrderedField comparison ledger exactness)**.** *For an accepted* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *packet with carrier*

$$F = (R, A, Q, O, B, H, C, P, N),$$

every ordered-field, Archimedean comparison, or rational-embedding consumer of the local comparison ledger B *reads* B *only through the displayed route* R, A, Q, O, H, C, P, N*. Here* A *is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row,* O *is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *row,* Q *is the* $\mathsf{Rat}^{\uparrow}$ *source row, and* R *is the* $\mathsf{Real}^{\uparrow}$ *endpoint family.*

Thus the ledger is an exhaustive consumer interface for those comparison reads, not a channel for trichotomy, quotient reals, selected least bounds, host ordered-field carriers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier definition of Definition 1419.1. The only coordinates adjacent to the ledger are the endpoint family R, the algebra/order row A, the rational source row Q, the Archimedean bound row O, the transport table H, the continuation routes C, the provenance row P, and the local naming row N.

Apply the NameCert obligations of Definition 1419.2. Carrier admission and classifier transport force every consumer-visible ledger row to be one of the displayed rows of F.

Field-row stability and Archimedean comparison stability keep ordered-field, rational-embedding, and rational-bound reads attached to the same R, A, Q, O coordinates.

The handoff theorem Definition 1419.3 identifies the ordered-field side with A over R and the Archimedean side with O and Q over the same endpoints. The rational-bound window of Definition 1419.4 then forces any Archimedean comparison read to pass through Q, O, R before it reaches B.

It follows that a downstream ordered-field, Archimedean comparison, or rational-embedding consumer has no remaining entry point into B except the route R, A, Q, O, H, C, P, N. Since the carrier and the cited obligations store no trichotomy, quotient-reals construction, selected least-bound operation, host ordered-field carrier, or ambient real-completeness coordinate, the ledger cannot export any of those structures. □

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldComparisonLedgerExactness`

Theorem 1419.6 (ArchimedeanOrderedField root obligation package)**.** *The carrier of Definition 1419.1, the NameCert obligations of Definition 1419.2, and the RealAlgOrder handoff of Definition 1419.3 assemble the root package consumed by* $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{RealAlgOrder}^{\uparrow}$. *The package consists of displayed ordered-field carrier rows, rational-bound rows, terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, comparison transport, displayed replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrdere}}$ *row.*

The package exports only those rows. It does not supply trichotomy, quotient equality of real names, selected rational bounds, a selected least-bound operation, a host ordered-field carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldRootObligationPackage`

Proof. Project the packet $F = (R, A, Q, O, B, H, C, P, N)$ from Definition 1419.1. The rows R, A, Q, O, B are respectively the real endpoint source, ordered algebra row, rational source family, Archimedean rational-bound row, and local comparison ledger; the rows H, C, P, N transport, replay, source, and name that finite data.

Apply Definition 1419.2. Its carrier-admission, classifier-transport, field-row stability, Archimedean comparison stability, and ledger-exactness clauses identify every root obligation with one of the displayed coordinates of F.

Apply Definition 1419.3. It routes ordered-field and Archimedean comparison reads through A, O, and Q over the same displayed $\mathsf{Real}^{\uparrow}$ endpoints R, so downstream $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealAlgOrder}^{\uparrow}$ consumers receive a single finite package.

Finally inspect the absent coordinates named in the carrier and handoff statements. Since no row stores trichotomy, quotient equality, selected bounds, host ordered-field structure, least-bound selection, or ambient completeness, the assembled root package cannot export any of those principles. □

Theorem 1419.7 (ArchimedeanOrderedField scoped closure)**.** *The carrier of Definition 1419.1, the NameCert obligations of Definition 1419.2, the RealAlgOrder handoff of Definition 1419.3, the*

rational-bound window of Definition 1419.4, comparison ledger exactness of Definition 1419.5, and the root obligation package of Definition 1419.6 bind every public comparison read to the displayed RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.ArchimedeanOrderedFieldUp.ArchimedeanOrderedFieldScopedClosure`

Proof. The carrier identifies the only public rows of an accepted packet:

$$F = (R, A, Q, O, B, H, C, P, N).$$

Thus every comparison read must begin with the displayed real endpoints, RealAlgOrder row, rational source row, Archimedean bound row, and local comparison ledger.

The NameCert obligations keep those rows stable under classifier transport. Field-row stability routes algebra and order reads through A over R, while Archimedean comparison stability routes rational-bound reads through Q and O over the same endpoints.

The RealAlgOrder handoff, rational-bound window, and comparison ledger exactness then close the consumer boundary. Ordered-field, rational-embedding, and Archimedean comparison reads reach B only through R, A, Q, O, H, C, P, N.

Finally the root obligation package repacks the same rows for RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, and Real$^{\uparrow}$ consumers. Since no coordinate remains outside the displayed transport, replay, provenance, and local naming rows, the chapter closes exactly this scoped comparison interface and no stronger ordered-field or completeness principle. □

Closure status for NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$.

Constructive story: An ArchimedeanOrderedField$^{\uparrow}$ packet is generated from displayed Real$^{\uparrow}$ endpoint rows, a RealAlgOrder$^{\uparrow}$ algebra/order row, Rat$^{\uparrow}$ rational source rows, an ArchimedeanReal$^{\uparrow}$ rational-bound row, a finite comparison ledger, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1419.7 binds every public comparison read to the displayed RealAlgOrder$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, Rat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ArchimedeanOrderedField}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completeness, trichotomy, quotient reals, selected least bounds, host ordered-field structure, or bridge-checked status.

Upgrade path: Scoped closure requires a checked finite carrier theorem binding the displayed RealAlgOrder and ArchimedeanReal rows to the local comparison ledger.

Chapter 1420

A Concrete Naming Certificate for $\mathsf{LocatedLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedLimit}^{\uparrow}$ packet records located limits as explicit modulus-and-window data for displayed real-name sequences. It is a finite route from the Cauchy-equivalence surface of chapter 1412, the regular-rational readback surface of chapter 301, the finite stream-window surface of chapter 126, and the dyadic tolerance ledger of chapter 291 to the real-name seal of chapter 38. The packet does not introduce host real equality, quotient limits, countable choice, trichotomy, or ambient completeness.

Definition 1420.1 (LocatedLimit carrier)**.** *A* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, M, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the displayed* $\mathsf{Real}^{\uparrow}$ *sequence surface whose entries are read only through named finite windows;*
2. *M is the Cauchy/modulus ledger recording the explicit tolerance budget for the displayed sequence;*
3. *T is the located finite-window schedule that says which sequence windows and dyadic tolerances are inspected for each requested precision;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled windows and the modulus ledger;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *limit seal reached through Q and through no independent host-limit coordinate;*
6. *H stores componentwise* hsame *transports for the sequence surface, modulus ledger, finite-window schedule, regular readback, and real seal;*
7. *C stores the displayed* Cont *replay routes by which consumers reread the same windows and tolerances;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected host limit, quotient equality of streams, countable-choice schedule, real trichotomy, or ambient completeness principle.

Lean4 (definition) `BEDC.Derived.LocatedLimitUp.LocatedLimitUp`

Definition 1420.2 (LocatedLimit classifier)**.** *Two accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *packets*

$$L = (S, M, T, Q, E, H, C, P, N) \quad \textit{and} \quad L' = (S', M', T', Q', E', H', C', P', N')$$

are classified together exactly when their displayed finite-window schedules expose the same dyadic tolerance comparisons after componentwise transport. Thus the classifier compares L *and* L' *only by shared windows in* T, T'*, the dyadic budgets read by* M, M'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q, Q'*, and the* $\mathsf{Real}^{\uparrow}$ *seal rows* E, E' *reached from those data.*

No classifier step may use host real equality, quotient equality of streams, a selected limiting subsequence, countable choice, trichotomy, or an ambient completeness row.

Theorem 1420.3 (LocatedLimit NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *surface for the carrier of Definition 1420.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet* $L = (S, M, T, Q, E, H, C, P, N)$*;*
2. *classifier: comparison is the finite-window and dyadic-tolerance classifier of Definition 1420.2;*
3. *transport:* H *and* C *preserve the same sequence surface, modulus ledger, schedule, readback, and seal rows under accepted consumer motion;*
4. *modulus exactness: every limit-facing tolerance request is answered by the stored Cauchy/modulus ledger* M *together with the schedule* T*;*
5. *Real-seal route: every* $\mathsf{Real}^{\uparrow}$ *limit read factors through* Q, E, H, C, P, N *after the displayed rows* S, M, T *have been read.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitNameCertObligations`
↪ variant `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier namecert obligations`

Proof. Carrier admission is projection from the packet $L = (S, M, T, Q, E, H, C, P, N)$. The sequence surface, modulus ledger, located schedule, regular readback, real seal, transport rows, replay routes, provenance, and local naming row are exactly the coordinates named by the carrier.

The classifier row is finite because it reads only the windows named by T, T' and the dyadic tolerances supplied by M, M'. The regular readbacks Q, Q' and real seals E, E' are compared only after those finite observations have been transported.

For transport, use H to move the displayed rows componentwise and C to replay the same consumer route. The replay never creates a new modulus, a host limit, or a quotient of streams; it returns to the stored rows.

Modulus exactness follows because the only tolerance ledger in the carrier is M, and the only located schedule is T. Therefore every downstream tolerance request has to pass through M and T before it can reach Q or E.

The Real-seal route is then the visible composition from S, M, T to Q and finally to E, with H, C, P, N carrying transport, replay, provenance, and naming. The excluded completeness, choice, trichotomy, quotient, and host-equality principles have no carrier coordinate. □

Theorem 1420.4 (LocatedLimit Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *limit read exported by an accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier factors through the displayed route*

$$S, M, T, Q, E, H, C, P, N.$$

The sequence surface S supplies the real-name entries, the ledger M supplies the explicit Cauchy/modulus budget, the schedule T chooses only finite windows for each dyadic tolerance, the row Q repacks those observations as a $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the row E is the sole* $\mathsf{Real}^{\uparrow}$ *limit seal available to consumers. The rows H, C, P, N provide transport, replay, provenance, and local naming for the same route.*

Proof. Project the accepted carrier. Its real-facing coordinates are not free-standing; E is stored together with the sequence surface S, the modulus ledger M, the located schedule T, and the regular readback Q.

Read the limit request in finite order. A consumer first asks T for the displayed windows attached to the requested dyadic tolerance, then reads the matching budget in M, and then uses Q to view the resulting data as a regular rational name.

Only after that route is visible can the consumer read E. The transport and replay rows H and C preserve the same route, while P and N record provenance and local naming.

Thus every $\mathsf{Real}^{\uparrow}$ limit read factors through $S, M, T, Q, E, H, C, P, N$. Since no other coordinate exists, the packet exports no real completeness theorem, selected host limit, quotient equality of streams, countable-choice schedule, trichotomy, or bridge-checked standard interpretation. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimit real seal route`

Theorem 1420.5 (LocatedLimit window schedule exactness)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier $L = (S, M, T, Q, E, H, C, P, N)$, each limit-facing tolerance request reads the displayed schedule row T and the modulus ledger M before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q or the* $\mathsf{Real}^{\uparrow}$ *seal row E. The exactness claim is the finite routing statement of Definition 1420.1 and Definition 1420.2, with the obligation order supplied by Definition 1420.3 and the terminal route supplied by Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitWindowScheduleExactness`

Proof. Project the carrier of Definition 1420.1. The only rows that can answer a tolerance request before readback are the schedule row T and the modulus ledger M.

Use the classifier of Definition 1420.2 to see that comparison is by shared finite windows and dyadic budgets, not by a host limit. Then apply Definition 1420.3: modulus exactness is one of the named local obligations and it places M and T before every read of Q or E.

Finally apply the route of Definition 1420.4. Since E is reached only after S, M, T, Q, every tolerance request has already passed through T and M before any regular readback or real seal can be exported. □

Theorem 1420.6 (LocatedLimit finite-window scope)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier $L = (S, M, T, Q, E, H, C, P, N)$, each finite-window tolerance request is scoped exactly by the* $\mathsf{StreamName}^{\uparrow}$ *window surface S, T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger M, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$ *row N. The scope is the source-row route named by the carrier, classifier, NameCert obligations, window-schedule exactness, and Real-seal route; it excludes quotient streams, countable choice, trichotomy, selected host limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier finite window scope`

Proof. Project the carrier of Definition 1420.1. Its source rows are precisely $S, M, T, Q, E, H, C, P, N$, so a tolerance request has no access to any row outside that finite packet.

Use the classifier of Definition 1420.2. It compares accepted packets only through transported finite windows and dyadic budgets, and therefore binds the request to the $\mathsf{StreamName}^{\uparrow}$ surface S, T and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger M before any readback is visible.

Apply Definition 1420.3 and Definition 1420.5. The obligation rows require modulus exactness and route every tolerance request through T and M before it reaches the $\mathsf{RegSeqRat}^{\uparrow}$ row Q or the $\mathsf{Real}^{\uparrow}$ seal E.

Finally use Definition 1420.4. The terminal route is $S, M, T, Q, E, H, C, P, N$, with H, C, P, N supplying componentwise transport, replay, provenance, and local naming. Quotient streams, countable choice, trichotomy, selected host limits, and ambient completeness have no coordinate in that route, so they are outside the finite-window scope. □

Theorem 1420.7 (LocatedLimit schedule monotonicity)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, enlarging a displayed precision request refines the stored finite-window schedule through the same modulus ledger* M *and schedule row* T*. The refined request may expose later sequence windows, stronger dyadic tolerances, and a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, but it may not replace* M*, allocate a new schedule row, or bypass the Real-seal route of Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier schedule monotonicity`

Proof. Project the carrier of Definition 1420.1. The only rows that answer precision requests before readback are the modulus ledger M and the finite-window schedule T.

Use Definition 1420.2 to compare the original request and the enlarged request through transported finite windows and dyadic budgets. The classifier has no host limit or quotient-stream coordinate, so the comparison remains inside M and T.

Apply Definition 1420.3 and Definition 1420.5. Their modulus exactness clause places M, T before every read of Q or E, and the exactness theorem identifies this placement as the stored route for every displayed tolerance request.

Finally use Definition 1420.4. Since the Real seal is reached only after S, M, T, Q, a stronger request can only refine the visible windows and tolerances in the same schedule; it cannot introduce a separate modulus or an independent seal. □

Theorem 1420.8 (LocatedLimit schedule-tail cofinality)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, any later displayed precision request that refines the same located schedule must reuse the same modulus ledger* M*, schedule row* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The refined tail may ask for later sequence windows and stronger dyadic tolerances, but it cannot allocate a second modulus, quotient stream, selected limit, choice schedule, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitScheduleTailCofinality`
↪ variant `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier schedule tail cofinality`

Proof. Project the accepted packet through Definition 1420.1. The only tolerance-facing rows are M and T, and the only readback and terminal seal rows available after them are Q and E.

Use Definition 1420.2 to read a refined request. Since the classifier compares accepted packets by transported finite windows and dyadic budgets, a later tail that refines the same schedule is still classified through the same displayed rows rather than through a fresh host limit or quotient stream.

Apply Definition 1420.5. Every tolerance request reaches Q or E only after it has already passed through the stored schedule row T and modulus ledger M, so a tail request has no alternative ledger from which to obtain its budget.

Finally apply Definition 1420.4. The public limit read is exhausted by the route $S, M, T, Q, E, H, C, P, N$, hence the refined tail can only continue along that route. The excluded second modulus, quotient stream, selected limit, choice schedule, and ambient completeness row have no carrier coordinate. □

Theorem 1420.9 (LocatedLimit Real seal comparison)**.** *Let $L = (S, M, T, Q, E, H, C, P, N)$ and $L' = (S', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carriers. If their transported schedule rows, modulus ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows match under the displayed* hsame *and* Cont *routes, then any comparison between the two accepted Real seals is exhausted by those rows. No comparison may appeal to host real equality, quotient equality of streams, a selected limit, countable choice, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier real seal comparison`

Proof. Read both carriers through Definition 1420.1. The only comparison-facing rows are the sequence surfaces, modulus ledgers, schedule rows, regular readbacks, Real seals, transport rows, replay rows, provenance rows, and local naming rows.

Apply the classifier of Definition 1420.2. It compares accepted carriers through shared finite windows, dyadic budgets, readback rows, and Real seal rows after componentwise transport; it does not expose host real equality or quotient stream equality.

Use Definition 1420.3 to force the comparison through the transported schedule and modulus rows before any readback or seal is consumed. Then Definition 1420.5 identifies the finite-window route as the exact route for each displayed precision request.

Finally apply Definition 1420.4. The terminal Real seal is reached only after S, M, T, Q on one side and S', M', T', Q' on the other. Therefore matching transported schedules, modulus ledgers, readbacks, and seal rows exhaust the accepted comparison, and the excluded external principles have no carrier coordinate from which to act. □

Theorem 1420.10 (LocatedLimit Cauchy tail uniqueness)**.** *Let $L = (S, M, T, Q, E, H, C, P, N)$ and $L' = (S', M', T', Q', E', H', C', P', N')$ be accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carriers. Suppose the two carriers expose matching transported finite schedules T, T', modulus ledgers M, M',* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks Q, Q', and terminal* $\mathsf{Real}^{\uparrow}$ *seals E, E' through the displayed* hsame *and* Cont *routes. Then every limit-facing consumer comparison of E and E' is exhausted by the common Cauchy-tail route*

$$S, M, T, Q, E, H, C, P, N.$$

It cannot appeal to host real equality, quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier cauchy tail uniqueness`

Proof. Project both accepted carriers through Definition 1420.1. The only rows that can feed a limit-facing comparison are the displayed sequence surfaces, modulus ledgers, finite schedules, regular readbacks, Real seals, transport rows, replay rows, provenance rows, and local naming rows.

Use Definition 1420.2 to identify the comparison surface. The classifier reads the transported windows T, T', the dyadic budgets M, M', the readbacks Q, Q', and the seals E, E'; it has no

coordinate for host equality, quotient streams, selected limits, countable choice, trichotomy, or ambient completeness.

Apply Definition 1420.5. Every tolerance request reaches Q or E only after the finite schedule and modulus ledger have already been read, so the two matching schedule-ledger pairs determine the Cauchy-tail route visible to the consumer.

Finally use Definition 1420.4 and Definition 1420.9. The terminal seal comparison is reached only through the same finite route, and the comparison theorem says matching transported schedules, ledgers, readbacks, and seals exhaust the accepted Real-seal comparison. Hence the two seals are unique for this accepted route in the stated Cauchy-tail sense, without invoking any ambient Real-completeness principle. □

Theorem 1420.11 (LocatedLimit RegSeqRat readback stability)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is stable under the same componentwise* hsame *transports and displayed* Cont *replay rows* H, C *that preserve the sequence surface* S*, modulus ledger* M*, located schedule* T*, and real seal* E*. The stability surface is the one described by Definition 1420.1, classified by Definition 1420.2, obligated by Definition 1420.3, and consumed by Definition 1420.4.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimit regseqrat readback stability`

Proof. Read the accepted packet through Definition 1420.1. The transport row H is componentwise and includes the regular readback row Q together with S, M, T, E.

The classifier in Definition 1420.2 compares only transported finite windows, dyadic budgets, regular readbacks, and real seals. Therefore a transported Q is not compared through quotient stream equality or through a new host sequence.

Apply the transport obligation in Definition 1420.3. The replay row C rereads the same consumer route after H has moved the displayed coordinates. The Real-seal route of Definition 1420.4 then consumes exactly that stable Q before reaching E. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier regseqrat readback stability`

Theorem 1420.12 (LocatedLimit source-row scope)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, every limit-facing consumer binds the finite-window schedule* T*, modulus ledger* M*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *to the source rows named by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *before the seal can be read. The binding is exactly the source-row route of the carrier, classifier, NameCert obligations, window schedule exactness, Real seal route, and RegSeqRat readback stability.*

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitCarrier source row scope`

Proof. Project the carrier of Definition 1420.1. Its consumer-facing rows are $S, M, T, Q, E, H, C, P, N$, so the schedule, modulus, readback, and seal are already stored with their transport, replay, provenance, and local naming rows.

Use the classifier of Definition 1420.2. It compares accepted packets only through transported finite windows, dyadic budgets, regular readbacks, and Real seals; hence T, M, Q, E cannot be detached from the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ source rows that classify them.

Apply Definition 1420.3, Definition 1420.5, and Definition 1420.4. These rows force every tolerance request through T and M before Q and E are consumed.

Finally use Definition 1420.11. The readback row Q is stable only under the same hsame and Cont rows that preserve S, M, T, E, and P, N record the same source route. Therefore no consumer can read the seal before these BEDC source rows are bound. □

Theorem 1420.13 (LocatedLimit Real seal nonescape)**.** *For every accepted* $\mathsf{LocatedLimit}^{\uparrow}$ *carrier* $L = (S, M, T, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *exports no selected host limit, no quotient stream equality, no countable-choice schedule, no trichotomy, and no ambient completeness principle. This nonescape is the carrier exclusion of Definition 1420.1, the classifier exclusion of Definition 1420.2, the Real-seal obligation of Definition 1420.3, and the terminal factorization of Definition 1420.4.*

Proof. Inspect the carrier in Definition 1420.1. The rows are $S, M, T, Q, E, H, C, P, N$, and none is a selected host limit, quotient equality of streams, countable-choice schedule, trichotomy witness, or ambient completeness field.

The classifier of Definition 1420.2 preserves the same boundary: it compares finite windows, dyadic budgets, regular readbacks, and real seals only after componentwise transport.

Apply Definition 1420.3 to force every real-facing read through the named finite rows. Then Definition 1420.4 identifies E as the sole Real seal reached after S, M, T, Q. Since the excluded principles have no carrier coordinate and no classifier clause, they cannot escape through E. □

Lean4 ✓ `BEDC.Derived.LocatedLimitUp.LocatedLimitRealSealNonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedLimit}^{\uparrow}$ packet is generated from a displayed $\mathsf{Real}^{\uparrow}$ sequence surface, an explicit Cauchy/modulus ledger, a located finite-window schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ limit seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1420.1, Definition 1420.2, Definition 1420.3, Definition 1420.5, Definition 1420.4, Definition 1420.7, Definition 1420.8, Definition 1420.11, Definition 1420.12, and Definition 1420.13 close the mature paper-side finite-window treatment: carrier, classifier, NameCert obligations, window schedule exactness, Real-seal route, schedule monotonicity, tail cofinality, readback stability, source-row scope, and nonescape for located-limit consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, selected host limits, quotient equality of streams, countable choice, trichotomy, or bridge checking.

Upgrade path: The next upgrade is Lean bridge checking for the paper-side located-limit route without adding Real completeness or selected limits.

Chapter 1421

A Concrete Naming Certificate for $\mathsf{CantorIntersection}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CantorIntersection}^{\uparrow}$ packet records the finite nested-interval route by which a displayed shrinking interval chain is consumed as a real-completion surface. It is a NameCert reconstruction of the Cantor nested-interval reading over the finite diagonal interface of chapter 389, with source rows also anchored at chapter 126, chapter 301, and chapter 38.

The chapter names a completion-facing consumer of the existing nested-interval readback surface. It does not add point choice, quotient equality of reals, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness; every real-facing read is required to pass through the displayed nested-interval intersection packet.

Definition 1421.1 (Cantor intersection carrier)**.** *A* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (N, W, D, E, R, H, C, P, Q)$$

with the following visible rows:

1. *N is the accepted* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *packet whose source chain is a displayed* $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *row;*
2. *W is the finite width-window ledger read from the selected* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint packets and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower and upper endpoint rows;*
3. *D is the diagonal selector row choosing, for each displayed precision request, one finite nested-interval window already present in N;*
4. *E records the endpoint enclosure rows selected by D and the exact finite dependency path back to W;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by reading the selected endpoint enclosures as a regular rational completion name;*
6. *H is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R and the nested-interval intersection packet N;*
7. *C stores the displayed* hsame *and* Cont *transports for N, W, D, E, R, H;*

8. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *Q is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$ *row for the finite completion-consumer name.*

The carrier predicate is $\mathsf{CantorIntersectionPacket}(K)$*. Its public rows are exactly the nested-interval intersection packet, width-window ledger, diagonal selector, endpoint enclosure rows, regular-rational handoff, real seal, transport rows, package provenance, and local naming row.*

Theorem 1421.2 (Cantor intersection NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$ *surface has five obligations:*

1. *carrier: admission of the finite packet of Definition 1421.1;*

2. *finite-window classifier: two accepted carriers classify together only by comparing the displayed* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *packet, width-window ledger, selected endpoint enclosures,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal row, transports, provenance, and local certificate row;*

3. *selector stability: transporting the nested-interval intersection packet and width-window ledger through the displayed* hsame *and* Cont *routes transports the selected finite window to the corresponding selected finite window;*

4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness: every regular-rational read in R is obtained from the selected endpoint enclosure rows E and no unrecorded tail of the interval chain;*

5. $\mathsf{Real}^{\uparrow}$ *seal nonescape: every real-facing read through H factors through N, W, D, E, R, C, P, Q and receives no host real equality, point selector, open-cover compactness, or ambient completeness datum.*

These obligations are the finite Cantor-intersection consumer surface over the already named nested-interval intersection route.

Proof. Carrier admission is projection from K. The tuple lists the nested-interval intersection source, width-window ledger, selector, selected endpoint enclosures, regular-rational handoff, real seal, transports, provenance, and local naming row as one finite history packet.

The finite-window classifier compares only those displayed rows after the recorded transports have moved both packets to the same precision request. Since the comparison is made at the visible window, it has no coordinate for quotient equality of completed reals or equality of host interval chains.

Selector stability is inherited from the selector row D and the transported width ledger W. A transported request still names a finite interval row of the transported $\mathsf{NestedIntervalIntersection}^{\uparrow}$ packet, so the selected endpoint enclosure remains a visible row of the same packet.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff exactness is the statement that R is read from E, not from an invisible tail of the chain. The $\mathsf{Real}^{\uparrow}$ seal nonescape then follows by routing H through R together with the displayed transport, package, and local certificate rows. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionCarrier namecert obligation surface`

Theorem 1421.3 (Cantor intersection width-window transport)**.** *For every accepted* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

the cofinal width-window row W is transported only together with the nested-interval intersection packet N, the diagonal selector row D, the selected endpoint enclosure row E, and the displayed

transport rows C. After any displayed hsame *or* Cont *route, a precision request still lands in a finite window already present in N.*

Proof. Project K through Definition 1421.1. The rows N, W, D, E are stored in the same finite packet, and C is the only displayed transport coordinate for them.

Apply Definition 1421.2. The selector-stability obligation says that transporting the nested-interval packet and width ledger transports the selected finite window with them.

Thus the cofinal window remains a visible row of N after transport. No hidden tail, point selector, or ambient completeness datum is introduced by the width-window route. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionCarrier width window transport`

Theorem 1421.4 (Cantor intersection RegSeqRat handoff exactness)**.** *For every accepted* CantorIntersection$^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

the RegSeqRat$^{\uparrow}$ *row R is obtained exactly from the selected endpoint enclosure row E, where E is selected by D from the cofinal width-window ledger W inside N. The handoff does not read a second completion name, an unrecorded interval tail, or quotient equality of real names.*

Proof. Start with the rows W, D, E, R in Definition 1421.1. The endpoint enclosure row E is not free-standing; it is the endpoint row selected by D from the finite ledger W attached to N.

Apply the handoff-exactness clause of Definition 1421.2. That clause identifies R with the regular-rational read from E and excludes any invisible tail of the interval chain.

The displayed transport and package rows C, P, Q only replay that finite route. They do not add a second regular name or quotient-real equality coordinate. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionRegSeqRatHandoffExactness`
↪ variant `BEDC.Derived.CantorIntersectionUp.CantorIntersectionCarrier regseqrat handoff exactness`

Theorem 1421.5 (Cantor intersection real seal nonescape)**.** *For every accepted* CantorIntersection$^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

the Real$^{\uparrow}$ *seal row H may be consumed only through the finite route*

$$N, W, D, E, R, C, P, Q.$$

In particular, the seal cannot be used as a proof of point choice, uniqueness of an intersection point, open-cover compactness, host real equality, quotient equality of real names, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the real-facing row H from K. Its source rows are exactly the selected nested-interval data, the regular-rational handoff, the displayed transports, package provenance, and local naming certificate.

Apply Definition 1421.2. The real-seal nonescape obligation states that every read through H factors through N, W, D, E, R, C, P, Q.

The excluded principles have no coordinate in that route. The seal is therefore a terminal consumer of the finite Cantor-intersection handoff, not an ambient completeness or choice principle. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionRealSealNonescape`

Theorem 1421.6 (Cantor intersection real seal handoff)**.** *For every accepted* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

every $\mathsf{Real}^{\uparrow}$*-facing read factors through the displayed* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *packet* N*, the selected finite nested-window ledger* W, D, E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the displayed transports* C*, the package provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$ *row* Q *before the seal row* H *is consumed.*

The handoff does not claim open-cover compactness, point choice, host real equality, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersection real seal handoff`

Proof. Start with the real-facing row H in Definition 1421.1. Its source is the regular-rational handoff R together with the displayed transports, provenance, and local naming row.

Apply Definition 1421.2. The obligation surface forces R to be read from the selected endpoint enclosures E, and those enclosures are selected by D from the finite width-window ledger W inside the nested-interval intersection packet N.

Thus every real-facing consumer reaches the seal only after passing through the displayed finite nested-interval intersection route. No remaining coordinate in K carries an open-cover compactness proof, a chosen intersection point, a host real equality row, quotient equality of real names, or an ambient completeness principle. □

Theorem 1421.7 (Cantor intersection diagonal cofinal readback)**.** *For every accepted* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

and every displayed precision request named by the finite width-window ledger W*, the diagonal selector row* D *selects a visible interval row already present in the nested-interval intersection packet* N*. The selected row determines the endpoint enclosure row in* E*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *is read from that same enclosure.*

Consequently the $\mathsf{Real}^{\uparrow}$ *seal row* H *is reachable only through the finite cofinal route*

$$W, D \longrightarrow N \longrightarrow E \longrightarrow R \longrightarrow H$$

together with the displayed transports C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$ *row* Q*.*

Proof. Project the accepted carrier K along the rows listed in Definition 1421.1. A displayed precision request is not an external real-completion demand; in the packet it is a row of the finite ledger W.

Apply the selector-stability obligation of Definition 1421.2. After any displayed hsame or Cont transport in C, the row D still points to a finite nested-window row of N for that request. Thus the selector has no access to a hidden tail of the chain and no coordinate for a chosen intersection point.

The endpoint enclosure row E is then the visible enclosure attached to the selected row, and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff exactness clause of Definition 1421.2 forces R to be obtained from E. There is no side input from quotient equality of real names, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally apply Definition 1421.6. The only route to the real-facing seal is the already displayed chain through N, W, D, E, R, C, P, Q, so every precision request consumed by H has already been answered by a finite diagonal row of N. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersection diagonal cofinal readback`

Theorem 1421.8 (Cantor intersection public completion consumer)**.** *For every accepted* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

the public completion consumer surface of K *is exactly the finite route through* $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$ *or* $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$ *row.*

Equivalently, every public consumer row factors through the displayed nested-interval intersection packet N*, the width-window ledger* W*, the diagonal selector* D*, the selected endpoint enclosures* E*, the regular-rational handoff* R*, the real seal row* H*, the transport rows* C*, the package provenance* P*, and the local naming row* Q*. The surface exports no point selector, quotient real equality, open-cover compactness, uniqueness of an intersection point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1421.2. The carrier, classifier, selector-stability, handoff-exactness, and seal-nonescape obligations list exactly the finite rows visible to a $\mathsf{CantorIntersection}^{\uparrow}$ consumer.

The real-facing boundary is fixed by Definition 1421.6. Any read reaching $\mathsf{Real}^{\uparrow}$ must first pass through the nested-interval packet, the selected finite window, the regular-rational handoff, the displayed transports, the provenance row, and the local certificate row.

The finite cofinality of those reads is fixed by Definition 1421.7. Each displayed precision request is answered by a visible diagonal row of N, so the consumer has no access to an unrecorded tail or to an external choice of intersection point.

Combining these three surfaces leaves precisely the listed public route and no further public coordinate. In particular, quotient equality of real names, host real equality, open-cover compactness, uniqueness of an intersection point, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not part of the exported consumer surface. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersection public completion consumer`
Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionPublicCompletionConsumer`

Theorem 1421.9 (Cantor intersection dyadic interval basis bridge)**.** *For every accepted* $\mathsf{CantorIntersection}^{\uparrow}$ *carrier*

$$K = (N, W, D, E, R, H, C, P, Q),$$

each accepted public completion consumer factors its selected endpoint enclosures through a $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *basis-cell route before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row are read. Thus the cofinal window selected by* D *is exposed as a finite dyadic basis-cell boundary over the displayed rows* N, W, D, E, C, P, Q*, and only then as the regular-rational readback* R *and real seal* H*.*

Proof. Begin with the public consumer described in Definition 1421.8. That surface reaches the selected endpoint enclosures only through the nested-interval packet N, the width-window ledger W, and the diagonal selector D.

Apply Definition 1421.7. For each displayed precision request, the selected enclosure in E comes from a visible finite cofinal row rather than from a hidden tail or selected intersection point.

Now apply Definition 1916.3. The selected dyadic endpoint enclosure may be refined through the $\mathsf{DyadicIntervalBasis}^{\uparrow}$ basis-cell route while preserving the transported endpoint, containment, stream-window, regular-rational, and real-seal rows.

Combining these routes gives the required factorization: basis-cell exposure precedes the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row H, while C, P, Q replay, source, and name the same finite carrier. The bridge adds no point selector, quotient real equality, host real equality, open-cover compactness, uniqueness of an intersection point, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CantorIntersectionUp.CantorIntersectionDyadicIntervalBasisBridge`

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersection}^{\uparrow}$ packet is generated from an accepted $\mathsf{NestedIntervalIntersection}^{\uparrow}$ packet, a finite width-window ledger over $\mathsf{NestedInterval}^{\uparrow}$ or $\mathsf{NestedDyadicInterval}^{\uparrow}$ source rows, selected $\mathsf{DyadicInterval}^{\uparrow}$ endpoint enclosures with $\mathsf{DyadicRatCore}^{\uparrow}$ reads, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, displayed hsame and Cont transports, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature paper treatment registers the carrier, finite-window classifier, selector stability, regular-rational handoff exactness, real-seal nonescape, public completion consumer, and dyadic interval basis bridge for a finite Cantor nested-interval completion consumer over $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{DyadicIntervalBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersection}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.CantorIntersectionUp.CantorIntersectionCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, point choice, host real equality, quotient equality of real names, uniqueness of an intersection point, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: No further paper closure upgrade remains; later work may synchronize verification or bridge status without changing the closed paper surface.

Chapter 1422

A Concrete Naming Certificate for $\mathsf{IntermediateValueBisection}^{\uparrow}$

Following Bishop and Bridges-style constructive bisection, the $\mathsf{IntermediateValueBisection}^{\uparrow}$ packet records the finite sign-change route that turns an inspected located interval, endpoint sign reads, and a displayed continuity budget into a regular rational window and a real-name seal. It is a NameCert consumer of the dyadic branch ledger of chapter 361, the rational endpoint cells of chapter 354, the located interval surface of chapter 394, the stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38.

The certificate is narrower than a classical intermediate value theorem. It stores only the finite branch prefix and the sign ledger needed by a consumer at a requested tolerance, then hands the retained window to the existing regular-sequence and real-name surfaces. No coordinate stores a completed root, a quotient stream, an arbitrary limit selector, real trichotomy, open-cover compactness, or an ambient completeness theorem.

Theorem 1422.1 (IntermediateValueBisection finite sign-change carrier)**.** *An* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (J, R, D, S, W, G, E, H, C, P, N)$$

whose visible rows have the following roles:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, including the lower and upper rational cells and their located order witness;*
2. *R is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger against which endpoint and midpoint sign reads are recorded;*
3. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch prefix that retains an interval with opposite displayed signs at the requested finite precision;*
4. *S is the sign ledger, containing the initial endpoint signs and the midpoint sign row used at each branch of D;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window surface for the retained lower and upper endpoint names;*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained stream windows and the requested dyadic tolerance;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal available to downstream consumers only through the retained regular readback;*

8. *H stores the componentwise* hsame *transports for J, R, D, S, W, G, E, and the local certificate row;*
9. *C stores the displayed* Cont *replay routes for endpoint inspection, midpoint inspection, branch retention, regular readback, and real-seal consumption;*
10. *P is the* Pkg *provenance over* LocatedInterval$^{\uparrow}$, RationalInterval$^{\uparrow}$, DyadicBisection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{IntermediateValueBisection}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite sign-change bisection certificate. It has no coordinate for completed root existence, real trichotomy, quotient streams, selected limits, open-cover compactness, or a global completeness principle for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntermediateValueBisectionUp.IntermediateValueBisectionTasteGate single carrier alignment`

Proof. The carrier is read by projection from the displayed packet I. The located interval row J and rational cell ledger R provide the endpoint cells in which signs are inspected.

The dyadic branch prefix D is the only row that can choose a retained interval. Its choices are justified by the sign ledger S, so every branch step is tied to an endpoint or midpoint sign row already present in the packet.

The retained endpoints then expose stream windows through W and regular rational readback through G. The real-facing row E is reached only after that readback route is fixed.

Finally, H, C, P, and N provide transport, replay, provenance, and local naming for the same rows. Since the tuple contains no other analytic coordinate, the excluded classical and completeness principles cannot be exported by the carrier. □

Theorem 1422.2 (IntermediateValueBisection sign-ledger stability)**.** *For every accepted* IntermediateValueBisection *carrier $I = (J, R, D, S, W, G, E, H, C, P, N)$, every branch prefix of D transports the same sign-ledger boundary under the displayed* hsame *rows and the displayed* Cont *replay routes. In particular, if a consumer reads a retained subinterval through C, then the lower and upper sign rows it receives are exactly the transported entries of S attached to the rational cells in R and the located endpoint packet J.*

Proof. Start with a branch prefix of D. Each step in the prefix names one retained dyadic subinterval and one midpoint sign row, so the only available sign data for the next step are the entries already stored in S.

Transport along H acts componentwise on J, R, D, and S. It preserves the attachment between a rational endpoint cell and its displayed sign row because there is no separate sign coordinate outside the ledger.

Replay along C reads the retained subinterval through the same endpoint inspection and midpoint inspection route. Thus a consumer cannot replace the lower or upper sign by an unrelated host comparison; it receives the transported ledger entries bound to the branch prefix.

Package provenance P and the local certificate row N record that this is the accepted IntermediateValueBisection$^{\uparrow}$ route. The branch-prefix and sign-ledger transport are therefore stable under the named hsame, Cont, and Pkg rows. □

Lean4 ✓ `BEDC.Derived.IntermediateValueBisectionUp.IntermediateValueBisectionCarrier sign ledger stability`

Theorem 1422.3 (IntermediateValueBisection regular seal handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *carrier factors through the route*

$$J, R, D, S, W, G, E, H, C, P, N.$$

The consumer first reads the located and rational endpoint cells, then the finite dyadic branch prefix and transported sign ledger, then the retained stream windows, then the regular rational readback, and only then the real-name seal. The handoff exports a finite sign-change window, not a completed root, quotient stream, selected limit, real trichotomy, open-cover compactness, or ambient completeness theorem.

Proof. Project the accepted carrier. A regular-sequence request begins at the retained endpoint windows, but those windows are not independent rows; they are the W coordinate stored with J, R, D, and S.

Read the finite route in order. The located endpoint packet and rational interval cells determine the initial boundary, the dyadic branch prefix selects the retained interval, and Definition 1422.2 keeps the sign ledger attached to that prefix under transport and replay.

The retained windows then pass through G, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The $\mathsf{Real}^{\uparrow}$ seal E is exposed only after this regular readback has been reached, with H, C, P, and N preserving the same route for consumers.

No step in this route introduces a host root or a completed intersection object. Therefore the handoff supplies exactly the finite regular window and real-name seal named in the packet, with the listed non-claims remaining outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntermediateValueBisection}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint packet, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicBisection}^{\uparrow}$ branch prefix, finite sign rows, $\mathsf{StreamName}^{\uparrow}$ retained windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1422.1, Definition 1422.2, and Definition 1422.3 seed the finite constructive bisection route from displayed sign change to regular rational and real-name consumption.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntermediateValueBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close completed root existence, real trichotomy, quotient streams, selected limits, open-cover compactness, completed nested intersections, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: A checked BEDC.Derived.IntermediateValueBisectionUp carrier and theorem surface matching the displayed finite sign-change packet would raise the verification axis.

Chapter 1423

A Concrete Naming Certificate for $\mathsf{RealCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCauchyModulus}^{\uparrow}$ packet records the explicit modulus-controlled read by which a regular rational Cauchy name reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite stream-window surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal boundary of chapter 38, and the generic modulus schedule of chapter 292. The packet keeps the modulus visible as finite certificate data before any Real-facing consumer may read the seal.

Definition 1423.1 (Real Cauchy modulus carrier)**.** *A* $\mathsf{RealCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the precision-indexed* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule assigning each requested dyadic tolerance a finite threshold.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *family of finite windows selected by the thresholds in M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the endpoint and radius comparisons read on those windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose regularity is witnessed only through M, S, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the displayed modulus, window, dyadic, and regular-readback rows have been consumed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic tolerance row, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C records the displayed* Cont *replay from each precision request through M, S, D, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, a selected limit, open-cover compactness, or an unbounded stream read.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier local cert ledger transport`

Theorem 1423.2 (Real Cauchy modulus NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ *surface for Definition 1423.1 consists exactly of carrier exposure for B, precision/modulus exposure through M, finite-window exposure through S, dyadic tolerance exposure through D, regular rational readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Each accepted precision request is classified only by the scheduled threshold in M, the finite windows in S, and the tolerance ledger in D.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier namecert obligations`

Proof. Project the carrier tuple $B = (M, S, D, Q, E, H, C, P, N)$. Each coordinate is a displayed finite row, and the final coordinate names the packet as the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ surface.

Read the analytic rows in order. The precision request first consumes the threshold row M, then the finite windows S, then the dyadic ledger D, and only then the regular readback Q and terminal seal E.

The structural rows do not add an analytic source. They transport, replay, source, and name the same route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface. □

Theorem 1423.3 (Real Cauchy modulus finite-window route)**.** *For every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet $B = (M, S, D, Q, E, H, C, P, N)$ and every displayed precision request, the Real-facing read factors through the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The consumer receives no Real completeness theorem, quotient stream equality, host real equality, countable-choice row, trichotomy row, selected-limit row, open-cover compactness row, hidden selector, or detached Real seal.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.TasteGate.RealCauchyModulusCarrier window modulus route`

Proof. Start with Definition 1423.2. It places the modulus schedule M before the StreamName window S and the dyadic tolerance ledger D.

The regular readback is then forced through the same finite window. The row Q is read only after M, S, D are displayed, so no quotient stream or host equality row can substitute for the scheduled window comparison.

Finally read the Real seal coordinate. The row E is reached after Q, and the rows H, C, P, N only preserve and name that route. This exhausts the advertised finite-window path for the precision request. □

Theorem 1423.4 (Real Cauchy modulus Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RealCauchyModulus}^{\uparrow}$ *carrier can bypass the displayed route*

$$\mathsf{CauchyModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Equivalently, every accepted terminal seal read is attached to the same packet $B = (M, S, D, Q, E, H, C, P, N)$ and cannot be separated from its explicit modulus schedule, finite windows, dyadic tolerance ledger, regular readback, transport, continuation, provenance, and local naming rows.

Proof. Use the finite-window route of Definition 1423.3. It already forces each terminal seal read through M, S, D, Q, E before the structural rows are consumed.

Now inspect the carrier of Definition 1423.1. There is no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, a selected limit, open-cover compactness, a hidden selector, or a detached Real seal.

Thus any purported Real seal read that omits one of the displayed rows is not a read of the accepted carrier. The only terminal route available to a consumer is the one named by B. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier real seal nonescape`

Definition 1423.5 (Real Cauchy modulus classifier)**.** *The* $\mathsf{RealCauchyModulus}^{\uparrow}$ *classifier accepts a packet* $B = (M, S, D, Q, E, H, C, P, N)$ *exactly when it is componentwise* hsame *to a displayed carrier packet*

$$B' = (M', S', D', Q', E', H', C', P', N')$$

with matching rows for the $\mathsf{CauchyModulus}^{\uparrow}$ *schedule,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming certificate. The classifier then replays the same* Cont *route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N)$$

and accepts no packet whose Real-facing read omits one of these coordinates.

Lean4 (definition) `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusClassifier`

Theorem 1423.6 (Real Cauchy modulus threshold stability)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Transporting* B *along the classifier of Definition 1423.5 preserves the scheduled threshold in* M*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row in* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal in* E*. In particular, the transported packet has the same displayed Real-facing route and cannot replace the modulus threshold by an unbounded stream read or a detached Real seal.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier threshold stability`

Proof. Project the accepted packet by Definition 1423.1. The rows M, S, D, Q, E are named coordinates of the same finite BHist tuple, while H records the componentwise hsame transport for those coordinates.

Apply the classifier of Definition 1423.5. Acceptance requires the transported packet to match the displayed carrier component by component, so the threshold row, finite windows, dyadic tolerance ledger, regular readback, and terminal seal are preserved as the rows read by the consumer.

Finally replay the finite-window route of Definition 1423.3. Since the Cont replay is the same route through $M, S, D, Q, E, H, C, P, N$, no unbounded stream read, hidden selector, or detached Real seal can enter during transport. □

Theorem 1423.7 (Real Cauchy modulus ledger exhaustion)**.** *Every accepted Real-facing read of a* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet is exhausted by the nine displayed rows* $M, S, D, Q, E, H, C, P, N$*. The read therefore cannot introduce Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, an unbounded stream read, or any analytic row not already present in the carrier.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier ledger exhaustion`

Proof. Start from the obligation surface in Definition 1423.2. It lists the analytic rows M, S, D, Q, E and the structural rows H, C, P, N as the complete seed surface for the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ packet.

Use Definition 1423.6 to transport any accepted read back to the same classifier route. The transported read still consumes the scheduled threshold, finite windows, tolerance ledger, regular readback, and terminal seal in the displayed order.

It remains to inspect the carrier. By Definition 1423.1, none of Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, an unbounded stream read, or a detached analytic source has a coordinate in B. Hence every accepted Real-facing read is exhausted by the displayed ledger. □

Theorem 1423.8 (Real Cauchy modulus root-unblock scope)**.** *Every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet exposes exactly the rows needed by downstream L10 Real-and-Completion consumers: the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the ambient* BHist *carrier, the* hsame *transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}$ *row* N*. No accepted packet exposes a quotient stream, selected limit, host real equality, countable-choice row, or ambient Real completeness row as part of the unblock surface.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier root unblock scope`

Proof. Project the tuple in Definition 1423.1. Its analytic coordinates are precisely M, S, D, Q, E, and its structural coordinates are precisely H, C, P, N over the finite BHist packet.

The obligation surface of Definition 1423.2 names the same coordinates as the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$ surface. Thus an L10 consumer can read the schedule, window, dyadic, regular-readback, seal, transport, continuation, package, and naming rows without adding any further analytic source.

Finally apply Definition 1423.3 and Definition 1423.7. The route inspection shows that quotient streams, selected limits, host real equality, countable choice, and ambient Real completeness have no coordinate in the accepted packet, so they cannot be part of the root-unblock scope. □

Theorem 1423.9 (Real Cauchy modulus threshold-window totality)**.** *For every accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet and every displayed precision request, the consumer first reads a threshold from* M *and then a finite window from* S *before the dyadic ledger* D*, the regular readback* Q*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is available. This ordering is total for the displayed route: there is no accepted access path from the precision request directly to* D, Q, E*, to a quotient stream, to a selected limit, to host real equality, to countable choice, or to ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier threshold window totality`

Proof. Start with the route theorem Definition 1423.3. It factors every accepted precision request through $M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$ before the structural rows are consumed.

The carrier projection in Definition 1423.1 makes the threshold and window rows named coordinates of the same finite packet. Therefore the consumer cannot obtain the dyadic tolerance ledger, regular readback, or terminal seal without first selecting the scheduled threshold and the corresponding finite window.

The NameCert obligations in Definition 1423.2 and the ledger exhaustion theorem Definition 1423.7 rule out all other sources. Since quotient streams, selected limits, host real equality, countable choice, and ambient Real completeness are absent from the tuple, they cannot provide an alternate totality witness. □

Theorem 1423.10 (Real Cauchy modulus RegSeqRat seal admission)**.** *The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet is admitted to the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *only through the displayed modulus, finite-window, and dyadic route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

No quotient stream equality, selected limit, host real equality, countable-choice principle, or ambient Real completeness theorem can admit Q *to the seal.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier regseqrat seal admission`

Proof. Read the accepted packet through Definition 1423.1. The row Q is a displayed coordinate between the dyadic ledger D and the terminal seal E, not a detached regular sequence or an external limit object.

Use Definition 1423.3. It places Q after the scheduled threshold M, finite window S, and dyadic ledger D, and before the terminal $\mathsf{Real}^{\uparrow}$ seal row E. The continuation row C then replays exactly that admission path.

The obligation list in Definition 1423.2 supplies the same route as local certificate data, while Definition 1423.7 excludes every non-displayed analytic source. Hence Q enters the seal only by the modulus-window-dyadic route, never by quotient streams, selected limits, host real equality, countable choice, or ambient Real completeness. □

Theorem 1423.11 (Real Cauchy modulus precision induction)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For every finite unary precision request, every finite precision refinement of the accepted request is handled by induction on that unary request and preserves the displayed route through*

$$\mathsf{CauchyModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

At each induction stage the refined read consumes only the scheduled modulus row M*, the finite window row* S*, the dyadic ledger row* D*, the regular readback row* Q*, the terminal seal row* E*, and the structural rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier precision induction`

Proof. Project the accepted packet with Definition 1423.1. The precision request is a finite unary object, so the proof proceeds by induction on the displayed request.

In the zero stage, Definition 1423.3 gives the route from M to S, then to D, Q, and E. The classifier rows H, C, P, N replay and name the same route, so no extra analytic coordinate is introduced.

For the successor stage, assume the route has been preserved for the preceding precision request. The next refinement reads the next threshold from M, selects the corresponding finite window in S, extends the tolerance ledger in D, and then replays the regular readback Q before exposing the same terminal $\mathsf{Real}^{\uparrow}$ seal row E.

The replay is the displayed Cont row C, transported componentwise by H and sourced by P, N. The ledger exhaustion theorem Definition 1423.7 rules out Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, or an unbounded stream read at any induction stage. □

Theorem 1423.12 (Real Cauchy modulus window monotonicity)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If a dyadic tolerance request is strengthened inside the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the scheduled threshold in M may enlarge the finite* $\mathsf{StreamName}^{\uparrow}$ *window in S, but the earlier window, dyadic ledger row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row remain present in the accepted carrier route. In particular, the stronger request cannot delete or detach any previously scheduled window, tolerance entry, regular readback, or terminal* $\mathsf{Real}^{\uparrow}$ *seal row.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier window monotonicity`

Proof. Begin with the accepted route of Definition 1423.3. A stronger tolerance is still classified through the same M, S, D, Q, E route, because the classifier of Definition 1423.5 accepts only componentwise matches of those displayed rows.

The threshold selected by M may point to a larger finite window in S. This changes the amount of displayed stream data read for the stronger tolerance, but it does not remove the earlier finite window, since that earlier window is already a coordinate of the accepted BHist packet and is transported by H.

The dyadic ledger and regular readback follow the same containment. The row D records the earlier tolerance entry and the stronger tolerance entry as finite ledger data, while Q is read only after D has been displayed. Replaying C therefore keeps the earlier window, ledger, readback, and terminal seal available to every downstream consumer. □

Theorem 1423.13 (Real Cauchy modulus cofinal window extraction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For any two finite unary precision requests whose second request refines the first, the displayed modulus row M selects a finite* $\mathsf{StreamName}^{\uparrow}$ *window in S for the refined request that is cofinal over the earlier window. The extraction preserves the earlier* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger entry, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier cofinal window extraction`

Proof. Proceed by induction on the finite unary refinement. The base request is handled by threshold-window totality, Definition 1423.9, which forces the read through M and then the displayed finite window in S before D, Q, E are available.

For the refinement step, use precision induction from Definition 1423.11. The stronger request is still classified by the same carrier rows and therefore reads a scheduled threshold from M and a finite window from S.

Apply window monotonicity, Definition 1423.12. The refined window may enlarge the displayed StreamName data, but it preserves the earlier finite window, dyadic tolerance entry, regular readback, and terminal seal.

The continuation row C, transported by H and sourced by P, N, repacks this preserved route as one accepted carrier read. Thus the finite window extracted for the refined request is cofinal over the earlier one, with the tolerance ledger, readback, seal, and structural rows intact. □

Theorem 1423.14 (Real Cauchy modulus seal determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ and $B' = (M', S', D', Q', E', H', C', P', N')$ be accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packets. If their modulus schedules, stream windows, dyadic tolerance ledgers, regular readbacks, terminal seal rows, continuation replays, provenance rows, and local naming rows are componentwise* hsame*, then the local classifier exposes the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route for B and B'.*

Proof. Project both accepted packets. By Definition 1423.5, acceptance already identifies each packet with a displayed carrier through componentwise hsame rows for the analytic and structural coordinates.

Use the assumed componentwise hsame witnesses for M and M', S and S', D and D', and Q and Q'. The threshold schedule, finite window, dyadic ledger, and regular readback are therefore read in the same classifier order for both packets.

It remains to compare the terminal seal route. The witnesses for E and E' identify the exposed $\mathsf{Real}^{\uparrow}$ seal rows, and the witnesses for C, P, N identify the replay, provenance, and local naming rows that present that seal to the consumer.

The Real seal nonescape theorem Definition 1423.4 prevents either packet from exposing a detached seal. Hence the classifier exposes one terminal $\mathsf{Real}^{\uparrow}$ route, carried by the componentwise hsame data, for both accepted packets. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier seal determinacy`

Theorem 1423.15 (Real Cauchy modulus window scope)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Every finite precision request scopes only the displayed rows $M, S, D, Q, E, H, C, P, N$ over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *In particular, the request has no scope over quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient Real completeness, or an unbounded stream read.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier window scope`

Proof. Project the accepted packet through Definition 1423.1. The carrier lists exactly the modulus schedule, stream window, dyadic ledger, regular readback, terminal seal, transport, replay, provenance, and local naming rows.

Apply the route theorem Definition 1423.3 and the threshold-window totality theorem Definition 1423.9. They force a finite precision request to pass through M, then S, then D, then Q, and finally E, with H, C, P, N supplying the local transport and certificate data.

The exhaustion theorem Definition 1423.7 supplies the negative part of the scope claim. Since the non-displayed analytic principles are not rows of the packet and are not reached by the finite route, they cannot enter the finite-window scope. □

Theorem 1423.16 (Real Cauchy modulus tail-threshold determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. For any finite unary precision request, strengthening the dyadic request may enlarge the scheduled finite* $\mathsf{StreamName}^{\uparrow}$ *window, but it cannot replace the threshold selected by M, detach the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, or introduce a selected limit. The tail of the accepted route remains determined by the same modulus-controlled passage through M, S, D, Q, E.*

Proof. Proceed by induction on the finite unary precision request. At the zero stage, Definition 1423.15 identifies the available rows, and Definition 1423.9 fixes the order from threshold to window, ledger, readback, and seal.

For the successor stage, assume the preceding request already follows the displayed route. A stronger dyadic tolerance is still classified by Definition 1423.5, so the request reads the next threshold through the same modulus row M and schedules the corresponding finite window in S.

The monotonicity theorem Definition 1423.12 allows this window to grow while preserving the earlier window, ledger entry, readback row, and terminal seal. The RegSeqRat seal admission theorem Definition 1423.10 then keeps Q attached to the route from D to E.

Finally, Definition 1423.7 rules out selected limits and other non-displayed analytic sources at every induction stage. Thus the stronger request changes only the finite amount of displayed tail data, not the threshold route or the seal handoff. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail threshold determinacy`

Theorem 1423.17 (Real Cauchy modulus RegSeqRat classifier determinacy)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ and $B' = (M', S', D', Q', E', H', C', P', N')$ be accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packets. If the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise* hsame*, then every terminal* $\mathsf{Real}^{\uparrow}$ *seal read factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier route before reaching E or E'.*

In particular, the terminal seal cannot be read directly from the dyadic ledger, from a detached stream representative, from host real equality, from countable choice, or from ambient Real completeness. The unique bridge from the dyadic tolerance ledger to the seal is the displayed $D \to Q \to E$ route transported by H, C, P, N.

Proof. Start with the finite-window route Definition 1423.3. It orders every accepted seal read through M, S, D, Q, E before the structural rows are consumed.

Use threshold-window totality from Definition 1423.9. The terminal route cannot skip from a precision request to D, Q, E; it first reads the scheduled threshold and finite StreamName window.

Now apply RegSeqRat seal admission, Definition 1423.10. It places the readback row Q as the only displayed bridge from the dyadic ledger D to the terminal seal E.

Finally use tail-threshold determinacy, Definition 1423.16. Strengthening the tail request may enlarge the finite window, but it preserves the same modulus-controlled route and keeps the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the seal.

The componentwise hsame assumptions identify the schedules, windows, ledgers, and readback rows of B and B'. Since ledger exhaustion excludes detached stream representatives, host equality, countable choice, and ambient completeness, both terminal seal reads factor through the same $\mathsf{RegSeqRat}^{\uparrow}$ classifier route. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier regseqrat classifier determinacy`

Theorem 1423.18 (Real Cauchy modulus tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *completion consumer of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet must pass through the displayed rows*

$$M, S, D, Q, E, H, C, P, N.$$

Tail-threshold determinacy composes with window monotonicity and seal determinacy: a stronger tail request may enlarge the finite $\mathsf{StreamName}^{\uparrow}$ *window selected by M, but it preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows that transport, replay, source, and name the route. No completion consumer can replace this handoff by quotient stream equality, selected limits, host real equality, countable choice, or ambient Real completeness.*

Proof. Start with the finite route theorem Definition 1423.3. It fixes the consumer route through M, S, D, Q, E before the structural rows are used.

Use window monotonicity from Definition 1423.12. Strengthening the requested tail may enlarge the displayed finite window, but it does not delete the earlier window, ledger entry, readback row, or terminal seal.

Apply seal determinacy from Definition 1423.14. Componentwise hsame agreement on the displayed rows exposes the same terminal $\mathsf{Real}^{\uparrow}$ route to the consumer.

Finally use tail-threshold determinacy, Definition 1423.16. The stronger tail request remains on the same modulus-controlled path, so every $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ completion consumer receives exactly the displayed handoff rows and no non-displayed completion principle. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail handoff`

Theorem 1423.19 (Real Cauchy modulus seal handoff)**.** *Every downstream Real-facing consumer of an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the same displayed finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

The handoff is mediated by the rows H, C, P, N *and cannot be replaced by Real completeness, quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or an unbounded stream read.*

Proof. Begin with the finite-window scope theorem Definition 1423.15. It identifies the only rows available to a Real-facing consumer and fixes the analytic carriers through which the request may pass.

Use Definition 1423.16 for strengthened tail requests. The consumer may require a larger displayed window, but the threshold route, dyadic ledger, RegSeqRat readback, and terminal seal remain tied to the same modulus-controlled passage.

Now combine the original route theorem Definition 1423.3, threshold stability Definition 1423.6, ledger exhaustion Definition 1423.7, and seal determinacy Definition 1423.14. These results show that H transports the displayed rows, C replays the route, P records provenance, and N names the local certificate without adding another Real-facing access path.

The terminal handoff therefore reaches $\mathsf{Real}^{\uparrow}$ exactly at E. The negative alternatives are absent from the packet and blocked by the exhaustion and nonescape theorems, so every accepted downstream consumer receives the seal through the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier seal handoff`

Theorem 1423.20 (Real Cauchy modulus cofinal refinement handoff)**.** *Let* $B = (M, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If* B *is read through a cofinal or monotone modulus schedule supplied by the existing* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *and* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *surfaces, then every downstream Real-facing read still factors through the displayed route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The cofinal or refined schedule may enlarge the finite window selected by M *and replay the same tail route, but it cannot replace the route by quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, Real completeness, or a detached Real seal.*

Proof. Begin with the finite route theorem Definition 1423.3. It fixes the order from the modulus row M to the StreamName window row S, the DyadicRatCore ledger D, the RegSeqRat readback Q, and the terminal Real seal E.

For a stronger precision request, use window monotonicity and tail-threshold determinacy, Definition 1423.12 and Definition 1423.16. These allow the displayed finite window to grow while preserving the threshold route, the dyadic ledger, the regular readback, and the terminal seal handoff.

Now read the sibling cofinal data only as route data. The $\mathsf{CauchyModulusRefinement}^{\uparrow}$ cofinal-window factorization of Definition 885.15 supplies a refinement row for the finite-window schedule, and the $\mathsf{CofinalModulusSeal}^{\uparrow}$ cofinal route of Definition 1198.12 supplies the cofinal seal row. Neither sibling inserts a second Real seal into the $\mathsf{RealCauchyModulus}^{\uparrow}$ carrier.

Finally apply the seal handoff theorem Definition 1423.19. Since every Real-facing consumer of B reaches $\mathsf{Real}^{\uparrow}$ only through M, S, D, Q, E and the support rows H, C, P, N, the cofinal or monotone schedule preserves the same StreamName–DyadicRatCore–RegSeqRat–Real seal route and leaves the excluded completeness, choice, quotient, and host-equality alternatives outside the packet. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier cofinal refinement handoff`

Theorem 1423.21 (Real Cauchy modulus bounded tail-ledger factorization)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet, and let*

$$R_{\mathrm{bd}} = (S', M', B', Q', A', H', C', P', N')$$

be an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet consumed with componentwise matching modulus, window, dyadic ledger, RegSeqRat readback, and Real seal rows. Then every bounded tail-ledger read exported by the joint consumer factors first through the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

and then through the bounded packet route

$$S' \longmapsto M' \longmapsto B' \longmapsto Q' \longmapsto A'.$$

The handoff exports no Real completeness theorem, quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, or host real equality.

Proof. Begin with the cofinal refinement handoff Definition 1423.20. It shows that strengthened or cofinal tail requests still read the $\mathsf{RealCauchyModulus}^{\uparrow}$ packet through M, S, D, Q, E before any Real-facing seal is reached.

Next apply the seal handoff theorem Definition 1423.19. The terminal row E is therefore not a detachable Real object; it is the endpoint of the displayed modulus, window, dyadic-ledger, and readback route.

Now pass to the bounded consumer. The obligation surface of Definition 1697.2 identifies the bounded packet rows $S', M', B', Q', A', H', C', P', N'$, and the Real-seal boundary of Definition 1697.3 forces every bounded read through M', S', B', Q', A' before the terminal seal A' can be consumed.

The assumed componentwise matches identify the shared modulus, window, dyadic, readback, and seal rows at the interface. Thus a bounded tail-ledger read has only the composite finite passage displayed above, while the excluded completeness, quotient, choice, compactness, trichotomy, selected-limit, and host-equality coordinates remain absent from both carriers. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier bounded tail ledger factorization`

Theorem 1423.22 (Real Cauchy modulus bounded readback coherence)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet and let $R_{\mathrm{bd}} = (S', M', B', Q', A', H', C', P', N')$ be an accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet. If the shared modulus, finite window, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows agree componentwise by* hsame*, then the* $\mathsf{RealCauchyModulus}^{\uparrow}$

readback and the BoundedRegularCauchy$^{\uparrow}$ *readback expose the same terminal finite Real-facing route. In particular, the bounded consumer cannot introduce a detached seal beyond the matched E/A' boundary.*

Proof. Use seal determinacy Definition 1423.14 to pin the terminal Real$^{\uparrow}$ row of the RealCauchyModulus$^{\uparrow}$ packet to the displayed M, S, D, Q route. Ledger exhaustion Definition 1423.7 then prevents a second dyadic or readback ledger from being inserted between Q and E.

On the bounded side, apply tail-window induction Definition 1697.4. Each requested finite threshold is read through the same bounded route to A', and the induction has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or Real completeness.

The componentwise hsame assumptions transport the modulus, window, dyadic ledger, readback, and seal rows across the interface. Hence the two readbacks terminate at the same finite Real-facing route, and any alleged detached seal would have to be a row not present in either displayed carrier. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier bounded readback coherence`

Theorem 1423.23 (Real Cauchy modulus uniform tail extraction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* RealCauchyModulus$^{\uparrow}$ *packet whose threshold row has passed through the cofinal refinement handoff of Definition 1423.20. Then there is a single displayed finite tail window, read from the same M, S, D, Q, E route, that every* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *consumer named in the bounded readback coherence theorem Definition 1423.22 may use. Enlarging a requested tolerance only replaces the displayed window by a later cofinal window; it does not introduce quotient stream equality, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, Real completeness, or a detached Real seal.*

Proof. Start with threshold stability, Definition 1423.6. It transports a stronger tolerance request along the same modulus row M, finite window row S, dyadic ledger D, RegSeqRat readback Q, and terminal Real seal E.

Next use the cofinal refinement handoff, Definition 1423.20. A later cofinal threshold may replace the initially requested tail window, but the replacement is still a displayed finite window in the same carrier route and is replayed by the same structural rows H, C, P, N.

Finally apply bounded readback coherence, Definition 1423.22. Its componentwise hsame matches force the RegSeqRat and Real-facing bounded consumers to read the same terminal finite route, so the cofinal window obtained above is uniform for all those consumers.

The excluded alternatives have no coordinate in either carrier and cannot be inserted between the matched readback and seal rows. Therefore the accepted packet exposes one reusable finite tail window for the bounded RegSeqRat and Real consumers. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier uniform tail extraction`

Theorem 1423.24 (Real Cauchy modulus finite-window tightening)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* RealCauchyModulus$^{\uparrow}$ *packet. Suppose a displayed precision request is tightened and the cofinal refinement row of Definition 1423.20 is available for the same packet. Then the resulting later finite* StreamName$^{\uparrow}$ *window still factors through the original route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The tightened request keeps the same CauchyModulus$^{\uparrow}$ *row M,* DyadicRatCore$^{\uparrow}$ *ledger D,* RegSeqRat$^{\uparrow}$ *readback Q, and* Real$^{\uparrow}$ *seal E. It introduces no detached seal, selected limit, quotient stream, host*

real equality, countable-choice schedule, open-cover compactness, trichotomy, or ambient completeness row.

Proof. Start with the cofinal refinement handoff Definition 1423.20. It permits the tightened precision request to move to a later finite window while preserving the route through M, S, D, Q, E.

Use uniform tail extraction, Definition 1423.23. The later window is not a new endpoint; it is the finite tail window that every named $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer may reuse through the same structural rows.

Apply bounded readback coherence from Definition 1423.22. It keeps the readback row Q and the terminal seal E componentwise matched with any bounded consumer that reads the tightened window.

Finally use ledger exhaustion, Definition 1423.7. Since the dyadic tolerance ledger is already exhausted by D before Q and E are read, there is no remaining coordinate for a detached seal, selected limit, quotient stream, host real equality, choice schedule, compactness proof, trichotomy principle, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier finite window tightening`

Theorem 1423.25 (Real Cauchy modulus obligation closure package)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. The carrier of Definition 1423.1, classifier of Definition 1423.5, NameCert obligations of Definition 1423.2, finite-window route of Definition 1423.3, threshold stability of Definition 1423.6, ledger exhaustion of Definition 1423.7, cofinal refinement handoff of Definition 1423.20, uniform tail extraction of Definition 1423.23, bounded readback coherence of Definition 1423.22, and finite-window tightening of Definition 1423.24 jointly supply the obligation surface for the nine-row packet B. The package introduces no Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, or open-cover compactness.*

Proof. Project the accepted packet through the carrier and classifier. The only visible rows are the modulus row M, finite $\mathsf{StreamName}^{\uparrow}$ window row S, dyadic tolerance ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, terminal $\mathsf{Real}^{\uparrow}$ seal E, componentwise transport H, displayed replay C, provenance P, and local naming row N.

Apply the NameCert obligations and finite-window route. These two sites bind the local naming surface to the displayed route from modulus requests through finite windows, dyadic tolerance data, readback, and the Real seal.

Use threshold stability, cofinal refinement handoff, uniform tail extraction, bounded readback coherence, and finite-window tightening. Together they show that stronger precision requests only move to later displayed windows on the same route and that every bounded consumer reuses the same readback and seal rows.

Finally apply ledger exhaustion. Since the dyadic ledger is already exhausted before the readback and terminal seal are consumed, there is no remaining carrier coordinate for Real completeness, quotient streams, host equality, countable choice, trichotomy, selected limits, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier obligation closure package`

Theorem 1423.26 (Real Cauchy modulus finite modulus-index induction)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Every finite family of precision requests indexed by a*

unary finite window is handled by induction on the displayed modulus-index spine. At each successor step the same route

$$M \to S \to D \to Q \to E \to H, C, P, N$$

is preserved, and the induction exposes no unbounded stream read, choice schedule, quotient stream equality, selected limit, or Real completeness row.

Proof. Start with precision induction from Definition 1423.11. The base request reads the displayed modulus row M, the finite window row S, the dyadic ledger D, the regular readback Q, and the terminal seal E through the transported and replayed rows H, C, P, N.

For the successor step, apply window monotonicity from Definition 1423.12. Enlarging the unary finite window preserves the earlier request rows and moves only to a later displayed modulus index on the same spine.

Use uniform tail extraction from Definition 1423.23 to reuse the later finite tail for the whole indexed family. This is finite induction on the BEDC request spine, not a host choice over all precisions.

Finally apply ledger exhaustion from Definition 1423.7. Since every step is already routed through $M, S, D, Q, E, H, C, P, N$, there is no coordinate for an unbounded stream read, quotient stream equality, selected limit, countable-choice schedule, or ambient Real-completeness argument. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier finite modulus index induction`

Theorem 1423.27 (Real Cauchy modulus tail-coherence completeness handoff)**.** *Let $B = (M, S, D, Q, E, H, C, P,$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. Any finite tail-coherence read that is passed from B to a completion witness must preserve the displayed route through* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The handoff may tighten a finite window or move to a later cofinal tail window, but it may not replace the route with quotient stream equality, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or ambient Real completeness.*

Proof. Begin with precision induction, Definition 1423.11. It makes each requested tail threshold a finite step on the displayed modulus spine rather than a global choice over all precisions.

Apply window monotonicity, Definition 1423.12. Enlarging the finite window keeps the same $M, S, D, Q, E, H, C, P, N$ route and only moves to a later displayed window.

Use uniform tail extraction, Definition 1423.23. The tail-coherence read exported to a completion witness is the same finite tail window already available to every named $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer.

Finally apply ledger exhaustion, Definition 1423.7. The dyadic tolerance data has no unused coordinate from which quotient stream equality, selected limits, countable choice, host equality, trichotomy, compactness, or ambient completeness could be inserted into the handoff. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier tail coherence completeness handoff`

Falsifiable prediction: An accepted $\mathsf{RealCauchyModulus}^{\uparrow}$ consumer that reads a terminal Real seal without the explicit modulus threshold, finite StreamName window, DyadicRatCore tolerance ledger, and RegSeqRat readback refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchymodulus: no listed sibling supplies the whole nine-row packet $B = (M, S, D, Q, E, H, C, P, N)$, because each sibling

contributes only a window, readback, tolerance, seal, or generic schedule surface while this chapter binds all five analytic faces with transport, replay, provenance, and local naming.

Theorem 1423.28 (Real Cauchy modulus uniform cofinal seal)**.** *Let $B = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet. If the modulus row M is transported through the sibling lattice*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyModulus}^{\uparrow}$$

named by chapter 292, then every finite precision request yields a classifier-stable cofinal seal through the displayed route

$$M \to S \to D \to Q \to E \to H, C, P, N.$$

The exported row is uniform over finite precision requests and contains no selected real representative, host convergence principle, quotient stream equality, countable-choice schedule, or detached Real seal.

Proof. Start with finite modulus-index induction from Definition 1423.26. It handles a finite family of precision requests by induction on the displayed modulus-index spine while preserving the route through $M, S, D, Q, E, H, C, P, N$.

Read the sibling lattice as carrier data. The StreamName row supplies the finite windows, the RegSeqRat row supplies the regular readback, the DyadicRatCore row supplies the tolerance ledger, the Real row supplies the terminal seal, and the CauchyModulus chapter cited above supplies the schedule discipline for cofinal requests.

Now apply cofinal refinement handoff and uniform tail extraction. A later cofinal threshold may enlarge the finite window, but the transported request still reaches the same terminal route and is replayed by the same classifier rows H, C, P, N.

Finally use ledger exhaustion. Since every precision request has already consumed the displayed modulus, window, tolerance, readback, and seal rows, the cofinal seal cannot import a selected representative, host convergence theorem, quotient stream equality, countable-choice schedule, or detached Real seal. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusUp.RealCauchyModulusCarrier uniform cofinal seal certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyModulus}^{\uparrow}$ packet is generated from an explicit $\mathsf{CauchyModulus}^{\uparrow}$ precision schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1423.1, Definition 1423.5, Definition 1423.2, Definition 1423.3, Definition 1423.4, Definition 1423.6, Definition 1423.7, Definition 1423.8, Definition 1423.9, Definition 1423.10, Definition 1423.11, Definition 1423.12, Definition 1423.14, Definition 1423.15, Definition 1423.16, Definition 1423.17, Definition 1423.18, Definition 1423.19, Definition 1423.20, Definition 1423.21, Definition 1423.22, Definition 1423.23, Definition 1423.24, Definition 1423.27, and Definition 1423.28 seed the explicit modulus-controlled Real Cauchy read over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedRegularCauchy}^{\uparrow}$, and the uniform cofinal seal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, open-cover compactness, or a Lean theorem.

Upgrade path: Mature closure requires a checked standard bridge for the nine-row carrier and finite-window Real seal route.

Chapter 1424

A Concrete Naming Certificate for $\mathsf{DedekindCauchyComparison}^{\uparrow}$

Following Bishop and Bridges and Troelstra–van Dalen, the $\mathsf{DedekindCauchyComparison}^{\uparrow}$ packet records the finite comparison boundary between rounded located cut observations and regular Cauchy name observations. It reads the cut surface of chapter 436, the Cauchy equivalence surface of chapter 1412, the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a representation comparison certificate, not a Dedekind construction of $\mathsf{Real}^{\uparrow}$ and not a completeness theorem.

Definition 1424.1 (DedekindCauchyComparison carrier)**.** *A* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, R, W, T, E, H, C, P, N)$$

with the following displayed coordinates:

1. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *row, carrying lower and upper rational-observation data together with roundedness, locatedness, and disjointness evidence;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, carrying the regular rational sequence surface that is allowed to answer finite tolerance requests;*
3. *W is the family of finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which R is read by downstream consumers;*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, comparison budgets, and finite refinement rows used to compare D with R;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row reached from the displayed cut and regular-sequence observations, with no independent completed-real coordinate;*
6. *H stores componentwise* hsame *transports for D, R, W, T, E;*
7. *C stores the displayed* Cont *replay routes for rereading the same finite cut observations, stream windows, tolerance rows, and seal boundary rows;*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a quotient cut, a host real equality proof, a selected limit, countable choice, a construction of $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DedekindCauchyComparisonUp.DedekindCauchyComparisonTasteGate single carrier alignment`

Theorem 1424.2 (DedekindCauchyComparison NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}}$ *surface for the carrier of Definition 1424.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the packet* $K = (D, R, W, T, E, H, C, P, N)$ *with its displayed cut row, regular sequence row, stream windows, dyadic tolerance ledger, seal boundary, transports, continuations, package provenance, and local naming evidence;*
2. *classifier: comparison reads only rational cut observations in* D*, finite regular-sequence observations in* R*, the windows* W*, and the dyadic budgets* T*;*
3. *transport stability:* H *and* C *preserve the same cut observations, sequence windows, tolerance rows, and seal boundary under accepted consumer motion;*
4. *exactness: every comparison-facing read is supplied by the displayed rows* D, R, W, T, E *after the componentwise transport and replay routes are fixed;*
5. *ledger: every downstream* $\mathsf{Real}^{\uparrow}$ *consumer first reads the comparison packet through* $D, R, W, T, E, H, C, P, N$ *and receives no hidden completion coordinate.*

Proof. Carrier admission is projection from the packet $K = (D, R, W, T, E, H, C, P, N)$. The displayed cut row, regular sequence row, stream-window family, dyadic tolerance ledger, seal boundary, transports, continuations, provenance, and local naming evidence are exactly the coordinates named by the carrier.

The classifier is finite because every comparison request is routed through the stored rational observations in D, the finite windows W of the regular sequence row R, and the dyadic tolerance ledger T. No classifier step can inspect a quotient cut or an unbounded real object.

Transport stability uses H componentwise and then replays the same consumer path through C. The replay returns to the stored rows D, R, W, T, E and therefore preserves the same finite observation surface.

Exactness and the ledger clause follow from the same inventory. Since the carrier contains no selected limit, host real equality proof, countable-choice schedule, Dedekind construction row for $\mathsf{Real}^{\uparrow}$, or completeness principle, every downstream comparison is exhausted by the displayed packet rows. □

Theorem 1424.3 (DedekindCauchyComparison Real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *carrier is only a representation boundary between the displayed* $\mathsf{DedekindCut}^{\uparrow}$ *row and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row. The boundary supplies no quotient cut, no host real equality, no selected limit, no countable choice, no Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, and no completeness theorem.*

Proof. Project the accepted carrier. Its real-facing coordinate is the seal boundary row E, and E is stored together with the cut row D, regular rational row R, finite stream-window family W, and dyadic tolerance ledger T.

Read a comparison request in finite order. The consumer first exposes rational cut observations from D, then reads the corresponding finite windows W of R, and then checks the dyadic budgets recorded in T.

Only after those rows have been fixed can the consumer read the seal boundary E. The transport and replay rows H and C preserve that same route, while P and N record provenance and local naming.

Thus the packet exports a comparison boundary only. A quotient cut, host real equality proof, selected limit, countable-choice schedule, Dedekind construction of $\mathsf{Real}^{\uparrow}$, and completeness theorem have no carrier coordinate and cannot be read from the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCauchyComparison}^{\uparrow}$ packet is generated from a $\mathsf{DedekindCut}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1424.1, Definition 1424.2, and Definition 1424.3 seed the finite representation-comparison boundary between Dedekind-cut observations and regular Cauchy name observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCauchyComparison}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DedekindCauchyComparisonUp.DedekindCauchyComparisonTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient cuts, host real equality, selected limits, countable choice, Dedekind construction of $\mathsf{Real}^{\uparrow}$, completeness, or bridge checking.

Upgrade path: Scoped closure requires a checked finite carrier theorem binding the displayed cut row, regular sequence row, stream windows, dyadic tolerance ledger, and Real-seal boundary.

Chapter 1425

A Concrete Naming Certificate for RegularCauchyCompletionUniversal$^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the RegularCauchyCompletionUniversal$^{\uparrow}$ packet isolates the regular-Cauchy universal-property seed for the L10 Real and Completion route. It refines the general CompletionUniversalProperty$^{\uparrow}$ surface of chapter 397 by forcing every extension and uniqueness read through the finite regular-name chain already exposed by chapter 652, chapter 1404, chapter 126, chapter 301, chapter 291, and chapter 38.

The purpose of the packet is local. It names the rows needed for extending a regular-Cauchy source route into the terminal Real$^{\uparrow}$ seal and for comparing any two such extensions by the displayed hsame ledger. It does not claim the full ambient completeness of Real$^{\uparrow}$, nor does it replace the mature general completion chapter.

Definition 1425.1 (RegularCauchyCompletionUniversal carrier)**.** *A* RegularCauchyCompletionUniversal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, D, R, A, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the regular source row whose admissible names are read as* RegSeqRat$^{\uparrow}$ *data;*
2. *W is the* StreamName$^{\uparrow}$ *finite-window schedule used by the same regular source;*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to those windows;*
4. *R is the* RegularCauchyEquivalence$^{\uparrow}$ *comparison row for replacing regular names without leaving the finite route;*
5. *A is the terminal* Real$^{\uparrow}$ *seal reached through the displayed regular source, window, tolerance, and equivalence rows;*
6. *E is the extension route row, read as the* CompletionUniversalProperty$^{\uparrow}$ *extension face restricted to S, W, D, R, A;*
7. *L is the uniqueness* hsame *ledger comparing two extension routes over the same regular source, finite windows, dyadic ledger, equivalence row, and terminal seal;*
8. *H records the componentwise* hsame *transports for S, W, D, R, A, E, L;*
9. *C records the displayed* Cont *routes that replay the extension and uniqueness reads;*

10. *P is the* Pkg *provenance over* CompletionUniversalProperty$^{\uparrow}$, RealCompletionWitnessExtractor$^{\uparrow}$, RegularCauchyEquivalence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{RegularCauchyCompletionUniversal}^{\uparrow}}$ *row.*

The accepted carrier stores no quotient stream, no selected host limit, no host real equality row, no countable-choice selector, and no ambient Real$^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversalTasteGate single carrier alignment`

Theorem 1425.2 (Regular extension route)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet. Every extension consumer of E first reads the regular source S, then the finite* StreamName$^{\uparrow}$ *windows W, then the* DyadicRatCore$^{\uparrow}$ *ledger D, then the* RegularCauchyEquivalence$^{\uparrow}$ *comparison row R, and only then reaches the terminal* Real$^{\uparrow}$ *seal A and the restricted* CompletionUniversalProperty$^{\uparrow}$ *extension face.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversal extension route`

Proof. Project the carrier of Definition 1425.1. The extension row E is not stored as a detached completion theorem; it appears after the regular source, the scheduled finite windows, the dyadic ledger, the equivalence row, and the terminal Real seal.

Read the sibling surfaces in the displayed order. The StreamName, RegSeqRat, DyadicRatCore, RegularCauchyEquivalence, and Real chapters provide exactly the finite route named by S, W, D, R, A, while the general completion chapter supplies the extension face only after that route has been exposed.

The continuation row C can replay only those displayed reads, and the provenance row P records no selected limit or host completion object. Therefore every extension consumer factors through the regular-Cauchy route before it reaches the universal-property face. □

Theorem 1425.3 (Regular uniqueness ledger)**.** *For accepted packets $U = (S, W, D, R, A, E, L, H, C, P, N)$ and $U' = (S, W, D, R, A, E', L', H', C', P, N')$ over the same regular source, finite windows, dyadic ledger, equivalence row, terminal seal, and provenance, any comparison between the exported extension routes is exhausted by the displayed uniqueness ledgers L and L' together with their componentwise* hsame *transports. No quotient-stream equality, selected host limit, host real equality, countable choice, or ambient completeness row can contribute to the comparison.*

Proof. Start with the shared coordinates S, W, D, R, A, P. They fix the regular source, scheduled windows, tolerance ledger, regular-Cauchy comparison row, and terminal Real seal for both packets.

The only coordinates that can distinguish the two extension exports are E, L, H, C, N and E', L', H', C', N'. The ledgers L and L' compare the extension rows over the same shared route, while H, H' and C, C' display the permitted transports and replays.

Since neither carrier contains a quotient stream, selected host limit, host real equality row, countable-choice selector, or ambient completeness coordinate, no external equality principle can enter the comparison. The uniqueness judgement is exactly the local hsame ledger over the shared regular-Cauchy route. □

Theorem 1425.4 (Regular section law)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet. If a consumer first includes the displayed regular source*

through S, W, D, R into the terminal $\mathsf{Real}^{\uparrow}$ *seal A and then reads the restricted extension face E, the exported route is compared by the local uniqueness ledger L to the original A-sealed route. No quotient stream, selected limit, host real equality, countable choice, or ambient completeness row mediates this section comparison.*

Proof. Project the carrier of Definition 1425.1. The displayed source inclusion has only the coordinates S, W, D, R, A available before the extension face is read.

Apply Definition 1425.2. It forces every consumer of E to pass through the source row, finite windows, dyadic ledger, regular-Cauchy comparison row, and terminal Real seal in that order, so the restricted extension face is attached to the same finite route as the original A-sealed read.

Apply Definition 1425.3 with the shared coordinates fixed by this route. The ledger L, together with the displayed transports H and replays C, is the only comparison surface for the extension export over S, W, D, R, A.

The remaining rows P and N record provenance and the local NameCert seal, and no carrier coordinate stores a quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness witness. Thus L, H, C, P, N exhaust the section comparison inside the packet. □

Theorem 1425.5 (Regular Real-seal factorization)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of U must read the finite route*

$$S \longrightarrow W \longrightarrow D \longrightarrow R \longrightarrow A$$

before it may inspect the extension face E or the uniqueness ledger L. No quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate can factor the same accepted read.*

Proof. Project the carrier of Definition 1425.1. The Real-facing rows appear as the regular source S, finite windows W, dyadic ledger D, regular-Cauchy comparison row R, and terminal Real seal A.

Apply Definition 1425.2 to the extension face E. It forces every extension read to pass through S, W, D, R, A before the restricted completion face is available.

Apply Definition 1425.3 to comparisons that use L. The ledger compares extension routes only over the same displayed source, window, dyadic, comparison, and Real-seal rows.

The remaining coordinates H, C, P, N transport, replay, source, and name this route. Since the carrier stores no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness row, coordinate exhaustion gives the stated factorization boundary. □

Theorem 1425.6 (RegularCauchyCompletionUniversal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCor}}$ *surface consists exactly of carrier admission, the regular extension route, the uniqueness ledger, the section law, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming over* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$*,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

Proof. Carrier admission is the packet shape of Definition 1425.1. Its rows name the regular source, finite windows, dyadic ledger, comparison row, terminal Real seal, extension route, uniqueness ledger, componentwise transports, continuation replays, provenance, and local certificate.

The regular extension route is bound by Definition 1425.2. The uniqueness ledger is bound by Definition 1425.3, and the section law is bound by Definition 1425.4.

No remaining coordinate is available in the carrier. Therefore the displayed rows exhaust the local NameCert obligation surface over the listed kernel and sibling objects. □

Theorem 1425.7 (RegularCauchyCompletionUniversal kernel scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. The extension, uniqueness, and section obligations of Definition 1425.6 are scoped through the displayed kernel rows* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*: the only admissible reads are carrier projection from U, componentwise comparison through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier $U = (S, W, D, R, A, E, L, H, C, P, N)$. The rows H, C, P, N are the only structural coordinates after the source, window, ledger, equivalence, seal, extension, and uniqueness entries have been displayed.

The extension route can therefore be replayed only through the Cont rows stored in C. The uniqueness and section comparisons can be transported only through the componentwise hsame rows stored in H.

The remaining structural access is provenance and naming. The rows P and N bind the route to Pkg and NameCert, so the kernel scope is exhausted by the displayed BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 1425.8 (RegularCauchyCompletionUniversal real seal scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. Every real-facing obligation in the extension, uniqueness, and section surface is scoped through the sibling rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*: W supplies the finite windows, S supplies the regular source, D supplies the dyadic ledger, R supplies the equivalence boundary, and A supplies the terminal Real seal.*

Proof. Project U and read the real-facing coordinates in order. The source row S is usable only with the finite windows W, and the tolerance row D is the only dyadic ledger available before equality-facing comparison.

The comparison row R is the displayed $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ boundary. It is reached after S, W, D and before the terminal seal, so no quotient equality or host real equality can intervene.

The row A is the only $\mathsf{Real}^{\uparrow}$ seal in the carrier. Hence every real-facing extension, uniqueness, and section read is sealed by A after the StreamName, RegSeqRat, DyadicRatCore, and RegularCauchyEquivalence rows have been displayed. □

Theorem 1425.9 (RegularCauchyCompletionUniversal extractor scope)**.** *Let $U = (S, W, D, R, A, E, L, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *packet. The completion-facing obligations are scoped through the displayed* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *and restricted completion row: the extractor is read only after S, W, D, R, A, the extension face is E, and the section comparison is the local ledger L transported through H and replayed through C.*

Proof. Project the carrier and isolate the completion-facing rows. The extractor-facing source data are exactly S, W, D, R, A, already sealed as the regular-Cauchy route of Definition 1425.8.

The extension face E is downstream of those rows. It cannot read a quotient completion, a selected host limit, or an ambient completeness theorem because none of those objects has a coordinate in U.

The section comparison is then local. The ledger L compares the extension route with the original Real-sealed route, while H and C provide the only permitted transport and replay rows for that comparison. □

Theorem 1425.10 (RegularCauchyCompletionUniversal public export)**.** *Every public consumer of an accepted* RegularCauchyCompletionUniversal$^{\uparrow}$ *packet receives exactly the extension route, uniqueness ledger, section law, componentwise transport, continuation replay, provenance, and local* NameCert *rows scoped by the displayed kernel rows of Definition 1425.7, the Real-seal rows of Definition 1425.8, and the completion-extractor rows of Definition 1425.9. The exported surface contains no quotient stream, selected host limit, host real equality, countable choice, ambient RealUp completeness, or standard bridge row.*

Proof. Start with the accepted carrier $U = (S, W, D, R, A, E, L, H, C, P, N)$. The kernel scope theorem fixes the only structural rows visible to a consumer: carrier projection through U, transport through H, replay through C, provenance through P, and local naming through N.

Next read the Real-facing route. The Real-seal scope theorem requires every extension, uniqueness, or section read to pass through S, W, D, R, A before any completion-facing row is visible, so a public consumer cannot substitute quotient stream equality or host real equality for the displayed seal.

Finally apply the completion-extractor scope theorem. It identifies the completion-facing export as the restricted extension row E together with the local uniqueness and section ledger L transported by H and replayed by C. Thus the public export is exactly the listed route, ledger, law, transport, replay, provenance, and local certificate rows. □

Closure status for NameCert$_{\text{RegularCauchyCompletionUniversal}^{\uparrow}}$.

Constructive story: A RegularCauchyCompletionUniversal$^{\uparrow}$ packet is generated from a regular source row, StreamName windows, a dyadic tolerance ledger, a RegularCauchyEquivalence row, a RealUp seal, an extension route, a uniqueness hsame ledger, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionUniversal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1425.7, Definition 1425.8, Definition 1425.9, and Definition 1425.10 bind carrier admission, extension, uniqueness, section, transport, replay, provenance, naming, public consumer export, and the restricted completion-interface bridge to the displayed kernel, Real-route, and completion-extractor rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionUniversal}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyCompletionUniversalUp.RegularCauchyCompletionUniversalTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, countable choice, ambient RealUp completeness, or a standard bridge.

Upgrade path: No further paper-side theory-axis upgrade remains; formal bridge checking remains on the independent verification axis.

Chapter 1426

A Concrete Naming Certificate for $\mathsf{CauchyControlSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyControlSequence}^{\uparrow}$ packet names the explicit control data carried by a regular Cauchy real-name route before the terminal $\mathsf{Real}^{\uparrow}$ seal is read. It is the finite NameCert surface that exposes the dependency route through chapter 291, chapter 126, chapter 301, chapter 38, and chapter 1423.

The packet separates the displayed control row from the sealed real object. A consumer receives a scheduled stream window, a dyadic tolerance row, a regular rational readback row, the control/modulus row that binds these data, and the provenance rows needed for local naming; it does not receive a host completeness principle or a chosen limit.

Definition 1426.1 (CauchyControlSequence carrier)**.** *A* $\mathsf{CauchyControlSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(R, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational name whose Cauchy control is being exposed;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect the same regular name;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row governing the displayed window bounds and dyadic comparison budget;*
4. *M is the control/modulus row, read as the local* $\mathsf{RealCauchyModulus}^{\uparrow}$ *surface binding the windows W, tolerances D, and regular readback R;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after the readback, window, tolerance, and control rows have been displayed;*
6. *H records the componentwise* hsame *transports for R, W, D, M, E;*
7. *C records the displayed* Cont *routes by which consumers replay the readback, finite-window, tolerance, control, and seal handoff;*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}$ *row.*

The carrier stores no quotient stream equality, no selected limit, no trichotomy branch, and no ambient completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyControlSequenceUp.TasteGate.CauchyControlSequenceTasteGate single carrier alignment`

Theorem 1426.2 (CauchyControlSequence NameCert obligations)**.** *For an accepted* $\mathsf{CauchyControlSequence}^{\uparrow}$ *carrier* $(R, W, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier formation for the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displayed control/modulus row,* $\mathsf{Real}^{\uparrow}$ *seal handoff, transports, continuations, provenance, and local naming data;*
2. *classifier stability, stating that replacement of the displayed rows through* H *and replay through* C *preserves the same accepted control-sequence packet;*
3. *control-window coverage, stating that the control row* M *covers exactly the finite windows* W *at the dyadic tolerances recorded by* D*;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, stating that the regular rational readback* R *is the only source row inspected by the control/modulus surface;*
5. *Real seal non-escape, stating that the handoff* E *reaches* $\mathsf{Real}^{\uparrow}$ *only through* R, W, D, M, H, C, P, N *and not through quotient stream equality, a chosen limit, trichotomy, or ambient completeness.*

Proof. Carrier formation is the tuple of Definition 1426.1. It exposes exactly one regular rational readback row, one finite-window schedule, one dyadic tolerance row, one control/modulus row, one Real seal handoff, and the local transport, continuation, package, and naming rows.

Classifier stability is componentwise over H and C. The replacement rows may transport the displayed coordinates and replay their consumers, but they do not introduce a second source row, a second tolerance ledger, or an external equality principle.

Control-window coverage is the binding role of M. The control row is accepted only as a displayed modulus for the windows W and tolerances D, with the regular readback R providing the rational values inspected inside those windows.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the Real seal non-escape are then immediate from the carrier inventory. Since E appears after R, W, D, M and the provenance rows contain no quotient, chosen limit, trichotomy, or completeness coordinate, every seal-facing read is forced through the finite control route. □

Theorem 1426.3 (CauchyControlSequence Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyControlSequence}^{\uparrow}$ *packet factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the control row* M*, the* hsame *and* Cont *provenance rows* H *and* C*, the package row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}$ *row* N *before reaching the seal handoff* E*.*

Proof. Project the accepted carrier. The seal handoff coordinate E is stored only after the regular readback, finite-window schedule, dyadic tolerance row, and displayed control row.

Read the sibling surfaces in the displayed order. The StreamName row supplies the finite windows, the DyadicRatCore row supplies the tolerance arithmetic, the RegSeqRat row supplies the rational readback, and the RealCauchyModulus row binds these data as a control surface.

The rows H, C, P, N then record the only allowed transports, replays, package provenance, and local naming evidence. Therefore a Real-facing consumer cannot bypass the displayed control route before it reaches E. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyControlSequence}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a displayed control/modulus row, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1426.1, Definition 1426.2, and Definition 1426.3 bind the explicit Cauchy control data used before the Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyControlSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host completeness, quotient stream equality, chosen limits, trichotomy, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, control-window coverage, RegSeqRat handoff, and Real seal non-escape for the displayed control packet.

Chapter 1427

A Concrete Naming Certificate for $\mathsf{LocatedSupremum}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{LocatedSupremum}^{\uparrow}$ packet records the least-upper-bound-shaped data that is available before any ambient completeness principle for $\mathsf{Real}^{\uparrow}$. It is a located upper-bound certificate over the located-cut and located-real surfaces of chapter 404 and **??**, the comparison discipline of chapter 1381, the regular-rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite interface for Real and Completion consumers that need upper-bound and lower-approximation reads; it is not a completeness theorem for arbitrary real subsets.

Definition 1427.1 (LocatedSupremum carrier)**.** *A* $\mathsf{LocatedSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is a located-cut or located-real family row, read through the finite locatedness and comparison surfaces of the sibling* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{BishopRealComparison}^{\uparrow}$ *chapters;*
2. *U is the upper-bound ledger, displaying the inspected bound claims and the finite comparison rows that certify each requested member of L below the proposed bound surface;*
3. *A is the lower-approximation and cofinality ledger, displaying the finite rational or dyadic witnesses that approach the bound surface from the family side at the precisions requested by the consumer;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *precision-window row that fixes which finite observations of the family, bound ledger, and approximation ledger are in scope;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the windowed observations as regular rational readback data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through L, U, A, W, R;*
7. *H records the componentwise* hsame *transports for the family row, upper-bound ledger, approximation ledger, precision windows, regular handoff, real seal, continuation routes, provenance, and local naming row;*
8. *C records the displayed* Cont *routes that replay upper-bound reads, lower-approximation reads, classifier transport, and real-seal readback;*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$ *row.*

The accepted carrier stores no quotient cut, no selected supremum element, no trichotomy oracle, no host real equality test, and no ambient order-completeness coordinate.

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier semantic name certificate`

Theorem 1427.2 (LocatedSupremum NameCert obligations)**.** *For an accepted packet $S = (L, U, A, W, R, E, H, C,$ the local* $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$ *obligations are exactly the following finite rows:*

1. *carrier admission: the accepted object is the displayed packet with a located family row, upper-bound ledger, lower-approximation ledger, precision window, regular-rational handoff, real seal, transports, continuations, provenance, and local naming data;*
2. *located upper-bound stability: replacing L, U, W, R, E along the displayed* hsame *transports preserves the same upper-bound ledger reads through the* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison surface;*
3. *lower-approximation cofinality: every requested precision in W receives a displayed approximation row in A whose readback reaches the same bound surface through R and E;*
4. *classifier transport: the componentwise transports in H are sufficient to move the located family, bound ledger, approximation ledger, precision window, regular handoff, and real seal through the same classifier routes;*
5. *ledger exhaustion: every admissible upper-bound or lower-approximation consumer must read one of the rows $L, U, A, W, R, E, H, C, P, N$ before obtaining any real-facing observation;*
6. *handoff stability: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and* $\mathsf{Real}^{\uparrow}$ *seal E remain attached to the same displayed precision window and ledgers under the continuation routes in C.*

No obligation may call on a quotient cut, selected supremum, trichotomy, host real equality, countable choice, or ambient order completeness.

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumTasteGate single carrier alignment`

Proof. Project the accepted carrier row by row. Carrier admission is immediate from the displayed BHist packet, and located upper-bound stability reads only the family row L, the upper-bound ledger U, the precision window W, the regular handoff R, the seal E, and the componentwise transports in H.

For lower approximation, inspect the requested precision row in W. The cofinality ledger A supplies the finite approximation witness at that precision, and the same witness is repacked by R before reaching the $\mathsf{Real}^{\uparrow}$ seal E.

Classifier transport and ledger exhaustion are structural. The continuation routes in C can replay only the displayed upper-bound, approximation, handoff, and seal reads, while P records exactly the sibling BEDC surfaces named in the carrier. Since the carrier contains no quotient cut, selected supremum, trichotomy row, host equality test, choice functional, or completeness coordinate, those principles cannot appear in the local NameCert obligations. □

Theorem 1427.3 (LocatedSupremum real-seal boundary)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *read exported by S factors through the located upper-bound ledger U, the lower-approximation row A, the* $\mathsf{StreamName}^{\uparrow}$ *precision window W, the*

$\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal* E*, together with the displayed transports, continuations, provenance, and local naming rows. The export contains no quotient cut, selected supremum, trichotomy, host real equality, or ambient order-completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremum real seal boundary`
↪ variant `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier real seal boundary`

Proof. Begin with a real-facing consumer of the packet. The only real-name coordinate present in the carrier is the seal row E, and the definition makes E reachable only after the located family, upper-bound ledger, approximation ledger, precision window, and regular handoff have been exposed.

The upper side of the read is supplied by U: it records the finite comparison rows certifying the requested family observations below the bound surface. The lower side is supplied by A: it records the finite approximants demanded by the precision window W.

The rows R and E then give the regular-rational and real-name handoff for exactly those displayed observations. The transport and continuation rows can replay this route but cannot add a quotient cut, chosen supremum, trichotomy oracle, host equality test, or order-completeness principle, because none of those rows occurs in the accepted packet. □

Theorem 1427.4 (LocatedSupremum upper-bound ledger stability)**.** *Let* $S = (L, U, A, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every upper-bound read exported by* S *is stable under the componentwise transports recorded in* H*: transporting the located family row, upper-bound ledger, precision window, regular handoff, Real seal, continuation route, provenance, and local naming row preserves the same finite upper-bound ledger reads through* U *and no additional order-completeness coordinate is exposed.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier upper bound ledger stability`

Proof. Project the carrier of Definition 1427.1. The upper-bound information is stored only in U, and the transport row H lists the componentwise transports for $L, U, A, W, R, E, C, P, N$.

Apply Definition 1427.2. Its located upper-bound stability and classifier transport rows state that replacing the displayed carrier coordinates along H preserves the same upper-bound ledger reads through the comparison surface.

The real-seal boundary Definition 1427.3 keeps those reads downstream of U, W, R, E and excludes quotient cuts, selected suprema, trichotomy, host equality, and ambient completeness. Thus the transported packet exports the same finite upper-bound ledger surface. □

Theorem 1427.5 (LocatedSupremum lower-approximation cofinality)**.** *Let* $S = (L, U, A, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. For every precision request displayed in the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the lower-approximation ledger* A *contains a displayed approximation row at that precision, and the approximation is read through the same regular-rational handoff* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier lower approximation cofinality`

Proof. Inspect a precision request in W. The carrier makes W the only precision-window coordinate and makes A the only lower-approximation ledger coordinate.

Apply the lower-approximation cofinality row of Definition 1427.2. It supplies the displayed approximation row in A for the requested precision and binds that row to the same bound surface.

Finally use Definition 1427.3. The theorem forces every Real-facing read to pass from A through W, R, E together with the structural rows, so the approximation cannot bypass the regular-rational handoff or introduce a selected supremum. □

Theorem 1427.6 (LocatedSupremum RegSeqRat Real handoff stability)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every exported real-facing approximation from S factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R before it reaches the* $\mathsf{Real}^{\uparrow}$ *seal E, and this factorization is stable under the continuation routes in C and the componentwise transports in H.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier regseqrat real handoff stability`

Proof. Start with a real-facing approximation read. By Definition 1427.1, the terminal real coordinate is E, and it appears only after the located family, upper-bound ledger, approximation ledger, precision window, and regular handoff rows have been displayed.

Apply Definition 1427.3. It identifies the exact route into E as the route through U, A, W, R together with H, C, P, N.

The continuation row C replays only that route, and the componentwise transports in H preserve each displayed coordinate. Therefore every exported approximation factors through R before E, with no detached real object, quotient cut, selected supremum, or ambient order-completeness principle. □

Theorem 1427.7 (LocatedSupremum root upper-lower window)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. The upper-bound reads exported through U and the lower-approximation reads exported through A are exposed through the same finite precision window W. Thus a root consumer that asks for both sides of the located-supremum surface receives one shared windowed route through $L, U, A, W, R, E, H, C, P, N$, rather than separate upper and lower access paths.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier root upper lower window`

Proof. Project the accepted carrier of Definition 1427.1. The carrier has one $\mathsf{StreamName}^{\uparrow}$ precision-window row W, and both the upper-bound ledger U and the lower-approximation ledger A are displayed before the regular handoff R and the $\mathsf{Real}^{\uparrow}$ seal E.

Apply the located upper-bound stability and lower-approximation cofinality rows of Definition 1427.2. They read their respective ledgers through the same W, R, E, H, C, P, N support rows.

Finally use Definition 1427.3. Every real-facing read is forced through U, A, W, R, E together with the structural rows, so the root upper and lower observations share the same finite precision window. □

Theorem 1427.8 (LocatedSupremum StreamName cut scope)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every located-cut or located-real read exported by S is scoped through the* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance surface carried by the windowed approximation ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, the componentwise* hsame *transports H, the displayed* Cont *replay C, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier streamname cut scope`

Proof. Begin with a located-cut or located-real consumer of S. The source row L is admitted only as part of the displayed packet, and the carrier stores no independent cut or real object apart from the route through W, R, E.

Use Definition 1427.7. It places the upper-bound and lower-approximation ledgers under one finite precision window, so the cut-facing read and the real-facing read share the same inspected finite observations.

Now apply Definition 1427.6. The regular-rational handoff and terminal seal remain stable under H and C, while P and N name the same route. The dyadic tolerance information is read only as the tolerance surface carried by the windowed approximation ledger, so no detached cut, selected supremum, or ambient completeness coordinate enters the scope. □

Theorem 1427.9 (LocatedSupremum Real seal nonescape)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. No* $\mathsf{Real}^{\uparrow}$*-facing consumer can bypass $L, U, A, W, R, E, H, C, P, N$ to obtain a selected supremum, trichotomy row, quotient cut, host real equality test, or ambient order-completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier real seal nonescape`

Proof. Start from the Real-facing boundary theorem Definition 1427.3. It identifies every exported real read with the finite route through the located family, upper-bound ledger, lower-approximation ledger, precision window, regular handoff, terminal seal, transports, continuations, provenance, and local name.

The root upper-lower window theorem Definition 1427.7 and the StreamName cut-scope theorem Definition 1427.8 keep the two order-facing sides on that same route before the seal is exposed.

The excluded objects are absent from the carrier of Definition 1427.1. Since H and C can only transport and replay displayed rows, a consumer has no coordinate from which to read selected suprema, trichotomy, quotient cuts, host equality, or ambient order completeness. □

Theorem 1427.10 (LocatedSupremum source envelope)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every source read exported by S factors first through the displayed* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{BishopRealComparison}^{\uparrow}$ *surfaces named by the source row L before the upper-bound ledger U or the lower-approximation ledger A is inspected. The envelope contains no selected supremum, trichotomy row, quotient cut, host equality test, countable-choice functional, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the source row in Definition 1427.1. The carrier admits L only as a located-cut or located-real family row, and the definition binds that row to the comparison surfaces of $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{BishopRealComparison}^{\uparrow}$.

The upper and lower ledgers are downstream of that source envelope. The upper-bound ledger U records comparison reads for requested family observations, while the lower-approximation ledger A records finite approximants attached to the same bound surface.

Apply the real-seal nonescape theorem Definition 1427.9. Since all consumer-facing reads remain inside $L, U, A, W, R, E, H, C, P, N$, the source envelope cannot be replaced by selected suprema, trichotomy, quotient cuts, host equality, countable choice, or real completeness. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier source envelope`

Theorem 1427.11 (LocatedSupremum dyadic bracket)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Each precision request in the* $\mathsf{StreamName}^{\uparrow}$ *window W exposes a shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket carried by the located source and lower-approximation*

rows. The same bracket is read by the upper-bound ledger U as a comparison tolerance and by the lower-approximation ledger A as an approximation tolerance before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff is exported.*

Proof. Inspect a precision request in W. The root upper-lower window theorem Definition 1427.7 places the upper-bound reads and lower-approximation reads under the same finite precision window.

The source envelope of Definition 1427.10 supplies the located-cut and located-real bracket surface through L. Reading the request therefore exposes one dyadic tolerance surface for both ledgers, not separate upper and lower ambient estimates.

The StreamName cut-scope theorem Definition 1427.8 keeps this bracket on the route through W, R, E, H, C, P, N. Thus the bracket is shared by U and A before the regular-rational and real-name handoffs, with no selected supremum, quotient cut, trichotomy, host equality, countable choice, or completeness principle. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier dyadic bracket`

Theorem 1427.12 (LocatedSupremum RegSeqRat handoff route)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. The shared dyadic bracket of Definition 1427.11 repacks through the* $\mathsf{StreamName}^{\uparrow}$ *window W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is available to any consumer. The handoff exports only the displayed source envelope, upper-bound ledger, lower-approximation ledger, window, regular readback, seal, transports, continuations, provenance, and local naming row.*

Proof. Begin with the shared bracket supplied by Definition 1427.11. The bracket has already been read through the source envelope and through the common precision window, so it is part of the displayed finite route rather than an ambient order-completeness witness.

Apply the RegSeqRat Real handoff stability theorem Definition 1427.6. It forces every real-facing approximation to pass through R before E and keeps the route stable under H and C.

Finally use Definition 1427.9. The terminal seal can expose only the displayed rows $L, U, A, W, R, E, H, C, P, N$, so the handoff cannot export selected suprema, trichotomy, quotient cuts, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier regseqrat handoff route`

Theorem 1427.13 (LocatedSupremum cofinal window totality)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every completion-facing precision request accepted by W exports one common finite window in which the upper-bound ledger U and the lower-approximation ledger A are both readable: the upper side reads the source envelope through the comparison rows of L and U, the lower side reads a cofinal approximation row through A, and both sides share the dyadic bracket,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, transports H, replays C, provenance P, and local name N.*

Proof. Start with a completion-facing precision request in W. By Definition 1427.7, upper-bound reads and lower-approximation reads are already exposed through one finite precision window, rather than by two unrelated routes.

Use the source envelope of Definition 1427.10. The upper side must read the located-cut or located-real family row through L before inspecting U, so the upper-bound comparison remains inside the displayed family and comparison surface.

For the lower side, apply Definition 1427.5. The ledger A supplies a displayed approximation row at the same requested precision, and Definition 1427.11 identifies the shared dyadic bracket read by both ledgers.

Finally use Definition 1427.12. The common bracket is repacked through R before E, and the route is replayed only by H, C, P, N. Thus every accepted completion-facing request receives a single finite window carrying both the upper-bound and cofinal lower-approximation reads. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier cofinal window totality`

Theorem 1427.14 (LocatedSupremum monotone sequence handoff)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Any monotone bounded real-sequence consumer that reads S must pass through the same upper-bound ledger U, lower-approximation ledger A,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance surface carried by the windowed approximation ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$ *row N. The handoff exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, selected supremum, trichotomy row, quotient cut, host sequence equality, or countable-choice selector.*

Proof. Project the carrier of Definition 1427.1. A monotone bounded real-sequence consumer can inspect the source family only through L and can read its upper and lower order information only through the displayed ledgers U and A.

Apply Definition 1427.3. The theorem places every real-facing export downstream of the upper-bound ledger, lower-approximation ledger, precision window, regular-rational handoff, and terminal Real seal.

Use Definition 1427.5 and Definition 1427.13. The lower ledger supplies the requested cofinal approximation inside the same window that carries the upper-bound read, and the shared dyadic tolerance surface is the one carried by the windowed approximation ledger.

Finally replay the route through H, C, P, N. Componentwise transport and continuation replay preserve the displayed rows but cannot add an ambient completeness coordinate, selected supremum, trichotomy row, quotient cut, host sequence equality, or countable-choice selector. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier monotone sequence handoff`

Theorem 1427.15 (LocatedSupremum monotone convergence consumer envelope)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Any monotone-convergence or completion-facing consumer that reads the upper/lower window of S as stable* $\mathsf{Real}^{\uparrow}$ *data must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and terminal* $\mathsf{Real}^{\uparrow}$ *seal through the Cauchy-equivalence classifier boundary of chapter 1820. The consumer receives a classifier-mediated regular-rational route, not a quotient real equality, host sequence equality, selected supremum, countable-choice selector, or ambient completeness principle.*

Proof. Start with the monotone sequence handoff. By Definition 1427.14, a monotone bounded real-sequence consumer must pass through the displayed upper-bound ledger, lower-approximation ledger, shared window, regular-rational handoff, dyadic tolerance surface, terminal seal, and structural rows.

Use the StreamName cut scope of Definition 1427.8. The cut-facing read is already confined to the finite window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, the dyadic tolerance surface, and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Now insert the Cauchy-equivalence classifier boundary. The regular-rational readback can be treated as a stable Real-facing route only after the comparison surface of chapter 1820 has classified equivalent Cauchy windows on the same displayed path.

Finally apply Definition 1427.9. Since the carrier has no selected supremum, quotient cut, host sequence equality, countable-choice selector, or completeness coordinate, the monotone-convergence consumer receives only the classifier-mediated route through $L, U, A, W, R, E, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier monotone convergence consumer envelope`

Theorem 1427.16 (LocatedSupremum window transport refusal)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Transporting the common finite window of Definition 1427.13 through H and replaying it through C cannot introduce a selected supremum, a trichotomy row, a quotient cut, a host real equality test, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every transported completion-facing read remains a read of $L, U, A, W, R, E, H, C, P, N$.*

Proof. Begin with the finite window supplied by Definition 1427.13. It contains the source envelope, upper-bound ledger, lower-approximation ledger, dyadic bracket, regular-rational readback, and terminal real seal as displayed rows of the same packet.

Apply Definition 1427.4 and Definition 1427.6. Componentwise transport through H preserves the upper-bound ledger reads and keeps the regular-rational handoff attached to R before E.

The nonescape clause follows from Definition 1427.9. Since C can only replay displayed routes and the carrier contains none of the excluded coordinates, window transport cannot manufacture selected suprema, trichotomy, quotient cuts, host equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier window transport`

Theorem 1427.17 (LocatedSupremum standard bridge route)**.** *Let $S = (L, U, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet. Every standard-interface read of S factors through the public finite rows $L, U, A, W, R, E, H, C, P, N$: the abstract bridge can inspect only the located family row, located upper-bound ledger, lower-approximation cofinality,* $\mathsf{StreamName}^{\uparrow}$ *precision window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$ *row. It receives no selected supremum, quotient cut, trichotomy row, host real equality test, countable-choice selector, or ambient real-completeness coordinate.*

Proof. Start from the public finite interface recorded by Definition 1427.2. It exposes only the rows $L, U, A, W, R, E, H, C, P, N$ and makes the located upper-bound ledger and lower-approximation cofinality packet data.

Use Definition 1427.12 and Definition 1427.3. Any real-facing read must pass through the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E before it can be exported.

Finally apply Definition 1427.16. The transport and replay rows H, C preserve the same route, while P, N only source and name it. Thus the standard-interface bridge reads exactly the displayed public rows and none of the excluded host or quotient principles. □

Lean4 ✓ `BEDC.Derived.LocatedSupremumUp.LocatedSupremumCarrier standard bridge route`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSupremum}^{\uparrow}}$.

Constructive story: A LocatedSupremum$^{\uparrow}$ packet is generated from a located family row, an upper-bound ledger, a lower-approximation ledger, a StreamName precision window, a RegSeqRat handoff, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedSupremum}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1427.1, Definition 1427.2, Definition 1427.4, Definition 1427.5, Definition 1427.6, Definition 1427.15, Definition 1427.16, and Definition 1427.17 expose the standard bridge route: located upper-bound ledger, lower-approximation cofinality, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming over the L10 Real and Completion interface.

Formal verification: FormalStatus(NameCert$_{\text{LocatedSupremum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full Dedekind completeness, arbitrary least-upper-bound completeness, quotient cuts, selected suprema, trichotomy, host real equality, countable choice, Real completeness, or Lean bridge checking.

Upgrade path: The theory axis has no higher closure grade for this packet; the formal verification axis remains independent.

Chapter 1428

A Concrete Naming Certificate for $\mathsf{DyadicIntervalCover}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicIntervalCover}^{\uparrow}$ packet records a finite dyadic interval-cover surface for the open Real and Completion route. It sits between the dyadic arithmetic of chapter 291, the finite observation windows of chapter 126, the regular-rational readback of chapter 301, the visible Real seal of chapter 38, and the finite interval-net surface of chapter 1410. The packet names cover data that can be inspected through finitely many rows; it does not assert an open-cover compactness theorem or any host completion principle.

Definition 1428.1 (DyadicIntervalCover carrier)**.** *A* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *L and U are finite rows of lower and upper dyadic endpoints, each endpoint read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic surface.*
2. *M is the mesh-index row that records the dyadic scale at which endpoint comparisons and interval-membership tests are performed.*
3. *R is the radius ledger assigning a displayed dyadic tolerance to every interval row and every inspected regular-name window.*
4. *V is the finite cover-membership row: each listed mesh cell carries the index of a displayed dyadic interval whose lower and upper endpoint rows contain that cell at the tolerance named by R.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by the consumer real names that are tested against the cover.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that translates the windows W into regular rational approximants at the mesh scale M.*
7. *A is the visible* $\mathsf{Real}^{\uparrow}$ *seal row reached through the same window and readback data.*
8. *H stores the componentwise* hsame *transports for endpoints, mesh indices, radius rows, cover-membership rows, windows, readbacks, Real seals, continuation, provenance, and local naming rows.*

9. *C stores the displayed* Cont *replay from dyadic endpoints to mesh cells, from mesh cells to cover-membership rows, from windows to regular readback, and from regular readback to the visible Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopIntervalCompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ *row.*

The accepted carrier stores only the finite cover surface just displayed. It carries no open-cover compactness principle, no Heine–Borel theorem, no sequential compactness selector, no selected limit, no quotient real equality, no host real equality test, no countable-choice functional, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverTasteGate single carrier alignment`

Theorem 1428.2 (DyadicIntervalCover NameCert obligations)**.** *For an accepted packet $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ *obligations are exhausted by the finite endpoint, mesh, radius, membership, window, readback, seal, transport, continuation, and provenance rows of Definition 1428.1. In particular, every obligation can be checked by finite projection from the displayed* BHist *packet and by replay of H, C, and P; no obligation may call on open-cover compactness, Heine–Borel compactness, sequential compactness, selected limits, quotient real equality, host real equality, countable choice, or Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverNamecertObligations`

Proof. Project the carrier row by row. The endpoint and mesh obligations read only L, U, M, R, V and the dyadic arithmetic certified by the $\mathsf{DyadicRatCore}^{\uparrow}$ sibling chapter. The real-name obligations read only the finite windows W, the regular-rational readback Q, and the visible seal A, hence they factor through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surfaces.

The remaining obligations are structural. The hsame row H transports equal displayed coordinates, the Cont row C replays the permitted reads, and the provenance row P records exactly the finite BEDC surfaces used by the packet, including the comparison with the $\mathsf{BishopIntervalCompact}^{\uparrow}$ finite-net interface. Since the carrier has no row for an open-cover selector, a host limit, a quotient equality, a choice functional, or a completeness principle, none of those principles can appear in a local NameCert obligation. □

Theorem 1428.3 (DyadicIntervalCover endpoint row admission)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Endpoint admission for the lower and upper rows is exactly the finite projection from Definition 1428.1 together with the local NameCert obligations of Definition 1428.2: the rows L, U are read as* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints, M, R provide the displayed mesh and tolerance data, V, W, Q, A remain the finite membership, StreamName, RegSeqRat, and Real-seal rows, and H, C, P, N transport, replay, package, and name that same endpoint surface.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverEndpointRowAdmission`

Proof. Project the carrier row $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$. The definition places L and U in the dyadic endpoint slots and places M and R beside them as the only mesh and tolerance data available to endpoint comparisons.

Use the NameCert-obligation theorem. It says that endpoint, mesh, radius, membership, window, readback, seal, transport, continuation, and provenance checks are exhausted by the displayed

rows. Thus admitting an endpoint row can only mean checking L, U through $\mathsf{DyadicRatCore}^{\uparrow}$ and then preserving the result through H, C, P, N.

The remaining rows cannot create an additional endpoint source. The membership row V, window row W, readback row Q, and seal row A are downstream consumers of the same finite endpoint data, so endpoint admission stays inside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert data. □

Theorem 1428.4 (DyadicIntervalCover window membership transport)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P,$ be an accepted $\mathsf{DyadicIntervalCover}^{\uparrow}$ packet. Finite cover-membership transport along StreamName windows is completely determined by the carrier and obligation rows of Definition 1428.1 and Definition 1428.2: the transported membership row V reads endpoint rows L, U, mesh row M, radius ledger R, finite window row W, regular readback row Q, and Real seal row A, and its only structural support is H, C, P, N.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverWindowMembershipTransport`

Proof. Begin with the window row W and membership row V. The carrier definition makes W the finite StreamName surface and V the finite list of cover-membership witnesses attached to the dyadic endpoint rows.

Apply the NameCert-obligation theorem to the same carrier. The theorem identifies H as the componentwise transport row and C as the replay row for endpoint, mesh, radius, membership, window, readback, and seal data. Therefore a membership transport can only move along the displayed route from L, U, M, R, W, Q, A to V.

The rows P and N package and name that route locally. Since no carrier coordinate supplies host interval equality, selected subcovers, quotient real equality, countable choice, or Real completeness, the transported finite membership row stays inside $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert data. □

Theorem 1428.5 (DyadicIntervalCover Real seal boundary)**.** *For an accepted $\mathsf{DyadicIntervalCover}^{\uparrow}$ carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$, every Real seal read through A is bounded by the carrier rows and NameCert obligations of Definition 1428.1 and Definition 1428.2. The seal may inspect L, U, M, R, V, W, Q and may be transported or replayed by H, C, P, N, but it cannot leave the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert surface.*

Proof. Project the seal row. In the displayed carrier, A is the only $\mathsf{Real}^{\uparrow}$ seal coordinate and it is listed after the endpoint, mesh, radius, membership, window, and regular readback rows.

Use the NameCert-obligation theorem. Its finite projection discipline forces every seal obligation to be checked from the same rows $L, U, M, R, V, W, Q, A, H, C, P, N$ rather than from an external completion or host real equality.

The structural rows do not enlarge the seal. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve only the route already present in the carrier, so the Real seal boundary cannot export open-cover compactness, selected limits, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealSealBoundary`

Theorem 1428.6 (Regular-sequence finite-window exhaustion)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted $\mathsf{DyadicIntervalCover}^{\uparrow}$ packet. Any consumer that asks whether a regular real name represented by the displayed $\mathsf{StreamName}^{\uparrow}$ windows W and $\mathsf{RegSeqRat}^{\uparrow}$ readback Q meets the cover*

named by L, U, M, R, V is exhausted by a finite window computation: it reads the mesh index M, the radius ledger R, the regular approximants in Q, the finite membership rows V, and the visible $\mathsf{Real}^{\uparrow}$ *seal A. No later tail of the regular sequence and no ambient equality of real names contributes to the cover decision.*

Proof. Begin with the window and readback coordinates. The $\mathsf{StreamName}^{\uparrow}$ row supplies exactly the finite windows available to the consumer, and the $\mathsf{RegSeqRat}^{\uparrow}$ row turns those windows into rational approximants at the mesh scale named by M and the radii named by R.

Next inspect the cover surface. The finite row V contains the whole list of membership witnesses for the dyadic endpoint rows L and U, so a consumer can only compare the displayed approximants with this finite list. The continuation row C has no replay path beyond those rows, and the Real seal A records only the visible endpoint reached through the same finite readback.

Thus the decision surface is exhausted by M, R, Q, V, A together with the displayed transports in H. A later regular-sequence tail, quotient equality of real names, host equality, countable choice, selected limits, and Real completeness are absent from the packet, so they cannot affect the finite-window cover read. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegularSequenceWindowExhaustion`

Theorem 1428.7 (DyadicIntervalCover RegSeqRat-StreamName coverage route)**.** *Every* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *consumer that reads a* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation through* $\mathsf{StreamName}^{\uparrow}$ *windows must expose the rows W, Q, M, R, V, A in that order before the cover-membership row is consumed. The route is stable under the componentwise* hsame *row H and the displayed* Cont *replay row C, and it exports no quotient real equality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1428.1. The StreamName window row W is the only finite source window, the RegSeqRat readback row Q is attached to that window, and the mesh, radius, membership, and seal rows are M, R, V, A.

Apply Definition 1428.6. It exhausts every regular-name cover read through the displayed finite-window computation, so the membership row V is not consumed until W, Q, M, R have been opened and the visible Real seal boundary A remains downstream.

Finally use the seal boundary of Definition 1428.5. Componentwise transport and continuation replay preserve the same route, while quotient real equality, selected limits, countable choice, and Real completeness remain outside the packet. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegSeqRatStreamNameCoverageRoute`

Theorem 1428.8 (DyadicIntervalCover finite-window carrier)**.** *Every accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet exposes only finite interval-window rows over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *For a carrier*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N),$$

the endpoint rows L, U, mesh row M, radius ledger R, membership row V, window row W, readback row Q, and seal row A form the whole displayed cover surface before the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ *row is exported.*

Proof. Project the carrier of Definition 1428.1. The endpoint, mesh, radius, and membership rows are finite dyadic interval-window data, and the only Real-facing rows are the displayed StreamName windows, RegSeqRat readback, and terminal Real seal.

Apply Definition 1428.6. It forces every regular-name cover read through M, R, Q, V, A together with the finite window row W before any cover decision is consumed. The L10 sibling dependencies

are exactly the dyadic arithmetic of $\mathsf{DyadicRatCore}^{\uparrow}$, the finite observation surface of $\mathsf{StreamName}^{\uparrow}$, the regular-rational readback of $\mathsf{RegSeqRat}^{\uparrow}$, and the visible seal of $\mathsf{Real}^{\uparrow}$.

Since the carrier contains no row beyond $L, U, M, R, V, W, Q, A, H, C, P, N$, the finite-window surface is exhausted by those rows. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverFiniteWindowCarrier`

Theorem 1428.9 (DyadicIntervalCover Bishop interval compact handoff)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Any consumer that hands its mesh-radius rows to* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *must expose the full finite packet $L, U, M, R, V, W, Q, A, H, C, P, N$: the dyadic endpoint rows L, U are read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, the mesh and radius rows M, R govern the finite cover-membership row V, the real-name rows W, Q, A supply the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *boundary, and H, C, P, N transport, replay, source, and name the same handoff. The finite-net membership route passes through chapter 1410 only after those dyadic endpoint rows are displayed and before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route exports no open-cover compactness theorem, selected finite subcover, quotient stream, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier from Definition 1428.1. The rows L, U, M, R, V provide the only dyadic endpoint, mesh, radius, and finite membership data available to a compact-interval consumer.

Use Definition 1428.8. It identifies L, U, M, R, V, W, Q, A as the whole finite interval-window surface before the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ row is exported.

Next apply Definition 1428.6 and Definition 1428.5. The former keeps every regular-name cover read inside W, Q, M, R, V, A, while the latter places the visible $\mathsf{Real}^{\uparrow}$ seal after the finite window and endpoint rows.

The $\mathsf{BishopIntervalCompact}^{\uparrow}$ sibling can therefore consume only the displayed finite-net route. Since I has no row for arbitrary open covers, selected subcovers, quotient streams, host equality, countable choice, or ambient Real completeness, the handoff remains a finite mesh-radius obligation rather than a compactness principle. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBishopIntervalCompactHandoff`

Theorem 1428.10 (DyadicIntervalCover compact-uniform window route)**.** *Any compact-uniform-continuity consumer using an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet must read the dyadic endpoint rows, mesh row, radius ledger, finite membership row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff, and* $\mathsf{UniformModulus}^{\uparrow}$*-facing lower-bound route in that order. The route exports no open-cover compactness, selected subcover, quotient real equality, host minimum, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier from Definition 1428.1. The endpoint admission and finite-window carrier theorems expose L, U, M, R, V, W, Q, A as the displayed dyadic endpoint, mesh, radius, membership, window, readback, and seal rows.

Apply Definition 1428.6 and Definition 1428.5. They place every regular-sequence cover read behind the finite windows and put the terminal $\mathsf{Real}^{\uparrow}$ seal after the endpoint and membership rows.

Finally use Definition 1428.9. The compact interval handoff receives only the displayed mesh-radius route, so any $\mathsf{UniformModulus}^{\uparrow}$ lower-bound consumer reads the same finite route and cannot obtain open-cover compactness, a selected subcover, quotient real equality, a host minimum, countable choice, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverCompactUniformWindowRoute`

Theorem 1428.11 (DyadicIntervalCover refinement stability)**.** *Let*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N)$$

and

$$I' = (L', U', M', R', V', W', Q', A', H', C', P', N')$$

be accepted $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packets. If a displayed refinement carries componentwise* hsame *data from the endpoint, mesh, radius, membership, window, readback, and seal rows of* I *to the corresponding rows of* I'*, and the replay rows display the same* Cont *route through those transported coordinates, then the refined cover preserves the same finite interval-window route over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project both carriers. The rows available to the refinement are precisely the finite endpoint rows, mesh rows, radius ledgers, membership rows, finite windows, regular readbacks, Real seals, and structural rows listed in Definition 1428.1.

Apply Definition 1428.6 to I and I'. The cover read in each packet is exhausted by the displayed mesh, radius, readback, membership, and seal rows. Componentwise hsame in H, H' identifies those rows as packet data, and the displayed Cont replay in C, C' preserves the route through the same L10 siblings.

Thus refinement changes only the displayed finite interval rows and their transported membership witnesses. It does not create an open-cover compactness row, a selected subcover, a host equality test, a quotient stream, a choice functional, or a Real-completeness witness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRefinementStability`

Theorem 1428.12 (DyadicIntervalCover route-cover stability)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *and* $I' = (L', U', M', R', V', W', Q', A', H', C', P', N')$ *be accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packets. If a displayed refinement preserves the endpoint rows* L, U*, the mesh and radius rows* M, R*, the finite cover-membership rows* V*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal rows* A*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, then* I *and* I' *expose the same finite cover route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Project both carriers. The only cover-facing coordinates are the endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and local naming rows displayed in Definition 1428.1.

Apply Definition 1428.6. It exhausts each regular-name cover read by the finite rows M, R, Q, V, A after the $\mathsf{StreamName}^{\uparrow}$ window has been opened, so no later regular-sequence tail or ambient real-name equality can affect the route.

Now apply Definition 1428.11. The refinement preserves the endpoint, mesh, radius, membership, window, readback, and seal rows through componentwise hsame data and the displayed Cont replay.

The provenance and local naming rows P, N and P', N' keep the transported route inside the same packet boundary. Therefore the refinement preserves the finite cover route and cannot introduce open-cover compactness, a selected subcover, quotient equality, countable choice, or Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRouteCoverStability`

Theorem 1428.13 (DyadicIntervalCover ledger exactness)**.** *The* DyadicIntervalCover$^{\uparrow}$ *cover ledger is exact for the displayed finite windows. In an accepted carrier*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N),$$

the radius ledger R and membership row V account for exactly the dyadic endpoint comparisons and regular-name windows read through L, U, M, W, Q, A. The ledger exports no open-cover compactness theorem, selected subcover, host real equality, quotient stream, countable choice, or Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverLedgerExactness`

Proof. Project the displayed carrier. The ledger row R supplies the tolerance data for endpoint comparisons and for each inspected regular-name window, while V records the finite membership witnesses at that tolerance.

Use Definition 1428.6. Every cover decision reads only M, R, Q, V, A after the StreamName window W has been opened, so the ledger has no hidden tail coordinate beyond the finite rows.

The sibling dependencies are the dyadic arithmetic surface DyadicRatCore$^{\uparrow}$, the window surface StreamName$^{\uparrow}$, the readback surface RegSeqRat$^{\uparrow}$, and the terminal seal Real$^{\uparrow}$. Since none of those rows supplies open-cover compactness, a selected subcover, host equality, quotient equality, choice, or completeness, the accepted ledger cannot export any such principle. □

Theorem 1428.14 (DyadicIntervalCover Bishop interval compact route)**.** *Any* BishopIntervalCompact$^{\uparrow}$ *consumer of an accepted* DyadicIntervalCover$^{\uparrow}$ *packet*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N)$$

must read the endpoint rows L, U, mesh row M, radius ledger R, membership row V, StreamName$^{\uparrow}$ *windows W,* RegSeqRat$^{\uparrow}$ *readback row Q,* Real$^{\uparrow}$ *seal row A, and structural rows H, C, P, N before the finite interval compact route is available. The route is finite-window interval compactness over the displayed dyadic ledger; it exports no arbitrary open-cover compactness, selected finite subcover, quotient stream equality, host real equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 1428.1. The endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and local naming rows are the only coordinates visible to a compact-interval consumer.

Apply finite-window exhaustion through Definition 1428.6. Each inspected interval window is bounded by M, R, V, W, Q, A, after the endpoint rows L, U have fixed the finite dyadic interval boundary.

Use the Real$^{\uparrow}$ seal row A only as the displayed boundary seal for the regular-name readback. The seal does not add host equality, quotient stream equality, selected limits, countable choice, or ambient Real completeness.

Finally apply Definition 1428.13. Since R, V are exact for the dyadic endpoint comparisons and regular-name windows against $L, U, M, W, Q, A, H, C, P, N$, the BishopIntervalCompact$^{\uparrow}$ route is available exactly through those rows and through no hidden compactness coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBishopIntervalCompactRoute`

Definition 1428.15 (DyadicIntervalCover obligation surface)**.** *For an accepted* DyadicIntervalCover$^{\uparrow}$ *packet*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N),$$

the obligation surface is the finite ledger consisting of the DyadicRatCore$^{\uparrow}$ *endpoint rows* L, U*, mesh row* M*, radius ledger* R*, cover-membership row* V*,* StreamName$^{\uparrow}$ *window row* W*,* RegSeqRat$^{\uparrow}$ *readback row* Q*, visible* Real$^{\uparrow}$ *seal row* A*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* N*. This surface records finite mesh, refinement, membership, and real-window handoff data only; it contains no open-cover compactness coordinate and no selected-subcover coordinate.*

Lean4 (definition) `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverObligationSurface`

Definition 1428.16 (DyadicIntervalCover root obligation surface)**.** *For an accepted* DyadicIntervalCover$^{\uparrow}$ *packet*

$$I = (L, U, M, R, V, W, Q, A, H, C, P, N),$$

the root obligation surface is the L10 tuple consisting of endpoint rows L, U*, mesh and radius rows* M, R*, finite cover-membership row* V*, StreamName window row* W*, RegSeqRat readback row* Q*, Real seal row* A*, and structural rows* H, C, P, N*. It is the part of Definition 1428.15 exposed to root consumers that need interval-cover data without importing open-cover compactness, selected finite subcovers, quotient stream equality, host real equality, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootObligationSurface` semantic name certificate

Theorem 1428.17 (DyadicIntervalCover root finite-cover boundary)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* DyadicIntervalCover$^{\uparrow}$ BHist *packet satisfying Definition 1428.16. Every root finite-cover read of* I *is bounded by the displayed endpoint rows* L, U*, mesh row* M*, radius ledger* R*, membership row* V*, StreamName window* W*, RegSeqRat readback* Q*, Real seal* A*, componentwise* hsame *row* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. The boundary exports no arbitrary open cover, selected finite subcover, quotient stream equality, host real equality, countable-choice schedule, or ambient* Real$^{\uparrow}$ *completeness row.*

Proof. Project the root surface. By Definition 1428.8, the only cover-facing coordinates are $L, U, M, R, V, W, Q, A, H, C, P, N$.

Use the Bishop interval compact handoff of Definition 1428.9. It places any compact-interval consumer behind the finite dyadic endpoint, mesh, radius, membership, window, readback, and seal rows before a local NameCert can be exported.

The rows H, C, P, N transport, replay, source, and name that same finite route. Since the accepted BHist packet has no coordinate for the excluded host or selection data, the root boundary is exactly the displayed finite-cover surface. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootFiniteCoverBoundary`

Theorem 1428.18 (DyadicIntervalCover root window coverage)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* DyadicIntervalCover$^{\uparrow}$ BHist *packet satisfying Definition 1428.16. Every root coverage request that begins from a* StreamName$^{\uparrow}$ *window row must read* W *first, then the* RegSeqRat$^{\uparrow}$ *readback* Q*, then the dyadic mesh and radius rows* M, R *with membership row* V*, and only then the visible* Real$^{\uparrow}$ *seal* A *and structural rows* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootWindowCoverage`

Proof. Start with the window and readback rows. The root surface contains W as the only finite $\mathsf{StreamName}^{\uparrow}$ window coordinate and Q as the only $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to it.

Apply Definition 1428.6. It forces every regular-name cover decision through M, R, Q, V, A after the finite window has been opened.

Finally use Definition 1428.12. Componentwise hsame transport and displayed Cont replay preserve the same route, while P, N only source and name it. Thus root coverage cannot bypass the displayed window-readback-cover order. □

Theorem 1428.19 (DyadicIntervalCover root compact handoff)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ BHist *packet satisfying Definition 1428.16. Any root consumer that hands I to* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *must expose the finite cover rows L, U, M, R, V, the real-window rows W, Q, A, and the structural rows H, C, P, N as one packet before the compact consumer may read the cover. The handoff is finite packet data, not open-cover compactness, selected subcover extraction, quotient equality, choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootCompactHandoff`

Proof. Use Definition 1428.9. It already identifies the $\mathsf{BishopIntervalCompact}^{\uparrow}$ consumer route as the displayed finite mesh-radius and real-window handoff.

Next apply Definition 1428.11. If the root consumer refines the dyadic cells before the compact read, the transported endpoint, mesh, radius, membership, window, readback, and seal rows remain in the same componentwise route.

The local BHist packet is exhausted by $L, U, M, R, V, W, Q, A, H, C, P, N$. Therefore the compact handoff cannot acquire an open-cover theorem or a selected subcover outside the visible packet rows. □

Theorem 1428.20 (DyadicIntervalCover root regular readback)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ BHist *packet satisfying Definition 1428.16. Every root* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the interval-cover packet is accounted for by Q against the finite window row W, dyadic endpoint rows L, U, mesh row M, radius ledger R, membership ledger V, seal row A, and structural rows H, C, P, N. No later regular-sequence tail, quotient real equality, host real equality, countable choice, selected limit, or ambient completeness row can alter the root readback.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootRegularReadback`

Proof. Project the readback row Q from the root surface. It is tied to the finite window row W and the mesh-radius rows M, R before any cover membership decision is read.

Use Definition 1428.13. The ledger rows R, V are exact for endpoint comparisons and regular-name windows against $L, U, M, W, Q, A, H, C, P, N$.

Route-cover stability then keeps the same readback under hsame transport and Cont replay. Since no tail or quotient coordinate exists in the carrier, the regular readback is exhausted by the displayed finite rows. □

Theorem 1428.21 (DyadicIntervalCover root nonescape)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ BHist *packet satisfying Definition 1428.16. Every root read exported by I remains inside the finite route named by $L, U, M, R, V, W, Q, A, H, C, P, N$ and the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ *row. In particular, neither the* $\mathsf{Real}^{\uparrow}$ *seal nor the compact handoff can escape to host real equality, quotient stream equality, selected limits, countable choice, arbitrary open-cover compactness, selected subcovers, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootNonescape`

Proof. Begin with carrier exhaustion from Definition 1428.8. The root surface has only endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and naming rows.

Use Definition 1428.5 for the Real-facing exit and Definition 1428.9 for the compact-facing exit. Both routes keep the downstream read behind the same finite dyadic, stream, regular-rational, Real, transport, replay, provenance, and naming rows.

Finally apply ledger exactness. The rows R, V account for the finite endpoint and regular-window comparisons, so there is no remaining coordinate from which a root consumer could read an excluded host, quotient, choice, selected-subcover, or completeness object. □

Theorem 1428.22 (DyadicIntervalCover root unblock package)**.** *Every accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet satisfying Definition 1428.16 exports exactly the root L10 handoff through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{BishopIntervalCompact}^{\uparrow}$. *The package reads endpoint, mesh, membership, regular-window, seal, transport, replay, provenance, and local naming rows, and it exports no open-cover compactness, selected finite subcover, quotient stream, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the root surface. Endpoint admission is supplied by Definition 1428.3, while mesh and radius reads are bound by Definition 1428.23. Thus every root interval cell is first read through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ rows L, U, M, R and the finite membership row V.

The regular-name handoff is forced through the window and readback rows. Apply Definition 1428.30 to route every $\mathsf{RegSeqRat}^{\uparrow}$ consumer through W, Q before it reaches the dyadic cover ledger.

The Real-facing exit is sealed by Definition 1428.31. Together with Definition 1428.45, this keeps finite tail windows, regular readbacks, cover rows, the $\mathsf{BishopIntervalCompact}^{\uparrow}$ finite-net comparison, and A, H, C, P, N inside one displayed L10 package.

The excluded principles are absent from the carrier and from the cited rows. Therefore the root package exposes the finite interval-cover handoff and nothing beyond the displayed dyadic, stream, regular-rational, real, compact-interval, transport, replay, provenance, and naming surface. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootUnblockPackage`

Theorem 1428.23 (DyadicIntervalCover mesh refinement obligation)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Any mesh refinement obligation for* I *is determined by the rows* $L, U, M, R, V, H, C, P, N$ *of Definition 1428.15: the refined endpoint comparisons still read through* $\mathsf{DyadicRatCore}^{\uparrow}$, *the refined mesh is still named by* M *and* R, *and the transported membership witnesses remain inside* V *with no appeal to open-cover compactness or selected subcovers.*

Proof. Project the obligation surface. The endpoint rows L, U and the mesh-radius rows M, R are the only rows that can refine dyadic cells, and the membership row V is the only row that can record which displayed interval contains a listed cell.

Transport and replay do not add a cover theorem. The rows H, C, P, N preserve the same displayed endpoint, mesh, radius, and membership coordinates over $\mathsf{DyadicRatCore}^{\uparrow}$.

Therefore a refinement obligation is a finite row check. Since the carrier has no row for an arbitrary open cover or a selected finite subcover, refinement cannot supply compactness data beyond the displayed mesh ledger. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMeshRefinementObligation`

Theorem 1428.24 (DyadicIntervalCover subcover exclusion obligation)**.** *For every accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *carrier* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$*, the finite cover-membership row* V *excludes any hidden selected-subcover obligation. A consumer may inspect* L, U, M, R, V *and transport them through* H, C, P, N*, but the packet does not provide a row selecting a finite subcover from an arbitrary open cover and does not turn finite membership data into Heine–Borel compactness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverSubcoverExclusionObligation`

Proof. Read the membership row in the carrier. The row V lists finite membership witnesses for the already displayed dyadic endpoint and mesh rows, so its source is L, U, M, R rather than an arbitrary family of open sets.

Use Definition 1428.13. The ledger is exact for endpoint comparisons and regular-name windows, and it explicitly exports no open-cover compactness theorem or selected subcover.

The structural rows cannot create a selector. They only transport and replay the finite rows already present in the packet, so the subcover principle remains absent from the obligation surface. □

Theorem 1428.25 (DyadicIntervalCover real-window handoff obligation)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P$ *be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every real-window handoff obligation factors through the finite route*

$$W, Q \longmapsto M, R, V \longmapsto A$$

over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The handoff may compare the displayed regular-name windows with the mesh and membership ledger, but it does not supply open-cover compactness, selected subcovers, quotient real equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealWindowHandoffObligation`

Proof. Start with the window row W and readback row Q. These rows expose only finite observations and regular rational approximants, so every Real-facing read begins inside the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ data.

Next pass through the dyadic cover ledger. The mesh row M, radius ledger R, and membership row V compare those approximants against the endpoint rows L, U already certified by $\mathsf{DyadicRatCore}^{\uparrow}$.

Finally read the Real seal row A. The seal records the visible endpoint of the same finite route, and the structural rows H, C, P, N preserve that route without adding compactness, a selector, quotient equality, choice, selected limits, or completeness. □

Theorem 1428.26 (DyadicIntervalCover membership transport)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every cover-membership read through the row* V *is transported only by the componentwise* hsame *row* H *and the displayed* Cont *replay row* C *over the endpoint rows* L, U*, mesh row* M*, radius ledger* R*, readback row* Q*, and membership row* V*. In particular, membership transport exports no host interval equality, selected subcover, quotient real equality, countable choice, or Real completeness row.*

Proof. Project the carrier of Definition 1428.1. The membership row V is displayed together with endpoint rows L, U, mesh row M, radius ledger R, window row W, readback row Q, and seal row A.

Use Definition 1428.2. The local NameCert obligations make H the componentwise transport row and C the replay route for endpoint, mesh, radius, membership, window, readback, and seal data.

Finally apply Definition 1428.11. Refinement preserves the finite interval-window route only through the transported rows, so a transported membership read cannot acquire host interval equality, selected subcover data, quotient equality, countable choice, or Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMembershipTransport`

Theorem 1428.27 (DyadicIntervalCover seal nonescape)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal read through A factors through the finite window row W and regular readback row Q before it reaches the cover rows L, U, M, R, V and the structural rows H, C, P, N. The seal exports no open-cover compactness, host equality, quotient stream, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with Definition 1428.6. It forces any regular-name cover read to inspect the finite windows W, readback row Q, mesh row M, radius ledger R, membership row V, and visible Real seal A.

Use Definition 1428.8 to identify the whole cover surface. The rows L, U, M, R, V, W, Q, A are the displayed interval-window data before the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ row can be exported.

The structural rows H, C, P, N preserve only that displayed route. Since the carrier of Definition 1428.1 contains no coordinate for open-cover compactness, host equality, quotient streams, selected limits, countable choice, or Real completeness, the seal cannot export any such data. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverSealNonescape`

Theorem 1428.28 (DyadicIntervalCover regular-window ledger)**.** *For every accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$, the regular-window ledger is exactly the finite accounting of W, Q, R against the mesh and membership rows M, V. Each regular-name window opened by W is read back through Q, assigned its displayed radius by R, and compared with the finite membership row V at the mesh named by M; no hidden tail ledger or ambient equality source contributes to the accounting.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegularWindowLedger`

Proof. Project the carrier and read the rows W, Q, R, M, V. The window row W supplies the finite observations, the readback row Q turns them into regular rational approximants, the radius ledger R supplies the displayed tolerance, and M, V give the mesh and membership rows that consume those approximants.

Apply Definition 1428.6. It says every cover decision is exhausted by the displayed finite window computation and has no later regular-sequence tail or ambient real-name equality input.

Now use Definition 1428.13. The existing ledger theorem identifies R and V as the exact accounting rows for the dyadic endpoint comparisons and regular-name windows. Therefore the regular-window accounting is exactly W, Q, R against M, V, with no hidden tail ledger, host equality, quotient equality, choice, selected limit, or completeness row. □

Theorem 1428.29 (DyadicIntervalCover finite-subcover ledger)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Any finite-subcover-facing read of I is only the displayed ledger projection*

$$L, U, M, R, V, W, Q, A, H, C, P, N.$$

The rows L, U, M, R, V expose dyadic endpoint, mesh, radius, and membership data; the rows W, Q, A expose the finite regular-window readback and visible $\mathsf{Real}^{\uparrow}$ *seal; and H, C, P, N only transport, replay, source, and name that same ledger. The packet supplies no open-cover compactness theorem, no selected subcover from an arbitrary cover, no quotient real equality, no countable choice, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1428.1. Its cover-facing rows are the lower endpoints L, upper endpoints U, mesh row M, radius ledger R, and finite membership row V.

Use Definition 1428.28. It identifies the regular-window accounting as W, Q, R against M, V, so any finite-subcover-facing read is already pinned to the displayed regular windows and membership ledger.

The structural rows H, C, P, N preserve only those displayed coordinates. Since the carrier has no row for an arbitrary open cover, no selector for extracting a subcover, and no equality or completeness coordinate beyond the visible $\mathsf{Real}^{\uparrow}$ seal A, the read cannot become open-cover compactness, selected subcover data, quotient real equality, countable choice, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverFiniteSubcoverLedger`

Theorem 1428.30 (DyadicIntervalCover RegSeqRat route)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer route through I first projects the displayed finite rows*

$$W, Q \longmapsto M, R, V \longmapsto L, U, A, H, C, P, N.$$

Thus regular rational readback is consumed only as finite window data compared against the dyadic endpoint and membership ledger. The route contains no open-cover compactness theorem, no selected subcover, no quotient real equality, no countable choice, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the rows W and Q. By the carrier definition, W is the only $\mathsf{StreamName}^{\uparrow}$ finite-window row and Q is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback row available to the interval-cover packet.

Apply Definition 1428.6. It forces every regular-name cover decision through M, R, Q, V, A and excludes later regular-sequence tails and ambient equality of real names.

Now apply Definition 1428.28. The readback row Q is accounted together with W and R against the mesh and membership rows M, V. Endpoint rows L, U and the seal row A remain displayed packet data, while H, C, P, N only transport, replay, package, and name the same finite route. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegSeqRatRoute`

Theorem 1428.31 (DyadicIntervalCover Real seal nonescape)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$ *seal read through A is downstream of the displayed endpoint, mesh, radius, membership, window, and readback rows L, U, M, R, V, W, Q and remains inside the structural rows H, C, P, N. The seal cannot export open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealSealNonescape`

Proof. Project A from the carrier. The seal row appears only after the dyadic endpoint rows, mesh row, radius ledger, membership row, finite StreamName window, and RegSeqRat readback row have been displayed.

Use Definition 1428.5. It bounds every seal read by the carrier rows and the local NameCert obligations, so the seal cannot leave the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert surface.

Finally apply Definition 1428.29. The finite ledger already denies arbitrary open-cover compactness and selected subcover data. Since quotient real equality, countable choice, and ambient Real completeness are also absent from I, no Real-seal consumer can recover them from A, H, C, P, N. □

Theorem 1428.32 (DyadicIntervalCover L10 finite-window obligations)**.** *Every* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *L10 consumer reads the finite cover surface of an accepted packet* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *only after consuming the dyadic endpoint rows* L, U*, mesh row* M*, radius ledger* R*, cover-membership row* V*,* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, visible* $\mathsf{Real}^{\uparrow}$ *seal* A*,* hsame *transport row* H*,* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N*. The exported route is therefore*

$$L, U, M, R, V, W, Q, A, H, C, P, N,$$

and it contains no open-cover compactness theorem, selected subcover, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverL10WindowObligations`

Proof. Apply Definition 1428.23. It places every refinement read behind the endpoint rows, mesh row, radius ledger, membership row, and structural replay data.

Apply Definition 1428.25. The Real-facing route must pass through the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic cover ledger, and visible $\mathsf{Real}^{\uparrow}$ seal.

Next use Definition 1428.26 and Definition 1428.27. Membership transport remains inside the displayed endpoint, mesh, radius, readback, and replay rows, while the seal cannot leave the finite window and readback route.

Finally apply Definition 1428.28. It identifies W, Q, R against M, V as the whole regular-window accounting surface. Since the carrier has no coordinate for open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no L10 interval-cover consumer can read such data from I. □

Theorem 1428.33 (DyadicIntervalCover root window obligations)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ BHist *packet. Every root-window obligation for* I *is read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows* L, U, R*, and visible* $\mathsf{Real}^{\uparrow}$ *seal* A *before the local* NameCert *row* N *is exported. The root window has no coordinate for open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootWindowObligations`

Proof. Project the carrier of Definition 1428.1. The BHist packet lists the endpoint rows, mesh row, radius ledger, membership row, finite window, readback, Real seal, transport, replay, provenance, and local naming row as its entire root surface.

Apply Definition 1428.8. It identifies L, U, M, R, V, W, Q, A as the whole interval-window data before the local NameCert row is exported.

Use Definition 1428.6 to bind regular-name reads to W, Q, M, R, V, A. Since L, U, R are the dyadic endpoint and tolerance data and A is the only Real seal, every root-window obligation is exhausted by the stated finite route. □

Theorem 1428.34 (DyadicIntervalCover root refinement obligations)**.** *For accepted* DyadicIntervalCover$^{\uparrow}$ *packets* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *and* $I' = (L', U', M', R', V', W', Q', A', H', C', P', N')$*, any root refinement displayed by the packet is an* hsame *and* Cont *route over endpoint, mesh, radius, membership, window, readback, and seal rows. Refinement may sharpen* M *and compare the dyadic endpoint and radius rows through* DyadicRatCore$^{\uparrow}$*, but it preserves the same* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootRefinementObligations`

Proof. Project both carriers. The only rows that can vary under refinement are the displayed endpoint, mesh, radius, membership, finite-window, readback, seal, transport, replay, provenance, and local naming rows.

Apply Definition 1428.11. Componentwise hsame data and the displayed Cont route preserve the same finite interval-window route through the L10 siblings.

Use Definition 1428.26. The membership row is transported only along H and replayed only along C over L, U, M, R, W, Q, A. Thus the root refinement obligation is endpoint and mesh refinement inside the displayed packet, not a selected subcover, host equality, quotient equality, choice principle, or Real completeness claim. □

Theorem 1428.35 (DyadicIntervalCover root ledger obligations)**.** *For every accepted* DyadicIntervalCover$^{\uparrow}$ *carrier* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$*, the root ledger obligations are exhausted by the radius ledger* R *and finite membership row* V *against the endpoint rows* L, U*, mesh row* M*, finite window* W*, readback row* Q*, Real seal* A*, and structural rows* H, C, P, N*. No hidden cover ledger, open-cover selector, later regular-sequence tail, quotient real equality, countable choice, or ambient completeness datum is available outside those rows.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootLedgerObligations`

Proof. Begin with Definition 1428.28. It identifies the regular-window ledger as the finite accounting of W, Q, R against the mesh and membership rows M, V.

Next use Definition 1428.13. The radius and membership rows are the exact accounting rows for the endpoint comparisons and regular-name windows.

Finally read the structural rows H, C, P, N through the carrier. They transport, replay, package, and name the same finite ledger, but they add no hidden cover list, selector, later tail, quotient equality, choice schedule, or completeness coordinate. Therefore the root ledger obligations are exhausted by the displayed finite packet. □

Theorem 1428.36 (DyadicIntervalCover root cell carrier rows)**.** *For every accepted* DyadicIntervalCover$^{\uparrow}$ *carrier* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$*, the root cell carrier rows consist exactly of the dyadic endpoint cell rows* L, U*, the mesh-level row* M*, the finite coverage ledger rows* R, V*, and the displayed* Cont *replay row* C *over the same cover surface. These rows are transported by* H*, packaged by* P*, and named by* N*, and they do not expose open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootCellCarrier`

Proof. Project the accepted carrier through Definition 1428.1. The endpoint cells are exactly L, U, the displayed mesh level is M, and the coverage accounting appears only as the radius and membership rows R, V.

Apply Definition 1428.35. It exhausts the root ledger by R, V against $L, U, M, W, Q, A, H, C, P, N$, so no hidden cover ledger or selector remains outside the displayed carrier.

Finally read the replay coordinate C together with H, P, N. These structural rows preserve the same finite cell surface and cannot add compactness, quotient equality, choice, or Real-completeness data. □

Theorem 1428.37 (DyadicIntervalCover root refinement coverage)**.** *For any finite displayed chain of root refinements of accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packets, induction over the refinement windows preserves coverage through the same endpoint cell rows, mesh-level rows, coverage ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. Preservation under* hsame *is a rowwise coverage statement, not an open-cover compactness theorem or selected-subcover principle.*

Proof. Use finite induction on the displayed refinement chain. The base case is the root cell carrier surface of Definition 1428.36: the cover is read only through $L, U, M, R, V, W, Q, A, H, C, P, N$.

For the successor step, apply Definition 1428.34. A refinement step transports endpoint, mesh, radius, membership, window, readback, and seal rows through componentwise hsame and displayed Cont replay.

It remains to preserve coverage rather than merely row shape. Use Definition 1428.35; the finite coverage ledger is exhausted by R, V at each stage, so the induction keeps the cover within the same displayed rows and introduces no selected subcover, quotient equality, choice schedule, or Real-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootRefinementCoverage`

Theorem 1428.38 (DyadicIntervalCover root Real-seal handoff)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer reading a root* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *seal must first pass through* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint comparison, the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$ *coverage surface. Only after those rows are displayed may the consumer read the Real seal A, and that read does not import host real equality, quotient stream equality, selected limits, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRootRealSealHandoff`

Proof. Start with Definition 1428.3. The lower and upper endpoint cells enter only as $\mathsf{DyadicRatCore}^{\uparrow}$ rows, with the mesh and ledger data displayed before any Real-facing seal is public.

Next read the finite window and readback rows. By Definition 1428.36, the $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{RegSeqRat}^{\uparrow}$ readback Q remain attached to the endpoint, mesh, radius, and membership rows.

Finally apply Definition 1428.35. The local NameCert surface forces the consumer through R, V, H, C, P, N before the Real seal A is read, so the handoff exports only a finite cover read and no host equality, quotient equality, selected limit, choice, compactness, or completeness claim. □

Theorem 1428.39 (DyadicIntervalCover cell refinement exhaustion)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every cell-refinement read of I is exhausted by the lower and upper endpoint rows L, U, mesh row M, radius ledger R, membership row V, and the transport, replay, provenance, and local naming rows H, C, P, N, with the StreamName, RegSeqRat, and Real-seal rows W, Q, A serving only as the finite readback surface for those same cells. No refinement read imports open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted packet through Definition 1428.8. The displayed cover surface is exactly the finite endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and naming data carried by $L, U, M, R, V, W, Q, A, H, C, P, N$.

Apply Definition 1428.11. A refinement can only transport endpoint, mesh, radius, membership, window, readback, and seal rows along componentwise hsame data and the displayed Cont replay route.

Now use Definition 1428.13 and Definition 1428.35. They identify R, V as the exact finite ledger against $L, U, M, W, Q, A, H, C, P, N$. Thus the cell-refinement read is exhausted by the displayed rows and has no coordinate for compactness, selected subcovers, quotient equality, choice, or Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverCellRefinementExhaustion`

Theorem 1428.40 (DyadicIntervalCover refinement-chain induction)**.** *For any finite displayed chain of accepted* DyadicIntervalCover$^{\uparrow}$ *refinements*

$$I_0 \rightsquigarrow I_1 \rightsquigarrow \cdots \rightsquigarrow I_k,$$

the cover surface at every stage is preserved as the same finite route through endpoint rows, mesh rows, radius ledgers, membership rows, StreamName windows, RegSeqRat readbacks, Real seals, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The chain induction exports no selected subcover, open-cover compactness theorem, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Use finite induction on the displayed refinement chain. The base stage is Definition 1428.8: the packet exposes only the finite cover surface $L, U, M, R, V, W, Q, A, H, C, P, N$.

For the successor step, apply Definition 1428.11. The refinement step preserves the endpoint, mesh, radius, membership, window, readback, and seal rows by componentwise transport and replay through H, C, P, N.

It remains to keep the ledger bounded along the chain. Use Definition 1428.13 together with Definition 1428.35: each stage reads only R, V against $L, U, M, W, Q, A, H, C, P, N$. Therefore finite induction preserves the same cover surface and introduces no hidden compactness, selected subcover, quotient equality, choice, or completeness row. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRefinementChainInduction`

Theorem 1428.41 (DyadicIntervalCover Real-seal cover handoff)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Any* Real$^{\uparrow}$ *seal* A *consumed by a cover reader factors through the finite window row* W*, regular-rational readback row* Q*, mesh row* M*, radius ledger* R*, and membership row* V *before the seal is exposed to the reader. The handoff may then be transported, replayed, packaged, and named by* H, C, P, N*, but it cannot expose host real equality, quotient stream equality, selected limits, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealSealCoverHandoff`

Proof. Start with Definition 1428.8. It places the Real seal A behind the displayed finite cover surface rather than before it.

Next use Definition 1428.35. The ledger obligations for a cover reader are exhausted by R, V against $L, U, M, W, Q, A, H, C, P, N$, so the seal has to pass through W, Q, M, R, V before it can be consumed as a cover read.

Finally apply Definition 1428.13 and Definition 1428.11. Exactness keeps the handoff inside the finite ledger, while refinement stability keeps transport and replay inside H, C, P, N. Hence the Real-seal handoff exports no host equality, quotient stream, selected limit, choice principle, compactness theorem, or completeness claim. □

Theorem 1428.42 (DyadicIntervalCover mesh refinement induction)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ BHist *packet. Every mesh refinement read of I is obtained by finite induction over the displayed mesh rows: the base row reads L, U, M, R, V through* DyadicRatCore$^{\uparrow}$*, and each successor row transports the refined endpoint, radius, membership,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *data through the same* hsame*,* Cont*,* Pkg*, and* NameCert *surface. No refinement step imports open-cover compactness, selected subcovers, quotient stream equality, countable choice, or Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMeshRefinementInduction`

Proof. Project the carrier of Definition 1428.1. The mesh row M and radius ledger R are finite BHist rows, and the endpoint and membership rows L, U, V are the only cover-facing rows available at the base mesh.

Apply Definition 1428.34. A displayed refinement is an hsame and Cont route over endpoint, mesh, radius, membership, window, readback, and seal rows, so the induction step preserves the same finite packet surface.

Use Definition 1428.35. The ledger is exhausted by R, V against $L, U, M, W, Q, A, H, C, P, N$, hence the induction has no extra coordinate from which to read compactness, a selected subcover, quotient stream equality, countable choice, or Real completeness. □

Theorem 1428.43 (DyadicIntervalCover finite mesh induction)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Finite induction over the displayed mesh rows preserves the endpoint rows L, U, mesh and radius rows M, R, membership row V,* StreamName$^{\uparrow}$ *window W,* RegSeqRat$^{\uparrow}$ *readback Q,* Real$^{\uparrow}$ *seal A, and structural rows H, C, P, N. The induction exports no selected subcover, quotient equality, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverFiniteMeshInduction`

Proof. Apply Definition 1428.42 for the finite induction spine and Definition 1428.34 for the endpoint, mesh, radius, membership, window, readback, and seal rows preserved at each step.

Use Definition 1428.13 to keep the route exhausted by $L, U, M, R, V, W, Q, A, H, C, P, N$, and then apply Definition 1428.51. The Real seal remains downstream of the same finite projection, so the induction has no coordinate for selected subcovers, quotient equality, countable choice, open-cover compactness, or ambient Real completeness. □

Theorem 1428.44 (DyadicIntervalCover endpoint comparison totality)**.** *For an accepted* DyadicIntervalCover$^{\uparrow}$ *carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$, every cover decision compares lower and upper* DyadicRatCore$^{\uparrow}$ *endpoint rows through the displayed mesh and tolerance ledger before the terminal* Real$^{\uparrow}$ *seal is read. The comparison route is total for the finite displayed cover rows because each membership row V is checked against L, U, M, R, W, Q and then transported, replayed, packaged, and named only by H, C, P, N.*

Proof. Start with Definition 1428.3. It admits L and U only as DyadicRatCore$^{\uparrow}$ endpoint rows and places M, R beside them as the finite mesh and tolerance data for comparison.

Next apply Definition 1428.28. The regular-window ledger accounts for W, Q, R against M, V, so every finite cover decision has a displayed row in which endpoint comparison, tolerance, window, and readback data meet.

Finally use Definition 1428.31. The $\mathsf{Real}^{\uparrow}$ seal A is downstream of L, U, M, R, V, W, Q and remains inside H, C, P, N. Thus endpoint comparison is total for the finite displayed ledger and cannot be replaced by host real equality, quotient equality, choice, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverEndpointComparisonTotality`

Theorem 1428.45 (DyadicIntervalCover tail pullback stability)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Pulling a* $\mathsf{StreamName}^{\uparrow}$ *consumer back to any displayed finite tail window preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$ *route W, Q and the same dyadic cover ledger L, U, M, R, V before the* $\mathsf{Real}^{\uparrow}$ *seal A is readable. The pullback route has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or Real completeness.*

Proof. Project the window and readback rows from Definition 1428.1. The row W is the only $\mathsf{StreamName}^{\uparrow}$ finite-window source, and Q is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback row attached to it.

Apply Definition 1428.6. Every regular-name cover read is exhausted by a finite window computation through M, R, Q, V, A, so pulling back to a displayed tail window changes only the finite window presented to the same route.

Use Definition 1428.11. The componentwise hsame row and displayed Cont replay preserve endpoint, mesh, radius, membership, window, readback, and seal rows, while Pkg provenance and $\mathsf{NameCert}$ naming keep the packet local. Hence tail pullback cannot import quotient stream equality, selected limits, countable choice, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverTailPullbackStability`

Theorem 1428.46 (DyadicIntervalCover scoped mesh refinement)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Every scoped mesh-refinement obligation is bound to the finite endpoint rows L, U over* $\mathsf{DyadicRatCore}^{\uparrow}$*, the displayed mesh row M, the cover-membership row V, the* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *finite-cover membership handoff, and the structural rows H, C, P, N. Refinement changes only displayed cover rows and never imports selected subcovers, quotient real equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows L, U, M, R, V from Definition 1428.1. The lower and upper endpoints are dyadic rows, and the mesh and membership data are finite cover rows attached to those endpoints.

Proceed by induction over the displayed refinement rows. The base case is the admitted mesh row M with its membership row V. The successor step uses componentwise transport H and displayed replay C to replace each endpoint, mesh, radius, and membership row by its refined displayed row.

The $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff reads only this finite cover membership surface. Since P, N source and name the same packet, the scoped obligation remains a finite refinement route and cannot acquire an unlisted compactness or choice principle. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMeshRefinementScope`

Theorem 1428.47 (DyadicIntervalCover scoped finite-window coverage)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P,$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Every finite-window coverage obligation factors through the* StreamName$^{\uparrow}$ *window row W, the* RegSeqRat$^{\uparrow}$ *readback row Q, the dyadic endpoint rows L, U, the mesh and radius rows M, R, and the cover-membership row V before any* Real$^{\uparrow}$ *seal is read. The coverage scope is finite and visible to* BishopIntervalCompact$^{\uparrow}$ *only through those rows.*

Proof. Begin with a coverage request for a displayed interval cell. The request first reads the endpoint rows L, U and the mesh-radius data M, R, because those rows determine which finite cover membership rows can be inspected.

The finite observation boundary is then supplied by W and Q. The stream row selects the visible windows, and the regular rational row packages the endpoint and radius reads before the real seal is available.

Finally use V, H, C, P, N to record membership, transport it along the displayed route, replay the same read, source it, and name it. This proves finite-window coverage by packet projection, not by a selected finite subcover or an open-cover compactness theorem. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverWindowCoverageScope`

Theorem 1428.48 (DyadicIntervalCover scoped Real-seal handoff)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Every scoped* Real$^{\uparrow}$ *handoff reaches the seal row A only after the finite* DyadicRatCore$^{\uparrow}$ *endpoint rows,* StreamName$^{\uparrow}$ *window row,* RegSeqRat$^{\uparrow}$ *readback row,* BishopIntervalCompact$^{\uparrow}$ *membership handoff, mesh row, radius ledger, and cover-membership rows have been displayed. The seal handoff exports no host real equality, quotient stream equality, selected subcover, countable choice, or ambient Real completeness.*

Proof. Read the seal route from the displayed carrier. The endpoint rows L, U and the mesh-radius rows M, R determine the finite cover cells, while V records membership for the cover consumer.

Next pass through W and Q. These rows make the cover data readable as finite stream windows and regular rational readbacks before the seal row A can be inspected.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. The BishopIntervalCompact$^{\uparrow}$ handoff therefore receives a finite Real-seal boundary, not a host compactness theorem or a completion principle. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealSealScope`

Theorem 1428.49 (DyadicIntervalCover mesh refinement obligation)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *carrier. Every mesh-refinement obligation for I is discharged by finite projection through the displayed endpoint rows L, U, mesh row M, radius ledger R, membership row V,* StreamName$^{\uparrow}$ *window row W,* RegSeqRat$^{\uparrow}$ *readback row Q,* Real$^{\uparrow}$ *seal row A, and the structural rows H, C, P, N. The refinement may only sharpen displayed dyadic cells through* DyadicRatCore$^{\uparrow}$ *and then replay the same route through* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with Definition 1428.42. It reads mesh refinement by finite induction over the displayed mesh rows, with the base step in L, U, M, R, V and each step transported through the same H, C, P, N surface.

Use Definition 1428.34 and Definition 1428.40. These keep the endpoint, mesh, radius, membership, window, readback, and seal rows componentwise visible along the refinement chain.

Finally apply Definition 1428.13. Since the ledger is exhausted by R, V against $L, U, M, W, Q, A, H, C, P, N$, the obligation has no source for selected subcovers, quotient equality, countable choice, open-cover compactness, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMeshRefinementObligationCertificate`

Theorem 1428.50 (DyadicIntervalCover subcover handoff obligation)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *carrier. Every finite subcover handoff consumed by a* BishopIntervalCompact$^{\uparrow}$ *or Completion-facing reader must first display the same rows $L, U, M, R, V, W, Q, A, H, C, P, N$: dyadic endpoints through* DyadicRatCore$^{\uparrow}$*, finite windows through* StreamName$^{\uparrow}$*, regular readback through* RegSeqRat$^{\uparrow}$*, the visible seal through* Real$^{\uparrow}$*, and the local route through* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Begin with Definition 1428.9. It places the compact-interval consumer behind the dyadic endpoint, mesh, radius, membership, window, readback, Real-seal, transport, replay, provenance, and naming rows.

Next apply Definition 1428.47. The coverage request factors through W, Q, L, U, M, R, V before any Real seal is consumed, so the handoff is a finite-window coverage row rather than an open-cover theorem.

The structural rows complete the handoff. By Definition 1428.36, H, C, P, N transport, replay, source, and name the same finite cell surface; no selected subcover, host compactness theorem, quotient real equality, countable choice, or ambient completion row is exported. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverSubcoverHandoffObligation`

Theorem 1428.51 (DyadicIntervalCover Real-seal nonescape obligation)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *carrier. Every* Real$^{\uparrow}$ *seal read from A is a nonescaping finite projection: it passes through* DyadicRatCore$^{\uparrow}$ *endpoint and radius rows,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, cover-membership rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before any Real-facing consumer may inspect the seal.*

Proof. Read the existing seal boundary through Definition 1428.5. The row A is the only Real-seal coordinate and is bounded by $L, U, M, R, V, W, Q, H, C, P, N$.

Then use Definition 1428.41 and Definition 1428.48. Both place the seal after the finite window, regular readback, mesh, radius, and membership rows, with the BishopIntervalCompact$^{\uparrow}$ handoff still inside the displayed route.

The nonescape claim follows from carrier exactness and ledger exactness. The accepted packet has no coordinate for host real equality, quotient stream equality, selected limits, countable choice, open-cover compactness, selected subcovers, or ambient Real completeness, so the seal cannot leave the finite projection route. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRealSealNonescapeObligation`

Theorem 1428.52 (DyadicIntervalCover scoped finite-cover chain)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Every scoped finite-cover chain exposed by I factors through lower and upper endpoint rows in* DyadicRatCore$^{\uparrow}$*, finite-cover membership rows consumed by* BishopIntervalCompact$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$ *readback through W, Q, and the terminal* Real$^{\uparrow}$ *seal row A. The structural rows H, C, P, N preserve the same finite route by* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming; the chain does not assert open-cover compactness, host selected subcovers, quotient real equality, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverScopedFiniteCoverChain`

Proof. Project the endpoint, mesh, radius, and membership rows from Definition 1428.1. The rows L, U are the dyadic endpoints, while M, R, V are the finite cells and membership data visible to the interval-compactness consumer.

Apply Definition 1428.50. It places any $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff behind the same displayed rows $L, U, M, R, V, W, Q, A, H, C, P, N$ and treats the handoff as a finite membership route rather than an open-cover theorem.

Use Definition 1428.51. The readback rows W, Q and the seal row A remain downstream of the displayed cover chain, so $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive only the finite route. The structural rows preserve that route and add no selected subcover or ambient compactness coordinate. □

Theorem 1428.53 (DyadicIntervalCover public certificate scope)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. The public certificate scope of I is exactly the finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows L, U,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q,* $\mathsf{Real}^{\uparrow}$ *seal row A, the* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *finite-net handoff, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N, with mesh, radius, and membership rows M, R, V retained as the cover ledger.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverPublicCertificateScope`

Proof. Start with Definition 1428.52. It already places the finite-cover chain inside the displayed rows $L, U, M, R, V, W, Q, A, H, C, P, N$ and keeps the route away from open-cover compactness, host selected subcovers, quotient real equality, countable choice, and ambient Real completeness.

Next use Definition 1428.9. The finite-net handoff to $\mathsf{BishopIntervalCompact}^{\uparrow}$ is available only after the dyadic endpoints, mesh, radius, cover membership, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming rows have been exposed.

Finally apply Definition 1428.13. The ledger rows M, R, V are exact only against the same endpoint, window, readback, seal, transport, replay, provenance, and NameCert route. Therefore the public certificate scope is precisely that finite route, with no extra coordinate for selected subcovers, quotient equality, choice, host compactness, or ambient completion. □

Theorem 1428.54 (DyadicIntervalCover public finite-cover export)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. The public export surface of I is the single finite-cover NameCert route that displays the finite mesh M, dyadic endpoint rows L, U, cover-membership ledger V,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q,* $\mathsf{Real}^{\uparrow}$ *seal row A, and* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff through H, C, P, N. This route exports finite-cover data only; it still refuses selected finite subcovers, quotient real equality, countable choice, open-cover compactness, host compactness, and ambient completion.*

Proof. Start from Definition 1428.53. It identifies the public certificate scope as the finite route through dyadic endpoints, StreamName windows, RegSeqRat readback, Real sealing, Bishop interval compact handoff, transport, replay, provenance, local naming, and the cover ledger rows M, R, V.

Use Definition 1428.50. The $\mathsf{BishopIntervalCompact}^{\uparrow}$ consumer is reached only after the dyadic endpoint, mesh, radius, membership, window, readback, Real-seal, transport, replay, provenance, and naming rows are displayed.

Finally apply Definition 1428.49 and Definition 1428.51. The first keeps the finite mesh and refinement data inside the displayed dyadic route, while the second keeps the Real seal downstream of the same finite projection. Hence the public export is the named finite-cover NameCert route itself, with no extra selector, quotient, choice, compactness, or completion coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverPublicFiniteCoverExport`

Theorem 1428.55 (DyadicIntervalCover bridge handoff)**.** *Every bridge-facing consumer of the public finite-cover export of Definition 1428.54 must preserve the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, mesh row, cover-membership ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The handoff is the finite public-cover route itself; it does not claim bridge checking, open-cover compactness, selected finite subcovers, quotient real equality, countable choice, or ambient completion.*

Proof. Start with the public finite-cover export. It identifies the available public surface as the displayed rows $L, U, M, R, V, W, Q, A, H, C, P, N$ together with the $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff.

Use the scoped Real-seal and nonescape obligations, Definition 1428.48 and Definition 1428.51. They keep every bridge-facing read downstream of the dyadic endpoints, mesh, membership, window, readback, Real seal, transport, replay, provenance, and naming rows.

Since the carrier has no coordinate for open-cover compactness, selected finite subcovers, quotient real equality, countable choice, or ambient completion, a bridge-facing consumer can preserve only the displayed finite-cover route. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeHandoff`

Theorem 1428.56 (DyadicIntervalCover content route preservation)**.** *Every Real-facing consumer that uses the bridge handoff of Definition 1428.55 preserves the same* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and interval-cover rows exposed by the public finite-cover export. Thus the moved route remains the displayed finite-cover NameCert route through dyadic endpoints, mesh and cover-membership data, stream windows, regular rational readback, Real sealing, interval compactness handoff, transport, replay, provenance, and local naming.*

Proof. Apply the public finite-cover export of Definition 1428.54. It names the public route through $L, U, M, R, V, W, Q, A, H, C, P, N$, with dyadic endpoints, mesh and membership data, stream windows, regular readback, the Real seal, and the interval-compactness handoff all displayed.

Then apply the bridge handoff of Definition 1428.55. A bridge-facing consumer must preserve exactly that displayed route and cannot replace it with open-cover compactness, selected finite subcovers, quotient real equality, countable choice, or ambient completion.

The Real-facing part is therefore still the same finite route: L, U give the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint surface, W gives the $\mathsf{StreamName}^{\uparrow}$ window, Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ readback, A gives the $\mathsf{Real}^{\uparrow}$ seal, and M, R, V, H, C, P, N preserve the interval-cover carrier and NameCert structure. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverContentSplitRoute`

Theorem 1428.57 (DyadicIntervalCover bridge budget)**.** *Let* $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. The displayed dyadic interval-cover rows provide exactly the bridge budget consumed by* $\mathsf{BishopIntervalCompact}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *routes: dyadic endpoints and mesh data, finite cover-membership rows, stream windows, regular-rational readbacks, the visible Real seal, transport, replay, provenance, and local naming. The budget exports no open-cover compactness, selected limit, selected finite subcover, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Start with the local obligation surface of Definition 1428.2. It exhausts every endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and naming obligation by finite projection from I.

Use the public export theorem, Definition 1428.54. It identifies the public surface as the finite-cover NameCert route through $L, U, M, R, V, W, Q, A, H, C, P, N$ and the $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff.

The same rows are the only data visible to the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ consumers. Since the accepted carrier has no coordinate for open-cover compactness, selected limits, selected subcovers, quotient equality, choice, or ambient completeness, the bridge budget is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeBudget`

Theorem 1428.58 (DyadicIntervalCover finite-cover bridge scope)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Any bridge consumer of I must expose the dyadic endpoint rows L, U, the mesh row M, the radius ledger R, the finite cover-membership row V, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{Real}^{\uparrow}$ *seal A, the* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff, the* $\mathsf{UniformModulus}^{\uparrow}$*-facing lower-bound route, componentwise transport H, displayed replay C, provenance P, and local naming N before the bridge budget can be read. The bridge scope exports finite-cover data only; it does not claim open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverFiniteCoverBridgeScope`

Proof. Start from the public finite-cover export, Definition 1428.54. It identifies the public surface as the finite route through $L, U, M, R, V, W, Q, A, H, C, P, N$ and the $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff.

Use the bridge budget theorem, Definition 1428.57. The budget is read only by the displayed dyadic, stream, regular-rational, Real, interval-compact, and uniform-modulus-facing consumers.

Combining the two statements gives the bridge scope. Every bridge consumer must first expose the finite endpoint, mesh, radius, membership, window, readback, seal, handoff, transport, replay, provenance, and naming rows, while the refused compactness, selector, quotient, choice, and completion endpoints remain outside the packet. □

Theorem 1428.59 (DyadicIntervalCover regular Cauchy bridge window)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P,$ be an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet. Any bridge consumer that reads a regular Cauchy name through I must first expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows L, U, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row A, the* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *finite-cover handoff, the* $\mathsf{UniformModulus}^{\uparrow}$ *lower-bound route, and the structural rows H, C, P, N. Only after that bridge window is displayed may the consumer read the bridge budget; the packet exports no selected limit, selected finite subcover, quotient real equality, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegularCauchyBridgeWindow`

Proof. Begin with finite-window exhaustion from Definition 1428.47. It places the stream window and regular-rational readback behind the displayed dyadic endpoint, mesh, radius, and membership rows.

Next use the public finite-cover export Definition 1428.54. The export identifies the $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff as part of the same finite route, with the Real seal still downstream of the cover-membership rows.

Apply the bridge budget Definition 1428.57 and the finite-cover bridge scope Definition 1428.58. Together they force every bridge consumer to expose the dyadic endpoints, StreamName windows,

RegSeqRat readback, Real seal, finite-cover handoff, uniform-modulus lower-bound route, and structural rows before any budget is consumed. □

Theorem 1428.60 (DyadicIntervalCover bridge export)**.** *Every accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *bridge export is the finite route isolated in Definition 1428.58: the dyadic endpoint and mesh rows* L, U, M*, radius budget* R*, finite cover-membership rows* V*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal row* A*, and structural rows* H, C, P, N*. No bridge consumer receives a selected finite subcover, quotient real equality, open-cover compactness, host compactness, countable choice, or ambient completion coordinate from the export.*

Proof. Read the bridge scope theorem Definition 1428.58. It names the endpoint rows L, U, mesh row M, radius budget R, finite cover-membership rows V, stream window row W, regular rational readback row Q, Real seal row A, and structural rows H, C, P, N before any bridge consumer can inspect the cover export.

The present theorem adds no carrier coordinate beyond that list. Its export is therefore the projection that forgets the internal proof route and keeps exactly the displayed endpoint, mesh, radius, membership, stream, readback, seal, transport, replay, provenance, and local naming rows.

Since the source bridge scope excludes selected finite subcovers, quotient real equality, open-cover compactness, host compactness, countable choice, and ambient completion, the projected export excludes them as well. A consumer can only replay the finite route already isolated by the bridge scope. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeExport`

Theorem 1428.61 (DyadicIntervalCover uniform-modulus bridge consumer)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or compact-uniform-continuity consumer of an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *bridge reads exactly the finite endpoint rows, mesh row, radius budget, finite cover-membership rows,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural* hsame*,* Cont*,* Pkg*, and* NameCert *rows isolated by Definition 1428.60. The consumer receives no selected finite subcover, open-cover compactness, quotient real equality, choice coordinate, or ambient completion coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverUniformModulusBridgeConsumer`

Proof. Start with the bridge export theorem. It projects the accepted bridge to the endpoint rows L, U, mesh row M, radius budget R, cover-membership rows V, stream window row W, regular rational readback row Q, Real seal row A, and structural rows H, C, P, N.

Apply the finite-cover bridge scope. The scope fixes the same rows before a downstream consumer can inspect the cover data, so a $\mathsf{UniformModulus}^{\uparrow}$ or compact-uniform-continuity reader can only replay that finite route.

Finally use the L10 uniform-modulus handoff narrative. Its consumer-facing input is the exported endpoint, mesh, radius, membership, stream, readback, seal, transport, replay, provenance, and local naming surface, not a selected subcover, an open-cover compactness principle, quotient real equality, choice, or an ambient completion coordinate. □

Theorem 1428.62 (DyadicIntervalCover bridge export cofinality)**.** *Every downstream compact-uniform-continuity or* $\mathsf{UniformModulus}^{\uparrow}$ *read of a* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *bridge export is cofinal in the displayed finite route*

$$L, U, M, R, V, W, Q, A, H, C, P, N.$$

Any precision request reaching $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *must first pass through the finite endpoint rows* L, U*, mesh row* M*, radius-budget row* R*, and cover-membership rows* V *isolated by Definition 1428.60.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeExportCofinality`

Proof. Apply Definition 1428.60. The export exposes exactly the endpoint, mesh, radius-budget, membership, stream-window, readback, seal, transport, replay, provenance, and local naming rows, so every downstream read starts from that finite coordinate list.

Use Definition 1428.61. A compact-uniform-continuity or $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the same rows and no further bridge coordinate, hence its precision requests cannot reach W, Q, A before the endpoint, mesh, radius, and membership rows have been displayed.

The cofinality claim follows by coordinate exhaustion. The listed route contains no selected finite subcover, open-cover compactness proof, quotient real equality, choice coordinate, or ambient completion row, so any consumer reaching $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ has already passed through the displayed finite endpoint, mesh, radius-budget, and cover-membership data. □

Theorem 1428.63 (DyadicIntervalCover bridge finite route)**.** *Every bridge consumer of an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet factors through the finite endpoint rows, mesh row, radius ledger, finite membership row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before any standard interval-cover interpretation is read. The bridge route exports only those displayed finite rows; it exports no open-cover compactness, selected finite subcover, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$. The rows L, U are the endpoint data, M is the mesh row, R is the radius ledger, and V is the finite membership row.

Read the window and seal rows next. The $\mathsf{StreamName}^{\uparrow}$ windows W, $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and $\mathsf{Real}^{\uparrow}$ seal A are downstream of the endpoint, mesh, radius, and membership rows.

Use the existing compact handoff and bridge export surface. The $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff is read only after those finite rows, while H, C, P, N transport, replay, source, and name the same route.

Thus any standard interval-cover interpretation available to a bridge consumer is a readback of $L, U, M, R, V, W, Q, A, H, C, P, N$. Since no excluded compactness, selector, quotient, choice, or completion coordinate occurs in the carrier, none can be exported by the bridge route. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeFiniteRoute`

Theorem 1428.64 (DyadicIntervalCover finite bridge budget)**.** *Any bridge read of an accepted* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *packet must consume the endpoint rows, mesh row, radius ledger, finite membership row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *handoff before a standard interval-cover interpretation is visible. The budget exports no selected finite subcover, open-cover compactness, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverFiniteBridgeBudget`

Proof. Project the same carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ used by Definition 1428.63. The endpoint rows L, U, mesh row M, radius ledger R, and finite membership row V are the first finite data consumed by the bridge.

The observation and seal rows follow. The $\mathsf{StreamName}^{\uparrow}$ windows W, $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and $\mathsf{Real}^{\uparrow}$ seal A make the endpoint and radius data visible as finite real-facing reads.

The $\mathsf{BishopIntervalCompact}^{\uparrow}$ handoff is available only after those rows, and the structural rows H, C, P, N preserve the same route. Since the carrier contains no selector, open-cover compactness,

quotient equality, choice, or ambient completeness coordinate, the finite bridge budget cannot export any of them. □

Theorem 1428.65 (DyadicIntervalCover bridge refinement boundary)**.** *A standard interval-cover bridge for an accepted* DyadicIntervalCover$^{\uparrow}$ *packet may refine endpoint rows, mesh rows, radius ledgers, membership rows,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, and* Real$^{\uparrow}$ *seals only by componentwise* hsame *transport and displayed* Cont *replay over the same finite packet. The bridge cannot introduce open-cover compactness, a selected finite subcover, quotient real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the bridge finite route of Definition 1428.63. It makes every bridge read pass through the finite endpoint, mesh, radius, membership, stream-window, readback, seal, compact-handoff, transport, replay, provenance, and local naming rows.

Apply the finite bridge budget of Definition 1428.64. Any bridge refinement must therefore act on the displayed rows already present in $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$.

Componentwise hsame transport refines the endpoint, mesh, radius, membership, window, readback, and seal rows without changing the packet boundary. Displayed Cont replay then reruns the same bridge route over the refined rows, while P, N keep the provenance and local naming attached to the original finite packet.

Since the refinement never leaves those rows, it has no coordinate from which to obtain open-cover compactness, a selected finite subcover, quotient real equality, countable choice, or ambient Real$^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeRefinementBoundary`

Theorem 1428.66 (DyadicIntervalCover bridge-budget child witness)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *packet. Any consumer that reads a finite-cover bridge from the bridge budget must first display the dyadic endpoint rows L, U, the mesh and refinement rows M, R, V, the* StreamName$^{\uparrow}$ *windows W, the* RegSeqRat$^{\uparrow}$ *readback Q, and the terminal* Real$^{\uparrow}$ *seal A, together with the* BishopIntervalCompact$^{\uparrow}$ *handoff and structural rows H, C, P, N. Without this displayed child witness, the bridge budget is not available as a finite-cover read.*

Proof. Project the existing bridge budget of Definition 1428.57. It identifies the budget as the displayed finite route through dyadic endpoints, mesh data, finite cover-membership rows, StreamName windows, RegSeqRat readbacks, the visible Real seal, transport, replay, provenance, and local naming.

Next use Definition 1428.58. That theorem forces every bridge consumer to expose the same rows before reading the budget, including the BishopIntervalCompact$^{\uparrow}$ handoff and the lower-bound route faced by uniform-modulus consumers.

Finally read the sibling routes named in the carrier: DyadicRatCore$^{\uparrow}$ supplies endpoint arithmetic, StreamName$^{\uparrow}$ supplies finite windows, RegSeqRat$^{\uparrow}$ supplies rational readback, Real$^{\uparrow}$ supplies the seal boundary, and BishopIntervalCompact$^{\uparrow}$ consumes only the displayed finite membership surface. Since none of these rows contains open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient Real completeness, the child witness is exactly the finite bridge-budget display. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeBudgetChildWitness`

Theorem 1428.67 (DyadicIntervalCover standard bridge premise)**.** *Every standard bridge read of an accepted* DyadicIntervalCover$^{\uparrow}$ *packet must expose the lower endpoint rows, upper endpoint rows,*

mesh row, radius ledger, finite membership row, StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal,* BishopIntervalCompact$^{\uparrow}$ *handoff,* UniformModulus$^{\uparrow}$ *lower-bound route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any interval-cover interpretation is public. The bridge premise exports no open-cover compactness theorem, selected subcover, quotient real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverStandardBridgePremise`

Proof. Project the accepted carrier $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ from Definition 1428.1. The rows L, U are the lower and upper endpoint rows, M is the mesh row, R is the radius ledger, and V is the finite cover-membership row.

Apply Definition 1428.63. It forces every bridge consumer through the endpoint, mesh, radius, membership, window, readback, seal, interval-compact handoff, transport, replay, provenance, and naming rows before a standard interval-cover interpretation can be read.

Next use Definition 1428.61. The UniformModulus$^{\uparrow}$-facing lower-bound route is therefore a consumer of the same displayed finite rows, not a detached open-cover or selected-subcover principle.

Finally apply Definition 1428.65. Componentwise hsame transport and displayed Cont replay can refine only the listed rows, while P, N preserve provenance and local naming. Since the carrier has no coordinate for open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient completeness, the standard bridge premise cannot export any of those objects. □

Theorem 1428.68 (DyadicIntervalCover bridge window)**.** *Let $I = (L, U, M, R, V, W, Q, A, H, C, P, N)$ be an accepted* DyadicIntervalCover$^{\uparrow}$ *finite-window carrier. If endpoint comparison, refinement-chain induction,* Real$^{\uparrow}$ *seal handoff, and finite-subcover ledger rows are all displayed for I, then I induces the bridge window required by the abstract dyadic interval cover interface. The induced window is the finite route through $L, U, M, R, V, W, Q, A, H, C, P, N$; it does not assert Heine–Borel compactness, open-cover compactness, selected limits, selected subcovers, quotient real equality, countable choice, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverBridgeWindow`

Proof. Start from the root finite-cover boundary of Definition 1428.17. It fixes the endpoint, mesh, radius, membership, window, readback, seal, transport, replay, provenance, and local naming rows before any interval-cover reader can inspect the carrier.

Apply mesh and refinement control through Definition 1428.42 and Definition 1428.40. Finite induction keeps each refined cell inside the same displayed rows, while endpoint comparison totality from Definition 1428.44 keeps the lower and upper endpoint decisions inside the dyadic ledger.

Finally compose the Real-facing and ledger boundaries. The Real$^{\uparrow}$ handoff of Definition 1428.41 and the nonescape obligation of Definition 1428.51 place the seal after the finite window and cover rows, while Definition 1428.29 accounts for the visible subcover ledger. The resulting bridge window is therefore a finite interface window only, with none of the excluded compactness, choice, limit, quotient, or standard-bridge data. □

Closure status for NameCert$_{\text{DyadicIntervalCover}^{\uparrow}}$.

Constructive story: A DyadicIntervalCover$^{\uparrow}$ packet is generated from dyadic lower and upper endpoint rows, a mesh index, a radius ledger, finite cover-membership rows, StreamName windows, RegSeqRat readbacks, a Real seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicIntervalCover}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1428.47, Definition 1428.50, Definition 1428.61, Definition 1428.13, and Definition 1428.55 bind the finite-window carrier, Bishop interval compact handoff, compact-uniform window route, ledger exactness surface, and bridge-facing public finite-cover route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel compactness, sequential compactness, selected limits, quotient real equality, host real equality, countable choice, Real completeness, or a standard bridge.

Upgrade path: Verification closure requires Lean coverage for the public finite-cover export route and then a standard bridge that preserves the same refusal surface.

Theorem 1428.69 (DyadicIntervalCover mature finite-cover export)**.** *Every mature* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *consumer that exports finite-cover data must first factor through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *finite-cover handoff,* $\mathsf{UniformModulus}^{\uparrow}$ *lower-bound route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The mature export is public only after those rows preserve the finite cover route isolated by the bridge-facing children; it does not export open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host compactness, or a hidden completion coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverMatureFiniteCoverExport`

Proof. Start with the public finite-cover export of Definition 1428.54. It exposes the accepted packet only through dyadic endpoints, mesh and radius data, finite cover-membership rows, stream windows, regular-rational readback, a Real seal, the Bishop interval compact handoff, transport, replay, provenance, and local naming.

Next read the finite-cover bridge scope of Definition 1428.58. That scope requires a bridge consumer to display the same endpoint, mesh, radius, membership, window, readback, seal, interval-compact, uniform-modulus, transport, replay, provenance, and naming rows before any bridge budget is consumed.

The regular Cauchy bridge window and bridge export children then bind the Real and Completion handoff. Apply Definition 1428.59 to force regular-name reads through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopIntervalCompact}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ rows, and apply Definition 1428.60 to project the same finite route as the exported bridge surface.

Finally compare with the bridge-facing public handoff of Definition 1428.55. It preserves exactly the finite public-cover route and explicitly leaves bridge checking and stronger compactness or completion principles outside the packet. Therefore a mature finite-cover export has to be the displayed route above, with open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient Real completeness, host compactness, and hidden completion data refused. □

Theorem 1428.70 (DyadicIntervalCover RegularCauchy finite-window handoff)**.** *Any* RegularCauchyCompletionU *or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of a mature* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *finite-cover export must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, mesh and radius data, finite cover-membership rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, RegularCauchy window routing, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *finite-cover handoff, the* $\mathsf{UniformModulus}^{\uparrow}$ *lower-bound route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can read the exported comparison. The handoff exports no open-cover compactness theorem, selected subcover, quotient real equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 1428.69. It already forces a mature export through dyadic endpoint rows, mesh and radius data, cover-membership rows, stream windows, regular-rational readback, the Real seal, interval-compact and uniform-modulus handoffs, and the structural rows.

Apply the regular-Cauchy bridge window theorem, Definition 1428.59. This identifies the RegularCauchy-facing read as a finite window route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopIntervalCompact}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, transport, replay, provenance, and local naming.

Finally compose the bridge export and bridge handoff theorems, Definition 1428.60 and Definition 1428.55. They keep the same finite-cover surface public while refusing open-cover compactness, selected subcovers, quotient real equality, countable choice, and ambient Real completeness, so every RegularCauchy or RegSeqRat consumer receives exactly the stated handoff. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalCoverUp.DyadicIntervalCoverRegularCauchyFiniteWindowHandoff`

Chapter 1429

A Concrete Naming Certificate for $\mathsf{DecimalEndpointNormalization}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ packet records the finite convention that relates terminating decimal prefixes to repeating-nine endpoint prefixes. It refines the finite decimal readback surface of chapter 1417 without selecting a host representative for a real number. The packet compares its decimal-window rows with the binary and continued-fraction readback neighbors of chapter 418 and chapter 342, and it hands all real-facing reads through chapter 126, chapter 291, chapter 301, and chapter 38.

Definition 1429.1 (Decimal endpoint normalization carrier)**.** *A* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, E, L, A, W, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite digit-prefix ledger. Its rows list the inspected decimal prefixes, their signs or integral prefixes when present, and the finite place window in which the prefix is read.*
2. *E is the terminating-versus-repeating-nine endpoint ledger. Each row records a displayed finite terminating prefix, the corresponding repeating-nine tail convention visible inside the same finite window, and the endpoint side on which the convention is invoked.*
3. *L is the decimal carry and normalization row. It records the finite carry propagation, the last changed digit, and the normalized prefix produced by that carry.*
4. *A is the base-ten rational place-value row for the prefixes displayed by D, E, L.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to request and replay exactly the inspected prefix and endpoint-convention entries.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback row comparing the base-ten place-value rows with the dyadic tolerance demanded by the consumer.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the same window, place-value, carry, and dyadic-tolerance rows.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, E, L, A, W, Q, R have been displayed.*

9. *H stores componentwise* hsame *transports for digit-prefix, endpoint-ledger, carry-normalization, place-value, stream-window, dyadic-tolerance, regular-readback, seal, continuation, provenance, and local naming rows.*

10. *C stores the displayed* Cont *replay from prefix windows to endpoint rows, from endpoint rows to carry normalization, from normalization to dyadic tolerance, from tolerance to regular readback, and from readback to the real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalEndpointNormalization}^{\uparrow}}$ *row.*

The carrier stores only the finite endpoint-convention certificate just displayed. It has no row for host decimal equality, quotient-real equality, selected representatives, trichotomy, countable choice, $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Definition 1429.2 (Decimal endpoint normalization classifier)**.** *The local classifier for an accepted packet $K = (D, E, L, A, W, Q, R, S, H, C, P, N)$ accepts exactly those finite reads that pass through the displayed rows in the order*

$$D, E, L, A, W, Q, R, S.$$

It treats a terminating endpoint row and its repeating-nine companion as two named endpoint conventions inside the same finite ledger E; it then reads the carry row L to produce the normalized finite prefix inspected by A. Classifier transport is componentwise through H, continuation is replay through C, and package admissibility is the single provenance row P together with the local naming row N.

Theorem 1429.3 (Decimal endpoint normalization NameCert obligations)**.** *For an accepted* DecimalEndpointNorm *carrier $K = (D, E, L, A, W, Q, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalEndpointNormalization}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the digit-prefix, endpoint-ledger, carry-normalization, place-value, stream-window, dyadic-tolerance, regular-readback, real-seal, transport, continuation, provenance, and naming rows;*

2. *classifier transport along the stored* hsame *coordinates and the displayed* Cont *route;*

3. *endpoint-ledger exactness, stating that every terminating-versus-repeating-nine comparison exported by the packet is one of the rows listed in E and normalized through L;*

4. *readback discipline, stating that every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the same finite window before receiving the corresponding tolerance, handoff, or seal row;*

5. *ledger exclusion, stating that host decimal equality, quotient-real equality, selected representatives, trichotomy, countable choice,* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge data are not exported by the packet.*

Lean4 ✓ `BEDC.Derived.DecimalEndpointNormalizationUp.TasteGate.DecimalEndpointNormalizationTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1429.1. The finite digit-prefix rows, endpoint ledger, carry-normalization row, place-value row, stream window, dyadic tolerance, regular

readback, real seal, transports, continuations, package row, and local naming row are precisely the displayed coordinates.

Classifier transport is componentwise. The digit and endpoint rows move through their stored hsame witnesses, while C replays the same window-to-endpoint, endpoint-to-carry, carry-to-tolerance, and tolerance-to-readback route after transport.

Endpoint-ledger exactness is local to E and L. A terminating prefix and a repeating-nine companion are compared only when the finite ledger lists that pair, and the only exported normalization is the carry row displayed for that pair.

The readback discipline then follows the sibling surfaces. Windows are read through chapter 126, dyadic tolerances through chapter 291, regular rational handoff through chapter 301, and the terminal seal through chapter 38.

The excluded principles are absent from the carrier inventory, so the local certificate cannot export them. □

Theorem 1429.4 (Decimal endpoint Real readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalEndpointNormalization}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The decimal packet supplies finite prefix rows, the endpoint-normalization packet supplies the displayed terminating-versus-repeating-nine ledger and carry row, the stream window supplies the inspected observation, the dyadic row supplies tolerance readback, the regular row supplies the rational handoff, and the real-facing consumer receives only the displayed seal. The binary and continued-fraction neighbors from chapter 418 and chapter 342 are comparison readback surfaces, not additional representative selectors.

Proof. Start with the accepted packet $K = (D, E, L, A, W, Q, R, S, H, C, P, N)$. The decimal-facing data are D, E, L, A: the prefix ledger, endpoint-convention ledger, carry-normalization row, and base-ten place-value row.

The stream-facing data are exactly the windows in W. A consumer can inspect only the finite prefix and endpoint rows named by that window, so no infinite decimal stream enters the read.

The dyadic row Q converts the same finite normalized prefix into the tolerance language used by downstream real consumers. The regular handoff row R is reached only after D, E, L, A, W, Q are fixed.

The seal row S is downstream of R, and the rows H, C, P, N preserve the same route under transport, continuation, provenance, and local naming. Thus the real readback has the displayed finite factorization and cannot add a selected representative or quotient-real equality test. □

Theorem 1429.5 (Decimal endpoint nonescape)**.** *Let* $K = (D, E, L, A, W, Q, R, S, H, C, P, N)$ *be an accepted* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *packet. Any consumer that reads an endpoint convention, a normalized decimal prefix, a dyadic tolerance, a regular rational handoff, or a visible* $\mathsf{Real}^{\uparrow}$ *seal is exhausted by the displayed rows of* K*. In particular, no route from* K *escapes to endpoint uniqueness as host equality, quotient reals, selected representatives, trichotomy, countable choice,* $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge.*

Proof. Read the requested datum by cases. Endpoint conventions use E, normalized decimal prefixes use L together with the prefix rows D, dyadic tolerances use Q, regular rational handoffs use R, and visible real seals use S.

The shared observation window is W, and all replay is mediated by C. Hence every case remains inside the finite window and the rows already displayed in the carrier.

Transport and provenance do not add new data. The row H transports equal displayed coordinates, while P records the dependence on the decimal, binary, continued-fraction, stream, dyadic, regular, and real sibling surfaces already cited in the chapter.

The listed escaped objects are not rows of K. Therefore the packet cannot supply them to any consumer of its endpoint-normalization certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalEndpointNormalization}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ packet is generated from finite digit-prefix rows, a terminating-versus-repeating-nine endpoint ledger, a carry-normalization row, StreamName windows, DyadicRatCore tolerance readback, RegSeqRat handoff, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalEndpointNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1429.1, Definition 1429.2, Definition 1429.3, Definition 1429.4, and Definition 1429.5 seed the finite decimal endpoint-convention surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalEndpointNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close endpoint uniqueness as host equality, quotient reals, selected representatives, trichotomy, countable choice, RealUp completeness, or a bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, endpoint-ledger exactness, readback discipline, nonescape, and RealUp seal handoff for the displayed packet.

Chapter 1430

A Concrete Naming Certificate for $\mathsf{BoundedRealFamily}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRealFamily}^{\uparrow}$ packet records a bounded family of real names by finite index-window rows, visible StreamName windows, regular rational readback, RealUp family seals, and a closed-bounded interval-net bound row. It is the family-facing analogue of the bounded sequence surface in chapter 1415, shares the explicit modulus discipline of chapter 1423, and reads cover-facing interval data through chapter 1428. Its bound row is the finite $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ surface of chapter 1250, and each terminal family member is sealed only through the $\mathsf{Real}^{\uparrow}$ route of chapter 38. The packet prepares later least-upper-bound, compactness, and cover-selection certificates without importing trichotomy, quotient equality, countable choice, or a supremum principle.

Definition 1430.1 (Bounded real family carrier)**.** *A* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, W, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-window ledger. Each row names an inspected family index together with the finite observation window requested at that index.*
2. *W is the family of* $\mathsf{StreamName}^{\uparrow}$ *windows displayed for the inspected index-window rows of I.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those StreamName windows and their visible dyadic tolerance observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *family-seal row. A family member is visible to a Real-facing consumer only after its index row, StreamName window, and RegSeqRat readback have been displayed.*
5. *B is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row carrying the located lower and upper interval data used to witness a common bound for the displayed family rows.*
6. *H stores componentwise* hsame *transport for the index-window ledger, StreamName windows, regular readback, Real family seals, bound row, continuation replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from index-window rows to StreamName windows, from StreamName windows to RegSeqRat readback, from RegSeqRat readback to Real family seals, and from Real family seals to the common bound ledger.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$ *row.*

The bound row is displayed as finite located interval data

$$B : \mathsf{ClosedBoundedIntervalNet}^{\uparrow},$$
$$\forall i \in I_{\text{shown}} \quad W_i \longmapsto Q_i \longmapsto R_i \in B.$$

Here I_{shown} denotes only the inspected finite index-window rows carried by the packet. The classifier compares accepted packets only through $I, W, Q, R, B, H, C, P, N$ and has no coordinate for trichotomy, quotient equality of real names, countable choice over indices, an ambient supremum, open-cover compactness, or host completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyTasteGate single carrier alignment`

Theorem 1430.2 (Bounded real family located upper row)**.** *For every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier $F = (I, W, Q, R, B, H, C, P, N)$, the bound row B is read through finite located lower and upper interval data. Each displayed family member reaches the common bound by the route*

$$I \longmapsto W \longmapsto Q \longmapsto R \longmapsto B$$

together with H, C, P, N. Thus the located upper row is a finite certificate row and imports no trichotomy, quotient equality of real names, countable choice, supremum principle, or host completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyLocatedUpperRow`

Proof. Project the carrier of Definition 1449.1. The only family data exposed to the classifier are the finite index-window ledger I, the displayed $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the terminal $\mathsf{Real}^{\uparrow}$ family seals R.

Read the bound coordinate next. The row B is a $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet, so its lower and upper interval data are displayed finite rows from chapter 1250. A family member can meet B only after its own window, readback, and seal route has already been replayed.

The structural rows preserve this same route. Componentwise transport H, continuation replay C, package provenance P, and the local name N have no field for trichotomy, quotient equality, countable choice, a supremum, or host completeness. Hence the located upper row is read as finite interval data and nothing stronger. □

Theorem 1430.3 (Bounded real family RegSeqRat scope)**.** *Every Real-facing read of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow}$$

as instantiated by the rows W, Q, R, B and transported by H, C, P, N. No consumer can read a family member, a bound assertion, or a cover-facing handoff outside the StreamName windows, RegSeqRat readback, RealUp family seals, and bound ledger carried by the packet.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyRegSeqRatScope`

Proof. Begin with a Real-facing request for an inspected family row. The carrier first exposes the index-window ledger I and the corresponding $\mathsf{StreamName}^{\uparrow}$ window in W.

The request then passes through the regular rational readback. The row Q is the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate, and the terminal $\mathsf{Real}^{\uparrow}$ seal R is reached only after Q has been displayed.

Finally the bound assertion is read through B. Since B is the only bound ledger and since H, C, P, N only transport, replay, source, and name the displayed rows, the request cannot leave the StreamName, RegSeqRat, RealUp, and bound-ledger scope of the packet. □

Theorem 1430.4 (Bounded real family bound-ledger nonescape)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet reads the common bound only through the displayed rows* $I, W, Q, R, B, H, C, P, N$*. In particular, a consumer cannot export trichotomy, quotient real equality, countable choice over indices, a supremum principle, open-cover compactness, or host completeness from the bound ledger.*

Proof. Project the carrier of Definition 1449.1. The common bound is the single row B, and it is available only after the finite index-window ledger I, StreamName windows W, RegSeqRat readback Q, and Real family seals R have been displayed.

Apply Definition 1449.2. It identifies the bound row as finite located lower and upper interval data rather than as an ambient order-completeness witness or a selected supremum.

Apply Definition 1449.3. It forces every Real-facing read of a family member or bound assertion through the StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route carried by W, Q, R, B.

It remains to inspect the structural rows. The transport row H preserves only componentwise equality of the displayed rows, the replay row C replays only the displayed route, the provenance row P names only the sibling sources, and the local row N names the packet. None of H, C, P, N contains a branch for trichotomy, quotient real equality, countable choice over indices, supremum extraction, open-cover compactness, or host completeness. Thus the common-bound read cannot escape the finite bound ledger. □

Theorem 1430.5 (Bounded real family obligation surface)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier exposes exactly the finite index-window ledger* I*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{Real}^{\uparrow}$ *family seals* R*,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$ *row* N *before any compactness, least-upper-bound, or cover-selection consumer may read it.*

Proof. Project the carrier of Definition 1449.1. Its displayed coordinates are exactly $I, W, Q, R, B, H, C, P, N$, so an obligation-level consumer has no other entry point.

Apply Definition 1449.2 to the bound coordinate. The common bound is finite located interval-net data, not a supremum, cover selector, or host completeness witness.

Apply Definition 1449.3 and Definition 1449.4. They force every Real-facing read through the StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route and bar trichotomy, quotient equality, countable choice, compactness extraction, least-upper-bound extraction, and cover-selection reads. Hence the listed rows are exactly the obligation surface available before downstream consumers add their own data. □

Theorem 1430.6 (BoundedRealFamily scoped dependency route)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet exposes its finite index-window ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *family-seal row,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before any compactness, least-upper-bound, or cover-selection consumer may read it. The route exports no trichotomy, quotient equality of real names, countable choice, supremum principle, open-cover compactness, or host completeness.*

Proof. Project the carrier in Definition 1449.1. Its finite dependency route is exactly $I, W, Q, R, B, H, C, P, N$.

The analytic rows are already ordered. Apply Definition 1449.2 for the located upper row, Definition 1449.3 for the StreamName and RegSeqRat scope, and Definition 1449.4 for the bound-ledger refusal.

Now apply Definition 1449.5. It exhausts the rows visible before a downstream compactness, least-upper-bound, or cover-selection consumer adds its own data.

Since the packet has no coordinate beyond those rows, it exports no trichotomy, quotient equality, countable choice, supremum, open-cover compactness, or host-completeness principle. □

Theorem 1430.7 (Bounded real family compactness consumer boundary)**.** *Every compactness consumer of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier* $F = (I, W, Q, R, B, H, C, P, N)$ *must add its own finite-net, total-boundedness, located-compactness, or compactness row before reading a compactness conclusion. The rows* $I, W, Q, R, B, H, C, P, N$ *expose the bounded-family obligation surface and common bound ledger, but they do not by themselves supply an open-cover compactness theorem, selected finite subcover, compact metric witness, or compactness classifier.*

Proof. Start from the carrier in Definition 1449.1. Its coordinates list finite index windows, StreamName windows, RegSeqRat readback, Real seals, one ClosedBoundedIntervalNet bound row, transport, replay, provenance, and the local NameCert row.

Apply Definition 1449.2 and Definition 1449.3. These route every Real-facing read through the displayed windows, readback, seals, and bound row, so a compactness consumer can only see the bounded-family route already named by $I, W, Q, R, B, H, C, P, N$.

Finally apply Definition 1449.4 and Definition 1449.5. The bound ledger refuses compactness extraction, and the obligation surface is exhausted before any downstream compactness consumer reads the packet. Therefore compactness requires an explicit downstream finite-net or compactness row. □

Theorem 1430.8 (Bounded real family least-upper-bound refusal)**.** *For every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier* $F = (I, W, Q, R, B, H, C, P, N)$*, the common bound row* B *is a* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row and not a supremum, least-upper-bound, order-completeness, or trichotomy witness. A least-upper-bound consumer must therefore provide its own supremum or completeness row outside the bounded-family packet.*

Proof. Read the carrier through Definition 1449.1. The bound coordinate is named as $B : \mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, attached downstream of the finite index-window, StreamName, RegSeqRat, and Real-family rows.

Apply Definition 1449.2. It identifies B as finite located lower and upper interval data, so the row witnesses a displayed common bound rather than a selected least upper bound.

Apply Definition 1449.4 and Definition 1449.5. These theorems remove trichotomy, quotient equality of real names, countable choice, host completeness, and supremum extraction from the rows available to a consumer. Thus a least-upper-bound read must come from a separate downstream row. □

Theorem 1430.9 (Bounded real family cover-selection boundary)**.** *Every cover-selection consumer of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier must pass through an explicit finite-cover, finite-net, or selected-cover sibling row before exporting a cover conclusion. The bounded-family packet supplies only* $I, W, Q, R, B, H, C, P, N$ *and cannot select a finite subcover from those rows alone.*

Proof. Project the carrier in Definition 1449.1. The only cover-facing data present before downstream consumption are the family windows, rational readback, Real seals, the ClosedBoundedIntervalNet bound row, and the structural rows H, C, P, N.

Use Definition 1449.3. It forces a cover-facing read of a family member or bound assertion to pass through the same StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route.

Now apply Definition 1449.4 and Definition 1449.5. They exclude open-cover compactness, cover selection, and hidden choice from the displayed bounded-family rows. Hence a cover-selection consumer must provide a separate finite-cover or finite-net sibling row before any cover conclusion is exported. □

Falsifiable prediction: Every $\mathsf{BoundedRealFamily}^{\uparrow}$ Real-facing read factors through finite index windows, StreamName windows, RegSeqRat readback, RealUp family seals, and a ClosedBoundedIntervalNet bound row.

Independence witness: boundedrealsequence, realcauchymodulus, dyadicintervalcover, closedboundedintervalnet, real: $\mathsf{BoundedRealFamily}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an index-window family ledger, StreamName windows, RegSeqRat readback, RealUp family seals, a common bound ledger, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealFamily}^{\uparrow}$ packet is generated from finite index-window rows, StreamName windows, RegSeqRat readback, RealUp family seals, a ClosedBoundedIntervalNet bound row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1449.1, Definition 1449.2, Definition 1449.3, Definition 1449.4, Definition 1449.5, Definition 1449.6, Definition 1449.7, Definition 1449.8, and Definition 1449.9 bind the finite bounded-family obligation surface, scoped dependency route, and public compactness, least-upper-bound, and cover-selection consumer boundaries over StreamNameUp, RegSeqRatUp, RealUp, and ClosedBoundedIntervalNetUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, trichotomy, quotient equality of real names, countable choice, an ambient supremum principle, open-cover compactness, or host completeness.

Upgrade path: Bridge closure requires the standard bridge row needed by downstream compactness, least-upper-bound, and cover-selection routes.

Chapter 1431

A Concrete Naming Certificate for $\mathsf{SternBrocotApproximation}^{\uparrow}$

Following Stern–Brocot and Farey rational approximation, the $\mathsf{SternBrocotApproximation}^{\uparrow}$ packet records finite rational bracketing as displayed BEDC data. It reads a pair of rational interval endpoints through chapter 354, a finite mediant path through dyadic tolerance bookkeeping from chapter 291, a $\mathsf{StreamName}^{\uparrow}$ observation window from chapter 126, a $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, and a terminal $\mathsf{Real}^{\uparrow}$ seal from chapter 38. The packet is a finite bracketing neighbor of the decimal and continued-fraction readback surfaces in chapter 1417 and chapter 342; it does not store an ambient real order or a quotient-real representative.

Definition 1431.1 (Stern–Brocot approximation carrier)**.** *A* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational endpoint row, carrying the left side of the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *bracket.*
2. *U is the upper rational endpoint row, carrying the right side of the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *bracket.*
3. *M is the finite mediant-path ledger, whose rows record the left/right Stern–Brocot choices and the rational mediants actually inspected.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the dyadic precision requested by the consumer and the displayed bound showing that the bracket width is below that tolerance.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window used to request and replay the inspected rational rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same endpoint and mediant data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the rational interval, tolerance, stream-window, and regular-rational readback rows have been displayed.*
8. *H stores componentwise* hsame *transports for endpoint, mediant-path, tolerance, stream-window, readback, and seal rows.*
9. *C stores the* Cont *replay from stream windows to mediant inspection, from mediant inspection to rational bracketing, from bracketing to regular readback, and from readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$ *row.*

The carrier stores only a finite rational bracketing certificate. It has no coordinate for Dedekind cuts, trichotomy, quotient reals, selected representatives, or RealUp completeness.

Lean4 ✓ `BEDC.Derived.SternBrocotApproximationUp.SternBrocotApproximationTasteGate single carrier alignment`

Theorem 1431.2 (Stern–Brocot NameCert obligations)**.** *For an accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *carrier* $K = (L, U, M, T, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower endpoint, upper endpoint, mediant-path, dyadic tolerance, stream-window, regular-rational readback, real-seal, transport, continuation, provenance, and naming rows;*
2. *classifier transport along the stored* hsame *rows and displayed* Cont *routes;*
3. *bracketing exactness, stating that every exported rational bracket is one of the endpoint or mediant rows displayed by L, U, M and checked against T;*
4. *readback discipline, stating that every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads through the finite window W and the regular readback row R before receiving the seal row E;*
5. *ledger exclusion, stating that Dedekind cuts, trichotomy, quotient reals, selected representatives, and RealUp completeness are not exported by the packet.*

Proof. Carrier admission is projection from the tuple in Definition 1431.1. The rational endpoints, mediant path, dyadic tolerance, stream window, regular readback, real seal, transports, continuations, package row, and local naming row are precisely the displayed coordinates.

Classifier transport is componentwise. The endpoint and mediant rows move through their stored hsame witnesses, while C replays the same window-to-mediant and bracket-to-readback route after transport.

Bracketing exactness uses the sibling interval and dyadic arithmetic surfaces. The endpoint rows are the displayed $\mathsf{RationalInterval}^{\uparrow}$ bounds from chapter 354, and the tolerance row is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ comparison row from chapter 291.

The readback discipline then follows the stored route through chapter 126, chapter 301, and chapter 38. The excluded principles are absent from the carrier inventory, so the local certificate cannot export them. □

Theorem 1431.3 (Finite rational bracketing)**.** *Every accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *packet exposes a finite rational bracket*

$$L \leq M_i \leq U$$

at each inspected mediant row M_i of the finite ledger M, and the final displayed bracket has width bounded by the dyadic tolerance row T. Every route to the bracket factors through

$$\mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow}.$$

The packet supplies no host trichotomy, no quotient-real equality test, and no choice of a distinguished representative for the sealed real.

Proof. Read the mediant ledger M in order. Each row records a finite left/right choice and the rational mediant computed from the currently displayed endpoints, so the next bracket is formed by replacing exactly one endpoint with that mediant.

The interval side is supplied by chapter 354. It records the endpoint rows and the local comparison data needed to keep the displayed lower endpoint below the displayed upper endpoint at the inspected scale.

The tolerance side is supplied by chapter 291. The row T records the dyadic precision and the displayed comparison showing that the final interval width is below that precision.

The window side is supplied by chapter 126. Since the carrier has no other observation coordinate, every bracket read factors through the displayed finite window and the rational rows it exposes. □

Theorem 1431.4 (Stern–Brocot Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *packet factors through the finite handoff*

$$\mathsf{SternBrocotApproximation}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The Stern–Brocot carrier provides the finite rational bracket; the decimal and continued-fraction siblings of chapter 1417 and chapter 342 provide comparison readback neighbors; and the real-facing consumer receives only the displayed regular rational row R and seal row E.

Proof. Start with the accepted packet $K = (L, U, M, T, W, R, E, H, C, P, N)$. The carrier exposes the bracket through L, U, M, T and exposes the observation request through W.

The regular-rational handoff is the row R. By the readback discipline of chapter 301, a real-facing consumer reads the finite rational data through the regular sequence surface before reading a real seal.

The seal row E is therefore downstream of R and the displayed window route. The transports H, continuations C, and provenance row P preserve the same route for classified packets and for repeated finite-window reads.

The comparison with decimal and continued-fraction siblings is only a readback comparison: all three chapters present finite rational observation surfaces before the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. None supplies Dedekind cuts, trichotomy, quotient reals, selected representatives, or RealUp completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SternBrocotApproximation}^{\uparrow}$ packet is generated from rational interval endpoints, a finite mediant path, a dyadic tolerance row, a finite StreamName window, a RegSeqRat readback, a RealUp seal, componentwise transports, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1431.1, Definition 1431.2, Definition 1431.3, and Definition 1431.4 seed the finite Stern–Brocot rational-bracketing surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SternBrocotApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, trichotomy, quotient reals, selected representatives, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, bracketing exactness, readback discipline, ledger exclusion, and RealUp seal handoff for the same finite packet.

Chapter 1432

Rational interval refinement NameCert construction

Following Moore–Kearfott–Cloud interval analysis and Bishop–Bridges constructive analysis, the $\mathsf{RationalIntervalRefinement}^{\uparrow}$ packet records finite rational-interval refinement as displayed endpoint, containment, and retained-window data. It sits between the rational interval surface of chapter 354, the dyadic interval surface of chapter 306, the nested-window surface of chapter 320, the branch route of chapter 361, and the subdivision ledger of chapter 366. The packet is a finite consumer primitive for real-completion arguments, not an infinite interval tree, a host subset of $\mathsf{Real}^{\uparrow}$, or a selected completed point.

Definition 1432.1 (RationalIntervalRefinement carrier)**.** *A* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, J, E, W, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the parent* $\mathsf{RationalInterval}^{\uparrow}$ *row whose endpoints and containment ledger are read from chapter 354;*
2. *J is the child* $\mathsf{RationalInterval}^{\uparrow}$ *row with displayed endpoint rows inside the parent window;*
3. *E is the endpoint-inclusion ledger certifying that the lower and upper endpoints of J are contained in the ordered endpoint surface of I;*
4. *W is the retained-window row, readable either as a* $\mathsf{NestedInterval}^{\uparrow}$ *step or as the finite rational window handed to a real-completion consumer;*
5. *K is the optional dyadic support ledger exposing the* $\mathsf{DyadicInterval}^{\uparrow}$ *boundary, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch row, or the* $\mathsf{DyadicSubdivision}^{\uparrow}$ *cell row used to produce this refinement when such a route is present;*
6. *H records componentwise* hsame *transport for the parent interval, child interval, endpoint-inclusion ledger, retained window, dyadic support rows, continuation rows, provenance, and local naming row;*
7. *C records the displayed* Cont *replay routes by which consumers reread I, J, E, W, K, H, P, N without changing the retained window;*
8. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalRefinement}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{DyadicSubdivision}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ *row.*
The carrier has no coordinate for countable choice, an infinite refinement branch, a completed real point, a host interval subset, or a standard bridge.

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementTasteGate single carrier alignment`

Theorem 1432.2 (RationalIntervalRefinement nested window)**.** *Every accepted* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *packet $R = (I, J, E, W, K, H, C, P, N)$ exposes one finite nested-window step: the child rational interval J is retained inside the parent rational interval I by the endpoint-inclusion ledger E, and the consumer-facing retained row is exactly W. If the support ledger K comes from a dyadic bisection or dyadic subdivision route, that route is read only as finite evidence for the same retained window.*

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementCarrier nested window`

Proof. Project the carrier of Definition 1432.1. The only parent interval coordinate is I, the only child interval coordinate is J, and the only endpoint-inclusion coordinate is E.

The retained-window row is W. It is stored beside I, J, E, so a consumer cannot read the retained window without also reading the parent, child, and endpoint-inclusion data that make the containment finite.

If K is present, it contributes only the dyadic route used to exhibit the same refinement. The transport rows in H and replay routes in C move those displayed rows componentwise, while P and N record provenance and local naming. No coordinate remains for an infinite branch or a completed real point. □

Theorem 1432.3 (RationalIntervalRefinement retained-window induction)**.** *Let a finite* $\mathsf{Nat}^{\uparrow}$*-indexed unary history ledger display accepted refinement packets*

$$R_i = (I_i, J_i, E_i, W_i, K_i, H_i, C_i, P_i, N_i)$$

in a chain where each child interval J_i is the parent interval I_{i+1} read by the next packet. Then the chain preserves a retained-window row from the first parent interval to the final child interval. The final child is read only through the successive endpoint-inclusion ledgers E_i, optional dyadic support rows K_i, componentwise hsame *transports H_i, displayed* Cont *replays C_i,* Pkg *provenance rows P_i, and local* $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ *rows N_i.*

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementRetainedWindowInduction`

Proof. Induct on the displayed finite $\mathsf{Nat}^{\uparrow}$ ledger. In the base case, the chain has one accepted packet, so Definition 1432.2 supplies the retained row W_0 from I_0 to J_0.

For the step, assume the retained-window row has been built through the current final child. The next accepted packet reads that child as its parent, supplies the next endpoint-inclusion row, and exposes its retained row by Definition 1432.2.

Compose the two retained reads by repacking the same finite coordinates. The endpoint ledgers provide the parent-child inclusions, any dyadic rows remain support for those inclusions, and H, C, P, N transport, replay, source, and name the same chain. No induction step introduces an infinite branch, a host interval subset, or a completed real point. □

Theorem 1432.4 (RationalIntervalRefinement chain exactness)**.** *Let a finite* $\mathsf{Nat}^{\uparrow}$*-indexed ledger display accepted* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *packets*

$$R_i = (I_i, J_i, E_i, W_i, K_i, H_i, C_i, P_i, N_i)$$

with J_i *read as the next parent interval* I_{i+1} *whenever the next packet is present. The public chain surface exposes exactly the initial parent interval* I_0*, the final child interval, the endpoint-inclusion ledgers* E_i*, retained-window rows* W_i*, optional dyadic support rows* K_i*, componentwise transports* H_i*, replay rows* C_i*, provenance rows* P_i*, and local* $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ *rows* N_i*. It exposes no infinite branch, completed real-point selection, host interval subset, or countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementChainExactness`

Proof. Induct on the finite $\mathsf{Nat}^{\uparrow}$ ledger. In the base case, the chain has one accepted packet, and Definition 1432.3 identifies the retained-window row from the parent interval to the displayed child.

For the step, apply the induction hypothesis to the prefix chain. The next accepted packet reads the previous final child as its parent, adds exactly one endpoint-inclusion ledger, one retained-window row, optional dyadic support, transport, replay, provenance, and local naming row.

Repack the prefix surface with the final packet surface using Definition 1432.3. The resulting chain records the first parent, the last child, and precisely the displayed rows from each finite packet.

No step introduces a coordinate for an infinite refinement branch, a completed real point, a host interval subset, or countable choice. Therefore the retained-window chain is exact as a finite BEDC ledger. □

Theorem 1432.5 (RationalIntervalRefinement NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ *surface for Definition 1432.1 consists exactly of carrier exposure for* R*, classifier exposure for the parent-child containment pair* I, J, E*, retained-window exactness through* W*, optional dyadic route exposure through* K*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier exposure is projection from the tuple $R = (I, J, E, W, K, H, C, P, N)$. These are precisely the rows that the local naming surface can expose.

The classifier part is finite because the parent-child containment read uses the displayed rational endpoint and containment ledgers in I, J, E. The retained-window obligation is then exactly the row W, with no hidden intersection or completion object.

The support ledger K does not change the carrier. It only records which dyadic interval, bisection, or subdivision rows were used to exhibit the same rational refinement when a dyadic route is available.

Finally H and C transport and replay the same displayed rows, while P and N bind provenance and local naming. Since the carrier has no other coordinate, the listed rows exhaust the NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementNameCertObligations`

Theorem 1432.6 (RationalIntervalRefinement classifier stability)**.** *Let* $R = (I, J, E, W, K, H, C, P, N)$ *be an accepted* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *packet. Any second displayed packet*

$$R' = (I', J', E', W', K', H', C', P', N')$$

whose parent interval, child interval, endpoint-inclusion ledger, retained-window row, optional dyadic support row, componentwise transport row, continuation replay row, provenance row, and local naming row agree componentwise with the corresponding rows of R is accepted by the same refinement classifier. The agreement is read only through the finite hsame *rows in H, H' and the displayed* Cont *replays in C, C'.*

Proof. Project both packets through Definition 1432.1. The carrier lists exactly $I, J, E, W, K, H, C, P, N$, so a classifier read has no row outside the two rational intervals, the endpoint-inclusion ledger, the retained window, the optional dyadic support, transport, replay, provenance, and local naming.

Use Definition 1432.2. The accepted refinement classifier is determined by the retained child-in-parent read supplied by I, J, E, W, with K only recording a finite dyadic route when such support is present.

The componentwise hsame rows and displayed Cont replays preserve those same coordinates. Since P and N source and name no additional coordinate, replacing the packet by R' with the same displayed rows preserves acceptedness. □

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementCarrier classifier stability`

Theorem 1432.7 (RationalIntervalRefinement endpoint exactness)**.** *Every public containment read from an accepted* RationalIntervalRefinement$^{\uparrow}$ *packet $R = (I, J, E, W, K, H, C, P, N)$ factors through the displayed endpoint-inclusion ledger E and the retained-window row W. The read may use I and J to expose the parent and child rational endpoints, and it may use K, H, C, P, N to support, transport, replay, source, and name the same containment, but it cannot replace E or W by a host interval subset, an infinite branch, a completed real point, or a standard bridge.*

Proof. Begin with the carrier of Definition 1432.1. The only endpoint-inclusion coordinate is E, and the only consumer-facing retained-window coordinate is W.

Apply Definition 1432.2. It identifies the finite retained-window step with the child rational interval J retained inside the parent rational interval I by E, and it identifies W as the retained row delivered to consumers.

The remaining rows do not create a second containment source. The optional support row K only records a dyadic route for the same refinement, while H, C, P, N transport, replay, source, and name the displayed rows. Thus every public containment read factors through E and W. □

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementCarrier endpoint exactness`

Theorem 1432.8 (RationalIntervalRefinement ledger exhaustion)**.** *An accepted* RationalIntervalRefinement$^{\uparrow}$ *packet exposes to public consumers only the rows $I, J, E, W, K, H, C, P, N$ of Definition 1432.1. In particular, a consumer receives no infinite refinement branch, completed real point, host interval subset, quotient, countable-choice schedule, or standard bridge from the packet.*

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementLedgerExhaustion`

Proof. Project the accepted packet as $R = (I, J, E, W, K, H, C, P, N)$. These nine rows are the whole displayed carrier, so the public ledger begins and ends with the parent interval, child interval, endpoint-inclusion ledger, retained window, optional dyadic support, transport, replay, provenance, and local naming.

Use Definition 1432.5. The local $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ surface exposes carrier data, classifier data, retained-window exactness, optional support, componentwise hsame, displayed Cont, Pkg provenance, and local naming, and no other obligation row.

The excluded objects would require an additional coordinate: an infinite branch, completed real point, host subset, quotient, choice schedule, or bridge row. Since neither the carrier nor the NameCert obligation surface contains such a coordinate, public consumers receive none of them. □

Theorem 1432.9 (RationalIntervalRefinement finite branch exhaustion)**.** *For any finite chain of accepted* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *packets, the public branch surface preserves only the displayed parent-child endpoint ledgers, retained-window rows, optional dyadic support rows, componentwise* hsame *transports, displayed* Cont *replays,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$ *rows. The chain cannot export an infinite branch, a completed* $\mathsf{Real}^{\uparrow}$ *point, a host interval subset, a quotient, a countable-choice schedule, or a standard bridge.*

Proof. Begin with the retained-window induction of Definition 1432.3. It identifies the finite chain by repeatedly reading the next parent interval as the previous child interval and by carrying forward the displayed retained-window rows.

Endpoint exactness from Definition 1432.7 forces every containment read at each step through the endpoint-inclusion ledger and the retained-window row. Optional dyadic support can witness the same containment, but it does not create another branch coordinate.

Finally apply ledger exhaustion from Definition 1432.8 to each accepted packet in the finite chain. Repacking the listed rows preserves exactly the parent-child endpoint ledgers, retained-window rows, optional dyadic support, transport, replay, provenance, and local naming. The excluded infinite branch, completed $\mathsf{Real}^{\uparrow}$ point, host subset, quotient, countable-choice schedule, and standard bridge have no coordinate in any packet and therefore cannot be exported by the chain. □

Lean4 ✓ `BEDC.Derived.RationalIntervalRefinementUp.RationalIntervalRefinementWindowBranchExhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalRefinement}^{\uparrow}$ packet is generated from a parent $\mathsf{RationalInterval}^{\uparrow}$ row, a child $\mathsf{RationalInterval}^{\uparrow}$ row, endpoint-inclusion evidence, a retained-window row, optional dyadic route support, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1432.7, Definition 1432.8, and Definition 1432.9 supply the public containment export by exposing endpoint exactness, retained-window ledger exhaustion, and finite branch exhaustion for every displayed refinement read.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalIntervalRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite refinement trees, completed real point selection, host interval subsets, countable choice, standard bridges, or a Lean theorem.

Upgrade path: A formal route can encode the carrier and classifier and prove the nested-window and NameCert obligation surfaces over the displayed finite history rows.

Chapter 1433

A Concrete Naming Certificate for $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ packet records the explicit finite-net compactness certificate for displayed located closed intervals. It consumes the endpoint-cell surface of chapter 394, the mesh surface of chapter 1250, the finite-net interface of chapter 127, the complete-limit interface of chapter 128, the finite-window rows of chapter 126, the regular rational rows of chapter 301, and the terminal real seal of chapter 38. The packet is an explicit finite-net compactness certificate for displayed closed intervals; it is not general Heine–Borel compactness, not open-cover compactness, not trichotomy, not quotient real equality, and not a selected subcover choice.

Definition 1433.1 (ClosedIntervalCompactNet carrier)**.** *A* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, M, T, B, S, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-cell row naming the displayed closed interval and its lower and upper located endpoints;*
2. *M is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *mesh row over I, including its dyadic cell ledger, center rows, and finite coverage ledger;*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net handoff row obtained from the same mesh and read at the requested positive tolerance;*
4. *B is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row used only to keep regular Cauchy endpoint and center reads inside the displayed metric-completion ledger;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the endpoint, center, and cell observations used by the mesh;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite-window observations;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached through I, M, S, R and through no detached real equality coordinate;*
8. *L is the local compact-net ledger stating that every accepted interval read is covered by one of the displayed mesh cells and its bundled center;*

9. *H records componentwise* hsame *transport for interval endpoints, mesh rows, total-bounded handoff rows, completion-boundary rows, windows, readbacks, real seals, and the compact-net ledger;*
10. *C records displayed* Cont *routes by which consumers inspect the interval, mesh, finite net, completion boundary, windows, readbacks, real seal, and compact-net ledger;*
11. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactNet}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite-net compactness route for the located closed interval. It has no coordinate for an arbitrary open family, open-cover compactness, selected finite subcover, trichotomy on real names, quotient real equality, host endpoint choice, or ambient Heine–Borel theorem.

Lean4 ✓ `BEDC.Derived.ClosedIntervalCompactNetUp.ClosedIntervalCompactNetTasteGate single carrier alignment`

Theorem 1433.2 (ClosedIntervalCompactNet NameCert obligations)**.** *For an accepted* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *carrier $K = (I, M, T, B, S, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactNet}^{\uparrow}}$ *surface consists exactly of carrier admission for those twelve rows, mesh-to-net coverage through M, T, L, completion-bounded real-name readback through B, S, R, E, componentwise classifier transport through H, consumer replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is the tuple of Definition 1433.1. The interval row, mesh row, finite-net handoff, completion boundary, finite windows, readbacks, real seal, compact-net ledger, transports, continuations, package provenance, and local naming row are all explicit BHist coordinates.

The coverage side is read through M, T, L. The closed bounded interval net supplies the dyadic cells and centers, the totally bounded row packages those centers as the finite net at the requested tolerance, and the local ledger records that interval reads enter one of those displayed cells.

The real-name side is read through B, S, R, E. The complete-metric boundary is only a ledger for regular Cauchy reads of the displayed endpoint and center names; the terminal $\mathsf{Real}^{\uparrow}$ seal is reached after the finite windows and regular rational readbacks have already been exposed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for open covers, selected subcovers, trichotomy, quotient real equality, host endpoint choice, or a general Heine–Borel theorem, none of those objects can appear as local NameCert obligations. □

Theorem 1433.3 (ClosedIntervalCompactNet finite-subcover route)**.** *Every compactness consumer of an accepted* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *packet receives the displayed finite route*

$$I, M, T, L, H, C, P, N.$$

The route may be read as a finite cover by mesh cells and bundled centers for the displayed interval, but it is already finite data and is not selected from an arbitrary open cover.

Proof. Start with the interval row I. The only coverage information attached to it in the carrier is the closed bounded interval net M and the compact-net ledger L.

Project the total-bounded handoff T. It repacks the mesh cells and centers of M as the finite-net row consumed by compactness-facing clients, with the requested tolerance still visible in the displayed ledger.

The rows H, C, P, N then transport, replay, source, and name the same finite route. Because the carrier contains no arbitrary open family and no selected subcover coordinate, a consumer cannot reinterpret this finite net as a choice from an open cover. □

Theorem 1433.4 (ClosedIntervalCompactNet totally bounded handoff)**.** *For every accepted* ClosedIntervalCompac *carrier $K = (I, M, T, B, S, R, E, L, H, C, P, N)$, the row T is the only* $\mathsf{TotallyBounded}^{\uparrow}$ *handoff exported by the packet. It is determined by the located interval row I, the closed bounded interval net row M, the compact-net ledger L, and the structural rows H, C, P, N.*

Proof. Project the carrier and isolate the total-bounded coordinate. The packet contains a single row T for the finite-net handoff, and it appears downstream of the interval row I and mesh row M.

The compact-net ledger L fixes the coverage read used by T. Thus the totally bounded export is the mesh-derived finite-net row, not a separate ambient total-boundedness principle.

Transport and replay are supplied only by H and C, with provenance and naming supplied by P and N. No additional $\mathsf{TotallyBounded}^{\uparrow}$ coordinate or hidden center selector is present. □

Theorem 1433.5 (ClosedIntervalCompactNet complete metric handoff)**.** *Every completion-facing or real-seal-facing consumer of an accepted* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *packet factors through*

$$B, S, R, E, H, C, P, N$$

after the interval and mesh rows I, M have been displayed. The handoff uses $\mathsf{CompleteMetric}^{\uparrow}$ *only as the visible regular Cauchy readback boundary for endpoint and center observations; it does not assert ambient real completeness, quotient real equality, or a selected real endpoint.*

Proof. Begin with the displayed interval and mesh. Endpoint and center observations are already fixed by I and M before any completion-facing read is available.

Read the finite-window and regular-rational rows next. The rows S and R expose the endpoint and center names through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ data, and the complete-metric boundary row B keeps those reads inside the displayed regular Cauchy ledger.

Finally the real seal E and structural rows H, C, P, N carry that same route to consumers. Since the carrier has no quotient real equality coordinate, selected endpoint coordinate, or ambient completeness coordinate, the completion-facing handoff cannot export those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactNet}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ packet is generated from a displayed LocatedIntervalUp row, a ClosedBoundedIntervalNetUp mesh row, a TotallyBoundedUp finite-net handoff, a CompleteMetricUp regular-Cauchy boundary row, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal, a local compact-net ledger, componentwise hsame transport, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactNet}^{\uparrow}}$, seed)

Scope closed: Definition 1433.1, Definition 1433.2, Definition 1433.3, Definition 1433.4, and Definition 1433.5 bind the explicit finite-net compactness certificate for displayed located closed intervals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedIntervalCompactNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness, open-cover compactness, trichotomy, quotient real equality, selected subcover choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, TotallyBoundedUp handoff, CompleteMetricUp handoff, and real-seal non-escape for BEDC.Derived.ClosedIntervalCompactNetUp.

Chapter 1434

A Concrete Naming Certificate for $\mathsf{MonotoneRealSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneRealSequence}^{\uparrow}$ packet records the finite source certificate for a bounded monotone real sequence before the monotone convergence route consumes it. It sits after the finite-window source of chapter 1415, the $\mathsf{StreamName}^{\uparrow}$ interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It is the human-origin bridge from bounded real sequence inspection to the convergence consumer of chapter 1435; it does not assert a selected limit, ambient completeness, quotient equality of streams, trichotomy, or open-cover compactness.

Definition 1434.1 (Monotone real sequence carrier)**.** *A* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, Q, R, B, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the inspected index schedule, listing the finite source windows on which the sequence is read.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows exposed at the indices in S.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and their dyadic tolerance observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream-window and regular-rational rows have been displayed.*
5. *B is the boundedness ledger inherited from the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source packet.*
6. *L is the monotonicity ledger, recording the displayed order direction and the finite adjacent or cofinal comparison rows visible in the inspected schedule.*
7. *H stores componentwise* hsame *transport for the index schedule, stream windows, readbacks, real seals, boundedness ledger, monotonicity ledger, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from inspected indices to stream windows, from stream windows to regular readback, from regular readback to the real seal, and from the joint boundedness and monotonicity ledgers to downstream convergence consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{MonotoneRealSequence}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MonotoneRealSequence}^{\uparrow}}$ *row.*
The carrier classifier compares accepted packets only through

$$S, W, Q, R, B, L, H, C, P, N.$$

It has no coordinate for host real order completeness, a least-upper-bound operator, quotient equality of stream names, a selected limit representative, open-cover compactness, or a Bolzano–Weierstrass subsequence selector.

Theorem 1434.2 (Monotone real sequence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *carrier* $M = (S, W, Q, R, B, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MonotoneRealSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, finite source-window coverage through* S *and* W*, regular rational readback through* Q*, real-name sealing through* R*, boundedness transport through* B*, monotonicity transport through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 1434.1. The only sequence observations are the finite schedule S, the $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the $\mathsf{Real}^{\uparrow}$ seal row R.

Read the two source ledgers next. The boundedness row B is inherited from the $\mathsf{BoundedRealSequence}^{\uparrow}$ source surface of chapter 1415, while the monotonicity ledger L records only displayed comparison rows on the same inspected windows.

Finally inspect the structural rows. Transport, continuation replay, provenance, and local naming preserve the same finite packet and name no further analytic principle. Therefore the listed rows exhaust the local certificate surface. □

Theorem 1434.3 (Monotone real sequence window transport)**.** *Let* $M = (S, W, Q, R, B, L, H, C, P, N)$ *and* $M' = (S', W', Q', R', B', L', H', C', P', N')$ *be accepted* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *packets. If the index schedules, stream windows, regular readbacks, real seals, boundedness ledgers, monotonicity ledgers, continuation rows, provenance rows, and local naming rows are replaced only by rows classified through the displayed* hsame *and* Cont *routes, then the finite monotone source surface transports unchanged: every accepted read of* M' *still factors through*

$$S' \longmapsto W' \longmapsto Q' \longmapsto R' \longmapsto B', L'$$

together with H', C', P', N'*.*

Proof. Begin with the source-window path in M. The carrier exposes an inspected index only by replaying it through W, then through Q, and finally through the real seal R.

Apply the componentwise hsame rows. They replace each displayed source coordinate by the corresponding coordinate in M' without introducing a second schedule, a second stream quotient, or a hidden real representative.

The continuation rows then replay the same route in M'. Since the boundedness row and monotonicity row are transported as rows B' and L' over the same inspected windows, the transported packet preserves the monotone source surface and exports no extra completeness or selection coordinate. □

Theorem 1434.4 (Monotone real sequence convergence handoff)**.** *Every* $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *packet must read the source through*

$$S, W, Q, R, B, L, H, C, P, N$$

before it can form a tail-Cauchy threshold or a convergence handoff. Consequently the consumer receives a bounded monotone real sequence source over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{BoundedRealSequence}^{\uparrow}$ *rows, but receives no selected limit, quotient equality of streams, host real equality, open-cover compactness, trichotomy, or ambient completeness theorem.*

Proof. Start from the carrier of Definition 1434.1. Its source route exposes finite stream windows, regular rational readbacks, and real-name seals before any boundedness or monotonicity statement is available.

Use the obligation theorem, Definition 1434.2, to identify B and L as the only boundedness and monotonicity surfaces in the packet. The transport theorem, Definition 1434.3, keeps those surfaces attached to the same finite source route under accepted classifier movement.

Now read the consumer in chapter 1435. Its tail-Cauchy threshold can consume a bounded monotone source only after the stream windows, regular readbacks, real seals, boundedness ledger, and monotonicity ledger have been displayed. Since the source packet contains no selected limit, quotient equality, host real equality, compactness, trichotomy, or ambient completeness coordinate, the handoff exports only the listed finite route. □

Theorem 1434.5 (Monotone real sequence socket taxonomy)**.** *Every accepted convergence-facing read of a* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *packet separates source-range display, located-limit handoff, and classifier transport. The source range is visible only through the* $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows carried by*

$$S, W, Q, R, B, L, H, C, P, N.$$

The limit handoff is visible only to a displayed located-convergence consumer, and equality movement is visible only through the $\mathsf{Real}^{\uparrow}$ *classifier transport row. No read may collapse these sockets into an unrestricted real-completeness assumption, a stream quotient equality, or a selected limit representative.*

Proof. Project the carrier of Definition 1434.1. The rows S, W, Q expose the source schedule, stream windows, and regular-rational readback, while B and L attach boundedness and monotonicity to those same windows.

Apply Definition 1434.2. It identifies the accepted source rows and leaves no carrier coordinate for an unrestricted real range or a selected limit representative.

Apply Definition 1434.3 and Definition 1434.4. The first theorem keeps equality movement inside the displayed $\mathsf{Real}^{\uparrow}$ classifier route; the second permits a convergence-facing consumer to receive the source only through the displayed handoff. Thus the three socket positions remain separate. □

Falsifiable prediction: Every $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ bounded-monotone source read factors through a $\mathsf{MonotoneRealSequence}^{\uparrow}$ packet displaying inspected StreamName windows, RegSeqRat readbacks, RealUp seals, boundedness and monotonicity ledgers, transport, continuation, provenance, and local naming.

Independence witness: boundedrealsequence, streamname, regseqrat, real, monotone-sequence-convergence: $\mathsf{MonotoneRealSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded real sequence source and a monotonicity ledger before the convergence consumer may form its tail-Cauchy route.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneRealSequence}^{\uparrow}}$.

Constructive story: A MonotoneRealSequence$^{\uparrow}$ packet is generated from inspected sequence indices, finite StreamName windows, RegSeqRat readbacks, RealUp sequence-name seals, a boundedness ledger, a monotonicity ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MonotoneRealSequence}^{\uparrow}}$, seed)

Scope closed: Definition 1434.1, Definition 1434.2, Definition 1434.3, and Definition 1434.4 seed the finite bounded-monotone source consumed by monotone sequence convergence.

Formal verification: FormalStatus(NameCert$_{\text{MonotoneRealSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected limits, quotient equality of streams, host real equality, open-cover compactness, trichotomy, or ambient completeness.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, window transport, convergence handoff, and TasteGate field faithfulness for BEDC.Derived.MonotoneRealSequenceUp.

Chapter 1435

A Concrete Naming Certificate for $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ packet records the constructive monotone convergence route for bounded real sequences as finite BEDC data. It sits after the bounded-sequence source of chapter 1415, the located monotone Cauchy route of chapter 438, the nonquotient comparison boundary of chapter 1404, the finite-window interface of chapter 126, and the terminal seal of chapter 38. The packet turns displayed bounded monotone observations into a regular Cauchy tail certificate; it does not select a limit representative or import host completeness.

Definition 1435.1 (Monotone sequence convergence carrier)**.** *A* $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, W, Q, R, B, M, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the inspected sequence schedule, listing the finite index windows used by the packet.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows observed at the indices in S.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and their dyadic tolerances.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream windows and regular rational readbacks have been displayed.*
5. *B is the bounded interval or bounded net row supplied by the bounded real sequence surface.*
6. *M is the monotonicity ledger, recording the displayed order direction and the finite comparison rows used by the inspected windows.*
7. *T is the tail-Cauchy modulus and threshold row; it names the finite threshold beyond which the Q-readbacks are forced into the displayed tolerance.*
8. *H stores componentwise* hsame *transport for the schedule, windows, readbacks, real seals, boundedness row, monotonicity ledger, tail threshold, continuation, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from inspected indices to windows, from windows to regular readback, from readback to real seals, and from the bounded monotone ledger to the tail-Cauchy threshold.*

10. *P is the* Pkg *provenance over* $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneSequenceConvergence}^{\uparrow}}$ *row.*

The carrier classifier reads only

$$S, W, Q, R, B, M, T, H, C, P, N.$$

It has no coordinate for trichotomy, host real equality, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, a Bolzano–Weierstrass subsequence selector, or a choice-selected limit.*

Lean4 ✓ `BEDC.Derived.MonotoneSequenceConvergenceUp.MonotoneSequenceConvergenceTasteGate single carrier alignment`

Theorem 1435.2 (Monotone sequence tail-Cauchy route)**.** *For every accepted* $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ *carrier $J = (S, W, Q, R, B, M, T, H, C, P, N)$, bounded monotone finite-window rows produce a regular Cauchy tail route through the displayed threshold T and the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier Q. Every tail comparison factors through*

$$S \longmapsto W \longmapsto Q \longmapsto B, M \longmapsto T$$

together with R, H, C, P, N, and no comparison uses trichotomy, host completeness, quotient equality, choice, or a selected limit representative.

Proof. Begin with the inspected schedule S. The continuation row C exposes only the finite $\mathsf{StreamName}^{\uparrow}$ windows W, and the readback row Q classifies those windows as regular rational observations before any $\mathsf{Real}^{\uparrow}$ seal row is consumed.

The boundedness and monotonicity rows are then read together. The row B supplies the bounded real sequence interval or net surface, while M supplies the displayed monotone ledger for the same inspected windows. Their joint use is finite: the packet never asks for a host set of all sequence values.

The threshold row T is the tail-Cauchy output of that bounded monotone read. It records the displayed tail index and tolerance route through the Q-classifier, so downstream regular Cauchy consumers receive a route already expressed in $\mathsf{RegSeqRat}^{\uparrow}$ data.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same route. Since the carrier contains no trichotomy, host-completeness, quotient-equality, choice, or selected-limit coordinate, the tail-Cauchy comparison cannot leave the displayed packet. □

Theorem 1435.3 (Monotone sequence convergence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ *carrier, the local* $\mathsf{NameCert}_{\mathsf{MonotoneSequenceConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window classification, boundedness transport, monotone-ledger transport, tail-Cauchy extraction,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, and package provenance for the displayed rows $S, W, Q, R, B, M, T, H, C, P, N$.*

Proof. Carrier admission is the tuple of Definition 1435.1. The finite-window classifier reads the schedule S, the $\mathsf{StreamName}^{\uparrow}$ windows W, and the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks Q; these rows are the only source of sequence observations.

Boundedness transport uses B together with the componentwise hsame row H and the displayed replay row C. Monotone-ledger transport is parallel: the same H and C rows move the finite order ledger M without adding a host order or trichotomy field.

Tail-Cauchy extraction is exactly the row T produced from the bounded monotone finite-window data. The real-seal boundary is the row R, which is readable only after the W and Q route has been displayed and therefore cannot introduce quotient real equality or ambient completeness.

Finally P records the package provenance and N records the local naming row. These rows name no data outside $S, W, Q, R, B, M, T, H, C, P, N$, so the seven listed obligations exhaust the local certificate surface. □

Falsifiable prediction: Every $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ tail-Cauchy read factors through finite StreamName windows, RegSeqRat readback, boundedness and monotonicity ledgers, a displayed tail threshold, and the RealUp seal row carried by the packet.

Independence witness: boundedrealsequence, monotonecauchy, regularcauchyequivalence, streamname, real: $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the inspected sequence schedule, finite windows, regular readbacks, real seals, boundedness row, monotonicity ledger, tail threshold, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneSequenceConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$ packet is generated from inspected sequence windows, StreamName observations, RegSeqRat readbacks, RealUp seals, a bounded interval or net row, a monotonicity ledger, a tail-Cauchy threshold row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneSequenceConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1435.1, Definition 1435.2, and Definition 1435.3 seed the finite route from bounded monotone RealUp sequence data to a regular Cauchy tail certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneSequenceConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, Bolzano–Weierstrass subsequence selection, open-cover compactness, trichotomy, host real equality, Quot equality, choice, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, finite-window classification, boundedness transport, monotone-ledger transport, tail-Cauchy extraction, RealUp seal non-escape, and package provenance for BEDC.Derived.MonotoneSequenceConvergenceU

Chapter 1436

A Concrete Naming Certificate for DiniUniformConvergence$^{\uparrow}$

Following Bishop and Bridges, the DiniUniformConvergence$^{\uparrow}$ packet records the constructive finite-net surface of the Dini phenomenon over compact metric data. It reads compact-metric domain rows from chapter 129, finite total-bounded nets from chapter 127, monotone continuous-map family rows from chapter 130, uniform-limit comparison rows from chapter 463, terminal real-name seals from chapter 38, finite stream windows from chapter 126, and regular rational readbacks from chapter 301. The packet is a finite certificate for extracting a displayed uniform modulus from monotone pointwise windows; it is not open-cover compactness, countable choice, quotient real equality, host real equality, or a selected limit principle.

Definition 1436.1 (DiniUniformConvergence carrier)**.** *A* DiniUniformConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, T, F, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *domain row, including the visible compact-metric carrier, distance witness, and metric transport surface used by the family;*
2. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row, listing the inspected net points and dyadic radius budgets over the same domain row;*
3. *F is the monotone family of* ContinuousMap$^{\uparrow}$ *rows, indexed only by the finite observation schedule admitted by the packet;*
4. *M is the dyadic modulus schedule assigning each tolerance row to a finite net radius, a family index, and a comparison budget;*
5. *W is the family of* StreamName$^{\uparrow}$ *windows through which all inspected function values and limiting values are read;*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows, carrying the regular rational comparisons used by the monotone test;*
7. *E is the* Real$^{\uparrow}$ *seal row for the limiting value at each inspected window, reached only after W and R have been displayed;*
8. *H stores componentwise* hsame *transports for K, T, F, M, W, R, E;*

9. *C stores the displayed* Cont *replay routes for the finite-net, monotone-family, window, readback, and seal reads;*
10. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open cover, no countable-choice selector, no quotient equality of real names, no host real equality proof, no selected limit, and no ambient compactness theorem beyond the displayed finite-net packet.

Lean4 (definition) `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceUp`

Theorem 1436.2 (Monotone window obligation)**.** *Let $D = (K, T, F, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier. Every monotonicity read used by the packet is confined to the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows in F, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. In particular, a consumer cannot replace the monotone window obligation by an open-cover argument, a countable-choice schedule, quotient real equality, host real equality, or a selected limiting value.*

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier monotone window obligation`

↪ variant `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier monotone window`

Proof. Project the accepted carrier. The only family data available to the monotonicity classifier are the rows F together with the windows W and regular rational readback R.

The schedule M names finite dyadic tolerances and finite family indices. It does not quantify over all indices by a hidden selector; it only points to the rows already present in the packet.

Transport and replay are componentwise. The rows H and C can reread the same finite windows and continuous-map family rows, but they cannot introduce an open cover, quotient equality of real names, host equality, or a selected limit. Hence the monotone obligation is exactly the displayed finite-window obligation. □

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergence monotone window obligation`

Theorem 1436.3 (Uniform modulus extraction)**.** *For every accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier $D = (K, T, F, M, W, R, E, H, C, P, N)$ and every dyadic tolerance row named by M, the packet exports a finite uniform modulus row whose source is the total-bounded net T, whose value comparisons are read through W and R, and whose limiting seals are the displayed* $\mathsf{Real}^{\uparrow}$ *rows in E. The extracted row supplies a* $\mathsf{UniformLimit}^{\uparrow}$*-facing modulus surface without open-cover compactness, countable choice, quotient real equality, host real equality, or selected limits.*

Proof. Begin with a tolerance row in M. Its coordinates name a finite net radius, a finite family index, and the comparison budget to be checked on the net points carried by T.

For each listed net point, read the relevant $\mathsf{ContinuousMap}^{\uparrow}$ value through F, the finite window through W, the regular rational comparison through R, and the limiting seal through E. Monotonicity lets the same family index control all later inspected rows on that finite window surface.

The compact-metric information used here is exactly the total-bounded finite-net row, not an open-cover subcover. Since the packet contains no selector for countable families, no quotient equality of real names, no host real equality proof, and no detached limit coordinate, the resulting modulus is a finite row over T, M, W, R, E and can be handed to the $\mathsf{UniformLimit}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceUniformModulusExtraction`

Theorem 1436.4 (DiniUniformConvergence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DiniUniformConvergence}^{\uparrow}}$ *surface for the carrier of Definition 1436.1 consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the packet* $D = (K, T, F, M, W, R, E, H, C, P, N)$ *with its compact-metric domain, total-bounded finite nets, monotone continuous-map family, dyadic schedule, stream windows, regular readbacks, real seals, transports, continuations, provenance, and local naming row;*
2. *classifier: comparison reads only the displayed domain row* K*, finite-net row* T*, family rows* F*, modulus schedule* M*, windows* W*, readbacks* R*, and seals* E*;*
3. *transport stability:* H *and* C *preserve the same compact-metric, finite-net, family, window, readback, and seal rows under accepted consumer motion;*
4. *exactness: every uniform-modulus export is supplied by the displayed rows* T, M, W, R, E *after the componentwise transport and replay routes are fixed;*
5. *ledger: every* $\mathsf{UniformLimit}^{\uparrow}$*-facing consumer first reads* $K, T, F, M, W, R, E, H, C, P, N$ *and receives no open cover, countable-choice selector, quotient real equality, host real equality, or selected limit.*

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier namecert obligations`

Proof. Carrier admission is projection from the finite packet. Each coordinate named in Definition 1436.1 is a displayed row, so the local certificate admits exactly that inventory.

The classifier is finite because every comparison request starts with a row of M and then reads only the finite net points in T, the family rows in F, the windows in W, the regular rational readback in R, and the real seals in E.

Transport stability uses H componentwise and then replays the same finite route through C. No transport step adds an open cover, quotient equality, host equality, selected limit, or choice schedule.

Exactness and the ledger clause follow from the same inventory. A uniform-limit-facing consumer receives the finite modulus route over T, M, W, R, E together with provenance and local naming; the carrier exposes no other route by which an unrestricted classical Dini theorem could enter. □

Definition 1436.5 (DiniUniformConvergence certificate route)**.** *For an accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier*

$$D = (K, T, F, M, W, R, E, H, C, P, N),$$

the finite certificate route is the ordered path

$$K, T, F, M, W, R, E, H, C, P, N.$$

This is the only accepted route from compact-domain data and finite windows to a $\mathsf{UniformLimit}^{\uparrow}$*-facing modulus surface.*

Theorem 1436.6 (DiniUniformConvergence compact-window exactness)**.** *For every accepted* DiniUniformConverge *carrier* $D = (K, T, F, M, W, R, E, H, C, P, N)$*, every uniform-modulus export factors through the compact metric row* K*, the totally bounded finite-net row* T*, the continuous-map family row* F*, the stream-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *before any* $\mathsf{UniformLimit}^{\uparrow}$*-facing row is read.*

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCompactWindowExactness`

Proof. Project the carrier and the route named in Definition 1436.5. The compact-domain coordinate is K, the finite-net coordinate is T, and the inspected family of continuous maps is stored in F.

Apply Definition 1436.3. Its extracted modulus reads the finite net, the dyadic schedule, the inspected stream windows, the regular rational readback, and the displayed Real seals.

The exactness statement records the same dependency order for any $\mathsf{UniformLimit}^{\uparrow}$-facing consumer. Such a consumer can read a modulus only after the compact row K, finite net T, family rows F, windows W, readback R, and seal E have been displayed, and the remaining rows H, C, P, N only transport, replay, source, and name that finite route. □

Theorem 1436.7 (DiniUniformConvergence monotone-family exhaustion)**.** *For every accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier*

$$D = (K, T, F, M, W, R, E, H, C, P, N),$$

after K and T have fixed the compact finite-net surface, every monotone-family comparison exposed by the packet passes exactly through the continuous-map family rows F, dyadic modulus schedule M, finite StreamName windows W, RegSeqRat readback R, and Real seal rows E. No consumer can add an open-cover proof, countable-choice schedule, quotient real equality, host equality, selected limit, or ambient compactness row to that comparison.

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier monotone family exhaustion`

Proof. Project the carrier and fix the domain boundary. The compact-metric row K and the totally bounded row T are the only rows that determine the finite-net surface inspected by the packet.

Use the monotone-window obligation. Every monotonicity read is confined to F, W, R, and the dyadic schedule M names only finite tolerances and finite family indices over those displayed rows.

Use uniform-modulus extraction and the certificate route. The extracted modulus reads T, M, W, R, E, while the route lists the full packet order $K, T, F, M, W, R, E, H, C, P, N$.

The forbidden objects have no carrier coordinate. Hence a consumer cannot extend the monotone-family comparison by open-cover compactness, countable choice, quotient real equality, host equality, a selected limit, or an ambient compactness row. □

Theorem 1436.8 (DiniUniformConvergence compact uniform-limit route)**.** *For every accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier*

$$D = (K, T, F, M, W, R, E, H, C, P, N),$$

any $\mathsf{UniformLimit}^{\uparrow}$*-facing consumer of the Dini conclusion must read* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *exactly through the displayed rows $K, T, F, W, R, E, H, C, P,$ with M as the dyadic modulus schedule. The route supplies no open-cover compactness, selected limiting value, quotient function equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The public certificate route is the ordered packet $K, T, F, M, W, R, E, H, C, P, N$ so a consumer first receives the compact-metric row K, the total-bounded finite-net row T, the

continuous-map family rows F, the dyadic schedule M, the stream windows W, the regular rational readbacks R, and the real seals E.

Apply Definition 1436.3. It identifies the modulus route through T, M, W, R, E and exposes that route as the finite surface handed to $\mathsf{UniformLimit}^{\uparrow}$.

Apply Definition 1436.6. It fixes the order in which K, T, F, W, R, E must be displayed before any uniform-limit-facing row is read.

Apply Definition 1436.7. It exhausts the monotone-family comparison through F, M, W, R, E after the compact finite-net surface K, T is fixed.

It remains only to account for the ledger rows. The transports H, continuations C, provenance P, and local certificate N reread and name the same finite route; none supplies an open cover, selected limiting value, quotient function equality, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every $\mathsf{UniformLimit}^{\uparrow}$-facing consumer is confined to the displayed route. □

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier compact uniform limit route`

Theorem 1436.9 (DiniUniformConvergence finite-net monotone-tail scope)**.** *For every accepted* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *carrier*

$$D = (K, T, F, M, W, R, E, H, C, P, N),$$

once K, T, F, M have fixed the compact metric row, finite net, monotone family, and dyadic schedule, every tail comparison consumed by a $\mathsf{UniformLimit}^{\uparrow}$*-facing route factors through*

$$W, R, E, H, C, P, N.$$

The exported comparison carries no open-cover compactness, countable choice, quotient real equality, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCarrier finite net monotone tail scope`

Proof. Start with the monotone-window obligation. It confines every monotonicity read to the displayed continuous-map family rows F, finite stream windows W, and regular rational readback R.

Apply uniform-modulus extraction. After the schedule row M fixes the tolerance and family index, the extracted modulus is read through the finite net T, windows W, readbacks R, and Real seals E.

Use compact-window exactness and monotone-family exhaustion. These two theorems keep the compact metric row K, finite-net row T, family rows F, and schedule row M upstream of all tail comparisons.

Finally apply the compact uniform-limit route. It hands the resulting finite comparison to $\mathsf{UniformLimit}^{\uparrow}$ only through W, R, E, H, C, P, N, while the carrier has no coordinate for open-cover compactness, countable choice, quotient real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1436.10 (DiniUniformConvergence cover-selector non-escape)**.** *For every accepted* DiniUniformConverge *carrier*

$$D = (K, T, F, M, W, R, E, H, C, P, N),$$

any cover or selector read used by a $\mathsf{UniformLimit}^{\uparrow}$*-facing Dini route must pass through the compact metric row* K*, total-bounded finite-net row* T*, monotone family row* F*, dyadic schedule* M*, windows* W*, readback* R*, seal* E*, and structural rows* H, C, P, N*. It cannot be replaced by open-cover compactness, countable choice, selected limits, quotient real equality, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the compact uniform-limit route of Definition 1436.8. It fixes the public consumer path through the displayed compact metric, finite-net, family, window, readback, seal, transport, replay, provenance, and local naming rows.

Use Definition 1436.9. Once K, T, F, M are fixed, all tail comparisons exposed to $\mathsf{UniformLimit}^{\uparrow}$ factor through W, R, E, H, C, P, N.

Finally apply the local obligations of Definition 1436.4. The ledger clause admits no open-cover compactness, countable-choice selector, selected limit, quotient real equality, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, so any purported cover or selector read must remain inside the displayed packet route. □

Lean4 ✓ `BEDC.Derived.DiniUniformConvergenceUp.DiniUniformConvergenceCoverSelectorNonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformConvergence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiniUniformConvergence}^{\uparrow}$ packet is generated from compact-metric domain data, total-bounded finite nets, monotone ContinuousMapUp family rows, dyadic modulus schedules, StreamName windows, RegSeqRat readbacks, RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiniUniformConvergence}^{\uparrow}}$, obligation)

Scope closed: Definition 1436.1, Definition 1436.2, Definition 1436.3, and Definition 1436.4 seed the finite-net Dini uniform-modulus packet over the displayed Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiniUniformConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the classical unrestricted Dini theorem beyond the displayed compact-metric finite-net packet, and it exports no open-cover compactness, countable choice, quotient real equality, host real equality, selected limits, or bridge checking.

Upgrade path: A later verification route can encode the carrier and prove the monotone-window obligation, uniform-modulus extraction, and local NameCert obligations over the displayed finite packet.

Chapter 1437

A Concrete Naming Certificate for $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ packet records the constructive one-dimensional Brouwer fixed-point theorem as a finite interval-return route. A candidate continuous self-map on a located compact interval is first read through the $\mathsf{LocatedInterval}^{\uparrow}$ endpoint surface and the $\mathsf{ContinuousMap}^{\uparrow}$ graph and modulus rows; its residual $g(x) = f(x) - x$ is then consumed by the intermediate-value surface of chapter 1402 and the bisection route of chapter 1422; the resulting finite windows are handed to the stream, regular-rational, and real-name surfaces, including the seal of chapter 38. The certificate names only this displayed finite route. It does not claim the general Brouwer fixed-point theorem, Heine–Borel compactness, host completeness, quotient equality of streams, real trichotomy, or a choice-selected fixed point.

Definition 1437.1 (Compact interval fixed-point carrier)**.** *A* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (J, G, R, B, W, Q, E, H, C, P, N)$$

with the following visible rows:

1. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *interval-endpoint packet, including the lower and upper endpoint names, rational cells, and located-order witness used for every later read;*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus evidence for the displayed interval-returning map, together with the residual read route for $g(x) = f(x) - x$;*
3. *R is the interval-return row asserting that every candidate read of the map is routed back into the same located interval packet J before residual signs are inspected;*
4. *B is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *residual-bisection row obtained from the endpoint residual signs, the modulus evidence in G, and the interval-return row R;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface for the retained bisection interval and the displayed candidate fixed-point name;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the retained windows into regular rational approximants at the requested dyadic tolerance;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal exposed to downstream consumers only after the retained windows and regular rational readback have been displayed;*
8. *H stores componentwise* hsame *transports for J, G, R, B, W, Q, E and the local certificate row;*

9. *C stores the displayed* Cont *replay routes for interval-endpoint reads, continuous-map reads, interval-return reads, residual bisection, finite windows, regular readback, and real-seal consumption;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFixedPoint}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite fixed-point certificate for a displayed interval-returning continuous map. The packet has no coordinate for a host fixed point, open-cover compactness, general Brouwer fixed points, real trichotomy, quotient equality of streams, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.CompactIntervalFixedPointTasteGate single carrier alignment`

Theorem 1437.2 (CompactIntervalFixedPoint NameCert obligations)**.** *For an accepted packet $F = (J, G, R, B, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalFixedPoint}^{\uparrow}}$ *obligations are exhausted by the carrier rows of Definition 1437.1. They consist of carrier admission, interval-return exactness, continuous-map modulus replay, residual-sign bisection, stream-window handoff, regular-rational readback, real-seal nonescape, componentwise transport, displayed continuation replay, provenance, and local naming. No obligation may call on a host fixed point, open-cover compactness, general Brouwer fixed points, real trichotomy, quotient stream equality, selected limits, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.CompactIntervalFixedPointNameCertObligations`
↪ variant `BEDC.Derived.CompactIntervalFixedPointUp.CompactIntervalFixedPointCarrier namecert obligations`

Proof. Carrier admission is projection from the displayed packet. The rows $J, G, R, B, W, Q, E, H, C, P, N$ are exactly the data listed in Definition 1437.1.

The analytic obligations are routed in order. The interval-return row R binds every map read back to the same located interval packet J; the continuous-map row G supplies the graph and modulus evidence; and the residual bisection row B reads endpoint signs only after those two rows have been fixed.

The consumer-facing obligations then read the retained windows through W, the regular rational approximants through Q, and the real seal through E. The rows H, C, P, and N transport, replay, record provenance, and register the same finite route.

Since no coordinate outside this route is present, the listed classical, host-completion, quotient, trichotomy, choice, and selected-limit principles cannot enter the local NameCert surface. □

Theorem 1437.3 (Interval-return fixed-point route)**.** *Every candidate fixed-point read accepted by a* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *carrier first consumes the interval-return row R and the continuous-map row G before it can reach the residual bisection row B. Consequently the read factors through*

$$J, G, R, B, W, Q, E, H, C, P, N.$$

It cannot inspect a candidate real name as a fixed point unless the same route has already tied the map value, residual sign data, retained bisection window, stream windows, regular readback, and real seal to the displayed located interval.

Proof. Start with an accepted carrier $F = (J, G, R, B, W, Q, E, H, C, P, N)$. The continuation row C provides no direct path from a consumer to E or to a candidate fixed-point name; it first replays the endpoint and map reads through J and G.

The interval-return row R is then mandatory. It records that every displayed value of the map is read back inside the same located interval packet, so the residual sign test is a row over J and G, not a host-level comparison.

Only after that read does the residual bisection row B become available. The retained interval supplies windows W, the windows supply regular rational readback Q, and the readback supplies the Real-facing seal E.

The transport, continuation, provenance, and local naming rows preserve this order. Thus every accepted fixed-point read factors through the displayed route and has no bypass through a host fixed point or external equality principle. □

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.IntervalReturnRoute.CompactIntervalFixedPoint interval return route`

Theorem 1437.4 (CompactIntervalFixedPoint residual-bisection handoff)**.** *Every accepted* CompactIntervalFixedP *fixed-point read must expose, in order, the* LocatedInterval$^{\uparrow}$ *endpoint packet J, the* ContinuousMap$^{\uparrow}$ *graph and modulus row G, the interval-return row R, the* IntermediateValueBisection$^{\uparrow}$ *residual-bisection row B, the* StreamName$^{\uparrow}$ *window row W, the* RegSeqRat$^{\uparrow}$ *readback row Q, and the* Real$^{\uparrow}$ *seal row E before the endpoint seal is public. Thus the public fixed-point seal is a residual-bisection handoff over displayed finite rows, not a host fixed-point existence assertion.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.ResidualBisectionHandoff.CompactIntervalFixedPoint residual bisection handoff`

Proof. The carrier of Definition 1437.1 lists the only rows from which an accepted read can be assembled. The NameCert obligations of Definition 1437.2 require carrier admission, interval-return exactness, residual bisection, stream-window exposure, regular-rational readback, and real-seal nonescape.

The route theorem Definition 1437.3 forces every candidate read through J, G, R before B can be consumed. The bisection row then supplies retained finite windows W, and those windows are the source for the regular readback row Q.

Finally, Definition 1437.6 makes E reachable only after the displayed bisection, window, and readback rows have been exposed. Since the packet has no coordinate for a host fixed point, the public endpoint seal is exactly the listed finite handoff. □

Theorem 1437.5 (CompactIntervalFixedPoint window regularity)**.** *For every accepted* CompactIntervalFixedPoint$^{\uparrow}$ *carrier $F = (J, G, R, B, W, Q, E, H, C, P, N)$, each retained bisection window exposed by B is consumed by the* StreamName$^{\uparrow}$ *window row W before the* RegSeqRat$^{\uparrow}$ *readback row Q can be read. Componentwise* hsame *transport preserves this window/readback pairing through H, and displayed* Cont *replay preserves the same order through C. Hence no selected limit, host fixed point, or ambient real-completeness principle can bypass the window/readback route.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.WindowRegularity.CompactIntervalFixedPoint window regularity`

Proof. Project the carrier rows of Definition 1437.1. The interval-return route of Definition 1437.3 forces every fixed-point read through J, G, R before the residual bisection row B is available.

Once B has displayed the retained bisection window, the carrier offers only the window row W as the next finite observation surface. The regular-rational row Q is therefore a readback from the already exposed window data, not an independent limit selector.

Transport and replay do not change this order. The row H carries componentwise hsame evidence for the same B, W, Q pairing, and C replays the displayed route without adding a host fixed point or completed real. Thus every accepted read reaches the public seal only through the window/readback surface. □

Theorem 1437.6 (CompactIntervalFixedPoint Real-seal nonescape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *packet is reachable only through* $J, G, R, B, W, Q, E, H, C, P, N$*. The packet exports no host fixed point, no open-cover compactness, no general Brouwer fixed-point theorem, no real trichotomy, no quotient stream equality, no selected limit, and no ambient completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

```
Lean4 ✓  BEDC.Derived.CompactIntervalFixedPointUp.RealSealNonescape.CompactIntervalFixedPoint
real seal nonescape
  ↪ variant  BEDC.Derived.CompactIntervalFixedPointUp.CompactIntervalFixedPointRealSealNonescape
```

Proof. The only Real-facing coordinate is E. By definition of the carrier, E is paired with the windows W and the regular rational readback Q, and both are paired with the residual bisection row B.

The bisection row itself is not free-standing. It is generated from the interval-return and continuous-map rows R and G over the same located interval packet J, with transports and replay stored in H and C.

Package provenance P and the local naming row N register precisely this finite route. Because no other Real-facing or compactness-facing coordinate exists in the packet, a consumer receives only the displayed real-name seal and none of the excluded classical or host principles. □

Theorem 1437.7 (CompactIntervalFixedPoint modulus residual stability)**.** *For every accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *carrier* $F = (J, G, R, B, W, Q, E, H, C, P, N)$*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row* G *and interval-return row* R *keep every residual sign read for* $g(x) = f(x) - x$ *scoped to the same* $\mathsf{LocatedInterval}^{\uparrow}$ *packet* J*. Refining the residual tolerance or replaying the read through* C *may refine the displayed cells of* J*, but it cannot move the residual comparison to a different interval packet, host order, real trichotomy, selected representative, or ambient compactness principle.*

```
Lean4 ✓  BEDC.Derived.CompactIntervalFixedPointUp.ModulusResidualStability.CompactIntervalFixedPointModu
Lean4 ✓  BEDC.Derived.CompactIntervalFixedPointUp.ModulusResidualStability.CompactIntervalFixedPoint
modulus residual stability
```

Proof. Project the rows J, G, R from the carrier. The graph and modulus row G supplies the displayed map read, while R records that the value is read back inside the same located interval packet before residual signs are inspected.

A residual tolerance refinement therefore has only one source for endpoint and cell data: the interval packet J. The bisection row B can consume the residual signs only after G and R have fixed that source.

Continuation replay in C and componentwise transport in H preserve the same row order. Since no carrier coordinate stores a second interval packet, a host order test, trichotomy, selected representatives, or ambient compactness, the residual signs remain scoped to J. □

Theorem 1437.8 (CompactIntervalFixedPoint residual-modulus readback)**.** *For every accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *carrier*

$$F = (J, G, R, B, W, Q, E, H, C, P, N),$$

each residual sign read for $g(x) = f(x) - x$ *must consume the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row* G*, the interval-return row* R*, and the retained residual-bisection row* B *before the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E *are public. Thus residual-modulus information reaches the fixed-point seal only as a finite L10 readback over* $J, G, R, B, W, Q, E, H, C, P, N$*, not as a host fixed point, selected limit, quotient stream equality, real trichotomy, or ambient Real completeness claim.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.ResidualModulusReadback.CompactIntervalFixedPointResi`

Proof. First use modulus residual stability, Definition 1437.7. It keeps every residual sign read scoped to the same located interval packet J after the continuous-map modulus row G and interval-return row R have fixed the displayed source of the residual comparison.

Next apply the residual-bisection handoff of Definition 1437.4. The residual signs can produce public fixed-point data only by entering the retained bisection row B and then handing that finite interval data to the window row W and regular readback row Q.

Finally use window regularity, Definition 1437.5. It preserves the order from B to W to Q, while the Real seal E remains downstream of those rows and the structural rows H, C, P, N transport, replay, record, and name the same route.

Since the carrier contains no coordinate for a host fixed point, selected limit, quotient stream equality, real trichotomy, or ambient Real completeness, no residual-modulus read can bypass the displayed finite route. □

Theorem 1437.9 (CompactIntervalFixedPoint bisection-window monotonicity)**.** *For every accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *carrier* $F = (J, G, R, B, W, Q, E, H, C, P, N)$*, each retained residual-bisection window produced by* B *is consumed as a narrower finite* $\mathsf{StreamName}^{\uparrow}$ *window in* W *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* Q *is available. Extending the bisection schedule refines the retained window and preserves the route* J, G, R, B, W, Q, E*; it cannot introduce a selected limit, quotient stream equality, host fixed point, or ambient Real completeness.*

Proof. The residual-bisection handoff of Definition 1437.4 already places B before W and Q. Thus a retained window is not a free real point; it is finite bisection data over the displayed interval route.

When the bisection schedule is extended, the retained interval is refined inside B. The only consumer row for that retained finite data is W, so the next observable object is a narrower $\mathsf{StreamName}^{\uparrow}$ window.

The regular rational readback row Q is downstream of W, and the Real seal E is downstream of Q. Transport and replay preserve this order, leaving no coordinate for selected limits, quotient stream equality, host fixed points, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.BisectionWindowMonotonicity.CompactIntervalFixedPoint`

Theorem 1437.10 (CompactIntervalFixedPoint Real-seal uniqueness)**.** *For every accepted* CompactIntervalFixedPo *carrier* $F = (J, G, R, B, W, Q, E, H, C, P, N)$*, any public* $\mathsf{Real}^{\uparrow}$ *seal exported as the fixed-point candidate factors through the same residual-bisection row* B*, retained-window row* W*, regular-readback row* Q*, and Real-seal row* E*. A second seal that bypasses* B, W, Q, E *or replaces them by host fixed-point existence is not a* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *public seal for this carrier.*

Proof. The only Real-facing carrier coordinate is E. By Definition 1437.6, E is reachable only after B, W, Q have been displayed.

Suppose a public fixed-point seal is read from the packet. Carrier admission and the obligation surface force that read through the residual-bisection handoff, then through the retained stream windows and regular rational readback.

Any purported second seal must therefore use the same B, W, Q, E route when it is read as part of this carrier. If it instead appeals to host fixed-point existence, open-cover compactness, quotient stream equality, selected limits, or ambient Real completeness, it uses data absent from the carrier and is not a public seal exported by F. □

Theorem 1437.11 (CompactIntervalFixedPoint scoped route)**.** *Every accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *fixed-point seal is scoped through the primitive route*

$$\mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{IntermediateValueBisection}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

The BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows preserve this route from carrier admission to public Real seal. No accepted seal may replace it by host compactness, open-cover compactness, quotient stream equality, selected limits, real trichotomy, or a general Brouwer fixed-point principle.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.ScopedRoute.CompactIntervalFixedPointScopedRoute`

Proof. Start with the carrier of Definition 1437.1. Its visible rows place the located interval packet J before the continuous-map and interval-return rows G, R, and Definition 1437.2 makes those rows part of the local NameCert surface.

The interval-return theorem Definition 1437.3 and residual-bisection handoff Definition 1437.4 force every candidate read through J, G, R before the residual-bisection row B is available. Window regularity Definition 1437.5 then routes the retained bisection data through W and the regular readback row Q.

The Real-facing end of the route is fixed by Definition 1437.6 and Definition 1437.10. The structural rows H, C, P, N carry componentwise transport, replay, provenance, and local naming without adding another primitive source. Thus the scoped primitive route is exactly the displayed chain from $\mathsf{LocatedInterval}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$. □

Theorem 1437.12 (CompactIntervalFixedPoint public seal)**.** *Every public* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *consumer receives a fixed-point seal only through the displayed route*

$$\mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{IntermediateValueBisection}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

together with the hsame*,* Cont*,* Pkg*, and* NameCert *rows of the accepted carrier. The public seal exposes carrier admission, interval-return exactness, residual-bisection handoff, retained-window regularity, Real-seal nonescape, Real-seal uniqueness, and the scoped route; it exports no general Brouwer fixed-point theorem, Heine–Borel compactness, host fixed-point existence, selected limit, quotient stream equality, real trichotomy, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.CompactIntervalFixedPointPublicSeal`

Proof. Start with the carrier and NameCert surface. The carrier of Definition 1437.1 supplies exactly $J, G, R, B, W, Q, E, H, C, P, N$, and Definition 1437.2 binds carrier admission, interval-return exactness, continuous-map replay, residual bisection, finite windows, regular readback, Real seal, transport, replay, provenance, and naming to those rows.

Next compose the route the consumer can read. The interval-return route of Definition 1437.3 places J, G, R before residual signs, and Definition 1437.4 moves the read from the residual row B to retained windows W and regular readback Q.

The public Real boundary is then fixed by window regularity, Real-seal nonescape, and Real-seal uniqueness, namely Definition 1437.5, Definition 1437.6, and Definition 1437.10. These theorems make E reachable only after B, W, Q have been displayed and prevent a second seal that bypasses the same route.

Finally apply the scoped route of Definition 1437.11. It identifies the primitive public chain from $\mathsf{LocatedInterval}^{\uparrow}$ through $\mathsf{Real}^{\uparrow}$, while H, C, P, N preserve the route under transport, continuation replay, provenance, and local naming. Since the carrier contains no row for Brouwer fixed points in higher dimension, Heine–Borel compactness, host fixed-point existence, selected limits, quotient stream equality, real trichotomy, countable choice, or ambient completeness, none of those objects is exported by the public seal. □

Theorem 1437.13 (CompactIntervalFixedPoint public consumer route)**.** *Every public consumer of an accepted* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *packet must read the located interval row* J*, continuous-map row* G*, interval-return row* R*, residual-bisection row* B*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *before it may use the fixed-point seal. The route is the finite public surface*

$$J, G, R, B, W, Q, E, H, C, P, N,$$

not a general Brouwer fixed-point theorem, Heine–Borel compactness, selected fixed point, quotient stream equality, real trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactIntervalFixedPointUp.PublicConsumerRoute.CompactIntervalFixedPointPublicCon`

Proof. Begin with the interval-return route of Definition 1437.3. It forces a public read through the located interval row, the continuous-map row, and the interval-return row before residual signs can be inspected.

Next apply the residual-bisection handoff of Definition 1437.4. The handoff moves the accepted read from B into the finite $\mathsf{StreamName}^{\uparrow}$ windows W and then into the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q.

The Real-facing boundary is fixed by Definition 1437.6. It makes E reachable only after the displayed bisection, window, and readback rows have been exposed.

Finally compose with the public seal theorem, Definition 1437.12. The structural rows H, C, P, N preserve the same carrier-local route, so no public consumer can replace it by general Brouwer fixed points, Heine–Borel compactness, selected fixed points, quotient stream equality, real trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Any accepted fixed-point consumer that reaches a candidate seal while bypassing the interval-return row, the residual-bisection row, the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, or the $\mathsf{Real}^{\uparrow}$ seal row refutes the carrier surface of $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$.

Independence witness: intermediatevalue, intermediate-value-bisection, real, continuousmap, locatedinterval: none is bijective to $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ because this packet binds interval endpoints, interval-returning continuous-map evidence, residual bisection, finite windows, regular readback, terminal Real seal, componentwise transport, continuation replay, package provenance, and local NameCert registration as one fixed-point route.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalFixedPoint}^{\uparrow}}$.

Constructive story: A CompactIntervalFixedPoint$^{\uparrow}$ packet is generated from a LocatedInterval$^{\uparrow}$ endpoint packet, ContinuousMap$^{\uparrow}$ graph and modulus evidence, an interval-return row, an IntermediateValueBisection$^{\uparrow}$ residual bisection row, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactIntervalFixedPoint}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1437.11 and Definition 1437.12 bind the carrier, NameCert obligations, interval-return route, residual-bisection handoff, window regularity, Real-seal nonescape, Real-seal uniqueness, and public fixed-point seal to the primitive route from LocatedInterval$^{\uparrow}$ through Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalFixedPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general Brouwer fixed points, open-cover compactness, Heine–Borel compactness, host fixed-point existence, real trichotomy, quotient stream equality, selected limits, countable choice, Real completeness, or a standard bridge.

Upgrade path: Public closure requires Lean formalization of the public fixed-point seal and the existing scoped route for BEDC.Derived.CompactIntervalFixedPointUp.

Chapter 1438

A Concrete Naming Certificate for $\mathsf{CauchyConvergenceModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ packet records the finite convergence-modulus surface needed by the L10 Real and Completion route. It sits between the finite windows of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the sequence-limit handoff of chapter 1413, and the regular-completion seed of chapter 1425. The packet names a convergence schedule as certificate data before any completion-facing consumer may speak about a tail limit.

Definition 1438.1 (Cauchy convergence modulus carrier)**.** *A* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, T, W, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence-source row whose entries are read as the common source for the convergence claim.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the requested dyadic radii.*
3. *T is the tail-index schedule assigning each tolerance row in D a finite threshold.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window family selected by T.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed tail windows and dyadic tolerances.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row reached only after S, D, T, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, tail schedule, windows, readback, limit handoff, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from each tolerance request through $S, D, T, W, Q, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient limits, host sequence convergence, host real equality, countable choice, trichotomy, selected representatives, or an unbounded tail read.*

Theorem 1438.2 (Cauchy convergence modulus NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}}$ *surface for Definition 1438.1 consists exactly of carrier exposure for K, sequence-source exposure through S, dyadic tolerance exposure through D, tail-index exposure through T, finite StreamName window exposure through W, regular rational readback through Q, RealSequenceLimit handoff through L, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Every accepted convergence request is classified only by the displayed tolerance row, the finite tail threshold, and the corresponding finite tail windows.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceModulusUp.CauchyConvergenceModulusNamecertObligations`

Proof. Project the carrier tuple $K = (S, D, T, W, Q, L, H, C, P, N)$. Each analytic coordinate is a displayed finite row, and the final coordinate names the same tuple as the local $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$ surface.

Read the rows in the convergence order. A tolerance request first consumes the source row S and dyadic ledger D, then the tail schedule T, then the finite tail windows W, and only then the regular readback Q and sequence-limit handoff L.

The structural rows add no hidden convergence principle. They transport, replay, source, and name the displayed route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface. □

Theorem 1438.3 (Cauchy convergence modulus tail-window cofinality)**.** *For every accepted* CauchyConvergenceMo *packet $K = (S, D, T, W, Q, L, H, C, P, N)$ and every displayed dyadic tolerance request, the completion-facing read factors through the cofinal finite route*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto (H, C, P, N).$$

Every later use by $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *must read the same displayed tail-index schedule and the same finite windows before it reaches the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff. The consumer receives no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient limit, host sequence convergence, host real equality, countable-choice row, trichotomy row, selected representative, or detached completion datum.*

Proof. Start with Definition 1438.2. It places the dyadic tolerance row D before the tail-index schedule T and places T before the finite StreamName windows W.

The regular readback is then forced through the same finite tail window. The row Q is read only after S, D, T, W are displayed, so no quotient limit, host convergence predicate, or selected representative can substitute for the scheduled tail comparison.

Finally read the handoff coordinate. The row L is reached after Q, and the rows H, C, P, N only preserve and name that route. Thus every completion-facing consumer factors through the displayed cofinal tail-window route. □

Theorem 1438.4 (Cauchy convergence modulus tail-window exactness)**.** *For every accepted* CauchyConvergenceMo *packet $K = (S, D, T, W, Q, L, H, C, P, N)$ and every displayed dyadic tolerance request in D, the request reaches exactly the tail schedule T and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q is inspected. No consumer may replace that schedule by an unbounded tail, a detached modulus, a host convergence predicate, a quotient limit, or a selected representative.*

Proof. Project the carrier definition. The tolerance ledger D is the only source of displayed precision requests, and the next analytic row in the packet is the tail-index schedule T.

Use the NameCert obligations and tail-window cofinality. They force each admitted request to pass from D through T and then through the finite window family W before the regular rational readback Q is available.

Apply the ledger refusal clause. Since the refused host, quotient, selected, and unbounded-tail objects have no carrier row or provenance edge, the displayed schedule and its finite windows are the exact pre-readback route. □

Theorem 1438.5 (Cauchy convergence modulus limit-readback exhaustion)**.** *Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet receives only the rows*

$$S, D, T, W, Q, L, H, C, P, N.$$

The consumer may read the source, tolerance ledger, tail schedule, finite windows, RegSeqRat readback, limit handoff, componentwise transport, displayed continuation replay, package provenance, and local NameCert row, but it receives no additional completion datum, host convergence predicate, quotient stream, selected representative, countable-choice selector, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the carrier and NameCert obligation theorem. Together they expose exactly the ten displayed rows and make the limit handoff L downstream of S, D, T, W, Q.

Tail-window cofinality and tail-window exactness identify the consumer route. A $\mathsf{RealSequenceLimit}^{\uparrow}$ or $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ read must pass through the same scheduled finite windows and the same regular readback before it can inspect L or any structural row.

Finally use ledger refusal. The packet has no row for an independent completion object, host convergence, quotient streams, selected representatives, choice, trichotomy, or ambient completeness, so the listed rows exhaust the consumer-visible readback. □

Theorem 1438.6 (Cauchy convergence modulus carrier transport)**.** *Let* $K = (S, D, T, W, Q, L, H, C, P, N)$ *be an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, and let*

$$K' = (S', D', T', W', Q', L', H', C', P', N')$$

be a second displayed ten-row packet. If the transport row supplies componentwise hsame *comparisons from* $S, D, T, W, Q, L, H, C, P, N$ *to* $S', D', T', W', Q', L', H', C', P', N'$ *in the listed order, then* K' *is accepted by the same* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$ *surface. The transported packet exposes the same convergence-modulus route through sequence source, dyadic tolerance ledger, tail-index schedule, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, replay, provenance, and local name.*

Proof. Project the accepted packet K and read the carrier rows in the displayed order. The source, tolerance, schedule, window, readback, and handoff coordinates are finite rows, so componentwise hsame transport sends each exposed coordinate to the corresponding coordinate of K'.

Next read the structural rows. The transported row H' records the same componentwise comparisons, C' replays the displayed route, P' keeps the same sibling provenance over chapter 291, chapter 126, chapter 301, chapter 1413, and chapter 1425, and N' names the transported tuple locally.

The NameCert surface therefore accepts K' without opening any coordinate outside the ten displayed rows. □

Theorem 1438.7 (Cauchy convergence modulus classifier stability)**.** *Let $K = (S, D, T, W, Q, L, H, C, P, N)$ and $K' = (S', D', T', W', Q', L', H', C', P', N')$ be accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packets. If the displayed source row, dyadic tolerance ledger, tail-index schedule, finite StreamName windows, RegSeqRat readback, and RealSequenceLimit handoff agree componentwise through the packet's* hsame *rows, then every completion-facing consumer sees the same finite convergence classifier from K and K'. In particular, the consumer can distinguish packets only by the displayed route*

$$S, D, T, W, Q, L$$

together with its transport, continuation replay, provenance, and local naming rows.

Proof. Begin with the tail-window cofinality theorem. Any completion-facing read must pass through S, D, T, W, Q, L before it reaches the structural rows, and the hypotheses identify those analytic rows componentwise.

Use carrier transport for the structural part. The rows H, C, P, N and H', C', P', N' preserve, replay, source, and name the same finite route, with the sibling anchors supplying only the displayed tolerance, window, readback, limit-handoff, and completion-facing surfaces.

Thus the classifier available to a completion consumer is determined by the shared finite rows. No ambient completion, quotient limit, host convergence predicate, or selected representative can refine the comparison. □

Theorem 1438.8 (Cauchy convergence modulus ledger refusal)**.** *Every consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet factors through the displayed rows $S, D, T, W, Q, L, H, C, P, N$. The packet refuses reads of* $\mathsf{Real}^{\uparrow}$ *completeness, quotient limits, host convergence, host real equality, countable choice, trichotomy, selected representatives, and any unbounded tail datum.*

Proof. Read the carrier definition and the NameCert obligation theorem together. They expose exactly the ten displayed rows and classify tolerance requests by the sequence source, dyadic ledger, scheduled finite tail threshold, StreamName windows, RegSeqRat readback, and RealSequenceLimit handoff.

The cofinality theorem forces later completion-facing uses through the same finite route. Carrier transport and classifier stability show that replacing the packet by a componentwise transported packet does not create a new consumer-visible coordinate.

Therefore every permitted consumer read is already a read of $S, D, T, W, Q, L, H, C, P, N$. The listed host, choice, quotient, representative, and unbounded-tail objects have no carrier row, continuation replay, provenance edge, or local name in the packet. □

Definition 1438.9 (Cauchy convergence modulus Real-seal surface)**.** *For an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, the Real-seal surface is the displayed boundary*

$$(D, T, W, Q, L, E, H, C, P, N)$$

where D is the dyadic tolerance ledger, T is the tail-index schedule, W is the finite $\mathsf{StreamName}^{\uparrow}$ *tail-window family, Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff, E is the completion-facing seal read by* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$*, and H, C, P, N carry transport, replay, provenance, and local naming. The surface is read only after the sequence source row of Definition 1438.1 has exposed the same tolerance request.*

Theorem 1438.10 (Cauchy convergence modulus completion readiness)**.** *Every accepted* $\mathsf{CauchyConvergenceModu}$ *packet exposes a completion-ready route from dyadic tolerance to regular Cauchy seal:*

$$D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto E \longmapsto (H, C, P, N).$$

The route is available to $\mathsf{RealSequenceLimit}^{\uparrow}$ *and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumers only through the displayed Real-seal surface of Definition 1438.9; it exports no host convergence predicate, quotient limit, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier and NameCert obligation theorems. They place the dyadic tolerance row before the tail-index schedule and finite StreamName windows, then route the resulting readback through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{RealSequenceLimit}^{\uparrow}$.

Apply tail-window cofinality. Any completion-facing consumer must replay the same finite windows before it can read the regular rational row Q or the limit handoff L.

Finally use classifier stability and ledger refusal. Transport preserves the displayed route, and the refused host, quotient, choice, representative, and ambient-completion objects have no row in the Real-seal surface. The remaining visible boundary is exactly the completion-ready route displayed above. □

Theorem 1438.11 (Cauchy convergence modulus regular-Cauchy handoff)**.** *Every* RegularCauchyCompletionUniver *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet must read the Real-seal surface*

$$(D, T, W, Q, L, E, H, C, P, N)$$

before it can treat the packet as a regular-Cauchy completion input. In particular, the consumer must pass through the dyadic tolerance ledger, the tail-index schedule, the finite StreamName windows, the RegSeqRat readback row, and the RealSequenceLimit handoff before it reaches the completion-facing seal E*.*

Proof. Use completion readiness to put every completion-facing read on the displayed route from D through E. The route includes exactly the tolerance, schedule, window, readback, and handoff rows needed by the regular-Cauchy consumer.

The carrier-transport theorem keeps the route stable under componentwise hsame replacement. The classifier-stability theorem then shows that two packets with the same displayed rows present the same regular-Cauchy handoff to a completion consumer.

The ledger-refusal theorem excludes the alternate routes. A consumer cannot bypass D, T, W, Q, L by appealing to host convergence, quotient limits, selected representatives, or ambient completeness, because those objects are absent from the packet. □

Theorem 1438.12 (Cauchy convergence modulus uniform threshold handoff)**.** *Every* RegularCauchyCompletionUni *or* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet that receives a displayed tolerance request must use the same threshold row* T *for all tail windows* W *associated to that request before it can read the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff* L*, or the completion-facing seal* E*.*

The handoff is the finite route

$$D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto E$$

inside the carrier and Real-seal surface. It exports no quotient limit, selected representative, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or detached completion datum.*

Proof. Project the carrier of Definition 1438.1. The tolerance ledger D is followed by the single displayed tail-index schedule T, and the finite window family W is read only after that threshold row is exposed.

Apply tail-window cofinality. It forces every completion-facing use of the displayed tolerance request through the same scheduled tail windows before any regular rational readback is available.

Use completion readiness and the regular-Cauchy handoff for the downstream boundary. They place Q, then L, then the completion-facing seal E after the same threshold and window rows.

The carrier and Real-seal surface contain no quotient limit, selected representative, countable-choice selector, ambient completeness theorem, or detached completion coordinate. Therefore the threshold handoff is uniform across all tail windows associated to the displayed request. □

Theorem 1438.13 (Cauchy convergence modulus uniform tail grid)**.** *Let $K = (S, D, T, W, Q, L, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, and suppose a displayed dyadic grid refinement is attached to the tolerance row D, the tail schedule T, and the finite tail-window family W. Then every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer of K must route the requested read through the same finite tail grid before reading the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff L, or the completion-facing seal E.*

The finite grid route is

$$D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto E$$

inside the carrier and Real-seal surface. It exports no quotient stream, selected representative, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, detached completion datum, or alternate tail grid.*

Proof. Project the carrier and apply the uniform threshold handoff. The tolerance row D selects the tail schedule T, and the displayed grid refinement is read only through the finite tail-window family W attached to that schedule.

Use tail-window cofinality and tail-schedule totality. They force each accepted Real-sequence or regular-completion request to pass through the same scheduled finite windows before any $\mathsf{RegSeqRat}^{\uparrow}$ readback can be inspected.

Completion readiness then places Q, L, and the completion-facing seal E downstream of that grid. Scoped support and ledger refusal exclude quotient streams, selected representatives, countable-choice schedules, ambient completeness rows, and alternate tail grids, so the displayed finite route is the only consumer-visible path. □

Theorem 1438.14 (Cauchy convergence modulus Bishop criterion handoff)**.** *Every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet must read only the displayed route*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The handoff is anchored in the BHist *carrier of Definition 1438.1, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff, and the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *completion boundary. It uses the NameCert obligations of Definition 1438.2, tail-window cofinality of Definition 1438.3, completion readiness of Definition 1438.10, and regular-Cauchy handoff of Definition 1438.11; it exposes no quotient stream, selected representative, host convergence predicate, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the accepted carrier. The displayed rows $S, D, T, W, Q, L, H, C, P, N$ are the only analytic and structural coordinates available to the completion criterion.

Use the NameCert obligations and tail-window cofinality. They place the dyadic tolerance ledger, tail-index schedule, finite windows, and regular readback before any sequence-limit or completion-facing consumer can read the packet.

Completion readiness and the regular-Cauchy handoff identify the boundary seen by $\mathsf{BishopCompletionCriterion}^{\uparrow}$. The consumer therefore receives exactly the displayed route, while the refused quotient, selected, host, choice, and ambient-completion objects remain outside the carrier. □

Theorem 1438.15 (Cauchy convergence modulus root-unblock tail route)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet* $K = (S, D, T, W, Q, L, H, C, P, N)$*, every downstream root-unblock request for a regular tail comparison must read the route*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto Q \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before any completion-facing row is consumed. The route exposes the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff, and structural rows, and it exports no quotient stream, selected representative, host convergence predicate, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 1438.1. The source row S, tolerance ledger D, tail schedule T, finite windows W, regular readback Q, and limit handoff L are displayed in the accepted packet.

Apply tail-window cofinality and tail-schedule totality. They force each regular tail request to pass through the scheduled $\mathsf{StreamName}^{\uparrow}$ windows after the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row and before the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the Bishop criterion handoff to identify the same route for downstream consumers. The structural rows H, C, P, N preserve the route, while the refused host, quotient, selected, choice, and ambient-completeness objects have no carrier row. □

Theorem 1438.16 (Cauchy convergence modulus RealSeries consumer handoff)**.** *Every* $\mathsf{RealSeries}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet must obtain its convergence-modulus read through*

$$S, D, T, W, Q, L, H, C, P, N.$$

The consumer first reads the $\mathsf{StreamName}^{\uparrow}$ *source windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the tail-index schedule and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, then the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff and structural rows. It receives no host series convergence theorem, quotient sum, selected limiting value, countable-choice schedule, trichotomy row, or ambient completeness witness.*

Proof. Begin with the Bishop criterion handoff. It already places completion-facing consumers on the finite route through $S, D, T, W, Q, L, H, C, P, N$.

Read the $\mathsf{RealSeries}^{\uparrow}$ consumer as a series-facing use of the same regular tail data. The only available convergence evidence is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ request, the scheduled $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff.

Completion readiness and the Real-seal nonescape theorem exclude external series convergence data. Thus the $\mathsf{RealSeries}^{\uparrow}$ consumer receives exactly the finite convergence-modulus packet and no host sum or selected limit. □

Theorem 1438.17 (Cauchy convergence modulus regular completion unblock)**.** *For every* RegularCauchyCompletio *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, the regular-completion unblock surface is exhausted by the displayed rows*

$$S, D, T, W, Q, L, H, C, P, N.$$

The unblock first consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and* $\mathsf{StreamName}^{\uparrow}$ *finite windows, then repacks the read as* $\mathsf{RegSeqRat}^{\uparrow}$ *data, then reaches the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff and the completion boundary. No quotient completion object, selected representative, host equality test, countable-choice selector, trichotomy split, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem is supplied.*

Proof. Use completion readiness and the regular-Cauchy handoff. They place $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ consumers on the route from the tolerance ledger through scheduled windows, regular readback, and the sequence-limit handoff.

Apply the root-unblock tail route above. The same S, D, T, W, Q, L rows are available before any completion boundary is consumed, and H, C, P, N keep transport, replay, provenance, and naming attached to the packet.

Finally apply ledger refusal and Real-seal nonescape. Since the refused completion, selected, host, choice, trichotomy, and ambient-completeness objects have no displayed row, the regular-completion unblock is exactly the finite route named in the statement. □

Theorem 1438.18 (Cauchy convergence modulus threshold totality)**.** *For every accepted* $\mathsf{CauchyConvergenceModu}$ *packet* $K = (S, D, T, W, Q, L, H, C, P, N)$*, every displayed completion-facing tolerance request has a threshold row inside the same packet. After the source row* S *and dyadic tolerance ledger* D *are read, the permitted route must pass through*

$$T \longmapsto W \longmapsto Q \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before any $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *or* $\mathsf{RealSeries}^{\uparrow}$ *consumer can inspect a completion boundary. No host convergence predicate, quotient limit, selected representative, countable-choice selector, or ambient completeness row supplies an alternate threshold.*

Proof. Project the carrier of Definition 1438.1. The rows $S, D, T, W, Q, L, H, C, P, N$ are the only displayed coordinates, and T is positioned between the tolerance ledger and finite-window readback.

Apply tail-window cofinality. Every accepted dyadic request that leaves D enters the scheduled finite window row W through the threshold row T, so the threshold is total for the packet's displayed requests.

Use ledger refusal and non-escape. Because the carrier has no host convergence predicate, quotient boundary, selected representative, choice schedule, or ambient completeness coordinate, no second threshold source can be consumed by downstream completion or series readers. □

Theorem 1438.19 (Cauchy convergence modulus RealSeries budget lock)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet* $K = (S, D, T, W, Q, L, H, C, P, N)$*, any* $\mathsf{RealSeries}^{\uparrow}$ *budget read is locked to the finite route*

$$S, D, T, W, Q, L, H, C, P, N.$$

The series-facing consumer may spend only the displayed dyadic tolerance ledger, scheduled tail threshold, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, transport, replay, provenance, and local naming rows. It receives no hidden series convergence theorem, quotient sum, selected limit, countable-choice budget, or ambient $\mathsf{Real}^{\uparrow}$ *completeness resource.*

Proof. Begin with the regular-completion unblock route and the RealSeries consumer handoff. Both place the series-facing read on the same finite carrier coordinates before any Real-facing seal can be consumed.

Threshold totality fixes the budget source. Once D has been read, the row T determines the finite windows W available to Q and L, and the structural rows H, C, P, N can replay only that route.

Finally apply ledger refusal. The packet stores no hidden convergence theorem, quotient sum, selected limit, choice budget, or ambient completeness resource, so the RealSeries budget is locked to the displayed rows. □

Theorem 1438.20 (Cauchy convergence modulus completion-boundary exactness)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet* $K = (S, D, T, W, Q, L, H, C, P, N)$*, the completion boundary exported to* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *is exact only along the displayed route*

$$S, D, T, W, Q, L, H, C, P, N.$$

The boundary may read the source, tolerance, threshold, finite windows, regular readback, limit handoff, componentwise transport, displayed continuation replay, package provenance, and local NameCert rows, but it may not import a quotient completion object, selected representative, host equality test, countable-choice selector, trichotomy split, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier and apply the regular-completion unblock theorem. It already exhausts the boundary surface by $S, D, T, W, Q, L, H, C, P, N$.

Use threshold totality and the RealSeries budget lock as compatibility checks. They show that both completion-facing and series-facing consumers reuse the same threshold, finite-window, readback, and handoff rows rather than creating a second boundary.

The refusal ledger closes the exactness claim. Since quotient completion objects, selected representatives, host equality tests, choice selectors, trichotomy splits, and ambient completeness theorems have no row in K, the exported completion boundary is exactly the displayed finite route. □

Theorem 1438.21 (Cauchy convergence modulus tail schedule totality)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet and every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer, the tail-index schedule* T *is total for the displayed finite-window route. After the source row* S *and dyadic tolerance ledger* D *are read, every permitted completion-facing request must pass through* T, W, Q, L, H, C, P, N *in the same* BHist *packet, with no alternate tail schedule, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from the carrier and NameCert obligation surfaces. They expose T as the named tail-index schedule between the tolerance ledger D and the finite window row W.

Tail-window cofinality supplies the totality witness for displayed requests: each accepted dyadic tolerance request is routed through a scheduled finite window before Q and L are available. The carrier-transport theorem preserves that row under componentwise hsame replacement.

Finally apply classifier stability and ledger refusal. Since no other tail schedule or completion coordinate is present in the packet, every Bishop-completion-facing request consumes the displayed T, W, Q, L, H, C, P, N route. □

Theorem 1438.22 (Cauchy convergence modulus regular completion readiness)**.** *An accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet is ready for current L10 regular-completion consumers exactly through the finite route*

$$S, D, T, W, Q, L, H, C, P, N.$$

The readiness claim is anchored in BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, *and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *data, and every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *consumer must use the displayed route rather than a quotient-stream boundary, selected host limit, host equality test, countable-choice selector, or ambient completeness theorem.*

Proof. Use completion readiness to put Real-sequence and regular-completion reads on the route from D through T, W, Q, L and the structural rows. The source row S supplies the displayed sequence input for that route.

The Bishop criterion handoff theorem then identifies the same route as the only surface available to $\mathsf{BishopCompletionCriterion}^{\uparrow}$ consumers. Tail-schedule totality prevents a consumer from replacing the displayed schedule with an unbounded or hidden tail.

The remaining exclusions follow from classifier stability and ledger refusal. The packet is therefore completion-ready only as finite BEDC certificate data over the displayed rows. □

Theorem 1438.23 (Cauchy convergence modulus real-sequence-limit consumer)**.** *Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet must read the displayed* BHist *carrier route*

$$S, D, T, W, Q, L, H, C, P, N$$

before it can consume the limit handoff row L. The consumer receives the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row, but it receives no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, selected limit, host convergence predicate, countable-choice selector, or quotient representative.*

Proof. Project the carrier route of Definition 1438.1. The limit consumer can reach L only after the source, tolerance, schedule, window, and regular readback rows have been displayed.

Use Definition 1438.3 to make the route cofinal for every displayed tolerance request. The finite window row W and regular readback row Q cannot be skipped, because they are the route through which the tolerance ledger D becomes a sequence-limit handoff.

Finally use classifier stability and ledger refusal. The structural rows H, C, P, N preserve the displayed packet, while the excluded host and choice objects have no row in the BHist carrier. Thus the $\mathsf{RealSequenceLimit}^{\uparrow}$ consumer receives exactly the displayed finite convergence-modulus data. □

Theorem 1438.24 (Cauchy convergence modulus completion-threshold exhaustion)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, the completion threshold exposed to a* $\mathsf{RegularCauchyCompletionUnivers}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer is exhausted by the finite rows D, T, W, Q, L together with H, C, P, N. Once the dyadic tolerance row D has selected a tail-index row T, every permitted completion read must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window family W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff L.*

Proof. Begin with the regular-Cauchy handoff theorem. It places completion consumers on the route from the dyadic tolerance ledger through the tail-index schedule, finite windows, regular readback, and limit handoff.

Apply completion readiness to identify the Real-seal surface. The seal row is reached only after the displayed threshold and window rows have been read, and its structural support is limited to hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

The carrier contains no further threshold coordinate. Hence the displayed tail-index row and its finite window family exhaust the completion threshold available to Real and completion consumers. □

Theorem 1438.25 (Cauchy convergence modulus Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{RealSequenceLimit}^{\uparrow}$ *rows and through the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The seal exports no ambient real completeness theorem, quotient limit, host sequence convergence, host real equality, countable choice, trichotomy, selected representative, or unbounded tail read.*

Proof. Read the Real-seal surface of Definition 1438.9. The seal is attached to the same dyadic tolerance ledger, tail-index schedule, finite windows, regular readback, and sequence-limit handoff as the carrier route.

Use Definition 1438.24. It says that no completion-facing threshold remains beyond D, T, W, Q, L and the structural rows, so a Real-facing consumer cannot replace the displayed route by an unbounded host tail.

Use Definition 1438.8 for the excluded objects. Since the packet has no row for real completeness, quotient limits, host equality, countable choice, trichotomy, selected representatives, or unbounded tail data, the exported $\mathsf{Real}^{\uparrow}$ seal cannot escape the finite convergence-modulus route. □

Theorem 1438.26 (Cauchy convergence modulus scoped support)**.** *For every accepted* CauchyConvergenceModulus *packet, the scoped support is exactly the sequence-source row S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the tail-index schedule T, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff L, the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer boundary, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N. No quotient limit, selected tail, host convergence predicate, host real equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or unbounded tail read lies in the packet scope.*

Proof. Project the carrier and NameCert obligations. The packet exposes $S, D, T, W, Q, L, H, C, P, N$, and the local NameCert obligations require every admitted convergence-modulus read to pass through those displayed rows.

Use tail-window cofinality and carrier transport next. Cofinality routes each dyadic tolerance request through the tail-index schedule and finite StreamName windows, while transport preserves the same route under componentwise hsame replacement.

Completion readiness, the regular-Cauchy handoff, and the Real-seal nonescape theorem identify the consumer boundary. They place $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers on the same finite path and exclude all host, quotient, selected, choice, trichotomy, ambient-completeness, and unbounded-tail rows from the scoped support. □

Theorem 1438.27 (Cauchy convergence modulus scope binding)**.** *For every accepted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *packet, the carrier rows*

$$S, D, T, W, Q, L, H, C, P, N$$

are scoped exactly by BHist *admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff, and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumption. No consumer can enlarge this scope by adding a quotient limit, selected representative, countable-choice selector, trichotomy row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or host convergence predicate.*

Proof. Project the carrier and NameCert obligations. They display $S, D, T, W, Q, L, H, C, P, N$ as the only rows available to an admitted convergence-modulus request.

Apply tail-window cofinality and carrier transport. Cofinality forces each dyadic tolerance request through the scheduled finite StreamName window before readback, while transport preserves the same row list under componentwise hsame replacement.

Use completion readiness, the regular-Cauchy handoff, scoped support, and ledger refusal to close the consumer boundary. The RealSequenceLimit and regular-completion consumers therefore receive exactly the displayed finite scope, and the refused host, quotient, selected, choice, trichotomy, and ambient-completeness objects remain outside the packet. □

Theorem 1438.28 (Cauchy convergence modulus RealSeries readback)**.** *Every* RealSeries$^{\uparrow}$ *consumer that cites a* CauchyConvergenceModulus$^{\uparrow}$ *packet must read* D, T, W, Q, L *through the existing tail-window cofinality route and through the* Real$^{\uparrow}$ *seal surface before using any series convergence claim. The consumer receives a finite convergence-modulus readback over* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealSequenceLimit$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows, not a host infinite sum, selected limit, quotient equality, countable-choice tail selector, or ambient real completeness principle.*

Proof. Begin with tail-window cofinality. It routes each tolerance request from the DyadicRatCore ledger through the shared tail schedule and the finite StreamName windows before the RegSeqRat readback can be consumed.

Next apply completion readiness and the Real-seal nonescape theorem. They place the limit handoff L and any real-facing series read behind the same finite route, so a RealSeries$^{\uparrow}$ consumer cannot use a series convergence claim before reading D, T, W, Q, L.

Finally use scoped support and ledger refusal. These identify the structural rows H, C, P, N as preservation and naming data, while host infinite sums, selected limits, quotient equalities, countable-choice selectors, and ambient completeness principles have no coordinate in the packet. □

Theorem 1438.29 (Cauchy convergence modulus completion consumer scope)**.** *Every* RegularCauchyCompletionUr *or* RealSequenceLimit$^{\uparrow}$ *consumer of a* CauchyConvergenceModulus$^{\uparrow}$ *packet receives only the displayed finite scope*

$$S, D, T, W, Q, L, H, C, P, N.$$

It receives the sequence source, dyadic tolerance ledger, tail schedule, finite windows, regular rational readback, limit handoff, componentwise transport, displayed replay, provenance, and local NameCert rows, but it receives no quotient limit, selected representative, countable-choice selector, trichotomy row, or ambient Real$^{\uparrow}$ *completeness theorem.*

Proof. Use the regular-Cauchy handoff and completion readiness theorems. They identify RegularCauchyCompletion and RealSequenceLimit$^{\uparrow}$ as consumers of the same finite route from S through D, T, W, Q, L.

Apply scope binding to attach the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming preserve the displayed packet without adding analytic coordinates.

The ledger-refusal and Real-seal nonescape theorems provide the exclusions. Since quotient limits, selected representatives, countable-choice selectors, trichotomy rows, and ambient completeness theorems are not rows of the carrier, no completion consumer can receive them from this packet. □

Theorem 1438.30 (Cauchy convergence modulus mature package)**.** *For every accepted* CauchyConvergenceModulu *carrier*

$$K = (S, D, T, W, Q, L, H, C, P, N),$$

the standard completion-facing interpretation is exactly the finite route through the StreamName$^{\uparrow}$ *windows* W, DyadicRatCore$^{\uparrow}$ *tolerances* D, RegSeqRat$^{\uparrow}$ *readback* Q, RealSequenceLimit$^{\uparrow}$ *handoff* L, RegularCauchyCompletionUniversal$^{\uparrow}$ *consumers, componentwise* hsame *transport* H, *displayed* Cont *replay* C, Pkg *provenance* P, *and local naming* N. *The interpretation exports no quotient limit, selected representative, countable-choice selector, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Proof. Start from the NameCert obligations and tail-window cofinality. They force every tolerance request to pass through the displayed sequence source, dyadic ledger, tail schedule, and finite StreamName window before any regular-rational readback is available.

Apply completion readiness, the regular-Cauchy handoff, and the completion consumer scope theorem. These place the RealSequenceLimit row and RegularCauchyCompletionUniversal consumers on the same finite route, so the completion-facing read is a standard interpretation of the ten carrier rows rather than an extra limit object.

Finally use the Real-seal nonescape, scoped support, and scope binding theorems. Transport, replay, provenance, and local naming preserve $S, D, T, W, Q, L, H, C, P, N$ as the whole visible package, while quotient limits, selected representatives, countable-choice selectors, and ambient Real completeness remain outside the carrier. □

Falsifiable prediction: An accepted $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ consumer that reaches a $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff without the displayed sequence source, dyadic tolerance ledger, tail-index schedule, finite StreamName windows, and RegSeqRat readback refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, realsequencelimit, regularcauchycompletionuniversal: no listed sibling supplies the whole ten-row packet $K = (S, D, T, W, Q, L, H, C, P, N)$, because each sibling contributes only a tolerance, window, readback, limit, or completion-facing surface while this chapter binds all five analytic faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ packet is generated from a sequence-source row, dyadic tolerance ledger, tail-index schedule, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1438.1, Definition 1438.2, Definition 1438.3, Definition 1438.4, Definition 1438.5, Definition 1438.6, Definition 1438.7, Definition 1438.8, Definition 1438.9, Definition 1438.10, Definition 1438.11, Definition 1438.12, Definition 1438.13, Definition 1438.14, Definition 1438.21, Definition 1438.22, Definition 1438.23, Definition 1438.24, Definition 1438.25, Definition 1438.26, Definition 1438.27, and Definition 1438.30 expose the public L10 export route through the sequence source, dyadic tolerance ledger, tail-index schedule, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, RegularCauchyCompletionUniversal consumers, and the structural rows H, C, P, N.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulus}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.CauchyConvergenceModulusUp.CauchyConvergenceModulusNamecertObligations`

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient limits, host sequence convergence, host real equality, countable choice, trichotomy, selected representatives, ambient completion, or bridge checking.

Upgrade path: No paper-side closure grade remains; further work is only Lean bridge checking.

Chapter 1439

A Concrete Naming Certificate for $\mathsf{StolzCesaro}^{\uparrow}$

Following Stolz and Cesaro, and in the Bishop–Bridges style of constructive convergence, the $\mathsf{StolzCesaro}^{\uparrow}$ packet records the finite certificate surface for a quotient-tail convergence argument before it reaches a Real-completion consumer. It sits over the finite windows of chapter 126, the regular rational source of chapter 301, the dyadic estimate ledger of chapter 291, the convergence schedule surface of chapter 1438, the monotone source discipline of chapter 1405, and the real-name seal of chapter 38. The chapter names the displayed finite route used by the convergence certificate; it does not import a host theorem about all real sequences.

Definition 1439.1 (Stolz-Cesaro finite convergence carrier)**.** *A* $\mathsf{StolzCesaro}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the numerator sequence rows inspected by the packet.*
2. *B is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the denominator sequence rows inspected by the same tail schedule.*
3. *ΔA is the numerator difference row, recording the displayed differences used by the ratio classifier.*
4. *ΔB is the positive denominator step row, recording the displayed denominator steps and their positivity evidence.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ratio-difference ledger for the inspected step quotients and requested tolerances.*
6. *T is the finite tail-window schedule deciding which inspected windows are admitted for each dyadic tolerance.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from $A, B, \Delta A, \Delta B, D, T$.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after the finite stream, difference, dyadic, schedule, and regular-rational rows have been displayed.*

9. *H records componentwise* hsame *transport for the two stream families, difference rows, positivity row, dyadic ledger, schedule, regular-rational handoff, Real seal-facing row, continuation, provenance, and local naming rows.*
10. *C records the displayed* Cont *replay from each quotient-tail request through $A, B, \Delta A, \Delta B, D, T, Q, E, H, P, N$.*
11. *P is the* Pkg *provenance over* StolzCesaro$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyConvergenceMod MonotoneSubsequence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{StolzCesaro}^{\uparrow}}$ *row.*

The carrier has no coordinate for host limits, quotient streams, choice-selected subsequences, ambient completeness, real-closed field reasoning, or an unbounded sequence oracle.

Theorem 1439.2 (Stolz-Cesaro NameCert obligation sketch)**.** *The seed* NameCert$_{\text{StolzCesaro}^{\uparrow}}$ *surface for Definition 1439.1 consists of the following five obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$*
2. *denominator-step positivity: every admitted quotient-tail request reads the positive denominator step row ΔB before any ratio-difference row is classified;*
3. *ratio-difference classifier transport: the dyadic ratio-difference ledger D is stable under the componentwise* hsame *transport H and the displayed* Cont *replay C;*
4. *Cauchy-convergence handoff: the tail-window schedule T and dyadic ledger D supply the finite route consumed by the* CauchyConvergenceModulus$^{\uparrow}$*-style handoff before Q is read;*
5. *non-escape: no accepted consumer obtains a host limit, quotient stream, choice-selected subsequence, ambient completeness witness, or detached real-seal row outside $A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.StolzCesaroUp.StolzCesaroTasteGate single carrier alignment`

Proof. Project the carrier tuple $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$. Carrier admission is exactly the assertion that the local NameCert$_{\text{StolzCesaro}^{\uparrow}}$ row names this displayed finite packet.

The denominator side is read before the ratio classifier. The row ΔB supplies the positive difference evidence used to admit a quotient-tail request, and the dyadic row D records only the ratio-difference estimates over the displayed numerator and denominator step rows.

Now transport the classifier. The componentwise hsame row H and continuation replay C preserve the same stream windows, difference rows, positivity row, and dyadic ledger, so the classifier cannot move to a hidden quotient stream or host equality source.

Finally read the completion-facing handoff. The schedule T and ledger D feed the regular-rational row Q, and the Real seal-facing row E is reached only after that finite handoff has been displayed. The excluded objects have no coordinate in Z, so the five listed rows exhaust the seed obligation surface. □

Theorem 1439.3 (Stolz-Cesaro finite tail handoff)**.** *For every accepted* StolzCesaro$^{\uparrow}$ *packet $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$ and every displayed quotient-tail request, the Real-completion-facing read factors through the finite route*

$$A, B \longmapsto \Delta A, \Delta B \longmapsto D \longmapsto T \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

No consumer can read the RegSeqRat$^{\uparrow}$ *handoff or the* Real$^{\uparrow}$ *seal-facing row without also reading the displayed stream windows, numerator difference row, positive denominator step row, dyadic ratio-difference ledger, and finite tail-window schedule.*

Proof. Begin with Definition 1439.2. Its carrier-admission and positivity rows place $A, B, \Delta A, \Delta B$ before any ratio-difference classifier can be used.

The classifier then passes through the dyadic and schedule rows. The row D records the finite ratio-difference estimates, while T records which tail windows are admitted for the requested tolerance.

The completion-facing rows are downstream of that finite route. The regular-rational handoff Q and Real seal-facing row E are reached only after the scheduled dyadic estimate surface is displayed, and H, C, P, N preserve and name that same route. Hence the read cannot escape into host limits, quotient streams, choice-selected subsequences, or ambient completeness. □

Definition 1439.4 (Stolz-Cesaro finite-difference subcarrier)**.** *The finite-difference subcarrier of an accepted* $\mathsf{StolzCesaro}^{\uparrow}$ *packet is the displayed six-row packet*

$$Z_{\Delta} = (A, B, \Delta A, \Delta B, D, T)$$

cut out of Definition 1439.1. Its rows expose only the numerator $\mathsf{StreamName}^{\uparrow}$ *windows, denominator* $\mathsf{StreamName}^{\uparrow}$ *windows, displayed numerator differences, positive denominator-step differences,* $\mathsf{DyadicRatCore}^{\uparrow}$ *quotient-estimate ledger, and finite tail schedules. The subcarrier carries no* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, no* $\mathsf{Real}^{\uparrow}$ *seal-facing row, no host limit, no quotient stream, and no ambient completeness witness.*

Theorem 1439.5 (Stolz-Cesaro finite-difference NameCert obligations)**.** *For every accepted* $\mathsf{StolzCesaro}^{\uparrow}$ *packet* $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$*, the subcarrier* $Z_{\Delta} = (A, B, \Delta A, \Delta B, D, T)$ *of Definition 1439.4 carries the complete finite* $\mathsf{NameCert}_{\mathsf{StolzCesaro}^{\uparrow}}$ *surface consumed before any completion-facing route is exposed:*

1. *carrier admission identifies* Z_{Δ} *as the only finite-difference source admitted from the ambient packet* Z*;*
2. *denominator positivity is read from* ΔB *before any quotient estimate in* D *is classified;*
3. *dyadic ratio transport in* D *is stable under the componentwise* hsame *row* H*;*
4. *the finite schedule* T *names the inspected tail windows used by every tolerance request;*
5. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* Q *may be read only after* $A, B, \Delta A, \Delta B, D, T$ *have all been displayed;*
6. *the* $\mathsf{Real}^{\uparrow}$ *seal-facing row* E *is a terminal readback from* Q *and cannot expose a separate real-completion witness;*
7. Cont *replay through* C*,* Pkg *provenance through* P*, and the local naming row* N *preserve the same displayed subcarrier route.*

Proof. Start from the carrier admission in Definition 1439.2. Projection to the six displayed coordinates gives the finite-difference packet Z_{Δ}, and the excluded coordinates show that no host quotient stream or completion witness is part of this subcarrier.

The denominator and ratio rows are then forced locally. The positive denominator-step evidence lives in ΔB, while D records dyadic quotient estimates over the displayed numerator and denominator windows, so every ratio read factors through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ surfaces already named in the packet.

Next apply Definition 1439.3. It places the finite schedule T before the regular-rational handoff Q, and places the Real seal-facing row E after Q.

Finally replay the structural rows. The componentwise hsame transport H, displayed Cont replay C, Pkg provenance P, and local $\mathsf{NameCert}_{\mathsf{StolzCesaro}^\uparrow}$ row N preserve the same $A, B, \Delta A, \Delta B, D, T$ route through the sibling surfaces $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{CauchyConvergenceModulus}^\uparrow$, $\mathsf{MonotoneSubsequence}^\uparrow$, and $\mathsf{Real}^\uparrow$. □

Theorem 1439.6 (Stolz-Cesaro Cauchy tail handoff)**.** *Every completion-facing read from an accepted* $\mathsf{StolzCesaro}^\uparrow$ *packet factors through the finite-difference subcarrier*

$$Z_\Delta = (A, B, \Delta A, \Delta B, D, T)$$

before any $\mathsf{RegSeqRat}^\uparrow$ *handoff or* $\mathsf{Real}^\uparrow$ *seal-facing row is exposed. Equivalently, a read of Q or E is accepted only as the continuation of the displayed route from* $\mathsf{StreamName}^\uparrow$ *windows through* $\mathsf{DyadicRatCore}^\uparrow$ *quotient estimates and the* $\mathsf{CauchyConvergenceModulus}^\uparrow$ *tail schedule; it is rejected when it bypasses A, B, ΔA, ΔB, D, or T.*

Proof. Use Definition 1439.3 to order the completion-facing route. That theorem already places $A, B, \Delta A, \Delta B, D, T$ before Q and E.

Now identify this ordered prefix with Definition 1439.4. The prefix is exactly Z_Δ, so the Cauchy-modulus read has no access to a separate numerator stream, denominator stream, denominator positivity proof, quotient estimate ledger, or tail schedule.

It remains to account for stability under accepted replay. By Definition 1439.5, hsame, Cont, Pkg, and local naming preserve the same finite-difference route, while the sibling $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{CauchyConvergenceModulus}^\uparrow$, $\mathsf{MonotoneSubsequence}^\uparrow$, and $\mathsf{Real}^\uparrow$ surfaces supply only their displayed rows. Thus every accepted completion-facing read is mediated by Z_Δ before Q or E becomes visible. □

Theorem 1439.7 (Stolz-Cesaro RegSeqRat handoff)**.** *For every accepted* $\mathsf{StolzCesaro}^\uparrow$ *packet $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$, an accepted read of the row Q is exactly the* $\mathsf{RegSeqRat}^\uparrow$ *handoff induced by the finite-difference subcarrier*

$$Z_\Delta = (A, B, \Delta A, \Delta B, D, T).$$

Conversely, any accepted finite-difference packet exposing Z_Δ together with the displayed transport H, replay C, provenance P, and local name N repacks as the same Q handoff, and cannot supply a host limit, quotient stream, choice-selected subsequence, ambient completeness witness, or detached $\mathsf{Real}^\uparrow$ *seal.*

Proof. Start with an accepted read of Q. By Definition 1439.6, that read is admitted only after the displayed route through $A, B, \Delta A, \Delta B, D, T$ has been exposed, so the source of Q is the finite-difference subcarrier Z_Δ.

The exact obligation surface is supplied by Definition 1439.5. It identifies carrier admission, denominator positivity, dyadic ratio transport, the tail schedule, Cont replay, Pkg provenance, and local naming as the rows that preserve the finite-difference route before Q is read.

For the converse, begin with an accepted finite-difference packet exposing Z_Δ and the structural rows H, C, P, N. The same obligation theorem preserves the route under displayed transport and replay, while Definition 1439.6 places the $\mathsf{RegSeqRat}^\uparrow$ row immediately after that route and before any $\mathsf{Real}^\uparrow$ seal-facing row.

Thus the repacked packet reaches exactly the same Q handoff. The excluded host limit, quotient stream, choice-selected subsequence, ambient-completeness, and detached-seal objects have no coordinate in Z_Δ or in the structural rows that name and replay it. □

Theorem 1439.8 (Stolz-Cesaro Real-limit handoff)**.** *Every accepted* $\mathsf{StolzCesaro}^{\uparrow}$ *packet whose finite-difference subcarrier has produced the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed numerator and denominator windows, numerator and denominator difference rows, dyadic ratio ledger, finite tail schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. For*

$$Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N),$$

the seal-facing row E *is therefore a finite Real-limit handoff from the route*

$$A, B, \Delta A, \Delta B, D, T, Q, H, C, P, N,$$

not a host limit, quotient stream equality, countable-choice selector, trichotomy argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C, P, N)$ and its finite-difference subcarrier $Z_{\Delta} = (A, B, \Delta A, \Delta B, D, T)$. By Definition 1439.5, the subcarrier rows must be displayed before any completion-facing consumer can read Q or E.

Apply Definition 1439.6. It places the numerator and denominator windows, difference rows, dyadic ledger, and finite tail schedule before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal-facing row.

Now use Definition 1439.7. Once Q has been produced, the only admissible route to E is the same finite-difference route preserved by H, C, P, N. The structural rows transport, replay, source, and name the handoff; they do not add a separate limit witness.

The carrier has no coordinate for a host limit, quotient stream equality, countable choice, trichotomy, or ambient Real completeness. Hence the terminal Real-limit read is exactly the finite handoff stated above. □

Falsifiable prediction: An accepted $\mathsf{StolzCesaro}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal-facing row without the displayed numerator windows, denominator windows, numerator differences, positive denominator steps, dyadic ratio-difference ledger, and finite tail-window schedule refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, cauchyconvergencemodulus, monotonesubsequence, real: no listed sibling supplies the whole twelve-row packet $Z = (A, B, \Delta A, \Delta B, D, T, Q, E, H, C,$ because each sibling contributes only a stream, regular-rational, dyadic, convergence-schedule, monotone-source, or real-seal surface while this chapter binds the quotient-tail certificate with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StolzCesaro}^{\uparrow}}$.

Constructive story: A $\mathsf{StolzCesaro}^{\uparrow}$ packet is generated from finite numerator and denominator stream windows, numerator difference rows, positive denominator step rows, dyadic ratio-difference estimates, finite tail-window schedules, a RegSeqRat handoff, a Real seal-facing row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StolzCesaro}^{\uparrow}}$, seed)

Scope closed: Definition 1439.1, Definition 1439.2, and Definition 1439.3 seed the finite quotient-tail convergence certificate over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StolzCesaro}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the full classical Stolz-Cesaro theorem as an imported real-analysis principle, host limits, quotient streams, choice-selected subsequences, ambient completeness, or a Lean theorem.

Upgrade path: A formal route can encode the twelve-row carrier and prove the NameCert obligation sketch and finite tail handoff theorem.

Chapter 1440

A Concrete Naming Certificate for $\mathsf{BoundedRealLowerCut}^{\uparrow}$

Following Bishop and Bridges, bounded real lower-cut data are read as finite rational observations attached to a $\mathsf{Real}^{\uparrow}$ seal and an Archimedean bound, not as a host supremum or completed cut quotient. The packet consumes the real-name boundary of chapter 38, the located lower-cut discipline of chapter 404, the explicit rational-bound surface of chapter 430, and the nonconvergent observation boundary of chapter 1406. It is a constructive observation surface for bounded names rather than a Dedekind-real construction.

Definition 1440.1 (BoundedRealLowerCut carrier)**.** *A* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, A, D, S, O, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row whose lower observations are being exposed.*
2. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *rational-bound row attached to the same displayed real name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row used to read rational lower observations at finite precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that names which observations of R are inspected.*
5. *O is the lower-observation ledger: each row records a rational candidate, its dyadic comparison budget, the finite window it reads, and the evidence that the candidate lies below the sealed name as far as the displayed packet can observe.*
6. *B is the boundedness ledger tying every lower-observation row in O to the Archimedean bound row A.*
7. *H records the componentwise* hsame *transports for R, A, D, S, O, B and the structural rows.*
8. *C records the displayed* Cont *routes by which consumers replay the Real seal, Archimedean bound, dyadic comparison, window schedule, observation ledger, and boundedness ledger.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}$ *row.*

The carrier stores no host supremum, no Dedekind real construction, no trichotomy branch, no quotient real equality, no selected limit, and no ambient completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Theorem 1440.2 (BoundedRealLowerCut NameCert obligations)**.** *For an accepted* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *carrier* $L = (R, A, D, S, O, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier formation for the* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, lower-observation ledger, boundedness ledger, transports, continuations, provenance, and local naming data;*
2. *classifier stability, stating that replacement of the displayed rows through* H *and replay through* C *preserves the same bounded lower-cut observation packet;*
3. *lower-observation exactness, stating that each row of* O *is read from the same Real seal* R *through the finite windows* S *and dyadic comparison budget* D*;*
4. *boundedness handoff, stating that the Archimedean row* A *bounds every lower-observation row recorded in* O *through the finite ledger* B*;*
5. *seal non-escape, stating that no consumer reaches a supremum, Dedekind cut quotient, trichotomy branch, quotient real equality, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness principle from the packet.*

Proof. Carrier formation is the displayed tuple of Definition 1440.1. It contains exactly one Real seal row, one Archimedean bound row, one dyadic comparison row, one finite-window schedule, one lower-observation ledger, one boundedness ledger, and the structural transport, continuation, provenance, and naming rows.

Classifier stability is componentwise over H and C. A classified replacement may transport the displayed rows and replay their consumers, but it cannot introduce a second Real seal, a second bound, a hidden quotient of cuts, or an external completeness witness.

Lower-observation exactness is the role of O. Each observation row records the rational candidate, the dyadic comparison budget, and the finite window through which the same seal R is inspected. The packet therefore exposes the lower-cut face only as observed finite data.

The boundedness handoff is supplied by A and B. The Archimedean row gives the explicit rational-bound surface, and the ledger B ties every accepted lower observation to that surface without selecting a least upper bound or a supremum.

Seal non-escape follows from the row inventory. The provenance row mentions the sibling $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{SpeckerSequence}^{\uparrow}$ boundaries only as observation disciplines; no row carries a Dedekind construction, trichotomy branch, quotient equality, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 1440.3 (BoundedRealLowerCut L10 handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Real}^{\uparrow}$ *seal* R*, the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound* A*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* S*, the lower-observation ledger* O*, the boundedness ledger* B*, the transport row* H*, the continuation row* C*, the provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}$ *row* N *before using the bounded lower-cut surface.*

Proof. Project the accepted carrier. The Real-facing coordinate is R, and every lower-cut read is mediated by D, S, O while every boundedness read is mediated by A and B.

Next replay the sibling route. The $\mathsf{Real}^{\uparrow}$ chapter supplies the seal boundary, $\mathsf{LocatedCut}^{\uparrow}$ supplies the finite lower-cut observation discipline, $\mathsf{ArchimedeanReal}^{\uparrow}$ supplies the rational-bound surface, and $\mathsf{SpeckerSequence}^{\uparrow}$ marks the refusal to turn bounded observations into a general convergence or completeness principle.

Finally use H, C, P, N to close the handoff route. These rows record the only permitted transports, continuations, provenance, and local naming evidence, so an L10 consumer cannot bypass the finite observation surface by reading a host supremum, a Dedekind real, trichotomy, quotient real equality, selected limits, or Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealLowerCut}^{\uparrow}$ packet is generated from a displayed $\mathsf{Real}^{\uparrow}$ seal, an $\mathsf{ArchimedeanReal}^{\uparrow}$ bound row, a $\mathsf{DyadicRatCore}^{\uparrow}$ comparison row, a $\mathsf{StreamName}^{\uparrow}$ window schedule, a lower-observation ledger, a boundedness ledger, componentwise hsame transports, Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1440.1, Definition 1440.2, and Definition 1440.3 bind the finite bounded lower-cut observation surface for Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealLowerCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host supremum, a Dedekind real construction, trichotomy, quotient real equality, selected limits, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier formation, classifier stability, lower-observation exactness, boundedness handoff, and seal non-escape for the displayed packet.

Chapter 1441

A Concrete Naming Certificate for $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ packet records the boundary where a regular rational Cauchy name, an explicit convergence modulus, and finite locatedness evidence are all visible before the $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the finite windows of chapter 126, the regular rational source of chapter 301, the endpoint and radius ledger of chapter 291, the real-name seal of chapter 38, the located enclosure surface of **??**, and the explicit rate rows of chapter 332. The certificate keeps convergence and location as finite rows rather than hiding them inside a later BishopReal or compact-interval consumer.

Definition 1441.1 (Regular located Cauchy real carrier)**.** *A* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, D, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the finite observation windows used by the packet.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational entries are read only on the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger recording the rational centers, dyadic radii, and interval comparisons attached to those windows.*
4. *M is the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *rate row assigning each requested tolerance a finite threshold over the displayed windows.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row recording the finite interval-location evidence attached to the same source, endpoint, radius, and threshold rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular-source, dyadic-ledger, modulus, and located-enclosure rows have been displayed.*
7. *H records componentwise* hsame *transport for the stream windows, regular rational source, dyadic endpoint and radius ledger, convergence-modulus row, located enclosure, Real seal, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from each precision and location request through $S, Q, D, M, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for Dedekind construction of $\mathsf{Real}^{\uparrow}$*, quotient equality, trichotomy, open-cover compactness, countable choice, completed metric-space points, or an unbounded stream read.*

Theorem 1441.2 (Regular located Cauchy real NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ *surface for Definition 1454.1 consists exactly of carrier exposure for R, finite-window exposure through S, regular rational source exposure through Q, endpoint and radius exposure through D, convergence-rate exposure through M, located-enclosure exposure through L, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Each accepted request is classified only by the displayed StreamName window, the RegSeqRat source row, the dyadic endpoint and radius ledger, the convergence threshold, and the located enclosure row.*

```
Lean4 ✓  BEDC.Derived.RegularLocatedCauchyRealUp.TasteGate.RegularLocatedCauchyRealTasteGate
single carrier alignment
```

Proof. Project the carrier tuple $R = (S, Q, D, M, L, E, H, C, P, N)$. Each analytic coordinate is a displayed finite row, and the final coordinate names the same tuple as the local $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ surface.

Read the rows in their dependency order. A request first consumes the stream window S and regular rational source Q, then the dyadic endpoint and radius ledger D, then the convergence threshold row M, and only then the located enclosure L and terminal Real seal E.

The structural rows add no hidden analytic object. They transport, replay, source, and name the displayed route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface. □

Theorem 1441.3 (Regular located Cauchy real RealUp seal route)**.** *For every accepted* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *packet $R = (S, Q, D, M, L, E, H, C, P, N)$ and every displayed precision request, the Real-facing read factors through the finite route*

$$S \longmapsto Q \longmapsto D \longmapsto M \longmapsto L \longmapsto E \longmapsto (H, C, P, N).$$

No consumer can read the terminal $\mathsf{Real}^{\uparrow}$ *seal from R without also reading the same StreamName window family, RegSeqRat source row, DyadicRatCore endpoint and radius ledger, ModulusOfConvergence rate row, and LocatedReal enclosure row.*

Proof. Start with Definition 1454.2. It places the finite window row S before the regular source row Q and places the dyadic ledger D before the convergence modulus M and located enclosure L.

The Real seal is therefore not an independent coordinate. The row E is reached after S, Q, D, M, L are displayed, so a consumer omitting any of those rows has not followed the carrier route.

Finally read the structural rows. The rows H, C, P, N preserve, replay, source, and name the same route; they do not add a second route to $\mathsf{Real}^{\uparrow}$. Thus the terminal seal factors through the displayed finite certificate path. □

Theorem 1441.4 (Regular located Cauchy real modulus-location factorization)**.** *For every accepted* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *packet $R = (S, Q, D, M, L, E, H, C, P, N)$, every locatedness read factors through the common finite data*

$$S, \quad Q, \quad D, \quad M$$

before it reaches L, and every convergence-modulus read uses the same S, Q, D rows before it reaches M. Hence the modulus and locatedness surfaces share the StreamName windows, RegSeqRat source, and DyadicRatCore endpoint and radius ledger rather than presenting two unrelated certificates.

Proof. Use the carrier definition. The row M is a rate row over the windows, source, endpoints, and radii already named by S, Q, D; the row L is a located enclosure over the same displayed source and dyadic ledger, with the rate row fixing the relevant tail threshold.

Now compare the two reads. The convergence side stops at M after S, Q, D, while the locatedness side continues through M to L on the same finite rows.

The componentwise hsame and Cont rows preserve that shared route, and the Pkg and local NameCert rows name it. Thus any accepted consumer sees one factored certificate surface, not a separated modulus certificate and a separated location certificate. □

Theorem 1441.5 (Regular located Cauchy real scoped dependency route)**.** *Every accepted* RegularLocatedCauchyR *read factors through exactly the displayed* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *readback row,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* ModulusOfConvergence$^{\uparrow}$ *location row,* LocatedReal$^{\uparrow}$ *locatedness row, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$ *row. No quotient stream equality, selected representative, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness coordinate is in scope for such a read.*

Proof. Use Definition 1454.2. It exposes the carrier as the tuple $R = (S, Q, D, M, L, E, H, C, P, N)$ and makes the stream windows, regular rational source, dyadic ledger, convergence row, locatedness row, and Real seal explicit.

Apply Definition 1454.3 to the endpoint-facing side. The terminal Real$^{\uparrow}$ read is downstream of the same S, Q, D, M, L rows, so it cannot be reached by a separate representative-selection path.

Apply Definition 1454.4 to the locatedness side. The modulus and locatedness reads share S, Q, D, M before reaching L, and the structural rows only transport, replay, source, and name that route.

The excluded principles have no coordinate in the displayed carrier. Hence the scoped route is exactly the finite dependency route named above. □

Falsifiable prediction: An accepted RegularLocatedCauchyReal$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ seal or a locatedness enclosure without the displayed StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ source, DyadicRatCore$^{\uparrow}$ endpoint and radius ledger, and ModulusOfConvergence$^{\uparrow}$ rate row refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, locatedreal, modulusofconvergence: no listed sibling supplies the whole ten-row packet $R = (S, Q, D, M, L, E, H, C, P, N)$, because each sibling contributes only a window, regular-source, dyadic-ledger, Real-seal, located-enclosure, or rate surface while this chapter binds all six analytic faces with transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$**.**

Constructive story: A RegularLocatedCauchyReal$^{\uparrow}$ packet is generated from finite StreamName$^{\uparrow}$ windows, a RegSeqRat$^{\uparrow}$ source row, a DyadicRatCore$^{\uparrow}$ endpoint and radius ledger, a ModulusOfConvergence$^{\uparrow}$ rate row, a LocatedReal$^{\uparrow}$ enclosure row, a terminal Real$^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1454.1, Definition 1454.2, Definition 1454.3, Definition 1454.4, and Definition 1454.5 bind the regular located Cauchy real-name route over StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, ModulusOfConvergence$^{\uparrow}$, LocatedReal$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularLocatedCauchyRealUp.TasteGate.RegularLocatedCauchyRealTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not claim Dedekind construction of $\mathrm{Real}^{\uparrow}$, quotient equality, trichotomy, open-cover compactness, countable choice, completeness, or completed metric-space points.

Upgrade path: Public closure requires a downstream public-object surface that reuses the scoped dependency route without adding quotient equality, selected representatives, countable choice, or ambient completeness.

Chapter 1442

A Concrete Naming Certificate for $\mathsf{SequentialContinuity}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{SequentialContinuity}^{\uparrow}$ packet records continuity along certified convergent sequences as a finite BEDC route. It reads the pointwise metric-continuity surface of chapter 59, the regular rational windows of chapter 301, the inspected stream schedules of chapter 126, the real-name seal of chapter 38, and the $\mathsf{Met}^{\uparrow}$ rows of chapter 57. The packet supplies a Real and Completion frontier surface for sequence consumers without importing host sequential continuity, function extensionality, choice, quotient equality, or ambient completeness.

Definition 1442.1 (Sequential continuity carrier)**.** *A* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$$

with the following displayed rows:

1. *M_X and M_Y are the source and target* $\mathsf{Met}^{\uparrow}$ *carrier rows with their visible classifiers and distance ledgers.*
2. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row for a displayed map from the source metric carrier to the target metric carrier.*
3. *S is the source sequence surface, exposed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *L is the inspected convergence ledger for S, listing the finite thresholds actually requested by the packet.*
5. *W_X and W_Y are the source and target finite stream windows read from S and from the displayed images of those windows under F.*
6. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows attached to the inspected source and target windows.*
7. *E_X and E_Y are the* $\mathsf{Real}^{\uparrow}$ *seal rows reached after the corresponding regular-rational handoffs have been displayed.*
8. *H stores componentwise* hsame *transport for the metric rows, the continuous-map row, the sequence row, the convergence ledger, both window families, both regular handoffs, both Real seals, continuation, provenance, and local naming.*

9. *C stores displayed* Cont *replay from S, L to W_X, from W_X, F to W_Y, from W_X to R_X, E_X, and from W_Y to R_Y, E_Y.*
10. *P is the* Pkg *provenance over* $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an external sequence function, quotient stream equality, function extensionality, countable choice, or host sequential-continuity theorem.

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityTasteGate single carrier alignment`

Theorem 1442.2 (Sequential continuity NameCert obligations)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed fifteen-row packet;*
2. *source and target metric-row stability through M_X, M_Y, H, C;*
3. *pointwise-continuity use only through the displayed* $\mathsf{Continuous}^{\uparrow}$ *row F;*
4. *inspected-threshold convergence coverage through the finite ledger L;*
5. *source and target sequence handoff through W_X, W_Y, R_X, R_Y before either* $\mathsf{Real}^{\uparrow}$ *seal is consumed;*
6. *non-escape from host sequential continuity, function extensionality, countable choice, quotient equality, and ambient Real completeness.*

Proof. Project the carrier of Definition 1466.1. The metric rows, continuous-map row, source sequence row, convergence ledger, window rows, regular-rational handoffs, Real seals, and structural rows are the only coordinates visible to the classifier.

The metric and continuity faces are already displayed. Source and target stability reads through M_X, M_Y, H, C, while pointwise continuity is consumed only through F and the finite windows named by the packet.

The sequential face is finite. The ledger L records the inspected convergence thresholds, W_X displays the corresponding source windows, and W_Y displays the image windows obtained by the recorded continuation route.

Finally read the endpoint route. The source side reaches E_X through R_X, and the target side reaches E_Y through R_Y. Since the excluded principles are not coordinates of Q, the six listed rows exhaust the local NameCert obligation surface. □

Theorem 1442.3 (Sequential continuity Cauchy handoff)**.** *Every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *packet hands both the source sequence windows and the image sequence windows to* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Equivalently, every seal-facing route factors through*

$$S, L \longmapsto W_X \longmapsto R_X \longmapsto E_X \qquad \text{and} \qquad W_X, F \longmapsto W_Y \longmapsto R_Y \longmapsto E_Y$$

together with M_X, M_Y, H, C, P, N.

Proof. Begin with the inspected source route. The continuation row records that W_X is read from the displayed sequence surface S using only the thresholds listed in L.

Next form the image route. The row F is the displayed $\mathsf{Continuous}^{\uparrow}$ pointwise-modulus surface, and C records the replay from the inspected source windows to the target windows W_Y.

Both sides then pass through regular-rational readback. The rows R_X and R_Y are the only regular handoffs attached to the source and target windows, and the seal rows E_X and E_Y are downstream from those handoffs.

Thus a consumer that tries to use either $\mathsf{Real}^{\uparrow}$ seal without the matching $\mathsf{RegSeqRat}^{\uparrow}$ handoff leaves the accepted $\mathsf{SequentialContinuity}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityCauchyHandoff`

Theorem 1442.4 (Sequential continuity metric-row stability)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier*

$$Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N),$$

the source and target metric rows used by the local classifier are exactly the displayed $\mathsf{Met}^{\uparrow}$ *rows* M_X *and* M_Y*, stabilized by the same componentwise* hsame *transport* H *and displayed* Cont *replay* C *carried by* Q*. No metric comparison may be supplied by an external topology, host equality of points, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityPacket metric row stability`

Proof. Project the accepted carrier. The only metric coordinates visible in the packet are M_X and M_Y, and the only structural coordinates that can move them between accepted packets are H and C.

The provenance and naming rows do not add a second metric surface. They record that the displayed rows belong to the same $\mathsf{SequentialContinuity}^{\uparrow}$ packet over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Thus every metric read used by the classifier is stable exactly along M_X, M_Y, H, C, P, N. The excluded external topology, host equality, quotient, and completeness data are not coordinates of Q. □

Theorem 1442.5 (Sequential continuity pointwise consumption)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier* $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$*, the pointwise-continuity information consumed by the sequence route is only the displayed* $\mathsf{Continuous}^{\uparrow}$ *row* F *together with the source metric row* M_X*, target metric row* M_Y*, finite windows* W_X, W_Y*, transport* H*, replay* C*, provenance* P*, and local naming* N*. The packet exports no host sequential-continuity theorem and no function-extensionality coordinate.*

Proof. Read the pointwise face of the carrier. The row F is the unique $\mathsf{Continuous}^{\uparrow}$ coordinate, and it is stored beside the source and target metric rows M_X, M_Y.

The sequence route inspects only finite windows. The continuation row sends the displayed source windows W_X through F to the target windows W_Y, and the transport row preserves that same route under componentwise hsame.

Finally, P and N source and name the same packet. Since neither a host sequential-continuity theorem nor a function-extensionality principle appears among the coordinates of Q, every pointwise-continuity use is consumed through the displayed rows named in the statement. □

Theorem 1442.6 (Sequential continuity inspected-threshold coverage)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier* $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$*, each threshold used by a Real-facing or regular-Cauchy consumer is first listed in the inspected convergence ledger* L*, then*

read through the source window W_X, the image window W_Y, the regular handoff rows R_X, R_Y, and only then through the seal rows E_X, E_Y. The coverage route is exhausted by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the same carrier.*

Proof. Project the carrier and read the inspected ledger. The row L is the only coordinate that lists convergence thresholds requested by the packet.

The source schedule then opens W_X from S at the listed thresholds. The pointwise row F and replay row C produce the corresponding image windows W_Y without adding a detached sequence object.

The regular and Real-facing reads are downstream from those windows. The rows R_X, R_Y repack the inspected windows as regular rational handoffs, and E_X, E_Y are consumed only after those handoffs.

The structural rows H, C, P, N preserve, replay, source, and name exactly this route. Therefore the inspected thresholds cover the whole consumer path and no unlisted threshold, quotient stream, selected limit, choice row, or ambient completeness datum can enter the packet. □

The mature treatment of $\mathsf{SequentialContinuity}^{\uparrow}$ is the Real and Completion sequence-consumer route obtained by reading the Cauchy handoff, metric-row stability, pointwise consumption, and inspected-threshold coverage together. A consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or a completion-facing sequence row must pass through the displayed finite windows, the regular-rational handoff rows, and the same metric and continuity coordinates already inspected by the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a displayed pointwise-continuity row, inspected sequence windows, a finite convergence ledger, source and target RegSeqRat handoffs, RealUp seal rows, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1466.1, Definition 1466.2, Definition 1466.3, Definition 1466.4, Definition 1466.5, and Definition 1466.6 bind the carrier, NameCert obligations, Cauchy handoff, metric-row stability, pointwise consumption, and inspected-threshold coverage for certified sequence consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SequentialContinuityUp.SequentialContinuityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim host sequential continuity, function extensionality, countable choice, quotient stream equality, semantic completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: No further paper-side closure grade remains; bridge verification stays on the independent formal axis.

Chapter 1443

A Concrete Naming Certificate for $\mathsf{SubsequenceLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SubsequenceLimit}^{\uparrow}$ packet records the finite consumer boundary for reading a displayed subsequence limit from already certified real sequence-limit data. It consumes the sequence-limit surface of chapter 1413, the cofinal selector route of chapter 405, the regular Cauchy subsequence boundary of chapter 381, and the real-name seal of chapter 38. Its purpose is to let downstream Bolzano–Weierstrass and sequence-convergence arguments read a selected limit surface without importing arbitrary subsequence choice, host sequence equality, quotient stream equality, or ambient completeness.

Definition 1443.1 (SubsequenceLimit carrier)**.** *A* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, \Phi, W, R, D, E, H, C, P, N)$$

with the following visible rows:

1. *L is the parent* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row, containing the displayed source sequence, candidate limit, tail-window schedule, dyadic tolerance ledger, and convergence classifier exposed by chapter 1413;*
2. *Φ is the accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row, including the displayed unary selector, monotonicity witnesses, cofinal witnesses, selected windows, and regular rational handoff rows;*
3. *W records the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows from the parent sequence and from the subsequence readback surface;*
4. *R records the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *observations and the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *route through which the subsequence read is displayed;*
5. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the selected subsequence windows with the parent limit row;*
6. *E is the shared* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the parent limit row, cofinal selector row, selected windows, regular rational handoff, and dyadic tolerance ledger have been displayed;*
7. *H records componentwise* hsame *transports for the parent limit row, selector row, selected windows, regular rational observations, dyadic ledger, real seal, and local naming row;*
8. *C records the displayed* Cont *routes by which consumers inspect the same rows in their certificate order;*

9. *P is the* Pkg *provenance over the* $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and naming dependencies;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$ *row.*

The carrier stores a finite subsequence-limit consumer interface. It contains no host-level infinite subsequence, no quotient stream representative, no host sequence equality, no independent real-completeness object, and no choice principle for selecting further subsequences.

Theorem 1443.2 (SubsequenceLimit NameCert obligations)**.** *Every accepted* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier of Definition 1467.1 carries five finite NameCert obligation rows:*

1. *carrier admission for the parent sequence-limit row, cofinal selector row, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff rows, dyadic tolerance ledger, shared* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, continuation routes, package provenance, and local naming row;*

2. *selector-to-limit alignment, stating that every subsequence window read by Φ is one of the displayed parent sequence windows named by L and compared against the same candidate limit row;*

3. *tolerance preservation, stating that the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D is the common tolerance ledger used by both the parent sequence-limit classifier and the selected subsequence read;*

4. *real-seal handoff discipline, stating that the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed only after the parent limit row, cofinal selector, selected windows, regular rational route, and tolerance ledger have been displayed;*

5. *non-escape, stating that the packet exports no Bolzano–Weierstrass conclusion, arbitrary subsequence choice, host sequence equality, quotient stream equality, ambient real completeness, or bridge-checked external schema.*

Lean4 ✓ `BEDC.Derived.SubsequenceLimitUp.SubsequenceLimitTasteGate single carrier alignment`

Proof. Carrier admission is the tuple in Definition 1467.1. Each row needed by a consumer is a visible BHist coordinate, and the local naming row is attached to that same finite packet.

Selector-to-limit alignment is read from the pair (L, Φ). The parent row supplies the displayed sequence-limit surface, while the selector row supplies only selected finite windows and their regular rational handoff data. Thus a subsequence read remains a read of the parent displayed sequence.

Tolerance preservation is the projection to D. The selected subsequence windows are compared with the parent candidate limit using the same dyadic tolerance ledger, so the packet has no second tolerance scale that could change the limit claim.

The real-seal handoff follows the order of the rows L, Φ, W, R, D, E. A consumer reaches the shared $\mathsf{Real}^{\uparrow}$ seal only after the parent limit data, selector route, selected windows, regular rational observations, and dyadic ledger are already visible.

The non-escape row is an inventory check. None of the coordinates stores a Bolzano–Weierstrass selection, a host-level subsequence, host sequence equality, quotient stream equality, ambient real completeness, or an external bridge schema, so the local NameCert cannot export those objects. □

Theorem 1443.3 (SubsequenceLimit Real seal handoff)**.** *Let $Q = (L, \Phi, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer read from Q factors through the parent* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row L, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row Φ, the selected* $\mathsf{StreamName}^{\uparrow}$

windows W, the selected $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *route R, and the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the seal row E is exposed.*

Proof. Start from a $\mathsf{Real}^{\uparrow}$-facing consumer read. The carrier places the seal coordinate E after the parent sequence-limit row, selector row, selected windows, regular rational handoff, and dyadic ledger.

Apply the selector-to-limit and tolerance-preservation rows of Definition 1467.2. They force the selected subsequence windows to remain parent sequence windows and force their limit comparison to use the same dyadic tolerance ledger as the parent limit classifier.

It remains to identify the public read route. The displayed hsame transports and Cont routes preserve the order through L, Φ, W, R, D, E, while the Pkg and local NameCert rows account for provenance. Hence every seal-facing read factors through the listed finite handoff surface. □

Theorem 1443.4 (SubsequenceLimit choice-free boundary)**.** *The* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier is a finite BEDC consumer boundary for selected sequence-limit reads. It preserves the parent sequence-limit row, cofinal selector row, selected regular rational observations, dyadic tolerance ledger, shared real seal, displayed transports, continuation routes, package provenance, and local naming row, but it does not construct a Bolzano–Weierstrass subsequence, choose an arbitrary infinite subsequence, identify host sequences, identify quotient streams, prove real completeness, or provide a checked standard bridge.*

Proof. The positive content is exactly the carrier inventory of Definition 1467.1. The rows $L, \Phi, W, R, D, E, H, C, P$, provide the parent limit data, selector route, selected observations, tolerance ledger, seal, transport, continuation, provenance, and local naming surface.

The consumer route is fixed by Definition 1467.2 and Definition 1467.3. Every public real-facing read passes through the finite selector-to-limit handoff rather than through an unrecorded host object.

The excluded principles are absent as coordinates. Since no row stores a Bolzano–Weierstrass construction, arbitrary subsequence choice, host equality, quotient equality, completeness proof, or bridge-checked schema, the packet cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{SubsequenceLimit}^{\uparrow}$ packet is generated from a parent sequence-limit row, a cofinal selector row, selected StreamName windows, RegSeqRat and RegularCauchySubsequence handoff rows, a shared DyadicRatCore tolerance ledger, a shared RealUp seal, componentwise transport, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$, seed)

Scope closed: Definition 1467.1, Definition 1467.2, Definition 1467.3, and Definition 1467.4 declare the finite subsequence-limit consumer boundary for selected RealUp sequence-limit reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Bolzano–Weierstrass, arbitrary subsequence choice, host sequence equality, quotient stream equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector-to-limit alignment, tolerance preservation, real-seal handoff, and choice-free refusal for the same finite packet.

Chapter 1444

A Concrete Naming Certificate for $\mathsf{RegularCauchyReciprocal}^{\uparrow}$

Bishop and Bridges-style constructive reciprocal data begin from a regular Cauchy rational name together with a displayed apartness window that keeps the inspected tail away from zero. The $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ packet names the finite interface needed to read that reciprocal surface. It refines the budget-only surface of chapter 703 by binding the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the finite windows of chapter 126, the dyadic endpoint and radius arithmetic of chapter 291, and the terminal seal of chapter 38 into one local NameCert carrier. The chapter records only the finite reciprocal-read surface; it does not construct a total reciprocal operation on $\mathsf{Real}^{\uparrow}$.

Definition 1444.1 (Regular Cauchy reciprocal carrier)**.** *A* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, A, M, W, D, B, T, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational entries are being inspected.*
2. *A is the apartness-window row displaying a rational lower bound that keeps the inspected tail of Q away from zero.*
3. *M is the finite modulus row selecting the precision at which the reciprocal read is requested.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened from Q under A and M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and comparison ledger read from W.*
6. *B is the sibling* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row that records the reciprocal-window budget over the same Q, A, M, W, D rows.*
7. *T is the regular rational readback row naming the reciprocal candidate as a* $\mathsf{RegSeqRat}^{\uparrow}$ *surface.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed source, apartness, modulus, window, dyadic, budget, and readback rows have been consumed.*
9. *H records componentwise* hsame *transport for $Q, A, M, W, D, B, T, E, C, P, N$.*
10. *C records the displayed* Cont *replay for source inspection, apartness exposure, finite-window selection, dyadic reciprocal arithmetic, budget handoff, regular readback, and Real seal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$ *row.*
The local classifier compares accepted carriers only through the displayed rows $Q, A, M, W, D, B, T, E, H, C, P, N$.

Theorem 1444.2 (Regular Cauchy reciprocal apartness window)**.** *Every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier exposes the apartness row A before any reciprocal dyadic row D, budget row B, regular readback row T, or* $\mathsf{Real}^{\uparrow}$ *seal row E can be consumed. A route that omits A remains a boundary attempt and cannot export a reciprocal candidate.*

Proof. Project the carrier of Definition 1471.1. The source-side rows are Q, A, M, W, and the reciprocal-facing rows begin only after the apartness and modulus rows have selected a finite window.

The dyadic ledger D reads endpoint, radius, and comparison data from W. Since the reciprocal denominator is controlled by the displayed lower bound in A, the dyadic reciprocal row has no carrier route that bypasses A.

The budget row B, readback row T, and seal row E are downstream of that dyadic row. Therefore omitting A leaves no displayed route from the source to a reciprocal candidate; the local classifier records such a route as boundary evidence. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalCarrier apartness window`

Theorem 1444.3 (Regular Cauchy reciprocal readback route)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the reciprocal readback row T factors through*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T.$$

The terminal $\mathsf{Real}^{\uparrow}$ *seal E is reached only after this finite route has been displayed.*

Proof. Begin with the apartness-window theorem. It places A before any dyadic reciprocal arithmetic, so the route to T cannot start from Q alone.

Next open the finite window row W by using the modulus row M. The dyadic ledger D is a read of that window, and the sibling budget row B is the same reciprocal-window discipline named by chapter 703.

Finally project T and then E from the carrier. The readback row names the reciprocal candidate as regular rational data, while the seal row is terminal and adds no second route around the displayed finite chain. □

Theorem 1444.4 (Regular Cauchy reciprocal classifier stability)**.** *Let*

$$R = (Q, A, M, W, D, B, T, E, H, C, P, N) \quad \text{and} \quad R' = (Q', A', M', W', D', B', T', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carriers. If the displayed transport row H supplies componentwise* hsame *comparisons from each row of R to the corresponding row of R', then the local classifier preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, apartness row, modulus row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row, reciprocal readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalClassifierStability`

Proof. Project the transport coordinate H from the carrier. Its comparisons are componentwise, so the source rows Q, Q', apartness rows A, A', modulus rows M, M', finite windows W, W', dyadic ledgers D, D', budget rows B, B', readback rows T, T', and seal rows E, E' are matched before any classifier conclusion is read.

Apply the apartness-window theorem to both carriers. Since the transported apartness rows remain the first reciprocal-facing rows, the reciprocal denominator witness is preserved by the classifier.

Apply the readback-route theorem to both carriers. The transported route from Q through B to T matches the transported route from Q' through B' to T', and the terminal seal rows are reached only after those finite routes.

Thus classifier equality cannot forget, reorder, or replace any analytic row of the reciprocal packet. It transports exactly the displayed source, apartness, modulus, window, dyadic, budget, readback, and Real seal coordinates. □

Theorem 1444.5 (Regular Cauchy reciprocal terminal seal non-totality)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row E of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier cannot be reread as a total reciprocal operation on all* $\mathsf{Real}^{\uparrow}$ *names. Any route from the seal back to a reciprocal candidate must still factor through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row Q, apartness row A, modulus row M, finite* $\mathsf{StreamName}^{\uparrow}$ *window W, dyadic ledger D,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B, and regular readback row T.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalTerminalSealNonTotality`

Proof. Inspect the carrier definition. The seal coordinate E is listed after the source, apartness, modulus, window, dyadic, budget, and readback rows, and the classifier reads it as one coordinate of the twelve-row packet.

The readback-route theorem states that E is reached only after the finite route through T has been displayed. Therefore the seal does not introduce a second route from arbitrary $\mathsf{Real}^{\uparrow}$ names to reciprocal data.

The apartness-window theorem keeps A mandatory for every reciprocal-facing row. Since a total reciprocal on all $\mathsf{Real}^{\uparrow}$ names would erase the need for such a displayed apartness row, it is not present in the packet.

Consequently the terminal seal records only the end of one finite regular-Cauchy reciprocal read. It cannot be exported as a total reciprocal operation or as a boundary bypass around the apartness obligation. □

Theorem 1444.6 (Regular Cauchy reciprocal denominator apartness)**.** *For every accepted* RegularCauchyReciproca *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the dyadic reciprocal row D and the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B are admissible only after the apartness row A displays a positive lower bound for the inspected* $\mathsf{RegSeqRat}^{\uparrow}$ *tail named by Q, M, W. A consumer that reads D or B without that positive lower-bound display has not reached a reciprocal row of the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalDenominatorApartness`

Proof. Project Definition 1471.1. The source row Q, apartness row A, modulus row M, and finite window row W precede the dyadic reciprocal row D and budget row B in the carrier route.

Apply Definition 1471.2. It states that A is exposed before D, B, T, E can be consumed, so the denominator-side condition is part of the carrier admission rather than a later Real-facing consequence.

Finally read Definition 1471.3. The route from Q to T passes through A, M, W, D, B in that order, and therefore D and B are reciprocal rows only when the inspected regular tail has the displayed positive lower bound supplied by A. □

Theorem 1444.7 (Regular Cauchy reciprocal nonzero-denominator window)**.** *Any accepted* RegularCauchyReciproc *packet whose dyadic inverse budget is read must expose a nonzero-denominator window consisting of the* RegSeqRat$^{\uparrow}$ *source row Q, apartness row A, finite modulus row M,* StreamName$^{\uparrow}$ *window W, and* DyadicRatCore$^{\uparrow}$ *row D. Without all five rows the packet has not reached the* CauchyInverseBudget$^{\uparrow}$ *row B.*

Proof. Project the carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ from Definition 1471.1. The only source, apartness, modulus, window, and dyadic rows available to the reciprocal budget are Q, A, M, W, D.

Apply Definition 1471.6. It places the positive lower-bound display in A before the dyadic reciprocal row D and budget row B can be consumed.

Then apply Definition 1471.3. The route to the reciprocal readback passes through Q, A, M, W, D, B, so a consumer that omits any one of Q, A, M, W, D remains before the CauchyInverseBudget$^{\uparrow}$ row rather than at the dyadic inverse budget. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalNonzeroDenominatorWindo`

Theorem 1444.8 (Regular Cauchy reciprocal tail stability)**.** *Let $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ be an accepted* RegularCauchyReciprocal$^{\uparrow}$ *carrier. If a later inspected* StreamName$^{\uparrow}$ *window preserves the* RegSeqRat$^{\uparrow}$ *source row Q together with the apartness row A, modulus row M, finite window row W, and dyadic reciprocal row D, then the same reciprocal readback row T and terminal* Real$^{\uparrow}$ *seal row E transport through H, C, P, N. Thus shifting to a later tail window cannot change the reciprocal readback unless it changes one of A, M, W, D.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalTailStability`

Proof. Use Definition 1471.6. A later window that preserves A, M, W, D keeps the same displayed positive lower-bound and dyadic reciprocal data for the inspected regular tail.

Apply Definition 1471.3. Since the route to T passes through Q, A, M, W, D, B, preserving the apartness, window, and dyadic rows preserves the reciprocal readback route once the same budget row is read.

Apply Definition 1471.4. The transport row H compares corresponding carrier rows componentwise, while C, P, N replay, source, and name the same packet. The terminal seal E is therefore transported with T rather than recomputed from a new denominator condition. □

Theorem 1444.9 (Regular Cauchy reciprocal window composition)**.** *Let $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ be an accepted* RegularCauchyReciprocal$^{\uparrow}$ *carrier. Suppose a later finite* StreamName$^{\uparrow}$ *window preserves the displayed rows Q, A, M, W, D and then reads the same* CauchyInverseBudget$^{\uparrow}$ *row B. The composed finite route still reaches the same regular rational readback row T and terminal* Real$^{\uparrow}$ *seal row E through*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T \longmapsto E.$$

Thus window composition changes neither the reciprocal readback nor the Real seal unless it changes one of the displayed source, apartness, modulus, window, dyadic, or budget rows.

Proof. Apply Definition 1471.6. It keeps the apartness row A in front of every dyadic reciprocal and budget read, so the later window cannot compose into a reciprocal route while dropping the positive lower-bound display.

Use Definition 1471.7. The preserved rows Q, A, M, W, D are exactly the nonzero-denominator window needed before the same budget row B can be read.

Apply Definition 1471.3. Once the same budget row is present, the route to T is the displayed chain through Q, A, M, W, D, B, and the seal E is terminal after that chain.

Finally use Definition 1471.8. Since the later window preserves the source, apartness, modulus, window, and dyadic rows and reads the same budget row, the transported packet reaches the same T and E rather than recomputing a new reciprocal surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalWindowComposition`

Theorem 1444.10 (Regular Cauchy reciprocal window denominator stability)**.** *Let $R = (Q, A, M, W, D, B, T, E, H$ be an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier. Suppose a later finite* $\mathsf{StreamName}^{\uparrow}$ *window preserves the displayed rows Q, A, M, W, D and reads the same* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row B. Then the denominator apartness witness recorded by A remains visible before the reciprocal readback row T and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are consumed. In particular, the later window cannot treat B, T, E as a hidden choice of denominator data.*

Proof. Apply Definition 1471.6. The dyadic reciprocal row D and budget row B are admissible only after A displays the positive lower-bound witness for the inspected regular tail.

Use Definition 1471.7. Since the later window preserves Q, A, M, W, D, it preserves exactly the nonzero-denominator surface that must be present before the same $\mathsf{CauchyInverseBudget}^{\uparrow}$ row is read.

Apply Definition 1471.8. Preserving the source, apartness, modulus, window, and dyadic rows transports the same readback row T and terminal seal row E through the carrier rows H, C, P, N.

Finally use Definition 1471.9. The composed route reaches T and then E only through the displayed chain Q, A, M, W, D, B, T, E, so the denominator apartness witness remains visible throughout the reciprocal handoff rather than becoming hidden choice data. □

Theorem 1444.11 (Regular Cauchy reciprocal located apartness window)**.** *Any* $\mathsf{Real}^{\uparrow}$*-seal or budget-facing consumer of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row Q, apartness row A, modulus row M,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row D, and* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B before reading the reciprocal readback row T or terminal seal row E. The route exports no total reciprocal on arbitrary* $\mathsf{Real}^{\uparrow}$ *names, trichotomy, quotient stream equality, selected limit, or ambient completeness.*

Proof. Project the accepted carrier. The only reciprocal-facing analytic rows before readback are the source, apartness, modulus, window, dyadic comparison, and budget rows Q, A, M, W, D, B.

Apply Definition 1471.2, Definition 1471.6, and Definition 1471.7. These place the located apartness display in A before the dyadic comparison row and identify Q, A, M, W, D as the nonzero-denominator window required before B is read.

Next use Definition 1471.9. It preserves the same route through Q, A, M, W, D, B, T, E under later finite windows, so reading T cannot detach the reciprocal candidate from its displayed denominator witness.

Finally apply Definition 1471.13. The terminal seal E is offered only after the source, apartness, modulus, window, dyadic, budget, and readback rows have been displayed. Since the carrier has no

coordinate for total reciprocals, trichotomy, quotient stream equality, selected limits, or ambient completeness, those principles cannot be exported by either the budget-facing route or the Real-seal route. □

Theorem 1444.12 (Regular Cauchy reciprocal scoped denominator apartness)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, the denominator-apartness witness is scoped exactly by the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, apartness row* A*, finite modulus row* M*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. It exports no total* $\mathsf{Real}^{\uparrow}$ *reciprocal, quotient stream, selected limit, or host equality.*

Proof. Project the carrier of Definition 1471.1. The only denominator-facing rows before reciprocal readback are Q, A, M, W, D, B, and the only structural rows that can transport or replay them are H, C, P, N.

Apply Definition 1471.2 and Definition 1471.6. These place the apartness row A before every dyadic reciprocal, budget, readback, and seal coordinate, and they make the positive lower-bound display part of carrier admission.

Use Definition 1471.7 and Definition 1471.10. The first theorem identifies Q, A, M, W, D as the nonzero-denominator window required before B is read, while the second preserves that witness under later finite windows.

The remaining rows H, C, P, N carry only componentwise transport, replay, provenance, and local naming for the same finite surface. Thus the scoped apartness witness is exactly the displayed BEDC primitive surface $Q, A, M, W, D, B, H, C, P, N$, with no exported total reciprocal on $\mathsf{Real}^{\uparrow}$, quotient stream, selected limit, or host equality. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalScopedDenominatorApartn`

Theorem 1444.13 (Regular Cauchy reciprocal Real seal handoff)**.** *Let* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier. Any reciprocal consumer that is offered the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *must first be offered the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, apartness row* A*, modulus row* M*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row* B*, and reciprocal readback row* T*. Thus the handoff from finite reciprocal-read data to the L10* $\mathsf{Real}^{\uparrow}$ *boundary is the displayed route*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T \longmapsto E$$

and not a direct operation from arbitrary $\mathsf{Real}^{\uparrow}$ *names to reciprocals.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalRealSealHandoff`

Proof. Apply Definition 1471.2. It places the apartness row A before every reciprocal-facing coordinate, so a consumer cannot treat the terminal seal as available before the denominator witness is displayed.

Next use Definition 1471.3. The route to the readback row T factors through Q, A, M, W, D, B, and the seal row E is terminal only after that finite route has been exposed.

Apply Definition 1471.6 and Definition 1471.7. These identify the source, apartness, modulus, window, and dyadic rows as the nonzero-denominator surface required before the budget row B can be consumed.

Finally use Definition 1471.8 together with Definition 1471.9. Transport to later windows and composition of finite windows preserve the same T and E only when the displayed rows

Q, A, M, W, D, B are preserved. The handoff therefore exports neither a total reciprocal, quotient real equality, trichotomy, selected limit, nor ambient completeness principle; it exports only the finite route from reciprocal readback data to the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 1444.14 (Regular Cauchy reciprocal NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$ *surface consists exactly of carrier exposure for* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, apartness-window exposure through* A*, finite-window exposure through* W*, dyadic reciprocal arithmetic through* D*, sibling budget exposure through* B*, regular readback through* T*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalNameCertObligations`

Proof. Carrier exposure is Definition 1471.1. It supplies the full row list and states that the classifier reads only those rows.

The analytic route is given by Definition 1471.2 and Definition 1471.3. Together they force every reciprocal read to pass through the source, apartness, modulus, finite window, dyadic arithmetic, and sibling budget rows before reaching T and E.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of these rows contains a total reciprocal, quotient stream equality, selected limit, trichotomy decision, or ambient completeness principle, the displayed obligations exhaust the local NameCert surface. □

Theorem 1444.15 (Regular Cauchy reciprocal window-budget coverage)**.** *Every budget-facing read of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ *must expose the rows*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B$$

in that order before it treats the packet as carrying a reciprocal-window budget.

Proof. Project the carrier of Definition 1471.1. The budget row B is displayed only after the source row Q, apartness row A, modulus row M, finite window row W, and dyadic row D.

Apply Definition 1471.2 and Definition 1471.6. These place the positive lower-bound display before any dyadic or budget read.

Use Definition 1471.7. It identifies Q, A, M, W, D as the nonzero-denominator window required before the $\mathsf{CauchyInverseBudget}^{\uparrow}$ row is reached.

Finally apply Definition 1471.9 and Definition 1471.14. Composed windows preserve the same displayed budget route, and the local NameCert surface contains no alternate budget coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalWindowBudgetCoverage`

Theorem 1444.16 (Regular Cauchy reciprocal readback-budget exactness)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, the readback row* T *is exactly the finite reciprocal-read route through the displayed budget row* B*. It is not a total reciprocal operation on arbitrary* $\mathsf{Real}^{\uparrow}$ *names, and it cannot be read without the preceding rows* Q, A, M, W, D, B*.*

Proof. Use Definition 1471.3. It factors T through the displayed chain from Q to B.

Apply Definition 1471.15. Any budget-facing read must first expose Q, A, M, W, D, B, so T has no route that bypasses the finite budget row.

Apply Definition 1471.8 and Definition 1471.9. Later windows and composed windows preserve the same readback only when the displayed source, apartness, modulus, window, dyadic, and budget rows are preserved.

Finally use Definition 1471.5. Since the terminal seal is not a total reciprocal on $\mathsf{Real}^{\uparrow}$, the preceding readback row is also only the finite route named by the carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalReadbackBudgetExactness`

Theorem 1444.17 (Regular Cauchy reciprocal seal-route conservativity)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier receives only the displayed route*

$$Q, A, M, W, D, B, T, E, H, C, P, N.$$

The seal row E conservatively packages that route and adds no total reciprocal, trichotomy, quotient stream equality, selected limit, or ambient completeness principle.

Proof. Apply Definition 1471.13. The terminal seal is offered only after the source, apartness, modulus, finite window, dyadic, budget, and readback rows have been displayed.

Use Definition 1471.16. It fixes T as the finite route through B, so the seal cannot detach E from the reciprocal-window budget.

Use Definition 1471.14. The remaining rows H, C, P, N transport, replay, source, and name the same packet, and no additional analytic coordinate exists.

The non-totality clause follows from Definition 1471.5. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only the displayed conservative route. □

Theorem 1444.18 (Regular Cauchy reciprocal scope dependency lock)**.** *For every accepted* RegularCauchyReciproc *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the reciprocal readback row T and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are scoped exactly by five L10 source rows: the* $\mathsf{RegSeqRat}^{\uparrow}$ *source Q, the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row B, and the* $\mathsf{Real}^{\uparrow}$ *seal route E. A consumer that drops any one of these rows remains outside the reciprocal readback and seal surface. The dependency lock exports no total reciprocal, quotient real equality, selected limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalScopeDependencyLock`

Proof. Begin with Definition 1471.2. It places the apartness row before every reciprocal-facing coordinate, so the route to the five named source rows is not optional.

Next apply Definition 1471.3. The readback row T factors through the displayed chain from the $\mathsf{RegSeqRat}^{\uparrow}$ source to the reciprocal budget row.

Use Definition 1471.15 together with Definition 1471.16. These two theorems identify the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{CauchyInverseBudget}^{\uparrow}$ row as the finite budget boundary for T.

Finally invoke Definition 1471.17. The $\mathsf{Real}^{\uparrow}$ seal E only packages the same displayed route, and its nonescape clause excludes total reciprocal, quotient equality, selected limits, and ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ consumer that reaches a reciprocal readback row or a $\mathsf{Real}^{\uparrow}$ seal without the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, apartness row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{CauchyInverseBudget}^{\uparrow}$ row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, cauchyinversebudget, real: no listed sibling supplies the whole twelve-row packet $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, because each sibling contributes only a regular source, finite window, dyadic ledger, reciprocal budget, or Real seal while this chapter binds all five analytic faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source, an apartness-window row, a finite modulus row, a $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{CauchyInverseBudget}^{\uparrow}$ budget row, a reciprocal readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1471.1, Definition 1471.2, Definition 1471.3, Definition 1471.4, Definition 1471.5, Definition 1471.14, Definition 1471.15, Definition 1471.16, Definition 1471.17, and Definition 1471.18 make the reciprocal chapter mature on the paper closure axis through the checked dependency lock, denominator-apartness window, $\mathsf{Real}^{\uparrow}$ seal handoff, and window-budget coverage.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalScopeDependencyLock`

Bridge status: none

Not claimed: This chapter does not claim a total reciprocal, trichotomy, quotient stream equality, selected limits, ambient real completeness, or a public bridge.

Upgrade path: Standard-bridge verification remains a separate formal route; the paper closure axis is mature for the displayed reciprocal dependency lock.

Chapter 1445

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilter}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki–Kelley filterbase terminology, the $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet names the finite bridge from directed Cauchy filter data to regular rational Cauchy names. It refines the filterbase surface of chapter 333, the regular-rational interface of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a current L10 Real and Completion route object: it stores the finite rows needed to read a regular Cauchy name from a directed Cauchy filterbase, but it stores no quotient of filters, selected limit, host equality of reals, countable-choice selector, or ambient completeness witness.

Definition 1445.1 (RegularCauchyFilter carrier)**.** *A* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, R, T, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase row, including its visible base packets, refinement ledger, and Cauchy-modulus row.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the filterbase by choosing only displayed regular rational windows.*
3. *T is the finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window schedule that lists the windows actually requested from the filterbase.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording endpoints, radii, and dyadic comparison rows for the same tail windows.*
5. *M is the directed tail-meet row: for any two requested tail windows it names a later base packet refining both and carrying the displayed dyadic bounds.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, R, T, D, M have been displayed.*
7. *H records componentwise* hsame *transport for B, R, T, D, M, E and for the structural rows below.*
8. *C records the displayed* Cont *replay from filterbase refinements to stream windows, from windows to regular-rational readback, from dyadic tolerances to the tail meet, and from the regular readback to the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten coordinates and their displayed transports. It has no coordinate for an external filter quotient, a host-selected limit, countable choice, ambient real completeness, or an unrecorded regular-name extraction.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterTasteGate single carrier alignment`

Theorem 1445.2 (RegularCauchyFilter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *carrier* $G = (B, R, T, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilter}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the displayed ten-row packet;*
2. *CauchyFilter source exposure through B and its finite filterbase rows;*
3. *StreamName tail-window exposure through T;*
4. *RegSeqRat readback through R after the tail windows are displayed;*
5. *DyadicRatCore tolerance exactness through D on the same tail windows;*
6. *directed tail-meet refinement through M;*
7. *Real seal handoff through E after B, R, T, D, M have been read;*
8. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No NameCert obligation may be discharged by a quotient of filters, a chosen host limit, countable choice, host real equality, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier namecert obligations`

Proof. Project the carrier of Definition 1445.1. The rows $B, R, T, D, M, E, H, C, P, N$ are the only coordinates visible to the local classifier.

The source side is inherited from the displayed $\mathsf{CauchyFilter}^{\uparrow}$ packet B. Its base packets, refinement rows, and Cauchy-modulus rows are read before any regular-rational or Real-facing consumer can inspect the packet.

The regular-name side is finite. The schedule T lists the requested tail windows, D attaches dyadic endpoints and radii to those windows, and R records the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the same displayed data.

The meet and seal rows close the obligation surface. The row M supplies the common directed refinement for requested tails, while E is downstream from B, R, T, D, M. Since the excluded principles are absent from the carrier, the eight listed rows exhaust the local NameCert surface. □

Theorem 1445.3 (RegularCauchyFilter directed tail meet)**.** *Let* $G = (B, R, T, D, M, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. For any two tail windows displayed in T, the row M names a later finite base packet of the Cauchy filterbase B that refines both windows through the recorded* Cont *routes and transports their dyadic tolerance rows through the recorded* hsame *comparisons. The resulting regular-rational readback remains the same R and reaches the same Real seal E.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier tail meet`

Proof. Start with the two displayed tail windows in T. Because T is part of the accepted carrier, both windows have visible source rows in the finite filterbase packet B.

Use the directed refinement surface of chapter 333. The carrier row M records the specific later base packet refining both requested windows, so the meet is a displayed finite row rather than an external directedness assertion.

Transport the dyadic bounds along the recorded structural rows. The ledger D supplies the endpoint and radius rows for each tail window, and H, C carry these rows to the common refinement named by M.

Finally read the regular and real-facing endpoints. Since R is the only $\mathsf{RegSeqRat}^{\uparrow}$ row and E is the only $\mathsf{Real}^{\uparrow}$ seal row of the packet, the common refinement preserves the displayed regular readback and seal route. □

Theorem 1445.4 (RegularCauchyFilter Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads B, then T and D, then the regular-rational row R, then the directed tail meet M, and only then the seal row E with its transports, continuations, provenance, and local naming. No second Real-facing route is available inside the carrier.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier real seal route`

Proof. Project the seal coordinate of Definition 1445.1. The row E is not stored as a detached Real object; it is downstream from the Cauchy filterbase, the finite tail schedule, the dyadic tolerance ledger, the regular-rational readback, and the directed meet row.

Read the displayed route in order. The Cauchy filterbase supplies B, the finite stream schedule supplies T, the dyadic core supplies D, and the regular-rational readback supplies R.

Now insert the meet row. By Definition 1445.3, any pair of requested tails has a displayed common refinement in M, and that refinement preserves the same R and E through H and C.

The package and naming rows P, N certify this same route. Since the carrier has no other Real seal coordinate, no quotient-filter route, and no ambient completeness coordinate, every Real-facing consumer factors through the listed finite chain. □

Theorem 1445.5 (RegularCauchyFilter basis factorization)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. The finite rows B, T, D, M, H, C, P, N determine a* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row whose basis elements are exactly the displayed tail windows in T, whose dyadic tolerance rows are exactly the rows recorded in D, and whose common-refinement and saturation handoff is the route from M into the finite* $\mathsf{CauchyFilter}^{\uparrow}$ *row B.*

Under this factorization, downstream readback reproduces the same $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the same* $\mathsf{Real}^{\uparrow}$ *seal E. The factorization uses only the displayed* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. It exports no quotient filter, selected limit, countable-choice selector, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier basis factorization`

Proof. Project the accepted carrier. The schedule T already lists the finite tail windows requested from the Cauchy filterbase, and the ledger D records the dyadic tolerances attached to exactly those windows. These are the basis elements and tolerance rows required by the $\mathsf{CauchyFilterBasis}^{\uparrow}$ carrier surface.

Next form the directed basis handoff. The row M supplies, for each pair of displayed tail windows, a later finite base packet refining both; this is precisely the common-refinement witness needed before the generated basis can be read as the saturation surface B of $\mathsf{CauchyFilter}^{\uparrow}$.

Transport and replay do not add coordinates. The rows H and C move the same T, D, M, B data componentwise through hsame and Cont, while P and N package and name the same finite route. Thus the basis row is obtained from the visible packet rather than from a completed filter or external selector.

Finally apply Definition 1445.4. Once the basis has handed off to B, the regular-rational readback remains the displayed row R, and the only Real-facing endpoint is the seal E. The excluded quotient, choice, host-equality, and completeness coordinates are absent from the carrier, so they cannot be exported by the factorization. □

Theorem 1445.6 (RegularCauchyFilter obligation scope package)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. It exports exactly the obligation-scope package formed by CauchyFilter source exposure through B, CauchyFilterBasis factorization through the displayed tail windows in T, DyadicRatCore tolerance exactness through D, directed tail-meet refinement through M, RegSeqRat readback through R, and RealUp seal handoff through E. The remaining rows H, C, P, N supply only componentwise transport, displayed replay, provenance, and local naming.*

Consequently the package exports no quotient filter, selected host limit, countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Proof. Apply Definition 1445.2. It identifies the local certificate surface as the displayed CauchyFilter source row, StreamName tail windows, RegSeqRat readback, DyadicRatCore tolerance ledger, directed meet row, Real seal row, and the structural transport, replay, provenance, and naming rows.

Apply Definition 1445.5. The tail windows in T, the dyadic rows in D, and the meet handoff in M determine the $\mathsf{CauchyFilterBasis}^{\uparrow}$ factorization over the same source row B.

Apply Definition 1445.3. The row M supplies the directed common-refinement witnesses for the displayed tails and preserves the same R and E through the recorded transports.

Apply Definition 1445.4. Every Real-facing read factors through the route ending in E, while H, C, P, N only transport, replay, package, and name the same finite data. The excluded quotient, choice, host-equality, and completeness coordinates are absent from the carrier, so they cannot be exported by the obligation package. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier obligation scope package`

Theorem 1445.7 (RegularCauchyFilter scoped handoff)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. The rows of G expose one scoped BEDC package over* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *whose visible components are exactly the $\mathsf{CauchyFilterBasis}^{\uparrow}$ factorization of B, the directed tail meet M, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows D, the terminal $\mathsf{Real}^{\uparrow}$ seal E, componentwise transport H, displayed replay C, provenance P, and local naming N.*

Every accepted consumer route through this package factors through the displayed rows $B, T, D, M, R, E, H, C, P,$ It exposes no quotient filter, hidden cofinal tail, selected limit, countable-choice selector, host equality, ambient completeness principle, or detached Real-facing route.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier scoped handoff`

Proof. First assemble the source and basis boundary. Apply Definition 1445.5 and Definition 1445.17 to read the $\mathsf{CauchyFilterBasis}^{\uparrow}$ witness from B, T, D, M, H, C, P, N.

Next close the tail side. Apply Definition 1445.3 and Definition 1445.8. These identify the displayed directed meet row M as the only common-refinement route for requested tails, with no hidden cofinal family.

Now expose the readback and seal. Apply Definition 1445.18 and Definition 1445.4; the regular rational row is R, and every Real-facing read reaches the same seal E only after the displayed windows and tolerance rows have been read.

Finally use Definition 1445.9, Definition 1445.12, and Definition 1445.6. These theorems repack transport, replay, provenance, and local naming as the single scoped package and rule out quotient filters, selected limits, choice selectors, host equality, hidden tails, and ambient completeness at the packet boundary. □

Theorem 1445.8 (RegularCauchyFilter tail-basis exhaustion)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. For any two finite windows accepted by the tail schedule T, the packet supplies a directed $\mathsf{CauchyFilter}^{\uparrow}$ meet row in M refining both windows inside the same source row B. The resulting tail read factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

by first reading the requested StreamName windows in T, then the dyadic tolerance rows in D, then the regular-rational row R, and finally the Real seal E.

Thus the tail basis is exhausted by the displayed rows $B, T, D, M, R, E, H, C, P, N$. It exports no quotient filter, selected host limit, countable-choice selector, hidden cofinal tail, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Proof. Begin with the two finite windows in T. Since T is an exposed coordinate of the accepted carrier, both windows are already tied to the source $\mathsf{CauchyFilter}^{\uparrow}$ row B and to the same componentwise transport rows H.

Apply Definition 1445.3. The meet coordinate M names the later base packet refining both requested windows, so the common tail is a displayed directed row rather than an external cofinality assertion.

Apply Definition 1445.5. The windows in T, the dyadic ledger D, and the meet handoff M form the $\mathsf{CauchyFilterBasis}^{\uparrow}$ row over B.

Finally apply Definition 1445.4 and Definition 1445.6. The downstream read reaches R and then E only through the displayed StreamName, DyadicRatCore, RegSeqRat, and Real seal route, while C, P, N replay, source, and name that same finite packet. The excluded quotient, choice, hidden-tail, host-equality, and completeness coordinates are absent from the carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier tail basis exhaustion`

Theorem 1445.9 (RegularCauchyFilter branch exhaustion)**.** *Every accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ consumer route from a finite Cauchy filterbase into a regular real name is exhausted by the displayed rows*

$$B, T, D, M, R, E, H, C, P, N.$$

The source row B supplies the finite $\mathsf{CauchyFilter}^{\uparrow}$ data, T supplies the requested $\mathsf{StreamName}^{\uparrow}$ windows, D supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, M supplies the directed meet, R supplies the

$\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* E *supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The structural rows* H, C, P, N *transport, replay, package, and name only that route. Thus no accepted consumer can split off a quotient filter, hidden cofinal tail, selected limit, countable-choice selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle from the filter-to-regular-name branch.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier branch exhaustion`

Proof. Project the displayed carrier. The source branch begins at B, and every requested tail window visible to the consumer appears in T with its dyadic tolerance data in D.

Apply Definition 1445.3. The meet row M supplies the later displayed base packet refining requested windows, so the consumer receives a finite directed refinement rather than an unrecorded cofinal tail.

Apply Definition 1445.5. The rows B, T, D, M, H, C, P, N form the visible $\mathsf{CauchyFilterBasis}^{\uparrow}$ route, and downstream readback is the same R with the same E.

Apply Definition 1445.4 and Definition 1445.8. The Real-facing endpoint is reached only after the StreamName, DyadicRatCore, RegSeqRat, and Real seal rows have been displayed. Since the carrier has no coordinate for quotient filters, hidden tails, selected limits, countable choice, host real equality, or ambient completeness, no consumer branch can export any of them. □

Theorem 1445.10 (RegularCauchyFilter tail-window exactness)**.** *Let* $G = (B, R, T, D, M, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. Each accepted tail window in the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*, together with its* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in* D*, determines the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the same* $\mathsf{Real}^{\uparrow}$ *seal* E *through the displayed* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *route over* B*.*

Equivalently, the finite tail-window read is exact at the packet boundary: the consumer may inspect the StreamName tail schedule, dyadic tolerance ledger, basis handoff, directed meet, regular readback, and Real seal rows, but it receives no quotient filter, selected limit, hidden cofinal tail, host real equality, countable-choice selector, or ambient completeness coordinate.

Proof. Project the accepted carrier. The only tail-window schedule is T, the only dyadic tolerance ledger for those windows is D, and the only regular readback and Real seal rows are R and E.

Apply Definition 1445.3 to the requested windows. The meet row M supplies the displayed later base packet inside the same source row B, so the tail read stays inside the finite Cauchy filterbase rather than passing to an unrecorded cofinal tail.

Apply Definition 1445.5. The rows B, T, D, M, H, C, P, N determine the visible $\mathsf{CauchyFilterBasis}^{\uparrow}$ route, and downstream readback along that route reproduces the same R and E.

Now use Definition 1445.4, Definition 1445.6, Definition 1445.8, and Definition 1445.9. These rows exhaust every accepted filter-to-regular-name consumer branch, so the excluded quotient, choice, hidden-tail, host-equality, and completeness coordinates are not available at the packet boundary. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier tail window exactness`

Theorem 1445.11 (RegularCauchyFilter directed refinement normal form)**.** *Let* $G = (B, R, T, D, M, E, H, C, P, N$ *be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. Every accepted carrier request for two tail windows in* T *factors through a single displayed directed tail-meet row in* M*, then through the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *factorization over* B*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Equivalently, the request has a normal form

$$(B, T, D, M) \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with componentwise transport, replay, package provenance, and local naming supplied by H, C, P, N. The normal form exports no quotient filter, selected limit, hidden cofinal tail, countable-choice selector, host real equality, or ambient completeness coordinate.

Proof. Begin with two accepted tail windows in the schedule T. By Definition 1445.3, the row M names one later displayed base packet refining both windows inside the same Cauchy filterbase row B.

Next pass through the basis factorization. Definition 1445.5 identifies the displayed windows in T, the dyadic rows in D, and the meet handoff in M as the $\mathsf{CauchyFilterBasis}^{\uparrow}$ route over B.

Now read the branch in its displayed order. The branch-exhaustion theorem confines every consumer to $B, T, D, M, R, E, H, C, P, N$, while tail-window exactness keeps the accepted windows tied to the same regular readback R and Real seal E.

Finally use the Real seal route and the obligation-scope package. They force the downstream read through the listed StreamName, RegSeqRat, DyadicRatCore, and Real rows, and coordinate exhaustion rules out quotient filters, selected limits, hidden cofinal tails, choice selectors, host real equality, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier directed refinement normal form`

Theorem 1445.12 (RegularCauchyFilter finite-window pullback)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. Every Real-facing finite tail-window read of G pulls back to the displayed $\mathsf{CauchyFilterBasis}^{\uparrow}$ tail window in T, the matching $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row in D, and the directed refinement row M before the same $\mathsf{RegSeqRat}^{\uparrow}$ readback R and terminal $\mathsf{Real}^{\uparrow}$ seal E can be consumed. Thus the pullback route is*

$$(T, D, M) \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto R \longmapsto E$$

inside the same source row B and the same structural rows H, C, P, N, with no quotient filter, hidden cofinal tail, selected limit, countable-choice selector, host real equality, or ambient completeness coordinate.

Proof. Begin with tail-window exactness, Definition 1445.10. It ties each accepted window in T and its dyadic tolerance row in D to the same readback row R and Real seal E.

Use tail-basis exhaustion and branch exhaustion next. Theorems Definition 1445.8 and Definition 1445.9 force every consumer route to expose the CauchyFilter source B, the displayed window row T, the dyadic row D, the meet row M, and the structural rows.

Now apply directed refinement normal form, Definition 1445.11. The requested finite windows are pulled back to a single displayed refinement in M before the basis route may be read as regular rational data.

Finally use the Real seal route, Definition 1445.4. Since the seal is reached only after B, T, D, M, R have been displayed and transported by H, C, P, N, the Real-facing read cannot bypass the finite-window pullback through quotient filters, hidden tails, selected limits, choice data, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier window pullback`

Theorem 1445.13 (RegularCauchyFilter carrier-basis rows)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. The rows $B, T, D, M, R, E, H, C, P, N$ decompose the packet into carrier admission, CauchyFilter source exposure, StreamName tail-window exposure,*

DyadicRatCore tolerance exactness, directed meet refinement, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming.

Under this decomposition, the visible basis consumed by later readers is exactly the $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row determined by* B, T, D, M, H, C, P, N*. No reader may replace this basis by a quotient filter, hidden cofinal family, selected limit, countable-choice selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier. The row B is the source CauchyFilter packet, while T lists the finite tail windows requested from that source.

The tolerance and meet rows are D and M. They attach dyadic bounds to the displayed windows and name the common directed refinements that make those windows a $\mathsf{CauchyFilterBasis}^{\uparrow}$ route over B.

The readback and seal rows are downstream. The row R records the regular rational readback, and E is reached only after the basis route has been displayed.

Finally, H, C, P, N transport, replay, source, and name the same rows. Because the listed coordinates exhaust the carrier, none of the excluded quotient, choice, host-equality, or completeness objects is available to a reader. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier basis rows`

Theorem 1445.14 (RegularCauchyFilter tail stability)**.** *Let* $G = (B, R, T, D, M, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. If two consumer requests read tail windows from* T*, then their common refinement is the displayed row in* M *over the same CauchyFilter source* B*, and the resulting tail read remains in the same* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *factorization with the same DyadicRatCore tolerance ledger* D*.*

Consequently the RegSeqRat readback R *and Real seal* E *are stable under tail-window refinement. The stability is transported by* H*, replayed by* C*, sourced by* P*, and named by* N*, without quotienting filters, choosing limits, invoking countable choice, or appealing to* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier tail stability`

Proof. Start with two displayed tail windows in T. Since they are carrier coordinates, both windows are already attached to the same source row B and the same dyadic tolerance ledger D.

The directed meet coordinate M supplies the common refinement. This refinement is a finite displayed row over B, not an external cofinality or choice assertion.

Read the basis factorization through the same rows. The refinement keeps the route inside $\mathsf{CauchyFilterBasis}^{\uparrow}$ and preserves the tolerance data in D for the requested windows.

The downstream rows are unchanged. The regular readback remains R, the real-facing seal remains E, and H, C, P, N preserve the same packet. Therefore the excluded quotient, selected-limit, choice, and completeness principles cannot enter the tail-stability claim. □

Theorem 1445.15 (RegularCauchyFilter ledger exhaustion)**.** *Every accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *consumer route is exhausted by the displayed ledger*

$$B, T, D, M, R, E, H, C, P, N.$$

The ledger first exposes the finite CauchyFilter source B*, then the StreamName tail windows* T*, then the DyadicRatCore tolerance rows* D*, then the directed meet row* M*, then the RegSeqRat readback* R*, and finally the Real seal* E *with transport, replay, provenance, and naming rows* H, C, P, N*.*

Thus the packet exports no quotient filters, selected limits, countable-choice selectors, hidden cofinal tails, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Any Real-facing read must pass through the displayed ledger before the seal is visible.*

Proof. Read the consumer route from the carrier coordinates. The source of every route is B, and every requested tail window appears in T with its tolerance data in D.

The row M supplies the only directed refinement data. Hence a consumer cannot bypass the displayed meet row by appealing to an unrecorded cofinal family.

The regular and real-facing rows are R and E. They are downstream from the source, tail-window, tolerance, and meet rows, and the structural rows H, C, P, N only preserve that route.

Since the ledger accounts for every coordinate of the carrier, no quotient filter, selected limit, countable-choice selector, hidden tail, host equality, or completeness coordinate remains to be exported. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier ledger`

Theorem 1445.16 (RegularCauchyFilter directed-tail closure)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. Every finite family of tail windows displayed in T is closed under the packet's directed meet row M: after finitely many displayed refinements, the family admits a single tail row over the same* $\mathsf{CauchyFilter}^{\uparrow}$ *source B that refines every listed window, preserves the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and reaches the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and* $\mathsf{Real}^{\uparrow}$ *seal E.*

The closure route is internal to $B, T, D, M, R, E, H, C, P, N$. It does not form quotient filters, select a host limit, use countable choice, import hidden cofinal tails, or appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed finite list of windows from T. The empty list and singleton list are immediate from carrier projection: the requested rows already lie inside the visible source row B and carry their tolerance data through D.

For the step, assume the previous subfamily has a displayed common tail row. Apply Definition 1445.3 to that row and the next listed window. The meet row M supplies a later displayed base packet refining both, so the enlarged family remains inside the same finite Cauchy filterbase surface.

Now use Definition 1445.5. The windows in T, the dyadic ledger D, and the iterated meet route through M remain a $\mathsf{CauchyFilterBasis}^{\uparrow}$ factorization over B.

Finally apply Definition 1445.15. The ledger exhaustion theorem keeps the downstream readback at the same R and the same terminal seal E, with transport, replay, provenance, and local naming supplied by H, C, P, N. Since the induction only composes displayed meet rows, none of the excluded quotient, choice, hidden-tail, limit, or completeness principles is introduced. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier directed tail closure`

Theorem 1445.17 (RegularCauchyFilter basis-witness admission)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet. The rows B, T, D, M, H, C, P, N admit a* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *witness whose basic windows are exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *tail windows in T, whose tolerance ledger is exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, and whose common-refinement witnesses are exactly the directed rows in M over the same* $\mathsf{CauchyFilter}^{\uparrow}$ *source B.*

The admitted basis witness carries the same hsame, Cont, Pkg, *and* NameCert *rows as G. It exports no quotient filter, hidden cofinal tail, selected limit, host real equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier basis witness admission`

Proof. Project the accepted carrier. The tail windows are visible in T, the dyadic tolerance data for those windows is visible in D, and the source Cauchy filterbase is the row B.

Use the directed meet rows. The row M supplies the common-refinement witnesses required by a $\mathsf{CauchyFilterBasis}^{\uparrow}$ admission, so the basis is built from displayed finite rows rather than from a completed filter.

The structural coordinates do not add a second source. The rows H, C, P, N transport, replay, source, and name the same B, T, D, M data. Hence the admitted basis witness has exactly the stated surface and none of the excluded external coordinates. □

Theorem 1445.18 (RegularCauchyFilter RegSeqRat window exposure)**.** *Let $G = (B, R, T, D, M, E, H, C, P, N)$ be an accepted $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet. Every accepted $\mathsf{RegSeqRat}^{\uparrow}$ consumer of G reads the same finite route*

$$B \longmapsto T \longmapsto D \longmapsto M \longmapsto R$$

through the $\mathsf{CauchyFilter}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ tail windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and $\mathsf{CauchyFilterBasis}^{\uparrow}$ directed-meet witness before the regular-rational row R is exposed.

The route is transported by H, replayed by C, sourced by P, and named by N. No consumer may read R through a quotient filter, hidden tail selector, host limit, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier regseqrat window exposure`

Proof. Begin with the displayed window schedule T. Each window is part of the accepted carrier and is attached to the same CauchyFilter source row B.

Read the tolerance and meet rows next. The row D supplies the dyadic bounds for the displayed windows, while M supplies the directed basis witness needed before regular rational readback is exposed.

Finally use Definition 1445.17. The same basis route reaches the unique regular-rational coordinate R, and H, C, P, N preserve that route. Since no external filter or selector coordinate is present, every $\mathsf{RegSeqRat}^{\uparrow}$ read factors through the displayed window exposure. □

Theorem 1445.19 (RegularCauchyFilter Real seal consumer exhaustion)**.** *Every accepted $\mathsf{Real}^{\uparrow}$ consumer of a $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet is exhausted by the route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the rows $B, T, D, M, R, E, H, C, P, N$. The Real seal E is visible only after the source row B, the basis witness admitted by T, D, M, and the regular-rational readback R have been read.

Thus the packet has no Real-facing consumer route through a filter quotient, host-selected limit, hidden cofinal family, countable-choice selector, host real equality, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier real seal consumer exhaustion`

Proof. Project the Real seal coordinate. The row E is downstream from the source, basis, tail-window, tolerance, meet, and regular-rational rows; it is not stored as an independent completed real.

Apply Definition 1445.18. Every regular-rational read reaches R through B, T, D, M and the same structural rows.

Now apply Definition 1445.4. The Real-facing endpoint is the same seal E, with H, C, P, N transporting, replaying, sourcing, and naming the route. The excluded quotient, choice, hidden-tail, equality, and completeness coordinates are absent, so the listed rows exhaust every accepted Real seal consumer. □

Theorem 1445.20 (RegularCauchyFilter scoped public package)**.** *Every accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet* $G = (B, R, T, D, M, E, H, C, P, N)$ *exports one scoped public package formed by the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row* B*, the displayed basis witness* T, D, M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N*. Every consumer route factors through this package and cannot split off quotient filters, selected limits, hidden cofinal tails, countable choice, host equality, or ambient completeness.*

Proof. First compose the source and basis rows. Apply Definition 1445.3, Definition 1445.5, and Definition 1445.17 to read the $\mathsf{CauchyFilter}^{\uparrow}$ source B through the displayed windows T, tolerance ledger D, and directed-meet witness M.

Next expose the regular-rational and Real-facing rows. Apply Definition 1445.18 to force every $\mathsf{RegSeqRat}^{\uparrow}$ readback through $B, T, D, M, R, H, C, P, N$, and apply Definition 1445.19 to force every $\mathsf{Real}^{\uparrow}$ consumer through the same route ending at E.

The scoped package is the single tuple of those rows. Transport is supplied by H, replay by C, provenance by P, and naming by N, so no consumer has a coordinate from which to extract a quotient filter, a host-selected limit, a hidden cofinal tail, a choice selector, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier scoped public package`

Theorem 1445.21 (RegularCauchyFilter mature route exhaustion)**.** *Every public or* $\mathsf{Real}^{\uparrow}$*-facing consumer route of an accepted* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *packet* $G = (B, R, T, D, M, E, H, C, P, N)$ *is exhausted by the displayed source row* B*, regular-rational readback* R*, tail-window row* T*, tolerance ledger* D*, directed meet row* M*, Real seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*, together with the scoped and public package theorems of this chapter. No consumer route factors through quotient filters, selected host limits, hidden cofinal tails, countable choice, host real equality, or ambient completeness.*

Proof. Start with Definition 1445.20. It packages the source, basis, regular-rational readback, Real seal, and structural rows as one public surface for the same carrier G.

For public routes, apply Definition 1445.6 and Definition 1445.7. These keep carrier admission, transport, replay, provenance, and naming on the displayed packet.

For $\mathsf{Real}^{\uparrow}$-facing routes, apply Definition 1445.18 and Definition 1445.19. The Real seal can be read only through the regular-rational window route ending at E.

The excluded routes are absent from the carrier and from the scoped public package. Hence the displayed rows exhaust every public or Real-facing consumer route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterUp.RegularCauchyFilterCarrier mature route exhaustion`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyFilter}^{\uparrow}$ packet is generated from a displayed CauchyFilter finite-filterbase row, a RegSeqRat readback row, StreamName tail windows, a DyadicRatCore tolerance ledger, a directed tail-meet row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyFilter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1445.1, Definition 1445.2, Definition 1445.3, Definition 1445.4, Definition 1445.5, Definition 1445.6, Definition 1445.7, Definition 1445.8, Definition 1445.9, Definition 1445.10, Definition 1445.17, Definition 1445.18, Definition 1445.19, Definition 1445.20, and Definition 1445.21 bind the admitted CauchyFilterBasis witness,

RegSeqRat window exposure, Real seal exhaustion, componentwise transport, displayed Cont replay, Pkg provenance, local NameCert rows, exported public package boundary, displayed carrier-to-public-route handoff, and public or Real-facing route exhaustion to the same accepted carrier $G = (B, R, T, D, M, E, H, C, P, N)$.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyFilter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected host limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Formal registration and Lean verification can encode the displayed RegularCauchyFilter carrier-to-public-route handoff for the same carrier.

Chapter 1446

A Concrete Naming Certificate for $\mathsf{LowerReal}^{\uparrow}$

Following Bishop and Bridges, lower real data are read as inhabited lower rational information with finite approximation windows, not as a completed Dedekind cut, a quotient real, or a host-completeness object. The $\mathsf{LowerReal}^{\uparrow}$ packet sits between the located cut discipline of chapter 404, the located real interface of **??**, the finite stream-window surface of chapter 126, the regular rational handoff of chapter 301, the dyadic ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It is the one-sided lower-real observation surface consumed by bound-shaped packets such as chapter 1427 and chapter 1452.

Definition 1446.1 (LowerReal carrier)**.** *A* $\mathsf{LowerReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (L_0, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. L_0 *is the lower rational or dyadic ledger: each row records a rational candidate, a dyadic comparison budget, and the evidence that the candidate is admitted by the lower side of the represented real.*
2. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the displayed lower observations are inspected.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attaching those windows to a regular rational name surface.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* L_0, W, R *have been displayed.*
5. H *is the componentwise* hsame *transport row for the lower ledger, the finite windows, the regular rational handoff, the Real seal, and the structural rows.*
6. C *is the displayed* Cont *replay route by which a consumer rereads the lower ledger through the same windows and handoff.*
7. P *is the* Pkg *provenance over* $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{LowerReal}^{\uparrow}}$ *row.*

The carrier exports no upper cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected supremum, no countable-choice row, and no host-completeness coordinate.

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealTasteGate single carrier alignment`

Theorem 1446.2 (LowerReal NameCert obligations)**.** *For an accepted* $\mathsf{LowerReal}^{\uparrow}$ *carrier* $L = (L_0, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LowerReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower rational or dyadic ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transports, continuation replay, package provenance, and local naming row;*
2. *lower-ledger transport, stating that replacement through* H *and replay through* C *preserve the same displayed lower candidates and their dyadic comparison budgets;*
3. *window scope, stating that every lower observation is read through the displayed finite windows* W *and no hidden stream tail;*
4. *regular-rational handoff, stating that each accepted lower-window read is attached to the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the Real seal is consumed;*
5. *real-seal nonescape, stating that* E *can be reached only after* L_0, W, R, H, C, P, N *have been displayed;*
6. *ledger exactness, stating that every accepted lower-real read uses exactly the rows recorded in* L_0 *and no second lower ledger, quotient equality, selected supremum, or ambient completeness coordinate.*

Proof. Carrier admission is the tuple displayed in Definition 1446.1. The packet has one lower ledger, one finite-window surface, one regular rational handoff, one Real seal, and the structural rows for transport, replay, provenance, and local naming.

Lower-ledger transport is componentwise over H and C. A classified replacement may carry the same rational candidates, dyadic budgets, windows, handoff, and seal through the displayed routes, but it cannot add an upper-cut coordinate, a second lower ledger, or a host equality witness.

Window scope and the regular-rational handoff are the roles of W and R. Every lower observation in L_0 is read through the finite $\mathsf{StreamName}^{\uparrow}$ window row, and the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational source before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Real-seal nonescape and ledger exactness follow from the row inventory. The only route to E passes through L_0, W, R, H, C, P, N, and the provenance row records the located-cut and located-real neighbors only as dependency anchors. No row stores a Dedekind cut completion, quotient real, trichotomy branch, selected supremum, countable choice, or host completeness principle. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealCarrier namecert obligations`

Theorem 1446.3 (LowerReal Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{LowerReal}^{\uparrow}$ *packet factors through the displayed lower ledger* L_0*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, with* H, C, P, N *carrying only componentwise transport, continuation replay, provenance, and local naming. The route exports no Dedekind cut completion, quotient real, trichotomy branch, selected supremum, countable-choice selector, or host-completeness coordinate.*

Proof. Project the accepted carrier. A consumer first sees the lower rational or dyadic ledger L_0, so the lower-real information available to it is exactly the displayed finite list of candidates and comparison budgets.

Next the consumer reads the finite observation route. The row W supplies the $\mathsf{StreamName}^{\uparrow}$ window, and R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the same displayed observations before any terminal seal is reachable.

Finally the read reaches E through the structural rows H, C, P, N. Since those rows only transport, replay, package, and name the already displayed ledger route, they cannot insert a Dedekind completion, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness object. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealCarrier realup handoff`

Theorem 1446.4 (LowerReal Dedekind-cut carrier admission)**.** *Every* $\mathsf{LowerReal}^{\uparrow}$ *read admitted as Dedekind-cut-facing data is admitted only through the displayed lower rational or dyadic ledger* L_0*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *of Definition 1446.1. The admission may cite* $\mathsf{LocatedCut}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *provenance through* P*, but it does not construct a Dedekind cut completion, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness witness.*

Proof. Project the accepted carrier. The only lower-real data row is L_0, so a Dedekind-cut-facing consumer first receives finite lower rational or dyadic candidates together with their comparison budgets.

Next force the observation through W and R. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite window, while the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational handoff before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally read provenance through P. The $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ entries are dependency anchors for the one-sided ledger, not extra coordinates carrying a completed cut, quotient equality, trichotomy, selected supremum, countable choice, or host completeness. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealCarrier dedekind cut admission`

Theorem 1446.5 (LowerReal located-cut ledger)**.** *The located-cut exactness obligation for* $\mathsf{LowerReal}^{\uparrow}$ *is the assertion that every located lower read replays the same* L_0, W, R, E, H, C, P, N *rows:* L_0 *supplies the lower rational or dyadic ledger,* W *supplies the* $\mathsf{StreamName}^{\uparrow}$ *window,* R *supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* E *supplies the* $\mathsf{Real}^{\uparrow}$ *seal, and* H, C, P, N *supply only transport, replay, provenance, and naming. No second ledger, quotient real equality, selected supremum, trichotomy branch, countable-choice selector, or host-completeness row is available.*

Proof. Start from the ledger exactness clause of Definition 1446.2. It already says that accepted lower-real reads use exactly the rows recorded in L_0 and no hidden second lower ledger.

The located-cut part is then read through the visible dependency anchors. $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ enter through P, while the observable lower data still pass through the finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ handoff before reaching the $\mathsf{Real}^{\uparrow}$ seal.

Thus a located lower read is exact only for the displayed one-sided ledger. Transport and replay preserve the same rows, and there is no coordinate from which a quotient real equality, trichotomy branch, selected supremum, countable choice, or host-completeness principle could be extracted. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealLocatedCutLedger`

Theorem 1446.6 (LowerReal UpperReal duality boundary)**.** *The* $\mathsf{UpperReal}^{\uparrow}$ *sibling may be cited by a* $\mathsf{LowerReal}^{\uparrow}$ *consumer only as a boundary row for one-sided comparison discipline. A* $\mathsf{LowerReal}^{\uparrow}$ *packet remains lower-sided: its admitted reads factor through* L_0, W, R, E, H, C, P, N

and the $\mathsf{LocatedCut}^{\uparrow}/\mathsf{LocatedReal}^{\uparrow}$ *provenance rows, then stop at the* $\mathsf{Real}^{\uparrow}$ *seal. The boundary does not supply upper-cut completion, quotient real equality, trichotomy, selected suprema, countable choice, or host completeness.*

Proof. Read the lower packet first. The carrier contains a lower ledger, finite stream windows, regular rational handoff, Real seal, transport, replay, provenance, and local naming; it contains no upper ledger coordinate.

The sibling boundary is therefore external provenance, not a second side of the carrier. It may identify where $\mathsf{UpperReal}^{\uparrow}$ would host upper-sided obligations, but it cannot change the row inventory of the current $\mathsf{LowerReal}^{\uparrow}$ packet.

Consequently all Real-facing reads still pass through the lower ledger, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal. The boundary records no upper-cut completion, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness witness. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealUpperRealDualityBoundary`

Theorem 1446.7 (LowerReal scoped kernel obligation)**.** *The displayed carrier, NameCert obligations, Real handoff, Dedekind-cut admission, located-cut ledger, and UpperReal boundary bind every* $\mathsf{LowerReal}^{\uparrow}$ *read to the rows for* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The scoped kernel exports no quotient real, trichotomy branch, selected supremum, countable-choice selector, or host-completeness coordinate.*

Proof. Project the carrier of Definition 1446.1. Its rows are the lower rational or dyadic ledger, finite stream windows, regular rational handoff, terminal Real seal, componentwise transport, continuation replay, provenance, and local naming.

Use Definition 1446.2, Definition 1446.3, and Definition 1446.4. These obligations keep every accepted read inside the displayed lower ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{LocatedCut}^{\uparrow}/\mathsf{LocatedReal}^{\uparrow}$ provenance, and $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 1446.5 and Definition 1446.6. They account for the located-cut ledger and the $\mathsf{UpperReal}^{\uparrow}$ sibling boundary without adding a second carrier side. The remaining structural rows are exactly BHist, hsame, Cont, Pkg, and NameCert, so the refused quotient, trichotomy, selected supremum, choice, and host-completeness objects have no row. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealCarrier scoped kernel obligation`

Definition 1446.8 (LowerReal public package)**.** *The* $\mathsf{LowerReal}^{\uparrow}$ *public package associated to an accepted carrier*

$$L = (L_0, W, R, E, H, C, P, N)$$

is the consumer-facing route that exposes the lower ledger L_0*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the support rows* H, C, P, N *as a single finite* Pkg*-named packet. The package may cite* $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *and* $\mathsf{UpperReal}^{\uparrow}$ *through provenance and boundary rows, but it exports no completed cut, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness witness.*

Lean4 (definition) `BEDC.Derived.LowerRealUp.LowerRealPublicPackage`

Theorem 1446.9 (LowerReal directed lower-cut budget)**.** *Let*

$$L = (L_0, W, R, E, H, C, P, N)$$

be an accepted $\mathsf{LowerReal}^{\uparrow}$ *carrier. Any two accepted lower-ledger rows in* L_0 *with displayed dyadic comparison budgets admit a common lower-read refinement only through the same route* L_0, W, R, E, H, C, P, N*: the refinement consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{LocatedCut}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *provenance rows, the* $\mathsf{UpperReal}^{\uparrow}$ *one-sided boundary, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any public consumer may read it. The route exports no selected supremum, no trichotomy branch, no quotient real equality, and no host-completeness principle.*

Proof. Start with the carrier of Definition 1446.1. The lower information is stored in L_0, and each row carries its dyadic comparison budget beside the finite observation route W, R, E.

Use the NameCert obligations of Definition 1446.2. Lower-ledger transport preserves the same rational candidates and dyadic budgets through H and C, while ledger exactness forbids a second lower ledger or hidden selector.

Now attach the located-cut ledger of Definition 1446.5. The $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ rows appear only through provenance over the same one-sided lower route, so a common refinement of two lower reads must stay inside L_0, W, R, E, H, C, P, N.

Finally apply the UpperReal duality boundary of Definition 1446.6. The $\mathsf{UpperReal}^{\uparrow}$ row records the sibling one-sided boundary but supplies no upper-cut completion. Therefore the common lower-read refinement is directed only by the displayed lower budgets and cannot use selected suprema, trichotomy, quotient real equality, or host completeness. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealDirectedLowerCutBudget`

Theorem 1446.10 (LowerReal public package scope)**.** *Every consumer of the public package of Definition 1446.8 must read the rows* L_0, W, R, E, H, C, P, N *in the same scoped route established by Definition 1446.7. The package boundary is public only for that displayed one-sided lower-real surface; it does not add a second lower ledger, an upper-cut carrier, quotient real equality, trichotomy, selected suprema, countable choice, or host completeness.*

Proof. Start with the carrier and NameCert obligations. The carrier gives exactly L_0, W, R, E, H, C, P, N, and Definition 1446.2 identifies those rows as the full local obligation surface.

Next apply the Real handoff and located-cut ledger theorems. They force each public read through W, R, E and keep $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ as provenance anchors rather than as completed cut coordinates.

Finally use the UpperReal boundary and scoped kernel obligation. The $\mathsf{UpperReal}^{\uparrow}$ row is only a sibling boundary, so the public package exposes the scoped lower route without adding any of the refused host or two-sided objects. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealPublicPackage scope`

Theorem 1446.11 (LowerReal public consumer export)**.** *Any downstream L10 consumer that imports* $\mathsf{LowerReal}^{\uparrow}$ *as public data receives only the package route of Definition 1446.8: lower ledger* L_0*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and support rows* H, C, P, N*. The export is therefore a public consumer route for the existing lower-real scope, not a completeness theorem, a quotient representative, a trichotomy branch, a supremum selector, or a countable-choice functional.*

Proof. Read the public package scope theorem first. It fixes the consumer-visible rows before any downstream L10 packet can project lower-real data.

Then use the Real handoff theorem. A consumer that wants Real-facing information must pass through L_0, W, R, E in that order, with H, C, P, N only transporting, replaying, sourcing, and naming the same route.

The located-cut ledger and UpperReal boundary account for the sibling dependencies. Since those dependencies are not carrier coordinates for completion, quotient equality, trichotomy, selected suprema, or countable choice, the public export is exactly the finite package route. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealPublicConsumer export`

Theorem 1446.12 (LowerReal public nonescape)**.** *The* $\mathsf{LowerReal}^{\uparrow}$ *public package has no escape route from the displayed rows* L_0, W, R, E, H, C, P, N *to a Dedekind-cut completion, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness coordinate. Every package read that reaches a public consumer is already covered by the carrier, NameCert obligations, Real handoff, located-cut ledger, UpperReal boundary, and scoped kernel obligation.*

Proof. Project the public package. Its only analytic rows are the lower ledger, stream window, regular rational handoff, and Real seal; its remaining rows are support for transport, replay, provenance, and local naming.

Apply the NameCert obligations, Real handoff, and located-cut ledger. These results keep every accepted package read inside the same lower ledger and prohibit hidden second ledgers, quotient equality, selected suprema, countable choice, and host completeness.

Close with the UpperReal boundary and scoped kernel obligation. They exhaust the sibling and kernel-facing dependencies, leaving no row from which a public consumer could extract the refused objects. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealPublicNonescape`

Theorem 1446.13 (LowerReal public RegSeqRat Real route)**.** *Every public* $\mathsf{LowerReal}^{\uparrow}$ *consumer that exposes a lower-real observation must route inhabited lower rational information through the displayed* $\mathsf{LocatedCut}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *provenance rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route reads only* L_0, W, R, E, H, C, P, N *and the named sibling rows; it exports no Dedekind completion, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host completeness coordinate.*

Proof. Project the public package of Definition 1446.8. Its consumer-visible data are the lower ledger L_0, finite stream window W, regular rational handoff R, Real seal E, and support rows H, C, P, N.

Use Definition 1446.10 and Definition 1446.12. These theorems force each public read to stay inside the displayed package route and block escape to completion, quotient equality, trichotomy, selected suprema, choice, or host completeness.

Now apply the located-cut ledger and Real handoff theorems. The $\mathsf{LocatedCut}^{\uparrow}$ and $\mathsf{LocatedReal}^{\uparrow}$ rows witness the provenance of inhabited lower rational information, while $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ provide the finite readback and terminal seal.

Therefore a public lower-real observation reaches consumers only through the RegSeqRat/Real corridor already named by the package. No additional completion object or selected supremum can be projected from the accepted $\mathsf{LowerReal}^{\uparrow}$ surface. □

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealPublicRegSeqRatRealRoute`

Theorem 1446.14 (LowerReal public LocatedSupremum handoff)**.** *Every* $\mathsf{LocatedSupremum}^{\uparrow}$*-facing consumer of the public* $\mathsf{LowerReal}^{\uparrow}$ *package may read only the lower ledger* L_0*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the support rows* H, C, P, N*, together with the existing* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{UpperReal}^{\uparrow}$ *provenance rows. The handoff exports the one-sided lower route used by the downstream supremum packet; it exports no selected supremum, trichotomy branch, quotient real equality, countable-choice schedule, or host-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealPublicLocatedSupremumHandoff`

Proof. Project the public package of Definition 1446.8. The consumer-visible lower-real rows are exactly L_0, W, R, E, H, C, P, N.

Use Definition 1446.13. It routes public lower observations through the located-cut and located-real provenance rows, finite stream windows, RegSeqRat readback, dyadic tolerance ledgers, and the terminal Real seal.

Now apply Definition 1446.12. The public package has no row for selected suprema, trichotomy, quotient real equality, countable choice, or host completeness.

Finally identify $\mathsf{LocatedSupremum}^{\uparrow}$ as a downstream consumer of the same one-sided lower route named in the chapter introduction. It may consume the displayed lower ledger and provenance boundary, but it cannot extract a supremum selector or host-completeness principle from the accepted $\mathsf{LowerReal}^{\uparrow}$ surface. □

Theorem 1446.15 (LowerReal located-cut seal boundary)**.** *Every public* $\mathsf{LowerReal}^{\uparrow}$ *read that presents a located-cut-facing seal must pass through* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before the seal is consumed. The read remains the displayed one-sided lower ledger* L_0, W, R, E, H, C, P, N *with located-cut and located-real provenance; it exports no completed Dedekind cut, quotient real equality, trichotomy branch, selected supremum, countable-choice selector, or host-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealLocatedCutSealBoundary`

Proof. Project the public package of Definition 1446.8. Its consumer-visible route is the lower ledger, finite stream window, regular rational handoff, Real seal, and support rows.

Use Definition 1446.13. It already routes public lower observations through located-cut and located-real provenance, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledgers, and the terminal Real seal.

Apply Definition 1446.12. The package has no row for completed cuts, quotient equality, trichotomy, selected suprema, countable choice, or host completeness. Thus a located-cut-facing seal is consumed only at the boundary named by the displayed route. □

Theorem 1446.16 (LowerReal UpperReal comparison nonescape)**.** *The* $\mathsf{UpperReal}^{\uparrow}$ *sibling is only a one-sided comparison boundary for the public* $\mathsf{LowerReal}^{\uparrow}$ *package. A consumer may use it to identify where upper-sided comparison data would be read, but it cannot supply Dedekind completion, quotient equality, trichotomy, selected suprema, countable choice, host completeness, or a two-sided real object. Every accepted comparison remains inside* L_0, W, R, E, H, C, P, N *and the displayed* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route.*

Lean4 ✓ `BEDC.Derived.LowerRealUp.LowerRealUpperRealComparisonNonescape`

Proof. Start with Definition 1446.6. It treats $\mathsf{UpperReal}^{\uparrow}$ as sibling provenance for one-sided comparison discipline, not as a second carrier side.

Next apply Definition 1446.15. The comparison must pass through the located-cut, located-real, stream-window, regular-readback, dyadic-budget, and Real-seal rows before any public consumer reads it.

The nonescape conclusion follows from Definition 1446.12. Since the package exposes only the lower route and support rows, the UpperReal boundary cannot supply completion, quotient equality, trichotomy, selected suprema, countable choice, host completeness, or a two-sided real object. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerReal}^{\uparrow}$ packet is generated from a displayed lower rational or dyadic ledger, finite StreamName windows, a RegSeqRat handoff row, a RealUp seal, componentwise transports, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerReal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1446.7 binds the finite one-sided lower-real surface to DyadicRatCore, StreamName, RegSeqRat, LocatedCut, LocatedReal, UpperReal, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind cut completion, quotient real equality, trichotomy, selected suprema, countable choice, host completeness, Lean verification, or a standard bridge.

Upgrade path: The chapter is mature on the paper closure axis; Lean bridge evidence remains governed only by the verification axis.

Chapter 1447

A Concrete Naming Certificate for $\mathsf{UpperReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, upper real data are read as finite upper rational or dyadic observations with explicit approximation windows, not as a Dedekind completion, quotient real, trichotomy branch, selected infimum, or host-completeness object. The $\mathsf{UpperReal}^{\uparrow}$ packet is the upper-ledger dual of the lower observation surface in chapter 1446; it supplies the one-sided upper data consumed by the located real interface of **??**, the located cut discipline of chapter 404, the finite stream-window surface of chapter 126, the regular rational handoff of chapter 301, the dyadic ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 1447.1 (UpperReal carrier)**.** *An* $\mathsf{UpperReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (U_0, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. U_0 *is the upper rational or dyadic ledger: each row records a rational candidate, a dyadic comparison budget, and the evidence that the candidate is admitted by the upper side of the represented real.*
2. L *is the sibling* $\mathsf{LowerReal}^{\uparrow}$ *comparison window, used only to expose a finite lower-upper apartness row when a located consumer asks for one-sided separation data.*
3. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the displayed upper observations are inspected.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attaching those windows to a regular rational name surface.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* U_0, W, R *have been displayed.*
6. H *is the componentwise* hsame *transport row for the upper ledger, the lower comparison window, the finite windows, the regular rational handoff, the Real seal, and the structural rows.*
7. C *is the displayed* Cont *replay route by which a consumer rereads the upper ledger through the same windows and handoff.*
8. P *is the* Pkg *provenance over* $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$ *row.*

The carrier exports no lower cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected infimum, no countable-choice row, and no host-completeness coordinate.

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealTasteGate single carrier alignment`

Theorem 1447.2 (UpperReal NameCert obligation surface)**.** *For an accepted* $\mathsf{UpperReal}^{\uparrow}$ *carrier* $U = (U_0, L, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$ *surface consists exactly of the following seven finite obligations:*

1. *carrier admission for the upper rational or dyadic ledger, the sibling lower comparison window, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transports, continuation replay, package provenance, and local naming row;*
2. *upper-ledger transport, stating that replacement through* H *and replay through* C *preserve the same displayed upper candidates and their dyadic comparison budgets;*
3. *lower-upper apartness scope, stating that every comparison with* $\mathsf{LowerReal}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *reads only the displayed lower comparison window* L *together with the upper candidates in* U_0*;*
4. *window scope, stating that every upper observation is read through the displayed finite windows* W *and no hidden stream tail;*
5. *regular-rational handoff, stating that each accepted upper-window read is attached to the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the Real seal is consumed;*
6. *real-seal nonescape, stating that* E *can be reached only after* U_0, L, W, R, H, C, P, N *have been displayed;*
7. *ledger exactness, stating that every accepted upper-real read uses exactly the rows recorded in* U_0 *and no second upper ledger, lower cut, located cut completion, quotient equality, selected infimum, countable-choice row, or ambient completeness coordinate.*

Proof. Carrier admission is the tuple displayed in Definition 1447.1. The packet has one upper ledger, one lower comparison window, one finite-window surface, one regular rational handoff, one Real seal, and the structural rows for transport, replay, provenance, and local naming.

Upper-ledger transport is componentwise over H and C. A classified replacement may carry the same rational candidates, dyadic budgets, lower comparison window, observation windows, handoff, and seal through the displayed routes, but it cannot add a second upper ledger or any hidden completion coordinate.

The lower-upper apartness scope is supplied by L together with U_0. A located or lower-real consumer can compare only those displayed rows, so the comparison window is finite and does not become a Dedekind cut, quotient equality test, or selected infimum.

Window scope and the regular-rational handoff are the roles of W and R. Every upper observation in U_0 is read through the finite $\mathsf{StreamName}^{\uparrow}$ window row, and the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational source before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Real-seal nonescape and ledger exactness follow from the row inventory. The only route to E passes through U_0, L, W, R, H, C, P, N, and the provenance row records the lower-real, located-real, and located-cut neighbors only as dependency anchors. The carrier exports no lower cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected infimum, no countable-choice row, and no host-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier obligation surface`

Theorem 1447.3 (UpperReal lower-upper apartness window)**.** *Every* $\mathsf{LowerReal}^{\uparrow}$*-,* $\mathsf{LocatedReal}^{\uparrow}$*-, or* $\mathsf{LocatedCut}^{\uparrow}$*-facing comparison read of an accepted* $\mathsf{UpperReal}^{\uparrow}$ *packet factors through the displayed lower comparison window* L*, the upper ledger* U_0*, and the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*. The comparison may expose a visible dyadic gap between the lower-side witness and an upper candidate, transported by* H *and replayed by* C*, but it cannot introduce lower-cut completion data, quotient real equality, a trichotomy branch, a selected infimum, countable choice, or host completeness.*

Proof. Project the accepted carrier $U = (U_0, L, W, R, E, H, C, P, N)$. A comparison consumer sees the lower-side window L and the upper rational or dyadic ledger U_0, so the available apartness information is exactly the displayed finite list of lower-side witnesses, upper candidates, and comparison budgets.

Next the consumer reads the finite observation route. The row W supplies the $\mathsf{StreamName}^{\uparrow}$ window, and R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the same displayed observations before any terminal seal is reachable.

Transport is internal to the packet. Moving along H and replaying C changes the presented row names but not the finite dependency route from L and U_0 through W and R.

The carrier exports no lower cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected infimum, no countable-choice row, and no host-completeness coordinate. Therefore the comparison remains a finite lower-upper apartness window over the displayed rows. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier lower upper apartness window`

Theorem 1447.4 (UpperReal public certificate)**.** *Every public* $\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$ *export of an accepted* $\mathsf{UpperReal}^{\uparrow}$ *packet is exhausted by the displayed carrier, upper-ledger transport, lower-upper apartness window, finite-window scope, regular-rational handoff, real-seal nonescape, and ledger exactness rows over* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Every* $\mathsf{Real}^{\uparrow}$*-facing read reaches the terminal seal* E *only through* U_0, L, W, R, H, C, P, N *and exports no lower cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected infimum, no countable-choice row, and no host-completeness coordinate.*

Proof. Read the carrier route in order. The upper ledger U_0 supplies the finite rational or dyadic observations, L supplies the lower-side comparison window, W supplies the window discipline, and R supplies the regular rational handoff for those same observations.

The terminal seal E is downstream of that finite route. Its continuation in C can replay the displayed rows, while H transports only the componentwise names already present in the packet.

The provenance and local naming rows P, N record the sibling certificates and primitive BEDC rows named in Definition 1447.1. They do not store any second completion route or any independent real-equality test.

Thus the carrier exports no lower cut, no located cut completion, no quotient real equality, no trichotomy branch, no selected infimum, no countable-choice row, and no host-completeness coordinate. The $\mathsf{Real}^{\uparrow}$ seal can be consumed only as the boundary of the displayed upper-ledger path. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier public certificate`

Theorem 1447.5 (UpperReal scoped kernel dependency route)**.** *Every accepted* $\mathsf{UpperReal}^{\uparrow}$ *packet reads through the displayed* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of Definition 1447.1 before either its upper ledger or its*

real-seal boundary is exposed. In particular the carrier admission, upper-ledger transport, lower-upper apartness scope, finite-window scope, regular-rational handoff, real-seal nonescape, and ledger exactness obligations are all scoped to the single route

$$U_0, L, W, R, E, H, C, P, N.$$

No accepted packet can expose a second upper ledger, a lower cut, a located cut completion, quotient real equality, trichotomy, selected infimum, countable choice, host completeness, or an unrecorded sibling dependency outside that route.

Proof. Start with the carrier admission of Definition 1447.1. It names the upper ledger U_0, the lower comparison row L, the finite window row W, the regular rational handoff R, the terminal Real seal E, and the structural rows H, C, P, N.

Apply Definition 1447.2. Its seven obligations list the displayed carrier admission, upper-ledger transport, lower-upper apartness scope, finite-window scope, regular-rational handoff, real-seal nonescape, and ledger exactness clauses, so each required dependency is already bound to the visible rows.

Use Definition 1447.3 for the sibling comparison route. It forces every $\mathsf{LowerReal}^{\uparrow}$-, $\mathsf{LocatedReal}^{\uparrow}$-, or $\mathsf{LocatedCut}^{\uparrow}$-facing read through L, U_0, W, R before any comparison boundary can be consumed.

Finally apply Definition 1447.4. It exhausts the public certificate through BHist, hsame, Cont, Pkg, and NameCert provenance and makes E reachable only through the displayed upper-ledger path. Hence both the upper ledger and the real-seal boundary are exposed only after the named dependency rows have been read. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier scoped kernel dependency route`

Theorem 1447.6 (UpperReal located-cut bridge surface)**.** *Every accepted* $\mathsf{UpperReal}^{\uparrow}$ *packet bridges to the located-cut and located-real completion route only through its upper rational or dyadic observation rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The bridge preserves the displayed route*

$$U_0, L, W, R, E, H, C, P, N$$

from Definition 1447.1; it exposes no Dedekind-completion object, quotient-real equality, trichotomy branch, selected infimum, ambient-completeness principle, or off-packet located-cut coordinate.

Proof. Start from the scoped dependency route of Definition 1447.5. It forces every accepted upper ledger and real-seal boundary through $U_0, L, W, R, E, H, C, P, N$ before the packet is exposed.

The located-cut and located-real consumers can therefore read the upper side only through the displayed observation rows. The row W supplies the finite $\mathsf{StreamName}^{\uparrow}$ windows, R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the dyadic budgets recorded in U_0 and the provenance row supply the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before E seals the $\mathsf{Real}^{\uparrow}$ route.

Apply Definition 1447.3 for the comparison boundary. It permits a visible lower-upper apartness window, but it does not export a lower cut or a completed located-cut object.

Finally, Definition 1447.4 exhausts the public certificate by the same finite rows. Any Dedekind completion, quotient real, trichotomy branch, selected infimum, or ambient-completeness reading would require a coordinate absent from that packet, so the bridge is exactly the displayed located-cut route and nothing stronger. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier located cut bridge`

Theorem 1447.7 (UpperReal completeness-facing consumer route)**.** *Every completeness-facing consumer of an accepted* $\mathsf{UpperReal}^{\uparrow}$ *packet reads the upper ledger only through the displayed* $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 1447.1. The consumer may use the finite route*

$$U_0, L, W, R, E, H, C, P, N$$

and the located-cut bridge of Definition 1447.6, but it receives no lower-cut completion, selected infimum, quotient real equality, trichotomy, countable choice, or ambient host-completeness object.

Proof. Project the accepted carrier $U = (U_0, L, W, R, E, H, C, P, N)$. The upper ledger U_0 is visible only beside the lower comparison window L, finite window row W, regular-rational handoff R, and terminal Real seal E.

Apply Definition 1447.3 and Definition 1447.6. They route every lower-real, located-real, and located-cut read through the displayed one-sided comparison and window rows before any Real-facing seal is consumed.

Finally use Definition 1447.4. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so a completeness-facing consumer receives exactly the named rows and none of the excluded completion, infimum, equality, trichotomy, choice, or host-completeness endpoints. □

Lean4 ✓ `BEDC.Derived.UpperRealUp.UpperRealCarrier completeness facing consumer route`

Closure status for $\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$**.**

Constructive story: An $\mathsf{UpperReal}^{\uparrow}$ packet is generated from a displayed upper rational or dyadic ledger, a lower comparison window, finite StreamName windows, a RegSeqRat handoff row, a RealUp seal, componentwise transports, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1447.1, Definition 1447.2, Definition 1447.3, Definition 1447.4, Definition 1447.5, Definition 1447.6, and Definition 1447.7 scope the finite one-sided upper-real surface through the displayed carrier admission, upper-ledger transport, lower-upper apartness window, finite-window scope, RegSeqRat handoff, Real-seal nonescape, ledger exactness rows, located-cut bridge boundary, and completeness-facing consumer route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UpperReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim lower cuts, located cut completion, quotient real equality, trichotomy, selected infima, countable choice, host completeness, Lean verification, or a standard bridge.

Upgrade path: No further paper-theory upgrade is required for the displayed upper-real route; Lean verification and standard bridge checking remain separate formal-axis work.

Chapter 1448

A Concrete Naming Certificate for $\mathsf{ClosedUnitInterval}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ClosedUnitInterval}^{\uparrow}$ packet records the displayed $[0, 1]$ real-name interval as finite dyadic windows, regular-rational readback, rational-density witnesses, and a visible $\mathsf{Real}^{\uparrow}$ seal. It is a closed endpoint packet over chapter 291, chapter 126, chapter 301, chapter 38, chapter 1410, and chapter 1409; it is not an open-cover compactness theorem, a selected subsequence, quotient real equality, or host real interval.

Definition 1448.1 (Closed unit interval carrier)**.** *A* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Z, O, W, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *Z is the displayed zero endpoint row, read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational and dyadic ledger and then through the visible* $\mathsf{Real}^{\uparrow}$ *seal;*
2. *O is the displayed one endpoint row, read through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational and dyadic ledger and the same* $\mathsf{Real}^{\uparrow}$ *seal;*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger used for the endpoint names and for any inspected interior regular name;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating those finite windows into regular rational approximants;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic scales at which endpoint, window, and boundary comparisons are inspected;*
6. *R is the rational-density or* $\mathsf{BishopIntervalCompact}^{\uparrow}$ *route row that records the finite witness surface by which an inspected regular-rational name lies between the displayed endpoints;*
7. *H records componentwise* hsame *transport for endpoints, windows, readbacks, dyadic tolerances, density witnesses, interval routes, continuation rows, package provenance, and the local naming row;*
8. *C records displayed* Cont *replay from endpoint rows to dyadic windows, from windows to regular-rational readback, from readback to rational-density or finite interval-net witnesses, and from those rows to the visible* $\mathsf{Real}^{\uparrow}$ *seal;*

9. *P is the* Pkg *provenance over* $\mathsf{ClosedUnitInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopIntervalCompact}^{\uparrow}$, $\mathsf{RationalDensityReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedUnitInterval}^{\uparrow}}$ *row.*

The carrier classifier compares two accepted packets only by replaying the displayed rows $Z, O, W, Q, D, R, H, C, P, N$. It has no coordinate for an arbitrary open cover, Heine–Borel compactness, sequential compactness, quotient real equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1448.2 (Closed unit interval NameCert obligations)**.** *For an accepted* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *packet $U = (Z, O, W, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedUnitInterval}^{\uparrow}}$ *obligation surface consists exactly of endpoint admission for Z and O, finite-window admission through W, regular-rational readback through Q, dyadic tolerance admission through D, rational-density or finite interval-net routing through R, componentwise* hsame *transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1448.1. The zero endpoint, one endpoint, window, readback, dyadic tolerance, interval-route, transport, continuation, provenance, and naming rows are the complete displayed coordinates of the finite packet.

Read the endpoint and real-name side first. The rows Z and O expose the two displayed endpoints, while W, Q, and D expose only finite windows, regular-rational approximants, and dyadic tolerances for those endpoints and inspected names.

Read the interval route next. The row R records the rational-density or finite interval-net witness surface that places the inspected read between the endpoints at the displayed tolerance. It does not select a subsequence, a limit, or a subcover.

Finally H, C, P, and N transport, replay, source, and name exactly these rows. Since the carrier has no coordinate for open covers, Heine–Borel compactness, quotient equality, host equality, choice, or ambient completeness, no such obligation can be read from the local NameCert surface. □

Theorem 1448.3 (Closed unit interval dyadic window boundary)**.** *Every accepted* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *consumer read of a point between the displayed endpoints factors through the finite boundary surface*

$$Z, O, W, Q, D, R.$$

The read first inspects the zero and one endpoint rows, then the finite $\mathsf{StreamName}^{\uparrow}$ *windows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then the rational-density or finite interval-net route named by R.*

Proof. Begin with the endpoint rows. The packet exposes the displayed zero row Z and one row O before any interior or boundary membership read is available.

Next inspect the finite windows and rational readback. The rows W and Q provide only the scheduled $\mathsf{StreamName}^{\uparrow}$ observations and regular-rational approximants that the dyadic tolerance ledger D can compare.

Finally read the interval route R. It is attached to those same windows and tolerances, so a consumer cannot bypass Z, O, W, Q, and D to obtain an ambient interval object or host real comparison.

The boundary is therefore a finite dyadic-window boundary. Later tails of a real name, quotient equality, selected limits, countable choice, and real completeness are absent from the packet and cannot affect the read. □

Theorem 1448.4 (Closed unit interval Real seal scope)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *packet must first consume the endpoint, finite-window, regular-rational, dyadic-tolerance, and rational-density or finite interval-net rows named by* Z, O, W, Q, D*, and* R*. The visible* $\mathsf{Real}^{\uparrow}$ *seal confirms the displayed endpoint packet and no stronger real interval principle.*

Proof. Expose the real-facing rows by projection from Definition 1448.1. The packet reaches the visible $\mathsf{Real}^{\uparrow}$ seal only after the endpoint rows and their finite window, readback, tolerance, and route rows have been displayed.

Use Definition 1448.2 to exhaust the local certificate surface. Its real-facing obligations are exactly the listed endpoint, window, readback, dyadic, interval-route, transport, continuation, provenance, and naming rows.

Use Definition 1448.3 for the consumer boundary. Any read that treats the packet as an interval first passes through the finite dyadic-window surface, so the seal cannot be reinterpreted as quotient real equality, host equality, open-cover compactness, selected limits, countable choice, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedUnitInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedUnitInterval}^{\uparrow}$ packet is generated from displayed zero and one endpoint rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a rational-density or BishopIntervalCompact route row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedUnitInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1448.1, Definition 1448.2, Definition 1448.3, and Definition 1448.4 bind the finite closed endpoint packet for the displayed unit interval.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedUnitInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel compactness for arbitrary subsets, sequential compactness, quotient real equality, selected limits, host real equality, countable choice, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic window boundary transport, rational-density or finite interval-net routing, and Real seal nonescape for BEDC.Derived.ClosedUnitIntervalUp.

Chapter 1449

A Concrete Naming Certificate for $\mathsf{BoundedRealFamily}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRealFamily}^{\uparrow}$ packet records a bounded family of real names by finite index-window rows, visible StreamName windows, regular rational readback, RealUp family seals, and a closed-bounded interval-net bound row. It is the family-facing analogue of the bounded sequence surface in chapter 1415, shares the explicit modulus discipline of chapter 1423, and reads cover-facing interval data through chapter 1428. Its bound row is the finite $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ surface of chapter 1250, and each terminal family member is sealed only through the $\mathsf{Real}^{\uparrow}$ route of chapter 38. The packet prepares later least-upper-bound, compactness, and cover-selection certificates without importing trichotomy, quotient equality, countable choice, or a supremum principle.

Definition 1449.1 (Bounded real family carrier)**.** *A* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, W, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-window ledger. Each row names an inspected family index together with the finite observation window requested at that index.*
2. *W is the family of* $\mathsf{StreamName}^{\uparrow}$ *windows displayed for the inspected index-window rows of I.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those StreamName windows and their visible dyadic tolerance observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *family-seal row. A family member is visible to a Real-facing consumer only after its index row, StreamName window, and RegSeqRat readback have been displayed.*
5. *B is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row carrying the located lower and upper interval data used to witness a common bound for the displayed family rows.*
6. *H stores componentwise* hsame *transport for the index-window ledger, StreamName windows, regular readback, Real family seals, bound row, continuation replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from index-window rows to StreamName windows, from StreamName windows to RegSeqRat readback, from RegSeqRat readback to Real family seals, and from Real family seals to the common bound ledger.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundedRealFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$ *row.*

The bound row is displayed as finite located interval data

$$B : \mathsf{ClosedBoundedIntervalNet}^{\uparrow},$$
$$\forall i \in I_{\text{shown}} \quad W_i \longmapsto Q_i \longmapsto R_i \in B.$$

Here I_{shown} denotes only the inspected finite index-window rows carried by the packet. The classifier compares accepted packets only through $I, W, Q, R, B, H, C, P, N$ and has no coordinate for trichotomy, quotient equality of real names, countable choice over indices, an ambient supremum, open-cover compactness, or host completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyTasteGate single carrier alignment`

Theorem 1449.2 (Bounded real family located upper row)**.** *For every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier $F = (I, W, Q, R, B, H, C, P, N)$, the bound row B is read through finite located lower and upper interval data. Each displayed family member reaches the common bound by the route*

$$I \longmapsto W \longmapsto Q \longmapsto R \longmapsto B$$

together with H, C, P, N. Thus the located upper row is a finite certificate row and imports no trichotomy, quotient equality of real names, countable choice, supremum principle, or host completeness.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyLocatedUpperRow`

Proof. Project the carrier of Definition 1449.1. The only family data exposed to the classifier are the finite index-window ledger I, the displayed $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the terminal $\mathsf{Real}^{\uparrow}$ family seals R.

Read the bound coordinate next. The row B is a $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ packet, so its lower and upper interval data are displayed finite rows from chapter 1250. A family member can meet B only after its own window, readback, and seal route has already been replayed.

The structural rows preserve this same route. Componentwise transport H, continuation replay C, package provenance P, and the local name N have no field for trichotomy, quotient equality, countable choice, a supremum, or host completeness. Hence the located upper row is read as finite interval data and nothing stronger. □

Theorem 1449.3 (Bounded real family RegSeqRat scope)**.** *Every Real-facing read of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow}$$

as instantiated by the rows W, Q, R, B and transported by H, C, P, N. No consumer can read a family member, a bound assertion, or a cover-facing handoff outside the StreamName windows, RegSeqRat readback, RealUp family seals, and bound ledger carried by the packet.

Lean4 ✓ `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyRegSeqRatScope`

Proof. Begin with a Real-facing request for an inspected family row. The carrier first exposes the index-window ledger I and the corresponding $\mathsf{StreamName}^{\uparrow}$ window in W.

The request then passes through the regular rational readback. The row Q is the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate, and the terminal $\mathsf{Real}^{\uparrow}$ seal R is reached only after Q has been displayed.

Finally the bound assertion is read through B. Since B is the only bound ledger and since H, C, P, N only transport, replay, source, and name the displayed rows, the request cannot leave the StreamName, RegSeqRat, RealUp, and bound-ledger scope of the packet. □

Theorem 1449.4 (Bounded real family bound-ledger nonescape)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet reads the common bound only through the displayed rows* $I, W, Q, R, B, H, C, P, N$*. In particular, a consumer cannot export trichotomy, quotient real equality, countable choice over indices, a supremum principle, open-cover compactness, or host completeness from the bound ledger.*

Proof. Project the carrier of Definition 1449.1. The common bound is the single row B, and it is available only after the finite index-window ledger I, StreamName windows W, RegSeqRat readback Q, and Real family seals R have been displayed.

Apply Definition 1449.2. It identifies the bound row as finite located lower and upper interval data rather than as an ambient order-completeness witness or a selected supremum.

Apply Definition 1449.3. It forces every Real-facing read of a family member or bound assertion through the StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route carried by W, Q, R, B.

It remains to inspect the structural rows. The transport row H preserves only componentwise equality of the displayed rows, the replay row C replays only the displayed route, the provenance row P names only the sibling sources, and the local row N names the packet. None of H, C, P, N contains a branch for trichotomy, quotient real equality, countable choice over indices, supremum extraction, open-cover compactness, or host completeness. Thus the common-bound read cannot escape the finite bound ledger. □

Theorem 1449.5 (Bounded real family obligation surface)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier exposes exactly the finite index-window ledger* I*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{Real}^{\uparrow}$ *family seals* R*,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$ *row* N *before any compactness, least-upper-bound, or cover-selection consumer may read it.*

Proof. Project the carrier of Definition 1449.1. Its displayed coordinates are exactly $I, W, Q, R, B, H, C, P, N$, so an obligation-level consumer has no other entry point.

Apply Definition 1449.2 to the bound coordinate. The common bound is finite located interval-net data, not a supremum, cover selector, or host completeness witness.

Apply Definition 1449.3 and Definition 1449.4. They force every Real-facing read through the StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route and bar trichotomy, quotient equality, countable choice, compactness extraction, least-upper-bound extraction, and cover-selection reads. Hence the listed rows are exactly the obligation surface available before downstream consumers add their own data. □

Theorem 1449.6 (BoundedRealFamily scoped dependency route)**.** *Every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *packet exposes its finite index-window ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *family-seal row,* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *bound row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before any compactness, least-upper-bound, or cover-selection consumer may read it. The route exports no trichotomy, quotient equality of real names, countable choice, supremum principle, open-cover compactness, or host completeness.*

Proof. Project the carrier in Definition 1449.1. Its finite dependency route is exactly $I, W, Q, R, B, H, C, P, N$.

The analytic rows are already ordered. Apply Definition 1449.2 for the located upper row, Definition 1449.3 for the StreamName and RegSeqRat scope, and Definition 1449.4 for the bound-ledger refusal.

Now apply Definition 1449.5. It exhausts the rows visible before a downstream compactness, least-upper-bound, or cover-selection consumer adds its own data.

Since the packet has no coordinate beyond those rows, it exports no trichotomy, quotient equality, countable choice, supremum, open-cover compactness, or host-completeness principle. □

Theorem 1449.7 (Bounded real family compactness consumer boundary)**.** *Every compactness consumer of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier* $F = (I, W, Q, R, B, H, C, P, N)$ *must add its own finite-net, total-boundedness, located-compactness, or compactness row before reading a compactness conclusion. The rows* $I, W, Q, R, B, H, C, P, N$ *expose the bounded-family obligation surface and common bound ledger, but they do not by themselves supply an open-cover compactness theorem, selected finite subcover, compact metric witness, or compactness classifier.*

Proof. Start from the carrier in Definition 1449.1. Its coordinates list finite index windows, StreamName windows, RegSeqRat readback, Real seals, one ClosedBoundedIntervalNet bound row, transport, replay, provenance, and the local NameCert row.

Apply Definition 1449.2 and Definition 1449.3. These route every Real-facing read through the displayed windows, readback, seals, and bound row, so a compactness consumer can only see the bounded-family route already named by $I, W, Q, R, B, H, C, P, N$.

Finally apply Definition 1449.4 and Definition 1449.5. The bound ledger refuses compactness extraction, and the obligation surface is exhausted before any downstream compactness consumer reads the packet. Therefore compactness requires an explicit downstream finite-net or compactness row. □

Theorem 1449.8 (Bounded real family least-upper-bound refusal)**.** *For every accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier* $F = (I, W, Q, R, B, H, C, P, N)$*, the common bound row* B *is a* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row and not a supremum, least-upper-bound, order-completeness, or trichotomy witness. A least-upper-bound consumer must therefore provide its own supremum or completeness row outside the bounded-family packet.*

Proof. Read the carrier through Definition 1449.1. The bound coordinate is named as $B : \mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, attached downstream of the finite index-window, StreamName, RegSeqRat, and Real-family rows.

Apply Definition 1449.2. It identifies B as finite located lower and upper interval data, so the row witnesses a displayed common bound rather than a selected least upper bound.

Apply Definition 1449.4 and Definition 1449.5. These theorems remove trichotomy, quotient equality of real names, countable choice, host completeness, and supremum extraction from the rows available to a consumer. Thus a least-upper-bound read must come from a separate downstream row. □

Theorem 1449.9 (Bounded real family cover-selection boundary)**.** *Every cover-selection consumer of an accepted* $\mathsf{BoundedRealFamily}^{\uparrow}$ *carrier must pass through an explicit finite-cover, finite-net, or selected-cover sibling row before exporting a cover conclusion. The bounded-family packet supplies only* $I, W, Q, R, B, H, C, P, N$ *and cannot select a finite subcover from those rows alone.*

Proof. Project the carrier in Definition 1449.1. The only cover-facing data present before downstream consumption are the family windows, rational readback, Real seals, the ClosedBoundedIntervalNet bound row, and the structural rows H, C, P, N.

Use Definition 1449.3. It forces a cover-facing read of a family member or bound assertion to pass through the same StreamName, RegSeqRat, RealUp, and ClosedBoundedIntervalNet route.

Now apply Definition 1449.4 and Definition 1449.5. They exclude open-cover compactness, cover selection, and hidden choice from the displayed bounded-family rows. Hence a cover-selection consumer must provide a separate finite-cover or finite-net sibling row before any cover conclusion is exported. □

Falsifiable prediction: Every $\mathsf{BoundedRealFamily}^{\uparrow}$ Real-facing read factors through finite index windows, StreamName windows, RegSeqRat readback, RealUp family seals, and a ClosedBoundedIntervalNet bound row.

Independence witness: boundedrealsequence, realcauchymodulus, dyadicintervalcover, closedboundedintervalnet, real: $\mathsf{BoundedRealFamily}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an index-window family ledger, StreamName windows, RegSeqRat readback, RealUp family seals, a common bound ledger, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealFamily}^{\uparrow}$ packet is generated from finite index-window rows, StreamName windows, RegSeqRat readback, RealUp family seals, a ClosedBoundedIntervalNet bound row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1449.1, Definition 1449.2, Definition 1449.3, Definition 1449.4, Definition 1449.5, Definition 1449.6, Definition 1449.7, Definition 1449.8, and Definition 1449.9 bind the finite bounded-family obligation surface, scoped dependency route, and public compactness, least-upper-bound, and cover-selection consumer boundaries over StreamNameUp, RegSeqRatUp, RealUp, and ClosedBoundedIntervalNetUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealFamily}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.BoundedRealFamilyUp.BoundedRealFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, trichotomy, quotient equality of real names, countable choice, an ambient supremum principle, open-cover compactness, or host completeness.

Upgrade path: Bridge closure requires the standard bridge row needed by downstream compactness, least-upper-bound, and cover-selection routes.

Chapter 1450

A Concrete Naming Certificate for $\mathsf{MetricBall}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{MetricBall}^{\uparrow}$ packet records the center-radius open ball used by metric-generated topology, but as finite BEDC NameCert rows rather than as an ambient subset. It extracts the singleton-ball datum used inside $\mathsf{Topology}^{\uparrow}$ into a reusable surface for topology, continuous maps, compact nets, and metric arguments that need the same carrier before generated-open exactness is invoked. The center-radius row is the shared dependency surface read by chapter 57, chapter 88, chapter 1757, and chapter 38; each consumer receives the same displayed distance and positive-radius ledger rather than a host subset.

Definition 1450.1 (MetricBall carrier)**.** *A* $\mathsf{MetricBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, d, c, r, \rho, m, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row with its point classifier and metric source data.*
2. *d is the* $\mathsf{MetricDistanceWitness}$ *row read from X and valued in the* $\mathsf{Real}^{\uparrow}$ *distance surface.*
3. *c is the center row, classified as a point of the carrier named by X.*
4. *r is a* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius row, according to the chosen radius ledger.*
5. *ρ is the positive-radius ledger for r, transported through the ordered-real comparison used by the metric source.*
6. *m is the membership and distance-comparison row. For a displayed point x, it records the comparison $d(c, x) <_{\mathsf{R}} r$ and names this comparison as the* $\mathsf{MetricOpenBall}_d(c, r, \rho)$ *membership read.*
7. *H records the* hsame *transports for the carrier, distance witness, center, radius, positive-radius ledger, and membership-comparison row.*
8. *C records the* Cont *routes that replay distance evaluation, radius comparison, membership transport, finite-base refinement, and singleton-ball handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$ *row.*

The accepted carrier contains no ambient powerset coordinate, no unindexed open predicate, no quotient equality test on points or radii, and no generated topology closure field.

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallTasteGate single carrier alignment`

Theorem 1450.2 (MetricBall NameCert obligations)**.** *For an accepted packet $B = (X, d, c, r, \rho, m, H, C, P, N)$, the local $\mathsf{NameCert}_{\mathsf{MetricBall}^\uparrow}$ obligation surface consists exactly of carrier admission, classifier transport, radius positivity stability, membership transport, finite refinement and intersection readiness, and ledger non-escape for the displayed rows $X, d, c, r, \rho, m, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier namecert obligations`

Proof. Carrier admission reads X as the metric carrier and d as the distance witness over that carrier. The center row c and radius row r are admitted only relative to those source rows, and ρ supplies the displayed positive-radius ledger.

Classifier transport is supplied by H. Transporting the carrier, distance witness, center, radius, and positive-radius rows moves the comparison data in m to the transported copy without introducing a second point classifier or a second radius equality test.

Membership transport is the distance-comparison part of the same ledger. The row m names the predicate $d(c, x) <_{\mathsf{R}} r$, while C replays the route from distance evaluation to ordered-real comparison.

Finite refinement and intersection readiness are only readiness claims. They expose the singleton-ball row and its replayable comparison so that a finite list of such rows can later be concatenated by $\mathsf{Topology}^\uparrow$, but they do not themselves close arbitrary generated opens.

Ledger non-escape follows by inspecting the carrier. Every exported read is one of $X, d, c, r, \rho, m, H, C, P, N$, so no ambient powerset, unindexed open predicate, quotient equality test, or generated topology closure field can be recovered from the packet. □

Theorem 1450.3 (MetricBall refinement obligations)**.** *For an accepted $\mathsf{MetricBall}^\uparrow$ packet $B = (X, d, c, r, \rho, m, H, C, P, N)$, the refinement-facing obligation surface supplies exactly three additional rows. The first row transports positive-radius data through the $\mathsf{Rat}^\uparrow$ or $\mathsf{Real}^\uparrow$ radius ledger before any consumer reads the comparison $d(c, x) <_{\mathsf{R}} r$. The second row refines the finite base into the topology singleton-ball index of Definition 88.10. The third row routes bounded-set containment through $\mathsf{BoundedSet}^\uparrow$ without replacing the center-radius packet by a host subset, selected supremum, compactness proof, or ambient real completeness.*

Proof. Start with the carrier and NameCert obligations of Definition 1450.1 and Definition 1450.2. They expose the carrier row, distance witness, center, radius, positive-radius ledger, membership comparison, transport, replay, provenance, and local name as the only rows of the packet.

The positive-radius row is the transport route of Definition 1450.6. It keeps r, ρ, and m together through the ordered-real comparison used by the metric source, so the $\mathsf{Rat}^\uparrow$ or $\mathsf{Real}^\uparrow$ radius cannot be read independently of its positivity evidence.

The finite-base refinement row is the topology handoff of Definition 1450.4. It sends the center-radius datum to the singleton-ball index in Definition 88.10, and then to the singleton carrying theorem already cited by that handoff.

The bounded containment row is the route of Definition 1450.5. It hands the same transported comparison to $\mathsf{BoundedSet}^\uparrow$ and leaves no coordinate for a host subset, selected maximum, compactness witness, or ambient $\mathsf{Real}^\uparrow$ completeness. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier refinement obligations`

Theorem 1450.4 (MetricBall topology handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$ *metric-generated singleton-ball open read factors through a* $\mathsf{MetricBall}^{\uparrow}$ *packet before it enters generated-open exactness. In particular, the singleton ball datum of Definition 88.10 and the singleton carrying theorem of Definition 88.11 are read as the* $X, d, c, r, \rho, m, H, C, P, N$ *rows of* $\mathsf{MetricBall}^{\uparrow}$ *and only then handed to the indexed topology carrier.*

Proof. Start with a metric-generated singleton-ball open in $\mathsf{Topology}^{\uparrow}$. By Definition 88.10, its datum is a center, a positive radius, and the open-ball predicate obtained from the distance comparison.

Package these entries as X, d, c, r, ρ, m. The metric carrier and distance witness supply X and d, the singleton datum supplies c, r, ρ, and the predicate $\mathsf{MetricOpenBall}_d$ supplies the membership-comparison row m.

Add the structural rows H, C, P, N. They are exactly the transports, continuation replays, provenance, and local name row required by Definition 1450.1.

Now apply Definition 88.11. Its singleton carrying equation reads the same membership row and places the singleton index inside $\mathcal{T}_{d,\mathsf{ball}}$. Thus the generated-open exactness package receives a carried singleton-ball index only after the $\mathsf{MetricBall}^{\uparrow}$ packet has exposed the center-radius NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier topology handoff`

Theorem 1450.5 (MetricBall bounded-set containment route)**.** *Every* $\mathsf{BoundedSet}^{\uparrow}$ *membership read that uses a* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger first exposes the* $\mathsf{MetricBall}^{\uparrow}$ *rows* $X, d, c, r, \rho, m, H, C, P, N$*. The bounded-set consumer receives the center-radius comparison only through that packet, not through a host set, supremum, compactness, selected maximum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1450.1. The row m is already the named comparison $d(c, x) <_{\mathsf{R}} r$, and the structural rows H, C, P, N are the only transports, replays, provenance, and local naming data attached to that comparison.

Now read the downstream bounded-set carrier of chapter 1757. Its containment coordinate is a $\mathsf{MetricBall}^{\uparrow}$ ledger, so a membership read reaches bounded-set data only after the center, radius, positive-radius ledger, and comparison row have been opened in the MetricBall packet.

Thus the containment ledger handed to $\mathsf{BoundedSet}^{\uparrow}$ is the same MetricBall row already available to topology, boundedness, diameter, and uniform-space consumers. Since the packet has no coordinate for a host subset, supremum, compactness proof, selected maximum, or ambient real completeness, none of those objects can supply the bounded-set containment route. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier boundedset containment route`

Theorem 1450.6 (MetricBall positive-radius transport)**.** *Let* $B = (X, d, c, r, \rho, m, H, C, P, N)$ *be an accepted* $\mathsf{MetricBall}^{\uparrow}$ *packet. If* B *is transported componentwise, then the center row* c*, distance witness* d*, radius row* r*, positive-radius ledger* ρ*, and membership comparison* m *transport together before any* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{BoundedSet}^{\uparrow}$ *consumer reads the ball. The dependency route is anchored in the metric source of chapter 57, the topology handoff of chapter 88, the bounded-set containment surface of chapter 1757, and the ordered-real comparison row of chapter 38.*

Proof. Project the carrier of Definition 1450.1. The row H transports exactly the carrier, distance witness, center, radius, positive-radius ledger, and membership-comparison rows named by X, d, c, r, ρ, m.

Apply the NameCert obligations of Definition 1450.2. Classifier transport, radius positivity stability, membership transport, and ledger non-escape all refer to the same displayed rows, so the transported center-radius comparison cannot split into separate topology and bounded-set readings.

For a topology consumer, use Definition 1450.4. The singleton-ball datum enters generated-open exactness only after $X, d, c, r, \rho, m, H, C, P, N$ have been packaged as the $\mathsf{MetricBall}^{\uparrow}$ surface.

For a bounded-set consumer, use Definition 1450.5. The containment ledger likewise reads the transported center, radius, positive-radius ledger, and comparison row before it can serve the downstream $\mathsf{BoundedSet}^{\uparrow}$ membership request. Thus both sibling consumers receive the same transported positive-radius route, with the metric and real dependency anchors still visible. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier positive radius transport`

Theorem 1450.7 (MetricBall finite-intersection readiness)**.** *Let*

$$B_0 = (X, d, c_0, r_0, \rho_0, m_0, H_0, C_0, P_0, N_0)$$

and

$$B_1 = (X, d, c_1, r_1, \rho_1, m_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{MetricBall}^{\uparrow}$ *packets over the same* $\mathsf{Met}^{\uparrow}$ *row. If their center, radius, positive-radius, and membership rows have been transported componentwise, then* $\mathsf{Topology}^{\uparrow}$ *may read the pair as a finite-intersection request only through the displayed rows* $X, d, c_i, r_i, \rho_i, m_i, H_i, C_i, P_i, N_i$*, the ordered-real positive-radius ledgers* ρ_i*, and the singleton-ball indexed-open route of Definition 88.10. The request has no coordinate for host powersets, selected radii, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1450.2 to each packet. It exposes the carrier, distance witness, center, radius, positive-radius ledger, membership comparison, transport, replay, provenance, and local name as the only available rows.

Apply Definition 1450.3. The finite-base row is already constrained to the topology singleton-ball index, while the bounded-set row remains a separate containment route and supplies no host subset.

Use Definition 1450.4 for the topology consumer. Each singleton-ball datum enters $\mathsf{Topology}^{\uparrow}$ through the indexed-open route of Definition 88.10, so the intersection request is a finite request over two displayed singleton-ball indices rather than a powerset operation.

Finally apply Definition 1450.6 to both packets. The transported rows keep c_i, r_i, ρ_i, m_i together through the ordered-real ledger, so no selected radius, compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle can be read from the finite-intersection request. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier finite intersection readiness`

Theorem 1450.8 (MetricBall obligation transport package)**.** *Every topology, bounded-set, finite-intersection, or compact-net consumer of an accepted* $\mathsf{MetricBall}^{\uparrow}$ *packet*

$$B = (X, d, c, r, \rho, m, H, C, P, N)$$

must read the packet through the same displayed $\mathsf{Met}^{\uparrow}$ *source row* X*, metric-distance row* d*, center row* c*, positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius rows* r, ρ*, membership comparison row* m*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$ *row* N*. The transport package has no coordinate for an ambient powerset, a selected replacement radius, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier obligation transport package`

Proof. Start with the carrier of Definition 1450.1. It displays exactly $X, d, c, r, \rho, m, H, C, P, N$, so every consumer first receives the metric source, distance witness, center, radius, positive-radius ledger, and membership comparison before any downstream read is available.

For topology consumers, apply Definition 1450.4. The singleton-ball datum reaches generated-open exactness only after the MetricBall rows have been packaged.

For bounded-set consumers, apply Definition 1450.5. The containment ledger reads the same center-radius comparison and does not introduce a host subset or supremum coordinate.

For finite-intersection and compact-net consumers, use Definition 1450.7 together with Definition 1450.6. These rows keep r, ρ, m attached to the same metric source and ordered-real comparison before any finite request is replayed.

Thus the shared obligation package is the displayed tuple $X, d, c, r, \rho, m, H, C, P, N$. Since the carrier contains no powerset, selected radius, compactness, or ambient completeness coordinate, none can be exported by transport. □

Theorem 1450.9 (MetricBall RegSeqRat Real window boundary)**.** *Every* $\mathsf{MetricBall}^{\uparrow}$ *read over* $\mathsf{Real}^{\uparrow}$*-facing points must display the regular rational readback and real-seal route before any ball-membership answer is exported. Concretely, a consumer receives the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, the* Cont *replay route,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$ *rows together with the carrier rows* X, d, c, r, ρ, m *of Definition 1450.1. The boundary is anchored in the finite-window L10 rows of chapter 301, chapter 291, and chapter 38; it exports no host-real equality, unrestricted metric topology, or hidden completion principle.*

Proof. Project the carrier of Definition 1450.1. The membership row m is not a bare subset predicate; it is the named comparison $d(c, x) <_{\mathsf{R}} r$ transported with $X, d, c, r, \rho, H, C, P, N$.

For a real-facing point, the comparison can be consumed only after the finite real-name route has been displayed. The $\mathsf{RegSeqRat}^{\uparrow}$ readback supplies the inspected rational observation, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ row supplies the terminal seal described by the L10 sibling chapters.

Apply Definition 1450.6. It keeps the center, distance witness, radius, positive-radius ledger, and membership comparison together under hsame transport before topology or bounded-set consumers read the ball.

Finally use Definition 1450.2. Ledger non-escape says that every exported read remains inside the displayed rows, so the real-facing ball answer cannot bypass the regular readback, dyadic tolerance, real seal, continuation replay, provenance, or local NameCert boundary. □

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier regseqrat real window boundary`

Theorem 1450.10 (MetricBall Cauchy-filter entry boundary)**.** *Completion and filter consumers may use* $\mathsf{MetricBall}^{\uparrow}$ *only as a finite tolerance-window entry row. The entry row exposes the carrier, distance witness, center, radius, positive-radius ledger, membership comparison,* hsame *transport,* Cont *route,* Pkg *provenance, and local* $\mathsf{NameCert}$ *data of Definition 1450.1, together with the finite-window real readback boundary of Definition 1450.9. It does not provide generated-open topology, arbitrary metric topology, a selected center outside the displayed row, quotient equality, compactness, or any unbounded filter object.*

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier cauchy filter entry boundary`

Proof. Start from the finite-entry view of Definition 1450.9. A completion or filter consumer sees the real-facing membership comparison only after the regular rational readback, dyadic tolerance, real seal, transport, replay, provenance, and local NameCert rows have been displayed.

Use Definition 1450.3 to locate the topology-facing part of the packet. The finite-base row is a singleton-ball index for Definition 88.10; it is not generated-open closure and does not carry arbitrary metric topology.

Use Definition 1450.7 for finite combinations. That theorem permits only a finite request over displayed singleton-ball indices, so a filter consumer may record a tolerance-window entry but cannot extract compactness, quotient equality, selected centers, or an unbounded family of opens.

The bounded-set route of Definition 1450.5 gives the same restriction from the containment side. It hands downstream consumers the transported center-radius comparison and no host subset, selected maximum, compactness witness, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 1450.11 (MetricBall public export)**.** *The public* $\mathsf{MetricBall}^{\uparrow}$ *export is the center-radius membership packet*

$$B = (X, d, c, r, \rho, m, H, C, P, N)$$

of Definition 1450.1. Topology, bounded-set, finite-intersection, compact-net, and metric consumers may consume the packet only through the displayed distance-comparison row m*, together with its metric source* X, d*, center* c*, radius and positivity rows* r, ρ*, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.MetricBallUp.MetricBallCarrier public export`

Proof. Project the carrier in Definition 1450.1. The only membership-facing coordinate is the comparison row m, and the rows $X, d, c, r, \rho, H, C, P, N$ are precisely the data that make that comparison transportable and replayable.

For topology consumers, apply Definition 1450.4. For bounded-set consumers, apply Definition 1450.5. For finite-intersection and metric-entry consumers, use Definition 1450.7 and Definition 1450.10.

Each route reads the same center-radius comparison before any downstream object is available. Since the carrier has no ambient powerset, selected radius, compactness, host subset, quotient equality, or Real completeness coordinate, the displayed packet is exactly the public export surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricBall}^{\uparrow}$ packet is generated from a MetricSpace carrier row, a MetricDistanceWitness row, a center row, a RatUp or RealUp positive-radius row, a membership comparison row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1450.11 exports the center-radius membership packet of Definition 1450.1, with Definition 1450.2, Definition 1450.3, Definition 1450.4, and Definition 1450.7 displaying the obligation surface over $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a full metric topology, arbitrary generated-open closure, compactness, continuity, quotient equality for metric points, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, metric membership transport, positive-radius stability, finite-base handoff, finite-intersection readiness, and ledger exactness for BEDC.Derived.MetricBallUp.

Chapter 1451

A Concrete Naming Certificate for $\mathsf{MetricClosedBall}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{MetricClosedBall}^{\uparrow}$ packet records center-radius closed balls as finite BEDC NameCert rows. It sits beside the open-ball carrier of chapter 1450, but its membership row is read through non-strict ordered-real comparison over MetricDistanceWitness, Real/Rat order, hsame, Cont, Pkg, and NameCert rows rather than through an ambient closed subset.

Definition 1451.1 (MetricClosedBall carrier)**.** *A* $\mathsf{MetricClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, d, c, r, \rho, m, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row with its point classifier and metric source data.*
2. *d is the* MetricDistanceWitness *row read from X and valued in the* $\mathsf{Real}^{\uparrow}$ *distance surface.*
3. *c is the center row, classified as a point of the carrier named by X.*
4. *r is a* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius row, according to the chosen radius ledger.*
5. *ρ is the radius order ledger for r, recording nonnegativity and, when the packet is handed to an open-ball consumer, the separate positive-radius comparison needed by* $\mathsf{MetricBall}^{\uparrow}$*.*
6. *m is the closed-membership and distance-comparison row. For a displayed point x, it records the comparison $d(c, x) \leq_{\mathsf{R}} r$ and names this comparison as the* $\mathsf{MetricClosedBall}_d(c, r, \rho)$ *membership read.*
7. *H records the* hsame *transports for the carrier, distance witness, center, radius, radius order ledger, and closed-membership comparison row.*
8. *C records the* Cont *routes that replay distance evaluation, non-strict radius comparison, closed-membership transport, and boundary handoff to open-ball rows when a strict comparison is separately supplied.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$ *row.*

The accepted carrier contains no ambient powerset coordinate, no topology-closure predicate, no quotient equality test on points or radii, no compactness witness, and no generated-open field.

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallTasteGate single carrier alignment`

Theorem 1451.2 (MetricClosedBall NameCert obligations)**.** *For an accepted packet $Q = (X, d, c, r, \rho, m, H, C, P, N)$ the local* $\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, classifier transport, radius nonnegativity stability, positive-radius stability for strict-boundary handoff when that handoff is requested, closed-membership transport, boundary comparison with* $\mathsf{MetricBall}^{\uparrow}$*, and ledger non-escape for the displayed rows $X, d, c, r, \rho, m, H, C, P, N$.*

Proof. Carrier admission reads X as the metric carrier and d as the distance witness over that carrier. The center row c and radius row r are admitted only relative to those source rows, while ρ supplies the displayed ordered-real radius ledger.

Classifier transport is supplied by H. Transporting the carrier, distance witness, center, radius, and radius ledger moves the comparison data in m to the transported copy without introducing a second point classifier or a second radius equality test.

Closed-membership transport is the non-strict distance-comparison part of the same ledger. The row m names the predicate $d(c, x) \leq_{\mathsf{R}} r$, while C replays the route from distance evaluation to ordered-real comparison.

Boundary comparison with $\mathsf{MetricBall}^{\uparrow}$ is conditional on a strict comparison row. When an open-ball consumer supplies the positive-radius and strict-membership data required by Definition 1450.1, the route passes through $\mathsf{MetricBall}^{\uparrow}$; otherwise the closed-ball packet exports only its non-strict comparison row.

Ledger non-escape follows by inspecting the carrier. Every exported read is one of $X, d, c, r, \rho, m, H, C, P, N$, so no ambient subset, topology closure, quotient equality test, compactness witness, or generated-open field can be recovered from the packet. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier namecert obligations`

Theorem 1451.3 (MetricClosedBall boundary handoff)**.** *Every open-ball consumer of a center-radius strict comparison passes through* $\mathsf{MetricBall}^{\uparrow}$*, while every closed-ball consumer of $Q = (X, d, c, r, \rho, m, H, C, P, N)$ can read only the non-strict comparison row $d(c, x) \leq_{\mathsf{R}} r$ together with its transports and replay routes. The* $\mathsf{MetricClosedBall}^{\uparrow}$ *packet therefore does not recover ambient subsets, quotient point equality, compactness, or topology closure.*

Proof. Start with the accepted closed-ball carrier of Definition 1451.1. Its membership coordinate is m, and m records only the non-strict comparison $d(c, x) \leq_{\mathsf{R}} r$.

If a downstream consumer asks for a strict center-radius ball, the required predicate is not m. It is the open comparison row used by Definition 1450.1, so the request factors through the $\mathsf{MetricBall}^{\uparrow}$ packet and its positive-radius ledger.

If the downstream consumer remains closed-ball facing, the only exported comparison is the non-strict row transported by H and replayed by C. Package provenance P and the local naming row N identify this as the $\mathsf{MetricClosedBall}^{\uparrow}$ surface, not as a topology closure or compactness object.

Thus the handoff is exact at the boundary: strict consumers use $\mathsf{MetricBall}^{\uparrow}$, and closed consumers receive no coordinate from which to reconstruct an ambient subset, quotient point equality, compactness proof, or generated topology closure. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier metricball boundary handoff`

Theorem 1451.4 (MetricClosedBall carrier obligations)**.** *For an accepted* $\mathsf{MetricClosedBall}^{\uparrow}$ *packet* $Q = (X, d, c, r, \rho, m, H, C, P, N)$*, carrier admission is exactly the simultaneous admission of the* $\mathsf{Met}^{\uparrow}$ *row* X*, the* MetricDistanceWitness *row* d*, the center row* c*, the radius row* r*, the nonnegative radius ledger* ρ*, the closed-membership comparison row* m*, and the support rows* H, C, P, N*. The packet exports no ambient closed subset, topology-closure predicate, compactness witness, quotient point equality, quotient radius equality, or generated-open field.*

Proof. Read the carrier tuple of Definition 1451.1. The rows X, d, c, r, ρ, m are the only mathematical rows admitted before the structural rows appear.

The support rows do not add a second carrier. The row H transports those displayed coordinates, C replays their distance and comparison route, P records the named provenance, and N gives the local certificate name.

Since no tuple coordinate stores an ambient subset, closure predicate, compactness object, quotient equality test, or generated-open field, none of those objects can be projected from an accepted packet. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier obligations`

Theorem 1451.5 (MetricClosedBall radius classifier stability)**.** *Let* $Q = (X, d, c, r, \rho, m, H, C, P, N)$ *and* $Q' = (X', d', c', r', \rho', m', H', C', P', N')$ *be accepted* $\mathsf{MetricClosedBall}^{\uparrow}$ *packets whose carrier, distance, center, radius, and radius-ledger rows are paired by the displayed componentwise* hsame *data in* H *and* H'*. Then the closed-membership classifier transported by* H *and replayed by* C *carries the non-strict comparison row* $d(c, x) \leq_{\mathsf{R}} r$ *to the corresponding row* $d'(c', x') \leq_{\mathsf{R}} r'$*. The transport preserves the radius nonnegativity ledger and the optional positive-radius handoff data used by* $\mathsf{MetricBall}^{\uparrow}$*.*

Proof. Start from the displayed hsame rows. They pair the metric carrier, distance witness, center, radius, and radius ledger componentwise, so the radius classifier is not transported by a host equality of reals or by a quotient radius comparison.

Now replay the closed-membership route. The row C reads distance evaluation first and then the non-strict ordered-real comparison in m, so transporting d, c, r, ρ transports the comparison row itself.

For strict-boundary consumers, the positive-radius data remain a separate part of ρ. Thus the same transport preserves the closed non-strict classifier and, when present, the positive-radius handoff required before the request can pass to $\mathsf{MetricBall}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier radius classifier stability`

Theorem 1451.6 (MetricClosedBall RealAlgOrder boundary)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *boundary read made by an accepted* $\mathsf{MetricClosedBall}^{\uparrow}$ *packet factors through the displayed distance witness* d*, the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$ *radius row* r*, the radius ledger* ρ*, and the closed-membership comparison row* m*. If a downstream consumer asks for a strict boundary comparison, the route first passes through the* $\mathsf{MetricBall}^{\uparrow}$ *handoff of Definition 1451.3; otherwise it remains the non-strict RealAlgOrder comparison* $d(c, x) \leq_{\mathsf{R}} r$ *carried by* m*.*

Proof. Inspect the comparison coordinates of Q. The distance side is the $\mathsf{Real}^{\uparrow}$-valued witness row d, the radius side is r, and the admissible radius facts are exactly those recorded in ρ.

The non-strict order read is m. It routes the displayed distance value and displayed radius through the RealAlgOrder or Rat-order rows named by P, then returns the membership classifier $d(c, x) \leq_{\mathsf{R}} r$.

If a strict comparison is requested, the packet must use the separate positive-radius and strict-membership data described in Definition 1451.3. That request therefore factors through $\mathsf{MetricBall}^{\uparrow}$ and does not create a second closed-ball boundary.

Thus every RealAlgOrder-facing read is either the displayed non-strict comparison row m or the explicit $\mathsf{MetricBall}^{\uparrow}$ strict-boundary handoff. No host closed subset, topology closure, quotient point equality, compactness witness, or generated-open field is exported. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier realalgorder boundary`

Theorem 1451.7 (MetricClosedBall boundary classifier transport)**.** *Let $Q = (X, d, c, r, \rho, m, H, C, P, N)$ and $Q' = (X', d', c', r', \rho', m', H', C', P', N')$ be accepted $\mathsf{MetricClosedBall}^{\uparrow}$ packets whose carrier, distance, center, radius, and radius-ledger rows are classified componentwise by the displayed* hsame *data. Then the boundary row m carrying $d(c, x) \leq_{\mathsf{R}} r$ transports through H and replays through C to the boundary row m' carrying $d'(c', x') \leq_{\mathsf{R}} r'$. The transport introduces no topology closure, compactness witness, quotient point equality, or ambient subset data.*

Proof. Project the two packets. The only rows that can affect the closed boundary classifier are the metric carrier, distance witness, center, radius, radius ledger, membership row, and their transport and continuation data.

Use the classifier definition from Definition 1451.1. The row m is precisely the non-strict comparison $d(c, x) \leq_{\mathsf{R}} r$, and the corresponding row in Q' is m'.

Apply radius classifier stability from Definition 1451.5. The componentwise hsame data preserve the radius ledger and the comparison row under transport.

Finally replay the RealAlgOrder boundary route of Definition 1451.6. Since the carrier inventory has no topology-closure, compactness, quotient-point, or ambient-subset coordinate, the transported classifier remains exactly the displayed closed-boundary row. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier boundary classifier transport`

Theorem 1451.8 (MetricClosedBall radius source scope)**.** *For an accepted $\mathsf{MetricClosedBall}^{\uparrow}$ packet $Q = (X, d, c, r, \rho, m, H, C, P, N)$, the radius row r is scoped only through the $\mathsf{Rat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ positivity data displayed by ρ. Every closed-membership read must project r, ρ from the same* BHist *carrier, transport them componentwise by H, replay their nonnegative ordered-real route by C, and source the comparison through P, N before using the boundary row m.*

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier radius source scope`

Proof. Project the carrier. The row r is not an ambient real parameter; it is paired with the radius ledger ρ inside the packet.

Use the NameCert obligation theorem to read ρ as the displayed nonnegative radius evidence, with the optional positive-radius handoff remaining separate for strict open-ball consumers.

Transport by H and replay by C. These rows preserve the same RatUp or RealUp positivity source and keep the closed-boundary comparison inside the named carrier rather than inside a quotient radius equality or topology-closure field. □

Theorem 1451.9 (MetricClosedBall center transport scope)**.** *Let $Q = (X, d, c, r, \rho, m, H, C, P, N)$ and $Q' = (X', d', c', r', \rho', m', H', C', P', N')$ be accepted $\mathsf{MetricClosedBall}^{\uparrow}$ packets related by displayed componentwise* hsame *rows. The center row c may be transported only along the $\mathsf{Met}^{\uparrow}$ carrier classifier named by X and X', and the transported boundary read remains stable only after the distance witness, radius ledger,* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been read.*

Proof. Read X and X' first. They are the only carrier rows that classify the centers, so center transport cannot be detached from the underlying $\mathsf{Met}^{\uparrow}$ classifier.

Use the boundary classifier transport theorem. It carries c together with the distance witness, radius row, radius ledger, and membership comparison; H, H' supply the componentwise identifications and C, C' replay the distance route.

The provenance rows P, P' and naming rows N, N' keep the transported center inside the same MetricClosedBall surface. No ambient subset, generated topology, or quotient point equality is introduced. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier center transport scope`

Theorem 1451.10 (MetricClosedBall RealAlgOrder boundary scope)**.** *For every accepted* $\mathsf{MetricClosedBall}^{\uparrow}$ *packet* $Q = (X, d, c, r, \rho, m, H, C, P, N)$*, the boundary inequality row* m *is scoped to the RealAlgOrder or Rat-order comparison named by the distance witness* d*, the radius row* r*, and the ledger* ρ*. Apartness-compatible strict comparisons may enter only through the separate* $\mathsf{MetricBall}^{\uparrow}$ *handoff; the closed-ball packet itself exports the non-strict comparison by* BHist *projection,* hsame *transport,* Cont *replay, and* $\mathsf{NameCert}$ *provenance.*

Proof. Inspect m. It records the non-strict boundary comparison $d(c, x) \leq_{\mathsf{R}} r$ and has no coordinate for an ambient closed subset.

Replay the RealAlgOrder boundary theorem. The comparison consumes d, r, ρ and then returns the closed-membership classifier through the same displayed order route.

If an apartness-compatible strict comparison is needed, use the explicit MetricBall boundary handoff. That handoff is separate from m, so the closed-ball scope remains the displayed non-strict boundary route and does not become compactness, topology closure, or quotient equality. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier realalgorder boundary scope`

Theorem 1451.11 (MetricClosedBall bridged export)**.** *For every accepted* $\mathsf{MetricClosedBall}^{\uparrow}$ *packet* $Q = (X, d, c, r, \rho, m, H, C, P, N)$*, the public export bridges the non-strict metric boundary row to any abstract closed-ball consumer exactly through* $X, d, c, r, \rho, m, H, C, P, N$*. Strict-radius consumers do not read this export directly: they factor through the sibling* $\mathsf{MetricBall}^{\uparrow}$ *handoff and the displayed RealAlgOrder comparison rows before returning to the closed-ball boundary.*

Proof. Apply Definition 1451.2. It fixes the carrier admission, classifier transport, radius ledger, closed-membership transport, boundary comparison, and ledger non-escape rows of Q.

Apply Definition 1451.3. It separates strict-radius consumers from the non-strict closed-ball export, so the strict route must pass through $\mathsf{MetricBall}^{\uparrow}$ rather than reading the closed boundary as an open ball.

Apply Definition 1451.6. It identifies the closed export as the RealAlgOrder comparison row carried by d, r, ρ, m and transported through H, C, P, N. Thus the bridge-facing consumer receives exactly the displayed packet rows and no ambient closed subset, quotient point equality, compactness witness, or topology closure. □

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallCarrier bridged export`

Definition 1451.12 (MetricClosedBall mature boundary inventory)**.** *The mature* $\mathsf{MetricClosedBall}^{\uparrow}$ *export inventory is exactly the finite list consisting of the* $\mathsf{Met}^{\uparrow}$ *carrier row, the* $\mathsf{MetricDistanceWitness}$ *row, the center row, the radius row, the radius ledger, the non-strict boundary comparison row, the strict-boundary handoff through* $\mathsf{MetricBall}^{\uparrow}$ *when separately supplied, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Rat}^{\uparrow}$

order source, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$ *row. No mature export includes compactness, a full metric topology, topology closure, quotient equality for points or radii, ambient subset data, Lean verification, or bridge checking.*

Lean4 (definition) `BEDC.Derived.MetricClosedBallUp.MetricClosedBallMatureBoundaryInventory`

Lean4 ✓ `BEDC.Derived.MetricClosedBallUp.MetricClosedBallMatureBoundaryInventoryCertificate`

Closure status for $\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricClosedBall}^{\uparrow}$ packet is generated from a MetricSpace carrier row, a MetricDistanceWitness row, a center row, a RatUp or RealUp radius row, a nonnegative radius ledger with optional positive-radius handoff data, a closed-membership comparison row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1451.8, Definition 1451.9, Definition 1451.10, Definition 1451.11, and Definition 1451.12 export the mature $\mathsf{MetricClosedBall}^{\uparrow}$ boundary inventory through carrier admission, closed-membership transport, radius nonnegativity, strict-boundary handoff, order-source routing, and ledger non-escape.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricClosedBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim compactness, full metric topology, topology closure, quotient equality for metric points or radii, ambient subsets, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode the same carrier admission, closed-membership transport, radius nonnegativity, strict-boundary handoff, and ledger non-escape obligations for BEDC.Derived.MetricClosedBallUp.

Chapter 1452

A Concrete Naming Certificate for $\mathsf{LocatedInfimum}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedInfimum}^{\uparrow}$ packet records greatest-lower-bound data by finite lower-bound ledgers and upper-approximation rows, before any ambient completeness principle for $\mathsf{Real}^{\uparrow}$ is available. It is the lower-bound companion to the $\mathsf{LocatedSupremum}^{\uparrow}$ certificate of chapter 1427 and consumes the bounded-family source supplied by chapter 1449. The carrier therefore gives later interval and bounded-family arguments a finite NameCert target for infimum-shaped reasoning while keeping quotient real equality, trichotomy, countable choice, and host completeness outside the packet.

Definition 1452.1 (LocatedInfimum carrier)**.** *A* $\mathsf{LocatedInfimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, L, G, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded real family row, read through the finite index-window,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and common-bound ledgers of* $\mathsf{BoundedRealFamily}^{\uparrow}$.
2. *L is the lower-bound ledger. It records the inspected lower-bound claims and the finite comparison rows that certify each requested member of F above the proposed lower surface.*
3. *G is the greatest-lower-approximation ledger. For each precision request in the displayed window, it stores a finite family-side witness above the proposed lower surface and below every displayed stronger lower-bound candidate at that precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *precision-window row that fixes the finite observations of the family, lower-bound ledger, and greatest-lower-approximation ledger in scope.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the windowed observations as regular rational readback data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through F, L, G, W, R.*
7. *H records componentwise* hsame *transports for the family row, lower-bound ledger, greatest-lower-approximation ledger, precision window, regular handoff, real seal, continuation replay, provenance, and local naming row.*
8. *C records displayed* Cont *routes that replay lower-bound reads, greatest-lower-approximation reads, classifier transport, regular-rational handoff, and real-seal readback.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedInfimum}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{BoundedRealFamily}^{\uparrow}$, BishopRealComparison StreamName$^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$ *row.*

The accepted carrier stores no Dedekind cut completion, no arbitrary infimum operator, no quotient real equality test, no trichotomy oracle, no countable-choice functional, and no host-completeness coordinate.

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.TasteGate.LocatedInfimumTasteGate single carrier alignment`

Theorem 1452.2 (LocatedInfimum lower-bound obligation)**.** *For an accepted packet $I = (F, L, G, W, R, E, H, C, P$ every lower-bound read exposed by the local* $\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$ *row factors through the bounded-family row F, the lower-bound ledger L, the precision window W, the regular-rational handoff R, and the* $\mathsf{Real}^{\uparrow}$ *seal E, together with the transport, continuation, provenance, and local naming rows. No lower-bound obligation may invoke Dedekind completeness, an arbitrary infimum operator, quotient real equality, trichotomy, countable choice, or host completeness.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumLowerBoundObligation`

Proof. Project the accepted carrier. The only family source is F, and the only lower-bound source is L, so a lower-bound consumer must first inspect the bounded-family row and then one of the displayed comparison entries of L.

The precision window W determines which finite observations are admitted. The handoff row R repacks those observations as regular rational data, and the seal E is reached only after this route has been replayed.

The structural rows cannot enlarge the obligation surface. The transports in H, continuations in C, provenance in P, and name row N preserve the displayed route, but none carries a Dedekind completion, arbitrary infimum operator, quotient equality test, trichotomy oracle, countable-choice functional, or host-completeness field. □

Theorem 1452.3 (LocatedInfimum greatest-lower-bound obligation)**.** *For an accepted packet $I = (F, L, G, W, R, E, H, C, P, N)$, the greatest-lower-bound obligation is exactly the finite demand that each precision row of W receives a displayed witness in G whose readback lies above the proposed lower surface and blocks every displayed stronger lower-bound candidate at that precision. The obligation is local to $F, L, G, W, R, E, H, C, P, N$ and does not assert a global infimum for arbitrary real families.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumGreatestLowerBoundObligation`

Proof. Fix a precision row in W. By definition of the carrier, G is the only ledger that can answer greatest-lower-approximation requests at that precision.

Read the corresponding G entry. It supplies a finite family-side witness, and the route through R and E gives the regular-rational and real-name readback for the same displayed observation.

The comparison is therefore local: the witness is checked against the lower surface and against the stronger lower-bound candidates that are actually present in the window. Since the packet has no coordinate for arbitrary families or an ambient completeness principle, the row proves only the displayed finite greatest-lower-bound obligation. □

Theorem 1452.4 (LocatedInfimum precision-window induction)**.** *For an accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet $I = (F, L, G, W, R, E, H, C, P, N)$, induction over the finite precision-window row W builds*

the displayed witness ledger G from the bounded-family row F, lower-bound ledger L, precision window W, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal E, transport row H, continuation row C, provenance row P, and local naming row N. The induction uses no Dedekind completeness, arbitrary infimum operator, quotient real equality, trichotomy, countable choice, or host completeness.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumWindowInduction`

Proof. Proceed over the displayed finite window W. The empty window has no precision request, so the witness ledger contributes no greatest-lower-approximation entry.

For a nonempty window, read the next precision request together with the already displayed bounded-family and lower-bound rows F and L. The finite comparison data at that request supplies the corresponding G entry, and the remaining window is handled by the induction hypothesis.

The handoff and seal rows are structural at this stage. The row R repacks the constructed window ledger as regular rational readback, and E seals only that readback. The rows H, C, P, N transport, replay, source, and name the same finite construction, so no completeness, quotient-equality, trichotomy, or choice principle enters the induction. □

Theorem 1452.5 (LocatedInfimum window ledger coverage)**.** *For an accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet $I = (F, L, G, W, R, E, H, C, P, N)$, every precision-window entry admitted by the induction of Definition 1452.4 is consumed by exactly one of the lower-bound ledger L, the greatest-lower-approximation ledger G, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, or the* $\mathsf{Real}^{\uparrow}$ *seal row E, with transport, replay, provenance, and local naming read only through H, C, P, N. The coverage remains internal to $F, L, G, W, R, E, H, C, P, N$ and exports no Dedekind completeness, arbitrary infimum operator, quotient real equality, trichotomy, countable choice, host completeness, or unlisted real-order oracle.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumWindowLedgerCoverage`

Proof. Proceed by induction over the displayed finite precision window W. The empty window has no admitted precision entry, so there is no ledger row to consume.

For a successor window, read the new precision entry and apply the lower-bound obligation of Definition 1452.2. If the entry asks for a lower-bound comparison, it is consumed by the displayed lower-bound ledger L and no other analytic ledger receives that entry.

If the entry asks for a greatest-lower-approximation witness, apply Definition 1452.3. The corresponding row is consumed by G at the same precision, and the induction hypothesis covers the remaining window entries.

The remaining Real-facing reads are structural consequences of the same covered entries. Classifier stability, Definition 1452.6, preserves the assignment under H, C, P, N, while ledger exactness, Definition 1452.7, forces every regular-rational handoff and Real seal read through R and E. Thus each admitted precision-window entry is covered by exactly one displayed analytic row before any handoff or seal is exported, and no forbidden endpoint has a carrier coordinate. □

Theorem 1452.6 (LocatedInfimum classifier stability)**.** *Let $I = (F, L, G, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet. Replacing any displayed component along the corresponding* hsame *transport in H preserves lower-bound reads, greatest-lower-approximation reads,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal access, continuation replay, provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$ *classifier. Thus the classifier is stable under componentwise BEDC transport.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumClassifierStability`

Proof. Use the transport ledger H component by component. The family row, lower-bound ledger, greatest-lower-approximation ledger, precision window, regular handoff, real seal, continuation row, provenance row, and name row each have their own displayed hsame transport.

Transport the lower-bound route first. The route still starts at F, reads L inside the same window W, passes through R, and reaches E.

Transport the greatest-lower-approximation route next. The row G is moved with the same precision window and the same handoff and seal rows, so the finite witness relation seen by the classifier is unchanged.

Finally replay C and P after transport. They can only replay and source the transported rows, so the local classifier has no new field through which to distinguish transported copies beyond the displayed hsame data. □

Theorem 1452.7 (LocatedInfimum ledger exactness)**.** *Every Real-facing observation exported by an accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet is obtained from one of the displayed rows* $F, L, G, W, R, E, H, C, P, N$*. In particular, lower-bound, greatest-lower-approximation, handoff, and seal observations are exhausted by the ledger rows, and the packet exports no hidden Dedekind-completeness, arbitrary-infimum, quotient-equality, trichotomy, countable-choice, or host-completeness witness.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumLedgerExactness`

Proof. Begin with a Real-facing observation. The carrier contains only one real seal row, namely E, and the definition makes E reachable only after the bounded-family row, lower-bound ledger, greatest-lower-approximation ledger, precision window, and regular handoff have been displayed.

Inspect the possible observation classes. Lower-bound reads come from L, greatest-lower-approximation reads come from G, window restrictions come from W, and regular rational readback comes from R.

The remaining rows are structural. They transport, replay, source, or name the same finite rows, and therefore cannot create an unlisted completion principle or choice principle. Hence every exported observation is ledger-exact for $F, L, G, W, R, E, H, C, P, N$. □

Theorem 1452.8 (LocatedInfimum lower-cut stability)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, the lower-cut boundary determined by the lower-bound row* L *and the greatest-lower-bound approximation row* G *is preserved under the classifier and ledger transport displayed by* H, C, P, N*. In particular, any transported consumer reads the same finite lower-bound comparisons, the same precision-window witnesses, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal before it treats the packet as an infimum boundary.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumLowerCutStability`

Proof. Start with the lower-bound side. By Definition 1452.2, every lower-bound read factors through F, L, W, R, E and the structural rows, so transport cannot move the lower surface to a row outside the carrier.

Now read the greatest-lower-bound side. By Definition 1452.3, every approximation witness is drawn from G at a precision row of W, and by Definition 1452.7 every Real-facing observation is exhausted by $F, L, G, W, R, E, H, C, P, N$.

Finally apply classifier stability from Definition 1452.6. Componentwise hsame transport preserves the lower-bound route, the greatest-lower-approximation route, the regular-rational handoff, the real seal, continuation replay, provenance, and local naming. The transported packet therefore

exposes the same lower-cut boundary and no Dedekind completeness, arbitrary infimum operator, quotient equality test, trichotomy oracle, countable-choice functional, or host-completeness coordinate. □

Theorem 1452.9 (LocatedInfimum supremum-dual route)**.** *Every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet whose bounded-family row is read through the dual upper-bound presentation first factors through the* $\mathsf{LocatedSupremum}^{\uparrow}$ *sibling of chapter 1427 and the* $\mathsf{BoundedRealFamily}^{\uparrow}$ *row before exposing its lower-bound ledger* L*, greatest-lower-approximation ledger* G*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The resulting route is still the finite carrier route of Definition 1452.1; it does not import a global supremum operator, arbitrary infimum operator, Dedekind completeness, quotient real equality, trichotomy, countable choice, or host completeness.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumSupremumDualRoute`

Proof. Start from the carrier of Definition 1452.1. The bounded-family source is the row F, and the introduction identifies its upper-bound companion as the $\mathsf{LocatedSupremum}^{\uparrow}$ certificate of chapter 1427.

Read the lower-bound side after that dual presentation. By Definition 1452.2, every lower-bound consumer must pass through F, L, W, R, E and the structural rows, so the upper-bound presentation cannot bypass the displayed lower ledger.

Now read the greatest-lower-bound approximation side. Apply Definition 1452.3: the witnesses live in G at precision rows of W, then pass through R and E before the local certificate exports an infimum-facing read.

Thus the dual route has a fixed order: $\mathsf{LocatedSupremum}^{\uparrow}$ and $\mathsf{BoundedRealFamily}^{\uparrow}$ supply the upper-bound and family presentation, while L, G, R, E, H, C, P, N expose the lower-bound, approximation, handoff, seal, transport, replay, provenance, and local naming rows. Since Definition 1452.1 contains no global order-completeness or quotient coordinate, none is added by the dual read. □

Theorem 1452.10 (LocatedInfimum supremum-dual exactness)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet*

$$I = (F, L, G, W, R, E, H, C, P, N),$$

any dual read through $\mathsf{LocatedSupremum}^{\uparrow}$ *and* $\mathsf{BoundedRealFamily}^{\uparrow}$ *preserves exactly the displayed lower-bound ledger* L*, greatest-lower witness ledger* G*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*. The read exports no host infimum, arbitrary supremum operator, quotient real equality, trichotomy, choice principle, or completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumSupremumDualExactness`

Proof. Start from the dual presentation of Definition 1452.9. It may orient the bounded-family row F through $\mathsf{LocatedSupremum}^{\uparrow}$ and $\mathsf{BoundedRealFamily}^{\uparrow}$, but the carrier still exposes the lower-bound surface only through L and the greatest-lower witness surface only through G.

The real-facing handoff is unchanged. The row R remains the only $\mathsf{RegSeqRat}^{\uparrow}$ readback before the terminal $\mathsf{Real}^{\uparrow}$ seal E, so the dual presentation cannot replace the lower-bound evidence by a host infimum or an arbitrary supremum operator.

Finally H, C, P, N preserve the same rows under transport, replay, provenance, and local naming. Since quotient real equality, trichotomy, choice, and completeness are not coordinates of Definition 1452.1, exactness consists precisely of preserving L, G, R, E, H, C, P, N and refusing those stronger principles. □

Theorem 1452.11 (LocatedInfimum bounded-family handoff)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, each bounded-family consumer reads the lower-bound and greatest-lower-approximation surface only through the displayed* $\mathsf{BoundedRealFamily}^{\uparrow}$ *row* F*, the lower-bound ledger* L*, the witness ledger* G*, the precision window* W*, the* $\mathsf{StreamName}^{\uparrow}$ *window discipline, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N*. The route is the lower-bound companion of the* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet and exports no arbitrary infimum operator.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumBoundedFamilyHandoff`

Proof. Project the accepted carrier. The bounded-family source is F, so a downstream bounded-family argument cannot inspect a lower-bound or greatest-lower-approximation claim before it has read the displayed family row.

The lower surface is then split between L and G. The row L supplies the finite lower-bound comparisons, while G supplies the displayed witnesses that block stronger lower-bound candidates inside the same precision window W.

The handoff remains finite. The $\mathsf{StreamName}^{\uparrow}$ window discipline fixes the observations in scope, R repacks them as $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E supplies the only $\mathsf{Real}^{\uparrow}$ seal available to the consumer.

Finally H, C, P, N transport, replay, source, and name the same route. The dual reference to $\mathsf{LocatedSupremum}^{\uparrow}$ identifies the companion orientation, but it adds no ambient completeness field and no arbitrary infimum operator. □

Theorem 1452.12 (LocatedInfimum RegSeqRat seal factorization)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *factors the real-seal export: any read of the* $\mathsf{Real}^{\uparrow}$ *seal* E *must pass through the bounded-family row* F*, the lower-bound ledger* L*, the greatest-lower-approximation ledger* G*, the* $\mathsf{StreamName}^{\uparrow}$ *precision window* W*, and then* R*. No consumer may reach* E *directly from* F*, from* L*, from* G*, or from a* $\mathsf{LocatedSupremum}^{\uparrow}$ *comparison route.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumRegSeqRatSealFactorization`

Proof. Start with a request for the real seal. By Definition 1452.1, E is reached only through the displayed family, lower-bound, greatest-lower-approximation, window, and regular-rational rows.

The window row W is the gate between ledger evidence and regular readback. It fixes which family reads, lower-bound comparisons, and greatest-lower witnesses are in scope before the regular sequence row is assembled.

The row R is therefore the unique $\mathsf{RegSeqRat}^{\uparrow}$ factor. It repacks the displayed $\mathsf{StreamName}^{\uparrow}$ observations as regular rational data, and only then can E be read as the public $\mathsf{Real}^{\uparrow}$ seal.

The structural rows H, C, P, N preserve this order under transport and replay. A $\mathsf{LocatedSupremum}^{\uparrow}$ comparison may provide the companion orientation, but it has no coordinate in I that bypasses W or R. □

Theorem 1452.13 (LocatedInfimum Real export nonescape)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *observation exported by the local certificate is sourced by* $F, L, G, W, R, E, H, C, P, N$ *and remains inside the* $\mathsf{BoundedRealFamily}^{\uparrow}$*,* $\mathsf{LocatedSupremum}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route displayed by the carrier. The export cannot escape to Dedekind completeness, arbitrary infima, quotient real equality, trichotomy, countable choice, host completeness, or an unlisted real-order oracle.*

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumRealExportNonescape`

Proof. Take an exported $\mathsf{Real}^{\uparrow}$ observation. Ledger exactness forces it to be read from the displayed rows $F, L, G, W, R, E, H, C, P, N$, and the carrier has no second real-facing coordinate.

The analytic route is bounded-family first, lower-bound and witness ledgers second, window and regular-rational handoff third, and real seal last. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows keep the export finite at each stage.

The $\mathsf{LocatedSupremum}^{\uparrow}$ reference supplies only the companion lower-versus-upper orientation used by downstream real-order arguments. It does not add a hidden completeness or equality principle to the packet.

Thus every exported real observation remains in the displayed route. Since the forbidden principles have no carrier row and no replay edge, the export cannot escape through them. □

Theorem 1452.14 (LocatedInfimum obligation tail lattice)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, any exported infimum read must expose lower-cut stability, tail-window monotonicity,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *observation, and* $\mathsf{Real}^{\uparrow}$ *sealing in that order before it is treated as a lower-bound lattice value. The accepted route is read through the displayed rows* $F, L, G, W, R, E, H, C, P, N$ *and cannot import a host quotient, trichotomy, ambient completeness, arbitrary infimum operator, or countable-choice selector.*

Proof. Read the displayed rows in order. The bounded-family row F fixes the family under inspection, and the lower-cut row L gives the stable lower-bound comparisons that any infimum read must preserve.

The tail lattice is controlled by G and W. The greatest-lower-approximation witnesses in G are inspected only through the finite tail windows in W, so monotonicity of the accepted tail route is a property of the displayed window ledger rather than an ambient order-completeness principle.

The observation route then passes through R and E. The $\mathsf{StreamName}^{\uparrow}$ windows in W determine the observations, R repacks those observations as $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same row order. Since no host quotient, trichotomy, ambient completeness theorem, arbitrary infimum operator, or choice selector appears in those rows, none is smuggled into the exported infimum read. □

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumObligationTailLattice`

Theorem 1452.15 (LocatedInfimum obligation boundary)**.** *For every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet* $I = (F, L, G, W, R, E, H, C, P, N)$*, the obligation surface is exhausted by the bounded-family row, lower-bound ledger, greatest-lower-approximation ledger,* $\mathsf{StreamName}^{\uparrow}$ *precision window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows. Any consumer that asks for arbitrary infima, Dedekind completeness, quotient real equality, trichotomy, countable choice, host completeness, or a standard bridge asks for data outside the packet.*

Proof. Project the carrier from Definition 1452.1. Its analytic coordinates are exactly F, L, G, W, R, E, and its structural coordinates are exactly H, C, P, N.

Use the lower-bound obligation of Definition 1452.2 and the greatest-lower-bound obligation of Definition 1452.3. These force every lower read through F, L, W, R, E and every greatest-lower approximation through F, L, G, W, R, E.

Now apply window ledger coverage, Definition 1452.5, together with ledger exactness, Definition 1452.7. The finite window entries are consumed by the displayed lower-bound, approximation, handoff, and seal rows, while H, C, P, N only transport, replay, source, and name those rows.

Finally read the exclusions recorded by the chapter's not-claimed boundary as absence of carrier coordinates. There is no row for arbitrary infima, Dedekind completeness, quotient real equality, trichotomy, countable choice, host completeness, or a standard bridge, so a consumer requesting one of those objects has left the accepted packet. □

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumObligationBoundary`

Theorem 1452.16 (LocatedInfimum scoped kernel dependency)**.** *The obligations of an accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet are scoped exactly by the displayed* $\mathsf{BoundedRealFamily}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. No consumer of the chapter obtains Dedekind completeness, an arbitrary infimum operator, quotient real equality, trichotomy, countable choice, host completeness, or a standard bridge.*

Proof. Start from the carrier of Definition 1452.1. Its analytic data are the bounded-family row, the lower-bound ledger, the greatest-lower-bound witness ledger, the finite precision window, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the $\mathsf{Real}^{\uparrow}$ seal; its structural data are the BHist, hsame, Cont, Pkg, and local NameCert rows.

Apply the lower-bound and greatest-lower-bound obligations, Definition 1452.2 and Definition 1452.3. These force lower-bound reads and greatest-lower-approximation reads through the displayed bounded-family, comparison, window, handoff, and seal rows.

Use window induction and ledger coverage from Definition 1452.4 and Definition 1452.5. Each finite precision entry is consumed by the listed ledger rows before any Real-facing observation is exported.

Finally apply classifier stability, ledger exactness, bounded-family handoff, and RegSeqRat seal factorization: Definition 1452.6, Definition 1452.7, Definition 1452.11, and Definition 1452.12. These preserve the same scoped route through $\mathsf{LocatedSupremum}^{\uparrow}$ and $\mathsf{BishopRealComparison}^{\uparrow}$ without adding any forbidden completeness, equality, choice, or bridge coordinate. □

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumScopedKernelDependency`

Theorem 1452.17 (LocatedInfimum public export)**.** *Every accepted* $\mathsf{LocatedInfimum}^{\uparrow}$ *packet exports a public lower-bound and infimum read only through the displayed* $\mathsf{BoundedRealFamily}^{\uparrow}$ *row, lower-bound ledger, greatest-lower-approximation ledger,* $\mathsf{StreamName}^{\uparrow}$ *precision window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The exported public object may be consumed by L10* $\mathsf{Real}^{\uparrow}$ *and completion lower-bound arguments, but it does not supply Dedekind completeness, arbitrary infima, quotient real equality, trichotomy, countable choice, host completeness, or a standard bridge.*

Proof. Start from the carrier of Definition 1452.1. The only analytic rows available to a public lower-bound read are the bounded-family row, lower-bound ledger, greatest-lower-approximation ledger, finite precision window, regular handoff, and real seal.

Apply Definition 1452.2 and Definition 1452.3. Lower-bound reads are forced through F, L, W, R, E, while greatest-lower-approximation reads are forced through F, L, G, W, R, E.

Next use window induction and ledger coverage from Definition 1452.4 and Definition 1452.5. These keep each precision request inside the displayed $\mathsf{StreamName}^{\uparrow}$ window before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ observation is exported.

Finally apply classifier stability, RegSeqRat seal factorization, Real export nonescape, obligation boundary, and scoped-kernel dependency: Definition 1452.6, Definition 1452.12, Definition 1452.13, Definition 1452.15, and Definition 1452.16. They transport, replay, source, and name the same route for L10 $\mathsf{Real}^{\uparrow}$ and completion lower-bound consumers while excluding Dedekind completeness, arbitrary infima, quotient equality, trichotomy, choice, host completeness, and bridge data. □

Lean4 ✓ `BEDC.Derived.LocatedInfimumUp.LocatedInfimumPublicExport`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedInfimum}^{\uparrow}$ packet is generated from a bounded real family row, a lower-bound ledger, a greatest-lower-approximation ledger, a StreamName precision window, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1452.1, Definition 1452.2, Definition 1452.3, Definition 1452.4, Definition 1452.5, Definition 1452.6, Definition 1452.7, Definition 1452.11, Definition 1452.12, and Definition 1452.16 bind the finite lower-bound, greatest-lower-approximation, bounded-family handoff, RegSeqRat seal, and scoped-kernel dependencies over the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedInfimum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, arbitrary infima, quotient real equality, trichotomy, countable choice, host completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, lower-bound readback, greatest-lower-approximation witnessing, classifier stability, ledger exactness, handoff stability, and real-seal factorization for BEDC.Derived.LocatedInfimumUp.

Chapter 1453

A Concrete Naming Certificate for $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ packet isolates the regular Cauchy dense embedding from $\mathsf{RegSeqRat}^{\uparrow}$ into the $\mathsf{Real}^{\uparrow}$ seal. It sits over the dyadic rational rows of chapter 291, the finite stream windows of chapter 126, the regular rational source of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 360. The certificate keeps the dense-embedding route visible as finite NameCert data before any completion consumer treats a regular rational name as a sealed real.

Definition 1453.1 (Regular Cauchy dense embedding carrier)**.** *A* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, S, D, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row whose entries supply the embedded Cauchy name.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite windows through which R is read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic precision requests and rational comparison rows used by the embedding.*
4. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the dense-embedding route after R, S, D have been displayed.*
5. *H records componentwise* hsame *transport for the regular source, stream schedule, dyadic tolerance ledger, Real seal, continuation, provenance, and local naming rows.*
6. *C records the displayed* Cont *replay from each precision request through R, S, D, A, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient selection, host real equality, ambient completeness, countable choice, trichotomy, a detached completion point, or an unbounded stream read.

Theorem 1453.2 (Regular Cauchy dense embedding NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyDenseEr}}$ *surface for Definition 1453.1 consists exactly of carrier exposure for E, classifier exposure through the displayed R, S, D, A route, stream-window stability for S, dyadic tolerance exactness for D,* $\mathsf{Real}^{\uparrow}$ *boundary non-escape through A, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Every accepted dense-embedding request is classified only by the regular rational source row, the scheduled finite windows, and the displayed dyadic tolerance ledger before it reaches the Real seal.*

Proof. Project the carrier tuple $E = (R, S, D, A, H, C, P, N)$. The rows R, S, D, A are the analytic coordinates, and the final row names the same tuple as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}}$ surface.

Read the classifier in dependency order. A precision request first consumes the regular rational source row R, then the finite stream schedule S, then the dyadic tolerance ledger D, and only then reaches the $\mathsf{Real}^{\uparrow}$ seal row A.

The structural rows add no hidden completion principle. They transport, replay, source, and name the displayed route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface and force the Real boundary to stay inside the displayed route. □

Theorem 1453.3 (Regular Cauchy dense embedding RealUp boundary)**.** *For every accepted* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *packet $E = (R, S, D, A, H, C, P, N)$ and every exported* $\mathsf{Real}^{\uparrow}$ *read, the read factors through the finite route*

$$R \longmapsto S \longmapsto D \longmapsto A \longmapsto (H, C, P, N).$$

Thus every exported Real seal read factors through the displayed $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, and no consumer can obtain a detached* $\mathsf{Real}^{\uparrow}$ *value from the packet without reading those rows.*

Proof. Start with Definition 1453.2. It places the regular rational source R, the stream schedule S, and the dyadic tolerance ledger D before the terminal Real seal A.

The Real-facing row is therefore not an independent coordinate. A consumer that omits R, S, or D has not followed the carrier route, so it has no exported read of A from this packet.

Finally read the structural rows. The rows H, C, P, N preserve, replay, source, and name the same route; they do not introduce a second Real seal route. Therefore every exported $\mathsf{Real}^{\uparrow}$ read factors through the displayed regular source, stream schedule, and dyadic ledger. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ schedule, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, real-cauchy-completion: no listed sibling supplies the whole eight-row packet $E = (R, S, D, A, H, C, P, N)$, because each sibling contributes only a dyadic ledger, stream schedule, regular rational source, Real seal, or completion boundary while this chapter binds the dense embedding route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ schedule row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 1453.1, Definition 1453.2, and Definition 1453.3 seed the dense-embedding boundary over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient construction of Real$^{\uparrow}$, host real equality, ambient completeness, countable choice, trichotomy, detached completion points, dense-image surjectivity, or a Lean theorem.

Upgrade path: A formal route can encode the eight-row carrier and prove the NameCert obligation surface and RealUp boundary theorem.

Chapter 1454

A Concrete Naming Certificate for $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ packet records the boundary where a regular rational Cauchy name, an explicit convergence modulus, and finite locatedness evidence are all visible before the $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the finite windows of chapter 126, the regular rational source of chapter 301, the endpoint and radius ledger of chapter 291, the real-name seal of chapter 38, the located enclosure surface of **??**, and the explicit rate rows of chapter 332. The certificate keeps convergence and location as finite rows rather than hiding them inside a later BishopReal or compact-interval consumer.

Definition 1454.1 (Regular located Cauchy real carrier)**.** *A* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, D, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the finite observation windows used by the packet.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational entries are read only on the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger recording the rational centers, dyadic radii, and interval comparisons attached to those windows.*
4. *M is the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *rate row assigning each requested tolerance a finite threshold over the displayed windows.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row recording the finite interval-location evidence attached to the same source, endpoint, radius, and threshold rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular-source, dyadic-ledger, modulus, and located-enclosure rows have been displayed.*
7. *H records componentwise* hsame *transport for the stream windows, regular rational source, dyadic endpoint and radius ledger, convergence-modulus row, located enclosure, Real seal, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from each precision and location request through $S, Q, D, M, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for Dedekind construction of $\mathsf{Real}^{\uparrow}$*, quotient equality, trichotomy, open-cover compactness, countable choice, completed metric-space points, or an unbounded stream read.*

Theorem 1454.2 (Regular located Cauchy real NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ *surface for Definition 1454.1 consists exactly of carrier exposure for R, finite-window exposure through S, regular rational source exposure through Q, endpoint and radius exposure through D, convergence-rate exposure through M, located-enclosure exposure through L, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Each accepted request is classified only by the displayed StreamName window, the RegSeqRat source row, the dyadic endpoint and radius ledger, the convergence threshold, and the located enclosure row.*

```
Lean4 ✓  BEDC.Derived.RegularLocatedCauchyRealUp.TasteGate.RegularLocatedCauchyRealTasteGate
single carrier alignment
```

Proof. Project the carrier tuple $R = (S, Q, D, M, L, E, H, C, P, N)$. Each analytic coordinate is a displayed finite row, and the final coordinate names the same tuple as the local $\mathsf{NameCert}_{\mathsf{RegularLocatedCauchyReal}^{\uparrow}}$ surface.

Read the rows in their dependency order. A request first consumes the stream window S and regular rational source Q, then the dyadic endpoint and radius ledger D, then the convergence threshold row M, and only then the located enclosure L and terminal Real seal E.

The structural rows add no hidden analytic object. They transport, replay, source, and name the displayed route, while the excluded objects have no coordinate in the carrier. Hence the listed rows exhaust the seed obligation surface. □

Theorem 1454.3 (Regular located Cauchy real RealUp seal route)**.** *For every accepted* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *packet $R = (S, Q, D, M, L, E, H, C, P, N)$ and every displayed precision request, the Real-facing read factors through the finite route*

$$S \longmapsto Q \longmapsto D \longmapsto M \longmapsto L \longmapsto E \longmapsto (H, C, P, N).$$

No consumer can read the terminal $\mathsf{Real}^{\uparrow}$ *seal from R without also reading the same StreamName window family, RegSeqRat source row, DyadicRatCore endpoint and radius ledger, ModulusOfConvergence rate row, and LocatedReal enclosure row.*

Proof. Start with Definition 1454.2. It places the finite window row S before the regular source row Q and places the dyadic ledger D before the convergence modulus M and located enclosure L.

The Real seal is therefore not an independent coordinate. The row E is reached after S, Q, D, M, L are displayed, so a consumer omitting any of those rows has not followed the carrier route.

Finally read the structural rows. The rows H, C, P, N preserve, replay, source, and name the same route; they do not add a second route to $\mathsf{Real}^{\uparrow}$. Thus the terminal seal factors through the displayed finite certificate path. □

Theorem 1454.4 (Regular located Cauchy real modulus-location factorization)**.** *For every accepted* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *packet $R = (S, Q, D, M, L, E, H, C, P, N)$, every locatedness read factors through the common finite data*

$$S, \quad Q, \quad D, \quad M$$

before it reaches L, and every convergence-modulus read uses the same S, Q, D rows before it reaches M. Hence the modulus and locatedness surfaces share the StreamName windows, RegSeqRat source, and DyadicRatCore endpoint and radius ledger rather than presenting two unrelated certificates.

Proof. Use the carrier definition. The row M is a rate row over the windows, source, endpoints, and radii already named by S, Q, D; the row L is a located enclosure over the same displayed source and dyadic ledger, with the rate row fixing the relevant tail threshold.

Now compare the two reads. The convergence side stops at M after S, Q, D, while the locatedness side continues through M to L on the same finite rows.

The componentwise hsame and Cont rows preserve that shared route, and the Pkg and local NameCert rows name it. Thus any accepted consumer sees one factored certificate surface, not a separated modulus certificate and a separated location certificate. □

Theorem 1454.5 (Regular located Cauchy real scoped dependency route)**.** *Every accepted* RegularLocatedCauchyR *read factors through exactly the displayed* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *readback row,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* ModulusOfConvergence$^{\uparrow}$ *location row,* LocatedReal$^{\uparrow}$ *locatedness row, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$ *row. No quotient stream equality, selected representative, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness coordinate is in scope for such a read.*

Proof. Use Definition 1454.2. It exposes the carrier as the tuple $R = (S, Q, D, M, L, E, H, C, P, N)$ and makes the stream windows, regular rational source, dyadic ledger, convergence row, locatedness row, and Real seal explicit.

Apply Definition 1454.3 to the endpoint-facing side. The terminal Real$^{\uparrow}$ read is downstream of the same S, Q, D, M, L rows, so it cannot be reached by a separate representative-selection path.

Apply Definition 1454.4 to the locatedness side. The modulus and locatedness reads share S, Q, D, M before reaching L, and the structural rows only transport, replay, source, and name that route.

The excluded principles have no coordinate in the displayed carrier. Hence the scoped route is exactly the finite dependency route named above. □

Falsifiable prediction: An accepted RegularLocatedCauchyReal$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ seal or a locatedness enclosure without the displayed StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ source, DyadicRatCore$^{\uparrow}$ endpoint and radius ledger, and ModulusOfConvergence$^{\uparrow}$ rate row refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, locatedreal, modulusofconvergence: no listed sibling supplies the whole ten-row packet $R = (S, Q, D, M, L, E, H, C, P, N)$, because each sibling contributes only a window, regular-source, dyadic-ledger, Real-seal, located-enclosure, or rate surface while this chapter binds all six analytic faces with transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$.

Constructive story: A RegularLocatedCauchyReal$^{\uparrow}$ packet is generated from finite StreamName$^{\uparrow}$ windows, a RegSeqRat$^{\uparrow}$ source row, a DyadicRatCore$^{\uparrow}$ endpoint and radius ledger, a ModulusOfConvergence$^{\uparrow}$ rate row, a LocatedReal$^{\uparrow}$ enclosure row, a terminal Real$^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1454.1, Definition 1454.2, Definition 1454.3, Definition 1454.4, and Definition 1454.5 bind the regular located Cauchy real-name route over StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, ModulusOfConvergence$^{\uparrow}$, LocatedReal$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RegularLocatedCauchyReal}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularLocatedCauchyRealUp.TasteGate.RegularLocatedCauchyRealTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not claim Dedekind construction of $\mathrm{Real}^{\uparrow}$, quotient equality, trichotomy, open-cover compactness, countable choice, completeness, or completed metric-space points.

Upgrade path: Public closure requires a downstream public-object surface that reuses the scoped dependency route without adding quotient equality, selected representatives, countable choice, or ambient completeness.

Chapter 1455

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ packet names the finite boundary at which a regular Cauchy rational source, locatedness evidence, and the displayed limit route are all visible before a $\mathsf{Real}^{\uparrow}$ completion-facing seal is read. It sits after the regular located Cauchy surface of chapter 1454, the located-limit route of chapter 1420, the located-real source of **??**, the real-name seal of chapter 38, the regular rational source of chapter 301, the stream-window source of chapter 126, and the dyadic arithmetic core of chapter 291. The chapter records only the NameCert boundary for located completion data; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 1455.1 (Bishop located completion boundary carrier)**.** *A* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, D, R, L, E, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the finite observations of the rational Cauchy source.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row read only through the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, endpoint, and radius ledger attached to those windows.*
4. *R is the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *row that binds the regular source, dyadic ledger, locatedness evidence, and terminal real-name seal as one visible route.*
5. *L is the* $\mathsf{LocatedLimit}^{\uparrow}$ *route by which the same displayed source is read as a located limit candidate.*
6. *E is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row used at the location boundary before any terminal real-name consumer is allowed to inspect a seal.*
7. *T is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, D, R, L, E have been displayed.*
8. *H records componentwise* hsame *transport for the stream, regular-source, dyadic, regular-located, located-limit, located-real, Real-seal, continuation, provenance, and local naming rows.*
9. *C records the displayed* Cont *replay for every completion-facing request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a proof of $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, real trichotomy, countable choice, open-cover compactness, or a host-selected limit.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.TasteGate.BishopLocatedCompletionBoundaryTasteGa`
`single carrier alignment`

Theorem 1455.2 (Regular Cauchy extraction at the located boundary)**.** *Every accepted* BishopLocatedCompletionB *packet $B = (S, Q, D, R, L, E, T, H, C, P, N)$ exposes its regular Cauchy source before any located completion read. The extracted source surface is exactly the finite route*

$$S, \quad Q, \quad D, \quad R$$

with the visible transport, replay, provenance, and naming rows H, C, P, N. No consumer can obtain a completion-facing $\mathsf{Real}^{\uparrow}$ *seal from B without first reading the displayed StreamName windows, RegSeqRat source, DyadicRatCore tolerance ledger, and RegularLocatedCauchyReal route.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryRegularCauchyExtr`

Proof. Project the carrier of Definition 1455.1. The only regular Cauchy data present are the stream windows S, regular rational source Q, dyadic tolerance ledger D, and regular located Cauchy row R.

Now read any completion-facing request. The row T is stored after the located-limit and located-real rows, and those rows are themselves attached to S, Q, D, R. Thus the request must first pass through the extracted regular Cauchy source route.

The transport, replay, provenance, and local naming rows preserve this same route. Since the carrier contains no independent completion coordinate, the displayed extraction exhausts the regular Cauchy surface available to the consumer. □

Theorem 1455.3 (Located limit route at the completion boundary)**.** *For every accepted* BishopLocatedCompletionB *packet $B = (S, Q, D, R, L, E, T, H, C, P, N)$, every located completion read factors through*

$$S \longmapsto Q \longmapsto D \longmapsto R \longmapsto L \longmapsto E \longmapsto T.$$

The route uses the located-limit row L and located-real enclosure row E as visible finite data before the $\mathsf{Real}^{\uparrow}$ *seal row T is read, and it exports no separate host limit, quotient equality row, or choice-selected subsequence.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryLocatedLimitRoute`

Proof. Begin with the extraction theorem Definition 1455.2. It places the stream, regular rational, dyadic tolerance, and regular located Cauchy rows before any completion-facing seal.

The carrier definition then places L and E between that extracted source and the row T. Hence a located completion read must first display the located-limit route and the located-real enclosure row attached to the same finite source.

Finally apply the structural rows. Componentwise transport and replay keep the consumer on S, Q, D, R, L, E, T, while provenance and local naming record the same packet. No coordinate outside this route can supply a host limit or a quotient equality proof. □

Theorem 1455.4 (Bishop located completion boundary NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{BishopLocatedCom}}$ *surface for Definition 1455.1 consists exactly of carrier exposure for B, regular Cauchy extraction through S, Q, D, R, located-limit routing through L, located-real enclosure through E, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through T, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. These obligations bind the finite located completion boundary without adding* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, trichotomy, countable choice, open-cover compactness, or a host-selected limit.*

Proof. Carrier exposure is projection from the displayed tuple $B = (S, Q, D, R, L, E, T, H, C, P, N)$. The analytic rows are exactly the stream, regular rational, dyadic, regular located Cauchy, located-limit, located-real, and Real-seal rows named in the carrier.

Regular Cauchy extraction is supplied by Definition 1455.2. Located routing is supplied by Definition 1455.3. Together they put every completion-facing read on one finite route through S, Q, D, R, L, E, T.

The remaining rows are structural. The hsame transports and Cont replay preserve the same finite route, while Pkg provenance and the local NameCert row name the packet. Since no excluded principle appears as a carrier coordinate, the listed rows exhaust the seed obligation surface. □

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryNameCertObligatio`

Theorem 1455.5 (Bishop located completion boundary located Real-seal handoff)**.** *Every accepted* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *packet reaches its* $\mathsf{Real}^{\uparrow}$ *seal T only through the displayed route*

$$S \longmapsto Q \longmapsto D \longmapsto R \longmapsto L \longmapsto E \longmapsto T.$$

The handoff first reads the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows D, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and readback row Q, and the located-real comparison row E. It preserves the same route through componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}$ *row, and it supplies no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem or quotient equality of real names.*

Proof. Unpack the carrier of Definition 1455.1. Its analytic rows are exactly S, Q, D, R, L, E, T, with T placed after the dyadic tolerance ledger, stream windows, regular rational source, regular located Cauchy route, located-limit route, and located-real enclosure row.

Apply regular Cauchy extraction, Definition 1455.2. It forces every completion-facing read to expose S, Q, D, R before a terminal seal can be inspected, so the dyadic tolerance rows, finite windows, and regularity/readback row are already displayed.

Next apply located-limit routing, Definition 1455.3. The row E is the located-real comparison row immediately before T, so the Real-seal handoff cannot bypass locatedness data or replace it by a host limit.

Finally use the obligation surface. Componentwise hsame transport and Cont replay keep the same route, while Pkg provenance and the local NameCert row name it as one packet. Since the carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness or quotient equality, the handoff exports only the displayed seal row T over the finite located boundary. □

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryLocatedRealSealHa`

Theorem 1455.6 (Bishop located completion boundary regular route)**.** *Any accepted* BishopLocatedCompletionBou *read that reaches the terminal seal T must first expose the rows*

$$S, Q, D, R, L, E$$

in that order. The RegularLocatedCauchyReal$^{\uparrow}$ *row R, the* LocatedLimit$^{\uparrow}$ *row L, and the* LocatedReal$^{\uparrow}$ *row E are therefore mandatory boundary rows, not replaceable by a host completion, quotient equality, selected limit, ambient* Real$^{\uparrow}$ *completeness theorem, trichotomy, countable-choice schedule, or open-cover compactness principle.*

Proof. Apply regular Cauchy extraction, Definition 1455.2. It exposes S, Q, D, R before any terminal seal can be read, so the regular located Cauchy row cannot be replaced by a host completion row.

Next apply the located-limit route, Definition 1455.3. This places L after the regular route and before any located Real enclosure can be consumed.

Now use the NameCert obligations of Definition 1455.4 and the located Real-seal handoff of Definition 1455.5. These keep E immediately before T on the same finite route and preserve it under transport, replay, provenance, and local naming.

Since the carrier has no coordinate for host completion, quotient equality, selected limits, completeness, trichotomy, countable choice, or compactness, every terminal read reaches T only through the displayed regular route. □

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryRegularRoute`

Theorem 1455.7 (Bishop located completion boundary window exhaustion)**.** *Every accepted* BishopLocatedCompletionBoundary$^{\uparrow}$ *packet $B = (S, Q, D, R, L, E, T, H, C, P, N)$ exposes all completion-facing finite windows through S and all tolerance and readback data through D, Q, R, L, E before T is consumed. No completion-facing consumer can add a second* StreamName$^{\uparrow}$ *schedule, quotient equality, selected limit, countable choice, trichotomy, open-cover compactness, or* Real$^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryWindowExhaustion`

Proof. Project the carrier of Definition 1455.1. The only finite-window row is S, and the only tolerance, regular readback, regular-located, located-limit, and located-real rows are D, Q, R, L, E.

Apply regular Cauchy extraction, Definition 1455.2. It forces every completion-facing read to display S, Q, D, R before the terminal seal row can be inspected.

Apply located-limit routing, Definition 1455.3, and then the located Real-seal handoff, Definition 1455.5. These two results place L, E after the extracted source and before T, so the consumer cannot bypass locatedness or replace the route by a host-selected limit.

Finally use H, C, P, N. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming preserve the same route S, Q, D, R, L, E, T. Since none of those structural rows contains a second stream schedule, quotient equality, choice principle, trichotomy, compactness principle, or completeness coordinate, the displayed finite windows are exhausted before T is consumed. □

Theorem 1455.8 (Bishop located completion boundary inductive route)**.** *Every accepted terminal read of a* BishopLocatedCompletionBoundary$^{\uparrow}$ *carrier*

$$B = (S, Q, D, R, L, E, T, H, C, P, N)$$

is generated by induction on the displayed route: first the StreamName$^{\uparrow}$ *windows S, then the* RegSeqRat$^{\uparrow}$ *source Q, the* DyadicRatCore$^{\uparrow}$ *ledger D, the* RegularLocatedCauchyReal$^{\uparrow}$ *row R, the* LocatedLimit$^{\uparrow}$ *row L, the* LocatedReal$^{\uparrow}$ *row E, and the terminal* Real$^{\uparrow}$ *seal T; only after those analytic rows are present may the structural rows H, C, P, N transport, replay, source, and name the*

same read. The induction exposes no quotient equality, selected limit, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryInductiveRoute`

Proof. The base of the route is the carrier projection in Definition 1455.1. It lists S as the only stream-window row and places the regular source, dyadic ledger, located rows, terminal seal, and structural rows in the displayed order.

For the first analytic steps, apply regular Cauchy extraction and the located-limit route. The extraction theorem forces S, Q, D, R to be visible before any completion-facing seal, while the located-limit theorem extends the route through L and E before T can be read.

The Real-seal handoff and the regular-route theorem provide the terminal step. They identify T as downstream of S, Q, D, R, L, E and rule out replacement by a host completion, quotient equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

It remains to account for the structural rows. The NameCert obligations and window-exhaustion theorem preserve the same finite route under H, C, P, N and show that no second window schedule or hidden completion coordinate is available. Thus induction over the displayed rows generates every accepted terminal read. □

Theorem 1455.9 (Bishop located completion boundary obligation carrier)**.** *Every accepted* BishopLocatedCompleti *packet is admitted to the obligation surface only as the eleven-row carrier*

$$B = (S, Q, D, R, L, E, T, H, C, P, N)$$

of Definition 1455.1. The admitted rows are exactly the $\mathsf{StreamName}^{\uparrow}$ *window family,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *route,* $\mathsf{LocatedLimit}^{\uparrow}$ *route,* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. No additional carrier coordinate may supply* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, trichotomy, countable choice, open-cover compactness, or a host-selected limit.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryObligationCarrier`

Proof. Project the carrier definition. It lists precisely $S, Q, D, R, L, E, T, H, C, P, N$ and places all completion-facing analytic data before the terminal seal.

Apply regular Cauchy extraction, located-limit routing, the located Real-seal handoff, and window exhaustion. These results show that the stream windows, regular rational source, dyadic ledger, located route, enclosure row, and seal are consumed as one visible boundary.

The structural rows only preserve that boundary. Since provenance and local naming record the same packet and no excluded principle is present as a row, the obligation carrier is exactly the displayed eleven-row surface. □

Theorem 1455.10 (Bishop located completion boundary scope)**.** *Every scoped route exposed by a* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *packet passes through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularLocatedCauchyReal}^{\uparrow} \longmapsto \mathsf{LocatedLimit}^{\uparrow} \longmapsto \mathsf{Located}$$

The route is read through the displayed rows S, Q, D, R, L, E, T *and the structural rows* H, C, P, N *before any completion-facing consumer can cite it. It does not expose a host completion, quotient real equality, trichotomy, countable-choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryScope`

Proof. Start with Definition 1455.6. It forces the rows S, Q, D, R, L, E to appear before the terminal seal T can be read.

Use window exhaustion and the NameCert obligations. They identify S as the only finite-window schedule, Q as the regular rational source, D as the dyadic ledger, and R, L, E as the located completion boundary before the $\mathsf{Real}^{\uparrow}$ seal.

Finally apply componentwise transport and displayed replay. The scope is preserved under H and C, while P and N name the same packet, so no outside completion, equality, order, choice, compactness, or completeness datum enters the route. □

Theorem 1455.11 (Bishop located completion boundary scoped route)**.** *Every scoped completion-boundary read from an accepted* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *packet must pass through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularLocatedCauchyReal}^{\uparrow} \longmapsto \mathsf{LocatedLimit}^{\uparrow} \longmapsto \mathsf{Located}$$

together with the componentwise hsame *row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}$ *row. The route is exactly the eleven-row carrier*

$$B = (S, Q, D, R, L, E, T, H, C, P, N)$$

and admits no host completion, quotient equality, selected limit, trichotomy, countable-choice schedule, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-checking row.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryScopedRoute`

Proof. Project the obligation carrier of Definition 1455.9. It admits only $S, Q, D, R, L, E, T, H, C, P, N$, so every scoped read begins inside the eleven displayed rows.

Apply regular Cauchy extraction and the located-limit route. These place the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$ rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Next use the located Real-seal handoff and the scope theorem. The seal row T is reachable only through the displayed located route, and the scope theorem records the same ordered kernel path as a boundary for completion-facing consumers.

Finally use the ledger theorem. The structural rows H, C, P, N transport, replay, source, and name the same route, while the ledger has no coordinate for host completion, quotient equality, selected limits, order principles, choice schedules, compactness, ambient completeness, or bridge checking. Thus every scoped read is confined to the stated kernel route. □

Theorem 1455.12 (Bishop located completion boundary ledger)**.** *The ledger of an accepted* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *packet is non-escaping: every completion-facing read is accounted for by the stream windows* S*, regular source* Q*, dyadic ledger* D*, regular located route* R*, located-limit route* L*, located-real enclosure* E*, terminal seal* T*, and structural rows* H, C, P, N*. The ledger has no row for* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, real trichotomy, countable choice, open-cover compactness, a host-selected limit, or bridge checking.*

Proof. Read the obligation carrier theorem. It admits only the displayed eleven rows and refuses every extra completion, equality, order, choice, compactness, selected-limit, or bridge coordinate.

Apply the scope theorem. Any consumer that reaches the terminal seal has already passed through the StreamName, RegSeqRat, DyadicRatCore, RegularLocatedCauchyReal, LocatedLimit, LocatedReal, and Real route.

The remaining rows are bookkeeping inside the same packet: H transports components, C replays displayed continuations, P records provenance, and N gives local naming. Therefore the ledger cannot export any refused endpoint. □

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryLedger`

Theorem 1455.13 (Bishop located completion boundary public interface)**.** *Every public read of an accepted* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularLocatedCauchyReal}^{\uparrow},\ \mathsf{LocatedLimit}^{\uparrow},\ \mathsf{LocatedReal}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the componentwise hsame *row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}$ *row. The public interface exposes only those finite rows and explicitly refuses* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, selected limits, countable choice, real trichotomy, open-cover compactness, and bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionBoundaryUp.BishopLocatedCompletionBoundaryPublicInterface`

Proof. Project the scoped route theorem and the ledger theorem. They identify every completion-facing read with the eleven-row carrier $S, Q, D, R, L, E, T, H, C, P, N$ and place the StreamName, RegSeqRat, DyadicRatCore, RegularLocatedCauchyReal, LocatedLimit, LocatedReal, and Real rows before any terminal seal can be consumed.

The structural rows do not enlarge the interface. Componentwise hsame transport and displayed Cont replay preserve the same route, while Pkg provenance and the local NameCert row name the packet.

Since the ledger has no coordinate for completeness, quotient equality, selected limits, choice, trichotomy, compactness, or bridge checking, the public read is exactly the displayed finite boundary and none of the refused endpoints is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ regular rational source, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ route, a $\mathsf{LocatedLimit}^{\uparrow}$ route, a $\mathsf{LocatedReal}^{\uparrow}$ enclosure row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1455.1, Definition 1455.2, Definition 1455.3, Definition 1455.9, Definition 1455.10, and Definition 1455.12 bind the finite located completion boundary over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient equality, trichotomy, countable choice, open-cover compactness, a host-selected limit, or bridge checking.

Upgrade path: The next paper obligation is bridged closure for the same located completion boundary, with formal verification still pending.

Chapter 1456

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Bourbaki language of universal mappings, the $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ packet isolates the L10 boundary where regular Cauchy names are admitted into a completion seal and uniformly continuous finite-window consumers are extended back out through a displayed universal row. It reads the naming spine of chapter 126, chapter 301, and chapter 38; the Cauchy filter boundary of chapter 333; the extension surface of chapter 397; the functorial completion row of chapter 426; and the generic adjunction packet of chapter 107. The chapter records only the finite NameCert boundary for this universal mapping shape.

Definition 1456.1 (CauchyCompletionAdjunction carrier)**.** *A* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, C, D, E, U, H, P, N)$$

with the following displayed rows:

1. *X is the source metric and finite Cauchy-observation row, including the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source, and the* $\mathsf{CauchyFilter}^{\uparrow}$ *stability ledger used before completion is read.*
2. *C is the dense embedding row from the source observations into the completion seal, with the target seal read through the* $\mathsf{Real}^{\uparrow}$ *boundary.*
3. *D is the completion-seal row exposed through* $\mathsf{CompletionFunctor}^{\uparrow}$ *and the adjacent real-name route, not as a quotient completion or host-level complete metric space.*
4. *E is the extension candidate row for uniformly continuous finite-window consumers, read through the displayed* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension surface.*
5. *U is the uniqueness ledger asserting that two extension rows agreeing on the dense rows named by C are compared only by the carried* hsame *route.*
6. *H records the componentwise* hsame *and* Cont *replay for X, C, D, E, U, including classifier transport through the source, dense embedding, completion seal, extension candidate, and uniqueness rows.*
7. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{Adjunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}$ *row.*

The carrier contains no quotient completion, no selected limit, no open-cover compactness field, no countable-choice selector, and no equality of host functions.

Theorem 1456.2 (CauchyCompletionAdjunction NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjuncti}}$ *surface for Definition 1456.1 consists exactly of six finite obligation rows:*

1. *carrier exposure: an accepted object exposes only the packet $A = (X, C, D, E, U, H, P, N)$;*
2. *classifier transport: accepted comparison is componentwise through the displayed* hsame *and* Cont *routes recorded in H;*
3. *dense embedding preservation: the source observations in X enter the completion seal only through the dense row C;*
4. *extension row: every uniformly continuous finite-window consumer reaches a completion-facing read through the extension coordinate E and the sibling* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *surface;*
5. *uniqueness by dense row: two extension candidates are compared only through the uniqueness ledger U after agreement on the dense rows named by C;*
6. *Real/RegSeqRat handoff non-escape: every public read remains inside the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, *and* $\mathsf{Adjunction}^{\uparrow}$ *rows displayed by P.*

Thus the packet binds the universal mapping boundary from regular Cauchy source data to the completion seal without adding quotient completions, selected limits, open-cover compactness, countable choice, or equality of host functions.

Proof. Carrier exposure is projection from the tuple in Definition 1456.1. Its source row is X, its dense embedding is C, its completion seal is D, its extension candidate is E, its uniqueness ledger is U, and its structural rows are H, P, N.

Classifier transport is componentwise. The row H is the only place where hsame comparisons and Cont replays are stored, so every accepted comparison of carriers must replay the same source, dense, seal, extension, and uniqueness coordinates.

Dense embedding preservation and extension are then read in order. The source observations carried by X reach the completion seal through C and D, while a uniformly continuous finite-window consumer reaches a completion-facing row only through E and the universal-property surface named in P.

Finally, uniqueness and non-escape are ledger statements over the displayed packet. The row U compares extensions only after agreement on the dense rows, and the provenance row P lists no quotient completion, selected limit, open-cover compactness, countable-choice selector, or host function equality coordinate. The six rows therefore exhaust the local NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionUp.CauchyCompletionAdjunctionTasteGate single carrier alignment`

Theorem 1456.3 (CauchyCompletionAdjunction extension handoff)**.** *For an accepted* $\mathsf{CauchyCompletionAdjunctio}$ *carrier*

$$A = (X, C, D, E, U, H, P, N)$$

as in Definition 1456.1, every uniformly continuous finite-window consumer reaches the completion-facing extension candidate only through the source row X, the dense embedding row C, the completion-seal row D, the extension row E, the uniqueness ledger U, the componentwise hsame *comparisons in H, the displayed* Cont *routes in H, the* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}$ *row N.*

Equivalently, the consumer route factors through the $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension surface, the* $\mathsf{CompletionFunctor}^{\uparrow}$ *completion row, and the* $\mathsf{Adjunction}^{\uparrow}$ *universal-mapping ledger named by P; no separate completion object or host function equality is exported.*

Proof. Project the accepted packet through Definition 1456.1. The source observations are stored in X, the dense embedding data in C, the completion seal in D, the extension candidate in E, and the dense-row uniqueness comparison in U.

Now route a uniformly continuous finite-window consumer. The $\mathsf{CompletionUniversalProperty}^{\uparrow}$ row named by P supplies the only extension-facing surface, while $\mathsf{CompletionFunctor}^{\uparrow}$ supplies the completion row and $\mathsf{Adjunction}^{\uparrow}$ records the universal-mapping ledger that compares source-side and completion-side reads.

It remains to bind transport and naming. The row H carries exactly the componentwise hsame comparisons and displayed Cont replays for X, C, D, E, U; the rows P and N then package the same route as local NameCert data. Since no other coordinate occurs in A, every extension read uses this displayed handoff. □

Theorem 1456.4 (CauchyCompletionAdjunction Real seal non-escape)**.** *For an accepted* CauchyCompletionAdjunc *carrier $A = (X, C, D, E, U, H, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing read of the completion seal factors through Definition 1456.3, the carrier inventory of Definition 1456.1, and the terminal route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{Adjunction}^{\uparrow}$ *named by P.*

Consequently the packet cannot export quotient completions, selected representatives, host equality of functions or real names, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the handoff theorem. It already forces every completion-facing consumer through the source, dense embedding, completion-seal, extension, uniqueness, transport, provenance, and local naming rows of the same carrier.

The Real-facing part of this route is terminal in the displayed naming spine. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ regular rational rows feed the $\mathsf{Real}^{\uparrow}$ seal, the $\mathsf{CauchyFilter}^{\uparrow}$ row records the Cauchy boundary before completion is read, and the $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, and $\mathsf{Adjunction}^{\uparrow}$ rows keep the extension and universal-mapping reads inside P.

Finally inspect the carrier inventory. It has no coordinate for a quotient completion, selected representative, host equality row, countable-choice selector, open-cover compactness principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. Therefore no accepted Real-facing read can export any of those objects. □

Theorem 1456.5 (CauchyCompletionAdjunction unit universal route)**.** *For an accepted* CauchyCompletionAdjunct *carrier $A = (X, C, D, E, U, H, P, N)$, every unit-admitted source read that is later consumed by the completion universal property factors through the displayed source row X, dense embedding row C, completion seal row D, extension candidate row E, uniqueness ledger U,* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff named by P, componentwise transport and replay row H, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}$ *row N of Definition 1456.1.*

Consequently a unit-side consumer has no route to quotient completions, selected limits, countable choice, host function equality, ambient completeness, or a completion object outside the displayed universal-property packet.

Proof. Begin with the NameCert obligation theorem. Theorem 1456.2 exposes the source row, dense embedding row, extension row, uniqueness ledger, and Real/RegSeqRat handoff as finite obligation rows of the same carrier.

Next apply the extension handoff theorem. Theorem 1456.3 forces a uniformly continuous finite-window consumer through X, C, D, E, U, H, P, N before the $\mathsf{CompletionUniversalProperty}^{\uparrow}$ surface can be read.

It remains to keep the unit read inside the Real seal. Theorem 1456.4 routes the terminal read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, and $\mathsf{Adjunction}^{\uparrow}$ as named by P.

The carrier has no coordinate for quotient completions, selected limits, countable choice, host function equality, or ambient completeness. Thus every unit-admitted source read consumed by the universal property uses exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ packet is generated from a source metric and Cauchy-observation row, a dense embedding into the completion seal, a completion-seal row, an extension candidate for uniformly continuous finite-window consumers, a dense-row uniqueness ledger, componentwise hsame and Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1456.1, Definition 1456.2, Definition 1456.3, Definition 1456.4, and Definition 1456.5 bind the L10 completion universal mapping boundary over the StreamName, RegSeqRat, CauchyFilter, Real, CompletionUniversalProperty, CompletionFunctor, and Adjunction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyCompletionAdjunctionUp.CauchyCompletionAdjunctionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, open-cover compactness, countable choice, host function equality, ambient completeness, or bridge checking.

Upgrade path: A formal route can encode the eight-row carrier and prove the classifier transport, dense embedding preservation, extension row, uniqueness ledger, and Real/RegSeqRat handoff non-escape clauses.

Chapter 1457

A Concrete Naming Certificate for $\mathsf{DedekindCauchySeal}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen constructive metamathematics, the $\mathsf{DedekindCauchySeal}^{\uparrow}$ packet records the finite bidirectional seal between located Dedekind-cut observations and regular Cauchy-name observations. It sits after the located-cut surface of chapter 436, the comparison boundary of chapter 1424, the regular rational source of chapter 301, the stream-window source of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only the finite seal surface by which the two representations can be read against one another; it does not construct $\mathsf{Real}^{\uparrow}$ from cuts and does not assert ambient completeness.

Definition 1457.1 (DedekindCauchySeal carrier)**.** *A* $\mathsf{DedekindCauchySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, R, W, T, F, G, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *row, carrying the located lower and upper rational observations that may be read by a seal consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, carrying the regular rational Cauchy name used for finite tolerance requests.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family through which the row R is exposed.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radii, rational comparison budgets, and finite refinement rows used on both sides of the seal.*
5. *F is the forward seal row that regularizes a displayed located-cut observation into a finite regular Cauchy window against T.*
6. *G is the reverse seal row that reads a displayed regular Cauchy window back as located cut data against T.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *boundary row reached only after D, R, W, T, F, G have been displayed.*
8. *H records componentwise* hsame *transport for the cut, regular-name, stream-window, dyadic, forward, reverse, Real-boundary, continuation, provenance, and local naming rows.*
9. *C records the displayed* Cont *replay route for rereading the same finite rows under an accepted consumer motion.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindCauchySeal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCauchySeal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a Dedekind construction of $\mathsf{Real}^{\uparrow}$*, ambient completeness, quotient stream equality, real trichotomy, countable choice, selected representatives, or host real equality.*

Definition 1457.2 (DedekindCauchySeal classifier)**.** *The* $\mathsf{DedekindCauchySeal}^{\uparrow}$ *classifier accepts exactly those finite reads of the carrier $S = (D, R, W, T, F, G, E, H, C, P, N)$ that proceed through one of the following displayed routes:*

1. *a cut-to-name route reading D, then the tolerance ledger T, then the forward seal row F, then the stream windows W and regular row R;*
2. *a name-to-cut route reading R through W, then the tolerance ledger T, then the reverse seal row G, then the located-cut row D;*
3. *a Real-boundary route reading D, R, W, T, F, G before the terminal row E.*

All accepted classifier answers are transported by H, replayed by C, and recorded by P, N. The classifier rejects any request whose answer would require data outside the displayed BHist *packet.*

Theorem 1457.3 (DedekindCauchySeal NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DedekindCauchySeal}^{\uparrow}}$ *surface for Definition 1457.1 consists exactly of carrier exposure for S, classifier exposure through Definition 1457.2, forward regularization through F, reverse cut readback through G, dyadic budget control through T, stream-window exposure through W, terminal* $\mathsf{Real}^{\uparrow}$ *boundary exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Carrier exposure is projection from the displayed BHist packet $S = (D, R, W, T, F, G, E, H, C, P, N)$. The analytic rows are exactly the $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, forward seal, reverse seal, and $\mathsf{Real}^{\uparrow}$ boundary rows named in the carrier.

The classifier is finite by Definition 1457.2. Each accepted read first chooses one of the displayed routes, and each route is exhausted by the stored rows D, R, W, T, F, G, E.

The structural rows then close the local naming surface. The hsame row transports each displayed component, the Cont row replays the same finite route, and the Pkg and NameCert rows record the public packet. Since no excluded principle appears as a coordinate of S, the listed obligations exhaust the seal surface. □

Theorem 1457.4 (Forward regularization through the seal)**.** *Every accepted* $\mathsf{DedekindCauchySeal}^{\uparrow}$ *packet $S = (D, R, W, T, F, G, E, H, C, P, N)$ exposes a finite forward route from located Dedekind-cut observations to regular Cauchy-name observations:*

$$D \longmapsto T \longmapsto F \longmapsto W \longmapsto R.$$

The route uses only displayed BHist, hsame, Cont, Pkg, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows, and it exports no quotient stream equality, selected representative, countable-choice schedule, or host real equality.*

Proof. Project the carrier of Definition 1457.1. A cut-facing request can read located rational observations only from D and can refine them only through the dyadic ledger T.

The forward seal row F is stored after those two rows. It therefore has access only to the displayed cut observations and tolerance budget, and its output is forced into the stream-window family W before the regular rational row R is read.

Transport and replay do not add analytic data. The rows H and C preserve the same route, while P and N record that the route is the public $\mathsf{DedekindCauchySeal}^{\uparrow}$ certificate. The missing quotient, representative, choice, and host-equality data cannot be recovered from the packet. □

Theorem 1457.5 (Reverse cut readback through the seal)**.** *Every accepted* $\mathsf{DedekindCauchySeal}^{\uparrow}$ *packet* $S = (D, R, W, T, F, G, E, H, C, P, N)$ *exposes a finite reverse route from regular Cauchy-name observations to located Dedekind-cut observations:*

$$R \longmapsto W \longmapsto T \longmapsto G \longmapsto D.$$

The route reads the regular rational row only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows, uses the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger as its comparison budget, and returns only the displayed* $\mathsf{DedekindCut}^{\uparrow}$ *row.*

Proof. Start with a name-facing request. The carrier makes the regular rational row R visible only through the finite windows W, so the request cannot inspect an infinite stream or a quotient stream object.

The dyadic ledger T supplies the finite comparison radii for the reverse read. The reverse seal row G then reads the same windowed name data back into the located-cut row D and has no coordinate from which to choose representatives or host equalities.

Finally apply the structural rows. Componentwise hsame transport and displayed Cont replay keep the consumer on R, W, T, G, D, while Pkg provenance and the local $\mathsf{NameCert}$ row bind the route to the same public packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCauchySeal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DedekindCauchySeal}^{\uparrow}$ packet is generated from a $\mathsf{DedekindCut}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, forward and reverse seal rows, a terminal $\mathsf{Real}^{\uparrow}$ boundary row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCauchySeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1457.1, Definition 1457.2, Definition 1457.3, Definition 1457.4, and Definition 1457.5 seed the finite bidirectional seal between located Dedekind cuts and regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCauchySeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a Dedekind construction of $\mathsf{Real}^{\uparrow}$, ambient completeness, quotient stream equality, trichotomy, countable choice, selected representatives, host real equality, or bridge checking.

Upgrade path: Scoped closure requires a checked finite carrier theorem binding the displayed cut row, regular Cauchy row, stream windows, dyadic tolerance ledger, forward seal, reverse seal, and Real-boundary row.

Chapter 1458

SeparableMetricUp NameCert Construction

Following Bishop and Bridges, and the classical Bourbaki presentation of separable metric spaces, the $\mathsf{SeparableMetric}^{\uparrow}$ packet records countable density as displayed finite name data rather than as a choice principle over arbitrary balls. It sits beside the distance certificate of chapter 57, the stream-window surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance rows of chapter 291, the real-facing seal of chapter 38, and the completion consumer of chapter 128. Its role is to expose the L10 countable-name surface consumed by completion and compact-metric chapters without importing selected dense points, quotient completion equality, open-cover compactness, or ambient Real completeness.

Definition 1458.1 (Separable metric carrier surface)**.** *A* $\mathsf{SeparableMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, A, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *carrier and its public distance-witness surface.*
2. *A is the countable dense-name family, represented as a unary index schedule together with the finite rows naming the indexed dense candidates inspected by a request.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for requests of the form "indices below a displayed bound and tolerance below a displayed scale".*
4. *T is the tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows and through the* $\mathsf{Real}^{\uparrow}$ *comparison rows already exposed by the metric distance certificate.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that packages the inspected finite windows and tolerances as regular rational observation data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row reached only after the metric carrier, dense-name window, tolerance ledger, and regular rational handoff have been displayed.*
7. *H stores componentwise* hsame *transport for the metric carrier, dense-name family, finite windows, tolerance ledger, regular handoff, real seal, continuation routes, package provenance, and local naming.*

8. *C stores the displayed* Cont *replay route from finite requests to dense-name windows, from windows to dyadic or real tolerance rows, and from those rows to the regular rational and distance-seal consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}$ *row.*

The carrier compares accepted separable-metric packets only through these rows. It contains no coordinate for choosing a dense point from every inhabited ball, no quotient equality for completion points, no open-cover compactness certificate, and no ambient Real-completeness principle.

Lean4 (definition) `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier`

Theorem 1458.2 (Separable metric NameCert obligation surface)**.** *For every accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *carrier $S = (M, A, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}$ *surface consists exactly of the following six obligations:*

1. *carrier admission for the displayed metric carrier, countable dense-name family, finite request windows, tolerance ledger, regular rational handoff, real distance seal, transport, continuation, package provenance, and local naming rows;*
2. *metric compatibility, stating that all distance reads are inherited from the displayed* $\mathsf{Met}^{\uparrow}$ *certificate M and not from a second host metric;*
3. *countable-name visibility, stating that dense candidates are inspected only through the indexed family A and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W;*
4. *tolerance discipline, stating that every density request carries its tolerance through T as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *row before any handoff is consumed;*
5. *regular handoff, stating that R is obtained only from the displayed dense-name windows and tolerance rows;*
6. *nonescape, stating that E, H, C, P, N preserve the same finite dependency surface and export no choice-selected dense point, quotient completion equality, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1458.1. The tuple already lists the only coordinates available to the classifier: the metric certificate, the countable dense-name family, finite windows, tolerance ledger, regular rational handoff, real seal, transport, continuation, provenance, and local naming rows.

Metric compatibility is inherited from M. A separability consumer may ask for distances from indexed candidates to a target row only through the public $\mathsf{Met}^{\uparrow}$ distance-witness surface, so no second host metric or ambient equality test is introduced.

Countable-name visibility is the role of A and W. The indexed family supplies candidates only when a finite request window is displayed, and hsame transport in H preserves that same windowed family under classifier replacement.

The tolerance and regular faces are downstream from the same window. The ledger T records the dyadic or real tolerance rows read by the request, and R repacks exactly those finite observations as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

Finally, continuation and package provenance do not add analytic strength. The rows E, C, P, N only replay and name the displayed dependency chain, so selected dense points, quotient completion equality, open-cover compactness, and ambient Real completeness are outside the packet. □

Theorem 1458.3 (Separable metric dense-window stability)**.** *Let $S = (M, A, W, T, R, E, H, C, P, N)$ be an accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *carrier. If two finite density requests read the same indexed dense-name window through W and the same tolerance row through T, then every* $\mathsf{CompleteMetric}^{\uparrow}$*-facing or* $\mathsf{Real}^{\uparrow}$*-distance consumer receives the same displayed route*

$$A, W, T \longmapsto R \longmapsto E$$

transported by H and replayed by C under the package provenance P and the local naming row N. The stable window route contains no choice-selected dense point, quotient completion equality, open-cover compactness, or ambient Real completeness.

Proof. Begin with the common finite window. Since both requests expose the same indexed rows of A through W, a consumer sees the same finite dense-name candidates before any distance or completion-facing read is available.

Read the tolerance ledger next. The shared row T fixes the $\mathsf{DyadicRatCore}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ tolerance data attached to that finite window, and the metric compatibility obligation keeps those reads inside the displayed $\mathsf{Met}^{\uparrow}$ distance surface.

The regular handoff is therefore fixed. The row R is built from the same dense-name window and tolerance ledger in both requests, and the terminal seal E is downstream from R.

Transport and replay preserve the route rather than choosing a new candidate. The rows H and C carry the same finite dependency chain to $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers, while P and N record package provenance and local naming. Thus the stable route exports no selected dense point, quotient completion equality, open-cover compactness, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier dense window stability`

Theorem 1458.4 (Separable metric countable-density carrier admission)**.** *Let $S = (M, A, W, T, R, E, H, C, P, N)$ be the displayed* $\mathsf{SeparableMetric}^{\uparrow}$ *packet of Definition 1458.1. Carrier admission for S is exactly the finite* BHist *admission of the* $\mathsf{Met}^{\uparrow}$ *distance row M, the indexed dense-name row A, the* $\mathsf{StreamName}^{\uparrow}$ *request window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, the* $\mathsf{Real}^{\uparrow}$ *distance seal E, componentwise* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}$ *naming row N. No admitted coordinate chooses a dense point from an arbitrary inhabited ball, identifies quotient-completion classes, or imports ambient real completeness.*

Proof. Project the carrier tuple from Definition 1458.1. Each admissible component is already named by a finite BHist row: the metric distance row, dense-name index row, window row, tolerance row, regular handoff, real seal, transport, replay, provenance, and local certificate.

The admission test is therefore componentwise. The $\mathsf{Met}^{\uparrow}$ row supplies the only distance surface, A and W supply the only countable-density observations, T supplies the dyadic or real tolerance data, and R and E are downstream of those displayed rows.

Transport, continuation, provenance, and naming do not add a selection principle. The rows H, C, P, N only preserve and package the finite route, so a selected dense point, quotient-completion equality, or ambient Real-completeness witness would require a coordinate not present in S. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier countable density admission`

Theorem 1458.5 (Separable metric dense-window metric compatibility)**.** *For an accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *packet* $S = (M, A, W, T, R, E, H, C, P, N)$*, every finite density request that reads an indexed candidate through the* $\mathsf{StreamName}^{\uparrow}$ *window* W *must compute its distance comparison through the displayed* $\mathsf{Met}^{\uparrow}$ *distance row* M *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row* T *before it can be handed to* R *or sealed by* E*. The* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and* $\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}$ *row* N *preserve that route for later consumers.*

Proof. Start with a finite density request. The only candidate names visible to the request are the indexed rows of A admitted through the window W, so the request has no direct access to a host dense set or to a separately chosen point.

Metric compatibility is forced by the available distance row. Before a candidate can be compared with a tolerance, the request must pass through M and then through the displayed tolerance ledger T, read as $\mathsf{DyadicRatCore}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ data.

Only after those rows are displayed can the request reach the regular handoff R and the real-facing seal E. The rows H, C, P, N transport, replay, package, and locally name the same route, so the metric comparison cannot be replaced by a second metric or by quotient-completion equality. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier dense window metric compatibility`

Theorem 1458.6 (Separable metric finite-density public export)**.** *Every accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *packet* $S = (M, A, W, T, R, E, H, C, P, N)$ *exports a public finite-density route through the displayed* $\mathsf{Met}^{\uparrow}$ *distance row* M*, indexed dense-name row* A*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. Any* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{PolishSpace}^{\uparrow}$ *consumer may cite only that finite route through* H, C, P, N *and may not cite selected dense points, quotient completion equality, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier finite density public export`

Proof. Project the carrier of Definition 1458.1. The rows $M, A, W, T, R, E, H, C, P, N$ are the only public data exposed by the accepted separable-metric packet.

Apply Definition 1458.2 and Definition 1458.4. These identify the metric distance row, indexed dense-name row, finite window, tolerance ledger, regular handoff, real seal, transport, replay, provenance, and local naming as the admitted finite-density surface.

Use Definition 1458.3 together with Definition 1458.5. The dense-name window is stable under matching finite requests, and every distance comparison passes through M and T before R or E can be consumed.

Thus a completion or Polish-space consumer receives only the displayed finite route through $M, A, W, T, R, E, H, C, P, N$. Since selected dense points, quotient completion equality, open-cover compactness, and ambient Real completeness are absent from the carrier and from the obligation surface, they cannot be exported. □

Theorem 1458.7 (Separable metric regular-handoff obligation)**.** *Let* $S = (M, A, W, T, R, E, H, C, P, N)$ *be an accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *packet. The* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *is obligated to use only the dense-name observations displayed by* A *and* W*, the* $\mathsf{Met}^{\uparrow}$ *distance row* M*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance rows in* T*. Its export to* E, H, C, P, N *is a finite NameCert-local route, not a selected Cauchy subsequence, quotient-completion equality, open-cover compactness witness, or ambient completeness proof.*

Proof. Read the handoff row in the order exposed by the carrier. The dense observations come from A through the finite $\mathsf{StreamName}^{\uparrow}$ window W, and all metric comparisons have already been routed through M and the tolerance ledger T.

The row R repackages exactly those observations as regular rational data. It has no independent coordinate for selecting a subsequence, choosing a dense point, or asserting equality of completion representatives.

The export surface is then the displayed seal, transport, replay, provenance, and local naming rows E, H, C, P, N. These rows make the handoff available to $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and later NameCert consumers while preserving the finite packet boundary and refusing open-cover compactness or ambient Real-completeness data. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier regular handoff obligation`

Theorem 1458.8 (Separable metric tolerance-net replay)**.** *Let $S = (M, A, W, T, R, E, H, C, P, N)$ be an accepted $\mathsf{SeparableMetric}^{\uparrow}$ carrier. Every finite density request whose indexed window is displayed by W and whose tolerance is displayed by T replays through the finite route*

$$M, A, W, T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

In particular, the request reads $\mathsf{Met}^{\uparrow}$ distance data, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff data, and $\mathsf{Real}^{\uparrow}$ distance seals only through the carrier rows of S.

Proof. Project the carrier tuple of Definition 1458.1. The rows M, A, W, T display the metric certificate, indexed dense-name family, finite request window, and tolerance ledger before any downstream handoff is available.

Use the NameCert obligation surface of Definition 1458.2. Metric compatibility keeps all distance reads inside M, countable-name visibility keeps the request inside A and W, and tolerance discipline keeps the precision data inside T.

The regular and real-facing rows are then forced. The row R repacks the same inspected window and tolerance ledger as $\mathsf{RegSeqRat}^{\uparrow}$ data, while E supplies the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name that route. No step introduces a selected dense point or a second tolerance witness outside the displayed packet. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier tolerance net replay`

Theorem 1458.9 (Separable metric completion-consumer handoff)**.** *Let $S = (M, A, W, T, R, E, H, C, P, N)$ be an accepted $\mathsf{SeparableMetric}^{\uparrow}$ carrier. Every $\mathsf{CompleteMetric}^{\uparrow}$ consumer of S receives only indexed dense-name windows, tolerance rows, regular handoff rows, and real distance seals displayed by*

$$A, W, T, R, E, H, C, P, N.$$

The consumer receives no selected dense point, quotient completion equality, open-cover compactness certificate, or ambient Real completeness principle.

Proof. Project the tuple $S = (M, A, W, T, R, E, H, C, P, N)$. The only separability coordinates available to a completion consumer are the indexed dense-name family, the finite windows, the tolerance ledger, the regular handoff, and the real distance seal.

Apply Definition 1458.2. Countable-name visibility prevents the consumer from replacing A by a selector over balls, and tolerance discipline prevents the consumer from reading a precision datum before T is displayed.

Apply Definition 1458.3. If two completion-facing reads share the same finite window and tolerance row, then the route through R and E is the same under H and C.

Thus the $\mathsf{CompleteMetric}^{\uparrow}$ handoff is windowed and indexed. Since selected points, quotient completion equality, open-cover compactness, and ambient Real completeness are absent from the carrier, they cannot be exported to the consumer. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier completion consumer handoff`

Theorem 1458.10 (Separable metric RegSeqRat density handoff)**.** *Let $S = (M, A, W, T, R, E, H, C, P, N)$ be an accepted $\mathsf{SeparableMetric}^{\uparrow}$ carrier. Every $\mathsf{RegSeqRat}^{\uparrow}$ handoff exported by the separability packet is downstream of the same countable dense-name family A and the same tolerance ledger T that feed the request window W. The handoff cannot be formed from a different dense family, an untracked tolerance, or an ambient real-completeness argument.*

Proof. Project A, W, T, R from Definition 1458.1. The row W displays the finite indices inspected from A, and the row T displays the dyadic or real tolerance data attached to that same request.

Use Definition 1458.2. The regular handoff obligation states that R is obtained only from the displayed dense-name windows and tolerance rows, while the nonescape obligation keeps all downstream rows on the same finite dependency surface.

The transport and replay rows preserve the source of the handoff. The rows H and C cannot replace the dense-name family or tolerance ledger; they only carry the displayed A, W, T, R, E route to later consumers.

Therefore every $\mathsf{RegSeqRat}^{\uparrow}$ density handoff is downstream of the same countable dense-name family and tolerance ledger. A different dense family, untracked tolerance, or ambient real-completeness argument has no carrier coordinate from which it could be read. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier regseqrat density handoff`

Theorem 1458.11 (Separable metric obligation frontier)**.** *For every accepted $\mathsf{SeparableMetric}^{\uparrow}$ countable-density read from a carrier*

$$S = (M, A, W, T, R, E, H, C, P, N),$$

the obligation frontier is exhausted by the displayed $\mathsf{Met}^{\uparrow}$ row M, the indexed dense-name row A, the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{DyadicRatCore}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ tolerance ledger T, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, the $\mathsf{Real}^{\uparrow}$ seal E, and the structural rows H, C, P, N. It refuses selected dense points, quotient completion equality, open-cover compactness, and ambient Real completeness.

Proof. Project the carrier S. The finite rows $M, A, W, T, R, E, H, C, P, N$ are the only rows exposed to a countable-density consumer.

Apply Definition 1458.2. The obligation surface already separates metric compatibility, countable-name visibility, tolerance discipline, regular handoff, and nonescape.

Use Definition 1458.3 and Definition 1458.8. The dense-name family is read only through the finite window W, while the tolerance ledger T is replayed through the metric row M before the regular handoff and Real seal are visible.

Apply Definition 1458.9 and Definition 1458.10. Completion and regular-rational consumers receive the same displayed route through $A, W, T, R, E, H, C, P, N$ rather than a selected dense point, quotient-completion equality, open-cover compactness certificate, or ambient Real-completeness principle.

Thus the frontier for L10 countable density is exactly the displayed Metric, indexed dense-name, StreamName, tolerance, RegSeqRat, Real, transport, replay, provenance, and local naming route. □

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricCarrier obligation frontier`

Theorem 1458.12 (Separable metric public density export)**.** *Every public countable-density read exported by an accepted* $\mathsf{SeparableMetric}^{\uparrow}$ *packet*

$$S = (M, A, W, T, R, E, H, C, P, N)$$

factors through the displayed Metric row M*, indexed dense-name row* A*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance ledger* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*. The export gives downstream consumers a public countable-density route, but it exports no selected dense point, quotient completion equality, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SeparableMetricUp.SeparableMetricPublicDensityExport`

Proof. Start with Definition 1458.2. It identifies the accepted carrier surface as exactly $M, A, W, T, R, E, H, C, P, N$, so a public read has no coordinate outside the displayed Metric, dense-name, window, tolerance, handoff, seal, transport, replay, provenance, and local naming rows.

Use Definition 1458.3. The indexed dense-name family can be consumed only through the finite $\mathsf{StreamName}^{\uparrow}$ windows W, and matching windows with matching tolerance rows replay the same route to completion-facing and Real-facing consumers.

Apply Definition 1458.9 and Definition 1458.10. Completion consumers receive only the displayed $A, W, T, R, E, H, C, P, N$ route, while the $\mathsf{RegSeqRat}^{\uparrow}$ density handoff remains downstream of the same dense-name family and tolerance ledger.

Thus the public export is the componentwise projection of the closed countable-density frontier in Definition 1458.11. Since selected dense points, quotient completion equality, open-cover compactness, and ambient Real completeness are absent from that frontier, they cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeparableMetric}^{\uparrow}$ packet is generated from a displayed MetricUp carrier, an indexed dense-name family, finite StreamName request windows, DyadicRatCore or RealUp tolerance rows, a RegSeqRat handoff, a RealUp distance seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1458.1, Definition 1458.2, Definition 1458.3, Definition 1458.8, Definition 1458.9, Definition 1458.10, Definition 1458.11, and Definition 1458.12 bind the displayed $\mathsf{Met}^{\uparrow}$ carrier, indexed dense-name family, $\mathsf{StreamName}^{\uparrow}$ request windows, $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows as the public L10 countable-density export scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparableMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected dense points, quotient completion equality, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: The paper-side countable-density, completion-handoff, RegSeqRat, StreamName, and Real-seal treatment is mature; the remaining work is independent Lean verification and bridge checking.

Chapter 1459

A Concrete Naming Certificate for $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ packet records the finite comparison test that lets a displayed majorant regular Cauchy source certify a target stream before the Real-facing route is consumed. It sits over the regular rational readback of chapter 301, the dyadic tolerance surface of chapter 291, the finite window discipline of chapter 126, and the explicit completion-modulus route of chapter 1469. The packet is a constructive comparison principle for the open L10 real-completion path; it is not a quotient of streams, a selected limit, a host real comparison, countable choice, or an ambient completeness theorem.

Definition 1459.1 (Regular Cauchy comparison test carrier)**.** *A* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the majorant regular Cauchy source row whose finite tolerances bound the comparison.*
2. *B is the target stream or source row to be tested against the displayed majorant.*
3. *M is the explicit modulus row assigning each displayed tolerance request a finite threshold.*
4. *D is the dyadic domination and tolerance ledger, including the dyadic scale, comparison endpoint rows, and displayed domination certificates at each inspected tolerance.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family on which the source row, target row, modulus row, and dyadic ledger are all read.*
6. *H records componentwise* hsame *transport for the majorant source, target source, modulus, dyadic ledger, stream windows, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each tolerance request through M to the window in W, then through D to the target Cauchy row and the downstream regular rational handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows above. It has no coordinate for quotient stream equality, a hidden infinite tail selector, a host real value, a selected completed limit, countable choice, or ambient completeness.

Theorem 1459.2 (Regular Cauchy comparison test NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *carrier* $T = (A, B, M, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed nine-row packet;*
2. *majorant-source exposure for* A *and target-source exposure for* B *on the same finite window family* W*;*
3. *modulus threshold readback, stating that every inspected tolerance request reads its threshold from the displayed row* M*;*
4. *dyadic domination compatibility, stating that each displayed tolerance in* D *bounds the target row* B *by the majorant row* A *on the requested window;*
5. *transport stability, stating that the threshold and domination rows are preserved by the componentwise* hsame *rows in* H*;*
6. *continuation replay, provenance, and local naming for exactly the route recorded by* C, P, N*.*

These obligations are finite-row obligations over the sibling surfaces of chapter 301, chapter 291, chapter 126, and chapter 1469.

Proof. Project the carrier $T = (A, B, M, D, W, H, C, P, N)$. The projection exposes exactly the majorant source, target source, modulus row, dyadic ledger, finite window family, transport rows, continuation replay, provenance, and local naming row.

For a displayed tolerance request, read the modulus row M to obtain the finite threshold. The same threshold selects the relevant finite windows in W, so both A and B are inspected only through displayed data.

The dyadic ledger then supplies domination at that tolerance. Since the comparison row in D is attached to the same threshold and window, the target Cauchy row is obtained by finite-row projection from the displayed domination certificate rather than from a hidden tail argument.

Finally apply the structural rows. The hsame rows in H preserve the threshold and domination data componentwise, while C, P, N only replay, source, and name the displayed route. No omitted coordinate remains from which an additional NameCert obligation could be read. □

Theorem 1459.3 (Regular Cauchy comparison test RegSeqRat handoff)**.** *Every accepted* $\mathsf{RegularCauchyComparison}$ *carrier hands the target row to* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the finite route*

$$A, B, M \longmapsto W, D \longmapsto H, C, P, N.$$

In particular, a completion-facing consumer can reach the downstream Real-completion path only after reading the displayed $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, and the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *threshold route that bind the target row to the majorant source.*

Proof. Begin with an accepted carrier and fix a displayed tolerance request. The modulus row M provides the threshold, and the finite windows in W are the only windows admitted for that request.

Read the target row through the domination ledger. At each displayed tolerance, the dyadic row D compares the target source B with the majorant regular Cauchy source A on the selected window. This supplies the target Cauchy row as a finite consequence of displayed domination.

Transport preserves the same route. The componentwise hsame rows in H carry the modulus threshold, the selected window, and the dyadic domination certificate together, so a transported carrier cannot change the RegSeqRat handoff while keeping the same comparison test.

The remaining rows are downstream bookkeeping. The Cont replay in C, the provenance row P, and the local naming row N send the displayed comparison route to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and then to Real-completion consumers only through the named sibling rows. Thus $\mathsf{Real}^{\uparrow}$ is reachable along this path only after the StreamName, DyadicRatCore, and BishopCompletionModulus data have been displayed. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ consumer that reads a target $\mathsf{RegSeqRat}^{\uparrow}$ handoff without the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ domination rows, and $\mathsf{BishopCompletionModulus}^{\uparrow}$ threshold route refutes the carrier surface.

Independence witness: regseqrat, dyadicratcore, streamname, bishopcompletionmodulus: no listed sibling supplies the whole nine-row packet $T = (A, B, M, D, W, H, C, P, N)$, because each sibling contributes only a regular-source, dyadic-ledger, window, or modulus face while this chapter binds all four analytic faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonTest}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ packet is generated from a majorant regular Cauchy source, a target stream or source row, an explicit modulus row, dyadic domination and tolerance rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1459.1, Definition 1459.2, and Definition 1459.3 seed the finite comparison-test boundary over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{BishopCompletionModulus}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completed limits, host real comparison, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and prove the finite NameCert obligation surface and RegSeqRat handoff theorem.

Chapter 1460

A Concrete Naming Certificate for $\mathsf{DarbouxIntegral}^{\uparrow}$

Following Darboux and Bishop-style constructive analysis, the $\mathsf{DarbouxIntegral}^{\uparrow}$ packet presents integration through finite partitions, upper and lower dyadic step approximants, refinement ledgers, and $\mathsf{Real}^{\uparrow}$ handoff rows. It does not use measure completion, quotient functions, choice-selected tags, or ambient real completeness. The packet consumes the step-function surface of chapter 428, the finite partition ledger of chapter 201, and the public integration consumer boundary of chapter 123.

1460.1 Carrier packet

Definition 1460.1 (Darboux integral finite carrier)**.** *A* $\mathsf{DarbouxIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (P, U, L, SU, SL, G, R, H, C, N)$$

with the following displayed coordinates.

1. *P is a* $\mathsf{Partition}^{\uparrow}$ *row, read as the finite dyadic partition ledger on the integration interval.*
2. *U and L are upper and lower* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximants over a displayed common refinement of P.*
3. *SU and SL are the finite upper and lower Darboux sum rows computed from the cells and carried values of U and L.*
4. *G is the upper-lower gap and modulus ledger comparing SU and SL at the displayed precision.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff surface produced from the gap ledger.*
6. *H records the componentwise* hsame *transports of partition, step-function, sum, gap, and handoff rows.*
7. *C records the finite* Cont *routes that replay refinements, cell reads, finite sums, and handoff rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}$ *row and* Pkg *provenance surface attached to the packet.*

The carrier classifies two accepted packets only by these displayed rows. In particular, it has no coordinate for a selected tag, a host Riemann sum oracle, a quotient function space, a measure completion, or an ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralTasteGate single carrier alignment`

1460.2 Certificate obligations

Theorem 1460.2 (Darboux integral NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}$ *surface consists of the following five rows.*

1. *Carrier habitation for the finite packet* $D = (P, U, L, SU, SL, G, R, H, C, N)$ *of Definition 1460.1.*
2. *Refinement-stable classification of the displayed partition and upper/lower step-function approximants by common dyadic refinement.*
3. *Finite Darboux-sum exactness for* SU *and* SL *over the carried cells and values of* U *and* L*.*
4. *Gap-modulus exposure through* G*, with* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff through* R*.*
5. *Ledger exclusion of selected tags, quotient functions, measure completion, and ambient real-completeness data.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket namecert obligations`

Proof. Carrier habitation is the finite packaging operation of Definition 1460.1. It displays the partition row, upper and lower step-function approximants, the two finite sum rows, the gap row, the handoff surface, transport routes, continuation routes, and the local name.

Refinement stability is read through the common-refinement machinery already exposed by the $\mathsf{DyadicStepFunction}^{\uparrow}$ packet. The same displayed refinement transports P, U, L, SU, and SL, so classification remains a finite ledger statement rather than a comparison of host functions.

The Darboux-sum and gap rows are then inspected as ledger coordinates. Since every downstream handoff to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{Integral}^{\uparrow}$ passes through G and R, the excluded host data have no carrier coordinate or continuation route. □

Theorem 1460.3 (Darboux partition carrier admission)**.** *Let* $D = (P, U, L, SU, SL, G, R, H, C, N)$ *be an accepted* $\mathsf{DarbouxIntegral}^{\uparrow}$ *carrier. A Darboux-integral read is admitted only when the finite* BHist *partition row* P *displays a closed interval source, a bounded real function row, the cell ledger used by the upper and lower step approximants, and the local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}$ *carrier row. The admission route is transported by the componentwise* hsame *rows in* H*, replayed by* C*, and sourced by the* Pkg *provenance inside* N*; it does not introduce Lebesgue measure, choice-selected partitions, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket partition carrier admission`

Proof. Project the carrier of Definition 1460.1. The partition row P is the only source row from which cells can be read, and the upper and lower approximants U, L are downstream of that same finite ledger.

Use the carrier obligations of Definition 1460.2. Carrier habitation and refinement-stable classification require the closed interval source, bounded real function row, partition cells, and local NameCert row to be read before any sum or gap row is consumed.

The structural rows preserve admission. Componentwise hsame transport in H, finite Cont replay in C, and provenance through N keep the read on the displayed packet, so no measure object, selected partition family, or ambient completeness coordinate is admitted. □

Theorem 1460.4 (Darboux upper-lower sum ledger)**.** *For every accepted* DarbouxIntegral$^{\uparrow}$ *carrier* $D = (P, U, L, SU, SL, G, R, H, C, N)$*, the upper and lower sum rows* SU *and* SL *are computed from the displayed partition row* P*, the upper and lower* DyadicStepFunction$^{\uparrow}$ *rows* U, L*, and the carried Rational and Real sum ledgers. Componentwise* hsame *transport preserves the same cell list and value reads before the gap row* G *is inspected, and* Cont *replay exposes no host Riemann-sum oracle or choice-selected tags.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket upper lower sum ledger`

Proof. Start from Definition 1460.3. The accepted read has already displayed the closed interval source, bounded function row, and finite partition ledger.

Compute the two sums by replaying C on the displayed cells. The upper row SU reads only the carried upper step values in U, and the lower row SL reads only the carried lower step values in L.

Transport through H keeps the partition cells and value rows aligned. The Rational and Real sum ledgers are therefore the finite ledgers already present in the packet, not a host summation oracle, a tagged Riemann sum, or a measure-completion read. □

Theorem 1460.5 (Darboux Cauchy gap nonescape)**.** *Let* $D = (P, U, L, SU, SL, G, R, H, C, N)$ *be an accepted* DarbouxIntegral$^{\uparrow}$ *carrier. Any Cauchy-facing or* Integral$^{\uparrow}$ *consumer that reads the gap row* G *must first consume the partition admission route and the upper/lower sum ledger, then read the* RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *handoff row* R *through the same* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *surface. The gap cannot escape to Lebesgue measure, choice-selected partitions, quotient function equality, host completeness, or a detached real sequence.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket cauchy gap nonescape`

Proof. Apply Definition 1460.3. It places the gap consumer behind the displayed partition row, closed interval source, bounded function row, and local carrier admission.

Apply Definition 1460.4. The gap row G compares the finite upper and lower sum ledgers SU, SL after their common partition and value rows have been read.

Finally inspect the handoff row R. The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ reads are downstream of G and are transported, replayed, sourced, and named by H, C, N. Since the carrier has no coordinate for measure completion, selected partitions, quotient function equality, ambient completeness, or detached real sequences, the Cauchy gap cannot leave the displayed packet. □

Theorem 1460.6 (Darboux upper-lower sum gap)**.** *For accepted* DarbouxIntegral$^{\uparrow}$ *packets with displayed common refinement of their partition rows, transporting to that refinement preserves the visible upper/lower sum gap recorded by* G*. The transported packet computes* SU *and* SL *only from displayed refined cells and carried values, and the gap comparison never introduces a selected tag, quotient function, or measure completion.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket upper lower sum gap`

Proof. Project the accepted packet through Definition 1460.1. The common refinement gives the finite cell list on which both approximants are read, and the rows U and L already carry the value data used on those cells.

Compute SU and SL by replaying the finite sum routes in C. Refinement only changes the displayed parent-cell ledger: each refined cell points back to a named cell, and its upper or lower value is the transported value already present in U or L.

The gap row G compares the two transported finite sums after this replay. Because the packet has no coordinate for tags, quotient functions, or measure completion, preserving the displayed gap cannot add any of those objects. □

Theorem 1460.7 (Darboux real handoff)**.** *An accepted* $\mathsf{DarbouxIntegral}^{\uparrow}$ *packet hands an* $\mathsf{Integral}^{\uparrow}$ *consumer only the displayed partition row, upper and lower* $\mathsf{DyadicStepFunction}^{\uparrow}$ *rows, upper and lower sum rows, gap ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *row, componentwise* hsame *transports, finite* Cont *routes,* Pkg *provenance, and local* NameCert *rows. No host integrand equality, measure completion, selected tag family, or ambient real-completeness witness is exported.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket integral consumer surface`

Proof. Start from Definition 1460.2. The public surface is exhausted by the carrier, refinement classifier, Darboux-sum exactness, gap-modulus row, and ledger exclusion rows.

Use Definition 1460.6 to route every consumer-visible approximation through common refinement before the gap is read. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows in R are therefore handoff rows for the displayed gap ledger, not independent real-completeness data.

Finally pass the same rows to the $\mathsf{Integral}^{\uparrow}$ consumer boundary. That boundary receives the partition, step-function, sum, gap, handoff, hsame, Cont, Pkg, and NameCert rows named by the packet, and no additional analytic surface is available. □

Theorem 1460.8 (Darboux public gap seal)**.** *Every* $\mathsf{Integral}^{\uparrow}$*-facing public read of an accepted* $\mathsf{DarbouxIntegral}^{\uparrow}$ *packet factors through the displayed* $\mathsf{Partition}^{\uparrow}$ *row,* $\mathsf{DyadicStepFunction}^{\uparrow}$ *upper and lower approximants, finite Darboux sum rows, gap ledger* G*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}$ *rows before any public integral seal is visible. No public gap seal exports a host integrand equality, measure completion, selected tag family, quotient function equality, detached real sequence, or ambient real-completeness witness.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket public gap seal`

Proof. Start from Definition 1460.2. The carrier and obligation rows expose the partition, upper and lower step approximants, finite sums, gap ledger, handoff rows, structural transport, replay, provenance, and local naming as the whole public surface.

Use Definition 1460.5. It places every Cauchy-facing or $\mathsf{Integral}^{\uparrow}$-facing read of G behind partition admission, the upper/lower sum ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$/$\mathsf{Real}^{\uparrow}$ handoff through the same finite packet.

Finally apply Definition 1460.7. The consumer boundary receives exactly the partition, step-function, sum, gap, regular-rational, real-name, transport, replay, provenance, and local NameCert rows. Since the carrier has no coordinate for the excluded host data, the public integral seal can only be reached through those rows. □

Definition 1460.9 (Darboux integral mature boundary)**.** *The mature* $\mathsf{DarbouxIntegral}^{\uparrow}$ *boundary is the finite packet formed by the carrier of Definition 1460.1, the NameCert obligations of Definition 1460.2, the partition admission route, upper/lower sum ledger, Cauchy gap nonescape row, refinement-stable gap theorem, real handoff, and public gap seal of this chapter. Its public surface contains the partition-refinement examples, the non-escape rows excluding measure completion, selected tags, quotient function equality, host integrand equality, detached real sequences, and ambient real-completeness data, and the bridge-facing finite partition boundary recorded by Definition 1460.7 and Definition 1460.8. No additional analytic coordinate is part of the mature packet.*

Lean4 ✓ `BEDC.Derived.DarbouxIntegralUp.DarbouxIntegralPacket mature boundary certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxIntegral}^{\uparrow}$ packet is generated from a finite partition row, upper and lower $\mathsf{DyadicStepFunction}^{\uparrow}$ approximants, upper and lower Darboux sum rows, a visible gap ledger, a $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff surface, componentwise hsame transports, finite Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1460.1, Definition 1460.2, Definition 1460.6, Definition 1460.7, and Definition 1460.9 bind the mature finite-partition Darboux real handoff over $\mathsf{Partition}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Integral}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a measure completion, quotient function space, selected tag family, host integrand equality, ambient real-completeness theorem, Lean verification, or bridge checking.

Upgrade path: The next theory step is a bridge theorem relating the public Darboux real handoff to the standard integration boundary.

Chapter 1461

A Concrete Naming Certificate for $\mathsf{RegularCauchyRatioTest}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ packet records the finite ratio-bound test that turns a displayed geometric tail estimate into a regular Cauchy source row before the Real-facing route is consumed. It sits over the regular rational readback of chapter 301, the finite-window discipline of chapter 126, the dyadic tolerance surface of chapter 291, the comparison handoff of chapter 1459, and the partial-sum ledger of chapter 121. The packet is a constructive ratio test for regular Cauchy tails; it is not a quotient of streams, a selected infinite tail, a host real norm, countable choice, or an ambient completeness theorem.

Definition 1461.1 (Regular Cauchy ratio test carrier)**.** *A* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, R, M, W, D, H, C, P, N)$$

with the following displayed rows:

1. *A is the source row for the series or tail family whose consecutive finite terms are inspected.*
2. *R is the ratio-bound row, displaying a dyadic bound strictly inside the unit tolerance scale together with the index after which the bound is claimed.*
3. *M is the modulus row assigning each displayed precision request a finite threshold beyond the ratio-bound index.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family on which the source row, ratio-bound row, modulus row, and downstream partial-sum rows are read.*
5. *D is the dyadic domination ledger that unfolds the finite geometric tail estimate from the ratio row and records the comparison majorant consumed by* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for the source row, ratio-bound row, modulus, finite windows, dyadic domination ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through R and M to the windows in W, then through D to the comparison-test handoff and the downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *row.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*,* $\mathsf{Series}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRatioTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient stream equality, an infinite product or power object, a hidden tail selector, a host real absolute value, a selected completed limit, countable choice, or ambient completeness.

Theorem 1461.2 (Regular Cauchy ratio test NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *carrier* $T = (A, R, M, W, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRatioTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed nine-row packet;*
2. *source exposure for* A *on the finite windows in* W*;*
3. *ratio-bound exposure for* R*, including the displayed dyadic bound and the displayed entry index of the tail;*
4. *modulus readback for* M*, stating that every inspected precision request receives a finite threshold beyond the ratio-bound index;*
5. *dyadic geometric domination in* D*, producing the finite majorant tail consumed by* $\mathsf{RegularCauchyComparisonTest}$
6. *transport stability for the source, ratio-bound, modulus, window, and domination rows along the componentwise* hsame *rows in* H*;*
7. *continuation replay, provenance, and local naming for exactly the route recorded by* C, P, N*.*

These obligations are finite-row obligations over the sibling surfaces of chapter 301, chapter 126, chapter 291, chapter 1459, and chapter 121.

Proof. Project the carrier $T = (A, R, M, W, D, H, C, P, N)$. The projection exposes exactly the source row, ratio-bound row, modulus row, finite window family, dyadic domination ledger, transport rows, continuation replay, provenance, and local naming row.

For a displayed precision request, read the ratio-bound row R and the modulus row M. The ratio row supplies the dyadic bound and the entry index for the tail, while the modulus row supplies a finite threshold beyond that index. Thus the relevant part of the source is inspected only through the windows recorded in W.

The dyadic ledger D unfolds the finite geometric domination calculation. Because every factor and tolerance scale is displayed in D, the comparison majorant sent to $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ is a finite row rather than an appeal to an external infinite estimate.

Finally apply the structural rows. The componentwise hsame entries in H preserve the ratio-bound, threshold, window, and domination data together, while C, P, N only replay, source, and name the displayed route. No omitted coordinate remains from which an additional NameCert obligation could be read. □

Theorem 1461.3 (Regular Cauchy ratio test comparison handoff)**.** *Every accepted* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *carrier hands a regular Cauchy tail to* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *only through the finite route*

$$A, R, M \longmapsto W, D \longmapsto H, C, P, N.$$

Consequently a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can use the ratio test only after reading the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, the ratio-bound row, and the comparison-test handoff row that binds the source to a regular Cauchy majorant.*

Proof. Begin with an accepted carrier and fix a displayed precision request. The rows R and M determine a finite threshold beyond the tail entry index, and W is the only family of source windows admitted for that request.

Read the finite domination ledger. The row D expands the ratio-bound data into a dyadic geometric tail estimate on the selected windows and presents that estimate in the same finite comparison format used by $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$.

Transport preserves the handoff. The componentwise hsame rows in H carry the source row, ratio-bound row, threshold, selected window, and domination certificate together, so a transported carrier cannot alter the comparison-test input while retaining the same displayed packet.

The remaining rows are bookkeeping for the public route. The Cont replay in C, the provenance row P, and the local naming row N send the comparison handoff to $\mathsf{RegSeqRat}^{\uparrow}$ and then to Real-facing consumers only through the named sibling rows. □

Theorem 1461.4 (Regular Cauchy ratio test nonescape)**.** *An accepted* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *carrier cannot expose a regular Cauchy conclusion through any route that omits one of R, M, W, or D. In particular, the carrier surface rejects any consumer that claims the ratio-test conclusion from the source row A alone, from a host real ratio, from an unrecorded infinite tail estimate, or from ambient completeness.*

Proof. Suppose a consumer reads a regular Cauchy conclusion from an accepted carrier. By the NameCert obligation surface of Definition 1461.2, the only conclusion row exported by the carrier is obtained after reading the ratio-bound row R, the modulus row M, the finite windows W, and the dyadic domination ledger D.

If any one of these rows is omitted, the displayed route no longer contains the data needed to form the comparison-test input: without R there is no ratio bound, without M there is no finite threshold, without W there is no admitted source window, and without D there is no dyadic majorant ledger.

The structural rows cannot repair the omission. Transport in H preserves displayed data, and C, P, N replay, source, and name only the displayed route. Therefore any conclusion obtained from a host real ratio, an unrecorded infinite tail estimate, or ambient completeness lies outside the accepted $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ classifier. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ consumer that reads a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ conclusion without the displayed ratio-bound row, modulus row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ domination ledger, and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: series, regseqrat, streamname, dyadicratcore, regular-cauchy-comparison-test: no listed sibling supplies the whole nine-row packet $T = (A, R, M, W, D, H, C, P, N)$, because each sibling contributes only a source, partial-sum, window, dyadic-ledger, or comparison face while this chapter binds the ratio-bound tail route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRatioTest}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ packet is generated from a source row, a displayed dyadic ratio-bound row, an explicit modulus row, finite $\mathsf{StreamName}^{\uparrow}$ windows, dyadic geometric domination rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRatioTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1461.1, Definition 1461.2, Definition 1461.3, and Definition 1461.4 seed the finite ratio-test boundary over $\mathsf{Series}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRatioTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completed limits, host real ratios, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and prove the finite NameCert obligation surface, comparison handoff theorem, and nonescape theorem.

Chapter 1462

A Concrete Naming Certificate for $\mathsf{ConnectedInterval}^{\uparrow}$

Following Bishop and Bridges, constructive connectedness of an interval is recorded by interval-refinement and separation-refusal data rather than by a host topological assertion. The $\mathsf{ConnectedInterval}^{\uparrow}$ packet packages finite closed $\mathsf{Real}^{\uparrow}$ endpoint rows, $\mathsf{NestedInterval}^{\uparrow}$ refinements, $\mathsf{DyadicBisection}^{\uparrow}$ branch ledgers, $\mathsf{IntermediateValue}^{\uparrow}$ sign-route rows, finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observation windows, a visible $\mathsf{Real}^{\uparrow}$ seal route, and local $\mathsf{NameCert}$ evidence into one interval-connectedness certificate.

The certificate depends on the real-name boundary of chapter 38, the finite nested-window carrier of chapter 320, the branch ledger of chapter 361, and the sign-route handoff of chapter 1402. It does not claim open-cover compactness, Heine–Borel, trichotomy, host connectedness, quotient equality of real names, or an ambient theorem about all host intervals.

Definition 1462.1 (Connected interval carrier)**.** *A* $\mathsf{ConnectedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, B, S, T, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the closed endpoint rows read as* $\mathsf{Real}^{\uparrow}$ *names through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks;*
2. *W is the interval-window row naming the displayed finite subintervals inspected between the endpoints;*
3. *B is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger used to refine any inspected separating witness into a finite branch row;*
4. *S is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row obtained from the retained bisection windows;*
5. *T is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row recording the finite route by which a sign-changing separation probe is forced to expose a crossing window;*
6. *E is the visible* $\mathsf{Real}^{\uparrow}$ *seal route offered to downstream interval consumers after the endpoint, refinement, branch, sign-route, and finite-window rows have been read;*
7. *H stores componentwise* hsame *transport for endpoints, windows, bisection rows, nested refinements, sign-route rows, real seals, continuation rows, package provenance, and local naming;*

8. *C stores displayed* Cont *replay from endpoint rows to observation windows, from windows to branch ledgers, from branch ledgers to nested refinements, from sign routes to the refusal row, and from those rows to the real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{ConnectedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}$ *evidence row.*

The carrier predicate asserts that these rows form one finite interval-connectedness packet. Its classifier compares accepted packets only by replaying the displayed coordinates above. There is no coordinate for open-cover compactness, a global host topology, real trichotomy, quotient equality of real names, a selected separating point, or Heine–Borel.

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalCarrierRows`

Theorem 1462.2 (Connected interval NameCert obligations)**.** *For an accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet $K = (L, R, W, B, S, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}$ *surface consists exactly of closed endpoint admission for L and R, finite interval-window admission through W, dyadic branch-ledger exactness through B, nested-refinement stability through S, sign-route refusal through T, real-seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 1462.1. The endpoint, window, branch, nested-refinement, sign-route, real-seal, transport, continuation, provenance, and naming rows are exactly the displayed coordinates of the finite packet.

Read the endpoint and window rows first. The two endpoint rows expose only closed $\mathsf{Real}^{\uparrow}$ names through finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observations, while W records the inspected interval windows between them.

Read the refinement rows next. The dyadic ledger B supplies finite branch choices, and the nested row S repacks the retained windows with the displayed endpoint and width data still attached.

Finally read the refusal and seal rows. The sign-route row T supplies the finite intermediate-value route used by the separation-refusal obligation, and E, H, C, P, N bind that route to the visible real seal and the local certificate. Since the packet has no coordinate for open covers, host topology, trichotomy, quotient equality, selected separators, or Heine–Borel, those data cannot be obligations of the local NameCert. □

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalNameCertObligations`

Theorem 1462.3 (Connected interval separation refusal)**.** *Every accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet refuses a displayed two-sided separation only through finite interval data: any proposed separation read must pass through the endpoint rows L, R, the inspected window row W, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger B, the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row S, and the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row T. The refusal asserts no host connectedness principle beyond those rows.*

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalCarrier separation refusal`

Proof. Begin with a displayed separation read against the accepted packet K. The continuation routes expose the endpoint rows L, R and the finite interval-window row W before any branch or sign information is available.

Refine the inspected window by the dyadic branch ledger. The row B records the finite midpoint decisions, and the row S repacks the retained windows as a nested-refinement packet over the same endpoint and observation data.

Apply the sign-route row. The row T is the only coordinate that can turn opposite-side displayed reads into a crossing-window refusal, so the proposed separation must pass through the same intermediate-value route that generated T.

The resulting refusal is therefore internal to L, R, W, B, S, T and their transport and continuation rows. No step reads a global topology on host intervals, a trichotomy oracle, quotient real equality, a selected separator, compactness, or Heine–Borel. □

Theorem 1462.4 (Connected interval dyadic nesting obligations)**.** *An accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet may expose the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row S only after the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger B has displayed the retained midpoint decisions over endpoint rows L, R and finite window row W. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer then reads S through B, W,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The route exports no host connectedness, trichotomy, selected separator, Heine–Borel, open-cover compactness, or quotient real equality.*

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalDyadicNestingObligations`

Proof. Project the carrier of Definition 1462.1. The nested-refinement row S is not a free coordinate: it is listed downstream of the endpoint rows L, R, the finite interval window W, and the dyadic branch ledger B.

Read the branch ledger before reading S. The ledger B displays the retained midpoint decisions over the same endpoints and window, so the nested row can only repack retained finite intervals already recorded by B.

Use Definition 1462.3 to keep those branch decisions inside the displayed interval data. Any attempted separation read must pass through L, R, W, B, S, T, and therefore cannot supply an external separator or host connectedness proof.

Finally read the consumer-facing rows. The $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observations, together with hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ evidence, keep the refinement route finite before the $\mathsf{Real}^{\uparrow}$ seal is consumed. No row remains for trichotomy, selected separators, Heine–Borel, open-cover compactness, quotient real equality, or ambient host connectedness. □

Theorem 1462.5 (Connected interval nested-bisection ledger)**.** *Every accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet that exposes a nested-bisection read routes its endpoint rows L, R through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger B, the* $\mathsf{NestedInterval}^{\uparrow}$ *row S, the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row T, the* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows H, C, P, N. In carrier coordinates the route is*

$$L, R, W, B, S, T, E, H, C, P, N.$$

It exports no trichotomy, selected separator, quotient real equality, open-cover compactness, or Heine–Borel principle.

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalCarrier nested bisection ledger`

Proof. Project the carrier of Definition 1462.1. The endpoint rows L, R are read only as closed $\mathsf{Real}^{\uparrow}$ names through finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and the nested-bisection read has no coordinate outside $W, B, S, T, E, H, C, P, N$.

Use Definition 1462.2 to order the rows. Endpoint admission and finite interval-window admission occur before the dyadic branch ledger, the nested-refinement row, the sign-route row, and the Real seal, while H, C, P, N keep the same displayed packet under transport, replay, provenance, and local naming.

Use Definition 1462.3 for the sign-route boundary. Any separating read must pass through L, R, W, B, S, T, so the packet cannot introduce trichotomy, a selected separator, host connectedness, or quotient equality of real names.

Finally apply Definition 1462.4. It identifies the B-to-S handoff as the only route by which retained midpoint decisions become a nested-refinement row. The Real-facing read therefore reaches E only after the finite bisection and nested-interval ledger has been exposed. □

Theorem 1462.6 (Connected interval real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet factors through the closed endpoint rows, finite observation windows, dyadic branch ledger, nested-refinement row, intermediate-value sign-route row, displayed* BHist *packet,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}$ *row. The exported real route is a certificate boundary, not quotient equality or host connectedness.*

Proof. Expose the consumer route by projection from the carrier. The real-facing row E is reachable only after the endpoint rows L, R, the finite window row W, the branch ledger B, the nested row S, and the sign-route row T have been read.

Use Definition 1462.2 to exhaust the local certificate surface. Its real-facing obligations are precisely the listed carrier rows together with the transport, continuation, provenance, and naming rows H, C, P, N.

Use Definition 1462.3 for the connectedness boundary. Any read that treats the packet as an interval-connectedness object first passes through the finite separation-refusal route, so the real seal cannot be reinterpreted as host connectedness, quotient equality of real names, trichotomy, open-cover compactness, or Heine–Borel. □

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalRealRoute`

Theorem 1462.7 (Connected interval obligation closure route)**.** *Every accepted* $\mathsf{ConnectedInterval}^{\uparrow}$ *packet presents a finite BEDC induction route from the connected-interval carrier through dyadic nesting obligations, separation refusal, nested-bisection ledger, and Real-route nonescape. The route uses only the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows already exposed by the packet, and it exports no open-cover compactness, trichotomy, quotient interval, selected separator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.ConnectedIntervalUp.ConnectedIntervalObligationClosureRoute`

Proof. Start from the carrier of Definition 1462.1. Its finite BHist packet displays the endpoint rows, interval windows, dyadic branch ledger, nested-refinement row, sign-route row, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Perform induction on the nested-bisection ledger. The base read is the accepted endpoint and finite-window surface, using the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observations named by the carrier. The induction step is the retained dyadic branch row: Definition 1462.4 routes each retained midpoint decision through the existing $\mathsf{DyadicRatCore}^{\uparrow}$-facing branch ledger before the nested-refinement row may be read.

At each step, apply Definition 1462.3. A proposed two-sided separation must pass through the endpoint rows, the inspected window, the dyadic ledger, the nested row, and the intermediate-value

sign route; it cannot supply trichotomy, a selected separator, quotient interval equality, open-cover compactness, or host connectedness.

The induction closes by Definition 1462.5 and Definition 1462.6. These results identify the complete row order and force every $\mathsf{Real}^{\uparrow}$-facing consumer to pass through the same finite packet before the Real seal is visible. Thus the obligation surface is the displayed BEDC route, not ambient $\mathsf{Real}^{\uparrow}$ completeness or any off-packet compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConnectedInterval}^{\uparrow}$ packet is generated from closed Real endpoint rows, finite StreamName windows, RegSeqRat readbacks, DyadicBisection branch ledgers, NestedInterval refinements, IntermediateValue sign-route rows, a Real seal route, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1462.1, Definition 1462.2, Definition 1462.3, Definition 1462.4, Definition 1462.5, Definition 1462.6, and Definition 1462.7 export the public connected-interval route through endpoint rows, $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ finite windows, $\mathsf{DyadicBisection}^{\uparrow}$ branch ledger, $\mathsf{NestedInterval}^{\uparrow}$ refinement row, $\mathsf{IntermediateValue}^{\uparrow}$ sign route, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert dependencies.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConnectedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel, trichotomy, host connectedness, quotient equality of real names, selected separators, Lean verification, or a standard bridge.

Upgrade path: A standard bridge for the public connected-interval route remains the next paper-side obligation, while Lean verification remains the formal-axis obligation for the same endpoint, finite-window, dyadic-branch, nested-refinement, sign-route, Real-seal, transport, replay, provenance, and local naming surface in BEDC.Derived.ConnectedIntervalUp.

Chapter 1463

Completion Uniqueness NameCert Construction

Following Bishop and Bridges-style separated completion uniqueness, the $\mathsf{CompletionUniqueness}^{\uparrow}$ packet records the finite BEDC route by which two completion-facing presentations are identified only through a displayed separated seal and a universal-property uniqueness ledger. It sits downstream of the separated surface of chapter 402, the extension-and-uniqueness surface of chapter 397, and the functorial route of chapter 423. The packet also reads the regular real-name spine through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, without turning those rows into quotient completion data.

Definition 1463.1 (Completion uniqueness carrier)**.** *A* $\mathsf{CompletionUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, E, H, F, L, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the compared regular names;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by both sides;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational source row carried by the shared schedule and tolerance ledger;*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from D, S, R;*
5. *H is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated seal that makes the comparison a separated-completion read rather than a quotient-class read;*
6. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *functorial handoff that transports the same regular source into each completion-facing presentation;*
7. *L is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *uniqueness ledger comparing the two transported extension rows over the same D, S, R, E, H, F spine;*
8. *T records the componentwise* hsame *transports for the displayed dyadic, stream, regular, Real, separated, functorial, and uniqueness rows;*
9. *C records the displayed* Cont *routes that replay the separated-limit and uniqueness reads;*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. N is the local $\mathsf{NameCert}_{\mathsf{CompletionUniqueness}^{\uparrow}}$ *row.*
The carrier contains no quotient completion object, no selected host limit, no ambient Real completeness field, and no host equality row.

Lean4 ✓ `BEDC.Derived.CompletionUniquenessUp.CompletionUniquenessTasteGate single carrier alignment`

Theorem 1463.2 (Completion uniqueness NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompletionUniqueness}^{\uparrow}}$ *surface for the carrier of Definition 1463.1 consists of five finite rows: carrier admission for the displayed packet U; classifier stability under the componentwise* hsame *transports T and* Cont *routes C; separated-seal compatibility through the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *row H; functorial source compatibility through the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row F; and universal-property ledger exactness through the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row L.*

Proof. Carrier admission is projection of Definition 1463.1. The dyadic tolerance ledger, finite stream schedule, regular-rational source, Real seal, separated-completion seal, functorial handoff, uniqueness ledger, transport rows, continuation rows, package provenance, and local naming row are displayed coordinates of one finite packet.

Classifier stability is componentwise. The transports in T move D, S, R, E, H, F, L, N without changing the dependency order, and the continuation rows in C replay only the separated-limit and uniqueness reads already exposed by those coordinates.

The remaining obligation rows are sibling handoffs. The separated surface is read through H, the functorial source route is read through F, and the uniqueness ledger is read through L after the shared D, S, R, E spine has been projected. Since no carrier coordinate contains a quotient completion, selected host limit, ambient Real completeness field, or host equality row, the listed obligations exhaust the local NameCert surface. □

Theorem 1463.3 (Completion uniqueness separated limit)**.** *Let $U = (D, S, R, E, H, F, L, T, C, P, N)$ be an accepted* $\mathsf{CompletionUniqueness}^{\uparrow}$ *carrier. Any comparison between the two completion-facing presentations exported by F and L first passes through the displayed route*

$$D, \quad S, \quad R, \quad E, \quad H.$$

Thus the comparison is a separated-limit read over the dyadic, stream, regular-rational, Real-seal, and Hausdorff-completion rows, not a host equality of completed points.

Proof. Project the accepted carrier. The rows D, S, R, E give the finite regular-name spine, and the row H is the only separated-completion seal attached to that spine.

Read the functorial and universal-property rows after the separated seal. The functorial handoff F transports the same regular source into the two completion-facing presentations, while the ledger L compares their extension rows only over the already displayed source and seal data.

The continuation row C can replay the comparison only along that route, and the provenance row P records no host completion object. Therefore every comparison is a separated-limit read over the displayed finite packet. □

Theorem 1463.4 (Completion uniqueness ledger exhaustion)**.** *For every accepted* $\mathsf{CompletionUniqueness}^{\uparrow}$ *carrier, the rows $D, S, R, E, H, F, L, T, C, P, N$ exhaust the proof surface for constructive separated completion uniqueness. A consumer may inspect dyadic tolerances, stream windows, regular rational source data, the Real seal, the Hausdorff separated seal, the Cauchy-completion functor handoff, the universal-property uniqueness ledger, componentwise transports, continuation routes, package provenance, and the local NameCert row. It obtains no quotient completion, selected host limit, ambient Real completeness theorem, or host equality witness.*

Proof. Start from the carrier decomposition of Definition 1463.1. Every coordinate available to a consumer is one of $D, S, R, E, H, F, L, T, C, P, N$.

Apply Definition 1463.2 to bind classifier stability and obligation exactness to those rows. Apply Definition 1463.3 to route every completion-facing comparison through the separated finite spine before the uniqueness ledger is read.

No additional field remains in the packet. In particular, there is no coordinate for quotient completion construction, selected host limits, ambient Real completeness, or host equality. The displayed rows therefore exhaust the constructive uniqueness ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionUniqueness}^{\uparrow}$ packet is generated from a dyadic tolerance ledger, a stream-window schedule, a regular rational source row, a RealUp seal, a Hausdorff separated seal, a CauchyCompletionFunctor handoff, a CompletionUniversalProperty uniqueness ledger, componentwise transports, continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1463.1, Definition 1463.2, Definition 1463.3, and Definition 1463.4 seed the separated constructive completion uniqueness route over the finite Real-completion spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CompletionUniquenessUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected host limits, ambient Real completeness, or host equality.

Upgrade path: A checked carrier theorem and obligation package for the same dyadic, stream, regular-rational, Real-seal, Hausdorff, functorial, universal-property, transport, continuation, package, and NameCert route would raise the verification axis.

Chapter 1464

A Concrete Naming Certificate for $\mathsf{CauchyCondensation}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensation}^{\uparrow}$ packet records the dyadic-block test for a monotone nonnegative series as finite BEDC data before any completion-facing consumer may read a real seal. Its source and tail rows are the series surface of chapter 121; its comparison budget sits beside chapter 367; its regular rational readback is the handoff surface of chapter 301; and its terminal seal is the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet is a constructive condensation certificate; it is not an ambient summability principle, a quotient of infinite series, a selected sum, host real equality, countable choice, trichotomy on arbitrary summands, or RealUp completeness.

Definition 1464.1 (Cauchy condensation carrier)**.** *A* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, B, L, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the monotone nonnegative* $\mathsf{Series}^{\uparrow}$ *summand source, read only through finite windows and partial-sum rows.*
2. *W is the summand-window ledger, recording the finite intervals on which monotonicity and nonnegativity are inspected.*
3. *B is the dyadic block schedule, listing the powers of two, block endpoints, and block lengths consumed by the condensation test.*
4. *L is the block-sum ledger, storing the original finite block sums and the condensed dyadic terms $2^n a_{2^n}$ over the same schedule.*
5. *T is the tail-bound row, packaging displayed tolerances, dyadic thresholds, and the domination inequalities that compare original tails with condensed block tails.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the block-sum and tail-bound ledgers.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the regular rational readback.*
8. *H records componentwise* hsame *transports for the source, windows, schedule, block sums, tail bounds, regular readback, real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *routes from each finite precision request through W, B, L, T to the readback row R and the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for an infinite choice of blocks, a quotient series equality, a host limit, a selected completed real, a trichotomy oracle, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationTasteGate single carrier alignment`

Theorem 1464.2 (Cauchy condensation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier* $K = (S, W, B, L, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *monotone nonnegative source exposure for S through finite* $\mathsf{Series}^{\uparrow}$ *summand and partial-sum windows;*
3. *window compatibility for W, stating that every inspected monotonicity and nonnegativity claim is tied to displayed summand rows;*
4. *dyadic schedule admission for B, with powers of two, endpoints, and block lengths available as finite ledger entries;*
5. *block-sum formation for L, relating each original finite block sum to the condensed term on the same schedule;*
6. *tail-bound compatibility for T, routing finite block domination into the comparison budget consumed by* $\mathsf{FastConvergentSeries}^{\uparrow}$*;*
7. $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal exposure for R and E;*
8. *transport stability, continuation replay, provenance, and local naming for exactly the displayed route recorded by H, C, P, N.*

No other analytic or choice-theoretic obligation is part of the local certificate.

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier namecert obligations`

Proof. Project the accepted packet $K = (S, W, B, L, T, R, E, H, C, P, N)$. The projection exposes exactly the monotone source, finite windows, dyadic schedule, block-sum ledger, tail-bound row, regular readback, real seal, transport rows, continuation replay, provenance, and local naming row.

Read a finite tail request through W and B. The schedule selects finitely many dyadic blocks, and the ledger L stores both the original block sums and the condensed terms on those same indices.

The monotone nonnegative source row S is used only through displayed summand windows. The tail-bound row T then turns the finite block comparisons into the comparison budget consumed by the fast-convergent-series surface, so the argument never appeals to an infinite choice of block representatives.

Finally use the structural rows. The componentwise transports in H preserve the displayed data, and C, P, N only replay, source, and name that route. Since the carrier has no further coordinate, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$ surface. □

Theorem 1464.3 (Cauchy condensation dyadic-tail obligation)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier* $K = (S, W, B, L, T, R, E, H, C, P, N)$*, any tail request presented to the condensation comparison must expose the dyadic schedule* B*, the block-sum ledger* L*, and the tail-bound row* T *before it can be read by the regular-rational or real-seal rows. Equivalently, the tail obligation is the finite projection*

$$S, W, B, L \longmapsto T$$

inside the same displayed carrier, with no detached infinite block choice or external summability premise.

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.DyadicTailObligation.CauchyCondensationCarrier dyadic tail obligation`

Proof. Project the accepted packet as $K = (S, W, B, L, T, R, E, H, C, P, N)$. The NameCert obligation surface of Definition 1464.2 exposes S, W, B, L, T as the source, window, dyadic schedule, block-sum, and tail-bound rows.

Apply Definition 1464.6. It forces every tail request through the finite route from S, W, B, L to T before R or E can be inspected.

The structural rows H, C, P, N only transport, replay, source, and name that route. Since the carrier has no coordinate for infinite block selection or host summability, the dyadic-tail obligation is exactly the displayed projection. □

Theorem 1464.4 (Cauchy condensation RegSeqRat readback obligation)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier* $K = (S, W, B, L, T, R, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is readable only after the finite dyadic-tail obligation through* S, W, B, L, T *has been displayed. A consumer that preserves* S, W, B, L, T *and the structural rows* H, C, P, N *preserves the same readback row* R*, and no quotient series equality, selected sum, or countable-choice schedule supplies an alternate readback.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.RegSeqRatReadbackObligation.CauchyCondensationCarrier regseqrat readback obligation`

Proof. Use Definition 1464.3 to bind every tail request to the displayed rows S, W, B, L, T. This places the tail-bound row immediately upstream of the regular-rational handoff.

Now apply Definition 1464.6. The theorem identifies R as the readback row reached from the dyadic-block route and the same structural rows.

Finally use Definition 1464.8. Componentwise hsame transport preserving W, B, L, T preserves the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row, so no quotient, selected-sum, or choice-based alternate readback is available. □

Theorem 1464.5 (Cauchy condensation Real-seal nonescape obligation)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier* $K = (S, W, B, L, T, R, E, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read of* E *must first pass through the dyadic-tail obligation on* S, W, B, L, T *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obligation on* R*. Thus the Real-seal obligation is the finite route*

$$S, W, B, L, T \longmapsto R \longmapsto E, H, C, P, N,$$

with no route through quotient series equality, selected infinite sums, countable choice, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.RealSealObligation.CauchyCondensationCarrier real seal nonescape obligation`

Proof. Start with the dyadic-tail obligation of Definition 1464.3. It fixes the finite route through S, W, B, L, T for every tail request.

Apply Definition 1464.4. The readback row R is downstream of the same finite dyadic-tail route and cannot be replaced by an external summability object.

Finally apply Definition 1464.7. The seal row E receives its analytic input from R together with H, C, P, N. Since the packet omits quotient equality, selected sums, countable choice, trichotomy, host real equality, and $\mathsf{Real}^{\uparrow}$ completeness, the displayed route exhausts the Real-seal obligation. □

Theorem 1464.6 (Cauchy condensation dyadic-block handoff)**.** *Every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier sends a monotone nonnegative series tail to a completion-facing comparison route only through the finite dyadic-block handoff*

$$S, W, B, L \longmapsto T \longmapsto R, E, H, C, P, N.$$

Thus a downstream consumer can read the condensation comparison only after reading the displayed summand windows, dyadic block schedule, block-sum ledger, tail-bound row, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier dyadic block handoff`

Proof. Begin with an accepted carrier and a displayed tail request. The window ledger W identifies the finite summand rows to inspect, while the dyadic schedule B fixes the block endpoints and powers of two.

Read the source row S through those windows. The block ledger L records the corresponding original block sums together with the condensed terms $2^n a_{2^n}$, so every comparison is attached to the displayed schedule.

Now apply the tail-bound row T. It packages the finite domination inequalities and tolerance thresholds that route the block comparison into the regular rational readback. No consumer can reach R without the data supplied by S, W, B, L, T.

Transport and replay preserve this route componentwise. The rows H, C, P, N therefore attach the handoff to the local certificate without creating an alternate path through host summability, selected sums, or completion. □

Theorem 1464.7 (Cauchy condensation Real-seal nonescape)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R produced from the finite dyadic-block route. The seal row E receives the monotone source, summand windows, dyadic schedule, block-sum ledger, tail bounds, componentwise transports, continuation replay, provenance, and local naming rows, but it receives no quotient series equality, selected infinite sum, countable choice, trichotomy premise, host real equality, or RealUp completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier real seal nonescape`

Proof. Project the seal-facing rows R and E from the accepted carrier. The regular handoff R is downstream of the tail-bound row T, and T is downstream of the displayed summand windows, dyadic schedule, and block-sum ledger.

Apply Definition 1464.6. It identifies the only route from a monotone nonnegative tail to the readback row as the finite path through S, W, B, L, T and the structural rows attached to the same packet.

Read the seal row E. Its analytic input is exactly R, together with H, C, P, N. Since the carrier has no coordinate for quotient series equality, a selected sum, countable choice, trichotomy outside the displayed windows, host real equality, or RealUp completeness, the seal-facing read cannot export any of those objects. □

Theorem 1464.8 (Cauchy condensation dyadic-block tail transport)**.** *Let*

$$K = (S, W, B, L, T, R, E, H, C, P, N)$$

and

$$K' = (S', W', B', L', T', R', E', H', C', P', N')$$

be accepted $\mathsf{CauchyCondensation}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies the summand windows, dyadic block schedule, block-sum ledger, and tail-bound rows* W, B, L, T *with* W', B', L', T'*, then it preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows* R, R' *and the* $\mathsf{Real}^{\uparrow}$ *seal nonescape boundary rows* E, E'*. The transported packet still routes every condensation tail through the displayed finite block surface before any real-seal read.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier dyadic block tail transport`

Proof. Unpack the NameCert obligations of Definition 1464.2. The accepted carrier exposes the summand windows, dyadic schedule, block-sum ledger, and tail-bound row as finite rows before the regular readback or real seal can be inspected.

Apply Definition 1464.6. It identifies the only route to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff as the finite path through S, W, B, L, T and the structural rows. Therefore transporting W, B, L, T through H preserves the handoff row R along the same displayed route.

Apply Definition 1464.7. The real seal receives its analytic input from R and the transported structural rows, so preserving the handoff preserves the nonescape boundary E as well.

The carrier contains no quotient series equality, selected infinite sum, countable-choice schedule, trichotomy premise, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. Thus the transported packet remains a finite dyadic-block tail route. □

Theorem 1464.9 (Cauchy condensation tail-lock uniqueness)**.** *For an accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier* $K = (S, W, B, L, T, R, E, H, C, P, N)$*, any two* $\mathsf{Real}^{\uparrow}$ *seal-facing reads that preserve the same summand windows* W*, dyadic block schedule* B*, block-sum ledger* L*, and tail-bound row* T *must factor through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before reaching the same Real seal boundary* E*. No quotient series equality, selected sum, countable choice, trichotomy, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can distinguish the two reads.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier tail lock uniqueness`

Proof. Use the NameCert obligations of Definition 1464.2. They expose the carrier only through $S, W, B, L, T, R, E, H, C, P, N$, so both seal-facing reads share the same finite condensation surface once W, B, L, T are preserved.

Apply Definition 1464.6. The handoff theorem forces each read through the displayed path from S, W, B, L to T and then to the regular rational row R.

Finally apply Definition 1464.7 and Definition 1464.8. The Real seal receives its analytic input from the same handoff row and the same transported structural rows, so both reads meet the same boundary E without an alternate quotient, selected-sum, choice, trichotomy, host-equality, or completeness discriminator. □

Theorem 1464.10 (Cauchy condensation test dyadic-block equivalence)**.** *An accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *packet and a* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *consumer agree exactly when the consumer reads the same displayed monotone source tail* S*, summand-window ledger* W*, dyadic block schedule* B*, block-sum ledger* L*, tail-bound row* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and* $\mathsf{Real}^{\uparrow}$ *seal route* E *through the structural rows* H, C, P, N*. Equivalently, the consumer may use the condensation-test conclusion only through the finite dyadic-block route already carried by*

$$K = (S, W, B, L, T, R, E, H, C, P, N),$$

and no quotient series equality, selected sum, countable-choice schedule, trichotomy oracle, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate supplies a second route.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier test dyadic block equivalence`

Proof. Start with the accepted carrier and apply Definition 1464.6. It identifies the displayed path from the monotone source tail through summand windows, dyadic schedule, block sums, and tail bounds to the regular readback.

Use Definition 1464.8. Any consumer that preserves the source-tail windows, block schedule, block-sum ledger, and tail-bound row preserves the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the Real-seal boundary along the same route.

Finally apply Definition 1464.9. Two seal-facing reads with the same W, B, L, T data cannot be distinguished by an alternate quotient, selected-sum, choice, trichotomy, host-equality, or completeness coordinate. Thus the condensation-test consumer agrees with the packet exactly on the displayed dyadic-block route. □

Theorem 1464.11 (Cauchy condensation scoped obligation route)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier*

$$K = (S, W, B, L, T, R, E, H, C, P, N),$$

any consumer read that cites the dyadic-block comparison factors first through the finite $\mathsf{Series}^{\uparrow}$ *summand windows, the* $\mathsf{FastConvergentSeries}^{\uparrow}$ *comparison budget recorded in* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* BHist *and* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$ *row. No consumer may cite the comparison theorem from an unsealed series tail, a detached completion object, or an external summability principle.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier scoped obligation route`

Proof. First use the dyadic-block handoff of Definition 1464.6. It sends every accepted tail request through the displayed $\mathsf{Series}^{\uparrow}$ source windows, the dyadic schedule, the block-sum ledger, and the tail-bound row before the regular readback can be inspected.

Next apply the Real-seal nonescape theorem of Definition 1464.7. The seal-facing read reaches E only after the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the structural rows attached to the same finite carrier are visible.

Transport is supplied by Definition 1464.8. It preserves the same windows, schedule, block ledger, tail-bound row, regular readback, and Real-seal boundary under componentwise hsame transport.

Finally use the test equivalence of Definition 1464.10. A consumer can cite the dyadic-block comparison exactly when it agrees with the displayed route through $S, W, B, L, T, R, E, H, C, P, N$, so the scoped obligation route is the finite carrier route itself rather than a detached analytic principle. □

Theorem 1464.12 (Cauchy condensation obligation-closure package)**.** *For every accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier*

$$K = (S, W, B, L, T, R, E, H, C, P, N),$$

the carrier, local $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$ *obligations, dyadic-tail obligation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obligation,* $\mathsf{Real}^{\uparrow}$ *seal nonescape obligation, dyadic-block handoff, tail transport, tail-lock uniqueness, and scoped obligation route assemble the full* obligation *surface over* $\mathsf{Series}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The package exports no selected sum, quotient series equality, countable choice, trichotomy, detached summability principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationUp.CauchyCondensationCarrier obligation closure package`

Proof. Start from the carrier and NameCert surfaces. Apply Definition 1464.1 and Definition 1464.2 to expose exactly the eleven displayed rows and the local certificate obligations attached to them.

Next bind the analytic tail route. Use Definition 1464.3, Definition 1464.4, and Definition 1464.5 to force every tail request through S, W, B, L, T, then R, then E, H, C, P, N.

Now add the comparison and transport layer. The dyadic-block handoff of Definition 1464.6 supplies the finite route consumed by $\mathsf{FastConvergentSeries}^{\uparrow}$, while Definition 1464.8 preserves that route under componentwise hsame transport.

Finally apply Definition 1464.9 and Definition 1464.11. They identify the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ seal boundary for all consumer reads that preserve the displayed finite data. Since every cited theorem uses only the rows of K, no selected sum, quotient series equality, countable choice, trichotomy, detached summability principle, or ambient completeness row is introduced. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensation}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed summand windows, dyadic block schedule, block-sum ledger, tail-bound row, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: series, fastconvergentseries, regseqrat, real: no listed sibling supplies the whole eleven-row packet $K = (S, W, B, L, T, R, E, H, C, P, N)$, because each sibling contributes only a source, comparison budget, regular readback, or seal face while this chapter binds all four faces with dyadic block scheduling, block-sum ledgers, tail bounds, transports, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCondensation}^{\uparrow}$ packet is generated from a monotone nonnegative $\mathsf{Series}^{\uparrow}$ source, finite summand windows, a dyadic block schedule, block-sum ledgers, tail-bound rows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1464.12 binds the carrier, local NameCert obligations, dyadic-tail obligation, RegSeqRat readback, Real-seal nonescape, dyadic-block handoff, tail transport, tail-lock uniqueness, scoped obligation route, replay, provenance, and local naming rows over the displayed finite packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCondensation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ambient summability, quotient series equality, selected sums, countable choice, trichotomy, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires kernel-scoped dependency wording over the same carrier, dyadic-tail, RegSeqRat readback, Real-seal, transport, replay, provenance, and local NameCert rows.

Chapter 1465

A Concrete Naming Certificate for $\mathsf{IntervalBisection}^{\uparrow}$

Following Bishop–Bridges constructive interval bisection, the $\mathsf{IntervalBisection}^{\uparrow}$ packet records the explicit split of a displayed located interval into left and right located subintervals through a finite midpoint route. It consumes the endpoint-cell surface of chapter 394, the dyadic-cell surface of chapter 306, the rational-cell surface of chapter 354, the finite-window interface of chapter 126, the regular rational readbacks of chapter 301, the terminal real seal of chapter 38, and the finite-net consumer boundary of chapter 1433. The packet is an explicit bisection certificate for one located interval; it is not open-cover compactness, trichotomy, quotient real equality, or a choice principle for selecting a branch.

Definition 1465.1 (IntervalBisection carrier)**.** *An* $\mathsf{IntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, M, L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the parent* $\mathsf{LocatedInterval}^{\uparrow}$ *row, including its lower and upper endpoint observations and locatedness ledger;*
2. *M is the midpoint route, computed through the displayed* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint cells and recorded as a midpoint row before any branch is inspected;*
3. *L is the left subinterval row with inherited lower endpoint, midpoint upper endpoint, and its local locatedness witness;*
4. *R is the right subinterval row with midpoint lower endpoint, inherited upper endpoint, and its local locatedness witness;*
5. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger aligning the parent cell, midpoint cell, and the two child cells at the requested mesh scale;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the parent endpoints, midpoint read, and child endpoint observations;*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite windows, including the rational midpoint calculation and the endpoint reuse checks;*
8. *E is the pair of* $\mathsf{Real}^{\uparrow}$ *seal rows reached by the left and right child intervals through S, Q, D and through no detached quotient-real equality coordinate;*

9. *H records componentwise* hsame *transport for the parent interval, midpoint route, left and right subinterval rows, dyadic ledger, windows, readbacks, real seals, and branch handoff;*
10. *C records displayed* Cont *routes by which consumers replay the parent row, midpoint route, both child rows, the dyadic ledger, finite windows, readbacks, real seals, and the nested-cell handoff;*
11. *P is the* Pkg *provenance over* $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalBisection}^{\uparrow}}$ *row.*

The carrier exports only the displayed parent-to-midpoint-to-children route. It has no coordinate for an arbitrary open cover, selected subcover, trichotomy on real names, quotient real equality, hidden endpoint choice, or ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionTasteGate single carrier alignment`

Theorem 1465.2 (IntervalBisection NameCert obligations)**.** *For an accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *carrier* $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalBisection}^{\uparrow}}$ *surface consists exactly of carrier admission for those twelve rows, midpoint formation from the displayed parent endpoint cells, left and right located-subinterval admission, dyadic refinement compatibility, finite-window readback compatibility, child real-seal exposure, componentwise classifier transport, consumer replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionNamecertObligations`

Proof. Carrier admission is the tuple of Definition 1465.1. The parent interval, midpoint route, two child intervals, dyadic ledger, finite windows, regular readbacks, real seals, transports, continuations, package provenance, and local naming row are all explicit BHist coordinates.

The midpoint obligation is read through I, M, Q. The parent row supplies the displayed endpoint cells, the midpoint route records the rational midpoint before branch inspection, and the regular readback row checks that the midpoint is read from the same finite endpoint windows.

The child-interval obligations are read through L, R, D, S, Q, E. The left row reuses the lower endpoint of I and the midpoint upper endpoint; the right row reuses the midpoint lower endpoint and the upper endpoint of I. The dyadic ledger, finite windows, readbacks, and real seals expose that both child rows remain inside the same displayed completion route.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for branch choice, trichotomy, quotient real equality, selected subcovers, or ambient compactness, none of those objects can become local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionCarrier namecert obligations`

Theorem 1465.3 (IntervalBisection endpoint transport)**.** *Every endpoint-facing consumer of an accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet receives the displayed transport route*

$$I, M, L, R, D, S, Q, E, H, C, P, N.$$

The route preserves the parent lower endpoint into the left child, the parent upper endpoint into the right child, and the midpoint endpoint into both children by componentwise hsame *transport over the displayed* BHist *rows.*

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionEndpointTransport`

Proof. Begin with the parent row I and midpoint route M. These two rows determine the three endpoint positions that can be exported: the parent lower endpoint, the midpoint endpoint, and the parent upper endpoint.

Project the child rows L and R. The left child has the inherited lower endpoint and the midpoint upper endpoint; the right child has the midpoint lower endpoint and the inherited upper endpoint. No further endpoint selector appears in the carrier.

Now read D, S, Q, E. The dyadic ledger aligns the parent and child cells, the finite windows expose only the required endpoint reads, the regular readbacks check the same midpoint calculation, and the real seals close the two child endpoint routes.

Finally the structural rows H, C, P, N transport, replay, source, and name this endpoint route. Componentwise hsame is therefore the only endpoint-transport mechanism available to a consumer. □

Theorem 1465.4 (IntervalBisection nested-cell handoff)**.** *Every nested-interval or finite-net consumer of an accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet factors its child-cell request through the displayed route*

$$I, M, L, R, D, H, C, P, N$$

before it may read $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *data downstream. Thus the packet hands off a pair of finite child cells, not a selected infinite branch or an open-cover compactness argument.*

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionCarrier nested cell handoff`

Proof. First read the parent interval and midpoint route. The row I fixes the displayed located interval, and M fixes the midpoint row used by both children.

Next use the child rows and dyadic ledger. The rows L and R are the only child-cell exports, while D records their refinement relation to the parent cell and midpoint cell.

The handoff to downstream consumers is then replayed through H, C, P, N. A nested-interval consumer may iterate the same finite packet shape, and a $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ consumer may read the two displayed child cells as finite mesh data, but neither consumer receives an infinite branch selector or an open-cover subcover selector from this packet.

The stream, regular-rational, and real-seal rows S, Q, E remain available only as finite readbacks for the already displayed cells. They cannot introduce quotient real equality or a hidden choice of a child branch because those coordinates are absent from the carrier. □

Theorem 1465.5 (IntervalBisection obligation carrier exposure)**.** *Every accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet exposes its root-unblock carrier rows before any located interval, dyadic interval, rational interval, StreamName, RegSeqRat, Real, or closed-interval compact-net consumer reads the bisection. The exposed rows are exactly the parent* $\mathsf{LocatedInterval}^{\uparrow}$ *row* I*, midpoint route* M*, left and right child rows* L, R*,* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*,* $\mathsf{Real}^{\uparrow}$ *child seals* E*, componentwise transport* H*, replay* C*, provenance* P*, and local naming* N*.*

Proof. Project the carrier of Definition 1465.1. Its displayed tuple is $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$, so the only carrier data available to a consumer are precisely those rows.

The interval-facing rows are I, M, L, R, D. They expose the located parent cell, the rational midpoint route, both child cells, and the dyadic refinement ledger before any downstream readback is opened.

The readback-facing rows are S, Q, E. They expose finite StreamName windows, regular rational readback, and the two Real child seals only after the bisection cells have been displayed.

The structural rows H, C, P, N preserve and name the same packet. Since no additional carrier coordinate exists, every listed consumer reads the bisection through the displayed root-unblock surface. □

Lean4 ✓ `BEDC.Derived.IntervalBisectionUp.IntervalBisectionObligationCarrier`

Theorem 1465.6 (IntervalBisection obligation transport)**.** *Let $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$ and $K' = (I', M', L', R', D', S', Q', E', H', C', P', N')$ be accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packets. If the located interval, midpoint, child interval, dyadic ledger, finite-window, readback, and Real-seal rows transport componentwise under the displayed* hsame *rows and replay through the displayed* Cont *routes, then every located interval, dyadic interval, rational interval, StreamName, RegSeqRat, Real, and* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *consumer receives the same bisection obligation surface from K and K'.*

Proof. The componentwise hypotheses identify the interval rows I, M, L, R, D with I', M', L', R', D' through the transport rows H, H'. Thus the parent cell, midpoint route, two child cells, and dyadic refinement ledger determine the same finite bisection shape.

The readback rows transport in the same way. The finite windows S, S', regular readbacks Q, Q', and Real seals E, E' are replayed through C, C', so no consumer can observe a second bisection route outside the transported rows.

Finally P, N and P', N' source and name the transported packet. The consumer list is exhausted by those interval, readback, and structural rows, so the transported packets expose the same obligation surface. □

Theorem 1465.7 (IntervalBisection obligation ledger exhaustion)**.** *For every accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet, the root-unblock ledger for located interval, dyadic interval, rational interval,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *consumers is exhausted by the rows*

$$I, M, L, R, D, S, Q, E, H, C, P, N.$$

The ledger contains no row for selected infinite branch choice, open-cover compactness, quotient real equality, host endpoint equality, trichotomy, or an ambient compactness theorem.

Proof. Use the carrier exposure theorem to list the only rows in the packet. The located, rational, and dyadic interval consumers read I, M, L, R, D; the StreamName, RegSeqRat, and Real consumers read S, Q, E after those interval rows have been displayed.

Use the transport theorem for stability. Any replayed or transported bisection obligation is still read through H, C, P, N together with the same interval and readback rows, so the ledger has no second channel.

The excluded objects are absent from the carrier and from the NameCert obligations of Definition 1465.2. Therefore the listed tuple exhausts the root-unblock ledger for all named consumers. □

Theorem 1465.8 (IntervalBisection midpoint formation)**.** *For every accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$, the midpoint row M is formed only from the displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *parent endpoint rows and the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint cells read through I and Q. No midpoint obligation may inspect a child branch, a Real equality row, or a compactness consumer before M has been displayed.*

Proof. Project I and Q from the carrier. The parent interval row supplies the lower and upper endpoint observations, while the regular rational readback records the rational endpoint cells visible to the bisection packet.

The midpoint row M is placed between the parent row and the child rows in Definition 1465.1. Hence it is available before L, R, D, S, Q, E are used as child-facing data, and its inputs are exactly the displayed endpoint cells.

Since the carrier has no separate midpoint selector, Real equality coordinate, or compactness oracle, the midpoint obligation is exhausted by the located parent endpoint rows and the rational endpoint-cell calculation. □

Theorem 1465.9 (IntervalBisection child-cell admission)**.** *For every accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet* $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$*, the left and right child cells* L, R *are admitted through the displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *child seals* E*. The packet admits exactly the inherited-left and inherited-right child cells; it does not admit a selected infinite branch or a quotient-real endpoint.*

Proof. Read the child rows L, R together with the dyadic ledger D. These rows display the inherited parent endpoints, the midpoint endpoint, and the refinement relation between parent and child cells.

Next read S, Q, E. The finite windows expose only the endpoint observations needed by the displayed cells, the regular rational row checks the midpoint calculation and endpoint reuse, and the Real seals close the two child routes after those finite rows are visible.

The admitted children are therefore the two displayed cells named by L, R. No carrier coordinate selects one child as an infinite branch, and no quotient-real equality row is present for replacing the displayed endpoint route. □

Theorem 1465.10 (IntervalBisection finite-net handoff)**.** *Every* $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntervalBisection}^{\uparrow}$ *packet reads the bisection through the finite rows*

$$I, M, L, R, D, S, Q, E, H, C, P, N.$$

The handoff supplies parent and child mesh cells with finite readback and Real seals, but it supplies no selected branch, no selected subcover, and no open-cover compactness theorem.

Proof. Start with the ledger exhaustion theorem. It lists the only rows available to located-interval, dyadic-interval, StreamName, RegSeqRat, Real, and closed-interval compact-net consumers.

A finite-net consumer first sees the parent interval, midpoint, child cells, and dyadic refinement rows I, M, L, R, D. These rows supply a finite mesh refinement rather than an infinite nested branch.

The readback rows S, Q, E then expose the finite observation windows, rational endpoint checks, and Real child seals. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite handoff without adding an open-cover or selected-subcover coordinate. □

Theorem 1465.11 (IntervalBisection obligation closure package)**.** *The carrier exposure, obligation transport, obligation ledger, midpoint formation, child-cell admission, and finite-net handoff theorems form the obligation-closure package for* $\mathsf{IntervalBisection}^{\uparrow}$*. Every root-unblock consumer named in this chapter receives exactly the displayed bisection route, and every excluded object remains outside the carrier.*

Proof. The carrier exposure theorem fixes the visible tuple $K = (I, M, L, R, D, S, Q, E, H, C, P, N)$, while the obligation transport theorem shows that componentwise hsame and displayed Cont replay preserve that tuple for all named consumers.

The obligation ledger theorem exhausts the root-unblock rows. The midpoint formation theorem binds M to the displayed located and rational endpoint rows, and the child-cell admission theorem binds L, R to the dyadic, StreamName, RegSeqRat, and Real rows.

Finally the finite-net handoff theorem routes $\mathsf{ClosedIntervalCompactNet}^{\uparrow}$ consumers through the same finite mesh data. Together these clauses close the obligation surface without adding branch selection, selected subcovers, open-cover compactness, quotient real equality, trichotomy, or ambient compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalBisection}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalBisection}^{\uparrow}$ packet is generated from a displayed LocatedIntervalUp parent row, a RationalIntervalUp midpoint route, left and right located subinterval rows, a DyadicIntervalUp refinement ledger, StreamNameUp finite windows, RegSeqRatUp readbacks, RealUp child seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalBisection}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1465.1, Definition 1465.2, Definition 1465.3, Definition 1465.4, Definition 1465.5, Definition 1465.6, Definition 1465.7, Definition 1465.8, Definition 1465.9, Definition 1465.10, and Definition 1465.11 bind exactly the carrier, endpoint transport, nested-cell handoff, obligation carrier, hsame transport, Cont replay, Pkg provenance, and local NameCert rows for displayed located intervals and finite-net consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalBisection}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.IntervalBisectionUp.IntervalBisectionCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, trichotomy, quotient real equality, selected branch choice, selected subcover choice, or bridge checking.

Upgrade path: Public closure requires an exported bisection consumer theorem showing that every accepted handoff remains stable under repeated finite-net refinement for BEDC.Derived.IntervalBisectionUp.

Chapter 1466

A Concrete Naming Certificate for $\mathsf{SequentialContinuity}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{SequentialContinuity}^{\uparrow}$ packet records continuity along certified convergent sequences as a finite BEDC route. It reads the pointwise metric-continuity surface of chapter 59, the regular rational windows of chapter 301, the inspected stream schedules of chapter 126, the real-name seal of chapter 38, and the $\mathsf{Met}^{\uparrow}$ rows of chapter 57. The packet supplies a Real and Completion frontier surface for sequence consumers without importing host sequential continuity, function extensionality, choice, quotient equality, or ambient completeness.

Definition 1466.1 (Sequential continuity carrier)**.** *A* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$$

with the following displayed rows:

1. *M_X and M_Y are the source and target* $\mathsf{Met}^{\uparrow}$ *carrier rows with their visible classifiers and distance ledgers.*
2. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row for a displayed map from the source metric carrier to the target metric carrier.*
3. *S is the source sequence surface, exposed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *L is the inspected convergence ledger for S, listing the finite thresholds actually requested by the packet.*
5. *W_X and W_Y are the source and target finite stream windows read from S and from the displayed images of those windows under F.*
6. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows attached to the inspected source and target windows.*
7. *E_X and E_Y are the* $\mathsf{Real}^{\uparrow}$ *seal rows reached after the corresponding regular-rational handoffs have been displayed.*
8. *H stores componentwise* hsame *transport for the metric rows, the continuous-map row, the sequence row, the convergence ledger, both window families, both regular handoffs, both Real seals, continuation, provenance, and local naming.*

9. *C stores displayed* Cont *replay from S, L to W_X, from W_X, F to W_Y, from W_X to R_X, E_X, and from W_Y to R_Y, E_Y.*
10. *P is the* Pkg *provenance over* $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an external sequence function, quotient stream equality, function extensionality, countable choice, or host sequential-continuity theorem.

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityTasteGate single carrier alignment`

Theorem 1466.2 (Sequential continuity NameCert obligations)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed fifteen-row packet;*
2. *source and target metric-row stability through M_X, M_Y, H, C;*
3. *pointwise-continuity use only through the displayed* $\mathsf{Continuous}^{\uparrow}$ *row F;*
4. *inspected-threshold convergence coverage through the finite ledger L;*
5. *source and target sequence handoff through W_X, W_Y, R_X, R_Y before either* $\mathsf{Real}^{\uparrow}$ *seal is consumed;*
6. *non-escape from host sequential continuity, function extensionality, countable choice, quotient equality, and ambient Real completeness.*

Proof. Project the carrier of Definition 1466.1. The metric rows, continuous-map row, source sequence row, convergence ledger, window rows, regular-rational handoffs, Real seals, and structural rows are the only coordinates visible to the classifier.

The metric and continuity faces are already displayed. Source and target stability reads through M_X, M_Y, H, C, while pointwise continuity is consumed only through F and the finite windows named by the packet.

The sequential face is finite. The ledger L records the inspected convergence thresholds, W_X displays the corresponding source windows, and W_Y displays the image windows obtained by the recorded continuation route.

Finally read the endpoint route. The source side reaches E_X through R_X, and the target side reaches E_Y through R_Y. Since the excluded principles are not coordinates of Q, the six listed rows exhaust the local NameCert obligation surface. □

Theorem 1466.3 (Sequential continuity Cauchy handoff)**.** *Every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *packet hands both the source sequence windows and the image sequence windows to* $\mathsf{RegSeqRat}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Equivalently, every seal-facing route factors through*

$$S, L \longmapsto W_X \longmapsto R_X \longmapsto E_X \qquad \textit{and} \qquad W_X, F \longmapsto W_Y \longmapsto R_Y \longmapsto E_Y$$

together with M_X, M_Y, H, C, P, N.

Proof. Begin with the inspected source route. The continuation row records that W_X is read from the displayed sequence surface S using only the thresholds listed in L.

Next form the image route. The row F is the displayed $\mathsf{Continuous}^{\uparrow}$ pointwise-modulus surface, and C records the replay from the inspected source windows to the target windows W_Y.

Both sides then pass through regular-rational readback. The rows R_X and R_Y are the only regular handoffs attached to the source and target windows, and the seal rows E_X and E_Y are downstream from those handoffs.

Thus a consumer that tries to use either $\mathsf{Real}^{\uparrow}$ seal without the matching $\mathsf{RegSeqRat}^{\uparrow}$ handoff leaves the accepted $\mathsf{SequentialContinuity}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityCauchyHandoff`

Theorem 1466.4 (Sequential continuity metric-row stability)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier*

$$Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N),$$

the source and target metric rows used by the local classifier are exactly the displayed $\mathsf{Met}^{\uparrow}$ *rows* M_X *and* M_Y*, stabilized by the same componentwise* hsame *transport* H *and displayed* Cont *replay* C *carried by* Q*. No metric comparison may be supplied by an external topology, host equality of points, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.SequentialContinuityUp.SequentialContinuityPacket metric row stability`

Proof. Project the accepted carrier. The only metric coordinates visible in the packet are M_X and M_Y, and the only structural coordinates that can move them between accepted packets are H and C.

The provenance and naming rows do not add a second metric surface. They record that the displayed rows belong to the same $\mathsf{SequentialContinuity}^{\uparrow}$ packet over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Thus every metric read used by the classifier is stable exactly along M_X, M_Y, H, C, P, N. The excluded external topology, host equality, quotient, and completeness data are not coordinates of Q. □

Theorem 1466.5 (Sequential continuity pointwise consumption)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier* $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$*, the pointwise-continuity information consumed by the sequence route is only the displayed* $\mathsf{Continuous}^{\uparrow}$ *row* F *together with the source metric row* M_X*, target metric row* M_Y*, finite windows* W_X, W_Y*, transport* H*, replay* C*, provenance* P*, and local naming* N*. The packet exports no host sequential-continuity theorem and no function-extensionality coordinate.*

Proof. Read the pointwise face of the carrier. The row F is the unique $\mathsf{Continuous}^{\uparrow}$ coordinate, and it is stored beside the source and target metric rows M_X, M_Y.

The sequence route inspects only finite windows. The continuation row sends the displayed source windows W_X through F to the target windows W_Y, and the transport row preserves that same route under componentwise hsame.

Finally, P and N source and name the same packet. Since neither a host sequential-continuity theorem nor a function-extensionality principle appears among the coordinates of Q, every pointwise-continuity use is consumed through the displayed rows named in the statement. □

Theorem 1466.6 (Sequential continuity inspected-threshold coverage)**.** *For every accepted* $\mathsf{SequentialContinuity}^{\uparrow}$ *carrier* $Q = (M_X, M_Y, F, S, L, W_X, W_Y, R_X, R_Y, E_X, E_Y, H, C, P, N)$*, each threshold used by a Real-facing or regular-Cauchy consumer is first listed in the inspected convergence ledger* L*, then*

read through the source window W_X, the image window W_Y, the regular handoff rows R_X, R_Y, and only then through the seal rows E_X, E_Y. The coverage route is exhausted by the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the same carrier.*

Proof. Project the carrier and read the inspected ledger. The row L is the only coordinate that lists convergence thresholds requested by the packet.

The source schedule then opens W_X from S at the listed thresholds. The pointwise row F and replay row C produce the corresponding image windows W_Y without adding a detached sequence object.

The regular and Real-facing reads are downstream from those windows. The rows R_X, R_Y repack the inspected windows as regular rational handoffs, and E_X, E_Y are consumed only after those handoffs.

The structural rows H, C, P, N preserve, replay, source, and name exactly this route. Therefore the inspected thresholds cover the whole consumer path and no unlisted threshold, quotient stream, selected limit, choice row, or ambient completeness datum can enter the packet. □

The mature treatment of $\mathsf{SequentialContinuity}^{\uparrow}$ is the Real and Completion sequence-consumer route obtained by reading the Cauchy handoff, metric-row stability, pointwise consumption, and inspected-threshold coverage together. A consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal or a completion-facing sequence row must pass through the displayed finite windows, the regular-rational handoff rows, and the same metric and continuity coordinates already inspected by the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a displayed pointwise-continuity row, inspected sequence windows, a finite convergence ledger, source and target RegSeqRat handoffs, RealUp seal rows, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1466.1, Definition 1466.2, Definition 1466.3, Definition 1466.4, Definition 1466.5, and Definition 1466.6 bind the carrier, NameCert obligations, Cauchy handoff, metric-row stability, pointwise consumption, and inspected-threshold coverage for certified sequence consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialContinuity}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.SequentialContinuityUp.SequentialContinuityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not claim host sequential continuity, function extensionality, countable choice, quotient stream equality, semantic completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: No further paper-side closure grade remains; bridge verification stays on the independent formal axis.

Chapter 1467

A Concrete Naming Certificate for $\mathsf{SubsequenceLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SubsequenceLimit}^{\uparrow}$ packet records the finite consumer boundary for reading a displayed subsequence limit from already certified real sequence-limit data. It consumes the sequence-limit surface of chapter 1413, the cofinal selector route of chapter 405, the regular Cauchy subsequence boundary of chapter 381, and the real-name seal of chapter 38. Its purpose is to let downstream Bolzano–Weierstrass and sequence-convergence arguments read a selected limit surface without importing arbitrary subsequence choice, host sequence equality, quotient stream equality, or ambient completeness.

Definition 1467.1 (SubsequenceLimit carrier)**.** *A* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, \Phi, W, R, D, E, H, C, P, N)$$

with the following visible rows:

1. *L is the parent* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row, containing the displayed source sequence, candidate limit, tail-window schedule, dyadic tolerance ledger, and convergence classifier exposed by chapter 1413;*
2. *Φ is the accepted* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row, including the displayed unary selector, monotonicity witnesses, cofinal witnesses, selected windows, and regular rational handoff rows;*
3. *W records the selected finite* $\mathsf{StreamName}^{\uparrow}$ *windows from the parent sequence and from the subsequence readback surface;*
4. *R records the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *observations and the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *route through which the subsequence read is displayed;*
5. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the selected subsequence windows with the parent limit row;*
6. *E is the shared* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the parent limit row, cofinal selector row, selected windows, regular rational handoff, and dyadic tolerance ledger have been displayed;*
7. *H records componentwise* hsame *transports for the parent limit row, selector row, selected windows, regular rational observations, dyadic ledger, real seal, and local naming row;*
8. *C records the displayed* Cont *routes by which consumers inspect the same rows in their certificate order;*

9. *P is the* Pkg *provenance over the* $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and naming dependencies;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$ *row.*

The carrier stores a finite subsequence-limit consumer interface. It contains no host-level infinite subsequence, no quotient stream representative, no host sequence equality, no independent real-completeness object, and no choice principle for selecting further subsequences.

Theorem 1467.2 (SubsequenceLimit NameCert obligations)**.** *Every accepted* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier of Definition 1467.1 carries five finite NameCert obligation rows:*

1. *carrier admission for the parent sequence-limit row, cofinal selector row, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff rows, dyadic tolerance ledger, shared* $\mathsf{Real}^{\uparrow}$ *seal, transport rows, continuation routes, package provenance, and local naming row;*

2. *selector-to-limit alignment, stating that every subsequence window read by Φ is one of the displayed parent sequence windows named by L and compared against the same candidate limit row;*

3. *tolerance preservation, stating that the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D is the common tolerance ledger used by both the parent sequence-limit classifier and the selected subsequence read;*

4. *real-seal handoff discipline, stating that the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed only after the parent limit row, cofinal selector, selected windows, regular rational route, and tolerance ledger have been displayed;*

5. *non-escape, stating that the packet exports no Bolzano–Weierstrass conclusion, arbitrary subsequence choice, host sequence equality, quotient stream equality, ambient real completeness, or bridge-checked external schema.*

Lean4 ✓ `BEDC.Derived.SubsequenceLimitUp.SubsequenceLimitTasteGate single carrier alignment`

Proof. Carrier admission is the tuple in Definition 1467.1. Each row needed by a consumer is a visible BHist coordinate, and the local naming row is attached to that same finite packet.

Selector-to-limit alignment is read from the pair (L, Φ). The parent row supplies the displayed sequence-limit surface, while the selector row supplies only selected finite windows and their regular rational handoff data. Thus a subsequence read remains a read of the parent displayed sequence.

Tolerance preservation is the projection to D. The selected subsequence windows are compared with the parent candidate limit using the same dyadic tolerance ledger, so the packet has no second tolerance scale that could change the limit claim.

The real-seal handoff follows the order of the rows L, Φ, W, R, D, E. A consumer reaches the shared $\mathsf{Real}^{\uparrow}$ seal only after the parent limit data, selector route, selected windows, regular rational observations, and dyadic ledger are already visible.

The non-escape row is an inventory check. None of the coordinates stores a Bolzano–Weierstrass selection, a host-level subsequence, host sequence equality, quotient stream equality, ambient real completeness, or an external bridge schema, so the local NameCert cannot export those objects. □

Theorem 1467.3 (SubsequenceLimit Real seal handoff)**.** *Let $Q = (L, \Phi, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer read from Q factors through the parent* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row L, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row Φ, the selected* $\mathsf{StreamName}^{\uparrow}$

windows W, the selected $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *route R, and the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the seal row E is exposed.*

Proof. Start from a $\mathsf{Real}^{\uparrow}$-facing consumer read. The carrier places the seal coordinate E after the parent sequence-limit row, selector row, selected windows, regular rational handoff, and dyadic ledger.

Apply the selector-to-limit and tolerance-preservation rows of Definition 1467.2. They force the selected subsequence windows to remain parent sequence windows and force their limit comparison to use the same dyadic tolerance ledger as the parent limit classifier.

It remains to identify the public read route. The displayed hsame transports and Cont routes preserve the order through L, Φ, W, R, D, E, while the Pkg and local NameCert rows account for provenance. Hence every seal-facing read factors through the listed finite handoff surface. □

Theorem 1467.4 (SubsequenceLimit choice-free boundary)**.** *The* $\mathsf{SubsequenceLimit}^{\uparrow}$ *carrier is a finite BEDC consumer boundary for selected sequence-limit reads. It preserves the parent sequence-limit row, cofinal selector row, selected regular rational observations, dyadic tolerance ledger, shared real seal, displayed transports, continuation routes, package provenance, and local naming row, but it does not construct a Bolzano–Weierstrass subsequence, choose an arbitrary infinite subsequence, identify host sequences, identify quotient streams, prove real completeness, or provide a checked standard bridge.*

Proof. The positive content is exactly the carrier inventory of Definition 1467.1. The rows $L, \Phi, W, R, D, E, H, C, P$, provide the parent limit data, selector route, selected observations, tolerance ledger, seal, transport, continuation, provenance, and local naming surface.

The consumer route is fixed by Definition 1467.2 and Definition 1467.3. Every public real-facing read passes through the finite selector-to-limit handoff rather than through an unrecorded host object.

The excluded principles are absent as coordinates. Since no row stores a Bolzano–Weierstrass construction, arbitrary subsequence choice, host equality, quotient equality, completeness proof, or bridge-checked schema, the packet cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{SubsequenceLimit}^{\uparrow}$ packet is generated from a parent sequence-limit row, a cofinal selector row, selected StreamName windows, RegSeqRat and RegularCauchySubsequence handoff rows, a shared DyadicRatCore tolerance ledger, a shared RealUp seal, componentwise transport, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$, seed)

Scope closed: Definition 1467.1, Definition 1467.2, Definition 1467.3, and Definition 1467.4 declare the finite subsequence-limit consumer boundary for selected RealUp sequence-limit reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubsequenceLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Bolzano–Weierstrass, arbitrary subsequence choice, host sequence equality, quotient stream equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector-to-limit alignment, tolerance preservation, real-seal handoff, and choice-free refusal for the same finite packet.

Chapter 1468

A Concrete Naming Certificate for $\mathsf{AdjointOperator}^{\uparrow}$

Following the Hilbert-space adjoint construction in standard functional analysis, the $\mathsf{AdjointOperator}^{\uparrow}$ packet records the finite BEDC surface of a bounded operator together with a displayed adjoint endpoint and the inner-product pairing law that characterizes it. It is read over the $\mathsf{Hilbert}^{\uparrow}$ source of chapter 91, the $\mathsf{InnerProduct}^{\uparrow}$ form rows of chapter 86, the bounded-linear-operator carrier of Definition 90.49, and the functional-analysis consumer frontier of chapter 94.

1468.1 Carrier packet

Definition 1468.1 (Adjoint operator carrier)**.** *For carried* $\mathsf{Hilbert}^{\uparrow}$ *sources* H *and* K*, an* $\mathsf{AdjointOperator}^{\uparrow}(H, K)$ *carrier is a finite* BHist *packet*

$$A = (H, K, T, T^*, B, B^*, \langle -, - \rangle_H, \langle -, - \rangle_K, J, L, P, N)$$

with the following displayed rows.

1. *H and K are the Hilbert source rows, including their inherited* $\mathsf{InnerProduct}^{\uparrow}$ *form rows and Banach normed-complete rows.*
2. *T is a carried bounded-linear-operator row from H to K, read through Definition 90.49.*
3. *T^* is a carried bounded-linear-operator row from K to H, with bound row B^* and its own visible operator ledger.*
4. *$\langle -, - \rangle_H$ and $\langle -, - \rangle_K$ are the two displayed inner-product form rows consumed from the* $\mathsf{InnerProduct}^{\uparrow}$ *surfaces of H and K.*
5. *J is the finite adjunction ledger stating the componentwise pairing comparison between applying T on the H side and applying T^* on the K side.*
6. *L records the* hsame *transports of Hilbert source rows, operator endpoints, bound endpoints, inner-product values, and adjunction comparisons.*
7. *P is the local* Pkg *provenance surface, and N is the local* $\mathsf{NameCert}_{\mathsf{AdjointOperator}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host Hilbert space, a choice-selected representing vector, a spectral resolution, a polar decomposition, or a full Riesz representation theorem.

Lean4 ✓ `BEDC.Derived.AdjointOperatorUp.AdjointOperatorTasteGate single carrier alignment`

1468.2 Adjunction law

Theorem 1468.2 (Adjoint operator inner-product adjunction)**.** *Let* $A = (H, K, T, T^*, B, B^*, \langle -, - \rangle_H, \langle -, - \rangle_K, J,$ *be an accepted* $\mathsf{AdjointOperator}^{\uparrow}(H, K)$ *packet. For every carried* H*-endpoint* x *and carried* K*-endpoint* y *visible to the operator ledgers, the packet exposes the classified scalar comparison*

$$\langle Tx, y \rangle_K \sim_{\mathsf{Real}^{\uparrow}} \langle x, T^* y \rangle_H.$$

This comparison is part of the finite ledger J*; it is not obtained by appealing to an ambient Hilbert-space duality theorem, a choice principle, or an external operator equality.*

Proof. Project the accepted packet through Definition 1468.1. The source and target Hilbert rows give the two inner-product forms, while the bounded-operator rows for T and T^* give the endpoint values Tx and T^*y in the carried Hilbert carriers.

Read the adjunction ledger J at the displayed pair (x, y). Its scalar comparison is already typed over the $\mathsf{Real}^{\uparrow}$ classifier inherited by the two inner-product rows, so the asserted equality is a BEDC endpoint comparison rather than host-level equality of scalar values.

The transport ledger L keeps the same comparison stable when x, y, T, T^*, or either Hilbert source row is replaced by a classified row. Hence the public adjunction law is exactly the row displayed by the packet. □

1468.3 Name certificate surface

Theorem 1468.3 (Adjoint operator NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AdjointOperator}^{\uparrow}}$ *surface consists of the following five rows.*

1. *Carrier habitation for the finite packet of Definition 1468.1.*
2. *Componentwise classification of the two Hilbert source rows, the two bounded-operator rows, the two bound rows, and the two inner-product form rows.*
3. *Inner-product adjunction exactness through the ledger* J *of Definition 1468.2.*
4. *Transport stability for source, target, operator, bound, form-value, and adjunction rows through* L*.*
5. *Ledger exclusion of Riesz representation, spectral theorem, polar decomposition, host Hilbert equality, and external operator equality.*

Proof. Carrier habitation is the finite packaging operation of Definition 1468.1. The Hilbert, inner-product, bounded-operator, bound, adjunction, transport, provenance, and local-name rows are visible coordinates of the packet.

The classifier row is componentwise. Hilbert source comparison is read through the $\mathsf{Hilbert}^{\uparrow}$ classifiers, bounded-operator comparison through the Banach bounded-operator surface, and scalar comparison through the $\mathsf{Real}^{\uparrow}$ classifiers used by the two $\mathsf{InnerProduct}^{\uparrow}$ form rows.

Adjunction exactness is the assertion of Definition 1468.2. Transport stability is supplied by L, which repacks classified replacements of the same rows without introducing any host duality or spectral data. The excluded analytic theorems therefore remain outside the local certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{AdjointOperator}^{\uparrow}}$**.**

Constructive story: An AdjointOperator$^{\uparrow}$ packet records two carried Hilbert source rows, a carried bounded operator T, a carried bounded operator T^* in the reverse direction, their bound ledgers, the two inner-product form rows, a finite adjunction ledger comparing $\langle Tx, y\rangle_K$ with $\langle x, T^*y\rangle_H$, componentwise transport rows, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{AdjointOperator}^{\uparrow}}$, seed)

Scope closed: Definition 1468.1, Definition 1468.2, and Definition 1468.3 seed the Hilbert bounded-operator adjoint boundary over Hilbert$^{\uparrow}$, InnerProduct$^{\uparrow}$, Banach$^{\uparrow}$, Real$^{\uparrow}$, and FunctionalAnalysis$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{AdjointOperator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Riesz representation, the spectral theorem, polar decomposition, host Hilbert-space duality, arbitrary existence of adjoints, Lean verification, or a standard bridge.

Upgrade path: A formal route can encode the finite carrier and prove the adjunction ledger, transport stability, and local NameCert obligation surface.

Chapter 1469

A Concrete Naming Certificate for $\mathsf{BishopCompletionModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionModulus}^{\uparrow}$ packet names the finite completion-modulus route that turns a displayed regular rational Cauchy input into a Real-facing seal. It refines the modulus surface of chapter 292, reads finite observation windows through chapter 126, uses the regular rational handoff of chapter 301, records dyadic thresholds through chapter 291, and reaches the terminal seal of chapter 38. The packet supplies an L10 completion-modulus object without quotient streams, selected limits, countable choice, or ambient completeness.

Definition 1469.1 (Bishop completion modulus carrier)**.** *A* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, n, k, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, read as a monotone finite modulus interface for rational Cauchy observations.*
2. *S is the underlying finite* $\mathsf{StreamName}^{\uparrow}$ *schedule whose requested windows are admitted by the modulus row.*
3. *n is the unary precision request and k is the displayed threshold row selected by M for that request.*
4. *W is the finite stream window of S read at and beyond the threshold k.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger containing the dyadic tolerance, endpoint, comparison, and scale rows used by the selected window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by repacking W with the dyadic threshold ledger D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, n, k, W, D, R have been displayed.*
8. *H stores componentwise* hsame *transport for the modulus row, stream schedule, request, threshold, window, dyadic ledger, regular handoff, real seal, continuation, provenance, and local naming.*
9. *C stores displayed* Cont *replay from the request through the modulus row to the window, from the window through the dyadic ledger to the regular handoff, and from the handoff to the real seal.*

10. *P is the* Pkg *provenance over* BishopCompletionModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for quotient stream equality, a selected completed limit, countable choice, a host real value, or an ambient completeness principle.

Lean4 (definition) `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier`
Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier rows exposed`

Theorem 1469.2 (Bishop completion modulus NameCert obligations)**.** *For every accepted* BishopCompletionModu *carrier $Q = (M, S, n, k, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionModulus}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the displayed twelve-row packet;*
2. *monotone finite modulus readback, stating that the threshold k is read from the displayed* CauchyModulus$^{\uparrow}$ *row M at the precision request n and is preserved by the stored transport rows;*
3. *dyadic threshold compatibility, stating that the window W and ledger D use the same threshold and tolerance scale;*
4. RegSeqRat$^{\uparrow}$ *handoff, stating that R is obtained only from the displayed stream window and dyadic threshold ledger;*
5. Real$^{\uparrow}$ *seal non-escape, stating that the seal row E is reachable only after the regular handoff and exports no quotient stream, selected limit, countable-choice row, host real, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1469.1. The displayed modulus, stream schedule, request, threshold, window, dyadic ledger, regular handoff, real seal, transport, continuation, provenance, and local naming rows are the only classifier-visible coordinates.

The modulus readback is finite. The row M supplies the threshold k for the request n, and the transport rows in H preserve that same threshold when the packet is compared with another accepted carrier.

The dyadic and regular faces are forced to use the same source data. The ledger D records the tolerance scale for the window W, and the handoff row R is read from W together with D rather than from a hidden rational sequence.

Finally the real-facing route is downstream from R. Since the carrier has no coordinate for quotient stream equality, a selected completed limit, countable choice, a host real value, or ambient completeness, the five listed obligations exhaust the local NameCert surface. □

Theorem 1469.3 (Bishop completion modulus L10 handoff)**.** *Every accepted* BishopCompletionModulus$^{\uparrow}$ *carrier hands its completion-modulus data to the L10 real-completion route through*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E$$

together with the displayed StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. No consumer of the packet can reach a* Real$^{\uparrow}$ *seal without first reading the monotone modulus threshold, the finite stream window, the dyadic threshold ledger, and the regular rational handoff.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier l10 handoff`

Proof. Begin with an accepted carrier and read the modulus face. The request n is sent through the displayed $\mathsf{CauchyModulus}^{\uparrow}$ row M, producing the threshold row k named in the carrier.

Next read the finite observation face. The stream schedule S supplies the window W at the selected threshold, while D records the dyadic tolerance and comparison rows attached to that same window.

The regular handoff is then fixed by the packet. The row R repacks the displayed window and dyadic ledger as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the continuation routes record no alternate path to the seal.

Finally the $\mathsf{Real}^{\uparrow}$ row E is read only after R. The non-escape clause of Definition 1469.2 excludes quotient streams, selected limits, countable choice, host real values, and ambient completeness from the exported L10 handoff. □

Theorem 1469.4 (Bishop completion modulus threshold coherence)**.** *For every accepted* BishopCompletionModulus *carrier $Q = (M, S, n, k, W, D, R, E, H, C, P, N)$, monotone threshold coherence is a local obligation on the route*

$$M, n \longmapsto k \longmapsto W, D.$$

The $\mathsf{CauchyModulus}^{\uparrow}$ *row M supplies the displayed threshold k for the precision request n, the* $\mathsf{StreamName}^{\uparrow}$ *row S reads the finite window W at that threshold, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D records the same threshold and tolerance scale. Componentwise* hsame *transport and displayed* Cont *replay may preserve this route, but they do not select limits or assert* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier threshold coherence`

Proof. Project the carrier and read the modulus face. The rows M, n, k are displayed before the finite window row, so the threshold used by the packet is the one selected from the visible $\mathsf{CauchyModulus}^{\uparrow}$ row.

Read the window and dyadic ledger next. The row W is the finite $\mathsf{StreamName}^{\uparrow}$ observation at the selected threshold, and D records the dyadic tolerance data for that same window.

The structural rows preserve coherence only componentwise. The rows H, C, P, N transport, replay, source, and name the displayed threshold route, and no row in the carrier supplies a selected completed limit or an ambient completeness theorem. □

Theorem 1469.5 (Bishop completion modulus finite threshold induction)**.** *For every accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier $Q = (M, S, n, k, W, D, R, E, H, C, P, N)$, every finite precision descent from n through the displayed monotone modulus row M preserves a single threshold route*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E.$$

Induction on the unary precision history may refine the requested threshold only through the displayed $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. It cannot introduce a second threshold source, quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness.*

Proof. Project the carrier of Definition 1469.1. The only rows capable of answering a precision request are the displayed modulus row M, the unary request n, the threshold k, the finite window W, the dyadic ledger D, the regular readback R, and the Real seal E.

Induct on the unary precision history. In the base case, the request is the displayed n, and Definition 1469.4 identifies the threshold route through M, n, k, W, D before any readback or seal is consumed.

For the successor step, the monotone modulus row may refine the requested threshold, but the refinement remains inside the same carrier route. Apply Definition 1469.4 again to keep the successor request attached to the displayed StreamName window and DyadicRatCore ledger rather than to a second threshold oracle.

Finally close the non-escape part by Definition 1469.7. Since every Real-facing read reaches E only after R, and R is downstream from W, D, the induction cannot export quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier finite threshold induction`

Theorem 1469.6 (Bishop completion modulus threshold heredity)**.** *Let $Q = (M, S, n, k, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier. For any displayed successor precision request refined through the same* $\mathsf{CauchyModulus}^{\uparrow}$ *row M, the completion route remains inside the same finite path*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E$$

after replacing the request and threshold by their displayed successor refinements. In particular, the refined read still uses the same $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal rows, transported and replayed only through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier threshold heredity`

Proof. Start with threshold coherence from Definition 1469.4. It identifies the selected threshold as a row of the displayed $\mathsf{CauchyModulus}^{\uparrow}$ carrier face and attaches the resulting stream window and dyadic ledger to the same request route.

Apply finite threshold induction from Definition 1469.5 to the successor precision history. The induction step permits threshold refinement only by another read of M, so the refined threshold remains tied to the same $\mathsf{StreamName}^{\uparrow}$ schedule and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than to a second threshold source.

Finally use Real seal ordering from Definition 1469.7. The refined route reaches E only after the corresponding W, D rows and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R have been displayed, and the structural rows H, C, P, N preserve that route without adding quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 1469.7 (Bishop completion modulus Real seal ordering)**.** *For every accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier $Q = (M, S, n, k, W, D, R, E, H, C, P, N)$, the* $\mathsf{Real}^{\uparrow}$ *seal row E is ordered strictly after the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R. A consumer may reach E only through the displayed sequence from* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold selection to* $\mathsf{StreamName}^{\uparrow}$ *window readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff; the seal does not supply selected limits, quotient stream equality, host real equality, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier real seal ordering`

Proof. Start from Definition 1469.3. It displays the only route to the Real-facing row:

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E.$$

The regular handoff is therefore prior to the seal. The row R repacks the finite window W and dyadic threshold ledger D before any $\mathsf{Real}^{\uparrow}$ row is available to a consumer.

Finally apply the non-escape obligation from Definition 1469.2. Since the carrier contains no row for quotient stream equality, selected completed limits, choice, host real equality, or completeness, the ordered seal cannot export any of those principles. □

Theorem 1469.8 (Bishop completion modulus obligation package)**.** *For the carrier of Definition 1469.1, the displayed carrier rows, NameCert obligations, L10 handoff, threshold coherence, finite threshold induction, threshold heredity, and Real seal ordering together discharge the local* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *obligations for carrier admission, monotone finite modulus readback, dyadic threshold compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal non-escape.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier obligation package`

Proof. Carrier admission and the local obligation list are exactly the content of Definition 1469.2. They expose the twelve-row packet and name the five rows that every accepted carrier must satisfy.

The handoff and threshold clauses are supplied by Definition 1469.3, Definition 1469.4, Definition 1469.5, and Definition 1469.6. These results keep the precision request, selected threshold, stream window, and dyadic ledger on one finite route.

Finally apply Definition 1469.7. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff precedes the $\mathsf{Real}^{\uparrow}$ seal, and the seal exports none of the excluded quotient, choice, host-equality, selected-limit, or completeness data. Thus the five local obligations are packaged by the displayed chapter surface. □

Theorem 1469.9 (Bishop completion modulus same-threshold transport)**.** *Let* $Q = (M, S, n, k, W, D, R, E, H, C, P$ *and* $Q' = (M', S', n', k', W', D', R', E', H', C', P', N')$ *be accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carriers related by the displayed classifier rows. If the relation is witnessed through the componentwise transport rows* H*, the continuation replay rows* C*, package provenance* P*, and local naming* N*, then the compared packets preserve one displayed threshold route:*

$$M, n, k, W, D, R, E \qquad \text{and} \qquad M', n', k', W', D', R', E'.$$

The $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal are transported as matching rows of the same route. No comparison may replace that route by quotient stream equality, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier same threshold transport`

Proof. Project both accepted carriers. Their classifier-readable rows are exactly the completion modulus, stream schedule, precision request, threshold, window, dyadic ledger, regular handoff, Real seal, transport, continuation, provenance, and local naming rows of Definition 1469.1.

Use threshold coherence on each side. Theorems 1469.4 and 1469.6 keep the $\mathsf{CauchyModulus}^{\uparrow}$ threshold attached to the displayed $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

Now read the transport and replay rows. The componentwise hsame entries in H and the displayed Cont rows in C compare only the named carrier coordinates, while P and N repack the same local source and naming surface. Thus the transported packet still exposes the same threshold route.

Finally apply Real seal ordering, Definition 1469.7. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains before the $\mathsf{Real}^{\uparrow}$ seal on both sides, so the comparison cannot introduce quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 1469.10 (Bishop completion modulus scoped dependency surface)**.** *The* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *chapter has scoped BEDC closure for the finite completion-modulus dependency surface generated by the carrier of Definition 1469.1. The closed scope consists of carrier admission, the five NameCert obligations, the L10 handoff, threshold coherence, finite threshold induction, threshold heredity, Real seal ordering, and same-threshold transport through* H, C, P, N*.*

Within that scope every accepted consumer reads the same displayed sequence

$$\mathsf{CauchyModulus}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before it reaches a completion-modulus seal. The scope does not include quotient stream equality, selected completed limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier scope closure`

Proof. Collect the carrier and obligation rows from Definition 1469.1 and Definition 1469.2. These fix the finite packet and the local NameCert surface.

Apply the route theorems. Theorems 1469.3, 1469.4, 1469.5, and 1469.6 bind every precision read to the same threshold, window, dyadic ledger, and regular rational handoff.

Use Definition 1469.7 for the terminal row and Definition 1469.9 for classifier-related packets. These results keep the Real seal and all transport rows inside the displayed carrier.

The excluded principles have no carrier coordinate and no scoped route theorem. Therefore the listed rows form the scoped BEDC dependency surface for $\mathsf{BishopCompletionModulus}^{\uparrow}$. □

Theorem 1469.11 (Bishop completion modulus public completion certificate)**.** *Every public completion-modulus consumer of an accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier* $Q = (M, S, n, k, W, D, R, E, H, C,$ *receives one certificate surface consisting of the carrier rows of Definition 1469.1, finite threshold induction, threshold heredity, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the ordered* $\mathsf{Real}^{\uparrow}$ *seal. The public read is the finite route*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E$$

preserved by H, C, P, N*. It exports no quotient stream, selected limit, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier public completion certificate`

Proof. Start with the carrier and obligation surface. Theorems 1469.2 and 1469.3 expose the twelve carrier rows and force every completion read through the displayed modulus, finite stream window, dyadic threshold ledger, regular handoff, and Real seal.

Next apply finite threshold induction and threshold heredity. Theorems 1469.5 and 1469.6 keep successor precision requests attached to the same $\mathsf{CauchyModulus}^{\uparrow}$ row, the same $\mathsf{StreamName}^{\uparrow}$ schedule, and the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than to a second threshold source.

Use Definition 1469.7 for the terminal ordering and Definition 1469.9 for public transport. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains prior to the $\mathsf{Real}^{\uparrow}$ seal, and componentwise transport preserves the displayed route.

Finally package the result by Definition 1469.10. The public certificate is exactly the scoped finite route above, so no quotient stream, selected completed limit, countable choice, host real equality, or ambient completeness can be exported. □

Theorem 1469.12 (Bishop completion modulus bridge-facing Real-seal boundary)**.** *Every bridge-facing read of an accepted* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *carrier preserves the displayed route*

$$M, n \longmapsto k \longmapsto W, D \longmapsto R \longmapsto E$$

together with H, C, P, N*. A standard interpretation may inspect this completion-modulus handoff and its terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it may not add quotient streams, selected limits, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionModulusUp.BishopCompletionModulusCarrier bridge facing real seal boundary`

Proof. Start from the public completion certificate, Definition 1469.11. It fixes the public route through the modulus request, threshold, window, dyadic ledger, regular rational handoff, and Real seal.

Next use the same-threshold transport theorem, Definition 1469.9. The rows H, C, P, N preserve the displayed route componentwise and replay only the named carrier data, so a bridge-facing read cannot change the threshold source or insert an alternate stream.

Finally apply Real seal ordering from Definition 1469.7. Since E is reachable only after R and the carrier has no quotient-stream, selected-limit, choice, host-equality, or ambient-completeness coordinate, the standard-facing boundary is exactly the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionModulus}^{\uparrow}$ packet is generated from a displayed CauchyModulus row, a finite StreamName schedule, a unary precision request, a selected threshold, a finite stream window, a DyadicRatCore threshold ledger, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionModulus}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1469.11 and Definition 1469.12 bind the public and bridge-facing export surface for the finite completion-modulus route through carrier rows, threshold induction, threshold heredity, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the ordered $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected completed limits, countable choice, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Only Lean formal verification and bridge checking remain outside the paper mature treatment of completion-modulus consumers.

Chapter 1470

RegularCauchyRing NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyRing}^{\uparrow}$ packet treats the algebra of regular Cauchy real names as a finite operation surface over displayed rational observations. Addition, negation, product, and rational scaling are not imported as host real operations; they are read through the existing $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, and $\mathsf{RegularCauchyScale}^{\uparrow}$ rows, then sealed through the same $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ completion boundary. Thus the packet records the L10 algebraic handoff without quotient real equality, trichotomy, countable choice, selected limits, or ambient completeness.

The sibling route runs through chapter 417, chapter 459, chapter 461, chapter 465, chapter 301, and chapter 38. Those chapters supply the operation rows and terminal $\mathsf{Real}^{\uparrow}$ seal consumed by the ring packet.

Definition 1470.1 (Regular Cauchy ring carrier)**.** *A* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W_A, W_B, D_A, D_B, S, G, M, L, R_S, R_G, R_M, R_L, E_S, E_G, E_M, E_L, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for regular rational Cauchy real names.*
2. *W_A and W_B are aligned* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the same requested precision scale.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers read from those windows.*
4. *S is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *operation row on A and B.*
5. *G is the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *operation row on the source row selected for additive inverse readback.*
6. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *operation row on A and B.*
7. *L is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *operation row for a displayed rational scalar and the selected source row.*
8. *R_S, R_G, R_M, R_L are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached respectively after the sum, negation, product, and scale readbacks.*
9. *E_S, E_G, E_M, E_L are the finite dyadic and error-budget ledgers consumed by those operation rows before the terminal seals are public.*

10. *H stores componentwise* hsame *transport for the two sources, aligned windows, dyadic ledgers, four operation rows, four terminal seal rows, and four finite budget ledgers.*

11. *C stores displayed* Cont *replay for opening the source windows, reading dyadic endpoint data, invoking the four operation packets, and exposing the terminal real-name seals.*

12. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *row.*

The carrier is a finite algebraic operation packet over regular Cauchy real names. It stores no quotient stream, no host real equality, no selected completed limit, no trichotomy, no countable-choice row, and no $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingTasteGate single carrier alignment`

Theorem 1470.2 (Regular Cauchy ring NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *surface for the carrier of Definition 1470.1 consists of the following finite obligations:*

1. *carrier admission for the displayed packet K with two regular-rational source rows, aligned stream windows, dyadic endpoint ledgers, four operation rows, four terminal real-name seal rows, four finite budget ledgers, transport, continuation, provenance, and local naming data;*

2. *source-window alignment, requiring S, M, and the selected-source rows for G and L to read only the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers;*

3. *operation admission, requiring the sum, negation, product, and scale rows to be accepted packets of their cited sibling surfaces;*

4. *componentwise classifier transport through the stored* hsame *rows and displayed* Cont *replay;*

5. *terminal seal handoff, requiring every real-facing consumer of the ring packet to pass through the operation readback row and its displayed budget ledger before reading one of R_S, R_G, R_M, R_L.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1470.1. The two regular-rational sources, aligned stream windows, dyadic endpoint rows, four operation rows, budget ledgers, terminal seals, transports, continuations, provenance, and local name are explicit coordinates of K.

The source-window obligation is inherited from the sibling operation packets. Apply the window and handoff surfaces of Definition 417.2, Definition 459.2, Definition 461.2, and Definition 465.2. Each cited surface consumes the displayed source rows and their finite dyadic ledgers before exporting a regular readback.

Classifier transport is componentwise. The stored H and C rows transport the two sources, the aligned windows, the dyadic ledgers, the four operation rows, and the four seal rows together, so no operation can detach from the finite packet.

Finally the terminal handoff is exactly the real-seal boundary supplied by the four sibling chapters. The local $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ row therefore names the assembled algebraic surface and adds no host real arithmetic, quotient equality, selected limit, trichotomy, choice, or completeness coordinate. □

Theorem 1470.3 (Regular Cauchy ring operation-row admission)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier*

$$K = (A, B, W_A, W_B, D_A, D_B, S, G, M, L, R_S, R_G, R_M, R_L, E_S, E_G, E_M, E_L, H, C, P, N),$$

the four operation coordinates S, G, M, L *are admitted only as displayed sibling rows of* $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, *and* $\mathsf{RegularCauchyScale}^{\uparrow}$. *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *row cannot replace any of those rows by host real addition, host real negation, host real multiplication, scalar multiplication on completed quotients, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1470.1. The only operation coordinates are the four displayed rows S, G, M, L, and each is listed beside the same source rows, stream windows, dyadic endpoint ledgers, budget ledgers, terminal seal rows, transport, replay, provenance, and local naming.

Use the sibling obligation surfaces for the four rows: Definition 417.2, Definition 459.2, Definition 461.2, and Definition 465.2. These chapters admit the sum, negation, product, and scale reads as finite operation packets over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data.

Finally read the local NameCert obligation theorem Definition 1470.2. It packages those sibling rows into one ring carrier but adds no fifth operation coordinate and no host arithmetic row, so admission of the operation rows is exactly sibling-row admission. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier operation row admission`

Theorem 1470.4 (Regular Cauchy ring componentwise transport)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier* K*, classifier transport through* H *preserves the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers, four operation rows, four terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, four finite budget ledgers, displayed replay, provenance, and local naming component by component. A transported ring packet therefore reads the same finite operation surface through* S, G, M, L *and cannot detach an operation row from its displayed source, budget, or seal data.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier componentwise transport`

Proof. Begin with the carrier coordinates of Definition 1470.1. The transport row H is explicitly componentwise over the two sources, the aligned windows, the dyadic ledgers, the four operation rows, the four terminal seal rows, the four finite budget ledgers, replay, provenance, and local naming.

Apply the operation-row admission theorem Definition 1470.3. Since S, G, M, L are admitted only as displayed sibling rows, transporting a packet transports those rows together with the source windows and dyadic ledgers that make the sibling reads meaningful.

The displayed replay row C and provenance row P preserve the same route for consumers. Thus transport can rename or replay the finite packet, but it cannot split an operation from its $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, budget, or $\mathsf{Real}^{\uparrow}$ seal boundary. □

Theorem 1470.5 (Regular Cauchy ring terminal-seal discipline)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier is one of* R_S, R_G, R_M, R_L*, and it is reachable only after the corresponding operation row* S*,* G*,* M*, or* L *and its finite budget ledger* E_S*,* E_G*,* E_M*, or* E_L *have been displayed. No* $\mathsf{Real}^{\uparrow}$ *consumer of the ring packet can bypass the sibling operation chapters, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, componentwise transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier terminal seal discipline`

Proof. Project the terminal coordinates of the carrier. The only $\mathsf{Real}^{\uparrow}$ seal rows present in K are R_S, R_G, R_M, R_L, paired respectively with S, G, M, L and with the finite budget ledgers E_S, E_G, E_M, E_L.

Use the sibling seal boundaries. The sum route is governed by Definition 417.8; the negation route by Definition 459.8; the product route by Definition 461.5; and the scale route by Definition 465.11. Each route exposes finite windows, regular readback, dyadic data, and budget rows before the terminal seal is public.

Combine those sibling boundaries with componentwise transport from Definition 1470.4. The structural rows preserve the four operation-to-seal routes and add no extra seal coordinate, so every real-facing consumer remains inside one of the four displayed terminal routes. □

Theorem 1470.6 (Regular Cauchy ring operation closure)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier K, the four displayed algebraic operations close inside the finite regular-Cauchy naming surface: addition exposes S and R_S, negation exposes G and R_G, product exposes M and R_M, and rational scaling exposes L and R_L. Each operation reads only the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, finite budget rows, componentwise* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingOperationClosure`

Proof. Read the sum component first. The carrier exposes the two source rows, aligned windows, dyadic endpoint ledgers, the finite sum budget row E_S, and the operation row S; Definition 417.6 and Definition 417.8 then route the readback to R_S.

Read the unary component next. The negation row G uses its selected source window and dyadic ledger, and Definition 459.4 with Definition 459.8 routes the sign-flipped regular readback to R_G.

For multiplication, project M together with E_M. The product handoff and seal boundary of Definition 461.4 and Definition 461.5 keep the product inside the displayed dyadic product and budget rows before R_M is public.

For rational scaling, project L together with E_L. The scale handoff and seal boundary of Definition 465.10 and Definition 465.11 keep the scaled output inside the displayed scalar, source, window, dyadic, and budget rows before R_L is public.

These four projections exhaust the carrier's algebraic coordinates. The operation closure is therefore closure of the finite BEDC packet surface, not a statement about quotient real equality or ambient real algebra. □

Theorem 1470.7 (Regular Cauchy ring windowwise associativity)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier, the displayed sum, product, and rational-scale windows compose through the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers under reassociation. More precisely, each reassociated route first reads the two source rows, aligned windows, dyadic endpoint ledgers, finite budget rows, operation rows, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, and then reaches the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary for the corresponding sum, product, or scale readback.*

Proof. Project the carrier of Definition 1470.1. The only operation coordinates are S, G, M, L together with the aligned source windows, dyadic ledgers, budget rows, transports, replay, provenance, and terminal seal rows.

For sums, reassociation changes only the order in which the finite $\mathsf{RegularCauchySum}^{\uparrow}$ rows are consumed. The source-window alignment and terminal seal handoff of Definition 1470.2 keep both reads on the same displayed $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers before R_S is public.

For products and rational scales, use the same argument with the product and scale routes. The product route is governed by Definition 461.4 and Definition 461.5; the scale route is governed by Definition 465.10 and Definition 465.11. These sibling handoffs expose finite readbacks and then the terminal $\mathsf{Real}^{\uparrow}$ seal, so reassociation cannot introduce host real equality, selected limits, trichotomy, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier windowwise associativity`

Theorem 1470.8 (Regular Cauchy ring componentwise distributivity)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier, a product-over-sum read is witnessed componentwise through the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *and* $\mathsf{RegularCauchySum}^{\uparrow}$ *budget rows. The left route reads the product row after the summed source row is displayed; the right route reads two product rows and then the sum row; both routes remain inside the displayed source windows, dyadic endpoint ledgers, finite budget rows, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, local* NameCert *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. No step appeals to host real distributivity.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier componentwise distributivity`

Proof. Start with the operation closure of Definition 1470.6. It identifies the available addition and product rows as projections of the finite carrier, not as ambient operations on completed real quotients.

Build the left route by reading the $\mathsf{RegularCauchySum}^{\uparrow}$ budget row for the two addends and then the $\mathsf{RegularCauchyProduct}^{\uparrow}$ row for the multiplier and the displayed sum readback. The sum handoff of Definition 417.6 and the product handoff of Definition 461.4 keep the route in regular rational readback data.

Build the right route by reading the two product rows first and then the sum row on their readbacks. The seal-boundary theorems Definition 461.5 and Definition 417.8 keep both products and the final sum behind the displayed dyadic and budget ledgers.

The componentwise hsame transport and displayed Cont replay rows move the two routes together. Therefore the distributive read is a finite packet obligation through $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{RegularCauchySum}^{\uparrow}$, not an imported law of host real multiplication and addition. □

Theorem 1470.9 (Regular Cauchy ring Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier reaches a terminal seal only through one of the four displayed routes*

$$S \longmapsto R_S, \qquad G \longmapsto R_G, \qquad M \longmapsto R_M, \qquad L \longmapsto R_L,$$

together with the relevant source windows, dyadic endpoint ledgers, finite budget ledger, componentwise hsame *transports,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *row. The packet exports no quotient stream, host real equality, selected completed limit, trichotomy, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier real seal nonescape`

Proof. Begin with the terminal seal clause of Definition 1470.2. It forces any real-facing consumer to choose one of the four operation routes and to keep the corresponding source, window, dyadic, budget, transport, continuation, provenance, and naming rows attached.

The sibling non-escape clauses then close the four cases. For the sum route use Definition 417.9; for the negation route use Definition 459.9; for the product route use Definition 461.11; for the scale route use Definition 465.18.

Since the ring carrier contains only those four terminal routes and their shared structural rows, no real-facing consumer can read a fifth route through host arithmetic, quotient real equality, selected limits, trichotomy, countable choice, or ambient completeness. □

Theorem 1470.10 (Regular Cauchy ring componentwise ring laws)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier, associativity of displayed addition and product routes, additive inverse readback, distributivity of product over sum, and rational-scalar compatibility are inherited only through the componentwise operation rows* $\mathsf{RegularCauchySum}^{\uparrow}$*,* $\mathsf{RegularCauchyNegation}^{\uparrow}$*,* $\mathsf{RegularCauchyProduct}^{\uparrow}$*, and* $\mathsf{RegularCauchyScale}^{\uparrow}$*, together with the source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The ring packet does not import host real ring laws, quotient real equality, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the operation-row admission theorem Definition 1470.3. It separates the four algebraic cases S, G, M, L and admits each one only as the displayed sibling operation row attached to the source windows, dyadic ledgers, finite budget rows, transport, replay, provenance, and local naming.

For the addition and additive-inverse cases, use the sibling non-escape clauses already used by the Real seal boundary: Definition 417.9 for S and Definition 459.9 for G. These clauses keep associativity reads and inverse reads inside $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal rows.

For the product and rational-scalar cases, use Definition 461.11 for M and Definition 465.18 for L. The componentwise distributivity theorem Definition 1470.8 supplies the mixed sum-product route, while the scale route remains a displayed rational-scalar handoff rather than host scalar multiplication on completed quotients.

These four cases exhaust the carrier's algebraic coordinates. Since componentwise transport from Definition 1470.4 preserves only the displayed operation rows and their seal boundaries, every ring-law read is a finite BEDC packet route and not an imported law of ambient completed reals. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier componentwise ring laws`

Theorem 1470.11 (Regular Cauchy ring finite operation induction)**.** *Every ring-law consumer of an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier is exhausted by finite induction over the four operation coordinates* S, G, M, L*. The* S *case reads only the* $\mathsf{RegularCauchySum}^{\uparrow}$ *sibling packet, the* G *case reads only the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *sibling packet, the* M *case reads only the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *sibling packet, and the* L *case reads only the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *sibling packet, together with their displayed source rows, budget rows, transport, replay, provenance, local naming, and terminal* $\mathsf{Real}^{\uparrow}$ *seals. No induction branch can route through host real algebra, quotient equality, selected limits, trichotomy, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier finite operation induction`

Proof. Project the carrier of Definition 1470.1. Its operation surface is the finite four-coordinate family S, G, M, L, so induction over possible ring-law reads has exactly four cases.

In the S and G cases, use operation-row admission and operation closure, Definition 1470.3 and Definition 1470.6. They keep addition and negation inside the $\mathsf{RegularCauchySum}^{\uparrow}$ and

$\mathsf{RegularCauchyNegation}^{\uparrow}$ sibling rows with their displayed source, window, dyadic, budget, and seal data.

In the M and L cases, use the same admission and closure results for the $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{RegularCauchyScale}^{\uparrow}$ sibling rows. The componentwise distributivity theorem Definition 1470.8 supplies the mixed sum-product route without introducing an ambient multiplication or addition law.

Finally apply the Real seal non-escape theorem Definition 1470.9. Since each branch reaches a terminal seal only through its displayed sibling operation packet, the four induction cases exhaust all ring-law consumers and leave no route to host real algebra, quotient equality, selected limits, trichotomy, choice, or completeness. □

Theorem 1470.12 (Regular Cauchy ring obligation surface)**.** *Every ring consumer of an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier*

$$K = (A, B, W_A, W_B, D_A, D_B, S, G, M, L, R_S, R_G, R_M, R_L, E_S, E_G, E_M, E_L, H, C, P, N)$$

must read exactly the four displayed operation rows S, G, M, L *together with the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows,* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledgers, terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *row before the ring packet can be consumed as algebraic data.*

The operation rows are the addition, negation, product, and rational-scale surfaces already supplied by the sibling $\mathsf{RegularCauchySum}^{\uparrow}$*,* $\mathsf{RegularCauchyNegation}^{\uparrow}$*,* $\mathsf{RegularCauchyProduct}^{\uparrow}$*, and* $\mathsf{RegularCauchyScale}^{\uparrow}$ *chapters. No consumer may substitute a host real operation, quotient equality branch, selected limit, trichotomy, countable-choice branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row for any part of this displayed surface.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier obligation surface`

Proof. Project the carrier of Definition 1470.1. The only algebraic operation coordinates are S, G, M, L, and each is paired with displayed source rows, aligned windows, dyadic ledgers, finite budget rows, a terminal seal, transport, replay, provenance, and local naming.

Apply operation-row admission Definition 1470.3 and operation closure Definition 1470.6. They identify the four rows as the complete operation surface and keep the readback inside the same finite regular-Cauchy packet.

Use finite operation induction Definition 1470.11. The S, G, M, and L cases cite the sibling handoffs for sum, negation, product, and scale, so every consumer reads the same L10 completion-facing rows before any algebraic conclusion is public.

Finally combine the terminal-seal discipline and Real seal non-escape theorems, Definition 1470.5 and Definition 1470.9. They force every terminal $\mathsf{Real}^{\uparrow}$ read to pass through one of R_S, R_G, R_M, R_L with its operation and budget row still visible, which is exactly the obligation surface stated above. □

Theorem 1470.13 (Regular Cauchy ring operation inversion)**.** *Let* K *be an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier. Any attempted algebraic read from* K *inverts by finite case split on the operation row that licenses it:*

1. *an addition read is the* S *branch through* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* R_S*;*
2. *an additive-inverse read is the* G *branch through* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *and* R_G*;*
3. *a product read is the* M *branch through* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *and* R_M*;*

4. a rational-scale read is the L branch through $\mathsf{RegularCauchyScale}^{\uparrow}$ *and* R_L.

These four cases exhaust the carrier's operation surface. There is no fifth branch through host real algebra, quotient equality, selected limits, trichotomy, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from finite operation induction, Definition 1470.11. It reduces every ring-law consumer to exactly one of the four operation coordinates S, G, M, L.

In the S case, apply operation-row admission and operation closure, Definition 1470.3 and Definition 1470.6. The sum read is therefore a $\mathsf{RegularCauchySum}^{\uparrow}$ sibling handoff with its source, window, dyadic, budget, transport, replay, provenance, naming, and terminal R_S rows attached.

In the G case, the same two theorems identify the negation read as the $\mathsf{RegularCauchyNegation}^{\uparrow}$ sibling handoff with terminal R_G. The selected source row and its displayed finite budget remain attached, so the branch cannot become host additive inverse on completed reals.

In the M and L cases, the product and rational-scale handoffs are read through $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{RegularCauchyScale}^{\uparrow}$, with terminal seals R_M and R_L. Componentwise distributivity, terminal-seal discipline, and Real seal non-escape keep the mixed algebraic reads behind the same displayed sibling routes.

Since the carrier contains no operation coordinate other than S, G, M, L, the case split has no residual branch. The inversion therefore refuses host real algebra, quotient equality, selected limits, trichotomy, countable choice, and ambient completeness as unavailable endpoints. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingCarrier operation inversion`

Theorem 1470.14 (Regular Cauchy ring obligation upgrade)**.** *For every accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier, the carrier admission of Definition 1470.1, operation-row admission, componentwise transport, terminal-seal discipline, operation closure, componentwise distributivity, finite operation induction, obligation surface, and operation inversion together enumerate the full finite* $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$ *obligation surface over* $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

The enumeration is exactly the four operation cases S, G, M, L with their displayed source rows, aligned windows, dyadic ledgers, finite budget rows, terminal seal rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It contains no branch through host ring laws, quotient real equality, trichotomy, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1470.1. The only algebraic coordinates are the four operation rows S, G, M, L, and every structural row is attached to those coordinates through the displayed source, window, dyadic, budget, seal, transport, replay, provenance, and naming data.

Split over the four operation rows. The S branch is admitted by Definition 1470.3 and closes by Definition 1470.6 through $\mathsf{RegularCauchySum}^{\uparrow}$ and R_S. The G branch uses the same theorems through $\mathsf{RegularCauchyNegation}^{\uparrow}$ and R_G.

The M branch is the product branch through $\mathsf{RegularCauchyProduct}^{\uparrow}$ and R_M, and the L branch is the rational-scale branch through $\mathsf{RegularCauchyScale}^{\uparrow}$ and R_L. Componentwise distributivity is supplied by Definition 1470.8, while finite operation induction in Definition 1470.11 says these four branches exhaust every ring-law consumer.

Finally apply the obligation surface and inversion theorems, Definition 1470.12 and Definition 1470.13. They repack the four case routes into one NameCert surface and rule out a residual branch through host real algebra, quotient equality, selected limits, trichotomy, choice, or completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingObligationUpgrade`

Theorem 1470.15 (Regular Cauchy ring scoped kernel dependency)**.** *The scoped kernel of an accepted* $\mathsf{RegularCauchyRing}^{\uparrow}$ *carrier consists only of the carrier coordinates of Definition 1470.1, the four operation rows* $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, *and* $\mathsf{RegularCauchyScale}^{\uparrow}$, *their distributivity and operation-inversion obligations, the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* Cont *and* hsame *transport, and the source surfaces* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$. *No part of the carrier, operation admission rows, distributivity route, Real seal boundary, or operation inversion depends on host real algebra, quotient real equality, selected completed limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRingUp.RegularCauchyRingScopedKernelDependency`

Proof. Project the carrier and operation-row admission theorems, Definition 1470.1 and Definition 1470.3. They name exactly the source rows, aligned stream windows, dyadic ledgers, four sibling operation rows, finite budget rows, structural transport, replay, provenance, and local naming data.

For algebraic use, apply operation closure, componentwise distributivity, finite operation induction, and operation inversion: Definition 1470.6, Definition 1470.8, Definition 1470.11, and Definition 1470.13. These theorems reduce every ring consumer to the four displayed sibling rows and their source data.

Finally apply terminal-seal discipline and Real seal non-escape, Definition 1470.5 and Definition 1470.9. The terminal rows stay behind the same operation and budget ledgers, so the scoped kernel is precisely the displayed BEDC dependency list and has no residual host-real branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, aligned StreamName windows, DyadicRatCore endpoint ledgers, RegularCauchySum, RegularCauchyNegation, RegularCauchyProduct, and RegularCauchyScale operation rows, terminal RealUp seal rows, finite budget ledgers, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1470.1, Definition 1470.2, Definition 1470.6, Definition 1470.9, Definition 1470.14, and Definition 1470.15 bind the carrier, operation rows, distributivity, terminal Real seal, operation inversion, DyadicRatCore, StreamName, RegSeqRat, RegularCauchySum, RegularCauchyNegation, RegularCauchyProduct, RegularCauchyScale, Cont/hsame transport, and RealUp sealing as the chapter's exact kernel scope.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyRing}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, host real arithmetic laws, trichotomy, countable choice, selected completed limits, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation-row admission, componentwise transport, and Real seal non-escape for the same finite algebraic packet.

Chapter 1471

A Concrete Naming Certificate for $\mathsf{RegularCauchyReciprocal}^{\uparrow}$

Bishop and Bridges-style constructive reciprocal data begin from a regular Cauchy rational name together with a displayed apartness window that keeps the inspected tail away from zero. The $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ packet names the finite interface needed to read that reciprocal surface. It refines the budget-only surface of chapter 703 by binding the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the finite windows of chapter 126, the dyadic endpoint and radius arithmetic of chapter 291, and the terminal seal of chapter 38 into one local NameCert carrier. The chapter records only the finite reciprocal-read surface; it does not construct a total reciprocal operation on $\mathsf{Real}^{\uparrow}$.

Definition 1471.1 (Regular Cauchy reciprocal carrier)**.** *A* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, A, M, W, D, B, T, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational entries are being inspected.*
2. *A is the apartness-window row displaying a rational lower bound that keeps the inspected tail of Q away from zero.*
3. *M is the finite modulus row selecting the precision at which the reciprocal read is requested.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened from Q under A and M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and comparison ledger read from W.*
6. *B is the sibling* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row that records the reciprocal-window budget over the same Q, A, M, W, D rows.*
7. *T is the regular rational readback row naming the reciprocal candidate as a* $\mathsf{RegSeqRat}^{\uparrow}$ *surface.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed source, apartness, modulus, window, dyadic, budget, and readback rows have been consumed.*
9. *H records componentwise* hsame *transport for $Q, A, M, W, D, B, T, E, C, P, N$.*
10. *C records the displayed* Cont *replay for source inspection, apartness exposure, finite-window selection, dyadic reciprocal arithmetic, budget handoff, regular readback, and Real seal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$ *row.*
The local classifier compares accepted carriers only through the displayed rows $Q, A, M, W, D, B, T, E, H, C, P, N$.

Theorem 1471.2 (Regular Cauchy reciprocal apartness window)**.** *Every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier exposes the apartness row A before any reciprocal dyadic row D, budget row B, regular readback row T, or* $\mathsf{Real}^{\uparrow}$ *seal row E can be consumed. A route that omits A remains a boundary attempt and cannot export a reciprocal candidate.*

Proof. Project the carrier of Definition 1471.1. The source-side rows are Q, A, M, W, and the reciprocal-facing rows begin only after the apartness and modulus rows have selected a finite window.

The dyadic ledger D reads endpoint, radius, and comparison data from W. Since the reciprocal denominator is controlled by the displayed lower bound in A, the dyadic reciprocal row has no carrier route that bypasses A.

The budget row B, readback row T, and seal row E are downstream of that dyadic row. Therefore omitting A leaves no displayed route from the source to a reciprocal candidate; the local classifier records such a route as boundary evidence. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalCarrier apartness window`

Theorem 1471.3 (Regular Cauchy reciprocal readback route)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the reciprocal readback row T factors through*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T.$$

The terminal $\mathsf{Real}^{\uparrow}$ *seal E is reached only after this finite route has been displayed.*

Proof. Begin with the apartness-window theorem. It places A before any dyadic reciprocal arithmetic, so the route to T cannot start from Q alone.

Next open the finite window row W by using the modulus row M. The dyadic ledger D is a read of that window, and the sibling budget row B is the same reciprocal-window discipline named by chapter 703.

Finally project T and then E from the carrier. The readback row names the reciprocal candidate as regular rational data, while the seal row is terminal and adds no second route around the displayed finite chain. □

Theorem 1471.4 (Regular Cauchy reciprocal classifier stability)**.** *Let*

$$R = (Q, A, M, W, D, B, T, E, H, C, P, N) \quad \text{and} \quad R' = (Q', A', M', W', D', B', T', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carriers. If the displayed transport row H supplies componentwise* hsame *comparisons from each row of R to the corresponding row of R', then the local classifier preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *source, apartness row, modulus row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row, reciprocal readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalClassifierStability`

Proof. Project the transport coordinate H from the carrier. Its comparisons are componentwise, so the source rows Q, Q', apartness rows A, A', modulus rows M, M', finite windows W, W', dyadic ledgers D, D', budget rows B, B', readback rows T, T', and seal rows E, E' are matched before any classifier conclusion is read.

Apply the apartness-window theorem to both carriers. Since the transported apartness rows remain the first reciprocal-facing rows, the reciprocal denominator witness is preserved by the classifier.

Apply the readback-route theorem to both carriers. The transported route from Q through B to T matches the transported route from Q' through B' to T', and the terminal seal rows are reached only after those finite routes.

Thus classifier equality cannot forget, reorder, or replace any analytic row of the reciprocal packet. It transports exactly the displayed source, apartness, modulus, window, dyadic, budget, readback, and Real seal coordinates. □

Theorem 1471.5 (Regular Cauchy reciprocal terminal seal non-totality)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row E of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier cannot be reread as a total reciprocal operation on all* $\mathsf{Real}^{\uparrow}$ *names. Any route from the seal back to a reciprocal candidate must still factor through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row Q, apartness row A, modulus row M, finite* $\mathsf{StreamName}^{\uparrow}$ *window W, dyadic ledger D,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B, and regular readback row T.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalTerminalSealNonTotality`

Proof. Inspect the carrier definition. The seal coordinate E is listed after the source, apartness, modulus, window, dyadic, budget, and readback rows, and the classifier reads it as one coordinate of the twelve-row packet.

The readback-route theorem states that E is reached only after the finite route through T has been displayed. Therefore the seal does not introduce a second route from arbitrary $\mathsf{Real}^{\uparrow}$ names to reciprocal data.

The apartness-window theorem keeps A mandatory for every reciprocal-facing row. Since a total reciprocal on all $\mathsf{Real}^{\uparrow}$ names would erase the need for such a displayed apartness row, it is not present in the packet.

Consequently the terminal seal records only the end of one finite regular-Cauchy reciprocal read. It cannot be exported as a total reciprocal operation or as a boundary bypass around the apartness obligation. □

Theorem 1471.6 (Regular Cauchy reciprocal denominator apartness)**.** *For every accepted* RegularCauchyReciproca *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the dyadic reciprocal row D and the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B are admissible only after the apartness row A displays a positive lower bound for the inspected* $\mathsf{RegSeqRat}^{\uparrow}$ *tail named by Q, M, W. A consumer that reads D or B without that positive lower-bound display has not reached a reciprocal row of the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalDenominatorApartness`

Proof. Project Definition 1471.1. The source row Q, apartness row A, modulus row M, and finite window row W precede the dyadic reciprocal row D and budget row B in the carrier route.

Apply Definition 1471.2. It states that A is exposed before D, B, T, E can be consumed, so the denominator-side condition is part of the carrier admission rather than a later Real-facing consequence.

Finally read Definition 1471.3. The route from Q to T passes through A, M, W, D, B in that order, and therefore D and B are reciprocal rows only when the inspected regular tail has the displayed positive lower bound supplied by A. □

Theorem 1471.7 (Regular Cauchy reciprocal nonzero-denominator window)**.** *Any accepted* RegularCauchyReciproc *packet whose dyadic inverse budget is read must expose a nonzero-denominator window consisting of the* RegSeqRat$^{\uparrow}$ *source row Q, apartness row A, finite modulus row M,* StreamName$^{\uparrow}$ *window W, and* DyadicRatCore$^{\uparrow}$ *row D. Without all five rows the packet has not reached the* CauchyInverseBudget$^{\uparrow}$ *row B.*

Proof. Project the carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ from Definition 1471.1. The only source, apartness, modulus, window, and dyadic rows available to the reciprocal budget are Q, A, M, W, D.

Apply Definition 1471.6. It places the positive lower-bound display in A before the dyadic reciprocal row D and budget row B can be consumed.

Then apply Definition 1471.3. The route to the reciprocal readback passes through Q, A, M, W, D, B, so a consumer that omits any one of Q, A, M, W, D remains before the CauchyInverseBudget$^{\uparrow}$ row rather than at the dyadic inverse budget. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalNonzeroDenominatorWindo`

Theorem 1471.8 (Regular Cauchy reciprocal tail stability)**.** *Let $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ be an accepted* RegularCauchyReciprocal$^{\uparrow}$ *carrier. If a later inspected* StreamName$^{\uparrow}$ *window preserves the* RegSeqRat$^{\uparrow}$ *source row Q together with the apartness row A, modulus row M, finite window row W, and dyadic reciprocal row D, then the same reciprocal readback row T and terminal* Real$^{\uparrow}$ *seal row E transport through H, C, P, N. Thus shifting to a later tail window cannot change the reciprocal readback unless it changes one of A, M, W, D.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalTailStability`

Proof. Use Definition 1471.6. A later window that preserves A, M, W, D keeps the same displayed positive lower-bound and dyadic reciprocal data for the inspected regular tail.

Apply Definition 1471.3. Since the route to T passes through Q, A, M, W, D, B, preserving the apartness, window, and dyadic rows preserves the reciprocal readback route once the same budget row is read.

Apply Definition 1471.4. The transport row H compares corresponding carrier rows componentwise, while C, P, N replay, source, and name the same packet. The terminal seal E is therefore transported with T rather than recomputed from a new denominator condition. □

Theorem 1471.9 (Regular Cauchy reciprocal window composition)**.** *Let $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ be an accepted* RegularCauchyReciprocal$^{\uparrow}$ *carrier. Suppose a later finite* StreamName$^{\uparrow}$ *window preserves the displayed rows Q, A, M, W, D and then reads the same* CauchyInverseBudget$^{\uparrow}$ *row B. The composed finite route still reaches the same regular rational readback row T and terminal* Real$^{\uparrow}$ *seal row E through*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T \longmapsto E.$$

Thus window composition changes neither the reciprocal readback nor the Real seal unless it changes one of the displayed source, apartness, modulus, window, dyadic, or budget rows.

Proof. Apply Definition 1471.6. It keeps the apartness row A in front of every dyadic reciprocal and budget read, so the later window cannot compose into a reciprocal route while dropping the positive lower-bound display.

Use Definition 1471.7. The preserved rows Q, A, M, W, D are exactly the nonzero-denominator window needed before the same budget row B can be read.

Apply Definition 1471.3. Once the same budget row is present, the route to T is the displayed chain through Q, A, M, W, D, B, and the seal E is terminal after that chain.

Finally use Definition 1471.8. Since the later window preserves the source, apartness, modulus, window, and dyadic rows and reads the same budget row, the transported packet reaches the same T and E rather than recomputing a new reciprocal surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalWindowComposition`

Theorem 1471.10 (Regular Cauchy reciprocal window denominator stability)**.** *Let $R = (Q, A, M, W, D, B, T, E, H$ be an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier. Suppose a later finite* $\mathsf{StreamName}^{\uparrow}$ *window preserves the displayed rows Q, A, M, W, D and reads the same* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row B. Then the denominator apartness witness recorded by A remains visible before the reciprocal readback row T and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are consumed. In particular, the later window cannot treat B, T, E as a hidden choice of denominator data.*

Proof. Apply Definition 1471.6. The dyadic reciprocal row D and budget row B are admissible only after A displays the positive lower-bound witness for the inspected regular tail.

Use Definition 1471.7. Since the later window preserves Q, A, M, W, D, it preserves exactly the nonzero-denominator surface that must be present before the same $\mathsf{CauchyInverseBudget}^{\uparrow}$ row is read.

Apply Definition 1471.8. Preserving the source, apartness, modulus, window, and dyadic rows transports the same readback row T and terminal seal row E through the carrier rows H, C, P, N.

Finally use Definition 1471.9. The composed route reaches T and then E only through the displayed chain Q, A, M, W, D, B, T, E, so the denominator apartness witness remains visible throughout the reciprocal handoff rather than becoming hidden choice data. □

Theorem 1471.11 (Regular Cauchy reciprocal located apartness window)**.** *Any* $\mathsf{Real}^{\uparrow}$*-seal or budget-facing consumer of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row Q, apartness row A, modulus row M,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row D, and* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *budget row B before reading the reciprocal readback row T or terminal seal row E. The route exports no total reciprocal on arbitrary* $\mathsf{Real}^{\uparrow}$ *names, trichotomy, quotient stream equality, selected limit, or ambient completeness.*

Proof. Project the accepted carrier. The only reciprocal-facing analytic rows before readback are the source, apartness, modulus, window, dyadic comparison, and budget rows Q, A, M, W, D, B.

Apply Definition 1471.2, Definition 1471.6, and Definition 1471.7. These place the located apartness display in A before the dyadic comparison row and identify Q, A, M, W, D as the nonzero-denominator window required before B is read.

Next use Definition 1471.9. It preserves the same route through Q, A, M, W, D, B, T, E under later finite windows, so reading T cannot detach the reciprocal candidate from its displayed denominator witness.

Finally apply Definition 1471.13. The terminal seal E is offered only after the source, apartness, modulus, window, dyadic, budget, and readback rows have been displayed. Since the carrier has no

coordinate for total reciprocals, trichotomy, quotient stream equality, selected limits, or ambient completeness, those principles cannot be exported by either the budget-facing route or the Real-seal route. □

Theorem 1471.12 (Regular Cauchy reciprocal scoped denominator apartness)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, the denominator-apartness witness is scoped exactly by the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, apartness row* A*, finite modulus row* M*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. It exports no total* $\mathsf{Real}^{\uparrow}$ *reciprocal, quotient stream, selected limit, or host equality.*

Proof. Project the carrier of Definition 1471.1. The only denominator-facing rows before reciprocal readback are Q, A, M, W, D, B, and the only structural rows that can transport or replay them are H, C, P, N.

Apply Definition 1471.2 and Definition 1471.6. These place the apartness row A before every dyadic reciprocal, budget, readback, and seal coordinate, and they make the positive lower-bound display part of carrier admission.

Use Definition 1471.7 and Definition 1471.10. The first theorem identifies Q, A, M, W, D as the nonzero-denominator window required before B is read, while the second preserves that witness under later finite windows.

The remaining rows H, C, P, N carry only componentwise transport, replay, provenance, and local naming for the same finite surface. Thus the scoped apartness witness is exactly the displayed BEDC primitive surface $Q, A, M, W, D, B, H, C, P, N$, with no exported total reciprocal on $\mathsf{Real}^{\uparrow}$, quotient stream, selected limit, or host equality. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.TasteGate.RegularCauchyReciprocalScopedDenominatorApartn`

Theorem 1471.13 (Regular Cauchy reciprocal Real seal handoff)**.** *Let* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier. Any reciprocal consumer that is offered the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *must first be offered the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, apartness row* A*, modulus row* M*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row* B*, and reciprocal readback row* T*. Thus the handoff from finite reciprocal-read data to the L10* $\mathsf{Real}^{\uparrow}$ *boundary is the displayed route*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B \longmapsto T \longmapsto E$$

and not a direct operation from arbitrary $\mathsf{Real}^{\uparrow}$ *names to reciprocals.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalRealSealHandoff`

Proof. Apply Definition 1471.2. It places the apartness row A before every reciprocal-facing coordinate, so a consumer cannot treat the terminal seal as available before the denominator witness is displayed.

Next use Definition 1471.3. The route to the readback row T factors through Q, A, M, W, D, B, and the seal row E is terminal only after that finite route has been exposed.

Apply Definition 1471.6 and Definition 1471.7. These identify the source, apartness, modulus, window, and dyadic rows as the nonzero-denominator surface required before the budget row B can be consumed.

Finally use Definition 1471.8 together with Definition 1471.9. Transport to later windows and composition of finite windows preserve the same T and E only when the displayed rows

Q, A, M, W, D, B are preserved. The handoff therefore exports neither a total reciprocal, quotient real equality, trichotomy, selected limit, nor ambient completeness principle; it exports only the finite route from reciprocal readback data to the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 1471.14 (Regular Cauchy reciprocal NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$ *surface consists exactly of carrier exposure for* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, apartness-window exposure through* A*, finite-window exposure through* W*, dyadic reciprocal arithmetic through* D*, sibling budget exposure through* B*, regular readback through* T*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalNameCertObligations`

Proof. Carrier exposure is Definition 1471.1. It supplies the full row list and states that the classifier reads only those rows.

The analytic route is given by Definition 1471.2 and Definition 1471.3. Together they force every reciprocal read to pass through the source, apartness, modulus, finite window, dyadic arithmetic, and sibling budget rows before reaching T and E.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of these rows contains a total reciprocal, quotient stream equality, selected limit, trichotomy decision, or ambient completeness principle, the displayed obligations exhaust the local NameCert surface. □

Theorem 1471.15 (Regular Cauchy reciprocal window-budget coverage)**.** *Every budget-facing read of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$ *must expose the rows*

$$Q \longmapsto A \longmapsto M \longmapsto W \longmapsto D \longmapsto B$$

in that order before it treats the packet as carrying a reciprocal-window budget.

Proof. Project the carrier of Definition 1471.1. The budget row B is displayed only after the source row Q, apartness row A, modulus row M, finite window row W, and dyadic row D.

Apply Definition 1471.2 and Definition 1471.6. These place the positive lower-bound display before any dyadic or budget read.

Use Definition 1471.7. It identifies Q, A, M, W, D as the nonzero-denominator window required before the $\mathsf{CauchyInverseBudget}^{\uparrow}$ row is reached.

Finally apply Definition 1471.9 and Definition 1471.14. Composed windows preserve the same displayed budget route, and the local NameCert surface contains no alternate budget coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalWindowBudgetCoverage`

Theorem 1471.16 (Regular Cauchy reciprocal readback-budget exactness)**.** *For every accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier* $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$*, the readback row* T *is exactly the finite reciprocal-read route through the displayed budget row* B*. It is not a total reciprocal operation on arbitrary* $\mathsf{Real}^{\uparrow}$ *names, and it cannot be read without the preceding rows* Q, A, M, W, D, B*.*

Proof. Use Definition 1471.3. It factors T through the displayed chain from Q to B.

Apply Definition 1471.15. Any budget-facing read must first expose Q, A, M, W, D, B, so T has no route that bypasses the finite budget row.

Apply Definition 1471.8 and Definition 1471.9. Later windows and composed windows preserve the same readback only when the displayed source, apartness, modulus, window, dyadic, and budget rows are preserved.

Finally use Definition 1471.5. Since the terminal seal is not a total reciprocal on $\mathsf{Real}^{\uparrow}$, the preceding readback row is also only the finite route named by the carrier. □

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalReadbackBudgetExactness`

Theorem 1471.17 (Regular Cauchy reciprocal seal-route conservativity)**.** *Any* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *carrier receives only the displayed route*

$$Q, A, M, W, D, B, T, E, H, C, P, N.$$

The seal row E conservatively packages that route and adds no total reciprocal, trichotomy, quotient stream equality, selected limit, or ambient completeness principle.

Proof. Apply Definition 1471.13. The terminal seal is offered only after the source, apartness, modulus, finite window, dyadic, budget, and readback rows have been displayed.

Use Definition 1471.16. It fixes T as the finite route through B, so the seal cannot detach E from the reciprocal-window budget.

Use Definition 1471.14. The remaining rows H, C, P, N transport, replay, source, and name the same packet, and no additional analytic coordinate exists.

The non-totality clause follows from Definition 1471.5. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only the displayed conservative route. □

Theorem 1471.18 (Regular Cauchy reciprocal scope dependency lock)**.** *For every accepted* RegularCauchyReciproc *carrier $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, the reciprocal readback row T and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are scoped exactly by five L10 source rows: the* $\mathsf{RegSeqRat}^{\uparrow}$ *source Q, the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row B, and the* $\mathsf{Real}^{\uparrow}$ *seal route E. A consumer that drops any one of these rows remains outside the reciprocal readback and seal surface. The dependency lock exports no total reciprocal, quotient real equality, selected limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalScopeDependencyLock`

Proof. Begin with Definition 1471.2. It places the apartness row before every reciprocal-facing coordinate, so the route to the five named source rows is not optional.

Next apply Definition 1471.3. The readback row T factors through the displayed chain from the $\mathsf{RegSeqRat}^{\uparrow}$ source to the reciprocal budget row.

Use Definition 1471.15 together with Definition 1471.16. These two theorems identify the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{CauchyInverseBudget}^{\uparrow}$ row as the finite budget boundary for T.

Finally invoke Definition 1471.17. The $\mathsf{Real}^{\uparrow}$ seal E only packages the same displayed route, and its nonescape clause excludes total reciprocal, quotient equality, selected limits, and ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ consumer that reaches a reciprocal readback row or a $\mathsf{Real}^{\uparrow}$ seal without the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, apartness row, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{CauchyInverseBudget}^{\uparrow}$ row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, cauchyinversebudget, real: no listed sibling supplies the whole twelve-row packet $R = (Q, A, M, W, D, B, T, E, H, C, P, N)$, because each sibling contributes only a regular source, finite window, dyadic ledger, reciprocal budget, or Real seal while this chapter binds all five analytic faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source, an apartness-window row, a finite modulus row, a $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{CauchyInverseBudget}^{\uparrow}$ budget row, a reciprocal readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1471.1, Definition 1471.2, Definition 1471.3, Definition 1471.4, Definition 1471.5, Definition 1471.14, Definition 1471.15, Definition 1471.16, Definition 1471.17, and Definition 1471.18 make the reciprocal chapter mature on the paper closure axis through the checked dependency lock, denominator-apartness window, $\mathsf{Real}^{\uparrow}$ seal handoff, and window-budget coverage.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReciprocal}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.RegularCauchyReciprocalUp.RegularCauchyReciprocalScopeDependencyLock`

Bridge status: none

Not claimed: This chapter does not claim a total reciprocal, trichotomy, quotient stream equality, selected limits, ambient real completeness, or a public bridge.

Upgrade path: Standard-bridge verification remains a separate formal route; the paper closure axis is mature for the displayed reciprocal dependency lock.

Chapter 1472

CompactLocatedSubcover NameCert Construction

Following Bishop and Bridges, the $\mathsf{CompactLocatedSubcover}^{\uparrow}$ packet records finite located subcover extraction as displayed BEDC data: a finite family of named located balls is already present, each member has a positive radius row, and the certificate extracts the finite bundle and lower-radius ledger read by compact and uniform consumers. The chapter treats the object as a located finite extraction surface, not as open-cover compactness. It sits between the finite-net compact packets of chapter 60 and chapter 129, the finite positive-radius ledger of chapter 452, and the downstream modulus consumer of chapter 369.

Definition 1472.1 (Compact located subcover carrier)**.** *A* $\mathsf{CompactLocatedSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, C, R, U, V, L, H, Q, P, N)$$

with the following displayed rows:

1. *B is a finite* ProbeBundle *spine naming the located cover candidates inspected by the packet;*
2. *C is the center-history ledger assigning to every bundled probe a center* BHist *row readable through the compact or compact-metric source;*
3. *R is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger attached to the bundled candidates;*
4. *U is the positive* $\mathsf{RationalBall}^{\uparrow}$ *radius readback for the same centers and dyadic radii;*
5. *V is the coverage row stating that every compact-source history accepted by the packet is covered by one bundled located ball;*
6. *L is the lower-radius fold over the finite bundle, producing one positive radius ledger below every displayed radius in R and U;*
7. *H records componentwise* hsame *transport for bundle, center, dyadic-radius, rational-ball, coverage, and lower-fold rows;*
8. *Q records the displayed* Cont *routes by which compact, compact-metric, finite-Lebesgue-number, and uniform-modulus consumers inspect the packet;*
9. *P is the* Pkg *provenance over* $\mathsf{CompactLocatedSubcover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactLocatedSubcover}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite located subcover extraction route. It has no coordinate for an arbitrary open family, an ambient compactness theorem, a selected open-cover subfamily, quotient real equality, countable choice, or an external modulus selector.

Theorem 1472.2 (CompactLocatedSubcover NameCert obligations)**.** *For an accepted* $\mathsf{CompactLocatedSubcover}^{\uparrow}$ *carrier $S = (B, C, R, U, V, L, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactLocatedSubcover}^{\uparrow}}$ *surface consists of five finite rows: carrier admission for the tuple of Definition 1472.1; located-ball source rows over B, C, R, U; coverage rows over V; lower-radius fold rows over L; and downstream transport, replay, provenance, and naming rows over H, Q, P, N.*

These five rows are exactly the finite located extraction packet read by compact and compact-metric consumers. No consumer can replace them by open-cover compactness, a host subcover selector, a hidden radius minimum, or a detached analytic compactness principle.

Proof. Carrier admission fixes the finite packet and the only rows the classifier may inspect. The bundle and center rows are the displayed located candidates, while the dyadic and rational-ball rows give the positive radii attached to those candidates.

The coverage coordinate V is the finite located subcover assertion. It is already indexed by the $\mathsf{ProbeBundle}$ spine B, so a consumer reads a bundled cover row rather than selecting a subfamily from an arbitrary cover.

The fold coordinate L supplies the lower-radius ledger. Since it folds only over the displayed finite bundle and is transported only by H, it cannot read a host minimum or an external choice operator.

Finally the rows Q, P, N replay, source, and name the same packet for compact, compact-metric, finite-Lebesgue-number, and uniform-modulus consumers. Thus the five finite rows are the whole local obligation surface. □

Theorem 1472.3 (CompactLocatedSubcover uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactLocatedSubcover}^{\uparrow}$ *carrier receives the same finite packet*

$$B, C, R, U, V, L, H, Q, P, N.$$

The rows V and L respectively supply the located coverage surface and the positive lower-radius ledger. They hand forward to chapter 452 and chapter 369 as finite BEDC rows, not as open-cover compactness, selected subcover choice, or an external modulus.

Proof. Project the carrier of Definition 1472.1. The compact source contributes only the finite bundle B, center ledger C, dyadic radius ledger R, rational-ball readback U, and coverage row V.

Read the lower-radius fold L over that same finite bundle. The fold provides one positive radius row below the displayed located-ball radii, which is the radius shape expected by finite-Lebesgue-number and uniform-modulus consumers.

Transport the resulting located coverage and radius rows through H, then replay them through Q with provenance and local naming supplied by P and N. Because every downstream read is forced through these rows, the handoff cannot invoke arbitrary open covers, chosen subcovers, host minima, countable choice, or detached real-completion data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLocatedSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactLocatedSubcover}^{\uparrow}$ packet is generated from a finite ProbeBundle of located cover candidates, center histories, positive DyadicRatCore and RationalBall radius rows, coverage rows, a finite lower-radius fold, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactLocatedSubcover}\uparrow}$, seed)

Scope closed: Definition 1472.1, Definition 1472.2, and Definition 1472.3 bind the finite located subcover extraction packet and its radius handoff to finite-Lebesgue-number and uniform-modulus consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactLocatedSubcover}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary cover selection, host compactness, host minima, quotient real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-ball coverage rows, positive-radius readback, lower-radius fold exactness, and uniform-consumer handoff for BEDC.Derived.CompactLocatedSubcoverUp.

Chapter 1473

A Concrete Naming Certificate for $\mathsf{UniformCompletion}^{\uparrow}$

Following Kelley, *General Topology*, and Bishop–Bridges constructive completion, the $\mathsf{UniformCompletion}^{\uparrow}$ packet records uniform completion as finite Cauchy-filter and Cauchy-net evidence over BEDC rows, not quotient completion, selected limits, ambient completeness, or choice. The packet threads the filter boundary of chapter 333, the directed witness boundary of chapter 589, the extension surface of chapter 397, the regular rational handoff of chapter 301, and the real readback boundary of chapter 38.

Definition 1473.1 (UniformCompletion carrier)**.** *A* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(F, D, U, E, H, C, P, N)$$

with the following visible rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *data row recording the finite filterbase, late-window, and cofinality evidence consumed by completion-facing readers;*
2. *D is the directed* $\mathsf{CauchyNet}^{\uparrow}$ *witness row recording the finite net indices, common-window witnesses, and extracted directed subwindows;*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *universal-extension row for uniformly continuous consumers served by the same F and D rows;*
4. *E is the completion equality classifier row, comparing two packet reads only through the displayed filter, net, and universal-extension coordinates;*
5. *H is the componentwise* hsame *row over F, D, U, E;*
6. *C is the displayed* Cont *route used to replay the Cauchy-filter, directed-net, extension, and classifier rows under finite observation;*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, ambient complete uniform spaces, open-cover compactness, or choice.

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletionTasteGate single carrier alignment`

Theorem 1473.2 (UniformCompletion NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{UniformCompletion}^{\uparrow}}$ *surface of Definition 1473.1 consists of five obligations:*

1. *carrier admission for the displayed packet* (F, D, U, E, H, C, P, N)*;*
2. *Cauchy-filter factorization, requiring every completion-facing consumer to read the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *before any completion handoff;*
3. *directed-net witness extraction, requiring directed witnesses to be projected from the* $\mathsf{CauchyNet}^{\uparrow}$ *row* D *rather than from a selected limit;*
4. *universal-extension uniqueness, requiring uniformly continuous extension comparisons to pass through the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row* U *and the classifier row* E*;*
5. *ledger non-escape, stating that the packet exports only the rows named in* P *and never a quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice principle.*

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion namecert obligation surface`

Proof. Carrier admission is projection from the finite tuple (F, D, U, E, H, C, P, N) in Definition 1473.1. Each named coordinate is already a BEDC row, and the local naming row N records that these coordinates are the public $\mathsf{UniformCompletion}^{\uparrow}$ surface.

Cauchy-filter factorization is the obligation that completion-facing reads start with F. Directed-net witness extraction is the parallel obligation that directed witnesses start with D, so neither route can introduce a selected representative or an ambient limit point.

Universal-extension uniqueness is then confined to U and E. The componentwise hsame row H and displayed Cont route C replay these same finite coordinates, while P records that all downstream handoffs stay inside the named Cauchy-filter, Cauchy-net, universal-property, RegSeqRat, and Real rows. □

Theorem 1473.3 (UniformCompletion Cauchy-filter factorization)**.** *For an accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier* (F, D, U, E, H, C, P, N)*, every completion-facing consumer factors through the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *and the* $\mathsf{CauchyNet}^{\uparrow}$ *witness row* D *before reading any* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension data or* $\mathsf{Real}^{\uparrow}$ *handoff data. Thus the completion boundary is exhausted by* F, D, U, E, H, C, P, N *and exports no quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice principle.*

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion cauchy filter factorization`

Proof. Project an accepted carrier (F, D, U, E, H, C, P, N). The first completion-facing coordinate is F, so a consumer reaches Cauchy-filter evidence before it can ask for any completion extension or real handoff.

The next witness coordinate is D. Its directed Cauchy-net subwindows provide the directed evidence used by the completion route, and the packet contains no separate selected-limit coordinate from which such witnesses could be read.

Only after F and D have been consumed can the route read the $\mathsf{CompletionUniversalProperty}^{\uparrow}$ row U, the classifier row E, or the $\mathsf{Real}^{\uparrow}$ handoff data named by P. The transport and replay rows H and C preserve this order because they are componentwise rows over the same finite packet. □

Theorem 1473.4 (UniformCompletion directed-net witness extraction)**.** *For an accepted* UniformCompletion$^{\uparrow}$ *carrier* (F, D, U, E, H, C, P, N)*, every directed witness consumed by a completion-facing route is extracted from the displayed* CauchyNet$^{\uparrow}$ *row* D *after the Cauchy-filter row* F *has fixed the request. The witness is therefore a finite directed-net subwindow, not a selected limit, quotient representative, or ambient complete-uniform-space point.*

Proof. Project the carrier rows of Definition 1473.1. The Cauchy-filter row F names the completion request, and the only directed row available to answer it is D.

The existing factorization route of Definition 1473.3 already forces completion-facing consumers to read F and D before extension or Real handoff data. Thus a directed witness is obtained by inspecting the finite subwindow stored in D.

The remaining rows U, E, H, C, P, N transport, replay, source, classify, and name the same packet. Since none of them stores a selected limit, quotient representative, or ambient point, the extracted witness cannot escape the displayed directed-net row. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion directed net witness certificate`

Theorem 1473.5 (UniformCompletion universal-extension uniqueness)**.** *For an accepted* UniformCompletion$^{\uparrow}$ *carrier* (F, D, U, E, H, C, P, N)*, any two displayed extension routes that read the same* RegSeqRat$^{\uparrow}$ *handoff data and the same* Real$^{\uparrow}$ *seal data agree under the classifier row* E*. The comparison passes through the* CompletionUniversalProperty$^{\uparrow}$ *row* U *and the displayed classifier, and it adds no parameter-transport equivalence or host equality of completion objects.*

Proof. Read both extension routes through the same accepted packet. By Definition 1473.3, both routes first consume F and D, so their completion-facing source data are already the displayed Cauchy-filter and directed-net rows.

The universal-extension coordinate is U. When the two routes present the same RegSeqRat and Real seal rows, U supplies the common extension surface attached to those rows, and E is the only classifier coordinate allowed to compare the two reads.

Componentwise transport and replay through H and C preserve the same classifier input, while P and N source and name only the displayed packet. Therefore agreement is a classifier agreement under E, not an added parameter-transport equivalence or host completion equality. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion universal extension uniqueness`

Theorem 1473.6 (UniformCompletion ledger non-escape)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer read factors through the displayed rows*

$$F, D, U, E, H, C, P, N.$$

In particular, the read enters through the Cauchy-filter and Cauchy-net rows, continues through universal-extension and classifier rows, and is preserved by BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data only. It exports no quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice principle.*

Proof. Start with a consumer read of the accepted carrier. Projection gives exactly the rows F, D, U, E, H, C, P, N.

The factorization theorem forces the analytic part of the read through F and D before any extension or Real-facing handoff is visible. The directed witness extraction row then remains inside the displayed Cauchy-net evidence.

The rows U and E are the only extension and classifier rows. The structural rows H, C, P, N preserve, replay, source, and name those same coordinates, and the carrier contains no further ledger entry.

Thus every consumer read factors through the displayed BHist, transport, continuation, provenance, and naming rows, and none of the excluded quotient, selection, ambient-completeness, open-cover, or choice data can be exported. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion ledger nonescape`

Theorem 1473.7 (UniformCompletion filter-net obligations)**.** *An accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier exposes exactly the* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows needed for carrier admission and filter-to-net factorization. Carrier admission reads the packet as* (F, D, U, E, H, C, P, N)*, the filter row* F *fixes the completion-facing request, the directed-net row* D *supplies the finite witness route for that request, and the universal, Real, and RegSeqRat rows are readable only after* F *and* D *have been displayed.*

Proof. Project the accepted carrier of Definition 1473.1. The displayed coordinates already list the Cauchy-filter row F, the Cauchy-net row D, the universal-property row U, the classifier row E, and the structural rows H, C, P, N.

The filter-to-net factorization is the local route from F to D. The filter row fixes the finite request and the directed-net row provides the corresponding finite witness, so the construction has no place to read a selected limit or a quotient completion object.

The RegSeqRat and Real handoff data are named in the provenance row P and replayed by C after F, D, U, E have been fixed. Componentwise hsame transport through H preserves the same displayed rows, and the local $\mathsf{NameCert}$ row N admits exactly this packet. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.UniformCompletion filter net obligations`

Theorem 1473.8 (UniformCompletion filter-net exactness)**.** *Every accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *packet routes a completion request through the Cauchy-filter row* F *and then through the Cauchy-net row* D *before any universal-extension or classifier read reaches* U *or* E*. In carrier notation, the only accepted completion-facing order is*

$$F \longmapsto D \longmapsto U, E \longmapsto H, C, P, N.$$

Thus the packet exposes filter data, directed finite witness data, extension comparison, transport, replay, provenance, and local naming, with no bypass through quotient completion, selected representatives, ambient complete uniform spaces, open-cover compactness, or choice.

Proof. Project the accepted carrier (F, D, U, E, H, C, P, N) from Definition 1473.1. The carrier projection gives the only displayed coordinates available to a completion-facing request.

Apply Definition 1473.7. It fixes the filter row F as the request boundary and the directed-net row D as the finite witness route for that request.

The universal and classifier rows are downstream of that boundary. The structural rows H, C, P, N transport, replay, source, and name the same route, so no excluded completion object has a coordinate from which to enter. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion filter net exactness`

Theorem 1473.9 (UniformCompletion directed subwindow extraction)**.** *For every accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier* (F, D, U, E, H, C, P, N)*, each directed witness used by a completion consumer is a finite* $\mathsf{CauchyNet}^{\uparrow}$ *subwindow extracted after the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *fixes the request. The extraction route is confined to*

$$F \longmapsto D \longmapsto H, C, P, N,$$

and the universal row U *may read the extracted witness only after this route is displayed.*

Proof. Begin with the finite request fixed by F. By Definition 1473.7, the directed witness is supplied by the row D for that same request, not by an ambient net chosen outside the packet.

Use Definition 1473.4. The witness row is the displayed Cauchy-net evidence available to the packet, so any subwindow extracted for a consumer remains a finite subwindow of D.

The transport, replay, provenance, and naming rows preserve that extracted row. Since U is read only after the filter-net witness is displayed, universal extension cannot introduce a separate directed object. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion directed subwindow extraction`

Theorem 1473.10 (UniformCompletion universal extension scope)**.** *Every extension comparison exported by an accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier is confined to the universal-extension row* U *and the classifier row* E *after the filter-net boundary has been displayed. Equivalently, any such comparison factors as*

$$F, D \longmapsto U, E \longmapsto H, C, P, N,$$

with $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoffs readable only as provenance downstream of that route.*

Proof. First use Definition 1473.8. It places all extension consumers behind the displayed filter row F and directed-net row D.

Next apply Definition 1473.5. The comparison itself is carried by the universal-extension row and read through the classifier row, so the comparison surface is U, E.

Finally use Definition 1473.11. The structural rows preserve the same route, and downstream RegSeqRat or Real handoffs are provenance reads rather than additional universal-extension coordinates. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion extension comparison certificate`

Theorem 1473.11 (UniformCompletion universal ledger non-escape)**.** *Any universal-extension read of an accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *packet factors through the displayed universal-extension uniqueness row and gains no host quotient, selected limit, choice, or ambient completeness coordinate. The only extension comparison route is*

$$F, D \longmapsto U, E \longmapsto H, C, P, N,$$

so a consumer may compare extension reads only through U *and the displayed classifier row* E*.*

Proof. Begin with a universal-extension read. By Definition 1473.3, the read first passes through the Cauchy-filter row F and the directed-net witness row D.

The universal ledger is then U, with comparison confined to the classifier row E. This is exactly the uniqueness route stated in Definition 1473.5.

Finally H and C transport and replay the same finite rows, while P and N source and name the packet. None of those structural rows contains a quotient completion, selected limit, choice principle, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion universal ledger nonescape`

Theorem 1473.12 (UniformCompletion Real-RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *packet reads the completion seal through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the displayed Cauchy filter and Cauchy net rows. The consumer route is therefore confined to the finite chain*

$$F, D \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N$$

named by the carrier provenance, not to a host completed-real object or a selected representative.

Proof. Project the packet and read its provenance row P. The only Real-facing source named there passes through the displayed Cauchy-filter row F, the Cauchy-net row D, the regular rational handoff, and the terminal Real seal.

The already established filter-net obligations force F and D to be consumed before the RegSeqRat and Real rows are public. Thus a Real-facing consumer cannot bypass the finite filter and net witness route.

Transport through H and replay through C are componentwise over the same rows, and N only names that local packet. Consequently the handoff exports no host completed-real object, selected representative, quotient stream, choice principle, or ambient completeness premise. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion real regseqrat handoff`

Theorem 1473.13 (UniformCompletion scope lock)**.** *The* $\mathsf{UniformCompletion}^{\uparrow}$ *filter-net obligations are scoped exactly by the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F*, the* $\mathsf{CauchyNet}^{\uparrow}$ *row* D*, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row* U*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows named by* P*, and the structural rows* H, C, P, N *for* hsame*,* Cont*,* Pkg*, and* NameCert*. Within that scope there is no quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice coordinate.*

Proof. Apply Definition 1473.7. It identifies carrier admission and filter-to-net factorization with the finite rows F, D, U, E, H, C, P, N, and it places the RegSeqRat and Real handoff data after the Cauchy-filter and Cauchy-net rows.

Apply Definition 1473.11. The universal-extension comparison route is confined to

$$F, D \longmapsto U, E \longmapsto H, C, P, N.$$

Thus the extension ledger can compare displayed reads only through the universal-property row and the classifier row.

Finally apply Definition 1473.12. The Real-facing route remains a finite RegSeqRat-to-Real handoff named by the provenance row, with transport and replay preserving the same carrier coordinates. The listed quotient, representative-selection, ambient-completeness, compactness, and choice data are outside the carrier and outside every continuation route. □

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion extension scope regseqrat handoff`

Theorem 1473.14 (UniformCompletion bridge-route interface)**.** *Every completion-facing read of an accepted* $\mathsf{UniformCompletion}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows named by the carrier and its provenance route. The bridge route therefore exposes only finite entourage*

requests, Cauchy-filter data, directed-net witnesses, universal-extension comparison, stream windows, regular rational readback, Real handoff data, componentwise transport, displayed continuation, provenance, and local naming. It exports no quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice principle.

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion bridge route`

Proof. Begin with the accepted carrier of Definition 1473.1. The completion-facing route first reads the Cauchy-filter row F and the directed-net witness row D, and these rows are interpreted over the displayed uniform-space entourage surface named by the surrounding completion route.

The extension comparison then passes through U and E. The stream, regular rational, and Real-facing handoff rows are reachable only through the provenance route already used by Definition 1473.12, so the read cannot bypass the displayed finite rows.

Finally H, C, P, N transport, replay, source, and name the same route. Since none of these coordinates stores a quotient completion, selected representative, ambient complete uniform space, open-cover compactness principle, or choice principle, the bridge-route interface is exhausted by the stated rows. □

Theorem 1473.15 (UniformCompletion public factorization)**.** *Every public* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer factors through the same finite completion route: the* $\mathsf{CauchyFilter}^{\uparrow}$ *evidence row* F*, the* $\mathsf{CauchyNet}^{\uparrow}$ *witness row* D*, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row* U*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row named by provenance, the* $\mathsf{Real}^{\uparrow}$ *seal, the supporting* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The public route exports no quotient completion equality, no selected representative, no countable-choice schedule, no host equality of completed objects, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.UniformCompletionUp.TasteGate.UniformCompletion public factorization`

Proof. Start with the public consumer and apply Definition 1473.3. The consumer first reaches the Cauchy-filter row F and the directed-net row D, before any extension or real handoff data is visible.

Next apply Definition 1473.4 and Definition 1473.5. The directed evidence is extracted from D, while extension comparisons pass through U and the classifier row E rather than through a selected limit or host equality.

Now use Definition 1473.12. Real-facing reads pass through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ seal after the filter and net rows have fixed the completion-facing source data.

Finally apply Definition 1473.13. The structural rows H, C, P, N supply the BHist support, componentwise hsame transport, Cont replay, provenance, and local naming for exactly that route, while the quotient, representative-selection, choice, host-equality, and ambient-completeness coordinates remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1473.1, Definition 1473.2, Definition 1473.3, Definition 1473.4, Definition 1473.5, Definition 1473.6, Definition 1473.7, Definition 1473.11, Definition 1473.12, Definition 1473.13, Definition 1473.15, and Definition 1473.14 close the mature paper package: local obligation surface, scoped filter-net boundary, and paper bridge route for carrier admission, filter-net factorization, directed-net witness extraction, universal-extension uniqueness, Real-RegSeqRat handoff, and ledger non-escape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close quotient completions, selected representatives, ambient complete uniform spaces, open-cover compactness, or choice principles.

Upgrade path: The remaining work is formal-axis alignment of the mature paper package with a canonical Lean target and bridge-ready TasteGate surface for BEDC.Derived.UniformCompletionUp.

Constructive story: The construction starts from displayed Cauchy-filter and directed Cauchy-net rows, then routes finite completion consumers through CompletionUniversalProperty, RegSeqRat, and Real handoff data without adding quotient or choice principles.

Chapter 1474

A Concrete Naming Certificate for $\mathsf{CalkinWilfEnumeration}^{\uparrow}$

Following Calkin and Wilf, the $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ packet records the positive-rational enumeration surface generated by the binary tree rule $a/b \mapsto a/(a+b)$ and $a/b \mapsto (a+b)/b$ as finite BEDC naming data. Its dependency lattice starts from the unary source of chapter 28, the rational carrier of chapter 37, the rational-stream interface of chapter 350, the finite schedule surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It is the breadth-first positive-rational sibling of the continued-fraction route in chapter 342 and the Stern–Brocot approximation route in chapter 1431. The packet is not quotient rational equality, not an ambient host enumeration, not an arbitrary choice of representatives, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1474.1 (Calkin-Wilf enumeration carrier)**.** *A* $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (U, Q, T, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the unary* $\mathsf{Nat}^{\uparrow}$ *address row naming the finite breadth-first tree position.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *numerator-denominator row exposed at that address, with both coordinates positive unary histories.*
3. *T is the local tree-step row recording whether the next finite edge is left, $a/b \mapsto a/(a+b)$, or right, $a/b \mapsto (a+b)/b$.*
4. *B is the breadth-first schedule row ordering finite addresses before they are read as a stream.*
5. *S is the* $\mathsf{RationalStream}^{\uparrow}$ *output row whose U-indexed finite window is filled by the displayed Q, T, B data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the enumeration is consumed as a rational source for Cauchy-style real-name packets.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row, exposed only as a downstream consumer of the rational stream and regular readback rows.*
8. *H stores componentwise* hsame *transport for U, Q, T, B, S, R, E.*
9. *C stores displayed* Cont *replay for opening an address, computing its tree edge, filling the stream window, and passing the selected rational row to regular-rational and real-name consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CalkinWilfEnumeration}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient rational equality, host-side enumeration of all rationals, choice-selected representatives, signed-rational coverage, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1474.2 (Calkin-Wilf enumeration NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CalkinWilfEnumeration}^{\uparrow}}$ *surface for Definition 1474.1 consists of five finite obligation rows:*

1. *carrier admission for the displayed packet $W = (U, Q, T, B, S, R, E, H, C, P, N)$;*
2. *tree-step exactness, requiring every exposed child rational to be obtained from its parent by the displayed left or right Calkin-Wilf rule;*
3. *breadth-first stream discipline, requiring stream windows to read the finite address order in B before exposing the corresponding Q row;*
4. *regular-rational handoff, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer to pass through S and R before the real-name seal E is public;*
5. *non-escape, stating that quotient rational equality, ambient host enumeration, arbitrary representative choice, signed-rational coverage, and* $\mathsf{Real}^{\uparrow}$ *completeness are absent from the packet.*

Proof. Carrier admission is projection from the finite tuple $W = (U, Q, T, B, S, R, E, H, C, P, N)$. Each coordinate is a displayed BEDC row, and the local naming row N records that exactly these coordinates form the public $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ surface.

Tree-step exactness is local to T. A left edge replaces a/b by $a/(a+b)$, while a right edge replaces a/b by $(a+b)/b$; both operations stay inside positive unary numerator-denominator data and therefore feed the same $\mathsf{Rat}^{\uparrow}$ row format.

Breadth-first stream discipline is supplied by B and S. The schedule row orders finite addresses before they are exposed as rational-stream windows, so the stream cannot be read as a host enumeration detached from the displayed tree.

Finally, R and E give the downstream handoff. Componentwise transport, continuation replay, provenance, and local naming are exactly H, C, P, N, so no additional quotient equality, representative selector, signed-rational cover, or ambient completeness coordinate remains available. □

Theorem 1474.3 (Calkin-Wilf positive-rational coverage)**.** *For every accepted* $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ *carrier, the public stream row S covers the positive* $\mathsf{Rat}^{\uparrow}$ *rows generated from $1/1$ by finite Calkin-Wilf left and right edges, and every public coverage read factors through the displayed address, tree-step, breadth-first schedule, transport, continuation, provenance, and local naming rows. The coverage claim is only for the generated positive numerator-denominator surface; it does not identify quotient-equal rational presentations, choose representatives in a host enumeration, cover signed rationals, or assert completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier and start from the root rational $1/1$ in the Q row. A finite address in U is read by repeated use of the displayed tree-step row T, so every addressed output remains a positive rational presentation.

The breadth-first row B then orders these addressed presentations before they enter the rational-stream row S. Thus a coverage read of S is a read of finite Calkin-Wilf address data, not a read of an external list of host rationals.

The statement is intentionally presentation-level. Since the packet has no quotient rational equality coordinate and no selector for preferred host representatives, it covers exactly the positive generated surface available through $U, Q, T, B, S, H, C, P, N$. □

Theorem 1474.4 (Calkin-Wilf Real completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{RationalStream}^{\uparrow}$ *row* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *before the handoff row* E *can be exposed. Consequently the enumeration can feed L10 rational streams and regular Cauchy real-name readback without exporting quotient rational equality, an ambient host enumeration, arbitrary choice of representatives, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a real-facing consumer request data from an accepted carrier. The only stream coordinate in the packet is S, so the consumer first sees a finite window of positive rational presentations generated by the Calkin-Wilf tree and ordered by B.

The next completion-facing coordinate is R. It is the regular-rational readback row attached to the same stream window, and it is the only row that connects the enumeration surface to the regular Cauchy route.

Only after S and R have been consumed can E expose a real-name handoff. The structural rows H, C, P, N replay and name this same finite path, so the consumer receives a rational-stream source for L10 completion work without obtaining quotient equality, host enumeration data, arbitrary representatives, or a completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CalkinWilfEnumeration}^{\uparrow}}$.

Constructive story: A $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ packet is generated from unary breadth-first addresses, positive rational numerator-denominator rows, displayed Calkin-Wilf tree steps, a rational-stream window, regular-rational readback, a real-name handoff row, componentwise transport, continuation replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CalkinWilfEnumeration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1474.1, Definition 1474.2, Definition 1474.3, and Definition 1474.4 seed the finite positive-rational enumeration packet feeding RationalStream, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CalkinWilfEnumeration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim quotient rational equality, ambient host enumeration, arbitrary choice of representatives, signed-rational coverage, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, tree-step exactness, breadth-first stream discipline, regular-rational handoff, and non-escape for the same finite Calkin-Wilf packet.

Chapter 1475

A Concrete Naming Certificate for $\mathsf{CantorSpace}^{\uparrow}$

Following Brouwer and Bishop–Bridges constructive analysis, the $\mathsf{CantorSpace}^{\uparrow}$ packet records the traditional Boolean sequence space as finite prefix data over the L10 countable-name substrate. It consumes the two-endpoint discipline of chapter 32, the finite window schedule of chapter 126, and the finite-data spine discipline of chapter 36. The carrier is a displayed BHist packet of $\mathsf{Bool}^{\uparrow}$ observations over unary $\mathsf{StreamName}^{\uparrow}$ windows; it is not a quotient of infinite streams, a host function space $\mathsf{Nat}^{\uparrow} \to \mathsf{Bool}^{\uparrow}$, a completed infinite product, a choice-selected branch, or an open-cover compactness theorem.

Definition 1475.1 (CantorSpace carrier)**.** *A* $\mathsf{CantorSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, W, B, L, E, H, K, P, N)$$

with the following visible rows:

1. *S is the unary* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose entries are the finite positions currently requested by the consumer;*
2. *W is the finite prefix-window row, recording the initial segment and any explicitly named subwindow inside that segment;*
3. *B is the* $\mathsf{Bool}^{\uparrow}$ *observation ledger, assigning displayed Boolean rows to the positions of W;*
4. *L is the* $\mathsf{List}^{\uparrow}$ *prefix-spine row that keeps the observed Boolean entries as finite ordered data;*
5. *E is the endpoint-exclusion row saying that every public bit is read through the* $\mathsf{Bool}^{\uparrow}$ *two-row surface rather than through a hidden host Boolean;*
6. *H is the componentwise* hsame *transport row for S, W, B, L, E;*
7. *K is the displayed* Cont *replay row for already accepted prefix windows and Boolean rows;*
8. *P is the* Pkg *provenance over* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no host-level function equality, no completed infinite product, no branch chosen by countable choice, and no compactness principle.

Definition 1475.2 (CantorSpace classifier)**.** *Two accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carriers*

$$C = (S, W, B, L, E, H, K, P, N) \quad \textit{and} \quad C' = (S', W', B', L', E', H', K', P', N')$$

are classified as the same at the local surface exactly when their $\mathsf{StreamName}^{\uparrow}$ *schedules request the same finite prefix window and the* $\mathsf{Bool}^{\uparrow}$ *observations agree componentwise on that common window. The comparison is mediated by the* $\mathsf{List}^{\uparrow}$ *spine rows, transported by* H, H'*, and replayed only through* K, K'*. No classifier clause compares host functions, quotient streams, or all infinite coordinates at once.*

Theorem 1475.3 (CantorSpace NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$ *surface of Definition 1475.1 consists of five finite obligations:*

1. *carrier admission for the displayed packet* $C = (S, W, B, L, E, H, K, P, N)$*;*
2. *classifier transport for common finite-prefix* $\mathsf{Bool}^{\uparrow}$ *equality under the same* $\mathsf{StreamName}^{\uparrow}$ *schedule;*
3. *prefix stability, requiring every shorter requested window inside* W *to reuse the matching visible Boolean rows already recorded in* B *and* L*;*
4. *continuation compatibility, requiring every replayed read to pass through* H, K, P, N *before it reaches a downstream L10 consumer;*
5. *non-escape, excluding quotient streams, host function equality, completed infinite products, choice-selected branches, and compactness claims.*

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier namecert obligation surface`

Proof. Carrier admission is projection of the nine displayed coordinates in Definition 1475.1. The schedule, prefix window, Boolean ledger, and list spine rows are finite BHist data.

Classifier transport uses Definition 1475.2. Once the common $\mathsf{StreamName}^{\uparrow}$ window is fixed, equality is read only as componentwise $\mathsf{Bool}^{\uparrow}$ agreement along the displayed $\mathsf{List}^{\uparrow}$ spine and is then transported by H.

Prefix stability follows because a shorter requested window is a subrow of the same displayed prefix row W. Its Boolean entries are already present in B and in the ordered spine L, so Cont may replay them without adding an infinite stream coordinate.

The non-escape clause is inspection of the carrier and classifier. The packet has rows for finite schedules, finite Boolean observations, transport, continuation, provenance, and the local $\mathsf{NameCert}$ row, and has no coordinate for quotient streams, host functions, infinite products, selected branches, or compactness witnesses. □

Theorem 1475.4 (CantorSpace binary-prefix obligations)**.** *For every accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier* $C = (S, W, B, L, E, H, K, P, N)$*, the binary-prefix obligation surface exposes exactly finite* $\mathsf{Bool}^{\uparrow}$ *digit rows,* $\mathsf{List}^{\uparrow}$ *prefix rows,* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$ *row. Every finite prefix read by a downstream consumer factors through those displayed rows before the consumer can inspect a cylinder, endpoint, or classifier comparison. The obligation surface contains no quotient-stream equality, choice principle, or host infinite binary sequence.*

Proof. Proceed by finite prefix induction over the displayed $\mathsf{List}^{\uparrow}$ spine.

In the empty-prefix case, the consumer reads only the schedule row S, the empty prefix window inside W, the empty ordered prefix in L, and the structural rows H, K, P, N. No Boolean digit, quotient stream, or host sequence coordinate is available.

For the cons step, assume the shorter prefix read factors through S, W, B, L, H, K, P, N. The next digit is supplied by the corresponding finite $\mathsf{Bool}^{\uparrow}$ row in B at the next position named by S and ordered by L.

Transport by H and replay by K preserve the shorter read and the next digit as one displayed packet. Provenance P and local naming N then present the extended prefix to the downstream consumer.

Thus every finite binary prefix is generated by repeated inspection of the displayed Boolean and list rows. Since the carrier has no quotient stream, choice operator, or host infinite binary sequence coordinate, none can enter the prefix obligation surface. □

Theorem 1475.5 (CantorSpace finite-prefix cylinder stability)**.** *Let $C = (S, W, B, L, E, H, K, P, N)$ be an accepted $\mathsf{CantorSpace}^{\uparrow}$ carrier. Any finite cylinder read determined by a prefix subwindow of W factors through the same $\mathsf{StreamName}^{\uparrow}$ schedule row S, the same $\mathsf{Bool}^{\uparrow}$ observation ledger B, the same $\mathsf{List}^{\uparrow}$ spine L, and the same H, K, P, N transport, continuation, provenance, and local naming rows. The cylinder read exports no extra infinite tail, no quotient-stream equality, no host function extensionality, and no compactness argument.*

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier finite prefix cylinder stability`

Proof. Choose a prefix subwindow of W. Since W is a finite row of the accepted carrier, the requested positions are already part of the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S.

The Boolean values on that subwindow are precisely the corresponding entries of B, ordered by the finite $\mathsf{List}^{\uparrow}$ spine L. Transport by H preserves the componentwise rows, and Cont replay through K reuses only those rows.

Thus every cylinder read is exhausted by S, W, B, L, H, K, P, N. The excluded objects are not coordinates of the carrier, so the read cannot produce an infinite tail, quotient equality, host-function extensionality, or compactness data. □

Theorem 1475.6 (CantorSpace prefix projection stability)**.** *Let $K = (S, W, B, L, E, H, K_0, P, N)$ be an accepted $\mathsf{CantorSpace}^{\uparrow}$ carrier. Any restriction of a public read to a shorter finite prefix is again a read through the displayed rows $S, W, B, L, E, H, K_0, P, N$: the schedule row S and window row W choose the visible positions, the $\mathsf{Bool}^{\uparrow}$ row B supplies endpoint values, and the $\mathsf{List}^{\uparrow}$ spine L records the prefix order. The projection contains no host infinite stream, no quotient equality of functions, no branch choice, and no compactness principle.*

Proof. Start with the accepted carrier and project its finite prefix spine. The shorter prefix is obtained by reading an initial segment of L and the corresponding entries of S and W.

For each retained position, the endpoint value is still read from B. Componentwise transport by H and replay through K_0 preserve only the retained rows, while P and N keep the same provenance and local certificate.

Thus prefix projection is closed inside the displayed carrier. Since no coordinate names an infinite stream, a quotient-function equality, a branch selector, or compactness data, none of those objects can appear in the projected read. □

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier prefix projection stability`

Theorem 1475.7 (CantorSpace prefix cylinder no-confusion)**.** *Let $K = (S, W, B, L, E, H, K_0, P, N)$ and $K' = (S', W', B', L', E', H', K'_0, P', N')$ be accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carriers whose finite cylinder reads are projected to the same certified unary index window. If the displayed* $\mathsf{List}^{\uparrow}$ *prefix readbacks first disagree at a certified unary index i and the corresponding* $\mathsf{Bool}^{\uparrow}$ *rows in B and B' have separated tags at i, then the two* $\mathsf{StreamName}^{\uparrow}$ *carrier reads are not classifier-equal on that finite window. The conclusion is only the finite-window failure of classifier equality for the displayed prefixes; it is not an inequality of ambient infinite sequences.*

Proof. Project both carrier reads to the common finite window using Definition 1475.6. The projected data are exactly the visible schedule rows, the two finite $\mathsf{List}^{\uparrow}$ prefix spines, and the corresponding $\mathsf{Bool}^{\uparrow}$ ledger entries.

At the first certified unary index i where the prefix readbacks disagree, $\mathsf{List}^{\uparrow}$ readback identifies the two compared positions as the same finite slot. The separated $\mathsf{Bool}^{\uparrow}$ tags at that slot cannot be transported into the same componentwise observation row.

StreamName pointwise classifier stability therefore blocks equality on the projected window: any classifier equality for the two carrier reads would have to identify the two Boolean rows at i. This contradicts the displayed tag separation, so no classifier-equality certificate exists on the finite window. □

Theorem 1475.8 (CantorSpace Bool cylinder endpoint determinacy)**.** *For an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier $K = (S, W, B, L, E, H, K_0, P, N)$, each cylinder endpoint visible to a downstream consumer is determined by the finite* $\mathsf{Bool}^{\uparrow}$ *ledger B along the positions ordered by L. If two consumer reads inspect the same prefix coordinates through S, W, L and transport by the same H, K_0, P, N rows, then their endpoint rows agree exactly as displayed by B; no endpoint is chosen from an ambient infinite product.*

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier bool cylinder endpoint determinacy`

Proof. Read the shared prefix coordinates from S, W, L. These rows specify which positions are visible and in which finite order they are consumed.

The endpoint value at each visible position is then the corresponding row of B. If two reads use the same visible coordinates and the same transport and replay data, they reach the same finite Bool ledger entries.

The structural rows H, K_0, P, N preserve this ledger route. There is no carrier coordinate for an infinite product endpoint, a choice-selected branch, or a host stream function, so endpoint determinacy is the finite determinacy of B over L. □

Theorem 1475.9 (CantorSpace StreamName prefix consumer exhaustion)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{List}^{\uparrow}$ *consumer of an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier factors through the finite rows $S, W, B, L, E, H, K_0, P, N$. The consumer may restrict, transport, and replay a displayed prefix cylinder, but after the rows $S, W, B, L, E, H, K_0, P, N$ are exhausted it has no further route to an infinite stream tail, a quotient of branches, a fan-theorem compactness object, or countable-choice data.*

Proof. Project the consumer read through the $\mathsf{StreamName}^{\uparrow}$ rows S, W and the $\mathsf{List}^{\uparrow}$ row L. These rows are the only coordinates that expose positions and prefix order.

The Bool observations are then consumed through B, with any endpoint and cylinder data carried by E. Transport and continuation replay remain componentwise through H and K_0.

Once P and N record provenance and local naming, every carrier coordinate has been inspected. The consumer is therefore exhausted by the displayed finite packet and cannot continue to an unlisted stream tail, quotient branch, compactness object, or choice schedule. □

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier streamname prefix consumer exhaustion`

Theorem 1475.10 (CantorSpace StreamName-Bool scope)**.** *Every public read exported by an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier reaches the L10 countable-name substrate only through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}.$$

The route binds the schedule, Boolean endpoint rows, finite prefix spine, and local certificate before any downstream consumer may inspect the packet. It does not claim Real completeness, quotient stream completeness, open-cover compactness, fan-theorem compactness, or choice of an infinite branch.

Lean4 ✓ `BEDC.Derived.CantorSpaceUp.CantorSpaceStreamNameBoolScope`

Proof. Project a public consumer read from the accepted carrier. The requested positions are first visible in the $\mathsf{StreamName}^{\uparrow}$ schedule row S and prefix row W.

Each visible value is then read through the $\mathsf{Bool}^{\uparrow}$ ledger B and ordered by the $\mathsf{List}^{\uparrow}$ spine L. The rows H and K only transport and replay that finite data, while P and N record the provenance and local certificate.

Therefore the public route is exactly the displayed finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, and $\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$. No coordinate in the packet supplies Real completeness, stream quotient completeness, open-cover compactness, fan-theorem compactness, or a choice-selected infinite branch. □

Theorem 1475.11 (CantorSpace choice refusal)**.** *Let* $K = (S, W, B, L, E, H, K_0, P, N)$ *be an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier. Every public finite-prefix read through the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, *and* $\mathsf{List}^{\uparrow}$ *surfaces cited in chapter 126, chapter 32, and chapter 36 is already determined by the displayed rows* $S, W, B, L, E, H, K_0, P, N$*. The read cannot import a selector for an infinite branch, a countable-choice schedule, host function extensionality, or a fan-theorem compactness witness.*

Proof. Project the accepted carrier to its nine displayed rows. The schedule row S, prefix window W, Boolean ledger B, and list spine L are the only rows that expose prefix positions and endpoint values.

Restrict the public read to the requested finite window. Transport by H and replay through K_0 preserve exactly those displayed rows, while P and N record the provenance and local name.

Since no carrier coordinate names an infinite branch selector, a countable-choice schedule, host function equality, or a fan-theorem object, the read has no route to any such datum. The refusal is therefore a carrier projection followed by prefix restriction. □

Theorem 1475.12 (CantorSpace cylinder coverage)**.** *For an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier* $K = (S, W, B, L, E, H, K_0, P, N)$*, every finite cylinder visible to a downstream consumer is covered by the displayed prefix rows* $S, W, B, L, E, H, K_0, P, N$*. The* $\mathsf{StreamName}^{\uparrow}$ *row chooses the requested positions, the* $\mathsf{Bool}^{\uparrow}$ *row supplies the endpoint entries, and the* $\mathsf{List}^{\uparrow}$ *row orders the finite prefix; no cylinder read requires an ambient product space or an open-cover compactness argument.*

Proof. Begin with a finite cylinder read. Carrier projection supplies the schedule S, the prefix window W, the Boolean ledger B, and the ordered spine L as the visible cylinder data.

Now restrict to the cylinder subwindow. The retained positions are still rows of S and W, and their endpoint values are still the corresponding entries of B ordered by L.

The remaining rows E, H, K_0, P, N record endpoint exclusion, componentwise transport, continuation replay, provenance, and local naming. Thus the cylinder is covered inside the displayed finite packet and does not appeal to an ambient infinite product or to open-cover compactness. □

Theorem 1475.13 (CantorSpace prefix-window composition)**.** *Let $K = (S, W, B, L, E, H, K_0, P, N)$ be an accepted* $\mathsf{CantorSpace}^{\uparrow}$ *carrier. If a public read first restricts to a finite prefix subwindow and then restricts again to a shorter subwindow, the composite read is the same finite* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{Bool}^{\uparrow}$–$\mathsf{List}^{\uparrow}$ *read obtained by restricting S, W, B, L once to the shorter window and carrying the same E, H, K_0, P, N rows. The composition exports no additional stream tail, quotient branch, or Real completion datum.*

Proof. Project the carrier to $S, W, B, L, E, H, K_0, P, N$. The first prefix restriction keeps a subrow of the schedule and window rows and the corresponding Boolean entries ordered by the list spine.

Apply the second prefix restriction to that finite subrow. Because both restrictions are prefix restrictions of the same displayed window, the result is exactly the shorter subrow of S, W, B, L.

Transport and replay remain the original componentwise rows H and K_0, and the same provenance and local naming rows P, N certify the composite read. The composition therefore stays within finite prefix data and cannot expose an extra stream tail, quotient branch, or Real completion witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1475.1, Definition 1475.2, Definition 1475.3, Definition 1475.5, Definition 1475.6, Definition 1475.8, Definition 1475.9, and Definition 1475.10 supply the finite-prefix obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorSpace}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.CantorSpaceUp.CantorSpaceCarrier\_namecert\_obligation\_surface`

Bridge status: paperBridge

Not claimed: The chapter does not close quotient streams, host function equality, completed infinite products, Real completeness, choice-selected branches, open-cover compactness, fan-theorem compactness, or Lean bridge checking.

Upgrade path: No further paper-side closure step remains; machine bridge checking is the remaining formal-axis route.

Constructive story: The construction starts from finite unary StreamName windows, records Bool observations along a List prefix spine, and exposes only displayed hsame, Cont, Pkg, and local NameCert rows to downstream L10 consumers.

Chapter 1476

A Concrete Naming Certificate for $\mathsf{CauchyCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ packet names the finite criterion that licenses admission of a regular Cauchy route into the Real completion seal. It sits between the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, the completion-modulus route of chapter 1469, and the terminal Real seal of chapter 38. The certificate is a finite admission surface for Real completion consumers, not a quotient stream, a selected host limit, or an ambient completeness principle.

Definition 1476.1 (Cauchy completion criterion carrier)**.** *A* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule through which every sequence read is windowed.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy handoff read from the displayed stream windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational thresholds, comparison rows, and dyadic scale used by the handoff.*
4. *M is the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row that sends each precision request to a displayed finite threshold for the same windows and tolerances.*
5. *Q is the local finite admission criterion, asserting that the stream windows, regular handoff, dyadic ledger, and modulus row are mutually compatible before any Real-facing seal is read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-admission row reached only after S, R, D, M, Q have been displayed.*
7. *H stores componentwise* hsame *transport for the stream schedule, regular handoff, dyadic ledger, modulus row, criterion row, seal row, continuation, provenance, and local naming.*
8. *C stores displayed* Cont *replay from finite stream windows to the regular handoff, from the handoff through the dyadic and modulus ledgers to the criterion row, and from the criterion row to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It contains no coordinate for quotient streams, selected host limits, countable choice, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionCriterionUp.CauchyCompletionCriterionTasteGate single carrier alignment`

Theorem 1476.2 (Cauchy completion criterion NameCert obligations)**.** *For every accepted* CauchyCompletionCrite *carrier* $K = (S, R, D, M, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *finite-window handoff, stating that* R *is read only through the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *and the stored continuation rows;*
3. *tolerance-modulus compatibility, stating that* D *and* M *use the same displayed thresholds, dyadic scales, and precision requests for the regular rational handoff;*
4. *criterion admission, stating that* Q *is the sole local row that licenses passage from* S, R, D, M *to a Real-facing completion read;*
5. *Real seal non-escape, stating that* E *exports no quotient streams, selected host limits, countable-choice row, host real equality, or ambient completeness witness.*

Proof. Project the carrier of Definition 1476.1. The stream schedule, regular handoff, dyadic ledger, completion modulus, criterion row, Real seal, transport, continuation, provenance, and local naming rows are exactly the classifier-visible coordinates.

The finite-window handoff is forced by the source rows. A consumer can read the regular rational handoff R only after it has read the displayed stream schedule S and the continuation routes that replay the relevant finite windows.

The tolerance and modulus faces are tied to the same requests. The dyadic ledger D records the comparison and scale data, while the Bishop completion modulus row M records the threshold chosen for each precision request; both are consumed before the criterion row Q is available.

The criterion row is the only admission point to the seal. Since the carrier has no coordinate for quotient streams, selected host limits, countable choice, host real equality, or ambient completeness, the five listed obligations exhaust the local NameCert surface. □

Theorem 1476.3 (Cauchy completion criterion Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier* $K = (S, R, D, M, Q, E, H, C, P, N)$ *factors through the displayed route*

$$S \longmapsto R \longmapsto D, M \longmapsto Q \longmapsto E$$

together with the stored hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. The consumer receives no quotient stream, selected host limit, countable-choice row, host real equality, or ambient completeness principle.*

Proof. Begin with a Real-facing consumer and project the accepted carrier. The only sequence data available to the consumer is the finite schedule S together with the regular rational handoff R read from that schedule.

Next read the threshold data. The dyadic ledger D and Bishop completion modulus row M supply the displayed tolerance and threshold information used by the handoff; the continuation row C replays this same route without adding a second source of convergence data.

The admission row Q is then the sole bridge from the regular Cauchy data to the seal row E. The transport and provenance rows H and P preserve the same ten-row packet, and the local naming row N records that no additional coordinate has been admitted.

Thus every Real-facing read factors through $S, R, D, M, Q, E, H, C, P, N$. The forbidden outputs are absent by inspection of the carrier, so the consumer cannot obtain quotient streams, selected host limits, countable choice, host real equality, or ambient completeness from this certificate. □

Theorem 1476.4 (Cauchy completion criterion finite-window handoff)**.** *For every accepted* CauchyCompletionCrite *carrier $K = (S, R, D, M, Q, E, H, C, P, N)$, each read of the regular rational row R factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S and the continuation replay stored in C. No consumer of R can bypass S by reading a hidden stream tail, a selected subsequence, a countable-choice schedule, or an ambient host sequence object.*

Proof. Project the carrier row of Definition 1476.1. The only sequence-facing coordinate is the finite schedule S, and the only regular-rational coordinate is the handoff row R.

The replay row C records the allowed continuation from the displayed finite windows to R. Since the classifier compares accepted carriers through the ten displayed rows, any use of R must preserve the same S-indexed route under the componentwise transport row H and the provenance row P.

There is no carrier coordinate for a hidden tail, selected subsequence, countable-choice schedule, or host sequence object. Thus every R read factors through S and C, and no bypass is available inside the local certificate. □

Theorem 1476.5 (Cauchy completion criterion tolerance-modulus compatibility)**.** *For every accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier $K = (S, R, D, M, Q, E, H, C, P, N)$, the dyadic tolerance ledger D and the Bishop completion modulus row M answer the same precision request for the same finite stream windows and regular rational handoff. The dyadic threshold route consumed by D is the threshold route selected by M before the criterion row Q is visible.*

Proof. Read D and M from the displayed packet. By definition of the carrier, D records the rational thresholds, comparison rows, and dyadic scale used by the handoff, while M sends each precision request to a displayed finite threshold for the same windows and tolerances.

The rows are consumed before Q. The continuation replay C supplies one route from the windows and handoff through the dyadic and modulus ledgers to the criterion row, and the componentwise transport row H keeps that route stable under accepted carrier comparison.

Consequently the two rows cannot answer different requests or use different threshold routes inside this certificate. A consumer reaching Q has already consumed the shared request, the same finite windows, the same regular rational handoff, and the same dyadic threshold route. □

Theorem 1476.6 (Cauchy completion criterion admission)**.** *For every accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier $K = (S, R, D, M, Q, E, H, C, P, N)$, the row Q is the sole local admission row from the displayed data S, R, D, M to the Real seal row E. Any* $\mathsf{Real}^{\uparrow}$*-facing read that reaches E must pass through Q after the finite-window handoff and tolerance-modulus compatibility rows have been consumed.*

Proof. Project the carrier and isolate the route to the seal. The displayed sequence schedule S, regular handoff R, dyadic ledger D, and modulus row M are upstream of the criterion row Q, while the Real seal row E is downstream of Q.

The continuation row C records this order as a replay route, and the provenance row P records it as part of the local package. The transport row H preserves the same row order under accepted carrier comparison, so a consumer cannot replace Q by another local admission coordinate.

The carrier has no second admission row between S, R, D, M and E. Therefore Q is the sole local admission row, and every Real-facing read of E passes through Q after the finite-window and tolerance-modulus obligations have been met. □

Theorem 1476.7 (Cauchy completion criterion regular-tail induction)**.** *Every accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier routes a tail-refinement request by induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows before exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The induction consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, local* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ $\mathsf{NameCert}$ *row, componentwise* hsame *transport,* Cont *replay, and* Pkg *provenance, and exports no quotient stream, selected limit, host equality, or ambient completeness.*

Proof. Project the accepted carrier $K = (S, R, D, M, Q, E, H, C, P, N)$ from Definition 1476.1. The only tail schedule available to a refinement request is the finite $\mathsf{StreamName}^{\uparrow}$ window row S, and the regular rational readback R is not visible before that schedule is traversed.

Induct over the finite visible tail-window schedule. At each window, use the dyadic tolerance ledger D and the modulus row M to keep the requested threshold inside the displayed packet, then use the continuation row C to replay the same route without adding a second convergence source.

The existing NameCert obligations provide carrier admission and regular-rational visibility for the handoff through R. The local row N, transport row H, and provenance row P preserve the same displayed packet during the induction.

Finally apply the Real seal non-escape result of Definition 1476.3. The terminal seal E is read only after the displayed induction route has reached Q, so the exported surface contains no quotient stream, selected limit, host equality, or ambient completeness. □

Theorem 1476.8 (Cauchy completion criterion tail-modulus criterion)**.** *Every accepted* $\mathsf{CauchyCompletionCriterion}$ *carrier routes a precision request through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row, criterion row, and* $\mathsf{Real}^{\uparrow}$ *seal before any completion consumer reads it. The route is the displayed finite path*

$$S \longmapsto R \longmapsto D, M \longmapsto Q \longmapsto E$$

together with the stored hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row; it exports no quotient stream, selected host limit, host real equality, countable choice, or ambient completeness principle.*

Proof. Project the accepted carrier $K = (S, R, D, M, Q, E, H, C, P, N)$ from Definition 1476.1. The precision request first reaches the finite stream schedule S, and the regular rational handoff R is readable only as the displayed readback from those windows.

Use the NameCert obligations of Definition 1476.2. They identify tolerance-modulus compatibility as a local obligation of the same packet: the dyadic ledger D and Bishop completion modulus row M must answer the same threshold and precision data before the criterion row Q can admit a completion read.

Apply regular-tail induction from Definition 1476.7. Tail refinement stays inside the same StreamName schedule and reuses the same dyadic tolerance ledger, regular handoff, transport, continuation replay, provenance, and local naming rows.

Finally apply Real seal non-escape from Definition 1476.3. Since the terminal seal E is visible only after S, R, D, M, Q have been traversed, a completion consumer cannot read quotient streams,

selected host limits, host real equality, countable choice, or ambient completeness from the packet. □

Theorem 1476.9 (Cauchy completion criterion window exhaustion)**.** *Every accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier routes any completion-facing request through the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *before the dyadic ledger* D*, modulus row* M*, criterion row* Q*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *are visible. Thus a completion consumer receives only the displayed finite-window route* $S, R, D, M, Q, E, H, C, P, N$*, not a hidden stream tail, selected host limit, quotient equality, countable-choice schedule, or ambient completeness principle.*

Proof. Project the accepted carrier $K = (S, R, D, M, Q, E, H, C, P, N)$ and read the NameCert obligations of Definition 1476.2. Carrier admission exposes exactly the ten displayed rows, while the finite-window handoff obligation places S before every read of the regular rational handoff R.

Apply the tail-modulus criterion of Definition 1476.8. The dyadic ledger D and modulus row M answer the same precision request after the displayed windows and readback have been consumed, and the criterion row Q is available only after that finite route has been traversed.

Finally use Real seal non-escape from Definition 1476.3. Since the seal row E exports no quotient stream, selected host limit, host equality, countable choice, or ambient completeness witness, exhaustion of the displayed windows is the only completion-facing route available inside the certificate. □

Theorem 1476.10 (Cauchy completion criterion cofinal-window induction)**.** *Let* $K = (S, R, D, M, Q, E, H, C, P, N$ *be an accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier. Any cofinal refinement of a completion-facing precision request must reuse the same finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row* M*, criterion row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*. The induction is over displayed finite windows only; it cannot introduce a second stream source, selected limit, quotient equality, countable-choice schedule, or ambient completeness principle.*

Proof. Project the carrier of Definition 1476.1. The cofinal request has access only to $S, R, D, M, Q, E, H, C, P, N$, so its base window is already one of the displayed StreamName windows.

Apply finite-window handoff from Definition 1476.4. Each refinement step reaches R through the same schedule S and cannot switch to a second stream source.

Use regular-tail induction from Definition 1476.7. The induction step preserves the same dyadic ledger, regular readback, transport, continuation replay, provenance, and local naming rows.

Apply the tail-modulus criterion of Definition 1476.8 and window exhaustion of Definition 1476.9. The route through D, M, Q remains attached to the finite window schedule at every cofinal stage.

Finally close by Real seal non-escape from Definition 1476.3. The seal row E exports no selected limit, quotient equality, countable-choice schedule, or ambient completeness witness, so the induction remains carrier-local. □

Theorem 1476.11 (Cauchy completion criterion scoped source exhaustion)**.** *Every scoped consumer of an accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row, criterion row* Q*,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. Within this scoped surface, no second completion source, hidden stream tail, selected limit, quotient equality, countable-choice schedule, host real equality, or ambient completeness principle is exposed.*

Proof. Project the accepted carrier $K = (S, R, D, M, Q, E, H, C, P, N)$ from Definition 1476.1. The carrier inventory names the finite window source S, regular handoff R, dyadic tolerance ledger D, completion modulus M, criterion row Q, Real seal E, transport H, replay C, provenance P, and local name N as the whole scoped packet.

Apply the finite-window handoff of Definition 1476.4 and tolerance-modulus compatibility of Definition 1476.5. These force every regular read through S and keep D and M attached to the same displayed precision route.

Apply criterion admission from Definition 1476.6, regular-tail induction from Definition 1476.7, and the tail-modulus criterion from Definition 1476.8. The only completion-facing route then passes from the finite windows through the regular handoff, dyadic ledger, modulus row, and Q.

Finally use window exhaustion from Definition 1476.9 and Real seal non-escape from Definition 1476.3. The structural rows H, C, P, N preserve the same route without adding data, so the scoped consumer has no second completion source outside the displayed packet. □

Theorem 1476.12 (Cauchy completion criterion scoped route)**.** *Let $K = (S, R, D, M, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier. Any downstream completion consumer,* $\mathsf{Real}^{\uparrow}$ *consumer, or regular-Cauchy consumer of K must reuse exactly the finite-window handoff, tolerance-modulus compatibility, criterion admission, and Real seal non-escape rows named in this chapter. Such a consumer cannot introduce a second completion source, quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.*

Consequently the scoped route for K is the displayed BEDC kernel path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Proof. Project the accepted carrier and read the NameCert obligations of Definition 1476.2. The carrier row exposes exactly $S, R, D, M, Q, E, H, C, P, N$, and the obligation surface names the finite-window handoff, tolerance-modulus compatibility, criterion admission, and Real seal boundary as the rows a completion-facing consumer may use.

Apply the finite-window handoff of Definition 1476.4. A downstream consumer reaches the regular rational handoff R only through the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S and the stored continuation replay, so no second stream source or hidden tail can be read.

Next apply tolerance-modulus compatibility from Definition 1476.5 and criterion admission from Definition 1476.6. The dyadic ledger D and modulus row M answer the same precision route before Q is visible, and Q is the sole local admission row from S, R, D, M to the seal row E.

Use regular-tail induction from Definition 1476.7 and the tail-modulus criterion of Definition 1476.8. These results keep tail refinement and precision routing inside the same finite window schedule, regular handoff, dyadic tolerance ledger, completion modulus row, transport, continuation replay, provenance, and local naming row.

Finally apply window exhaustion from Definition 1476.9 and Real seal non-escape from Definition 1476.3. The only terminal route is the displayed BEDC kernel path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, and the forbidden completion sources are absent from the carrier. □

Theorem 1476.13 (Cauchy completion criterion public export)**.** *Every accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier $K = (S, R, D, M, Q, E, H, C, P, N)$ exports exactly the public completion-object surface $S, R, D, M, Q, E, H, C, P, N$ to* $\mathsf{Real}^{\uparrow}$*-facing consumers. It exports no quotient stream, selected host limit, countable-choice row, ambient completeness principle, or bridge certificate.*

Proof. Start with the NameCert obligations of Definition 1476.2. They identify carrier admission, finite-window handoff, tolerance-modulus compatibility, criterion admission, and Real seal non-escape as the whole local obligation surface for the ten-row packet.

Apply the finite-window handoff of Definition 1476.4, tolerance-modulus compatibility of Definition 1476.5, and criterion admission of Definition 1476.6. These results force every completion-facing read to traverse S, R, D, M, Q before the seal row is visible.

Use regular-tail induction from Definition 1476.7, the tail-modulus criterion from Definition 1476.8, and window exhaustion from Definition 1476.9. The public route is therefore the displayed finite completion path, with H, C, P, N preserving the same packet by transport, replay, provenance, and local naming.

Finally apply scoped source exhaustion from Definition 1476.11, the downstream scoped route of Definition 1476.12, and Real seal non-escape from Definition 1476.3. The carrier has no coordinate for quotient streams, selected host limits, countable-choice rows, ambient completeness, or bridge certificates, so exactly $S, R, D, M, Q, E, H, C, P, N$ is exported. □

Theorem 1476.14 (Cauchy completion criterion mature consumer route)**.** *Every mature Real or Completion consumer of an accepted* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *carrier*

$$K = (S, R, D, M, Q, E, H, C, P, N)$$

factors through the finite-window handoff, tolerance-modulus compatibility, criterion admission, Real seal non-escape, scoped consumer route, downstream consumer route, and public object surface of this chapter. The consumer receives no quotient stream, selected host limit, countable choice, host equality, ambient completeness, or bridge credential.

Proof. Start with finite-window handoff, Definition 1476.4. It forces every mature consumer to reach the regular rational handoff R through the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S and the stored continuation replay.

Next apply tolerance-modulus compatibility and criterion admission, Definition 1476.5 and Definition 1476.6. These route the dyadic ledger D and completion modulus row M through the same precision request before the criterion row Q admits a Real-facing seal read.

Use Real seal non-escape from Definition 1476.3. The seal row E is downstream of S, R, D, M, Q and cannot export quotient streams, selected host limits, countable choice, host equality, or ambient completeness.

Finally compose scoped source exhaustion, the downstream scoped route, and public export: Definition 1476.11, Definition 1476.12, and Definition 1476.13. These keep H, C, P, N attached to the same packet and expose exactly the public object surface $S, R, D, M, Q, E, H, C, P, N$, with no bridge credential added. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ packet is generated from a finite StreamName schedule, RegSeqRat handoff, DyadicRatCore tolerance ledger, BishopCompletionModulus row, local completion-admission criterion, RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionCriterion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1476.1, Definition 1476.2, Definition 1476.3, Definition 1476.4, Definition 1476.5, Definition 1476.6, Definition 1476.7, Definition 1476.8, Definition 1476.9, Definition 1476.11, Definition 1476.12, Definition 1476.13, and Definition 1476.14 declare the finite carrier, its NameCert obligation surface, the Real seal boundary, the finite-window handoff, the tolerance-modulus compatibility route, the unique criterion admission row, the tail route, the window exhaustion route, scoped source exhaustion, the downstream scoped consumer lock, the public completion-object export, the mature consumer route, and the checked TasteGate carrier-alignment route cited below.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionCriterion}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.CauchyCompletionCriterionUp.CauchyCompletionCriterionTasteGate\_single\_carrier\_alignm`
Bridge status: none
Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, host real equality, ambient completeness, or bridge checking.
Upgrade path: Bridged closure requires an explicit standard bridge for the public completion criterion.

Chapter 1477

A Concrete Naming Certificate for $\mathsf{DyadicMax}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicMax}^{\uparrow}$ packet records the finite dyadic-grid maximum certificate used when the Real and Completion route must choose the larger endpoint before exposing an interval, radius, or tolerance ledger. It sits over the endpoint rows of chapter 291, the adjacent-grid surface of chapter 433, the lower-grid route of chapter 1416, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a constructive finite comparison surface; it is not trichotomy for arbitrary reals, a quotient-real maximum, a host order decision, ambient completeness, or a selected supremum.

Definition 1477.1 (DyadicMax carrier)**.** *A* $\mathsf{DyadicMax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (x, y, n, m, L, U, H, C, P, N)$$

with the following displayed rows:

1. *x and y are* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows at the shared precision n.*
2. *n is the grid precision row that both endpoint ledgers cite.*
3. *m is the selected endpoint row, displayed as either the transported x row or the transported y row on the same grid.*
4. *L is the finite comparison ledger recording the inspected dyadic inequality that justifies the selected endpoint.*
5. *U is the upper-bound ledger witnessing that both x and y read below or at the selected endpoint m on the displayed grid.*
6. *H records componentwise* hsame *transports for the two endpoint rows, precision row, selected endpoint, comparison ledger, and upper-bound ledger.*
7. *C records displayed* Cont *replay for opening the shared grid, reading the two endpoints, selecting m, exposing L, and exposing U.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicMax}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicRounding}^{\uparrow}$, $\mathsf{DyadicFloor}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMax}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for order trichotomy on arbitrary reals, quotient-real equality, a host maximum function, ambient completeness, or an unbounded supremum argument.

Lean4 ✓ `BEDC.Derived.DyadicMaxUp.DyadicMaxTasteGate single carrier alignment`

Theorem 1477.2 (DyadicMax NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMax}^{\uparrow}$ *carrier* $M = (x, y, n, m, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicMax}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the displayed ten-row packet;*
2. *classifier stability under componentwise* hsame *transport and displayed* Cont *replay of* x, y, n, m, L, U*;*
3. *selected endpoint determinacy, stating that* m *is one of the two displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows at precision* n*;*
4. *upper-bound ledger exactness, requiring the two endpoint-bound reads to be precisely the rows stored in* U *and justified by* L*;*
5. $\mathsf{Real}^{\uparrow}$*-facing non-escape, requiring every real or completion consumer to read the maximum through the displayed dyadic, rounding, floor, absolute-value, transport, continuation, package, and local naming rows.*

No further obligation can be read from the carrier.

Proof. Carrier exposure is projection from Definition 1477.1. The two dyadic endpoint rows, shared precision row, selected endpoint, comparison ledger, upper-bound ledger, transport, replay, provenance, and local name are explicit coordinates of M.

Classifier stability is componentwise. The rows H and C transport and replay exactly the displayed endpoint, precision, selection, comparison, and bound data, so the selected endpoint cannot detach from the grid on which it was certified.

Determinacy and upper-bound exactness are finite ledger reads. The selected row m is named as one of the two endpoint rows, while L records the inspected comparison and U records both upper-bound checks at the same precision.

Finally inspect the carrier inventory. Since no coordinate stores arbitrary real trichotomy, quotient equality, a host maximum, completeness, or a supremum argument, every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives only the displayed finite route. □

Theorem 1477.3 (DyadicMax grid boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicMax}^{\uparrow}$ *carrier reads the maximum only through the rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{DyadicRounding}^{\uparrow},\quad \mathsf{DyadicFloor}^{\uparrow},\quad \mathsf{DyadicAbsoluteValue}^{\uparrow},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

In particular, the consumer can read the selected endpoint m*, the comparison ledger* L*, and the upper-bound ledger* U*, but it cannot read a host maximum operation, quotient-real comparison, trichotomy branch for arbitrary reals, ambient completeness witness, or selected supremum.*

Proof. Start from an accepted carrier $M = (x, y, n, m, L, U, H, C, P, N)$. The endpoint rows x and y are already $\mathsf{DyadicRatCore}^{\uparrow}$ rows at the shared precision n, and the selected row m is displayed on that same grid.

The comparison route is finite. The ledger L records the inspected dyadic comparison, and U records the two upper-bound checks used by the downstream route. The rounding, floor, and

absolute-value sibling surfaces supply only grid normalization and tolerance interpretation around those same rows.

Transport and replay add no analytic choice data. The rows H and C carry the displayed grid data componentwise, while P and N provide provenance and local naming for the same route.

Thus any Real or Completion consumer reaches the maximum through the listed finite rows. The carrier has no remaining coordinate from which a host maximum, quotient-real comparison, trichotomy branch, completeness witness, or selected supremum could be exported. □

Falsifiable prediction: An accepted $\mathsf{DyadicMax}^{\uparrow}$ consumer that exposes a $\mathsf{Real}^{\uparrow}$ or completion-facing maximum without reading the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, selected endpoint, comparison ledger, upper-bound ledger, and structural rows refutes the carrier surface.

Independence witness: dyadicratcore, dyadicrounding, dyadicfloor, dyadicabsolutevalue, real: no listed sibling supplies the whole ten-row packet $M = (x, y, n, m, L, U, H, C, P, N)$, because each sibling contributes only endpoint, grid-normalization, lower-grid, absolute-value, or seal-facing rows while this chapter binds the selected maximum endpoint to comparison, upper-bound, transport, replay, provenance, and local naming data.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMax}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMax}^{\uparrow}$ packet is generated from two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows at a shared precision, a selected endpoint row, a finite comparison ledger, an upper-bound ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMax}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1477.1, Definition 1477.2, and Definition 1477.3 seed the finite dyadic-grid maximum boundary for Real and Completion consumers over DyadicRatCore, DyadicRounding, DyadicFloor, DyadicAbsoluteValue, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMax}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary real trichotomy, quotient-real maximum, host order decision, ambient completeness, selected supremum, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove carrier exposure, classifier stability, selected endpoint determinacy, upper-bound ledger exactness, and the Real-facing finite-grid boundary.

Chapter 1478

A Concrete Naming Certificate for $\mathsf{CauchyCondensationTest}^{\uparrow}$

Following Cauchy's condensation test, the $\mathsf{CauchyCondensationTest}^{\uparrow}$ packet records the finite dyadic-block comparison certificate for a monotone nonnegative series tail before any Real-facing consumer is allowed to read a completion seal. It consumes the partial-sum and tail-budget surfaces of chapter 121 and chapter 367, the dyadic scale rows of chapter 291, the regular rational handoff of chapter 301, the seal boundary of chapter 38, and the comparison-test route of chapter 1459. The packet is a constructive block-comparison surface; it is not a quotient of series, a selected infinite sum, countable choice, trichotomy on arbitrary terms, RealUp completeness, or host real equality.

Definition 1478.1 (Cauchy condensation test carrier)**.** *A* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, I, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the nonnegative monotone* $\mathsf{Series}^{\uparrow}$ *source row whose displayed summand observations are read through finite prefixes.*
2. *B is the dyadic block index ledger, recording the inspected powers of two, block endpoints, and the finite windows on which the monotone tail is evaluated.*
3. *I is the condensed block term row, displaying the term $2^n a_{2^n}$ as a row over the same dyadic index ledger.*
4. *D is the dyadic block domination ledger, asserting that each inspected block of the original tail is bounded above and below by the displayed condensed block term rows in the monotone direction required by the test.*
5. *M is the comparison-test budget row, packaging the finite tolerance requests, dyadic thresholds, and comparison-test calls consumed by the downstream route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed block ledger and comparison budget.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row attached to the regular rational handoff.*
8. *H records componentwise* hsame *transports for the series source, dyadic index ledger, condensed block terms, domination ledger, comparison budget, regular handoff, real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each finite tail request through B, I, D, M to the regular rational handoff R and the seal row E.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient series equality, a selected sum, countable choice, trichotomy outside the inspected dyadic ledger, host real equality, or RealUp completeness.

Lean4 ✓ `BEDC.Derived.CauchyCondensationTestUp.CauchyCondensationTestTasteGate single carrier alignment`

Theorem 1478.2 (Cauchy condensation test NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *carrier $K = (A, B, I, D, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *nonnegative monotone source exposure for A through finite* $\mathsf{Series}^{\uparrow}$ *summand and partial-sum rows;*
3. *dyadic block index readback for B through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale ledger;*
4. *condensed block term formation for I on the same displayed dyadic indices;*
5. *block domination compatibility for D, stating that monotonicity turns each inspected original tail block into the corresponding finite comparison against I;*
6. *comparison-budget routing for M into the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *and* $\mathsf{FastConvergentSeries}^{\uparrow}$ *surfaces;*
7. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal exposure for R and E;*
8. *transport stability, continuation replay, provenance, and local naming for exactly the route recorded by H, C, P, N.*

These obligations are finite-row obligations over the sibling surfaces named in the chapter introduction.

Proof. Project the carrier $K = (A, B, I, D, M, R, E, H, C, P, N)$. The projection exposes exactly the monotone nonnegative series source, dyadic index ledger, condensed block terms, domination ledger, comparison budget, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

For a displayed tail request, read the dyadic index ledger B. The ledger selects finitely many dyadic blocks, and the condensed row I records the corresponding block terms on those same indices.

Monotonicity is used only inside the displayed domination ledger D. Each inspected original tail block is compared with the condensed block row by finite endpoint reads, so the comparison budget M receives a block comparison rather than a hidden infinite tail argument.

Finally apply the structural rows. The hsame rows in H preserve the source, block, domination, budget, handoff, and seal data componentwise, while C, P, N only replay, source, and name the displayed route. No omitted coordinate remains from which an additional NameCert obligation could be read. □

Theorem 1478.3 (Cauchy condensation test dyadic-block handoff)**.** *Every accepted* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *carrier sends a monotone nonnegative tail to the comparison-test route only through the finite dyadic-block handoff*

$$A, B, I \longmapsto D, M \longmapsto R, E, H, C, P, N.$$

Thus a downstream consumer can read the condensed comparison only after it has read the displayed dyadic block indices, the condensed block term rows, the finite block domination ledger, and the comparison-test budget.

Proof. Begin with an accepted carrier and fix a displayed precision or tail request. The dyadic ledger B supplies the finite block endpoints, and the condensed row I supplies the block term associated with each endpoint.

Read the monotone source row A only on those displayed endpoints. The domination ledger D binds the finite collection of original tail terms to the condensed block row, using monotonicity as a finite block comparison.

The comparison budget M then packages the resulting block estimates for the sibling comparison-test surface. Since M receives only the displayed rows A, B, I, D, the handoff cannot bypass the dyadic block ledger.

Transport and replay preserve the same route. The hsame rows in H carry the dyadic indices, condensed block terms, domination rows, and comparison budget together, while C, P, N attach the displayed route to the regular handoff and local certificate. This gives the announced finite dyadic-block handoff. □

Theorem 1478.4 (Cauchy condensation test Real-seal boundary)**.** *For every accepted* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R produced from the dyadic-block comparison route. The seal row E receives the series source, dyadic ledger, condensed block terms, domination ledger, comparison budget, componentwise transports, continuation replay, provenance, and local naming rows, but it receives no quotient series equality, selected sum, countable choice, trichotomy premise, host real equality, or RealUp completeness theorem.*

Proof. Project the seal-facing coordinates R and E from the accepted carrier. The row R is downstream of the comparison budget M, which is itself downstream of the dyadic block ledger and domination rows.

Apply the dyadic-block handoff theorem. It shows that any comparison route reaching R has already passed through the finite rows A, B, I, D, M and through the displayed transport and replay rows attached to the same carrier.

Now read the seal row E. Its only input from the analytic route is the regular rational handoff R, together with the structural rows H, C, P, N. Since the carrier has no coordinate for quotient equality, selected sums, countable choice, trichotomy outside the inspected ledger, host real equality, or RealUp completeness, none of those objects can be exported by the seal-facing read. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationTest}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed dyadic block ledger, condensed block term rows, domination ledger, comparison budget, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: series, fastconvergentseries, dyadicratcore, regseqrat, real, regular-cauchy-comparison-test: no listed sibling supplies the whole eleven-row packet $K = (A, B, I, D, M, R, E, H, C, P, N)$, because each sibling contributes only a source, tail-budget, dyadic-scale, regular-handoff, seal, or comparison face while this chapter binds all six analytic faces with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationTest}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationTest}^{\uparrow}$ packet is generated from a nonnegative monotone series source, a dyadic block index ledger, condensed block term rows, finite block domination rows, a comparison-test budget, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1478.1, Definition 1478.2, Definition 1478.3, and Definition 1478.4 seed the finite dyadic-block comparison boundary over $\mathsf{Series}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient series equality, selected sums, countable choice, trichotomy, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite NameCert obligation surface, dyadic-block handoff theorem, and Real-seal boundary theorem.

Chapter 1479

A Concrete Naming Certificate for $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ packet records the finite integer-bracketing step used before decimal and real-approximation consumers may read an integer part. It sits over the dyadic floor surface of chapter 1416, the ordered-field comparison surface of chapter 1419, the decimal-prefix route of chapter 1417, the integer and natural-number rows of chapter 31 and chapter 28, the rational source rows of chapter 37, and the real-name routes of chapter 38 and chapter 1454. The packet is a finite bracketing certificate; it is not a host floor operation, an integer trichotomy principle, quotient real equality, a selected representative, countable choice, or an ambient completeness theorem.

Definition 1479.1 (ArchimedeanIntegerPart carrier)**.** *An* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, D, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *or regular-Cauchy read of the requested real name, exposed only through the routes of chapter 38 and chapter 1454.*
2. *A is the* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *evidence row that supplies the ordered-field comparison route over the displayed real endpoint.*
3. *D is the* $\mathsf{DyadicFloor}^{\uparrow}$ *scale row that binds the request to a finite dyadic lower grid point and its displayed dyadic successor.*
4. *Z is the* $\mathsf{Int}^{\uparrow}$ *integer candidate row, sourced through the* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*, and sign data required to read it as an integer bracket endpoint.*
5. *S is the successor bracket row, recording the displayed integer successor of Z and the two comparisons $Z \leq R$ and $R < S$ through the rows A, D.*
6. *H records componentwise* hsame *transports for the real read, Archimedean ordered-field evidence, dyadic floor scale, integer candidate, successor bracket, continuation, provenance, and local naming rows.*
7. *C records the displayed* Cont *replay that sends each finite precision or decimal-prefix request through R, A, D, Z, S without adding another search coordinate.*

8. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$, $\mathsf{DyadicFloor}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanIntegerPart}^{\uparrow}}$ *row.*

The carrier has no coordinate for host floor, trichotomy over arbitrary real names, quotient real equality, a selected representative of a real, countable choice, or ambient completeness.

Definition 1479.2 (ArchimedeanIntegerPart classifier)**.** *The local* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *classifier accepts exactly those carrier rows $K = (R, A, D, Z, S, H, C, P, N)$ for which R exposes the displayed real or regular-Cauchy read, A supplies the Archimedean ordered-field comparison evidence for that read, D supplies the finite dyadic floor scale used to locate an integer bracket, Z is the integer candidate read through the displayed* $\mathsf{Int}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{Rat}^{\uparrow}$ *rows, and S records the successor bracket with lower and upper comparisons transported through A and D. The structural rows H, C, P, N preserve and name the same route componentwise. They do not add an ambient floor operator, a second integer search, or a quotient-real comparison.*

Theorem 1479.3 (ArchimedeanIntegerPart NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{ArchimedeanIntegerPart}^{\uparrow}}$ *surface for Definition 1479.1 consists exactly of the following obligations:*

1. *carrier admission for the displayed nine-row packet $K = (R, A, D, Z, S, H, C, P, N)$;*
2. *classifier exposure from Definition 1479.2;*
3. *real-read exposure through the* $\mathsf{Real}^{\uparrow}$ *or regular-Cauchy rows in R;*
4. *Archimedean ordered-field comparison exposure through A;*
5. *finite dyadic floor-scale exposure through D;*
6. *integer-candidate exposure through Z, with its* $\mathsf{Int}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, *and* $\mathsf{Rat}^{\uparrow}$ *source rows displayed;*
7. *successor-bracket exposure through S, binding $Z \leq R < S$ to the same comparison rows;*
8. *transport stability, continuation replay, provenance, and local naming through exactly H, C, P, N.*

No other analytic, order-theoretic, or choice-theoretic obligation is part of the local certificate.

Proof. Project the accepted packet $K = (R, A, D, Z, S, H, C, P, N)$. The projection exposes exactly the real read, Archimedean ordered-field row, dyadic floor scale, integer candidate, successor bracket, transport row, continuation replay, provenance, and local naming row.

The classifier supplies the only admissible comparison route. It reads the real data through R, the ordered-field comparison through A, the finite dyadic location through D, and the integer endpoint through Z before the successor bracket row S can be formed.

The integer candidate is therefore not selected by a host floor operation. Its source is the displayed $\mathsf{Int}^{\uparrow}$ row, with the necessary $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ rows already present in the carrier provenance, and the two bracket inequalities are replayed through the same finite comparison rows.

Finally inspect the structural coordinates. The transports in H are componentwise, C only replays the displayed route, and P, N only source and name that route. Since the carrier contains no further coordinate, the listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{ArchimedeanIntegerPart}^{\uparrow}}$ surface. □

Theorem 1479.4 (ArchimedeanIntegerPart bracket route)**.** *Every accepted* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *packet exposes its integer-bracketing read through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{ArchimedeanOrderedField}^{\uparrow},\ \mathsf{DyadicFloor}^{\uparrow},\ \mathsf{Int}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DecimalExpansion}^{\uparrow}.$$

Concretely, a decimal-expansion or $\mathsf{Real}^{\uparrow}$ *approximation consumer first reads the displayed real or regular-Cauchy row* R*, then the Archimedean comparison row* A*, then the dyadic floor scale row* D*, then the integer candidate row* Z*, and only then the successor bracket row* S*.*

Proof. Start from an accepted carrier and a downstream precision or decimal-prefix request. The continuation row C routes that request to the displayed real read R, so the consumer begins with a $\mathsf{Real}^{\uparrow}$ or regular-Cauchy surface rather than a detached host real.

Next the ordered-field evidence row A supplies the comparison language used by the bracket. The finite dyadic floor row D provides the scale and dyadic lower-grid information that make the comparison request finite.

The integer endpoint is then read through Z. Its source rows are the displayed $\mathsf{Int}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ data recorded in the packet provenance, so the route reaches an integer candidate only after the real, Archimedean, dyadic, and rational data have been exposed.

The successor row S packages the bracket $Z \leq R < S$ for downstream decimal and real-approximation consumers. Because S is downstream of R, A, D, Z, every bracket read factors through the displayed finite route. □

Theorem 1479.5 (ArchimedeanIntegerPart no-hidden-search boundary)**.** *For every accepted* ArchimedeanIntegerP *carrier, no consumer can obtain an integer-part read except through the displayed rows* $R, A, D, Z, S, H, C, P, N$*. In particular, the packet exports no host floor, trichotomy over arbitrary real names, quotient real equality, selected representative, countable-choice row, hidden Archimedean search, or ambient completeness coordinate.*

Proof. Let a consumer read an integer bracket from an accepted carrier. By Definition 1479.4, the read passes through the real or regular-Cauchy row, the Archimedean comparison row, the dyadic floor scale, the integer candidate, and the successor bracket.

Inspect the carrier definition. The only remaining coordinates are H, C, P, N, and those rows merely transport, replay, source, and name the same finite route. None of them stores an alternate search procedure or a host comparison oracle.

The excluded principles are therefore absent as carrier coordinates and absent as classifier clauses. Any attempted read through host floor, trichotomy, quotient equality, selected representatives, countable choice, hidden Archimedean search, or ambient completeness would leave the displayed packet surface. □

Falsifiable prediction: An accepted $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ consumer that reaches an integer bracket without reading the displayed $\mathsf{Real}^{\uparrow}$ or regular-Cauchy row, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ comparison row, $\mathsf{DyadicFloor}^{\uparrow}$ scale row, $\mathsf{Int}^{\uparrow}$ candidate row, and successor bracket row refutes the carrier surface.

Independence witness: dyadicfloor, archimedeanorderedfield, decimalexpansion, int, nat, rat, real, regular-located-cauchy-real: no listed sibling is bijective to $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ because this packet jointly binds a real read, Archimedean comparison evidence, dyadic scale location, integer candidate, successor bracket, transport, replay, provenance, and local naming for the integer-part route.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanIntegerPart}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ or regular-Cauchy read, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ comparison evidence, a $\mathsf{DyadicFloor}^{\uparrow}$ scale row, an $\mathsf{Int}^{\uparrow}$ integer candidate with $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ sources, a successor bracket row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanIntegerPart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1479.1, Definition 1479.2, Definition 1479.3, Definition 1479.4, and Definition 1479.5 seed the finite integer-bracketing route used by decimal-expansion and RealUp approximation consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{ArchimedeanIntegerPart}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, ambient completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and classifier and prove the NameCert obligation surface, bracket route, and no-hidden-search boundary.

Chapter 1480

A Concrete Naming Certificate for $\mathsf{RealSequenceLimitLaws}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ packet records the finite algebra used when two displayed $\mathsf{Real}^{\uparrow}$ sequence limits are combined by a standard arithmetic operation. It sits over the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the sandwich route of chapter 407. The certificate keeps limit arithmetic as displayed finite route data, not as quotient stream equality, a selected host limit, trichotomy, or ambient completeness.

Definition 1480.1 (Real sequence limit laws carrier)**.** *A* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, S_X, S_Y, R_X, R_Y, D, E_X, E_Y, O, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are two* $\mathsf{Real}^{\uparrow}$ *sequence source routes whose terms are read only through finite observation windows.*
2. *S_X and S_Y are the* $\mathsf{StreamName}^{\uparrow}$ *window rows opened for the two sources at the requested precision scale.*
3. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error ledger recording the two source budgets, the operation budget, and the output tolerance request.*
5. *E_X and E_Y are the two limit-seal rows that certify the displayed source routes before any arithmetic read is public.*
6. *O is the operation row, recording whether the read is sum, difference, product under a displayed bound, scalar multiple, or squeeze-mediated comparison, together with its required dyadic error estimate.*
7. *H stores componentwise* hsame *transport for the two sources, their windows, readbacks, limit seals, dyadic ledger, and operation row.*
8. *C stores displayed* Cont *replay for opening windows, reading regular rational approximants, consuming dyadic error rows, invoking the operation row, and exposing the output seal route.*

9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SqueezeReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequenceLimitLaws}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no host equality of real sequences, no selected limit value, no trichotomy row, and no ambient completeness principle.

Theorem 1480.2 (Real sequence limit laws NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealSequenceLimitLaws}^{\uparrow}}$ *surface for Definition 1480.1 consists of five finite obligation rows:*

1. *carrier admission for the displayed packet* $L = (X, Y, S_X, S_Y, R_X, R_Y, D, E_X, E_Y, O, H, C, P, N)$*;*
2. *classifier stability under componentwise* hsame *transport and displayed* Cont *replay of the two source routes, window rows, readback rows, limit seals, dyadic ledger, and operation row;*
3. *operation-row exactness, requiring the output tolerance read to be the dyadic estimate stored in D and cited by O;*
4. *seal handoff, requiring every public limit-law read to pass through E_X, E_Y, O, and the displayed transport, replay, provenance, and local naming rows before any* $\mathsf{Real}^{\uparrow}$ *output seal is exposed;*
5. *non-escape, stating that quotient stream equality, selected host limits, trichotomy, host real-sequence equality, and ambient completeness are absent from the packet.*

Proof. Carrier admission is projection from Definition 1480.1. The two source routes, two window rows, two readback rows, dyadic ledger, two limit seals, operation row, transport, replay, provenance, and local name are explicit coordinates of L.

Classifier stability is componentwise. The stored H and C rows transport and replay exactly the coordinates named by the carrier, so a classified read cannot detach an operation row from its two source seals or from the dyadic error ledger.

Operation-row exactness is the role of D and O. The dyadic ledger records the input and output tolerances, while O declares which arithmetic or squeeze-mediated operation consumes those estimates.

The seal handoff then forces a public limit-law consumer to read E_X and E_Y before it invokes O. Since the carrier has no coordinate for quotient stream equality, selected host limits, trichotomy, host sequence equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 1480.3 (Real sequence limit laws RealUp handoff)**.** *Every public limit-law read exported by an accepted* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 1480.1. The handoff exports no quotient stream equality, selected host limit, trichotomy, host equality of real sequences, or ambient completeness principle.*

Proof. Begin with a public limit-law consumer and project the accepted carrier. The consumer first receives the two source routes X, Y through the finite windows S_X, S_Y and the regular rational readbacks R_X, R_Y.

The dyadic error ledger D is then read at the same precision scale. It is the only tolerance coordinate listed in the carrier, and the operation row O must cite it before an output route can be exposed.

The two limit-seal rows E_X, E_Y bind the source routes to the terminal $\mathsf{Real}^{\uparrow}$ surface, while H, C, P, N provide the componentwise transport, replay, provenance, and local naming route.

These rows are exactly the public handoff surface. No further coordinate remains from which a consumer could read a quotient stream equality, a selected host limit, trichotomy, host sequence equality, or ambient completeness. □

Theorem 1480.4 (Real sequence limit laws dyadic operation exhaustion)**.** *Every accepted operation row O in a* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *carrier*

$$L = (X, Y, S_X, S_Y, R_X, R_Y, D, E_X, E_Y, O, H, C, P, N)$$

must cite the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D for its input budgets, operation budget, and output tolerance request. After $D, O, E_X, E_Y, H, C, P, N$ are fixed, no quotient stream equality, selected host limit, trichotomy row, host sequence equality, or ambient completeness principle can alter the public limit-law read.*

Proof. Project the accepted carrier. The operation row O and the dyadic ledger D are the only coordinates that describe arithmetic error accounting for the two displayed source routes.

Apply operation-row exactness from Definition 1480.2. The row O must consume the tolerance data stored in D, including the two input budgets, the operation budget, and the output tolerance request.

Apply the RealUp handoff theorem Definition 1480.3. Once the two source seals E_X, E_Y and the structural rows H, C, P, N are fixed with D and O, the public limit-law read is already routed through the displayed finite packet.

The carrier has no coordinate for quotient stream equality, selected host limits, trichotomy, host sequence equality, or ambient completeness. Hence none of those objects can revise the dyadic operation read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceLimitLaws}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ packet is generated from two $\mathsf{Real}^{\uparrow}$ sequence source routes, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ error ledger, two limit-seal rows, an operation row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSequenceLimitLaws}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1480.1, Definition 1480.2, Definition 1480.3, and Definition 1480.4 seed the finite constructive limit-law packet over StreamName, RegSeqRat, DyadicRatCore, RealUp, and SqueezeReal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSequenceLimitLaws}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, trichotomy, host equality of real sequences, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, operation-row exactness, seal handoff, and non-escape for the same finite limit-law packet.

Chapter 1481

A Concrete Naming Certificate for $\mathsf{RegularCauchyRootTest}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyRootTest}^{\uparrow}$ packet records the finite root-bound test that turns a displayed eventually bounded root ledger into a regular Cauchy tail before the Real-facing seal is consumed. It sits over the fast-tail surface of chapter 367, the dyadic budget rows of chapter 379, the regular rational readback of chapter 301, the Real seal surface of chapter 38, the majorant route of chapter 1459, and the geometric-tail route of chapter 1461. The packet is a constructive root test for L10 Real and Completion certificates; it is not a host root-limit theorem, not a quotient of streams, not a selected sum, not countable choice, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 1481.1 (Regular Cauchy root test carrier)**.** *A* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, Q, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the series or tail source row whose finite terms are inspected through displayed windows.*
2. *Q is the root-bound row, recording a dyadic bound strictly inside the unit tolerance scale together with the finite entry index after which the bound is admitted.*
3. *M is the modulus row assigning each displayed precision request a finite threshold beyond the root-bound entry index.*
4. *D is the dyadic tail-domination ledger that turns the root-bound row into a geometric majorant consumed by* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *and checked against the geometric-tail discipline of* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed tail-domination ledger and modulus row.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only through the regular rational handoff.*
7. *H records componentwise* hsame *transport for the source, root-bound, modulus, dyadic domination, regular handoff, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through Q and M, then through D to the comparison-test surface, the ratio-test geometric budget, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRootTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host root limits, quotient streams, selected sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 1481.2 (Regular Cauchy root test NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *carrier $T = (A, Q, M, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRootTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *root-bound tail domination, requiring Q, M, D to turn each inspected tail request into a finite dyadic geometric majorant routed through* $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, *and* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*;*
3. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be read only from the displayed source, root-bound, modulus, and tail-domination rows;*
4. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E to factor through R and to export no host root limit, quotient stream, selected sum, countable choice, or ambient completeness row;*
5. hsame, Cont, Pkg, *and* NameCert *transport for exactly the rows recorded by H, C, P, N.*

These obligations are finite-row obligations over the sibling surfaces named in the chapter introduction.

Proof. Project the carrier $T = (A, Q, M, D, R, E, H, C, P, N)$. The projection exposes the source, root-bound, modulus, tail-domination, regular handoff, Real seal, transport, continuation, provenance, and local naming rows.

For a displayed precision request, read Q and M. The root-bound row supplies the dyadic bound and entry index, while the modulus row chooses a finite threshold beyond that entry index.

The dyadic ledger D performs the tail-domination step. It converts the displayed root bound into a geometric majorant, sends the majorant through the $\mathsf{DyadicTailBound}^{\uparrow}$ and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ surfaces, and checks the geometric budget against the $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ route.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is then obtained from the same displayed route. The seal row E is downstream of R, so it cannot expose a host root-limit object, quotient stream, selected sum, countable-choice row, or ambient completeness theorem.

Finally apply the structural rows. The hsame transports in H preserve the analytic rows componentwise, while C, P, N replay, source, and name exactly the displayed route. No omitted coordinate remains from which an additional NameCert obligation could be read. □

Theorem 1481.3 (Regular Cauchy root test tail-bound handoff)**.** *Every accepted* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *carrier hands a root-bound tail to* $\mathsf{RegSeqRat}^{\uparrow}$ *and then to* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$A, Q, M \longmapsto D \longmapsto R, E, H, C, P, N.$$

Consequently a Real-facing consumer can use the root test only after reading the displayed root-bound row, modulus row, dyadic tail-domination ledger, comparison-test handoff, ratio-test geometric budget, regular rational handoff, and local transport package.

Proof. Begin with an accepted carrier and a displayed precision request. The rows Q and M determine the admitted root bound, entry index, and finite tail threshold.

Read the domination ledger D. This ledger is the only row that converts the root-bound data into a geometric majorant, and it names the $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, and $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ routes consumed by the handoff.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is obtained by replaying that displayed domination route. Since R depends on A, Q, M, D and not on a host root-limit theorem, the downstream $\mathsf{Real}^{\uparrow}$ seal row E can only read the regular rational handoff presented by the carrier.

Transport and provenance preserve the same route. The rows H, C, P, N carry the root-bound tail data, replay the comparison and ratio-test handoffs, and name the local certificate without adding quotient streams, selected sums, countable choice, RealUp completeness, or bridge checking. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyRootTest}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed root-bound row, modulus row, dyadic tail-domination ledger, comparison-test handoff, ratio-test geometric budget, and transport package refutes the carrier surface.

Independence witness: fastconvergentseries, dyadictailbound, regseqrat, real, regular-cauchy-comparison-test, regular-cauchy-ratio-test: no listed sibling supplies the whole ten-row packet $T = (A, Q, M, D, R, E, H, C, P, N)$, because each sibling contributes only a tail budget, dyadic bound, regular handoff, Real seal, comparison face, or geometric-tail face while this chapter binds the root-bound route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRootTest}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRootTest}^{\uparrow}$ packet is generated from a source row, a displayed root-bound row, an explicit modulus row, a dyadic tail-domination ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRootTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1481.1, Definition 1481.2, and Definition 1481.3 seed the finite root-test boundary over $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, and $\mathsf{RegularCauchyRatioTest}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRootTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host root limits, quotient streams, selected sums, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface and tail-bound handoff theorem.

Chapter 1482

A Concrete Naming Certificate for $\mathsf{ChoiceSequence}^{\uparrow}$

Brouwer/Bishop constructive choice-sequence tradition treats an unfolding sequence by the finite information currently displayed to a proof or construction. The $\mathsf{ChoiceSequence}^{\uparrow}$ packet records that surface as a finite-prefix NameCert neighbour of chapter 334 and chapter 126. It uses $\mathsf{Nat}^{\uparrow}$ answer rows, scheduled $\mathsf{StreamName}^{\uparrow}$ prefix requests, and the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer boundaries only as visible handoff rows; it does not complete a host function $\mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$, quotient infinite streams, or add a countable-choice selector.

Definition 1482.1 (ChoiceSequence prefix carrier)**.** *A* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier is a finite* BHist *packet*

$$Q = (S, W, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite prefix requests visible to the consumer;*
2. *W is the finite prefix-window ledger listing the requested natural-number positions;*
3. *A is the* $\mathsf{Nat}^{\uparrow}$ *answer ledger assigning displayed natural-number rows to the positions of W;*
4. *R is the optional* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the same finite prefix is consumed by a regular-rational route;*
5. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal row reached only through R and the displayed prefix data;*
6. *H is the componentwise* hsame *transport row for S, W, A, R, E;*
7. *C is the displayed* Cont *route for prefix restriction and already accepted prefix replay;*
8. *P is the* Pkg *provenance over* $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ *row.*

The carrier accepts only displayed finite prefixes with visible answer rows. It exports no completed host function, no quotient stream equality, no countable-choice selector, no real-completeness principle, and no external topology.

Theorem 1482.2 (ChoiceSequence prefix stability)**.** *Let $Q = (S, W, A, R, E, H, C, P, N)$ be an accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier. Every prefix subwindow of W, together with the corresponding displayed* $\mathsf{Nat}^{\uparrow}$ *answer rows of A, repacks as an accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier*

through the same hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The restricted carrier has no access to tail values outside the displayed subwindow.*

Proof. Project the carrier of Definition 1482.1. The schedule row S, prefix ledger W, and answer ledger A are finite BHist rows, so a prefix subwindow is obtained by deleting later displayed entries rather than by reading an infinite sequence.

Transport the retained entries through the componentwise hsame row H. The Cont route C is already a prefix-restriction route for accepted reads, so replaying it on the retained rows preserves the order of the visible positions and the attached $\mathsf{Nat}^{\uparrow}$ answers.

The provenance row P and local naming row N then repack the restricted data as the same kind of certificate. Since no coordinate in Q stores a completed host function or a hidden tail oracle, the restricted carrier cannot export values beyond the displayed subwindow. □

Theorem 1482.3 (ChoiceSequence prefix amalgamation)**.** *Let* $Q_0 = (S_0, W_0, A_0, R_0, E_0, H_0, C_0, P_0, N_0)$ *and* $Q_1 = (S_1, W_1, A_1, R_1, E_1, H_1, C_1, P_1, N_1)$ *be accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carriers. Suppose the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{Nat}^{\uparrow}$ *answer ledgers agree on the overlap of* W_0 *and* W_1 *under componentwise* hsame*, and suppose the optional* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows and* $\mathsf{Real}^{\uparrow}$ *seal rows, when present on both sides, are the same displayed handoff rows over that overlap. Then the finite union of the two displayed windows repacks as an accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier using the common* $\mathsf{StreamName}^{\uparrow}$ *schedule on the overlap, the single-valued union of the* $\mathsf{Nat}^{\uparrow}$ *answer rows, the inherited optional* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows,* Cont *replay over the combined window,* Pkg *provenance, and a local* $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ *row.*

Proof. Project the two carriers through Definition 1482.1. The rows W_0, W_1 are finite displayed ledgers, so their union is again a finite prefix-window ledger rather than a completed host function.

On the common part of the two ledgers, the hypotheses identify the displayed schedules and $\mathsf{Nat}^{\uparrow}$ answer rows by componentwise hsame. Hence the union ledger is single-valued: an overlapping position carries one answer row, while a position belonging to only one side keeps the answer row already displayed there.

The optional R and E rows are copied only along the visible handoff routes. Where both carriers expose such rows, the overlap hypothesis makes them the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff; where only one side exposes them, the row remains attached to that side's displayed prefix data.

Replay the continuation rows C_0 and C_1 over their respective subwindows inside the combined finite ledger. Prefix stability from Definition 1482.2 certifies that each original carrier is recovered by restricting the union back to its displayed window.

Finally, the provenance rows P_0, P_1 and local naming rows N_0, N_1 supply the same kernel ingredients named in the carrier definition, so the combined finite data repacks as a local $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ carrier. The construction adds no completed host function, countable-choice selector, quotient stream equality, real-completeness principle, or external topology. □

Theorem 1482.4 (ChoiceSequence StreamName handoff)**.** *Every public read of an accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule row before it reaches any* $\mathsf{Nat}^{\uparrow}$ *answer row, any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, or any* $\mathsf{Real}^{\uparrow}$ *seal row. The handoff route is*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{Nat}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

when the last two rows are present, and otherwise stops at the displayed natural-number prefix.

Proof. Begin with a public consumer read of Q. The consumer first receives the finite request schedule S and the prefix ledger W; without these rows there is no named position at which a natural-number answer can be inspected.

The $\mathsf{Nat}^{\uparrow}$ answer ledger A is read only over those scheduled positions. If a regular-rational consumer is present, the readback row R is attached to the same displayed prefix, and the $\mathsf{Real}^{\uparrow}$ seal row E is reached only after R has been replayed through H, C, P, N.

Thus the public route must pass through $\mathsf{StreamName}^{\uparrow}$ before any answer or completion-facing handoff is available. If R and E are absent, the same argument terminates at the finite $\mathsf{Nat}^{\uparrow}$ prefix surface and yields no regular-rational or real-facing row. □

Theorem 1482.5 (ChoiceSequence tail nonanticipation)**.** *Let $Q = (S, W, A, R, E, H, C, P, N)$ and $Q' = (S, W, A', R', E', H, C, P, N)$ be accepted $\mathsf{ChoiceSequence}^{\uparrow}$ prefix carriers with the same displayed $\mathsf{StreamName}^{\uparrow}$ schedule row, prefix-window ledger, transport, continuation, provenance, and local naming rows. Every public read restricted to the displayed prefix factors only through S, W, A, H, C, P, N and through the optional R, E rows already attached to that prefix. If the displayed $\mathsf{Nat}^{\uparrow}$ answer rows of A and A' agree on W, then no consumer can distinguish Q from Q' by any tail value outside W.*

Any claimed distinction between Q and Q' outside the displayed window would require one of the following absent coordinates: a completed host function, a hidden tail oracle, quotient stream equality, a countable-choice selector, or a $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Proof. Project the two carriers through Definition 1482.1. The rows S, W, H, C, P, N are shared, and the visible answer ledgers agree on the whole displayed window W.

Apply prefix stability from Definition 1482.2 to restrict any public read to the displayed subwindow. The restriction removes later undisplayed positions rather than evaluating them, so the resulting consumer surface is determined by the shared schedule and the visible $\mathsf{Nat}^{\uparrow}$ answers.

Apply the StreamName handoff of Definition 1482.4. Any answer read, and any optional $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback, must pass through S before reaching A, R, E or A', R', E'. Since the read is restricted to W and the displayed answers agree there, the public route has the same visible data in both carriers.

It remains to inspect the possible source of a tail distinction. The carrier definition exports no completed host function, hidden tail oracle, quotient stream equality, countable-choice selector, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. Therefore no local $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ consumer can distinguish Q from Q' by any value outside the displayed prefix. □

Theorem 1482.6 (ChoiceSequence NameCert obligation surface)**.** *The seed $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ obligation surface consists exactly of carrier admission for Definition 1482.1, prefix stability of Definition 1482.2, compatible-prefix amalgamation of Definition 1482.3, StreamName handoff of Definition 1482.4, tail nonanticipation of Definition 1482.5, componentwise* hsame *transport of all displayed rows,* Cont *prefix restriction and replay,* Pkg *provenance, and local naming through N. It exposes the finite-prefix choice-sequence neighbour of $\mathsf{BaireSpace}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ while exporting no completed host function, countable-choice selector, quotient stream equality, real completeness, or external topology.*

Proof. Carrier admission is the projection of the nine displayed rows in Definition 1482.1. The schedule, prefix ledger, answer ledger, optional regular readback, optional real seal, transport, continuation, provenance, and local naming rows are exactly the rows available to a NameCert consumer.

Prefix stability is supplied by Definition 1482.2. It shows that every shorter displayed prefix is certified by the same finite rows after hsame transport and Cont prefix restriction.

Compatible-prefix amalgamation is supplied by Definition 1482.3. It shows that two accepted displayed prefixes with the same overlap answers form one finite carrier, so downstream readers can treat compatible finite evidence as a single $\mathsf{ChoiceSequence}^{\uparrow}$ surface.

The handoff clause is supplied by Definition 1482.4. It forces every answer read and every regular or real-facing consumer route to pass through the displayed $\mathsf{StreamName}^{\uparrow}$ schedule and the visible $\mathsf{Nat}^{\uparrow}$ prefix answers.

Tail nonanticipation is supplied by Definition 1482.5. It states that two accepted carriers with the same displayed schedule and agreeing visible answers cannot be separated by data outside the prefix window.

It remains to inspect exclusions. The packet has no coordinate for a completed host function, countable-choice selector, quotient stream equality, real-completeness witness, or external topology, so none of those objects can be exported by the local $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceSequence}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ request schedule, displayed $\mathsf{Nat}^{\uparrow}$ answer rows, optional $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumer handoffs, componentwise hsame transport, Cont prefix replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1482.1, Definition 1482.2, Definition 1482.3, Definition 1482.4, Definition 1482.5, and Definition 1482.6 seed the finite-prefix choice-sequence surface over $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{Nat}^{\uparrow}$ answer rows, optional $\mathsf{RegSeqRat}^{\uparrow}$ readback, optional $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close completed host functions, countable choice, quotient stream equality, real completeness, or external topology.

Upgrade path: The next step is obligationClosure: carrier admission, prefix stability, compatible-prefix amalgamation, StreamName handoff, tail nonanticipation, transport, continuation, provenance, and non-escape must be discharged as local rows.

Chapter 1483

A Concrete Naming Certificate for $\mathsf{AlternatingSeriesTest}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{AlternatingSeriesTest}^{\uparrow}$ packet records the alternating-series test as finite BEDC data before a completion-facing consumer may read a regular rational handoff or a real seal. Its source and partial-sum rows sit beside the Cauchy criterion of chapter 392; its regular rational endpoint is the route of chapter 301; its terminal boundary is the $\mathsf{Real}^{\uparrow}$ seal of chapter 38; and its dyadic tail estimate is the finite comparison discipline of chapter 1459. The packet is a constructive convergence-test surface for a displayed alternating source; it is not a selected sum, a quotient series equality, host real equality, countable choice, trichotomy on arbitrary summands, or RealUp completeness.

Definition 1483.1 (Alternating series test carrier)**.** *An* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, \Sigma, M, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the series source row, displaying a rational summand family a_n and the partial-sum histories for $\sum_{i<n}(-1)^i a_i$.*
2. *Σ is the alternating-sign ledger, recording the finite parity row that attaches the sign $(-1)^n$ to the nth inspected summand.*
3. *M is the monotone dyadic magnitude ledger, displaying nonnegative dyadic majorants for the inspected magnitudes and the finite rows witnessing that the majorants decrease along the observed tail.*
4. *T is the tail-bound schedule, assigning each displayed precision request a finite threshold after which the next omitted magnitude is below the requested dyadic tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the partial-sum source, alternating-sign ledger, monotone magnitude ledger, and tail-bound schedule.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the regular rational handoff.*
7. *H records componentwise* hsame *transports for the source, sign ledger, magnitude ledger, tail-bound schedule, regular handoff, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *routes from each precision request through Σ, M, T to the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R and then to the* $\mathsf{Real}^{\uparrow}$ *seal row E.*

9. *P is the* Pkg *provenance over* $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for a chosen infinite sum, quotient series equality, host-level equality of reals, countable choice, trichotomy outside the inspected dyadic magnitude rows, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.AlternatingSeriesTestUp.AlternatingSeriesTestTasteGate single carrier alignment`

Theorem 1483.2 (Alternating series test NameCert obligations)**.** *For every accepted* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *carrier $A = (S, \Sigma, M, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *source exposure for the rational summand and partial-sum rows in S;*
3. *alternating-sign exposure for Σ, tying every inspected partial-sum step to its finite parity row;*
4. *monotone dyadic magnitude exposure for M, including the nonnegative magnitude row and the displayed decrease row on the observed tail;*
5. *tail-bound compatibility for T, sending every precision request to a finite threshold with next omitted magnitude below that tolerance;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal exposure for R and E;*
7. *transport stability, continuation replay, provenance, and local naming for exactly the route recorded by H, C, P, N.*

These obligations are finite-row obligations over the Cauchy criterion of chapter 392, the regular rational route of chapter 301, the real seal boundary of chapter 38, and the dyadic comparison route of chapter 1459.

Proof. Project the accepted packet $A = (S, \Sigma, M, T, R, E, H, C, P, N)$. The projection exposes exactly the rational source, alternating-sign ledger, monotone dyadic magnitude ledger, tail-bound schedule, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

Read a displayed precision request through T. The schedule selects a finite threshold, and the magnitude ledger M states that the next omitted majorant past that threshold is below the requested dyadic tolerance.

The sign ledger Σ is used only to identify the finite cancellation pattern in the partial sums from S. Together with monotonicity in M, this is the displayed Cauchy-criterion input: tails after the chosen threshold are bounded by the next omitted magnitude, not by an unrecorded infinite choice.

Finally apply the structural rows. The componentwise transports in H preserve the source, sign, magnitude, threshold, handoff, and seal rows together, while C, P, N only replay, source, and name that route. Since the carrier has no further coordinate, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}$ surface. □

Theorem 1483.3 (Alternating series test Cauchy handoff)**.** *Every accepted* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *carrier sends the alternating partial-sum family to the Cauchy-facing route only through the finite handoff*

$$S, \Sigma, M, T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *consumer can use the alternating-series test only after reading the displayed summand source, alternating-sign ledger, monotone dyadic magnitude ledger, and tail-bound schedule that bind the tail estimate to the regular rational handoff.*

Proof. Begin with an accepted carrier and a displayed precision request. The tail-bound schedule T supplies a finite threshold, while the source row S supplies only the partial sums visible around that threshold.

Read the sign and magnitude rows. The ledger Σ determines which finite summand positions are added and subtracted, and M supplies the monotone dyadic magnitude comparison on the same positions.

These rows are exactly the data consumed by the Cauchy criterion route: the difference of two sufficiently late partial sums is bounded by the next omitted displayed magnitude. The handoff row R records that regular rational readback and does not receive any separate completed real or quotient series object.

Transport and replay preserve this handoff componentwise. The rows H, C, P, N attach the route to the local certificate without creating an alternate path through host convergence, selected sums, or RealUp completeness. □

Theorem 1483.4 (Alternating series test Real-seal boundary)**.** *For every accepted* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R produced from the displayed alternating partial sums, sign ledger, monotone dyadic magnitude ledger, and tail-bound schedule. The seal row E receives only R together with the componentwise transports, continuation replay, provenance, and local naming rows; it receives no selected sum, quotient series equality, host real equality, countable choice, trichotomy premise, or RealUp completeness theorem.*

Proof. Project the seal-facing rows R and E from the accepted carrier. The regular handoff R is downstream of S, Σ, M, T, and the seal row E is downstream of R.

Apply Definition 1483.3. It identifies the only route from alternating partial sums to the regular readback as the finite path through the displayed source, sign, magnitude, and tail-bound rows.

Read the seal row. Its analytic input is exactly the regular handoff R, together with H, C, P, N. Since the carrier has no coordinate for a selected sum, quotient series equality, host real equality, countable choice, unrestricted trichotomy, or RealUp completeness, the seal-facing read cannot export any of those objects. □

Theorem 1483.5 (Alternating series test public convergence export)**.** *The public convergence export of an accepted* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *carrier is exactly the finite statement that its displayed alternating partial sums form a* $\mathsf{RegSeqRat}^{\uparrow}$ *row and may be sealed as the displayed* $\mathsf{Real}^{\uparrow}$ *boundary. Any downstream consumer receives the source row S, alternating-sign ledger Σ, monotone dyadic magnitude ledger M, tail-bound schedule T, regular handoff R, real seal E, componentwise transports H, continuation replay C, provenance P, and local naming row N, and receives no further convergence principle.*

Proof. Unpack the public export through the carrier definition. The only exported convergence data are the regular handoff R and the seal row E, and both are indexed by the same source, sign, magnitude, and tail-bound rows.

Use Definition 1483.2 to identify the certificate obligations attached to those rows. They expose carrier admission, finite source inspection, sign parity, monotone dyadic magnitude, tail-bound scheduling, regular readback, seal exposure, and structural transport.

Use Definition 1483.4 for the boundary condition. The real-facing consumer reads the displayed seal and cannot recover any selected sum, quotient series equality, host real equality, choice principle, trichotomy oracle, or completeness theorem beyond the finite packet. □

Falsifiable prediction: An accepted $\mathsf{AlternatingSeriesTest}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed alternating-sign ledger, monotone dyadic magnitude ledger, tail-bound schedule, and source partial-sum row refutes the carrier surface.

Independence witness: cauchycriterion, regseqrat, real, regular-cauchy-comparison-test: no listed sibling supplies the whole ten-row packet $A = (S, \Sigma, M, T, R, E, H, C, P, N)$, because each sibling contributes only a Cauchy route, regular readback, seal face, or dyadic comparison face while this chapter binds the alternating sign, monotone magnitude, tail-bound, transport, replay, provenance, and naming rows into one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}$.

Constructive story: An $\mathsf{AlternatingSeriesTest}^{\uparrow}$ packet is generated from a rational summand source, finite alternating-sign ledger, monotone dyadic magnitude ledger, tail-bound schedule, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1483.1, Definition 1483.2, Definition 1483.3, Definition 1483.4, and Definition 1483.5 seed the finite alternating-series-test boundary over $\mathsf{Series}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlternatingSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected sums, quotient series equality, host real equality, countable choice, trichotomy, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface, Cauchy handoff theorem, Real-seal boundary theorem, and public convergence export theorem.

Chapter 1484

A Concrete Naming Certificate for $\mathsf{DyadicDensityReal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicDensityReal}^{\uparrow}$ packet records the finite dyadic witness surface used when a displayed real-name interval must expose a dyadic point at a displayed radius. It sits over the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the constant dyadic real-name route of chapter 467, the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names density as finite BEDC route data, not as quotient streams, selected host limits, host real equality, trichotomy, or ambient completeness.

Definition 1484.1 (DyadicDensityReal carrier)**.** *A* $\mathsf{DyadicDensityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (L, U, D, \rho, S_L, S_U, R_L, R_U, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seal rows for the displayed interval, read only as Real-facing name data.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row naming the candidate dyadic rational witness and its comparison scales.*
3. *ρ is the dyadic radius row recording the requested positive window and the two endpoint tolerances used by the witness read.*
4. *S_L and S_U are the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows opened for the two endpoint names at the scales requested by ρ.*
5. *R_L and R_U are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those two finite windows.*
6. *E is the* $\mathsf{DyadicEmbedding}^{\uparrow}$ *handoff row sending the dyadic source named by D into the same real-name surface as the two endpoint seals.*
7. *H stores componentwise* hsame *transport for the endpoint seals, dyadic source row, radius row, stream windows, regular readbacks, embedding handoff, continuations, provenance, and local naming.*
8. *C stores displayed* Cont *routes for opening endpoint windows, reading the regular rational rows, consuming the dyadic source and radius rows, invoking the embedding handoff, and exposing the dyadic witness to the* $\mathsf{Real}^{\uparrow}$ *interval consumer.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityReal}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, selected host limit, host real equality, trichotomy row, or ambient completeness principle.

Theorem 1484.2 (DyadicDensityReal NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDensityReal}^{\uparrow}$ *carrier* $W = (L, U, D, \rho, S_L, S_U, R_L, R_U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityReal}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the displayed endpoint, dyadic, window, readback, handoff, transport, continuation, provenance, and local naming rows;*
2. *endpoint-window compatibility, stating that S_L, S_U, R_L, R_U are read from the same displayed radius row ρ and the same endpoint seals L, U;*
3. *dyadic-source exactness, stating that the exported witness is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source named by D at the radius and comparison scales recorded in ρ;*
4. *embedding handoff, stating that the only Real-facing route for the dyadic witness is the* $\mathsf{DyadicEmbedding}^{\uparrow}$ *row E followed by the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *reads;*
5. *non-escape, stating that quotient streams, selected host limits, host real equality, trichotomy, and ambient completeness are absent from the carrier.*

Proof. Project the carrier of Definition 1484.1. The two endpoint seals, dyadic source, radius row, two window rows, two readback rows, embedding handoff, transports, continuations, provenance, and local name are exactly the coordinates available to the classifier.

Endpoint-window compatibility follows from the shared radius row. Both S_L, S_U and their readbacks R_L, R_U are listed as windows opened at the scales requested by ρ, so a classified interval read cannot use an endpoint window detached from the displayed tolerance request.

Dyadic-source exactness and embedding handoff are the same finite route read from opposite sides. The row D supplies the dyadic rational witness, while E is the only coordinate that embeds that source into the Real-facing naming surface.

The non-escape row is inspection of the packet. Since the carrier has no coordinate for quotient streams, selected host limits, host real equality, trichotomy, or ambient completeness, the five rows exhaust the local NameCert surface. □

Theorem 1484.3 (DyadicDensityReal witness route)**.** *Every exported dyadic witness of an accepted* $\mathsf{DyadicDensityReal}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{DyadicEmbedding}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

together with the carrier's componentwise hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route exports no quotient stream, selected host limit, host real equality, trichotomy row, or ambient completeness principle.*

Proof. Begin with an exported dyadic witness and project the accepted carrier. The only dyadic source coordinate is D, so the witness begins as the $\mathsf{DyadicRatCore}^{\uparrow}$ row named by the packet.

The witness then crosses the embedding handoff. The row E is the sole $\mathsf{DyadicEmbedding}^{\uparrow}$ coordinate, and the continuation routes C expose it only after the endpoint windows and readbacks attached to ρ have been displayed.

Finally the Real-facing consumer reads the same finite surface through S_L, S_U, R_L, R_U and the endpoint seals L, U. Componentwise transport H, provenance P, and the local naming row N preserve that route as a single packet over the displayed primitives.

No other carrier coordinate can export a witness. Hence every public dyadic density read factors through the stated $\mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{DyadicEmbedding}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}$ route and cannot obtain any of the excluded host or quotient data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDensityReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDensityReal}^{\uparrow}$ packet is generated from lower and upper Real endpoint seals, a DyadicRatCore source row, a dyadic radius row, finite StreamName windows, RegSeqRat readbacks, a DyadicEmbedding handoff, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDensityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1484.1, Definition 1484.2, and Definition 1484.3 seed the finite dyadic-density packet over DyadicRatCore, DyadicEmbedding, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDensityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-window compatibility, dyadic-source exactness, embedding handoff, and non-escape for the same finite dyadic-density packet.

The $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ route consumes the $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{CauchyModulusMeet}^{\uparrow}$ schedules of chapter 292 and chapter 462, then hands shared thresholds to the $\mathsf{RegularCauchySum}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$ consumer rows before the $\mathsf{Real}^{\uparrow}$ seal is read.

Chapter 1485

A Concrete Naming Certificate for $\mathsf{CauchyModulusArithmetic}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, explicit Cauchy moduli are part of the data by which regular rational sequences are consumed: addition, multiplication, and common-tail comparison require visible schedules rather than a hidden appeal to completed real equality. The $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ packet records this arithmetic surface as a finite NameCert bridge between the modulus chapters chapter 292 and chapter 462, the regular operation chapters chapter 417 and chapter 461, and the real seal surface of chapter 38.

Definition 1485.1 (Cauchy-modulus arithmetic carrier)**.** *A* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *windows whose rational observations are read back through* $\mathsf{RegSeqRat}^{\uparrow}$ *rows.*
2. *μ_0 and μ_1 are the two source* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, each carrying its own precision request, threshold row, and late-pair ledger.*
3. *$\mu_\wedge$ is the shared-threshold* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row that places the two source schedules on one dyadic tolerance surface.*
4. *σ and π are the* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *consumer rows; they read only the two source schedules, the shared meet row, and the regular rational readbacks.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the split tolerances for the sum route, the product route, and the shared meet route.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger that names the inspected late observations for both source streams and both arithmetic consumers.*
7. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks obtained from the displayed windows, tolerance rows, and modulus schedules.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular rational sum or product handoff.*

9. *H records componentwise* hsame *transport for the two streams, two source schedules, shared meet row, sum row, product row, dyadic tolerance ledger, window ledger, regular readbacks, real seal rows, continuation rows, provenance, and local name.*
10. *C records displayed* Cont *replay from a precision request through the two source schedules and the shared meet row, then through D, W, R to σ, π, and E.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, RegularCauchyProduc $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common finite surface. It has no coordinate for quotient streams, selected limits, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier`
Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.TasteGate.CauchyModulusArithmeticTasteGate`
`single carrier alignment`

Theorem 1485.2 (Cauchy-modulus arithmetic NameCert obligations)**.** *For every accepted* CauchyModulusArithme *carrier $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed packet with two source modulus schedules, one shared meet row, two arithmetic consumer rows, tolerance ledgers, windows, readbacks, seal rows, transport, continuation, provenance, and local name;*
2. *classifier transport to a common* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *surface;*
3. *stability of μ_0, μ_1, $\mu_\wedge$, σ, and π under the componentwise* hsame *rows recorded in H;*
4. *exactness of the dyadic tolerance ledger D as the only source of split-tolerance and shared-threshold arithmetic reads;*
5. Cont *replay,* Pkg *provenance,* $\mathsf{Real}^{\uparrow}$ *handoff, and local NameCert rows for exactly the displayed packet.*

No NameCert obligation is stored outside those five rows.

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier namecert obligations`

Proof. Project the carrier of Definition 1485.1. The projection exposes precisely the two stream windows, two source modulus schedules, shared meet row, sum and product consumers, tolerance ledger, regular readbacks, real seal rows, transport rows, continuation rows, provenance row, and local name.

Classifier transport is componentwise. The rows in H move the two source schedules, the shared meet schedule, and the two arithmetic consumers to the same displayed stream, regular rational, and dyadic tolerance surface.

The dyadic exactness row is exhausted by D. The sum route reads split tolerances from D, the product route reads product bounds from D, and the meet route reads its shared tolerance from the same finite ledger.

Finally replay C and attach P, N. These rows route, source, and name the already displayed packet; they do not introduce quotient streams, selected limits, host real equality, or ambient completeness. Hence the five listed rows exhaust the local naming obligations. □

Theorem 1485.3 (Cauchy-modulus arithmetic carrier admission row)**.** *For every accepted* CauchyModulusArithmet *carrier* $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$*, the carrier-admission obligation for* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^\uparrow}$ *is exactly the displayed row admitting* $S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N$*. No source stream, modulus schedule, meet row, arithmetic consumer, tolerance ledger, window ledger, readback, seal, transport, replay, provenance, or local name may be read outside this finite carrier row.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier namecert obligation carrier`

Proof. Project the first obligation row of Definition 1485.2. It is the carrier-admission row for the tuple displayed in Definition 1485.1.

The listed coordinates already contain the two source streams, the two source moduli, the meet row, the sum and product consumers, the dyadic ledger, the window ledger, the regular readbacks, the Real seal, and the structural rows. Since the carrier definition has no further coordinate, admission is exhausted by this one finite row. □

Theorem 1485.4 (Cauchy-modulus arithmetic classifier transport row)**.** *For every accepted* CauchyModulusArithm *carrier* $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$*, classifier transport is permitted only through the common finite surface made from* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{CauchyModulus}^\uparrow$*,* $\mathsf{CauchyModulusMeet}^\uparrow$*, and* $\mathsf{DyadicRatCore}^\uparrow$*. In particular the classifier cannot compare arithmetic carriers by quotient stream equality, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier namecert obligation classifier`

Proof. Read the classifier-transport row of Definition 1485.2. It names exactly the stream, regular rational, source modulus, meet modulus, and dyadic tolerance surfaces.

The componentwise hsame rows in H move the two source streams, the two source schedules, the meet row, the arithmetic consumers, the tolerance ledger, and the readbacks to that same surface. The classifier therefore has only this transport route; no quotient, selected-limit, host-equality, or completeness coordinate is present to compare instead. □

Theorem 1485.5 (Cauchy-modulus arithmetic ledger nonescape row)**.** *For every accepted* CauchyModulusArithmet *carrier* $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$*, every completion-facing or* $\mathsf{Real}^\uparrow$ *consumer that reads the seal row* E *must first pass through the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^\uparrow}$ *row* N*. Thus* E *is a ledger endpoint for the displayed arithmetic route, not an escape hatch to quotient streams, selected limits, host real equality, or ambient completeness.*

Proof. Start from a consumer that asks for E. The carrier definition places E after the source schedules, shared meet, arithmetic consumers, dyadic ledger, windows, and regular readbacks.

Replay must then pass through C. The continuation row records the finite route from a precision request through the displayed source schedules and ledger rows before the Real seal is available.

Finally attach P and N. These rows record provenance and local naming for the same packet, so a consumer of E remains inside the displayed $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^\uparrow}$ route and cannot extract any unlisted completion or equality principle. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier ledger nonescape`

Theorem 1485.6 (Cauchy-modulus arithmetic sum and meet closure)**.** *Let A be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier. Every displayed precision request for the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row σ is answered by first reading the two source schedules μ_0, μ_1, then reading the shared meet row $\mu_\wedge$, and then reading the split dyadic tolerance ledger D. The resulting sum schedule is a finite* $\mathsf{CauchyModulus}^{\uparrow}$ *read over the same* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and its shared threshold is stable under the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *transport rows.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier sum meet route closure`

Proof. Start with a precision request for σ. Carrier admission exposes the two source schedules and the shared meet row before the sum row can be consumed.

Read D at the requested tolerance. Its split-tolerance rows provide the dyadic requests sent to μ_0 and μ_1, while its shared-threshold row is the tolerance read by $\mu_\wedge$.

Replay the Cont rows from C. They inspect only the windows in W and the regular rational readbacks in R, so the produced sum schedule remains a finite modulus read over the displayed packet.

Stability follows from H and the meet obligation surface. The shared meet row transports the two source schedules together, and the sum consumer can therefore reuse the same threshold packet after componentwise hsame transport. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier sum meet closure`

Theorem 1485.7 (Cauchy-modulus arithmetic Real consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *packet must factor through the displayed route*

$$S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N.$$

The consumer may read the regular sum or product handoff and then the real seal row E, but it cannot extract quotient stream equality, selected limits, host real equality, or ambient completeness from the arithmetic carrier.

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier real consumer boundary`

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ consumer route. The carrier has no real-facing coordinate before the regular rational readback rows R and the seal row E.

For the sum route, Definition 1485.6 forces the consumer through $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before σ can be sealed. For the product route, the carrier definition gives the same source schedules, shared meet row, dyadic ledger, windows, and readbacks before π can be sealed.

The remaining rows H, C, P, N only transport, replay, source, and name that finite route. Since the carrier exposes no quotient stream row, selected-limit row, host real equality row, or ambient-completeness row, no $\mathsf{Real}^{\uparrow}$ consumer can extract such data from the arithmetic packet. □

Theorem 1485.8 (Cauchy-modulus arithmetic shared threshold totality)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N),$$

every precision request reaching either arithmetic consumer has a displayed shared-threshold answer in the finite BHist *route. The answer is read through the two source stream rows S_0, S_1, the source*

schedules μ_0, μ_1, the meet row $\mu_\wedge$, the dyadic tolerance ledger D, the common window W, the regular rational readback R, the Real seal row E, and the structural rows H, C, P, N; no external threshold oracle or host real comparison supplies the answer.

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier shared threshold totality`

Proof. Project the carrier tuple from Definition 1485.1. The only precision schedules available to a consumer are the rows μ_0, μ_1 and the shared meet row $\mu_\wedge$, and all three are inside the same displayed BHist packet.

Read the dyadic tolerance ledger D next. Its split-tolerance entries address the two source schedules, while its shared entry addresses $\mu_\wedge$ before the sum row σ or product row π is consumed.

The window and readback rows W, R then record the finite observations used by the arithmetic consumer, and E, H, C, P, N only seal, transport, replay, source, and name that same route. Since the carrier has no coordinate for an external threshold oracle, the displayed shared-threshold answer is total for every accepted arithmetic request. □

Theorem 1485.9 (Cauchy-modulus arithmetic sum-product threshold stability)**.** *For every accepted* CauchyModulusArithmetic$^\uparrow$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N),$$

the threshold route used by the sum consumer σ and the threshold route used by the product consumer π remain the same finite shared-meet route after componentwise hsame *transport and displayed* Cont *replay. The stable route is exactly the route through $S_0, S_1, \mu_0, \mu_1, \mu_\wedge, D, W, R, E, H, C, P, N$, not a paired field-transport equivalence between independent arithmetic packets.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier sum product threshold stability`

Proof. Start with the sum route. By Definition 1485.6, σ is consumed only after the two source schedules, the shared meet row, the tolerance ledger, the common window, and the regular readback have been displayed.

The product route has no separate threshold coordinate in the carrier. Its consumer row π is placed over the same source schedules, the same $\mu_\wedge$ row, and the same D, W, R rows before the Real seal can be read.

Transport by H and replay by C preserve the displayed rows rather than creating another schedule. Thus both consumers remain attached to the same shared threshold under the naming rows P, N and the Real seal row E. □

Theorem 1485.10 (Cauchy-modulus arithmetic Real-seal modulus nonescape)**.** *For every accepted* CauchyModulusArithmetic$^\uparrow$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N),$$

any Real$^\uparrow$ *seal read from E must keep the modulus information inside the same finite route through $S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, H, C, P, N$. The seal may certify the displayed arithmetic readback, but it cannot export a free modulus, quotient-stream equality, selected real representative, host real equality, or ambient completeness claim.*

Proof. Apply the Real consumer boundary above. A Real-facing request reaches E only after the source schedules, the shared meet row, the arithmetic consumer row, the dyadic tolerance ledger, the common window, and the regular rational readback have already been displayed.

Use shared-threshold totality to locate every requested modulus answer inside the same BHist packet. The rows H, C, P, N transport, replay, source, and name that packet without adding a new modulus coordinate.

Finally inspect the carrier inventory. There is no row for quotient-stream equality, selected representatives, host real equality, ambient completeness, or an independent modulus export. Hence the Real seal cannot let the modulus data escape the displayed arithmetic route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier real seal modulus nonescape`

Theorem 1485.11 (Common tail window obligation)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$$

inside the displayed BHist *packet of Definition 1485.1, the common-tail window read for any* $\mathsf{Real}^{\uparrow}$ *or Completion consumer is forced through the route*

$$H, C, P, N : \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}.$$

Concretely, W may be consumed only after the two source schedules μ_0, μ_1 and the meet schedule $\mu_\wedge$ have been displayed, and any later seal row E must keep the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *provenance. No quotient stream, selected limit, host real equality, or ambient completeness row is exported by this common-tail obligation.*

Proof. Project the carrier tuple of Definition 1485.1. The common window row W occurs after the two source stream rows, the two source schedules, the shared meet row, the arithmetic consumers, and the dyadic tolerance ledger.

The hsame row H then transports those displayed rows componentwise. Thus a consumer may not replace the common window by a quotient stream or by a host comparison of completed values.

The replay row C records the finite route from a precision request through $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before the arithmetic consumers can reach E. The provenance and local naming rows P, N attach that same route to Pkg and $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$.

Hence the common tail window is a visible BHist obligation anchored in the displayed schedules and structural rows, not a hidden completeness principle. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier common tail window`

Theorem 1485.12 (Sum-product shared threshold obligation)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N),$$

the $\mathsf{RegularCauchySum}^{\uparrow}$ *row σ and the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *row π share the threshold supplied by $\mu_\wedge$ through the same* BHist, hsame, Cont, Pkg, *and* NameCert *route. Both consumers must read the two source modulus rows μ_0, μ_1, the meet row $\mu_\wedge$, the dyadic ledger D, the finite window row W, and the regular readback row R before either arithmetic result is sealed for* $\mathsf{Real}^{\uparrow}$. *No separate product threshold, quotient-stream comparison, selected limit, host real equality, or ambient completeness witness is available.*

Proof. Begin with the sum consumer. By Definition 1485.6, the row σ reads the two source schedules, then the shared meet row, and then the dyadic split tolerances in D before it reaches W and R.

Now turn to the product consumer. By Definition 1485.15, the row π reuses the same $\mu_0, \mu_1, \mu_\wedge, D, W, R$ route rather than opening a separate product comparison coordinate.

Transport through H and replay through C preserve this shared threshold for both arithmetic rows. The provenance and local name rows P, N then package the same BHist route as the public $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^\uparrow}$ surface.

Since the carrier contains no row for quotient streams, selected limits, host real equality, or ambient completeness, the shared threshold obligation is exactly the displayed meet route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier sum product shared threshold`

Theorem 1485.13 (Real seal schedule nonescape obligation)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^\uparrow$ *carrier*

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N),$$

the terminal $\mathsf{Real}^\uparrow$ *seal row* E *is reachable only through the finite schedule surface*

$$\mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, H, C, P, N$$

inside the displayed BHist *carrier. The seal preserves the same* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}}$ *route used by the source schedules, meet schedule, sum/product consumers, dyadic ledger,* $\mathsf{StreamName}^\uparrow$ *windows, and* $\mathsf{RegSeqRat}^\uparrow$ *readbacks. It exports no quotient stream, selected limit, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier real seal schedule nonescape`

Proof. Start from a consumer that requests the terminal seal row E. By Definition 1485.7, the consumer must pass through either the displayed sum route or the displayed product route before the Real-facing seal can be read.

The sum and product routes are already bound to the shared threshold by Definition 1485.12. Thus both routes expose $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before they reach the arithmetic consumer row that feeds E.

The common-tail obligation of Definition 1485.11 keeps the window and readback rows inside the same BHist surface. Transport H, replay C, provenance P, and local naming N therefore remain attached to the seal.

Consequently E is a schedule endpoint for the displayed arithmetic carrier. It is not an escape route to quotient streams, selected limits, host real equality, or ambient completeness. □

Theorem 1485.14 (Diagonal schedule dominates arithmetic consumers)**.** *Let* A *be an accepted* $\mathsf{CauchyModulusArithmetic}^\uparrow$ *carrier. Any displayed* $\mathsf{CauchyModulusDiagonal}^\uparrow$ *tail witness consumed by the sum or product route is admissible only after the two source schedules* μ_0, μ_1*, the shared meet row* $\mu_\wedge$*, the* $\mathsf{RegularCauchySum}^\uparrow$ *and* $\mathsf{RegularCauchyProduct}^\uparrow$ *rows, the* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledger* D*, the* $\mathsf{StreamName}^\uparrow$ *window ledger* W*, the* $\mathsf{RegSeqRat}^\uparrow$ *readbacks* R*, and the* $\mathsf{Real}^\uparrow$ *seal row* E *have all been exposed. Thus a diagonal tail witness is a downstream read of the displayed arithmetic route, not an independent schedule for choosing a hidden limit comparison.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier diagonal dominates sum product`

Proof. Read the carrier in the order of Definition 1485.1. The diagonal tail request has no coordinate before the two source schedules, the meet row, and the arithmetic consumer rows are present.

The tolerance and window rows then force the request through D and W. Their entries expose exactly the dyadic tolerance and finite stream windows from which the regular rational readbacks R are obtained.

Finally the consumer reaches E only through the regular sum or product handoff. The diagonal witness therefore remains below the displayed sum/product arithmetic route and cannot serve as a separate host limit comparison. □

Theorem 1485.15 (Product consumers reuse the meet threshold)**.** *Let A be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier. Every* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *consumer of A shares the same meet-threshold route as the* $\mathsf{RegularCauchySum}^{\uparrow}$ *consumer: it reads μ_0, μ_1, then $\mu_\wedge$, then the displayed ledger rows D, W, R, before reaching the product row π. In particular product compatibility is witnessed by the finite* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *route, not by a host comparison of completed limits.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier product meet compatibility`

Proof. Begin with a product request for π. Carrier admission places π after the two source schedules and the shared meet row, so the product consumer cannot bypass $\mu_0, \mu_1, \mu_\wedge$.

The same ledger rows used by the sum route are then read by the product route. The dyadic ledger D supplies the tolerance row, W supplies the inspected finite windows, and R supplies the regular rational readbacks.

Transport through H and replay through C preserve this common route. Since no completed-limit comparison coordinate is present in the carrier, product compatibility is exactly compatibility with the displayed meet threshold. □

Theorem 1485.16 (Real seal distribution through arithmetic rows)**.** *Let A be an accepted* $\mathsf{CauchyModulusArithmeti}$ *carrier. Every Real-facing consumer of the arithmetic-modulus packet distributes through* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *only by reading the finite rows*

$$\mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N.$$

No Real-facing consumer may distribute the seal through quotient streams, selected representatives, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier real seal distribution`

Proof. Start from a Real-facing request for the seal row E. By Definition 1485.7, the request must pass through the displayed sum or product handoff before the seal is available.

For the sum side, Definition 1485.6 supplies the source schedules, shared meet, tolerance ledger, windows, and readbacks. For the product side, Definition 1485.15 supplies the same meet-threshold route.

The structural rows H, C, P, N transport, replay, source, and name this finite route. Therefore distribution of the Real seal is distribution through the displayed arithmetic rows alone. □

Theorem 1485.17 (Cauchy-modulus arithmetic obligation closure)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier A, the obligation surface of the local packet is exhausted by carrier admission, classifier transport, the two source modulus schedules, the shared meet row, the sum and product consumer rows, dyadic tolerance exactness,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *row. No further obligation is supplied by quotient streams, selected limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier obligation closure`

Proof. Start with the carrier inventory of Definition 1485.1. It exposes exactly the two source streams, two modulus schedules, shared meet, arithmetic consumers, tolerance ledger, windows, readbacks, Real seal, and structural rows.

The local NameCert obligations are the five rows of Definition 1485.2. The first row admits the displayed packet, the second transports it through the classifier, the third stabilizes the source schedules and arithmetic consumers, the fourth fixes the dyadic tolerance ledger, and the fifth records replay, provenance, Real handoff, and local naming.

The sum route is locked to the shared meet by Definition 1485.6. Thus the arithmetic consumer rows cannot be read without the source schedules, shared threshold, dyadic tolerance ledger, finite windows, and regular rational readbacks.

The product and diagonal routes are locked to the same finite surface by Definition 1485.14 and Definition 1485.15. These routes cannot add a separate tail selector or a host comparison of completed limits.

Finally apply Definition 1485.7 and Definition 1485.16. The Real-facing consumer receives only the displayed route and cannot extract quotient streams, selected limits, host real equality, or ambient completeness. Hence the listed rows exhaust the obligation surface. □

Theorem 1485.18 (Cauchy-modulus arithmetic public exactness)**.** *Every Real-facing arithmetic read for an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *sum or product regular-Cauchy consumer is exact exactly when the two source schedules μ_0, μ_1, the shared meet threshold $\mu_\wedge$, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E are displayed in the same carrier together with H, C, P, N. Equivalently, the public sum/product read is accepted only through that one finite carrier surface and not through quotient streams, selected representatives, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier public exactness`

Proof. For the forward direction, start with an accepted Real-facing sum or product read. Apply Definition 1485.6 and Definition 1485.15; they force the sum and product consumers to reuse the two source schedules, the shared meet threshold, the dyadic tolerance ledger, finite windows, and regular rational readbacks.

The diagonal and common-tail rows keep the same public surface. Use Definition 1485.14 and Definition 1485.11 to place every diagonal or late-window read under the same $\mu_\wedge, D, W, R$ route.

Now apply Definition 1485.7 and Definition 1485.16. They put the Real-facing read behind E, H, C, P, N, so exactness is exactness of the displayed carrier rows.

Conversely, if all listed rows are displayed in the same carrier, the obligation closure theorem Definition 1485.17 supplies the finite NameCert surface. The same sum-meet, product-meet, diagonal, common-tail, boundary, and seal-distribution theorems then assemble the public read, with no coordinate left for the excluded host-level principles. □

Theorem 1485.19 (Cauchy-modulus arithmetic obligation scope package)**.** *The carrier, classifier, sum-meet closure, product meet compatibility, diagonal domination, Real-seal distribution, dyadic sum threshold, product tail bound, and Real-seal congruence rows jointly scope the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *NameCert surface over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *Every accepted arithmetic read must pass through the source schedules, shared meet threshold, arithmetic consumer rows, dyadic tolerance ledger, finite windows, regular rational readbacks, terminal Real seal, transport, replay, provenance, and local naming rows before it can expose a sum, product, or Real-facing output.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier obligation scope package`

Proof. Start with the carrier and classifier rows of Definition 1485.1 and Definition 1485.2. They expose the two source schedules, the shared meet row, the sum and product consumers, the dyadic ledger, the stream windows, the regular readback, the Real seal, and the structural rows.

For the arithmetic routes, use Definition 1485.6, Definition 1485.15, and Definition 1485.14. These force both $\mathsf{RegularCauchySum}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$ consumers to reuse $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before any output can be read.

For the Real-facing boundary, apply Definition 1485.16, Definition 1485.22, Definition 1485.23, and Definition 1485.24. These rows keep every terminal read attached to the same finite ledger and to the same H, C, P, N transport, replay, provenance, and local naming route.

Thus the scoped obligation package is finite: each named dependency is one displayed sibling or kernel row, and no quotient stream, selected limit, host real equality, or ambient completeness principle is introduced. □

Theorem 1485.20 (Cauchy-modulus arithmetic sum-product carrier admission)**.** *Every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *arithmetic packet exposes the sum row* σ*, product row* π*, diagonal schedule reads, meet threshold* $\mu_\wedge$*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *row* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *row* N *before any Real arithmetic consumer reads the result. The admission route is therefore a concrete finite* BHist *surface, not a quotient stream, selected limit, host real equality, or ambient completeness read.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier sum product carrier admission`

Proof. Project the carrier of Definition 1485.1. It already lists the sum and product consumers, the shared meet row, the dyadic tolerance ledger, the window ledger, the regular rational readbacks, the Real seal, and the structural rows.

Use the sum-meet closure of Definition 1485.6 and the product meet compatibility of Definition 1485.15. These two routes force σ and π to reuse $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before either arithmetic consumer can reach E.

The diagonal dominance theorem Definition 1485.14 keeps any diagonal schedule read downstream of the same rows. Finally Definition 1485.16 routes the Real-facing read through H, C, P, N, so the consumer receives only the displayed finite packet. □

Theorem 1485.21 (Cauchy-modulus arithmetic diagonal-meet stability)**.** *For every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *packet, any diagonal schedule read used by the sum or product consumer is stable only along the same meet threshold* $\mu_\wedge$ *and the same finite rows* D, W, R, E, H, C, P, N*.*

Thus diagonal refinement, meet-threshold transport, DyadicRatCore$^{\uparrow}$ *tolerance reads,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* Real$^{\uparrow}$ *sealing,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming remain one concrete route before any Real arithmetic consumer accepts the result.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier diagonal meet stability`

Proof. Begin with a diagonal schedule read. By Definition 1485.14, the read is admissible only after the source schedules, shared meet row, arithmetic consumers, tolerance ledger, windows, readbacks, and Real seal are visible.

Apply Definition 1485.15 to the product route and Definition 1485.6 to the sum route. Both consumers reuse the same meet threshold $\mu_\wedge$ and the same ledger rows D, W, R.

Transport and replay are then fixed by H and C. The provenance and local naming rows P, N record the same finite route, and the Real seal distribution theorem keeps the final arithmetic read inside it. Therefore diagonal stability is meet stability for the displayed packet, not a separate tail selector or host comparison. □

Theorem 1485.22 (Dyadic sum threshold obligation)**.** *Let A be an accepted* CauchyModulusArithmetic$^{\uparrow}$ *carrier. Every dyadic tolerance request consumed by the* RegularCauchySum$^{\uparrow}$ *row σ is answered by a finite threshold route through the source schedules μ_0, μ_1, the common-tail window, the shared meet row $\mu_\wedge$, and the split tolerance entries of D. The sum threshold therefore depends on the displayed* DyadicRatCore$^{\uparrow}$ *ledger and the* RegSeqRat$^{\uparrow}$ *readback row R, not on a quotient stream, selected limit, host real equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 1485.1. The sum consumer σ has access only to the two source schedules, the shared meet row, the dyadic ledger, finite stream windows, and regular rational readbacks.

Use the common-tail window of Definition 1485.11 to place the two late stream reads on one finite window surface. Then apply the sum-meet closure of Definition 1485.6 and the sum-product shared threshold of Definition 1485.12.

The split entries in D supply the dyadic tolerances read by σ, while R records the regular rational readback obtained from those windows. Since no other carrier coordinate can answer the request, the dyadic sum threshold is exactly this displayed finite route. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier dyadic sum threshold obligation`

Theorem 1485.23 (Product tail bound obligation)**.** *Let A be an accepted* CauchyModulusArithmetic$^{\uparrow}$ *carrier. Every product tail bound consumed by the* RegularCauchyProduct$^{\uparrow}$ *row π is witnessed inside the same meet threshold $\mu_\wedge$ and the same rows D, W, R used by the sum route. Product bounds may refine the diagonal schedule read, but they remain downstream of $\mu_0, \mu_1, \mu_\wedge$ and upstream of the* Real$^{\uparrow}$ *seal row E.*

Proof. Begin with a product request for π. The product-meet compatibility theorem Definition 1485.15 forces the request through $\mu_0, \mu_1, \mu_\wedge, D, W, R$ before the product row is available.

If the product request uses a diagonal tail witness, apply Definition 1485.14 and Definition 1485.21. These theorems keep diagonal refinement on the same meet threshold and the same finite ledger rows.

The product bound is then replayed through H and C and sourced by P, N. It cannot introduce a separate tail selector or compare completed limits, because the carrier has no coordinate for those principles. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier product tail bound`

Theorem 1485.24 (Real seal congruence obligation)**.** *Let A be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier. If two displayed arithmetic reads are transported by the componentwise* hsame *row H and share the same source schedules, meet threshold, dyadic ledger, stream windows, and regular rational readbacks, then every* $\mathsf{Real}^{\uparrow}$ *seal read through E is congruent along the same* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *route. The congruence is a seal boundary for the displayed packet, not a proof of quotient-real equality or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier real seal congruence`

Proof. Project the two displayed arithmetic reads to their common rows. By the carrier definition and the classifier transport row, H can transport only the stream windows, source schedules, meet row, sum and product rows, dyadic ledger, readbacks, seal row, replay, provenance, and local name.

Use Definition 1485.16 to route both Real-facing reads through σ or π and then through $\mu_0, \mu_1, \mu_\wedge, D, W, R, E, H, C, P, N$. The Real seal schedule nonescape obligation Definition 1485.13 keeps the seal attached to that same finite route.

Thus congruence of the seal is exactly congruence along the displayed transport, replay, provenance, and local NameCert rows. No quotient-real equality or completeness principle is exported by the argument. □

Theorem 1485.25 (Cauchy-modulus arithmetic uniform product budget)**.** *Let A be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier. Any product consumer that reads a requested* $\mathsf{Real}^{\uparrow}$ *tolerance must first pass through the shared meet threshold $\mu_\wedge$, then the product tail-bound ledger over D, W, R, and then the Real-seal congruence row E, H, C, P, N. Consequently the exported product budget is uniform over the same* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks that the carrier already displays; it cannot be selected from a separate tail oracle, quotient real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier uniform product budget`

Proof. Project the carrier of Definition 1485.1. The product row π can read only the source schedules, the shared meet threshold, the dyadic ledger, the stream windows, the regular-rational readbacks, and the structural rows of the displayed packet.

Apply the product tail bound obligation Definition 1485.23. It places every product tail read downstream of $\mu_0, \mu_1, \mu_\wedge, D, W, R$ and upstream of the $\mathsf{Real}^{\uparrow}$ seal.

Use diagonal-meet stability, Definition 1485.21, to keep any diagonal refinement on the same shared threshold. Then apply Real seal congruence, Definition 1485.24, so the final seal read remains attached to E, H, C, P, N.

The budget exported to the product consumer is therefore the unique displayed route through the meet threshold, tail ledger, readback windows, seal, transport, replay, provenance, and local NameCert rows. No other carrier coordinate can supply a product budget. □

Theorem 1485.26 (Cauchy-modulus arithmetic scope window reuse)**.** *Every* $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier must reuse the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *row before exporting a shared threshold. The shared threshold is therefore scoped by the existing finite rows of the packet, not by a separate tail oracle, quotient-stream selector, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier scope window reuse`

Proof. Project the accepted carrier $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$. The displayed stream windows, dyadic ledger, regular-rational readbacks, Real seal, transport, replay, provenance, and naming rows are already part of the finite packet before any arithmetic consumer reads a threshold.

Apply the sum-meet closure and uniform product budget theorems. They force sum and product consumers through the same $\mu_\wedge$ threshold together with D, W, R, E, H, C, P, N.

For a downstream $\mathsf{Real}^{\uparrow}$ seal, use Real-seal distribution and congruence. Those results keep the seal read attached to the same displayed rows, so the consumer cannot select a new threshold from quotient streams, host real equality, or ambient completeness. □

Theorem 1485.27 (Cauchy-modulus arithmetic scoped evidence route)**.** *The scoped* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *marker is read through the carrier* $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusArithmetic}^{\uparrow}}$ *obligations, and the finite window rows* D, W, R, E, H, C, P, N*. In particular, scoped closure is supported by the two source* $\mathsf{StreamName}^{\uparrow}$ *windows, the two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *threshold, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *consumers, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows.*

Proof. Project the carrier of Definition 1485.1. It lists the exact tuple $S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N$ that the hub marker reads through the child file.

Apply the NameCert obligations theorem Definition 1485.2. Its carrier, classifier, stability, dyadic-exactness, replay, provenance, seal, and local-name rows bind the finite scope to those displayed coordinates.

Finally use scope-window reuse, Definition 1485.26. It forces sum, product, and Real-seal consumers to reuse D, W, R, E, H, C, P, N, so the scoped marker is tied to the child evidence rather than to a detached hub assertion. □

Theorem 1485.28 (Cauchy modulus arithmetic windowed product distributivity)**.** *Let* A *be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *packet whose sum and product modulus schedules expose the same* $\mathsf{StreamName}^{\uparrow}$ *window. Any Real consumer that reads the product distributivity budget for those schedules must route the comparison through the displayed* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *shared threshold, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it receives the product budget. The common window is therefore the only route by which distributivity of the finite product budget reaches a downstream Real consumer.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier windowed product distributivity`

Proof. Start with the NameCert obligations of Definition 1485.2. They expose the two source schedules, the shared threshold row, the arithmetic consumer rows, the dyadic tolerance ledger, the readback rows, the Real seal, and the structural transport rows as one packet.

Apply the sum and product shared-threshold route of Definition 1485.6 together with the product tail bound of Definition 1485.23. These clauses keep the product budget downstream of the same $\mathsf{StreamName}^{\uparrow}$ window and the same $\mathsf{CauchyModulusMeet}^{\uparrow}$ threshold used by the sum schedule.

Finally use the uniform product budget of Definition 1485.25 and Real-seal congruence, Definition 1485.24. The Real consumer receives only the route through the displayed tolerance ledger, readback rows, terminal seal, transport, replay, provenance, and local naming, so the common-window product distributivity obligation is exhausted by the carrier surface. □

Theorem 1485.29 (Cauchy-modulus arithmetic tail-meet exhaustion)**.** *Let $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E$ be an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *carrier. Every accepted carrier consumer that reads a* $\mathsf{RegularCauchySum}^{\uparrow}$ *row, a* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *row, or the downstream* $\mathsf{Real}^{\uparrow}$ *seal must expose the two source schedules μ_0, μ_1, the shared meet row $\mu_\wedge$, the dyadic ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R, and the structural rows H, C, P, N before it exports any threshold, product budget, or seal. Thus the sum, product, and Real-facing reads exhaust the same tail-meet surface rather than three separate consumer routes.*

Proof. Start with scope-window reuse from Definition 1485.26. It forces every sum, product, or $\mathsf{Real}^{\uparrow}$ seal consumer to reuse the displayed windows, dyadic ledger, readbacks, seal row, transport, replay, provenance, and local naming rows of the accepted packet.

For product-facing reads, apply the uniform product budget theorem Definition 1485.25. The product budget is downstream of the shared meet row and the same D, W, R, E, H, C, P, N surface, so it cannot be supplied by a separate tail oracle or a quotient stream.

Finally use Definition 1485.28. Its common-window route keeps the product-distributivity budget attached to the shared $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{CauchyModulusMeet}^{\uparrow}$ threshold, dyadic ledger, readbacks, and terminal $\mathsf{Real}^{\uparrow}$ seal. Repacking these three routes gives the claimed exhaustion of all downstream sum, product, and Real-seal reads through the single displayed meet-and-tail surface. □

Falsifiable prediction: An accepted $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ consumer that reaches a $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the two displayed $\mathsf{CauchyModulus}^{\uparrow}$ schedules, the $\mathsf{CauchyModulusMeet}^{\uparrow}$ shared-threshold row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{StreamName}^{\uparrow}$ windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks refutes the carrier surface.

Independence witness: cauchymodulus, cauchymodulusmeet, regularcauchysum, regularcauchyproduct, real: no listed sibling supplies the whole arithmetic packet $A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$, because each sibling contributes only a source schedule, a shared threshold, one arithmetic consumer, or the downstream real seal while this chapter binds the schedule-combination route with transport, replay, provenance, and local naming.

Theorem 1485.30 (Cauchy modulus arithmetic public export)**.** *Every public Real or Completion read of an accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *packet is exported through the same finite carrier rows used by the local obligation surface: the two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules, the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *threshold, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *consumer rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P, N. Thus the public object surface*

is the displayed arithmetic carrier itself, not quotient streams, selected representatives, host real equality, or ambient completeness.

Proof. Start from the obligation closure of Definition 1485.17. It identifies the local NameCert surface with the displayed source schedules, shared meet row, arithmetic consumer rows, dyadic tolerance ledger, finite windows, regular readbacks, Real seal, transport, replay, provenance, and local naming.

Apply the public exactness row of Definition 1485.18. A public sum or product read is accepted only when those same rows are present in one carrier, so the public export does not introduce a second arithmetic object.

Finally use the scope package of Definition 1485.19. It keeps the exported packet inside the same BHist route over CauchyModulus$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, RegularCauchySum$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert. □

Closure status for NameCert$_{\text{CauchyModulusArithmetic}^{\uparrow}}$.

Constructive story: A CauchyModulusArithmetic$^{\uparrow}$ packet is generated from two StreamName$^{\uparrow}$ windows, two CauchyModulus$^{\uparrow}$ schedules, one CauchyModulusMeet$^{\uparrow}$ shared-threshold row, RegularCauchySum$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ consumer rows, a DyadicRatCore$^{\uparrow}$ tolerance ledger, RegSeqRat$^{\uparrow}$ readbacks, a Real$^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusArithmetic}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1485.31, Definition 1485.24, Definition 1485.25, Definition 1485.26, Definition 1485.27, Definition 1485.29, and Definition 1485.30 bind the finite modulus-arithmetic scope and public export surface over CauchyModulus$^{\uparrow}$, CauchyModulusMeet$^{\uparrow}$, RegularCauchySum$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusArithmetic}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticCarrier\_public\_exactness`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, host real equality, ambient completeness, or bridge checking.

Upgrade path: Bridge closure requires a checked standard-bridge theorem for the public arithmetic carrier surface.

Theorem 1485.31 (Cauchy-modulus arithmetic scoped route)**.** *Every accepted* CauchyModulusArithmetic$^{\uparrow}$ *packet scopes its arithmetic reads through the displayed carrier of Definition 1485.1, the NameCert obligation rows of Definition 1485.2, and the common-tail and scope-window routes of Definition 1485.11 and Definition 1485.26. Thus every sum, product, or Real-seal consumer must read* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows inside the same finite packet before exporting an arithmetic handoff.*

The scoped route cannot import quotient stream equality, selected limits, host natural-number search, host real equality, or ambient Real$^{\uparrow}$ *completeness. Those endpoints are absent from the carrier and from the child obligation rows that the hub inputs before this route.*

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticScopedRoute`

Proof. Project the carrier

$$A = (S_0, S_1, \mu_0, \mu_1, \mu_\wedge, \sigma, \pi, D, W, R, E, H, C, P, N)$$

from Definition 1485.1. The rows D, W, R, H, C, P, N supply the dyadic tolerance ledger, finite stream windows, regular rational readbacks, componentwise transport, displayed replay, provenance, and local naming before an arithmetic consumer can read σ, π, or E.

Apply the NameCert obligation surface and the common-tail window obligation. These force every precision request through the displayed source schedules, shared meet threshold, common window, readback rows, and structural rows of the same BHist packet.

Finally use scope-window reuse. It keeps the Real-seal and arithmetic consumers on the existing finite rows rather than on a host search or quotient-stream comparison. Since the carrier has no coordinate for selected limits, host natural-number search, host real equality, quotient stream equality, or ambient completeness, the scoped route exports none of those endpoints. □

Theorem 1485.32 (Cauchy-modulus arithmetic diagonal meet route)**.** *Every accepted* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *packet whose consumers read both sum and product arithmetic must pass through the displayed meet schedule* $\mu_\wedge$*, the common finite window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal* E *before either arithmetic handoff is exported. Thus a paired consumer reads*

$$\mu_\wedge, W, D, R, E \longmapsto \sigma, \pi$$

inside the same finite BHist *packet, with the structural rows* H, C, P, N *preserving transport, replay, provenance, and local naming. The route exports no quotient stream equality, host natural-number search, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host real equality.*

Proof. Start with the scoped route of Definition 1485.31. It already forces every arithmetic read through the displayed carrier rows and the NameCert obligation surface before a sum, product, or Real-seal consumer can export a handoff.

Now restrict to a consumer that reads both σ and π. The carrier of Definition 1485.1 contains the meet schedule $\mu_\wedge$ and the common finite window W exactly for this paired read, so the consumer cannot use separate hidden schedules for the two arithmetic branches.

The dyadic and readback rows are then common. The ledger D supplies the tolerance accounting, R supplies the regular rational readback, and E seals the Real-facing endpoint after those finite rows have already been displayed.

Finally H, C, P, N keep the paired route inside one packet. Since the existing scoped route refuses quotient stream equality, host natural-number search, selected limits, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness, the diagonal meet route exports none of those endpoints. □

Lean4 ✓ `BEDC.Derived.CauchyModulusArithmeticUp.CauchyModulusArithmeticDiagonalMeetRoute`

Chapter 1486

A Concrete Naming Certificate for $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$

Following Bishop and Bridges' constructive-analysis use of explicit moduli, the $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ packet records pointwise continuity data for Cauchy names as a visible finite schedule rather than as a host continuity predicate. It sits between the explicit Cauchy schedule surface of chapter 292, the finite Cauchy-name transformer of chapter 1401, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the compact uniform consumer boundary of chapter 369. The pointwise packet records the threshold route for one displayed source Cauchy name and its target tolerances; it is not compactness, a uniform modulus over all inputs, or a host function equality principle.

Definition 1486.1 (Cauchy modulus of continuity carrier)**.** *A* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, \mu, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window surface whose finite rational observations are inspected before a target tolerance is answered.*
2. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *window surface promised by the pointwise Cauchy-name transformer.*
3. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule assigned to each requested target tolerance, including the finite source-window bound that the consumer must inspect.*
4. *F is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer row that consumes S, μ and produces the displayed target windows T.*
5. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks for the source and target windows, with the target readback reached only through F and μ.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the target* $\mathsf{RegSeqRat}^{\uparrow}$ *readback after the displayed schedule has been replayed.*
7. *H records componentwise* hsame *transport for S, T, μ, F, R, E and for the continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a target tolerance request through the threshold schedule μ, the source window S, the transformer row F, the target window T, the regular readbacks R, and the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common finite source-window, target-window, threshold, transformer, readback, and seal surface. It has no coordinate for quotient streams, host function equality, selected limits, compactness, a uniform modulus, or ambient completeness.

Theorem 1486.2 (Cauchy modulus of continuity NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier $K = (S, T, \mu, F, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed source windows, target windows, threshold schedule, transformer row, regular readbacks, seal row, transport, continuation, provenance, and local name;*
2. *classifier transport to a common* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *finite surface;*
3. *stability of S, T, μ, F, R, E under the componentwise* hsame *rows recorded in H;*
4. *exactness of μ as the only threshold route read by the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *consumer;*
5. Cont *replay,* Pkg *provenance,* $\mathsf{Real}^{\uparrow}$ *seal factorization, and local NameCert rows for exactly the displayed packet.*

No NameCert obligation is stored outside those five rows.

Proof. Project the carrier of Definition 1486.1. The projection exposes exactly the source windows, target windows, threshold schedule, transformer row, regular readbacks, real seal, transport rows, continuation rows, provenance row, and local name.

Classifier transport is componentwise. The rows in H move the source and target windows, the threshold schedule, the transformer row, the readbacks, and the seal row to the same displayed finite surface.

The threshold exactness row is exhausted by μ. The transformer row F reads a target tolerance by first reading the schedule μ, then inspecting the named source windows S, and then producing the target windows T.

Finally replay C and attach P, N. These rows route, source, and name the already displayed packet; they do not introduce quotient streams, host function equality, selected limits, compactness, a uniform modulus, or ambient completeness. Hence the five listed rows exhaust the local naming obligations. □

Lean4 ✓ `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityCarrier namecert obligations`

Theorem 1486.3 (CauchyContinuousMap threshold handoff)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier $K = (S, T, \mu, F, R, E, H, C, P, N)$, the displayed schedule μ is the only threshold route used by a* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *consumer. Any read of the target window T must pass through the source window S, the schedule μ, the transformer row F, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R, and the displayed* Cont, Pkg, *and local* NameCert *rows.*

Proof. Start with a target tolerance request. Carrier admission exposes the schedule μ before the transformer row F can be consumed.

Replay the displayed continuation row C. The route sends the tolerance request to μ, obtains the finite source-window threshold, reads the corresponding source observations in S, and then lets F produce the target windows T.

The readback row R is reached only after that transformer route. Transport through H preserves the same order of dependencies, and P, N merely record provenance and local naming.

Since the carrier contains no second threshold coordinate, no quotient stream, and no host function row, a $\mathsf{CauchyContinuousMap}^{\uparrow}$ consumer has no alternate route by which to obtain T. □

Lean4 ✓ `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityCarrier cauchy continuous map threshold handoff`

Theorem 1486.4 (Cauchy modulus of continuity scoped route)**.** *Every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *read factors through the* $\mathsf{StreamName}^{\uparrow}$ *source and target windows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule, the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any compact-uniform consumer may read it. The route exports exactly the pointwise Cauchy-name continuity surface over $S, T, \mu, F, R, E, H, C, P, N$, with no compactness coordinate, uniform modulus over all inputs, host function equality, quotient stream equality, selected-limit witness, or ambient completeness principle.*

Proof. Start from the carrier definition Definition 1486.1. It fixes the finite rows $S, T, \mu, F, R, E, H, C, P, N$ and gives the packet no coordinate outside source windows, target windows, threshold schedule, transformer route, regular readbacks, real seal, transport, replay, provenance, and local naming.

Apply the NameCert obligations of Definition 1486.2. They bind carrier admission, classifier transport, componentwise stability, threshold exactness, replay, provenance, seal factorization, and local naming to those same rows.

Use the CauchyContinuousMap threshold handoff of Definition 1486.3. Any target-window read must pass through S, μ, F, T, R before reaching the structural rows, so a compact-uniform consumer cannot bypass the displayed pointwise schedule.

Finally read the $\mathsf{Real}^{\uparrow}$ seal through E and the structural rows H, C, P, N. Since the route has no compactness, uniform-modulus, host-function, quotient-stream, selected-limit, or ambient-completeness coordinate, the listed rows form the scoped route for every accepted consumer read. □

Lean4 ✓ `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityCarrier scoped route`

Theorem 1486.5 (Cauchy modulus of continuity finite-window monotonicity)**.** *Let $K = (S, T, \mu, F, R, E, H, C, P, N)$ and $K^+ = (S, T^+, \mu^+, F, R^+, E^+, H^+, C^+, P^+, N^+)$ be accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carriers with the same source window S, the same transformer row F, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the same* $\mathsf{Real}^{\uparrow}$ *seal route. Suppose that, for one displayed target tolerance, the schedule μ^+ asks for a larger finite source-window bound than μ while preserving the transformer, readback, and seal rows. Then every target-window read admitted by K through μ is also admitted by K^+ through μ^+, using the same* Cont, Pkg, *and local* NameCert *rows after the larger source-window inspection is replayed.*

Proof. Project both carriers with Definition 1486.1. The shared rows identify the same source window, transformer route, regular-rational readback route, and real-seal route; only the displayed source-window threshold is enlarged.

Use the threshold handoff of Definition 1486.3. It says that a target-window read has a unique displayed route through the schedule before the transformer row F can produce T.

Replay the larger schedule μ^+. Its source-window inspection contains the smaller inspection required by μ and then continues through the same transformer, readback, and seal rows. Thus the target-window read admitted by K is admitted again by K^+ after the larger finite source window has been displayed.

Finally transport through H^+ and attach C^+, P^+, N^+. These rows preserve the same continuation, provenance, and local NameCert route, so the enlarged schedule creates no alternate host function, quotient stream, selected limit, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityCarrier window monotonicity`

Theorem 1486.6 (Cauchy modulus of continuity schedule stability)**.** *Let $K = (S, T, \mu, F, R, E, H, C, P, N)$ and $K' = (S', T', \mu', F', R', E', H', C', P', N')$ be accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carriers. Suppose the S, T, F, R, E rows of K and the S', T', F', R', E' rows of K' are related componentwise by the displayed H, H'* hsame *transports, and suppose μ and μ' answer the same target tolerance with transported finite source-window bounds. Then every target-window read admitted through μ is admitted through μ' after displayed* Cont *replay, with the same* Pkg *and local* NameCert *route.*

Proof. Project both carriers through Definition 1486.1. The displayed rows expose only the source windows, target windows, schedules, transformer rows, readback rows, Real seals, transport, replay, provenance, and local naming data.

Use the componentwise transports H and H'. They identify the S, T, F, R, E surface of K with the S', T', F', R', E' surface of K' at the transported finite source-window and target-window bounds.

Replay C and C' through the schedules before reading the transformer row. The schedule rows μ and μ' answer the same target tolerance, so the target-window request reaches the transformer surface only after the matching finite source-window inspection has been displayed.

Apply the threshold handoff of Definition 1486.3 and the finite-window monotonicity of Definition 1486.5. They keep the admitted target-window read inside the transported schedule route and allow the replay to use the transported finite source-window bound.

Finally attach the same provenance and local naming route. No quotient stream, host function equality, selected limit, compactness, or completeness coordinate is introduced, so schedule transport preserves the accepted read. □

Theorem 1486.7 (Cauchy modulus of continuity Real consumer boundary)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier $K = (S, T, \mu, F, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$ *consumer factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R and the* $\mathsf{Real}^{\uparrow}$ *seal row E. The consumer surface exposes only the source and target windows, the schedule μ, the transformer row F, the readbacks R, the seal E, and the recorded* hsame, Cont, Pkg, *and* NameCert *rows; it exposes no quotient stream, host function equality, selected limit, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityRealConsumerBoundary`

Proof. Read the carrier inventory. The only real-facing coordinate is E, and it is listed as the seal row reached from the target $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the handoff theorem Definition 1486.3. It shows that the target readback has already passed through S, μ, F, and T before R can be consumed by a real-facing route.

Replay the seal route through C and transport it through H. The package row P and the local name N certify that the same displayed finite rows source the read.

Thus every $\mathsf{Real}^{\uparrow}$ consumer factors through R and E. The packet has no coordinate from which to read a quotient stream, host function equality, selected limit, or ambient completeness principle. □

Theorem 1486.8 (Cauchy modulus of continuity obligation-closure package)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier* $K = (S, T, \mu, F, R, E, H, C, P, N)$*, the carrier definition, the local NameCert obligations, the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *threshold handoff, finite-window monotonicity, and the* $\mathsf{Real}^{\uparrow}$ *consumer boundary jointly supply the full obligation surface over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{CauchyContinuousMap}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The package exposes exactly the displayed source windows, target windows, threshold schedule, transformer route, regular readbacks, real seal, transport rows, continuation replay, provenance, and local naming data.*

Proof. Start with Definition 1486.1. It fixes the tuple $K = (S, T, \mu, F, R, E, H, C, P, N)$ and states that every visible coordinate is one of the named BEDC rows.

Apply Definition 1486.2. The local NameCert surface is exhausted by carrier admission, classifier transport, componentwise stability, threshold-route exactness, replay, provenance, seal factorization, and local naming.

Use Definition 1486.3 and Definition 1486.5 for the consumer side. They force every target-window read to pass through S, μ, F, T, R and show that enlarging a finite source window preserves the same displayed route.

Finally apply Definition 1486.7. The real consumer reaches only the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal after the threshold route has been replayed. Hence the package excludes uniform continuity, compactness, quotient streams, host function equality, selected limits, and ambient completeness. □

Theorem 1486.9 (Cauchy modulus of continuity obligation exactness)**.** *For every accepted* CauchyModulusOfConti *carrier* $K = (S, T, \mu, F, R, E, H, C, P, N)$*, every* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the same displayed threshold schedule* μ*, source window* S*, transformer row* F*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E*. The obligation surface has no alternate compactness coordinate, host function equality row, selected-limit witness, or second uniform-modulus coordinate.*

Proof. Project the carrier of Definition 1486.1. The rows $S, T, \mu, F, R, E, H, C, P, N$ are the only visible coordinates, so any consumer read begins with the displayed schedule, window, transformer, readback, and seal surface.

Use Definition 1486.2. Its local NameCert clauses bind carrier admission, classifier transport, threshold route exactness, replay, provenance, and local naming to those same rows.

Apply the threshold handoff of Definition 1486.3, the finite-window monotonicity of Definition 1486.5, and the schedule stability of Definition 1486.6. Together they force every $\mathsf{CauchyContinuousMap}^{\uparrow}$ threshold read to pass through μ, S, F, T, and R with the same transported finite-window route.

Finally apply Definition 1486.7. The $\mathsf{Real}^{\uparrow}$ consumer can reach the seal only after the readback route has passed through the displayed threshold schedule, so no compactness, host function equality, selected-limit, or alternate uniform-modulus row is available. □

Theorem 1486.10 (Cauchy modulus threshold transport)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier* $K = (S, T, \mu, F, R, E, H, C, P, N)$*, the displayed threshold schedule* μ *is transported through the* $\mathsf{StreamName}^{\uparrow}$ *source window* S*, the target* $\mathsf{StreamName}^{\uparrow}$ *window* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance discipline used by those readbacks, and the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer row* F *before any Real-facing consumer reads the seal row* E*.*

Proof. Project the carrier of Definition 1486.1. The threshold schedule μ is displayed between the source and target finite-window rows and the transformer row, so it is visible before the regular rational readback and the Real seal.

Apply the NameCert obligations of Definition 1486.2. They bind carrier admission, classifier transport, componentwise stability, and threshold-route exactness to the same finite surface.

Use the handoff theorem Definition 1486.3. It forces each target-window read to pass through S, μ, F, T, R before the Real consumer can reach E. □

Theorem 1486.11 (Cauchy modulus Real seal nonescape)**.** *For every accepted* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *carrier* $K = (S, T, \mu, F, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reachable only after the modulus row* μ*, threshold route, finite-window rows* S, T*, readback row* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N *have been displayed.*

Proof. Start from Definition 1486.7. It places every Real-facing read behind the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks and the seal row of the same carrier.

Now use threshold transport from Definition 1486.10. The target readback cannot appear before S, μ, F, T and the dyadic tolerance route used by R are visible.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the already displayed threshold route; none supplies a bypass around μ, F, R or a detached Real seal. □

Theorem 1486.12 (Cauchy modulus local NameCert surface)**.** *The rows*

$$S, T, \mu, F, R, E, H, C, P, N$$

exhaust the local $\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}$ *obligations for an accepted packet. The packet excludes host continuity, selected limits, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite tuple in Definition 1486.1. Its coordinates are exactly the source and target windows, threshold schedule, transformer, readbacks, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1486.2, Definition 1486.10, and Definition 1486.11. Together they show that carrier admission, classifier transport, threshold-route exactness, Real-seal factorization, transport, replay, provenance, and local naming are all read from the displayed rows.

Since the carrier contains no host continuity predicate, selected limit, quotient stream, countable-choice schedule, or ambient completeness row, those objects are outside the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ packet is generated from source and target $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{CauchyModulus}^{\uparrow}$ threshold schedule for each requested target tolerance, a $\mathsf{CauchyContinuousMap}^{\uparrow}$ transformer row, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1486.1, Definition 1486.2, Definition 1486.4, and Definition 1486.12 bind the local route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows for the pointwise Cauchy-name continuity surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusOfContinuity}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.CauchyModulusOfContinuityUp.CauchyModulusOfContinuityCarrier\_cauchy\_continuous\_map\_`

Bridge status: none

Not claimed: This chapter does not close uniform continuity, compactness, quotient stream equality, host function equality, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public-object surface for the same $S, T, \mu, F, R, E, H, C, P, N$ threshold handoff and its Real-consumer reads.

Chapter 1487

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$

Following Bishop and Bridges' constructive-analysis treatment of regular Cauchy names, the $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ packet records explicit finite tail-estimate data before a $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits between the dyadic tolerance surface of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the real seal of chapter 38, and the modulus-facing route of chapter 1423. The packet names the finite rows a consumer must inspect for a requested tail tolerance; it is not a quotient stream, a selected limit, a host real equality principle, or an ambient completeness theorem.

Definition 1487.1 (Regular Cauchy tail estimate carrier)**.** *A* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus-threshold row. For each requested tail tolerance it records the finite index after which the regular Cauchy estimate is allowed to be read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose finite observations are inspected at and beyond the threshold stored in M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic error budget used by the tail estimate.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the window W under the threshold and tolerance rows M, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that may be consumed only after the threshold, window, tolerance, and regular readback rows have been replayed.*
6. *H records componentwise* hsame *transport for M, W, D, R, E and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through M, W, D, R, and then E.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed modulus threshold, finite window, dyadic tolerance, regular readback, real seal, continuation, provenance, and local naming rows to a common finite surface. It has no coordinate for quotient streams, host real equality, selected limits, choice, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateTasteGate single carrier alignment`

Theorem 1487.2 (Regular Cauchy tail estimate NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed modulus threshold, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local name;*
2. *classifier transport to a common* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite surface;*
3. *stability of* M, W, D, R, E *under the componentwise* hsame *rows recorded in* H*;*
4. *seal-route exactness requiring every real-facing read to pass through* M, W, D, R, E *in that order;*
5. *refusal rows excluding quotient streams, host real equality, choice, trichotomy, selected limits, open-cover compactness, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateNameCertObligations`

Proof. Project the carrier of Definition 1487.1. The projection exposes exactly the modulus threshold, finite window, dyadic tolerance, regular readback, real seal, transport rows, continuation row, provenance row, and local naming row.

Classifier transport is componentwise. The rows in H move the five mathematical coordinates M, W, D, R, E and the administrative coordinates C, P, N to the same finite comparison surface.

Seal-route exactness is read from the displayed continuation row. A tail tolerance request first reaches the threshold row M, then the finite window W, then the dyadic tolerance row D, then the regular readback R, and only then the real seal E.

The refusal rows are exhausted by the carrier inventory. Since the packet has no coordinate for quotient stream equality, host real equality, choice, trichotomy, selected limits, open-cover compactness, or completeness, no local NameCert obligation can export those data. □

Theorem 1487.3 (Regular Cauchy tail estimate Real seal route)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal consumer of* E *factors through the modulus threshold* M*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E *before it can inspect the transport, continuation, provenance, and local naming rows* H, C, P, N*. No consumer route can bypass* M, W, D, R, E *or read a quotient stream, a selected host limit, host real equality, choice, trichotomy, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier real seal route`

Proof. Start with a real-facing consumer of E. Carrier admission lists E only as the seal row reached from the regular readback coordinate R.

Replay the continuation row C. Its route sends a tolerance request first to the threshold row M, uses M to choose the finite window reads in W, checks the dyadic tolerance row D, and then reaches the regular readback R.

Transport through H preserves this order of displayed dependencies. The package row P and local name N record provenance and naming for the same finite packet; they do not add a second seal coordinate.

Thus every $\mathsf{Real}^{\uparrow}$ seal consumer factors through M, W, D, R, E before H, C, P, N. The carrier contains no coordinate from which a bypass route could obtain quotient streams, selected limits, host equality, choice, trichotomy, open-cover compactness, or completeness. □

Definition 1487.4 (Regular Cauchy tail estimate tail-envelope lock)**.** *The tail-envelope lock of an accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$ *is the displayed route*

$$M, W, D, R \longmapsto \mathsf{RegularCauchyTailEnvelope}^{\uparrow}$$

that reads the modulus threshold, finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the* $\mathsf{Real}^{\uparrow}$ *seal row is inspected. The lock records that the Real-seal route is available only after the finite tail-envelope surface has been exposed.*

Theorem 1487.5 (Regular Cauchy tail estimate tail-envelope lock)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, any consumer that exposes the* $\mathsf{Real}^{\uparrow}$ *seal row* E *must first expose the finite tail-envelope route through* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*. No selected limit, quotient stream, countable-choice schedule, host equality, open-cover compactness, or ambient completeness principle can replace that route.*

Proof. Project the carrier of Definition 1487.1. The rows M, W, D, R, E are the displayed mathematical coordinates, and E occurs only after the threshold, window, tolerance, and regular readback rows.

Apply Definition 1487.2. The local certificate forces classifier transport and stability for M, W, D, R, E componentwise, so the tail estimate cannot detach the seal row from the finite tail data.

Now apply Definition 1487.3. Every real-facing read of E factors through M, W, D, R before the structural rows H, C, P, N are inspected. The definition of the tail-envelope lock identifies that preceding route with the $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ handoff.

Since none of the excluded objects is a coordinate of the carrier, no bypass can replace the finite envelope route. The seal remains downstream of the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier tail envelope lock`

Theorem 1487.6 (Regular Cauchy tail estimate tail dominance)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$ *whose tail-envelope route is locked, every later* $\mathsf{Real}^{\uparrow}$*-facing comparison exposed by the packet is dominated by the displayed finite tail surface. The comparison can read only the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the tail-envelope row, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *row* N*.*

No later comparison exports a host limit, quotient stream equality, selected tail, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier tail dominance`

Proof. Project the carrier of Definition 1487.1. The only Real-facing seal coordinate is E, and Definition 1487.3 places every read of E downstream of the threshold, window, tolerance, and regular readback rows M, W, D, R.

Apply Definition 1487.5. The lock identifies that preceding route with the finite tail-envelope surface through $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$, so any later comparison must reuse the same displayed window, tolerance ledger, and readback.

The structural rows do not loosen the dominance claim. The transport row H moves only the displayed coordinates, the replay row C preserves the same route, and P, N record provenance and local naming for that finite packet.

Thus every later $\mathsf{Real}^{\uparrow}$-facing comparison remains dominated by the displayed tail surface. Since no host limit, quotient stream equality, selected tail, or ambient completeness coordinate occurs in the carrier, none can be exported. □

Theorem 1487.7 (Regular Cauchy tail estimate tail-induction scope)**.** *For every accepted* RegularCauchyTailEstima *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, induction over displayed finite tail windows preserves the route*

$$M, W, D, R, E, H, C, P, N.$$

The threshold row M and dyadic tolerance row D supply the modulus and tolerance scope before any $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R or* $\mathsf{Real}^{\uparrow}$ *seal through E is visible. The induction is confined to displayed finite windows in W and exports no quotient streams, selected limits, choice, trichotomy, open-cover compactness, ambient completeness, or independent Real seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier tail induction scope`

Proof. Start with the carrier definition, Definition 1487.1. The only rows available to a finite tail-window induction are the threshold, window, dyadic tolerance, readback, seal, transport, replay, provenance, and local naming rows $M, W, D, R, E, H, C, P, N$.

Apply the Real-seal route, tail-envelope lock, and tail dominance results. Theorems Definition 1487.3, Definition 1487.5, and Definition 1487.6 force every step to read M, W, D before R and to keep the terminal seal E downstream of the same finite tail-envelope route.

For the induction step, extend only the displayed finite window row W while keeping the same M, D, R, E, H, C, P, N surface. Since the carrier has no quotient stream, selected limit, choice, trichotomy, open-cover compactness, ambient-completeness, or second-seal coordinate, the induction preserves exactly the displayed route. □

Theorem 1487.8 (Regular Cauchy tail estimate dyadic-modulus scope)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, any Real-facing or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing reader of the tail estimate must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *row N.*

The dyadic modulus exposed by M, D is therefore scoped to the displayed finite tail-envelope route, and it exports no selected limit, quotient stream, countable-choice schedule, host equality, open-cover compactness, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier dyadic modulus scope`

Proof. Project the carrier of Definition 1487.1. The rows $M, W, D, R, E, H, C, P, N$ are the only coordinates available to a reader of the accepted tail estimate.

Apply Definition 1487.2. The local NameCert surface binds carrier admission, classifier transport, componentwise stability, seal-route exactness, and refusal rows to that same finite packet.

Use Definition 1487.3 to place the Real-facing read downstream of the modulus threshold, stream window, dyadic tolerance, and regular readback rows. A $\mathsf{RegSeqRat}^{\uparrow}$-facing reader reaches the same R only after M, W, D have been displayed.

Finally apply Definition 1487.5. The lock identifies the route through $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ as the required envelope before the seal is exposed, so selected limits, ambient completeness, and the other excluded coordinates are not exported. □

Theorem 1487.9 (Regular Cauchy tail difference)**.** *Let $T_0 = (M_0, W_0, D_0, R_0, E_0, H_0, C_0, P_0, N_0)$ and $T_1 = (M_1, W_1, D_1, R_1, E_1, H_1, C_1, P_1, N_1)$ be accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carriers. Suppose the threshold and dyadic tolerance rows are transported to a common finite surface, $M_0 \mathsf{hsame} M_1$ and $D_0 \mathsf{hsame} D_1$, and suppose both packets display their* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal routes through those transported rows. Then the finite comparison of the two tails is read through*

$$M_i, W_i, D_i, R_i, E_i, H_i, C_i, P_i, N_i$$

for $i = 0, 1$, together with the shared transported M, D surface.

The comparison exports no selected limit, quotient stream equality, host real equality, countable-choice schedule, open-cover compactness coordinate, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier tail difference`

Proof. Project the two accepted carriers. The only rows available to the comparison are the two displayed threshold, window, tolerance, readback, seal, transport, replay, provenance, and naming packets.

Transport the threshold and tolerance rows through the given hsame data. After this transport, both packets read their tail windows over the same finite M, D surface.

Replay the continuation rows C_0 and C_1. For each packet, the read passes through the displayed $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any Real-facing comparison is public.

Finally use Definition 1487.3 and Definition 1487.8. These results force both seal reads to remain behind the transported dyadic-modulus route, so the tail difference is a finite two-packet comparison rather than a selected limit or quotient-stream equality. □

Theorem 1487.10 (Regular Cauchy tail estimate window triangle)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier $T = (M, W, D, R, E, H, C, P, N)$, any two tail-window reads that share the displayed* $\mathsf{StreamName}^{\uparrow}$ *window W and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D compose through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row E. The composed read may use the threshold row M, the tail-envelope lock, tail dominance, dyadic-modulus scope, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *row N.*

It exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, quotient stream equality, selected limit, host real equality, countable-choice schedule, open-cover compactness coordinate, or second terminal seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier window triangle`

Proof. Start with two accepted tail-window reads sharing W and D. By Definition 1487.8, each read must expose the same finite window row and dyadic tolerance ledger before any $\mathsf{RegSeqRat}^{\uparrow}$-facing or $\mathsf{Real}^{\uparrow}$-facing coordinate is inspected.

Apply the tail-envelope lock of Definition 1487.5. The lock places both reads inside the route through $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$, so their common W, D surface has only the displayed readback row R available downstream.

Use tail dominance from Definition 1487.6. Any later Real-facing comparison remains dominated by the same finite tail surface, and the structural rows H, C, P, N only transport, replay, package, and name that surface.

Finally use the Real-seal route of Definition 1487.3. Since every read of E factors through M, W, D, R in order, the two tail-window reads compose through the same readback row R and terminal seal E.

The refused principles have no coordinate in the carrier. Ambient Real completeness, quotient stream equality, selected limits, host equality, choice, open-cover compactness, and a second terminal seal therefore cannot replace the finite window triangle. □

Theorem 1487.11 (Regular Cauchy tail estimate window monotone nesting)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$ *with two displayed finite tail windows where the second refines the first, every consumer must route the coarser read through the same* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, tail-envelope lock, and terminal* $\mathsf{Real}^{\uparrow}$ *seal route* E *before reading the refined estimate. The refinement adds only visible tolerance rows and displayed finite-window rows inside the same packet.*

It cannot create a quotient stream, selected limit, host convergence predicate, second terminal seal, open-cover compactness coordinate, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateWindowMonotoneNesting`

Proof. Project the carrier of Definition 1487.1. The displayed rows $M, W, D, R, E, H, C, P, N$ are the only rows available to both finite windows, and the refinement hypothesis only adds rows admitted by the same W and D.

Apply the tail-envelope lock of Definition 1487.5. The coarser read must expose the route through $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the seal row E is inspected.

Use tail dominance and dyadic-modulus scope, Definition 1487.6 and Definition 1487.8. These rows force both the coarser and refined reads to keep the same displayed window, tolerance ledger, readback, transport, replay, provenance, and local naming surface.

Finally apply the window triangle theorem, Definition 1487.10. Since the two reads share the same finite window route and dyadic tolerance ledger, the refined read composes after the coarser read through the same R and E coordinates. The extra rows are visible finite refinements, not new quotient, choice, convergence, compactness, or completeness coordinates. □

Theorem 1487.12 (Regular Cauchy tail estimate obligation scope)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, the obligation-closure surface is exactly the displayed rows* $M, W, D, R, E, H, C, P, N$ *together with the routes already cited through* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

The surface refuses selected limits, quotient streams, countable choice, host equality, open-cover compactness, and ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier obligation scope`

Proof. Project the carrier of Definition 1487.1. Its rows are precisely the modulus threshold, finite window, dyadic tolerance, regular readback, real seal, componentwise transport, displayed replay, provenance, and local naming rows $M, W, D, R, E, H, C, P, N$.

Apply Definition 1487.2. The local certificate binds carrier admission, classifier transport, componentwise stability, seal-route exactness, and refusal rows to the same finite packet.

Use Definition 1487.3, Definition 1487.5, and Definition 1487.8. Together these route every Real-facing or $\mathsf{RegSeqRat}^{\uparrow}$-facing read through $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the seal is consumed.

No additional coordinate remains for selected limits, quotient streams, countable choice, host equality, open-cover compactness, or ambient completeness. Hence the obligation scope is exactly the displayed kernel surface and the named sibling routes. □

Theorem 1487.13 (Regular Cauchy tail estimate obligation register)**.** *For every accepted* $\mathsf{RegularCauchyTailEstima}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, the carrier of Definition 1487.1, the NameCert obligations of Definition 1487.2, the Real seal route of Definition 1487.3, the tail-envelope lock of Definition 1487.5, the tail dominance row of Definition 1487.6, and the dyadic-modulus scope of Definition 1487.8 form the obligationClosure register for* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$*.*

The register exports only the displayed rows $M, W, D, R, E, H, C, P, N$ *and the named routes through* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier obligation register`

Proof. Carrier admission is the finite packet of Definition 1487.1. The local certificate rows are exactly the obligation surface listed in Definition 1487.2.

The Real-facing dependency route is fixed by Definition 1487.3. Its tail-envelope lock and later comparison dominance are supplied by Definition 1487.5 and Definition 1487.6.

Finally, Definition 1487.8 binds the dyadic tolerance and modulus rows to the same finite packet. Together these cited rows account for the carrier, route, stability, dominance, and refusal obligations, so the displayed surface is the obligationClosure register. □

Theorem 1487.14 (Regular Cauchy tail estimate finite precision induction)**.** *Let* $T = (M, W, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier. Suppose* T *exposes a tail estimate at a displayed unary precision window, and suppose the next precision window is admitted by the same* $\mathsf{StreamName}^{\uparrow}$ *row* W*. Then the estimate refines through the same* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row* M*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *route, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*.*

The refinement is obtained by induction on the displayed finite precision-window extension. It does not use host $\mathbb{N}$ *recursion, countable choice, a selected limit, quotient stream equality, host real equality, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier finite precision induction`

Proof. Begin with the displayed precision window already admitted by W. By Definition 1487.8, the corresponding tail estimate must read the dyadic tolerance row D, the stream window W, the regular readback row R, and the terminal seal row E through the modulus data M, D.

For the induction step, append one admitted finite precision window to the displayed StreamName row. The hypothesis says that the appended window is still a row of the same W, so the carrier is not replaced and no new consumer surface is introduced.

Apply Definition 1487.12. The obligation surface remains exactly $M, W, D, R, E, H, C, P, N$ together with the named sibling routes through $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Thus each finite extension step refines the estimate through the same modulus, dyadic, readback, tail-envelope, and Real rows. Since the induction ranges only over the displayed finite extension, it never invokes host natural-number recursion, choice, selected limits, quotient streams, host equality, open-cover compactness, or ambient completeness. □

Theorem 1487.15 (Regular Cauchy tail estimate obligation basis)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, the carrier row of Definition 1487.1, the local NameCert obligations of Definition 1487.2, the tail-envelope lock of Definition 1487.5, the tail dominance theorem of Definition 1487.6, the dyadic-modulus scope of Definition 1487.8, the window triangle of Definition 1487.10, the obligation scope of Definition 1487.12, and the finite precision induction of Definition 1487.14 exhaust the finite NameCert obligation basis for* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$*.*

Consequently every L10 Real and Completion consumer reads the packet only through the displayed rows $M, W, D, R, E, H, C, P, N$ *and the named sibling routes through* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. No additional obligation row remains for quotient streams, selected limits, countable choice, host equality, open-cover compactness, or ambient completeness.*

Proof. Start with carrier admission. The carrier definition exposes exactly the threshold, window, dyadic tolerance, regular readback, real seal, transport, replay, provenance, and local naming rows, and the NameCert obligations assign carrier admission, classifier transport, componentwise stability, seal-route exactness, and refusal to those rows.

Next use the tail-envelope lock and tail dominance theorems. They force each real-facing comparison through the finite route from M, W, D, R to $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ before the seal row E is consumed.

The dyadic-modulus scope and window triangle then account for all tolerance and window reads. They show that shared finite windows, dyadic ledgers, and regular readback rows compose through the same terminal seal rather than through a second completion or quotient coordinate.

Finally apply the obligation scope and finite precision induction. The obligation scope lists the full sibling route surface, while the induction shows that finite precision extensions stay inside the same packet. These rows cover every displayed obligation named by the carrier, so the packet has no further NameCert basis outside the cited surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateObligationBasis`

Theorem 1487.16 (Regular Cauchy tail estimate RegSeqRat handoff)**.** *For every accepted* $\mathsf{RegularCauchyTailEstim}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, any completion-facing consumer must first expose the handoff route*

$$D, W, R, E, H, C, P, N.$$

Thus the consumer reads the dyadic tolerance row, the $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg

provenance, and the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEstimate}^{\uparrow}}$ *row before any completion-facing tail estimate is public.*

The route exports no quotient stream, selected limit, countable-choice schedule, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateRegSeqRatHandoff`

Proof. Begin with the Real seal route of Definition 1487.3. It places every seal-facing read behind the threshold row M, finite window W, dyadic tolerance row D, and regular readback row R before E is consumed.

Use the dyadic-modulus scope and window triangle, Definition 1487.8 and Definition 1487.10. These results keep the dyadic tolerance ledger and shared finite windows attached to the same $\mathsf{RegSeqRat}^{\uparrow}$ readback row, so the consumer cannot detach the tail estimate from D, W, R.

Next apply monotone nesting, Definition 1487.11. Refining a finite tail window stays inside the same packet and adds only displayed tolerance and window rows before the readback and seal are replayed.

Finally invoke the obligation basis of Definition 1487.15. It exhausts the available NameCert rows by $M, W, D, R, E, H, C, P, N$ and the named sibling routes, so completion-facing consumers receive only the displayed RegSeqRat-to-Real handoff and none of the excluded completion principles. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCarrier regseqrat handoff`

Theorem 1487.17 (Regular Cauchy tail estimate modulus-dominance criterion)**.** *For every accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier* $T = (M, W, D, R, E, H, C, P, N)$*, the finite precision induction of Definition 1487.14 and the window monotone nesting of Definition 1487.11 produce a modulus-dominance witness for the displayed tail route. Consequently every later* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison consuming* R *or* E *is bounded by the displayed* M, W, D *tail route.*

No selected limit, quotient stream, trichotomy branch, choice schedule, open-cover compactness row, or ambient completeness principle may replace that dominance witness.

Proof. Start with the Real seal route of Definition 1487.3. It places every consumer of E behind the threshold row M, finite window row W, dyadic tolerance row D, and regular readback row R.

Apply the tail-envelope lock and dyadic-modulus scope, Definition 1487.5 and Definition 1487.8. These rows identify the route through M, W, D, R as the finite tail surface that every $\mathsf{RegSeqRat}^{\uparrow}$-facing or $\mathsf{Real}^{\uparrow}$-facing read must expose.

Use window monotone nesting together with finite precision induction. Refining a displayed window remains inside the same W row and extends only the finite precision data already governed by M and D, so each later comparison carries the same modulus-dominance witness.

Finally invoke the RegSeqRat handoff of Definition 1487.16. The carrier inventory contains no coordinate for selected limits, quotient streams, trichotomy, choice, open covers, or ambient completeness, so the ledger refusal leaves only the displayed M, W, D route available to consumers of R or E. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateModulusDominanceCriterion`

Theorem 1487.18 (Regular Cauchy tail estimate public interface)**.** *Every consumer of a Real-facing tail estimate through an accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier*

$$T = (M, W, D, R, E, H, C, P, N)$$

must read the estimate through the dyadic tolerance surface D, finite $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the modulus-dominance row of Definition 1487.17. This public route is the only exported interface for completion-facing consumers.*

The interface exports no quotient stream, selected limit, countable-choice schedule, trichotomy branch, host real equality, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimatePublicInterface`

Proof. Begin with the Real seal route of Definition 1487.3. It places every consumer of the terminal seal E behind the threshold row M, finite window row W, dyadic tolerance row D, and regular readback row R.

Apply tail dominance and the window triangle, Definition 1487.6 and Definition 1487.10. These theorems keep the displayed dyadic tolerance ledger and compared windows attached to the same regular readback before any Real-facing read is allowed.

Use window monotone nesting and the RegSeqRat handoff, Definition 1487.11 and Definition 1487.16. Refining a tail window remains inside the same carrier, and the handoff exposes R only through the displayed M, W, D rows before the seal E is read.

Finally invoke the modulus-dominance criterion of Definition 1487.17. It supplies the reusable dominance witness for later $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ comparisons, while the carrier inventory excludes quotient streams, selected limits, choice schedules, trichotomy, host equality, open covers, and ambient completeness. □

Theorem 1487.19 (Regular Cauchy tail estimate public tail export)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *carrier*

$$T = (M, W, D, R, E, H, C, P, N)$$

must read the public tail route

$$M, W, D, R, E, H, C, P, N$$

through $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any completion seal is visible. The route exports the finite tail estimate and its modulus-dominance witness; it exports no quotient stream, selected limit, countable-choice schedule, host real equality, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the public interface of Definition 1487.18. It requires every Real-facing tail estimate to pass through the dyadic tolerance surface, finite StreamName windows, RegSeqRat readback, terminal Real seal, and the displayed modulus-dominance row.

Next use the tail-envelope lock and Real seal route, Definition 1487.5 and Definition 1487.3. These theorems keep the threshold row M, window row W, tolerance row D, readback row R, and seal row E in the same finite packet before any completion-facing read is exposed.

Finally apply the modulus-dominance criterion and RegSeqRat handoff, Definition 1487.17 and Definition 1487.16. They identify the reusable dominance witness and force the structural rows H, C, P, N to travel with the same route, while the carrier inventory excludes quotient streams, selected limits, choice schedules, host equality, open covers, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimatePublicTailExport`

Theorem 1487.20 (Regular Cauchy tail estimate completion consumer route)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion consumer that cites the public* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *interface must consume the public tail export of Definition 1487.19. In particular the consumer sees $M, W, D, R, E, H, C, P, N$*

as one finite route through RealCauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *not as a detached completion seal or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimateCompletionConsumerRoute`

Proof. Apply the public tail export. It already lists the finite rows and sibling route that every completion-facing read must expose before the seal row is visible.

The public interface of Definition 1487.18 supplies the same route as the exported interface for Real-facing consumers. Therefore a later RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ chapter can cite this theorem as its completion-consumer entrance without adding quotient streams, selected limits, choice, host equality, open covers, or ambient completeness. □

Theorem 1487.21 (Regular Cauchy tail estimate diagonal-tail fusion)**.** *Any diagonal-limit consumer that cites* RegularCauchyTailEstimate$^{\uparrow}$ *must fuse the displayed route*

$$M, W, D, R, E, H, C, P, N$$

with its finite diagonal-budget route before any Real$^{\uparrow}$ *seal is consumed. The fused route reads the public tail export, the completion consumer route, the* RegSeqRat$^{\uparrow}$ *handoff, and the modulus-dominance criterion as one finite tail-to-diagonal surface. It exports no quotient streams, selected limits, countable choice, host real equality, open-cover compactness, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the public tail export of Definition 1487.19. It exposes the finite route $M, W, D, R, E, H, C, P, N$ through RealCauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ before any completion seal is visible.

Apply the completion consumer route, Definition 1487.20. A later diagonal-limit chapter may enter the tail-estimate interface only by consuming that public route, not by reading a detached Real$^{\uparrow}$ seal or an ambient completion theorem.

Next use the RegSeqRat handoff of Definition 1487.16. It keeps the dyadic tolerance row, finite StreamName windows, regular readback, terminal seal, transport, replay, provenance, and local naming rows in the same completion-facing packet.

Finally apply the modulus-dominance criterion of Definition 1487.17. The criterion bounds later RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ comparisons by the displayed M, W, D tail route, so a diagonal budget must fuse with this finite surface before the seal E is consumed. Since none of the refused objects is a carrier coordinate, the fused route cannot export quotient streams, selected limits, countable choice, host equality, open-cover compactness, or ambient completeness. □

Closure status for NameCert$_{\text{RegularCauchyTailEstimate}^{\uparrow}}$.

Constructive story: A RegularCauchyTailEstimate$^{\uparrow}$ packet is generated from a RealCauchyModulus$^{\uparrow}$ threshold row, a finite StreamName$^{\uparrow}$ window, a DyadicRatCore$^{\uparrow}$ tolerance row, a RegSeqRat$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailEstimate}^{\uparrow}}$, maturePackageClosed)

Scope closed: The public interface is now bridged by the checked `RegularCauchyTailEstimatePublicInterface` target over the displayed modulus, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailEstimate}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyTailEstimateUp.RegularCauchyTailEstimatePublicInterface`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, host real equality, choice, trichotomy, selected limits, open-cover compactness, axiom-purity certification, or bridge checking.

Upgrade path: No further paper-side theory closure remains; bridge checking is the remaining verification-axis work.

Chapter 1488

A Concrete Naming Certificate for $\mathsf{LocatedCompact}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{LocatedCompact}^{\uparrow}$ packet records located compactness as finite net and located-distance data before any Real or Completion consumer may read a compact boundary. It sits between the finite-net discipline of chapter 127, the separated completion seal of chapter 402, the apartness-real boundary of chapter 304, the terminal real-name seal of chapter 38, and the finite-window source of chapter 126. The packet names a constructive L10 boundary object: locatedness and finite nets are displayed rows, while open covers, quotient compacta, host-selected points, and ambient completeness remain outside the carrier.

Definition 1488.1 (LocatedCompact carrier)**.** *A* $\mathsf{LocatedCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, L, F, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact carrier row, naming the inspected metric or real-facing source as a finite BEDC carrier rather than a host compact space;*
2. *L is the locatedness test row, returning for each displayed point-and-radius request the finite comparison data that witnesses whether the request lies inside the located boundary or is apart from it by a recorded margin;*
3. *F is the finite-net witness row, exposing for each positive tolerance a closed* $\mathsf{ProbeBundle}$ *of centers,* $\mathsf{InBundle}$ *membership rows, and coverage distances that factor through the* $\mathsf{TotallyBounded}^{\uparrow}$ *surface of chapter 127;*
4. *A is the apartness and separatedness row, tying the locatedness test to the* $\mathsf{ApartnessReal}^{\uparrow}$ *boundary of chapter 304 and to the separated completion discipline of chapter 402;*
5. *H records componentwise* hsame *transports for carrier, locatedness, finite-net, apartness, separatedness, continuation, provenance, and local naming rows;*
6. *C records displayed* Cont *routes from finite-window requests through chapter 126, the net row F, the locatedness row L, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38;*
7. *P is the* Pkg *provenance over* $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eight displayed rows. It has no coordinate for an open cover, a selected finite subcover, a quotient compactum, host choice, a host completion object, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactTasteGate single carrier alignment`

Theorem 1488.2 (LocatedCompact NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompact}^{\uparrow}$ *carrier* $K = (X, L, F, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed compact carrier row* X *together with its locatedness, finite-net, apartness, transport, continuation, provenance, and naming support;*
2. *locatedness stability, stating that the test row* L *is preserved by the stored* hsame *transports and by the displayed comparison rows reaching the* $\mathsf{Real}^{\uparrow}$ *seal;*
3. *finite-net exposure, stating that each tolerance request reaches only the closed* ProbeBundle, InBundle*, center, radius, and coverage-distance rows displayed in* F*;*
4. *apartness and separatedness compatibility, stating that* A *is the only route from located-distance evidence to the separated completion and apartness-real sibling surfaces;*
5. *ledger exactness for the* Cont *replay,* Pkg *provenance, and local* NameCert *row, with no hidden open-cover, quotient, choice, or ambient-completeness coordinate.*

These obligations name located compactness as a finite BEDC packet over the displayed L10 sibling route, not as a host compact-space principle.

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactNameCertObligations`
↪ variant `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier namecert obligations`

Proof. Project the accepted packet $K = (X, L, F, A, H, C, P, N)$. The projection exposes exactly the compact carrier row, locatedness test, finite-net witness, apartness and separatedness row, componentwise transports, continuation replay, provenance, and local naming row.

The locatedness obligation is componentwise. A transported point-and-radius request is read through H, and the same displayed comparison route in L supplies the inside-or-apart evidence before any $\mathsf{Real}^{\uparrow}$ seal is consumed.

The finite-net obligation is the $\mathsf{TotallyBounded}^{\uparrow}$ handoff. A tolerance request reaches the closed ProbeBundle in F, then reads only its bundled centers, membership constructors, radii, and coverage-distance rows. No external list of centers or selected subcover is present.

The apartness and separatedness obligation uses A as the unique boundary row. It ties the located-distance evidence to the apartness-real and Hausdorff-completion sibling surfaces while preserving the same H, C, P, N support.

Finally, exactness follows from the carrier definition: C replays only the displayed route, P records only the named sibling certificates, and N names the local packet. Since the carrier has no additional coordinate, the five obligations exhaust the local $\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}$ surface. □

Theorem 1488.3 (LocatedCompact totally bounded handoff)**.** *Every accepted* $\mathsf{LocatedCompact}^{\uparrow}$ *carrier exports to compact and complete consumers only the displayed finite-net row* F *and locatedness row* L*, together with the structural support* A, H, C, P, N*. Thus a downstream consumer may read total boundedness through the finite* ProbeBundle *witnesses and may read locatedness through the finite comparison tests, but it receives no open-cover compactness, Heine–Borel theorem, quotient compactum, host choice, or ambient-completeness field.*

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier totally bounded handoff`

Proof. Begin with an accepted carrier $K = (X, L, F, A, H, C, P, N)$. The compact-facing data exposed by the carrier are L and F, and the only structural support rows are A, H, C, P, N.

Read the finite-net handoff through F. The handoff is the same finite-net surface consumed by chapter 127: a positive tolerance request, a closed ProbeBundle of centers, membership constructors, and coverage rows.

Read the locatedness handoff through L. The comparison evidence is attached to the same carrier row X and is transported only by H before the continuation route C reaches the real-facing seal.

The separated and apartness consumers can use A only after those displayed handoffs are present. Since no open-cover, finite-subcover selector, quotient compactum, host-choice row, or ambient-completeness field occurs in the packet, compact and complete consumers receive exactly the finite-net and locatedness rows displayed above. □

Theorem 1488.4 (Located compact finite-net obligation package)**.** *For every accepted* LocatedCompact$^{\uparrow}$ *carrier* $K = (X, L, F, A, H, C, P, N)$*, the finite-net obligation package exposes only finite* ProbeBundle *centers,* InBundle *membership constructors, located-distance comparison rows, and the structural* hsame/Cont *replay stored in* H *and* C*. Thus total-bounded consumers may read the closed net witnesses in* F *and the locatedness comparisons in* L*, but they receive no open-cover compactness, selected finite subcover, host choice, quotient compactum, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactFiniteNet rows from obligation surface`

Proof. Project the carrier using Definition 1488.1. The net row F is the only row containing centers, membership, radii, and coverage-distance data, and it is explicitly finite through the displayed ProbeBundle and InBundle constructors.

Use Definition 1488.3 to identify the consumer route. A tolerance request reaches F, reads its finite center and membership rows, and then uses the locatedness row L for the corresponding distance comparisons.

The structural rows do not add compactness principles. Componentwise hsame transport in H preserves the displayed row types, and Cont replay in C replays the same finite-window route. Since no open-cover, finite-subcover selector, quotient compactum, host-choice row, or ambient-completeness field is present, the finite-net obligation package is exhausted by F, L, H, C together with A, P, N. □

Theorem 1488.5 (Located compact Real-window handoff)**.** *Every* Real$^{\uparrow}$*-facing consumer of an accepted* LocatedCompact$^{\uparrow}$ *carrier reads through the* StreamName$^{\uparrow}$ *finite windows, the finite-net row* F*, the locatedness row* L*, and the terminal* Real$^{\uparrow}$ *seal recorded by the displayed continuation route* C*. The handoff exports a located compact window certificate over finite data, not an open-cover compactness principle or a host-selected point.*

Proof. Begin with the continuation row of Definition 1488.1. It routes finite-window requests through chapter 126, then through the finite-net row, the locatedness row, and finally the terminal real seal.

Apply Definition 1488.3. The compact-facing information available to a consumer is the net row F, the locatedness row L, and the structural support A, H, C, P, N.

Now use Definition 1488.4. The finite-net row supplies only closed ProbeBundle centers, membership rows, and coverage-distance data, while the locatedness row supplies only finite comparisons. The real-facing consumer therefore receives a finite Real-window handoff and no open-cover

compactness, host choice, quotient compactum, selected finite subcover, or ambient completeness theorem. □

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier real window handoff`

Theorem 1488.6 (LocatedCompact public finite-net boundary)**.** *The public* LocatedCompact$^{\uparrow}$ *interface exports exactly the finite* ProbeBundle *net data, locatedness comparison rows, apartness and separatedness support,* StreamName$^{\uparrow}$ *windows,* Real$^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *surface carried by Definition 1488.1. It exports no open-cover compactness theorem, selected finite subcover, quotient compactum, host choice, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier public finite net boundary`

Proof. Project the carrier $K = (X, L, F, A, H, C, P, N)$. The row F is the finite-net surface, L is the locatedness comparison surface, A is the apartness and separatedness support, and C routes the stream windows to the terminal Real seal.

Use Definition 1488.2, Definition 1488.4, and Definition 1488.5. These identify the finite ProbeBundle centers, InBundle memberships, locatedness comparisons, stream-window route, and Real handoff as checked local rows.

The remaining rows H, P, N transport, source, and name the same packet, while Cont replay is already stored in C. Since the carrier has no row for open covers, selected subcovers, quotient compacta, host choice, or ambient completeness, the public boundary is exactly the displayed finite interface. □

Theorem 1488.7 (LocatedCompact public export surface)**.** *The public* LocatedCompact$^{\uparrow}$ *export is exactly the displayed finite-net, locatedness, apartness and separatedness, Real-window,* hsame, Cont, Pkg*, and local* NameCert *surface of Definition 1488.1, Definition 1488.2, Definition 1488.3, Definition 1488.4, and Definition 1488.5. A consumer may read the finite* ProbeBundle *centers and* InBundle *membership rows in F, the locatedness comparisons in L, the apartness and separatedness row A, the terminal* Real$^{\uparrow}$ *window route in C, and the structural rows H, P, N. It may not read an open-cover compactness theorem, selected finite subcover, quotient compactum, host choice, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier public export surface`

Proof. Project the accepted carrier $K = (X, L, F, A, H, C, P, N)$. The carrier definition lists exactly the finite-net row F, locatedness row L, apartness and separatedness row A, componentwise transport row H, continuation row C, provenance row P, and local naming row N.

Use the obligation and handoff theorems cited above. They identify the finite-net, locatedness, Real-window, and structural support read by compact and Real-facing consumers. Since no additional row occurs in the carrier, the public export surface is exhausted by those displayed rows and none of the excluded compactness or completeness principles is exported. □

Theorem 1488.8 (LocatedCompact mature finite-net consumer boundary)**.** *The mature public consumer boundary of an accepted* LocatedCompact$^{\uparrow}$ *carrier is exactly the finite* ProbeBundle *net row, locatedness comparison row, apartness and separatedness support,* StreamName$^{\uparrow}$ *windows, terminal* Real$^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row displayed by Definition 1488.1. A mature consumer may compose these rows with the TotallyBounded, HausdorffCompletion, ApartnessReal, StreamName, and Real sibling routes, but it*

receives no open-cover compactness theorem, selected finite subcover, quotient compactum, host choice principle, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.LocatedCompactUp.LocatedCompactCarrier mature consumer boundary`

Proof. Combine the public finite-net boundary of Definition 1488.6 with the public export surface of Definition 1488.7. These two surfaces expose the same rows F, L, A, H, C, P, N together with the finite stream windows and terminal real seal route.

The mature consumer has no additional carrier coordinate beyond those rows. Thus any compactness-facing or Real-facing read must pass through the displayed net, locatedness, apartness, separatedness, stream-window, seal, transport, replay, provenance, and local naming data before reaching a sibling route.

Since the carrier has no row for open covers, selected subcovers, quotient compacta, host choice, or ambient completeness, none of those principles can be exported by the mature boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompact}^{\uparrow}$ packet is generated from a compact carrier row, a locatedness test row, a finite ProbeBundle net witness, an apartness and separatedness row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1488.1, Definition 1488.2, Definition 1488.3, Definition 1488.4, Definition 1488.5, Definition 1488.6, Definition 1488.7, and Definition 1488.8 give the mature finite-net consumer boundary through the TotallyBounded, HausdorffCompletion, ApartnessReal, StreamName, and Real sibling routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel as a general theorem, quotient compacta, host choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eight-row carrier and prove the finite NameCert obligation surface and total-bounded handoff theorem.

Chapter 1489

A Concrete Naming Certificate for $\mathsf{IntervalHalving}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{IntervalHalving}^{\uparrow}$ packet records the finite dyadic bisection step used when a displayed interval is routed toward a regular Cauchy or real-completion consumer. It sits after the dyadic endpoint ledger of chapter 291 and before the finite nested-window handoff of chapter 320, with downstream reads through chapter 126, chapter 301, and chapter 38. The packet names one finite halving decision; it does not supply quotient streams, host real equality, selected limits, countable choice, or ambient completeness.

Definition 1489.1 (IntervalHalving carrier)**.** *An* $\mathsf{IntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (L, R, M, B, \rho, S, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row of the parent interval, together with its endpoint-order ledger;*
2. *R is the right* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row of the parent interval, together with the displayed witness that L precedes R;*
3. *M is the midpoint row, computed from the dyadic endpoint ledger and recorded before any predicate-facing half is selected;*
4. *B is the chosen half row, whose tag is either left or right and whose endpoints are respectively (L, M) or (M, R);*
5. *ρ is the radius-halving ledger, displaying that the child width is bounded by half of the parent width at the same dyadic scale;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the endpoint and midpoint observations used by the halving step;*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reading the same dyadic endpoints, midpoint, chosen half, and radius ledger as finite regular-rational data;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row available to completion consumers only through the finite S, Q, ρ route;*
9. *T records componentwise* hsame *transport for the endpoint rows, midpoint row, chosen half row, radius ledger, finite-window row, regular-rational handoff, and real-facing boundary;*

10. *C records visible* Cont *routes by which consumers replay the same halving packet without adding an external selector;*
11. *P is the* Pkg *provenance over* $\mathsf{IntervalHalving}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalHalving}^{\uparrow}}$ *row.*

The carrier exports only this finite dyadic endpoint-to-midpoint-to-half route. It has no coordinate for a quotient stream, host real equality, selected limit, countable choice sequence, or ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.IntervalHalvingUp.IntervalHalvingCarrier`

Definition 1489.2 (IntervalHalving classifier)**.** *Two accepted* $\mathsf{IntervalHalving}^{\uparrow}$ *carriers classify together when their left and right dyadic endpoints, endpoint-order ledgers, midpoint rows, chosen-half tags, chosen-half endpoint rows, radius-halving ledgers, stream windows, regular-rational handoffs, real-facing boundaries, package provenance, and local name rows are connected by the componentwise* hsame *transports and visible* Cont *replays recorded in their transport tables.*

The classifier is finite-window exact: changing a downstream quotient-stream representative, a host real equality proof, a selected limiting point, a countable choice schedule, or an ambient completeness witness is unreadable unless it changes one of the displayed packet rows above.

Lean4 ✓ `BEDC.Derived.IntervalHalvingUp.TasteGate.IntervalHalvingTasteGate single carrier alignment`

Theorem 1489.3 (IntervalHalving NameCert obligations)**.** *For an accepted* $\mathsf{IntervalHalving}^{\uparrow}$ *carrier* $H = (L, R, M, B, \rho, S, Q, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalHalving}^{\uparrow}}$ *surface consists exactly of carrier admission for those twelve rows, dyadic midpoint formation, chosen-half endpoint admission, radius-halving ledger exactness, finite-window readback through* $\mathsf{StreamName}^{\uparrow}$*, regular-rational handoff through* $\mathsf{RegSeqRat}^{\uparrow}$*, real-facing boundary exposure through* $\mathsf{Real}^{\uparrow}$*, componentwise classifier transport, visible consumer replay, package provenance, and local naming.*

Proof. Carrier admission is the tuple displayed in Definition 1489.1. The dyadic endpoint rows L, R and midpoint row M are read through the endpoint ledger supplied by chapter 291.

The chosen half row B and radius ledger ρ give the finite bisection content. The tag in B selects one displayed child cell, while ρ records the halved radius at the same dyadic scale. No infinite branch or limiting object is named by these rows.

The consumer rows S, Q, E route the same finite data through chapter 126, chapter 301, and chapter 38. The structural rows T, C, P, N transport, replay, source, and name that route.

Since the carrier contains no quotient stream coordinate, host real equality coordinate, selected limit coordinate, countable choice coordinate, or ambient completeness coordinate, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.IntervalHalvingUp.IntervalHalvingCarrier namecert obligations`

Theorem 1489.4 (IntervalHalving dyadic nested handoff)**.** *Every nested-interval consumer of an accepted* $\mathsf{IntervalHalving}^{\uparrow}$ *packet receives exactly one displayed child interval route*

$$L, R, M, B, \rho, T, C, P, N.$$

The route factors through the dyadic endpoint ledger of chapter 291 and the finite nested packet discipline of chapter 320; it supplies a next finite cell, not a completed branch, quotient stream, selected point, or completeness argument.

Lean4 ✓ `BEDC.Derived.IntervalHalvingUp.interval halving dyadic nested handoff carrier route`

Proof. Begin with the parent endpoint rows L, R and midpoint row M. These rows determine the two possible dyadic child cells before any downstream consumer reads the packet.

Read the chosen half row B and the radius ledger ρ. They expose one child cell and the displayed radius-halving bound required by a finite $\mathsf{NestedInterval}^{\uparrow}$ consumer. This is the only nested-cell output available from the packet.

The handoff is replayed through T, C, P, N. The hsame and Cont rows preserve the same endpoint, midpoint, chosen-half, and radius rows, while the package and local certificate rows keep the route inside the dyadic and nested-interval source certificates.

The stream, regular-rational, and real-facing rows may be read only after this finite child-cell route is fixed. They cannot supply an infinite branch, quotient stream equality, selected limit, countable choice sequence, or ambient completeness theorem because none is present in the handoff route. □

Lean4 ✓ `BEDC.Derived.IntervalHalvingUp.IntervalHalvingDyadicNestedHandoff`

Theorem 1489.5 (IntervalHalving real completion boundary)**.** *The* $\mathsf{Real}^{\uparrow}$*- and completion-facing boundary of an accepted* $\mathsf{IntervalHalving}^{\uparrow}$ *carrier is exactly the finite dyadic halving data read through*

$$S, Q, E, T, C, P, N.$$

Every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the endpoint, midpoint, chosen-half, and radius-halving rows as finite readback data through chapter 126, chapter 301, and chapter 38. The boundary does not identify quotient streams, assert host real equality, select a limit, invoke countable choice, or appeal to ambient completeness.*

Proof. Project the finite-window row S. It contains exactly the endpoint and midpoint observations already fixed by the dyadic halving rows.

Use the regular-rational row Q to read those observations as finite approximants with the displayed radius-halving ledger. This gives a regular Cauchy consumer a bounded step, not a tail selector or selected limit.

Finally read the real-facing row E through T, C, P, N. The transport and replay rows preserve the same finite data, and the package and local certificate rows identify the source route back to chapter 291, chapter 320, chapter 126, chapter 301, and chapter 38.

Any quotient stream, host real equality, selected limit, countable choice schedule, or ambient completeness principle would require an additional readable coordinate. The carrier has none, so the completion boundary is exactly the finite route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalHalving}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntervalHalving}^{\uparrow}$ packet is generated from displayed $\mathsf{DyadicRatCore}^{\uparrow}$ left and right endpoint rows, a dyadic midpoint row, a chosen-half row, a radius-halving ledger, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, a $\mathsf{Real}^{\uparrow}$ seal-facing boundary, componentwise hsame transport, visible Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntervalHalving}^{\uparrow}}$, scopedClosed)

Scope closed: The carrier admission, classifier transport, midpoint formation, radius-halving exactness, nested-window handoff, and Real-completion boundary are bound as the scoped finite route for the displayed dyadic halving certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntervalHalving}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, host real equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a named consumer surface that exports the checked dyadic nested handoff without adding quotient streams, selected limits, countable choice, or ambient completeness.

The $\mathsf{PolishSpace}^{\uparrow}$ hub reads its complete metric, separable metric, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows through chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38.

Chapter 1490

A Concrete Naming Certificate for $\mathsf{PolishSpace}^{\uparrow}$

Following Alexandrov and Urysohn, a Polish space is treated here as a complete separable metric object whose admissible data are displayed by finite naming routes rather than by ambient set-theoretic selection. The $\mathsf{PolishSpace}^{\uparrow}$ packet synthesizes the metric carrier of chapter 57, the completion handoff of chapter 128, the countable dense surface of chapter 1458, the finite stream-window interface of chapter 126, and the regular rational readback surface of chapter 301. Its role is to expose the L10 complete-separable metric synthesis node without claiming choice-selected dense points, quotient equality, open-cover compactness, or a host-level completion principle.

Definition 1490.1 (Polish-space carrier)**.** *A* $\mathsf{PolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, K, D, S, R, W, H, C, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted point-history carrier, metric readbacks, distance classifier, and local metric* $\mathsf{NameCert}$*.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row; it records the displayed Cauchy-window and limit-seal route over the same metric carrier.*
3. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *row; it stores the countable dense schedule as public finite name data, not as a choice function over arbitrary balls.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that enumerates the inspected dense names and the Cauchy-window requests made by downstream consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to observe finite rational windows on the scheduled dense names and Cauchy approximants.*
6. *W is the finite observation ledger joining requested tolerances, stream positions, regular rational windows, metric-radius checks, and completion handoff rows.*
7. *H records componentwise* hsame *transport for M, K, D, S, R, W and for the local certificate rows.*
8. *C records the displayed* Cont *routes from a metric-radius request to the dense schedule, from a Cauchy-window request to the completion handoff, and from either request to the regular rational readback.*

9. *G is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common metric, complete-metric, separable-metric, stream-name, and regular-readback surface. It has no coordinate for selected dense points, quotient equality of completions, arbitrary open covers, or ambient host sets.

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceTasteGate single carrier alignment`

Theorem 1490.2 (Polish-space NameCert obligations)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows:*

1. *carrier admission for the displayed metric, complete-metric, separable-metric, stream-name, regular-readback, observation-ledger, transport, continuation, provenance, and local-name rows;*
2. *classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *surface before either the complete-metric or separable-metric row is consumed;*
3. *stability of the completion handoff K and the countable dense schedule D under the componentwise* hsame *transports recorded in H;*
4. *exactness of the observation ledger W as the only source of metric radius checks, Cauchy windows, dense-name inspections, and regular rational readbacks;*
5. Cont *replay from radius or Cauchy-window requests through S, R, W to the corresponding metric, completion, or dense-schedule row;*
6. Pkg *provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these six rows.

Proof. Project the carrier of Definition 1490.1. The projection exposes the metric row, complete-metric handoff, separable-metric dense schedule, stream schedule, regular rational readback, observation ledger, transport rows, continuation rows, provenance row, and local name.

The classifier route begins at the metric surface. The rows in H transport M, K, D, S, R, W together, so the completion handoff and dense schedule are never compared over different metric carriers.

The observation exactness row is exhausted by W. A metric-radius request, a Cauchy-window request, and a dense-name inspection all read their finite stream positions and regular rational observations through this ledger.

Finally replay C and attach G, N. These rows route, source, and name the already displayed packet; they do not introduce selected dense points, quotient completion equality, arbitrary open covers, or ambient host sets. Hence the six listed rows exhaust the local Polish-space naming obligations. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceNamecertObligations`

Theorem 1490.3 (Polish-space root density obligations)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The root density obligation is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule row D read over the same* $\mathsf{Met}^{\uparrow}$ *carrier M, through* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the observation ledger W, the supporting* BHist *packet, componentwise* hsame *transport*

H, displayed Cont *replay C,* Pkg *provenance G, and local* NameCert *row N. It exports no selected dense point, external enumeration, or host topology.*

Proof. Project the carrier of Definition 1490.1. The displayed rows include $M, D, S, R, W, H, C, G, N$, with D stored as a separable-metric row beside the metric source M.

Apply Definition 1490.2. Carrier admission and classifier transport keep D over the transported $\mathsf{Met}^{\uparrow}$ surface, while ledger exactness forces every dense-name inspection through S, R, W.

The rows H, C, G, N then transport, replay, source, and name that same finite route. Since no further coordinate is present, the root density obligation is exactly the displayed BHist route and not a selected dense subset or host enumeration theorem. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootDensityObligations`

Theorem 1490.4 (Polish-space root completion obligations)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The root completion obligation is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row K over the same* $\mathsf{Met}^{\uparrow}$ *carrier M used by the dense schedule D, with Cauchy-window requests read through* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, observation ledger W,* BHist *support, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and local* NameCert *row N.*

Proof. Project P through Definition 1490.1. The completion row is K, and its metric source is the same row M that supports the separable schedule D.

Use Definition 1490.2. The classifier transport and stability clauses keep K and D over the same transported metric carrier before either row is consumed.

The observation ledger W supplies the Cauchy-window route through S and R, while H, C, G, N provide transport, replay, provenance, and local naming. No quotient completion, selected limit, separate host metric, or ambient completeness principle is present in the root obligation. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCarrier completion density handoff`

Theorem 1490.5 (Polish-space root stream readback obligations)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every root readback of dense names, metric-radius checks, and Cauchy windows factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S and* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before it can be consumed by the* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{SeparableMetric}^{\uparrow}$ *rows M, K, D. The factorization is supported exactly by W, H, C, G, N as observation ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootStreamReadbackObligations`

Proof. Start from an accepted packet and project its stream-facing rows from Definition 1490.1. The stream schedule S, regular rational readback R, and observation ledger W are displayed before any analytic consumer reads M, K, or D.

Apply Definition 1490.2. Ledger exactness makes W the only route for metric-radius checks, Cauchy windows, dense-name inspections, and regular rational readbacks.

The transport, replay, provenance, and naming rows H, C, G, N do not add a second readback channel. Therefore every root stream readback remains inside the finite BHist packet and cannot escape to an external oracle, selected limit, host enumeration, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1490.6 (Polish-space StreamName basis)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every basis-level dense-name read of P factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before it can be consumed as* $\mathsf{Met}^{\uparrow}$,

$\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{SeparableMetric}^{\uparrow}$ *evidence. The factorization is supported exactly by the* BHist *packet, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and the local* NameCert *row* N*.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamNameBasis`

Proof. Project the carrier of Definition 1490.1. The stream schedule S, regular rational readback R, and observation ledger W are displayed before any dense-name consumer can read the separable row D.

Use Definition 1490.2. Its observation-exactness clause makes W the only route from a basis request to finite stream positions and regular rational windows, while H keeps the metric, completion, separable, stream, and readback rows on the same accepted packet.

Replay the request through C and source it through G, N. Since the carrier has no selected dense sequence, quotient point equality, host complete separable space, or choice-selected oracle coordinate, the basis read is exactly the displayed StreamName and RegSeqRat route. □

Theorem 1490.7 (Polish-space complete-separable handoff)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. A complete-separable consumer may read* P *only as the finite handoff from the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *schedule* D*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, with* W, H, C, G, N *preserving observation, transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableHandoff`

Proof. First expose the carrier rows. The metric row M is the common source, while K and D are only the displayed completion and separability rows over that source.

The StreamName and RegSeqRat rows provide the finite inspection route for the dense schedule and Cauchy-window requests. The ledger W records those inspections, and H transports all five mathematical rows together.

Finally replay the handoff through C and attach G, N. The packet contains no host complete separable space, quotient completion equality, choice-selected dense sequence, or ambient $\mathsf{Real}^{\uparrow}$ completeness row, so the complete-separable handoff is bounded by $M, K, D, S, R, W, H, C, G, N$. □

Theorem 1490.8 (Polish-space completion nonescape)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every completion-facing consumer read factors through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, observation ledger* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, local* NameCert *row* N*, and any terminal* $\mathsf{Real}^{\uparrow}$ *seal already exposed by those rows. No read escapes to host complete separable spaces, quotient point equality, choice-selected dense sequences, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionNonescape`

Proof. Start from a completion-facing request and project the accepted packet. The only completion row is K, and its finite windows are recorded through S, R, W before a terminal real-name seal can be consumed.

Apply Definition 1490.2. Carrier admission and classifier transport keep the completion handoff over the metric source M, while observation exactness prevents a completion request from bypassing the stream and regular-rational readback rows.

The structural rows H, C, G, N transport, replay, source, and name that same finite route. Since no coordinate names a host completion, quotient point equality, selected dense sequence, or ambient real-completeness theorem, every completion-facing consumer remains inside the displayed packet. □

Theorem 1490.9 (Polish-space dense-schedule root obligation)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The root dense-schedule obligation is the finite route that reads the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D only after the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger W have been fixed by* BHist *support, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It exports no choice-selected dense point and no ambient enumeration theorem.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceDenseScheduleRootObligationCert`

Proof. Apply Definition 1490.2. Carrier admission exposes $M, D, S, R, W, H, C, G, N$ as displayed rows of the same packet.

The classifier-transport clause uses H to keep the dense schedule over the metric surface M. The exactness clause makes W the only route from a dense-name request to the stream schedule S and the regular rational readback row R.

Replay the request through C and source it through G, N. Since the carrier has no coordinate for a selected dense subset or an external enumeration, the root dense-schedule obligation is exactly the finite route just displayed. □

Theorem 1490.10 (Polish-space complete-metric root obligation)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The root complete-metric obligation reads the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K only over the already fixed* $\mathsf{Met}^{\uparrow}$ *surface M and through the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and observation-ledger rows S, R, W used by dense-schedule consumers. The supporting rows are precisely the displayed* BHist *packet,* hsame *transport H,* Cont *replay C,* Pkg *provenance G, and local* NameCert *row N.*

Proof. Use Definition 1490.2. Its carrier-admission and classifier-transport rows require K to remain over the metric carrier M before any completion-facing request is consumed.

The stability row transports K together with D, S, R, W. Thus a Cauchy-window request and a dense-name request share the same stream schedule, regular rational readback, and finite observation ledger.

The replay and provenance rows C, G, N add only routing, sourcing, and local naming. No quotient completion, selected limit, ambient completeness principle, or separate host metric is present in the carrier, so the complete-metric root obligation is bounded by these rows. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteMetricRootObligation`

Theorem 1490.11 (Polish-space StreamName-to-Real root obligation)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every StreamName-to-Real readback at the root factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and observation ledger W before it can be consumed as metric, complete-metric, or separable-metric evidence. The factorization is supported only by the* BHist *rows, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceStreamNameRealRootObligation`

Proof. Start with the admitted packet and its observation-ledger exactness from Definition 1490.2. Any readback request must enter through the stream schedule S and the regular rational row R.

The ledger W records which finite rational windows are inspected before the result is used by M, K, or D. Componentwise transport in H keeps these rows on the same accepted carrier, while C replays the displayed route.

The provenance and naming rows G, N identify the packet but do not add an ambient real-completeness claim, a selected limit, or an external readback oracle. Therefore the StreamName-to-Real root obligation is precisely the finite factorization through S, R, W, H, C, G, N. □

Theorem 1490.12 (Polish-space StreamName dense handoff)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every consumer that reads the countable dense surface must factor through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the finite observation ledger W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before using the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D or the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K. Consequently the packet exports a countable dense handoff by finite names and regular rational windows, not by a choice-selected dense subset or a quotient-completion representative.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamNameDenseHandoff`

Proof. Start with a consumer of the dense surface. Carrier admission exposes D only as a separable-metric row over the metric carrier M and the stream schedule S.

Read the requested dense-name index through S and the finite ledger W. The regular rational observation used by the consumer is then obtained through R, so the handoff has a visible finite window before the dense row is consumed.

If the same consumer also requests a completion read, the route passes through K after the same metric surface and observation ledger have been fixed. The transports in H keep the dense schedule, completion handoff, and metric classifier on one admitted surface.

The remaining rows C, G, N only replay, source, and name this finite route. Since the carrier has no row for a choice-selected dense subset or a quotient completion representative, the dense handoff can export only the displayed stream-name and regular-readback data. □

Theorem 1490.13 (Polish-space observation-ledger exhaustion)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read from P factors through the finite observation ledger W together with the componentwise* hsame *rows H, the displayed* Cont *replay rows C, the* Pkg *provenance row G, and the local* $\mathsf{NameCert}$ *row N. In particular, dense-schedule reads and completion reads cannot bypass W: the rows D and K are consumable only after the finite stream windows in S and regular rational readbacks in R have been displayed inside the observation ledger.*

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCarrier observation ledger exhaustion`

Proof. Project the accepted carrier as $P = (M, K, D, S, R, W, H, C, G, N)$. The projection exposes the metric row, complete-metric handoff, separable-metric dense schedule, stream-name schedule, regular rational readback, observation ledger, transport, replay, provenance, and local naming rows.

Apply Definition 1490.2. Its exactness row identifies W as the only source of metric-radius checks, Cauchy-window requests, dense-name inspections, and regular rational readbacks.

A dense-schedule consumer reaches D only through an index displayed by S and an observation recorded in W. A completion consumer reaches K only after its Cauchy-window request has been recorded in the same ledger. In both cases the inspected value is read back through R before the metric, completion, or separability row is consumed.

The remaining rows H, C, G, N transport, replay, source, and name this finite route. Since the carrier has no coordinate outside the displayed ledger route, no consumer of M, K, D, S, or R can bypass W. □

Theorem 1490.14 (Polish-space root obligation package)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Before any complete-separable metric synthesis is consumed, P exposes exactly the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K, the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D, the* $\mathsf{StreamName}^{\uparrow}$ *row S, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, together with the finite observation ledger W, componentwise* hsame *transports H, displayed* Cont *replay C,* Pkg *provenance G, and local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootObligationPackage`

Proof. Project the carrier from Definition 1490.1. Its displayed rows are precisely the metric, complete, separable, stream, regular-readback, observation-ledger, transport, replay, provenance, and local naming rows.

Use Definition 1490.2 to read those rows as the local NameCert obligation package. It binds metric admission, completion handoff, dense schedule, stream windows, regular rational readback, and ledger exactness in one finite surface.

Finally apply Definition 1490.13. The root obligation package has no route around W, H, C, G, N, so every complete-separable consumer first sees exactly the five displayed analytic rows and their finite support. □

Theorem 1490.15 (Polish-space completion observation triad)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The complete-metric handoff K and the separable dense schedule D are both consumable only after the finite* $\mathsf{StreamName}^{\uparrow}$ *observation ledger S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the support ledger W have been displayed over the same metric row M. Thus every complete-separable consumer reads the triad*

$$S, R, W$$

before either K or D, and the packet exports no selected dense points, quotient completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionObservationTriad`

Proof. Start from Definition 1490.14. It displays $M, K, D, S, R, W, H, C, G, N$ before the complete-separable synthesis can be consumed.

Apply Definition 1490.13. Dense-schedule reads and completion reads are both routed through W, and their finite windows and rational observations are supplied by S and R.

The componentwise transport H keeps K and D over the same metric row M, while C, G, N replay, source, and name the route. Since no carrier coordinate stores selected dense points, quotient completion equality, a countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported. □

Theorem 1490.16 (Polish-space complete-metric source lock)**.** *For an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet $P = (M, K, D, S, R, W, H, C, G, N)$, every read of the complete-metric handoff K is locked to the source metric row M after the finite observation rows S, R, W have been displayed. The handoff may use the dense schedule D only through the same observation route, and it cannot replace that route by quotient completion equality, selected dense representatives, countable choice, or completeness of an ambient* $\mathsf{Real}^{\uparrow}$ *object.*

Proof. Project P through Definition 1490.1. The row K is a $\mathsf{CompleteMetric}^{\uparrow}$ handoff over the metric carrier M, not an independent completion object.

Use Definition 1490.2. The classifier transport row fixes the common metric surface before K or D is read, and the exactness row identifies W as the only source of Cauchy-window requests and regular rational readbacks.

Now apply Definition 1490.15. The rows S and R display the finite observations on which the handoff depends. Therefore a complete-metric consumer receives K only as a source-locked finite route over M, with no additional quotient, selector, choice, or ambient-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteMetricSourceLock`

Theorem 1490.17 (Polish-space separable dense schedule handoff)**.** *For an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet* $P = (M, K, D, S, R, W, H, C, G, N)$*, every read of the separable dense schedule* D *is handed off only after the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the observation ledger* W *have been displayed. The dense schedule is therefore a finite NameCert route over* M *and* K*, not a choice-selected dense subset, countable-choice enumeration, quotient-completion representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceSeparableDenseScheduleHandoff`

Proof. Begin with a consumer of D. By Definition 1490.12, the consumer must read the displayed stream schedule S, ledger W, and regular readback R before using the separable-metric row.

The root obligation package fixes the surrounding rows M, K, H, C, G, N. Transport through H keeps the dense schedule on the same metric carrier as the complete-metric handoff, and replay through C records the finite request route.

Finally invoke Definition 1490.15. Since the carrier has no row for selected dense points, countable choice, quotient completion representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the only exported dense-schedule handoff is the displayed finite NameCert route. □

Theorem 1490.18 (Polish-space complete-separable source lock)**.** *For an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet* $P = (M, K, D, S, R, W, H, C, G, N)$*, every complete-separable consumer reads the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K *and the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule* D *over the same* $\mathsf{Met}^{\uparrow}$ *row* M*. The componentwise transport row* H *may move* K *and* D *only together with* M, S, R, W, C, G, N*, so the complete side cannot be sourced from one metric carrier while the separable side is sourced from another.*

Proof. Project the carrier of Definition 1490.1. The rows K and D are both stored beside the single metric row M, while S, R, W provide the finite observation route before either analytic row is consumed.

Apply the NameCert obligations and complete-metric source lock. The classifier-transport clause of Definition 1490.2 fixes the common metric surface, and Definition 1490.16 keeps the complete-metric handoff tied to M after the observation rows have been displayed.

Finally use the separable dense schedule handoff. Theorem 1490.17 routes D through the same S, R, W support, while H, C, G, N only transport, replay, source, and name that packet. Thus complete separability is read from one displayed metric source. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableSourceLock`

Theorem 1490.19 (Polish-space dense completion ledger refusal)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every dense-completion read exported by* P *factors through the*

stream windows S, regular rational readback R, and observation ledger W before the rows K and D are consumed. The packet refuses selected dense points, quotient completion equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness as sources for the complete-separable handoff.*

Proof. Start from the separable dense schedule handoff. Theorem 1490.17 requires every read of D to pass through S, R, W before the dense schedule is exported.

The completion side follows the same ledger. Theorems 1490.15 and 1490.16 require completion-facing reads of K to use the displayed stream windows, regular readbacks, and observation ledger over M.

Now apply the L10 consumer boundary. Theorem 1490.23 excludes selected dense points, quotient completion equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the public boundary. Since H, C, G, N only preserve the displayed ledger route, no refused object can replace S, R, W. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceDenseCompletionLedger refusal`

Theorem 1490.20 (Polish-space StreamName readback stability)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Transporting the packet along the componentwise* hsame *row H and replaying the same request through C preserves the* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R componentwise before any complete-separable synthesis is consumed. The transported synthesis therefore reads the same finite stream and regular-rational observations through W, G, N.*

Proof. Begin with metric readback stability. Theorem 1490.25 states that componentwise transport through H preserves the metric, completion, dense-schedule, stream, regular-readback, ledger, replay, provenance, and naming rows.

Apply observation-ledger exhaustion. Theorem 1490.13 forces every dense-schedule and completion read through the finite ledger W, with windows supplied by S and rational observations supplied by R.

Finally attach complete-separable source lock. Theorem 1490.18 keeps the complete and separable rows over the same transported M. Hence the complete-separable synthesis cannot alter the StreamName or RegSeqRat readbacks before it is consumed. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamReadbackStability`

Theorem 1490.21 (Polish-space complete separable synthesis)**.** *Every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet synthesizes a* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *read over the same metric carrier named by M. The synthesis consumes the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K, the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the ledger route W, H, C, G, N; it exports no selected dense points, quotient completion equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableSynthesis`

Proof. Start with the root obligation package of Definition 1490.14. The metric row M is fixed before both completion and separability are read, so the synthesis is over one displayed metric carrier.

The completion side is supplied by K, and the separability side is supplied by D. The dense schedule can be inspected only through S, while regular rational observations pass through R after their finite windows have been displayed.

The ledger exhaustion theorem, Definition 1490.13, keeps both sides inside W, H, C, G, N. Since the carrier contains no selected dense-point row, quotient-completion equality, countable-choice coordinate, or ambient real-completeness row, the resulting complete separable metric read is exactly the finite synthesis surface named above. □

Theorem 1490.22 (Polish-space complete separable non-escape)**.** *Every* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet factors through the finite route*

$$M, K, D, S, R, W, H, C, G, N.$$

The route is forced by Definition 1490.2, exhausted by Definition 1490.13, and repacked by Definition 1490.21. Hence the synthesis exports no selected dense points, quotient completion equality, countable-choice schedule, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableNonescape`

Proof. Start with a $\mathsf{CompleteSeparableMetric}^{\uparrow}$ consumer of an accepted packet. By Definition 1490.2, the packet presents only the metric carrier M, completion handoff K, separable schedule D, stream windows S, regular readback R, and support rows W, H, C, G, N.

Use Definition 1490.13. It forces dense-schedule reads and completion reads through the observation ledger W, componentwise transport H, displayed replay C, provenance G, and local name N before the consumer can use the complete or separable side.

Finally apply Definition 1490.21. The synthesis repacks the same rows as a complete-separable metric read over M. Since none of those rows stores selected dense points, quotient completion equality, a countable-choice schedule, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no such datum can be exported by the consumer. □

Theorem 1490.23 (Polish-space L10 consumer boundary)**.** *An L10 consumer of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet may import only the finite boundary consisting of* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{SeparableMetric}^{\uparrow}$*,* $\mathsf{CompleteSeparableMetric}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, the observation ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The boundary excludes selected dense points, quotient completion equality, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet through Definition 1490.14. This gives the finite L10 rows M, K, D, S, R and the support rows W, H, C, G, N before any downstream consumer can inspect the packet.

Use Definition 1490.21. It repacks the completion and separability sides as a $\mathsf{CompleteSeparableMetric}^{\uparrow}$ read over the same metric carrier while preserving the stream windows, regular readback, and ledger route.

The exclusion list follows from the carrier and ledger exactness. None of the displayed rows supplies a selected dense point, quotient-completion equality, countable-choice schedule, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum, so an L10 consumer cannot cite any of those objects through $\mathsf{PolishSpace}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCarrier l10 consumer boundary`

Theorem 1490.24 (Polish-space completion dense schedule obligation)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every completion-facing dense-schedule obligation exported from* P *factors through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *schedule* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the observation ledger*

W, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and the local* NameCert *row N.*

The obligation is exactly the finite route certified by Definition 1490.2, constrained by Definition 1490.13, synthesized by Definition 1490.21, and bounded at L10 by Definition 1490.23. It exposes no choice-selected dense point, quotient completion, countable-choice schedule, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 1490.1. The displayed analytic rows are precisely M, K, D, S, R, and the support rows are W, H, C, G, N.

Apply Definition 1490.2 to read those rows as local NameCert obligations. The exactness clause makes W the only route for metric-radius checks, Cauchy windows, dense-name inspections, and regular rational readbacks.

Use Definition 1490.13 and Definition 1490.21. They force the completion side and dense side through the same finite stream and readback route before the complete-separable metric synthesis can be consumed.

Finally apply Definition 1490.23. Since the boundary contains no coordinate for selected dense points, quotient completions, countable choice, open-cover compactness, or ambient real completeness, the exported obligation is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionDenseScheduleObligation`

Theorem 1490.25 (Polish-space metric readback stability)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Suppose a downstream consumer transports the metric row, completion handoff, dense schedule, stream windows, regular rational readback, and ledger rows along the componentwise* hsame *data recorded in H and replays the same request through the displayed* Cont *routes in C. Then the transported packet preserves the same* $\mathsf{Met}^{\uparrow}$ *readback boundary for M, the same* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{SeparableMetric}^{\uparrow}$ *handoff surface for K, D, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite observations through S, R, W, G, N.*

Thus metric readback stability is a finite NameCert property of P and not a quotient-completion equality, selected dense representative, countable-choice enumeration, or host-level equality of metric spaces.

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceMetricReadbackStability`

Proof. Begin with the carrier projection and the componentwise transport row H. The row transports M, K, D, S, R, W together with the local certificate data, so a metric readback cannot move independently from the completion or dense-schedule rows.

Apply Definition 1490.2. Its classifier-transport and stability clauses require K and D to remain over the transported metric surface before either row is consumed.

Next use Definition 1490.13. Every finite observation read by the transported consumer still passes through S, R, W, and the replay row C records the same request route.

The L10 boundary theorem, Definition 1490.23, supplies the exclusion list. Because no accepted packet coordinate stores quotient completion equality, selected dense representatives, countable-choice enumerations, or host-level metric equality, the stability claim is only the displayed finite transport-and-replay property. □

Theorem 1490.26 (Polish-space L10 root unblock package)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The L10 root obligation package exported by P consists of the*

completion-dense schedule route of Definition 1490.24, the metric readback stability route of Definition 1490.25, the finite observation exhaustion of Definition 1490.13, the complete-separable synthesis of Definition 1490.21, and the consumer boundary of Definition 1490.23.

Consequently every downstream complete-separable metric chapter may import only $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows through the displayed tuple* $P = (M, K, D, S, R, W, H, C, G, N)$. *The package does not supply choice-selected dense points, quotient completions, countable-choice schedules, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceL10RootUnblockPackage`

Proof. First assemble the finite route. The completion-dense schedule obligation displays $M, K, D, S, R, W, H, C, G,$ before any dense or complete read leaves the chapter boundary.

Next apply metric readback stability. It shows that transported consumers still read the same metric, completion, dense-schedule, stream, regular-readback, ledger, replay, provenance, and naming rows.

The observation-ledger and complete-separable synthesis theorems then repack this route as a $\mathsf{CompleteSeparableMetric}^{\uparrow}$ surface over the metric row M. This is the only synthesis route available to an L10 consumer.

Finally invoke the consumer boundary theorem. It permits the listed BEDC rows and refuses selected dense points, quotient completions, countable choice, host equality, open-cover compactness, and ambient real completeness, so the root unblock package is exactly the displayed finite package. □

Theorem 1490.27 (Polish-space complete-separable consumer package)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every L10 consumer that reads* P *as a complete-separable metric input must receive a finite* BHist *route whose visible rows are the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the observation ledger* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and the local* NameCert *row* N *over the same metric source* M*.*

The package refuses choice-selected dense points, quotient equality of completed points, arbitrary open covers, host completion, and any complete-separable witness not routed through the displayed tuple $P = (M, K, D, S, R, W, H, C, G, N)$*.*

Proof. Project the accepted carrier. The row K is the completion-facing input, D is the separability-facing input, and S, R, W are the finite stream, regular-readback, and observation rows that make those inputs consumable.

Apply Definition 1490.26. It lists exactly the routes that an L10 root obligation may export from P and keeps the support rows H, C, G, N attached to the same packet.

Use Definition 1490.23. Since the carrier has no row for a chosen dense point, quotient equality, arbitrary open cover, host completion, or ambient completion object, a complete-separable consumer can only receive the displayed finite package. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceL10CompleteSeparableConsumerPackage`

Theorem 1490.28 (Polish-space completion schedule stability)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Any componentwise* hsame *transport of the completion side preserves the same schedule route from the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K *through the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the observation ledger* W*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and the local* NameCert *row* N *over* M*.*

Thus the transported schedule remains a finite BHist *route and cannot be replaced by a choice-selected dense sequence, quotient equality, arbitrary open-cover refinement, host completion, or external completion of the metric carrier.*

Proof. Start from the transport row H. It transports the metric, completion, separability, stream, readback, ledger, replay, provenance, and naming rows componentwise rather than moving K alone.

Apply Definition 1490.25. The theorem keeps metric readback attached to S, R, W after transport, so completion-side schedule reads still pass through the same finite observation route.

Finally use Definition 1490.13. The ledger exhaustion prevents the transported consumer from substituting selected dense points, quotient equality, arbitrary open covers, host completion, or an external completion object for the displayed schedule. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionScheduleStability`

Theorem 1490.29 (Polish-space observation ledger consumer exactness)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* PolishSpace$^{\uparrow}$ *carrier. For every complete-separable metric consumer of P, the observation ledger W is exact: all requests for the* CompleteMetric$^{\uparrow}$ *row K, the* SeparableMetric$^{\uparrow}$ *row D, the* StreamName$^{\uparrow}$ *windows S, and the* RegSeqRat$^{\uparrow}$ *readback row R are replayed through H, C, G, N before the local* NameCert *exports the result.*

Consequently no consumer can read beyond the displayed BHist *ledger into choice-selected dense points, quotient equality, arbitrary open covers, host completion, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Read the observation ledger theorem, Definition 1490.13. It states that the finite observation route covers metric-radius checks, Cauchy-window checks, dense-name inspections, and regular rational readbacks before any consumer leaves the packet.

Attach the complete-separable synthesis route. By Definition 1490.21, the complete and separable rows are consumed over the same metric source and through the same finite stream and readback rows.

The support rows close exactness. Transport H, replay C, provenance G, and local naming N keep the consumer on the displayed BHist route, so no selected dense point, quotient equality, open-cover argument, host completion, or ambient Real-completeness row is available. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceObservationLedgerConsumerExactness`

Theorem 1490.30 (Polish-space complete-separable metric row exposure)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N$ be an accepted* PolishSpace$^{\uparrow}$ *carrier. Before any complete-separable metric consumer reads P, the packet exposes the* CompleteMetric$^{\uparrow}$ *row K, the* SeparableMetric$^{\uparrow}$ *row D, the synthesized* CompleteSeparableMetric$^{\uparrow}$ *surface, the* StreamName$^{\uparrow}$ *finite windows S, the* RegSeqRat$^{\uparrow}$ *readback row R, the componentwise* hsame *transport H, the displayed* Cont *replay C, the* Pkg *provenance G, and the local* NameCert *row N over the same* Met$^{\uparrow}$ *row M.*

Thus the complete-separable consumer receives the finite route

$$M, K, D, S, R, W \longmapsto H, C, G, N$$

and no selected dense point, quotient completion, countable-choice schedule, open-cover compactness principle, host equality of metric spaces, or ambient Real$^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCompleteSeparableMetricRow`

Proof. Project the carrier through Definition 1490.1. The metric source is M, the completion handoff is K, and the dense schedule is D; the stream, regular readback, and observation rows are S, R, W.

Apply Definition 1490.21. It repacks K and D as the complete-separable metric surface only after the stream windows and regular rational readback have been made visible over the same metric carrier.

Use Definition 1490.26. The package keeps the support rows H, C, G, N attached to the same displayed route, so a consumer cannot read complete separability before the listed rows have been exposed. □

Theorem 1490.31 (Polish-space stream-density readback row)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every dense-schedule consumer that reads the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D must first pass through the* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, then replay the observation ledger W through H, C, G, N before treating the schedule as a complete-separable metric input.*

In particular, a consumer cannot read separability from D as a bare countable enumeration, choice-selected dense point, quotient-completion representative, open-cover compactness argument, or ambient real-completeness witness.

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamDensityReadbackRow`

Proof. Start with the dense side of Definition 1490.24. It places D downstream of the metric row M and ties it to the finite stream windows S and regular rational readback R.

Read the observation ledger next. By Definition 1490.13, metric-radius checks, Cauchy windows, dense-name inspections, and regular readbacks all pass through W before the packet exports a separability-facing read.

Finally use Definition 1490.25. Transport, replay, provenance, and local naming preserve the same S, R, W route, so a dense-schedule consumer receives finite stream-density readback rather than any hidden enumeration or completion principle. □

Definition 1490.32 (Polish-space root carrier)**.** *The root-facing* $\mathsf{PolishSpace}^{\uparrow}$ *carrier is the tuple $P = (M, K, D, S, R, W, H, C, G, N)$ of Definition 1490.1, read as a single* BHist *packet whose consumer-visible rows are* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg *provenance, and local* $\mathsf{NameCert}$ *data. The root view has no additional row for selected dense points, quotient completions, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootCarrier rows exposed`

Theorem 1490.33 (Polish-space separable complete metric obligations)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted root-facing* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *obligation exported from P factors through the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K, the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the observation ledger W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and the local* $\mathsf{NameCert}$ *row N.*

Proof. Start with Definition 1490.32. The root carrier is only the displayed tuple $P = (M, K, D, S, R, W, H, C, G, N$

Apply Definition 1490.30. It exposes the complete, separable, stream, regular-readback, replay, provenance, and naming rows over the same metric source before any complete-separable consumer reads the packet.

Finally use Definition 1490.31. It forces dense-schedule reads through S, R, W, H, C, G, N, so the complete-separable obligation has exactly the listed finite route. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCarrier separable complete metric obligations`

Theorem 1490.34 (Polish-space NameCert ledger transport)**.** *For an accepted root-facing* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, componentwise* hsame *transport may move the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{SeparableMetric}^{\uparrow}$*,* $\mathsf{CompleteSeparableMetric}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows only together with the observation ledger* W*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and local* NameCert *row* N*. No transported consumer can read a complete-separable metric surface after dropping the ledger.*

Proof. Read the componentwise transport row H from Definition 1490.32. It is part of the same finite packet as the metric, completion, separability, stream, regular-readback, and ledger rows.

Apply Definition 1490.25. Transport preserves the metric readback boundary together with the StreamName and RegSeqRat finite observations through S, R, W, G, N.

The L10 root package of Definition 1490.26 then keeps every complete-separable consumer inside that transported ledger. Thus ledger-free transport is not an exported row of the packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceNamecertLedgerTransport`

Theorem 1490.35 (Polish-space standard readiness)**.** *An accepted root-facing* $\mathsf{PolishSpace}^{\uparrow}$ *carrier is ready for standard complete-separable metric consumers only as the displayed* BHist *route through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{SeparableMetric}^{\uparrow}$*,* $\mathsf{CompleteSeparableMetric}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. A downstream* $\mathsf{Real}^{\uparrow}$ *consumer may receive only the regular-readback side exposed through this route; it does not receive ambient Real completeness or host equality of metric spaces.*

Proof. Use Definition 1490.33. It displays the complete-separable metric route over $M, K, D, S, R, W, H, C, G, N$.

Apply Definition 1490.34. Standard consumers must preserve the observation ledger and replay rows when they move the metric, completion, separability, stream, and readback data.

The $\mathsf{Real}^{\uparrow}$ clause is only a downstream boundary. The root carrier exposes regular rational readback through R but has no coordinate for ambient Real completeness or host metric equality, so those claims cannot be imported. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceStandardReadiness`

Theorem 1490.36 (Polish-space L10 root unblock package)**.** *The root-facing* $\mathsf{PolishSpace}^{\uparrow}$ *packet unlocks L10 metric-completion consumers exactly by the finite carrier of Definition 1490.32, the complete-separable metric obligations of Definition 1490.33, the ledger transport of Definition 1490.34, and the standard readiness boundary of Definition 1490.35. It exports no selected dense point, quotient completion, countable-choice schedule, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Assemble the carrier route first. The definition exposes $M, K, D, S, R, W, H, C, G, N$ as one finite BHist packet.

The separable complete metric obligations show how $\mathsf{CompleteSeparableMetric}^{\uparrow}$ consumers read that packet, and ledger transport keeps the rows tied together under hsame movement.

Finally apply standard readiness. The only exported L10 boundary is the displayed route through metric, completion, separability, stream, regular readback, transport, replay, provenance, and local naming rows, with the listed objects excluded from the carrier. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceL10RootUnblockPackage`

Theorem 1490.37 (Polish-space effective-representation handoff)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Any* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *or* $\mathsf{ComputablePolishSpace}^{\uparrow}$ *consumer of P must first expose the Polish complete-separable package formed by the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K, the countable dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the observation ledger W, the componentwise* hsame *transports H, the displayed* Cont *replay C, the* Pkg *provenance G, the local* NameCert *row N, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route before represented-space or computable-analysis vocabulary is read.*

Consequently the represented-space boundary of chapter 1494, the admissible-representation boundary of chapter 1760, the effective Polish packet of chapter 1706, and the computable Polish packet of chapter 4146 can consume $\mathsf{PolishSpace}^{\uparrow}$ *only through the displayed finite route of this chapter. They receive no selected dense points, quotient completion equality, unscheduled name oracle, host representation equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Polish carrier projection. The complete-separable source is displayed by the metric row M, complete-metric handoff K, separable dense schedule D, finite stream windows S, regular readback R, and observation ledger W.

Apply Definition 1490.12. Any dense-schedule read first opens the $\mathsf{StreamName}^{\uparrow}$ schedule and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before it can consume the separable row. Apply Definition 1490.31 to keep that dense schedule attached to W, H, C, G, N.

Next apply Definition 1490.13. Completion reads and dense reads both pass through the same finite ledger, so an effective or computable Polish consumer cannot replace the displayed route by an external representation source.

Finally read the sibling consumer surfaces. The $\mathsf{EffectivePolishSpace}^{\uparrow}$ and $\mathsf{ComputablePolishSpace}^{\uparrow}$ chapters import $\mathsf{PolishSpace}^{\uparrow}$ only after their dense-realizer, represented-space, StreamName, RegSeqRat, and Real-seal rows have been displayed; the represented-space and admissible-representation chapters likewise require scheduled names and regular readback before target-facing vocabulary is available. Since P has no coordinate for selected dense points, quotient completions, host representation equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the handoff is exactly the finite route named in the statement. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceEffectiveRepresentationHandoff`

Theorem 1490.38 (PolishSpace effective-name root obligations)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier $P = (M, K, D, S, R, W, H, C, G, N)$, any effective-name consumer of P reaches the displayed names only through the* $\mathsf{StreamName}^{\uparrow}$ *row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the observation ledger W, the componentwise* hsame *rows H, the replay row C, the provenance row G, and the local* NameCert *row N of Definition 1490.1. It receives no unscheduled dense-name oracle, selected dense point, quotient completion representative, host equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootEffectiveNameObligations`

Proof. Project the carrier first. The rows S, R, W, H, C, G, N are displayed coordinates of the same finite packet P, so an effective-name read cannot leave the packet before the schedule and readback rows have been opened.

Apply Definition 1490.12 and Definition 1490.31. These theorems identify the dense schedule and finite readbacks as the route by which downstream effective names inspect the Polish carrier.

The excluded objects are not coordinates of Definition 1490.1. Transport, replay, provenance, and local naming preserve the displayed route and do not add an external oracle. □

Theorem 1490.39 (PolishSpace complete-separable root schedule)**.** *Every complete-separable consumer of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier factors through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *schedule* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* W, H, C, G, N*. In particular, a* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *read cannot be separated from the displayed finite observation schedule of the same packet.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootCompleteSeparableSchedule`

Proof. Start from Definition 1490.1. The metric, completion, separability, stream, readback, ledger, transport, replay, provenance, and naming rows are carried together as $M, K, D, S, R, W, H, C, G, N$.

Use Definition 1490.33 to read the complete and separable obligations from the displayed carrier. Then use Definition 1490.34 to keep those obligations attached to W, H, C, G, N under componentwise movement.

Thus a downstream complete-separable read receives the finite schedule and readback route, not an independent completion or dense-set choice principle. □

Theorem 1490.40 (PolishSpace represented-space root handoff)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Any handoff from* P *to a* $\mathsf{RepresentedSpace}^{\uparrow}$ *consumer must expose the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *schedule* D*, and the structural rows* W, H, C, G, N *before representation vocabulary is available.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootRepresentationHandoff`

Proof. Open the represented-space boundary through Definition 1490.37. That theorem already fixes the route from the Polish packet to represented and effective Polish consumers.

The root rows in the statement are precisely the rows used by that handoff: completion comes from K, separability from D, scheduled names from S, regular readback from R, and structural preservation from W, H, C, G, N.

Since no host representation equality, selected name, quotient completion, or countable-choice selector is present in P, the represented-space consumer can read only this finite route. □

Theorem 1490.41 (Polish-space effective-representation separable readback)**.** *Let* $P = (M, K, D, S, R, W, H, C, G$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Any represented-space, admissible-representation, effective Polish, or computable Polish consumer of* P *must display the complete-separable metric row, the dense schedule, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *support before it reads an effective representation. Thus the represented or computable vocabulary is downstream of the same finite route*

$$M, K, D, S, R, W, H, C, G, N$$

rather than a detached name oracle, selected dense sequence, host equality, countable-choice schedule, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceEffectiveRepresentationSeparableReadback`

Proof. Start with Definition 1490.37. It already forces represented-space, admissible-representation, effective Polish, and computable Polish consumers to expose the metric row, complete-metric row, dense schedule, stream windows, rational readback, observation ledger, transport, replay, provenance, local naming, and terminal real seal before downstream vocabulary is available.

Next apply Definition 1490.12 and Definition 1490.31. These two rows make the dense schedule readable only through the $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to W, H, C, G, N.

Finally use Definition 1490.36. The root-facing package exports only the complete-separable metric synthesis route and excludes selected dense points, quotient completion, countable choice, host equality, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. The effective-representation readback therefore consumes exactly the displayed Polish packet. □

Theorem 1490.42 (Polish-space effective complete-separable obligation)**.** *Every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *complete-separable consumer reads the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-schedule row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the Real-facing observation ledger* W*, and the structural rows* H, C, G, N *through the same finite packet before any completion use is public. The obligation exports no selected dense sequence, quotient completion equality, unscheduled name oracle, host representation equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceEffectiveCompleteSeparableObligation`

Proof. Apply Definition 1490.43. It identifies the complete-separable surface as the displayed carrier route through metric, completion, separability, stream windows, regular readback, observation, transport, replay, provenance, and local naming rows.

Apply Definition 1490.46. Completion-facing reads cannot leave that finite route for a quotient completion, selected limit representative, or host completeness principle.

Finally apply Definition 1490.23. The L10 consumer boundary keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the Real-facing rows in one packet, so an effective complete-separable consumer receives the shared obligation surface and no additional completion coordinate. □

Falsifiable prediction: An accepted $\mathsf{PolishSpace}^{\uparrow}$ consumer that reaches a $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, or $\mathsf{Met}^{\uparrow}$ row without reading the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, finite observation ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transports, Cont routes, Pkg provenance, and local $\mathsf{NameCert}$ row refutes the carrier surface.

Independence witness: metric, completemetric, separablemetric, streamname, regseqrat: no listed sibling supplies the whole Polish packet $P = (M, K, D, S, R, W, H, C, G, N)$, because each sibling contributes only the distance carrier, completion handoff, dense schedule, stream schedule, or rational readback while this chapter binds the complete-separable metric synthesis route with transport, replay, provenance, and local naming.

Theorem 1490.43 (Polish-space complete-separable synthesis)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Its root complete-separable route exposes the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K *and the* $\mathsf{SeparableMetric}^{\uparrow}$ *row* D *only over the same* $\mathsf{Met}^{\uparrow}$ *carrier* M*, with* $\mathsf{StreamName}^{\uparrow}$ *windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, observation ledger* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and local* $\mathsf{NameCert}$ *rows* N*. Consequently every complete-separable consumer of* P *reads one common route*

$$M \longmapsto (K, D) \longmapsto (S, R, W) \longmapsto (H, C, G, N),$$

and no choice-selected dense sequence, external completion coordinate, separate host metric, or quotient-completion equality is exported.

Proof. Project P by Definition 1490.1. The carrier exposes $M, K, D, S, R, W, H, C, G, N$ as displayed rows of one finite BHist packet.

Apply Definition 1490.2. Carrier admission places the complete-metric handoff K and the separable-metric schedule D inside the same local $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ surface, while classifier transport uses H to compare both rows only after transport to the common metric source M.

Use Definition 1490.3 and Definition 1490.4. The density obligation forces D to be read through S, R, W over M, and the completion obligation forces K to use the same metric source and the same finite observation route.

It remains to package the shared readback. The rows C, G, N replay, source, and name the route already fixed by M, K, D, S, R, W, H. Because the carrier has no coordinate for the excluded dense selector, external completion, separate host metric, or quotient-completion equality, the displayed route is the only complete-separable synthesis exported by an accepted Polish-space packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootCompleteSeparableSynthesis`

Theorem 1490.44 (Polish-space complete-separable shared-source route)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted $\mathsf{PolishSpace}^{\uparrow}$ packet. Every complete-separable consumer must read the complete row K and the separable row D over the same $\mathsf{Met}^{\uparrow}$ row M, then pass through the finite route*

$$M \longmapsto (K, D) \longmapsto (S, R, W) \longmapsto (H, C, G, N).$$

It cannot split completion and separability across different host metrics, separate quotient-completion coordinates, selected dense sequences, or unscheduled readback windows.

Proof. Apply Definition 1490.43. It already displays K and D as rows of one accepted packet over the same metric source M.

Next use Definition 1490.3. The dense schedule D is read through the shared stream and regular rational windows S, R together with the observation ledger W, so it cannot be detached from the metric row that supplies its distances.

Use Definition 1490.4 for the complete side. The completion row K is transported and replayed through the same H, C, G, N route as D.

The carrier has no coordinate for a second host metric, separate quotient completion, selected dense sequence, or unscheduled readback window. Thus the complete-separable route is a shared-source route through $M, K, D, S, R, W, H, C, G, N$. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableSharedSourceRoute`

Definition 1490.45 (Polish-space root unblock surface)**.** *The root unblock surface of an accepted $\mathsf{PolishSpace}^{\uparrow}$ packet is the displayed finite packet*

$$(M, K, D, S, R, W, H, C, G, N)$$

viewed simultaneously as the $\mathsf{Met}^{\uparrow}$ carrier row M, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff K, the $\mathsf{SeparableMetric}^{\uparrow}$ dense schedule D, the $\mathsf{StreamName}^{\uparrow}$ window row S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, the observation ledger W, componentwise hsame transport H, displayed Cont replay C, Pkg provenance G, and local $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ row N.

A downstream root consumer may read complete-separable data only through this single surface. It receives no separate metric carrier, selected dense point, quotient completion equality, unscheduled name oracle, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum.

Lean4 (definition) `BEDC.Derived.PolishspaceUp.PolishSpaceRootUnblockSurface`

Theorem 1490.46 (Polish-space complete-separable unblock package)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. The complete row K and the separable row D are consumed over one metric carrier M: every complete-separable root read first exposes M, then reads both K and D through the same S, R, W observation surface, and finally replays the common packet through H, C, G, N.*

Consequently the complete side cannot be sourced from one carrier while the separable side is sourced from another, and neither side can bypass the finite StreamName and RegSeqRat windows.

Proof. Start from Definition 1490.45. The displayed packet has a single metric coordinate M, and both K and D are rows of that same packet.

Use the complete-separable synthesis of Definition 1490.43. It already routes every complete-separable consumer through

$$M \longmapsto (K, D) \longmapsto (S, R, W) \longmapsto (H, C, G, N).$$

It remains to rule out split consumption. The observation ledger W and the componentwise transport row H move K and D only together with M, S, R, C, G, N. Thus a consumer has no coordinate from which to read a second carrier, an unscheduled complete row, or a dense schedule outside the displayed finite windows. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableUnblockPackage`

Theorem 1490.47 (Polish-space RegSeqRat window unblock)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Every downstream Polish-space read that reaches a* $\mathsf{Real}^{\uparrow}$*-facing consumer passes first through the* $\mathsf{StreamName}^{\uparrow}$ *row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R of Definition 1490.45. The read may then use the complete row K, the separable row D, and any Real-facing seal already exposed by the packet, but it cannot replace S, R by a selected dense point, a quotient-completion representative, host equality, or ambient completeness.*

Proof. Expose the root unblock surface. The only rows that schedule names and regular rational observations are S and R, and the local NameCert row names them as part of the same finite packet as M, K, D, W, H, C, G.

The Real-facing boundary is downstream of those observations. A consumer can reach Real-facing data only after the stream windows have supplied finite requests and the regular rational readback has supplied the corresponding observations.

Finally apply the nonescape clauses of the complete-separable synthesis. Since the packet has no row for selected dense points, quotient-completion representatives, host equality, or ambient completeness, every downstream read passes through the StreamName and RegSeqRat windows before any Real-facing consumer is available. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRegSeqRatWindowUnblock`

Theorem 1490.48 (Polish-space RegSeqRat–Real handoff)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet whose complete-separable route is read through the root unblock surface of Definition 1490.45. Any* $\mathsf{Real}^{\uparrow}$*-facing consumer of that route can be reached only after the* $\mathsf{StreamName}^{\uparrow}$ *row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R expose their finite windows over the same* $\mathsf{Met}^{\uparrow}$ *row M that supports the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K and the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D. The handoff exports no selected dense point, quotient completion representative, host equality, or ambient completeness coordinate.*

Proof. Start from the root unblock surface. The rows $M, K, D, S, R, W, H, C, G, N$ are displayed as one finite packet, so the stream and regular-rational windows are already tied to the same metric row as the complete and separable data.

Apply Definition 1490.46. It forces both K and D to be consumed over M and through the common S, R, W observation surface before the packet can be replayed by H, C, G, N.

Now use Definition 1490.47. Every downstream read that reaches a Real-facing consumer must pass through S and R first. Combining the two routes gives the required top-window handoff from complete separability to the finite RegSeqRat readback.

The excluded dense selector, quotient completion representative, host equality, and ambient completeness rows are absent from the carrier. Therefore the Real-facing consumer receives only the displayed route over M. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableRegSeqRatRealHandoff`
↪ variant `BEDC.Derived.PolishspaceUp.PolishSpaceRegSeqRatRealHandoff`

Definition 1490.49 (Polish-space root-unblock carrier)**.** *The root-unblock carrier of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet is the finite* BHist *row*

$$U = (M, K, D, S, R, B, H, C, P, N)$$

read as follows:

1. M *is the common* $\mathsf{Met}^{\uparrow}$ *source row.*
2. K *is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row over* M*.*
3. D *is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-schedule row over* M*.*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for dense-name and Cauchy-window inspections.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. B *is the supporting* BHist *observation ledger, including metric-radius requests, complete-metric requests, separable-density requests, and the finite links among them.*
7. H *records componentwise* hsame *transport for* M, K, D, S, R, B *and the local certificate rows.*
8. C *records displayed* Cont *replay from each root consumer request through the same complete-separable packet.*
9. P *is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row.*

The carrier excludes selected dense points, quotient completion equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1490.50 (Polish-space common metric root unblock)**.** *Let* $U = (M, K, D, S, R, B, H, C, P, N)$ *be a Polish-space root-unblock carrier. Every complete-separable root consumer reads the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K *and the* $\mathsf{SeparableMetric}^{\uparrow}$ *row* D *only over the same* $\mathsf{Met}^{\uparrow}$ *row* M*, with the finite* BHist *ledger* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *recording that common source.*

Consequently the complete side cannot use a different metric carrier from the separable side, and no selected dense point, quotient completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum is exported by the root packet.*

Proof. Project the carrier of Definition 1490.49. The complete-metric handoff K and the separable schedule D are both listed after the same metric source row M.

Use the observation ledger B to read root requests. Each complete-metric request and each separable-density request is recorded with the same metric-radius surface, so a downstream consumer cannot split the two sides across different carriers.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no selected dense point, quotient completion equality, countable-choice schedule, or ambient real-completeness coordinate appears in the carrier, the common metric row exhausts the root-unblock source. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootUnblockCommonMetric`

Theorem 1490.51 (PolishSpace complete-separable obligation surface)**.** *Let $U = (M, K, D, S, R, B, H, C, P, N)$ be a Polish-space root-unblock carrier. The complete-separable obligation surface is the joint read in which M is the* $\mathsf{Met}^{\uparrow}$ *carrier, K supplies the* $\mathsf{CompleteMetric}^{\uparrow}$ *convergence rows over M, D supplies the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense enumeration rows over the same M, S supplies finite* $\mathsf{StreamName}^{\uparrow}$ *observations, R supplies* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy witnesses and the resulting* $\mathsf{Real}^{\uparrow}$ *seals for observed distances, and B, H, C, P, N record the observation ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *transport.*

Every downstream complete-separable consumer must factor through this surface before reading any Polish-space data. The surface exports no split metric carrier, no selected dense point, no quotient-completion representative, no countable-choice schedule, and no ambient real-completeness principle.

Proof. Project the carrier of Definition 1490.49. The metric carrier M, complete row K, separable dense row D, stream row S, regular rational row R, ledger B, transport row H, replay row C, provenance row P, and local name N are the displayed rows.

The complete and separable sides share the same metric source. By Definition 1490.50, K and D cannot be consumed over different carriers.

The finite observation discipline is supplied by Definition 1490.54. Dense-name requests, Cauchy-window requests, and metric-radius comparisons pass through S and R, with the Real-facing distance seals read only after the regular Cauchy witnesses have been displayed.

Finally B, H, C, P, N transport, replay, source, and name that single route. Since the carrier has no field for the excluded host objects, the listed rows form the complete-separable obligation surface. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceCompleteSeparableObligationSurface`

Theorem 1490.52 (PolishSpace StreamName Real dependency anchor)**.** *Let $U = (M, K, D, S, R, B, H, C, P, N)$ be a Polish-space root-unblock carrier. Any root consumer that reaches a* $\mathsf{Real}^{\uparrow}$*-sealed distance, convergence tolerance, or dense-radius observation must first read the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy witness row R. The dependency then returns to the* $\mathsf{Met}^{\uparrow}$ *carrier M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *convergence rows K, and the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense enumeration rows D through the* BHist *ledger B, componentwise transport H, replay C, provenance P, and local* NameCert *row N.*

Thus Real-facing data in a $\mathsf{PolishSpace}^{\uparrow}$ *root packet are anchored by scheduled observations and regular Cauchy witnesses, not by an external real-number oracle or host metric completion.*

Proof. Expose the stream-facing fields of Definition 1490.49. The only row that schedules finite observations is S, and the only row that turns those observations into regular rational Cauchy data is R.

Read a Real-facing request. The request must pass through B, where dense-radius, Cauchy-window, and metric comparison observations are paired with their stream positions and regular rational witnesses.

The analytic rows are then reached over the same packet. The metric carrier M, complete convergence row K, and dense enumeration row D are transported together by H and replayed through C.

The provenance and naming rows P, N do not add a second Real-facing source. Therefore the dependency anchor is exactly the StreamName and RegSeqRat route followed by the common metric, complete, separable, ledger, transport, replay, provenance, and NameCert rows. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceStreamnameRealDependencyAnchor`
↪ variant `BEDC.Derived.PolishspaceUp.PolishSpaceStreamnameRealDependencyAnchor`

Theorem 1490.53 (PolishSpace NameCert ledger exactness)**.** *Let $U = (M, K, D, S, R, B, H, C, P, N)$ be a Polish-space root-unblock carrier. The local $\mathsf{NameCert}_{\mathsf{PolishSpace}^\uparrow}$ ledger transport is exact: N names precisely the $\mathsf{Met}^\uparrow$ carrier M, the $\mathsf{CompleteMetric}^\uparrow$ convergence rows K, the $\mathsf{SeparableMetric}^\uparrow$ dense enumeration rows D, the finite $\mathsf{StreamName}^\uparrow$ observations S, the $\mathsf{RegSeqRat}^\uparrow$ regular Cauchy witnesses R, the resulting $\mathsf{Real}^\uparrow$ seals recorded by the observation ledger B, and the componentwise* hsame, Cont, *and* Pkg *rows H, C, P.*

Consequently a classified replacement of U preserves the same complete-separable root packet component by component and cannot introduce hidden quotient equality, selected dense names, unrecorded limits, external real seals, or additional NameCert obligations.

Proof. Project the carrier. The local name N appears only after the metric, complete, separable, stream, regular rational, ledger, transport, replay, and provenance rows have been displayed.

Use Definition 1490.51. It identifies the exact complete-separable surface consumed by downstream root readers, including the Real-facing seals recorded by B after S and R have supplied finite observations and regular Cauchy witnesses.

Now transport the packet componentwise. The row H compares each displayed coordinate with its replacement, C replays the same requests, and P preserves the stated provenance.

Since no other field remains in the root-unblock carrier, N cannot name an additional obligation. The NameCert ledger transport is therefore exact for the listed rows and excludes the hidden host data named in the statement. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceNamecertLedgerExactness`

Theorem 1490.54 (Polish-space window-discipline root unblock)**.** *Let $U = (M, K, D, S, R, B, H, C, P, N)$ be a Polish-space root-unblock carrier. Every root read of dense-name coverage, Cauchy-window completion data, or metric-radius comparison factors through the $\mathsf{StreamName}^\uparrow$ row S and the $\mathsf{RegSeqRat}^\uparrow$ row R before it is consumed by M, K, D. The supporting* BHist *row B records exactly those windows and readbacks, while H, C, P, N preserve the same route for downstream complete-separable consumers.*

Thus a Polish-space root consumer receives finite window discipline over one complete-separable packet, not an unscheduled name oracle, selected dense sequence, quotient completion equality, countable-choice row, or ambient $\mathsf{Real}^\uparrow$ completeness principle.

Proof. Start from the carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to it is R.

Read a dense-name, Cauchy-window, or metric-radius request through B. By definition of the root-unblock carrier, B records those requests only together with the finite stream positions and regular rational observations supplied by S and R.

It remains to route the analytic rows. The complete row K and separable row D are consumed only after the window and readback rows have been displayed, and the common metric row M is the metric source for both.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no unscheduled oracle, selected dense sequence, quotient equality, choice row, or ambient completeness coordinate, every root read respects the displayed window discipline. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRootUnblockWindowDiscipline`

Definition 1490.55 (Polish-space root Cauchy-basis carrier)**.** *The root Cauchy-basis carrier of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet is the finite* BHist *row*

$$W = (M, K, D, S, R, B, A, H, C, P, N)$$

where $M, K, D, S, R, B, H, C, P, N$ *have the meanings of Definition 1490.49, and* A *is the finite Cauchy-basis alignment row binding the complete-metric row* K *and the separable dense-schedule row* D *over the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback data.*

The alignment row records only the scheduled dense Cauchy basis seen by the local packet. It does not choose dense points, identify quotient-completion representatives, add a countable-choice schedule, or assert ambient $\mathsf{Real}^{\uparrow}$ *completeness outside the displayed readback seals.*

Lean4 (definition) `BEDC.Derived.PolishspaceUp.PolishspaceRootCauchyBasisCarrier`

Theorem 1490.56 (PolishSpace root Cauchy-basis obligations)**.** *Let* $W = (M, K, D, S, R, B, A, H, C, P, N)$ *be a Polish-space root Cauchy-basis carrier. The local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *surface consists exactly of common metric admission through* M*, complete-metric handoff through* K*, separable-density handoff through* D*,* $\mathsf{StreamName}^{\uparrow}$ *window discipline through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*, Cauchy-basis alignment through* A*, observation ledger support through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

No local obligation exports a selected dense point, quotient-completion equality, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or second metric carrier.*

Proof. Project the carrier of Definition 1490.55. The rows $M, K, D, S, R, B, A, H, C, P, N$ are displayed as finite coordinates, with A placed after the stream and regular-rational readback rows that make the dense Cauchy basis observable.

The common metric admission is inherited from Definition 1490.50. Both the complete-metric row K and the separable-density row D are read only over M.

The finite observation discipline is inherited from Definition 1490.54. Dense-name coverage, Cauchy-window data, and metric-radius comparisons pass through S and R before reaching K or D.

It remains to account for the Cauchy-basis row. By definition, A binds the complete and separable reads to the same scheduled dense Cauchy basis over M, S, R; it adds no selected point or quotient representative. The rows B, H, C, P, N then record the ledger, transport, replay, provenance, and local name for exactly that displayed packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootCauchyBasisObligations`

Theorem 1490.57 (PolishSpace completion-separability handoff)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be a Polish-space root Cauchy-basis carrier. Any completion-facing consumer or separability-facing consumer must factor through the single route*

$$M \longmapsto S \longmapsto R \longmapsto A \longmapsto K, D$$

before it is used as a $\mathsf{PolishSpace}^{\uparrow}$ *root read. The rows B, H, C, P, N then record the same observation ledger, componentwise transport, replay, provenance, and local naming for that route.*

Consequently the completion and separability faces cannot be consumed as independent host facts; they meet only through the scheduled dense Cauchy basis aligned by A.

Proof. Start with a completion-facing request. The request has to read the metric source M, then the finite stream schedule S and regular-rational readback R that expose the Cauchy-window observations used by the complete row.

Now read a separability-facing request. Its dense-name coverage uses the same M, S, R route, since Definition 1490.55 gives only one stream row and one readback row for the root packet.

The alignment row A is therefore the unique meeting point before K and D can be consumed together. It binds the complete-metric row and the separable dense-schedule row to the scheduled dense Cauchy basis, while B, H, C, P, N preserve that route for downstream root reads.

Because no selected dense point, quotient-completion equality, choice schedule, or ambient completeness coordinate is present in the carrier, the displayed route is the only completion-separability handoff exported by the packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootCompletionSeparabilityHandoff`

Theorem 1490.58 (PolishSpace root basis-window exactness)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be a Polish-space root Cauchy-basis carrier. The alignment row A is exact only relative to the shared* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. A complete row K and a separable row D are accepted by the root packet only through that same scheduled basis window; they cannot be consumed through a different dense schedule, a host choice of dense points, a quotient-completion representative, or a second metric carrier.*

Thus the basis row records a finite BEDC alignment over M, S, R, not an ambient assertion that completeness and separability may be read independently.

Proof. Project the carrier of Definition 1490.55. The row A is listed precisely as the finite Cauchy-basis alignment binding the complete-metric row K and the separable dense-schedule row D over the same metric source, stream schedule, and regular-rational readback data.

Use Definition 1490.54. Every dense-name, Cauchy-window, or metric-radius read must pass through S and R before it can be consumed by M, K, D, so a second dense schedule has no carrier coordinate.

Now apply Definition 1490.57. The complete and separable faces meet only along the route $M \longmapsto S \longmapsto R \longmapsto A \longmapsto K, D$.

The remaining rows B, H, C, P, N preserve that route as observation ledger, transport, replay, provenance, and local naming. Since the carrier has no field for host-selected dense points, quotient representatives, or another metric source, A is exact only over the displayed M, S, R window. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootBasisWindowExactness`

Theorem 1490.59 (PolishSpace root countable dense Cauchy readback)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. Every countable dense Cauchy-basis read of*

W must expose the finite $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the alignment row A before the complete-metric handoff K or the separable dense-schedule row D can be consumed. The observation ledger B, transport row H, replay row C, provenance row P, and local* $\mathsf{NameCert}$ *row N preserve exactly that route from the common metric source M to the completion and separability faces.*

The readback exports no selected dense point, quotient completion equality, countable-choice schedule, alternate dense enumeration, second metric carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 1490.55. The only countable dense Cauchy-basis route available to the accepted packet is carried by the common metric row M, the stream row S, the regular-rational readback row R, and the alignment row A.

Use Definition 1490.54. Any dense-name, Cauchy-window, or metric-radius request is forced through S and R before it reaches the complete row K or the separable row D.

Now apply Definition 1490.57. The complete and separable faces meet only after the scheduled basis has been aligned by A, so neither K nor D can be consumed as an independent host fact.

The remaining rows B, H, C, P, N are ledger, transport, replay, provenance, and local naming rows for the same route. Since the carrier has no coordinate for selected dense points, quotient representatives, countable choice, a second dense schedule, or ambient Real completeness, the stated readback surface is exact. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootCountableDenseCauchyReadback`

Theorem 1490.60 (PolishSpace root-basis NameCert exhaustion)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be a Polish-space root Cauchy-basis carrier. Before any complete-separable consumer can use Polish-space data, the accepted root packet exposes exactly the rows $M, K, D, S, R, B, A, H, C, P, N$: M is the common* $\mathsf{Met}^{\uparrow}$ *row, K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff, D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule, S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, A is the basis-alignment row, and B, H, C, P, N are the ledger, transport, replay, provenance, and local-name rows.*

The packet exports no selected dense point, countable-choice schedule, quotient-completion representative, second metric carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project Definition 1490.55. It lists precisely the rows $M, K, D, S, R, B, A, H, C, P, N$, with A added as the Cauchy-basis alignment over the root-unblock carrier rows.

Use Definition 1490.50 and Definition 1490.52. The complete row K, separable row D, stream row S, and regular-rational row R all return to the same metric source M and cannot import an external real oracle.

Apply Definition 1490.56 and Definition 1490.57. They place the complete and separable faces on the displayed route through M, S, R, A, K, D and keep B, H, C, P, N as the supporting ledger, transport, replay, provenance, and local name.

Finally apply Definition 1490.58. The basis window is exact only over M, S, R, so no selected dense points, choice schedules, quotient-completion representatives, second metric carriers, or ambient completeness coordinates remain available. The listed rows therefore exhaust the root-basis NameCert surface. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootBasisNamecertExhaustion`

Theorem 1490.61 (PolishSpace root completion-separability exhaustion)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P,$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. If a downstream root packet exposes the common metric row M, complete-metric handoff K, separable-metric basis row D, effective represented-space windows,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport H, replay C, provenance P, and local* NameCert *row N, then the displayed rows $M, K, D, S, R, B, A, H, C, P, N$ exhaust the completion-separability surface. No additional host metric completion, choice-selected dense sequence, quotient representative, or ambient Real-completeness coordinate can be read from the accepted packet.*

Proof. Start with Definition 1490.58. It fixes the complete and separable faces over the same metric source, stream window, and regular-rational readback route, so K and D cannot be consumed through independent schedules.

Apply Definition 1490.60. The accepted root packet already lists the common metric, complete handoff, separable basis, stream window, readback, basis ledger, alignment, transport, replay, provenance, and local-name rows as the whole root-basis surface.

The CompleteMetric and SeparableMetric handoff rows add no external selector. They are available only through M, S, R, A and then through H, C, P, N, while the effective represented-space windows and terminal $\mathsf{Real}^{\uparrow}$ seal are read back through the same finite route.

Since the carrier has no coordinate for host metric completion, countable choice, quotient representatives, or ambient Real completeness, those objects cannot be exported by the local NameCert. The completion-separability surface is therefore exactly the displayed root packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootCompletionSeparabilityExhaustion`

Theorem 1490.62 (PolishSpace root-basis metric-separability determinacy)**.** *Let $W = (M, K, D, S, R, B, A, H, C,$ and $W' = (M, K', D', S, R, B', A, H, C, P, N)$ be accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carriers. If the two carriers agree componentwise on the common* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the alignment row A, and the structural rows H, C, P, N, then the* $\mathsf{CompleteMetric}^{\uparrow}$ *rows K, K' and the* $\mathsf{SeparableMetric}^{\uparrow}$ *rows D, D' expose the same root-basis consumer surface.*

Consequently a downstream complete-separable consumer can read only the common route through M, S, R, A, H, C, P, N. No second metric carrier, selected dense sequence, quotient-completion equality, or countable-choice schedule is determined by either accepted packet.

Proof. Project both accepted carriers through Definition 1490.55. Their shared rows M, S, R, A, H, C, P, N give the same metric source, finite stream windows, regular-rational readback, Cauchy-basis alignment, transport, replay, provenance, and local name.

Use Definition 1490.56 and Definition 1490.57. They force every complete-metric and separable-metric read through the route $M \longmapsto S \longmapsto R \longmapsto A \longmapsto K, D$, with the same route for K', D' because the determining rows are componentwise identical.

Now apply Definition 1490.58 and Definition 1490.59. The basis window is exact over the shared M, S, R data, and countable dense Cauchy reads pass through the same alignment row A before either completion or separability can be consumed.

Finally use Definition 1490.60. The accepted root packet has no coordinate for a second metric carrier, selected dense sequence, quotient-completion equality, or countable-choice schedule. Thus K, K' and D, D' may differ only as internal handoff rows over the same displayed route; their public root-basis consumer surface is the same. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRootBasisMetricSeparabilityDeterminacy`

Theorem 1490.63 (PolishSpace regular-Cauchy root window)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. Every complete-separable consumer of a Polish root window must first display the common* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K, the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R, the basis ledger and alignment rows B, A, the transport and replay rows H, C, the* Pkg *provenance row P, the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row N, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal read through those regular-Cauchy windows.*

The root window exports no selected dense sequence, quotient completion equality, host complete separable space, countable-choice schedule, second metric carrier, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted root Cauchy-basis carrier. The rows $M, K, D, S, R, B, A, H, C, P, N$ are the whole displayed packet, with S and R carrying the regular-Cauchy window data used by both the complete and separable faces.

Apply Definition 1490.57. It routes completion-facing and separability-facing requests through the same M, S, R, A, K, D spine rather than through independent host facts.

Use Definition 1490.59 and Definition 1490.61. The dense Cauchy readback exposes finite stream windows, regular-rational readbacks, and the alignment row before either K or D can be consumed, and the completion-separability surface is exhausted by the listed rows.

Finally apply Definition 1490.62. The determining rows are the common metric, stream window, readback, alignment, transport, replay, provenance, and local name. Since the carrier has no coordinate for selected dense sequences, quotient completion equality, host complete separable spaces, countable choice, a second metric carrier, or ambient Real completeness, the regular-Cauchy root window is exactly the displayed route. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRegularCauchyRootWindow`

Theorem 1490.64 (PolishSpace effective presentation handoff)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. Every effective-presentation read of W must expose the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger B before any consumer treats the PolishSpace packet as presented. The presentation is therefore a finite route through $M, D, S, R, B, A, H, C, P, N$, not a selected dense sequence or host represented-space object.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceEffectivePresentationHandoff`

Proof. Project Definition 1490.1 and Definition 1490.55. The root-basis carrier displays the common metric source M, completion row K, separable schedule D, StreamName row S, RegSeqRat readback R, observation ledger B, alignment row A, and structural rows H, C, P, N.

Use Definition 1490.2. Its carrier and classifier clauses require every public PolishSpace read to expose the finite metric, dense, stream, readback, transport, replay, provenance, and naming rows before it becomes a consumer-facing packet.

Apply Definition 1490.54 and Definition 1490.59. Together they place S, R, B before the complete-metric and separable-metric handoffs whenever a dense Cauchy-basis read is consumed.

Finally, Definition 1490.60 excludes selected dense points, choice schedules, quotient representatives, and host presentation data. Hence an effective presentation read is exactly the displayed handoff through D, S, R, B over M. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceEffectivePresentationHandoff`

Theorem 1490.65 (PolishSpace complete-separable route)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. The* $\mathsf{CompleteMetric}^{\uparrow}$ *row K and the* $\mathsf{SeparableMetric}^{\uparrow}$ *row D are consumed over the same* $\mathsf{Met}^{\uparrow}$ *row M and are transported together through H. No complete-separable consumer may replace this common metric source by an independent completion carrier, alternate dense schedule, or host topology.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceCompleteSeparableRoute`

Proof. Start with Definition 1490.55. The rows K and D appear in the same carrier tuple and both are attached to the common metric row M.

Apply Definition 1490.50. It identifies M as the common source for the complete-metric and separable-metric faces, so neither K nor D can be read as a detached host fact.

Use Definition 1490.57. Completion-facing and separability-facing requests pass through the same M, S, R, A, K, D route, with the stream and regular-rational rows aligning the two faces before either is consumed.

Transport is componentwise by the row H. The NameCert obligations of Definition 1490.2 and the exhaustion theorem Definition 1490.61 keep K, D on the same route and leave no coordinate for an independent completion, alternate dense schedule, or host topology. □

Theorem 1490.66 (PolishSpace Real-RegSeqRat nonescape)**.** *Let $W = (M, K, D, S, R, B, A, H, C, P, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *root Cauchy-basis carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing read through the PolishSpace packet factors through the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger B. It exports no selected dense point, quotient completion equality, host topology, countable-choice schedule, second metric carrier, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted root carrier. The only Real-facing regular-Cauchy route available to the packet is the displayed path through S, R, B over the common metric row M and alignment row A.

Use Definition 1490.52. It places the Real-facing dependency after StreamName windows and regular-rational readback, rather than before them.

Apply Definition 1490.59 and Definition 1490.63. These theorems force dense Cauchy readback and terminal Real seal reads through S, R, B, A before completion or separability faces are consumed.

Finally invoke Definition 1490.60. Since the carrier has no coordinate for selected dense points, quotient completion equality, host topology, countable choice, a second metric carrier, or ambient Real completeness, the Real-facing route cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRealRegseqratNonescape`

Definition 1490.67 (Polish-space completion-density unblock)**.** *For an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier $P = (M, K, D, S, R, W, H, C, G, N)$, the completion-density unblock surface is the finite route that exposes the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K, and the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D only through the shared* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, observation ledger W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and local* NameCert *row N. A downstream consumer may read complete and dense data from this surface, but it receives no split metric carrier, selected dense point, quotient completion equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionDensityUnblock`

Theorem 1490.68 (Polish-space completion-density obligation)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every completion-density root request first fixes the common* $\mathsf{Met}^{\uparrow}$ *row M, then reads the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K and the* $\mathsf{SeparableMetric}^{\uparrow}$ *schedule D through the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and observation-ledger rows S, R, W. The supporting rows H, C, G, N transport, replay, source, and name only that displayed packet.*

Proof. Start from Definition 1490.67. The metric coordinate M is the single source for both the complete row K and the dense row D.

Read a completion-density request through S, R, W. The stream row supplies the finite dense-name and Cauchy-window positions, the regular rational row supplies the observable windows, and the ledger records that both kinds of request use the same route.

It remains to transport and name the packet. Componentwise hsame transport in H keeps M, K, D, S, R, W together, while C, G, N replay, source, and name the same finite data. No row remains from which a consumer could extract a split metric carrier, selected dense point, quotient completion equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompletionDensityObligation`

Theorem 1490.69 (Polish-space countable dense readback)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. Every countable dense readback of P is the finite route*

$$M \longmapsto D \longmapsto (S, R, W) \longmapsto (H, C, G, N),$$

with any completion-facing use of K read over the same M, S, R, W surface. Thus the dense readback is a displayed StreamName and RegSeqRat schedule, not a selected dense subset, host enumeration, quotient point equality, or countable-choice row.

Proof. Project the completion-density unblock surface. The dense schedule is the D row, and its only observation route is through the stream windows S, regular rational readback R, and ledger W.

Use Definition 1490.68. The obligation keeps D over M and forces completion-facing consumers of K to share the same S, R, W readback surface.

Finally attach H, C, G, N. These rows do not add a second dense source; they only transport, replay, source, and name the displayed route. Hence the countable dense readback remains finite and schedule-based. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCountableDenseReadback`

Theorem 1490.70 (Polish-space complete-metric separable synthesis)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier. The complete-metric and separable-metric data synthesize into one complete-separable root packet exactly when both K and D are consumed over the common* $\mathsf{Met}^{\uparrow}$ *row M and through the shared* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows S, R, W, H, C, G, N. The synthesis exports no independent completion coordinate, selected dense sequence, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the accepted carrier and apply Definition 1490.68. The complete row K and the separable row D are both read over M and through the same S, R, W observation surface.

Next use Definition 1490.69. The dense side has already been reduced to finite StreamName and RegSeqRat readback, and the completion-facing use of K cannot bypass that surface.

The remaining rows H, C, G, N assemble the synthesis. They provide componentwise transport, displayed replay, provenance, and local naming for the single packet, while no carrier coordinate names an independent completion, selected dense sequence, host equality, or ambient real-completeness principle. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteMetricSeparableSynthesis`

Theorem 1490.71 (Polish-space L10 sibling route)**.** *Every completion-density read of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier is routed through the sibling chapters chapter 57, chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38. The route uses the completion-density unblock surface of Definition 1490.67, the completion-density obligation of Definition 1490.68, the countable dense readback of Definition 1490.69, and the complete-metric separable synthesis of Definition 1490.70. It adds no selected dense point, quotient completion equality, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceL10SiblingRoute`

Proof. Start with the unblock definition. It exposes $M, K, D, S, R, W, H, C, G, N$ as the only rows available to a completion-density consumer, with M supplying the common metric source.

Apply the completion-density obligation and the countable dense readback. These place the complete-metric handoff and the separable schedule over the same StreamName and RegSeqRat windows, so the route must pass through the MetricSpace, CompleteMetric, SeparableMetric, StreamName, RegSeqRat, and Real sibling chapters named in the statement.

Finally apply the complete-metric separable synthesis. The synthesis repacks those sibling reads into the single $\mathsf{PolishSpace}^{\uparrow}$ surface through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since no carrier coordinate names a selected dense point, quotient completion equality, countable-choice schedule, or ambient Real-completeness theorem, the sibling route adds none of those objects. □

Definition 1490.72 (Polish-space complete-separable obligation carrier)**.** *A complete-separable obligation carrier for* $\mathsf{PolishSpace}^{\uparrow}$ *is a finite* BHist *packet*

$$P = (M, C, D, S, R, E, H, K, N)$$

whose rows display a $\mathsf{Met}^{\uparrow}$ *source* M*, a* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* C*, a* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-schedule row* D*,* $\mathsf{StreamName}^{\uparrow}$ *windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, a* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* K*, and local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *naming* N*. The carrier has no coordinate for host completion, quotient equality of completion points, a selected dense set, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1490.73 (Polish-space complete-separable NameCert obligations)**.** *For every accepted complete-separable obligation carrier*

$$P = (M, C, D, S, R, E, H, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *surface requires carrier admission, complete-metric handoff through* C*, separable dense-schedule exposure through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through* E*, componentwise transport* H*, displayed replay* K*, and local naming* N*. These rows bind the complete and separable routes over the same* $\mathsf{Met}^{\uparrow}$ *source* M*.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceCompleteSeparableNamecertObligations`

Proof. Project the carrier of Definition 1490.72. The metric row M is the single source for both the complete-metric handoff C and the separable dense-schedule row D.

Read observations through S, R, E. The StreamName row supplies finite Cauchy and dense-name windows, the regular rational row supplies the displayed readback, and the Real seal is downstream of those rows.

Finally attach H, K, N. Transport, replay, and local naming preserve the same finite packet, so the complete and separable obligations remain bound to M, C, D, S, R, E rather than to any host completion or selected dense set. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableNamecertObligations`

Theorem 1490.74 (Polish-space real completion readback nonescape)**.** *No* $\mathsf{PolishSpace}^{\uparrow}$ *consumer of the complete-separable obligation carrier*

$$P = (M, C, D, S, R, E, H, K, N)$$

may bypass $M, C, D, S, R, E, H, K, N$ *to obtain host completion, quotient equality of completion points, a selected dense set, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any real-facing completion read must first expose the metric row, complete-metric handoff, separable dense schedule, StreamName windows, RegSeqRat readback, and Real seal.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceRealCompletionReadbackNonescape`

Proof. Apply Definition 1490.73. It places every complete or separable read over the common metric source and through the displayed observation and real-readback rows.

The structural rows H, K, N cannot add a second completion route. They only transport, replay, and name the carrier, so any consumer that reaches a real-facing completion read remains inside M, C, D, S, R, E.

Since the carrier definition contains no host completion, quotient equality, selected dense set, countable-choice schedule, open-cover compactness row, or ambient real-completeness datum, none of those objects can be exported by a $\mathsf{PolishSpace}^{\uparrow}$ consumer of this packet. □

Theorem 1490.75 (Polish-space complete-separable obligation surface)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier*

$$P = (M, K, D, S, R, W, H, C, G, N),$$

the complete-separable obligation surface consists exactly of the common $\mathsf{Met}^{\uparrow}$ *source row* M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*, the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule* D*, the shared* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* S, R*, the observation ledger* W*, componentwise transport* H*, displayed replay* C*, provenance* G*, and local naming* N*.*

Proof. Project the carrier described in the PolishSpace spine. The rows M, K, D are the only complete and separable coordinates, and both are attached to the same metric source.

Use the NameCert-obligation theorem already attached to $\mathsf{PolishSpace}^{\uparrow}$. It places S, R, W as the finite readback route for complete and dense requests, so no consumer can split the complete metric from the separable schedule.

Finally attach H, C, G, N. Transport, replay, provenance, and local naming preserve the same packet, giving precisely the complete-separable surface carried by $P = (M, K, D, S, R, W, H, C, G, N)$. □

Theorem 1490.76 (Polish-space StreamName density scope)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier*

$$P = (M, K, D, S, R, W, H, C, G, N),$$

the density scope read through $\mathsf{StreamName}^{\uparrow}$ *is the route* D, S, R, W, H, C, G, N *over the common metric row* M*. It reads dense schedules as displayed stream windows, not as a host enumeration, selected dense subset, countable-choice schedule, or quotient point equality.*

Proof. Start from Definition 1490.75. It identifies D as the dense schedule and places it over the shared metric row M.

Read the density request through S, R, W. The StreamName row supplies finite windows, the RegSeqRat row supplies rational readback, and the observation ledger records the route used by the dense schedule.

The structural rows H, C, G, N only preserve and name this route. Since the accepted carrier has no host enumeration or selected dense-subset coordinate, the density scope remains the displayed StreamName route. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamNameDensityScope`

Theorem 1490.77 (Polish-space RegSeqRat real readback scope)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier*

$$P = (M, K, D, S, R, W, H, C, G, N),$$

every real-facing completion or density readback must pass through the $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *after the shared stream row* S *and before the structural rows* H, C, G, N*. The readback scope is therefore finite and carrier-local; it does not export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient completion equality, or a separate real-name source.*

Proof. Use Definition 1490.75 to place complete and separable data over M, K, D, S, R, W. The readback row R is part of that same surface, not an external real object.

Apply Definition 1490.76 to the dense side. The stream row S supplies the inspected windows, and R is the rational readback consumed from those windows.

The remaining rows H, C, G, N transport, replay, source, and name the packet. Therefore every real-facing read remains inside $P = (M, K, D, S, R, W, H, C, G, N)$ and cannot reach ambient completeness, selected limits, quotient completion equality, or a separate real-name source. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceRegSeqRatRealReadbackScope`

Theorem 1490.78 (Polish-space complete-separable window exhaustion)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier*

$$P = (M, K, D, S, R, W, H, C, G, N),$$

complete-separable reads are exhausted by the shared window M, K, D, S, R, W*. A consumer may preserve and name that window through* H, C, G, N*, but it receives no selected dense subset, host enumeration, quotient completion equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limit, or separate real-name source.*

Proof. Start from the complete-separable obligation surface of Definition 1490.75. It places the common metric source M, the complete handoff K, and the separable dense schedule D inside one carrier.

Use the StreamName density scope, Definition 1490.76. The dense side is read through S, R, W as finite stream windows with rational readback and an observation ledger.

The structural rows H, C, G, N preserve, replay, source, and name this same window. Since the carrier has no separate coordinate for the excluded objects, the complete-separable read is exhausted by M, K, D, S, R, W. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableWindowExhaustion`

Theorem 1490.79 (Polish-space effective represented source boundary)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the effective represented source boundary is the common metric row* M *together with the displayed StreamName and RegSeqRat route* S, R, W*. It is not a host enumeration, a selected dense subset, a quotient-completion equality, or a separate representation source outside the carrier.*

Proof. Project the carrier through Definition 1490.75. The complete row K and dense row D are both attached to the same metric source M.

Apply Definition 1490.76. Effective access to the dense side is carried by the StreamName row S, the RegSeqRat row R, and the observation ledger W.

The rows H, C, G, N do not create another represented source. They only preserve and name the displayed route, so any consumer that asks for host enumeration, a chosen dense subset, quotient equality, or an external representation leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceEffectiveRepresentedSourceBoundary`

Theorem 1490.80 (Polish-space real readback nonescape)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, any real-facing readback from the complete-separable surface must pass through* S, R, W *and then through* H, C, G, N*. It cannot escape to ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit, quotient completion equality, host enumeration, a selected dense subset, or a separate real-name source.*

Proof. Use the RegSeqRat real readback scope of Definition 1490.77. The readback row R is reached after the shared stream row S and before the structural rows.

Combine this with Definition 1490.78. The complete and separable data remain in the same window M, K, D, S, R, W, so real-facing access is not a separate source.

Finally read the structural rows H, C, G, N. They transport, replay, source, and name the finite carrier-local route, and the carrier has no coordinate for the excluded real or completion objects. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRealRegSeqRatReadbackNonescape`

Theorem 1490.81 (Polish-space compact-metric factorization)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, every compact-metric or completion-facing read of the packet factors through the common metric row* M*, the complete handoff* K*, the separable dense schedule* D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the observation ledger* W*, and the structural rows* H, C, G, N*. The factorization is the same finite route named by Definition 1490.75, Definition 1490.76, Definition 1490.77, and Definition 1490.78; it does not introduce a separate host Polish-space object, ambient completeness oracle, quotient-completion equality, or selected dense subset.*

Proof. Start with the complete-separable obligation surface of Definition 1490.75. It places the metric row M, complete handoff K, and dense schedule D inside the same accepted carrier, so a compact-metric consumer cannot split the metric source from the complete or separable rows.

Now apply Definition 1490.76 and Definition 1490.77. The dense and real-facing sides are both read through the finite window rows S, R, W, with R appearing after the stream windows and before the structural rows.

Finally use Definition 1490.78. The carrier-local window M, K, D, S, R, W is already exhaustive for complete-separable access, and H, C, G, N only transport, replay, source, and name that route. Therefore every compact-metric or completion-facing read factors through the displayed rows and has no separate host Polish-space object or ambient completeness oracle to consult. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompactMetricFactorization`

Theorem 1490.82 (Polish-space complete-separable obligation scope)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the complete-separable obligation scope is the common* $\mathsf{Met}^{\uparrow}$ *source row* M *together with the complete-metric handoff* K*, separable dense schedule* D*,* $\mathsf{StreamName}^{\uparrow}$ *windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, observation ledger* W*, componentwise transport* H*, displayed replay* C*, provenance* G*, and local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row* N*. The scope exports no selected dense subset, quotient-completion equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected completion limit, host enumeration, or external represented-space source.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableObligationScope`

Proof. Start with the complete-separable obligation surface of Definition 1490.75. It places $M, K, D, S, R, W, H, C, G$ inside one accepted packet and keeps the complete and separable rows over the same metric source.

Use the StreamName density scope of Definition 1490.76. The separable side is read through displayed windows S over the dense schedule D, so the consumer receives finite window data rather than a selected dense subset or host enumeration.

Apply the RegSeqRat readback scope of Definition 1490.77. The real-facing read passes through R after the StreamName row and before the structural rows, which excludes ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, and quotient-completion equality.

Finally use the compact-metric factorization of Definition 1490.81. It repacks compact-metric and completion-facing reads through the same carrier-local route, so no external represented-space source or detached completion object is exported. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceCompleteSeparableObligationScope`

Theorem 1490.83 (Polish-space pointwise density handoff)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N)$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *complete-separable packet whose complete metric, separable metric,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows are displayed through chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38. Every pointwise dense-read handoff through* P *must route through the countable-density row* D *before any completion-facing* $\mathsf{Real}^{\uparrow}$ *readback is consumed.*

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpacePointwiseDensityHandoff`

Proof. Start with the complete-separable obligation scope of Definition 1490.82. It places the complete-metric handoff K, separable dense schedule D, stream windows S, rational readback R, and observation ledger W inside the same accepted carrier.

Apply dense-window induction from Definition 1490.84. A pointwise dense-read handoff advances by reading the displayed countable-density row D through $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback, so the handoff cannot be supplied by a host enumeration or selected dense subset.

Use the observation lock of Definition 1490.85. It forces every combined complete-metric and separable read to inspect the shared ledger W along the same S, R route before completion-facing data are consumed.

The sibling chapters chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38 therefore appear only through the displayed rows K, D, S, R, W. Since no coordinate of P stores a separate completion source, a completion-facing $\mathsf{Real}^{\uparrow}$ readback may be consumed only after the countable-density row has been read. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparablePointwiseDensityHandoff`

Theorem 1490.84 (Polish-space complete-separable dense-window induction)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, complete-separable access supports induction over displayed dense windows. The base window reads the shared* $\mathsf{Met}^{\uparrow}$ *row* M *together with the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*. Each successor window extends the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule* D *through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, with* W *recording the inspected window. The structural rows* H, C, G, N *preserve the same packet at every step. Thus any accepted induction over the dense-window route remains inside* $M, K, D, S, R, W, H, C, G, N$ *and does not select a dense subset, import a host enumeration, identify a quotient completion, invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness, choose a limit, or introduce a separate representation source.*

Proof. Begin with the complete-separable obligation surface of Definition 1490.75. It names M as the shared metric source and K as the complete handoff, so the base window is already internal to the accepted carrier.

For the successor step, use the StreamName density scope and the RegSeqRat real readback scope. Definition 1490.76 reads the dense schedule through displayed stream windows, and Definition 1490.77 places rational readback after the stream row and before the structural rows.

Finally apply complete-separable window exhaustion from Definition 1490.78. The rows H, C, G, N only transport, replay, source, and name the same window route. Since no carrier coordinate supplies the excluded host objects, dense-window induction cannot escape the displayed packet. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableDenseWindowInduction`

Theorem 1490.85 (Polish-space complete-separable observation lock)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, any consumer that combines the complete-metric handoff* K *with the separable dense schedule* D *must first read the shared observation ledger* W *through the same* $\mathsf{StreamName}^{\uparrow}$ *row* S *and* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*. Hence complete and separable reads cannot split into two host sources, selected dense subsets, quotient-completion equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or external represented-space data.*

Proof. Start with the complete-separable obligation surface of Definition 1490.75. It places K and D over the same metric source M and inside the same carrier $P = (M, K, D, S, R, W, H, C, G, N)$.

Use the StreamName density scope and RegSeqRat real readback scope. Definition 1490.76 forces dense access through S, and Definition 1490.77 places R after the stream row and before the structural rows.

Apply complete-separable window exhaustion and compact-metric factorization, Definition 1490.78 and Definition 1490.81. They exhaust complete and separable reads through the displayed window M, K, D, S, R, W and then preserve that route through H, C, G, N.

Therefore a consumer that combines K and D must read the shared ledger W along the same S, R route. Since the accepted carrier has no coordinate for two host sources, selected dense subsets, quotient-completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or external represented-space data, those reads cannot split away from the observation ledger. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCompleteSeparableObservationLock`

Theorem 1490.86 (Polish-space complete-separable obligation closure)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the metric source* M*,* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*,* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule* D*,* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, observation ledger* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row* N *form the complete obligation surface for* $\mathsf{PolishSpace}^{\uparrow}$*. The closure is finite and carrier-local: it exports no selected dense subset, quotient-completion equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected completion limit, host enumeration, or external represented-space source.*

Proof. Start with the complete-separable obligation surface of Definition 1490.75. It identifies $M, K, D, S, R, W, H, C, G, N$ as the available rows of the accepted packet.

Use the StreamName density scope and RegSeqRat readback scope, Definition 1490.76 and Definition 1490.77. These place the dense schedule and real-facing readback inside the displayed window route rather than in a selected dense subset or ambient real object.

Apply window exhaustion, effective represented source boundary, and real-readback nonescape: Definition 1490.78, Definition 1490.79, and Definition 1490.80. Together they rule out external represented sources, host enumerations, selected limits, quotient-completion equality, and ambient completeness.

Finally read Definition 1490.82. It packages the same rows as the complete-separable scope, so the listed packet is the complete obligation surface and no further coordinate is available to a $\mathsf{PolishSpace}^{\uparrow}$ consumer. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishspaceCompleteSeparableObligationClosure`

Theorem 1490.87 (Polish-space public complete-separable real readback)**.** *Let* $P = (M, K, D, S, R, W, H, C, G, N$ *be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Every public read of* P *offered to an L10-adjacent downstream consumer must expose the common* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* K*,* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule* D*,* $\mathsf{StreamName}^{\uparrow}$ *window row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{Real}^{\uparrow}$*-facing seal route, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row* N *before the consumer may use the packet as a complete-separable real readback.*

The route is public only as the displayed finite surface over $M, K, D, S, R, W, H, C, G, N$*. It does not assert classical descriptive set theory, a choice-selected dense sequence, host enumeration of points, quotient completion equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier through Definition 1490.1. The rows $M, K, D, S, R, W, H, C, G, N$ are the accepted packet, so a public read cannot detach the complete handoff, dense schedule, stream windows, rational readback, transport, replay, provenance, or local name from the carrier.

Use the complete-separable obligation surface of Definition 1490.75. It places K and D over the same metric row M and keeps S, R, W, H, C, G, N visible as the common route.

Next apply Definition 1490.76, Definition 1490.77, and Definition 1490.78. These theorems force dense access through $\mathsf{StreamName}^{\uparrow}$, force real-facing readback through $\mathsf{RegSeqRat}^{\uparrow}$, and exhaust the complete-separable window before any downstream consumer can read it.

Finally read the sibling anchors chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38 only through the displayed rows. Since the carrier has no coordinate for the excluded choice, quotient, selected-limit, or ambient-completeness principles, the public route is exactly the stated finite readback surface. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpacePublicCompleteSeparableRealReadback`

Theorem 1490.88 (Polish-space public finite-window boundary)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Every public read of P offered to an L10 consumer is exhausted by the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *row K,* $\mathsf{SeparableMetric}^{\uparrow}$ *row D,* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, terminal* $\mathsf{Real}^{\uparrow}$ *readback through W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and the local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row N over the metric source M.*

The boundary cannot introduce a choice-selected dense sequence, quotient completion equality, open-cover compactness, selected limits, a host enumeration of points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness outside the displayed finite packet.*

Proof. Start with Definition 1490.87. It already identifies the public readback surface as $M, K, D, S, R, W, H, C, G, N$ and refuses the choice, quotient, selected-limit, and ambient-completeness coordinates.

Use the sibling anchors named there. The complete-metric row is read only through chapter 128, the separable row only through chapter 1458, the stream and regular-rational rows only through chapter 126 and chapter 301, and the real-facing endpoint only through chapter 38.

Finally apply Definition 1490.75 and Definition 1490.78. They keep the complete and separable sides over the same metric source and exhaust the finite window before any consumer reads it. Since no carrier coordinate stores the refused objects, the public boundary is exactly the displayed route. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpacePublicFiniteWindowBoundary`

Theorem 1490.89 (Polish-space public readback export)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. A public export of P to L10 consumers is exactly the finite route through the* $\mathsf{Met}^{\uparrow}$ *source M,* $\mathsf{CompleteMetric}^{\uparrow}$ *row K,* $\mathsf{SeparableMetric}^{\uparrow}$ *row D,* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, terminal* $\mathsf{Real}^{\uparrow}$ *seal W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and local* $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ *row N.*

The export refuses choice-selected dense sequences, quotient completion equality, selected limits, host enumeration of points, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness outside the displayed packet.*

Proof. Start with Definition 1490.87. It identifies the complete-separable readback surface as the displayed rows $M, K, D, S, R, W, H, C, G, N$ and keeps the real-facing endpoint inside the packet.

Apply Definition 1490.88. It says that every L10-facing public read is exhausted by the same finite window through complete metric, separable metric, stream, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Now compose these two boundaries with the sibling anchors already named in the file: chapter 57, chapter 128, chapter 1458, chapter 126, chapter 301, and chapter 38. Since the carrier has no coordinate for the refused choice, quotient, selected-limit, host-enumeration, compactness, or ambient-completeness principles, the public export is exactly the displayed L10 readback route. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpacePublicReadbackExport`

Theorem 1490.90 (Polish-space public readback uniqueness)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packet. Suppose two public exports of P expose the same* $\mathsf{CompleteMetric}^{\uparrow}$ *row K,* $\mathsf{SeparableMetric}^{\uparrow}$ *row D,* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and terminal* $\mathsf{Real}^{\uparrow}$ *route W, and suppose their displayed transport rows identify the corresponding components by componentwise* hsame*.*

Then the two exports agree as finite L10 readback routes over $M, K, D, S, R, W, H, C, G, N$. This uniqueness is only the displayed packet uniqueness: it does not select a dense sequence, identify quotient completions, choose limits, or claim ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the public readback export of Definition 1490.89. It says that any public export of P is exactly the finite route through $M, K, D, S, R, W, H, C, G, N$ and has no hidden choice, quotient, selected-limit, host-enumeration, compactness, or ambient-completeness coordinate.

Use the finite-window boundary of Definition 1490.88. Since both exports expose the same complete, separable, stream, regular-rational, and real rows, the L10-facing window of each export is exhausted by the same displayed components.

Finally apply the readback theorem of Definition 1490.87. Componentwise hsame on the exposed rows transports the same complete-separable real readback route through H, while C, G, N only replay, source, and name that route. Thus the two public exports agree as finite readback routes, without adding any of the refused external principles. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpacePublicReadbackUniqueness`

Theorem 1490.91 (Polish-space complete-separable public interface)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* PolishSpace$^{\uparrow}$ *packet. A public L10-facing read of P is the conjunction of the metric source row M, the* CompleteMetric$^{\uparrow}$ *row K, the* SeparableMetric$^{\uparrow}$ *dense schedule D, the* StreamName$^{\uparrow}$ *finite-window service S, the* RegSeqRat$^{\uparrow}$ *readback R, the terminal* Real$^{\uparrow}$ *seal W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance G, and the local* NameCert$_{\text{PolishSpace}^{\uparrow}}$ *row N.*

This interface refuses choice-selected dense sequences, quotient completion equality, host enumeration of points, selected limits, open-cover compactness, and ambient Real$^{\uparrow}$ *completeness outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceCompleteSeparableRealReadbackPublicInterface`

Proof. Start with Definition 1490.87. It identifies the accepted packet rows $M, K, D, S, R, W, H, C, G, N$ as the complete-separable real readback surface.

Apply the finite-window boundary of Definition 1490.88. It keeps every L10-facing read inside the complete metric, separable metric, stream, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Use the public readback export theorem Definition 1490.89. The export theorem supplies the metric source row together with the same complete, separable, window, readback, seal, and structural rows, so the interface is the displayed conjunction rather than a new abstract equivalence schema.

Finally apply public readback uniqueness, Definition 1490.90. Since any two public exports with the same displayed rows agree only as finite readback routes, the refused choice, quotient, host-enumeration, selected-limit, compactness, and ambient-completeness principles cannot enter the interface. □

Theorem 1490.92 (Polish-space public complete-separable real readback frontier)**.** *Let $P = (M, K, D, S, R, W, H, C, G, N)$ be an accepted* PolishSpace$^{\uparrow}$ *packet whose public complete-separable readback has been accepted through Definition 1490.91. A Real-facing point can be exposed to a downstream consumer only through the displayed* CompleteMetric$^{\uparrow}$ *row K,* SeparableMetric$^{\uparrow}$ *dense-window row D,* StreamName$^{\uparrow}$ *window row S,* RegSeqRat$^{\uparrow}$ *readback row R,* DyadicRatCore$^{\uparrow}$ *tolerance discipline inherited by the regular readback route, and the terminal* Real$^{\uparrow}$ *seal W, together with H, C, G, N.*

The frontier exports no open-cover compactness principle, selected limit, quotient real equality, choice-selected dense sequence, host enumeration of points, or ambient Real$^{\uparrow}$ *completeness outside that finite route.*

Proof. First read the four public-interface blocks already present in this child file. The complete-separable real readback theorem, the finite-window boundary, the public readback export, and the complete-separable public interface keep every public read inside the same packet rows $M, K, D, S, R, W, H, C, G, N$

Next inspect the sibling sources. The complete metric row is grounded in chapter 128, the complete-separable surface is read through chapter 1735, the window route through chapter 126, the regular readback through chapter 301, and the terminal seal through chapter 38. The dyadic tolerance discipline enters only through the regular-rational route and its chapter 291 tolerance ledger.

Finally apply public readback uniqueness. Since any accepted public export agrees only as the displayed finite route, a downstream consumer has no coordinate from which to read open-cover compactness, selected limits, quotient real equality, a choice-selected dense sequence, host enumeration, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The Real-facing point is therefore exposed exactly at the stated frontier. □

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpacePublicCompleteSeparableRealReadbackFrontier`

Theorem 1490.93 (Polish-space complete-separable obligation frontier)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the obligation frontier for complete-separable reads is exactly the complete-metric row* K*, the separable-metric row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal route read through* W*, and the structural rows* H, C, G, N *over the common metric row* M*. The frontier is the obligation surface of Definition 1490.75, Definition 1490.76, and Definition 1490.77; it has no coordinate for a selected dense set, quotient completion equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the complete-separable obligation surface. The rows $M, K, D, S, R, W, H, C, G, N$ are the whole displayed packet available to the complete and separable consumers.

Apply the StreamName density scope and the RegSeqRat real readback scope. They place finite stream windows and rational readback between the common metric source and every Real-facing completion or density read.

The structural rows only preserve and name this same packet. Since the carrier has no coordinate for the excluded host objects, the complete-separable obligation frontier is exactly the displayed surface. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceCompleteSeparableObligationFrontier`

Theorem 1490.94 (Polish-space stream realization obligation)**.** *Every* $\mathsf{PolishSpace}^{\uparrow}$ *obligation read that realizes a separable dense schedule must expose the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and the observation ledger* W *before either the* $\mathsf{CompleteMetric}^{\uparrow}$ *row* K *or the* $\mathsf{SeparableMetric}^{\uparrow}$ *row* D *is consumed as public data. The realization is the carrier-local route named by Definition 1490.59, Definition 1490.63, and Definition 1490.76, not a host enumeration, selected dense sequence, or countable-choice schedule.*

Proof. Start with the root Cauchy-basis carrier. Its dense reads are displayed by the shared rows $M, S, R, B, A, H, C, P, N$ before the complete and separable faces are consumed.

Use the countable dense Cauchy readback and regular-Cauchy root window. They force dense realization through scheduled windows and regular rational readback, not through an unscheduled host enumeration.

Finally apply the StreamName density scope for the complete-separable obligation carrier. The same stream-readback discipline survives in the $P = (M, K, D, S, R, W, H, C, G, N)$ surface, so no selected dense sequence or countable-choice schedule is exported. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceStreamRealizationObligation`

Theorem 1490.95 (Polish-space dense-schedule transport obligation)**.** *Any dense-schedule read transported across accepted* $\mathsf{PolishSpace}^{\uparrow}$ *packets must preserve the common metric row* M*, complete row* K*, separable schedule* D*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*,* $\mathsf{Real}^{\uparrow}$ *readback row* W*, and structural rows* H, C, G, N*. The transported read is therefore a row-local dense schedule over the same complete-separable packet, not a selected dense sequence, host enumeration, or countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.PolishSpaceUp.PolishSpaceDenseScheduleTransportObligation`

Proof. Project the accepted carrier $P = (M, K, D, S, R, W, H, C, G, N)$. The dense-schedule read is attached to the separable row D over the common metric source M, while completion and real-facing consumers see it only together with K, S, R, W and the structural rows.

Apply the stream realization obligation Definition 1490.94. It places every dense realization through S, R, W before complete or separable data is consumed as public data.

Now cite the real completion consumer boundary Definition 1490.96. It keeps Real-facing reads on the same $M, K, D, S, R, W, H, C, G, N$ surface and prevents a separate real-name source or selected limit from entering the packet.

Repack the transported dense read with H, C, G, N. Since the carrier contains no coordinate for selected dense sequences, host enumeration, or countable choice, transport preserves only the displayed dense-schedule surface. □

Theorem 1490.96 (Polish-space real completion consumer boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing completion consumer of an accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier must pass through the complete-metric row* K*, the separable-metric row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the real-readback ledger* W*, and the structural rows* H, C, G, N *over the common metric row* M*. The boundary is read through Definition 1490.80, Definition 1490.66, and Definition 1490.75; it exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limit, quotient completion equality, host topology, or separate real-name source.*

Proof. Project the accepted Polish carrier and the complete-separable obligation surface. The only Real-facing rows available to a completion consumer are the StreamName, RegSeqRat, and observation rows attached to the common metric source.

Apply the two real-readback nonescape theorems. They force every Real-facing read through the regular-rational window and rule out an alternate real-name or ambient-completeness source.

The complete and separable rows remain tied to the same metric packet by transport, replay, provenance, and local naming. Hence a real completion consumer receives only the displayed rows named in the statement. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceRealCompletionConsumerBoundary`

Theorem 1490.97 (Polish-space obligation-closure witness)**.** *For every accepted* $\mathsf{PolishSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, W, H, C, G, N)$*, the displayed complete metric row* K*, countable dense schedule* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{Real}^{\uparrow}$ *seal read through* W*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* G*, and local* NameCert *row* N *supply the chapter's* obligation *surface over the common metric row* M*. This closure statement is a paper-side obligation witness; it does not change the independent formalstatus axis or claim theorem-checked Lean coverage.*

Proof. Project the complete-separable obligation frontier, stream realization obligation, and real completion consumer boundary above. Together they display the complete row, dense schedule, finite stream windows, regular-rational readback, Real-facing seal route, transport, replay, provenance, and local naming rows.

The rows are read over the same metric source M, so a complete or separable consumer cannot split the completion surface from the dense schedule or from the finite readback route.

Thus the chapter has a finite obligation-closure witness for $\mathsf{PolishSpace}^{\uparrow}$. The witness records the paper-side closure surface only; formalstatus remains governed by the separate Lean marker and closurestatus discipline. □

Lean4 ✓ `BEDC.Derived.PolishspaceUp.PolishSpaceObligationClosureWitness`

Closure status for $\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}$ •

Constructive story: A $\mathsf{PolishSpace}^{\uparrow}$ obligation packet is generated from a common metric row, complete and separable metric rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1490.93, Definition 1490.94, Definition 1490.96, and Definition 1490.97 bind the complete and separable $\mathsf{PolishSpace}^{\uparrow}$ route through $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PolishSpaceUp.PolishSpacePublicCompleteSeparableRealReadbackFrontier`

Bridge status: none

Not claimed: This chapter does not close selected dense sequences, host enumeration, countable-choice schedules, quotient completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Bridge checking requires a standard bridge theorem for the mature complete-separable $\mathsf{PolishSpace}^{\uparrow}$ route.

Chapter 1491

A Concrete Naming Certificate for $\mathsf{RationalEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive rational constants as real names, the $\mathsf{RationalEmbedding}^{\uparrow}$ packet records the standard constant rational route from a $\mathsf{Rat}^{\uparrow}$ row into a $\mathsf{Real}^{\uparrow}$ seal without turning host rational equality, quotient streams, selected limits, trichotomy, or ambient completeness into primitive coordinates. It sits between the rational source surface of chapter 37, the constant finite-window surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal boundary of chapter 38, and the dyadic constant-name route of chapter 467.

Definition 1491.1 (RationalEmbedding carrier surface)**.** *A* $\mathsf{RationalEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (q, S, R, D, E, H, C, P, N)$$

with the following displayed coordinates:

1. *q is the* $\mathsf{Rat}^{\uparrow}$ *source row naming the accepted rational point;*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule whose admitted windows all read the same rational source row q;*
3. *R is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the constant schedule S;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation row used to compare the stationary rational reads at the displayed finite precisions;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after q, S, R, D have been exposed;*
6. *H stores componentwise* hsame *transports for the rational source, constant schedule, stationary readback, dyadic tolerance row, real seal, package provenance, and local naming row;*
7. *C stores the displayed* Cont *routes from q to S, from S to R, from R to D, and from D to E;*
8. *P is the* Pkg *provenance over* $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalEmbedding}^{\uparrow}}$ *row.*

The carrier is a finite certificate for rational constants as real names. It contains no quotient stream, no host real equality row, no selected limit, no trichotomy principle, no host rational equality oracle, and no ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.RationalEmbeddingUp.RationalEmbeddingTasteGate single carrier alignment`

Theorem 1491.2 (RationalEmbedding constant sequence route)**.** *Every accepted* $\mathsf{RationalEmbedding}^{\uparrow}$ *packet routes its* $\mathsf{Rat}^{\uparrow}$ *source row* q *through the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *and the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *before any* $\mathsf{Real}^{\uparrow}$ *read is available. Thus a consumer of the real seal first sees the rational source, then the constant finite-window schedule, then the stationary regular-rational row, and only then the dyadic observation row and seal.*

Lean4 ✓ `BEDC.Derived.RationalEmbeddingUp.RationalEmbeddingCarrier constant sequence route`

Proof. Project the coordinates of Definition 1491.1. The continuation row C displays the route from q to S and the route from S to R, while H records that these coordinates are transported componentwise with the same package provenance.

The constant schedule S has no independent stream seed: each admitted window is read from q. The stationary row R is therefore the regular-rational readback of that constant schedule, not a separate rational sequence.

Only after R is displayed can the packet expose the dyadic observation row D and then the real seal E. Hence every real-facing consumer reads q, S, R in that order before reaching any $\mathsf{Real}^{\uparrow}$ coordinate. □

Lean4 ✓ `BEDC.Derived.RationalEmbeddingUp.RationalEmbeddingConstantSequenceRoute`

Theorem 1491.3 (RationalEmbedding RealUp seal boundary)**.** *For every accepted* $\mathsf{RationalEmbedding}^{\uparrow}$ *carrier* $K = (q, S, R, D, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the displayed dependency chain*

$$\mathsf{Rat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{NameCert}_{\mathsf{RationalEmbedding}^{\uparrow}}.$$

The seal exports only the local rows q, S, R, D, H, C, P, N *as its boundary data. It does not export a quotient stream, host real equality, a selected limit, trichotomy, host rational equality as an oracle, or ambient completeness for arbitrary Cauchy names.*

Proof. Start with a read of the real seal E. By the carrier definition, E is reachable only through the route entries in C whose predecessors are D, R, S, and q, and the same predecessors are named in the package row P and local certificate row N.

Apply Definition 1491.2. It identifies the route to R as the constant rational schedule read from q through S. The dyadic row D then supplies only the finite tolerance and observation data attached to that stationary readback.

The remaining rows H, C, P, N are transport, continuation, provenance, and local naming rows for the same finite route. Since the carrier contains no coordinate for a quotient stream, host real equality, selected limit, trichotomy, host rational equality oracle, or ambient completeness, the $\mathsf{Real}^{\uparrow}$ seal boundary is exactly the displayed finite dependency chain. □

Lean4 ✓ `BEDC.Derived.RationalEmbeddingUp.RationalEmbeddingRealSealBoundary`

Theorem 1491.4 (RationalEmbedding stationary readback exactness)**.** *For every accepted* $\mathsf{RationalEmbedding}^{\uparrow}$ *carrier* $K = (q, S, R, D, E, H, C, P, N)$*, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is exactly the finite readback of the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *sourced by the* $\mathsf{Rat}^{\uparrow}$ *row* q*. The later* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *may inspect only this stationary readback before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed.*

Proof. Project the carrier surface of Definition 1491.1. The rows q, S, R, D, E appear in dependency order, and the continuation row C displays the route from the rational source to the constant schedule, then from the schedule to the stationary readback.

Apply Definition 1491.2. It identifies S as the constant finite-window schedule whose admitted windows all read q, and identifies R as the stationary regular-rational readback of that schedule.

Apply the seal-boundary theorem Definition 1491.3. It confines the real-facing route to q, S, R, D, H, C, P, N and gives D no coordinate other than the dyadic tolerance and observation row attached to the stationary readback.

Therefore the finite readback seen by D is exactly the stationary row R generated from S and q. No quotient stream, selected limit, host real equality, or ambient completeness coordinate can alter that readback before the seal is reached. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalEmbedding}^{\uparrow}$ packet is generated from a RatUp source row, constant StreamName schedule, stationary RegSeqRat readback, DyadicRatCore tolerance and observation row, RealUp seal, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1491.1, Definition 1491.2, and Definition 1491.3 seed the finite rational-constant embedding route from $\mathsf{Rat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient-stream equality, selected host limits, host rational equality as an oracle, trichotomy, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, constant StreamName routing, stationary RegSeqRat readback, DyadicRatCore observation exactness, RealUp seal factorization, and non-escape.

Chapter 1492

A Concrete Naming Certificate for $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$

Following Bolzano, Weierstrass, and Bishop–Bridges, the $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ packet records the constructive finite boundary from a bounded real sequence to a $\mathsf{Real}^{\uparrow}$ cluster seal. It consumes the bounded source surface of chapter 1415, the subsequence compactness route of chapter 1254, the selector handoff of chapter 1408, the monotone subsequence ledger of chapter 1405, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name surface of chapter 38. The packet is a boundary certificate for the bounded-sequence cluster route; it is not open-cover compactness, a host completeness theorem, quotient equality of streams, a least cluster selector, or a choice-selected arbitrary subsequence.

Definition 1492.1 (Bolzano–Weierstrass boundary carrier)**.** *A* $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, Q, D, BW, L, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded source row, read as a* $\mathsf{BoundedRealSequence}^{\uparrow}$ *packet and carrying only inspected sequence indices, finite bounds, and the displayed source route needed by downstream consumers.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family exposed for the inspected source indices and selected subsequence indices.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the displayed windows at the precisions requested by Q and by the seal row.*
5. *BW is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet that routes the bounded real sequence through finite interval nets and regular subsequence readback.*
6. *L is the* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *selector packet that keeps the finite selector surface visible before any real-facing seal is consumed.*
7. *T is the* $\mathsf{MonotoneSubsequence}^{\uparrow}$ *route naming the monotone selected subsequence windows used by L and BW.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal reached only after S, W, Q, D, BW, L, T have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, stream windows, regular readback, dyadic tolerances, Bolzano–Weierstrass route, selector route, monotone subsequence route, Real seal, continuation rows, provenance rows, and local naming rows.*

10. *C records displayed* Cont *replay from the bounded source to stream windows, from stream windows to dyadic and regular readback, from selector and monotone subsequence rows to the Bolzano–Weierstrass route, and from that route to the Real cluster seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, ambient completeness, quotient-stream equality, a least cluster selector, a hidden tail selector, or choice over arbitrary subsequences.

Theorem 1492.2 (Bolzano–Weierstrass boundary NameCert obligations)**.** *For every accepted* $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ *carrier* $B = (S, W, Q, D, BW, L, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassBo}}$ *surface consists exactly of carrier admission for the twelve displayed rows, bounded-source admission through S, finite-window coverage through W, regular readback through Q, dyadic tolerance coverage through D, Bolzano–Weierstrass route admission through BW, selector-route admission through L, monotone-subsequence admission through T, Real cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1492.1. The bounded source row S is read through chapter 1415, so the source side is already a finite sequence-window and bound certificate before the cluster route is inspected.

Read the analytic route next. The packet BW supplies the bounded-sequence subsequence compactness route of chapter 1254, while L and T expose the selector and monotone-subsequence rows consumed by that route.

Now inspect the finite completion-facing rows. The windows W, regular readback Q, and dyadic tolerance ledger D are precisely the surfaces read through the StreamName, RegSeqRat, and DyadicRatCore chapters before the $\mathsf{Real}^{\uparrow}$ seal E is available.

Finally the structural rows H, C, P, N transport, replay, source, and name only the displayed source, route, selector, window, tolerance, readback, and seal rows. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 1492.3 (Bolzano–Weierstrass boundary source handoff)**.** *Every accepted* $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ *carrier hands a bounded real sequence source to the Bolzano–Weierstrass route only through*

$$S \longmapsto W \longmapsto Q, D \longmapsto L, T \longmapsto BW$$

together with H, C, P, N. Consequently the source consumed by the cluster route is the finite $\mathsf{BoundedRealSequence}^{\uparrow}$ *surface, not a host sequence object, an open-cover compactness principle, trichotomy of reals, quotient equality of streams, or a choice-selected arbitrary subsequence.*

Proof. Start with S. By chapter 1415, this row exposes bounded sequence data through inspected indices, finite stream windows, regular readback, Real-name sealing, and an interval-net bound row.

The carrier then reads W, Q, D. These rows make the inspected source and selected subsequence windows visible through the finite StreamName, RegSeqRat, and DyadicRatCore surfaces before the selector or cluster route can be consumed.

Pass through L and T. The selector packet and monotone subsequence packet display the selected-window route that the Bolzano–Weierstrass packet reads; the route is finite and cofinal on the inspected thresholds, not a choice-selected subsequence.

Only after these rows does BW receive the source. Transport, replay, provenance, and local naming preserve the same path, so the handoff cannot bypass the finite bounded-source certificate. □

Theorem 1492.4 (Bolzano–Weierstrass boundary Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *cluster seal read from an accepted* $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ *carrier factors through*

$$S, W, Q, D, L, T, BW, E, H, C, P, N.$$

The seal row E is reachable only after the bounded source, finite windows, regular readback, dyadic tolerance ledger, selector route, monotone-subsequence route, and Bolzano–Weierstrass route have been displayed. Thus a real-facing consumer receives a finite boundary certificate for the bounded-sequence cluster route and receives no open-cover compactness theorem, host completeness row, quotient equality of streams, least cluster selector, hidden tail selector, or arbitrary subsequence choice.

Proof. Begin with the handoff of Definition 1492.3. It places the bounded source, finite windows, regular readback, dyadic tolerances, selector route, monotone subsequence route, and Bolzano–Weierstrass packet before any terminal real-facing row.

Read the seal row through chapter 38. The row E is therefore a $\mathsf{Real}^{\uparrow}$ cluster seal for the displayed subsequence route, not a host limit, quotient representative, or separate completeness field.

The continuation and transport rows preserve the same factorization. Since the carrier has no coordinate for open-cover compactness, host completeness, quotient equality, least cluster selection, hidden tail selection, or arbitrary subsequence choice, no real-facing read can escape the listed finite rows. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ cluster-seal read factors through $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$, $\mathsf{MonotoneSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before reaching the $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: boundedrealsequence, bolzanoweierstrass, bolzanoweierstrassselector, monotonesubsequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded source row, finite windows, regular readback, dyadic tolerances, Bolzano–Weierstrass routing, selector routing, monotone subsequence routing, Real cluster sealing, transport, continuation, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}$ packet is generated from a bounded real sequence source, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, Bolzano-Weierstrass routing, selector routing, monotone-subsequence routing, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BolzanoWeierstrassBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1492.1, Definition 1492.2, Definition 1492.3, and Definition 1492.4 bind the finite boundary from bounded real sequences to Real cluster seals.

Formal verification: FormalStatus(NameCert$_{\text{BolzanoWeierstrassBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, host completeness, quotient equality of streams, least cluster selection, hidden tail selection, or arbitrary subsequence choice.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, bounded-source handoff, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.BolzanoWeierstrassBoundaryUp.

Chapter 1493

A Concrete Naming Certificate for $\mathsf{ExtremeValue}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{ExtremeValue}^{\uparrow}$ packet records the finite BEDC reading of the extreme-value theorem. Continuous real-valued maps on a compact metric source expose an attained-value packet only through finite nets, uniform-continuity moduli, a finite lower-bound fold, located supremum rows, $\mathsf{Real}^{\uparrow}$ seal rows, hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$. The packet consumes the compact metric source of chapter 129, the continuous-map surface of chapter 130, the compact-uniform-continuity route of chapter 388, the finite-net minimum fold of chapter 619, the located supremum boundary of chapter 1427, and the terminal real-name seal of chapter 38. It is an attainment certificate for inspected finite nets, not an open-cover compactness principle, a host maximum operator, quotient real equality, or a choice-selected point.

Definition 1493.1 (Extreme value carrier)**.** *An* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, F, U, M, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the admitted metric name, compact finite-net schedules, radius requests, and the displayed source certificate.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming a real-valued map from the source surface to* $\mathsf{Real}^{\uparrow}$ *output seals.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row that turns the compact metric source and continuous map into a displayed finite modulus ledger.*
4. *M is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *row computing the finite lower-bound candidate over the inspected net selected by U.*
5. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row recording the located upper boundary for the displayed value windows.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the attained value and for the real comparisons used by M and S.*
7. *H records componentwise* hsame *transport for the compact metric source, continuous map, uniform-continuity ledger, finite-net fold, located supremum row, real seals, continuation rows, provenance rows, and local naming rows.*

8. *C records displayed* Cont *replay from compact-metric radius requests to finite nets, from the map row to value windows, from uniform-continuity requests to the fold row, and from the fold and located-supremum rows to the terminal real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ExtremeValue}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for open-cover compactness, host-level maximum operators, quotient real equality, selected arbitrary points, countable choice, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueTasteGate single carrier alignment`

Theorem 1493.2 (Extreme value NameCert obligations)**.** *For every accepted* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier* $E = (X, F, U, M, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact metric source, continuous real-valued map, compact uniform-continuity ledger, finite-net minimum fold, located supremum boundary, terminal real seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueNameCertObligations`

Proof. Project the carrier of Definition 1493.1. The source row X is read through chapter 129, so every metric, radius, and compact-net request is already displayed before a value row is inspected.

Read the map and modulus rows next. The row F uses the continuous-map surface of chapter 130, and U uses the compact-uniform-continuity route of chapter 388 to bind value variation on the finite nets supplied by X.

Now inspect the real-valued boundary rows. The row M applies the finite-net fold of chapter 619, while S reads the located upper boundary through chapter 1427 and R supplies only the real-name seals admitted by chapter 38.

Finally the structural rows H, C, P, N transport, replay, source, and name the already displayed source, map, modulus, fold, supremum, and seal rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1493.3 (Extreme value finite-net attainment)**.** *Every accepted* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier factors any attained-value read through*

$$X \longmapsto F \longmapsto U \longmapsto M \longmapsto S \longmapsto R$$

together with H, C, P, N. Consequently the exported attainment statement is a finite-net and real-seal certificate over the compact metric source and continuous map, not a host maximum operator, an arbitrary selected point of the source, or an open-cover compactness argument.

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueCarrier finite net attainment`

Proof. Begin with the compact metric source. By chapter 129, X exposes the finite net schedule and the metric-radius data that any later value comparison must use.

Feed this source into the map row. The continuous-map packet of chapter 130 supplies only displayed value windows, and the compact-uniform-continuity packet of chapter 388 turns those windows into a finite modulus ledger on the same compact source.

The fold row M then searches the displayed net by the finite-net minimum fold of chapter 619. The located supremum row S supplies the opposing located boundary through chapter 1427, and the real-seal row R is read through chapter 38.

Transport and replay preserve the same factorization. Thus a consumer can see only the finite source net, the displayed map windows, the compact-uniform modulus, the finite fold, the located boundary, and the real seal; no separate host maximum or arbitrary point selector is available. □

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueFiniteNetAttainment`

Theorem 1493.4 (Extreme value Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *value seal exported by an accepted* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier is downstream of the compact metric source* X*, continuous map row* F*, compact uniform-continuity ledger* U*, finite-net minimum fold* M*, and located supremum row* S*. The seal row* R *therefore records only the boundary visible through finite-net attainment data and does not provide quotient real equality, detached* $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, or an independent real maximum principle.*

Proof. Start from the factorization in Definition 1493.3. It places the compact metric source, map windows, compact-uniform modulus, finite-net fold, and located supremum row before any real-facing seal is exported.

The upstream chapters fix the only admissible route: chapter 129 supplies X, chapter 130 supplies F, chapter 388 supplies U, chapter 619 supplies M, and chapter 1427 supplies S.

Read R through chapter 38. This makes the terminal row a $\mathsf{Real}^{\uparrow}$ seal for the displayed finite-net value and located boundary, not a quotient representative or a host completeness witness.

The continuation, transport, provenance, and local naming rows do not add new real equality or selection data. Since the carrier has no coordinate for detached $\mathsf{Real}^{\uparrow}$ completeness, countable choice, quotient equality, or a host maximum operator, every exported real seal remains inside the finite-net attainment boundary. □

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueRealSealBoundary`

Theorem 1493.5 (Extreme value Real seal factorization)**.** *Every Real-facing attained-value read from an accepted* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier factors through*

$$X \longmapsto F \longmapsto U \longmapsto M \longmapsto S \longmapsto R$$

together with H, C, P, N*. The source row* X *is supplied by* $\mathsf{CompactMetric}^{\uparrow}$*, the map row* F *by* $\mathsf{ContinuousMap}^{\uparrow}$*, the modulus row* U *by* $\mathsf{CompactUniformContinuity}^{\uparrow}$*, the finite fold row* M *by* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*, the located boundary row* S *by* $\mathsf{LocatedSupremum}^{\uparrow}$*, and the terminal seal row* R *by* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueRealSealFactorization`

Proof. Start with Definition 1493.3. It places every attained-value read on the route from compact metric source to continuous map, compact-uniform modulus, finite-net fold, located boundary, and terminal real seal.

Apply Definition 1493.4. The terminal row R is a $\mathsf{Real}^{\uparrow}$ seal only for the finite-net value and located boundary already exposed by X, F, U, M, S.

The sibling chapters supply the source rows named in the statement: compact metric data, continuous map windows, compact uniform-continuity ledgers, finite-net minimum folds, located supremum rows, and Real seals. The structural rows H, C, P, N transport, replay, source, and name that same chain; they do not add open-cover compactness, a host maximum operator, quotient equality, countable choice, or detached $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1493.6 (Extreme value compact-net maximum row)**.** *Every accepted* $\mathsf{ExtremeValue}^{\uparrow}$ *carrier exposes the finite-net candidate row* M *from* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *before the located supremum row* S *and terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *are consumed. Thus the maximum-facing read is certified by displayed finite net data over* $X, F, U, M, S, R, H, C, P, N$*, not by a host maximum selector, an open-cover compactness argument, quotient real equality, or countable choice.*

Lean4 ✓ `BEDC.Derived.ExtremeValueUp.ExtremeValueCompactNetMaximumRow`

Proof. Project the carrier of Definition 1493.1. The row M is part of the displayed packet and is positioned after the compact metric source, continuous map, and compact-uniform modulus rows.

Use finite-net attainment from Definition 1493.3. Any attained-value consumer must pass through the finite fold before it can read the located boundary or the terminal Real seal.

Finally apply the Real seal boundary of Definition 1493.4. The seal row records only the boundary already visible through the finite net and located supremum rows, so no open-cover compactness coordinate or choice-based maximum selector is available. □

Falsifiable prediction: Every $\mathsf{ExtremeValue}^{\uparrow}$ attained-value read factors through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before reaching the local $\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}$ row.

Independence witness: compactmetric, continuousmap, compactuniformcontinuity, finitenetminimumfold, locatedsupremum, real: $\mathsf{ExtremeValue}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact metric source, a continuous real-valued map, a compact-uniform modulus ledger, a finite-net minimum fold, a located supremum boundary, terminal real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}$.

Constructive story: An $\mathsf{ExtremeValue}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source, a $\mathsf{ContinuousMap}^{\uparrow}$ real-valued map, a $\mathsf{CompactUniformContinuity}^{\uparrow}$ modulus ledger, a $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ finite fold, a $\mathsf{LocatedSupremum}^{\uparrow}$ boundary row, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1493.1, Definition 1493.2, Definition 1493.3, Definition 1493.4, and Definition 1493.5 seed the constructive finite-net attainment reading of extreme values on compact metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtremeValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host maximum operators, quotient real equality, selected arbitrary points, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, finite-net attainment, Real seal boundary, and TasteGate field faithfulness for BEDC.Derived.ExtremeValueUp.

Chapter 1494

A Concrete Naming Certificate for $\mathsf{RepresentedSpace}^{\uparrow}$

Following Weihrauch-style represented spaces and Bishop–Bridges constructive analysis, the $\mathsf{RepresentedSpace}^{\uparrow}$ packet records the interface at which a computable name is allowed to present a real-facing object. It sits over the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The carrier below treats representation as a finite NameCert interface rather than as a quotient of names or a selected host representative.

Definition 1494.1 (RepresentedSpace carrier)**.** *A* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, \rho, X, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the name carrier: a finite source row whose observations are read through the* $\mathsf{StreamName}^{\uparrow}$ *interface.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule recording the inspected precision requests, stream positions, and replay windows.*
3. *ρ is the representation relation row; it relates the displayed stream observations to the target row only through the local finite schedule.*
4. *X is the target carrier row, with the real-facing case routed through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any consumer reads it.*
5. *T is classifier transport from the name row, schedule row, representation relation, and target row to a common finite comparison surface.*
6. *H records componentwise* hsame *transport for $S, W, \rho, X, T, C, P, N$.*
7. *C records displayed* Cont *replay from a precision request through the finite schedule, representation relation, target row, and classifier transport.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and the local NameCert row.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common finite surface. It has no coordinate for quotient names, selected representatives, host real equality, an ambient completeness row, or a choice principle.

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceTasteGate single carrier alignment`

Theorem 1494.2 (RepresentedSpace NameCert obligations)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed name carrier, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, representation relation, target carrier, classifier transport, continuation, provenance, and local name;*
2. *classifier transport to a common finite surface on which the name row, schedule row, representation row, and target row can be compared without adding host equality;*
3. *stability of the name, schedule, representation relation, target row, and classifier transport under the componentwise* hsame *rows recorded in* H*;*
4. *exactness of the representation relation row* ρ *as the only permitted route from stream observations to the target carrier;*
5. Cont *replay,* Pkg *provenance, and local NameCert rows for exactly the displayed packet.*

No NameCert obligation is stored outside those five rows.

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier namecert obligations`

Proof. Project the carrier of Definition 1494.1. The projection exposes the name carrier, finite stream schedule, representation relation, target carrier, transport rows, continuation row, provenance row, and local name.

Classifier transport is componentwise. The rows in H move the name, schedule, representation relation, target row, and transport row to the same finite comparison surface, while C replays only the displayed requests and windows.

Representation exactness is exhausted by ρ. A consumer can reach X from the stream observations only through the stored relation row, and the real-facing case must then pass through the regular rational, dyadic, and real seal rows already named in the carrier.

Finally attach P, N. These rows source and name the displayed packet; they add no quotient name, selected representative, host real equality, completeness row, or choice principle. Hence the five listed rows exhaust the local naming obligations. □

Theorem 1494.3 (RepresentedSpace name schedule obligation)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, every represented-space read opens the displayed name row* S *and the* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *before the representation relation row* ρ *is consumed. The schedule exposes the inspected precision requests and stream positions as finite* BHist *data, not as an implicit oracle over names.*

Proof. Project the carrier in Definition 1494.1. The name row S and schedule row W are the only coordinates from which stream observations may be read.

The NameCert obligations of Definition 1494.2 place carrier admission and classifier transport before relation exactness. Thus a consumer cannot reach ρ without first displaying the finite name and schedule rows.

The structural rows H, C, P, N only transport, replay, source, and name that same scheduled route. They do not add an unscheduled representation oracle. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier name schedule obligation`

Theorem 1494.4 (RepresentedSpace representation relation exactness)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, the row* ρ *is the only route from the stream observations displayed by* S, W *to the target carrier* X*. Any target-facing consumer that bypasses* ρ *is outside the local* $\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}$ *surface.*

Proof. Read the exactness clause in Definition 1494.2. It identifies ρ as the sole displayed relation between scheduled stream observations and the target row.

Classifier transport through T, H compares the same components on a finite surface. It can move an accepted representation route, but it cannot create a second relation from stream data to X.

Replay and provenance through C, P, N preserve the route already named by ρ. Therefore every accepted target read factors through ρ and no bypass is certified. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier representation relation exactness`

Theorem 1494.5 (RepresentedSpace target handoff nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing target handoff from an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The handoff exports no quotient names, selected representatives, host equality, completeness witness, or choice principle.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier target handoff nonescape`

Proof. Start with the name schedule obligation. It forces the target handoff to read S, W before the representation relation is consumed.

Apply representation relation exactness. The relation row ρ is the only certified route from the scheduled stream observations to X, and the real-facing case of X passes through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally use the structural rows. The BHist, hsame, Cont, Pkg, and NameCert data transport, replay, source, and name the handoff without adding quotient names, representatives, host equality, completeness, or choice. □

Theorem 1494.6 (RepresentedSpace StreamName Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing represented-space read through an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, a consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ *schedule* W*, then the regular rational readback and dyadic tolerance rows attached to the target carrier* X*, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and only then consumes the local* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route contains no quotient name, selected representative, host real equality, completeness row, or choice principle.*

Proof. Start with a real-facing read of the target row X. Carrier admission exposes the finite schedule W before the representation relation ρ can be consumed, so the first visible route is the $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126.

Next read the analytic handoff rows. The target carrier is real-facing only by passing through the $\mathsf{RegSeqRat}^{\uparrow}$ surface of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Finally replay H, C, P, N. These rows keep the handoff inside the local BHist, hsame, Cont, Pkg, and NameCert data. Since the carrier has no coordinate for quotient names, selected representatives, host real equality, completeness, or choice, no such route can be read by a consumer. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceStreamnameRealHandoff`

Theorem 1494.7 (RepresentedSpace schedule representation exactness)**.** *Every target read from an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *must expose the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* W *before the representation relation row* ρ *can be used. The representation relation is therefore exact only as a displayed route from the inspected stream observations to the target row* X*, with* Cont *replay and* Pkg *provenance preserving that schedule.*

Proof. Project the carrier of Definition 1494.1. The representation relation ρ is stored after the name carrier S and the finite schedule W, so no target read can consume ρ without first displaying the inspected stream positions and precision requests.

Replay through C preserves this order. The continuation row starts from the precision request, exposes the schedule, and only then reaches the representation relation and target row.

Finally attach P and N. Provenance and local naming source the same scheduled route, and the carrier has no coordinate for an unscheduled representation oracle. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier schedule representation exactness`

Theorem 1494.8 (RepresentedSpace classifier transport)**.** *Accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier sameness is transported componentwise through* $S, W, \rho, X, T, H, C, P, N$*. The transport compares the name carrier, schedule, representation relation, target row, classifier transport, structural replay, provenance, and local naming rows; it never appeals to host equality of names, target objects, real representatives, or quotient classes.*

Proof. Read the classifier row T together with the transport data H. These rows compare accepted packets only after their displayed components have been moved to a common finite surface.

The representation and target rows move with the same schedule. Thus ρ and X are compared only through the corresponding scheduled carrier rows, not by host equality.

The structural rows C, P, N then replay, source, and name the transported packet. Since no host equality, quotient class, or selected representative is present in the carrier, transport is exhausted by the listed components. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier classifier transport`

Theorem 1494.9 (RepresentedSpace target seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing target read from an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *packet must pass through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The read cannot bypass the regular rational readback, dyadic tolerance ledger, terminal Real seal, componentwise transport, replay, provenance, or local naming evidence.*

Proof. Start with the target carrier X. In the real-facing case, the carrier definition routes X through the regular rational readback surface, the dyadic tolerance ledger, and the terminal Real seal.

Apply the schedule exactness theorem. The target read has already exposed the finite StreamName schedule before ρ is consumed, so the analytic rows are downstream of a displayed name schedule rather than an ambient target object.

Finally replay the structural rows H, C, P, N. They keep the read inside BHist, hsame, Cont, Pkg, and NameCert data. Coordinate exhaustion blocks any route that skips RegSeqRat, DyadicRatCore, the Real seal, transport, replay, provenance, or local naming. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier target seal nonescape`

Theorem 1494.10 (RepresentedSpace realizer readback window)**.** *For an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, every realizer readback window that reaches a real-facing target must expose the* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to* X*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before the represented target is consumed. Thus readback is a finite windowed route through the displayed carrier, not a quotient of names or a selected host real.*

Proof. Project the carrier from Definition 1494.1. The name carrier S and the finite schedule W are displayed before the representation relation ρ and target row X can be read.

Use Definition 1494.2. Its exactness clause makes ρ the only route from stream observations to the target carrier, while its transport clause keeps S, W, ρ, X, T on one finite comparison surface.

Finally apply Definition 1494.9. In the real-facing case the route reaches the regular rational readback, dyadic tolerance ledger, and terminal Real seal before replaying the BHist, hsame, Cont, Pkg, and NameCert evidence. Since no quotient-name or selected-host coordinate is present, every realizer readback window remains inside the displayed route. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier realizer readback window`

Theorem 1494.11 (RepresentedSpace NameCert scope package)**.** *The scope package of an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *is exactly the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback path to the real-facing* X *row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, finite* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No represented-space scope clause may add host real equality, a choice-selected representative, or an ambient completion row outside that package.*

Proof. Start with Definition 1494.2. Carrier admission exposes the rows $S, W, \rho, X, T, H, C, P, N$, and the obligation list stores no additional scope coordinate.

Classifier transport contributes only T together with the componentwise transports in H. This keeps the stream schedule, representation relation, and target carrier on a common finite surface without importing host equality.

The real-facing part is bounded by Definition 1494.6 and Definition 1494.9. Those theorems force the target route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. The listed package is therefore the whole represented-space scope. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.ScopePackage.RepresentedSpaceCarrier namecert scope package`

Theorem 1494.12 (RepresentedSpace translation nonescape)**.** *Every translation from a* $\mathsf{StreamName}^{\uparrow}$ *name to a real-facing represented target in an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier passes through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, finite* BHist *packet, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The translation cannot escape into an external name quotient, host equality of reals, selected representative, or ambient completeness principle.*

Proof. Read the translation as a target read from the carrier. By Definition 1494.7, the finite $\mathsf{StreamName}^{\uparrow}$ schedule is exposed before the representation relation row can be used.

The classifier-transport theorem then fixes the comparison surface for $S, W, \rho, X, T, H, C, P, N$. Translation therefore moves the displayed name, schedule, representation relation, and target row together rather than replacing them by host equality.

For a real-facing target, Definition 1494.10 supplies the readback window through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ before the structural rows are replayed. The carrier has no coordinate outside BHist, hsame, Cont, Pkg, and NameCert for this route, so the translation has no escape path. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier translation nonescape`

Theorem 1494.13 (RepresentedSpace admissible name coverage)**.** *Every admissible* $\mathsf{RepresentedSpace}^{\uparrow}$ *name accepted by the displayed representation row is covered by the finite route*

$$S, W, \rho, X, T, H, C, P, N.$$

In particular, the read starts with the $\mathsf{StreamName}^{\uparrow}$ *finite windows in* W*, passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal exposed by the target row* X*, transports classifier data by the componentwise* hsame *rows in* H*, replays the requested observations by* C*, and ends in the* Pkg *provenance and local* NameCert *row* N*.*

Proof. Project the carrier of Definition 1494.1. The admissible name is not a free stream; it is read through the displayed name carrier S and finite schedule W, so the only inspected observations are the $\mathsf{StreamName}^{\uparrow}$ windows stored in the packet.

Use the representation relation row ρ next. It is the sole route from those stream observations to the target row X, and the real-facing case of X exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal before any consumer receives a target value.

Finally transport and replay the route. The componentwise hsame rows in H keep $S, W, \rho, X, T, C, P, N$ on the same finite comparison surface, while C replays the inspected windows and P, N source the resulting NameCert row. Thus every admissible name covered by the representation row is covered by the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier admissible name coverage`

Theorem 1494.14 (RepresentedSpace completion-boundary refusal)**.** *An accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *packet does not by itself certify a completion boundary, a quotient completion, or a host-real representative. Any completion-facing consumer must first display the representation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback path, the* $\mathsf{Real}^{\uparrow}$ *seal, the* hsame *transport,* Cont *replay,* Pkg *provenance, and the same local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier completion boundary refusal`

Proof. Inspect the carrier coordinates. They include a name carrier, finite windows, a representation row, a target row, classifier transport, structural transport, replay, provenance, and local naming, but no coordinate for a completed ambient space, quotient completion, or selected host-real representative.

The refusal is stable under the analytic handoff. A completion-facing read still has to pass through ρ, the finite $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ readback exposed by X, and the terminal $\mathsf{Real}^{\uparrow}$ seal; none of those rows adds a completion object.

It remains to check transport and provenance. The hsame rows, Cont replay, Pkg provenance, and local NameCert only preserve and source the displayed route. Since the packet never stores an additional completion boundary, the represented-space certificate refuses to export one. □

Theorem 1494.15 (RepresentedSpace realization transport)**.** *If two accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *packets compare by the displayed classifier, then every realized target read transported from the first packet to the second is transported through the same finite representation route:* $\mathsf{StreamName}^{\uparrow}$ *windows,* ρ*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *comparison,* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Start from the classifier transport theorem. Equality of accepted packets is not host equality of names or targets; it is the componentwise hsame comparison of $S, W, \rho, X, T, H, C, P, N$ on a common finite surface.

Transport the realized read along those rows. The finite $\mathsf{StreamName}^{\uparrow}$ windows move with W, the representation relation moves with ρ, and the target read moves with X, where the real-facing route still passes through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal.

Finally replay the structural evidence on the second packet. The Cont row reproduces the same inspected windows, while Pkg provenance and the local NameCert row source the transported certificate. No unrecorded realization map, quotient equality, or host representative is introduced by the transport. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier realization transport`

Theorem 1494.16 (RepresentedSpace StreamName representation source)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, a representation-name read must open the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W *before any target-facing inspection of* X *is available. The source of the representation is therefore the displayed name carrier and schedule pair* S, W*, not an implicit name oracle, a quotient of names, or a selected representative.*

Proof. Project the carrier. The only source rows that expose name observations are S and the finite $\mathsf{StreamName}^{\uparrow}$ schedule W.

Apply the NameCert obligations and the realization transport theorem. Both place the representation relation ρ and the target row X downstream of the displayed schedule, so a target consumer cannot inspect X before the finite windows have been opened.

The structural rows T, H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for an implicit oracle, quotient name, or selected representative, the representation source is exactly the scheduled StreamName surface. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier streamname representation source`

Theorem 1494.17 (RepresentedSpace realization exactness)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, the accepted realizations are exactly the displayed relation row* ρ *between the scheduled source observations* S, W *and the target carrier row* X*. A row belongs to the local realization surface precisely when it is read through* S, W, ρ, X *and then preserved by* T, H, C, P, N*.*

Proof. Use Definition 1494.4. It identifies ρ as the sole route from scheduled stream observations to the target row.

The forward direction is immediate from carrier admission: an accepted realization displays S, W, consumes ρ, reaches X, and then carries the route through classifier transport, componentwise hsame, replay, provenance, and local naming.

Conversely, any row read through the displayed S, W, ρ, X route and preserved by T, H, C, P, N satisfies the carrier and NameCert obligations. There is no second realization coordinate in the packet, so the displayed relation is exact. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier realization exactness`

Theorem 1494.18 (RepresentedSpace target seal ledger package)**.** *Every real-facing target seal exported by an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier is packaged through the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,*

Real$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows attached to* $S, W, \rho, X, T, H, C, P, N$. *The package excludes quotient representatives, selected names, host real equality, choice-selected realizers, and ambient completion data.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceTargetSealLedgerPackage`

Proof. Start from the target carrier row X. In the real-facing case, the carrier definition and target-seal theorem route X through regular rational readback, dyadic tolerance data, and the terminal Real seal.

Transport and replay are then fixed by the structural rows. The componentwise hsame rows in H keep the target route on the same finite comparison surface, while C replays the inspected windows and P, N source the package.

No excluded object is a coordinate of R. Thus quotient representatives, selected names, host real equality, choice-selected realizers, and ambient completion data cannot be exported by the target seal ledger. □

Theorem 1494.19 (RepresentedSpace effective metric realizer handoff)**.** *Let an accepted* RepresentedSpace$^{\uparrow}$ *carrier be used by an* EffectiveMetricSpace$^{\uparrow}$ *or* EffectiveMetricCompletion$^{\uparrow}$ *consumer. Before any distance read, completion-tail read, or Real-facing seal is available, the carrier must expose its finite* StreamName$^{\uparrow}$ *schedule, representation relation,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance row, and* Real$^{\uparrow}$ *seal through the displayed rows of Definition 1494.1.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier effective metric realizer handoff`

Proof. Begin with Definition 1494.16. It forces every target-facing read to open the finite StreamName$^{\uparrow}$ schedule before the target row can be inspected.

Apply Definition 1494.17. The realized target is exactly the displayed relation row between the scheduled source observations and the target carrier, so an effective metric consumer cannot substitute an implicit realizer or selected name.

Finally use Definition 1494.18. Its ledger route packages the real-facing target through RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance, the terminal Real$^{\uparrow}$ seal, transport, replay, provenance, and local naming. Hence the effective metric or completion consumer receives the finite handoff before any distance or completion read is available. □

Theorem 1494.20 (RepresentedSpace effective-metric root obligations)**.** *Every effective metric read of an accepted* RepresentedSpace$^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *must pass through the displayed* StreamName$^{\uparrow}$ *schedule* S*, observation windows* W*, representation relation* ρ*, target row* X*, classifier transport* T*, componentwise* hsame *rows* H*, replay row* C*, provenance row* P*, and local* NameCert *row* N*. In the real-facing case, the same read exposes the* RegSeqRat$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *rows named by the target seal before any* Real$^{\uparrow}$ *seal is available.*

Proof. Open Definition 1494.1. The source schedule, observation windows, relation, target row, transport, replay, provenance, and naming rows are coordinates of the same finite packet.

Apply Definition 1494.2 and Definition 1494.17. They force every target-facing read to move through $S, W, \rho, X, T, H, C, P, N$ rather than through a selected name or host representation equality.

For real-facing metric reads, apply Definition 1494.18. It records the RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ route attached to the displayed target seal. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier root effective metric obligations`

Theorem 1494.21 (RepresentedSpace completion-readback root obligations)**.** *Every completion readback consumed from an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier is routed through* $S, W, \rho, X, T, H, C, P, N$ *and through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows before completion vocabulary is available. It cannot use quotient representatives, selected Cauchy names, host equality, or ambient completion data.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier root completion readback obligations`

Proof. Start with Definition 1494.19. It places effective metric and completion consumers behind the finite stream schedule, representation relation, regular readback, dyadic tolerance, and Real seal route.

The realization is exact by Definition 1494.17, so the completion readback remains attached to ρ and X. The target seal ledger package then supplies the regular-rational and dyadic rows used by completion-facing reads.

The excluded objects are not coordinates of Definition 1494.1; transport, replay, provenance, and local naming preserve the displayed route. □

Theorem 1494.22 (RepresentedSpace Real-seal root obligations)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier is obtained from the displayed* $S, W, \rho, X, T, H, C, P, N$ *route and from the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *target-seal ledger. No* $\mathsf{Real}^{\uparrow}$ *consumer receives host real equality, quotient representation data, selected realizers, countable choice, or an ambient completeness theorem from the represented-space packet.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier root real seal obligations`

Proof. Read the target row X through Definition 1494.17. The target is available only through the displayed relation between scheduled source observations and the carrier row.

Apply Definition 1494.18. It packages every real-facing target seal through regular rational readback, dyadic tolerance data, the terminal Real seal, componentwise transport, replay, provenance, and local naming.

Since the packet has no coordinate for host equality, quotient representatives, selected realizers, choice, or ambient completeness, the Real-seal route is exactly the displayed finite handoff. □

Theorem 1494.23 (RepresentedSpace admissible finality)**.** *Let an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *be consumed by a* $\mathsf{Real}^{\uparrow}$*-facing represented-space consumer that also exposes an* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *packet. Before the consumer reads output data, the displayed route must factor through*

$$\mathsf{RepresentedSpace}^{\uparrow} \longmapsto \mathsf{CauchyRepresentation}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The factorization is the finite represented-space carrier route of Definition 1494.1, the exact representation relation of Definition 1494.4, the StreamName-to-Real handoff of Definition 1494.6, and the admissible Cauchy finality interface of chapter 1760 and chapter 1680. It excludes quotient names, selected representatives, host real equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier admissible finality`

Proof. Open the represented-space carrier. By Definition 1494.1, the only source available to the consumer is the displayed finite route through $S, W, \rho, X, T, H, C, P, N$.

Use Definition 1494.4. The representation relation ρ is the sole certified path from scheduled stream observations to the target carrier, so an admissible consumer cannot bypass the displayed representation row.

Apply Definition 1494.6. In the real-facing case the target route opens the $\mathsf{StreamName}^{\uparrow}$ schedule, then the regular rational and dyadic readback rows, and only then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal.

The exposed $\mathsf{AdmissibleRepresentation}^{\uparrow}$ packet supplies the finality clause of chapter 1760, while chapter 1680 supplies the Cauchy source used by that clause. Since none of the represented-space carrier coordinates stores quotient names, selected representatives, host real equality, countable choice, or ambient real completeness, the displayed factorization is the whole admissible-finality route. □

Theorem 1494.24 (RepresentedSpace scoped package)**.** *Let an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *packet expose the name schedule, representation relation, target handoff,* $\mathsf{StreamName}^{\uparrow}$ *Real handoff, effective-metric realizer handoff, completion-readback obligations, Real-seal obligations, and admissible finality rows named in Definition 1494.1. Then it exports a scoped* $\mathsf{NameCert}$ *package over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{EffectiveMetricSpace}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The package is read only through the displayed carrier route* $S, W, \rho, X, T, H, C, P, N$, *and it does not add quotient names, selected representatives, host real equality, countable choice, ambient completion data, or a separate parameter-transport equivalence.*

Proof. Begin with the local carrier and obligation surface. Apply Definition 1494.2, Definition 1494.3, Definition 1494.4, and Definition 1494.5 to expose the schedule, representation relation, target handoff, transport, replay, provenance, and local naming rows.

For the real-facing route, use Definition 1494.6, Definition 1494.19, and Definition 1494.22. These rows force every metric or Real-seal consumer through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff before any target read is available.

For completion and admissibility, apply Definition 1494.21 and Definition 1494.23. The completion-readback and admissible-finality rows are downstream of the same carrier route, so the scoped package is exactly the displayed finite $\mathsf{NameCert}$ surface and not a new equivalence between unrelated parameter families. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.ScopePackage.RepresentedSpaceCarrier scoped package`

Theorem 1494.25 (RepresentedSpace Cauchy-name scope package)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$, *each represented-space Cauchy-name read is scoped by the displayed representation schedule* S, W, *the target row* X, *the representation relation row* ρ, *the classifier transport row* T, *and the replay package* H, C, P, N. *Cauchy names arriving from* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *enter the packet only through those displayed representation rows and through the target-seal ledger already attached to* X. *The read exports no source-equivalence transport statement, quotient representation, selected Cauchy name, host equality, countable choice, or ambient completion datum.*

Proof. Open the carrier of Definition 1494.1. The source schedule, observation windows, representation relation, target row, classifier transport, replay, provenance, and local naming rows are the only coordinates available to a represented-space read.

Apply the NameCert obligations and name-schedule coverage, Definition 1494.2 and Definition 1494.3. They force every represented read to display the finite schedule and its observation windows before the target row can be consumed.

The Cauchy-name content is then read through the existing target route. Theorems 1494.16, 1494.17, and 1494.18 place $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data behind the representation relation and the target-seal ledger attached to X.

Finally use the scoped package theorem Definition 1494.24. It keeps transport, replay, provenance, and local naming inside H, C, P, N and does not add an equivalence between source names. Since the excluded objects are not carrier coordinates, the Cauchy-name read remains scoped by the displayed representation rows. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier cauchy name scope package`

Theorem 1494.26 (RepresentedSpace realizer scope)**.** *For every accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$*, each realizer read is scoped by the displayed rows* $S, W, \rho, X, T, H, C, P, N$*. The read must open the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule in* W *before it consumes the representation relation* ρ *or the target row* X*, and it cannot bypass the schedule by using an implicit name oracle, quotient representation, selected realizer, host equality, countable choice, or ambient completion datum.*

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier realizer scope`

Proof. Project the carrier of Definition 1494.1. The only realizer-facing source rows are the name carrier S, the finite schedule W, the representation relation ρ, and the target row X, with T, H, C, P, N preserving and naming the same route.

Apply the schedule obligation of Definition 1494.3 and the source theorem Definition 1494.16. They force every target-facing realizer read to expose the finite $\mathsf{StreamName}^{\uparrow}$ windows before ρ or X can be inspected.

Use representation exactness and target nonescape, Definition 1494.17 and Definition 1494.5. The realized target is read through the displayed relation and cannot be replaced by an implicit oracle or host equality.

Finally apply Definition 1494.24. Componentwise transport, replay, provenance, and local naming remain inside T, H, C, P, N, so the excluded quotient, selected-realizer, choice, and completion objects have no carrier coordinate from which a bypass could be read. □

Theorem 1494.27 (RepresentedSpace admissible Cauchy handoff)**.** *Let an accepted* $\mathsf{RepresentedSpace}^{\uparrow}$ *carrier* $R = (S, W, \rho, X, T, H, C, P, N)$ *be consumed by an admissible Cauchy-facing represented-space consumer. Before any target row is sealed for that consumer, the read factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff is mediated by the displayed schedule W*, representation relation* ρ*, target row* X*, componentwise* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local name* N*. It exports no quotient Cauchy names, selected representatives, host real equality, countable choice, ambient real completeness, or separate admissibility oracle.*

Proof. Begin with the Cauchy-name scope package Definition 1494.25. It places Cauchy-facing reads behind the displayed schedule, representation relation, target row, classifier transport, replay, provenance, and local naming rows.

Use Definition 1494.6 and Definition 1494.18. They route the real-facing target through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal before a target consumer can read the sealed object.

Apply admissible finality from Definition 1494.23. The admissible consumer receives the same finite route rather than a separate oracle over names.

The scoped package theorem Definition 1494.24 then replays T, H, C, P, N around that route. Since quotient Cauchy names, selected representatives, host equality, choice, and ambient completeness are absent from Definition 1494.1, the displayed handoff is the whole admissible Cauchy interface. □

Lean4 ✓ `BEDC.Derived.RepresentedSpaceUp.RepresentedSpaceCarrier admissible cauchy handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RepresentedSpace}^{\uparrow}$ packet is generated from a displayed name carrier, finite $\mathsf{StreamName}^{\uparrow}$ schedule, representation relation, target carrier, classifier transport, componentwise hsame rows, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1494.1, Definition 1494.2, Definition 1494.4, Definition 1494.6, Definition 1494.15, Definition 1494.5, Definition 1494.19, Definition 1494.21, Definition 1494.22, Definition 1494.23, Definition 1494.24, and Definition 1494.27 give mature paper treatment of the public represented-space carrier route as the bridge-facing surface through CauchyRepresentation-style name reads, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RepresentedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient representations, selected host representatives, host equality, a choice principle, or a Lean theorem.

Upgrade path: The paper axis is mature; the verification axis remains available for a checked represented-space theorem and standard bridge.

Chapter 1495

A Concrete Naming Certificate for $\mathsf{LawlessSequence}^{\uparrow}$

Following Brouwer and Bishop-style choice sequence analysis, the $\mathsf{LawlessSequence}^{\uparrow}$ packet records only the finite observation surface of a lawless sequence. It sits between the $\mathsf{StreamName}^{\uparrow}$ finite-prefix interface of chapter 126, the $\mathsf{BaireSpace}^{\uparrow}$ naming surface of chapter 334, the $\mathsf{Bool}^{\uparrow}$ digit row of chapter 32, and the $\mathsf{Nat}^{\uparrow}$ prefix-index ledger of chapter 28. The packet is a NameCert for finite-prefix inspection and replay; it is not a host infinite sequence, not a primitive lawless choice object, and not a countable-choice principle.

The finite-window route consumes the $\mathsf{StreamName}^{\uparrow}$ finite-prefix carrier of chapter 126 and the $\mathsf{BaireSpace}^{\uparrow}$ prefix row of chapter 334 before any digit is read, so the chapter cannot be read as a host infinite-sequence surface.

Definition 1495.1 (Lawless sequence carrier)**.** *A* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, B, I, H, C, P, N)$$

with the following displayed rows:

1. *W is a* $\mathsf{StreamName}^{\uparrow}$ *finite-prefix window recording exactly the prefixes currently inspected.*
2. *B is the* $\mathsf{Bool}^{\uparrow}$ *digit row attached to each inspected prefix in W.*
3. *I is the* $\mathsf{Nat}^{\uparrow}$ *prefix-index ledger ordering the displayed prefix requests and their finite extension steps.*
4. *H transports finite prefixes and their digit rows componentwise by* hsame*.*
5. *C records displayed* Cont *replay routes from an index request in I to its prefix window in W and from that window to the corresponding digit row in B.*
6. *P records* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{LawlessSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the finite rows W, B, I, H, C, P, N. It has no coordinate for host infinite sequence data, countable choice, primitive lawless choice, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceTasteGate single carrier alignment`

Theorem 1495.2 (Lawless sequence NameCert obligations)**.** *For every accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier* $L = (W, B, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LawlessSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the seven displayed rows, finite-prefix window coverage through* W*, Boolean digit readback through* B*, unary prefix-index ordering through* I*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceCarrier namecert obligations`

Proof. Project the finite packet of Definition 1495.1. The only sequence observations available to the classifier are the displayed $\mathsf{StreamName}^{\uparrow}$ prefix windows in W, the corresponding $\mathsf{Bool}^{\uparrow}$ digit rows in B, and the $\mathsf{Nat}^{\uparrow}$ prefix-index ledger I.

Perform prefix induction over the finite row I. The base case reads the empty displayed prefix and its digit row when present. The step case extends the current inspected prefix by one unary index move and then reads the next visible digit through B.

Transport is componentwise. The rows in H move each displayed prefix and digit row by hsame, while C replays the route from index requests to windows and from windows to digits. Repacking these projections with P and N gives the stated local NameCert surface without adding any hidden infinite-sequence coordinate. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceNameCertObligations`
↪ variant `BEDC.Derived.LawlessSequenceUp.NameCertObligations.LawlessSequenceNameCertObligations`

Theorem 1495.3 (Lawless sequence finite-window scope)**.** *Every consumer read of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *packet factors through the finite route*

$$I \longmapsto W \longmapsto B$$

together with H, C, P, N*. Thus the packet supplies finite prefix observation and digit replay, but it does not supply host infinite sequences, countable choice, lawless choice as a primitive, quotient streams, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read and project its requested prefix index from I. Prefix induction over the displayed $\mathsf{Nat}^{\uparrow}$ row gives a finite chain of inspected prefixes and no appeal to an ambient sequence.

At each step, read the corresponding window from W and the digit row from B. The $\mathsf{Bool}^{\uparrow}$ row is attached to the displayed prefix, so the digit is a finite observation rather than a value extracted from an unseen total function.

Finally route the read through H, C, P, N. Componentwise transport and continuation replay preserve the same finite window and digit rows, and the carrier has no coordinate from which a host infinite sequence, choice principle, quotient stream, or real-completeness object could be read. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceFiniteWindowScope`

Theorem 1495.4 (Lawless sequence StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{BaireSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier receives only the displayed finite-prefix window* W*, its* $\mathsf{Bool}^{\uparrow}$ *digit rows* B*, the* $\mathsf{Nat}^{\uparrow}$ *prefix-index ledger* I*, and the structural rows* H, C, P, N*. The handoff therefore presents lawless-sequence observation as a StreamName finite-window route, not as an infinite stream object or a choice-selected total sequence.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceStreamNameHandoff`

Proof. Read the accepted carrier of Definition 1495.1. The $\mathsf{StreamName}^{\uparrow}$ face is exactly W, and the $\mathsf{BaireSpace}^{\uparrow}$ use is restricted to the same displayed finite prefixes.

Next read the digit and index rows. The $\mathsf{Bool}^{\uparrow}$ row B supplies only the observed digits for prefixes listed by I, while the $\mathsf{Nat}^{\uparrow}$ ledger I supplies only the finite prefix-ordering route.

The handoff is completed by the structural rows H, C, P, N. They transport, replay, source, and name the same finite prefix windows, so downstream consumers receive a StreamName-compatible finite observation packet and receive no host infinite stream, primitive lawless-choice source, countable-choice row, quotient-stream equality, or $\mathsf{Real}^{\uparrow}$ completeness surface. □

Theorem 1495.5 (Lawless sequence choice-free carrier obligation)**.** *Every accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier read is a finite prefix packet over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. In particular, every consumer-visible observation is read from the displayed rows* W, B, I, H, C, P, N *of Definition 1495.1, and no host infinite sequence, primitive lawless choice source, quotient stream equality, or countable-choice schedule is exported.*

Proof. Project the carrier rows in Definition 1495.1. The only prefix data available to a consumer is the $\mathsf{StreamName}^{\uparrow}$ window W together with the $\mathsf{Bool}^{\uparrow}$ digit row B and the $\mathsf{Nat}^{\uparrow}$ index ledger I.

Apply the finite-window scope theorem Definition 1495.3. It factors every consumer read through $I \longmapsto W \longmapsto B$ before the structural rows can be consumed.

Finally use the StreamName handoff theorem Definition 1495.4. The handoff transports and replays the same finite rows through H, C, P, N, so no accepted read can recover a host infinite sequence, a primitive lawless-choice source, quotient stream equality, or countable-choice schedule. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceChoiceFreeCarrierObligation`

Theorem 1495.6 (Lawless sequence StreamName window obligation)**.** *Every replayed observation of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier factors through displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{BaireSpace}^{\uparrow}$ *prefix rows before a digit read is exposed. The replay route is finite: it reads an index from* I*, a window from* W*, the corresponding digit from* B*, and then the transport, continuation, provenance, and local naming rows* H, C, P, N*.*

Proof. Begin with a replayed observation and project its index request from I. The finite-window scope theorem Definition 1495.3 sends that request through the displayed window route $I \longmapsto W \longmapsto B$.

The $\mathsf{BaireSpace}^{\uparrow}$ use is the prefix-row reading already named in Definition 1495.4. It is attached to the same $\mathsf{StreamName}^{\uparrow}$ window W, not to a completed stream or an ambient function.

Now replay through C and transport through H. Provenance P and the local $\mathsf{NameCert}$ row N name the same finite packet, so every accepted observation remains a displayed StreamName window with BaireSpace prefix coverage and a visible Bool digit row. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceStreamNameWindowObligation`

Theorem 1495.7 (Lawless sequence Real-boundary nonescape)**.** *No accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *packet exports* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a host infinite sequence, or a countable-choice schedule. Any downstream countable-name or real-boundary consumer must first pass through the finite rows* W, B, I, H, C, P, N *and receives only the StreamName-compatible finite prefix surface of Definition 1495.4.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceRealBoundaryNonescape`

Proof. Read the carrier of Definition 1495.1. Its classifier compares packets only through the finite rows W, B, I, H, C, P, N, and the definition explicitly leaves $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host infinite sequence data, and countable choice outside the carrier.

Apply Definition 1495.5. Every accepted read is already a finite prefix packet over StreamName, Bool, Nat, transport, replay, provenance, and local naming.

Finally apply the StreamName handoff theorem Definition 1495.4. A countable-name consumer receives only the finite prefix window and its digit rows, so any real-boundary use that asks for completeness, quotient equality, a host sequence, or a countable-choice schedule exits the accepted $\mathsf{LawlessSequence}^{\uparrow}$ packet. □

Definition 1495.8 (Lawless sequence finite-window carrier)**.** *The finite-window refinement of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier* $L = (W, B, I, H, C, P, N)$ *is the displayed prefix packet*

$$L_{\mathrm{fin}} = (I, W, B, H, C, P, N)$$

read in dependency order: I *names the finite observed prefix requests,* W *supplies exactly the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows,* B *attaches the visible* $\mathsf{Bool}^{\uparrow}$ *digit rows to those windows, and* H, C, P, N *transport, replay, source, and name the same observed packet. The refinement forgets no consumer-visible row of* L*, but it records explicitly that every observation is made through a finite prefix packet before any downstream sequence or real-boundary reader is served.*

Theorem 1495.9 (Lawless sequence StreamName choice window)**.** *Every observed choice window of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *packet factors through the finite rows*

$$I, W, B, H, C, P, N.$$

In particular, the window first reads a $\mathsf{Nat}^{\uparrow}$ *prefix index from* I*, then a* $\mathsf{StreamName}^{\uparrow}$ *prefix window from* W*, then the corresponding* $\mathsf{Bool}^{\uparrow}$ *digit row from* B*, and only then the transport, continuation, provenance, and local naming rows. No consumer can read an observed choice window as a total sequence, an unseen future law, or a countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceStreamNameChoiceWindow`

Proof. Begin with an observed choice window and project the finite-window carrier of Definition 1495.8. The first available datum is the prefix-index row I, so the consumer must name a finite $\mathsf{Nat}^{\uparrow}$ request before any digit can be read.

Use finite prefix induction over I. The base case reads the empty displayed prefix when it is present in W. The step case extends the observed prefix by one displayed request, consults the matching $\mathsf{StreamName}^{\uparrow}$ window in W, and reads only the attached $\mathsf{Bool}^{\uparrow}$ digit in B.

Finally replay the structural rows. The rows H, C, P, N transport and name the same finite window; they have no coordinate for future values outside W or for a choice schedule selecting all positions at once. □

Theorem 1495.10 (Lawless sequence finite-prefix induction)**.** *Every finite extension of the* $\mathsf{Nat}^{\uparrow}$ *prefix-index ledger* I *in an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier* $L = (W, B, I, H, C, P, N)$ *preserves the displayed route*

$$I \longmapsto W \longmapsto B \longmapsto H, C, P, N.$$

The induction exposes only the prefixes already listed by the finite ledger and their visible $\mathsf{Bool}^{\uparrow}$ *digit rows. It exposes no unseen future values, no host infinite sequence, and no countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceFinitePrefixInduction`

Proof. Induct on the displayed finite prefix ledger I. The base case is the empty-prefix read: the consumer reads the initial request from I, consults the corresponding $\mathsf{StreamName}^{\uparrow}$ window in W when present, and obtains only the attached visible digit row in B.

For the step, extend the displayed request by one unary prefix move. The carrier offers no direct digit coordinate outside W, so the next read must pass through the extended StreamName window and then through its attached Bool row.

Finally replay the structural route. Componentwise transport in H, continuation replay in C, provenance in P, and local naming in N preserve the same finite prefix packet at each stage, so the induction cannot reveal any value beyond the displayed ledger. □

Theorem 1495.11 (Lawless sequence Real nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *packet receives* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a host infinite sequence, or a countable-choice schedule. Every such consumer receives only the finite-window packet* $L_{\mathrm{fin}} = (I, W, B, H, C, P, N)$ *of Definition 1495.8, routed through the StreamName choice window of Definition 1495.9.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceRealNonescape`

Proof. Read the Real-boundary nonescape theorem Definition 1495.7. It already excludes completeness, quotient equality, host infinite sequences, and countable-choice schedules from the accepted packet.

Now refine the carrier read through Definition 1495.8. The only prefix-facing object passed to a real-boundary consumer is L_{fin}, whose rows are the observed indices, StreamName windows, Bool digits, transport, replay, provenance, and local naming.

Apply Definition 1495.9. Since each observed choice window factors through I, W, B, H, C, P, N, a Real consumer cannot bypass the finite prefix packet to obtain quotient stream equality, a host sequence, or a countable-choice schedule. □

Theorem 1495.12 (Lawless sequence prefix-induction obligation)**.** *Every accepted prefix read of a* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier* $L = (W, B, I, H, C, P, N)$ *is obtained by finite induction over the* $\mathsf{Nat}^{\uparrow}$ *index row* I*. The base read exposes the displayed empty prefix when it is present in* W*; each step extends the current displayed request by one index move in* I*, reads the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window from* W*, and then reads the attached* $\mathsf{Bool}^{\uparrow}$ *digit row from* B*. No accepted prefix read is supplied by a host infinite sequence or by a countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequencePrefixInductionObligation`

Proof. Project the carrier of Definition 1495.1. The only index row available to a prefix consumer is I, so every request starts from the displayed finite prefix ledger.

Induct over the finite row I. The base case reads the empty displayed prefix and stops if no digit row is attached. The step case consumes one displayed successor request, reads the matching window from W, and attaches only the visible digit row in B.

Finally replay through H, C, P, N. These structural rows transport and name the same finite read, and none contains a coordinate for unseen future values or a choice schedule. □

Theorem 1495.13 (Lawless sequence prefix-window determinacy)**.** *Let* $L = (W, B, I, H, C, P, N)$ *and* $L' = (W', B', I', H', C', P', N')$ *be accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carriers. If the rows* I, W, B, H, C, P, N *and* $I', W', B', H', C', P', N'$ *are transported componentwise by* hsame *and replayed through the displayed* Cont *routes, then every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*, or real-boundary consumer sees the same*

finite observed prefix window and the same visible $\mathsf{Bool}^{\uparrow}$ *digit rows. The determined surface is exactly the transported finite window; it exposes no future values, no host infinite sequence, and no countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequencePrefixWindowDeterminacy`

Proof. Project both carriers through Definition 1495.1. The only rows capable of affecting a prefix-window read are the $\mathsf{Nat}^{\uparrow}$ ledgers I, I', the displayed StreamName windows W, W', the Bool digit rows B, B', and the structural rows that transport and replay them.

Apply the prefix-induction obligation, Definition 1495.12. It forces each accepted prefix read to proceed by finite induction over the transported Nat ledger, so both reads inspect the same finite sequence of displayed prefix requests.

Use Baire-window stability, Definition 1495.14. Since the prefix windows and Bool rows are transported componentwise by H, H' and replayed through C, C', each Baire-facing consumer receives the same finite window and the same visible digit rows.

Finally apply the countable-name boundary, Definition 1495.15. Real-boundary and countable-name consumers remain inside W, B, I, H, C, P, N, so determinacy of the displayed window does not determine any future value, host infinite sequence, or countable-choice schedule. □

Theorem 1495.14 (Lawless sequence Baire-window stability)**.** *For accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carriers, every* $\mathsf{BaireSpace}^{\uparrow}$ *prefix window read from* W *remains stable under componentwise* hsame *transport and displayed* Cont *replay. If two accepted reads agree on* I, W, B *by the transport row* H*, then the replay row* C *returns the same finite Baire window and the same visible Bool digit rows, with no appeal to quotient stream equality or an ambient total sequence.*

Proof. Begin with two accepted window reads and project their I, W, B rows. Componentwise hsame in H compares only displayed prefix indices, displayed StreamName windows, and displayed Bool digit rows.

Use the replay row C. The route from an index request to a Baire-facing prefix window is the displayed route already named by Definition 1495.6; it does not pass through a completed stream.

Repack the transported read with P and N. Since the carrier has no coordinate for quotient stream equality or host total sequence data, stability is exactly the finite Baire-window stability stated above. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceBaireWindowStability`

Theorem 1495.15 (Lawless sequence countable-name boundary)**.** *Every countable-name consumer of an accepted* $\mathsf{LawlessSequence}^{\uparrow}$ *carrier receives only the rows* W, B, I, H, C, P, N*. The consumer may inspect displayed finite StreamName windows, Bool digit rows, Nat prefix indices, componentwise transport, continuation replay, provenance, and local naming, but it receives no countable-choice schedule, no host infinite sequence, no quotient stream equality, and no* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the carrier of Definition 1495.1. Its classifier compares accepted packets only through W, B, I, H, C, P, N.

Apply the prefix-induction obligation and Baire-window stability above. They force every countable-name request through finite indices, finite windows, visible digits, and displayed replay.

Thus a countable-name consumer can only receive the seven displayed rows. Any request for a countable-choice schedule, a host infinite sequence, quotient stream equality, or real completeness leaves the accepted packet. □

Lean4 ✓ `BEDC.Derived.LawlessSequenceUp.LawlessSequenceCountableNameBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{LawlessSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LawlessSequence}^{\uparrow}$ packet is generated from finite StreamName prefix windows, Bool digit rows, Nat prefix indices, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LawlessSequence}^{\uparrow}}$, publicClosed)

Scope closed: Definition 1495.1, Definition 1495.3, Definition 1495.4, Definition 1495.12, Definition 1495.14, and Definition 1495.15 export the countable-name boundary theorem over finite StreamName windows, BaireSpace prefix rows, Bool digit rows, Nat prefix-index ledgers, hsame transport, Cont replay, Pkg provenance, local NameCert data, and the real-boundary nonescape surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawlessSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite sequences, countable choice, lawless choice as a primitive, quotient streams, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, finite-window scope, StreamName handoff, and TasteGate field faithfulness for BEDC.Derived.LawlessSequenceUp.

Chapter 1496

A Concrete Naming Certificate for $\mathsf{AbelTest}^{\uparrow}$

Following Abel and Bishop–Bridges constructive analysis, the $\mathsf{AbelTest}^{\uparrow}$ packet records bounded partial sums, a monotone null coefficient sequence, finite summation-by-parts rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{DyadicRatCore}^{\uparrow}$ estimates, $\mathsf{UniformLimit}^{\uparrow}$ rows, and a $\mathsf{Real}^{\uparrow}$ seal. It sits beside the Dirichlet-series source discipline of chapter 70, the uniform convergence boundary of chapter 463, the bounded real sequence rows of chapter 1415, and the Cauchy product route of chapter 464; its finite observation route also uses the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational readback of chapter 301, the dyadic estimate surface of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet is a constructive Abel-test surface for a displayed scalar coefficient route over a bounded partial-sum source; it is not a selected series sum, a quotient stream equality, host real completeness, countable choice, or compactness by open covers.

Definition 1496.1 (AbelTest carrier)**.** *An* $\mathsf{AbelTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, A, M, D, T, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded partial-sum row, displaying the finite partial sums of the source series together with the bound used by the Abel-test estimate.*
2. *A is the coefficient stream row, displaying the inspected scalar coefficient values on the finite windows read by the packet.*
3. *M is the monotone/null-tail ledger, recording the finite decrease checks for the coefficient row and the displayed thresholds at which the coefficient tail lies below a requested dyadic tolerance.*
4. *D is the dyadic summation-by-parts estimate row, exposing the finite Abel transform calculation that bounds a tail by the bounded partial-sum row and the coefficient variation ledger.*
5. *T is the finite tail schedule, assigning each precision request to a finite source window and a finite coefficient threshold used by D.*
6. *U is the* $\mathsf{UniformLimit}^{\uparrow}$*-facing row, displaying the uniform tail estimate consumed by downstream convergence readers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, A, M, D, T, U.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the regular rational handoff.*

9. *H records componentwise* hsame *transport for the partial-sum, coefficient, monotone/null-tail, summation-by-parts, tail-schedule, uniform-limit, regular-handoff, real-seal, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a precision request through T to the* $\mathsf{StreamName}^{\uparrow}$ *windows, through M and D to the uniform row U, then through R to the* $\mathsf{Real}^{\uparrow}$ *seal row E.*

11. *P is the* Pkg *provenance over* $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{Dirichlet}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these twelve displayed rows. It has no coordinate for a chosen infinite sum, quotient stream equality, choice-selected limit, open-cover compactness, host series summation, or ambient completeness.

Lean4 ✓ `BEDC.Derived.AbelTestUp.AbelTestTasteGate single carrier alignment`

Theorem 1496.2 (AbelTest NameCert obligations)**.** *For every accepted* $\mathsf{AbelTest}^{\uparrow}$ *carrier $Q = (S, A, M, D, T, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed twelve-row packet;*

2. *bounded-partial-sum transport for S, preserving the displayed bound and the inspected partial sums along the componentwise* hsame *route;*

3. *monotone-null coefficient stability for A and M, preserving the displayed decrease checks and null-tail thresholds on the admitted* $\mathsf{StreamName}^{\uparrow}$ *windows;*

4. *summation-by-parts ledger exactness for D, stating that the dyadic tail estimate consumed by U is the finite Abel transform calculation recorded in the packet;*

5. *Real-seal non-escape for U, R, E, H, C, P, N, forcing every seal-facing consumer through the uniform estimate row and regular rational handoff.*

These obligations are finite-row obligations over the sibling surfaces of chapter 70, chapter 463, chapter 1415, chapter 464, chapter 126, chapter 301, chapter 291, and chapter 38.

Proof. Project the accepted packet $Q = (S, A, M, D, T, U, R, E, H, C, P, N)$. The projection exposes exactly the bounded partial-sum row, coefficient stream row, monotone/null-tail ledger, dyadic summation-by-parts estimate, finite tail schedule, uniform-limit-facing row, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

Read a displayed precision request through T. The schedule supplies a finite source window and a finite coefficient threshold, and the rows S, A, M provide the bounded partial sums, the inspected coefficient values, and the monotone null-tail data visible on that window.

The dyadic ledger D records the finite summation-by-parts calculation on the same rows. Therefore the estimate sent to U is the displayed Abel transform bound, not a hidden infinite summation argument or a choice-selected limiting process.

Finally apply the structural rows. The componentwise transports in H preserve the analytic rows together, while C, P, N replay, source, and name only that displayed route. Since the carrier has no further coordinate, the five listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}$ surface. □

Theorem 1496.3 (AbelTest uniform-limit handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{UniformLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{AbelTest}^{\uparrow}$ *carrier factors through*

$$S, A, M, D, T, U, R, E, H, C, P, N.$$

In particular, no accepted consumer can read host series sums, quotient streams, choice-selected limits, open-cover compactness, or ambient completeness from the Abel-test packet.

Proof. Begin with an accepted carrier and a downstream $\mathsf{Real}^{\uparrow}$ or $\mathsf{UniformLimit}^{\uparrow}$ read. By the carrier definition, the only convergence-facing rows are U, R, and E, and each is downstream of the same finite route through S, A, M, D, T.

Read the handoff route. The tail schedule T selects finite windows, the rows A and M display monotone null coefficient control on those windows, and the bounded partial-sum row S supplies the bound used by the dyadic estimate.

The row D is the only summation-by-parts ledger available to the consumer. Thus the uniform row U receives a finite dyadic Abel transform estimate, and the regular handoff R and real seal E receive only that displayed estimate together with the transport, continuation, provenance, and local naming rows.

No prohibited object remains in the packet. Host series sums, quotient streams, choice-selected limits, open-cover compactness, and ambient completeness are absent from the carrier coordinates and cannot be recovered by hsame transport or Cont replay. □

Falsifiable prediction: An accepted $\mathsf{AbelTest}^{\uparrow}$ consumer that reaches a $\mathsf{UniformLimit}^{\uparrow}$ row, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, or $\mathsf{Real}^{\uparrow}$ seal without reading the bounded partial-sum row, coefficient stream row, monotone/null-tail ledger, dyadic summation-by-parts estimate, and finite tail schedule refutes the carrier surface.

Independence witness: dirichlet-series, uniformlimit, boundedrealsequence, cauchyproduct, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole twelve-row packet $Q = (S, A, M, D, T, U, R, E, H, C, P, N)$, because each sibling contributes only a source, uniform tail, bounded-row, product, window, regular-readback, dyadic-estimate, or seal face while this chapter binds the Abel-test coefficient route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelTest}^{\uparrow}$ packet is generated from bounded partial-sum rows, a displayed monotone null coefficient stream, a finite dyadic summation-by-parts ledger, a finite tail schedule, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{UniformLimit}^{\uparrow}$ row, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1496.1, Definition 1496.2, and Definition 1496.3 seed the finite Abel-test boundary over $\mathsf{Dirichlet}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected series sums, quotient stream equality, host real equality, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the twelve-row carrier and prove the finite NameCert obligation surface and uniform-limit handoff theorem.

Chapter 1497

A Concrete Naming Certificate for $\mathsf{DirichletTest}^{\uparrow}$

Following Dirichlet and Bishop–Bridges constructive analysis, the $\mathsf{DirichletTest}^{\uparrow}$ packet records the bounded-partial-sum convergence test as finite BEDC data before a completion-facing consumer may read a regular rational handoff or a real seal. Its source series face sits beside $\mathsf{Series}^{\uparrow}$ in chapter 121; its summation-by-parts sibling is $\mathsf{AbelTest}^{\uparrow}$ in chapter 1496; its Dirichlet-series source discipline is chapter 70; its uniform tail boundary is chapter 463; and its bounded-row discipline is chapter 1415. The finite observation route also uses $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, dyadic estimates from chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet is a constructive Dirichlet-test surface for a displayed scalar coefficient route over a bounded partial-sum source; it is not a selected series sum, a quotient series equality, host real completeness, countable choice, or compactness by open covers.

Definition 1497.1 (DirichletTest carrier)**.** *A* $\mathsf{DirichletTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, B, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded partial-sum row, displaying finite partial sums of the source series and the bound used in the Dirichlet tail estimate.*
2. *A is the coefficient stream row, displaying the inspected scalar coefficient values on the finite windows read by the packet.*
3. *B is the null-tail ledger, recording the displayed thresholds at which the coefficient tail lies below a requested dyadic tolerance and the finite variation rows needed by the estimate.*
4. *T is the Dirichlet tail estimate schedule, assigning each precision request to a finite source window and a coefficient threshold used to bound the displayed tail.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, A, B, T.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the regular rational handoff.*
7. *H records componentwise* hsame *transport for the partial-sum, coefficient, null-tail, tail-estimate, regular-handoff, real-seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through T to the* $\mathsf{StreamName}^{\uparrow}$ *windows, through S, A, B to the Dirichlet estimate, then through R to the* $\mathsf{Real}^{\uparrow}$ *seal row E.*

9. *P is the* Pkg *provenance over* $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{Dirichlet}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for a chosen infinite sum, quotient series equality, choice-selected limit, open-cover compactness, host series summation, or ambient completeness.

Theorem 1497.2 (DirichletTest NameCert obligations)**.** *For every accepted* $\mathsf{DirichletTest}^{\uparrow}$ *carrier* $D = (S, A, B, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *bounded-partial-sum transport for S, preserving the displayed bound and inspected partial sums along the componentwise* hsame *route;*
3. *coefficient-stream exposure for A, preserving the inspected scalar coefficient rows on the admitted* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *null-tail and finite-variation exposure for B, preserving the displayed thresholds and variation rows used by the Dirichlet estimate;*
5. *tail-estimate schedule exactness for T, stating that each requested tolerance is answered by the finite source window and coefficient threshold recorded in the packet;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal exposure for R and E;*
7. *transport stability, continuation replay, provenance, and local naming for exactly the route recorded by H, C, P, N.*

These obligations are finite-row obligations over the sibling surfaces of chapter 121, chapter 1496, chapter 70, chapter 463, chapter 1415, chapter 126, chapter 301, chapter 291, and chapter 38.

Proof. Project the accepted packet $D = (S, A, B, T, R, E, H, C, P, N)$. The projection exposes exactly the bounded partial-sum row, coefficient stream row, null-tail ledger, tail-estimate schedule, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

Read a displayed precision request through T. The schedule supplies a finite source window and a coefficient threshold, while S supplies the bounded partial sums and A, B supply the inspected coefficient values together with the null-tail and finite-variation data visible on that window.

The estimate is therefore a finite Dirichlet tail calculation on the displayed rows. It is not an appeal to an external infinite series sum, a hidden quotient stream equality, or a choice-selected limiting process.

Finally apply the structural rows. The componentwise transports in H preserve the analytic rows together, while C, P, N replay, source, and name only that displayed route. Since the carrier has no further coordinate, the seven listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}$ surface. □

Theorem 1497.3 (DirichletTest Cauchy handoff)**.** *Every accepted* $\mathsf{DirichletTest}^{\uparrow}$ *carrier sends the displayed partial-sum family to the Cauchy-facing route only through the finite handoff*

$$S, A, B, T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *consumer can use the Dirichlet test only after reading the bounded partial-sum row, coefficient stream row, null-tail ledger, and tail-estimate schedule that bind the finite tail estimate to the regular rational handoff.*

Proof. Begin with an accepted carrier and a displayed precision request. The schedule T supplies a finite source window and a coefficient threshold, while the source row S supplies only the partial sums and bound visible on that window.

Read the coefficient rows. The stream row A displays the inspected coefficient values, and the ledger B supplies the null-tail threshold and finite variation rows needed by the Dirichlet estimate.

These rows are exactly the data consumed by the Cauchy-facing route: sufficiently late partial-sum tails are bounded by the displayed bounded-partial-sum constant and the coefficient tail data named by B and T. The handoff row R records that regular rational readback and does not receive any separate completed real or quotient series object.

Transport and replay preserve this handoff componentwise. The rows H, C, P, N attach the route to the local certificate without creating an alternate path through host convergence, selected sums, quotient equality, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1497.4 (DirichletTest Real-seal boundary)**.** *For every accepted* $\mathsf{DirichletTest}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *seal-facing read factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R produced from the displayed bounded partial sums, coefficient stream, null-tail ledger, and tail-estimate schedule. The seal row E receives only R together with the componentwise transports, continuation replay, provenance, and local naming rows; it receives no selected sum, quotient series equality, host real completeness, countable choice, or open-cover compactness.*

Proof. Project the seal-facing rows R and E from the accepted carrier. The regular handoff R is downstream of S, A, B, T, and the seal row E is downstream of R.

Apply Definition 1497.3. It identifies the only route from bounded partial sums and coefficient-tail data to the regular readback as the finite path through the displayed source, coefficient, null-tail, and schedule rows.

Read the seal row. Its analytic input is exactly the regular handoff R, together with H, C, P, N. Since the carrier has no coordinate for a selected sum, quotient series equality, host real completeness, countable choice, or open-cover compactness, the seal-facing read cannot export any of those objects. □

Falsifiable prediction: An accepted $\mathsf{DirichletTest}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without reading the bounded partial-sum row, coefficient stream row, null-tail ledger, and Dirichlet tail estimate schedule refutes the carrier surface.

Independence witness: series, abel-test, dirichlet-series, uniformlimit, boundedrealsequence, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole ten-row packet $D = (S, A, B, T, R, E, H, C, P, N)$, because each sibling contributes only a source, summation-by-parts route, Dirichlet-series face, uniform tail, bounded-row, window, regular-readback, dyadic-estimate, or seal face while this chapter binds the coefficient null-tail route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DirichletTest}^{\uparrow}$ packet is generated from bounded partial-sum rows, a displayed coefficient stream, a null-tail ledger, a finite Dirichlet tail estimate schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1497.1, Definition 1497.2, Definition 1497.3, and Definition 1497.4 seed the finite Dirichlet-test boundary over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{Dirichlet}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected sums, quotient series equality, host real completeness, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface, Cauchy handoff theorem, and Real-seal boundary theorem.

Chapter 1498

A Concrete Naming Certificate for $\mathsf{LocatedOpen}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedOpen}^{\uparrow}$ packet records a neighbourhood interface whose evidence is located in finite real and cut data before any topological consumer reads it. It sits over the located cut surface of chapter 404, the located real enclosure of **??**, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing filter route of chapter 333. The chapter names only a finite L10 neighbourhood packet: it does not identify opens with quotient subsets of host reals and does not invoke open-cover compactness.

Definition 1498.1 (LocatedOpen carrier)**.** *A* $\mathsf{LocatedOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, R, S, Q, W, E, T, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *source row exposing the finite lower and upper cut comparisons used by the local membership read.*
2. *R is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row tying the same comparisons to a located real presentation.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family recording the inspected stream positions and replayable requests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to those windows.*
5. *W is the displayed located-membership window row. For each inspected candidate it records a finite interval-entry witness or a positive dyadic radius witness around the candidate.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located cut, located real, stream, regular rational, and membership-window rows have been displayed.*
7. *T is the local classifier transport row moving L, R, S, Q, W, E to a common finite comparison surface.*
8. *H records componentwise* hsame *transport for the cut, real, stream, regular rational, membership-window, Real-seal, classifier, continuation, provenance, and local naming rows.*
9. *C records the displayed* Cont *replay from a neighbourhood request through the inspected windows, radius or interval-entry evidence, readback rows, Real seal, and classifier transport.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedOpen}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed finite windows and transports. It has no coordinate for quotient subsets of reals, host real equality, a choice-selected neighbourhood, arbitrary-union completion, open-cover compactness, or a proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.LocatedOpenUp.TasteGate.LocatedOpenTasteGate single carrier alignment`

Theorem 1498.2 (LocatedOpen window stability)**.** *For every accepted* $\mathsf{LocatedOpen}^{\uparrow}$ *packet* $O = (L, R, S, Q, W, E, T, H, C, P, N)$*, the located-membership windows in* W *are stable under the componentwise* hsame *transports recorded in* H*. In particular, a transported consumer reads the same finite interval-entry witnesses and positive dyadic radius witnesses after* L, R, S, Q, W, E *have been moved to the common comparison surface named by* T*.*

Lean4 ✓ `BEDC.Derived.LocatedOpenUp.LocatedOpenCarrier window stability`

Proof. Project the carrier of Definition 1498.1. The only membership evidence available to a consumer is the row W, and each entry of W is already displayed as a finite interval-entry witness or a positive dyadic radius witness.

Apply the transport row H componentwise. The located cut, located real, stream, regular rational, membership-window, and Real-seal rows are moved together to the comparison surface named by T, so the transported packet contains the same window indices and the same radius or interval-entry witnesses.

Finally replay the continuation row C. Replay follows the transported windows and readbacks; it does not introduce an ambient subset, a host equality test, a selected neighbourhood, or an arbitrary union. Thus the located membership surface is stable exactly over the displayed finite windows. □

Lean4 ✓ `BEDC.Derived.LocatedOpenUp.LocatedOpen window stability`

Theorem 1498.3 (LocatedOpen Real consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedOpen}^{\uparrow}$ *packet factors through*

$$\mathsf{LocatedCut}^{\uparrow},\quad \mathsf{LocatedReal}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{CauchyFilter}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},$$

Concretely, the consumer first reads the located cut and located real rows, then the finite stream and regular rational readbacks, then the displayed located-membership windows, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal and filter-facing continuation route.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing read of the seal row E. The carrier stores E after the located cut row L, located real row R, stream window row S, regular rational readback row Q, and located-membership window row W. Therefore no consumer can reach the seal without first exposing these finite rows.

The completion-facing part is only a route. The provenance row names $\mathsf{CauchyFilter}^{\uparrow}$ so that a filtered consumer can replay the finite neighbourhood request through C, but the packet supplies no completeness field and no arbitrary-union constructor.

Attach T, H, P, N to the same route. Classifier transport, componentwise hsame, package provenance, and local naming keep the consumer inside the listed rows. Since the carrier has no coordinate for quotient subsets, host real equality, selected neighbourhoods, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, no such datum can be recovered from the read. □

Theorem 1498.4 (LocatedOpen obligation surface)**.** *For every accepted* $\mathsf{LocatedOpen}^{\uparrow}$ *packet* $O = (L, R, S, Q, W, E, T, H, C, P, N)$*, the rows* $L, R, S, Q, W, E, T, H, C, P, N$ *jointly supply carrier admission, located-membership window stability, the Real-facing route, classifier transport, replay, provenance, and local* NameCert *non-escape. The exported surface is exactly the finite neighbourhood interface of Definition 1498.1; it exports no quotient subset, selected neighbourhood, arbitrary union, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 1498.1. Carrier admission is the displayed tuple $O = (L, R, S, Q, W, E, T, H, C, P, N)$, with L, R, S, Q, W, E naming the located cut, located real, stream, regular rational, membership-window, and Real-seal rows.

Window stability is supplied by Definition 1498.2. The theorem keeps the interval entry witnesses and positive dyadic radius witnesses inside W after the componentwise transports in H move the packet to the comparison surface named by T.

The Real-facing route is supplied by Definition 1498.3. It forces every consumer to read the located cut, located real, stream windows, regular rational readbacks, membership windows, terminal Real seal, and filter-facing replay before any Real-facing output is available.

The remaining rows are bookkeeping rows for non-escape. The classifier transport is T, replay is C, provenance is P, and local naming is N. Since the carrier definition contains no coordinate for quotient subsets, selected neighbourhoods, arbitrary unions, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the obligation surface cannot export any such row. □

Lean4 ✓ `BEDC.Derived.LocatedOpenUp.LocatedOpenObligationSurface`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedOpen}^{\uparrow}$ packet is generated from located cut rows, located real enclosure rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, displayed membership windows with positive dyadic radius or interval-entry witnesses, a terminal $\mathsf{Real}^{\uparrow}$ seal, classifier transport, componentwise hsame rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1498.1, Definition 1498.2, and Definition 1498.3 seed the finite L10 located-neighbourhood interface over located real and cut data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general topology, open-cover compactness, Heine–Borel, arbitrary-union completion, quotient subsets of reals, host real equality, choice-selected neighbourhoods, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: A formal route can encode BEDC.Derived.LocatedOpenUp.LocatedOpenCarrier and prove window stability, Real-facing consumer routing, classifier transport, provenance, and local NameCert exactness.

Chapter 1499

A Concrete Naming Certificate for $\mathsf{RegularCauchyAbs}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyAbs}^{\uparrow}$ packet lifts the finite $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ endpoint-and-tolerance route through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic absolute-value surface of chapter 293 and beside the regular Cauchy operation surfaces of chapter 417. The packet exposes absolute value for regular Cauchy names as finite NameCert data; it is not trichotomy, quotient streams, a host absolute-value operation, selected limits, or ambient completeness.

Definition 1499.1 (RegularCauchyAbs carrier)**.** *A* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, W, D, V, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row;*
2. *W is the scheduled family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows through which X is observed;*
3. *D is the family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from those windows;*
4. *V is the family of* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint-and-tolerance rows applied to the displayed dyadic endpoints;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the absolute-value windows and their finite tolerance ledger;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R has been exposed;*
7. *H stores componentwise* hsame *transports for the source, windows, dyadic endpoints, absolute-value rows, readback, seal, provenance, and local name;*
8. *C stores the displayed* Cont *routes from X to W and D, from D to V, from V to R, and from R to E;*
9. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, trichotomy of reals, a host absolute-value operation, selected limits, or a $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsTasteGate single carrier alignment`

Theorem 1499.2 (RegularCauchyAbs NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}^{\uparrow}}$ *surface for Definition 1499.1 consists exactly of source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, scheduled* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint readback,* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint-and-tolerance application,* $\mathsf{RegSeqRat}^{\uparrow}$ *absolute-value handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet A. The coordinates list the regular source name, stream windows, dyadic endpoint rows, dyadic absolute-value rows, regular readback, real seal, transport, continuation, provenance, and local name.

The endpoint-and-tolerance obligation is inherited from the finite dyadic packet named in chapter 293. Each observed endpoint in D is routed through a visible $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ row in V, and no order oracle or completed-real absolute value is consulted.

The remaining obligations are the regular and real-facing handoffs. The rows H, C, P, N transport the same finite route, replay it for consumers, record the package provenance, and bind the local naming surface without adding quotient streams, selected limits, trichotomy, host absolute value, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier namecert obligations`

Theorem 1499.3 (RegularCauchyAbs RegSeqRat handoff)**.** *For every accepted* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *carrier* $A = (X, W, D, V, R, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is assembled only from the displayed source row* X*, the scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows* D*, and the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *tolerance rows* V*. A* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer can read the absolute-value name only through those rows and through the stored* hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Proof. Project the accepted carrier. The source regular name is visible only as X, and the finite observations of that name are the scheduled windows W together with the endpoint rows D.

Apply the dyadic absolute-value route to each displayed endpoint row. The rows in V supply the nonnegative endpoint and tolerance ledger for the same window schedule, so they are the only data available for building the absolute-value readback.

Repack the displayed absolute-value windows as the readback row R. The continuation routes in C and package row P record this handoff over the same BHist support, and the transport rows in H preserve it componentwise for consumers. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier regseqrat handoff`

Theorem 1499.4 (RegularCauchyAbs Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *carrier reaches the seal row* E *only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *absolute-value handoff* R *has been produced from* X, W, D, V*. The real-facing boundary exposes the route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$*, and it exposes no quotient stream, host real absolute value, selected limit, trichotomy witness, or ambient completeness principle.*

Proof. Begin with a real-facing consumer. By Definition 1499.3, the consumer cannot reach a regular absolute-value readback until the finite window route X, W, D, V has been displayed.

The seal row E is downstream of that readback. The stored continuation routes first replay the source windows, then the dyadic endpoint observations, then the $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ tolerance rows, and only then the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R consumed by the real seal.

Since Definition 1499.1 contains no separate coordinate for quotient streams, trichotomy, host absolute value, selected limits, or completeness, the $\mathsf{Real}^{\uparrow}$ seal boundary is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier real seal boundary`

Theorem 1499.5 (RegularCauchyAbs non-escape boundary)**.** *No accepted* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *packet exports data outside the displayed tuple* $X, W, D, V, R, E, H, C, P, N$*. In particular, it cannot be used to recover a quotient of regular Cauchy streams, a host absolute-value operation, a trichotomy proof for real names, a selected limiting real, or a* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the carrier of Definition 1499.1. Its only source data are the regular name, finite stream windows, dyadic endpoints, dyadic absolute-value rows, readback, seal, and structural rows.

Apply Definition 1499.2. Every local obligation is a projection of those rows or a transport, continuation, provenance, or naming route for those rows.

Finally combine Definition 1499.3 with Definition 1499.4. All regular and real-facing consumers pass through $X, W, D, V, R, E, H, C, P, N$, so no unlisted analytic or quotient object can be extracted from the certificate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier non escape`

Theorem 1499.6 (RegularCauchyAbs obligation closure package)**.** *The carrier of Definition 1499.1 together with Definition 1499.2, Definition 1499.3, Definition 1499.4, and Definition 1499.5 forms the complete obligation surface for* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The package preserves the refusals of trichotomy, quotient streams, host absolute value, selected limits, and ambient completeness.*

Proof. Project the carrier row. It exposes exactly the source regular name X, scheduled stream windows W, dyadic endpoint rows D, dyadic absolute-value tolerance rows V, regular readback R, real seal E, transport H, continuation replay C, provenance P, and local name N.

Use the NameCert obligations to identify the local certificate surface over those rows. The handoff theorem then routes every $\mathsf{RegSeqRat}^{\uparrow}$ consumer through $X, W, D, V, R, H, C, P, N$, while the seal-boundary theorem routes every $\mathsf{Real}^{\uparrow}$ consumer through the same finite prefix before E is available.

Finally apply the non-escape boundary. Since every obligation is read from the displayed rows and since the excluded principles have no carrier coordinate, the package closes exactly the finite obligation surface and preserves the stated refusals. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier obligation closure`

Theorem 1499.7 (RegularCauchyAbs public export surface)**.** *The public export of an accepted* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *packet is exactly the finite route from the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row through scheduled* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint observations,* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *tolerance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *absolute-value handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}^{\uparrow}}$ *row. Every public consumer must read the route as*

$$X, W, D, V, R, E, H, C, P, N.$$

It exports no trichotomy, quotient-stream equality, host real absolute value, selected limit, ambient completeness principle, or public bridge.

Proof. Project the carrier of Definition 1499.1. The displayed public rows are the source regular name X, windows W, dyadic endpoints D, dyadic absolute-value tolerance rows V, regular readback R, real seal E, transport H, replay C, provenance P, and local name N.

Apply the obligation closure package of Definition 1499.6. It identifies the NameCert obligations, RegSeqRat handoff, Real seal boundary, and non-escape theorem as one finite route over the same displayed rows.

Therefore a public consumer can reuse the packet only through $X, W, D, V, R, E, H, C, P, N$. The excluded analytic and bridge objects have no carrier coordinate and no obligation row, so they cannot be exported by the public surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAbsUp.RegularCauchyAbsCarrier public export surface`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyAbs}^{\uparrow}$ packet is generated from one RegSeqRat source row, scheduled StreamName windows, DyadicRatCore endpoint observations, DyadicAbsoluteValue endpoint-and-tolerance rows, a RegSeqRat readback row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}\uparrow}$, publicClosed)

Scope closed: Definition 1499.1, Definition 1499.2, Definition 1499.3, Definition 1499.4, Definition 1499.5, Definition 1499.6, and Definition 1499.7 close the public finite absolute-value route through carrier admission, NameCert obligations, RegSeqRat handoff, Real seal boundary, nonescape, obligation closure, and the public export surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyAbs}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, trichotomy, quotient-stream equality, host real absolute value, selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Lean verification requires checked carrier admission, window exposure, dyadic absolute-value tolerance, RegSeqRat handoff, RealUp seal boundary, public export surface, transport stability, provenance, and non-escape for the same finite packet.

Chapter 1500

A Concrete Naming Certificate for $\mathsf{NormedRealLine}^{\uparrow}$

Following Bishop and Bridges' constructive normed-space analysis, the $\mathsf{NormedRealLine}^{\uparrow}$ packet records the real line as a normed object only through the regular Cauchy absolute-value route. It binds the real seal of chapter 38, the regular rational readback of chapter 301, the stream-window interface of chapter 126, the dyadic observation core of chapter 291, the absolute-value packet of chapter 1499, and the metric interface of chapter 57. The certificate names the RealUp line equipped with that displayed norm route; it does not assert RealUp completeness, trichotomy, quotient equality of names, or a host norm function.

Definition 1500.1 (Normed real line carrier)**.** *A* $\mathsf{NormedRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, S, D, A, M, H, C, P, N)$$

with the following visible coordinates:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the underlying real-line object.*
2. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source and readback row through which the real seal is observed by regular rational names.*
3. *D is the joined* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger recording the finite windows and dyadic endpoints read from S.*
4. *A is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *norm row, applied only to the displayed regular Cauchy name and its dyadic observations.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-and-norm interface that reads the distance of the real line through the same absolute-value handoff.*
6. *H records componentwise* hsame *transport for R, S, D, A, M and for the structural rows.*
7. *C records the displayed* Cont *routes from real-seal requests to regular windows, from those windows to dyadic observations, from the observations to the absolute-value row, and from that row to the metric interface.*
8. *P records* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NormedRealLine}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a selected host real, quotient equality of real names, trichotomy, a host norm function, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 1500.2 (Normed real line NameCert obligations)**.** *For every accepted* $\mathsf{NormedRealLine}^{\uparrow}$ *carrier* $L = (R, S, D, A, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NormedRealLine}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real seal, regular rational source and readback, stream and dyadic observation ledger, regular Cauchy absolute-value norm row, metric distance-and-norm interface, componentwise transport, continuation replay, package provenance, and local naming row. No obligation is stored outside those rows.*

Proof. Project the carrier of Definition 1500.1. The projection exposes the real seal R, the regular rational source and readback S, the observation ledger D, the absolute-value norm row A, the metric interface M, and the structural rows H, C, P, N.

The analytic obligation is the norm route. The row A is already the displayed $\mathsf{RegularCauchyAbs}^{\uparrow}$ handoff over S and D, so the certificate reads norm data only after the stream windows and dyadic endpoints have been displayed.

The metric obligation is then a consumer of the same handoff. The row M receives its distance-and-norm interface through A, and the rows H, C, P, N transport, replay, source, and name that finite route without adding a host norm function, quotient equality, trichotomy, or completeness. □

Theorem 1500.3 (Normed real line regular Cauchy norm handoff)**.** *Every norm-facing consumer of an accepted* $\mathsf{NormedRealLine}^{\uparrow}$ *carrier* $L = (R, S, D, A, M, H, C, P, N)$ *reaches the norm row only through the route*

$$R \longrightarrow S \longrightarrow D \longrightarrow A \longrightarrow M.$$

Thus the exported norm route is the $\mathsf{RegularCauchyAbs}^{\uparrow}$ *handoff over displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, not a separate host-level norm operation.*

Proof. Begin with a norm-facing consumer. Carrier admission gives a single real seal row R and a single regular rational source and readback row S through which R is observed.

Read the finite observations next. The observation ledger D joins the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows for the same regular name, so the absolute-value row cannot be reached without first displaying those windows and endpoints.

Apply the $\mathsf{RegularCauchyAbs}^{\uparrow}$ row A to the displayed observations. The metric interface M is downstream of A, and the continuation rows in C record this order for every consumer route. Hence the norm handoff is exactly the displayed regular Cauchy absolute-value route. □

Theorem 1500.4 (Normed real line non-escape boundary)**.** *No accepted* $\mathsf{NormedRealLine}^{\uparrow}$ *packet exports data outside the tuple* $R, S, D, A, M, H, C, P, N$*. In particular, the packet cannot be used to recover RealUp completeness, real trichotomy, quotient equality of names, a selected host real, or a host norm function.*

Proof. Inspect the carrier in Definition 1500.1. Its only analytic data are the real seal, the regular rational source and readback, the stream and dyadic observations, the regular Cauchy absolute-value row, and the metric interface.

Use Definition 1500.2 to exhaust the local certificate. Every named obligation is a projection of those rows or a transport, continuation, provenance, or naming route for those rows.

Finally combine this exhaustion with Definition 1500.3. Any norm-facing or metric-facing consumer must pass through $R, S, D, A, M, H, C, P, N$, so no unlisted completeness, trichotomy, quotient, selected-real, or host-norm object can be extracted. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormedRealLine}^{\uparrow}}$.

Constructive story: A $\mathsf{NormedRealLine}^{\uparrow}$ packet is generated from a RealUp seal row, a RegSeqRat source and readback row, StreamName and DyadicRatCore observations, a RegularCauchyAbs norm row, a MetricSpace distance-and-norm interface, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormedRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1500.1, Definition 1500.2, Definition 1500.3, and Definition 1500.4 seed the finite normed real-line route through $\mathsf{RegularCauchyAbs}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormedRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, RealUp completeness, trichotomy, quotient equality of real names, selected host reals, host norm functions, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, regular Cauchy norm handoff, metric-interface routing, transport stability, provenance, and non-escape for the same finite packet.

Chapter 1501

A Concrete Naming Certificate for $\mathsf{CauchyFilterBasis}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley terminology of filter bases, the $\mathsf{CauchyFilterBasis}^{\uparrow}$ packet records the finite basis generator that sits between regular Cauchy windows and saturated Cauchy-filter evidence. It reads the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the dyadic tolerance rows of chapter 291; it hands the resulting directed basis to the saturation surface of chapter 333 before any $\mathsf{Real}^{\uparrow}$ seal from chapter 38 can be inspected. The packet is a finite generator interface, not a completed filter, not a quotient of Cauchy names, and not a host real equality principle.

The same basis row is consumed by chapter 5840 and chapter 7853 as the finite refinement surface before located or regular completion reads.

Definition 1501.1 (Cauchy filter basis carrier)**.** *A* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, S, R, D, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite list of basis elements. Each basis element names a nonempty finite family of late-window neighbourhood requests over the same observation surface.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite stream windows inspected by the basis elements and the tail indices at which those windows become available.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It turns the displayed stream windows in S into regular rational observations before any filter-basis comparison is made.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the dyadic radius and inclusion ledger used to compare the rational observations carried by R.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation handoff row. It consumes the directed basis generated by B, S, R, D and exposes it only as the finite-filterbase surface already named by* $\mathsf{CauchyFilter}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for B, S, R, D, F and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a basis request through S, R, D, and then the saturation handoff F.*

8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBasis}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed filter object, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1501.2 (Cauchy filter basis classifier)**.** *The classifier of an accepted* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *packet relates two basis reads precisely when their finite basis lists B admit a common displayed refinement whose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows agree after the componentwise* hsame *transport recorded in H, and when the resulting saturation handoff is the same* $\mathsf{CauchyFilter}^{\uparrow}$ *row F after replay by C.*

Equivalently, comparison is the finite common-refinement route

$$B, S, R, D \longmapsto F \longmapsto H, C, P, N.$$

It is not equality of completed filters, not equality of quotient Cauchy names, and not equality of host real limits.

Theorem 1501.3 (Cauchy filter basis NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *carrier $Q = (B, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis elements,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation handoff, transport, continuation, provenance, and local naming row;*

2. *common-refinement classification for two basis reads, using only the displayed basis elements and their transported window, readback, and tolerance rows;*

3. *directedness of the generated basis, requiring every pair of displayed basis elements to name a third displayed basis element refining both after the S, R, D rows are replayed;*

4. *saturation exactness, requiring the handoff to* $\mathsf{CauchyFilter}^{\uparrow}$ *to consume the generated finite basis before any filter-facing or real-facing consumer can proceed;*

5. *ledger non-escape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1501.1. The projection exposes exactly the finite basis list, the window row, the regular readback row, the dyadic tolerance row, the saturation handoff, and the structural rows H, C, P, N.

Classifier transport is componentwise. The rows in H move the basis elements, windows, rational readbacks, dyadic tolerances, and saturation handoff to a common finite comparison surface, while C replays the same dependency order.

Directedness is read from the finite list B together with the tolerance ledger D. For two displayed basis elements, the accepted packet names a common refinement whose windows and rational observations are routed through S and R before dyadic inclusion is checked in D.

Saturation exactness is the role of F. Since every consumer reaches F only after the basis, window, readback, and tolerance rows have been replayed, the package and local naming rows cannot export any completed-filter, quotient, choice, host-equality, or real-completeness principle. □

Theorem 1501.4 (Cauchy filter basis RegSeqRat handoff)**.** *Let $Q = (B, S, R, D, F, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *packet. Every regular-rational consumer of a displayed basis element factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window row S, then through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, then through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, before the generated basis can be handed to* $\mathsf{CauchyFilter}^{\uparrow}$ *by F.*

The handoff route is

$$B \longmapsto S \longmapsto R \longmapsto D \longmapsto F \longmapsto H, C, P, N.$$

It does not bypass the regular readback row and does not select a Cauchy representative or host real limit.

Proof. Start with a consumer of a displayed basis element. Carrier admission lists such a consumer under B, but the basis element only stores a finite neighbourhood request, not a rational observation by itself.

Replay the continuation row C. The request first reaches the stream-window row S, where the relevant finite observations and tail indices are displayed.

The next row is R. It turns the finite stream observations into $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, after which the dyadic tolerance row D checks the requested radius and inclusion ledger.

Only then can the same generated basis be consumed by F as a $\mathsf{CauchyFilter}^{\uparrow}$ saturation handoff. The remaining rows transport, package, and name that same route, so no bypass through a selected representative, quotient stream, host real limit, or choice principle is available. □

Theorem 1501.5 (Cauchy filter basis Real seal non-escape)**.** *For every accepted* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *carrier $Q = (B, S, R, D, F, H, C, P, N)$, any* $\mathsf{Real}^{\uparrow}$*-facing read exposed by the provenance row P factors through the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation handoff F after the finite basis route B, S, R, D has already been replayed. Therefore the packet cannot export a* $\mathsf{Real}^{\uparrow}$ *seal directly from basis elements, stream windows, rational readbacks, or dyadic tolerance rows.*

The only real-facing route visible in the packet is

$$B, S, R, D \longmapsto F \longmapsto P, N.$$

It is a non-escape condition for the basis generator: the chapter names finite basis data and its saturation handoff, not a completed real, not a chosen limit, and not a proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Project the provenance row P. Its $\mathsf{Real}^{\uparrow}$ occurrence is recorded only as a downstream boundary of the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, not as a coordinate of the basis rows B, S, R, D.

The continuation row C fixes the route order. A real-facing read must first replay the basis element, stream window, regular readback, and dyadic tolerance rows, and then pass through the saturation handoff F.

Transport by H preserves the same order under classifier comparison. The local naming row N names the packet only after this replay, so no separate real seal can be read from an earlier coordinate.

Thus every $\mathsf{Real}^{\uparrow}$-facing route remains downstream of $\mathsf{CauchyFilter}^{\uparrow}$. The packet carries no completed real, selected limit, host equality, countable-choice schedule, trichotomy, or real-completeness certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterBasis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1501.1, Definition 1501.2, Definition 1501.3, Definition 1501.4, and Definition 1501.5 seed the finite basis-generator interface between regular windows and saturated Cauchy filters.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyFilterBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed filters, quotient Cauchy names, selected representatives, host real equality, countable choice, trichotomy, or completeness of Real$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement classification, directed basis generation, RegSeqRat handoff, CauchyFilter saturation exactness, Real seal non-escape, and ledger refusal for the same packet.

Constructive story: The construction starts from finite basis elements, reads StreamName windows through RegSeqRat and DyadicRatCore rows, hands the directed basis to CauchyFilter saturation, and treats Real-facing access only as downstream provenance.

Chapter 1502

A Concrete Naming Certificate for $\mathsf{RegularCauchyMax}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMax}^{\uparrow}$ packet reads the maximum of two regular Cauchy names constructively through the identity $max(x, y) = (x + y + |x - y|)/2$. The certificate factors that identity through the finite pointwise-sum surface of chapter 417, the absolute-value surface of chapter 1499, the rational scaling surface of chapter 465, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It does not compare completed reals by trichotomy, choose a larger host real, quotient streams, or appeal to ambient completeness.

Definition 1502.1 (RegularCauchyMax carrier)**.** *A* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W_A, W_B, D_A, D_B, S, \Delta, V, T, Q, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *A and B are the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name rows.*
2. *W_A and W_B are the aligned* $\mathsf{StreamName}^{\uparrow}$ *windows used to observe the two sources at the same precision requests.*
3. *D_A and D_B are the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from those windows.*
4. *S is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row for the source sum $A + B$.*
5. *Δ is the displayed* $\mathsf{RegularCauchySum}^{\uparrow}$ *difference row for $A + (-B)$, with the negation supplied by the same rational scaling route used by the sibling operation surface.*
6. *V is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *row for $|\Delta|$.*
7. *T is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row for $S + V$.*
8. *Q is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row scaling T by the displayed rational one-half.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the scaled regular Cauchy name.*
10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R has been exposed.*
11. *H records componentwise* hsame *transport for the two sources, aligned windows, dyadic endpoints, sum, difference, absolute-value, second sum, half-scale, readback, seal, provenance, and local name.*

12. *C records the displayed* Cont *routes from the source windows to the dyadic endpoint rows, through the sum-absolute-value factorization, through the half-scale row, and onward to the readback and real seal.*

13. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMax}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy on real names, host real order, quotient stream equality, selected limits, or a $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCarrierNameCertUp.CauchyCarrierNameCertTasteGate single carrier alignment`

Theorem 1502.2 (RegularCauchyMax NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMax}^{\uparrow}}$ *surface for Definition 1502.1 consists exactly of two-source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, aligned* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint readback, the* $\mathsf{RegularCauchySum}^{\uparrow}$ *rows for $A+B$, $A+(-B)$, and $S+V$, the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *row for the displayed difference, the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row by one-half, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet M. The coordinates expose the two source regular names, their aligned stream windows, their dyadic endpoint rows, the three sum rows, the difference absolute-value row, the half-scale row, the regular readback, the real seal, and the structural rows.

The operation obligations are inherited from the cited sibling surfaces. Use chapter 417 for the source sum, the difference row, and the post-absolute-value sum; use chapter 1499 for $|\Delta|$; and use chapter 465 for multiplication by the displayed rational one-half.

The readback and seal obligations are then the finite handoffs through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The rows H, C, P, N transport, replay, source, and name this same route without adding trichotomy, host real order, quotient streams, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMaxUp.RegularCauchyMaxTasteGate single carrier alignment`

Theorem 1502.3 (RegularCauchyMax sum-abs factorization)**.** *For every accepted* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier $M = (A, B, W_A, W_B, D_A, D_B, S, \Delta, V, T, Q, R, E, H, C, P, N)$, every maximum-facing consumer factors through the displayed route*

$$(A, B, W_A, W_B, D_A, D_B) \longrightarrow (S, \Delta, V, T) \longrightarrow Q \longrightarrow R.$$

Thus the maximum readback is the regular Cauchy name obtained from $max(x, y) = (x + y + |x - y|)/2$ by finite sum, absolute-value, and one-half scaling rows, not by a comparison oracle for completed reals.

Proof. Begin with a maximum-facing consumer of the accepted packet. The only source reads available to it are the two $\mathsf{RegSeqRat}^{\uparrow}$ rows A, B, the aligned windows W_A, W_B, and the endpoint rows D_A, D_B.

Next build the algebraic route. The row S displays the source sum, Δ displays the difference route, V displays the absolute value of that difference, and T displays the sum of S with V. Each row is a finite sibling NameCert row over the same visible window and endpoint data.

Finally apply the half-scale row Q and expose the readback R. Since R is downstream of Q, and Q is downstream of S, Δ, V, T, the consumer route is exactly the stated sum-absolute-value factorization. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMaxUp.RegularCauchyMaxCarrier sum abs factorization`

Theorem 1502.4 (RegularCauchyMax lattice absorption)**.** *For every accepted* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier* $M = (A, B, W_A, W_B, D_A, D_B, S, \Delta, V, T, Q, R, E, H, C, P, N)$*, a consumer reading* $max(A, max(A, B))$ *factors through the same displayed source rows* A, B*, source-sum row* S*, difference and absolute-value rows* Δ, V*, post-absolute-value sum row* T*, half-scale row* Q*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *as the consumer reading* $max(A, B)$*. The absorbed read has no additional trichotomy branch, selected host maximum, quotient stream equality, selected limit, or ambient completeness coordinate.*

Proof. Apply Definition 1502.3 to the inner read $max(A, B)$. The inner consumer is forced through the displayed route from the shared source rows A, B and aligned windows to S, Δ, V, T, Q, and then to the readback row R.

Apply Definition 1502.3 again to the outer read $max(A, max(A, B))$. The first input is the same row A, while the second input is available only through the inner readback route already displayed in M.

The repeated A route is therefore identified by componentwise transport rather than by a host-level equality test. The $\mathsf{RegularCauchyAbs}^{\uparrow}$ row keeps the difference read inside Δ, V, and the $\mathsf{RegularCauchyScale}^{\uparrow}$ row keeps the one-half read inside Q, so both the inner and outer reads remain in the same finite packet.

Finally the $\mathsf{RegSeqRat}^{\uparrow}$ readback row and the $\mathsf{Real}^{\uparrow}$ seal boundary close the consumer route at R and E. Since Definition 1502.1 has no coordinate for trichotomy, selected host maxima, quotient streams, selected limits, or ambient completeness, the absorbed read cannot escape the displayed maximum packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMaxUp.RegularCauchyMaxCarrier lattice absorption`

Theorem 1502.5 (RegularCauchyMax Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier reaches the seal row* E *only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *has been assembled from the aligned source windows, dyadic endpoint rows,* $\mathsf{RegularCauchySum}^{\uparrow}$ *rows,* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *difference row, and* $\mathsf{RegularCauchyScale}^{\uparrow}$ *half-scale row. The real-facing boundary exposes no trichotomy witness, host real order, quotient stream equality, selected limit, or ambient completeness principle.*

Proof. Start with a real-facing consumer. By Definition 1502.3, the consumer cannot reach the maximum readback R until the finite sum, difference, absolute-value, post-absolute-value sum, and one-half scale rows have all been displayed.

The seal row E is downstream of R. The continuation rows first replay the aligned source windows and dyadic endpoints, then the sibling operation rows, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by the terminal $\mathsf{Real}^{\uparrow}$ seal.

Since Definition 1502.1 contains no coordinate for trichotomy, host real order, quotient stream equality, selected limits, or completeness, the real-facing boundary is exhausted by the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMaxUp.RegularCauchyMaxCarrier real seal boundary`

Theorem 1502.6 (RegularCauchyMax scoped lattice route)**.** *Every lattice-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier factors through the displayed concrete route*

$$\mathsf{RegularCauchySum}^{\uparrow}, \quad \mathsf{RegularCauchyAbs}^{\uparrow}, \quad \mathsf{RegularCauchyScale}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The scoped route is the sum-absolute-value maximum construction, its lattice absorption ledger, and the terminal Real seal boundary; it is not a trichotomy on completed reals, a host selected maximum, a quotient stream representative, or an ambient completeness argument.

Proof. Apply Definition 1502.3. Every maximum-facing read first passes from the two source names and aligned windows through the finite sum, difference, absolute-value, second-sum, and half-scale rows before any regular readback is exposed.

Next apply Definition 1502.4. Lattice reads such as repeated maximum consumption reuse the same displayed source rows and operation rows, so no extra branch for real trichotomy or a selected host maximum can enter the packet.

Finally apply Definition 1502.5. Any Real-facing consumer reaches $\mathsf{Real}^{\uparrow}$ only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback row assembled from $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, and $\mathsf{RegularCauchyScale}^{\uparrow}$ has been displayed. This composes the stated scoped lattice route and excludes quotient streams, selected limits, and ambient completeness coordinates. □

Theorem 1502.7 (RegularCauchyMax scoped dependency package)**.** *Every consumer route for an accepted* $\mathsf{RegularCauchyMax}^{\uparrow}$ *carrier is scoped exactly by* $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *In carrier notation, the only available dependency package is*

$$(S, \Delta, V, T, Q, R, E, H, C, P, N),$$

together with the two source rows A, B *and their displayed windows and dyadic endpoint observations. The package exposes no real trichotomy, selected host maximum, quotient stream equality, selected limit, or ambient completeness assumption.*

Proof. Apply Definition 1502.3. It forces every maximum-facing read to pass through the source sum, difference, absolute-value, second-sum, and half-scale rows before the regular readback row R is visible.

Next use Definition 1502.4. Lattice-facing reads reuse the same rows rather than adding a branch for host comparison or a selected maximum.

Finally apply Definition 1502.5. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of the same $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the structural rows H, C, P, N only transport, replay, source, and name that displayed dependency package. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMax}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMax}^{\uparrow}$ packet is generated from two RegSeqRat source rows, aligned StreamName windows, DyadicRatCore endpoint observations, RegularCauchySum rows for sum and difference assembly, a RegularCauchyAbs difference row, a RegularCauchyScale half-scale row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyMax}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1502.1, Definition 1502.2, Definition 1502.3, Definition 1502.4, and Definition 1502.5 bind the finite constructive maximum route through $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as a scoped Real-name construction.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyMax}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyMaxUp.RegularCauchyMaxTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy on real names, host real order, quotient stream equality, selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Public closure requires a reusable consumer-facing maximum interface and bridge coverage for the same finite Real-name route.

Chapter 1503

A Concrete Naming Certificate for $\mathsf{MetricDiameter}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{MetricDiameter}^{\uparrow}$ packet records the diameter interface of a bounded metric object as finite BEDC rows. The packet reads diameter through distance-comparison ledgers and displayed ordered-real bounds, not through a host supremum. It sits beside the metric carrier of chapter 57, the ordered-real comparison surface of chapter 766, the real-name seal of chapter 38, the center-radius rows of chapter 1450, and the entourage-facing boundary of chapter 339.

Definition 1503.1 (MetricDiameter carrier)**.** *A* $\mathsf{MetricDiameter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, d, B, W, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row with its point classifier and metric-source data.*
2. *d is the* $\mathsf{Real}^{\uparrow}$*-valued distance witness read from the metric carrier X.*
3. *B is the displayed bound row, read through the ordered-real comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$ *and the rational bound rows of* $\mathsf{Rat}^{\uparrow}$ *when a rational presentation is used.*
4. *W is the finite witness list of compared pairs. Each row names a pair (x_i, y_i) admitted by the point classifier of X.*
5. *R is the distance-comparison ledger. For every displayed pair in W, it records the ordered-real comparison from the distance row $d(x_i, y_i)$ to the bound row B.*
6. *H records componentwise* hsame *transport for the metric carrier, distance witness, bound row, witness list, and distance-comparison ledger.*
7. *C records the displayed* Cont *route replaying point admission, distance evaluation, ordered-real comparison, and bound handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricDiameter}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an ambient supremum, compactness argument, selected maximum pair, geodesic length structure, open-cover principle, or bridge datum.

Theorem 1503.2 (MetricDiameter bound obligation)**.** *For an accepted* $\mathsf{MetricDiameter}^{\uparrow}$ *carrier* $D = (X, d, B, W, R, H, C, P, N)$*, every displayed pairwise distance row in the witness list* W *is bounded by the named diameter row* B *through the ledger* R*. Concretely, each displayed pair* (x_i, y_i) *admitted by* X *has a distance row* $d(x_i, y_i)$*, and* R *records the ordered-real comparison*

$$d(x_i, y_i) \leq_{\mathsf{R}} B.$$

No pairwise-distance bound is accepted unless it is one of the finite rows replayed by C *and transported by* H*.*

Proof. Project the accepted carrier of Definition 1503.1. The rows X, d, B, W, R expose the metric carrier, distance witness, displayed bound, finite pair list, and comparison ledger.

For a listed pair (x_i, y_i), point admission is read from X, distance evaluation is read from d, and the comparison to B is read from R. Thus the bound has a displayed ordered-real route through $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

The rows H and C then preserve and replay the same route under classifier transport. Since the carrier exports no coordinate outside $X, d, B, W, R, H, C, P, N$, an unlisted pairwise-distance assertion is not part of the local certificate. □

Theorem 1503.3 (MetricDiameter classifier stability)**.** *Let* $D = (X, d, B, W, R, H, C, P, N)$ *be an accepted* $\mathsf{MetricDiameter}^{\uparrow}$ *carrier. Under any classifier transport admitted by* H*, the transported packet preserves the metric carrier, bound row, finite comparison ledger, and replay route:*

$$X, d, B, W, R, C \longmapsto X', d', B', W', R', C'.$$

The transported packet therefore compares exactly the transported finite pair list against the transported bound and introduces no second metric carrier, no independent diameter value, and no host supremum.

Proof. The transport row H is componentwise. It carries the metric carrier, distance witness, bound row, finite witness list, and comparison ledger to their transported rows without changing the packet shape.

The replay row C is downstream from the same transported coordinates. Point admission, distance evaluation, ordered-real comparison, and bound handoff are replayed after transport in the same order as before transport.

Package provenance P and the local naming row N identify this transported packet as the same $\mathsf{MetricDiameter}^{\uparrow}$ surface. Because no coordinate exists for a second metric carrier, independent diameter value, or host supremum, classifier stability preserves exactly the displayed ledger. □

Theorem 1503.4 (MetricDiameter NameCert obligation surface)**.** *For every accepted* $\mathsf{MetricDiameter}^{\uparrow}$ *carrier* $D = (X, d, B, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricDiameter}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed metric carrier, distance witness, bound row, witness list, comparison ledger, transport, replay, provenance, and local naming rows;*
2. *finite bound exposure, requiring the bound row* B *to be read through the ordered-real and rational surfaces already named by the packet;*
3. *pairwise distance comparison stability, requiring every finite pair in* W *to have its distance-to-bound comparison recorded in* R*;*

4. *classifier transport, requiring H and C to preserve the metric carrier, bound row, witness list, ledger, and replay route;*
5. *ledger non-escape, stating that the packet exports no host supremum, compactness argument, selected maximum pair, geodesic data, open-cover principle, or bridge check.*

Proof. Carrier admission is projection from Definition 1503.1. The tuple lists exactly the rows available to the local classifier and to downstream metric or ordered-real consumers.

Finite bound exposure is the role of B. It is consumed through $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Rat}^{\uparrow}$ rows named in P, so the diameter value is visible as a displayed comparison bound rather than as a supremum in an ambient order.

Pairwise comparison stability is the role of W and R. The witness list provides only finitely many displayed pairs, and the ledger records the distance-to-bound comparison for each such pair.

Classifier transport and replay are supplied by H and C. They move the same rows to a common comparison surface and replay the same finite route, while P and N record provenance and local naming. Hence every export remains inside the finite ledger and no excluded analytic object can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricDiameter}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricDiameter}^{\uparrow}$ packet is generated from a displayed MetricSpace carrier, a RealUp-valued distance witness, an ordered-real bound row, a finite witness list of compared pairs, a distance-comparison ledger, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricDiameter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1503.1, Definition 1503.2, Definition 1503.3, and Definition 1503.4 seed the finite ordered-real diameter surface for metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricDiameter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host supremum, compactness theorem, selected maximum pair, geodesic data, open-cover principle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bound exposure, pairwise distance comparison stability, classifier transport, and ledger non-escape for the same packet.

Chapter 1504

A Concrete Naming Certificate for $\mathsf{DedekindCompleteReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DedekindCompleteReal}^{\uparrow}$ packet records the finite interface by which a located, inhabited, bounded $\mathsf{Real}^{\uparrow}$ family may expose a displayed least-upper-bound witness. It reads the terminal real-name surface of chapter 38, the upper-bound and lower-approximation discipline of chapter 1427, the representation boundary of chapter 1424, and the sequence-limit ledger of chapter 1480. The packet is a located-supremum completion interface for finite BEDC consumers; it is not a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, not a quotient-real construction, not a trichotomy principle, and not a general completeness theorem for arbitrary real subsets.

Definition 1504.1 (DedekindCompleteReal source packet)**.** *A* $\mathsf{DedekindCompleteReal}^{\uparrow}$ *source packet is a finite* BHist *tuple*

$$D = (F, I, B, W, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the located* $\mathsf{Real}^{\uparrow}$ *family row whose members are readable only through finite real-name observations;*
2. *I is the inhabitedness ledger, displaying one accepted member witness for the family before any supremum read is requested;*
3. *B is the boundedness ledger, displaying an upper-bound witness and the comparison rows that place every inspected family observation below that bound;*
4. *W is the finite precision-window row that lists the exact family observations, rational tolerances, and comparison budgets currently in scope;*
5. *S is the located-supremum witness row, displaying an upper-bound face and a lower-approximation face for every precision listed in W;*
6. *Q is the* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *row that compares any cut-shaped observation used by S with the regular Cauchy readback accepted by* $\mathsf{Real}^{\uparrow}$*;*
7. *L is the* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *row that records the finite limit-law checks used when S is read through sequence approximants;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed supremum witness, reached only after F, I, B, W, S, Q, L have been exposed;*

9. *H records componentwise* hsame *transports for the family, inhabitedness ledger, boundedness ledger, precision window, supremum witness, comparison row, sequence-limit row, real seal, continuation routes, provenance, and local naming row;*
10. *C records the displayed* Cont *routes that replay family reads, boundedness checks, lower-approximation reads, comparison handoff, sequence-limit handoff, and real-seal readback;*
11. *P is the* Pkg *provenance over* $\mathsf{DedekindCompleteReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{RealSequenceLimitLaws}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCompleteReal}^{\uparrow}}$ *row.*

The accepted packet contains no quotient-real coordinate, no trichotomy oracle, no host real equality test, no countable-choice schedule, no arbitrary subset classifier, and no ambient completeness principle beyond the displayed located, inhabited, bounded family interface.

Theorem 1504.2 (DedekindCompleteReal NameCert obligation surface)**.** *For an accepted packet $D = (F, I, B, W, S, Q, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindCompleteReal}^{\uparrow}}$ *obligations are exactly the following finite rows:*

1. *carrier admission: the accepted object is the displayed tuple with located family data, inhabitedness, boundedness, precision windows, a located-supremum witness, comparison handoff, sequence-limit handoff, a real seal, transports, continuations, provenance, and local naming evidence;*
2. *inhabited bounded family stability: replacing F, I, B, W along the componentwise transports in H preserves the same inhabitedness witness and the same inspected upper-bound ledger;*
3. *located supremum witness: every precision request listed in W receives both an upper-bound read and a lower-approximation read from S, with no appeal to an unbounded family or a hidden subset classifier;*
4. *representation handoff: any cut-shaped observation used by S must pass through the comparison row Q before reaching the real-seal row E;*
5. *sequence-limit handoff: any sequence-shaped approximation used by S must pass through the finite limit-law row L before reaching E;*
6. *classifier transport: the transports in H and the replay routes in C move only the displayed rows $F, I, B, W, S, Q, L, E, P, N$;*
7. *ledger exhaustion: every admissible downstream consumer must read the packet through one of the displayed rows before obtaining a real-facing supremum observation.*

No local obligation may invoke quotient reals, trichotomy, host real equality, countable choice, an arbitrary subset classifier, or a general completeness principle.

Proof. Project the accepted carrier row by row. Carrier admission is immediate from the tuple D, while inhabited bounded family stability reads only F, I, B, W and the corresponding transports in H.

For the supremum witness, inspect a requested precision in W. The row S supplies the upper-bound face and the lower-approximation face for that precision, and the carrier contains no coordinate from which an unbounded-family argument or hidden subset classifier could be read.

The two handoff obligations are then forced by the route inventory. Cut-shaped observations pass through Q, sequence-shaped approximations pass through L, and both routes reach the real-facing surface only at E.

Classifier transport and ledger exhaustion are structural. The continuation row C can replay only the displayed routes, and the provenance row P names exactly the sibling BEDC surfaces available to the packet. Since quotient reals, trichotomy, host real equality, countable choice, arbitrary subset classification, and general completeness are absent from the carrier, they cannot enter the local NameCert surface. □

Theorem 1504.3 (DedekindCompleteReal located-supremum handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing observation exported by an accepted* $\mathsf{DedekindCompleteReal}^{\uparrow}$ *packet factors through the displayed located, inhabited, bounded family rows* F, I, B, W*, the located-supremum witness row* S*, the* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *handoff row* Q*, the* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *handoff row* L*, and the terminal real-seal row* E*. The export supplies a finite least-upper-bound witness for the named family window only, and it supplies no Dedekind-cut construction of* $\mathsf{Real}^{\uparrow}$*, quotient real, trichotomy principle, host real equality, countable-choice schedule, arbitrary subset classifier, or general completeness theorem.*

Proof. Begin with a real-facing consumer of the packet. The only real-name coordinate present in the carrier is E, and E is reachable only after the family row, inhabitedness ledger, boundedness ledger, precision window, and supremum witness row have been fixed.

The upper-bound side of the observation is supplied by B and S. The lower-approximation side is supplied by S at the requested precision in W, with inhabitedness grounded by I.

If the observation is read through a cut-shaped presentation, the comparison row Q is the only available handoff. If it is read through sequence approximants, the row L is the only available limit-law handoff.

Thus every real-facing observation factors through F, I, B, W, S, Q, L, E and then through the transport, continuation, provenance, and naming rows. The carrier inventory contains no Dedekind construction of $\mathsf{Real}^{\uparrow}$, quotient-real coordinate, trichotomy oracle, host equality test, choice schedule, arbitrary subset classifier, or general completeness coordinate, so none can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCompleteReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DedekindCompleteReal}^{\uparrow}$ packet is generated from a located $\mathsf{Real}^{\uparrow}$ family row, inhabitedness and boundedness ledgers, a finite precision window, a located-supremum witness, comparison and sequence-limit handoff rows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCompleteReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1504.1, Definition 1504.2, and Definition 1504.3 seed the located-supremum completion interface for finite inhabited bounded $\mathsf{Real}^{\uparrow}$ family windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCompleteReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient reals, trichotomy, host real equality, countable choice, arbitrary subset completeness, general least-upper-bound completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, inhabited bounded family stability, located supremum witnessing, comparison handoff, sequence-limit handoff, classifier transport, ledger exhaustion, and real-seal factorization for the displayed packet.

Chapter 1505

A Concrete Naming Certificate for $\mathsf{DedekindCompleteness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DedekindCompleteness}^{\uparrow}$ packet records the finite completeness route by which a displayed inhabited, bounded, located cut family exposes a real-name supremum witness. It sits over the located-cut surface of chapter 404, the located-real surface of **??**, the comparison discipline of chapter 1381, the stream and regular-rational readbacks of chapter 126 and chapter 301, and the terminal real seal of chapter 38. The packet names a concrete finite route for later $\mathsf{Real}^{\uparrow}$ consumers; it does not assert an ambient host completeness principle for arbitrary subsets.

Definition 1505.1 (Dedekind completeness carrier)**.** *A* $\mathsf{DedekindCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, B, W, A, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *family row whose lower and upper faces are inspected only through finite rational windows;*
2. *R is the* $\mathsf{LocatedReal}^{\uparrow}$ *readback row attached to the same family observations;*
3. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *row that records boundedness and comparison evidence between the cut, located-real, and real-seal faces;*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window ledger used to request finite observations from the family;*
5. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation ledger that repacks the windows as a regular rational route;*
6. *S is the located upper-bound stability row, displaying that transported family observations remain below the same accepted upper-bound witness;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, R, B, W, A, S have been displayed;*
8. *H records componentwise* hsame *transports for the cut family, located-real readback, comparison row, stream windows, regular approximation ledger, stability row, real seal, continuation routes, provenance, and local naming row;*
9. *C records the displayed* Cont *replay routes for family reads, upper-bound checks, lower-approximation witnesses, comparison handoff, regular-rational handoff, and real-seal readback;*

10. *P is the* Pkg *provenance over* DedekindCompleteness$^{\uparrow}$, LocatedCut$^{\uparrow}$, LocatedReal$^{\uparrow}$, BishopRealComparison$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* NameCert$_{\text{DedekindCompleteness}^{\uparrow}}$ *row.*

The accepted carrier contains no quotient-real coordinate, no trichotomy oracle, no host real equality test, no countable-choice schedule, no arbitrary subset classifier, and no completeness principle beyond the displayed finite packet.

Theorem 1505.2 (Dedekind completeness NameCert obligations)**.** *For an accepted carrier $K = (L, R, B, W, A, S, E, H, C, P, N)$, the local* NameCert$_{\text{DedekindCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure: every consumer can inspect precisely the displayed rows $L, R, B, W, A, S, E, H, C, P, N$;*
2. *classifier transport: the transports in H and replay routes in C preserve the same cut-family classifier, located-real readback, comparison ledger, stream windows, regular approximation ledger, upper-bound stability row, real seal, provenance, and local naming row;*
3. *located upper-bound stability: the row S transports accepted upper-bound evidence through the* BishopRealComparison$^{\uparrow}$ *comparison row without changing the family window named by L and W;*
4. *lower-approximation witness: every precision requested in W receives a displayed lower-face witness from the located cut row L and its regular-rational handoff A;*
5. Real$^{\uparrow}$ *seal factorization: every terminal real-name read factors through L, R, B, W, A, S before it reaches E;*
6. *non-escape: no consumer can obtain quotient reals, trichotomy, host real equality, countable choice, an arbitrary subset classifier, or ambient completeness from the packet.*

Proof. Project the accepted carrier row by row. Carrier exposure is the tuple inventory itself, and classifier transport follows because H and C mention only the displayed rows.

For located upper-bound stability, read S together with the comparison row B. The transported family observations remain tied to the same cut window L and stream requests W, so no separate upper-bound classifier is introduced.

For lower approximation, inspect a requested precision in W. The located cut row L supplies the lower face, and the regular-rational row A repacks that witness into the same finite route consumed by the real-seal boundary.

The real-facing exit is then forced. The only terminal real row is E, and the carrier reaches it only after the cut, located-real, comparison, stream, approximation, and stability rows have been displayed.

The non-escape clause is structural. Quotient reals, trichotomy, host real equality, countable choice, arbitrary subset classifiers, and ambient completeness are not carrier coordinates and are not named by the package provenance P. □

Closure status for NameCert$_{\text{DedekindCompleteness}^{\uparrow}}$**.**

Constructive story: A DedekindCompleteness$^{\uparrow}$ packet is generated from displayed LocatedCut$^{\uparrow}$, LocatedReal$^{\uparrow}$, BishopRealComparison$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ rows, together with located upper-bound stability, lower-approximation witnessing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DedekindCompleteness}^{\uparrow}}$, seed)

Scope closed: Definition 1505.1 and Definition 1505.2 seed the finite Dedekind-completeness route from located cut data to a terminal Real$^{\uparrow}$ seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{DedekindCompleteness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient reals, trichotomy, host real equality, countable choice, arbitrary subset completeness, general least-upper-bound completeness, or a standard bridge.

Upgrade path: Formal closure requires checked carrier exposure, classifier transport, located upper-bound stability, lower-approximation witnessing, Real$^{\uparrow}$ seal factorization, and non-escape for the displayed packet.

Chapter 1506

A Concrete Naming Certificate for $\mathsf{MetricCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Kelley-style metric completion, the $\mathsf{MetricCompletion}^{\uparrow}$ packet records the L10 metric-completion interface as finite Cauchy-filter or Cauchy-net evidence over a displayed metric source. It sits between the metric carrier of chapter 57, the filter surface of chapter 333, the directed-net surface of chapter 589, the separated-limit uniqueness surface of chapter 1411, and the uniform-completion boundary of chapter 1473. The packet gives later complete-metric and real-completion consumers a named landing surface without quotienting Cauchy names, selecting representatives, assuming an ambient complete space, invoking open-cover compactness, using countable choice, or asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1506.1 (MetricCompletion carrier)**.** *A* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, D, R, S, H, C, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, including its point classifier, metric-distance witness surface, and displayed finite observation histories.*
2. *F is either a* $\mathsf{CauchyFilter}^{\uparrow}$ *cofinal packet or an empty marker when the completion route is presented by directed nets.*
3. *D is either a* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window packet or an empty marker when the completion route is presented by Cauchy filters.*
4. *R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row used when a completion-facing consumer asks for numerical tolerance data.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness row, used only to identify two displayed limit readings through zero-distance classifier evidence.*
6. *H records componentwise* hsame *transport for the source metric, cofinal Cauchy packet, finite numerical readback, separatedness row, replay row, package row, and local naming row.*
7. *C records the displayed* Cont *route replaying metric admission, Cauchy cofinality, readback extraction, separated-limit comparison, and completion handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletion}^\uparrow}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, an ambient complete metric space, open-cover compactness, countable choice, or completeness of $\mathsf{Real}^\uparrow$*.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.TasteGate.MetricCompletionTasteGate single carrier alignment`

Definition 1506.2 (MetricCompletion classifier)**.** *The classifier of an accepted* $\mathsf{MetricCompletion}^\uparrow$ *packet relates two completion-facing reads precisely when their source metric row is the same displayed X, their cofinal Cauchy evidence is read from the same F or D branch, their finite* $\mathsf{RegSeqRat}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *tolerance rows agree after the componentwise* hsame *transport H, and the separatedness row S identifies any two displayed limit readings through the zero-distance route already available in* $\mathsf{SeparatedMetric}^\uparrow$*.*

Equivalently, the classifier is the finite replay path

$$X, (F \text{ or } D), R, S, H, C, P, N.$$

It is not a quotient of Cauchy names, not a choice of canonical representatives, and not a claim that a host metric space is complete.

Lean4 (definition) `BEDC.Derived.MetricCompletionUp.ClassifierRoute.MetricCompletionClassifier`

Theorem 1506.3 (MetricCompletion NameCert obligations)**.** *For an accepted* $\mathsf{MetricCompletion}^\uparrow$ *carrier $M = (X, F, D, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^\uparrow}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed* $\mathsf{Met}^\uparrow$ *source,* $\mathsf{CauchyFilter}^\uparrow$ *or* $\mathsf{CauchyNet}^\uparrow$ *cofinal packet, finite* $\mathsf{RegSeqRat}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *readback,* $\mathsf{SeparatedMetric}^\uparrow$ *uniqueness row, componentwise transport, continuation replay, provenance row, and local naming row;*
2. *cofinal evidence selection, requiring each completion-facing read to pass through exactly one visible branch, either F or D;*
3. *finite readback compatibility, requiring numerical tolerance requests to factor through R before any completion handoff is consumed;*
4. *separated-limit uniqueness, requiring equality of two displayed limit readings to pass through the zero-distance classifier route stored in S;*
5. *ledger non-escape, stating that no quotient completion, selected representative, ambient complete metric space, open-cover compactness principle, countable-choice schedule, or* $\mathsf{Real}^\uparrow$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.MetricCompletionUp StdBridge`

Proof. Carrier admission is projection from Definition 1506.1. The tuple lists the source metric row, the two possible Cauchy-evidence branches, the finite numerical readback, the separatedness row, and the structural support rows.

Cofinal evidence selection is local because the accepted carrier records the chosen branch in F or D. A filter-presented route consumes the $\mathsf{CauchyFilter}^\uparrow$ row before any completion handoff, while a net-presented route consumes the $\mathsf{CauchyNet}^\uparrow$ directed-window row in the same position.

Finite readback compatibility is the role of R. Every tolerance-facing consumer reads the displayed $\mathsf{RegSeqRat}^\uparrow$ or $\mathsf{Real}^\uparrow$ data before it can compare completion endpoints.

Separated-limit uniqueness is supplied by S and replayed by C after transport by H. The provenance row P and local naming row N record that no further coordinate is available, so the excluded quotient, choice, compactness, ambient completeness, and real-completeness principles cannot be read from the certificate. □

Theorem 1506.4 (MetricCompletion Cauchy-filter handoff)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet whose cofinal evidence is presented by the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch F. Then every completion-facing consumer of M factors first through the source metric row X, then through the displayed Cauchy-filter row F, then through the finite readback row R when numerical tolerance data are requested, and only then through the separatedness row S for comparison of displayed limit readings.*

The handoff is therefore the finite route

$$X \longmapsto F \longmapsto R \longmapsto S \longmapsto H, C, P, N,$$

not a quotient completion, not a selected representative, not an ambient complete metric space, and not an appeal to countable choice or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.NameCertObligations.MetricCompletionCarrier filter branch handoff`

Proof. Project the accepted packet and choose the filter branch. The first row available to a completion consumer is the metric source X, so point admission and distance evidence are read before any completion-facing operation.

The next row is F. It is already a $\mathsf{CauchyFilter}^{\uparrow}$ finite-filterbase and late-window packet, so cofinality is consumed as displayed filter evidence rather than as a selected Cauchy representative.

When the consumer asks for tolerance data, the route passes through R. The replay row C then carries the same source, filter, and readback rows to S, where separated-limit comparison is performed only through the local zero-distance classifier route.

The remaining rows H, P, N transport and name the same finite packet. Since the route has no coordinate for quotient equality, selected representatives, ambient completeness, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, none of those principles is exported by the handoff. □

Theorem 1506.5 (MetricCompletion separated limit uniqueness)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet, and suppose two completion-facing reads of the same source metric packet use the same displayed cofinal branch F or D and the same finite readback row R after transport by H. If the separatedness row S displays zero distance between the two resulting limit readings, then the* $\mathsf{MetricCompletion}^{\uparrow}$ *classifier identifies the two reads.*

This identification is internal to the displayed $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{SeparatedMetric}^{\uparrow}$ *rows. It is not host equality, not equality of quotient classes, and not a selected-limit principle.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.SeparatedLimit.MetricCompletion separated limit uniqueness`

Proof. Read the two completion-facing paths through the common source metric row X. The shared cofinal branch, either F or D, keeps both paths on the same displayed Cauchy evidence.

Transport by H aligns the finite readback row R for the two paths. The replay row C then presents the two limit readings to the separatedness row S.

By the meaning of S, zero distance between the two displayed limit readings yields classifier equivalence inside the underlying metric carrier. The definition of Definition 1506.2 uses exactly this route as the comparison criterion.

No quotient class, host equality, selected representative, ambient completeness row, countable-choice schedule, or $\mathsf{Real}^{\uparrow}$ completeness theorem appears among the fields of M, so the uniqueness conclusion remains a local classifier identification. □

Theorem 1506.6 (MetricCompletion Cauchy-net handoff)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet whose cofinal evidence is presented by the* $\mathsf{CauchyNet}^{\uparrow}$ *branch D. Then every completion-facing consumer of M factors first through the source metric row X, then through the displayed directed Cauchy-net row D, then through the finite readback row R when numerical tolerance data are requested, and only then through the separatedness row S for comparison of displayed limit readings.*

The handoff is therefore the finite route

$$X \longmapsto D \longmapsto R \longmapsto S \longmapsto H, C, P, N,$$

not a quotient completion, not a selected representative, not an ambient complete metric space, and not an appeal to countable choice or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.HandoffRoutes.MetricCompletion net branch handoff`

Proof. Project the accepted packet and choose the net branch. The source metric row X supplies the displayed point classifier, metric-distance witness surface, and finite observation histories before any completion-facing read is available.

The next row is D. It is already the $\mathsf{CauchyNet}^{\uparrow}$ directed-window packet, so cofinality is consumed as visible net evidence rather than as a chosen representative of a completion class.

When a consumer requests tolerance data, the route passes through R. The replay row C then carries the same source, net, and readback rows to S, where separated-limit comparison is performed through the local zero-distance classifier route.

The rows H, P, N transport, package, and name the same finite packet. Since none of the carrier coordinates is a quotient-completion equality, a selected limit, a countable-choice schedule, an ambient completeness witness, or a $\mathsf{Real}^{\uparrow}$ completeness theorem, the net handoff exports none of those principles. □

Theorem 1506.7 (MetricCompletion filter-net branch determinacy)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Every completion-facing read of M uses exactly one visible cofinal branch, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F or the* $\mathsf{CauchyNet}^{\uparrow}$ *row D. A downstream read cannot mix filter evidence and net evidence unless an additional branch-conversion row is added, and no such row occurs in the carrier of Definition 1506.1.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchDeterminacy.MetricCompletion filter net branch determinacy`

Proof. The classifier of Definition 1506.2 reads the cofinal evidence from the single displayed branch recorded in the packet, written there as F or D. Thus the accepted route selects one branch before the finite readback row R and separatedness row S are inspected.

In the filter case, every downstream comparison factors through the row F named in the finite route. In the net case, the corresponding route factors through D in the same position.

A mixed read would require a new row relating the filter and net evidence before R and S could be shared as one classifier path. The carrier has no coordinate for such a row, so the classifier route cannot manufacture it downstream. □

Theorem 1506.8 (MetricCompletion finite-readback ledger non-escape)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier* $M = (X, F, D, R, S, H, C, P, N)$*, every tolerance read, readback request, and limit-comparison read exported by* M *is exhausted by the rows*

$$X, F, D, R, S, H, C, P, N.$$

Consequently the packet cannot introduce quotient classes, host equality, selected limits, open-cover compactness, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Tolerance-facing reads pass through the finite readback row R after the source metric row X and the chosen cofinal branch F or D have been displayed. The componentwise transport row H and replay row C preserve that same route.

Limit-comparison reads then pass through S. The separatedness row supplies only local zero-distance classifier evidence, so it compares displayed limit readings without changing the carrier into a host equality or quotient equality.

The provenance row P and local naming row N close the ledger at exactly the listed rows. Since no remaining coordinate names quotient classes, selected limits, open-cover compactness, countable choice, host equality, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, none of those data can escape from the finite readback ledger. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.ReadbackLedger.MetricCompletion finite readback ledger nonescape`

Theorem 1506.9 (MetricCompletion branch coverage)**.** *Let*

$$M = (X, F, D, R, S, H, C, P, N)$$

be an accepted $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Every completion-facing read of* M *uses either the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch* F *or the* $\mathsf{CauchyNet}^{\uparrow}$ *branch* D *before any finite readback, separated-limit comparison, or completion handoff is inspected. A consumer that has not chosen one of these displayed branches has no route to* R*,* S*,* H*,* C*,* P*, or* N*.*

Proof. Project the carrier of Definition 1506.1. The only cofinal evidence fields are F and D, and the classifier of Definition 1506.2 reads completion data through one of those fields before it reaches the readback row.

The existing handoff theorems identify the two possible routes. In the filter case, Definition 1506.4 forces the path through F before R and S are read. In the net case, Definition 1506.6 forces the same position to be occupied by D.

The remaining rows are structural. Transport, replay, provenance, and local naming preserve the chosen branch; they do not create a third Cauchy-evidence coordinate. Hence branch coverage is exactly the displayed filter-or-net dichotomy. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchCoverage.MetricCompletion branch coverage`

Theorem 1506.10 (MetricCompletion filter-net compatibility)**.** *Let* $M = (X, F, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Any consumer comparing filter and net presentations of the same completion-facing read must pass through the same source* $\mathsf{Met}^{\uparrow}$ *row* X*, the same finite readback row* R*, the same* $\mathsf{SeparatedMetric}^{\uparrow}$ *row* S*, the componentwise* hsame *transport row* H*, the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *row* N*. The comparison cannot be made by quotienting the two Cauchy presentations or by selecting a representative common limit.*

Proof. Start with the branch coverage of Definition 1506.9. A filter presentation and a net presentation are readable only as the two visible cofinal branches of the same carrier, so any comparison between them must keep the source row X fixed.

Finite numerical data are then compared through R. The finite-readback ledger non-escape theorem, Definition 1506.8, prevents a consumer from replacing this row by an external tolerance oracle or a quotient-class read.

Separated comparison is performed at S. The separated-limit uniqueness theorem, Definition 1506.5, supplies only local zero-distance classifier evidence, and H, C, P, N preserve and name that same route.

Since no carrier coordinate stores quotient equality or a selected representative, the filter-net comparison is exhausted by the listed rows. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.Compatibility.MetricCompletion filter net compatibility`

Theorem 1506.11 (MetricCompletion source-branch readback)**.** *Every accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *read factors through the common source metric row X, exactly one displayed Cauchy branch F or D, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row R, the separatedness row S, and the structural rows H, C, P, N. In carrier notation the route is*

$$X, (F \text{ or } D), R, S, H, C, P, N.$$

The readback cannot be replaced by quotient completion equality, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1506.1. The visible source row is X, the only Cauchy evidence rows are the filter branch F and the net branch D, and the only numerical readback coordinate is R.

Use branch coverage from Definition 1506.9. A completion-facing request must choose one of the displayed branches before it can read the finite $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row.

Use filter-net compatibility from Definition 1506.10. It keeps the selected branch attached to the same source metric, the same readback row, the same separatedness row, and the same transport, replay, provenance, and naming route.

Finally apply the finite-readback ledger non-escape theorem, Definition 1506.8. Since quotient equality, selected representatives, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in the packet, no source-branch readback can be replaced by any of those objects. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.MetricCompletion source branch readback`

Theorem 1506.12 (MetricCompletion filter-net branch exhaustion)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier $M = (X, F, D, R, S, H, C, P, N)$, every completion-facing consumer reads exactly one of the displayed Cauchy evidence branches before readback: either the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch F or the* $\mathsf{CauchyNet}^{\uparrow}$ *branch D. The remaining rows R, S, H, C, P, N can transport, replay, compare, package, and name the selected route, but they cannot introduce a third completion evidence route.*

Proof. Project the carrier of Definition 1506.1. The only Cauchy evidence rows in the accepted tuple are F and D, and the classifier of Definition 1506.2 reads one of them before the finite readback row R is available.

Use the NameCert obligations together with branch determinacy and compatibility, Definition 1506.3, Definition 1506.7, and Definition 1506.10. These keep the selected branch attached to

the same source metric, readback row, separatedness comparison, transport, replay, provenance, and local naming route.

It remains to exclude a hidden third route. The finite-readback ledger non-escape theorem, Definition 1506.8, rules out quotient completions, selected representatives, ambient complete spaces, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness. Since R, S, H, C, P, N are structural or readback rows downstream of the chosen branch, they cannot supply independent Cauchy evidence. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchExhaustion.MetricCompletion filter net branch exhaustion`

Theorem 1506.13 (MetricCompletion UniformCompletion handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet reads the packet only through the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *evidence, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness row, componentwise transport, replay, provenance, and local naming. The handoff provides no quotient row, no selected representative row, no host equality row, and no* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.MetricCompletionUniformCompletionHandoff`

Proof. Use Definition 1506.9 to select the visible Cauchy-evidence branch. The $\mathsf{UniformCompletion}^{\uparrow}$ consumer therefore begins with the same filter or net route already displayed by the metric-completion packet.

Next the consumer reads finite tolerance information through R. This is the only $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback coordinate in the carrier, so every numerical request remains attached to the same source metric and branch evidence.

Finally the consumer reaches S, H, C, P, N. The separatedness row supplies local uniqueness, while the structural rows transport, replay, source, and name the same finite packet.

The excluded completion principles are absent from Definition 1506.1 and are barred again by Definition 1506.8. The uniform-completion handoff therefore exports exactly the displayed finite evidence and no quotient or selected-representative data. □

Theorem 1506.14 (MetricCompletion obligation branch readback)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$*, every finite readback exported by the metric-completion surface first chooses one displayed cofinal branch, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch* F *or the* $\mathsf{CauchyNet}^{\uparrow}$ *branch* D*. Only after that branch is fixed may a consumer read the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row* R*.*

Proof. Use Definition 1506.9. It states that every completion-facing read chooses one of the visible Cauchy-evidence branches before any finite readback, separated comparison, or completion handoff is inspected.

The carrier of Definition 1506.1 has only one readback coordinate R. Thus a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ tolerance read can occur only after the selected filter or net branch has been displayed.

The filter-net compatibility theorem, Definition 1506.10, keeps R, H, C, P, N attached to the same source metric and branch route. The finite-readback non-escape theorem, Definition 1506.8, excludes quotient classes, selected limits, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness from the readback path. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchReadback.MetricCompletion obligation branch readback`

Theorem 1506.15 (MetricCompletion obligation separated comparison)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Any two displayed completion reads of M compare only through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier row S together with the componentwise* hsame, Cont, Pkg, *and* NameCert *route. They are not compared by host equality, quotient equality, or a selected representative of a completion class.*

Proof. Start from the classifier of Definition 1506.2. It identifies completion-facing reads only when the source metric, selected Cauchy branch, readback row, separatedness row, and structural routes agree componentwise.

Apply Definition 1506.5. The zero-distance evidence in S is the only comparison row that can identify two displayed limit readings.

The carrier and finite-readback non-escape theorem rule out host equality, quotient equality, selected representatives, ambient completeness, and countable choice. Therefore every accepted comparison remains a local $\mathsf{SeparatedMetric}^{\uparrow}$ classifier comparison transported through H, C, P, N. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchReadback.MetricCompletion obligation separated comparison`

Theorem 1506.16 (MetricCompletion regular-Cauchy readback)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Any completion consumer that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row first chooses one visible Cauchy evidence branch, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F or the* $\mathsf{CauchyNet}^{\uparrow}$ *row D. Only after that branch choice may the consumer read R, and every such readback is replayed through H, C, P, N over the same metric source X.*

Proof. Project the carrier of Definition 1506.1. The only Cauchy evidence rows are F and D, and the only numerical readback row is R.

Apply the branch coverage and branch determinacy theorems, Definition 1506.9 and Definition 1506.7. Together they force every completion-facing read to select exactly one displayed branch before the finite readback row can be inspected.

Now apply the obligation branch-readback theorem Definition 1506.14. It places the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ tolerance row after the selected filter or net evidence.

The support rows do not alter the branch. The componentwise transport, replay, provenance, and local naming rows H, C, P, N preserve the same source metric, selected Cauchy evidence, and readback row. Thus no consumer reaches R before a visible branch choice. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RegularCauchyReadback.MetricCompletion regular cauchy readback`

Theorem 1506.17 (MetricCompletion separated-branch readback)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Every completion consumer that reaches separated comparison first reads the selected* $\mathsf{CauchyFilter}^{\uparrow}$ *branch F or* $\mathsf{CauchyNet}^{\uparrow}$ *branch D, then the finite readback row R, and only then the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row S. The separated comparison is therefore a branch-indexed local zero-distance classifier route, not host equality, quotient equality, selected representative equality, or ambient completeness.*

Proof. Start with the regular-Cauchy readback theorem above. It fixes the route to R as a route through exactly one displayed Cauchy evidence branch over the source metric row X.

Apply Definition 1506.15. It states that any comparison of displayed completion reads uses the separatedness row S together with the componentwise transport, continuation, provenance, and naming route.

The separated-limit uniqueness theorem, Definition 1506.5, identifies the content of S as local zero-distance classifier evidence. This comparison is downstream of the selected branch and readback rows, not an independent completion equality.

Finally use Definition 1506.8. Quotient equality, host equality, selected representatives, ambient complete spaces, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, so separated comparison cannot be reached through any route other than the displayed branch, readback, and S, H, C, P, N rows. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.SeparatedBranchReadback.MetricCompletion separated branch readback`

Theorem 1506.18 (MetricCompletion obligation uniform handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet sees exactly the same displayed branch, finite readback row, separatedness row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *route as the metric-completion packet itself. The handoff adds no quotient completion row, no branch conversion row, no selected limit, and no independent uniform-completion classifier.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchReadback.MetricCompletion obligation uniform handoff`

Proof. Use Definition 1506.13. The uniform consumer reads the displayed $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyNet}^{\uparrow}$ evidence, R, S, and the structural rows H, C, P, N.

The branch-readback obligation above fixes that the same branch precedes the same readback row. The separated-comparison obligation fixes that any comparison available to the uniform consumer is still routed through S rather than through host or quotient equality.

Finally Definition 1506.10 and Definition 1506.8 keep the handoff on the same source metric, readback, separatedness, transport, replay, provenance, and naming route. No additional uniform-completion data is introduced by the handoff. □

Theorem 1506.19 (MetricCompletion branch-conversion refusal)**.** *Let* $M = (X, F, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. Any completion-facing read that uses data from both the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch* F *and the* $\mathsf{CauchyNet}^{\uparrow}$ *branch* D *must display a branch-conversion row between those two branches. Since the carrier of Definition 1506.1 contains no such row, no accepted read of* M *may mix filter evidence and net evidence as one completion route.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchReadback.MetricCompletion branch conversion refusal`

Proof. Apply Definition 1506.7. The accepted packet selects the visible branch before a read reaches the readback row R or the separatedness row S.

Now apply Definition 1506.9. A consumer that reads F follows the filter handoff, while a consumer that reads D follows the net handoff. To combine the two as one route, the consumer would need an additional row witnessing conversion from one displayed branch to the other.

The carrier inventory in Definition 1506.1 lists only $X, F, D, R, S, H, C, P, N$. The structural rows transport, replay, source, and name the selected route; they do not add a conversion coordinate. Hence a mixed read has no projection from the accepted packet. □

Theorem 1506.20 (MetricCompletion uniform-seal factorization)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet factors through the selected* $\mathsf{CauchyFilter}^{\uparrow}$ *branch*

F or $\mathsf{CauchyNet}^{\uparrow}$ *branch D, the finite readback row R, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row S, the componentwise* hsame *row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *row N. No uniform-facing seal is available outside that selected branch and structural route.*

Proof. Start with Definition 1506.18. It gives the uniform consumer exactly the branch, readback, separatedness, transport, replay, provenance, and naming rows of the metric-completion packet.

Use Definition 1506.14 to place R after the selected Cauchy-evidence branch. Then use Definition 1506.15 to place every comparison through S and the structural rows H, C, P, N.

The branch-conversion refusal above prevents the uniform consumer from combining F and D through an absent conversion row. Thus the uniform seal factors through one selected branch together with R, S, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.UniformSealFactorization.MetricCompletion uniform seal factorization`

Theorem 1506.21 (MetricCompletion completion-consumer totality)**.** *Every current completion-facing consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet receives only the rows listed in the carrier*

$$X, F, D, R, S, H, C, P, N.$$

It receives no quotient-completion class, selected representative, countable-choice schedule, branch-conversion coordinate, open-cover compactness row, or $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CompletionConsumerTotality.MetricCompletion completion consumer totality`

Proof. Project the carrier of Definition 1506.1. The finite packet lists the metric source, two alternative Cauchy-evidence rows, one readback row, one separatedness row, and the transport, replay, provenance, and naming rows.

The obligation theorems above exhaust the possible consumer routes. Branch readback fixes how R is reached; separated comparison fixes how S is used; uniform handoff and uniform-seal factorization fix the completion-facing handoff; branch-conversion refusal rejects mixed branches without an explicit row.

Finally apply Definition 1506.8. The excluded quotient, representative, countable-choice, compactness, and ambient-completeness principles have no coordinate in the accepted tuple and cannot be recovered by the structural rows. □

Theorem 1506.22 (MetricCompletion carrier admission)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet, carrier admission is exactly projection of the finite rows*

$$X, F, D, R, S, H, C, P, N.$$

The admitted rows are the displayed $\mathsf{Met}^{\uparrow}$ *source X, the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch F or* $\mathsf{CauchyNet}^{\uparrow}$ *branch D, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row R, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row S, the componentwise* hsame *transport row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *row N. No carrier admission judgment for this packet has a coordinate for quotient completion equality, selected representatives, host equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CarrierAdmission.MetricCompletion carrier admission`

Proof. Project the carrier of Definition 1506.1. Its displayed tuple is exactly $M = (X, F, D, R, S, H, C, P, N)$.

The row inventory is already separated by the NameCert obligations of Definition 1506.3. The branch coverage theorem, Definition 1506.9, forces completion-facing reads to choose F or D before R and S can be inspected.

The finite-readback ledger non-escape theorem, Definition 1506.8, closes the negative side. Since every exported tolerance read, comparison read, and handoff read is exhausted by the displayed tuple, no excluded completion principle can be admitted as an additional carrier coordinate. □

Theorem 1506.23 (MetricCompletion classifier stability)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ and $M' = (X', F', D', R', S', H', C', P', N')$ be accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packets whose displayed rows are related componentwise by the* hsame *transport rows and replayed by the corresponding* Cont *rows. Then the* $\mathsf{MetricCompletion}^{\uparrow}$ *classifier preserves branch choice, finite readback, separated comparison, and completion handoff along that componentwise route.*

In particular, a classified replacement cannot change a filter branch into an unrelated net branch, replace R by an external tolerance oracle, compare endpoints outside S, or introduce a new $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row.*

Proof. Start with the classifier of Definition 1506.2. It reads the source metric, the chosen Cauchy branch, the finite readback row, the separatedness row, and the support rows as one displayed route.

Apply branch determinacy and branch coverage, Definition 1506.7 and Definition 1506.9. These results keep the branch position fixed as either F or D before any downstream read reaches R or S.

Next apply Definition 1506.14 and Definition 1506.15. The first keeps tolerance evidence at the transported readback row, and the second keeps endpoint comparison at the transported separatedness row.

Finally use Definition 1506.18. The $\mathsf{UniformCompletion}^{\uparrow}$ consumer sees the same displayed branch, readback, separatedness, transport, replay, provenance, and naming rows, so classifier transport preserves the handoff rather than adding an independent completion surface. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.ClassifierStability.MetricCompletion classifier stability`

Definition 1506.24 (MetricCompletion branch-interface scope)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet $M = (X, F, D, R, S, H, C, P, N)$, the branch-interface rows are the displayed source metric row X, exactly one Cauchy-evidence branch F or D, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row R, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row S, the componentwise* hsame *transport row H, the displayed* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *row N. This scope is a row boundary: it names what filter and net consumers may read, and it does not add carrier/classifier transport beyond the accepted packet.*

Lean4 (definition) `BEDC.Derived.MetricCompletionUp.BranchInterfaceScope.MetricCompletionBranchInterfaceSco`

Theorem 1506.25 (MetricCompletion filter-branch scope)**.** *On the filter branch of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet, every completion-facing read consumes the rows in the order*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \text{ or } \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Equivalently, in the carrier notation it reads X, then F, then R, then S, and finally H, C, P, N. It has no route to a net branch, quotient completion row, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the branch-interface scope of Definition 1506.24. The filter case selects the displayed row F after the source metric row X.

Apply the branch-readback and separated-comparison obligations, Definition 1506.14 and Definition 1506.15. These place R after the chosen filter branch and place comparison at S together with H, C, P, N.

Finally use the filter handoff and finite-readback ledger non-escape theorems, Definition 1506.4 and Definition 1506.8. The selected route remains X, F, R, S, H, C, P, N, and no absent completion principle is introduced downstream. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchScope.MetricCompletion filter branch scope`

Theorem 1506.26 (MetricCompletion net-branch scope)**.** *On the net branch of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet, every completion-facing read consumes the rows in the order*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}\ \textit{or}\ \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Equivalently, in the carrier notation it reads X, then D, then R, then S, and finally H, C, P, N. It has no route to a filter branch, quotient completion row, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the branch-interface scope of Definition 1506.24. The net case selects the displayed row D after the source metric row X.

Apply the branch-readback and separated-comparison obligations, Definition 1506.14 and Definition 1506.15. These place R after the chosen net branch and place comparison at S together with H, C, P, N.

Finally use the net handoff and finite-readback ledger non-escape theorems, Definition 1506.6 and Definition 1506.8. The selected route remains X, D, R, S, H, C, P, N, and no absent completion principle is introduced downstream. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BranchScope.MetricCompletion net branch scope`

Theorem 1506.27 (MetricCompletion Cauchy-evidence handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet reaches separated comparison only after the exact Cauchy-evidence handoff*

$$X \longmapsto (F\ \textit{or}\ D) \longmapsto R \longmapsto S.$$

On the filter branch this is the $\mathsf{CauchyFilter}^{\uparrow}$ *route of Definition 1506.4; on the net branch this is the* $\mathsf{CauchyNet}^{\uparrow}$ *route of Definition 1506.6. The row R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback used before separated comparison, and the structural rows H, C, P, N only transport, replay, source, and name the same route.*

Proof. Use Definition 1506.9. A consumer that has not chosen the filter branch F or the net branch D has no route to readback, separated comparison, or completion handoff.

In the filter case, apply Definition 1506.4. The route is X to F, then to R, then to S, with the support rows preserving the same displayed evidence.

In the net case, apply Definition 1506.6. The route is X to D, then to R, then to S, again with only the support rows available downstream.

The finite-readback ledger non-escape theorem and the UniformCompletion handoff theorem, Definition 1506.8 and Definition 1506.13, prevent the consumer from replacing this sequence by quotient completion data, selected representatives, host equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CauchyEvidenceHandoff.MetricCompletion cauchy evidence handoff`

Theorem 1506.28 (Metric completion Cauchy-filter/net seal)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* MetricCompletion$^{\uparrow}$ *packet. Every accepted completion-facing seal exposes either* CauchyFilter$^{\uparrow}$ *evidence through F or* CauchyNet$^{\uparrow}$ *evidence through D over the same displayed metric source X, and then reads the finite tolerance row R, separatedness row S, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The seal does not quotient Cauchy names, select a representative completion point, or replace X by an ambient metric source.*

Proof. Project the carrier of Definition 1506.1. The only metric source is X, and the only Cauchy evidence rows are the filter branch F and the net branch D.

Use branch coverage and branch determinacy, Definition 1506.9 and Definition 1506.7. These force every completion-facing seal to choose the displayed filter branch or the displayed net branch before any read reaches R or S.

The local handoff theorems then identify the two allowed routes. Apply Definition 1506.4 in the filter case and Definition 1506.6 in the net case.

Finally use Definition 1506.8 and Definition 1506.19. The structural rows H, C, P, N preserve and name the same selected route, while quotient names, representative selection, and ambient metric replacement have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CauchyFilterNetSeal.MetricCompletion cauchy filter net seal`

Theorem 1506.29 (Metric completion separated-limit uniqueness boundary)**.** *For an accepted* MetricCompletion$^{\uparrow}$ *packet $M = (X, F, D, R, S, H, C, P, N)$, separatedness is the only uniqueness route exported to completion-facing consumers. Any identification of two displayed completion reads must pass through the* SeparatedMetric$^{\uparrow}$ *row S together with H, C, P, N, and no representative choice, quotient equality, host equality, or selected limit is exported by the packet.*

Proof. Start from the classifier of Definition 1506.2. It identifies reads only through the displayed metric source, selected Cauchy branch, finite readback, separatedness row, and structural rows.

Apply Definition 1506.5. The zero-distance evidence stored in S is the route by which two displayed completion readings may be identified.

The comparison cannot move to host equality or quotient equality. The finite-readback ledger non-escape theorem, Definition 1506.8, excludes quotient classes, selected limits, representative choice, countable choice, and ambient completeness from the exported rows.

Thus uniqueness remains a local separatedness boundary: S supplies the comparison, and H, C, P, N transport, replay, source, and name that same comparison without adding any representative selector. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.SeparatedLimitBoundary.MetricCompletion separated limit uniqueness boundary`

Theorem 1506.30 (Metric completion uniform-completion handoff)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer of an accepted* MetricCompletion$^{\uparrow}$ *packet consumes only the displayed metric source, the selected* CauchyFilter$^{\uparrow}$ *or* CauchyNet$^{\uparrow}$ *row, the finite readback row, separatedness row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. In the carrier notation this is exactly*

$$X, (F \text{ or } D), R, S, H, C, P, N.$$

The handoff exports no quotient completion row, branch-conversion row, representative choice, host equality row, or independent uniform-completion classifier.

Proof. Use Definition 1506.13. It states that a uniform-completion consumer reads the same Cauchy evidence, finite readback, separatedness, transport, replay, provenance, and naming rows as the metric-completion packet.

Next apply Definition 1506.14. This places the selected branch before R, so the uniform consumer cannot obtain tolerance data without first using the displayed filter or net evidence over X.

Apply Definition 1506.15 and Definition 1506.20. They keep any comparison through S, H, C, P, N and factor the seal through one selected branch.

The refusal and non-escape boundaries complete the handoff. By Definition 1506.19 and Definition 1506.8, no absent branch-conversion row, quotient row, representative selector, host equality row, or independent uniform-completion classifier can be recovered downstream. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.UniformCompletionHandoffSurface.MetricCompletion uniform completion handoff surface`

Theorem 1506.31 (MetricCompletion Cauchy-filter carrier admission)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier whose completion route uses the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch. Then carrier admission for the filter route is exactly the displayed row chain*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto R \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow},$$

where R is the finite $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row named in Definition 1506.1. The admission route has no coordinate for* $\mathsf{CauchyNet}^{\uparrow}$ *conversion, quotient completion, selected representatives, selected limits, host equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CauchyFilterCarrierAdmission.MetricCompletion cauchyfilter carrier admission`

Proof. Project the carrier row by row. The filter-presented route reads the metric source X first and then the displayed $\mathsf{CauchyFilter}^{\uparrow}$ row F; the net row D remains an empty marker for this branch.

The numerical request is admitted only through R. Thus $\mathsf{RegSeqRat}^{\uparrow}$ windows or $\mathsf{Real}^{\uparrow}$ seal data are read as finite rows attached to the filter branch, not as selected limit representatives.

Separated comparison and uniform-completion handoff then use S and the existing replay surface C. Since the tuple has no further coordinate, the excluded net conversion, quotient, selector, host-equality, choice, and ambient-completeness data cannot be part of filter carrier admission. □

Theorem 1506.32 (MetricCompletion Cauchy-filter limit factorization)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet $M = (X, F, D, R, S, H, C, P, N)$ with displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *branch F, every completion-facing limit read factors through X, F, R, S, H, C, P, N before any* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer may inspect it. If a* $\mathsf{CauchyNet}^{\uparrow}$ *branch is also named elsewhere in the ambient theory, it is not read by this factorization unless it appears as the selected branch of the same packet.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.CauchyFilterLimitFactorization.MetricCompletion cauchyfilter limit factorization`

Proof. Start with the completion-facing request. The replay row C sends the request to the metric source X, then to the filter evidence F, and then to the finite readback row R.

The limit comparison is not selected externally. It is checked through the separatedness row S, which supplies only zero-distance classifier evidence over the displayed source metric and the chosen Cauchy-filter branch.

The final handoff to a uniform-completion consumer is therefore a repacking of the same rows under H, P, N. A separately named $\mathsf{CauchyNet}^{\uparrow}$ route is outside this branch unless it is the selected row of the packet, so no branch conversion or selected limit can be inferred. □

Theorem 1506.33 (MetricCompletion Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet factors through the displayed finite readback row R after the selected* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *branch and before the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row. Consequently a downstream* $\mathsf{UniformCompletion}^{\uparrow}$ *or L10 real-completion consumer cannot recover* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected representatives, selected limits, countable choice, host equality, or ambient complete metric space data from the packet.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RealSealNonescape.MetricCompletion real seal nonescape`

Proof. Inspect the carrier definition. The only numerical coordinate is R, and R is explicitly the finite $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback row used after a selected Cauchy branch.

The separatedness row S can compare displayed limit readings only after that finite readback has been opened. It supplies zero-distance classifier evidence, not a proof that the host real line is complete.

Uniform-completion and L10 real-completion consumers then replay the same route through H, C, P, N. Since quotient streams, representatives, selected limits, countable-choice schedules, host equality rows, and ambient complete metric spaces are absent from the tuple, the Real-seal read cannot escape to any of them. □

Theorem 1506.34 (MetricCompletion root carrier obligation)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet. The root carrier projection of M is exhausted by the displayed route*

$$\mathsf{Met}^{\uparrow} \longmapsto (F : \mathsf{CauchyFilter}^{\uparrow}) \text{ or } (D : \mathsf{CauchyNet}^{\uparrow}) \longmapsto R \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto H, C, P, N,$$

where R is the finite $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row attached to the selected branch. Thus every root consumer of M must expose the metric source X, one selected Cauchy branch, the finite readback row R, the separated comparison row S, and the structural rows H, C, P, N before it can read any completion-facing endpoint.*

Proof. Project the accepted carrier of Definition 1506.1. Its visible analytic source is X, and the only Cauchy evidence available to a consumer is the displayed filter row F or the displayed net row D.

Use branch coverage from Definition 1506.9. The packet selects one branch for the requested read, and the filter-net compatibility theorem Definition 1506.10 allows only the recorded compatibility rows to mediate between the two presentations.

The numerical endpoint then passes through R and the separated comparison row S. The finite readback compatibility of Definition 1506.8 keeps the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row attached to the selected branch.

Finally the transport, replay, provenance, and naming fields H, C, P, N repack the same carrier. Since there is no additional coordinate, a root consumer cannot read quotient completions, selected

representatives, choice schedules, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness from carrier admission. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligations.MetricCompletion root obligation carrier`

Theorem 1506.35 (MetricCompletion root classifier obligation)**.** *Let $M = (X, F, D, R, S, H, C, P, N)$ and $M' = (X', F', D', R', S', H', C', P', N')$ be accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packets. If $X, F, D, R, S, H, C, P, N$ and $X', F', D', R', S', H', C', P', N'$ are related componentwise by the displayed* hsame *rows and the replay rows* Cont *transport the same selected Cauchy branch, then the root classifier for* $\mathsf{MetricCompletion}^{\uparrow}$ *is stable between M and M'. The stability claim is only this componentwise replay; it does not identify quotient completions, selected representatives, host equality, open-cover compactness, or ambient complete metric spaces.*

Proof. Begin with the classifier of Definition 1506.2. It compares packets by reading the metric source, the selected Cauchy branch, the finite readback, the separated comparison, and the structural rows.

Apply carrier projection from Definition 1506.34. Componentwise hsame transport moves each visible row of M to the matching visible row of M' without changing the order in which the branch and readback rows are inspected.

The replay hypothesis supplies the Cont part of the comparison. It ensures that a request routed through F or D in one packet is routed through the matching branch in the other packet, using only the compatibility rows already displayed by the packets.

Thus the classifier is stable under the stated componentwise transport and replay. The proof has not introduced a quotient completion, representative selector, host equality row, open-cover compactness proof, or ambient complete metric space; those objects remain outside the classifier surface. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligations.MetricCompletion root obligation classifier`

Theorem 1506.36 (MetricCompletion root ledger nonescape)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet $M = (X, F, D, R, S, H, C, P, N)$, every root ledger read factors through the selected Cauchy branch, the finite readback row R, and the separated comparison row S. Consequently the packet exports no quotient completion ledger, selected representative ledger, choice ledger, open-cover compactness ledger, ambient complete-metric ledger, or* $\mathsf{Real}^{\uparrow}$ *completeness ledger.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligationLedger.MetricCompletion root obligation ledger`

Proof. Use the finite readback ledger nonescape theorem Definition 1506.8. It places every numerical ledger entry after the selected Cauchy branch and before the downstream separated or uniform-completion read.

Next use separated-limit uniqueness, Definition 1506.5 and Definition 1506.29. These results identify displayed zero-distance comparisons through S; they do not create a selected host representative.

The uniform handoff theorem Definition 1506.13 then exports only the same branch, readback, separatedness, transport, replay, provenance, and naming rows. Downstream consumers therefore see the ledger already present in M.

Since quotient completions, representative selectors, countable-choice schedules, open-cover compactness, ambient complete metric spaces, and $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier tuple, no root ledger read can escape to those objects. □

Theorem 1506.37 (MetricCompletion Cauchy-filter scope lock)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$ *whose selected branch is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F*, every root-unblock read of the filter branch is confined to* BHist *provenance, componentwise* hsame *transport, displayed* Cont *replay, the package row* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *ledger before it may cite chapter 333. The read exports no Cauchy-net conversion, quotient completion, selected representative, selected limit, host equality, countable-choice schedule, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.FilterScopeLock.MetricCompletion filter scope lock`

Proof. Project the accepted packet through Definition 1506.34. On the filter branch, the only Cauchy evidence exposed to a root consumer is F together with the history row carried by P, N.

Transport is componentwise. The hsame rows in H preserve the displayed metric source, the selected filter branch, the finite readback row R, and the separated comparison row S without adding a branch conversion.

Replay is likewise local. The Cont row C routes the same request through F and then through the finite readback and separatedness rows, after which the package and naming rows supply the ledger used by chapter 333.

Since the tuple contains no coordinate for a net conversion, quotient carrier, representative selector, selected limit, host equality row, countable-choice schedule, or ambient completeness witness, the filter branch remains locked to that finite scope. □

Theorem 1506.38 (MetricCompletion Cauchy-net scope lock)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$ *whose selected branch is the* $\mathsf{CauchyNet}^{\uparrow}$ *row* D*, every root-unblock read of the net branch is confined to* BHist *provenance, componentwise* hsame *transport, displayed* Cont *replay, the package row* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$ *ledger before it may cite chapter 589. The read exports no filter selector, quotient completion, selected representative, selected limit, host equality, countable-choice schedule, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.NetScopeLock.MetricCompletion net scope lock`

Proof. Start from the root classifier obligation Definition 1506.35. The net branch is compared by the displayed row D and by the same finite readback, separatedness, transport, replay, package, and naming rows used by the packet.

The hsame rows in H move the net presentation componentwise. They do not choose a filter, a subnet, a limit representative, or a host equality witness.

The Cont replay row C then routes the accepted request through D, R, and S before the ledger rows P, N are opened. This is the only route available to the citation of chapter 589.

Thus a root-unblock consumer can read the net branch only through the finite kernel surface already present in M. Filter selection, quotient completion, selected representatives, selected limits, countable-choice schedules, host equality, and ambient completeness remain outside the packet. □

Theorem 1506.39 (MetricCompletion Real-seal scope lock)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet factors through the selected Cauchy branch, the finite readback row* R*, the separated comparison row* S*, and the structural rows* H, C, P, N *before it may cite chapter 301 or chapter 38. The handoff refuses quotient completion, quotient stream equality, selected representatives, selected limits, host equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1506.36. It places every root ledger read after the selected Cauchy branch and before any downstream completion-facing endpoint.

The finite readback row R is the only numerical coordinate. If R is a $\mathsf{RegSeqRat}^{\uparrow}$ row, the handoff cites chapter 301; if R is a $\mathsf{Real}^{\uparrow}$ seal row, the handoff cites chapter 38.

Separated comparison then passes through S, while H, C, P, N preserve transport, replay, provenance, and local naming. None of these rows changes the finite readback into a quotient carrier or a selected representative.

Consequently the Real-seal route is locked to the displayed branch and readback rows. Quotient completion, quotient stream equality, selected representatives, selected limits, host equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not exported. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.MetricCompletion real seal scope lock`

Theorem 1506.40 (MetricCompletion root carrier witness obligation)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$ *and any* BHist*-anchored root carrier request, the displayed witness is the finite row sequence consisting of the metric source* X*, the selected Cauchy branch, the finite readback row* R*, the separatedness row* S*, and the structural rows* H, C, P, N*. If the selected branch is the filter route, the witness reads* F *before* R*; if it is the net route, the witness reads* D *before* R*. No root carrier witness may consume a completion endpoint before this same route has exposed* R, S, H, C, P, N*.*

Proof. Start with carrier projection from Definition 1506.34. It already exhausts the accepted packet by X, one chosen Cauchy branch, R, S, H, C, P, N.

Use branch coverage from Definition 1506.9. The request selects exactly the displayed filter row F or the displayed net row D for the root read, so the witness cannot mix the two routes.

Finally apply finite-readback nonescape, Definition 1506.8. The readback row R remains downstream of the selected branch and upstream of separatedness and structural replay, giving the required BHist-anchored witness form. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligationCarrierWitness.MetricCompletion root obligation carrier witness`

Theorem 1506.41 (MetricCompletion root classifier witness obligation)**.** *Let* $M = (X, F, D, R, S, H, C, P, N)$ *and* $M' = (X', F', D', R', S', H', C', P', N')$ *be accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packets. A* BHist*-anchored root classifier witness between them consists of componentwise* hsame *rows for the metric source, the selected Cauchy branch, the finite readback row* R*, the separatedness row* S*, and the structural rows* H, C, P, N*, together with replay evidence that the same selected branch is read on both sides. Thus the classifier witness is branch-specific and cannot be replaced by quotient completion equality, selected representatives, or host metric equality.*

Proof. Begin with the root classifier obligation, Definition 1506.35. It states classifier stability using componentwise transport and replay for the displayed rows of the two packets.

The branch-specific part is supplied by branch coverage and filter-net compatibility. The selected Cauchy route is fixed before R is read, and compatibility can only mediate through recorded rows of the accepted packets.

The witness is therefore the finite transport data for the selected route plus R, S, H, C, P, N. Since the cited classifier theorem never introduces a quotient completion, representative selector, or host metric equality, the witness has no coordinate for those objects. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligationClassifierWitness.MetricCompletion root obligation classifier witness`

Theorem 1506.42 (MetricCompletion root ledger witness obligation)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$*, every* BHist*-anchored root ledger witness is the displayed ledger route through the selected Cauchy branch, the finite readback row* R*, the separatedness row* S*, and the structural rows* H, C, P, N*. The witness records whether the selected branch is* F *or* D*, records the finite readback row* R *attached to that branch, and records the separatedness row* S *before any uniform-completion or Real-facing consumer may use the ledger.*

Proof. Use the root ledger nonescape theorem Definition 1506.36. It places every root ledger read behind the selected branch, R, and S.

Refine the ledger route by finite-readback nonescape. The row R is not an ambient endpoint; it is the finite $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback attached to the displayed Cauchy branch.

Finally use carrier projection and branch coverage to repack the same rows as a witness. The structural rows H, C, P, N transport, replay, source, and name this ledger, while no additional coordinate supplies quotient completion ledgers, selected representatives, choice schedules, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.RootObligationLedgerWitness.MetricCompletion root obligation ledger witness`

Theorem 1506.43 (MetricCompletion source-separated scope)**.** *Let* $M = (X, F, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet that reaches a completion-facing consumer. The consumer first exposes the* $\mathsf{Met}^{\uparrow}$ *source row* X*, exactly one cofinal Cauchy branch, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *or the* $\mathsf{CauchyNet}^{\uparrow}$ *row* D*, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row* R*, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance uniqueness row* S*, and then the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff through* H, C, P, N*. Thus the source-separated scope of the packet is exactly the displayed route*

$$\mathsf{Met}^{\uparrow} \longmapsto (\mathsf{CauchyFilter}^{\uparrow} \textit{ or } \mathsf{CauchyNet}^{\uparrow}) \longmapsto (\mathsf{RegSeqRat}^{\uparrow} \textit{ or } \mathsf{Real}^{\uparrow}) \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}.$$

This scope excludes quotient completion equality, selected representatives, countable choice, ambient completeness, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 1506.1. Its visible source is the metric row X, and its only completion evidence rows are the displayed filter row F and net row D.

The NameCert obligation theorem, Definition 1506.3, keeps every accepted packet on the same carrier, classifier, stability, transport, replay, provenance, and naming surface. Branch coverage from Definition 1506.9 then forces a completion-facing consumer to choose exactly one of the two cofinal Cauchy routes before reading numerical data.

The finite readback ledger nonescape theorem, Definition 1506.8, places the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row after that selected branch and before any separated or uniform-completion read. The root obligation ledger, Definition 1506.36, repacks this same order at the root level.

It remains only to mark the boundary. Since the carrier has no quotient-equality coordinate, representative selector, choice schedule, ambient complete-space row, or $\mathsf{Real}^{\uparrow}$ completeness row, those principles cannot be opened by a completion-facing consumer. The consumer therefore reaches $\mathsf{UniformCompletion}^{\uparrow}$ only through X, the selected branch, R, S, and the structural rows H, C, P, N. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.SourceSeparatedScope.MetricCompletion source separated scope`

Theorem 1506.44 (MetricCompletion scoped dependency package)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$*, the scoped proof surface is bound by the existing carrier, classifier, Cauchy-filter handoff, Cauchy-net handoff, finite readback, separated-limit uniqueness, uniform-completion handoff, and root obligation rows. More precisely, every completion-facing consumer factors through the displayed route over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*, and through no quotient completion, selected representative, ambient complete metric space, open-cover compactness, countable-choice schedule, or host equality row.*

Proof. Carrier and classifier scope come from Definition 1506.1, Definition 1506.2, and Definition 1506.3. These fix the visible rows $X, F, D, R, S, H, C, P, N$ and require all local naming reads to pass through the same finite packet.

The Cauchy branch surface is then supplied by Definition 1506.4, Definition 1506.28, Definition 1506.6, Definition 1506.7, Definition 1506.9, and Definition 1506.10. Together these rows bind the filter and net presentations to the displayed branch evidence and prevent an unrecorded Cauchy presentation from entering the packet.

The endpoint and separation surface is supplied by Definition 1506.8, Definition 1506.5, and Definition 1506.29. These theorems place every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback behind the selected branch and keep zero-distance uniqueness inside the displayed $\mathsf{SeparatedMetric}^{\uparrow}$ row.

Finally, Definition 1506.13, Definition 1506.30, Definition 1506.34, Definition 1506.35, Definition 1506.36, and Definition 1506.43 repack the same data as the root and uniform-completion boundary. Since each cited row is a projection or replay of the displayed carrier, the scoped package contains exactly the named BEDC objects and none of the excluded ambient or choice principles. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.ScopedDependencyPackage.MetricCompletion scoped dependency package`

Theorem 1506.45 (MetricCompletion public export surface)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$*, the public interface exports exactly the displayed route through the* $\mathsf{Met}^{\uparrow}$ *source row* X*, one Cauchy branch* F *or* D*, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row* R*, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row* S*, the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. A public consumer may inspect this route, but it receives no quotient completion, selected representative, ambient complete metric space, open-cover compactness row, countable-choice schedule, branch-conversion row, host equality row, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the scoped dependency package of Definition 1506.44. It already binds every completion-facing consumer to the displayed carrier rows and excludes every ambient completion coordinate.

The public surface is therefore a projection of that same package, not a new carrier. It exposes the source row X, exactly one Cauchy branch F or D, the corresponding readback row R, separated comparison through S, and the uniform-completion handoff replayed through H, C, P, N.

Finally include the root BHist and structural rows as public provenance. They transport, replay, source, and name the route, while no carrier coordinate supplies quotient equality, representative choice, branch conversion, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.PublicExportSurface.MetricCompletion public export surface`

Definition 1506.46 (MetricCompletion scoped readback scope)**.** *For an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet* $M = (X, F, D, R, S, H, C, P, N)$*, the scoped readback scope is the displayed route*

$$\mathsf{Met}^{\uparrow} \longmapsto (\mathsf{CauchyFilter}^{\uparrow} \textit{ or } \mathsf{CauchyNet}^{\uparrow}) \longmapsto (\mathsf{RegSeqRat}^{\uparrow} \textit{ or } \mathsf{Real}^{\uparrow}) \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}$$

together with the carrier rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *that transport, replay, source, and name that route. A read belongs to this scope exactly when it exposes the same metric source row* X*, exactly one of the Cauchy branch rows* F *or* D*, the corresponding readback row* R*, the separatedness row* S*, and the uniform-completion handoff through* H, C, P, N*.*

Lean4 (definition) `BEDC.Derived.MetricCompletionUp.ScopedReadbackScope.MetricCompletionScopedReadbackScope`

Theorem 1506.47 (MetricCompletion uniform source lock)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet must keep the same displayed* $\mathsf{Met}^{\uparrow}$ *source row and the same chosen Cauchy branch. If the consumer reads through the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch, all later finite readback, separatedness, transport, replay, provenance, and naming rows are read over* X, F, R, S, H, C, P, N*; if it reads through the* $\mathsf{CauchyNet}^{\uparrow}$ *branch, the corresponding rows are read over* X, D, R, S, H, C, P, N*. No uniform-completion read may switch between the two branches or replace* X *by an ambient metric source.*

Proof. Start with Definition 1506.46. A scoped read is admitted only after the displayed $\mathsf{Met}^{\uparrow}$ row X and exactly one Cauchy branch have been opened.

The branch is fixed by the existing coverage and compatibility rows. Theorems Definition 1506.9, Definition 1506.10, and Definition 1506.43 place finite readback after the selected branch and before separatedness or uniform completion.

Finally apply Definition 1506.44. Its package route carries the same $X, F, D, R, S, H, C, P, N$ rows through transport and replay, so a uniform-completion consumer has no coordinate from which to substitute a new metric source or cross from the chosen filter route to the net route. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.UniformSourceLock.MetricCompletion uniform source lock`

Theorem 1506.48 (MetricCompletion no ambient completion scope)**.** *No scoped read of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet can introduce a quotient completion, selected representative, ambient complete metric space, open-cover compactness row, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem. Every completion-facing read is already exhausted by the scoped readback route of Definition 1506.46 and the source lock of Definition 1506.47.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.NoAmbientCompletionScope.MetricCompletion no ambient completion scope`

Proof. Inspect the carrier of Definition 1506.1. Its visible coordinates are the metric source, filter branch, net branch, readback row, separatedness row, transport, continuation, provenance, and local name.

The scoped readback definition repacks exactly those coordinates as the route through $\mathsf{Met}^{\uparrow}$, a chosen Cauchy branch, finite readback, $\mathsf{SeparatedMetric}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$. The source-lock theorem then prevents a consumer from replacing the displayed source or changing branches after the read has begun.

The excluded objects are not fields of the carrier and are not introduced by the cited package rows. Therefore quotient completions, representatives, ambient complete spaces, open-cover compactness, countable-choice data, and $\mathsf{Real}^{\uparrow}$ completeness remain outside the scoped MetricCompletion surface. □

Theorem 1506.49 (MetricCompletion bridge surface lock)**.** *Every bridge-facing consumer of an accepted* $\mathsf{MetricCompletion}^{\uparrow}$ *packet must preserve the same displayed* $\mathsf{Met}^{\uparrow}$ *source row, the chosen* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *branch, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row, the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff, and the structural* BHist, hsame, Cont, Pkg, *and* NameCert *rows. The bridge surface may replay the public export route, but it may not replace the source, switch the Cauchy branch, introduce an ambient completion, or discharge readback and separatedness through any row outside the displayed packet.*

Proof. Begin with Definition 1506.45. It identifies the public interface as the route through the displayed $\mathsf{Met}^{\uparrow}$ source, one Cauchy branch, finite readback, separatedness, uniform-completion handoff, and structural carrier rows.

Use Definition 1506.46 to bind every readback to that same scoped route. The definition places $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback after the chosen branch and before the $\mathsf{SeparatedMetric}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ rows.

Apply Definition 1506.47. A bridge-facing replay therefore keeps the same source row and the same selected filter or net branch throughout the handoff.

Finally, Definition 1506.48 excludes quotient completions, selected representatives, ambient complete spaces, open-cover compactness, countable-choice schedules, and $\mathsf{Real}^{\uparrow}$ completeness. Hence the bridge surface is locked to the displayed MetricCompletion packet. □

Lean4 ✓ `BEDC.Derived.MetricCompletionUp.BridgeSurfaceLock.MetricCompletion bridge surface lock`

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1506.1, Definition 1506.2, Definition 1506.3, Definition 1506.44, Definition 1506.45, Definition 1506.46, Definition 1506.47, Definition 1506.48, and Definition 1506.49 bind the finite L10 metric-completion interface to the displayed Metric source, chosen Cauchy branch, readback, separatedness, uniform-completion, BHist, hsame, Cont, Pkg, and NameCert rows, and bridge the public MetricCompletion export through the locked acceptance schema.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletion}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.MetricCompletionUp.MetricCompletionUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close quotient completions, selected representatives, ambient complete metric spaces, open-cover compactness, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: No further theory-axis upgrade remains; later work may add bridge verification or external consumers without changing the closed MetricCompletion surface.

Constructive story: The construction starts from a displayed metric source, routes cofinal evidence through CauchyFilter or CauchyNet rows, reads finite RegSeqRat or Real tolerance data when requested, and uses SeparatedMetric uniqueness only as a local zero-distance classifier route.

Chapter 1507

A Concrete Naming Certificate for $\mathsf{ArchimedeanApproximation}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ArchimedeanApproximation}^{\uparrow}$ packet records the finite Archimedean approximation boundary used before a rational or dyadic scale computation may be sealed as a real-name observation. It sits over the natural-number bound rows of chapter 28, the rational source rows of chapter 37, the dyadic tolerance ledger of chapter 291, the finite name windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal real-name seal of chapter 38. The packet names approximation as displayed BEDC route data, not as quotient real equality, a selected host limit, countable choice, trichotomy, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1507.1 (ArchimedeanApproximation carrier)**.** *An* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (N, Q, D, \epsilon, S, R, E, H, C, P, M)$$

with the following displayed rows:

1. *N is the* $\mathsf{Nat}^{\uparrow}$ *bounding row that records the finite index or scale bound used by the approximation request.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *rational source row naming the displayed lower and upper rational bounds read at that scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that refines the rational bounds into a dyadic tolerance ledger.*
4. *ϵ is the displayed tolerance request, read only through D and the* $\mathsf{Nat}^{\uparrow}$ *bound row N.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the bound and tolerance named by N, D, ϵ.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same finite window S.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that exposes the approximation to a Real-facing consumer after the finite window and readback have been displayed.*
8. *H records componentwise* hsame *transport for the natural bound, rational source, dyadic tolerance, stream window, regular readback, real seal, continuation, provenance, and local naming rows.*
9. *C records the displayed* Cont *replay that opens the finite window, reads the rational and dyadic rows, extracts the regular rational readback, and hands it to the* $\mathsf{Real}^{\uparrow}$ *seal.*

10. *P is the* Pkg *provenance over* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *M is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient real equality, selected representatives, host limits, countable choice, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationTasteGate single carrier alignment`

Theorem 1507.2 (ArchimedeanApproximation NameCert obligations)**.** *For every accepted* ArchimedeanApproxima *carrier $A = (N, Q, D, \epsilon, S, R, E, H, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Nat}^{\uparrow}$ *bound,* $\mathsf{Rat}^{\uparrow}$ *rational source,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local naming rows;*
2. *bound compatibility, stating that the requested finite approximation scale is read from the same N row that controls the tolerance request ϵ;*
3. *rational-bound exposure, stating that the lower and upper rational rows exported to the consumer are the displayed* $\mathsf{Rat}^{\uparrow}$ *data in Q;*
4. *dyadic tolerance exactness, stating that every tolerance comparison used by the packet is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger named by D;*
5. *window and readback compatibility, stating that S and R are the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows opened at the bound and tolerance already displayed by N, D, ϵ;*
6. *real-seal handoff, stating that the only Real-facing route is the row E after the displayed finite window and regular rational readback;*
7. *non-escape, stating that quotient real equality, selected host limits, countable choice, trichotomy, and ambient completeness are absent from the carrier.*

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier namecert obligations`

Proof. Project the carrier of Definition 1507.1. The only available coordinates are the natural bound, rational source, dyadic tolerance, finite stream window, regular rational readback, real seal, and the structural support rows.

Bound compatibility and rational-bound exposure are immediate from the rows N, Q, ϵ. The tolerance request is attached to the displayed natural bound, and the lower and upper rational readings are the $\mathsf{Rat}^{\uparrow}$ data named by Q rather than host rational choices outside the packet.

Dyadic tolerance exactness is the role of D. The finite window S can be opened only at the bound and tolerance already named by N, D, ϵ, and the readback R is the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ extraction from that same window.

The Real-facing consumer then sees only the seal row E transported and replayed by H and C. The provenance P and local naming row M name the same route and add no further coordinate, so the listed obligation rows exhaust the local $\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}$ surface. □

Theorem 1507.3 (ArchimedeanApproximation real-seal handoff)**.** *Every Real-facing read exported by an accepted* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Nat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, a consumer first reads the natural bound and rational source rows, then the dyadic tolerance row, then the finite stream window and regular rational readback, and only then the $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no quotient real equality, selected host limit, countable-choice schedule, trichotomy, or ambient completeness principle.*

Proof. Start from an accepted carrier and project its Real-facing read. The row E is the only $\mathsf{Real}^{\uparrow}$ seal coordinate, so the read cannot begin from a host real or a quotient class.

Before E can be consumed, the continuation row C displays the bound, rational source, tolerance, window, and readback rows. Thus the route first passes through N and Q, then through D, then through S and R.

The transport row H preserves the same components, while P and M record only provenance and local naming. Since the carrier has no additional coordinate, every Real-facing approximation read factors through the stated finite route and cannot obtain any of the excluded principles. □

Theorem 1507.4 (ArchimedeanApproximation window obligations)**.** *For every accepted* ArchimedeanApproximatio *carrier*

$$A = (N, Q, D, \epsilon, S, R, E, H, C, P, M),$$

the $\mathsf{StreamName}^{\uparrow}$ *window row* S *opens only after the natural bound row* N*, dyadic tolerance row* D*, and tolerance request* ϵ *have been displayed. A consumer that reads* S *before those rows is not reading the accepted packet.*

Proof. Project the carrier of Definition 1507.1. The definition places S as the finite-window row opened at the bound and tolerance named by N, D, ϵ.

Use the NameCert obligations of Definition 1507.2. Their bound-compatibility and window-readback compatibility clauses require the same N, D, ϵ rows before S can be consumed.

The handoff theorem Definition 1507.3 then keeps every Real-facing read downstream of the same finite window. Thus a packet that opens S without first displaying N, D, ϵ leaves the accepted ArchimedeanApproximation surface. □

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier window obligations`

Theorem 1507.5 (ArchimedeanApproximation dyadic cofinal window)**.** *For every accepted* ArchimedeanApproxima *carrier*

$$A = (N, Q, D, \epsilon, S, R, E, H, C, P, M),$$

any later unary bound refining N *and any smaller displayed dyadic tolerance refining* D *open a* $\mathsf{StreamName}^{\uparrow}$ *window whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback remains on the same* $\mathsf{Real}^{\uparrow}$ *seal route.*

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier dyadic cofinal window`

Proof. Start from the carrier definition and the NameCert obligations. The rows N, D, ϵ control the window row S, and the obligation surface keeps the bound-compatibility, dyadic tolerance exactness, and window-readback compatibility rows in the same finite packet.

Apply Definition 1507.4. A later unary bound and a smaller dyadic tolerance refine the displayed gate for opening a StreamName window; they do not create a separate host approximation source.

The readback then uses the same R row named by Definition 1507.6, and the real-seal handoff of Definition 1507.3 keeps the resulting read downstream of E. Thus the cofinal window remains a refinement of the accepted $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. □

Theorem 1507.6 (ArchimedeanApproximation dyadic readback)**.** *For every accepted* ArchimedeanApproximation$^{\uparrow}$ *carrier* $A = (N, Q, D, \epsilon, S, R, E, H, C, P, M)$*, rational bounds from the* Rat$^{\uparrow}$ *row* Q *become Real-facing only through the* DyadicRatCore$^{\uparrow}$ *row* D *and the* RegSeqRat$^{\uparrow}$ *readback row* R*. No consumer may expose the rational bounds directly as a* Real$^{\uparrow}$ *observation.*

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier dyadic readback`

Proof. Read the carrier definition. The rational source row Q is a displayed source of lower and upper rational bounds, while D records dyadic tolerance exactness and R records the regular rational readback from the finite window.

Apply the NameCert obligations. Rational-bound exposure, dyadic tolerance exactness, and window-readback compatibility are separate rows of the same certificate, so the rational data in Q must pass through D and R before the real seal row is available.

Finally use Definition 1507.3. The Real-facing route runs through DyadicRatCore$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ before Real$^{\uparrow}$, excluding any direct host read of rational bounds as real observations. □

Theorem 1507.7 (ArchimedeanApproximation real seal nonescape)**.** *For every accepted* ArchimedeanApproximatio *carrier* $A = (N, Q, D, \epsilon, S, R, E, H, C, P, M)$*, the* Real$^{\uparrow}$ *seal row* E *exports no quotient equality, selected host limit, countable choice, trichotomy, or ambient* Real$^{\uparrow}$ *completeness. The seal is only the terminal boundary of the displayed bound, dyadic, window, and readback route.*

Proof. The carrier definition lists the only coordinates available to a consumer: the natural bound, rational source, dyadic tolerance, tolerance request, StreamName window, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming rows.

The NameCert obligations identify non-escape as part of the local surface, and Definition 1507.3 fixes the route into E through the displayed finite rows.

Since no coordinate stores quotient equality, a selected host limit, a choice schedule, trichotomy, or completeness data, the seal row cannot export any of those principles. □

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier real seal nonescape`

Theorem 1507.8 (Dyadic upper ladder for Archimedean approximation)**.** *Every accepted* ArchimedeanApproximati *packet over* DyadicRatCore$^{\uparrow}$*,* Rat$^{\uparrow}$*,* Nat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *exposes a finite unary exponent ladder whose dyadic radius eventually lies below any displayed positive* Real$^{\uparrow}$ *tolerance row. The ladder is transported only through the displayed* hsame*,* Cont*, and* Pkg *rows of the same carrier, and it exports no trichotomy, no selected host limit, and no ambient choice of a real representative.*

Proof. Start from the carrier of Definition 1507.1 and the obligation rows of Definition 1507.2. The rows $N, D, \epsilon, S, R, E, H, C, P, M$ display the natural bound, dyadic tolerance ledger, positive tolerance request, finite window, regular readback, real seal, transport, replay, provenance, and local naming route.

Induct on the displayed unary ladder attached to the Nat$^{\uparrow}$ bound row. The base step reads the initial dyadic scale from D, and the successor step applies the dyadic scale monotonicity rows to replace the current radius by the next smaller dyadic radius while preserving the same DyadicRatCore$^{\uparrow}$ ledger.

The positive tolerance readback is then consumed through the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ route already fixed by Definition 1507.5 and Definition 1507.6. Componentwise hsame transport, displayed Cont replay, and Pkg provenance keep the ladder inside the accepted packet, so no trichotomy, host limit selection, or external real representative is introduced. □

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier dyadic upper ladder`

Theorem 1507.9 (ArchimedeanApproximation kernel scope)**.** *The carrier of Definition 1507.1, the NameCert obligations, the real-seal handoff, the window obligations, the dyadic cofinal window, the dyadic readback, the seal nonescape theorem, and the dyadic upper ladder bind every accepted* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *packet to the displayed* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. No accepted packet contains a row for quotient real equality, a selected host limit, countable choice, trichotomy, or ambient completeness.*

Proof. Start with the carrier and NameCert obligations, Definition 1507.1 and Definition 1507.2. They list the natural bound, rational source, dyadic tolerance, finite StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows as the whole accepted packet.

Next apply the real-seal handoff and window obligations, Definition 1507.3 and Definition 1507.4. These keep every Real-facing read downstream of the displayed $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route.

The dyadic cofinal-window and dyadic readback theorems, Definition 1507.5 and Definition 1507.6, preserve the same finite route under later bounds and smaller dyadic tolerances.

Finally use seal nonescape and the dyadic upper ladder, Definition 1507.7 and Definition 1507.8. They keep the terminal $\mathsf{Real}^{\uparrow}$ read and the unary ladder inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, so none of the excluded host principles is available. □

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationCarrier kernel scope`

Theorem 1507.10 (ArchimedeanApproximation public finite export)**.** *Every accepted* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *packet exports its public approximation read only through the displayed route*

$$\mathsf{Nat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The public read may expose a natural bound, rational bounds, a dyadic tolerance ledger, a finite stream window, a regular rational readback, and a terminal real-name seal, but it cannot read quotient real equality, a selected host limit, countable choice, trichotomy, or ambient completeness outside that route.

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.PublicFiniteExport.ArchimedeanApproximationCarrier public finite export`

Proof. Start with the carrier of Definition 1507.1 and the NameCert obligations of Definition 1507.2. They list $N, Q, D, \epsilon, S, R, E, H, C, P, M$ as the whole accepted packet and require each public read to preserve the same local naming surface.

Apply the real-seal handoff of Definition 1507.3. A Real-facing consumer reaches E only after the $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows have been displayed.

The dyadic cofinal window and dyadic readback theorems, Definition 1507.5 and Definition 1507.6, keep later bounds, smaller dyadic tolerances, rational bounds, finite windows, and regular readbacks on that same route. The seal nonescape theorem Definition 1507.7 prevents the terminal $\mathsf{Real}^{\uparrow}$ row from exporting any hidden real-equality or limit principle.

Finally use the dyadic upper ladder and kernel scope, Definition 1507.8 and Definition 1507.9. They keep the public approximation read inside the displayed hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, so the exported object is exactly the finite BEDC route named in the statement. □

Lean4 ✓ `BEDC.Derived.ArchimedeanApproximationUp.ArchimedeanApproximationPublicFiniteExport`

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanApproximation}^{\uparrow}$ packet is generated from a natural-number bound, rational bounding rows, a DyadicRatCore tolerance ledger, a StreamName finite window, a RegSeqRat readback, a RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1507.1, Definition 1507.2, Definition 1507.3, Definition 1507.4, Definition 1507.5, Definition 1507.6, Definition 1507.7, Definition 1507.8, and Definition 1507.9 close the carrier, NameCert obligation, real-seal handoff, window, dyadic cofinal-window, dyadic readback, seal-nonescape, dyadic upper-ladder, and kernel-scope surface for the displayed packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected host limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public export theorem for the same $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert packet.

Chapter 1508

A Concrete Naming Certificate for $\mathsf{BishopCompletionComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCompletionComparison}^{\uparrow}$ packet compares the displayed regular located Cauchy route with the completion-facing real-name seal as finite BEDC data. It sits after the regular located Cauchy surface of chapter 1454, the boundary packet of chapter 1455, the located-limit route of chapter 1420, the located-real enclosure of **??**, the real-name seal of chapter 38, the regular rational source of chapter 301, the stream-window source of chapter 126, and the dyadic arithmetic core of chapter 291. The chapter records the finite comparison route from regular located Cauchy data to the $\mathsf{Real}^{\uparrow}$ seal; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 1508.1 (BishopCompletionComparison carrier)**.** *A* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, B, L, E, T, H, K, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *source row, including its visible* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source,* $\mathsf{StreamName}^{\uparrow}$ *finite-window route, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
2. *B is the* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *boundary row attached to the same regular located Cauchy source.*
3. *L is the* $\mathsf{LocatedLimit}^{\uparrow}$ *route read from the displayed boundary and regular located source rows.*
4. *E is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row exposed before the terminal real-name seal is inspected.*
5. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, B, L, E have been displayed.*
6. *H records componentwise* hsame *transport for the regular located source, boundary route, located-limit route, located-real enclosure, real seal, replay rows, provenance, and local naming row.*
7. *K records the displayed* Cont *replay from R to B, from B to L, from L to E, and from E to T.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{BishopLocatedComplet}$ $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, real trichotomy, countable choice, open-cover compactness, or a host-selected limit.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonTasteGate single carrier alignment`

Theorem 1508.2 (BishopCompletionComparison NameCert obligations)**.** *For the carrier of Definition 1508.1, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for* $C = (R, B, L, E, T, H, K, P, N)$*, regular located source exposure through* R*, boundary-route exposure through* B*, located-limit exposure through* L*, located-real enclosure exposure through* E*, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is tuple projection from the displayed packet C. The rows expose the regular located Cauchy source, Bishop boundary route, located-limit route, located-real enclosure, terminal real seal, transport, replay, provenance, and local name.

The analytic obligations are exactly the rows R, B, L, E, T. The source row R carries the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data used by the regular located Cauchy surface; the boundary row B keeps the same source visible before the located-limit and located-real rows are read.

The structural obligations are exactly H, K, P, N. They transport the same rows, replay the same finite route, record provenance for the named packets, and bind the local certificate without adding any coordinate outside the carrier. □

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonNameCertObligations`

Theorem 1508.3 (BishopCompletionComparison regular located route)**.** *Every accepted* $\mathsf{BishopCompletionCompar}$ *packet* $C = (R, B, L, E, T, H, K, P, N)$ *routes a completion-facing read through*

$$R \longmapsto B \longmapsto L \longmapsto E \longmapsto T.$$

The row R *supplies the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *source, including the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;* B *supplies the* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *comparison boundary;* L *and* E *supply the* $\mathsf{LocatedLimit}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *rows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier regular located route`

Proof. Project the accepted carrier. The only source row for regular located Cauchy data is R, and its regular rational, stream-window, and dyadic tolerance coordinates are present only as part of that row.

Now read the completion-facing side. The replay row K places B after R, then places L and E before the terminal seal T. Thus every such read factors through the displayed boundary and locatedness route.

Transport and provenance preserve the same factorization. The rows H and P identify the transported route as the same finite comparison packet, while N names the local surface for downstream consumers. □

Theorem 1508.4 (BishopCompletionComparison seal non-escape)**.** *No accepted* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *carrier exports data outside the tuple* $R, B, L, E, T, H, K, P, N$*. In particular, a* $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal cannot recover* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, real trichotomy, countable choice, open-cover compactness, or a host-selected limit from the packet.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier seal nonescape`

Proof. Inspect the carrier of Definition 1508.1. Its only analytic rows are the regular located source, boundary route, located-limit route, located-real enclosure, and terminal real seal.

Apply Definition 1508.2. Every local obligation is a projection of those rows or a structural transport, replay, provenance, or naming obligation for those rows.

Finally apply Definition 1508.3. Any terminal seal read first passes through R, B, L, E, so the finite comparison route is exhausted before the $\mathsf{Real}^{\uparrow}$ seal is reached. Since none of the excluded principles is a carrier coordinate, none can be exported by the packet. □

Theorem 1508.5 (BishopCompletionComparison Real route)**.** *Every accepted* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *carrier* $C = (R, B, L, E, T, H, K, P, N)$ *exposes the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the regular located Cauchy source row* R*, Bishop located completion boundary row* B*, located-limit route* L*, located-real enclosure* E*, componentwise transport* H*, replay row* K*, provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *have all been read. The route refuses* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, trichotomy, countable choice, open-cover compactness, and host-selected limits.*

Proof. Project the carrier of Definition 1508.1. The terminal seal T is a displayed row of the same packet and is not available before the source, boundary, located-limit, and located-real rows have been exposed.

Apply Definition 1508.2. Its surface lists the source, boundary, located-limit, located-real, terminal seal, transport, replay, provenance, and local naming rows as the complete local obligation surface.

Use the regular located route of Definition 1508.3 and the non-escape theorem Definition 1508.4. Together they force any $\mathsf{Real}^{\uparrow}$ terminal read through R, B, L, E, T and exclude every principle not carried by the tuple. □

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier real route`

Theorem 1508.6 (BishopCompletionComparison regular seal)**.** *Every comparison between a Bishop located completion row and a regular Cauchy completion row accepted by* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedLimit}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *rows carried by* $C = (R, B, L, E, T, H, K, P, N)$*. The comparison exports no separate Cauchy quotient, selected limit, trichotomy, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier regular seal`

Proof. Start with the regular located route of Definition 1508.3. It exposes the regular located Cauchy source through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows inside R, then routes the comparison through B, L, E before the terminal seal is visible.

Apply the Real route theorem, Definition 1508.5. It places every terminal $\mathsf{Real}^{\uparrow}$ read after the regular source, boundary, located-limit, located-real, transport, replay, provenance, and local naming rows.

Finally use the seal non-escape theorem Definition 1508.4. Since the carrier has no coordinate for the excluded quotient, choice, compactness, selected-limit, or completeness principles, the regular comparison reaches the Real seal only through the listed finite rows. □

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonRegularSeal`

Theorem 1508.7 (BishopCompletionComparison boundary transport)**.** *For every accepted* BishopCompletionComp *carrier* $C = (R, B, L, E, T, H, K, P, N)$*, componentwise* hsame *transport and displayed* Cont *replay preserve the route*

$$R \longmapsto B \longmapsto L \longmapsto E \longmapsto T$$

from the regular located Cauchy source through the Bishop located completion boundary to the terminal $\mathsf{Real}^{\uparrow}$ *seal. A consumer cannot bypass the boundary through* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of Cauchy names, trichotomy, countable choice, open-cover compactness, or host-selected limits.*

Proof. Begin with the NameCert obligations of Definition 1508.2. They list H as the componentwise transport row and K as the replay row for the same regular source, boundary, located-limit, located-real, and terminal seal coordinates.

Apply the regular located route of Definition 1508.3. Its displayed order is $R \longmapsto B \longmapsto L \longmapsto E \longmapsto T$, so transport can only move the named rows componentwise and replay can only re-expose that route.

Finally use the seal non-escape theorem Definition 1508.4 together with the Real route of Definition 1508.5. Since the terminal seal is read only after the boundary and locatedness rows, none of the excluded completeness, quotient, trichotomy, choice, compactness, or host-limit principles is available as a bypass. □

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier boundary transport`

Theorem 1508.8 (BishopCompletionComparison located-filter comparison)**.** *Every accepted* BishopCompletionCor *carrier that receives a* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row routes that row through the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows used by the regular located route before any completion comparison is consumed. No quotient filter, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row may replace that route.*

Lean4 ✓ `BEDC.Derived.BishopCompletionComparisonUp.BishopCompletionComparisonCarrier located filter comparison`

Proof. Start with the carrier of Definition 1508.1. Its source row R is the only regular located Cauchy source, and that row already exposes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data consumed by the comparison route.

Apply the regular located route of Definition 1508.3 and the Real route of Definition 1508.5. Together they force a completion-facing read through the displayed source, boundary, located-limit, located-real, and terminal $\mathsf{Real}^{\uparrow}$ seal rows before the comparison is public.

Now use the regular seal theorem Definition 1508.6 and boundary transport theorem Definition 1508.7. A $\mathsf{LocatedCauchyFilter}^{\uparrow}$ row handed to the comparison surface is therefore compared only after the sibling route through chapter 1661, chapter 126, chapter 301, chapter 291, and chapter 38 has been displayed.

Finally apply the seal non-escape theorem Definition 1508.4. Since the carrier has no coordinate for quotient filters, selected limits, countable-choice schedules, or ambient completeness, none of those rows can replace the located-filter route. □

Theorem 1508.9 (BishopCompletionComparison public L10 route)**.** *Every completion-facing comparison read accepted by* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *factors through* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$*,* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$*,* $\mathsf{LocatedCauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,*

and the terminal $\mathsf{Real}^{\uparrow}$ *seal before any public comparison boundary is exported. The exported boundary is therefore the displayed finite L10 route*

$$R \longmapsto B \longmapsto L \longmapsto E \longmapsto T$$

together with the located-filter comparison handoff, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row; it is not a quotient completion, a selected limit, trichotomy, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 1508.2. They expose the carrier rows $R, B, L, E, T, H, K, P, N$ as the whole local obligation surface.

Apply the regular located route of Definition 1508.3. It places the $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ source before the boundary row and keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data inside the displayed source route.

Next apply the Real route of Definition 1508.5 and the regular seal theorem Definition 1508.6. These force the terminal $\mathsf{Real}^{\uparrow}$ read to occur only after the source, boundary, located-limit, located-real, transport, replay, provenance, and local naming rows have been exposed.

Now use the boundary transport theorem Definition 1508.7. It preserves the same route under componentwise hsame transport and displayed Cont replay, so the boundary cannot be bypassed by a transported seal read.

Finally use the located-filter comparison theorem Definition 1508.8 and the seal non-escape theorem Definition 1508.4. The $\mathsf{LocatedCauchyFilter}^{\uparrow}$ handoff is consumed through the same finite rows, and none of the excluded quotient, selection, choice, compactness, trichotomy, or completeness principles is a carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionComparison}^{\uparrow}$ packet is generated from a regular located Cauchy source row, a Bishop located completion boundary row, a located-limit row, a located-real enclosure row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionComparison}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1508.9 binds the public L10 comparison route through $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, BishopLocatedCompletionB $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any comparison boundary is exported.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of Cauchy names, real trichotomy, countable choice, open-cover compactness, host-selected limits, Lean verification, or bridge checking.

Upgrade path: A checked Lean theorem for Definition 1508.9 upgrades the verification axis without changing the paper closure scope.

Chapter 1509

A Concrete Naming Certificate for $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ packet records the constructive Cauchy convergence criterion as finite BEDC route data. It sits between the finite stream schedule of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38, the modulus surface of chapter 292, and the limit-seal route of chapter 386. The certificate names the finite rows that make a convergence read admissible; it is not host completeness, quotient streams, selected limits, trichotomy, or open-cover compactness.

Definition 1509.1 (Cauchy convergence criterion carrier)**.** *A* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are the only sequence observations visible to the carrier.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning each dyadic tolerance request a displayed finite threshold for the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic scales, rational comparisons, and exhaustion budget used by the criterion.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled windows under the threshold and tolerance data M, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, M, D, R have been displayed and the local convergence criterion has been admitted.*
6. *H records componentwise* hsame *transport for the stream schedule, modulus, dyadic ledger, regular-rational handoff, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each tolerance request through S, M, D, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for host completeness, quotient-stream equality, a selected host limit, trichotomy, open-cover compactness, or an unbounded stream read.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionTasteGate single carrier alignment`

Theorem 1509.2 (Cauchy convergence criterion NameCert obligations)**.** *For every accepted* CauchyConvergenceCri *carrier $G = (S, M, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyConvergenceCriterion}^{\uparrow}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier admission for the displayed stream schedule, Cauchy modulus, dyadic tolerance ledger, regular-rational handoff, Real seal, transport, continuation, provenance, and local name;*
2. *finite-window classification, requiring every sequence read to pass through the* $\mathsf{StreamName}^{\uparrow}$ *schedule S;*
3. *modulus stability, requiring the threshold row M to be preserved by the componentwise* hsame *transports in H and replayed by C for the same tolerance requests;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the regular-rational row R to be read only after the scheduled windows, modulus thresholds, and dyadic tolerance ledger have been displayed;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the seal row E to remain downstream of S, M, D, R and the stored structural rows;*
6. *ledger exhaustion, requiring every accepted tolerance request to be accounted for by the dyadic ledger D and the provenance row P before any convergence-facing consumer can read the seal.*

No obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionNameCertObligations`
↪ variant `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier namecert obligations`

Proof. Project the carrier of Definition 1509.1. Its visible coordinates are exactly the stream schedule, modulus, dyadic ledger, regular-rational handoff, Real seal, transport, continuation, provenance, and local naming rows.

The finite-window classifier is forced by the first coordinate. A sequence read has no source except the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S, and the continuation row can replay only the windows attached to that schedule.

Modulus stability and ledger exhaustion are read next. The threshold row M and the dyadic ledger D answer the same tolerance requests, and the transport row H preserves those answers before the handoff R is available.

The Real-facing rows are downstream. The regular-rational handoff R is admitted only from S, M, D, and the seal row E is consumed only after that handoff. Since the carrier has no coordinate for host completeness, quotient streams, selected limits, trichotomy, open-cover compactness, or an unbounded stream read, the six listed rows exhaust the local NameCert surface. □

Theorem 1509.3 (Cauchy convergence criterion Real seal handoff)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$*-facing convergence read of a* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier $G = (S, M, D, R, E, H, C, P, N)$ factors through the displayed finite route*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

together with the stored hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. Before reaching the* $\mathsf{Real}^{\uparrow}$ *seal, the read is compatible with the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary and exports no host completeness theorem, quotient stream, selected limit, trichotomy row, open-cover compactness row, or detached seal.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier real seal handoff`

Proof. Start with an accepted Real-facing convergence read. Carrier admission gives only the finite schedule S as sequence data, so the read first factors through the displayed stream windows.

The tolerance part of the route is then forced. The modulus row M supplies thresholds for the requests, while the dyadic ledger D records the scales and comparisons consumed at those thresholds. The continuation row C replays this same request path.

Next pass to regular-rational data and the seal. The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ handoff in the carrier, and the $\mathsf{Real}^{\uparrow}$ seal row E is downstream of R. The transport and provenance rows H and P preserve the same packet, while the local name N records the route as a $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ certificate.

The route has no extra coordinate after E. Thus compatibility with the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary is witnessed by the same finite handoff, and the read cannot export host completeness, quotient streams, selected limits, trichotomy, open-cover compactness, or a detached seal. □

Theorem 1509.4 (Cauchy convergence criterion limit-seal determinacy)**.** *Let* $G = (S, M, D, R, E, H, C, P, N)$ *and* $G' = (S, M, D, R, E', H, C, P, N)$ *be accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carriers sharing the same displayed stream schedule, Cauchy modulus, dyadic ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise transport, continuation replay, provenance, and local* NameCert *rows. If their Real-facing convergence reads are compatible with the same* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary over the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, then the two reads determine the same seal route.*

The comparison cannot introduce a detached seal, quotient stream, selected host limit, trichotomy row, open-cover compactness row, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier limit seal determinacy`

Proof. Apply Definition 1509.3 to each Real-facing convergence read. Both reads are forced through the displayed finite route

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

or through the corresponding notational endpoint E' for the second carrier, with the same transport, continuation, provenance, and local naming rows.

The shared $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary now supplies the seal comparison. By Definition 386.23, once the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal boundary, hsame transport, Cont replay, Pkg provenance, and NameCert rows agree, there is no second seal route available to a completion or $\mathsf{Real}^{\uparrow}$ consumer.

Finally return to the non-escape clauses in the handoff theorem. The carrier has no coordinate for a detached seal, quotient stream, selected host limit, trichotomy row, open-cover compactness row, or ambient completeness theorem, so the identified $\mathsf{CauchyLimitSeal}^{\uparrow}$ route is determined entirely by the displayed finite packet. □

Theorem 1509.5 (Cauchy convergence criterion NullSequence factorization)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing convergence read of G whose comparison row is a zero-difference read factors through a displayed* $\mathsf{NullSequence}^{\uparrow}$ *packet before the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary may consume it.*

The displayed $\mathsf{NullSequence}^{\uparrow}$ *packet uses the same* $\mathsf{RegSeqRat}^{\uparrow}$ *difference row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{StreamName}^{\uparrow}$ *window schedule S,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row M,* $\mathsf{Real}^{\uparrow}$ *seal row E, and provenance rows H, C, P, N already carried by G. No second stream representative, quotient stream, host limit, trichotomy row, choice principle, open-cover compactness object, or ambient completeness coordinate is introduced by this zero-difference handoff.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier nullsequence factorization`

Proof. Project the accepted carrier G. The coordinates S, M, D, R give the finite tolerance-window route: stream observations are read only through S, tolerance requests are bounded by M, dyadic scales and comparisons are recorded by D, and the regular-rational comparison row is R.

Now impose the zero-difference condition on the comparison read. By Definition 380.6, the tolerance request must pass through the displayed dyadic tolerance row, observation window, and threshold row before any $\mathsf{Real}^{\uparrow}$ seal read. In the present carrier these rows are exactly D, S, M, with R as the displayed difference row.

Apply Definition 380.7. It makes the $\mathsf{NullSequence}^{\uparrow}$ packet the finite consumer boundary for the $\mathsf{Real}^{\uparrow}$ equality read, so the seal row E is reached only through the same difference, tolerance, window, threshold, transport, continuation, package, and local certificate rows.

Finally return to the remaining coordinates of G. The rows H, C, P, N record component-wise transport, continuation replay, provenance, and local naming for the same packet, while the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary can consume only the finite handoff already displayed. Since the carrier has no coordinate for a quotient stream, host limit, trichotomy, choice, open-cover compactness, or ambient completeness theorem, the zero-difference read factors through $\mathsf{NullSequence}^{\uparrow}$ and exports nothing beyond that route. □

Theorem 1509.6 (Cauchy convergence criterion source-triad exhaustion)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing read of G factors through the displayed source triad*

$$S, M, D, R$$

before the $\mathsf{Real}^{\uparrow}$ *seal row E is available.*

The read has no bypass through quotient streams, host completeness, selected limits, trichotomy, or open-cover compactness. In particular, every convergence-facing consumer sees the stream schedule, modulus threshold row, dyadic tolerance ledger, and regular-rational handoff before it can consume a seal boundary.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier source triad exhaustion`

Proof. Project the carrier in Definition 1509.1. The only sequence observations are the finite windows named by S, the only tolerance thresholds are carried by M, the only scale and comparison ledger is D, and the only regular-rational handoff is R.

Apply Definition 1509.2. Its finite-window classification, modulus stability, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, Real-seal nonescape, and ledger-exhaustion rows require the consumer to traverse S, M, D, R before any convergence-facing seal read.

For a $\mathsf{Real}^{\uparrow}$ consumer, Definition 1509.3 identifies the displayed route from the stream schedule through the threshold, ledger, and handoff rows to E. For a $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer, Definition 1509.4 reads the same boundary data and admits no second seal route.

The final exclusion is structural. The carrier definition contains no coordinate for quotient streams, host completeness, selected limits, trichotomy, or open-cover compactness, and Definition 1509.5 keeps zero-difference reads inside the same finite packet. Hence every accepted read is exhausted by the displayed source triad before E. □

Theorem 1509.7 (Cauchy convergence criterion window-threshold coverage)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Every finite* $\mathsf{StreamName}^{\uparrow}$ *window consumed by the convergence criterion has a matching* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold in M and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in D in the same packet. Thus no unbounded stream read is accepted by the local classifier route.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionDyadicRoute determinacy`

Proof. Start from a finite window read admitted by the local classifier. By Definition 1509.1, the window can only come from the displayed $\mathsf{StreamName}^{\uparrow}$ schedule S.

The threshold and tolerance data are then forced by the same carrier projection. The row M assigns the relevant $\mathsf{CauchyModulus}^{\uparrow}$ threshold for the tolerance request, while D records the dyadic scale, rational comparison, and exhaustion budget consumed by that threshold.

Apply Definition 1509.2. Finite-window classification keeps the read on S, modulus stability preserves the threshold row under H and C, and ledger exhaustion accounts for the request in D and P before the handoff R or seal row E can be consumed.

The handoff and seal results do not add a stream source. By Definition 1509.3 and Definition 1509.5, downstream $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and zero-difference consumers reuse the same S, M, D rows. Since the carrier has no unbounded stream coordinate, the classifier accepts only covered finite windows. □

Theorem 1509.8 (Cauchy convergence criterion tail-diameter scope)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Every tail-diameter request consumed by a convergence-facing read factors through the same finite route*

$$S \longmapsto M \longmapsto D \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal row E is available. The request is therefore scoped by the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff of chapter 126, chapter 301, chapter 291, chapter 38, and chapter 386; it does not introduce a second tail classifier, detached limit seal, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier tail diameter scope`

Proof. Project the accepted carrier. By Definition 1509.1, the only stream observations are the finite windows in S, the only thresholds are in M, the only tolerance ledger is D, and the only regular handoff is R.

Apply Definition 1509.7. A tail-diameter request must be covered by a window and a matching threshold in the same packet, so the request cannot leave S, M, D before the handoff row is formed.

Now use Definition 1509.6 and Definition 1509.3. These force every convergence-facing read through S, M, D, R before E, while Definition 1509.4 keeps the $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer on the same finite boundary. □

Theorem 1509.9 (Cauchy convergence criterion scoped closure)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing, or zero-difference convergence read of G is scoped exactly by the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal E,* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

Consequently such a read has no coordinate outside

$$S, M, D, R, E, H, C, P, N$$

and cannot introduce quotient streams, selected host limits, detached limit seals, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier scoped closure`

Proof. Start with the source rows. The source-triad exhaustion theorem, Definition 1509.6, forces every $\mathsf{Real}^{\uparrow}$- or $\mathsf{CauchyLimitSeal}^{\uparrow}$-facing read through S, M, D, R before E is visible.

The window and tail requirements stay on the same route. By Definition 1509.7, every finite stream window has its matching threshold and dyadic ledger inside the carrier, and by Definition 1509.8, every tail-diameter request passes through S, M, D, R before the seal.

Zero-difference reads add no second source. The null-sequence scope theorem, Definition 1509.10, keeps comparison reads inside the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$ route.

Finally read the structural rows. The carrier definition supplies only H, C, P, N as transport, replay, provenance, and local naming data for this route. Since the listed theorems exhaust all Real-facing, limit-seal-facing, and zero-difference reads, the displayed rows $S, M, D, R, E, H, C, P, N$ are exactly the scope of the convergence criterion packet. □

Theorem 1509.10 (Cauchy convergence criterion null-sequence scope)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. If a convergence-facing comparison is a zero-difference read, then the read uses the existing* $\mathsf{NullSequence}^{\uparrow}$ *factorization route over $S, M, D, R, E, H, C, P, N$ and not a second stream representative. In particular, the zero-difference consumer remains inside the finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route recorded by the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier null sequence scope`

Proof. Start from the zero-difference comparison. The carrier has one stream schedule S and one regular-rational comparison row R, so there is no separate stream representative available for the comparison.

Apply Definition 1509.5. The zero-difference read factors through the displayed $\mathsf{NullSequence}^{\uparrow}$ packet using the same difference row R, tolerance ledger D, stream schedule S, threshold row M, Real seal row E, and structural rows H, C, P, N.

Finally combine Definition 1509.7 with Definition 1509.4. The finite window-threshold ledger and the limit-seal boundary are already fixed by the carrier, so the null-sequence route cannot introduce quotient streams, selected host limits, or a second pre-seal representation. □

Theorem 1509.11 (Cauchy convergence criterion regular-seal handoff)**.** *Let $G = (S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. The* $\mathsf{RegSeqRat}^{\uparrow}$ *row R is the unique pre-seal route available to any* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of G: every such consumer first reads the* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{CauchyModulus}^{\uparrow}$ *thresholds M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, and then R, before it can consume the* $\mathsf{Real}^{\uparrow}$ *seal row E. The route is preserved by H, C, P, N and contains no quotient-stream equality, selected host limit, or ambient completeness principle.*

Proof. Project the consumer-facing rows of G. The regular-rational handoff coordinate is exactly R, and the seal row E is downstream of it by Definition 1509.1.

Apply Definition 1509.6. Every $\mathsf{Real}^{\uparrow}$- or $\mathsf{CauchyLimitSeal}^{\uparrow}$-facing read must pass through the displayed source triad S, M, D, R before any seal row is visible.

The uniqueness claim follows from Definition 1509.4. Once the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{Real}^{\uparrow}$ seal boundary, transport, continuation, provenance, and name rows agree, no second seal route remains for the consumer. The structural rows H, C, P, N only preserve and name this route. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier regular seal handoff`

Theorem 1509.12 (Cauchy convergence criterion regular-tail coverage)**.** *Every accepted* $\mathsf{CauchyConvergenceCriteri}$ *packet $G = (S, M, D, R, E, H, C, P, N)$ whose finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{CauchyModulus}^{\uparrow}$ *row pass the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold exposes a* $\mathsf{RegSeqRat}^{\uparrow}$ *tail handoff and a* $\mathsf{Real}^{\uparrow}$ *seal through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Concretely, each covered tolerance request first reads the windows in S, then the matching threshold in M, then the dyadic tolerance ledger D, then the regular tail row R, and only then the seal row E.*

The coverage route exports no quotient stream, selected tail representative, detached limit seal, host completeness theorem, trichotomy row, or open-cover compactness object.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier regular tail coverage`

Proof. Project the carrier of Definition 1509.1. The rows $S, M, D, R, E, H, C, P, N$ are the only visible rows of the accepted packet, so every tolerance request begins inside the same BHist packet and the same structural provenance.

Apply Definition 1509.7. The finite $\mathsf{StreamName}^{\uparrow}$ window consumed by the request has a matching $\mathsf{CauchyModulus}^{\uparrow}$ threshold in M and a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row in D, so the request cannot read an unbounded stream or an unrecorded threshold.

Next use Definition 1509.11. Once the covered threshold has passed through S, M, D, the row R is the unique pre-seal $\mathsf{RegSeqRat}^{\uparrow}$ route available to the $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer.

Finally combine Definition 1509.5 with Definition 1509.17. Zero-difference comparisons remain inside the same null-sequence factorization, and compatible boundary reads determine the same classifier route through H, C, P, N. Hence the covered packet exposes the regular tail handoff and the Real seal through the same finite rows, with none of the excluded host-side data in scope. □

Theorem 1509.13 (Cauchy convergence criterion cofinal-modulus extraction)**.** *Every accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier $G = (S, M, D, R, E, H, C, P, N)$ determines a cofinal finite*

modulus route for convergence-facing reads. Any later tolerance request is first reduced to a displayed $\mathsf{StreamName}^{\uparrow}$ *window in* S*, then to the matching* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold in* M*, then to the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and then to the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed.*

The route exports no quotient stream, selected host limit, trichotomy branch, open-cover compactness object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier cofinal modulus extraction`

Proof. Start with a later tolerance request. By Definition 1509.7, any accepted finite window is a displayed $\mathsf{StreamName}^{\uparrow}$ row in S with a matching $\mathsf{CauchyModulus}^{\uparrow}$ threshold in M and a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row in D.

Next apply Definition 1509.8. Tail-diameter data cannot leave the displayed route; they pass through S, M, D, R before the terminal seal row is available.

Use Definition 1509.11. The row R is the unique pre-seal $\mathsf{RegSeqRat}^{\uparrow}$ handoff for the $\mathsf{CauchyLimitSeal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer, and the structural rows H, C, P, N preserve that handoff.

Finally apply Definition 1509.12. It identifies the covered tolerance request with the same regular tail and Real seal route. Since the carrier has no coordinate for quotient streams, selected host limits, trichotomy, open-cover compactness, or ambient completeness, the cofinal modulus route exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CofinalModulusExtraction.CauchyConvergenceCriterionCa cofinal modulus extraction`

Theorem 1509.14 (Cauchy convergence criterion scoped readback strengthening)**.** *Let* $G = (S, M, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing,* $\mathsf{NullSequence}^{\uparrow}$*-facing, or regular-tail consumer of* G *reads the same finite scope*

$$S, M, D, R, E, H, C, P, N.$$

Thus the convergence criterion exposes no quotient stream, selected host limit, detached seal, trichotomy row, open-cover compactness row, or ambient completeness coordinate to any of these consumers.

Proof. First read the Real and limit-seal consumers. The source-triad exhaustion theorem and the Real seal handoff theorem force such consumers through S, M, D, R before E, while the limit-seal determinacy theorem keeps the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary on the same displayed rows.

Next read the window and tail data. The window-threshold coverage theorem puts each finite $\mathsf{StreamName}^{\uparrow}$ window together with its threshold and dyadic ledger inside S, M, D, and the tail-diameter scope theorem sends every tail-diameter request through S, M, D, R before the seal row.

For zero-difference and regular-tail consumers, use the null-sequence scope theorem, the regular-seal handoff theorem, and the regular-tail coverage theorem. These results keep $\mathsf{NullSequence}^{\uparrow}$ factorization and regular-tail handoff inside the same S, M, D, R, E route.

Finally project the structural rows of the accepted carrier. The only remaining coordinates are hsame transport H, Cont replay C, Pkg provenance P, and the local $\mathsf{NameCert}$ row N, so every listed consumer reads exactly the displayed scope and none of the excluded host-side data. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier scoped readback strengthening`

Theorem 1509.15 (Cauchy convergence criterion Specker boundary)**.** *Let* $G = (S, M, D, R, E, H, C, P, N)$ *be an accepted* CauchyConvergenceCriterion$^{\uparrow}$ *packet read through an accepted* SpeckerSequence$^{\uparrow}$ *carrier surface. The read exposes only finite* StreamName$^{\uparrow}$ *windows,* CauchyModulus$^{\uparrow}$ *threshold rows,* DyadicRatCore$^{\uparrow}$ *tolerance rows, the* RegSeqRat$^{\uparrow}$ *handoff, the* Real$^{\uparrow}$ *seal, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows shared by the two packets.*

Consequently the criterion cannot yield a located supremum, ambient Real completeness, quotient stream, or selected host limit from the Specker read. Such a consumer must supply an additional located-modulus row outside the displayed CauchyConvergenceCriterion$^{\uparrow}$ *and* SpeckerSequence$^{\uparrow}$ *packets.*

Proof. First apply Definition 1509.6. Any Real$^{\uparrow}$- or CauchyLimitSeal$^{\uparrow}$-facing read of G is forced through the source triad S, M, D, R before the seal row E is visible.

Next use Definition 1509.12. The covered tolerance request exposes only the scheduled windows, matching threshold, dyadic ledger, regular tail row, seal row, and the structural rows H, C, P, N.

Now read the same route through the Specker carrier surface. By Definition 1406.2, Definition 1406.12, and Definition 1406.10, the SpeckerSequence$^{\uparrow}$ packet supplies a finite regular-source and seal route but no located supremum coordinate, selected limit, quotient stream, countable-choice selector, trichotomy branch, or ambient completeness theorem.

The remaining located-supremum claim therefore has no row to read. A downstream consumer that needs located convergence must add an external located-modulus row; it cannot recover that row from the displayed Cauchy criterion and Specker packets. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier specker boundary`

Theorem 1509.16 (Cauchy convergence criterion located-modulus frontier)**.** *Let* $G = (S, M, D, R, E, H, C, P, N)$ *be an accepted* CauchyConvergenceCriterion$^{\uparrow}$ *carrier. Any downstream consumer claiming located convergence from* G *must display an additional* LocatedReal$^{\uparrow}$ *or located-modulus row outside the carrier. The stored* CauchyModulus$^{\uparrow}$ *thresholds,* DyadicRatCore$^{\uparrow}$ *ledger,* SpeckerSequence$^{\uparrow}$ *boundary read,* StreamName$^{\uparrow}$ *schedule,* RegSeqRat$^{\uparrow}$ *handoff, and* Real$^{\uparrow}$ *seal do not produce such a row.*

Proof. First apply Definition 1509.2. The accepted packet exposes exactly the carrier rows $S, M, D, R, E, H, C, P, N$ and the listed stability, exactness, handoff, transport, replay, provenance, and local naming obligations.

Next use Definition 1509.6. Every Real$^{\uparrow}$- or CauchyLimitSeal$^{\uparrow}$-facing read passes through the source triad S, M, D, R before the seal row is available, so the only tolerance data visible before the seal are the displayed CauchyModulus$^{\uparrow}$ thresholds and DyadicRatCore$^{\uparrow}$ ledger.

Now apply Definition 1509.15. Reading the same carrier through the SpeckerSequence$^{\uparrow}$ surface still exposes only finite stream windows, threshold rows, dyadic tolerance rows, regular rational handoff, terminal seal, and the structural rows; it does not expose a located supremum, selected host limit, quotient stream, or ambient completeness coordinate.

Thus a located-convergence consumer has no carrier row from which to read locatedness. It must add a separate LocatedReal$^{\uparrow}$ or located-modulus row, and that row lies outside the displayed CauchyConvergenceCriterion$^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier located modulus frontier`

Theorem 1509.17 (Cauchy convergence criterion ledger-classifier exactness)**.** *Let*

$$G = (S, M, D, R, E, H, C, P, N)$$

and

$$G' = (S, M, D, R, E', H, C, P, N)$$

be accepted CauchyConvergenceCriterion$^{\uparrow}$ *carriers with the same displayed* StreamName$^{\uparrow}$ *schedule,* CauchyModulus$^{\uparrow}$ *thresholds,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff, componentwise transport, continuation replay, provenance, and local name rows. If their* Real$^{\uparrow}$ *and* CauchyLimitSeal$^{\uparrow}$ *boundary reads are compatible, then the two carriers determine the same local classifier route.*

Proof. Project both carriers. The displayed classifier data agree on S, M, D, R, H, C, P, N, so their finite windows, threshold assignments, dyadic tolerance rows, regular-rational handoff, transports, replays, provenance, and local names are the same.

Apply Definition 1509.6. Any Real$^{\uparrow}$- or CauchyLimitSeal$^{\uparrow}$-facing read from either carrier must pass through the same source triad S, M, D, R before reaching its seal row.

Next use Definition 1509.7. The shared ledger covers exactly the stream windows accepted by the classifier, so there is no residual unbounded stream read that could distinguish the two carriers.

The compatible boundary reads identify the remaining endpoint. By Definition 1509.4, compatible CauchyLimitSeal$^{\uparrow}$ boundaries over the same displayed schedule, handoff, dyadic ledger, seal boundary, transport, continuation, provenance, and name rows determine the same seal route. The Real handoff theorem supplies the corresponding Real$^{\uparrow}$ route, and the NullSequence factorization theorem keeps zero-difference comparisons inside the same packet. Therefore the accepted carriers determine the same local classifier route. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier ledger classifier exactness`

Theorem 1509.18 (Cauchy convergence criterion obligation closure upgrade)**.** *For every accepted* CauchyConvergenceCriterion$^{\uparrow}$ *carrier* $G = (S, M, D, R, E, H, C, P, N)$*, the carrier admission, finite-window classification, modulus stability,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal nonescape, and ledger exhaustion rows listed in Definition 1509.2 form the full obligation surface of the chapter. Every convergence-facing read must pass through the displayed* StreamName$^{\uparrow}$ *schedule,* CauchyModulus$^{\uparrow}$ *threshold row,* DyadicRatCore$^{\uparrow}$ *ledger,* RegSeqRat$^{\uparrow}$ *handoff,* Real$^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row, with no quotient stream, selected host limit, ambient completeness, trichotomy, or open-cover compactness coordinate.*

Proof. Apply Definition 1509.2. It enumerates the six obligation rows and states that no obligation is stored outside the displayed carrier coordinates.

The Real-facing side is controlled by Definition 1509.3. It forces every seal read through $S, M, D, R, E, H, C, P, N$ and refuses a detached Real endpoint.

The boundary comparison is controlled by Definition 1509.4. Compatible reads over the same displayed schedule, handoff, dyadic ledger, seal boundary, transport, replay, provenance, and naming rows determine the same seal route.

Zero-difference and source coverage are supplied by Definition 1509.5 and Definition 1509.6. The former keeps comparison reads inside the same finite packet, and the latter exhausts every Real- or limit-seal-facing read by S, M, D, R before E.

Together these theorems account for carrier admission, window coverage, threshold stability, regular handoff, Real-seal nonescape, and ledger exhaustion. Since the carrier has no additional coordinate, the chapter reaches obligation closure exactly over the displayed packet. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier obligation closure upgrade`

Theorem 1509.19 (Cauchy convergence criterion regular-tail window)**.** *For every accepted* CauchyConvergenceCrit *carrier $G = (S, M, D, R, E, H, C, P, N)$, the first criterion read opens the regular-tail window only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R. In particular, every tail-diameter request must expose the dyadic modulus rows M, D before the regular rational row R or the terminal seal row E can be consumed.*

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier regular tail window`

Proof. Project the carrier rows. A criterion read has no stream observation outside S, no threshold source outside M, and no tolerance ledger outside D.

Apply Definition 1509.7. Each finite window accepted by the classifier has its matching threshold and dyadic tolerance row in the same packet, so the regular-tail request is already inside S, M, D before any handoff is read.

Now use Definition 1509.8. Tail-diameter data pass through S, M, D, R before E is available. The structural rows H, C, P, N preserve and name that route, so the first criterion read opens exactly the displayed regular-tail window and dyadic modulus rows. □

Theorem 1509.20 (Cauchy convergence criterion Real seal route)**.** *For every accepted* CauchyConvergenceCriterion *carrier $G = (S, M, D, R, E, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal is available only after the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*, and* $\mathsf{NullSequence}^{\uparrow}$ *rows used by the criterion have been displayed. The seal read therefore factors through the source schedule, dyadic tolerance ledger, modulus threshold, regular rational handoff, zero-difference comparison route when present, and the structural rows H, C, P, N before it reaches E.*

Proof. Start with the source and modulus rows. By Definition 1509.6, every Real-facing read must pass through S, M, D, R before the seal row E is visible.

The dyadic and regular parts remain displayed. Definition 1509.19 forces the first regular-tail read through the $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{CauchyModulus}^{\uparrow}$ threshold, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, so the seal cannot bypass those rows.

For zero-difference comparisons, apply Definition 1509.5. The $\mathsf{NullSequence}^{\uparrow}$ route uses the same $S, M, D, R, E, H, C, P, N$ packet before any $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary consumes the comparison. Finally Definition 1509.3 identifies E as the downstream $\mathsf{Real}^{\uparrow}$ seal, with no detached endpoint or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier real seal route`

Theorem 1509.21 (Cauchy convergence criterion scope window)**.** *For every accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier $G = (S, M, D, R, E, H, C, P, N)$, the convergence criterion's window-threshold and tail-diameter reads are scoped by the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{NullSequence}^{\uparrow}$ *zero-difference factorization, and* $\mathsf{Real}^{\uparrow}$ *seal row E. A consumer must traverse*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

with H, C, P, N before it can read a convergence boundary.

Proof. Project the carrier rows. The only windows are named by S, the only threshold row is M, the only dyadic tolerance ledger is D, and the only regular-rational handoff is R.

Apply Definition 1509.7 and Definition 1509.8. These theorems force every finite window and tail-diameter request through S, M, D, R before the seal row E is available.

Zero-difference and seal reads remain on the same route. By Definition 1509.5, any zero-difference comparison factors through the displayed $\mathsf{NullSequence}^{\uparrow}$ packet; by Definition 1509.3, the $\mathsf{Real}^{\uparrow}$ consumer can only read the downstream seal row. The structural rows H, C, P, N preserve and name that scope. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier scope window`

Theorem 1509.22 (Cauchy convergence criterion kernel-dependency scope)**.** *Every exported read from an accepted* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *carrier depends only on* BHist, hsame, *displayed* Cont *replay,* Pkg *provenance, the local* $\mathsf{NameCert}$ *row, and the listed L10 dependencies* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$. *It does not depend on host limits, quotient equality, trichotomy, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition exposes exactly $S, M, D, R, E, H, C, P, N$. Thus the kernel rows available to a consumer are the history packet, componentwise equality transport, continuation replay, provenance, and local naming row attached to those displayed dependencies.

The obligation theorem and the source-triad exhaustion theorem account for carrier admission, finite-window classification, threshold stability, regular handoff, ledger exhaustion, and the pre-seal route. The null-sequence factorization and limit-seal determinacy theorems account for zero-difference and boundary comparisons without adding a new coordinate.

Finally read the exclusions from the carrier surface. No row names a host limit, quotient equality, trichotomy branch, countable-choice selector, open-cover compactness object, or ambient completeness principle. Hence every exported criterion read is scoped by the stated kernel and L10 dependency rows. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier kernel dependency scope`

Theorem 1509.23 (Cauchy convergence criterion modulus-tail lock)**.** *Every accepted convergence-facing read whose null-difference row is certified must pass through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{NullSequence}^{\uparrow}$ *factorization, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before it may be consumed as a* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *criterion. The route admits no selected limits, quotient stream equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the window-threshold coverage row. It binds each finite $\mathsf{StreamName}^{\uparrow}$ window to the matching $\mathsf{CauchyModulus}^{\uparrow}$ threshold and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, so a convergence-facing request cannot reach the regular handoff before the modulus and dyadic rows have been displayed.

Next use null-sequence factorization. A certified null-difference row is accepted only as a $\mathsf{NullSequence}^{\uparrow}$ route over the same displayed window, threshold, dyadic tolerance, and regular-rational handoff rows.

Finally apply the regular seal handoff and the kernel dependency scope. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after the displayed $\mathsf{RegSeqRat}^{\uparrow}$ handoff and structural rows have replayed the same packet, and the kernel scope has no coordinate for selected limits, quotient stream equality, trichotomy, open-cover compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionCarrier modulus tail lock`

Theorem 1509.24 (Cauchy convergence criterion scoped dependency package)**.** *Every accepted* CauchyConvergenceCriterion$^{\uparrow}$ *consumer factors through the displayed dependency package consisting of* StreamName$^{\uparrow}$ *windows,* CauchyModulus$^{\uparrow}$ *thresholds,* DyadicRatCore$^{\uparrow}$ *tolerance ledgers,* RegSeqRat$^{\uparrow}$ *handoff,* CauchyLimitSeal$^{\uparrow}$ *boundary rows,* NullSequence$^{\uparrow}$ *zero-difference packages, the terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The package contains no ambient completeness principle, quotient-stream equality, selected host limit, trichotomy branch, open-cover compactness argument, or external limit oracle.*

Proof. Project the carrier $G = (S, M, D, R, E, H, C, P, N)$. The finite stream windows are stored in S, the threshold and modulus data in M, the dyadic tolerance ledger in D, and the regular-rational handoff in R.

Apply Definition 1509.7, Definition 1509.17, and Definition 1509.19. These rows force every window, threshold, tolerance, and regular-tail read to remain inside the displayed S, M, D, R package before the terminal seal is available.

Next read the boundary comparisons. By Definition 1509.4 and Definition 1509.5, any CauchyLimitSeal$^{\uparrow}$ or NullSequence$^{\uparrow}$ consumption is a downstream route over the same displayed windows, thresholds, dyadic ledgers, and handoff rows.

Finally use Definition 1509.20 and Definition 1509.22. The Real$^{\uparrow}$ seal and the structural rows H, C, P, N preserve and name precisely this dependency package. Since the carrier has no coordinate for ambient completeness, quotient-stream equality, selected host limits, trichotomy, open-cover compactness, or an external limit oracle, every accepted consumer is scoped by the stated route. □

Closure status for NameCert$_{\text{CauchyConvergenceCriterion}^{\uparrow}}$**.**

Constructive story: A CauchyConvergenceCriterion$^{\uparrow}$ packet is generated from a finite StreamName$^{\uparrow}$ schedule, a CauchyModulus$^{\uparrow}$ threshold row, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a RegSeqRat$^{\uparrow}$ handoff, a Real$^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyConvergenceCriterion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1509.1, Definition 1509.2, Definition 1509.3, Definition 1509.4, Definition 1509.5, Definition 1509.6, Definition 1509.7, Definition 1509.17, and Definition 1509.24 expose the public dependency package from StreamName windows through Cauchy modulus and dyadic ledger rows to the RegSeqRat handoff, CauchyLimitSeal route, NullSequence zero-difference package, and Real seal.

Formal verification: FormalStatus(NameCert$_{\text{CauchyConvergenceCriterion}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.CauchyConvergenceCriterionUp.CauchyConvergenceCriterionNameCertObligations`

Bridge status: paperBridge

Not claimed: This chapter does not close ambient completeness, quotient-stream equality, selected host limits, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Only a future standard bridge check can raise the independent formal bridge status for the same public StreamName, CauchyModulus, DyadicRatCore, RegSeqRat, Real seal, transport, replay, provenance, and local NameCert packet.

Chapter 1510

A Concrete Naming Certificate for $\mathsf{FormalBall}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Vickers-style formal-ball presentations, the $\mathsf{FormalBall}^{\uparrow}$ packet records a radius-indexed observation interface for constructive completion. It sits between the metric carrier of chapter 57, the dyadic radius ledger of chapter 291, the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the real-name seal of chapter 38, and the uniform-completion boundary of chapter 1473. The packet gives completion consumers a finite formal-ball window without quotienting Cauchy names, selecting limits, using countable choice, invoking open-cover compactness, or assuming ambient completeness.

Definition 1510.1 (FormalBall carrier)**.** *A* $\mathsf{FormalBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, R, D, W, H, C, P, N)$$

with the following visible rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, distance witness surface, and finite observation histories.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *radius-name row read as a positive rational tolerance through finite requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger recording radius refinement, dyadic addition, and the strict comparison budget attached to R.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose entries display the selected center observations and requested radii before any completion handoff.*
5. *H records componentwise* hsame *transport for the metric source, radius row, dyadic ledger, finite window, replay row, provenance row, and local naming row.*
6. *C records the displayed* Cont *replay from metric observations to radius refinements, window reads,* $\mathsf{Real}^{\uparrow}$ *tolerance seals, and the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff boundary.*
7. *P is the* Pkg *provenance over* $\mathsf{FormalBall}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallTasteGate single carrier alignment`

Theorem 1510.2 (FormalBall NameCert obligations)**.** *For the carrier of Definition 1510.1, the local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *surface consists exactly of carrier admission for* $B = (M, R, D, W, H, C, P, N)$*, metric source exposure through* M*, radius-name exposure through* R*, dyadic refinement exposure through* D*, finite-window exposure through* W*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every completion-facing read of an accepted carrier must pass through the displayed route

$$M \longmapsto R \longmapsto D \longmapsto W \longmapsto H, C, P, N.$$

No such read exports quotient completion equality, a chosen Cauchy representative, a selected limit, countable choice, open-cover compactness, or ambient completeness.

Proof. Carrier admission is tuple projection from Definition 1510.1. The tuple displays the metric source, radius name, dyadic ledger, finite window, transport, replay, provenance, and local name.

The analytic obligations are exactly M, R, D, W. The metric row supplies point and distance reads, the radius row supplies the requested rational scale, the dyadic ledger records the refinement budget, and the stream window exposes the center observations available at that scale.

The structural obligations are exactly H, C, P, N. They transport the same rows, replay the same finite route, record package provenance, and bind the local certificate. Since the carrier contains no quotient, selected representative, selected limit, countable-choice schedule, compactness principle, or ambient-completeness coordinate, none can be exported by the NameCert surface. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier namecert obligations`

Theorem 1510.3 (FormalBall radius refinement)**.** *Let* $B = (M, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. A refinement request from radius* r *to a displayed smaller radius* s *is admissible only when the dyadic ledger* D *records the comparison and the replay row* C *reads the corresponding finite center window* W *over the same metric source* M*.*

The refined formal ball is therefore a smaller window inside the same packet. It is not a second metric, not a quotient of centers, not a selected Cauchy representative, and not an appeal to ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallRadiusRefinement`
↪ variant `BEDC.Derived.FormalBallUp.FormalBallCarrier radius refinement`

Proof. Project the accepted packet and read the metric source M. Any center comparison used by the refinement is interpreted through the point classifier and distance rows already displayed there.

Next read the radius row R and the dyadic ledger D. The ledger is the only coordinate that records the comparison between r and s, together with the finite arithmetic budget needed to keep the smaller radius inside the requested tolerance.

Finally replay the finite window through W and C. The transport row H preserves the same metric, radius, dyadic, and window rows, while P and N record provenance and naming. Thus refinement narrows an existing displayed formal-ball window and introduces no extra metric, quotient, representative choice, selected limit, or completeness principle. □

Theorem 1510.4 (FormalBall radius refinement chain)**.** *Let* $B = (M, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. For any finite displayed chain of radius refinements*

$$r_0 \geq r_1 \geq \cdots \geq r_k,$$

whose comparisons are all recorded in the same dyadic ledger D and replayed through the same row C, every formal-ball window reached along the chain is read over the same metric source M and the same finite stream-window row W.

The chain therefore refines only the displayed radius budget of the accepted packet. It does not identify quotient-completion classes, choose Cauchy representatives, select limits, use countable choice, invoke open-cover compactness, or assume ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier radius refinement chain`

Proof. For the empty chain there is no new refinement step. The accepted packet already exposes the metric source M, the radius row R, the dyadic ledger D, the stream-window row W, and the replay row C through Definition 1510.1.

For a nonempty chain, apply Definition 1510.3 to the first displayed comparison. The theorem keeps the smaller formal-ball window inside the same accepted packet, with its comparison recorded by D and its finite center read through W and C over M.

Iterate the same argument over the remaining displayed comparisons. The transport row H carries the metric, radius, dyadic, window, and replay rows componentwise at each step, while P and N only record provenance and local naming for the same packet.

No step introduces a field for quotient-completion equality, a selected Cauchy representative, a selected limit, a countable-choice schedule, an open-cover compactness principle, or an ambient complete space. Hence the whole finite chain preserves the same finite formal-ball window interface. □

Theorem 1510.5 (FormalBall completion-window handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier* $B = (M, R, D, W, H, C, P, N)$ *receives only the finite window route through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{UniformCompletion}^{\uparrow}$ *rows recorded by B. The handoff exposes radius-indexed formal balls as finite observation data before any completion boundary is read.*

Consequently the handoff can support constructive completion arguments that use formal balls as observable neighborhoods, but it does not close quotient completion, selected limits, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier completion window handoff`

Proof. Begin with the metric source and radius data. The rows M and R fix the center-observation surface and the requested radius scale before any completion-facing route is available.

Read the dyadic and stream windows next. The row D records the arithmetic budget for the radius, and W records the finite center windows inspected at that budget. Any $\mathsf{Real}^{\uparrow}$ tolerance seal reached by the consumer is read only through these rows.

Now pass to the completion boundary through C. The replay route carries the same finite formal-ball data to the $\mathsf{UniformCompletion}^{\uparrow}$ handoff, while H, P, N preserve transport, provenance, and local naming.

The carrier fields are exhausted by this route. Since no field stores quotient equality, a selected limit, a countable-choice schedule, an open-cover compactness principle, or an ambient complete space, the completion-window handoff exports none of those principles. □

Theorem 1510.6 (FormalBall center-radius admission)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier exposes its* $\mathsf{Met}^{\uparrow}$ *center row,* $\mathsf{RegSeqRat}^{\uparrow}$ *radius row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, and* $\mathsf{StreamName}^{\uparrow}$

finite window before the local $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *row is consumed. The admission route is a* BHist *route over*

$$M, R, D, W, H, C, P, N,$$

and no consumer may read the formal ball as an accepted packet while omitting any one of M, R, D, W.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier center radius admission`

Proof. Project the carrier of Definition 1510.1. Its first four analytic coordinates are exactly the metric source M, radius row R, dyadic ledger D, and finite window W.

The local naming row N appears only after the structural rows H, C, P. Thus an accepted read must first display the center source, radius name, comparison ledger, and window route, and only then consume the local certificate.

Since the tuple has no alternate center or radius coordinate, a consumer that drops any of M, R, D, W has no accepted BHist route to the same $\mathsf{FormalBall}^{\uparrow}$ packet. □

Theorem 1510.7 (FormalBall formal-order monotonicity)**.** *Let* $B = (M, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. Any radius refinement or formal-order shrinkage of* B *can only pass through the displayed dyadic ledger* D *and the replay row* C*. It cannot introduce quotient completion equality, selected limits, countable choice, open-cover compactness, or ambient completeness as evidence for the shrinkage.*

Proof. Read the accepted carrier and isolate the rows that can mention radii. The radius row is R, and the comparison data for shrinking that radius is recorded only in D.

Replay through C preserves the same metric source and finite window while reading the smaller radius through the displayed ledger. The transport row H keeps this shrinkage componentwise inside the same packet, and P, N only source and name the route.

The remaining forbidden principles are absent by row exhaustion. There is no field for quotient completion equality, selected limits, countable choice, open-cover compactness, or ambient completeness, so none can be used to justify formal-order monotonicity. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier formal order monotonicity`

Definition 1510.8 (FormalBall rounded-filter boundary)**.** *For an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier* $B = (M, R, D, W, H, C, P, N)$*, the rounded-filter boundary row is the finite row*

$$\partial_{\mathrm{rf}}(B) = (M, R, D, W, C, H, P, N)$$

read after the metric source, radius row, dyadic refinement ledger, finite center-window row, replay row, componentwise transport, provenance, and local naming have all been displayed. The boundary records only rounded formal-ball windows generated by the same carrier; it does not contain quotient completion equality, selected Cauchy representatives, countable-choice schedules, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter boundary certificate`

Theorem 1510.9 (FormalBall completion dependency scope)**.** *Every completion-facing read of an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier* $B = (M, R, D, W, H, C, P, N)$ *factors through the displayed rows* M, R, D, W, H, C, P, N *and through the named* $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{UniformCompletion}^{\uparrow}$ *surfaces. The read may consume the rounded-filter boundary of Definition 1510.8, but it cannot bypass the carrier rows or replace them by a quotient completion principle.*

Proof. Start with carrier projection and the NameCert obligations. Theorem 1510.2 exposes the metric source, radius name, dyadic ledger, finite window, transport, replay, provenance, and local name as the only carrier rows available to a completion-facing consumer.

Next apply the radius-refinement chain. Theorem 1510.4 keeps every smaller-radius window inside the same dyadic ledger D, stream window W, and replay row C over the same source M.

Finally apply formal-order monotonicity. Theorem 1510.7 prevents a shrinkage read from introducing quotient completion equality, selected representatives, countable choice, compactness, or ambient completeness. Therefore the completion dependency scope is exactly the named finite route through $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier completion dependency scope`

Theorem 1510.10 (FormalBall public window export)**.** *Every public* $\mathsf{FormalBall}^{\uparrow}$ *consumer receives exactly the displayed route*

$$M, R, D, W, H, C, P, N$$

before any $\mathsf{UniformCompletion}^{\uparrow}$ *handoff or* $\mathsf{Real}^{\uparrow}$ *tolerance seal is read. The exported window is the metric source, radius row, dyadic ledger, finite stream window, componentwise transport, replay, provenance, and local NameCert surface of the accepted carrier.*

Proof. Begin with the NameCert obligations and center-radius admission. They expose M, R, D, W as the analytic prefix of every accepted formal-ball read.

Next apply the radius-refinement chain and completion-window handoff. Radius comparisons remain in D, finite center observations remain in W, and the route to $\mathsf{Real}^{\uparrow}$ or $\mathsf{UniformCompletion}^{\uparrow}$ is replayed through C over the same packet.

Finally attach H, P, N. These rows transport, source, and name the public window, so the consumer receives exactly M, R, D, W, H, C, P, N before any completion or tolerance seal is consumed. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier public window export`

Theorem 1510.11 (FormalBall radius ledger exhaustion)**.** *Every radius shrinkage, formal-order comparison, and completion-window read from an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier is exhausted by the radius row R, dyadic ledger D, finite window W, and replay row C, together with the fixed source, transport, provenance, and local name in M, H, P, N. The route exports no quotient completion, selected limit, countable choice, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier radius ledger exhaustion`

Proof. Read the radius-refinement chain. It keeps every shrinkage inside the displayed radius row and the dyadic comparison ledger.

Apply formal-order monotonicity and completion dependency scope. Formal-order comparisons and completion-window reads are replayed through W, C over the same metric source rather than through a quotient completion or external complete space.

The remaining structural rows H, P, N preserve the same packet. Since the accepted carrier has no coordinate for selected limits, countable choice, open-cover compactness, quotient completion, or ambient completeness, the radius ledger exhausts the handoff. □

Theorem 1510.12 (FormalBall rounded-filter non-escape)**.** *For an accepted* FormalBall$^{\uparrow}$ *carrier, a rounded-filter read exposes no quotient completion equality, no selected Cauchy representative, no countable-choice schedule, no open-cover compactness argument, and no ambient completeness principle. It exposes only the finite boundary row* $\partial_{\mathrm{rf}}(B)$ *and the rows from which that boundary is projected.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter nonescape`

Proof. Project the rounded-filter boundary of Definition 1510.8. Its entries are M, R, D, W, C, H, P, N, all inherited from the accepted carrier.

Use the NameCert obligations and the radius-refinement chain. The first theorem exhausts the accepted packet by the displayed carrier rows, and the second keeps every rounded refinement inside D, W, C over the same metric source.

The formal-order monotonicity theorem supplies the final exclusion. Since every shrinkage remains inside the displayed dyadic and replay rows, the rounded-filter read has no coordinate from which to expose quotient equality, selected representatives, countable choice, compactness, or ambient completeness. □

Theorem 1510.13 (FormalBall Cauchy-filter handoff)**.** *Every* CauchyFilter$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer that reads an accepted* FormalBall$^{\uparrow}$ *packet must factor through the displayed* Met$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows. The handoff exports formal balls only as finite center-radius windows in the accepted* BHist *packet, not as quotient completions or selected limits.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier cauchy filter handoff`

Proof. Begin with the analytic rows. The metric source M supplies the center surface, R supplies the regular rational radius, D supplies the dyadic comparison ledger, and W supplies the finite center-window observations.

The Real-facing tolerance seal is reached only through the replay row C, which reads the same M, R, D, W data before any completion-facing consumer can inspect the handoff. Componentwise transport is carried by H.

Finally P and N record provenance and local naming for this route. Therefore a CauchyFilter$^{\uparrow}$ or UniformCompletion$^{\uparrow}$ consumer receives exactly the displayed finite formal-ball packet and no quotient-completion equality, chosen Cauchy representative, selected limit, countable-choice schedule, compactness principle, or ambient complete space. □

Theorem 1510.14 (FormalBall rounded-filter Cauchy-filter boundary)**.** *Every* CauchyFilter$^{\uparrow}$ *consumer of the rounded-filter boundary of an accepted* FormalBall$^{\uparrow}$ *packet must first read the same rows*

$$M, R, D, W, C, H, P, N$$

displayed by $\partial_{\mathrm{rf}}(B)$ *in Definition 1510.8. Equivalently, a filter-limit boundary may be consumed only after the metric source, radius row, dyadic refinement ledger, finite center window, replay row, componentwise transport, provenance, and local NameCert row of the accepted packet have been displayed. The route exports no quotient completion, selected limit, countable-choice schedule, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter cauchyfilter boundary`

Proof. Start from the rounded-filter boundary definition. The boundary row is exactly the projection $\partial_{\mathrm{rf}}(B) = (M, R, D, W, C, H, P, N)$, so a consumer of that boundary already sits inside the displayed accepted carrier.

Apply rounded-filter non-escape and rounded-filter admission. The non-escape theorem excludes quotient completion, selected representatives, choice schedules, compactness, and ambient completeness from the rounded-filter read, while admission forces the completion-facing or Cauchy-facing route to display M, R, D, W, C, H, P, N before the boundary is consumed.

Finally apply the Cauchy-filter handoff. It attaches every $\mathsf{CauchyFilter}^{\uparrow}$ consumer to the same metric, dyadic, regular-rational, finite-window, real-seal, transport, replay, provenance, and local naming rows. Thus a filter-limit boundary cannot be read before those rows, and cannot export any of the excluded completion or compactness data. □

Theorem 1510.15 (FormalBall located-radius window)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *located-radius completion read must display the* $\mathsf{Met}^{\uparrow}$ *source* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *radius row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, and the* Cont *replay row* C *before any* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff is consumed. The read is therefore a finite located-radius window in the carrier* $B = (M, R, D, W, H, C, P, N)$*, not a quotient completion, selected limit, countable-choice schedule, open-cover compactness argument, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier located radius window`

Proof. Project the carrier of Definition 1510.1. The located-radius data available to a completion-facing consumer is exactly M, R, D, W together with transport, replay, provenance, and naming rows.

Apply the radius-refinement chain of Definition 1510.4. It keeps every displayed radius comparison inside the dyadic ledger D and the replayed finite window W, C over the same metric source M.

Finally apply the completion-window handoff of Definition 1510.5 and the Cauchy-filter handoff of Definition 1510.13. These handoffs force $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$ consumers to read the finite center-radius packet before the completion boundary, so no quotient completion, selected limit, choice schedule, compactness principle, or ambient completeness row can enter. □

Theorem 1510.16 (FormalBall rounded-filter admission)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer admitted through an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier must display the* $\mathsf{Met}^{\uparrow}$ *source* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *radius row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, and the* Cont *replay row* C *before reading a rounded-filter boundary. The admitted route also preserves componentwise transport, formal-order monotonicity, provenance, and local naming through* H, P, N*.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter admission`

Proof. Start with the NameCert obligations of Definition 1510.2. They make carrier admission, metric exposure, radius exposure, dyadic refinement, finite-window exposure, transport, replay, provenance, and naming part of one packet.

Use the radius-refinement route of Definition 1510.4 and the monotonicity boundary of Definition 1510.7. They force every rounded shrinkage to remain inside the dyadic ledger D and replay row C over the same source M and window W.

Finally apply the completion-window and Cauchy-filter handoffs, Definition 1510.5 and Definition 1510.13. A completion-facing consumer therefore reads the rounded-filter boundary only

after the displayed finite rows M, R, D, W, C, H, P, N, and receives no quotient completion, selected representative, choice schedule, compactness principle, or ambient completeness row. □

Theorem 1510.17 (FormalBall rounded-filter stability)**.** *Let $B = (M, R, D, W, H, C, P, N)$ be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. Any rounded-filter read admitted by Definition 1510.16 remains stable under componentwise* hsame *transport of the metric source M,* $\mathsf{RegSeqRat}^{\uparrow}$ *radius row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D,* $\mathsf{StreamName}^{\uparrow}$ *finite window W, and* Cont *replay row C. The transported read is again the rounded-filter boundary of Definition 1510.8 for the same displayed packet route.*

The stability route refuses quotient completion equality, selected limits, countable choice, open-cover compactness, and ambient completeness as evidence for the transported rounded-filter read.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter stability`

Proof. Start with the boundary definition. The rounded-filter row of Definition 1510.8 is projected from the displayed rows M, R, D, W, C, H, P, N, so every component available for transport is already named in the accepted packet.

Apply rounded-filter admission, Definition 1510.16. It forces any completion-facing consumer to read M, R, D, W, C before the rounded-filter boundary, with H, P, N preserving transport, provenance, and naming.

Finally apply directed radius transport, Definition 1510.19. Componentwise hsame transport keeps radius refinements inside the same dyadic ledger D and replay row C over the transported metric and window rows. Since the carrier contains no quotient equality, selected limit, choice schedule, compactness principle, or ambient complete-space coordinate, none can justify or alter the transported rounded-filter read. □

Theorem 1510.18 (FormalBall carrier admission)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *packet must display the metric source M, radius row R, dyadic comparison ledger D, and finite center-window row W before the local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *row N is read. Thus the admitted carrier route has the visible analytic prefix*

$$M, R, D, W$$

inside the BHist *packet, and no completion consumer may treat N as naming a formal ball without first reading those four rows.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier admission`

Proof. Project the accepted carrier of Definition 1510.1. Its displayed tuple is $B = (M, R, D, W, H, C, P, N)$, so the analytic rows M, R, D, W occur before the structural transport, replay, provenance, and naming rows.

Apply Definition 1510.2. The local NameCert surface names carrier admission only for the same displayed route and records no alternate entry point into N.

The completion handoff theorems then keep every $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{CauchyFilter}^{\uparrow}$ consumer downstream of that prefix. Hence an accepted packet must expose M, R, D, W before N can be consumed as the local name. □

Theorem 1510.19 (FormalBall directed radius transport)**.** *Let $B = (M, R, D, W, H, C, P, N)$ be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. If a displayed radius refinement is preserved componentwise by the* hsame *row H, then the transported refinement is still read through the same dyadic ledger D and replay row C. In particular, radius refinements cannot be transported by quotient completion equality, selected representatives, countable choice, open-cover compactness, or an ambient complete-space principle.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier directed radius transport`

Proof. Start with a displayed refinement. By Definition 1510.3, the comparison from the larger radius to the smaller one is recorded in D, and the corresponding center-window read is replayed through C over the same metric source.

Now apply the componentwise transport row H from the carrier. It transports the metric source, radius row, dyadic ledger, finite window, replay row, provenance row, and local naming row together; it does not supply an independent equality of completions or a selected representative.

The transported route therefore has the same radius-comparison evidence in D and the same replay evidence in C. Since Definition 1510.13 exhausts the consumer-facing fields by the displayed packet rows, no forbidden completion or compactness coordinate can justify the refinement. □

Theorem 1510.20 (FormalBall rounded ideal handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier reads only the rounded finite center-radius route generated by the metric source M, radius row R, dyadic ledger D, finite window W, componentwise transport H, replay row C, provenance row P, and local naming row N. The handoff exposes rounded formal-ball data, not quotient limits, selected Cauchy representatives, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded ideal handoff`

Proof. First read the carrier-admission route of Definition 1510.18. It forces every consumer to expose M, R, D, W before the local naming row can certify the packet.

Next use Definition 1510.4 and Definition 1510.19. Finite radius shrinkage remains inside the displayed dyadic ledger and replay route, while componentwise transport preserves the same center-radius packet.

Finally apply Definition 1510.5 and Definition 1510.13. These handoff statements exhaust the $\mathsf{CauchyFilter}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ reads by the finite rows M, R, D, W, H, C, P, N, so the exported rounded ideal interface contains no quotient limit, selected representative, choice schedule, compactness principle, or ambient completeness row. □

Theorem 1510.21 (FormalBall rounded-basis comparison)**.** *Let $B = (M, R, D, W, H, C, P, N)$ be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. Any consumer comparing a formal-ball rounded basis must display the center source M, radius row R, dyadic refinement ledger D, finite center-window row W, componentwise* hsame *transport H, and displayed* Cont *replay C before it reads the local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *row. The comparison is therefore a row-level comparison of finite center-radius windows, not quotient equality of balls, a selected limit, countable choice, open-cover compactness, or ambient completion.*

Proof. Project the accepted carrier. The rows M, R, D, W display the metric center surface, the regular rational radius, the dyadic refinement budget, and the finite observation window.

Rounded-basis comparison can only shrink or compare those displayed radii through D and replay the corresponding center windows through C. Componentwise transport through H preserves the same source, radius, dyadic, and window rows.

The local name and provenance rows P, N do not add analytic data. Hence the comparison remains a finite formal-ball basis comparison and cannot become quotient equality of balls, a selected limit, a choice schedule, an open-cover compactness argument, or an ambient-completion principle. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded basis comparison`

Theorem 1510.22 (FormalBall Cauchy-net admission)**.** *Every* $\mathsf{CauchyNet}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *admission route that consumes an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier must read the center-radius rows* M, R*, the rounded refinement ledger* D*, the finite window* W*, and the replay row* C *before admitting Cauchy evidence to a completion-facing packet. The admitted Cauchy data is attached to the displayed* BHist *route*

$$M, R, D, W, H, C, P, N$$

and not to a quotient of formal balls or a selected representative.

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier cauchy net admission`

Proof. Begin with carrier admission. By Definition 1510.18, every consumer must expose M, R, D, W before the local NameCert row is read.

Use the rounded ideal handoff. The Cauchy-facing route reads rounded finite center-radius data through D, W, C over the same metric source and radius row, and H transports that data componentwise.

The admission row is therefore downstream of the finite packet. Since the carrier has no coordinate for quotient-ball equality, selected Cauchy representatives, selected limits, countable choice, open-cover compactness, or ambient completeness, neither a Cauchy net nor a Cauchy filter can obtain such evidence from the formal-ball packet. □

Theorem 1510.23 (FormalBall uniform-completion route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier factors through the displayed rounded-basis and Cauchy-admission rows before reading any completion handoff. The route first exposes* M, R, D, W*, then transports and replays the finite window through* H, C*, then records provenance and local naming through* P, N*; it does not quotient formal balls, select limits, use countable choice, or appeal to ambient completeness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier uniform completion route`

Proof. First apply the rounded-basis comparison. It fixes the center source, radius row, dyadic refinement ledger, and finite window as the visible formal-ball basis data.

Next apply Cauchy-net admission. Completion-facing Cauchy evidence is admitted only after the finite rounded basis route is displayed and replayed through C with componentwise transport H.

Finally use the existing completion-window and rounded-ideal handoffs. They send exactly the rows M, R, D, W, H, C, P, N to uniform or metric completion consumers, so the route has no place for quotient formal-ball equality, selected limits, countable choice, open-cover compactness, or an ambient complete-space assumption. □

Theorem 1510.24 (FormalBall filter-basis route)**.** *Every* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier must first read the center-radius rows* M, R*, the rounded refinement ledger* D*, the finite center window* W*, and the replay row* C*. The basis row is therefore admitted only after the rounded-basis comparison and Cauchy-net admission surface have been exposed; it cannot be consumed as a detached filter basis, a quotient of balls, a selected Cauchy representative, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier filter basis route`

Proof. Start with Definition 1510.21. It fixes the center source, radius row, refinement ledger, finite window, transport, and replay rows as the only basis-facing data.

Apply Definition 1510.22. Any Cauchy-facing basis read is downstream of the displayed route M, R, D, W, H, C, P, N, so the basis cannot enter as an external filter object.

Finally use Definition 1510.23. Completion consumers receive the basis only through the same formal-ball packet, with no row for quotient equality, selected representatives, choice schedules, or ambient completeness. □

Theorem 1510.25 (FormalBall radius-order handoff)**.** *Every* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *comparison handed off by an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier is read through the center-radius rows* M, R*, the rounded refinement ledger* D*, and the replayed window* W, C *before any* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer may inspect it. The comparison is a finite radius-order row inside the same packet, not a host order on quotient balls or a selected completion representative.*

Proof. Apply Definition 1510.4. The radius comparison is recorded in the dyadic ledger D and is tied to the same center-radius route.

Use Definition 1510.19. Componentwise hsame transport preserves the comparison together with M, R, W, C, P, N rather than replacing it by a quotient-level order.

The completion route of Definition 1510.23 then forces every completion-facing consumer to read this radius-order handoff before it reads the terminal completion surface. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier radius order handoff`

Theorem 1510.26 (FormalBall completion-basis nonescape)**.** *Let* $B = (M, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier. Any* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer that uses* B *must first read the center-radius rows, the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *comparison row encoded by* D*, and the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row encoded by the rounded Cauchy admission route. The consumer cannot bypass those rows to a quotient completion, selected limit, open-cover compactness proof, countable choice principle, or ambient complete-space witness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier completion basis nonescape`

Proof. First combine the filter-basis route with the radius-order handoff. The basis read is downstream of M, R, D, W, H, C, P, N, and the radius comparison is stored in D over the same center-radius rows.

Next apply the rounded ideal and Cauchy-net admission theorems. They attach all completion-facing Cauchy evidence to the displayed finite packet and replay it through C.

Finally apply the uniform-completion route. Since that route exposes no coordinate beyond M, R, D, W, H, C, P, N, the completion consumer cannot escape to quotient equality, selected limits, compactness, choice, or ambient completeness. □

Theorem 1510.27 (FormalBall rounded-filter admission package)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier* $B = (M, R, D, W, H, C, P, N)$ *whose center-radius row, directed radius transport row, rounded ideal handoff row, filter-basis route row, radius-order handoff row, and completion-basis nonescape row are displayed admits the rounded-filter package read by completion consumers. The package passes through* BHist, hsame, Cont, Pkg, NameCert, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$*, and* $\mathsf{UniformCompletion}^{\uparrow}$*, and exports no quotient Cauchy name, selected limit, countable-choice schedule, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier rounded filter admission package`

Proof. Project the carrier of Definition 1510.1. The analytic prefix M, R, D, W is visible by Definition 1510.18, so the center-radius read starts from a displayed $\mathsf{Met}^{\uparrow}$ source, a regular rational radius, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, and a finite window.

Apply directed radius transport and the rounded ideal handoff. Theorems 1510.19 and 1510.20 keep radius shrinkage inside the same dyadic ledger, componentwise hsame transport, and replay row C.

Next read the Cauchy-facing basis rows. Theorems 1510.22 and 1510.24 attach $\mathsf{CauchyNet}^{\uparrow}$ and $\mathsf{CauchyFilterBasis}^{\uparrow}$ evidence to the displayed packet rather than to an external quotient or representative.

Finally apply the radius-order handoff and completion-basis nonescape, Definition 1510.25 and Definition 1510.26. These rows exhaust the completion-facing package by M, R, D, W, H, C, P, N, so the admitted rounded-filter package exports no quotient Cauchy name, selected limit, countable choice, or ambient completeness. □

Theorem 1510.28 (FormalBall completion obligation triad)**.** *Let $B = (M, R, D, W, H, C, P, N)$ be an accepted $\mathsf{FormalBall}^{\uparrow}$ carrier. A completion-facing read of B is admitted only when the center-radius row M, R, the $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ comparison row encoded by D, and the $\mathsf{CauchyFilterBasis}^{\uparrow}$ basis route through W, H, C are all displayed before any $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads the packet. The three rows form one finite BEDC route through M, R, D, W, H, C, P, N and do not introduce a quotient Cauchy name, selected limit, countable-choice schedule, open-cover compactness theorem, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier completion obligation triad`

Proof. First project the rounded-filter package of Definition 1510.27. It exposes the center-radius source, dyadic refinement ledger, finite window, transport, replay, provenance, and local name before any completion handoff is available.

Next apply the radius-order handoff, Definition 1510.25. The comparison read is stored in the same dyadic row D over M, R and is replayed with W, C rather than with a quotient order on formal balls.

Finally use completion-basis nonescape, Definition 1510.26. It forces the Cauchy-filter basis row and completion consumer to remain on M, R, D, W, H, C, P, N. Thus the center-radius row, radius-order comparison, and Cauchy basis route are a single finite obligation triad, with no selected limit, quotient name, choice schedule, compactness principle, or ambient completeness coordinate. □

Theorem 1510.29 (FormalBall radius-order scope package)**.** *Let $B = (M, R, D, W, H, C, P, N)$ be an accepted $\mathsf{FormalBall}^{\uparrow}$ carrier. Its radius-order scope package consists of the center-radius source M, R, the $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ comparison row stored in D, the $\mathsf{MetricClosedBall}^{\uparrow}$ boundary read available from the same center and radius window, the $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, and the terminal $\mathsf{Real}^{\uparrow}$ tolerance rows replayed by C. A completion-facing consumer may inspect a radius-order comparison only through this package; it cannot replace the package by a host order on balls, quotient equality of completions, a selected center representative, or an ambient complete-space witness.*

Proof. Project the accepted carrier. The center and radius rows are M, R, the dyadic comparison ledger is D, and the finite window and replay rows are W, C.

Use the radius-order handoff, Definition 1510.25. It places every $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ comparison inside the same dyadic row D and ties the comparison to the center-radius route rather than to a quotient order.

Now read the metric closed-ball boundary and real tolerance rows through the same center, radius, window, and replay data. Componentwise transport through H and provenance through P, N preserve that package, so the completion consumer receives only the displayed L10 dependency surface. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier radius order scope package`

Theorem 1510.30 (FormalBall Cauchy filter-basis triad)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier that is consumed by a completion-facing route must expose three Cauchy-facing rows before the completion handoff: a rounded-filter route over the displayed center-radius packet, a* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row read through the same finite windows, and a* $\mathsf{CauchyNet}^{\uparrow}$ *basis row with the same dyadic tolerance ledger. The triad is attached to* M, R, D, W, H, C, P, N *and does not introduce detached filters, quotient Cauchy names, selected limits, countable-choice schedules, or ambient completeness.*

Proof. Start with the rounded ideal handoff, Definition 1510.20. It forces the rounded-filter route to read the finite center-radius data M, R, D, W, H, C, P, N.

Apply Cauchy-net admission and the filter-basis route. Theorems 1510.22 and 1510.24 attach $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, and basis evidence to the same rounded formal-ball packet before any completion row is consumed.

Finally use the completion obligation triad. The completion-facing read sees the rounded-filter, filter-basis, and net-basis data as one finite L10 surface, with no coordinate for detached filters, quotient names, selected limits, choice schedules, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier cauchy filter basis triad`

Theorem 1510.31 (FormalBall uniform-completion ledger exhaustion)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff from an accepted* $\mathsf{FormalBall}^{\uparrow}$ *carrier is exhausted by radius refinement, finite-window replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *row. Equivalently, the handoff consumes only* M, R, D, W, H, C, P, N *together with the local radius-order, Cauchy-filter, Cauchy-net,* $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *tolerance rows already displayed by the packet. It exports no quotient completion equality, selected representative, countable-choice schedule, open-cover compactness, or ambient complete-space assumption.*

Proof. Begin with the uniform-completion route, Definition 1510.23. That route already factors the completion consumer through M, R, D, W, H, C, P, N before any completion handoff is read.

Next add the L10 dependency rows. The radius-order scope package supplies the radius comparison, metric closed-ball boundary, dyadic ledger, and real tolerance seal, while the Cauchy filter-basis triad supplies the rounded-filter, $\mathsf{CauchyFilter}^{\uparrow}$, and $\mathsf{CauchyNet}^{\uparrow}$ basis data.

It remains to account for structural stability. Radius refinement is recorded in D, finite windows are replayed through W, C, componentwise hsame transport is carried by H, and provenance and local naming are exactly P, N.

Thus the displayed ledger is exhausted by the accepted packet itself. Since no field remains for quotient completion equality, selected representatives, countable choice, open-cover compactness, or ambient completeness, the $\mathsf{UniformCompletion}^{\uparrow}$ handoff cannot export those objects. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier uniform completion ledger exhaustion`

Theorem 1510.32 (FormalBall obligation-closure export)**.** *An accepted* $\mathsf{FormalBall}^{\uparrow}$ *packet exports an obligation-closure surface exactly when its carrier admission, radius refinement, rounded-filter*

boundary, Cauchy-filter handoff, radius-order handoff, completion-basis nonescape, and uniform-completion route all factor through the same finite rows

$$M, R, D, W, H, C, P, N.$$

The exported surface is the shared row-ordering of Definition 1510.18, Definition 1510.4, Definition 1510.16, Definition 1510.13, Definition 1510.25, Definition 1510.26, and Definition 1510.23; it contains no quotient Cauchy name, selected limit, countable-choice schedule, open-cover compactness row, or ambient completeness witness.

Proof. Project carrier admission through Definition 1510.18. It places every accepted read on the displayed formal-ball rows M, R, D, W, H, C, P, N before local naming is consumed.

Next read the radius route. The radius-refinement chain and directed transport theorems keep every shrinkage and comparison inside the dyadic ledger D, replay row C, and the same center-window data.

Now attach the rounded-filter and Cauchy-facing boundaries. The rounded-filter admission theorem and the Cauchy-filter handoff theorem force filter consumers to read the same finite packet before any completion-facing row is available.

Finally apply the radius-order handoff, completion-basis nonescape, and uniform-completion route. These theorems order the remaining completion consumers over the same rows, so the packet exports one obligation surface precisely when all listed routes factor through that row list. □

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier obligation closure export`

Theorem 1510.33 (FormalBall public completion-interface export)**.** *Every accepted* $\mathsf{FormalBall}^{\uparrow}$ *packet exports a public completion-facing interface exactly through* $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{FormalBall}^{\uparrow}}$ *rows. A* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer may read the center-radius packet, dyadic radius order, finite stream windows, regular rational tolerance route, and real tolerance seal, but it cannot read quotient completion equality, selected limits, countable choice, open-cover compactness, or ambient completeness from the FormalBall surface.*

Lean4 ✓ `BEDC.Derived.FormalBallUp.FormalBallCarrier public completion interface export`

Proof. Start with the obligation-closure export. Theorem 1510.32 places carrier admission, radius refinement, rounded-filter boundary, Cauchy-filter handoff, radius-order handoff, completion-basis nonescape, and uniform-completion routing on the same rows M, R, D, W, H, C, P, N.

Next read the radius-order scope package. It exposes the metric source, dyadic radius ledger, regular rational tolerance route, stream-window readback, and real tolerance seal without adding an external completion object.

Attach the Cauchy filter-basis triad and the uniform-completion ledger exhaustion theorem. Theorems 1510.30 and 1510.31 show that $\mathsf{CauchyFilter}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers receive only those displayed rows, transported by hsame, replayed by Cont, and sealed by Pkg and the local $\mathsf{NameCert}$ row.

Thus the public completion-facing interface is the listed finite BEDC surface. Since every completion consumer is already routed through that surface, no quotient completion equality, selected limit, countable-choice schedule, open-cover compactness proof, or ambient completeness witness remains available. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBall}\uparrow}$.

Constructive story: A $\mathsf{FormalBall}^{\uparrow}$ packet is generated from a displayed MetricSpaceUp source, a RegSeqRat radius row, a DyadicRatCore refinement ledger, a StreamName finite-window row, componentwise hsame transport, displayed Cont replay toward RealUp tolerance seals and UniformCompletionUp handoff, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalBall}\uparrow}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1510.1, Definition 1510.2, the radius-order scope package, the metric and dyadic-radius rows, the Cauchy filter-basis triad, the completion-basis nonescape row, the uniform-completion ledger exhaustion theorem, and the public completion-interface export bind the completed finite formal-ball observation surface for constructive completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalBall}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Mature closure requires an explicit standard-bridge schema tying the public completion-interface export to the accepted bridge rows while retaining the same finite packet inventory.

Chapter 1511

A Concrete Naming Certificate for $\mathsf{CauchyRealPresentation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRealPresentation}^{\uparrow}$ packet records the plain Cauchy-real presentation surface used before any completion-facing seal is consumed. It sits over the dyadic arithmetic row of chapter 291, the finite stream-window row of chapter 126, the regular rational readback row of chapter 301, and the terminal real seal of chapter 38. It is the presentation face adjacent to the Cauchy sequence space of chapter 406, the located Bishop-real route of chapter 343, and the phase exit certificate of chapter 1327.

Definition 1511.1 (CauchyRealPresentation carrier)**.** *A* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose entries are the only visible Cauchy-name observations.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic scales, comparison budget, and rational arithmetic used by the presentation.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S under the tolerance data in D.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed route through S, D, R has been admitted.*
5. *H records componentwise* hsame *transport for the stream-window, dyadic-tolerance, regular-rational readback, real-seal, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each finite window and tolerance request through S, D, R, E and then through the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyRealPresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealPresentation}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient stream equality, a selected representative, host real equality, countable choice, or ambient completeness.

Theorem 1511.2 (CauchyRealPresentation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *carrier* $Q = (S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealPresentation}^{\uparrow}}$ *surface consists exactly of the following seven obligation rows:*

1. *carrier admission for the displayed stream-window, dyadic-tolerance, regular-rational readback, real-seal, transport, continuation, provenance, and local naming rows;*
2. *classifier transport by common finite windows, requiring any comparison of accepted carriers to inspect only matching windows of* S *preserved by* H *and replayed by* C*;*
3. *dyadic tolerance exactness, requiring each tolerance request to be accounted for by the row* D *before readback is available;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring* R *to be read only downstream of the displayed windows and dyadic tolerance data;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring* E *to remain terminal with respect to the route* S, D, R *and the structural rows;*
6. *provenance, requiring the package row* P *to name the same presentation packet and dependency rows;*
7. *local naming, requiring the row* N *to bind the accepted packet as a* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *certificate.*

No obligation is stored outside these rows.

Proof. Carrier admission is tuple projection from Definition 1511.1. The tuple displays exactly the stream-window, dyadic-tolerance, regular-rational readback, real-seal, transport, replay, provenance, and local name.

The analytic obligations are forced by S, D, R, E. Finite-window classification reads S, dyadic tolerance exactness reads D, regular-rational readback reads R, and the terminal seal reads E only after those rows have been displayed.

The structural obligations are forced by H, C, P, N. Transport and replay preserve the same finite windows and tolerance requests, provenance records the same dependency rows, and the local name binds the accepted packet. Since the carrier has no coordinate beyond these rows, the listed obligations exhaust the local NameCert surface. □

Theorem 1511.3 (CauchyRealPresentation Real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *carrier* $Q = (S, D, R, E, H, C, P, N)$ *factors through the displayed finite route*

$$S \longmapsto D \longmapsto R \longmapsto E$$

together with the stored hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. The factorization exports no quotient stream, selected representative, host real equality, countable choice, or ambient completeness principle.*

Proof. Start with an accepted Real-facing read. Carrier admission exposes the finite stream-window row S as the only source of Cauchy-name observations.

The tolerance and readback rows then determine the route. The dyadic ledger D records the scale and comparison budget for the request, and the row R is the only regular-rational readback available from those displayed windows and tolerances.

The read reaches the real boundary only through E. The rows H and C transport and replay the same finite route, while P and N record provenance and local naming. The carrier fields are

exhausted by these rows, so the factorization cannot export quotient streams, selected representatives, host real equality, countable choice, or ambient completeness. □

Theorem 1511.4 (CauchyRealPresentation phase-exit handoff)**.** *An accepted* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *carrier supplies the Cauchy-presentation face consumed by* $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ *only through the four local L10 records* S, D, R, E*: finite stream windows, dyadic tolerance rows, regular-rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff may be transported, replayed, packaged, and locally named by* H, C, P, N*, but it does not add quotient stream equality, a selected representative, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted packet and read the four L10 records in order. The stream row S supplies finite windows, the dyadic row D supplies tolerance arithmetic, the readback row R supplies regular rational data, and the seal row E is terminal.

Now pass to the phase-exit consumer. The continuation row can replay only this same route, and the transport row can preserve only its displayed components. The package and local naming rows record the same accepted carrier.

Thus $\mathsf{PhaseRealCompletionExit}^{\uparrow}$ receives a Cauchy-presentation face through S, D, R, E and no additional analytic principle. The excluded quotient, representative, host equality, choice, and completeness data are absent from the carrier itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealPresentation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealPresentation}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealPresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1511.1, Definition 1511.2, Definition 1511.3, and Definition 1511.4 seed the finite Cauchy-real presentation surface used by the real-completion phase exit.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealPresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport by common finite windows, dyadic tolerance exactness, RegSeqRat readback, RealUp seal nonescape, provenance, and local naming for the same displayed packet.

Chapter 1512

A Concrete Naming Certificate for $\mathsf{CompletionEmbedding}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki treatment of completion by separated metric envelopes, the $\mathsf{CompletionEmbedding}^{\uparrow}$ packet names the dense isometric embedding surface of completion as finite BEDC data. It sits after the metric-completion surface of chapter 1506, the displayed isometry route of chapter 439, and the regular Cauchy dense embedding boundary of chapter 1453. The packet binds those sibling surfaces before any downstream consumer treats a source point, regular Cauchy name, or Real seal as a point of a completed host space.

Definition 1512.1 (CompletionEmbedding carrier)**.** *A* $\mathsf{CompletionEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, D, I, R, S, A, L, C, P, N)$$

with the following visible rows:

1. *X is the source metric row whose point classifier and distance witnesses are the domain of the embedding.*
2. *Y is the completion target row, read through the finite* $\mathsf{MetricCompletion}^{\uparrow}$ *surface rather than through an ambient complete metric space.*
3. *D is the dense image row recording that every displayed completion-facing tolerance request is met by a source-side image window.*
4. *I is the isometry row, read through* $\mathsf{IsometricEmbedding}^{\uparrow}$ *distance reflection and not through host function equality.*
5. *R is the regular-Cauchy source row supplying the finite rational or regular source names used to approach target reads.*
6. *S is the Hausdorff separated boundary row, comparing two target reads only through displayed zero-distance classifier evidence.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the regular Cauchy and distance-reflection routes.*
8. *L records componentwise* hsame *transport for the source metric, completion target, dense image, isometry, regular-Cauchy source, Hausdorff boundary, Real seal, continuation, provenance, and local naming rows.*

9. *C records the displayed* Cont *replay through $X, Y, D, I, R, S, A, L, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, host equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingTasteGate single carrier alignment`

Theorem 1512.2 (CompletionEmbedding NameCert obligations)**.** *For an accepted* $\mathsf{CompletionEmbedding}^{\uparrow}$ *carrier $E = (X, Y, D, I, R, S, A, L, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionEmbedding}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed source metric, completion target, dense image, isometry, regular-Cauchy source, Hausdorff separated boundary, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *dense-image coverage, requiring each completion-facing tolerance request in Y to route through the image row D and a regular-Cauchy source read in R;*
3. *isometric reflection, requiring each source-target distance comparison to pass through I and then through the* $\mathsf{Real}^{\uparrow}$ *seal row A;*
4. *Hausdorff separated comparison, requiring any identification of two target reads to pass through S after the dense and isometric rows have been displayed;*
5. *ledger non-escape, stating that no quotient completion, selected representative, host equality, countable-choice schedule, or ambient completeness theorem is exported by the packet.*

Proof. Carrier admission is projection from Definition 1512.1. The tuple lists the source, target, dense-image, isometry, regular-Cauchy, Hausdorff, Real-seal, transport, replay, provenance, and naming rows.

Dense-image coverage reads a completion-facing tolerance request in Y through D before a target read is accepted. The row R supplies the finite regular-Cauchy source data used by that image window.

Isometric reflection is then read through I and A. The isometry row preserves the source distance route, while the Real seal row records the displayed distance comparison consumed by later completion readers.

Hausdorff separated comparison is local to S. The continuation row C replays the already displayed dense and isometric route before S compares target reads, and the package row P and naming row N record that no excluded completion or choice principle is available. □

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier namecert obligations`

Theorem 1512.3 (CompletionEmbedding dense isometric handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionEmbedding}^{\uparrow}$ *packet factors through the finite route*

$$X \longmapsto R \longmapsto D \longmapsto I \longmapsto A \longmapsto Y \longmapsto S \longmapsto (L, C, P, N).$$

Thus the consumer receives a dense source image and a distance-reflecting route into the completion target, not a quotient-completion point, not a selected representative, not host equality, and not an ambient completeness principle.

Proof. Start with the source metric row X and the regular-Cauchy row R. These two rows provide the displayed source-side data from which the completion-facing image can be read.

The dense image row D is the next route component. A consumer cannot read a target point from Y before the dense source image has been displayed through D.

The isometry row I and Real seal row A then attach distance preservation to that image. The target row Y is reached only after this dense isometric route has been recorded.

Finally the Hausdorff row S compares target reads through separated zero-distance evidence, while L, C, P, N transport, replay, source, and name the same packet. Since the route has no field for quotient classes, selected representatives, host equality, countable choice, or ambient completeness, the handoff exports only the displayed dense isometric surface. □

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier dense isometric handoff`

Theorem 1512.4 (CompletionEmbedding Real seal non-escape)**.** *For every accepted* $\mathsf{CompletionEmbedding}^{\uparrow}$ *packet* $E = (X, Y, D, I, R, S, A, L, C, P, N)$*, each* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by* E *factors through the regular-Cauchy source row* R*, the dense image row* D*, the isometry row* I*, and the Hausdorff separated boundary row* S*. The Real seal cannot be used as a detached completed point, quotient-class representative, host equality witness, countable-choice selection, or ambient completeness witness.*

Proof. Read the Real seal coordinate A from the carrier. It is listed after the regular-Cauchy source row, dense image row, isometry row, and Hausdorff separated boundary row, so the packet exposes it only as part of that displayed route.

Apply the NameCert obligations of Definition 1512.2. Dense-image coverage places R and D before any completion-facing read, isometric reflection places I before the Real distance comparison, and separated comparison places S around target-read identification.

The replay and provenance rows C and P record no second Real route. Transport by L preserves the same fields componentwise, and the local naming row N names exactly this packet.

Therefore any exported $\mathsf{Real}^{\uparrow}$ read remains inside $R, D, I, S, A, L, C, P, N$. The excluded detached point, quotient representative, host equality, choice schedule, and ambient completeness data have no coordinate from which a consumer could read them. □

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier real seal nonescape`

Theorem 1512.5 (CompletionEmbedding source-density reflection)**.** *For every accepted* $\mathsf{CompletionEmbedding}^{\uparrow}$ *packet* $E = (X, Y, D, I, R, S, A, L, C, P, N)$*, any completion-target read that the packet classifies back to a source-side approximation factors through the regular-Cauchy source row* R*, dense image row* D*, isometry row* I*,* $\mathsf{Real}^{\uparrow}$ *seal row* A*, and Hausdorff separated boundary row* S*. The resulting comparison is a local classifier reflection recorded by* L, C, P, N*, not host equality, quotient equality, selected representatives, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier source density reflection`

Proof. Begin with the accepted carrier and its NameCert obligations. The dense-image coverage obligation requires a completion-facing tolerance read in Y to pass first through the source-side regular-Cauchy data in R and the dense image window in D.

Use the dense isometric handoff of Definition 1512.3 to extend that source-density route through I and A. The isometry row supplies the distance-reflecting comparison, and the $\mathsf{Real}^{\uparrow}$ seal records the displayed real-valued boundary consumed by completion readers.

The Real-seal non-escape theorem, Definition 1512.4, keeps that comparison inside the same finite route. The separated boundary row S is therefore the only target-read comparison surface available after the dense and isometric rows have been replayed.

Finally, L, C, P, N transport, replay, source, and name the reflected classifier route. Since the carrier has no coordinate for host equality, quotient equality, selected representatives, countable choice, or ambient completeness, the source-density read exports only the local reflection described above. □

Theorem 1512.6 (CompletionEmbedding obligation closure package)**.** *The carrier of Definition 1512.1, the NameCert obligations of Definition 1512.2, the dense-isometric handoff of Definition 1512.3, the Real-seal non-escape boundary of Definition 1512.4, and the source-density reflection of Definition 1512.5 jointly form the* obligation *package for* $\mathsf{CompletionEmbedding}^{\uparrow}$ *over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package keeps the existing exclusions: it exports no quotient completion, selected representative, host equality, countable-choice schedule, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier obligation closure package`

Proof. First take carrier admission from Definition 1512.1. The displayed rows are exactly the source metric, completion target, dense image, isometry, regular-Cauchy source, Hausdorff separated boundary, Real seal, componentwise transport, continuation replay, provenance, and local naming data.

Next apply Definition 1512.2. It turns those rows into the local NameCert surface: dense-image coverage, isometric reflection, Hausdorff separated comparison, and ledger non-escape are all obligations on the same finite packet.

The public route is supplied by Definition 1512.3. It sends every completion-facing consumer through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any target read is accepted.

Finally combine Definition 1512.4 with Definition 1512.5. The first keeps Real-facing reads inside the same route, and the second reflects target reads back to the source-density classifier. The excluded quotient, selected-representative, host-equality, choice, and ambient-completeness data therefore remain absent throughout the obligation package. □

Theorem 1512.7 (CompletionEmbedding public interface)**.** *The public* $\mathsf{CompletionEmbedding}^{\uparrow}$ *interface for downstream completion consumers is the package consisting of the carrier of Definition 1512.1, the NameCert obligations of Definition 1512.2, the dense isometric handoff of Definition 1512.3, the Real seal non-escape theorem of Definition 1512.4, the source-density reflection of Definition 1512.5, and the obligation closure package of Definition 1512.6.*

Every public consumer must read $\mathsf{CompletionEmbedding}^{\uparrow}$ *over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *through that package. The interface exports dense source coverage, distance reflection, source-density reflection, and Real-seal non-escape, but exports no quotient completion, selected representative, host equality, countable-choice schedule, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier public interface`

Proof. Project the accepted public consumer request through Definition 1512.1. The only displayed source rows are the source metric, completion target, dense image, isometry, regular-Cauchy source, Hausdorff boundary, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1512.2. The NameCert surface admits exactly those rows and records dense-image coverage, isometric reflection, Hausdorff separated comparison, and ledger non-escape.

Use Definition 1512.3 for the public handoff. It routes completion consumers through the dense source image and distance-reflecting embedding before a target read is accepted.

The Real-facing and source-facing public boundaries are supplied by Definition 1512.4 and Definition 1512.5. These theorems keep Real seal reads and target-source reflections inside the same finite route.

Finally package the route with Definition 1512.6. Since every cited row belongs to that package and the excluded data have no carrier coordinate, the named package is the public $\mathsf{CompletionEmbedding}^{\uparrow}$ interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionEmbedding}^{\uparrow}$ packet is generated from a source metric row, a completion target row, a dense image row, an isometry row, a regular-Cauchy source row, a Hausdorff separated boundary row, a Real seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionEmbedding}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1512.7, Definition 1512.3, Definition 1512.5, and Definition 1512.6 give the mature paper treatment of the dense isometric completion-embedding interface over $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionEmbedding}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CompletionEmbeddingUp.CompletionEmbeddingCarrier\_public\_interface`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality, countable choice, or ambient completeness.

Upgrade path: Bridge checking remains separate from the mature paper treatment displayed here.

Chapter 1513

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ packet records the finite regular-Cauchy completion surface needed when a real-name consumer wants both separatedness and uniqueness in the same displayed route. It reads the regular source of chapter 301, the finite-window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the metric-completion handoff of chapter 1506, the zero-distance classifier route of chapter 1411, the completion uniqueness ledger of chapter 1463, and the regular universal-property surface of chapter 1425.

The packet is a local L10 bridge. It names the rows by which regular Cauchy names enter a completion-facing metric route, get compared through separated zero-distance evidence, and then feed the uniqueness ledger. It does not turn regular names into quotient streams or assert completeness of an ambient host real line.

Definition 1513.1 (RegularCauchyCompletionSeparated carrier)**.** *A* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, R, M, Z, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose accepted histories are regular rational Cauchy names.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect those regular names at bounded precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the windows of W.*
4. *R is the regular-name comparison row, read through the existing regular Cauchy comparison surface and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that exposes completion-facing Cauchy evidence without quotienting the source names.*
6. *Z is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance and separation row used to identify displayed limit readings inside the metric classifier.*
7. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *ledger comparing completion-facing presentations over the same regular source, window schedule, tolerance ledger, comparison row, metric handoff, and separatedness row.*

8. *H records componentwise* hsame *transports for S, W, D, R, M, Z, U and the naming support rows.*
9. *C records the displayed* Cont *replay from regular names to windows, tolerances, comparison, metric completion, separated zero-distance evidence, and uniqueness ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}}$ *row.*

The carrier stores no quotient stream, no selected host limit, no host real equality coordinate, no countable-choice selector, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no open-cover compactness principle, and no trichotomy row.*

Theorem 1513.2 (RegularCauchyCompletionSeparated NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *carrier $Q = (S, W, D, R, M, Z, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCo}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed regular source, finite windows, dyadic tolerance ledger, regular comparison row, metric-completion handoff, separatedness row, completion uniqueness ledger, transports, replay routes, provenance, and local naming row;*
2. *classifier stability under the componentwise* hsame *transports H and the replay routes C;*
3. *regular-source compatibility, requiring completion consumers to read S, W, D, R before the metric handoff M;*
4. *separatedness compatibility, requiring any identification of displayed completion readings to pass through the zero-distance row Z;*
5. *uniqueness-ledger exactness, requiring comparison of two completion-facing presentations to be exhausted by the ledger U over the same displayed route.*

Proof. Carrier admission is projection from Definition 1513.1. The tuple displays the regular source, stream windows, dyadic tolerances, regular comparison row, metric-completion handoff, separatedness row, uniqueness ledger, and the structural support rows.

Classifier stability is componentwise. The transports in H move each displayed coordinate without changing the order of dependency, and the continuation row C replays the same route after transport.

The remaining obligations are sibling handoffs. The regular-source compatibility clause reads through S, W, D, R before M; the separatedness clause reads through Z after the metric-completion handoff; and the uniqueness exactness clause reads through U only after the same finite route has been projected.

No carrier coordinate contains quotient streams, selected host limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, or trichotomy. The listed obligations therefore exhaust the local NameCert surface. □

Theorem 1513.3 (Regular separated completion route)**.** *Let $Q = (S, W, D, R, M, Z, U, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *packet. Every separated comparison exported by Q factors through the route*

$$S, \quad W, \quad D, \quad R, \quad M, \quad Z.$$

Thus a completion-facing consumer identifies two displayed readings only by projecting the regular source, scheduled windows, tolerance ledger, comparison row, metric-completion handoff, and zero-distance separation row. The conclusion is classifier equivalence inside the displayed separated metric surface, not equality of quotient streams or equality in a host real line.

Proof. Project the accepted carrier. The first four rows S, W, D, R keep the comparison attached to a regular Cauchy name inspected through finite windows and dyadic tolerances.

Read the metric-completion handoff only after those rows have been fixed. The handoff M supplies completion-facing evidence over the displayed regular route, so a consumer cannot replace it by a selected host limit.

Apply the separatedness row Z to the displayed zero-distance reading. The replay row C exposes exactly this route, and the transport row H moves it componentwise without adding a host equality coordinate.

Since the packet carries no quotient stream, host real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness field, every separated comparison is the displayed finite route. □

Theorem 1513.4 (Regular separated uniqueness ledger)**.** *For accepted packets $Q = (S, W, D, R, M, Z, U, H, C, P, N$ and $Q' = (S, W, D, R, M, Z, U', H', C', P, N')$ over the same regular source, window schedule, tolerance ledger, comparison row, metric-completion handoff, separatedness row, and provenance, any comparison between their completion-facing exports is exhausted by the uniqueness ledgers U and U' together with their displayed transport and replay rows.*

Proof. Start from the shared coordinates S, W, D, R, M, Z, P. They fix the regular Cauchy source, the finite windows, the dyadic tolerance ledger, the comparison row, the metric-completion handoff, the separatedness row, and the provenance surface for both packets.

The remaining comparison data are U, H, C, N and U', H', C', N'. The ledgers U and U' compare completion-facing presentations over the same shared route, while H, H' and C, C' display the permitted transports and replays.

No further field remains from which a consumer could read quotient-stream equality, a selected host limit, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, or trichotomy. The uniqueness judgement is therefore exactly the local ledger over the displayed regular separated completion route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1513.1, Definition 1513.2, Definition 1513.3, and Definition 1513.4 seed the regular Cauchy completion route with separatedness and uniqueness rows over a finite L10 packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, host real equality, selected host limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, trichotomy, or a standard bridge.

Upgrade path: A later verification route can encode the carrier and prove carrier admission, classifier stability, separation routing, and uniqueness-ledger exactness for the same displayed packet.

Constructive story: A $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ packet is generated from a regular rational Cauchy source, StreamName windows, a DyadicRatCore tolerance ledger, a regular-name comparison row, a MetricCompletion handoff, a SeparatedMetric zero-distance row, a CompletionUniqueness ledger, componentwise transports, replay routes, provenance, and a local NameCert row.

Chapter 1514

A Concrete Naming Certificate for $\mathsf{ApartnessSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ApartnessSpace}^{\uparrow}$ packet records the constructive neighbour relation needed between located positive-gap evidence and separated metric recognition. It reads the finite gap handoff of chapter 1380 and exposes the zero-distance boundary consumed by chapter 1411. The packet is an interface for positive apartness over displayed carriers; it is not a source of trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1514.1 (ApartnessSpace carrier)**.** *An* $\mathsf{ApartnessSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, L, G, T, C_0, C_1, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying carried-object row, including the local point classifier used by the packet.*
2. *L is the* $\mathsf{LocatedApartness}^{\uparrow}$ *source row that supplies two located observations, a shared finite window, and the displayed positive-gap evidence.*
3. *G is the positive-gap evidence row read from L as a finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap ledger attached to the two observations of X.*
4. *T is the apartness transport row stating that the positive-gap ledger is stable under componentwise transport of the two observations and their shared finite window.*
5. *C_0 is the classifier-exclusion row: if G is displayed between two observations, the local point classifier of X does not identify them.*
6. *C_1 is the zero-distance boundary row exported to* $\mathsf{SeparatedMetric}^{\uparrow}$*: any consumer that has only a displayed zero-distance route must pass through the separated-metric classifier boundary before it can identify observations.*
7. *H stores componentwise* hsame *transport for the carried-object row, the located-apartness source row, the positive-gap evidence, both classifier boundary rows, the continuation row, the provenance row, and the local naming row.*
8. *C stores the* Cont *replay from finite located observations to positive-gap evidence, from positive-gap evidence to classifier exclusion, and from zero-distance readings to the separated-metric boundary.*

9. *P is the* Pkg *provenance over* LocatedApartness$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{ApartnessSpace}^{\uparrow}}$ *row.*

The carrier records apartness as displayed positive-gap evidence over finite histories. It carries no trichotomy row, no quotient real equality, no host inequality, no selected limit, no open-cover compactness principle, and no Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceTasteGate single carrier alignment`

Definition 1514.2 (ApartnessSpace classifier)**.** *The classifier of an accepted* ApartnessSpace$^{\uparrow}$ *packet relates two carried observations a, b as apart when the packet displays a finite route*

$$a, b \longmapsto L(a, b) \longmapsto G(a, b) \longmapsto C_0(a, b).$$

The first step reads the two observations through the LocatedApartness$^{\uparrow}$ *source row, the second step extracts the positive* DyadicRatCore$^{\uparrow}$ *gap row, and the last step records classifier exclusion inside the carried-object surface X. Transport of this classifier is componentwise: if a, b, the shared window, and $G(a, b)$ are moved by the stored* hsame *rows, then the replayed* Cont *path produces the transported positive-gap evidence and the transported classifier-exclusion row.*

The classifier is local positive information. It does not decide every pair of observations, does not turn apartness into host inequality, and does not form a quotient equality surface for constructive reals.

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceClassifier positive gap exactness`

Theorem 1514.3 (ApartnessSpace NameCert obligations)**.** *For an accepted* ApartnessSpace$^{\uparrow}$ *carrier $A = (X, L, G, T, C_0, C_1, H, C, P, N)$, the local* NameCert$_{\text{ApartnessSpace}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the carried-object row, the* LocatedApartness$^{\uparrow}$ *source row, the positive-gap evidence row, the apartness transport row, the two classifier boundary rows, componentwise transport, continuation replay, package provenance, and local naming row;*

2. *classifier stability, stating that transported observations and transported finite windows replay the same positive-gap route and the same classifier-exclusion row;*

3. *positive-gap exactness, stating that every apartness assertion exported by the packet factors through the displayed finite G row read from L;*

4. *separated-metric boundary compatibility, stating that zero-distance consumers use only the displayed C_1 route into* SeparatedMetric$^{\uparrow}$ *and do not identify observations by host equality;*

5. *ledger visibility, stating that the packet exports no trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier namecert obligations`

Proof. Carrier admission is the finite tuple of Definition 1514.1. Each row named in the obligation list is visible in A, and the packet has no coordinate from which an excluded classical or completion principle could be read.

Classifier stability follows from the paired transport and replay rows. The hsame fields move the two observations, their shared finite window, the located-apartness source, the positive-gap

row, and the classifier boundary rows componentwise. The Cont row then replays the same finite route after transport.

Positive-gap exactness is the defining use of G. An apartness assertion exported by the packet must pass through L and then through the displayed positive dyadic gap row before it reaches the classifier-exclusion row C_0.

For zero-distance consumers, the only exported boundary is C_1. That row points toward the separated-metric classifier route, so it does not create host equality, quotient equality, or a total comparison principle inside the apartness-space packet. □

Theorem 1514.4 (ApartnessSpace located positive-gap route)**.** *Let A be an accepted* ApartnessSpace$^{\uparrow}$ *packet and let a, b be two carried observations of its object row X. If the* LocatedApartness$^{\uparrow}$ *row L displays a shared finite window for a, b and the positive* DyadicRatCore$^{\uparrow}$ *gap row $G(a, b)$ on that window, then A exports a constructive apartness route*

$$a, b \longmapsto L(a, b) \longmapsto G(a, b) \longmapsto C_0(a, b)$$

inside the local point classifier of X. The route is positive-gap evidence; it is not a proof of trichotomy, quotient real inequality, host inequality, a selected limiting comparison, open-cover compactness, or Real$^{\uparrow}$ *completeness.*

Proof. Read a and b through the located-apartness source row L. This exposes the two located observations, their shared finite window, and the finite endpoint data used by the positive-gap ledger.

Attach the displayed $G(a, b)$ row to that common window. By Definition 1514.1, G is already a row of the accepted packet and is transported componentwise with the two observations.

Replay the continuation route from $G(a, b)$ to the classifier-exclusion row $C_0(a, b)$. The resulting apartness route is therefore a finite BHist calculation inside the packet.

The final boundary is an inventory statement. Since the carrier has no row for trichotomy, quotient equality, host inequality, selected limits, open-cover compactness, or Real$^{\uparrow}$ completeness, none of those principles can be read from this positive-gap route. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier located positive gap route`

Theorem 1514.5 (ApartnessSpace positive-gap transport)**.** *Let A be an accepted* ApartnessSpace$^{\uparrow}$ *packet, and let a, b, a', b' be carried observations of its object row X. Suppose the two observations a, b, their transported observations a', b', the shared finite window, and the positive dyadic gap row are related componentwise by the displayed* hsame *transport of Definition 1514.1. Then the same route*

$$L \longmapsto G \longmapsto C_0$$

replays through Cont *for a', b'. The transported route is still positive apartness evidence only; it does not introduce trichotomy, host inequality, quotient real inequality, selected limits, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Proof. Start from the located positive-gap route of Definition 1514.4. That theorem reads the two observations, their shared located window, and the positive DyadicRatCore$^{\uparrow}$ gap row before reaching C_0.

Now apply the carrier inventory of Definition 1514.1 and the classifier surface of Definition 1514.2. The hsame row transports exactly the displayed observations, window, and gap row, while the Cont row replays the same finite continuation from L to G and then to C_0.

The replay has no additional carrier coordinate. Therefore the transported positive-gap evidence remains a local apartness route and cannot become a comparison principle, quotient real inequality, selected limit, compactness principle, or real-completeness theorem. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier positive gap transport`

Theorem 1514.6 (ApartnessSpace classifier stability)**.** *Let A be an accepted* ApartnessSpace$^{\uparrow}$ *packet. If two observations, their shared finite window, and the displayed positive-gap row are transported componentwise by the* hsame *rows of Definition 1514.1, then the transported route preserves classifier exclusion through the same local C_0 row. In particular, the transported located positive-gap evidence remains apartness evidence and cannot be reread as trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Proof. Start with the transported positive-gap route of Definition 1514.5. That route carries the two observations, their shared finite window, and the positive DyadicRatCore$^{\uparrow}$ gap row through the same $L \longmapsto G \longmapsto C_0$ spine.

Now read the classifier definition in Definition 1514.2. The final row is classifier exclusion inside the carried-object surface X, so componentwise transport preserves exclusion by replaying the displayed Cont path rather than by adding a new comparison principle.

The carrier inventory has no coordinate for trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or Real$^{\uparrow}$ completeness. Thus the transported classifier route remains exactly the positive apartness route named by C_0. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier classifier stability`

Theorem 1514.7 (ApartnessSpace transport and replay boundary)**.** *For an accepted* ApartnessSpace$^{\uparrow}$ *packet,* hsame *transport followed by displayed* Cont *replay preserves the finite route through $X, L, G, T, C_0, C_1, H, C,$ and cannot create trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or* Real$^{\uparrow}$ *completeness. Any consumer claiming one of those endpoints has left the displayed apartness-space carrier.*

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier transport and replay`

Proof. Project the carrier of Definition 1514.1. The transport row H moves only the displayed carried-object, located-apartness, positive-gap, classifier-boundary, continuation, provenance, and local naming rows.

Replay is then bounded by C. It sends finite located observations to the positive-gap row, then to classifier exclusion, and sends zero-distance readings only toward the separated-metric boundary row C_1.

Since neither H nor C contains a decision, equality, selected-limit, compactness, or completeness coordinate, replay cannot supply any of the listed endpoints. Such a claim would require data absent from the accepted packet. □

Theorem 1514.8 (ApartnessSpace scoped separated positive-gap route)**.** *Every accepted* ApartnessSpace$^{\uparrow}$ *consumer that exports separated-metric nonidentification must first display the* LocatedApartness$^{\uparrow}$ *source row, the* DyadicRatCore$^{\uparrow}$ *positive-gap ledger, the classifier-exclusion row, and the* SeparatedMetric$^{\uparrow}$ *boundary row before any separation conclusion is read. The route is the one named by Definition 1514.1, Definition 1514.2, Definition 1514.4, Definition 1514.5, and Definition 1514.7; it does not export trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1514.1. A separated nonidentification read has access only to the displayed carried object, the located-apartness source row L, the positive-gap row G, the separated-metric boundary row C_1, and the transport and replay rows already listed in the carrier.

Use Definition 1514.4 to force any positive separation claim through the $\mathsf{LocatedApartness}^{\uparrow}$ source row and the $\mathsf{DyadicRatCore}^{\uparrow}$ positive-gap ledger. Then apply Definition 1514.5 to keep the same ledger under componentwise hsame transport.

The classifier surface is fixed by Definition 1514.2. It reads exclusion only after the positive-gap ledger has been displayed, and the zero-distance boundary can enter only through the $\mathsf{SeparatedMetric}^{\uparrow}$ row already present in the carrier.

Finally apply Definition 1514.7. Replay cannot create trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects is a carrier row. The separated conclusion is therefore consumed only through the displayed positive-gap and separated-metric route. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier scoped separated positive gap route`

Theorem 1514.9 (ApartnessSpace NameCert obligation package)**.** *The paper-side obligation surface for* $\mathsf{ApartnessSpace}^{\uparrow}$ *is the package consisting of carrier admission, positive-gap exactness, separated-metric boundary compatibility, classifier stability, transport, replay, provenance, and local naming:*

$$X, L, G, T, C_0, C_1, H, C, P, N,\ \mathsf{NameCert}_{\mathsf{ApartnessSpace}^{\uparrow}}.$$

This package records exactly the finite positive-gap classifier route of the chapter and does not export trichotomy, host inequality, quotient real equality, selected limits, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier namecert obligation package`

Proof. Carrier admission is supplied by Definition 1514.1. Positive-gap exactness is the route through Definition 1514.4 and Definition 1514.5.

The separated-metric boundary is supplied by Definition 1514.11, while classifier stability is supplied by Definition 1514.6. These results keep the positive route through C_0 and the zero-distance route through C_1 distinct.

Finally apply the transport-and-replay boundary above. The provenance row P and local naming row N name the same finite packet, so the obligation surface is exhausted by the displayed carrier fields and no excluded endpoint is available to a consumer. □

Theorem 1514.10 (ApartnessSpace zero-distance nonescape)**.** *Any zero-distance consumer of an accepted* $\mathsf{ApartnessSpace}^{\uparrow}$ *packet must pass through the displayed* C_1 *separated-metric boundary of Definition 1514.11. In particular, absence of a positive gap cannot be combined inside the packet with equality, quotient real equality, selected limits, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet through Definition 1514.1. The only zero-distance-facing row is C_1, and it is exposed as a boundary toward the separated-metric consumer, not as a host equality row.

Apply Definition 1514.11. That boundary separates the positive-gap route through C_0 from the zero-distance route through C_1 and keeps both routes inside the finite carrier.

Finally compare with the positive-gap route of Definition 1514.4. A missing positive gap is not a row of equality evidence, and the packet contains no quotient real equality, selected-limit, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. The zero-distance consumer therefore cannot escape the displayed C_1 boundary. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier zero distance nonescape`

Theorem 1514.11 (ApartnessSpace separated-metric zero-distance boundary)**.** *Every* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{ApartnessSpace}^{\uparrow}$ *packet can use the packet only through the displayed boundary*

$$\mathsf{LocatedApartness}^{\uparrow},\ \mathsf{ApartnessSpace}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow}.$$

Concretely, a positive-gap row may yield classifier exclusion through C_0*, while a zero-distance reading may be passed to* $\mathsf{SeparatedMetric}^{\uparrow}$ *through* C_1*. The packet does not allow a consumer to combine the two directions into trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier inventory in Definition 1514.1 gives the full consumer surface. A positive-gap consumer reads L, G, and C_0; a zero-distance consumer reads C_1 and then passes to the separated-metric classifier boundary.

The hsame and Cont rows preserve this order under transport. They move finite observation rows and replay finite routes, but they do not add a decision row comparing every pair of observations.

Thus the separated-metric boundary is one-way data flow from the apartness packet to the metric classifier consumer. No quotient equality, host inequality, selected limit, compactness principle, or real-completeness theorem is a field of A, so no local NameCert obligation can export such data. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier separatedmetric zero distance boundary`

Theorem 1514.12 (ApartnessSpace mature route induction)**.** *Every public consumer of an accepted* $\mathsf{ApartnessSpace}^{\uparrow}$ *packet*

$$A = (X, L, G, T, C_0, C_1, H, C, P, N)$$

is handled by induction over the displayed consumer route. Positive-gap consumers factor through L, G, C_0*; zero-distance consumers factor through* C_1 *and the* $\mathsf{SeparatedMetric}^{\uparrow}$ *boundary; transport and replay consumers factor through* H, C*; provenance and naming consumers factor through* P, N*. These are the only consumer cases supplied by the packet, and none of them exports trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the consumer route exposed by Definition 1514.1. The carrier rows split the public reads into positive-gap, zero-distance, transport/replay, and provenance/name cases.

1. In the positive-gap case, apply Definition 1514.4. The read begins at the $\mathsf{LocatedApartness}^{\uparrow}$ source row L, displays the positive $\mathsf{DyadicRatCore}^{\uparrow}$ gap row G, and reaches classifier exclusion only through C_0.

2. In the transported positive-gap case, apply Definition 1514.5 and Definition 1514.8. These theorems keep the same L, G, C_0 spine under componentwise hsame transport and force separated-metric consumers to display the boundary row before a separation conclusion is read.

3. In the zero-distance case, apply Definition 1514.10 and Definition 1514.11. The consumer can only pass through C_1 toward $\mathsf{SeparatedMetric}^{\uparrow}$; it cannot turn absence of a positive gap into equality or a total comparison principle.

It remains to account for structural consumers. The rows H and C transport and replay only the displayed finite route, while P and N package and name that same route. Since no row of A contains trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the induction exhausts the public consumer surface and refuses those endpoints. □

Lean4 ✓ `BEDC.Derived.ApartnessSpaceUp.ApartnessSpaceCarrier mature route induction`

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ApartnessSpace}^{\uparrow}$ packet is generated from a carried-object row, a $\mathsf{LocatedApartness}^{\uparrow}$ source row, one positive dyadic gap ledger, apartness transport, classifier-exclusion and separated-metric boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1514.1, Definition 1514.2, Definition 1514.4, Definition 1514.8, Definition 1514.10, Definition 1514.11, and Definition 1514.12 close the carrier/classifier scope, located positive-gap route, scoped separated positive-gap route, zero-distance nonescape, separated-metric zero-distance boundary, and mature route induction as the bridge-facing surface for constructive apartness-space use.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, host inequality, selected limits, open-cover compactness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No further theory-axis upgrade remains; Lean verification and bridge checking remain separate.

Chapter 1515

A Concrete Naming Certificate for $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style metric completion, the SequentiallyCompleteMe packet records the L10 sequence-completeness interface as finite Cauchy-sequence data over a displayed metric source. It sits between the metric carrier of chapter 57, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the limit-witness boundary of chapter 128, and the completion route of chapter 1506. The packet gives later real-completion consumers a traditional sequence-completeness bridge without quotienting Cauchy names, selecting representatives, using countable choice, invoking open-cover compactness, assuming ambient completeness, or asserting that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 1515.1 (SequentiallyCompleteMetric carrier)**.** *A* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, S, M, L, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, including the admitted point classifier, distance witness surface, and finite metric observation histories.*
2. *S is the regular Cauchy sequence row, read through* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readbacks over the same metric source.*
3. *M is the displayed modulus row assigning each positive tolerance request a finite late-index threshold.*
4. *L is the displayed limit row, carried as an admitted point history of X before any completion-facing consumer can read it.*
5. *D is the distance-bound ledger whose rows display, for every requested tolerance and every sufficiently late sequence window, the metric-distance witness from the sequence value to L and the rational comparison below that tolerance.*
6. *H records componentwise* hsame *transport for the metric source, Cauchy-sequence rows, modulus, displayed limit, distance-bound ledger, replay row, provenance row, and local naming row.*
7. *C records the displayed* Cont *replay from source metric admission to stream-window reads, regular rational readbacks, modulus thresholds, late-distance bounds, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness handoff.*

8. *P is the* Pkg *provenance over* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricUp`

Theorem 1515.2 (SequentiallyCompleteMetric NameCert obligations)**.** *For an accepted* SequentiallyCompleteMetr *carrier $Q = (X, S, M, L, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed metric source, regular Cauchy sequence row, modulus, displayed limit, late-distance ledger, componentwise transport, continuation replay, provenance, and local naming;*

2. *classifier stability, requiring any accepted replacement of Q to transport X, S, M, L, D componentwise by H before the local classifier identifies the two packets;*

3. *sequence convergence, requiring each tolerance request to read a threshold from M and then a late-distance row from D witnessing convergence of the displayed sequence S to the displayed limit L inside X;*

4. $\mathsf{CompleteMetric}^{\uparrow}$ *handoff, requiring the same rows S, M, L, D to supply the limit-witness surface consumed by chapter 128;*

5. *ledger non-escape, stating that no quotient completion, selected representative, countable-choice schedule, open-cover compactness principle, ambient-completeness witness, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricPacket semantic name certificate`

Proof. Carrier admission is projection from Definition 1515.1. The tuple exposes the metric source, sequence row, modulus, limit row, distance-bound ledger, transport, replay, provenance, and local name as visible rows.

Classifier stability is componentwise. A comparison of two accepted packets first transports the metric source, stream windows, regular rational readbacks, modulus thresholds, displayed limits, and distance-bound rows by H; only after that transport can the local classifier identify the two packets.

Sequence convergence is the content of M and D. For each tolerance request, the modulus row supplies a late threshold, and the distance-bound ledger supplies the metric-distance witness from every displayed late sequence value to L together with the rational comparison below the requested tolerance.

The complete-metric handoff consumes the same convergence rows rather than a second completion object. The replay row C presents S, M, L, D to the $\mathsf{CompleteMetric}^{\uparrow}$ limit-witness boundary, while P and N record that the route is sourced and named by the same finite packet.

The rows of Q are exhausted by this route. Since none of them stores quotient equality, a selected representative, a countable-choice schedule, an open-cover compactness principle, an ambient complete space, or a $\mathsf{Real}^{\uparrow}$ completeness theorem, the ledger non-escape clause follows. □

Theorem 1515.3 (SequentiallyCompleteMetric complete-metric handoff)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. If the source metric X, the stream rows S, the modulus row M, the displayed limit L, and all late-distance bounds D are displayed, then Q supplies the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness surface for that regular Cauchy sequence over X.*

The handoff is the finite route

$$X \longmapsto S \longmapsto M \longmapsto L \longmapsto D \longmapsto H, C, P, N.$$

It exports sequence completeness only as a displayed limit-witness packet for the given regular Cauchy data, not as quotient completion, representative selection, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompleteMetricHandoff`

Proof. Start with the accepted carrier and project the metric source X. The point classifier and metric-distance witnesses used by the handoff are read from this source row, so the limit-witness surface is over the same metric carrier.

Next read the sequence and modulus rows. The row S supplies the finite stream windows and regular rational readbacks of the Cauchy sequence, while M supplies the threshold requested by each tolerance.

Now read the proposed limit and late-distance ledger. The row L is already an admitted point history of X, and D records the metric-distance witness from each sufficiently late displayed sequence value to L together with the required rational comparison.

Finally replay the route through C and attach H, P, N. These rows transport, source, and name exactly the data required by the $\mathsf{CompleteMetric}^{\uparrow}$ limit-witness boundary of Definition 128.2. Because the route has no further coordinate, the handoff cannot export quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.4 (SequentiallyCompleteMetric classifier transport)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ and $Q' = (X', S', M', L', D', H', C', P', N')$ be accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carriers. If the transport rows identify X with X', S with S', M with M', L with L', D with D', C with C', P with P', and N with N' componentwise by* hsame*, then the sequence-completeness read exposed by Q is the same read exposed by Q'. The comparison is made only through the displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, modulus, distance-ledger, replay, provenance, and local naming rows.*

Proof. Project the carrier definition for both packets. The local classifier has no coordinate beyond $X, S, M, L, D, H, C, P, N$, so a comparison can inspect only those rows.

Use the transport row componentwise. The source metric, stream windows, rational readbacks, modulus thresholds, displayed limit, and late-distance ledger are carried by hsame before any convergence read is compared.

The replay and provenance rows preserve the same route to the $\mathsf{CompleteMetric}^{\uparrow}$ handoff. Since no quotient completion, representative-selection, choice, open-cover, ambient-completeness, or $\mathsf{Real}^{\uparrow}$ completeness coordinate is present, the transported classifier read is exactly the same sequence-completeness surface. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricClassifierTransport`

Theorem 1515.5 (SequentiallyCompleteMetric Cauchy-sequence obligation)**.** *Every accepted* $\mathsf{SequentiallyComplete}$ *packet $Q = (X, S, M, L, D, H, C, P, N)$ must display the* $\mathsf{Met}^{\uparrow}$ *source X, the* $\mathsf{StreamName}^{\uparrow}$ *windows*

in S, the RegSeqRat$^{\uparrow}$ *readback of those windows, and the modulus threshold row M before any sequence-completeness read can be consumed. The read then reaches the displayed limit row L and late-distance ledger D only through the same* BHist, hsame, Cont, Pkg, *and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCauchySequenceObligation`

Proof. Project the carrier definition for Q. The sequence row S is meaningful only over the displayed metric source X, and its finite windows are the only sequence observations present in the packet.

Read the rational and threshold rows next. The RegSeqRat$^{\uparrow}$ readback attached to S supplies the rational window data, while M supplies the late threshold requested by the sequence-completeness consumer.

Only after those rows are displayed can the consumer read L and D. The rows H, C, P, N transport, replay, source, and name this same route, so no sequence-completeness read bypasses the metric source, stream windows, rational readback, or modulus threshold. □

Theorem 1515.6 (SequentiallyCompleteMetric limit-stability obligation)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ and $Q' = (X', S', M', L', D', H', C', P', N')$ be accepted* SequentiallyCompleteMetric$^{\uparrow}$ *packets. If the displayed transport identifies X, S, M, L, D, C, P, N with $X', S', M', L', D', C', P', N'$ componentwise by* hsame*, then the limit read exposed by Q is the same limit read exposed by Q'. In particular,* hsame *transport preserves the displayed limit row and the late-distance ledger before any* CompleteMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer can compare the packets.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitStabilityObligation`

Proof. Start with the classifier-transport theorem. It confines comparison of accepted packets to the displayed metric source, stream-window, regular-readback, modulus, distance-ledger, replay, provenance, and local naming rows.

Use the componentwise transport hypotheses. The displayed limit rows L and L' are compared only after the common metric source, sequence windows, and modulus thresholds have been transported, and the late-distance ledgers D and D' carry the corresponding distance comparisons.

Replay through C, P, N and C', P', N'. Since neither carrier contains a quotient-completion equality, selected representative, ambient completeness witness, or independent completed carrier, the transported limit read is exactly the same finite limit-stability surface. □

Theorem 1515.7 (SequentiallyCompleteMetric separated uniqueness boundary)**.** *For an accepted* SequentiallyCompleteMetric$^{\uparrow}$ *carrier $Q = (X, S, M, L, D, H, C, P, N)$, separated uniqueness is only a boundary condition on displayed limit reads. Any uniqueness consumer must first read the* Met$^{\uparrow}$ *classifier in X, the common* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readback of S, the modulus row M, the displayed limit row L, and the late-distance ledger D. The packet exports neither quotient completion nor completeness of* Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSeparatedUniquenessBoundary`

Proof. Project the rows of Q. The source metric classifier in X is the only equality-facing surface available to a separated uniqueness consumer, and the sequence data are displayed only through S and its regular rational readback.

Read the convergence rows. The modulus row M supplies the late threshold, while L and D provide the displayed limit and the finite distance comparisons used at the separation boundary.

The structural rows H, C, P, N preserve this finite route without adding a completed quotient carrier or a real-completeness theorem. Therefore separated uniqueness can be consumed only as a boundary condition on the displayed rows, not as quotient completion or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.8 (SequentiallyCompleteMetric late-bound coverage)**.** *Every tolerance request made to an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$ *factors first through the modulus row* M *and then through the distance-bound ledger* D *before any* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff is available. Thus a completion-facing consumer receives late convergence data only as a displayed threshold and a displayed metric-distance comparison inside* X*.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLateBoundCoverage`

Proof. Begin with a tolerance request. The carrier definition stores tolerance handling in M, so the request first reads a finite late-index threshold rather than a chosen subsequence or a countable schedule.

Next read the distance-bound ledger. The row D records the metric-distance witness from each sufficiently late displayed sequence value in S to the displayed limit L, together with the rational comparison below the requested tolerance.

Only after those rows are exposed can the route replay through C toward Definition 1515.3. The handoff therefore consumes M and D as late-bound evidence and cannot bypass them through quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.9 (SequentiallyCompleteMetric finite-modulus induction coverage)**.** *Let* $Q = (X, S, M, L, D, H, C, P, N)$ *be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier, and let* ρ *be a finite late-distance request history addressed to the completion-facing read of* Q*. Every accepted late-distance read in* ρ *factors by induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows in* S *and the modulus row* M*: the base read inspects the first admitted window, the successor read advances only through the displayed threshold selected by* M *and the corresponding distance-bound entry in* D*, and the resulting read is replayed through* H, C, P, N *before any* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer may use it. The route introduces no quotient completion, representative selection, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or independent completed carrier.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricFiniteModulusInductionCover`

Proof. Project the carrier rows. The Cauchy-sequence obligation of Definition 1515.5 exposes only the source metric X, the displayed stream windows S, the regular rational readback, the modulus row M, the displayed limit L, and the late-distance ledger D before a completion-facing read is available.

Induct on the finite displayed request history ρ. In the base case, the first accepted read has only the first admitted $\mathsf{StreamName}^{\uparrow}$ window and the carrier rows projected above. Thus it is already routed through S and M before any distance comparison can be consumed.

For the successor step, apply the induction hypothesis to the earlier displayed windows. The next request can advance only by reading the threshold supplied by M and then the matching distance-bound entry in D; this is exactly the late-bound coverage route of Definition 1515.8. Replay through H, C, P, N preserves the same finite route for the $\mathsf{CompleteMetric}^{\uparrow}$ handoff.

Finally apply the nonescape boundary of Definition 1515.10. Since the carrier contains no coordinate for quotient completion, representative selection, countable choice, open-cover compactness, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, or an independent completed carrier, the induction covers only the displayed finite modulus route. □

Theorem 1515.10 (SequentiallyCompleteMetric ledger nonescape)**.** *For every accepted* SequentiallyCompleteMetri *carrier* $Q = (X, S, M, L, D, H, C, P, N)$*, the local ledger cannot export quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or any assertion that* $\mathsf{Real}^{\uparrow}$ *is complete. Any consumer asking for those claims is returned to the displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, modulus, late-bound, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLedgerNonescape`

Proof. Inspect the tuple Q. Its rows name a metric source, regular Cauchy sequence data, a modulus, a displayed limit, a late-distance ledger, transport, replay, provenance, and a local NameCert row.

Apply the NameCert obligation theorem. Its ledger non-escape clause states that the local surface contains no row for quotient equality, representative selection, countable-choice schedules, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

The complete-metric handoff adds no extra coordinate. It only exposes the same displayed convergence route to $\mathsf{CompleteMetric}^{\uparrow}$, so every forbidden request remains outside the carrier and outside the consumer-facing handoff. □

Theorem 1515.11 (SequentiallyCompleteMetric root limit witness)**.** *Let* $Q = (X, S, M, L, D, H, C, P, N)$ *be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. The root limit witness exported by* Q *is exactly the displayed limit row* L *together with the late-distance ledger* D *over the* $\mathsf{Met}^{\uparrow}$ *source* X*, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of* S*, the modulus row* M*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The witness is finite carrier data and does not assert that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootLimitWitness`

Proof. Project the carrier Q. The only limit coordinate is L, and it is admitted as a point history of the source metric X before any completion-facing handoff can inspect it.

The ledger D attaches L to the stream and rational readback rows. For each displayed tolerance request, M gives the late threshold, while S and its $\mathsf{RegSeqRat}^{\uparrow}$ reads provide the finite sequence windows whose distances to L are recorded in D.

The rows H, C, P, N transport, replay, source, and name the same finite limit witness. Since no row of Q stores ambient real completeness or a completed real carrier, the root witness remains a displayed sequence-completeness packet over X. □

Theorem 1515.12 (SequentiallyCompleteMetric root modulus coverage)**.** *For every accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$*, each root tolerance request is covered by the modulus row* M *before the late-distance ledger* D *can be consumed. The coverage factors through the* $\mathsf{Met}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff,* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer boundary,* BHist *rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootModulusCoverage`

Proof. Start with a tolerance request at the root sequence-completeness interface. The carrier places all threshold data in M, so the request first reads the displayed modulus rather than a host choice function or an infinite schedule.

After the threshold is displayed, S supplies the finite stream windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback, and D supplies the metric-distance comparison from the corresponding late values to L inside X. Thus the late-bound coverage is a finite BHist route.

The replay row C exposes this same coverage to the $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ boundaries, while H, P, N preserve the route under transport, provenance, and naming. No step requires or exports $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.13 (SequentiallyCompleteMetric source-first tail read)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Every accepted tail-read packet that is later consumed as a displayed limit read must expose the* $\mathsf{Met}^{\uparrow}$ *source row X, the* $\mathsf{StreamName}^{\uparrow}$ *windows in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to those windows, and the modulus row M before any displayed limit row L or late-distance ledger D is read. Thus a completion-facing tail read is source-first: it cannot begin from a free-standing limit, a quotient-completion class, or a selected representative.*

Proof. Project the carrier of Definition 1515.1. The only metric source is X, the only sequence observations are the windows in S, and the regular rational readback attached to those observations is part of the same finite sequence row.

Apply root modulus coverage from Definition 1515.12. A tolerance request cannot inspect late-distance data until the modulus row M has supplied its displayed threshold, so the tail read already contains X, S and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before D is available.

Now use root nonescape from Definition 1515.21 and the complete-metric handoff of Definition 1515.3. The handoff sends the same source-first rows to the $\mathsf{CompleteMetric}^{\uparrow}$ boundary, and the nonescape clause blocks any route that would start from a quotient class, representative selector, or independent limit row. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSourceFirstTailRead`

Theorem 1515.14 (SequentiallyCompleteMetric tail induction)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Any finite induction over completion-facing tails of the displayed sequence must keep the* $\mathsf{Met}^{\uparrow}$ *source X, the* $\mathsf{StreamName}^{\uparrow}$ *tail windows in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of those windows, the modulus row M, the displayed limit row L, and the late-distance ledger D on the same route. The base tail reads X, S, M before L, D, and each successor tail may extend only the displayed window and late-distance rows before replay through H, C, P, N.*

Proof. Use the source-first tail read. The base tail cannot enter through L or D alone, so it first exposes the metric source, StreamName windows, regular rational readback, and modulus row.

For the successor tail, append one displayed window request to S. The regular rational readback remains the handoff from the window to the tolerance ledger, and the modulus row still supplies the threshold before the new late-distance entry in D is visible.

Project the carrier rows H, C, P, N at the end of the induction step. They transport, replay, source, and name the same finite route, so tail induction cannot introduce quotient completion, selected representatives, countable choice, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricTailInduction`

Theorem 1515.15 (SequentiallyCompleteMetric limit ledger coverage)**.** *For an accepted* SequentiallyCompleteMetr *carrier $Q = (X, S, M, L, D, H, C, P, N)$, every tolerance request that reaches the displayed limit ledger must first read the modulus row M and the late-distance ledger D over the* $\mathsf{Met}^{\uparrow}$ *source X and the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *sequence rows. Only after this coverage is displayed may the request be handed to* $\mathsf{CompleteMetric}^{\uparrow}$*; no tolerance request reaches that handoff through an unrecorded limit, quotient-completion class, selected representative, or host Cauchy schedule.*

Proof. Start from root modulus coverage. It requires each tolerance request to obtain its threshold from M before any late-distance comparison is consumed.

Apply the source-first tail read and tail induction. Together they keep the sequence windows and regular rational readback upstream of the late-distance ledger for the base tail and for every successor tail.

Finally use the complete-metric handoff. The $\mathsf{CompleteMetric}^{\uparrow}$ boundary receives only the already covered route $X, S, M, L, D, H, C, P, N$, so the handoff has no separate entry for unrecorded limits, quotient classes, selected representatives, host schedules, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitLedgerCoverage`

Theorem 1515.16 (SequentiallyCompleteMetric completion-consumer route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier receives only the displayed sequence-completeness packet*

$$X, S, M, L, D, H, C, P, N.$$

The consumer must read the source-first tail route, the tail-induction route, and the limit-ledger coverage route before it can use the $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It receives no second completion carrier, quotient equality, selected representative, countable-choice schedule, ambient complete space, or proof that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionConsumerRoute`

Proof. Begin with the source-first tail route. It confines every completion-facing read to the metric source, sequence windows, regular rational readback, and modulus row before limit data are inspected.

Next use tail induction and limit-ledger coverage. These results ensure that finite tail refinement and every tolerance request keep the late-distance ledger attached to the same displayed carrier rows.

Apply the complete-metric handoff and root nonescape. The handoff exposes the packet to $\mathsf{CompleteMetric}^{\uparrow}$, while nonescape refuses all coordinates for an independent completion carrier, quotient equality, representative selection, countable choice, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. The $\mathsf{MetricCompletion}^{\uparrow}$ consumer therefore receives exactly the displayed packet. □

Theorem 1515.17 (SequentiallyCompleteMetric displayed-limit uniqueness)**.** *Let* $Q = (X, S, M, L, D, H, C, P, N)$ *be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. If two displayed limit reads for the same* BHist *source pass through the same* $\mathsf{Met}^{\uparrow}$ *row* X*,* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *sequence row* S*, modulus row* M*, late-distance ledger* D*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *row* P*, and local* $\mathsf{NameCert}$ *row* N*, then the uniqueness route is confined to the displayed limit row* L *and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It cannot compare limits through quotient completion, selected representatives, countable choice, or a hidden ambient complete metric.*

Proof. Project the carrier and keep both limit reads inside the same packet Q. The source metric, stream windows, regular rational readback, modulus, late-distance ledger, transport, replay, package provenance, and local naming rows are shared before any uniqueness comparison is made.

Apply the completion-consumer route. It forces every completion-facing read to pass through the displayed $\mathsf{Met}^{\uparrow}/\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{CompleteMetric}^{\uparrow}$ route and refuses an independent completion carrier.

The comparison therefore has only the displayed limit row L and the late-distance ledger D available. The structural rows H, C, P, N preserve that route, so uniqueness is a displayed-limit claim rather than an equality imported from an ambient completion. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricDisplayedLimitUniqueness`

Theorem 1515.18 (SequentiallyCompleteMetric Cauchy tail stability)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. Any cofinal tail replacement of the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *sequence row S that preserves the displayed* BHist *source X, the modulus row M, the late-distance ledger D, componentwise* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N remains on the same* $\mathsf{Met}^{\uparrow}$ *to* $\mathsf{CompleteMetric}^{\uparrow}$ *route. The replacement may refine tail windows, but it may not introduce a second limit carrier, quotient-completion equality, selected representative, or* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCauchyTailStability`

Proof. Read the tail replacement as a finite refinement of S. The regular rational readback remains attached to the same $\mathsf{Met}^{\uparrow}$ source X, and the modulus row M still supplies the displayed thresholds for the late-distance ledger D.

Use the tail-induction and completion-consumer routes. Tail induction keeps the refined windows in the same carrier, while the completion-consumer theorem admits downstream reads only after the displayed route through $X, S, M, L, D, H, C, P, N$ is visible.

Transport and replay through H, C, P, N preserve the packet after the tail refinement. Hence the cofinal tail stays on the $\mathsf{Met}^{\uparrow}/\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{CompleteMetric}^{\uparrow}$ path and cannot export an external completion object or ambient real completeness. □

Theorem 1515.19 (SequentiallyCompleteMetric completion-boundary refusal)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. A completion-boundary consumer that does not display the* BHist *source X, the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *sequence row S, the modulus row M, the displayed limit row L, the late-distance ledger D, componentwise* hsame *row H,* Cont *replay row C,* Pkg *row P, and local* NameCert *row N is refused by the* $\mathsf{Met}^{\uparrow}$ *to* $\mathsf{CompleteMetric}^{\uparrow}$ *route. The boundary exports no Real completeness theorem, countable-choice schedule, quotient completion, selected representative, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionBoundaryRefusal`

Proof. Start from root nonescape. A root-facing consumer is confined to the displayed metric, sequence, completion, history, transport, replay, package, and naming rows of Q.

Apply the completion-consumer route. The $\mathsf{CompleteMetric}^{\uparrow}$ handoff is available only after the source-first tail route, tail induction route, and limit-ledger coverage route have all been displayed.

If any row of $X, S, M, L, D, H, C, P, N$ is absent, the consumer no longer has the packet required by the handoff. The boundary therefore refuses the request rather than supplying Real completeness, countable choice, quotient completion, selected representatives, or open-cover compactness. □

Theorem 1515.20 (SequentiallyCompleteMetric limit ledger nonescape)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. The limit ledger formed by L, D, H, C, P, N cannot export quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or any assertion that* $\mathsf{Real}^{\uparrow}$ *is complete. Every such request is returned to the carrier rows of Definition 1515.1, the source-first read of Definition 1515.13, the root nonescape boundary of Definition 1515.21, and the complete-metric handoff of Definition 1515.3.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitLedgerNonescape`

Proof. Read the ledger coordinates. The displayed limit row L is an admitted point history of the metric source, while D records late-distance comparisons from the sequence windows to that point; the remaining rows H, C, P, N only transport, replay, source, and name the same packet.

Use the source-first tail-read theorem. It forces every completion-facing read of L and D to pass through X, S with its $\mathsf{RegSeqRat}^{\uparrow}$ readback and through the modulus row M before the limit ledger is consumed.

Apply root nonescape and the complete-metric handoff. Those two results identify the downstream completion read with the same finite rows and refuse all coordinates for quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.21 (SequentiallyCompleteMetric root nonescape)**.** *Every root-facing consumer of an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$ *is confined to the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of* Q*. In particular, the root route cannot escape to quotient completion equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootNonescape`

Proof. Inspect the root route exposed by the carrier. A consumer can read the metric source X, the stream row S, the modulus M, the displayed limit L, the late-distance ledger D, and the structural rows H, C, P, N.

Apply the complete-metric handoff and late-bound coverage theorems. They show that completion-facing reads pass through the same finite modulus and distance-ledger rows before reaching the $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ boundary.

There is no coordinate in Q for quotient completion equality, selected representatives, choice schedules, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. Hence every root-facing request for those claims remains outside the local NameCert surface. □

Theorem 1515.22 (SequentiallyCompleteMetric Real refusal boundary)**.** *An accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$ *supplies a displayed limit witness for a regular Cauchy sequence over its metric source* X*, but it cannot be consumed as completeness of* $\mathsf{Real}^{\uparrow}$ *or as ambient metric completeness unless an additional Real-specific completion carrier is displayed. The available route is the finite sequence-completeness surface through* $X, S, M, L, D, H, C, P, N$ *and not a global completeness theorem.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRealRefusalBoundary`

Proof. Start with the local NameCert obligations of Definition 1515.2. They admit the metric source, regular Cauchy sequence row, modulus, displayed limit, late-distance ledger, transport, replay, provenance, and local naming rows, and their ledger non-escape clause refuses ambient completeness and $\mathsf{Real}^{\uparrow}$ completeness.

Apply the complete-metric handoff of Definition 1515.3. It sends the same rows X, S, M, L, D to the $\mathsf{CompleteMetric}^{\uparrow}$ limit-witness surface for that sequence, without creating an independent complete carrier or Real-specific completion row.

Apply the root limit witness and root nonescape theorems. The witness is exactly the displayed limit row L with late-distance data over X, while every root-facing consumer remains confined to

$\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Consequently the carrier supplies sequence completeness only for the displayed Cauchy data. A consumer that asks for completeness of $\mathsf{Real}^{\uparrow}$ or ambient metric completeness must display a separate Real-specific completion carrier; it cannot obtain that claim from Q. □

Theorem 1515.23 (SequentiallyCompleteMetric root Cauchy limit existence)**.** *Let $Q = (X, S, M, L, D, H, C, P, N$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Any root consumer asserting Cauchy limit existence for the displayed sequence must obtain the claim by reading the* $\mathsf{Met}^{\uparrow}$ *source X, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in S, the modulus row M, the displayed limit row L, and the late-distance ledger D through the same* BHist, hsame, Cont, Pkg, *and* NameCert *route. The existence read may then be handed to* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumers, but it refuses quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, and completeness of* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootCauchyLimitExistence`

Proof. Project the tuple Q. The only available source for a limit-existence read is the displayed metric row X together with the sequence, modulus, limit, and late-distance rows S, M, L, D.

Read the Cauchy data before the limit. The stream windows and regular rational readback in S expose the observed sequence, while M supplies the threshold demanded by a tolerance request.

Now read L and D. The displayed limit row gives the candidate point inside X, and the late-distance ledger records the finite convergence comparison needed by the root consumer.

Finally replay the same rows through H, C, P, N. This makes the handoff visible to $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ without adding any quotient class, chosen representative, choice schedule, open-cover compactness object, ambient complete carrier, or $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Theorem 1515.24 (SequentiallyCompleteMetric root modulus-limit factorization)**.** *For an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier $Q = (X, S, M, L, D, H, C, P, N)$, every root read that combines a modulus with a displayed limit factors through*

$$X, S, M \longmapsto L, D \longmapsto H, C, P, N.$$

The factorization first fixes the $\mathsf{Met}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *sequence row, and the modulus threshold row; only then may a consumer compare the displayed limit through the late-distance ledger and replay the result through* BHist, hsame, Cont, Pkg, *and* NameCert. *The factorization exports no quotient completion, selected representative, countable-choice schedule, open-cover compactness principle, ambient completeness row, or* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with a modulus-limit request. The carrier places metric admission in X, sequence observations in S, and threshold data in M, so these rows form the first stage of the read.

The displayed limit cannot be inspected earlier. It is meaningful only after the metric source and Cauchy tail threshold have been fixed, because the ledger D compares late sequence values with L inside X.

The structural rows then preserve the factorization. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming all act on the same rows rather than constructing a second completion object.

Since the route has no coordinate for quotient equality, representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, each such request remains outside the root factorization. □

Lean4 ✓ BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootModulusLimitFactorizati

Theorem 1515.25 (SequentiallyCompleteMetric root completion consumer interface)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet* $Q = (X, S, M, L, D, H, C, P, N)$ *is confined to the displayed interface*

$$\mathsf{Met}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

as instantiated by the rows of Q*. The consumer may read sequence-completeness evidence only through* $X, S, M, L, D, H, C, P, N$*, and must refuse quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, and completeness of* $\mathsf{Real}^{\uparrow}$ *unless a separate carrier displays those data.*

Proof. Start with a completion-facing consumer. The carrier inventory exposes exactly the metric source, sequence row, modulus, displayed limit, late-distance ledger, transport, replay, provenance, and local name.

Apply the root Cauchy limit existence theorem. It forces the consumer to read X, S, M, L, D before any limit-existence handoff is visible.

Apply the root modulus-limit factorization. It keeps the threshold and limit comparison in the displayed order, so the completion consumer receives a finite sequence-completeness surface rather than a detached completed object.

The remaining rows H, C, P, N merely transport, replay, source, and name that interface. Therefore every request for quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness is refused unless supplied by another displayed carrier. □

Lean4 ✓ BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRootCompletionConsumerInter

Theorem 1515.26 (SequentiallyCompleteMetric regular Cauchy limit coverage)**.** *Let* $Q = (X, S, M, L, D, H, C, P,$ *be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Every regular Cauchy limit read exposed at the root must display the* $\mathsf{Met}^{\uparrow}$ *source row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* S*, the modulus row* M*, the displayed limit row* L*, the late-distance ledger* D*, the ambient* BHist *packet, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N *before any* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer may use the read. The coverage refuses quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, and completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRegularCauchyLimitCoverage

Proof. Project the carrier. The source metric and sequence data are exactly X and S, with the regular rational readback attached to the displayed $\mathsf{StreamName}^{\uparrow}$ windows.

Read the convergence rows next. The modulus M supplies the late threshold, while L and D provide the displayed limit and the metric-distance comparisons used by the root consumer.

Finally replay through H, C, P, N. These rows keep the same finite BHist route available to $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ consumers, and no carrier coordinate supplies quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1515.27 (SequentiallyCompleteMetric limit ledger stability)**.** *For accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packets* $Q = (X, S, M, L, D, H, C, P, N)$ *and* $Q' = (X', S', M', L', D', H', C', P', N')$*, any comparison of their limit ledgers must transport the* $\mathsf{Met}^{\uparrow}$ *source rows,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$

sequence rows, modulus rows, displayed limits, late-distance ledgers, BHist *histories,* Cont *replay rows,* Pkg *provenance rows, and local* NameCert *rows componentwise by* hsame*. The resulting* CompleteMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *read is stable only through those displayed rows and refuses quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, and completeness of* Real$^{\uparrow}$*.*

Proof. Start with the two carrier inventories. Each packet exposes only its metric source, sequence row, modulus, displayed limit, late-distance ledger, transport, replay, provenance, and local name.

Use componentwise hsame transport before comparing limit ledgers. The metric source, finite stream windows, regular rational readback, thresholds, displayed limits, and late-distance rows must move together.

The replay and provenance rows then preserve the transported route for CompleteMetric$^{\uparrow}$ and MetricCompletion$^{\uparrow}$ consumers. Since neither packet contains quotient-completion equality, selected representatives, countable-choice schedules, open-cover compactness, ambient completeness, or Real$^{\uparrow}$ completeness, stability is confined to the displayed ledger rows. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitLedgerStability`

Theorem 1515.28 (SequentiallyCompleteMetric completion consumer nonescape)**.** *Every* CompleteMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer of an accepted* SequentiallyCompleteMetric$^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$ *must read the displayed* Met$^{\uparrow}$ *source* X*,* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readback* S*, modulus row* M*, limit row* L*, late-distance ledger* D*, finite* BHist *history,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. No completion consumer can escape from that route to quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of* Real$^{\uparrow}$*.*

Proof. Inspect the completion-facing entry points of the carrier. The only rows available to a CompleteMetric$^{\uparrow}$ or MetricCompletion$^{\uparrow}$ consumer are $X, S, M, L, D, H, C, P, N$.

Apply regular Cauchy limit coverage. It forces the consumer to read the metric source, finite sequence windows, regular rational readback, modulus, limit, and late-distance data before using a limit claim.

Apply limit ledger stability for transported reads. The same rows remain the only stable comparison surface, so a consumer has no route to quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or Real$^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionConsumerNonescape`

Theorem 1515.29 (SequentiallyCompleteMetric Cauchy-filter bridge)**.** *Let* $Q = (X, S, M, L, D, H, C, P, N)$ *be an accepted* SequentiallyCompleteMetric$^{\uparrow}$ *carrier. Every Cauchy-filter consumer of* Q *must read the metric source row* X*, the finite sequence-window row* S*, the modulus row* M*, the displayed limit row* L*, and the late-distance ledger* D *before it can export a filter-facing convergence observation. The bridge is therefore the finite read*

$$X, S, M, L, D \longmapsto H, C, P, N$$

and not a host filter object, quotient completion equality, selected representative, or countable-choice schedule.

Proof. Project the carrier rows of Q. A Cauchy-filter consumer has no entry point except the displayed metric source, finite sequence windows, modulus, limit row, and late-distance ledger.

The late-bound route supplies the filter basis. The row M gives the threshold for each tolerance, while S and D display the late sequence windows and the metric comparisons to L inside X.

Replay the same rows through H, C, P, N. Since the carrier has no coordinate for a host filter, quotient completion equality, selected representative, or countable-choice schedule, the Cauchy-filter bridge consumes only the finite late-bound and limit rows already exposed by Q. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCauchyFilterBridge`

Theorem 1515.30 (SequentiallyCompleteMetric completion route)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer that reads sequence-completeness data from Q factors through the same displayed rows $X, S, M, L, D, H, C, P, N$ named by Definition 1515.21. In particular, the completion route consumes finite late thresholds and displayed limit evidence; it does not manufacture an independent completed carrier, quotient completion class, representative selector, countable-choice sequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from a completion-facing request. The root non-escape theorem confines such a request to the metric source, stream-window, regular-readback, modulus, displayed-limit, late-ledger, replay, provenance, and local naming rows of Q.

Read the analytic rows in order. The metric source X fixes the distance surface, S supplies the finite sequence windows, M supplies the late threshold, and L, D supply the displayed limit and its distance ledger.

The structural rows H, C, P, N then present this finite surface to $\mathsf{MetricCompletion}^{\uparrow}$. Because the same carrier exposes no independent completed carrier, quotient class, selected representative, countable-choice sequence, or $\mathsf{Real}^{\uparrow}$ completeness theorem, the completion route is exactly the finite route above. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionRoute`

Theorem 1515.31 (SequentiallyCompleteMetric separated-limit uniqueness boundary)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Any separated-limit consumer comparing two displayed limits for the sequence row S must read the source classifier of X, the common finite windows S, the modulus row M, the displayed limit row L, and the late-distance ledger D before using uniqueness. Thus separated-limit uniqueness is bounded by the finite rows already exposed by Q and cannot be sourced from host point equality, quotient completion equality, selected representatives, or ambient completeness.*

Proof. Project the consumer-facing comparison through the carrier. The only source classifier available is the one displayed by X, and the only convergence evidence available is carried by S, M, L, D.

The common finite windows and modulus force both sides of the comparison to use the same late-bound route. The ledger D records the distance comparisons to the displayed limit row L inside the source metric, so separated-limit use is already bounded by the packet.

Transport and replay through H, C, P, N preserve that boundary. Since Q contains no host point equality, quotient completion equality, selected representative, or ambient completeness coordinate, no separated-limit consumer can bypass the finite rows named above. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitUniquenessBoundary`

Theorem 1515.32 (SequentiallyCompleteMetric limit-modulus induction)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier, and let ρ be any finite tolerance-request history addressed to the sequence-completeness surface of Q. Every initial segment of ρ factors through the displayed route*

$$M \longmapsto S \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto D \longmapsto L$$

inside the same BHist *packet before the* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *boundary can read a limit claim. Thus each finite request path reaches the displayed limit row L only through the modulus row M, the stream-window row S, the regular rational readback of S, the late-distance ledger D, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitModulusInduction`

Proof. Project the carrier rows of Q. The tolerance interface has only the displayed modulus row M, the stream row S, the readback route through $\mathsf{RegSeqRat}^{\uparrow}$, the displayed limit L, and the late-distance ledger D available for convergence data.

Induct on the finite request history ρ. For the empty history, there is no consumer-facing limit read, so the factorization is the trivial carrier projection through M, S, L, D, H, C, P, N.

For the successor history, apply the induction hypothesis to the preceding requests. The next tolerance is replayed through Cont by first reading a threshold from M, then reading the corresponding stream windows from S and their regular rational data, and then reading the late-distance row in D which compares the displayed sequence values to L inside X.

The rows H, P, N transport, source, and name exactly this replay. Since the carrier exposes no quotient equality, selected representative, choice schedule, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, every finite request path remains inside BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ data. □

Theorem 1515.33 (SequentiallyCompleteMetric completion-consumer factorization)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet $Q = (X, S, M, L, D, H, C, P, N)$ factors through the same visible surface S, M, L, D, H, C, P, N together with the source metric row X. In particular, the consumer reads sequence completeness only as finite stream windows, regular rational readback, a displayed modulus, a displayed limit, late-distance bounds, componentwise transport, continuation replay, provenance, and local naming; it cannot read quotient-completion equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or an independent complete carrier.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionConsumerFactoriza`

Proof. Start from a completion-facing request. The carrier definition gives the consumer no entry point except the named rows $X, S, M, L, D, H, C, P, N$.

The $\mathsf{CompleteMetric}^{\uparrow}$ handoff reads S, M, L, D as its limit-witness surface over X. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same route through C, with H preserving componentwise transport and P, N preserving provenance and local naming.

No second surface is present. Since quotient completion, representative selection, countable-choice schedules, open-cover compactness, ambient completeness, and independent completed carriers are absent from Q, every completion consumer factors through the displayed finite route. □

Theorem 1515.34 (SequentiallyCompleteMetric choice-free uniqueness boundary)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ and $Q' = (X, S, M, L', D', H', C', P', N')$ be accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$

packets with the same displayed source metric row X, stream row S, and modulus row M. If D and D' satisfy the same late-distance requests under the same classifier rows, then any consumer-facing comparison between the displayed limits L and L' is identified only through the source $\mathsf{Met}^{\uparrow}$ *classifier and the transported sequence-completeness rows. It is not identified by host equality, selected representatives, quotient completion, or choice of a preferred limit name.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricChoiceFreeUniquenessBoundar`

Proof. Project both packets. Their shared source, stream, and modulus rows force every convergence comparison to begin with the same $\mathsf{Met}^{\uparrow}$ classifier, the same $\mathsf{StreamName}^{\uparrow}$ windows, and the same $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the hypothesis on the late-distance ledgers. For each displayed tolerance request, D and D' provide the same kind of late comparison from the same sequence row to the displayed limit rows L and L' inside the source metric classifier.

The comparison is then transported only by the visible hsame rows and replayed only by Cont. Because neither packet contains host equality of points, selected representatives, quotient-completion equality, or a choice operator choosing one limit name, the consumer-facing uniqueness boundary is exactly the source metric classifier together with the displayed finite convergence rows. □

Theorem 1515.35 (SequentiallyCompleteMetric separation obligations)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier, and let ρ be a finite* BHist *request history addressed to a separation comparison at the completion boundary. Before the comparison can be consumed, the request must display the source* $\mathsf{Met}^{\uparrow}$ *classifier in X, the* $\mathsf{StreamName}^{\uparrow}$ *windows in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of those windows, the late-distance ledger D, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff, the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. No separation read may be sourced from quotient-completion equality, selected representatives, host equality of points, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSeparationObligations`

Proof. Start with the complete-metric handoff and late-bound coverage routes already displayed for Q. They force any completion-facing comparison to read the source metric classifier, stream windows, regular readback, displayed modulus, and late-distance bounds before a limit comparison is available.

Next apply the ledger nonescape theorem. It excludes quotient-completion equality, selected representatives, host equality, and ambient completeness from the carrier coordinates, so a separation request has no alternate source.

Finally replay the comparison through H, C, P, N. These rows transport, continue, source, and name the same finite BHist history, so the separation obligation surface is exactly the visible packet rows. □

Theorem 1515.36 (SequentiallyCompleteMetric tail stability)**.** *For an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier $Q = (X, S, M, L, D, H, C, P, N)$, every finite tail replacement of a request history that preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and late-distance entries in D preserves the completion-facing limit route through* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$*. The preserved route remains transported by* hsame*, replayed by* Cont*, sourced by* Pkg*, and named by* NameCert *inside the same carrier.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricTailStability`

Proof. Use limit-modulus induction on the finite request history. The base case is the empty tail, where the visible carrier rows are unchanged.

For a successor tail, the preserved $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback give the same displayed tolerance data. The late-distance ledger D then supplies the same comparison from the sequence row to L inside the source metric classifier.

The complete-metric handoff and completion-consumer factorization identify the downstream read with the same displayed route. The rows H, C, P, N preserve the transported, replayed, sourced, and named packet, so tail replacement does not create a new completion surface. □

Theorem 1515.37 (SequentiallyCompleteMetric completion-route exhaustion)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *request served by an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet* $Q = (X, S, M, L, D, H, C, P,$ *is exhausted by the rows* $X, S, M, L, D, H, C, P, N$ *together with finite* BHist *histories,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. In particular, after the carrier displays the complete-metric handoff, late-bound coverage, ledger nonescape, limit-modulus induction, and completion-consumer factorization routes, no remaining completion request can demand quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or an independent completed carrier.*

Proof. Collect the existing local routes. Complete-metric handoff supplies the limit-witness surface, late-bound coverage forces tolerance reads through M, S, D, L, and ledger nonescape removes the excluded host objects from the carrier.

Use limit-modulus induction to cover arbitrary finite BHist request histories. Each initial segment reaches the displayed limit only through the same stream-window, regular-readback, and late-distance rows.

Finally apply completion-consumer factorization. A $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives no entry point beyond $X, S, M, L, D, H, C, P, N$, so the displayed finite routes exhaust the completion-facing obligation surface. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionRouteExhaustion`

Definition 1515.38 (SequentiallyCompleteMetric finite Cauchy window)**.** *For an accepted* SequentiallyCompleteMe *carrier* $Q = (X, S, M, L, D, H, C, P, N)$*, a finite Cauchy-window obligation package is the displayed tuple*

$$\Omega_Q = (X, S, R, M, L, D, H, C, P, N)$$

where X *is the metric source row of Definition 1515.2,* S *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to those windows,* M *is the modulus threshold row,* L *is the displayed limit row,* D *is the late-distance ledger, and* H, C, P, N *are the transport, replay, provenance, and local naming rows. The package contains no quotient-completion equality, selected representative, countable-choice schedule, ambient-completeness witness, independent completed carrier, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 (definition) `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricFiniteCauchyWindow`

Theorem 1515.39 (SequentiallyCompleteMetric Cauchy-window obligation)**.** *For an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$*, every finite Cauchy-window obligation package* $\Omega_Q = (X, S, R, M, L, D, H, C, P, N)$ *must display the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows reached from* D*, the ambient* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$ *row before any completion-facing consumer can read the window.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCauchyWindowObligation`

Proof. Project the package Ω_Q from Definition 1515.38. Its displayed rows already contain $S, R, M, L, D, H, C, P, N$ over the same metric source X.

Read the analytic front of the package first. The finite $\mathsf{StreamName}^{\uparrow}$ windows S and their $\mathsf{RegSeqRat}^{\uparrow}$ readback R expose the inspected sequence data, while M, L, D bind the requested tolerance, displayed point row, and late-distance comparisons.

The support rows then close the obligation surface. The BHist carrier records the finite packet, H transports it by hsame, C replays it by Cont, P sources it through Pkg, and N names it by the local $\mathsf{NameCert}$ row. No further completion object is read by this package. □

Theorem 1515.40 (SequentiallyCompleteMetric RegSeqRat handoff)**.** *In the notation of Definition 1515.38, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *is the unique handoff surface from the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S *to the late-distance ledger* D *and the* $\mathsf{Real}^{\uparrow}$ *seal-facing rows. The handoff is recorded inside the same* BHist *packet and is transported, replayed, sourced, and named only by* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRegSeqRatHandoff`

Proof. Start with the window rows S of Ω_Q. They are finite $\mathsf{StreamName}^{\uparrow}$ observations over the metric source X, so a consumer can read only the displayed windows before the rational handoff appears.

The row R is the regular rational readback attached to those same windows. It is therefore the row that connects the source-window data to the tolerance, point, and late-distance rows M, L, D before any $\mathsf{Real}^{\uparrow}$-facing seal is available.

Finally inspect H, C, P, N. These rows do not add another readback surface; they transport the same handoff by hsame, replay it by Cont, source it by Pkg, and name it by the local $\mathsf{NameCert}$ row. □

Theorem 1515.41 (SequentiallyCompleteMetric Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read produced from a finite Cauchy-window obligation package* $\Omega_Q = (X, S, R, M, L, D, H, C, P, N)$ *is confined to the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the late-distance ledger* D*, the* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The seal-facing read exports no host completion equality, representative choice, countable-choice schedule, ambient-completeness witness, or independent completed carrier.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRealSealNonescape`

Proof. Read the package order. The $\mathsf{Real}^{\uparrow}$-facing boundary can be reached only after S and R have displayed the finite stream windows and regular rational data, and after M, L, D have displayed the tolerance and late-distance rows.

Apply the Cauchy-window obligation and the RegSeqRat handoff above. Together they force every seal-facing request through the same finite BHist packet and the same rows S, R, D, H, C, P, N.

There is no remaining coordinate in Ω_Q for host completion equality, representative choice, choice schedules, ambient completeness, or an independent completed carrier. Hence the $\mathsf{Real}^{\uparrow}$ seal-facing read cannot escape the displayed finite route. □

Theorem 1515.42 (SequentiallyCompleteMetric limit readback package)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* SequentiallyCompleteMetric$^{\uparrow}$ *carrier, and let $\Omega_Q = (X, S, R, M, L, D, H, C, P, N)$ be its finite Cauchy-window obligation package. The rows of Ω_Q expose the completion-facing limit readback by first reading the metric source X, then the finite StreamName windows S and their* RegSeqRat$^{\uparrow}$ *readbacks R, then the modulus threshold M, displayed limit L, and late-distance ledger D. The readback is the same route named by Definition 1515.32 and exhausted by Definition 1515.37; it does not pass through quotient completion or a selected representative.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitReadbackPackage`

Proof. Start with the finite package of Definition 1515.38. Its first row is the metric source X, so every distance comparison is read inside the same source admitted by Definition 1515.2.

Read the sequence surface next. The windows S and the regular rational readback R supply exactly the finite stream data used by the induction route of Definition 1515.32.

Now read the modulus threshold, limit row, and late-distance ledger. The rows M, L, D give the displayed threshold, the displayed point, and the finite distance comparisons, so the limit readback is already present before any completion-facing consumer asks for it.

Finally replay through H, C, P, N and apply Definition 1515.37. Since the exhausted route has no quotient-completion, representative-selection, choice, ambient-completeness, independent completed-carrier, or Real$^{\uparrow}$ completeness coordinate, the package exposes only the finite limit readback. □

Theorem 1515.43 (SequentiallyCompleteMetric completion handoff obligations)**.** *Every* CompleteMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer of an accepted* SequentiallyCompleteMetric$^{\uparrow}$ *carrier $Q = (X, S, M, L, D, H, C, P, N)$ must read the finite Cauchy-window obligation package of Definition 1515.38 before using the handoff. In particular, the consumer must read the same metric source row, finite StreamName windows, RegSeqRat readbacks, modulus threshold, displayed limit row, late-distance ledger, transport, replay, provenance, and local naming rows named by Definition 1515.33 and Definition 1515.37.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionHandoffObligatio`

Proof. Project the completion-facing request. By Definition 1515.3, the CompleteMetric$^{\uparrow}$ boundary receives its limit witness from X, S, M, L, D rather than from an independent completion object.

Apply completion-consumer factorization from Definition 1515.33. A MetricCompletion$^{\uparrow}$ consumer has the same entry point, so it must read $X, S, M, L, D, H, C, P, N$ before using the completion route.

Insert the finite package Ω_Q from Definition 1515.38. Its RegSeqRat readback row R records the rational window data already consumed by the limit-modulus induction of Definition 1515.32.

The limit readback theorem Definition 1515.42 then supplies the displayed limit row and late-distance ledger to both consumers. Since Definition 1515.37 exhausts all completion requests through the same rows, no consumer can bypass the package through quotient completion, representative selection, countable choice, open-cover compactness, ambient completeness, or Real$^{\uparrow}$ completeness. □

Theorem 1515.44 (SequentiallyCompleteMetric regular limit extraction)**.** *For an accepted* SequentiallyCompleteM *carrier $Q = (X, S, M, L, D, H, C, P, N)$, the finite Cauchy-window package $\Omega_Q = (X, S, R, M, L, D, H, C, P, N)$ extracts the displayed limit row L and late-distance ledger D before any* CompleteMetric$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer can read a completion-facing witness. The extraction is the route of Definition 1515.38, the modulus induction of Definition 1515.32, and the readback package of Definition 1515.42.*

Proof. Project Ω_Q from the finite Cauchy-window definition. Its rows list the source metric, stream-window data, regular rational readback, modulus threshold, displayed limit, late-distance ledger, and structural transport before any consumer-facing handoff is considered.

Apply the limit-modulus induction. The induction forces each tolerance request to pass through M, then through the $\mathsf{StreamName}^{\uparrow}$ windows S and $\mathsf{RegSeqRat}^{\uparrow}$ readback R, before the displayed limit row L and ledger D are available.

Finally apply the limit readback package. It identifies L and D as the already displayed completion-facing witness, so a $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives the limit data only after the finite package has exposed those rows. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRegularLimitExtraction`

Theorem 1515.45 (SequentiallyCompleteMetric MetricCompletion consumer admission)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier factors through the same* $\mathsf{Met}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CompleteMetric}^{\uparrow}$ *limit boundary,* BHist *packet,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$ *rows as the finite Cauchy-window package. No consumer obtains a second completion carrier or a separate representative-selection surface.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricMetricCompletionConsumerAdm`

Proof. Start with the consumer admission point. By completion handoff obligations, the consumer must read the finite package of Definition 1515.38 before using any completion-facing witness.

Read the analytic rows of that package. The metric source is X, the finite sequence surface is S, the rational readback is R, and the limit boundary is supplied through M, L, D toward $\mathsf{CompleteMetric}^{\uparrow}$.

Read the structural rows next. The same BHist packet, hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row carry the route into the $\mathsf{MetricCompletion}^{\uparrow}$ request. Because the consumer has no other entry point in the carrier, the admission factors through exactly those rows. □

Theorem 1515.46 (SequentiallyCompleteMetric Real route boundary)**.** *The* $\mathsf{Real}^{\uparrow}$*-facing route exposed by an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the late-distance ledger, the finite seal-facing boundary, and the local* $\mathsf{NameCert}$ *rows of the same package. It exports no quotient completion, selected representative, countable choice, ambient completeness, or assertion that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRealRouteBoundary`

Proof. Read the route in package order. The finite $\mathsf{StreamName}^{\uparrow}$ windows are displayed first, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is the only rational handoff from those windows toward the late-distance ledger.

Apply the Real-seal nonescape theorem. The seal-facing read is confined to S, R, D, H, C, P, N inside the same BHist packet, with hsame transport, Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserving the package boundary.

There is no coordinate in the carrier for quotient completion, selected representatives, countable-choice schedules, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$. Therefore the Real-facing route ends at the displayed readback-and-seal boundary rather than at any host completion principle. □

Theorem 1515.47 (SequentiallyCompleteMetric display route)**.** *Every accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet exposes its root sequence-completeness surface in the order*

$$X, S, R, M, L, D, H, C, P, N.$$

Here X is the metric source, S is the $\mathsf{StreamName}^{\uparrow}$ *window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to that row, M is the displayed modulus, L is the displayed limit, D is the late-distance ledger, H is componentwise* hsame *transport, C is* Cont *replay, P is* Pkg *provenance, and N is the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricDisplayRoute`

Proof. Project the carrier of Definition 1515.1 and the finite Cauchy-window package of Definition 1515.38. The carrier supplies $X, S, M, L, D, H, C, P, N$, and the package displays the regular rational readback row R between the stream windows and the modulus.

Apply the completion handoff obligations of Definition 1515.43. They require every completion-facing consumer to read the metric source, finite stream windows, regular rational readback, modulus threshold, displayed limit row, late-distance ledger, transport, replay, provenance, and local naming rows before using the handoff.

Finally compose with regular limit extraction from Definition 1515.44. The extraction reaches L and D only after X, S, R, M have been displayed, and the route is then replayed through H, C, P, N. □

Theorem 1515.48 (SequentiallyCompleteMetric limit nonescape)**.** *For an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier, the displayed limit row L cannot be replaced by quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness. Any limit-facing read is confined to the displayed metric source, stream windows, regular rational readback, modulus, late-distance ledger, transport, replay, provenance, and local naming rows.*

Proof. Start with the ledger nonescape theorem Definition 1515.10. It states that the carrier has no row for quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Use the limit readback package of Definition 1515.42. The displayed limit row is read together with the late-distance ledger through the same finite Cauchy-window package, so the limit witness remains a row of the accepted packet.

The Real route boundary of Definition 1515.46 gives the final exclusion. Even when a consumer asks for a $\mathsf{Real}^{\uparrow}$-facing route, it reaches only the displayed readback-and-seal boundary, not a completeness theorem for $\mathsf{Real}^{\uparrow}$ or an ambient completion object. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricLimitNonescape`

Theorem 1515.49 (SequentiallyCompleteMetric RegSeqRat source route)**.** *Every sequence-completeness consumer reaches the late-distance ledger D only after the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows have been displayed over the same metric source. The route is source-first: it reads X, then S and R, then the modulus M, and only then the displayed limit row L and late-distance ledger D.*

Proof. Project the finite Cauchy-window package. Its sequence surface is not a free stream: the windows S and regular rational readback R are displayed over the metric source row X before any modulus or limit row is available.

Apply the Cauchy-sequence obligation of Definition 1515.5. It forces the source metric, stream windows, rational readback, and modulus threshold to be exposed before a sequence-completeness read can consume L or D.

Now use source-first tail read from Definition 1515.13. A tail read later consumed as a displayed limit read cannot begin from a free-standing limit, quotient-completion class, or selected representative, so every path to the late-distance ledger passes through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ first. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricRegSeqRatSourceRoute`

Theorem 1515.50 (SequentiallyCompleteMetric window-limit extraction)**.** *Every accepted* SequentiallyCompleteM *packet extracts its displayed limit row from finite* BHist *windows before any completion-facing consumer may use the packet. The extraction route reads the metric source row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the modulus row, and the* $\mathsf{Real}^{\uparrow}$ *seal-facing late-distance row through the same* hsame *transports and* Cont *replays, and only then admits the resulting boundary to a* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricWindowLimitExtraction`

Proof. Start from the display route of Definition 1515.47. It orders the rows as $X, S, R, M, L, D, H, C, P, N$, so the limit row L is never available before the finite stream windows and regular rational readback rows.

Apply regular limit extraction and the Real route boundary. The $\mathsf{RegSeqRat}^{\uparrow}$ source rows provide the rational tolerance readback, while the $\mathsf{Real}^{\uparrow}$ seal is reached only through the late-distance ledger D inside the same packet.

Finally use the completion handoff obligations. A $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives the extracted limit boundary only after hsame transport and Cont replay have preserved the displayed finite-window route; quotient completion, selected representatives, countable choice, and ambient Real completeness remain outside the carrier. □

Theorem 1515.51 (SequentiallyCompleteMetric tail cofinality)**.** *For every accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier, tail replacement by a cofinal finite window preserves the same completion-facing obligation surface. The replacement may change only the displayed* BHist *window inspected by the* $\mathsf{StreamName}^{\uparrow}$ *row; it must keep the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, modulus row,* $\mathsf{Real}^{\uparrow}$ *seal-facing ledger,* hsame *transport,* Cont *replay, provenance, and local* NameCert *rows on the same route before a* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer is admitted.*

Proof. Use source-first tail read from Definition 1515.13. A tail read cannot enter through a free-standing limit, so it must first display the metric source together with the finite stream windows and regular rational readback.

Apply the RegSeqRat source route above. The cofinal tail keeps the same R row as the rational source for the modulus and the late-distance ledger, so the $\mathsf{Real}^{\uparrow}$ seal-facing boundary is transported rather than regenerated.

The completion-consumer factorization then admits the same $\mathsf{MetricCompletion}^{\uparrow}$ consumer route. Since H, C, P, N preserve only the displayed packet, the cofinal tail cannot introduce quotient completion, selected representatives, countable choice, ambient completeness, or an independent completed carrier. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricTailCofinality`

Theorem 1515.52 (SequentiallyCompleteMetric completion consumer admission)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet must first display the finite* BHist *window-limit extraction and tail-cofinality routes over the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows. The admitted consumer sees the* $\mathsf{Real}^{\uparrow}$ *seal only as the terminal readback of the late-distance ledger, with* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *preserving the packet; it does not receive quotient completion, selected representatives, countable choice, open-cover compactness, ambient Real completeness, or a proof that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Begin with completion-consumer factorization. It identifies the consumer entry point with the displayed rows of the $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ carrier and refuses any independent completion object.

Attach window-limit extraction. The consumer must read the finite BHist windows, the $\mathsf{StreamName}^{\uparrow}$ route, the $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the modulus, and the $\mathsf{Real}^{\uparrow}$ seal-facing ledger before the displayed limit row can be used.

Attach tail cofinality. Cofinal tail replacement preserves the same source rows and structural replay, so admission to $\mathsf{MetricCompletion}^{\uparrow}$ is stable under finite-window refinement but still confined to the packet. The refusal clauses of the nonescape and Real boundary theorems exclude quotient completion, selected representatives, countable choice, open-cover compactness, ambient Real completeness, and $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionConsumer`

Theorem 1515.53 (SequentiallyCompleteMetric tail fusion)**.** *Let* $Q = (X, S, M, L, D, H, C, P, N)$ *and* $Q' = (X, S, M', L', D', H', C', P', N')$ *be accepted late-bound* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *ledgers sharing the same* $\mathsf{StreamName}^{\uparrow}$ *row and* $\mathsf{RegSeqRat}^{\uparrow}$ *tail schedule. If both ledgers are admitted by the displayed* $\mathsf{Met}^{\uparrow}$ *classifier, then their tail reads fuse through the modulus rows, late-distance ledgers, componentwise transports, replay rows, package provenance, and local NameCert rows before any* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff is available.*

Proof. Start from the shared source route. By Definition 1515.12, each tolerance request must pass through the displayed X, S rows and the same regular tail read before it reaches M, D or M', D'.

Apply Definition 1515.52. A completion-facing consumer is admitted only after the finite window-limit route and tail-cofinality route have been displayed through the same source rows.

Finally use Definition 1515.3. The $\mathsf{CompleteMetric}^{\uparrow}$ boundary receives its limit witness only after the source-first package has already replayed through H, C, P, N and H', C', P', N', so the two late-bound ledgers fuse at the displayed tail route before any handoff can read them. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricTailFusion`

Theorem 1515.54 (SequentiallyCompleteMetric displayed-limit uniqueness route)**.** *For an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier* $Q = (X, S, M, L, D, H, C, P, N)$*, separated uniqueness of the displayed limit is routed only through the metric source* X*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the modulus row* M*, the displayed limit row* L*, the late-distance ledger* D*, and the* $\mathsf{Met}^{\uparrow}$ *classifier. It is not routed through quotient completion, selected representatives, countable choice, ambient completion, or a host equality of completed points.*

Proof. Read uniqueness inside the displayed carrier. The source row X supplies the $\mathsf{Met}^{\uparrow}$ classifier, while S, M, L, D give the finite sequence, modulus, candidate limit, and late-distance evidence inspected by separatedness.

Apply Definition 1515.34. It already confines uniqueness to the displayed metric and sequence rows and refuses quotient completion, selected representatives, countable choice, and ambient completion.

The route is then preserved by the carrier boundary of Definition 1515.12. A uniqueness comparison may tighten the tolerance through M and D, but it cannot skip from L to an external completed-point equality. □

Lean4 ✓ `BEDC.Derived.Sequentially CompleteMetricUp.SequentiallyCompleteMetricDisplayedLimitUniquenessRou`

Theorem 1515.55 (SequentiallyCompleteMetric completion-factorization ledger)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier factors through the finite ledger route*

$$X \to S \to M \to L \to D \to H, C, P, N.$$

The consumer therefore receives the metric source, stream windows, modulus, displayed limit, late-distance ledger, transport, replay, provenance, and local naming rows as the complete visible route, not as an ambient completion object.

Proof. First project the carrier and the display route. The rows $X, S, M, L, D, H, C, P, N$ are the only rows available for sequence completeness, and the finite $\mathsf{StreamName}^{\uparrow}$ windows must be read before the limit-facing ledger is consumed.

Use Definition 1515.33 and Definition 1515.52. These results identify the $\mathsf{MetricCompletion}^{\uparrow}$ entry point with the displayed packet and require the window-limit and tail-cofinality routes before the consumer is admitted.

For a $\mathsf{CompleteMetric}^{\uparrow}$ consumer, apply Definition 1515.3. The handoff receives the same X, S, M, L, D rows and then the structural H, C, P, N replay. Thus every completion-facing read factors through the finite ledger route and imports no quotient completion, selected representative, countable choice, or independent completed carrier. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricCompletionFactorizationLedg`

Theorem 1515.56 (SequentiallyCompleteMetric source-first subsequence read)**.** *Let* $Q = (X, S, M, L, D, H, C, P,$ *be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Every subsequence, tail, or cofinal-window consumer that is later used by a* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff must first expose the* $\mathsf{Met}^{\uparrow}$ *source row* X*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback attached to those windows, the modulus row* M*, the displayed limit row* L*, and the late-distance ledger* D*. The handoff cannot begin from quotient completion, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independent completed-point carrier.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSourceFirstSubsequenceRead`

Proof. Start with the source-first tail read of Definition 1515.13. It forces every completion-facing tail packet to expose the metric source, finite stream windows, regular rational readback, and modulus row before any displayed limit or late-distance row can be inspected.

Pass from tails to subsequences by tail induction. The base tail has the displayed route X, S, M before L, D, and the successor step preserves the same source rows while tightening only the late-window request. Apply the limit-ledger coverage of Definition 1515.15; the displayed limit row and late-distance ledger are therefore read after the regular-source route rather than as free completion data.

Finally use the completion-factorization ledger of Definition 1515.55. Any $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives $X, S, M, L, D, H, C, P, N$ as the visible route, so quotient

completion, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and independent completed carriers remain outside the handoff. □

Theorem 1515.57 (Sequentially complete metric source-first subsequence readback)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. Every source-first subsequence read that is later inspected by a completion-facing consumer must replay through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before the consumer may inspect the subsequence. The replay exports no selected subsequence, countable-choice schedule, quotient limit, host equality proof, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSourceFirstSubsequenceReadb`

Proof. Project the packet rows. The source-first subsequence read of Definition 1515.56 already places the metric source X, stream windows S, regular rational readback, modulus row M, displayed limit row L, and late-distance ledger D before a completion handoff.

Replay the displayed route through the completion-facing rows. The completion-factorization ledger sends the same $X, S, M, L, D, H, C, P, N$ packet to $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$, while the StreamName, RegSeqRat, and Real rows remain the finite readback corridor through which the subsequence is observed.

The structural rows H, C, P, N transport, replay, source, and name the same data. Because no selected subsequence, countable-choice function, quotient limit, host equality proof, or ambient completeness row appears in the carrier, the completion-facing consumer cannot inspect the subsequence outside the displayed readback route. □

Theorem 1515.58 (Sequentially complete metric source-first completion nonescape)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *packet. Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of the source-first subsequence route must read the displayed rows $X, S, M, L, D, H, C, P, N$ together with the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readback rows before it can inspect subsequence data. The consumer cannot replace that route by quotient completion, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricSourceFirstSubsequenceCompl`

Proof. Apply the source-first subsequence read of Definition 1515.56. It exposes the metric source X, stream-window row S, regular rational readback, modulus row M, displayed limit row L, and late-distance ledger D before any completion handoff.

Then apply the source-first subsequence readback theorem, Definition 1515.57. Its replay keeps $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the visible route and exports no selected subsequence, quotient limit, countable-choice schedule, or ambient Real completeness theorem.

Finally attach the structural rows H, C, P, N. They transport, replay, source, and name the same packet, so every completion-facing consumer remains inside $X, S, M, L, D, H, C, P, N$ and the displayed readback rows. □

Theorem 1515.59 (SequentiallyCompleteMetric tail source exhaustion)**.** *Let $Q = (X, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier. Every completion-facing tail read is exhausted by the displayed source route through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows: the read must expose the metric source X, stream windows S, regular rational readback attached to S, modulus row M, displayed limit row L, late-distance ledger D,*

transport H, replay C, provenance P, and local naming N before any completion consumer is admitted. No further quotient completion, selected representative, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or independent completed-point carrier is exported.*

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricTailSourceExhaustion`

Proof. Start from the source-first subsequence read of Definition 1515.56. It already forces subsequence, tail, and cofinal-window consumers through X, S, M, L, D before either a $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ handoff can inspect the packet.

Next attach the display route and regular readback rows. By Definition 1515.47, the source route is ordered through the metric source, stream windows, regular rational readback, modulus, limit, late-distance ledger, transport, replay, provenance, and local naming rows.

Finally use Definition 1515.55. A completion-facing consumer receives exactly this finite ledger route, so the tail read is exhausted at the displayed $\mathsf{CompleteMetric}^{\uparrow}$ boundary and cannot export quotient completion, selected representatives, countable choice, ambient Real completeness, or an independent completed carrier. □

Theorem 1515.60 (SequentiallyCompleteMetric mature consumer boundary)**.** *Every mature consumer of an accepted* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *carrier factors through* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *the finite* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$ *rows before it may read a displayed limit. The consumer boundary has no route through quotient completion, selected representatives, countable choice, ambient completion, or a host equality of completed points.*

Proof. Start with the completion handoff and its obligations. By Definition 1515.3 and Definition 1515.43, a $\mathsf{CompleteMetric}^{\uparrow}$ request receives its limit witness only after the carrier has displayed the metric source, stream windows, modulus, limit row, late-distance ledger, and structural replay rows.

Now route the $\mathsf{MetricCompletion}^{\uparrow}$ side through the consumer theorems. The completion-consumer route, factorization theorem, and completion-factorization ledger force the same finite package through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before any completion-facing consumer is admitted.

Finally apply regular limit extraction and the nonescape ledger. The displayed limit row is extracted from the finite Cauchy-window package, while the ledger refuses quotient completion, selected representatives, countable choice, ambient completeness, and host completed-point equality. Thus every mature consumer reaches the displayed limit only through the named BEDC rows. □

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteMetricUp.SequentiallyCompleteMetricMatureConsumerBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ packet is generated from a displayed $\mathsf{Met}^{\uparrow}$ source, a regular Cauchy sequence read through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, a modulus, a displayed limit point, late-distance bound rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1515.3, Definition 1515.16, Definition 1515.11, Definition 1515.10, Definition 1515.43, and Definition 1515.60 bind the SequentiallyCompleteMetric carrier as a mature complete-metric and MetricCompletion consumer surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentiallyCompleteMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, or any assertion that $\mathsf{Real}^{\uparrow}$ is complete.

Upgrade path: The remaining work is formal-axis verification and bridge checking for the mature consumer boundary.

Chapter 1516

A Concrete Naming Certificate for $\mathsf{RegularCauchyBounded}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyBounded}^{\uparrow}$ packet records boundedness of a regular Cauchy name by finite dyadic ledgers and displayed window reads. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ carrier of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, the finite-window equivalence boundary of chapter 1404, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the bounded sequence consumer surface of chapter 1415. The certificate stores a usable finite bound ledger for one regular name; it does not identify quotient streams or select a limiting real.

Definition 1516.1 (Regular Cauchy bounded carrier)**.** *A* $\mathsf{RegularCauchyBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, W, D, E, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *request schedule used to inspect the name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and radius ledger attached to the inspected requests.*
4. *E is the optional* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *comparison row used when the same bound is transported across an equivalent regular name.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-name, stream-window, dyadic-ledger, and equivalence rows have been displayed.*
6. *L is the finite bound ledger, consisting of a displayed dyadic center-radius window and a finite verification row saying that every inspected request in W is contained in that window after the tolerance data in D are applied.*
7. *H stores componentwise* hsame *transport for the source name, request schedule, dyadic ledger, comparison row, real seal, bound ledger, continuations, provenance, and local naming row.*
8. *C stores displayed* Cont *replay from the regular name to stream requests, from stream requests to dyadic tolerance rows, from the tolerance rows to the bound ledger, and from the bound ledger to the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBounded}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBounded}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, W, D, E, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, real trichotomy, countable choice, host real order completeness, or Dedekind-cut construction.

Theorem 1516.2 (Finite bound ledger for regular Cauchy names)**.** *Every accepted* $\mathsf{RegularCauchyBounded}^{\uparrow}$ *carrier exposes boundedness through the finite route*

$$Q \longmapsto W \longmapsto D \longmapsto L$$

together with E, R, H, C, P, N. Thus a consumer can read a dyadic center-radius bound for the inspected regular Cauchy name without importing quotient stream equality, selected limits, real trichotomy, countable choice, host order completeness, or a Dedekind-cut construction.

Proof. Project the carrier of Definition 1516.1. The source row Q is inspected only through the displayed request schedule W, and each request in W is measured against the dyadic tolerance ledger D.

Read the bound ledger L next. Its center-radius row and verification row are finite data over the displayed requests, so boundedness is certified by the visible regular-name windows rather than by an ambient completeness or choice principle.

The optional equivalence row E transports the same bound shape across a displayed $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ comparison, while R records only the later real seal reached after the ledger is already present. The rows H, C, P, N preserve, replay, source, and name this route. Since the carrier lists no quotient stream equality, selected limit, trichotomy branch, countable-choice selector, host order-completeness row, or Dedekind-cut coordinate, the finite bound ledger exhausts the boundedness surface. □

Theorem 1516.3 (Real window readback from a bounded regular name)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{BoundedRealSequence}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyBounded}^{\uparrow}$ *packet must read the bound through the displayed regular Cauchy source, stream request schedule, dyadic tolerance ledger, and finite bound ledger before it reaches the stored real seal. The exported readback is exactly*

$$Q, W, D, L, E, R, H, C, P, N,$$

with no quotient stream equality, selected limit, trichotomy, countable choice, host completeness, or Dedekind-cut construction.

Proof. Begin with a seal-facing read. By Definition 1516.1, the real seal row R is downstream from Q, W, D, E, L, so the consumer must first inspect the regular rational source, the finite stream requests, the dyadic tolerances, and the bound ledger.

Apply Definition 1516.2. It identifies L as the finite center-radius boundedness surface and keeps the route attached to the same displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

A $\mathsf{BoundedRealSequence}^{\uparrow}$ consumer may reuse this bounded regular name as one inspected sequence value, but it receives only the same finite route and the stored real seal. No additional coordinate in the carrier can supply quotient stream equality, a selected limiting real, real trichotomy, countable choice, host completeness, or a Dedekind-cut construction. □

Falsifiable prediction: Every $\mathsf{Real}^{\uparrow}$ read from a $\mathsf{RegularCauchyBounded}^{\uparrow}$ packet factors through a finite regular-name request schedule, a dyadic tolerance ledger, and a displayed center-radius bound ledger before the real seal is consumed.

Independence witness: regseqrat, streamname, dyadicratcore, regularcauchyequivalence, real, boundedrealsequence: $\mathsf{RegularCauchyBounded}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one regular Cauchy source, request windows, dyadic tolerances, an optional equivalence transport row, a real seal, a finite bound ledger, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBounded}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName request windows, DyadicRatCore tolerance data, an optional RegularCauchyEquivalence transport row, a RealUp seal, a finite center-radius bound ledger, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1516.1, Definition 1516.2, and Definition 1516.3 bind the finite boundedness ledger for one regular Cauchy name and its real-window readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general completeness, selected limits, quotient stream equality, trichotomy, Dedekind-cut construction, countable choice, host real order completeness, or Lean verification.

Upgrade path: Lean closure requires carrier admission, finite bound ledger preservation, real-window readback, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyBoundedUp.

Chapter 1517

A Concrete Naming Certificate for $\mathsf{RealNameClassifier}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealNameClassifier}^{\uparrow}$ packet records equality inspection for two regular Cauchy real names as a finite tolerance classifier rather than as host equality of completed objects. It sits over the regular-name source of chapter 371, the regular rational rows of chapter 301, the common finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later $\mathsf{Real}^{\uparrow}$, metric, and algebra chapters a named classifier surface for regular Cauchy names without quotienting Cauchy classes, selecting limits, or importing trichotomy.

Definition 1517.1 (RealNameClassifier source packet)**.** *A* $\mathsf{RealNameClassifier}^{\uparrow}$ *source packet is a finite* BHist *record*

$$C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N)$$

with the following displayed rows:

1. *A and B are the two source* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows whose equality is being inspected.*
2. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposing the same finite precision requests to both sources.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows read from the two sources along W.*
4. *T is the finite tolerance classifier row, recording that every inspected precision request is discharged by a dyadic comparison between the corresponding observations of A and B.*
5. *R_A and R_B are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the two regular Cauchy names.*
6. *H records componentwise* hsame *transport for the two source names, common stream windows, dyadic tolerance rows, readback rows, classifier row, package provenance, seal row, and local naming row.*
7. *K records the displayed* Cont *replay from A, B through W, D_A, D_B, T, R_A, R_B.*
8. *P is the* Pkg *provenance over* $\mathsf{RealNameClassifier}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tolerance classifier and both regular rational readbacks have been exposed.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}$ *row.*

The packet has no coordinate for quotient Cauchy classes, host sequence equality, selected limits, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealNameClassifierUp.RealNameClassifierSourcePacketRows`

Theorem 1517.2 (RealNameClassifier tolerance obligation)**.** *For an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet* $C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N)$*, every classifier query factors through the common window* W*, the two dyadic tolerance rows* D_A, D_B*, and the finite tolerance classifier row* T *before either regular rational readback or terminal real seal is consumed.*

Proof. Project the accepted packet. The only precision schedule visible to both regular Cauchy names is the shared $\mathsf{StreamName}^{\uparrow}$ row W, so a classifier query has no independent observation route for A or B.

The dyadic tolerance rows D_A and D_B are read from W. The classifier row T records the finite comparison of these two displayed dyadic observations at the requested precisions.

The readback rows R_A, R_B and the seal row S are downstream of this finite comparison. Hence every equality-facing consumer first consumes the common window and dyadic tolerance data, then the tolerance classifier, and only afterward reads regular rational or real-facing rows. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierSource tolerance obligation`

Theorem 1517.3 (RealNameClassifier reflexive window)**.** *Let* $C = (A, A, W, D_A, D_A, T, R_A, R_A, H, K, P, S, N)$ *be an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet whose two displayed source rows are the same* $\mathsf{RegularCauchyName}^{\uparrow}$ *row and whose classifier query is read through the same* $\mathsf{StreamName}^{\uparrow}$ *window. Then the* $A \sim A$ *read is admitted by the reflexive classifier route through the existing* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient equality of Cauchy names, selected limit, host sequence equality, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierReflexiveWindow`

Proof. Project the accepted packet. Since both source entries are the same row A and both observation requests use the same window W, the displayed source routes agree componentwise before any classifier row is consumed.

Apply the tolerance obligation of Definition 1517.2. The classifier query must pass through W, D_A, D_A, T before the regular rational readback or terminal seal can be inspected, so the reflexive read has no independent equality oracle.

The two readback rows are the same $\mathsf{RegSeqRat}^{\uparrow}$ row attached to A. Transport by H and replay by K therefore preserve the displayed source, tolerance, classifier, and readback rows without changing the packet surface.

Finally apply the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the finite reflexive route has been displayed, and every consumed row is already accounted for by the carrier ledger. □

Theorem 1517.4 (RealNameClassifier common-window symmetry)**.** *Let* $C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K,$ *be an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet. Swapping the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *rows* A, B*, swapping* D_A, D_B*, and swapping* R_A, R_B *preserves the same local* $\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}$ *surface whenever the common* $\mathsf{StreamName}^{\uparrow}$ *window* W *and finite tolerance classifier row* T *are unchanged. The swapped packet exports no new equality oracle, selected limit, quotient completion, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierCommonWindowSymmetry`

Proof. Project the source packet of Definition 1517.1. The only shared observation route for the two regular Cauchy names is the common window W, and the only classifier row is T.

Use the tolerance obligation of Definition 1517.2. It forces both source directions through W, D_A, D_B, T before either readback row or the terminal seal can be consumed, so exchanging the two sides changes only the row order inside the displayed pair.

Apply refinement stability and the seal boundary, Definition 1517.10 and Definition 1517.11. Transport and replay preserve the same finite classifier surface, and the terminal seal still lacks coordinates for quotient equality, selected limits, external equality oracles, and completeness. □

Theorem 1517.5 (RealNameClassifier Cauchy-completion handoff)**.** *Let $C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K,$ be an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet. Every* $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer that reads equality-classifier data from C can read it only through the common* $\mathsf{StreamName}^{\uparrow}$ *window W, the dyadic rows D_A, D_B, the finite classifier row T, the readbacks R_A, R_B, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal S, transported and replayed by H, K, P, N. The handoff exports no quotient Cauchy class, selected limit, host equality of real names, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierCauchyCompletionHandoff`
↪ variant `BEDC.Derived.RealNameClassifierUp.RealNameClassifierCommonWindowCauchyCompletionHandoff`

Proof. Start with the tolerance obligation of Definition 1517.2. It forces every classifier query to pass through W, D_A, D_B, T before either readback row or the terminal seal is consumed.

Apply common-window symmetry, Definition 1517.4. Completion-facing consumers may exchange the two source names only by swapping the displayed dyadic and readback rows while keeping the same common window and classifier row.

Finally use the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. The seal is reached after the finite classifier route, and the ledger contains no coordinate for quotient Cauchy classes, selected limits, host equality, countable choice, or Real completeness. Hence a completion consumer receives exactly the common-window handoff described above. □

Theorem 1517.6 (RealNameClassifier common-window transitivity)**.** *Let $C_{AB} = (A, B, W, D_A, D_B, T_{AB}, R_A, R_B,$ and $C_{BC} = (B, C, W, D'_B, D_C, T_{BC}, R'_B, R_C, H_{BC}, K_{BC}, P_{BC}, S_{BC}, N_{BC})$ be accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packets. If both packets inspect $A \sim B$ and $B \sim C$ through the same common* $\mathsf{StreamName}^{\uparrow}$ *window W, and the two middle-source rows D_B, D'_B, R_B, R'_B are compatible by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback transports, then the composed $A \sim C$ classifier read is admitted through W, the transported rows D_A, D_C, R_A, R_C, the componentwise* hsame *transport through B, and the same* $\mathsf{Real}^{\uparrow}$ *seal boundary. The composed read exports no quotient equality of Cauchy names, selected limits, host sequence equality, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierCommonWindowTransitivity`

Proof. Project the two accepted packets. The hypotheses identify their observation route as the single common window W, so the $A \sim B$ and $B \sim C$ classifier reads cannot use independent precision schedules.

Apply the tolerance obligation of Definition 1517.2 to both packets. The first read exposes $W, D_A, D_B, T_{AB}, R_A, R_B$ before reaching its seal, and the second read exposes $W, D'_B, D_C, T_{BC}, R'_B, R_C$ before reaching its seal.

Use the displayed compatibility of D_B, D'_B and R_B, R'_B to transport the middle source componentwise through the two hsame rows. This composes the dyadic tolerance checks along the shared B row and leaves the endpoint rows D_A, D_C, R_A, R_C on the same window W.

Finally invoke ledger exhaustion and the seal boundary, Definition 1517.12 and Definition 1517.11. They account for every row consumed by the composed classifier read and rule out any escape through quotient equality, selected limits, host equality, choice, trichotomy, or completeness. □

Theorem 1517.7 (RealNameClassifier shared-window equivalence)**.** *For accepted* RealNameClassifier$^{\uparrow}$ *packets over the same* StreamName$^{\uparrow}$ *window, the tolerance obligation, common-window symmetry, and common-window transitivity assemble an equivalence-style classifier route on the displayed* RegularCauchyName$^{\uparrow}$ *rows. The route uses only* RegularCauchyName$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, the terminal* Real$^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already present in the packets. It exports no quotient equality, selected limit, host sequence equality, trichotomy, countable choice, or* Real$^{\uparrow}$ *completeness.*

Proof. Start with the tolerance obligation of Definition 1517.2. It forces every classifier read through the shared window, dyadic tolerance rows, finite classifier row, regular rational readbacks, and terminal seal before any equality-facing consumer receives the packet.

Symmetry is supplied by Definition 1517.4. Swapping the two source names changes only the displayed side order, because the same window, classifier row, readback rows, transport, replay, provenance, and local naming surface remain in place.

Transitivity is supplied by Definition 1517.6. When two adjacent accepted reads use the same window and compatible middle rows, the middle source is transported through the displayed hsame entries and the endpoint classifier read remains on that window.

Finally apply the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. These theorems account for every consumed row and prevent the equivalence-style route from becoming quotient equality, selected limits, host sequence equality, trichotomy, choice, or Real completeness. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierSharedWindowEquivalence`

Theorem 1517.8 (RealNameClassifier finite window coverage)**.** *Every accepted* RealNameClassifier$^{\uparrow}$ *classifier route assembled from the reflexive window route, common-window symmetry route, and common-window transitivity route consumes exactly one displayed* StreamName$^{\uparrow}$ *window W, the two* DyadicRatCore$^{\uparrow}$ *tolerance rows attached to that window, and the two* RegSeqRat$^{\uparrow}$ *readback rows before reaching the terminal* Real$^{\uparrow}$ *seal. The route may reuse the displayed window through componentwise* hsame *transport and* Cont *replay, but it may not introduce a second precision schedule, an external equality oracle, or a completion principle outside the carrier rows of Definition 1517.1.*

Proof. Start with the tolerance obligation of Definition 1517.2. It forces every classifier query to expose the common window W, the two dyadic tolerance rows, and the finite classifier row before any readback or seal row can be consumed.

For reflexive reads, use Definition 1517.3. The two source entries are the same displayed row, so the single window is reused rather than copied into a second schedule.

For exchanged and composed reads, use Definition 1517.4 and Definition 1517.6. Symmetry swaps the endpoint rows while leaving W fixed, and transitivity transports the middle source through compatible dyadic and regular readback rows on the same window.

Finally apply the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. They place the $\mathsf{Real}^{\uparrow}$ seal after the displayed window, tolerance, classifier, readback, transport, replay, provenance, and naming rows, and they leave no coordinate for a second window, external equality oracle, quotient completion, selected limit, trichotomy, choice, or Real completeness. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierWindowCoverage`

Theorem 1517.9 (RealNameClassifier window trichotomy refusal)**.** *Every accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *read over a common* $\mathsf{StreamName}^{\uparrow}$ *window exports only tolerance comparison, regular-rational readback, transport, replay, provenance, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows for the two displayed real names. Consequently the classifier cannot decide a constructive trichotomy row for those names: no accepted read supplies a row deciding equality, strict left order, or strict right order as an exhaustive alternative.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierWindowTrichotomyRefusal`

Proof. Apply the tolerance obligation of Definition 1517.2. A classifier read must first pass through the common window, the two dyadic tolerance rows, and the finite tolerance classifier row; this produces only bounded comparison data on the inspected window.

The shared-window equivalence route of Definition 1517.7 then assembles reflexive, symmetric, and transitive classifier behavior from the same displayed rows. It does not add an order row or a decision row beyond the tolerance classifier.

Finally use the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. The terminal seal is reached only after the displayed tolerance, readback, transport, replay, provenance, and naming rows have been consumed, and the ledger has no coordinate for trichotomy. Hence no constructive trichotomy witness can be exported from the accepted classifier packet. □

Theorem 1517.10 (RealNameClassifier refinement stability)**.** *Let* $C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N$ *be an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet. If a downstream consumer refines the common window* W *by a displayed* $\mathsf{StreamName}^{\uparrow}$ *transport and replays the two dyadic rows through* H *and* K*, then the resulting classifier read is the same local* $\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}$ *surface. The refinement changes only the finite observation window, not the two source names, dyadic tolerance obligation, regular rational readbacks, package provenance, or terminal seal boundary.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierRefinementStability`

Proof. Start with the displayed refinement of W. The transport row H identifies the refined window with the original common schedule in the componentwise hsame sense recorded by the packet.

Replay by K carries both dyadic rows through the same refinement. Thus the comparison row T is read after transporting D_A and D_B along a single common route, not by changing one source independently of the other.

The remaining rows R_A, R_B, P, S, N are attached to the same transported packet. The local naming surface is therefore stable under the displayed refinement, because no new source coordinate or external equality principle is introduced. □

Theorem 1517.11 (RealNameClassifier Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet reaches the seal row* S *only after the two* $\mathsf{RegularCauchyName}^{\uparrow}$ *source rows, the common* $\mathsf{StreamName}^{\uparrow}$ *window, the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the tolerance classifier row, and the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows have been displayed. The seal exports no*

quotient equality of Cauchy names, host sequence equality, selected limit, countable-choice schedule, trichotomy witness, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the carrier of Definition 1517.1. The terminal seal row S appears after the source rows, common finite window, dyadic tolerance rows, tolerance classifier, regular rational readbacks, transport, replay, and provenance rows.

Apply Definition 1517.2. A seal-facing consumer cannot bypass W, D_A, D_B, T, because those rows are the only route by which the classifier query is discharged.

The carrier contains no coordinate for any of the excluded principles. Consequently the $\mathsf{Real}^{\uparrow}$ seal is only a boundary row for the displayed finite classifier packet, not a source of quotient equality, host equality, choice, trichotomy, selected limits, or completeness. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierSealBoundary`

Theorem 1517.12 (RealNameClassifier ledger exhaustion)**.** *The local* $\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}$ *ledger for Definition 1517.1 is exhausted by carrier admission for* $A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N$*, common-window tolerance comparison, refinement transport, regular rational readback exposure, package provenance, terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, and local naming. No additional equality oracle or completion principle is available to a consumer of the packet.*

Proof. Carrier admission supplies exactly the displayed tuple. The analytic rows are the two regular Cauchy sources, the common window, the two dyadic tolerance rows, the classifier row, the regular rational readbacks, and the terminal real seal.

The structural rows are H, K, P, N. They transport, replay, source, and name the same finite packet, so their obligations do not add a hidden quotient or host-level equality coordinate.

The tolerance obligation and refinement stability theorems account for the classifier-facing reads. The seal-boundary theorem accounts for the real-facing reads. These routes cover every row in the ledger, and the carrier explicitly omits the excluded equality, choice, trichotomy, selected-limit, and completeness principles. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierLedgerExhaustion`

Theorem 1517.13 (RealNameClassifier obligation closure route)**.** *The source packet of Definition 1517.1, together with the tolerance obligation, reflexive window route, common-window symmetry, common-window transitivity, shared-window equivalence, finite window coverage, trichotomy refusal, refinement stability, seal boundary, and ledger exhaustion, assembles the full obligation surface for* $\mathsf{RealNameClassifier}^{\uparrow}$ *over* $\mathsf{RegularCauchyName}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. The assembled route closes the finite classifier obligations for equality inspection of two regular Cauchy real names while excluding quotient Cauchy classes, host sequence equality, selected limits, countable choice, trichotomy, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierObligationClosureRoute`

Proof. The carrier is fixed by Definition 1517.1. Its rows are exactly the two regular Cauchy sources, the common stream window, two dyadic tolerance rows, the finite classifier row, two regular rational readbacks, componentwise transport, continuation replay, package provenance, the terminal real seal, and local naming.

The first classifier-facing obligations are supplied by Definition 1517.2, Definition 1517.3, Definition 1517.4, and Definition 1517.6. These results account for tolerance comparison, reflexive

reuse of the common window, exchange of the two sources, and compatible composition through a shared middle source.

The remaining classifier-facing obligations are supplied by Definition 1517.7, Definition 1517.8, and Definition 1517.9. They account for equivalence-style reuse on the displayed window, finite window coverage, and refusal of a trichotomy row.

The boundary obligations are supplied by Definition 1517.10, Definition 1517.11, and Definition 1517.12. They show that refinement only transports the same finite route, that the $\mathsf{Real}^{\uparrow}$ seal is reached after the displayed classifier data, and that no ledger coordinate remains for a hidden equality oracle or completion principle.

Combining these rows gives the named obligation surface for $\mathsf{RealNameClassifier}^{\uparrow}$. Since every cited row is internal to the displayed packet, the route still excludes quotient Cauchy classes, host sequence equality, selected limits, countable choice, trichotomy, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1517.14 (RealNameClassifier scoped kernel route)**.** *The obligation closure route of Definition 1517.13 is scoped exactly by the displayed packet rows*

$$A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N$$

and by the primitives BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. Any accepted equality-inspection consumer of* $\mathsf{RealNameClassifier}^{\uparrow}$ *must consume the tolerance obligation, finite window coverage, refinement stability, seal boundary, and ledger exhaustion through those rows before it reads the local classifier surface. The route exports no quotient Cauchy class, host sequence equality, selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierScopedKernelRoute`

Proof. Start with the source packet of Definition 1517.1. Its carrier rows are precisely $A, B, W, D_A, D_B, T, R_A, R_B, H$ and its structural primitives are the displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Apply the tolerance obligation and finite window coverage, Definition 1517.2 and Definition 1517.8. They force every classifier read through the common $\mathsf{StreamName}^{\uparrow}$ window, the two $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, the finite classifier row, and the two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks before any real-facing seal is inspected.

Next use refinement stability, Definition 1517.10. Refining the observation window changes only the displayed window route and transported dyadic rows, so the same componentwise transport, replay, provenance, and local naming rows remain in force.

Finally use the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the finite classifier data has been displayed, and the ledger contains no coordinate beyond the rows listed above. Thus the scoped route is exactly the displayed kernel surface and no excluded completion or equality principle is exported. □

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierScopeRoute`

Theorem 1517.15 (Real name classifier refinement scope factorization)**.** *Every accepted* $\mathsf{RealNameClassifier}^{\uparrow}$ *refinement read factors through the common finite-window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D_A, D_B*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R_A, R_B*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S *before exposing a classifier decision. The factorization is scoped by the sibling surfaces of chapter 371, chapter 291, chapter 126, chapter 301, and chapter 38, together with the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 1517.1. It uses no quotient real, host equality, selected representative, trichotomy, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierRefinementScopeFactorization`

Proof. Project the carrier of Definition 1517.1. A refinement read begins with the same displayed rows $A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N$ as the original classifier packet.

Apply refinement stability, Definition 1517.10. The refined observation changes only the displayed window route and the transported dyadic rows; it does not introduce a second source, a host equality test, or a new decision coordinate.

Next compose the readback and seal obligations. The tolerance obligation, read through chapter 291 and chapter 126, forces the classifier through W, D_A, D_B, T; the regular rational rows are the readback surface of chapter 301; and the terminal seal is the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38.

The source names remain the regular Cauchy rows supplied by chapter 371. Componentwise transport through H, replay through K, provenance through P, and local naming through N preserve the same finite route, so the classifier decision is exposed only after the common window, dyadic ledger, readback rows, and terminal seal have been displayed.

No carrier coordinate supplies quotient real equality, host equality, a selected representative, trichotomy, or ambient completeness. The refinement read therefore factors through the stated finite scope rather than through any external real-number principle. □

Theorem 1517.16 (RealNameClassifier kernel-scope determinacy)**.** *Let*

$$C = (A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N)$$

and

$$C' = (A', B', W, D'_A, D'_B, T', R'_A, R'_B, H', K', P', S', N')$$

be accepted $\mathsf{RealNameClassifier}^{\uparrow}$ *reads whose classifier queries use the same common* $\mathsf{StreamName}^{\uparrow}$ *window* W*. Suppose the dyadic tolerance rows* D_A, D_B, D'_A, D'_B *are matched at the displayed precision requests, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_A, R_B, R'_A, R'_B *are matched by componentwise* hsame *transport, the two* Cont *replays* K, K' *follow the same window route, the* Pkg *provenance rows* P, P' *source the same classifier packet surface, and the* $\mathsf{Real}^{\uparrow}$ *seals* S, S' *are reached after the same finite tolerance and readback rows. Then* C *and* C' *expose the same classifier decision route inside the* $\mathsf{RealNameClassifier}^{\uparrow}$ *kernel scope. The route cannot branch through quotient Cauchy classes, host sequence equality, selected representatives, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierKernelScopeDeterminacy`

Proof. Apply the tolerance obligation of Definition 1517.2. Both accepted reads must pass through the shared window W, the displayed dyadic tolerance rows, and the finite classifier row before any readback or seal row is visible.

Use finite window coverage, Definition 1517.8. Since the window is common and the dyadic rows are matched at the displayed precision requests, neither read can introduce a second precision schedule or an external equality oracle.

Next apply refinement stability, Definition 1517.10. The componentwise hsame matches on R_A, R_B, R'_A, R'_B and the Cont replays K, K' transport the same regular readback surface through the same window route.

The seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12, account for the Real seals, provenance rows, and local names. The seals are downstream of the finite classifier data, and the ledger has no coordinate for quotient Cauchy classes, host sequence equality, selected representatives, trichotomy, or completeness.

Finally apply the refinement scope factorization of Definition 1517.15. With the same common window, matched tolerance rows, matched regular readbacks, transported replay, shared provenance surface, and downstream Real seal, both accepted reads expose the same classifier decision route inside the displayed kernel scope. □

Theorem 1517.17 (RealNameClassifier refined-window synchronization)**.** *Let*

$$C_0 = (A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N)$$

be an accepted $\mathsf{RealNameClassifier}^{\uparrow}$ *read, and let* C_1, C_2 *be two accepted refinement reads whose classifier queries refine the same base* $\mathsf{StreamName}^{\uparrow}$ *window* W*. Suppose both refinements transport the displayed dyadic rows through the same componentwise* hsame *rows, replay the same window route through* Cont*, match the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows before opening the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and keep the* Pkg *provenance and local* NameCert *row on the carrier surface of* C_0*. Then* C_1 *and* C_2 *expose one synchronized classifier route through the matching* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and terminal* $\mathsf{Real}^{\uparrow}$ *rows. The synchronized route exports no quotient Cauchy class, host sequence equality, selected representative, trichotomy, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierRefinedWindowSynchronization`

Proof. Start with the tolerance obligation and finite window coverage, Definition 1517.2 and Definition 1517.8. Both refinements must consume the base window W, the transported dyadic tolerance rows, the finite classifier row, and the regular rational readbacks before any seal-facing row is visible.

Apply refinement stability, Definition 1517.10. Since the two reads transport their dyadic rows by the same hsame rows and replay the same window route through Cont, the refinements change only the finite observation window and not the source names, classifier row, provenance, or local naming surface.

Use the seal boundary and ledger exhaustion, Definition 1517.11 and Definition 1517.12. The terminal $\mathsf{Real}^{\uparrow}$ row is downstream of the displayed dyadic and $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the ledger has no coordinate for quotient Cauchy classes, host equality, selected representatives, trichotomy, or completeness.

Finally apply kernel-scope determinacy, Definition 1517.16. With the same base window, transported dyadic rows, matched readbacks, common replay, shared provenance, and downstream seal, the two refinement reads determine the same synchronized classifier route inside the $\mathsf{RealNameClassifier}^{\uparrow}$ kernel scope. □

Theorem 1517.18 (RealNameClassifier bridge route)**.** *The public classifier surface of Definition 1517.15 bridges the concrete* $\mathsf{RealNameClassifier}^{\uparrow}$ *packet to the abstract real-name equality-inspection interface through the displayed* $\mathsf{RegularCauchyName}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exposes the common-window classifier and terminal seal as public packet data, without quotient equality, selected limits, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealNameClassifierUp.RealNameClassifierBridgeRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1517.14 and Definition 1517.15 make the finite classifier packet mature as a paper-side route over $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, the displayed packet rows $A, B, W, D_A, D_B, T, R_A, R_B, H, K, P, S, N$, and the primitives BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealNameClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host sequence equality, selected limits, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The paper-side route is mature; Lean verification and bridge checking for the same finite classifier packet remain unclaimed.

Constructive story: A $\mathsf{RealNameClassifier}^{\uparrow}$ packet is generated from two RegularCauchyName source rows, one common StreamName window, two DyadicRatCore tolerance rows, a finite tolerance classifier, two RegSeqRat readbacks, componentwise transport, continuation replay, package provenance, a RealUp seal, and local naming.

Chapter 1518

A Concrete Naming Certificate for $\mathsf{FastCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FastCauchySubsequence}^{\uparrow}$ packet records the finite extraction route from inspected Cauchy-modulus data to a fast regular subsequence. It reads the modulus surface of chapter 292, the fast-rate route of chapter 323, the selector ledger of chapter 405, the regular subsequence handoff of chapter 381, the regular rational readback of chapter 301, and the real-name seal of chapter 38.

The bridge is a local L10 NameCert object. It turns displayed Cauchy-modulus inspections into a cofinal fast subsequence route, then hands that route to the existing regular-subsequence and regular-rational surfaces without importing host subsequence equality, arbitrary countable choice, quotient Cauchy classes, selected limits, trichotomy, or real completeness.

Definition 1518.1 (FastCauchySubsequence carrier)**.** *A* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, M, Q, F, R, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *or Cauchy-name row whose windows are inspected before any subsequence is read;*
2. *M is the accepted* $\mathsf{CauchyModulus}^{\uparrow}$ *row recording the inspected modulus bounds for S;*
3. *Q is the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row, including the finite monotone-cofinal ledger used to choose later windows;*
4. *F is the* $\mathsf{FastCauchy}^{\uparrow}$ *fast-rate row obtained from the inspected modulus data along Q;*
5. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row that repacks the selected fast windows as a regular subsequence surface;*
6. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row determined by the same selected windows, radius data, and fast-rate ledger;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R and W have been projected;*
8. *H records componentwise* hsame *transports for S, M, Q, F, R, W, E and the naming support rows;*
9. *C records the displayed* Cont *replay from source windows to modulus inspection, cofinal selection, fast-rate routing, regular-subsequence handoff, regular rational readback, and real seal;*

10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares two carriers only through these displayed rows and stores no host subsequence equality, arbitrary countable-choice functional, quotient Cauchy class, selected limit, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 (definition) `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceUp`

Theorem 1518.2 (FastCauchySubsequence NameCert obligations)**.** *For an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source row, modulus row, selector row, fast-rate row, regular-subsequence row, regular rational readback, real seal, transports, replay routes, provenance, and local naming row;*
2. *modulus coverage, requiring every selected window used by the downstream route to be covered by the inspected bounds in M;*
3. *cofinal selector admission for Q, including the displayed monotone-cofinal ledger and the absence of a host-level choice selector;*
4. *fast-rate exactness, requiring F to be read from the modulus coverage row along the selected cofinal windows;*
5. *regular-subsequence window handoff, requiring R to consume the selected windows and transported dyadic radius data exposed by Q and F;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring W to be obtained from the same selected regular windows and not from an independent rational sequence;*
7. $\mathsf{Real}^{\uparrow}$ *seal delivery, requiring E to be reached only after R and W have been projected;*
8. hsame *transport through H for the source, modulus, selector, fast-rate, regular-subsequence, readback, seal, and support rows;*
9. Cont *replay through C for the same dependency route;*
10. Pkg *provenance through P over the cited sibling certificates and BEDC support rows;*
11. *local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *admission through N.*

No obligation row exports host subsequence equality, arbitrary countable choice, quotient Cauchy classes, selected limits, trichotomy, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The tuple displays exactly the source, inspected modulus, cofinal selector, fast-rate row, regular-subsequence handoff, regular rational readback, real seal, transports, replay routes, provenance, and local naming row.

Use the modulus and selector rows first. The row M supplies only inspected Cauchy-modulus bounds, and Q supplies only a finite monotone-cofinal selection ledger. Thus selected downstream windows must be covered by displayed modulus data rather than by a host-level choice functional.

Next use the fast-rate and regular-subsequence rows. The row F reads the chosen windows through the modulus coverage route, and R repacks exactly those windows and their transported dyadic radius data as a $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff.

Finally read the regular rational and real-facing rows. The row W is tied to the same selected windows, while E is placed after R and W in the carrier. The transport, replay, package, and local naming rows preserve that order componentwise, and the carrier has no remaining coordinate for host equality, countable choice, quotient classes, selected limits, trichotomy, or completeness. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceNamecertObligations`

Theorem 1518.3 (FastCauchySubsequence regular route)**.** *Every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet factors any* $\mathsf{Real}^{\uparrow}$*-facing read through the inspected modulus row, cofinal selector row, fast-rate row,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed. In the notation of Definition 1518.1, every such read follows*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad E$$

with the displayed hsame*,* Cont*,* Pkg*, and local* NameCert *provenance.*

Proof. Begin with the carrier order. The real seal row E occurs only after the source row S, the inspected modulus row M, the cofinal selector row Q, the fast-rate row F, the regular-subsequence row R, and the regular rational row W.

Apply the NameCert obligations of Definition 1518.2. They force modulus coverage before selection, selector admission before fast-rate readback, fast-rate exactness before the regular-subsequence handoff, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before any real-facing seal is consumed.

The rows H, C, P, and N carry the same dependency order under transport, replay, provenance, and local naming. Since no carrier coordinate stores host subsequence equality, arbitrary countable choice, quotient Cauchy classes, selected limits, trichotomy, or real completeness, every $\mathsf{Real}^{\uparrow}$-facing read factors through the displayed route. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegularRoute`

Theorem 1518.4 (FastCauchySubsequence modulus-selector coherence)**.** *For an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *row* Q *determine the same selected* $\mathsf{StreamName}^{\uparrow}$ *windows before either the fast-rate row* F *or the regular-subsequence handoff row* R *is read. Thus the selected windows used downstream are fixed by the displayed modulus-selector pair and not by host subsequence equality or a countable-choice selector.*

Proof. Project the carrier of Definition 1518.1. The only rows available before F and R are the source row S, the inspected modulus row M, and the cofinal selector row Q.

Apply the NameCert obligations of Definition 1518.2. They require modulus coverage before selection and selector admission before fast-rate readback, so the windows chosen by Q must be windows whose bounds are already displayed in M.

The regular route theorem, Definition 1518.3, then prevents any later row from changing that selection before F or R is consumed. Since the carrier has no coordinate for host subsequence equality or countable choice, coherence is exactly the displayed M, Q agreement. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceModulusSelectorCoherence`

Theorem 1518.5 (FastCauchySubsequence selector-window coverage)**.** *Every selected window in an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *is covered by the displayed cofinal selector* Q *together with the dyadic and fast-rate ledger read through* M *and* F*. The coverage route exposes only the finite selector ledger, inspected modulus data, fast-rate row, regular-subsequence handoff, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback; it does not import an independent subsequence function.*

Proof. Start with the selected windows named by Q. By carrier admission, these windows are rows of the finite packet and are read from the source row S rather than from an external selector.

Use Definition 1518.4. It binds the selected windows to the same inspected modulus row M before the fast-rate row F and the handoff row R are visible.

Apply the regular route theorem, Definition 1518.3. The route from Q to W passes through F and R, so each selected window is covered by the displayed cofinal selector, dyadic radius data carried in the fast-rate ledger, and regular rational readback. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSelectorWindowCoverage`

Theorem 1518.6 (FastCauchySubsequence selected-window induction)**.** *For every accepted* FastCauchySubsequenc *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the finite list of selected windows is determined by the displayed rows* M, Q, F, R, W*. More precisely, induction over the selected-window rows exposed by* Q *reads each step through*

$$M \longmapsto Q \longmapsto F \longmapsto R \longmapsto W,$$

with S, H, C, P, N *only transporting and naming the route. No induction step may branch on host subsequence equality, an independent selector function, countable choice, quotient Cauchy classes, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSelectedWindowInduction`

Proof. Project the selected windows named by Q. The carrier exposes them as a finite BHist ledger, so the induction is ordinary finite row induction over the displayed selector entries.

At the base row, Definition 1518.4 binds the selected source window to the inspected modulus row M. The fast-rate data F, regular-subsequence handoff R, and regular rational readback W then read the same window by Definition 1518.5.

For the successor row, repeat the same projection on the next displayed selector entry. Since the carrier has no coordinate for host subsequence equality, countable choice, quotient Cauchy classes, selected limits, or ambient completeness, the step can only reuse the displayed M, Q, F, R, W route. □

Theorem 1518.7 (FastCauchySubsequence tail-choice refusal)**.** *Let* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier. Any tail-choice claim made by a downstream consumer is accepted only when it is read from the finite selector windows already covered by* M, Q, F, R, W*. A claim whose witness is a host subsequence functional, a countable-choice schedule, quotient equality of Cauchy representatives, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness has no row in the carrier and is therefore not a* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *read.*

Proof. Start with selector-window coverage. It makes every selected window pass through the cofinal selector Q, inspected modulus data M, fast-rate row F, regular-subsequence handoff R, and $\mathsf{RegSeqRat}^{\uparrow}$ readback W.

A tail-choice claim must therefore name one of those finite rows. If it names an external functional or a countable schedule instead, there is no carrier coordinate that can be transported by H, replayed by C, sourced by P, or named by N.

The refusal is stable under finite-window induction from Definition 1518.6. Each additional selected window is another displayed M, Q, F, R, W row, not an opportunity to import host equality, quotient classes, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailChoiceRefusal`

Theorem 1518.8 (FastCauchySubsequence Real-seal nonescape induction)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, finite induction over the selected-window ledger preserves the Real-seal boundary. At every selected window, any route that reaches the terminal seal row* E *has already consumed the displayed rows*

$$M, \quad Q, \quad F, \quad R, \quad W.$$

Thus Real-facing reads cannot escape through host subsequence equality, countable choice, quotient Cauchy classes, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness at any finite induction stage.*

Proof. Use Definition 1518.6 as the induction spine. The base selected window is read through M, Q, F, R, W before any Real-facing consumer can ask for E.

For the successor window, apply the same finite row projection. Selector-window coverage keeps the new window inside the fast-rate ledger and regular-subsequence handoff before the regular rational readback W is public.

The Real-facing conclusion follows from Definition 1518.7. Since no induction stage carries a host selector, quotient Cauchy class, selected limit, trichotomy witness, or completeness coordinate, the terminal seal cannot expose any such data. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealNonescapeInduction`

Theorem 1518.9 (Fast Cauchy subsequence tail normal form)**.** *Every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet has a tail normal form whose selector is exposed only after the packet has factored through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$ *rows. For a carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the normal-form read is*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad E, \quad H, \quad C, \quad P, \quad N.$$

The exposed tail selector is therefore the cofinal finite-window row Q *read through inspected modulus bounds, dyadic radius support, fast-rate evidence, regular-subsequence handoff, regular rational readback, source windows, and the terminal Real seal, not an independent subsequence function or a countable-choice object.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequence tail normal form`

Proof. Project the carrier of Definition 1518.1. The source windows S and inspected modulus row M are the first rows available to a tail request.

Read the cofinal index next. The selector row Q supplies the finite monotone-cofinal ledger, and by Definition 1518.4 its windows are already tied to the inspected modulus data in M.

Read the dyadic radius and fast-rate information through F. The provenance row records $\mathsf{DyadicRatCore}^{\uparrow}$ support, and Definition 1518.5 keeps the selected windows inside the displayed fast-rate ledger.

Finally pass through the regular-Cauchy and Real-facing rows. The handoff R, readback W, and seal E are read in order by Definition 1518.3, while H, C, P, N transport, replay, source, and name the same route. Thus the tail selector appears only in the displayed normal form and cannot be replaced by a host subsequence or choice functional. □

Theorem 1518.10 (FastCauchySubsequence RealUp tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing tail read of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet factors through the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row, cofinal finite-window selector,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger,* $\mathsf{FastCauchy}^{\uparrow}$ *rate evidence,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *source windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. For a carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the handoff route is*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad E.$$

The structural rows H, C, P, N *transport, replay, source, and name this same route, so the tail read exposes no quotient Cauchy class, selected limit, independent subsequence function, countable-choice selector, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealUpTailHandoff`

Proof. Begin with the regular route of Definition 1518.3. It places every $\mathsf{Real}^{\uparrow}$-facing read downstream of the source row, inspected modulus, cofinal selector, fast-rate evidence, regular-subsequence handoff, and regular rational readback.

Use Definition 1518.4 to bind the selector to the inspected modulus windows. Then apply Definition 1518.5; the selected windows remain inside the displayed dyadic radius and fast-rate ledger before the handoff row R is read.

Finally apply Definition 1518.9. It records the same ordered read $S, M, Q, F, R, W, E, H, C, P, N$, so the Real-facing seal is reached only by the finite tail route and not by an external subsequence or choice object. □

Theorem 1518.11 (FastCauchySubsequence Real-seal selector boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumer of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet must pass through the modulus row* M*, cofinal selector row* Q*, fast-rate row* F*, regular-subsequence row* R*, and regular rational readback row* W *before the seal row* E *is visible. The seal boundary is therefore the displayed route*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad E$$

inside the packet $A = (S, M, Q, F, R, W, E, H, C, P, N)$*.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealSelectorBoundary`

Proof. Begin with Definition 1518.3. It already places every $\mathsf{Real}^{\uparrow}$-facing read downstream of S, M, Q, F, R, W and before the terminal row E.

Use Definition 1518.4 to bind the first two active rows of the boundary. The seal consumer cannot replace the displayed M, Q window selection by host subsequence equality or a countable-choice selector.

Then use Definition 1518.5. It forces the selected windows through the fast-rate ledger, regular-subsequence handoff, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before E is consumed. The structural rows H, C, P, N transport and name this route without adding another seal path. □

Definition 1518.12 (FastCauchySubsequence source surface)**.** *The* source surface *of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *is the finite* BHist *route*

$$\Theta_A = (S, M, Q, F, R, W, E, H, C, P, N)$$

read in the following order: source windows S*, inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *bounds* M*,* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector rows* Q*,* $\mathsf{FastCauchy}^{\uparrow}$ *rate rows* F*,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff rows* R*,* $\mathsf{RegSeqRat}^{\uparrow}$

readback rows W, terminal $\mathsf{Real}^{\uparrow}$ *seal rows E, and structural support H, C, P, N. Every Real-facing read of the source surface must pass through all six analytic rows M, Q, F, R, W, E after the source row S and before public data is named by N.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSourceSurface`

Theorem 1518.13 (FastCauchySubsequence cofinal-window obligation)**.** *For every accepted* $\mathsf{FastCauchySubsequen}$ *carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$, every cofinal window accepted by the source surface is a finite* BHist *row whose readback passes through the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M,* $\mathsf{CofinalSubsequence}^{\uparrow}$ *row Q,* $\mathsf{FastCauchy}^{\uparrow}$ *row F,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row R,* $\mathsf{RegSeqRat}^{\uparrow}$ *row W, and terminal* $\mathsf{Real}^{\uparrow}$ *row E. The obligation rejects any route that reads a Real-facing comparison from Q alone, from an independent subsequence function, or from a host-level choice selector.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalWindowObligation`

Proof. Project the source surface of Definition 1518.12. The carrier displays S, M, Q, F, R, W, E before the support rows, so a cofinal window entering through Q is still tied to source access and inspected modulus data.

Use Definition 1518.4 for the front of the route. It binds selector admission to the inspected modulus row, preventing a consumer from replacing M, Q by an external selector.

Apply Definition 1518.3 and Definition 1518.11 for the downstream route. They force the selected window through F, R, W, E before any Real-facing public read is exposed, while H, C, P, N only transport, replay, source, and name the same finite rows. □

Theorem 1518.14 (FastCauchySubsequence Real seal nonescape)**.** *Let $A = (S, M, Q, F, R, W, E, H, C, P, N)$ be an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier. No Real-facing read can escape the source surface before consuming the Cauchy-modulus, cofinal-selector, fast-rate, regular-subsequence, regular-rational, and Real-seal rows M, Q, F, R, W, E. In particular, a consumer that reaches E still receives only the finite route named by $S, M, Q, F, R, W, E, H, C, P, N$, not a selected limit, quotient Cauchy class, host real equality, trichotomy, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealNonescape`

Proof. Start with the Real-seal selector boundary. It places the seal row E behind M, Q, F, R, W and denies a direct path from the selector row to a Real-facing observation.

Use the cofinal-window obligation above for every selected finite window. Each such window remains a BHist row in the source surface and is read through the displayed CauchyModulus, CofinalSubsequence, FastCauchy, RegularCauchySubsequence, RegSeqRat, and Real rows.

Finally apply the NameCert obligations of Definition 1518.2. The support rows H, C, P, N preserve componentwise transport, replay, provenance, and local naming only for the displayed finite route, so the forbidden host-level data has no coordinate through which it could escape. □

Theorem 1518.15 (FastCauchySubsequence formal target ledger)**.** *Any later Lean target for* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *must encode the same displayed carrier rows used in this chapter: inspected source windows S, the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *row Q, the* $\mathsf{FastCauchy}^{\uparrow}$ *rate row F, the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff R, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback W, the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *row N. The target may prove only obligations over these rows and must not encode quotient stream equality, selected limits, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness as hidden support.*

Proof. Start with the carrier definition. It lists exactly the displayed source windows, CauchyModulus, CofinalSubsequence, FastCauchy, RegularCauchySubsequence, RegSeqRat, Real seal, transport, replay, provenance, and local NameCert rows.

Apply the local obligation theorems in order. The NameCert obligations, regular route, modulus-selector coherence, selected-window induction, cofinal-window obligation, and Real-seal nonescape theorem all consume the same rows and never introduce a second carrier.

The formal target therefore has no room for quotient stream equality, selected limits, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness. Any such field would be outside $S, M, Q, F, R, W, E, H, C, P, N$ and would not be the chapter's finite BHist ledger. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceFormalTargetLedger`

Theorem 1518.16 (FastCauchySubsequence cofinal tail obligation)**.** *Let $A = (S, M, Q, F, R, W, E, H, C, P, N)$ be an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier. Before any downstream consumer may read the* $\mathsf{FastCauchy}^{\uparrow}$ *row F, the cofinal selector row Q must expose, for every inspected modulus threshold in M, a finite tail window whose selector index lies beyond that threshold and whose source window is still covered by M. The obligation is a row-order condition inside $S, M, Q, F, R, W, E, H, C, P, N$; it is not supplied by a host subsequence function, countable choice, quotient Cauchy equality, selected limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalTailObligation`

Proof. Project the carrier and inspect the rows before F. The available analytic rows are the source windows S, the inspected modulus row M, and the cofinal selector row Q.

Use Definition 1518.5. Every selected window is already a finite row covered by the displayed selector, inspected modulus data, and fast-rate ledger, so a downstream fast-Cauchy read cannot invent a later selector outside the packet.

For each inspected threshold in M, exhaust the finite row order of Q until the first displayed selector row beyond that threshold is reached. Cofinality is read as the presence of that later row in the selector ledger, and coverage is read back through M.

Only after this tail window has been displayed may F consume it as fast-rate evidence. The structural rows H, C, P, N transport, replay, source, and name the same tail route, while the carrier lacks every refused host-level coordinate. □

Theorem 1518.17 (FastCauchySubsequence RegSeqRat and Real route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{CauchyModulus}^{\uparrow},\quad \mathsf{CofinalSubsequence}^{\uparrow},\quad \mathsf{FastCauchy}^{\uparrow},\quad \mathsf{RegularCauchySubsequence}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRat}$$

In carrier notation, the consumer reads M, Q, F, R, W, E with dyadic radius support recorded in the provenance row P before the terminal seal is available, and it reads no independent rational sequence, quotient Cauchy class, selected limit, host subsequence equality, countable-choice selector, trichotomy, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegSeqRatRealRoute`

Proof. Begin with the regular route of Definition 1518.3. It already places every Real-facing read downstream of the inspected modulus row M, cofinal selector row Q, fast-rate row F, regular-subsequence handoff R, and regular rational readback W.

Use the cofinal tail obligation above to expose the tail window before F is consumed. The selector row therefore supplies a later covered window for every inspected modulus threshold, rather than delegating the choice to an external subsequence.

Next apply Definition 1518.5. The selected window carries the dyadic radius support named in P, passes through the fast-rate row F, and is repacked by R before the $\mathsf{RegSeqRat}^{\uparrow}$ row W can be read.

Finally the terminal row E seals the same route as a $\mathsf{Real}^{\uparrow}$ read. Transport through H, replay through C, provenance through P, and local naming through N preserve the route and do not add any of the refused host-level data. □

Theorem 1518.18 (FastCauchySubsequence bridge scope)**.** *The bridge-facing public route of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet is exhausted by the displayed* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows, together with componentwise transport, replay, provenance, and local* NameCert *rows. No bridge-facing read can enter through host subsequence equality, a countable-choice selector, a quotient Cauchy class, a selected limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the regular route of Definition 1518.3. It orders every public read through the inspected modulus row, cofinal selector row, fast-rate row, regular-subsequence handoff, regular rational readback, and terminal real seal.

Use the cofinal tail obligation of Definition 1518.16 and the selector-window coverage of Definition 1518.5. These theorems keep the selected windows inside the displayed modulus and selector ledgers before the fast-rate and regular-subsequence rows are consumed.

Finally apply the RegSeqRat and Real route of Definition 1518.17. The dyadic support, regular rational readback, and real seal are read through the same finite packet, while H, C, P, N only transport, replay, source, and name the route. Since none of the displayed rows stores host equality, countable choice, quotient classes, selected limits, trichotomy, or ambient completeness, those endpoints cannot enter the bridge-facing read. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1518.1, Definition 1518.2, Definition 1518.3, Definition 1518.16, Definition 1518.17, and Definition 1518.18 bind the bridge-facing FastCauchySubsequence route through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceNamecertObligations`

Bridge status: none

Not claimed: This chapter does not claim host subsequence equality, arbitrary countable choice, quotient Cauchy classes, selected limits, trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: The finite fast-Cauchy subsequence route is mature on the paper axis; remaining work is limited to Lean-side cleanliness and bridge synchronization when a standard bridge theorem is supplied.

Constructive story: A $\mathsf{FastCauchySubsequence}^{\uparrow}$ packet is generated from inspected source windows, a CauchyModulus row, a CofinalSubsequence selector, a FastCauchy rate row, a RegularCauchySubsequence handoff, a RegSeqRat readback, a RealUp seal, componentwise transports, replay routes, provenance, and a local NameCert row.

Definition 1518.19 (FastCauchySubsequence selection surface)**.** *The* selection surface *of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *is the finite* BHist *subpacket*

$$\Sigma_A = (M, Q, F, W, H, C, P, N)$$

whose rows expose:

1. *the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *bounds in* M*;*

2. *the finite cofinal-window ledger in Q, with each displayed window tied to a source window of S;*
3. *the* FastCauchy$^{\uparrow}$ *rate row in F read from those same inspected windows;*
4. *the* RegSeqRat$^{\uparrow}$ *readback row W reached through the selected regular windows;*
5. *the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows carried by H, C, P, N.*

The surface has no coordinate for an infinite subsequence object, a choice functional, host stream equality, or a selected limit.

Lean4 (definition) `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSelectionSurface`
Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSelectionSurface namecert window`

Theorem 1518.20 (FastCauchySubsequence cofinal finite window)**.** *For every accepted* FastCauchySubsequence$^{\uparrow}$ *carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$ and every displayed target precision row consumed by the fast-rate route, the selection surface Σ_A contains a finite cofinal window row in Q whose source window is covered by the inspected* CauchyModulus$^{\uparrow}$ *data in M and whose transported radius data is read by F. The witness is a finite* BHist *row carried through H, C, P, N, not an infinite selected subsequence and not an appeal to countable choice.*

Proof. Project the selection surface of Definition 1518.19. The target precision row can be consumed only by the displayed fast-rate row F, and F is listed downstream of the modulus row M and cofinal-window ledger Q.

Use the NameCert obligations of Definition 1518.2. They require modulus coverage for every selected downstream window and require selector admission before the fast-rate readback is exposed.

The row Q is finite carrier data: it records a displayed monotone-cofinal ledger of windows rather than a host selector. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve that finite witness as the only available cofinal-window row. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalWindowCertificate`

Theorem 1518.21 (FastCauchySubsequence window nonescape)**.** *For every accepted* FastCauchySubsequence$^{\uparrow}$ *carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$, every selected window used by the packet is read first through the source row S, inspected* CauchyModulus$^{\uparrow}$ *row M, and cofinal selector row Q before the fast-rate row F, regular-subsequence handoff R, regular rational readback W, and terminal seal E are available. The support rows H, C, P, N transport, replay, source, and name that same finite-window route.*

Proof. Project the carrier of Definition 1518.1. Its selected-window data begins at S, M, Q, while F, R, W, E are downstream consumer rows.

Use the NameCert obligations of Definition 1518.2. They require modulus coverage before selection and selector admission before fast-rate readback, so no selected window can be consumed outside the displayed S, M, Q front.

Finally apply the regular route theorem, Definition 1518.3. The rows F, R, W, E read the same inspected windows in order, and H, C, P, N preserve the route without adding an independent selector, selected limit, or choice functional. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceWindowNonescape`

Theorem 1518.22 (FastCauchySubsequence tail-modulus lock)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, every downstream tail or window consumed by* F, R, W, E *is already covered by the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and the cofinal selector row* Q*. Thus the finite tail witness used by the fast-rate ledger, regular-subsequence handoff, regular rational readback, and terminal seal is locked to the inspected modulus-selector pair; no independent tail selector, host subsequence functional, or countable-choice functional can enter the route.*

Proof. Begin with modulus-selector coherence, Definition 1518.4. It binds the selected windows read by Q to the inspected modulus row M before the downstream rows F and R are consumed.

Next apply selector-window coverage, Definition 1518.5. That coverage carries each selected finite window from M, Q through the same fast-rate and regular-subsequence route used by F, R, W, E.

Finally use window nonescape, Definition 1518.21. Since every selected window must pass through S, M, Q before the downstream rows are available, the route has no coordinate for an independent tail selector or countable-choice functional. □

Theorem 1518.23 (FastCauchySubsequence tail-comparison extraction)**.** *For every accepted* FastCauchySubsequen *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, any two downstream tail comparisons consumed by* F, R, W, E *are answered by a displayed later cofinal window in* Q *whose source is covered by* S *and* M*, whose fast-rate row is read by* F*, and whose regular readback passes through* R, W *before the terminal seal* E*. Thus the comparison used by the L10* $\mathsf{Real}^{\uparrow}$ *completion route is extracted from the finite cofinal-window ledger and not from an independent tail selector, quotient stream equality, selected limit, or countable-choice functional.*

Proof. Start with the cofinal finite-window theorem, Definition 1518.20. It supplies a finite row in Q for the requested precision, and that row is already tied to source access in S and inspected modulus data in M.

Apply the tail-modulus lock above. The two tail comparisons consumed downstream cannot leave the displayed M, Q pair, because F, R, W, E read only windows covered by that inspected modulus-selector route.

Use cofinal $\mathsf{RegSeqRat}^{\uparrow}$ coverage and the modulus regular handoff, Definition 1518.24 and Definition 1518.25. They route the selected later window through F, then R, W, and only then to E.

The structural rows H, C, P, N transport, replay, source, and name the same finite witness. Since the carrier has no coordinate for an independent tail selector, quotient stream equality, selected limit, or countable-choice functional, the extracted comparison is exactly the displayed cofinal-window comparison used by the L10 Real completion route. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailComparisonExtraction`

Theorem 1518.24 (FastCauchySubsequence cofinal RegSeqRat coverage)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, each cofinal selected window that reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *reaches it only through the fast-rate row* F*, regular-subsequence handoff* R*, and regular rational readback row* W*. The route keeps the source and modulus rows* S, M, Q *visible and uses* H, C, P, N *only as structural support.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalRegSeqRatCoverage`

Proof. Begin with the cofinal finite-window theorem above. It gives each precision request a finite window in Q already covered by inspected data in M and source access in S.

Use the regular route theorem. It places every $\mathsf{RegSeqRat}^{\uparrow}$ readback behind F, R, W, so a cofinal selected window cannot jump from Q directly to a rational sequence or a terminal seal.

The NameCert obligations close the coverage claim. They force fast-rate exactness before the regular-subsequence handoff and the regular rational readback before any Real-facing row, while H, C, P, N transport and name the same finite route. □

Theorem 1518.25 (FastCauchySubsequence modulus regular handoff)**.** *For every accepted* FastCauchySubsequenc *carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$, the inspected modulus row M and regular-subsequence handoff R form one route to the* $\mathsf{Real}^{\uparrow}$ *seal:*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad E, \quad H, \quad C, \quad P, \quad N.$$

Thus the modulus witness, cofinal selector, fast-rate evidence, regular-subsequence handoff, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback are consumed as one finite package before the terminal seal is visible.*

Proof. Project A and read the displayed carrier order. The modulus row M is not separate from the selector row Q or the handoff row R; it is the inspected source of the selected windows later consumed by F, R, W.

Apply Definition 1518.2 and Definition 1518.3. Together they order modulus coverage, selector admission, fast-rate exactness, regular-subsequence handoff, regular rational readback, and terminal Real sealing along one route.

Finally use the existing source and structural support rows S, H, C, P, N. They preserve the same route under source access, componentwise transport, displayed replay, provenance, and local naming, so the handoff cannot be split into an external subsequence, quotient stream, selected limit, or countable-choice datum. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceModulusRegularHandoff`

Theorem 1518.26 (FastCauchySubsequence regular readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier factors through the selection surface*

$$\Sigma_A = (M, Q, F, W, H, C, P, N).$$

In particular, W reads the same cofinal finite windows certified by Q, the same inspected modulus bounds certified by M, and the same fast-rate ledger certified by F before any $\mathsf{Real}^{\uparrow}$ *seal E is visible. The readback exports no independent rational stream, no selected infinite subsequence, and no quotient Cauchy-class representative.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegularReadback`

Proof. Begin with Definition 1518.3. It places every real-facing route through M, Q, F, R, W before E.

Now restrict attention to the $\mathsf{RegSeqRat}^{\uparrow}$ row W. By Definition 1518.19, the rows available to W inside the selection surface are exactly the inspected modulus row M, the cofinal finite-window ledger Q, the fast-rate row F, and the structural rows H, C, P, N.

Apply Definition 1518.20 to each displayed precision read. The cofinal witness is already a finite BHist window row transported through the same support rows, so the regular rational readback cannot be replaced by an independent stream, a choice-selected subsequence, or a quotient Cauchy representative. □

Theorem 1518.27 (FastCauchySubsequence selector regularity pullback)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row* R *pulls its regularity witness back to the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row* Q *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* W *is read. The pullback is the displayed route*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad H, \quad C, \quad P, \quad N,$$

and it exports no selected limit, quotient stream equality, host real equality, or countable-choice selector.

Proof. Project the selection surface Σ_A of Definition 1518.19. The only selector-facing rows available to the regular handoff are M, Q, F, W, H, C, P, N, with R sitting in the carrier as the regular-subsequence handoff from those rows.

Use the modulus-selector coherence theorem, Definition 1518.4. It fixes the selected windows by the displayed M, Q pair before either the fast-rate row F or the regular-subsequence handoff R can be consumed.

Apply Definition 1518.26. The $\mathsf{RegSeqRat}^{\uparrow}$ row W reads the same finite cofinal windows and inspected modulus bounds, so the regularity witness in R is pulled back to the finite selector surface rather than to an independent stream or a selected limit. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSelectorRegularityPullback`

Theorem 1518.28 (FastCauchySubsequence cofinal readback non-escape)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the cofinal readback route from the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *row* Q *to the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* W *remains inside*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad H, \quad C, \quad P, \quad N.$$

It may feed $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{CofinalSubsequence}^{\uparrow}$*,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers through the displayed rows, but it cannot escape as a selected limit, quotient stream equality, host real equality, arbitrary countable choice, or an independent rational stream.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalReadbackNonescape`

Proof. Begin with the cofinal finite-window theorem, Definition 1518.20. Each precision request receives a finite window row in Q covered by the inspected modulus row M and transported by H, C, P, N.

Next apply the selector-window coverage theorem, Definition 1518.5. The route from Q to the regular rational readback passes through F and R, so W cannot be read from a separate stream or a host-level subsequence object.

Finally use the Real-seal selector boundary of Definition 1518.11. The $\mathsf{Real}^{\uparrow}$ seal E is visible only after M, Q, F, R, W, and the carrier has no coordinate for selected limits, quotient stream equality, host real equality, countable choice, or an independent rational sequence. □

Theorem 1518.29 (FastCauchySubsequence effective Cauchy consumer)**.** *Any effective-Cauchy or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *must consume the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M*, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row* Q*, the fast-rate row* F*, the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff* R*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* W *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The consumer receives finite evidence for an effective Cauchy route from* $\mathsf{StreamName}^{\uparrow}$ *windows to regular rational readback; it receives no selected limit, quotient stream equality, host real equality, or countable-choice functional.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceEffectiveCauchyConsumer`

Proof. Start with the regular route theorem, Definition 1518.3. It forces every real-facing read through S, M, Q, F, R, W before the seal E.

Use selector regularity pullback, Definition 1518.27. The regular handoff R is not an independent regular subsequence; it is pulled back to the inspected modulus and cofinal selector rows.

Now apply cofinal readback non-escape, Definition 1518.28. The readback route remains inside $M, Q, F, R, W, H, C, P, N$, so an effective-Cauchy consumer gets finite displayed evidence and cannot replace it by a selected limit, quotient stream equality, host real equality, or countable choice. □

Theorem 1518.30 (FastCauchySubsequence source carrier obligation)**.** *For every accepted* $\mathsf{FastCauchySubsequence}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the source carrier obligation binds the displayed source row* S *to inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *data* M *and* $\mathsf{FastCauchy}^{\uparrow}$ *rate data* F *before any cofinal readback is exported. The obligation is witnessed inside the finite* BHist *route*

$$S, \quad M, \quad Q, \quad F, \quad H, \quad C, \quad P, \quad N,$$

where H *supplies componentwise* hsame *transport,* C *supplies displayed* Cont *replay,* P *supplies provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *support, and* N *supplies the local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSourceCarrier`

Proof. Project the carrier of Definition 1518.1. The only source-facing rows available before regular readback are S, M, Q, F and the support rows H, C, P, N.

Use the NameCert obligations and modulus-selector coherence. They bind each selected source window to inspected Cauchy-modulus data in M before the fast-rate row F can be consumed.

The support rows preserve the same finite route: hsame transport is componentwise through H, Cont replay is displayed by C, provenance is recorded by P, and local naming is recorded by N. Since the carrier has no selected-limit or countable-choice coordinate, the source obligation is exactly the displayed BHist route. □

Theorem 1518.31 (FastCauchySubsequence root obligation)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the root obligation surface is the complete finite route over the inspected source row* S*,* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M*,* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row* Q*,* $\mathsf{FastCauchy}^{\uparrow}$ *rate row* F*,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row* R*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* W*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FastCauchySubsequence}^{\uparrow}}$ *row* N*. Every downstream effective-Cauchy, regular-rational, or real-facing consumer must read this surface before using the packet; no coordinate exports a selected limit, quotient Cauchy class, host real equality, arbitrary countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRootObligation`

Proof. Begin with the source carrier theorem, Definition 1518.30. It binds S, M, Q, F, H, C, P, N as the source-facing route before any cofinal readback is exported.

Add the regular readback obligation, Definition 1518.41. It forces R and W to consume the same displayed selector and fast-rate rows rather than an independent subsequence or rational stream.

Finally apply the Real-seal handoff and Real seal theorems, Definition 1518.35 and Definition 1518.42. They place the terminal seal E after S, M, Q, F, R, W and keep H, C, P, N as structural support. Since the carrier lists no escape coordinate, the displayed rows exhaust the root obligation surface. □

Theorem 1518.32 (FastCauchySubsequence modulus-selector obligation)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *and every selector window read by a downstream consumer, the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *covers that window before* $\mathsf{FastCauchy}^{\uparrow}$ *rate data,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff data, or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback data are consumed. The selector row* Q *therefore acts only as a finite cofinal-window ledger attached to* M*; it is not a countable-choice functional, a host subsequence object, or a quotient Cauchy representative.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceModulusSelectorObligation`

Proof. Start with modulus-selector coherence, Definition 1518.4. It identifies each selector window with inspected modulus data already present in M.

Use selector-window coverage and modulus-threshold extraction, Definition 1518.5 and Definition 1518.36. These keep every selected window and every threshold request inside S, M, Q, H, C, P, N before the regular route is read.

Now apply selector regularity pullback and cofinal readback non-escape. The regular-subsequence handoff R and readback row W consume the same M, Q, F route, so no later consumer can reinterpret Q as an external choice selector or as an independent rational stream. □

Theorem 1518.33 (FastCauchySubsequence Real-seal obligation)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed only after the inspected modulus row* M*, selector row* Q*, fast-rate row* F*, regular-subsequence handoff row* R*, and regular rational readback row* W *have all been displayed. The seal is a terminal row of the finite selector route; it does not supply a selected limit, quotient stream equality, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealObligation`

Proof. Read the Real-seal selector boundary, Definition 1518.11. It already prevents a real-facing consumer from reaching E before M, Q, F, R, W.

Use the effective Cauchy consumer theorem and the Real-seal handoff obligation. Together they keep every effective-Cauchy or real-facing read downstream of the inspected modulus, cofinal selector, fast-rate, regular-subsequence, and readback rows.

Finally apply the root obligation above. The structural rows H, C, P, N preserve and name the same route, while the carrier has no coordinate for selected limits, quotient classes, choice functionals, host equality, trichotomy, or completeness. Thus the Real seal is exposed only as the terminal row of the displayed route. □

Theorem 1518.34 (FastCauchySubsequence cofinal tail stability)**.** *Let* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier. If a finite tail-window comparison preserves the inspected* S, M, Q, F *rows by* hsame*, then every cofinal selector read used by* R *and* W *remains in the same displayed tail route*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad H, \quad C, \quad P, \quad N.$$

Thus selector stability is a finite tail-window fact over $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, not a host subsequence equality and not a countable-choice principle.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalTailStability`

Proof. Start from the finite tail-window comparison. The hypothesis says that the inspected source, modulus, selector, and fast-rate rows are preserved by hsame before the regular handoff is read.

Apply selector-window coverage and selector regularity pullback. These force the selected cofinal windows used by R and W to be the same windows certified by M, Q, F, with the dyadic and stream-window support carried through the displayed route.

Finally use cofinal readback non-escape. The $\mathsf{RegSeqRat}^{\uparrow}$ readback cannot leave $M, Q, F, R, W, H, C, P, N$, and the carrier has no coordinate for host subsequence equality or countable choice. Stability is therefore the finite hsame-tail statement over the displayed rows. □

Theorem 1518.35 (FastCauchySubsequence Real-seal handoff obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal handoff from an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier consumes only the displayed finite selector route*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad E, \quad H, \quad C, \quad P, \quad N.$$

The route opens $\mathsf{StreamName}^{\uparrow}$ *windows, inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius support,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the seal E; it does not claim a selected limit, quotient stream equality, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealHandoff`

Proof. Begin with the effective Cauchy consumer theorem above. It places every effective-Cauchy or real-facing consumer downstream of M, Q, F, R, W before E is exposed.

Use source carrier obligation and cofinal tail stability. The first binds the source to inspected modulus and fast-rate rows, while the second keeps the cofinal selector route inside the same finite tail-window package.

The terminal row E is therefore reached only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback W and the structural rows H, C, P, N preserve the displayed route. Since no carrier row stores a selected limit, quotient stream equality, host real equality, choice functional, or completeness principle, the $\mathsf{Real}^{\uparrow}$ handoff consumes only that finite route. □

Theorem 1518.36 (FastCauchySubsequence modulus-threshold extraction)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *and every downstream precision request visible to the packet, the threshold row used by the cofinal selector is extracted from the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row M before the selector row Q is consumed. Thus an effective consumer can cite the finite route*

$$S, \quad M, \quad Q, \quad H, \quad C, \quad P, \quad N$$

as displayed modulus-threshold evidence, without appealing to a selected subsequence, quotient Cauchy class, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceModulusThresholdExtraction`

Proof. Project the carrier and the NameCert obligations. The row M is listed before Q, and the obligations require modulus coverage for every selected downstream window before selector admission is read.

Use the regular route and modulus-selector coherence. They identify the windows consumed by Q with windows already covered by the inspected bounds in M, while H, C, P, N preserve the same finite source replay and local naming route.

The Real-seal handoff theorem keeps all real-facing consumption downstream of M, Q, F, R, W. Hence the threshold needed by an effective consumer is exported from M before Q and never from an ambient choice functional or a completed real object. □

Theorem 1518.37 (FastCauchySubsequence cofinal selector strict growth)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *row* Q *exposes a finite strict-growth ledger for the selected windows used by* F, R*, and* W*. Each later selected window is certified as a strictly later inspected source window, and the ledger is cofinal in the precision requests visible to the packet.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalSelectorStrictGrowth`

Proof. Begin with the cofinal finite-window theorem. It supplies a finite selector row in Q for each precision request and binds that row to the inspected bounds in M.

Apply selector-window coverage and cofinal tail stability. Coverage prevents Q from skipping outside the displayed modulus data, and tail stability records that later selector reads stay in the same hsame-preserved finite route.

The regular-subsequence handoff R and regular rational readback W consume only those selected windows. Therefore strict selector growth and cofinality are exported as finite data in Q, not as host subsequence equality or countable choice. □

Theorem 1518.38 (FastCauchySubsequence dyadic tail budget)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the fast-rate row* F *exposes the dyadic tail budget used by downstream regular consumers through* $\mathsf{DyadicRatCore}^{\uparrow}$ *support before the regular-subsequence handoff* R *and the readback row* W *are consumed. The visible budget route is*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad H, \quad C, \quad P, \quad N.$$

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceDyadicTailBudget`

Proof. Use the NameCert obligations for fast-rate exactness. They force F to be read from the modulus coverage row along the selected cofinal windows supplied by Q.

Next apply the selector regularity pullback. The handoff R repacks the same selected windows and transported dyadic radius data rather than introducing an independent regular subsequence.

Finally use cofinal readback non-escape. The $\mathsf{RegSeqRat}^{\uparrow}$ row W reads the same dyadic and selector route through M, Q, F, R, H, C, P, N, so the tail budget remains visible to effective consumers as $\mathsf{DyadicRatCore}^{\uparrow}$ data. □

Theorem 1518.39 (FastCauchySubsequence source replay exhaustion)**.** *For every accepted* FastCauchySubsequenc *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the structural rows* S, H, C, P, N *exhaust the source replay needed by any effective Cauchy consumer. Every read from source windows to modulus inspection, selector growth, fast-rate routing, regular-subsequence handoff, regular rational readback, and the terminal seal factors through these displayed rows.*

Proof. Start with the source carrier obligation. It binds the displayed source row S to the inspected modulus and fast-rate rows before any cofinal readback can be exported.

Use H, C, P, N in their carrier order. The row H supplies componentwise hsame transport, C supplies displayed Cont replay, P records provenance over $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ support, and N records the local NameCert row.

The Real-seal handoff theorem places all real-facing consumption after the same replay route. Since the carrier has no coordinate for a selected limit, quotient Cauchy class, countable-choice functional, or real completeness proof, S, H, C, P, N exhaust the source replay surface. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSourceReplayExhaustion`

Theorem 1518.40 (FastCauchySubsequence effective consumer export)**.** *Every effective-Cauchy, regular-rational, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier may export exactly the finite consumer package*

$$M, \quad Q, \quad F, \quad R, \quad W, \quad E.$$

The package gives inspected modulus thresholds, strict cofinal selector growth, dyadic fast-rate budgets, regular-subsequence handoff, regular rational readback, and the terminal real-name seal; it does not export selected subsequences, quotient Cauchy classes, countable-choice functionals, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Combine modulus-threshold extraction with cofinal selector strict growth. These two facts give the finite M, Q front of the package consumed by effective Cauchy sequence routes.

Add the dyadic tail-budget theorem. It supplies the F, R, W readback surface over $\mathsf{DyadicRatCore}^{\uparrow}$ and regular rational data, with no independent stream or selected limit.

Use source replay exhaustion only to justify that the package has already passed through the displayed source support. The exported consumer interface is therefore M, Q, F, R, W, E; the structural rows certify the route but do not enlarge it with choice, quotient equality, or completeness data. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceEffectiveConsumerExport`

The downstream explicit-modulus sequence route of chapter 1705 may consume $\mathsf{FastCauchySubsequence}^{\uparrow}$ when it needs finite fast-subsequence evidence: the inspected modulus data M and cofinal selector rows Q become the effective Cauchy sequence witness after the fast-rate and regular-subsequence handoff rows are displayed. The same relationship also feeds rate-realization consumers through the visible F, R, W route, without selecting a limit and without invoking countable choice.

Theorem 1518.41 (FastCauchySubsequence regular readback obligation)**.** *Every accepted* FastCauchySubsequence *packet exposes the inspected modulus row* M*, cofinal selector row* Q*, fast-rate row* F*, regular-subsequence row* R*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* W*, and the source* $\mathsf{StreamName}^{\uparrow}$ *window rows before any* $\mathsf{Real}^{\uparrow}$*, effective-Cauchy, or regular-rational consumer may read it. The obligation is the finite route*

$$S, \quad M, \quad Q, \quad F, \quad R, \quad W, \quad H, \quad C, \quad P, \quad N,$$

not an independent subsequence, quotient Cauchy class, selected limit, or countable-choice functional.

Proof. Project the accepted carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$. The carrier order displays the source windows, inspected modulus data, selector ledger, fast-rate row, regular-subsequence handoff, regular rational readback, and structural support rows before the real seal row can be consumed.

Use the NameCert obligations to bind the early route. Modulus coverage is read through M, selector admission through Q, and fast-rate exactness through F; no later consumer can introduce an unseen selector or an independent rational stream.

The regular-subsequence and readback rows then consume the same finite route. The handoff row R repacks the selected windows, and W reads those windows as regular rational data, while H, C, P, N only transport, replay, source, and name the displayed package. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegularReadbackObligation`

Theorem 1518.42 (FastCauchySubsequence Real seal)**.** *For an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal is reached only after the finite route through inspected modulus data, cofinal selection, fast-rate evidence, regular-subsequence handoff, regular rational readback, source-window support, componentwise transport, displayed replay, provenance, and local naming has been displayed. Thus any Real-facing read factors through the same root obligation surface before the seal row* E *is visible.*

Proof. Begin with the Real-seal selector boundary. It already places every $\mathsf{Real}^{\uparrow}$ seal consumer downstream of M, Q, F, R, W.

Add the source and structural rows from the carrier. The source row S supplies the inspected windows, while H, C, P, N preserve the route under componentwise transport, replay, provenance, and local naming.

Apply the regular readback obligation above. It identifies the complete root surface that a Real-facing read must traverse, so the seal cannot be reached by quotient stream equality, selected limits, arbitrary countable choice, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSeal`

Theorem 1518.43 (FastCauchySubsequence Phase Real and Completion exit route)**.** *Every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *consumer that exports a Phase Real and Completion exit read must factor through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow},$$

together with the displayed cofinal-selector and modulus rows of the carrier. More precisely, the exit read first consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget in* F*, the* $\mathsf{StreamName}^{\uparrow}$ *source-window schedule in* S*, the cofinal selector row* Q*, the inspected modulus row* M*, the regular readback row* W*, the regular-subsequence handoff* R*, and the terminal seal* E *before it may be used by a Real or Completion downstream packet. It cannot pass through quotient streams, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequencePhaseRealCompletionExitRoute`

Proof. Start from the Real seal theorem Definition 1518.42. It places every Real-facing read after the inspected modulus data, cofinal selection, fast-rate evidence, regular-subsequence handoff, regular rational readback, source-window support, transport, replay, provenance, and local naming rows.

Read the dyadic and selector components next. The dyadic tail budget theorem Definition 1518.38 exposes the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget through F, R, W, while the cofinal selector strict-growth theorem Definition 1518.37 keeps the selected windows inside the displayed $\mathsf{StreamName}^{\uparrow}$ source schedule and the inspected modulus row M.

Apply the effective consumer export theorem Definition 1518.40. It identifies the consumer-visible package as M, Q, F, R, W, E, with structural support from S, H, C, P, N and without an independent rational stream.

Thus a Phase Real and Completion exit read has already traversed the finite dyadic tolerance budget, source-window schedule, regular rational readback, and terminal Real seal before it is exported. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those routes can supply the exit. □

Theorem 1518.44 (FastCauchySubsequence choice-free completion window)**.** *Every completion-facing window served by an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *factors through* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any completion or Real seal is read. The window is therefore the finite BEDC route*

$$S \to M \to Q \to F \to R \to W \to E \to H, C, P, N$$

and not a quotient stream, selected limit, countable-choice functional, host subsequence equality, or ambient completeness row.

Proof. Begin with the regular route and selector-window coverage theorems. They force each selected window through the inspected modulus row M, cofinal selector row Q, fast-rate row F, regular-subsequence handoff R, and $\mathsf{RegSeqRat}^{\uparrow}$ readback W.

Next apply the Real-seal boundary and Phase Real and Completion exit route. These place the terminal row E after the regular rational readback and before any Real or completion-facing export.

Finally use the root obligation surface. The support rows H, C, P, N transport, replay, source, and name the same finite route, while the carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceChoiceFreeCompletionWindow`

Theorem 1518.45 (FastCauchySubsequence cofinal selector totality)**.** *For every displayed precision request made of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$, *the cofinal selector ledger* Q *supplies a finite source window whose readback is covered by* M, F, R, W, E. *This totality is a finite* BHist *coverage claim over the selected ledger and does not use host countable choice, an infinite subsequence object, or a selected real limit.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalSelectorTotality`

Proof. Use the selector-window coverage theorem for the requested precision. It reads the selected finite window from Q together with the inspected modulus row M and the fast-rate row F.

The cofinal selector strict-growth and dyadic tail-budget theorems keep the selected window inside the displayed source schedule S and the dyadic tolerance budget used by F, R, W.

Apply the effective consumer export theorem and the Real-seal obligation. They identify the consumer-visible package as M, Q, F, R, W, E with structural support from S, H, C, P, N, so the requested precision is served by a finite ledger row rather than by countable choice. □

Theorem 1518.46 (FastCauchySubsequence RegSeqRat completion handoff)**.** *Every completion-facing rational readback exported by an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* W *reached from* M, Q, F, R. *The handoff*

$$\mathsf{CauchyModulus}^{\uparrow},\ \mathsf{CofinalSubsequence}^{\uparrow},\ \mathsf{FastCauchy}^{\uparrow},\ \mathsf{RegularCauchySubsequence}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

is the only completion-facing rational readback route named by the carrier, and it exports no independent rational stream, quotient Cauchy representative, selected subsequence, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegSeqRatCompletionHandoff`

Proof. First read the regular readback and regular-subsequence pullback theorems. They identify W as the $\mathsf{RegSeqRat}^{\uparrow}$ row obtained from the same inspected modulus, cofinal selector, fast-rate, and regular-subsequence handoff data.

Next use cofinal readback non-escape. It prevents the selector row Q from bypassing F, R, W or from exporting an independent rational stream.

Finally compose with the Real-seal handoff obligation and Phase Real and Completion exit route. Since the terminal seal is reached only after W, every completion-facing rational readback uses that regular rational row and no separate quotient, choice, selected-limit, or completeness coordinate. □

Theorem 1518.47 (FastCauchySubsequence tail extraction obligations)**.** *Every tail extracted from an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *is a finite route from source windows through* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. The extraction reads the source schedule* S*, inspected modulus row* M*, cofinal selector ledger* Q*, fast-rate row* F*, regular-subsequence handoff* R*, regular rational readback* W*, terminal seal* E*, and structural rows* H, C, P, N*; it exports no host tail object, selected infinite subsequence, quotient stream equality, countable-choice functional, selected limit, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailExtractionObligations`

Proof. Project the source surface and selection surface. Definitions 1518.12 and 1518.19 expose the finite source windows, inspected modulus data, cofinal selector rows, fast-rate evidence, and support rows before any regular rational readback is consumed.

Use cofinal finite windows and dyadic tail budget. Theorems 1518.20, 1518.34, and 1518.38 keep each extracted tail inside the displayed BHist, hsame, and $\mathsf{DyadicRatCore}^{\uparrow}$ route.

Finally apply the regular readback obligation and the RegSeqRat completion handoff. They place the extracted tail through R, W, E before any Real or Completion consumer reads it, so the refused host subsequence, quotient, choice, selected-limit, and completeness objects have no carrier coordinate. □

Theorem 1518.48 (FastCauchySubsequence synchronized finite-window routing)**.** *Every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet synchronizes its finite source windows with the inspected* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *window route, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and support rows* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *The synchronized route exposes no countable-choice selector, selected limit, quotient Cauchy representative, host subsequence equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSynchronizedWindow`

Proof. Project the accepted carrier $A = (S, M, Q, F, R, W, E, H, C, P, N)$. The source windows S are read beside the modulus row M, cofinal selector Q, fast-rate row F, regular-subsequence handoff R, regular rational readback W, and Real seal E.

Apply the cofinal window, dyadic tail budget, and tail extraction obligations. These results keep every visible tail inside the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route.

The synchronized window is therefore a displayed finite route from source inspection to W and then E. Since the carrier has no coordinate for countable choice, selected limits, quotient representatives, host subsequence equality, or completeness, none of those objects is exported. □

Theorem 1518.49 (FastCauchySubsequence tail-selector coverage)**.** *The cofinal tail selector of an accepted* FastCauchySubsequence$^{\uparrow}$ *packet is covered exactly by finite* CofinalSubsequence$^{\uparrow}$ *windows tied to the same* CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. A downstream consumer can read only those covered tail windows and cannot extract a countable-choice functional, selected infinite subsequence, selected limit, quotient stream equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailSelectorCoverage`

Proof. Start with the selection surface and cofinal readback non-escape theorems. They bind the selector row Q to displayed finite windows and prevent it from bypassing the fast-rate, regular-subsequence, and regular rational rows.

Use the tail extraction obligations to cover each consumed tail. The extracted data pass through M, Q, F, R, W, E with StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ budgets, and the structural H, C, P, N rows.

Thus the selector coverage is finite and carrier-internal. It records which inspected windows are available to downstream consumers, but it does not supply a host function choosing infinitely many indices, a selected limit, quotient equality, or Real completeness. □

Theorem 1518.50 (FastCauchySubsequence Real-seal factorization)**.** *Every* Real$^{\uparrow}$ *seal exported by an accepted* FastCauchySubsequence$^{\uparrow}$ *packet factors through the displayed* CauchyModulus$^{\uparrow}$, CofinalSubsequence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows before reaching the terminal seal E. The factorization exposes only the finite regular rational route and exports no selected limit, quotient real equality, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealSealFactorization`

Proof. Read the RegSeqRat completion handoff and RegSeqRat tail readback theorems. They identify W as the regular rational row reached from the inspected modulus, cofinal selector, fast-rate, and regular-subsequence data.

Next apply the Phase Real and Completion exit route together with tail extraction obligations. These force every Real-facing read to pass through the same M, Q, F, R, W, E corridor with dyadic budgets, stream windows, transport, replay, provenance, and local naming.

The terminal seal E is therefore factored through W and its finite support rows. No accepted packet contains a selected limit, quotient real equality, countable-choice selector, host real equality, or ambient completeness coordinate from which a consumer could bypass that factorization. □

Theorem 1518.51 (FastCauchySubsequence cofinal fast-rate scope)**.** *For every accepted* FastCauchySubsequence$^{\uparrow}$ *carrier, the scoped route from the cofinal selector row Q to the fast-rate ledger F is generated by the inspected source windows, the* CauchyModulus$^{\uparrow}$ *row M, the* DyadicRatCore$^{\uparrow}$ *tail budget, and the structural H, C, P, N rows. It supplies only the finite fast-rate ledger consumed by R, W, E and supplies no host subsequence equality, countable-choice schedule, quotient Cauchy representative, selected limit, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalFastRateScope`

Proof. Begin with modulus-selector coherence and selector-window coverage. They bind the cofinal selector Q to the same inspected source windows and modulus row M before the fast-rate row F is read.

Next use the cofinal selector strict-growth theorem. The strict-growth ledger is a finite row of selected windows visible to the carrier, not a host subsequence function or a countable-choice schedule.

Close with the dyadic tail budget and effective consumer export theorems. They show that the fast-rate ledger reaches downstream consumers only through F, R, W, E with H, C, P, N as support, so the scoped route contains no extra equality, quotient, selected-limit, or completeness coordinate. □

Theorem 1518.52 (FastCauchySubsequence RegSeqRat tail readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *tail readback consumed by Real-and-Completion routes from an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet is read through the same* M, Q, F, R, W, E *route named by the carrier. The readback uses the displayed source windows, cofinal selector, dyadic tail budget, fast-rate row, regular-subsequence handoff, regular rational row, terminal Real seal, transport, replay, provenance, and local naming rows; it does not introduce an independent rational stream, a selected subsequence, quotient Cauchy equality, host real equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegSeqRatTailReadback`

Proof. Start with the regular readback theorem and selector regularity pullback. They identify the $\mathsf{RegSeqRat}^{\uparrow}$ row W as the readback obtained from the inspected modulus row M, selector row Q, fast-rate row F, and regular-subsequence handoff R.

Use cofinal readback non-escape and tail extraction obligations. The cofinal selector cannot bypass F, R, W, and each tail remains inside the same finite route over source windows, dyadic support, transport, replay, provenance, and local naming.

Finally apply the Phase Real and Completion exit route. Since E is reached only after W, every Real-and-Completion tail readback consumes the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row and no independent stream, selected subsequence, quotient, host equality, choice, or completeness row. □

Theorem 1518.53 (FastCauchySubsequence modulus selector tail extraction)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row* Q *determine the selected tail windows before the* $\mathsf{FastCauchy}^{\uparrow}$ *rate row* F *and the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row* R *are read. Each tail window is a finite* BHist *row read through* S, M, Q, H, C, P, N *and then handed to the fast-rate ledger; it is not an external subsequence function, a countable-choice schedule, or a selected limit.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceModulusSelectorTailExtraction`

Proof. Start with the modulus-selector coherence and selector-window coverage theorems. They bind every selected window in Q to inspected modulus data in M before any downstream fast-rate or regular-subsequence row is visible.

Use modulus-threshold extraction and cofinal tail stability. The precision request is first read from M, then served by a finite cofinal window in Q, with source replay and componentwise transport carried by S, H, C, P, N.

Only after this finite tail window is fixed can the route enter F and R. Since the carrier contains no coordinate for a host subsequence function, countable-choice schedule, quotient class, or selected limit, the selected tail is exactly the displayed modulus-selector row. □

Theorem 1518.54 (FastCauchySubsequence regular subsequence handoff totality)**.** *Every selected fast tail in an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier is handed to* $\mathsf{RegularCauchySubsequence}^{\uparrow}$

and $\mathsf{RegSeqRat}^{\uparrow}$ *through the same dyadic radius ledger. In the carrier notation* $A = (S, M, Q, F, R, W, E, H, C, P, N$ *the handoff reads*

$$M, \quad Q, \quad F, \quad R, \quad W$$

with the $\mathsf{DyadicRatCore}^{\uparrow}$ *budget displayed by* F *and preserved by* H, C, P, N*. No regular-subsequence or regular-rational consumer may replace that ledger by an independent rational stream, a selected subsequence, quotient Cauchy equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegularSubsequenceHandoffTotality`

Proof. Apply the dyadic tail budget theorem. It places the finite radius budget in the fast-rate row F after the inspected modulus and cofinal selector rows have been read.

Next use selector regularity pullback and the regular readback obligation. The $\mathsf{RegularCauchySubsequence}^{\uparrow}$ row R repacks the same selected windows and transported dyadic radii, and the $\mathsf{RegSeqRat}^{\uparrow}$ row W reads that same package rather than a separate rational sequence.

Finally apply the RegSeqRat completion handoff and RegSeqRat tail readback theorems. They keep every completion-facing rational read on the route through M, Q, F, R, W, E with structural support H, C, P, N, excluding independent streams, selected subsequences, quotient equality, host equality, and completeness. □

Theorem 1518.55 (FastCauchySubsequence seal boundary exhaustion)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier is exhausted by the displayed route*

$$S, M, Q, F, R, W, E, H, C, P, N.$$

Every Real-facing or completion-facing read must consume the source window, inspected modulus, cofinal selector, fast-rate ledger, regular-subsequence handoff, regular rational readback, terminal seal, transport, replay, provenance, and local naming rows in that route. The packet exports no quotient class, selected limit, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the Real-seal factorization and Real seal theorem. They place the terminal row E after the inspected modulus, cofinal selector, fast-rate ledger, regular-subsequence handoff, and regular rational readback.

Use the synchronized finite-window routing theorem to restore the source and support rows. It keeps S, H, C, P, N visible as the route moves from selected windows to regular readback and then to the terminal seal.

Finally combine the Phase Real and Completion exit route with cofinal readback non-escape. These results exhaust every Real-facing or completion-facing endpoint by the same finite carrier rows and leave no coordinate from which a quotient class, selected limit, choice schedule, trichotomy row, or completeness theorem could be exported. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceSealBoundaryExhaustion`

Theorem 1518.56 (FastCauchySubsequence uniform tail window)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the fast-rate selector supplies a uniform tail window for every* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row consumed by* W*. The window is read from the displayed route*

$$S, M, Q, F, R, W, H, C, P, N$$

with $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data visible through* M *and* F*; no later Real or Completion consumer may replace it by a host subsequence, quotient Cauchy class, selected limit, or countable-choice schedule.*

Proof. Read the cofinal fast-rate scope theorem. It generates the route from the selector row Q to the fast-rate ledger F using the inspected source windows, modulus row, dyadic tail budget, and structural rows.

Use the regular subsequence handoff totality theorem. The same dyadic radius ledger passes through R and then W, so each tolerance row seen by $\mathsf{RegSeqRat}^{\uparrow}$ is served by the previously displayed fast tail.

The uniformity is therefore carrier-internal: all tolerance requests use the same finite selector surface $S, M, Q, F, R, W, H, C, P, N$. The refused host subsequence, quotient, selected-limit, and choice objects have no coordinate in that surface. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceUniformTailWindow`

Theorem 1518.57 (FastCauchySubsequence cofinal modulus consumption)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier, the cofinal modulus row is consumed before any fast-rate, regular-subsequence, regular-rational, or Real-seal row can be exported. The consumption is a finite* BHist *handoff from* M *and* Q *through* F, R, W, E, H, C, P, N*, and it does not use countable choice, an external subsequence functional, or an infinite choice sequence.*

Proof. Start with modulus-threshold extraction. Each downstream precision request first reads the inspected $\mathsf{CauchyModulus}^{\uparrow}$ row M and only then enters the cofinal selector row Q.

Apply the modulus selector tail extraction theorem. It fixes the selected tail as a finite BHist row carried by S, M, Q, H, C, P, N before F and R are read.

Finally use the RegSeqRat tail readback and seal boundary exhaustion theorems. They carry the already consumed modulus row through W and E, so no countable-choice functional or infinite selection object is introduced. □

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalModulusConsumption`

Theorem 1518.58 (FastCauchySubsequence Real-completion tail readback route)**.** *Every Real-completion consumer of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier reads the cofinal fast tail through the displayed route*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqR}$

For $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, the source-window row* S*, inspected modulus row* M*, cofinal selector row* Q*, fast-rate row* F*, regular-subsequence handoff* R*, regular rational readback* W*, dyadic tail budget, and terminal seal row* E *are all displayed before any completion-facing read is named. The route exports no quotient Cauchy class, selected limit, countable-choice selector, trichotomy row, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealCompletionTailReadbackRoute`

Proof. Start with the modulus selector tail extraction theorem. It fixes the selected tail window from the source row S, inspected modulus row M, and cofinal selector row Q before the fast-rate or regular-subsequence rows are consumed.

Use the dyadic tail budget and regular subsequence handoff totality theorems. They carry the selected tail through the $\mathsf{DyadicRatCore}^{\uparrow}$ budget in F, the regular-subsequence handoff R, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row W without introducing an independent rational stream.

Finally apply seal boundary exhaustion and Real-name nonescape. The terminal seal E is reached only after the same finite route has been displayed, while H, C, P, N transport, replay,

source, and name that route. Since no carrier coordinate stores quotient Cauchy classes, selected limits, countable choice, trichotomy, host equality, or completeness, a Real-completion consumer can read only this finite tail-readback route. □

Theorem 1518.59 (FastCauchySubsequence Real-name nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *name row exported by an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *packet receives only the displayed regular-readback route*

$$S, M, Q, F, R, W, E, H, C, P, N.$$

The row refuses selected limits, quotient Cauchy classes, host real equality, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRealNameNonescape`

Proof. Begin with Real-seal factorization. It places the terminal seal E after the inspected modulus, cofinal selector, fast-rate ledger, regular-subsequence handoff, and $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use the uniform tail window theorem above to identify the regular readback received by W as the same finite tail route selected by M, Q, F, R. The support rows H, C, P, N preserve that route by componentwise transport, replay, provenance, and local naming.

The Real-name row therefore sees only the displayed regular-readback route. Since no accepted carrier contains a coordinate for selected limits, quotient Cauchy classes, host real equality, trichotomy, countable choice, or ambient completeness, the row cannot escape to those objects. □

Theorem 1518.60 (FastCauchySubsequence tail regularity)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, a cofinal fast-subsequence tail that preserves the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row* R *remains a* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible tail before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. The compatibility is witnessed by the same* BHist *tail window,* hsame *transport row,* Cont *replay,* Pkg *provenance, and local* NameCert *row used by* W*; it exports no selected limit, quotient stream equality, countable-choice selector, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailRegularity`

Proof. Start with the displayed carrier route $S, M, Q, F, R, W, E, H, C, P, N$. The source window S, modulus row M, cofinal selector row Q, and fast-rate row F determine the tail window before any regular rational readback is available.

Use regular subsequence handoff totality. It shows that the selected fast tail reaches R with the same dyadic radius ledger that was exposed by M and F, so the regular-subsequence row cannot replace the tail by an independent rational stream.

Now read the RegSeqRat tail readback theorem. The row W consumes the transported tail produced by R, and the structural rows H, C, P, N keep the source window, modulus, selector, and regular-subsequence data attached to the same finite packet.

Only after this $\mathsf{RegSeqRat}^{\uparrow}$-compatible row is displayed may the route enter E. Since the carrier has no coordinate for selected limits, quotient stream equality, countable choice, or ambient completeness, tail regularity is exhausted before Real-seal consumption. □

Theorem 1518.61 (FastCauchySubsequence cofinal regular window)**.** *For every accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier, the cofinal selector rows and* $\mathsf{StreamName}^{\uparrow}$ *windows expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers. In the carrier notation* $A = (S, M, Q, F, R, W, E, H, C, P, N)$*, a downstream consumer reads the displayed budget through*

$$S, M, Q, F, R, W, E, H, C, P, N$$

and cannot insert a separate dyadic tolerance, quotient rational stream, host subsequence equality, selected limit, or countable-choice schedule.

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceCofinalRegularWindow`

Proof. Read the synchronized finite-window routing theorem. It binds the $\mathsf{StreamName}^{\uparrow}$ windows in S to the inspected modulus row M, cofinal selector row Q, fast-rate row F, regular-subsequence handoff R, and regular rational readback W.

Apply the dyadic tail budget and uniform tail window theorems. The $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data visible to the selector is the same budget carried into F and then through R to W.

The terminal Real route is now fixed. Seal boundary exhaustion places E after the displayed regular rational row and keeps H, C, P, N as the transport, replay, provenance, and local naming support.

Thus $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive one displayed tolerance budget. The refused separate dyadic tolerance, quotient stream, host subsequence equality, selected-limit, and choice-schedule objects have no carrier coordinate. □

Theorem 1518.62 (FastCauchySubsequence tail Real-seal cofinality)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing fast-Cauchy subsequence tail read of an accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carrier* $A = (S, M, Q, F, R, W, E, H, C, P, N)$ *exposes, in order, the cofinal selector row* Q*, the inspected modulus row* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carried by* F, R*, and the terminal Real seal row* E*. The route may transport, replay, source, and name the same finite window through* H, C, P, N*, but it exports no separate cofinal selector, quotient stream equality, selected limit, countable-choice schedule, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceTailRealSealCofinality`

Proof. Project the carrier route for a Real-facing tail read. The stream window S is fixed first, and the cofinal selector row Q can be interpreted only against the inspected modulus row M.

Use Definition 1518.61. It identifies the selector window and the displayed tolerance budget as the same finite window consumed by the regular-subsequence handoff R and the $\mathsf{RegSeqRat}^{\uparrow}$ readback W.

The DyadicRatCore tolerance data is therefore already attached to the tail before E is read. The Real seal can certify only that displayed route, while H, C, P, N preserve it as transport, replay, provenance, and local naming.

Since the carrier contains no coordinate for an independent selector, quotient stream equality, selected limits, countable-choice schedules, or ambient Real completeness, the Real-facing read cannot escape the cofinal window. □

Theorem 1518.63 (FastCauchySubsequence RegSeqRat tail transport)**.** *For accepted* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *carriers with componentwise* hsame *transport on the cofinal window rows, the transported tail preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read. Concretely, transport of* S, M, Q, F, R, W *along* H *carries the same cofinal selector, inspected modulus, fast-rate ledger, regular-subsequence handoff, regular rational readback, and DyadicRatCore tolerance budget into the transported packet; only then may the Real seal row be consumed.*

Lean4 ✓ `BEDC.Derived.FastCauchySubsequenceUp.FastCauchySubsequenceRegSeqRatTailTransport`

Proof. Begin with componentwise transport through H. It identifies the source window S, inspected modulus M, cofinal selector Q, fast-rate ledger F, regular-subsequence row R, and regular rational readback W before any terminal Real data is compared.

Apply the tail regularity theorem and the cofinal regular-window theorem. The transported rows preserve the same BHist tail window and the same DyadicRatCore tolerance budget through R and into W.

The Real-name nonescape theorem then applies after the transported RegSeqRat route has been read. The seal row E is terminal, so it cannot replace the transported tail by a quotient stream, selected limit, countable-choice schedule, or ambient Real-completeness witness. □

Chapter 1519

A Concrete Naming Certificate for $\mathsf{MetricSubspace}^{\uparrow}$

Following Kelley and Bishop–Bridges metric-space practice, the $\mathsf{MetricSubspace}^{\uparrow}$ packet records the standard subspace metric as a finite BEDC NameCert carrier. It sits between the parent metric surface of chapter 57, the topology surface of chapter 88, the ball rows of chapter 1450 and chapter 1451, and the numerical rows of chapter 38 and chapter 37. The packet gives compact-subspace, closed-ball, completion, and topology consumers a displayed carrier for restricting distance to admitted points, without importing host subsets, quotient equality, ambient choice, or a second metric law.

Definition 1519.1 (MetricSubspace carrier)**.** *A* $\mathsf{MetricSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, A, D, H, C, P, N)$$

with the following visible rows:

1. *X is the parent* $\mathsf{Met}^{\uparrow}$ *row, including its point classifier, distance witness, metric-law ledger, and finite observation histories.*
2. *A is the displayed subset or admission-predicate row. A point is available to the subspace packet only when the parent point classifier of X and the admission row A both accept it.*
3. *D is the inherited distance ledger. It contains no independent distance value; each subspace distance read is the parent distance row of X applied to two admitted points.*
4. *H records componentwise* hsame *transport for the parent metric row, admission row, inherited distance ledger, restricted replay row, provenance row, and local naming row.*
5. *C records the displayed* Cont *route replaying point admission, subset membership, parent-distance evaluation, restricted metric-law transport, and downstream ball or topology handoff.*
6. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{MetricClosedBall}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSubspace}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host subset equality, quotient membership, an independently chosen distance, an ambient open-cover principle, countable choice, or a replacement topology.

Lean4 ✓ `BEDC.Derived.MetricSubspaceUp.MetricSubspaceTasteGate single carrier alignment`

Theorem 1519.2 (MetricSubspace NameCert obligations)**.** *For an accepted* $\mathsf{MetricSubspace}^{\uparrow}$ *carrier* $S = (X, A, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricSubspace}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the parent metric row, displayed admission row, inherited distance ledger, componentwise transport, restricted replay, provenance row, and local naming row;*
2. *subset-membership stability, requiring classifier transport by* H *to preserve the admission row* A *before any distance read is exposed;*
3. *inherited-distance restriction, requiring every subspace distance query to be a parent* $\mathsf{Met}^{\uparrow}$ *distance query on two points already admitted by* A*;*
4. *restricted metric-law transport, requiring symmetry, zero-distance, triangle, and rational or real comparison reads to be replayed only along the parent metric-law ledger named by* X *and transported by* H*;*
5. *ledger non-escape, stating that the packet exports no host subset equality, quotient membership, independent distance, ambient choice principle, open-cover principle, or replacement topology.*

Proof. Carrier admission is projection from Definition 1519.1. The tuple lists the parent metric row, admission predicate row, inherited distance ledger, transport, replay, provenance, and local naming rows.

Subset-membership stability is local to A and H. Transport first aligns the parent point classifier and admission row, so a downstream consumer cannot read a subspace point without replaying both admissions.

Inherited-distance restriction is the role of D and C. Every distance request is routed through two admitted points and then through the parent distance witness of X; no separate distance coordinate is present.

Restricted metric-law transport is inherited from the metric-law ledger of X. The replay row C uses the same parent symmetry, zero-distance, triangle, and numerical comparison rows after the admission checks have been performed.

The provenance row P and naming row N record exactly the dependencies available to the packet. Since the carrier has no coordinate outside X, A, D, H, C, P, N, the excluded host, quotient, choice, open-cover, and replacement-topology principles cannot be exported. □

Theorem 1519.3 (MetricSubspace distance restriction)**.** *Let* $S = (X, A, D, H, C, P, N)$ *be an accepted* $\mathsf{MetricSubspace}^{\uparrow}$ *carrier. If two displayed reads* x *and* y *pass the parent point classifier of* X *and the admission row* A*, then the subspace distance row between* x *and* y *is exactly the parent* $\mathsf{Met}^{\uparrow}$ *distance row read through* D *and replayed by* C*.*

Consequently every ball, closed-ball, completion, or topology consumer of S *factors through admitted endpoints before it can consume a distance comparison:*

$$(x, y) \longmapsto A(x), A(y) \longmapsto D_X(x, y) \longmapsto C, H, P, N.$$

The route is not a quotient of subsets, not host equality, and not a second metric carried beside X*.*

Proof. Project the accepted carrier. The only point rows visible to the subspace packet are parent points of X that also pass the admission row A.

For admitted x and y, the inherited ledger D reads the parent distance witness from X. The replay row C records this read after the two admission checks, so the subspace distance has no value other than the displayed parent distance.

The rows H, P, N transport and name the same finite route. Thus any ball, closed-ball, completion, or topology consumer receives a distance comparison only after admission of both endpoints and parent-distance evaluation, while quotient, host-equality, and independent-metric data remain absent from the carrier. □

Theorem 1519.4 (MetricSubspace closed complete inheritance)**.** *Let $S = (X, A, D, H, C, P, N)$ be an accepted* $\mathsf{MetricSubspace}^{\uparrow}$ *carrier. Suppose the parent metric row X carries a* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness route for a regular* $\mathsf{StreamName}^{\uparrow}$ *Cauchy stream, every displayed observation of the stream passes the admission row A, and the admission row A is closed under the parent limit witness.*

Then the same stream has a subspace $\mathsf{CompleteMetric}^{\uparrow}$ *limit witness with the same limit row. Every convergence read in the subspace factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow X, A, D, H, C, P, N \longrightarrow \mathsf{CompleteMetric}^{\uparrow},$$

and each late-distance comparison is the parent metric comparison identified by Definition 1519.3. The inherited witness exports no quotient subset, selected host equality, replacement limit row, or ambient completeness coordinate beyond the displayed parent $\mathsf{CompleteMetric}^{\uparrow}$ *route.*

Proof. Project the accepted carrier. The stream observations visible to the subspace packet are parent observations of X together with admission proofs from A, and the distance row available to them is the inherited row D replayed by C.

Apply the parent $\mathsf{CompleteMetric}^{\uparrow}$ route on X to the same regular $\mathsf{StreamName}^{\uparrow}$ Cauchy stream. This produces the parent limit witness and the parent late-distance comparisons used for convergence.

Use closedness of the admission row A under that parent limit witness to admit the limit row into the subspace packet. The limit is not chosen again; it is the same parent row now accompanied by the displayed admission evidence.

Finally apply Definition 1519.3. Every subspace convergence read is therefore the corresponding parent distance read after endpoint admission, transported by H and replayed by C through the same provenance and naming rows P, N. Since no additional carrier coordinate is present, quotient subset data, selected host equality, replacement limits, and independent ambient completeness data cannot enter the subspace witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSubspace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1519.1, Definition 1519.2, and Definition 1519.3 seed the finite L10 subspace-metric interface over parent metric, displayed admission, inherited distance, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSubspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subset equality, quotient membership, independent distance construction, ambient choice, open-cover compactness, replacement topologies, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subset-membership stability, inherited-distance restriction, restricted metric-law transport, and ledger non-escape for the same packet.

Constructive story: The construction starts from a displayed parent Metric row, filters points through an admission row, reads distance only from the parent metric ledger, and replays ball, closed-ball, completion, and topology consumers through the same finite route.

Chapter 1520

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyPair}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRegularCauchyPair}^{\uparrow}$ packet records the finite comparison surface between two regular Cauchy real names before the terminal $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the common finite-window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the convergence admission route of chapter 1509, the terminal real-name seal of chapter 38, and the finite equality-inspection classifier of chapter 1517. The chapter names the paired regular-Cauchy comparison object needed by later completion and real-classifier consumers without quotienting Cauchy classes or selecting host limits.

Definition 1520.1 (BishopRegularCauchyPair carrier)**.** *A* $\mathsf{BishopRegularCauchyPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, W, D_A, D_B, R_A, R_B, C, T, H, K, P, S, N)$$

with the following displayed rows:

1. *A and B are the two regular Cauchy real-name source rows being compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to expose both sources at the same precision requests.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers read from A and B along W.*
4. *R_A and R_B are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows produced from the same window and tolerance data.*
5. *C is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *admission row witnessing that the displayed reads are regular Cauchy reads at the inspected tolerances.*
6. *T is the* $\mathsf{RealNameClassifier}^{\uparrow}$ *finite comparison row comparing the two displayed regular rational readbacks before the real seal is consumed.*
7. *H records componentwise* hsame *transport for $A, B, W, D_A, D_B, R_A, R_B, C, T$ and the structural support rows.*
8. *K records the displayed* Cont *replay from the paired sources through the common windows, dyadic tolerance ledgers, regular rational readbacks, convergence criterion, classifier row, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyPair}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{RealNameClassifier}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both regular rational readbacks and the finite classifier row have been displayed.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyPair}^{\uparrow}}$ *row.*

The carrier stores no quotient Cauchy class, no selected host limit, no trichotomy witness, no open-cover compactness principle, and no host $\mathsf{Real}^{\uparrow}$ *equality coordinate.*

Theorem 1520.2 (BishopRegularCauchyPair RegSeqRat factorization)**.** *For an accepted* $\mathsf{BishopRegularCauchyPair}$ *carrier $Q = (A, B, W, D_A, D_B, R_A, R_B, C, T, H, K, P, S, N)$, every comparison read factors through the finite route*

$$A, B \longmapsto W \longmapsto D_A, D_B \longmapsto R_A, R_B.$$

Thus the paired comparison reaches $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks only by using the common* $\mathsf{StreamName}^{\uparrow}$ *window schedule and the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and it has no independent route to host sequence equality or quotient Cauchy classes.*

Proof. Project the carrier of Definition 1520.1. The only shared observation schedule for the two source names is W, so neither source has a separate precision route.

The tolerance rows D_A and D_B are attached to that same schedule. The replay row K therefore carries both sources through one displayed window route before any regular rational readback is visible.

The readback rows R_A and R_B occur after W, D_A, D_B in the carrier, and the transport row H preserves the same component order. Since no carrier coordinate contains host sequence equality or quotient classes, the displayed route is the whole $\mathsf{RegSeqRat}^{\uparrow}$ factorization surface. □

Theorem 1520.3 (BishopRegularCauchyPair Real classifier handoff)**.** *Let $Q = (A, B, W, D_A, D_B, R_A, R_B, C, T, H$ be an accepted* $\mathsf{BishopRegularCauchyPair}^{\uparrow}$ *carrier. Any* $\mathsf{RealNameClassifier}^{\uparrow}$ *consumer of Q reads the classifier row T only after the two regular Cauchy sources, the common finite-window schedule, the dyadic tolerance ledgers, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *admission row have been displayed. The subsequent* $\mathsf{Real}^{\uparrow}$ *seal row S exports only this finite classifier handoff, not quotient Cauchy equality, selected limits, trichotomy, open-cover compactness, or host real equality.*

Proof. Start with a classifier-facing consumer. Carrier admission displays $A, B, W, D_A, D_B, R_A, R_B, C, T$ in the stated dependency order, and the continuation row K replays the same order for every accepted read.

The classifier row T is downstream of the two $\mathsf{RegSeqRat}^{\uparrow}$ rows and the convergence row C. Hence it compares the displayed regular rational readbacks under the common finite window and dyadic tolerance data, rather than comparing host-level completed objects.

The seal row S is later than T. The package provenance P and local name N record the same packet, while the componentwise transport H moves the displayed rows without adding a new equality principle.

The carrier explicitly omits quotient Cauchy classes, selected limits, trichotomy, open-cover compactness, and host $\mathsf{Real}^{\uparrow}$ equality. Consequently the real-facing export is exactly the finite classifier handoff. □

Theorem 1520.4 (BishopRegularCauchyPair common-window induction)**.** *Let $Q = (A, B, W, D_A, D_B, R_A, R_B, C,$ be an accepted* $\mathsf{BishopRegularCauchyPair}^{\uparrow}$ *carrier. Every finite prefix of the paired comparison route is generated by induction on the displayed common* $\mathsf{StreamName}^{\uparrow}$ *window W. At each step, both*

source reads advance through the same W-coordinate before D_A, D_B, R_A, R_B, C, T, or S can be consumed.

Consequently no side-specific window, hidden tail oracle, quotient Cauchy equality, selected limit, trichotomy, or ambient completeness row can enter between the two sources and the $\mathsf{RealNameClassifier}^{\uparrow}$ *handoff.*

Proof. Induct on the finite displayed window rows in W. In the empty-window case, carrier admission exposes only the source rows, provenance rows, and local naming data, so no tolerance ledger, readback, classifier, or seal row has yet been consumed.

For the step case, extend the inspected window by one displayed coordinate. The componentwise transport row H keeps the new coordinate shared by the two sources, and the continuation replay row K advances both source reads through that same coordinate before any downstream row is replayed.

Projecting Definition 1520.1 then fixes the remaining order: the dyadic ledgers D_A, D_B feed the readbacks R_A, R_B, the convergence row C admits the displayed reads, the classifier row T follows those rows, and the seal S is terminal.

Composing this induction with Definition 1520.2 and Definition 1520.3 leaves no carrier coordinate through which a side-specific precision route, hidden tail oracle, quotient equality, selected limit, trichotomy, or ambient completeness witness could intervene. □

Theorem 1520.5 (BishopRegularCauchyPair symmetric classifier route)**.** *Let $Q = (A, B, W, D_A, D_B, R_A, R_B, C, T$ be an accepted* $\mathsf{BishopRegularCauchyPair}^{\uparrow}$ *carrier. Form the swapped displayed packet*

$$Q^{\mathrm{op}} = (B, A, W, D_B, D_A, R_B, R_A, C, T, H, K, P, S, N).$$

Then Q and Q^{op} expose the same common-window comparison route to $\mathsf{RealNameClassifier}^{\uparrow}$ *and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. The swap changes only the order in which the two source sides and their displayed tolerance and readback rows are named; it does not add a quotient Cauchy class, selected limit, side-specific window oracle, trichotomy, open-cover compactness, host equality, or ambient completeness coordinate.*

Proof. Apply Definition 1520.4 to Q. Both source reads are generated over the single displayed $\mathsf{StreamName}^{\uparrow}$ window W, so swapping the names of the two source sides preserves the finite induction spine and leaves no coordinate where a side-specific window route can enter.

Next use Definition 1520.2. The factorization route has the form

$$A, B \longmapsto W \longmapsto D_A, D_B \longmapsto R_A, R_B.$$

After the displayed swap, this is the same route with the two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows and the two $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers interchanged over the unchanged window W.

Finally apply Definition 1520.3. Since the classifier row T is downstream of those displayed readbacks and the terminal seal row S is downstream of T, the handoff to $\mathsf{RealNameClassifier}^{\uparrow}$ and the $\mathsf{Real}^{\uparrow}$ seal boundary are the same finite BEDC packet after the swap. The carrier projection still contains no coordinate for quotient classes, selected limits, trichotomy, open-cover compactness, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyPair}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1520.1, Definition 1520.2, Definition 1520.3, Definition 1520.4, and Definition 1520.5 seed the finite paired regular-Cauchy comparison surface before the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, selected limits, trichotomy, open-cover compactness, host Real$^{\uparrow}$ equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window factorization, dyadic tolerance compatibility, RegSeqRat readback exposure, convergence admission, classifier handoff, terminal seal nonescape, provenance, and ledger exhaustion for the same finite packet.

Constructive story: A BishopRegularCauchyPair$^{\uparrow}$ packet is generated from two regular Cauchy real-name source rows, a shared StreamName window schedule, two DyadicRatCore tolerance ledgers, two RegSeqRat readbacks, a CauchyConvergenceCriterion admission row, a RealNameClassifier comparison row, componentwise transports, replay routes, provenance, a terminal RealUp seal, and a local NameCert row.

Chapter 1521

A Concrete Naming Certificate for $\mathsf{MooreSmithConvergence}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{MooreSmithConvergence}^{\uparrow}$ packet records ordinary Moore–Smith convergence as a finite directed-net observation surface. It sits between the directed Cauchy evidence of chapter 589 and the boundary consumers in chapter 1473 and chapter 1506: the packet names convergence toward a displayed completion boundary without turning the directed net into a quotient object, a chosen limit point, or an ambient completeness theorem.

Definition 1521.1 (Moore–Smith convergence carrier)**.** *A* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier is a finite directed-net* BHist *packet*

$$M = (D, V, U, C, B, R, L, H, K, P, N)$$

with the following displayed rows:

1. *D is the finite directed index-window ledger, naming the inspected indices, the tail windows, and the common upper-window witnesses used by the packet.*
2. *V is the net-value window, exposing the finitely displayed values of the Moore–Smith net on the addresses recorded by D.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, recording the displayed entourages used to test convergence at the inspected boundary.*
4. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row, stating that the same finite directed windows support the Cauchy side of the net evidence.*
5. *B is the boundary row, read either through* $\mathsf{UniformCompletion}^{\uparrow}$ *or through* $\mathsf{MetricCompletion}^{\uparrow}$*, naming the displayed completion boundary to which the net is tested.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite rational coordinates exposed by the inspected net values and boundary reads.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *readback row reached from the regular rational row when a real-facing consumer asks for a displayed boundary coordinate.*
8. *H records componentwise* hsame *transport for D, V, U, C, B, R, L and the structural support rows.*
9. *K records the displayed* Cont *replay from directed windows through net values, entourages, Cauchy tail evidence, completion boundary reads, regular rational readbacks, and real readbacks.*

10. *P is the* Pkg *provenance over* $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithConvergence}^{\uparrow}}$ *row.*

The carrier exports only this finite convergence-to-boundary packet. It contains no quotient of nets, no selected Moore–Smith limit, no global completeness assertion, no cofinal choice function, and no host-level equality of completion points.

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergenceTasteGate single carrier alignment`

Theorem 1521.2 (Moore–Smith convergence NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MooreSmithConvergence}^{\uparrow}}$ *surface for Definition 1521.1 consists of six finite obligations:*

1. *carrier admission for the displayed directed-net packet M;*
2. *directed-tail coverage, stating that every displayed entourage request is routed to a tail window recorded in D;*
3. *net-value exposure, stating that the values used in the convergence test are exactly the displayed coordinates of V;*
4. *Cauchy compatibility, stating that the same directed windows feed the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row C;*
5. *completion-boundary exposure, stating that boundary reads pass through B before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback is exported;*
6. *ledger non-escape, stating that a consumer request without a displayed tail witness or boundary row is recorded outside the packet rather than exported as convergence data.*

Proof. Carrier admission is projection from the tuple in Definition 1521.1. The directed ledger, net-value window, entourage row, Cauchy tail row, completion boundary row, readback rows, transports, continuations, provenance, and local name are visible coordinates of one finite BHist packet.

Directed-tail coverage is read from D together with the entourage row U. A convergence request can use only the inspected entourages whose tail witnesses have been displayed in the directed ledger.

Net-value exposure is then forced through V. The continuation row K replays every accepted request from the directed window to the displayed net values before it reaches either the Cauchy tail row or the completion boundary row.

Cauchy compatibility follows because C is carried in the same packet and is indexed by the same displayed directed windows. The transport row H preserves those coordinates componentwise and does not add a cofinal choice function or a quotient relation on nets.

The boundary row B is the only route to $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$. Consequently the downstream readback rows R and L expose only boundary data already certified by the directed-window, entourage, and Cauchy-tail rows. If a request lacks one of these displayed witnesses, no coordinate of M can export it, giving ledger non-escape. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket namecert obligations`

Theorem 1521.3 (Moore–Smith convergence uniform handoff)**.** *Let $M = (D, V, U, C, B, R, L, H, K, P, N)$ be an accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{UniformSpace}^{\uparrow}$ *consumer of M reads convergence evidence through the finite route*

$$D \longmapsto U \longmapsto V \longmapsto C \longmapsto B.$$

Thus the uniform handoff exposes displayed entourages, directed tail witnesses, net values, Cauchy tail compatibility, and the completion boundary row, but it does not export a selected limit point, a quotient net, an ambient completeness theorem, or a cofinal choice operator.

Proof. Start with a uniform-facing consumer. The only entourage coordinate in the carrier is U, and U is attached to the finite directed ledger D.

The continuation row K sends each displayed entourage request through the net-value window V. The same replay then reaches the $\mathsf{CauchyNet}^{\uparrow}$ tail row C, so the uniform convergence read cannot separate from the finite directed-tail evidence.

The boundary row B occurs after those rows in the displayed packet. The package row P and local name row N record the same route, while H transports the listed coordinates componentwise.

Since no coordinate names a selected limit point, quotient net, ambient completeness theorem, or cofinal choice operator, the exported uniform evidence is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket uniform handoff`

Theorem 1521.4 (Moore–Smith convergence completion boundary)**.** *For an accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier* $M = (D, V, U, C, B, R, L, H, K, P, N)$*, every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *read exported by* B *is downstream of the directed index-window ledger, the net-value window, the entourage row, and the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row. The subsequent* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readbacks* R *and* L *therefore expose only finite boundary data certified by the same Moore–Smith convergence packet.*

Proof. Project the boundary-facing coordinates of Definition 1521.1. The boundary row B is included with the same directed ledger D, net-value window V, entourage row U, and Cauchy tail row C.

The continuation row K records the order in which boundary reads are replayed. A completion-boundary consumer therefore reaches B only after the directed window, value, entourage, and Cauchy-tail evidence has been displayed.

The readback rows R and L are later than B. They expose regular rational and real-facing coordinates read from the displayed boundary row, not from an independent completion object.

The carrier omits selected completion points, quotient nets, global completeness, and host-level equality of boundary points. Hence the completion boundary exported by M is exactly the finite Moore–Smith convergence boundary named by the same packet. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket completion boundary`

Theorem 1521.5 (Moore–Smith convergence boundary scope)**.** *For every accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *packet* $M = (D, V, U, C, B, R, L, H, K, P, N)$*, a route reaches* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *only after the directed index-window ledger* D*, the net-value window* V*, the entourage row* U*, and the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row* C *have been displayed. The* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readbacks* R, L *are downstream of the same boundary row* B*, so no completion-facing consumer can bypass the directed-net boundary ledger.*

Proof. Project the carrier of Definition 1521.1. The only completion-facing coordinate is B, and it is carried with the directed ledger D, net-value window V, uniform entourage row U, and Cauchy-tail row C.

Apply Definition 1521.3 to force uniform-facing reads through the directed window, entourage, value, and tail rows. Then apply Definition 1521.4 to place every $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ read at B and every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback after B.

The remaining rows H, K, P, N only transport, replay, source, and name this displayed boundary route. Because the carrier has no coordinate for quotient nets, selected limits, global completeness, or cofinal choice, ledger nonescape records those requests outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket boundary scope`

Theorem 1521.6 (Moore–Smith convergence directed-window totality)**.** *For every accepted convergence read of a* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier* $M = (D, V, U, C, B, R, L, H, K, P, N)$*, the read must first expose the directed index-window ledger* D*, the net-value window* V*, the entourage row* U*, the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row* C*, and the boundary row* B *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback is exported. The exposed route is total for the accepted directed-window packet, and it exports no quotient net, selected limit, cofinal choice, host equality of completion points, or ambient completeness.*

Proof. Project the carrier in Definition 1521.1. The convergence-facing coordinates available before readback are exactly the directed ledger D, value window V, uniform entourage row U, Cauchy-tail row C, and boundary row B.

Apply the NameCert obligations of Definition 1521.2. They force carrier admission, directed-tail coverage, net-value exposure, Cauchy compatibility, completion-boundary exposure, and ledger non-escape through the same displayed rows.

Next use the uniform handoff of Definition 1521.3. Every uniform-facing request reaches the entourage and value data through D, U, V and then passes through the Cauchy tail row C before boundary data can be consumed.

Apply the completion boundary theorem Definition 1521.4 and the boundary scope theorem Definition 1521.5. These place every $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ read at B and keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readbacks downstream of that boundary row.

The carrier has no coordinate for quotient nets, selected limits, cofinal choice, host equality, or ambient completeness. Thus the directed-window route D, V, U, C, B is total for accepted convergence reads before R or L can be exported. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket directed window totality`

Theorem 1521.7 (Moore–Smith convergence obligation exhaustion)**.** *For every accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier* $M = (D, V, U, C, B, R, L, H, K, P, N)$*, every consumer-visible convergence read is exhausted by the displayed rows*

$$D, V, U, C, B, R, L, H, K, P, N.$$

The rows are, respectively, the directed index window, value window, uniform entourage, $\mathsf{CauchyNet}^{\uparrow}$ *tail, completion boundary,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *readback, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$*. Quotient nets, selected limits, cofinal choice, and ambient completeness have no coordinate in the accepted packet.*

Proof. Project the carrier of Definition 1521.1. The tuple exposes exactly the directed index window D, value window V, uniform entourage row U, Cauchy tail row C, completion boundary row B, readback rows R and L, transport row H, replay row K, provenance row P, and local name row N.

Apply the NameCert obligations of Definition 1521.2. They account for carrier admission, directed-tail coverage, net-value exposure, Cauchy compatibility, completion-boundary exposure, and ledger non-escape using only those projected rows.

Next apply the uniform handoff of Definition 1521.3. It forces every uniform-facing convergence read through D, U, V, C, and B before any boundary information is available.

Finally apply the completion boundary theorem Definition 1521.4. It places the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readbacks downstream of B and prevents them from reading an independent completion object.

The excluded data have no projected coordinate. There is no row for a quotient net, selected limit, cofinal choice operator, or ambient completeness theorem, so ledger non-escape records such requests outside the packet rather than exporting them as convergence evidence. □

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket obligation exhaustion`

Theorem 1521.8 (Moore–Smith convergence directed-window stability)**.** *Let $M = (D, V, U, C, B, R, L, H, K, P, N$ and $M' = (D', V', U', C', B', R', L', H', K', P', N')$ be accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carriers. If D, V, U, C, B and D', V', U', C', B' are matched componentwise by displayed* hsame *rows, then every directed-window convergence read admitted by M transports to an admitted directed-window convergence read of M'. The transport preserves the inspected directed windows, net-value exposure, entourage row, Cauchy-tail row, and completion-boundary row, and it introduces no quotient-net coordinate or cofinal choice row.*

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket directed window stability`

Proof. Project the two accepted carriers. The only rows used by a directed-window convergence read are D, V, U, C, B together with the transport and replay rows H, K.

The displayed hsame witnesses match each of D, V, U, C, B with its primed counterpart. Apply the componentwise transport row to move the directed index window first, then the net values, the entourage row, the Cauchy-tail row, and finally the completion-boundary row.

The replay row K' records the same accepted route in the primed packet after the transport has reached B'. Since neither carrier has a coordinate for quotient nets or cofinal choice, the transported read remains a finite directed-window read and cannot escape into an ambient net construction. □

Theorem 1521.9 (Moore–Smith convergence completion-readback exactness)**.** *For every accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier $M = (D, V, U, C, B, R, L, H, K, P, N)$, any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback exported by R or L is downstream of the displayed completion-boundary row B, and B is downstream of the directed window, net-value, entourage, and Cauchy-tail rows D, V, U, C. Thus completion-facing readback is exact for the finite packet: it cannot read a selected limit, a quotient of nets, an ambient completeness theorem, or host equality of completion points.*

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket completion readback exactness`

Proof. Start with a readback request for R or L. The carrier places both rows after the completion-boundary coordinate B.

By Definition 1521.4, every boundary read reaches B only after the directed index window, value window, entourage row, and Cauchy-tail evidence have been displayed. Hence the readback request is already routed through D, V, U, C, B before any regular rational or real-facing coordinate is exposed.

The tuple has no row for a selected limit, quotient-net object, ambient completeness theorem, or host equality relation on completion points. Therefore R and L read back exactly the finite boundary data certified by the packet and nothing outside it. □

Theorem 1521.10 (Moore–Smith convergence ledger refusal)**.** *Let $M = (D, V, U, C, B, R, L, H, K, P, N)$ be an accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier. A consumer request that asks for a quotient net, a selected Moore–Smith limit, a cofinal choice function, ambient completeness, or host equality of completion points is not exported by the local* $\mathsf{NameCert}_{\mathsf{MooreSmithConvergence}^{\uparrow}}$ *row. It is recorded as outside the finite ledger because none of those objects is a row of M.*

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket ledger refusal`

Proof. Inspect the carrier rows in Definition 1521.1. They are exactly the directed window, net-value window, entourage row, Cauchy-tail row, completion-boundary row, two readback rows, transport, replay, provenance, and local name.

Apply the ledger non-escape clause of Definition 1521.2. A request without a displayed tail witness or boundary row is not exported as convergence data.

The named excluded objects have no coordinate in M. Consequently the local name row cannot certify them, and the packet refuses the request rather than treating it as Moore–Smith convergence evidence. □

Theorem 1521.11 (Moore–Smith convergence Cauchy-completion readback totality)**.** *For every accepted* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *carrier $M = (D, V, U, C, B, R, L, H, K, P, N)$, every* $\mathsf{CauchyNet}^{\uparrow}$ *completion-facing read exported by M traverses*

$$D, V, U, C, B$$

before any $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback R, L is visible. Consequently the total completion-readback surface of M is exactly the finite route*

$$D, V, U, C, B, R, L, H, K, P, N,$$

and it exports no quotient net, selected limit, cofinal choice, host equality of completion points, or global completeness route.

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket cauchy completion readback totality`

Proof. Project the carrier of Definition 1521.1. The directed ledger D, net-value window V, entourage row U, $\mathsf{CauchyNet}^{\uparrow}$ tail row C, and completion-boundary row B are the only rows that can feed a completion-facing read.

Apply the uniform handoff of Definition 1521.3, the completion boundary theorem Definition 1521.4, and the boundary scope theorem Definition 1521.5. These place every completion-facing route through D, V, U, C, B before any regular rational or real-facing readback can be consumed.

Now apply completion-readback exactness from Definition 1521.9. The readback rows R, L expose only data downstream of B, while H, K, P, N transport, replay, source, and name the same finite packet route.

Finally use ledger refusal from Definition 1521.10. Since quotient nets, selected limits, cofinal choice, host equality of completion points, and global completeness have no carrier coordinate, the displayed route is total for exported Cauchy-completion readback and nothing outside it is exported. □

Theorem 1521.12 (Moore–Smith directed-window obligation)**.** *Every accepted Moore–Smith convergence read exposes the directed-set row D, tail-window schedule V, net-value row U, Cauchy compatibility row C, completion-boundary row B, transport row H, replay row K, provenance row P, and local* NameCert *row N before any completion consumer sees a limit candidate. The read exports no cofinal choice, quotient nets, selected limits, host equality of completion points, or global completeness principle.*

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket directed window obligation`

Proof. Begin with the carrier of Definition 1521.1. Its convergence-facing route displays exactly the directed-set, tail-window, net-value, Cauchy compatibility, completion-boundary, readback, transport, replay, provenance, and local naming rows.

Use the uniform handoff of Definition 1521.3 and the completion boundary of Definition 1521.4. These force the directed window, net-value row, entourage or Cauchy compatibility evidence, and completion-boundary row to appear before any completion-facing readback.

The boundary scope of Definition 1521.5 and the obligation exhaustion theorem Definition 1521.7 identify the permitted rows as the displayed packet rows. The directed-window stability theorem Definition 1521.8 then shows that transport preserves the same finite route through H and K.

Finally apply completion readback exactness from Definition 1521.9 and ledger refusal from Definition 1521.10. Cofinal choice, quotient nets, selected limits, host equality of completion points, and global completeness have no carrier coordinate, so they cannot be exposed before or after the completion consumer sees the finite boundary row. □

Theorem 1521.13 (Moore–Smith convergence obligation-closure package)**.** *The carrier of Definition 1521.1, the directed-window stability of Definition 1521.8, the uniform handoff of Definition 1521.3, the completion boundary of Definition 1521.4, and the ledger refusal of Definition 1521.10 jointly form the complete paper-side obligation package for promoting the* MooreSmithConvergence$^{\uparrow}$ *surface from seed closure toward obligation closure. The package may cite the sibling* CauchyNet$^{\uparrow}$*,* UniformCompletion$^{\uparrow}$*, and* MetricCompletion$^{\uparrow}$ *chapters only through their displayed finite boundary rows, and it produces no selected limit and no quotient net.*

Lean4 ✓ `BEDC.Derived.MooreSmithConvergenceUp.MooreSmithConvergencePacket obligation closure package`

Proof. Project the carrier. The displayed packet already contains the directed index-window ledger, net-value window, uniform entourage row, Cauchy-net tail row, completion-boundary row, rational and real readbacks, transport, replay, provenance, and local naming rows.

Use directed-window stability to transport admitted reads between accepted packets without adding cofinal choice. Use the uniform handoff to force every uniform-facing read through the finite directed window, entourage, value, Cauchy-tail, and boundary rows.

Next use the completion-boundary theorem. It places every UniformCompletion$^{\uparrow}$ or MetricCompletion$^{\uparrow}$ consumer downstream of the same CauchyNet$^{\uparrow}$-compatible directed evidence, so completion-facing access remains a finite boundary read rather than a quotient construction.

Finally apply ledger refusal. Since selected limits, quotient nets, cofinal choice, ambient completeness, and host equality of completion points have no row in the carrier, the listed theorems exhaust the paper-side obligation package and no stronger completion object is produced. □

Closure status for NameCert$_{\text{MooreSmithConvergence}^{\uparrow}}$**.**

Constructive story: A MooreSmithConvergence$^{\uparrow}$ packet is generated from a finite directed index-window ledger, a net-value window, a UniformSpace$^{\uparrow}$ entourage row, a CauchyNet$^{\uparrow}$ tail row, a UniformCompletion$^{\uparrow}$ or MetricCompletion$^{\uparrow}$

boundary row, RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ readbacks, componentwise hsame transports, displayed Cont routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MooreSmithConvergence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1521.1, Definition 1521.2, Definition 1521.5, Definition 1521.6, Definition 1521.9, Definition 1521.10, and Definition 1521.13 close the finite directed-window route $D, V, U, C, B, R, L, H, K, P, N$ from CauchyNet$^{\uparrow}$ and UniformSpace$^{\uparrow}$ evidence into UniformCompletion$^{\uparrow}$ and MetricCompletion$^{\uparrow}$ boundary consumers.

Formal verification: FormalStatus(NameCert$_{\text{MooreSmithConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient nets, selected limits, global completeness, cofinal choice, host equality of completion points, Lean verification, or bridge checking.

Upgrade path: The paper theory axis is mature for the directed-window bridge; Lean verification and bridge checking remain separate gates.

Chapter 1522

A Concrete Naming Certificate for $\mathsf{CauchySequenceBounded}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySequenceBounded}^{\uparrow}$ packet records the boundedness surface that a Cauchy sequence exposes before any completed-real consumer is allowed to read it. It sits between the $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126, the $\mathsf{CauchyModulus}^{\uparrow}$ threshold rows of chapter 292, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the convergence boundary of chapter 1509. The classifier-facing boundedness handoff also consumes the Cauchy equivalence setoid route of chapter 1820 when a downstream consumer compares two bounded schedules by regular-rational closeness. The certificate names a finite bound packet for one displayed Cauchy schedule; it does not turn Cauchy data into a host-complete object, quotient stream, or selected limit.

Definition 1522.1 (Cauchy sequence bounded carrier)**.** *A* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, M, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule exposing the finite window of sequence reads.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row that says which tail requests are governed by the displayed Cauchy modulus.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the window reads and tail tolerances in dyadic or rational form.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that packages the accepted regular rational readback induced by S, M, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream schedule, modulus row, dyadic tolerance ledger, regular-rational handoff, and bound ledger have been displayed.*
6. *B is the explicit dyadic or rational bound ledger, consisting of a finite initial-window bound, a tail-bound row extracted from M and D, and a reconciliation row saying that both windows are contained in the same displayed bound.*
7. *H stores componentwise* hsame *transport for the stream schedule, modulus row, dyadic ledger, regular-rational handoff, real seal, bound ledger, continuation replay, provenance, and local naming row.*

8. *C stores displayed* Cont *replay from stream requests to modulus thresholds, from thresholds to dyadic tolerances, from tolerances to the bound ledger, and from the bound ledger to the regular-rational and real-facing rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceBounded}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, D, R, E, B, H, C, P, N$. It has no coordinate for host completeness, quotient streams, selected limits, real trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedTasteGate single carrier alignment`

Theorem 1522.2 (Cauchy boundedness NameCert obligations)**.** *For an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier $X = (S, M, D, R, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceBounded}^{\uparrow}}$ *obligations are exactly carrier admission, finite-window classification, modulus-to-bound extraction, dyadic bound ledger stability,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and ledger non-escape. These obligations are discharged by the displayed route*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E \longmapsto B$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier namecert obligations`

Proof. Carrier admission is the projection of the displayed tuple in Definition 1522.1. The finite-window classifier reads the schedule S and then uses M to identify the tail threshold where the Cauchy comparison becomes governed by the displayed modulus row.

The modulus-to-bound extraction applies the tolerance ledger D to the inspected initial window and to the tail window named by M. The ledger B records both finite data: an initial-window bound and a tail-bound row, reconciled into one dyadic or rational bound.

The rows H and C transport and replay the same route componentwise, so dyadic bound stability is a property of the displayed ledger rather than of an external order principle. The rows R and E are downstream handoff rows; they receive a bounded sequence packet after B is present.

Finally, P and N source and name the packet. Since the carrier has no coordinate for host completeness, quotient streams, selected limits, trichotomy, open-cover compactness, choice, or ambient real completeness, the listed obligations exhaust the NameCert surface. □

Theorem 1522.3 (Cauchy window bound handoff)**.** *Every accepted boundedness read from a* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packet factors through*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E \longmapsto B$$

before any $\mathsf{Real}^{\uparrow}$ *consumer or convergence consumer reads the sequence. The exported handoff is the finite tuple $S, M, D, R, E, B, H, C, P, N$, with no hidden host completeness, quotient-stream equality, selected limit, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier window bound handoff`

Proof. Start with a consumer that asks for boundedness. By Definition 1522.1, the only visible observation schedule is S, and the only tail control row is M.

The dyadic tolerance ledger D is read next, because it is the sole row that compares finite observations and tail tolerances in the packet. The regular-rational handoff R and real seal row E are downstream of this tolerance data.

Apply Definition 1522.2. The ledger B is the row where the initial-window bound and the tail bound are reconciled, so every real-facing or convergence-facing read must pass through it before consuming R or E as a public boundary.

The structural rows H, C, P, N preserve, replay, source, and name the same route. No row in the carrier can supply the excluded completeness, quotient, selection, trichotomy, compactness, or ambient-real principle. □

Theorem 1522.4 (Cauchy boundedness modulus tail bound)**.** *For an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, every tail-bound read exposed by the boundedness ledger* B *must first read the stream schedule* S*, the Cauchy modulus threshold row* M*, and the dyadic tolerance ledger* D*. The tail-bound component of* B *is therefore a finite consequence of* S, M, D*, not an ambient completeness claim, quotient stream equality, selected limit, real trichotomy, open-cover compactness, or countable-choice principle.*

Proof. Project the rows S, M, D, B from Definition 1522.1. The stream schedule fixes the finite inspected windows, while the modulus row fixes the tail threshold at which the displayed Cauchy comparison is allowed to govern later reads.

Use Definition 1522.2. The dyadic ledger D is the only coordinate that compares inspected values with the tail tolerance, and the bound ledger B records the initial-window and tail-bound data only after that comparison route is visible.

Thus the tail-bound read is generated inside the displayed finite packet. No excluded completeness, quotient, selection, trichotomy, compactness, or choice coordinate can supply a second tail-bound source. □

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier modulus tail bound`

Theorem 1522.5 (Cauchy boundedness dyadic stability)**.** *For an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, any componentwise* hsame *transport that preserves the displayed rows* $S, M, D, R, E, B, H, C, P, N$ *also preserves the dyadic or rational bound recorded in* B*. In particular, a transported consumer cannot replace* B *by an unrecorded order bound or by a bound obtained from host* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The bound ledger is a carrier coordinate. A transport of the packet must carry $S, M, D, R, E, B, H, C, P, N$ componentwise, so the row named B is transported together with the schedule, modulus, dyadic ledger, handoff, and Real seal rows that generate it.

Apply Definition 1522.3. Every boundedness read consumes B before a real-facing or convergence-facing row is available, so a transported consumer has no opportunity to substitute a different bound after the handoff.

The replay row C and provenance row P name the same route, and the local row N names the same packet. Therefore dyadic bound stability is exactly stability of the displayed ledger B under the displayed transports. □

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier dyadic bound stability`

Theorem 1522.6 (Cauchy boundedness Real seal handoff)**.** *For an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *may be consumed by a downstream real-facing packet only after the route*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E \longmapsto B, H, C, P, N$$

has displayed the bound ledger and structural rows. The seal exports the bounded source route, not a selected completed-real limit, quotient stream representative, trichotomy decision, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier real seal handoff`

Proof. Start with the carrier. The Real-facing row E is downstream of the stream schedule, modulus threshold, dyadic tolerance ledger, and regular-rational handoff rows S, M, D, R.

Use Definition 1522.2 to expose the boundedness ledger B. The obligations make B the row where the initial-window bound and tail-bound data are reconciled before a real-facing consumer may use the source.

Finally apply Definition 1522.3. Any downstream $\mathsf{Real}^{\uparrow}$ read must carry H, C, P, N with the same route, so the seal is a handoff for the displayed bounded source and nothing outside the packet. □

Theorem 1522.7 (Cauchy boundedness ledger nonescape)**.** *For an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, no boundedness, Real-seal, or convergence-facing consumer can bypass the rows* $S, M, D, R, E, B, H, C, P, N$*. The packet has no coordinate that names host boundedness, quotient stream equality, selected limits, real trichotomy, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier ledger nonescape`

Proof. Read the carrier surface. Its only displayed rows are the stream schedule S, modulus row M, dyadic ledger D, regular-rational handoff R, Real seal E, bound ledger B, transport H, replay C, provenance P, and local name N.

The two earlier handoff theorems force all boundedness and real-facing reads through those same rows. A consumer that omits any of them lacks the schedule, threshold, tolerance, handoff, seal, bound, transport, replay, provenance, or local name needed to identify the packet.

Since no additional carrier coordinate exists, none of the excluded host principles can be recovered from the packet. Ledger nonescape is therefore the direct exhaustion of the displayed rows. □

Theorem 1522.8 (CauchySequenceBounded scoped dependency surface)**.** *The* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packet closes its dependency scope over exactly* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. For an accepted carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, every boundedness, tail, Real-seal, or convergence-facing read factors through the displayed rows*

$$S, M, D, R, E, B, H, C, P, N$$

and no read may import host boundedness, quotient stream equality, selected limits, real trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier scoped dependency surface`

Proof. Project the carrier of Definition 1522.1. The rows S, M, D, R, E, B are respectively the stream schedule, Cauchy modulus threshold, dyadic tolerance ledger, regular rational handoff, Real seal, and explicit bound ledger.

Apply Definition 1522.2 and Definition 1522.3. These rows force every boundedness read through the same finite schedule, threshold, dyadic, regular-readback, seal, and bound coordinates.

Use Definition 1522.5, Definition 1522.6, and Definition 1522.7. The structural rows H, C, P, N preserve, replay, source, and name the same dependency surface, so the listed excluded objects remain outside the packet. □

Theorem 1522.9 (Cauchy boundedness bridge surface)**.** *The public* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packet bridges to the standard bounded-sequence reading exactly through its displayed modulus tail bounds, dyadic stability,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and ledger nonescape rows. For an accepted carrier* $X = (S, M, D, R, E, B, H, C, P, N)$*, a standard bounded-sequence consumer may read the finite initial-window bound and the tail bound only through*

$$S \longmapsto M \longmapsto D \longmapsto B \longmapsto R \longmapsto E$$

together with H, C, P, N*. The bridge exports a boundedness surface for the displayed Cauchy schedule; it does not select a limit, prove host sequence convergence, identify quotient stream representatives, or import ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier bridge surface`

Proof. Start with the public carrier and read the modulus tail rows. The stream schedule S fixes the finite requests, the modulus row M fixes the tail threshold, and the dyadic ledger D supplies the only tolerance comparisons available to the packet.

Apply Definition 1522.4 and Definition 1522.5. These theorems make the ledger B the stable source of the displayed initial-window and tail bounds, with no unrecorded order bound available after transport.

Next apply Definition 1522.6. The $\mathsf{Real}^{\uparrow}$ seal row E is a handoff for the bounded source route, and the regular-rational row R is consumed only after the bound ledger has been displayed.

Finally use Definition 1522.7. Since every read must remain inside $S, M, D, R, E, B, H, C, P, N$, the bridge can expose standard boundedness for the displayed schedule without selecting limits, proving host convergence, quotienting streams, or importing ambient completeness. □

Falsifiable prediction: Every $\mathsf{Real}^{\uparrow}$ or Cauchy-convergence read from a $\mathsf{CauchySequenceBounded}^{\uparrow}$ packet factors through the displayed stream schedule, Cauchy modulus threshold, dyadic tolerance ledger, regular-rational handoff, real seal row, and explicit bound ledger before the sequence is consumed.

Independence witness: streamname, cauchymodulus, dyadicratcore, regseqrat, real, cauchyconvergencecriterion: $\mathsf{CauchySequenceBounded}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a stream schedule, modulus threshold row, dyadic tolerance ledger, regular-rational handoff, real seal, explicit bound ledger, componentwise transport, continuation replay, provenance, and local naming.

Theorem 1522.10 (Cauchy boundedness mature consumer grid)**.** *Every completion-style consumer of an accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *carrier* $X = (S, M, D, R, E, B, H, C, P, N)$ *reads*

the modulus-tail row, dyadic-stability row, $\mathsf{Real}^{\uparrow}$ *seal handoff, and ledger-nonescape bridge surface through the same finite grid*

$$S, M, D, R, E, B, H, C, P, N.$$

The grid supports $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readers that need bounded Cauchy source data, but it does not assert ambient* $\mathsf{Real}^{\uparrow}$ *completeness or select a completed-real limit.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier mature consumer grid`

Proof. Project the displayed carrier. The stream schedule, modulus row, dyadic ledger, regular-rational handoff, Real seal, bound ledger, transport, replay, provenance, and local name are the only rows available to the consumer.

Apply Definition 1522.4 and Definition 1522.5. They make the tail bound and its dyadic stability properties reads of S, M, D, B, H, C, P, N, not of an external order or completeness principle.

Apply Definition 1522.6 and Definition 1522.7. These keep $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers on the displayed handoff route and prevent any bypass through quotient streams, selected limits, trichotomy, compactness, choice, or ambient completeness. □

Theorem 1522.11 (Cauchy boundedness completion consumer example)**.** *A completion-style reader that consumes a* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packet may use the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *and* $\mathsf{Real}^{\uparrow}$ *seal* E *only as bounded source data certified by* S, M, D, B, H, C, P, N*. In particular, the reader may transport a bounded regular-rational source into a later completion argument, but it must supply any completeness theorem, limit selector, quotient-real equality, or convergence conclusion from outside this chapter.*

Lean4 ✓ `BEDC.Derived.CauchySequenceBoundedUp.CauchySequenceBoundedCarrier completion consumer example`

Proof. Start from the bounded source route in Definition 1522.9. The standard bounded-sequence reading consumes S, M, D, B before it can use the handoff rows R and E.

The mature consumer grid theorem identifies the same route as the complete paper-side consumer surface for completion-style readers. Thus a later argument may reuse the bounded regular-rational source and Real seal as named input data.

Nothing in the carrier supplies a completion result. The nonescape theorem excludes host completeness, quotient streams, selected limits, trichotomy, open-cover compactness, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so any downstream convergence or limit conclusion must be supplied by a separate packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceBounded}^{\uparrow}$ packet is generated from a StreamName request schedule, a CauchyModulus threshold row, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, a RealUp seal, an explicit dyadic or rational bound ledger, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequenceBounded}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1522.1, Definition 1522.2, Definition 1522.3, Definition 1522.4, Definition 1522.5, Definition 1522.6, Definition 1522.7, Definition 1522.8, Definition 1522.9, Definition 1522.10, and Definition 1522.11 close the displayed public boundedness dependency surface, its standard bounded-sequence bridge, and its completion-style consumer surface for one Cauchy sequence schedule.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequenceBounded}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host completeness, quotient streams, selected limits, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: No paper-side theory upgrade remains below the standard bridge axis; Lean verification and bridge checking remain separate obligations.

Chapter 1523

A Concrete Naming Certificate for $\mathsf{AitkenDeltaSquared}^{\uparrow}$

Following Aitken's delta-squared process in numerical analysis, the $\mathsf{AitkenDeltaSquared}^{\uparrow}$ packet records finite acceleration evidence for a displayed real-name sequence. It sits next to the sequence surface of chapter 120, the rate ledger of chapter 415, the dyadic arithmetic core of chapter 291, the stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate names one finite acceleration window and its rate handoff; it does not import host limits, ambient completeness, quotient real equality, trichotomy, or a selected convergence witness.

Definition 1523.1 (Aitken delta-squared carrier)**.** *An* $\mathsf{AitkenDeltaSquared}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, W, O, F, G, Q, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Seq}^{\uparrow}$ *source sequence row whose finite real-name observations are to be accelerated.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting three consecutive displayed observations from S.*
3. *O stores the three observations x_n, x_{n+1}, x_{n+2} as finite* $\mathsf{Real}^{\uparrow}$*-facing names before any acceleration read is accepted.*
4. *F stores the first-difference rows $x_{n+1} - x_n$ and $x_{n+2} - x_{n+1}$ in dyadic or regular rational readback form.*
5. *G stores the second-difference row $x_{n+2} - 2x_{n+1} + x_n$ as a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger computation.*
6. *Q is the denominator-apartness ledger asserting that the displayed second-difference row is apart from the zero denominator at the requested finite precision.*
7. *K is the dyadic quotient estimate for the local accelerated read*

$$x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}$$

   computed only from O, F, G, Q.

8. *R is the* $\mathsf{CauchyRate}^{\uparrow}$ *handoff row recording the finite rate evidence obtained from the quotient estimate and the displayed source window.*

9. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the accelerated observation.*
10. *H stores componentwise* hsame *transport for the source sequence, selected window, three observations, first-difference rows, second-difference row, apartness ledger, quotient estimate, CauchyRate handoff, RegSeqRat readback, Real seal, continuation replay, provenance, and local naming row.*
11. *C stores displayed* Cont *replay from the sequence window to the three observations, from observations to first and second differences, from the apartness ledger to the quotient estimate, and from the quotient estimate to the CauchyRate, RegSeqRat, and Real-facing rows.*
12. *P is the* Pkg *provenance over* $\mathsf{AitkenDeltaSquared}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{AitkenDeltaSquared}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, O, F, G, Q, K, R, E, H, C, P, N$. It has no coordinate for host limits, ambient completeness, quotient real equality, trichotomy, selected convergence witnesses, or division by a hidden denominator.

Lean4 ✓ `BEDC.Derived.AitkenDeltaSquaredUp.AitkenDeltaSquaredTasteGate single carrier alignment`

Theorem 1523.2 (Aitken delta-squared NameCert obligations)**.** *For an accepted* $\mathsf{AitkenDeltaSquared}^{\uparrow}$ *carrier $A = (S, W, O, F, G, Q, K, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AitkenDeltaSquared}^{\uparrow}}$ *obligations are exactly carrier admission, three-observation window classification, first-difference extraction, second-difference ledger stability, denominator-apartness validation, dyadic quotient estimate construction,* $\mathsf{CauchyRate}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal factorization, and ledger non-escape. These obligations are discharged by the displayed route*

$$S \longmapsto W \longmapsto O \longmapsto F \longmapsto G \longmapsto Q \longmapsto K \longmapsto R \longmapsto E$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 1523.1. The tuple lists exactly the sequence row, stream window, observation rows, finite-difference rows, apartness ledger, quotient estimate, rate handoff, regular-rational readback, real seal, transport, replay, provenance, and local name.

The window classifier reads S only through the finite $\mathsf{StreamName}^{\uparrow}$ row W. The row O then supplies three consecutive displayed observations, and F and G are computed from those displayed rows by the dyadic arithmetic ledger rather than by an ambient real operation.

Denominator validity is the row Q. The quotient estimate K may be read only after Q is present, so the accelerated expression is a finite dyadic or regular-rational computation with an explicit apartness witness at the requested precision.

The rows R and E are downstream handoffs. They consume the same finite quotient route before exposing CauchyRate evidence, RegSeqRat readback, or a Real-facing seal.

Finally, H and C transport and replay the route componentwise, while P and N source and name it. Since the carrier has no field for host limits, ambient completeness, quotient real equality, trichotomy, selected convergence witnesses, or hidden division, the listed obligations exhaust the NameCert surface. □

Theorem 1523.3 (Aitken delta-squared CauchyRate handoff)**.** *Every accepted* $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read from an* $\mathsf{AitkenDeltaSquared}^{\uparrow}$ *packet factors through the finite acceleration route*

$$S \longmapsto W \longmapsto O \longmapsto F \longmapsto G \longmapsto Q \longmapsto K \longmapsto R \longmapsto E.$$

Thus Real-facing consumers receive only the displayed acceleration estimate and rate ledger, not a host limit, an ambient completeness claim, quotient real equality, trichotomy, or a selected convergence witness.

Proof. Start with any rate, regular-rational, or real-facing consumer of the accepted packet. By Definition 1523.1, the consumer has no direct access to the source sequence except through the displayed window W over S.

The three observations in O are then the only rows from which first and second differences can be read. The first-difference row F and second-difference row G therefore determine the finite acceleration denominator and numerator surface visible to the packet.

Before the quotient estimate is exported, the consumer must read the apartness ledger Q. This row is the certificate that the displayed denominator is usable at the finite precision being requested, and K is the only row that stores the resulting dyadic quotient estimate.

Apply Definition 1523.2. The CauchyRate handoff R, the RegSeqRat readback, and the Real seal in E all stand downstream of K, and their transport and replay data are exactly H and C.

No row remains that could supply host limits, ambient completeness, quotient real equality, trichotomy, or a selected convergence witness. Hence every Real-facing read passes through the finite acceleration and rate ledgers. □

Falsifiable prediction: Every $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read from an $\mathsf{AitkenDeltaSquared}^{\uparrow}$ packet factors through the displayed sequence window, three consecutive observations, first-difference rows, second-difference row, denominator-apartness ledger, dyadic quotient estimate, CauchyRate handoff, regular-rational readback, real seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: seq, streamname, dyadicratcore, cauchyrate, regseqrat, real: $\mathsf{AitkenDeltaSquared}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source sequence, a three-observation stream window, finite-difference ledgers, denominator apartness, a quotient estimate, rate handoff, regular-rational and real-seal rows, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AitkenDeltaSquared}^{\uparrow}}$.

Constructive story: An $\mathsf{AitkenDeltaSquared}^{\uparrow}$ packet is generated from a source sequence, a finite stream window with three consecutive observations, first- and second-difference ledgers, an explicit denominator-apartness row, a dyadic quotient estimate, CauchyRate and RegSeqRat handoffs, a Real seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AitkenDeltaSquared}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1523.1, Definition 1523.2, and Definition 1523.3 seed the finite delta-squared acceleration packet for one displayed real-name sequence window.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AitkenDeltaSquared}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limits, ambient completeness, quotient real equality, trichotomy, selected convergence witnesses, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, three-observation classification, first- and second-difference ledger stability, denominator-apartness validation, quotient estimate construction, CauchyRate handoff, Real-facing factorization, and ledger non-escape for BEDC.Derived.AitkenDeltaSquaredUp.

Chapter 1524

A Concrete Naming Certificate for $\mathsf{DyadicRoundingWindow}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicRoundingWindow}^{\uparrow}$ packet records the finite precision-window surface used when a real name or regular rational name is inspected at one displayed unary precision and then rounded through adjacent dyadic endpoints. It sits between the finite windows of chapter 126, the adjacent endpoint packet of chapter 433, the readback route of chapter 335, the regular rational handoff of chapter 301, the dyadic arithmetic ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names precision-window rounding as finite BEDC route data, without quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness.

Definition 1524.1 (DyadicRoundingWindow carrier)**.** *A* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, q, D, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window opened for the inspected real or regular rational name.*
2. *q is the unary precision request that fixes the finite observation scale.*
3. *D is the adjacent endpoint* $\mathsf{DyadicRounding}^{\uparrow}$ *packet at precision q, including its* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows and tolerance ledger.*
4. *A is the* $\mathsf{DyadicApproximation}^{\uparrow}$ *readback row exposing the rounded finite observation produced from S, q, D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that packages the same rounded window as regular rational data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream window, unary precision, adjacent endpoint packet, dyadic readback, and regular rational handoff are displayed.*
7. *H stores componentwise* hsame *transport for the stream window, unary precision, adjacent endpoint packet, dyadic readback, regular rational handoff, real seal, continuation replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from the precision request to the stream window, from the window to the adjacent endpoint packet, from the endpoint packet to the dyadic readback, and from the readback to the regular rational and real-facing rows.*

9. *P is the* Pkg *provenance over* DyadicRoundingWindow$^\uparrow$, StreamName$^\uparrow$, DyadicRounding$^\uparrow$, DyadicApproximation$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, DyadicRatCore$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DyadicRoundingWindow}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, q, D, A, R, E, H, C, P, N$. It has no coordinate for quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowTasteGate single carrier alignment`

Theorem 1524.2 (DyadicRoundingWindow NameCert obligations)**.** *For an accepted* DyadicRoundingWindow$^\uparrow$ *carrier $W = (S, q, D, A, R, E, H, C, P, N)$, the local* NameCert$_{\text{DyadicRoundingWindow}^\uparrow}$ *obligations are exactly carrier admission, precision-window classification, adjacent-endpoint compatibility, dyadic readback exactness,* RegSeqRat$^\uparrow$ *handoff,* Real$^\uparrow$ *seal factorization, and ledger non-escape. These obligations are discharged by the displayed route*

$$S \longmapsto q \longmapsto D \longmapsto A \longmapsto R \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1524.1. The finite stream window, unary precision, adjacent endpoint packet, dyadic readback, regular rational handoff, real seal, transport rows, continuation routes, package provenance, and local naming row are exactly the coordinates available to the classifier.

Precision-window classification first reads q and then opens S at the displayed scale. Adjacent-endpoint compatibility reads the same precision row through the DyadicRounding$^\uparrow$ packet D, whose dyadic endpoints and tolerance ledger are already tied to DyadicRatCore$^\uparrow$.

Dyadic readback exactness and the two handoff clauses are route projections. The row A exposes the DyadicApproximation$^\uparrow$ observation produced by S, q, D; the row R repacks that observation as regular rational data; and the row E is the only real-facing seal coordinate.

The non-escape obligation is packet inspection. Since no row records quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness, every accepted consumer remains inside the displayed finite route. □

Theorem 1524.3 (DyadicRoundingWindow precision-window obligation)**.** *Every accepted* DyadicRoundingWindow *packet $W = (S, q, D, A, R, E, H, C, P, N)$ must open the* StreamName$^\uparrow$ *row S at unary precision q before D is read as the* DyadicRounding$^\uparrow$ *endpoint packet. Only after this displayed read may the* DyadicApproximation$^\uparrow$ *row A, the* RegSeqRat$^\uparrow$ *handoff R, and the* Real$^\uparrow$ *seal E be consumed, and then only through the displayed* hsame, Cont, Pkg, *and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier precision window obligation`

Proof. Project the carrier of Definition 1524.1. Its ordered rows expose S and q before the endpoint packet D, so an accepted consumer has a finite stream window and a unary precision request before any rounded endpoint is inspected.

Apply Definition 1524.2. Its route order is

$$S \longmapsto q \longmapsto D \longmapsto A \longmapsto R \longmapsto E$$

together with H, C, P, N, so the precision-window obligation is a projection of the local NameCert$_{\text{DyadicRoundingWindow}^\uparrow}$ surface.

Use Definition 1524.4 for nonescape. The error bound follows the same endpoint and readback route into $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, and the structural rows merely transport, replay, source, and name that route. No quotient equality, selected limit, trichotomy, host rounding, or ambient completeness row is introduced. □

Theorem 1524.4 (DyadicRoundingWindow error bound)**.** *Every accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier exports a finite error-bound route: the readback row A is bounded by the adjacent endpoint packet D at unary precision q, transported through H, replayed through C, and handed to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through R and E. No exported bound may be replaced by quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier error bound`

Proof. Begin with the accepted carrier and project its displayed precision row q. The row q fixes the only observation scale available to the packet, so an error-bound consumer must read the same finite window S at that scale.

Next read the adjacent endpoint packet D. By the $\mathsf{DyadicRounding}^{\uparrow}$ dependency surface, D supplies the adjacent dyadic endpoint rows and tolerance ledger at the displayed precision, with those endpoint rows sourced from $\mathsf{DyadicRatCore}^{\uparrow}$.

Now read the dyadic approximation row A. It is the only coordinate that exposes the rounded observation from S, q, D, so its error bound is the endpoint-and-tolerance surface already carried by D.

Finally route the bound through R, E, H, C, P, N. These rows preserve, replay, source, and name the same finite packet for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The carrier inventory contains no coordinate that can replace the bound by quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness. □

Theorem 1524.5 (DyadicRoundingWindow error-envelope monotonicity)**.** *Let*

$$W = (S, q, D, A, R, E, H, C, P, N)$$

and

$$W' = (S', q', D', A', R', E', H', C', P', N')$$

be accepted $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *packets. Suppose the two packets share the same* $\mathsf{StreamName}^{\uparrow}$ *row, endpoint route, and readback route under the displayed* hsame *and* Cont *data, and suppose the unary precision q' refines q. Then the exported error-bound route of W' factors through the endpoint ledger of W and cannot widen the tolerance envelope before the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff. The comparison is made only through the displayed unary precisions and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Proof. Project both packets through Definition 1524.1. The only rows that can affect the error envelope are the stream rows S, S', the unary precisions q, q', the adjacent endpoint packets D, D', and the readback rows A, A'.

Apply Definition 1524.4 to each packet. For W, the exported bound is the endpoint-and-tolerance surface carried by D at precision q. For W', the bound is the corresponding surface carried by D' at precision q'.

The shared StreamName row and shared endpoint/readback route force both bounds to read the same finite observation path. Since q' refines q, the later $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger is a subledger of the earlier endpoint ledger for the same route; it supplies a narrower observation budget, not an extra coordinate.

Finally inspect the remaining rows R, E, H, C, P, N and R', E', H', C', P', N'. They transport, replay, source, and name the already compared endpoint ledgers for the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff. No quotient equality, host rounding mode, trichotomy row, selected limit, or completeness row is present to widen the envelope. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier error envelope monotonicity`

Theorem 1524.6 (DyadicRoundingWindow precision classification)**.** *Every accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier opens the same* $\mathsf{StreamName}^{\uparrow}$ *row* S *at the displayed unary precision* q *before the adjacent endpoint packet* D *is read as* $\mathsf{DyadicRounding}^{\uparrow}$ *data. The precision classifier is therefore the finite route through* $S, q, D, A, R, E, H, C, P, N$*, not a host rounding mode, selected real approximation, trichotomy test, or ambient completeness argument.*

Proof. Start with the carrier inventory of Definition 1524.1. The rows S and q are displayed before the endpoint packet, so the classifier has a named stream window and a named unary precision before any rounded endpoint is consumed.

Apply the NameCert obligations of Definition 1524.2. The precision-window classification clause reads q and then opens S at that scale as part of the route

$$S \longmapsto q \longmapsto D \longmapsto A \longmapsto R \longmapsto E.$$

Finally use the error-bound surface of Definition 1524.4. Since the bound is attached to the same q and the same D, the classifier cannot swap in a host rounding mode, selected approximation, trichotomy test, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier precision classification`

Theorem 1524.7 (DyadicRoundingWindow adjacent endpoint compatibility)**.** *In every accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier, the endpoint packet* D *and the readback row* A *share the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger at precision* q*, and their componentwise compatibility is transported only through the displayed* hsame *row* H*. Thus the adjacent endpoint readback cannot be replaced by quotient real equality, host rounding, selected limits, or an unstated metric-completeness witness.*

Proof. Project $W = (S, q, D, A, R, E, H, C, P, N)$ from Definition 1524.1. The row D contains the adjacent endpoint packet and its dyadic tolerance ledger, while A is the only dyadic readback coordinate produced from S, q, D.

The obligation theorem Definition 1524.2 ties adjacent-endpoint compatibility and dyadic readback exactness to the same route. Therefore D and A are compared inside the displayed precision-window packet rather than through an external real-equality principle.

Now apply the error-bound theorem Definition 1524.4. The endpoint tolerance ledger bounds A at the unary precision q, and H transports this componentwise compatibility through the later rows. No carrier coordinate can replace that ledger by quotient real equality, host rounding, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier adjacent endpoint compatibility`

Theorem 1524.8 (DyadicRoundingWindow readback handoff)**.** *Every accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier hands the approximation row* A *to later consumers only through the regular rational row* R *and then through the real seal row* E*. The route is*

$$A \longmapsto R \longmapsto E$$

inside the displayed packet $W = (S, q, D, A, R, E, H, C, P, N)$, so a consumer cannot bypass R by invoking quotient real equality, a selected limit, host rounding, or ambient completeness.

Proof. Project W from Definition 1524.1. The row A is the only dyadic approximation readback coordinate, while R and E are the only regular rational and real-facing rows.

Use Definition 1524.4. It states that the bound on A is preserved for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers only through R and E, with H and C transporting and replaying the same finite route.

Finally apply Definition 1524.7. The endpoint packet D and the readback row A remain tied to the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger at the displayed precision q. Since no other readback or real-facing coordinate is present, the handoff cannot escape the $A \mapsto R \mapsto E$ route. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier readback handoff`

Theorem 1524.9 (DyadicRoundingWindow Real-seal factorization)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *packet factors through $S, q, D, A, R, E, H, C, P, N$ before the rounded observation is consumed. The seal route cannot use quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness.*

Proof. Begin with Definition 1524.1. The rows R and E are downstream rows: R repackages the rounded observation as regular rational data, and E is the real-facing seal reached only after S, q, D, A are displayed.

Use Definition 1524.2. Its regular rational handoff, Real seal factorization, and ledger non-escape clauses force every consumer to read the same finite route through the stream window, unary precision, adjacent endpoints, dyadic readback, handoff, and seal.

Finally invoke Definition 1524.4. The error bound is preserved through R, E, H, C, P, N, so a consumer cannot bypass S, q, D, A by citing quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier real seal factorization`

Theorem 1524.10 (DyadicRoundingWindow ledger nonescape)**.** *In every accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *carrier, the rows H, C, P, N exhaust transport, replay, provenance, and local naming for S, q, D, A, R, E. Therefore the ledger exports no quotient real equality, selected limit, trichotomy, host rounding operation, ambient completeness theorem, or unnamed real-facing approximation row.*

Proof. Start with the carrier inventory of Definition 1524.1. Its structural rows are exactly H, C, P, N: H transports the displayed coordinates componentwise, C replays the continuation route, P records the package provenance, and N names the local packet.

Apply Definition 1524.8. The only path from the approximation row to a later consumer is through $A \mapsto R \mapsto E$, and that path is already guarded by the same H, C, P, N rows.

Use Definition 1524.9. Real-facing consumers factor through the displayed stream window, precision request, adjacent endpoint packet, readback row, regular rational handoff, and real seal before any rounded observation is consumed.

No remaining carrier coordinate can name quotient real equality, selected limits, trichotomy, host rounding, ambient completeness, or an extra approximation row. Thus the ledger is nonescaping by finite packet exhaustion. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier ledger nonescape`

Theorem 1524.11 (DyadicRoundingWindow mature completion boundary)**.** *The mature* DyadicRoundingWindow$^{\uparrow}$ *packet exhausts precision-window rounding through the displayed rows*

$$S, q, D, A, R, E, H, C, P, N$$

over StreamName$^{\uparrow}$*,* DyadicRounding$^{\uparrow}$*,* DyadicApproximation$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *data. Every rounded-observation consumer is confined to that route. No quotient real equality, selected limit, trichotomy, host rounding operation, or ambient completeness row is exported.*

Proof. Begin with the obligation package of Definition 1524.12. It identifies the carrier rows, precision-window classification, adjacent endpoint compatibility, dyadic readback, regular rational handoff, Real seal factorization, and ledger nonescape as one finite packet.

Now apply the scoped consumer and kernel-dependency boundaries, Definition 1524.13 and Definition 1524.14. They force each consumer to pass through StreamName$^{\uparrow}$, DyadicRounding$^{\uparrow}$, DyadicApproximation$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ before the rounded observation is read.

Finally read the bridge-facing handoff of Definition 1524.15. The handoff preserves the same rows H, C, P, N and adds no coordinate for quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness. Thus the completion boundary is exhausted by the displayed packet. □

Theorem 1524.12 (DyadicRoundingWindow obligation-closure package)**.** *The carrier of Definition 1524.1, together with Definition 1524.2, Definition 1524.4, Definition 1524.6, Definition 1524.7, Definition 1524.8, Definition 1524.9, and Definition 1524.10, assembles the full obligationClosure surface for* DyadicRoundingWindow$^{\uparrow}$*. The closed surface is exactly the displayed packet over* StreamName$^{\uparrow}$*,* DyadicRounding$^{\uparrow}$*,* DyadicApproximation$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. First project the carrier. The rows $S, q, D, A, R, E, H, C, P, N$ give the only finite precision-window object available to the certificate, with S naming the stream window, q fixing the unary precision request, D carrying the adjacent dyadic endpoint packet, A exposing the dyadic readback, R giving regular-rational readback, and E giving the real-facing seal.

Next apply the obligation and error-bound theorems. They identify carrier admission, precision-window classification, adjacent-endpoint compatibility, dyadic readback exactness, regular rational handoff, Real seal factorization, and ledger non-escape as obligations of the same route

$$S \longmapsto q \longmapsto D \longmapsto A \longmapsto R \longmapsto E.$$

Now read the remaining rows separately. Precision classification fixes the observation scale before endpoint data are consumed; adjacent endpoint compatibility ties D and A to the same DyadicRatCore$^{\uparrow}$ tolerance ledger; readback handoff forces A through R before E; and Real-seal factorization prevents a later consumer from bypassing those rows.

Finally use ledger nonescape. The structural rows H, C, P, N transport, replay, source, and name the same packet, and no coordinate records quotient real equality, selected limits, trichotomy, host rounding, ambient completeness, or an unnamed approximation row. This exhausts the obligationClosure surface over the listed BEDC rows. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier obligation closure package`

Theorem 1524.13 (DyadicRoundingWindow scoped consumer boundary)**.** *Every consumer of an accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *packet must scope the rounded observation through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRounding}^{\uparrow}$ *adjacent endpoint packet,* $\mathsf{DyadicApproximation}^{\uparrow}$ *readback row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. No consumer may replace that packet by quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness.*

Proof. Start with the carrier of Definition 1524.1. Its visible rows $S, q, D, A, R, E, H, C, P, N$ are the only coordinates available to a rounded-observation consumer, so the consumer first receives a stream window, a unary precision request, adjacent dyadic endpoints, a readback row, regular-rational handoff, a real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 1524.12. The obligation package assembles those rows together with precision classification, adjacent endpoint compatibility, readback handoff, Real-seal factorization, and ledger nonescape. Thus a consumer cannot separate the rounded observation from the displayed finite route.

The handoff theorems identify the downstream boundary. By Definition 1524.8, the readback row passes only through $A \mapsto R \mapsto E$; by Definition 1524.9, any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ use factors through the same stream, endpoint, readback, handoff, and seal rows; and by Definition 1524.10, H, C, P, N exhaust the structural ledger.

Since the carrier contains no coordinate for quotient real equality, selected limits, trichotomy, host rounding, or ambient completeness, no scoped consumer can introduce one. The rounded observation remains a finite BEDC packet over the listed rows. □

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier scoped consumer boundary`

Theorem 1524.14 (DyadicRoundingWindow scoped kernel dependencies)**.** *Every precision-window rounding consumer of an accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *packet factors through the displayed* BHist *request rows,* $\mathsf{StreamName}^{\uparrow}$ *window rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic,* $\mathsf{DyadicRounding}^{\uparrow}$ *adjacent endpoint rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. These rows are the scoped kernel dependencies for the obligationClosure-to-scopedClosure step; no quotient real equality, selected limit, trichotomy, host rounding mode, or ambient completeness row is part of the factorization.*

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier scoped kernel dependencies`

Proof. Start with the precision-window obligation, Definition 1524.3. It places the request row and the $\mathsf{StreamName}^{\uparrow}$ window in front of the adjacent endpoint packet.

Apply the error-bound theorem Definition 1524.4 and endpoint compatibility Definition 1524.7. Together they force the rounded observation to read the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance arithmetic and $\mathsf{DyadicRounding}^{\uparrow}$ adjacent endpoint rows before any readback is exported.

Next use the readback handoff and Real-seal factorization, Definition 1524.8 and Definition 1524.9. They route the approximation through $\mathsf{RegSeqRat}^{\uparrow}$ and then through the terminal $\mathsf{Real}^{\uparrow}$ seal, with no bypass around the displayed finite packet.

Finally combine ledger nonescape, Definition 1524.10, with the scoped consumer boundary of Definition 1524.13. The structural rows H, C, P, N supply transport, replay, provenance, and local naming for the same route, so the listed dependencies exhaust the scoped kernel surface. □

Theorem 1524.15 (DyadicRoundingWindow bridge-facing handoff)**.** *Every standard-facing read of an accepted* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *packet preserves the dyadic source row, the precision-window row, the tolerance budget, componentwise transport, displayed replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicRoundingWindow}^{\uparrow}}$ *rows before any rounded observation is exposed. The bridge-facing handoff is exactly the displayed route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRounding}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *already scoped by Definition 1524.14. It does not assert* $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, selected limits, trichotomy, host rounding, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier bridge facing handoff`

Proof. Start with Definition 1524.14. It fixes the finite dependency route through the stream window, regular rational readback, dyadic tolerance arithmetic, adjacent endpoint rows, transport, replay, provenance, and the terminal real seal.

Apply Definition 1524.13. A consumer cannot read the rounded observation without preserving $S, q, D, A, R, E, H, C, P, N$, so the standard-facing route keeps the dyadic source row, window row, tolerance budget, and local naming together.

Use ledger nonescape from Definition 1524.10. The structural rows only preserve and name the displayed finite route. Since no carrier coordinate stores $\mathsf{Real}^{\uparrow}$ completeness, quotient equality, selected limits, trichotomy, host rounding, or ambient completion, the bridge-facing handoff cannot export those claims. □

Falsifiable prediction: Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read from a $\mathsf{DyadicRoundingWindow}^{\uparrow}$ packet factors through the displayed stream window, unary precision, adjacent $\mathsf{DyadicRounding}^{\uparrow}$ endpoint packet, $\mathsf{DyadicApproximation}^{\uparrow}$ readback, regular rational handoff, real seal row, componentwise transport, continuation replay, provenance, and local naming before the rounded observation is consumed.

Independence witness: streamname, dyadicrounding, dyadicapproximation, regseqrat, real, dyadicratcore: $\mathsf{DyadicRoundingWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a stream window, unary precision request, adjacent endpoint packet, dyadic readback, regular rational handoff, real seal, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRoundingWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRoundingWindow}^{\uparrow}$ packet is generated from a StreamName window, a unary precision request, an adjacent DyadicRounding endpoint packet, a DyadicApproximation readback row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicRoundingWindow}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1524.12, Definition 1524.11, Definition 1524.14, and Definition 1524.15 bind every public and standard-facing consumer to the displayed StreamName, DyadicRounding, DyadicApproximation, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicRoundingWindow}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.DyadicRoundingWindowUp.DyadicRoundingWindowCarrier\_bridge\_facing\_handoff`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, host rounding, ambient completeness, or bridge checking.

Upgrade path: No theory-axis upgrade remains; verification upgrade requires bridge checking for the mature precision-window boundary and bridge-facing handoff.

Chapter 1525

A Concrete Naming Certificate for $\mathsf{FanFunctional}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bishop constructive analysis, the $\mathsf{FanFunctional}^{\uparrow}$ packet records the finite modulus-extraction surface that a continuous functional on Cantor space may expose after a finite fan argument. It consumes the finite-prefix carrier of chapter 1475, the decidable finite-bar and depth rows of chapter 400, the tolerance-to-precision ledger of chapter 391, and the compact metric finite-net surface of chapter 129. The packet exports an explicit $\mathsf{UniformModulus}^{\uparrow}$ row for Cantor-prefix consumers; it does not export a host functional, a choice-selected branch, open-cover compactness, or real completeness.

Definition 1525.1 (FanFunctional carrier surface)**.** *A* $\mathsf{FanFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$$

with the following visible rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix row, including the StreamName schedule, Boolean observation ledger, and List prefix spine;*
2. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row, including the decidable bar classifier and the unary depth ledger for the displayed binary spread;*
3. *ε is the positive tolerance request read by the downstream metric or real-valued consumer;*
4. *B is the finite branch-prefix window obtained by restricting the Cantor prefix rows of C to the depth ledger of F;*
5. *D is the finite bar-depth evidence saying that every retained branch prefix in B reaches a barred prefix by the displayed unary depth;*
6. *W is the finite witness window pairing each retained prefix with the pointwise tolerance row requested at that prefix;*
7. *M is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, read as a unary prefix length sufficient for all Cantor histories sharing the retained prefix;*
8. *H records componentwise* hsame *transport for the Cantor rows, fan rows, tolerance request, witness window, depth evidence, and extracted modulus;*
9. *K records the displayed* Cont *replay route from Cantor-prefix reads through the finite fan rows into the modulus row;*

10. *P is the* Pkg *provenance over* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite branch, a host functional $2^{\mathsf{Nat}^{\uparrow}} \to \mathsf{Real}^{\uparrow}$, *an open cover, a selected bar proof, or a hidden compactness principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalTasteGate single carrier alignment`

Theorem 1525.2 (FanFunctional NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *surface of Definition 1525.1 consists of six finite obligations: carrier admission for the displayed packet* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$*; classifier stability for the Cantor-prefix and finite-fan rows under componentwise* hsame *transport; bar-depth exactness for the unary ledger D; witness-window coverage for all retained branch prefixes in B; uniform-modulus extraction of the row M; and non-escape, requiring every public read to factor through H, K, P, N and no host functional or choice principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalNameCertObligations`

Proof. Carrier admission is projection of the displayed coordinates of Definition 1525.1. The Cantor row, fan row, tolerance request, branch-prefix window, depth evidence, witness window, and modulus row are all finite BHist data.

Classifier stability is componentwise. The Cantor rows are compared by the finite-prefix classifier of chapter 1475, while the fan rows are compared by the finite bar classifier of chapter 400. Transport by H keeps the tolerance, witness window, depth evidence, and extracted modulus attached to the same displayed packet.

Bar-depth exactness reads the unary depth ledger of the $\mathsf{FanTheorem}^{\uparrow}$ row. For each retained prefix in B, the witness D records that the displayed decidable bar has already been met within the finite depth; no infinite branch or bar-induction rule is introduced.

The witness-window and modulus clauses are obtained by replaying K. Each retained prefix carries the local tolerance row in W, and the finite fan depth bounds the largest prefix length that a downstream consumer must inspect. Repacking this bounded inspection length as M gives the $\mathsf{UniformModulus}^{\uparrow}$ row.

Non-escape follows from coordinate exhaustion. The carrier contains only $C, F, \varepsilon, B, D, W, M, H, K, P, N$, so a public read cannot produce a host functional, a choice-selected branch, open-cover compactness, or real completeness. □

Theorem 1525.3 (FanFunctional root carrier admission)**.** *Every root-obligation read of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet must admit the full finite carrier*

$$\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$$

as BHist *data before any compact-uniform consumer reads it. The rows $C, F, \varepsilon, B, D, W, M, H, K, P, N$ expose the Cantor prefix, fan bar, epsilon request, branch window, finite depth, witness window, modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *surface as one packet.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootCarrierAdmission`

Proof. Project Definition 1525.1. Carrier admission reads the displayed tuple before any downstream consumer may inspect a modulus row.

The structural rows then bind the read. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve exactly $C, F, \varepsilon, B, D, W, M, H, K, P, N$, so a consumer has no entry point outside the admitted carrier. □

Theorem 1525.4 (FanFunctional root finite bar depth)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$*, the finite bar-depth row* D *is exposed after the* $\mathsf{CantorSpace}^{\uparrow}$ *and* $\mathsf{FanTheorem}^{\uparrow}$ *rows* C, F *and before any witness-window use of* W *or modulus read of* M*. Thus every* BHist *route through* hsame*,* Cont*,* Pkg*, and* NameCert *sees finite depth before it sees witness-window coverage.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootFiniteBarDepth`

Proof. Use the NameCert obligations. Bar-depth exactness is a separate row obligation, and witness-window coverage is downstream from that exactness.

The order is therefore C, F first, then D, then W, M. Transport through H, replay through K, provenance through P, and naming through N keep this order inside the same finite packet and do not add a selected branch or infinite bar proof. □

Theorem 1525.5 (FanFunctional root uniform-modulus handoff)**.** *Every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$ *routes its* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{UniformContinuity}^{\uparrow}$ *before the row* M *is exported. The route is a finite* BHist *package through* hsame*,* Cont*,* Pkg*, and* NameCert*, not a host functional, open-cover compactness proof, selected branch, or ambient Real-completeness argument.*

Proof. First read C and F to obtain the finite Cantor-prefix and fan-bar surfaces. The compact-metric row consumes only the finite windows B, D, W, and the uniform-continuity row consumes the epsilon request ε over those same windows.

The extracted row M is then the $\mathsf{UniformModulus}^{\uparrow}$ handoff. Rows H, K, P, N transport, replay, source, and name the same finite route, so the exported modulus is exactly the Cantor, fan, compact-metric, and uniform-continuity route displayed by the carrier. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUniformModulusHandoff`

Theorem 1525.6 (FanFunctional prefix-window carrier admission)**.** *Every consumer-visible modulus read from an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier must first admit the Cantor prefix row* C*, finite fan row* F*, tolerance row* ε*, branch-window row* B*, depth row* D*, witness-window row* W*, modulus row* M*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* K*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row* N*. No* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer may read a modulus before those rows have appeared on the same finite packet.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalPrefixWindowCarrierAdmission`

Proof. Project the carrier of Definition 1525.1. Its visible coordinates are exactly $C, F, \varepsilon, B, D, W, M, H, K, P, N$, so carrier admission exposes the Cantor prefix, finite fan, tolerance, branch-window, depth, witness, and modulus rows before any consumer-facing read.

Apply Definition 1525.2. The carrier admission, classifier stability, bar-depth exactness, witness-window coverage, uniform-modulus extraction, and non-escape clauses keep those rows on the same displayed hsame, Cont, Pkg, and local naming route.

Coordinate exhaustion closes the boundary. Since no coordinate names a host functional, selected branch, open cover, choice schedule, or ambient real-completeness object, every consumer-visible modulus read is forced through the admitted prefix-window carrier. □

Theorem 1525.7 (FanFunctional finite bar-modulus obligations)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier, the finite bar-depth row* D *and witness-window coverage row* W *jointly determine the* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *through the displayed* $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *packet. The determination uses the finite fan and witness windows only; it does not select branches, evaluate a host functional, import bar induction, or appeal to open-cover compactness.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalFiniteBarModulusObligations`

Proof. Project Definition 1525.1. The row D records the finite depth at which every retained branch prefix in B reaches the displayed bar, and W records the tolerance witness attached to each retained prefix.

Use Definition 1525.2. Its bar-depth exactness and witness-window coverage clauses are the only clauses that feed the uniform-modulus extraction clause, so M is read as the bounded prefix length supported by those finite rows.

The remaining rows H, K, P, N transport, replay, source, and name the same extraction. Coordinate exhaustion excludes selected branches, host functionals, bar-induction rules, and open-cover compactness from the modulus determination. □

Theorem 1525.8 (FanFunctional uniform-modulus consumer boundary)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet receives only the extracted finite prefix-modulus route through* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, *and* $\mathsf{UniformModulus}^{\uparrow}$. *The handoff exposes* $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *and no host functional, selected branch, open-cover compactness theorem, countable-choice selector, or ambient Real-completeness principle.*

Proof. Project the carrier in Definition 1525.1. The $\mathsf{CantorSpace}^{\uparrow}$ row C and $\mathsf{FanTheorem}^{\uparrow}$ row F supply the finite prefix and fan surfaces, while ε, B, D, W, M supply the tolerance, branch-window, depth, witness, and modulus rows read by downstream consumers.

Apply Definition 1525.2. It forces the modulus extraction through the same carrier admission, classifier stability, bar-depth exactness, witness-window coverage, and non-escape clauses that govern the local NameCert surface.

Finally route the public handoff through H, K, P, N. These rows give only componentwise transport, displayed replay, provenance, and local naming, so $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{UniformContinuity}^{\uparrow}$ consumers receive the finite $\mathsf{UniformModulus}^{\uparrow}$ route and no hidden compactness, choice, host-functional, or Real-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalUniformModulusConsumerBoundary`

Theorem 1525.9 (FanFunctional uniform modulus extraction)**.** *Let* Φ *be an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier. The finite fan depth row of* F*, the Cantor prefix window* B*, and the tolerance witness rows* W *determine the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row* M*. For any two accepted* $\mathsf{CantorSpace}^{\uparrow}$ *histories whose Boolean ledgers agree through the prefix length named by* M*, every consumer-visible value of the functional is routed through the same barred prefix window and hence through the same tolerance row. The extraction uses finite prefix inspection only.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalUniformModulusExtraction`

Proof. Project the accepted carrier and read its $\mathsf{FanTheorem}^{\uparrow}$ row. The finite bar handoff supplies a unary depth ledger and a finite branch-prefix window, so the inspection bound is visible before any downstream modulus row is formed.

Restrict the $\mathsf{CantorSpace}^{\uparrow}$ carrier to that window. Agreement of two Cantor histories through the prefix length named by M gives componentwise agreement of their Boolean ledgers on every prefix in B.

Replay the witness rows W through K. Each retained prefix already carries the local tolerance request, and the bar-depth evidence D says that no longer branch read is required. The same prefix window therefore supports the same consumer tolerance row for both histories.

Package the resulting bounded prefix length as the $\mathsf{UniformModulus}^{\uparrow}$ row M. Since the construction reads only the finite Cantor rows, finite fan depth, witness window, transports, continuation, provenance, and local certificate, it introduces no host functional and no choice-selected branch. □

Theorem 1525.10 (FanFunctional Cantor-prefix modulus handoff)**.** *An accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier routes every Cantor-prefix finite-bar row through* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *before exposing the extracted modulus row* M*. The handoff reads the Cantor prefix window* B*, finite bar and depth rows* F, D*, tolerance witnesses* W*, compact finite-prefix coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it exports no selected branch, host functional, open-cover compactness principle, or countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCantorPrefixModulusHandoff`

Proof. Project the carrier of Definition 1525.1. Its Cantor-facing finite rows are the prefix window B, the fan row F, the depth ledger D, and the tolerance witness family W.

Apply the NameCert obligations of Definition 1525.2. They keep those rows on the same displayed route through hsame, Cont, Pkg, and local naming before any consumer may inspect the extracted modulus.

Now use uniform modulus extraction, Definition 1525.9. The extracted row M is determined by the finite fan depth, Cantor prefix window, and witness rows, so the $\mathsf{UniformContinuity}^{\uparrow}$ and $\mathsf{CompactMetric}^{\uparrow}$ consumers receive only the finite route that reaches $\mathsf{UniformModulus}^{\uparrow}$.

Thus every Cantor-prefix finite-bar read passes through $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ before M is exposed. Any selected branch, host functional, open-cover compactness proof, or countable-choice schedule would require a coordinate absent from the accepted $\mathsf{FanFunctional}^{\uparrow}$ carrier. □

Theorem 1525.11 (FanFunctional compact-continuity consumer)**.** *A compact-continuity consumer citing an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet may read exactly the Cantor-prefix rows of* C*, the finite-bar and depth rows of* F*, the tolerance request* ε*, the witness window* W*, the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row* M*, and the support rows* H, K, P, N*. This is the Cantor-space instance of the compact-metric and uniform-continuity consumer surface: compactness is consumed only through finite prefix coverage and finite bar depth, while continuity is consumed only through the displayed tolerance-to-modulus ledger.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCompactContinuityConsumer`

Proof. Start with the compact-metric dependency. In the Cantor-space case, the compact carrier is presented by finite Boolean prefix coverage, so the consumer can read compactness only as the prefix rows already present in C and B.

Use the $\mathsf{FanTheorem}^{\uparrow}$ dependency to bound the prefix inspection. Its decidable bar and unary depth rows supply the finite branch-depth evidence D, and the witness window W records the tolerance rows attached to those prefixes.

Use the $\mathsf{UniformContinuity}^{\uparrow}$ dependency only at the modulus handoff. The row M is the explicit $\mathsf{UniformModulus}^{\uparrow}$ output supplied by the finite fan inspection, and the continuation route K is the only path by which a consumer may read it.

Thus the compact-continuity consumer receives exactly the displayed finite surface. Host compactness, open covers, selected branches, semantic continuity, and real completeness are absent from the carrier and cannot be exported. □

Theorem 1525.12 (FanFunctional Cantor modulus ledger)**.** *Let $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$ be an accepted $\mathsf{FanFunctional}^{\uparrow}$ carrier. The extracted $\mathsf{UniformModulus}^{\uparrow}$ row M is determined only by the finite Cantor-prefix row C, the fan row F, the branch-prefix window B, the depth evidence D, the witness window W, and the support route H, K, P, N. Any two accepted packets with the same displayed rows $C, F, B, D, W, H, K, P, N$ expose the same consumer-visible modulus ledger, and no host functional, selected branch, open-cover compactness principle, or $\mathsf{Real}^{\uparrow}$ completeness datum contributes to the determination of M.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCantorModulusLedger`

Proof. Project the carrier surface of Definition 1525.1. The Cantor row C, fan row F, branch-prefix window B, depth evidence D, and witness window W are finite BHist rows, so the modulus read begins from displayed prefix and fan data rather than from a host functional.

Use Definition 1525.2 to bind the carrier admission, classifier stability, bar-depth exactness, witness-window coverage, uniform-modulus extraction, and non-escape clauses. These obligations force M to be read only after the finite prefix, fan, depth, and witness rows have been admitted.

Apply Definition 1525.9 for the extraction route and Definition 1525.11 for the consumer scope. Transport by H, replay by K, provenance by P, and local naming by N preserve the same finite route, so equal displayed ledger rows give the same public modulus read and the excluded compactness, branch-selection, host-functional, and Real-completeness data never enter. □

Theorem 1525.13 (FanFunctional compact-metric finite-net route)**.** *For an accepted $\mathsf{FanFunctional}^{\uparrow}$ carrier $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$, the compact-metric input read by any consumer is the finite Cantor-prefix net B together with the depth ledger D and witness window W. The route from $\mathsf{CompactMetric}^{\uparrow}$ to the $\mathsf{UniformModulus}^{\uparrow}$ output is therefore*

$$\mathsf{CompactMetric}^{\uparrow}(B, D, W) \longmapsto \mathsf{FanTheorem}^{\uparrow}(F, D) \longmapsto \mathsf{UniformContinuity}^{\uparrow}(\varepsilon, W) \longmapsto \mathsf{UniformModulus}^{\uparrow}(M),$$

with all transport and replay confined to H, K, P, N.

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCompactMetricFiniteNetRoute`

Proof. Project the accepted carrier. The only compact-metric data present are the finite Cantor-prefix rows of C restricted to the retained window B, the bar-depth ledger D, and the tolerance witnesses W.

Replay the compact route through K. The finite-net read first exposes B, D, W, then uses the fan row F to certify that every retained prefix has met the displayed decidable bar within the named depth.

Use the uniform-continuity row only after that finite coverage has been read. The tolerance request ε and witness window W determine the output row M, while H, P, N preserve the same

finite packet under transport, provenance, and local naming. No open cover, selected branch, or host compact space is introduced. □

Theorem 1525.14 (FanFunctional fan-theorem uniform depth)**.** *Let Φ be an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier. The* $\mathsf{FanTheorem}^{\uparrow}$ *row F supplies a single unary depth ledger D for the displayed finite fan, and the extracted modulus row M may inspect Cantor histories only through prefixes bounded by that ledger. Thus any two accepted* $\mathsf{CantorSpace}^{\uparrow}$ *histories agreeing through the depth named by D reach the same retained prefix window before the* $\mathsf{UniformModulus}^{\uparrow}$ *row is read.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalFanTheoremUniformDepth`

Proof. Read F and D from the carrier. The fan theorem row is already presented as finite decidable-bar data, so the depth ledger is a displayed unary object and not a host proof over all infinite branches.

Restrict the Cantor rows of C to that depth. Agreement of two histories up to the ledger named by D gives the same retained prefix window B and hence the same indexed witness family W.

The modulus output M is downstream from this window. Replay through K and transport through H keep the read inside the displayed packet, while P and N record provenance and local naming. The construction extracts a uniform depth from finite fan data only. □

Theorem 1525.15 (FanFunctional uniform-modulus consumer scope)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet may read exactly the Cantor-prefix row C, finite prefix net B, fan-depth ledger D, tolerance request ε, witness window W, extracted modulus row M, and support rows H, K, P, N. The consumer scope excludes host branch choice, open-cover compactness, semantic continuity over a function space, and any modulus not obtained from the displayed finite fan route.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalUniformmodulusConsumerScope`

Proof. Start from the carrier coordinates. They expose $C, F, \varepsilon, B, D, W, M$ and the structural rows H, K, P, N; there is no coordinate for a host function space, a selected branch, or an open cover.

The consumer first reads the finite prefix net B and the depth ledger D. Those rows determine which Cantor prefixes are available before the modulus handoff.

It then reads the tolerance witnesses W and output row M through the replay route K. Componentwise transport, provenance, and local naming preserve this same finite surface, so every consumer-visible modulus factors through the displayed fan route. □

Definition 1525.16 (FanFunctional root bar carrier)**.** *A root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *carrier is the finite packet*

$$\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$$

where C is the $\mathsf{CantorSpace}^{\uparrow}$ *prefix StreamName row, K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-prefix coverage row, F is the* $\mathsf{FanTheorem}^{\uparrow}$ *decidable-bar row, U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance-to-prefix row, M is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row, B is the finite root-bar window, W is the retained* $\mathsf{ProbeBundle}$ *witness family, D is the unary depth ledger, H is componentwise* hsame *transport, R is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row. The packet has no coordinate for countable choice, selected infinite branches, host functionals, or open-cover compactness.*

Theorem 1525.17 (FanFunctional root uniform-modulus obligation)**.** *Every accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$ *exposes the* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *only after the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-prefix coverage,* $\mathsf{FanTheorem}^{\uparrow}$ *decidable bar,* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance row, finite root-bar window,* $\mathsf{ProbeBundle}$ *witnesses, and unary depth ledger have all been displayed. The obligation is a finite* BHist *handoff and never a countable-choice selection of branch witnesses.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUniformModulusObligation`

Proof. Project the carrier of Definition 1525.16. The rows C, K, F, U expose the Cantor prefix data, compact finite-prefix coverage, decidable fan bar, and tolerance-to-prefix ledger before any modulus output is available.

Read the finite root data next. The window B, witness family W, and depth ledger D record the finite bar and prefix budget consumed by the modulus row.

The row M is therefore downstream of finite displayed data. Transport, replay, provenance, and local naming through H, R, P, N preserve that ordering, so no countable-choice selector, selected infinite branch, host functional, or open-cover compactness row can be read. □

Theorem 1525.18 (FanFunctional root Cantor-window handoff)**.** *Every* $\mathsf{CantorSpace}^{\uparrow}$ *consumer of an accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet receives only the finite prefix window*

$$C \longmapsto B, W, D$$

before the $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$ *rows are inspected. The handoff is a StreamName and ProbeBundle window over Boolean prefixes, not an infinite branch, not a selected path through the fan, and not a countable-choice sequence.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootCantorWindowHandoff`

Proof. Start with the Cantor prefix row C. It exposes Boolean observations through finite StreamName windows and contains no infinite branch coordinate.

The root-bar rows B, W, D restrict those observations to the displayed finite prefix window. The witness family W is indexed by the finite ProbeBundle data, and the depth ledger D bounds the inspection.

Only after this window is displayed may F and M be read. Thus the Cantor consumer receives a finite handoff over C, B, W, D, with no selected branch or countable-choice sequence. □

Theorem 1525.19 (FanFunctional root choice-free nonescape)**.** *No accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet can export countable choice, a selected infinite branch, or a host functional. Every public read factors through the finite rows* $C, K, F, U, M, B, W, D, H, R, P, N$*, and any attempted read that bypasses the finite root-bar window or the displayed* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row leaves the packet.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootChoiceFreeNonescape`

Proof. Inspect the carrier coordinates in Definition 1525.16. They are finite Cantor-prefix, compact-prefix, fan-bar, uniform-continuity, uniform-modulus, root-window, witness-family, depth, transport, replay, provenance, and naming rows.

There is no coordinate whose type is a stream of choices, a selected branch, or a host-level functional. Componentwise hsame transport in H can move only the displayed coordinates, and replay in R reuses only the same finite route.

Thus every public read is exhausted by $C, K, F, U, M, B, W, D, H, R, P, N$. A reader that claims choice, branch selection, or a host functional has introduced data not present in the accepted packet. □

Theorem 1525.20 (FanFunctional root uniform-continuity consumer)**.** *Every* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer of an accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet reads the tolerance-to-prefix row* U *together with the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row* C*,* $\mathsf{CompactMetric}^{\uparrow}$ *finite coverage row* K*,* $\mathsf{FanTheorem}^{\uparrow}$ *bar row* F*, finite root-bar window* B, W, D*, and the output* $\mathsf{UniformModulus}^{\uparrow}$ *row* M*. The consumer receives a finite modulus surface over Cantor prefixes and ProbeBundle witnesses, not semantic continuity over a host function space.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUniformContinuityConsumer`

Proof. Project the accepted packet and read the uniform-continuity row U. Its tolerance request is attached to the same Cantor prefix and compact finite coverage rows C, K.

Use the fan row F and root-bar window B, W, D to bound the inspection. These rows provide the finite prefix budget that the consumer may use when replaying the tolerance-to-prefix ledger.

The output row M is then the visible $\mathsf{UniformModulus}^{\uparrow}$ handoff, and H, R, P, N preserve it as finite NameCert data. Since the packet has no host functional-space coordinate, the consumer obtains only the displayed finite modulus surface. □

Theorem 1525.21 (FanFunctional root Cantor compact modulus)**.** *For every accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$*, the row family*

$$C, \quad F, \quad U, \quad K, \quad M, \quad B, \quad D, \quad W, \quad H, \quad R, \quad P, \quad N$$

forces the exported $\mathsf{UniformModulus}^{\uparrow}$ *row* M *through the displayed* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, and* $\mathsf{CompactMetric}^{\uparrow}$ *rows only. A consumer reads* M *by first reading the Cantor prefix row of chapter 1475, the finite fan row of chapter 400, the compact finite-net row of chapter 129, and the tolerance row of chapter 391; the modulus surface then lands at chapter 369 as displayed* BHist *rows, not as a host compactness or continuity principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootCantorCompactModulus`

Proof. Project the root-bar carrier. The rows C, F, U, K are the only Cantor, fan, uniform-continuity, and compact-metric source rows available before the modulus row M is read.

The finite window rows B, D, W bind those sources to displayed prefixes. They record the root-bar window, unary depth ledger, and retained ProbeBundle witnesses, so the compact metric input is a finite-net row over the same Cantor prefixes.

Finally transport and replay through H, R, P, N. These structural rows preserve the displayed BHist route and cannot add a host compactness theorem, a semantic continuity principle, or a hidden modulus selector. Thus M is forced through the visible Cantor, fan, compact-net, and tolerance rows. □

Theorem 1525.22 (FanFunctional root finite-depth exhaustion)**.** *Every finite-depth read of an accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$ *is exhausted by the displayed rows*

$$C, \quad F, \quad B, \quad D, \quad W, \quad K, \quad U, \quad M, \quad H, \quad R, \quad P, \quad N.$$

The CantorSpace, FanTheorem, CompactMetric, UniformContinuity, and UniformModulus siblings are consumed only through these finite BHist *rows:* C *supplies Boolean prefix windows,* F

supplies the decidable fan bar, K supplies finite-net coverage, U supplies tolerance-to-prefix evidence, and M is exported only at the retained depth.

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootFiniteDepthExhaustion`

Proof. Start with the visible depth data. The rows C, B, D, W are finite Cantor-prefix, root-window, unary-depth, and witness-family data, so they contain no selected infinite branch.

Read the fan and compact rows at the same depth. The row F verifies the finite bar over the displayed prefixes, while K gives compact finite-net coverage for that same bounded window.

The tolerance and modulus rows are then downstream. The row U attaches the tolerance request to the bounded window, and M is the resulting UniformModulus readback. Transport, replay, provenance, and local naming keep this route inside H, R, P, N, so finite depth is exhausted before any hidden selector could be claimed. □

Theorem 1525.23 (FanFunctional root uniform-modulus readback)**.** *For every accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$*, the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row M is read back from the* $\mathsf{UniformContinuity}^{\uparrow}$ *row U only after the* $\mathsf{CantorSpace}^{\uparrow}$ *row C, the* $\mathsf{FanTheorem}^{\uparrow}$ *row F, the* $\mathsf{CompactMetric}^{\uparrow}$ *row K, and the finite root rows B, D, W have been displayed. The readback is a displayed* BHist *route through H, R, P, N, not a quotient of host functionals, an open-cover compactness argument, or a selected branch through Cantor space.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUniformModulusReadback`

Proof. Project Θ and read the source tolerance row U. Its request is attached to the displayed Cantor prefix row C and cannot be interpreted as semantic continuity over a host function space.

Next use the finite fan and compact rows. The row F supplies the decidable bar depth, K supplies finite-net coverage, and B, D, W record the retained root-bar window and witnesses at that depth.

Only then replay the output row M. The rows H, R, P, N transport, replay, source, and name the same finite route, so the UniformModulus export is a displayed readback from Cantor, FanTheorem, CompactMetric, and UniformContinuity rows rather than an external compactness or choice principle. □

Theorem 1525.24 (FanFunctional modulus-extraction obligations)**.** *For an accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$*, the modulus-extraction obligation is the finite route that first reads the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C, then the* $\mathsf{FanTheorem}^{\uparrow}$ *decidable-bar row F, then the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance row U, and only then the* $\mathsf{UniformModulus}^{\uparrow}$ *output row M. The extracted modulus is valid exactly for the Cantor prefixes retained by B, W, D and transported by H, R, P, N.*

Proof. Project the carrier of Definition 1525.16. The rows C, F, U are displayed before M, and the finite window B, W, D records which Cantor prefixes reach the decidable fan bar.

Replay the route through R. The tolerance row U is consumed only after the bar window has been exposed, so the output M is a bounded prefix row rather than a host functional-space modulus.

Componentwise transport through H and provenance through P, N keep the same finite packet attached to the modulus read. Therefore modulus extraction is exactly the displayed $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ handoff. □

Theorem 1525.25 (FanFunctional Cantor-window coverage obligations)**.** *Every accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet covers its public Cantor-window reads through the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-prefix row* K*, the retained window* B*, the witness family* W*, and the depth ledger* D*. A consumer may inspect only the finite-net rows displayed by* K, B, W, D *before reading the modulus row* M*.*

Proof. Read the compact row K from Θ. In this packet the compact-metric input is not an open cover; it is the finite Cantor-prefix net already recorded by the root-bar carrier.

The rows B, W, D refine that net. They select the retained prefix window, attach the Probe-Bundle witnesses, and give the unary depth ledger by which the fan row certifies coverage.

Replay through R and transport through H. Since every public read of M must pass through K, B, W, D, P, N, the Cantor-window coverage obligation is finite-net coverage and no stronger compactness principle is exported. □

Theorem 1525.26 (FanFunctional choice-free nonescape obligations)**.** *The root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet exports no selected infinite branch, no countable choice, no open-cover compactness principle, and no host continuity principle. Every public read factors through the displayed rows* $C, K, F, U, M, B, W, D, H, R, P, N$ *and through the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row.*

Proof. Inspect the carrier coordinates. They are finite Cantor-prefix, compact finite-net, decidable fan-bar, tolerance, modulus, root-window, witness, depth, transport, replay, provenance, and naming rows.

There is no row for an infinite branch selector, countable-choice sequence, open cover, or host continuity axiom. The replay row R can only revisit the displayed route, and H can only transport the listed coordinates.

Thus any consumer that remains inside the packet reads through $C, K, F, U, M, B, W, D, H, R, P, N$. A claim of branch selection, countable choice, open-cover compactness, or host continuity would require data absent from the finite $\mathsf{NameCert}$ surface. □

Theorem 1525.27 (FanFunctional root depth exhaustion)**.** *For every accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$*, every root-depth read is exhausted by the displayed* $\mathsf{CantorSpace}^{\uparrow}$ *windows* C, B*, the* $\mathsf{FanTheorem}^{\uparrow}$ *finite bar row* F*, the finite witness family* W*, the depth ledger* D*, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row* K*, and the structural rows* H, R, P, N*. The packet exports no countable choice sequence, host compactness principle, or hidden modulus selector.*

Proof. Project the carrier of Definition 1525.16. The rows C, B, W, D are finite Cantor-window and depth data, while F records the finite-bar route and K records the compact finite-net input.

Replay the public read through R. The depth ledger is consumed only together with C, K, F, B, W, so a consumer cannot detach D as an independent branch selector or as an open-cover compactness theorem.

Transport and provenance through H, P, N preserve the same finite packet. Since no additional coordinate names countable choice or a hidden selector, root depth is exhausted by the displayed rows. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootDepthExhaustion`

Theorem 1525.28 (FanFunctional Cantor-window modulus readback)**.** *For an accepted root-unblock* $\mathsf{FanFunctional}^{\uparrow}$ *packet* $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$*, the* $\mathsf{UniformContinuity}^{\uparrow}$ *source rows* U *may be read back into a* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row* M *only through the retained*

CantorSpace$^{\uparrow}$ *windows* C, B*, the* FanTheorem$^{\uparrow}$ *finite bar/depth rows* F, D*, the witness family* W*, and the* CompactMetric$^{\uparrow}$ *finite-net row* K*. The readback is a finite NameCert route through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, not a host continuity or compactness export.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCantorWindowModulusReadback`

Proof. Start with the uniform-continuity row U. Its tolerance request is attached to displayed Cantor prefixes rather than to a semantic function space.

Use the Cantor-window rows C, B and the fan rows F, D. These rows bound the prefix depth and identify the finite bar evidence used by the witness family W.

The compact row K supplies only finite-net coverage for those displayed windows. Replaying through R then delivers the modulus row M, while H, P, N keep the BHist, hsame, Cont, Pkg, and NameCert structure visible. □

Theorem 1525.29 (FanFunctional UniformModulus consumer package)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted root-unblock* FanFunctional$^{\uparrow}$ *packet receives the package*

$$C, K, F, U, M, B, W, D, H, R, P, N$$

as visible finite carrier rows: CantorSpace$^{\uparrow}$ *windows,* FanTheorem$^{\uparrow}$ *finite bar and depth rows,* UniformContinuity$^{\uparrow}$ *source rows, the* UniformModulus$^{\uparrow}$ *consumer row,* CompactMetric$^{\uparrow}$ *finite-net rows,* BHist *witnesses, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It receives no countable choice principle, hidden compactness theorem, or unrecorded modulus selector.*

Proof. Inspect the displayed package. The consumer row M is not read alone; it is attached to the source row U, the Cantor windows C, B, the fan/depth rows F, D, witness data W, and finite-net data K.

The replay row R supplies the route from source tolerance to consumer modulus. Componentwise transport through H and provenance through P, N preserve the same finite carrier rows and do not add a selector coordinate.

Therefore a UniformModulus$^{\uparrow}$ consumer gets exactly the visible NameCert package. Any claim of countable choice, host compactness, or a hidden modulus selector would require a row outside Θ. □

Theorem 1525.30 (FanFunctional root finite-modulus route)**.** *Every accepted root-unblock* FanFunctional$^{\uparrow}$ *packet exposes a* UniformModulus$^{\uparrow}$ *read only through the finite route*

$$C, K, F, U, M, B, W, D, H, R, P, N.$$

Here C *and* B *are the* CantorSpace$^{\uparrow}$ *prefix-window rows,* K *is the* CompactMetric$^{\uparrow}$ *finite-net row,* F *is the* FanTheorem$^{\uparrow}$ *finite-bar row,* U *is the* UniformContinuity$^{\uparrow}$ *source row,* W *is the* ProbeBundle *witness family,* D *is the displayed depth ledger,* H *transports the rows componentwise by* hsame*,* R *replays the displayed* Cont *route,* P *records provenance, and* N *is the local* NameCert *row. No other coordinate of the accepted packet can expose a selected branch, a countable-choice operator, a host functional, or an open-cover compactness theorem.*

Proof. Project the accepted carrier $\Theta = (C, K, F, U, M, B, W, D, H, R, P, N)$. The only rows that mention a modulus request are the UniformContinuity$^{\uparrow}$ row U, the UniformModulus$^{\uparrow}$ row M, and the finite Cantor, fan, witness, depth, transport, replay, provenance, and naming rows displayed with them.

The root-bar data are finite. The Cantor rows C, B bound the prefix windows, F, D name the accepted bar and depth, W supplies the displayed ProbeBundle witnesses, and K supplies only the compact finite-net input needed by those windows.

Replay through R and transport through H. These rows can only move the same finite packet to the consumer row M, while P, N record its source and local name. Since the carrier has no row for a branch selector, countable choice, host functional evaluation, or open-cover compactness, the $\mathsf{UniformModulus}^{\uparrow}$ read is exhausted by the displayed finite route. □

Theorem 1525.31 (FanFunctional compact-uniform consumer route)**.** *Any compact-uniform-continuity consumer that receives an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *root-unblock packet receives only the finite route of Definition 1525.30: Cantor prefix windows, the compact finite-net row, fan-bar and depth rows, uniform-continuity source rows, the uniform-modulus consumer row,* ProbeBundle *witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The handoff supplies no selected branch, countable choice, host functional, or open-cover compactness object.*

Proof. Start with a compact-uniform-continuity consumer read. By Definition 1525.29, the consumer receives the package $C, K, F, U, M, B, W, D, H, R, P, N$ as visible finite carrier rows, not as an ambient function-space principle.

Apply Definition 1525.30. The route from U to M factors through the Cantor windows, finite-net row, fan-bar row, witness bundle, depth ledger, replay row, transport row, provenance, and local name certificate.

Thus the compact-uniform-continuity handoff is a finite NameCert route. A selected branch, countable-choice schedule, host functional, or open-cover compactness object would require an additional row outside the accepted packet, so none is delivered to the consumer. □

Theorem 1525.32 (FanFunctional Cantor compact route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *row exported by an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *root-unblock packet factors through the Cantor finite-prefix rows, the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-depth ledger, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, and the* $\mathsf{UniformModulus}^{\uparrow}$ *readback route before any* $\mathsf{Real}^{\uparrow}$*-facing or compact-uniform consumer can read it. Thus the exported modulus is a finite Cantor-compact NameCert route over*

$$C, K, F, U, M, B, W, D, H, R, P, N$$

and not a selected branch, countable-choice schedule, host functional, or open-cover compactness theorem.

Proof. Start from the compact-uniform consumer route of Definition 1525.31. It already places every consumer read behind the finite package $C, K, F, U, M, B, W, D, H, R, P, N$.

Use Definition 1525.30 to locate the modulus row. The route from U to M passes through the Cantor prefix windows C, B, the fan-bar and finite-depth rows F, D, the witness windows W, and the compact finite-net row K before the readback row R is available.

Finally apply Definition 1525.29. The package delivered to a $\mathsf{UniformModulus}^{\uparrow}$ consumer is exactly the displayed finite NameCert packet, so no branch selector, countable-choice schedule, host functional, or open-cover compactness object can be exported to the Real-facing consumer. □

Definition 1525.33 (FanFunctional finite-window carrier)**.** *A* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window carrier is a finite* BHist *packet*

$$\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$$

where B is the $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix row, Q is the inspected prefix-query ledger, F, D are the* $\mathsf{FanTheorem}^{\uparrow}$ *finite bar and depth rows, U, T are the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance ledger and requested tolerance row, K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net consumer route, W is the finite witness-window family, H transports all displayed rows componentwise by* hsame*, R replays the displayed* Cont *route, P is the* Pkg *provenance over the displayed packets, and N is the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row. The classifier compares accepted packets only through those rows and has no coordinate for a host functional, selected branch, open-cover compactness proof, countable choice, or Real completeness.*

Theorem 1525.34 (FanFunctional Cantor-window obligation)**.** *Every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window carrier must expose its* $\mathsf{CantorSpace}^{\uparrow}$ *information as finite* BHist *rows B, Q, W before the* $\mathsf{FanTheorem}^{\uparrow}$ *bar/depth rows F, D or the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row may be consumed. No consumer may replace those rows by a selected infinite branch or host functional evaluation.*

Proof. Project the carrier of Definition 1525.33. The only Cantor-facing coordinates are B, Q, W, and they are finite rows in the same BHist packet.

The bar and naming rows are downstream. The $\mathsf{FanTheorem}^{\uparrow}$ rows F, D read prefix coverage and depth only from the displayed Cantor windows, while H, R, P, N transport, replay, source, and name that same finite route.

Since the carrier contains no selected branch or host functional coordinate, any consumer that bypasses B, Q, W leaves the accepted packet. □

Theorem 1525.35 (FanFunctional uniform-modulus obligation)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window carrier $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$, the uniform-modulus read must pass through the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance ledger U, T, the finite* $\mathsf{FanTheorem}^{\uparrow}$ *bar/depth rows F, D, the witness-window row W, componentwise* hsame *transport H, displayed* Cont *replay R, package provenance P, and the local* NameCert *row N. The packet exports no host functional, branch selector, countable-choice schedule, or Real-completeness coordinate.*

Proof. Read the tolerance face first. The rows U, T are the only $\mathsf{UniformContinuity}^{\uparrow}$ coordinates that name the requested tolerance and the ledger supporting it.

Attach the fan and window rows. The finite bar/depth rows F, D and witness row W provide the only bounded prefix information that can answer the tolerance request.

Transport and replay do not enlarge the surface. The rows H, R, P, N preserve, replay, source, and name the same finite packet, so the excluded host and choice principles cannot be recovered from an accepted uniform-modulus read. □

Theorem 1525.36 (FanFunctional compact-metric consumer obligation)**.** *Any* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net consumer of a* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window carrier receives only the displayed route $K, B, Q, F, D, U, T, W, H, R, P, N$. In particular the compact-metric handoff consumes finite Cantor prefixes, finite bar/depth rows, the uniform-continuity tolerance ledger, witness windows,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it receives no open-cover compactness theorem, selected branch, countable-choice selector, host functional, or Real completeness.*

Proof. Project the compact-facing row of the carrier. The row K is the only $\mathsf{CompactMetric}^{\uparrow}$ coordinate, and it is stored beside the finite Cantor, fan, tolerance, witness, transport, replay, provenance, and naming rows.

The consumer route is therefore finite. It reads B, Q for prefixes, F, D for bar and depth, U, T for the tolerance ledger, and W for the witness windows before receiving K.

Finally H, R, P, N keep the handoff on the same packet. Since no open-cover compactness proof, selected branch, choice selector, host functional, or Real-completeness row appears in Ω, the compact-metric consumer receives only the displayed route. □

Theorem 1525.37 (FanFunctional NameCert nonescape)**.** *The local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^\uparrow}$ *row of an accepted finite-window carrier* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$ *is exhausted by finite* BHist *rows for* $\mathsf{CantorSpace}^\uparrow$ *prefixes,* $\mathsf{FanTheorem}^\uparrow$ *bar/depth data,* $\mathsf{UniformContinuity}^\uparrow$ *tolerance ledger,* $\mathsf{CompactMetric}^\uparrow$ *finite-net consumption, witness windows, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and local naming. It cannot expose a host functional, selected branch, open-cover compactness proof, countable choice, or Real-completeness principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalNameCertNonescape`

Proof. Combine the three obligation blocks. The Cantor-window obligation, Definition 1525.34, fixes the finite prefix rows that feed the fan bar/depth data.

The uniform-modulus obligation, Definition 1525.35, fixes the tolerance ledger and witness-window route. The compact-metric consumer obligation, Definition 1525.36, fixes the finite-net consumer route.

All three routes use the same structural rows H, R, P, N. Therefore the local NameCert row names exactly the displayed finite packet, and any host functional, selected branch, open-cover compactness proof, countable-choice selector, or Real-completeness principle would require a coordinate outside the accepted carrier. □

Theorem 1525.38 (FanFunctional root schema carrier)**.** *The root schema row for* $\mathsf{FanFunctional}^\uparrow$ *is the finite Cantor–fan–modulus carrier*

$$\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$$

of Definition 1525.33. Its root-facing fields are exactly the $\mathsf{CantorSpace}^\uparrow$ *finite-prefix rows* B, Q, W*, the* $\mathsf{FanTheorem}^\uparrow$ *bar and depth rows* F, D*, the* $\mathsf{UniformModulus}^\uparrow$ *and* $\mathsf{UniformContinuity}^\uparrow$ *tolerance rows* U, T*, the* $\mathsf{CompactMetric}^\uparrow$ *finite-net consumer row* K*, and the structural rows* H, R, P, N*.*

Proof. Project Definition 1525.33. The tuple displayed there already separates the Cantor prefix coordinates, fan bar-depth coordinates, uniform-modulus coordinates, compact-metric consumer coordinate, and structural coordinates.

No root schema row is added outside the tuple. The carrier has no coordinate for a host functional, selected branch, open-cover compactness theorem, Real completeness, or countable-choice schedule, so the listed rows are exactly the root-facing carrier fields. □

Theorem 1525.39 (FanFunctional root schema classifier)**.** *Two accepted* $\mathsf{FanFunctional}^\uparrow$ *root schema carriers classify together whenever their* $\mathsf{CantorSpace}^\uparrow$ *prefix windows,* $\mathsf{FanTheorem}^\uparrow$ *finite bar depths,* $\mathsf{UniformContinuity}^\uparrow$ *tolerance rows,* $\mathsf{UniformModulus}^\uparrow$ *witness windows,* $\mathsf{CompactMetric}^\uparrow$ *finite-net rows, and structural rows* H, R, P, N *transport componentwise by* hsame *over the displayed finite windows. The classifier has no branch-selector, host-functional, open-cover, or Real-completeness coordinate.*

Proof. Compare the two carriers coordinate by coordinate. Agreement on the Cantor rows fixes the inspected prefix windows, and agreement on the fan rows fixes the finite bar depth available to the modulus read.

Transport the tolerance and witness rows through the same windows. Componentwise hsame keeps U, T, W aligned with the displayed bar depth, and the compact-metric row K reads only that finite packet.

The structural rows preserve the comparison. Since H, R, P, N transport, replay, source, and name exactly the displayed fields, a classifier disagreement would require a coordinate outside the root schema carrier. □

Theorem 1525.40 (FanFunctional root schema ledger)**.** *Every downstream root-schema consumer of* FanFunctional$^{\uparrow}$ *must pass through* CompactMetric$^{\uparrow}$, UniformContinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *as displayed by* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$. *The ledger records non-escape from finite Cantor windows and finite bar depth to the compact and uniform-modulus consumers, and it exports no host functional, selected infinite branch, open-cover compactness proof, countable-choice selector, or Real-completeness principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootSchemaLedger`

Proof. Start from the root schema carrier. The compact consumer row K is downstream from B, Q, F, D, U, T, W, so it cannot be read before the Cantor window, fan bar depth, tolerance, and witness rows are displayed.

Next read the uniform-modulus side. The rows U, T, W provide the finite tolerance and witness-window route, and the UniformModulus$^{\uparrow}$ consumer receives that route through the same finite bar depth.

Finally attach the structural ledger. Rows H, R, P, N keep the consumer inside the finite packet by componentwise transport, displayed continuation replay, package provenance, and local naming, so the excluded host and choice objects have no ledger slot. □

Definition 1525.41 (FanFunctional finite-modulus carrier)**.** *A* FanFunctional$^{\uparrow}$ *finite-modulus carrier is a finite* BHist *packet*

$$\Theta = (C, F, K, U, M, W, D, H, R, P, N)$$

whose rows are as follows:

1. *C is the* CantorSpace$^{\uparrow}$ *finite-prefix source row, including the bounded prefix windows inspected by the packet.*
2. *F is the* FanTheorem$^{\uparrow}$ *finite-bar row that supplies the displayed depth ledger for those windows.*
3. *K is the* CompactMetric$^{\uparrow}$ *finite-net row consumed by compact-metric downstream routes.*
4. *U is the* UniformContinuity$^{\uparrow}$ *tolerance request row.*
5. *M is the* UniformModulus$^{\uparrow}$ *extracted modulus row.*
6. *W is the finite witness-window family linking each inspected prefix to the modulus read.*
7. *D is the depth ledger that records the bounded fan depth used before M may be exported.*
8. *H, R, P, N are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert$_{\text{FanFunctional}^{\uparrow}}$ *rows.*

The classifier compares accepted packets only through these finite rows and has no coordinate for a host functional, selected branch, open-cover compactness object, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 1525.42 (FanFunctional NameCert obligation surface)**.** *For an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-modulus carrier* $\Theta = (C, F, K, U, M, W, D, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *surface consists exactly of carrier admission for* Θ*,* $\mathsf{CantorSpace}^{\uparrow}$ *prefix exposure through* C*,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar exactness through* F, D*,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net handoff through* K*,* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance admission through* U*,* $\mathsf{UniformModulus}^{\uparrow}$ *extraction through* M, W*, and structural transport, replay, provenance, and naming through* H, R, P, N*.*

Proof. Project the carrier of Definition 1525.41. Its displayed rows are exactly the Cantor prefix source, fan bar and depth ledger, compact finite-net handoff, uniform-continuity tolerance request, extracted uniform-modulus row, witness windows, transport, replay, provenance, and local naming row.

The NameCert obligations are therefore the admission and exactness checks for those rows. Prefix exposure reads C; finite-bar exactness reads F, D; the compact-metric handoff reads K; the tolerance request reads U; and modulus extraction reads M, W.

The rows H, R, P, N account for componentwise transport, displayed consumer replay, package provenance, and local naming. Since no host functional, selected branch, open-cover compactness object, choice schedule, or Real-completeness row appears in Θ, the displayed list exhausts the local obligation surface. □

Theorem 1525.43 (FanFunctional Real uniform-modulus handoff)**.** *Every Real-facing uniform-modulus consumer of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-modulus carrier factors through the finite route*

$$\mathsf{CantorSpace}^{\uparrow} \longrightarrow \mathsf{FanTheorem}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{UniformContinuity}^{\uparrow} \longrightarrow \mathsf{UniformModulus}^{\uparrow}$$

as represented by the rows $C, F, K, U, M, W, D, H, R, P, N$*. The consumer receives only finite prefix windows, a finite fan depth, a compact finite-net row, a uniform-continuity tolerance row, witness windows, transport, replay, provenance, and local naming; it receives no host functional, selected branch, open-cover compactness proof, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Start with an accepted finite-modulus carrier. The only Cantor-facing coordinate is C, and the only fan-depth coordinates are F, D, so a consumer must read finite prefixes and the finite-bar ledger before any modulus row is visible.

Next the compact and uniform-continuity rows are read. Row K supplies the compact finite-net handoff, while row U supplies the tolerance request that the modulus must answer.

Finally the modulus export reads M through the witness windows W and the structural rows H, R, P, N. These rows preserve the same finite packet, so a Real-facing consumer cannot bypass the displayed route through a host functional, selected branch, open-cover compactness proof, choice schedule, or Real-completeness principle. □

Theorem 1525.44 (FanFunctional monotone bar modulus obligation)**.** *Every exported modulus row of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-modulus carrier must expose a monotone bar-modulus packet through the displayed route*

$$\mathsf{CantorSpace}^{\uparrow} \longrightarrow \mathsf{FanTheorem}^{\uparrow} \longrightarrow \mathsf{CompactMetric}^{\uparrow}, \mathsf{UniformContinuity}^{\uparrow} \longrightarrow \mathsf{UniformModulus}^{\uparrow}.$$

The obligation requires the $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix row* C*, the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar depth rows* F, D*, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net consumer row* K*, the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance row* U*, the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff rows* M, W*, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *rows* H, R, P, N*. It excludes selected branches, bar induction as a primitive, open-cover compactness, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $\Theta = (C, F, K, U, M, W, D, H, R, P, N)$. The prefix row C is the only Cantor source, and the rows F, D are the only finite-bar depth witnesses available before a modulus is exported.

The compact and continuity consumers are therefore forced to read K and U after the prefix-depth rows. The modulus handoff can then read only M, W, with H, R, P, N preserving transport, replay, provenance, and local naming for the same finite packet.

Since no row records a selected branch, primitive bar induction, open-cover compactness proof, countable-choice schedule, or ambient Real-completeness assertion, the monotone bar-modulus obligation is exactly the displayed finite route. □

Theorem 1525.45 (FanFunctional root finite-cover retention)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$*, the root finite-cover read retains the displayed* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row* C*,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row* F*,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net consumer row,* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance row,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row* M*,* BHist *packet route, componentwise* hsame *row* H*, displayed* Cont *replay row* K*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *row* N *on the same packet before any modulus consumer may inspect* M*. The retention surface has no selected branch, host functional, open-cover compactness proof, countable-choice selector, or ambient Real-completeness coordinate.*

Proof. Project the carrier of Definition 1525.1. The rows $C, F, \varepsilon, B, D, W, M, H, K, P, N$ are the only available root coordinates, and the compact-metric and uniform-continuity consumers read them as finite BHist data.

The finite-cover side is retained by F, B, D, W. These rows expose the finite fan depth, retained prefix window, and witness window before the extracted $\mathsf{UniformModulus}^{\uparrow}$ row can be consumed.

Finally H, K, P, N transport, replay, source, and name the same packet. Since selected branches, host functionals, open covers, choice schedules, and Real-completeness principles have no coordinate, the root read cannot leave the displayed finite-cover surface. □

Theorem 1525.46 (FanFunctional root modulus-row totality)**.** *Every accepted root read of* $\mathsf{FanFunctional}^{\uparrow}$ *exposes a total displayed route from* $\mathsf{CantorSpace}^{\uparrow}$ *and* $\mathsf{FanTheorem}^{\uparrow}$ *through* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *before exporting the modulus row. Concretely, the packet route through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *must show* $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *together; omitting any one of the Cantor, fan, compact-metric, uniform-continuity, or modulus rows leaves the accepted carrier.*

Proof. Begin with the NameCert obligations of Definition 1525.2. They separate carrier admission, classifier stability, bar-depth exactness, witness-window coverage, uniform-modulus extraction, and non-escape as rows of one finite packet.

The modulus row is downstream from the other root rows. The Cantor and fan rows fix the finite prefix and bar-depth surface, while the compact-metric and uniform-continuity rows consume only the displayed finite windows and tolerance request.

The structural route through H, K, P, N is total for that packet. A read missing one displayed row cannot reconstruct it from a selected branch, host functional, open-cover compactness proof, countable-choice schedule, or Real-completeness principle, because none of those objects is part of the carrier. □

Theorem 1525.47 (FanFunctional root choice boundary)**.** *No accepted* $\mathsf{FanFunctional}^{\uparrow}$ *root obligation may replace the finite packet route through* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *by a branch selector, host functional, open-cover compactness theorem, countable-choice schedule, or ambient*

Real-completeness argument. The only admissible root boundary is the displayed carrier $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$ *and its componentwise transport, replay, provenance, and local naming rows.*

Proof. Use Definition 1525.1. The accepted carrier lists every coordinate available to a root obligation, including the Cantor prefix, finite fan, compact-metric, uniform-continuity, uniform-modulus, transport, replay, provenance, and naming rows.

Apply the root schema ledger of Definition 1525.40. Downstream consumers must pass through the compact and uniform-modulus route displayed by Ω, hence through the same finite root boundary.

Coordinate exhaustion closes the choice boundary. A selected branch, host functional, open-cover compactness proof, countable-choice selector, or ambient Real-completeness argument would require a coordinate not present in Φ, so such a read is outside the accepted $\mathsf{FanFunctional}^{\uparrow}$ packet. □

Theorem 1525.48 (FanFunctional root finite-prefix uniform modulus)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$*, the root finite-prefix read exports the row* M *only after the carrier surface of Definition 1525.1 and the local obligations of Definition 1525.2 have exposed* $C, F, \varepsilon, B, D, W, H, K, P, N$ *on the same* BHist *packet. Thus* M *is a finite-prefix* $\mathsf{UniformModulus}^{\uparrow}$ *row supported by the displayed Cantor prefix, fan-depth, tolerance, branch-window, and witness-window rows, not by a host functional, selected branch, or ambient compactness principle.*

Proof. Project the carrier. The rows $C, F, \varepsilon, B, D, W, M, H, K, P, N$ are already displayed as finite BHist coordinates, and the NameCert obligation surface separates carrier admission from the modulus extraction clause.

Read the finite fan-depth ledger before the modulus row. Rows C, F expose the Cantor prefix and finite fan surface, row ε fixes the tolerance request, rows B, D bind the branch window and depth, and row W records the witnesses whose bounded inspection length is repacked as M.

The structural rows keep the handoff root-local. Componentwise transport through H, displayed replay through K, provenance through P, and local naming through N preserve the same finite packet, so no external branch selector, host functional, or open-cover compactness object can contribute to the exported modulus. □

Theorem 1525.49 (FanFunctional root witness-window coverage)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier* $\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$*, the witness-window row* W *covers exactly the retained branch prefixes in* B *whose finite bar-depth exposure is certified by* D*. Every root read of the modulus row* M *must first pass through that* $C, F, \varepsilon, B, D, W$ *coverage surface and then through* H, K, P, N*; it may not omit a displayed prefix, add an unseen prefix, or use a selected infinite branch.*

Proof. Start with the carrier surface. The retained prefixes are named by B, and the fan row F together with the depth ledger D determines which finite prefixes have reached the displayed bar before any witness is read.

Use the NameCert obligations. Witness-window coverage is a separate finite clause between bar-depth exactness and uniform-modulus extraction, so W cannot be replaced by a later choice of branch or by a host-level compactness proof.

Finally route the coverage through H, K, P, N. These rows transport, replay, source, and name the same packet, forcing every public modulus read of M to use the displayed coverage window and excluding omitted, added, or infinitely selected prefixes. □

Theorem 1525.50 (FanFunctional root nonescape ledger)**.** *Every accepted root-obligation read of* $\mathsf{FanFunctional}^{\uparrow}$ *is exhausted by the rows* $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *of Definition 1525.1 and by the local NameCert obligations of Definition 1525.2. The nonescape ledger exports the finite Cantor-prefix route, finite fan-depth route, witness-window coverage, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; it exports no host functional, choice-selected branch, open-cover compactness proof, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. List the available rows. Carrier admission gives exactly $C, F, \varepsilon, B, D, W, M, H, K, P, N$, and the existing NameCert surface assigns the root obligations to these coordinates.

Read the finite obligations in order. Classifier stability uses the displayed Cantor and fan rows, bar-depth exactness uses D, witness-window coverage uses W, and uniform-modulus extraction exports M only through those finite rows.

Coordinate exhaustion supplies the nonescape claim. Since the carrier has no coordinate for a host functional, branch selector, open cover, choice schedule, or Real-completeness witness, the structural route through H, K, P, N is the whole public ledger. □

Theorem 1525.51 (FanFunctional root bar-depth exactness)**.** *For an accepted finite-window carrier* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$*, the root read exposes the* $\mathsf{FanTheorem}^{\uparrow}$ *bar-depth row* D *before any* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{UniformContinuity}^{\uparrow}$ *modulus row is consumed. The depth seen by a modulus consumer is exactly the finite depth displayed by* F, D *over the Cantor prefix rows* B, Q*, not a selected branch, host functional, open-cover compactness proof, or Real-completeness witness.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootBarDepthExactness`

Proof. Project the finite-window carrier. Rows B, Q display the inspected Cantor prefixes, while F, D display the decidable bar and its finite depth before the tolerance and witness rows are read.

The modulus rows U, T, W are downstream from that display. Transport through H, R and provenance through P, N preserve the same finite depth, so a consumer cannot substitute an external branch selector or compactness proof for D. □

Theorem 1525.52 (FanFunctional root modulus-window coverage)**.** *Every root-schema modulus read of* $\mathsf{FanFunctional}^{\uparrow}$ *covers exactly the Cantor prefixes displayed in the finite windows* B, Q, W *of* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$*. The witness windows neither omit a displayed prefix before the modulus row is exported nor add a prefix outside the bar-depth boundary exposed by* F, D*.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootModulusWindowCoverage`

Proof. Use the root schema carrier and classifier theorems. The carrier fixes the prefix rows B, Q, W, and the classifier compares only those finite windows together with the bar-depth and tolerance rows.

The root schema ledger keeps the compact and uniform-modulus consumers on that route. Since no coordinate names an infinite branch, countable choice schedule, or open-cover compactness proof, coverage is exactly the displayed finite prefix family. □

Theorem 1525.53 (FanFunctional root NameCert acceptance)**.** *The local root-facing* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *acceptance row receives only* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *rows carried by* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$*. It does not receive a host functional, selected branch, arbitrary open-cover compactness row, choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootNamecertAcceptance`

Proof. Combine the root schema carrier, classifier, and ledger. They expose the finite Cantor prefix, bar-depth, tolerance, witness-window, compact-metric, uniform-continuity, and uniform-modulus rows before the local name is read.

The structural rows H, R, P, N provide exactly displayed replay, componentwise transport, package provenance, and local naming. Because the carrier has no host-functional, branch-selector, open-cover, choice, or Real-completeness coordinate, local NameCert acceptance receives only the listed rows. □

Theorem 1525.54 (FanFunctional root-unblock Cantor prefix)**.** *Every root-unblock read of an accepted* FanFunctional$^{\uparrow}$ *packet must first admit the displayed* CantorSpace$^{\uparrow}$ *finite prefix surface. Concretely, the read exposes the List spine, Bool-valued finite observation windows, StreamName schedule, and the surrounding* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before any fan-depth or modulus consumer can inspect the packet.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUnblockCantorPrefix`

Proof. Project the carrier of Definition 1525.1 and the finite-window carrier of Definition 1525.33. Their Cantor-facing coordinates are finite prefix data: List spines, Bool ledgers, StreamName windows, and the structural transport rows.

The root schema carrier of Definition 1525.38 reads those coordinates before the compact and uniform-modulus rows. Thus a root-unblock consumer receives the Cantor prefix as displayed finite data, not as an infinite branch or host functional.

Finally apply the root choice boundary of Definition 1525.47. Since no selector, open-cover proof, or ambient Real-completeness row is present, the Cantor prefix admission remains inside the accepted packet. □

Theorem 1525.55 (FanFunctional root-unblock finite fan depth)**.** *Every root-unblock read of* FanFunctional$^{\uparrow}$ *exposes the* FanTheorem$^{\uparrow}$ *finite fan-depth row through the displayed UnaryHistory depth ledger before witness-window coverage or uniform-modulus handoff is consumed. The depth read is a finite carrier projection over the same Cantor-prefix rows; it is not a bar-induction principle, selected branch argument, or open-cover compactness claim.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUnblockFanDepth`

Proof. Read the finite fan rows from Definition 1525.1. The row F supplies the decidable bar classifier, and the row D supplies the finite depth evidence attached to the retained branch-prefix window.

Use Definition 1525.4 and Definition 1525.51. Together they place the UnaryHistory depth after the Cantor-prefix rows and before any witness-window or modulus read.

The route is preserved by componentwise hsame, displayed Cont replay, Pkg provenance, and the local NameCert row. Coordinate exhaustion then excludes bar induction, selected branches, host functionals, and open-cover compactness from the fan-depth exposure. □

Theorem 1525.56 (FanFunctional root-unblock uniform modulus handoff)**.** *Every root-unblock consumer that reads the* UniformModulus$^{\uparrow}$ *row of an accepted* FanFunctional$^{\uparrow}$ *packet must keep the route through* CompactMetric$^{\uparrow}$*,* UniformContinuity$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *visible. The handoff uses the displayed Cantor prefix, finite fan depth, tolerance request, compact finite-window consumer row, and witness windows; it exports no selected branch,*

host functional, bar-induction principle, open-cover compactness proof, quotient stream equality, or ambient Real-completeness row.

Proof. Start with the Cantor-prefix admission of Definition 1525.54. It fixes the finite prefix windows and the structural rows available to the root-unblock read.

Next apply the finite fan-depth exposure of Definition 1525.55. The depth ledger bounds the retained prefix window before witness coverage or modulus handoff is consumed.

Use Definition 1525.5 and Definition 1525.52. They route the compact-metric and uniform-continuity consumers through the same finite windows before the $\mathsf{UniformModulus}^{\uparrow}$ row is exported.

Finally attach the root schema ledger. Rows BHist, hsame, Cont, Pkg, and NameCert keep the handoff on the displayed packet, so none of the excluded branch, host, induction, compactness, quotient, or Real-completeness objects can be read from the root-unblock surface. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUnblockUniformModulus`

Theorem 1525.57 (FanFunctional root-unblock compact route)**.** *Every root-unblock read of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet routes through* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *before exporting a modulus. The route exposes the Cantor prefix, finite fan-depth ledger, compact finite-window consumer, tolerance request, witness-window coverage, componentwise transport, displayed replay, package provenance, and local naming rows; it exports no selected branch, bar-induction principle, open-cover compactness proof, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUnblockCompactRoute`

Proof. Project the carrier rows of Definition 1525.1 and Definition 1525.33. The visible compact-route data are the Cantor-prefix windows, the finite fan-depth row, the compact metric consumer row, the uniform-continuity request, and the modulus handoff rows.

Invoke Definition 1525.54 for the Cantor-facing prefix and Definition 1525.55 for the finite fan-depth exposure. These theorems place $\mathsf{CantorSpace}^{\uparrow}$ and $\mathsf{FanTheorem}^{\uparrow}$ before any compact or modulus consumer can read the packet.

Use Definition 1525.56 to carry the compact-metric and uniform-continuity rows through the witness-window handoff to $\mathsf{UniformModulus}^{\uparrow}$. The structural rows BHist, hsame, Cont, Pkg, and NameCert keep the read on the same accepted packet.

Coordinate exhaustion gives the boundary. Since the accepted packet contains no selected branch, bar-induction row, open-cover compactness proof, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, the compact route exports only the displayed finite modulus surface. □

Theorem 1525.58 (FanFunctional root-unblock witness-window totality)**.** *Every root-unblock read that exposes the* $\mathsf{UniformModulus}^{\uparrow}$ *row of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet must also expose the witness-window row pairing retained* $\mathsf{CantorSpace}^{\uparrow}$ *prefixes with pointwise tolerance evidence. This window is read through* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*; it exports no selected branch, host functional, bar-induction row, open-cover compactness proof, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1525.54. It supplies the retained Cantor-prefix surface as finite List, Bool, and StreamName data before a modulus consumer can inspect the packet.

Apply Definition 1525.55. The finite fan-depth ledger bounds the retained prefix window and places the witness-window row after the Cantor prefix but before the uniform-modulus handoff.

Use Definition 1525.56. It routes the compact metric and uniform-continuity consumers through the witness-window handoff to UniformModulus$^{\uparrow}$, so exposing the modulus row entails exposing the paired prefix and pointwise tolerance evidence.

Finally invoke Definition 1525.57. The structural rows BHist, hsame, Cont, Pkg, and NameCert keep the totality read on the displayed packet. Coordinate exhaustion excludes selected branches, host functionals, bar-induction rows, open-cover compactness, quotient streams, and ambient Real$^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRootUnblockWitnessWindowTotality`

Theorem 1525.59 (FanFunctional finite bar-depth carrier scope)**.** *Every accepted* FanFunctional$^{\uparrow}$ *root read that exposes finite bar-depth data must route the* CantorSpace$^{\uparrow}$ *prefix row,* FanTheorem$^{\uparrow}$ *bar row,* CompactMetric$^{\uparrow}$ *finite-window consumer, and* UniformModulus$^{\uparrow}$ *target through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The carrier scope contains no choice-selected branch, host functional, open-cover compactness proof, quotient stream equality, or ambient Real-completeness principle.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalBarScopeObligations`

Proof. Project the carrier of Definition 1525.1. The Cantor-prefix row, finite fan row, branch-prefix window, bar-depth evidence, witness window, and extracted modulus are all displayed finite rows.

Use the root-unblock Cantor-prefix and finite fan-depth routes of Definition 1525.54 and Definition 1525.55. They place the CantorSpace$^{\uparrow}$ and FanTheorem$^{\uparrow}$ rows before any compact or modulus consumer can inspect the packet.

The CompactMetric$^{\uparrow}$ and UniformModulus$^{\uparrow}$ rows are then replayed through the same BHist, hsame, Cont, Pkg, and NameCert ledger. Since no coordinate contains a selected infinite branch or host functional, the finite bar-depth carrier cannot export one. □

Theorem 1525.60 (FanFunctional Cantor compactness scope)**.** *Every* FanFunctional$^{\uparrow}$ *consumer that reads Cantor compactness data reaches it through* CantorSpace$^{\uparrow}$*,* FanTheorem$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by the accepted packet. The scope is finite-prefix compactness replay and not an open-cover compactness theorem, choice-selected branch, host fan functional, quotient stream equality, or ambient real-order argument.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCantorCompactnessScope`

Proof. Read the Cantor prefix row from Definition 1525.1. It supplies finite Boolean observation windows and StreamName schedules before the fan-depth row is visible.

Apply Definition 1525.59. The finite bar-depth row bounds the retained prefix window, so the compactness-facing read is a finite-prefix replay over CantorSpace$^{\uparrow}$ and FanTheorem$^{\uparrow}$.

The compact metric consumer is admitted only through the displayed CompactMetric$^{\uparrow}$ row and the same transport, replay, provenance, and naming rows. Thus the compactness scope cannot detach into an open-cover theorem, selected branch, host functional, quotient stream equality, or ambient real-order argument. □

Theorem 1525.61 (FanFunctional uniform-modulus export scope)**.** *Every* UniformModulus$^{\uparrow}$ *row exported by an accepted* FanFunctional$^{\uparrow}$ *packet is scoped over* CantorSpace$^{\uparrow}$*,* FanTheorem$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The export is a finite row replay*

from prefix depth and witness windows, not a choice-selected branch, host functional, bar-induction proof, open-cover compactness proof, quotient stream equality, or ambient Real-completeness row.

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalUniformModulusScope`

Proof. Start with the finite bar-depth scope of Definition 1525.59. It fixes the Cantor prefix, fan-depth, compact-window, and structural rows available to a modulus export.

Use the Cantor compactness scope of Definition 1525.60. The compact metric row is consumed only as finite-prefix replay through the accepted packet.

Finally apply the root-unblock uniform-modulus handoff of Definition 1525.56. The exported $\mathsf{UniformModulus}^{\uparrow}$ row stays downstream of BHist, hsame, Cont, Pkg, and NameCert rows, so none of the excluded branch, host-functional, induction, compactness, quotient, or Real-completeness data can escape. □

Theorem 1525.62 (FanFunctional finite-window coverage)**.** *Every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window packet exposes a finite Cantor-prefix window before any* $\mathsf{UniformModulus}^{\uparrow}$ *row is read. Concretely, for the carrier* $\Omega = (B, Q, F, D, U, T, K, W, H, R, P, N)$ *of Definition 1525.33, the rows* B, Q, W *display the Cantor-prefix windows consumed by the root read before the fan bar row* F*, the finite depth row* D*, or the modulus row* M *of the root-schema consumer is exported. The coverage route is the displayed* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{UniformModulus}^{\uparrow}$ *path, not a selected infinite branch, host functional, open-cover compactness proof, or countable-choice schedule.*

Proof. Project the finite-window carrier of Definition 1525.33. Its Cantor-facing coordinates are B, Q, W, and Definition 1525.34 requires those finite rows to be visible before the fan bar, depth, or local naming rows are consumed.

Next use the root schema carrier and root NameCert acceptance, Definition 1525.38 and Definition 1525.53. These results keep the compact and uniform-modulus consumers on the same $B, Q, F, D, U, T, K, W, H, R, P, N$ route.

Finally apply the Cantor-window coverage obligations in the spine. The displayed CantorSpace, FanTheorem, UniformContinuity, CompactMetric, and UniformModulus rows exhaust the finite-window surface, so no consumer may replace them by an infinite branch selector, host functional, open-cover compactness proof, or countable-choice schedule. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalFiniteWindowCoverage`

Theorem 1525.63 (FanFunctional modulus-window monotonicity)**.** *The* $\mathsf{UniformModulus}^{\uparrow}$ *row extracted from an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window packet is read through the finite bar-depth ledger* F, D *and never through an infinite branch selector. If a later modulus request asks for a stronger displayed tolerance row, its witness-window read refines the earlier read only by extending the finite Cantor-prefix window already displayed in* B, Q, W *and by reusing the same* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Proof. Start with Definition 1525.51. It places the bar-depth row D after the Cantor-prefix and fan rows and before any uniform-modulus or uniform-continuity row is consumed.

Use Definition 1525.52 to identify the witness-window surface. A stronger displayed tolerance can only enlarge or refine the finite witness windows already carried by B, Q, W under the same finite depth row.

Finally apply Definition 1525.35 together with Definition 1525.53. The extracted modulus row is therefore read through $F, D, U, T, W, H, R, P, N$ and the compact finite-prefix row K, with no coordinate for an infinite branch selector or host compactness argument. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalModulusWindowMonotonicity`

Theorem 1525.64 (FanFunctional Cantor-prefix nonescape)**.** *All Cantor-prefix consumers of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *finite-window packet remain inside the displayed route*

$$C, F, B, D, W, M, H, K, P, N.$$

Equivalently, a consumer may read Cantor prefixes, fan bar data, retained prefix windows, finite depth, witness windows, modulus output, componentwise transport, compact finite-prefix support, provenance, and the local NameCert *row, but it cannot escape to a selected infinite branch, host functional, open-cover compactness proof, countable-choice selector, or ambient Real-completeness witness.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCantorPrefixNonescape`

Proof. Begin with the carrier and NameCert obligation theorem in the spine, Definition 1525.33 and Definition 1525.53. They display the Cantor, fan, witness, modulus, compact, transport, replay, provenance, and local naming rows before any root consumer can accept the packet.

Next apply Definition 1525.34 and Definition 1525.36. The Cantor rows are finite prefix windows, and the compact-metric consumer receives only the displayed finite-net support attached to those prefixes.

Finally use Definition 1525.26. Since selected branches, host functionals, open-cover compactness proofs, countable-choice selectors, and ambient Real-completeness witnesses have no carrier coordinate, every Cantor-prefix consumer stays inside the displayed $C, F, B, D, W, M, H, K, P, N$ route. □

Theorem 1525.65 (FanFunctional local obligation surface)**.** *For the carrier*

$$\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$$

of Definition 1525.1, the rows $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *jointly form the complete local obligation surface for* $\mathsf{FanFunctional}^{\uparrow}$*. Carrier admission is read from the displayed* BHist *packet, classifier stability is read from the Cantor-prefix and finite-fan rows, and the remaining public obligations are exactly bar-depth exactness, witness-window coverage, uniform-modulus handoff, and nonescape.*

Proof. Project the displayed carrier. The coordinates $C, F, \varepsilon, B, D, W, M$ name the finite Cantor-prefix, finite fan, tolerance, branch-window, bar-depth, witness-window, and modulus rows.

The structural coordinates H, K, P, N then close the local reading surface: H supplies componentwise hsame transport, K supplies displayed Cont replay, P records package provenance, and N records the local NameCert name. No further carrier coordinate is available to the chapter. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalLocalObligationSurface`

Theorem 1525.66 (FanFunctional bar-depth and witness-window precedence)**.** *Every* $\mathsf{FanFunctional}^{\uparrow}$ *obligation read consumes the finite bar-depth row* D *and the witness-window row* W *before any modulus row* M *is exported to a* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer. Thus finite bar-depth exactness and witness-window coverage are prerequisites for modulus consumption, not consequences of an already accepted modulus.*

Proof. Read the finite rows in carrier order. The branch window B is cut from the Cantor-prefix row C by the finite fan row F, and D certifies the displayed bar depth over that finite window.

Only after this depth row is fixed can W attach the pointwise tolerance witnesses that feed the modulus row M. Consequently a consumer of M sees the already displayed D, W surface and cannot use the modulus row to choose or repair the preceding bar-depth and witness-window data. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalBarDepthWitnessWindowPrecedence`

Theorem 1525.67 (FanFunctional modulus handoff and nonescape)**.** *The uniform-modulus handoff M, H, K, P, N of Definition 1525.1 and Definition 1525.2 excludes host functionals, selected branches, open-cover compactness, countable choice, and Real completeness from the* $\mathsf{FanFunctional}^{\uparrow}$ *public surface. Every accepted read factors through the finite carrier rows, componentwise transport, displayed continuation replay, package provenance, and the local NameCert row.*

Proof. The handoff row M is produced from the finite windows B, D, W and is transported only by the displayed structural rows H, K, P, N.

Since the carrier contains no coordinate for a host functional $2^{\mathsf{Nat}^{\uparrow}} \to \mathsf{Real}^{\uparrow}$, selected branch, open-cover compactness proof, countable-choice schedule, or $\mathsf{Real}^{\uparrow}$ completeness witness, there is no projection through which such data can enter a public read. Nonescape is therefore a finite-coordinate statement about the displayed carrier. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalModulusHandoffNonescape`

Theorem 1525.68 (FanFunctional Real and RegSeqRat scope)**.** *Every* $\mathsf{FanFunctional}^{\uparrow}$ *modulus read consumed by a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *route must first pass through* $\mathsf{CantorSpace}^{\uparrow}$ *finite prefixes,* $\mathsf{FanTheorem}^{\uparrow}$ *decidable finite-bar depth,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data,* $\mathsf{UniformContinuity}^{\uparrow}$ *local tolerance rows, and the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row. The exported route contains no host function space, selected branch, open-cover compactness, countable choice, or Real completeness.*

Proof. Project the carrier surface of Definition 1525.1. The visible route begins with the finite Cantor-prefix row C, the finite fan row F, the tolerance row ε, the branch-window row B, the bar-depth row D, the witness-window row W, and the extracted modulus row M.

Apply the local obligation surface of Definition 1525.65. It fixes carrier admission and classifier stability before any downstream consumer can read a modulus value.

Now use the bar-depth coverage theorem and the modulus nonescape theorem. The row order is

$$C, F, \varepsilon, B, D, W, M, H, K, P, N.$$

Thus a Real or regular-rational consumer receives a finite modulus route through Cantor prefixes, decidable fan depth, compact-metric finite nets, uniform-continuity tolerance rows, and the extracted UniformModulus row.

Because no carrier coordinate stores a host function space, selected branch, open-cover compactness proof, countable-choice schedule, or $\mathsf{Real}^{\uparrow}$ completeness witness, those objects cannot be exported by the scoped consumer route. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalRealRegSeqRatScope`

Theorem 1525.69 (FanFunctional scoped consumer route)**.** *Every scoped consumer of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *carrier*

$$\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N)$$

must read the finite $\mathsf{CantorSpace}^{\uparrow}$ *prefix row* C*, the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row* F*, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-window row encoded by* B, D, W*, the* $\mathsf{UniformContinuity}^{\uparrow}$ *tolerance request* ε*, and the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *on the same* BHist *route before it may use the modulus. The consumer surface is transported only by the displayed* hsame *rows* H*, replayed only by* K*, sourced only by* P*, and named only by* N*.*

Proof. Project the carrier of Definition 1525.1. The rows $C, F, \varepsilon, B, D, W, M$ are the only analytic rows visible to a consumer, and H, K, P, N are the only structural rows.

Apply the local obligation surface of Definition 1525.65. Carrier admission and classifier stability keep the Cantor-prefix and finite-fan rows in scope before any compact or uniform consumer reads M.

Use the precedence theorem Definition 1525.66. It requires the bar-depth and witness-window rows to be consumed before the modulus row is exported. Thus every scoped consumer sees the finite $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ route in the displayed order. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalScopedConsumerRoute`

Theorem 1525.70 (FanFunctional scoped choice nonescape)**.** *The scoped* $\mathsf{FanFunctional}^{\uparrow}$ *consumer route of Definition 1525.69 exports no selected branch, no host functional* $2^{\mathsf{Nat}^{\uparrow}} \to \mathsf{Real}^{\uparrow}$*, no open-cover compactness proof, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness. Any accepted read of* M *factors through the finite rows* $C, F, \varepsilon, B, D, W, M, H, K, P, N$ *and through the local* $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$ *obligations.*

Proof. The modulus row M is a coordinate of the displayed carrier, not a host functional. Projection of Definition 1525.1 gives no coordinate for selected branches, open covers, choice schedules, or ambient real completeness.

Apply Definition 1525.67. Its nonescape clause already exhausts the handoff through M, H, K, P, N and excludes the same host objects from the public surface.

The scoped consumer route adds only the destination names $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$. Since those destinations receive the same finite packet, they cannot receive any omitted host datum. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalScopedChoiceNonescape`

Theorem 1525.71 (FanFunctional scoped finite-modulus export)**.** *For every accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet, the row* M *exported to* $\mathsf{UniformModulus}^{\uparrow}$ *is a finite modulus extracted from the displayed Cantor prefix, fan-bar, compact-window, and tolerance rows. Any* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer receives* M *together with its route through* $C, F, \varepsilon, B, D, W, H, K, P, N$*, and receives no separate compactness, function-space, branch-selection, choice, or Real-completion principle.*

Proof. Read C and F first. They expose the finite Cantor-prefix and finite fan-bar surfaces that bound the branch-prefix window.

Next read ε, B, D, W. The tolerance request is attached to the finite witness windows, and the bar-depth row is already fixed before the witness rows are used.

Finally export M through H, K, P, N. Componentwise transport, displayed replay, package provenance, and the local name preserve the same finite route, so a downstream compact or uniform-continuity consumer receives only the finite modulus packet. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalScopedFiniteModulusExport`

Theorem 1525.72 (FanFunctional finite bar-modulus scope)**.** *Any scoped read of the modulus row M in an accepted* FanFunctional$^{\uparrow}$ *carrier must first expose the* CantorSpace$^{\uparrow}$ *prefix row C, the* FanTheorem$^{\uparrow}$ *finite-bar-depth row F, the* CompactMetric$^{\uparrow}$ *finite-window rows B, D, W, the* UniformContinuity$^{\uparrow}$ *tolerance row ε, and the structural* BHist, hsame, Cont, Pkg, *and* NameCert *rows H, K, P, N. A consumer that omits one of these rows has not reached the scoped modulus surface.*

Proof. Begin with the scoped consumer route of Definition 1525.69. It orders the analytic rows before the exported modulus and records the structural route by which the consumer reaches them.

Use finite-modulus export in Definition 1525.71. The row M is not independent data: it is exported only after $C, F, \varepsilon, B, D, W$ have fixed a finite prefix, bar depth, tolerance request, and witness window.

The structural rows then bind the same read to the BEDC packet. Transport through H, replay through K, provenance through P, and naming through N keep the modulus read inside the displayed BHist route, so a scoped read missing any listed row is not an accepted modulus read. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalFiniteBarModulusScope`

Theorem 1525.73 (FanFunctional compact-uniform consumer scope)**.** *No* CompactMetric$^{\uparrow}$, UniformContinuity$^{\uparrow}$, *or* UniformModulus$^{\uparrow}$ *consumer of an accepted* FanFunctional$^{\uparrow}$ *packet may replace the finite fan evidence C, F, B, D, W with open-cover compactness, selected branches, countable choice, function-space compactness, or a host modulus theorem. The consumer surface is the finite route through $C, F, \varepsilon, B, D, W, M, H, K, P, N$.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCompactUniformConsumerScope`

Proof. Read the choice nonescape theorem Definition 1525.70. It excludes selected branches, host functionals, open-cover compactness, choice schedules, and ambient Real completeness from the scoped route.

Apply finite bar-modulus scope. A compact or uniform consumer obtains M only after the finite Cantor prefix, finite bar-depth, compact-window, tolerance, witness-window, and structural rows are present.

The excluded compactness and choice principles have no carrier slot in this finite route. Therefore a consumer that uses one of them is using evidence outside the accepted FanFunctional$^{\uparrow}$ packet rather than replacing the finite fan evidence inside it. □

Theorem 1525.74 (FanFunctional compact-metric uniform bar scope)**.** *Every accepted* FanFunctional$^{\uparrow}$ *compact/uniform consumer must expose the* CompactMetric$^{\uparrow}$ *finite-net support, the* UniformContinuity$^{\uparrow}$ *tolerance data, the* FanTheorem$^{\uparrow}$ *finite bar-depth rows, and the* UniformModulus$^{\uparrow}$ *output on the same* BHist, hsame, Cont, Pkg, *and* NameCert *route before it reads the modulus row M. The accepted bar-scope read is therefore the finite route through $C, F, \varepsilon, B, D, W, M, H, K, P, N$, not an open-cover compactness proof, a selected branch family, a host functional, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalCompactMetricUniformBarScope`

Proof. Start with the scoped consumer route of Definition 1525.69. It fixes the row order in which a consumer reads the finite Cantor prefix, finite fan bar, compact-metric support, tolerance request, exported modulus, and structural rows.

Apply finite bar-modulus scope in Definition 1525.72. The bar-depth and witness-window rows must be visible before the modulus row is consumed, so the $\mathsf{FanTheorem}^{\uparrow}$ bar data cannot be delayed until after M is read.

Finally use compact-uniform consumer scope in Definition 1525.73. It excludes replacement by open-cover compactness, selected branches, host functionals, and ambient $\mathsf{Real}^{\uparrow}$ completeness, leaving exactly the displayed compact-metric finite-net support, uniform-continuity tolerance row, finite bar-depth rows, output modulus, and BEDC structural route. □

Theorem 1525.75 (FanFunctional scoped closure record)**.** *The scoped consumer route, finite bar-modulus scope, compact-uniform consumer scope, and finite-modulus export assemble a paper witness for scoped closure of* $\mathsf{FanFunctional}^{\uparrow}$ *over the displayed* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalScopedClosureRecord`

Proof. First collect the carrier-facing route from Definition 1525.69. It states which finite analytic and structural rows every scoped consumer must read.

Next apply finite bar-modulus scope and finite-modulus export. These theorems force the modulus row to remain downstream of the Cantor-prefix, fan-bar, compact-window, tolerance, and witness-window rows.

Finally apply compact-uniform consumer scope. It rules out replacement by open-cover compactness, selected branches, choice schedules, function-space compactness, or host modulus theorems. The remaining consumer surface is exactly the displayed finite packet, which is the scoped paper witness recorded here. □

Theorem 1525.76 (FanFunctional uniform-modulus compact consumer)**.** *Any* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FanFunctional}^{\uparrow}$ *packet must read* $\mathsf{CantorSpace}^{\uparrow}$ *prefix rows,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar depth,* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise tolerance rows,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data, and the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row through the finite packet*

$$\Phi = (C, F, \varepsilon, B, D, W, M, H, K, P, N).$$

Here C records Cantor prefix rows, F records the finite bar supplied by $\mathsf{FanTheorem}^{\uparrow}$*, ε records the pointwise tolerance request, B records the bar-depth boundary, D records the extracted finite depth, W records the witness windows, M records the compact finite-net and uniform-modulus handoff, and H, K, P, N record componentwise transport, displayed replay, provenance, and local naming. The consumer exports no host functional, choice-selected branch, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the existing $\mathsf{FanFunctional}^{\uparrow}$ NameCert obligations. They admit a finite packet only by displaying Cantor prefix rows, the FanTheorem bar row, tolerance data, witness windows, transport, replay, provenance, and local naming.

Read the root finite-bar depth next. The root bar-depth exactness theorem in the FanFunctional chapter forces every modulus consumer to use the finite depth displayed by the FanTheorem rows rather than a selected branch or host functional.

The compact and uniform-modulus consumers are downstream of that depth. Compact finite-net data and the extracted $\mathsf{UniformModulus}^{\uparrow}$ row are therefore read through the same packet Φ, with M as the handoff row and H, K, P, N preserving, replaying, sourcing, and naming the route.

Since no coordinate of Φ stores a host functional, branch selector, open-cover compactness proof, or $\mathsf{Real}^{\uparrow}$ completeness theorem, none of those objects can enter the compact-uniform consumer handoff. □

Lean4 ✓ `BEDC.Derived.FanfunctionalUp.FanFunctionalUniformModulusCompactConsumer`

Closure status for $\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$, publicClosed)

Scope closed: The public export surface is closed over $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert through the finite modulus route and the compact consumer of Definition 1525.76.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanFunctional}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.FanfunctionalUp.FanFunctionalScopedClosureRecord`

Bridge status: none

Not claimed: This chapter does not close a host functional, selected branch, open-cover compactness, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Bridged closure asks for a standard bridge from the finite fan and modulus packet to the public FanFunctional interface.

Constructive story: A $\mathsf{FanFunctional}^{\uparrow}$ packet is generated from finite $\mathsf{CantorSpace}^{\uparrow}$ prefix rows, a $\mathsf{FanTheorem}^{\uparrow}$ finite-bar depth ledger, compact-metric finite-net data, uniform-continuity tolerance rows, an extracted $\mathsf{UniformModulus}^{\uparrow}$ row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 1526

A Concrete Naming Certificate for $\mathsf{RiemannIntegral}^{\uparrow}$

Following Riemann and Darboux, the $\mathsf{RiemannIntegral}^{\uparrow}$ packet records a tagged-mesh integration surface whose accepted sums are routed through the Darboux upper-lower gap and dyadic step-function mesh rows. It sits after the Darboux carrier of chapter 1460, the stepwise approximation surface of chapter 428, and the dyadic mesh ledger of chapter 358. The packet names a finite constructive integration boundary over one displayed interval and integrand window; it does not assert host completeness, quotient equality of real functions, open-cover compactness, or a measure-theoretic integral.

Definition 1526.1 (Riemann integral carrier)**.** *A* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, T, F, S, D, G, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the interval endpoints, cell subdivision, mesh bound, and cell adjacency ledger.*
2. *T is the finite tag row, with each tag carried inside the corresponding mesh cell by an explicit dyadic or rational containment witness.*
3. *F is the integrand-read window, exposing only the finitely displayed values needed at the tags and at the upper/lower step approximants.*
4. *S is the finite Riemann-sum row computed from the tag values, cell widths, and displayed arithmetic ledger.*
5. *D is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *comparison row, recording upper and lower dyadic step-function approximants that bracket the same displayed reads.*
6. *G is the Darboux gap and mesh-refinement ledger that bounds the distance between the tagged sum and the Darboux interval determined by D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff surface reached only after the displayed sums and gap ledger are present.*
8. *H records componentwise* hsame *transport for mesh, tags, integrand reads, sums, Darboux comparison, gap, handoff, provenance, and the local naming row.*

9. *C records displayed* Cont *replay from mesh cells to tag reads, from tag reads to finite sums, from sums to Darboux comparison, and from the gap ledger to the real-facing handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, F, S, D, G, R, H, C, P, N$. It has no coordinate for a host function quotient, a completed real equality principle, a choice-selected tag family, open-cover compactness, a measure space, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralTasteGate single carrier alignment`

Theorem 1526.2 (Riemann integral NameCert obligations)**.** *For an accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier $I = (M, T, F, S, D, G, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}$ *surface consists of carrier admission, mesh-tag containment, finite sum exactness, Darboux bracket compatibility, gap-to-real handoff, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged by the displayed route*

$$M \longmapsto T \longmapsto F \longmapsto S \longmapsto D \longmapsto G \longmapsto R$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralPacket namecert obligations`

Proof. Carrier admission is the projection of the tuple in Definition 1526.1. The mesh row M displays the finite cells, and the tag row T records containment inside those cells before any value of the integrand is read.

Finite sum exactness is then a ledger calculation over F and S. Each summand uses only the carried tag value and the carried cell width, so the sum is a finite BHist computation rather than a host-level limiting assertion.

Darboux bracket compatibility is provided by the comparison row D. The upper and lower dyadic step-function approximants bound the same displayed reads, and the gap row G records the visible estimate between the tagged sum and the Darboux bracket.

The rows H and C transport and replay the same route componentwise. The package row P and name row N source the packet, while R is downstream of the mesh, sum, Darboux, and gap data. Since none of the excluded host objects appears as a carrier coordinate, the listed obligations exhaust the certificate surface. □

Theorem 1526.3 (Darboux and step scope of Riemann sums)**.** *Every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *read factors through a* $\mathsf{DyadicMesh}^{\uparrow}$ *row, a finite tag row, a* $\mathsf{DyadicStepFunction}^{\uparrow}$ *bracket supplied by the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *comparison row, and the displayed gap ledger before the packet can be consumed as a real-facing integral. The exported scope is therefore a finite tagged-mesh comparison surface, not a quotient class of functions and not an ambient completeness theorem.*

Proof. Project the accepted carrier through Definition 1526.1. The mesh row M supplies the finite cell surface, and the tag row T is checked against those cells before the integrand-read row F is available.

The finite sum row S is computed from those displayed reads. The comparison row D then supplies the Darboux upper and lower step-function bracket, so any step-function evidence used by the packet is already part of the visible Darboux route.

The gap row G is read after the sum and bracket rows. Thus a real-facing consumer reaches R only through the displayed mesh, tags, finite sum, Darboux bracket, and gap data. No carrier coordinate can export quotient function equality or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier darboux step scope`

Theorem 1526.4 (Riemann integral real seal boundary)**.** *An accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *packet exports to* $\mathsf{Real}^{\uparrow}$ *only the real-facing handoff row R produced from the finite route*

$$M \longmapsto T \longmapsto F \longmapsto S \longmapsto D \longmapsto G.$$

The boundary seal records the tagged-mesh sum and Darboux-gap evidence already displayed in the packet, and it does not claim host completeness, quotient real equality, open-cover compactness, or a full measure-theoretic integral.

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralPacket real seal boundary`

Proof. The handoff row R is a coordinate of the carrier, but it is downstream of the mesh, tag, read, sum, Darboux, and gap rows. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only a row whose construction has passed through the displayed finite route.

Componentwise transport through H preserves the same coordinates, and continuation replay through C rebuilds the same route when the packet is consumed. The package row P and local name N record this provenance without adding analytic structure.

The excluded objects are absent from the carrier: there is no host completion witness, no quotient equality of real functions, no open-cover compactness row, and no measure-theoretic carrier. Hence the real seal boundary is exactly the displayed tagged-mesh and Darboux-gap handoff. □

Theorem 1526.5 (Riemann integral scope closure package)**.** *Every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *read is exhausted by the dyadic mesh row, tag-containment row, integrand-read window, finite-sum row, Darboux bracket, gap ledger, regular-rational and* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert rows named in Definition 1526.1. The scope contains no quotient function equality, open-cover compactness, measure-integration carrier, or ambient completeness coordinate.*

Proof. Start from the carrier description in Definition 1526.1. Its coordinates are exactly $M, T, F, S, D, G, R, H, C, P$ so any accepted read first exposes a dyadic mesh, tag containment, integrand reads, finite sum data, Darboux comparison, gap estimate, and real-facing handoff.

Apply Definition 1526.2. The NameCert surface names the same carrier admission, mesh-tag containment, finite sum exactness, Darboux bracket compatibility, gap-to-real handoff, transport, replay, provenance, and ledger non-escape rows.

Use Definition 1526.3 and Definition 1526.4. The first confines every integration read to the tagged-mesh and Darboux-step route; the second makes the $\mathsf{Real}^{\uparrow}$ seal downstream of that finite route.

The excluded host objects have no row in the displayed carrier and cannot be introduced by H, C, P, or N. Therefore the listed rows form the scoped package for every accepted $\mathsf{RiemannIntegral}^{\uparrow}$ read. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier scope closure package`

Theorem 1526.6 (Riemann integral tagged-sum totality)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier* $I = (M, T, F, S, D, G, R, H, C, P, N)$*, the tagged-sum row* S *is total only over the displayed* $\mathsf{DyadicMesh}^{\uparrow}$ *row* M*, tag-containment row* T*, and integrand-read window* F*. A scoped consumer cannot read* S *before the mesh cells, tags, and finite values have been exposed and replayed through* H, C, P, N*.*

Proof. Project the carrier of Definition 1526.1. The rows M, T, F, S are the only tagged-sum coordinates available before the Darboux comparison row D is read.

Apply the NameCert obligations of Definition 1526.2. They place mesh-tag containment and finite sum exactness on the route from M through T, F to S.

Use the Darboux-step scope and scope-closure package, Definition 1526.3 and Definition 1526.5. These results keep every scoped integration read inside the same $\mathsf{DyadicMesh}^{\uparrow}$ and finite tagged-sum boundary before a consumer reaches $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ rows. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier tagged sum totality`

Theorem 1526.7 (Riemann integral Darboux-gap exactness)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier* $I = (M, T, F, S, D, G, R, H, C, P, N)$*, the gap ledger* G *is exact only relative to the finite tagged sum* S *and the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *comparison row* D *built from the same* M, T, F *data. The scoped packet admits no Darboux bracket or gap estimate detached from the displayed* $\mathsf{DyadicStepFunction}^{\uparrow}$ *and tagged-mesh route.*

Proof. Start with the carrier and NameCert obligations. The Darboux comparison row D is read after the tagged-sum row S, and the gap row G is read after D on the displayed route.

Apply Darboux-step scope. Theorem 1526.3 binds the step-function bracket to the same $\mathsf{DyadicMesh}^{\uparrow}$ row, tag row, integrand reads, and finite sum already displayed by the packet.

The scope-closure package then keeps the gap ledger in that chain. Since G has no independent coordinate for host function equality, open-cover compactness, measure integration, or ambient completeness, its exactness is exactly the replayed relation between S and D through H, C, P, N. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier darboux gap exactness`

Theorem 1526.8 (Riemann integral RegSeqRat handoff totality)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier* $I = (M, T, F, S, D, G, R, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row* R *is total only after the tagged-sum row* S*,* $\mathsf{DarbouxIntegral}^{\uparrow}$ *comparison row* D*, and Darboux-gap ledger* G *have been displayed. A scoped consumer reaches* R *through* $M, T, F, S, D, G, H, C, P, N$ *and cannot import a separate real-completion assertion or quotient equality of functions.*

Proof. Read the real-facing boundary through Definition 1526.4. The theorem places R downstream of the finite route from mesh and tags to finite sum, Darboux comparison, and gap ledger.

Apply the NameCert obligations and Darboux-gap exactness. The obligations expose the carried route M, T, F, S, D, G, R, while Definition 1526.7 keeps G attached to the same tagged-sum and Darboux rows before R is read.

Finally use scope closure. Theorem 1526.5 confines the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff to R with componentwise transport, replay, provenance, and local naming supplied by H, C, P, N, so the consumer receives no separate host completion, quotient function equality, open-cover compactness, or measure-integration carrier. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier regseqrat handoff totality`

Theorem 1526.9 (Riemann integral Darboux-RegSeqRat handoff)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier* $I = (M, T, F, S, D, G, R, H, C, P, N)$*, any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff read from* R *factors through the same finite* $\mathsf{DarbouxIntegral}^{\uparrow}$ *bracket row* D *and gap ledger* G *that compare the tagged sum* S *against the displayed mesh and tag data. A consumer cannot replace that route by a detached regular rational sequence, a host integral, or a completed real equality claim.*

Proof. Start with the real-seal boundary and the RegSeqRat handoff totality theorem. Theorems 1526.4 and 1526.8 place R downstream of M, T, F, S, D, G.

The Darboux bracket is fixed by the same displayed rows. Theorem 1526.3 supplies the step-function bracket row D, while Definition 1526.7 keeps the gap ledger G attached to that bracket and to the finite tagged sum S.

Finally apply the NameCert obligations and scope-closure package. Theorems 1526.2 and 1526.5 preserve the route through H, C, P, N, so a regular-rational or real-facing read has no coordinate outside the displayed Darboux and gap ledger chain. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier darboux regseqrat handoff`

Theorem 1526.10 (Riemann integral regular-Cauchy Darboux handoff)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier* $I = (M, T, F, S, D, G, R, H, C, P, N)$*, any integral-value consumer must first read the Darboux upper and lower rows in* D*, the dyadic mesh row* M*, the dyadic step-function bracket implicit in* D*, the finite tagged-sum row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback carried by* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the value is consumed. The handoff excludes selected partitions, quotient equality of functions, measure completion, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1526.1. The rows $M, T, F, S, D, G, R, H, C, P, N$ are the only source of an accepted Riemann integral read, so the mesh, tag, integrand, finite-sum, Darboux, gap, handoff, and structural data are all displayed.

Use the Darboux-step scope and Darboux-gap exactness theorems. Theorems 1526.3 and 1526.7 attach the upper and lower dyadic step-function bracket in D to the same $\mathsf{DyadicMesh}^{\uparrow}$ row M, tag row T, integrand reads F, finite sum S, and gap ledger G.

Next apply the RegSeqRat handoff and real-seal results. Theorems 1526.8, 1526.9, and 1526.4 force every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read through R after the Darboux and gap rows have already been exposed.

Finally repack by the NameCert obligations. Theorem 1526.2 supplies transport, replay, provenance, and local naming, while the carrier contains no coordinate for selected partitions, quotient function equality, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier regular cauchy darboux handoff`

Theorem 1526.11 (Riemann integral Cauchy route)**.** *Every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *read used as a Cauchy integral value factors through a finite tagged-gauge row, a Darboux upper/lower gap row, a* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap witness, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}$ *row. The route is the integration-facing path through chapter 4115, chapter 1880, and chapter 1960; it is not a prose-level integral value, a host limit, quotient equality of functions, selected partition data, or ambient real completeness.*

Proof. Project the carrier of Definition 1526.1. The tagged-mesh and finite-sum rows M, T, F, S supply the finite tagged-gauge surface, while the Darboux comparison and gap rows D, G supply the upper/lower gap evidence.

Next read the Cauchy-facing handoff. The $\mathsf{RiemannIntegrable}^{\uparrow}$ gap witness is the displayed integrability surface attached to that Darboux gap, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R is available only after the tagged-gauge, Darboux, and gap rows have been exposed.

Finally apply the existing handoff results. Theorems 1526.9 and 1526.10 keep the readback and terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite route, while Definition 1526.2 supplies hsame transport, Cont replay, Pkg provenance, and the local naming row. Since no excluded host object appears in the carrier, every Cauchy integral-value read factors through the displayed route. □

Theorem 1526.12 (Riemann integral mesh-refinement locality)**.** *Every mesh-refinement read admitted by a* $\mathsf{RiemannIntegral}^{\uparrow}$ *packet can only refine the displayed* $\mathsf{DyadicMesh}^{\uparrow}$ *row M and tag-containment row T, then replay the corresponding finite reads through $F, S, D, G, R, H, C, P, N$. It cannot introduce host function equality, a new tag family selected outside the packet, an open-cover compactness assertion, or an ambient integration theorem.*

Proof. Project the carrier through Definition 1526.1. The only mesh and tag coordinates are M and T, so any refinement request begins by replacing or subdividing those displayed rows.

Use tagged-sum totality and Darboux-step scope. Theorem 1526.6 requires S to be computed from M, T, F, and Definition 1526.3 keeps the Darboux comparison row D on the same mesh and tag route.

The gap, handoff, and structural rows then replay the refined route. Theorems 1526.7 and 1526.8 keep G and R downstream, while H, C, P, N preserve provenance and naming. Since no carrier row stores host function equality or an outside tag selector, locality is exactly the displayed refinement route. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralCarrier mesh refinement locality`

Theorem 1526.13 (Riemann integral tagged-sum ledger nonescape)**.** *For every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *carrier $I = (M, T, F, S, D, G, R, H, C, P, N)$, the tagged-sum ledger exposes only the displayed values in F, the cell widths carried by M, tag containment from T, the finite sum row S, componentwise transport H, replay C, provenance P, and local NameCert row N. The tagged sum cannot export a hidden integrand, a host equality of functions, a measure-theoretic integral, or an ambient completeness witness.*

Proof. Read the tagged-sum row first. Theorem 1526.6 makes S total only over the displayed mesh row M, tag-containment row T, and integrand-read window F.

Attach the certificate surface. Theorem 1526.2 lists finite sum exactness, transport, replay, provenance, and local naming as explicit obligations, so the ledger is exhausted by M, T, F, S, H, C, P, N before any Darboux or real-facing handoff is read.

Finally use the scope package and real-seal boundary. Theorems 1526.5 and 1526.4 exclude quotient function equality, open-cover compactness, measure-integration carriers, and ambient completeness from the packet. Hence the tagged-sum ledger has no escape beyond the displayed finite rows. □

Theorem 1526.14 (Riemann integral public tagged-mesh certificate)**.** *The public export of a* $\mathsf{RiemannIntegral}^{\uparrow}$ *packet is exactly the tagged mesh, finite tag row, integrand-read window, finite-sum ledger, Darboux bracket, Darboux-gap ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local* $\mathsf{NameCert}$ *route already scoped by this chapter. No public consumer receives a host function quotient, choice-selected tag family, open-cover compactness theorem, measure-space integral, or ambient completeness assertion.*

Proof. Project the carrier of Definition 1526.1. Its public-facing rows are $M, T, F, S, D, G, R, H, C, P, N$, so a consumer can inspect only the displayed dyadic mesh, tag containment, integrand reads, finite sum, Darboux comparison, gap ledger, handoff, transport, replay, provenance, and local naming data.

Apply the NameCert obligations, Darboux-step scope, and Real-seal boundary from Definition 1526.2, Definition 1526.3, and Definition 1526.4. These results force the route through tagged mesh, finite sum, Darboux bracket, gap, and real-facing handoff before any public read is exported.

Use the scope-closure package and tagged-sum ledger nonescape theorem, Definition 1526.5 and Definition 1526.13. They identify the same rows as the complete scoped certificate and exclude host function equality, selected tags, open-cover compactness, measure integration, and ambient completeness from the carrier. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralPacket public tagged mesh certificate`

Theorem 1526.15 (Riemann integral bridge-schema handoff)**.** *The public tagged mesh, finite tag row, integrand-read window, finite-sum ledger, Darboux bracket, Darboux-gap ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *route form the paper bridge surface for* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumers. Any standard bridge consumer must read a Riemann integral claim through exactly these displayed rows before comparing it with an outside integration presentation. The surface contains no host function quotient, no choice-selected tag family, no open-cover compactness proof, no measure-space carrier, and no ambient completeness theorem.*

Proof. Begin with Definition 1526.14. It identifies the public export of the packet with the tagged mesh, tag row, integrand reads, finite sum, Darboux bracket, gap ledger, real-facing handoff, transport, replay, provenance, and local naming data.

Next attach the finite bridge ingredients. The NameCert obligations of Definition 1526.2, the tagged-sum ledger nonescape theorem of Definition 1526.13, and the Darboux-step scope of Definition 1526.3 force every consumer-facing read through the displayed finite rows.

Finally use the real-seal and scope-closure boundaries, Definition 1526.4 and Definition 1526.5. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff inside the packet and leave no coordinate for quotient function equality, hidden tag selection, open-cover compactness, measure integration, or ambient completeness. Thus the named rows are exactly the paper bridge surface. □

Lean4 ✓ `BEDC.Derived.RiemannIntegralUp.RiemannIntegralUp StdBridge`

Theorem 1526.16 (Riemann integral mature package)**.** *The* $\mathsf{RiemannIntegral}^{\uparrow}$ *chapter exports a downstream comparison package whose rows are exactly the tagged-sum ledger, Darboux-gap ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, mesh-refinement locality route, public tagged-mesh certificate, and bridge-schema handoff displayed above. Any comparison consumer must read the route through*

$$M, T, F, S, D, G, R, H, C, P, N$$

before it compares a Riemann integral claim with another integration presentation. The package does not expose quotient equality of functions, choice-selected tags, open-cover compactness, measure-space integration, host completeness, or an additional paper-side closure obligation.

Proof. Start with the public tagged-mesh certificate of Definition 1526.14. It names $M, T, F, S, D, G, R, H, C, P, N$ as the public export surface for every $\mathsf{RiemannIntegral}^{\uparrow}$ packet.

Attach the finite computational rows. The tagged-sum totality and tagged-sum ledger nonescape theorems, Definition 1526.6 and Definition 1526.13, keep S over the displayed mesh, tag, and integrand-read rows.

Next attach the Darboux and regular-rational rows. The Darboux-gap exactness, RegSeqRat handoff totality, and Darboux-RegSeqRat handoff theorems, Definition 1526.7, Definition 1526.8, and Definition 1526.9, force G and R to remain downstream of the same tagged-mesh and Darboux bracket route.

Finally apply mesh-refinement locality and the bridge-schema handoff, Definition 1526.12 and Definition 1526.15. They make refinement and bridge comparison consume the same rows before any outside presentation is compared, while the carrier has no coordinate for the excluded host objects. Thus the listed rows form the mature downstream comparison package. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegral}^{\uparrow}$ packet is generated from a dyadic mesh row, finite tag containment row, integrand-read window, finite Riemann-sum row, Darboux comparison row, gap ledger, regular-rational and real-facing handoff, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1526.1, Definition 1526.2, Definition 1526.3, Definition 1526.4, Definition 1526.15, and Definition 1526.16 bind the carrier, NameCert obligations, Darboux-step scope, Real-seal boundary, paper bridge surface, and mature downstream comparison package for the tagged-mesh Riemann integration route over $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegral}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.RiemannIntegralUp.RiemannIntegralUp\_StdBridge`

Bridge status: bridgeChecked

Not claimed: This chapter does not close host completeness, quotient real equality, open-cover compactness, measure-theoretic integration, Lean verification, or bridge checking.

Upgrade path: No further formal-axis upgrade remains for the recorded standard bridge.

Chapter 1527

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUnit}^{\uparrow}$

Following Bishop and Bridges constructive completion, and the Kelley-style unit map from a metric source into its Cauchy completion, the $\mathsf{CauchyCompletionUnit}^{\uparrow}$ packet records the finite BEDC certificate for sending source data to the completion-facing surface. It sits over the finite stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the metric-completion boundary of chapter 1506. The packet is the unit-map handoff from displayed source rows into completion consumers; it is not a quotient completion, not a selected representative, not countable choice, not open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1527.1 (CauchyCompletionUnit carrier)**.** *A* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, S, D, R, E, M, H, C, P, N)$$

with the following visible rows:

1. *X is the source row whose displayed point, metric, or regular-Cauchy data are to be admitted into a completion-facing consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row obtained by reading the source data as a constant stream or as an already displayed finite window schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to answer finite precision requests before any completion handoff is accepted.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from S and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the displayed route through S, D, R has been admitted.*
6. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *target row that consumes the source-side route as the unit image.*
7. *H records componentwise* hsame *transport for the source, stream-window, dyadic ledger, regular readback, real seal, metric-completion target, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from source admission through constant-window generation, tolerance readback, real sealing, and completion-unit handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitTasteGate single carrier alignment`

Definition 1527.2 (CauchyCompletionUnit classifier)**.** *The classifier of two accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packets relates them exactly when their source rows agree after the displayed* hsame *transport, their finite stream windows are the same constant-window or scheduled-window reads, their dyadic tolerance ledgers and regular readbacks agree, and their completion target rows consume the same routed unit image under the stored* Cont *replay.*

Equivalently, the classifier is the finite route

$$X \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto M \longmapsto H, C, P, N.$$

It is not equality of quotient classes, not host equality of completed points, and not a choice of a canonical Cauchy representative.

Lean4 (definition) `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitClassifier`

Theorem 1527.3 (CauchyCompletionUnit NameCert obligations)**.** *For an accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *carrier* $U = (X, S, D, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnit}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed source, stream-window, dyadic ledger, regular readback, real seal, metric-completion target, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *unit-window generation, requiring each source read to enter the completion route through the constant stream or finite scheduled window stored in S;*
3. *tolerance and readback exactness, requiring every precision request to pass through D and then R before the real seal or completion target is consumed;*
4. *classifier stability, requiring transported packets with the same X, S, D, R, E, M route to be identified by the classifier of Definition 1527.2;*
5. *ledger non-escape, stating that no quotient completion equality, selected representative, countable-choice schedule, open-cover compactness principle, ambient completeness theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitNameCert obligations`

Proof. Carrier admission is projection from Definition 1527.1. The tuple lists exactly the source row, finite-window row, tolerance ledger, readback row, real seal, completion target, transport, replay, provenance, and local name.

Unit-window generation is constructive by induction on the finite request window. The empty window reads no source data, and the extension step appends the same source read to the displayed constant stream or preserves the next scheduled window already stored in S.

Tolerance and readback exactness then follow by tuple projection from D and R. The continuation row C replays this route before the seal E or completion target M can be inspected.

Classifier stability is componentwise hsame transport along H followed by the classifier definition. Since the carrier fields are exhausted by $X, S, D, R, E, M, H, C, P, N$, the excluded quotient, choice, compactness, ambient completeness, and real-completeness principles have no row from which a consumer could read them. □

Theorem 1527.4 (CauchyCompletionUnit dense embedding handoff)**.** *Every dense-embedding or completion-facing consumer of an accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packet factors through the finite unit route*

$$X \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto M.$$

The source row is therefore handed to the completion boundary as displayed finite BEDC data, not as a quotient class, not as a selected Cauchy representative, not as host equality, and not as an appeal to countable choice or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitCarrier dense embedding handoff`

Proof. Start with the source row X. A consumer cannot inspect the completion target before the unit route has displayed how the source read becomes a finite stream-window request.

The window row S is generated as a constant stream when the source read is a point input, or read as the existing finite schedule when the source is already presented by a regular-Cauchy name. Induction on the finite window gives the required handoff at each precision request.

The dyadic ledger D and regular readback R answer the request before any Real or completion-facing row is reached. The seal E is terminal for the real-facing part of the route, and M consumes only this displayed route as the unit image.

Finally H, C, P, N transport, replay, source, and name the same packet. No field contains quotient completion equality, a selected representative, host equality, countable choice, or ambient completeness, so the dense-embedding handoff exports only the finite unit route. □

Theorem 1527.5 (CauchyCompletionUnit real-completion consumer boundary)**.** *Let* $U = (X, S, D, R, E, M, H, C$ *be an accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packet. Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of* U *must read the source through* S*, account for tolerance by* D*, obtain regular rational readback by* R*, and then consume the terminal seal* E *or the completion target* M *through the same replay row* C*.*

Thus the consumer boundary is the displayed finite route S, D, R, E, M *over* X*, with componentwise transport by* H *and provenance by* P, N*. It does not close quotient completion equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitCarrier real completion consumer boundary`

Proof. Project the accepted packet and read the consumer-facing rows. The source row X is available to $\mathsf{Real}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ consumers only through the stream-window row S.

Precision requests are then routed through D and R. The dyadic ledger records the requested scale, while the regular readback row records the finite rational response supplied by the displayed source windows.

The continuation row C replays this same route to E and M. Transport by H preserves the route componentwise, and the package and local name rows P, N record that the consumer is reading the same accepted unit packet.

Since every consumer-facing read has passed through S, D, R, E, M over X, and since the carrier has no coordinate for the excluded principles, the packet supplies only the finite real-completion boundary stated above. □

Theorem 1527.6 (CauchyCompletionUnit constant-window induction)**.** *Let* $U = (X, S, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packet, and let a consumer request a finite source window from the unit route. Before any* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *row can*

be read, every admitted source row is generated either as the stored constant-window $\mathsf{StreamName}^{\uparrow}$ *row or as the already scheduled finite-window* $\mathsf{StreamName}^{\uparrow}$ *row recorded in* S*.*

Consequently the visible prefix of every consumer read has the form

$$X \longmapsto S$$

before the tolerance ledger, regular readback, real seal, or completion target is available.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitCarrier constant window induction`

Proof. Induct on the finite request window. For the empty window, no source read is exposed, so the only available consumer prefix is the stored packet projection X, S.

For the extension step, assume the claim for a finite window W. A further request appends one displayed source read. If S is the constant-window row, the same source row X is appended as the next constant stream entry. If S is a scheduled-window row, the next scheduled entry already belongs to the finite row stored in S.

In both cases the appended read is still a $\mathsf{StreamName}^{\uparrow}$ read. The tuple order in Definition 1527.1 places D, R, E, M after S, and the replay row C records the same order. Thus no dyadic ledger, regular readback, real seal, or completion target can be inspected before the finite stream-window generation step has occurred. □

Theorem 1527.7 (CauchyCompletionUnit classifier stability)**.** *Let* $U = (X, S, D, R, E, M, H, C, P, N)$ *and* $U' = (X', S', D', R', E', M', H', C', P', N')$ *be accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packets. If componentwise* hsame *transport identifies the displayed source, stream-window, dyadic ledger, regular readback, real seal, completion target, continuation replay, provenance, and local naming rows, then the classifier of Definition 1527.2 identifies* U *and* U'*.*

The transport uses only the listed tuple components. It cannot introduce quotient completion equality, selected representatives, countable-choice schedules, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the two accepted packets componentwise. The hypotheses give hsame transport for X with X', S with S', D with D', R with R', E with E', M with M', and likewise for H, C, P, N.

Apply the classifier definition. It asks exactly that source rows agree after displayed transport, stream windows agree as constant-window or scheduled-window reads, dyadic ledgers and regular readbacks agree, and the completion targets consume the same routed unit image under the stored continuation replay.

No further coordinate remains in the carrier. Since quotient completion equality, selected representatives, countable choice, compactness, ambient completeness, and real completeness are explicitly absent from Definition 1527.1, componentwise transport has no row from which any of those principles could be generated. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitCarrier classifier stability`

Theorem 1527.8 (CauchyCompletionUnit dense embedding universal route)**.** *Every dense-embedding or completion-universal-property consumer of an accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *packet factors through the finite unit route*

$$X \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto M.$$

Thus consumers of chapter 1453, chapter 1512, and chapter 397 can use the unit packet only after the source row has been routed through the stream window, tolerance ledger, regular readback, real seal, and metric-completion target displayed by the packet.

Proof. Start with a consumer asking for a dense-embedding or completion-universal-property handoff. The consumer receives only an accepted $\mathsf{CauchyCompletionUnit}^{\uparrow}$ packet, so its available data are the tuple components listed in Definition 1527.1.

The constant-window induction of Definition 1527.6 supplies the first step $X \mapsto S$. The dyadic ledger and regular readback projections then force every tolerance request through D and R before any real-facing or completion-facing row is consumed.

The terminal seal E and completion target M are reached only by the continuation replay C. Componentwise transport by H preserves the same route, while P and N record the provenance and local name of the packet being consumed.

Therefore any consumer in the cited dense-embedding, completion-embedding, or universal-property chapters factors through X, S, D, R, E, M. Since the packet has no quotient, selected-representative, countable-choice, compactness, ambient-completeness, or $\mathsf{Real}^{\uparrow}$-completeness coordinate, the factorization exports only this finite unit route. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitCarrier dense embedding universal route`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The mature finite unit-map treatment includes the carrier, classifier, NameCert obligations, dense-embedding handoff, real-completion consumer boundary, constant-window induction, classifier stability, dense-embedding universal route, constructive story, and nonescape exclusions over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnit}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.CauchyCompletionUnitUp.CauchyCompletionUnitNameCert\_obligations`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: A checked standard bridge would upgrade the verification axis for the same public unit-map certificate surface without adding quotient completion equality, selected representatives, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Constructive story: The construction starts from a displayed source row, generates finite constant or scheduled stream windows, routes tolerance requests through DyadicRatCore and RegSeqRat readback, reaches the Real seal, and hands the same route to MetricCompletion consumers as the unit image.

Chapter 1528

A Concrete Naming Certificate for $\mathsf{RegularCauchyMin}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMin}^{\uparrow}$ packet reads the pointwise minimum of two regular Cauchy real names by a finite selector over displayed rational windows. The certificate sits beside the maximum and absolute-value surfaces of chapter 1502 and chapter 1499, uses the regular rational readback of chapter 301, and seals only through the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. It does not compare completed reals by host trichotomy, quotient streams, selected limits, or ambient completeness.

Definition 1528.1 (RegularCauchyMin carrier)**.** *A* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

with the following visible coordinates:

1. *A and B are the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name rows.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to observe both sources at the same finite precision requests.*
3. *D_A and D_B are the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-and-tolerance rows read from the shared window schedule for the two sources.*
4. *J is the comparison-selector row whose finite alternatives are $A \preceq_W B$ and $B \preceq_W A$ on the displayed dyadic window data.*
5. *S is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *selector row: in the first alternative it replays the A windows, and in the second alternative it replays the B windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the selected windows and the displayed tolerance ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R has been exposed.*
8. *H records componentwise* hsame *transport for the two sources, the shared window, dyadic observations, selector row, selected regular row, readback, seal, provenance, and local name.*
9. *C records the displayed* Cont *routes from the sources to the shared windows and dyadic rows, through the selector row, through the selected regular row, and onward to the readback and real seal.*

10. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyMax}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *row.*

The carrier has no coordinate for host trichotomy on completed real names, quotient stream equality, selected limits, or a $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinTasteGate single carrier alignment`

Theorem 1528.2 (RegularCauchyMin NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *surface for Definition 1528.1 consists exactly of two-source* $\mathsf{RegSeqRat}^{\uparrow}$ *admission, shared* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance observations, the displayed comparison-selector row, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet M. The coordinates expose the two source regular names, their shared stream window, their dyadic endpoint-and-tolerance rows, the selector row, the selected regular row, the readback, the real seal, and the structural rows.

The operation obligation is finite because the only branch point is the displayed row J. In the $A \preceq_W B$ alternative, the selected row S replays the A windows and their tolerance ledger; in the $B \preceq_W A$ alternative, it replays the B windows and their tolerance ledger.

The sibling operation surfaces provide the algebraic checks used by consumers that prefer an identity route. The $\mathsf{RegularCauchyMax}^{\uparrow}$ and $\mathsf{RegularCauchyAbs}^{\uparrow}$ certificates already expose finite window transport for lattice-facing regular Cauchy operations, so the minimum packet can route compatibility checks through the same displayed-window, tolerance-ledger, readback, and seal surfaces without importing host order on completed reals.

The rows H, C, P, N transport, replay, source, and name this same finite route. No clause asks for trichotomy, quotient stream equality, a selected limiting real, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier namecert obligations`

Theorem 1528.3 (RegularCauchyMin selector ledger exactness)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier* $M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$*, every selector-facing consumer factors through exactly one of the two displayed selector alternatives in* J*. In the* $A \preceq_W B$ *case the selected regular row* S *is the finite replay of* A *on* W, D_A*; in the* $B \preceq_W A$ *case* S *is the finite replay of* B *on* W, D_B*. No further comparison data are exported.*

Proof. Begin with a selector-facing consumer of the accepted packet. The carrier gives it only the two source rows, the shared window, the two dyadic observation ledgers, and the displayed selector row.

Case-split on J. In the $A \preceq_W B$ branch, the continuation route in C points from A, W, D_A to the selected row S, and the transport row H preserves that same selected replay componentwise. Thus the readback data downstream of S are exactly the A window data recorded in the packet.

In the $B \preceq_W A$ branch, the same argument uses B, W, D_B. The selected row S is then the finite replay of the second source and its displayed tolerance ledger.

There is no third branch and no hidden comparison coordinate in Definition 1528.1. Hence the selector ledger is exhausted by the two displayed alternatives and exports no host trichotomy, quotient stream equality, selected limit, or completeness witness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinSelector ledger exactness`

Theorem 1528.4 (RegularCauchyMin selector idempotence)**.** *Let*

$$M = (A, A, W, D_A, D_A, J, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier whose two source rows are componentwise the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row over the shared* $\mathsf{StreamName}^{\uparrow}$ *window and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. Then both displayed selector alternatives replay that same source row through* W, D_A, D_A, J, S, R, E*. Consequently the minimum packet exposes the original* $\mathsf{Real}^{\uparrow}$ *seal route and no additional comparison coordinate.*

Proof. Apply Definition 1528.3. In the first selector alternative, S is the finite replay of the first source through W, D_A. Since the second source row and the second dyadic ledger are componentwise the same displayed data, this is the original regular Cauchy row.

In the second selector alternative, the same ledger identifies S as the replay of the second source through W, D_A. The displayed equality of the two source rows and ledgers makes this the same selected row, with the same readback R and terminal seal E.

Use Definition 1528.9 to keep this selected row fixed under the componentwise transport recorded by H, and use Definition 1528.7 to read the downstream order-facing route only through S, R, E. Thus idempotence is witnessed inside the existing selector ledger, not by a new host comparison coordinate. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinSelector idempotence certificate`

Theorem 1528.5 (RegularCauchyMin lattice absorption)**.** *Any lattice-facing consumer that combines the* $\mathsf{RegularCauchyMax}^{\uparrow}$ *and* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *sibling routes with an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier must read absorption through the same displayed windows* W*, dyadic ledgers* D_A, D_B*, selector row* J*, selected row* S*, readback* R*, and seal* E*. The absorption route therefore remains a finite BEDC ledger over the visible regular Cauchy data, not a host trichotomy or quotient equality principle for completed real names.*

Proof. Start with the minimum carrier and read its selector ledger. By Definition 1528.3, every branch of the minimum route is already one of the two displayed replays through W, D_A or W, D_B.

The sibling $\mathsf{RegularCauchyMax}^{\uparrow}$ and $\mathsf{RegularCauchyAbs}^{\uparrow}$ chapters supply the max and absolute-value routes used by lattice consumers, but they are consumed as provenance-facing sibling surfaces named in P, not as hidden coordinates of M. Therefore an absorption check can only combine their visible regular Cauchy windows with the minimum selector route already exposed here.

Apply Definition 1528.7 to force the order-facing read through J, S, R, E, and apply Definition 1528.9 to preserve the same branch under componentwise transport. The resulting absorption surface is the displayed-window and dyadic-ledger route shared by the three sibling operations; it does not introduce host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier lattice absorption`

Theorem 1528.6 (RegularCauchyMin selector monotonicity obligation)**.** *Let*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier. If the displayed selector row* J *chooses* A *over* B*, then every* $\mathsf{RealAlgOrder}^{\uparrow}$*-facing consumer reads a minimum row bounded by the selected source through* W, D_A, S, R, E*. If* J *chooses* B *over* A*, the same bounded route is read through* W, D_B, S, R, E*. Moreover, any componentwise transport that preserves the selected row preserves the same monotone branch for transported packets.*

Proof. Case-split on the displayed selector row J. In the $A \preceq_W B$ branch, Definition 1528.3 identifies S as the replay of the first source through the shared window W and the dyadic ledger D_A. The downstream rows R and E therefore expose a minimum row bounded by the selected source, and the order-facing consumer reaches it through the route named in Definition 1528.7.

In the $B \preceq_W A$ branch, the same ledger exactness identifies S as the replay of the second source through W, D_B. The handoff route again passes through S, R, E, so the consumer receives the second selected source as the finite minimum row.

For transported packets, apply Definition 1528.9. Componentwise transport preserves the selected row, readback, real seal, and selector alternative together. Hence the same branch remains monotone after transport, without adding host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier selector monotonicity`

Theorem 1528.7 (RegularCauchyMin RealAlgOrder handoff)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$*-facing use of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier reaches the minimum row through the finite route*

$$(A, B, W, D_A, D_B) \longrightarrow J \longrightarrow S \longrightarrow R \longrightarrow E.$$

The handoff is therefore a regular Cauchy selector ledger over shared windows and dyadic tolerance observations, not a host-order comparison of completed real names.

Proof. Start with a $\mathsf{RealAlgOrder}^{\uparrow}$-facing consumer. The only real-facing endpoint in the carrier is the seal row E, and E is downstream of the readback row R.

By Definition 1528.3, the readback row R is downstream of a selected regular row S obtained by case-splitting only on the displayed selector row J. Each selector branch uses the same shared window schedule and one of the two displayed dyadic tolerance ledgers.

The handoff to order-facing consumers is therefore the listed finite route. Compatibility with existing lattice surfaces can be checked against the sibling $\mathsf{RegularCauchyMax}^{\uparrow}$ and $\mathsf{RegularCauchyAbs}^{\uparrow}$ routes, but the minimum packet itself does not expose host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier realalgorder handoff`

Theorem 1528.8 (RegularCauchyMin carrier admission obligation)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

carrier admission is exactly the admission of the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* A, B*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D_A, D_B*, the selector row* J*, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* hsame *transport row* H*, the* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *row* N*. No admitted carrier has a further host-order, quotient-stream, selected-limit, or completeness coordinate.*

Proof. Project the displayed tuple M. The first projections expose the two regular rational source names, the shared finite window, and the two dyadic endpoint-and-tolerance ledgers.

Case-split on J. In the $A \preceq_W B$ branch, the selector admits S as the replay of the A source through W, D_A; in the $B \preceq_W A$ branch, it admits S as the replay of the B source through W, D_B.

The remaining projections expose the readback R, the terminal real seal E, the componentwise transport row H, the displayed continuation row C, the provenance row P, and the local naming

row N. This list is the full carrier inventory, so no hidden comparison or completion coordinate is available. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier admission obligation`

Theorem 1528.9 (RegularCauchyMin selector transport stability)**.** *Let*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier. Any componentwise* hsame *transport recorded in* H *preserves the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*, the downstream readback* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the same selector alternative in* J *over the transported* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Inspect the componentwise transport row H. It transports the two source rows, the shared window row, both dyadic ledgers, the selector row, the selected regular row, the readback, the seal, provenance, and local name together.

Case-split on J. In the $A \preceq_W B$ branch, transport carries the selected replay of A through the transported W, D_A rows, so S, R, and E remain downstream of the first source. In the $B \preceq_W A$ branch, the same argument carries the selected replay of B through the transported W, D_B rows.

The continuation row C replays the transported route in the same order, and P, N only record provenance and local naming. Thus selector transport is stable without adding host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier selector transport stability`

Theorem 1528.10 (RegularCauchyMin source-swap invariance)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

exchanging the two source rows A, B *together with their paired dyadic ledgers* D_A, D_B *only exchanges the two displayed selector branches. The selected row* S*, readback row* R*, and seal route* E *remain the same finite route after branch relabeling.*

Proof. By Definition 1528.3, the carrier exposes exactly the two selector alternatives listed by J: one branch replays A through W, D_A, and the other replays B through W, D_B. Swapping the paired source-and-ledger rows therefore has no third branch to enter.

Apply Definition 1528.9 to the componentwise transport that exchanges the two source rows and their dyadic ledgers while preserving the shared window, selected row, readback, seal, continuation, provenance, and local naming rows. The theorem keeps S, R, and E fixed along that transported carrier route; only the displayed name of the selector alternative changes.

Finally, Definition 1528.7 identifies the downstream order-facing path as the same finite chain through J, S, R, E. Hence the source-swap operation relabels the selector display but leaves the exported row, readback, and real seal route invariant. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier source swap invariance`

Theorem 1528.11 (RegularCauchyMin real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

reaches the seal row E only through

$$A, B, W, D_A, D_B, J, S, R.$$

The packet exports no host trichotomy on completed real names, no quotient stream equality, no selected limiting real, no hidden use of $\mathsf{RegularCauchyMax}^{\uparrow}$ *or* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *as an extra carrier coordinate, and no ambient completeness claim.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinRealSealNonescape`

Proof. Start with a Real-facing consumer. The carrier has only one Real endpoint, namely E, and the definition places E downstream of the selected regular row S and readback row R.

Case-split on J. In the first branch, S is obtained from A, W, D_A; in the second branch, S is obtained from B, W, D_B. Both branches still expose the two source rows and the shared window before the selected readback is consumed.

The sibling $\mathsf{RegularCauchyMax}^{\uparrow}$ and $\mathsf{RegularCauchyAbs}^{\uparrow}$ packets may be cited by later algebraic consumers for compatibility, but they are not additional projections of M. Since H, C, P, N only transport, replay, source, and name the same route, the Real seal cannot escape the finite selector ledger. □

Theorem 1528.12 (RegularCauchyMin obligation classifier scope)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

the classifier for accepted minimum packets is exhausted by the selected $\mathsf{RegSeqRat}^{\uparrow}$ *row S, the readback row R, and the* $\mathsf{Real}^{\uparrow}$ *seal E under the displayed selector J. No classifier-facing consumer can read an additional host-order comparison, quotient-stream equality, selected-limit row, or ambient completeness witness from the packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier obligation classifier scope`

Proof. Start with carrier admission, Definition 1528.8. It lists all projections of the displayed tuple and exposes no classifier coordinate beyond the selector, selected regular row, readback, seal, transport, replay, provenance, and local name rows.

Apply selector ledger exactness, Definition 1528.3. The selected row S is obtained by one of the two displayed alternatives in J, and each alternative replays one source row through the shared window and its dyadic tolerance ledger.

Finally use the RealAlgOrder handoff, Definition 1528.7. Any order-facing classifier read must pass through J, S, R, E, so the classifier surface is exhausted by those rows and cannot export a hidden comparison or completion datum. □

Theorem 1528.13 (RegularCauchyMin selector stability scope)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier, componentwise* hsame *transport and displayed* Cont *replay preserve the chosen selector branch, its dyadic tolerance ledger, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *row, the readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinSelector stability scope`

Proof. Use selector transport stability, Definition 1528.9. The row H transports the two sources, the shared window, the two dyadic ledgers, the selector, the selected regular row, readback, seal, provenance, and local name together.

Case-split on the displayed selector row J. In the $A \preceq_W B$ branch, transport preserves the replay through W, D_A; in the $B \preceq_W A$ branch, it preserves the replay through W, D_B.

The continuation row C replays that transported route in the same order. Hence selector stability is scoped to the displayed branch and tolerance ledger, with no extra host-order, quotient, limit, or completeness coordinate. □

Theorem 1528.14 (RegularCauchyMin dyadic-order scope)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$*-facing minimum read of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

must expose the two $\mathsf{RegSeqRat}^{\uparrow}$ *sources* A, B*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D_A, D_B*, the selector row* J*, the selected* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*, the readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before any order-facing consumer sees the minimum. The exposed route is the displayed L10 chain*

$$(A, B, W, D_A, D_B) \longrightarrow J \longrightarrow S \longrightarrow R \longrightarrow E,$$

with componentwise hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *row* N *attached to the same chain. It exports no host trichotomy, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier dyadic order scope`

Proof. Project the accepted carrier. The tuple already displays the two regular Cauchy sources, shared stream window, dyadic tolerance ledgers, selector row, selected regular row, readback, real seal, transport, replay, provenance, and local name.

Use selector ledger exactness, Definition 1528.3. It forces the selector row J to choose one of the two visible source routes and to produce S from the corresponding dyadic ledger while keeping both A and B in the exposed packet.

Apply selector transport stability, Definition 1528.13. The same branch, tolerance ledger, selected row, readback row, and real seal are preserved under componentwise hsame transport and displayed Cont replay.

Finally route the read through the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff, Definition 1528.7. Since an order-facing consumer reaches the minimum only after J, S, R, E have been displayed, and since P, N name only the listed sources, no host trichotomy, quotient stream equality, selected limit, or ambient completeness witness is exported. □

Theorem 1528.15 (RegularCauchyMin provenance scope)**.** *The* Pkg *and* NameCert *rows of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier cite only* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegularCauchyMax}^{\uparrow}$*,* $\mathsf{RegularCauchyAbs}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *as source surfaces for the finite minimum packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier provenance scope`

Proof. Read the provenance coordinate P in Definition 1528.1. Its source list is exactly the two regular rational source surfaces, the shared stream window, the dyadic tolerance rows, the sibling maximum and absolute-value operation surfaces, the real seal boundary, and the structural BHist, hsame, Cont, and NameCert rows.

The local naming coordinate N records the same packet as $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$. By Definition 1528.2, this naming row is tied to the displayed carrier inventory and not to an unlisted comparison or completion principle.

Apply real seal non-escape, Definition 1528.11. Since every real-facing read reaches E only through the displayed selector ledger, the provenance row has no hidden source beyond the listed finite surfaces. □

Theorem 1528.16 (RegularCauchyMin obligation closure route)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

the carrier admission row, selector ledger exactness, selector monotonicity, $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff, transport stability, Real-seal nonescape, classifier scope, and provenance scope exhaust the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *obligation surface. No host trichotomy on completed real names, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row remains outside those displayed obligations.*

Proof. Start with carrier admission, Definition 1528.8. It fixes the visible packet as the two source $\mathsf{RegSeqRat}^{\uparrow}$ rows, the shared $\mathsf{StreamName}^{\uparrow}$ window, the two $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, the selector row, the selected regular row, readback, seal, transport, replay, provenance, and local name.

Use selector ledger exactness and selector monotonicity, Definition 1528.3 and Definition 1528.6. They force every selector-facing read through one of the two displayed alternatives and keep the selected branch bounded along the same window and tolerance rows.

Apply the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff and transport-stability rows, Definition 1528.7 and Definition 1528.9. These place order-facing consumers downstream of J, S, R, E and preserve the same route under componentwise hsame transport and displayed Cont replay.

Finally combine real seal nonescape, classifier scope, selector stability scope, and provenance scope: Definition 1528.11, Definition 1528.12, Definition 1528.13, and Definition 1528.15. Their source lists leave no carrier coordinate for host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier obligation closure route`

Theorem 1528.17 (RegularCauchyMin shared-window scope)**.** *Every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *selector read must expose the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source readbacks* A, B*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows* D_A, D_B*, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison row used by* J*, the sibling* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *and* $\mathsf{RegularCauchyMax}^{\uparrow}$ *rows named by* P*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *row* N *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. The scope exports only this shared finite window; it does not expose quotient streams, host minimum on completed real names, trichotomy, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier shared window scope`

Proof. Project the accepted carrier $M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$. The carrier inventory already lists the two regular rational source rows, the shared window, dyadic endpoint-and-tolerance rows, selector row, selected regular row, readback row, real seal, and structural rows.

Use selector transport stability and real-seal nonescape. The transport theorem keeps W, D_A, D_B, J, S, R, E tied together under componentwise hsame, while nonescape forces every $\mathsf{Real}^{\uparrow}$-facing read of E through the displayed selector ledger.

Apply the classifier-scope and provenance-scope theorems. The classifier surface is exhausted by J, S, R, E, and the provenance row names only $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyMax}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, and NameCert surfaces.

Thus a selector consumer must expose the shared window, dyadic tolerance rows, order comparison, sibling operation rows, transport, replay, provenance, and local naming before it consumes the real seal. No quotient stream, host minimum, trichotomy, or ambient completeness row is present in the carrier. □

Theorem 1528.18 (RegularCauchyMin apartness selector scope)**.** *Every apartness-facing read of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

must pass through the shared $\mathsf{StreamName}^{\uparrow}$ *window* W*, the dyadic ledgers* D_A, D_B*, the selector row* J*, the selected row* S*, the readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. Any citation of* $\mathsf{RealAlgOrder}^{\uparrow}$ *is available only through the displayed selector route*

$$(A, B, W, D_A, D_B) \longrightarrow J \longrightarrow S \longrightarrow R \longrightarrow E.$$

Thus the packet supports apartness-facing minimum reads without exporting host trichotomy of completed reals, quotient streams, selected limits, or ambient completeness.

Proof. Start with the selector ledger exactness theorem, Definition 1528.3. It forces every selector-facing consumer to choose one of the two displayed branches in J and to read the selected row S from the shared window and the corresponding dyadic ledger.

Use selector monotonicity and the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff, Definition 1528.6 and Definition 1528.7. They place every order-facing read downstream of the same J, S, R, E chain, with the selected dyadic ledger still visible.

Apply selector stability scope and shared-window scope, Definition 1528.13 and Definition 1528.17. These keep W, D_A, D_B, J, S, R, E tied together under componentwise transport and displayed replay.

An apartness-facing consumer can therefore cite order information only through that displayed selector route. Since the carrier has no host trichotomy, quotient stream, selected-limit, or ambient-completeness coordinate, none is exported by the apartness read. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier apartness selector scope`

Theorem 1528.19 (RegularCauchyMin kernel scope route)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

every public minimum-selector read factors through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *source rows* A, B*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *order-and-tolerance rows* D_A, D_B*, the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *and* $\mathsf{RegularCauchyMax}^{\uparrow}$ *comparison siblings named by* P*, the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness-and-order handoff at* J*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *row* N *before any minimum consumer reads the result. The scoped route exports no host trichotomy on completed real names, quotient stream equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with selector ledger exactness, Definition 1528.3. It forces every minimum-selector read to inspect the two source rows, the shared window, the dyadic tolerance rows, and the selector row before the selected regular row can be consumed.

Apply the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff and dyadic-order scope, Definition 1528.7 and Definition 1528.16. These routes keep the apartness-and-order read attached to J, S, R, E and to the displayed dyadic tolerance ledgers rather than to a host trichotomy principle.

Next use shared-window scope and apartness-selector scope, Definition 1528.17 and Definition 1528.18. They keep W, D_A, D_B, J, S, R, E visible together with the sibling $\mathsf{RegularCauchyAbs}^{\uparrow}$ and $\mathsf{RegularCauchyMax}^{\uparrow}$ comparison rows named by provenance.

Finally apply selector transport stability and provenance scope, Definition 1528.9 and Definition 1528.15. The structural rows H, C, P, N transport, replay, source, and name the same finite kernel route, so a public minimum consumer cannot bypass the displayed rows into quotient stream equality, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier kernel scope route`

Theorem 1528.20 (RegularCauchyMin public NameCert export)**.** *For every accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N),$$

the carrier inventory, selector ledger exactness, idempotence, lattice absorption, selector monotonicity, $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff, classifier scope, dyadic-order scope, apartness selector scope, and kernel scope route assemble a public* $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ *export over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyMax}^{\uparrow}$, *and* $\mathsf{RegularCauchyAbs}^{\uparrow}$. *The export is exactly the displayed finite selector surface and does not introduce host trichotomy, quotient stream equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier public namecert`

Proof. Start from the carrier and obligation surface of Definition 1528.1 and Definition 1528.2. These rows display the two source $\mathsf{RegSeqRat}^{\uparrow}$ reads, the shared $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, the selector, selected readback, terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming.

Use selector ledger exactness, idempotence, lattice absorption, and selector monotonicity from Definition 1528.3, Definition 1528.4, Definition 1528.5, and Definition 1528.6. These bind the selector row to the displayed alternatives and to the sibling maximum and absolute-value surfaces rather than to a host minimum operator.

Apply the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff, classifier scope, dyadic-order scope, apartness selector scope, and kernel scope route of Definition 1528.7, Definition 1528.12, Definition 1528.16, Definition 1528.18, and Definition 1528.19. They keep every public read inside the same finite route through $D_A, D_B, W, J, S, R, E, H, C$

The public export is therefore the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ row attached to that route. Since no carrier coordinate contains host trichotomy, quotient stream equality, selected limits, or ambient completeness, the export cannot supply any of those principles. □

Theorem 1528.21 (RegularCauchyMin public selector export)**.** *Every* $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{DyadicIntervalNet}^{\uparrow}$, *or finite-net L10 consumer of an accepted* $\mathsf{RegularCauchyMin}^{\uparrow}$ *packet must read the public selector surface through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, the selector row, the selected readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal,*

and the sibling $\mathsf{RegularCauchyMax}^{\uparrow}$ *and* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *comparison routes before interval or finite-net data are exported. The export route is therefore*

$$A, B, W, D_A, D_B, J, S, R, E, H, C, P, N$$

together with the sibling comparison rows named in P*, and it contains no host trichotomy, quotient stream equality, selected limit, ambient completeness row, or hidden interval endpoint choice.*

Proof. Begin with the selector ledger and kernel scope route. By Definition 1528.3 and Definition 1528.19, every accepted selector read displays the two $\mathsf{RegSeqRat}^{\uparrow}$ sources, the shared $\mathsf{StreamName}^{\uparrow}$ window, the two $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers, the selector row J, the selected row S, the readback R, and the terminal seal E.

Next apply the public NameCert export of Definition 1528.20. It attaches H, C, P, N to the same finite selector route and keeps the sibling $\mathsf{RegularCauchyMax}^{\uparrow}$ and $\mathsf{RegularCauchyAbs}^{\uparrow}$ comparison surfaces visible as provenance-facing routes, not as unlisted carrier coordinates.

For a $\mathsf{RealInterval}^{\uparrow}$ consumer, the terminal $\mathsf{Real}^{\uparrow}$ seal can be used only after the selector route has reached E and the sibling comparison rows named by P have been exposed. The interval read therefore receives a sealed Real-name boundary, not a host-selected endpoint or a quotient stream.

For a $\mathsf{DyadicIntervalNet}^{\uparrow}$ or finite-net L10 consumer, use the finite-cell route already visible in the dyadic tolerance ledgers and shared stream window. Since $D_A, D_B, W, J, S, R, E, H, C, P, N$ are the only exported rows, the finite-net data are read after the selector surface and its sibling comparison routes, with no additional completeness or endpoint-choice principle. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier public selector export`

Theorem 1528.22 (RegularCauchyMin apartness stability)**.** *Let*

$$M = (A, B, W, D_A, D_B, J, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMin}^{\uparrow}$ *carrier. Suppose the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* A, B *are classifier-stable under the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers* D_A, D_B*, and suppose the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness rows for the two sources are transported through the sibling* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *and* $\mathsf{RegularCauchyMax}^{\uparrow}$ *routes named by* P*.*

Then the pointwise minimum packet preserves the $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and the* $\mathsf{Real}^{\uparrow}$ *apartness boundary. In the* $A \preceq_W B$ *branch it preserves the classifier and apartness boundary by replaying the* A *row through* S, R, E*; in the* $B \preceq_W A$ *branch it does the same through the* B *row. The exported route contains no host trichotomy, quotient stream equality, selected limit, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier apartness stability`

Proof. Case-split on the displayed selector row J. In the $A \preceq_W B$ branch, selector ledger exactness identifies S with the replay of the A source through W, D_A. The assumed classifier stability for A therefore transports directly to the selected $\mathsf{RegSeqRat}^{\uparrow}$ row and then to the readback R.

The apartness boundary in this branch is read through the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff and the sibling $\mathsf{RegularCauchyAbs}^{\uparrow}$ and $\mathsf{RegularCauchyMax}^{\uparrow}$ routes named by provenance. These rows are already transported componentwise, so the terminal $\mathsf{Real}^{\uparrow}$ seal E receives the same apartness boundary as the selected A row.

In the $B \preceq_W A$ branch, the same argument uses the B source and the dyadic ledger D_B. Selector ledger exactness identifies the selected row, and the transported absolute-value and maximum routes carry the apartness boundary through the same S, R, E chain.

Finally apply selector transport stability and the public NameCert export. They keep H, C, P, N attached to the displayed minimum route and prevent any hidden comparison coordinate from entering. Thus the pointwise minimum preserves both the regular Cauchy classifier and the Real apartness boundary without using host trichotomy, quotient stream equality, selected limits, or ambient completeness. □

Theorem 1528.23 (RegularCauchyMin formal target handoff)**.** *The public minimum selector surface consumed by future formal targets is exactly the finite chain through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchyMax}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *already assembled by the checked carrier, selector, handoff, stability, nonescape, scope, and export obligations of this chapter. A formal target may read the selected minimum only after the source regular names, shared window, dyadic tolerance rows, selector branch, selected readback, real seal, transport, replay, provenance, and local naming rows have all been displayed, and it receives no host trichotomy, quotient stream equality, selected limit, hidden completion row, or extra order primitive.*

Proof. Start with the public NameCert export, Definition 1528.20. It packages the two $\mathsf{RegSeqRat}^{\uparrow}$ sources, shared $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, selector row, readback, $\mathsf{Real}^{\uparrow}$ seal, and local $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$ row into one public surface.

Apply the kernel scope route of Definition 1528.19. This route inserts the sibling $\mathsf{RegularCauchyAbs}^{\uparrow}$ and $\mathsf{RegularCauchyMax}^{\uparrow}$ comparison surfaces and the $\mathsf{RealAlgOrder}^{\uparrow}$ handoff exactly where the selector branch is read, while keeping BHist, hsame, Cont, and Pkg visible as structural rows.

Finally use apartness stability, Definition 1528.22, together with real seal nonescape, Definition 1528.11. They keep the selected branch and the terminal real boundary inside the displayed chain. Since the carrier has no coordinate for host trichotomy, quotient streams, selected limits, completion, or an extra order primitive, future formal targets inherit exactly this public selector surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinUp.RegularCauchyMinCarrier formal target handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMin}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName window schedule, DyadicRatCore endpoint-and-tolerance observations, a comparison-selector row, a selected RegSeqRat row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1528.19 binds every public minimum-selector read to the displayed RegSeqRat sources, DyadicRatCore order-and-tolerance rows, RegularCauchyAbs and RegularCauchyMax siblings, RealAlgOrder handoff, StreamName window, RealUp seal, transport, replay, provenance, and local naming before any consumer reads the result.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyMin}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, host trichotomy on completed real names, quotient stream equality, selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Public closure requires a checked consumer-facing route for the same selector surface without adding host trichotomy, quotient equality, selected limits, or completeness.

Chapter 1529

A Concrete Naming Certificate for $\mathsf{CauchyProductBounded}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyProductBounded}^{\uparrow}$ packet records the bounded-product surface for two bounded Cauchy sequence packets before the product is exposed as a regular rational sequence or a real seal. It refines the product route of $\mathsf{CauchyProduct}^{\uparrow}$ from chapter 464 with the two bounded-source ledgers of $\mathsf{CauchySequenceBounded}^{\uparrow}$ from chapter 1522, then routes the result through $\mathsf{RegSeqRat}^{\uparrow}$ from chapter 301 and $\mathsf{Real}^{\uparrow}$ from chapter 38. The packet contains no host multiplication on completed reals, quotient stream representative, selected product limit, countable choice, or ambient completeness principle.

Definition 1529.1 (Cauchy product bounded carrier)**.** *A* $\mathsf{CauchyProductBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, Y, P, W, D, R, E, B, H, C, L, N)$$

with the following displayed rows:

1. *X and Y are accepted* $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packets, carrying the two stream schedules, Cauchy-modulus rows, dyadic tolerance ledgers, regular-rational handoffs, real-seal rows, and explicit bound ledgers for the factors.*

2. *P is the accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *route whose product observation is read only from the two factor schedules and dyadic product ledger.*

3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule aligning the two bounded factor windows with the product window used by P.*

4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and magnitude ledger containing the two factor bounds, the product tolerance row, and the finite cross-error budget used to compare the product read.*

5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row that receives the product readback after X, Y, P, W, D have been displayed.*

6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the product, consumed only after the bounded-source rows, product route, dyadic ledger, and regular-rational classifier row are present.*

7. *B is the product-bound ledger, recording the two source bounds, the product-bound row, and the reconciliation row saying that the product read remains inside the displayed dyadic or rational bound.*

8. *H stores componentwise* hsame *transport for $X, Y, P, W, D, R, E, B, C, L, N$ and the local packet shape.*
9. *C stores displayed* Cont *replay from bounded-source reads to product windows, from product windows to dyadic tolerances, from tolerances to the product bound, and from the bound ledger to the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows.*
10. *L is the* Pkg *provenance over* $\mathsf{CauchyProductBounded}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, P, W, D, R, E, B, H, C, L, N$. It has no coordinate for host real multiplication, quotient-stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1529.2 (Cauchy product bounded NameCert obligations)**.** *For an accepted* $\mathsf{CauchyProductBounded}^{\uparrow}$ *carrier $Z = (X, Y, P, W, D, R, E, B, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}$ *obligations are exactly carrier admission, bounded-source admission for X and Y, product-route admission for P, shared-window exactness for W, dyadic product-bound stability for D and B,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier handoff for R,* $\mathsf{Real}^{\uparrow}$ *seal handoff for E, componentwise transport and continuation replay through H and C, provenance through L, and ledger non-escape through N.*

Proof. Carrier admission is the projection of the displayed tuple in Definition 1529.1. The first two coordinates are accepted $\mathsf{CauchySequenceBounded}^{\uparrow}$ packets, so each factor already exposes a stream schedule, modulus row, dyadic tolerance ledger, regular-rational handoff, real seal row, and explicit bound ledger.

The product-route admission is the row P. It is read through the $\mathsf{CauchyProduct}^{\uparrow}$ sibling surface, so the product observation comes from the paired schedules and dyadic product ledger rather than from host multiplication over completed reals.

The shared-window row W aligns the factor reads with the product read. The dyadic ledger D and product-bound ledger B then record the finite magnitude and tolerance data needed to keep the product read inside the displayed bound.

The rows R and E are downstream handoffs. They receive only the bounded product packet assembled from X, Y, P, W, D, B; the transport rows H, continuation rows C, provenance row L, and local naming row N preserve and name that same route.

No coordinate in the carrier supplies host real multiplication, quotient streams, selected limits, trichotomy, countable choice, or ambient completeness. Hence the listed obligations exhaust the local certificate surface. □

Theorem 1529.3 (Cauchy product bounded regular-sequence route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchyProductBounded}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y \longmapsto W \longmapsto P \longmapsto D \longmapsto B \longmapsto R.$$

The two source rows are consumed as $\mathsf{CauchySequenceBounded}^{\uparrow}$ *packets, the product row is consumed as the* $\mathsf{CauchyProduct}^{\uparrow}$ *sibling route, and the regular-sequence classifier receives no quotient stream, selected product limit, host multiplication primitive, trichotomy row, countable-choice row, or ambient completeness witness.*

Proof. Begin with a consumer that asks for the regular rational product readback. By Definition 1529.1, the only regular-sequence classifier row is R.

The row R is not source data. It is downstream of the bounded factor rows X, Y, the aligned window row W, the product route P, and the dyadic product-bound ledger D, B.

The sibling $\mathsf{CauchySequenceBounded}^{\uparrow}$ packets supply the two factor bounds, while $\mathsf{CauchyProduct}^{\uparrow}$ supplies the product observation route. Combining those rows gives the displayed finite path to R.

Since no other row in the carrier is a regular-sequence classifier, every exported $\mathsf{RegSeqRat}^{\uparrow}$ read factors through that path and cannot obtain any excluded object from the packet. □

Theorem 1529.4 (Cauchy product bounded RegSeqRat window factorization)**.** *For every accepted* $\mathsf{CauchyProductBounded}^{\uparrow}$ *carrier* $Z = (X, Y, P, W, D, R, E, B, H, C, L, N)$*, any* $\mathsf{RegSeqRat}^{\uparrow}$ *read through* R *first consumes the two bounded source rows* X, Y*, then the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule* W*, then the product route* P *and the dyadic/product-bound ledgers* D, B*. The read exports no host multiplication, selected product limit, quotient stream equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier using Definition 1529.1. The rows X and Y are the only bounded source rows, and R is the only $\mathsf{RegSeqRat}^{\uparrow}$ classifier row available to a product read.

Apply Definition 1529.3. It gives the finite route

$$X, Y \longmapsto W \longmapsto P \longmapsto D \longmapsto B \longmapsto R.$$

In that route, W is the only schedule-aligning row: it is where the two bounded factor windows and the product window are displayed as one shared $\mathsf{StreamName}^{\uparrow}$ surface before P can be read.

The rows P, D, B then supply the product observation, dyadic tolerance data, and product-bound ledger consumed by R. Since the carrier has no coordinates for host multiplication, selected product limits, quotient stream equality, trichotomy, countable choice, or ambient completeness, the displayed window factorization is non-escaping. □

Theorem 1529.5 (Cauchy product bounded real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyProductBounded}^{\uparrow}$ *carrier factors through*

$$X, Y \longmapsto W \longmapsto P \longmapsto D \longmapsto B \longmapsto R \longmapsto E$$

together with the displayed hsame *transports,* Cont *replays,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}$ *row. Thus the real-facing product boundary is a finite bounded-source product certificate, not a host multiplication operation, quotient stream, selected limit, or ambient completeness theorem.*

Proof. The real-seal consumer can read only the row E as a $\mathsf{Real}^{\uparrow}$ boundary. In the carrier, E is declared downstream of the bounded-source packets, shared window row, product route, dyadic ledger, product-bound ledger, and regular-rational classifier.

Apply Definition 1529.3. The classifier row R is reached only after X, Y, W, P, D, B are displayed, so the seal row E inherits the same source and product-bound route.

The structural rows H, C, L, N transport, replay, source, and name this route without adding a second seal source. Therefore any real-facing product read is exhausted by $X, Y, P, W, D, R, E, B, H, C, L, N$.

Because the carrier has no coordinate for host multiplication, quotient streams, selected limits, trichotomy, countable choice, or ambient completeness, none of those objects can be exported by the seal boundary. □

Falsifiable prediction: Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer of a $\mathsf{CauchyProductBounded}^{\uparrow}$ packet factors through the two $\mathsf{CauchySequenceBounded}^{\uparrow}$ source ledgers, the $\mathsf{CauchyProduct}^{\uparrow}$ product

route, the shared $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{DyadicRatCore}^{\uparrow}$ product-bound ledger, the regular-rational classifier row, and the real-seal row before the product is consumed.

Independence witness: cauchysequencebounded, cauchyproduct, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyProductBounded}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two bounded Cauchy source packets, a product route, a shared schedule, a dyadic product-bound ledger, regular-rational and real-seal handoffs, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductBounded}^{\uparrow}$ packet is generated from two bounded Cauchy sequence packets, a CauchyProduct product route, a shared StreamName schedule, a DyadicRatCore product-bound ledger, a RegSeqRat classifier row, a RealUp seal row, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1529.1, Definition 1529.2, Definition 1529.3, and Definition 1529.5 seed the finite bounded-product packet for two bounded Cauchy sequence schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real multiplication, quotient stream equality, selected product limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source admission, product-route admission, shared-window exactness, dyadic product-bound stability, regular-sequence handoff, real-seal handoff, and ledger non-escape for BEDC.Derived.CauchyProductBoundedUp.

Chapter 1530

A Concrete Naming Certificate for $\mathsf{CantorSet}^{\uparrow}$

Following Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CantorSet}^{\uparrow}$ packet records the middle-third closed subset of the real line as finite ternary-prefix and deleted-middle interval data. It is a real-set certificate downstream of the nested interval discipline of chapter 320, the dyadic endpoint surface of chapter 306, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The carrier names only finite prefixes, finite gap ledgers, nested interval rows, and real-seal routing; it does not import open-cover compactness, host powersets, quotient real equality, or a completed infinite intersection.

Definition 1530.1 (Cantor set carrier)**.** *A* $\mathsf{CantorSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (T, G, I, D, R, E, H, K, P, N)$$

with the following visible rows:

1. *T is the ternary-prefix ledger, a finite word over the left and right retained thirds, together with its generation depth.*
2. *G is the deleted-middle interval ledger, listing the middle-third gaps excluded at the displayed depths.*
3. *I is the retained nested-interval row whose endpoints are determined by T and whose containment route is read through the* $\mathsf{NestedInterval}^{\uparrow}$ *surface.*
4. *D is the auxiliary* $\mathsf{DyadicInterval}^{\uparrow}$ *or rational endpoint ledger used to expose the retained interval endpoints to finite consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for a real-name readback compatible with the retained interval chain.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the ternary prefix, deleted-gap ledger, retained interval row, endpoint ledger, and regular-rational handoff have been displayed.*
7. *H stores componentwise* hsame *transport for the ternary prefix, gap ledger, interval row, endpoint ledger, handoff row, real seal, continuation replay, provenance, and local naming row.*
8. *K stores displayed* Cont *replay from prefix requests to gap checks, from gap checks to retained intervals, and from retained intervals to R and E.*
9. *P is the* Pkg *provenance over* $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CantorSet}^\uparrow}$ *row.*

The classifier accepts a packet only when the displayed prefix T selects retained thirds, the ledger G contains exactly the finite deleted-middle gaps inspected at those depths, the interval row I avoids those gaps, and every real-facing read factors through R, E, H, K, P, N. It has no coordinate for a host subset of $\mathsf{Real}^\uparrow$*, open-cover compactness, quotient real equality, selected infinite branch, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetTasteGate single carrier alignment`

Theorem 1530.2 (Cantor set NameCert obligations)**.** *For an accepted* $\mathsf{CantorSet}^\uparrow$ *carrier $C = (T, G, I, D, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorSet}^\uparrow}$ *obligations are exactly carrier admission, finite ternary-prefix classification, deleted-middle gap exclusion, nested-interval containment, endpoint-ledger compatibility,* $\mathsf{Real}^\uparrow$ *seal handoff, and non-escape. These obligations are discharged by the displayed route*

$$T \longmapsto G \longmapsto I \longmapsto D \longmapsto R \longmapsto E$$

together with H, K, P, N.

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetNameCertObligations`

Proof. Carrier admission projects the finite tuple of Definition 1530.1. The prefix classifier reads T as a finite retained-third word and therefore inspects only finitely many ternary branch decisions.

The deleted-middle exclusion row G is the only place where removed middle-third intervals are named. The retained interval row I must pass through the $\mathsf{NestedInterval}^\uparrow$ containment surface, so every accepted prefix determines a retained nested interval before any real-facing row is read.

Endpoint compatibility is checked by D. This ledger exposes the same retained interval endpoints to dyadic or rational interval consumers and prevents a later consumer from replacing the ternary interval by an unrelated host interval.

Finally, R and E are downstream readback and seal rows. The transport, continuation, provenance, and naming rows H, K, P, N preserve the same finite route, and no carrier coordinate supplies open-cover compactness, host powersets, quotient real equality, a selected infinite branch, or ambient real completeness. □

Theorem 1530.3 (Cantor set finite-prefix induction)**.** *For every accepted* $\mathsf{CantorSet}^\uparrow$ *carrier $C = (T, G, I, D, R, E, H, K, P, N)$, retained-third prefixes are generated only by finite induction over the displayed ternary-prefix ledger T. No membership-facing read may replace this finite induction with a selected infinite branch, a host powerset element, quotient real equality, open-cover compactness, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 1530.1. The row T is a finite word over retained left and right thirds together with its displayed generation depth, so its generation rule is finite ledger induction rather than branch selection.

Use Definition 1530.2. Its finite ternary-prefix classification is the first obligation read by the local $\mathsf{NameCert}_{\mathsf{CantorSet}^\uparrow}$ surface, before deleted-gap exclusion, nested-interval containment, endpoint compatibility, or Real seal handoff.

The structural rows preserve this induction route. Componentwise hsame transport in H and displayed Cont replay in K move only the finite prefix ledger named by T, while P and N record the same finite provenance and local naming. Since no carrier coordinate stores a selected infinite branch or host powerset, every retained-prefix read is generated by the displayed finite induction. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetFinitePrefixInduction`

Theorem 1530.4 (Cantor set deleted-gap ledger exhaustion)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, every deleted-middle exclusion read factors through the displayed route*

$$G \longmapsto I \longmapsto D$$

before any regular-rational handoff R *or* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. Thus the exclusion surface is exhausted by the finite gap ledger, retained interval row, and endpoint ledger, not by open-cover compactness, host powersets, quotient real equality, selected infinite branches, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier deleted gap ledger exhaustion`

Proof. Begin with the deleted-middle coordinate in Definition 1530.1. The row G is the only carrier row where removed middle-third gaps are named, and the accepted classifier requires the retained interval row I to avoid exactly the displayed gap entries at the inspected depths.

Next read the endpoint ledger D. It exposes the retained interval endpoints determined by T and checked against G, so an exclusion read cannot pass directly from a gap name to R or E without first traversing I and D.

Finally apply Definition 1530.2. Deleted-middle gap exclusion, nested-interval containment, and endpoint-ledger compatibility all precede Real seal handoff. The rows H, K, P, N transport, replay, package, and name this same exhausted route, and no excluded host-level coordinate is available. □

Theorem 1530.5 (Cantor set Real membership handoff)**.** *Every Real-facing membership consumer of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier receives only the route*

$$T, G, I, D, R, E, H, K, P, N.$$

In particular, the handoff to a $\mathsf{Real}^{\uparrow}$ *membership surface cannot bypass the ternary-prefix ledger, deleted-gap ledger, retained interval row, endpoint ledger, regular-rational readback, seal, transport, continuation replay, provenance, or local* $\mathsf{NameCert}$ *row.*

Proof. Start from the Real seal coordinate E in Definition 1530.1. It is downstream of the finite prefix row T, the deleted-gap ledger G, the retained nested interval row I, the endpoint ledger D, and the regular-rational handoff R.

Use Definition 1530.2 to identify the local obligation order. The membership-facing read reaches the Real seal only after carrier admission, prefix classification, gap exclusion, nested-interval containment, endpoint compatibility, and seal handoff have all been read on the same packet.

The same route is stable under hsame and Cont. Rows H and K preserve the componentwise transport and replay, while P and N bind the package provenance and local name. Therefore a Real-facing membership consumer receives precisely $T, G, I, D, R, E, H, K, P, N$ and no open-cover compactness, host powerset, quotient real equality, selected branch, or ambient completeness claim. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier real membership handoff`

Theorem 1530.6 (Cantor set prefix-gap separation)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, a retained-prefix read from* T *excludes each displayed deleted-middle gap row in* G *before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The exclusion is witnessed inside the route*

$$T \longmapsto G \longmapsto I \longmapsto D$$

through the retained $\mathsf{NestedInterval}^{\uparrow}$ *row and the endpoint ledger, and it has no access to host powersets, quotient real equality, selected infinite branches, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1530.1. The prefix ledger T names only retained left and right thirds, while the deleted-middle ledger G is the unique row where excluded middle-third intervals are displayed.

The retained interval row I is read before the endpoint row D and before the Real seal E. Thus a prefix request must first compare its retained interval with the displayed gap ledger, using the same $\mathsf{NestedInterval}^{\uparrow}$ containment route and the same dyadic or rational endpoint ledger.

The structural rows H, K, P, N preserve this route under hsame transport and Cont replay. Since the carrier has no host powerset, quotient real equality, selected branch, compactness, or completeness coordinate, a retained-prefix read can exclude only the displayed gap rows before it reaches the seal. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier prefix gap separation`

Theorem 1530.7 (Cantor set triadic-window split stability)**.** *Let* $C = (T, G, I, D, R, E, H, K, P, N)$ *and* $C' = (T', G', I', D', R', E', H', K', P', N')$ *be accepted* $\mathsf{CantorSet}^{\uparrow}$ *carriers. If* T *and* T' *agree on the same retained ternary prefix window through a finite depth and* G *and* G' *display the same deleted-middle gap ledger through that depth, then every downstream* $\mathsf{Real}^{\uparrow}$*-facing read through* I, D, R, E *and* I', D', R', E' *factors through the same retained interval split at that depth. The factorization uses only retained prefixes, deleted gaps, endpoint ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *sealing; it uses no selected infinite branch, host powerset, quotient real equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the two accepted carriers. The common finite prefix window is stored in T and T', while the common deleted-middle gap ledger is stored in G and G'. By finite-prefix induction, the retained left-third or right-third decisions determine the same retained interval split at the inspected depth.

Apply Definition 1530.6. The displayed gap ledgers are checked before any Real seal is consumed, so the common retained prefix excludes the same deleted gaps in both packets and routes both reads into the retained interval rows I and I'.

Now read the endpoint and Real-facing rows. The retained interval split is passed through D and D', then through the regular-rational handoff rows R and R', and finally through the $\mathsf{Real}^{\uparrow}$ seal rows E and E'. The structural rows H, K, P, N and H', K', P', N' preserve the same finite route.

Since neither carrier has a coordinate for an infinite branch selector, host powerset, quotient real equality, compactness theorem, or ambient completeness witness, the downstream reads can factor only through the displayed retained interval split. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier triadic window split stability`

Theorem 1530.8 (Cantor set ternary-prefix classifier)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, a finite prefix read is classified only by the retained left-third and right-third symbols recorded in* T *before any deleted-gap ledger or* $\mathsf{Real}^{\uparrow}$ *row is read. The classifier therefore exposes a finite ternary-retention surface and not a host powerset, quotient real equality, selected infinite branch, compactness theorem, or ambient completeness principle.*

Proof. Project the carrier of Definition 1530.1. The prefix coordinate T is the only row that records retained left and right thirds, while the deleted-gap row G and Real seal E are downstream rows.

Use Definition 1530.2. Its finite ternary-prefix classification row is read before deleted-middle exclusion, nested-interval containment, endpoint compatibility, and Real seal handoff.

Finally apply Definition 1530.6 and Definition 1530.16. They keep every gap and Real-facing read behind the displayed route from T through G, I, D, R, E, so the prefix classifier cannot supply any excluded host-level principle. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier ternary prefix classifier`

Theorem 1530.9 (Cantor set nested-interval handoff)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, the retained prefix ledger* T *and deleted-gap ledger* G *hand the consumer to the retained* $\mathsf{NestedInterval}^{\uparrow}$ *row* I *before any endpoint readback through* D, R, E *is available. The handoff is a finite row route and not a selected infinite branch, quotient real equality, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Proof. Start from the retained-prefix and gap rows in Definition 1530.1. The row I is defined as the retained nested-interval row whose endpoints are determined by T and whose avoided gaps are the displayed entries of G.

Apply Definition 1530.6. It forces the consumer to compare the retained prefix with the deleted-middle ledger before the endpoint row D and before the Real seal E.

Use Definition 1530.2 for the remaining handoff. Nested-interval containment precedes endpoint-ledger compatibility and regular-rational readback, so D, R, E can only read the retained interval already named by I. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier nested interval handoff`

Theorem 1530.10 (Cantor set ternary-prefix density)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, every retained finite ternary prefix exposes a nonempty retained interval row through* T, G, I, D *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *or* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. The density read is finite-prefix data over the displayed ledger and not a host powerset, selected infinite branch, quotient real equality, compactness theorem, or ambient completeness principle.*

Proof. Induct on the finite prefix ledger T. The empty prefix exposes the initial retained interval, and the carrier definition places that interval in the retained $\mathsf{NestedInterval}^{\uparrow}$ row I with endpoint data in D.

For an appended retained left-third or right-third symbol, read the new prefix entry before any downstream row. The deleted-gap ledger G records the middle-third exclusions inspected at that depth, while I and D display the retained subinterval and its endpoints.

Finally apply Definition 1530.9. The handoff keeps the retained interval route ahead of R and E, so each retained finite prefix exposes a nonempty interval row before any Real-facing read is available. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier ternary prefix density`

Theorem 1530.11 (Cantor set deleted-gap density refusal)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, deleted-middle gap data refines only the finite exclusion ledger* G *and its retained-interval check through* I, D*. It cannot supply host powersets, selected infinite branches, quotient real equality, open-cover compactness, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the gap coordinate in Definition 1530.1. The row G lists exactly the displayed middle-third gaps inspected at finite depths and has no coordinate for a host subset or an infinite branch selector.

Use Definition 1530.4. Each exclusion read passes through G, I, D before the regular-rational handoff or Real seal is visible.

The structural rows H, K, P, N transport, replay, source, and name that same finite route. Since the carrier lacks any powerset, quotient-equality, compactness, or completeness row, deleted-gap data can only refine the displayed finite exclusion ledger. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetDeletedGapDensityRefusal`

Theorem 1530.12 (Cantor set nested real handoff)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, the route from retained intervals to a real-facing read factors through the same* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows named in the carrier. No consumer may replace this route by a selected branch, quotient real equality, host powerset membership, compactness, or ambient completeness.*

Proof. Begin with the retained interval row I. By Definition 1530.1, its endpoints are determined by the retained prefix and deleted-gap ledgers and are exposed through the endpoint ledger D.

Next use Definition 1530.9. It forces the consumer to read T, G, I before endpoint readback is available, so the $\mathsf{NestedInterval}^{\uparrow}$ row is the entrance to the real-facing route.

Finally apply Definition 1530.16. The route from D to R and E is the displayed $\mathsf{DyadicInterval}^{\uparrow}$ or rational endpoint ledger followed by the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and terminal $\mathsf{Real}^{\uparrow}$ seal, with H, K, P, N preserving the same path. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier nested real handoff`

Theorem 1530.13 (Cantor set Real-seal membership boundary)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, every membership-facing* $\mathsf{Real}^{\uparrow}$ *seal is downstream of* $T, G, I, D, R, E, H, K, P, N$*. It cannot supply a host powerset, quotient real equality, selected infinite branch, compactness theorem, or ambient completeness principle.*

Proof. Begin with Definition 1530.16. It states that every Real-facing read factors through T, G, I, D, R, E and the structural rows H, K, P, N.

Use Definition 1530.2. The same rows cover carrier admission, prefix classification, gap exclusion, nested-interval containment, endpoint compatibility, Real seal handoff, and non-escape.

Finally apply Definition 1530.6. Since the carrier has no coordinate outside the displayed finite rows, the membership-facing seal cannot become a host powerset, quotient equality, selected branch, compactness, or completeness object. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetRealSealMembershipBoundary`

Theorem 1530.14 (Cantor set downstream membership consumer)**.** *Any downstream Real-facing closed-set or interval-membership consumer of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet must read the ternary-prefix ledger* T*, the deleted-gap ledger* G*, the retained* $\mathsf{NestedInterval}^{\uparrow}$ *row* I*, the endpoint ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the* $\mathsf{Real}^{\uparrow}$ *seal* E *before treating the packet as membership data. In particular, a locator-facing consumer such as chapter 1669 receives only this displayed finite route and not a host powerset, quotient real equality, selected infinite branch, compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the Real-seal membership boundary of Definition 1530.13. It places every membership-facing Real seal downstream of the displayed rows $T, G, I, D, R, E, H, K, P, N$.

Next use the nested-interval handoff, Definition 1530.9. The retained interval row I is available only after the prefix and gap ledgers have been read, and the endpoint ledger D is downstream of that retained interval route.

Finally apply Real seal nonescape, Definition 1530.16. It forces each Real-facing read through T, G, I, D, R, E with the same transport, replay, provenance, and local naming rows. Hence any downstream closed-set, interval-membership, or locator-facing consumer receives exactly the finite membership route and none of the excluded host-level data. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetDownstreamMembershipConsumer`

Theorem 1530.15 (Cantor set public membership route)**.** *Every public closed-set, finite-net, bounded-sequence, or locator-facing consumer of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet must read the finite route*

$$T, G, I, D, R, E, H, K, P, N$$

through $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before it treats the packet as membership data. The route exports the ternary-prefix ledger, deleted-gap ledger, retained nested interval row, endpoint ledger, regular-rational readback, terminal Real seal, transport, replay, provenance, and local naming rows, and it exports no host powerset, quotient Real equality, selected infinite branch, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the downstream membership consumer theorem, Definition 1530.14. It already places closed-set, interval-membership, and locator-facing reads downstream of the finite rows $T, G, I, D, R, E, H, K, P, N$.

Use the bounded-sequence and finite-net routes, Definition 1530.19 and Definition 1530.20. These make the bounded-window and finite-net consumers read the same ternary-prefix, retained-interval, endpoint, readback, and Real-seal route before any downstream package is visible.

Finally apply closed-interval nonescape, Definition 1530.25. The closed-set and interval consumers receive only the displayed finite rows, while the locator-facing route remains a Real-facing membership read through the same terminal seal. Hence the public membership surface is exactly the stated route through the four sibling packets and no excluded compactness or completeness datum is exported. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetPublicMembershipRoute`

Theorem 1530.16 (Cantor set Real seal nonescape)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the finite route*

$$T \longmapsto G \longmapsto I \longmapsto D \longmapsto R \longmapsto E$$

with the same transport, replay, provenance, and local name rows H, K, P, N*. No Real-facing consumer can bypass the retained-prefix ledger, deleted-gap ledger, nested interval row, endpoint ledger, regular-rational handoff, or local NameCert surface.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier real seal nonescape`

Proof. Start from a Real-facing read. By Definition 1530.1, the seal coordinate E is downstream of the ternary prefix, gap ledger, retained interval, endpoint ledger, and regular-rational handoff.

Use Definition 1530.2. Its deleted-middle exclusion, nested-interval containment, endpoint compatibility, and Real seal handoff rows require the consumer to traverse T, G, I, D, R before E is visible.

Finally read the structural coordinates. The rows H, K, P, N preserve and name the same finite route, and there is no coordinate from which a consumer could construct a bypassing seal, host subset equality, selected infinite branch, or ambient completeness proof. □

Theorem 1530.17 (Cantor set endpoint-ledger stability)**.** *Let $C = (T, G, I, D, R, E, H, K, P, N)$ and $C' = (T', G', I', D', R', E', H', K', P', N')$ be accepted* $\mathsf{CantorSet}^{\uparrow}$ *carriers whose prefix, gap, retained-interval, regular-rational, seal, provenance, and local naming rows are related by the componentwise* hsame *transports stored in H and H', and whose displayed continuations replay the same retained-prefix requests through K and K'. If the endpoint ledger D is admitted for C, then the transported endpoint ledger D' is admitted for C' and determines the same retained endpoint route for Real-facing reads.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier endpoint ledger stability`

Proof. Project the endpoint row. In an accepted carrier, D is not a free interval datum; it is read after the retained prefix T, the displayed deleted-gap ledger G, and the retained $\mathsf{NestedInterval}^{\uparrow}$ row I.

Apply the componentwise hsame transport rows. They carry the prefix, gap, retained interval, endpoint, regular handoff, seal, provenance, and local name rows together, so the endpoint ledger cannot be transported without preserving the retained interval route that generated it.

Now use the displayed Cont replay rows. The same prefix requests replay through the same gap checks and retained interval rows before the endpoint ledger is consumed. Therefore D' is admitted exactly when it continues the transported retained route, and it determines the same endpoint path for the downstream $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ reads. □

Theorem 1530.18 (Cantor set middle-third window exhaustion)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier $C = (T, G, I, D, R, E, H, K, P, N)$, every membership-facing read of the middle-third construction is exhausted by the displayed ternary prefix window T, deleted-gap ledger G, retained nested-interval handoff I, endpoint ledger D,* $\mathsf{Real}^{\uparrow}$ *handoff rows R, E, componentwise* hsame *transport H, displayed* Cont *replay K,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CantorSet}^{\uparrow}}$ *row N. No membership read may add a host powerset, quotient real equality, selected infinite branch, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct on the displayed ternary prefix window T. The empty window reads only the initial retained interval, and Definition 1530.8 places that read before any deleted-gap ledger, endpoint ledger, or Real seal is available.

For the induction step, append one retained left-third or right-third symbol to the finite window. The prefix classifier again reads the appended symbol as a retained-third decision, while Definition 1530.6 forces the corresponding deleted-middle exclusion to pass through the finite gap ledger G.

The retained interval is then handed to I by Definition 1530.9. Endpoint compatibility is preserved by Definition 1530.17, so the endpoint row D continues to determine the same retained endpoint route for the downstream regular-rational and Real-facing rows.

Finally apply Definition 1530.16. The rows R, E, H, K, P, N seal, transport, replay, source, and name exactly the finite route already constructed by the induction. Since the carrier has no coordinate outside those rows, the displayed middle-third window exhausts every membership-facing read. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier middle third window exhaustion`

Theorem 1530.19 (Cantor set bounded sequence window route)**.** *Every bounded Real sequence consumer of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier must read its sequence window through the finite ternary prefix ledger* T*, deleted-middle gap ledger* G*, retained* $\mathsf{NestedInterval}^{\uparrow}$ *row* I*, endpoint ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the same componentwise transport, replay, provenance, and local naming rows* H, K, P, N*. The read may be passed onward to a* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-net surface only after this displayed route has been consumed.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier root bounded sequence window`

Proof. Project the carrier of Definition 1530.1. The bounded-sequence consumer receives no row outside $T, G, I, D, R, E, H, K, P, N$, so the first usable data are the finite ternary prefix and deleted-gap ledgers.

Apply Definition 1530.18 to exhaust the membership-facing read by the displayed middle-third window. Then apply Definition 1530.9 and Definition 1530.12 to route the retained prefix through the $\mathsf{NestedInterval}^{\uparrow}$ row, endpoint ledger, regular-rational handoff, and terminal Real seal.

The finite-net handoff is therefore downstream of the same route. A $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ consumer may inspect the bounded window as named finite interval data, but it does not receive open-cover compactness, a selected infinite branch, quotient Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1530.20 (Cantor set finite-net subsequence refusal)**.** *For an accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier, finite-net reads over bounded sequence windows may use the displayed* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *handoff only as finite interval evidence over* $T, G, I, D, R, E, H, K, P, N$*. They cannot select an infinite subsequence branch, collapse the retained prefixes to quotient* $\mathsf{Real}^{\uparrow}$ *equality, import open-cover compactness, or appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier finite net subsequence refusal`

Proof. Begin with the finite-net request and project Definition 1530.1. The request must first traverse the finite retained-prefix ledger, gap ledger, nested interval row, endpoint ledger, regular-rational readback, and Real seal.

Use Definition 1530.19. It places the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ handoff after the Cantor finite-window route, so a finite-net read is evidence about displayed bounded cells rather than a branch selector.

Finally apply Definition 1530.13 and Definition 1530.16. These seal every Real-facing read behind the same finite rows. Hence finite-net evidence cannot become quotient Real equality, open-cover compactness, selected subsequence data, or ambient completeness. □

Theorem 1530.21 (Cantor set diagonal non-surjection route)**.** *For an accepted* $\mathsf{CantorSet}^{\uparrow}$ *carrier* $C = (T, G, I, D, R, E, H, K, P, N)$*, any consumer that passes a Cantor finite-prefix, deleted-gap, retained-interval, or endpoint route to* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *must do so through the displayed Cantor diagonal disagreement row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route of chapter 3793. The handoff may inspect* T, G, I, D, R, E *together with the structural rows* H, K, P, N*, but it cannot export a surjection witness, quotient real equality, selected infinite branch, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetRealDiagonalNonSurjectionRoute`

Proof. Start with the bounded-sequence window of Definition 1530.19. It forces every diagonal-facing read to open the finite ternary prefix, deleted-gap ledger, retained nested interval, endpoint ledger, regular readback, and Real seal before any downstream consumer sees the packet.

Next apply Definition 1530.20. The finite-net handoff remains evidence about displayed bounded cells and cannot select an infinite branch or identify retained prefixes by quotient Real equality.

Finally route the resulting finite read through chapter 3793. That sibling chapter consumes Cantor data only at its diagonal disagreement row, then reads the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seals. Since the present carrier exposes no row beyond $T, G, I, D, R, E, H, K, P, N$, the route is a Cantor-source disagreement handoff rather than a surjection, branch-selection, or completeness principle. □

Theorem 1530.22 (Cantor set root-unblock package)**.** *The accepted* $\mathsf{CantorSet}^{\uparrow}$ *NameCert package supplies the root-unblock surface needed by bounded-sequence and finite-net consumers exactly through finite prefixes, retained nested intervals, endpoint ledgers, regular-rational readback, Real seals, and the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *handoff. This package routes downstream compactness-adjacent reads without importing open-cover compactness, selected infinite branches, quotient Real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier root unblock package`

Proof. First project the local obligations of Definition 1530.2. Carrier admission, finite prefix classification, gap exclusion, nested-interval containment, endpoint compatibility, Real-seal handoff, and nonescape are the only rows exposed by the local certificate.

Next combine Definition 1530.19 with Definition 1530.20. The first theorem routes bounded Real sequence inspection through the displayed Cantor and interval rows; the second theorem keeps finite-net evidence from selecting branches or quotienting Real names.

It remains to repack the route as the downstream package. The transport, replay, provenance, and local naming rows H, K, P, N preserve the same finite route from $\mathsf{CantorSet}^{\uparrow}$ to $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$. Since no additional coordinate is available, the root-unblock package closes the stated surface without importing the refused compactness or completeness principles. □

Theorem 1530.23 (Cantor set triadic window stability)**.** *For an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet, every finite triadic window that survives the prefix classifier remains stable under the middle-third window exhaustion and endpoint-ledger transport. A membership-facing consumer may therefore read retained prefixes, deleted-gap witnesses, nested interval rows, endpoint rows, regular readback, and Real sealing, but it cannot introduce a selected infinite branch or an open-cover compactness witness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetCarrier triadic window stability`

Proof. Start with the prefix classifier from the NameCert obligations of Definition 1530.2. The classifier reads only the finite ternary prefix ledger T and the deleted-middle gap ledger G.

Apply middle-third window exhaustion from Definition 1530.18. Any membership-facing read is confined to the displayed retained prefix, gap, nested interval, endpoint, regular readback, and Real seal rows.

Use the endpoint-ledger stability supplied by the nested and Real handoffs of Definition 1530.9 and Definition 1530.12. The endpoint row is transported with the same finite prefix, so the triadic window remains the displayed finite window and not a chosen infinite branch. □

Theorem 1530.24 (Cantor set dyadic separability handoff)**.** *Every density, separability, or finite real-window consumer of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet must route its request through* $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{RealSeparability}^{\uparrow}$ *before* $\mathsf{Real}^{\uparrow}$ *membership is read. The consumer sees only the displayed triadic prefix, deleted-gap ledger, dyadic endpoint refinement, regular rational readback, finite density rows, and terminal Real seal.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetDyadicSeparabilityHandoff`

Proof. Project the carrier rows $T, G, I, D, R, E, H, K, P, N$. The dyadic endpoint information is the displayed endpoint ledger D, while R and E are the only regular readback and Real seal rows.

Use the nested interval and Real handoffs. They route retained prefixes through the dyadic or rational endpoint ledger, then through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal.

For a separability-facing read, insert the sibling finite-density surface of chapter 1809 between the regular readback and the final membership read. Since the carrier supplies no other density coordinate, the request remains a finite dyadic-separability handoff rather than a selected point or quotient-real argument. □

Theorem 1530.25 (Cantor set closed-interval nonescape)**.** *Closed bounded interval, finite-net, and bounded-sequence consumers of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet receive only the displayed prefix row* T*, gap row* G*, nested-interval row* I*, endpoint row* D*, regular readback row* R*, Real seal* E*, and structural rows* H, K, P, N*. They receive no open-cover compactness, no selected infinite branch, no quotient-real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the root-unblock package of Definition 1530.22. It packages bounded-sequence and finite-net reads exactly through finite prefixes, retained nested intervals, endpoint ledgers, regular rational readback, Real seals, and the closed bounded interval handoff.

Use Definition 1530.19 for bounded-sequence consumers and Definition 1530.20 for finite-net consumers. Both routes keep the consumer inside the displayed Cantor and interval rows.

Finally apply the Real-seal nonescape boundary of Definition 1530.16. The terminal seal prevents a closed-interval read from becoming open-cover compactness, a selected infinite branch, quotient-real equality, or an ambient completeness claim. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetClosedIntervalNonescape`

Theorem 1530.26 (Cantor set scoped nested-interval route)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ BHist *packet* $C = (T, G, I, D, R, E, H, K, P, N)$*, the scoped nested-interval route is the displayed path from the ternary-prefix row* T *and deleted-gap ledger* G *into the retained* $\mathsf{NestedInterval}^{\uparrow}$ *row* I*, then through* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint data* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface read by* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The route is preserved only by the componentwise* hsame *row* H*, displayed* Cont *replay* K*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CantorSet}^{\uparrow}}$ *row* N*.*

Proof. Project the carrier of Definition 1530.1. The BHist packet exposes finite prefix data, deleted gaps, a retained nested interval, endpoint rows, regular readback, a Real seal, and the structural rows H, K, P, N.

Apply Definition 1530.9. It places I after T and G and before any endpoint, regular-rational, or Real-facing read.

Use Definition 1530.5 to continue the route through D, R, E, H, K, P, N. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff reads the same finite $\mathsf{StreamName}^{\uparrow}$ window surface before the Real seal is consumed, so the scoped route remains a finite kernel route and not compactness, choice, quotient equality, selected branch, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetScopedNestedIntervalRoute`

Theorem 1530.27 (Cantor set scoped dyadic-branch route)**.** *Every dyadic branch read of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet* $C = (T, G, I, D, R, E, H, K, P, N)$ *is scoped by the finite rows* $T, G, I, D, R, E, H, K, P, N$*. The retained branch may be exposed to* $\mathsf{DyadicInterval}^{\uparrow}$ *consumers through* D *only after the* $\mathsf{NestedInterval}^{\uparrow}$ *row* I *has been read, and any downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer must preserve the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetScopedDyadicBranchRoute`

Proof. Start with the prefix and gap rows. The ternary-prefix classifier and prefix-gap separation results force every retained branch to be a finite retained-third read from T checked against the deleted-middle ledger G.

Next apply Definition 1530.9 and Definition 1530.24. The dyadic endpoint row D is downstream of the retained nested interval row I, and separability-facing or finite-window reads pass through the dyadic endpoint surface before regular readback and Real sealing.

Finally read the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming preserve the same finite packet; none of these rows adds compactness, branch selection, quotient equality, countable choice, or ambient Real completeness. □

Theorem 1530.28 (Cantor set scoped Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *membership or closed-interval read of an accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet is scoped by the ordered route*

$$\mathsf{BHist} \ni C = (T, G, I, D, R, E, H, K, P, N) \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRa}$$

The route supplies scoped kernel support for the existing CantorSet rows, but it does not claim compactness, selected infinite branches, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetScopedRealSealRoute`

Proof. Project the BHist packet and read the route in order. The rows T, G, I, D, R, E expose the finite Cantor data, nested interval, dyadic endpoint ledger, regular readback through finite stream windows, and terminal Real seal.

Apply Definition 1530.25. Closed-interval and finite-net consumers receive only the displayed prefix, gap, nested-interval, endpoint, readback, seal, and structural rows.

Use Definition 1530.16 for the terminal boundary. The seal cannot leave the finite window and readback route, while H, K, P, N only transport, replay, package, and name that same route.

Thus the existing obligation rows acquire scoped kernel support through the listed BEDC primitives. The conclusion is a routing statement, not a compactness theorem, choice principle, selected branch, quotient-real equality, or ambient completeness assertion. □

Theorem 1530.29 (Cantor set public Real-membership bridge)**.** *For every accepted* $\mathsf{CantorSet}^{\uparrow}$ *packet* $C = (T, G, I, D, R, E, H, K, P, N)$*, the public* $\mathsf{Real}^{\uparrow}$ *membership route is exactly the finite ternary-prefix classifier transported through the retained nested interval row* I*, the dyadic branch or endpoint data* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, with* H, K, P, N *preserving the same route. The bridge exports no compactness import, completeness import, selected infinite branch, host powerset, or quotient real equality.*

Proof. First apply the ternary-prefix classifier of Definition 1530.8. It makes membership-facing data start with the finite retained-prefix ledger T, before any deleted-gap, interval, endpoint, readback, or seal row can be consumed.

Next use the nested-interval handoff, Definition 1530.9. The retained prefix and deleted-gap rows are transported into the retained nested interval row I, and the endpoint or dyadic branch data D can be read only after that handoff.

Finally apply the scoped Real seal route, Definition 1530.28. It carries the same finite route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while the structural rows H, K, P, N only preserve and name the displayed packet. Therefore public membership is the stated finite bridge rather than an appeal to compactness, completeness, branch selection, host powersets, or quotient equality. □

Lean4 ✓ `BEDC.Derived.CantorSetUp.CantorSetPublicRealMembershipBridge`

Falsifiable prediction: Every accepted $\mathsf{CantorSet}^{\uparrow}$ real-facing read factors through the displayed ternary-prefix ledger, deleted-middle interval ledger, retained nested-interval row, endpoint ledger, regular-rational handoff, and $\mathsf{Real}^{\uparrow}$ seal before the set membership surface is consumed.

Independence witness: nestedinterval, dyadicinterval, regseqrat, real, cantorspace: $\mathsf{CantorSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a ternary retained-prefix ledger, deleted-middle gap ledger, retained nested interval row, endpoint compatibility row, regular-rational handoff, real seal, transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSet}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorSet}^{\uparrow}$ packet is generated from a finite ternary retained-prefix ledger, a deleted-middle gap ledger, a retained NestedInterval row, a DyadicInterval or rational endpoint ledger, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorSet}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1530.1, Definition 1530.2, finite-prefix induction, deleted-gap ledger exhaustion, Real membership handoff, prefix-gap separation, nested-interval handoff, Real-seal nonescape, Definition 1530.15, and Definition 1530.29 bind the finite-prefix, deleted-gap, Real-membership, product, boundary, and cross-chapter rows for the mature paper treatment.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close located compactness, open-cover compactness, host powerset equality, quotient real equality, selected infinite branches, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Further verification requires Lean coverage for the displayed mature paper route; compactness, completeness, selected branches, host powersets, and quotient real equality remain outside the chapter.

Chapter 1531

A Concrete Naming Certificate for $\mathsf{PrecompactMetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis, where precompact metric data is read through explicit finite nets rather than through open-cover compactness, the $\mathsf{PrecompactMetric}^{\uparrow}$ packet records the finite bridge from total boundedness toward completion-facing Cauchy filters. It connects the net surface of $\mathsf{TotallyBounded}^{\uparrow}$ (chapter 127) with the finite filterbase and regular-filter surfaces of $\mathsf{CauchyFilter}^{\uparrow}$ (chapter 333), $\mathsf{RegularCauchyFilter}^{\uparrow}$ (chapter 1445), and $\mathsf{CauchyFilterBasis}^{\uparrow}$ (chapter 1501), using $\mathsf{RegularityModulus}^{\uparrow}$ (chapter 353) as the visible rate ledger.

Definition 1531.1 (Precompact metric finite packet carrier)**.** *A* $\mathsf{PrecompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, D, N, F, R, M, H, C, G, Q)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row, including the accepted point histories and the metric-distance classifier rows used by downstream reads;*
2. *D records the metric carrier comparison data, so every endpoint, distance history, and rational bound used below is transported by displayed* hsame *rows and by the metric classifier;*
3. *N assigns to each positive* $\mathsf{Rat}^{\uparrow}$ *tolerance history ε a closed finite* $\mathsf{ProbeBundle}$ *of center names, visible* $\mathsf{InBundle}$ *membership rows, center-carrier rows, distance-witness rows, and rational comparison ledgers, as in the totally bounded carrier of Definition 127.3;*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *inspection row exposing the finite filterbase packets, refinement rows, regular Cauchy observation rows, dyadic neighbourhood witnesses, and precision-to-tail reads;*
5. *R is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *inspection row naming the regular tail windows and their componentwise regularity checks;*
6. *M is a* $\mathsf{RegularityModulus}^{\uparrow}$ *row whose dyadic windows control the net tolerance and the Cauchy-filter tail requests through the same displayed rate ledger;*
7. *H is the evidence ledger of componentwise* hsame *transports for X, D, N, F, R, M and for every center, tail, distance, dyadic radius, and regular readback row exposed by those coordinates;*
8. *C is the finite* Cont *route ledger: net reads route from N to X, D, M, filter reads route from F, R to M, and basis reads route through the finite generator of* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *before any completion-facing consumer can inspect them;*

9. *G is the handoff row saying that a* $\mathsf{TotallyBounded}^{\uparrow}$ *consumer reads only the finite net rows in N, the metric rows in X, D, and the modulus row in M;*

10. *Q is the* Pkg *provenance and local* NameCert *row naming the source boundaries* $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, *and* $\mathsf{RegularityModulus}^{\uparrow}$.

The classifier on such packets is componentwise transport of these ten rows through H and the displayed Cont *routes in C. No host finite set, selected sequence, quotient completion, ambient complete space, or choice operator is a coordinate of the carrier.*

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetricTasteGate single carrier alignment`

Theorem 1531.2 (PrecompactMetricUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PrecompactMetric}^{\uparrow}}$ *surface of Definition 1531.1 consists exactly of five obligations:*

1. *carrier admission: the ten displayed rows $P = (X, D, N, F, R, M, H, C, G, Q)$ are present and every row cited by a consumer is one of their finite subrows;*

2. *classifier transport: packet equality is componentwise* hsame *transport of metric endpoints, center histories, filterbase tails, regular observations, dyadic radii, modulus windows, provenance rows, and local naming rows, with every read following a displayed* Cont *route;*

3. *finite-net exactness: for each positive tolerance history, Definition 127.3 supplies the closed* ProbeBundle*, membership constructors, center rows, distance witnesses, and rational comparison ledgers recorded in N;*

4. *Cauchy-filter inspection: the rows F and R expose only finite filterbase packets, regular tail windows, refinement ledgers, regular Cauchy readbacks, and dyadic neighbourhood witnesses, with the rate read controlled by M;*

5. *provenance and handoff: Q names the source certificates and G states that every total-boundedness or completion-facing consumer reaches the packet through the displayed finite net, filter, regularity, and modulus rows.*

These obligations define a constructive precompact metric packet as a finite-row interface between total boundedness and Cauchy-filter completion data.

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetricNameCert obligations`

Proof. Carrier admission is Definition 1531.1. It lists the metric row, finite-net row, Cauchy-filter inspection row, regular-filter inspection row, modulus row, transport ledger, route ledger, handoff row, provenance row, and local naming row.

Classifier transport is also read from the carrier. The evidence ledger H moves each displayed history by hsame, while C records the continuation route used by each net, filter, basis, and modulus read. Thus two packets classify together only when the same finite rows can be replayed after transport.

Finite-net exactness is supplied by the $\mathsf{TotallyBounded}^{\uparrow}$ component named in N. The packet keeps the closed ProbeBundle spine and its InBundle constructors visible, so a consumer never replaces the net by an ambient finite set of representatives.

Cauchy-filter inspection is the combined role of F, R, and M. The filter rows expose finite tails, refinements, regular Cauchy observations, dyadic neighbourhoods, and regularity windows; the modulus row supplies the shared rate ledger that connects a net tolerance with a later Cauchy-tail request.

Finally, Q and G package the same rows as the local NameCert surface. They do not add a completed point, a quotient real, or a selected subsequence; they merely state which source certificates and which finite rows may be read by downstream consumers. □

Theorem 1531.3 (PrecompactMetricUp finite-net root obligations)**.** *For every accepted* PrecompactMetric$^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$*, the root finite-net obligations are exactly the carrier admission of* X, D, N, M, H, C, G, Q*, the* TotallyBounded$^{\uparrow}$ *finite-net rows in* N*, the* RegularityModulus$^{\uparrow}$ *tolerance ledger in* M*, and the provenance row* Q*. A downstream total-boundedness consumer may read the closed* ProbeBundle *nets,* InBundle *memberships, center histories, metric-distance witness rows, and rational comparison ledgers only through those displayed coordinates.*

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetricCarrier root finite net obligations`

Proof. Project the carrier of Definition 1531.1. The finite-net coordinate is N, and the only metric, tolerance, transport, route, handoff, provenance, and naming rows available with it are X, D, M, H, C, G, Q.

Apply Definition 1531.2. Its finite-net exactness clause identifies the TotallyBounded$^{\uparrow}$ net surface with the closed ProbeBundle, membership constructors, center rows, distance witnesses, and rational comparison ledgers already recorded in N.

The same theorem's carrier-admission, classifier-transport, and provenance clauses keep the read inside X, D, N, M, H, C, G, Q. Thus root finite-net obligations expose total-boundedness data as finite rows and do not export open-cover compactness, selected subsequences, quotient completion, host choice, or ambient completeness. □

Theorem 1531.4 (PrecompactMetricUp Cauchy-filter handoff obligations)**.** *For every accepted* PrecompactMetric$^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$*, every Cauchy-filter or regular-Cauchy-filter handoff reads* F *and* R *only after the metric rows* X, D*, finite-net row* N*, regularity-modulus row* M*, transport row* H*, continuation row* C*, handoff row* G*, and provenance row* Q *have fixed the same packet. The handoff reaches the* CauchyFilter$^{\uparrow}$*,* RegularCauchyFilter$^{\uparrow}$*, and* CauchyFilterBasis$^{\uparrow}$ *sibling surfaces as finite inspection rows, not as a completed point or quotient filter.*

Proof. Start from Definition 1531.1. The rows F and R are the only Cauchy-filter and regular-filter inspection coordinates, while C records that basis reads route through the finite generator of CauchyFilterBasis$^{\uparrow}$.

Use Definition 1531.2. Its Cauchy-filter inspection clause says that F and R expose finite filterbase packets, regular tail windows, refinement ledgers, regular Cauchy readbacks, and dyadic neighbourhood witnesses, all controlled by M.

The carrier-admission and provenance clauses force the same packet rows $X, D, N, F, R, M, H, C, G, Q$ to remain visible for the handoff. Therefore no downstream consumer can detach a completed point, a quotient filter, a selected subsequence, or ambient completeness from the Cauchy-filter route. □

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric cauchy filter handoff obligations`

Theorem 1531.5 (PrecompactMetricUp filter-net modulus handoff)**.** *For every accepted* PrecompactMetric$^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$*, a completion-facing handoff that consumes both finite nets and Cauchy-filter tails must read the finite-net coordinate* N*, the Cauchy-filter coordinates* F, R*, and the* RegularityModulus$^{\uparrow}$ *coordinate* M *inside the same packet before it can expose any basis row. The handoff is replayed only through* H, C, G, Q*, so the finite-net radius, filter tail, and modulus window are one displayed NameCert route rather than a quotient completion, selected subsequence, host finite set, or ambient complete space.*

Proof. Project the packet $P = (X, D, N, F, R, M, H, C, G, Q)$. The row N is the only finite-net coordinate, F and R are the only Cauchy-filter and regular-Cauchy-filter coordinates, and M is the only visible regularity-modulus coordinate.

Use Definition 1531.3 for the root net surface. It identifies the readable net data with the closed ProbeBundle rows, InBundle memberships, center histories, metric-distance witnesses, and rational comparison ledgers stored in N.

Apply Definition 1531.4 to the filter side. It keeps every filterbase packet, regular tail window, refinement ledger, regular Cauchy readback, and dyadic neighbourhood witness inside F, R and under the same modulus row M.

The remaining rows H, C, G, Q provide transport, replay, handoff, and provenance. Since these rows mention only the displayed carrier coordinates, the combined handoff exports no quotient completion, selected subsequence, host finite set, or ambient complete space. □

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric filter net modulus handoff`

Theorem 1531.6 (PrecompactMetricUp completion consumer non-escape)**.** *Let $P = (X, D, N, F, R, M, H, C, G, Q$ be an accepted* PrecompactMetric$^{\uparrow}$ *packet. Any completion-facing consumer that reads P can inspect only the metric rows X, D, finite-net row N, Cauchy-filter rows F, R, regularity-modulus row M, transport and route rows H, C, handoff row G, and provenance row Q. In particular, such a consumer cannot obtain a quotient completion point, a selected convergent subsequence, an external finite set of centers, a host complete-space witness, or a choice operator from the packet.*

Proof. Begin with carrier admission in Definition 1531.1. The carrier has exactly the ten coordinates $X, D, N, F, R, M, H, C, G, Q$, so every admissible consumer read must name one of those rows or a finite subrow displayed by them.

The finite-net part is controlled by Definition 1531.3. It exposes centers and distance witnesses only as ProbeBundle and InBundle data recorded in N, not as an external host finite set.

The filter part is controlled by Definition 1531.4. It exposes Cauchy-filter and regular-filter data only through F, R with the shared modulus row M, not through a completed point, quotient filter, or selected subsequence.

Finally H, C, G, Q only transport, replay, hand off, and name these same displayed rows. Therefore no completion-facing consumer can escape from the finite NameCert surface to a quotient completion, host finite set, selected sequence, ambient complete space, or choice operator. □

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric completion consumer nonescape`

Theorem 1531.7 (PrecompactMetricUp regular filter finite-net route)**.** *For every accepted* PrecompactMetric$^{\uparrow}$ *packet $P = (X, D, N, F, R, M, H, C, G, Q)$, the regular-filter route from finite nets to completion-facing basis data is the displayed path*

$$N \longrightarrow M \longrightarrow F, R \longrightarrow C, G, Q$$

inside the fixed metric context X, D, H. The route first reads the finite net row, then the regularity-modulus row, then the Cauchy-filter and regular-Cauchy-filter rows, and finally the continuation, handoff, and provenance rows; it does not pass through quotient completion, selected subsequence, host finite-set enumeration, or ambient completeness.

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetricRegularFilterFiniteNetRoute`

Proof. Read the metric context first. The rows X, D, H fix the metric carrier, metric comparison data, and componentwise transport for every later net, filter, and modulus read.

Next use the finite-net theorem Definition 1531.3. It makes N the displayed source of the finite centers, membership rows, distance witnesses, and rational comparison ledgers used by the route.

Then use the modulus and filter handoff theorems Definition 1531.8 and Definition 1531.5. The row M scopes the tolerance request shared by N and the filter rows, while F, R expose only finite filterbase and regular-tail inspection data.

The route terminates through C, G, Q. These rows replay the path, hand it to the relevant total-boundedness or completion-facing consumer, and record provenance without adding quotient completion, a selected subsequence, host finite-set enumeration, or ambient completeness. □

Theorem 1531.8 (PrecompactMetricUp regularity-modulus scope)**.** *For every accepted* PrecompactMetric$^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$*, the regularity-modulus row* M *is scoped simultaneously over finite-net tolerance requests in* N *and Cauchy-filter tail requests in* F, R*. The scope is transported through* H*, replayed through* C*, handed to total-boundedness consumers through* G*, and named by* Q*; it does not introduce an independent convergence rate, ambient compactness theorem, choice-selected tail, or completed-space coordinate.*

Proof. Read M from Definition 1531.1. The carrier says that its dyadic windows control both the net tolerance and the Cauchy-filter tail requests through one displayed rate ledger.

Apply Definition 1531.2. Finite-net exactness uses M with N, and Cauchy-filter inspection uses M with F and R; the provenance and handoff clauses keep the same row available to G and Q.

Finally use classifier transport from the same theorem. The row H transports modulus windows componentwise and C records the replay route, so the modulus scope is local to $P = (X, D, N, F, R, M, H, C, G, Q)$ and exports no independent rate object or completed ambient space. □

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric regularity modulus scope`

Theorem 1531.9 (PrecompactMetricUp totally bounded handoff)**.** *Every* TotallyBounded$^{\uparrow}$ *consumer of an accepted* PrecompactMetric$^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$ *reads only the displayed finite net, modulus, and filter-inspection rows:*

$$X, D, N, M, H, C, G, Q.$$

In particular, the consumer may inspect the metric carrier, the closed ProbeBundle *nets, the* InBundle *membership rows, the center histories, the* MetricDistanceWitness *rows, the rational bound ledgers, the* RegularityModulus$^{\uparrow}$ *windows, the componentwise* hsame *transports, the displayed* Cont *routes, and the provenance row, but it cannot read an open-cover compactness proof, a selected subsequence, a quotient completion, a host choice function, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric totally bounded handoff`

Proof. Project the handoff coordinate G from Definition 1531.1. It explicitly names the TotallyBounded$^{\uparrow}$ read route through the metric rows X, D, the finite-net row N, the modulus row M, the transport ledger H, the continuation ledger C, and the provenance row Q.

Now replay the route. A total-boundedness request first reads the accepted metric carrier X and its classifier data D. It then selects a positive tolerance history and follows C into the finite net row N, where the closed ProbeBundle, membership constructors, center rows, distance witnesses, and rational comparisons are displayed.

The only rate information available to the consumer is the $\mathsf{RegularityModulus}^{\uparrow}$ row M. The same row also governs the later $\mathsf{CauchyFilter}^{\uparrow}$ and $\mathsf{RegularCauchyFilter}^{\uparrow}$ inspection coordinates, but those coordinates are read only as finite filterbase and regular-tail evidence; they do not expose a completed object.

Therefore the handoff is exhausted by the displayed finite rows X, D, N, M, H, C, G, Q. The excluded objects are absent from the carrier coordinates, absent from the route ledger, and absent from the provenance row, so they cannot be produced by this local NameCert read. □

Theorem 1531.10 (PrecompactMetricUp bridged completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{PrecompactMetric}^{\uparrow}$ *packet* $P = (X, D, N, F, R, M, H, C, G, Q)$ *bridges to downstream completion data only through the displayed rows*

$$X, D, N, F, R, M, H, C, G, Q.$$

The bridge may read metric carrier data, finite-net rows, Cauchy-filter rows, regular-filter rows, regularity-modulus windows, componentwise transport, continuation replay, the handoff row, and provenance. It cannot read open-cover compactness, a selected subsequence, quotient completion, host choice, or ambient completeness.

Proof. Project the carrier of Definition 1531.1. The packet has exactly the displayed coordinates $X, D, N, F, R, M, H, C, G, Q$, and every completion-facing read must be routed by C and named by Q.

Use Definition 1531.5. It ties the finite-net coordinate N, the Cauchy-filter coordinates F, R, and the regularity-modulus coordinate M into one completion-facing handoff before any basis row is exposed.

Then use Definition 1531.6. It states that a completion-facing consumer can inspect only the metric, net, filter, modulus, transport, route, handoff, and provenance rows of the same packet. The excluded open-cover, subsequence, quotient, choice, and ambient completeness objects are therefore absent from the bridge surface. □

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric bridged completion handoff`

Theorem 1531.11 (PrecompactMetricUp mature completion reflection)**.** *Every accepted* $\mathsf{PrecompactMetric}^{\uparrow}$ *packet has a mature completion-facing reflection surface: finite nets, regular filters,* $\mathsf{CauchyFilterBasis}^{\uparrow}$*,* $\mathsf{RegularityModulus}^{\uparrow}$*, and the completion handoff are all read through the same ten carrier rows*

$$X, D, N, F, R, M, H, C, G, Q.$$

The reflection exposes finite-net centers, regular-filter tails, basis generators, modulus windows, transport, replay, handoff, provenance, and local naming as one packet. It excludes open-cover compactness, selected subsequences, quotient completion, host choice, and ambient completeness.

Lean4 ✓ `BEDC.Derived.PrecompactMetricUp.PrecompactMetric mature completion reflection`

Proof. Start with Definition 1531.7. It routes finite nets through the regularity modulus and then through the Cauchy-filter rows before the completion-facing continuation, handoff, and provenance rows can be read.

Use Definition 1531.8. The same modulus row M controls finite-net tolerance requests in N and Cauchy-filter tail requests in F, R, so the route has one completion-facing rate ledger.

Apply Definition 1531.9. It identifies the total-bounded finite-net surface as the displayed metric, net, modulus, transport, replay, handoff, and provenance rows, with no open-cover or choice-selected replacement.

Finally apply Definition 1531.10. The completion-facing consumer receives exactly $X, D, N, F, R, M, H, C, G, Q,$ and the excluded quotient, selected-subsequence, host-choice, and ambient-completeness objects remain outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrecompactMetric}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1531.1, Definition 1531.2, Definition 1531.3, Definition 1531.7, Definition 1531.8, and Definition 1531.10 bind the finite-net and regular-filter dependency route through total-bounded nets, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, and $\mathsf{RegularityModulus}^{\uparrow}$ inspection rows before any completion-facing bridge can consume it.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrecompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: Open-cover compactness, selected subsequences, quotient completion, host choice, and ambient completeness are not claimed.

Upgrade path: Bridge checking requires Lean standard-bridge evidence for the finite-net and regular-filter completion handoff.

Constructive story: The construction starts from an already certified metric carrier, attaches explicit finite epsilon-net rows and finite Cauchy-filter inspection rows, and uses a regularity-modulus ledger to route total-boundedness consumers toward completion-facing data without importing classical compactness.

Chapter 1532

A Concrete Naming Certificate for $\mathsf{SpreadLaw}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on spreads and choice sequences, the $\mathsf{SpreadLaw}^{\uparrow}$ packet records the finite prefix law that licenses which StreamName windows may extend a displayed prefix. It sits between the fan and tree surfaces of chapter 400, chapter 1475, and chapter 334; the finite-window naming discipline of chapter 126; and the regular-rational readback route of chapter 301. The packet names a finite admissibility law before any regular-Cauchy or $\mathsf{Real}^{\uparrow}$ consumer reads a seal row, and it does not import a host infinite branch, countable choice, quotient stream, or ambient real completeness.

Definition 1532.1 (SpreadLaw carrier)**.** *A* $\mathsf{SpreadLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, A, E, W, H, C, P, N)$$

with the following visible rows:

1. *T is the prefix-closed tree row, listing finite prefixes over the displayed* $\mathsf{Bool}^{\uparrow}$*, list, and* $\mathsf{Nat}^{\uparrow}$ *digit/index surfaces and recording the prefix relation used by the classifier.*
2. *A is the admissibility decision row. For every displayed finite prefix in the packet, A records whether the prefix is licensed by the spread law and whether all shorter prefixes are licensed.*
3. *E is the extension witness row. Each admissible prefix in T carries a displayed one-step or bounded finite continuation whose initial segment remains in T and passes the decision row A.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed to downstream consumers; it contains only prefixes admitted by A and only continuations witnessed by E.*
5. *H stores componentwise* hsame *transport for the tree row, admissibility decisions, extension witnesses, finite windows, continuation replay, provenance, and local naming row.*
6. *C stores displayed* Cont *replay from prefix requests to admissibility checks, from admissibility checks to extension witnesses, and from extension witnesses to W.*
7. *P is the* Pkg *provenance over* $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SpreadLaw}^{\uparrow}}$ *row.*

The classifier accepts a packet only when T is prefix closed, A decides the displayed finite prefixes, E supplies continuation witnesses for every admissible prefix, W contains no window outside that finite law, and H, C, P, N preserve the same route. The carrier has no coordinate for a total choice sequence, a host infinite branch, countable choice, quotient-stream equality, regular-Cauchy completion, or a $\mathsf{Real}^{\uparrow}$ *seal.*

Theorem 1532.2 (SpreadLaw NameCert obligations)**.** *For an accepted* $\mathsf{SpreadLaw}^{\uparrow}$ *carrier* $S = (T, A, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpreadLaw}^{\uparrow}}$ *obligations are exactly carrier exposure, classifier stability, prefix-extension closure, finite-window compatibility, continuation replay, and ledger exactness. These obligations are discharged by projecting the displayed rows*

$$T \longmapsto A \longmapsto E \longmapsto W$$

together with H, C, P, N*.*

Proof. Carrier exposure projects the tuple of Definition 1532.1. The row T is the only tree surface, and the classifier can inspect only its displayed finite prefixes.

Classifier stability is read from A. If two presentations are componentwise hsame through H, the same admissibility decisions and the same rejected prefixes are replayed, so no later consumer can substitute a different law.

Prefix-extension closure is supplied by E. For every prefix accepted by A, the witness row names a finite continuation still governed by T and A, and the continuation replay row C records the route from the request to the witness.

Finite-window compatibility is the assertion that W contains only the prefixes and continuations exposed by T, A, E. The provenance and local naming rows P, N then make the ledger exact: every accepted $\mathsf{StreamName}^{\uparrow}$ window is accounted for by the same finite spread-law route, and no unrecorded infinite branch or completion seal is available. □

Theorem 1532.3 (SpreadLaw StreamName window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{SpreadLaw}^{\uparrow}$ *packet must first read the finite prefix law* T, A*, the extension witness row* E*, and the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*. Only after that route is displayed may the consumer read any regular-Cauchy row, regular-rational readback, or* $\mathsf{Real}^{\uparrow}$ *seal row.*

Proof. Read the accepted carrier of Definition 1532.1. The rows T, A, E, W are the only rows that expose finite prefix admission, continuation witnesses, and StreamName windows.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer is downstream of the window row. Its regular-rational readback therefore receives a prefix already certified by A and a continuation already witnessed by E.

A $\mathsf{Real}^{\uparrow}$ consumer is still later in the same route. Since the carrier contains no independent real seal coordinate, any real-facing read must pass through the finite law and the window handoff before it reaches a completion-facing row. □

Falsifiable prediction: Every accepted $\mathsf{SpreadLaw}^{\uparrow}$ handoff to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ factors through the displayed prefix tree, admissibility decision row, extension witness row, finite StreamName window, transport, continuation replay, provenance, and local NameCert rows.

Independence witness: fantheorem, cantorspace, bairespace, streamname, regseqrat: $\mathsf{SpreadLaw}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a prefix-closed finite tree row, admissibility decision row, extension witness row, finite StreamName window, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SpreadLaw}^{\uparrow}}$.

Constructive story: A $\mathsf{SpreadLaw}^{\uparrow}$ packet is generated from a prefix-closed finite Bool/List/Nat tree row, an admissibility decision row, extension witnesses, a finite StreamName window, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpreadLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1532.1, Definition 1532.2, and Definition 1532.3 seed the finite-prefix law surface that licenses StreamName windows before RegSeqRat or Real-facing consumers read completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpreadLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close FanTheorem compactness, CantorSpace or BaireSpace completeness, host infinite sequences, countable choice, quotient-stream equality, regular-Cauchy completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, admissibility classification, prefix-extension closure, StreamName window compatibility, RegSeqRat handoff, Real-facing non-escape, and ledger exactness for BEDC.Derived.SpreadLawUp.

Chapter 1533

A Concrete Naming Certificate for $\mathsf{LocatedMetric}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedMetric}^{\uparrow}$ packet records the located metric carrier required before the completion interface reads an arbitrary metric source. It sits over the metric-distance witness of chapter 57, the dyadic locatedness ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the separated zero-distance boundary of chapter 1411. The packet supplies the finite interface consumed by $\mathsf{Completion}^{\uparrow}$; it is not a quotient completion, a selected limit, trichotomy on distances, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1533.1 (Located metric carrier)**.** *A* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, D, S, R, E, Z, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-carrier row with its local point classifier and the finite observation rows on which distance requests are evaluated.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-witness row. It displays, for each accepted pair of point observations, the finite metric evidence that a distance request may inspect.*
3. *D is the dyadic locatedness row. For each distance request and each positive dyadic tolerance, it records a finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison window locating the distance below or apart from the requested bound.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that determines which finite distance reads, tolerance reads, and locatedness comparisons are visible to a consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that packages the scheduled dyadic comparisons as a regular rational name for the displayed distance value.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the metric witness, dyadic locatedness window, stream schedule, and regular rational readback have all been displayed.*
7. *Z is the separated zero-distance boundary row. It exports the zero-distance-to-point-classifier route through* $\mathsf{SeparatedMetric}^{\uparrow}$ *and does not identify points by host equality.*
8. *H stores componentwise* hsame *transport for the point carrier, metric witness, dyadic locatedness row, stream windows, regular rational readback, real seal, zero-distance boundary, continuation replay, provenance, and local naming row.*

9. *C stores* Cont *replay from point observations to metric distance witnesses, from distance witnesses to dyadic locatedness comparisons, from locatedness comparisons to stream and regular-rational readbacks, and from those readbacks to the real seal and separated zero-distance boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$ *row.*

The classifier compares accepted located-metric packets only through the displayed tuple $X, M, D, S, R, E, Z, H, C, P,$ It has no coordinate for quotient completion, selected limits, total trichotomy of distances, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricTasteGate single carrier alignment`

Theorem 1533.2 (Located metric NameCert obligations)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier $L = (X, M, D, S, R, E, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$ *obligations are exactly carrier admission, finite point classification, metric-distance witness readback, dyadic locatedness stability, scheduled regular-rational readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, separated zero-distance compatibility, and ledger non-escape. These obligations are discharged by the displayed route*

$$X \longmapsto M \longmapsto D \longmapsto S \longmapsto R \longmapsto E$$

together with Z, H, C, P, N.

Proof. Carrier admission is the projection of the displayed tuple in Definition 1533.1. The finite point classifier is read from X, and every distance request must pass through the metric witness row M before any dyadic or real-facing consumer may inspect it.

The locatedness obligation is the defining use of D. For each scheduled tolerance request, D supplies the finite $\mathsf{DyadicRatCore}^{\uparrow}$ comparison window that makes the metric distance located at that dyadic scale. Transport by H moves the points, windows, metric witness, and locatedness comparison componentwise, while C replays the same finite route after transport.

The scheduled readback obligation factors through S and R. The stream schedule determines the finite observations being read, and the regular rational row packages those reads before the real-facing seal E is consumed.

Zero-distance compatibility is isolated in Z. A consumer that wants to identify points from zero distance must use the separated-metric route of chapter 1411; no host equality or quotient equality is introduced inside the located-metric packet.

Finally, P and N source and name the same finite surface. Since the carrier has no coordinate for completion quotients, selected limits, trichotomy, choice, compactness, or ambient real completeness, the listed obligations exhaust the local NameCert interface. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier namecert obligations`

Theorem 1533.3 (Located metric completion handoff)**.** *Every* $\mathsf{Completion}^{\uparrow}$ *consumer that reads an arbitrary located metric source from an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *packet must factor through*

$$X \longmapsto M \longmapsto D \longmapsto S \longmapsto R \longmapsto E$$

and the separated boundary Z before forming completion data. The exported handoff is the finite tuple $X, M, D, S, R, E, Z, H, C, P, N$, with no hidden quotient completion, selected representative, countable choice, total comparison of distances, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a completion consumer asking for a metric source. The point carrier X supplies only finite point observations and their local classifier, so the consumer must obtain distance information from the metric witness row M.

Locatedness is then read through D. The dyadic row gives the requested finite comparison window and attaches it to the same point observations and distance witness. This is the step that turns a bare metric witness into a located metric input suitable for regular Cauchy completion.

The handoff to real-valued distance names proceeds through S, R, and E. The stream schedule fixes the finite reads, the regular rational readback packages them as a regular name, and the real seal is reached only after those rows are present.

If the completion route needs point identification at zero distance, it must use Z and hence the separated-metric sibling. The structural rows H, C, P, N preserve, replay, source, and name the same finite route, so the completion interface receives located metric data without importing any excluded quotient, selection, choice, trichotomy, compactness, or ambient-completeness principle. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier completion handoff`

Theorem 1533.4 (Dyadic radius localization)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, every locatedness read of a distance radius factors through*

$$M \longmapsto D \longmapsto S \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal* E *is available. Thus a consumer sees the metric witness, the positive dyadic comparison row, the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any real-valued distance observation is exposed.*

Proof. Begin with a locatedness request for a distance radius. The point observations in X do not carry a real value, so the request first reaches the metric witness M and obtains only the finite distance evidence displayed there.

The radius comparison is then localized by D. That row records the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance window and binds the comparison to the same metric witness, so no real seal or completion consumer can bypass the dyadic localization step.

Finally, S and R schedule and package the finite comparison reads. Only after these rows are present may E be consumed as a real-valued seal, which gives the stated factorization. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier dyadic radius localization`

Theorem 1533.5 (Separated zero route)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P$ *any identification of two point observations from zero distance must route through the separated metric row* Z*. The rows* $M, D, S, R, E, H, C, P, N$ *may transport, replay, source, and name the zero-distance evidence, but they do not identify points without the* $\mathsf{SeparatedMetric}^{\uparrow}$ *boundary recorded in* Z*.*

Proof. Suppose a consumer asks to identify two point observations because their displayed distance is zero. The metric and locatedness rows can exhibit the distance evidence and its dyadic localization, while S, R, E can package the same evidence as a scheduled real-valued read.

None of those rows contains the point-classifier quotient. The only row with a zero-distance-to-point-classifier route is Z, whose definition explicitly imports the separated metric boundary from chapter 1411.

The structural rows H, C, P, N preserve the route already chosen; they do not create a new equality principle. Therefore any valid zero-distance identification in the packet must pass through Z. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier separated zero route`

Theorem 1533.6 (Real seal readback)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$ *every* $\mathsf{Real}^{\uparrow}$*-valued distance read reaches* E *only through the finite route*

$$M \longmapsto D \longmapsto S \longmapsto R \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The readback therefore records the metric witness, dyadic locatedness row, $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *package, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$ *row before the real seal is consumed.*

Proof. Start from a request for a real-valued distance. The request must first be grounded in M, because only the metric witness row displays the finite distance evidence associated with the point observations.

The dyadic and scheduled readback rows then determine the visible approximation data. Row D gives the located comparison window, S fixes the finite schedule of reads, and R packages those reads as the regular rational object that can face the real seal.

It remains to check that the route is stable under the BEDC structural operations. Row H transports each component, row C replays the same route after transport, row P records provenance, and row N names the local certificate surface. Thus E is reached only after the listed route is present. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier real seal readback`

Falsifiable prediction: Every $\mathsf{Completion}^{\uparrow}$ read from a $\mathsf{LocatedMetric}^{\uparrow}$ packet factors through the displayed point carrier, metric distance witness, dyadic locatedness row, stream schedule, regular rational readback, real seal, separated zero-distance boundary, componentwise transport, continuation replay, provenance, and local naming row before completion data is consumed.

Independence witness: metric, dyadicratcore, streamname, regseqrat, real, separatedmetric: $\mathsf{LocatedMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a point carrier, metric distance witness, dyadic locatedness comparison, StreamName window schedule, RegSeqRat readback, RealUp seal, separated zero-distance boundary, componentwise transport, continuation replay, provenance, and local NameCert row.

Theorem 1533.7 (LocatedMetric radius-locality obligation)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, every radius-locality read is a finite BEDC obligation over the existing carrier rows* M, D, S, R, H, C, P, N*. The metric row* M *supplies the distance witness, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *supplies the positive tolerance window, the* $\mathsf{StreamName}^{\uparrow}$ *row* S *selects the finite read, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *packages the readback before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Project the carrier rows from Definition 1533.1. A radius request has no independent real-valued coordinate; it first reads the finite metric evidence in M and the dyadic locatedness window in D.

The schedule and readback rows are then forced by S and R. They keep the request inside a finite stream window and a regular rational package, while H, C, P, N only transport, replay, source, and name that same route.

Thus radius locality is a projection-and-replay obligation on the displayed rows. It is not supplied by quotient distance equality, a chosen radius representative, trichotomy on distances, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier radius locality obligation`

Theorem 1533.8 (LocatedMetric separated-zero obligation)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, any zero-distance-to-point-classifier read is a finite obligation through the existing separated boundary row* Z*. The rows* M, D, S, R, E *may display, locate, schedule, package, and seal the zero-distance evidence, but the point identification route must pass through* $\mathsf{SeparatedMetric}^{\uparrow}$ *as recorded by* Z*.*

Proof. Start with the zero-distance evidence. Rows M, D, S, R, E expose it as metric evidence, dyadic locatedness, a finite window, a regular rational readback, and a real-facing seal.

The carrier definition isolates point identification in Z. Since no other row contains the separated point-classifier boundary, projection from Definition 1533.1 leaves Z as the only admissible route from zero distance to point classification.

The structural rows H, C, P, N preserve the selected displayed route by transport, replay, provenance, and naming. They do not add quotient point equality or host equality, so the separated-zero obligation remains finite and local. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier separated zero obligation`

Theorem 1533.9 (LocatedMetric Real-seal obligation)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *distance seal read is a finite obligation over* $M, D, S, R, E, H, C, P, N$ *and the separated boundary* Z *when zero-distance compatibility is requested. The seal row* E *is consumed only after the* $\mathsf{Met}^{\uparrow}$ *witness,* $\mathsf{DyadicRatCore}^{\uparrow}$ *locatedness row,* $\mathsf{StreamName}^{\uparrow}$ *finite window, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed.*

Proof. Read the seal route in the order displayed by the carrier. The metric witness M grounds the distance request, D locates it at the requested dyadic tolerance, S selects the finite observations, and R packages the scheduled reads.

Only after those rows are present can E be consumed as a $\mathsf{Real}^{\uparrow}$ seal. If the same read is used to identify points at zero distance, Z supplies the separated metric boundary before the point classifier is reached.

Finally, H, C, P, N transport, replay, source, and name the same finite rows. The obligation therefore records a BEDC seal route rather than quotient real equality, choice of limits, or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier real seal obligation`

Theorem 1533.10 (LocatedMetric obligation closure package)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, the obligation surface of the chapter is the finite package consisting of carrier admission, the local* $\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$ *obligations, completion handoff, dyadic radius localization, separated zero route, Real seal readback, and ledger non-escape. Every public read factors through*

$$X \longmapsto M \longmapsto D \longmapsto S \longmapsto R \longmapsto E$$

together with Z, H, C, P, N *before any completion-facing, radius-facing, zero-distance-facing, or Real-facing consumer may inspect the packet.*

Proof. Carrier admission and the local naming obligations are projections of Definition 1533.1 and Definition 1533.2. They expose the point carrier, metric-distance witness, dyadic locatedness

row, stream schedule, regular rational readback, Real seal, separated boundary, transport, replay, provenance, and local NameCert row.

The completion and radius routes are already finite obligation rows. Apply Definition 1533.3 for the completion-facing consumer and Definition 1533.4 together with Definition 1533.7 for radius-local reads.

The zero-distance and Real-seal routes use the remaining displayed rows. Apply Definition 1533.5, Definition 1533.8, Definition 1533.6, and Definition 1533.9. Since the carrier has no coordinate for quotient completion, selected limits, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed package exhausts the local obligation surface and supplies ledger non-escape. □

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier obligation closure package`

Theorem 1533.11 (LocatedMetric scoped obligation rows)**.** *For an accepted* $\mathsf{LocatedMetric}^{\uparrow}$ *carrier* $L = (X, M, D, S, R, E, Z, H, C, P, N)$*, the radius-locality obligation, separated-zero obligation, and* $\mathsf{Real}^{\uparrow}$*-seal obligation depend only on displayed* BHist*,* hsame*,* Cont*,* NameCert*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows together with the carrier rows* $M, D, S, R, E, Z, H, C, P, N$*. No radius, zero-distance, or seal consumer may add a quotient metric, selected limit, trichotomy decision, countable-choice selector, open-cover compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row outside that scope.*

Lean4 ✓ `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier scoped obligation rows`

Proof. The radius-locality theorem Definition 1533.7 displays the finite route through the metric row, dyadic locatedness row, stream schedule, regular-rational readback, transport, replay, provenance, and local naming row.

For zero distance, Definition 1533.8 keeps the separated boundary Z visible before any point classifier is reached. The surrounding rows only transport and name the same finite evidence.

For the real seal, Definition 1533.9 consumes E after M, D, S, R and retains H, C, P, N. Together with Definition 1533.10, these three obligations close the paper-side scope of the located-metric rows without changing the Lean verification state. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedMetric}^{\uparrow}$ packet is generated from a point carrier, a MetricUp distance witness, a DyadicRatCore locatedness ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, a separated zero-distance boundary, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1533.1, Definition 1533.3, Definition 1533.10, and Definition 1533.11 bind the public completion handoff, radius-locality, separated-zero, and Real-seal obligations to displayed BHist, hsame, Cont, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedMetric}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.LocatedMetricUp.LocatedMetricCarrier\_completion\_handoff`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, trichotomy on distances, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: No further paper-side theory-grade closure remains; bridge checking remains a separate verification-axis matter.

Chapter 1534

A Concrete Naming Certificate for $\mathsf{SturmRootIsolation}^{\uparrow}$

Following Sturm and Bishop–Bridges constructive analysis, the $\mathsf{SturmRootIsolation}^{\uparrow}$ packet reads a finite polynomial root-isolation surface through Sturm-chain sign-variation data rather than through host algebraic closure, real-root choice, or ambient completeness. It consumes the polynomial source of chapter 50, the rational and dyadic interval ledgers of chapter 354 and chapter 306, and the adjacent sign-change bisection packet of chapter 1407. Its downstream completion boundary is the finite window route through chapter 126, chapter 301, and chapter 38.

Definition 1534.1 (Sturm root-isolation carrier)**.** *A* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$$

with the following visible rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, including the displayed coefficient ledger and the finite derivative and pseudo-remainder inputs from which the Sturm chain is read;*
2. *I is the* $\mathsf{RationalInterval}^{\uparrow}$ *interval ledger carrying the inspected rational cells and their endpoint order rows;*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger used for finite cell subdivision and endpoint-window export;*
4. *V is the Sturm-chain division and sign-variation ledger, recording each displayed division row, the endpoint sign rows for every chain entry, and the variation-count differences for the inspected cells;*
5. *B is the selected isolating branch, represented as the retained cell together with the finite comparison rows showing that its variation drop is the accepted one-root branch and that discarded cells have the displayed alternative counts;*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window surface exposed after the selected rational or dyadic cell has been fixed;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the selected interval windows and the requested unary precision;*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal for the retained endpoint names and for downstream real-facing consumers of the isolating window;*

9. *H stores the componentwise* hsame *transports for the polynomial row, interval rows, Sturm ledger, branch row, stream windows, regular readbacks, real seal rows, package provenance, and local certificate;*
10. *C stores the displayed* Cont *replay routes for polynomial inspection, Sturm-chain division replay, sign-variation counting, branch selection, window export, regular-sequence handoff, and real-seal consumption;*
11. *Q_0 is the* Pkg *provenance over* $\mathsf{Poly}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{BisectionRootIsolation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmRootIsolation}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite Sturm-chain root-isolation packet. It has no coordinate for an algebraic root object, a host factorization theorem, a classical completeness principle, a quotient real equality, or an ambient real-root existence theorem.

Lean4 (definition) `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier`

Theorem 1534.2 (Sturm root-isolation NameCert obligations)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$, the local* $\mathsf{NameCert}_{\mathsf{SturmRootIsolation}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the finite packet of Definition 1534.1;*
2. *polynomial-source admission through the coefficient, derivative, and pseudo-remainder rows of P;*
3. *interval admission through the rational cell ledger I and the dyadic refinement ledger D;*
4. *Sturm-ledger exactness, requiring every division row and every endpoint sign row used in V to be one of the displayed finite rows;*
5. *branch isolation, requiring the selected row B to be justified by the displayed sign-variation drop in V and handed to the adjacent bisection packet only as a finite retained-cell surface;*
6. *retained-window handoff through W, R, and S;*
7. *componentwise transport, replay, provenance, and naming through H, C, Q_0, and N.*

No further analytic or algebraic obligation is part of the packet.

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier namecert obligations`

Proof. Project the carrier of Definition 1534.1. The polynomial row supplies only the displayed coefficient, derivative, and pseudo-remainder inputs, while the rational and dyadic interval rows supply only the inspected cells and endpoint windows.

The Sturm ledger V accounts for the chain computation. Each sign row used in a variation count is attached to a displayed endpoint and a displayed chain entry, and each division row is recorded as finite ledger data rather than as an external factorization theorem.

The selected branch B is then read from the displayed variation-drop ledger. Its retained cell may be handed to the bisection surface of chapter 1407, but only as the already selected rational or dyadic window with its sign and variation evidence still attached.

Finally, W, R, and S expose the finite window to stream, regular-rational, and real-facing consumers. The rows H, C, Q_0, and N supply transport, replay, provenance, and local naming, so the packet has no remaining coordinate from which to extract an algebraic root object, host factorization, quotient real equality, or completeness principle. □

Theorem 1534.3 (Sturm root-isolation sign-variation handoff)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier and every selected branch row* B*, the branch handed to downstream interval consumers factors through the* $\mathsf{Poly}^{\uparrow}$ *row* P*, the* $\mathsf{RationalInterval}^{\uparrow}$ *and* $\mathsf{DyadicInterval}^{\uparrow}$ *interval rows* I, D*, the Sturm-chain division and sign-variation ledger* V*, the retained-cell row of* B*, and the replay rows* H, C, Q_0, N*. The handoff exports a finite isolating cell with displayed variation-count evidence and exports no host root selection.*

Proof. Read the selected branch coordinate. By construction, it is not independent of the Sturm ledger: the retained cell in B is accepted only after the endpoint signs and variation-count difference have been read from V.

The interval coordinates I and D provide the cell on which those signs are inspected. The polynomial coordinate P provides the finite chain entries whose signs are counted, and H, C, Q_0, N replay the same rows under transport and package provenance.

Consequently any downstream interval consumer receives the retained cell together with its displayed variation-count evidence. Since no coordinate stores a chosen host root or a completed real-root witness, the handoff remains a finite NameCert surface. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier sign variation handoff`

Theorem 1534.4 (Sturm root-isolation squarefree route)**.** *Every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier reads the squarefree polynomial route before a sign-variation count can be consumed. Concretely, the polynomial source row* P *and Sturm ledger* V *must display the squarefree factor row, the pseudo-remainder or resultant compatibility row, the endpoint sign rows, and the retained interval rows from* I, D *before the branch row* B *is available. The interval-refinement certificate may cite chapter 2079, chapter 7329, chapter 354, chapter 38, chapter 301, and chapter 126 only through these displayed* BHist *and* NameCert *rows, and it exports no host squarefree factorization theorem.*

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier squarefree route`

Proof. Project the carrier of Definition 1534.1. The polynomial source row P contains the coefficient, derivative, pseudo-remainder, and squarefree-route data from which the finite Sturm sequence is read.

Use the sign-variation handoff of Definition 1534.3. It places every branch consumer downstream of P, I, D, V, B and therefore downstream of the displayed squarefree and endpoint-sign rows that feed V.

The sibling chapters supply only their finite certificate surfaces: chapter 2079 supplies the chain rows, chapter 7329 supplies compatibility data, and the rational, stream, regular-rational, and Real siblings supply interval and completion rows. Since no carrier coordinate stores a host factorization theorem, the squarefree route is exactly the displayed packet route. □

Theorem 1534.5 (Sturm root-isolation subresultant sign-variation route)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier* $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$*, the sign-variation count used by* B *is downstream of the subresultant or pseudo-remainder compatibility rows displayed in* P *and* V*. The interval-refinement certificate cannot bypass those rows: any refinement that reaches* W, R, S *must first replay the polynomial resultant route, the Sturm sequence route, the endpoint sign rows, and the rational or dyadic interval rows through* H, C, Q_0, N*.*

Proof. Read the Sturm ledger V. Its variation-count entries are tied to displayed chain entries and endpoint signs, so a count cannot be consumed before the subresultant or pseudo-remainder compatibility rows for those chain entries have been displayed.

Apply the squarefree route theorem Definition 1534.4. It places the branch row B downstream of the squarefree polynomial route and the resultant compatibility rows, while chapter 2079 and chapter 7329 provide only named finite sibling surfaces.

Finally use interval refinement and Real seal nonescape, Definition 1534.7 and Definition 1534.9. The retained interval cannot reach $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ rows before the same polynomial, resultant, sign-variation, and interval rows have been replayed through H, C, Q_0, N. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier subresultant sign variation`

Theorem 1534.6 (Sturm root-isolation sign-variation scope)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier* $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$*, every accepted branch selection reads the polynomial chain rows in* P *and the endpoint sign rows in* V *before the branch row* B *is available. The branch classifier therefore acts only on displayed polynomial, endpoint-sign, and variation-count data, and it exports no root object or external sign oracle.*

Proof. Project the carrier of Definition 1534.1. The polynomial source row P stores the displayed coefficient, derivative, and pseudo-remainder inputs from which the finite Sturm chain is read.

Use the sign-variation handoff of Definition 1534.3. It places the selected branch downstream of the Sturm ledger V, so the endpoint sign rows and variation-count difference are already visible before B can be consumed.

The structural rows H, C, Q_0, N replay and name the same finite rows. Since the carrier has no coordinate for a host root, algebraic closure, or independent sign oracle, branch selection is scoped to the displayed polynomial chain and endpoint sign ledger. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier sign variation scope`

Theorem 1534.7 (Sturm root-isolation interval refinement)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier* $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$*, rational interval cells from* I *refine through the* $\mathsf{DyadicInterval}^{\uparrow}$ *row* D *before any* $\mathsf{StreamName}^{\uparrow}$ *window* W *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is exposed. Thus a completion-facing consumer sees stream and regular-rational windows only after the retained rational cell has passed through the displayed dyadic refinement ledger.*

Proof. Read the interval coordinates of Definition 1534.1. The rational interval row I carries the inspected cells, while D records the dyadic refinement ledger used for finite subdivision and endpoint-window export.

Apply the branch handoff of Definition 1534.3. It keeps the retained cell attached to the rational and dyadic interval rows before the branch is handed to downstream interval consumers.

The Real seal nonescape theorem Definition 1534.9 then places the $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{RegSeqRat}^{\uparrow}$ readback R downstream of that selected interval route. Hence no stream or regular-rational window is exposed before dyadic refinement has been displayed. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier interval refinement`

Theorem 1534.8 (Sturm root-isolation certificate ledger)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier* $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$*, the local certificate ledger is exhausted by the rows*

$$P, I, D, V, B, W, R, S, H, C, Q_0, N.$$

It exports polynomial source data, rational and dyadic interval rows, sign-variation rows, a selected branch, stream windows, regular rational readback, a Real-facing seal, transport, replay, provenance, and local naming, but no host root, algebraic closure, quotient equality, real trichotomy, or completeness principle.

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier certificate ledger`

Proof. Start from the NameCert obligations of Definition 1534.2. They list exactly the carrier, polynomial, interval, Sturm-ledger, branch, retained-window, transport, replay, provenance, and naming obligations.

The sign-variation scope theorem shows that branch selection consumes only P, V, B together with the interval route. The interval refinement theorem shows that I, D precede the stream and regular-rational rows W, R.

Finally use the Real seal nonescape theorem. It places the seal row S downstream of $I, D, V, W, R, H, C, Q_0, N$ and excludes algebraic roots, host root existence, quotient real equality, trichotomy, and ambient completeness. The displayed tuple therefore exhausts the certificate ledger. □

Theorem 1534.9 (Sturm root-isolation Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier is reached only after the selected rational or dyadic interval row, the displayed Sturm-chain sign-variation ledger, the* $\mathsf{StreamName}^{\uparrow}$ *window surface, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the transport, replay, provenance, and local naming rows have been read. The seal exposes the finite isolating window and exports no algebraic root object, host root existence theorem, quotient real equality, real trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the seal coordinate S. It is stored with the selected interval row, the Sturm ledger, the stream windows, the regular-rational handoff, and the BEDC transport and replay rows; it is not stored as a free real object.

First the selected branch is fixed by Definition 1534.3. Then the retained interval windows pass through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces at the requested precision.

Only after that route is fixed can the $\mathsf{Real}^{\uparrow}$ seal be read. Thus the seal records the same finite isolating window and cannot export an algebraic root object, host root existence, quotient real equality, trichotomy data, or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier real seal nonescape`

Theorem 1534.10 (Sturm root-isolation obligation closure route)**.** *For every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier* $Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$*, the polynomial source row, rational and dyadic interval rows, Sturm sign-variation ledger, selected branch,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* hsame*,* Cont*,* Pkg*, and* NameCert *rows exhaust the local obligation surface. The packet has no remaining obligation for a host root object, algebraic closure theorem, quotient real equality, real trichotomy, selected real root, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier obligation closure route`

Proof. Start with the NameCert obligations of Definition 1534.2. They list carrier admission, polynomial-source admission, rational and dyadic interval admission, the Sturm ledger, branch isolation, retained-window handoff, transport, replay, provenance, and local naming.

Use the sign-variation handoff and scope rows, Definition 1534.3 and Definition 1534.6. They place every accepted branch downstream of the displayed polynomial chain rows, endpoint sign rows, and variation-count differences rather than a host root selector.

Apply interval refinement and certificate ledger exhaustion, Definition 1534.7 and Definition 1534.8. These force rational cells through the dyadic refinement ledger before any stream window, regular-rational readback, or real-facing seal is exposed.

Finally apply Real seal nonescape, Definition 1534.9. Since the seal is downstream of the selected interval, Sturm ledger, stream windows, regular-rational readback, transport, replay, provenance, and naming rows, no algebraic root object, host root theorem, quotient equality, trichotomy datum, selected root, or completeness principle remains. □

Theorem 1534.11 (Sturm root-isolation scoped kernel binding)**.** *Every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *carrier*

$$Q = (P, I, D, V, B, W, R, S, H, C, Q_0, N)$$

is scoped exactly by the displayed $\mathsf{Poly}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{BisectionRootIsolation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before any branch consumer or Real-seal consumer reads it. The scoped packet exposes no algebraic root object, host factorization theorem, quotient real equality, real trichotomy, selected real root, or* $\mathsf{Real}^{\uparrow}$ *completeness principle outside those rows.*

Proof. Start with the NameCert obligations of Definition 1534.2. They bind carrier admission to the displayed polynomial source, rational and dyadic interval ledgers, Sturm sign-variation ledger, selected branch, stream window, regular-rational readback, Real seal, transport, replay, provenance, and local naming rows.

Use the sign-variation handoff of Definition 1534.3. It places the branch row downstream of the polynomial, interval, and sign-variation data, with the adjacent $\mathsf{BisectionRootIsolation}^{\uparrow}$ consumer receiving only the retained finite cell and its variation-count evidence.

Apply interval refinement, Definition 1534.7. It forces rational cells through the dyadic interval ledger before stream windows or regular-rational readback become visible.

Finally apply Real seal nonescape, Definition 1534.9. The seal remains downstream of the same displayed rows and cannot expose host algebraic roots, quotient equality, trichotomy, selected roots, or ambient completeness. Hence every branch or Real-seal consumer reads exactly the listed kernel rows before it can consume the packet. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier scoped kernel binding`

Theorem 1534.12 (Sturm root-isolation scoped interval and Real-seal route)**.** *Every accepted* $\mathsf{SturmRootIsolation}^{\uparrow}$ *branch-selection consumer must pass through the* $\mathsf{Poly}^{\uparrow}$ *source row,* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger,* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger, Sturm sign-variation ledger,* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *branch handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows before it can be used as a scoped interval or real-facing root-isolation route. The route does not claim algebraic root existence, factorization completeness, real trichotomy, quotient equality, selected real roots, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 1534.2. They expose the polynomial source row, rational and dyadic interval rows, Sturm ledger, branch row, stream windows, regular-rational readback, Real seal, transport, replay, provenance, and local naming as the finite carrier surface.

Use the sign-variation handoff and sign-variation scope, Definition 1534.3 and Definition 1534.6. These place every accepted branch downstream of the polynomial chain rows, endpoint sign rows, variation-count differences, and the retained $\mathsf{BisectionRootIsolation}^{\uparrow}$ branch handoff.

Apply interval refinement and certificate ledger exhaustion, Definition 1534.7 and Definition 1534.8. The rational cell ledger and dyadic refinement ledger therefore remain visible before any $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, or real-facing consumer is available.

Finally use Real seal nonescape, the obligation-closure route, and scoped kernel binding: Definition 1534.9, Definition 1534.10, and Definition 1534.11. Together they keep the branch and seal consumers inside the displayed polynomial, interval, Sturm-ledger, bisection, stream, regular-rational, and real-seal rows. Since the carrier has no coordinate for algebraic root existence, factorization completeness, trichotomy, quotient equality, selected real roots, or Real completeness, the consumer route is exactly the scoped interval and Real-seal route stated above. □

Lean4 ✓ `BEDC.Derived.SturmRootIsolationUp.SturmRootIsolationCarrier scoped interval real seal route`

Closure status for $\mathsf{NameCert}_{\mathsf{SturmRootIsolation}^{\uparrow}}$.

Constructive story: A $\mathsf{SturmRootIsolation}^{\uparrow}$ packet is generated from a $\mathsf{Poly}^{\uparrow}$ source row, $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{DyadicInterval}^{\uparrow}$ interval rows, a finite Sturm-chain division and sign-variation ledger, a selected isolating branch, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmRootIsolation}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1534.1, Definition 1534.2, Definition 1534.3, Definition 1534.9, and Definition 1534.11 bind finite Sturm-chain root isolation to the displayed polynomial, rational interval, dyadic interval, stream, regular-rational, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmRootIsolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close algebraic root existence, a host factorization theorem, quotient real equality, real trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Scoped kernel binding is the next paper-theory step, tying the finite Sturm-chain obligation package to a public interval and real-seal consumer boundary.

Chapter 1535

A Concrete Naming Certificate for $\mathsf{UltrametricSpace}^{\uparrow}$

Following Schikhof's non-Archimedean analysis and the Bosch–Guntzer–Remmert treatment of ultrametric geometry, the $\mathsf{UltrametricSpace}^{\uparrow}$ packet records a metric surface whose triangle evidence is the strong max-radius comparison. It is a standard metric neighbour of $\mathsf{Met}^{\uparrow}$ in chapter 57, the p-adic example surface of chapter 100, the ball ledgers of chapter 311 and chapter 1450, and the completion-facing boundary of chapter 128. The carrier names only finite distance comparisons, radius ledgers, nested-ball witnesses, example handoffs, transport, replay, provenance, and a local certificate; it does not import quotient classes, choice of representatives, host real equality, or a completed ambient field.

Definition 1535.1 (UltrametricSpace carrier)**.** *An* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, V, T, B, E, H, K, P, N)$$

with the following visible rows:

1. *M is the parent* $\mathsf{Met}^{\uparrow}$ *row carrying the displayed point names, distance classifier, symmetry ledger, zero-distance boundary, and ordinary triangle route.*
2. *V is the value and radius comparison ledger, recording the finite distance values or distance bounds inspected by the packet and the max-radius comparison used for ball consumers.*
3. *T is the strong-triangle ledger, asserting for every displayed triple read by the packet that the third distance is routed through the maximum of the other two displayed radii.*
4. *B is the nested-ball ledger, classifying displayed balls by the ultrametric rule that intersecting balls are nested once their centers and radii pass through V and T.*
5. *E is the example handoff row, exposing either a* $\mathsf{Padic}^{\uparrow}$ *prime-scale route or a* $\mathsf{DyadicBall}^{\uparrow}$ *radius route as a finite witness of the same strong-triangle discipline.*
6. *H stores componentwise* hsame *transport for the parent metric row, comparison ledger, strong-triangle ledger, nested-ball ledger, example handoff, continuation replay, provenance, and local naming row.*
7. *K stores displayed* Cont *replay from metric reads to comparison reads, from comparison reads to strong-triangle checks, and from strong-triangle checks to nested-ball,* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{DyadicBall}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers.*

8. *P is the* Pkg *provenance over* $\mathsf{UltrametricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UltrametricSpace}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the parent metric row is already displayed, every inspected distance comparison is routed through V, every triangle request is answered by the strong max-radius row T, nested-ball claims are read from B, example consumers factor through E, and all downstream reads preserve H, K, P, N. It has no coordinate for quotient metric completion, selected representatives, host real equality of distances, open-cover compactness, or an ambient non-Archimedean field.

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceTasteGate single carrier alignment`

Theorem 1535.2 (UltrametricSpace NameCert obligations)**.** *For an accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier $U = (M, V, T, B, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UltrametricSpace}^{\uparrow}}$ *obligations are exactly the following:*

1. *carrier admission for the displayed parent metric, comparison ledger, strong-triangle ledger, nested-ball ledger, example handoff, transport, replay, provenance, and local naming rows;*
2. *inherited metric-row compatibility, requiring all point names, distance reads, symmetry checks, and ordinary triangle routes to be read from M before the ultrametric rows are inspected;*
3. *strong-triangle stability, requiring transported packets to preserve the max-radius comparison row T under the componentwise* hsame *data in H;*
4. *nested-ball classification, requiring every ball-consumer claim to pass through B after the relevant centers and radii have been compared by V and T;*
5. *p-adic or dyadic witness compatibility, requiring the example row E to expose only a* $\mathsf{Padic}^{\uparrow}$ *or* $\mathsf{DyadicBall}^{\uparrow}$ *route that reuses the same strong-triangle comparison;*
6. *ledger non-escape, excluding quotient completions, selected representatives, host real equality of distance values, open-cover compactness, and ambient non-Archimedean field laws.*

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1535.1. The tuple lists exactly the parent metric row, comparison ledger, strong-triangle ledger, nested-ball ledger, example handoff, transport, replay, provenance, and local name.

Inherited metric-row compatibility is read first. A consumer cannot ask for an ultrametric comparison before the point names, distance reads, symmetry checks, and ordinary metric triangle route have been displayed by M.

Strong-triangle stability is then componentwise transport along H. The comparison row V fixes the inspected radii, and T records that each displayed triple uses the maximum of the two comparison radii as the controlling bound.

Nested-ball classification follows from B. Every ball-facing read must pass through the displayed center and radius comparisons, so a later $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, or $\mathsf{CompleteMetric}^{\uparrow}$ consumer cannot replace the nested-ball route by an unrelated host-space assertion.

Finally, E, K, P, N replay, source, and name the same finite route for p-adic or dyadic examples. Since the carrier fields are exhausted by $M, V, T, B, E, H, K, P, N$, the excluded quotient, representative-choice, host-equality, compactness, and ambient-field principles have no row from which a consumer could read them. □

Theorem 1535.3 (UltrametricSpace root strong-triangle window)**.** *For every accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier* $U = (M, V, T, B, E, H, K, P, N)$*, every root-level strong-triangle read is a* BHist*-anchored window over the parent metric row* M*, the value and radius comparison ledger* V*, and the strong max-radius row* T*. The read may refine the displayed comparison window, but it cannot replace* M*, bypass Definition 1535.2, or import host quotient fields or chosen distance representatives.*

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceRootStrongTriangleWindow`

Proof. Project the carrier of Definition 1535.1. The parent metric data are stored in M, finite value comparisons are stored in V, and the strong-triangle evidence is stored in T.

Apply Definition 1535.2. Its inherited metric-row compatibility requires every point-name and distance read to pass through M before any ultrametric comparison is consumed, and its strong-triangle stability clause preserves T under the displayed transport row.

Thus a root consumer can only widen or refine the finite comparison window already visible in M, V, T. Quotient metric fields, selected representatives, and host equality of distances have no carrier row and cannot supply the strong max-triangle comparison. □

Theorem 1535.4 (UltrametricSpace root nested-ball ledger)**.** *For every accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier* $U = (M, V, T, B, E, H, K, P, N)$*, root-level nested-ball classification is read from the same* BHist *ledger* B *after the relevant center and radius comparisons have passed through* M, V, T*. A nested-ball consumer may not allocate a separate ball classifier, quotient completion row, selected-center representative, or ambient compactness principle.*

Proof. Read the packet through Definition 1535.1. The nested-ball row B is displayed as a carrier coordinate, and its inputs are the center and radius comparisons routed through the parent metric row, comparison ledger, and strong-triangle ledger.

Use Definition 1535.2. The nested-ball classification obligation says that every ball-consumer claim passes through B after V and T have compared the relevant data.

Therefore the root nested-ball surface is the finite ledger B over the same BHist route. No separate classifier, quotient completion, representative choice, or compactness principle is exposed by the accepted packet. □

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceRootNestedBallLedger`

Theorem 1535.5 (UltrametricSpace root example handoff)**.** *For every accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier* $U = (M, V, T, B, E, H, K, P, N)$*, any root example handoff to a* p*-adic, dyadic-ball, metric-ball, or complete-metric consumer must pass through the same finite* BHist *route displayed by the carrier and obligated by Definition 1535.2. The row* E *may expose a* $\mathsf{Padic}^{\uparrow}$ *or* $\mathsf{DyadicBall}^{\uparrow}$ *witness, but it cannot choose representatives, add a quotient field, or assert ambient non-Archimedean field laws.*

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceRootExampleHandoff`

Proof. Start from Definition 1535.1. The example handoff row E is stored together with M, V, T, B, H, K, P, N, so it is not an independent model of a host ultrametric space.

Apply Definition 1535.2. Its example-compatibility clause restricts E to the displayed p-adic or dyadic route, and its ledger non-escape clause excludes quotient completions, selected representatives, host equality, and ambient field laws.

Consequently every root example read reuses the same finite route. The handoff can serve downstream $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ consumers only by replaying the accepted BHist data already named by the carrier. □

Theorem 1535.6 (UltrametricSpace strong-triangle handoff)**.** *Every accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *packet exposes distance comparisons to* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{DyadicBall}^{\uparrow}$*,* $\mathsf{Padic}^{\uparrow}$*, or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers only through the finite route*

$$M \longmapsto V \longmapsto T \longmapsto B \longmapsto E.$$

In particular, the max-radius strong-triangle row T *is read before any metric-ball, dyadic-ball, p-adic, or complete-metric consumer can inspect the route.*

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceStrongTriangle handoff`

Proof. Start with the parent metric row M. It supplies point names and distance reads, but it does not by itself certify the non-Archimedean comparison needed by an ultrametric consumer.

The comparison ledger V records the finite radii under inspection. The strong-triangle row T then binds every displayed triple to the max-radius bound determined by those comparisons.

Only after T has been read can the nested-ball row B classify ball relations. The example row E and the continuation replay K preserve that order for $\mathsf{Padic}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Thus each downstream consumer receives the same displayed finite route through M, V, T, B, E. No consumer-facing field bypasses the strong-triangle row, so host equality of distances, quotient completion equality, selected representatives, and ambient field laws remain outside the packet. □

Theorem 1535.7 (UltrametricSpace strong-triangle bridge route)**.** *Every accepted* $\mathsf{UltrametricSpace}^{\uparrow}$ *carrier exports a bridge route from the abstract metric interface to the strong max-radius triangle interface through the finite rows*

$$M, V, T, B, E, H, K, P, N.$$

The route preserves the parent $\mathsf{Met}^{\uparrow}$ *surface, the* $\mathsf{Padic}^{\uparrow}$ *or* $\mathsf{DyadicBall}^{\uparrow}$ *example handoff, the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers, the* BHist *ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.UltrametricSpaceUp.UltrametricSpaceStrongTriangleBridgeRoute`

Proof. Start with the abstract metric interface. The carrier row M supplies the point names, distance reads, symmetry ledger, zero-distance boundary, and ordinary triangle route, so every later ultrametric read remains anchored in the public $\mathsf{Met}^{\uparrow}$ surface.

The bridge then reads the finite comparison rows. The value ledger V fixes the inspected radii, the strong-triangle row T turns those comparisons into the max-radius triangle interface, and the nested-ball row B exposes the ball-facing consequence to $\mathsf{MetricBall}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

The example handoff row E keeps the route tied to the displayed $\mathsf{Padic}^{\uparrow}$ or $\mathsf{DyadicBall}^{\uparrow}$ witness. The transport, replay, provenance, and naming rows H, K, P, N preserve the same finite path, so no consumer can bypass the strong-triangle row by importing quotient completion data, selected representatives, host equality of distances, or ambient non-Archimedean field laws. □

Falsifiable prediction: Every accepted $\mathsf{UltrametricSpace}^{\uparrow}$ ball-facing read factors through the parent $\mathsf{Met}^{\uparrow}$ row, value and radius comparison ledger, max-radius strong-triangle ledger, nested-ball ledger, example handoff, componentwise transport, continuation replay, provenance, and local NameCert before any $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, or $\mathsf{CompleteMetric}^{\uparrow}$ consumer reads the route.

Independence witness: metric, padic, dyadicball, metricball, completemetric: $\mathsf{UltrametricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries parent metric data, value and radius comparisons, a strong-triangle ledger, nested-ball classification, a p-adic or dyadic example handoff, transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UltrametricSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{UltrametricSpace}^{\uparrow}$ packet starts from a displayed Metric row, records finite value and radius comparisons, certifies the strong max-radius triangle row, routes nested-ball classification through that row, exposes p-adic or dyadic examples as handoffs, and preserves the same finite route by transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UltrametricSpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1535.1, Definition 1535.2, Definition 1535.3, Definition 1535.4, Definition 1535.5, Definition 1535.6, and Definition 1535.7 give the carrier, NameCert obligations, root windows, nested-ball ledger, example handoff, strong-triangle handoff, and strong-triangle bridge route over $\mathsf{Met}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UltrametricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient metric completion, selected representatives, host real equality of distances, open-cover compactness, ambient non-Archimedean field laws, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, inherited metric-row compatibility, strong-triangle stability, nested-ball classification, p-adic or dyadic witness compatibility, handoff ordering, and ledger non-escape for BEDC.Derived.UltrametricSpaceUp.

Chapter 1536

A Concrete Naming Certificate for $\mathsf{LowerSemicontinuous}^{\uparrow}$

Following Bishop and Bridges constructive analysis, lower semicontinuity is read through lower bounds and finite epigraph witnesses rather than through a host topology. The $\mathsf{LowerSemicontinuous}^{\uparrow}$ packet records that interface over the lower-real surface of chapter 1446, the upper-real comparison boundary of chapter 1447, the located readbacks of **??**, the modulus-bearing map rows of chapter 130, the scheduled stream windows of chapter 126, the regular rational source rows of chapter 301, and the terminal real-name seal of chapter 38.

Definition 1536.1 (LowerSemicontinuous carrier)**.** *A* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, F, E, W, R, O, H, C, P, N)$$

with the following visible rows:

1. *X is the source carrier row whose accepted points are read through displayed* BHist *histories and their classifier transport;*
2. *F is the function graph or map row, read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise carrier only when such a map surface is present;*
3. *E is the lower-real epigraph ledger, consisting of finite rows (x, q) saying that the value at the displayed source point admits the rational lower bound q at the inspected window;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting finite source and value windows for each epigraph inspection;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row giving the regular rational value windows used by E;*
6. *O is the* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison ledger used to read lower windows, upper exclusions, located endpoint checks, and finite lower-bound refinements;*
7. *H records componentwise* hsame *transport for source histories, graph rows, lower-real epigraph rows, stream windows, regular readbacks, comparison ledgers, provenance rows, and local naming rows;*
8. *C records displayed* Cont *replay from source admission through window selection, regular rational readback, lower-real epigraph inspection, located readback, and real-seal handoff;*
9. *P is the* Pkg *provenance over* $\mathsf{LowerSemicontinuous}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^\uparrow}$ *row.*

The classifier relates two accepted packets exactly when their source rows, finite epigraph windows, regular rational readbacks, lower-real comparisons, located readbacks, $\mathsf{Real}^\uparrow$ *seal rows, and provenance rows transport componentwise along H and replay through the same displayed routes in C. No host topology, open-set lattice, quotient real equality, semantic limit operator, or completeness principle is a carrier coordinate.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousTasteGate single carrier alignment`

Definition 1536.2 (LowerSemicontinuous BHist carrier row)**.** *The root* $\mathsf{LowerSemicontinuous}^\uparrow$ BHist *row is $L = (X, F, E, W, R, O, H, C, P, N)$ of Definition 1536.1, read only as source histories, epigraph windows,* $\mathsf{StreamName}^\uparrow$ *schedules,* $\mathsf{RegSeqRat}^\uparrow$ *value rows, located* $\mathsf{Real}^\uparrow$ *handoff data,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

Lean4 (definition) `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousBHistCarrierRow`

Theorem 1536.3 (LowerSemicontinuous kernel-source obligations)**.** *Every kernel-source read of an accepted* $\mathsf{LowerSemicontinuous}^\uparrow$ *carrier enters through Definition 1536.2, then passes through* $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, *located* $\mathsf{Real}^\uparrow$ *handoff,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before reaching any lower-bound consumer. No carrier/classifier parameter transport is available outside that route.*

Proof. Project the root carrier row. A lower-bound request chooses a $\mathsf{StreamName}^\uparrow$ window, reads the $\mathsf{RegSeqRat}^\uparrow$ value row, passes the located $\mathsf{Real}^\uparrow$ handoff, and uses hsame and Cont only to preserve the displayed packet; the final Pkg and $\mathsf{NameCert}$ rows source and name that finite route. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousKernelSource obligations`

Theorem 1536.4 (LowerSemicontinuous NameCert scope)**.** *The* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^\uparrow}$ *scope for the root packet is exactly the finite* BHist, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, hsame, Cont, Pkg, *and local naming route of Definition 1536.2. It covers lower-bound windows and located handoff rows, but not parameter equivalence, host topology, semantic liminf data, quotient real equality, countable choice, compactness, or ambient completeness.*

Proof. Apply the carrier definition and the kernel-source obligations above. The $\mathsf{NameCert}$ row certifies only the listed finite route; parameter equivalence or host analytic structure would require an additional carrier coordinate. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousNameCertScope`

Theorem 1536.5 (LowerSemicontinuous NameCert obligation surface)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^\uparrow$ *carrier $L = (X, F, E, W, R, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^\uparrow}$ *surface consists exactly of six obligations:*

1. *carrier admission for the displayed source, graph, epigraph, schedule, readback, comparison, transport, replay, provenance, and local naming rows;*
2. *epigraph-window exactness, requiring every lower-bound read to appear as a finite (x, q) row in E after the schedule in W and readback in R have been replayed;*
3. *lower-real classifier stability, requiring lower refinements and rational-bound comparisons to be transported through the* $\mathsf{LowerReal}^\uparrow$ *rows in O;*

4. *located readback compatibility, requiring every inspected source or value window to pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *rows before a real-facing consumer can read it;*
5. hsame *and* Cont *replay, requiring all source, graph, epigraph, readback, comparison, and seal rows to move together under the displayed transport and continuation ledgers;*
6. *provenance and non-escape, stating that the package exports only the named finite rows and no host topology, quotient equality, semantic limit, open-cover compactness, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 1536.1. The packet lists exactly the source row, graph row, epigraph ledger, finite schedule, regular readback, comparison ledger, transport ledger, replay ledger, provenance row, and local name.

Epigraph-window exactness is read from E only after the requested source window has passed through W and the value readback through R. Thus a lower-bound claim is not an external predicate; it is a displayed finite row over the inspected source and rational bound.

Lower-real classifier stability is the role of O together with H. The lower-real comparisons, upper-side exclusions, and located endpoint checks move componentwise by hsame, and the continuation row C records the same inspection route after transport.

Located readback compatibility follows because every real-facing window is read through the $\mathsf{LocatedReal}^{\uparrow}$ rows before the terminal $\mathsf{Real}^{\uparrow}$ seal is available. The readback is therefore a finite stream and regular-rational route, not a host limit operation.

Finally, P and N package those rows as the local certificate. Since the carrier has no coordinate for the excluded principles, no consumer can obtain them from the $\mathsf{LowerSemicontinuous}^{\uparrow}$ NameCert surface. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousNameCertObligationSurface`

Theorem 1536.6 (LowerSemicontinuous epigraph window exactness)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, every lower-bound or epigraph read exposed by* E *is exact only along the finite window route*

$$W \longrightarrow R \longrightarrow E.$$

A consumer must first select the displayed source window in W*, then read the regular rational value data in* R*, and only then inspect the epigraph row* E*. The row* E *cannot be read as a host lower-set predicate, a semantic liminf assertion, quotient real equality, selected limits, countable choice, or ambient completeness.*

Proof. Project the carrier coordinates from Definition 1536.1. The epigraph row is one displayed coordinate of L, and the only finite schedule for inspecting it is W.

Apply Definition 1536.5. Its epigraph-window obligation places every lower-bound read after a scheduled window and a regular rational readback.

The route is therefore W, R, E inside the same BHist packet. Since the carrier contains no coordinate for the excluded host principles, the epigraph read cannot detach from that route. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphWindowExactness`

Theorem 1536.7 (LowerSemicontinuous located epigraph transport)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, the located lower-real and upper-real comparison rows recorded in* O *transport the displayed epigraph row* E *through* H *and* C*. A lower-bound read that*

has passed through W, R, E, O remains the same BEDC row after componentwise hsame *transport and displayed continuation replay; it cannot become a host lower-set predicate, a semantic liminf assertion, quotient equality, selected endpoints, countable choice, or ambient completeness.*

Proof. Start with the epigraph route of Definition 1536.6. It places every lower-bound read on W, R, E before the located comparison ledger can be used.

The row O then supplies the lower-real, upper-real, and located comparison data for that same displayed epigraph row. These comparisons are packet rows, not external topological predicates.

Transport and replay are componentwise. The row H moves E and O by hsame, and C replays the same finite inspection route, so the transported read is still the displayed W, R, E, O route. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedEpigraphTransport`

Theorem 1536.8 (LowerSemicontinuous epigraph located stability)**.** *For an accepted* LowerSemicontinuous$^{\uparrow}$ *carrier $L = (X, F, E, W, R, O, H, C, P, N)$, any lower-bound read transported through W, R, E, O, H, C remains a located lower-real epigraph row. It cannot become a host liminf, quotient equality, selected endpoint datum, countable-choice schedule, compactness principle, or ambient-completeness assertion.*

Proof. Use Definition 1536.6 to place the read on W, R, E and Definition 1536.7 to carry the same epigraph row through O, H, C.

The NameCert obligation surface records only finite windows, regular readback, located comparison rows, componentwise transport, replay, provenance, and local naming. None of these coordinates stores the excluded host principles, so the transported row remains a located lower-real epigraph row. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphLocatedStability`

Theorem 1536.9 (LowerSemicontinuous regular readback nonescape)**.** *Every lower-bound read from an accepted* LowerSemicontinuous$^{\uparrow}$ *carrier $L = (X, F, E, W, R, O, H, C, P, N)$ factors through*

$$W, R, E, O, H, C, P, N.$$

The readback route uses finite windows, regular-rational values, the epigraph row, located comparison rows, transport, replay, provenance, and local naming; it has no coordinate for host liminf, quotient equality, selected limits, countable choice, compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRegularReadbackNonescape`

Proof. The scheduled window row W fixes the inspected source window, and the regular-rational row R supplies the displayed value readback. By Definition 1536.6, the epigraph row E is readable only after those two rows are present.

Apply Definition 1536.7. The located comparison ledger O and structural rows H, C keep the lower-bound read on the displayed epigraph route under transport and replay.

Finally P and N source and name the same packet. Since no listed carrier row stores a host liminf operator, quotient equality, selected limit, choice schedule, compactness principle, or completeness theorem, the readback cannot escape the displayed route. □

Theorem 1536.10 (LowerSemicontinuous Real seal boundary)**.** *Any* Real$^{\uparrow}$*-facing consumer of an accepted* LowerSemicontinuous$^{\uparrow}$ *packet may read the terminal seal only after the* LowerReal$^{\uparrow}$, UpperReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, StreamName$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *rows have been displayed through E, O, W, R, H, C, P, N.*

The Real-facing boundary therefore records a finite lower-semicontinuity certificate rather than host topology, semantic liminf data, quotient real equality, selected representatives, countable choice, compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealBoundary`

Proof. Use the NameCert obligation surface of Definition 1536.5. It requires source windows, graph data, epigraph rows, finite schedules, regular readback, located comparison rows, transport, replay, provenance, and local naming.

The lower-real and upper-real rows are displayed inside E and O, while W and R supply the StreamName and RegSeqRat route used to inspect them. The located comparison surface cannot be skipped because Definition 1536.7 keeps the epigraph row tied to O, H, C.

Now apply Definition 1536.9. It forces every lower-bound read through W, R, E, O, H, C, P, N, so a Real-facing consumer reaches the terminal seal only after the displayed lower, upper, located, stream, and regular rows are visible. □

Theorem 1536.11 (LowerSemicontinuous located readback compatibility)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing output read must pass through the located comparison ledger* O *before the terminal real seal is available. Thus source windows, regular rational value rows, lower-real epigraph data, and upper-side comparison rows may be exported to a real-facing consumer only along*

$$X, F, W, R, E, O, H, C, P, N.$$

The output cannot bypass O *by invoking host real equality, selected endpoint representatives, semantic limits, countable choice, open-cover compactness, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedReadbackCompatibility`

Proof. Start with a real-facing consumer of L. The source, graph, schedule, readback, and epigraph coordinates are displayed as X, F, W, R, E, but none of them alone supplies a located readback.

Use Definition 1536.5. The located-readback obligation requires the route to pass through O before a real-facing output can be read.

Transport and replay then use only H and C, and the resulting route is packaged and named by P and N. No additional host coordinate is present, so the real-facing output remains the stated finite BEDC route. □

Theorem 1536.12 (LowerSemicontinuous transport replay exactness)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, componentwise* hsame *transport and displayed* Cont *replay preserve exactly the rows*

$$X, F, E, W, R, O, H, C, P, N.$$

They do not add a host topology row, an open-set lattice, quotient equality, semantic limit data, selected witnesses, countable choice, open-cover compactness, or ambient completeness. Every transported lower-semicontinuity read is therefore the same finite carrier route with the same local $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousTransportReplayExactness`

Proof. Read the transport and replay ledgers from the carrier. The H row records componentwise hsame only for the displayed coordinates, while C records continuation replay only for the same inspection routes.

Apply Definition 1536.5. Its replay obligation lists the source, graph, epigraph, readback, comparison, seal, provenance, and naming rows as the complete transport surface.

It remains to identify what cannot appear after replay. Because $L = (X, F, E, W, R, O, H, C, P, N)$ has no coordinate for the excluded host principles, H and C can only preserve the displayed rows and cannot introduce a new semantic route. □

Theorem 1536.13 (LowerSemicontinuous NameCert basis obligations)**.** *For an accepted* LowerSemicontinuous$^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, every root-unblock obligation that exposes lower-real epigraph data to a downstream consumer is a finite row of the same packet. The basis consists of the* LowerReal$^{\uparrow}$ *epigraph ledger* E*, the* StreamName$^{\uparrow}$ *schedule* W*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the* $UpperRealUp$ *and* LocatedReal$^{\uparrow}$ *comparison rows in* O*, the terminal* Real$^{\uparrow}$ *seal boundary,* BHist *histories,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

No basis obligation is discharged by host topology, a semantic liminf operator, quotient real equality, selected limits, countable choice, or ambient completeness. Epigraph and comparison rows are finite carrier data rather than host-theoretic evidence.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousNameCertBasisObligations`
↪ variant `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousNameCertObligationBasis`

Proof. Project the carrier in Definition 1536.1. The only epigraph data available to a consumer is the finite ledger E, and the only inspection route to it is through the scheduled windows W and regular rational readbacks R.

The comparison row O records lower-real, upper-real, and located-real readbacks before the terminal Real$^{\uparrow}$ seal is exposed. Thus every root-unblock read remains inside the displayed carrier coordinates.

The structural rows H, C, P, N transport, replay, package, and name the same finite route. Since the excluded host principles are absent from $L = (X, F, E, W, R, O, H, C, P, N)$, they cannot serve as basis obligations. □

Theorem 1536.14 (LowerSemicontinuous Real seal nonescape)**.** *Let* $L = (X, F, E, W, R, O, H, C, P, N)$ *be an accepted* LowerSemicontinuous$^{\uparrow}$ *carrier. Any* Real$^{\uparrow}$ *readback from* L *must pass through the* LowerReal$^{\uparrow}$ *epigraph row* E*, the* StreamName$^{\uparrow}$ *schedule* W*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the* UpperReal$^{\uparrow}$ *and* LocatedReal$^{\uparrow}$ *comparisons in* O*, the* BHist *history rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The Real$^{\uparrow}$ *seal is therefore a finite carrier boundary. It is not a route to host topology, a semantic limit, quotient real equality, selected limits, countable choice, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealNonescapeRoute`

Proof. Begin with a Real$^{\uparrow}$-facing consumer of L. By Definition 1536.1, the consumer can inspect only the displayed source, graph, epigraph, schedule, readback, comparison, transport, replay, provenance, and naming rows.

Apply Definition 1536.5. Its non-escape obligation places the terminal real seal downstream from epigraph-window exactness, lower-real classifier stability, located readback compatibility, and replay.

Hence a real readback cannot detach from E, W, R, O, H, C, P, N. The excluded principles have no coordinate in the carrier and no continuation row that could introduce them. □

Theorem 1536.15 (LowerSemicontinuous UpperReal handoff)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, every* $\mathsf{UpperReal}^{\uparrow}$ *comparison handed to a downstream lower-semicontinuity consumer is read from the same comparison ledger* O *that also records the* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readbacks. The handoff preserves the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *value windows,* BHist *histories,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming of the carrier.*

Thus upper-side comparison is finite BEDC data paired with lower-real epigraph rows. It is not an ambient order-completeness principle, a host topology assertion, a semantic upper envelope, quotient real equality, a selected limit, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the comparison ledger from Definition 1536.1. The row O is the only carrier coordinate that contains upper-real comparison data, and it is tied to the same lower-real and located-real checks used by epigraph reads.

The schedule and readback rows W and R determine which finite value windows the comparison may inspect. Transport through H and replay through C preserve those windows together with the epigraph and comparison rows.

The provenance and local name rows P, N certify this as a $\mathsf{LowerSemicontinuous}^{\uparrow}$ handoff. Since the carrier has no coordinate for the excluded host principles, the $\mathsf{UpperReal}^{\uparrow}$ consumer receives only the finite comparison surface stated above. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousUpperRealHandoff`

Theorem 1536.16 (Lower-real epigraph route)**.** *Every lower-real or real-seal consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier factors through the finite epigraph route*

$$X, F, W, R, E, O, H, C, P, N.$$

The consumer may inspect scheduled source windows, regular rational value readbacks, displayed lower-bound rows, lower-real refinements, located endpoint checks, componentwise transports, continuation replay, provenance, and local naming rows. It cannot bypass the epigraph ledger to read host topological openness, quotient real equality, a semantic liminf operator, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start from a real-facing consumer of the packet. The source row X and graph row F are visible only through the finite schedule W, so every value inspection begins with a displayed source window rather than an ambient neighbourhood.

The regular rational readback row R then supplies the finite value window. The epigraph ledger E records the lower-bound row (x, q) attached to that same inspection, and the comparison ledger O checks lower-real refinements, upper-side exclusions, and located endpoint compatibility.

Transport and replay are exhausted by H and C. They move the source, graph, readback, epigraph, comparison, and seal rows together, so the route remains the same after classifier transport.

The package and local name rows P, N record exactly this route for consumers. Since the excluded host principles are absent from the route and absent from the carrier coordinates, the lower-real epigraph handoff exposes only the finite BEDC rows stated above. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLowerRealEpigraphRoute`

Theorem 1536.17 (LowerSemicontinuous located readback stability)**.** *Let $L = (X, F, E, W, R, O, H, C, P, N)$ be an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier. Every epigraph-window comparison accepted by E and transported by H preserves the* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint checks in O before any* $\mathsf{Real}^{\uparrow}$*-facing consumer can read the transported lower bound. Equivalently, a consumer that receives a transported row (x, q) must pass through the same scheduled window in W, the same regular rational readback in R, and the located endpoint ledger in O before the terminal real-seal route is available.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedReadbackStability`

Proof. Project the carrier rows from Definition 1536.1. The only accepted lower-bound comparisons are the finite epigraph rows in E, and the only transport available for those rows is the componentwise hsame ledger H.

Apply Definition 1536.16. Its route places W and R before E, and places the lower-real, upper-real, and located-real comparison ledger O before any terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Thus the transported comparison cannot expose a bound by itself. The located endpoint checks in O remain on the route after transport, and C replays the same scheduled inspection for every Real-facing consumer. □

Theorem 1536.18 (LowerSemicontinuous Real seal nonescape)**.** *An accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet exports* $\mathsf{Real}^{\uparrow}$*-facing data only through the displayed rows*

$$E, W, R, O, H, C, P, N.$$

In particular, the packet cannot expose host topology, a semantic liminf operator, quotient real equality, a selected limit, countable choice, open-cover compactness, or ambient completeness as a route to the terminal real seal.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealTransportNonescape`

Proof. Read the export surface from Definition 1536.1. The terminal real-facing route is represented by epigraph rows, scheduled windows, regular rational readbacks, comparison rows, componentwise transport, continuation replay, package provenance, and the local name.

Apply Definition 1536.16. The theorem states that every lower-real or real-seal consumer factors through the finite epigraph route and cannot bypass it to host topological openness, quotient real equality, semantic liminf data, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

The listed excluded principles are not carrier coordinates and are not introduced by H or C. Therefore the $\mathsf{Real}^{\uparrow}$ seal row remains a finite BEDC export through E, W, R, O, H, C, P, N rather than a host-theoretic escape. □

Theorem 1536.19 (LowerSemicontinuous Baire-one source handoff)**.** *Let $L = (X, F, E, W, R, O, H, C, P, N)$ be an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier. Any* $\mathsf{BaireOneFunction}^{\uparrow}$ *source read that uses L as its lower-semicontinuous boundary must first pass through the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row F, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the epigraph ledger E, and the comparison ledger O before the structural rows H, C, P, N may hand the read to a* $\mathsf{Real}^{\uparrow}$*-facing consumer. The handoff supplies finite epigraph data only; it does not supply a host pointwise limit, quotient equality of functions, a selected approximating sequence, or semantic topology.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousBaireOneSourceHandoff`

Proof. Project the carrier from Definition 1536.1. The only map-facing coordinate is F, and it is tied to the scheduled windows W and regular rational readbacks R before a lower-bound row can be inspected.

Apply Definition 1536.5. Its epigraph-window exactness and located readback clauses force the $\mathsf{BaireOneFunction}^{\uparrow}$ source handoff through E, W, R, O rather than through an ambient pointwise-limit predicate.

Finally use Definition 1536.18. The structural rows H, C, P, N replay and name the same finite route before the $\mathsf{Real}^{\uparrow}$ seal is visible, so no selected approximating sequence, quotient function equality, or host topology is exported. □

Theorem 1536.20 (LowerSemicontinuous Real and regular-rational scope)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *carrier* $L = (X, F, E, W, R, O, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *scope is exactly the route that reads a finite schedule from* W*, regular rational value data from* R*, lower-bound evidence from* E*, comparison data from* O*, and then the transport, replay, provenance, and local name rows* H, C, P, N*. Thus a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the epigraph-ledger route, not a semantic liminf operation, quotient real equality, selected limit, countable choice, or ambient completeness.*

Proof. Begin with the root carrier route. The rows W and R determine the finite windows and regular rational values that can be read before any lower-bound claim is visible.

Use Definition 1536.16. It places the epigraph ledger E and comparison ledger O downstream of the schedule and readback rows, and upstream of any real-facing seal.

Now apply Definition 1536.17 and Definition 1536.18. Transport and replay preserve the same route, while the seal theorem excludes liminf semantics, quotient equality, selected limits, countable choice, and ambient completeness from the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scope. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealRegSeqRatScope`

Theorem 1536.21 (LowerSemicontinuous epigraph-ledger refusal)**.** *No accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet lets a consumer read* $\mathsf{BaireOneFunction}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *data while bypassing the epigraph rows* E, W, R, O, H, C, P, N*. A purported consumer that reaches a Baire-one source handoff, a continuous-map graph row, a regular rational readback, or a terminal real seal without those rows is refused by the local NameCert obligation surface.*

Proof. Read the purported consumer against the carrier. The graph row F has no standalone export path: every lower-semicontinuous read proceeds through scheduled windows, regular rational readbacks, epigraph rows, and comparison rows.

Apply Definition 1536.5. Carrier admission, epigraph-window exactness, located readback compatibility, replay, provenance, and non-escape are all obligations on the same packet.

Finally combine the Baire-one handoff and the Real–regular-rational scope theorems above. They identify the only visible source and seal routes with E, W, R, O, H, C, P, N, so any route omitting one of those rows has no accepted NameCert read. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphLedgerRefusal`

Definition 1536.22 (LowerSemicontinuous root carrier)**.** *The root packet for* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *is the accepted finite carrier*

$$L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$$

of Definition 1536.1, read with X as the displayed source row, F as the graph row, E as the lower-real epigraph ledger, W as the stream schedule, R as the regular rational readback row, O as the lower-real, upper-real, located-real, and real-seal comparison ledger, H as componentwise transport, C as displayed replay, P as provenance, and N as the local NameCert row. A root consumer may inspect only these rows and the routes certified by Definition 1536.5; it receives no separate topology carrier, semantic liminf operation, quotient real equality, selected limit, countable choice, open-cover compactness, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootCarrier rows exposed`

Theorem 1536.23 (LowerSemicontinuous real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of the root packet* $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ *factors through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. Concretely, a consumer first reads a stream window from W, then a regular rational value row from R, then a lower-real epigraph witness from E, then the located comparison data in O, and only then the terminal real-seal route recorded by O, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousReal handoff`

Proof. The root packet is the carrier of Definition 1536.22, so a consumer can start only from the rows $X, F, E, W, R, O, H, C, P, N$.

Apply Definition 1536.5. Its obligation surface places carrier admission, epigraph-window exactness, lower-real classifier stability, located readback compatibility, transport replay, provenance, and non-escape on the same local NameCert route.

Apply Definition 1536.16. That route orders the read through scheduled windows, regular rational readbacks, displayed lower-bound rows, lower-real refinements, located endpoint checks, transport, replay, provenance, and local naming.

Finally apply Definition 1536.17 and Definition 1536.18. They keep the located readback before the terminal real seal and exclude every host route not present in the finite packet. □

Theorem 1536.24 (LowerSemicontinuous open-sublevel route)**.** *Every sublevel or open-window read of the root packet* $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ *is a displayed epigraph route through* E, W, R, O, H, C, P, N*. It is not a read of host topological openness: the only available witness is a finite lower-bound row transported through the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison ledgers and replayed by the same continuation route.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousOpenSublevelRoute`

Proof. Start with a sublevel or open-window consumer of the root packet. By Definition 1536.22, the packet has no coordinate for a topology carrier or open-set lattice, so the consumer must read one of the displayed finite rows.

Use Definition 1536.16. The theorem identifies the lower-real route as

$$X, F, W, R, E, O, H, C, P, N.$$

Thus an open-window read can only mean that a scheduled source window and a regular rational value row have reached a displayed epigraph witness in E.

The located readback stability theorem keeps the comparison in O on the same route after transport. The real-seal nonescape theorem then prevents the consumer from replacing that displayed epigraph route by host topology, semantic liminf data, quotient real equality, choice, compactness, or ambient completeness. □

Theorem 1536.25 (LowerSemicontinuous epigraph window induction)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet*

$$L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N),$$

every finite epigraph inspection window is generated by induction over the displayed BHist *rows* W, R, E, O, H, C, P, N*. The base window reads the source row* X *and graph row* F *before exposing one scheduled stream window, and each successor window extends only by a* $\mathsf{StreamName}^{\uparrow}$ *schedule row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *value row, an epigraph row in* E*, and the comparison data in* O*. No induction step may replace this route by host topology, semantic liminf data, quotient real equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphWindowInduction`

Proof. Start with the root carrier of Definition 1536.22. The available rows are exactly $X, F, E, W, R, O, H, C, P, N$, so the induction base can expose only the source row, graph row, and one scheduled finite window.

For the successor step, use Definition 1536.24. It identifies an open-window read with a displayed epigraph route through E, W, R, O, H, C, P, N, not with a host open-set predicate.

The kernel-source obligations in Definition 1536.5 keep each added window inside the same local NameCert scope. Transport and replay through H and C preserve the finite row order, while P and N source and name the same packet.

Finally apply the real handoff of Definition 1536.23. Any Real-facing endpoint of the induction remains downstream of $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 1536.26 (LowerSemicontinuous located epigraph transport induction)**.** *Let*

$$L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$$

be an accepted $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet. Transporting a finite lower-bound row through the* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{LowerReal}^{\uparrow}$ *comparison ledgers is an induction over the same* BHist *route: each step first reads the stream schedule* W*, then the regular rational value row* R*, then the lower-bound witness in* E*, and finally the located comparison data in* O*. The transported row remains a displayed epigraph row and never becomes parameter-equivalence transport, quotient real equality, semantic topology, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedEpigraphTransportInduction`

Proof. Read the located comparison ledger from Definition 1536.22. The row O is the only coordinate carrying lower-real, upper-real, located-real, and real-seal comparison data.

The induction base is a single lower-bound row already exposed by E after the schedule and readback rows W and R. The route theorem Definition 1536.16 places this row before the located comparison ledger.

For the induction step, apply Definition 1536.17. It transports the same regular rational readback and lower-bound row through the located comparison surface without introducing a new carrier or a parameter-equivalence argument.

The NameCert scope and real handoff then close the route. Use Definition 1536.5 and Definition 1536.23 to keep the transported row inside W, R, E, O, H, C, P, N before any terminal $\mathsf{Real}^{\uparrow}$ read is visible. □

Theorem 1536.27 (LowerSemicontinuous Real seal window exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read generated from an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet is exhausted by the finite window route*

$$X, F, W, R, E, O, H, C, P, N.$$

In particular, the terminal Real seal can be reached only after the $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, lower-real epigraph row, and located comparison ledger have been displayed. It exports no hidden Baire-one pointwise limit, quotient real equality, selected approximating sequence, countable choice, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealWindowExhaustion`

Proof. Begin with Definition 1536.23. It states that every Real-facing read factors through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Use the epigraph window induction above to inspect the finite schedule. Each window reaches the epigraph row through W, R, E and is replayed through H, C, P, N inside the same BHist packet.

Next apply the located transport induction. It keeps lower-bound rows in the comparison ledger O and rules out parameter-equivalence transport, quotient real equality, and semantic topology as alternate exits.

Finally use Definition 1536.18. The terminal seal is exhausted by the displayed route, so the absent pointwise-limit, selection, compactness, and completeness principles are not exported by the root packet. □

Theorem 1536.28 (LowerSemicontinuous epigraph public export)**.** *Every public lower-semicontinuity read from an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet is exported through the finite route*

$$X, F, W, R, E, O, H, C, P, N.$$

The export combines the root epigraph basis, located readback route, transport replay, and terminal $\mathsf{Real}^{\uparrow}$ *seal boundary. It is a public BEDC NameCert route over displayed* BHist*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows, not a semantic topology theorem, host liminf operator, quotient-real equality, countable-choice schedule, compactness principle, or ambient completeness assertion.*

Proof. Start with the root epigraph basis of Definition 1536.45. It identifies the consumer-visible lower-real basis as the finite source, graph, schedule, readback, epigraph, comparison, transport, replay, provenance, and naming route.

Apply located readback and transport compatibility, using Definition 1536.46 together with Definition 1536.26. These results keep each lower-bound read on the same W, R, E, O, H, C, P, N path when it is transported or replayed.

Finally use the Real-seal exhaustion results, Definition 1536.27 and Definition 1536.51. They force the public endpoint through the terminal $\mathsf{Real}^{\uparrow}$ seal boundary while excluding host topology, semantic liminf data, quotient real equality, countable choice, compactness, and ambient completeness. The exported statement is therefore the finite NameCert route above. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphPublicExport`

Theorem 1536.29 (LowerSemicontinuous epigraph sibling route)**.** *The public export route of Definition 1536.28 factors every lower-real comparison through the* $\mathsf{Epigraph}^{\uparrow}$ *sibling of chapter 6224 before a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer inspects sublevel or superlevel rows. The sibling supplies the domain row, value row, lower-bound row, and comparison row consumed by the lower-semicontinuity packet, while* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *remain terminal readback rows.*

Proof. Start with the public export theorem Definition 1536.28. Its route names the finite source, graph, schedule, readback, epigraph, comparison, transport, replay, provenance, and naming rows used by public lower-semicontinuity consumers.

Read the epigraph coordinate through chapter 6224. The $\mathsf{Epigraph}^{\uparrow}$ sibling is the packet that supplies the displayed domain, value, lower-bound, and comparison rows before lower-semicontinuity sublevel or superlevel rows are inspected.

The located readback and Real-seal rows stay downstream. Thus $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ remain terminal readback boundaries, while the horizontal route through $\mathsf{Epigraph}^{\uparrow}$ provides the comparison lattice consumed by $\mathsf{LowerSemicontinuous}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphSiblingRoute`

Definition 1536.30 (LowerSemicontinuous closed-superlevel carrier)**.** *A closed-superlevel row for the root packet $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ is a finite* BHist *packet*

$$S = (q, E_q, O_q, B_q, H_q, C_q, P_q, N_q)$$

where q is the rational threshold being tested, E_q is the lower-real epigraph window read from E at threshold q, O_q is the $\mathsf{Real}^{\uparrow}$*-seal and* $\mathsf{LocatedReal}^{\uparrow}$ *comparison route attached to the same threshold, B_q is the* BHist *evidence that the complement of the displayed sublevel window remains stable under the shown* hsame *and* Cont *transports, and H_q, C_q, P_q, N_q transport, replay, source, and name the same threshold row. The row records a constructive complement boundary for the finite lower-real window; it does not assert classical topological closedness of a host superlevel set.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousClosedSuperlevelCarrier surface`

Theorem 1536.31 (LowerSemicontinuous closed-superlevel stability)**.** *For every accepted closed-superlevel row*

$$S = (q, E_q, O_q, B_q, H_q, C_q, P_q, N_q)$$

attached to L_{root}, transport by the displayed hsame *rows and replay by the displayed* Cont *route preserves the finite superlevel complement boundary determined by q. Equivalently, a consumer that avoids the open-sublevel window at threshold q may keep that avoidance only through the lower-real epigraph row E_q, the located readback route O_q, the* $\mathsf{Real}^{\uparrow}$ *seal data, and the stability evidence B_q; it cannot replace them by host closedness, semantic liminf data, quotient real equality, countable choice, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousClosedSuperlevel stability`

Proof. Project the closed-superlevel carrier. Its threshold is the displayed rational q, and all boundary information is stored in $E_q, O_q, B_q, H_q, C_q, P_q, N_q$. There is no coordinate for a host topological closed set.

Apply Definition 1536.16. The threshold row E_q is read only through the lower-real epigraph window attached to the original root packet, after the scheduled source windows and regular rational value rows have been displayed.

Use Definition 1536.17. The located comparison row O_q remains on the same route under the componentwise hsame transport and the displayed Cont replay. Hence the complement of the open-sublevel window is carried by B_q as finite transport evidence, not as a detached topological predicate.

Finally apply Definition 1536.24 together with Definition 1536.18. The open-sublevel route identifies the boundary being avoided, and the real-seal nonescape row prevents the consumer

from substituting classical closedness, semantic liminf data, quotient real equality, countable choice, compactness, or ambient completeness for the displayed finite boundary. □

Definition 1536.32 (LowerSemicontinuous superlevel threshold surface)**.** *For a rational threshold q attached to an accepted closed-superlevel row*

$$S = (q, E_q, O_q, B_q, H_q, C_q, P_q, N_q),$$

the superlevel threshold surface is the finite reusable read

$$(q, E_q, O_q, B_q, H_q, C_q, P_q, N_q).$$

It records the threshold, lower-real epigraph window, located comparison and $\mathsf{Real}^{\uparrow}$*-seal route, complement-boundary evidence, componentwise transport, replay, provenance, and local naming needed by any closed-superlevel consumer.*

Lean4 (definition) `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousSuperlevelThresholdSurface`
Lean4 (definition) `BEDC.Derived.LowerSemicontinuousUp.lowerSemicontinuousSuperlevelThresholdSurface`
Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousSuperlevelThresholdSurface exact`

Theorem 1536.33 (LowerSemicontinuous closed-superlevel threshold exhaustion)**.** *Every closed-superlevel consumer of an accepted threshold surface*

$$(q, E_q, O_q, B_q, H_q, C_q, P_q, N_q)$$

must exhaust $E_q, O_q, B_q, H_q, C_q, P_q, N_q$ before reading a boundary at q. The consumer cannot replace those rows by host closedness, semantic liminf data, quotient real equality, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousClosedSuperlevelThresholdExhaustion`

Proof. Start from the threshold surface of Definition 1536.32. Its only analytic rows are the lower-real epigraph window E_q, located comparison and Real-seal route O_q, and complement-boundary evidence B_q.

Apply closed-superlevel stability, Definition 1536.31. It preserves the finite complement boundary only through the displayed transport and replay rows, so H_q and C_q must be read with the same threshold.

The provenance and local naming rows P_q, N_q bind the threshold row to the original packet. Since no coordinate stores host topology, semantic liminf data, quotient equality, choice, compactness, or ambient completeness, the listed rows exhaust every boundary read at q. □

Theorem 1536.34 (LowerSemicontinuous Baire-one superlevel handoff)**.** *Every Baire-one or lower-real consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet receives the same superlevel threshold surface*

$$(q, E_q, O_q, B_q, H_q, C_q, P_q, N_q)$$

when it asks for a closed-superlevel read at threshold q. The handoff contains no host topology coordinate, semantic liminf coordinate, choice coordinate, quotient real equality, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousBaireOneSuperlevelHandoff`

Proof. Project the closed-superlevel carrier and the threshold surface. Both expose the same rows $q, E_q, O_q, B_q, H_q, C_q, P_q, N_q$, so there is no second Baire-one or lower-real boundary object to receive.

Use the threshold exhaustion theorem. It forces the consumer to read the epigraph window, located comparison and seal route, complement-boundary evidence, transport, replay, provenance, and local naming before a boundary at q is available.

Finally apply the Baire-window factorization theorem, Definition 1536.35. Baire-one and lower-real consumers already factor through the finite W, R, E, O route, and the threshold surface is the closed-superlevel read of that same route rather than a host topology, liminf, choice, quotient, or completeness coordinate. □

Theorem 1536.35 (LowerSemicontinuous Baire-window factorization)**.** *Every Baire-one or lower-real consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet* $L = (X, F, E, W, R, O, H, C, P, N)$ *factors its finite request through the route*

$$W \longmapsto R \longmapsto E \longmapsto O$$

before any $\mathsf{Real}^{\uparrow}$*-facing seal is read. The consumer cannot replace that route by host topology, semantic liminf data, quotient equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousBaireWindowRealSealFactorization`

Proof. Project the carrier rows from Definition 1536.1. The finite request first chooses the inspected windows in W, then reads regular rational data from R, then reaches the lower-real epigraph ledger E, and only then enters the located comparison row O.

Use epigraph exactness and located readback compatibility, Definition 1536.6 and Definition 1536.11. These theorems keep lower-bound reads and real-facing comparison data on the displayed route rather than on a host liminf or topology predicate.

Finally apply closed-superlevel stability from Definition 1536.31. The open-sublevel and closed-superlevel boundary reads remain finite BEDC rows, so a Baire-one or lower-real consumer receives exactly the W, R, E, O factorization before any terminal seal is available. □

Theorem 1536.36 (LowerSemicontinuous kernel scope)**.** *Every lower-semicontinuous epigraph, located-readback, or Real-seal consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must route through the visible* BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the carrier. The route may read epigraph windows, open-sublevel boundaries, closed-superlevel stability, transport replay, and Real handoff data, but it cannot replace those rows by host topology, semantic liminf data, quotient real equality, countable choice, compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousKernelScope`

Proof. Start with the NameCert obligation surface of Definition 1536.5. It lists the finite source, graph, epigraph, window, readback, located comparison, transport, replay, provenance, and local naming rows that any consumer must expose.

Apply epigraph exactness and the epigraph-window route, Definition 1536.16 and Definition 1536.24. These results keep lower-real and sublevel reads inside the displayed BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, and $\mathsf{UpperReal}^{\uparrow}$ rows.

Next use located-readback stability and transport replay. The theorems Definition 1536.17 and Definition 1536.12 keep the $\mathsf{LocatedReal}^{\uparrow}$, hsame, and Cont rows visible when a comparison is moved or replayed.

Finally apply the Real handoff and nonescape statements, Definition 1536.23 and Definition 1536.18. They force every Real-facing read through the same Pkg and NameCert rows and exclude host topology, semantic liminf data, quotient real equality, countable choice, compactness, and ambient completeness. □

Theorem 1536.37 (LowerSemicontinuous root obligation split)**.** *Every lower-real epigraph window, located comparison row, and* $\mathsf{Real}^{\uparrow}$ *seal read consumed by a Baire-one or lower-real reader of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must factor through the displayed route*

$$W \longrightarrow R \longrightarrow E \longrightarrow O \longrightarrow P \longrightarrow N$$

before any consumer can inspect it. The route first displays source windows, then regular rational value rows, then lower-real epigraph witnesses, then located comparison and Real-seal data, then provenance and local naming; it exports no host topology, semantic liminf operator, quotient real equality, choice schedule, compactness principle, or ambient completeness.

Proof. Start with the NameCert obligation surface, Definition 1536.5. It forces the source windows, value rows, epigraph rows, comparison rows, provenance, and local name to be visible for the accepted packet.

Apply the lower-real epigraph route, Definition 1536.16. That route orders the lower-real read from W to R to E before any located comparison row can be consumed.

Finally use the kernel scope theorem, Definition 1536.36. It keeps the located comparison, Real-seal read, provenance, and local naming rows inside the same finite packet boundary, so every Baire-one or lower-real consumer receives exactly the displayed W, R, E, O, P, N route and none of the excluded host or completeness principles. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootObligationSplit`

Theorem 1536.38 (LowerSemicontinuous superlevel tail obligation)**.** *Every closed-superlevel tail consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must first expose the threshold row, lower-real epigraph window, located comparison route,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows carried by the same finite packet. The consumer cannot substitute host topology, semantic liminf data, quotient real equality, countable choice, open-cover compactness, or ambient completeness for those rows.*

Proof. Start with the closed-superlevel threshold exhaustion theorem, Definition 1536.33. It forces any boundary read at a displayed rational threshold to consume $E_q, O_q, B_q, H_q, C_q, P_q, N_q$ rather than a host closedness predicate.

Next apply the Baire-window factorization theorem, Definition 1536.35. It keeps lower-real and Baire-one requests on the finite route through the displayed window, readback, epigraph, and located-comparison rows before any Real-facing seal is read.

Use the kernel scope theorem, Definition 1536.36. The BHist, hsame, Cont, Pkg, and NameCert rows remain part of the same packet when the threshold data is transported or replayed.

Finally invoke root obligation split, Definition 1536.37. It orders the exposed route through the lower-real epigraph window, located comparison row, Real seal, provenance, and local naming, so the listed host and completeness principles have no coordinate from which to enter the superlevel tail. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousSuperlevelTailObligation`

Theorem 1536.39 (LowerSemicontinuous root epigraph carrier)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet*

$$L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N),$$

there is a finite root epigraph subcarrier

$$L_{\mathrm{epi}} = (X, F, W, R, E, O, H, C, P, N)$$

whose rows are read in that order. The row W *displays the finite* $\mathsf{StreamName}^{\uparrow}$ *inspection window,* R *displays the* $\mathsf{RegSeqRat}^{\uparrow}$ *value readback,* E *displays the lower-real epigraph ledger,* O *displays the* $\mathsf{LowerReal}^{\uparrow}$–$\mathsf{UpperReal}^{\uparrow}$ *comparison and located* $\mathsf{Real}^{\uparrow}$ *seal boundary,* H *records componentwise* hsame *transport,* C *records displayed* Cont *replay,* P *records* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$ *row. The carrier has no host liminf predicate, quotient real equality, selected approximating sequence, countable choice, open-cover compactness, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphCarrier`
↪ variant `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRoot epigraph carrier`

Proof. Project the rows of the root packet. The carrier definition already displays $X, F, E, W, R, O, H, C, P, N$, and the reordered subcarrier only makes the consumer route explicit: source and graph first, then finite window, rational readback, epigraph evidence, comparison, transport, replay, provenance, and local name.

The finite epigraph window is obtained by induction on the displayed window rows. The base read sees the source and graph rows; the successor read adds one $\mathsf{StreamName}^{\uparrow}$ schedule row, one $\mathsf{RegSeqRat}^{\uparrow}$ value row, and the corresponding epigraph witness in E.

Transport and replay do not add analytic data. They carry the same rows through H and C, with P and N sourcing and naming the accepted packet.

The exclusions follow from the carrier inventory. Since no coordinate contains a host liminf predicate, quotient equality, selector, choice row, compactness theorem, or completeness witness, no consumer can extract one from the subcarrier. □

Theorem 1536.40 (LowerSemicontinuous regular readback handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback consumed from* L_{epi} *reaches a* $\mathsf{LocatedReal}^{\uparrow}$ *handoff only through the displayed route*

$$W \longrightarrow R \longrightarrow E \longrightarrow O \longrightarrow H \longrightarrow C \longrightarrow P \longrightarrow N.$$

The comparison row O *keeps the lower-real and upper-real boundary attached to the same finite readback, so the handoff exports located comparison data and not semantic topology, hidden lower envelopes, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, or a detached completion principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootRegularReadbackHandoff`

Proof. Read the finite schedule row W and the regular rational value row R. These are the only readback rows displayed before the epigraph ledger is inspected.

Apply the lower-real epigraph route and located readback stability already present in the chapter. The epigraph row E is reached from the scheduled value read, and O then compares the same lower-real and upper-real boundary before the located handoff is visible.

The remaining rows preserve that route. Componentwise hsame transport through H, displayed Cont replay through C, Pkg provenance through P, and the local $\mathsf{NameCert}$ row N keep the handoff attached to the same readback.

Thus the consumer receives only the finite located comparison surface. The absent semantic topology, envelope, quotient equality, choice, and completion coordinates cannot be supplied by this handoff. □

Theorem 1536.41 (LowerSemicontinuous Real seal boundary)**.** *For every accepted root epigraph subcarrier L_{epi}, a terminal* $\mathsf{Real}^{\uparrow}$ *seal can be read only after the finite epigraph window,* $\mathsf{LowerReal}^{\uparrow}$–$\mathsf{UpperReal}^{\uparrow}$ *comparison row,* $\mathsf{LocatedReal}^{\uparrow}$ *handoff,* BHist *transport,* hsame *ledger,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row have all been displayed. The seal boundary exports no Baire-one pointwise limit, quotient real equality, selected representative, countable choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Proof. Start from the subcarrier of Definition 1536.39. Its Real-facing route is finite because every row is already part of the accepted BHist packet.

Use the regular readback handoff of Definition 1536.40. It places the terminal seal after the finite window W, rational readback R, epigraph row E, comparison row O, transport row H, replay row C, provenance row P, and local name N.

The seal is therefore a boundary row, not an analytic limit principle. It confirms that the displayed located comparison is ready for Real-facing consumers while refusing the absent pointwise-limit, quotient-equality, selection, choice, compactness, and completeness data. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootRealSealBoundary`

Theorem 1536.42 (LowerSemicontinuous root epigraph obligation rows)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet exposes, as one finite surface, the lower-real row, epigraph row, located comparison row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, the consumer route is*

$$X, F, W, R, E, O, H, C, P, N,$$

where R is the regular readback, E is the lower-real epigraph ledger, O is the located comparison boundary, and the terminal Real seal is downstream of those rows. No row in this surface is a host epigraph closure theorem, quotient real equality, selected limit, countable-choice schedule, open-cover compactness proof, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphRows`

Proof. Project the root epigraph carrier of Definition 1536.39. It already displays the source, graph, finite window, regular readback, epigraph ledger, comparison row, transport, replay, provenance, and local naming rows.

Apply the regular readback handoff of Definition 1536.40. This places the $\mathsf{RegSeqRat}^{\uparrow}$ row before the epigraph and located comparison rows are consumed.

Use the Real seal boundary of Definition 1536.41. The terminal seal is visible only after the lower-real epigraph and located comparison rows, so the listed exclusions cannot be exported by the accepted packet. □

Theorem 1536.43 (LowerSemicontinuous root Real-seal route)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet passes through the regular readback row and epigraph rows before the Real seal is visible. Equivalently, the read must expose W, R, E, O, H, C, P, N in that order, and the terminal seal cannot be consumed directly from the source row, graph row, or an ambient completion principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootRealSealRoute`

Proof. Begin with the regular readback handoff Definition 1536.40. It forces the finite window W and regular rational readback R to appear before the epigraph ledger E and located comparison row O.

Apply the obligation rows theorem Definition 1536.42. The lower-real and epigraph rows are part of the same finite surface as the located comparison row.

Finally use the Real seal boundary Definition 1536.41. Since the seal is terminal for this route, a Real-facing consumer cannot bypass the regular readback and epigraph rows by appealing to the source row, graph row, or an ambient completion object. □

Theorem 1536.44 (LowerSemicontinuous root epigraph nonescape)**.** *No accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet exports host epigraph closure, quotient real equality, selected limits, countable choice, or ambient completeness. Every export is exhausted by the finite rows* $X, F, W, R, E, O, H, C, P, N$ *and by the downstream Real seal boundary already attached to those rows.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphNonescape`

Proof. Project the finite row surface of Definition 1536.42. The only analytic-facing data available to a consumer are the scheduled window, regular readback, epigraph ledger, located comparison row, and terminal seal boundary.

Use the Real-seal route of Definition 1536.43. It prevents a Real-facing read from detaching from W, R, E, O, H, C, P, N.

The carrier exclusions in Definition 1536.39 then finish the nonescape claim. The packet has no coordinate for host closure, quotient equality, selected limit, countable choice, or ambient completeness, so those objects cannot be returned as $\mathsf{LowerSemicontinuous}^{\uparrow}$ evidence. □

Theorem 1536.45 (LowerSemicontinuous root epigraph basis)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet carries its lower-real epigraph basis through the finite rows*

$$X, F, W, R, E, O, H, C, P, N.$$

The basis row is generated by scheduled source windows, regular rational readbacks, and displayed lower-bound witnesses; it is not a host liminf predicate, quotient real equality, selected approximating sequence, countable choice schedule, open-cover compactness proof, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphBasis`

Proof. Project the root epigraph carrier of Definition 1536.39. The only rows before the comparison ledger are the source, graph, finite window, regular rational readback, and lower-real epigraph rows.

Apply Definition 1536.40. It keeps each readback attached to the same epigraph witness before the located comparison row is consumed.

Finally use Definition 1536.41. The terminal seal is downstream of the displayed basis, so no excluded host analytic object can be exported by the root packet. □

Theorem 1536.46 (LowerSemicontinuous root located readback)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet, every located readback of a lower-bound row factors through the same scheduled window* W*, regular rational value row* R*, epigraph ledger* E*, comparison row* O*, transport row* H*, replay row* C*, provenance row* P*, and local name* N*. The readback cannot detach into a semantic topology, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected representative, countable choice, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedReadback`

Proof. Start with the regular readback handoff of Definition 1536.40. It places W, R, E, O, H, C, P, N in the exact order consumed by a located readback.

The comparison row O is reached only after the lower-bound witness in E. Transport through H and replay through C preserve that same finite route rather than creating a second located-real carrier.

The Real seal boundary of Definition 1536.41 excludes semantic topology, quotient equality, selection, choice, and ambient completeness from the endpoint. □

Theorem 1536.47 (LowerSemicontinuous root located threshold exhaustion)**.** *For an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet, every located threshold read from the accepted epigraph factors through the finite route*

$$W \longmapsto R \longmapsto E \longmapsto O \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Here W is the $\mathsf{StreamName}^{\uparrow}$ *window, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the lower-real epigraph ledger, and O is the located comparison boundary before the structural rows. The route exports no host liminf predicate, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable-choice schedule, open-cover compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedThresholdExhaustion`

Proof. First use the regular readback handoff of Definition 1536.40. It places the scheduled window W and regular rational readback R before the lower-real epigraph ledger E and the located comparison boundary O.

Next apply the located readback theorem of Definition 1536.46. A threshold read accepted by the root epigraph packet therefore remains on W, R, E, O, H, C, P, N and cannot detach into a semantic topology or a selected approximating sequence.

Finally apply the Real seal boundary of Definition 1536.41. Since the terminal boundary is downstream of the same finite route, quotient real equality, countable choice, open-cover compactness, and ambient Real completeness have no export row. □

Theorem 1536.48 (Lower semicontinuous epigraph source scope)**.** *Every accepted root epigraph consumer of a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet factors through the concrete rows*

$$X, F, W, R, E, O, H, C, P, N.$$

Here W is the finite $\mathsf{StreamName}^{\uparrow}$ *inspection window, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, E is the lower-real epigraph evidence, O is the* $\mathsf{LowerReal}^{\uparrow}$–$\mathsf{UpperReal}^{\uparrow}$ *located comparison and terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, and H, C, P, N are the componentwise transport, replay, provenance, and local NameCert rows. Thus the consumer is source-scoped by the displayed stream-name and rational-readback route; it has no coordinate for semantic topology, host liminf, quotient real equality, selected representatives, countable choice, open-cover compactness, or ambient completion.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphSourceScope`

Proof. Begin with the root epigraph carrier of Definition 1536.39. It displays the only source rows available to an epigraph consumer: the source row X, graph-facing row F, finite window W, rational readback R, lower-real epigraph row E, comparison row O, and structural rows H, C, P, N.

Next use the regular readback handoff of Definition 1536.40. This keeps the $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the same lower-bound evidence before any located endpoint is visible.

Apply the Real seal boundary of Definition 1536.41. The row O is therefore a displayed $\mathsf{LowerReal}^{\uparrow}$–$\mathsf{UpperReal}^{\uparrow}$ comparison with a terminal $\mathsf{Real}^{\uparrow}$ boundary, not a hidden analytic completion principle.

Finally use the located readback theorem of Definition 1536.46. Every accepted located read of the epigraph endpoint has already passed through W, R, E, O, H, C, P, N, so the root epigraph consumer factors through the listed source-scope route and cannot export any of the excluded host objects. □

Theorem 1536.49 (LowerSemicontinuous epigraph to LocatedSet handoff)**.** *Any accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph read that is consumed as a* $\mathsf{LocatedSet}^{\uparrow}$ *packet must pass through the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, the root epigraph basis, the* $\mathsf{LocatedSet}^{\uparrow}$ *route,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The handoff exports no host topology, semantic liminf operator, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphLocatedSetHandoff`

Proof. Start with the root epigraph carrier of Definition 1536.39. It displays the finite source, graph, window, readback, epigraph, comparison, transport, replay, provenance, and local naming rows.

Use the regular readback handoff of Definition 1536.40 and the root epigraph basis of Definition 1536.45. Together they keep every $\mathsf{RegSeqRat}^{\uparrow}$ read attached to the same lower-bound row before any located consumer sees the endpoint.

Apply the Real seal boundary of Definition 1536.41. The terminal seal is visible only after the epigraph basis and located comparison rows have been displayed.

Finally use the located readback theorem above. A $\mathsf{LocatedSet}^{\uparrow}$ consumer therefore receives the same finite route through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows, and the excluded host analytic objects remain outside the packet. □

Theorem 1536.50 (LowerSemicontinuous epigraph to LocatedSet exactness)**.** *An accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph read becomes a* $\mathsf{LocatedSet}^{\uparrow}$ *packet exactly when the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *value row* R*, lower-real epigraph ledger* E*, upper/lower comparison row* O*,* $\mathsf{LocatedReal}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary,* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* NameCert *row* N *are all displayed in the same finite route. The exactness route exports no host topology, semantic liminf operator, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphLocatedSetExactness`

Proof. Start with the root epigraph carrier of Definition 1536.39. It supplies the scheduled window, regular rational value row, epigraph ledger, comparison row, transport, replay, provenance, and local name as one finite packet.

Use the regular readback handoff and the root located readback theorem. These force the $\mathsf{RegSeqRat}^{\uparrow}$ value row and every $\mathsf{LocatedReal}^{\uparrow}$ endpoint to pass through the same W, R, E, O, H, C, P, N route before a located consumer can read it.

Apply the Real seal boundary of Definition 1536.41. The seal is terminal for the displayed epigraph comparison, so quotient equality, selected representatives, countable choice, compactness, and ambient completeness have no export coordinate.

Finally use the epigraph-to-$\mathsf{LocatedSet}^{\uparrow}$ handoff of Definition 1536.49. The forward direction reads the listed rows before forming the located-set packet; the reverse direction projects those same rows from any accepted located-set consumption of the epigraph read. Thus the handoff is exact for the displayed finite route and for no hidden host object. □

Theorem 1536.51 (LowerSemicontinuous root nonescape)**.** *Every root-level consumer of the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *epigraph packet is exhausted by the displayed source, graph, finite-window, readback, epigraph, located-comparison, transport, replay, provenance, and local naming rows. The packet exports no host topology, semantic liminf operator, quotient real equality, selected sequence, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootNonescapeStrengthened`

Proof. Use the root epigraph basis theorem to identify the consumer-visible rows. They are all part of the finite BHist packet and all pass through the local NameCert route.

The located readback theorem then accounts for every Real-facing endpoint. A consumer reaches the endpoint only after W, R, E, O, H, C, P, N have been displayed.

Since the excluded host objects have no coordinate in the carrier and no replay row, the root packet cannot export them. □

Theorem 1536.52 (LowerSemicontinuous finite epigraph threshold exhaustion)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet exhausts all displayed finite threshold reads through*

$$X, F, W, R, E, O, H, C, P, N.$$

Any lower-bound threshold visible to a $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer must appear in the scheduled* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window rows before the epigraph ledger is consumed. The packet exports no semantic liminf predicate, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice, open-cover compactness, or ambient Real completeness.*

Proof. Start with the root epigraph carrier of Definition 1536.39. Its visible threshold route is exactly the source row X, graph row F, scheduled window row W, regular rational readback R, epigraph ledger E, comparison row O, transport H, replay C, provenance P, and local name N.

Apply the regular readback handoff of Definition 1536.40. A threshold cannot reach a lower-real, upper-real, located-real, or Real-facing consumer until the scheduled $\mathsf{StreamName}^{\uparrow}$ window and matching $\mathsf{RegSeqRat}^{\uparrow}$ value row have already been displayed.

Use the Real seal boundary of Definition 1536.41. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of W, R, E, O, H, C, P, N, so any visible lower-bound threshold must already be present in those finite rows before the seal is read.

Finally apply root nonescape. Since the carrier has no semantic liminf coordinate, quotient Real equality, selected approximating sequence, countable-choice schedule, open-cover compactness proof, or ambient completeness primitive, the displayed finite threshold rows exhaust every accepted epigraph read. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousFiniteEpigraphThresholdExhaustion`

Theorem 1536.53 (LowerSemicontinuous root epigraph closure package)**.** *The root epigraph carrier, regular readback handoff,* $\mathsf{Real}^{\uparrow}$ *seal boundary, epigraph basis, located readback,* $\mathsf{LocatedSet}^{\uparrow}$ *handoff, exactness, nonescape, finite threshold exhaustion, and* $\mathsf{LocatedSet}^{\uparrow}$ *threshold route assemble the* obligation *surface for* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Every accepted root epigraph consumer reads the finite route*

$$X, F, W, R, E, O, H, C, P, N$$

or its threshold subroute

$$W, R, E, O, H, C, P, N$$

before consuming the lower-real, upper-real, located-real, located-set, or Real-facing endpoint. The package exports no semantic liminf predicate, quotient $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice, open-cover compactness, or ambient Real completeness.*

Proof. First assemble the carrier and readback spine. The root epigraph carrier of Definition 1536.39 displays the finite rows $X, F, W, R, E, O, H, C, P, N$, and the regular readback handoff of Definition 1536.40 forces every $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ read to pass through W, R, E, O, H, C, P, N before any located endpoint is consumed.

Next attach the Real-facing and epigraph-facing boundaries. The Real seal boundary of Definition 1536.41 places the terminal $\mathsf{Real}^{\uparrow}$ seal after the same finite rows, while the root epigraph basis of Definition 1536.45 identifies the lower-real epigraph rows generated by the scheduled windows and regular rational readbacks.

Now pass to located consumption. The root located readback theorem Definition 1536.46 keeps every located endpoint on the displayed route, the $\mathsf{LocatedSet}^{\uparrow}$ handoff of Definition 1536.49 gives the located-set consumer, and exactness from Definition 1536.50 says that consuming the located-set packet is equivalent to displaying those rows in one finite route.

Finally close the obligation package. Root nonescape from Definition 1536.51 excludes all absent analytic coordinates, finite threshold exhaustion from Definition 1536.52 accounts for every displayed lower-bound threshold, and the $\mathsf{LocatedSet}^{\uparrow}$ threshold route of Definition 1536.54 places located-set threshold consumption downstream of the same rows. Thus the listed results form the paper-side obligation surface and do not export semantic liminf data, quotient equality, selected sequences, choice, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphClosurePackage`

Theorem 1536.54 (LowerSemicontinuous root LocatedSet threshold route)**.** *Every* $\mathsf{LocatedSet}^{\uparrow}$ *consumer of a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph threshold must read the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *value row* R*, the lower-real epigraph ledger* E*, the comparison row* O*, the* $\mathsf{LocatedReal}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, the* hsame *transport row* H*, the displayed* Cont *replay row* C*, the* Pkg *provenance row* P*, and the local* NameCert *row* N *before it consumes the located-set packet. Thus the threshold route is the finite route*

$$W, R, E, O, H, C, P, N$$

with the located-real and Real-seal boundary rows displayed at the comparison endpoint, not a host topology, semantic liminf operator, quotient $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Proof. First build the root epigraph basis. Apply Definition 1536.45 to expose the scheduled source window, regular rational value row, lower-real epigraph ledger, comparison row, transport, replay, provenance, and local name.

Next read the located endpoint. The located readback theorem Definition 1536.46 keeps every lower-bound read attached to the same W, R, E, O, H, C, P, N route, and the exactness theorem Definition 1536.50 identifies exactly when that route becomes consumable as a $\mathsf{LocatedSet}^{\uparrow}$ packet.

Finally apply finite threshold exhaustion from Definition 1536.52. It says that every displayed lower-bound threshold visible to lower-real, upper-real, located-real, or Real-facing consumers has already appeared in the same finite rows. Hence the located-set consumer must read the listed rows before consumption, and the excluded host analytic objects have no route into the packet. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedSetThresholdRoute`

Theorem 1536.55 (LowerSemicontinuous root epigraph specialization)**.** *Let an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet expose a concrete lower-bound threshold read. Then that read specializes the root route*

$$W, R, E, O, H, C, P, N$$

to the scheduled $\mathsf{StreamName}^{\uparrow}$ *window, the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *value row, the lower-real epigraph ledger, the comparison endpoint, transport, replay, provenance, and local naming rows already displayed by the packet. The specialization exports no semantic liminf predicate, selected infimum, host topology, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, open-cover compactness proof, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphSpecialization`

Proof. Start with the root epigraph carrier and the regular readback handoff. They place the scheduled window W, value row R, epigraph ledger E, comparison row O, and structural rows H, C, P, N before any concrete threshold read reaches a consumer.

Apply finite threshold exhaustion. Every lower-bound threshold visible to lower-real, upper-real, located-real, located-set, or Real-facing consumers has already appeared in the same scheduled rows, so specializing to one concrete threshold is just projecting that finite ledger.

Finally use the Real seal boundary and the root LocatedSet threshold route. The terminal seal and located-set consumer both remain downstream of W, R, E, O, H, C, P, N, and the excluded analytic objects have no carrier coordinate from which to enter the specialized read. □

Theorem 1536.56 (LowerSemicontinuous root LowerReal projection)**.** *Before the terminal* $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet is consumed, the comparison row* O *projects the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *boundary reads through the displayed* W, R, E *threshold ledger. Thus a lower-real projection reaches the located endpoint only after the scheduled* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have been displayed, and not by host real order, trichotomy, selected infimum, quotient equality, choice, compactness, or ambient completeness.*

Proof. Project the comparison row from the root epigraph carrier. The row O is downstream of the scheduled window, regular rational readback, and epigraph ledger, so its lower-real and upper-real reads are not independent endpoints.

Use the regular readback handoff and Real seal boundary. They force every lower or upper boundary value to pass through W, R, E, O before the $\mathsf{LocatedReal}^{\uparrow}$ handoff or terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Finite threshold exhaustion supplies the nonescape clause. Since every visible threshold is already in the finite scheduled rows, the projection cannot be replaced by host order, trichotomy, selected infimum, quotient equality, choice, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLowerRealProjection`

Theorem 1536.57 (LowerSemicontinuous root LocatedSet nonescape)**.** *No* $\mathsf{LocatedSet}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet can bypass the finite scheduled* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. Any such consumer must read the displayed route*

$$W, R, E, O, H, C, P, N$$

including the lower-real epigraph ledger, located comparison endpoint, terminal Real-seal boundary, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives no host topology, semantic liminf operator, selected approximating sequence, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedSetNonescape`

Proof. Begin with the epigraph-to-$\mathsf{LocatedSet}^{\uparrow}$ handoff and exactness theorems. They identify located-set consumption with the scheduled window, regular rational value row, epigraph ledger, comparison endpoint, transport, replay, provenance, and local name.

Apply the root LocatedSet threshold route. It requires the located-set consumer to pass through the same W, R, E, O, H, C, P, N rows before the located-set packet is consumed.

Finally use finite threshold exhaustion together with the Real seal boundary. The finite scheduled rows already contain every visible lower-bound threshold, and the terminal seal is downstream of those rows. Hence a located-set consumer has no route around the finite schedule and no coordinate for the excluded host analytic principles. □

Theorem 1536.58 (LowerSemicontinuous epigraph threshold monotonicity)**.** *Let two displayed* $\mathsf{Real}^{\uparrow}$ *threshold rows of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet be ordered by the dyadic tolerance ledger before the terminal seal is consumed. The later epigraph read refines the earlier read through the same finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{LocatedReal}^{\uparrow}, \mathsf{LowerReal}^{\uparrow}, \mathsf{UpperReal}^{\uparrow}, \mathsf{LocatedSet}^{\uparrow}$$

and through the same W, R, E, O, H, C, P, N *rows. The refinement is therefore a located-set routing fact internal to the displayed packet, not a host real-order comparison, trichotomy principle, selected supremum, countable-choice schedule, or ambient completeness argument.*

Proof. Start with finite threshold exhaustion from Definition 1536.52. It places every visible lower-bound threshold in the scheduled $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows before the epigraph ledger and comparison endpoint are read.

Next use the root epigraph basis and the regular readback handoff. The basis theorem Definition 1536.45 keeps both threshold rows in the same W, R, E, O, H, C, P, N carrier, while Definition 1536.40 attaches each read to the corresponding located endpoint before any $\mathsf{Real}^{\uparrow}$ seal is exported.

Finally apply the LocatedSet threshold route of Definition 1536.54. Since the dyadic tolerance ledger orders the displayed threshold rows before the located-set consumer reads them, the later row can only refine the earlier one along that finite route.

The exclusions follow by route exhaustion. No coordinate of the carrier supplies host real order, trichotomy, selected suprema, countable choice, or ambient completeness, so monotonicity is witnessed only by the displayed epigraph and located-set rows. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphThresholdMonotonicity`

Theorem 1536.59 (LowerSemicontinuous root epigraph carrier scope)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet reads its located-set predicate,* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *cut rows, regular rational stream witness, and terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the root epigraph carrier. The packet introduces no open-cover compactness proof, selected infimum, semantic liminf operator, host real order, quotient real equality, or countable-choice schedule.*

Proof. Start with the root epigraph carrier of Definition 1536.39. Its rows expose the source, graph, scheduled finite window, regular rational readback, lower-real epigraph ledger, comparison row, transport, replay, provenance, and local naming data.

The located-set predicate is reached through the handoff of Definition 1536.49. The lower and upper cuts are read through the regular readback handoff of Definition 1536.40 and the root epigraph basis of Definition 1536.45.

The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of those same rows by Definition 1536.41. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row keep the located-set and real-cut reads attached to the accepted BHist packet.

The exclusions are coordinate exhaustion. Since the carrier contains no open-cover compactness proof, selected infimum, host real order, semantic liminf operator, quotient real equality, or countable-choice schedule, the scoped root epigraph read cannot export any of them. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphCarrierScope`

Theorem 1536.60 (LowerSemicontinuous root epigraph coverage)**.** *Every accepted lower-bound threshold of a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet is covered by the displayed route*

$$W, R, E$$

before any $\mathsf{LocatedReal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer reads it. The finite window row* W*, regular rational readback* R*, and epigraph ledger* E *therefore exhaust threshold coverage before the comparison, transport, replay, provenance, and local NameCert rows are consumed.*

Proof. Start with the root epigraph carrier of Definition 1536.39. It displays the scheduled finite window, regular rational value row, and lower-real epigraph ledger before the comparison endpoint.

Apply the regular readback handoff of Definition 1536.40. A threshold can reach a located or Real-facing consumer only after the W, R, E route has already attached the scheduled value to its epigraph witness.

Finally use the root epigraph carrier scope of Definition 1536.59. The remaining rows O, H, C, P, N preserve and name the covered threshold; they do not create a hidden threshold outside the displayed finite window. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphCoverage`

Theorem 1536.61 (LowerSemicontinuous root epigraph coverage exactness)**.** *An accepted lower-bound threshold of a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet is covered exactly when the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *value row* R*, epigraph ledger* E*, located comparison*

endpoint O, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$ *row N are all displayed in one route. The exactness boundary exports no semantic liminf predicate, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected infimum, countable choice, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphCoverageExactness`

Proof. For the forward direction, start with the root epigraph coverage theorem. It places every accepted lower-bound threshold in the scheduled window, regular rational value row, and epigraph ledger before any located or Real-facing consumer reads the threshold.

Apply finite epigraph threshold exhaustion. The theorem forces any displayed threshold visible to lower-real, upper-real, located-real, or Real-facing consumers to pass through the same W, R, E, O, H, C, P, N route.

Use the LocatedSet threshold route to identify the comparison endpoint. The located-set consumer can read the threshold only after O and the structural rows H, C, P, N are displayed with the same scheduled value.

For the reverse direction, project an accepted route with all listed rows. The scheduled window and regular value row provide the finite threshold read, the epigraph ledger covers it, and the located comparison endpoint keeps the lower-bound threshold attached to the same packet.

Finally apply the root epigraph carrier scope. Since semantic liminf, quotient equality, selected infimum, countable choice, and ambient completeness have no carrier coordinate, the exactness statement is only the displayed finite coverage boundary. □

Theorem 1536.62 (LowerSemicontinuous root located comparison)**.** *The comparison row O of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet compares* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, and* $\mathsf{LocatedReal}^{\uparrow}$ *endpoints only after the epigraph ledger E has been displayed from the scheduled W, R readback. Thus located comparison is downstream of finite epigraph evidence, not a host order comparison, trichotomy principle, selected infimum, or quotient real equality.*

Proof. Project the rows of Definition 1536.39. The row order places W and R before E, and places O after the displayed epigraph ledger.

Use Definition 1536.40. It keeps every $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the same lower-bound witness before O may compare the lower, upper, and located endpoints.

The scope theorem Definition 1536.59 supplies the boundary. Since host order, trichotomy, selected infimum, and quotient equality have no coordinate in the carrier, the comparison row can only read the displayed epigraph ledger. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedComparison`

Theorem 1536.63 (LowerSemicontinuous root Real seal refusal)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet exports no semantic liminf operator, selected infimum, host topology, quotient equality, countable-choice schedule, or ambient completeness theorem. It seals only the displayed W, R, E, O, H, C, P, N route supplied by the finite epigraph carrier.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootRealSealRefusal`

Proof. Begin with the Real seal boundary of Definition 1536.41. It places the terminal seal after the finite epigraph window, comparison row, transport, replay, provenance, and local naming rows.

Apply root epigraph coverage and located comparison above. Coverage accounts for every accepted lower-bound threshold through W, R, E, while the comparison row O reads lower, upper, and located endpoints only after that ledger is displayed.

Finally use Definition 1536.59. The refused liminf, selected infimum, topology, quotient equality, choice, and completeness objects have no carrier coordinate, so the terminal seal cannot export them. □

Theorem 1536.64 (LowerSemicontinuous root epigraph finite infimum handoff)**.** *Any finite lower-bound infimum read consumed by a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet must pass through the scheduled* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *value row, the lower-real epigraph ledger, the located comparison row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise transport, displayed replay, provenance, and local* NameCert *rows. The packet exports no selected infimum, semantic liminf operator, host topology, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, open-cover compactness proof, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphFiniteInfimumHandoff`

Proof. Start with the root epigraph carrier of Definition 1536.39. Its finite route displays the scheduled window, regular rational value row, lower-real epigraph ledger, comparison row, transport, replay, provenance, and local naming before any real-facing endpoint is consumed.

Apply finite threshold exhaustion through Definition 1536.52. Every lower-bound threshold read is accounted for inside the scheduled $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ value row, so an infimum-facing finite read cannot skip the epigraph ledger.

Use located comparison from Definition 1536.62. The comparison row reads lower, upper, and located endpoints only after the epigraph ledger has been displayed, so the lower-bound infimum route remains a finite BEDC comparison route rather than a host-order selection.

Finally apply the Real seal refusal of Definition 1536.63. The terminal seal refuses semantic liminf data, selected infima, host topology, quotient equality, choice schedules, compactness proofs, and ambient completeness. Hence the finite infimum handoff exports only the displayed W, R, E, O, H, C, P, N route. □

Theorem 1536.65 (LowerSemicontinuous epigraph threshold refinement)**.** *Let two displayed* $\mathsf{Real}^{\uparrow}$ *threshold rows of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph packet be ordered before the terminal seal is consumed. The refined row reaches every* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LocatedSet}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumer through the same*

$$W, R, E, O, H, C, P, N$$

route as the coarser row, with the refinement witnessed before the terminal seal and not by a host real-order comparison, selected infimum, semantic liminf operator, countable-choice schedule, open-cover compactness proof, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphThresholdRefinement`

Proof. Start with finite epigraph threshold exhaustion. It places both displayed threshold rows in the scheduled $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows before the epigraph ledger is consumed.

Use the LocatedSet threshold route and epigraph threshold monotonicity. The located-set consumer reads the same W, R, E, O, H, C, P, N rows, and the ordered threshold ledger lets the refined row sharpen the coarser read only along that finite route.

Finally apply the root epigraph carrier scope. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of the displayed transport, replay, provenance, and local naming rows, so the refinement cannot import host order, selected infima, semantic liminf data, choice, compactness, or ambient completeness. □

Theorem 1536.66 (LowerSemicontinuous located epigraph filter stability)**.** *Every located-set filter consumer of an accepted* LowerSemicontinuous$^{\uparrow}$ *root epigraph packet reads the same finite epigraph threshold rows under componentwise* hsame *transport and displayed* Cont *replay. Thus a filter-stable located epigraph read is carried by*

$$W, R, E, O, H, C, P, N$$

and exports no detached topology, quotient Real$^{\uparrow}$ *equality, selected approximating sequence, countable-choice schedule, open-cover compactness proof, or ambient Real-completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedEpigraphFilterStability`

Proof. Begin with the epigraph-to-LocatedSet$^{\uparrow}$ exactness theorem. It identifies located-set consumption with the scheduled window, regular rational value row, epigraph ledger, comparison endpoint, transport, replay, provenance, and local name.

Apply finite threshold exhaustion and threshold refinement. Every lower-bound threshold visible to the located filter is already displayed in the same finite rows, and ordered threshold reads refine before the terminal seal.

Transport stability is then carrier-local. The hsame row preserves the displayed threshold rows componentwise, and the Cont row replays the same route for the filter consumer. The carrier scope theorem excludes detached topology, quotient equality, selection, choice, compactness, and ambient completeness from the endpoint. □

Theorem 1536.67 (LowerSemicontinuous epigraph filter route)**.** *Let an accepted* LowerSemicontinuous$^{\uparrow}$ *root epigraph packet expose threshold rows that are stable under located comparison. Then every filter-facing read of its epigraph factors through* LocatedSet$^{\uparrow}$ *rows,* LowerReal$^{\uparrow}$ *and* UpperReal$^{\uparrow}$ *boundary rows,* RegSeqRat$^{\uparrow}$ *readback,* StreamName$^{\uparrow}$ *windows, and the terminal* Real$^{\uparrow}$ *seal along the same*

$$W, R, E, O, H, C, P, N$$

route. The exported route contains no selected minimizer, quotient function equality, detached topology, selected approximating sequence, countable-choice schedule, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphFilterRoute`

Proof. Start with the root epigraph carrier. The rows W, R, E, O, H, C, P, N place the stream window, regular rational readback, epigraph ledger, located comparison endpoint, transport, replay, provenance, and local name before any filter consumer may inspect the packet.

Use the located readback and epigraph-to-LocatedSet$^{\uparrow}$ handoff. These rows identify the filter-facing epigraph as a located-set read whose lower and upper real boundary data are still tied to the same displayed threshold window.

Apply finite threshold exhaustion and threshold refinement. Every threshold requested by the filter is already present in the finite ledger, and each ordered located comparison refines through the displayed lower, upper, and located endpoint rows.

Finally use located epigraph filter stability. Componentwise hsame transport and displayed Cont replay preserve the same route for the filter consumer, while the root carrier scope excludes minimizer selection, quotient function equality, detached topology, selected sequences, choice schedules, and ambient completeness. □

Theorem 1536.68 (LowerSemicontinuous root epigraph filter directedness)**.** *Let an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet expose two epigraph threshold/filter rows over the same* $\mathsf{Real}^{\uparrow}$ *seal and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback source. Then the displayed located comparison, epigraph threshold refinement, and epigraph filter route admit a common refined threshold row carried by the same finite route*

$$W, R, E, O, H, C, P, N.$$

The common row is a BHist *and* hsame*-stable filter refinement whose* Cont *replay and* NameCert *provenance remain internal to the accepted packet. It exports no host liminf theorem, open-sublevel theorem, detached topology, selected approximating sequence, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphFilterDirectedness`

Proof. Start with the two displayed threshold/filter rows. Since they share the same $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{RegSeqRat}^{\uparrow}$ readback source, both rows are already visible through the scheduled window W, regular value row R, epigraph ledger E, and located comparison endpoint O.

Apply the root located comparison theorem and the epigraph threshold refinement theorem. Located comparison keeps both lower-bound reads attached to the same W, R, E, O route, while threshold refinement supplies a row below the two displayed reads before the terminal seal is consumed.

Now use the epigraph filter route. The filter-facing read of each threshold factors through the same located-set, lower-real, upper-real, regular-readback, stream-window, and Real-seal rows, so the refined threshold is a common filter row rather than a second host filter object.

Finally repack through the structural rows. Componentwise hsame transport keeps the common row in the same BHist packet, Cont replay preserves the route for consumers, and P, N supply provenance and local naming. The excluded host liminf, open-sublevel, topology, selector, quotient, choice, and completeness objects have no coordinate in this directedness witness. □

Theorem 1536.69 (LowerSemicontinuous root epigraph filter basis closure)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph filter basis over the same* $\mathsf{Real}^{\uparrow}$ *seal and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback source is closed under displayed threshold refinement and common filter rows along*

$$W, R, E, O, H, C, P, N.$$

The closure read is a BHist*-routed basis package: lower thresholds refine through the epigraph ledger, common rows remain under the same located comparison endpoint, and componentwise* hsame *transport plus displayed* Cont *replay keep provenance and local naming internal to the accepted packet. It exports no host liminf predicate, detached topology, selected approximating sequence, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable choice schedule, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphFilterBasisClosure`

Proof. Begin with finite threshold exhaustion. Definition 1536.52 places every displayed lower-bound request inside the scheduled $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, epigraph ledger, comparison endpoint, transport, replay, provenance, and local naming route.

Next apply epigraph threshold refinement and located epigraph filter stability. Definition 1536.65 supplies refined threshold rows before the terminal seal is consumed, while Definition 1536.66 keeps those rows stable under componentwise transport and replay.

Now use the epigraph filter route and Real threshold boundary. Definition 1536.67 forces every filter-facing read through the same located-set, lower-real, upper-real, regular-readback, stream-window, and Real-seal rows, and Definition 1536.72 keeps any Real-facing threshold downstream of the displayed finite route.

Finally repack with directedness. Definition 1536.68 supplies the common refined filter row over the shared $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{RegSeqRat}^{\uparrow}$ source; the structural rows H, C, P, N name that row as a basis closure witness. Since the carrier has no coordinate for host liminf, detached topology, selectors, quotient equality, choice, or ambient completeness, none of those objects is exported by the closure package. □

Theorem 1536.70 (LowerSemicontinuous root epigraph filter-basis exactness)**.** *An accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph filter-basis read is exact precisely when the displayed route*

$$W, R, E, O, H, C, P, N$$

is present and every common refinement is obtained from epigraph threshold refinement followed by root epigraph filter directedness. Thus the exact basis has no hidden host topology, selected infimum, semantic liminf predicate, quotient $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphFilterBasisExactness`

Proof. For the forward read, apply Definition 1536.69. It keeps the basis inside the displayed scheduled-window, regular-readback, epigraph-ledger, located-comparison, transport, replay, provenance, and local naming route.

The refinement witness is forced by the finite threshold route. Use Definition 1536.65 to refine displayed lower-bound thresholds before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed, and then apply Definition 1536.68 to obtain the common filter row over the same $\mathsf{RegSeqRat}^{\uparrow}$ source and $\mathsf{Real}^{\uparrow}$ seal.

Conversely, suppose the route W, R, E, O, H, C, P, N is present and the common refinement is supplied by threshold refinement plus filter directedness. The Real-facing threshold is already downstream of the finite packet by Definition 1536.72, so no outside topology, infimum selector, liminf predicate, quotient equality, choice schedule, or completeness principle can enter the basis read. This is exactly the finite epigraph basis advertised by the accepted root packet. □

Theorem 1536.71 (LowerSemicontinuous root filter-basis directed refinement)**.** *Let two displayed* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph filter-basis rows lie over the same* W, R, E, O, H, C, P, N *route. They admit a common displayed refinement only through epigraph threshold refinement, located epigraph filter stability, and root epigraph filter directedness. The refinement exports no host liminf theorem, open-sublevel theorem, detached topology, selected approximating sequence, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootFilterBasisDirectedRefinement`

Proof. Begin with the filter-basis closure package. Definition 1536.69 keeps both basis rows inside the same scheduled window, regular readback, epigraph ledger, located comparison, transport, replay, provenance, and local naming route.

Use epigraph threshold refinement next. Definition 1536.65 supplies the only permitted way to refine the displayed lower-bound thresholds before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally apply filter directedness. Definition 1536.68 provides the common refined filter row over the shared $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{RegSeqRat}^{\uparrow}$ source, while located epigraph filter stability keeps componentwise transport and replay inside the accepted packet. Since none of the refused host objects appears as a carrier coordinate, the directed refinement is exactly the displayed finite witness. □

Theorem 1536.72 (LowerSemicontinuous root filter real threshold)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph filter read that reaches a* $\mathsf{Real}^{\uparrow}$*-facing threshold must first expose the* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LocatedSet}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows already named by the root carrier. In carrier notation the threshold route is the finite read*

$$W, R, E, O, H, C, P, N$$

with the scheduled window W, regular rational value row R, epigraph ledger E, located comparison endpoint O, transport row H, displayed replay row C, provenance row P, and local naming row N visible before the Real-facing threshold is public. The route exports no semantic liminf operator, quotient $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable choice schedule, open-cover compactness proof, or ambient Real-completeness theorem.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootFilterRealThreshold`

Proof. Start with the epigraph filter route above. It forces every filter-facing read to pass through the located-set handoff, the lower and upper real boundary rows, the scheduled stream window, the regular rational readback, and the terminal Real seal on the same displayed route.

Use located comparison and located readback. The comparison row O is the only endpoint at which $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$ data meet, and it is downstream of the finite W, R, E threshold ledger rather than a host order predicate.

Finally apply the Real seal refusal block. Since the terminal $\mathsf{Real}^{\uparrow}$ seal can only seal the already displayed W, R, E, O, H, C, P, N route, a Real-facing threshold cannot be read before those rows are exposed. The refused liminf, quotient-equality, selection, choice, compactness, and completeness objects have no carrier coordinate from which to enter the threshold read. □

Theorem 1536.73 (LowerSemicontinuous root epigraph threshold continuity)**.** *Every accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root epigraph filter read preserves the same threshold row when transported through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, located-comparison,* $\mathsf{LocatedSet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. In carrier notation, if the threshold is read along*

$$W, R, E, O, H, C, P, N,$$

then every transported filter-facing threshold read of the same accepted packet reads that same row through the same finite route. The transport exports no host liminf predicate, quotient $\mathsf{Real}^{\uparrow}$ *equality, selected approximating sequence, countable-choice schedule, open-cover compactness proof, or ambient completeness theorem.*

Proof. Begin with the root epigraph carrier and the regular readback handoff. The carrier fixes W, R, E, O, H, C, P, N as the scheduled window, regular rational value row, epigraph ledger, located comparison endpoint, transport, replay, provenance, and local naming route, while the readback handoff keeps the threshold downstream of the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows.

Use located comparison next. The $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ rows meet only at the located comparison endpoint O, and the $\mathsf{LocatedSet}^{\uparrow}$ read is attached to that endpoint before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed.

Apply filter directedness and filter-basis exactness. Directedness supplies common displayed threshold rows for filter-facing reads over the same source, and exactness says those rows are obtained by epigraph threshold refinement rather than by an external topological or liminf object.

Finally use Definition 1536.72. It places every Real-facing threshold downstream of the same W, R, E, O, H, C, P, N route, so hsame transport and Cont replay preserve the displayed row without introducing host liminf, quotient equality, selectors, countable choice, open-cover compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphThresholdContinuity`

Theorem 1536.74 (LowerSemicontinuous public epigraph export route)**.** *Every public* LowerSemicontinuous$^{\uparrow}$ *export that exposes an epigraph read factors through the displayed root epigraph carrier, located threshold route,* LowerReal$^{\uparrow}$ *projection,* RegSeqRat$^{\uparrow}$ *readback, and terminal* Real$^{\uparrow}$ *seal rows. In carrier notation the public route is*

$$X, F, W, R, E, O, H, C, P, N,$$

with the scheduled window W and value row R read before the epigraph ledger E, the lower-real and upper-real boundary projection read at O, and the Real seal consumed only after transport, replay, provenance, and local naming are visible. The export provides a public epigraph surface for LowerReal, LocatedSet, RegSeqRat, and Real consumers; it exports no host topology, semantic liminf operator, selected infimum, quotient Real$^{\uparrow}$ *equality, countable-choice schedule, open-cover compactness proof, or ambient completeness theorem.*

Proof. Start with the root epigraph threshold continuity theorem Definition 1536.73. It keeps every transported filter-facing threshold on the same finite W, R, E, O, H, C, P, N route and prevents a public read from detaching into an external threshold object.

Next apply the LowerReal projection block Definition 1536.56. The lower-real and upper-real rows meet only through the displayed comparison endpoint O, so the public epigraph export cannot bypass the scheduled StreamName$^{\uparrow}$ window or RegSeqRat$^{\uparrow}$ value row.

Finally use the root epigraph carrier, Real seal boundary, and located-set handoff already assembled in this file. They place X, F before the threshold subroute and place the terminal Real$^{\uparrow}$ seal after E and O. Thus the public export is exactly the displayed carrier route and has no coordinate for the refused host analytic objects. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousPublicEpigraphExportRoute`

Theorem 1536.75 (LowerSemicontinuous formal-target bridge request)**.** *Every formalization target for the public* LowerSemicontinuous$^{\uparrow}$ *epigraph export must factor through the root epigraph carrier, located threshold exhaustion, lower-real projection, located-set nonescape, finite-infimum handoff, epigraph filter route, and real-threshold theorem already displayed in this file. In carrier notation the requested bridge target is confined to*

$$X, F, W, R, E, O, H, C, P, N,$$

with W the StreamName$^{\uparrow}$ *inspection window, R the* RegSeqRat$^{\uparrow}$ *readback row, E the lower-real epigraph ledger, O the located comparison endpoint, and H, C, P, N the structural transport, replay, provenance, and local naming rows. The requested target refuses host topology, quotient* Real$^{\uparrow}$ *equality, semantic limits, and ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousFormalTargetBridgeRequest`

Proof. Start with the root epigraph carrier and located threshold exhaustion, Definition 1536.39 and Definition 1536.52. These make the scheduled source, stream window, regular rational readback, epigraph ledger, comparison endpoint, and structural rows explicit before any public epigraph read is exported.

Apply the lower-real projection and located-set nonescape theorems, Definition 1536.56 and Definition 1536.57. They force the lower-real, upper-real, located endpoint, and located-set reads to remain on the same finite route rather than entering through host topology or quotient real equality.

Finally use the finite-infimum handoff, epigraph filter route, and real-threshold theorem, Definition 1536.64, Definition 1536.67, and Definition 1536.72. Their rows place the filter-facing and Real-facing threshold reads downstream of W, R, E, O, H, C, P, N, so a formal target for the public export can request exactly that bridge surface and no semantic limit or ambient completeness principle. □

Theorem 1536.76 (LowerSemicontinuous lower-real epigraph classifier)**.** *For the root packet $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$, every* $\mathsf{LowerReal}^{\uparrow}$ *epigraph classifier read is determined by the displayed source row X, graph row F, finite stream schedule W, regular rational value row R, and epigraph ledger E. The classifier may be transported by H and replayed by C, but it cannot be replaced by a host liminf predicate, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the root carrier from Definition 1536.22. The only lower-real coordinate available to a consumer is the epigraph ledger E attached to the source, graph, schedule, and readback rows.

Use Definition 1536.16. It orders the classifier read through X, F, W, R, E before the located comparison, transport, replay, provenance, and local naming rows are consumed.

Finally apply Definition 1536.18. Since the excluded host liminf, quotient equality, choice, and completeness principles have no carrier coordinate and no replay row, the epigraph classifier remains finite $\mathsf{LowerReal}^{\uparrow}$ data. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLowerRealEpigraphClassifier`

Theorem 1536.77 (LowerSemicontinuous regular readback transport)**.** *Every transported lower-semicontinuous readback of $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ carries the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row W and* $\mathsf{RegSeqRat}^{\uparrow}$ *value row R into the located comparison ledger O. The transport exports a finite readback route through W, R, E, O, H, C, P, N and not a semantic liminf operation, selected representative, quotient real equality, countable choice, or ambient Real-completeness witness.*

Proof. Start with the transported readback. The root carrier gives H as the only componentwise transport row and C as the only displayed replay row, so the readback cannot leave the carrier surface.

Apply Definition 1536.17. It keeps the transported epigraph comparison on the same scheduled window W and regular rational readback R before the located route O is available.

Combine this with Definition 1536.23. The Real handoff reads the transported route only after W, R, E, O, H, C, P, N are visible, so the consumer receives no host liminf, selector, quotient equality, choice, or ambient completeness data. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRegularReadback transport`

Theorem 1536.78 (LowerSemicontinuous root epigraph located transport)**.** *Every located comparison read from $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ factors through the same finite schedule W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, epigraph ledger E, comparison row O,* hsame *transport row H,* Cont *replay row C, provenance row P, and local NameCert row N. The transported route may compare* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *evidence, but it cannot replace the displayed epigraph data by semantic liminf data, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the lower-real epigraph classifier of Definition 1536.76. It pins the classifier to the displayed source, graph, finite stream schedule, regular rational readback, and epigraph ledger rows.

Next apply Definition 1536.77. The transport row keeps the located comparison on the same W, R, E, O, H, C, P, N surface, so a $\mathsf{LowerReal}^{\uparrow}$ or $\mathsf{UpperReal}^{\uparrow}$ comparison has no independent readback source.

Finally use Definition 1536.18. The Real-facing seal excludes semantic liminf data, quotient equality, selected representatives, choice, and ambient completeness. Hence the located transport factors through the root epigraph route already displayed. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphLocatedTransport`

Theorem 1536.79 (LowerSemicontinuous root epigraph carrier admission)**.** *Every accepted root packet $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ must expose the source row X, graph row F, epigraph ledger E, finite stream schedule W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, located comparison row O,* hsame *transport row H,* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}$ *row N before any* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer reads lower-semicontinuity data. No accepted root packet may supply that read through semantic liminf data, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphCarrierAdmission`

Proof. Project the root carrier of Definition 1536.22. The displayed coordinates of L_{root} are exactly $X, F, E, W, R, O, H, C, P, N$, so any consumer-side read must first select one of those rows.

Apply Definition 1536.76 and Definition 1536.78. The classifier pins the lower-real epigraph data to X, F, E, W, R, while the located transport keeps the comparison on O, H, C, P, N before a $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer can inspect it.

Finally use coordinate exhaustion against Definition 1536.18. The carrier has no row for semantic liminf data, quotient equality, countable choice, or ambient Real completeness, so none of those readings can serve as the root epigraph admission route. □

Theorem 1536.80 (LowerSemicontinuous root epigraph RegSeqRat handoff)**.** *Every Real-facing read from the epigraph ledger of $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ must first pass through the* $\mathsf{StreamName}^{\uparrow}$ *schedule row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *value row R, the* $\mathsf{LowerReal}^{\uparrow}$ *epigraph row E, the located comparison row O, and the structural rows H, C, P, N. The handoff therefore exports a finite scheduled readback of the epigraph route, not an unscheduled Real value or a host liminf operation.*

Proof. Start with Definition 1536.76. It fixes the epigraph classifier on the displayed rows X, F, W, R, E before any located comparison is considered.

Use Definition 1536.77. The transported readback keeps the same StreamName schedule and RegSeqRat row visible when the located row O is consumed.

Now apply Definition 1536.78 and Definition 1536.82. These boundary steps force the Real-facing read through W, R, E, O, H, C, P, N, so the export is a scheduled epigraph handoff rather than a semantic Real operation. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootEpigraphRegSeqRatHandoff`

Theorem 1536.81 (LowerSemicontinuous located Real non-escape)**.** *The located comparison row O of $L_{\text{root}} = (X, F, E, W, R, O, H, C, P, N)$ cannot export semantic liminf data, quotient $\mathsf{Real}^{\uparrow}$ equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Any Real-facing located comparison remains a read of the displayed epigraph ledger E, schedule W, regular rational row R, transport H, replay C, provenance P, and local NameCert row N.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRootLocatedRealNonescape`

Proof. Read the comparison through Definition 1536.80. The only available Real-facing handoff has already exposed W, R, E, O, H, C, P, N.

The classifier and transport rows are controlled by Definition 1536.76 and Definition 1536.77. They keep the located comparison attached to the displayed lower-real epigraph route.

Finally use Definition 1536.78 with Definition 1536.82. Since the carrier has no coordinates for semantic liminf, quotient equality, selected representatives, choice, or ambient completeness, none of those data can escape through the located Real comparison. □

Theorem 1536.82 (LowerSemicontinuous Real-seal export boundary)**.** *Every $\mathsf{Real}^{\uparrow}$ export of $L_{\text{root}} = (X, F, E, W, R, O, H, C, P, N)$ is bounded by the displayed rows E, W, R, O, H, C, P, N. The export may name lower-real epigraph evidence, located comparison data, regular rational readback, componentwise transport, replay, provenance, and the local NameCert row, but it does not export host topology, semantic liminf data, quotient real equality, countable choice, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.*

Proof. Read the Real-facing consumer through Definition 1536.23. The handoff orders the export from the stream schedule W to the regular rational row R, then through the epigraph ledger E and located comparison ledger O.

Use Definition 1536.76 and Definition 1536.77. The first keeps the lower-real classifier on the displayed epigraph route, and the second keeps transported readbacks on the same W, R, E, O, H, C surface.

Finally apply Definition 1536.18. It excludes host topology, semantic liminf data, quotient real equality, countable choice, selected limits, compactness, and ambient completeness from the terminal Real seal, so the export boundary is exactly the finite route stated above. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealExportBoundary`

Theorem 1536.83 (LowerSemicontinuous obligation closure route)**.** *For the root packet $L_{\text{root}} = (X, F, E, W, R, O, H, C, P, N)$, the carrier of Definition 1536.22, the NameCert obligation surface of Definition 1536.5, epigraph-window exactness from Definition 1536.6, located epigraph transport from Definition 1536.78, regular readback nonescape from Definition 1536.77, the Real seal boundary of Definition 1536.82, and the open-sublevel route of Definition 1536.24 jointly form the obligation-closure package for the displayed lower-semicontinuity certificate. Every consumer route is forced through $X, F, E, W, R, O, H, C, P, N$ before it may read a lower-real epigraph, located comparison, open-sublevel, regular rational readback, or terminal $\mathsf{Real}^{\uparrow}$ seal.*

Proof. Start with the root carrier and NameCert obligation surface. The carrier of Definition 1536.22 exposes exactly the source, graph, epigraph, stream schedule, regular readback, located comparison, transport, replay, provenance, and local naming rows, while Definition 1536.5 binds those rows as the local certificate surface.

Use epigraph-window exactness and located transport. The theorem Definition 1536.6 keeps every epigraph read attached to the displayed W, R, E window, and Definition 1536.78 carries that same route through O, H, C, P, N without importing semantic liminf data, quotient equality, selected representatives, countable choice, or ambient completeness.

Finally combine regular readback transport, the Real seal export boundary, and the open-sublevel route. The readback theorem keeps $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows visible, the seal theorem forces the terminal $\mathsf{Real}^{\uparrow}$ endpoint through E, W, R, O, H, C, P, N, and Definition 1536.24 identifies open-sublevel consumption with the same finite epigraph route. These clauses exhaust the obligation package and leave no hidden host-topological or completeness coordinate. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousObligationClosureRoute`

Theorem 1536.84 (LowerSemicontinuous epigraph-window monotone refinement)**.** *Let* $L_{\mathrm{root}} = (X, F, E, W, R, O, H, C, P, N)$ *be an accepted root packet. Refining the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row* R *preserves the lower-real epigraph classifier and the located comparison route. The refined read remains a consumer of the same displayed rows* W, R, E, O, H, C, P, N *and cannot be reinterpreted as semantic liminf data, quotient* $\mathsf{Real}^{\uparrow}$ *equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 1536.76. The classifier is already fixed by the source row X, graph row F, stream window W, regular rational row R, and epigraph ledger E, so a finer finite window only changes which displayed rows of the same schedule and tolerance surface are read.

Use Definition 1536.77. Transport of the refined readback keeps the same $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ coordinates visible before the located comparison row O is consumed.

Finally apply Definition 1536.80. The Real-facing handoff still runs through W, R, E, O, H, C, P, N, so the refinement has no coordinate for semantic liminf, quotient equality, selected representatives, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphWindowMonotoneRefinement`

Theorem 1536.85 (LowerSemicontinuous obligation closure package)**.** *The* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *root packet carries the paper-side obligation package through the finite route*

$$X, F, W, R, E, O, H, C, P, N.$$

Carrier admission, epigraph-window exactness, lower-real classifier stability, located readback compatibility, transport replay, provenance, and Real-seal nonescape are all read from the displayed rows of Definition 1536.1, Definition 1536.5, Definition 1536.16, and the root epigraph obligations of Definition 1536.53. The package does not add host topology, semantic liminf data, quotient real equality, selected approximating sequences, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousObligationClosurePackage`

Proof. Project the carrier and NameCert obligation surface. They list the source, graph, epigraph, schedule, readback, comparison, transport, replay, provenance, and local naming rows as the complete obligation inventory.

Use the lower-real epigraph route and the root epigraph obligation package. These results place every lower-bound read on $X, F, W, R, E, O, H, C, P, N$ and account for exactness, located comparison, finite thresholds, and root nonescape within that route.

The remaining rows are structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same finite packet, so no refused analytic principle is introduced by packaging the obligations together. □

Theorem 1536.86 (LowerSemicontinuous public epigraph export)**.** *Every public epigraph consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must read the finite route*

$$X, F, W, R, E, O, H, C, P, N.$$

The window row W is read through $\mathsf{StreamName}^{\uparrow}$*, the value row R through* $\mathsf{RegSeqRat}^{\uparrow}$*, the epigraph and comparison rows E, O through* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, and* $\mathsf{LocatedReal}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the transport, replay, provenance, and local naming rows H, C, P, N remain visible. The export provides no semantic liminf operator, host topology, selected approximants, quotient equality, choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousPublicEpigraphExport`

Proof. Start with the obligation closure package. It identifies $X, F, W, R, E, O, H, C, P, N$ as the complete obligation inventory and fixes carrier admission, epigraph-window exactness, located readback, transport replay, provenance, and Real-seal nonescape on those rows.

Apply the located epigraph route and located-real obligation route. These force a public epigraph reader to expose W through $\mathsf{StreamName}^{\uparrow}$, R through $\mathsf{RegSeqRat}^{\uparrow}$, and E, O through the lower-real, upper-real, and located-real surfaces before the lower-bound claim can be used.

Use Real-seal nonescape and the scoped route next. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of the same displayed rows, while H, C, P, N preserve only componentwise transport, displayed replay, provenance, and local naming.

Finally apply public route export. Since that export is already the public composite of the displayed lower-bound route and none of the refused analytic principles is a carrier coordinate, every public epigraph consumer receives exactly the finite route above. □

Theorem 1536.87 (LowerSemicontinuous located epigraph route)**.** *Every located epigraph read from an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must expose the finite schedule row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the lower-real epigraph row E, and the comparison ledger O before the located consumer can use the lower-bound evidence. The read is therefore routed through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal only as displayed rows, not as a host topology or semantic liminf operator.*

Proof. Start with the root epigraph carrier. It orders the lower-bound read as $X, F, W, R, E, O, H, C, P, N$, so the located consumer cannot inspect O before the scheduled window and regular rational value row are visible.

Apply located readback stability and located epigraph transport induction. They keep the lower-bound witness in E attached to the same comparison ledger O after transport through the located and lower-real surfaces.

Finally use the Real handoff. The terminal seal is reached only downstream of the same displayed route, and the carrier has no coordinate for host topology, liminf semantics, quotient equality, selected sequences, choice, compactness, or completeness. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedEpigraphRoute`

Theorem 1536.88 (LowerSemicontinuous located-real obligation route)**.** *Every* $\mathsf{LocatedReal}^{\uparrow}$*-facing read of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must expose the full finite route*

$$X, F, W, R, E, O, H, C, P, N$$

before any lower-bound claim is used. The route opens the $\mathsf{StreamName}^{\uparrow}$ *window* W*, reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *value row* R*, compares the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *epigraph rows through* E *and* O*, exposes the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison data, and reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after transport, replay, provenance, and local naming remain visible.*

Proof. Project the carrier and obligation closure package. They make $X, F, W, R, E, O, H, C, P, N$ the complete displayed route for lower-bound reads, with the schedule, readback, epigraph row, comparison ledger, transport, replay, provenance, and local naming all present before the claim is consumed.

Use the located epigraph route above. It places every located lower-bound read through W, R, E, O and accounts for the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$ rows as displayed data rather than as a host topology.

Finally apply Real-seal nonescape. The terminal seal cannot be reached by semantic liminf data, selected approximants, quotient equality, countable choice, open-cover compactness, or ambient Real completeness, because none of those principles is a row of the accepted packet. □

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousLocatedRealObligationRoute`

Theorem 1536.89 (LowerSemicontinuous Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *obligation package remains downstream of the finite route*

$$X, F, W, R, E, O, H, C, P, N.$$

The route first opens the $\mathsf{StreamName}^{\uparrow}$ *finite window* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, then reads the* $\mathsf{LocatedReal}^{\uparrow}$ *threshold rows carried by the comparison ledger* O*, then consumes the one-sided* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *ledgers attached to the epigraph row* E *and comparison row* O*, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal through* H, C, P, N*. The seal can certify only this displayed L10 source route; it cannot export a semantic liminf operator, quotient real equality, selected approximating sequence, countable-choice schedule, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSealFullRouteNonescape`

Proof. Begin with the Real-seal window exhaustion theorem. It confines every Real-facing read to the displayed route through the scheduled $\mathsf{StreamName}^{\uparrow}$ window W, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, epigraph row E, and comparison ledger O.

Use the located epigraph route and the located-real obligation route to bind that route to its threshold rows. These theorems place the $\mathsf{LocatedReal}^{\uparrow}$ comparison data inside O and force the $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ one-sided ledgers in E, O to remain visible before a lower-bound claim is consumed.

Use the obligation closure package above to account for the same rows as the complete obligation inventory. The package adds no analytic coordinate beyond $X, F, W, R, E, O, H, C, P, N$.

Transport, replay, provenance, and local naming preserve that inventory. Since the refused principles are absent from the carrier and from the Real-seal handoff, the endpoint cannot escape the displayed finite route. □

Theorem 1536.90 (LowerSemicontinuous scoped route)**.** *Every scoped consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must traverse the finite route*

$$X, F, W, R, E, O, H, C, P, N$$

before any lower-bound claim is consumed. The route exposes the source row X*, the graph row* F*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *epigraph rows* E, O*, the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison data, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *still visible.*

The scoped route exports no semantic liminf operator, host topology, selected approximants, countable-choice schedule, quotient equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousScopedRoute`

Proof. Start with the obligation closure package. It identifies $X, F, W, R, E, O, H, C, P, N$ as the complete obligation inventory and refuses host topology, semantic liminf data, quotient equality, selected approximating sequences, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness at the package boundary.

Apply the located epigraph route and the located-real obligation route. These place every located lower-bound read through W, R, E, O and keep the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$ rows visible before the lower-bound claim is used.

Finally apply Real-seal nonescape. The terminal $\mathsf{Real}^{\uparrow}$ seal remains downstream of the same displayed route, while H, C, P, N preserve transport, replay, provenance, and naming. Since no refused principle is a coordinate of the accepted packet, a scoped consumer receives only the finite route stated above. □

Theorem 1536.91 (LowerSemicontinuous public route export)**.** *Every public consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet receives the finite route*

$$X, F, W, R, E, O, H, C, P, N$$

as the exported lower-bound route. The route factors through $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any lower-bound claim is consumed, and exports no semantic liminf operator, host topology, selected approximants, quotient equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousPublicRouteExport`

Proof. Start with the obligation closure package. It identifies $X, F, W, R, E, O, H, C, P, N$ as the complete lower-bound obligation inventory and adds no analytic coordinate beyond those displayed rows.

Use the located epigraph route and the located-real obligation route. Together they force every lower-bound read through the finite window W, the regular rational readback R, the lower and upper epigraph rows E, O, and the $\mathsf{LocatedReal}^{\uparrow}$ comparison data before the claim is available to a consumer.

Finally apply Real-seal nonescape and the scoped route. The terminal $\mathsf{Real}^{\uparrow}$ seal remains downstream of the same displayed route, and H, C, P, N preserve transport, replay, provenance, and local naming. The public export is therefore the composite of the existing route theorems and contributes no semantic liminf operator, host topology, selected approximant, quotient equality, countable-choice schedule, open-cover compactness theorem, or ambient completeness principle. □

Theorem 1536.92 (LowerSemicontinuous public-route exactness triad)**.** *Every public* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer factors through exactly three visible row groups. The first group is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window-readback surface* W, R*. The second group is the* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*, and* $\mathsf{LocatedReal}^{\uparrow}$ *epigraph-comparison surface* E, O*. The third group is the terminal* $\mathsf{Real}^{\uparrow}$ *seal together with the structural rows* H, C, P, N *for componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. No public route exports a semantic liminf operator, host topology, selected approximants, quotient equality, countable-choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness outside this triad.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousPublicRouteExactnessTriad`

Proof. Start with the obligation closure package. It fixes $X, F, W, R, E, O, H, C, P, N$ as the complete public inventory, so the source and graph rows only feed the three consumer-facing groups named in the statement.

Use the located epigraph route and the located-real obligation route. They place every lower-bound read through the window-readback rows W, R and then through the epigraph-comparison rows E, O, with $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, and $\mathsf{LocatedReal}^{\uparrow}$ all visible before the claim is consumed.

Apply Real-seal nonescape, the scoped route, and the public route export. These theorems put the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same displayed route and keep H, C, P, N attached as transport, replay, provenance, and local naming rows.

The three groups therefore exhaust the public route. Since the carrier has no coordinate for semantic liminf data, host topology, selected approximants, quotient equality, countable choice, open-cover compactness, or ambient completeness, no public consumer can obtain such data outside the triad. □

Theorem 1536.93 (LowerSemicontinuous real-series sibling route)**.** *Every* $\mathsf{RealSeries}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet in chapter 1581 must read the lower-semicontinuous endpoint through the displayed* $\mathsf{Real}^{\uparrow}$ *seal after the finite rows*

$$X, F, W, R, E, O, H, C, P, N$$

remain visible. The series-facing route may consume lower-bound information only after the $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *epigraph rows,* $\mathsf{LocatedReal}^{\uparrow}$ *comparison ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$ *row have all been exposed.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousRealSeriesSiblingRoute`

Proof. Apply the obligation closure package. It fixes $X, F, W, R, E, O, H, C, P, N$ as the complete lower-semicontinuous inventory and keeps semantic liminf data, host topology, quotient equality, selected approximants, countable choice, open-cover compactness, and ambient completeness outside the carrier.

Use the located epigraph route and located-real obligation route. They force the lower-bound read through W, R, E, O before the endpoint can be handed to a located or real-facing consumer.

Finally apply Real-seal nonescape and public route export. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of the same finite route, so a real-series consumer receives only the displayed endpoint rows and cannot bypass the lower-semicontinuous packet. □

Theorem 1536.94 (LowerSemicontinuous epigraph dependency route)**.** *Every* $\mathsf{Epigraph}^{\uparrow}$ *or* $\mathsf{RealSeries}^{\uparrow}$ *consumer using lower-bound information from an accepted* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *packet must read the same route*

$$X, F, W, R, E, O, H, C, P, N$$

through chapter 6224 and chapter 1581 before the lower-bound row can be consumed. The route displays the finite domain row, function row, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, lower and located-real endpoint rows, comparison ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$ *row. It exports no semantic liminf data, host topology, quotient equality, selected approximants, countable choice, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.LowerSemicontinuousUp.LowerSemicontinuousEpigraphDependencyRoute`

Proof. Apply Definition 1536.93. It already forces every real-series read through $X, F, W, R, E, O, H, C, P, N$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Use the obligation closure package for the same carrier. The package fixes the lower-semicontinuous inventory as the displayed finite route and keeps liminf semantics, topology, quotient equality, choice-selected approximants, open-cover compactness, and ambient completeness outside the packet.

The epigraph consumer reads the lower-bound row through the same E, O endpoint and comparison rows before passing to chapter 6224. Hence both epigraph and real-series consumers depend on the identical sibling route rather than on an external lower-bound principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1536.86 and Definition 1536.74 give the bridge-facing public epigraph surface over finite source windows, epigraph rows, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, NameCert, and the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LowerSemicontinuous}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close host topological openness, semantic liminf operators, quotient real equality, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Lean-side verification remains to certify the public epigraph export route and any standard lower-semicontinuous bridge theorem over the same epigraph-window rows and Real seal.

Constructive story: The construction starts from finite source windows, reads regular rational value rows, records lower-real epigraph witnesses, transports them through hsame and Cont ledgers, checks located readbacks, and hands the accepted route to Real-facing consumers without importing host topology.

Chapter 1537

A Concrete Naming Certificate for $\mathsf{ClosedGraphTheorem}^{\uparrow}$

Following the Banach closed graph theorem tradition in Banach and Bourbaki-style functional analysis, the $\mathsf{ClosedGraphTheorem}^{\uparrow}$ packet records the finite BEDC certificate for reading a graph-closed linear or structured map as a locally bounded route between complete normed or complete-metric presentations. The packet is anchored in the Banach carrier of chapter 90, the complete-metric surface of chapter 128, the metric observation rows of chapter 57, and the bounded-map comparison interface of chapter 410. Its finite real-facing names pass through chapter 126, chapter 301, and chapter 38, so the certificate exposes only displayed graph closure, transported Cauchy control, and local boundedness rows rather than host graph equality, quotient function spaces, choice-selected limits, or ambient completeness.

Definition 1537.1 (ClosedGraphTheorem carrier)**.** *A* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, T, \Gamma, C_M, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *row, including the displayed metric observations through the* $\mathsf{Met}^{\uparrow}$ *surface used to read Cauchy input windows.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *row, including the displayed metric observations through the* $\mathsf{Met}^{\uparrow}$ *surface used to read Cauchy output windows.*
3. *T is the map row, read as a finite graph or action table whose bounded-map comparisons may be inspected through the* $\mathsf{LipschitzMap}^{\uparrow}$ *interface when a modulus or local bound is displayed.*
4. *Γ is the graph row, consisting of finite source–target observation pairs and the graph-membership witnesses by which a consumer reads that y is the displayed image of x.*
5. *C_M is the closed-graph witness row, recording the paired Cauchy-route test: source windows converge in X, output windows converge in Y, graph rows persist along the paired route, and the target limit row is accepted as the graph value.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting finite source, target, graph, and modulus windows for each inspection.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplying regular rational distance and bound observations for the scheduled metric windows.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row through which numeric distance, bound, and convergence reads are consumed.*
9. *H records componentwise* hsame *transport for source, target, map, graph, closed-graph, schedule, readback, seal, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from source and target admission through stream scheduling, regular rational readback, graph inspection, closed-graph transfer, bound extraction, and real-seal handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{ClosedGraphTheorem}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedGraphTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for the open mapping theorem, bounded inverse theorem, host graph equality, quotient function spaces, choice-selected limits, or ambient completeness beyond the source and target completeness rows explicitly displayed in X and Y.

Theorem 1537.2 (ClosedGraphTheorem NameCert obligations)**.** *For an accepted* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *carrier $G = (X, Y, T, \Gamma, C_M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedGraphTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source complete row, target complete row, map row, graph row, closed-graph witness, stream schedule, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*
2. *source and target completeness routing, requiring every Cauchy input and Cauchy output inspection to pass through the* $\mathsf{Banach}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *rows in X and Y and their displayed* $\mathsf{Met}^{\uparrow}$ *observations;*
3. *graph-membership exactness, requiring every accepted graph read to be a finite row of Γ over the scheduled source and target windows;*
4. *closed-graph transfer, requiring the paired limit route in C_M to preserve Γ along the same stream schedule and regular rational readback;*
5. *bounded-route extraction, requiring any local boundedness or Lipschitz-facing consumer to read through T, Γ, C_M, S, R, and E rather than through an ambient function-space theorem;*
6. *classifier stability, requiring transported packets with the same displayed rows to be identified by the carried* hsame *data and replayed by the same* Cont *route;*
7. *ledger non-escape, stating that no open mapping theorem, bounded inverse theorem, host graph equality, quotient function space, choice-selected limit, or ambient-completeness principle is exported by the packet.*

Proof. Carrier admission is projection from Definition 1537.1. The listed rows are exactly the source, target, map, graph, closed-graph, schedule, readback, real-seal, transport, replay, provenance, and local name data available to a consumer.

The completeness routing is inherited only as displayed source and target rows. The source side is read through chapter 90 or chapter 128, and the target side is read through the same complete-metric vocabulary; the metric observations used by both sides are those of chapter 57.

Graph-membership exactness is the role of Γ. A graph consumer cannot infer equality of host functions; it reads finite source–target observation pairs, and those pairs are scheduled by S and measured through R and E using the naming route of chapter 126, chapter 301, and chapter 38.

Closed-graph transfer is then the finite witness row C_M. It records the paired Cauchy route, the persistence of graph rows along that route, and the accepted target limit row. When a bounded-map consumer is present, the read is compared through the interface of chapter 410, but the packet exports only the displayed route and not an external open-mapping or inverse theorem.

Finally H and C transport and replay the same fields, while P and N package the local certificate. Since the carrier fields are exhausted by $X, Y, T, \Gamma, C_M, S, R, E, H, C, P, N$, the excluded principles have no row from which a consumer could read them. □

Theorem 1537.3 (ClosedGraphTheorem graph-closure route)**.** *Let $G = (X, Y, T, \Gamma, C_M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *carrier. Every graph-closure read exported by G factors through the displayed route*

$$X, Y, T, \Gamma, S, R, E, C_M, H, C, P, N.$$

Thus a convergent source schedule and a convergent target schedule identify a graph value only after their finite graph rows have been replayed through C_M and sealed through the StreamName, RegSeqRat, and Real rows; the route does not assert host graph equality or equality inside a quotient function space.

Proof. Begin with the source and target Cauchy inspections. The rows X and Y expose the complete Banach or complete-metric data, and their metric windows are the observations read by the route rather than ambient points.

The map and graph information is then read through T and Γ. The schedule S selects the finite source, target, and graph windows; the regular readback R and real seal E measure the requested distances and bounds.

The closed-graph witness C_M is the only row that turns the paired Cauchy route into a graph-value read. It requires the graph rows to persist along the same schedule and accepts the target limit row as a displayed graph value.

Transport by H and replay by C preserve the same route after classifier movement, and P, N record its provenance and local certificate. Because no quotient function-space coordinate or host graph equality coordinate is present, the graph-closure read is exactly the displayed finite route. □

Theorem 1537.4 (ClosedGraphTheorem non-escape boundary)**.** *For every accepted* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *carrier $G = (X, Y, T, \Gamma, C_M, S, R, E, H, C, P, N)$, any boundedness, continuity-facing, or graph-facing conclusion exported by the packet is confined to the rows displayed in G. In particular, the packet does not export the open mapping theorem, the bounded inverse theorem, host graph equality, quotient function spaces, choice-selected limits, or ambient completeness beyond the source and target rows X and Y.*

Proof. Read an exported conclusion from the carrier. A graph-facing read must use Γ and C_M; a boundedness or continuity-facing read must use T together with the same graph, schedule, readback, and seal rows.

The source and target completeness information is limited to X and Y. These rows may expose the Banach or complete-metric certificate and the associated metric observations, but they do not add any global function-space carrier.

The numeric side is likewise finite. Stream windows come from S, regular rational observations from R, and real-facing distance or bound reads from E.

All remaining movement is componentwise transport by H, continuation replay by C, and package bookkeeping by P, N. Since no excluded principle is a coordinate of G, no accepted consumer can obtain one from the $\mathsf{ClosedGraphTheorem}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedGraphTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedGraphTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1537.1, Definition 1537.2, Definition 1537.3, and Definition 1537.4 seed the finite closed-graph certificate over displayed complete source and target rows, graph witnesses, StreamName schedules, RegSeqRat readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedGraphTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the open mapping theorem, bounded inverse theorem, host graph equality, quotient function spaces, choice-selected limits, or ambient completeness beyond the displayed source and target rows.

Upgrade path: Obligation closure requires checked carrier admission, graph-membership exactness, closed-graph transfer, bounded-route extraction, classifier stability, and ledger non-escape for the same packet.

Constructive story: The construction starts from displayed complete source and target rows, schedules finite graph observations, seals distance and bound reads through RegSeqRat and Real data, and exports only the local graph-closure route recorded by hsame and Cont ledgers.

Chapter 1538

A Concrete Naming Certificate for $\mathsf{OpenMappingTheorem}^{\uparrow}$

Following Banach and Rudin-style functional analysis, the $\mathsf{OpenMappingTheorem}^{\uparrow}$ packet records the finite BEDC certificate for reading a surjective bounded linear map between Banach rows as an open-ball image route with a bounded-inverse-facing handoff. It is downstream of the source and target Banach certificates of chapter 90, the bounded linear map surface of chapter 48, the normed real-line and real-seal surfaces of chapter 1500 and chapter 38, and the closed-graph consumer boundary of chapter 1537. The packet exposes only displayed finite rows: a source Banach row, a target Banach row, a bounded linear map row, a surjective image ledger, an open-ball image witness, an inverse-window ledger, transport and replay rows, package provenance, and a local NameCert row.

Definition 1538.1 (Open mapping theorem carrier)**.** *An* $\mathsf{OpenMappingTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, Y, T, S, B, I, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm and metric observations through the* $\mathsf{NormedRealLine}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *surfaces used to read source balls.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm and metric observations through the same real-facing surfaces used to read target balls.*
3. *T is the bounded* $\mathsf{LinMap}^{\uparrow}$ *row from X to Y, carrying only the finite action, linearity, and boundedness data displayed by the map certificate.*
4. *S is the surjective image ledger, recording that every displayed target window admitted by Y has a source preimage window scheduled through X and read through T.*
5. *B is the open-ball image witness, recording finite radii, norm comparisons, image windows, and ball-containment rows that turn source ball data into a target-neighborhood read.*
6. *I is the inverse-window ledger, recording the bounded-inverse-facing handoff: a target window may request a displayed source window only through S, B, and the same Banach and linear-map rows.*
7. *H records componentwise* hsame *transport for the source, target, map, surjectivity, open-ball, inverse-window, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from Banach admission through bounded linear map inspection, surjective image lookup, open-ball witness readback, inverse-window handoff, and closed-graph consumer routing.*

9. *P is the* Pkg *provenance over* $\mathsf{OpenMappingTheorem}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{NormedRealLine}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedGraphTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OpenMappingTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for host quotient function spaces, choice-selected inverse maps, non-displayed inverse functions, or ambient completeness beyond the Banach rows X and Y.

Theorem 1538.2 (Open mapping theorem NameCert obligations)**.** *For an accepted* $\mathsf{OpenMappingTheorem}^{\uparrow}$ *carrier $O = (X, Y, T, S, B, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OpenMappingTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source Banach row, target Banach row, bounded linear map row, surjective image ledger, open-ball image witness, inverse-window ledger, componentwise transport, continuation replay, provenance, and local naming rows;*

2. *classifier stability, requiring transported packets with the same displayed rows to be identified by the carried* hsame *data and replayed by the same* Cont *route;*

3. *image-open ledger, requiring every target-neighborhood read to pass through the source ball data in X, the target ball data in Y, the bounded map row T, the surjective ledger S, and the open-ball witness B;*

4. *bounded-inverse handoff, requiring every inverse-facing consumer to read a source window through S, B, and I before any inverse boundary is exposed;*

5. *no-escape, stating that no host quotient function space, choice-selected inverse map, non-displayed inverse function, or ambient completeness principle is exported beyond the displayed Banach rows.*

Proof. Carrier admission is projection from Definition 1538.1. The tuple supplies exactly the source, target, map, surjectivity, open-ball, inverse-window, transport, replay, provenance, and local name rows, so a consumer has no other coordinate from which to read an open-mapping conclusion.

Classifier stability is componentwise. The transport row H compares the same displayed coordinates, and the continuation row C replays the same route after transport. Thus moving the packet cannot replace the Banach, LinearMap, norm, Real, or ClosedGraphTheorem dependencies by a host function-space principle.

For the image-open ledger, a target-neighborhood read begins with the source and target Banach rows X and Y. The norm and real observations used by those rows pass through chapter 1500 and chapter 38. The bounded map row T is then read through chapter 48, the surjective ledger S supplies the displayed preimage window, and B records the finite ball-containment witness.

For the bounded-inverse handoff, the inverse-facing consumer receives only the ledger row I. That row is downstream of S and B, so it cannot expose a selected inverse map; it can only report the displayed source-window request already justified by the open-ball image witness.

Finally P and N package the same finite route. Since the carrier fields are exhausted by $X, Y, T, S, B, I, H, C, P, N$, the excluded quotient, selection, inverse-function, and ambient-completeness principles have no row in the packet. □

Theorem 1538.3 (Open mapping theorem closed-graph consumer route)**.** *For every accepted* $\mathsf{OpenMappingTheorem}^{\uparrow}$ *carrier* $O = (X, Y, T, S, B, I, H, C, P, N)$*, every bounded-inverse or closed-graph consumer reads through the displayed route*

$$X, Y, T, S, B, I, H, C, P, N.$$

In particular, the consumer must pass through the Banach rows of chapter 90, the bounded linear-map row of chapter 48, the norm and Real rows of chapter 1500 and chapter 38, and the closed-graph boundary of chapter 1537 before any inverse boundary is exposed.

Proof. Start with the Banach source and target rows. A consumer can inspect source and target balls only through X and Y, and the finite norm data in those rows is read through the normed real-line and real-seal surfaces.

Next inspect the map route. The row T supplies bounded linear-map data, while S supplies the displayed surjective image ledger. No inverse-facing conclusion is available before these two rows have been read.

The open-ball and inverse-window rows then determine the handoff. The row B records the finite image-open witness, and I records the bounded-inverse-facing source-window request that depends on that witness.

Finally the route is transported, replayed, and packaged by H, C, P, N. A closed-graph consumer may compare this handoff with chapter 1537, but it still reads only the displayed Banach, LinearMap, norm, Real, and closed-graph rows before any inverse boundary is exposed. □

Closure status for $\mathsf{NameCert}_{\mathsf{OpenMappingTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpenMappingTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1538.1, Definition 1538.2, and Definition 1538.3 seed the finite open-mapping certificate over displayed Banach, LinearMap, normed Real, surjective-image, open-ball, inverse-window, and closed-graph consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpenMappingTheorem}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.OpenMappingTheoremUp`

Bridge status: none

Not claimed: This chapter does not close host quotient function spaces, choice-selected inverse maps, non-displayed inverse functions, or ambient completeness beyond the displayed Banach rows.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, image-open ledger exactness, bounded-inverse handoff, closed-graph consumer routing, and no-escape for the same packet.

Constructive story: The construction starts from displayed source and target Banach rows, reads the bounded LinearMap action through finite image windows, records surjectivity and open-ball witnesses as ledgers, and exposes bounded-inverse-facing consumers only through the same hsame and Cont route.

Chapter 1539

A Concrete Naming Certificate for $\mathsf{CauchyCompletionCounit}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley completion terminology, the $\mathsf{CauchyCompletionCounit}^{\uparrow}$ packet records the finite BEDC certificate for retracting a completion-facing name back through the terminal real seal whenever the name is already displayed by the Real/RegSeqRat route. It is the counit side of the boundary opened by the unit packet of chapter 1527 and the reflection packet of chapter 403. The packet consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational readback of chapter 301, the finite-window naming surface of chapter 126, and the dyadic tolerance ledger of chapter 291; it does not form quotient streams, compare host real equality, choose representatives, assume countable choice, or appeal to ambient completeness.

Definition 1539.1 (CauchyCompletionCounit carrier)**.** *A* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, E, U, R, S, D, H, C, P, N)$$

with the following visible rows:

1. *M is the completion-facing source row whose accepted reads must be returned to the displayed real-seal boundary.*
2. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which the returned read is consumed.*
3. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route that records how source-side data entered the completion-facing surface.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplying regular rational observations before the seal is inspected.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the regular readback is named.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that accounts for every precision request used by S and R.*
7. *H records componentwise* hsame *transport for M, E, U, R, S, D and the structural rows.*
8. *C records displayed* Cont *replay from completion-facing reads through the unit route, window row, dyadic ledger, regular readback, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCounit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, host real equality, selected representatives, countable-choice schedules, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1539.2 (CauchyCompletionCounit NameCert obligations)**.** *For an accepted* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *carrier $Q = (M, E, U, R, S, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionCounit}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the completion-facing source row, terminal Real seal, unit route, regular readback, stream window, dyadic ledger, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *counit exposure, requiring every completion-facing read in M to return through U before the seal E can be consumed;*
3. *readback exactness, requiring every returned read to pass through S, D, and R before reaching E;*
4. *classifier stability, requiring transported packets with the same M, E, U, R, S, D route to be identified by the carried* hsame *and* Cont *data;*
5. *ledger non-escape, stating that no quotient completion equality, host real equality, selected representative, countable-choice schedule, open-cover compactness, ambient-completeness theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Proof. Carrier admission is projection from Definition 1539.1. The tuple lists exactly the completion-facing source, terminal Real seal, unit route, regular readback, stream window, dyadic ledger, transport, replay, provenance, and local name.

Counit exposure reads the completion-facing source row M together with the unit route U. A consumer cannot inspect E directly from M; the displayed replay row C first returns the read through the route that admitted the source-side data.

Readback exactness is then forced by the order of the packet. The finite-window row S names the request, the dyadic ledger D accounts for its tolerance, and the regular readback row R supplies the rational observation before the terminal seal E is consumed.

Classifier stability is componentwise transport along H followed by replay through C. Since the carrier fields are exhausted by $M, E, U, R, S, D, H, C, P, N$, the excluded quotient, host-equality, representative-choice, compactness, ambient-completeness, and real-completeness principles have no row from which a consumer could read them. □

Theorem 1539.3 (CauchyCompletionCounit real-seal retraction)**.** *Let $Q = (M, E, U, R, S, D, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by Q factors through the displayed retraction route*

$$M \longmapsto U \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

Thus the counit returns completion-facing source data to the terminal Real seal only as finite BEDC route data, not as a quotient completion point, not as host real equality, not as a selected representative, and not as countable choice or ambient completeness.

Proof. Project the accepted carrier and begin with the completion-facing source row M. The packet exposes M only with the unit route U, so any retraction read starts by returning through the displayed route that admitted the source row.

The finite observation part is then fixed. The stream window S supplies the named finite request, the dyadic ledger D records its precision, and the regular rational readback R supplies the observation consumed by the real seal.

The terminal row E is reached only after that finite route has been replayed. Transport by H and continuation by C preserve the same route for equivalent packets, while P and N record its provenance and local certificate.

No further row is available in the carrier. Therefore the exported real-seal read is exactly the displayed retraction route, with no quotient point, host equality, selected representative, choice schedule, or ambient completeness witness. □

Theorem 1539.4 (CauchyCompletionCounit unit-counit triangle)**.** *For every accepted* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *carrier* $Q = (M, E, U, R, S, D, H, C, P, N)$ *whose completion-facing source row was admitted by the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route* U*, the composite*

$$M \longmapsto U \longmapsto S, D, R \longmapsto E$$

has the same displayed $\mathsf{Real}^{\uparrow}$ *seal read as the direct regular route*

$$S, D, R \longmapsto E.$$

The triangle is witnessed by the shared finite rows S, D, R, E, H, C, P, N *and does not assert equality of quotient completions, host real equality, a chosen representative, countable choice, or ambient completeness.*

Proof. Read the unit side first. The row U records the admission of source-side data into the completion-facing row M, and the counit packet requires any return from M to pass back through the same displayed route before E is consumed.

Read the direct regular side next. The rows S, D, and R already name the finite window, tolerance ledger, and regular rational readback that feed the terminal Real seal E.

Both sides therefore present the same seal-facing route once the completion-facing row has been returned through U. Componentwise transport in H and replay in C identify the shared rows, while P and N record that the comparison is internal to the same local certificate.

The carrier contains no quotient-completion equality, host real equality, selected representative, choice schedule, or ambient completeness coordinate. The triangle is therefore a finite-route triangle over S, D, R, E, not an external equality principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionCounit}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionCounit}^{\uparrow}$ packet is generated from a completion-facing source row, terminal Real seal, CauchyCompletionUnit route, RegSeqRat readback, StreamName finite window, DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionCounit}^{\uparrow}}$, seed)

Scope closed: Definition 1539.1, Definition 1539.2, Definition 1539.3, and Definition 1539.4 seed the finite counit/retraction side of the constructive completion boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionCounit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, host real equality, selected representatives, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, counit exposure, readback exactness, classifier stability, real-seal retraction, unit-counit triangle, and ledger non-escape for the same packet.

Chapter 1540

A Concrete Naming Certificate for $\mathsf{CauchyCompletionProduct}^{\uparrow}$

Following Bourbaki product uniformities, Kelley completion terminology, and Bishop–Bridges constructive metric completion, the $\mathsf{CauchyCompletionProduct}^{\uparrow}$ packet records the finite BEDC certificate for forming the completion-facing product of two displayed Cauchy-completion routes. It consumes the unit route of chapter 1527, the counit route of chapter 1539, the monadic completion boundary of chapter 378, the reflection packet of chapter 403, the regular readback surface of chapter 301, the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet is a finite product handoff for two coordinate routes; it is not a quotient completion product, not a selected pair of representatives, not countable choice, not a host product theorem, and not an ambient completeness principle.

Definition 1540.1 (CauchyCompletionProduct carrier)**.** *A* $\mathsf{CauchyCompletionProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (L, R, U_L, U_R, K_L, K_R, M, F, S_L, S_R, D_L, D_R, G_L, G_R, E_L, E_R, H, C, P, N)$$

with the following visible rows:

1. *L and R are the left and right completion-facing source rows whose product read is being exposed.*
2. *U_L and U_R are the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *routes admitting the two displayed source rows into completion-facing consumers.*
3. *K_L and K_R are the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *routes returning coordinate reads to their terminal real seals when the coordinates are already displayed.*
4. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *bind-and-return ledger tying the two coordinate routes to a single completion-facing product packet.*
5. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *seal that keeps the product read on the displayed completion boundary rather than on an external host product.*
6. *S_L and S_R are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the two coordinates.*
7. *D_L and D_R are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers used by the two coordinate window reads.*
8. *G_L and G_R are the coordinate* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows assembled from the corresponding window and dyadic ledger rows.*

9. *E_L and E_R are the* $\mathsf{Real}^{\uparrow}$ *terminal seal rows reached after the coordinate readbacks.*
10. *H records componentwise* hsame *transport for all coordinate, monad, reflection, window, dyadic, seal, continuation, provenance, and naming rows.*
11. *C records displayed* Cont *replay from the product read through the coordinate unit routes, coordinate counit routes, monad ledger, reflection seal, windows, dyadic ledgers, regular readbacks, and real seals.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionProduct}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProduct}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, host product equality, selected representatives, countable-choice schedules, open-cover compactness, product completeness of $\mathsf{Real}^{\uparrow}$*, or an ambient completeness theorem.*

Theorem 1540.2 (CauchyCompletionProduct NameCert obligations)**.** *For an accepted* $\mathsf{CauchyCompletionProduct}^{\uparrow}$ *carrier Z, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProduct}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two completion-facing source rows, the two unit routes, the two counit routes, the monad ledger, the reflection seal, the two stream-window rows, the two dyadic ledgers, the coordinate regular readbacks, the two real seals, componentwise transport, continuation replay, provenance, and local naming;*
2. *coordinatewise completion handoff, requiring each product read to split through the left and right* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *routes, return through the corresponding* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *routes when a coordinate is already displayed, and expose the two* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *routes before the product read is consumed;*
3. *product seal compatibility, requiring the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *ledger and* $\mathsf{CompletionReflection}^{\uparrow}$ *seal to bind the two coordinate routes into a single product-facing packet without adding a host product equality or quotient-completion equality;*
4. *transport stability, requiring* hsame *transport and displayed* Cont *replay to preserve the same coordinate routes, monad ledger, reflection seal, package provenance, and local* NameCert *row;*
5. *ledger exactness, stating that the packet exports exactly the rows sourced by* Pkg *over* $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and exports no selected representatives, countable-choice schedule, product-completeness theorem for* $\mathsf{Real}^{\uparrow}$*, or ambient completeness principle.*

Proof. Carrier admission is projection from Definition 1540.1. The tuple lists exactly the two source rows, the unit and counit coordinate routes, the monad ledger, the reflection seal, the window, tolerance, readback, seal, transport, replay, provenance, and local naming rows.

The coordinate handoff reads the product packet one coordinate at a time. The left coordinate must pass through U_L, K_L, S_L, D_L, E_L, and the right coordinate must pass through U_R, K_R, S_R, D_R, E_R, with the regular readback route named by the surrounding $\mathsf{RegSeqRat}^{\uparrow}$ rows before either terminal seal is consumed.

The product seal is then the finite bind-and-reflect step. The monad ledger M ties the two coordinate routes into one product-facing route, and the reflection row F keeps that route inside the displayed completion boundary rather than creating an external host product.

Transport stability is componentwise transport through H followed by replay through C. Since the provenance row P and local naming row N source only the displayed rows, the excluded quotient, representative-choice, product-completeness, and ambient-completeness principles have no row from which a consumer could read them. □

Theorem 1540.3 (CauchyCompletionProduct coordinate handoff)**.** *For an accepted* $\mathsf{CauchyCompletionProduct}^{\uparrow}$ *carrier*

$$Z = (L, R, U_L, U_R, K_L, K_R, M, F, S_L, S_R, D_L, D_R, G_L, G_R, E_L, E_R, H, C, P, N),$$

every completion-facing product consumer factors through the coordinate route

$$L, R \longmapsto U_L, U_R \longmapsto K_L, K_R \longmapsto S_L, S_R \longmapsto D_L, D_R \longmapsto G_L, G_R \longmapsto E_L, E_R \longmapsto M, F, H, C, P, N.$$

Thus the product handoff is coordinatewise finite BEDC data, not a quotient product, not a host product equality, not a selected pair of representatives, and not countable choice or ambient completeness.

Proof. Project the accepted carrier and begin with the two source rows L and R. A product-facing consumer cannot inspect a product point before the unit rows U_L and U_R have admitted the two displayed coordinate routes.

For each coordinate, the counit row returns already displayed data to the real-facing boundary. The stream-window rows S_L, S_R and dyadic ledgers D_L, D_R account for the finite precision requests that produce the corresponding regular readbacks and terminal real seals E_L, E_R.

The monad row M binds these two coordinate reads into one completion-facing packet, and the reflection row F records that the packet remains on the displayed completion boundary. Transport, continuation, provenance, and local naming preserve the same route through H, C, P, N.

No further coordinate occurs in the carrier. Hence every product consumer factors through the displayed coordinate handoff and cannot obtain quotient equality, host product equality, selected representatives, countable choice, or ambient completeness from this packet. □

Theorem 1540.4 (CauchyCompletionProduct non-escape)**.** *An accepted* $\mathsf{CauchyCompletionProduct}^{\uparrow}$ *carrier exports no read outside the finite packet*

$$(L, R, U_L, U_R, K_L, K_R, M, F, S_L, S_R, D_L, D_R, G_L, G_R, E_L, E_R, H, C, P, N).$$

In particular, it cannot expose a product-completion consumer without the two coordinate unit routes, the two coordinate counit routes, the monad ledger, the completion-reflection seal, the stream-window rows, the dyadic ledgers, the regular readback rows, the real seals, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Proof. Assume a consumer reads an accepted product packet. By carrier admission, the only available rows are the fields displayed in Definition 1540.1.

If the consumer drops a coordinate unit row, one source has not entered the completion-facing surface. If it drops a coordinate counit row, an already displayed coordinate cannot return to its real seal. If it drops a window, dyadic ledger, regular readback, or real seal, the corresponding coordinate read is not displayed at the requested precision.

If the consumer drops M or F, the two coordinate routes are no longer bound and reflected as a single completion-facing product route. If it drops H, C, P, or N, the packet loses transport, replay, provenance, or local certificate evidence.

The carrier has no additional row. Therefore no product-completion read escapes the displayed packet, and none of the excluded quotient, host-product, representative-choice, countable-choice, or ambient-completeness principles is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionProduct}^{\uparrow}$ packet is generated from two completion-facing source rows, coordinate unit routes, coordinate counit routes, a CauchyCompletionMonad ledger, a CompletionReflection seal, paired StreamName windows, paired DyadicRatCore ledgers, paired RegSeqRat readbacks, paired RealUp seals, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1540.1, Definition 1540.2, Definition 1540.3, and Definition 1540.4 seed the finite coordinatewise product-completion certificate through unit, counit, monad, reflection, readback, window, tolerance, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, host product equality, selected representatives, countable choice, open-cover compactness, product completeness of $\mathsf{Real}^{\uparrow}$, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, coordinatewise completion handoff, product seal compatibility, transport stability, ledger exactness, and non-escape for the same packet.

Chapter 1541

A Concrete Naming Certificate for $\mathsf{LocatedSet}^{\uparrow}$

Following Bishop and Bridges, a located subset of a metric space is not merely a subset predicate: it is a constructive distance interface from ambient points to the set, readable at finite rational or real tolerances. The $\mathsf{LocatedSet}^{\uparrow}$ packet records this standard object on the L10 real and completion frontier. It sits over the parent metric surface of chapter 57, the admission discipline of chapter 1519, the located-distance surface of chapter 1533, and the real terminal seal of chapter 38. The carrier gives compactness, total-boundedness, completion, and constructive-analysis consumers a displayed located-subset dependency without importing open-cover compactness, choice-selected nearest points, quotient subset equality, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1541.1 (LocatedSet carrier)**.** *A* $\mathsf{LocatedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, Q, R, E, T, H, C, P, N)$$

with the following visible rows:

1. *X is the parent* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance witness, metric-law ledger, and finite observation histories.*
2. *A is the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row. It records which parent observations are admitted as members of the displayed set and exposes no host subset equality or quotient membership.*
3. *Q is the located-distance ledger. For each accepted ambient point and positive rational tolerance request, it records the finite evidence used to locate the distance from the point to the admitted set.*
4. *R is the rational distance row. It packages the finite lower and upper rational comparison windows supplied by Q and keeps them tied to the same parent metric observations from X and admission data from A.*
5. *E is the real distance row. It seals the compatible rational windows as a* $\mathsf{Real}^{\uparrow}$ *distance name only after the parent metric, subspace admission, located-distance ledger, and rational rows have all been displayed.*
6. *T is the located-metric handoff row. It records that any consumer requiring a located metric route reads the parent metric witness, located-distance ledger, rational windows, and real seal through the interface named by chapter 1533.*

7. *H stores componentwise* hsame *transport for the parent metric row, admission row, located-distance ledger, rational and real distance rows, located-metric handoff, continuation replay, provenance, and local naming row.*
8. *C stores* Cont *replay from parent point observations to subspace admission, from admission to located-distance comparisons, from rational windows to the real distance seal, and from the seal to located-metric, completion, compactness, or total-boundedness consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSet}^{\uparrow}}$ *row.*

The classifier compares located-set packets only through the displayed tuple $X, A, Q, R, E, T, H, C, P, N$. It has no coordinate for open-cover compactness, choice-selected nearest points, quotient subset equality, an independently completed ambient space, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedSetUp.LocatedSetCarrier`

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetTasteGate single carrier alignment`

Theorem 1541.2 (LocatedSet NameCert obligations)**.** *For an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *carrier $L = (X, A, Q, R, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSet}^{\uparrow}}$ *obligations are exactly carrier admission, parent metric compatibility,* $\mathsf{MetricSubspace}^{\uparrow}$ *admission stability, located-distance stability, rational-window readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, located-metric consumer compatibility, and ledger non-escape. These obligations are discharged by the displayed route*

$$X \longmapsto A \longmapsto Q \longmapsto R \longmapsto E \longmapsto T$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 1541.1. The tuple lists exactly the parent metric row, subspace admission row, located-distance ledger, rational windows, real seal, located-metric handoff, transport, replay, provenance, and local naming row.

Parent metric compatibility and subspace admission stability are read first. A located-distance request can inspect an ambient point only through the point classifier and distance witness of X, and it can inspect set membership only through the admission row A inherited from the $\mathsf{MetricSubspace}^{\uparrow}$ discipline.

Located-distance stability is the role of Q. For each positive rational tolerance request, the ledger records the finite comparison data that locates the distance from the ambient point to the admitted set, while H transports the same point, admission, and comparison rows componentwise.

The rational and real readback obligations factor through R and E. The rational row stores the visible comparison windows, and the real row seals them as a $\mathsf{Real}^{\uparrow}$ distance name only after the preceding metric, admission, and locatedness rows have been displayed.

Finally, T, C, P, N replay, source, and name the same finite route for located-metric, completion, compactness, and total-boundedness consumers. Since the carrier has no coordinate for open-cover compactness, nearest-point selection, quotient subset equality, completed ambient space, or full real completeness, the listed obligations exhaust the local NameCert interface. □

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetCarrier namecert obligations`

Theorem 1541.3 (LocatedSet distance handoff)**.** *Every accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet exposes the distance from an ambient point to the displayed set only through the finite route*

$$X \longmapsto A \longmapsto Q \longmapsto R \longmapsto E.$$

Any $\mathsf{LocatedMetric}^{\uparrow}$*, completion, compactness, or total-boundedness consumer must read the parent metric row,* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row, located-distance ledger, rational comparison windows, and* $\mathsf{Real}^{\uparrow}$ *seal before it consumes located-subset distance data.*

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSet distance handoff`

Proof. Start with an ambient point request. The parent metric row X supplies the point classifier and all metric-distance observations available to the packet.

The admission row A then determines which displayed observations count as set members. No host subset equality or quotient membership is exposed, so the distance request must proceed through the finite admission surface rather than an ambient subset object.

The located-distance ledger Q supplies the constructive comparison windows for the requested tolerance. The rational row R packages those windows, and the real row E seals the compatible readback as a $\mathsf{Real}^{\uparrow}$ distance name.

Thus every downstream consumer receives located-set distance data through X, A, Q, R, E and the structural rows T, H, C, P, N. The route does not select a nearest point, prove open-cover compactness, identify quotient subsets, complete the ambient metric space, or import full $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1541.4 (LocatedSet Real-distance seal exactness)**.** *Let* $L = (X, A, Q, R, E, T, H, C, P, N)$ *be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Every Real distance read from* L *is exhausted by the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricSubspace}^{\uparrow} \longmapsto Q \longmapsto R \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{LocatedMetric}^{\uparrow}.$$

Concretely, the consumer first reads the parent metric row X*, then the displayed subspace admission row* A*, then the located-distance ledger* Q*, then the rational comparison windows* R*, then the Real distance seal* E*, and only then the located-metric handoff* T*, with transport and replay through* H, C, P, N*.*

The exactness claim exports no nearest-point selection, open-cover compactness, quotient subset equality, completed ambient metric space, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetRealDistanceSealExactness`

Proof. Project the carrier of Definition 1541.1. The only rows available to a distance consumer are the parent metric row, admission row, located-distance ledger, rational windows, Real seal, located-metric handoff, and structural rows.

Apply Definition 1541.2. It identifies parent metric compatibility, subspace admission stability, located-distance stability, rational-window readback, Real seal handoff, and located-metric compatibility as the local certificate obligations.

Now apply Definition 1541.3. The distance handoff forces every ambient-point read through X, A, Q, R, E before any completion, compactness, total-boundedness, or located-metric consumer can inspect the result.

The final handoff is therefore exact. The rows H, C, P, N transport, replay, source, and name this same route, while T exposes it to $\mathsf{LocatedMetric}^{\uparrow}$ consumers only after the Real seal has been displayed. Since the carrier has no coordinate for nearest points, open covers, quotient subset equality, completed ambient spaces, or full real completeness, none of those principles can appear in the Real-distance seal. □

Theorem 1541.5 (LocatedSet nearest-point refusal boundary)**.** *Every consumer asking for a nearest-point or projection endpoint from an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet must supply an additional endpoint witness row outside*

$$(X, A, Q, R, E, T, H, C, P, N).$$

The $\mathsf{LocatedSet}^{\uparrow}$ *packet itself exports only the parent* $\mathsf{Met}^{\uparrow}$ *row,* $\mathsf{MetricSubspace}^{\uparrow}$ *admission, located-distance ledger, rational comparison windows,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff, componentwise transport, continuation replay, provenance, and local* $\mathsf{NameCert}$ *row.*

Proof. Use Definition 1541.1. The carrier tuple lists the parent metric row, subspace admission row, located-distance ledger, rational windows, real distance row, located-metric handoff, and the four structural rows.

Apply Definition 1541.3. It forces every distance consumer through X, A, Q, R, E before the result can be read by located-metric, completion, compactness, or total-boundedness consumers.

Apply Definition 1541.4. The real distance read is exactly the displayed metric, subspace, located-distance, rational-window, real-seal, and located-metric route.

No row of $(X, A, Q, R, E, T, H, C, P, N)$ is an endpoint witness row. Hence a $\mathsf{ProximinalSet}^{\uparrow}$, $\mathsf{MetricProjection}^{\uparrow}$, compactness, or projection consumer that needs a nearest point or projection endpoint must provide that endpoint evidence outside the $\mathsf{LocatedSet}^{\uparrow}$ packet. □

Theorem 1541.6 (LocatedSet metric-projection carrier admission)**.** *Every* $\mathsf{MetricProjection}^{\uparrow}$ *carrier that consumes an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet must first display the parent* $\mathsf{Met}^{\uparrow}$ *row* X*, the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row* A*, the located-distance ledger* Q*, the rational windows* R*, the* $\mathsf{Real}^{\uparrow}$ *distance seal* E*, the* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff* T*, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows* H, C, P, N*. The admission route may then add its own projection-endpoint witness, but that witness is not a coordinate of the* $\mathsf{LocatedSet}^{\uparrow}$ *packet.*

Proof. Project the carrier of Definition 1541.1. It lists exactly the metric, subspace, located-distance, rational-window, real-seal, located-metric, transport, replay, provenance, and local naming rows.

Use Definition 1541.2 to bind these rows as the local certificate surface. A projection carrier can read set admission only through A and distance data only through Q, R, E, T.

Finally apply Definition 1541.5. Since the located-set packet contains no endpoint witness row, metric-projection carrier admission must add any projection endpoint outside the displayed route. □

Theorem 1541.7 (LocatedSet distance-witness obligation)**.** *Let* $L = (X, A, Q, R, E, T, H, C, P, N)$ *be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Any downstream located-metric, projection, or proximinal consumer that cites a distance witness for* L *must read it through*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricSubspace}^{\uparrow} \longmapsto \mathsf{LocatedMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the visible BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows retained. No consumer may replace the displayed located-distance ledger* Q*, rational windows* R*, or real seal* E *by a selected nearest point.*

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetCarrier distance witness obligation`

Proof. Start with the distance handoff of Definition 1541.3. It forces every distance read through X, A, Q, R, E before any located-metric, completion, compactness, or total-boundedness consumer sees the result.

Apply the Real-distance seal exactness theorem, Definition 1541.4. The $\mathsf{Real}^{\uparrow}$ row is exact only as the seal of the parent metric, subspace admission, located-distance ledger, rational windows, and located-metric handoff.

The structural rows H, C, P, N preserve the same route by componentwise transport, continuation replay, provenance, and local naming. The nearest-point refusal boundary then excludes replacing this witness by a selected endpoint. □

Theorem 1541.8 (LocatedSet distance-window monotonicity)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted $\mathsf{LocatedSet}^{\uparrow}$ packet. If a displayed positive rational tolerance request for L is refined to a smaller positive rational tolerance, then the corresponding rational distance windows remain monotone inside the same route*

$$X, A, Q, R, E, T, H, C, P, N.$$

The refinement may shrink the visible lower and upper rational comparison windows in R, but it cannot change the parent metric row, the $\mathsf{MetricSubspace}^{\uparrow}$ admission row, the located-distance ledger, the $\mathsf{Real}^{\uparrow}$ seal, or the structural transport rows.

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetCarrier distance window monotonicity`

Proof. Project the carrier in Definition 1541.1. The only coordinates available to a tolerance request are the parent metric row X, admission row A, located-distance ledger Q, rational windows R, real seal E, located-metric handoff T, and structural rows H, C, P, N.

Use Definition 1541.2. Positive tolerance handling is part of the located-distance stability and rational-window readback surface, so a refinement must be recorded in the same displayed rows rather than in a detached metric object.

Apply Definition 1541.3. It keeps every distance read downstream of X, A, Q, R, E, and Definition 1541.4 keeps the $\mathsf{Real}^{\uparrow}$ readback sealed to the same rational windows. Hence refinement changes only the window bounds displayed in R, preserving monotonicity of the accepted route. □

Theorem 1541.9 (LocatedSet positive-tolerance readback)**.** *For an accepted $\mathsf{LocatedSet}^{\uparrow}$ packet $L = (X, A, Q, R, E, T, H, C, P, N)$, every $\mathsf{Real}^{\uparrow}$ distance readback from L factors through a positive rational tolerance request in Q and a displayed rational-window read in R. Thus a consumer may read the Real distance seal E only after the route*

$$X \longmapsto A \longmapsto Q \longmapsto R$$

has supplied the parent metric observations, subspace admission data, positive tolerance evidence, and rational comparison windows.

Proof. Start from Definition 1541.1. The row Q is the located-distance ledger for positive rational tolerance requests, and R is the rational distance row packaging the lower and upper comparison windows supplied by Q.

Use Definition 1541.2. It makes located-distance stability, rational-window readback, and $\mathsf{Real}^{\uparrow}$ seal handoff local obligations of the same $\mathsf{NameCert}_{\mathsf{LocatedSet}^{\uparrow}}$ row.

Now apply Definition 1541.3 and Definition 1541.4. Together they force every Real-distance read to pass through X, A, Q, R, E in that order, so E cannot be consumed before a positive tolerance request in Q and a rational-window read in R have been displayed. □

Theorem 1541.10 (LocatedSet subspace-carrier transport)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Carrier transport along the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row preserves the full located-distance route*

$$X, A, Q, R, E, T, H, C, P, N.$$

In particular, an admitted subspace presentation may transport the parent metric row and membership evidence through H and replay it through C, but it may not detach Q, R, E, T, P, N from the same carrier route.

Proof. Project Definition 1541.1. The admission row A is displayed as part of the same finite packet as the parent metric row, located-distance ledger, rational windows, real seal, located-metric handoff, transport, replay, provenance, and local name.

Use Definition 1541.2. Parent metric compatibility and $\mathsf{MetricSubspace}^{\uparrow}$ admission stability are local obligations, and componentwise hsame transport is carried by H for every analytic and structural row.

Finally apply Definition 1541.3 and Definition 1541.4. The distance and Real-readback routes remain $X, A, Q, R, E, T, H, C, P, N$, so carrier transport along admission preserves the located-distance route and supplies no nearest-point choice, quotient subset equality, ambient completion, or full real-completeness principle. □

Theorem 1541.11 (LocatedSet open-ball distance window)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Any open-ball, proximinal, or projection consumer that reads a located-set distance window from L must first pass through the parent* $\mathsf{Met}^{\uparrow}$ *row of chapter 57, the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission route of chapter 1519, the* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff of chapter 1533, the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the displayed* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows H, C, P, N.*

The window therefore exposes only the route $X, A, Q, R, E, T, H, C, P, N$; it does not supply an open-ball center selector, nearest-point endpoint, projection endpoint, quotient subset equality, ambient completed metric space, or full $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the distance handoff of Definition 1541.3. It orders every distance read through the parent metric row X, the subspace admission row A, the located-distance ledger Q, the rational window row R, and the Real seal E.

Apply the Real-distance seal exactness theorem, Definition 1541.4. It identifies the sibling route through chapter 57, chapter 1519, chapter 1533, and chapter 38 as the only route by which a Real distance read becomes available.

Now use Definition 1541.7. A downstream open-ball, proximinal, or projection consumer that cites the distance window must retain the visible BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows while reading the same witness.

The nearest-point refusal boundary and metric-projection carrier admission then keep endpoint evidence outside the $\mathsf{LocatedSet}^{\uparrow}$ packet. Thus the open-ball distance window is a routed finite witness, not a selected center, nearest point, projection endpoint, quotient subset, completed metric space, or full real-completeness row. □

Theorem 1541.12 (LocatedSet proximinal consumer nonescape)**.** *Every* $\mathsf{ProximinalSet}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet must expose the carrier route $X, A, Q, R, E, T, H, C, P, N$ before using the packet as located-set input. If the proximinal consumer supplies a nearest-point or projection endpoint, that endpoint is an additional consumer row; it cannot be read from the* $\mathsf{LocatedSet}^{\uparrow}$ *packet itself and cannot bypass* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *or* $\mathsf{NameCert}$.

Proof. Use Definition 1541.2. It identifies the finite obligation surface that every accepted $\mathsf{LocatedSet}^{\uparrow}$ packet displays.

Then use Definition 1541.7. A proximinal consumer that needs a distance witness reads the same metric, subspace, located-metric, real-seal, and structural rows.

Finally apply Definition 1541.5 and Definition 1541.6. Endpoint or projection evidence is a consumer-supplied row outside the located-set carrier, so the proximinal route cannot escape the displayed packet or import open-cover compactness. □

Theorem 1541.13 (LocatedSet parent-subspace admission)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Every consumer that admits L as a located subset must first read X as the parent* $\mathsf{Met}^{\uparrow}$ *row and A as the displayed* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row before any located-distance, real-seal, or proximinal route is public. The admission surface is preserved only through the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N.*

Proof. Project the carrier of Definition 1541.1. The first two analytic rows are exactly the parent metric row X and the subspace admission row A.

Use Definition 1541.2. Parent metric compatibility and $\mathsf{MetricSubspace}^{\uparrow}$ admission stability are local NameCert obligations, so a consumer cannot begin with Q, R, E, T as detached located-distance data.

Finally apply Definition 1541.12. Even a proximinal consumer must expose the full route $X, A, Q, R, E, T, H, C, P, N$ before reading the packet as located-set input. Thus parent and admission rows remain public prerequisites, with H, C, P, N preserving the same route. □

Theorem 1541.14 (LocatedSet carrier admission)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Carrier admission for L is exhausted by the displayed* $\mathsf{Met}^{\uparrow}$ *row X,* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row A,* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff row T,* $\mathsf{Real}^{\uparrow}$ *seal E,* BHist *transport row H, componentwise* hsame *data,* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N. The admission route does not add nearest-point selection, open-cover compactness, quotient subset equality, ambient completion, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1541.1. The tuple lists exactly the parent metric, subspace admission, located-distance ledger, rational-window, Real-seal, located-metric handoff, transport, replay, provenance, and local naming rows.

Apply Definition 1541.2. It identifies carrier admission, parent metric compatibility, $\mathsf{MetricSubspace}^{\uparrow}$ admission stability, located-distance stability, rational-window readback, $\mathsf{Real}^{\uparrow}$ seal handoff, located-metric consumer compatibility, and ledger non-escape as the local NameCert obligations.

Finally use Definition 1541.3 and Definition 1541.13. They keep every consumer downstream of $X, A, Q, R, E, T, H, C, P, N$, so admission exposes only the displayed finite route and none of the excluded endpoint, compactness, quotient, completion, or full-completeness principles. □

Theorem 1541.15 (LocatedSet classifier stability)**.** *For accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packets, classifier stability is computed only from the displayed rows $X, A, Q, R, E, T, H, C, P, N$ over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricSubspace}^{\uparrow}$*,* $\mathsf{LocatedMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. If those rows are transported componentwise by the same* hsame *data and replayed by the same* Cont *route, the located-set classifier remains stable; it cannot be changed by a hidden nearest-point selector, open-cover compactness proof, quotient subset equality, ambient completion, or full Real-completeness row.*

Proof. Read the classifier clause in Definition 1541.1. It compares located-set packets through the displayed tuple and has no coordinate for endpoint selection, open covers, quotient subset equality, completion, or full Real completeness.

Use Definition 1541.2. The stability obligations for parent metric compatibility, subspace admission, located-distance data, rational windows, Real sealing, and located-metric handoff are all local obligations of the same NameCert row.

The transport row H and replay row C preserve those rows componentwise through hsame and Cont. Therefore any accepted classifier read remains the same finite BEDC route through $X, A, Q, R, E, T, H, C, P, N$ and cannot be altered by data outside the carrier. □

Theorem 1541.16 (LocatedSet distance exactness)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Every distance read from L is exact as the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricSubspace}^{\uparrow} \longmapsto Q \longmapsto R \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{LocatedMetric}^{\uparrow}$$

with BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows retained. The exactness route exposes no nearest-point selection, open-cover compactness, quotient subset equality, ambient completion, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1541.3. It forces each ambient-point distance read through X, A, Q, R, E before any located-metric, completion, compactness, or total-boundedness consumer sees the result.

Apply Definition 1541.4. The $\mathsf{Real}^{\uparrow}$ read is exact only as the seal of the parent metric row, subspace admission row, located-distance ledger, rational windows, and located-metric handoff.

Use Definition 1541.5. Endpoint evidence is not a carrier coordinate, and the same refusal excludes open-cover compactness, quotient subset equality, ambient completion, and full Real completeness from the distance route. □

Theorem 1541.17 (LocatedSet ledger nonescape)**.** *Every* $\mathsf{LocatedSet}^{\uparrow}$ *root-obligation ledger read that cites* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricSubspace}^{\uparrow}$*,* $\mathsf{LocatedMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *remains inside the accepted carrier route $X, A, Q, R, E, T, H, C, P, N$. A downstream projection, proximinal, completion, compactness, or Real-distance consumer may add its own endpoint or completion evidence, but it cannot read nearest-point selection, open-cover compactness, quotient subset equality, ambient completion, or full Real completeness from the* $\mathsf{LocatedSet}^{\uparrow}$ *ledger itself.*

Proof. Start with Definition 1541.14. It fixes the admitted carrier rows and the local NameCert obligations inside $X, A, Q, R, E, T, H, C, P, N$.

Use Definition 1541.15. Componentwise hsame transport and Cont replay preserve the same classifier-facing rows, so the ledger cannot introduce a hidden classifier coordinate.

Finally apply Definition 1541.16 together with Definition 1541.12. Distance and projection consumers must keep the displayed route public before adding their own evidence, so the located-set ledger cannot escape to selected endpoints, open-cover compactness, quotient subset equality, ambient completion, or full Real completeness. □

Theorem 1541.18 (LocatedSet distance-ledger totality)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Every admitted ambient-point distance request handled by L is accounted for by the finite route*

$$X \longmapsto A \longmapsto Q \longmapsto R \longmapsto E \longmapsto T \longmapsto H, C, P, N.$$

No accepted request may skip the located-distance ledger Q, replace the rational-window row R, or read the $\mathsf{Real}^{\uparrow}$ *seal E before the parent metric and subspace admission rows have been displayed.*

Lean4 ✓ `BEDC.Derived.LocatedSetUp.LocatedSetDistanceLedger totality`

Proof. Start with Definition 1541.3. It orders every distance read through X, A, Q, R, E before any located-metric, completion, compactness, or total-boundedness consumer can use the result.

Apply Definition 1541.4. The real distance seal is exact only as the seal of the displayed parent metric, admission, located-distance, rational-window, and located-metric route.

The totality claim is therefore local to accepted requests: once a request is admitted by X and A, its comparison evidence is in Q, its rational windows are in R, and its real seal is in E. The rows T, H, C, P, N expose this same ledger to consumers without adding nearest-point selection, open-cover compactness, or hidden completion data. □

Definition 1541.19 (LocatedSet root obligation carrier)**.** *The root-obligation carrier of an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet* $L = (X, A, Q, R, E, T, H, C, P, N)$ *is the three-row public interface*

$$\mathsf{Root}_{\mathsf{LocatedSet}^{\uparrow}}(L) = (D, J, S)$$

where:

1. *D is the distance-witness row that reads the parent* $\mathsf{Met}^{\uparrow}$ *row X, the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row A, the located-distance ledger Q, the rational windows R, the* $\mathsf{Real}^{\uparrow}$ *seal E, and the* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff T.*
2. *J is the projection-admission row. It records that any* $\mathsf{ProximinalSet}^{\uparrow}$ *or* $\mathsf{MetricProjection}^{\uparrow}$ *consumer must add its own endpoint or projection witness after reading $X, A, Q, R, E, T, H, C, P, N$.*
3. *S is the Real-metric nonescape row. It records that every Real-distance or metric-projection read stays inside the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *route.*

The root carrier has no coordinate for selected nearest points, quotient subset equality, ambient completion, open-cover compactness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1541.20 (LocatedSet root distance witness)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet and let* $\mathsf{Root}_{\mathsf{LocatedSet}^{\uparrow}}(L) = (D, J, S)$ *be its root-obligation carrier. The row D is total for accepted distance-witness requests: every downstream* $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{ProximinalSet}^{\uparrow}$, *or* $\mathsf{MetricProjection}^{\uparrow}$ *consumer that cites a distance witness from L must read $X, A, Q, R, E, T, H, C, P, N$ and cannot replace Q, R, E by an endpoint selector.*

Proof. Start from Definition 1541.18. It accounts for each admitted ambient-point distance request by the route $X, A, Q, R, E, T, H, C, P, N$.

The row D of Definition 1541.19 is exactly this route presented as the root distance-witness obligation. It does not introduce a second distance object or a detached witness.

Apply Definition 1541.7. A downstream located-metric, proximinal, or projection consumer must retain the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows while reading the same $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Finally use Definition 1541.5. Since no row of L is an endpoint witness row, the distance witness in D cannot be replaced by selected nearest-point data. □

Theorem 1541.21 (LocatedSet root projection obligation)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet with root-obligation carrier (D, J, S). Any* $\mathsf{ProximinalSet}^{\uparrow}$ *or* $\mathsf{MetricProjection}^{\uparrow}$ *consumer admitted over L must first expose the located-set route $X, A, Q, R, E, T, H, C, P, N$ and then supply any endpoint, projection, or proximinal witness as a consumer row outside the* $\mathsf{LocatedSet}^{\uparrow}$ *packet.*

Proof. Read the projection-admission row J from Definition 1541.19. Its only source data are the displayed rows of the accepted located-set packet and the statement that endpoint evidence is added by the consumer.

Apply Definition 1541.6. A metric-projection carrier may consume L only after displaying $X, A, Q, R, E, T, H, C, P, N$.

Now apply Definition 1541.12. The same route is required for a proximinal consumer, and any nearest-point or projection endpoint remains outside the carrier.

Thus the projection obligation is a route-exhaustion statement: all available located-set evidence is read before the consumer endpoint row appears, and no projection row is smuggled back into L. □

Theorem 1541.22 (LocatedSet root Real-metric nonescape)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet with root-obligation carrier (D, J, S). Every consumer that reads a* $\mathsf{Real}^{\uparrow}$ *distance seal or a* $\mathsf{MetricProjection}^{\uparrow}$ *route from L remains inside the displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricSubspace}^{\uparrow}$*,* $\mathsf{LocatedMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it receives no host metric equality, quotient subset equality, selected endpoint, completed ambient metric space, or full real-completeness principle from the located-set packet.*

Proof. Read the nonescape row S from Definition 1541.19. It names the exact sibling route available to Real-distance and projection consumers.

Use Definition 1541.4. The $\mathsf{Real}^{\uparrow}$ distance read is exhausted by the parent metric row, subspace admission row, located-distance ledger, rational windows, Real seal, located-metric handoff, and structural rows.

Use Definition 1541.23. Completion, compactness, projection, and proximinal consumers must keep E downstream of X, A, Q, R and upstream of T, H, C, P, N.

Finally combine the root distance witness with the projection obligation. The first fixes the distance witness inside D, and the second keeps endpoint or projection evidence in the consumer row. Hence a Real-metric or projection read cannot escape to host equality, quotient subset equality, selected endpoint data, ambient completion, or full real completeness. □

Theorem 1541.23 (LocatedSet real-seal consumer boundary)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Any completion, compactness, projection, or proximinal consumer that reads the* $\mathsf{Real}^{\uparrow}$ *distance seal E must keep that seal downstream of the displayed route X, A, Q, R and upstream of the located-metric handoff T and structural rows H, C, P, N. The consumer receives no selected nearest point, open-cover compactness proof, quotient subset equality, completed ambient metric space, or full real-completeness row from E.*

Proof. Use Definition 1541.4. It identifies E as the seal of the parent metric, subspace admission, located-distance ledger, rational windows, and located-metric handoff.

Now use Definition 1541.5. No row of the carrier is an endpoint witness row, so a projection or proximinal consumer cannot reinterpret E as a selected nearest point.

Finally combine Definition 1541.7 with Definition 1541.12. The consumer boundary remains the same route through $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, and the excluded completion, compactness, quotient, and endpoint principles are not carrier coordinates. □

Falsifiable prediction: Every $\mathsf{LocatedSet}^{\uparrow}$ read from an accepted packet factors through the parent $\mathsf{Met}^{\uparrow}$ row, $\mathsf{MetricSubspace}^{\uparrow}$ admission row, located-distance ledger, rational comparison

windows, $\mathsf{Real}^{\uparrow}$ distance seal, located-metric handoff, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any completion, compactness, or total-boundedness consumer reads the route.

Independence witness: metric, metricsubspace, locatedmetric, real: $\mathsf{LocatedSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries parent metric data, displayed subspace admission, a located-distance ledger, rational windows, a RealUp distance seal, located-metric handoff, transport, continuation replay, provenance, and local naming.

Theorem 1541.24 (LocatedSet root distance obligation)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted $\mathsf{LocatedSet}^{\uparrow}$ packet. Every root distance consumer of L factors by projection through the displayed $\mathsf{Met}^{\uparrow}$ row X, $\mathsf{MetricSubspace}^{\uparrow}$ admission row A, $\mathsf{LocatedMetric}^{\uparrow}$ distance ledger Q, rational comparison windows R, terminal $\mathsf{Real}^{\uparrow}$ distance seal E, located-metric handoff T, BHist transport row H, hsame identifications, Cont replay row C, Pkg provenance row P, and local $\mathsf{NameCert}$ row N.*

Proof. Project the accepted packet onto $X, A, Q, R, E, T, H, C, P, N$. These rows are the only coordinates displayed by the located-set carrier.

Use the root distance witness theorem together with the Real-distance seal exactness theorem. The distance read starts at the parent $\mathsf{Met}^{\uparrow}$ row, passes through the displayed $\mathsf{MetricSubspace}^{\uparrow}$ and $\mathsf{LocatedMetric}^{\uparrow}$ rows, and reaches $\mathsf{Real}^{\uparrow}$ only through Q, R, E.

Transport, replay, provenance, and local naming are then read from H, C, P, N, with the componentwise hsame data carried by the same packet. Since no other coordinate is available, every root distance consumer follows this route. □

Theorem 1541.25 (LocatedSet root locatedness window)**.** *For an accepted $\mathsf{LocatedSet}^{\uparrow}$ packet $L = (X, A, Q, R, E, T, H, C, P, N)$, every locatedness-window consumer is exhausted by the finite rational windows R sitting between the $\mathsf{LocatedMetric}^{\uparrow}$ ledger Q and the $\mathsf{Real}^{\uparrow}$ seal E. The consumer must also preserve the surrounding $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, and it receives no selected nearest point, open-cover compactness proof, quotient subset equality, completed ambient metric space, or full Real-completeness row.*

Proof. Begin with the rational-window rows displayed in the carrier. They are downstream of the located-distance ledger and upstream of the Real seal.

Apply the distance-witness obligation and the real-seal consumer boundary. They identify the only route by which a locatedness read can reach a Real-facing endpoint: $X, A, Q, R, E, T, H, C, P, N$.

The structural rows carry this finite route through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The excluded endpoint, compactness, quotient, completion, and full-completeness data are not rows of the packet. □

Theorem 1541.26 (LocatedSet root nonescape)**.** *Every root consumer of an accepted $\mathsf{LocatedSet}^{\uparrow}$ packet $L = (X, A, Q, R, E, T, H, C, P, N)$ that asks for distance data, locatedness windows, or a downstream root route remains inside the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. No root consumer obtains host metric equality, quotient subset equality, selected endpoint data, open-cover compactness, completed ambient metric space, or full $\mathsf{Real}^{\uparrow}$ completeness from L.*

Proof. Use the root distance obligation to account for distance reads. It fixes the route through $X, A, Q, R, E, T, H, C, P, N$.

Use the root locatedness-window theorem for rational-window reads. The same carrier rows are displayed, with R placed between the located-distance ledger and the Real seal.

Finally combine the existing projection obligation with the Real-metric nonescape theorem. Downstream root consumers may add their own endpoint or projection evidence, but that evidence is outside the $\mathsf{LocatedSet}^{\uparrow}$ carrier, so the listed host and completion objects cannot be exported by L. □

Definition 1541.27 (LocatedSet MetricSubspace exactness row)**.** *For an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet* $L = (X, A, Q, R, E, T, H, C, P, N)$*, the MetricSubspace exactness row is the finite subrow*

$$M_L = (X, A, Q, R, E, T, H, C, P, N)$$

read with X *as the parent* $\mathsf{Met}^{\uparrow}$ *row,* A *as the displayed* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row,* Q *as the located-distance ledger,* R *as the rational-window row,* E *as the* $\mathsf{Real}^{\uparrow}$ *distance seal, and* T, H, C, P, N *as the located-metric handoff, transport, replay, provenance, and local naming rows. It synchronizes the parent metric, subspace admission, and located-distance ledger before any consumer reads the Real-facing seal.*

Theorem 1541.28 (LocatedSet MetricSubspace exactness)**.** *Every accepted* $\mathsf{LocatedSet}^{\uparrow}$ *read preserves the MetricSubspace exactness row* M_L *through* $X, A, Q, R, E, T, H, C, P, N$*. In particular, the parent* $\mathsf{Met}^{\uparrow}$ *row,* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row, and located-distance ledger are read together before rational windows, the* $\mathsf{Real}^{\uparrow}$ *seal, located-metric handoff, transport, replay, provenance, and local naming are exposed to a consumer.*

Proof. Start with the accepted carrier projection. The definition of M_L uses exactly the displayed rows of the packet, so preserving the carrier rows preserves the exactness row.

Apply the root distance obligation to force distance reads through $X, A, Q, R, E, T, H, C, P, N$. It places the parent metric and subspace admission before the located-distance ledger reaches the rational windows and Real seal.

Use the root nonescape theorem for the consumer side. Since no host metric equality, quotient subset equality, or selected endpoint coordinate is present, a consumer cannot detach Q from the same X, A source route. The read therefore preserves M_L. □

Theorem 1541.29 (LocatedSet LocatedMetric readback coverage)**.** *Every* $\mathsf{LocatedMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet exhausts the exactness row* M_L *before reading the* $\mathsf{Real}^{\uparrow}$ *seal* E*. The consumer receives the located-distance ledger* Q *only after the parent metric row* X *and MetricSubspace admission row* A*, and it reaches* E *only through the rational-window row* R *and the displayed handoff route* T, H, C, P, N*.*

Proof. Read the LocatedMetric-facing route from the carrier. The handoff row T is attached to the same tuple as $X, A, Q, R, E, H, C, P, N$, so it cannot expose an independent located-metric ledger.

Apply the locatedness-window theorem. It exhausts R between Q and E, while the root distance obligation keeps Q downstream of X, A.

Finally use the exactness theorem. The consumer can read the Real seal only after the synchronized row M_L has supplied the parent metric, subspace admission, ledger, windows, handoff, transport, replay, provenance, and local naming data. □

Theorem 1541.30 (LocatedSet Real-seal window monotonicity)**.** *If an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet refines its rational-window row* R *along the displayed located-distance route, the* $\mathsf{Real}^{\uparrow}$ *seal route remains monotone over the same exactness row* M_L*. The refinement may shrink or*

strengthen the rational windows, but it cannot change the parent metric row, MetricSubspace admission, located-distance ledger, located-metric handoff, transport, replay, provenance, or local NameCert row used to read the Real seal.

Proof. Use the locatedness-window theorem to locate the refinement inside R. The surrounding rows $X, A, Q, E, T, H, C, P, N$ remain the rows of the same accepted packet.

Apply MetricSubspace exactness. It synchronizes X, A, Q before the rational windows and keeps E downstream of that same route.

The Real-seal consumer boundary and root nonescape theorem rule out changing the route by importing completion, compactness, quotient equality, or selected endpoint data. Thus refinement of R is monotone for the Real-seal route over M_L. □

Theorem 1541.31 (LocatedSet distance-window totality obligation)**.** *Let $L = (X, A, Q, R, E, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet. Every accepted distance-window request handled by L is total over the displayed route*

$$X, A, Q, R, E, T, H, C, P, N.$$

That is, the request first reads the parent $\mathsf{Met}^{\uparrow}$ *row X, the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row A, and the located-distance ledger Q; then it reads the rational windows R, the* $\mathsf{Real}^{\uparrow}$ *distance seal E, the located-metric handoff T, the* BHist *transport row H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}$ *row N. No accepted distance-window request may replace those rows by a selected endpoint, a quotient-subset equality, an open-cover compactness proof, an ambient completion, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the distance-ledger totality theorem. It accounts for every admitted ambient-point distance request by the finite route $X, A, Q, R, E, T, H, C, P, N$.

Use the locatedness-window theorem and the readback coverage theorem. They place the rational-window row R between the located-distance ledger Q and the Real seal E, with the same parent metric and subspace admission rows upstream.

Finally apply MetricSubspace exactness and root nonescape. The exactness row keeps X, A, Q synchronized before any window is read, while nonescape excludes endpoint choice, quotient equality, compactness, completion, and full real-completeness data from the request. □

Theorem 1541.32 (LocatedSet projection-proximinal obligation package)**.** *For an accepted* $\mathsf{LocatedSet}^{\uparrow}$ *packet $L = (X, A, Q, R, E, T, H, C, P, N)$, projection and proximinal admission is a consumer-supplied endpoint row outside* $\mathsf{LocatedSet}^{\uparrow}$*. A* $\mathsf{MetricProjection}^{\uparrow}$ *or* $\mathsf{ProximinalSet}^{\uparrow}$ *consumer may cite the route $X, A, Q, R, E, T, H, C, P, N$, but it must package any projection point, nearest-point witness, or proximinal endpoint as its own row rather than as a coordinate of L.*

Proof. Read the root projection obligation. It requires every projection or proximinal consumer to expose the full located-set route before any endpoint-facing row appears.

Apply metric-projection carrier admission and proximinal nonescape. These theorems permit the consumer to reuse $X, A, Q, R, E, T, H, C, P, N$, but they keep projection endpoints and nearest-point witnesses outside the located-set carrier.

The nearest-point refusal boundary supplies the final exclusion. Since no row of L stores an endpoint witness, the projection or proximinal package is admitted only as consumer evidence beyond $\mathsf{LocatedSet}^{\uparrow}$. □

Theorem 1541.33 (LocatedSet obligation consumer nonescape assembly)**.** *The carrier row of Definition 1541.1, the distance handoff of Definition 1541.3, the* $\mathsf{Real}^{\uparrow}$ *seal discipline of Definition 1541.4, the projection obligation of Definition 1541.32, and the nonescape rows of Definition 1541.26 assemble an obligation-closure surface for* $\mathsf{LocatedSet}^{\uparrow}$*. Every downstream metric projection, proximinal, compactness, completion, or total-boundedness consumer receives only the displayed rows* $X, A, Q, R, E, T, H, C, P, N$ *unless it supplies its own endpoint, compactness, completion, or bridge row.*

Proof. Project the carrier first. The displayed rows $X, A, Q, R, E, T, H, C, P, N$ are the entire located-set interface available to a consumer.

Use distance-window totality and distance handoff. They force distance reads through the parent metric row, subspace admission, located-distance ledger, rational windows, Real seal, and located-metric handoff.

Use the projection-proximinal obligation package for endpoint-facing consumers. It allows later projection and proximinal packets to add their own endpoint rows, while keeping those rows outside $\mathsf{LocatedSet}^{\uparrow}$.

Finally apply root nonescape and Real-seal boundary results. They exclude host equality, quotient subset equality, nearest-point choice, open-cover compactness, ambient completion, and full $\mathsf{Real}^{\uparrow}$ completeness from the located-set packet itself, so the assembled surface is an obligation-closure interface rather than a stronger compactness or choice principle. □

Theorem 1541.34 (LocatedSet public export)**.** *The public* $\mathsf{LocatedSet}^{\uparrow}$ *export surface consists of the carrier row of Definition 1541.1, the distance handoff of Definition 1541.3, the* $\mathsf{Real}^{\uparrow}$ *distance seal exactness of Definition 1541.4, the MetricSubspace synchronization of Definition 1541.28, the distance-window totality of Definition 1541.31, the projection and proximinal obligation package of Definition 1541.32, and the root nonescape assembly of Definition 1541.33. A downstream public consumer may cite those rows as the located-set interface, but it receives no endpoint selector, quotient subset equality, open-cover compactness, ambient completion, or full* $\mathsf{Real}^{\uparrow}$ *completeness unless it supplies a separate consumer packet.*

Proof. Project the carrier first. The public object exposes the displayed route $X, A, Q, R, E, T, H, C, P, N$ and no hidden row.

Use the distance handoff, Real-seal exactness, MetricSubspace exactness, and distance-window totality theorems to identify the public distance interface. Together they force every accepted distance read through the parent metric row, subspace admission row, located-distance ledger, rational windows, Real seal, located-metric handoff, transport, replay, provenance, and local naming.

Use the projection-proximinal obligation package for endpoint-facing consumers. It lets projection and proximinal chapters cite the located-set route while keeping nearest-point or projection witnesses in their own packets.

Finally apply the root nonescape assembly. Since the excluded compactness, completion, quotient, endpoint-selection, and Real-completeness data have no coordinate in $\mathsf{LocatedSet}^{\uparrow}$, the listed rows form the public export surface and no stronger interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedSet}^{\uparrow}$ packet starts from a displayed Metric row, filters candidate set members through a MetricSubspace admission row, records finite located-distance comparisons, packages rational windows, seals a RealUp distance name, and preserves the same route by located-metric handoff, transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSet}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1541.1, Definition 1541.2, Definition 1541.3, Definition 1541.4, Definition 1541.28, Definition 1541.31, Definition 1541.32, and Definition 1541.33 close the carrier, NameCert obligations, distance handoff, Real-seal exactness, MetricSubspace exactness, distance-window totality, projection-proximinal package, and obligation consumer nonescape assembly.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedSet}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.LocatedSetUp.LocatedSetCarrier\_distance\_witness\_obligation`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected nearest points, quotient subset equality, completed ambient metric spaces, full Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Bridge checking can upgrade the verification axis once a standard bridge theorem is available; the theory axis records the displayed located-set package and downstream examples.

Chapter 1542

A Concrete Naming Certificate for $\mathsf{BaireCategory}^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the $\mathsf{BaireCategory}^{\uparrow}$ packet records the finite certificate surface for complete-metric dense-open intersection data. It depends on the prefix interface of chapter 334, the limit-witness row of chapter 128, the distance and ball classifier of chapter 57, the finite schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter names the finite BEDC handoff for displayed dense-open rows; it does not assert the full classical Baire category theorem as a host theorem.

The dense-open schedule is the sibling row consumed by chapter 1843 and chapter 2267: $\mathsf{CompactBaire}^{\uparrow}$ restricts the $\mathsf{BaireCategory}^{\uparrow}$ row to a compact-metric source, while $\mathsf{MeagreSet}^{\uparrow}$ hands complementary dense-open rows to the same $\mathsf{BaireCategory}^{\uparrow}$ packet.

The same dense-open schedule is consumed through the Baire-space source of chapter 334 and the complete-metric Cauchy-thread row of chapter 128.

Definition 1542.1 (BaireCategory carrier)**.** *A* $\mathsf{BaireCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, M, D, O, R, T, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet supplying the finite stream window through which dense-open requests are named.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row supplying the metric carrier, ball classifier, displayed Cauchy modulus, and limit-witness route used by the intersection construction.*
3. *D is the finite dense-open schedule. Each listed row contains an open-ball admission datum, a density witness for the displayed predecessor ball, and the regular-rational tolerance read needed to shrink the next ball.*
4. *O is the open-row ledger recording the displayed ball inclusions and the finite meet of the dense-open rows in D.*
5. *R is the refinement row. It records the nested-ball shrinkage, positive-radius comparisons, and metric-distance witnesses that keep each next ball inside the previous ball and inside the next dense-open row.*
6. *T is the Cauchy thread produced by the refinement row. It carries the finite list of centers, radii, metric comparison rows, and the terminal* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff for the displayed limit read.*

7. *H records componentwise* hsame *transport for B, M, D, O, R, T and the structural rows.*
8. *C records displayed* Cont *replay of the prefix schedule, metric ball reads, dense-open admissions, refinement route, Cauchy thread, and terminal real-name handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, a host set intersection, open-cover compactness, ambient completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BaireCategoryUp.BaireCategoryUp`

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryTasteGate single carrier alignment`

Theorem 1542.2 (BaireCategory complete-metric obligation surface)**.** *For an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier $Q = (B, M, D, O, R, T, H, C, P, N)$, the complete-metric obligation surface consists of the following displayed BEDC rows:*

1. *the complete metric carrier row M, including the metric carrier, ball classifier, Cauchy-modulus route, and limit-witness handoff;*
2. *the nested dense-open request row D, O, R, where each listed dense-open request is read only after its predecessor ball is displayed and its successor ball is recorded by the refinement ledger;*
3. *the meagre-set exclusion ledger, consumed only as a negative boundary saying that complementary meagre rows cannot be exported as a positive dense-open intersection object;*
4. *the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *finite observation row B, T, recording the prefix schedule, finite center list, regular-rational tolerance reads, and terminal real-name handoff;*
5. *the* $\mathsf{Real}^{\uparrow}$*-valued radius budget, recording positive-radius comparisons and distance bounds inside the same metric source;*
6. *the local* $\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}$ *provenance row P, N together with componentwise transport H and replay C.*

No row in this surface exports open-cover compactness, a countable-choice selector, or a host-level intersection theorem.

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCompleteMetricObligationSurface`

Proof. Project the BaireCategory carrier. The complete metric carrier is the row M, and every dense-open request is routed through the same metric ball classifier before it can be refined.

Route each dense-open request through the displayed complete metric and radius rows. The schedule D supplies the request, O records the predecessor ball, R records the successor ball and positive-radius comparison, and T stores the finite center and tolerance observations used for the Cauchy thread.

The meagre-set ledger is used only as a negative boundary. It can say which complementary route is excluded from the displayed dense-open schedule, but it does not create a positive intersection object or a selector for all dense-open rows.

Finally, read the structural rows. The $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ observations and $\mathsf{Real}^{\uparrow}$ radius budget are carried by B, T, R, while H, C, P, N transport, replay, source, and name exactly that finite complete-metric surface. Since no carrier coordinate stores open-cover compactness or choice, no such principle is exported. □

Theorem 1542.3 (BaireCategory NameCert obligations)**.** *For an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier* $Q = (B, M, D, O, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the Baire-space prefix packet, complete-metric row, dense-open finite schedule, open-row ledger, refinement row, Cauchy thread, transport, replay, provenance, and local naming row;*
2. *dense-open classifier exposure, requiring every displayed dense-open row to present an open-ball admission datum and a density witness over the current predecessor ball;*
3. *refinement stability, requiring the nested-ball shrinkage, positive radius comparison, and metric-distance rows to transport componentwise by* hsame *and replay by* Cont*;*
4. *Cauchy-thread exactness, requiring the terminal* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *read to pass through the displayed center list, radius ledger, and complete-metric limit-witness route in* M*;*
5. *ledger non-escape, stating that no countable-choice selector, host set intersection, open-cover compactness theorem, ambient-completeness theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1542.1. The tuple lists exactly the prefix, metric, dense-open, open-ledger, refinement, Cauchy-thread, transport, replay, provenance, and local naming rows.

The dense-open classifier exposure is read from D and O. Each displayed row in the finite schedule supplies an open-ball admission and a density witness for the predecessor ball already present in the ledger.

Refinement stability is componentwise. The row R stores the nested-ball shrinkage, radius comparison, and metric-distance witnesses, while H and C transport and replay those rows without changing the carrier shape.

Cauchy-thread exactness is then forced by T and M. The thread lists the centers and radii generated by the finite refinement route, and the complete-metric row supplies the displayed limit-witness route consumed by the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff.

No remaining packet coordinate names a selector over all dense-open rows, a host intersection object, an open-cover compactness principle, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$. The five rows therefore exhaust the local NameCert surface. □

Theorem 1542.4 (BaireCategory finite meet density handoff)**.** *Let* $Q = (B, M, D, O, R, T, H, C, P, N)$ *be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. The finite meet of the dense-open rows displayed by* D *routes through the complete-metric Cauchy thread* T*: each successive density witness refines the current displayed ball, the row* R *records the nested shrinkage and positive radius comparison, and the thread* T *presents the terminal regular-rational and real-name handoff through* M*.*

This handoff uses only the listed finite rows. It does not use countable choice, a host set intersection, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier finite meet density handoff`
↪variant `BEDC.Derived.BaireCategoryUp.BaireCategory finite meet density handoff`

Proof. Start with the finite schedule D. Its first dense-open row is applied to the initial displayed ball supplied by the metric and prefix data, producing an open subball and a density witness recorded in O.

For the next listed row, use the predecessor ball supplied by the previous refinement step. The refinement row R records the new center, the smaller positive radius, the inclusion into the predecessor ball, and the metric-distance witnesses showing that the new ball lies inside the next dense-open row.

Iterating over the displayed finite list yields the thread T. The thread is not a choice sequence: every center, radius, inclusion row, and comparison row appears in the finite packet, and each step is replayed by the displayed Cont data.

The complete-metric row M is then read only along this thread. It supplies the Cauchy limit-witness route for the displayed centers and radii, after which the terminal read is handed to the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

Since the construction consumes a finite list of dense-open rows and records each refinement explicitly, there is no call to countable choice. Since the carrier exposes only ball, refinement, thread, and handoff rows, it also has no host intersection object, open-cover compactness principle, ambient completeness theorem, or $\mathsf{Real}^{\uparrow}$ completeness theorem available for the handoff. □

Theorem 1542.5 (BaireCategory complete-metric dense thread)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. The complete-metric dense-thread obligation is the finite route that starts with the complete-metric row M, reads each displayed dense-open request through D and O, records the nested refinement in R, and assembles the Cauchy thread T before any terminal* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *read is available.*

The route exports no countable-choice selector, no host intersection object, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row. Every accepted dense-thread read remains inside $B, M, D, O, R, T, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier complete metric dense thread`

Proof. Project the carrier of Definition 1542.1. The complete-metric limit witness is held in M, while dense-open admission is held in the finite schedule D and the open ledger O.

Apply the finite meet density handoff, Definition 1542.4. It supplies each successor ball by reading the predecessor ball from O, the dense-open witness from D, and the nested-ball data from R.

The thread row T is then the displayed Cauchy thread assembled from those refinements. Since the structural rows H, C, P, N only transport, replay, source, and name the same finite packet, no selector over an external dense-open family or host intersection object is introduced.

Finally consume M along the thread. The complete-metric row supplies the limit-witness handoff only for the displayed centers and radii, so the terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ read has no coordinate for countable choice, host intersection, or ambient real completeness. □

Theorem 1542.6 (BaireCategory Cauchy thread limit handoff)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Once the refinement row R has assembled the thread row T, every terminal limit read factors through the* $\mathsf{CompleteMetric}^{\uparrow}$ *witness in M and then through the recorded* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff inside T. The handoff cannot replace the thread by a host intersection point, a countable-choice selection of centers, or a separate completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier cauchy thread limit handoff`

Proof. Start with complete-metric thread extraction, Definition 1542.14. It identifies T as the finite thread assembled from D, O, R and transported by H, C, P, N.

The complete-metric row M is consumed only after that thread is present. Its limit witness applies to the displayed centers, radii, and metric comparison rows recorded in T, so the limit read remains attached to the accepted packet.

The regular-rational and real-name handoff is likewise part of the thread row. The carrier has no separate coordinate for a host intersection point, a chosen sequence outside the finite schedule, or completeness of $\mathsf{Real}^{\uparrow}$ as an ambient theorem.

Therefore the terminal read factors through M and T exactly, with D, O, R supplying the dense-open and refinement history needed for the handoff. □

Theorem 1542.7 (BaireCategory ProbeBundle refinement induction)**.** *For an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier* $Q = (B, M, D, O, R, T, H, C, P, N)$*, every finite ProbeBundle-style refinement of the dense-open schedule is handled by induction over the displayed schedule row* D*. At the base stage the predecessor ball is read from the prefix and metric rows* B, M*; at each successor stage the dense-open row in* D*, the open ledger* O*, and the refinement row* R *determine the next recorded ball and extend the Cauchy thread* T*.*

The induction exports no countable-choice selector, host intersection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row;* H, C, P, N *only transport, replay, source, and name the same finite refinement packet.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier probe bundle refinement induction`

Proof. Proceed by induction on the finite schedule displayed in D. In the base case, the prefix row B and complete-metric row M supply the initial ball and metric classifier, so the thread has only the initial recorded center and radius data.

For the successor step, assume the previous predecessor ball and partial thread are already recorded. The next dense-open row in D is read against that predecessor ball, the open ledger O records its admission, and the refinement row R records the smaller successor ball together with the positive-radius and metric-distance witnesses.

Extend T by this displayed successor data. The extension is finite and recorded in the packet, so it is ProbeBundle-style replay of visible rows rather than a choice over an external family.

Finally use the structural rows H, C, P, N to transport and name the result. Since the carrier lacks coordinates for a selector, host intersection, or ambient real completeness theorem, the induction remains a finite refinement of the accepted $\mathsf{BaireCategory}^{\uparrow}$ packet. □

Theorem 1542.8 (BaireCategory ProbeBundle dense-open induction)**.** *Let* $Q = (B, M, D, O, R, T, H, C, P, N)$ *be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. The finite dense-open schedule in* D *is consumed by induction over the displayed ProbeBundle schedule: the base read uses the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row* B *and the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *ball classifier in* M*, while each successor read consumes the next dense-open row through* O *and* R *before extending the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *thread* T*.*

The induction is exactly the carrier, finite meet, and refinement-thread surface of Definition 1542.1, Definition 1542.4, and Definition 1542.7. It does not assert the full classical Baire theorem, countable choice, a host intersection point, open-cover compactness, ambient completeness, or a separate completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier probe bundle dense open induction`

Proof. Project the carrier of Definition 1542.1. The rows available for the schedule are $B, M, D, O, R, T, H, C, P, N$, with B supplying the $\mathsf{BaireSpace}^{\uparrow}$ prefix, M supplying the $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ source, and T holding the terminal $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff.

For the base stage, read the initial displayed ball from B and the metric ball classifier in M. No dense-open choice is made at this stage; the packet has only the displayed predecessor ball and its metric source.

For a successor stage, apply Definition 1542.4. The next dense-open entry of D is read against the predecessor ball in O, and the refinement row R records the smaller successor ball, positive-radius comparison, and metric-distance witnesses.

Finally apply Definition 1542.7. The displayed successor data extend the finite thread T, and the structural rows H, C, P, N transport, replay, source, and name the same packet. Since no carrier row stores countable choice, a host intersection point, open-cover compactness, ambient completeness, or a separate $\mathsf{Real}^{\uparrow}$ completeness theorem, the induction remains the finite dense-open ProbeBundle schedule claimed above. □

Theorem 1542.9 (BaireCategory choice-free refinement thread)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. Every finite dense-open schedule read by Q routes through displayed predecessor balls, positive-radius comparisons, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ tolerance reads, and the $\mathsf{CompleteMetric}^{\uparrow}$ limit witness. The route is the finite thread*

$$B, M, D, O, R, T, H, C, P, N,$$

so no countable-choice selector, host-level intersection object, meagre cover, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is available to the consumer.

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier refinement choice free`

Proof. Project the carrier of Definition 1542.1. The available rows are exactly $B, M, D, O, R, T, H, C, P, N$, with D holding the finite dense-open schedule and M holding the complete-metric limit witness.

Induct over the displayed ProbeBundle schedule in D. The base stage reads the predecessor ball from B, M. At a successor stage, the open ledger O displays the dense-open admission, the refinement row R records the successor ball and positive-radius comparison, and the thread row T records the $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ tolerance read.

At the terminal stage, consume M only through the Cauchy thread already recorded in T. The structural rows H, C transport and replay that same route, while P, N preserve provenance and local naming.

The carrier has no coordinate for a countable-choice selector, a host intersection, a meagre cover, open-cover compactness, or a separate ambient real-completeness theorem. Thus every accepted refinement read remains choice-free and packet-local. □

Theorem 1542.10 (BaireCategory refinement-schedule obligation)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. The dense-open schedule row D, open ledger O, refinement row R, and Cauchy thread T form the complete finite refinement surface of Q. Every complete-metric limit read and every terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff must pass through D, O, R, T before it can consume the $\mathsf{CompleteMetric}^{\uparrow}$ row M or the structural rows H, C, P, N.*

This surface refuses countable choice, host intersections, and ambient $\mathsf{Real}^{\uparrow}$ completeness: the schedule is a finite displayed row, the refinements are recorded predecessor-ball steps, and the terminal handoff is the thread row already present in the carrier.

Proof. Proceed by induction on the finite schedule row D. In the empty stage, the only available predecessor ball is the one supplied by the prefix and complete-metric rows B, M, and the thread T has no dense-open refinement beyond the initial displayed data.

At a successor stage, the next dense-open entry of D is read against the predecessor ball already recorded by O. The refinement row R records the successor ball, its positive-radius comparison, and its metric-distance witnesses, so the partial Cauchy thread is extended by visible packet rows rather than by a selected external point.

After the finite induction terminates, the thread T contains the complete list of centers, radii, inclusions, and tolerance rows generated by D, O, R. The complete-metric row M is then consumed only along this thread, and the terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff is read from the same displayed thread.

The carrier inventory leaves no coordinate for a countable-choice selector, a host intersection of dense opens, or a separate completeness theorem for $\mathsf{Real}^{\uparrow}$. Hence D, O, R, T are the complete refinement surface for accepted $\mathsf{BaireCategory}^{\uparrow}$ limit reads. □

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier refinement schedule obligation`

Theorem 1542.11 (BaireCategory dense-open thread obligation)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. Every dense-open request consumed by Q is routed through the finite source chain*

$$B, M, D, O, R, T, H, C, P, N,$$

where B supplies the $\mathsf{BaireSpace}^{\uparrow}$ prefix window, M supplies the $\mathsf{CompleteMetric}^{\uparrow}$ metric and limit-witness rows, D and O display the dense-open request and predecessor-ball ledger, R records the $\mathsf{Met}^{\uparrow}$ refinement witnesses, and T records the $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ thread handoff.

The obligation exports no countable-choice selector, host intersection theorem, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness row.

Proof. Project the accepted carrier to $B, M, D, O, R, T, H, C, P, N$. The prefix row B fixes the finite Baire-space window in which dense-open requests may be read, and the complete-metric row M supplies the metric carrier and ball classifier used by each request.

Read a dense-open entry from D only after the predecessor ball is present in O. The refinement row R then records the successor ball, positive-radius comparison, and metric-distance witnesses inside the same $\mathsf{Met}^{\uparrow}$ source.

The thread row T stores the resulting finite center, radius, and tolerance data as a $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Re}$ handoff. The rows H, C, P, N only transport, replay, record provenance, and name this same finite route.

Since the carrier has no coordinate for a selector over external dense-open rows, no host intersection object, no open-cover compactness principle, and no ambient real-completeness theorem, the displayed chain exhausts the dense-open thread obligation. □

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier dense open thread obligation`

Theorem 1542.12 (BaireCategory limit-thread nonescape)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. Every terminal limit read from Q factors through the displayed Cauchy thread T and the complete-metric witness row M after the dense-open schedule has already been recorded by B, D, O, R. No accepted read may replace T by a host intersection point, a countable-choice sequence of centers, open-cover compactness, ambient completeness, or a separate completeness theorem for $\mathsf{Real}^{\uparrow}$.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier limit thread nonescape`

Proof. Start with the refinement-schedule obligation, Definition 1542.10. It says that the complete finite refinement surface is D, O, R, T, with every terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff passing through those rows.

The complete-metric row M is consumed only along that displayed thread. Thus a terminal limit read must use the centers, radii, inclusions, and tolerance rows already recorded in T, rather than an external sequence or a host intersection point.

The structural rows H, C, P, N preserve this route componentwise. They do not add a coordinate for countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, so the limit-thread read cannot escape the accepted packet. □

Theorem 1542.13 (BaireCategory NameCert export surface)**.** *For an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier* $Q = (B, M, D, O, R, T, H, C, P, N)$*, the exported NameCert surface consists exactly of the prefix row* B*, complete-metric row* M*, dense-open schedule* D*, open ledger* O*, refinement row* R*, Cauchy thread* T*, componentwise transport* H*, replay row* C*, provenance row* P*, and local naming row* N*. A downstream consumer may read these rows through* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, but no countable-choice selector, host intersection theorem, open-cover compactness row, or ambient real-completeness row is exported.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier namecert export surface`

Proof. The carrier definition lists precisely $B, M, D, O, R, T, H, C, P, N$. The NameCert obligations of Definition 1542.3 bind carrier admission, dense-open classifier exposure, refinement stability, Cauchy-thread exactness, and ledger nonescape for those rows.

Apply Definition 1542.11 to identify the dense-open route through $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Apply Definition 1542.12 for the terminal limit-read boundary.

The exported surface is therefore the finite NameCert packet itself: $B, M, D, O, R, T, H, C, P, N$. The refusal clauses remove countable choice, host intersection, open-cover compactness, and ambient real completeness from the exported interface. □

Theorem 1542.14 (BaireCategory complete-metric thread extraction)**.** *Let* $Q = (B, M, D, O, R, T, H, C, P, N)$ *be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. The finite dense-open schedule in* D*, the open ledger* O*, and the refinement row* R *determine a displayed nested-ball Cauchy thread* T*. Every terminal* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *read from the packet factors through the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness route in* M*, the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet* B*, and the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier complete metric thread extraction`

Proof. Project the carrier along Definition 1542.1. The rows $B, M, D, O, R, T, H, C, P, N$ are the only available coordinates, so the extraction must be assembled from the displayed schedule, ledger, refinement data, thread, transport, replay, provenance, and local naming row.

Apply Definition 1542.4 to the finite schedule D. At each listed dense-open row, the current predecessor ball is already present in O, and the handoff supplies a successor subball admitted by the dense-open witness for that row.

Read the positive-radius comparison, nested inclusion, and metric-distance witnesses from R. These rows give the center and radius data for the next displayed ball and certify that it sits inside the predecessor ball and inside the next dense-open row.

Assembling these finitely many center and radius rows gives the thread T. The componentwise hsame data in H and displayed Cont replay in C keep the thread stable under the accepted packet identifications, while P and N record that the same data are the named $\mathsf{BaireCategory}^{\uparrow}$ surface.

Finally, consume M only through this thread. The complete-metric row supplies the displayed limit-witness route for the nested centers and radii, and the terminal read is then handed through the recorded $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows. No terminal real-name read bypasses T or appeals to an ambient completeness theorem. □

Theorem 1542.15 (BaireCategory dense-open thread obligations)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. The dense-open thread surface consists of the following obligations:*

1. *dense-open schedule exposure through the finite row D;*
2. *predecessor-ball density witnessing through the open ledger O;*
3. *nested refinement through the refinement row R;*
4. *Cauchy-thread extraction through T using the $\mathsf{CompleteMetric}^{\uparrow}$ row M;*
5. *ledger refusal of countable choice, host set intersection, open-cover compactness, ambient completeness, and completeness of $\mathsf{Real}^{\uparrow}$.*

No dense-open thread obligation is stored outside $B, M, D, O, R, T, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier dense open thread obligations`

Proof. Project the carrier of Definition 1542.1. The dense-open schedule is visible only in D, and the predecessor-ball density witnesses are recorded by the open ledger O.

Apply Definition 1542.14. The refinement row R supplies the nested successor balls, and the thread row T is assembled from those finitely displayed refinements before the complete-metric row M is consumed.

The structural rows H, C, P, N preserve and package the same route. Since the carrier inventory has no coordinate for countable choice, host set intersection, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, the thread obligations are exhausted by the displayed finite rows. □

Theorem 1542.16 (BaireCategory complete-metric refinement scope)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted $\mathsf{BaireCategory}^{\uparrow}$ carrier. Every refinement read exported by Q factors through the displayed $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. Concretely, the predecessor ball is read through the metric classifier in M, the finite schedule is named by the stream-prefix row B, the positive-radius and tolerance entries are read through the regular-rational row of T, and the terminal scalar seal is read through the $\mathsf{Real}^{\uparrow}$ handoff recorded by the same thread.*

Proof. Project the carrier of Definition 1542.1. The refinement row R is not an independent object; it is tied to the complete-metric row M, the finite dense-open schedule D, the open ledger O, and the thread T.

Apply the dense-open thread obligations of Definition 1542.15. They force each predecessor-ball request to pass through the displayed open ledger and then through the nested refinement row.

Now use the complete-metric thread extraction theorem. The extracted thread reads the metric carrier and ball classifier from M, reads the finite schedule through the prefix packet B, and records the regular-rational tolerance and terminal real handoff in T.

Thus every refinement read reaches the complete-metric, metric, regular-rational, stream-name, and real rows as displayed carrier rows. There is no separate refinement surface outside those rows. □

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier complete metric refinement scope`

Theorem 1542.17 (BaireCategory dense-open schedule scope)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Finite dense-open schedules are consumed only through the displayed ball-admission rows in D and the predecessor-density rows recorded in O. A downstream consumer may replay the schedule through R and T, but it cannot consume an unlisted dense-open family, an external intersection object, or a host enumeration of opens.*

Proof. Start with the finite schedule row D. By the carrier definition, each schedule entry contains its own open-ball admission datum and is not a pointer to an unrecorded family of dense-open sets.

The predecessor-density data are held in O. Applying the finite meet density handoff shows that each listed dense-open row is used only after the previous displayed ball has been recorded in the open ledger.

The replay through R and T preserves that order. The refinement row records the successor ball and the thread row records the resulting center and radius data, while H, C, P, N transport, replay, source, and name the same finite list.

Since no coordinate stores an external enumeration, a host intersection, or a family of opens outside D, the dense-open schedule is consumed exactly through the displayed ball-admission and predecessor-density rows. □

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier dense open schedule scope`

Theorem 1542.18 (BaireCategory countable-choice boundary)**.** *The* $\mathsf{BaireCategory}^{\uparrow}$ *packet exports no countable-choice selector, host intersection, open-cover compactness theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem. Every accepted consumer read is confined to the carrier rows of Definition 1542.1, the five NameCert obligations of Definition 1542.3, the dense-open handoff of Definition 1542.4, and the complete-metric thread extraction of Definition 1542.14.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier countable choice boundary`

Proof. Read the carrier inventory. The rows $B, M, D, O, R, T, H, C, P, N$ contain a prefix packet, complete-metric row, finite schedule, open ledger, refinement row, Cauchy thread, transport, replay, provenance, and local naming row.

The NameCert obligations include ledger non-escape. That row explicitly refuses countable-choice selectors, host set intersections, open-cover compactness, ambient completeness, and completeness of $\mathsf{Real}^{\uparrow}$ as exported coordinates.

The finite meet density handoff and complete-metric thread extraction describe the accepted route that remains: finite dense-open rows refine displayed predecessor balls, assemble a finite Cauchy thread, and hand terminal reads through the regular-rational and real-name surface.

Therefore a consumer asking for a selector, host intersection, open-cover compactness theorem, or real completeness theorem asks for data that are not carried by the packet. The boundary is exactly the displayed finite route. □

Theorem 1542.19 (BaireCategory countable-choice refusal)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Any* $\mathsf{BaireCategory}^{\uparrow}$ *consumer that asks for a countable selector, a host intersection, an open-cover compactness proof, an ambient completeness theorem, or a* $\mathsf{Real}^{\uparrow}$ *completeness row lies outside the carrier surface. Accepted reads are exhausted by $B, M, D, O, R, T, H, C, P, N$.*

Proof. The carrier inventory in Definition 1542.1 lists the entire accepted surface. Its coordinates are the prefix packet, complete-metric row, finite dense-open schedule, open ledger, refinement row, Cauchy thread, transport row, replay row, provenance row, and local naming row.

The ledger non-escape clause in Definition 1542.3 rules out extra exported coordinates for countable selectors, host intersections, open-cover compactness, ambient completeness, and completeness of $\mathsf{Real}^{\uparrow}$. Thus a consumer requesting one of those objects has left the named packet.

The finite meet density handoff, Definition 1542.4, explains the only accepted dense-open route: displayed finite rows are refined into nested balls and then handed to the thread. The extraction theorem Definition 1542.14 identifies that thread as the sole complete-metric path to the terminal $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ read.

Combining the inventory, non-escape clause, finite handoff, and thread extraction leaves no remaining carrier coordinate through which a countable choice principle or host-level completeness theorem could be read. Therefore the accepted reads are exactly the displayed rows $B, M, D, O, R, T, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier countable choice refusal`

Theorem 1542.20 (BaireCategory finite-schedule choice-free induction)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Every dense-open schedule read in Q is handled by finite induction over the displayed row D: the base stage reads the prefix and complete-metric rows B, M, and each successor stage consumes the next dense-open entry through D, O, R before extending the Cauchy thread T.*

The induction uses the finite meet density handoff, the complete-metric dense thread, ProbeBundle refinement induction, dense-open schedule scope, and countable-choice refusal already displayed by the packet. It exports no countable-choice selector, host intersection, open-cover compactness theorem, ambient completeness theorem, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier finite schedule choice free induction`

Proof. Proceed by induction on the finite schedule row D. In the base stage, the prefix packet B and complete-metric row M supply the initial displayed ball and metric classifier.

At a successor stage, apply the finite meet density handoff of Definition 1542.4. The next dense-open row is read against the predecessor ball already recorded in O, and the refinement row R records the smaller successor ball and its positive-radius data.

Use the ProbeBundle refinement induction of Definition 1542.7 to extend the displayed thread T. The complete-metric dense thread of Definition 1542.5 then keeps the terminal handoff attached to M and T.

Finally apply dense-open schedule scope and countable-choice refusal, Definition 1542.17 and Definition 1542.19. The carrier never leaves the finite rows $B, M, D, O, R, T, H, C, P, N$. □

Theorem 1542.21 (BaireCategory nested-ball thread totality)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Every accepted finite-schedule dense-open read determines a nested-ball thread inside T, and every terminal* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff from that read passes through*

$$B, M, D, O, R \longmapsto T \longmapsto H, C, P, N.$$

No terminal read may bypass the displayed thread by using a host intersection point, a countable-choice sequence of centers, an open-cover compactness theorem, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier nested ball thread totality`

Proof. Start with Definition 1542.14. The finite schedule in D, the open ledger O, and the refinement row R determine the displayed nested-ball Cauchy thread T.

Apply Definition 1542.20. The induction over D accounts for the base and each successor refinement without selecting data outside the packet.

Now consume the complete-metric row through Definition 1542.5. The terminal regular-rational and real-name reads are attached to the same thread T and the same structural rows H, C, P, N.

The refusal clauses close the boundary. By Definition 1542.19, the carrier has no coordinate for a host intersection, choice sequence, compactness theorem, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1542.22 (BaireCategory dense-open thread totality)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. Every accepted finite dense-open schedule D, O, R, T yields a displayed complete-metric Cauchy thread through T before any* $\mathsf{Real}^{\uparrow}$ *seal or meagre-boundary refusal can be read. The totality route factors through the complete-metric obligation surface, dense-open thread obligations, limit-thread nonescape, and countable-choice refusal, and it exports no countable selector, host intersection point, open-cover compactness principle, ambient completeness theorem, or detached* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier dense open thread totality`

Proof. Begin with the complete-metric obligation surface. The rows B and M display the prefix packet and complete-metric classifier before the dense-open schedule is inspected.

Next apply the dense-open thread obligations and the finite-schedule choice-free induction. The rows D, O, R determine each dense-open refinement stage, and the induction extends the same displayed thread T at every accepted step.

Use limit-thread nonescape to keep the terminal thread inside the complete-metric surface. Thus any later regular-rational or $\mathsf{Real}^{\uparrow}$ read is downstream of T and cannot bypass the displayed Cauchy thread.

Finally apply countable-choice refusal. Since the carrier has no coordinate for a selector, host intersection object, open-cover compactness theorem, ambient completeness principle, or detached $\mathsf{Real}^{\uparrow}$ completeness row, the meagre-boundary read must occur only after the finite thread totality route has been displayed. □

Theorem 1542.23 (BaireCategory meagre boundary refusal)**.** *Let $Q = (B, M, D, O, R, T, H, C, P, N)$ be an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier. A meagre-boundary consumer may read only the negative boundary supplied by the dense-open schedule and refinement rows. It cannot convert the packet into a full classical Baire theorem, a countable intersection object, an open-cover compactness principle, a countable-choice selector, ambient completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier meagre boundary refusal`

Proof. Project the carrier surface. The only rows available for dense-open and meagre-boundary reasoning are D, O, R, T together with the prefix, metric, transport, replay, provenance, and local naming rows.

Use Definition 1542.15. The dense-open thread obligations include ledger refusal of countable choice, host set intersection, open-cover compactness, ambient completeness, and completeness of $\mathsf{Real}^{\uparrow}$.

Apply Definition 1542.21. Any positive terminal read remains attached to the nested-ball thread rather than to a host intersection point.

Thus the meagre-boundary consumer receives only a refusal boundary over the finite schedule. It has no carrier coordinate that could supply the full classical Baire conclusion or any of the excluded principles. □

Definition 1542.24 (BaireCategory complete-metric source route)**.** *For an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier* $Q = (B, M, D, O, R, T, H, C, P, N)$*, the complete-metric source route is the ordered finite route*

$$B \longrightarrow M \longrightarrow D, O, R \longrightarrow T \longrightarrow H, C, P, N.$$

The first row is the $\mathsf{BaireSpace}^{\uparrow}$ *prefix source. The second row is the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ *ball-classifier source. The middle rows display the finite dense-open schedule, open ledger, and refinement thread. The thread row* T *carries the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff, and* H, C, P, N *transport, replay, source, and name that same route.*

Lean4 (definition) `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier complete metric source route`

Theorem 1542.25 (BaireCategory source route nonescape)**.** *Every downstream theorem-level read of an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *carrier factors through the complete-metric source route of Definition 1542.24. The carrier exports no host intersection object, countable-choice selector, open-cover compactness principle, quotient stream equality, ambient completeness theorem, or detached* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier source route nonescape`

Proof. Project the carrier of Definition 1542.1. The rows $B, M, D, O, R, T, H, C, P, N$ are exactly the rows listed by the complete-metric source route.

Use the dense-open thread obligation theorem Definition 1542.11. It places every dense-open read over the Baire prefix row, the complete-metric and metric rows, the finite schedule, and the refinement thread.

Apply limit-thread nonescape, Definition 1542.12. The terminal $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads remain attached to the same thread row T rather than to a host intersection or ambient completion object.

Finally H, C, P, N only transport, replay, source, and name rows already present in the route. Since the excluded host, choice, compactness, quotient, and detached-completeness coordinates are absent from the carrier, no theorem-level read can export them. □

Theorem 1542.26 (BaireCategory L10 completion-source boundary)**.** *Every accepted* $\mathsf{BaireCategory}^{\uparrow}$ *packet whose dense-open schedule is consumed by an L10 completion route exposes the ordered finite source chain*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompleteMetric}^{\uparrow}/\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the same BHist, hsame, Cont, Pkg, *and* NameCert *rows. The packet exports no countable-choice selector, host intersection, open-cover compactness, quotient stream, or detached* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier l10 completion source boundary`

Proof. Project the carrier of Definition 1542.1. The rows $B, M, D, O, R, T, H, C, P, N$ expose the Baire-space prefix source, the complete-metric and metric ball classifier, the finite dense-open schedule, the refinement ledger, the terminal thread, and the structural transport, replay, provenance, and local naming rows.

Apply Definition 1542.2. The complete-metric source and the dense-open schedule are read through M, D, O, R, while the finite observation and regular-rational readback are carried by B, T before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Apply Definition 1542.12 and Definition 1542.25. These force every terminal completion read to remain on the same thread T and on the complete-metric source route, with H, C, P, N preserving only that displayed route.

No remaining carrier coordinate names a countable-choice selector, host intersection object, open-cover compactness theorem, quotient stream, or detached $\mathsf{Real}^{\uparrow}$ completeness row. Hence an L10 completion consumer sees exactly the stated ordered source chain and no excluded boundary object. □

Theorem 1542.27 (BaireCategory mature consumer completeness)**.** *Every* $\mathsf{CompactBaire}^{\uparrow}$ *or* $\mathsf{MeagreSet}^{\uparrow}$ *consumer of an accepted* $\mathsf{BaireCategory}^{\uparrow}$ *packet must read the complete-metric row, dense-open schedule, finite refinement row, Cauchy thread,* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows before using the dense-intersection surface. No such consumer obtains countable choice, host intersection, open-cover compactness, ambient completeness, or* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Lean4 ✓ `BEDC.Derived.BaireCategoryUp.BaireCategoryCarrier mature consumer completeness`

Proof. Start with the complete-metric obligation surface, Definition 1542.2. It forces dense-open use through the complete-metric source, the dense-open schedule, the open-row ledger, and the finite refinement row.

Next use the finite meet density handoff and complete-metric dense thread results already established in the chapter. They make the refinement row and dense thread the only route by which a consumer can pass from the schedule to the dense-intersection surface.

Apply the Cauchy thread limit handoff, together with Definition 1542.25 and Definition 1542.26. The terminal observations must pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff carried by the thread row, with H, C, P, N preserving only that displayed route.

The carrier therefore gives $\mathsf{CompactBaire}^{\uparrow}$ and $\mathsf{MeagreSet}^{\uparrow}$ consumers a complete finite reading order before any dense-intersection use. Since the excluded objects have no carrier coordinates, the consumer-completeness row exports no countable choice, host intersection, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BaireCategory}^{\uparrow}$ packet is generated from a $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, $\mathsf{CompleteMetric}^{\uparrow}$ row, finite dense-open schedule, open-row ledger, refinement row, Cauchy thread, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1542.1, Definition 1542.16, Definition 1542.17, Definition 1542.25, and Definition 1542.26 bind the public complete-metric dense-open export surface, including refinement scope, dense-open schedule scope, source-route nonescape, and L10 completion-source boundary for the finite packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close the full classical Baire category theorem as a host theorem, countable choice, host set intersections, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: The theory axis is mature for the finite consumer surface; formal verification and bridge checking remain on the verification axis.

Chapter 1543

A Concrete Naming Certificate for $\mathsf{IntervalDomain}^{\uparrow}$

Following Bishop and Bridges' constructive completion through located intervals and Scott's interval-domain reading of real approximation, the $\mathsf{IntervalDomain}^{\uparrow}$ packet records a finite approximation-domain certificate for the Real-completion boundary. It sits between the endpoint-cell packet of chapter 394, the nested-window packet of chapter 320, the rational-cell packet of chapter 354, the stream-window packet of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The certificate names the directed finite route by which completion-facing consumers read approximants before a $\mathsf{Real}^{\uparrow}$ seal is exposed; it is not a quotient-real construction, a host equality principle, an open-cover compactness theorem, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1543.1 (IntervalDomain carrier)**.** *An* $\mathsf{IntervalDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, R, N, W, Q, E, H, C, P, A)$$

with the following visible rows:

1. *L is a finite family of* $\mathsf{LocatedInterval}^{\uparrow}$ *cells whose endpoint windows are already named by the underlying real-name interface.*
2. *R is the* $\mathsf{RationalInterval}^{\uparrow}$ *refinement ledger attached to those cells, including endpoint order, containment, and width-budget rows.*
3. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *directed nesting row recording which later cells refine earlier cells under the displayed containment route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which endpoint requests and refinement checks are inspected.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family serving the same windows with regular rational observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located cells, rational refinements, nesting row, stream windows, and regular readbacks have been displayed.*
7. *H records componentwise* hsame *transport for L, R, N, W, Q, E and the structural rows.*
8. *C records displayed* Cont *routes for replaying endpoint reads, refinement reads, nesting reads, window reads, regular readbacks, and seal reads in that order.*

9. *P is the* Pkg *provenance over* $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *A is the local* $\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}$ *evidence row.*

The carrier exports only the displayed finite approximation domain. It contains no coordinate for quotient real equality, host real equality, open-cover compactness, a selected limiting representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainTasteGate single carrier alignment`

Theorem 1543.2 (IntervalDomain NameCert obligations)**.** *For an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *carrier* $D = (L, R, N, W, Q, E, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}$ *surface consists exactly of carrier admission, refinement stability, directedness, endpoint-window transport, and real-seal non-escape:*

1. *carrier admission exposes precisely the ten rows* $L, R, N, W, Q, E, H, C, P, A$*;*
2. *refinement stability keeps every* $\mathsf{RationalInterval}^{\uparrow}$ *refinement row attached to its transported* $\mathsf{LocatedInterval}^{\uparrow}$ *cell and its transported window data;*
3. *directedness requires every pair of displayed cells admitted for a common completion read to pass through a later* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row before the seal row is inspected;*
4. *endpoint-window transport moves endpoint reads only through the paired* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows named by the carrier;*
5. *real-seal non-escape states that any* $\mathsf{Real}^{\uparrow}$ *consumer of* E *must first read the located cells, rational refinements, directed nesting row, stream windows, regular readbacks, and structural rows* H, C, P, A*.*

These obligations do not export quotient reals, host real equality, open-cover compactness, selected limiting representatives, or ambient completeness.

Proof. Carrier admission is projection from Definition 1543.1. The packet lists the located interval cells, rational refinement ledger, nesting row, stream windows, regular readbacks, real seal, transports, continuations, provenance, and local naming row as its whole visible surface.

Refinement stability reads L, R, H, C. A transported located cell carries its rational interval ledger with it, and the displayed continuation route keeps the refinement read attached to the same endpoint-window data.

Directedness reads N after the refinement ledger. For a common completion-facing request, two visible cells cannot be consumed independently; the nesting row supplies the later common refinement before the real seal can be exposed.

Endpoint-window transport reads W and Q as a pair. The stream row names the finite endpoint request, and the regular rational row supplies the readback for that request, so a transported endpoint read has no route outside the displayed window and readback rows.

Finally, real-seal non-escape follows because E is downstream of L, R, N, W, Q and the structural rows. Since the carrier has no quotient, host-equality, compactness, selected-limit, or completeness coordinate, none of those principles can be read from the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainNameCertObligations`

Theorem 1543.3 (IntervalDomain completion approximation scope)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *carrier. Every completion-facing consumer of the displayed* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{RationalInterval}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto (H, C,$$

Thus the completion-facing interface is an interval-domain approximation route over finite NameCert data, not a quotient of real names, not host real equality, not open-cover compactness, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the endpoint cells. The row L supplies the visible $\mathsf{LocatedInterval}^{\uparrow}$ data, and R attaches the $\mathsf{RationalInterval}^{\uparrow}$ containment and width ledger used by each cell.

Next read the directed approximation layer. The row N records the $\mathsf{NestedInterval}^{\uparrow}$ refinement order, so a completion-facing consumer moves from finite cells to a common directed refinement before any seal-facing request is available.

Then read the observation layer. The stream windows W and regular rational readbacks Q supply the finite endpoint observations for the same directed refinement route.

The seal row E is reached only after those rows have been consumed. The structural rows H, C, P, A transport, replay, source, and name the same route, so every completion-facing consumer factors through the displayed chain and receives no additional quotient, host-equality, compactness, selected-limit, or completeness data. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainCompletionApproximationScope`

Theorem 1543.4 (IntervalDomain carrier admission)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. Before any interval-domain read is exported, the packet must expose* $\mathsf{LocatedInterval}^{\uparrow}$ *cells,* $\mathsf{NestedInterval}^{\uparrow}$ *nesting,* $\mathsf{RationalInterval}^{\uparrow}$ *refinement rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}$ *rows.*

Proof. Project the carrier of Definition 1543.1. Its visible rows are exactly $L, R, N, W, Q, E, H, C, P, A$, and the introductory dependency route ties those rows to the located interval, nested interval, rational interval, stream, regular rational, and Real seal sibling chapters.

Now apply the carrier-admission clause of Definition 1543.2. A public interval-domain read must consume the listed rows as one finite packet before the local NameCert row A can certify the read.

The completion approximation scope of Definition 1543.3 keeps the same row order for Real-facing consumers, so no interval-domain read can be exported from a strict subpacket or from a host interval primitive. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainCarrierAdmission`

Theorem 1543.5 (IntervalDomain regular Cauchy window)**.** *For every accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet $D = (L, R, N, W, Q, E, H, C, P, A)$, each interval refinement read factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks Q, with* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance support supplied by the rational refinement route and* $\mathsf{Real}^{\uparrow}$ *seal support supplied by E. It does not factor through host intervals, quotient cuts, selected endpoints, or ambient real equality.*

Proof. Start from the endpoint-window transport clause of Definition 1543.2. It binds every endpoint and refinement read to the paired stream-window and regular rational rows W, Q.

Next use Definition 1543.1. The rational refinement ledger R supplies the finite width and containment data for the located cells, while the sibling dyadic and rational interval routes named in the introduction provide the tolerance surface consumed before Q is read.

Finally apply Definition 1543.3. The Real seal row E is downstream of L, R, N, W, Q, so the refinement read is the displayed regular Cauchy window route and not a host interval, quotient cut, selected endpoint, or host equality principle. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainRegularCauchyWindow`

Theorem 1543.6 (IntervalDomain finite endpoint order)**.** *Every accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet* $D = (L, R, N, W, Q, E, H, C, P, A)$ *must expose the* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint-order and containment rows attached to each* $\mathsf{LocatedInterval}^{\uparrow}$ *cell before any* $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer may read the approximation route. The endpoint-order row is a finite BEDC row of the packet, not a host interval primitive, selected endpoint comparison, quotient cut, or ambient completeness witness.*

Proof. Project Definition 1543.1. The located cell row L is paired with the rational refinement ledger R, and the carrier definition states that R includes endpoint order, containment, and width-budget rows for those cells.

Apply Definition 1543.2. Its refinement-stability clause keeps every $\mathsf{RationalInterval}^{\uparrow}$ refinement row attached to its transported $\mathsf{LocatedInterval}^{\uparrow}$ cell and its transported window data.

Finally use Definition 1543.3. The Real-facing route reads $\mathsf{LocatedInterval}^{\uparrow}$ and $\mathsf{RationalInterval}^{\uparrow}$ before $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$, so no consumer can bypass endpoint order through host intervals, selected endpoints, quotient cuts, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainFiniteEndpointOrder`

Theorem 1543.7 (IntervalDomain dyadic endpoint coverage)**.** *Every accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet* $D = (L, R, N, W, Q, E, H, C, P, A)$ *whose* $\mathsf{RationalInterval}^{\uparrow}$ *ledger exposes endpoint order also exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows used by the endpoint windows before any* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer reads the approximation. The coverage row is a finite dependency of the interval-domain packet through chapter 354 and chapter 291; it does not select endpoints, identify quotient cuts, assert host extensional equality, or close ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 1543.1. The rational refinement row R is the endpoint-order, containment, and width-budget ledger attached to the located cells in L.

Read R through the rational-interval dependency named in the chapter introduction. Endpoint order is accepted only together with the tolerance budget that makes the endpoint windows readable, and that budget is supplied by the dyadic-rational core route cited in chapter 291.

Now apply the endpoint-window transport clause of Definition 1543.2. The stream windows W and regular rational readbacks Q may replay an endpoint request only after the same transported rational refinement row has been displayed.

Finally use Definition 1543.3. Since $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are downstream of L, R, N, every completion-facing read first sees the dyadic tolerance support carried by R. The statement therefore records finite endpoint-window coverage only, excluding selected endpoints, quotient cuts, host extensional equality, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDyadicEndpointCoverage`

Theorem 1543.8 (IntervalDomain rational-width monotonicity)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. If a displayed* $\mathsf{LocatedInterval}^{\uparrow}$ *cell is refined through the* $\mathsf{NestedInterval}^{\uparrow}$ *row N, then the* $\mathsf{RationalInterval}^{\uparrow}$ *width-budget row in R is monotonically tightened along the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route before any* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer reads the seal. The monotonicity is a finite relation among the accepted carrier rows L, R, N, H, C, P, A, not a host metric-width comparison, quotient-cut equality, selected endpoint oracle, or ambient completeness claim.*

Proof. Project the carrier of Definition 1543.1. The located cells are carried by L, the rational endpoint-order, containment, and width-budget ledger is carried by R, and the displayed refinement relation is carried by N.

Use Definition 1543.6 and Definition 1543.7. The first keeps endpoint order and containment inside the finite rational-interval row, while the second supplies the dyadic tolerance support needed to compare successive endpoint windows without selecting external endpoints.

Now apply Definition 1543.12. A later nested cell remains inside the same finite route through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the later width budget can only be read as the transported tightening of the earlier displayed budget.

Consequently every stream, regular-rational, or Real-seal consumer sees the tightened rational-width row before it reads the seal. The statement records this monotone finite budget discipline and exports no host comparison or completeness principle. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainRationalWidthMonotonicity`

Theorem 1543.9 (IntervalDomain regular-Cauchy directed width)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. If a directed refinement through N tightens the* $\mathsf{RationalInterval}^{\uparrow}$ *width budget recorded in R, then the same refinement preserves the paired* $\mathsf{StreamName}^{\uparrow}$ *window row W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q before any* $\mathsf{Real}^{\uparrow}$ *seal consumer may inspect E. Thus rational-width monotonicity supplies a regular-Cauchy directed readback route inside the displayed packet, not a host metric-width comparison, quotient cut, endpoint selector, or ambient completeness principle.*

Proof. Project the accepted carrier. The located cells are read from L, the endpoint order, containment, and width-budget ledger from R, the directed refinement row from N, and the paired stream and regular rational observations from W, Q.

Use Definition 1543.6 and Definition 1543.7. These keep endpoint order, dyadic tolerance support, and width comparisons inside the finite rational-interval route attached to the displayed located cells.

Apply Definition 1543.8. Along a nested refinement, the later width budget is the transported tightening of the earlier displayed budget. Then apply Definition 1543.12: the same later cell is read through the common $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ observation surface before the Real seal is available.

Finally use Definition 1543.3. The seal row E lies downstream of L, R, N, W, Q and the structural rows H, C, P, A, so the directed-width route is exactly the regular-Cauchy readback route of the packet. No carrier coordinate supplies a host metric-width comparison, quotient cut, endpoint selector, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainRegularCauchyDirectedWidth`

Theorem 1543.10 (IntervalDomain directed width non-completion)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. The rational-width monotonicity row and regular-Cauchy*

directed-width row expose only finite directed approximation through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *They do not supply quotient cuts, selected endpoints, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1543.8. It reads width tightening only from the displayed located cell, rational refinement ledger, nested refinement row, dyadic tolerance support, and structural rows of the accepted packet.

Next apply Definition 1543.9. The directed-width readback keeps the same refinement inside the paired $\mathsf{StreamName}^{\uparrow}$ window row W and $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q before the $\mathsf{Real}^{\uparrow}$ seal E is inspected.

Finally use Definition 1543.14. That theorem records the complete completion-facing route as the finite BEDC surface through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the structural rows. Since none of the cited rows contains quotient cuts, selected endpoints, trichotomy, host equality, or ambient completeness, directed width remains a finite approximation route and not a completion principle. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDirectedWidthNoncompletion`

Theorem 1543.11 (IntervalDomain directed approximation induction)**.** *Let* $D = (L, R, N, W, Q, E, H, C, P, A)$ *be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet, and let a finite displayed chain of nested refinements be read from the rows* L, R, N. *Induction on the accepted refinement rows gives a stable route through* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any completion-facing consumer reads the seal row* E. *The induction exports no quotient cuts, selected endpoints, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the base refinement row, use the finite endpoint order theorem and the dyadic endpoint coverage theorem, Definition 1543.6 and Definition 1543.7. They place endpoint order, containment, rational width, and dyadic tolerance support inside the displayed $\mathsf{LocatedInterval}^{\uparrow}$ and $\mathsf{RationalInterval}^{\uparrow}$ rows before any stream or seal read occurs.

For the induction step, apply rational-width monotonicity, Definition 1543.8. A later nested row can only be read as the transported tightening of the earlier displayed rational-width budget along the same $\mathsf{NestedInterval}^{\uparrow}$ route.

Next apply regular-Cauchy directed width, Definition 1543.9. The tightened rational-width row preserves the paired $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the $\mathsf{Real}^{\uparrow}$ seal is inspected.

Finally use directed width non-completion, Definition 1543.10. It identifies the resulting chain as a finite approximation route through the seven displayed faces and excludes quotient cuts, selected endpoints, trichotomy, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness at every induction stage. □

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDirectedApproximationInduction`

Theorem 1543.12 (IntervalDomain directed refinement covers)**.** *Let* $D = (L, R, N, W, Q, E, H, C, P, A)$ *be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. Any two accepted interval-domain cells whose displayed* $\mathsf{Real}^{\uparrow}$ *seal readbacks overlap admit a finite directed refinement row through the existing* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. The refinement is read only as a displayed later cell with common stream and regular-rational observations; it does not assert quotient real equality, choose a limit representative, or use ambient completeness.*

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDirectedRefinementCovers`

Proof. Project the carrier rows of Definition 1543.1. The located cells are read from L, their rational containment and width ledgers from R, and their directed nesting relation from N.

Replay the nested-window refinement using the directedness clause of Definition 1543.2. Two cells admitted for a common completion-facing read must pass through a later displayed $\mathsf{NestedInterval}^{\uparrow}$ row before the Real seal can be inspected.

Now read the common observation surface. The $\mathsf{StreamName}^{\uparrow}$ windows W and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks Q expose the shared endpoint observations, while Definition 1543.5 keeps the refinement inside the regular Cauchy window route.

Finally use Definition 1543.3. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of the displayed located, rational, nested, stream, and regular-rational rows, so the directed refinement covers the overlapping readback without exporting quotient real equality, selected limits, or ambient completeness. □

Theorem 1543.13 (IntervalDomain directed-refinement obligation scope)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. For any two overlapping interval-domain cells, the directed refinement row must cite* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *as its complete finite scope before any* $\mathsf{Real}^{\uparrow}$ *consumer reads the seal row E. The obligation exports no quotient cuts, selected limits, host equality, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDirectedRefinementObligationScope`

Proof. Project $D = (L, R, N, W, Q, E, H, C, P, A)$. The carrier rows place located cells in L, rational refinement data in R, directed nesting in N, stream windows in W, regular rational readbacks in Q, the terminal seal in E, and the structural transport, replay, provenance, and naming rows in H, C, P, A.

Apply Definition 1543.12. For two overlapping cells, the common later cell is read only through the displayed route over $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 1543.16. Any consumer of E still factors through the finite route and the structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the refinement row is the complete obligation scope before the seal is read.

Since neither the carrier nor the two cited theorems supplies quotient cuts, selected limits, host equality, trichotomy, or ambient completeness, those objects remain outside the directed-refinement obligation. □

Theorem 1543.14 (IntervalDomain directed completion scope)**.** *Let $D = (L, R, N, W, Q, E, H, C, P, A)$ be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. Any directed-refinement read from D that is consumed by* $\mathsf{Real}^{\uparrow}$ *or by a completion-facing chapter must pass through the complete finite route*

$$\mathsf{LocatedInterval}^{\uparrow}, \mathsf{RationalInterval}^{\uparrow}, \mathsf{NestedInterval}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{C}$$

The handoff is exactly a displayed interval-domain completion route over BEDC rows. It exports no quotient cuts, selected endpoints, trichotomy, host equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainDirectedCompletionScope`

Proof. First read the directed-refinement witness through Definition 1543.12. The common later cell is available only through the located interval, rational interval, nested interval, stream-window, regular-rational, and Real seal surfaces already carried by D.

Next apply Definition 1543.13. That theorem records the complete obligation scope of the same refinement row, adding the structural BHist, hsame, Cont, Pkg, and NameCert rows before any $\mathsf{Real}^{\uparrow}$ consumer can inspect the seal.

Finally use Definition 1543.16. A completion-facing consumer of E receives the directed route only after the displayed approximation and structural rows have been read. The dyadic tolerance support is the same endpoint-budget surface named by the rational-refinement route through chapter 291, so the handoff includes $\mathsf{DyadicRatCore}^{\uparrow}$ without adding a new host principle.

The excluded objects require coordinates absent from the carrier and from the three cited theorems. Thus quotient cuts, selected endpoints, trichotomy, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the completion-facing directed-refinement read. □

Theorem 1543.15 (Interval-domain endpoint directedness scope)**.** *Every completion-facing consumer of an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet* $D = (L, R, N, W, Q, E, H, C, P, A)$ *must expose the directed endpoint rows, rational width monotonicity, dyadic endpoint coverage, directed refinement covers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal route before reading a completion consumer. The scope is the displayed finite route over* $L, R, N, W, Q, E, H, C, P, A$ *and exports no quotient cuts, selected endpoints, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainEndpointDirectednessScope`

Proof. Project the accepted carrier. Endpoint data enters through the located cell row L and rational refinement ledger R, while directedness is carried by the nested refinement row N.

Apply Definition 1543.8, Definition 1543.9, and Definition 1543.7. These bind endpoint order, dyadic tolerance support, and rational width tightening to the same displayed cell route before stream or regular-rational reads are available.

Next use Definition 1543.12 and Definition 1543.13. They require every completion-facing directed refinement to pass through the located, rational, nested, stream-window, regular-rational, Real-seal, and structural rows.

Finally apply Definition 1543.14 and Definition 1543.16. The completion consumer can read the terminal route only after $L, R, N, W, Q, E, H, C, P, A$ have been displayed, so the endpoint-directedness scope remains a finite L10 completion route and not a host completion principle. □

Theorem 1543.16 (IntervalDomain Real seal nonescape)**.** *Let* $D = (L, R, N, W, Q, E, H, C, P, A)$ *be an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *packet. A consumer of the* $\mathsf{Real}^{\uparrow}$ *seal row* E *receives only the finite route through located cells, rational refinements, nested directedness, stream windows, regular rational readbacks, transport, replay, provenance, and local naming. The packet exports no host completeness, selected endpoints, trichotomy, quotient real equality, or ambient interval compactness.*

Lean4 ✓ `BEDC.Derived.IntervalDomainUp.IntervalDomainRealSealNonescape`

Proof. Use the real-seal non-escape clause of Definition 1543.2. It places E after L, R, N, W, Q and after the structural rows H, C, P, A.

The completion approximation scope theorem then identifies the whole Real-facing path as the finite chain through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The sibling anchors in the chapter introduction supply those displayed surfaces.

Since Definition 1543.1 contains no row for host completeness, endpoint selection, trichotomy, quotient real equality, or interval compactness, none of those principles can be exported by consuming E. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalDomain}^{\uparrow}$ packet is generated from finite LocatedIntervalUp cells, RationalIntervalUp refinement ledgers, a NestedIntervalUp directed nesting row, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1543.1, Definition 1543.2, Definition 1543.3, Definition 1543.14, and Definition 1543.16 close the interval-domain route from located cells, rational refinements, nested directedness, stream windows, regular rational readbacks, and the finite $\mathsf{Real}^{\uparrow}$ seal through the displayed hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, host real equality, open-cover compactness, selected limiting representatives, ambient RealUp completeness, or bridge checking.

Upgrade path: Bridge checking remains the separate formal-axis obligation for the same finite interval-domain packet.

Chapter 1544

A Concrete Naming Certificate for $\mathsf{RealSequentialCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style sequential completeness, the $\mathsf{RealSequentialCompleteness}^{\uparrow}$ packet records the real-line sequential-completeness face as finite NameCert data. It reads a regular Cauchy sequence of real names through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ surfaces of chapter 301 and chapter 126, carries the real Cauchy modulus of chapter 1423, and exposes the displayed limit only through the $\mathsf{Real}^{\uparrow}$ surface of chapter 38. The chapter is a real-facing sibling of the metric sequence bridge in chapter 1515; it does not claim quotient completion, countable choice, open-cover compactness, or a classical ambient completeness principle.

Definition 1544.1 (RealSequentialCompleteness carrier)**.** *A* $\mathsf{RealSequentialCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, S, M, L, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row naming the real surface on which the sequence and limit are displayed.*
2. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *sequence data read through* $\mathsf{StreamName}^{\uparrow}$ *finite windows for the real names under consideration.*
3. *M is the displayed real Cauchy modulus row, assigning each requested real-facing tolerance a late window for the sequence.*
4. *L is a displayed* $\mathsf{Real}^{\uparrow}$ *limit row, available to consumers only as a named real surface rather than as a host real or selected quotient representative.*
5. *D is the late-distance ledger; for every tolerance request admitted by M, it records the sufficiently late* $\mathsf{Real}^{\uparrow}$ *distance bounds from the rows of S to L.*
6. *H records componentwise* hsame *transport for R, S, M, L, D, the replay row, the provenance row, and the local naming row.*
7. *C records the displayed* Cont *replay from real-source admission to stream windows, regular rational readbacks, modulus thresholds, late-distance bounds, and the real-limit handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RealSequentialCompleteness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequentialCompleteness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, countable choice, open-cover compactness, a selected host limit, or a classical completeness theorem for the ambient real line.

Theorem 1544.2 (RealSequentialCompleteness NameCert obligations)**.** *For an accepted* $\mathsf{RealSequentialCompletene}$ *carrier* $Q = (R, S, M, L, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSequentialCompleteness}^{\uparrow}}$ *surface consists exactly of five obligations:*

1. *carrier admission for the displayed* $\mathsf{Real}^{\uparrow}$ *source, regular Cauchy real-name sequence, real Cauchy modulus, displayed limit, late-distance ledger, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *classifier stability, requiring any accepted comparison of carriers to transport* R, S, M, L, D *componentwise by* H *before the local classifier identifies the two packets;*
3. *regular-Cauchy convergence, requiring each tolerance request to read a late threshold from* M *and then read the corresponding late-distance rows from* D *through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *sequence surface;*
4. *real-limit handoff, requiring the same finite rows* $R, S, M, L, D, H, C, P, N$ *to present* L *as the real-facing limit surface for the displayed sequence;*
5. *ledger non-escape, stating that no quotient completion, countable-choice schedule, open-cover compactness principle, selected host limit, or ambient completeness proof is exported by the packet.*

Proof. Carrier admission is projection from Definition 1544.1. The packet exposes exactly the real source, sequence row, modulus, limit row, late-distance ledger, transport, replay, provenance, and local name listed there.

Classifier stability is componentwise. A comparison of two accepted packets must first transport the real source, stream windows, regular rational readbacks, modulus thresholds, displayed limit rows, and late-distance ledger by H; the local classifier then reads only the transported rows.

Regular-Cauchy convergence is the joint content of S, M, D. The sequence data is observed through $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the modulus row supplies the late threshold for each tolerance, and the ledger row supplies the corresponding real-facing distance bounds to L.

The real-limit handoff uses no second limit object. The replay row C presents the same rows R, S, M, L, D to the real-facing consumer, while P and N record that the route is sourced and named by this finite packet.

The non-escape clause follows because the displayed rows are exhausted by $R, S, M, L, D, H, C, P, N$. None stores quotient completion equality, a countable-choice schedule, open-cover compactness, a selected host limit, or an ambient completeness proof. □

Theorem 1544.3 (RealSequentialCompleteness regular-Cauchy handoff)**.** *Let* $Q = (R, S, M, L, D, H, C, P, N)$ *be an accepted* $\mathsf{RealSequentialCompleteness}^{\uparrow}$ *packet. If the real source* R*, the sequence data* S*, the modulus* M*, the displayed limit* L*, and the late-distance ledger* D *are displayed, then the same finite packet supplies the real-facing limit surface for the regular Cauchy sequence.*

The handoff route is

$$R \longmapsto S \longmapsto M \longmapsto L \longmapsto D \longmapsto H, C, P, N.$$

It exports sequential completeness only as the displayed $\mathsf{Real}^{\uparrow}$ *row* L *reached from regular Cauchy real-name data, not as quotient completion, countable choice, open-cover compactness, selected host limits, or a classical ambient completeness theorem.*

Proof. Start with the accepted carrier and project R. The real-facing classifier and observation surface used by the handoff are read from this displayed $\mathsf{Real}^{\uparrow}$ row, so the limit claim remains internal to the named real surface.

Next read the sequence and modulus rows. The row S supplies the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks for the regular Cauchy real-name sequence, while M supplies the late threshold requested by each tolerance.

Now read L and D. The row L is already a displayed $\mathsf{Real}^{\uparrow}$ limit row, and D records the late-distance bounds that connect sufficiently late sequence windows to L under the thresholds supplied by M.

Finally replay the route through C and attach H, P, N. These rows transport, source, and name exactly the data required for the real-facing limit surface, and they introduce no coordinate for quotient completion, countable choice, open-cover compactness, selected host limits, or ambient completeness. □

Theorem 1544.4 (RealSequentialCompleteness CauchyCriterion handoff)**.** *Let K be an accepted* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *packet and let $Q = (R, S, M, L, D, H, C, P, N)$ be an accepted* $\mathsf{RealSequentialCompleteness}^{\uparrow}$ *packet. Suppose the two packets expose the same displayed* $\mathsf{Real}^{\uparrow}$ *source, the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and Cauchy ledgers, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed limit row L.*

Then the Cauchy-criterion certificate of K determines one finite BEDC route to the displayed sequential-completeness limit in Q:

$$K \longmapsto R, S \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto L, D, H, C, P, N.$$

The route consumes the Cauchy-criterion certificate only through the matching finite rows and exports no quotient streams, selected host limits, choice principle, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the Cauchy-criterion certificate. By Definition 1590.2, the accepted $\mathsf{RealCauchyCriterion}^{\uparrow}$ packet exposes exactly the carrier admission, window-readback compatibility, dyadic exactness, transport and replay stability, and provenance exhaustion needed for its finite certificate surface.

Next follow the readback and seal rows. Apply Definition 1590.3 to obtain the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and Cauchy ledgers from the same certificate. Then use Definition 1590.4 to factor the terminal real seal through those rows rather than through an independent completion object.

Now compare the resulting rows with the sequential-completeness packet. The hypotheses identify the real source, stream windows, regular rational readbacks, dyadic ledgers, and terminal real seal of K with the corresponding rows of Q, so Definition 1544.2 supplies the accepted $\mathsf{RealSequentialCompleteness}^{\uparrow}$ admission, classifier stability, regular-Cauchy convergence, real-limit handoff, and ledger non-escape clauses for the same finite data.

Finally apply Definition 1544.3. It reads the displayed limit row L and the late-distance ledger D from Q and replays them through H, C, P, N, giving the advertised finite route from the Cauchy-criterion certificate to the displayed sequential-completeness limit.

The non-export claim is inherited from both packets' exhaustion clauses. Neither cited certificate adds quotient streams, selected host limits, choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness to the displayed BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequentialCompleteness}^{\uparrow}}$.

Constructive story: A RealSequentialCompleteness$^{\uparrow}$ packet is generated from a displayed Real$^{\uparrow}$ source, RegSeqRat$^{\uparrow}$ sequence data read through StreamName$^{\uparrow}$ windows, a RealCauchyModulus$^{\uparrow}$ row, a displayed Real$^{\uparrow}$ limit row, late-distance bounds, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealSequentialCompleteness}^{\uparrow}}$, seed)

Scope closed: Definition 1544.1, Definition 1544.2, and Definition 1544.3 seed the finite bridge from regular Cauchy real-name data to the displayed Real$^{\uparrow}$ limit surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealSequentialCompleteness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, countable choice, open-cover compactness, selected host limits, or classical ambient completeness for Real$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, regular-Cauchy convergence, real-limit handoff, and ledger non-escape for the same finite packet.

Chapter 1545

A Concrete Naming Certificate for $\mathsf{IntervalDomainWayBelow}^{\uparrow}$

Following Scott's finite-information order and the Gierz–Hofmann–Keimel–Lawson–Mislove–Scott account of continuous domains, with Bishop–Bridges located intervals as the constructive endpoint ground, the $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ packet records the finite way-below relation used by the L10 interval-domain route. It refines the carrier of chapter 1543 by naming when one displayed approximation cell sits compactly inside another before the $\mathsf{Real}^{\uparrow}$ seal is read. The route is grounded in the sibling certificates for chapter 394, chapter 320, chapter 354, chapter 126, chapter 301, chapter 291, and chapter 38. It is only the finite Scott-domain approximation relation; it is not quotient real equality, host real equality, open-cover compactness, a selected limit, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1545.1 (IntervalDomainWayBelow carrier)**.** *An* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (O, I, M, N, S, Q, E, H, C, P, A)$$

with the following visible rows:

1. *O is the coarser* $\mathsf{IntervalDomain}^{\uparrow}$ *cell whose displayed window is being approximated.*
2. *I is the inner cell, recorded simultaneously as* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint data and a* $\mathsf{RationalInterval}^{\uparrow}$ *containment ledger.*
3. *M is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *margin row witnessing that the inner rational interval lies strictly inside the coarser interval cell.*
4. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement route carrying later cells from O toward I without leaving the displayed interval-domain packet.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which finite endpoint requests for O and I are inspected.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row serving the same windows with regular rational observations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the coarser cell, inner cell, positive margin, nested route, stream windows, and regular readbacks have been displayed.*
8. *H records componentwise* hsame *transport for O, I, M, N, S, Q, E and the structural rows.*
9. *C records displayed* Cont *replay for reading the coarser cell, inner cell, margin, nesting route, windows, readbacks, and seal boundary in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *A is the local* $\mathsf{NameCert}_{\mathsf{IntervalDomainWayBelow}^{\uparrow}}$ *evidence row.*

The intended reading is $I \ll O$: the inner located rational cell is compactly contained in the coarser interval-domain cell by the displayed positive dyadic margin and can be replayed through the finite L10 observation route before any real seal is exposed. The carrier has no coordinate for quotient real equality, host real equality, open-cover compactness, selected limits, or ambient completeness.

Theorem 1545.2 (IntervalDomainWayBelow NameCert obligations)**.** *For an accepted* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *carrier $W = (O, I, M, N, S, Q, E, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalDomainWayBelow}^{\uparrow}}$ *surface consists exactly of carrier admission, strict-margin evidence, nested-route compatibility, observation replay, transport stability, and seal non-escape:*

1. *carrier admission exposes precisely the coarser interval-domain cell, inner located rational cell, positive dyadic margin, nested route, stream windows, regular rational readbacks, seal boundary, transports, continuations, provenance, and local naming row;*
2. *strict-margin evidence requires M to be positive and to certify that both rational endpoints of I lie inside the displayed bounds of O with that margin;*
3. *nested-route compatibility requires every refinement consumer of O that asks for the inner cell to pass through the displayed* $\mathsf{NestedInterval}^{\uparrow}$ *route before reading I;*
4. *observation replay requires the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to serve the same endpoint requests on both sides of the way-below comparison;*
5. *transport stability identifies transported carriers only through the componentwise* hsame *row and the displayed* Cont *replay route;*
6. *seal non-escape states that any* $\mathsf{Real}^{\uparrow}$ *consumer of E must first read $O, I, M, N, S, Q, H, C, P, A$.*

No obligation exports quotient real equality, host real equality, open-cover compactness, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 1545.1. The tuple lists the whole visible surface: coarser cell, inner cell, positive margin, nested route, observation windows, rational readbacks, seal boundary, transport, replay, provenance, and local naming evidence.

Strict-margin evidence is read from I and M. The inner cell carries located endpoints and a rational interval ledger, while the positive dyadic margin records a strict separation from the coarser endpoint bounds named by O.

Nested-route compatibility is read from N. A consumer that starts with the coarser $\mathsf{IntervalDomain}^{\uparrow}$ cell can request the inner cell only by following the displayed $\mathsf{NestedInterval}^{\uparrow}$ route, so the way-below comparison is a finite refinement relation rather than an ambient order assertion.

Observation replay uses S and Q as a paired window/readback layer. The same endpoint request is served through the stream-name window and the regular rational observation row, so replay cannot change which finite data witnesses the comparison.

Transport stability follows from H and C. Componentwise hsame moves each row of the carrier, and the displayed continuation route replays the same reads after transport.

Finally, seal non-escape follows because E is downstream of all approximation rows. Since the carrier has no quotient, host-equality, compactness, selected-limit, or completeness coordinate, none of those principles can be read from the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.IntervalDomainWayBelowUp.IntervalDomainWayBelowCarrier namecert obligations`

Theorem 1545.3 (IntervalDomainWayBelow approximation route)**.** *Let* $W = (O, I, M, N, S, Q, E, H, C, P, A)$ *be an accepted* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *carrier. Every L10 completion-facing consumer of the way-below comparison factors through the finite route*

$$\mathsf{IntervalDomain}^{\uparrow} \longmapsto (\mathsf{LocatedInterval}^{\uparrow}, \mathsf{RationalInterval}^{\uparrow}) \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}$$

Thus the relation $I \ll O$ *is available as finite BEDC approximation data for interval-domain refinement. It is not quotient real equality, host real equality, open-cover compactness, a selected limit, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.IntervalDomainWayBelowUp.IntervalDomainWayBelowApproximationRoute`

Proof. Begin with the coarser approximation cell. The row O supplies the public $\mathsf{IntervalDomain}^{\uparrow}$ boundary, and the row I supplies the inner located rational cell to be compared with it.

The margin row M is then read as a positive $\mathsf{DyadicRatCore}^{\uparrow}$ witness. It is the finite separation datum that turns containment into a way-below comparison rather than a bare subset relation.

Next the nested route N carries the comparison through the directed interval-domain layer. The route forces later finite approximants of O to encounter the displayed inner cell before the seal boundary is available.

The observation layer is read through S and Q. Stream-name windows and regular rational readbacks supply the endpoint observations for the same finite route, and H, C, P, A transport, replay, source, and name it.

The seal row E is reached only at the end of this chain. Therefore every completion-facing consumer factors through the displayed BEDC data and receives no quotient, host-equality, open-cover, selected-limit, or ambient-completeness principle. □

Theorem 1545.4 (IntervalDomainWayBelow obligation-closure package)**.** *For an accepted* IntervalDomainWayBelo *carrier* $W = (O, I, M, N, S, Q, E, H, C, P, A)$*, the carrier admission, strict-margin evidence, nested-route compatibility, observation replay, transport stability, seal non-escape, and completion-consumer factorization jointly supply the full finite way-below obligation surface for the displayed tuple* W*.*

Proof. Apply Definition 1545.1. The tuple $W = (O, I, M, N, S, Q, E, H, C, P, A)$ fixes the coarser interval-domain cell, the inner located rational cell, the positive dyadic margin, the nested route, stream windows, regular rational readbacks, the seal boundary, and the structural rows.

Use Definition 1545.2. It supplies carrier admission, strict-margin evidence, nested-route compatibility, observation replay, transport stability, and seal non-escape as the exact local NameCert surface.

Then apply Definition 1545.3. Every completion-facing consumer factors through the same finite Scott-domain route before the $\mathsf{Real}^{\uparrow}$ seal is read.

Thus the package remains a BHist/Cont/NameCert comparison route. It does not export quotient real equality, host equality, open-cover compactness, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.IntervalDomainWayBelowUp.IntervalDomainWayBelowObligationClosurePackage`
↪ variant `BEDC.Derived.IntervalDomainWayBelowUp.IntervalDomainWayBelowCarrier obligation closure package`

Theorem 1545.5 (IntervalDomainWayBelow strict-margin obligation scope)**.** *Every accepted* IntervalDomainWayB *carrier must expose the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *margin row before any* $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$,

$\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer may read the way-below comparison. The accepted route is therefore a strict-margin route through the displayed finite carrier, not a quotient-real comparison, host equality test, open-cover compactness proof, selected limit, or ambient completeness principle.*

Proof. Project the carrier of Definition 1545.1. The row M is the positive dyadic margin, and it sits before the nested route N, stream windows S, regular readbacks Q, and seal boundary E in the displayed tuple.

Use Definition 1545.2. Its strict-margin clause requires M to certify that the inner rational interval lies strictly inside the coarser interval cell before the local NameCert accepts the comparison.

Apply Definition 1545.3. The completion-facing route factors through $\mathsf{DyadicRatCore}^{\uparrow}$ before it reaches $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$, so every downstream consumer receives the margin row first.

The exclusions follow from the same carrier projection. Since the carrier has no quotient-real equality, host-equality, open-cover compactness, selected-limit, or ambient-completeness coordinate, none of those principles can be supplied by the strict-margin obligation surface. □

Theorem 1545.6 (IntervalDomainWayBelow consumer boundary)**.** *For every accepted* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *carrier* $W = (O, I, M, N, S, Q, E, H, C, P, A)$*, every* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumer must read the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *margin row* M *and the nested route* N *before the seal row* E *is available. The consumer boundary is therefore the displayed strict-margin route through* $O, I, M, N, S, Q, E, H, C, P, A$*, not a quotient-real comparison, host-equality test, selected-limit witness, open-cover compactness argument, or ambient completeness principle.*

Proof. Project the carrier of Definition 1545.1. The positive margin row M precedes the nested route N, the stream windows S, the regular readbacks Q, and the seal boundary E in the displayed packet.

Use Definition 1545.2. Its local NameCert clauses accept the way-below comparison only after the margin row certifies strict inclusion and the nested route records where the comparison is consumed.

Apply Definition 1545.3 together with Definition 1545.5. They force the completion-facing approximation route to pass through M and N before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read can reach the seal row.

Finally inspect the remaining rows H, C, P, A. They transport, replay, source, and name the same strict-margin packet; none supplies a bypass around M, N or an alternate seal. Thus the stated consumers meet exactly the displayed boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalDomainWayBelow}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ packet is generated from a coarser IntervalDomainUp cell, an inner LocatedIntervalUp/RationalIntervalUp cell, a positive DyadicRatCore margin, a NestedIntervalUp refinement route, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntervalDomainWayBelow}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1545.1, Definition 1545.2, Definition 1545.3, Definition 1545.4, Definition 1545.5, and Definition 1545.6 close the finite strict-margin scope for L10 interval-domain way-below approximation.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntervalDomainWayBelow}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.IntervalDomainWayBelowUp.IntervalDomainWayBelowCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, host real equality, open-cover compactness, selected limits, ambient RealUp completeness, public export, or bridge checking.

Upgrade path: Public closure requires an exported way-below approximation certificate consumable by the IntervalDomainUp and RealUp routes without leaving the displayed strict-margin packet.

$\mathsf{Cauchy}^{\uparrow}$ is an L10 route whose regular-sequence, finite-window, dyadic-tolerance, and terminal seal reads are supplied through chapter 301, chapter 126, chapter 291, and chapter 38.

Chapter 1546

A Concrete Naming Certificate for $\mathsf{CauchyRealLocator}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRealLocator}^{\uparrow}$ packet records the finite locator surface by which a displayed Cauchy real answers Real-facing precision and side-location requests. It refines the modulus route of chapter 1423 and chapter 292, reads finite observations through chapter 126, uses the regular rational handoff of chapter 301, records dyadic tolerances through chapter 291, and reaches the terminal seal of chapter 38. The packet is a traditional constructive-analysis locator object reconstructed as finite NameCert data; it does not select a host limit, identify quotient streams, invoke countable choice, decide trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1546.1 (Cauchy real locator carrier)**.** *A* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row. It records the displayed RealCauchyModulus and CauchyModulus threshold data used for every locator request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the locator at the requested precision and side test.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the positive dyadic scale, comparison cells, and slack budget for the displayed windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows into regular rational data before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *locator seal. It records only the Real-facing locator boundary delivered after M, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic ledger, regular readback, terminal locator seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a locator request through M, S, D, Q, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$*, RealCauchyModulus, CauchyModulus,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $M, S, D, Q, R, H, C, P, N$. It has no coordinate for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 1546.2 (Cauchy real locator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier*

$$L = (M, S, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the modulus schedule, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal Real locator seal, transport, replay, provenance, and local naming;*
2. *modulus exactness, requiring every locator request to use the displayed row M rather than an independent threshold, host convergence theorem, or selected limit;*
3. *window and tolerance compatibility, requiring the inspected windows in S to be compared only through the dyadic ledger D;*
4. *RegSeqRat readback, requiring the rational data consumed by the locator seal to pass through the displayed row Q;*
5. *route ordering, requiring R to be read only after M, S, D, Q have been displayed;*
6. *nonescape, excluding host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, and bridge checking.*

No local locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3410.1. The accepted packet displays exactly the modulus schedule, finite windows, dyadic ledger, regular readback, terminal locator seal, and structural rows.

Modulus exactness is forced by the single coordinate M. Every locator request reads its threshold from this row, so the packet has no second convergence witness, hidden host completeness theorem, or selected limit coordinate.

Window and tolerance compatibility are supplied by S and D. The stream row exposes only finite windows, while the dyadic row records the positive scale, comparison cells, and slack budget used to interpret those windows.

The readback and ordering clauses follow from Q and R. The regular rational row is the only rational readback coordinate, and the terminal Real locator seal occurs downstream of M, S, D, Q.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no row for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 1546.3 (Cauchy real locator modulus route)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier factors each locator request through the displayed finite route*

$$M \longmapsto S, D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Thus the request reaches the $\mathsf{Real}^{\uparrow}$ *locator seal only through the visible modulus schedule, finite stream windows, dyadic tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Proof. Begin with an accepted carrier $L = (M, S, D, Q, R, H, C, P, N)$ and a displayed locator request. The only threshold coordinate available to the request is M, so the modulus schedule is read first.

The finite observation step is next. The request opens the window row S and interprets its tolerance through D, whose dyadic ledger records the positive scale and comparison cells.

The rational handoff is then forced through Q. No independent stream or host rational approximation is present in the carrier, so the locator request reaches the Real-facing boundary only after the regular rational readback has been displayed.

Finally the terminal row R is consumed and the structural rows H, C, P, N preserve the same route. This proves the stated factorization and excludes a bypass through selected limits, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Theorem 1546.4 (Cauchy real locator Real seal nonescape)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *locator-seal read reaches* R *only after the displayed rows* M, S, D, Q *have been consumed. The seal exports no host Real equality, quotient stream representative, selected limit, countable-choice selector, trichotomy branch, open-cover compactness witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Inspect the terminal coordinate of the carrier. The only Real-facing locator seal is R, and it is listed after the modulus schedule, stream-window, dyadic-ledger, and regular-readback rows.

Apply the obligation surface of Definition 3410.2. It requires route ordering through M, S, D, Q before R can be read by a Real-facing consumer.

The structural rows preserve this boundary. The hsame transports and Cont replay keep the same displayed packet, while Pkg provenance and local naming record only that route.

Because no excluded coordinate appears in L, the terminal seal cannot export host Real equality, quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. Any consumer asking for one of those principles must leave the local carrier. □

Theorem 1546.5 (Cauchy real locator four-object exit)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *Real-facing locator read factors through exactly the four L10 exit objects named by Definition 3412.8: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *locator seal. In carrier coordinates, any accepted read of* $L = (M, S, D, Q, R, H, C, P, N)$ *uses* S, D, Q, R *as the four object faces and uses* M, H, C, P, N *only as the modulus schedule, transport, replay, provenance, and local naming support. It exports no quotient stream equality, host Real equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier tuple. The row S is the finite $\mathsf{StreamName}^{\uparrow}$ window, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and R is the terminal $\mathsf{Real}^{\uparrow}$ locator seal.

The modulus route theorem Definition 3410.3 fixes the order of consumption. A locator request reads the schedule M, opens the finite windows S, interprets them through the dyadic ledger D, passes through the regular readback Q, and only then reaches R.

The Real-seal nonescape theorem Definition 3410.4 supplies the boundary condition for the terminal read. It rules out every Real-facing bypass that would identify quotient streams, appeal to host Real equality, choose a limit, branch by trichotomy, use open covers, or import ambient completeness.

Finally compare this carrier projection with Definition 3412.8. The four object faces are exactly S, D, Q, R, while M, H, C, P, N supply only schedule, transport, replay, provenance, and naming support. Hence an accepted locator read has the claimed four-object L10 exit route and no additional exported coordinate. □

Theorem 1546.6 (Cauchy real locator induction route)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every locator-side read available to induction is obtained by the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

The induction step reads finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal rows without deciding trichotomy and without choosing a host limit.*

Proof. Project the carrier of Definition 3410.1. The modulus schedule M is the only threshold source, so a locator-side induction step begins by reading the same schedule used in the modulus route.

The finite observation is then S interpreted through D. The stream row exposes only bounded windows, while the dyadic row supplies the tolerance scale and comparison cells needed to use those windows.

The rational and Real-facing parts follow through Q and R. The modulus route theorem Definition 3410.3 fixes this order, and the four-object exit theorem Definition 3410.5 identifies S, D, Q, R as the four L10 exit faces.

Finally H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the carrier has no coordinate for trichotomy or selected host limits, the induction route cannot decide a side by trichotomy or choose a limit outside the displayed locator data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3410.1, Definition 3410.2, Definition 3410.3, Definition 3410.4, Definition 3410.5, and Definition 3410.6 seed the finite locator route over Cauchy modulus schedules, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and the Real locator seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, window and tolerance compatibility, RegSeqRat readback, route ordering, nonescape, and local NameCert registration for BEDC.Derived.CauchyRealLocatorUp.

Constructive story: A $\mathsf{CauchyRealLocator}^{\uparrow}$ packet is generated from a RealCauchyModulusUp and CauchyModulusUp schedule row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp locator seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1547

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 1547.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1547.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 1547.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 1547.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 1547.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 1547.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 1547.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1548

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1548.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through $X, Y, W, D, A, M, S, H, P, N$.*
9. *P is the* Pkg *provenance over* CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 1548.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* CauchySequenceAlgebra$^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *operation budgets,* RegularCauchyAddition$^{\uparrow}$ *route,* RegularCauchyProduct$^{\uparrow}$ *route,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* RegSeqRat$^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 1548.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 1548.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 1548.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 1548.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Cauchy$^{\uparrow}$ gateway reads its completion-facing source through chapter 126, chapter 301, chapter 291, and chapter 38 before any terminal seal is public.

Theorem 1548.7 (Cauchy L10 sibling route)**.** *The Cauchy packet family exported by this hub is consumed through the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

Concretely, the finite-window surface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the completion boundary of chapter 1506 are sibling inputs to the Cauchy carriers rather than hidden ambient structure.

Proof. Read any Cauchy carrier exported by the hub through its finite observation rows. The stream coordinate supplies the inspected windows, the regular-rational coordinate supplies the numerical readback, and the dyadic coordinate supplies the tolerance budget.

The Real-facing seal is reached only after those rows have been displayed. When a completion consumer is present, it receives the same displayed route through the metric-completion boundary.

Thus the hub's Cauchy packet family is attached to the Real and MetricCompletion siblings by visible L10 rows, with no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient completeness coordinate added by the route. □

The Cauchy$^{\uparrow}$ route consumes the L10 sibling chapters before the real locator and sequence algebra inputs are read. In particular, the regular-rational source of chapter 301, the finite stream windows of chapter 126, and the terminal seal of chapter 38 are already part of the displayed dependency spine for every downstream Cauchy locator or algebra consumer. The completion-side consumer in chapter 1937 reads the same RegSeqRat, StreamName, and RealUp spine before exposing its functorial completion route.

Theorem 1548.8 (Cauchy L10 four-object seal route)**.** *Every accepted* Cauchy$^{\uparrow}$ *Real-facing seal route used by the current L10 phase factors through the four displayed sibling faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Concretely, the finite window family of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38 are the only Real-facing sibling rows exported by the route. The supporting rows are the displayed BHist *carrier data, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *registration.*

Proof. Read the route through the existing CauchyRealLocator carrier and its Real-seal nonescape theorem. The carrier exposes a finite StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, and a terminal RealUp locator seal before any Real-facing consumer can read the packet.

Project those four named faces. The StreamName row supplies only inspected finite windows, the DyadicRatCore row supplies the tolerance and comparison budget, the RegSeqRat row supplies the rational readback, and the RealUp row supplies the terminal seal.

The structural rows preserve this factorization. The BHist data name the carried packet, hsame transports the displayed coordinates, Cont replays the same route, Pkg records provenance, and NameCert records the local accepted surface.

Since the carrier has no coordinate for quotient streams, host equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the Cauchy Real-facing seal route. □

Chapter 1549

A Concrete Naming Certificate for $\mathsf{Cauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{Cauchy}^{\uparrow}$ packet records the ordinary constructive Cauchy-real carrier as finite BEDC naming data. It sits above the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational sequence surface of chapter 301, and the terminal seal of chapter 38. The packet is not a quotient construction of reals; it is a finite route by which a modulus, rational readback, transport, replay, provenance, and local naming row become visible before any $\mathsf{Real}^{\uparrow}$ consumer can read the Cauchy object.

Definition 1549.1 (Cauchy carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the finite windows of rational approximants inspected by the carrier.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the positive dyadic scale, comparison cells, and slack conversions used to interpret each requested window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational readback consumed downstream.*
4. *M is the modulus row, recording the displayed threshold assignment used by Cauchy requests and refusing any hidden convergence theorem.*
5. *Q is the modulus/readback compatibility row, requiring the stream window, dyadic tolerance, regular rational sequence, and modulus rows to agree at every displayed precision request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after S, D, R, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-sequence, modulus, compatibility, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy request through $S, D, R, M, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $S, D, R, M, Q, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyUp.CauchyCarrier`

Theorem 1549.2 (Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *carrier*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 3413.1.*

Proof. Carrier admission is projection from Definition 3413.1. The accepted packet displays precisely the stream, dyadic, regular-sequence, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

The stability and exactness obligations are rowwise. The stream row supplies only finite windows, the dyadic row supplies only displayed tolerance cells, the regular-sequence row supplies only rational readback, and the modulus row supplies only the visible threshold assignment.

The compatibility row Q binds these four data rows before any $\mathsf{Real}^{\uparrow}$ seal can be consumed. The structural rows H, C, P, N then transport, replay, source, and name the same route without adding quotient equality, selected limits, host real equality, choice, trichotomy, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCarrier namecert obligations`

Theorem 1549.3 (Cauchy modulus readback route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer route factors through the displayed path*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a precision request reads finite $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the explicit modulus row, the compatibility row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal-facing boundary.*

Proof. Begin with an accepted carrier $X = (S, D, R, M, Q, E, H, C, P, N)$ and a displayed precision request. The only finite observations available to the request are the windows in S, and their tolerance scale is read through D.

The request next reads the explicit modulus row M. The compatibility row Q is the only coordinate that relates the chosen window, tolerance, threshold, and regular-rational readback, so the route cannot bypass it.

After Q is displayed, the consumer may read the $\mathsf{RegSeqRat}^{\uparrow}$ row R and then the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row E. The rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming.

Since the carrier has no quotient, selected-limit, host equality, countable-choice, trichotomy, open-cover, or ambient-completeness coordinate, the displayed route is the whole public Cauchy readback surface. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusReadbackRoute`

Theorem 1549.4 (Cauchy StreamName RegSeqRat window)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet exposes its* $\mathsf{RegSeqRat}^{\uparrow}$ *row only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and modulus/readback compatibility row. Thus a regular rational read of* $X = (S, D, R, M, Q, E, H, C, P, N)$ *must factor through* S, D, M, Q *before it can inspect* R*.*

Proof. Project the carrier of Definition 3413.1. The row S is the only finite observation surface, D is the only displayed tolerance surface, and M is the only modulus row available to the packet.

Apply Definition 3413.2 to identify Q as the compatibility row binding stream windows, dyadic tolerances, regular rational readback, and modulus data. Then apply Definition 3413.3: the displayed route reads S, D, M and Q before R.

The carrier has no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient completeness coordinate. Hence the stated window and compatibility rows exhaust the public access path to $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 1549.5 (Cauchy finite-tail criterion)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

be an accepted $\mathsf{Cauchy}^{\uparrow}$ *carrier. Suppose every displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window admitted by* S *and every dyadic tolerance cell admitted by* D *satisfy the tail bound named by the modulus row* M*, and suppose the compatibility row* Q *identifies those windows with the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of* X *is routed through the same finite tail ledger*

$$S, D, M \longmapsto Q \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The criterion exports no selected limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows available for a tail read are the finite StreamName window row S, the DyadicRatCore tolerance row D, the modulus row M, the compatibility row Q, the RegSeqRat readback row R, and the terminal seal row E.

Induct over the displayed window list retained by S. The empty window has no readback outside the recorded tolerance ledger. In the successor step, the new inspected position is checked against the dyadic cell in D and the threshold supplied by M, so the same finite tail bound remains attached to the extended window.

Apply Definition 3413.4. It forces any access to R through the displayed S, D, M, Q route. Apply Definition 3413.3 to identify Q as the point where the window, tolerance, modulus, and readback rows are jointly inspected.

Finally use Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal E is downstream of the regular rational readback, so no Real-facing consumer can bypass the finite tail ledger. Since the carrier has no selected-limit, quotient-stream, countable-choice, trichotomy, or ambient-completeness coordinate, the displayed finite induction exhausts the criterion. □

Definition 1549.6 (Cauchy finite-window modulus carrier)**.** *The finite-window modulus carrier associated to an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet is the displayed handoff surface*

$$W = (S, D, M, Q, R, E, H, C, P, N)$$

read from the carrier of Definition 3413.1. Its rows have the following roles:

1. *S records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by a tail request.*

2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells and slack conversions attached to those windows.*
3. *M records the explicit modulus threshold row used to accept later-window requests.*
4. *Q records the compatibility row tying each displayed window, dyadic tolerance, modulus threshold, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
5. *R records the regular-rational readback exposed only after S, D, M, Q have been displayed.*
6. *E records the downstream* $\mathsf{Real}^{\uparrow}$ *seal-facing row.*
7. *H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming for the same finite route.*

The carrier admits only these displayed coordinates. It has no coordinate for quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1549.7 (Cauchy finite-window tail induction)**.** *Let $W = (S, D, M, Q, R, E, H, C, P, N)$ be the finite-window modulus carrier of Definition 3413.6. Every accepted tail request obtained by extending the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row preserves the dyadic tolerance row D, the modulus threshold row M, the compatibility row Q, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. Thus the induction route is*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E$$

with H, C, P, N transporting, replaying, sourcing, and naming the same displayed handoff.

Proof. Project the underlying accepted Cauchy carrier through Definition 3413.1 and read it in the window/modulus order specified by Definition 3413.6. The only tail-request rows available are $S, D, M, Q, R, E, H, C, P, N$.

Induct over the displayed $\mathsf{StreamName}^{\uparrow}$ windows retained by S. The empty window has no tolerance request outside D and no readback outside the route named by Definition 3413.4.

For the successor step, the new inspected window is accepted only through the dyadic cell in D and the threshold supplied by M. Apply Definition 3413.5 to keep the same finite tail ledger attached to the extended request.

The compatibility row Q is then the unique displayed row where the window, dyadic tolerance, modulus threshold, and regular readback are jointly inspected. Apply Definition 3413.3 to place R after Q and E after R.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Since none of the rows supplies quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the induction cannot expose any such principle. □

Theorem 1549.8 (Cauchy RegSeqRat handoff obligation)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the finite-window modulus carrier of Definition 3413.6 and the tail-induction route of Definition 3413.7. In particular, the consumer reads S, D, M, Q before R, and any Real-facing use reads E only after that* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Begin with an accepted Cauchy packet and project its rows by Definition 3413.1. The carrier supplies finite stream windows, dyadic tolerances, a modulus row, a compatibility row, regular-rational readback, a terminal Real-facing seal, and structural transport, replay, provenance, and naming rows.

Apply Definition 3413.4. It forces access to the $\mathsf{RegSeqRat}^{\uparrow}$ row through the displayed S, D, M, Q route, so no consumer can inspect R independently of the finite window and tolerance data.

Next apply Definition 3413.5 and Definition 3413.7. These identify the finite tail ledger and show that accepted extensions preserve the dyadic tolerance, modulus threshold, compatibility, and regular readback rows before the Real seal is exposed.

Finally use Definition 3413.3. The structural rows H, C, P, N transport and name the same handoff, while the carrier contains no quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ consumer factors through the stated finite-window induction surface. □

Theorem 1549.9 (Cauchy RegSeqRat Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal real seal. In the notation of Definition 3413.1, the seal-facing route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

It does not expose quotient streams, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $X = (S, D, R, M, Q, E, H, C, P, N)$ from Definition 3413.1. The seal row E is declared terminal and seal-facing; it is not an independent source of Cauchy data.

Use Definition 3413.3 to order the displayed read. The route reaches R after S, D, M, Q and reaches E only after R has been displayed.

Finally apply Definition 3413.2. The rows H, C, P, N transport, replay, source, and name the same route, so they cannot insert quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient completeness between $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 1549.10 (Cauchy modulus transport stability)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

and

$$X' = (S', D', R', M', Q', E', H', C', P', N')$$

be accepted $\mathsf{Cauchy}^{\uparrow}$ *packets. Suppose the stream row, dyadic tolerance row, modulus row, modulus/readback compatibility row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, replay row, provenance row, and local naming row are transported componentwise by the displayed* hsame *data in* H *and* H'*, and suppose both packets are replayed through the same displayed* Cont *route. Then the public Cauchy readback route*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

is the same BEDC route as

$$S', D', M' \longmapsto Q' \longmapsto R' \longmapsto E' \longmapsto H', C', P', N'.$$

In particular, componentwise transport of the modulus/readback surface cannot introduce quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, or ambient completeness.

Proof. Project both accepted packets through Definition 3413.1. The only visible coordinates are the stream, dyadic tolerance, regular-rational readback, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

Apply Definition 3413.3 to X and to X'. The two projected routes must read their displayed stream, dyadic, and modulus rows before their compatibility rows, and the shared Cont replay identifies these reads through the componentwise hsame data.

Next use Definition 3413.4. Access to R and R' is forced through the transported S, D, M, Q and S', D', M', Q' rows, so no other regular-rational readback path is available.

Finally apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal remains downstream of the transported $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the structural rows H, C, P, N and H', C', P', N' only transport, replay, source, and name that path. Since neither carrier contains quotient streams, selected limits, host equality, choice, trichotomy, or ambient completeness, the two displays name the same BEDC route. □

Theorem 1549.11 (Cauchy uniform-modulus Real seal)**.** *For an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

any uniform-modulus consumer of the explicit modulus row M *that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must read* S, D, M, Q *before* R*, and* R *before* E*. The route contains no quotient stream equality, selected limit, countable choice, trichotomy, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 3413.1. The explicit modulus row M is packaged with the finite stream window S, the dyadic tolerance row D, the compatibility row Q, the regular rational readback R, and the terminal seal-facing row E.

Apply Definition 3413.3. It forces a precision request to read S, D, M and then Q before any regular-rational row can be inspected.

Next apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of R, so a uniform-modulus consumer cannot reach the seal before the regular rational readback has been displayed.

Finally use Definition 3413.10. Componentwise transport and the displayed Cont replay preserve the same route for every accepted consumer, and the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1549.12 (Cauchy root filter-completion anchor)**.** *Every Cauchy root consumer that first reaches the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *surface of chapter 1569 must already have read the Cauchy core route, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Project the root hub through its Cauchy core child. The core route exposes the stream-window, tolerance, readback, transport, replay, provenance, and naming rows before any completion-facing consumer can inspect the root packet.

Now read the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ chapter. Its completion handoff consumes the same finite windows and regular rational readback before the terminal Real seal is visible, so the root consumer cannot enter through a detached filter object.

The structural rows preserve the path. Componentwise hsame transport and displayed Cont replay keep the consumer inside the Cauchy root packet, while Pkg provenance and the local $\mathsf{NameCert}$ row record that no quotient filter, selected limit, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported. □

Theorem 1549.13 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 1549.14 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream

consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 1549.15 (Cauchy finite-window modulus pullback)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *modulus read pulls back to the same finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any consumer can inspect a Cauchy value. In the notation of Definition 3413.1, the pullback route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No quotient stream equality, selected limit, host real equality, countable-choice coordinate, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is available along this route.*

Proof. Project the accepted carrier through Definition 3413.1. The finite stream window S, dyadic tolerance row D, modulus row M, compatibility row Q, regular-rational readback R, and terminal Real seal E are the only rows available before the structural rows are read.

Apply Definition 3413.2. It identifies the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ surface with stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility, Real seal ordering, transport, replay, provenance, and local naming.

Now use Definition 3413.3. A precision request first reads S, D, M, Q, then reaches R, and only then reaches E. The route is sharpened by Definition 3413.4, which keeps the $\mathsf{RegSeqRat}^{\uparrow}$ row behind the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, modulus, and compatibility rows.

Finally apply Definition 3413.9. The terminal Real seal is reached only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback has been displayed. Thus every accepted modulus read pulls back to the same four L10 sibling rows before a consumer sees a Cauchy value, and none of the excluded quotient, selected-limit, choice, order, compactness, or completeness coordinates can be inserted. □

Theorem 1549.16 (Regular-Cauchy modulus pullback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of the* $\mathsf{Cauchy}^{\uparrow}$ *finite-window modulus route obtains a dyadic tolerance schedule that is cofinal with the route displayed in Definition 3413.15. In the notation of Definition 3413.1, the consumer may read*

$$S, D, M, Q \longmapsto R \longmapsto E$$

only as a regular-rational handoff: the $\mathsf{Real}^{\uparrow}$ *seal is reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, not before it. The route exports no quotient stream equality, selected limit, host real equality, choice principle, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3413.15. It fixes the finite route from the StreamName window S, dyadic tolerance row D, modulus row M, and compatibility row Q into the regular-rational readback R, and only then into the terminal Real seal E.

Project the route at the $\mathsf{RegSeqRat}^{\uparrow}$ boundary. A RegSeqRat-facing consumer reads the dyadic tolerance row D through the same finite windows S and compatibility row Q, so its tolerance requests are bounded by a cofinal schedule already displayed by the Cauchy modulus route.

Transport the schedule through the regular-rational classifier row. The readback row R is the first row that presents the schedule as regular-rational data, and the Real seal E is reachable only after that readback has been displayed.

Thus the pullback is cofinal at the RegSeqRat surface and terminal at the Real seal. Since the carrier has no coordinate for quotient stream equality, selected limits, host equality, choice,

trichotomy, or Real completeness, none of those principles can be inserted into the consumer route. □

Theorem 1549.17 (Cauchy finite-window pullback uniqueness)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing Cauchy modulus consumer accepted by the* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the same finite route*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

displayed in Definition 3413.15. The consumer cannot replace this route by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows visible to a modulus consumer are the finite StreamName window, dyadic tolerance ledger, modulus row, compatibility row, RegSeqRat readback, Real seal, and the structural rows.

Apply Definition 3413.2. It identifies the local NameCert surface with precisely those rows, so there is no second carrier coordinate from which an alternative Real-facing route could be read.

Use Definition 3413.3. A precision request reaches R only after S, D, M, Q have been displayed, and it reaches the terminal seal E only after the regular-rational readback has been displayed.

Finally apply Definition 3413.15 and Definition 3413.16. The first theorem fixes the finite pullback route to the L10 sibling rows, while the second shows that every RegSeqRat-facing consumer receives the same cofinal handoff. Hence any Real-facing Cauchy modulus consumer factors through the displayed route and gains no quotient, selected-limit, choice, host-equality, or completeness coordinate. □

Every completion-facing consumer of the $\mathsf{Cauchy}^{\uparrow}$ hub reads the finite Cauchy packet through chapter 301 and chapter 38 before any Real seal is exposed to completion rows. The route first displays the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the dyadic tolerance ledger, then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal, and only then lets Cauchy completion consumers inspect the same BHist, hsame, Cont, Pkg, and local NameCert rows. Thus the hub records a structural dependency route, not quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 1549.18 (Cauchy RegSeqRat Real forward dependency)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *source-window read consumed by a Real-facing downstream chapter must pass through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is exposed. The route reads only displayed finite windows, regular rational tail data, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier route in Definition 3413.1. Its source-window reads are displayed before the regular rational readback and terminal real seal rows are available.

Use the regular readback and real-seal routes already named by the Cauchy child files. They order downstream consumption through $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$, with the structural rows transporting and naming the same finite windows.

Since the hub exposes only child-file surfaces and no independent carrier coordinate, a Real-facing consumer can cite $\mathsf{Cauchy}^{\uparrow}$ only through that forward dependency route. The read therefore reaches chapter 301 and chapter 38 before any Real-facing seal is exposed. □

Theorem 1549.19 (Cauchy located-uniform filter forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *route whose explicit modulus row is consumed by a located or uniform Cauchy filter completion must pass through the displayed carrier route*

$$S, D, M, Q, R, E, H, C, P, N$$

of Definition 3413.1, then through chapter 1661 or chapter 1694, before any $\mathsf{Real}^{\uparrow}$ *seal-facing completion row is exported. The forward route has no quotient stream equality, selected limit, countable choice, metric-completeness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host equality.*

Proof. Start with the explicit modulus readback of the Cauchy carrier. By Definition 3413.3, a precision request reads the finite StreamName row S, the DyadicRatCore row D, the explicit modulus row M, and the compatibility row Q before it can inspect the regular rational row R.

Apply Definition 3413.9. The seal-facing row E is downstream of R, so a Real-facing completion consumer cannot bypass the regular rational readback.

Now use Definition 3413.11. Any uniform-modulus consumer of M that reaches a $\mathsf{Real}^{\uparrow}$ seal preserves the same ordering S, D, M, Q before R and R before E.

The remaining completion-neighbour branch is supplied only by the sibling chapters chapter 1661 and chapter 1694. These chapters consume the already displayed Cauchy route as located or uniform filter evidence; neither adds quotient stream equality, selected limits, choice, metric-completeness data, ambient completeness, or host equality to the core Cauchy carrier. □

Theorem 1549.20 (Cauchy route to CauchyCriterion)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *hub's finite route is consumed by chapter 392 through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exports no detached completion carrier, quotient sequence equality, selected modulus outside the displayed rows, countable-choice schedule, or host completeness principle.*

Proof. Read the hub through its existing finite siblings. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows provide the only Cauchy data visible to downstream criteria.

The structural rows hsame, Cont, Pkg, and NameCert transport, replay, source, and name that same finite route. Therefore the Cauchy criterion consumer receives exactly the displayed BEDC rows and no external completion or choice object. □

Theorem 1549.21 (Cauchy hub Bishop completion forward route)**.** *The regular-source material routed by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by chapter 1639 through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The forward route exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness principle.*

Proof. Read the Cauchy hub through its regular-source children. Those children expose dyadic tolerance arithmetic, finite stream windows, regular rational readback, and the terminal real seal as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The Bishop regular Cauchy completion chapter consumes exactly that L10-facing route. Its completion read is mediated by the same finite BHist carrier discipline, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Since the hub route contains no detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness coordinate, the forward edge to the Bishop completion packet is precisely the displayed finite dependency route. □

Chapter 1550

A Concrete Naming Certificate for $\mathsf{IntegerRealEmbedding}^{\uparrow}$

Following Bishop and Bridges on integers as exact rational constants inside constructive real analysis, the $\mathsf{IntegerRealEmbedding}^{\uparrow}$ packet records the finite route by which an $\mathsf{Int}^{\uparrow}$ signed-pair row becomes a $\mathsf{Real}^{\uparrow}$ name. The route does not ask the real layer to forget the integer source: it passes first through the integer certificate of chapter 31, then the rational source surface of chapter 37, the rational-to-real route of chapter 1491, the stationary diagonal route of chapter 362, the constant stream surface of chapter 126, the regular-rational handoff of chapter 301, the dyadic tolerance row of chapter 291, and the terminal real seal of chapter 38.

Definition 1550.1 (IntegerRealEmbedding carrier)**.** *An* $\mathsf{IntegerRealEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (z, q, S, R, D, E, H, C, P, N)$$

with the following displayed coordinates:

1. z *is the* $\mathsf{Int}^{\uparrow}$ *signed-pair row naming the integer source;*
2. q *is the induced* $\mathsf{Rat}^{\uparrow}$ *row obtained by reading* z *as the rational constant* $z/1$*;*
3. S *is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule whose admitted finite windows all read the same rational row* q*;*
4. R *is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff obtained from the constant schedule* S*;*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation row used to compare the stationary rational reads at the displayed finite precisions;*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the rows* z, q, S, R, D *have been exposed;*
7. H *records componentwise* hsame *transports for the integer source, rational image, constant schedule, stationary readback, dyadic tolerance row, real seal, package provenance, and local naming row;*
8. C *records the visible* Cont *routes from* z *to* q*, from* q *to* S*, from* S *to* R*, from* R *to* D*, and from* D *to* E*;*
9. P *is the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}$ *row.*

The carrier is finite data for integer constants as real names. It contains no quotient stream, no host integer equality oracle, no host real equality row, no selected limit, no trichotomy principle, and no ambient completeness coordinate.

Definition 1550.2 (IntegerRealEmbedding classifier)**.** *The* $\mathsf{IntegerRealEmbedding}^{\uparrow}$ *classifier compares two accepted carriers*

$$K = (z, q, S, R, D, E, H, C, P, N), \qquad K' = (z', q', S', R', D', E', H', C', P', N')$$

only through the displayed route rows. It first compares the $\mathsf{Int}^{\uparrow}$ *signed-pair sources* z, z' *by the integer classifier, then compares the induced* $\mathsf{Rat}^{\uparrow}$ *rows* q, q' *by the rational classifier, then compares the constant* $\mathsf{StreamName}^{\uparrow}$ *schedules, stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *handoffs, dyadic tolerance rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows through their visible* hsame *and* Cont *transports. The provenance rows* P, P' *and local certificate rows* N, N' *must name the same finite dependency surface.*

Theorem 1550.3 (IntegerRealEmbedding NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier: the finite packet of Definition 1550.1;*
2. *classifier: comparison through the* $\mathsf{Int}^{\uparrow}$ *source, the induced* $\mathsf{Rat}^{\uparrow}$ *row, the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the* $\mathsf{Real}^{\uparrow}$ *seal;*
3. *stability: componentwise* hsame *transport preserves the integer source, rational image, constant schedule, stationary readback, dyadic observations, and real seal;*
4. *exactness/readback: every real-facing read factors through* z, q, S, R, D *before consulting* E*;*
5. *ledger: the* Pkg *and local* $\mathsf{NameCert}$ *rows expose only* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local certificate coordinates.*

No row of the surface is a quotient stream, host integer equality oracle, host real equality, trichotomy principle, selected limit, or ambient completeness principle.

Proof. Project the carrier of Definition 1550.1. Its visible coordinates already name the integer source, rational image, constant stream schedule, stationary regular-rational readback, dyadic observation row, real seal, transports, continuation routes, package provenance, and local certificate row.

The classifier row is fixed by Definition 1550.2. It never compares real names directly before the displayed integer-to-rational and rational-to-stationary route has been read.

Stability follows by transporting each listed coordinate through the stored hsame row and using the displayed Cont route for the corresponding handoff. Exactness is then forced because the only route to E is the route through z, q, S, R, D.

The ledger is exhausted by P and N. Since the carrier has no coordinate for quotient streams, host integer equality, host real equality, trichotomy, selected limits, or ambient completeness, none of those objects can be exported by the local certificate. □

Theorem 1550.4 (IntegerRealEmbedding stationary stream route)**.** *Every accepted* $\mathsf{IntegerRealEmbedding}^{\uparrow}$ *carrier routes its* $\mathsf{Int}^{\uparrow}$ *source row* z *through the induced rational row* q *and the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *before any regular-rational or real-facing coordinate is available. Every admitted stream window therefore reads the same rational constant induced by* z*.*

Proof. Read the continuation coordinates of the carrier. The first displayed handoff is z to q, and the next displayed handoff is q to the constant schedule S.

The schedule S has no independent stream seed. Each admitted finite window is generated from the same rational row q, so the stream route is stationary at the rational constant induced by the integer source.

Only after this constant schedule is displayed can the packet expose the regular-rational handoff R and the later real-facing rows. Thus every consumer that reaches those later rows has already passed through z, q, S. □

Theorem 1550.5 (IntegerRealEmbedding RegSeqRat handoff)**.** *For every accepted* $\mathsf{IntegerRealEmbedding}^{\uparrow}$ *carrier* $K = (z, q, S, R, D, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is the stationary regular-rational readback of the constant stream schedule* S*. The dyadic tolerance row* D *can inspect only finite precision reads of that stationary row before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is opened.*

Proof. Apply Definition 1550.4. It identifies every admitted stream window of S with the same rational row q obtained from z.

The handoff coordinate R is reached from S by the displayed continuation route and has no independent sequence source. Its readback is therefore stationary at the same rational row.

The dyadic row D is downstream from R in the carrier. It records only the finite tolerance and observation data attached to that stationary readback, so it cannot inspect an unrelated regular sequence before the real seal is reached. □

Theorem 1550.6 (IntegerRealEmbedding RealUp seal nonescape)**.** *For every accepted* $\mathsf{IntegerRealEmbedding}^{\uparrow}$ *carrier* $K = (z, q, S, R, D, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *factors through the displayed dependency chain*

$$\mathsf{Int}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}.$$

The seal exports only the local rows $z, q, S, R, D, H, C, P, N$ *as boundary data. It does not export quotient-stream equality, host integer equality, host real equality, trichotomy, selected limits, or ambient completeness.*

Proof. Start with a read of the real seal E. By the carrier definition, E is reachable only through the displayed predecessor D, whose predecessor is R, whose predecessor is S, whose predecessor is q, whose source is z.

Use Definition 1550.5. It shows that the regular-rational row and dyadic observation row are attached to the stationary constant readback induced by the integer source.

The remaining rows H, C, P, N provide transport, continuation, provenance, and local naming for the same finite route. Since the carrier contains no coordinate for quotient-stream equality, host integer equality, host real equality, trichotomy, selected limits, or ambient completeness, the real seal cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}$.

Constructive story: An $\mathsf{IntegerRealEmbedding}^{\uparrow}$ packet is generated from an IntUp signed-pair source row, induced RatUp constant row, constant StreamName schedule, stationary RegSeqRat handoff, DyadicRatCore tolerance row, RealUp seal, componentwise transports, continuation routes, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 1550.1, Definition 1550.2, Definition 1550.3, Definition 1550.4, Definition 1550.5, and Definition 1550.6 seed the finite integer-constant route from $\mathsf{Int}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntegerRealEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient-stream equality, host integer equality, trichotomy, selected limits, ambient completeness, or a checked Lean certificate.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, stationary StreamName routing, RegSeqRat handoff, DyadicRatCore observation exactness, RealUp seal factorization, and nonescape.

Chapter 1551

A Concrete Naming Certificate for $\mathsf{IsometryExtension}^{\uparrow}$

Following Bishop and Bridges constructive completion and the standard extension theorem for isometries on dense subspaces, the $\mathsf{IsometryExtension}^{\uparrow}$ packet records the finite BEDC surface by which a dense isometric embedding extends across an L10 completion route. It sits after the dense completion entry of chapter 1512, the displayed distance-reflection route of chapter 439, the extension-and-uniqueness surface of chapter 397, and the unit boundary of chapter 1527. The packet also fixes the finite name route through chapter 126, chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38; it is a named obligation surface, not a quotient-completion construction.

Definition 1551.1 (IsometryExtension carrier)**.** *An* $\mathsf{IsometryExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, J, I, U, W, R, S, A, L, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row together with the dense source reads admitted by the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *packet.*
2. *Y is the target completion-facing metric row whose target reads are reached through displayed completion evidence.*
3. *J is the dense completion embedding row, read through* $\mathsf{CompletionEmbedding}^{\uparrow}$ *rather than through a host inclusion map.*
4. *I is the distance-preserving graph row supplied by* $\mathsf{IsometricEmbedding}^{\uparrow}$*.*
5. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-and-uniqueness row attached to the same dense source and target route.*
6. *W is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *unit row that admits source points into the completion-facing surface.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row used by the source completion read.*
8. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row used to inspect the displayed regular Cauchy name.*
9. *A is the* $\mathsf{Real}^{\uparrow}$ *distance and seal row reached after the stream-window and regular-rational rows have been read.*

10. *L records componentwise* hsame *transport for the source, target, dense embedding, isometry, universal-property, unit, stream, regular-rational, and real-seal rows.*
11. *C records displayed* Cont *replay through $X, Y, J, I, U, W, R, S, A, L$.*
12. *P is the* Pkg *provenance over* CompletionEmbedding$^{\uparrow}$, IsometricEmbedding$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\mathsf{IsometryExtension}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, or completeness of Real$^{\uparrow}$.

Theorem 1551.2 (IsometryExtension NameCert obligations)**.** *For an accepted* IsometryExtension$^{\uparrow}$ *carrier $E = (X, Y, J, I, U, W, R, S, A, L, C, P, N)$, the local* NameCert$_{\mathsf{IsometryExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source, target, dense embedding, isometry, universal-property, unit, stream-window, regular-rational, real-seal, transport, continuation, provenance, and local naming rows;*
2. *dense-source compatibility, requiring every extension read to enter through J and the unit row W before any target completion-facing row is consumed;*
3. *isometric distance preservation, requiring each source-distance read to pass through I and then through the* Real$^{\uparrow}$ *row A;*
4. *universal-property extension, requiring the extension candidate and its uniqueness ledger to be read only through U over the same dense isometric route;*
5. *finite-name handoff, requiring stream windows S and regular rational names R to be displayed before the real-seal row A is consumed;*
6. *ledger non-escape, stating that L, C, P, N expose no quotient completion equality, selected representatives, countable choice, open-cover compactness, or completeness of* Real$^{\uparrow}$.

Proof. Carrier admission is projection from Definition 1551.1. The tuple lists exactly the source and target rows, dense embedding row, isometry row, universal-property row, unit row, finite-name rows, real-seal row, and structural BEDC rows available to the certificate.

Dense-source compatibility is read from J and W. The completion embedding supplies the dense source image, and the unit row is the only admission route from a displayed source point into the completion-facing surface.

Distance preservation is then forced by I and A. The isometry row records the distance-reflection route, while the real-seal row records the terminal distance observation after the source stream and regular-rational names have been displayed.

The universal-property obligation is local to U. Since U is attached to the same J, I, W, R, S, A route, extension existence and uniqueness can be read only as a ledger over those displayed rows.

Finally inspect L, C, P, N. Transport is componentwise, replay follows the displayed route, and provenance names only the listed sibling packets. Hence the excluded quotient, representative-choice, countability, compactness, and real-completeness principles have no row from which a consumer could read them. □

Theorem 1551.3 (IsometryExtension completion handoff)**.** *Every completion-facing consumer of an accepted* IsometryExtension$^{\uparrow}$ *carrier factors through the displayed route*

$$X, J, W \longmapsto R, S \longmapsto I, A \longmapsto U \longmapsto Y.$$

Thus the extension reaches the target completion-facing row only after the dense completion embedding, unit admission, finite stream and regular-rational name rows, real-distance seal, and universal-property extension ledger have been replayed inside the same packet.

Proof. Begin with the source side. The source metric row X reaches a completion-facing read only through the dense embedding row J and the unit row W, so no detached source completion point is exposed.

Next read the finite name route. The regular rational row R and stream-window row S display the source Cauchy evidence before any real-distance seal is available.

Apply the isometry route through I and A. The distance-preserving graph transports the displayed source evidence to the target side, and the real-seal row records the resulting distance observation.

The universal-property row U then supplies the extension and uniqueness ledger for the same dense isometric route. Only after this ledger is displayed can a consumer read the target row Y as the completion-facing endpoint of the extension.

The structural rows L, C, P, N preserve the same ordered route and add no quotient completion equality, selected representative, countable-choice schedule, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 1551.4 (IsometryExtension Real seal boundary)**.** *For every accepted* $\mathsf{IsometryExtension}^{\uparrow}$ *carrier* $E = (X, Y, J, I, U, W, R, S, A, L, C, P, N)$*, each* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by the extension is determined by the displayed chain*

$$J, W, R, S, I, A, U.$$

The seal cannot be read as a quotient-completion equality, a selected representative, a countable-choice selection, an open-cover compactness witness, or a proof of completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier and isolate the real-seal coordinate A. It appears only after the dense embedding row J, unit row W, regular-rational row R, stream-window row S, and isometry row I have been displayed.

Use the completion handoff of Definition 1551.3. Any completion-facing consumer reaches the target side by replaying the same J, W, R, S, I, A, U route before reading Y.

The universal-property row U keeps the extension ledger attached to that real-seal route. A second seal, a selected representative, or a quotient equality would require a carrier coordinate outside the displayed chain.

No such coordinate exists in E. Transport, continuation, provenance, and local naming preserve the listed rows componentwise, so every $\mathsf{Real}^{\uparrow}$ read remains bounded by $J, W, R, S, I, A, U, L, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{IsometryExtension}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IsometryExtension}^{\uparrow}$ packet is generated from source and target metric rows, a CompletionEmbedding row, an IsometricEmbedding row, a CompletionUniversalProperty row, a CauchyCompletionUnit row, StreamName and RegSeqRat finite-name rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IsometryExtension}^{\uparrow}}$, seed)

Scope closed: Definition 1551.1, Definition 1551.2, Definition 1551.3, and Definition 1551.4 seed the finite isometry-extension surface for L10 completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IsometryExtension}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-source compatibility, isometric distance preservation, universal-property extension, finite-name handoff, Real-seal boundary, and ledger non-escape for BEDC.Derived.IsometryExtensionUp.

Chapter 1552

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySubsequenceCriterion}^{\uparrow}$ packet records the finite route by which a displayed source sequence is accepted as having a Cauchy subsequence before any completion-facing consumer reads it. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the inspected fast-subsequence route of chapter 1518, the monotone-cofinal regular-subsequence surface of chapter 381, and the terminal real-name boundary of chapter 38. The packet is a finite criterion surface for downstream Bolzano–Weierstrass and completion consumers; it does not select an infinite subsequence by choice, identify quotient streams, compare host real equalities, or assert completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1552.1 (CauchySubsequenceCriterion carrier)**.** *A* $\mathsf{CauchySubsequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, W, Q, D, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row whose finite* $\mathsf{StreamName}^{\uparrow}$ *windows are inspected before any subsequence is accepted;*
2. *T is the monotone subsequence schedule, recorded as a finite unary reindexing ledger with visible order and later-index comparisons on inspected indices;*
3. *W records the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows before and after applying the displayed schedule T;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and no independent rational stream;*
5. *D is the dyadic tolerance ledger tying the selected windows to the requested Cauchy radii and pointwise comparison rows;*
6. *F is the* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *witness that the selected windows satisfy the displayed fast-Cauchy bounds;*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row that repacks the same schedule, windows, dyadic tolerance ledger, and regular-rational readback for downstream consumers;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after Q, F, R and the tolerance ledger have been displayed;*

9. *H records componentwise* hsame *transport for the source row, schedule, window rows, regular-rational readback, dyadic ledger, fast witness, regular-subsequence handoff, real seal, and support rows;*

10. *C records displayed* Cont *replay from source inspection through schedule selection, finite windows, tolerance ledger, fast-Cauchy witness, regular-rational readback, regular-subsequence handoff, and real-seal delivery;*

11. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FastCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceCriterion}^{\uparrow}}$ *row.*

The carrier stores no selected infinite subsequence object, arbitrary choice functional, quotient stream, host real equality, independent real carrier, or completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 1552.2 (CauchySubsequenceCriterion NameCert obligations)**.** *For an accepted* CauchySubsequenceCrite *carrier $K = (S, T, W, Q, D, F, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source row, monotone schedule, inspected window rows, regular-rational readback, dyadic tolerance ledger, fast-Cauchy witness, regular-subsequence handoff, real seal, transport, replay, provenance, and local naming row;*

2. *schedule monotonicity for T, requiring each displayed later index to be read from the finite unary order ledger rather than from a host-level selector;*

3. *window transport, requiring every selected* $\mathsf{StreamName}^{\uparrow}$ *window in W to be transported from an inspected source window along T;*

4. *fast-Cauchy handoff, requiring the witness F to consume exactly the transported windows and dyadic tolerance rows displayed by W and D;*

5. *regular-rational readback, requiring Q to be obtained from the same selected windows and not from an independent rational sequence;*

6. *Real seal non-escape, requiring E to be reached only through the displayed fast witness, regular-rational readback, and* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff;*

7. *ledger exactness for D, H, C, P, N, requiring dyadic tolerance, transport, replay, package provenance, and local naming to mention only the rows of the carrier.*

No obligation row exports a selected infinite subsequence, arbitrary choice functional, quotient stream, host real equality, independent real carrier, or completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier. The tuple displays exactly the source row, monotone schedule, inspected windows, regular-rational readback, dyadic tolerance ledger, fast witness, regular-subsequence handoff, real seal, componentwise transport, continuation replay, package provenance, and local naming row.

Read the schedule and window rows first. The row T supplies only finite monotone later-index comparisons, and W records only the source and selected windows transported along that ledger. Thus the criterion has no coordinate from which an arbitrary subsequence selector could be read.

Use the analytic handoff rows next. The row D records the dyadic tolerances, F verifies the fast-Cauchy subsequence route over those same rows, Q gives the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and R repacks the displayed data as a $\mathsf{RegularCauchySubsequence}^{\uparrow}$ consumer surface.

Finally the real seal and structural rows preserve the same dependency order. The row E sits downstream of Q, F, R and D, while H, C, P, N transport, replay, source, and name only the displayed coordinates. Since the carrier is exhausted by $S, T, W, Q, D, F, R, E, H, C, P, N$, the excluded choice, quotient, host-equality, independent-real, and completeness data have no row. □

Theorem 1552.3 (CauchySubsequenceCriterion fast-subsequence handoff)**.** *Every accepted* CauchySubsequenceCri *packet hands any downstream* BolzanoWeierstrass$^{\uparrow}$, Real$^{\uparrow}$, *or completion-facing consumer only the displayed subsequence schedule, finite windows, dyadic tolerance rows, fast-Cauchy witness,* RegSeqRat$^{\uparrow}$ *handoff,* RegularCauchySubsequence$^{\uparrow}$ *route, and real seal. In the notation of Definition 1552.1, every such read factors through*

$$S, T, W, D, F, Q, R, E, H, C, P, N.$$

The consumer receives no selected infinite subsequence, arbitrary choice functional, quotient stream, host real equality, independent real carrier, or Real$^{\uparrow}$ *completeness principle.*

Proof. Start with a downstream read of an accepted criterion packet. The only exposed source of subsequence data is the schedule row T together with the selected window row W.

The dyadic tolerance ledger D is read before the fast witness F, so the Cauchy assertion is witnessed by finite radius and comparison rows. The readback row Q and the regular-subsequence row R consume the same selected windows, preventing any independent rational stream or separate subsequence route from entering the handoff.

The real seal E is reached only after those rows have been displayed. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route for equivalent packets.

No other coordinate exists in the carrier. Hence every Bolzano–Weierstrass, real-name, or completion-facing consumer receives exactly the displayed finite subsequence criterion package and cannot read a selected infinite subsequence, choice functional, quotient stream, host real equality, independent real carrier, or completeness field. □

Closure status for NameCert$_{\text{CauchySubsequenceCriterion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySubsequenceCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 1552.1, Definition 1552.2, and Definition 1552.3 seed the finite criterion route from inspected source windows to fast subsequence evidence, regular rational readback, regular-subsequence handoff, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{CauchySubsequenceCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, arbitrary choice, quotient streams, host real equality, independent real carriers, Real$^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Formal closure requires checked carrier admission, schedule monotonicity, window transport, fast-Cauchy handoff, regular-rational readback, Real seal non-escape, and ledger exactness for BEDC.Derived.CauchySubsequenceCriterionUp.

Constructive story: A CauchySubsequenceCriterion$^{\uparrow}$ packet is generated from a source sequence row, a monotone finite subsequence schedule, inspected StreamName windows, RegSeqRat readback, dyadic tolerance rows, a FastCauchySubsequence witness, a RegularCauchySubsequence handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1553

A Concrete Naming Certificate for $\mathsf{Liminf}^{\uparrow}$

Following Hardy and Bishop–Bridges, $\mathsf{Liminf}^{\uparrow}$ records the finite tail-lower-cut companion to the tail-upper-cut surface of chapter 768. The packet consumes the bounded sequence rows of chapter 1415, the lower-cut observation discipline of chapter 1440, the upper-cut comparison discipline of **??**, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It gives Real-and-Completion consumers a finite lower-tail boundary for bounded real sequences without importing Real completeness, trichotomy, a host infimum, quotient stream equality, selected subsequences, or open-cover compactness.

The lower-tail rows are consumed together with the bounded upper-envelope data of chapter 7101 in Real sequence oscillation packets. In that route, tail-cut lower envelope rows from $\mathsf{Liminf}^{\uparrow}$ pair with bounded upper envelope rows from $\mathsf{BoundedRealLimsup}^{\uparrow}$ before any gap-width or zero-gap convergence certificate is exported.

Definition 1553.1 (Liminf carrier)**.** *A* $\mathsf{Liminf}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, L, D, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row whose finite windows, bound ledger, and displayed* $\mathsf{Real}^{\uparrow}$ *entries are being inspected.*
2. *L is the tail lower-cut schedule, recording for each displayed tail index the bounded lower-cut observations read through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*.*
3. *D is the dyadic tolerance ledger, recording the finite rational tolerances used to compare tail lower cuts with the upper observation surface.*
4. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row named as the liminf candidate exposed by the packet.*
5. *H records componentwise* hsame *transports for S, L, D, T and the structural rows.*
6. *C records the displayed* Cont *routes by which consumers replay the bounded sequence row, tail lower-cut schedule, dyadic tolerance ledger, terminal real seal, transports, provenance, and local naming row.*
7. *P is the* Pkg *provenance over* $\mathsf{Liminf}^{\uparrow}$*,* $\mathsf{BoundedRealSequence}^{\uparrow}$*,* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*,* $\mathsf{BoundedRealUpperCut}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Liminf}^{\uparrow}}$ *row.*

The carrier stores a finite lower-tail cut surface for a bounded real sequence. It has no coordinate for Real completeness, trichotomy, a host infimum, quotient stream equality, selected subsequences, or open-cover compactness.

Lean4 (definition) `BEDC.Derived.LiminfUp.LiminfCarrier`

Lemma 1553.2 (Liminf carrier route rows)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *carrier of Definition 1553.1 projects the three route rows*

$$\mathsf{Cont}(S, L, D), \qquad \mathsf{Cont}(D, T, C), \qquad \mathsf{Pkg}^{\mathrm{sig}}(B, P, \mathsf{pkg}),$$

namely bounded-sequence to lower-cut to dyadic replay, dyadic to terminal-real to continuation replay, and bundle provenance through the displayed package row.

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfCarrier route rows`

Proof. Project the displayed carrier rows of Definition 1553.1. The route coordinate C contains the bounded sequence, lower-cut, dyadic, terminal, and replay continuations, while the provenance row P supplies the package signature.

Thus the accepted carrier exposes exactly the two Cont rows and the $\mathsf{Pkg}^{\mathrm{sig}}$ provenance row named above, without adding a lower-envelope theorem or a host infimum. □

Definition 1553.3 (Liminf classifier)**.** *Two* $\mathsf{Liminf}^{\uparrow}$ *carriers*

$$I = (S, L, D, T, H, C, P, N) \qquad \text{and} \qquad I' = (S', L', D', T', H', C', P', N')$$

are classified as the same displayed liminf packet exactly when the bounded sequence rows match componentwise, the tail lower-cut schedules expose the same finite tail observations, the dyadic tolerance ledgers carry the same rational comparison budgets, the terminal $\mathsf{Real}^{\uparrow}$ *seals are transported by the displayed* hsame *data, the continuation replays preserve the same consumer routes, and the provenance and local NameCert rows name the same* $\mathsf{Liminf}^{\uparrow}$ *surface. The classifier is therefore rowwise over* S, L, D, T, H, C, P, N *and does not identify carriers by host sequence equality, quotient streams, selected cofinal subsequences, or a completed infimum construction.*

Lean4 (definition) `BEDC.Derived.LiminfUp.LiminfClassifier`

Theorem 1553.4 (Liminf NameCert obligations)**.** *For an accepted* $\mathsf{Liminf}^{\uparrow}$ *carrier* $I = (S, L, D, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Liminf}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier formation for the bounded real sequence row, tail lower-cut schedule, dyadic tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transports, continuations, provenance, and local naming data;*
2. *classifier stability for the componentwise comparison of Definition 1553.3;*
3. *lower-tail exactness, stating that every lower observation in* L *is read from a finite tail of* S *through the bounded lower-cut surface;*
4. *upper-cut compatibility, stating that every tolerance row in* D *compares the same tail data against the bounded upper-cut observation surface;*
5. *terminal seal routing, stating that the candidate row* T *is consumed only as a* $\mathsf{Real}^{\uparrow}$ *seal after the displayed tail-cut checks have been replayed;*
6. *ledger non-escape, excluding Real completeness, trichotomy, host infimum selection, quotient stream equality, selected subsequences, and open-cover compactness.*

Proof. Carrier formation is projection from Definition 1553.1. The packet displays exactly the bounded sequence row, tail lower-cut schedule, dyadic tolerance ledger, terminal real seal, componentwise transports, continuation routes, provenance, and local name.

Classifier stability is the componentwise comparison of Definition 1553.3. Transport along H may replace the visible rows only by rows with the same scheduled tail observations, the same dyadic budgets, and the same terminal seal route, while C preserves the consumer replay order.

Lower-tail exactness is supplied by L together with the bounded sequence row S. Each accepted lower observation names the finite tail index, the inspected sequence windows, and the bounded lower-cut certificate through which the read is made.

Upper-cut compatibility is supplied by D and the sibling bounded upper-cut discipline. The tolerance ledger records only finite dyadic comparisons between the same tail data and the upper observation surface, so it does not become a host supremum or a completeness witness.

Terminal seal routing is the role of T. A consumer may read the candidate liminf only after the sequence tail, lower-cut, upper-cut, and tolerance rows have been replayed; the terminal coordinate remains a $\mathsf{Real}^{\uparrow}$ seal rather than a completed infimum object.

The non-escape row follows from the carrier inventory. No displayed field carries Real completeness, trichotomy, a host infimum, quotient stream equality, selected subsequences, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfNameCertObligations`

Theorem 1553.5 (Liminf tail-cut route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto L \longmapsto D \longmapsto T \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before using the displayed liminf surface.

Proof. Begin with the bounded sequence row S. It supplies the only displayed source of sequence entries, finite windows, and boundedness evidence available to the packet.

The tail lower-cut schedule L then names which tails and lower observations are inspected. The dyadic ledger D records the tolerance comparisons that align those tail-lower observations with the bounded upper-cut surface.

Only after those rows are replayed can a consumer inspect the terminal real seal T. The structural rows H, C, P, N then transport, replay, source, and name the same finite route, so the consumer cannot bypass the tail-cut surface by reading a host infimum or a selected subsequence. □

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfTailCutRoute`

Theorem 1553.6 (Liminf tail lower-envelope compatibility)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *packet*

$$I = (S, L, D, T, H, C, P, N)$$

whose bounded sequence row S is paired with the bounded upper-envelope surface of $\mathsf{BoundedRealLimsup}^{\uparrow}$ *factors its lower-tail schedule through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *and its dyadic tolerance ledger through* $\mathsf{BoundedRealUpperCut}^{\uparrow}$ *before any gap-width or zero-gap convergence consumer reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal T.*

Proof. Project the accepted carrier into its displayed rows S, L, D, T, H, C, P, N. The row S is the bounded sequence source described in Definition 1553.1, and the paired upper-envelope data is read through the sibling $\mathsf{BoundedRealLimsup}^{\uparrow}$ chapter rather than through an ambient supremum.

Apply the tail-cut route of Definition 1553.5. That route forces every consumer to replay S before L, and L is exactly the finite lower-tail schedule whose observations are certified by the bounded lower-cut chapter. Thus the lower-envelope half of the paired liminf–limsup surface reaches the consumer only through $\mathsf{BoundedRealLowerCut}^{\uparrow}$.

The dyadic tolerance row D is then replayed before the terminal seal T. By the upper-cut compatibility row in Definition 1553.4, those tolerances compare the same tail data against the bounded upper-cut surface, so the upper comparison half factors through $\mathsf{BoundedRealUpperCut}^{\uparrow}$.

The structural rows H, C, P, N only transport, replay, source, and name this ordered route. Hence a later gap-width or zero-gap convergence certificate can read T only after the bounded sequence row, lower-cut schedule, upper-cut tolerance ledger, and limsup sibling surface have all been exposed as finite certificate data. □

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfTailLowerEnvelopeCompatibility`

Theorem 1553.7 (Liminf RegSeqRat lower-cut public export)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or Real-and-Completion consumer of an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet receives the lower-envelope claim only after the displayed route exposes the bounded sequence row* S*, the* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *tail schedule* L*, the* $\mathsf{BoundedRealUpperCut}^{\uparrow}$ *tolerance comparison through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*. The public export is the finite packet*

$$S, L, D, T, H, C, P, N$$

read through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it does not export a host infimum, selected tail subsequence, quotient stream equality, trichotomy decision, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfRegSeqRatLowerCutPublicExport`

Proof. Project the accepted carrier of Definition 1553.1. Its public read surface is exactly the row family S, L, D, T, H, C, P, N, where S is the bounded sequence source, L is the finite lower-tail schedule, D is the dyadic tolerance ledger, and T is the terminal Real seal.

Apply the route theorem Definition 1553.5. It orders every consumer read through S, L, D, T before the structural rows H, C, P, N can transport, replay, source, and name the packet.

Use the lower-envelope compatibility theorem, Definition 1553.6. It identifies the lower side with $\mathsf{BoundedRealLowerCut}^{\uparrow}$ and the comparison side with $\mathsf{BoundedRealUpperCut}^{\uparrow}$ through the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any Real-and-Completion consumer may inspect T.

The export is therefore public only as a finite RegSeqRat-facing handoff. Since no carrier row stores a host infimum, selected subsequence, quotient stream equality, trichotomy decision, open-cover compactness, or ambient completeness principle, the lower-envelope claim cannot be strengthened beyond those displayed rows. □

Theorem 1553.8 (Liminf tail-envelope monotone refinement)**.** *Let an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet* $I = (S, L, D, T, H, C, P, N)$ *be paired with bounded upper-envelope data from* $\mathsf{BoundedRealLimsup}^{\uparrow}$*. If the displayed tail schedule* L *is refined by a later monotone tail schedule while preserving the same bounded sequence row* S*, lower-cut route through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*, upper-cut tolerance ledger* D*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*, then every gap-width or zero-gap convergence consumer reads the refined packet through the same bounded-sequence, lower-envelope, upper-tolerance, and terminal-seal route.*

Proof. Project the carrier and classifier. The bounded sequence row S is unchanged, and the classifier of Definition 1553.3 permits only rowwise comparison of the displayed tail schedule, dyadic ledger, terminal seal, replay, provenance, and local name.

Refine the tail schedule monotonically. The new lower observations remain lower-cut reads through $\mathsf{BoundedRealLowerCut}^{\uparrow}$, and the tolerance row D still compares the same bounded-sequence data against the upper-cut surface before the terminal seal is visible.

Apply the lower-envelope compatibility theorem above. It already forces every liminf–limsup consumer to expose S, L, D, T in that order, with the bounded upper-envelope surface present before any gap-width or zero-gap convergence certificate can inspect T.

The structural rows H, C, P, N transport, replay, source, and name the refined route. Since the refinement changes only the displayed tail schedule and preserves the bounded sequence row, lower-cut route, upper-cut tolerance ledger, and terminal Real seal, no zero-gap consumer can bypass the finite envelope or read a host infimum, selected subsequence, quotient stream equality, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfTailEnvelopeMonotoneRefinement`

Theorem 1553.9 (Liminf–limsup zero-gap convergence)**.** *Let an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet*

$$I = (S, L, D, T, H, C, P, N)$$

be paired with an accepted $\mathsf{BoundedRealLimsup}^{\uparrow}$ *packet whose bounded sequence row is the same displayed S, whose tail schedule is the same displayed L, and whose gap-width ledger is displayed as zero along the same dyadic tolerance row D. Then every convergence consumer that treats the lower and upper envelopes as coincident must first read the terminal* $\mathsf{Real}^{\uparrow}$ *seal through the shared route*

$$S, L, D, T, H, C, P, N$$

before exporting the convergence result.

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfLimsupZeroGapConvergence`

Proof. Start with the lower envelope. Theorem 1553.6 forces the $\mathsf{Liminf}^{\uparrow}$ half of the pair to expose the bounded sequence row S, the lower-tail schedule L, and the dyadic tolerance row D before the terminal seal T can be consumed.

Next use the monotone refinement theorem, Definition 1553.8. It keeps later tail reads on the same bounded-sequence, lower-envelope, upper-tolerance, and terminal-seal route whenever the displayed schedule is refined without changing S, D, T.

Now attach the sibling $\mathsf{BoundedRealLimsup}^{\uparrow}$ packet. Its upper-envelope surface is paired with the same S, the same tail schedule L, and the same zero gap-width ledger over D, so a convergence consumer has no separate upper route from which to read a different terminal value.

Therefore the zero-gap read can only identify the two envelope surfaces after the common route S, L, D, T, H, C, P, N has been displayed. The packet exports no Real completeness principle, trichotomy, selected subsequence, quotient stream equality, host infimum, or host supremum as a substitute for that finite route. □

Theorem 1553.10 (Liminf tail lower-upper gap width)**.** *Let an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet*

$$I = (S, L, D, T, H, C, P, N)$$

be paired with a $\mathsf{BoundedRealLimsup}^{\uparrow}$ *upper-envelope packet over the same bounded sequence row* S*. Any zero-gap or convergence-facing read from the pair is exposed only after the shared bounded sequence row* S*, lower-cut schedule* L*, upper-cut comparison surface, dyadic tolerance ledger* D*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*, and structural rows* H, C, P, N *have been displayed. The resulting gap-width handoff is a finite dyadic envelope route, not a Real completeness principle, host infimum or supremum, selected subsequence, or quotient stream equality.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfTailLowerUpperGapWidth`

Proof. Start with lower-tail compatibility, Definition 1553.6. It forces the $\mathsf{Liminf}^{\uparrow}$ half of the pair to expose S, L, D, T before any lower-envelope consumer can inspect the terminal seal.

Next apply monotone refinement, Definition 1553.8. It keeps later tail reads on the same bounded-sequence, lower-cut, upper-comparison, dyadic-tolerance, and terminal-seal route whenever the schedule is refined without changing S, D, T.

Finally use zero-gap convergence, Definition 1553.9. Once the $\mathsf{BoundedRealLimsup}^{\uparrow}$ packet is paired with the same S and dyadic tolerance row D, a convergence-facing read can only pass through the shared finite gap-width route.

The structural rows H, C, P, N transport, replay, source, and name that route. Since the proof uses only the displayed lower envelope, upper comparison surface, dyadic ledger, and Real seal, it introduces no Real completeness, host infimum, host supremum, selected subsequence, or quotient stream equality. □

Theorem 1553.11 (Liminf Real seal non-escape)**.** *An accepted* $\mathsf{Liminf}^{\uparrow}$ *packet exposes its candidate value only as the terminal* $\mathsf{Real}^{\uparrow}$ *row* T *reached through the displayed tail-cut route. In particular, it does not export Real completeness, trichotomy, a host infimum, quotient stream equality, selected subsequences, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfRealSealNonescape`

Proof. The only value-facing coordinate in the carrier is T. All sequence-facing information is confined to S, L, D, and all structural access is confined to H, C, P, N.

A consumer that asks for the candidate value must therefore follow the continuation replay from the bounded sequence row through the tail lower-cut and tolerance ledgers before reaching T. The replay exposes a sealed real name, not a construction by infimum, compactness, or subsequence selection.

Since no carrier field stores the excluded principles, componentwise transport and continuation replay cannot introduce them. The $\mathsf{Real}^{\uparrow}$ seal remains terminal for the packet, and the liminf surface remains a finite tail-cut certificate. □

Theorem 1553.12 (Liminf tail-infimum obligation route)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *obligation read must expose the rows*

$$S, L, D, T, H, C, P, N$$

before any tail-infimum consumer can read the packet. The source row S*, tail lower-cut schedule* L*, dyadic tolerance ledger* D*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* T*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N *are therefore the whole tail-infimum surface visible to the consumer.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfObligationTailInfimumRoute`

Proof. Project the accepted carrier of Definition 1553.1. Its tuple already orders the bounded sequence source, lower-tail schedule, dyadic ledger, terminal seal, transport, replay, provenance, and local naming rows.

Apply the route theorem Definition 1553.5. It forces every Real-and-Completion consumer to replay S, L, D, T, H, C, P, N before using the displayed liminf surface.

A tail-infimum consumer is such a value-facing consumer, so it receives no additional row. The read is exhausted by the displayed carrier route and cannot bypass it through a host infimum or selected subsequence. □

Theorem 1553.13 (Liminf bounded-cut compatibility obligation)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *obligation read that compares lower-cut and upper-cut data must use the same bounded sequence row* S *and the same dyadic tolerance ledger* D*. The lower-cut observations in* L *and the upper-cut comparison surface are compatible only as finite reads from that shared bounded sequence and shared tolerance ledger before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T *is consumed.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfBoundedCutCompatibilityObligation`

Proof. Read the carrier rows S, L, D, T, H, C, P, N. The row S is the only bounded sequence source, and D is the only displayed tolerance ledger available to comparisons.

Use Definition 1553.4. Its lower-tail exactness row routes lower observations through the finite tail data of S, while its upper-cut compatibility row uses D to compare that same tail data against the bounded upper-cut surface.

The structural rows H, C, P, N preserve this common source and tolerance ledger under transport, replay, provenance, and local naming. Thus a consumer cannot pair the lower cut with a different bounded sequence, a different dyadic budget, a host supremum, or ambient completeness data. □

Theorem 1553.14 (Liminf Real-seal exhaustion obligation)**.** *Every value-facing read of an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet ends at the displayed* $\mathsf{Real}^{\uparrow}$ *seal* T*. It cannot expose a host infimum, selected subsequence, trichotomy, quotient stream equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness beyond the finite route* S, L, D, T, H, C, P, N*.*

Proof. Start from Definition 1553.11. The theorem identifies T as the only value-facing coordinate and places it after the bounded sequence, lower-cut, and tolerance rows.

The obligation route of Definition 1553.12 gives the complete list of rows a value-facing consumer may read. The bounded-cut compatibility obligation keeps lower-cut and upper-cut reads on the same S, D route before T is reached.

No carrier row remains after T except the structural transport, replay, provenance, and local naming rows already listed. Hence the displayed Real seal exhausts the value-facing surface and the excluded host principles are not exported by the packet. □

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfRealSealExhaustionObligation`

Theorem 1553.15 (Liminf lower-cut readback scope)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *lower-tail read must expose the bounded sequence source* S*, the finite lower-cut schedule* L*, the dyadic tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* T *before any lower-cut readback is consumed. The source row is read through* $\mathsf{BoundedRealSequence}^{\uparrow}$*, the lower observations through* $\mathsf{BoundedRealLowerCut}^{\uparrow}$*, the comparison rows through* $\mathsf{BoundedRealUpperCut}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal value only as the* $\mathsf{Real}^{\uparrow}$ *seal named in Definition 1553.1, Definition 1553.4, Definition 1553.5, and Definition 1553.11.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfLowerCutReadbackScope`

Proof. Project the carrier $I = (S, L, D, T, H, C, P, N)$ of Definition 1553.1. The source row S is the displayed bounded real sequence, and the schedule L is the finite lower-cut ledger attached to tails of that same source.

Use the NameCert obligations of Definition 1553.4. Their lower-tail exactness row routes each accepted lower observation through $\mathsf{BoundedRealLowerCut}^{\uparrow}$, while the upper-cut compatibility row routes comparisons through $\mathsf{BoundedRealUpperCut}^{\uparrow}$ using the dyadic tolerance ledger D.

Apply the tail-cut route of Definition 1553.5. It orders the read as S, L, D, T, H, C, P, N, so the terminal value cannot be inspected before the bounded source, lower schedule, and dyadic comparison rows have been displayed.

Finally use the Real-seal non-escape theorem, Definition 1553.11. The endpoint is only the row T, not a host infimum, trichotomy decision, selected subsequence, quotient stream equality, or ambient completeness principle. □

Theorem 1553.16 (Liminf scoped tail envelope)**.** *Every accepted* $\mathsf{Liminf}^{\uparrow}$ *consumer must route lower-tail data through the shared* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row, the* $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{BoundedRealUpperCut}^{\uparrow}$ *comparison surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any tail-envelope read is public. The scoped route is the finite envelope*

$$S, L, D, T, H, C, P, N$$

read through $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host infimum, trichotomy decision, selected subsequence, quotient stream equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfScopedTailEnvelope`

Proof. Start with the NameCert obligations of Definition 1553.4. They identify the accepted packet as the finite row family S, L, D, T, H, C, P, N and place lower-tail exactness, upper-cut compatibility, terminal sealing, transport, replay, provenance, and local naming in that family.

Use the tail-cut route of Definition 1553.5. It orders every value-facing read through the bounded sequence row, lower-tail schedule, dyadic tolerance ledger, and terminal Real seal before the structural rows can replay the packet to a consumer.

Apply the bounded-cut compatibility obligation, Definition 1553.13. It forces lower-cut and upper-cut comparisons to use the same source row S and the same tolerance ledger D, so the lower envelope cannot be paired with an unrelated upper surface.

Finally read the lower-cut scope theorem, Definition 1553.15. It identifies the public lower-tail readback as a route through $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{BoundedRealUpperCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal, with BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed rows. □

Theorem 1553.17 (Liminf lower-cut readback)**.** *Every lower-tail observation exposed by an accepted* $\mathsf{Liminf}^{\uparrow}$ *carrier*

$$I = (S, L, D, T, H, C, P, N)$$

is read through the $\mathsf{BoundedRealLowerCut}^{\uparrow}$ *row* L*, compared through the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and sealed only at the terminal* $\mathsf{Real}^{\uparrow}$ *row* T*. The readback may cite the bounded source* S *and the upper comparison surface, but it does not expose a host infimum, a selected subsequence, trichotomy, quotient stream equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfLowerCutReadback`

Proof. Project the carrier of Definition 1553.1. The source S is the bounded sequence row, L is the finite lower-tail schedule, D is the dyadic tolerance ledger, and T is the only terminal Real seal.

Use Definition 1553.15. It already forces public lower-tail readback to pass through $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{BoundedRealLowerCut}^{\uparrow}$, $\mathsf{BoundedRealUpperCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal in that same displayed packet.

The present readback only isolates the lower-cut side of that route. Every lower observation is therefore a read from L compared against D, and the value-facing endpoint remains T rather than a host infimum, selected subsequence, trichotomy decision, quotient stream equality, or ambient completeness principle. □

Theorem 1553.18 (Liminf regular-sequence route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{Liminf}^{\uparrow}$ *packet must pass through*

$$\mathsf{BoundedRealSequence}^{\uparrow},\quad \mathsf{BoundedRealLowerCut}^{\uparrow},\quad \mathsf{BoundedRealUpperCut}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{R}$$

before reading the liminf surface S, L, D, T, H, C, P, N*. The route exposes finite regular-sequence and cut data only; it does not claim Real completeness, an actual infimum object, a quotient-real equality, a selected tail subsequence, or an open-cover compactness principle.*

Lean4 ✓ `BEDC.Derived.LiminfUp.LiminfRegularSequenceRoute`

Proof. Begin with Definition 1553.16. The scoped envelope names the bounded sequence, lower-cut schedule, dyadic tolerance ledger, upper comparison surface, and terminal Real seal as the public order of the liminf packet.

Next read the regular-sequence-facing part of the route. The bounded source row S is interpreted through $\mathsf{BoundedRealSequence}^{\uparrow}$, its lower and upper observations pass through $\mathsf{BoundedRealLowerCut}^{\uparrow}$ and $\mathsf{BoundedRealUpperCut}^{\uparrow}$, and all finite comparisons use the common $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D.

The consumer may then pass through $\mathsf{RegSeqRat}^{\uparrow}$ and only finally through the $\mathsf{Real}^{\uparrow}$ seal T. The support rows H, C, P, N transport, replay, source, and name the same finite packet, so the route supplies no host infimum, selected subsequence, quotient equality, Real-completeness principle, or compactness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{Liminf}\uparrow}$**.**

Constructive story: A $\mathsf{Liminf}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a finite tail lower-cut schedule, a dyadic tolerance ledger, a terminal RealUp seal, componentwise hsame transports, Cont replay routes, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Liminf}\uparrow}$, scopedClosed)

Scope closed: Definition 1553.1, Definition 1553.3, Definition 1553.4, Definition 1553.5, Definition 1553.11, Definition 1553.15, and Definition 1553.16 scope the finite tail-cut liminf surface for bounded real sequences.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Liminf}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, a host infimum, quotient stream equality, selected subsequences, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Public closure requires checked carrier formation, classifier stability, lower-tail exactness, upper-cut compatibility, terminal seal routing, tail-cut route ordering, lower-cut readback, and Real seal non-escape for the displayed packet.

Chapter 1554

A Concrete Naming Certificate for $\mathsf{RegulatedRealFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegulatedRealFunction}^{\uparrow}$ packet names a real-valued function on a closed interval by a finite ledger of dyadic step approximations together with a uniform tail route, rather than by a host function-space object.

The packet sits after four visible neighbours: the dyadic step-function approximation surface of chapter 428, the uniform-limit route of chapter 463, the closed interval domain row of chapter 1448, and the endpoint seal surface of chapter 1454. It exposes the approximation and seal rows needed by later integration-facing work, but it does not supply integration, open-cover compactness, or ambient completeness.

Definition 1554.1 (Regulated real function NameCert surface)**.** *A* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, A, T, U, E, H, C, P, N)$$

with the following displayed coordinates:

1. *I is a* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *domain row naming the source interval and its endpoint order data;*
2. *A is a finite ledger of* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation rows, indexed by displayed dyadic tolerances and read through chapter 428;*
3. *T is the uniform tail and modulus ledger recording the common threshold beyond which the approximants are uniformly close;*
4. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *route that consumes A and T through chapter 463;*
5. *E is the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *endpoint seal row, read only after the closed interval and uniform-limit routes of chapter 1448 and chapter 1454;*
6. *H stores componentwise* hsame *transports for the interval row, approximation ledger, tail ledger, uniform-limit route, endpoint seal, provenance, and local name;*
7. *C stores the displayed* Cont *continuations that read a domain request, choose the approximation threshold, replay the step-function ledger, pass through U, and expose E;*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedUnitInterval}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{RegularLocatedCauchyRea}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed interval row, the finite dyadic step-function ledger, the uniform tail ledger, the uniform-limit route, the endpoint seal row, and the visible transport, continuation, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.RegulatedRealFunctionUp.RegulatedRealFunctionTasteGate single carrier alignment`

Theorem 1554.2 (Regulated real function NameCert obligations)**.** *For the carrier of Definition 1554.1, the local* $\mathsf{NameCert}_{\mathsf{RegulatedRealFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for* $G = (I, A, T, U, E, H, C, P, N)$*, interval-domain admission through chapter 1448, finite step-approximation admission through chapter 428, uniform-tail routing through chapter 463, endpoint sealing through chapter 1454, componentwise transport, continuation replay, provenance, local naming, and non-escape from hidden function-space equality.*

Lean4 ✓ `BEDC.Derived.RegulatedRealFunctionUp.RegulatedRealFunctionCarrier namecert obligations`

Proof. Project the accepted carrier. The interval row I, approximation ledger A, uniform tail ledger T, uniform-limit route U, endpoint seal row E, transports H, continuations C, provenance P, and local name N are the only public coordinates.

The source side is finite. The closed interval is read through I using chapter 1448, and each displayed approximant in A is a dyadic step-function row governed by chapter 428.

The limiting side is also routed. The ledger T selects common thresholds for the step approximants, then U consumes that ledger through chapter 463, and only then can the endpoint seal E be read through chapter 1454. The rows H, C, P, N preserve this same route under transport, replay, package provenance, and local naming. □

Theorem 1554.3 (Regulated real function step-approximation coverage)**.** *Every accepted* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *carrier exposes each domain-facing request as a finite approximation read over the* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *row of chapter 1448 and the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *ledger of chapter 428. The selected approximation row is then checked against the uniform tail ledger and passed to the* $\mathsf{UniformLimit}^{\uparrow}$ *route of chapter 463 before any* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *endpoint seal from chapter 1454 is exposed.*

Proof. Begin with a domain-facing consumer request. The continuation row C first sends the request to I, so the request is scoped by the closed interval surface and its displayed endpoint order data.

Next choose the displayed tolerance row named by the request. The ledger A supplies a dyadic step-function approximant for that tolerance, and the dyadic step-function certificate supplies finite cell coverage, endpoint transport, and carried-value reads.

The approximation is not an endpoint by itself. The uniform tail ledger T records the threshold at which later approximants remain uniformly close, and U is the route that repacks the same ledger as a uniform limit. The endpoint seal E can therefore be read only after the closed interval, step approximation, and uniform-limit rows have all been consumed. □

Theorem 1554.4 (Regulated real function uniform-limit seal nonescape)**.** *For an accepted* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *carrier, every endpoint-facing read factors through the finite route*

$$I, A, T \longmapsto U \longmapsto E$$

using the closed interval row of chapter 1448, the dyadic step-function ledger of chapter 428, the uniform-limit route of chapter 463, and the regular located Cauchy real seal of chapter 1454. The packet exports no Riemann integration row, host function-space limit, quotient function equality, open-cover compactness, or ambient Real completeness.

Proof. Inspect the carrier coordinates. The only endpoint-facing row is E, and the continuation data reaches it by replaying the interval, approximation, tail, and uniform-limit rows in order.

Apply Definition 1554.3. Any consumer read first lands on the closed interval row and a displayed dyadic step approximant, then reads the uniform tail ledger before reaching the uniform-limit route.

Finally use the carrier and NameCert obligations of Definition 1554.2. They exhaust the visible rows $I, A, T, U, E, H, C, P, N$, so a consumer has no coordinate for integration, host function-space limits, quotient equality of functions, open-cover compactness, or a completeness theorem outside the displayed seal route. □

Theorem 1554.5 (Regulated real function integration-facing handoff boundary)**.** *Every later Darboux, Riemann, or regulated-integral consumer of an accepted* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *packet must first read the* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *row I, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation ledger A, the uniform tail ledger T, the* $\mathsf{UniformLimit}^{\uparrow}$ *route U, and the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *endpoint seal E, with transport, replay, provenance, and naming carried by H, C, P, N. The packet exports no integration row, host function quotient, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegulatedRealFunctionUp.RegulatedRealFunctionCarrier integration facing handoff boundary`

↪ variant `BEDC.Derived.RegulatedRealFunctionUp.RegulatedRealFunction integration facing handoff boundary`

Proof. Begin with Definition 1554.2. It identifies the accepted carrier rows as exactly $I, A, T, U, E, H, C, P, N$ and places the interval, step-ledger, uniform-limit, endpoint-seal, transport, replay, provenance, and local naming obligations on that same finite packet.

Apply Definition 1554.3. A consumer request first enters through the closed interval row and a displayed dyadic step-function approximation before the uniform tail ledger and uniform-limit route can be read.

Finally apply Definition 1554.4. Endpoint-facing data is exposed only through the route from I, A, T to U and then to E. Since the carrier has no integration, quotient-function, compactness, or ambient-completeness coordinate, later integration-facing chapters may cite this handoff boundary without importing integration into the seed packet. □

Theorem 1554.6 (Regulated real function obligation closure)**.** *For the carrier of Definition 1554.1, the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *obligation surface is closed by carrier admission, dyadic step-approximation coverage, uniform-tail routing,* $\mathsf{UniformLimit}^{\uparrow}$ *nonescape, and* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *endpoint sealing. Equivalently, every accepted packet $G = (I, A, T, U, E, H, C, P, N)$ exposes the route*

$$\mathsf{ClosedUnitInterval}^{\uparrow},\ \mathsf{DyadicStepFunction}^{\uparrow},\ \mathsf{UniformLimit}^{\uparrow},\ \mathsf{RegularLocatedCauchyReal}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{N}$$

and no integration row, host function-space limit, quotient function equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission and local naming are supplied by Definition 1554.2. They identify the visible rows as $I, A, T, U, E, H, C, P, N$ and keep every read on the same finite packet.

Step-approximation coverage is supplied by Definition 1554.3. Each domain-facing read first passes through the closed interval row and a displayed dyadic step-function approximation before the uniform tail ledger is inspected.

The tail and endpoint boundary is supplied by Definition 1554.4. It places endpoint-facing reads on the route from I, A, T to U and then to E, excluding host limits, quotient equality, compactness, and ambient completeness.

Finally use Definition 1554.5. Later Darboux, Riemann, or regulated-integral consumers may consume the displayed interval, approximation, uniform-tail, uniform-limit, and endpoint rows, but they do not add an integration coordinate to the regulated-function packet. These four rows close the obligation surface exactly at the stated boundary. □

Theorem 1554.7 (Regulated real function uniform step route)**.** *Every consumer of an accepted* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *packet reads the* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *row, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation ledger, the* $\mathsf{UniformLimit}^{\uparrow}$ *route, and the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *endpoint seal in that order. The supporting* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows are preserved componentwise along the same route, so no consumer can reach an endpoint, integration-facing handoff, or local certificate row without first replaying the closed-domain and dyadic-step data.*

Proof. Start with the NameCert obligations of Definition 1554.2. They identify the only public rows as $I, A, T, U, E, H, C, P, N$, with I carrying the closed interval, A the dyadic step approximations, T the uniform tail ledger, U the uniform-limit route, and E the endpoint seal.

Apply Definition 1554.3. A consumer request first enters through the $\mathsf{ClosedUnitInterval}^{\uparrow}$ row and then chooses a displayed $\mathsf{DyadicStepFunction}^{\uparrow}$ approximant; the approximation ledger cannot be bypassed because the request is scoped by I and A before T is inspected.

Next use Definition 1554.4. The tail ledger feeds the $\mathsf{UniformLimit}^{\uparrow}$ route, and endpoint-facing data appears only after that route reaches the $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ seal.

Finally combine Definition 1554.5 with Definition 1554.6. Integration-facing reads and local certificate reads consume the same ordered surface, while H, C, P, N preserve transport, replay, provenance, and naming without adding any alternate endpoint or integration coordinate. □

Theorem 1554.8 (Regulated real function scope)**.** *Every Darboux, Riemann, or regulated-integral consumer of an accepted* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *packet must read the scoped route*

$$\mathsf{ClosedUnitInterval}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto \mathsf{RegularLocatedCauchyReal}^{\uparrow} \longmapsto (H, C, P, N)$$

before any integration-facing handoff is available. The route is exactly the interval row I, approximation ledger A, uniform tail ledger T, uniform-limit row U, endpoint seal E, and structural rows H, C, P, N of Definition 1554.1; it does not add an integration row to the packet.

Proof. Begin with the uniform step route. It sends every consumer through the closed-domain row, dyadic step approximation ledger, uniform-limit route, and regular located Cauchy endpoint seal in that order.

Apply the integration-facing handoff boundary. Darboux, Riemann, and regulated-integral consumers are permitted to consume the same displayed route, but the packet exports no integration coordinate, quotient function, compactness row, or ambient Real-completeness object.

Thus the stated route is the scoped consumer surface. The structural rows preserve transport, replay, provenance, and local naming without opening a second handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedRealFunction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegulatedRealFunction}^{\uparrow}$ packet is generated from a ClosedUnitInterval domain row, a finite DyadicStepFunction approximation ledger, a uniform tail and modulus ledger, a UniformLimit route, a RegularLocatedCauchyReal endpoint seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedRealFunction}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1554.1, Definition 1554.2, Definition 1554.3, Definition 1554.4, Definition 1554.5, Definition 1554.6, Definition 1554.7, and Definition 1554.8 export the regulated-function interface over interval, step-approximation, uniform-tail, UniformLimit, endpoint-seal, transport, replay, provenance, naming, and integration-facing consumer rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegulatedRealFunction}\uparrow}$, axiomClean)

Lean target: `BEDC.Derived.RegulatedRealFunctionUp.RegulatedRealFunctionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Riemann integration, host function-space limits, quotient function equality, open-cover compactness, or ambient Real completeness.

Upgrade path: No further theory-axis upgrade is pending for the displayed regulated-function interface; bridge checking remains a separate formal-axis and bridge task.

Chapter 1555

A Concrete Naming Certificate for $\mathsf{UniformCauchyFamily}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformCauchyFamily}^{\uparrow}$ packet records the finite NameCert surface for a displayed finite family of regular Cauchy rational names whose Cauchy control is witnessed by one shared threshold schedule. It sits between the stream-window source of chapter 126, the regular rational readbacks of chapter 301, the dyadic tolerance ledger of chapter 291, the shared-tail criterion of chapter 435, the Cauchy-name consumer surface of chapter 406, and the terminal real-name boundary of chapter 38. The packet names the family-level carrier consumed by uniform-tail and completion routes; it does not select a limit, quotient streams, host real equality, countable choice, or an ambient completeness principle.

Definition 1555.1 (UniformCauchyFamily carrier)**.** *A* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, \sigma, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family row, listing the inspected family members and their visible component identifiers;*
2. *σ is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule that selects one threshold window surface for the whole family;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *family readback row, assigning to each displayed component the rational readbacks inspected at the scheduled windows;*
4. *D is the single* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every componentwise comparison in the family;*
5. *T is an accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row whose shared threshold consumes the same family row, schedule, regular readbacks, and dyadic tolerance ledger;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after the shared-threshold criterion and regular readbacks have been displayed;*
7. *H records componentwise* hsame *transport for the family row, shared schedule, regular readbacks, dyadic ledger, uniform criterion row, real seal, replay row, provenance row, and local naming row;*

8. *C records displayed* Cont *replay from family inspection through the shared schedule, regular readbacks, dyadic tolerance ledger, uniform Cauchy criterion, and real-seal delivery;*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyFamily}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}$ *row.*

The carrier stores no separate threshold per component, no quotient stream, no selected host limit, no host equality of real names, no countable-choice selector, and no ambient completeness field.

Theorem 1555.2 (UniformCauchyFamily NameCert obligations)**.** *For an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier $U = (F, \sigma, R, D, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed family row, shared stream schedule, regular rational readbacks, dyadic tolerance ledger, accepted uniform Cauchy criterion row, real seal, transports, replay route, provenance, and local naming row;*
2. *shared-threshold exactness, requiring every componentwise Cauchy comparison to use the same* $\mathsf{StreamName}^{\uparrow}$ *schedule σ and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D;*
3. *readback compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows in R to be the only rational names consumed by the shared criterion T;*
4. *criterion transport, requiring T to remain attached to F, σ, R, D under the componentwise* hsame *row H and displayed* Cont *replay C;*
5. *real-seal non-escape, requiring the handoff E to be reached only after T and the componentwise readbacks in R have been displayed;*
6. *CauchySequenceSpace compatibility, requiring any* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *consumer to read the family through the same $F, \sigma, R, D, T, E, H, C, P, N$ route;*
7. *ledger exhaustion, stating that no per-component hidden threshold, quotient stream, selected host limit, host real equality, countable-choice selector, or ambient completeness principle is exported.*

Proof. Carrier admission is projection from Definition 1555.1. The tuple displays the finite family row, shared schedule, regular rational readbacks, dyadic tolerance ledger, uniform criterion row, real seal, transport, replay, provenance, and local name.

Shared-threshold exactness is forced by the single schedule coordinate σ and the single tolerance coordinate D. Every componentwise comparison named by the packet must therefore read the same window and tolerance ledger before consulting the uniform criterion row.

Readback compatibility and criterion transport are the role of R, T, H, C. The criterion row T consumes the displayed regular readbacks in R, while H and C move and replay only those rows together with the family and tolerance data.

The real-seal and CauchySequenceSpace boundaries are downstream of the same route. Since E and any $\mathsf{CauchySequenceSpace}^{\uparrow}$ consumer are named after $F, \sigma, R, D, T, H, C, P, N$ have been displayed, neither can import a selected limit, a quotient stream, host real equality, countable choice, or an ambient completeness principle. □

Theorem 1555.3 (UniformCauchyFamily common-window extraction)**.** *Let $U = (F, \sigma, R, D, T, E, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *packet. Every componentwise Cauchy comparison extracted from U factors through the same finite route*

$$F, \sigma, R, D, T, H, C, P, N.$$

In particular, the extracted comparison cannot replace the shared $\mathsf{StreamName}^{\uparrow}$ *schedule by a component-specific threshold, cannot replace the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks by an independent rational stream, and cannot replace the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger by a hidden host tolerance.*

Proof. Project the accepted carrier. The family row F has only one schedule row σ, one regular readback family R, and one dyadic tolerance ledger D.

Use the shared-threshold and readback clauses of Definition 1555.2. They force each componentwise comparison to read σ, R, D before it reaches the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row T.

Transport and replay do not add coordinates. The rows H, C, P, N preserve the same finite route, so the extracted comparison has no access to a component-specific threshold, independent rational stream, or hidden host tolerance. □

Theorem 1555.4 (UniformCauchyFamily RegSeqRat handoff)**.** *Let* $U = (F, \sigma, R, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *packet. Every shared threshold window extracted from* U *produces a rowwise* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for each displayed family member before any* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal consumer may read the family. The handoff uses the same family row* F*, shared* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*, dyadic tolerance ledger* D*, accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row* T*, and transport route* H, C, P, N*.*

Proof. Project the accepted carrier. The finite family row F lists the displayed members, the shared schedule σ names the threshold window, and the row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ family readback coordinate.

Apply common-window extraction from Definition 1555.3. Each componentwise comparison factors through $F, \sigma, R, D, T, H, C, P, N$, so the same extracted window supplies the regular rational readback row for every displayed member.

Use shared-threshold transport from Definition 1555.6. The rows H and C move the schedule σ and tolerance ledger D together with F, R, T, preventing a transported component from acquiring a hidden threshold or an independent rational stream.

The downstream boundary is therefore ordered. A $\mathsf{CauchySequenceSpace}^{\uparrow}$ consumer or a $\mathsf{Real}^{\uparrow}$ seal consumer may read the family only after the rowwise $\mathsf{RegSeqRat}^{\uparrow}$ readbacks in R have been displayed over the shared window and criterion route. □

Theorem 1555.5 (UniformCauchyFamily Real seal boundary)**.** *For an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier* $U = (F, \sigma, R, D, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read and every completion-facing consumer read of the family reaches* E *only after passing through the finite family row* F*, shared* $\mathsf{StreamName}^{\uparrow}$ *schedule* σ*,* $\mathsf{RegSeqRat}^{\uparrow}$ *family readbacks* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row* T*. The boundary exports no quotient stream, selected host limit, host real equality, countable choice, or ambient completeness row.*

Proof. Begin with a real-seal or completion-facing read of U. By the carrier definition, the only terminal seal coordinate is E.

The NameCert obligations place E downstream of the shared-threshold criterion T and the regular readback row R. Hence the read must first traverse F, σ, R, D, T.

Finally, H, C, P, N preserve the same displayed route for equivalent packets and consumer replays. The carrier lists no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient completeness coordinate, so the real seal boundary is exhausted by the stated finite packet. □

Theorem 1555.6 (UniformCauchyFamily shared-threshold transport)**.** *Let $U = (F, \sigma, R, D, T, E, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier. The transport rows H and C move the single shared* $\mathsf{StreamName}^{\uparrow}$ *schedule σ and the single* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D together with the displayed family row F and the accepted criterion row T. Thus a transported component cannot acquire its own hidden threshold, tolerance ledger, or per-component Cauchy witness outside the route*

$$F, \sigma, R, D, T, H, C, P, N.$$

Proof. Project the carrier coordinates. The only schedule coordinate available to the family is σ, and the only tolerance ledger coordinate available to componentwise comparisons is D.

The transport row H is componentwise hsame over the displayed family row, shared schedule, regular readbacks, dyadic ledger, criterion row, real seal, replay row, provenance row, and local naming row. It therefore transports the criterion surface with σ and D still attached.

The replay row C follows the same displayed route from family inspection through the shared schedule, regular readbacks, dyadic tolerance ledger, uniform criterion, and real seal. Since neither H nor C adds a threshold coordinate, every transported component remains bound to the shared schedule and ledger. □

Theorem 1555.7 (UniformCauchyFamily Real seal factorization)**.** *For an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier $U = (F, \sigma, R, D, T, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$ *seal read factors through the finite packet*

$$F, \sigma, R, D, T, E, H, C, P, N.$$

The seal coordinate E is therefore readable only as the terminal handoff of the displayed family row, shared $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *family readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accepted* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row, transport rows, replay row, provenance, and local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}$ *row.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ seal read of U. By the carrier definition, the only seal coordinate is E, and it is listed after the family row, schedule, readbacks, tolerance ledger, and accepted criterion row.

The NameCert obligations require real-seal non-escape. Thus the read cannot reach E before the shared criterion T and the regular readback family R have been displayed over F, σ, D.

Transport and replay preserve the factorization. The rows H and C move the same displayed data, while P and N record provenance and local naming for the packet; no extra seal source is available. □

Theorem 1555.8 (UniformCauchyFamily CauchySequenceSpace compatibility)**.** *Let $U = (F, \sigma, R, D, T, E, H, C, P$ be an accepted* $\mathsf{UniformCauchyFamily}^{\uparrow}$ *carrier. Any* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *consumer of the family receives exactly the finite route*

$$F, \sigma, R, D, T, E, H, C, P, N$$

through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *surfaces. Such a consumer cannot import quotient streams, selected limits, countable choice, host real equality, or ambient completeness.*

Proof. Read the consumer through the compatibility clause of Definition 1555.2. The clause names the same carrier route $F, \sigma, R, D, T, E, H, C, P, N$ for every $\mathsf{CauchySequenceSpace}^{\uparrow}$ consumer.

The upstream rows are finite and displayed. They pass through the $\mathsf{StreamName}^{\uparrow}$ schedule σ, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks R, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, the accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row T, and the $\mathsf{Real}^{\uparrow}$ handoff E.

The exclusion follows from ledger exhaustion. Since the carrier has no quotient-stream coordinate, selected-limit coordinate, countable-choice selector, host real equality, or ambient completeness row, the consumer cannot import any of those structures. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1555.1, Definition 1555.2, Definition 1555.3, Definition 1555.5, Definition 1555.6, Definition 1555.7, and Definition 1555.8 seed the shared-threshold family packet over StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerances, UniformCauchyCriterion rows, CauchySequenceSpace consumers, and Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close per-component hidden thresholds, quotient streams, selected host limits, host real equality, countable choice, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, shared-threshold exactness, readback compatibility, criterion transport, Real seal non-escape, CauchySequenceSpace compatibility, and ledger exhaustion for BEDC.Derived.UniformCauchyF

Constructive story: A $\mathsf{UniformCauchyFamily}^{\uparrow}$ packet is generated from a finite family row, a shared $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ family readbacks, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, an accepted $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1556

A Concrete Naming Certificate for $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ packet records the finite NameCert surface for constructive metric completion as a route through located metric data, regular Cauchy names, dyadic tolerances, filter bases, separated equality, dense embedding, and uniformly continuous extension. It reads the metric-completion packet of chapter 1506, the located-metric source of chapter 1533, the Cauchy-filter basis row of the completion interface, the separated regular-Cauchy route of chapter 1513, the embedding handoff of chapter 1512, the extension surface of chapter 769, and the naming spine of chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a finite completion certificate for displayed Cauchy-filter reads; it is not a quotient construction, a selected host limit, a countable-choice schedule, an open-cover compactness principle, or a full completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1556.1 (Constructive metric completion carrier)**.** *A* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, A, D, F, E, S, U, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{LocatedMetric}^{\uparrow}$ *row, including the displayed* $\mathsf{Met}^{\uparrow}$ *distance witness and the finite locatedness route used by completion-facing consumers.*
2. *A is the regular Cauchy name family row. It stores the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks by which a completion point is inspected at finite precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the windows of A, including the monotone tolerance requests used to compare finite prefixes.*
4. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row, recording the Cauchy-filter neighborhoods generated by the same regular names and dyadic tolerances.*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that embeds source observations densely into the displayed completion route.*
6. *S is the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *row, exposing separated completion equality only through the displayed regular-Cauchy and zero-distance comparison route.*
7. *U is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff row for uniformly continuous consumers of the dense source.*

8. *H records componentwise* hsame *transport for the located source, regular Cauchy names, dyadic ledger, Cauchy filter basis, completion embedding, separated equality row, extension handoff, replay row, provenance row, and local naming row.*

9. *C records the displayed* Cont *replay from located-metric observations to regular Cauchy windows, dyadic tolerances, Cauchy-filter neighborhoods, embedding reads, separated comparisons, and uniformly continuous extension reads.*

10. *P is the* Pkg *provenance over* ConstructiveMetricCompletion$^{\uparrow}$, LocatedMetric$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, RegularCauchyCompleti UniformContinuousExtension$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ConstructiveMetricCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient representatives, host-selected limits, countable choice, open-cover compactness, detached Real$^{\uparrow}$ *completeness, or an ambient complete metric space.*

Theorem 1556.2 (ConstructiveMetricCompletion NameCert obligations)**.** *For an accepted* ConstructiveMetricCom *carrier $K = (X, A, D, F, E, S, U, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveMetricCompletion}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed located metric source, regular Cauchy name family, dyadic tolerance ledger, Cauchy filter basis, completion embedding, separated equality row, uniformly continuous extension handoff, transports, replay route, provenance, and local naming row;*

2. *classifier stability, requiring two accepted completion reads to be compared only through the same ordered route $X, A, D, F, E, S, U, H, C, P, N$ after componentwise* hsame *transport;*

3. *embedding coverage, requiring each source observation used by the completion packet to enter through the dense* CompletionEmbedding$^{\uparrow}$ *row E;*

4. *separatedness exactness, requiring every equality judgement between completion reads to pass through S and hence through the displayed regular-Cauchy separated route;*

5. *universal handoff, requiring each uniformly continuous consumer of the dense source to read the completion point through F, S, U before any exported value is named;*

6. *transport replay, requiring H and C to preserve the located source, regular names, dyadic tolerances, filter basis, embedding row, separated row, and extension row as one packet;*

7. *ledger non-escape, stating that no quotient representative, host-selected limit, countable-choice selector, open-cover compactness principle, detached* Real$^{\uparrow}$ *completeness theorem, or ambient complete metric space is exported.*

Proof. Carrier admission is projection from Definition 1556.1. The tuple displays the located source, regular Cauchy name family, dyadic ledger, Cauchy-filter basis, completion embedding, separated equality row, extension handoff, and structural support rows.

Classifier stability is componentwise. The transport row H moves each displayed coordinate in the same order, and the replay row C forces any accepted comparison to traverse the located source, regular names, dyadic tolerances, filter basis, embedding row, separated row, and extension row before the local name is read.

Embedding coverage is the role of E. A source observation becomes completion-facing data only through the dense embedding row, so no selected representative or host completion point can be substituted for it.

Separatedness exactness is isolated in S. Equality of completion reads is admitted only after the regular Cauchy windows and dyadic tolerances have been displayed, and only through the separated completion route named by that row.

The universal handoff is carried by U after F and S are present. The provenance row P and local naming row N record no further coordinate, so the excluded quotient, choice, compactness, host-limit, real-completeness, and ambient-completeness principles cannot be read from the packet. □

Theorem 1556.3 (ConstructiveMetricCompletion universal handoff)**.** *Let $K = (X, A, D, F, E, S, U, H, C, P, N)$ be an accepted* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *packet. Any uniformly continuous consumer of the dense source reads a completed point only through the displayed Cauchy-filter basis F, the separated completion equality row S, and the uniformly continuous extension row U, after the located source X, regular Cauchy name family A, dyadic tolerance ledger D, and completion embedding row E have been displayed.*

Thus every such consumer factors through the finite route

$$X, A, D, E \longmapsto F \longmapsto S \longmapsto U \longmapsto H, C, P, N,$$

not through quotient representatives, host-selected limits, countable choice, open-cover compactness, detached $\mathsf{Real}^{\uparrow}$ *completeness, or an ambient complete metric space.*

Proof. Begin with the dense source. The located metric row X supplies finite distance and locatedness reads, while A and D supply the scheduled regular Cauchy windows and dyadic tolerance ledger.

The source observation becomes completion-facing only through E. This places the dense embedding row before any consumer can inspect a completed point.

The Cauchy-filter basis F is then the unique neighborhood route available to the consumer. It is generated from the same regular names and dyadic tolerances, so the consumer cannot replace it by a selected representative or host limit.

Separated comparison is read next through S. The uniformly continuous extension row U may be consumed only after that separated route is available, ensuring that the extension depends on the displayed Cauchy-filter and separated completion rows.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient representatives, host-selected limits, countable choice, open-cover compactness, detached $\mathsf{Real}^{\uparrow}$ completeness, or ambient completeness, every uniformly continuous consumer factors through the stated handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveMetricCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1556.1, Definition 1556.2, and Definition 1556.3 seed the finite Bishop-style constructive metric completion packet over located metrics, regular Cauchy names, dyadic tolerances, Cauchy filters, separated equality, dense embedding, and uniformly continuous extension.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient representatives, host-selected limits, countable choice, open-cover compactness, full $\mathsf{Real}^{\uparrow}$ completeness, ambient complete metric spaces, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, embedding coverage, separatedness exactness, universal handoff, transport replay, tolerance-ledger non-escape, and refusal rows for BEDC.Derived.ConstructiveMetricComp

Constructive story: A $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ packet is generated from a located metric source, regular Cauchy name family, dyadic tolerance ledger, Cauchy filter basis, dense completion embedding, separated completion equality row, uniformly continuous extension handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 1557

A Concrete Naming Certificate for $\mathsf{RegulatedIntegral}^{\uparrow}$

Following Riemann, Darboux, and Bishop–Bridges constructive analysis, the packet names the integral of a regulated real function by finite dyadic step primitives and Darboux/Riemann compatibility rows, not by host function-space limits or measure completion.

The packet consumes the regulated-function approximation surface of chapter 1554, the finite step primitive of chapter 428, the Darboux gap route of chapter 1460, and the tagged-mesh boundary of chapter 1526. It is a BEDC-native consumer target for Real and Completion work: every integral read is routed through displayed step sums, compatibility ledgers, transport rows, replay rows, provenance, and a local name.

Definition 1557.1 (Regulated integral NameCert surface)**.** *A* $\mathsf{RegulatedIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, G, A, S, D, R, E, H, C, P, N)$$

with the following displayed coordinates.

1. *I is the interval row naming the integration domain, endpoint order data, and cell-facing boundary reads.*
2. *G is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row, read through chapter 1554, naming the regulated integrand by finite dyadic step approximations and a uniform tail ledger.*
3. *A is the finite dyadic step approximation ledger selected from G at displayed tolerances.*
4. *S is the step-integral primitive, computing finite sums from the cells and carried values of the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *rows in A.*
5. *D is the Darboux/Riemann compatibility row, recording that the same step approximation route fits the upper-lower gap boundary of chapter 1460 and the tagged-mesh surface of chapter 1526.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached from the displayed finite sums and compatibility ledger.*
7. *E records the error and modulus bounds connecting the chosen step rows, the regulated-function tail ledger, and the real-facing handoff.*
8. *H records componentwise* hsame *transport for interval, integrand, approximation, step-integral, compatibility, error, handoff, provenance, and local naming rows.*

9. *C records the displayed* Cont *replay from regulated-function reads to step approximants, from step approximants to finite sums, from compatibility rows to the Darboux/Riemann boundary, and from error bounds to the real handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, A, S, D, R, E, H, C, P, N$. It has no coordinate for a host function quotient, a measure completion, a selected limiting tag family, open-cover compactness, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralTasteGate single carrier alignment`

Theorem 1557.2 (Regulated integral step primitive)**.** *Every accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet exposes its integral through finite* $\mathsf{DyadicStepFunction}^{\uparrow}$ *primitive sums. A consumer may read the interval row I, the regulated-function row G, the selected approximation ledger A, the finite sum primitive S, and the recorded error bounds E, but it cannot bypass these rows through a host integral or a hidden limiting functional.*

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralStepPrimitive`

Proof. Project the carrier of Definition 1557.1. The integrand-facing row is G, and chapter 1554 exposes G by finite dyadic step approximations rather than by a host function-space object.

The selected approximation ledger A chooses the displayed $\mathsf{DyadicStepFunction}^{\uparrow}$ rows at the tolerances recorded by E. The primitive S computes the corresponding step integrals from finite cells, endpoint transports, and carried values supplied by chapter 428.

The continuation row C can replay only this route: interval read, regulated-function approximation, step ledger, finite sum, and error ledger. Since no other integral coordinate is part of J, every accepted integral read factors through the displayed step primitive. □

Theorem 1557.3 (Regulated integral Darboux–Riemann compatibility)**.** *The regulated step approximation route of an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet is compatible with the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *gap boundary and the* $\mathsf{RiemannIntegral}^{\uparrow}$ *tagged-mesh surface. The compatibility is recorded by the displayed row D and is consumed before the* $\mathsf{Real}^{\uparrow}$ *handoff row R becomes available.*

Proof. Start from the finite step primitive of Definition 1557.2. The same approximation ledger A supplies the lower and upper step data needed by the Darboux gap row and the tagged-cell data needed by the Riemann mesh surface.

The compatibility row D records both readings. Through chapter 1460, it checks the visible upper-lower gap and its modulus; through chapter 1526, it checks that the tagged-mesh sums are read against the same displayed cells and carried values.

Only after this row is present can the packet expose R. The transport row H and continuation row C preserve the same order of consumption, so the Darboux and Riemann boundaries share the regulated step ledger rather than introducing separate host analytic data. □

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralDarbouxRiemannCompatibility`

Theorem 1557.4 (Regulated integral visible step handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet first consumes the visible ledger*

$$I, G, A, S, E.$$

Thus the interval row, regulated-function approximation row, selected finite step ledger, step-integral primitive, and error/modulus row are present before the packet can export any real-facing integral value.

Proof. Project the carrier of Definition 1557.1. The rows I, G, A, S, E are the only coordinates that name the interval, choose the finite regulated approximation, compute the step sum, and record the error budget.

Apply Definition 1557.2. It routes every accepted integral read through finite $\mathsf{DyadicStepFunction}^{\uparrow}$ primitive sums and the displayed error ledger, so no real-facing read can begin with a hidden integral functional.

The continuation row C preserves this order when the handoff proceeds toward D and R. Hence every public $\mathsf{Real}^{\uparrow}$ read first consumes I, G, A, S, E. □

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralVisibleStepHandoff`

Theorem 1557.5 (Regulated integral dyadic step approximation factorization)**.** *Every accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *read factors its approximation row through the finite* $\mathsf{DyadicStepApproximation}^{\uparrow}$ *route before the Darboux/Riemann compatibility row and the* $\mathsf{Real}^{\uparrow}$ *handoff are visible. In the carrier notation of Definition 1557.1, the rows* I, G, A, S, E *expose the regulated-function source, dyadic step approximation ledger, finite step-sum primitive, and error budget before* D *and* R *can be consumed.*

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralDyadicStepApproximationFactorization`

Proof. Start from the visible step handoff. It places the interval row, regulated-function row, selected finite approximation ledger, step primitive, and error row before any Real-facing integral value is exported.

Read the approximation ledger through the dyadic step approximation route. The ledger A is the finite witness that the regulated-function row has selected displayed dyadic step cells, values, and tolerances, so the step primitive S can compute only from that finite route.

Only after this route is present can D compare the same step data with the Darboux and Riemann boundaries, and only after D can R expose the Real handoff. Thus the accepted read factors through $\mathsf{DyadicStepApproximation}^{\uparrow}$ before compatibility or endpoint data is public. □

Theorem 1557.6 (Regulated integral convergence consumer boundary)**.** *Every* $\mathsf{RegulatedConvergence}^{\uparrow}$ *endpoint or integral-facing consumer that reaches an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet must first display the regulated-function row, uniform-tail row, finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert row carried by the regulated-integral route. The consumer receives no host convergence theorem, measure completion, quotient function equality, selected limit object, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralConvergenceConsumerBoundary`

Proof. Project the regulated-integral carrier. Its integrand-facing row G supplies the regulated-function source, while A and E record the finite windows and uniform error data that a regulated-convergence consumer can inspect.

The endpoint read then passes through the regular rational and Real-facing support already used by the public regulated-integral interface. The finite step primitive and Darboux/Riemann compatibility row keep the endpoint behind displayed step sums before the Real seal is public.

Finally read H, C, P, N. These rows transport, replay, source, and locally name the same finite route, so a $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer can reach $\mathsf{RegulatedIntegral}^{\uparrow}$ only through the displayed boundary and not through hidden convergence, quotient, measure, selected-limit, or completeness data. □

Theorem 1557.7 (Regulated integral Real completion route)**.** *The* $\mathsf{Real}^{\uparrow}$ *handoff row* R *of an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet remains behind the Darboux/Riemann compatibility row* D*. A downstream Real or Completion consumer may use* R *only through the route*

$$I, G, A, S, E \longmapsto D \longmapsto R, H, C, P, N,$$

and the route imports neither measure completion, full $\mathsf{Real}^{\uparrow}$ *completeness, quotient function equality, nor selected limiting tag families.*

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralRealCompletionRoute`

Proof. First use Definition 1557.4 to expose the interval, regulated-function, finite approximation, step primitive, and error rows. These rows provide the only step-sum data available to the packet.

Apply Definition 1557.3. The compatibility row D checks that the same step ledger fits the Darboux gap boundary and the Riemann tagged-mesh surface before R is available.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for measure completion, full Real completeness, quotient function equality, or selected limiting tags, the Real handoff remains inside the displayed Darboux/Riemann route. □

Theorem 1557.8 (Regulated integral nonescape)**.** *No accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet exports a host function quotient, measure completion, selected limiting tag family, open-cover compactness witness, or ambient Real-completeness principle. Every consumer read is confined to the displayed ledger*

$$I, G, A, S, D, R, E, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralCarrier nonescape`

Proof. Inspect the carrier coordinates in Definition 1557.1. They display the interval row, regulated-function row, finite approximation ledger, step-sum primitive, Darboux/Riemann compatibility row, real handoff, error ledger, transports, continuations, provenance, and local naming row.

Apply Definition 1557.2 to route every integral read through finite dyadic step sums. Apply Definition 1557.3 to route the real-facing read through the visible Darboux gap and Riemann tagged-mesh compatibility row.

The excluded objects have no carrier coordinate and no continuation route. Thus a downstream Real or Completion consumer receives only the displayed ledger and cannot access a quotient of host functions, a measure-completion object, selected limiting tags, open-cover compactness, or a completeness principle outside the packet. □

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralNonescape`

Theorem 1557.9 (Regulated integral kernel-source scope)**.** *Every regulated-integral read from an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet is scoped to the interval row,* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row,* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation rows,* $\mathsf{DarbouxIntegral}^{\uparrow}$ *and* $\mathsf{RiemannIntegral}^{\uparrow}$ *compatibility row,*

$\mathsf{Real}^{\uparrow}$ *handoff row, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. Equivalently, every read remains inside*

$$I, G, A, S, D, R, E, H, C, P, N$$

and no consumer can replace the displayed step-primitive source by a host function quotient, measure completion, selected limiting tag family, open-cover compactness witness, or ambient Real-completeness principle.

Proof. Start with the step primitive. By Definition 1557.2, each accepted integral read exposes the interval row I, the regulated-function row G, the selected dyadic step approximation ledger A, the finite step-sum primitive S, and the error row E before any real-facing handoff is available.

Apply Darboux–Riemann compatibility. The row D records that the same displayed step approximation route fits the Darboux gap boundary and the Riemann tagged-mesh surface, so the compatibility source is a kernel row rather than a host integration theorem.

Now use the visible step handoff and Real completion route. These place R behind I, G, A, S, E and then behind D, so a $\mathsf{Real}^{\uparrow}$ consumer receives only the displayed finite-sum route and its compatibility ledger.

Finally apply nonescape. The structural rows H, C, P, N transport, replay, source, and name the same packet, and the carrier has no coordinate for the excluded function-space, measure, tag-selection, compactness, or completeness objects. Therefore the regulated-integral read is scoped exactly to the listed kernel-source rows. □

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralKernelSourceScope`

Theorem 1557.10 (Regulated integral public interface)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$*, Completion, or analysis consumer of an accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet receives only the interval row, regulated-function row, dyadic step approximation rows, Darboux/Riemann compatibility row,* $\mathsf{Real}^{\uparrow}$ *handoff row, error and modulus rows, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row:*

$$I, G, A, S, D, R, E, H, C, P, N.$$

The public interface exports no measure completion, quotient function equality, selected limiting tag family, open-cover compactness witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegulatedIntegralUp.RegulatedIntegralPublicInterface`

Proof. Use the step primitive, Definition 1557.2, and the visible step handoff, Definition 1557.4. Together they expose the interval row I, regulated-function row G, dyadic step ledger A, finite step primitive S, and error row E before any real-facing consumer may read the packet.

Apply Darboux–Riemann compatibility, Definition 1557.3. The compatibility row D is the only bridge from the displayed step ledger to the Darboux gap boundary and the Riemann tagged-mesh surface.

Now use the Real completion route, Definition 1557.7. It places R behind I, G, A, S, E and then behind D, so Real and Completion consumers receive the handoff only through the same finite route.

Finally apply nonescape and kernel-source scope, Definition 1557.8 and Definition 1557.9. The structural rows H, C, P, N transport, replay, source, and name the displayed ledger, and the excluded analytic objects have no carrier coordinate. Thus the listed rows form the entire public interface. □

Theorem 1557.11 (Regulated integral mature treatment)**.** *The public regulated-integral interface of Definition 1557.10 is a mature consumer route over* $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Every accepted* $\mathsf{RegulatedIntegral}^{\uparrow}$ *packet exposes the interval row, regulated-function row, dyadic step approximation rows, Darboux/Riemann compatibility row,* $\mathsf{Real}^{\uparrow}$ *handoff row, error and modulus rows, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any downstream integral read is available. The route still refuses measure completion, quotient function equality, open-cover compactness, full* $\mathsf{Real}^{\uparrow}$ *completeness, and selected limiting tag families.*

Proof. Start from the public interface. It already lists exactly the rows $I, G, A, S, D, R, E, H, C, P, N$ exported to downstream Real, Completion, and analysis consumers.

Read those rows through their mature inputs. The regulated-function row supplies the displayed integrand, the dyadic step rows supply finite approximants, the Darboux and Riemann rows supply the compatibility boundary, and the Real row supplies the terminal handoff without adding an ambient function space.

Finally apply kernel-source scope and nonescape. Those results keep transport, replay, provenance, and local naming attached to the same finite packet, while the excluded analytic objects remain absent carrier coordinates. Hence the public interface is the mature consumer treatment of the regulated-integral packet and not a measure-theoretic or quotient-function completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedIntegral}^{\uparrow}$ packet is generated from an interval row, a $\mathsf{RegulatedRealFunction}^{\uparrow}$ integrand row, a finite $\mathsf{DyadicStepFunction}^{\uparrow}$ approximation ledger, a step-integral primitive, a Darboux/Riemann compatibility row, a $\mathsf{Real}^{\uparrow}$ handoff, error and modulus bounds, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedIntegral}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1557.10 and Definition 1557.11 bind Definition 1557.1, Definition 1557.2, Definition 1557.3, Definition 1557.7, Definition 1557.8, and Definition 1557.9 to the interval, regulated-function, dyadic-step, Darboux/Riemann, Real handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic integration, quotient function equality, open-cover compactness, full Real completeness, Lean verification, or bridge checking.

Upgrade path: Bridge-grade closure requires Lean verification, axiom-purity certification, and a checked standard bridge for the public regulated-integral interface.

Chapter 1558

A Concrete Naming Certificate for $\mathsf{CauchyFilterLimit}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley terminology of filter bases, the $\mathsf{CauchyFilterLimit}^{\uparrow}$ packet names the finite limit boundary reached after a generated Cauchy-filter basis has been consumed and before a $\mathsf{Real}^{\uparrow}$ seal can be read. It sits downstream of chapter 1501 and chapter 333, reads the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the dyadic tolerance ledger of chapter 291, and exposes the seal boundary of chapter 38 only after those rows have been replayed. The chapter records a finite limit packet, not a quotient of streams, not a selected host limit, and not an ambient completeness principle.

Definition 1558.1 (CauchyFilterLimit carrier)**.** *A* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row. It names the finite basis elements and common-refinement data from which the limit request starts.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation row. It consumes the generated basis and exposes only the saturated finite-filter surface.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the saturated filter request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning the displayed stream windows into regular rational observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the rational observations at the requested limit scale.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, F, S, R, D have been replayed.*
7. *H records componentwise* hsame *transport for the basis, filter, window, readback, tolerance, seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a basis request through saturation, finite windows, regular readback, dyadic tolerance, and then the seal row.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterLimit}^\uparrow}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected host limit, host real equality, countable-choice selection, or ambient completeness of $\mathsf{Real}^\uparrow$.

Lean4 (definition) `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier`
Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitTasteGate single carrier alignment`

Theorem 1558.2 (CauchyFilterLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterLimit}^\uparrow$ *carrier* $L = (B, F, S, R, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterLimit}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed basis generator, saturated filter, finite stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, continuation, provenance, and local naming row;*
2. *basis consumption and filter saturation, requiring the B row to be consumed by F before any limit-facing read can inspect stream, rational, tolerance, or seal data;*
3. *regular readback, requiring each limit-scale observation to pass through S and R before the tolerance ledger D is consulted;*
4. *Real seal non-escape, requiring the seal row A to remain downstream of B, F, S, R, D;*
5. *ledger refusal, excluding quotient streams, selected host limits, host real equality, countable-choice selectors, and ambient* $\mathsf{Real}^\uparrow$ *completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier namecert obligations`

Proof. Project the carrier of Definition 1558.1. The projection exposes exactly the generator row, saturation row, stream-window row, regular readback row, dyadic tolerance ledger, Real seal, and structural rows H, C, P, N.

Basis consumption is forced by the row order. The continuation row C routes a limit request from B into F before the request can reach the finite windows S, regular readback R, tolerance ledger D, or seal row A.

The regular readback and seal conditions are then internal to the same packet. Transport by H preserves the route, while P and N package and name only that displayed route, so no separate quotient, host-limit, choice, or completeness coordinate remains available. □

Theorem 1558.3 (CauchyFilterLimit regular seal handoff)**.** *Let* $L = (B, F, S, R, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyFilterLimit}^\uparrow$ *packet. Every* $\mathsf{Real}^\uparrow$*-facing limit read factors through the finite route*

$$B, F, S, R, D \longmapsto A \longmapsto H, C, P, N.$$

Thus the limit seal is read only after the $\mathsf{CauchyFilterBasis}^\uparrow$ *generator, the* $\mathsf{CauchyFilter}^\uparrow$ *saturation row, the* $\mathsf{StreamName}^\uparrow$ *windows, the* $\mathsf{RegSeqRat}^\uparrow$ *readback, and the* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledger have been displayed. The route contains no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier regular seal handoff`

Proof. Begin with a Real-facing consumer of the packet. Carrier admission gives access to the seal row A, but only as a row downstream of B, F, S, R, D.

Replay the continuation row C. The request first consumes the displayed Cauchy-filter basis generator B, then the saturated filter row F, then the finite stream windows S, the regular rational readback R, and the dyadic tolerance ledger D.

Only after this replay can the seal row A be read. The transport, provenance, and local naming rows record the same dependency order, so the packet exports no direct route through quotient streams, selected host limits, host real equality, countable choice, or ambient completeness. □

Theorem 1558.4 (CauchyFilterLimit directed-net handoff)**.** *Let $L = (B, F, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *packet. Any directed-net completion consumer can read the packet only after the* $\mathsf{DirectedSet}^{\uparrow}$ *index substrate, the* $\mathsf{CauchyNet}^{\uparrow}$ *tail evidence, and the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *boundary row have been displayed. The handoff preserves the finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route of Definition 1558.3, and exports no quotient filter, selected subnet, countable choice, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier directed net handoff`

Proof. Begin with the regular seal handoff of Definition 1558.3. It fixes the route from the Cauchy-filter basis and saturation rows through stream windows, regular rational readback, dyadic tolerance, and the Real seal.

For a directed-net completion consumer, the filter basis is first read through the displayed $\mathsf{DirectedSet}^{\uparrow}$ substrate. The same finite request is then routed through the $\mathsf{CauchyNet}^{\uparrow}$ tail row, so the consumer sees directed windows rather than a quotient filter.

Apply the $\mathsf{MooreSmithConvergence}^{\uparrow}$ boundary row to the displayed directed windows and tail evidence. This places the directed-net read downstream of the same StreamName, RegSeqRat, DyadicRatCore, and Real route, while omitting selected subnets, countable-choice selectors, and ambient completeness. □

Theorem 1558.5 (CauchyFilterLimit basis saturation obligation)**.** *For every accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *carrier $L = (B, F, S, R, D, A, H, C, P, N)$ of Definition 1558.1, the local saturation obligation requires the basis row B and saturated filter row F to be admitted before any* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, or* $\mathsf{Real}^{\uparrow}$ *seal A is read. The same obligation is transported by H, replayed by C, sourced by P, and named by N, and it refuses quotient stream equality, selected host limits, host real equality, countable-choice selectors, and ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier basis saturation obligation`

Proof. Project the carrier rows listed in Definition 1558.1. The basis generator B and saturated filter row F are the only rows that can expose the limit request before window, readback, tolerance, and seal data appear.

Replay the continuation row C. Its route sends the request from B to F before it reaches S, R, D, A, while H preserves the same order under componentwise transport.

The provenance row P and local naming row N package only this finite route. Since no carrier coordinate stores quotient stream equality, selected host limits, host real equality, countable-choice selectors, or ambient completeness, the saturation obligation cannot export those principles. □

Theorem 1558.6 (CauchyFilterLimit regular readback obligation)**.** *For every accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *carrier $L = (B, F, S, R, D, A, H, C, P, N)$, a* $\mathsf{Real}^{\uparrow}$*-facing limit read is admitted only after the*

$\mathsf{StreamName}^{\uparrow}$ *row S has displayed the finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R has supplied regular rational readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D has supplied the tolerance ledger. The admission is invariant under H, replayed by C, sourced by P, and named by N; it does not identify quotient streams, choose a host limit, assert host real equality, use countable choice, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier regular readback obligation`

Proof. Begin after the basis-saturation obligation of Definition 1558.5. The remaining limit-facing read must pass through the rows S, R, D, A in the carrier.

The row S exposes finite windows only. The row R turns those windows into regular rational observations, and the row D records the dyadic tolerance comparison before the seal row A can be read.

Rows H, C, P, N preserve this readback order under transport, replay, provenance, and naming. The refused principles are absent from the tuple, so the regular readback obligation cannot be consumed as quotient stream equality, a selected host limit, host real equality, countable choice, or ambient Real completeness. □

Theorem 1558.7 (CauchyFilterLimit ledger refusal obligation)**.** *For every accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *carrier $L = (B, F, S, R, D, A, H, C, P, N)$, the local ledger refusal is part of the NameCert surface: any consumer that reads beyond the displayed $B, F, S, R, D, A, H, C, P, N$ rows is rejected. In particular, the carrier refuses quotient stream equality, selected host limits, host real equality, countable-choice selectors, and ambient Real completeness while retaining the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *route.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier ledger refusal obligation`

Proof. Use Definition 1558.2. It states that the local NameCert surface is exhausted by carrier admission, basis consumption, filter saturation, regular readback, Real seal non-escape, and ledger refusal.

The first two obligation theorems identify the positive route: B, F admit and saturate the filter request, while S, R, D, A expose finite windows, regular readback, dyadic tolerance, and the Real seal.

It remains to record what the packet cannot export. Because H, C, P, N transport, replay, source, and name only the displayed route, any read asking for quotient stream equality, selected host limits, host real equality, countable-choice selectors, or ambient completeness has no carrier coordinate and is refused by the local ledger. □

Theorem 1558.8 (CauchyFilterLimit directed-net coverage)**.** *Let $L = (B, F, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *packet. Every directed-net completion consumer of L must pass through the displayed route*

$$B, F, S, R, D, A, H, C, P, N.$$

In particular, the consumer first exposes the $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator and* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation rows, then the shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then the* $\mathsf{Real}^{\uparrow}$ *seal and structural rows. The coverage claim does not export a quotient filter, selected subnet, countable-choice selector, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier directed net coverage`

Proof. Start with the directed-net handoff of Definition 1558.4. It admits a directed-net consumer only after the displayed index and tail evidence have been routed through the same finite filter-limit packet.

Apply the regular seal handoff. Theorem 1558.3 fixes the route from the Cauchy-filter basis and saturation rows through the finite stream windows, regular rational readback, dyadic tolerance ledger, and Real seal.

Finally attach the structural rows. Transport, continuation, provenance, and local naming only preserve the displayed route, so a directed-net completion consumer has no carrier coordinate from which to bypass $B, F, S, R, D, A, H, C, P, N$. □

Theorem 1558.9 (CauchyNetLimit filter comparison)**.** *The* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *and* $\mathsf{CauchyNetLimit}^{\uparrow}$ *packets can be compared along their shared finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows. The comparison records that both routes expose finite windows, regular rational readbacks, dyadic tolerance ledgers, and downstream Real seals before a completion-facing read is exported. It does not identify filters with nets, does not assert quotient equality of their classifiers, and does not choose a common representative.*

Proof. Project the $\mathsf{CauchyFilterLimit}^{\uparrow}$ carrier. The displayed route of Definition 1558.1 includes the stream row S, regular readback row R, dyadic tolerance ledger D, and Real seal row A.

Compare with the Cauchy-net packet only at the shared rows. The directed-net coverage theorem above requires any net-facing consumer of the filter-limit packet to preserve the same S, R, D, A route, so those rows form the comparison surface.

The comparison stops at row sharing. Since the filter packet has no coordinate for quotient filters, selected subnets, quotient equality of nets, or a common representative, the result records finite route compatibility without collapsing the two classifiers. □

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier filter comparison`

Theorem 1558.10 (CauchyFilterLimit bridge surface)**.** *For every accepted* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *carrier* $L = (B, F, S, R, D, A, H, C, P, N)$*, the public route supplies the bridge surface from the concrete Cauchy-filter-limit carrier to the abstract completion-limit interface. The surface preserves basis saturation, regular readback, dyadic tolerance, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming, and it refuses quotient filters, selected subnets, host limits, choice, and ambient completeness.*

Proof. Start with the basis-saturation and regular-readback obligations of Definition 1558.5 and Definition 1558.6. They force every completion-facing request to expose B, F, S, R, D, A before any limit read is exported.

Apply the ledger refusal theorem Definition 1558.7. It keeps quotient filters, selected host limits, host real equality, countable-choice selectors, and ambient completeness outside the carrier surface.

The directed-net coverage theorem Definition 1558.8 and the comparison theorem Definition 1558.9 show that downstream net or completion consumers can only compare the shared finite rows. Transport, replay, provenance, and naming then repack the same route as the bridge surface without adding a host-limit coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyFilterLimitUp.CauchyFilterLimitCarrier bridge surface`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterLimit}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1558.1, Definition 1558.2, Definition 1558.3, Definition 1558.4, Definition 1558.5, Definition 1558.6, Definition 1558.7, Definition 1558.8, Definition 1558.9, and Definition 1558.10 close the directed-net handoff, regular seal handoff, filter comparison, and bridge surface from generated filter data to completion-limit consumers and the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterLimit}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, host real equality, countable-choice selectors, ambient $\mathsf{Real}^{\uparrow}$ completeness, or stronger Lean verification.

Upgrade path: Further work is a Lean-side verification and bridge audit for the same mature paper-side surface.

Constructive story: A $\mathsf{CauchyFilterLimit}^{\uparrow}$ packet is generated from a CauchyFilterBasis row, a CauchyFilter saturation row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1559

A Concrete Naming Certificate for MetricCompletionFinite$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the MetricCompletionFinite$^{\uparrow}$ packet records the finite-window completion boundary that can be inspected before any Real$^{\uparrow}$ seal is read. It sits downstream of the metric-completion surface of chapter 1506, the generated Cauchy-filter basis interface of chapter 1501, the dyadic precision window of chapter 1524, and the embedding route of chapter 1512. Its role is to bind those rows to a regular-rational or Real-facing readback without importing quotient completions, selected representatives, countable choice, open-cover compactness, host equality, or an ambient complete metric space.

Definition 1559.1 (MetricCompletionFinite carrier)**.** *A* MetricCompletionFinite$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, B, W, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* MetricCompletion$^{\uparrow}$ *row naming the completion-facing metric source and its finite admissibility boundary.*
2. *B is the* CauchyFilterBasis$^{\uparrow}$ *row, or the finite CauchyNet branch when the source is presented by nets, from which the completion request is generated.*
3. *W is the* DyadicRoundingWindow$^{\uparrow}$ *precision row that fixes the finite dyadic scale inspected by the request.*
4. *E is the* CompletionEmbedding$^{\uparrow}$ *row carrying the source object into the completion-facing interface.*
5. *R is the* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *seal readback row exposed only after M, B, W, E have been replayed.*
6. *S is the local seal selector that records which finite window is consumed before the readback row is public.*
7. *H records componentwise* hsame *transport for the metric-completion, Cauchy-basis, dyadic-window, embedding, readback, seal-selector, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through the Cauchy basis, dyadic window, completion embedding, finite seal selector, and readback row.*
9. *P is the* Pkg *provenance over* MetricCompletion$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, DyadicRoundingWindow$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFinite}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completion equality, a selected representative, countable-choice selection, open-cover compactness, host equality, or an ambient complete metric space.

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteTasteGate single carrier alignment`

Theorem 1559.2 (MetricCompletionFinite total-bounded window)**.** *Every accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet exposes its finite total-bounded branch through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *row, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row or finite CauchyNet branch, the* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *precision row, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row, componentwise transport, replay, provenance, and local* NameCert *row before any completion consumer may read a finite total-bounded window.*

The exposed window is exactly the carrier route

$$M, B, W, E, S \longmapsto R \longmapsto H, C, P, N,$$

and it exports no quotient completion equality, selected representative, countable-choice selector, open-cover compactness row, host equality, or ambient complete metric space.

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteTotalBoundedWindow`

Proof. Project the carrier of Definition 1559.1. Its rows $M, B, W, E, R, S, H, C, P, N$ are the only admitted coordinates, so the finite total-bounded branch must be read through that tuple.

Use the finite-window and readback rows already displayed in the carrier. The Cauchy source row B, dyadic window row W, embedding row E, and local selector S precede the readback row R, and the transport, replay, provenance, and naming rows preserve that order.

Finally apply the ledger refusal clause of Definition 1559.3. Since none of the excluded completion principles has a carrier coordinate, no consumer can replace the finite window by quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or an ambient complete metric space. □

Theorem 1559.3 (MetricCompletionFinite NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *carrier* $F = (M, B, W, E, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFinite}^{\uparrow}}$ *surface consists exactly of these five obligation rows:*

1. *carrier admission for the displayed metric-completion, Cauchy-basis, dyadic-window, embedding, readback, seal-selector, transport, continuation, provenance, and local naming rows;*
2. *generated-Cauchy source exposure, requiring B to be read before a completion-facing consumer inspects W, E, R, or S;*
3. *finite-window precision exposure, requiring W and S to display the dyadic scale consumed by the readback route;*
4. *embedding and readback handoff, requiring E to precede every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read through R;*
5. *ledger refusal, excluding quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, and ambient complete metric spaces.*

No local NameCert row is stored outside those five obligations and the structural rows H, C, P, N.

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteNameCertObligations`

Proof. Project the carrier of Definition 1559.1. Its public rows are exactly $M, B, W, E, R, S, H, C, P, N$, so carrier admission cannot expose a second completion object, representative selector, cover datum, host equality, or ambient completeness witness.

The generated source and window obligations come from the order of the displayed continuation. A completion-facing request first reads the metric-completion row M and the Cauchy basis row B, then the dyadic precision row W and local selector S.

The embedding and readback handoff is the remaining public route. The row E carries the source into the completion-facing interface, and R is readable only after E and the finite window rows have been replayed through H, C, P, N. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteCarrier namecert obligations`

Theorem 1559.4 (MetricCompletionFinite obligation boundary)**.** *Every accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet exposes carrier admission, generated-Cauchy source exposure, finite-window precision exposure, embedding and readback handoff, and ledger refusal before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal consumer can read completion-facing data. The boundary is the displayed tuple*

$$F = (M, B, W, E, R, S, H, C, P, N),$$

and no consumer may replace it by quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or an ambient complete metric space.

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteCarrier obligation boundary`

Proof. Project the carrier of Definition 1559.1. The rows $M, B, W, E, R, S, H, C, P, N$ are the only available coordinates, so carrier admission and ledger refusal are already fixed by the displayed packet.

Apply the NameCert obligations of Definition 1559.3. They require the generated-Cauchy source row B, the finite dyadic window W, the local selector S, and the embedding row E to be exposed before a readback row can be public.

Finally use the window readback theorem Definition 1559.5. It forces every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through the finite route $M, B, W, E, S \longmapsto R \longmapsto H, C, P, N$, so none of the excluded quotient, representative, choice, compactness, host-equality, or ambient-completeness data can be read from the packet. □

Theorem 1559.5 (MetricCompletionFinite window readback)**.** *Let* $F = (M, B, W, E, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet. Every completion-facing read that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or a* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite route*

$$M, B, W, E, S \longmapsto R \longmapsto H, C, P, N.$$

Thus the consumer sees the $\mathsf{MetricCompletion}^{\uparrow}$ *row, the generated Cauchy-filter or CauchyNet finite branch, the* $\mathsf{DyadicRoundingWindow}^{\uparrow}$ *precision row, and the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row before it reads any regular-rational or Real-facing seal.*

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteWindowReadback`

Proof. Begin with an accepted completion-facing consumer. Carrier admission gives access to the readback row R, but only as part of the displayed packet.

Replay the continuation row C. The request passes through M for the completion-facing metric source, B for the finite generated Cauchy data, W for the dyadic precision window, E for the embedding route, and S for the local seal selector.

Only after those rows are replayed can the consumer read R. The transport, provenance, and local naming rows preserve that dependency order, so the visible route factors through the finite window before any $\mathsf{Real}^{\uparrow}$ seal is public. □

Theorem 1559.6 (MetricCompletionFinite classifier exactness)**.** *Let $F = (M, B, W, E, R, S, H, C, P, N)$ and $F' = (M', B', W', E', R', S', H', C', P', N')$ be accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packets. The local classifier identifies F with F' exactly when the metric-completion, Cauchy-source, dyadic-window, embedding, readback, seal-selector, transport, continuation, provenance, and local naming rows are componentwise equal along the displayed* hsame *route:*

$$M\mathsf{hsame}M', \quad B\mathsf{hsame}B', \quad W\mathsf{hsame}W', \quad E\mathsf{hsame}E', \quad R\mathsf{hsame}R', \quad S\mathsf{hsame}S', \quad H\mathsf{hsame}H', \quad C\mathsf{hsame}C', \quad P$$

There is no additional classifier coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or ambient complete metric spaces.

Proof. Use the carrier of Definition 1559.1. Both accepted packets have exactly the rows $M, B, W, E, R, S, H, C, P, N$ and their primed counterparts, so the classifier can inspect only those coordinates.

Apply the NameCert obligations of Definition 1559.3. They force carrier admission, source exposure, precision exposure, embedding handoff, and ledger refusal to be read through the same displayed rows.

Finally use the readback route of Definition 1559.5. A classifier equality that survives the readback must therefore be componentwise hsame on those rows, and componentwise hsame supplies the same finite route in the reverse direction. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteClassifierExactness`

Theorem 1559.7 (MetricCompletionFinite stability transport)**.** *If accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packets $F = (M, B, W, E, R, S, H, C, P, N)$ and $F' = (M', B', W', E', R', S', H', C', P', N')$ are classifier-equal in the sense of Definition 1559.6, then carrier admission and readback eligibility transport from F to F' and from F' to F. In particular, every consumer allowed to read R through M, B, W, E, S is allowed to read R' through M', B', W', E', S' after the componentwise* hsame *transport.*

Proof. Start with the componentwise classifier supplied by Definition 1559.6. It gives transport for each carrier coordinate, including the Cauchy-source row, dyadic-window row, completion-embedding row, seal selector, and readback row.

Carrier admission is one of the obligations of Definition 1559.3, so it is preserved by the displayed transport row H and the local naming row N.

Readback eligibility is the route of Definition 1559.5. Transport each row in the route $M, B, W, E, S \longmapsto R$ to the primed route and repeat the same argument in the opposite direction. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteCarrier stability transport`

Theorem 1559.8 (MetricCompletionFinite ledger exactness)**.** *The refusal ledger of an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet is exact: it excludes quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, and ambient complete metric spaces, and it excludes no displayed coordinate of the carrier $F = (M, B, W, E, R, S, H, C, P, N)$.*

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteCarrier ledger exactness`

Proof. The carrier of Definition 1559.1 lists every available coordinate. The rows $M, B, W, E, R, S, H, C, P, N$ are admitted by the packet and therefore cannot be part of the refusal ledger.

The obligation theorem Definition 1559.3 names the ledger refusal row. Its excluded objects are exactly the quotient, representative, choice, compactness, host-equality, and ambient-completeness data.

The readback theorem Definition 1559.5 confirms that every completion-facing read stays inside the displayed finite route. Hence the ledger refuses precisely the non-displayed completion principles and leaves the admitted carrier coordinates available. □

Theorem 1559.9 (MetricCompletionFinite non-escape)**.** *An accepted* MetricCompletionFinite$^{\uparrow}$ *packet exports no quotient completion equality, no selected representative, no countable-choice selector, no open-cover compactness row, no host equality, and no ambient complete metric space. Every completion-facing consumer of the packet must read only the displayed rows $M, B, W, E, R, S, H, C, P, N$.*

Proof. The NameCert obligations of Definition 1559.3 list the only carrier, source, precision, embedding, readback, and refusal rows available to the packet.

The window readback theorem Definition 1559.5 then covers every route that reaches a regular-rational readback or Real seal. Since the carrier has no additional coordinate beyond $M, B, W, E, R, S, H, C, P, N$, none of the excluded quotient, representative, choice, compactness, host-equality, or ambient-completeness objects can be projected from an accepted packet. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteNonescape`

Theorem 1559.10 (MetricCompletionFinite obligation closure)**.** *The displayed packet rows M, B, W, E, R, S, H, C close the* MetricCompletionFinite$^{\uparrow}$ *obligation surface: carrier admission, classifier exactness, finite-window readback, stability transport, ledger exactness, and non-escape are all accounted for inside the accepted finite-window completion packet.*

Proof. Carrier admission is the NameCert obligation package of Definition 1559.3. It exposes exactly the metric-completion row, basis row, finite precision window, embedding row, readback row, local seal row, transport, replay, provenance, and local naming row.

The finite-window readback and classifier surface are then fixed by Definition 1559.5 and Definition 1559.6. Together they keep every completion-facing read on the displayed $M, B, W, E, R, S, H, C, P, N$ route.

Finally apply stability transport, ledger exactness, and non-escape from Definition 1559.7, Definition 1559.8, and Definition 1559.9. These rows preserve the route under hsame transport, identify the refusal ledger exactly, and rule out quotient completion equality, selected representatives, countable choice, compactness, host equality, and ambient complete metric spaces. □

Theorem 1559.11 (MetricCompletionFinite obligation closure register)**.** *For every accepted* MetricCompletionFinit *packet, the displayed carrier row of Definition 1559.1, the local NameCert obligations of Definition 1559.3, classifier exactness from Definition 1559.6, finite-window readback from Definition 1559.5, stability transport from Definition 1559.7, ledger exactness from Definition 1559.8, and non-escape from Definition 1559.9 are exactly the obligationClosure packet for* MetricCompletionFinite$^{\uparrow}$.

Proof. Carrier admission is read from Definition 1559.1. The NameCert surface and the displayed completion-facing obligation boundary are supplied by Definition 1559.3 and Definition 1559.10.

The reader-facing route is fixed by Definition 1559.5, while classifier comparison and transport are fixed by Definition 1559.6 and Definition 1559.7.

Finally, Definition 1559.8 and Definition 1559.9 account for the refusal ledger and rule out non-displayed completion principles. These cited rows exhaust the required obligationClosure register, so no additional packet coordinate remains. □

Theorem 1559.12 (MetricCompletionFinite finite-net limit readback)**.** *Let $F = (M, B, W, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet. Any completion-facing finite-net consumer must read the metric-completion row M, the finite Cauchy-basis or CauchyNet branch B, the dyadic window W, the embedding row E, and the local seal selector S before it reads R, and it must keep the structural rows H, C, P, N attached to that route.*

Consequently the finite-net readback cannot be replaced by quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or an ambient completeness principle.

Proof. Start from the carrier of Definition 1559.1. It lists $M, B, W, E, R, S, H, C, P, N$ as the complete finite completion-facing packet.

Apply Definition 1559.5. The consumer can reach a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal only through

$$M, B, W, E, S \longmapsto R \longmapsto H, C, P, N.$$

Use Definition 1559.6 to keep the readback on this displayed finite route under classifier comparison. Then use Definition 1559.9 to exclude the non-displayed completion principles from the exported packet. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteFiniteNetLimit`

Theorem 1559.13 (MetricCompletionFinite scoped dependency package)**.** *For every accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet, the carrier of Definition 1559.1, the local NameCert obligations of Definition 1559.3, classifier exactness, finite-window readback, finite-net limit readback, stability transport, ledger exactness, and non-escape bind every completion-facing read to the displayed dependency package*

$\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHi

The CauchyNet branch is read only through the finite source row B, and every readback into a regular-rational or Real-facing seal remains attached to the same $M, B, W, E, R, S, H, C, P, N$ route.

The package exports no quotient completion, selected representative, countable choice family, open-cover compactness proof, host equality, or ambient complete metric space.

Proof. First project the displayed carrier. The rows $M, B, W, E, R, S, H, C, P, N$ are the whole completion-facing packet, and the NameCert obligation theorem makes carrier admission, source exposure, finite-window precision, embedding handoff, and ledger refusal visible before any readback.

Next apply classifier exactness, window readback, finite-net limit readback, and stability transport. These theorems force every completion-facing consumer to pass through the metric-completion row, the Cauchy-filter or CauchyNet source branch, the dyadic window, the completion embedding, the local seal selector, and the readback row before the structural rows replay and name the same route.

Finally use ledger exactness and non-escape. The refusal ledger names exactly the quotient-completion, representative-selection, countable-choice, open-cover-compactness, host-equality, and ambient-completeness principles that are absent from the carrier, so the scoped package contains only the displayed completion rows and their hsame, Cont, Pkg, and $\mathsf{NameCert}$ structure. □

Theorem 1559.14 (MetricCompletionFinite public export route)**.** *Every public consumer of an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet receives exactly the displayed carrier, local* $\mathsf{NameCert}$ *obligations, classifier exactness, finite-window readback, finite-net limit readback, stability transport, ledger exactness, and non-escape rows before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is public. Equivalently, the public export route is the visible packet*

$$F = (M, B, W, E, R, S, H, C, P, N)$$

together with the dependency package of Definition 1559.13; it exports no quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or ambient complete metric space.

Proof. Project the displayed carrier of Definition 1559.1. This gives exactly the rows $M, B, W, E, R, S, H, C, P, N$ and no second completion object.

Apply Definition 1559.3 to expose carrier admission, generated-Cauchy source exposure, finite-window precision exposure, embedding and readback handoff, and ledger refusal. Then use Definition 1559.5 and Definition 1559.12 to keep both regular-rational and finite-net completion reads upstream of the seal row.

Classifier exactness and stability transport keep the same rows public under componentwise hsame comparison. Ledger exactness and non-escape, Definition 1559.8 and Definition 1559.9, refuse the non-displayed completion principles.

Finally apply Definition 1559.13. It binds the public consumer to the same scoped finite-window completion dependency package before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal is public, so the export route contains exactly the listed rows. □

Theorem 1559.15 (MetricCompletionFinite bridge window)**.** *An accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *carrier whose finite-net limit readback and public export route are displayed can hand a finite completion window to* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal consumers. The handoff is only a finite* $\mathsf{BHist/Cont/Pkg}$ *ledger over* $M, B, W, E, R, S, H, C, P, N$ *and the dependency package of Definition 1559.13; it does not export quotient completion equality, selected representatives, ambient complete metric spaces, or Real completeness.*

Proof. Apply Definition 1559.12. It forces finite-net consumers to read the metric-completion row, finite Cauchy-basis or CauchyNet branch, dyadic window, embedding row, local seal selector, readback row, and structural rows before any completion-facing output is available.

Next apply Definition 1559.13. The package keeps the handoff attached to $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so $\mathsf{StreamName}^{\uparrow}$ consumers see only the displayed finite windows and $\mathsf{RegSeqRat}^{\uparrow}$ consumers see only the regular-rational readback.

Finally apply the public export route of Definition 1559.14. The public consumer receives the same finite packet before any $\mathsf{Real}^{\uparrow}$ seal is public. Since the ledger and non-escape rows refuse quotient completion equality, selected representatives, ambient complete metric spaces, and Real completeness, the bridge window is a finite completion handoff rather than an ambient completion theorem. □

Theorem 1559.16 (MetricCompletionFinite L10 public bridge)**.** *Any public completion-facing export through an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet factors through*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CauchyFilterBasis}^{\uparrow},\ \mathsf{DyadicRoundingWindow}^{\uparrow},\ \mathsf{CompletionEmbedding}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before it can be read as a public L10 completion datum. The bridge exposes only the displayed finite-window carrier, local NameCert obligations, readback rows, dependency package, and public export route. It refuses quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, and ambient complete metric spaces.

Proof. Start with the accepted carrier of Definition 1559.1. The obligation surface of Definition 1559.3 exposes the metric-completion row, Cauchy-filter basis, dyadic rounding window, completion embedding, regular rational readback, Real seal, transport, replay, provenance, and local naming rows before a public consumer can read the packet.

Next apply Definition 1559.5. It keeps every regular-rational or Real-facing read inside the finite dyadic window and prevents the export from bypassing $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$.

The scoped package theorem, Definition 1559.13, then binds the same rows as the only completion-facing dependency package. The public export theorem, Definition 1559.14, makes that package the visible surface for consumers.

Since the ledger in these rows has no coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or ambient complete metric spaces, the public read is an L10 finite-window completion datum only after passing through the listed rows. □

Theorem 1559.17 (MetricCompletionFinite bridge scope)**.** *Every bridge-facing read of an accepted* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *packet factors through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it can serve as the abstract completion interface. The bridge scope is therefore the finite route named by the carrier, NameCert obligations, window readback, scoped dependency package, and public export route; it does not export quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or ambient complete metric spaces.*

Proof. Project the carrier of Definition 1559.1. Its rows already display the metric-completion source, Cauchy-filter basis branch, dyadic window, completion embedding, regular-rational readback, Real seal, structural transport, replay, provenance, and local naming.

Apply the NameCert obligations of Definition 1559.3 and the window readback of Definition 1559.5. Together they force every bridge-facing read to expose the finite-window packet before any regular-rational or Real-facing completion datum is public.

The scoped dependency package Definition 1559.13 then pins the same read to $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{DyadicRoundingWindow}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The public export route Definition 1559.14 makes that package the visible boundary for downstream consumers.

Since the carrier and ledger rows have no coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, or ambient complete metric spaces, the bridge-facing read is exactly a finite abstract-completion interface and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFinite}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFinite}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1559.1, Definition 1559.3, Definition 1559.5, Definition 1559.16, and Definition 1559.17 bind the mature finite-window completion package through carrier rows, NameCert obligations, window readback, public export routing, and bridge-facing scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFinite}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricCompletionFiniteUp.MetricCompletionFiniteWindowReadback`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, ambient complete metric spaces, or bridge checking.

Upgrade path: Bridge checking requires a Lean bridge theorem connecting the mature finite-window completion packet to the abstract MetricCompletion$^{\uparrow}$ interface.

Constructive story: A MetricCompletionFinite$^{\uparrow}$ packet is generated from a MetricCompletion row, a CauchyFilterBasis or CauchyNet finite branch, a DyadicRoundingWindow precision row, a CompletionEmbedding row, a RegSeqRat or Real seal readback, local seal selection, componentwise hsame transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1560

A Concrete Naming Certificate for $\mathsf{CompletionDensity}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompletionDensity}^{\uparrow}$ packet records the finite density surface by which source names remain visible inside the regular Cauchy completion route. It sits between the embedding packet of chapter 1512, the regular dense embedding of chapter 1453, the regular rational name surface of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the dense image of displayed source names before any completion-facing consumer can treat the route as a completed real object; it is not a quotient completion, a selected limit, a countable-choice construction, a host real equality principle, or an ambient completeness assertion.

Definition 1560.1 (CompletionDensity carrier)**.** *A* $\mathsf{CompletionDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, W, R, T, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-name row whose entries are to remain dense after entering the completion route;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source names are inspected;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows;*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that controls the finite density test;*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that carries the source names into the completion-facing surface;*
6. *Q is the accepted* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *row witnessing that every regular Cauchy density request reads the image through S, W, R, T, E before reaching the* $\mathsf{Real}^{\uparrow}$ *seal;*
7. *H records componentwise* hsame *transport for the source row, windows, readbacks, dyadic ledger, embedding row, dense-image row, continuation row, provenance row, and local naming row;*
8. *C records displayed* Cont *replay from a source-name request through finite windows, regular readback, dyadic tolerance, completion embedding, dense-image verification, and real-seal hand-off;*
9. *P is the* Pkg *provenance over* $\mathsf{CompletionDensity}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionDensity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient completion, a selected representative limit, countable choice, host real equality, ambient completeness, or an external dense subset.

Theorem 1560.2 (CompletionDensity NameCert obligations)**.** *For an accepted* $\mathsf{CompletionDensity}^{\uparrow}$ *carrier $D = (S, W, R, T, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionDensity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source names, finite stream windows, regular rational readbacks, dyadic tolerance ledger, completion embedding row, regular Cauchy dense-image row, transports, replay route, provenance, and local naming row;*
2. *source-image exactness, requiring every dense-image claim to start from the row S and to inspect it only through W, R, T;*
3. *embedding compatibility, requiring the row E to be the only completion-facing handoff of those source names;*
4. *dense-image stability, requiring the accepted row Q to remain attached to S, W, R, T, E under the componentwise* hsame *row H and displayed* Cont *replay C;*
5. *provenance exhaustion, requiring P and N to name only the displayed route and to export no quotient completion, selected limit, countable-choice selector, host real equality, ambient completeness row, or detached dense subset.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1560.1. The tuple displays the source row, window row, regular readback, tolerance ledger, completion embedding, dense-image row, transports, replay, provenance, and local name.

Source-image exactness is forced by the order of the displayed rows. A density request first reads S, then inspects the source name through W, obtains regular rational readback through R, and consults the dyadic ledger T before the completion embedding row E is available.

Dense-image stability is the role of Q, H, C. The accepted regular Cauchy dense embedding row Q is transported only componentwise by H and replayed only along the continuation route C, so no second dense-image witness or detached subset can enter the certificate.

Finally, P and N package and name the same finite route. Since the carrier lists no quotient completion, selected limit, countable-choice selector, host real equality, ambient completeness, or external dense subset coordinate, those data cannot be exported by the local NameCert surface. □

Theorem 1560.3 (CompletionDensity regular Cauchy dense image)**.** *Let $D = (S, W, R, T, E, Q, H, C, P, N)$ be an accepted* $\mathsf{CompletionDensity}^{\uparrow}$ *packet. Every regular Cauchy density read of the source image factors through the finite route*

$$S, W, R, T, E, Q, H, C, P, N.$$

In particular, a $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *consumer can read the dense image only after the source names, stream windows, regular rational readbacks, dyadic tolerance ledger, and completion embedding row have been displayed. The read cannot substitute a quotient completion, a selected host limit, a countable-choice schedule, host real equality, ambient completeness, or an external dense subset.*

Proof. Start with a regular Cauchy dense-image consumer. The carrier gives the consumer only the tuple rows of Definition 1560.1.

The consumer reaches the dense-image row Q after the source route S, W, R, T and the embedding row E have been displayed. Thus every density read is downstream of the same finite source-name and tolerance route.

Transport and replay preserve this route through H and C, while P and N record it as provenance and local naming data. Since no quotient, selected host limit, countable-choice schedule, host equality, ambient completeness, or external dense subset appears in the tuple, the dense image is exhausted by the stated finite packet. □

Theorem 1560.4 (CompletionDensity Real handoff)**.** *For an accepted* $\mathsf{CompletionDensity}^{\uparrow}$ *carrier* $D = (S, W, R, T, E, Q, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing completion read that uses the dense image first consumes the displayed route*

$$\mathsf{StreamName}^{\uparrow}(W), \quad \mathsf{RegSeqRat}^{\uparrow}(R), \quad \mathsf{DyadicRatCore}^{\uparrow}(T), \quad \mathsf{CompletionEmbedding}^{\uparrow}(E), \quad \mathsf{RegularCauchyDenseEr}$$

The terminal real-name seal therefore receives only a dense image generated from source names already carried by S and replayed through H, C, P, N. It receives no quotient completion, selected limit, countable choice, host real equality, ambient completeness, or independent real equality witness.

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing completion read. The packet exposes no terminal real coordinate except through the dense-image row Q and the structural rows H, C, P, N.

The row Q is downstream of the completion embedding row E, and E is downstream of the source-name route S, W, R, T. Hence the real-facing consumer must first read the StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, completion embedding, and regular Cauchy dense-image row displayed in the statement.

The remaining rows only transport, replay, package, and name that route. Because the carrier has no quotient-completion coordinate, selected limit, countable choice, host real equality, ambient completeness row, or independent real equality witness, the Real handoff is exactly the displayed finite density route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionDensity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1560.1, Definition 1560.2, Definition 1560.3, and Definition 1560.4 seed the dense source-image surface between CompletionEmbedding, RegularCauchyDenseEmbedding, RegSeqRat, StreamName, DyadicRatCore, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionDensity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, host real equality, ambient completeness, independent real equality witnesses, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-image exactness, embedding compatibility, dense-image stability, Real handoff, and provenance exhaustion for BEDC.Derived.CompletionDensityUp.

Constructive story: A $\mathsf{CompletionDensity}^{\uparrow}$ packet is generated from source names, StreamName finite windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a CompletionEmbedding row, an accepted RegularCauchyDenseEmbedding row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1561

A Concrete Naming Certificate for $\mathsf{CauchyNetLimit}^{\uparrow}$

Following Kelley and Bishop–Bridges constructive analysis, the $\mathsf{CauchyNetLimit}^{\uparrow}$ packet names the directed-net limit boundary reached after a $\mathsf{CauchyNet}^{\uparrow}$ directed-window row has been consumed and before a $\mathsf{Real}^{\uparrow}$ seal can be read. It sits downstream of chapter 589, reads the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the dyadic tolerance ledger of chapter 291, and exposes the seal boundary of chapter 38 only through the completion-facing boundary of chapter 1506. The chapter records a finite directed-net limit packet, not a quotient of nets, not a selected host limit, and not an ambient completeness principle.

Definition 1561.1 (CauchyNetLimit carrier)**.** *A* $\mathsf{CauchyNetLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row. It names the inspected directed windows and the common upper-window witnesses consumed by the limit request.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row read from the directed windows exposed by K.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning the scheduled window observations into regular rational data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the regular rational observations at the requested directed-limit scale.*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, W, R, D have been replayed.*
6. *H records componentwise* hsame *transport for the directed-window, window, readback, tolerance, seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a directed-net limit request through finite windows, regular readback, dyadic tolerance, and then the seal row.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient net equality, a selected host limit, cofinal-choice selection, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier`

Theorem 1561.2 (CauchyNetLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetLimit}^{\uparrow}$ *carrier* $L = (K, W, R, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyNetLimit}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed CauchyNet directed-window row, StreamName windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, continuation, provenance, and local naming row;*
2. *directed-window consumption, requiring the* K *row to supply the displayed directed windows and common upper-window witnesses before any limit-facing read can inspect* W, R, D, A*;*
3. *regular readback, requiring each limit-scale observation to pass through* W *and* R *before the tolerance ledger* D *is consulted;*
4. *Real seal handoff, requiring the seal row* A *to remain downstream of* K, W, R, D*;*
5. *ledger refusal, excluding quotient nets, selected limits, cofinal choice, host equality, and Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1561.1. The projection exposes exactly the accepted CauchyNet row, StreamName window row, regular readback row, dyadic tolerance ledger, Real seal, and structural rows H, C, P, N.

Directed-window consumption is forced by the row order. The continuation row C routes a limit request from K into the finite windows W before the request can reach the regular readback R, the tolerance ledger D, or the seal row A.

The regular readback and seal conditions are internal to the same packet. Transport by H preserves the route, while P and N package and name only that displayed route, so no separate quotient, selected-limit, cofinal-choice, host-equality, or completeness coordinate remains available. □

Lean4 ✓ `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier semantic name certificate`

Theorem 1561.3 (CauchyNetLimit Real seal handoff)**.** *Let* $L = (K, W, R, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyNetLimit}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the finite route*

$$K, W, R, D \longmapsto A \longmapsto H, C, P, N.$$

Thus the limit seal is read only after the $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger have been displayed. The route contains no quotient net, selected host limit, cofinal-choice selector, host real equality, or ambient Real completeness principle.*

Proof. Begin with a Real-facing consumer of the packet. Carrier admission gives access to the seal row A, but only as a row downstream of K, W, R, D.

Replay the continuation row C. The request first consumes the accepted CauchyNet directed-window row K, then the finite StreamName windows W, the regular rational readback R, and the dyadic tolerance ledger D.

Only after this replay can the seal row A be read. The transport, provenance, and local naming rows record the same dependency order, so the packet exports no direct route through quotient nets, selected host limits, cofinal choice, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier real seal handoff`

Theorem 1561.4 (CauchyNetLimit directed-window obligations)**.** *Let $L = (K, W, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyNetLimit}^{\uparrow}$ *packet. Every directed-limit read must consume the accepted* $\mathsf{CauchyNet}^{\uparrow}$ *row K before it reads the window row W, regular readback row R, dyadic tolerance ledger D, or* $\mathsf{Real}^{\uparrow}$ *seal row A. Common upper-window witnesses exposed by K remain stable under the componentwise transport row H and replay through C before the packet can be sealed by P and N.*

Proof. Project the carrier of Definition 1561.1. The row K is the only source for the directed windows and their common upper-window witnesses, while W, R, D, A are downstream reads.

Replay C in the displayed order. The request first consumes K, then the finite windows W, the regular rational readback R, and the dyadic tolerance ledger D before the seal row A is available.

Transport by H preserves the same finite route. The provenance row P and local naming row N therefore package the directed-window obligation without adding quotient nets, selected limits, cofinal choice, host equality, or Real completeness. □

Lean4 ✓ `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier directed window obligations`

Theorem 1561.5 (CauchyNetLimit dyadic-tolerance handoff)**.** *Let $L = (K, W, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyNetLimit}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing read consumes the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row K, the* $\mathsf{StreamName}^{\uparrow}$ *finite window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the seal row A is visible. Thus the Real seal handoff is a dyadic-tolerance handoff, not a direct passage from directed nets to host real equality or ambient completeness.*

Proof. Project the carrier of Definition 1561.1. The rows K, W, R, D are the only rows from which a Real-facing request can obtain directed-window evidence, finite stream observations, regular rational readback, and dyadic tolerance data.

Use Definition 1561.2. Its directed-window consumption, regular readback, Real seal handoff, and ledger refusal clauses require the request to pass through K, W, R, D before the seal row A is available.

The order is the same as in Definition 1561.3 and Definition 1561.4. Replay by C first opens the directed-window row, then the finite window and regular readback rows, then the dyadic tolerance ledger; transport by H preserves this route when the packet is moved componentwise.

The directed-window component consumes the finite upper-window carrier of chapter 1607 before any Cauchy-net limit seal is read.

Therefore any $\mathsf{Real}^{\uparrow}$-facing consumer reads A only after the dyadic tolerance ledger has been displayed. The packet still contains no quotient net, selected host limit, cofinal-choice selector, host real equality, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier dyadic tolerance handoff`

Theorem 1561.6 (CauchyNetLimit abstract completion bridge)**.** *The public* $\mathsf{CauchyNetLimit}^{\uparrow}$ *boundary factors into the abstract completion-facing interface exactly through the displayed rows $K, W, R, D, A, H, C, P, N$ of Definition 1561.1. The factorisation reads the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row K, the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance ledger D, the Real$^{\uparrow}$ *seal row A, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N. It has no quotient nets, selected host limits, cofinal choice, host equality, or ambient Real completeness coordinate.*

Proof. Project the carrier in Definition 1561.1. The abstract completion-facing interface can inspect only the displayed directed-window, finite-window, regular-readback, dyadic-tolerance, and Real-seal rows, followed by the structural rows H, C, P, N.

Use the route constraints already isolated in Definition 1561.2, Definition 1561.3, and Definition 1561.5. They force every public read to pass through K, W, R, D before A, and then through the transport, replay, provenance, and local naming rows.

Thus the public boundary and the completion-facing interface share exactly the same finite row list. Since no carrier coordinate supplies quotient-net equality, a selected limit, cofinal choice, host equality, or Real completeness, none can escape through the bridge. □

Lean4 ✓ `BEDC.Derived.CauchyNetLimitUp.CauchyNetLimitCarrier abstract completion bridge`

Closure status for NameCert$_{\text{CauchyNetLimit}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchyNetLimit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1561.1, Definition 1561.2, Definition 1561.3, Definition 1561.4, Definition 1561.5, and Definition 1561.6 bind the directed-window, regular-readback, dyadic-tolerance, Real$^{\uparrow}$-seal, abstract completion-facing bridge, hsame, Cont, Pkg, and NameCert scope for finite Cauchy-net limit reads.

Formal verification: FormalStatus(NameCert$_{\text{CauchyNetLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected host limits, cofinal choice, host real equality, ambient Real$^{\uparrow}$ completeness, or Lean verification.

Upgrade path: No further paper-side closure upgrade is claimed beyond bridge-maintenance and formal verification.

Constructive story: A CauchyNetLimit$^{\uparrow}$ packet is generated from a CauchyNet directed-window row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1562

A Concrete Naming Certificate for $\mathsf{DyadicRatCoreArithmetic}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicRatCoreArithmetic}^{\uparrow}$ packet names the finite arithmetic-window face of the dyadic rational core used before regular rational names and real completions inspect arithmetic data. It sits downstream of chapter 291, reads the signed rows of chapter 31, the rational mantissa rows of chapter 37, the finite schedule rows of chapter 126, the regular readback rows of chapter 301, and the completion-facing seal of chapter 38. The packet records arithmetic stability for displayed dyadic windows; it is not full $\mathsf{Real}^{\uparrow}$ arithmetic, not a quotient construction, not an order principle, not completeness, and not a standard bridge.

Definition 1562.1 (DyadicRatCoreArithmetic carrier)**.** *A* $\mathsf{DyadicRatCoreArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M_0, M_1, E_0, E_1, L_0, L_1, S, W, H, C, P, N)$$

with the following displayed rows:

1. *M_0 and M_1 are signed mantissa rows, each read through the* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *surfaces used by accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets.*
2. *E_0 and E_1 are positive unary exponent rows, carrying the dyadic denominator scale before any common-scale readback is allowed.*
3. *L_0 and L_1 are the denominator ledgers attached to the two mantissa-exponent pairs.*
4. *S is the common-scale normalization row, displaying the finite exponent window into which both mantissas are transported.*
5. *W is the arithmetic-window ledger for sum, negation, difference, and product readbacks at that common scale.*
6. *H records componentwise* hsame *transport for mantissas, exponents, denominator ledgers, common-scale rows, arithmetic windows, continuation rows, provenance rows, and the local naming row.*
7. *C records* Cont *replay from the two input dyadic packets through common-scale normalization and then through the arithmetic-window ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRatCoreArithmetic}^{\uparrow}}$ *row.*

The carrier has no coordinate for full $\mathsf{Real}^{\uparrow}$ *arithmetic, order, quotient rational equality, completion endpoints, completeness, or a standard bridge.*

Lean4 (definition) `BEDC.Derived.DyadicRatCoreUp.DyadicRatCoreArithmeticWindow`

Theorem 1562.2 (DyadicRatCoreArithmetic NameCert obligations)**.** *For every accepted* DyadicRatCoreArithmetic *carrier* $A = (M_0, M_1, E_0, E_1, L_0, L_1, S, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicRatCoreArithmetic}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two signed mantissas, two unary exponents, two denominator ledgers, common-scale normalization row, arithmetic-window ledger, transport, continuation, provenance, and local naming row;*
2. *source compatibility, requiring the mantissa and exponent rows to be read as the arithmetic face of accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets;*
3. *common-scale exactness, requiring* S *to be consumed before the arithmetic row* W *is read;*
4. *arithmetic-window transport, requiring* H *and* C *to preserve the displayed sum, negation, difference, and product windows under* hsame *transport;*
5. *downstream handoff, requiring* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers to inspect only the finite window exposed by* W*;*
6. *ledger refusal, excluding full* $\mathsf{Real}^{\uparrow}$ *arithmetic, order, quotient rational equality, completion endpoints, completeness, and standard-bridge data.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1562.1. The projection exposes exactly the two signed mantissa rows, two positive unary exponent rows, denominator ledgers, common-scale row, arithmetic-window ledger, and structural rows H, C, P, N.

Source compatibility is forced by the attachment to chapter 291. The mantissas can be inspected only through the displayed $\mathsf{Int}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ rows already packaged by the dyadic core, and the exponent rows can be inspected only through the visible unary denominator ledgers.

Replay the continuation row C. A consumer reaches W only after the two input ledgers have been transported into the common-scale row S, so arithmetic readback never bypasses normalization.

Transport by H preserves the displayed coordinates of the same finite window. The package and local naming rows record only this route, hence no full real arithmetic, order, quotient equality, completion endpoint, completeness, or standard bridge is available. □

Theorem 1562.3 (DyadicRatCoreArithmetic normalization handoff)**.** *Let* $A = (M_0, M_1, E_0, E_1, L_0, L_1, S, W, H, C$ *be an accepted* $\mathsf{DyadicRatCoreArithmetic}^{\uparrow}$ *carrier. Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of its arithmetic row factors through the finite route*

$$M_0, M_1, E_0, E_1, L_0, L_1 \longmapsto S \longmapsto W \longmapsto H, C, P, N.$$

Thus the downstream phase reads arithmetic only as a normalized dyadic window over displayed mantissa, exponent, denominator, transport, continuation, provenance, and naming data. The route supplies no full $\mathsf{Real}^{\uparrow}$ *arithmetic, order, quotient rational equality, completion endpoint, completeness principle, or standard bridge.*

Proof. Begin with a downstream consumer of the arithmetic row. Carrier admission exposes W, but only as a row downstream of the mantissa, exponent, denominator, and common-scale coordinates.

Replay C from the two dyadic inputs. The route first reads the signed mantissas M_0, M_1, then the unary exponent rows E_0, E_1, then the denominator ledgers L_0, L_1, and only then the common-scale row S.

The arithmetic ledger W is therefore a normalized finite dyadic window. Its visible rows are exactly the sum, negation, difference, and product readbacks transported by H and packaged by P, N.

Since the route ends at the displayed window and structural rows, it cannot export a completed real operation, order classifier, quotient equality, completion endpoint, completeness witness, or standard-bridge component. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRatCoreArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicRatCoreArithmetic}^{\uparrow}$ packet is generated from two signed mantissa rows, two unary exponent rows, denominator ledgers, a common-scale normalization row, an arithmetic-window ledger, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicRatCoreArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1562.1, Definition 1562.2, and Definition 1562.3 seed the finite arithmetic-window face of $\mathsf{DyadicRatCore}^{\uparrow}$ used by downstream regular and completion phases.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicRatCoreArithmetic}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close full $\mathsf{Real}^{\uparrow}$ arithmetic, order, quotient construction, completion endpoints, completeness, a standard bridge, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, common-scale normalization, arithmetic-window transport, downstream handoff, and ledger refusal proofs for the same packet.

Chapter 1563

A Concrete Naming Certificate for $\mathsf{CauchyShift}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyShift}^{\uparrow}$ packet records the finite certified route by which a regular Cauchy name may ignore an initial finite prefix and still present the same tail-facing regularity surface to a downstream $\mathsf{Real}^{\uparrow}$ seal. It sits beside the schedule-changing packet of chapter 1568, reads the explicit modulus discipline of chapter 292, and consumes the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The chapter names a finite tail shift of a regular Cauchy name, not quotient stream equality, not a selected limit, not countable choice, not host real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1563.1 (CauchyShift carrier)**.** *A* $\mathsf{CauchyShift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, k, M, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose finite observation windows are available before the shift is requested.*
2. *T is the shifted* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose nth displayed request is routed to the source request $n + k$.*
3. *k is the finite unary shift-length row, recording the displayed prefix length and no unbounded selector.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row that certifies every post-shift precision request by replaying the corresponding source-tail modulus request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the source and shifted observations at the finite budgets named by k and M.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row showing that the shifted schedule remains a regular rational name after the tail route has been replayed.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, T, k, M, D, R have been consumed.*
8. *H records componentwise* hsame *transport for the source schedule, shifted schedule, shift length, modulus row, dyadic tolerance row, regularity row, Real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a shifted-window request through the finite shift length, back to the source schedule, then forward through the modulus row, dyadic tolerance, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyShift}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, countable-choice tail selection, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1563.2 (CauchyShift NameCert obligations)**.** *For every accepted* $\mathsf{CauchyShift}^{\uparrow}$ *carrier* $K = (S, T, k, M, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyShift}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source schedule, shifted schedule, finite shift length, Cauchy modulus row, dyadic tolerance row, regularity row, Real seal, transport, continuation, provenance, and local naming row;*

2. *shift admissibility, requiring every shifted-window read in T to replay through the displayed finite unary shift k before it can consult the source schedule S;*

3. *tail-modulus discipline, requiring M to be consumed before a post-shift observation can be treated as a regular Cauchy tail observation;*

4. *tolerance and readback preservation, requiring D and R to certify the finite regular rational readback after the shifted route has been replayed;*

5. *seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row A to remain downstream of S, T, k, M, D, R;*

6. *ledger refusal, excluding quotient streams, selected limits, countable-choice tail selectors, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1563.1. The projection exposes exactly the source schedule, shifted schedule, finite shift length, Cauchy modulus row, dyadic tolerance row, regularity row, Real seal, and structural rows H, C, P, N.

Shift admissibility is forced by the continuation row C. A shifted-window request reaches T only by replaying the displayed unary shift length k back to the source schedule S, so the packet has no independent tail stream coordinate.

The modulus row M is then the only row that permits the shifted request to be treated as a Cauchy-tail request. The dyadic tolerance row D and regularity row R sit downstream of that tail route, so tolerance preservation and regular readback cannot be exported before the finite shift discipline has been consumed.

Finally, the seal row A is downstream of the same route. Transport by H preserves each listed coordinate, while P and N package only that finite route; hence no quotient stream, selected limit, countable-choice tail selector, host real equality, or ambient completeness coordinate remains available. □

Theorem 1563.3 (CauchyShift regularity transport)**.** *Let $K = (S, T, k, M, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyShift}^{\uparrow}$ *carrier. Every displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity read on the shifted schedule T factors through the finite route*

$$S, k, M, D \longmapsto T \longmapsto R.$$

Thus finite-prefix shift preserves the displayed regularity surface through the Cauchy modulus row and the dyadic tolerance row before any downstream consumer can read R as a regular rational name.

Proof. Begin with a shifted regularity read. Carrier admission gives access to the $\mathsf{RegSeqRat}^{\uparrow}$ row R, but only as the row downstream of the shifted schedule T, finite shift length k, modulus row M, and dyadic tolerance row D.

Replay the continuation row C. The shifted request is translated by the finite unary length k into the source schedule S. The modulus row M then supplies the displayed tail budget for that source request, and the dyadic tolerance row D records the finite comparison at the requested scale.

Once those rows have been consumed, T has exactly the displayed tail observations licensed by S, k, M, D. The regular readback row R therefore transports regularity along the finite shift route, with H preserving the same dependency order and P, N naming only that packet. □

Theorem 1563.4 (CauchyShift Real seal boundary)**.** *Let $K = (S, T, k, M, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyShift}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read of the shifted name factors through the finite route*

$$S, T, k, M, D, R \longmapsto A \longmapsto H, C, P, N.$$

The shifted name reaches the seal only after the source schedule, shifted schedule, finite shift length, Cauchy modulus row, dyadic tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed. The route contains no quotient stream equality, selected limit, countable-choice tail selector, host real equality, or ambient completeness principle.*

Proof. Begin with a Real-facing consumer of the packet. Carrier admission gives access to the seal row A, but only as a row downstream of S, T, k, M, D, R.

Replay the continuation row C. The request first reads the shifted schedule T as a finite-prefix shift of the source schedule S, then consumes the shift length k and the Cauchy modulus row M that make the tail request admissible.

After those rows have been consumed, the dyadic tolerance row D and regularity row R provide the finite readback accepted by the Real seal. The transport, provenance, and local naming rows record the same dependency order, so the packet exports no direct route through quotient streams, selected limits, countable-choice tail selection, host real equality, or ambient completeness. □

Theorem 1563.5 (CauchyShift reindexing selector handoff)**.** *Let $K = (S, T, k, M, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyShift}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing read of K that is also consumed by a* $\mathsf{CauchyReindexing}^{\uparrow}$ *route or by a* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *route factors through the ordered handoff*

$$S, k, M, D \longmapsto T \longmapsto R \longmapsto \mathsf{RegularCauchyModulusRealization}^{\uparrow} \longmapsto \mathsf{RealCauchyModulusSelector}^{\uparrow} \longmapsto A$$

with the structural rows H, C, P, N preserving that order. Thus the shifted name may be reindexed, realized as a regular Cauchy modulus surface, selected by the Real Cauchy modulus selector, and then sealed at A, but it exports no quotient stream, selected limit, countable-choice tail selector, host real equality, detached modulus selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Apply Definition 1563.3. Every shifted regularity read first consumes the source schedule, finite shift length, Cauchy modulus row, and dyadic tolerance row before it can read the shifted schedule T and the regular rational row R.

The downstream reindexing side is the route recorded in chapter 1568. A consumer that asks for a $\mathsf{CauchyReindexing}^{\uparrow}$ read therefore receives only the displayed regular readback produced by the shift packet; it does not obtain a second stream coordinate or an independent equality of streams.

The modulus-selection side is the route recorded in chapter 1577 and chapter 1576. The regular Cauchy modulus realization reads the finite M, S, D, R-style window as a displayed regular surface, and the Real Cauchy modulus selector then chooses only the threshold, finite window, tolerance, readback, and seal rows already displayed by that route.

Finally apply Definition 1563.4. The seal row A remains downstream of the shifted route and of the structural rows H, C, P, N. Since neither this carrier nor the cited sibling routes contain quotient streams, selected limits, countable-choice tail selectors, host real equality, detached modulus selectors, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the composite handoff cannot export any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyShift}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyShift}^{\uparrow}$ packet is generated from a source StreamName schedule, a shifted StreamName schedule, a finite unary shift length, a CauchyModulus tail row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyShift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1563.1, Definition 1563.2, Definition 1563.3, Definition 1563.4, and Definition 1563.5 seed the finite tail-shift route from regular Cauchy names through reindexing, regular modulus realization, selector handoff, and the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyShift}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable-choice tail selectors, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires carrier admission, regularity transport, Real seal boundary, and ledger refusal proofs for the same finite tail-shift packet.

Chapter 1564

A Concrete Naming Certificate for $\mathsf{FrechetFilter}^{\uparrow}$

Following Bourbaki and Kelley, and in the Bishop–Bridges constructive reading of filters, the $\mathsf{FrechetFilter}^{\uparrow}$ packet names the cofinite-tail filter interface used when a sequence tail is read as a finite BEDC filter surface. It refines the finite tail-neighborhood packet of chapter 1119, records the filter-base rows of chapter 1601, reads the finite schedule rows of chapter 126, and hands the same tail route to the sequential-completion interface of chapter 1515 before any regular rational readback or Real-facing completion consumer can inspect the route. The same cofinite-tail filter basis is also the finite route consumed by chapter 1569 before any completion-facing read can inspect a Cauchy-filter limit boundary. The chapter records only the finite cofinite-tail carrier and its handoff rows, not Real completeness, quotient filter equality, selected limits, countable choice, or ambient completion.

Definition 1564.1 (FrechetFilter carrier)**.** *A* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, T, S, M, B, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the unary threshold row. It names the displayed tail bound after which the filter request may read sequence data.*
2. *T is the cofinite tail-neighborhood row. It records finite complements and the tail containment checks supplied by chapter 1119.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose finite windows are inspected only after U and T have been consumed.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row whose tolerance requests are read only through the displayed stream windows and tail threshold.*
5. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row. It packages the cofinite subsets as a finite filter-base surface and records the common-tail meet operation used by downstream consumers.*
6. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row. It receives the cofinite-tail neighborhood surface after the metric and filter-base rows have been displayed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows and tail budgets.*

8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row available only to Real and completion-facing consumers after U, T, S, M, B, Q, R have been replayed.*
9. *H records componentwise* hsame *transport for thresholds, tail neighborhoods, StreamName schedule rows, Metric rows, FilterBase rows, CauchyFilter handoff rows, RegSeqRat readback, Real seal, continuation, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a filter request through the unary threshold, cofinite tail neighborhood, StreamName schedule, Metric row, FilterBase row, CauchyFilter handoff, RegSeqRat readback, and then the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, quotient filter equality, selected limits, countable-choice tail selection, or ambient completion.

Lean4 (definition) `BEDC.Derived.FrechetFilterUp.FrechetFilterCarrier`
Lean4 (definition) `BEDC.Derived.FrechetFilterUp.FrechetFilterUp`

Theorem 1564.2 (FrechetFilter NameCert obligations)**.** *For every accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the unary threshold, cofinite tail-neighborhood row, StreamName schedule row, Metric row, FilterBase row, CauchyFilter handoff row, RegSeqRat readback row, Real seal, transport, continuation, provenance, and local naming row;*
2. *tail-neighborhood admission, requiring every filter request to pass through U and T before it can inspect a stream window;*
3. *cofinal-tail stability, requiring any two displayed thresholds over the same cofinite-tail surface to have a common displayed tail through the FilterBase meet row B;*
4. *convergence-consumer handoff, requiring the displayed cofinite-tail surface to be the only row exported to* $\mathsf{CauchyFilter}^{\uparrow}$ *and then to sequential-completion consumers;*
5. *readback and seal order, requiring S and R to be consumed before the Real seal A can be read;*
6. *ledger refusal, excluding Real completeness, quotient filter equality, selected limits, countable-choice selectors, and ambient completion.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1564.1. The projection exposes exactly the unary threshold, cofinite tail-neighborhood row, StreamName schedule, Metric row, FilterBase row, CauchyFilter handoff, RegSeqRat readback, Real seal, and structural rows H, C, P, N.

Tail-neighborhood admission is forced by the continuation row C. A filter request reaches the schedule row S only after it has replayed through the unary threshold U and the cofinite tail row T, so the packet has no independent infinite-tail coordinate.

The FilterBase row B supplies the common-tail meet for any pair of displayed thresholds over that same cofinite surface. The Cauchy-source row Q is then the only row exported to $\mathsf{CauchyFilter}^{\uparrow}$, and the same route is the handoff offered to sequential-completion consumers.

The readback row R and Real seal A sit downstream of the same route, while H preserves componentwise transport and P, N package only that finite carrier. Hence no Real completeness, quotient filter equality, selected limit, countable-choice selector, or ambient completion coordinate remains available. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterNameCertObligations`

Theorem 1564.3 (FrechetFilter carrier admission obligation)**.** *For every accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier* $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$*, the rows* $U, T, S, M, B, Q, R, A, H, C, P, N$ *are the complete admitted source surface for the local* $\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$ *packet. In particular every* $\mathsf{FiniteTailFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{FilterBase}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, transport, continuation, provenance, and local naming read is admitted only through those displayed rows.*

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterCarrierAdmissionObligation`

Proof. Project the carrier of Definition 1564.1. Its displayed coordinates are exactly the unary threshold, cofinite tail-neighborhood, schedule, metric, filter-base, handoff, readback, seal, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 1564.2. The carrier-admission clause lists the same rows and the final clause says that no NameCert obligation is stored outside them.

The later handoff and consumer-route theorems use only this admitted surface. In particular Definition 1564.9, Definition 1564.11, and Definition 1564.15 all replay through subroutes of $U, T, S, M, B, Q, R, A, H, C, P, N$, so no extra source coordinate is available to any consumer. □

Theorem 1564.4 (FrechetFilter cofinal-tail stability obligation)**.** *Let* $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ *be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. If two accepted cofinite-tail requests are based on the same finite-tail row* T*, then their displayed thresholds refine through the* $\mathsf{FilterBase}^{\uparrow}$ *meet row* B *before any* $\mathsf{StreamName}^{\uparrow}$ *schedule row or* $\mathsf{Met}^{\uparrow}$ *tolerance row is read.*

Proof. Project the carrier of Definition 1564.1. The rows U and T present the unary threshold and cofinite tail-neighborhood data, while B is the displayed row where the common-tail meet is stored.

Use Definition 1564.2. Its cofinal-tail stability clause requires two displayed thresholds over the same cofinite-tail surface to pass through the common displayed tail supplied by B.

The route is then the one isolated by Definition 1564.9 and Definition 1564.10. Those theorems place S and M downstream of U, T, B, so schedule and metric consumers cannot read before the FilterBase meet has refined the two requests. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterCofinalTailStabilityObligation`

Theorem 1564.5 (FrechetFilter convergence-consumer handoff obligation)**.** *Let* $F = (U, T, S, M, B, Q, R, A, H, C,$ *be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$*-facing consumer receives the FrechetFilter route only through the displayed rows*

$$U, T, S, M, B, Q, R, A, H, C, P, N.$$

No convergence-facing consumer receives a separate quotient filter, selected limit, countable-choice selector, Real-completeness proof, or ambient completion object.

Proof. Project Definition 1564.1. The rows Q, R, A are downstream handoff, readback, and seal rows, and H, C, P, N only transport, replay, package, and name the same finite packet.

Apply Definition 1564.2. The convergence-consumer handoff clause identifies the displayed cofinite-tail surface as the only export to $\mathsf{CauchyFilter}^{\uparrow}$ and sequential-completion consumers, while the readback and seal clause fixes the order in which R and A may be read.

The consumer skeleton is supplied by Definition 1564.11, Definition 1564.13, Definition 1564.15, and Definition 1564.20. Together these routes replay exactly $U, T, S, M, B, Q, R, A, H, C, P, N$, and the ledger refusal theorem keeps quotient equality, selected limits, choice selectors, completeness, and ambient completion outside the handoff. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterConvergenceConsumerHandoffObligation`

Theorem 1564.6 (FrechetFilter root tail-base admission)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every cofinite-tail read admitted at the root of the packet must pass through the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *threshold rows U, T, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row S, the* $\mathsf{Met}^{\uparrow}$ *tolerance row M, the* $\mathsf{FilterBase}^{\uparrow}$ *meet row B, the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal row A, and the structural rows H, C, P, N. No root read may expose a tail-neighborhood basis without that ordered finite packet.*

Proof. Project the carrier of Definition 1564.1. The root rows U, T are the only coordinates where the cofinite-tail threshold and finite-complement data are admitted.

Use Definition 1564.9. It routes any tail-neighborhood consumer through U, T before the schedule, metric, and FilterBase rows S, M, B can be read.

Now apply Definition 1564.5. It places Q, R, A, H, C, P, N downstream of the same finite cofinite-tail surface, so a root tail-base admission cannot bypass the CauchyFilter, RegSeqRat, Real, transport, replay, provenance, and naming rows when those consumers are present. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterRootTailBaseAdmission`

Theorem 1564.7 (FrechetFilter root Cauchy consumer boundary)**.** *For an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$, every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer at the root boundary reads the cofinite-tail filter only through*

$$\mathsf{FiniteTailFilter}^{\uparrow}(U, T),\ \mathsf{StreamName}^{\uparrow}(S),\ \mathsf{Met}^{\uparrow}(M),\ \mathsf{FilterBase}^{\uparrow}(B),\ \mathsf{CauchyFilter}^{\uparrow}(Q),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(A),$$

The boundary exports no quotient filter equality, selected limit, countable-choice tail selector, or independent completion coordinate.

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterRootCauchyConsumerBoundary`

Proof. Begin with the convergence-consumer handoff obligation, Definition 1564.5. It states that every CauchyFilter, RegSeqRat, SequentiallyCompleteMetric, and Real-facing consumer receives exactly the displayed rows $U, T, S, M, B, Q, R, A, H, C, P, N$.

The Cauchy side is isolated by Definition 1564.11. That theorem forces the consumer to read the unary threshold, cofinite-tail row, schedule row, metric row, and FilterBase meet before the CauchyFilter handoff row Q is visible.

The regular-rational read is then supplied by Definition 1564.13. It places R after the same cofinite-tail and CauchyFilter route, while A, H, C, P, N seal and name the packet without adding a selected limit or quotient equality. □

Theorem 1564.8 (FrechetFilter root sequential-completion handoff)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *consumer at the root boundary must consume the same finite cofinite-tail route*

$$\mathsf{FiniteTailFilter}^{\uparrow}(U, T),\ \mathsf{StreamName}^{\uparrow}(S),\ \mathsf{Met}^{\uparrow}(M),\ \mathsf{FilterBase}^{\uparrow}(B),\ \mathsf{CauchyFilter}^{\uparrow}(Q),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(A),$$

before it treats the packet as a sequential-completion source. The handoff does not provide ambient metric completion, selected limits, quotient filters, countable-choice selectors, or a completed $\mathsf{Real}^{\uparrow}$ *object.*

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterRootSequentialCompletionHandoff`

Proof. Start from Definition 1564.20. It already identifies the sequential-completion consumer route as the displayed rows $U, T, S, M, B, Q, R, A, H, C, P, N$.

Next use Definition 1564.6. This pins the root admission of the cofinite-tail basis to the $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{FilterBase}^{\uparrow}$ rows before any downstream completion-facing consumer appears.

Finally combine Definition 1564.7 with Definition 1564.14. The Cauchy and readback rows Q, R, A remain downstream of the same finite tail, and the nonescape theorem refuses ambient completion, selected limits, quotient filters, and countable-choice selectors. □

Theorem 1564.9 (FrechetFilter tail-neighborhood handoff)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every displayed cofinite-tail neighborhood read factors through the finite route*

$$U, T \longmapsto S, M, B \longmapsto H, C, P, N.$$

Thus the $\mathsf{FiniteTailFilter}^{\uparrow}$ *source supplies the unary threshold and finite-complement rows before the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{Met}^{\uparrow}$ *tolerance row, and* $\mathsf{FilterBase}^{\uparrow}$ *meet row are allowed to serve any tail window.*

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterTailNeighborhoodHandoff`

Proof. Begin with a tail-neighborhood consumer. Carrier admission gives the threshold row U and the cofinite tail-neighborhood row T as explicit coordinates of the same BHist packet.

Replay C. The request first reads the unary threshold, then the finite complement and tail-containment data in T, and only then reaches the $\mathsf{StreamName}^{\uparrow}$ schedule row S, the $\mathsf{Met}^{\uparrow}$ tolerance row M, and the $\mathsf{FilterBase}^{\uparrow}$ meet row B.

Transport by H preserves this order under classifier movement, while P and N name the same local packet. Therefore a tail-neighborhood read cannot bypass the finite threshold and finite-complement surface. □

Theorem 1564.10 (FrechetFilter common-tail refinement)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ and $F' = (U', T', S', M', B', Q', R', A', H', C', P', N')$ be two accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carriers over the same finite-tail filter row. Then the two unary threshold requests admit a displayed common tail threshold W read through that finite-tail row, with W above both U and U' by BEDC row comparison. Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer of the two requests factors through the same finite cofinite-tail surface*

$$U, U', T \longmapsto W \longmapsto S, M, B, Q, R, A.$$

The refinement exports neither quotient filter equality, selected limits, countable-choice tail selectors, nor ambient Real completeness.

Proof. Project the two carriers and read their unary threshold rows U and U' over the shared finite-tail filter row. Since the thresholds are displayed unary $\mathsf{Nat}^{\uparrow}$ rows, the finite-tail row supplies the BEDC row comparison needed to choose the larger displayed threshold as W.

Replay both requests through their continuation rows. The request from F reaches the shared tail surface by first consuming U and T, while the request from F' reaches it by first consuming U' and the same finite-tail row. The common threshold W refines both routes, so any tail window admitted after W is admitted for both original requests.

Now apply the handoff route of Definition 1564.9. A $\mathsf{StreamName}^{\uparrow}$ consumer reads only the finite cofinite-tail surface determined by W before inspecting schedule data.

For Cauchy and Real-facing consumers, apply Definition 1564.11 and Definition 1564.15. These routes force the $\mathsf{Met}^{\uparrow}$ row, $\mathsf{FilterBase}^{\uparrow}$ row, $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal to factor through the same common finite tail. Hence the refinement cannot expose quotient filter equality, selected limits, countable-choice selectors, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterTailCommonRefinement`

Theorem 1564.11 (FrechetFilter Cauchy-source boundary)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyFilter}^{\uparrow}$ *source read exported from the packet factors through*

$$U, T, S, M, B \longmapsto Q \longmapsto H, C, P, N.$$

The Cauchy-filter consumer receives a cofinite-tail source built from unary thresholds, cofinite tail neighborhoods, displayed StreamName schedule rows, Metric tolerance rows, and FilterBase meet rows, with no quotient filter equality, countable choice, or selected limit.

Proof. Start from a Cauchy-source request. The carrier exposes the handoff row Q, but only as a row downstream of U, T, S, M, B.

The continuation row C replays the request through the unary threshold and the cofinite tail-neighborhood row before the schedule, metric, and filter-base rows can be inspected. This replay is exactly the finite cofinite-tail surface that Q hands to $\mathsf{CauchyFilter}^{\uparrow}$.

Since H transports the same coordinates and P, N name no additional source, the packet exports only the displayed cofinite-tail source boundary. Quotient filter equality, countable-choice selection, selected limits, and ambient completion are outside the carrier. □

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterCauchySourceBoundary`

Theorem 1564.12 (FrechetFilter Cauchy-filter export)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every exported* $\mathsf{CauchyFilter}^{\uparrow}$ *read factors through*

$$U, T, S, M, B, Q, H, C, P, N.$$

The export is therefore a displayed cofinite-tail Cauchy-filter source rather than quotient filter equality, countable-choice tail selection, a selected limit, or an ambient completion object.

Proof. Project the carrier of Definition 1564.1. The CauchyFilter handoff row Q is present only after the unary threshold, cofinite tail-neighborhood, StreamName schedule, Metric tolerance, and FilterBase meet rows have been displayed.

Apply the NameCert obligations of Definition 1564.2. They identify Q as the sole convergence-consumer handoff and require ledger refusal for quotient filter equality, selected limits, countable-choice selectors, and ambient completion.

Finally replay the tail route through Definition 1564.9 and the common-tail refinement of Definition 1564.10. The exported read remains anchored in the same finite cofinite-tail surface, with H, C, P, N supplying transport, replay, provenance, and local naming. □

Theorem 1564.13 (FrechetFilter RegSeqRat tail readback)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exported by F is downstream of the cofinite tail threshold U, the cofinite tail-neighborhood row T, the schedule row S, and the FilterBase meet row B. No readback consumer can inspect R as an independent regular rational stream.*

Proof. Read the carrier definition. The row R is not a source coordinate; it is the readback row attached to the displayed stream windows and tail budgets.

Use tail-neighborhood handoff to force each stream window through U and T before S is visible. Then use common-tail refinement to show that simultaneous tail requests are compared by a displayed common threshold and by the FilterBase meet row B.

The NameCert obligations place readback and seal order after the convergence-consumer handoff. Hence the $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives only a tail-indexed readback row, not an independent stream selector or a completed real endpoint. □

Theorem 1564.14 (FrechetFilter completion nonescape)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. The packet exports no quotient filter equality, selected limit, countable-choice tail selector,* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completion. Every completion-facing read must pass through the displayed route*

$$U, T, S, M, B, Q, R, A, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterCompletionNonescape`

Proof. Start with the ledger refusal clause in Definition 1564.2. It excludes quotient filter equality, selected limits, countable-choice selectors, and ambient completion from the local NameCert surface.

Attach the Cauchy-filter export theorem above. It forces any completion-facing request to receive its Cauchy source through $U, T, S, M, B, Q, H, C, P, N$ rather than through an external filter object.

Finally apply the Real and completion consumer route of Definition 1564.15. The route reaches A only after Q and R have been consumed, so the Real-facing seal is a finite read boundary and not a proof of $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1564.15 (FrechetFilter Real and completion consumer route)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every Real or completion-facing consumer of the Frechet filter route factors through*

$$U, T, S, M, B, Q, R \longmapsto A \longmapsto H, C, P, N.$$

Thus the consumer reads unary thresholds, cofinite tail neighborhoods, StreamName schedule rows, Metric tolerance rows, FilterBase meet rows, the CauchyFilter handoff, and RegSeqRat readback before reaching the Real seal. The route does not claim Real completeness, quotient filter equality, selected limits, countable choice, or ambient completion.

Proof. Begin with a Real or completion-facing consumer. Carrier admission gives access to A only as the seal row downstream of the threshold, tail-neighborhood, schedule, CauchyFilter handoff, and readback rows.

Replay C along the displayed route. The request consumes U and T, inspects the schedule row S, reads the metric and filter-base rows M, B, hands the cofinite-tail source through Q, and then reads the regular rational row R.

Only after those rows have been consumed can the seal row A be reached. Componentwise transport by H and provenance through P, N preserve the same dependency order, so the packet supplies a finite consumer route rather than Real completeness, quotient filter equality, selected limits, countable choice, or ambient completion. □

Theorem 1564.16 (FrechetFilter FilterBase meet obligation)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every pair of cofinite tail requests accepted by F must be compared through the displayed* $\mathsf{FilterBase}^{\uparrow}$ *meet row B before any* $\mathsf{StreamName}^{\uparrow}$ *schedule row or* $\mathsf{Met}^{\uparrow}$ *tolerance row is consumed. The meet row is therefore a finite cofinite-tail obligation, not a consequence of stream inspection or metric convergence.*

Proof. Project the carrier of Definition 1564.1. The rows U, T expose the unary threshold and cofinite tail-neighborhood surface, while B is the only coordinate that packages common-tail meets.

Apply the tail-neighborhood handoff of Definition 1564.9. It places S and M downstream of U, T and records that stream windows and metric tolerances may be read only after the finite tail surface is visible.

Use the common-tail refinement of Definition 1564.10. The displayed common threshold is read through the same cofinite-tail row, so the meet obligation belongs to B before any schedule or metric consumer can inspect later rows. □

Theorem 1564.17 (FrechetFilter cofinite-tail closure induction)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Any finite iteration of the* $\mathsf{FilterBase}^{\uparrow}$ *meet row B over displayed cofinite-tail requests preserves the common-tail route of Definition 1564.10. After the iteration, every resulting request still factors through*

$$U, T \longmapsto S, M, B \longmapsto Q, R, A, H, C, P, N,$$

with no selected tail, quotient filter equality, countable-choice selector, or ambient completion coordinate.

Proof. Use induction on the finite list of cofinite-tail requests displayed by the packet. The empty and singleton cases are exactly the carrier route of Definition 1564.9.

For the step, apply Definition 1564.10 to the current common tail and the next displayed request. The $\mathsf{FilterBase}^{\uparrow}$ meet row B supplies the next common threshold while keeping the route inside U, T, S, M, B.

Finally replay the downstream handoff through Definition 1564.16. The CauchyFilter, RegSeqRat, Real, transport, continuation, provenance, and naming rows remain downstream of the same finite cofinite-tail surface. □

Theorem 1564.18 (FrechetFilter threshold monotonicity)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Enlarging the displayed cofinite-tail threshold preserves every downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumer of the packet. The enlarged read may replay the same rows $S, M, Q, R, A, H, C, P, N$ only after the new threshold has passed through T and the* $\mathsf{FilterBase}^{\uparrow}$ *meet row B.*

Proof. Begin with the threshold and tail-neighborhood handoff of Definition 1564.9. A larger displayed threshold remains a cofinite-tail request over the same finite complement row T.

Apply Definition 1564.10. The original threshold and the enlarged threshold admit the enlarged threshold itself as a displayed common tail, so the schedule and metric rows still read the same cofinite surface through S, M, B.

The downstream consumers are then forced by Definition 1564.16. Since Q, R, A, H, C, P, N are reached only after the filter-base meet row, threshold enlargement preserves their route without adding a selector or completion witness. □

Theorem 1564.19 (FrechetFilter cofinite meet associativity)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. For any three displayed cofinite-tail requests in the same packet, the two iterated meet routes through B,*

$$(U_1 \wedge U_2) \wedge U_3 \qquad \textit{and} \qquad U_1 \wedge (U_2 \wedge U_3),$$

determine the same displayed BHist *common-tail route for downstream consumers. The associativity claim is only a route-level equality over the carrier rows; it does not assert quotient filter equality.*

Proof. Apply the finite closure induction of Definition 1564.17 to the ordered triple of displayed cofinite-tail requests. Both parenthesizations consume the same three requests through the same $\mathsf{FilterBase}^{\uparrow}$ meet row B.

Use Definition 1564.10 at each binary meet. In each parenthesization the final threshold is the displayed common tail above all three inputs, so the downstream route begins with the same finite cofinite-tail surface.

Finally apply Definition 1564.16. Since schedule, metric, CauchyFilter, RegSeqRat, Real, transport, replay, provenance, and local naming rows are read only after B, associativity holds at the displayed BHist route level and does not create quotient filter equality. □

Theorem 1564.20 (FrechetFilter sequential-completion handoff)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *consumer of F receives only the rows*

$$U, T, S, M, B, Q, R, A, H, C, P, N.$$

In particular the handoff exports a displayed cofinite-tail source, regular rational readback, Real-facing seal, transport, replay, provenance, and local naming, but no selected limit and no ambient completion witness.

Lean4 ✓ `BEDC.Derived.FrechetFilterUp.FrechetFilterSequentialCompletionHandoff`

Proof. Begin with the Cauchy-source boundary of Definition 1564.11. It sends any convergence-facing request through U, T, S, M, B before the Cauchy-filter handoff row Q is visible.

Next apply Definition 1564.15. The Real and completion-facing route reaches the readback row R and the seal row A only after the Cauchy-filter handoff has been consumed.

Finally use Definition 1564.14. Since the carrier has no coordinate for a selected limit or ambient completion object, a sequential-completion consumer receives only the displayed finite rows named in the statement. □

Theorem 1564.21 (FrechetFilter tail meet monotone refinement)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. For any two displayed cofinite-tail requests read from the finite-tail row T, the* $\mathsf{FilterBase}^{\uparrow}$ *meet row B refines them monotonically: enlarging either unary threshold only enlarges the displayed common-tail threshold, and every downstream* $\mathsf{StreamName}^{\uparrow}$ *schedule row,* $\mathsf{Met}^{\uparrow}$ *tolerance row,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal continues to factor through the same finite cofinite-tail route.*

Proof. Project the carrier. The rows U, T, B are the only rows where threshold data, finite complements, and common-tail meets are displayed before schedule or metric consumers can read later coordinates.

Apply Definition 1564.10 to the two cofinite-tail requests. The common threshold is read through the same finite tail row, and enlarging either input threshold can only move the displayed common tail farther along that row.

Now replay downstream consumption through Definition 1564.16. Since S, M, Q, R, A are all downstream of B, the monotone refinement preserves their route and does not introduce quotient filter equality, a selected tail, countable choice, a selected limit, or ambient completion. □

Theorem 1564.22 (FrechetFilter Cauchy-filter basis cofinal route)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. Every Cauchy-filter-basis consumer of the packet reads the same cofinal tail route*

$$U, T \longmapsto B \longmapsto S, M, Q \longmapsto H, C, P, N.$$

The consumer may use the $\mathsf{FilterBase}^{\uparrow}$ *meet row to refine finite tails, but it cannot replace the displayed basis by quotient filter equality, selected representatives, countable-choice tail selection, or an ambient completion object.*

Proof. Begin with the Cauchy-source boundary of Definition 1564.11. It places every $\mathsf{CauchyFilter}^{\uparrow}$ source read after U, T, S, M, B and before the structural rows H, C, P, N.

Use Definition 1564.21. Any basis refinement demanded by the Cauchy consumer is a refinement through the same cofinite-tail meet row B, so the basis remains cofinal in the displayed finite-tail surface.

Finally apply Definition 1564.12. The export is the row Q attached to that same route, not a quotient filter or selected representative. The absent quotient, selection, choice, and completion data have no carrier coordinates. □

Theorem 1564.23 (FrechetFilter sequential-completion tail exhaustion)**.** *Let $F = (U, T, S, M, B, Q, R, A, H, C, P,$ be an accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier. The sequential-completion handoff exhausts the tail ledger: every* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *read consumes the unary threshold row, the cofinite-tail row, the FilterBase meet row, the StreamName schedule row, the Metric tolerance row, the Cauchy-filter handoff, the regular rational readback, the Real seal, and the structural rows H, C, P, N, with no remaining tail coordinate outside the displayed packet.*

Proof. Start from Definition 1564.20. It states that a sequential-completion consumer receives only $U, T, S, M, B, Q, R, A, H, C, P, N$.

Insert the cofinal basis route from Definition 1564.22. This identifies the Cauchy-filter input to the completion consumer with the same finite cofinite-tail route through $U, T, B, S, M, Q, H, C, P, N$.

Complete the route with Definition 1564.13 and Definition 1564.14. The readback row R and seal row A are downstream of the exhausted tail ledger, so the handoff cannot export quotient filter equality, a selected limit, countable choice, a selected representative, or ambient completion. □

Theorem 1564.24 (FrechetFilter tail-ledger refusal)**.** *For every accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$, the local tail ledger refuses quotient filter equality, countable-choice tail selection, selected limits, ambient completion, and* $\mathsf{Real}^{\uparrow}$ *completeness. Each refused conclusion is absent because it is not one of the carrier coordinates $U, T, S, M, B, Q, R, A, H, C, P, N$ and is not produced by the displayed handoff theorems.*

Proof. Project the carrier definition and the NameCert obligations. The carrier contains threshold, tail-neighborhood, schedule, metric, filter-base, Cauchy-filter, readback, seal, transport, replay, provenance, and local naming rows; the obligation list explicitly records the ledger refusal.

Use the Cauchy-source boundary and sequential-completion handoff above. Both routes preserve the same finite row order and never introduce quotient filter equality, a countable-choice selector, or a selected limit.

Finally apply Definition 1564.15. The Real-facing seal is downstream of the finite readback route, so it is not a proof of $\mathsf{Real}^{\uparrow}$ completeness and cannot be inflated into ambient completion. □

Theorem 1564.25 (FrechetFilter tail eventuality obligations)**.** *Every accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *tail-event read exposes a unary threshold, cofinite tail row, common-tail refinement,* $\mathsf{FilterBase}^{\uparrow}$ *meet route,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff,* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *window, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. Equivalently, any eventuality consumer reads only the finite route*

$$U, T, S, M, B, Q, R, A, H, C, P, N$$

and receives no quotient filter equality, countable-choice tail selector, selected limit, selected representative, ambient completion theorem, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the cofinite-tail closure induction and threshold monotonicity rows. They place every tail event behind the unary threshold row U and cofinite tail row T, and they identify larger thresholds with refinements of the same displayed tail surface.

Use tail meet monotone refinement. The common-tail row is read through the $\mathsf{FilterBase}^{\uparrow}$ meet route B, so parallel tail events are combined inside the same finite basis ledger rather than through a quotient filter.

Apply the Cauchy-filter basis cofinal route and the sequential-completion tail exhaustion theorem. The $\mathsf{CauchyFilter}^{\uparrow}$ handoff Q and the $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ consumer both inspect the same U, T, S, M, B, Q prefix before any completion-facing row is exposed.

Finally use the regular rational readback, StreamName window, and tail-ledger refusal rows. The rows R, A, H, C, P, N preserve the readback, seal, transport, replay, provenance, and naming boundary, while the refusal theorem excludes quotient equality, choice selectors, selected limits, selected representatives, ambient completion, and $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1564.26 (FrechetFilter root obligation triad)**.** *For every accepted* $\mathsf{FrechetFilter}^{\uparrow}$ *carrier* $F = (U, T, S, M, B, Q, R, A, H, C, P, N)$*, the unary threshold row, finite-tail row,* $\mathsf{FilterBase}^{\uparrow}$ *meet row,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *consumer route form one displayed root obligation surface. Any root consumer must read that surface through*

$$U, T, B \longmapsto Q \longmapsto R, A \longmapsto H, C, P, N$$

with the $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *handoff attached to the same finite route. The surface exports no quotient filter equality, selected limit, countable-choice tail selection, selected representative, or ambient completion.*

Proof. Begin with carrier admission. By Definition 1564.3, the rows $U, T, S, M, B, Q, R, A, H, C, P, N$ are the whole admitted source surface, so the unary threshold, finite-tail row, meet row, handoff, readback, seal, and structural coordinates have no hidden companion row.

Use cofinal-tail stability next. The theorem Definition 1564.4 sends every pair of tail requests through the same $\mathsf{FilterBase}^{\uparrow}$ meet row B before any schedule or metric row is consumed.

Apply the convergence-consumer handoff of Definition 1564.5. It places the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ consumer route, and $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite cofinite-tail surface.

Finally use Definition 1564.23. The sequential-completion consumer exhausts the tail ledger rather than adding a separate completion coordinate, and the displayed refusal rows exclude quotient filter equality, selected limits, countable-choice tail selection, selected representatives, and ambient completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetFilter}^{\uparrow}$ packet is generated from unary thresholds, cofinite tail neighborhoods, StreamName schedule rows, Metric tolerance rows, FilterBase meet rows, CauchyFilter handoff rows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1564.1, Definition 1564.2, Definition 1564.3, Definition 1564.4, Definition 1564.5, Definition 1564.23, and Definition 1564.26 bind the finite cofinite-tail and Cauchy route through $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetFilter}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.FrechetFilterUp.FrechetFilterCauchySourceBoundary`

Bridge status: paperBridge

Not claimed: This chapter does not close Real completeness, quotient filter equality, selected limits, countable-choice selectors, ambient completion, or bridge checking.

Upgrade path: Mature closure requires a checked standard bridge theorem and downstream stability for the cofinite-tail filter route.

Chapter 1565

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$

Following Bishop and Bridges / Kelley-style completion-unit naturality, the $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ packet records the finite square in which a source row first enters the completion unit and is then read by completion consumers, while the parallel route reads the same source through the monad and real-completion surfaces. It sits over the unit route of chapter 1527, the monad surface of chapter 378, the real-completion boundary of chapter 360, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names the L10 completion-unit naturality square as finite BEDC data; it is not a quotient of streams, not a chosen representative, not countable choice, not ambient completeness, and not completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1565.1 (CauchyCompletionUnitNaturality carrier)**.** *A* $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, M, R, S, D, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route that sends the displayed source row into the completion-facing unit surface.*
2. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *route that reads the same unit source through the monad consumer surface.*
3. *R is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *route reached only after the unit and monad reads have exposed the same completion-facing source.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the two routes.*
5. *D is the joint* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared finite windows.*
6. *H records componentwise* hsame *transport for the unit route, monad route, real-completion route, stream-window row, dyadic ledger, regular rational readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for both sides of the square: unit then completion consumer on one side, monad and real-completion consumer on the other side.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected representatives, countable-choice schedules, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1565.2 (CauchyCompletionUnitNaturality NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ *carrier* $Q = (U, M, R, S, D, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitNatura}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed unit route, monad route, real-completion route, finite stream windows, dyadic tolerance ledger, regular rational readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local naming row;*
2. *unit-side exposure, requiring every source read to pass through the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row U before a completion-facing consumer can inspect it;*
3. *monad-side exposure, requiring the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row M and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *row R to read the same displayed finite windows and dyadic ledger as the unit-side route;*
4. *square stability, requiring componentwise* hsame *transport through H and replay through C to identify the two routes at the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger non-escape, excluding quotient streams, selected representatives, countable-choice schedules, ambient completeness, and* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1565.1. The projection exposes exactly the unit, monad, real-completion, stream-window, dyadic-ledger, transport, replay, provenance, and local naming rows, together with the regular readback and $\mathsf{Real}^{\uparrow}$ seal rows named by those displayed routes.

The unit-side exposure is inherited from the U row. The monad-side exposure is inherited from M and R, but the carrier forces both sides to use the same S and D rows before any read reaches the terminal seal.

Square stability is componentwise transport through H followed by replay through C. Since P and N package only the displayed rows of Q, no separate quotient, representative selector, choice schedule, ambient completeness row, or $\mathsf{Real}^{\uparrow}$ completeness row is available to a consumer. □

Theorem 1565.3 (CauchyCompletionUnitNaturality square)**.** *Let* $Q = (U, M, R, S, D, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ *packet. The route that first reads the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row U and then passes to a completion consumer reaches the same displayed* $\mathsf{Real}^{\uparrow}$ *seal as the route that reads the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row M and the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *row R. Both routes pass through the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface named by P, and the same structural rows H, C, P, N.*

Thus the L10 completion-unit square is a finite route equality at the displayed real-seal boundary:

$$U \longmapsto S \longmapsto D \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$
$$M \longmapsto R \longmapsto S \longmapsto D \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Proof. Start with the unit-side route. The U row admits the source only as a finite unit read, and the carrier of Q records that the next visible rows are the stream window S and dyadic ledger D.

Now read the monad and real-completion side. The M row supplies the completion-consumer surface, while R ties that surface to the real-completion route. The same packet records the same S and D rows for this side of the square.

The package row P names the regular rational readback and terminal $\mathsf{Real}^{\uparrow}$ seal used by both sides. Componentwise transport through H and replay through C identify the two displayed paths at that seal, and no additional coordinate remains in the carrier from which a different completion read could be produced. □

Theorem 1565.4 (CauchyCompletionUnitNaturality real-seal consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ *packet must read the square through*

$$U, M, R, S, D, H, C, P, N.$$

The consumer may inspect the unit route, the monad route, the real-completion route, the shared finite windows, the shared dyadic tolerance ledger, the named regular readback surface, the terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming rows. It obtains no quotient stream, no selected representative, no countable-choice schedule, no ambient completeness theorem, and no completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Let a consumer ask for a real-seal or completion-facing read of Q. Carrier admission exposes $U, M, R, S, D, H, C, P, N$ and no other row.

If the consumer follows the unit side, the read begins with U and then uses the shared stream and dyadic rows S, D. If it follows the monad side, the read begins with M and R and then uses the same S, D rows.

The regular readback surface and terminal $\mathsf{Real}^{\uparrow}$ seal are named by the provenance row P and reached only by replay through C. Transport through H preserves the same rows on both sides of the square.

Because the accepted packet has no quotient, selected-representative, choice, ambient-completeness, or $\mathsf{Real}^{\uparrow}$-completeness coordinate, every consumer read is confined to the displayed finite square. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1565.1, Definition 1565.2, Definition 1565.3, and Definition 1565.4 seed the finite L10 completion-unit naturality square over unit, monad, real-completion, stream-window, dyadic-ledger, regular-readback, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, countable-choice schedules, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, square stability, consumer-boundary replay, and ledger non-escape for the same packet.

Constructive story: A $\mathsf{CauchyCompletionUnitNaturality}^{\uparrow}$ packet is generated from a CauchyCompletionUnit route, a CauchyCompletionMonad route, a RealCauchyCompletion route, shared StreamName windows, a shared DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1566

A Concrete Naming Certificate for $\mathsf{CountableDenseSubset}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CountableDenseSubset}^{\uparrow}$ packet presents countable dense subsets as finite unary index windows over a displayed $\mathsf{Met}^{\uparrow}$ source. It sits downstream of chapter 57, reads the finite windows of chapter 126, the regular rational handoff of chapter 301, the dyadic ledger of chapter 291, and the distance seal of chapter 38, while providing the explicit density row consumed by chapter 1458 and chapter 1453. The chapter records an enumerated finite BEDC density surface, not a choice-selected dense set, not a quotient completion, and not an open-cover compactness or Real-completeness principle.

Definition 1566.1 (CountableDenseSubset carrier)**.** *A* $\mathsf{CountableDenseSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, A, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *source row whose point histories, distance classifier, and metric compatibility fields are the only source surface read by the density packet.*
2. *A is the indexed dense candidate family. Each visible unary index names one candidate point over the source row M.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *request-window row. It records the inspected point name, unary index budget, and requested tolerance scale.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance ledger. It stores the dyadic tolerance token together with the Real distance boundary that the request must witness.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row translating each inspected tolerance window into regular rational readback data before the Real distance boundary is consulted.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *distance seal certifying that the chosen indexed candidate lies within the tolerance recorded by T after the handoff R has been replayed.*
7. *H records componentwise* hsame *transport for the metric source, index family, request windows, tolerance ledger, regular handoff, distance seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite density request through M, A, W, T, R, E.*

9. *P is the* Pkg *provenance row over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableDenseSubset}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected dense point, quotient completion, open-cover compactness extraction, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1566.2 (CountableDenseSubset NameCert obligations)**.** *For every accepted* $\mathsf{CountableDenseSubset}^{\uparrow}$ *carrier $D = (M, A, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CountableDenseSubset}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed metric source, indexed candidate family, finite request windows, dyadic and Real tolerance ledger, regular rational handoff, Real distance seal, transport, continuation, provenance, and local naming row;*

2. *metric-source compatibility, requiring every candidate point named by A and every point inspected by W to live over the same* $\mathsf{Met}^{\uparrow}$ *row M;*

3. *countable-index visibility, requiring each density response to expose a finite unary index before any downstream consumer reads a point witness;*

4. *tolerance exactness, requiring the requested dyadic scale and its Real boundary to be read from the same ledger row T;*

5. *regular handoff, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to mediate the tolerance-window read before the distance seal E is consulted;*

6. *nonescape, excluding choice-selected dense points, quotient completions, open-cover compactness, ambient Real completeness, and host equality of completed points.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CountableDenseSubsetUp.TasteGate.CountableDenseSubsetTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1566.1. The projection exposes exactly the metric source row, indexed candidate family, finite request windows, tolerance ledger, regular rational handoff, Real distance seal, and structural rows H, C, P, N.

Metric-source compatibility is forced by the shared row M. The candidate family A and every request window in W are read only as histories over that row, so a density response cannot change the metric source while preserving the packet.

Countable-index visibility and tolerance exactness are part of the same finite replay. The continuation row C routes a request through the displayed index and window data before the dyadic and Real tolerance row T and the regular handoff R can be used.

Finally, the seal E is downstream of T and R, and transport by H preserves this dependency order. The provenance and local naming rows package only that route, so the carrier contains no choice-selected point, quotient completion, compactness extraction, host equality, or ambient completeness coordinate. □

Theorem 1566.3 (Enumerated approximation handoff)**.** *Let $D = (M, A, W, T, R, E, H, C, P, N)$ be an accepted* $\mathsf{CountableDenseSubset}^{\uparrow}$ *packet. Every finite density request factors through the route*

$$M, A, W, T, R, E \longmapsto H, C, P, N.$$

Thus a $\mathsf{SeparableMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer may read an approximation only after the indexed candidate row* A*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance ledger* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the* $\mathsf{Real}^{\uparrow}$ *distance seal* E *have all been displayed over the metric source* M*.*

Proof. Begin with a finite density request from a downstream consumer. Carrier admission gives the consumer access to the metric source and candidate rows, but the candidate is visible only through an index in the displayed family A.

Replay the continuation row C. The request inspects the StreamName window W, reads the dyadic and Real tolerance ledger T, passes through the regular rational handoff R, and only then reaches the Real distance seal E.

The rows H, P, N transport, package, and name the same route. Hence a SeparableMetric or MetricCompletion consumer receives only the enumerated approximation packet and no external dense-point choice, quotient completion, compactness extraction, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableDenseSubset}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableDenseSubset}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1566.1, Definition 1566.2, and Definition 1566.3 seed the finite indexed density surface over a displayed $\mathsf{Met}^{\uparrow}$ source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableDenseSubset}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completions, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires carrier admission, metric-source compatibility, countable-index visibility, tolerance exactness, regular handoff, approximation handoff, and nonescape proofs for the same packet.

Constructive story: A $\mathsf{CountableDenseSubset}^{\uparrow}$ packet is generated from a MetricUp source, an indexed candidate family, StreamName request windows, a DyadicRatCore and RealUp tolerance ledger, a RegSeqRat handoff, a RealUp distance seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1567

A Concrete Naming Certificate for $\mathsf{RealApartnessCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealApartnessCompletion}^{\uparrow}$ packet records the separated completion boundary for $\mathsf{Real}^{\uparrow}$ names when apartness evidence is the visible equality discriminator. It reads the apartness surface of chapter 304, the Hausdorff separation boundary of chapter 402, the constructive completion route of chapter 1556, the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the displayed boundary between regular Cauchy completion data and apartness-separated real reads; it does not assert completeness of $\mathsf{Real}^{\uparrow}$ or import quotient equality, selected limits, countable choice, trichotomy, or host real equality.

Definition 1567.1 (RealApartnessCompletion carrier)**.** *A* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Q, M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{ApartnessReal}^{\uparrow}$ *source row, exposing apartness evidence for the real-name boundary under inspection.*
2. *Q is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separation row attached to the same source read.*
3. *M is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *completion-route row through which the separated completion request is replayed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by the completion read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback extracted from those stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback rows at the requested precision.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the apartness, separation, completion, window, readback, and tolerance rows have been displayed.*
8. *H records componentwise* hsame *transport for the apartness row, separation row, completion route, window schedule, readback row, tolerance ledger, real seal, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from apartness inspection through Hausdorff separation, constructive completion, stream windows, regular readback, dyadic tolerance, and then the real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RealApartnessCompletion}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, host real equality, selected limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionTasteGate single carrier alignment`

Theorem 1567.2 (RealApartnessCompletion obligation surface)**.** *For every accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *carrier* $A = (X, Q, M, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealApartnessCompletion}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission for the displayed apartness, Hausdorff separation, constructive completion, stream-window, regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows;*
2. *separated-source exactness, requiring the* $\mathsf{ApartnessReal}^{\uparrow}$ *row X and* $\mathsf{HausdorffCompletion}^{\uparrow}$ *row Q to be read before any completion-facing consumer can inspect M;*
3. *regular readback discipline, requiring every completion observation to pass through S, R, and D before the* $\mathsf{Real}^{\uparrow}$ *seal E is available;*
4. *componentwise stability, requiring H and C to transport and replay only the displayed rows $X, Q, M, S, R, D, E, P, N$;*
5. *ledger exhaustion, requiring the packet to export no quotient completion, host real equality, selected limit, countable-choice schedule, trichotomy row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionObligationSurface`

Proof. Carrier admission is projection from Definition 1567.1. The tuple displays exactly the apartness row, Hausdorff separation row, constructive completion route, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name.

Separated-source exactness is forced by the order of the visible route. The completion row M is named only after the apartness row X and the Hausdorff separation row Q have been displayed, so a consumer cannot read completion data before seeing the separated source surface.

The regular readback discipline is the role of S, R, D, E. The seal row E is downstream of the finite windows S, regular readback R, and dyadic tolerance ledger D, which keeps each Real-facing observation attached to the displayed regular-Cauchy route.

Componentwise stability and ledger exhaustion follow from H, C, P, N. These rows transport, replay, package, and name only the displayed packet, so they cannot add quotient completions, host real equality, selected limits, countable choice, trichotomy, or ambient completeness. □

Theorem 1567.3 (RealApartnessCompletion obligation routing)**.** *Every accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet routes each local* NameCert *obligation through*

$$X, Q, M, S, R, D, E, H, C, P, N$$

in that order. Carrier admission, separated-source exactness, regular readback discipline, componentwise stability, and ledger exhaustion are exactly the five projections of this finite route over the ApartnessReal$^{\uparrow}$, HausdorffCompletion$^{\uparrow}$, ConstructiveMetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionObligationRouting`

Proof. Project the carrier of Definition 1567.1. The displayed tuple places the apartness source, Hausdorff separation row, constructive completion route, stream windows, regular readback, dyadic tolerance ledger, and Real seal before the structural rows.

Apply the obligation surface of Definition 1567.2. Its five clauses are carrier admission, separated-source exactness, regular readback discipline, componentwise stability, and ledger exhaustion, so each clause is a projection of the same ordered route.

Finally use the non-escape boundary of Definition 1567.5. Since quotient completion, choice, trichotomy, host equality, and ambient completeness rows are absent, the obligation surface has no route outside the displayed packet. □

Theorem 1567.4 (RealApartnessCompletion obligation partition)**.** *For every accepted* RealApartnessCompletion$^{\uparrow}$ *packet* $A = (X, Q, M, S, R, D, E, H, C, P, N)$*, the local obligation surface partitions into separated-source rows* (X, Q, M)*, finite window and readback rows* (S, R, D)*, the terminal seal row* E*, and structural ledger rows* (H, C, P, N)*. No consumer may discharge an obligation in one partition by borrowing a row from another partition unless an explicit displayed transport or replay row in* H *or* C *carries that row across the named boundary.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionObligationPartition`

Proof. Project the carrier of Definition 1567.1. The tuple displays the apartness source X, Hausdorff separation row Q, constructive completion row M, stream window S, regular readback R, dyadic tolerance row D, seal E, and structural rows H, C, P, N as distinct coordinates.

Apply the obligation surface of Definition 1567.2 and the routing theorem of Definition 1567.3. Their carrier, separated-source, regular readback, stability, and ledger clauses are projections of the same ordered packet, so the source rows, window/readback rows, seal row, and structural rows cannot be identified by prose alone.

The non-escape boundary of Definition 1567.5 prevents quotient completion, selected limits, choice, trichotomy, host equality, and ambient completeness from supplying substitute rows. The separated-source exactness and regular readback discipline theorems, Definition 1567.13 and Definition 1567.14, fix the entry to M through X, Q and the entry to E through S, R, D.

It remains only to account for permitted movement between partitions. Such movement must be displayed by H or C, because those are the only transport and replay rows in the carrier. Hence a consumer cannot satisfy one partition by borrowing from another unless the borrowing is itself recorded as an explicit structural route. □

Theorem 1567.5 (RealApartnessCompletion non-escape boundary)**.** *Let* $A = (X, Q, M, S, R, D, E, H, C, P, N)$ *be an accepted* RealApartnessCompletion$^{\uparrow}$ *packet. Every separated completion read of* A *factors through the finite route*

$$X, Q, M, S, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus the boundary can expose apartness-separated real-name evidence only through the displayed ApartnessReal$^{\uparrow}$, HausdorffCompletion$^{\uparrow}$, ConstructiveMetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$,

DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. It excludes quotient completions, host real equality, selected limits, countable choice, trichotomy, and ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionNonescape`

Proof. Begin with an accepted separated completion read. Carrier admission gives a single visible route from the apartness source X and Hausdorff separation row Q into the constructive completion row M.

Replay the observation path through the finite windows. The stream schedule S, regular rational readback R, and dyadic tolerance ledger D must be displayed before the Real seal E can be read.

Finally transport and package the same route through H, C, P, N. Since no other coordinate occurs in the carrier or obligation surface, the read cannot escape to quotient completions, host real equality, selected limits, countable choice, trichotomy, or ambient Real completeness. □

Definition 1567.6 (RealApartnessCompletion apartness-Cauchy window)**.** *For an accepted* RealApartnessCompleti *packet* $A = (X, Q, M, S, R, D, E, H, C, P, N)$*, the apartness-Cauchy window is the finite prefix*

$$W_{\mathrm{ac}}(A) = (X, Q, M, S, R, D).$$

Its rows are read with the following roles:

1. X *is the* ApartnessReal$^{\uparrow}$ *apartness source for the boundary read.*
2. Q *is the* HausdorffCompletion$^{\uparrow}$ *separated-completion witness attached to that source.*
3. M *is the* ConstructiveMetricCompletion$^{\uparrow}$ *route by which the completion request is replayed.*
4. S *is the* StreamName$^{\uparrow}$ *finite-window ledger for the inspected name.*
5. R *is the* RegSeqRat$^{\uparrow}$ *regular rational readback exposed by those windows.*
6. D *is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to compare the readback at the requested precision.*

The window has no coordinate for the Real$^{\uparrow}$ *seal* E *and no coordinate for quotient completion, trichotomy, selected limits, countable choice, host equality, or ambient completeness.*

Lean4 (definition) `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionApartnessCauchyWindow`

Definition 1567.7 (RealApartnessCompletion separated seal route)**.** *For an accepted* RealApartnessCompletion$^{\uparrow}$ *packet* $A = (X, Q, M, S, R, D, E, H, C, P, N)$*, the separated seal route is the finite terminal path*

$$\Sigma_{\mathrm{sep}}(A) = (X, Q, M, S, R, D, E).$$

It records the apartness source, Hausdorff separation row, constructive completion route, stream-window ledger, regular readback, dyadic tolerance ledger, and terminal Real$^{\uparrow}$ *seal as one ordered read. The route has no coordinate for a second source, quotient completion, host equality of completed names, selected limit, countable-choice schedule, trichotomy row, or ambient* Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionSeparatedSealRoute`

Theorem 1567.8 (RealApartnessCompletion separated seal determinacy)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P,$ and $A' = (X, Q, M, S, R, D, E', H', C', P', N')$ be accepted* RealApartnessCompletion$^{\uparrow}$ *packets with the same* ApartnessReal$^{\uparrow}$ *source row X, the same* HausdorffCompletion$^{\uparrow}$ *separation row Q, the same* ConstructiveMetricCompletion$^{\uparrow}$ *route M, the same stream windows S, the same* RegSeqRat$^{\uparrow}$ *regular readback R, and the same* DyadicRatCore$^{\uparrow}$ *tolerance ledger D. If both packets expose a terminal Real seal through the displayed obligation surface, then they expose the same separated terminal Real-seal route:*

$$\Sigma_{\text{sep}}(A) = (X, Q, M, S, R, D, E)$$

and

$$\Sigma_{\text{sep}}(A') = (X, Q, M, S, R, D, E')$$

are determined by the same apartness-Cauchy window, and no alternative seal-facing route is available outside that window.

Proof. First apply the obligation partition of Definition 1567.4. It separates the source rows (X, Q, M), finite window and readback rows (S, R, D), terminal seal row, and structural rows, so the two packets have the same analytic prefix before either seal is read.

Separated-source exactness fixes the entry to the constructive completion route. Since both packets display the same X, Q, M, neither packet can enter the terminal route through a quotient completion, host equality row, selected limit, or external completion object.

Regular readback discipline then fixes the seal-facing window. The shared S, R, D rows are the only rows through which a Real-facing read may reach a terminal seal, and Definition 1567.6 names this common prefix as W_{ac}.

Finally use the non-escape boundary and located Real seal non-escape theorem. The structural rows H, C, P, N and H', C', P', N' only transport, replay, package, and name the displayed packet; they cannot add a second source, second window, quotient completion, selected limit, countable-choice schedule, trichotomy row, host equality, or ambient completeness principle. Therefore the terminal Real-seal route is determined by the shared apartness-Cauchy window. □

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionSeparatedSealDeterminacy`

Theorem 1567.9 (RealApartnessCompletion separated completion handoff)**.** *Let $A = (X, Q, M, S, R, D, E, H, C,$ be an accepted* RealApartnessCompletion$^{\uparrow}$ *packet. Every completion-facing read of A that reaches the* Real$^{\uparrow}$ *seal E first factors through the apartness-Cauchy window*

$$X, Q, M, S, R, D.$$

Thus no consumer can inspect E as a separated completion seal without first reading the ApartnessReal$^{\uparrow}$ *source, the* HausdorffCompletion$^{\uparrow}$ *separation row, the* ConstructiveMetricCompletion$^{\uparrow}$ *route, the* StreamName$^{\uparrow}$ *windows, the* RegSeqRat$^{\uparrow}$ *readback, and the* DyadicRatCore$^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionSeparatedCompletionHandoff`

Proof. Project the accepted packet through Definition 1567.1. The seal row E is displayed only after the source, separation, completion route, window, readback, and tolerance rows have already appeared.

Apply the obligation surface of Definition 1567.2. Its separated-source exactness clause forces X, Q to be read before the completion row M, and its regular readback clause forces S, R, D to be read before E.

The handoff is therefore the finite prefix named in Definition 1567.6. Since the carrier contains no bypass coordinate, a completion-facing consumer cannot reach E by quotient completion, selected limit, countable choice, trichotomy, host equality, or ambient completeness. □

Theorem 1567.10 (RealApartnessCompletion RegSeqRat seal handoff)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N$ be an accepted* RealApartnessCompletion$^{\uparrow}$ *packet. Every Real-facing apartness-completion seal passes through the* StreamName$^{\uparrow}$ *windows S, the* RegSeqRat$^{\uparrow}$ *readback row R, and the* DyadicRatCore$^{\uparrow}$ *tolerance row D before the terminal seal E is exposed. The seal therefore remains downstream of the displayed apartness-Cauchy window and cannot be read through a quotient completion, selected limit, countable-choice schedule, trichotomy row, host equality, or ambient completeness principle.*

Proof. Start with Definition 1567.2. Its regular readback and ledger clauses place the finite window row S, regular rational readback row R, and dyadic tolerance row D in the displayed obligation route before the seal is available.

Apply Definition 1567.9. It identifies the seal-facing prefix as the apartness-Cauchy window X, Q, M, S, R, D, so the seal cannot be consumed before the StreamName, RegSeqRat, and DyadicRatCore rows have been read.

Finally use Definition 1567.11 and Definition 1567.18. The downstream Real-facing consumer receives only $W_{\mathrm{ac}}(A), E, H, C, P, N$, and the ledger has no remaining coordinate for a bypass principle. Hence every Real-facing seal handoff passes through S, R, D before E. □

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionRegSeqRatSeal`

Theorem 1567.11 (RealApartnessCompletion located Real seal non-escape)**.** *Let $A = (X, Q, M, S, R, D, E, H, C,$ be an accepted* RealApartnessCompletion$^{\uparrow}$ *packet. Every* LocatedReal$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of the seal receives only the displayed apartness-Cauchy window together with the seal, transport, replay, provenance, and local naming rows*

$$W_{\mathrm{ac}}(A), E, H, C, P, N.$$

The consumer receives no quotient completion, trichotomy, selected limit, countable-choice schedule, host equality row, or ambient Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionLocatedRealSealNonescape`

Proof. Begin with a LocatedReal$^{\uparrow}$ or Real$^{\uparrow}$ consumer of the seal. By Definition 1567.9, the consumer must first pass through $W_{\mathrm{ac}}(A)$ before it can read E.

After the seal is read, Definition 1567.5 routes the remaining observation through H, C, P, N. These rows transport, replay, package, and name the same displayed packet; they do not add a second completion object or an external equality principle.

Combining the handoff with the non-escape boundary gives exactly $W_{\mathrm{ac}}(A), E, H, C, P, N$. The excluded principles are absent from both the carrier coordinates and the local obligation surface, so no located-real or real-seal consumer can obtain them from the packet. □

Theorem 1567.12 (RealApartnessCompletion finite apartness-completion obligation)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* RealApartnessCompletion$^{\uparrow}$ *packet. Every apartness-completion obligation exported by A is exposed only through the* DyadicRatCore$^{\uparrow}$ *tolerance row D, the* StreamName$^{\uparrow}$ *finite-window row S, the* RegSeqRat$^{\uparrow}$ *readback row R, the* LocatedReal$^{\uparrow}$ *or* ApartnessReal$^{\uparrow}$ *comparison surface carried by X, Q, M, and the* Real$^{\uparrow}$ *seal E, followed by H, C, P, N.*

Consequently the packet exports no quotient real equality, trichotomy, selected Cauchy limit, countable-choice schedule, host equality row, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionFiniteApartnessCompletionObligati`

Proof. Project the accepted carrier through Definition 1567.1. The apartness source, Hausdorff separation row, constructive completion route, stream windows, regular readback, dyadic tolerance ledger, and Real seal are the only analytic rows before transport, replay, provenance, and local naming.

Apply Definition 1567.2. Its separated-source exactness and regular readback clauses force every completion read to pass through X, Q, M, S, R, D before E is available.

The separated completion handoff of Definition 1567.9 and the located Real seal non-escape theorem of Definition 1567.11 identify the downstream consumer surface as $W_{\mathrm{ac}}(A), E, H, C, P, N$. Since the excluded principles are absent from both the carrier rows and the local obligation surface, they cannot be exported by the apartness-completion obligation. □

Theorem 1567.13 (RealApartnessCompletion separated-source exactness)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet. Any completion-facing consumer that reads the constructive completion row M must first read the* $\mathsf{ApartnessReal}^{\uparrow}$ *source row X and the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separation row Q through the same componentwise transport and replay rows H, C. The consumer cannot enter through a quotient completion, host real equality, selected limit, countable-choice schedule, trichotomy row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 1567.1. The route displays X and Q before M, and the structural rows H, C transport and replay that same ordered packet.

Apply Definition 1567.2. Its separated-source exactness clause is precisely the assertion that X, Q are consumed before any completion-facing inspection of M.

The separated completion handoff of Definition 1567.9 keeps every later read of the seal downstream of the same prefix. Since Definition 1567.11 excludes all bypass rows at the seal, no completion-facing consumer can reach M through an external completion or equality principle. □

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionSeparatedSourceExactness`

Theorem 1567.14 (RealApartnessCompletion regular readback discipline)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet. Every Real-facing read of the seal E must pass through the stream-window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D. Thus E is available only as the terminal seal of the displayed regular-Cauchy route, not as a selected limit or host completion point.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionRegularReadbackDiscipline`

Proof. Start from Definition 1567.9. It identifies the apartness-Cauchy window X, Q, M, S, R, D as the finite prefix required before E can be read.

Apply Definition 1567.2. Its regular readback discipline clause names S, R, D as the rows that mediate every completion observation before the Real seal becomes visible.

Finally use Definition 1567.11. The downstream $\mathsf{LocatedReal}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives $W_{\mathrm{ac}}(A), E, H, C, P, N$ and no selected limit, quotient completion, trichotomy, countable-choice schedule, host equality row, or ambient completeness theorem. □

Theorem 1567.15 (RealApartnessCompletion located tail obligation)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet. Before any separated-completion tail of A is consumed, the packet must expose the located apartness source carried by X, Q, M, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger*

D, and the terminal $\mathsf{Real}^{\uparrow}$ *seal E. Thus every tail-facing read reaches the separated completion boundary only through*

$$X, Q, M, S, R, D, E, H, C, P, N.$$

It has no coordinate for countable choice, selected limits, quotient equality, trichotomy, or ambient Real completeness.

Proof. Project the carrier of Definition 1567.1. The displayed order places the apartness source and separated completion route in X, Q, M, then the finite window, regular readback, dyadic tolerance, and seal rows in S, R, D, E.

Use separated seal determinacy, separated completion handoff, and located Real seal non-escape. These results force every seal-facing consumer to pass through $W_{\mathrm{ac}}(A) = (X, Q, M, S, R, D)$ before the seal E is available, and they keep the same transport, replay, provenance, and naming rows visible downstream.

The regular readback discipline fixes the Real-facing part of the route through S, R, D. Therefore a separated-completion tail cannot be consumed before located apartness, finite windows, regular readback, dyadic tolerance, and the Real seal have all been exposed; the refused principles are absent from the carrier and from the local obligation surface. □

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionLocatedTailObligation`

Theorem 1567.16 (RealApartnessCompletion RegSeqRat refinement)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet. Any refinement of the separated tail that is visible to a Real-facing consumer factors through the existing* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D after the finite window row S has been read. The refinement may tighten the displayed regular rational readback along S, R, D, but it cannot export quotient equality, selected limits, trichotomy, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionRegSeqRatRefinement`

Proof. Begin with the located tail obligation above and the regular readback discipline. Together they identify S, R, D as the only finite refinement surface between the separated completion route M and the terminal seal E.

Apply separated-source exactness and separated completion handoff. The consumer cannot enter the refinement from a quotient completion or selected limit; it reaches M through X, Q and then reaches E only through $W_{\mathrm{ac}}(A)$.

Finally use located Real seal non-escape and ledger exhaustion. The rows H, C, P, N preserve the same displayed packet, while the seal row E remains downstream of S, R, D. Hence the refinement is a RegSeqRat-and-dyadic-tolerance refinement of the displayed tail, not an exported equality, trichotomy, choice, selected-limit, or ambient-completeness principle. □

Theorem 1567.17 (RealApartnessCompletion window determinacy)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet that reaches the* $\mathsf{Real}^{\uparrow}$ *seal E. Then the packet determines the same apartness-Cauchy prefix*

$$W_{\mathrm{ac}}(A) = (X, Q, M, S, R, D)$$

through the displayed $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows. No second window source, quotient completion, selected limit, countable-choice schedule, trichotomy row, or host equality principle can supply an alternate prefix for the same seal-facing read.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionWindowDeterminacy`

Proof. Start with the carrier definition and the apartness-Cauchy window definition. They name X, Q, M, S, R, D as the only prefix rows available before the seal coordinate E is read.

Apply the separated completion handoff, the located Real seal non-escape theorem, and separated-source exactness. These force every completion-facing route to enter through the apartness source, Hausdorff separation row, and constructive metric completion row before it can reach the terminal seal.

Next apply the regular readback discipline. It fixes the seal-facing observation through the finite stream window, regular rational readback, and dyadic tolerance ledger, so the same S, R, D rows mediate every Real-facing read.

Finally use ledger exhaustion. The remaining rows H, C, P, N only transport, replay, package, and name the displayed packet, leaving no coordinate for a second window source, quotient completion, selected limit, countable choice, trichotomy, or host equality. □

Theorem 1567.18 (RealApartnessCompletion ledger exhaustion)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *packet. The complete obligation ledger exported by A is the displayed carrier route together with its transport, replay, provenance, and local naming rows. Every separated-source, regular-readback, componentwise-stability, and seal-facing consumer is therefore exhausted by $X, Q, M, S, R, D, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.RealApartnessCompletionUp.RealApartnessCompletionLedgerExhaustion`

Proof. Use Definition 1567.3. It routes carrier admission, separated-source exactness, regular readback discipline, componentwise stability, and ledger exhaustion through the single finite packet $X, Q, M, S, R, D, E, H, C, P, N$.

The separated-source theorem above fixes the entry to M through X, Q, and the regular readback theorem above fixes the entry to E through S, R, D. These two routes are the analytic part of the same ledger.

The remaining rows are structural. By Definition 1567.5 and Definition 1567.11, H, C, P, N only transport, replay, package, and name the displayed packet, so the ledger has no coordinate left for quotient completion, selected limits, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApartnessCompletion}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1567.1, Definition 1567.2, Definition 1567.3, Definition 1567.4, Definition 1567.5, Definition 1567.6, Definition 1567.9, Definition 1567.10, Definition 1567.11, Definition 1567.13, Definition 1567.14, Definition 1567.15, Definition 1567.16, and Definition 1567.18 close the displayed apartness-completion obligation surface over ApartnessReal, HausdorffCompletion, ConstructiveMetricCompletion, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApartnessCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host real equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Public closure requires a public-object surface and, independently, a verified Lean target for BEDC.Derived.RealApartnessC

Constructive story: A $\mathsf{RealApartnessCompletion}^{\uparrow}$ packet is generated from an ApartnessReal source row, a HausdorffCompletion separation row, a ConstructiveMetricCompletion route, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1568

A Concrete Naming Certificate for $\mathsf{CauchyReindexing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyReindexing}^{\uparrow}$ packet records the finite certified route by which a regular Cauchy name may be inspected along a displayed monotone cofinal schedule before it reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits between the finite observation schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite NameCert surface for changing observation indices; it is not quotient-stream equality, not a selected limit, not countable choice, not host real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1568.1 (CauchyReindexing carrier)**.** *A* $\mathsf{CauchyReindexing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, I, M, Q, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *schedule row whose finite windows are available to the packet;*
2. *I is the target* $\mathsf{StreamName}^{\uparrow}$ *schedule row obtained by reading S along a displayed unary index map;*
3. *M is the monotone row for that unary index map, recording that accepted target windows never ask for source observations in reverse order;*
4. *Q is the cofinal row for the same unary map, recording that every displayed source tolerance budget is eventually reached by some displayed target window;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the source and target observations at the finite budgets named by M and Q;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row showing that the target schedule remains a regular rational name after the reindexing route has been replayed;*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, I, M, Q, D, R have been consumed;*
8. *H records componentwise* hsame *transport for the source schedule, target schedule, monotone row, cofinal row, dyadic tolerance row, regularity row, Real seal, continuation, provenance, and local naming rows;*

9. *C records displayed* Cont *replay from a target-window request through the unary reindexing rows, back to the source schedule, then forward through dyadic tolerance, regular readback, and the Real seal;*
10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyReindexing}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, countable-choice selection of a subsequence, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 1568.2 (CauchyReindexing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *carrier $K = (S, I, M, Q, D, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyReindexing}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source schedule, reindexed schedule, monotone row, cofinal row, dyadic tolerance row, regularity row, Real seal, transport, continuation, provenance, and local naming row;*
2. *index admissibility, requiring every target-window read in I to replay through the displayed unary map before it can consult the source schedule S;*
3. *monotone and cofinal discipline, requiring M and Q to be read together before a consumer treats the target schedule as a regular Cauchy name;*
4. *tolerance and regularity preservation, requiring D and R to certify the finite regular readback after the reindexing route has been replayed;*
5. *seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row A to remain downstream of S, I, M, Q, D, R;*
6. *ledger refusal, excluding quotient streams, selected limits, countable-choice selectors, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1568.1. The projection exposes exactly the source schedule, target schedule, monotone row, cofinal row, dyadic tolerance row, regularity row, Real seal, and structural rows H, C, P, N.

Index admissibility is forced by the continuation row C. A target-window request reaches I only by replaying the displayed unary index route back to the source schedule S, so the packet has no independent target stream coordinate.

The monotone and cofinal rows are then the only rows that justify treating the target schedule as a regular Cauchy inspection surface. The dyadic tolerance row D and regularity row R sit downstream of those two rows, so tolerance preservation and regular readback cannot be exported before the reindexing discipline has been consumed.

Finally, the seal row A is downstream of the same route. Transport by H preserves each listed coordinate, while P and N package only that finite route; hence no quotient stream, selected limit, countable-choice selector, host real equality, or ambient completeness coordinate remains available. □

Theorem 1568.3 (CauchyReindexing RegSeqRat to Real handoff)**.** *Let $K = (S, I, M, Q, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read of the reindexed regular name factors through the finite route*

$$S, I, M, Q, D, R \longmapsto A \longmapsto H, C, P, N.$$

Thus the $\mathsf{RegSeqRat}^{\uparrow}$ *row reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after the source* $\mathsf{StreamName}^{\uparrow}$ *schedule, reindexed* $\mathsf{StreamName}^{\uparrow}$ *schedule, monotone row, cofinal row, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row have been displayed. The route contains no quotient stream, selected limit, countable-choice selector, host real equality, or ambient Real completeness principle.*

Proof. Begin with a Real-facing consumer of the packet. Carrier admission gives access to the seal row A, but only as a row downstream of the regularity row R and the dyadic tolerance row D.

Replay the continuation row C. The request first reads the target schedule I as a displayed reindexing of the source schedule S, then consumes the monotone row M and cofinal row Q that make the index change admissible for regular Cauchy inspection.

After those rows have been consumed, the dyadic tolerance row D and regularity row R provide the finite readback accepted by the Real seal. The transport, provenance, and local naming rows record the same dependency order, so the packet exports no direct route through quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 1568.4 (CauchyReindexing composition stability)**.** *Let* K_{01} *be an accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *packet from a source schedule* S_0 *to a middle schedule* S_1*, and let* K_{12} *be an accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *packet from* S_1 *to a target schedule* S_2*. Suppose that the two packets display the same middle* $\mathsf{StreamName}^{\uparrow}$ *schedule row and that their middle-facing monotone rows, cofinal rows, dyadic tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rows,* $\mathsf{Real}^{\uparrow}$ *seal rows,* hsame *transports,* Cont *routes,* Pkg *provenance rows, and local* $\mathsf{NameCert}_{\mathsf{CauchyReindexing}^{\uparrow}}$ *rows agree on that shared schedule. Then the finite route obtained by composing the displayed unary index maps from* S_0 *through* S_1 *to* S_2 *is again an accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *surface, with carrier rows in the same order*

$$S, I, M, Q, D, R, A, H, C, P, N.$$

Its downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers factor through the composed monotone–cofinal route and acquire no quotient stream, selected limit, countable-choice selector, host real equality, or ambient completeness coordinate.*

Proof. Project the two accepted carriers. The target schedule row of K_{01} and the source schedule row of K_{12} are the same displayed middle schedule, so their endpoint data can be read as one finite route from S_0 to S_2.

Compose the two unary index routes at the level of the displayed monotone and cofinal rows. Monotonicity is preserved because a target-window request first moves monotonically along the second route and then monotonically along the first route; cofinality is preserved because any source tolerance requested at S_0 is reached by the first route and then by the second route through the shared middle schedule.

The shared dyadic tolerance rows and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows are reused along the composed route. Componentwise hsame transport identifies the common middle schedule and its tolerance, regularity, seal, continuation, package, and local naming rows, while Cont concatenates the two finite replays without adding a new coordinate.

Finally apply Definition 1568.3 to the repacked route. The Real-facing consumer still reaches the seal only after reading the source schedule, target schedule, composed monotone row, composed cofinal row, dyadic tolerance row, and regularity row, so the composed surface has the same carrier order and the same ledger refusals as an accepted $\mathsf{CauchyReindexing}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyReindexing}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyReindexing}^{\uparrow}$ packet is generated from source and target StreamName schedules, monotone and cofinal unary reindexing rows, a DyadicRatCore tolerance row, a RegSeqRat regularity row, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyReindexing}\uparrow}$, seed)

Scope closed: Definition 1568.1, Definition 1568.2, Definition 1568.3, and Definition 1568.4 seed the finite certified reindexing route from regular Cauchy names to the downstream Real$^{\uparrow}$ seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyReindexing}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable-choice selectors, host real equality, ambient Real$^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, monotone and cofinal reindexing discipline, dyadic tolerance preservation, RegSeqRat readback, Real seal factorization, and ledger refusal.

Chapter 1569

A Concrete Naming Certificate for $\mathsf{CauchyFilterCompletion}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ packet records the finite completion boundary generated by Cauchy filters before any completed Real seal is exposed. It sits between the saturated filter surface of chapter 333, the limit-facing packet of chapter 1558, the finite windows of chapter 126, the regular readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Real seal boundary of chapter 38, and the uniform completion surface of chapter 1473. The packet exposes a finite completion route, not a quotient filter, selected limit, host equality, countable-choice selector, or ambient completeness theorem.

Definition 1569.1 (CauchyFilterCompletion carrier)**.** *A* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, W, T, R, L, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row naming the saturated Cauchy-filter data consumed by the completion request.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows through which the request observes the filter at displayed precision levels.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and dyadic tolerances.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after C, W, T, R have been replayed.*
6. *H records componentwise* hsame *transport for the filter, window, tolerance, readback, seal, continuation, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a Cauchy-filter request through finite windows, dyadic tolerance, regular readback, and the Real seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient filters, selected limits, host real equality, countable-choice selection, ambient completeness, or a separate uniform-completion object beyond the displayed provenance row.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionTasteGate single carrier alignment`

Theorem 1569.2 (CauchyFilterCompletion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed Cauchy filter, finite windows, dyadic tolerance ledger, regular readback, Real seal handoff, transport, continuation, provenance, and local naming row;*
2. *filter-to-window exposure, requiring* C *to be consumed before any completion-facing read inspects* W, T, R*, or* L*;*
3. *tolerance exactness, requiring every accepted readback in* R *to use the displayed dyadic ledger* T *for the same window row* W*;*
4. *completion handoff, requiring the Real seal row* L *to remain downstream of* C, W, T, R *and the structural rows* H, K, P, N*;*
5. *ledger refusal, excluding quotient filters, selected limits, host real equality, countable-choice selectors, ambient completeness, and detached uniform-completion coordinates.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket semantic name certificate`

Proof. Project the carrier of Definition 1569.1. The public rows are exactly $C, W, T, R, L, H, K, P, N$, so carrier admission exposes the Cauchy-filter row, the finite observation route, the Real seal handoff, and the structural rows without adding quotient, selector, host-equality, or completeness data.

The filter-to-window and tolerance obligations come from the displayed continuation row K. A completion-facing request must first consume C, then inspect the finite windows W, then read the dyadic ledger T before the regular readback row R is available.

The completion handoff is the remaining public route. The row L is downstream of C, W, T, R, while H, P, and N transport, package, and name only that route. Thus every excluded object lies outside the accepted packet. □

Theorem 1569.3 (CauchyFilterCompletion Real seal handoff)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Every completion-facing or Real-facing consumer that reads the terminal seal factors through the finite route*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

Thus the $\mathsf{Real}^{\uparrow}$ *seal handoff is available only after the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed. The same route is compatible with the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary and the* $\mathsf{UniformCompletion}^{\uparrow}$ *provenance surface, but it exports no quotient filter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or detached uniform-completion coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal handoff`

Proof. Begin with a consumer of the terminal seal. Carrier admission gives access to L only as a row of the accepted packet F.

Replay the continuation row K. The request reads the saturated Cauchy-filter row C, the finite windows W, the dyadic tolerance ledger T, and the regular rational readback R before the Real seal handoff can be consumed.

The structural rows preserve this order. Transport through H, provenance through P, and local naming through N record the same route, so the consumer has no available coordinate for quotient filters, selected limits, host equality, choice, ambient completeness, or an independent uniform-completion object. □

Theorem 1569.4 (CauchyFilterCompletion window-tolerance totality)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *packet. Every accepted completion read opens the finite window row W and the dyadic tolerance row T before it can read the regular rational row R. Thus no completion-facing consumer reaches R by a detached stream window, a free tolerance parameter, a quotient filter, or a selected filter limit.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket window readback transport`

Proof. Project the carrier of Definition 1569.1. The regular readback row R is part of the same finite packet as C, W, T, L, H, K, P, N.

Replay the continuation row K. It exposes the Cauchy-filter source C, then the finite windows W, then the dyadic tolerance ledger T before any regular readback is available.

The Real seal handoff of Definition 1569.3 keeps every terminal read downstream of that route. Hence a consumer that bypasses W or T is not reading the accepted CauchyFilterCompletion$^{\uparrow}$ packet. □

Theorem 1569.5 (CauchyFilterCompletion uniform-filter readback)**.** *Every* UniformCauchyFilter$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer of an accepted* CauchyFilterCompletion$^{\uparrow}$ *packet reads only through*

$$C, W, T, R, L, H, K, P, N.$$

The uniform-facing route may use the saturated Cauchy-filter row, finite observation windows, dyadic tolerance ledger, regular readback, Real seal, transport, continuation, provenance, and local naming rows, but it cannot introduce a detached uniform-completion object.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform filter readback`

Proof. Start with a uniform-facing consumer. The only Cauchy source in the carrier is C, and the only completion-facing terminal row is L.

By Definition 1569.4, the consumer reaches R only after W and T have been displayed. By Definition 1569.3, the Real seal is then read through L and the structural rows H, K, P, N.

Since no other row exists in the carrier, UniformCauchyFilter$^{\uparrow}$ and UniformCompletion$^{\uparrow}$ readback is exactly this finite route. Quotient filters, selected limits, ambient completeness, and detached completion coordinates are outside the packet. □

Theorem 1569.6 (CauchyFilterCompletion Real seal uniqueness route)**.** *All* Real$^{\uparrow}$ *seal reads exported by an accepted* CauchyFilterCompletion$^{\uparrow}$ *packet coincide at the displayed row L. No accepted read can replace L by quotient-filter equality, a selected limit, host real equality, a countable-choice selector, or a separate completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRealSealUniquenessRoute`

Proof. Begin with two Real-facing reads of the same packet $F = (C, W, T, R, L, H, K, P, N)$. Carrier projection gives a single terminal Real seal row, namely L.

The window-tolerance route forces both reads through C, W, T, R before L is available. The continuation and transport rows K, H preserve that route rather than choosing new representatives.

Provenance and local naming through P, N record the same displayed seal. Therefore all accepted Real readbacks meet at L, and the excluded quotient, selector, equality, choice, and completeness objects have no coordinate inside the packet. □

Theorem 1569.7 (CauchyFilterCompletion located-uniform gluing)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Suppose a* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *real-seal read and a* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *completion handoff are both consumed through this packet. Then the two reads have the same finite BEDC route inside F:*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

The located read cannot reach the $\mathsf{Real}^{\uparrow}$ *seal by any row other than the displayed L, and the uniform handoff cannot reach a completion-facing consumer by any row detached from the displayed $C, W, T, R, L, H, K, P, N$. Thus the joint* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumption glues the located-filter and uniform-filter surfaces through one packet, not through quotient filters, selected limits, host equality, countable choice, ambient completeness, or a separate completion object.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket located uniform gluing`

Proof. Start with the carrier of Definition 1569.1. It displays only the rows $C, W, T, R, L, H, K, P, N$, so both imported reads must be interpreted through those coordinates once they are consumed as $\mathsf{CauchyFilterCompletion}^{\uparrow}$ data.

Apply Definition 1569.2 to fix the local obligation surface. Filter-to-window exposure and tolerance exactness force any completion-facing read to consume C, then the window and tolerance rows W, T, before the regular readback row R is available.

Apply Definition 1569.3. The terminal seal read is downstream of C, W, T, R and upstream of the structural rows H, K, P, N, so the packet exposes exactly the displayed route.

The imported boundary theorems match this route. The $\mathsf{LocatedCauchyFilter}^{\uparrow}$ real-seal read is constrained by Definition 1661.6, while the $\mathsf{UniformCauchyFilter}^{\uparrow}$ completion handoff is constrained by Definition 1694.3. Since both are consumed through the same accepted packet F, their public exits coincide at L and then pass through the same transport, continuation, provenance, and local naming rows.

No excluded object can mediate the gluing. Quotient filters, selected limits, host equality, countable choice, ambient completeness, and detached completion objects are absent from the carrier and refused by the NameCert obligations, so there is no second route available inside the packet. □

Theorem 1569.8 (CauchyFilterCompletion filterbase handoff)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Every accepted completion read first consumes the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{FilterBase}^{\uparrow}$ *rows behind C before it can inspect the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal L.*

Consequently the completion route factors through the finite handoff

$$C_{\mathsf{FilterBase}^{\uparrow}}, C_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto W, T, R \longmapsto L \longmapsto H, K, P, N.$$

No accepted read may bypass the displayed filterbase row by using a detached stream window, a free dyadic tolerance, a selected filter limit, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filterbase handoff`

Proof. Project the carrier of Definition 1569.1. The row C is the only Cauchy-filter coordinate, and its admissible source is the displayed filterbase route used by the saturated Cauchy-filter packet.

Replay the continuation row K. A completion request reaches W only after C has been consumed, reaches T only through the same windows, reaches R only through that tolerance ledger, and reaches L only after the regular readback is displayed.

The structural rows preserve the route. Transport through H, provenance through P, and local naming through N record the same filterbase-first handoff, so detached windows, free tolerances, selected filter limits, and ambient completeness have no coordinate inside the accepted packet. □

Theorem 1569.9 (CauchyFilterCompletion finite-window induction)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Every induction over the finite filterbase windows of F preserves the displayed Cauchy tolerance row T, the located-uniform readback rows through R, the separatedness boundary carried by the terminal* $\mathsf{Real}^{\uparrow}$ *seal L, and the Real-seal handoff rows H, K, P, N. The induction may only use the finite* BHist *support of W, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it cannot insert quotient filters, selected limits, host equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket finite window induction`

Proof. Start from Definition 1569.8. It places the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source before every finite window, so induction on the window spine begins with a fixed source row C.

At each finite step, use Definition 1569.4. The window row being inspected carries the same dyadic tolerance ledger T, and the readback row R remains downstream of that ledger.

The terminal step is governed by Definition 1569.3 and Definition 1569.7. They keep located, uniform, separatedness, and Real-seal consumers on the same $C, W, T, R, L, H, K, P, N$ route.

Finally the structural rows H, K, P, N transport, replay, source, and name only the displayed packet. Since the refused objects are absent from the carrier and from the NameCert obligation surface, the finite-window induction preserves exactly the listed rows. □

Theorem 1569.10 (CauchyFilterCompletion uniform-limit compatibility)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Every* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{UniformCauchyFilter}^{\uparrow}$*, or separated-completion consumer of F that also reads a uniform-limit or separated-seal boundary must use the same finite completion route*

$$C, W, T, R, L, H, K, P, N.$$

The consumer cannot replace L by quotient-filter equality, selected limits, host real equality, countable choice, or a detached completion object.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform limit compatibility`

Proof. First use Definition 1569.4. It places the window row W and dyadic tolerance row T before the readback row R for every accepted completion read.

Next apply Definition 1569.3. The terminal seal row L is reached only after C, W, T, R and is then transported, replayed, sourced, and named by H, K, P, N.

For uniform-facing consumers, use Definition 1569.5. It identifies the $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{UniformCauchyFilter}^{\uparrow}$ readback route with the same displayed rows $C, W, T, R, L, H, K, P, N$.

Finally apply Definition 1569.16. Quotient-filter equality, selected limits, host real equality, countable choice, and detached completion objects have no coordinate inside F, so a uniform-limit or separated-seal boundary cannot substitute them for the displayed route. □

Theorem 1569.11 (CauchyFilterCompletion filter-convergence handoff)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Every accepted convergence-facing read that uses a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary must read that convergence row only after the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *source C, finite* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R have been exposed. The handoff is therefore confined to*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

No convergence consumer may replace this route by a selected filter limit, quotient filter, host equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filter convergence handoff`

Proof. Start with Definition 1569.4. It places every accepted regular readback behind the displayed window row and dyadic tolerance ledger.

Apply Definition 1569.3. The terminal seal and any convergence-facing handoff are downstream of the Cauchy-filter source, finite windows, tolerance ledger, and regular readback.

Finally use Definition 1569.5 and Definition 1569.16. The first keeps downstream completion consumers on the same finite route, while the second excludes selected limits, quotient filters, host equality, choice, and ambient completeness from the packet. □

Theorem 1569.12 (CauchyFilterCompletion separated-completion boundary)**.** *Let $F = (C, W, T, R, L, H, K, P, N$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Any separated-completion uniqueness consumer of F may use only the displayed* $\mathsf{Real}^{\uparrow}$ *seal row L and the componentwise* hsame *transport row H after the finite source route C, W, T, R has been read. It cannot use quotient filters, selected limits, host real equality, countable choice, or an ambient separated-completion theorem as a hidden coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket separated completion boundary`

Proof. First apply Definition 1569.4. It forces the separated-completion consumer to reach any regular readback through the displayed windows and tolerances.

Next use Definition 1569.3. The only terminal completion row available to the consumer is the displayed seal L, reached from C, W, T, R.

The uniqueness comparison is then transported only by H and replayed through K, P, N. By Definition 1569.16, no quotient-filter equality, selected limit, host equality, choice principle, or ambient separated-completion theorem can enter the boundary. □

Theorem 1569.13 (CauchyFilterCompletion uniform-completion comparison)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet compares completion data through the same displayed route*

$$C, W, T, R, L, H, K, P, N.$$

The comparison reads the Cauchy-filter source, finite windows, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows from one packet; it cannot replace them by a detached uniform-completion carrier or by quotient-filter, selected-limit, host-equality, countable-choice, or ambient-completeness data.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform completion comparison`

Proof. Begin with Definition 1569.5. It identifies the uniform-facing readback route with the displayed rows $C, W, T, R, L, H, K, P, N$.

Use Definition 1569.4 and Definition 1569.3 to locate the comparison source. The windows and tolerances precede R, and the seal L remains downstream of the same source route.

Finally apply Definition 1569.16. Detached uniform-completion carriers and the excluded quotient, selector, host-equality, choice, and completeness rows are absent from the carrier, so the comparison cannot leave the displayed route. □

Theorem 1569.14 (CauchyFilterCompletion p-variation window bound)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. Any bounded-variation or Young-integral consumer that reads a finite p-variation window from F must obtain that window through the displayed route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may reuse the rows $C, W, T, R, L, H, K, P, N$ as finite variation-window data, but it receives no quotient filter, choice-selected limit, host real equality, ambient Real completeness, detached variation seminorm, or hidden refinement selector.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket pvariation window bound`

Proof. Begin with the window and tolerance route. By Definition 1569.4, every accepted completion read opens the stream-window row W and dyadic tolerance ledger T before it can inspect the regular readback row R.

Use the filterbase handoff of Definition 1569.8. The source of those windows is the displayed $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ route behind C, so a downstream variation consumer cannot begin from a detached family of partitions.

Now hand the finite windows to the bounded-variation side. The $\mathsf{BoundedVariation}^{\uparrow}$ read may consume the displayed window and tolerance data as a finite control ledger, but it must keep the same C, W, T, R route and the same structural rows H, K, P, N.

Finally apply Definition 1569.3 and Definition 1569.16. The Real seal remains the displayed row L, and quotient filters, selected limits, host equality, choice, ambient completeness, detached variation data, and hidden refinement selectors have no coordinate in the packet. □

Theorem 1569.15 (CauchyFilterCompletion regulated-integral handoff)**.** *Every regulated-integral or* $\mathsf{YoungIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet reads completion data only through the finite source route $C, W, T, R, L, H, K, P, N$. In particular, the consumer may pass the stream windows, dyadic tolerances, regular readback, and Real seal to* $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, *and* $\mathsf{YoungIntegral}^{\uparrow}$ *rows, but it cannot import quotient filters, choice-selected limits, ambient Real completeness, host equality of real names, or a measure-completion integral.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket regulated integral handoff`

Proof. Start with the uniform-completion comparison theorem Definition 1569.13. It fixes every completion-facing comparison to the displayed rows $C, W, T, R, L, H, K, P, N$.

The regulated-function handoff reads only finite windows. The rows W, T, R supply the observed stream windows, dyadic tolerances, and regular rational readbacks required by a $\mathsf{RegulatedFunction}^{\uparrow}$ consumer, while L supplies the downstream Real seal.

The Young-integral consumer is then downstream of those rows. It may combine the regulated-function read with $\mathsf{BoundedVariation}^{\uparrow}$ and $\mathsf{RiemannSum}^{\uparrow}$ data, but the completion contribution remains the same finite route through $C, W, T, R, L, H, K, P, N$.

Close with Definition 1569.16. Since quotient filters, choice-selected limits, host equality, ambient Real completeness, and measure-completion integrals are absent from the carrier, no regulated-integral handoff can export them. □

Theorem 1569.16 (CauchyFilterCompletion quotient nonescape)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet. No accepted read of F can introduce quotient filters, selected limits, host equality of filters or real names, countable choice, ambient completeness, or a completion object not displayed by the carrier.*

Every consumer of F is confined to the rows $C, W, T, R, L, H, K, P, N$ and to the componentwise transport, continuation replay, provenance, and local naming generated by those rows.

Proof. Inspect the carrier of Definition 1569.1. Its coordinates are exactly the Cauchy-filter row, finite windows, dyadic tolerance ledger, regular readback, Real seal handoff, transport, continuation, provenance, and local naming row.

Apply Definition 1569.2. The ledger-refusal obligation excludes quotient filters, selected limits, host equality, countable-choice selectors, ambient completeness, and detached completion coordinates from the local certificate surface.

It remains to check that consumers cannot add them later. Any consumer of F receives only the displayed rows and their hsame, Cont, Pkg, and $\mathsf{NameCert}$ structure, so there is no exported field from which an excluded object could be read. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket quotient nonescape`

Theorem 1569.17 (CauchyFilterCompletion L10 source scope)**.** *The L10 source scope of the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *chapter is exactly*

$$\mathsf{CauchyFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},$$

The $\mathsf{FilterBase}^{\uparrow}$ *route is read only through the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row, and no quotient filter, selected limit, host equality, countable-choice selector, ambient completeness theorem, or detached completion carrier belongs to the scope.*

Proof. Read the carrier definition. The public coordinates $C, W, T, R, L, H, K, P, N$ name precisely the Cauchy-filter source, stream-window observation, dyadic tolerance, regular readback, Real seal, transport, continuation, provenance, and local naming data.

The $\mathsf{UniformCompletion}^{\uparrow}$ occurrence is provenance-level only. It appears in P as a downstream completion-facing surface, while the terminal read still passes through C, W, T, R, L and the structural rows.

The filterbase dependency is internal to C. By Definition 1569.8, every accepted read consumes that base before the stream and tolerance rows, but the chapter does not export a separate $\mathsf{FilterBase}^{\uparrow}$ coordinate beyond the Cauchy-filter source row.

The nonescape boundary is supplied by Definition 1569.16. Since excluded objects are absent from the carrier and refused by the local NameCert surface, the listed objects are the exact L10 source scope. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket l10 source scope`

Theorem 1569.18 (CauchyFilterCompletion Real seal nonescape)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every consumer that reaches the displayed* $\mathsf{Real}^{\uparrow}$ *seal row L must first pass through the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row C, the finite* $\mathsf{StreamName}^{\uparrow}$ *window family W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The rows H, K, P, N then transport, replay, source, and name that same route. No accepted read can escape from L into a selected limit, quotient filter, host real equality, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completion object.*

Proof. Project the carrier of Definition 1569.1. The row L is present only as the terminal seal handoff of the displayed tuple, and the preceding public coordinates are exactly C, W, T, R.

Apply Definition 1569.2 and Definition 1569.3. The local obligations force filter-to-window exposure, tolerance exactness, and regular readback before a completion-facing consumer may read the Real seal.

Finally use Definition 1569.16. The structural rows H, K, P, N preserve the same finite route, and the refused quotient, selector, host-equality, choice, completeness, and detached-completion objects have no coordinate inside F. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal nonescape`

Theorem 1569.19 (CauchyFilterCompletion uniform handoff)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Any* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer that uses F sees only the provenance-level handoff*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

The handoff reads the underlying completion request through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal; it does not introduce a second uniform-completion carrier, quotient filter, selected limit, choice principle, host equality, or ambient completeness theorem.*

Proof. Begin with the provenance row P in Definition 1569.1. The $\mathsf{UniformCompletion}^{\uparrow}$ occurrence is stored there as provenance for the displayed packet, not as an additional carrier coordinate.

Apply Definition 1569.7. A uniform-facing read consumed through F shares the finite route $C, W, T, R, L, H, K, P, N$ with the located and Real-facing reads already exposed by the packet.

Use Definition 1569.17 to close the source list. The only uniform-completion access is the provenance-level handoff through P, so no detached completion object, selected limit, quotient filter, choice selector, host equality, or ambient completeness row can be exported. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform handoff`

Theorem 1569.20 (CauchyFilterCompletion filterbase window exactness)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every filterbase-facing completion read consumes the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row behind C before it inspects a window in W, and every accepted window read uses the matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in T before producing the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. Thus the filterbase-to-window route is exact inside the packet:*

$$C_{\mathsf{FilterBase}^{\uparrow}}, C_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto W, T \longmapsto R \longmapsto L, H, K, P, N.$$

No accepted read may replace this route by a free stream window, detached dyadic tolerance, selected filter limit, quotient-filter equality, countable choice, host equality, or ambient completeness.

Proof. Project the source row C and the window rows W from Definition 1569.1. The filterbase dependency is internal to the displayed $\mathsf{CauchyFilter}^{\uparrow}$ row, while W is the only window family exposed by the completion packet.

Apply Definition 1569.8. Every accepted completion read consumes the filterbase route through C before any stream window, tolerance ledger, readback row, or Real seal is available.

Apply Definition 1569.2. Tolerance exactness ties each accepted readback in R to the displayed dyadic ledger T for the same window row W, and the rows H, K, P, N preserve, replay, source, and name this exact route.

The nonescape theorem Definition 1569.16 excludes the remaining alternatives. Detached windows, free tolerances, selected limits, quotient filters, countable choice, host equality, and ambient completeness are absent from the carrier and cannot replace the displayed route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filterbase window exactness`

Theorem 1569.21 (CauchyFilterCompletion filterbase totality)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every accepted completion read consumes the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *route behind C before any consumer can inspect the window family W, the dyadic tolerance ledger T, the regular readback row R, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal L. Hence the route*

$$C_{\mathsf{FilterBase}^{\uparrow}}, C_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto W, T, R, L \longmapsto H, K, P, N$$

is total for accepted completion reads inside the packet.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filterbase totality`

Proof. Start with the carrier of Definition 1569.1. The only filter-facing coordinate is C, and the packet exposes W, T, R, L only as rows downstream of that source.

Apply Definition 1569.8. It states that every accepted completion read first consumes the displayed filterbase and Cauchy-filter route before any stream window, dyadic tolerance, regular readback, or Real seal is available.

The remaining structural rows do not add another source. Transport through H, replay through K, provenance through P, and local naming through N preserve the same order, so the filterbase route behind C is total for all accepted completion reads. □

Theorem 1569.22 (CauchyFilterCompletion window tolerance exactness)**.** *For every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier $F = (C, W, T, R, L, H, K, P, N)$, each accepted window row in W uses its matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row in T before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row in R can be consumed. The resulting readback route is*

$$W, T \longmapsto R \longmapsto L, H, K, P, N.$$

No accepted read may replace the displayed tolerance with a free dyadic row, a detached stream window, a selected limit, a quotient-filter equality, host equality, countable choice, or ambient completeness.

Proof. Project the window, tolerance, and readback coordinates of Definition 1569.1. The packet has one displayed window family W, one dyadic tolerance ledger T, and one regular readback row R.

Use Definition 1569.2. Its tolerance-exactness row ties every accepted readback in R to the displayed dyadic ledger T for the same window row W.

Now apply Definition 1569.20. The same theorem keeps the window read behind the accepted filterbase route and prevents detached windows, free tolerances, selected limits, quotient-filter equality, countable choice, host equality, and ambient completeness from replacing the displayed route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket window tolerance exactness`

Theorem 1569.23 (CauchyFilterCompletion Real seal refusal)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The terminal* $\mathsf{Real}^{\uparrow}$ *seal row L exports only the finite route already displayed by the packet:*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

It cannot export quotient filters, selected limits, host equality of filters or real names, countable choice, ambient completeness, or any completion object detached from F.

Proof. Begin with Definition 1569.3. A consumer reaches L only after the Cauchy-filter source row, finite windows, dyadic tolerance ledger, and regular readback row have been consumed.

Use Definition 1569.7 and Definition 1569.19. They show that located-facing and uniform-facing reads share the same displayed route and do not introduce a second completion coordinate.

Finally apply Definition 1569.16. Quotient filters, selected limits, host equality, countable choice, ambient completeness, and detached completion objects are absent from the carrier and refused by the local NameCert surface, so the Real seal cannot export them. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal refusal`

Theorem 1569.24 (CauchyFilterCompletion root obligation surface)**.** *For every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier $F = (C, W, T, R, L, H, K, P, N)$, the root completion obligation surface is exactly the finite route through* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *displayed by the tuple. Every completion handoff must first read the Cauchy-filter source row C, then the finite windows W, the tolerance ledger T, the regular readback R, the Real seal L, and finally the structural rows H, K, P, N.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootObligationSurface`

Proof. Project the carrier of Definition 1569.1. It lists exactly $C, W, T, R, L, H, K, P, N$, and the provenance row names the $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$ dependencies.

Apply Definition 1569.2. Carrier admission, filter-to-window exposure, tolerance exactness, completion handoff, and ledger refusal are all attached to the same displayed packet.

Use Definition 1569.33. The filterbase-totality, window-tolerance exactness, and Real-seal refusal routes cover the root handoff, while H, K, P, N provide only transport, replay, provenance, and local naming. □

Theorem 1569.25 (CauchyFilterCompletion filterbase admission)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The root completion surface admits a downstream read only after the* BHist *rows behind C display the* $\mathsf{FilterBase}^{\uparrow}$ *source, the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *or* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *Cauchy condition read on that source, and the matching* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *admission row. The admitted read is then transported by H, replayed by K, sourced by P, and named by N; it cannot start from a detached filterbase, quotient completion, selected representative, host equality proof, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filterbase admission`

Proof. Project the root obligation surface of Definition 1569.24. The only source coordinate available before any window or readback is C, and its admissible content is the displayed filterbase and Cauchy-filter data.

Use Definition 1569.8. It forces every accepted completion read to consume the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ route before any $\mathsf{StreamName}^{\uparrow}$ window, dyadic tolerance, regular readback, or Real seal is inspected.

The structural rows preserve admission rather than add a second source. Componentwise hsame transport in H, displayed Cont replay in K, Pkg provenance in P, and local $\mathsf{NameCert}$ naming in N keep the read on the same finite BHist route and refuse quotient completions, selected representatives, host equality, and ambient completeness. □

Theorem 1569.26 (CauchyFilterCompletion uniform Cauchy lift)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *consumer of the root surface lifts through the same finite packet:*

$\mathsf{FilterBase}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BI

The lift first reads the admitted filterbase row in C, then the finite $\mathsf{StreamName}^{\uparrow}$ *windows W, the dyadic tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal L. No quotient completion, selected representative, or detached uniform-completion carrier is exported.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionUniformCauchyLift`

Proof. Begin with Definition 1569.25. The uniform consumer is admitted only through the displayed filterbase and Cauchy-filter source behind C.

Read the finite observation data next. The window row W and tolerance row T are the only rows from which a uniform Cauchy comparison can produce the regular readback R, and Definition 1569.22 keeps those rows tied to the same source.

Finally use Definition 1569.19 and Definition 1569.3. Uniform completion is recorded through provenance while the terminal seal remains L, and the route is carried by H, K, P, N without quotient completion, selected representatives, detached carriers, or ambient completeness. □

Theorem 1569.27 (CauchyFilterCompletion root data exhaustion)**.** *For every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier $F = (C, W, T, R, L, H, K, P, N)$, the root data consumed by downstream completion and filter chapters are exhausted by the displayed rows for* $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *Any admitted downstream consumer must factor through Definition 1569.25 and Definition 1569.26 before it reads L, H, K, P, or N.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootDataExhaustion`

Proof. Use Definition 1569.24 to identify the root surface with the tuple $C, W, T, R, L, H, K, P, N$. There is no root coordinate outside that tuple.

Apply Definition 1569.25 to account for the filterbase and Cauchy-filter source. Apply Definition 1569.26 to account for the uniform-facing readback through stream windows, tolerances, regular names, and the Real seal.

The remaining rows are structural. Transport, replay, provenance, and local naming preserve the admitted route, so downstream chapters cannot obtain quotient completion data, selected representatives, host equality, detached uniform-completion carriers, or ambient completeness from the root surface. □

Theorem 1569.28 (CauchyFilterCompletion root filterbase admission)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The root obligation surface admits a completion request only when the finite* BHist *rows behind C display both the* $\mathsf{FilterBase}^{\uparrow}$ *admission data and the saturated* $\mathsf{CauchyFilter}^{\uparrow}$ *source before any* $\mathsf{StreamName}^{\uparrow}$ *window in W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R, or terminal* $\mathsf{Real}^{\uparrow}$ *seal in L is consumed. The admission route is transported only by the componentwise* hsame *rows in H, replayed only by the displayed* Cont *rows in K, and named only by the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row N together with provenance P.*

Proof. Project the carrier and the root obligation surface. The source coordinate C is the only row from which a filterbase-facing completion request can enter the packet, and the finite BHist data displayed there supplies the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ admission route.

The observation rows are downstream. A consumer reaches W, R, or L only after the source row has been admitted, and the tolerance ledger T remains attached to the same finite window route.

The structural rows do not add another admission channel. Componentwise hsame transport in H, displayed Cont replay in K, provenance in P, and local $\mathsf{NameCert}$ naming in N preserve the same finite route, so detached filterbases, selected limits, and ambient completeness have no admitted coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root filterbase admission`

Theorem 1569.29 (CauchyFilterCompletion root uniform-filter readback)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Any* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *consumer reading the root completion surface receives its regular readback only through the finite route*

$$\mathsf{FilterBase}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by C, W, R, L and the finite BHist *rows of the packet. The readback is preserved by* hsame *transport in H, by* Cont *replay in K, and by the* $\mathsf{Pkg/NameCert}$ *rows P, N; it cannot be replaced by a quotient filter, detached uniform-filter object, selected limit, host equality, countable choice, or ambient completeness.*

Proof. Start with a uniform-filter consumer. Once it cites the accepted $\mathsf{CauchyFilterCompletion}^{\uparrow}$ packet, the only available uniform-facing data are the provenance row P and the displayed route through C, W, T, R, L, H, K, N.

Read the finite source and window rows. The $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source is exposed through C, the finite $\mathsf{StreamName}^{\uparrow}$ observations are exposed through W, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback R is consumed before the terminal $\mathsf{Real}^{\uparrow}$ seal L.

Finally apply the structural rows. The componentwise hsame row H transports the same source, window, readback, and seal coordinates; the Cont row K replays only that route; and P, N source and name it. Thus the uniform-filter readback has no coordinate for any detached or choice-based completion object. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root uniform filter readback`

Theorem 1569.30 (CauchyFilterCompletion root Real-seal no-choice)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. The terminal* Real$^{\uparrow}$ *seal row L is a no-choice exit from the root obligation surface: every accepted read of L passes through finite* BHist *rows for* FilterBase$^{\uparrow}$, CauchyFilter$^{\uparrow}$, UniformCauchyFilter$^{\uparrow}$ *provenance,* StreamName$^{\uparrow}$ *windows, and* RegSeqRat$^{\uparrow}$ *readback, then through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. No selected filter limit, quotient-filter representative, host real equality, countable-choice selector, or ambient* Real$^{\uparrow}$ *completeness theorem is available after the seal is reached.*

Proof. Project the terminal seal coordinate. The row L occurs only in the accepted tuple $F = (C, W, T, R, L, H, K, P, N)$, downstream of the filter source, finite windows, dyadic tolerance ledger, and regular readback.

Use the filterbase admission and uniform-filter readback routes. They place the FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ source, UniformCauchyFilter$^{\uparrow}$ provenance, StreamName$^{\uparrow}$ windows, and RegSeqRat$^{\uparrow}$ readback before any Real-seal consumer may inspect L.

The no-choice conclusion follows from carrier exhaustion. The rows H, K, P, N only transport, replay, source, and name the finite BHist route, so no selected limit, quotient representative, host equality, countable-choice functional, or ambient completeness theorem can be read after the seal. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root real seal no choice`

Theorem 1569.31 (CauchyFilterCompletion regular readback totality)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every accepted completion read that reaches the terminal* Real$^{\uparrow}$ *seal L must consume a* RegSeqRat$^{\uparrow}$ *readback row R produced from the same* StreamName$^{\uparrow}$ *windows W and* DyadicRatCore$^{\uparrow}$ *tolerance ledger T behind the* CauchyFilter$^{\uparrow}$ *source C. The readback route is total inside the tuple and cannot be replaced by a detached stream, free dyadic tolerance, quotient filter, selected limit, host equality, countable choice, or ambient completeness row.*

Proof. Start with Definition 1569.22. It forces each accepted window read in W to use the matching dyadic tolerance row in T before R can be consumed.

Apply Definition 1569.21. The same read is already behind the displayed FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ route contained in C, so R is not a detached regular sequence row.

Finally use Definition 1569.23. A consumer reaches L only after C, W, T, R have been displayed, and the structural rows H, K, P, N preserve that route without adding quotient, selector, choice, equality, or completeness coordinates. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket regular readback totality`

Theorem 1569.32 (CauchyFilterCompletion uniform reflection nonescape)**.** *For every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, any* $\mathsf{UniformCompletion}^{\uparrow}$ *reflection of the completion handoff is confined to the provenance-level row* P *and the displayed route*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

The reflection cannot introduce a separate uniform-completion carrier, quotient filter, selected limit, host equality, countable-choice selector, ambient completeness theorem, or completion object detached from the BHist/hsame/Cont/Pkg/NameCert *route.*

Proof. Begin with Definition 1569.19. It places the $\mathsf{UniformCompletion}^{\uparrow}$ occurrence in the provenance row and keeps the completion-facing consumer on the route through $C, W, T, R, L, H, K, P, N$.

Use Definition 1569.7. Located-facing and uniform-facing reads share the same Real-seal exit, so the uniform reflection cannot split away from the displayed Cauchy-filter, window, tolerance, and readback rows.

Apply Definition 1569.16. The excluded quotient, selector, host-equality, choice, completeness, and detached-completion objects have no coordinate in F, and uniform reflection through P does not create one. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform reflection nonescape`

Theorem 1569.33 (CauchyFilterCompletion carrier-ledger package)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The rows* $C, W, T, R, L, H, K, P, N$ *form a single carrier-admission and ledger-refusal package for the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$*: every accepted completion read is exhausted by the displayed filterbase-totality route, the window-tolerance exactness route, and the Real-seal refusal route, and no other row is consumed by the certificate.*

In particular, the local NameCert admits only the displayed Cauchy-filter source, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal handoff, transport, continuation replay, provenance, and local naming rows. It refuses quotient filters, selected limits, host real equality, countable-choice selectors, ambient completeness, and detached completion objects over the same finite packet.

Proof. Project Definition 1569.2. It identifies the local certificate surface with carrier admission, filter-to-window exposure, tolerance exactness, completion handoff, and ledger refusal over exactly the rows $C, W, T, R, L, H, K, P, N$.

Apply Definition 1569.21. This accounts for every accepted completion read at the source side: the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ route behind C is consumed before W, T, R, L and the structural rows can be inspected.

Apply Definition 1569.22 and Definition 1569.23. The first theorem ties each window read to the matching dyadic tolerance and regular readback, while the second keeps the terminal Real seal on the same finite route and refuses quotient, selector, host-equality, choice, completeness, and detached-completion exits.

These three routes cover the carrier side, the tolerance ledger side, and the terminal refusal side of the same packet. Since the carrier definition and NameCert obligations expose no additional coordinate, the local certificate consumes no row outside $C, W, T, R, L, H, K, P, N$. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket carrier ledger package`

Theorem 1569.34 (CauchyFilterCompletion Real seal choice-free tail totality)**.** *Let $F = (C, W, T, R, L, H, K, P, N$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every completion read that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal L has a displayed tail route*

$$C_{\mathsf{FilterBase}^{\uparrow}}, C_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto W, T, R \longmapsto L \longmapsto H, K, P, N.$$

The route is total for accepted reads and uses no quotient filter, selected limit, countable-choice selector, host real equality, ambient completeness theorem, or detached completion carrier.

Proof. Begin with the filterbase handoff of Definition 1569.8. It forces the accepted completion read to consume the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source behind C before any window, tolerance, readback, or seal row is available.

Next apply Definition 1569.22. The window row W and dyadic ledger T determine the regular readback R before the terminal seal can be inspected.

Finally use Definition 1569.3 and Definition 1569.16. The seal row L is downstream of C, W, T, R, and the refused quotient, selector, choice, equality, completeness, and detached-completion objects have no coordinate inside F. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal choice free tail totality`

Theorem 1569.35 (CauchyFilterCompletion window Real seal determinacy)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. If two completion-facing consumers read the same displayed filter source C, the same window family W, and the same dyadic tolerance ledger T, then any accepted terminal* $\mathsf{Real}^{\uparrow}$ *seal read from F is determined by the same regular readback row R and the same structural route through L, H, K, P, N.*

Thus no consumer may replace the displayed window route by a second Real seal, a selected limit, quotient-filter equality, countable choice, host equality, ambient completeness, or detached uniform-completion object.

Proof. Project the accepted packet. The rows $C, W, T, R, L, H, K, P, N$ are the only coordinates available to either consumer.

Use Definition 1569.20. Matching C, W, T fixes the filterbase-to-window route and ties every accepted readback in R to the displayed dyadic ledger.

Apply Definition 1569.23. Once R is fixed, the terminal seal row L is consumed only through the same finite route and cannot be replaced by a selected limit, quotient equality, choice, host equality, ambient completeness, or detached completion row.

The structural rows preserve this determinacy. Transport H, replay K, provenance P, and local naming N record the same route for both consumers. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket window real seal determinacy`

Theorem 1569.36 (CauchyFilterCompletion uniform provenance exhaustion)**.** *Let $F = (C, W, T, R, L, H, K, P, N$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The* $\mathsf{UniformCompletion}^{\uparrow}$ *occurrence in the packet is exhausted by the provenance row P and the displayed route*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

It contributes no additional uniform-completion carrier, quotient filter, selected limit, countable-choice selector, host equality, ambient completeness theorem, or Real seal outside the rows of F.

Proof. Start with Definition 1569.19. It places the uniform-completion surface at provenance level and keeps the completion-facing read on the finite route through $C, W, T, R, L, H, K, P, N$.

Apply Definition 1569.32. Any uniform reflection is confined to P and cannot split off a second completion object.

Use Definition 1569.33. The carrier admission and ledger refusal rows already exhaust the accepted NameCert surface, while the quotient nonescape theorem rules out the remaining forbidden exits.

Therefore uniform provenance is not an extra source of completion data. It is exactly the package row naming the same Real-seal tail route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket uniform provenance exhaustion`

Theorem 1569.37 (CauchyFilterCompletion Cauchy-filter limit uniform lock)**.** *Let $F = (C, W, T, R, L, H, K, P, N$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every completion read through F factors through the* CauchyFilter$^{\uparrow}$ *source row C, the* CauchyFilterLimit$^{\uparrow}$ *boundary named by the same filter row, finite* StreamName$^{\uparrow}$ *windows W,* DyadicRatCore$^{\uparrow}$ *tolerances T,* RegSeqRat$^{\uparrow}$ *readback R, and* UniformCompletion$^{\uparrow}$ *provenance P before any terminal* Real$^{\uparrow}$ *seal L is consumed. The remaining rows H, K, N only transport, replay, and name that same route.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket cauchyfilter limit uniform lock`

Proof. Start with Definition 1569.8. It forces the accepted completion read to expose the filter source before any window, tolerance, readback, or terminal seal row is available.

Use Definition 1569.22. The finite StreamName windows W and dyadic tolerance ledger T determine the regular readback row R inside the displayed packet.

Apply Definition 1569.3 and Definition 1569.36. The Real seal L and the uniform provenance row P remain downstream of the same finite route, while Definition 1569.16 excludes quotient filters, selected limits, choice, host equality, ambient completeness, and detached completion data.

The structural rows preserve the lock. Componentwise hsame transport, displayed Cont replay, and local naming can only repeat the route already exposed by C, W, T, R, L, P. □

Theorem 1569.38 (CauchyFilterCompletion completion-reflector consumer)**.** *Every* MetricCompletion$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer of an accepted* CauchyFilterCompletion$^{\uparrow}$ *packet receives only the finite route*

$$C \longmapsto W \longmapsto T \longmapsto R \longmapsto L \longmapsto H, K, P, N.$$

The consumer may reflect that route into a completion-facing packet, but it cannot import quotient filters, selected limits, countable choice, host equality, ambient completeness, or any completion carrier not already displayed by the rows of F.

Proof. Read the accepted packet through Definition 1569.2. Its carrier and NameCert rows expose only the Cauchy-filter source, finite windows, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming.

Then apply Definition 1569.37. A completion-facing consumer reaches the CauchyFilterLimit boundary, UniformCompletion provenance, and Real seal only by following that displayed order.

Finally use Definition 1569.39. The basis exhaustion theorem rules out any second completion carrier or host-side principle, so MetricCompletion and UniformCompletion consumers receive only the listed finite route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket completion reflector consumer`

Theorem 1569.39 (CauchyFilterCompletion Real seal basis exhaustion)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every accepted completion read through F is exhausted by the displayed* CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *source row, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *seal handoff, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. No quotient filter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or detached completion object is available after those rows are read.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal basis exhaustion`

Proof. Project the NameCert obligations of Definition 1569.2. They identify the accepted packet with carrier admission, filter-to-window exposure, tolerance exactness, completion handoff, transport, replay, provenance, and local naming over $C, W, T, R, L, H, K, P, N$.

Use Definition 1569.21 and Definition 1569.22. The first theorem exhausts the displayed filter-base source behind C, while the second ties each finite window in W to its dyadic tolerance ledger T and regular readback row R.

Apply Definition 1569.23 and Definition 1569.36. The terminal seal L and the uniform provenance occurrence remain downstream of the same finite route and cannot split off a quotient, selector, host equality, choice, ambient-completeness, or detached-completion exit.

Finally preserve the route through H, K, P, N. These rows transport, replay, source, and name only the already displayed basis data, so every accepted completion read is exhausted by the listed basis. □

Theorem 1569.40 (CauchyFilterCompletion observation base)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every accepted completion read starts from the displayed* CauchyFilter$^{\uparrow}$ *source row C before it may inspect any* StreamName$^{\uparrow}$ *window in W, any* DyadicRatCore$^{\uparrow}$ *tolerance in T, any* RegSeqRat$^{\uparrow}$ *readback in R, or any terminal* Real$^{\uparrow}$ *seal in L. The structural rows H, K, P, N preserve that source order rather than supplying a second observation base.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket observation base`

Proof. Begin with Definition 1569.8. It places every accepted completion read behind the displayed FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ route contained in C before any finite window or tolerance row is available.

Use Definition 1569.22. Once the source row has been exposed, the stream windows W and dyadic tolerances T determine the regular readback row R inside the same carrier.

Apply Definition 1569.39. The terminal seal and the structural rows remain downstream of the same basis, so no completion read can begin from a detached window, readback, host equality, selected limit, quotient filter, choice principle, or ambient completeness theorem. □

Theorem 1569.41 (CauchyFilterCompletion located-uniform basis)**.** *Located-filter and uniform-filter consumers of an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier share exactly the same basis rows $C, W, T, R, L, H, K, P, N$. A located read and a uniform read may emphasize different consumer boundaries, but both must pass through the displayed Cauchy-filter source, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket located uniform basis`

Proof. Start from Definition 1569.7. It states that located-facing and uniform-facing reads are glued along the same finite completion route rather than by a second completion carrier.

Next apply Definition 1569.19. Uniform provenance is recorded through P while the read remains on C, W, T, R, L, H, K, N; it does not create a detached uniform source.

Finally use Definition 1569.39. The shared basis exhausts both consumers, and quotient filters, selected limits, host equality, choice, ambient completeness, and detached completion data are absent from every row. □

Theorem 1569.42 (CauchyFilterCompletion completion scope)**.** *The* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet closes only the finite completion route displayed by* $C, W, T, R, L, H, K, P, N$*. It supplies a named route from Cauchy-filter observations through finite stream windows, dyadic tolerances, regular rational readback, and a Real seal, but it refuses quotient filters, selected limits, host equality, countable choice, ambient completeness, and any completion object not displayed by the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket completion scope`

Proof. Project the local NameCert surface with Definition 1569.2. The carrier rows account for source admission, window exposure, tolerance exactness, completion handoff, transport, replay, provenance, and local naming.

Use the observation base theorem Definition 1569.40 and the located-uniform basis theorem Definition 1569.41. Together they show that all completion-facing consumers start from the same source row and share the same displayed basis through located and uniform reads.

Close the scope with Definition 1569.16. The refused quotient, selector, host-equality, choice, ambient-completeness, and detached-completion objects have no carrier coordinate, so the packet closes exactly the finite route it displays. □

Theorem 1569.43 (CauchyFilterCompletion choice-free completion row)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every accepted completion read is choice-free and factors through the displayed rows*

$$C, W, T, R, L, H, K, P, N.$$

In particular, the read starts from the $\mathsf{CauchyFilter}^{\uparrow}$ *source row* C*, uses only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L*, and is then transported, replayed, sourced, and named by* H, K, P, N*.*

Proof. Project the carrier of Definition 1569.1. Its displayed rows are exactly $C, W, T, R, L, H, K, P, N$, and no coordinate stores a countable-choice selector, selected limit, quotient filter, host equality, or ambient completeness theorem.

Apply Definition 1569.8. The read must first consume the filterbase and Cauchy-filter route behind C before any window, tolerance, readback, or Real seal row is available.

Apply Definition 1569.22 and Definition 1569.3. These force the finite order W, T, R, L after the source row and before the structural rows.

Finally use Definition 1569.16. Since every excluded object is absent from the carrier and refused by the ledger, every accepted completion read is exhausted by the displayed choice-free route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket choicefree completion row`

Theorem 1569.44 (CauchyFilterCompletion regular readback completeness)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. For every accepted finite stream-window and dyadic-tolerance read through W and T, the* RegSeqRat$^{\uparrow}$ *readback row R is total inside the packet: any completion-facing consumer that reaches the accepted window and tolerance rows must receive its regular rational readback through R before it may inspect L, H, K, P, or N.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRegularReadbackCompleteness`

Proof. Start with the carrier definition Definition 1569.1. The only readback coordinate displayed after the stream-window and dyadic-tolerance rows is R.

Use Definition 1569.4. It states that every accepted completion read opens W and T before it can read R, so R is tied to the same finite stream-window and tolerance data.

Use Definition 1569.22. The matching dyadic tolerance row in T is required before the regular readback row can be consumed, ruling out detached windows and free tolerance parameters.

The downstream rows cannot replace the readback. By Definition 1569.18, any consumer that reaches L has already passed through R, and H, K, P, N only transport, replay, source, and name the same route. □

Theorem 1569.45 (CauchyFilterCompletion metric-uniform compatibility)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Any* MetricCompletion$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer of F reads the same finite completion route*

$$C, W, T, R, L, H, K, P, N.$$

Neither consumer may replace this route by quotient filters, selected limits, host equality, ambient completeness, countable choice, or a detached metric-completion or uniform-completion object.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket metric uniform compatibility`

Proof. Begin with the shared source route from Definition 1569.1. The packet exposes one Cauchy-filter source, one window family, one tolerance ledger, one regular readback row, and one Real seal handoff.

For metric-facing reads, apply Definition 1569.42. It confines all completion-facing consumers to the rows $C, W, T, R, L, H, K, P, N$ and refuses detached completion objects.

For uniform-facing reads, apply Definition 1569.19 and Definition 1569.10. The UniformCompletion$^{\uparrow}$ occurrence is provenance-level and shares the same route through the terminal seal and structural rows.

Finally use Definition 1569.16. Quotient filters, selected limits, host equality, ambient completeness, countable choice, and detached metric or uniform completion carriers have no coordinate in F, so both consumers must read the same finite route. □

Theorem 1569.46 (CauchyFilterCompletion filter-base admission)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every completion-facing read admits the displayed* CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *source behind C before any consumer can inspect W, T, R, or L. The structural rows H, K, P, N preserve this source order and introduce no detached basis, selected limit, quotient filter, host equality, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 1569.1. The row C is the only Cauchy-filter source coordinate, and the packet exposes W, T, R, L only as downstream rows.

Apply Definition 1569.8 and Definition 1569.21. They place the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ admission route before every stream-window, tolerance, readback, and seal read.

Finally use Definition 1569.33. The local NameCert consumes only the displayed rows $C, W, T, R, L, H, K, P, N$, so the refused quotient, selector, host-equality, choice, and completeness objects have no admission coordinate. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filter base source order`

Theorem 1569.47 (CauchyFilterCompletion completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier shares the same completion handoff route*

$$C, W, T, R, L, H, K, P, N.$$

Metric-facing and uniform-facing reads may use different downstream vocabulary, but both receive the displayed Cauchy-filter source, stream windows, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket completion handoff`

Proof. Begin with Definition 1569.46. It fixes the source side of any completion-facing read at the displayed $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ row behind C.

Use Definition 1569.45. It identifies the $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumer routes with the same finite rows $C, W, T, R, L, H, K, P, N$.

The handoff is then exhausted by Definition 1569.39. No detached metric-completion object, detached uniform-completion object, quotient filter, selected limit, countable choice, host equality, or ambient completeness row remains outside the displayed route. □

Theorem 1569.48 (CauchyFilterCompletion filterbase window minimality)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. A completion-facing read may open a* $\mathsf{StreamName}^{\uparrow}$ *window in* W *only after the* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source behind* C *has been admitted, and the only rows available after that opening are the matching dyadic tolerance row* T*, regular readback row* R*, Real seal* L*, and structural rows* H, K, P, N*. The carrier therefore contains no quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness row between the source and the window.*

Proof. Project the tuple F. The source coordinate is C, while W, T, R, L are downstream observation rows.

Apply Definition 1569.46. It places every completion-facing read behind the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source before any stream window is consumed.

Use Definition 1569.82 to exhaust the remaining coordinates. Since the refused quotient, selector, choice, equality, and completeness objects are not fields of F, the displayed source-to-window route is minimal. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket filterbase window minimality`

Theorem 1569.49 (CauchyFilterCompletion tail readback admission)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every tail readback admitted by the root completion surface passes through the finite route*

$$C \longmapsto W \longmapsto T \longmapsto R \longmapsto L$$

together with H, K, P, N, where C carries the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source, W carries the* $\mathsf{StreamName}^{\uparrow}$ *window, T carries the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, R carries the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and L carries the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with Definition 1569.31. It forces each accepted readback in R to use the matching stream window W and dyadic tolerance ledger T behind the Cauchy-filter source C.

Then apply Definition 1569.30. The terminal Real seal L is read only after that finite source-window-tolerance-readback route.

The structural rows H, K, P, N preserve the same route. Thus no detached tail, quotient filter, selected limit, host equality, countable choice, or ambient completeness row is admitted. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket tail readback admission`

Theorem 1569.50 (CauchyFilterCompletion Real-seal source exhaustion)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Once a consumer reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal L, every source row available to that consumer has already appeared in C, W, T, R and the structural rows H, K, P, N. The seal exports only the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *route, not a quotient filter, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket real seal source exhaustion`

Proof. Read L through Definition 1569.49. The theorem places the Real seal after the source, window, tolerance, and regular-readback rows.

Apply Definition 1569.33. The local certificate consumes no row outside $C, W, T, R, L, H, K, P, N$.

Finally use Definition 1569.82. The refused objects have no coordinate in the accepted carrier, so the Real seal exhausts the displayed finite source route. □

Theorem 1569.51 (CauchyFilterCompletion root exactness)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The accepted root reads of F are exhausted by the* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source row C,* $\mathsf{StreamName}^{\uparrow}$ *windows W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal L,* $\mathsf{UniformCompletion}^{\uparrow}$ *provenance carried by P, componentwise* hsame *transport H, displayed* Cont *replay K,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *rows N. No root read adds a quotient filter, selected limit, detached completion carrier, host equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root exactness`

Proof. Project the carrier of Definition 1569.1. Its root completion route is the displayed tuple $C, W, T, R, L, H, K, P, N$.

Apply Definition 1569.48 and Definition 1569.49. These place every source and tail read behind the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Use Definition 1569.50 and Definition 1569.60. The Real seal and uniform-completion consumers inherit the same finite route and cannot introduce detached completion data.

Finally the structural rows H, K, P, N transport, replay, source, and name exactly those reads. Since the refused quotient, selected-limit, host-equality, choice, and ambient-completeness objects have no coordinate in the carrier, the listed rows exhaust the root completion surface. □

Theorem 1569.52 (CauchyFilterCompletion completion-functor factorization)**.** *Let $F = (C, W, T, R, L, H, K, P, N$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{MetricCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer factors through the same source-window-tolerance-readback-seal route already displayed by F and by Definition 1569.47. The consumer receives only the Cauchy-filter source, stream-window family, dyadic tolerance ledger, regular readback, terminal seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert rows; no quotient filter, selected limit, host equality, detached completion object, or ambient completeness row is exported.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionCompletionFunctorFactorization`

Proof. Start with Definition 1569.47. It identifies every metric-facing or uniform-facing completion read with the route $C, W, T, R, L, H, K, P, N$.

Use Definition 1569.27 to keep the source, window, tolerance, readback, and seal rows inside the accepted carrier. The functor-facing vocabulary can ask for completion transport, but it has no additional source row beyond C and no seal row beyond L.

Apply Definition 1569.45. This places the metric and uniform completion consumers on the same finite route, so a $\mathsf{MetricCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, or $\mathsf{UniformCompletion}^{\uparrow}$ read can only repackage the displayed carrier rows.

Finally the visible BHist, hsame, Cont, Pkg, and NameCert rows provide transport, replay, provenance, and local naming for that same route. Since quotient filters, selected limits, host equality, detached completion objects, and ambient completeness have no coordinate in F, no completion-functor consumer can export them. □

Theorem 1569.53 (CauchyFilterCompletion cofinal regular readback)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier whose filterbase row admits the completion-functor factorization of Definition 1569.52. Then every completion-facing consumer has a cofinal finite readback route through the* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row is consumed. The route is exactly the displayed carrier order $C, W, T, R, L, H, K, P, N$, with C exposing the Cauchy-filter and filterbase source, P recording the uniform-completion provenance, R supplying the regular readback, and L supplying the terminal Real seal. No quotient filter, selected limit, host equality, or choice over a cofinal family is introduced.*

Proof. Project the carrier rows of Definition 1569.1. The only source coordinate is C, the finite observation rows are W, T, R, the terminal seal is L, and the remaining rows provide transport, replay, provenance, and local naming.

Use filter-base admission and uniform reflection. Theorems 1569.46 and 1569.19 place the $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ source before all window reads and keep the $\mathsf{UniformCompletion}^{\uparrow}$ occurrence in the same provenance-level route.

Next apply regular readback completeness, metric-uniform compatibility, and completion handoff. Theorems 1569.44, 1569.45, and 1569.47 force every completion-facing read to pass from C, W, T through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback row R before reaching the $\mathsf{Real}^{\uparrow}$ seal L.

Finally apply the completion-functor factorization hypothesis. The factorization can repackage the finite route for functor-facing consumers, but it has no additional coordinate beyond $C, W, T, R, L, H, K, P, N$. The local NameCert therefore consumes only that cofinal finite readback route, and the refused quotient filter, selected limit, host equality, and choice-over-family objects never enter the packet. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket cofinal regular readback`

Definition 1569.54 (CauchyFilterCompletion root unblock carrier)**.** *The root-unblock carrier for* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *is the same finite packet*

$$F = (C, W, T, R, L, H, K, P, N)$$

read with the following obligation order: the $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ *source row* C*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L*, componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport* H*, displayed* Cont *replay* K*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row* N*. A root consumer receives no quotient filter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or detached* $\mathsf{MetricCompletion}^{\uparrow}/\mathsf{UniformCompletion}^{\uparrow}$ *carrier.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootUnblockCarrier semantic name certificate`

The universal completion read treats the Cauchy-equivalence classifier of chapter 1820 as a displayed sibling dependency. Before any $\mathsf{Real}^{\uparrow}$ seal or filter-completion consumer reads equality of completed names, the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ comparison boundary has already been classified by that sibling route.

Theorem 1569.55 (CauchyFilterCompletion root window exposure)**.** *Every completion-facing read of a root-unblock carrier* $F = (C, W, T, R, L, H, K, P, N)$ *opens the displayed* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ *source row* C *and the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W *before it consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T *or the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*.*

Proof. Project the root-unblock carrier of Definition 1569.54. Its first completion-facing coordinates are the source row C, window row W, tolerance row T, and readback row R.

Apply Definition 1569.40. It forces every accepted completion read to start from the displayed source row before any window, tolerance, readback, or seal row is available.

Use Definition 1569.22. Once C and W are open, the matching tolerance row T is required before R can be consumed, so the read cannot start from a detached regular sequence. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootWindowExposure`

Theorem 1569.56 (CauchyFilterCompletion root Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of a root-unblock carrier is downstream of the same source, window, tolerance, and readback rows:*

$$C, W, T, R \longmapsto L \longmapsto H, K, P, N.$$

The seal route exports no quotient filter, selected limit, host equality, countable choice, ambient completeness, or detached completion object.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootRealSealRoute`

Proof. Start with Definition 1569.55. It exposes C, W, T, R before the terminal seal can be inspected by a completion-facing consumer.

Apply Definition 1569.23. The $\mathsf{Real}^{\uparrow}$ seal L exports only the finite route already displayed by the packet and refuses quotient, selector, host-equality, choice, completeness, and detached-completion exits.

Finally preserve the route through H, K, P, N. These rows transport, replay, source, and name the same packet, so none of them can replace the source-window-readback chain. □

Theorem 1569.57 (CauchyFilterCompletion root uniform route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of a root-unblock carrier receives only the displayed route*

$$C, W, T, R, L, H, K, P, N.$$

Uniform and metric completion reads may inspect the provenance row, but they cannot introduce a second completion carrier, quotient filter, selected limit, host equality, countable choice, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootUniformRoute`

Proof. Project the carrier of Definition 1569.54. The only completion-facing rows available to a consumer are $C, W, T, R, L, H, K, P, N$.

Use Definition 1569.45. It states that $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers read the same finite completion route and cannot replace it by detached completion data.

Apply Definition 1569.56. The terminal Real seal is already downstream of the source, window, tolerance, and readback rows, so uniform and metric consumers inherit the same displayed boundary. □

Theorem 1569.58 (CauchyFilterCompletion root NameCert surface)**.** *The root obligation surface of* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *is exactly the root-unblock carrier of Definition 1569.54 together with the window exposure of Definition 1569.55, the Real seal route of Definition 1569.56, and the metric-uniform route of Definition 1569.57. A root consumer must therefore read the displayed* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{FilterBase}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows before exporting any completion-facing result.*

Proof. Combine the three root routes. The window theorem exposes the source and finite observation rows, the Real seal theorem fixes the terminal readback route, and the metric-uniform theorem keeps completion consumers on the same displayed carrier.

Now apply Definition 1569.33. It identifies the local NameCert package with the carrier rows and ledger refusals over $C, W, T, R, L, H, K, P, N$.

Finally use Definition 1569.42. The chapter closes only this finite completion route and refuses quotient filters, selected limits, host real equality, countable-choice selectors, ambient completeness, and detached completion objects. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootSurface semantic name certificate`

Theorem 1569.59 (CauchyFilterCompletion root Cauchy-filter limit existence)**.** *Every completion-facing consumer of a root-unblock carrier that claims a Cauchy-filter limit must display the finite route*

$$C, W, T, R, L, H, K, P, N.$$

The limit obligation is therefore carried by the CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *source row C, the* StreamName$^{\uparrow}$ *windows W, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger T, the* RegSeqRat$^{\uparrow}$ *readback R, the terminal* Real$^{\uparrow}$ *seal L, and the structural* BHist/hsame, Cont, Pkg, *and* NameCert *rows. It is not a selected filter limit, quotient representative, host real equality proof, countable-choice selector, or ambient completeness theorem.*

Proof. Start from the root-unblock carrier. The displayed tuple contains only the rows $C, W, T, R, L, H, K, P, N$, so any asserted Cauchy-filter limit must be read through those coordinates.

Apply the root window exposure theorem to force the source and finite windows before tolerance and readback. Apply the root Real seal route to put the terminal seal downstream of that same source-window-readback path.

The NameCert surface then packages the route as a finite obligation. Since the packet has no coordinate for quotient representatives or selected limits, the existence readback is the displayed completion route itself, not an external completeness principle. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root cauchy filter limit existence`

Theorem 1569.60 (CauchyFilterCompletion uniform-completion universal property)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer of a root-unblock carrier factors its universal-completion obligation through the same finite rows*

$$C, W, T, R, L, H, K, P, N.$$

The universal property can use the CauchyFilter$^{\uparrow}$ *source,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal,* BHist/hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, but it cannot use quotient filter representatives, selected limits, detached uniform-completion carriers, host equality, countable choice, or ambient completeness.*

Proof. Project the root-unblock carrier. The only uniform-facing coordinate beyond the source, windows, tolerance, readback, and Real seal is the provenance row P.

Apply the root uniform route. It identifies every UniformCompletion$^{\uparrow}$ or MetricCompletion$^{\uparrow}$ consumer with the displayed packet route and blocks detached completion data.

The universal property is therefore stated over the same finite route. Transport through H, replay through K, provenance through P, and local naming through N preserve the packet; none of those rows creates quotient representatives, selected limits, host equality, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionUniformCompletionUniversalProperty`

Theorem 1569.61 (CauchyFilterCompletion Real seal uniqueness interface)**.** *Any two root-unblock consumers that read the same displayed* CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *row C,* StreamName$^{\uparrow}$ *windows W,* DyadicRatCore$^{\uparrow}$ *tolerance ledger T, and* RegSeqRat$^{\uparrow}$ *readback R reach the same* Real$^{\uparrow}$ *seal interface L inside the accepted packet. The interface is transported only by H, replayed only by K, sourced only by P, and named only by N.*

Proof. Use the root window exposure theorem for both consumers. Once C, W, T, R are fixed, neither consumer has another source, window, tolerance, or readback coordinate available inside the carrier.

Apply the root Real seal route. The only terminal Real-facing row downstream of C, W, T, R is L.

Finally apply the root NameCert surface. The structural rows H, K, P, N preserve, replay, source, and name the same interface, so the comparison cannot escape to quotient filter equality, selected limits, host equality, countable choice, ambient completeness, or detached completion objects. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRealSealUniquenessInterface`

Theorem 1569.62 (CauchyFilterCompletion filterbase-window exhaustion)**.** *Every completion-facing read from a root-unblock carrier must first open the* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ *source row* C *and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W *before tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, uniform or metric completion consumers, or local* NameCert *rows are available. No detached window, selected limit, quotient filter, host equality, choice route, or ambient completeness route exists inside the root carrier.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionFilterbaseWindowExhaustion`

Proof. Start with Definition 1569.55. It exposes the source row C and finite windows W before any tolerance or regular-rational readback coordinate can be consumed.

Next use Definition 1569.56 and Definition 1569.57. The Real seal, metric-completion reads, and uniform-completion reads are all downstream of the same C, W, T, R, L route.

Finally apply Definition 1569.58 and Definition 1569.61. The local NameCert surface and uniqueness interface preserve the displayed packet through H, K, P, N, so no consumer can replace the finite filterbase-window route by a detached window, selected limit, quotient filter, host equality, choice route, or ambient completeness route. □

Theorem 1569.63 (CauchyFilterCompletion public export package)**.** *The public* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *export is the package consisting of the root-unblock carrier, source-window obligations, uniform route, Real-seal uniqueness interface, filterbase-window exhaustion, and completion-functor factorization. Every public consumer receives only*

$$C, W, T, R, L, H, K, P, N.$$

It can read the $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ *source, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming rows, but it receives no quotient filter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or detached completion carrier.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPublicExportPackage`

Proof. The root-unblock carrier fixes the public tuple $C, W, T, R, L, H, K, P, N$. The source-window exposure theorem, Real seal route, uniform route, root NameCert surface, Cauchy-filter limit theorem, uniform-completion universal property, Real-seal uniqueness interface, and filterbase-window exhaustion theorem all read subroutes of that same tuple.

The completion-functor factorization is therefore public only through these displayed rows. Transport and replay preserve the tuple, provenance records the same source, and the local NameCert row names exactly this package.

Since the carrier and all root theorems refuse quotient filters, selected limits, host real equality, countable-choice selectors, ambient completeness, and detached completion carriers, no public consumer receives more than the listed rows. □

Theorem 1569.64 (CauchyFilterCompletion bridge-facing interface)**.** *Any future bridge consumer of the public* CauchyFilterCompletion$^{\uparrow}$ *package must factor through the displayed rows*

$$C, W, T, R, L, H, K, P, N.$$

The bridge-facing read preserves the CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *source row* C*, finite* StreamName$^{\uparrow}$ *windows* W*,* DyadicRatCore$^{\uparrow}$ *tolerance ledger* T*,* RegSeqRat$^{\uparrow}$ *readback* R*, terminal* Real$^{\uparrow}$ *seal* L*, componentwise* hsame *transport* H*, displayed* Cont *replay* K*,* Pkg *provenance* P*, and local* NameCert$_{\text{CauchyFilterComplet}}$ *row* N*. It has no interface coordinate for quotient filters, selected limits, host equality, countable choice, ambient* Real$^{\uparrow}$ *completeness, or detached completion carriers.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionBridgeFacingInterface`

Proof. Begin with Definition 1569.63. It identifies the public export as exactly the rows $C, W, T, R, L, H, K, P, N$, so a bridge consumer can only receive that package.

Use Definition 1569.58 to keep carrier admission, source-window obligations, tolerance readback, transport, replay, provenance, and local naming on the displayed root route. The NameCert surface does not expose a quotient filter or selected limit.

Finally apply Definition 1569.61. The Real-facing row is reached through the same finite windows, dyadic tolerance ledger, regular readback, and terminal seal, so host equality, countable choice, ambient completeness, and detached completion carriers have no bridge-facing export row. □

Closure status for NameCert$_{\text{CauchyFilterCompletion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchyFilterCompletion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1569.54, Definition 1569.55, Definition 1569.56, Definition 1569.57, Definition 1569.58, Definition 1569.59, Definition 1569.60, Definition 1569.61, Definition 1569.62, Definition 1569.63, and Definition 1569.89 close the bridged root obligation surface for the displayed Cauchy-filter source, finite windows, tolerance ledger, regular readback, terminal Real seal, structural transport, continuation replay, provenance, local NameCert rows, public export package, and completion-facing bridge route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyFilterCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host real equality, countable-choice selectors, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No further paper-side closure remains; verification and bridge checking are Lean-side obligations for this root completion surface.

Constructive story: A CauchyFilterCompletion$^{\uparrow}$ packet is generated from a CauchyFilter row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal handoff, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theorem 1569.65 (CauchyFilterCompletion root carrier admission obligation)**.** *Let* $F = (C, W, T, R, L, H, K, P, N$ *be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier as in Definition 1569.1. The root carrier-admission obligation is the* BHist*-anchored demand that the Cauchy-filter source* C*, finite* StreamName$^{\uparrow}$ *windows* W*, tolerance ledger* T*,* RegSeqRat$^{\uparrow}$ *readback* R*, terminal* Real$^{\uparrow}$ *seal* L*, transport row* H*, replay row* K*, provenance row* P*, and local NameCert row* N *are all admitted as rows of the same finite packet before any completion-facing consumer reads them.*

Proof. Project Definition 1569.1. Its displayed carrier rows are exactly $C, W, T, R, L, H, K, P, N$.

The obligation is rooted in the same BHist packet. Carrier admission therefore does not create a detached completion object; it only exposes the already displayed CauchyFilterUp, StreamNameUp, RegSeqRatUp, RealUp, UniformCompletionUp, transport, replay, provenance, and naming rows.

The refusal side follows from the absence of any quotient-filter, selected-limit, host-equality, choice, or ambient-completeness coordinate in the carrier. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootCarrierAdmissionObligation`

Theorem 1569.66 (CauchyFilterCompletion root filterbase source coverage obligation)**.** *For an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, the filterbase-directed source coverage obligation requires every completion-facing read of* W, T, R, L *to pass through the same* FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *source row behind* C*. The finite windows, regular readback, Real seal, and* UniformCompletion$^{\uparrow}$ *provenance row cannot be read from a second source packet.*

Proof. Use the carrier admission obligation of Definition 1569.65. It keeps $C, W, T, R, L, H, K, P, N$ inside one BHist packet.

Read the source row first. The CauchyFilterUp carrier behind C supplies the filterbase-facing data, and the finite StreamNameUp windows W are downstream from that source.

The readback and seal rows remain attached to the same route. Since R is the RegSeqRatUp readback extracted from W and T, and L is the RealUp seal reached after that extraction, the UniformCompletionUp provenance in P has no second filterbase source to expose. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootFilterbaseSourceCoverageObligat`

Theorem 1569.67 (CauchyFilterCompletion root Real seal handoff obligation)**.** *For an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, the Real-seal completion handoff obligation requires every* Real$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer to reach the terminal seal* L *only after the* FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *source* C*, finite* StreamName$^{\uparrow}$ *windows* W*, tolerance ledger* T*, and* RegSeqRat$^{\uparrow}$ *readback* R *have been replayed through* K *and preserved by* H, P, N*.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket root real seal handoff`

Proof. Start from Definition 1569.66. It fixes the source and window rows for every completion-facing read.

The Real seal is downstream from the readback row. A consumer of L therefore must first traverse C, W, T, R, and the displayed replay row K records that route inside the same BHist packet.

The structural rows preserve the handoff. Componentwise transport H, UniformCompletionUp provenance P, and local naming N carry the same route forward, so the seal handoff does not introduce selected limits, host equality of completion points, quotient filters, or ambient completeness. □

Theorem 1569.68 (CauchyFilterCompletion Cauchy-equivalence classifier route)**.** *Every* CauchyFilterCompletion$^{\uparrow}$ *read that compares Cauchy-filter presentations by an equivalence classifier must first consume chapter 1820 through the displayed* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback rows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger, terminal* Real$^{\uparrow}$ *seal, and the separated-completion rows of chapter 1612. The comparison is classifier evidence for the completion-facing read of* $C, W, T, R, L, H, K, P, N$*; it is not quotient equality of filters, a selected representative, host equality of completed points, or an ambient completion principle.*

Proof. Project the root filter route of Definition 1569.66. The source row C and finite window rows W, T, R, L are already fixed before any completion-facing consumer can inspect a completed point.

When the consumer also asks whether two Cauchy-filter presentations determine the same completion read, route that comparison through chapter 1820. That packet exposes only StreamName windows, RegSeqRat readbacks, dyadic tolerance rows, the Real seal, transport, replay, provenance, and local naming as classifier data.

The separated-completion handoff is then read as a consumer of the same classifier evidence. It may preserve and replay the displayed comparison, but it supplies no quotient class, no chosen representative, and no host equality coordinate.

Finally combine the classifier route with the Real-seal handoff above. The accepted CauchyFilterCompletion packet keeps the comparison inside $C, W, T, R, L, H, K, P, N$, so the completion read remains a finite classifier-mediated route rather than a quotient or selection principle. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionCauchyEquivalenceClassifierRoute`

Theorem 1569.69 (CauchyFilterCompletion root filter scope closure)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Before any completion-facing consumer can read a completed point, the root-filter completion read must expose the* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *rows behind C, the* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{SeparatedCompletion}^{\uparrow}$ *handoff surface, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{Real}^{\uparrow}$ *seal L, componentwise* hsame *transport H, displayed* Cont *replay K,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row N.*

Consequently the root-filter completion scope exports only this displayed finite route. It does not export quotient completions, selected limits, countable-choice schedules, host equality of completed points, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootFilterScopeClosure`

Proof. Project the carrier of Definition 1569.1. The source coordinate C is the only place where a root filter can enter, and the root carrier admission obligation keeps $C, W, T, R, L, H, K, P, N$ inside the same finite BHist packet.

Route filter-base requests through the existing source coverage obligation. The $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ rows behind C must be read before the $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger T, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback R can be inspected.

Now route the completion handoff. The $\mathsf{Real}^{\uparrow}$ seal L and the $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{SeparatedCompletion}^{\uparrow}$ consumer surface are reached only after the displayed source, window, tolerance, and readback rows have been replayed through K and transported by H.

Finally P and N source and name the same route. Since the packet has no quotient-completion, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, the root-filter completion scope is exactly the finite route stated above. □

Theorem 1569.70 (CauchyFilterCompletion root filter route)**.** *Every accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier read by a completion-facing consumer factors through the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row before the completion data are consumed. The route cannot use quotient filters, selected limits, countable choice, host equality of completed points, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootFilterRoute`

Proof. Project the carrier of Definition 1569.1. Its displayed rows are exactly $C, W, T, R, L, H, K, P, N$, with the root filter source entering through C.

Apply Definition 1569.66. It forces every completion-facing read of the finite windows, tolerance ledger, regular readback, and Real seal to pass through the same $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source row behind C.

Next apply Definition 1569.67. The terminal seal L is available only after the source row, windows, tolerance ledger, and readback have been replayed through K and preserved by H, P, N.

Finally apply Definition 1569.69. The scope-closure theorem exhausts the displayed finite route and excludes quotient filters, selected limits, countable-choice schedules, host equality, and ambient completeness coordinates. □

Theorem 1569.71 (CauchyFilterCompletion public export)**.** *An accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier exports a public completion-facing route only through the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row. The exported route exposes no quotient filters, selected limits, countable-choice schedule, host equality of completed points, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPublicExport`

Proof. First compose carrier admission with source coverage. Apply Definition 1569.65 to keep $C, W, T, R, L, H, K, P, N$ inside one finite BHist packet, then apply Definition 1569.66 to force every read of W, T, R, L through the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source behind C.

Next compose the Real-seal handoff. Apply Definition 1569.67; the terminal seal L is reached only after the source, finite windows, tolerance ledger, and regular readback have been replayed through K and preserved by H, P, N.

Finally apply Definition 1569.70. That theorem packages the same source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame, displayed Cont, Pkg, and local naming data as the only completion-facing route.

The refusal clause follows because none of the composed obligations introduces a quotient-filter coordinate, selected limit, countable-choice schedule, host equality of completed points, or ambient-completeness witness. □

Theorem 1569.72 (CauchyFilterCompletion root sibling dependency route)**.** *Every accepted root completion read for* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *first consumes the sibling surfaces chapter 1601, chapter 333, chapter 1473, chapter 1612, chapter 126, chapter 301, chapter 291, and chapter 38 before it exposes the public completion route. The route therefore records the actual L10 sibling lattice rather than a kernel-only completion island.*

Proof. Start from carrier admission in Definition 1569.65. The finite packet already binds the completion carrier, source windows, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1569.66. It forces the root route to read the $\mathsf{FilterBase}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$ source surfaces before the completion-facing rows may be inspected.

Next use Definition 1569.67. The readback passes through finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 1569.70 and Definition 1569.71. These package the same rows as the public completion route, with the uniform and separated completion siblings supplying the completion-facing boundary and with no quotient filters, selected limits, countable choice, host equality, or ambient completeness. □

Lean4 ✓ BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionRootSiblingDependencyRoute

Definition 1569.73 (CauchyFilterCompletion basis route)**.** *For an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, the* basis-completion route *is the ordered completion-facing read that opens the displayed* FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *basis behind* C*, then the finite* StreamName$^{\uparrow}$ *windows* W*, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* T*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the terminal* Real$^{\uparrow}$ *seal* L*, and finally the structural rows* H, K, P, N*. The route exports no quotient filter, selected limit, host equality, countable-choice selector, detached completion object, or ambient completeness witness.*

Lean4 (definition) BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionBasisRoute

Theorem 1569.74 (CauchyFilterCompletion basis totality)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Every completion-facing consumer first reads the basis-completion route of Definition 1569.73; in particular, the displayed* FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *source behind* C *is admitted before any window, tolerance, readback, seal, transport, replay, provenance, or local naming row can be inspected.*

Lean4 ✓ BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket basis totality

Proof. Apply Definition 1569.46. It places the FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ source behind C before every downstream completion-facing read.

Then read the carrier rows in the order displayed by Definition 1569.73. The windows W, tolerance ledger T, readback R, and seal L are downstream from the admitted source, and H, K, P, N only transport, replay, source, and name that same route.

Finally use Definition 1569.82. The refused quotient, selected-limit, host-equality, choice, detached-completion, and ambient-completeness objects have no coordinate in F, so every completion-facing consumer is total only through the displayed basis route. □

Theorem 1569.75 (CauchyFilterCompletion basis Real seal)**.** *For an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, any consumer that reaches the terminal* Real$^{\uparrow}$ *seal* L *has already traversed the displayed* FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *basis behind* C*, the* StreamName$^{\uparrow}$ *windows* W*, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* T*, and the* RegSeqRat$^{\uparrow}$ *readback* R*.*

Proof. Start with Definition 1569.74. It fixes the source and window side of every completion-facing read at C, W, T, R before the seal can be inspected.

Apply Definition 1569.47. Metric-facing and uniform-facing readers share the same finite carrier route, so neither can reach a different seal source downstream.

The terminal row L is therefore a seal for the displayed basis-completion route, not a selected limit or host equality proof. The structural rows H, K, P, N preserve and name that same route after the seal boundary has been reached. □

Lean4 ✓ BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket basis real seal

Theorem 1569.76 (CauchyFilterCompletion basis uniform handoff)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer of an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier receives only the basis-completion route*

$$C, W, T, R, L, H, K, P, N.$$

The uniform handoff reads the FilterBase$^{\uparrow}$/CauchyFilter$^{\uparrow}$ *source behind* C*, finite windows, dyadic tolerances, regular readback, terminal Real seal, transport, replay, provenance, and local naming rows; it does not import quotient filters, selected limits, countable choice, host equality, or ambient completeness.*

Proof. Use Definition 1569.47. It identifies every uniform-facing read with the same carrier route used by the metric-facing completion consumer.

Apply Definition 1569.75. The terminal Real seal exposed to the uniform handoff is downstream from the displayed source, windows, tolerance ledger, and readback row.

Close with Definition 1569.82. Since the refused objects have no carrier coordinate, the uniform handoff cannot add a quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness witness beyond the basis-completion route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket basis uniform handoff`

Definition 1569.77 (CauchyFilterCompletion basis ledger route)**.** *For an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, the* basis-ledger route *is the source-window-tolerance-readback-seal ledger*

$$C \longmapsto W \longmapsto T \longmapsto R \longmapsto L$$

together with the structural transport, replay, provenance, and local naming rows H, K, P, N*. It is the visible ledger read that begins at the* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source behind* C*, opens the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*, extracts the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L *without adding quotient filters, selected limits, host equality, or countable-choice selectors.*

Lean4 (definition) `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionBasisLedgerRoute`

Theorem 1569.78 (CauchyFilterCompletion basis ledger exhaustion)**.** *Every basis-facing completion read of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier is exhausted by the basis-ledger route of Definition 1569.77. In particular, once the* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source behind* C *has been admitted, all visible completion data pass through* W, T, R, L, H, K, P, N *and no additional source, tolerance, readback, seal, or completion coordinate remains available.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket basis readback route`

Proof. Start with Definition 1569.74. It places the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source behind C before any basis-facing completion read can inspect downstream rows.

Apply Definition 1569.75. The terminal $\mathsf{Real}^{\uparrow}$ seal is reachable only after the finite windows W, tolerance ledger T, and regular readback row R have been traversed.

Finally use Definition 1569.76. Uniform-facing reads share the same displayed carrier route, so the structural rows H, K, P, N only transport, replay, package, and name the already visible ledger. The sibling FilterBaseUp, RegSeqRatUp, RealUp, and UniformCompletionUp chapters are therefore consumed through this finite route rather than through detached completion data. □

Theorem 1569.79 (CauchyFilterCompletion basis universal consumer route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier factors through the same basis-ledger route of Definition 1569.77. The consumer may repackage the route for metric, uniform, functorial, or Real-facing use, but it receives only the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source, windows, dyadic tolerances, regular readback, terminal seal, transport, replay, provenance, and local NameCert rows.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionBasisUniversalConsumerRoute`

Proof. Use Definition 1569.78 to identify the only available basis-facing completion data with $C, W, T, R, L, H, K, P, N$.

For the Real-facing branch, apply Definition 1569.75. The $\mathsf{Real}^{\uparrow}$ consumer reaches L only through the same source-window-tolerance-readback route.

For the uniform and functorial branches, apply Definition 1569.76. The $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ consumers can transport or repackage the finite ledger, but they cannot add quotient filters, selected limits, host equality, countable choice, or an ambient completeness witness. The $\mathsf{MetricCompletion}^{\uparrow}$ branch is the same route read through the metric-completion vocabulary. □

Theorem 1569.80 (CauchyFilterCompletion public basis-completion route)**.** *Every public* CauchyFilterCompletion *consumer factors through the displayed* $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source behind C, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal L, and the structural rows H, K, P, N of the accepted carrier $F = (C, W, T, R, L, H, K, P, N)$. The route exports no quotient filter, selected limit, host equality, countable choice, detached completion object, or ambient completeness witness.*

Proof. Start with basis totality, Definition 1569.74. It places the $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source behind C before any public completion-facing read can inspect downstream rows.

Use the Real seal and uniform handoff next. By Definition 1569.75, the terminal $\mathsf{Real}^{\uparrow}$ seal L is downstream of W, T, R; by Definition 1569.76, uniform-facing consumers receive the same finite route rather than a detached completion object.

Now apply the universal consumer route, Definition 1569.79. Metric, uniform, functorial, and Real-facing consumers all repackage the same basis-ledger route.

Finally attach the separated reflector of Definition 1569.81. The separated consumer may read its classifier only after the same source, window, tolerance, readback, seal, transport, replay, provenance, and local NameCert rows have been displayed, so the public export is exactly the basis-completion route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPublicBasisCompletionRoute`

Theorem 1569.81 (CauchyFilterCompletion separated reflector basis)**.** *Every* $\mathsf{SeparatedCompletion}^{\uparrow}$ *or separated* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier factors through the same basis-ledger route*

$$C, W, T, R, L, H, K, P, N.$$

The separated classifier is read only after the displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ *source behind C, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal L. The reflector route imports no quotient filter, selected limit, host equality, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket basis separated reflector`

Proof. Start with basis-ledger exhaustion, Definition 1569.78. It leaves $C, W, T, R, L, H, K, P, N$ as the only available completion-facing route after the displayed filter basis has been admitted.

Read the Real and uniform sides next. The basis Real seal of Definition 1569.75 places L after the source, window, tolerance, and readback rows, while Definition 1569.76 prevents a separated metric consumer from adding another completion surface.

Now attach the separated-completion packet route of chapter 1612. A SeparatedCompletion$^{\uparrow}$ consumer may read its separated classifier only after the same finite source, dense or metric handoff, tolerance, readback, and Real-seal rows are displayed.

The structural rows H, K, P, N transport, replay, source, and name that route. Since quotient filters, selected limits, host equality, countable choice, and ambient completeness have no coordinate in the accepted carrier, the separated reflector factors through the displayed basis-ledger route. □

Theorem 1569.82 (CauchyFilterCompletion no-choice boundary)**.** *For every accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, quotient filters, selected limits, countable-choice selectors, host equality, and ambient* Real$^{\uparrow}$ *completeness have no carrier coordinate in any completion-facing read. Such a read is confined to the source, window, tolerance, readback, seal, transport, replay, provenance, and local naming rows displayed by* F*.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket non choice boundary`

Proof. Inspect Definition 1569.1. The coordinates of F are exactly $C, W, T, R, L, H, K, P, N$.

Apply Definition 1569.2. Its ledger-refusal row excludes quotient filters, selected limits, host equality, countable-choice selectors, ambient completeness, and detached completion coordinates from the local certificate surface.

Close with Definition 1569.47. Since every metric-facing or uniform-facing consumer shares the same displayed route, none of the refused objects can be recovered by changing the completion vocabulary downstream. □

Theorem 1569.83 (CauchyFilterCompletion source-window totality)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier. Before any* MetricCompletion$^{\uparrow}$ *or* UniformCompletion$^{\uparrow}$ *consumer may read a completion-facing row, the packet exposes the* CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ *source* C*, the finite* StreamName$^{\uparrow}$ *window family* W*, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger* T*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the terminal* Real$^{\uparrow}$ *seal* L*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket source window totality`

Proof. Begin with Definition 1569.46. It places every completion-facing read behind the displayed CauchyFilter$^{\uparrow}$/FilterBase$^{\uparrow}$ source row C.

Apply Definition 1569.47. The metric and uniform completion consumers share the same finite route $C, W, T, R, L, H, K, P, N$, so the source, window, tolerance, readback, seal, transport, replay, provenance, and local naming rows are all exposed before the consumer reads completion vocabulary.

Use Definition 1569.39. The terminal Real seal is downstream of the same displayed source-window-tolerance-readback route, and no detached completion object remains outside that route. □

Theorem 1569.84 (CauchyFilterCompletion seal factorization obligation)**.** *Every* MetricCompletion$^{\uparrow}$*,* UniformCompletion$^{\uparrow}$*,* MetricCompletionFunctor$^{\uparrow}$*, or* CauchyCompletionFunctoriality$^{\uparrow}$ *consumer of an accepted* CauchyFilterCompletion$^{\uparrow}$ *carrier factors through the displayed rows*

$$C, W, T, R, L, H, K, P, N.$$

The factorization may repackage the Cauchy-filter source, StreamName windows, dyadic tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows, but it cannot add quotient filters, selected limits, host equality, choice, detached completion carriers, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket seal factorization obligation`

Proof. Start from Definition 1569.83. It supplies the complete source-window-tolerance-readback-seal route that every completion-facing consumer must display.

Apply Definition 1569.47. The metric and uniform consumers differ only in downstream vocabulary; both receive the same finite carrier rows.

Finally use Definition 1569.82. Quotient filters, selected limits, host equality, choice, detached completion carriers, and ambient completeness have no coordinate in the accepted packet, so no completion-functor consumer can add them while preserving the displayed factorization. □

Theorem 1569.85 (CauchyFilterCompletion window cofinality)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. Every completion-facing tolerance request that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal has a displayed cofinal window route through the same finite rows $C, W, T, R, L, H, K, P, N$: the request first reads the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row C, then a* $\mathsf{StreamName}^{\uparrow}$ *window in W, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and only then the seal row L with the structural rows H, K, P, N.*

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket window cofinality`

Proof. Start with the carrier of Definition 1569.1. It displays C as the only Cauchy-filter source and W as the only finite observation-window family.

Apply Definition 1569.4. This forces every accepted readback to pass through W and T before R is available.

Finally use the Real seal handoff of Definition 1569.3. The terminal seal remains downstream of C, W, T, R and upstream of H, K, P, N, so every completion-facing tolerance request has the displayed cofinal route. □

Theorem 1569.86 (CauchyFilterCompletion source exactness)**.** *Let $F = (C, W, T, R, L, H, K, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The source of any accepted completion read is exactly the displayed L10 route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

carried by C, W, T, R, L and preserved by the BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows H, K, P, N. No read can replace this source by a quotient filter, selected limit, detached stream, free tolerance, host real equality, countable-choice selector, or ambient completeness witness.*

Proof. Project F and apply Definition 1569.85. The cofinal window route already identifies the source side as C, W, T, R before the terminal seal L is read.

Use Definition 1569.46 to pin C to the displayed $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{FilterBase}^{\uparrow}$ source and Definition 1569.6 to pin every terminal read to L.

The structural rows then preserve the same route. Componentwise hsame transport is stored in H, displayed Cont replay is stored in K, and P, N source and name only this packet. Since the carrier has no row for the excluded objects, the accepted source is exact. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket source exactness`

Theorem 1569.87 (CauchyFilterCompletion ledger exhaustion)**.** *For an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier* $F = (C, W, T, R, L, H, K, P, N)$*, every completion, filter-limit, metric-completion, or uniform-completion consumer read is exhausted by the displayed ledger rows* $C, W, T, R, L, H, K, P, N$*. In particular, the L10 source route through* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *is the whole public source ledger; there is no additional quotient-filter row, selected-limit row, detached completion object, host equality row, countable-choice row, or ambient-completeness row available to any consumer.*

Proof. Begin with source exactness from Definition 1569.86. It confines every accepted source read to $C, W, T, R, L, H, K, P, N$.

For filter-limit consumers, use Definition 1569.11. For metric-facing and uniform-facing consumers, use Definition 1569.45 and Definition 1569.47. These theorems all reuse the same displayed route rather than adding a second consumer-specific source.

Finally apply Definition 1569.82. The refused quotient, selector, choice, host-equality, and completeness objects have no carrier coordinate. Thus the ledger is exhausted by the displayed L10 source route and the structural rows of F. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket ledger exhaustion`

Theorem 1569.88 (CauchyFilterCompletion public NameCert export)**.** *Let* $F = (C, W, T, R, L, H, K, P, N)$ *be an accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *carrier. The public* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *export is exactly the route* $C, W, T, R, L, H, K, P, N$ *isolated by the carrier, the completion handoff, the completion-functor factorization, the cofinal regular readback, and source/ledger exhaustion. A public consumer receives the displayed Cauchy-filter source, finite windows, dyadic tolerance ledger, regular readback, terminal Real seal, transport, replay, provenance, and local naming rows, and receives no quotient-filter, selected-limit, host-equality, choice, or ambient-completeness object.*

Proof. Project the carrier of Definition 1569.1. The visible packet is $C, W, T, R, L, H, K, P, N$, with C as source, W, T, R as finite readback rows, L as terminal seal, and H, K, P, N as transport, replay, provenance, and local naming rows.

Apply completion handoff and completion-functor factorization, Definition 1569.47 and Definition 1569.52. They place every completion-facing, metric-facing, uniform-facing, and functor-facing read on the same displayed route.

Use cofinal regular readback together with source exactness, Definition 1569.53 and Definition 1569.86. These theorems identify the cofinal public readback through the $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows already carried by F.

Finally apply ledger exhaustion, Definition 1569.87. The quotient-filter, selected-limit, host-equality, choice, and ambient-completeness alternatives have no carrier coordinate, so the public NameCert export is exactly the displayed route. □

Lean4 ✓ `BEDC.Derived.CauchyFilterCompletionUp.CauchyFilterCompletionPublicNameCert export`

Theorem 1569.89 (CauchyFilterCompletion bridged route)**.** *The accepted* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *route exports its completion-facing bridge only through the existing semantic* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *row, uniform filter readback, completion handoff, and completion-functor factorization rows. Thus any bridge-facing consumer receives exactly the displayed route*

$$C, W, T, R, L, H, K, P, N$$

from the accepted carrier and receives no quotient filter, selected limit, host equality, countable-choice selector, detached completion object, or ambient completeness coordinate.

Proof. Project the carrier and semantic NameCert rows of Definition 1569.1 and Definition 1569.2. They fix the only exported coordinates as the Cauchy-filter source, finite windows, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Use uniform readback and completion handoff, Definition 1569.5 and Definition 1569.47. They keep uniform-facing and completion-facing consumers on that same displayed route.

Finally apply Definition 1569.52. The functorial vocabulary can repackage the route for completion consumers, but it adds no coordinate outside $C, W, T, R, L, H, K, P, N$, so the bridge-facing export is exactly the finite NameCert route. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket bridged route`

Theorem 1569.90 (CauchyFilterCompletion bridged-route maturity)**.** *The* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *bridged route is mature exactly when the semantic* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletion}^{\uparrow}}$ *obligations, uniform-filter readback, completion handoff, completion-functor factorization, and displayed rows*

$$C, W, T, R, L, H, K, P, N$$

are read as one public export. In that case every bridge-facing consumer receives the Cauchy-filter source, finite windows, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows together, and receives no quotient filter, selected limit, countable-choice selector, detached completion object, host equality, or ambient completeness coordinate.

Proof. Use the bridged route of Definition 1569.89. It already identifies the exported bridge as the semantic NameCert row, uniform readback, completion handoff, and completion-functor factorization over $C, W, T, R, L, H, K, P, N$.

The forward direction reads maturity as public export. Apply Definition 1569.2, Definition 1569.5, Definition 1569.47, and Definition 1569.52; their rows are the same carrier rows displayed by the bridged route.

For the reverse direction, assume those rows are read as one public export. The semantic obligations name the carrier surface, uniform readback and completion handoff keep the consumer on the displayed route, and completion-functor factorization repackages no additional coordinate. Thus the bridged route has exactly the mature public surface described in the statement. □

Lean4 ✓ `BEDC.Derived.CauchyfiltercompletionUp.CauchyFilterCompletionPacket bridged route maturity`

Chapter 1570

A Concrete Naming Certificate for $\mathsf{IntervalNewton}^{\uparrow}$

Following Moore–Kearfott–Cloud interval analysis, the $\mathsf{IntervalNewton}^{\uparrow}$ packet records interval Newton and Krawczyk enclosure certificates as finite BEDC rows. It names a displayed box, function enclosure, slope or derivative enclosure, correction row, narrowing ledger, validated readback route, real-name readback, transport witnesses, replay data, provenance, and local naming row. It is not a solver oracle, a host zero-existence theorem, a global root-isolation method, a completeness assertion, or a floating-point semantics package.

The carrier reads interval operations through chapter 302, finite validated enclosures through chapter 357, displayed correction provenance through chapter 223, and finite Taylor or slope remainders through chapter 442. These sibling rows are the entire dependency surface for the enclosure certificates below.

Definition 1570.1 (Interval Newton carrier)**.** *An* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, F, D, N, K, V, R, H, C, P, L)$$

with the following displayed coordinates:

1. *B is the interval box row, read through* $\mathsf{IntervalArithmetic}^{\uparrow}$ *endpoint and enclosure ledgers;*
2. *F is the function-enclosure row over B, with every enclosure read routed through finite interval operations;*
3. *D is the derivative or slope enclosure row, supplied either directly as an interval row or through a finite* $\mathsf{TaylorModel}^{\uparrow}$ *remainder packet;*
4. *N is the finite Newton or Krawczyk correction row, including the displayed center, inverse or preconditioner row, and correction enclosure;*
5. *K is the containment and narrowing ledger recording that the correction row stays inside, or narrows, the displayed box B;*
6. *V is the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *enclosure row read by downstream consumers;*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route for the center, endpoint, and correction observations;*
8. *H records componentwise* hsame *transports for the box, function, derivative, correction, containment, readback, and validated rows;*

9. *C records the* Cont *routes by which the displayed rows are replayed inside their source certificates;*
10. *P is the* Pkg *provenance over* IntervalArithmetic$^{\uparrow}$, ValidatedNumerics$^{\uparrow}$, NewtonIteration$^{\uparrow}$, TaylorModel$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and the displayed kernel rows;*
11. *L is the local* NameCert$_{\text{IntervalNewton}^{\uparrow}}$ *row.*

Two accepted carriers are classified together only by transporting these displayed coordinates through the recorded hsame *and* Cont *rows. The classifier has no coordinate for a hidden solver state, a selected zero, host floating-point behavior, quotient real equality, or completeness of the ambient real line.*

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonTasteGate single carrier alignment`

Theorem 1570.2 (Interval Newton NameCert obligations)**.** *The local* NameCert$_{\text{IntervalNewton}^{\uparrow}}$ *surface for the carrier of Definition 1570.1 consists of six rows:*

1. *carrier admission for the finite packet $I = (B, F, D, N, K, V, R, H, C, P, L)$;*
2. *correction containment saying that the displayed Newton or Krawczyk row N is admitted only with its containment and narrowing ledger K;*
3. *interval-arithmetic soundness for the box, function, derivative or slope, and correction enclosures;*
4. *validated-consumer handoff through the displayed* ValidatedNumerics$^{\uparrow}$ *row V and the readback row R;*
5. *transport under the componentwise* hsame *rows of H and replay under the* Cont *routes of C;*
6. *ledger non-escape, meaning every consumer-visible claim is projected from $B, F, D, N, K, V, R, H, C, P, L$ and from no other numerical object.*

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonNameCertObligations`

Proof. Carrier admission is projection from Definition 1570.1. The box, function enclosure, derivative or slope enclosure, correction, containment ledger, validated readback, real-name route, transport, replay, provenance, and local name are visible fields of one finite packet.

Correction containment is read from N together with K. The packet may expose a Krawczyk or interval Newton row only when the same displayed ledger records how that row is contained in, or narrows, the box B.

Interval-arithmetic soundness is inherited by routing all endpoint and enclosure operations through the IntervalArithmetic$^{\uparrow}$ rows named in P. When a derivative or slope enclosure is supplied by a Taylor model, the packet reads only the finite TaylorModel$^{\uparrow}$ remainder and interval-operation rows.

The consumer handoff is then V together with R. Transport and replay use only the componentwise rows H and C, so every exported obligation is exhausted by the carrier coordinates and no solver oracle or external real-semantics datum remains. □

Theorem 1570.3 (Interval Newton Krawczyk enclosure)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted* IntervalNewton$^{\uparrow}$ *carrier. Every Krawczyk or interval Newton enclosure exported by I narrows the displayed box B only through the* IntervalArithmetic$^{\uparrow}$ *operation rows named by P and through the displayed* NewtonIteration$^{\uparrow}$ *or* TaylorModel$^{\uparrow}$ *rows used to form N and D. The exported enclosure is exactly the row recorded by K and validated by V.*

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonKrawczykEnclosure`

Proof. Start with a consumer-visible narrowing claim. By the carrier definition, the only box available to the packet is B, and the only correction row available is N.

Read the arithmetic used by the correction through the $\mathsf{IntervalArithmetic}^{\uparrow}$ dependency rows named in P. The center, inverse or preconditioner, function enclosure, derivative or slope enclosure, and final interval operation are all displayed rows of the same finite ledger.

If the correction row is presented as a Newton step, its provenance is the finite $\mathsf{NewtonIteration}^{\uparrow}$ row named in P. If the slope or derivative enclosure is presented by finite Taylor data, it is read through the $\mathsf{TaylorModel}^{\uparrow}$ row named in P. In either case, the packet keeps the operation inside visible BEDC rows.

Finally project K and V. The containment ledger K records the narrowed box, and the validated row V is the only downstream enclosure exported by the packet. No separate analytic root theorem, host solver state, or floating-point rounding semantics is consulted. □

Theorem 1570.4 (Interval Newton Krawczyk radius lock)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted $\mathsf{IntervalNewton}^{\uparrow}$ carrier whose interval-arithmetic row supplies a Krawczyk enclosure. Then the correction radius, the Taylor or slope remainder row, the $\mathsf{Real}^{\uparrow}$ seal, and the $\mathsf{ValidatedNumerics}^{\uparrow}$ consumer boundary remain in the same finite packet I before any zero-enclosure read is exported.*

In particular, the exported zero-enclosure read factors through the visible rows $B, F, D, N, K, V, R, H, C, P, L$: N supplies the correction radius, D supplies the slope or $\mathsf{TaylorModel}^{\uparrow}$ remainder row, R supplies the real-name seal route, and V supplies the validated-consumer boundary.

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonKrawczykRadiusLock`

Proof. Project the accepted carrier of Definition 1570.1. The correction radius is not a separate analytic datum; it is part of the displayed correction row N, and its containment is read with the ledger K.

Apply Definition 1570.3. A Krawczyk enclosure can narrow the displayed box only through the interval-arithmetic operation rows named in P and through the displayed $\mathsf{NewtonIteration}^{\uparrow}$ or $\mathsf{TaylorModel}^{\uparrow}$ rows used to form N and D.

Use the interval-arithmetic soundness obligation of Definition 1570.2. It keeps the box, function, slope or Taylor remainder, and correction enclosures inside the same finite interval-arithmetic replay before K and V are visible.

Finally apply the validated-consumer handoff obligation of Definition 1570.2 together with Definition 1570.7. The validated row V and readback row R are downstream of B, F, D, N, K, so no zero-enclosure read can leave the packet before the correction radius, remainder row, real-name route, and validated boundary have all been bound. □

Theorem 1570.5 (Interval Newton correction transport scope)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted $\mathsf{IntervalNewton}^{\uparrow}$ carrier. Every transported correction row exported by I factors through the visible interval box B, the $\mathsf{IntervalArithmetic}^{\uparrow}$ operation rows named in P, the finite $\mathsf{TaylorModel}^{\uparrow}$ row or derivative and slope row D, the $\mathsf{NewtonIteration}^{\uparrow}$ correction provenance for N, and the $\mathsf{ValidatedNumerics}^{\uparrow}$ row V before the validated enclosure is exported.*

Consequently the transported correction cannot read a solver oracle, a host zero, floating-point semantics, quotient real equality, or ambient completeness outside the displayed carrier rows.

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonCorrectionTransportScope`

Proof. Project the carrier of Definition 1570.1. The correction surface is B, F, D, N, K, V, while R, H, C, P, L supply readback, transport, replay, provenance, and local naming.

Invoke Definition 1570.3. It forces every Krawczyk or interval Newton enclosure to narrow the displayed box B only through $\mathsf{IntervalArithmetic}^{\uparrow}$ operations and through the displayed $\mathsf{NewtonIteration}^{\uparrow}$ or $\mathsf{TaylorModel}^{\uparrow}$ rows used to form N and D.

The transported row then passes through H and C without changing its dependency route. The validated export is V, so a downstream validated-numerics consumer reads the correction only after the interval, derivative or Taylor-model, Newton-iteration, containment, and validation rows have appeared.

No excluded datum is a carrier coordinate or a provenance row in P. Thus the correction transport remains inside the visible interval, Taylor-model or derivative, Newton-iteration, validated-numerics, transport, replay, and local naming rows. □

Theorem 1570.6 (Interval Newton Krawczyk scope)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier. Every Krawczyk enclosure read exported by I factors through the displayed* $\mathsf{IntervalArithmetic}^{\uparrow}$ *rows for B, F, D, N, through either the finite* $\mathsf{TaylorModel}^{\uparrow}$ *remainder row or the derivative and slope row D, through the* $\mathsf{NewtonIteration}^{\uparrow}$ *correction provenance for N, through the validated enclosure row V, and through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route R before any zero-enclosure consumer reads it.*

Consequently a zero-enclosure consumer cannot inspect a Krawczyk row without first exposing the containment ledger K, the correction-transport rows H, C, the provenance row P, and the local $\mathsf{NameCert}_{\mathsf{IntervalNewton}^{\uparrow}}$ *row L for the same packet.*

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonKrawczykScope`

Proof. Start with the Krawczyk enclosure theorem Definition 1570.3. It keeps the displayed narrowing inside the interval-arithmetic operation rows and the Newton or Taylor-model rows that form N and D.

Apply the radius lock of Definition 1570.4. This places the correction radius, Taylor or slope remainder, real-name route, and validated-consumer boundary inside the same finite carrier before any zero-enclosure read is exported.

Use the correction transport scope of Definition 1570.5. Transport through H and replay through C preserve the visible interval, derivative or Taylor-model, Newton-iteration, and validated-numerics route rather than adding a solver oracle or host zero.

Finally apply the validated consumer boundary of Definition 1570.7. The consumer-facing read is V followed by R, with P and L recording provenance and local naming. Since the carrier has no coordinate outside $B, F, D, N, K, V, R, H, C, P, L$, the stated dependency route is forced for every Krawczyk read. □

Theorem 1570.7 (Interval Newton validated consumer boundary)**.** *Every* $\mathsf{ValidatedNumerics}^{\uparrow}$ *or root-isolation consumer of an accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *packet reads the correction only through the displayed carrier*

$$I = (B, F, D, N, K, V, R, H, C, P, L).$$

The consumer receives the interval box, function enclosure, derivative or slope enclosure, correction row, containment ledger, validated enclosure row, real-name readback route, transport rows, replay routes, provenance, and local naming row. It receives no solver oracle, no host zero, no floating-point semantics, no quotient real equality, and no completeness datum.

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonValidatedConsumerBoundary`

Proof. Project the accepted carrier. A validated or root-isolation consumer can inspect only the fields $B, F, D, N, K, V, R, H, C, P, L$.

Use Definition 1570.2 to identify the exported obligation rows. The consumer handoff is V and R, while transport and replay are restricted to H and C.

Use Definition 1570.3 for the correction read. The Krawczyk or interval Newton row is visible only through $\mathsf{IntervalArithmetic}^{\uparrow}$ operations and through the displayed $\mathsf{NewtonIteration}^{\uparrow}$ or $\mathsf{TaylorModel}^{\uparrow}$ provenance named in P.

Thus every consumer-facing row is a projection of the finite packet. Since no carrier coordinate names a solver oracle, selected zero, hardware arithmetic semantics, quotient-real equality witness, or completeness theorem, those data cannot be exported by the boundary. □

Theorem 1570.8 (Interval Newton containment obligation)**.** *For every accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier* $I = (B, F, D, N, K, V, R, H, C, P, L)$*, the containment obligation is exactly the visible pair* (N, K)*. The correction row* N *may be consumed only with the containment and narrowing ledger* K*, and every* hsame *transport or* Cont *replay of the correction must preserve the same box row* B*, the same correction row* N*, and the same containment ledger* K *before the local* $\mathsf{NameCert}_{\mathsf{IntervalNewton}^{\uparrow}}$ *row* L *is read.*

Proof. Project the carrier. The only correction coordinate is N, and the only containment coordinate attached to it is K.

Use Definition 1570.2. It requires correction containment as a named row of the local NameCert surface, so a consumer cannot detach N from the ledger K.

Transport and replay then use H and C. Since those rows are componentwise over the carrier coordinates, they preserve B, N, K before the local naming row L is exposed. □

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonContainmentObligation`

Theorem 1570.9 (Interval Newton interval-arithmetic soundness obligation)**.** *For every accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier* $I = (B, F, D, N, K, V, R, H, C, P, L)$*, interval-arithmetic soundness is the obligation that the box row* B*, the function-enclosure row* F*, the derivative or slope row* D*, and the correction row* N *are all replayed through the* $\mathsf{IntervalArithmetic}^{\uparrow}$ *rows named by* P*. When* D *is supplied by a finite* $\mathsf{TaylorModel}^{\uparrow}$ *remainder packet, the same obligation routes the Taylor remainder through interval endpoint operations before* K *and* V *are visible to consumers.*

Proof. Read the arithmetic provenance from P. It names the interval-arithmetic surface through which endpoint, enclosure, derivative or slope, and correction operations are replayed.

Apply Definition 1570.3. The narrowing claim is not a free analytic assertion; it is assembled from B, F, D, N through the displayed interval operations.

If the derivative or slope row is supplied by $\mathsf{TaylorModel}^{\uparrow}$, the finite Taylor remainder is still routed through the same interval endpoint operations before containment is recorded in K. The validated row V is therefore downstream of the interval-arithmetic replay, not a replacement for it. □

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonIntervalArithmeticSoundnessObligation`

Theorem 1570.10 (Interval Newton validated handoff obligation)**.** *For every accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier* $I = (B, F, D, N, K, V, R, H, C, P, L)$*, the validated handoff obligation is the route from the validated enclosure row* V *and readback row* R *through the transport row* H*, replay row* C*, provenance row* P*, and local naming row* L*. A downstream* $\mathsf{ValidatedNumerics}^{\uparrow}$ *consumer may read the correction only after* V, R, H, C, P, L *expose the same interval box, correction containment, and provenance rows already present in* B, F, D, N, K*.*

Proof. Start from the validated consumer boundary. It says the consumer receives the displayed carrier rows and no solver oracle, host zero, floating-point semantics, quotient real equality, or completeness datum.

The handoff itself is V together with R. These rows expose the validated enclosure and readback route, while H and C preserve them under transport and replay.

Finally P and L provide provenance and local naming. Since the route still points back to B, F, D, N, K, the validated consumer can read only the correction already certified by containment and interval-arithmetic soundness. □

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonValidatedHandoffObligation`

Theorem 1570.11 (Interval Newton Real readback scope)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing validated enclosure read first exposes the interval box B, function-enclosure row F, derivative or slope row D, correction row N, and containment ledger K. It then exposes the validated row V and only then the readback row R.*

Transport through H, replay through C, provenance through P, and local naming through L preserve that same route. They cannot add a solver oracle, selected zero, quotient real equality, floating-point semantics, or ambient completeness.

Proof. Project the carrier of Definition 1570.1. The only Real-facing or RegSeqRat-facing readback coordinate is R, and it is stored with $B, F, D, N, K, V, H, C, P, L$ in the same finite packet.

Apply Definition 1570.3. The Krawczyk or interval Newton enclosure is assembled from the displayed box, function, derivative or slope, correction, and containment rows before it can be named as an enclosure.

Use Definition 1570.9. This keeps B, F, D, N inside the interval-arithmetic replay named by P before the containment ledger K and validated row V become consumer-visible.

Finally apply Definition 1570.10. The handoff places V before R, and H, C, P, L only transport, replay, name, and package the same route. Since no excluded datum is a carrier coordinate, the Real or RegSeqRat readback cannot export one. □

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonRealReadbackScope`

Theorem 1570.12 (Interval Newton scoped consumer route)**.** *Let $I = (B, F, D, N, K, V, R, H, C, P, L)$ be an accepted* $\mathsf{IntervalNewton}^{\uparrow}$ *carrier. Every scoped consumer route for a zero-enclosure or correction read is exactly the displayed path through containment, interval-arithmetic soundness, validated handoff, and Real or RegSeqRat readback:*

$$B, F, D, N \longmapsto K \longmapsto V \longmapsto R \longmapsto H, C, P, L.$$

The route first binds the correction row N to the containment ledger K, then routes B, F, D, N through the $\mathsf{IntervalArithmetic}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, *and* $\mathsf{TaylorModel}^{\uparrow}$ *provenance named by P, then exposes the* $\mathsf{ValidatedNumerics}^{\uparrow}$ *handoff V, and only then permits the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. It exports no solver oracle, host zero, floating-point semantics, quotient real equality, selected representative, or ambient completeness.*

Proof. Begin with the containment obligation of Definition 1570.8. It identifies the scoped correction surface as the pair (N, K) over the displayed box B, so a consumer cannot detach a correction read from the containment and narrowing ledger.

Next apply the interval-arithmetic soundness obligation and the Krawczyk enclosure theorem, Definition 1570.9 and Definition 1570.3. These route the box, function enclosure, derivative or

slope row, and correction row through the interval-arithmetic operations and through the displayed Newton or Taylor-model provenance named by P.

Use the validated handoff obligation and the validated consumer boundary, Definition 1570.10 and Definition 1570.7. They place V after B, F, D, N, K and keep transport, replay, provenance, and local naming inside H, C, P, L.

Finally apply the Real readback scope of Definition 1570.11. The readback row R is downstream of the validated row, and the carrier has no coordinate for any excluded solver, host, quotient, selection, or completeness datum. Therefore the scoped consumer route is exactly the displayed path. □

Lean4 ✓ `BEDC.Derived.IntervalNewtonUp.IntervalNewtonScopedConsumerRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalNewton}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalNewton}\uparrow}, \mathsf{scopedClosed})$

Constructive story: An $\mathsf{IntervalNewton}^{\uparrow}$ packet is generated from a displayed interval box, function enclosure, derivative or slope enclosure, finite Newton or Krawczyk correction row, containment ledger, validated readback row, real-name route, transport witnesses, replay routes, provenance, and local NameCert row.

Scope closed: Containment, interval-arithmetic soundness, validated handoff, transport, and ledger non-escape are bound to the displayed BEDC primitive rows of the same interval Newton and Krawczyk packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalNewton}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close solver correctness, existence of a zero, uniqueness, floating-point hardware semantics, classical real completeness, quotient real equality, or bridge checking.

Upgrade path: Verification closure requires Lean coverage for the displayed containment, arithmetic, handoff, transport, and non-escape routes.

Chapter 1571

A Concrete Naming Certificate for $\mathsf{SequentialClosure}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{SequentialClosure}^{\uparrow}$ packet records the finite route by which a displayed sequence-limit witness may be consumed as closure evidence without importing host topological closure. It sits over the topology surface of chapter 88, the metric source of chapter 57, the finite windows of chapter 126, the regular rational handoff of chapter 301, the Real seal of chapter 38, the located-set boundary of chapter 1541, and the sequence-consumer interface of chapter 1466. The packet is a finite NameCert surface for certified sequence-limit closure; it is not an ambient closure operator, not a compactness theorem, not quotient stream equality, not arbitrary subsequence choice, and not completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1571.1 (SequentialClosure carrier)**.** *A* $\mathsf{SequentialClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row, including the named neighbourhood indices and the classifier transport available to topology-facing consumers.*
2. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row over the same visible carrier, including the distance classifier and metric comparison rows used when neighbourhood tests are read metrically.*
3. *S is the displayed subset or* $\mathsf{LocatedSet}^{\uparrow}$ *row whose admitted points form the source surface for the sequence.*
4. *Q is the sequence source row. It records a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of admitted sequence entries over S together with the point-name rows inspected by the packet.*
5. *L is the candidate limit row, naming the target point over the common topology and metric source without asserting that it is host-closure equivalent to S.*
6. *U is the neighbourhood and metric test ledger. Each displayed request records either a topology neighbourhood row around L or a metric tolerance row around L, together with the sequence-entry test that answers it.*
7. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the inspected sequence windows and the index budgets consumed by U.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row translating the displayed metric tolerance windows into regular rational readback data before the Real distance boundary is consulted.*
9. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row certifying the accepted distance comparison downstream of R and U.*
10. *H records componentwise* hsame *transport for the topology row, metric row, subset or located-set row, sequence source, candidate limit, test ledger, StreamName windows, RegSeqRat handoff, Real seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a closure-consumer request through $T, M, S, Q, L, U, W, R, A$.*
12. *P is the* Pkg *provenance over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for host topological closure, arbitrary subsequence choice, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached* $\mathsf{SequentialContinuity}^{\uparrow}$ *theorem.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureTasteGate single carrier alignment`

Theorem 1571.2 (SequentialClosure NameCert obligations)**.** *For every accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *carrier $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the ambient topology, ambient metric, subset or located-set row, sequence source, candidate limit, neighbourhood and metric tests, finite StreamName windows, RegSeqRat handoff, Real seal, transport, continuation, provenance, and local naming row;*
2. *neighbourhood-test stability, requiring each topology or metric request around L to replay through the displayed test ledger U over the same T, M, S, Q, L rows;*
3. *sequence-window coverage, requiring every accepted closure read to expose the finite StreamName windows W that name the inspected sequence entries and index budgets;*
4. *limit-row handoff, requiring the RegSeqRat row R and Real seal A to be read downstream of the displayed metric test ledger before a distance conclusion is exported;*
5. *classifier transport, requiring all* hsame *comparison of closure packets to preserve the topology, metric, subset, sequence, limit, test, window, handoff, seal, continuation, provenance, and local NameCert rows componentwise;*
6. *non-escape from the host closure boundary, excluding host topological closure, arbitrary subsequence choice, quotient stream equality, open-cover compactness, and detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureCarrier namecert obligations`

Proof. Project the carrier of Definition 1571.1. The projection exposes exactly the topology row, metric row, subset or located-set row, sequence source, candidate limit, test ledger, finite StreamName windows, RegSeqRat handoff, Real seal, and structural rows H, C, P, N.

Neighbourhood-test stability is forced by the continuation row C. A closure-facing request reaches a neighbourhood or metric conclusion only by replaying the common T, M, S, Q, L source rows and then consulting the displayed ledger U.

Sequence-window coverage is the next step in the same replay. The ledger W is the only place where inspected sequence windows and index budgets are stored, so an accepted read cannot replace the displayed sequence by an arbitrary subsequence or by quotient stream equality.

The limit-row handoff is downstream of those windows. The RegSeqRat row R translates the metric tolerance read into regular rational data, and the Real seal A certifies the accepted distance comparison only after U and W have been consumed.

Finally, componentwise transport by H preserves every displayed coordinate, while P and N package only that finite route. Since host topological closure, arbitrary subsequence choice, quotient stream equality, open-cover compactness, and detached Real completeness have no carrier row, no continuation edge, and no provenance entry, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureNameCert obligation surface`

Theorem 1571.3 (SequentialClosure sequence-limit handoff)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. Every closure consumer of the displayed candidate limit factors through*

$$T, M, S, Q, L \longmapsto U, W \longmapsto R, A \longmapsto H, C, P, N.$$

Thus the consumer receives closure evidence only through the displayed sequence windows, the RegSeqRat handoff, the Real seal, the topology and metric tests, and the local NameCert rows; no host closure operator, arbitrary sequence selection, quotient stream equality, compactness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureSequenceLimitHandoff`

Proof. Start with a closure consumer of the candidate limit row L. Carrier admission gives access to L only together with the common topology row T, metric row M, subset or located-set row S, and sequence source row Q.

Replay the continuation row C. The consumer must pass through the neighbourhood and metric test ledger U and then through the finite StreamName window ledger W, so the inspected sequence entries and their index budgets remain displayed throughout the read.

The metric side of the request then factors through R and A. The RegSeqRat handoff records the regular rational readback for the tolerance window, and the Real seal records the accepted distance comparison after the topology and metric tests have already been displayed.

Transport, provenance, and local naming through H, P, N preserve the same dependency order. This proves the stated factorization and leaves no available row from which a consumer could extract host closure, arbitrary sequence selection, quotient stream equality, compactness, or detached completeness. □

Theorem 1571.4 (SequentialClosure RegSeqRat limit route)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. Any metric-side closure read of the candidate limit row L must expose the finite window ledger W, translate the requested tolerance through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R, and seal the resulting distance comparison through the* $\mathsf{Real}^{\uparrow}$ *row A. Thus the packet exports a sequence-limit route only as displayed RegSeqRat/Real data downstream of T, M, S, Q, L, U, W.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureRegSeqRatLimitRoute`

Proof. Project the packet carrier. The rows T, M, S, Q, L name the ambient topology, ambient metric, source row, sequence source, and candidate limit, while U records the neighbourhood and metric requests that a closure consumer may ask.

The metric request cannot bypass the finite windows. The continuation row C first exposes the StreamName windows W, so the inspected sequence entries and index budgets remain visible before any distance comparison is read.

The handoff then passes through R and A. The row R records the regular rational readback for the displayed tolerance, and the row A seals the corresponding Real comparison. Transport by H and packaging by P, N preserve this order, so no detached Real completeness or host limit principle is exported. □

Theorem 1571.5 (SequentialClosure LocatedSet stability)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet whose source row S is read as a* $\mathsf{LocatedSet}^{\uparrow}$ *surface. If a second packet is componentwise* hsame *to K on $T, M, S, Q, L, U, W, R, A, C, P, N$, then admitted sequence entries, located-set membership reads, RegSeqRat handoffs, and Real seals are transported through the same S row. The stability claim is only this finite LocatedSet-preserving replay.*

Proof. Start from the componentwise hsame data recorded by H. It identifies the topology and metric rows, the located-set source row S, the displayed sequence row Q, the candidate limit row L, and the test and window ledgers U, W.

Membership stability is read before the limit handoff. Each accepted sequence entry is admitted through the same located-set source surface S, and the transported packet can only replay those displayed entries because no arbitrary subsequence or external closure row appears in the carrier.

Finally transport the metric side through R and A. The RegSeqRat and Real rows are downstream of the same located-set and sequence windows, so the transported closure read preserves located-set admission while adding no compactness, quotient stream equality, or detached completeness assertion. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureLocatedSetStability`

Theorem 1571.6 (SequentialClosure topology handoff)**.** *For an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$, every topology-facing closure consumer receives the candidate limit row through the finite handoff*

$$T, M, S, Q, L, U, W, R, A, H, C, P, N.$$

Neighbourhood requests are answered through U over the common topology row T, metric requests are answered through M and then R, A, and the exported closure evidence is repacked by the local $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row N without a host closure operator.*

Proof. Begin with a topology-facing request around the candidate limit L. The carrier admits L only beside the ambient topology row T, metric row M, source row S, and displayed sequence row Q.

The request ledger U determines how the consumer may proceed. A purely neighbourhood-facing read stays over T, S, Q, L, U, W, while a metric-facing read also consumes the distance comparison route through M, R, A.

The final packaging is local. Transport by H, replay by C, provenance by P, and naming by N repack the same finite rows as $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ evidence. Since the packet has no coordinate for host topological closure, compactness, arbitrary sequence selection, or $\mathsf{Real}^{\uparrow}$ completeness, the topology and metric handoff exports only the displayed closure route. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureTopologyHandoff`

Theorem 1571.7 (SequentialClosure root sequence carrier)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. A sequence-closure carrier read is available only after the packet displays the* $\mathsf{LocatedSet}^{\uparrow}$ *or subset source row S, the* $\mathsf{Met}^{\uparrow}$ *row M, the* $\mathsf{StreamName}^{\uparrow}$ *row Q, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the* $\mathsf{Real}^{\uparrow}$ *seal A, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row N in the same finite* BHist *carrier.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureRootSequenceCarrier`

Proof. Project the carrier of Definition 1571.1. The sequence row Q is not a detached stream; it is read beside the source row S, metric row M, test ledger U, window ledger W, handoff row R, and Real seal A.

Use Definition 1571.2 to identify the local NameCert obligations and Definition 1571.3 to force every closure consumer through the displayed sequence-limit route. The RegSeqRat limit route of Definition 1571.4 supplies the rational readback boundary for metric requests.

The located-set stability theorem Definition 1571.5 keeps source membership and admitted sequence entries over the same S row, while Definition 1571.6 repacks the topology-facing read through H, C, P, N. Hence the carrier read is available only with the listed finite BHist rows displayed. □

Theorem 1571.8 (SequentialClosure root limit ledger)**.** *For an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$, the limit witness ledger is finite: all accepted reads of the candidate limit row L pass through U, W, R, A, H, C, P, N. The packet refuses selected subsequences, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as limit-witness sources.*

Proof. The finite ledger is visible in the carrier definition: requests are stored in U, inspected sequence windows in W, rational readback in R, Real comparison in A, and structural packaging in H, C, P, N.

Apply Definition 1571.3 for the dependency order from T, M, S, Q, L to U, W, R, A and then to H, C, P, N. Apply Definition 1571.4 for the metric side of the same ledger.

The refusal clauses are forced by absence of carrier coordinates. Neither selected subsequences nor quotient stream equality appear in Q or W, countable choice has no row in U or C, and ambient $\mathsf{Real}^{\uparrow}$ completeness is not part of the seal A. The NameCert obligations of Definition 1571.2 therefore close only the displayed finite ledger. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureRootLimitLedger`

Theorem 1571.9 (SequentialClosure root topology transport)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. Any topology-facing transport of the closure read preserves the same sequence-limit rows $T, M, S, Q, L, U, W, R, A$ under componentwise* hsame *and displayed* Cont *replay, and then repackages the transported read through P, N.*

Proof. Start with the componentwise hsame row H. By Definition 1571.5, transport preserves the source row S, admitted sequence entries, RegSeqRat handoffs, and Real seals along the same displayed coordinates.

Replay is controlled by C. The handoff theorem Definition 1571.3 keeps the sequence-limit route over $T, M, S, Q, L, U, W, R, A$, and the topology handoff theorem Definition 1571.6 states that neighbourhood-facing consumers receive no host closure operator.

Finally P and N repackage the transported rows as the local NameCert surface described in Definition 1571.2. Thus topology transport preserves the same finite sequence-limit ledger rather than creating a second closure witness. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureRootTopologyTransport`

Theorem 1571.10 (SequentialClosure Kuratowski idempotence)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. A second sequential-closure read whose source row is the closure output of K contracts to the same displayed limit ledger whenever its witnesses are expressed by* $\mathsf{StreamName}^{\uparrow}$ *windows over Q,* $\mathsf{RegSeqRat}^{\uparrow}$ *tail data through R, and* $\mathsf{Real}^{\uparrow}$ *equality seals through A. The contraction is an idempotence obligation for the finite packet only: it does not assert an ambient Kuratowski closure operator on all subsets of the host space.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureKuratowskiIdempotence`

Proof. Read the first closure packet through Definition 1571.8. The candidate limit row is exported only after the finite test ledger U, StreamName windows W, RegSeqRat handoff R, and Real seal A have been displayed.

For the second read, inspect its source witness. If a sequence entry points to a prior closure output, replay that output by the same rows $T, M, S, Q, L, U, W, R, A$ rather than introducing a new host closure coordinate.

The contraction is then componentwise. Transport through H identifies the topology, located-source, sequence, handoff, and Real-seal rows, while C, P, N repack the same witness as a single $\mathsf{SequentialClosure}^{\uparrow}$ certificate. Since every step is a finite replay of displayed windows, idempotence holds only inside this BEDC carrier surface. □

Theorem 1571.11 (SequentialClosure closed-set fixpoint)**.** *Let $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet, and suppose the* $\mathsf{LocatedSet}^{\uparrow}$ *source row S carries a displayed sequential closedness witness: every* $\mathsf{StreamName}^{\uparrow}$ *sequence over S with* $\mathsf{RegSeqRat}^{\uparrow}$ *tail data and* $\mathsf{Real}^{\uparrow}$ *equality seal has its candidate limit admitted back through S. Then the closure output named by K is fixed by S under the same located-set transport. This is a NameCert fixpoint for the displayed closedness witness, not a choice-based closed-set theorem.*

Proof. Start with the located source row S. The closedness witness is read as finite data over S, not as a detached predicate on an ambient topology.

Apply the sequence-limit handoff of Definition 1571.3. The closure read presents exactly the sequence source, StreamName windows, RegSeqRat handoff, and Real seal required by the displayed closedness witness.

Now replay the witness back into S. LocatedSet transport preserves the admitted membership row, and componentwise `hsame` transport by H keeps the topology and metric rows fixed. Therefore the closure output is repacked as an S-admitted point by C, P, N, with no appeal to countable choice or host closedness. □

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureClosedSetFixpoint`

Theorem 1571.12 (SequentialClosure topology sequence-filter handoff)**.** *For an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$, every topology-facing neighbourhood request around L that is also read as a finite sequence-filter request factors through the same route*

$$T, S, Q, U, W, R, A, H, C, P, N.$$

The Topology$^{\uparrow}$ *request supplies the neighbourhood row,* StreamName$^{\uparrow}$ *supplies the inspected tail windows,* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *seal the metric comparison, and* LocatedSet$^{\uparrow}$ *transport keeps the source membership boundary visible.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureTopologySequenceFilterHandoff`

Proof. Begin with a neighbourhood request from the topology row T. The topology handoff of Definition 1571.6 forces the request through the displayed source, sequence, test, and window ledgers rather than through a host closure operator.

Read the same request as a sequence-filter witness by naming the finite StreamName windows in W. Each tail inspection remains tied to the source row S, so located-set membership is transported before the metric side is sealed.

Finally consume R and A. The RegSeqRat row records the rational tail budget and the Real row seals the comparison at the candidate limit. The structural rows H, C, P, N then package the topology and sequence-filter views as the same finite handoff. □

Theorem 1571.13 (SequentialClosure obligation closure surface)**.** *For every accepted* SequentialClosure$^{\uparrow}$ *carrier* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$*, the obligation surface is the finite list consisting of carrier admission, neighbourhood-test stability, sequence-window coverage, limit-row handoff, componentwise transport, continuation replay, package provenance, local naming, and nonescape from host closure, selected subsequences, quotient stream equality, open-cover compactness, and ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureObligationClosureSurface`

Proof. Project the carrier and apply Definition 1571.2. That theorem lists the local NameCert obligations over exactly the rows $T, M, S, Q, L, U, W, R, A, H, C, P, N$.

Use Definition 1571.7 to bind the carrier admission side to the displayed source, metric, sequence, RegSeqRat, Real, transport, replay, provenance, and local naming rows. Use Definition 1571.8 to bind the limit side to U, W, R, A, H, C, P, N.

The remaining exclusions are read from the same absence-of-coordinate argument used in Definition 1571.3. Since no host closure, selected subsequence, quotient stream equality, compactness, or ambient completeness row appears in the packet, the displayed finite list exhausts the obligation surface. □

Theorem 1571.14 (SequentialClosure neighbourhood-test stability)**.** *Let* $K = (T, M, S, Q, L, U, W, R, A, H, C, P,$ *be an accepted* SequentialClosure$^{\uparrow}$ *packet. Every neighbourhood request around the candidate limit row* L *is stable under componentwise* hsame *transport only when the transported packet preserves the same topology row* T*, metric row* M*, source row* S*, sequence row* Q*, test ledger* U*, and downstream window-handoff rows* W, R, A*.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureNeighbourhoodTestStability`

Proof. Start with a neighbourhood request around L. By Definition 1571.6, the request is answered through U over the common topology row T, with metric-facing reads continuing through M, R, A.

Transport is controlled by H. The located-set stability theorem Definition 1571.5 keeps source membership and admitted sequence entries over the same S row, and Definition 1571.9 preserves the same sequence-limit rows under componentwise hsame and displayed replay.

Thus a transported neighbourhood test remains stable exactly when the packet preserves T, M, S, Q, U, W, R, A and repackages them through H, C, P, N. No separate neighbourhood oracle or host closure coordinate is available. □

Theorem 1571.15 (SequentialClosure sequence-window coverage)**.** *For an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$*, every closure read of* L *exposes the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff or* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The coverage obligation is only this displayed finite window coverage over* Q *and* U*.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureSequenceWindowCoverage`

Proof. Read the closure consumer through Definition 1571.3. The consumer factors from T, M, S, Q, L to U, W before reaching R, A and the structural rows.

Apply Definition 1571.7. It makes the sequence source row Q available only with the source row S, metric row M, test ledger U, and finite window ledger W in the same carrier.

Finally use Definition 1571.12. The topology-facing sequence-filter read names the same finite windows and then passes through R, A, H, C, P, N. Hence every accepted closure read covers the inspected windows explicitly, without selected subsequences or quotient stream equality. □

Theorem 1571.16 (SequentialClosure limit-row handoff)**.** *Let* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$ *be an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *packet. The candidate limit row* L *may be exported to a closure consumer only through the route*

$$T, M, S, Q, L \longmapsto U, W \longmapsto R, A \longmapsto H, C, P, N.$$

In particular, the limit row is handed off through the displayed neighbourhood tests, finite sequence windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal, not through host topological closure or ambient completeness.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureLimitRowHandoff`

Proof. Begin with the candidate limit row. The sequence-limit handoff theorem Definition 1571.3 gives the displayed route from T, M, S, Q, L through U, W, R, A and then through H, C, P, N.

On the metric side, apply Definition 1571.4. It requires the finite window ledger W, the regular-rational handoff R, and the Real seal A before a distance conclusion is exported.

The root limit ledger theorem Definition 1571.8 excludes selected subsequences, quotient stream equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness as limit-witness sources. Therefore the only available handoff for L is the displayed finite route. □

Theorem 1571.17 (SequentialClosure kernel scope surface)**.** *For an accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *carrier* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$*, the scoped BEDC kernel surface is exactly the displayed packet over* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{LocatedSet}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *rows. Every topology or metric closure read must compose the existing sequence-limit handoff, neighbourhood-test stability, sequence-window coverage, and limit-row handoff rows before it can be presented to a downstream consumer.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureKernelScopeSurface`

Proof. Project the carrier. The rows $T, M, S, Q, L, U, W, R, A, H, C, P, N$ name the topology, metric, located source, displayed sequence, candidate limit, test ledger, finite windows, rational readback, Real seal, transport, replay, provenance, and local NameCert coordinates.

Compose the already displayed handoffs in dependency order. The sequence-limit handoff of Definition 1571.3 keeps every closure read on $T, M, S, Q, L, U, W, R, A, H, C, P, N$, while Definition 1571.14 fixes the topology and metric request rows.

The final scope is supplied by Definition 1571.15 and Definition 1571.16. Finite StreamName windows precede the RegSeqRat readback, and the Real seal follows that readback. Thus the kernel surface is scoped to the listed BEDC rows rather than to host closure or detached completeness. □

Theorem 1571.18 (SequentialClosure carrier admission scope)**.** *Carrier admission for* $\mathsf{SequentialClosure}^{\uparrow}$ *admits a closure packet only when the same finite* BHist *row displays the* $\mathsf{Topology}^{\uparrow}$ *source* T*,* $\mathsf{Met}^{\uparrow}$ *source* M*,* $\mathsf{LocatedSet}^{\uparrow}$ *or subset row* S*,* $\mathsf{StreamName}^{\uparrow}$ *sequence row* Q *and windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* A*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row* N*. Admission does not create a second carrier for an ambient closure operator or for a detached limit theorem.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureCarrierAdmissionScope`

Proof. Start from Definition 1571.1. Its displayed tuple contains one topology row, one metric row, one source row, one sequence source, one candidate limit, one test ledger, one finite window ledger, one RegSeqRat handoff, one Real seal, and the structural rows.

Use Definition 1571.7 for the carrier side. It binds the sequence row to the LocatedSet or subset source, the metric row, StreamName windows, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming rows in the same packet.

Use Definition 1571.8 for the limit side. The candidate limit row is admitted only through U, W, R, A, H, C, P, N, so the carrier has no place to store host closure, arbitrary selection, quotient stream equality, or an independent completeness proof. □

Theorem 1571.19 (SequentialClosure nonescape scope)**.** *For every accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *carrier* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$*, a downstream* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{LocatedSet}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer can read closure evidence only through* $T, M, S, Q, L, U, W, R, A, H, C$ *and the local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row. The packet exports no host topological closure, selected subsequence, quotient stream equality, open-cover compactness, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or additional bridge row.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosureNonescapeScope`

Proof. Read the consumer route through Definition 1571.17. Topology and metric requests are scoped to T, M, U, located-source requests to S, Q, sequence windows to W, and metric limit reads to R, A.

Apply Definition 1571.18. Since all accepted carrier data lives in the same BHist tuple and is transported only by H and replayed only by C, a consumer cannot introduce a second closure source while preserving the local NameCert.

The exclusions follow from Definition 1571.2 and Definition 1571.3. The absent claims have no carrier coordinate, no continuation edge, no provenance entry, and no NameCert row, so they cannot escape through the scoped kernel packet. □

Theorem 1571.20 (SequentialClosure public export)**.** *For every accepted* $\mathsf{SequentialClosure}^{\uparrow}$ *carrier* $K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$*, any downstream topology or metric consumer receives closure evidence only through the displayed source row* S*, sequence windows* W*, RegSeqRat handoff* R*, Real seal* A*, componentwise transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$ *row* N*. The public export packages the existing scoped route; it does not add a host closure operator, selected subsequence, quotient stream equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a detached bridge row.*

Lean4 ✓ `BEDC.Derived.SequentialClosureUp.SequentialClosurePublicExport`

Proof. Begin with Definition 1571.2. It identifies the carrier rows and requires every closure-facing read to expose the source, sequence, test, handoff, Real seal, transport, replay, provenance, and local naming data.

Apply Definition 1571.3 and Definition 1571.4. These theorems put the sequence windows before the regular-rational readback and put the Real seal after that readback.

Now use the public topology surface already recorded by Definition 1571.6, Definition 1571.17, and Definition 1571.9. They keep topology and metric consumers on the same displayed rows $T, M, S, Q, L, U, W, R, A, H, C, P, N$.

Finally apply Definition 1571.18 and Definition 1571.19. The carrier admission theorem repacks the rows as one finite packet, and the nonescape theorem excludes every host closure or selection object absent from that packet. □

Theorem 1571.21 (SequentialClosure mature treatment)**.** *The complete paper surface for* $\mathsf{SequentialClosure}^{\uparrow}$ *is the finite carrier*

$$K = (T, M, S, Q, L, U, W, R, A, H, C, P, N)$$

together with the public carrier admission route, the local NameCert obligations, the sequence-limit handoff, the $\mathsf{RegSeqRat}^{\uparrow}$ *limit route,* $\mathsf{LocatedSet}^{\uparrow}$ *stability, the topology handoff, the root sequence and limit ledgers, root topology transport, Kuratowski idempotence, the closed-set fixpoint, kernel scope, carrier-admission scope, nonescape, and public export. A downstream closure consumer that cites* $\mathsf{SequentialClosure}^{\uparrow}$ *must stay on this displayed surface and receives no host topological closure, arbitrary sequence selection, general sequential compactness, quotient stream equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or detached bridge row.*

Proof. Start with the carrier of Definition 1571.1 and the NameCert obligations of Definition 1571.2. They expose the topology, metric, located-source, sequence, candidate-limit, test, window, RegSeqRat, Real, transport, replay, provenance, and naming rows as one finite packet.

The limit-facing route is already fixed. Apply Definition 1571.3, Definition 1571.4, Definition 1571.15, and Definition 1571.16 to force every closure read through finite StreamName windows before the RegSeqRat handoff and Real seal are consumed.

The source and topology routes are also fixed. Use Definition 1571.5, Definition 1571.6, Definition 1571.7, Definition 1571.8, and Definition 1571.9 to keep located-set membership, topology requests, and root ledgers on the same carrier rows.

Finally close the paper surface by Definition 1571.10, Definition 1571.11, Definition 1571.17, Definition 1571.18, Definition 1571.19, and Definition 1571.20. These theorems repack idempotence, displayed closedness, public consumer exposure, and nonescape as consequences of the same finite packet, so no excluded host closure or completeness object remains part of the paper claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1571.1, Definition 1571.2, Definition 1571.3, Definition 1571.13, Definition 1571.14, Definition 1571.15, Definition 1571.16, Definition 1571.17, Definition 1571.18, Definition 1571.19, Definition 1571.20, and Definition 1571.21 enumerate the finite obligation surface, public consumer nonescape route, and displayed sequence-limit, RegSeqRat, topology, root-ledger, Kuratowski, and closed-set fixpoint handoffs for metric closure consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topological closure, arbitrary sequence selection, general sequential compactness, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal closure requires checked carrier admission, classifier transport, nonescape, and bridge data for the displayed sequence-limit, RegSeqRat, and topology handoff.

Constructive story: A SequentialClosure$^{\uparrow}$ packet is generated from ambient TopologyUp and MetricUp rows, a subset or LocatedSet row, a sequence source, a candidate limit row, a neighbourhood and metric test ledger, finite StreamName windows, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1572

A Concrete Naming Certificate for $\mathsf{CofinalCauchyTail}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy convergence, the $\mathsf{CofinalCauchyTail}^{\uparrow}$ packet records the finite route by which a Cauchy source is read only through certified cofinal tail thresholds before it reaches a regular rational handoff and a terminal Real seal. It sits between the cofinal-subsequence discipline of chapter 405, the Cauchy criterion surface of chapter 392, the fast Cauchy route of chapter 323, the regular rational readback of chapter 301, and the Real naming boundary of chapter 38. The packet is a finite NameCert carrier for tail-normalized Cauchy access; it is not a quotient stream, not a selected limit, not host real equality, not countable choice, not trichotomy, and not a completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 1572.1 (Cofinal Cauchy tail carrier)**.** *A* $\mathsf{CofinalCauchyTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, \tau, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy row or* $\mathsf{StreamName}^{\uparrow}$ *row whose late observations are to be read through the tail packet.*
2. *τ is the cofinal tail-threshold ledger, recording the finite threshold cells that certify that later requests stay inside the accepted tail route.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that supplies the precision-to-threshold budget consumed by τ.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested dyadic radius against the tail observations named by S.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, τ, M, D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after the RegSeqRat handoff row has been replayed.*
7. *H records componentwise* hsame *transport for the source, threshold ledger, modulus, dyadic tolerance, RegSeqRat handoff, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through the cofinal threshold ledger, Cauchy modulus, dyadic tolerance ledger, RegSeqRat handoff, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchyTail}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, countable-choice selectors, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1572.2 (Cofinal Cauchy tail NameCert obligations)**.** *For every accepted* $\mathsf{CofinalCauchyTail}^{\uparrow}$ *carrier $T = (S, \tau, M, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalCauchyTail}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the source Cauchy or* $\mathsf{StreamName}^{\uparrow}$ *row, cofinal threshold ledger, Cauchy modulus, dyadic tolerance ledger, RegSeqRat handoff, Real seal, transport, continuation, provenance, and local naming rows;*
2. *cofinal-tail coverage, requiring each accepted precision request to read the threshold ledger τ together with the modulus row M before any late observation of S is consumed;*
3. *dyadic tolerance exactness, requiring the comparison row D to be the only tolerance row used for the accepted tail observations;*
4. *RegSeqRat handoff, requiring S, τ, M, D to repack as the regular rational readback row R before any Real-facing row is available;*
5. *Real seal refusal, excluding quotient streams, selected limits, host real equality, countable-choice selectors, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1572.1. The projection exposes the source row S, threshold ledger τ, modulus row M, dyadic tolerance row D, RegSeqRat handoff R, Real seal E, and structural rows H, C, P, N.

The continuation row C records the only replay route through the packet. Its first dependencies are the threshold ledger τ and the modulus row M; after those rows are consumed, the same route reads the late source observations in S and the tolerance comparisons in D.

The row R occurs downstream of S, τ, M, D, and E occurs downstream of R. Since the carrier contains no coordinates for quotient streams, selected limits, host real equality, countable-choice selectors, trichotomy, or completeness, the displayed five rows exhaust the local NameCert obligation surface. □

Theorem 1572.3 (Cofinal Cauchy tail RegSeqRat handoff)**.** *Let $T = (S, \tau, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CofinalCauchyTail}^{\uparrow}$ *carrier. Every accepted tail packet reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the rows S, τ, M, D, R: the source row S supplies the late observations, τ supplies the cofinal threshold ledger, M supplies the Cauchy modulus budget, D supplies the dyadic tolerance comparisons, and R is the resulting regular rational readback. No separate regular-sequence oracle is available.*

Proof. Begin with an accepted tail packet and project its carrier. The only rows before the RegSeqRat handoff are S, τ, M, D together with the structural transport and replay rows.

Replay C at the requested precision. The route first reads τ and M, then uses those rows to select the late observations of S and the dyadic comparisons in D.

The handoff row R is listed only after that route. Transport by H preserves the same row order, while P and N name the same finite route. Hence every accepted packet reaches $\mathsf{RegSeqRat}^{\uparrow}$ through S, τ, M, D, R and through no independent regular-sequence oracle. □

Theorem 1572.4 (Cofinal Cauchy tail Real seal boundary)**.** *Let $T = (S, \tau, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CofinalCauchyTail}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read passes through the RegSeqRat handoff row R before the seal row E is read. The exported surface contains no quotient stream, selected limit, host real equality, countable-choice selector, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Take a Real-facing consumer of the packet. Carrier admission exposes the seal row E, but only as a row downstream of the RegSeqRat handoff row.

Replay the continuation row C. The consumer must first pass through the source row S, threshold ledger τ, Cauchy modulus M, and dyadic tolerance row D; those rows then repack as R.

Only after R has been replayed can E be read. Because the carrier has no coordinate for quotient streams, selected limits, host real equality, countable-choice selectors, trichotomy, or completeness, the Real-facing surface exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalCauchyTail}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalCauchyTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1572.1, Definition 1572.2, Definition 1572.3, and Definition 1572.4 seed the finite route from cofinal Cauchy tail thresholds through CauchyModulus, DyadicRatCore, RegSeqRat, and the downstream Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalCauchyTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable-choice selectors, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-tail coverage, dyadic tolerance exactness, RegSeqRat handoff, Real seal boundary, and ledger refusal proofs for the same packet.

Constructive story: A $\mathsf{CofinalCauchyTail}^{\uparrow}$ packet is generated from a source Cauchy or StreamName row, a cofinal tail-threshold ledger, a CauchyModulus row, a DyadicRatCore tolerance row, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1573

A Concrete Naming Certificate for $\mathsf{TotallyBoundedCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ packet records the finite route by which a totally bounded metric carrier passes through constructive completion while retaining explicit net data. It reads the finite-net surface of chapter 127, the precompact bridge of chapter 1531, the basis generator of chapter 1501, the dense embedding row of chapter 1512, the constructive completion route of chapter 1556, and the uniformly continuous handoff of chapter 769. The packet is a finite completion-facing certificate for displayed total-bounded net ledgers; it is not an open-cover compactness theorem, a quotient completion, a selected limit-point operation, a countable-choice principle, or a full completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1573.1 (Totally bounded completion carrier)**.** *A* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, T, N, F, E, C, S, U, H, P, L)$$

with the following visible rows:

1. *X is the source metric carrier row, including the accepted point histories, distance-witness rows, and classifier transports read by the total-bounded and completion-facing coordinates.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger. For each displayed positive tolerance it records the closed* $\mathsf{ProbeBundle}$ *of centers,* $\mathsf{InBundle}$ *membership rows, center-carrier rows, distance witnesses, and rational bound ledgers.*
3. *N is the net-refinement row. It records how a finer tolerance request refines a previously displayed finite net without replacing the net by an open cover or by a host finite set.*
4. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row generated from the same metric and net data, including the finite basis packets and common-refinement reads consumed by the completion route.*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that sends source observations into the dense completion boundary while preserving the displayed metric and net ledgers.*
6. *C is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route through regular Cauchy names, dyadic tolerances, Cauchy-filter basis data, dense embedding, separated comparison, and uniformly continuous extension.*
7. *S is the separated completion equality row. It compares completion reads only through the displayed Cauchy-filter and zero-distance route.*

8. *U is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff row for consumers that extend uniformly continuous maps from the dense source through the completed packet.*

9. *H records componentwise* hsame *transport for the source metric, finite nets, net refinements, Cauchy-filter basis, completion embedding, constructive completion route, separated equality, extension handoff, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{PrecompactMetric}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *row naming the displayed packet and the permitted reads from it.*

The classifier on such packets is componentwise transport through H together with replay through the displayed route rows. No coordinate exports open-cover compactness, quotient representatives, selected limit points, countable choice, an ambient complete metric space, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TasteGate.TotallyBoundedCompletionTasteGate single carrier alignment`

Theorem 1573.2 (TotallyBoundedCompletion NameCert obligations)**.** *For every accepted* TotallyBoundedComplet *carrier $K = (X, T, N, F, E, C, S, U, H, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed source metric, total-bounded net ledger, net-refinement row, Cauchy-filter basis, dense completion embedding, constructive completion route, separated equality, uniformly continuous handoff, transport ledger, provenance, and local naming row;*

2. *finite-net persistence, requiring every completion-facing tolerance request to read the centers, membership constructors, distance witnesses, and rational bounds recorded in T and refined through N;*

3. *basis and embedding replay, requiring each completion read to consume F and E before it reaches the constructive completion route C;*

4. *separated and extension handoff, requiring equality of completion reads to pass through S and every uniformly continuous consumer to pass through U after F, E, C have been displayed;*

5. *ledger non-escape, excluding open-cover compactness, quotient completion, selected limit points, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, and ambient completeness.*

No NameCert obligation is stored outside these displayed rows.

Proof. Carrier admission is projection from Definition 1573.1. The packet lists the metric source, finite-net ledger, net-refinement row, Cauchy-filter basis, dense embedding, constructive completion route, separated equality, extension handoff, transport ledger, provenance, and local naming row.

Finite-net persistence is read from T and N. A tolerance request first inspects the closed $\mathsf{ProbeBundle}$, its $\mathsf{InBundle}$ constructors, the center-carrier rows, and the metric distance witnesses in T; refinement then follows the displayed row N and keeps the same ledger style at the finer tolerance.

Basis and embedding replay is forced by the route order. The Cauchy-filter basis row F consumes the finite net data before a completion-facing read is formed, and the embedding row E

places source observations inside the dense completion boundary before the constructive route C can be read.

The separated and extension obligations are then local to S and U. Equality of completion reads is compared through S, while uniformly continuous consumers can inspect the completed packet only through U after the basis, embedding, and constructive completion rows are already present.

Finally, the provenance row P and local naming row L name only the displayed coordinates. Since none of those coordinates contains open-cover compactness, quotient representatives, selected limit points, countable choice, detached Real completeness, or ambient completeness, no such datum can be exported by the NameCert surface. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier namecert obligations`

Theorem 1573.3 (TotallyBoundedCompletion finite-net Cauchy handoff)**.** *For every completion-facing read from an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, the consumer must expose the finite* ProbeBundle *centers and membership rows of* T*, the refinement ledger* N*, the* CauchyFilterBasis$^{\uparrow}$ *row* F*, the* CompletionEmbedding$^{\uparrow}$ *row* E*, and the* ConstructiveMetricCompletion$^{\uparrow}$ *route* C *before any separated comparison in* S *or uniformly continuous handoff in* U *is read. Componentwise* hsame *transport through* H *and displayed* Cont *replay preserve this order; they do not replace the finite-net-to-Cauchy-filter route.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionFiniteNetCauchyHandoff`

Proof. Start with the NameCert obligations of Definition 1573.2. Carrier admission displays the rows $X, T, N, F, E, C, S, U, H, P, L$ and finite-net persistence requires each tolerance request to read the center and membership data recorded in T.

The refinement row N is then the only permitted route from a displayed finite net to a finer tolerance read. It keeps the same ProbeBundle centers, membership constructors, distance witnesses, and rational bound ledgers visible before a Cauchy-filter basis is generated.

Apply the basis and embedding replay clause. The row F consumes the finite-net and refinement data, the row E embeds the resulting source observations, and the route C records the constructive completion read.

Finally use the separated and extension handoff clause. The rows S and U are downstream of F, E, C, while H, P, L only transport, source, and name the same packet. Thus no completion-facing consumer can bypass the finite-net-to-Cauchy-filter handoff. □

Theorem 1573.4 (TotallyBoundedCompletion scope dependency seal)**.** *Every accepted scoped consumer of* TotallyBoundedCompletion$^{\uparrow}$ *must expose the full route*

$$X, T, N, F, E, C, S, U, H, P, L$$

with TotallyBounded$^{\uparrow}$ *read first through the finite-net ledger* T*,* CauchyFilterBasis$^{\uparrow}$ *read next through* F*,* CompletionEmbedding$^{\uparrow}$ *read through* E*,* ConstructiveMetricCompletion$^{\uparrow}$ *read through* C*, and* UniformContinuousExtension$^{\uparrow}$ *read through* U*. The route is sealed only when the carrier definition, NameCert obligation surface, finite-net Cauchy handoff, and net-refinement stability all remain visible. No scoped read may replace this route by open-cover compactness, selected limits, quotient completion, countable choice, or detached Real completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier scope dependency seal`

Proof. Project the accepted carrier of Definition 1573.1. The rows $X, T, N, F, E, C, S, U, H, P, L$ are the complete packet inventory, so a scoped consumer cannot name a completion-facing read without exposing the finite-net, basis, embedding, constructive-completion, extension, transport, provenance, and local NameCert rows.

Apply the obligation surface of Definition 1573.2. It forces carrier admission, finite-net persistence, basis-and-embedding replay, separated and extension handoff, and ledger non-escape on the same tuple.

Use the finite-net Cauchy handoff of Definition 1573.3. That theorem places T, N, F, E, C before any separated comparison in S or uniformly continuous handoff in U can be read.

Finally apply net-refinement stability from Definition 1573.9. Refinement preserves the displayed finite-net route through T, N, F, E, C rather than replacing it by compactness, a selected limit, a quotient-completion representative, countable choice, or detached Real completeness. Thus the scoped dependency route is sealed exactly by the displayed L10 rows. □

Definition 1573.5 (TotallyBoundedCompletion carrier admission obligation)**.** *The carrier admission obligation for* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *is the requirement that every accepted* BHist *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$ *display all eleven rows before its local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *row is read. The source metric row* X *feeds the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger* T*; the refinement row* N *feeds the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row* F*; the dense* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* E *and* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route* C *precede the separated equality row* S *and the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff* U*; and* H, P, L *record componentwise* hsame *transport, package provenance, and local naming. The displayed* Cont *replay may follow only this row order.*

Theorem 1573.6 (TotallyBoundedCompletion refinement ledger obligation)**.** *For every accepted* BHist *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$ *satisfying Definition 1573.5, every refinement request admitted by* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *must consume the* $\mathsf{TotallyBounded}^{\uparrow}$ *ledger* T*, the net-refinement row* N*, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row* F*, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* E*, and the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route* C *before it is transported by* H *or replayed by* Cont*. The request cannot bypass* T, N, F, E, C *by reading only* S, U, H, P, L*.*

Proof. Project the refinement request from the displayed packet. Carrier admission supplies the rows $X, T, N, F, E, C, S, U, H, P, L$ and names the permitted dependency order through the local $\mathsf{NameCert}$ row.

The refinement request is indexed by a tolerance read, so it first inspects the finite centers, membership rows, distance witnesses, and rational bounds in the $\mathsf{TotallyBounded}^{\uparrow}$ ledger T. The row N is then the only permitted refinement coordinate, and the generated basis row F records the corresponding $\mathsf{CauchyFilterBasis}^{\uparrow}$ packet.

The completion-facing part of the request must pass through E and C before separated comparison or extension handoff is visible. Componentwise hsame transport through H and displayed Cont replay preserve this route; they do not create a shortcut around the finite-net, basis, embedding, and constructive-completion rows. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier refinement ledger obligation`

Theorem 1573.7 (TotallyBoundedCompletion separated extension obligation)**.** *For every accepted* BHist *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$ *admitted by* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$*, a separated equality read in* S *or a uniformly continuous extension read in* U *is valid only after the source row* X*, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger* T*, the refinement row* N*, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$

row F, the CompletionEmbedding$^{\uparrow}$ *row E, and the* ConstructiveMetricCompletion$^{\uparrow}$ *route C have been displayed. Componentwise* hsame *transport through H and displayed* Cont *replay may transport the separated or extension read, but they may not replace the dependency on T, N, F, E, C.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier separated extension obligation`

Proof. Read the separated equality coordinate S and the extension handoff coordinate U from the carrier. Both coordinates are downstream rows of the same BHist packet $K = (X, T, N, F, E, C, S, U, H, P, L)$, so their admission is governed by the local NameCert surface rather than by an external completion theorem.

The source, finite-net, refinement, basis, embedding, and constructive-completion rows are already displayed before S or U can be consumed. Thus a separated equality comparison still depends on the Cauchy-filter and completion route, and a uniformly continuous extension consumer still depends on the finite-net route that produced the completed packet.

Transport and replay preserve that dependency. The hsame row H moves each coordinate componentwise, while the displayed Cont replay follows the same row order; neither operation supplies a new separated quotient, a selected limit, or an extension principle outside TotallyBounded$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, ConstructiveMetricCompletion$^{\uparrow}$, and UniformContinuousExtension$^{\uparrow}$. □

Theorem 1573.8 (TotallyBoundedCompletion net handoff)**.** *Let $K = (X, T, N, F, E, C, S, U, H, P, L)$ be an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet. Every completion-facing consumer that requests finite total-bounded data factors through*

$$X, T, N \longmapsto F \longmapsto E \longmapsto C \longmapsto S, U \longmapsto H, P, L.$$

Thus the consumer receives the original finite net ledger and its refinement row as part of the completion route, rather than an open cover, a selected convergent subnet, a quotient completion class, or an ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier net handoff`

Proof. Start with the source metric row X and the finite-net ledger T. These two rows provide the point classifier, distance witnesses, center names, membership constructors, and rational bounds that any total-bounded read may inspect.

The refinement row N is the only refinement operation available to the consumer. It moves from one displayed tolerance request to a finer one while retaining the same finite ledger shape, so the consumer cannot replace the packet by an open cover or by a host-level finite set.

The completion route then consumes this refined net data through F, E, and C. The Cauchy-filter basis is generated from the displayed rows, the embedding row places source observations densely in the completion boundary, and the constructive completion route records the finite regular-Cauchy and dyadic reads.

The separated equality row S and the uniformly continuous handoff U are downstream of that route. Transport, provenance, and local naming through H, P, L preserve the same dependency order, which proves the stated factorization and excludes the unavailable compactness, subnet, quotient, and ambient-completeness reads. □

Theorem 1573.9 (TotallyBoundedCompletion net-refinement stability)**.** *For every accepted* TotallyBoundedCompl *packet $K = (X, T, N, F, E, C, S, U, H, P, L)$, any net-refinement read transported through T, N, F, E, C*

remains a read of the same displayed finite-net ledger before it reaches separated comparison, uniformly continuous extension, transport, provenance, or local naming. In particular, refinement stability cannot be witnessed by S, U, H, P, L alone and cannot replace the finite net row T or refinement row N by an open-cover compactness witness.

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier net refinement stability`

Proof. Project the packet using Definition 1573.1. The only finite-net and refinement coordinates are T and N, while F, E, C are the downstream Cauchy-filter, embedding, and constructive-completion rows generated from them.

Apply the NameCert obligations of Definition 1573.2. Their finite-net persistence and basis-and-embedding replay clauses require every refinement read to display T, N, F, E, C before it is transported or replayed.

Finally use the route of Definition 1573.8. Since the displayed route sends X, T, N through F, E, C before S, U, H, P, L are available, refinement stability is carried by the same finite-net ledger and not by an external compactness or open-cover row. □

Theorem 1573.10 (TotallyBoundedCompletion Cauchy-filter-basis handoff)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, each* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *handoff exposed by F must first expose the total-bounded ledger T and the refinement row N before the dense embedding row E or constructive completion route C is read. No Cauchy-filter-basis consumer can enter the dense completion boundary while bypassing T, N.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier cauchy filter basis handoff`

Proof. Open the carrier of Definition 1573.1. The row F is not an independent basis source; it is the Cauchy-filter-basis row generated from the same metric and net data displayed in X, T, N.

The NameCert obligations of Definition 1573.2 require basis and embedding replay. Thus the handoff through F is admitted only after the finite-net ledger and refinement rows have already been exposed.

The net handoff theorem, Definition 1573.8, fixes the order $X, T, N \longmapsto F \longmapsto E \longmapsto C$. Therefore dense embedding reads in E and constructive completion reads in C cannot replace or skip the total-bounded ledger that generated the basis. □

Theorem 1573.11 (TotallyBoundedCompletion extension nonescape)**.** *For every accepted* TotallyBoundedComplet *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, a uniformly continuous extension read through U remains inside the displayed route*

$$X, T, N, F, E, C, S, U, H, P, L.$$

It cannot bypass the source metric, total-bounded ledger, refinement row, Cauchy-filter basis, embedding, constructive completion route, separated comparison, transport, provenance, or local naming rows.

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier extension nonescape`

Proof. Start from the carrier definition. The extension handoff row U is one coordinate of the same finite packet, and its permitted context includes the source, net, refinement, basis, embedding, completion, separated equality, transport, provenance, and naming rows.

Apply Definition 1573.2. The separated and extension handoff clause places U downstream from F, E, C, while finite-net persistence keeps F, E, C tied to T, N.

Use Definition 1573.8 to read the complete route. Since the extension handoff occurs after X, T, N, F, E, C, S, and H, P, L only transport, source, and name the same packet, the extension read cannot bypass any displayed row in the listed ledger. □

Theorem 1573.12 (TotallyBoundedCompletion uniform-extension ledger)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, every* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff through* U *factors through the finite-net ledger* T, N*, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row* F*, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* E*, the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route* C*, and the separated equality row* S *before* U, H, P, L *are public. The handoff exports a finite uniform-extension ledger for the same packet and no quotient completion, selected limit, open-cover compactness, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or second completion route.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier uniform extension ledger`

Proof. Start with the net handoff of Definition 1573.8. It forces every completion-facing consumer to read the total-bounded finite-net ledger T, refinement row N, Cauchy-filter basis row F, embedding row E, and constructive completion route C before separated comparison or extension handoff is available.

Use the Cauchy-filter-basis handoff. The row F is generated from the same finite-net data T, N, so an extension consumer cannot enter the dense completion boundary while bypassing the net ledger.

Apply the separated extension obligation. It places separated equality S and uniformly continuous extension U downstream of X, T, N, F, E, C, with transport through H and replay by Cont preserving that dependency.

Finally invoke extension nonescape. Since a U-read remains inside $X, T, N, F, E, C, S, U, H, P, L$, the public rows U, H, P, L export only the displayed uniform-extension ledger and no quotient completion, selected limit, open-cover compactness, countable-choice schedule, detached Real completeness, or second completion route. □

Theorem 1573.13 (TotallyBoundedCompletion CompactMetric scope)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet*

$$K = (X, T, N, F, E, C, S, U, H, P, L)$$

must expose the finite-net ledger T, N*, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row* F*, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* E*, the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route* C*, separated equality* S*, the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff* U*, componentwise* hsame *transport* H*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* L*. The scope is finite-net completion data only; it is not open-cover compactness, selected subnet convergence, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the net handoff theorem. It requires a completion-facing consumer to read the finite total-bounded ledger T, N before the Cauchy-filter basis row F, embedding row E, and constructive completion route C are available.

Use the Cauchy-filter-basis handoff and uniform-extension ledger next. The row F remains tied to T, N, while E, C, S, U are downstream rows of the same finite route and not independent compactness data.

Apply ledger non-escape. It excludes open-cover compactness, quotient completion, selected limit points, countable choice, detached Real completeness, and ambient completeness from the packet.

Finally read the structural rows H, P, L. Transport preserves the displayed finite-net completion route, provenance records the same packet source, and local naming admits only that route for a $\mathsf{CompactMetric}^{\uparrow}$-facing consumer. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier compact metric scope`

Theorem 1573.14 (TotallyBoundedCompletion net-completion image)**.** *Every public completion-facing image of an accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$ *factors through* T, N, F, E, C, S, U *before the structural rows* H, P, L *can be read. Consequently the image exports only finite-net completion data: total-bounded net ledgers, refinement, Cauchy-filter-basis handoff, completion embedding, constructive completion route, separated comparison, and uniform-extension handoff. It does not export open-cover compactness, selected subnet convergence, quotient completion, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completeness.*

Proof. Start with the net handoff theorem, Definition 1573.8. It places every completion-facing read behind the finite-net ledger T, refinement row N, Cauchy-filter basis F, completion embedding E, and constructive completion route C.

Use the Cauchy-filter-basis handoff and uniform-extension ledger, Definition 1573.10 and Definition 1573.12. These keep F, E, C, S, U attached to the same finite-net data before a public completion consumer can receive an image.

Apply the CompactMetric scope theorem above. A $\mathsf{CompactMetric}^{\uparrow}$-facing read sees the finite-net completion scope only through $T, N, F, E, C, S, U, H, P, L$, so H, P, L are structural transport, provenance, and naming rows rather than image-forming data.

Finally invoke ledger non-escape, Definition 1573.20. Since the excluded open-cover, selected-subnet, quotient, choice, detached-Real, and ambient-completeness objects have no carrier row, the public image is exactly the finite-net completion data exported by T, N, F, E, C, S, U. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier net completion image`

Theorem 1573.15 (TotallyBoundedCompletion root finite-net ledger)**.** *For every accepted* TotallyBoundedComplet *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, the root finite-net ledger exposed to completion consumers is exactly the pair* T, N*. The row* T *records the* $\mathsf{TotallyBounded}^{\uparrow}$ *centers, membership constructors, distance witnesses, and rational bounds for each displayed positive tolerance, while* N *records permitted refinement between such displayed ledgers. No root consumer may replace* T, N *by an open cover, a host finite subset, a selected subnet, or a compactness theorem.*

Proof. Project the carrier through Definition 1573.1. The only rows typed as total-bounded finite-net data are T and N.

Use the route surface of Definition 1573.24. Its first completion-facing segment is X, T, N, so every later completion row must inherit the finite-net ledger from the same pair.

The non-escape theorem then excludes replacement data. Since open covers, host finite subsets, selected subnets, and compactness witnesses have no carrier coordinate, the root ledger remains exactly T, N. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier root net ledger certificate`

Theorem 1573.16 (TotallyBoundedCompletion root completion handoff)**.** *For every accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, every completion-facing handoff forced by the root ledger factors through* F, E, C, S *in that order after* T, N *have been displayed. The* CauchyFilterBasis$^{\uparrow}$ *row* F *consumes the finite-net rows, the* CompletionEmbedding$^{\uparrow}$ *row* E *inserts source observations, the* ConstructiveMetricCompletion$^{\uparrow}$ *route* C *records the constructive completion read, and the separated comparison row* S *is available only after that route.*

Proof. Start with the root finite-net ledger theorem above. It identifies T, N as the input that any completion consumer must read.

Apply Definition 1573.27. The only completion-facing rows are F, E, C, and the separated comparison row is downstream of those rows.

Finally use Definition 1573.25. It makes the route exhaustive, so the handoff from the finite-net ledger to separated completion comparison must pass through F, E, C, S in the displayed order. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier root completion handoff`

Definition 1573.17 (TotallyBoundedCompletion effective net completion selector)**.** *For an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, the effective net completion selector is the finite BEDC read that first opens the total-bounded ledger pair* T, N *and only then follows*

$$F, E, C, S, U.$$

It returns the completion-facing route selected from the displayed finite net and its refinement ledger: F *is the generated* CauchyFilterBasis$^{\uparrow}$ *row,* E *is the dense* CompletionEmbedding$^{\uparrow}$ *row,* C *is the* ConstructiveMetricCompletion$^{\uparrow}$ *route,* S *is the separated comparison row, and* U *is the uniformly continuous extension handoff. The selector has no coordinate for open-cover compactness, quotient completion, selected subnet limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier effective net completion selector`

Theorem 1573.18 (TotallyBoundedCompletion selector consumption)**.** *Every use of the effective selector of Definition 1573.17 is consumed by the root completion handoff and the root extension nonescape rows. In particular, any selected completion route reads* T, N *before* F, E, C, S *and reaches* U, H, P, L *only as downstream extension, transport, provenance, and naming data.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier effective net completion consumption`

Proof. Start with the selector definition. Its first action is to open T, N, so the selected route is already tied to the root finite-net ledger rather than to an open cover or host finite subset.

Apply Definition 1573.16. The selected completion segment must pass through F, E, C, S in order after the finite-net rows are displayed.

Finally use Definition 1573.19. The rows U, H, P, L can only consume that displayed route; they do not add quotient completion, selected limits, countable choice, open-cover compactness, ambient completeness, or a second completion route. □

Theorem 1573.19 (TotallyBoundedCompletion root extension nonescape)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, the downstream rows* U, H, P, L *export only the uniformly continuous extension handoff, componentwise transport, provenance, and local naming for the already displayed route. They export no quotient completion, selected limits, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or second completion route.*

Proof. Read U, H, P, L from the carrier. These rows occur after X, T, N, F, E, C, S in the displayed route and therefore have no independent source of completion data.

Use the extension consumer boundary Definition 1573.28. It requires uniformly continuous consumers to enter through T, N, F, E, C before they can read U.

The ledger non-escape theorem Definition 1573.20 supplies the final exclusion. Quotient completion, selected limits, countable choice, open-cover compactness, ambient Real completeness, and a second completion route have no coordinate in U, H, P, L and cannot be exported from them. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier root extension nonescape`

Theorem 1573.20 (TotallyBoundedCompletion ledger non-escape)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, every exported completion read remains inside the finite ledger*

$$X, T, N, F, E, C, S, U, H, P, L.$$

It cannot produce open-cover compactness, a quotient completion, a selected limit point, countable choice, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier no escape`

Proof. Project an exported completion read from the carrier. By Definition 1573.1, the read has access only to the eleven displayed coordinates and their componentwise transport rows.

Apply Definition 1573.2. Finite-net persistence keeps the read inside T, N; basis and embedding replay moves it through F, E, C; separated and extension handoff moves it through S, U; and the structural rows H, P, L record only the same route.

Every excluded principle is absent from this route. Open-cover compactness, quotient representatives, selected limit points, countable-choice schedules, and full $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, no replay edge, and no provenance row in the packet, so the exported read cannot escape the displayed finite ledger. □

Theorem 1573.21 (TotallyBoundedCompletion finite-net completion obligations)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, a completion-facing read is admitted only after it displays the finite-net rows of* $\mathsf{TotallyBounded}^{\uparrow}$*, the generated* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row, the dense* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route, and the downstream* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff. The carrier has no completion obligation outside those rows.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier finite net completion obligations`

Proof. Project the read from Definition 1573.1. The finite-net rows are exactly T and their refinement requests are exactly N.

Next apply Definition 1573.8. The displayed route sends T, N through the Cauchy-filter basis F, the completion embedding E, and the constructive completion route C before separated comparison or extension reads are available.

Finally use Definition 1573.2. Its separated and extension handoff clause forces S and U to remain downstream of F, E, C, while H, P, L only transport, source, and name the same finite route. □

Theorem 1573.22 (TotallyBoundedCompletion refinement stability)**.** *Let $K = (X, T, N, F, E, C, S, U, H, P, L)$ and $K' = (X', T', N', F', E', C', S', U', H', P', L')$ be accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packets. If their displayed source metric rows, finite-net ledgers, Cauchy-filter basis rows, embedding rows, completion routes, extension rows, provenance rows, and local naming rows are related componentwise by* hsame*, and the displayed* Cont *replay sends each tolerance request in K to the corresponding request in K', then every refinement request in K' remains on the same finite-net and Cauchy-filter route as the corresponding request in K.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionRefinementStability`

Proof. Begin with the refinement coordinate. By carrier projection, a refinement request can inspect only the finite-net ledger T, the refinement row N, and their transported counterparts T', N'.

The componentwise hsame hypotheses identify the displayed source metric, centers, membership constructors, distance witnesses, and rational bounds. The Cont replay preserves the tolerance request, so the target request is not replaced by an open cover, a host finite set, or a selected subnet.

Apply Definition 1573.8 on both packets. The source refinement moves through F, E, C and the target refinement moves through F', E', C' in the same order, with S, U, H, P, L and S', U', H', P', L' only recording separated comparison, extension handoff, transport, provenance, and local naming. □

Theorem 1573.23 (TotallyBoundedCompletion extension handoff)**.** *Every* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *consumer of an accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet receives only the displayed route*

$$X, T, N, F, E, C, S, U, H, P, L.$$

In particular, the consumer cannot read a quotient completion, a selected limit, countable choice, an open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness in place of the finite completion route.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionExtensionHandoff`

Proof. The extension consumer first opens the carrier coordinate U. By Definition 1573.1, that row is downstream from the source metric, finite-net ledger, refinement row, Cauchy-filter basis, embedding, constructive completion route, and separated equality row.

Use Definition 1573.21. It requires the finite-net, Cauchy-filter, embedding, constructive completion, and extension rows to be displayed before the handoff can be read.

The non-escape conclusion follows from Definition 1573.20. The excluded objects have no carrier coordinate and no provenance row, so the uniformly continuous extension read is exactly the listed route. □

Definition 1573.24 (TotallyBoundedCompletion route surface)**.** *The* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *route surface of an accepted carrier* $K = (X, T, N, F, E, C, S, U, H, P, L)$ *is the displayed BEDC route*

$$X, T, N \longmapsto F \longmapsto E \longmapsto C \longmapsto S, U \longmapsto H, P, L.$$

It is admissible exactly when the source metric row X*, the total-bounded finite-net ledger* T*, and the net-refinement row* N *are consumed before the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row* F*;* F *is consumed before the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* E*;* E *is consumed before the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *row* C*;* C *is consumed before separated comparison* S *or uniformly continuous extension handoff* U*; and* H, P, L *only transport, source, and name that displayed route. No other row order is an admissible* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *route surface.*

Theorem 1573.25 (TotallyBoundedCompletion route coverage)**.** *Every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet displays exactly the route surface of Definition 1573.24. In particular, every accepted packet has the route*

$$X, T, N \longmapsto F \longmapsto E \longmapsto C \longmapsto S, U \longmapsto H, P, L$$

and has no second completion-facing route through open-cover compactness, quotient representatives, selected limit points, countable choice, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier route coverage`

Proof. Open the carrier of Definition 1573.1. Its visible coordinates are precisely $X, T, N, F, E, C, S, U, H, P, L$, and the classifier exports only componentwise transport through those coordinates.

The NameCert obligations of Definition 1573.2 make the same list exhaustive. Finite-net persistence reads T, N; basis and embedding replay reads F, E, C; separated and extension handoff reads S, U; and the remaining rows H, P, L transport, source, and name the same packet.

The route displayed in Definition 1573.8 is therefore not merely one possible consumer path. Together with the ledger non-escape statement of Definition 1573.20, it is the whole completion-facing surface, and the excluded completion principles have no carrier row from which a second route could be read. □

Theorem 1573.26 (TotallyBoundedCompletion total-bounded input admission)**.** *For every accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, the total-bounded finite-net rows* T, N *are consumed before any Cauchy-filter, embedding, constructive-completion, separated-equality, or uniformly continuous extension read is admitted. Thus no read of* F, E, C, S*, or* U *can supply the finite-net input that the route surface requires from* T, N*.*

Proof. Carrier admission gives the ordered coordinates of K by Definition 1573.1. The rows T and N are the only rows that record finite centers, membership constructors, distance witnesses, rational bounds, and tolerance refinement.

Apply the finite-net persistence clause of Definition 1573.2. It requires each completion-facing tolerance request to inspect the displayed $\mathsf{TotallyBounded}^{\uparrow}$ ledger and its refinement row before a completion read is formed.

The handoff theorem Definition 1573.8 then transports that input through F, E, C and only afterward reaches S or U. Since Definition 1573.20 excludes external compactness and selected-limit rows, none of F, E, C, S, or U can replace the total-bounded input supplied by T, N. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier total bounded input admission`

Theorem 1573.27 (TotallyBoundedCompletion completion route exactness)**.** *For every accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet, the only completion-facing route rows are the* CauchyFilterBasis$^{\uparrow}$ *row F, the* CompletionEmbedding$^{\uparrow}$ *row E, and the* ConstructiveMetricCompletion$^{\uparrow}$ *route C. All separated comparison, uniformly continuous extension, transport, provenance, and local naming reads are downstream of F, E, C and do not form another completion route.*

Proof. Project an accepted packet through Definition 1573.1. The rows named as completion-facing are exactly F, E, C: F forms the Cauchy-filter basis, E supplies the dense embedding boundary, and C records the constructive completion route.

The basis and embedding replay clause of Definition 1573.2 admits completion reads only by passing through those three rows. The displayed route of Definition 1573.8 places S, U, H, P, L strictly after them.

It remains to exclude hidden completion rows. The ledger non-escape theorem Definition 1573.20 says that exported completion reads stay inside the finite ledger $X, T, N, F, E, C, S, U, H, P, L$, and no coordinate outside F, E, C is typed as a Cauchy-filter, embedding, or constructive-completion route. Hence F, E, C are exact. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier completion route exactness`

Theorem 1573.28 (TotallyBoundedCompletion extension consumer boundary)**.** *Every* UniformContinuousExtensio *consumer of an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet must enter through the displayed finite-net and completion route surface of Definition 1573.24. Such a consumer cannot bypass T, N, F, E, C by reading only separated equality, transport, provenance, local naming, quotient completion, selected limits, countable choice, or open-cover compactness.*

Proof. Start from the extension coordinate. By Definition 1573.23, a uniformly continuous extension consumer receives the displayed route $X, T, N, F, E, C, S, U, H, P, L$.

The total-bounded input admission theorem above forces T, N to be read before the extension consumer reaches completion-facing rows. The completion route exactness theorem above then identifies F, E, C as the only completion-facing rows between those finite-net inputs and the downstream separated or extension reads.

Finally apply the ledger non-escape theorem Definition 1573.20. Since quotient completion, selected limits, countable choice, and open-cover compactness have no coordinate in the packet, and since S, H, P, L only compare, transport, source, and name the already displayed route, the extension consumer has no bypass around T, N, F, E, C. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier extension consumer boundary`

Theorem 1573.29 (TotallyBoundedCompletion root route exhaustion)**.** *Every accepted* TotallyBoundedCompletion *packet has all accepted packet routes exhausted by the finite BEDC rows*

$$X, T, N, F, E, C, S, U, H, P, L.$$

The rows X, T, N supply the root metric and total-bounded input, F, E, C supply the exact completion-facing route, S, U supply separated comparison and uniformly continuous extension handoff, and H, P, L supply transport, provenance, and local naming. No accepted route may enter through an open-cover compactness theorem, a quotient-completion carrier, a selected limit point, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Start with the route surface of Definition 1573.24. It lists the only admissible row order for an accepted carrier and places every consumer read inside the displayed finite route.

The coverage theorem Definition 1573.25 makes that displayed surface exhaustive for accepted packets. The exactness theorem Definition 1573.27 then identifies F, E, C as the only completion-facing segment of the route.

It remains to rule out an external root entry. The ledger non-escape theorem Definition 1573.20 excludes the open-cover, quotient, selected-limit, countable-choice, and ambient-real objects from the carrier. Thus every accepted route is exhausted by $X, T, N, F, E, C, S, U, H, P, L$. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier root route exhaustion`

Theorem 1573.30 (TotallyBoundedCompletion root extension ledger)**.** *Every uniformly continuous extension read through the U row of an accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet inherits the finite-net and completion rows T, N, F, E, C, S from the same* BHist *and* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *carrier. The read exports no open-cover compactness, quotient completion, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness beyond that ledger.*

Proof. The extension consumer boundary theorem Definition 1573.28 forces every U-read to enter through the displayed route surface before it may be treated as a uniformly continuous extension.

The route surface already carries the finite-net rows T, N, the completion rows F, E, C, and the separated comparison row S in the same accepted carrier. Since the read is through U, the transport and provenance rows H, P, L only preserve that finite ledger and do not supply another completion principle.

Finally apply ledger non-escape, Definition 1573.20. The excluded open-cover, quotient, selected-limit, countable-choice, and ambient-real principles have no row in the accepted BHist packet and no local $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ obligation from which an extension consumer could export them. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier root extension ledger`

Theorem 1573.31 (TotallyBoundedCompletion net-filter carrier admission)**.** *Every accepted* $\mathsf{TotallyBoundedComp}$ *packet admits its completion-facing carrier only after the source* $\mathsf{Met}^{\uparrow}$ *row X, finite epsilon-net ledger T, refinement ledger N,* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *handoff F, separated* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row E, constructive completion route C, separated comparison row S,* $\mathsf{Real}^{\uparrow}$*-facing completion seal implicit in the route, componentwise* hsame *transport H, displayed* Cont *replay,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$ *row L have all been read from the same finite* BHist *packet. No downstream* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$*, or uniformly continuous extension consumer may replace those rows by open-cover compactness, a quotient completion, selected limits, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier through Definition 1573.1. The rows $X, T, N, F, E, C, S, U, H, P, L$ are exactly the metric source, finite-net ledger, refinement ledger, Cauchy-filter basis, dense embedding, constructive completion, separated comparison, extension handoff, transport, provenance, and local naming rows.

Use finite-net persistence and basis-and-embedding replay from Definition 1573.2. They force a completion-facing consumer to read T, N before F, and then to read F, E, C before separated comparison or extension handoff.

The root route exhaustion theorem Definition 1573.29 and the root extension ledger theorem above keep S, U, H, P, L downstream of the same route. Thus later complete-metric or compact-metric consumers inherit the finite net and Cauchy-filter rows as one carrier surface, not as an open-cover or selected-limit substitute. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier net filter carrier admission`

Theorem 1573.32 (TotallyBoundedCompletion Cauchy-filter stability)**.** *For every accepted* TotallyBoundedCompl*packet, any* CauchyFilterBasis$^{\uparrow}$ *read generated from finite epsilon-net data remains stable under refinement only through the same rows* $X, T, N, F, E, C, S, U, H, P, L$*. The refinement ledger, separated completion extension,* Met$^{\uparrow}$ *source,* Real$^{\uparrow}$*-facing completion seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row therefore stay attached to the displayed Cauchy-filter route before any downstream completion consumer reads the result.*

Proof. Begin with the Cauchy-filter-basis handoff of Definition 1573.10. It requires the total-bounded ledger T and refinement row N before the basis row F can enter the dense embedding or constructive completion route.

Apply net-refinement stability, Definition 1573.9. Refining a tolerance request preserves the same finite centers, membership rows, distance witnesses, rational bounds, and Cauchy-filter handoff rather than switching to an open cover or host compactness theorem.

Finally use completion route exactness and root extension non-escape. The rows E, C, S, U, H, P, L only embed, complete, compare, extend, transport, source, and name the already displayed Cauchy-filter route. Hence stability is a property of the finite net-filter carrier itself and exports no quotient completion, selected limit, countable choice, or ambient Real-completeness row. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier cauchy filter stability`

Theorem 1573.33 (TotallyBoundedCompletion finite-net completion route)**.** *Every accepted* TotallyBoundedComp*carrier that exports a completion-facing extension must first expose the* TotallyBounded$^{\uparrow}$ *finite-net row* T*, the* CauchyFilterBasis$^{\uparrow}$ *row* F*, the* CompactMetric$^{\uparrow}$ *and* CompleteMetric$^{\uparrow}$ *completion service rows read through the displayed completion route, the* CompletionEmbedding$^{\uparrow}$ *row* E*, the* Real$^{\uparrow}$*,* StreamName$^{\uparrow}$*, and* RegSeqRat$^{\uparrow}$ *readback rows implicit in the finite tolerance schedule, and the* UniformContinuousExtension$^{\uparrow}$ *consumer row* U*. The export is a finite-net completion route through the same packet* $K = (X, T, N, F, E, C, S, U, H, P, L)$*, not a quotient completion, selected representative, choice schedule, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionFiniteNetCompletionRoute`

Proof. Project the carrier using Definition 1573.1. The source, finite-net ledger, refinement row, Cauchy-filter basis, embedding, constructive completion route, separated comparison, extension handoff, transport, provenance, and local naming rows are exactly $X, T, N, F, E, C, S, U, H, P, L$.

Use net-filter carrier admission and Cauchy-filter stability, Definition 1573.31 and Definition 1573.32. They force the finite-net rows T, N to remain attached to the Cauchy-filter row F before any downstream completion consumer can read the result.

Next apply completion route exactness and the extension consumer boundary, Definition 1573.27 and Definition 1573.28. The only completion-facing segment is F, E, C, and the extension row U is available only as a consumer of that segment.

Finally use the root extension ledger above. The Real, stream-name, and regular-rational readbacks remain schedule data for the finite tolerance route, while compact and complete metric service rows are read only through the displayed completion route. Hence no quotient completion, selected representative, choice schedule, or ambient Real completeness can be exported. □

Theorem 1573.34 (TotallyBoundedCompletion obligation ledger)**.** *Let $K = (X, T, N, F, E, C, S, U, H, P, L)$ be an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet. Its obligation ledger is exhausted by six rows of work: carrier admission; finite-net persistence;* CauchyFilterBasis$^{\uparrow}$ *generation; completion embedding; constructive completion route; and separated equality plus uniformly continuous extension handoff. The ledger exports no open-cover compactness, selected subnet, quotient completion, countable choice, ambient completeness, or detached* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier obligation ledger`

Proof. Carrier admission is the first row of work. It is supplied by the carrier definition and by Definition 1573.2, which display $X, T, N, F, E, C, S, U, H, P, L$ as the local NameCert surface.

Finite-net persistence is the second row. The net handoff theorem Definition 1573.8 keeps every completion-facing read on the finite centers, membership rows, distance witnesses, rational bounds, and refinement ledger carried by T, N.

The third and fourth rows are the Cauchy-filter-basis generation and completion embedding handoffs. Apply Definition 1573.10: the basis row F is generated only after T, N are exposed, and the dense embedding row E cannot be read while bypassing that finite-net source.

The fifth row is the constructive completion route. The completion route exactness theorem Definition 1573.27 identifies F, E, C as the only completion-facing segment, and the root route exhaustion theorem Definition 1573.29 keeps all accepted routes inside the same finite packet.

The sixth row is the separated and uniform-extension handoff. The extension nonescape theorem Definition 1573.11 and uniform-extension ledger Definition 1573.12 keep S, U, H, P, L downstream of the displayed completion route.

Finally apply ledger nonescape, Definition 1573.20. Since open-cover compactness, selected subnets, quotient completion, countable choice, ambient completeness, and detached Real completeness have no coordinate in the accepted packet, the six rows exhaust the obligation ledger. □

Theorem 1573.35 (TotallyBoundedCompletion obligation-closure route)**.** *Let $K = (X, T, N, F, E, C, S, U, H, P, L$ be an accepted* TotallyBoundedCompletion$^{\uparrow}$ *packet. The six rows of Definition 1573.34, together with carrier admission, finite-net persistence,* CauchyFilterBasis$^{\uparrow}$ *generation,* CompletionEmbedding$^{\uparrow}$, ConstructiveMetricCompletion$^{\uparrow}$, *separated equality, and* UniformContinuousExtension$^{\uparrow}$ *handoff, exhaust the obligation surface for K.*

The route is exactly the finite-net completion surface displayed by the same packet. It does not claim open-cover compactness, quotient completion, selected limit points, countable choice, detached Real$^{\uparrow}$ *completeness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier obligation closure route`

Proof. Start with carrier admission and the obligation ledger. The carrier definition and Definition 1573.2 display exactly the rows $X, T, N, F, E, C, S, U, H, P, L$, and Definition 1573.34 organizes the accepted work into the six listed obligation rows.

Next apply the finite-net handoff and persistence theorems. The net handoff Definition 1573.8, the Cauchy-filter-basis handoff Definition 1573.10, and the completion route exactness theorem

Definition 1573.27 force every completion-facing read through T, N, F, E, C before separated equality or extension data can be consumed.

Finally use the extension consumer boundary and ledger non-escape. The extension boundary Definition 1573.28 keeps uniformly continuous consumers on the same displayed route, while Definition 1573.20 excludes open-cover compactness, quotient completion, selected limit points, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, and ambient completeness as absent carrier coordinates. No further obligation row remains outside the packet. □

Theorem 1573.36 (TotallyBoundedCompletion scoped consumer coverage)**.** *Every completion-facing or uniform-extension consumer of an accepted* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *packet factors through the same finite rows*

$$X, T, N, F, E, C, S, U, H, P, L.$$

The rows are read as the source metric row X, the $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net rows T, N, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row F, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row E, the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route C, separated equality S, the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *consumer row U, componentwise transport H, provenance P, and local naming L. Thus the scoped surface covers finite-net completion, Cauchy-filter stability, completion embedding, constructive completion, separated equality, uniform continuous extension, transport, replay, provenance, and local* NameCert*; it exports no open-cover compactness, quotient completion, selected limit, countable-choice schedule, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier scoped consumer coverage`

Proof. Start with the obligation ledger and obligation-closure route, Definition 1573.34 and Definition 1573.35. They enumerate carrier admission, finite-net persistence, Cauchy-filter-basis generation, completion embedding, constructive completion, separated equality, and uniformly continuous extension handoff as rows of the same packet.

Apply the finite-net completion route and Cauchy-filter stability theorems, Definition 1573.33 and Definition 1573.32. These force every completion-facing read through T, N, F, E, C before a downstream completion or extension consumer can inspect the result.

Use the extension consumer boundary, Definition 1573.28, to place uniformly continuous extension reads after the same separated and constructive completion route. The rows S, U, H, P, L only compare, extend, transport, source, and name that route.

Finally apply the ledger non-escape theorem Definition 1573.20. Since the excluded completion principles have no coordinate in $X, T, N, F, E, C, S, U, H, P, L$, the displayed rows give the whole scoped consumer surface. □

Theorem 1573.37 (TotallyBoundedCompletion public finite-net route)**.** *Every public* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *consumer factors through the total-bounded input admission of Definition 1573.1, the finite-net and net-filter carrier admission of Definition 1573.8, the Cauchy-filter stability row of Definition 1573.32, the finite-net completion route of Definition 1573.33, the obligation ledger of Definition 1573.34, and the scoped consumer coverage of Definition 1573.36 before reading a completion result.*

Thus the public route is the displayed finite packet $K = (X, T, N, F, E, C, S, U, H, P, L)$: source metric row, total-bounded finite-net rows, net-refinement row, Cauchy-filter-basis row, completion-embedding row, constructive completion route, separated equality, uniform-extension consumer row, componentwise transport, provenance, and local NameCert*. It excludes open-cover compactness,*

hidden limit selection, quotient equality, countable choice, detached $\mathsf{Real}^{\uparrow}$ *completeness, and ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionPublicFiniteNetRoute`

Proof. Start with the carrier and finite-net admission rows. The carrier displays $X, T, N, F, E, C, S, U, H, P, L$, while the net handoff keeps every completion-facing read behind the finite centers, membership rows, distance witnesses, rational bounds, and refinement ledger.

Apply Cauchy-filter stability and the finite-net completion route. These theorems place the Cauchy-filter-basis row, completion-embedding row, and constructive completion route downstream of the same total-bounded finite-net data before any completion result can be consumed.

Finally use the obligation ledger and scoped consumer coverage. They exhaust the public-facing rows of the packet and keep separated equality, uniform extension, transport, provenance, and local naming attached to the displayed completion route, so no open-cover compactness, hidden limit selection, quotient equality, choice schedule, detached Real completeness, or ambient completeness row is available. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionCarrier public finite net route`

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1573.1, Definition 1573.2, Definition 1573.34, Definition 1573.35, and Definition 1573.36 bind every completion-facing and uniform-extension consumer to the finite rows $X, T, N, F, E, C, S, U, H, P, L$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.TotallyBoundedCompletionUp.TotallyBoundedCompletionPublicFiniteNetRoute`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient completion, selected limit points, countable choice, full $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: The next paper step is a bridge route over the same finite-net completion surface.

Constructive story: A $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ packet is generated from a metric source, a total-bounded finite-net ledger, a net-refinement row, a CauchyFilterBasis row, a CompletionEmbedding row, a ConstructiveMetricCompletion route, separated completion equality, a UniformContinuousExtension handoff, componentwise transport, provenance, and a local NameCert row.

Chapter 1574

A Concrete Naming Certificate for $\mathsf{PrincipalCauchyFilter}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ packet names the finite point-generated Cauchy-filter boundary used by Phase Real and Completion before a generic $\mathsf{CauchyFilter}^{\uparrow}$ surface is consumed. It starts from a displayed $\mathsf{RegSeqRat}^{\uparrow}$ point name, reads the finite observation windows of chapter 126, uses the dyadic tolerance rows of chapter 291, saturates into the finite filter surface of chapter 333, and leaves the downstream seal of chapter 38 behind that route. The packet is a point-generated filter certificate, not a quotient filter, not a selected limit, not host real equality, and not ambient completeness.

Definition 1574.1 (PrincipalCauchyFilter carrier)**.** *A* $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, S, R, D, G, F, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-name row whose observations generate the principal Cauchy filter.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the point-name row is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare observations at the requested filter scale.*
5. *G is the principal-generator row. For each displayed precision request it names the finite tail neighbourhood generated by X, S, R, D.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation row obtained from the generator row G by closing only the finite directed-refinement obligations recorded in the packet.*
7. *A is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row reached only after X, S, R, D, G, F have been consumed.*
8. *H records componentwise* hsame *transport for the point name, stream schedule, regular readback, dyadic tolerance ledger, generator row, saturated filter row, Real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a filter request back through the principal generator and then through the stream, regular rational, dyadic tolerance, and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrincipalCauchyFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable-choice selectors, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1574.2 (PrincipalCauchyFilter NameCert obligations)**.** *For every accepted* $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ *carrier $K = (X, S, R, D, G, F, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PrincipalCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the point-name row, stream schedule, regular rational readback, dyadic tolerance ledger, principal generator, saturated Cauchy filter, Real seal, transport, continuation, provenance, and local naming row;*
2. *point generation, requiring every displayed filter neighbourhood in G to be generated from the same X, S, R, D rows;*
3. *saturation discipline, requiring the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F to consume the finite generator row G before any completion-facing read can use a generic Cauchy-filter surface;*
4. *seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row A to remain downstream of X, S, R, D, G, F;*
5. *ledger refusal, excluding quotient filters, selected limits, countable-choice selectors, host real equality, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1574.1. The projection exposes exactly the point-name row, stream schedule, regular readback, dyadic tolerance ledger, principal generator, saturated filter row, Real seal, and structural rows H, C, P, N.

Point generation is forced by the row inventory. The generator row G has no independent source; each of its displayed neighbourhoods reads the same point observation through S, the same regular rational readback through R, and the same dyadic tolerance ledger through D.

Saturation discipline is then internal to F. A completion-facing consumer reaches the generic $\mathsf{CauchyFilter}^{\uparrow}$ surface only after G has been consumed, so the packet cannot exchange the principal generator for an arbitrary quotient filter.

Finally, the seal row A is downstream of the same route. Transport by H preserves each coordinate, while C, P, and N replay and name only that finite route; hence no selected limit, countable-choice selector, host real equality, or ambient completeness coordinate remains available. □

Theorem 1574.3 (PrincipalCauchyFilter RegSeqRat handoff)**.** *Let $K = (X, S, R, D, G, F, A, H, C, P, N)$ be an accepted* $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ *carrier. Every completion-facing read that starts from the regular rational point name factors through the finite route*

$$X, S, R, D \longmapsto G \longmapsto F \longmapsto A \longmapsto H, C, P, N.$$

Thus the $\mathsf{RegSeqRat}^{\uparrow}$ *row reaches the generic* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation surface only through the principal generator row, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal remains after that saturation handoff.*

Proof. Begin with a completion-facing consumer of the accepted packet. Carrier admission gives access to the saturated filter row F and the seal row A, but only as rows downstream of the point-name route X, S, R, D, G.

Replay the continuation row C. The request first reads the stream schedule S, then the regular rational readback R, then the dyadic tolerance ledger D attached to the same point-name row X.

Only after those rows have been consumed can the principal generator G be saturated into the $\mathsf{CauchyFilter}^{\uparrow}$ row F. The seal row A and the structural rows H, C, P, N record the same dependency order, which proves the stated factorization. □

Theorem 1574.4 (PrincipalCauchyFilter ledger non-escape)**.** *For every accepted* $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ *carrier* $K = (X, S, R, D, G, F, A, H, C, P, N)$*, every exported filter or Real-facing read remains inside the finite ledger*

$$X, S, R, D, G, F, A, H, C, P, N.$$

It cannot produce a quotient filter, a selected limit, a countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an exported read from the carrier. By Definition 1574.1, the read has access only to the eleven displayed coordinates and their componentwise transport and continuation rows.

Apply Definition 1574.2. Point generation keeps the read inside X, S, R, D, G; saturation discipline moves it through F; seal nonescape moves it through A; and the structural rows H, C, P, N record only the same route.

Every excluded principle is absent from this ledger. Quotient filters, selected limits, countable-choice selectors, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, no replay edge, and no provenance row in the packet, so the exported read cannot leave the displayed finite surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrincipalCauchyFilter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrincipalCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1574.1, Definition 1574.2, Definition 1574.3, and Definition 1574.4 seed the finite point-generated Cauchy-filter boundary between regular rational names, generic Cauchy-filter evidence, and the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrincipalCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable-choice selectors, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires carrier admission, point generation, CauchyFilter saturation handoff, RegSeqRat readback, Real seal factorization, and ledger refusal for the same packet.

Constructive story: A $\mathsf{PrincipalCauchyFilter}^{\uparrow}$ packet is generated from a point-name row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a principal-generator row, a CauchyFilter saturation row, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1575

A Concrete Naming Certificate for $\mathsf{LocatedCauchyEmbedding}^{\uparrow}$

Bishop–Bridges constructive analysis reads a located Cauchy object through explicit approximation and location data before it is admitted to a completed real surface. The $\mathsf{LocatedCauchyEmbedding}^{\uparrow}$ packet binds the located Cauchy source of chapter 340 and the located real boundary of **??** to the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the completion handoff of chapter 1512 and chapter 1560, and the terminal real-name seal of chapter 38. It is a finite embedding route for located Cauchy data, not a quotient completion, a selected host limit, countable choice, host real equality, trichotomy, open-cover compactness, or an ambient completeness principle.

Definition 1575.1 (Located Cauchy embedding carrier)**.** *A* $\mathsf{LocatedCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, S, R, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the located Cauchy source row, including the accepted approximation schedule, Cauchy modulus evidence, and locatedness witnesses that the embedding route may inspect.*
2. *S is a finite* $\mathsf{StreamName}^{\uparrow}$ *window over the source schedule, recording the precision requests and observation slots used by the embedding read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S, turning the displayed windows into regular rational observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that budgets the endpoint, radius, and comparison tolerances used by the located Cauchy source and the completion handoff.*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *or* $\mathsf{CompletionDensity}^{\uparrow}$ *handoff row by which the displayed source observations enter the completion-facing boundary.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. It accepts only the route already displayed through X, S, R, D, E.*
7. *H records componentwise* hsame *transport for the source row, stream window, readback row, dyadic ledger, embedding handoff, real seal, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a located Cauchy precision request through S, R, D, E, Q.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchy}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionDensity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyEmbedding}^{\uparrow}}$ *row naming the displayed embedding route and the permitted reads from it.*

The classifier on such packets is componentwise transport through H together with replay through C. No coordinate exports quotient completion, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 1575.2 (Located Cauchy embedding NameCert obligations)**.** *For every accepted* LocatedCauchyEmbeddi *carrier $L = (X, S, R, D, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located Cauchy source, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, completion handoff, Real seal, transports, replay route, provenance, and local naming row;*
2. *located-source exactness, requiring every embedding read to start from X and to use only the approximation schedule, Cauchy modulus evidence, and locatedness witnesses displayed there;*
3. *window and readback compatibility, requiring the finite window S and the regular rational readback R to remain attached to the same source requests;*
4. *dyadic tolerance compatibility, requiring the tolerance row D to budget the located-source comparisons and the completion-facing handoff without introducing a host real comparison;*
5. *embedding handoff, requiring the row E to be the only passage from the located Cauchy source route into the completion-facing surface;*
6. *Real-seal non-escape, requiring the terminal row Q to read only the already displayed route X, S, R, D, E;*
7. *provenance exhaustion, requiring H, C, P, N to transport, replay, package, and name only these rows.*

No NameCert obligation is stored outside this displayed packet.

Proof. Carrier admission is projection from Definition 1575.1. The tuple displays the source row, finite window, regular rational readback, dyadic tolerance ledger, completion handoff, Real seal, transports, replay, provenance, and local name.

Located-source exactness is forced by the first coordinate. Any embedding read begins with X, so its approximation schedule, Cauchy modulus evidence, and locatedness witnesses are the only source-side data available.

Window and readback compatibility is projection from S and R. The stream window records the finite requests, and the regular rational row reads exactly those requests before any dyadic or completion-facing row can be inspected.

Dyadic tolerance compatibility and the embedding handoff are then read from D and E. The tolerance row budgets the displayed comparisons, while the handoff row is the unique passage from the source route into the completion boundary.

The Real seal and structural rows close the surface. The row Q is downstream of X, S, R, D, E, and H, C, P, N only transport, replay, package, and name that same route; hence every listed obligation is a row projection and no further obligation is stored outside the packet. □

Theorem 1575.3 (Located Cauchy embedding Real-seal handoff)**.** *Let* $L = (X, S, R, D, E, Q, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyEmbedding}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing read from the located Cauchy embedding factors through*

$$X, S, R, D, E, Q, H, C, P, N.$$

Consequently the read cannot use quotient completion, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or ambient Real completeness.

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing read. The carrier exposes the terminal Real seal only as the row Q, and that row is listed after the source route X, S, R, D, E in Definition 1575.1.

The source route is finite. The located Cauchy row X supplies the approximation and locatedness data, the window S fixes the finite requests, the readback R supplies regular rational observations, the tolerance ledger D budgets comparisons, and the handoff E is the only completion-facing passage.

Transport, replay, provenance, and local naming do not enlarge this route. The rows H, C, P, N preserve the same displayed coordinates under componentwise sameness, continuation replay, packaging, and NameCert naming.

None of the excluded principles has a coordinate, replay edge, or provenance row in the packet. The Real-facing read therefore factors through the displayed tuple and cannot import quotient completion, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyEmbedding}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1575.1, Definition 1575.2, and Definition 1575.3 seed the finite route from located Cauchy source data through StreamName, RegSeqRat, DyadicRatCore, CompletionEmbedding or CompletionDensity, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, located-source exactness, window and readback compatibility, dyadic tolerance compatibility, embedding handoff, Real-seal non-escape, and provenance exhaustion for BEDC.Derived.LocatedCauchyEmbeddingUp.

Constructive story: A $\mathsf{LocatedCauchyEmbedding}^{\uparrow}$ packet is generated from located Cauchy source data, a finite StreamName window, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CompletionEmbedding or CompletionDensity handoff, a Real seal boundary, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1576

A Concrete Naming Certificate for $\mathsf{RealCauchyModulusSelector}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ packet records the finite selection surface by which a displayed modulus schedule for a regular rational Cauchy name chooses the threshold, stream window, tolerance ledger, readback row, and seal route consumed by a downstream $\mathsf{Real}^{\uparrow}$ reader. It sits over the explicit Real Cauchy modulus packet of chapter 1423, the generic modulus schedule of chapter 292, the finite observation windows of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet makes the selector row visible as finite BEDC data before any consumer treats a selected threshold as part of a Real-facing read.

Definition 1576.1 (Real Cauchy modulus selector carrier)**.** *A* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, D, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *row whose precision-indexed schedule is being inspected;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row named by the selected threshold;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row read at the selected window;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to that tolerance and window;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal route reached only after M, S, D, Q have been consumed;*
6. *R is the selector row choosing the displayed threshold, finite window, dyadic tolerance, regular readback, and seal route from the modulus schedule in M;*
7. *H records componentwise* hsame *transport for the modulus row, selected window, dyadic tolerance, regular readback, Real seal, selector row, continuation row, provenance row, and local naming row;*
8. *C records displayed* Cont *replay from a precision request through R and then through M, S, D, Q, E, H, P, N;*
9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyModulusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits outside the displayed row R, open-cover compactness, or an unbounded stream read.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorTasteGate single carrier alignment`

Theorem 1576.2 (Real Cauchy modulus selector NameCert obligations)**.** *For an accepted* RealCauchyModulusSelec *carrier $T = (M, S, D, Q, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyModulusSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Real Cauchy modulus row, finite stream window, dyadic tolerance ledger, regular rational readback, Real seal, selector row, transports, replay route, provenance, and local naming row;*
2. *selector exactness, requiring R to choose only the threshold, window, tolerance, readback, and seal route already displayed by M, S, D, Q, E;*
3. *modulus compatibility, requiring every selected threshold to be read from the accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *row and therefore from the upstream* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule;*
4. *finite-window stability, requiring the rows S, D, Q, E to remain attached to the selector row R under componentwise* hsame *transport H and displayed* Cont *replay C;*
5. *provenance exhaustion, requiring P and N to name only the displayed route and to export no Real completeness theorem, quotient stream equality, countable-choice selector, trichotomy, host real equality, selected limit outside R, open-cover compactness, or unbounded stream read.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1576.1. The tuple displays the modulus row, selected finite window, dyadic tolerance ledger, regular rational readback, Real seal, selector row, transport, replay, provenance, and local name.

Selector exactness is forced by the position of R in the tuple. A precision request reaches R as a row selecting from the displayed modulus route, and the selected data are precisely the threshold, window, tolerance, readback, and seal rows already exposed as M, S, D, Q, E.

Modulus compatibility follows by reading M as the accepted $\mathsf{RealCauchyModulus}^{\uparrow}$ row. Since the upstream Real Cauchy modulus packet is itself over the $\mathsf{CauchyModulus}^{\uparrow}$ schedule, the selector cannot draw a threshold from an independent stream or from an ambient host principle.

Stability is the role of H and C. Componentwise hsame transport preserves the selected rows, while continuation replay repeats the same finite route from the precision request through the selector and then through M, S, D, Q, E.

Finally, P and N package and name only the displayed rows. The carrier contains no coordinate for Real completeness, quotient stream equality, countable choice, trichotomy, host real equality, selected limits outside R, open-cover compactness, or unbounded stream reads, so the local NameCert surface cannot export any such data. □

Theorem 1576.3 (Real Cauchy modulus selector finite-window soundness)**.** *Let $T = (M, S, D, Q, E, R, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packet. For every finite unary precision request, the*

selector row R produces a finite Real-facing route

$$R \longmapsto M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N)$$

whose selected threshold, window, tolerance, readback, and seal row are all displayed in the carrier. The route cannot be replaced by Real completeness, quotient stream equality, countable choice, trichotomy, host real equality, a selected limit outside R, open-cover compactness, or an unbounded stream read.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier finite window soundness`

Proof. Proceed by induction on the finite unary precision request. In the zero stage, project the accepted carrier and read the selector row R. The selector exposes a displayed threshold in the modulus row M, the corresponding finite StreamName window S, the dyadic tolerance row D, the regular rational readback Q, and the terminal seal row E.

For the successor stage, assume the preceding precision request has been handled by the displayed route. The next request asks R for the next threshold in the same finite modulus schedule. The route may enlarge the selected window in S and refine the tolerance row in D, but it still reaches the regular readback row Q and terminal seal row E through the same carrier coordinates.

Transport and replay close the induction step. The componentwise row H preserves the selected coordinates, while C replays the selector route from the precision request through R, M, S, D, Q, E and then through P, N.

The exclusion clauses follow by inspecting the carrier. Since none of Real completeness, quotient stream equality, countable choice, trichotomy, host real equality, selected limits outside R, open-cover compactness, or unbounded stream reads appears as a coordinate of T, no induction stage can substitute such a principle for the finite selector route. □

Theorem 1576.4 (Real Cauchy modulus selector monotone window)**.** *Let $T = (M, S, D, Q, E, R, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packet. If the selector row R accepts a finite window at unary precision n, then every displayed larger unary precision window accepted by the same* $\mathsf{RealCauchyModulus}^{\uparrow}$ *row M is accepted after replay through the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. The replay may enlarge the observed prefix and refine the tolerance ledger, but it must still pass through S, D, Q, E, H, C, P, N and cannot use Real completeness, quotient stream equality, countable choice, host real equality, or an unbounded stream read.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier window monotone`

Proof. Project the finite-window route of Definition 1576.3. Acceptance at precision n exposes the selected threshold in M, the finite StreamName window S, the dyadic tolerance row D, the regular rational readback Q, and the terminal Real seal E.

Now inspect a displayed larger unary precision request. Because the request is read through the same $\mathsf{RealCauchyModulus}^{\uparrow}$ row M, its threshold is a later entry of the same finite modulus schedule rather than an independent selector. The StreamName window may be longer and the dyadic tolerance may be sharper, but both remain rows of the displayed S, D, Q, E route.

Monotone reuse is therefore a finite-window projection. The earlier accepted prefix remains visible inside the larger displayed window, and the tolerance ledger is replayed by C through R, M, S, D, Q, E before provenance and local naming are read through P, N.

Finally H transports the same coordinates componentwise. Since the carrier has no coordinate for Real completeness, quotient stream equality, countable choice, host real equality, or unbounded

stream reads, no larger-window acceptance can be supplied by anything outside the displayed selector packet. □

Theorem 1576.5 (Real Cauchy modulus selector seal factorization)**.** *Let* $T = (M, S, D, Q, E, R, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packet. Every accepted selector route reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through*

$$R \longmapsto M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The route cannot factor through Real completeness, quotient stream equality, countable choice, host equality, or an unbounded stream read before the $\mathsf{Real}^{\uparrow}$ *seal is named.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier seal factorization`

Proof. Use finite-window soundness, Definition 1576.3. A finite precision request first reads the selector row R, then the accepted modulus row M, the selected stream window S, the dyadic tolerance row D, the regular rational readback row Q, and the terminal seal row E.

Use monotone-window reuse, Definition 1576.4. Larger precision requests may enlarge the displayed window and refine the tolerance ledger, but they replay the same R, M, S, D, Q, E route before H, C, P, N package and name the packet.

The exclusion clause follows by projection of Definition 1576.1. None of Real completeness, quotient stream equality, countable choice, host equality, or unbounded stream reads is a carrier coordinate, so none can be an intermediate factor in the accepted seal route. □

Theorem 1576.6 (Real Cauchy modulus selector transport naturality)**.** *Let* $T = (M, S, D, Q, E, R, H, C, P, N)$ *and* $T' = (M', S', D', Q', E', R', H', C', P', N')$ *be accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packets. Suppose the displayed transport row* H *relates the coordinates componentwise:*

$$M\mathsf{hsame}M', \quad S\mathsf{hsame}S', \quad D\mathsf{hsame}D', \quad Q\mathsf{hsame}Q', \quad E\mathsf{hsame}E', \quad R\mathsf{hsame}R', \quad C\mathsf{hsame}C', \quad P\mathsf{hsame}P', \quad N\mathsf{hs}$$

Then every selected finite precision route

$$R \longmapsto M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

in T *is transported to the corresponding route*

$$R' \longmapsto M' \longmapsto S' \longmapsto D' \longmapsto Q' \longmapsto E'$$

in T' *with the same excluded readings: no Real completeness, quotient stream equality, countable choice, host equality, selected limit outside the selector row, or unbounded stream read is introduced.*

Proof. Project both accepted carriers using Definition 1576.1. The selected route in T is a displayed route through the selector row, modulus row, finite StreamName window, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Apply finite-window soundness, Definition 1576.3, to the route in T. It identifies R, M, S, D, Q, E as the only selected coordinates visible to the precision request. Use monotone-window reuse, Definition 1576.4, when the transported route is read at a larger displayed precision.

Now use the componentwise H row. It transports the selector, threshold source, window, tolerance, readback, seal, replay, provenance, and local name coordinates to $R', M', S', D', Q', E', C', P', N'$. The transported route is therefore the same finite selector route in T', read through corresponding displayed rows rather than through a new carrier.

Finally apply seal factorization, Definition 1576.5. Since neither carrier has a coordinate for Real completeness, quotient stream equality, countable choice, host equality, selected limits outside the selector row, or unbounded stream reads, componentwise transport cannot introduce any of those readings. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier selector transport naturality`

Theorem 1576.7 (Real Cauchy modulus selector tail bound)**.** *Let $T = (M, S, D, Q, E, R, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packet. A selected unary precision request bounds the displayed StreamName window before the terminal seal row E is read: the selector must pass through the modulus row M, the finite window row S, the dyadic tolerance row D, and the regular rational readback row Q before any Real-facing seal consumer can inspect E. The bound is therefore a finite tail-bound row over M, S, D, Q, H, C, P, N, not a completeness principle, quotient stream equality, countable-choice selector, host real equality, or unbounded stream read.*

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier tail bound`

Proof. Start with finite-window soundness, Definition 1576.3. A unary precision request first exposes the selected threshold in M, the StreamName window S, the dyadic tolerance row D, and the regular rational readback row Q.

Use monotone-window reuse, Definition 1576.4. Larger displayed precision requests may enlarge the observed prefix and refine the tolerance ledger, but they still replay the same M, S, D, Q route before reaching the terminal Real seal.

Finally apply seal factorization, Definition 1576.5. The seal row E is downstream of the selector route, so a Real-facing consumer cannot inspect it until the finite tail-bound data have already been displayed through H, C, P, N. □

Theorem 1576.8 (Real Cauchy modulus selector Real-seal route)**.** *Every Real-facing seal read exported by an accepted* $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ *packet factors through*

$$\mathsf{RealCauchyModulus}^{\uparrow}, \mathsf{CauchyModulus}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCe}$$

The selector contributes only the finite route from a displayed precision request to a StreamName window, dyadic tolerance ledger, regular rational readback, and terminal Real seal. It does not export Real completeness, quotient stream equality, countable choice, trichotomy, host real equality, selected limits outside R, open-cover compactness, or unbounded stream reads.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusSelectorUp.RealCauchyModulusSelectorCarrier real seal route`

Proof. Apply the NameCert obligations of Definition 1576.2. They force every accepted packet to display the Real Cauchy modulus row, selected StreamName window, dyadic tolerance ledger, regular rational readback, selector row, transport, replay, provenance, and local naming.

Apply the tail-bound theorem, Definition 1576.7. It places the StreamName window, dyadic tolerance, and regular rational readback before the terminal Real seal in every selected precision route.

Use seal factorization, Definition 1576.5. The Real-facing seal is therefore reached only after the upstream $\mathsf{RealCauchyModulus}^{\uparrow}$ and $\mathsf{CauchyModulus}^{\uparrow}$ schedule has been read through M, and after the S, D, Q, H, C, P, N structural rows have preserved the selected finite window.

The exclusions are exactly the non-coordinates of Definition 1576.1. Since the carrier contains none of the listed ambient principles, the Real-seal route remains the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyModulusSelector}\uparrow}$.

Constructive story: A $\mathsf{RealCauchyModulusSelector}^{\uparrow}$ packet is generated from an accepted $\mathsf{RealCauchyModulus}^{\uparrow}$ modulus row, a selected $\mathsf{StreamName}^{\uparrow}$ finite window, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, a selector row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyModulusSelector}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 1576.1, Definition 1576.2, Definition 1576.3, Definition 1576.4, Definition 1576.5, Definition 1576.6, Definition 1576.7, and Definition 1576.8 bind the finite selector route from Real Cauchy modulus schedules to Real-facing seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyModulusSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, countable choice, trichotomy, host real equality, selected limits outside the displayed selector row, open-cover compactness, Lean verification, or a public standard bridge.

Upgrade path: Scoped closure requires checked carrier admission, selector exactness, modulus compatibility, finite-window stability, finite unary precision induction, and provenance exhaustion for BEDC.Derived.RealCauchyModulusSelectorUp.

Chapter 1577

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ packet records the finite route by which an explicit Cauchy modulus schedule becomes a regular rational real-name surface before any $\mathsf{Real}^{\uparrow}$ seal is read. It sits after the explicit rate rows of chapter 292, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the modulus-to-real boundary of chapter 1423. The packet is a finite NameCert realization of a displayed modulus schedule; it is not a quotient stream, not host real equality, not countable choice, not a selected limit, and not an ambient completeness principle.

Definition 1577.1 (Regular Cauchy modulus realization carrier)**.** *A* $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, carrying the precision request, the displayed threshold, and the finite late-pair budget consumed by the realization.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the stream entries inspected after the threshold row of M has been read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row containing the dyadic radius and comparison cells used for the requested precision.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by routing the windows in S through the dyadic tolerance ledger D under the schedule exposed by M.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, D, Q have been replayed.*
6. *H records componentwise* hsame *transport for the modulus, window, dyadic tolerance, regular readback, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through the modulus threshold, finite stream windows, dyadic tolerance rows, regular rational readback, and then the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealization}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient streams, host real equality, countable-choice selection, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 1577.2 (Regular Cauchy modulus realization window coverage)**.** *Let $R = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ *carrier. Every precision request admitted by M reads through a finite window containing the displayed modulus threshold, the corresponding* $\mathsf{StreamName}^{\uparrow}$ *observations in S, and the dyadic tolerance rows in D. No precision request can reach Q or E without first consuming that finite window.*

Proof. Project the carrier of Definition 1577.1. The projection exposes the modulus row M, the window row S, the dyadic tolerance row D, and the downstream rows Q, E, H, C, P, N.

Replay the continuation row C at the chosen precision request. Its first visible dependency is the threshold row carried by M; after that threshold is read, the same route names the finite observations in S and the dyadic cells in D.

The rows Q and E occur downstream of that replay. Since the carrier has no second route to a regular readback or Real seal, each accepted precision request is covered by the displayed finite window and tolerance ledger. □

Theorem 1577.3 (Regular Cauchy modulus realization RegSeqRat handoff)**.** *Let $R = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ *carrier. The same rows that cover each precision request repack as the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q: the modulus row M supplies the late-window budget, the stream row S supplies the inspected entries, and the dyadic row D supplies the tolerance comparisons. Thus Q is a readback of the displayed finite route, not a separate regular-sequence oracle.*

Proof. Use Definition 1577.2. For the chosen precision request, the finite route has already displayed the modulus threshold, the StreamName observations, and the dyadic tolerance rows.

The regular readback row Q is listed in the carrier only after M, S, D. Repacking therefore uses exactly those rows: M contributes the late-window budget, S contributes the inspected entries, and D contributes the dyadic comparisons that certify the requested tolerance.

Transport by H preserves this row order, while P and N name only the same finite route. Hence Q cannot be read as an independent regular-sequence oracle. □

Theorem 1577.4 (Regular Cauchy modulus realization Real seal nonescape)**.** *Let $R = (M, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the finite route*

$$M, S, D, Q \longmapsto E \longmapsto H, C, P, N.$$

The route rules out quotient streams, host real equality, countable choice, selected limits, and ambient completeness: none of those objects is a row of the carrier, and none can be reconstructed from the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealization}^{\uparrow}}$ *surface.*

Proof. Begin with a Real-facing consumer of the packet. Carrier admission gives access to the seal row E, but only as a row downstream of the modulus, window, dyadic tolerance, and regular readback rows.

Replay the continuation row C. The consumer first reads the explicit schedule in M, then the finite windows in S, the dyadic tolerance cells in D, and the regular rational readback in Q.

Only after this route has been consumed can the seal row E be read. The structural rows H, C, P, N record the same dependency order, so there is no carrier coordinate from which to read quotient streams, host real equality, countable choice, selected limits, or ambient completeness. □

Theorem 1577.5 (Regular Cauchy modulus realization NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ *carrier* $R = (M, S, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMod}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the explicit Cauchy modulus row, finite stream-window row, dyadic tolerance ledger, regular rational readback, Real seal, transport, continuation, provenance, and local naming row;*
2. *window coverage, requiring each precision request to consume the displayed threshold in* M*, the inspected windows in* S*, and the tolerance cells in* D *before reading* Q *or* E*;*
3. *RegSeqRat handoff, requiring the rows* M, S, D *to repack as the regular rational readback* Q*;*
4. *Real seal nonescape, requiring the seal row* E *to remain downstream of* M, S, D, Q*;*
5. *ledger refusal, excluding quotient streams, host real equality, countable choice, selected limits, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1577.1. The projection exposes exactly the explicit modulus row, finite window row, dyadic tolerance row, regular rational readback row, Real seal, and structural rows H, C, P, N.

Window coverage is Definition 1577.2. It forces every precision request to pass through the displayed modulus threshold, StreamName observations, and dyadic tolerance cells before any readback or seal is available.

The RegSeqRat handoff is Definition 1577.3. It identifies the row Q as a repacking of the same finite route, not as an independent regular-sequence oracle.

The seal refusal clause is Definition 1577.4. Since quotient streams, host real equality, countable choice, selected limits, and ambient completeness are not carrier rows, no additional NameCert obligation can name them. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealization}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1577.1, Definition 1577.2, Definition 1577.3, Definition 1577.4, and Definition 1577.5 seed the finite route from an explicit $\mathsf{CauchyModulus}^{\uparrow}$ schedule through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the downstream $\mathsf{Real}^{\uparrow}$ seal, with $\mathsf{RealCauchyModulus}^{\uparrow}$ as the sibling real-modulus boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, host real equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, RegSeqRat handoff, Real seal nonescape, and ledger refusal proofs for the same packet.

Constructive story: A $\mathsf{RegularCauchyModulusRealization}^{\uparrow}$ packet is generated from an explicit CauchyModulus schedule, StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1578

A Concrete Naming Certificate for $\mathsf{RegularCauchyShift}^{\uparrow}$

Following Bishop and Bridges-style regular Cauchy sequence tail invariance, the $\mathsf{RegularCauchyShift}^{\uparrow}$ packet records the finite route by which a regular rational Cauchy name may discard a displayed finite prefix without leaving the visible regularity, tolerance, and real-seal surface. It sits over the finite schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the Cauchy-name space of chapter 406. The packet is a NameCert for tail-shift invariance of regular Cauchy names, not a quotient of streams, not a selected host limit, not countable choice, and not ambient completeness.

Definition 1578.1 (Regular Cauchy shift carrier)**.** *A* $\mathsf{RegularCauchyShift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, S, D, k, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regular rational Cauchy readback supplies the inspected values and the regularity budget.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row whose shifted window request sends the displayed index n to the source index $n + k$.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to preserve the same dyadic error budget after the source window has been shifted.*
4. *k is the unary shift witness, displayed as a finite prefix length and carrying no unbounded selector.*
5. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal route reached only after Q, S, D, k have been consumed.*
6. *H records componentwise* hsame *transport for the source regular sequence row, schedule row, dyadic ledger, unary shift witness, Real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a shifted finite-window request through k, then through S, Q, D, and finally to the seal row E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, host sequence equality, countable-choice tail selection, or ambient completeness.

Theorem 1578.2 (Regular Cauchy shift NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyShift}^{\uparrow}$ *carrier* $K = (Q, S, D, k, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, unary shift witness,* $\mathsf{Real}^{\uparrow}$ *seal route, transport, continuation, provenance, and local naming row;*
2. *finite-window shift stability, requiring every shifted request to replay through the displayed unary shift witness before it can inspect the source schedule;*
3. *dyadic tolerance preservation, requiring the tolerance ledger* D *to be read after the shifted source window and before any regularity or seal handoff is exported;*
4. *RegSeqRat handoff, requiring the shifted finite windows to remain attached to the same displayed regular rational Cauchy row* Q*;*
5. *RealUp seal routing, requiring every downstream seal read to pass through* Q, S, D, k *before it reaches* E*;*
6. *ledger refusal, excluding quotient streams, selected limits, host sequence equality, countable-choice tail selectors, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyShiftUp.RegularCauchyShiftTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1578.1. The projection exposes exactly the source regular sequence row, schedule row, dyadic tolerance ledger, unary shift witness, Real seal route, and structural rows H, C, P, N.

Replay the continuation row C at a shifted finite-window request. The request first consumes the unary witness k, then reads the source schedule S at the translated window and consults the regular rational row Q at that displayed source position.

The dyadic row D is downstream of the shifted window route, so the same tolerance budget is preserved before the RegSeqRat handoff or the Real seal can be inspected. Since the seal row E is downstream of Q, S, D, k, every Real-facing obligation is routed through those finite rows.

The listed rows exhaust the carrier. The excluded quotient streams, selected limits, host sequence equality, countable-choice tail selectors, and ambient completeness principles therefore have no coordinate from which a local $\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}$ obligation could be read. □

Theorem 1578.3 (Regular Cauchy shift real-seal non-escape)**.** *Let* $K = (Q, S, D, k, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyShift}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer read and every CauchySequenceSpace-facing consumer read factors through the displayed shifted finite-window route*

$$Q, S, k, D \longmapsto E \longmapsto H, C, P, N.$$

The packet exports no quotient stream, selected limit, host sequence equality, countable-choice tail selector, or ambient completeness principle.

Proof. Begin with a consumer read of the shifted regular Cauchy name. Carrier admission gives the downstream row E and the structural rows H, C, P, N, but only with the source regularity row Q, schedule row S, unary shift witness k, and dyadic ledger D already present.

Replay C. A consumer request first passes through the finite shift witness k, then consults the schedule row S at the translated source window. The readback row Q supplies the regular rational Cauchy observation for that window, and D supplies the dyadic tolerance ledger attached to the same read.

Only after those rows have been consumed can the request reach the $\mathsf{Real}^{\uparrow}$ seal route or the CauchySequenceSpace-facing handoff. Transport by H preserves the same row order, while P and N package only that finite route; hence the packet has no export path for quotient streams, selected limits, host sequence equality, countable-choice tail selection, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyShift}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName shifted schedule, DyadicRatCore tolerance ledger, unary shift witness, RealUp seal route, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1578.1, Definition 1578.2, and Definition 1578.3 seed the finite tail-shift route from $\mathsf{RegSeqRat}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the downstream $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyShift}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host sequence equality, countable-choice tail selection, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-window shift stability, dyadic tolerance preservation, RegSeqRat handoff, RealUp seal routing, and ledger refusal for the same packet.

Chapter 1579

A Concrete Naming Certificate for $\mathsf{LimitPoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LimitPoint}^{\uparrow}$ packet records a constructive cluster-point and accumulation-point surface for real sequences by finite neighborhoods and cofinal subsequence windows. It reads finite windows from chapter 126, dyadic neighborhood radii from chapter 291, regular rational readback from chapter 301, the terminal real-name seal of chapter 38, the bounded source surface of chapter 1415, and the subsequence compactness route of chapter 1254. The packet gives downstream compactness arguments a BEDC-native cluster-point interface; it is not host completeness, quotient equality of streams, open-cover compactness, or a choice-selected subsequence.

Definition 1579.1 (LimitPoint carrier)**.** *A* $\mathsf{LimitPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, D, Q, \Phi, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real sequence row, including the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source data when the packet is supplied from a bounded sequence and the sequence observations consumed by cluster-point readers.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the source windows and the selected cofinal windows inspected at the displayed precision requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *neighborhood-radius row containing the positive dyadic radii, comparison cells, and finite inclusion tests used to read a neighborhood of the proposed cluster point.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and dyadic radius ledger before any real seal is exposed.*
5. *Φ is the cofinal subsequence witness row. It stores only the inspected monotone selector cells, later-index comparisons, and cofinal threshold witnesses needed by the finite windows in W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *cluster seal row reached only after S, W, D, Q, Φ have been replayed.*
7. *H records componentwise* hsame *transport for the source sequence, StreamName windows, DyadicRatCore radii, RegSeqRat readback, cofinal subsequence witness, Real cluster seal, continuation row, provenance, and local naming row.*

8. *C records displayed* Cont *replay from a neighborhood request through the source sequence, finite windows, dyadic radius ledger, cofinal subsequence witness, regular rational readback, and then the Real cluster seal.*

9. *P is the* Pkg *provenance over* $\mathsf{LimitPoint}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitPoint}^{\uparrow}}$ *row naming the displayed packet and its permitted consumer reads.*

The classifier compares accepted packets only through these rows and their transported replay paths. It has no coordinate for host real completeness, quotient streams, open covers, arbitrary choice of subsequences, or a least cluster-point selector.

Theorem 1579.2 (LimitPoint NameCert obligations)**.** *For every accepted* $\mathsf{LimitPoint}^{\uparrow}$ *carrier* $L = (S, W, D, Q, \Phi, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LimitPoint}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source real sequence, finite StreamName windows, DyadicRatCore neighborhood radii, RegSeqRat readback, cofinal subsequence witness, Real cluster seal, transport, continuation replay, provenance, and local naming row;*

2. *neighborhood coverage, requiring every cluster-point request to consume a displayed dyadic radius in D and a finite selected window in W before it reads the regular rational row Q;*

3. *cofinal-window persistence, requiring the selected windows in W to travel through the cofinal witness row Φ with displayed monotone and later-index evidence;*

4. *Real seal routing, requiring the cluster seal row E to be reached only after S, W, D, Φ, Q have been displayed;*

5. *ledger non-escape, excluding host completeness, quotient stream equality, open-cover compactness, arbitrary choice-selected subsequences, and least cluster-point selection.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1579.1. The projection exposes exactly the source sequence, finite window, dyadic radius, regular readback, cofinal witness, Real seal, transport, continuation, provenance, and local naming rows.

The neighborhood coverage row is read through W and D. A cluster-point request first fixes the inspected finite source and selected windows, then reads the positive dyadic radius and comparison cells attached to that request, and only then reaches the regular rational row Q.

The cofinal-window row Φ controls the selected windows. It supplies the inspected monotone selector cells, later-index comparisons, and cofinal threshold witnesses, so the selected windows cannot be replaced by an arbitrary host subsequence.

Finally, the seal row E sits downstream of S, W, D, Φ, Q, while H, C, P, N preserve and name the same route. Since none of these rows carries host completeness, quotient stream equality, open-cover compactness, arbitrary choice of subsequences, or a least cluster-point selector, the five listed obligations exhaust the local NameCert surface. □

Theorem 1579.3 (LimitPoint Bolzano–Weierstrass handoff)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *cluster consumer supplied by a* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source and the regular-subsequence route of chapter 381 factors through an accepted* $\mathsf{LimitPoint}^{\uparrow}$ *packet. In the notation of Definition 1579.1, the*

consumer reads the finite route

$$S, W, D, \Phi \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the Bolzano–Weierstrass cluster row can be consumed as a LimitPoint surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *without importing host completeness, quotient streams, open-cover compactness, or a choice-selected subsequence.*

Proof. Start with the bounded source row. The $\mathsf{BoundedRealSequence}^{\uparrow}$ packet of chapter 1415 supplies the source sequence surface consumed by the $\mathsf{BolzanoWeierstrass}^{\uparrow}$ carrier.

Use the Bolzano–Weierstrass route of chapter 1254. Its cluster consumer first reads finite interval and sequence data, then passes through the regular-subsequence route, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ cluster seal.

Repack that same downstream surface as S, W, D, Φ, Q, E. The source sequence becomes S, the inspected source and selected windows become W, the neighborhood mesh radii become D, the regular-subsequence evidence becomes Φ, the regular rational handoff becomes Q, and the real cluster row becomes E.

The structural rows H, C, P, N then transport, replay, source, and name this finite route as a $\mathsf{LimitPoint}^{\uparrow}$ packet. Since the original Bolzano–Weierstrass consumer exposes no host completeness, quotient stream equality, open-cover compactness, or arbitrary subsequence choice, the factored LimitPoint packet exposes none. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitPoint}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LimitPoint}^{\uparrow}}$, seed)

Scope closed: Definition 1579.1, Definition 1579.2, and Definition 1579.3 seed the finite cluster-point interface from real-sequence windows through cofinal subsequence evidence, dyadic neighborhoods, regular rational readback, and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LimitPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host completeness, quotient stream equality, open-cover compactness, arbitrary choice-selected subsequences, or least cluster-point selection.

Upgrade path: Obligation closure requires checked carrier admission, neighborhood coverage, cofinal-window persistence, Real seal routing, ledger non-escape, and Bolzano–Weierstrass handoff for the same packet.

Constructive story: A $\mathsf{LimitPoint}^{\uparrow}$ packet is generated from a source real sequence, finite StreamName windows, DyadicRatCore neighborhood radii, RegSeqRat readback, a cofinal subsequence witness, a RealUp cluster seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1580

A Concrete Naming Certificate for $\mathsf{RealSequenceClusterPoint}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ packet records the finite carrier surface for a cluster point of a real sequence between the displayed limit interface of chapter 1413 and the bounded-subsequence extraction route of chapter 1254. It receives its sequence source from chapter 1415, its cofinal subsequence route from chapter 381, and its downstream compactness consumer from chapter 1254. The packet displays a local cluster seal without importing host subsequence choice, quotient equality of streams, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1580.1 (Real sequence cluster point carrier)**.** *A* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, J, W, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-sequence row, read only through the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *surface when a bounded source is required.*
2. *J is the cofinal subsequence row, carrying the monotone index schedule and tail-access route supplied by the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *packet.*
3. *W is the finite family of selected* $\mathsf{StreamName}^{\uparrow}$ *windows for the source entries named by J and for the proposed cluster real.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those selected windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the inspected cluster comparisons.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *cluster seal reached after the selected windows, regular readbacks, and dyadic tolerance rows have been displayed.*
7. *H stores componentwise* hsame *transport for the source sequence, cofinal subsequence route, selected windows, regular readbacks, tolerance ledger, cluster seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *routes from source indices to selected windows, from selected windows to regular readback, from regular readback to the dyadic tolerance ledger, and from the ledger to the* $\mathsf{Real}^{\uparrow}$ *cluster seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealSequenceClusterPoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, J, W, Q, D, R, H, C, P, N$. It has no coordinate for host subsequence choice, quotient-stream equality, full sequence convergence, compactness of arbitrary covers, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1580.2 (Real sequence cluster point NameCert obligations)**.** *For every accepted* RealSequenceClusterPoin *carrier $K = (S, J, W, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealSequenceClusterPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, source-sequence access through S, cofinal subsequence access through J, selected* $\mathsf{StreamName}^{\uparrow}$ *window coverage through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, dyadic cluster tolerance accounting through D, the* $\mathsf{Real}^{\uparrow}$ *cluster seal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealSequenceClusterPointUp.RealSequenceClusterPointTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1580.1. The only sequence data available to the classifier are the source row S, the cofinal subsequence row J, the selected stream windows W, the regular rational readback Q, and the tolerance ledger D.

The cluster seal is then read through R. This places the proposed cluster point behind the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows rather than behind a host convergence predicate.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 1580.3 (Real sequence cluster point subsequence handoff)**.** *Every accepted* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *carrier factors its cluster-point read through the finite route*

$$S \longmapsto J \longmapsto W \longmapsto Q \longmapsto D \longmapsto R$$

together with H, C, P, N. Consequently a consumer may pass from the displayed sequence-limit surface of chapter 1413 to the cofinal subsequence route of chapter 381 without importing host subsequence choice or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealSequenceClusterPointUp.RealSequenceClusterPointSubsequence handoff`

Proof. Start with the source sequence row S. The cofinal row J restricts attention to the displayed subsequence indices supplied by the regular Cauchy subsequence route.

Next read the selected windows W and their regular rational readback Q. The dyadic ledger D records only the inspected cluster comparisons for those windows, so the route remains finite at every observation level.

Finally consume the real cluster seal R and the structural rows H, C, P, N. The handoff therefore uses the subsequence certificate as a displayed carrier row and exports no host-level selector or completeness principle. □

Theorem 1580.4 (Real sequence cluster point public export)**.** *The* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer of chapter 1254 may read a cluster candidate from an accepted* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *carrier only through the bounded source row of chapter 1415, the cofinal subsequence row of chapter 381, and the displayed route*

$$W, Q, D, R, H, C, P, N.$$

Thus the exported surface supplies a local cluster-point certificate for the real-and-completion boundary while leaving host subsequence choice, quotient-stream equality, open-cover compactness, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ *completeness outside the packet.*

Proof. Use Definition 1580.3. It places every cluster read on the route from the source sequence through the cofinal subsequence row, selected windows, readback, dyadic ledger, and real cluster seal.

The bounded source row is read through chapter 1415. The downstream $\mathsf{BolzanoWeierstrass}^{\uparrow}$ carrier can therefore consume a bounded sequence source, a cofinal subsequence surface, and a cluster seal without receiving an independent compactness or choice coordinate.

The remaining rows H, C, P, N preserve and name the displayed route. Since the carrier has no coordinate for host subsequence choice, quotient-stream equality, open-cover compactness, trichotomy, or ambient real completeness, none of those principles is part of the public export. □

Theorem 1580.5 (Real sequence cluster point window localization)**.** *Every accepted* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *packet exposes its cluster seal only through the bounded source row, the regular Cauchy subsequence route, and the displayed real sequence limit surface. For*

$$K = (S, J, W, Q, D, R, H, C, P, N),$$

each cluster read factors through

$$S \longmapsto J \longmapsto W, Q, D \longmapsto R \longmapsto H, C, P, N$$

where S *is read through* $\mathsf{BoundedRealSequence}^{\uparrow}$*,* J *is read through* $\mathsf{RegularCauchySubsequence}^{\uparrow}$*, and* R *is the displayed* $\mathsf{Real}^{\uparrow}$ *cluster seal. The exposed surface gives no ambient compactness theorem, no host subsequence choice, and no independent* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Unpack the carrier rows $S, J, W, Q, D, R, H, C, P, N$ from Definition 1580.1. The bounded source is available only as the displayed source row S, and when boundedness is required that row is read through chapter 1415.

Use the subsequence handoff of Definition 1580.3. It routes the cluster read through the cofinal subsequence row J, the selected windows W, the regular readback row Q, and the dyadic tolerance ledger D before the real cluster seal R is visible.

Now apply the public export of Definition 1580.4. The export gives a Bolzano–Weierstrass consumer only the bounded source row, cofinal subsequence surface, selected windows, readback, tolerance ledger, cluster seal, and structural rows, so no ambient compactness, host subsequence selector, or independent real-completeness coordinate is delivered. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ cluster-candidate read factors through a RealSequenceClusterPo route displaying a bounded source row, cofinal subsequence row, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp cluster seal, transport, continuation, provenance, and local naming.

Independence witness: real-sequence-limit, regularcauchysubsequence, boundedrealsequence, bolzanoweierstrass: $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source sequence access, cofinal subsequence access, selected windows, regular readback, tolerance accounting, cluster sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceClusterPoint}^{\uparrow}}$.

Constructive story: A RealSequenceClusterPoint$^{\uparrow}$ packet is generated from a source real-sequence row, a cofinal subsequence row, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp cluster seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealSequenceClusterPoint}\uparrow}$, seed)

Scope closed: Definition 1580.1, Definition 1580.2, Definition 1580.3, and Definition 1580.4 bind the finite cluster-point certificate surface between displayed sequence limits and Bolzano-Weierstrass.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealSequenceClusterPoint}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host subsequence choice, quotient equality of streams, open-cover compactness, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, subsequence handoff, public export, and TasteGate field faithfulness for BEDC.Derived.RealSequenceClusterPointUp.

Chapter 1581

A Concrete Naming Certificate for $\mathsf{RealSeries}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealSeries}^{\uparrow}$ packet records the finite partial-sum and Cauchy-tail surface for a real-valued term stream. It sits after the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the displayed limit interface of chapter 1413, the bounded Cauchy schedule of chapter 1522, the regular rational rows of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names a series only through displayed finite sums and a shared tail threshold; it does not import host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1581.1 (Real series carrier)**.** *A* $\mathsf{RealSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, W, Q, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the real-valued term stream, read only through selected* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows for the displayed terms.*
2. *S is the partial-sum window row, listing the finite initial indices, their accumulated* $\mathsf{Real}^{\uparrow}$ *sum seals, and the adjacent difference reads induced by T.*
3. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows used for the terms and for the displayed partial sums.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every selected term and partial-sum window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records finite addition errors, comparison radii, and tail-request tolerances.*
6. *M is the shared Cauchy-tail threshold row: one displayed threshold governs every partial-sum comparison requested by the packet.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after the term windows, partial-sum windows, regular readbacks, dyadic ledger, and shared tail threshold have been displayed.*
8. *H stores componentwise* hsame *transport for the term stream, partial-sum window, selected stream windows, regular readbacks, dyadic ledger, tail threshold, endpoint seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from term reads to partial sums, from partial sums to regular readback, from readback to the dyadic ledger, from the ledger to the shared tail threshold, and from that threshold to the endpoint seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, W, Q, D, M, E, H, C, P, N$. It has no coordinate for a host infinite sum, selected sequence limit, quotient equality of real streams, countable-choice selector, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Definition 1581.2 (RealSeries term carrier surface)**.** *The root term carrier surface of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is the displayed finite route*

$$T, S, W, Q$$

from Definition 1581.1: T is the real-valued term stream, S is the finite partial-sum ledger, W is the selected $\mathsf{StreamName}^{\uparrow}$ *window family, and Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed term and partial-sum windows. This surface carries only named term observations and regular rational readback; it does not assert convergence outside the displayed Cauchy criterion rows.*

Theorem 1581.3 (Real series NameCert obligations)**.** *For every accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier $K = (T, S, W, Q, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, term-stream access through T, partial-sum window coverage through S, selected* $\mathsf{StreamName}^{\uparrow}$ *windows through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, dyadic tolerance accounting through D, shared Cauchy-tail thresholding through M,* $\mathsf{Real}^{\uparrow}$ *endpoint sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesUpTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1581.1. The only term data available to the classifier are the stream row T, the partial-sum window row S, the selected stream windows W, the regular rational readback Q, the dyadic ledger D, and the shared tail threshold M.

The endpoint seal is then read through E. This places every real-facing consumer behind the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and tail-threshold rows rather than behind an external infinite-sum predicate.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 1581.4 (RealSeries Cauchy majorant obligation)**.** *For every accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier $K = (T, S, W, Q, D, M, E, H, C, P, N)$, any majorant or Cauchy-criterion read must pass through the dyadic tolerance ledger D, the shared tail threshold M, and the endpoint seal E after the term, partial-sum, window, and readback rows T, S, W, Q have been displayed. The route exports no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeries cauchy majorant obligation`

Proof. Project the carrier of Definition 1581.1. The rows D, M, E are the only rows that record tolerance comparison, shared tail control, and the endpoint seal.

Use Definition 1581.3. It places those rows behind T, S, W, Q and before the structural rows H, C, P, N, so a majorant or Cauchy-criterion consumer receives a finite route.

The excluded host-side objects are not coordinates of K. Hence a majorant read cannot leave D, M, E for an external convergence principle. □

Theorem 1581.5 (RealSeries uniform Cauchy obligation)**.** *Every* $\mathsf{UniformCauchySeries}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier inspects only the finite rows* T, S, W, Q, D, M *before any endpoint or structural row is read. Uniformity is therefore a displayed shared-tail route, not a family of externally selected thresholds.*

Proof. Read the rows T, S, W, Q, D, M from the carrier. They expose the term stream, partial-sum windows, selected stream windows, regular readback, dyadic tolerances, and the shared threshold.

Apply Definition 1581.4. It forces Cauchy-style consumers through the same tolerance and threshold rows before E can be consumed.

Since the carrier has no countable-choice schedule or selected family of thresholds, the uniform Cauchy read is exhausted by the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeries uniform cauchy obligation`

Theorem 1581.6 (RealSeries RegSeqRat tail obligation)**.** *For every accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is required before any tail comparison can pass to the dyadic ledger* D*, the shared threshold* M*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. No Real endpoint seal may be inspected from a partial-sum row without this regular rational readback.*

Proof. Use the carrier order in Definition 1581.1. The row Q is the only regular rational readback for selected terms and partial-sum windows.

The dyadic ledger D compares displayed readbacks, the threshold row M governs the tail after those comparisons, and the endpoint row E is downstream of both. Thus Q is mandatory before any Real endpoint seal.

The structural rows H, C, P, N preserve and name the same route. They do not add another readback row or a direct endpoint channel. □

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesRegSeqRatTailObligation`

Theorem 1581.7 (RealSeries partial-sum ledger nonescape)**.** *Every partial-sum read from an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is routed only through the finite term carrier surface of Definition 1581.2 and the displayed continuation replay row* C*. The ledger may inspect* T, S, W, Q *and may replay the carried term-to-sum route, but it cannot bypass those rows to a host infinite sum, selected limit, quotient equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeries partial sum ledger nonescape`

Proof. Project the carrier in Definition 1581.1. The rows T, S, W, Q are the only rows that expose terms, finite partial sums, selected windows, and regular rational readback.

The continuation row C replays the route from term reads to partial sums and from partial sums to regular readback. Because the carrier has no other partial-sum coordinate, a ledger read cannot leave this finite surface.

The excluded host-side objects are not fields of the carrier. Hence partial-sum accounting remains inside the displayed term ledger and continuation replay. □

Theorem 1581.8 (Real series partial-sum window)**.** *Every accepted* $\mathsf{RealSeries}^{\uparrow}$ *read factors each displayed finite series observation through*

$$T \longmapsto S \longmapsto W \longmapsto Q \longmapsto D$$

together with H, C, P, N*. The row* S *records only finite partial sums and adjacent term contributions, and the row* D *records only the dyadic tolerances needed to compare those displayed windows.*

Proof. Start with a requested series observation. By Definition 1581.1, the request first reads the term stream row T and then the finite partial-sum window row S.

The selected windows W and regular readbacks Q expose the terms and partial sums visible to the packet. The dyadic ledger D is the only row that compares those finite reads and records the corresponding addition and tolerance information.

The rows H, C, P, N preserve, replay, source, and name the same route. Therefore the observation remains a finite partial-sum window and exports no host infinite-sum object or quotient comparison. □

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesPartialSumWindowRoute`

Theorem 1581.9 (Real series nonescape boundary)**.** *Every endpoint-facing read from an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier passes through the shared route*

$$T \longmapsto S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M \longmapsto E$$

together with H, C, P, N*. Consequently a consumer may use the displayed interfaces of chapter 1413 and chapter 1522 for finite partial-sum and Cauchy-tail accounting, but it receives no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 1581.8 to place every visible series observation on the finite route through T, S, W, Q, D. The shared threshold row M is then the only tail-control coordinate available to compare later partial sums.

The endpoint row E consumes this already displayed route. It is a $\mathsf{Real}^{\uparrow}$ seal for the packet endpoint, not a separate assertion that a host sequence of partial sums has been completed.

Finally read the structural rows H, C, P, N. Because the carrier has no coordinate for host infinite sums, selected limits, quotient equality, countable choice, or ambient real completeness, none of those principles can escape through the endpoint-facing boundary. □

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesNonescapeBoundary`

Theorem 1581.10 (Real series tail-threshold stability)**.** *Let* $K = (T, S, W, Q, D, M, E, H, C, P, N)$ *be an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier. If* $K' = (T', S', W', Q', D', M', E', H', C', P', N')$ *is obtained by componentwise* hsame *transport along the rows recorded in* H*, then the single shared Cauchy-tail threshold row is preserved as the comparison gate for later partial sums: every transported partial-sum comparison still factors through*

$$T, S, W, Q, D, M, H, C, P, N.$$

In particular, no transported row may replace the threshold coordinate by a family of unrelated tail choices or by a host infinite-sum witness.

Proof. Project the carrier inventory in Definition 1581.1. The row M is declared as the shared Cauchy-tail threshold, and the row H stores componentwise hsame transport for the term stream, partial-sum window, selected windows, readbacks, ledger, threshold, endpoint, continuation, provenance, and local naming rows.

Use Definition 1581.8 to place every visible finite comparison through T, S, W, Q, D before any tail control is read. Transporting the displayed rows componentwise changes only their names inside the same hsame class; it does not add a new coordinate to the classifier.

The threshold coordinate available after transport is therefore still the transported instance of the single row M. The structural rows C, P, N replay, source, and name that same route, so later partial-sum comparisons remain governed by the shared threshold surface. □

Definition 1581.11 (Real series shared-tail consumer surface)**.** *The shared-tail consumer surface of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier is the finite row package*

$$(S, D, M, E, H, C, P, N),$$

where S is the partial-sum window row, D is the dyadic tolerance ledger, M is the shared Cauchy-tail threshold, E is the endpoint $\mathsf{Real}^{\uparrow}$ *seal, H records componentwise* hsame *transport, C records displayed* Cont *replay, P records* Pkg *provenance, and N records the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *name. A consumer of this surface may inspect only those rows together with the term and readback rows already displayed in Definition 1581.1; it receives no selected limit, quotient equality of partial-sum streams, countable-choice schedule, or ambient completeness witness.*

Theorem 1581.12 (Real series endpoint nonescape obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$ *endpoint read through the row E of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier must first consume the finite partial-sum window, regular readback, dyadic ledger, and stabilized shared threshold. Its public handoff route is*

$$T \longmapsto S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M \longmapsto E,$$

together with H, C, P, N; it cannot bypass those rows by appealing directly to a selected limit, quotient equality of partial-sum streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1581.10 to fix the shared threshold row as the only tail-control surface stable under componentwise transport. This places any later endpoint request behind the same M row that governs the partial-sum comparisons.

Apply Definition 1581.9 to the endpoint-facing read. The theorem already forces the route through T, S, W, Q, D, M, E and records that the endpoint seal exports no host infinite-sum or selected-limit principle.

The handoff to $\mathsf{Real}^{\uparrow}$ is therefore a seal after the finite window, regular readback, dyadic ledger, and shared threshold have all been displayed. The rows H, C, P, N preserve the same route and do not create an alternate endpoint channel. □

Theorem 1581.13 (Real series Cauchy consumer obligation)**.** *The rows S, D, M, E, H, C, P, N of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier provide exactly the finite Cauchy surface required by downstream* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*, and* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *consumers: S supplies displayed partial-sum windows, D supplies dyadic comparison radii and finite addition errors, M supplies the shared tail threshold, E supplies the endpoint seal after that threshold, and H, C, P, N transport, replay, source, and name the same route. This consumer obligation adds no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read S, D, M, E, H, C, P, N from Definition 1581.11. The row S is the only partial-sum window surface, D is the only dyadic tolerance ledger, M is the only shared tail-threshold row, and E is the endpoint seal downstream of those rows.

Definition 1581.10 keeps the threshold common under componentwise transport, while Definition 1581.12 ensures that endpoint access consumes the finite window, readback, ledger, and threshold before reaching E.

Thus the downstream Cauchy-style consumers receive a finite criterion surface rather than a completed host sum. Since the classifier has no coordinate for selected limits, quotient equality, countable choice, or ambient real completeness, none of those principles is introduced by the readiness package. □

Definition 1581.14 (Real series tail-majorant carrier)**.** *A* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier is a finite* BHist *packet*

$$J = (W, R, T, B, U, G, H, C, P, N)$$

whose rows are read only as follows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family for the displayed tail indices and partial-sum comparison windows.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the terms, partial sums, and displayed majorant entries.*
3. *T is the tail-request ledger inherited from the* $\mathsf{RealSeries}^{\uparrow}$ *carrier of Definition 1581.1.*
4. *B is the displayed tail-majorant row, recording finite upper bounds for the requested tails.*
5. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff row, read only through W, R, T, B.*
6. *G is the* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *route row, exposing the same finite majorant data to Cauchy-criterion consumers.*
7. *H, C, P, N are componentwise transport, displayed replay, package provenance, and local naming rows for W, R, T, B, U, G.*

The classifier has no coordinate for a host infinite sum, an unbounded majorant family, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1581.15 (RealSeries root-unblock obligation surface)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier together with a tail-majorant carrier*

$$J = (W, R, T, B, U, G, H, C, P, N),$$

the root-unblock $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *surface consists exactly of finite* BHist *obligations for the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the inherited tail-request ledger T, the tail-majorant row B, the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff U, the* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *route G, and the structural rows H, C, P, N. No obligation is stored as a host infinite series, a selected limit, an unbounded majorant, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the two carriers. The base $\mathsf{RealSeries}^{\uparrow}$ carrier supplies the finite term, partial-sum, tolerance, threshold, and $\mathsf{Real}^{\uparrow}$ endpoint rows; Definition 1581.14 supplies only $W, R, T, B, U, G, H, C, P, N$.

The root-unblock obligations are therefore the displayed finite rows that downstream consumers can actually read. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback row expose the terms, partial sums, and majorant entries; the tail ledger and majorant row expose the requested Cauchy bounds.

The handoff rows U and G consume the same finite data for $\mathsf{UniformCauchySeries}^{\uparrow}$ and $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$. The structural rows preserve and name that route, so no excluded host-side principle has an available coordinate. □

Theorem 1581.16 (RealSeries root tail window)**.** *Every accepted* $\mathsf{RealSeries}^{\uparrow}$ *root-unblock packet exposes its tail information through finite* BHist *windows before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is read. Concretely, a downstream consumer first reads the displayed* $\mathsf{RealSeries}^{\uparrow}$ *term route, the* $\mathsf{StreamName}^{\uparrow}$ *window rows, the partial-sum ledger, and the dyadic tolerance row carried by the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *packet; only after those rows have been transported by* hsame*, replayed by* Cont*, and sourced by* Pkg *may the consumer inspect a regular rational readback or endpoint seal.*

Proof. Project the base carrier of Definition 1581.1. The term stream T, partial-sum row S, selected windows W, and dyadic ledger D are the first readable tail rows of the packet.

The BHist route therefore exposes finite windows before completion-facing data. The $\mathsf{StreamName}^{\uparrow}$ rows name the visible indices, and the local NameCert row records that these are the only windows a root-unblock consumer may use.

The structural rows H, C, P, N transport, replay, source, and name the same route. Thus $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal are downstream of the displayed finite tail windows, not sources of hidden tail choices. □

Theorem 1581.17 (RealSeries majorant route)**.** *Every comparison or majorant read from an accepted* $\mathsf{RealSeries}^{\uparrow}$ *root-unblock packet factors through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *rows before it reaches a Cauchy-facing consumer. The route reads dyadic tolerances and finite majorant data from the displayed ledger, then reads regular rational windows, and only then exposes the* $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*, or* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *handoff.*

Proof. Start with the tail-majorant carrier of Definition 1581.14. The rows W, R, T, B record finite windows, regular readback, inherited tail requests, and displayed majorant bounds.

The comparison side is supplied by the dyadic ledger in the base carrier and by the majorant row B in the tail carrier. These rows are named locally by the $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ surface before any downstream Cauchy-facing row can be inspected.

Finally apply the handoff routes Definition 1581.23, Definition 1581.25, and Definition 1581.27. Each route consumes the same dyadic, regular-readback, local-certificate, transport, replay, and provenance rows before exposing its consumer. □

Theorem 1581.18 (RealSeries uniform tail budget)**.** *Every uniform tail-budget consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet receives a single displayed finite window schedule. The schedule is the common* BHist *route through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the shared tail threshold,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *rows; it is not a selected infinite sum, not a choice of infinitely many tail indices, and not an ambient completeness witness.*

Proof. Read the root carrier budget in Definition 1581.28. The finite rows $T, S, W, Q, D, M, E, H, C, P, N$ and the tail-majorant rows $W', R, T', B, U, G, H', C', P', N'$ are the only rows available to a uniform tail-budget consumer.

Use Definition 1581.31 to isolate the uniform route. It identifies the visible budget as a displayed finite tuple, so the consumer receives one schedule rather than a family selected outside the packet.

The transport, replay, provenance, and local naming rows preserve that same schedule. Since neither carrier contains a host infinite sum, countable-choice row, selected limit, or ambient completeness primitive, no such object can be exported by the uniform budget. □

Theorem 1581.19 (RealSeries root Cauchy-tail budget)**.** *Every* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, *and* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet must read the shared finite route*

$$T, S, W, Q, D, M, E, H, C, P, N$$

before a Cauchy-tail conclusion is exported. The route exposes the term stream, partial-sum window, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, shared threshold, endpoint seal, transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *row as one budget surface.*

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesRootCauchyTailBudget`

Proof. First apply Definition 1581.16. It places every tail-facing read behind the finite term route, selected windows, partial-sum ledger, and dyadic tolerance row before any regular rational readback or endpoint seal can be inspected.

Next use Definition 1581.17 and Definition 1581.18. These results force majorant, uniform, criterion, and product-facing consumers to share the same displayed dyadic, regular-readback, threshold, transport, replay, provenance, and local naming rows.

Finally apply Definition 1581.13. The downstream consumer receives a finite Cauchy surface and not a host infinite sum, selected limit, quotient equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness. □

Theorem 1581.20 (RealSeries root series criterion surface)**.** *Every* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, *and* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet reads the root criterion surface only through*

$$T, S, W, Q, D, M, E, H, C, P, N.$$

Here T *is the term stream row,* S *is the finite partial-sum window row,* W *is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row,* Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* M *is the shared tail threshold,* E *is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal, and* H, C, P, N *are the transport, replay, provenance, and local naming rows. The surface exports no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1581.1. The rows $T, S, W, Q, D, M, E, H, C, P, N$ are already displayed as the root route, with the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback visible before the endpoint seal.

Route criterion, uniform, majorant, and product requests through Definition 1581.19. The shared threshold row M is consumed after D and before E, so every endpoint-facing read reaches the same finite budget row.

The support rows H, C, P, N then preserve, replay, source, and name the displayed route. Since the accepted packet contains none of the excluded objects, no downstream criterion can export them. □

Definition 1581.21 (RealSeries root summability surface)**.** *The root summability surface of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is the finite* BHist *subpacket*

$$\mathsf{RootSum}(T, S, W, Q, D, M, E, H, C, P, N)$$

whose rows are exactly the term stream, partial-sum window, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, shared Cauchy-tail threshold,* $\mathsf{Real}^{\uparrow}$ *endpoint seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *evidence. A*

consumer may cite this surface as summability evidence only by reading those rows in that order; it is not a host summation operator, a selected limiting value, quotient equality of streams, countable choice, or ambient completeness.

Theorem 1581.22 (RealSeries root product readiness surface)**.** *Every* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer that asks an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet for product-ready data receives the root summability surface of Definition 1581.21 and no larger analytic object. The handoff cites* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, *and* $\mathsf{CauchyProduct}^{\uparrow}$ *only through*

$$T, S, W, Q, D, M, E, H, C, P, N,$$

with the shared threshold row M fixed before endpoint sealing.

Proof. Start from the root summability surface. It packages the finite term stream, partial-sum windows, selected stream windows, regular readback, dyadic ledger, threshold, endpoint, transport, replay, provenance, and naming rows in one subpacket.

Apply Definition 1581.20. The criterion-facing and uniform Cauchy consumers have already been routed through the same threshold row before E, so the product consumer cannot request a separate limit or completion coordinate.

Finally use Definition 1581.19. It includes the $\mathsf{CauchyProduct}^{\uparrow}$ consumer among the downstream reads and concludes that no host infinite sum, selected limit, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness is exported. □

Theorem 1581.23 (RealSeries uniform Cauchy series handoff)**.** *Every* $\mathsf{UniformCauchySeries}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier factors through the finite route*

$$W \longmapsto R \longmapsto T \longmapsto B \longmapsto U \longmapsto H, C, P, N.$$

The route supplies $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a displayed tail-request ledger, and a finite tail-majorant row before the uniform Cauchy handoff is exposed. It does not supply a host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a $\mathsf{UniformCauchySeries}^{\uparrow}$ consumer request. The request first reads W, so the visible indices and partial-sum comparisons remain inside finite $\mathsf{StreamName}^{\uparrow}$ windows.

The readback row R then exposes the regular rational data for those windows, and T records which tail comparisons are being requested. The majorant row B is the only bound source for those requests.

The handoff row U is downstream of W, R, T, B. Transport, replay, provenance, and local naming by H, C, P, N preserve that same route and do not add any host completion or selection coordinate. □

Theorem 1581.24 (RealSeries Cauchy criterion handoff)**.** *A downstream* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet receives only the displayed rows named by the carrier and tail-majorant route:* $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. The handoff records the finite Cauchy surface needed by the criterion and does not claim convergence outside the displayed Cauchy criterion.*

Proof. Project the base carrier. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are the only term-window, readback, and tolerance surfaces available before the shared threshold and endpoint rows.

Project the tail-majorant route when a criterion consumer is present. The $\mathsf{CauchyMajorantSeries}^{\uparrow}$ and $\mathsf{UniformCauchySeries}^{\uparrow}$ rows are read as finite handoff rows, while hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserve, replay, source, and name the same displayed route.

No carrier row supplies a host convergence predicate or selected limit. The $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ consumer therefore receives exactly the finite BEDC rows listed in the statement. □

Theorem 1581.25 (RealSeries Cauchy criterion route)**.** *Every* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier factors through the finite route*

$$W \longmapsto R \longmapsto T \longmapsto B \longmapsto G \longmapsto H, C, P, N.$$

Combined with the $\mathsf{Real}^{\uparrow}$ *endpoint row of Definition 1581.1, this route gives Cauchy-criterion consumers a finite partial-sum and tail-majorant surface rather than a completed host sum. The route does not claim convergence of all series, selected limits, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the tail-majorant carrier. The windows W and readback row R expose only finite partial-sum and majorant observations, while T and B identify the displayed tail requests and their finite bounds.

The criterion row G consumes those rows as the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ route. It does not bypass the finite windows or create a separate convergence predicate.

The endpoint boundary remains the $\mathsf{Real}^{\uparrow}$ row already named by Definition 1581.1. Thus a criterion consumer receives a finite route to the endpoint-facing surface, with H, C, P, N preserving the same data and no excluded host-side principle exported. □

Theorem 1581.26 (RealSeries root source carrier)**.** *The root-facing source surface of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is the combined finite carrier*

$$K = (T, S, W, Q, D, M, E, H, C, P, N), \qquad J = (W', R, T', B, U, G, H', C', P', N')$$

from Definition 1581.1 and Definition 1581.14. Its readable rows are exactly the real-valued term stream, finite partial-sum windows, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, shared tail threshold,* $\mathsf{Real}^{\uparrow}$ *endpoint seal, inherited tail-request ledger, displayed majorant row,* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff,* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *route, and the structural transport, replay, provenance, and naming rows.*

Proof. Project the base carrier K. It exposes the term row T, partial-sum row S, selected windows W, regular readback Q, dyadic ledger D, shared threshold M, endpoint seal E, and structural rows H, C, P, N.

Project the tail-majorant carrier J. It adds only finite windows W', readback R, inherited tail requests T', majorant row B, uniform Cauchy handoff U, criterion route G, and structural rows H', C', P', N'.

The combined source therefore has no host infinite sum, selected limit, quotient equality, countable-choice selector, or ambient completeness coordinate. Every root-facing consumer must read one of the displayed rows. □

Theorem 1581.27 (RealSeries Cauchy majorant handoff)**.** *Every* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *obligation read from an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier factors through the finite route*

$$W', R, T', B \longmapsto U, G \longmapsto H', C', P', N'.$$

The majorant row B is the only source of tail bounds, and it is read together with $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the inherited tail-request ledger, the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff, and the* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *route before any downstream series consumer may inspect the certificate.*

Proof. Read the tail-majorant carrier. The finite windows W' determine which tail and partial-sum comparisons are visible, while R supplies the regular-rational readback for those observations.

The inherited ledger T' states the requested tail bounds, and B is the only displayed row that can answer them. Thus a Cauchy majorant consumer has no unbounded majorant family or external convergence theorem available.

The handoff rows U and G consume the same W', R, T', B surface. Transport, replay, provenance, and naming through H', C', P', N' preserve that finite route for later uniform Cauchy and Cauchy-criterion consumers. □

Theorem 1581.28 (RealSeries root carrier budget)**.** *The root-facing carrier budget of any accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is exactly the displayed base rows*

$$T, S, W, Q, D, M, E, H, C, P, N$$

of Definition 1581.1 together with the tail-majorant rows

$$W', R, T', B, U, G, H', C', P', N'$$

of Definition 1581.14. The budget is exhausted by the partial-sum window route of Definition 1581.8, the shared threshold route of Definition 1581.10, and the endpoint nonescape route of Definition 1581.9; it contains no host infinite-sum coordinate, selected limit, quotient equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the base carrier. Its readable rows are exactly the term stream T, partial-sum windows S, selected windows W, regular readback Q, dyadic ledger D, shared threshold M, endpoint seal E, and structural rows H, C, P, N.

Project the tail-majorant carrier. Its readable rows are exactly the windows W', readback R, inherited tail ledger T', majorant row B, uniform handoff U, Cauchy-criterion route G, and structural rows H', C', P', N'.

The cited partial-sum, threshold-stability, and endpoint nonescape theorems show that every downstream read remains inside those two finite row lists. Since neither carrier has any other coordinate, the root-facing budget is exactly the displayed surface. □

Theorem 1581.29 (RealSeries tail-ledger majorant)**.** *Every tail-majorant consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet must read the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows W', the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the inherited tail-request ledger T', and the finite majorant row B before any* $\mathsf{UniformCauchySeries}^{\uparrow}$ *or* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *handoff. The handoff route is the one displayed in Definition 1581.14, Definition 1581.23, and Definition 1581.25.*

Proof. Start from a tail-majorant consumer request. The carrier definition exposes windows W' before readback, so the requested tail and partial-sum comparisons are fixed as finite $\mathsf{StreamName}^{\uparrow}$ observations.

The row R then supplies the regular-rational data, and the inherited ledger T' records which tail comparisons are being asked for. The row B is the only displayed majorant source for those requests.

Only after these four rows have been read can the handoff rows U and G be inspected. The uniform Cauchy and Cauchy-criterion route theorems preserve that same route through H', C', P', N' and add no external completion principle. □

Theorem 1581.30 (RealSeries Cauchy criterion scope)**.** *Downstream* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, *and* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *consumers receive only the finite surface displayed by Definition 1581.1 and Definition 1581.14: finite term and partial-sum windows, regular readback, dyadic tolerance accounting, the shared threshold, the inherited tail ledger, the majorant row, handoff rows, componentwise transport, replay, provenance, and local naming. They receive no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 1581.28 to place every root-facing read inside the two displayed finite carriers. This fixes the base series rows and the tail-majorant rows before any downstream Cauchy-style consumer is considered.

Use Definition 1581.29 for the tail-facing route. It forces the windows, readback row, inherited ledger, and majorant row to be read before the uniform Cauchy and Cauchy-criterion handoffs.

Use Definition 1581.27 for the majorant consumer. Together these routes exhaust the consumer surface, and the omitted host-side objects are absent from the carrier inventory. □

Theorem 1581.31 (RealSeries uniform Cauchy budget)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier* $J = (W', R, T', B, U, G, H', C', P', N')$*, the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *budget is exactly the displayed tuple* (W', R, T', B, U)*. It binds finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, tail requests, and Cauchy majorant bounds before the uniform Cauchy handoff is exposed, and it does not assert host convergence, selected limits, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project J and isolate the uniform Cauchy handoff row U. By Definition 1581.23, U is reached only after the finite windows, readback row, tail ledger, and majorant row have been displayed.

The tuple (W', R, T', B, U) is therefore the whole budget available to a uniform Cauchy consumer. The structural rows H', C', P', N' can transport, replay, source, and name the same budget, but they cannot add a host-side convergence principle. □

Theorem 1581.32 (RealSeries root ledger exactness)**.** *The root ledger of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *packet is exact for the Cauchy majorant, uniform Cauchy, and Cauchy-criterion routes: the dyadic ledger* D*, shared threshold* M*, tail-request ledger* T'*, majorant row* B*, endpoint seal* E*, uniform handoff* U*, criterion row* G*, and structural rows* $H, C, P, N, H', C', P', N'$ *are the only rows that may witness tail exactness for downstream* $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, *and* $\mathsf{UniformCauchySeries}^{\uparrow}$ *consumers.*

Proof. Project the base carrier and read the endpoint-facing rows D, M, E, H, C, P, N. These rows account for dyadic tolerance, the shared Cauchy-tail threshold, and the terminal Real seal.

Project the tail-majorant carrier and read $T', B, U, G, H', C', P', N'$. These rows account for tail requests, displayed majorant bounds, uniform Cauchy handoff, Cauchy-criterion routing, and the structural preservation of those routes.

No other ledger row exists in either carrier. Hence downstream bounded-sequence, sequence-limit, majorant-series, and uniform-Cauchy consumers receive exactly the displayed finite ledger and cannot appeal to ambient real equality or completeness. □

Theorem 1581.33 (RealSeries uniform Cauchy threshold coherence)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier*

$$J = (W', R, T', B, U, G, H', C', P', N'),$$

any $\mathsf{UniformCauchySeries}^{\uparrow}$ *read through* U *and any* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *read through* G *must use the same displayed* W', R, T'*, and* B *rows. No route may replace the shared threshold and majorant data by an unrelated tail selector, host infinite series, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read J through Definition 1581.14. The rows W', R, T', B are the only displayed finite windows, regular readbacks, inherited tail requests, and majorant bounds available to either downstream handoff.

Apply Definition 1581.31 to the $\mathsf{UniformCauchySeries}^{\uparrow}$ read. It identifies the uniform Cauchy budget as exactly (W', R, T', B, U), so the handoff U cannot choose a different window or threshold route.

Apply Definition 1581.25 to the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ read. It routes G through the same W', R, T', B surface before structural transport, replay, provenance, and naming are inspected.

The two consumer reads therefore share the same finite tuple. Since the carrier has no coordinate for a host infinite series, selected limit, quotient equality, countable-choice selector, or ambient real completeness, none of those objects can replace the displayed threshold data. □

Theorem 1581.34 (RealSeries majorant endpoint nonescape)**.** *Every endpoint-facing read that consumes* $\mathsf{CauchyMajorantSeries}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*, or* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *factors through the finite route*

$$W', R, T', B \longmapsto U, G \longmapsto H', C', P', N'$$

before any $\mathsf{Real}^{\uparrow}$ *endpoint seal from the base carrier may be cited. The route exports no host infinite sum, selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 1581.27 for the $\mathsf{CauchyMajorantSeries}^{\uparrow}$ read. The majorant obligation first exposes the finite windows, regular readback, inherited tail ledger, and displayed majorant row W', R, T', B.

Use Definition 1581.23 and Definition 1581.25 for the two downstream handoffs. Both routes consume the same finite majorant surface before the rows U and G are exposed.

Finally apply Definition 1581.32. The endpoint seal from the base carrier is cited only after the dyadic ledger, shared threshold, tail-request ledger, majorant row, uniform handoff, criterion row, and structural rows have been read.

Thus every endpoint-facing route factors through $W', R, T', B \longmapsto U, G \longmapsto H', C', P', N'$ before reaching the base $\mathsf{Real}^{\uparrow}$ seal. No excluded host-side principle has a carrier coordinate from which it could escape. □

Theorem 1581.35 (RealSeries shared majorant threshold coherence)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *tail-majorant carrier*

$$J = (W', R, T', B, U, G, H', C', P', N'),$$

every $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*, and* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *consumer reads the same displayed* W', R, T', B *threshold surface before its handoff row. The consumer may route through* U*,* G*, or the majorant handoff, but it cannot substitute a different tail selector, majorant ledger, host infinite sum, selected limit, quotient equality, countable-choice witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read J through Definition 1581.14. The rows W', R, T', B are the unique finite windows, regular readback, inherited tail-request ledger, and displayed majorant bounds in the accepted tail-majorant carrier.

For the uniform route, apply Definition 1581.31. For the criterion route, apply Definition 1581.25. For the majorant route, apply Definition 1581.27. Each cited handoff consumes the same W', R, T', B surface before exposing its consumer-specific row.

The structural rows H', C', P', N' transport, replay, source, and locally name that displayed surface. They do not create another threshold or another majorant source, so the three consumer routes remain coherent over the same finite BEDC rows. □

Theorem 1581.36 (RealSeries endpoint majorant factorization)**.** *Let $I = (S, D, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier and let $J = (W', R, T', B, U, G, H', C', P', N')$ be an accepted tail-majorant carrier for the same displayed series route. The* $\mathsf{Real}^{\uparrow}$ *endpoint row E of I factors first through S, D, M and then, whenever a majorant consumer is present, through the shared W', R, T', B surface before any downstream endpoint-facing read may cite the seal.*

Proof. Project I through Definition 1581.1. The stream row S, dyadic ledger D, and shared threshold row M are read before the endpoint seal E is available.

Now project J through Definition 1581.14. The rows W', R, T', B display the finite windows, regular readback, inherited tail requests, and majorant bounds that a majorant consumer must inspect.

Use Definition 1581.10 to identify the threshold side of the base route, and use Definition 1581.27 for the majorant side. The endpoint row cannot bypass either route because Definition 1581.34 excludes host infinite sums, selected limits, quotient equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the carrier inventory. □

Theorem 1581.37 (RealSeries Cauchy consumer readiness)**.** *The shared threshold coherence of Definition 1581.35 and the endpoint factorization of Definition 1581.36 supply the finite Cauchy surface consumed by the downstream* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*, and* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *chapters. The ready surface consists exactly of the base S, D, M, E route, the tail-majorant W', R, T', B route, the consumer rows U, G, and the structural rows $H, C, P, N, H', C', P', N'$.*

Proof. The coherence theorem fixes the threshold and majorant rows shared by the three Cauchy consumers. Thus every consumer begins from the same W', R, T', B surface rather than from a parameter chosen outside the packet.

The endpoint-factorization theorem ties that surface back to the base S, D, M, E route. Consequently a downstream endpoint read sees the real-series stream, dyadic ledger, shared threshold, endpoint seal, tail windows, regular readback, inherited tail requests, and displayed majorant bounds in one finite route.

Apply Definition 1581.32 to identify the remaining structural rows. These rows provide transport, continuation replay, provenance, and local naming for the same route, which is precisely the finite Cauchy surface needed by the realseriescauchycriterion, uniformcauchyseries, and cauchymajorantseries consumers. □

Theorem 1581.38 (RealSeries majorant-threshold readiness)**.** *Any* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*, or* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *consumer that treats an accepted* $\mathsf{RealSeries}^{\uparrow}$ *endpoint as series-ready must first read the same finite tuple*

$$S, D, M, E, W', R, T', B, U, G, H, C, P, N, H', C', P', N'.$$

Here S, D, M, E are the base stream, dyadic ledger, shared threshold, and endpoint seal rows; W', R, T', B are the majorant-window, regular-readback, inherited tail-request, and displayed bound rows; U, G are the uniform and criterion-facing consumer rows; and $H, C, P, N, H', C', P', N'$ are the transport, replay, provenance, and local naming rows for the base and tail-majorant packets. No consumer may certify readiness by appealing to a host infinite sum, a selected limit, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First apply Definition 1581.35. It forces the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, and $\mathsf{CauchyMajorantSeries}^{\uparrow}$ routes to read the same W', R, T', B threshold surface before their consumer-specific rows are visible.

Next apply Definition 1581.36. The endpoint seal E is downstream of the base rows S, D, M and, in the presence of a majorant consumer, downstream of the shared W', R, T', B surface.

Finally apply Definition 1581.37. It attaches the rows U, G and the structural rows $H, C, P, N, H', C', P', N'$ to the same finite route. Thus endpoint readiness is exactly the displayed tuple, with no coordinate for host infinite sums, selected limits, quotient equality, countable choice, or ambient Real completeness. □

Theorem 1581.39 (RealSeries uniform majorant tail coherence)**.** *Let*

$$K = (T, S, W, Q, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{RealSeries}^{\uparrow}$ *Cauchy-facing carrier whose uniform Cauchy tail and Cauchy majorant consumers are read through the same finite tail threshold. Then the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff, the* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *handoff, and the* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *handoff agree on the displayed rows $T, S, W, Q, D, M, E, H, C, P, N$: T supplies the term-stream row, S supplies the finite partial-sum windows, W supplies the selected* $\mathsf{StreamName}^{\uparrow}$ *windows, Q supplies* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, D supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and M supplies the shared Cauchy-tail threshold. No consumer may replace that threshold by a host infinite sum, a selected limit, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier to its base route. The rows T, S, W, Q, D, M give the term stream, partial-sum windows, selected stream windows, regular readback, dyadic tolerance ledger, and common tail threshold before any endpoint row is visible.

Apply Definition 1581.23 to the uniform consumer and Definition 1581.27 to the majorant consumer. Each handoff reads the same finite window and tolerance surface rather than choosing an external threshold.

Finally use Definition 1581.25 and Definition 1581.37. The criterion-facing route consumes the same threshold surface and the same structural rows H, C, P, N, so the three consumers are coherent over one finite Cauchy tail. □

Theorem 1581.40 (RealSeries RegSeqRat endpoint seal route)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier*

$$K = (T, S, W, Q, D, M, E, H, C, P, N),$$

every endpoint-facing $\mathsf{Real}^{\uparrow}$ *seal read factors through the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the shared threshold M before citing E. The route is then transported by H, replayed by C, sourced by P, and named by N; it exports no independent endpoint carrier, quotient real equality, selected limit, host infinite sum, countable-choice row, or ambient completeness principle.*

Proof. Begin with the endpoint side of the base carrier. By Definition 1581.12, the endpoint seal cannot be read without the finite partial-sum and threshold rows carried by the accepted packet.

Next attach the readback route. The shared-threshold statement Definition 1581.10 fixes M, and Definition 1581.36 ties the majorant-facing window surface back to the base endpoint row.

The structural rows close the route. Transport, continuation replay, package provenance, and the local NameCert row preserve the same displayed route from W, Q, D, M to E, so no separate real endpoint or host-level completeness object can be inserted. □

Theorem 1581.41 (RealSeries Cauchy majorant threshold exhaustion)**.** *Let*

$$K = (T, S, W, Q, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{RealSeries}^{\uparrow}$ *carrier and suppose a Cauchy majorant consumer reads it through the Cauchy-facing surface of Definition 1581.37. Then the majorant consumer is exhausted by the rows* $T, S, W, Q, D, M, E, H, C, P, N$ *together with the already displayed tail-majorant rows of Definition 1581.14. In particular, every requested tail bound is answered through the shared threshold* M*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before the* $\mathsf{Real}^{\uparrow}$ *endpoint seal is cited.*

Proof. First read the majorant consumer through Definition 1581.27. The handoff exposes finite tail windows, regular readback, inherited tail requests, and displayed majorant bounds; it does not add another threshold source.

Next use Definition 1581.39. The uniform Cauchy route, Cauchy criterion route, and majorant route all consume the same T, S, W, Q, D, M surface, so a requested bound cannot escape to an external sequence or a choice principle.

Finally apply Definition 1581.40. The endpoint seal E is downstream of the shared readback and tolerance ledger, and the rows H, C, P, N carry only transport, replay, provenance, and local naming. This exhausts the Cauchy majorant threshold route without importing quotient equality, selected limits, host infinite sums, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1581.42 (RealSeries root tail majorant window)**.** *Any root-unblock* $\mathsf{RealSeries}^{\uparrow}$ *consumer reading a tail-majorant row factors through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, the shared threshold* M*, and the endpoint seal* E*. In particular, a consumer of the accepted carrier*

$$K = (T, S, W, Q, D, M, E, H, C, P, N)$$

may cite a tail-majorant row only after the finite window route of W*, the regular readback route of* Q*, the dyadic tolerance ledger* D*, the shared threshold* M*, and the endpoint seal* E *have all been retained in the same transported packet.*

Proof. Start from the accepted carrier in Definition 1581.1. The displayed rows $T, S, W, Q, D, M, E, H, C, P, N$ are the only route available to a root-unblock consumer.

Use the tail-majorant carrier of Definition 1581.14. Its majorant read is finite because it exposes windows and readback data before any Cauchy-facing consumer is served.

Apply Definition 1581.37. The readiness surface ties the downstream Cauchy criterion, uniform Cauchy, and majorant consumers to the same base route.

Then apply Definition 1581.39 and Definition 1581.41. These theorems identify the shared threshold M and force every requested tail bound through W, Q, D, M before the endpoint seal E is cited. □

Theorem 1581.43 (RealSeries root tail threshold exhaustion)**.** *All downstream* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, *and* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *reads of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *root-unblock carrier are exhausted by the displayed rows*

$$T, S, W, Q, D, M, E, H, C, P, N.$$

No such read may introduce host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness outside the finite carrier and the tail-majorant rows of Definition 1581.14.*

Proof. First use Definition 1581.37. It identifies the downstream Cauchy criterion, uniform Cauchy, and majorant consumers as readers of the same finite surface rather than as independent sources of convergence data.

Next use Definition 1581.39. The three consumer routes agree on $T, S, W, Q, D, M, E, H, C, P, N$, with W, Q, D, M carrying the finite window, regular readback, dyadic tolerance, and shared-threshold part of the route.

Finally apply Definition 1581.41 together with Definition 1581.42. The first exhausts the majorant request through the existing threshold surface, and the second locks root-unblock reads to the same finite windows and endpoint seal.

The excluded host principles are absent from both displayed carriers. Since neither the base carrier nor the tail-majorant carrier has a coordinate for host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no downstream read can export one. □

Definition 1581.44 (RealSeries Dirichlet cancellation surface)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier*

$$K = (T, S, W, Q, D, M, E, H, C, P, N),$$

the Dirichlet cancellation surface is the finite row package

$$\mathsf{DirCancel}(K) = (T, S, W, Q, D, M, E, H, C, P, N;\ A, B, V)$$

where A *is a bounded partial-sum ledger for the displayed windows of* S*,* B *is a monotone null coefficient ledger read through the same* $\mathsf{StreamName}^{\uparrow}$ *windows* W *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* V *is the finite Abel-difference variation ledger that compares adjacent coefficient rows inside the dyadic tolerance ledger* D*. The surface has no coordinate for a host infinite sum, selected series limit, quotient equality of partial-sum streams, countable choice over tail indices, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1581.45 (RealSeries Dirichlet bounded-variation row)**.** *Let* $K = (T, S, W, Q, D, M, E, H, C, P, N)$ *be an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier with a Dirichlet cancellation surface* $\mathsf{DirCancel}(K)$*. Every finite window comparison for the transformed tails factors through the bounded partial-sum ledger* A*, the monotone null coefficient ledger* B*, and the Abel-difference variation ledger* V *before the shared threshold row* M *may be read. Thus the only exported route is*

$$S \longmapsto A \longmapsto B \longmapsto V \longmapsto D \longmapsto M,$$

with H, C, P, N *transporting, replaying, sourcing, and naming the same finite route.*

Proof. Project the base carrier of Definition 1581.1 and the cancellation surface of Definition 1581.44. The partial sums remain the displayed finite windows of S, while A records the bound used by every cancellation request.

Read the coefficient side through B and V. The null and monotone information is stored as finite window data, and the Abel-difference ledger records only adjacent coefficient differences inside the dyadic tolerance row D.

Finally use the shared threshold discipline of Definition 1581.43. Since the route reaches M only after S, A, B, V, D, no consumer can replace it by a host infinite sum, a selected limit, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1581.46 (RealSeries Dirichlet tail cancellation)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier equipped with* $\mathsf{DirCancel}(K)$*, every tail request between displayed indices* $m < n$ *is answered by a finite Abel-summation window:*

$$(m, n, S) \longmapsto A \longmapsto (B, V) \longmapsto D \longmapsto M \longmapsto E.$$

The endpoint seal E *is cited only after the bounded partial-sum ledger, monotone null coefficient ledger, Abel-difference variation ledger, dyadic tolerance accounting, and shared threshold have all been retained in the same transported packet.*

Proof. Begin with Definition 1581.45. It gives the finite route from the partial-sum window through the bounded ledger, coefficient ledger, variation ledger, dyadic tolerance row, and shared threshold.

Apply Definition 1581.10. The threshold row remains stable under componentwise hsame transport, so the same m, n tail comparison is replayed through H and C rather than by choosing another tail selector.

Use Definition 1581.12 for the final endpoint step. The endpoint seal E is downstream of the displayed threshold route, and no excluded host-side principle is a field of either the base carrier or the cancellation surface. □

Theorem 1581.47 (RealSeries Dirichlet Cauchy readiness)**.** *The Dirichlet cancellation surface of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier supplies exactly the Cauchy-ready route needed by downstream* $\mathsf{DirichletSeriesTest}^{\uparrow}$*,* $\mathsf{AbelDirichletComparison}^{\uparrow}$*, and* $\mathsf{UniformCauchySeries}^{\uparrow}$ *consumers:*

$$T, S, W, Q, A, B, V, D, M, E, H, C, P, N.$$

The route records finite term access, partial-sum windows, stream windows, regular readback, bounded partial sums, monotone null coefficients, Abel finite differences, dyadic tolerance, shared threshold, endpoint sealing, transport, continuation replay, provenance, and local naming, and nothing stronger.

Proof. First combine Definition 1581.44 with the base carrier of Definition 1581.1. This lists all rows visible to a Dirichlet-style downstream consumer.

Next apply Definition 1581.46. It shows that every requested tail comparison reaches the endpoint seal through a finite Abel-summation window and the shared threshold.

The consumer boundary is then a row-exhaustion claim. The displayed tuple has no coordinate for host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the downstream chapters receive only the Cauchy-ready finite certificate surface. □

Theorem 1581.48 (RealSeries Abel partial summation window)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier equipped with* $\mathsf{DirCancel}(K)$*, every finite Abel partial-summation request is read through the displayed rows*

$$S, A, B, V, D, M, E.$$

The row S supplies the inspected term and partial-sum windows, A supplies the bounded partial-sum ledger, B supplies the monotone null coefficient ledger, V supplies the adjacent Abel-difference ledger, D supplies dyadic tolerance accounting, M supplies the shared threshold, and E supplies the endpoint seal after those rows have already been retained. No host infinite sum, selected limit, quotient equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is exposed.*

Proof. Begin with the cancellation surface of Definition 1581.44. It displays A, B, V as the bounded partial-sum, coefficient, and Abel-difference rows over the same finite windows of S.

Apply Definition 1581.47. The readiness route keeps $T, S, W, Q, A, B, V, D, M, E, H, C, P, N$ in one finite packet, so the Abel window cannot be replaced by a host summation theorem.

Finally restrict to the Abel-facing analytic rows. The support rows H, C, P, N transport, replay, source, and name the same route, while the visible estimate itself is exhausted by S, A, B, V, D, M, E. □

Theorem 1581.49 (RealSeries Dirichlet uniform tail bound)**.** *For any accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier with* $\mathsf{DirCancel}(K)$*, a downstream* $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{CauchyMajorantSeries}^{\uparrow}$*, or* $\mathsf{DirichletSeriesTest}^{\uparrow}$ *consumer receives a uniform tail-bound route only through*

$$A, B, V, D, M, E, H, C, P, N.$$

The common threshold row M is readable only after the bounded partial-sum row A, coefficient row B, Abel-difference row V, and dyadic ledger D have been fixed over displayed finite windows.

Proof. Project the Dirichlet cancellation surface. The route to a tail estimate first names A and B, then records the finite adjacent-difference budget in V.

Use Definition 1581.46. It places every displayed tail request between $m < n$ on the finite chain from the partial-sum window through A, B, V, D, M and only then to E.

Use Definition 1581.47 for the consumer boundary. Uniform Cauchy, majorant, and Dirichlet-test consumers receive the same threshold row by continuation replay and provenance, not by selected tail indices, quotient equality, infinite sums, or ambient completeness. □

Theorem 1581.50 (RealSeries Cauchy-product finite window)**.** *Every* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier equipped with* $\mathsf{DirCancel}(K)$ *reads product-facing partial sums through finite windows only. The allowed handoff is*

$$S, A, B, V, D, M, E \longmapsto H, C, P, N,$$

where S supplies the finite partial-sum windows and A, B, V, D, M, E supply the cancellation, tolerance, threshold, and endpoint rows needed before any product window can be replayed.

Proof. Read the RealSeries carrier of Definition 1581.1. Product-facing access begins with the displayed partial-sum row S and cannot introduce a second series object outside the local packet.

Apply Definition 1581.44 and Definition 1581.48. The cancellation data available to a product window is exactly A, B, V, D, M, E over finite windows.

Finally apply Definition 1581.47. It exports the Cauchy-ready route to downstream consumers through H, C, P, N, so the $\mathsf{CauchyProduct}^{\uparrow}$ handoff receives finite product windows rather than host infinite products, selected sums, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{RealSeries}^{\uparrow}$ endpoint-facing read factors through a real-valued term stream, finite partial-sum window, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a shared Cauchy-tail threshold, a RealUp endpoint seal, transport, continuation, provenance, and local naming.

Independence witness: real, real-sequence-limit, cauchy-sequence-bounded, regseqrat, streamname, dyadicratcore: $\mathsf{RealSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries term-stream access, finite partial-sum windows, selected stream windows, regular readback, dyadic tolerance accounting, shared tail thresholding, endpoint sealing, transport, continuation, provenance, and local naming.

Theorem 1581.51 (RealSeries finite partial-sum carrier)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$*, every downstream root-facing read of finite partial sums is already determined by the displayed* BHist *rows* $T, S, W, Q, D, M, E, H, C, P, N$*. The row* T *supplies the term stream,* S *supplies the finite partial-sum windows,* W *supplies selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* Q *supplies* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* D *supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* M *supplies the shared threshold,* E *supplies the endpoint seal, and* H, C, P, N *provide componentwise* hsame *transport,* Cont *replay, provenance, and local* NameCert *naming.*

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesRootUnblockCarrier`

Proof. Project the carrier of Definition 1581.1. The finite partial-sum data is stored in S and is read only after the term stream T has been admitted.

Route those finite sums through the selected windows W and regular readback Q. The dyadic ledger D and shared threshold M then account for the comparisons before the endpoint row E is available.

The structural rows H, C, P, N transport, replay, source, and name this same finite route. Since the carrier has no coordinate for host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no root-facing read can bypass the displayed rows. □

Theorem 1581.52 (RealSeries tail ledger)**.** *For every accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C,$ *each tail comparison exported to a* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{CauchyMajorantSeries}^{\uparrow}$*, or* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer factors through the single ledger chain*

$$S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M$$

together with T, E, H, C, P, N*. In particular, the threshold row* M *is shared across the exported tail requests and cannot be replaced by a family of selected tail indices.*

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesTailLedger`

Proof. Begin with the finite partial-sum windows S. A tail request selects only finitely many windows through W and reads the corresponding rational data through Q.

The comparisons are then recorded in the dyadic tolerance ledger D. The shared threshold M is the only row admitted after D, so all exported tail bounds use the same displayed threshold surface.

Endpoint access through E and structural access through H, C, P, N preserve that route. The downstream consumers receive a tail ledger over finite rows, not a host infinite-sum principle or a chosen sequence of tail witnesses. □

Theorem 1581.53 (RealSeries endpoint readback)**.** *Every endpoint read of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$ *reaches* $\mathsf{Real}^{\uparrow}$ *through* E *only after the finite rows* T, S, W, Q, D, M *have been displayed. The readback route is therefore*

$$T, S, W, Q, D, M \longmapsto E \longmapsto H, C, P, N,$$

with BHist *admission, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming attached to the same packet.*

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesEndpointReadback`

Proof. Read the endpoint row from the carrier only after the term stream, finite partial sums, selected windows, regular readback, dyadic ledger, and shared threshold have been projected.

Apply Definition 1581.12. It places every $\mathsf{Real}^{\uparrow}$ endpoint seal behind T, S, W, Q, D, M and refuses selected limits, quotient equality of partial-sum streams, countable choice, and ambient completeness.

Finally replay the endpoint read through H, C, P, N. These rows preserve and name the finite packet; they do not provide a second endpoint channel outside the displayed $\mathsf{RealSeries}^{\uparrow}$ carrier. □

Theorem 1581.54 (RealSeries root tail factorization)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$*, every root-facing Cauchy consumer factors partial sums through finite tail windows before any convergence claim is visible. The forced route is*

$$S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M \longmapsto E,$$

where S *supplies the partial-sum ledger,* W *supplies the selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* Q *supplies* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for terms and partial sums,* D *supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* M *supplies the shared tail threshold, and* E *supplies the* $\mathsf{Real}^{\uparrow}$ *seal only after those rows have been displayed.*

Thus a downstream $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$*,* $\mathsf{UniformCauchySeries}^{\uparrow}$*,* $\mathsf{CauchyMajorantSeries}^{\uparrow}$*, or* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer receives a finite BEDC tail-window certificate. It receives no host infinite sum, selected limit, quotient equality of partial-sum streams, countable-choice tail selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the partial-sum row S. It names the finite windows of partial sums that a root-facing consumer may inspect, and it is already tied to the term stream T by the carrier of Definition 1581.1.

Move from S to selected windows and regular readback. The row W records the finite $\mathsf{StreamName}^{\uparrow}$ windows, and Q records the $\mathsf{RegSeqRat}^{\uparrow}$ readback for each displayed term and partial-sum window.

The Cauchy comparison is then ledgered. The row D records dyadic tolerances and finite addition errors, while the shared threshold row M is the only admitted tail gate after those comparisons.

Finally the endpoint seal E is reached only after the finite route has been displayed. The rows H, C, P, N transport, replay, source, and name the same route, so no convergence-facing consumer can bypass finite tail windows for a host infinite-sum or completeness principle. □

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesRootTailFactorization`

Theorem 1581.55 (RealSeries root-tail threshold fusion)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$*, every root-facing Cauchy consumer that inspects partial-sum windows must use the same threshold row* M *after the route*

$$S \longmapsto W \longmapsto Q \longmapsto D$$

and before the endpoint row E*. Thus no family of selected tail indices can replace the shared finite threshold while still claiming to consume the* $\mathsf{RealSeries}^{\uparrow}$ *packet.*

Proof. Start from the finite partial-sum carrier. The row S supplies the displayed partial-sum windows, and the route through W and Q exposes only finite StreamName windows and RegSeqRat readback.

Use the tail ledger and endpoint readback. The dyadic comparison row D is followed by the single threshold row M, and the endpoint seal E is not available until that threshold has been read.

Apply root tail factorization to downstream Cauchy consumers. Since every root-facing read factors through the same S, W, Q, D, M, E chain, selected families of tail indices have no carrier coordinate and cannot replace the shared threshold. □

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesRootTailThresholdFusion`

Theorem 1581.56 (RealSeries Cauchy consumer finite readback)**.** *Every* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, *or* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier receives only the finite route*

$$T, S, W, Q, D, M, E, H, C, P, N.$$

The consumer receives the term stream, finite partial-sum windows, selected StreamName windows, RegSeqRat readback, DyadicRatCore ledger, shared threshold, endpoint seal, componentwise transport, displayed replay, provenance, and local NameCert rows, but it receives no host infinite sum, selected limit, quotient equality, countable-choice tail selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the finite partial-sum carrier and tail ledger. They expose T, S, W, Q, D, M as the only route by which a Cauchy-series consumer can inspect terms, partial sums, windows, readback, tolerances, and the shared threshold.

Apply endpoint readback and root-tail threshold fusion. The endpoint seal E sits after the same finite threshold, so no consumer can reach a real-facing series claim by choosing separate tail indices.

Finally use root tail factorization to attach the structural rows H, C, P, N. These rows preserve and name the finite packet, while host infinite sums, selected limits, quotient equality, countable-choice tail selectors, and ambient completeness principles are absent from the carrier. □

Definition 1581.57 (RealSeries root Cauchy-product consumer surface)**.** *For an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier* $K = (T, S, W, Q, D, M, E, H, C, P, N)$*, the root Cauchy-product consumer surface is the finite* BHist *route*

$$\mathsf{RootProd}(T, S, W, Q, D, M, H, C, P, N)$$

whose rows are exactly the term stream, partial-sum windows, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, shared tail threshold, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *naming. A* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer may read this surface before endpoint sealing, but it receives no host infinite sum, selected product value, quotient equality of series, countable-choice tail selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1581.58 (RealSeries root product tail budget)**.** *Every* $\mathsf{CauchyProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier first consumes the root Cauchy-product surface of Definition 1581.57. Its tail request is therefore budgeted by the displayed route*

$$S \longrightarrow W \longrightarrow Q \longrightarrow D \longrightarrow M \longrightarrow H, C, P, N,$$

where S gives finite partial-sum windows, W gives selected stream windows, Q gives regular rational readback, D gives dyadic tolerance accounting, and M gives the shared tail threshold.

Lean4 ✓ `BEDC.Derived.RealseriesUp.RealSeriesRootProductTailBudget`

Proof. Project the carrier K. The product-facing rows visible before endpoint sealing are the finite partial-sum, window, readback, tolerance, threshold, transport, replay, provenance, and naming rows.

Read the route in the order displayed in the statement. The $\mathsf{RegSeqRat}^{\uparrow}$ row cannot be inspected before the $\mathsf{StreamName}^{\uparrow}$ windows are selected, and the shared threshold row cannot be inspected before the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger has named the requested budget.

The structural rows H, C, P, N preserve and name the same surface. Since no selected product value or host summation object is a carrier coordinate, the $\mathsf{CauchyProduct}^{\uparrow}$ consumer receives only the finite tail budget. □

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesRootProductTailBudget`

Theorem 1581.59 (RealSeries partial-sum product route)**.** *For every accepted* $\mathsf{RealSeries}^{\uparrow}$ *carrier, any product-facing read from partial-sum windows to a* $\mathsf{CauchyProduct}^{\uparrow}$ *handoff factors through*

$$\mathsf{partial\text{-}sum\ windows} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{shared\ tail\ threshold} \longrightarrow \mathsf{CauchyProduct}^{\uparrow}.$$

The route exposes finite window data, dyadic comparison radii, regular rational readback, the single shared tail threshold, and the local $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$ *rows; it does not expose a second product criterion outside the displayed series packet.*

Lean4 ✓ `BEDC.Derived.RealSeriesUp.RealSeriesPartialSumProductRoute`

Proof. Begin with the finite partial-sum windows in S. They are the only root rows that a product-facing consumer may treat as summation data.

Use the dyadic ledger D and regular readback row Q to account for finite comparison radii and rational window values. The shared threshold M is then the unique tail gate available to the consumer.

Finally hand the result to the $\mathsf{CauchyProduct}^{\uparrow}$ consumer through H, C, P, N. This is the same root surface named in Definition 1581.58, so no external product test, host infinite sum, or ambient completeness witness is exported. □

Theorem 1581.60 (RealSeries scoped certificate closure)**.** *The* $\mathsf{RealSeries}^{\uparrow}$ *chapter closes the finite certificate scope generated by its existing child files. The spine file Definition 1581.1 and Definition 1581.3 fixes the carrier and NameCert surface; the root-tail child records the tail ledger, endpoint readback, and root factorization routes; and the root Cauchy-product child records the product-facing tail budget. Together these files bind exactly the displayed StreamName, RegSeqRat, DyadicRatCore, Real,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the chapter.*

Proof. Read the three child files as one ordered certificate surface. The spine file supplies the real-valued term stream, partial-sum windows, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, shared tail threshold, $\mathsf{Real}^{\uparrow}$ endpoint seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

The root-tail child then proves that every endpoint-facing and Cauchy-facing read remains on that route. Its tail ledger, endpoint readback, root-tail factorization, threshold fusion, and

consumer readback statements leave no carrier coordinate for host infinite sums, selected limits, quotient equality, countable-choice tail selectors, or ambient completeness.

The product child closes the remaining root product consumer. It reads product-facing partial sums only through the same finite windows, dyadic ledger, regular readback, shared threshold, and structural rows. Thus the listed child files exhaust the scoped $\mathsf{RealSeries}^{\uparrow}$ certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}$.

Constructive story: A public $\mathsf{RealSeries}^{\uparrow}$ packet is generated from finite partial-sum windows, StreamName requests, RegSeqRat readback, DyadicRatCore budgets, a shared Cauchy-tail threshold, RealUp endpoint readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public $\mathsf{RealSeries}^{\uparrow}$ surface is the root tail factorization of Definition 1581.54 together with the CauchyProduct budget route of Definition 1581.58, binding partial-sum windows, StreamName requests, RegSeqRat readback, DyadicRatCore budgets, the shared tail threshold, RealUp endpoint readback, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSeries}^{\uparrow}}, \mathsf{axiomClean})$

Lean target: `BEDC.Derived.RealSeriesUp.RealSeriesRootTailFactorization`

Bridge status: paperBridge

Not claimed: This chapter does not close bridge checking, host infinite sums, selected limits, quotient equality, countable choice, ambient RealUp completeness, or CauchyProduct convergence outside the displayed finite route.

Upgrade path: Bridge checking is the remaining verification-side route beyond the mature paper closure recorded here.

Chapter 1582

A Concrete Naming Certificate for $\mathsf{CauchyLocalization}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyLocalization}^{\uparrow}$ packet records the finite-window point at which a regular rational Cauchy source is inspected only as far as a requested tolerance requires, then sealed into the Real route without exporting a quotient stream or a selected limit. It sits between the dyadic arithmetic surface of chapter 291, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, the effective threshold discipline of chapter 415, and the terminal real-name seal of chapter 38. The chapter isolates the Cauchy localization row consumed by Real and Completion arguments as finite BHist data, not as ambient completeness, countable choice, host real equality, or quotient equality of streams.

Definition 1582.1 (Cauchy localization carrier)**.** *A* $\mathsf{CauchyLocalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, R, D, T, L, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the finite source positions inspected for the requested tolerance.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational readback supplies the values visible through W.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational radius, dyadic comparison cells, and arithmetic budget attached to the request.*
4. *T is the* $\mathsf{CauchyRate}^{\uparrow}$ *threshold row selecting the displayed late index after which the window is sufficient for the tolerance in D.*
5. *L is the localized request window, pairing the request precision with the subwindow of W certified by T and read through R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, R, D, T, L have been displayed.*
7. *H records componentwise* hsame *transport for the StreamName window, RegSeqRat source, DyadicRatCore tolerance ledger, CauchyRate threshold, localized request window, Real seal, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a precision request through D, T, W, L, R, and finally to the seal row E.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyLocalization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, T, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, detached Real completeness, or an ambient completion theorem.

Theorem 1582.2 (Cauchy localization NameCert obligations)**.** *For an accepted* $\mathsf{CauchyLocalization}^{\uparrow}$ *carrier $K = (W, R, D, T, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the finite StreamName window, RegSeqRat source row, DyadicRatCore tolerance ledger, CauchyRate threshold, localized request window, Real seal, transport, replay, provenance, and local naming row;*
2. *classifier stability, requiring accepted packets to be compared componentwise only through $W, R, D, T, L, E, H, C,$*
3. *threshold exactness, requiring T to bind the request tolerance in D to a displayed late index before any localized window is exposed;*
4. *window extraction, requiring L to expose only the finite subwindow of W certified by T and read through the same source row R;*
5. *RegSeqRat handoff, requiring all rational readback for the localized window to pass through R rather than through a host stream;*
6. *Real seal non-escape, requiring every Real-facing read to pass through W, R, D, T, L before it reaches E;*
7. *transport, requiring componentwise* hsame *in H to preserve the displayed row order;*
8. *replay, requiring displayed* Cont *in C to repeat the same precision-to-window-to-seal route;*
9. *provenance, requiring P to name only the sibling surfaces displayed in the carrier;*
10. *naming, requiring N to certify exactly the local* $\mathsf{CauchyLocalization}^{\uparrow}$ *route and no hidden quotient, selected limit, host equality, countable-choice, or ambient-completeness row.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1582.1. The packet displays the window, source, tolerance, threshold, localized request, seal, transport, replay, provenance, and naming rows as the full local inventory.

Classifier stability follows because the classifier is rowwise. Every accepted comparison is made through $W, R, D, T, L, E, H, C, P, N$, and no further classifier coordinate is present.

Threshold exactness and window extraction are the analytic core of the packet. The dyadic tolerance row D states the requested radius, the CauchyRate row T supplies the late threshold for that radius, and the localized row L exposes only the finite part of W certified by that threshold.

The RegSeqRat handoff and Real seal boundary are ordered by the carrier. The rational readback is read through R at the displayed window, and the seal row E is downstream of W, R, D, T, L.

Transport, replay, provenance, and naming are structural. The rows H and C preserve and replay the same displayed route, while P and N package only that route. Since the carrier contains no hidden quotient, selected limit, host equality, countable-choice, or ambient-completeness row, none can be exported by the local certificate. □

Theorem 1582.3 (Cauchy localization window extraction)**.** *Every accepted* $\mathsf{CauchyLocalization}^{\uparrow}$ *packet exposes a finite Cauchy window sufficient for the requested tolerance. More precisely, if* $K = (W, R, D, T, L, E, H, C, P, N)$ *is accepted, then the request row in* D *and the threshold row* T *determine a finite localized window* L *whose entries are read through the* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *source* R *before any* $\mathsf{Real}^{\uparrow}$ *seal read reaches* E*. The packet cannot export quotient stream equality, selected limits, host real equality, countable choice, or ambient completeness.*

Proof. Begin with an accepted packet and project its displayed rows. The tolerance ledger D fixes the requested radius, while the threshold row T supplies the finite late boundary attached to that radius.

Use the localized request row L. It is downstream of D and T, so it may expose only the portion of the StreamName window W needed for the certified late-bound request. The RegSeqRat source row R supplies the rational observations for exactly that displayed window.

The Real seal row E is reached only after W, R, D, T, L have been consumed. The structural rows H, C, P, N preserve, replay, and name this same route, so the packet has no export path for quotient stream equality, selected limits, host real equality, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocalization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyLocalization}^{\uparrow}$ packet is generated from a StreamName finite window, RegSeqRat source row, DyadicRatCore tolerance ledger, CauchyRate threshold, localized request window, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1582.1, Definition 1582.2, and Definition 1582.3 seed the finite Cauchy localization route between Cauchy-rate data and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, threshold exactness, window extraction, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and naming targets for BEDC.Derived.CauchyLocalizationU

Chapter 1583

A Concrete Naming Certificate for $\mathsf{RealPowerSeries}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealPowerSeries}^{\uparrow}$ packet records a real power series only through a displayed coefficient stream, center, argument, radius row, finite windows, partial-sum rows, and named convergence-test majorants. It refines the finite partial-sum discipline of $\mathsf{RealSeries}^{\uparrow}$ in chapter 1581, consumes the majorant route of $\mathsf{WeierstrassMTest}^{\uparrow}$ in chapter 399, and may use the geometric tail witnesses of $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ and $\mathsf{RegularCauchyRootTest}^{\uparrow}$ in chapter 1461 and chapter 1481. It also sits beside the coefficient-only $\mathsf{FPS}^{\uparrow}$ surface of chapter 51 and the displayed radius rows of chapter 66, while all real-facing reads pass through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 38, chapter 301, chapter 126, and chapter 291. The packet names analytic convergence as a finite certificate route; it does not import a host power-series convergence theorem, host analytic continuation, quotient function spaces, selected infinite sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1583.1 (Real power series carrier)**.** *A* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, Z, X, R, W, S, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the coefficient stream or* $\mathsf{FPS}^{\uparrow}$ *row, read only through selected coefficient windows and their* $\mathsf{StreamName}^{\uparrow}$ *provenance.*
2. *Z is the center* $\mathsf{Real}^{\uparrow}$ *row.*
3. *X is the argument* $\mathsf{Real}^{\uparrow}$ *row.*
4. *R is the displayed radius or subradius row used to state the accepted window of evaluation.*
5. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows for coefficients, center-readbacks, argument-readbacks, powers of $X - Z$, and partial sums.*
6. *S is the finite partial-sum and* $\mathsf{RealSeries}^{\uparrow}$ *row listing the inspected polynomial sums and their real seals.*
7. *M is the majorant data imported from* $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *rows.*
8. *E is the endpoint* $\mathsf{Real}^{\uparrow}$ *seal produced only after the displayed windows, partial sums, radius row, and majorant route have been read.*

9. *H stores componentwise* hsame *transport for the coefficient row, center, argument, radius row, windows, partial sums, majorant data, endpoint seal, continuation, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from coefficient reads and real-point reads to finite powers, from powers to partial sums, from partial sums to the chosen majorant route, and from that route to the endpoint seal.*

11. *P is the* Pkg *provenance over* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{FPS}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Z, X, R, W, S, M, E, H, C, P, N$. It has no coordinate for host analytic continuation, quotient function spaces, selected infinite sums, countable-choice selectors, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier semantic name certificate`

Definition 1583.2 (Real power series obligation carrier)**.** *A* $\mathsf{RealPowerSeries}^{\uparrow}$ *obligation carrier over a finite evaluation request is a displayed subpacket*

$$O = (A, R, B, U, T, V, Q, E, H, C, P, N)$$

inside the carrier of Definition 1583.1. The rows have the following meanings:

1. *A is the finite coefficient-window row read from the coefficient stream or* $\mathsf{FPS}^{\uparrow}$ *surface.*

2. *R is the displayed convergence-radius or subradius row, with only the finite comparison budget needed by the request.*

3. *B is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product ledger for the coefficient windows used to multiply finite polynomial truncations.*

4. *U is the selected* $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *handoff row used to justify the radius read.*

5. *T is the* $\mathsf{StreamName}^{\uparrow}$ *family of coefficient, power, product, and partial-sum windows.*

6. *V is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected partial sums and endpoint budgets.*

7. *Q is the* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row obtained from A, B, T, V.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, available only after A, R, B, U, T, V, Q have been displayed.*

9. *H records componentwise* hsame *transport for all analytic rows and the structural rows.*

10. *C records displayed* Cont *replay from coefficient windows to products, from products to partial sums, from radius data to the selected test handoff, and from those rows to the terminal seal.*

11. *P is the* Pkg *provenance over* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$ *obligation row.*

The classifier for obligation carriers compares only $A, R, B, U, T, V, Q, E, H, C, P, N$. It has no coordinate for analytic continuation, host infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesObligationCarrier`

Theorem 1583.3 (Real power series radius obligation)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *obligation carrier* $O = (A, R, B, U, T, V, Q, E, H, C, P, N)$*, a convergence-radius read reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$A, R \longmapsto T \longmapsto Q \longmapsto U \longmapsto V \longmapsto E$$

together with H, C, P, N*. The radius row* R *supplies a displayed radius or subradius budget; the selected test row* U *supplies only M-test, ratio-test, or root-test evidence over finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{RealSeries}^{\uparrow}$ *windows.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesObligationCarrier radius obligation`

Proof. Project the obligation carrier of Definition 1583.2. The radius-facing rows before the endpoint are exactly the coefficient window A, radius row R, stream windows T, partial-sum row Q, selected test row U, and regular rational readback V.

Read R only as displayed radius or subradius data. The row T supplies the finite coefficient, power, and partial-sum windows needed to test the radius claim, while Q records the inspected $\mathsf{RealSeries}^{\uparrow}$ partial sums.

The selected test row U is the only convergence justification. It may cite $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, or $\mathsf{RegularCauchyRootTest}^{\uparrow}$, and its endpoint-facing output is read through V before E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the obligation carrier contains no field for analytic continuation, host infinite sums, selected limits, quotient equality, countable choice, or ambient completeness, the radius obligation is confined to the displayed rows. □

Theorem 1583.4 (Real power series Cauchy-product handoff)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *obligation carrier* $O = (A, R, B, U, T, V, Q, E, H, C, P, N)$*, every endpoint-facing product of coefficient windows must pass through the* $\mathsf{CauchyProduct}^{\uparrow}$ *row* B *before it can be read as a* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row or a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The product handoff is the finite route*

$$A \longmapsto B \longmapsto T \longmapsto Q \longmapsto V \longmapsto E$$

with the same radius row R*, selected test row* U*, transport, replay, provenance, and local NameCert rows retained.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier cauchy product handoff`

Proof. Begin with the coefficient-window row A. A product consumer may inspect only the finite coefficient windows displayed in the obligation carrier, so it has no direct route to a host multiplication of infinite series.

The row B is the unique $\mathsf{CauchyProduct}^{\uparrow}$ ledger for those coefficient windows. Reading B fixes the finite convolution budget before the stream windows T and the $\mathsf{RealSeries}^{\uparrow}$ partial-sum row Q can be consumed.

The readback row V then turns the displayed partial sums and product budget into regular rational endpoint data, and the terminal seal E is downstream of that readback. The radius row R and selected test row U remain in the same packet, so the product handoff also preserves the convergence budget used by the endpoint read.

The structural rows H, C, P, N carry only componentwise transport, replay, provenance, and local naming. Because the carrier has no field for host products, quotient series equality, selected infinite sums, analytic continuation, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the Cauchy-product route is exactly the finite handoff displayed above. □

Theorem 1583.5 (Real power series coefficient-window obligation)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the coefficient and* $\mathsf{FPS}^{\uparrow}$ *obligation is exactly the route from* A *through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W *into the partial-sum row* S*. The route retains the center and argument* $\mathsf{Real}^{\uparrow}$ *rows* Z, X*, the displayed radius or subradius row* R*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback budgets,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RealSeries}^{\uparrow}$ *partial sums, componentwise transport* H*, replay* C*, provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$ *row* N*.*

Proof. Project the displayed carrier. The only coefficient-facing coordinate is A, and Definition 1583.1 says that all coefficient, real-readback, power, and partial-sum inspections pass through the finite window row W.

The partial-sum row S is the first row after those windows. It is therefore the place where $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ evidence can be read for coefficient windows and tolerances.

Finally read the structural rows H, C, P, N. They transport, replay, source, and locally name the same displayed route, while the carrier excludes host coefficient streams, quotient function spaces, selected infinite sums, and ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient window obligation`

Theorem 1583.6 (Real power series majorant-radius obligation)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the displayed radius or subradius row* R *reaches endpoint-facing convergence evidence only through* W, S, M*. The majorant row* M *must be read as the selected* $\mathsf{WeierstrassMTest}^{\uparrow}$*,* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*, or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *route over the finite* $\mathsf{RealSeries}^{\uparrow}$ *partial sums, with* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, transport, replay, provenance, and local naming kept in the same packet.*

Proof. Start with the radius clause of the carrier. The row R is displayed as radius or subradius evidence, not as a host convergence theorem.

The finite windows W and partial sums S mediate every read from R to convergence data. Thus the majorant row M can be consumed only after the coefficient, center, argument, radius, window, and partial-sum rows have been exposed.

The allowed shapes of M are the named M-test, ratio-test, and root-test routes. Since H, C, P, N only preserve the same carrier, the radius obligation cannot escape into analytic continuation, quotient function values, selected sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier majorant radius obligation`

Theorem 1583.7 (Real power series radius-window obligation basis)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the radius row* R *is usable by convergence consumers only after the coefficient stream row* A*, center and argument rows* Z, X*, finite* $\mathsf{StreamName}^{\uparrow}$

windows W, partial-sum row S, and selected majorant row M have been displayed. The basis row is the finite route

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E$$

with the same BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N. It may cite* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RealSeries$^{\uparrow}$*, and* Real$^{\uparrow}$ *only through those displayed rows.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius window obligation basis`

Proof. First expose the coefficient and point windows. By Definition 1583.5, the coefficient stream row A and the center and argument rows Z, X reach a partial-sum consumer only through the finite StreamName$^{\uparrow}$ windows W.

Next read the radius and majorant surface. By Definition 1583.6, the radius row R is not independently consumable; it must pass through W, S, M, where S is the finite RealSeries$^{\uparrow}$ partial-sum row and M is the selected majorant row.

The handoff is fixed by Definition 1583.11. It gives the displayed route from A, Z, X, R through W, S, M before the endpoint seal E and preserves the same BHist, hsame, Cont, Pkg, and NameCert rows.

Finally apply Definition 1583.9. The terminal Real$^{\uparrow}$ seal and the RegSeqRat$^{\uparrow}$ and DyadicRatCore$^{\uparrow}$ readbacks are available only along that finite route, so a convergence consumer cannot read R by host convergence, selected infinite sums, or ambient Real$^{\uparrow}$ completeness. □

Theorem 1583.8 (Real power series obligation scope)**.** *For every accepted* RealPowerSeries$^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, coefficient data are exposed only through the* FPS$^{\uparrow}$ *or* StreamName$^{\uparrow}$ *windows in A and W, radius data only through the displayed* ConvRad$^{\uparrow}$ *or subradius row R, partial sums only through the* RealSeries$^{\uparrow}$ *row S, convergence justification only through the selected* WeierstrassMTest$^{\uparrow}$*,* RegularCauchyRatioTest$^{\uparrow}$*, or* RegularCauchyRootTest$^{\uparrow}$ *majorant row M, and terminal output only through the* Real$^{\uparrow}$ *seal E after the* DyadicRatCore$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *readbacks named by W, S, M have been consumed.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesObligationScope`

Proof. Project the carrier of Definition 1583.1. Its coefficient-facing coordinates are A and the finite windows W, its radius coordinate is R, its partial-sum coordinate is S, its convergence coordinate is M, and its terminal row is E.

Apply Definition 1583.5 to force coefficient and formal-series reads through A and W. Apply Definition 1583.6 to place radius reads before the partial-sum and majorant rows. Apply Definition 1583.9 to place the endpoint seal after the dyadic and regular-rational readbacks.

The structural rows H, C, P, N only transport, replay, source, and name the same route. Since the carrier has no coordinate for analytic continuation, quotient function spaces, selected sums, countable choice, or ambient Real$^{\uparrow}$ completeness, row exhaustion excludes each of those objects from the accepted obligation scope. □

Theorem 1583.9 (Real power series terminal Real seal obligation)**.** *For every accepted* RealPowerSeries$^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, the terminal* Real$^{\uparrow}$ *seal E is available only after the coefficient or* FPS$^{\uparrow}$ *row A, center and argument* Real$^{\uparrow}$ *rows Z, X, radius row R, finite* StreamName$^{\uparrow}$ *windows W,* RegSeqRat$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *tolerance readbacks,* RealSeries$^{\uparrow}$ *partial sums S, and M-test or ratio/root-test majorant row M have been read. The seal is transported, replayed, sourced, and named only by H, C, P, N.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier terminal real seal obligation`

Proof. Read the endpoint clause in the carrier. The row E is produced only after the displayed windows, partial sums, radius row, and majorant route have been assembled.

The preceding rows determine the terminal route: coefficients or formal-series data enter through A, real input data through Z, X, tolerances through R, W, S, and convergence justification through M.

The field-exclusion clause of the definition then supplies nonescape. No carrier coordinate remains for a host infinite sum, function-space quotient, analytic continuation theorem, countable-choice selector, or ambient real-completeness witness, so E is exactly the local terminal Real seal of the packet. □

Theorem 1583.10 (Real power series NameCert obligations)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows, coefficient-stream access through* A*, center and argument sealing through* Z *and* X*, radius-window admission through* R*, selected* $\mathsf{StreamName}^{\uparrow}$ *windows through* W*, finite partial-sum and* $\mathsf{RealSeries}^{\uparrow}$ *handoff through* S*, majorant-route admission through* M*, endpoint sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesUpTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1583.1. The coefficient, point, radius, window, partial-sum, and majorant coordinates are exactly A, Z, X, R, W, S, M, so every analytic observation must enter through those rows.

The endpoint row E is read only after that displayed route has been assembled. Thus the real-facing output is a packet endpoint seal rather than a host infinite sum or an ambient completeness witness.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1583.11 (Real power series radius-window handoff)**.** *Every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *evaluation read factors through*

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E$$

together with H, C, P, N*. The row* R *supplies only a displayed radius or subradius window,* S *supplies only finite polynomial partial sums and their* $\mathsf{RealSeries}^{\uparrow}$ *handoff, and* M *supplies only the selected* $\mathsf{WeierstrassMTest}^{\uparrow}$*,* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*, or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *majorant route.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius window handoff`

Proof. Begin with a requested evaluation. By Definition 1583.1, the request can inspect the coefficient row A, center Z, argument X, and displayed radius row R before it chooses any finite windows.

The selected windows W expose only the coefficients, real-point readbacks, powers of $X - Z$, and inspected partial sums needed for the request. The partial-sum row S repacks those reads as the finite $\mathsf{RealSeries}^{\uparrow}$ surface visible to the convergence machinery.

The majorant row M is then the only route from the finite partial sums to endpoint convergence data. Reading E and the structural rows H, C, P, N therefore hands off a finite radius-window certificate and no host convergence theorem. □

Theorem 1583.12 (Real power series center evaluation window)**.** *Let $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier. An evaluation request whose displayed argument row X is the same center row Z may read the endpoint seal only through the coefficient row A, the finite window row W restricted to the constant coefficient and the zero-power window, the partial-sum row S, the selected majorant row M, and then E, H, C, P, N. The center evaluation exports no host Taylor theorem, analytic continuation, selected infinite sum, quotient function equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCenterEvaluationWindow`

Proof. Project the accepted carrier. The rows available to an evaluation request are exactly $A, Z, X, R, W, S, M, E, H, C, P, N$.

Apply Definition 1583.11. It routes every evaluation through A, Z, X, R before the finite window row W, and then through S and M before the endpoint seal E can be read.

Use the displayed equality of the argument row with the center row. In that local request, the finite power window inside W collapses to the zero-power read, so the only coefficient-facing data that can reach S is the constant coefficient route from A through the corresponding window in W.

Finally apply Definition 1583.23 and Definition 1583.9. The endpoint remains inside the displayed packet and the terminal $\mathsf{Real}^{\uparrow}$ seal is read only after the finite rows already named, so none of the excluded host analytic, selector, quotient, choice, or ambient-completeness objects can be exported. □

Theorem 1583.13 (Real power series coefficient-shift route)**.** *Let $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier. Shifting the center row Z to any displayed subcenter admitted by the finite radius-window row R preserves coefficient reads only along*

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E$$

together with H, C, P, N. The shifted read may reindex a displayed finite coefficient window, but it cannot import an ambient Taylor expansion, a chosen analytic continuation, or an infinite coefficient selector.

Proof. Begin with the radius-window handoff of Definition 1583.11. It places every endpoint-facing read after the coefficient, center, argument, and radius rows A, Z, X, R.

A subcenter is therefore visible only as data already admitted by R and inspected through the finite window row W. Reindexing coefficients changes the window read, not the carrier route.

The partial-sum and majorant rows S, M consume that displayed window before the endpoint seal E is available. The structural rows H, C, P, N transport, replay, source, and name the same packet, so no host analytic continuation or infinite coefficient selection is introduced. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient shift route`

Theorem 1583.14 (Real power series Cauchy-product radius budget)**.** *For an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, every product of accepted partial-sum windows must consume the displayed* $\mathsf{CauchyProduct}^{\uparrow}$ *budget inside S before it reaches the endpoint seal E. The product route remains bounded by the same $A, Z, X, R, W, S, M, H, C, P, N$ rows and cannot pass directly from two host sums to an endpoint value.*

Proof. Read the product request as a consumer of the partial-sum row S. By Definition 1583.11, S is reached only after the coefficient, center, argument, radius, and finite-window rows have been displayed.

The Cauchy-product budget is therefore a finite ledger inside the partial-sum route. It pairs only accepted windows already exposed by W and then hands the resulting product window to the majorant row M.

Only after that majorant route is consumed may the endpoint seal E be read. Thus the product construction has no coordinate for multiplying selected infinite sums, appealing to host convergence, or bypassing the displayed radius budget. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier cauchy product radius budget`

Theorem 1583.15 (Real power series majorant consumer coverage)**.** *Every convergence-facing consumer of an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier must select exactly one displayed majorant route from* $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *at row M before reading E. A consumer that uses host analytic continuation, a selected infinite sum, or an unstated completeness theorem is not a* $\mathsf{RealPowerSeries}^{\uparrow}$ *consumer of the displayed packet.*

Proof. Apply Definition 1583.11. The endpoint-facing path passes through S and then through M, so convergence data is visible only at the majorant row.

The carrier description gives three displayed majorant routes at M: the M-test route, the ratio-test route, and the root-test route. Choosing more than one route would be additional data, while choosing none would leave no convergence-facing coordinate before E.

The endpoint seal consumes exactly the selected route. Host analytic continuation, selected infinite sums, and ambient completeness are absent from the carrier and are refused by the finite-window boundary. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier majorant consumer coverage`

Theorem 1583.16 (Real power series coefficient-window transport)**.** *Let $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier. Any coefficient-window consumer transported along H and replayed along C reaches the partial-sum row S only through the displayed coefficient row A and the finite window row W. The transported read preserves the same center row Z, argument row X, radius row R, majorant row M, endpoint seal E, provenance P, and local naming N.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient window transport`

Proof. Project the carrier of Definition 1583.1. The only coefficient-facing coordinate is A, and the only row that exposes coefficient observations to downstream evaluation is W.

Apply Definition 1583.11. The handoff places W before S, so transport through H and replay through C can move a coefficient-window read only along the displayed A, W, S route.

The remaining rows are carried componentwise. Thus Z, X, R, M, E, P, N stay attached to the same packet, and no host coefficient stream, quotient function space, or selected infinite coefficient object is introduced. □

Theorem 1583.17 (Real power series radius subwindow monotonicity)**.** *For an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, replacing the displayed radius-window row R by a finite subwindow already admitted by W preserves the evaluation route*

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E$$

and keeps the same transport, replay, provenance, and naming rows H, C, P, N. The subwindow operation narrows the inspected radius evidence; it does not create an endpoint outside the displayed majorant route.

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius subwindow monotonicity`

Proof. Read the radius clause in Definition 1583.1. The row R is only a displayed radius or subradius row, while W records the finite windows through which the evaluation is inspected.

Use Definition 1583.11. A smaller admitted radius window remains before the same finite-window row W, the same partial-sum row S, and the same majorant row M.

The endpoint seal E is still downstream of the majorant route. Since the carrier has no coordinate for ambient analytic continuation or completeness, narrowing R cannot export a new endpoint principle. □

Theorem 1583.18 (Real power series partial-sum majorant readback)**.** *Every partial-sum readback from an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ *factors from the finite polynomial-sum row* S *into the selected majorant row* M *before the endpoint seal* E *can be read. The readback may cite* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *through* S *and* M*, but it receives no selected infinite sum or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier partial sum majorant readback`

Proof. Start from the partial-sum clause of Definition 1583.1. The row S lists only the inspected polynomial sums and their finite $\mathsf{RealSeries}^{\uparrow}$ handoff.

Apply Definition 1583.11. The endpoint-facing path from S must pass through M, where the selected M-test, ratio-test, or root-test route is displayed.

Finally use Definition 1583.23. The carrier exports the finite majorant readback and the endpoint seal, not a host infinite series object, selected sum, or completeness theorem. □

Theorem 1583.19 (Real power series radius-majorant coverage)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the radius and majorant coverage surface is exactly the route through the coefficient-window obligation, majorant-radius obligation, terminal* $\mathsf{Real}^{\uparrow}$ *seal obligation, radius-window handoff, and partial-sum majorant readback rows. Endpoint convergence is visible only through* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$ *or the ratio/root-test rows,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius majorant coverage`

Proof. Project the carrier K. The coefficient row A, radius row R, finite windows W, partial sums S, selected majorant row M, and endpoint seal E are the only analytic-facing coordinates.

Apply the coefficient-window obligation, the majorant-radius obligation, and the terminal seal obligation. These rows place all coefficient reads, radius reads, and endpoint reads inside A, R, W, S, M, E with the same structural rows H, C, P, N.

Use the radius-window handoff and partial-sum majorant readback to identify the route from finite windows to convergence evidence. The allowed endpoint evidence is therefore exhausted by the named BEDC rows, and no analytic continuation, host infinite sum, or $\mathsf{Real}^{\uparrow}$ completeness coordinate is introduced. □

Theorem 1583.20 (Real power series stream-window convergence route)**.** *Every convergence-facing read from an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ *must first expose the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W *for coefficients, point readbacks, powers, and partial sums before it can consume the selected majorant row* M *or the endpoint seal* E*. The route may pass through* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$,

RegSeqRat$^{\uparrow}$, *and* DyadicRatCore$^{\uparrow}$, *but it cannot bypass* W *by using a host convergence theorem or a selected infinite sum.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier stream window convergence route`

Proof. Start with Definition 1583.5. It forces coefficient and formal-series access through the finite window row W before any partial-sum row is available.

Apply Definition 1583.11. The handoff places W before S, then S before M, and finally M before the endpoint seal E.

The partial-sum majorant readback keeps the convergence witness inside the displayed majorant route. Since the carrier contains no host convergence theorem or selected sum coordinate, every convergence-facing read preserves the stream-window route. □

Theorem 1583.21 (Real power series RegSeqRat endpoint handoff)**.** *For every accepted* RealPowerSeries$^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the* RegSeqRat$^{\uparrow}$*-facing endpoint handoff is read only after the coefficient windows, radius row, partial-sum row, and majorant route have been displayed. The handoff may cite* RealSeries$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegularCauchyRatioTest$^{\uparrow}$, RegularCauchyRootTest$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *through* W, S, M, E, H, C, P, N*, and it exports no analytic continuation, host infinite sum, quotient function space, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Use Definition 1583.26. It already identifies the RegSeqRat$^{\uparrow}$-facing budget as the route from A, Z, X, R through W, S, M, E.

The radius-majorant coverage theorem above ties that budget to the coefficient-window, majorant-radius, terminal seal, radius-window, and partial-sum readback rows. Thus endpoint-facing RegSeqRat$^{\uparrow}$ data are visible only after the finite radius and majorant surface has been read.

Finally apply Definition 1583.23. The endpoint handoff remains inside K and its structural rows, so no excluded analytic or completeness object can be exported. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier regseqrat endpoint handoff`

Theorem 1583.22 (Real power series endpoint-seal factorization)**.** *The endpoint row* E *of an accepted* RealPowerSeries$^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ *factors through the coefficient row* A*, center row* Z*, argument row* X*, radius row* R*, window row* W*, partial-sum row* S*, and majorant row* M *before it can be transported or replayed through* H, C, P, N*. Thus the endpoint seal is a packet-local* Real$^{\uparrow}$ *read, not a host analytic continuation, quotient function value, selected infinite sum, countable-choice output, or ambient completeness datum.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier endpoint seal factorization`

Proof. Use the factorization displayed in Definition 1583.11. It places every endpoint-facing read after A, Z, X, R, W, S, M.

Use Definition 1583.10 to attach H, C, P, N to the same route. These rows transport, replay, source, and name the packet-local endpoint seal without adding analytic coordinates.

The excluded objects are absent from the carrier and from the handoff route. Therefore E remains the finite endpoint seal of the accepted packet. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesEndpointSealFactorization`

Theorem 1583.23 (Real power series nonescape)**.** *Every endpoint-facing read from an accepted* RealPowerSeries$^{\uparrow}$ *carrier passes through* $A, Z, X, R, W, S, M, E, H, C, P, N$*. Consequently a consumer may use the displayed* RealSeries$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegularCauchyRatioTest$^{\uparrow}$, RegularCauchyRootTest$^{\uparrow}$,

$\mathsf{FPS}^{\uparrow}$, *and* $\mathsf{ConvRad}^{\uparrow}$ *interfaces named by the packet, but it receives no host analytic continuation, quotient function space, selected infinite sum, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 1583.11 to place every endpoint-facing evaluation on the finite route through A, Z, X, R, W, S, M. The endpoint seal E consumes that route after the coefficient, point, radius, partial-sum, and majorant data have been displayed.

The formal series and convergence-radius rows remain named inputs to the packet, not ambient host objects. They expose coefficient and radius evidence only through the finite rows retained by the carrier.

Finally read H, C, P, N. These rows preserve, replay, source, and name the same displayed data, so none of the excluded analytic or choice principles can escape through the accepted classifier. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier nonescape`

Theorem 1583.24 (Real power series coefficient-window ledger)**.** *Every coefficient-facing or* $\mathsf{FPS}^{\uparrow}$ *read from an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier must pass through A and then through the finite window row W before it can contribute to the partial-sum row S. Thus coefficients reach endpoint-facing evidence only as displayed stream windows, center-readbacks, argument-readbacks, powers of $X - Z$, and inspected polynomial sums.*

Proof. Start from the carrier of Definition 1583.1. The only coefficient or formal-series coordinate is A, and the only row that exposes coefficient inspections to downstream evaluation is the finite window row W.

Apply Definition 1583.11. The handoff places W before S, so partial sums can use only those displayed coefficient and power windows.

Finally apply Definition 1583.23. Since the accepted carrier has no other coefficient coordinate, no host stream, selected infinite coefficient object, or ambient analytic series can bypass the A, W, S route. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient window ledger`

Theorem 1583.25 (Real power series majorant-route totality)**.** *All convergence evidence exported by an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier passes through M as a selected* $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *route. The carrier therefore exposes convergence only through the displayed majorant row and not through a host convergence theorem, selected infinite sum, or ambient completeness proof.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier majorant route totality`

Proof. Use Definition 1583.1. The row M is the only convergence-test coordinate, and it is explicitly limited to the named M-test, ratio-test, or root-test routes.

By Definition 1583.11, the finite partial sums S must pass through M before an endpoint seal can be read. Thus every convergence-facing consumer receives the selected majorant route rather than a host analytic convergence object.

The boundary statement of Definition 1583.23 then excludes host analytic continuation, quotient function spaces, selected infinite sums, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the exported evidence. □

Theorem 1583.26 (Real power series majorant RegSeqRat budget)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the selected majorant row* M *yields a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing budget only through the displayed route*

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E.$$

Here S *supplies the finite* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum surface, while* M *selects exactly one of the* $\mathsf{WeierstrassMTest}^{\uparrow}$*,* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*, or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *evidence routes. The budget exports no independent regular-rational stream, selected infinite sum, host convergence theorem, quotient function space, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the radius-window handoff of Definition 1583.11. It forces every endpoint-facing evaluation through A, Z, X, R, then through the finite windows W, before any partial-sum or majorant row is available.

Apply the partial-sum majorant readback theorem, Definition 1583.18. The row S is therefore the finite $\mathsf{RealSeries}^{\uparrow}$ surface from which a $\mathsf{RegSeqRat}^{\uparrow}$-facing budget can be read, and the next row is necessarily the selected majorant row M.

Use the majorant-route totality theorem above to identify the only possible shapes of M: $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, or $\mathsf{RegularCauchyRootTest}^{\uparrow}$. Finally apply Definition 1583.23; since no other convergence or stream coordinate exists in the carrier, the budget cannot escape into the excluded host or choice principles. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier majorant regseqrat budget`

Theorem 1583.27 (Real power series majorant exhaustion)**.** *Every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *convergence read must choose exactly one of the* $\mathsf{WeierstrassMTest}^{\uparrow}$*,* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*, or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *majorant rows before the endpoint seal* E *is available. The choice is read as the displayed row* M *of the carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, and no convergence read may bypass* M *by using a host convergence theorem, selected infinite sum, analytic continuation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1583.1. Its only convergence-test coordinate is M, and the definition restricts M to the M-test, ratio-test, or root-test routes.

Apply Definition 1583.25. It places every exported convergence evidence inside that selected row rather than in a detached theorem of host analysis.

Finally use Definition 1583.26. The endpoint-facing route reaches E only after S has passed through M, so the selected majorant row is exhausted before the endpoint seal is read. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier majorant exhaustion`

Theorem 1583.28 (Real power series radius-source compatibility)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the radius row* R*, coefficient* $\mathsf{StreamName}^{\uparrow}$ *windows inside* W*, and* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row* S *share the same* BHist *source route. A consumer that reads any one of these rows must preserve the common carrier source through* H, C, P, N *before it can reach the selected majorant row* M *or the endpoint seal* E*.*

Proof. Start with the coefficient-window obligation of Definition 1583.5. The coefficient row A, radius row R, window row W, and partial-sum row S are projected from one accepted packet.

Apply the radius-window handoff of Definition 1583.11. It gives a single route from A, Z, X, R through W and S before any majorant data can be consumed.

The rows H, C, P, N then preserve, replay, source, and name that same route. Hence a radius read, coefficient-window read, or partial-sum read cannot switch to another BHist source before reaching M or E. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius source compatibility`

Theorem 1583.29 (Real power series endpoint nonescape triad)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier, the endpoint seal E cannot export any of the following three objects: a selected infinite sum, host analytic continuation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Each endpoint-facing read remains confined to the displayed route through $A, Z, X, R, W, S, M, E, H, C, P, N$.*

Proof. Use Definition 1583.22. It places the endpoint seal after the coefficient, center, argument, radius, window, partial-sum, and majorant rows, with H, C, P, N attached to the same packet.

Apply Definition 1583.23. The carrier has no coordinate for selected infinite sums, analytic continuation, or ambient completeness, and the endpoint-facing route introduces no additional row.

Therefore the three excluded objects cannot be exported by E. The endpoint seal remains the packet-local $\mathsf{Real}^{\uparrow}$ read produced by the displayed majorant route. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier endpoint nonescape triad`

Theorem 1583.30 (Real power series CauchyProduct coefficient budget)**.** *Every endpoint-facing product or convolution read from an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ must first expose the coefficient row A, center row Z, argument row X, radius row R, finite window row W, partial-sum row S, and selected majorant row M. Only after those rows are displayed may the read consume the* $\mathsf{CauchyProduct}^{\uparrow}$ *budget route named by Definition 464.101, Definition 464.118, and Definition 464.119; the endpoint seal E is exported only after that product-budget route has been read. The carrier exports no selected infinite sums, quotient function spaces, host analytic continuation, host multiplication primitive, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the RegSeqRat-facing budget of Definition 1583.26. It forces an endpoint-facing read through A, Z, X, R, W, S, M, E and identifies S as the finite $\mathsf{RealSeries}^{\uparrow}$ surface from which coefficient evidence is read.

Next apply the partial-sum majorant readback theorem, Definition 1583.18. The row M is therefore already selected before any product or convolution consumer asks for a downstream coefficient budget.

Now use the CauchyProduct budget route. The root budget product lock, source-window budget compatibility, and RegSeqRat product budget exhaustion cited in the statement route every product classifier read through the displayed CauchyProduct source windows, dyadic budget rows, product row, budget ledger, and classifier row before any Real seal is available.

Finally return to the endpoint seal E through the structural rows H, C, P, N. Since neither the power-series carrier nor the CauchyProduct budget route contains a coordinate for selected infinite sums, quotient function spaces, host analytic continuation, host multiplication, or ambient completeness, none of those objects can be exported with the endpoint seal. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier cauchy product coefficient budget`

Theorem 1583.31 (Real power series endpoint-seal scope)**.** *The endpoint row E of an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier is scoped only after the displayed route through $A, Z, X, R, W, S, M, H, C, P, N$. It exports no host analytic continuation, quotient function space, selected infinite sum, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier endpoint seal scope`

Proof. Read the endpoint clause of Definition 1583.1. The row E is produced only after the displayed windows, partial sums, radius row, and majorant route have been read.

Apply Definition 1583.11 to place the endpoint after A, Z, X, R, W, S, M and alongside the structural rows H, C, P, N. Apply Definition 1583.23 to identify the excluded analytic and choice principles.

Thus E is a finite endpoint seal for the accepted packet, not a host analytic object or a proof of ambient completeness. □

Theorem 1583.32 (Real power series root radius-majorant triad)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, root-test radius evidence must pass through the triad* R, W, S *and then through the majorant row* M *before the endpoint seal* E *is available. In particular, a* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *read cannot reach* E *directly from* R *and cannot bypass the finite coefficient windows or partial-sum row.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier root radius majorant triad`

Proof. Start with Definition 1583.11. It places the radius row R before the finite window row W, the partial-sum row S, and the majorant row M on every endpoint-facing route.

Apply Definition 1583.25. The root-test evidence is one of the allowed shapes of M, so it is consumed as majorant data rather than as a detached host convergence theorem.

Finally use Definition 1583.22. Since E is downstream of A, Z, X, R, W, S, M, root-radius evidence must pass through the majorant row before endpoint sealing. □

Theorem 1583.33 (Real power series root endpoint nonescape)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ *whose majorant row* M *is read through a* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *route, the endpoint row* E *exports no selected infinite sum, analytic continuation, quotient function space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness. The root-test route remains confined to the displayed rows of* K*.*

Proof. Read the endpoint scope theorem, Definition 1583.31. It says that E is available only after the route through $A, Z, X, R, W, S, M, H, C, P, N$.

Apply Definition 1583.25 to identify the root-test route as one possible reading of M. This does not add a new endpoint coordinate; it only selects the displayed majorant evidence.

Now use Definition 1583.23. The carrier excludes selected sums, analytic continuation, quotient function spaces, countable choice, and ambient completeness, so the root-test endpoint read cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesRootEndpointNonescape`

Theorem 1583.34 (Real power series root coefficient-window coverage)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, every coefficient read needed by a* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *majorant is already covered by the coefficient row* A*, the finite window row* W*, and the partial-sum row* S *before* M *or* E *is consumed. No root-test consumer may introduce a fresh coefficient stream outside the displayed carrier.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier root coefficient window coverage`

Proof. Start with Definition 1583.24. It forces every coefficient-facing read through A and then through the finite windows W before it contributes to S.

Use Definition 1583.11. The handoff puts S before the majorant row M, so the root-test route can consume only coefficient windows already displayed in the packet.

Finally apply Definition 1583.25. Since a root-test majorant is read as M, it receives coefficient evidence from A, W, S and cannot import a separate host coefficient stream or selected infinite series object. □

Theorem 1583.35 (Real power series evaluation-window exhaustion)**.** *Every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *evaluation consumer must pass through the coefficient or formal-series row A, the center and argument rows Z, X, the radius row R, the finite window ledger W, the partial-sum rows S, the selected majorant row M, the endpoint seal E, and the structural rows H, C, P, N. No accepted evaluation read may inspect analytic continuation, host infinite sums, quotient function spaces, selected sums, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness outside those rows.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier evaluation window exhaustion`

Proof. Project the accepted carrier through Definition 1583.1. The only evaluation-facing analytic rows are A, Z, X, R, W, S, M, E, followed by the structural rows H, C, P, N.

The coefficient-window obligation of Definition 1583.24 forces coefficient access through A and the displayed finite windows W. The radius-window handoff of Definition 1583.11 places R before the window and partial-sum rows.

Use Definition 1583.25 to bind convergence evidence to the selected majorant row M. Then Definition 1583.22 makes the endpoint seal E downstream of the same coefficient, center, argument, radius, window, partial-sum, and majorant route.

Finally apply Definition 1583.33. The forbidden objects have no row in the carrier, so an evaluation consumer has already exhausted every permitted read once it has traversed $A, Z, X, R, W, S, M, E, H, C, P, N$. □

Theorem 1583.36 (Real power series radius-majorant public export)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, the public radius-majorant evaluation surface is the route obtained by composing the coefficient-window ledger, radius-majorant coverage, CauchyProduct coefficient-budget, RegSeqRat endpoint handoff, and endpoint nonescape rows. A downstream consumer may cite R as public radius evidence only along*

$$A, Z, X, R \longmapsto W \longmapsto S \longmapsto M \longmapsto E \longmapsto H, C, P, N,$$

with product and convolution reads passing through the $\mathsf{CauchyProduct}^{\uparrow}$ *budget cited by Definition 1583.30. The export contains no host convergence theorem, selected infinite sum, analytic continuation, quotient function space, host multiplication primitive, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius majorant public export`

Proof. Project the local carrier. The carrier projection exposes $A, Z, X, R, W, S, M, E, H, C, P, N$ as the only rows available to a public evaluation read.

Apply Definition 1583.24 to place coefficient access inside A and W. Then use Definition 1583.19 to bind the radius row R to the finite window, partial-sum, selected majorant, and endpoint rows.

Next apply Definition 1583.30. Product and convolution consumers may use only the displayed CauchyProduct source windows, dyadic budget rows, product row, budget ledger, and classifier route before returning to the power-series endpoint.

Finally use Definition 1583.21 and Definition 1583.23. These rows put the $\mathsf{RegSeqRat}^{\uparrow}$ endpoint handoff and the Real seal boundary inside the same finite packet, so the listed excluded host and choice objects cannot be exported. □

Theorem 1583.37 (Real power series RegSeqRat evaluation handoff)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, every evaluation read handed to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint consumer factors through the coefficient-window ledger, the radius-majorant public export of Definition 1583.36, the CauchyProduct coefficient-budget route, and the endpoint seal boundary. The handoff is therefore a finite route from coefficient windows and radius evidence to regular-rational endpoint readback; it does not export host analytic continuation, quotient function-space equality, selected infinite sums, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier regseqrat evaluation handoff`

Proof. Begin with Definition 1583.36. It already composes the carrier projection, coefficient-window ledger, radius-majorant coverage, CauchyProduct budget route, RegSeqRat endpoint handoff, and Real seal boundary into one public evaluation surface.

Read the $\mathsf{RegSeqRat}^{\uparrow}$ endpoint side through Definition 1583.21. This forces endpoint-facing regular-rational data to wait until $A, Z, X, R, W, S, M, E, H, C, P, N$ have all been displayed.

Use Definition 1583.30 for product-sensitive evaluations. The product budget remains downstream of the same coefficient, radius, window, partial-sum, and selected-majorant rows, so it cannot introduce a host multiplication primitive or a separate coefficient source.

Finally apply Definition 1583.31 and Definition 1583.23. The endpoint seal is packet-local and the excluded analytic, quotient, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 1583.38 (Real power series root-unblock obligation surface)**.** *Every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet exposes exactly the finite coefficient stream row, the* $\mathsf{CauchyProduct}^{\uparrow}$ *multiplication ledger, the* $\mathsf{ConvRad}^{\uparrow}$ *radius row, the* $\mathsf{RealSeries}^{\uparrow}$ *summation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *finite-window reads, the* $\mathsf{WeierstrassMTest}^{\uparrow}$ *bound row, and the local* $\mathsf{NameCert}$ *route before any real-valued power-series consumer can inspect convergence. This surface is routed through chapter 464, chapter 66, chapter 1581, chapter 301, chapter 126, and chapter 399; it exports no analytic continuation, global holomorphicity, open-disk completeness, or host real equality.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier root unblock obligation surface`

Proof. Project the carrier $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ of Definition 1583.1. The coefficient stream is read through A and W, the radius data through R, the finite summation data through S, the selected majorant data through M, and the endpoint through E.

Use Definition 1583.36. It composes the coefficient-window ledger, radius-majorant coverage, CauchyProduct coefficient budget, RegSeqRat endpoint handoff, and endpoint nonescape rows into one consumer-facing route. The cited CauchyProduct sibling supplies the multiplication ledger, while the ConvergenceRadius, RealSeries, RegSeqRat, StreamName, and WeierstrassMTest siblings supply the named rows consumed by that route.

Finally apply Definition 1583.37. A real-valued endpoint consumer must pass through the same finite rows before convergence can be inspected. Since the carrier and the cited sibling routes have no coordinates for analytic continuation, global holomorphicity, open-disk completeness, or host real equality, those claims are refused at this obligation surface. □

Theorem 1583.39 (Real power series coefficient-tail obligations)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, every endpoint-facing read first exposes the coefficient stream row* A*, a finite tail window inside* W*, and the* $\mathsf{CauchyProduct}^{\uparrow}$ *ledger rows named by* P*. These rows are displayed before the endpoint seal* E *is available and before any consumer may inspect a power-series value.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient tail obligations`

Proof. Project the carrier of Definition 1583.1. The coefficient stream is the row A, while W carries the finite windows for coefficients, point readbacks, powers, and partial sums.

Use the product-sensitive route of Definition 1583.30. It forces endpoint-facing product or convolution reads to consume the coefficient row, finite window row, partial-sum row, and selected majorant row before the endpoint seal can be read.

Finally preserve the route through H, C, P, N. Since the $\mathsf{CauchyProduct}^{\uparrow}$ ledger is part of the displayed provenance route and not a host multiplication theorem, the endpoint read sees only finite coefficient-tail and product-budget rows. □

Theorem 1583.40 (Real power series radius-test window exhaustion)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, each radius-test read factors through the displayed* $\mathsf{ConvRad}^{\uparrow}$ *row* R *and then through a* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *or* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *window recorded in* M *and* W*. The route cannot inspect an endpoint seal, host radius theorem, or infinite coefficient tail before those finite windows are exhausted.*

Proof. Start with Definition 1583.11. It places the radius row R before endpoint-facing convergence evidence and routes that evidence through W, S, M.

Then apply Definition 1583.25. The selected majorant row must be one of the displayed M-test, ratio-test, or root-test routes, and the ratio/root alternatives are finite windows carried by the same packet.

The endpoint seal E remains downstream by Definition 1583.32 and Definition 1583.34. Thus a radius-test read is exhausted at the finite $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, or $\mathsf{RegularCauchyRootTest}^{\uparrow}$ windows before any endpoint value can be consumed. □

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius test window exhaustion`

Theorem 1583.41 (Real power series endpoint Cauchy budget)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *carrier* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, endpoint evaluation is routed through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback budgets,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows before it can be exported. The route exports no selected infinite sum, quotient function space, analytic continuation, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier endpoint cauchy budget`

Proof. Read the endpoint side through Definition 1583.21. It forces every $\mathsf{RegSeqRat}^{\uparrow}$-facing endpoint handoff to consume the finite windows, regular-rational budgets, dyadic tolerances, partial-sum rows, selected majorant rows, and terminal seal as one packet route.

Use Definition 1583.22. The endpoint seal factors through $A, Z, X, R, W, S, M, H, C, P, N$, so the $\mathsf{Real}^{\uparrow}$ row is reached only after the Cauchy-facing budget has been displayed.

Finally apply Definition 1583.23. The excluded analytic, quotient, choice, and ambient-completeness objects are not carrier fields and cannot be exported by endpoint evaluation. □

Theorem 1583.42 (Real power series product coefficient-window lock)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, every coefficient product read used for powers of* $X - Z$ *must pass through the coefficient row* A*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* W*, the finite partial-sum row* S*, the selected majorant row* M*, and the* $\mathsf{CauchyProduct}^{\uparrow}$ *product route cited by the provenance row* P*. Such a read cannot use host multiplication, quotient functions, selected infinite sums, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness outside the displayed carrier.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier product coefficient window lock`

Proof. Project the carrier of Definition 1583.1. Coefficient access is stored in A, and the finite windows for coefficient, point, power, and partial-sum reads are stored in W before S or M can be consumed.

Apply Definition 1583.24 and Definition 1583.30. These theorems route product-sensitive coefficient reads through the displayed coefficient windows, partial-sum row, selected majorant row, and CauchyProduct source windows before returning to the power-series packet.

Finally use Definition 1583.23. Since host multiplication, quotient function spaces, selected infinite sums, and ambient completeness are not carrier fields, the coefficient product read remains locked to the finite product route. □

Theorem 1583.43 (Real power series product endpoint nonescape)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal for a power-series evaluation is downstream of the* $\mathsf{CauchyProduct}^{\uparrow}$ *product route together with* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealSeries}^{\uparrow}$*,* $\mathsf{WeierstrassMTest}^{\uparrow}$*, ratio-test, and root-test majorant rows. The endpoint cannot escape from that route to host analytic continuation, quotient function equality, selected infinite sums, host multiplication, countable choice, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier product endpoint nonescape`

Proof. Begin with Definition 1583.42. It places every product coefficient read on the route through A, W, S, M and the CauchyProduct product surface before the endpoint seal is available.

Use Definition 1583.36. That theorem composes the coefficient-window ledger, radius-majorant coverage, CauchyProduct coefficient budget, RegSeqRat endpoint handoff, and endpoint nonescape rows into the public evaluation surface.

Now apply Definition 1583.41. The endpoint seal is read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, partial sums, selected majorants, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The majorant row may be supplied by $\mathsf{WeierstrassMTest}^{\uparrow}$, ratio-test, or root-test evidence, but it remains a displayed row of M.

The excluded analytic, quotient, multiplication, choice, and ambient-completeness objects have no carrier coordinate. Thus the terminal endpoint remains downstream of the product and regular-rational route. □

Theorem 1583.44 (Real power series radius composition)**.** *Let* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$ *and* $K' = (A', Z', X', R', W', S', M', E', H', C', P', N')$ *be accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packets whose displayed radius windows are compatible through their* $\mathsf{ConvRad}^{\uparrow}$ *rows and whose product-side coefficient windows are admitted by the same* $\mathsf{CauchyProduct}^{\uparrow}$ *budget route. Then any product-side radius consumer reads only the common radius-window and coefficient-window route*

$$A, A', R, R' \longmapsto W, W' \longmapsto S, S' \longmapsto M, M' \longmapsto E, E'$$

together with the structural rows of the two packets. The composed read may cite $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *only through those displayed rows, and it exports no host*

product of infinite sums, selected limit, analytic-continuation equality, countable choice, or ambient Real-completeness claim.

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier radius composition`

Proof. First apply Definition 1583.11 to both packets. Each endpoint-facing read must expose its coefficient, center, argument, radius, finite-window, partial-sum, selected-majorant, and endpoint rows before any radius consumer can inspect the seal.

Next use Definition 1583.14. The product request is a finite $\mathsf{CauchyProduct}^{\uparrow}$ budget inside the displayed partial-sum route, so it can pair only coefficient windows already exposed by W, W'.

Finally apply Definition 1583.22 and Definition 1583.43. The endpoint seal remains downstream of the common window route and the excluded host, selector, analytic-continuation, choice, and completeness objects have no carrier coordinate. □

Theorem 1583.45 (Real power series coefficient-product-radius exhaustion)**.** *Let $K = (A, Z, X, R, W, S, M, E, H,$ be an accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet, and let a downstream endpoint-facing consumer request an evaluation, product evaluation, or radius-limited read of K. Then the consumer must first expose the coefficient windows A, W; when products are present it must also expose a displayed* $\mathsf{CauchyProduct}^{\uparrow}$ *ledger B for the product coefficients cited by P; it must then expose the radius or subradius row R, the selected majorant row M, the partial-sum rows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback route, and only then the endpoint seal E. The resulting read exports no host product of infinite sums, endpoint selector, analytic-continuation equality, quotient function equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier coefficient product radius exhaustion`

Proof. Begin with the coefficient-window ledger. Theorems Definition 1583.24 and Definition 1583.16 force every evaluation or product request to display the coefficient source A and the finite windows W before the endpoint row can be inspected.

For product-facing requests, apply Definition 1583.14 and Definition 1583.42. These place the $\mathsf{CauchyProduct}^{\uparrow}$ product ledger on the same finite coefficient-window route and prevent a product consumer from reading undisplayed product coefficients.

Next read the radius side. The radius-window handoff and radius-majorant coverage theorems require the radius or subradius row R and the selected majorant row M before the partial sums and endpoint seal are consumed.

Finally apply Definition 1583.22, Definition 1583.29, and Definition 1583.44. The partial-sum rows S, the regular-rational and dyadic readback route, and the endpoint seal E are downstream of the displayed coefficient, product, radius, and majorant rows. Since the excluded analytic, quotient, selector, choice, and completeness objects have no carrier coordinate, the listed route exhausts every endpoint-facing read. □

Theorem 1583.46 (Real power series obligation export)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$, the downstream obligation package exported to* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers is the conjunction of the radius obligation, coefficient-window obligation,* $\mathsf{CauchyProduct}^{\uparrow}$ *handoff, majorant route,* $\mathsf{RegSeqRat}^{\uparrow}$ *evaluation handoff, endpoint seal, and product coefficient-window lock. Every consumer reads this package through $A, Z, X, R, W, S, M, E, H$ together with* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$, *and no consumer receives host analytic continuation, quotient function equality, selected infinite sums, host multiplication, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier obligation radius export`

Proof. Begin with the carrier and NameCert surface, Definition 1583.1 and Definition 1583.10. These rows expose the coefficient source, center, argument, radius, finite windows, partial sums, majorant data, endpoint seal, transport, continuation, provenance, and local naming.

The radius and coefficient obligations are supplied by Definition 1583.11, Definition 1583.24, and Definition 1583.16. Together they force a consumer to read A, R, W before endpoint or product data.

The product side is supplied by Definition 1583.14, Definition 1583.42, and Definition 1583.45. These theorems place the $\mathsf{CauchyProduct}^{\uparrow}$ handoff on the same finite coefficient-window and radius route.

Finally apply Definition 1583.36, Definition 1583.22, and Definition 1583.43. They bind the majorant route, regular-rational evaluation handoff, and endpoint seal, while excluding analytic continuation, quotient equality, selected sums, host multiplication, choice, and ambient completeness. The cited rows therefore form the exported obligation package. □

Theorem 1583.47 (Real power series obligation closure ledger)**.** *For every accepted* $\mathsf{RealPowerSeries}^{\uparrow}$ *packet* $K = (A, Z, X, R, W, S, M, E, H, C, P, N)$*, the obligation package is exactly the carrier admission row, coefficient-window route, radius-window handoff,* $\mathsf{CauchyProduct}^{\uparrow}$ *budget, selected majorant route,* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint handoff, endpoint seal, transport, replay, provenance, and local naming rows displayed by this chapter. No obligation row outside* $A, Z, X, R, W, S, M, E, H, C, P, N$ *is exported to an endpoint-facing, product-facing, or radius-facing consumer.*

Lean4 ✓ `BEDC.Derived.RealPowerSeriesUp.RealPowerSeriesCarrier obligation closure ledger`

Proof. Carrier admission and local naming are fixed by Definition 1583.1 and Definition 1583.10. These rows expose exactly the coefficient stream or formal-series row, center and argument rows, radius row, finite windows, partial sums, selected majorant route, endpoint seal, transport, replay, provenance, and local NameCert row.

The analytic obligations are the existing ledger rows. The coefficient-window route is Definition 1583.24; the radius-window handoff is Definition 1583.11; the $\mathsf{CauchyProduct}^{\uparrow}$ budget is Definition 1583.30; the selected majorant route is Definition 1583.27; and the $\mathsf{RegSeqRat}^{\uparrow}$ endpoint handoff is Definition 1583.21.

The endpoint and structural rows are sealed by Definition 1583.22, Definition 1583.41, Definition 1583.43, and Definition 1583.46. Together with Definition 1583.23, these results leave no carrier coordinate for analytic continuation, quotient function equality, selected infinite sums, host multiplication, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the listed rows are the complete obligation package. □

Falsifiable prediction: Every $\mathsf{RealPowerSeries}^{\uparrow}$ endpoint-facing read factors through a coefficient or FormalPowerSeries row, center Real row, argument Real row, displayed radius or subradius row, finite StreamName windows, partial-sum RealSeries row, majorant data from M-test or ratio/root-test rows, endpoint Real seal, transport, continuation, provenance, and local naming.

Independence witness: realseries, weierstrass-mtest, regular-cauchy-ratio-test, regular-cauchy-root-test, formalpowerseries, convergence-radius, real, regseqrat, streamname, dyadicratcore: $\mathsf{RealPowerSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries coefficient-stream access, center and argument Real rows, displayed radius windows, finite polynomial partial sums, selected majorant routes, endpoint sealing, transport, continuation, provenance, and local naming.

Theorem 1583.48 (RealPowerSeries scoped radius ledger)**.** *The scoped surface of* $\mathsf{RealPowerSeries}^{\uparrow}$ *is exactly the displayed coefficient-window carrier, radius witness,* $\mathsf{CauchyProduct}^{\uparrow}$ *handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay, and* Pkg *provenance recorded by the carrier*

$$K = (A, Z, X, R, W, S, M, E, H, C, P, N).$$

Every endpoint-facing, product-facing, or radius-facing consumer must read the packet through $A, Z, X, R, W, S, M, E, H, C, P, N$ *before it may use the endpoint seal, and no consumer receives host analytic continuation, quotient function equality, selected infinite sums, host multiplication, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as an additional coordinate.*

Proof. Project the carrier of Definition 1583.1. The rows $A, Z, X, R, W, S, M, E, H, C, P, N$ are the only rows available to a consumer, so the coefficient source, center, argument, radius witness, finite windows, partial sums, majorant route, endpoint seal, transport, replay, provenance, and local naming are the visible scope boundary.

Apply Definition 1583.47. It identifies carrier admission, the coefficient-window route, the radius-window handoff, the $\mathsf{CauchyProduct}^{\uparrow}$ budget, selected majorant route, $\mathsf{RegSeqRat}^{\uparrow}$ endpoint handoff, endpoint seal, transport, replay, provenance, and local naming as the complete obligation package.

Finally use Definition 1583.45 and Definition 1583.43. They keep product and endpoint requests downstream of the same coefficient, radius, tolerance, readback, and seal rows, while the excluded analytic, quotient, selector, choice, and completeness objects have no carrier coordinate. Hence the displayed rows are exactly the scoped ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealPowerSeries}^{\uparrow}$ packet is generated from a coefficient stream or formal-series row, center and argument $\mathsf{Real}^{\uparrow}$ rows, a displayed radius or subradius row, finite $\mathsf{StreamName}^{\uparrow}$ windows, finite partial-sum and $\mathsf{RealSeries}^{\uparrow}$ rows, majorant data from M-test or ratio/root-test rows, a $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1583.1, Definition 1583.10, Definition 1583.47, and Definition 1583.48 bind the coefficient-window carrier, radius witness, product handoff, majorant route, $\mathsf{RealSeries}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks, terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, local naming, and nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealPowerSeries}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, quotient function spaces, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Lean verification and bridge checking remain for the finite coefficient-window, radius, Cauchy-product, majorant, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ seal route displayed by this chapter.

Chapter 1584

A Concrete Naming Certificate for NestedIntervalCompactness$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the NestedIntervalCompactness$^{\uparrow}$ packet records the finite-window form of nested-interval compactness before any general Heine–Borel or open-cover principle is available. It reads the nested interval carrier of chapter 320, the located endpoint surface of chapter 394, the dyadic interval ledger of chapter 306, finite stream observations from chapter 126, regular rational readback from chapter 301, and the terminal real seal of chapter 38. The packet gives Real-facing countable-name arguments a finite nested-window certificate; it does not select real endpoints, identify quotient real names, invoke countable choice, or assert compactness for arbitrary covers.

Definition 1584.1 (NestedIntervalCompactness carrier)**.** *A* NestedIntervalCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* NestedInterval$^{\uparrow}$ *row naming the inspected finite prefix of nested intervals.*
2. *L is the* LocatedInterval$^{\uparrow}$ *endpoint-location row for the same displayed prefix.*
3. *D is the* DyadicInterval$^{\uparrow}$ *row containing the dyadic endpoint boxes, inclusion checks, and width comparisons used by the finite prefix.*
4. *W is the* StreamName$^{\uparrow}$ *window row through which candidate real-name observations are read at the displayed precisions.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from the windows and dyadic boxes.*
6. *E is the* Real$^{\uparrow}$ *seal boundary reached only after I, L, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for nested intervals, located endpoints, dyadic boxes, stream windows, regular readback, the real seal, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite prefix request through the located interval row, dyadic interval row, stream windows, regular readback, and the real seal boundary.*
9. *P is the* Pkg *provenance over* NestedIntervalCompactness$^{\uparrow}$, NestedInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCompactness}^{\uparrow}}$ *row naming the displayed finite-window packet and its permitted consumer reads.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for a general open-cover principle, a Heine–Borel theorem, a selected host endpoint, quotient real equality, countable choice, or an infinite intersection object outside the displayed finite windows.

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessTasteGate single carrier alignment`

Theorem 1584.2 (NestedIntervalCompactness finite-intersection window)**.** *Let $K = (I, L, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *carrier. Every finite prefix request for the nested family reads the displayed route*

$$I, L, D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the finite-window intersection witness available to a Real-facing consumer is the regular-rational readback over the same displayed dyadic and located interval rows. The witness is not a selected point of the full countable intersection and is not an open-cover compactness certificate.

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier finite intersection window`

Proof. Project the carrier of Definition 1584.1. The prefix request first exposes I, then the located endpoint row L, and then the dyadic interval row D for the same finite prefix.

The continuation row C routes the request from those interval rows into the stream windows W and the regular rational readback R. Thus the Real-facing observation is read from displayed finite data rather than from an ambient completeness or compactness principle.

Finally E, H, P, N seal, transport, source, and name that route. Since no row contains a selected endpoint of the full intersection, quotient real equality, countable choice, or open-cover data, the finite prefix witness is exactly the displayed finite-window route. □

Theorem 1584.3 (NestedIntervalCompactness NameCert obligations)**.** *For every accepted* NestedIntervalCompactn *carrier $K = (I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCompactness}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the nested interval prefix, located endpoints, dyadic interval ledger, stream windows, regular rational readback, real seal boundary, transport, continuation replay, provenance, and local naming row;*
2. *finite-prefix monotonicity, requiring every accepted read to consume the same displayed nesting and inclusion cells in I and D;*
3. *located dyadic exactness, requiring the endpoint-location row L and dyadic interval row D to govern the same interval prefix before any readback row R is exposed;*
4. *Real seal routing, requiring E to remain downstream of I, L, D, W, R and upstream of only the structural rows H, C, P, N;*
5. *ledger non-escape, excluding general Heine–Borel compactness, open-cover compactness, selected real endpoints, quotient real equality, countable choice, and detached infinite-intersection objects.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier namecert obligations`

Proof. Carrier admission is the row projection of Definition 1584.1. It exposes exactly $I, L, D, W, R, E, H, C, P, N$.

Finite-prefix monotonicity is read from I together with the inclusion cells in D. Located dyadic exactness then binds the endpoint-location row L to the same dyadic interval prefix before W and R can be consumed.

The seal row E is downstream of those displayed rows, while H, C, P, N transport, replay, source, and name only that finite route. The excluded compactness, choice, selected-endpoint, quotient-equality, and infinite-intersection coordinates therefore do not belong to the local NameCert surface. □

Theorem 1584.4 (NestedIntervalCompactness finite-prefix monotonicity)**.** *Let $K = (I, L, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *carrier. Any consumer extending the displayed finite prefix must preserve the same nesting cells in I and the same inclusion cells in D before it reads the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, or the* $\mathsf{Real}^{\uparrow}$ *seal row E. The extension therefore introduces no selected endpoint, quotient equality, countable choice, or open-cover compactness row.*

Proof. Project the carrier of Definition 1584.1. The rows visible before any analytic read are the nested interval prefix I, the located endpoint row L, and the dyadic interval ledger D.

Apply the finite-prefix monotonicity row from Definition 1584.3. It requires each accepted consumer read to preserve the displayed nesting and inclusion cells in I and D when the finite prefix is extended.

Only after those cells are fixed may the route expose W, R, and E. Because the local obligation surface contains no row for selected endpoints, quotient equality, countable choice, or open-cover compactness, extending the prefix cannot introduce any such coordinate. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier finite prefix monotonicity`

Theorem 1584.5 (NestedIntervalCompactness Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *packet factors through a displayed finite nested-window route*

$$\mathsf{NestedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The exported boundary is therefore a finite-window compactness-shaped certificate for nested intervals, not a proof of general Heine–Borel compactness, not compactness of arbitrary open covers, not a quotient-real equality principle, and not a countable-choice selector for endpoints.

Proof. Start with a consumer of the terminal $\mathsf{Real}^{\uparrow}$ boundary. By Definition 1584.2, the consumer reaches E only through I, L, D, W, R for the displayed finite prefix.

Apply Definition 1584.3. The local obligations require the same finite prefix, located endpoint row, dyadic ledger, stream windows, and regular readback to govern every accepted Real-facing read.

Thus the exported Real boundary is exhausted by the displayed nested-window route and the structural rows H, C, P, N. Any general cover compactness theorem, selected endpoint, quotient equality, countable-choice selector, or detached infinite-intersection object lies outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier real seal boundary`

Theorem 1584.6 (NestedIntervalCompactness located endpoint scope)**.** *For every accepted* NestedIntervalCompact *carrier* $K = (I, L, D, W, R, E, H, C, P, N)$*, the located endpoint row* L *is scoped by the same displayed* $\mathsf{NestedInterval}^{\uparrow}$ *prefix* I *and the same* $\mathsf{DyadicInterval}^{\uparrow}$ *ledger* D *before any* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, or* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. Thus a consumer reads located endpoints only as finite packet data, not as selected real endpoints or quotient real equality.*

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier located endpoint scope`

Proof. Project I, L, D from the carrier of Definition 1584.1. The located endpoint row is not an independent real object; it is listed as the endpoint-location row for the same displayed prefix named by I.

Apply Definition 1584.3. The located dyadic exactness obligation requires L and D to govern the same interval prefix before a readback row is exposed, and the finite-prefix monotonicity obligation preserves the displayed nesting cells in I and D under extension.

Finally apply Definition 1584.5. Any Real-facing read reaches E only after the route through $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ has been displayed. The located endpoint scope is therefore finite and packet-local. □

Theorem 1584.7 (NestedIntervalCompactness dyadic mesh exhaustion)**.** *Let* $K = (I, L, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *carrier. Every finite nested-window read that reaches* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *has already exhausted the displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *mesh row* D *over the same* $\mathsf{NestedInterval}^{\uparrow}$ *and* $\mathsf{LocatedInterval}^{\uparrow}$ *rows* I, L*. No later consumer can replace that mesh by open-cover compactness, countable choice, trichotomy, or an ambient completeness row.*

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier dyadic mesh exhaustion`

Proof. Start with a read reaching W, R, or E. By Definition 1584.2, the read travels through the route $I, L, D \longmapsto W \longmapsto R \longmapsto E$ for the displayed finite prefix.

Use Definition 1584.4. Extending the prefix preserves the nesting cells in I and inclusion cells in D before any stream, regular-rational, or real-seal row can be read.

The NameCert obligation theorem, Definition 1584.3, supplies the ledger non-escape clause. Since the carrier has no row for open-cover compactness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the dyadic mesh row D is the whole finite mesh surface available to the later consumer. □

Theorem 1584.8 (NestedIntervalCompactness located dyadic scope)**.** *For every accepted* NestedIntervalCompactne *carrier* $K = (I, L, D, W, R, E, H, C, P, N)$*, each located endpoint read is scoped by the same* $\mathsf{NestedInterval}^{\uparrow}$ *prefix* I *and the same* $\mathsf{DyadicInterval}^{\uparrow}$ *mesh row* D *before any* $\mathsf{StreamName}^{\uparrow}$ *window* W *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is exposed. Thus the located data consumed by a* $\mathsf{Real}^{\uparrow}$ *boundary read is finite packet data over the displayed prefix, not a selected endpoint or quotient-real equality.*

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier located dyadic scope`

Proof. Start with a located endpoint read. The carrier definition exposes L only as the endpoint-location row for the prefix named by I, while the dyadic mesh row D records the finite boxes and inclusion checks for that same prefix.

Apply the NameCert obligation theorem. Its located dyadic exactness clause binds L and D to the same interval prefix before a readback row can be consumed, and its finite-prefix monotonicity clause preserves those nesting and inclusion cells when the prefix is extended.

Finally use Definition 1584.7. Any read reaching W, R, or E has already exhausted the displayed dyadic mesh over I, L. Therefore the located endpoint scope is the displayed finite prefix and not an external endpoint choice. □

Theorem 1584.9 (NestedIntervalCompactness prefix readback coverage)**.** *Every finite prefix request reaching the* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *packet has already displayed the nested interval prefix I, located endpoint row L, dyadic mesh row D,* $\mathsf{StreamName}^{\uparrow}$ *windows W, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R. No Real-facing consumer can bypass those rows by appealing to open-cover compactness, countable choice, quotient real equality, or an ambient completeness row.*

Proof. Begin with a finite prefix request that reaches E. By the finite-intersection window theorem, the route to the Real seal is

$$I, L, D \longmapsto W \longmapsto R \longmapsto E.$$

Use prefix monotonicity to keep the displayed nesting cells in I and inclusion cells in D fixed under prefix extension. The located endpoint scope theorem then places L on the same prefix and mesh before stream or regular-rational reads are exposed.

The real seal boundary theorem identifies E as downstream of the displayed route, and the dyadic mesh exhaustion theorem rules out replacing D by a later compactness or choice row. Hence every finite prefix request reaching $\mathsf{Real}^{\uparrow}$ has already displayed I, L, D, W, R inside the packet. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier prefix readback coverage`

Theorem 1584.10 (NestedIntervalCompactness stream-to-real route)**.** *For every accepted* NestedIntervalCompactn *carrier $K = (I, L, D, W, R, E, H, C, P, N)$, each* $\mathsf{StreamName}^{\uparrow}$ *observation window W used by the packet is routed through* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed, and the route remains downstream of the displayed* $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, *and* $\mathsf{DyadicInterval}^{\uparrow}$ *rows I, L, D.*

Proof. Project the route rows from Definition 1584.1. The continuation row C records replay from a finite prefix request through the located interval row, dyadic interval row, stream windows, regular readback, and the real seal boundary.

Apply Definition 1584.2. The theorem identifies W as the window row reached only after I, L, D, and identifies R as the readback row reached before E.

The Real seal boundary theorem, Definition 1584.5, then fixes the exported boundary as the displayed finite route through $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Thus no stream observation bypasses the regular-rational readback or the interval rows that generated it. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier stream real route`

Theorem 1584.11 (NestedIntervalCompactness finite-window exhaustion)**.** *Every accepted* NestedIntervalCompact *consumer of a* $\mathsf{Real}^{\uparrow}$*-facing nested interval read factors through the finite-prefix monotonicity row, located endpoint scope, dyadic mesh exhaustion, prefix readback coverage, and stream-to-real route before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The visible route is therefore exhausted by the finite packet rows* $I, L, D, W, R, E, H, C, P, N$*, with no exported selected endpoint, quotient real equality, countable choice, arbitrary open-cover compactness, or full countable-intersection principle.*

Proof. Start with a consumer read that reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. By Definition 1584.10, the read reaches E only after the $\mathsf{StreamName}^{\uparrow}$ window W has passed through the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and that route remains downstream of the displayed interval rows I, L, D.

The finite-prefix monotonicity theorem, Definition 1584.4, keeps the nesting cells in I and the inclusion cells in D fixed under prefix extension. The located endpoint scope theorem, Definition 1584.6, binds L to the same displayed prefix before any stream, readback, or real-seal row is consumed.

Use Definition 1584.7 to exhaust the mesh row D over I, L, and then apply Definition 1584.9. These two rows force every Real-facing finite prefix request to display I, L, D, W, R before E is visible.

Thus the consumer sees exactly the finite-window route carried by $K = (I, L, D, W, R, E, H, C, P, N)$. The excluded endpoint-selection, quotient-equality, choice, arbitrary open-cover, and full countable-intersection coordinates have no carrier row from which they could be exported. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier finite window exhaustion`

Theorem 1584.12 (NestedIntervalCompactness scoped package)**.** *The carrier, finite-intersection window, NameCert obligations, prefix monotonicity, Real seal boundary, located endpoint scope, dyadic mesh exhaustion, located dyadic scope, prefix readback coverage, stream-to-real route, and finite-window exhaustion bind the complete paper scope of the* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *packet over* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Consequently any accepted consumer read must pass through the displayed finite nested-window rows $I, L, D, W, R, E, H, C, P, N$ *before it can use the local certificate. The scoped package exports no open-cover compactness principle, selected point of the full intersection, quotient equality of Real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 1584.1. It lists exactly the nested interval prefix I, located endpoint row L, dyadic interval ledger D, stream windows W, regular readback R, Real seal E, and structural rows H, C, P, N.

Apply Definition 1584.2, Definition 1584.3, and Definition 1584.4. These rows force every finite prefix request to preserve the displayed nesting and inclusion cells before any stream, regular-rational, or Real-facing observation is consumed.

Next apply Definition 1584.5, Definition 1584.6, and Definition 1584.7. They place the Real seal downstream of the same located and dyadic interval rows and exhaust the mesh surface available to later consumers.

Finally use Definition 1584.8, Definition 1584.9, Definition 1584.10, and Definition 1584.11. These theorems identify the whole accepted route as the finite packet $I, L, D, W, R, E, H, C, P, N$ over the named BEDC rows.

Since no cited row contains open-cover data, a selected point of the full intersection, quotient equality of Real names, or an ambient completeness field, the package scope is precisely the displayed finite nested-window route. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier scope package`

Theorem 1584.13 (NestedIntervalCompactness public certificate)**.** *The public* NestedIntervalCompactness$^{\uparrow}$ *certificate is exactly the finite route*

$$I, L, D, W, R, E, H, C, P, N$$

over NestedInterval$^{\uparrow}$*,* LocatedInterval$^{\uparrow}$*,* DyadicInterval$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. It exports a displayed nested-window compactness packet and no open-cover compactness theorem, selected endpoint, quotient equality of real names, countable choice, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the scoped package of Definition 1584.12. It already binds every accepted consumer read to the carrier rows $I, L, D, W, R, E, H, C, P, N$ over the named interval, stream, regular-rational, and real-seal surfaces.

Use the finite-window exhaustion theorem, Definition 1584.11. It prevents any Real-facing read from bypassing the finite-prefix monotonicity, located endpoint scope, dyadic mesh, prefix readback, and stream-to-real route.

The public certificate is therefore the same displayed route, now exposed as the chapter-level object. Since the carrier has no row for open covers, endpoint selection, quotient real equality, countable choice, or ambient completeness, none of those principles is exported. □

Lean4 ✓ `BEDC.Derived.NestedIntervalCompactnessUp.NestedIntervalCompactnessCarrier public certificate`

Closure status for NameCert$_{\text{NestedIntervalCompactness}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{NestedIntervalCompactness}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1584.1, Definition 1584.2, Definition 1584.3, Definition 1584.4, Definition 1584.5, Definition 1584.6, Definition 1584.7, Definition 1584.8, Definition 1584.9, Definition 1584.10, Definition 1584.11, Definition 1584.12, and Definition 1584.13 bind the mature paper-side finite-window bridge data from displayed located and dyadic interval rows through StreamName, RegSeqRat, and Real seal reads.

Formal verification: FormalStatus(NameCert$_{\text{NestedIntervalCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, general Heine–Borel compactness, compactness of arbitrary open covers, selected real endpoints, quotient real equality, countable choice, or full countable intersections.

Upgrade path: Lean verification and bridge checking require checked carrier admission, finite-prefix monotonicity, located dyadic exactness, Real seal routing, and ledger non-escape for the same finite-window bridge packet.

Constructive story: A NestedIntervalCompactness$^{\uparrow}$ packet is generated from a nested interval prefix, located endpoint rows, dyadic interval boxes, StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1585

A Concrete Naming Certificate for CauchyCompletionReflector$^{\uparrow}$

Following Bishop and Bridges constructive completion, the CauchyCompletionReflector$^{\uparrow}$ packet records the public finite reflector surface for the L10 completion route. It gathers the unit surface of chapter 1527, the counit surface of chapter 1539, the idempotent surface of **??**, the extension route of chapter 397, and the reflection boundary of chapter 403. Its local real-facing rows are read through chapter 360, chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a finite reflector certificate; it is not a quotient completion, not a selected limit, not host equality of completed objects, not countable choice, and not ambient completeness.

Definition 1585.1 (CauchyCompletionReflector carrier)**.** *A* CauchyCompletionReflector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, U, Q, I, E, R, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row for the displayed metric, regular-Cauchy, or real-name data being sent toward completion.*
2. *M is the completion-object row, read through the* RealCauchyCompletion$^{\uparrow}$ *and* MetricCompletion$^{\uparrow}$ *surfaces rather than through a host quotient carrier.*
3. *U is the* CauchyCompletionUnit$^{\uparrow}$ *row admitting S into the completion-facing surface.*
4. *Q is the* CauchyCompletionCounit$^{\uparrow}$ *row returning already completed or real-sealed reads through the displayed regular route.*
5. *I is the* CauchyCompletionIdempotent$^{\uparrow}$ *row recording that a second completion pass reuses the same completion-object surface.*
6. *E is the* CompletionUniversalProperty$^{\uparrow}$ *extension row for uniformly or continuously routed consumers attached to the same unit route.*
7. *R is the* CompletionReflection$^{\uparrow}$ *row binding the construction, extension, separated boundary, and real-seal rows into one reflection surface.*
8. *T is the local transport row through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *windows used by U, Q, I, E, R.*
9. *H records componentwise* hsame *transport for source, completion object, unit, counit, idempotence, extension, reflection, local transport, continuation, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a source read through unit admission, completion-object access, counit return, idempotent replay, extension use, reflection, and the local real-name route.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}$ *row.*

The classifier on accepted packets compares only these twelve rows under the displayed transport and replay data. It has no coordinate for quotient completions, selected limits, host equality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorTasteGate single carrier alignment`

Theorem 1585.2 (CauchyCompletionReflector NameCert obligations)**.** *For every accepted* CauchyCompletionRefle *carrier $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed source, completion object, unit, counit, idempotence, universal-property extension, reflection, local transport, continuation, provenance, and naming rows;*
2. *unit admission, requiring every completion-facing source read to pass through U before M can be consumed;*
3. *counit return, requiring every completed read that is already real-sealed to return through Q before the terminal* $\mathsf{Real}^{\uparrow}$ *route is exposed;*
4. *idempotent replay, requiring a repeated completion request to reuse I and the same completion-object row M;*
5. *extension and reflection compatibility, requiring every universal-property consumer to read E and every reflection consumer to read R over the same S, M, U, Q, I, T route;*
6. *reflector non-escape, excluding quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

No reflector obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket semantic name certificate`

Proof. Carrier admission is projection from Definition 1585.1. The packet exposes the source, completion object, unit, counit, idempotence, extension, reflection, local transport, continuation, provenance, and naming rows, and no other rows.

The unit and counit obligations are then read in order. A completion-facing source read must first pass through the unit row U before M is available, while an already completed real-facing read returns through Q before the $\mathsf{Real}^{\uparrow}$ route carried by T can be consumed.

Idempotent replay uses the idempotent row I. Since I is stored beside the same M, U, Q, T rows, a second completion pass has no independent completion-object coordinate to inspect.

Finally, the extension row E and reflection row R are downstream of the same displayed route. Componentwise transport by H, replay by C, provenance by P, and naming by N preserve exactly that finite route, so quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached real completeness cannot be exported. □

Theorem 1585.3 (CauchyCompletionReflector unit-counit route)**.** *Let $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. Every public reflector consumer factors through the finite route*

$$S \longmapsto U \longmapsto M \longmapsto Q \longmapsto I \longmapsto E, R \longmapsto T \longmapsto H, C, P, N.$$

The route exposes the unit and counit faces of completion, replays idempotence, and then serves universal-property and reflection consumers through the same real-name transport. It does not supply quotient completion equality, a selected limit, host equality of completed objects, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket unit counit route`

Proof. Begin with a public reflector consumer. The source row S is visible only as the source of the displayed unit route U, so the consumer first reads unit admission before it can inspect the completion-object row M.

The counit row Q supplies the return route for reads that are already on the real-seal side. This return is tied to the same local transport row T, whose $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ windows are the only real-facing rows available to the packet.

Next read the idempotent row I. It states that repeating the completion pass reuses the same completion-object row and the same local transport, so the consumer cannot branch to a second completed carrier.

The universal-property row E and reflection row R are finally consumed over that fixed unit-counit-idempotent route. The structural rows H, C, P, N transport, replay, source, and name only the displayed packet, which proves the factorization and the stated non-escape boundary. □

Theorem 1585.4 (CauchyCompletionReflector unit-counit obligation)**.** *Let $K = (S, M, U, Q, I, E, R, T, H, C, P, N$ be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. The unit-counit surface is obligation-complete exactly at the rows S, U, M, Q, T: every public source read must expose S and U before it consumes the completion-object row M, and every already completed or real-sealed read must return through Q and the shared local transport row T before any reflector consumer receives the result. The route exports no quotient completion, selected limit, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket unit counit obligation`

Proof. Project the accepted carrier of Definition 1585.1. The source, completion object, unit, counit, and local transport rows are displayed as S, M, U, Q, T.

The unit direction reads S first and then U. Only after that admission is the completion-object row M available to a public reflector consumer.

The counit direction reads Q for already completed or real-sealed data and then the same local transport row T. Since T is the only route to the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ windows, the return cannot detach from the displayed finite carrier.

The structural rows H, C, P, N preserve this order and add no excluded completion or choice coordinate. Thus S, U, M, Q, T are exactly the unit-counit obligation surface. □

Theorem 1585.5 (CauchyCompletionReflector idempotent-extension obligation)**.** *Let $K = (S, M, U, Q, I, E, R, T,$ be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. The idempotent-extension part of the reflector*

surface is exactly the route through I, E, R over the same S, U, M, Q, T rows: repeated completion reuses I and the displayed completion-object row, extension consumers read E only after the unit-counit route is visible, and reflection consumers read R only over that same finite route. No second completion carrier, selected limit, host equality, countable choice, ambient completeness, or detached real-completeness row is introduced.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket idempotent extension obligation`

Proof. Start with the unit-counit obligation of Definition 1585.4. It fixes the public completion-facing route at S, U, M, Q, T.

The idempotent row I is read over this route. A repeated completion pass therefore reuses the displayed completion-object row M rather than branching to a second carrier.

The extension row E and reflection row R are downstream of the same route. The extension consumer cannot bypass U, Q, T, and the reflection consumer receives only the already displayed unit, counit, idempotent, extension, and local transport rows.

Finally H, C, P, N transport, replay, source, and name that route. Because the carrier contains no second completion object or excluded host-side coordinate, the stated non-escape boundary follows. □

Theorem 1585.6 (CauchyCompletionReflector extension-reflection scope)**.** *Let $K = (S, M, U, Q, I, E, R, T, H, C,$ be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. Every* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension read and every* $\mathsf{CompletionReflection}^{\uparrow}$ *read must expose the common route $S, U, M, Q, I, T, H, C, P, N$ before E or R is consumed. Neither row supplies a second completion carrier, selected limit, quotient equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket extension reflection scope`

Proof. Project the carrier of Definition 1585.1. The source, completion-object, unit, counit, idempotent, local transport, and structural rows are displayed as $S, M, U, Q, I, T, H, C, P, N$.

Apply the unit-counit obligation of Definition 1585.4. It fixes the shared source and return route before any extension or reflection consumer may inspect its own row.

Next apply the idempotent-extension obligation of Definition 1585.5. The idempotent row I, extension row E, and reflection row R are read over the same unit-counit surface rather than over an independent completion object.

The structural rows H, C, P, N preserve that route through componentwise transport, replay, provenance, and local naming. Since the carrier contains no coordinate for the excluded objects, the extension and reflection reads expose the stated common scope before E or R is consumed. □

Theorem 1585.7 (CauchyCompletionReflector public obligation package)**.** *For every accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier*

$$K = (S, M, U, Q, I, E, R, T, H, C, P, N),$$

every public reflector consumer receives exactly the route through the source row S, completion-object row M, unit row U, counit row Q, idempotence row I, extension row E, reflection row R, local transport row T, componentwise hsame *transport row H, displayed* Cont *replay row C, provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}$ *row N. The package exports no quotient completion, selected limit, host equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket public obligation package`

Proof. Start with the NameCert obligations of Definition 1585.2. They expose the carrier rows $S, M, U, Q, I, E, R, T, H, C, P, N$ as the entire local certificate surface.

Apply the unit-counit route of Definition 1585.3. It forces each public source read through S, U, M, Q before the downstream reflector rows can be consumed.

Next apply the idempotent-extension obligation of Definition 1585.5. The idempotent row I, extension row E, and reflection row R are read over the same unit-counit surface rather than over a second completion carrier.

Finally use the extension-reflection scope of Definition 1585.6. The structural rows T, H, C, P, N transport, replay, source, and name only the displayed route, so no quotient completion, selected limit, host equality, countable-choice selector, or ambient-completeness row can be exported. □

Theorem 1585.8 (CauchyCompletionReflector real-transport nonescape)**.** *For every accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier* $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$*, the real-facing transport row* T *and the structural rows* H, C, P, N *preserve exactly the displayed reflector route. A consumer may transport and replay the source, completion object, unit, counit, idempotent, extension, and reflection rows through* T, H, C, P, N*, but it cannot export quotient completions, selected limits, host equality, countable choice, ambient completeness, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or any real route not displayed by* T*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket real transport nonescape`

Proof. Read the local transport row from the carrier. By definition, T is the only row that exposes the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ windows used by U, Q, I, E, R.

Apply the unit-counit and idempotent-extension obligation theorems. They place the completion-object route, the return route, repeated completion, extension, and reflection all over the same displayed S, U, M, Q, T surface.

The rows H, C, P, N do not enlarge that surface. They provide componentwise hsame transport, displayed Cont replay, package provenance, and local naming for the rows already present in the packet.

Hence every real-facing read remains inside the displayed transport route. The excluded quotient, selection, host-equality, choice, ambient-completeness, detached-real-completeness, and off-route real data have no coordinate from which they could be exported. □

Theorem 1585.9 (CauchyCompletionReflector naturality square)**.** *Let* $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. The public reflector square classifies the route*

$$S \longmapsto U \longmapsto M \longmapsto E, R \longmapsto H, C, P, N$$

with the route

$$S \longmapsto T \longmapsto Q, I \longmapsto E, R \longmapsto H, C, P, N.$$

Equivalently, a consumer that first applies unit admission and then reads the extension and reflection rows has the same displayed classifier as a consumer that first reads the source through the local $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *transport row* T *and then returns by the counit and idempotent rows. The comparison is made only through the displayed* hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}$ *rows, and it exports no quotient completion equality, selected limit, host equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket naturality square`

Proof. Apply Definition 1585.2 to expose the only rows available in an accepted carrier. The unit row U, counit row Q, idempotent row I, extension row E, reflection row R, transport row T, and structural rows H, C, P, N are the entire public surface.

Apply Definition 1585.3 to place every public consumer on the same finite route. The left leg of the square reads S through U before it consumes M, E, R; the right leg reads S through T before the return and replay rows Q, I feed the same E, R consumers.

The sibling certificates supply the public faces used by these two readings: the unit surface is chapter 1527, the counit surface is chapter 1539, the idempotent replay surface is **??**, the extension surface is chapter 397, and the reflection boundary is chapter 403.

Since both legs end by the same H, C, P, N rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row classify them as the same public reflector square. No extra classifier coordinate remains from which a quotient completion, selected limit, host equality, countable choice, or ambient completeness claim could be read. □

Theorem 1585.10 (CauchyCompletionReflector idempotent nonescape)**.** *Let $K = (S, M, U, Q, I, E, R, T, H, C, P,$ be an accepted* CauchyCompletionReflector$^{\uparrow}$ *carrier. Any repeated completion consumer must read the idempotent row I over the same S, U, M, Q, T route before it can inspect E, R, H, C, P, N. Consequently the repeated completion route cannot export a second completion carrier, quotient completion equality, selected limit, host equality, countable choice, or detached* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket idempotent nonescape`

Proof. Project the carrier of Definition 1585.1. The only completion-facing rows are the displayed source S, completion object M, unit U, counit Q, idempotent row I, extension row E, reflection row R, local transport row T, and structural rows H, C, P, N.

Use the unit-counit obligation of Definition 1585.4. It fixes every public completion read at the route through S, U, M, Q, T before extension or reflection data are exposed.

Use the idempotent-extension obligation of Definition 1585.5. A repeated completion consumer must read I over the same route, so it reuses the displayed completion-object row rather than producing an independent carrier.

Finally apply the real-transport nonescape theorem Definition 1585.8. The rows H, C, P, N preserve the same route through T and add no quotient, selected-limit, host-equality, choice, or detached real-completeness coordinate. □

Theorem 1585.11 (CauchyCompletionReflector unit-counit scope)**.** *For every accepted* CauchyCompletionReflecto *carrier $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$, the unit-counit scope is exactly the finite row family S, U, M, Q, T. A reflector consumer must expose the source row S, unit row U, completion-object row M, counit row Q, and local transport row T before any idempotent, extension, reflection, transport-replay, provenance, or local naming row is consumed.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket unit counit scope`

Proof. Project the carrier of Definition 1585.1. The rows S, U, M, Q, T are the only source, unit, completion-object, counit, and local real-transport coordinates.

Apply the unit-counit route theorem, Definition 1585.3, and the unit-counit obligation theorem, Definition 1585.4. Together they force every public reflector read through $S \mapsto U \mapsto M$ and every return read through Q and the shared transport row T.

The remaining rows I, E, R, H, C, P, N are downstream of that scope. Since none of them is a second source, second completion object, selected limit, quotient carrier, host equality, countable-choice schedule, or ambient completeness row, the unit-counit scope is exactly S, U, M, Q, T. □

Theorem 1585.12 (CauchyCompletionReflector idempotent scope)**.** *For every accepted* CauchyCompletionReflecto *carrier $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$, the idempotent-extension scope is exactly I, E, R over the same S, U, M, Q, T route. Repeated completion reads I, universal-property consumers read E, and reflection consumers read R, all after the unit-counit scope has been exposed and without adding a second completion carrier.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket idempotent scope`

Proof. Start from unit-counit scope, Definition 1585.11. It fixes the shared route S, U, M, Q, T for every public reflector consumer.

Apply idempotent-extension obligation, Definition 1585.5. The repeated-completion row I, extension row E, and reflection row R are read over the same unit-counit route rather than over an independent completed carrier.

Then apply idempotent nonescape, Definition 1585.10. It excludes a second completion carrier, quotient completion equality, selected limits, host equality, countable choice, and detached real completeness from the repeated-completion route. □

Theorem 1585.13 (CauchyCompletionReflector obligation closure package)**.** *The accepted* CauchyCompletionRefle *carrier*

$$K = (S, M, U, Q, I, E, R, T, H, C, P, N)$$

supports an obligationClosure package whose finite boundary is the unit-counit scope S, U, M, Q, T, the idempotent-extension scope I, E, R over that route, real-transport nonescape through T, H, C, P, N, and the naturality square comparing the displayed unit and transport-return legs. The package refuses quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket obligation closure package`

Proof. Build the unit-counit boundary from Definition 1585.11. This supplies the exact S, U, M, Q, T row family consumed before any downstream reflector row is read.

Build the idempotent-extension boundary from Definition 1585.12. It places I, E, R over the same route and prevents a repeated completion pass from producing a second completion carrier.

Add real-transport nonescape, Definition 1585.8. The rows T, H, C, P, N preserve the displayed StreamName-RegSeqRat-DyadicRatCore-Real route and add no off-route real object.

Finally attach the naturality square, Definition 1585.9. Both legs end in the same displayed hsame, Cont, Pkg, and NameCert rows, so the package closes the paper-side reflector obligation surface while refusing quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached Real completeness. □

Theorem 1585.14 (CauchyCompletionReflector reflection scope)**.** *Let $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier. Every reflection consumer must read the reflection row R only after the displayed route*

$$S \longmapsto U \longmapsto M \longmapsto Q \longmapsto I \longmapsto E \longmapsto T$$

has been exposed. Thus R is a scoped reflector row over the already displayed source, unit, completion object, counit, idempotent replay, extension, and local real-name transport rows, not an independent host-level reflection or quotient-completion classifier.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket reflection scope`

Proof. Start from the carrier of Definition 1585.1. The only reflection coordinate in the packet is the displayed row R, and it sits beside the rows $S, M, U, Q, I, E, T, H, C, P, N$.

Use the unit-counit obligation of Definition 1585.4. It forces public completion-facing reads through S, U, M and already completed or real-sealed reads through Q, T before any downstream reflector row is consumed.

Use the idempotent-extension obligation of Definition 1585.5. The row I replays repeated completion over the same route, and the row E is the extension surface read before reflection consumption.

Finally apply real-transport nonescape and the naturality square, Definition 1585.8 and Definition 1585.9. They keep T, H, C, P, N on the same displayed route, so R cannot be read as an independent quotient, selected-limit, host-equality, choice, ambient-completeness, or detached-real-completeness assertion. □

Theorem 1585.15 (CauchyCompletionReflector obligation triple)**.** *For every accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *carrier $K = (S, M, U, Q, I, E, R, T, H, C, P, N)$, the unit row U, counit row Q, and idempotent replay row I are the three obligatory reflector faces before any universal-property or reflection consumer may read E or R. The triple is carried over the same S, M, T, H, C, P, N structural surface and exports no second completion carrier, quotient completion equality, selected limit, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket obligation triple`

Proof. Carrier admission comes from Definition 1585.1. The unit, counit, and idempotent rows are visible as U, Q, I in the same finite packet.

The unit-counit obligation theorem places U and Q before public completion and return consumption. The unit face admits S to M, and the counit face returns already completed or real-sealed reads through the local transport row T.

The idempotent-extension obligation theorem places I over that same route. A repeated completion pass therefore reuses the displayed completion-object row rather than introducing a second carrier.

The real-transport nonescape theorem and naturality square keep T, H, C, P, N attached to the same triple. Consequently the extension row E and reflection row R are read only after U, Q, I have been exposed, and the excluded quotient, selection, host-equality, choice, ambient-completeness, and detached-real-completeness data remain outside the packet. □

Theorem 1585.16 (CauchyCompletionReflector public reflector export)**.** *Every accepted* CauchyCompletionReflect *packet exports the public completion reflector interface exactly through the unit row U, completion object row M, counit row Q, idempotent row I, extension row E, reflection row R, and local* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *transport row T. The structural rows H, C, P, N preserve that route and do not export quotient completions, selected limits, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the unit-counit route, Definition 1585.3. It places every public reflector consumer on the path through S, U, M, Q before the downstream reflector rows are read.

Apply the idempotent-extension scope, Definition 1585.12, and the reflection scope, Definition 1585.14. These results make I, E, R the displayed repeated-completion, extension, and reflection faces over the same unit-counit route.

Add real-transport nonescape, Definition 1585.8. The row T is the only local StreamName-RegSeqRat-DyadicRatCore-Real route, and the rows H, C, P, N only transport, replay, source, and name the already displayed packet.

Finally use the obligation triple, Definition 1585.15. Since U, Q, I are obligatory before E or R is consumed, the public interface is exactly the row family $U, M, Q, I, E, R, T, H, C, P, N$, with no coordinate for the excluded quotient, selection, host-equality, choice, ambient-completeness, or detached-real-completeness data. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket public reflector export`

Theorem 1585.17 (CauchyCompletionReflector mature universal factorization)**.** *Every mature public consumer of an accepted* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *packet factors through the unit row U, counit row Q, idempotent replay row I, universal-property extension row E, reflection row R, and local* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{DyadicRatCore}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *transport row T before any reflector export is consumed. The same route preserves the real-seal nonescape boundary through H, C, P, N and exports no quotient completion, selected limit, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the public reflector export of Definition 1585.16. It identifies the public interface exactly at the unit, completion object, counit, idempotent, extension, reflection, and local real-transport rows.

Use the unit-counit route, idempotent scope, and reflection scope results. Together they force a public consumer through U, Q, I, E, R over the same displayed completion-object and local transport route before any reflector export can be read.

Finally use real-transport nonescape. The rows T, H, C, P, N preserve the StreamName-RegSeqRat-DyadicRatCore-Real route and add no excluded completion or choice coordinate, so the public universal factorization remains inside the finite reflector packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflector}^{\uparrow}$ packet is generated from a displayed source row, a completion-object row, unit admission, counit return, idempotent replay, universal-property extension, completion reflection, local StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1585.1, Definition 1585.16, and Definition 1585.17 close the public reflector export and universal factorization surface through unit, counit, idempotent replay, extension, reflection, and RealUp nonescape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflector}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.CauchyCompletionReflectorUp.CauchyCompletionReflectorPacket\_public\_reflector\_export`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality, countable choice, ambient completeness, detached $\mathsf{Real}^{\uparrow}$ completeness, or standard bridge checking.

Upgrade path: Bridge checking requires a standard bridge theorem connecting the public reflector export to the acceptance schema.

Chapter 1586

A Concrete Naming Certificate for $\mathsf{CauchyProductCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyProductCriterion}^{\uparrow}$ packet isolates the finite criterion under which synchronized Cauchy product data is sufficient for the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the product route of chapter 464, the bounded-product ledger of chapter 1529, and the completion-facing product route of chapter 1540; it reads synchronized windows from chapter 126, dyadic budget rows from chapter 291, the regular rational classifier of chapter 301, and the real-name boundary of chapter 38. The packet names a finite sufficiency surface, not multiplication of completed real objects, quotient equality of streams, a selected limit, countable choice, trichotomy, or an ambient completeness theorem.

Definition 1586.1 (Cauchy product criterion carrier)**.** *A* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, Y, W, D, P, R, E, H, C, L, N)$$

with the following displayed rows:

1. *X and Y are the two source rows. Each row exposes a Cauchy schedule, a regular-rational readback boundary, a real-seal boundary, and the local source certificate needed by the product route.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, aligning the two source schedules with the product window consumed downstream.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget ledger, containing the source tolerances, magnitude bounds, cross-error budget, and the finite window budget used to certify the product read.*
4. *P is the* $\mathsf{CauchyProduct}^{\uparrow}$ *route, read only after X, Y, W, and D have been displayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row receiving the product read from the synchronized product route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after the regular-rational classifier row has accepted the product read.*
7. *H records componentwise* hsame *transport for $X, Y, W, D, P, R, E, C, L, N$ and the local packet shape.*
8. *C records displayed* Cont *replay from the two source rows through the synchronized window, dyadic product budget, product route, regular-rational classifier, and real-seal row.*

9. *L is the* Pkg *provenance over* CauchyProductCriterion$^{\uparrow}$, CauchyProduct$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchyProductCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, P, R, E, H, C, L, N$. It has no coordinate for host multiplication of completed reals, quotient stream equality, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1586.2 (Cauchy product criterion NameCert obligations)**.** *For an accepted* CauchyProductCriterion$^{\uparrow}$ *carrier $Z = (X, Y, W, D, P, R, E, H, C, L, N)$, the local* NameCert$_{\text{CauchyProductCriterion}^{\uparrow}}$ *obligations are exactly carrier admission for the two source rows, synchronized-window admission for W, dyadic product-budget exactness for D, product-route admission for P, regular-rational handoff for R, real-seal handoff for E, componentwise transport through H, displayed continuation replay through C, provenance through L, and ledger non-escape through N.*

Proof. Carrier admission is projection from Definition 1586.1. The displayed packet lists the two source rows, the synchronized window, the dyadic budget, the product route, the regular-rational classifier, the real seal, and the structural rows.

The source and window obligations are read from X, Y, W. These rows display the two schedules and the common product window before any product observation is sent to a downstream classifier.

The dyadic obligation is the row D. It contains the finite tolerance, magnitude, and cross-error budget needed by the product route, so product admission through P is budgeted data rather than host multiplication of completed real names.

The rows R and E are downstream boundaries. Transport through H, replay through C, provenance through L, and local naming through N preserve the same route and add no extra coordinate.

Since the carrier has no row for quotient streams, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking, the displayed obligations exhaust the local certificate surface. □

Theorem 1586.3 (Cauchy product criterion window-budget sufficiency)**.** *For an accepted* CauchyProductCriterion$^{\uparrow}$ *carrier, synchronized windows and dyadic product-budget data suffice to route the product read to the regular-rational classifier:*

$$X, Y \longmapsto W \longmapsto D \longmapsto P \longmapsto R.$$

Every RegSeqRat$^{\uparrow}$ *consumer of the product packet factors through this finite route and cannot bypass it by reading a quotient stream, selected product limit, host multiplication primitive, countable-choice row, trichotomy row, or ambient completeness witness.*

Proof. Begin with a consumer asking for the regular-rational product read. The only RegSeqRat$^{\uparrow}$ row in the carrier is R.

The row R is downstream of the displayed product route P. The route P is downstream of the synchronized window W and dyadic budget D, and those rows are in turn downstream of the two source rows X and Y.

Thus the consumer must pass through the displayed chain X, Y, W, D, P, R. The structural rows H, C, L, N transport, replay, source, and name this same chain without creating a second product route.

No other regular-rational classifier occurs in the packet. Therefore the synchronized window and dyadic product budget are sufficient for the RegSeqRat$^{\uparrow}$ handoff, and the excluded objects cannot be exported by the criterion. □

Theorem 1586.4 (Cauchy product criterion source-window stability)**.** *Let $Z = (X, Y, W, D, P, R, E, H, C, L, N)$ be an accepted* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *carrier. Suppose the source rows X, Y and the synchronized window row W are replaced only by displayed componentwise* hsame *rows and the replacement is replayed through the displayed* Cont *row C. Then the same dyadic product-budget row D, product route P,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, and* $\mathsf{Real}^{\uparrow}$ *seal E remain the admissible criterion route:*

$$X, Y \longmapsto W \longmapsto D \longmapsto P \longmapsto R \longmapsto E.$$

Thus source-window replacement along the classifier data preserves the product criterion and cannot introduce host multiplication, quotient stream equality, a selected product limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier to its displayed rows. By Definition 1586.3, the original $\mathsf{RegSeqRat}^{\uparrow}$ product read factors through X, Y, W, D, P, R.

The replacement hypothesis is internal to the carrier data: H supplies componentwise hsame transport for the source rows and synchronized window, while C replays the continuation from the transported source-window data through the same displayed product route.

Therefore the transported source-window presentation still reaches D, then P, then R, and the real-facing row E reads the same admissible criterion route. No row in the carrier can insert host multiplication, a quotient-stream witness, a selected limit, countable choice, trichotomy, or ambient completeness between these displayed stages. □

Theorem 1586.5 (Cauchy product criterion real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *carrier factors through*

$$X, Y \longmapsto W \longmapsto D \longmapsto P \longmapsto R \longmapsto E$$

together with the displayed hsame *transports,* Cont *replays,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchyProductCriterion}^{\uparrow}}$ *row. Hence the real-facing boundary is a finite synchronized Cauchy product criterion, not host multiplication, quotient stream equality, a selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The real-facing consumer can read only E as a $\mathsf{Real}^{\uparrow}$ seal row. In the carrier, E is explicitly downstream of R, and R is the regular-rational handoff produced by the product route.

Apply Definition 1586.3. The route to R already passes through the two source rows, synchronized window, dyadic product-budget ledger, and product route.

The structural rows H, C, L, N preserve that route under transport, replay, provenance, and local naming. They do not add a second seal source or a host-level product operation.

Since there is no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient completeness, every real-seal read remains inside the displayed finite criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCriterion}^{\uparrow}$ packet is generated from two Cauchy source rows, a synchronized StreamName window, a DyadicRatCore product-budget ledger, a CauchyProduct route, a RegSeqRat classifier row, a RealUp seal row, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 1586.1, Definition 1586.2, Definition 1586.3, Definition 1586.4, and Definition 1586.5 seed the finite criterion for synchronized Cauchy product data before RegSeqRat and Real seal consumers read the product.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host multiplication, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic product-budget exactness, product-route admission, regular-rational handoff, real-seal handoff, transport stability, continuation replay, and ledger non-escape for BEDC.Derived.CauchyProductCriterionUp.

Chapter 1587

A Concrete Naming Certificate for $\mathsf{PositiveReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PositiveReal}^{\uparrow}$ packet names a positive real not by a host order predicate but by displayed apartness-from-zero evidence over the finite real-name substrate. It sits at the L10 boundary between the sealed $\mathsf{Real}^{\uparrow}$ name surface of chapter 38, the located interval route of **??**, the positive separation rows of chapter 304, and the located-apartness handoff of chapter 1380.

1587.1 Carrier and positivity packet

Definition 1587.1 (PositiveReal carrier)**.** *A* $\mathsf{PositiveReal}^{\uparrow}$ *carrier is a finite packet* $(R, A, D, W, Q, H, C, P, N)$ *with the following displayed rows:*

1. *R, a sealed* $\mathsf{Real}^{\uparrow}$ *row for the real-name presentation being classified;*
2. *A, a positive apartness-to-zero row, read as an* $\mathsf{ApartnessReal}^{\uparrow}$ *finite-window separation packet whose left endpoint is the zero real-name row and whose right endpoint is R;*
3. *D, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger for the separation radius used by A;*
4. *W, a finite* $\mathsf{StreamName}^{\uparrow}$ *observation window shared by the zero row and R;*
5. *Q, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the scheduled rational observations of R on W;*
6. *H, the componentwise* hsame *transport rows for the real seal, apartness row, radius ledger, window, and readback;*
7. *C, the displayed* Cont *replay routes keeping the finite observations attached to the transported carrier;*
8. *P, the* Pkg *provenance rows for the real seal, dyadic radius, stream window, regular-rational readback, and apartness ledger;*
9. *N, the local* $\mathsf{NameCert}$ *row binding these eight rows as one positive-real certificate.*

The carrier accepts only this finite positivity packet. It contains no trichotomy coordinate, no host real order predicate, no quotient stream equality, no selected limit, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealTasteGate single carrier alignment`

1587.2 NameCert obligations

Theorem 1587.2 (PositiveReal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PositiveReal}^{\uparrow}}$ *surface for the carrier of Definition 1587.1 consists of six finite obligations:*

1. *carrier admission: the displayed rows* $(R, A, D, W, Q, H, C, P, N)$ *are present in the same finite* BHist *packet;*
2. *classifier transport: accepted carriers compare only by componentwise transport of the real seal row, apartness row, positive radius ledger, finite window, readback, replay, provenance, and local certificate rows;*
3. *positive-radius exactness: the radius consumed by the apartness-to-zero row is exactly the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *displayed by the carrier;*
4. *apartness handoff: every positivity read factors through the* $\mathsf{ApartnessReal}^{\uparrow}$ *and* $\mathsf{LocatedApartness}^{\uparrow}$ *finite-window rows named in* A*;*
5. *real-seal non-escape: the real component is exported only through the sealed* $\mathsf{Real}^{\uparrow}$ *row* R *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*;*
6. *ledger exhaustion: downstream metric or order consumers can read only the nine displayed carrier rows and their componentwise transport routes.*

These obligations certify positivity as finite apartness-from-zero evidence. They do not certify a classical positive cone, a total order, trichotomy of constructive reals, quotient equality of streams, selected limits, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealNameCert obligations`

Proof. Project the carrier of Definition 1587.1. The projection exposes exactly the sealed real row, apartness-to-zero row, positive dyadic radius, finite observation window, regular-rational readback, transport, replay, provenance, and local certificate rows.

Classifier transport is then componentwise. The hsame and Cont rows move the real seal, apartness packet, radius ledger, stream window, readback, and provenance together, so the transported packet has the same visible carrier shape.

Positive-radius exactness follows by reading D as the only dyadic radius row available to the apartness coordinate A. The apartness handoff is therefore the finite-window separation route already displayed by the $\mathsf{ApartnessReal}^{\uparrow}$ and $\mathsf{LocatedApartness}^{\uparrow}$ packets.

The real-seal boundary is negative. Since R and Q are the only real-name coordinates, no consumer can leave the sealed $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ surface. The listed rows exhaust the packet, so no trichotomy, host order, quotient stream equality, selected limit, or ambient completeness datum can be exported. □

Theorem 1587.3 (PositiveReal apartness handoff)**.** *Every accepted* $\mathsf{PositiveReal}^{\uparrow}$ *read factors through the displayed* $\mathsf{ApartnessReal}^{\uparrow}$ *or* $\mathsf{LocatedApartness}^{\uparrow}$ *separation row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of Definition 1587.1. Consequently a downstream real-order, metric, or algebraic consumer receives positivity only as finite apartness-from-zero evidence with visible* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

No such read supplies trichotomy, host order, quotient stream equality, selected limits, or ambient completeness for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with an accepted carrier and project its apartness coordinate A. By Definition 1587.1, A is already an apartness-to-zero finite-window row whose positive radius is displayed by D and whose window is displayed by W.

Read the same packet through the surrounding L10 siblings. The real-name endpoint is sealed by R, the scheduled rational observations are exposed by Q, and the located-apartness route repacks the positive gap as the finite separation ledger consumed by A.

Transport preserves only these rows. The hsame, Cont, Pkg, and NameCert coordinates move the real seal, apartness packet, dyadic radius, finite window, and readback componentwise, so every transported read still factors through the displayed apartness and radius surfaces.

The carrier has no coordinate for an order decision, quotient representative, chosen limit, or completeness witness. Therefore every downstream consumer is confined to the finite apartness-from-zero evidence stated above. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier apartness handoff`

Theorem 1587.4 (PositiveReal RegSeqRat window positivity scope)**.** *Every accepted* $\mathsf{PositiveReal}^{\uparrow}$ *packet* $(R, A, D, W, Q, H, C, P, N)$ *exposes positivity reads only through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *attached to the sealed* $\mathsf{Real}^{\uparrow}$ *row* R*. The apartness coordinate* A *may consume the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius* D *only after those window and readback rows are displayed, and every transported read remains confined to the same* $R, A, D, W, Q, H, C, P, N$ *carrier surface.*

Proof. Project the carrier of Definition 1587.1. The real endpoint is R, the apartness-to-zero evidence is A, the positive radius is D, and the only observation route is the pair W, Q.

Use the obligation surface of Definition 1587.2. Its real-seal non-escape and apartness handoff rows require each positivity read to pass through the displayed window and readback rows before the apartness evidence is exported.

Apply Definition 1587.3. The handoff keeps A tied to the finite $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the dyadic radius D.

Transport is componentwise through H and replayed through C, with P, N recording provenance and local naming. Hence the positivity scope never leaves the displayed $R, A, D, W, Q, H, C, P, N$ packet. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier regseqrat window positivity scope`

Theorem 1587.5 (PositiveReal apartness radius transport)**.** *For every accepted* $\mathsf{PositiveReal}^{\uparrow}$ *carrier* $(R, A, D, W, Q, H, C, P, N)$*, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D *transported by* H *and replayed by* C *is the same radius read by the apartness-to-zero row* A*. A downstream consumer may move this radius only together with the sealed real row* R*, the stream window* W*, the regular readback* Q*, the provenance row* P*, and the local name* N*.*

Proof. Read D from the carrier definition. It is the unique positive dyadic radius ledger assigned to the apartness coordinate A, so a radius consumer has no second radius row to inspect.

Invoke Definition 1587.2. Positive-radius exactness identifies the radius consumed by A with the displayed D row, while classifier transport moves the real seal, apartness row, radius ledger, window, readback, replay, provenance, and local certificate componentwise.

The apartness handoff of Definition 1587.3 supplies the consumer boundary. It lets downstream order, metric, or algebraic reads use only the finite apartness-from-zero evidence carried by A, D, W, Q.

Therefore H and C transport the positive radius as part of the whole $R, A, D, W, Q, H, C, P, N$ packet. The transported radius is not a host real selector, a trichotomy branch, a quotient representative, or a completeness witness. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier apartness radius transport`

Theorem 1587.6 (PositiveReal RealAlgOrder consumer boundary)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *or metric consumer of an accepted* $\mathsf{PositiveReal}^{\uparrow}$ *carrier* $(R, A, D, W, Q, H, C, P, N)$ *receives only apartness-from-zero evidence: the sealed real row* R*, the apartness row* A*, the positive radius ledger* D*, the finite window* W*, the readback* Q*, and the support rows* H, C, P, N*. Such a consumer receives no trichotomy, host order predicate, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier and project its finite rows. The only order-facing data are the apartness row A and the positive dyadic radius D attached to the sealed real row R.

The NameCert obligations of Definition 1587.2 force real-seal non-escape and ledger exhaustion. Thus any consumer reads R through W, Q, reads positivity through A, D, and reads stability only through H, C, P, N.

Use Definition 1587.3 to identify the exported evidence. The handoff is apartness-from-zero evidence with visible transport, replay, provenance, and local naming.

Since no carrier coordinate stores trichotomy, a host order predicate, quotient stream equality, a selected limit, or real completeness, $\mathsf{RealAlgOrder}^{\uparrow}$ and metric consumers cannot receive any such principle from the packet. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier realalgorder consumer boundary`

Theorem 1587.7 (PositiveReal multiplicative radius transport)**.** *Let two accepted* $\mathsf{PositiveReal}^{\uparrow}$ *packets expose positive dyadic radius rows through the same finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observation window. Then the product-facing positivity read transports through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledgers, the componentwise* hsame *rows, the* Cont *replay routes, the* Pkg *provenance rows, and the local* NameCert *rows of the two packets. The transported read cannot use trichotomy, host order, quotient stream equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier multiplicative radius transport`

Proof. Project the two accepted carriers of Definition 1587.1. Each projection exposes a sealed real row, an apartness-to-zero row, a positive dyadic radius, a finite stream window, a regular-rational readback, transport, replay, provenance, and a local certificate row.

Read the common observation window componentwise. The apartness-to-zero rows use their displayed positive dyadic radii, and the shared $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ window keeps the product-facing read attached to the same finite schedule of rational observations.

Transport the two packets through the visible hsame and Cont rows. These rows preserve the radius ledgers, windows, readbacks, and provenance entries as finite data, so the product consumer receives only the displayed apartness-radius evidence.

The carrier definition supplies no trichotomy coordinate, host order predicate, quotient stream equality, selected limit, or completeness witness. Hence the product-facing positivity read cannot draw on any evidence outside the two finite PositiveReal packets. □

Theorem 1587.8 (PositiveReal product-radius window compatibility)**.** *Two accepted* $\mathsf{PositiveReal}^{\uparrow}$ *packets used by a product-facing* $\mathsf{RealAlgOrder}^{\uparrow}$ *or metric consumer may combine their positive*

DyadicRatCore$^{\uparrow}$ *radius rows only after both packets expose compatible* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readbacks. The product-radius read is then transported through the same* $R, A, D, W, Q, H, C, P, N$ *carrier rows for each packet, with the radius ledgers, windows, readbacks, replay routes, provenance, and local names kept componentwise.*

No product-facing consumer may infer radius compatibility from a host real order predicate, quotient stream equality, selected limits, or ambient Real$^{\uparrow}$ *completeness.*

Proof. First use Definition 1587.4. It forces each positivity read to pass through the displayed StreamName$^{\uparrow}$ window W and the RegSeqRat$^{\uparrow}$ readback row Q before the apartness radius is available to a downstream consumer.

Next apply Definition 1587.5. The positive dyadic radius row D is the same row consumed by the apartness-to-zero packet A, and it is moved only with R, W, Q, H, C, P, N.

Finally invoke Definition 1587.7. Its product-facing route combines two positive radius reads only through shared finite observation data and componentwise transport. Therefore a RealAlgOrder$^{\uparrow}$ or metric consumer may reuse the product radius exactly when the two packets display compatible stream windows and regular readbacks, and no stronger real-order or completeness datum is exported. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier product radius window compatibility`

Theorem 1587.9 (PositiveReal dyadic radius lower bound)**.** *Every accepted* PositiveReal$^{\uparrow}$ *packet whose positivity is read through the* Real$^{\uparrow}$ *apartness row exposes a finite* DyadicRatCore$^{\uparrow}$ *radius row below the positive witness. The row is transported through the displayed* RegSeqRat$^{\uparrow}$ *readback and* StreamName$^{\uparrow}$ *observation windows, and it is available to* MetricBall$^{\uparrow}$ *and* RealAlgOrder$^{\uparrow}$ *consumers as finite apartness-radius evidence without trichotomy, a host real selector, quotient stream equality, selected limits, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 1587.1. Its apartness coordinate A is read over the sealed real row R, and its dyadic radius ledger D is already displayed before any downstream order or metric consumer can inspect the packet.

Use the apartness handoff of Definition 1587.3. The handoff keeps the positive witness attached to the same finite StreamName$^{\uparrow}$ window and the same RegSeqRat$^{\uparrow}$ readback row, so D is read as a lower-radius row beneath that witness rather than as a host real choice.

Transport follows the multiplicative radius route of Definition 1587.7. The hsame, Cont, Pkg, and NameCert rows preserve the dyadic radius ledger, the finite observation window, and the regular-rational readback componentwise.

Thus a MetricBall$^{\uparrow}$ or RealAlgOrder$^{\uparrow}$ consumer receives exactly the displayed dyadic lower-radius row. Since the carrier contains no trichotomy, selector, quotient equality, selected limit, or completeness coordinate, no stronger real-order principle is exported. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier dyadic radius lower bound`

Theorem 1587.10 (PositiveReal product lower-bound apartness)**.** *Let two accepted* PositiveReal$^{\uparrow}$ *packets expose positive dyadic lower-bound rows through displayed* DyadicRatCore$^{\uparrow}$ *radius ledgers. A product-facing apartness-from-zero read can be formed only from the two radius rows, the common* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *observation windows, the componentwise* hsame *transports, the displayed* Cont *replays, the* Pkg *provenance rows, and the two local* NameCert *rows. The resulting lower-bound ledger is therefore a finite product-radius witness for* RealAlgOrder$^{\uparrow}$ *and metric consumers, not a trichotomy branch, host order comparison, quotient stream equality, selected limit, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the two accepted carriers of Definition 1587.1. Their positive reads expose exactly the two apartness rows, dyadic radius ledgers, finite observation windows, regular-rational readbacks, transport, replay, provenance, and local naming rows.

Apply Definition 1587.7. It supplies the product-facing radius transport while keeping the two dyadic ledgers inside the displayed finite packets. Then use Definition 1587.8 to require the common $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows before the product read is available.

Finally apply Definition 1587.9 to each factor. The only lower-bound data exported to the product consumer are the two finite dyadic radius rows and their transported window evidence. Since neither carrier contains trichotomy, host order, quotient stream equality, selected limits, or ambient completeness, the product apartness read cannot export any of those principles. □

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier product lower bound apartness`

Theorem 1587.11 (PositiveReal obligation closure)**.** *The* $\mathsf{PositiveReal}^{\uparrow}$ *packet of Definition 1587.1 presents an obligation-closed finite BEDC surface over the sealed* $\mathsf{Real}^{\uparrow}$ *row. Its local obligation rows consist exactly of the* $\mathsf{ApartnessReal}^{\uparrow}$ *and* $\mathsf{LocatedApartness}^{\uparrow}$ *apartness handoff, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *observation window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Consequently every positivity consumer must enter through these displayed rows and no consumer receives trichotomy, host order, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from this certificate.*

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier obligation closure`

Proof. Project the carrier rows of Definition 1587.1. The sealed real row R fixes the ambient real-name surface, while A, D, W, Q display the apartness packet, positive radius ledger, finite stream window, and rational readback.

Use Definition 1587.2. Its carrier admission, classifier transport, positive-radius exactness, apartness handoff, real-seal non-escape, and ledger exhaustion rows are precisely the finite obligations listed in the statement.

The remaining support rows are structural. The hsame row transports the displayed coordinates, Cont replays the finite reads, Pkg records provenance, and N binds the local name. Since the carrier has no further coordinate, the named rows close the paper-side obligation surface. □

Theorem 1587.12 (PositiveReal apartness obligation handoff)**.** *For every accepted* $\mathsf{PositiveReal}^{\uparrow}$ *carrier* $(R, A, D, W, Q, H, C, P, N)$*, the apartness-to-zero assertion is available to downstream consumers only as the finite handoff from the sealed* $\mathsf{Real}^{\uparrow}$ *row through the* $\mathsf{ApartnessReal}^{\uparrow}/\mathsf{LocatedApartness}^{\uparrow}$ *row* A*, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*. The handoff is stable under the displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier apartness obligation handoff`

Proof. Read the accepted packet componentwise. The real endpoint is sealed by R, and the positive assertion is not a host order predicate; it is the displayed apartness row A with radius D.

Apply Definition 1587.3 and Definition 1587.4. Together they force every positivity read through W, Q before it can be exported as apartness-from-zero evidence.

Transport is confined to the packet support. The rows H, C, P, N move and record the same finite handoff, so no consumer can detach A or D from the sealed real row, the observation window, or the readback surface. □

Theorem 1587.13 (PositiveReal carrier transport obligations)**.** *Any transport of an accepted* $\mathsf{PositiveReal}^{\uparrow}$ *carrier preserves the whole finite obligation packet* $(R, A, D, W, Q, H, C, P, N)$*: the sealed* $\mathsf{Real}^{\uparrow}$ *row, the* $\mathsf{ApartnessReal}^{\uparrow}/\mathsf{LocatedApartness}^{\uparrow}$ *apartness row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* hsame *transport rows, the* Cont *replay rows,* Pkg *provenance, and the local* NameCert *row. The transported carrier therefore exposes exactly the same obligation surface as the source carrier.*

Lean4 ✓ `BEDC.Derived.PositiveRealUp.PositiveRealCarrier transport obligations`

Proof. Start with the carrier definition and isolate the transport row H. Its components are attached to the real seal, apartness row, radius ledger, window, and readback, so it has no independent target outside the displayed packet.

Use Definition 1587.5 for the radius coordinate and Definition 1587.8 for window compatibility. These results keep D tied to A, W, Q under any consumer-facing movement.

Replay and provenance preserve the same boundary. The Cont rows replay the finite observations, Pkg records their source, and the local NameCert row binds the transported carrier. Thus the target carrier has exactly the obligation rows stated above and no stronger real-order or completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PositiveReal}^{\uparrow}$ packet is generated from a sealed real-name row, an apartness-to-zero row, one positive dyadic radius ledger, one finite observation window, one regular-rational readback, transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PositiveReal}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1587.1, Definition 1587.2, Definition 1587.11, Definition 1587.12, and Definition 1587.13 close the exported positive-real apartness packet as the public finite apartness-from-zero interface through $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PositiveReal}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not claim trichotomy of constructive reals, total order, quotient stream equality, selected limits, ambient real completeness, or a standard bridge.

Upgrade path: Remaining work is Lean verification and bridge checking over the existing positive-real classifier, apartness handoff, transport rows, and terminal real-facing seal, not further paper closure.

Chapter 1588

A Concrete Naming Certificate for CompleteUniformSpace$^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges constructive completion, the CompleteUniformSpace$^{\uparrow}$ packet names the finite Cauchy-filter convergence surface over chapter 339, chapter 333, chapter 1501, chapter 1558, chapter 126, chapter 301, and chapter 38. The packet is a constructive completeness interface for displayed Cauchy-filter data; it is not a quotient filter, not a selected limit, not a countable-choice selector, not host equality, and not an ambient completeness principle.

Definition 1588.1 (Complete uniform-space carrier)**.** *A* CompleteUniformSpace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (U, F, B, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformSpace$^{\uparrow}$ *entourage row. It stores the finite entourage filterbase, diagonal, refinement, symmetry, and composition data through which every Cauchy request is read.*
2. *F is the* CauchyFilter$^{\uparrow}$ *row. It records the saturated finite Cauchy-filter request admitted over the entourage surface of U.*
3. *B is the* CauchyFilterBasis$^{\uparrow}$ *generator row. It displays the finite basis elements and common-refinement data consumed by F.*
4. *L is the* CauchyFilterLimit$^{\uparrow}$ *boundary row. It records the limit-facing boundary reached only after the basis and saturated filter rows have been replayed.*
5. *S is the* StreamName$^{\uparrow}$ *window row inspected by the convergence request.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed windows in S.*
7. *E is the* Real$^{\uparrow}$ *seal row reached only after U, F, B, L, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the entourage, filter, basis, limit boundary, window, readback, seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an entourage request through the Cauchy-filter basis, saturation, limit boundary, finite windows, regular readback, and Real seal.*
10. *P is the* Pkg *provenance over* CompleteUniformSpace$^{\uparrow}$, UniformSpace$^{\uparrow}$, CauchyFilter$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, B, L, S, R, E, H, C, P, N$. The carrier has no coordinate for quotient filters, selected limits, countable-choice selectors, host equality, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CompleteUniformSpaceUp.CompleteUniformSpaceTasteGate single carrier alignment`

Theorem 1588.2 (Complete uniform-space NameCert obligations)**.** *For every accepted* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *carrier $M = (U, F, B, L, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteUniformSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed uniform-space entourage data, CauchyFilter row, CauchyFilterBasis generator, CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, Real seal, transport, replay, provenance, and local naming row;*
2. *Cauchy request factorization, requiring every filter request to pass through the finite entourage filterbase in U, the basis generator B, and the saturation row F before the limit boundary L is read;*
3. *convergence readback, requiring every limit-facing observation to pass through S and R before the Real seal row E is available;*
4. *transport and replay stability, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the same ordered route U, B, F, L, S, R, E;*
5. *ledger non-escape, requiring P and N to export no quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness principle.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1588.1. The projection exposes exactly the entourage row, the Cauchy-filter saturation row, the basis generator, the limit boundary, the window and readback rows, the Real seal, and the structural rows H, C, P, N.

Cauchy request factorization is forced by the displayed row order. A request begins at the finite entourage filterbase in U, reads a generated basis element through B, and is saturated by F before any limit boundary row can be inspected.

The convergence readback is downstream of this filter route. The boundary row L only exposes its finite observations through the StreamName windows S and the RegSeqRat readbacks R, and the Real seal row E is available only after those rows have been replayed.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, and Cont replay in C repeats the route U, B, F, L, S, R, E for every accepted request.

Finally, provenance is exhausted by P and N. Since quotient filters, selected limits, countable-choice selectors, host equality, and ambient completeness are not rows of M, the local NameCert surface cannot export them. □

Theorem 1588.3 (Complete uniform-space Cauchy-filter convergence)**.** *Let $M = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *carrier. Every Cauchy-filter convergence read factors through the finite route*

$$U \longmapsto B \longmapsto F \longmapsto L \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus every Cauchy-filter request passes through the displayed entourage filterbase and the limit boundary before any Real-facing seal is read. The route contains no quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness principle.

Proof. Begin with a convergence read of the packet. The uniform-space row U supplies only finite entourage data, so the request first chooses a displayed entourage demand and its refinement ledger rather than an external filter object.

The basis and saturation rows consume that demand. The row B displays the finite Cauchy-filter basis route, and F records the saturated Cauchy-filter request over the same entourage surface.

Only then can the limit boundary be read. The row L is downstream of B and F, while S and R provide the finite window and regular rational observations attached to that boundary.

The Real seal row E is terminal for this route. The structural rows H, C, P, N transport, replay, source, and name exactly this path, so no quotient filter, chosen limit, choice schedule, host equality, or detached completeness witness is available. □

Theorem 1588.4 (Complete uniform-space filter-basis limit exactness)**.** *Let $M = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *carrier. A completion-facing read from M is exposed only after the* $\mathsf{UniformSpace}^{\uparrow}$ *row U,* $\mathsf{CauchyFilter}^{\uparrow}$ *row F,* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row B,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary L,* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N are all present in the same packet. The read is exact in the sense that it factors through those displayed rows and cannot recover a quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness principle.*

Proof. Project the accepted carrier of Definition 1588.1. Its only completion-facing data rows are U, F, B, L, S, R, E, and its only structural support rows are H, C, P, N.

Apply the convergence route of Definition 1588.3. It places the basis row B between the entourage row U and the saturated Cauchy-filter row F, and it permits the limit boundary L to be read only after that finite basis route is displayed.

Now apply ledger non-escape, Definition 1588.6. The window row S, readback row R, Real seal E, and support rows H, C, P, N preserve the same route, so the packet has no remaining coordinate from which quotient filters, selected limits, countable-choice selectors, host equality, or ambient completeness could be exported. □

Theorem 1588.5 (Complete uniform-space Cauchy sequence, entourage, and Real-uniform factorization)**.** *Let $M = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *packet. Suppose downstream consumers read M through the* $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{CauchyEntourageFilter}^{\uparrow}$, *and* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *interfaces. Then their common completion-facing route factors as follows:* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *supplies the scheduled regular-family and finite-window surface,* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *supplies the entourage-to-basis dyadic handoff,* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *supplies the basis, saturation, and limit boundary, and* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *supplies the Real-uniform completion handoff to the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Equivalently, every such consumer read is confined to the displayed path

$$\mathsf{CauchySequenceSpace}^{\uparrow} \longmapsto \mathsf{CauchyEntourageFilter}^{\uparrow} \longmapsto \mathsf{CompleteUniformSpace}^{\uparrow} \longmapsto \mathsf{BishopRealUniformCompletion}^{\uparrow}$$

No quotient filter, selected limit, countable-choice selector, host equality, ambient Real completeness theorem, or bridge credential enters outside those rows.

Proof. First read the Cauchy-sequence surface. By Definition 406.3, the completion-facing consumer of the sequence packet passes through the displayed regular family, schedule, finite Cauchy windows, dyadic tolerance row, transports, continuations, provenance, and local certificate before any terminal handoff is consumed.

Next read the entourage-filter handoff. Apply Definition 1791.3. It identifies the route from UniformSpace entourage data to the CauchyFilterBasis dyadic surface, so the sequence-side tolerance and window data can meet the uniform-space request only through that finite basis handoff.

Now project the accepted $\mathsf{CompleteUniformSpace}^{\uparrow}$ carrier M. The convergence route of Definition 1588.3 consumes $U, B, F, L, S, R, E, H, C, P, N$ in order, placing the basis, saturation, and limit boundary between the entourage handoff and the Real-facing seal.

Finally apply Definition 1806.3. The Real-uniform completion consumer reads the terminal handoff only after the L10 completion data reaches its displayed $\mathsf{Real}^{\uparrow}$ seal. Since none of the cited packets contains a coordinate for a quotient filter, selected limit, countable-choice selector, host equality, ambient Real completeness theorem, or bridge credential, carrier projection and the non-escape clauses force every common consumer to remain on the stated path. □

Theorem 1588.6 (Complete uniform-space ledger non-escape)**.** *For every accepted* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *carrier* $M = (U, F, B, L, S, R, E, H, C, P, N)$*, any Real-facing or completion-facing read exposed by* P *factors through the displayed route*

$$U, B, F, L, S, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet cannot export a quotient filter from F*, a selected limit from* L*, a countable-choice selector from* S *or* R*, host equality from* H*, or an ambient completeness principle from its provenance ledger.*

Proof. Project the provenance row P. Its Real-facing and completion-facing occurrences are recorded only as downstream consumers of the displayed route through U, B, F, L, S, R, E.

The Cauchy-filter row F is not a quotient coordinate. It is read only after the basis generator B and the entourage data U, and it hands the request to the boundary row L rather than to a completed filter object.

The boundary row L is not a selected limit. It exposes only the finite limit-facing route whose observations pass through the StreamName windows S and RegSeqRat readbacks R.

Transport by H and replay by C preserve this same dependency order. The local naming row N names the packet only after that order has been replayed, so the ledger has no separate place from which quotient filters, selected limits, countable-choice selectors, host equality, or ambient completeness can escape. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteUniformSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteUniformSpace}^{\uparrow}$ packet is generated from UniformSpace entourage data, a CauchyFilter row, a CauchyFilterBasis generator, a CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompleteUniformSpace}^{\uparrow}}$, seed)

Scope closed: Definition 1588.1, Definition 1588.2, Definition 1588.3, and Definition 1588.6 seed the constructive uniform-space completeness surface for finite Cauchy-filter convergence data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompleteUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable-choice selectors, host equality, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy request factorization, convergence readback, transport stability, replay stability, Real seal factorization, and ledger non-escape for BEDC.Derived.CompleteUniformSpaceUp.

Chapter 1589

A Concrete Naming Certificate for $\mathsf{FastCauchyReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FastCauchyReal}^{\uparrow}$ packet records the fast dyadic-modulus real-name carrier used by the L10 Real and Completion phase. It sits over the dyadic tolerance arithmetic of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback row of chapter 301, the terminal real seal of chapter 38, the completion boundary of chapter 360, and the Cauchy-real presentation route of chapter 1511. The packet makes the fast tolerance modulus visible as finite BEDC data before any completion-facing consumer treats the resulting name as a real.

Definition 1589.1 (FastCauchyReal carrier)**.** *A* $\mathsf{FastCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *fast-tolerance row recording the dyadic scale 2^{-k}, comparison budget, rounding window, and arithmetic ledger for each finite precision request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose visible entries are exactly the stream observations requested by the fast tolerance row.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S under the dyadic tolerance data in D.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed route through D, S, R has been admitted.*
5. *H records componentwise* hsame *transport for the dyadic fast-tolerance, stream-window, regular-readback, real-seal, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from every finite precision request through D, S, R, E and then through the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected representative, host real equality, countable choice, located completeness, or an ambient completion object outside the displayed finite rows.

Theorem 1589.2 (FastCauchyReal NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyReal}^{\uparrow}$ *carrier* $F = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyReal}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed dyadic fast-tolerance, finite stream-window, regular rational readback, real-seal, transport, continuation, provenance, and local naming rows;*
2. *fast tolerance exactness, requiring each finite precision request to be accounted for by* D *before any stream window or readback row is consumed;*
3. *window compatibility, requiring* S *to expose only the finite observations named by the dyadic tolerance row;*
4. *regular readback, requiring* R *to be read only downstream of the displayed* D, S *route;*
5. *real-seal nonescape, requiring* E *to remain terminal with respect to the route* D, S, R *and the structural rows;*
6. *componentwise stability, requiring* hsame *transport* H *and* Cont *replay* C *to preserve the same displayed precision route;*
7. *provenance and local naming, requiring* P *and* N *to bind the accepted packet as a* $\mathsf{FastCauchyReal}^{\uparrow}$ *certificate.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1589.1. The tuple displays exactly the dyadic tolerance row, stream window, regular readback, real seal, transport, replay, provenance, and local name.

The analytic obligations are forced by the route D, S, R, E. A precision request first reads the fast dyadic ledger D, then the finite stream window S, then the regular rational readback R, and finally the terminal real seal E.

The structural obligations are forced by H, C, P, N. Componentwise transport and continuation replay preserve the same finite route, while provenance and local naming bind that accepted packet. Since the carrier has no coordinate beyond these rows, the listed clauses exhaust the local NameCert surface. □

Theorem 1589.3 (FastCauchyReal regular handoff)**.** *An accepted* $\mathsf{FastCauchyReal}^{\uparrow}$ *carrier supplies a regular rational Cauchy name to the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *routes only through the finite chain*

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

The handoff exposes dyadic fast tolerance, finite stream observations, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It does not add quotient stream equality, a selected representative, host real equality, countable choice, located completeness, or an ambient completion object outside the displayed carrier.*

Proof. Project the accepted packet and read the precision route in order. The dyadic row D supplies the fast tolerance budget, the stream row S supplies the finite observations named by that budget, the row R supplies regular rational readback, and the seal row E is terminal.

Now pass to the completion-facing consumers. The $\mathsf{RealCauchyCompletion}^{\uparrow}$ and $\mathsf{CauchyRealPresentation}^{\uparrow}$ routes may transport, replay, package, and locally name the handoff through H, C, P, N, but they receive only the displayed D, S, R, E data from this carrier.

The excluded principles are absent by inspection of the tuple. Quotient stream equality, selected representatives, host real equality, countable choice, located completeness, and ambient completion objects are not coordinates of F, so the regular handoff cannot export them. □

Theorem 1589.4 (FastCauchyReal modulus extraction route)**.** *For an accepted* $\mathsf{FastCauchyReal}^{\uparrow}$ *carrier* $F = (D, S, R, E, H, C, P, N)$*, the fast dyadic row* D *and stream window row* S *expose a convergence modulus only through* $\mathsf{FastCauchyModulus}^{\uparrow}$ *or chapter 3718 before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *is read. The carrier exports no detached modulus selector, quotient stream equality, or ambient completion object outside the displayed route* D, S, R, E, H, C, P, N*.*

Proof. Project the carrier of Definition 1589.1. The rows D and S are the only source rows that can answer a finite precision request before regular readback.

Apply the NameCert obligations of Definition 1589.2. Fast tolerance exactness keeps the modulus information in D, and window compatibility keeps the observations in S tied to that same budget.

The regular handoff theorem Definition 1589.3 places R downstream of D, S. Thus any modulus consumed before R must be the displayed fast-modulus surface or the regular Cauchy extraction route, not a separate selector. □

Theorem 1589.5 (FastCauchyReal modulus readback determinacy)**.** *Let* $F = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchyReal}^{\uparrow}$ *carrier. Any two accepted reads of the same fast tolerance row* D *through the same* $\mathsf{StreamName}^{\uparrow}$ *window row* S *and* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *reach the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary* E *under componentwise* hsame *transport and displayed* Cont *replay.*

The determinacy route is exactly

$$D, S, R \longmapsto E \longmapsto H, C, P, N.$$

It does not use selected limits, quotient streams, countable choice, host real equality, or ambient Real completeness.

Proof. Project the carrier of Definition 1589.1. The rows D, S, R, E are stored in one finite BHist packet, and E is terminal only for the displayed route from fast tolerance to stream window to regular readback.

Use Definition 1589.4. Both accepted reads expose their modulus data through the same D and S before the row R is consumed, so neither read can introduce a detached modulus selector or a second stream representative.

Finally apply Definition 1589.3. The regular handoff places E downstream of the same D, S, R route, and the rows H, C, P, N transport, replay, source, and name that route componentwise. Since the carrier has no selected-limit, quotient-stream, choice, host-equality, or ambient-completeness coordinate, the two reads have the same terminal Real seal boundary. □

Theorem 1589.6 (FastCauchyReal completion consumer route)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{CauchyRealPresentation}^{\uparrow}$ *consumer of an accepted* $\mathsf{FastCauchyReal}^{\uparrow}$ *packet factors through the displayed rows* D, S, R, E, H, C, P, N*. The completion-facing read first uses the fast dyadic tolerance data, then the finite stream windows, then the regular rational handoff, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and finally the structural transport, replay, provenance, and naming rows. It receives no selected host limit, quotient Cauchy class, countable-choice schedule, or ambient completeness witness from the packet.*

Proof. Start from Definition 1589.3. It already forces completion-facing consumers to receive the regular rational Cauchy name through D, S, R, E and then through the structural rows.

Use Definition 1589.4 for the pre-handoff part of the read. The modulus data is exposed only through the fast dyadic row and stream window row before R is consumed.

Finally project the excluded-coordinate clause of Definition 1589.1. Since selected host limits, quotient Cauchy classes, choice schedules, and ambient completeness witnesses are not coordinates of F, no Real or Completion consumer can obtain them from this route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyReal}^{\uparrow}$ packet is generated from a $\mathsf{DyadicRatCore}^{\uparrow}$ fast-tolerance row, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{RegSeqRat}^{\uparrow}$ regular readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1589.1, Definition 1589.2, and Definition 1589.3 seed the finite fast dyadic-modulus real-name carrier consumed by the L10 Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchyReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, located completeness, ambient completion objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fast tolerance exactness, window compatibility, regular readback, real-seal nonescape, componentwise stability, provenance, and local naming for the same displayed packet.

Chapter 1590

A Concrete Naming Certificate for $\mathsf{RealCauchyCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealCauchyCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed sequence of Real names is admitted as Cauchy data before any completion-facing consumer reads a terminal Real seal. It sits between the finite-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Real seal boundary of chapter 38, and the exact completion boundary of chapter 1120. The packet is a Real-sequence Cauchy readiness certificate, not a quotient stream, selected limit, host real equality, trichotomy, countable-choice source, or ambient Real completeness principle.

Definition 1590.1 (Real Cauchy criterion carrier)**.** *A* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, Q, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Real}^{\uparrow}$ *sequence-source row, exposing the finite list of Real-name entries and precision requests inspected by the criterion.*
2. *W is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows selected for the inspected source entries and late-pair requests.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only through the windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-test ledger recording the rational radii, late-pair comparisons, and dyadic triangle-budget cells used by the Cauchy test.*
5. *L is the Cauchy-test ledger tying each accepted precision request to a displayed late-pair window, regular readback, and dyadic tolerance comparison.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, Q, D, L have been displayed.*
7. *H records componentwise* hsame *transport for the sequence source, windows, regular readback, dyadic tolerance ledger, Cauchy-test ledger, Real seal, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a precision request through the sequence source, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance test, Cauchy-test ledger, and Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBoundaryUp, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, Q, D, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, detached Real completeness, or an ambient completion theorem.

Theorem 1590.2 (Real Cauchy criterion NameCert obligations)**.** *For an accepted* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *carrier $K = (S, W, Q, D, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission for the Real sequence source, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Cauchy-test ledger, Real seal, transport, replay, provenance, and local naming row;*
2. *window-readback compatibility, requiring every regular rational read in Q to be obtained from a displayed window in W attached to the source row S;*
3. *dyadic Cauchy-test exactness, requiring D and L to record the requested tolerance, late-pair budget, and triangle cells before the criterion may expose the seal row E;*
4. *transport and replay stability, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the same ordered route S, W, Q, D, L, E;*
5. *provenance exhaustion, requiring P and N to name only this displayed route and to export no quotient streams, selected limits, countable-choice selector, host real equality, trichotomy, detached Real completeness, or ambient completion principle.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1590.1. The packet displays the Real sequence source, selected StreamName windows, regular rational readback, dyadic tolerance ledger, Cauchy-test ledger, terminal seal row, and the structural rows H, C, P, N.

Window-readback compatibility is read from the order of W and Q. The readback row Q appears only after the selected finite windows W, and those windows are attached to the source row S rather than to an unbounded host stream.

Dyadic exactness is the role of D and L. The tolerance row D stores the rational radii and comparison cells, while the ledger L records which late-pair request has been tested before the Real seal E is available.

Transport and replay are structural. Componentwise hsame in H preserves the displayed rows, and Cont replay in C repeats the route from the precision request through S, W, Q, D, L, E.

Finally, provenance is exhausted by P and N. Since the carrier has no row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, detached Real completeness, or ambient completion, the local NameCert surface cannot export any such object. □

Theorem 1590.3 (Real Cauchy criterion regular readback route)**.** *Every accepted* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *read factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the Real sequence source, Cauchy-test ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}$ *row, and the* RealCompletionExactBoundaryUp

sibling boundary. The route cannot expose quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.

Proof. Begin with an accepted read of the criterion packet. Carrier projection gives the sequence source S and the selected finite windows W before any regular rational readback is available.

The readback row Q is downstream of W. Thus each accepted read first passes through the $\mathsf{StreamName}^{\uparrow}$ window discipline and then through the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational surface attached to that same displayed window.

The dyadic Cauchy test is downstream of the readback. The rows D and L record the requested tolerance, late-pair budget, and comparison cells before the terminal $\mathsf{Real}^{\uparrow}$ seal E may be consumed.

The exact-boundary sibling is read only as provenance discipline. The row P names RealCompletionExactBounda to keep the Real-completion faces aligned, but it contributes no selected limit, quotient equality, or independent completeness witness.

The exclusions follow from the carrier inventory. Since quotient streams, selected limits, countable choice, host real equality, trichotomy, and ambient Real completeness are not rows of K, no accepted read through H, C, P, N can expose them. □

Theorem 1590.4 (Real Cauchy criterion seal factorization)**.** *For every accepted* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *packet*

$$K = (S, W, Q, D, L, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}$ *seal row E is reachable only through the ordered finite route $S, W, Q, D, L, E, H, C, P, N$. Along this route W supplies the* $\mathsf{StreamName}^{\uparrow}$ *windows, Q supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and D, L supply the dyadic Cauchy-test ledger. No quotient stream, selected limit, countable choice, host equality, trichotomy, or ambient completeness row can bypass E.*

Proof. Project the carrier using Definition 1590.1. The projection displays the ordered rows $S, W, Q, D, L, E, H, C, P, N$, with the terminal Real seal placed after the finite windows, regular readback, dyadic tolerance data, and Cauchy-test ledger.

Definition 1590.2 identifies the five-row certificate surface governing carrier admission, window-readback compatibility, dyadic exactness, transport and replay stability, and provenance exhaustion. In particular, the seal row E is not an independent input; it is admitted only after the S, W, Q, D, L route has been recorded.

Definition 1590.3 supplies the StreamName–RegSeqRat–DyadicRatCore readback path. Combining that route with the carrier projection shows that every completion-facing use of E factors through the same finite Cauchy-test ledger.

The excluded objects are absent from the carrier inventory and from the NameCert surface. Therefore quotient stream equality, selected limits, countable choice, host equality, trichotomy, and ambient completeness cannot supply an alternate row reaching the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCauchyCriterion}^{\uparrow}$ packet is generated from a RealUp sequence source, StreamName finite windows, a RegSeqRat readback row, a DyadicRatCore tolerance-test ledger, a Cauchy-test ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1590.1, Definition 1590.2, Definition 1590.3, and Definition 1590.4 seed the Real-specific Cauchy criterion route through StreamName, RegSeqRat, DyadicRatCore, Real, and RealCompletionExactBoundary faces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, window-readback compatibility, dyadic Cauchy-test exactness, transport stability, replay stability, regular readback factorization, Real seal factorization, and provenance exhaustion for BEDC.Derived.RealCauchyCriterionUp.

Chapter 1591

A Concrete Naming Certificate for $\mathsf{ClosedBoundedInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ packet records a displayed closed bounded real interval as finite BEDC rows rather than as an ambient set of host reals. It consumes the located endpoint discipline of chapter 394, supplies the base carrier read by the finite net route of chapter 1250, and fixes the interval object consumed by the uniform-modulus chapter chapter 1253. The packet is a finite certificate for lower and upper located endpoints, endpoint order, rational and dyadic containment ledgers, StreamName windows, RegSeqRat readbacks, RealUp sealing, transport, replay, provenance, and local naming. It is not an open-cover compactness theorem, quotient equality of reals, trichotomy, a selected point of the interval, or an ambient completeness principle.

Definition 1591.1 (Closed bounded interval carrier)**.** *A* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located endpoint row, read through the located real and interval-endpoint surface of* $\mathsf{LocatedInterval}^{\uparrow}$*;*
2. *U is the upper located endpoint row, read through the same located endpoint surface;*
3. *O is the endpoint-order row saying that the lower endpoint is admitted no later than the upper endpoint in every displayed containment read;*
4. *Q is the rational containment ledger, recording finite rational tests that enter the closed interval through the endpoint rows and O;*
5. *D is the dyadic containment ledger, refining the rational tests by finite dyadic cells and dyadic endpoint comparisons;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the endpoint names and containment tests at finite precision;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint, rational-containment, and dyadic-containment windows;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, O, Q, D, S, R have been displayed;*

9. *H records componentwise* hsame *transports for the lower endpoint, upper endpoint, endpoint order, containment ledgers, windows, readbacks, and Real seal;*
10. *C records displayed* Cont *replay from located endpoints through containment tests, StreamName windows, RegSeqRat readbacks, and RealUp sealing;*
11. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row.*

The carrier exports no open cover, hidden selector for a point in the interval, quotient real equality, trichotomy row, or completeness datum. Its public object is the finite endpoint-and-containment packet I.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalTasteGate single carrier alignment`

Definition 1591.2 (Closed bounded interval classifier)**.** *Two accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers*

$$I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

and

$$I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$$

are classified together exactly when the displayed hsame *rows transport L to L', U to U', O to O', Q to Q', D to D', S to S', R to R', and E to E', and the displayed* Cont *rows replay the same endpoint-to-containment-to-seal route on both packets. The classifier has no coordinate for host-level set equality, order trichotomy, quotient equality of real names, or a choice of representatives for endpoint streams.*

Lean4 (definition) `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalClassifier`

Theorem 1591.3 (Closed bounded interval NameCert obligations)**.** *For the carrier of Definition 1591.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$; endpoint admission for L and U through* $\mathsf{LocatedInterval}^{\uparrow}$*; endpoint-order admission through O; rational and dyadic containment admission through Q and D; StreamName and RegSeqRat readback through S and R; RealUp sealing through E; classifier preservation under the componentwise* hsame *rows of H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket semantic name certificate`

Proof. Project the carrier. Its endpoint rows L and U, order row O, containment ledgers Q and D, window row S, readback row R, seal row E, transport row H, replay row C, provenance row P, and local name N are visible coordinates of one finite packet.

Endpoint and containment admission are read before the Real seal. The rational and dyadic ledgers are attached to the same lower and upper located endpoints and to the same endpoint-order row, so a containment consumer cannot replace the packet by a host interval or by an unrecorded order decision.

The structural rows close the naming surface. The componentwise hsame transports and displayed Cont routes preserve exactly the rows listed in the carrier, and the Pkg and NameCert rows name that finite route without adding any hidden selector, quotient equality, trichotomy, or completeness datum. □

Theorem 1591.4 (Closed bounded interval public export)**.** *The public* ClosedBoundedInterval$^{\uparrow}$ *export consists exactly of the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName window, RegSeqRat readback, RealUp seal, componentwise transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows of the accepted carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*. It exports no open-cover compactness, quotient real equality, trichotomy, selected point, selected subcover, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket public export`

Proof. Project the carrier and classifier of Definition 1591.1 and Definition 1591.2. They expose the endpoint rows L, U, endpoint order O, containment ledgers Q, D, StreamName window S, RegSeqRat readback R, Real seal E, transport row H, replay row C, provenance row P, and local naming row N.

Apply Definition 1591.3. The obligation theorem identifies those rows as the local NameCert surface, so the public export cannot add a host interval, quotient equality, order trichotomy, selected point, selected subcover, or ambient completeness coordinate.

Thus every L10 consumer receives exactly the displayed finite packet. Open-cover compactness and completeness-facing claims must enter through separate downstream certificates rather than through the ClosedBoundedInterval$^{\uparrow}$ public export itself. □

Theorem 1591.5 (Closed bounded interval root obligation spine)**.** *Every downstream* ClosedBoundedInterval$^{\uparrow}$ *consumer must read three root obligations through the existing carrier rows: endpoint-order exposure through* L, U, O*, containment-ledger exposure through* Q, D*, and* StreamName$^{\uparrow}$/RegSeqRat$^{\uparrow}$/Real$^{\uparrow}$ *seal handoff through* S, R, E*. These obligations are transported only by* H*, replayed only by* C*, recorded only by* P*, and named only by* N*.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root obligation spine`

Proof. Project the carrier and classifier. The carrier of Definition 1591.1 supplies the endpoint, containment, schedule, readback, seal, transport, replay, provenance, and local naming rows, while Definition 1591.2 preserves those rows componentwise.

Apply the NameCert obligations of Definition 1591.3. They expose endpoint order through O, containment through Q, D, and the StreamName, RegSeqRat, and Real seal handoff through S, R, E.

The source nonescape and finite-net source theorems then keep interval-net, compactness-facing, and uniform-modulus consumers on the same root spine. Thus the downstream route cannot replace the endpoint order, containment ledger, or seal handoff by a host interval, selected subcover, quotient equality, or ambient completeness row. □

Theorem 1591.6 (Closed bounded interval carrier gate)**.** *Every accepted consumer route for a* ClosedBoundedInterval$^{\uparrow}$ *packet must display the carrier rows*

$$L, U, O, Q, D, S, R, E, H, C, P, N$$

before it can read interval-net or compactness rows. In particular, an interval-net, nested-window, located-cover, compactness-facing, or uniform-modulus consumer cannot bypass lower endpoint, upper endpoint, endpoint order, rational containment, dyadic containment, StreamName window, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, or local naming.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket carrier gate`

Proof. Project the carrier of Definition 1591.1. The rows $L, U, O, Q, D, S, R, E, H, C, P, N$ are the whole finite source surface of the closed bounded interval packet.

Apply the NameCert obligation theorem. Theorem 1591.3 lists exactly those rows as the local $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ surface, so a consumer has no independent route into interval-net or compactness data.

Use the source-window and nonescape theorems. The source window of Definition 1591.34, source transport of Definition 1591.35, and source nonescape of Definition 1591.36 keep every downstream consumer behind the same displayed carrier gate. □

Theorem 1591.7 (Closed bounded interval dyadic-net ledger)**.** *Dyadic containment subledgers are the only finite-net source for* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *consumers. Any finite-net read exported from an accepted packet must pass through the dyadic containment row D, together with the endpoint rows L, U, endpoint-order row O, rational containment row Q, StreamName window S, RegSeqRat readback R, Real seal E, and structural rows H, C, P, N.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket dyadic net ledger`

Proof. Start with the carrier and classifier. The carrier of Definition 1591.1 contains exactly one dyadic containment coordinate D, and the classifier of Definition 1591.2 transports that coordinate componentwise with the endpoint, rational-containment, StreamName, RegSeqRat, and Real-seal rows.

Apply the source-window route. The source window of Definition 1591.34 reads dyadic containment through the same endpoint-source packet before any interval-net or compactness consumer opens its own rows.

Finally apply finite-net source and source nonescape. Theorem 1591.38 makes D part of the required finite-net source, and Definition 1591.36 excludes any alternate host interval, open-cover, or choice-selected net source. □

Theorem 1591.8 (Closed bounded interval endpoint transport)**.** *Classifier transport of a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier is determined by componentwise* hsame *transport of lower endpoint, upper endpoint, endpoint order, rational containment, dyadic containment, StreamName window, RegSeqRat readback, and Real seal. The transported interval-net and compactness-facing routes are then replayed only through the displayed* Cont*, provenance, and local naming rows of the same packet.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket endpoint transport`

Proof. Project two classified carriers through Definition 1591.2. The classifier compares L, U, O, Q, D, S, R, E only through the componentwise hsame rows stored in H.

Use source transport. Theorem 1591.35 states that endpoint source windows move by exactly those componentwise rows before any interval-net or compactness consumer reads them.

The replay step is fixed by the carrier. The displayed Cont row C, provenance row P, and local naming row N preserve the same transported endpoint route, and Definition 1591.36 prevents the transported consumer from importing host interval equality, quotient real equality, trichotomy, or ambient completeness. □

Definition 1591.9 (Closed bounded interval root source ledger)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$, the root source ledger is the finite* BHist *row*

$$\Lambda_I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

read before any interval-net, compactness-facing, located-cover, nested-window, or uniform-modulus consumer opens a downstream row. The ledger records lower endpoint, upper endpoint, endpoint order, rational containment, dyadic containment, stream schedule, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming as one source surface.

Lean4 (definition) `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalRootSourceLedger`

Theorem 1591.10 (Closed bounded interval endpoint-containment route)**.** *Every containment consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier must pass through the endpoint and containment prefix*

$$L, U, O, Q, D$$

of Definition 1591.9 before any finite-net, located-cover, nested-window, compactness, or uniform-modulus row is available. In particular, no downstream consumer can read a finite-net or compactness row while bypassing the lower endpoint, upper endpoint, endpoint-order row, rational containment ledger, or dyadic containment ledger of the root source packet.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket endpoint containment route`

Proof. Project the carrier and the root source ledger. By Definition 1591.1 and Definition 1591.9, the endpoint rows L, U, order row O, and containment ledgers Q, D are the first displayed source rows of the packet.

Apply the NameCert obligation theorem. The obligations of Definition 1591.3 admit endpoint rows, endpoint order, and rational and dyadic containment before StreamName, RegSeqRat, Real sealing, transport, replay, provenance, or local naming can be consumed.

Finally use the source-window route. The source window of Definition 1591.34 and source transport of Definition 1591.35 keep the same prefix upstream of every interval-net or compactness-facing read, so no downstream row can bypass L, U, O, Q, D. □

Theorem 1591.11 (Closed bounded interval Real-seal root handoff)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is downstream of the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, which themselves are downstream of the endpoint and containment prefix* L, U, O, Q, D*. The root handoff exports no selected point, quotient equality, trichotomy row, selected subcover, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalPacket real seal root handoff`

Proof. Start with the root source ledger. It records the route $L, U, O, Q, D, S, R, E, H, C, P, N$ as one finite BHist source surface, so the seal row is not available before the schedule and readback rows are displayed.

Use the existing source-window theorem. By Definition 1591.34, the $\mathsf{StreamName}^{\uparrow}$ row S, $\mathsf{RegSeqRat}^{\uparrow}$ row R, and $\mathsf{Real}^{\uparrow}$ seal E are read from the same endpoint-source packet after the endpoint and containment rows.

The non-escape conclusion follows from Definition 1591.36. Since the carrier and source route contain no selected point, quotient equality, trichotomy row, selected subcover, or ambient completeness coordinate, the Real-seal handoff cannot export any of them. □

Theorem 1591.12 (Closed bounded interval root source split)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier as in Definition 1591.1. The root source for every interval-net, compactness-facing, and uniform-modulus consumer is the concrete row split $L, U, O, Q, D, S, R, E, H, C, P, N$: endpoints and order are carried by L, U, O, containment by Q, D, readback by S, R, E, and structural persistence by H, C, P, N. The local NameCert obligation theorem Definition 1591.3 forces this split before any downstream row is read.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source split`

Proof. Project the carrier. Its only displayed root rows are the endpoint rows L, U, endpoint-order row O, containment ledgers Q, D, finite window S, regular rational readback R, Real seal E, transport H, replay C, provenance P, and local name N.

Apply Definition 1591.3. The theorem lists those same rows as the local $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ surface, so an interval-net, compactness-facing, or uniform-modulus consumer has no accepted source outside that split.

The split is therefore a source discipline rather than an extra compactness principle. Consumer rows may be appended after the source has been displayed, but the source itself remains $L, U, O, Q, D, S, R, E, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedInterval fromEventFlow exactness`

Theorem 1591.13 (Closed bounded interval root source scope route)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The accepted closed bounded interval root source route factors through dyadic interval rows, rational interval bounds, located interval covers, closed bounded interval net rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *endpoint seals before any compactness-facing or uniform-modulus consumer reads the interval. The displayed dependency order is*

$$L, U, O \longmapsto Q, D \longmapsto S, R, E \longmapsto H, C, P, N.$$

No compactness consumer, located-cover consumer, nested-window consumer, or uniform-modulus consumer may replace this route by open-cover compactness, selected subcovers, trichotomy, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The endpoint rows L, U and endpoint-order row O supply the rational interval bounds and located interval source before a downstream consumer can inspect a cover, mesh, or modulus row.

The containment rows Q, D then expose rational and dyadic interval data. These are the rows through which located interval covers and closed bounded interval net rows read the interval, so the consumer-facing cover data remain downstream of the same root source.

Next read the finite Real-facing route. The $\mathsf{StreamName}^{\uparrow}$ window S, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and terminal $\mathsf{Real}^{\uparrow}$ seal E expose endpoint names before compactness-facing or uniform-modulus rows are allowed to consume the interval.

The structural rows H, C, P, N preserve that source route by transport, replay, provenance, and local naming. Since the carrier contains no coordinate for the excluded compactness, choice, quotient, trichotomy, or ambient completeness objects, every named consumer factors through the displayed source route. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source scope route`

Theorem 1591.14 (Closed bounded interval root-source carrier admission)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C,$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Carrier admission for the root source is exactly the admission of a* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet whose rational cell row is read through* $\mathsf{RationalInterval}^{\uparrow}$*, whose refinement row is read through* $\mathsf{DyadicInterval}^{\uparrow}$*, whose endpoint windows are read through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, and whose terminal seal is read through* $\mathsf{Real}^{\uparrow}$*. The admitted BEDC rows are the displayed* BHist *entries $L, U, O, Q, D, S, R, E, H, C, P, N$, with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row; no compactness, Heine–Borel principle, selected point, trichotomy, or quotient real equality is part of admission.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source carrier admission`

Proof. Project the accepted carrier through Definition 1591.1. The endpoint rows L, U and order row O give the located interval source, while Q, D expose rational and dyadic containment as finite interval rows.

Read the endpoint names through the finite window route. The $\mathsf{StreamName}^{\uparrow}$ row S, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and $\mathsf{Real}^{\uparrow}$ seal E are the only Real-facing rows available before a consumer can inspect the interval.

The structural rows close admission. Transport by H, replay by C, provenance by P, and local naming by N preserve the same BHist packet, so carrier admission is exactly the displayed root source and not a compactness or Heine–Borel assertion. □

Theorem 1591.15 (Closed bounded interval root-source window exactness)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C,$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every window read of the root source exposes exactly the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows, the* $\mathsf{RationalInterval}^{\uparrow}$ *rational cell row, the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural rows H, C, P, N. Thus interval-net, located-cover, compactness-facing, and uniform-modulus consumers may refine the window only after the same $L, U, O, Q, D, S, R, E, H, C, P, N$ route has been displayed.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source window exactness`

Proof. Begin with the root source split. It already identifies $L, U, O, Q, D, S, R, E, H, C, P, N$ as the complete source packet for downstream consumers.

The window rows are not independent of the interval rows. Rational cells in Q and dyadic cells in D are read only with the endpoint windows S and regular readbacks R, and the Real seal E terminates that same route.

Transport and replay preserve exactness. Since H, C, P, N carry only the displayed packet, a downstream consumer cannot omit a window row, insert a host compact interval, or replace the source by selected subcover data before adding its own consumer rows. □

Theorem 1591.16 (Closed bounded interval root-source Real-seal route)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P,$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The root source reaches every Real-facing consumer through the finite route*

$$\mathsf{LocatedInterval}^{\uparrow}, \mathsf{RationalInterval}^{\uparrow}, \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}$$

The route records endpoint windows, rational and dyadic containment, regular readback, terminal Real sealing, componentwise transport, continuation replay, provenance, and local naming. It does

not assert compactness, Heine–Borel, selected endpoints, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source real seal route`

Proof. Read the interval side first. The $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, and $\mathsf{DyadicInterval}^{\uparrow}$ rows are represented by the endpoint, order, rational containment, and dyadic containment coordinates L, U, O, Q, D.

Pass to the seal side. The endpoint names are exposed only through $\mathsf{StreamName}^{\uparrow}$ windows S, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Finally attach the BEDC structural rows. The BHist packet, hsame transport, Cont replay, Pkg provenance, and NameCert row are exactly H, C, P, N together with the displayed carrier coordinates, so the Real-seal route has no coordinate for compactness, Heine–Borel, selected endpoints, quotient real equality, countable choice, or ambient completeness. □

Theorem 1591.17 (Closed bounded interval root containment split)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The containment part of the root source is exactly* L, U, O, Q, D *transported and replayed through* H, C *and recorded through* P, N*. Thus interval-net cells, compactness-facing finite covers, and uniform-modulus mesh requests must pass through the endpoint, order, rational-containment, and dyadic-containment rows of Definition 1591.1 and Definition 1591.3 before they read any mesh, cover, or modulus row.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root containment split`

Proof. Start with the endpoint-containment route. The carrier gives L, U, O, Q, D before the schedule and readback rows, and the NameCert obligation theorem admits containment only through those displayed coordinates.

The structural rows do not create new containment data. They preserve the same containment route by H, replay it by C, record its source through P, and name it through N.

Net, compactness-facing, and uniform-modulus consumers therefore receive containment only as a root source. A downstream mesh, cover, or modulus row may refine Q, D, but it cannot replace the endpoint and containment prefix by host interval membership, a selected point, or an open-cover compactness object. □

Theorem 1591.18 (Closed bounded interval root seal split)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The Real-seal part of the root source is the route* S, R, E *downstream of* L, U, O, Q, D *and upstream of every interval-net, compactness-facing, and uniform-modulus consumer. Together with* H, C, P, N*, Definition 1591.1 and Definition 1591.3 force all Real-facing reads through the displayed* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal route.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root seal split`

Proof. Read the carrier in order. The $\mathsf{StreamName}^{\uparrow}$ row S and $\mathsf{RegSeqRat}^{\uparrow}$ row R are attached to the same endpoint and containment source L, U, O, Q, D, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is available only after those rows are displayed.

Apply the NameCert obligation theorem. It treats S, R, E as local certificate obligations of the closed bounded interval packet, not as an ambient Real completeness theorem.

The replay and provenance rows keep the seal attached to the root source. Hence interval-net, compactness-facing, and uniform-modulus consumers may use Real-facing data only through

$L, U, O, Q, D, S, R, E, H, C, P, N$, with no quotient real equality, trichotomy, selected representative, selected subcover, or ambient completeness route. □

Theorem 1591.19 (Closed bounded interval root unblock obligations)**.** *The root obligations exposed by an accepted* ClosedBoundedInterval$^{\uparrow}$ *packet are endpoint admission, endpoint-order transport, rational containment exposure, dyadic containment exposure,* StreamName$^{\uparrow}$ *readback,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal nonescape, transport, replay, provenance, and local* NameCert *naming. Any downstream finite-net, located-cover, nested-window, or compact-uniform route must discharge those obligations before adding its own consumer rows.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root unblock obligations`

Proof. Read the NameCert obligation theorem and the endpoint-order transport theorem. Together they make endpoint admission and endpoint-order transport part of the local ClosedBoundedInterval$^{\uparrow}$ surface.

Read the density ledger and density readback theorems. They expose rational containment, dyadic containment, StreamName windows, RegSeqRat readbacks, and Real endpoint seals as rows of the same source package.

Apply source readback stability and the compact-uniform source package. Transport, replay, provenance, and local naming do not create additional interval data; they preserve the displayed package until the finite-net or compact-uniform consumer adds its own rows.

Thus the root obligations are exactly the listed rows. A downstream consumer that omits any one of them has not received an accepted ClosedBoundedInterval$^{\uparrow}$ source. □

Theorem 1591.20 (Closed bounded interval root-source dependency scope)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C,$ be an accepted* ClosedBoundedInterval$^{\uparrow}$ *carrier. The packet $L, U, O, Q, D, S, R, E, H, C, P, N$ is the required dependency prefix for every* ClosedBoundedIntervalNet$^{\uparrow}$, LocatedIntervalCover$^{\uparrow}$, NestedIntervalCompactness$^{\uparrow}$, *and* ClosedBoundedIntervalUniformModulus$^{\uparrow}$ *consumer of I. The interval-net route of chapter 1250, the located-cover route, the nested-compactness route of chapter 1584, and the compact-uniform route of chapter 1253 may add consumer rows only after that prefix has exposed endpoint admission, endpoint-order transport, rational and dyadic containment, StreamName windows, RegSeqRat readback, the Real seal, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source dependency scope`

Proof. Start with the root-source split and the root-source window exactness theorem. They identify the endpoint rows L, U, endpoint order O, rational and dyadic containment rows Q, D, finite window S, regular readback R, terminal Real seal E, and structural rows H, C, P, N as the complete source before a consumer reads mesh, cover, nesting, or modulus data.

Use the Real-seal route next. The route forces every Real-facing consumer through LocatedInterval$^{\uparrow}$, RationalInte then through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and finally through the BHist, hsame, Cont, Pkg, and NameCert rows. Thus the interval-net and located-cover consumers cannot detach their finite cells from the displayed source packet.

For nested compactness, apply the nested-window factorization already recorded in this file and read it against chapter 1584. The retained nested prefix is downstream of the same endpoint, containment, window, readback, seal, transport, replay, provenance, and naming rows.

For the compact-uniform consumer, apply the uniform-modulus source theorem and the compact-uniform route of chapter 1253. The modulus and compact-uniform rows are consumer rows, so they enter only after the same root packet has been displayed.

The downstream coverage surface and root consumer completeness theorem combine these four cases. Hence $L, U, O, Q, D, S, R, E, H, C, P, N$ is the common dependency prefix for the named L10 consumers, and none of them may substitute open-cover compactness, selected subcovers, trichotomy, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness for that prefix. □

Theorem 1591.21 (Closed bounded interval root-source scoped coverage)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, *and* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *consumer must first expose endpoint admission, endpoint-order transport, rational and dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *rows of the accepted interval packet. This scoped coverage is the common source surface supplied by the root-source split, root-source dependency scope, nested-window factorization, and uniform-modulus source theorems before any consumer-specific mesh, cover, nesting, modulus, or compact-uniform row is read.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root source scoped coverage`

Proof. Start with the root-source split. It identifies $L, U, O, Q, D, S, R, E, H, C, P, N$ as the endpoint, order, containment, window, readback, seal, transport, replay, provenance, and naming rows of the accepted interval packet.

Apply the root-source dependency scope. The theorem places every interval-net, located-cover, nested-compactness, and uniform-modulus consumer downstream of the same dependency prefix, so the named consumers cannot read their own mesh, cover, nesting, or modulus rows before the interval source has been displayed.

For nested-compactness consumers, use the nested-window factorization. It keeps the retained nested prefix behind the displayed endpoint, containment, readback, seal, and structural rows.

For uniform-modulus consumers, use the uniform-modulus source theorem. It places each modulus and compact-uniform row after the same root packet.

Thus the four consumer families have one scoped coverage surface: endpoint admission, endpoint-order transport, rational and dyadic containment, StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert naming. No consumer-specific row can replace that accepted interval packet by host compactness, selected subcovers, trichotomy, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1591.22 (Closed bounded interval root consumer completeness)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, *and* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *consumer factors through the complete source packet $L, U, O, Q, D, S, R, E, H, C, P, N$ before any consumer-specific row is read. The completeness surface composes the root source split, scoped source map, downstream coverage surface, finite-cover window exactness, nested-window factorization, and uniform-modulus source, and it explicitly excludes open-cover compactness, selected subcovers, trichotomy, quotient real equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root consumer completeness`

Proof. Begin with the root source split and root unblock obligations. They identify $L, U, O, Q, D, S, R, E, H, C, P, N$ as the endpoint, order, containment, window, readback, seal, transport, replay, provenance, and local naming rows that every downstream consumer must receive first.

Pass to the scoped source map. The definition of Σ_I records the same rows as the scoped source packet, and the downstream coverage surface states that located-cover, nested-window, finite-net, and compact-uniform consumers may only add their own rows after that packet has been displayed.

For interval-net and located-cover consumers, apply finite-cover window exactness. It forces mesh centers, cover cells, and refinement rows to remain downstream of the complete root packet.

For nested-window consumers, apply nested-window factorization. The retained nested prefix and compactness ledger are read only after $L, U, O, Q, D, S, R, E, H, C, P, N$ have been exposed.

For uniform-modulus consumers, apply the uniform-modulus source theorem. The modulus and compact-uniform rows are added after the same endpoint, containment, StreamName, RegSeqRat, Real-seal, and structural route.

The cited source theorems use no coordinate for open-cover compactness, selected subcovers, trichotomy, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Therefore the four named consumer families are complete exactly over the displayed root packet and over no stronger ambient principle. □

Theorem 1591.23 (Closed bounded interval downstream coverage surface)**.** *Every downstream located-cover, nested-window, finite-net, and compact-uniform consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier factors through the same source packet*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

The consumer may refine that packet by adding cover, nesting, mesh, modulus, or continuity rows, but the source interval itself remains the displayed carrier rather than an open-cover compactness object or host real interval.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket downstream coverage surface`

Proof. Begin with Definition 1591.71. It places every net, uniform-modulus, located-cover, nested-window, and compactness-facing consumer behind the same root rows.

Apply Definition 1591.72 for finite-net reads and the compactness consumer boundary for compact-uniform reads. Both routes require L, U, O, Q, D, S, R, E before consumer-specific cover, net, or modulus rows are inspected.

Finally use Definition 1591.19. The structural rows H, C, P, N carry the source across transport, replay, provenance, and local naming, so each downstream route factors through the same source packet and cannot replace it by host compactness data. □

Theorem 1591.24 (Closed bounded interval finite-cover window exactness)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C,$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-cover window read must expose exactly the source rows* $L, U, O, Q, D, S, R, E, H, C, P, N$ *before any mesh center, cover cell, or refinement row is read. In particular, Definition 1591.37, Definition 1591.69, and Definition 1591.23 make the finite-cover window an exact downstream read of the root packet, not a separate open-cover compactness object.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket finite cover window exactness`

Proof. Project the accepted carrier and the finite-cover window. The root rows L, U, O, Q, D, S, R, E supply endpoints, order, containment, StreamName scheduling, RegSeqRat readback, and the terminal Real seal before a finite-cover consumer can inspect its own rows.

Apply the source-window exhaustion and downstream coverage surface. These theorems force every interval-net and located-cover read through the same displayed source packet, so the finite-cover window cannot omit any of $L, U, O, Q, D, S, R, E, H, C, P, N$.

The structural rows finish the exactness claim. Transport by H, replay by C, provenance by P, and local naming by N preserve the source packet without adding selected subcovers, host compactness, quotient real equality, trichotomy, or ambient completeness. □

Theorem 1591.25 (Closed bounded interval nested-window factorization)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *nested-window consumer factors through the displayed root route $L, U, O, Q, D, S, R, E, H, C, P, N$ before it reads a retained nested prefix, compactness ledger, or real-seal boundary. The factorization is the composition of Definition 1591.46, Definition 1591.19, and Definition 1591.23.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket nested window factorization`

Proof. Start with the nested compactness handoff. It already places the nested-window consumer downstream of the endpoint-order row, rational and dyadic containment ledgers, StreamName window, RegSeqRat readback, and RealUp seal.

Use the root unblock obligations. They add the required transport, replay, provenance, and local naming rows H, C, P, N to the endpoint and readback route, so the nested consumer receives a complete $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source packet.

Finally apply the downstream coverage surface. The nested prefix may refine the displayed interval rows, but it cannot replace them by selected endpoints, quotient equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1591.26 (Closed bounded interval uniform-modulus source)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *consumer obtains its interval source from the rows $L, U, O, Q, D, S, R, E, H, C, P, N$ before reading any modulus, compact-uniform, or Heine–Cantor-style row. The source is therefore the displayed package of Definition 1591.68, Definition 1591.57, and Definition 1591.23, not an ambient compact interval theorem.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket uniform modulus source`

Proof. Read the source obligation package first. It identifies $L, U, O, Q, D, S, R, E, H, C, P, N$ as the lower endpoint, upper endpoint, order, containment, window, readback, seal, transport, replay, provenance, and naming rows available to interval consumers.

Apply the compact-uniform source package. A uniform-modulus consumer may add modulus and compact-uniform rows only after the displayed endpoint, containment, StreamName, RegSeqRat, and Real-seal route has been consumed.

The downstream coverage surface keeps the consumer inside that package. Since no row of I names open-cover compactness, selected subcovers, quotient real equality, trichotomy, or ambient completeness, the uniform-modulus source is exactly the finite root packet. □

Theorem 1591.27 (Closed bounded interval cover-refinement source)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$*, located-cover, or*

compact-uniform refinement consumer must read $L, U, O, Q, D, S, R, E, H, C, P, N$ before any refinement map, subcover ledger, Lebesgue-number row, or modulus row. The route is the same root source exposed by the finite-cover window exactness and uniform-modulus source theorems.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root cover refinement source`

Proof. Project the carrier. The endpoint, order, containment, window, readback, Real-seal, transport, replay, provenance, and local naming rows are exactly $L, U, O, Q, D, S, R, E, H, C, P, N$.

Apply Definition 1591.24. A net or located-cover consumer must expose the complete source packet before it may inspect mesh centers, cover cells, refinement rows, or a subcover ledger.

Apply Definition 1591.26. A compact-uniform or modulus consumer receives the same source packet before any Lebesgue-number row or modulus row is read.

Finally use Definition 1591.23. The downstream refinement surface remains inside the displayed packet and excludes host open-cover compactness, selected subcovers, quotient real equality, trichotomy, countable choice, and ambient completeness. □

Theorem 1591.28 (Closed bounded interval located endpoint grid)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. If I feeds an interval-net, compactness-facing, or uniform-modulus consumer, then any located endpoint grid or dyadic cell grid read must expose the endpoint rows L, U, endpoint order O, rational and dyadic containment rows Q, D,* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows H, C, P, N before the grid can be consumed. Thus grid readiness is downstream of the displayed root source packet and is not a route through selected interval points, trichotomy, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket located endpoint grid`

Proof. Project the accepted carrier by Definition 1591.1. The only endpoint-facing and containment-facing source coordinates available before a consumer row are $L, U, O, Q, D, S, R, E, H, C, P, N$.

Apply Definition 1591.12. It identifies those rows as the root source for interval-net, compactness-facing, and uniform-modulus consumers, so a located endpoint grid cannot be read before the endpoint, order, containment, readback, seal, transport, replay, provenance, and naming rows are displayed.

Use Definition 1591.24 for the dyadic cell side. A finite-cover or grid window must expose the complete source packet before any mesh center, cover cell, or refinement row is inspected.

Finally apply Definition 1591.27. Grid readiness is therefore downstream of the displayed source packet. Since selected points, trichotomy, quotient equality, and ambient Real completeness are not coordinates of that packet, no such object can serve as the grid source. □

Definition 1591.29 (Closed bounded interval scoped source map)**.** *The scoped source map for an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier is the finite packet*

$$\Sigma_I = (L, U, O, Q, D, S, R, E, H, C, P, N).$$

Here L and U are the lower and upper endpoint rows, O is the endpoint-order row, Q and D are the rational and dyadic containment ledgers, S is the finite $\mathsf{StreamName}^{\uparrow}$ *window, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row. The map has no*

coordinate for open-cover compactness, trichotomy, selected interval points, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1591.30 (Closed bounded interval scoped kernel surface)**.** *Every scoped consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier reads only the kernel rows displayed by Definition 1591.29: endpoint rows, endpoint order, rational and dyadic containment ledgers, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.*

Proof. Project the accepted carrier onto Σ_I. The carrier definition supplies exactly the endpoint rows L, U, the order row O, containment ledgers Q, D, window and readback rows S, R, seal row E, and structural rows H, C, P, N.

Any scoped consumer enters through the same projection. If it reads an endpoint, containment fact, dyadic cell, regular rational readback, or terminal Real seal, that read is one of the displayed coordinates of Σ_I.

Transport and replay do not add a new interval coordinate. They preserve the displayed source rows through H and C, while P and N record provenance and local naming. Hence every scoped consumer stays on the listed kernel surface. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket scoped kernel surface`

Theorem 1591.31 (Closed bounded interval net consumer readiness)**.** *Closed-bounded-interval net, uniform-modulus, and nested-compactness consumers become ready only through the displayed source map* Σ_I *of Definition 1591.29. They may add mesh, modulus, nesting, or compact-uniform rows after reading* Σ_I*, but they cannot become ready through open-cover compactness, trichotomy, selected points, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the scoped kernel surface. Definition 1591.30 forces every accepted source read through $L, U, O, Q, D, S, R, E, H, C, P, N$.

For net consumers, the mesh rows are downstream of the endpoint and containment rows. For uniform-modulus consumers, the modulus rows are downstream of the same dyadic containment and StreamName window. For nested-compactness consumers, each nested window is downstream of the endpoint-order row and the displayed readback route.

The excluded principles are not coordinates of Σ_I. Therefore readiness is exactly the displayed source-map route followed by consumer-specific rows, not a route through open-cover compactness, trichotomy, selected points, quotient equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket net consumer readiness`

Theorem 1591.32 (Closed bounded interval scoped source package)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The scoped source map* Σ_I *of Definition 1591.29, the scoped kernel surface of Definition 1591.30, the net consumer readiness of Definition 1591.31, the finite-cover window exactness of Definition 1591.24, the nested-window factorization of Definition 1591.25, and the uniform-modulus source of Definition 1591.26 jointly form the scoped L10 source package for* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*. Every endpoint, containment, finite-cover, nested-window, net, and uniform-modulus consumer reads through* $L, U, O, Q, D, S, R, E, H, C, P, N$ *before any consumer-specific row is available; no open-cover compactness, trichotomy, selected point, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate enters the package.*

Proof. Project the accepted carrier through Definition 1591.29. The projected packet is exactly $\Sigma_I = (L, U, O, Q, D, S, R, E, H, C, P, N)$, so endpoint, containment, readback, seal, transport, replay, provenance, and naming data have one displayed source.

Apply Definition 1591.30. Any scoped consumer that reads an endpoint, containment ledger, dyadic cell, regular rational readback, or Real seal must remain on the displayed kernel rows of Σ_I.

For the downstream rows, use Definition 1591.31, Definition 1591.24, Definition 1591.25, and Definition 1591.26. These theorems route net, located-cover, finite-cover, nested-window, and uniform-modulus consumers through the same endpoint, containment, window, readback, seal, and structural rows before the consumer-specific mesh, cover, nesting, or modulus rows are read.

The carrier and the cited routes contain no coordinate for open-cover compactness, trichotomy, selected points, quotient equality, or ambient Real completeness. Therefore the displayed rows form the scoped L10 source package and the excluded objects cannot serve as alternate sources. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket scoped source package`

Theorem 1591.33 (Closed bounded interval source scope package)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. The root-source route for I is scoped exactly by the dyadic interval endpoint rows L, U, located interval coverage O, Q, nested interval compactness windows, finite-cover windows, the* $\mathsf{Real}^{\uparrow}$ *seal E,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. Every Real or Completion consumer of the interval source must read through this package before inspecting consumer-specific mesh, cover, nesting, modulus, or completion rows, and no open-cover compactness, arbitrary Heine–Borel theorem, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is exported.*

Proof. Start with root-source carrier admission and window exactness. Definition 1591.14 and Definition 1591.15 expose the endpoint, located coverage, stream-window, readback, Real-seal, transport, replay, provenance, and naming rows as the displayed source.

Use the Real-seal route, finite-cover window exactness, and nested-window factorization. Definition 1591.16, Definition 1591.24, and Definition 1591.25 force Real, cover, and nested compactness consumers to pass through the same source route.

Apply Definition 1591.26 and Definition 1591.30. The uniform-modulus and scoped-kernel surfaces bind the consumer-facing rows to the same dyadic endpoint, located interval, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and NameCert package.

Therefore the source scope package is exactly the displayed carrier route. The excluded open-cover compactness, arbitrary Heine–Borel theorem, selected limit, and ambient completeness principles are not rows of I and cannot be exported by the packet. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source scope package`

Definition 1591.34 (Closed bounded interval source window)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$, the source window is the finite* BHist *row*

$$\Omega_I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

read as the endpoint-source packet before any interval-net, compactness, or uniform-modulus consumer opens its own rows. The lower and upper endpoint rows L, U are read through $\mathsf{LocatedInterval}^{\uparrow}$*, the dyadic containment ledger D is read through* $\mathsf{DyadicInterval}^{\uparrow}$*, the finite observation schedule S is read through* $\mathsf{StreamName}^{\uparrow}$*, the regular rational readback R is read through* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal endpoint seal E is read through* $\mathsf{Real}^{\uparrow}$*. The rows H, C, P, N record the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *naming for the same endpoint-source route.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalSourceWindow exactness`

Theorem 1591.35 (Closed bounded interval source transport)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ and $I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$ be accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers classified together by Definition 1591.2. Any endpoint source window transported from I to I' carries L, U, O, Q, D, S, R, E through the componentwise* hsame *rows of H and then replays the same route through the displayed* Cont *rows of C before any interval-net or compactness consumer reads it.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source transport`

Proof. Project the two carriers and the classifier. The rows L, U, O, Q, D, S, R, E and $L', U', O', Q', D', S', R', E'$ are the endpoint, containment, window, readback, and Real-seal coordinates being compared.

Use the source window of Definition 1591.34. It lists the same rows as a BHist endpoint-source packet, so transport of the source window is exactly componentwise transport of lower endpoint, upper endpoint, order, rational containment, dyadic containment, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal rows.

The replay step is downstream. The displayed Cont rows in C carry the transported endpoint-source route to consumers only after the hsame transport has preserved every listed coordinate. Thus an interval-net, compactness, or uniform-modulus consumer cannot read a transported source window before the endpoint rows have passed through H and C. □

Theorem 1591.36 (Closed bounded interval source nonescape)**.** *Every interval-net, nested-window, located-cover, compactness-facing, or uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source window factors through*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

In particular, the consumer first reads the BHist *endpoint-source rows, the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint route, the* $\mathsf{DyadicInterval}^{\uparrow}$ *containment ledger, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal before it reads any downstream interval-net or compactness rows. The source window cannot expose a host interval set, open-cover compactness, selected point, selected subcover, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source nonescape`

Proof. Start from the source window. By Definition 1591.34, every endpoint-source read is one of $L, U, O, Q, D, S, R, E, H, C, P, N$.

Next apply source transport. Theorem 1591.35 forces transported endpoint-source reads through the hsame and Cont rows before any downstream consumer can inspect them.

Finally compare with the existing finite-cover, dyadic-net, located-cover, and compactness handoffs in this chapter. Each of those consumers reads the closed bounded interval through the displayed endpoint, containment, schedule, readback, seal, transport, replay, provenance, and local naming rows. Since none of those rows is a host interval set, open cover, selected point, quotient equality, trichotomy decision, choice selector, or ambient completeness theorem, the source window has no route to any excluded object. □

Definition 1591.37 (Closed bounded interval finite-cover window)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ as in Definition 1591.1, a finite-cover window is a finite* BHist *row*

$$W_I = (S_0, R_0, Q_0, D_0, E_0)$$

whose $\mathsf{StreamName}^{\uparrow}$ *coordinate* S_0 *is a subwindow of the carried row* S*, whose* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate* R_0 *is read through* R*, whose rational and dyadic containment ledgers* Q_0, D_0 *are subledgers of* Q, D*, and whose endpoint seal* E_0 *is reached through the same* $\mathsf{Real}^{\uparrow}$ *row* E*. The window is admissible only when its lower-endpoint, upper-endpoint, endpoint-order, rational-containment, dyadic-containment, readback, and seal reads are all classifier-preserved by the carrier rows* H *and* C *of Definition 1591.2.*

Lean4 (definition) `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalFiniteCoverWindow`

Theorem 1591.38 (Closed bounded interval finite-net source)**.** *Every finite-net consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier*

$$I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

must read the lower endpoint L*, upper endpoint* U*, endpoint-order row* O*, rational containment ledger* Q*, and dyadic containment ledger* D *before any net row is available. The* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *are downstream of the same endpoint and containment route, with* H, C, P, N *preserving the route under transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket finite net source`

Proof. Start with the carrier of Definition 1591.1. Its finite-net source coordinates are the lower endpoint, upper endpoint, endpoint order, rational containment, and dyadic containment rows L, U, O, Q, D.

Apply the classifier and source-window definitions. The classifier of Definition 1591.2 preserves these rows componentwise, while Definition 1591.34 reads them before any downstream interval-net, compactness, or uniform-modulus consumer opens its own rows.

Finally use source nonescape, Definition 1591.36. It forces every interval-net read through $L, U, O, Q, D, S, R, E, H, C, P, N$, so the endpoint and containment rows are the required source before any finite-net row can be consumed. □

Theorem 1591.39 (Closed bounded interval RegSeqRat window)**.** *For accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers*

$$I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

and

$$I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$$

classified together, each finite-cover window preserves its $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows under componentwise* hsame *transport. In particular, every transported window carries* S, R *to* S', R' *only after* L, U, O, Q, D *have been transported to* L', U', O', Q', D' *and replayed through the displayed* Cont *rows.*

Proof. Project the two carriers and the classifier of Definition 1591.2. The classifier compares endpoint, containment, window, readback, and seal rows only by componentwise hsame transport and displayed replay.

Use the finite-cover window of Definition 1591.37. Its S_0 and R_0 rows are subwindow and readback rows of the carried S and R, while Q_0, D_0, E_0 remain attached to the same endpoint and containment route.

Now apply source transport, Definition 1591.35. It transports the endpoint and containment rows before the StreamName and RegSeqRat rows are available to consumers. Hence a transported

finite-cover window preserves the S, R readback route under the same componentwise transport and replay. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket regseqrat window`

Theorem 1591.40 (Closed bounded interval Real seal nonescape)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *exports only the endpoint route displayed by* $L, U, O, Q, D, S, R, H, C, P, N$*. It exports no open-cover compactness, selected point, selected subcover, quotient equality of Real names, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalPacket real seal nonescape`

Proof. Read the carrier and source window. The seal row E is reached only after the endpoint rows L, U, endpoint order O, rational and dyadic containment ledgers Q, D, StreamName window S, and RegSeqRat readback R have been displayed.

Apply source nonescape and finite-cover nonescape. Theorems 1591.36 and 1591.41 keep every consumer route inside the displayed endpoint, containment, window, readback, seal, transport, replay, provenance, and local naming rows.

The excluded objects have no carrier coordinate and no replay edge. Since H, C, P, N only transport, replay, source, and name the same finite route, the Real seal cannot export open-cover compactness, selected points or subcovers, quotient equality, trichotomy, countable choice, or ambient Real completeness. □

Theorem 1591.41 (Closed bounded interval finite-cover nonescape)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier, and let* $W_I = (S_0, R_0, Q_0, D_0, E_0)$ *be a finite-cover window in the sense of Definition 1591.37. Every finite-cover read exported to an interval-net, nested-window, or uniform-modulus consumer factors through*

$$L, U, O, Q_0, D_0, S_0, R_0, E_0, H, C, P, N.$$

Consequently the read cannot escape to a host open cover, a selected subcover, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite-cover window. By definition, S_0 and R_0 are read through the carried $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, while Q_0 and D_0 are subledgers of the rational and dyadic containment rows attached to the same endpoints.

The endpoint route is fixed before the cover window is visible. The rows L, U, O must be displayed before Q_0, D_0, S_0, R_0 can be read, and the terminal seal E_0 is reached only through the carried $\mathsf{Real}^{\uparrow}$ row E.

The structural rows close the nonescape claim. Since H and C preserve exactly the carrier and window rows named above, and since P, N only source and name that route, no exported finite-cover read contains a host open cover, selected subcover, quotient real equality, trichotomy, choice selector, or ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket finite cover nonescape`

Theorem 1591.42 (Closed bounded interval finite-net compactness obligations)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier, and let* $W_I = (S_0, R_0, Q_0, D_0, E_0)$ *be a finite-cover window in the sense of Definition 1591.37. Any finite-net compactness consumer must expose the endpoint rows* L, U, O*, the rational and dyadic containment rows* Q, D*, the* $\mathsf{StreamName}^{\uparrow}$ *and*

$\mathsf{RegSeqRat}^{\uparrow}$ *rows* S, R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the transport, replay, provenance, and local naming rows* H, C, P, N *before it may read mesh centers, finite cover refinements, or interval compactness ledgers.*

Consequently such a consumer factors through Definition 1591.1, Definition 1591.37, Definition 1591.36, and Definition 1591.45. It does not export open-cover compactness, selected subcovers, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket finite net compactness route`

Proof. Project the carrier and finite-cover window. The carrier exposes only $L, U, O, Q, D, S, R, E, H, C, P, N$, while the window exposes S_0, R_0, Q_0, D_0, E_0 as subwindow, readback, containment, dyadic, and seal rows attached to the same endpoint route.

Apply source nonescape. Theorem 1591.36 keeps every interval-net or compactness-facing read downstream of the displayed endpoint, containment, window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Apply the net handoff. Theorem 1591.45 allows a net consumer to add mesh-specific cells, centers, and coverage data only after the base interval route has been displayed. Hence finite cover refinements and compactness ledgers remain downstream of $L, U, O, Q, D, S, R, E, H, C, P, N$.

The excluded compactness principles have no coordinate in either displayed packet. Since neither the carrier nor the window stores an open cover, selected subcover, choice selector, quotient-real equality, trichotomy decision, or ambient completeness datum, the finite-net compactness consumer cannot read any of them from this obligation surface. □

Theorem 1591.43 (Closed bounded interval located endpoint readback)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, every endpoint-facing* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *readback of the lower or upper endpoint factors through the displayed route*

$$L, U, O \longmapsto S, R \longmapsto E \longmapsto H, C, P, N.$$

The readback therefore preserves the endpoint-order row and the rational and dyadic containment ledgers of Definition 1591.1; it does not replace the endpoints by host reals or by quotient representatives.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket located endpoint readback`

Proof. Start with an endpoint-facing request. The carrier exposes lower and upper endpoint data only as L and U, and the order row O is carried before any rational, dyadic, stream, or regular-readback inspection.

The finite readback rows are next. The $\mathsf{StreamName}^{\uparrow}$ windows in S and the $\mathsf{RegSeqRat}^{\uparrow}$ rows in R inspect the displayed endpoint names and containment ledgers, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of those finite windows.

Finally apply the classifier of Definition 1591.2. Its componentwise hsame rows and displayed Cont replay preserve the same endpoint, order, containment, window, readback, and seal coordinates, so the readback cannot be reinterpreted as a host-real endpoint or as a quotient representative outside the accepted packet. □

Theorem 1591.44 (Closed bounded interval endpoint-order transport)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N$ *and* $I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers*

classified together by Definition 1591.2. Every containment read transported from I to I' carries the endpoint-order row O to O' before it transports the rational ledger Q to Q' or the dyadic ledger D to D'. Consequently no consumer can preserve containment while dropping endpoint order or replacing it by a host-level comparison.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket endpoint order transport`

Proof. Start from the classifier. It classifies accepted carriers only by componentwise hsame transport and replay of the same endpoint-to-containment-to-seal route.

The endpoint-order coordinate is upstream of both containment ledgers. A transported rational or dyadic containment read therefore has already transported L, U, O to L', U', O' before it reaches Q', D'.

The replay rows preserve the same order. Since neither the carrier nor the classifier contains a host comparison, trichotomy row, quotient-real equality witness, or alternate endpoint selector, the transported containment read cannot drop O' or replace it by data outside the accepted packet. □

Theorem 1591.45 (Closed bounded interval net handoff)**.** *Every* ClosedBoundedIntervalNet$^{\uparrow}$ *or* ClosedBoundedIntervalUniformModulus$^{\uparrow}$ *consumer of an accepted* ClosedBoundedInterval$^{\uparrow}$ *carrier reads the base interval through the finite route*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

The interval-net chapter chapter 1250 may add mesh scales, cells, centers, and coverage rows only after the base endpoint and containment route has been displayed. The uniform-modulus chapter chapter 1253 may read Heine–Cantor interval data only through that base route and its downstream finite-cover ledgers.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket net handoff`

Proof. Read the base carrier first. The route $L, U, O, Q, D, S, R, E, H, C, P, N$ exposes lower and upper located endpoints, endpoint order, rational and dyadic containment, finite windows, readbacks, Real sealing, transport, replay, provenance, and local naming.

The interval-net handoff then appends its mesh-specific rows. Its finite cells, dyadic refinements, centers, and coverage ledger are downstream of the displayed endpoint and containment rows, so the net packet cannot treat the mesh as the interval object itself.

The uniform-modulus handoff is later still. It reads compactness and modulus data through finite cover and Lebesgue-number ledgers, while the source interval remains the accepted ClosedBoundedInterval$^{\uparrow}$ route. Thus no consumer receives an open cover, selected subcover, quotient real equality, host interval set, or ambient completeness row in place of the base carrier. □

Theorem 1591.46 (Closed bounded interval nested compactness handoff)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C, P$ be an accepted* ClosedBoundedInterval$^{\uparrow}$ *carrier, and let a finite prefix of a nested-interval display be read inside the same closed bounded interval. The accepted packet supplies the endpoint-order row O, the rational and dyadic containment ledgers Q, D, the* StreamName$^{\uparrow}$ *window row S, the* RegSeqRat$^{\uparrow}$ *readback row R, and the* Real$^{\uparrow}$ *seal row E required by the finite-window* NestedIntervalCompactness$^{\uparrow}$ *route. The handoff is therefore a row-level bridge from the closed bounded interval carrier to the nested-window consumer of Definition 1584.5; it does not export open-cover compactness, selected endpoints, quotient real equality, trichotomy, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket nested compactness handoff`

Proof. Start with the accepted $\mathsf{ClosedBoundedInterval}^{\uparrow}$ carrier. By Definition 1591.44, any transported containment read keeps the endpoint-order row upstream of the rational and dyadic ledgers.

Apply Definition 1591.45. The finite consumer receives the base interval only through the displayed route $L, U, O, Q, D, S, R, E, H, C, P, N$, so a nested prefix displayed inside that interval must inherit the same endpoint, containment, window, readback, and seal rows.

The nested-window side then reads exactly the finite route isolated by Definition 1584.5. Since neither source packet contains an open cover, endpoint selector, quotient-real equality, trichotomy decision, countable-choice selector, or ambient completeness datum, the handoff cannot add any such row. □

Theorem 1591.47 (Closed bounded interval Cantor intersection handoff)**.** *Let a decreasing finite-window family of accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *packets be consumed by* $\mathsf{CantorIntersection}^{\uparrow}$*. The consumer may read the family only through endpoint rows, endpoint-order transport, rational and dyadic containment ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *rows of the accepted packets.*

The handoff supplies the finite nested-window data needed by the Cantor-intersection route. It does not export open-cover compactness, a choice-selected point, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with endpoint-order transport. The endpoint classifier stability route keeps lower endpoints, upper endpoints, and order rows upstream of every containment read in the decreasing family.

Use the finite-net source and RegSeqRat window theorems. They place rational containment ledgers, dyadic containment ledgers, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks inside the displayed finite packet before any nested-family consumer can inspect the windows.

The Real seal nonescape theorem then keeps each endpoint and containment read behind the same $\mathsf{Real}^{\uparrow}$ seal row. Thus the Cantor-facing consumer receives sealed finite windows, not quotient real representatives or an ambient completeness theorem.

Finally apply the nested compactness handoff Definition 1591.46. It already routes nested-window compactness through the accepted closed bounded interval packet, so the Cantor-intersection handoff is exactly the same finite endpoint, containment, window, readback, seal, transport, replay, provenance, and local-name surface. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket cantor intersection handoff`

Theorem 1591.48 (Closed bounded interval dyadic-net obligations)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every* $\mathsf{DyadicInterval}^{\uparrow}$ *net or* $\mathsf{LocatedInterval}^{\uparrow}$ *refinement read attached to* I *must display the endpoint rows* L, U*, the endpoint-order row* O*, the rational and dyadic containment ledgers* Q, D*, and the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readback route* S, R, E *before the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row is consumed. The dyadic-net obligations are therefore rows of the same* BHist *packet and not an open-cover compactness principle, a selected mesh point, quotient real equality, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket dyadic net obligations`

Proof. Project the accepted carrier. Its endpoint, order, rational containment, dyadic containment, finite-window, regular readback, and Real seal rows are already displayed as L, U, O, Q, D, S, R, E.

A dyadic-net consumer may refine the interval only through the dyadic ledger D and the rational containment ledger Q over the same located endpoints. The rows S, R, E then read those finite tests through StreamName, RegSeqRat, and Real-facing windows without replacing the interval by a host set.

Transport and replay preserve the same route. Since H, C, P, N only carry componentwise hsame, displayed Cont, provenance, and local naming for the listed rows, no dyadic-net read can add an open cover, selected mesh point, quotient equality, trichotomy decision, choice schedule, or ambient completeness coordinate. □

Theorem 1591.49 (Closed bounded interval containment-ledger obligations)**.** *Let $I = (L, U, O, Q, D, S, R, E, H, C$ be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every containment-ledger obligation for I is a* BHist *obligation over the displayed endpoint rows L, U, endpoint-order row O, rational and dyadic containment rows Q, D, componentwise* hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row N. The ledger must preserve endpoint order before rational or dyadic containment is exported, and it cannot read host interval sets, quotient real equality, trichotomy, selected points, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket containment ledger obligations`

Proof. Project the carrier of Definition 1591.1 and the classifier of Definition 1591.2. The displayed endpoint rows, order row, containment ledgers, transport, replay, provenance, and local name are exactly $L, U, O, Q, D, H, C, P, N$.

Apply source nonescape, finite-cover nonescape, and endpoint-order transport. Theorems 1591.36, 1591.41, and 1591.44 force every containment read to keep O upstream of Q and D and to pass through the displayed hsame and Cont rows.

Finally use the dyadic-net obligations of Definition 1591.48. They place every dyadic or located refinement inside the same BHist packet, while P and N only source and name that route. Thus no excluded host object is introduced by the containment ledger. □

Theorem 1591.50 (Closed bounded interval finite-window transport obligations)**.** *Let $I = (L, U, O, Q, D, S, R, E,$ and $I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$ be accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers classified together. Every finite-window transport obligation from I to I' is a* BHist *transport obligation that carries S, R, E to S', R', E' only after L, U, O, Q, D have been transported to L', U', O', Q', D' through the displayed* hsame *row H and replayed by the displayed* Cont *row C. The transported window remains sourced by* Pkg *provenance and named by* NameCert*, not by host equality of intervals or selected representatives.*

Proof. Start with source transport. Theorem 1591.35 says that a transported source window carries lower endpoint, upper endpoint, endpoint order, containment, window, readback, and seal rows componentwise before any downstream consumer reads them.

Use located endpoint readback and finite-cover nonescape. Theorems 1591.43 and 1591.41 identify S, R, E as downstream finite windows over the same endpoint and containment route, so their transport cannot be separated from L, U, O, Q, D.

The nested compactness handoff and dyadic-net obligations preserve the same route for later consumers. Theorems 1591.46 and 1591.48 keep the finite-window rows inside the displayed BHist packet, with P, N recording provenance and local naming only. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket finite window transport obligations`

Theorem 1591.51 (Closed bounded interval consumer nonescape obligations)**.** *Every interval-net, located-cover, nested-compactness, finite-window, or uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier factors its* BHist *read through*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

The consumer therefore sees hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *naming for the displayed carrier rows before it sees any downstream conclusion. It cannot escape to open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First project the carrier and source nonescape route. The carrier definition and Definition 1591.36 expose only $L, U, O, Q, D, S, R, E, H, C, P, N$ to endpoint-source consumers.

Next compose the finite-cover, located-cover, endpoint-order, and dyadic-net obligations. Theorems 1591.41, 1591.52, 1591.44, and 1591.48 keep every net or cover read downstream of the same displayed endpoint, containment, window, readback, and seal rows.

Finally apply nested compactness and compactness consumer boundary. Theorems 1591.46 and 1591.54 show that later compactness-facing consumers inherit the same finite packet. Since the packet contains no open-cover compactness theorem, selected object, quotient equality, trichotomy decision, countable-choice row, or ambient completeness row, the consumer obligations are nonescaping. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket consumer nonescape obligations`

Theorem 1591.52 (Closed bounded interval located-cover handoff)**.** *Every located-cover consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier factors through the displayed route*

$$L, U, O, Q, D, S, R, E, H, C, P, N$$

before reading any $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *or compactness-facing row. The* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{RationalInterval}^{\uparrow}$ *data supply finite containment tests, the* $\mathsf{DyadicInterval}^{\uparrow}$ *ledger supplies the mesh boundary, and the* $\mathsf{Real}^{\uparrow}$ *readback remains downstream of the same displayed packet.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket located cover handoff`

Proof. Start with the located endpoint surface. The rows L, U, O must be visible before a located-cover row can decide which rational or dyadic containment tests are being inspected.

The cover handoff then reads Q, D, S, R, E. Rational and dyadic ledgers provide finite containment data, while StreamName and RegSeqRat windows carry those tests to the Real seal without choosing a subcover or a point of the interval.

Finally use H, C, P, N. These structural rows replay and name exactly the displayed route, so a compactness-facing consumer receives a located-cover handoff over the accepted packet and not an open-cover theorem, quotient real equality, trichotomy, countable choice, or ambient completeness. □

Theorem 1591.53 (Closed bounded interval located-cover source totality)**.** *Any located-cover consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *must read the endpoint rows* L, U, O*, containment ledgers* Q, D*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N *before consuming a* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row. The located-cover source is therefore the full displayed endpoint-and-containment packet, not a parameter echo of the cover consumer.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket located cover source totality`

Proof. Project the carrier and the source window. The carrier of Definition 1591.1 lists $L, U, O, Q, D, S, R, E, H, C,$ while Definition 1591.34 reads the same rows as the endpoint-source packet.

Apply source nonescape and finite-net source. Theorems 1591.36 and 1591.38 force every interval-net, located-cover, or compactness-facing read through the endpoint rows, containment ledgers, observation schedule, readback, and seal before any downstream cover row is available.

Finally use the located-cover handoff. Theorem 1591.52 keeps the cover consumer on the same displayed route, with H, C, P, N preserving replay, provenance, and local naming. Thus a $\mathsf{LocatedIntervalCover}^{\uparrow}$ row can only be consumed after the full source package has been exposed. □

Theorem 1591.54 (Closed bounded interval compactness consumer boundary)**.** *Any* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *consumer, finite-net compactness consumer, or Real-completion consumer that reads an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *packet must keep the endpoint rows, dyadic-net rows, located-cover rows, and nested compactness rows displayed before any compactness conclusion is consumed. Thus every such consumer factors through the carrier route of Definition 1591.1, the dyadic-net obligations of Definition 1591.48, the located-cover handoff of Definition 1591.52, and the nested compactness handoff of Definition 1591.46.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket compactness consumer boundary`

Proof. First read the source carrier. The endpoint, order, containment, window, readback, seal, transport, replay, provenance, and naming rows of I are the only rows a consumer can project from $\mathsf{ClosedBoundedInterval}^{\uparrow}$.

Next apply the dyadic-net and located-cover handoffs. They force finite mesh and cover consumers to pass through Q, D, S, R, E over the same L, U, O endpoint route.

Finally pass to nested compactness. The nested-window consumer reads the same displayed interval data before it uses any compactness-facing row, so the boundary excludes open-cover compactness, selected subcovers, selected endpoints, quotient real equality, trichotomy, countable choice, and ambient completeness as hidden inputs. □

Theorem 1591.55 (Closed bounded interval endpoint classifier stability)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P,$ *and* $I' = (L', U', O', Q', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carriers classified together. Any classifier-preserved endpoint read from* I *to* I' *keeps the lower endpoint row* L*, upper endpoint row* U*, and endpoint-order row* O *upstream of the rational containment ledger* Q*, dyadic containment ledger* D*,* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row* N*.*

Proof. Start with the classifier of Definition 1591.2. It classifies two carriers only by transporting L, U, O, Q, D, S, R, E componentwise and replaying the same endpoint-to-containment-to-seal route through the displayed Cont rows.

The endpoint rows are therefore the source of the route. The finite-net source theorem and endpoint-order transport theorem, Definition 1591.38 and Definition 1591.44, place L, U, O before any rational, dyadic, window, readback, or seal row can be consumed.

Finally read the structural rows. The rows H, C, P, N transport, replay, source, and name exactly that displayed route, so classifier preservation cannot expose $Q, D, S, R, E, H, C, P, N$ before the endpoint classifier has already preserved L, U, O. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket endpoint classifier stability`

Theorem 1591.56 (Closed bounded interval dyadic containment totality)**.** *Every rational or dyadic containment consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier factors through the rational containment ledger* Q *and dyadic containment ledger* D *over the displayed* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route. Thus containment totality for this packet is a finite ledger-totality obligation over* $L, U, O, Q, D, S, R, E, H, C, P, N$, *not a host interval set, an open-cover principle, a selected point, quotient real equality, trichotomy, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalPacket dyadic containment totality`

Proof. Project the containment surface of the carrier. The rows Q and D are the only rational and dyadic containment ledgers, and by Definition 1591.1 they are attached to the lower endpoint, upper endpoint, and endpoint-order rows L, U, O.

Next use the located-cover handoff and dyadic-net obligations. Theorems 1591.52 and 1591.48 force every located or dyadic refinement through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, Q, and D before any compactness-facing read.

The readback route is finite as well. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows S, R, E inspect the same containment windows, while H, C, P, N preserve and name the route. Hence every containment consumer is accounted for by the displayed ledger route and no excluded host object is introduced. □

Theorem 1591.57 (Closed bounded interval compact-uniform source package)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier, the finite-net source, located-cover handoff, nested-compactness handoff, and compactness-consumer boundary assemble a compact-uniform source package*

$$L, U, O, Q, D, S, R, E, H, C, P, N$$

for closed bounded interval nets and uniform-modulus consumers. The package records endpoint classifier stability, rational and dyadic containment totality, finite-window readback, Real sealing, transport, replay, provenance, and local naming before any net, nested-window, compactness-facing, or uniform-modulus conclusion is consumed.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket compact uniform source package`

Proof. First build the finite source package. The finite-net source theorem and endpoint classifier stability identify L, U, O as the stable endpoint classifier rows and place Q, D downstream as the rational and dyadic containment ledgers.

Next attach the cover and compactness routes. The located-cover handoff, nested-compactness handoff, and compactness-consumer boundary, Definition 1591.52, Definition 1591.46, and Definition 1591.54, keep every finite-net, located-cover, and nested-window consumer inside the same displayed packet.

Finally add the uniform readback surface. The dyadic containment totality theorem supplies Q, D, S, R, E as the ledger and readback route, and H, C, P, N preserve, replay, source, and name it. Therefore closed bounded interval nets and uniform-modulus consumers receive a single compact-uniform source package without open-cover compactness, selected subcovers, selected

points, quotient real equality, trichotomy, countable choice, or ambient completeness as hidden inputs. □

Falsifiable prediction: Every interval-net or uniform-modulus read of a ClosedBoundedIntervalUp packet factors through displayed lower and upper located endpoints, endpoint order, rational and dyadic containment ledgers, StreamName windows, RegSeqRat readback, and a RealUp seal.

Independence witness: locatedinterval, closedboundedintervalnet, closedboundedintervaluniformmodulus, real: $\mathsf{ClosedBoundedInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower and upper located endpoints, endpoint order, rational and dyadic containment ledgers, finite readback windows, a RealUp seal, transport, replay, provenance, and local naming.

Definition 1591.58 (Closed bounded interval constructive source)**.** *A constructive source for a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *packet is a finite* BHist *source*

$$J = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

where L and U are located endpoint rows, O records the endpoint order, Q and D record rational and dyadic containment ledgers, S is the $\mathsf{StreamName}^{\uparrow}$ *finite-window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, E is the* $\mathsf{Real}^{\uparrow}$ *seal row, H is the componentwise* hsame *transport row, C is the displayed* Cont *replay row, P is the* Pkg *provenance row, and N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedInterval}^{\uparrow}}$ *row. The source has no coordinate for open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalConstructiveSource`

Theorem 1591.59 (Closed bounded interval constructive carrier)**.** *Every constructive source $J = (L, U, O, Q, D, S, R, E, H, C, P, N)$ for* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *exposes exactly the root carrier rows of Definition 1591.1: located lower endpoint, located upper endpoint, endpoint order, rational containment, dyadic containment, StreamName window, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the source of Definition 1591.58. Its coordinates are precisely the rows $L, U, O, Q, D, S, R, E, H, C, P,$

Compare with the carrier definition. The root carrier of Definition 1591.1 has the same endpoint, containment, finite-window, readback, seal, transport, replay, provenance, and naming rows, and no additional coordinate is available to a carrier projection.

The excluded host objects are absent from the source and from the carrier. Therefore the constructive source exposes exactly the root carrier rows and cannot extend them by compactness, choice, quotient equality, trichotomy, or ambient completeness data. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket constructive carrier`

Theorem 1591.60 (Closed bounded interval constructive handoff)**.** *Every net, nested-window, finite-cover, located-cover, and uniform-modulus consumer of a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *constructive source factors through*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

The factorization first reads located endpoints and endpoint order, then rational and dyadic containment, then finite windows, RegSeqRat readback, the Real seal, transport, replay, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket constructive handoff`

Proof. Begin with the root carrier route. By Definition 1591.59, any consumer of the constructive source can project only $L, U, O, Q, D, S, R, E, H, C, P, N$.

Net and finite-cover consumers use the containment ledgers. The finite-net source, consumer nonescape obligations, dyadic containment totality, and located-cover handoff force such reads through L, U, O, Q, D before they reach S, R, E.

Nested-window and uniform-modulus consumers use the same readback route. The nested compactness handoff and compact-uniform source package keep S, R, E, H, C, P, N attached to the endpoint and containment rows, so these consumers cannot bypass the displayed source. □

Definition 1591.61 (Closed bounded interval Cauchy modulus window)**.** *A Cauchy-modulus window for a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *constructive source is the finite subpacket*

$$W_{\mathsf{mod}} = (L, U, O, D, S, R, E, M)$$

where L and U are the located endpoint rows, O is the endpoint-order row, D is the dyadic containment and tolerance ledger, S is the $\mathsf{StreamName}^{\uparrow}$ *finite-window row, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and M is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *consumer slot. The window exposes endpoint order, dyadic containment, finite StreamName windows, RegSeqRat readback, and the RealCauchyModulus-facing slot before any compactness, net, or uniform-modulus consumer may inspect the interval packet.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalCauchyModulusWindow single carrier alignment`

Theorem 1591.62 (Closed bounded interval modulus window transport)**.** *Every classifier transport of a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *constructive source preserves the Cauchy-modulus window $W_{\mathsf{mod}} = (L, U, O, D, S, R, E, M)$ componentwise. In particular, the transported packet keeps the same endpoint-order row, dyadic containment ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and* $\mathsf{RealCauchyModulus}^{\uparrow}$ *consumer slot.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket modulus window transport`

Proof. Project the source of Definition 1591.58. The rows L, U, O, D, S, R, E are already displayed as part of the constructive source, and Definition 1591.61 only repackages them with the modulus consumer slot M.

Apply endpoint classifier stability to L, U, O. The transport cannot alter the lower endpoint, upper endpoint, or endpoint-order route before the containment rows are read.

Apply dyadic containment totality and the compact-uniform source package to D, S, R, E. These results keep the dyadic ledger attached to the same finite StreamName window, RegSeqRat readback, and Real seal.

Finally route the preserved rows to the $\mathsf{RealCauchyModulus}^{\uparrow}$ surface of chapter 1423. Since the modulus slot reads only D, S, R, E and the endpoint-order rows already transported above, it is preserved componentwise as part of the same finite window. □

Theorem 1591.63 (Closed bounded interval RealCauchyModulus handoff)**.** *Every* $\mathsf{RealCauchyModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *constructive source receives its precision data through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

after reading the endpoint order and containment rows L, U, O, D. The handoff supplies the modulus consumer with dyadic tolerance data, finite observation windows, regular rational readback, and the Real seal, but not open-cover compactness, selected subcovers, quotient real equality, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket real cauchy modulus handoff`

Proof. Start from the window of Definition 1591.61. The endpoint rows L, U and the order row O are read before the dyadic ledger D, so the consumer receives a bounded interval request rather than a detached Real name.

The dyadic row D supplies the tolerance and containment data through chapter 291. The StreamName and RegSeqRat rows S, R then provide the finite observation window and regular rational readback through chapter 126 and chapter 301.

The terminal seal E is read only after that route reaches chapter 38. The $\mathsf{RealCauchyModulus}^{\uparrow}$ consumer of chapter 1423 therefore receives the displayed tolerance, window, readback, and seal data without gaining any excluded compactness or choice principle. □

Theorem 1591.64 (Closed bounded interval Cauchy modulus nonescape)**.** *No* $\mathsf{RealCauchyModulus}^{\uparrow}$ *read admitted by a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *Cauchy-modulus window can bypass endpoint order, dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Equivalently, every accepted modulus read factors through L, U, O, D, S, R, E, M and cannot introduce open-cover compactness, selected points, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1591.63. It fixes the RealCauchyModulus-facing route through endpoint order, dyadic tolerance and containment, finite StreamName windows, RegSeqRat readback, and the Real seal.

The nonescape claim follows from the carrier inventory. Neither Definition 1591.1 nor Definition 1591.61 contains a coordinate for open-cover compactness, selected points, quotient real equality, trichotomy, countable choice, or ambient Real completeness.

Transport and replay rows do not add analytic data. They only preserve the displayed route already used by chapter 1423, chapter 126, chapter 301, chapter 291, and chapter 38. Hence every accepted modulus read remains inside L, U, O, D, S, R, E, M. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket cauchy modulus nonescape`

Definition 1591.65 (Closed bounded interval density ledger)**.** *For an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$, the root density ledger is the finite* BHist *packet*

$$\Delta_I = (L, U, O, Q, D, S, R, E, H, C, P, N)$$

read as the source for dyadic mesh cells and interval-net consumers. The rows L, U, O fix the endpoint route, Q records rational containment tests, D records finite dyadic mesh cells and dyadic containment comparisons, S records the $\mathsf{StreamName}^{\uparrow}$ *finite windows, R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, E records the* $\mathsf{Real}^{\uparrow}$ *endpoint seals, and H, C, P, N record transport, replay, provenance, and local naming for the same endpoint-and-containment surface.*

Lean4 (definition) `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalDensityLedger`

Theorem 1591.66 (Closed bounded interval density readback)**.** *Every density or interval-net readback admitted by an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier factors through the root density ledger*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

In particular, dyadic mesh cells and rational containment tests are read after the endpoint route L, U, O*, the finite observation windows and regular rational readbacks are read after* Q, D*, and the endpoint seals are read through the terminal* $\mathsf{Real}^{\uparrow}$ *row* E *with transport, replay, provenance, and local naming carried by* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket density readback`

Proof. Project the density ledger. Its coordinates are exactly the endpoint rows, rational and dyadic containment rows, finite windows, regular rational readbacks, endpoint seals, and structural rows already present in the carrier.

Apply the existing source and compact-uniform package theorems. The finite-net source theorem places L, U, O, Q, D before any mesh consumer, while the compact-uniform source package keeps S, R, E, H, C, P, N attached to the same endpoint-and-containment route.

Thus density readback cannot start from a free mesh. It must pass through $L, U, O, Q, D, S, R, E, H, C, P, N$, so every downstream interval-net or uniform-modulus consumer receives dyadic density data only as a readback of the root endpoint ledger. □

Theorem 1591.67 (Closed bounded interval net-unblock package)**.** *Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$*, finite-cover, nested-window, and uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier must obtain its interval source from Definition 1591.65. No such consumer can bypass the root endpoint-and-containment package*

$$L, U, O, Q, D, S, R, E, H, C, P, N$$

or replace it by a host interval set, an open-cover compactness object, selected subcover, selected point, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the density readback theorem. It identifies $L, U, O, Q, D, S, R, E, H, C, P, N$ as the complete source route for dyadic mesh cells, rational containment tests, StreamName windows, RegSeqRat readbacks, and Real endpoint seals.

Now compare the downstream consumers. Closed bounded interval nets, finite-cover windows, nested-window compactness, and uniform-modulus handoffs all use the same endpoint rows, containment rows, windows, readbacks, and seals before adding their own consumer rows.

The non-bypass claim follows from the carrier inventory. Since the root density ledger contains no host interval set, open-cover compactness object, selected subcover, selected point, quotient equality, trichotomy decision, choice selector, or ambient completeness datum, no downstream consumer can obtain one without leaving the displayed $\mathsf{ClosedBoundedInterval}^{\uparrow}$ package. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket net unblock package`

Theorem 1591.68 (Closed bounded interval source obligation package)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the source-obligation surface for downstream L10 consumers is exactly the endpoint route* L, U, O*, the rational and dyadic containment rows* Q, D*, the finite observation and regular readback rows* S, R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N*. This package includes endpoint containment, endpoint order, finite-net source admission, and Real-seal handoff before any interval-net, nested-window, or completion consumer may read the interval.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source obligation package`

Proof. Project the carrier rows. The endpoint-containment and endpoint-order results already identify L, U, O, Q as the only accepted lower-bound, upper-bound, order, and rational containment surface.

Apply the finite-net source and net-unblock package theorems. They force every mesh, finite-window, and downstream interval-net read through D, S, R before the consumer adds its own rows.

The Real-seal handoff keeps E after the displayed endpoint and readback rows, while H, C, P, N only transport, replay, source, and name that same package. Hence the listed rows are the complete source-obligation surface. □

Theorem 1591.69 (Closed bounded interval source cover boundary)**.** *Every finite-cover, compactness, interval-net, nested-window, and uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *packet must enter through the source-obligation package*

$$L, U, O, Q, D, S, R, E, H, C, P, N.$$

Such a consumer may refine the displayed finite source by its own cover or net rows, but it cannot replace the source by a host interval set, open-cover compactness principle, selected subcover, selected point, quotient real equality, trichotomy, countable-choice schedule, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source cover boundary`

Proof. Start from the finite-net source theorem and the net-unblock package. Together they make the density ledger the only admitted source for dyadic cells, rational containment tests, finite observation windows, regular readbacks, and endpoint seals.

Finite-cover and compactness consumers add rows after that source has been displayed. They may inspect D, S, R, E through the same replay row C, but they do not receive a separate open-cover object or a selector coordinate.

The exclusion follows from the carrier inventory and the endpoint-containment boundary. Since I contains only $L, U, O, Q, D, S, R, E, H, C, P, N$, every accepted cover boundary remains a finite BEDC source route rather than host compactness data. □

Theorem 1591.70 (Closed bounded interval source readback stability)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback used by a downstream interval consumer is stable under the displayed componentwise* hsame *row* H *and the displayed* Cont *replay row* C*: transporting the source rows by* H *and replaying them by* C *preserves the same finite endpoint, containment, dyadic, StreamName, RegSeqRat, and Real-seal route.*

Proof. Read the RegSeqRat and Real handoff rows through the density ledger. The rows S, R, E are reached only after the endpoint route L, U, O, containment rows Q, D, and finite source windows have been displayed.

Use the componentwise transport row H. It preserves lower and upper endpoint rows, order, containment ledgers, dyadic cells, StreamName windows, RegSeqRat readbacks, and the terminal Real seal as components of the same packet.

Use the replay row C for downstream consumers. Since C replays the same displayed route and P, N only source and name it, the readback cannot drift into quotient real equality, selected representatives, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket source readback stability`

Theorem 1591.71 (Closed bounded interval root consumer obligations)**.** *Every net, uniform-modulus, located-cover, nested-window, or compactness-facing consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier must first expose the same lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName window, RegSeqRat readback, Real seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The consumer may add its own downstream rows only after this root package has been displayed; it cannot replace the root by quotient real equality, trichotomy, selected points, selected subcovers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the source-obligation package. It identifies $L, U, O, Q, D, S, R, E, H, C, P, N$ as the finite endpoint, containment, window, readback, seal, transport, replay, provenance, and local naming surface for downstream interval reads.

Apply the source cover boundary. It forces cover, compactness, interval-net, nested-window, and uniform-modulus consumers to enter through that source package before adding their own cover or net rows.

Finally use source readback stability. The RegSeqRat and Real rows remain attached to the same componentwise hsame transport and Cont replay route, so no consumer can detach the root package into quotient equality, trichotomy, selected interval points, selected subcovers, or an ambient completeness assertion. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root consumer obligations`

Theorem 1591.72 (Closed bounded interval root net readiness)**.** *Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier must receive the endpoint rows* L, U, O*, the containment ledgers* Q, D*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the structural rows* H, C, P, N *before reading mesh centers, coverage rows, or net refinements.*

Proof. Project the carrier of Definition 1591.1 and the classifier of Definition 1591.2. The endpoint rows are L, U, O, containment is recorded by Q, D, finite windows and rational readback by S, R, and the terminal seal by E.

Apply the source nonescape and finite-cover nonescape boundaries already established for the chapter. They keep every interval source read inside the displayed carrier rows and refuse host interval sets, selected points, quotient real equality, trichotomy, and ambient completeness.

The finite-net consumer can therefore add mesh centers and coverage rows only after the source package has been displayed. The replay and provenance rows H, C, P, N carry exactly that source into the consumer, so net readiness is the endpoint-and-containment package rather than an independent mesh object. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket root net readiness`

Theorem 1591.73 (Closed bounded interval carrier-classifier scope)**.** *Let* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier. Every downstream read of the lower endpoint, upper endpoint, endpoint-order row, dyadic containment ledger, StreamName window, RegSeqRat readback, RealUp seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows routes through the carrier of Definition 1591.1 and the classifier of Definition 1591.2. No consumer may replace that pair by quotient real equality, trichotomy, selected interval points, selected subcovers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 1591.3. It identifies the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, window, readback, seal, transport, replay, provenance, and local naming rows as the local obligation surface of the accepted packet.

Apply Definition 1591.44. The endpoint rows and their order are preserved only by the displayed componentwise transport and classifier comparison, so the lower and upper endpoints cannot be reclassified through host real equality or trichotomy.

Use Definition 1591.45 and Definition 1591.46. Net, cover, and nested compactness consumers receive the same endpoint, containment, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows before adding their own downstream data.

Thus every accepted read is scoped by the carrier/classifier pair of Definition 1591.1 and Definition 1591.2. Since neither definition contains a coordinate for the refused host data, no downstream consumer can add such a coordinate while preserving the displayed route. □

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalUp.ClosedBoundedIntervalPacket carrier classifier scope`

Theorem 1591.74 (Closed bounded interval obligation closure)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the carrier and classifier of Definition 1591.1 and Definition 1591.2, the NameCert obligations of Definition 1591.3, endpoint-order transport of Definition 1591.44, net handoff of Definition 1591.45, nested compactness handoff of Definition 1591.46, source obligation package of Definition 1591.68, and nonescape rows of Definition 1591.36, Definition 1591.41, and Definition 1591.51 exhaust the full finite NameCert obligation surface for closed bounded interval consumers.*

Proof. Project the carrier and classifier. They expose exactly the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows.

Apply the obligation and transport theorems. The NameCert obligation theorem lists the local rows, endpoint-order transport keeps O upstream of containment, and the net and nested compactness handoffs place all downstream interval consumers behind the same displayed package.

Finally apply the source package and nonescape boundaries. They force net, finite-cover, nested-window, located-cover, uniform-modulus, and compactness-facing consumers to remain inside $L, U, O, Q, D, S, R, E, H, C, P, N$, so the listed results close the obligation surface without importing open-cover compactness, selected subcovers, quotient equality, trichotomy, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket obligation closure`

Theorem 1591.75 (ClosedBoundedInterval scoped kernel envelope)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the carrier, classifier, endpoint containment, dyadic-net, located-cover, nested-compactness, uniform-modulus, and consumer non-escape rows displayed in this chapter are all scoped over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No interval-net, nested-window, located-cover, uniform-modulus, or compactness-facing consumer can read beyond that kernel envelope while preserving the displayed carrier route.*

Proof. Start from Definition 1591.74. It already collects the carrier, classifier, endpoint-order transport, net handoff, nested compactness handoff, source obligation package, and nonescape rows as the finite obligation surface.

The source rows in that package are exactly the lower and upper $\mathsf{Real}^{\uparrow}$ endpoint rows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{DyadicRatCore}^{\uparrow}$ containment ledger, and the structural hsame, Cont, Pkg, and NameCert rows. The net, located-cover, nested-window, uniform-modulus, and compactness-facing consumers enter only through the handoff and nonescape theorems named there.

Thus every consumer route remains inside the displayed kernel envelope. Since no theorem in the package adds quotient real equality, selected interval points, trichotomy, open-cover compactness, or ambient completeness, no accepted consumer can expose those coordinates. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket scoped kernel envelope`

Theorem 1591.76 (Closed bounded interval mature consumer surface)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the public consumer surface of Definition 1591.77 and the public bridge boundary of Definition 1591.78 determine the mature* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *consumer route. Every interval-facing read must pass through the displayed endpoint, containment, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local NameCert rows before it can serve a net, located-cover, nested-window, uniform-modulus, or compactness-facing consumer.*

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPacket mature consumer surface`

Proof. Start from the public consumer surface. It identifies the available public rows as exactly $L, U, O, Q, D, S, R, E, H, C, P, N$ and records that no consumer receives open-cover compactness, a selected subcover, quotient real equality, trichotomy, choice, a selected point, or ambient completeness.

Then apply the public bridge boundary. It forces finite-cover, located-cover, uniform-modulus, and compactness-facing reads to factor through the same endpoint, containment, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and NameCert rows.

Thus the mature route exposes no data beyond the displayed finite packet. The consumer may reuse the public surface and its bridge boundary, but it cannot bypass either one or export a completeness-facing coordinate not already present in the carrier. □

Theorem 1591.77 (Closed bounded interval public consumer surface)**.** *For every accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, the carrier of Definition 1591.1, classifier of Definition 1591.2, NameCert obligations of Definition 1591.3, root source ledger of Definition 1591.68, source nonescape rows of Definition 1591.36, Definition 1591.41, and Definition 1591.51, and scoped kernel envelope of Definition 1591.75 form the public* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *consumer surface. A public consumer may read only the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName, RegSeqRat, Real-seal, hsame, Cont, Pkg, and NameCert rows displayed by the packet, and receives no open-cover compactness, selected subcover, quotient real equality, trichotomy, choice, selected interval point, or ambient completeness coordinate.*

Proof. Project the accepted packet through the carrier and classifier. The public rows are exactly $L, U, O, Q, D, S, R, E, H, C, P, N$, with endpoint order and containment read through the same displayed transport and replay routes.

Apply the NameCert obligation theorem and the root source ledger. They identify the endpoint, containment, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows before any consumer-facing route is exposed.

Use the source nonescape and consumer nonescape rows to block external coordinates. Net, finite-cover, nested-window, located-cover, uniform-modulus, and compactness-facing consumers remain inside the displayed finite packet.

Finally use the scoped kernel envelope. It packages the same rows as the public object surface, so the public consumer receives the named finite carrier and none of the refused open-cover, subcover-selection, quotient-equality, trichotomy, choice, point-selection, or completeness data. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPublicConsumerSurface`

Theorem 1591.78 (Closed bounded interval public bridge boundary)**.** *For every accepted* ClosedBoundedInterval$^{\uparrow}$ *carrier* $I = (L, U, O, Q, D, S, R, E, H, C, P, N)$*, any finite-cover, located-cover, uniform-modulus, or compactness-facing consumer of* I *must factor through the displayed lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName, RegSeqRat, Real-seal, hsame, Cont, Pkg, NameCert rows and the public consumer surface of Definition 1591.77 before it can read interval data. The bridge boundary exposes no open-cover compactness theorem, selected subcover data, quotient real equality, trichotomy, choice, selected interval point, or ambient completeness coordinate.*

Proof. Project the public consumer surface theorem. It identifies the public rows as exactly $L, U, O, Q, D, S, R, E, H, C, P, N$ and records the nonescape clauses that block external compactness and choice data.

Route each consumer class through those rows. A finite-cover consumer reads only the displayed endpoint and containment rows; a located-cover consumer also passes through S, R, E before reading a located interval verdict; a uniform-modulus consumer must replay the same Real-sealed route through H, C; and a compactness-facing consumer reaches only the package and NameCert boundary P, N.

The stated refusals follow because none of the displayed rows stores open-cover compactness, a selected subcover, quotient real equality, trichotomy, choice, a selected point, or ambient completeness. The bridge therefore has no coordinate from which such data can be exported. □

Lean4 ✓ `BEDC.Derived.ClosedboundedintervalUp.ClosedBoundedIntervalPublicBridgeBoundary`

Closure status for NameCert$_{\text{ClosedBoundedInterval}^{\uparrow}}$**.**

Constructive story: A ClosedBoundedInterval$^{\uparrow}$ packet is generated from lower and upper located endpoint rows, an endpoint-order row, rational and dyadic containment ledgers, StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosedBoundedInterval}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1591.76 and Definition 1591.77 bind the carrier, classifier, NameCert obligations, root source ledger, source nonescape rows, scoped kernel envelope, public consumer surface, and bridge boundary as the mature ClosedBoundedInterval$^{\uparrow}$ consumer surface over Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{ClosedBoundedInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient real equality, trichotomy, selected points of the interval, or ambient Real completeness.

Upgrade path: Verification closure requires Lean verification of the public consumer surface for BEDC.Derived.ClosedBoundedIntervalUp.

Chapter 1592

A Concrete Naming Certificate for $\mathsf{CauchySubnet}^{\uparrow}$

Following Bishop–Bridges and Kelley filter terminology, the $\mathsf{CauchySubnet}^{\uparrow}$ packet records the finite restriction surface by which a displayed Cauchy-filter source is read along a cofinal subnet before a filter-limit handoff or completion-facing consumer may inspect the route. It sits after the $\mathsf{CauchyFilter}^{\uparrow}$ surface of chapter 333, the $\mathsf{CauchyFilterLimit}^{\uparrow}$ route of chapter 1558, and the $\mathsf{MetricCompletion}^{\uparrow}$ boundary of chapter 1506; it reads finite windows through $\mathsf{StreamName}^{\uparrow}$ in chapter 126, regular rational rows through $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, dyadic tolerances through $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291, and terminal seals through $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet names cofinal subnet restriction as BEDC data, not as quotient filter equality, selected host subnets, countable choice, host real equality, or ambient completeness.

Definition 1592.1 (Cauchy subnet carrier)**.** *A* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, J, W, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row whose neighborhoods are to be restricted.*
2. *J is the cofinal subnet-index ledger. It records the finite dominance checks showing that every accepted request to the subnet is backed by a displayed request to the source filter.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected after F and J have been replayed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the restricted windows and regular readback.*
6. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff row available only after the source filter, cofinal ledger, windows, readback, and tolerance rows have been displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *seal row reached by completion-facing consumers through the same handoff route.*
8. *H records componentwise* hsame *transport for the source filter, cofinal ledger, windows, readback, tolerance ledger, limit handoff, seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a filter request through F, J, W, R, D, L, E in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected host subnets, countable choice, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetTasteGate single carrier alignment`

Theorem 1592.2 (Cauchy subnet NameCert obligations)**.** *For every accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier* $S = (F, J, W, R, D, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source Cauchy filter, cofinal subnet-index ledger, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchyFilterLimit handoff, completion or Real seal, transport, continuation, provenance, and local naming rows;*
2. *cofinal restriction coverage, requiring every accepted subnet request to read the ledger J together with the source filter row F before any window row is inspected;*
3. *finite-window and tolerance exactness, requiring W, R, D to be the only finite rows through which the restricted source reaches a regular rational comparison;*
4. *filter-limit handoff, requiring F, J, W, R, D to repack as the handoff row L before any completion-facing or Real-facing seal is available;*
5. *ledger refusal, excluding quotient filters, selected host subnets, countable choice, host real equality, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetNameCertObligations`

Proof. Project the carrier of Definition 1592.1. The projection exposes exactly the source Cauchy filter row F, cofinal subnet-index ledger J, StreamName window W, RegSeqRat readback R, DyadicRatCore tolerance ledger D, filter-limit handoff L, seal row E, and structural rows H, C, P, N.

The continuation row C gives the only replay route. A subnet request first reads the source filter row and the cofinal ledger, then reaches the finite windows, regular readback, and dyadic tolerance ledger.

The handoff row L is downstream of those five displayed rows, and the completion or Real seal E is downstream of L. Since the carrier contains no quotient-filter coordinate, selected host subnet, countable-choice selector, host real equality row, or ambient-completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 1592.3 (Cauchy subnet filter-limit handoff)**.** *Let $S = (F, J, W, R, D, L, E, H, C, P, N)$ be an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier. Every completion-facing or Real-facing consumer of the restricted Cauchy source factors through the finite route*

$$F, J, W, R, D \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff only after the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, cofinal subnet-index ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger have been displayed. The route exports no quotient filters, selected host subnets, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnet filter limit handoff`

Proof. Begin with a completion-facing or Real-facing consumer. Carrier admission provides the handoff row L and seal row E, but only as rows downstream of F, J, W, R, D.

Replay the continuation row C. The request first consumes the source Cauchy filter row and the cofinal subnet-index ledger, then inspects the finite window row, the regular rational readback, and the dyadic tolerance ledger.

Only after that displayed route has been replayed can the packet expose L, and only after L can it expose E. Componentwise transport through H and provenance through P, N preserve the same dependency order, so the handoff is a finite BEDC route rather than quotient filter equality, selected host subnet choice, countable choice, host real equality, or ambient completeness. □

Theorem 1592.4 (CauchySubnet RegSeqRat tail handoff)**.** *Let $S = (F, J, W, R, D, L, E, H, C, P, N)$ be an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier. Every subnet tail read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must pass through the displayed route*

$$F, J, W, R, D, L, E, H, C, P, N.$$

In particular, the $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff route through L, E are all visible before the* $\mathsf{Real}^{\uparrow}$ *seal is public. The packet exports no direct tail-to-real read, no hidden regular sequence, no unrecorded dyadic tolerance, no ambient completion equality, and no host real quotient.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetRegSeqRatTailHandoff`

Proof. Project the carrier of Definition 1592.1. Its endpoint row E can be a $\mathsf{Real}^{\uparrow}$ seal only as part of the displayed Cauchy subnet packet, and the continuation row C records the route into that seal.

Read the route before the endpoint. The source filter row F and cofinal ledger row J expose the accepted tail request, after which the finite window row W supplies the $\mathsf{StreamName}^{\uparrow}$ display inspected by the tail.

Next the regular and dyadic rows are forced to appear. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the displayed windows, and D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger used by the same readback.

Finally apply Definition 1592.3. The handoff theorem places L and then E downstream of F, J, W, R, D, so a public $\mathsf{Real}^{\uparrow}$ seal cannot be reached before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ route has been displayed. □

Theorem 1592.5 (Cauchy subnet cofinal ledger coverage)**.** *Let $S = (F, J, W, R, D, L, E, H, C, P, N)$ be an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier. Every accepted subnet request admitted by the cofinal ledger row J has a displayed source-filter request in F. Every completion-facing read must pass through*

$$F, J, W, R, D, L, E, H, C, P, N$$

before it exposes a $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal. Thus cofinal restriction is a finite ledger coverage route, not a quotient-filter equality, selected host subnet, countable-choice selector, host real equality, or ambient completeness assertion.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnet cofinal ledger coverage`
↪ variant `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier cofinal ledger coverage`

Proof. Project the carrier of Definition 1592.1. The row J is specified as the cofinal subnet-index ledger, and its accepted requests are backed by displayed source requests to the filter row F.

Use Definition 1592.2 for row exhaustion. After F and J, every admissible read proceeds through the finite-window row W, regular rational readback R, dyadic tolerance ledger D, handoff row L, seal row E, and structural rows H, C, P, N.

Finally apply Definition 1592.3. It identifies the only endpoint route to $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$, so no completion-facing consumer can bypass the cofinal ledger coverage. □

Theorem 1592.6 (CauchySubnet cofinal reindexing route)**.** *Let $S = (F, J, W, R, D, L, E, H, C, P, N)$ be an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier. Any accepted cofinal subnet request whose ledger J reindexes* $\mathsf{StreamName}^{\uparrow}$ *windows preserves the route*

$$F, J, W, R, D, L, E, H, C, P, N.$$

Consequently the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *seal access remain downstream of the cofinal ledger. The route exports no quotient filter equality, selected subnet, countable-choice schedule, host real equality, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCofinalReindexingRoute`

Proof. Start from the local carrier and obligation surface. Definition 1592.2 requires every accepted subnet request to read the source filter row F and cofinal ledger row J before any finite StreamName window is inspected.

Use Definition 1592.5. It gives a displayed source-filter request for each accepted cofinal ledger request, so reindexing the StreamName windows cannot detach W from the preceding rows F, J.

Finally apply Definition 1592.3. The handoff theorem fixes the downstream order through $W, R, D, L, E, H, C, P, N$, so RegSeqRat readback, DyadicRatCore tolerance, MetricCompletion access, and Real seal access remain behind the cofinal ledger. □

Theorem 1592.7 (Cauchy subnet obligation-closure package)**.** *The carrier definition of Definition 1592.1, together with Definition 1592.2, Definition 1592.3, and Definition 1592.5, closes the* $\mathsf{CauchySubnet}^{\uparrow}$ *obligation surface for carrier admission, cofinal coverage, finite-window exactness, tolerance exactness, filter-limit handoff, and ledger refusal. In particular, every accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *consumer read is forced through*

$$F, J, W, R, D, L, E, H, C, P, N$$

before it may expose a $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *endpoint, and the packet has no remaining local obligation coordinate for quotient filters, selected host subnets, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnet obligation closure package`

Proof. Carrier admission is exactly the row projection in Definition 1592.1. The source filter, cofinal ledger, finite windows, rational readback, dyadic tolerance ledger, limit handoff, completion or Real seal, transport, replay, provenance, and local naming rows are all visible there.

Apply Definition 1592.2 to exhaust the local NameCert surface. Its obligation list accounts for carrier admission, cofinal restriction coverage, finite-window and tolerance exactness, the downstream filter-limit handoff, and refusal of the excluded host-level data.

The two route theorems then identify the order of consumption. Definition 1592.3 forces every endpoint through $F, J, W, R, D, L, E, H, C, P, N$, while Definition 1592.5 prevents a completion-facing read from bypassing the cofinal ledger. These clauses leave no further local obligation row for the $\mathsf{CauchySubnet}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier obligation closure package`

Theorem 1592.8 (Cauchy subnet scoped route)**.** *Every accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *read is scoped exactly by the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *source row,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff row,* $\mathsf{MetricCompletion}^{\uparrow}$ *endpoint row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal row, the surrounding* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ *row. A consumer that omits any one of these rows receives only a partial subnet restriction and cannot expose a completion-facing endpoint for the accepted packet.*

Proof. Start with the local obligation inventory. Definition 1592.2 lists carrier admission, cofinal restriction coverage, finite-window and tolerance exactness, filter-limit handoff, and ledger refusal as the only local rows.

Next read the endpoint route. Definition 1592.3 forces every completion-facing or Real-facing consumer through $F, J, W, R, D, L, E, H, C, P, N$, while Definition 1592.5 prevents the cofinal ledger J from being bypassed before the finite windows are inspected.

Finally apply Definition 1592.7. It packages the carrier, handoff, cofinal coverage, and refusal clauses into the same displayed packet, so the exact scope of an accepted read is precisely the listed CauchyFilter, CauchyFilterLimit, MetricCompletion, StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and NameCert surface. □

Theorem 1592.9 (CauchySubnet scoped kernel route)**.** *Every accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *obligation in this chapter depends only on the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff,* $\mathsf{MetricCompletion}^{\uparrow}$ *endpoint row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ *rows. Within that kernel scope, cofinal ledger coverage, finite-window exactness, tolerance exactness, filter-limit handoff, and endpoint nonescape are exactly the obligations named by the carrier and obligation package; outside that scope the packet exports no quotient filters, selected host subnets, countable choice, host real equality, or ambient completeness.*

Proof. Use Definition 1592.2 to list the local obligation rows. They name carrier admission, cofinal restriction coverage, finite-window and tolerance exactness, filter-limit handoff, and ledger refusal, all inside the displayed carrier surface.

Use Definition 1592.3 and Definition 1592.5 to fix the consumption order. Every completion-facing read must pass through the source filter, cofinal ledger, StreamName window, RegSeqRat readback, DyadicRatCore tolerance row, CauchyFilterLimit handoff, and MetricCompletion or Real endpoint.

Finally apply Definition 1592.7. The same package includes componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row, so the obligation surface is scoped to the named kernel rows and does not acquire any excluded host-level coordinate. □

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier scoped kernel route`

Theorem 1592.10 (CauchySubnet public certificate route)**.** *Every public completion-facing consumer of an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *packet reads the cofinal ledger through* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it can expose the local* $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ *row. The public route exports no quotient filters, selected host subnets, countable choice, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier public certificate route`

Proof. First use Definition 1592.2. It identifies the local rows that must be displayed before the packet can be consumed: source Cauchy filter, cofinal subnet-index ledger, finite windows, regular readback, dyadic tolerance, handoff, endpoint seal, transport, replay, provenance, and local naming.

Next apply Definition 1592.3 and Definition 1592.5. These results force every completion-facing read through the cofinal ledger and then through the finite-window, readback, tolerance, handoff, and endpoint rows.

Finally apply Definition 1592.9. It pins the route to the displayed kernel rows and excludes quotient filters, selected host subnets, countable choice, host real equality, and ambient completeness before the local $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$ row is public. □

Theorem 1592.11 (Cauchy subnet endpoint nonescape)**.** *Every* $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *endpoint read of an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *packet factors through the displayed rows*

$$F, J, W, R, D, L, E, H, C, P, N.$$

The endpoint is therefore reached only by the source Cauchy filter row F*, cofinal subnet ledger* J*, finite-window row* W*, regular-rational readback* R*, dyadic tolerance ledger* D*, filter-limit handoff* L*, endpoint seal* E*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming row* N*.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier endpoint nonescape`

Proof. Start with the filter-limit handoff of Definition 1592.3. It forces every completion-facing or Real-facing consumer through the displayed route from F to E before any $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ endpoint can be exposed.

Next use the cofinal ledger coverage of Definition 1592.5. This prevents a consumer from bypassing J and fixes the finite-window, readback, and tolerance rows W, R, D as part of the same endpoint route.

Finally apply the scoped kernel route of Definition 1592.9 and the public certificate route of Definition 1592.10. They add the structural rows H, C, P, N and exclude quotient filters, selected host subnets, countable choice, host real equality, and ambient completeness from every accepted endpoint read. □

Theorem 1592.12 (Cauchy subnet public cofinal-filter handoff)**.** *Let* $S = (F, J, W, R, D, L, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchySubnet}^{\uparrow}$ *carrier. Before any public completion-facing consumer reads* S *as a cofinal-filter handoff, the packet exposes the cofinal subnet map ledger* J*, the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F*, the* $\mathsf{CauchyNet}^{\uparrow}$ *source route when the consumer enters through a net presentation, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The handoff is therefore a public finite route through the displayed subnet, filter, net, window, readback, tolerance, and seal rows, not a selected subnet choice, quotient-net equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchySubnetUp.CauchySubnetCarrier public cofinal filter handoff`

Proof. Project the carrier of Definition 1592.1. The rows $F, J, W, R, D, L, E, H, C, P, N$ expose the source Cauchy filter, cofinal ledger, finite windows, regular rational readback, dyadic tolerance ledger, filter-limit handoff, endpoint seal, transport, replay, provenance, and local naming.

Use the Cauchy subnet obligation surface of Definition 1592.2. It forces every accepted subnet request to read the source filter row F and cofinal ledger J before any StreamName window or RegSeqRat readback is public.

Now apply the cofinal ledger coverage and endpoint nonescape routes. By Definition 1592.5, every completion-facing read passes through $F, J, W, R, D, L, E, H, C, P, N$; by Definition 1592.11, the endpoint remains downstream of the same rows.

A $\mathsf{CauchyNet}^{\uparrow}$ presentation can therefore only enter as a source-route view of the displayed Cauchy data before the same cofinal ledger is consumed. Since the carrier contains no selected subnet, quotient-net equality, or ambient completeness coordinate, the public handoff exports only the finite cofinal-filter route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1592.1, Definition 1592.2, Definition 1592.3, Definition 1592.5, Definition 1592.7, Definition 1592.9, Definition 1592.10, and Definition 1592.11 close the public-facing finite cofinal subnet route between CauchyFilter sources, StreamName and RegSeqRat readback, DyadicRatCore tolerance rows, CauchyFilterLimit handoff, MetricCompletion or Real consumers, endpoint nonescape, and the hsame, Cont, Pkg, and NameCert support rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySubnet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected host subnets, countable choice, host real equality, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, cofinal restriction coverage, finite-window exactness, tolerance exactness, filter-limit handoff, and ledger refusal proofs for the same packet.

Constructive story: A $\mathsf{CauchySubnet}^{\uparrow}$ packet is generated from a source CauchyFilter row, a cofinal subnet-index ledger, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyFilterLimit handoff, a MetricCompletion or Real seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1593

A Concrete Naming Certificate for $\mathsf{GershgorinDisk}^{\uparrow}$

Following Gershgorin's circle theorem and standard finite-dimensional linear algebra, the $\mathsf{GershgorinDisk}^{\uparrow}$ packet records the finite row-sum surface used to bound spectral locations for a displayed square matrix. It consumes matrix entries from chapter 49, linear-map interpretation from chapter 48, vector rows from chapter 47, and scalar comparison data from the real-facing boundary of chapter 38. The packet names the disk-bound route only; it does not assert eigenvalue existence over arbitrary fields, algebraic closure, spectral radius equality, analytic completeness, or a host-library spectrum predicate.

Definition 1593.1 (Gershgorin disk carrier)**.** *A* $\mathsf{GershgorinDisk}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, R, D, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Mat}^{\uparrow}$ *square-matrix row, including the finite row index, diagonal entries, and off-diagonal entry slots whose scalar boundary is read through the real-facing comparison surface.*
2. *R is the row-sum radius ledger, carrying for each displayed row the finite sum of absolute off-diagonal magnitudes used as the disk radius.*
3. *D is the disk-center and radius classifier row, naming the center a_{ii} and radius r_i for every displayed row i.*
4. *B is the displayed bound route saying that any spectral witness already supplied by a downstream linear-map or eigen-facing packet must be compared against one of the finitely displayed disks.*
5. *H stores componentwise* hsame *transport for the matrix row, radius ledger, disk classifier, bound route, continuation, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from matrix entries to row-sum radii, from radii to disk classifiers, and from the disk classifiers to the bound route.*
7. *P is the* Pkg *provenance over* $\mathsf{GershgorinDisk}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{GershgorinDisk}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, R, D, B, H, C, P, N. It has no coordinate for producing eigenvalues, choosing algebraic closures, asserting spectral radius equality, completing scalar fields, or importing analytic spectral theory.

Theorem 1593.2 (Gershgorin disk NameCert obligations)**.** *For every accepted* $\mathsf{GershgorinDisk}^{\uparrow}$ *carrier* $G = (M, R, D, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GershgorinDisk}^{\uparrow}}$ *surface consists exactly of carrier admission for the eight displayed rows, matrix entry access through* M*, finite row-sum radius accounting through* R*, disk-center and radius classification through* D*, the displayed spectral-bound route through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.GershgorinDiskUp.GershgorinDiskUpTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1593.1. The only matrix data available to the classifier are the square-matrix row M, its finite row index, and the displayed diagonal and off-diagonal entry slots read from chapter 49.

The row-sum ledger R and disk classifier D are then read from those displayed entries. They name centers and radii by finite scalar comparisons against the real-facing boundary of chapter 38; no eigenvalue-producing field extension or analytic completeness row is introduced.

Finally the bound route B and the structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate beyond these rows, the listed obligations exhaust the local NameCert surface. □

Theorem 1593.3 (Gershgorin disk row-sum bound)**.** *Every accepted* $\mathsf{GershgorinDisk}^{\uparrow}$ *carrier exposes its displayed bound through the finite route*

$$M \longmapsto R \longmapsto D \longmapsto B$$

together with H, C, P, N*. For each displayed row* i*, the packet reads the center from the diagonal entry of* M*, reads the radius from the finite off-diagonal row-sum ledger* R*, and records the disk classifier in* D *without asserting that the matrix has an eigenvalue over the ambient scalar boundary.*

Proof. Begin with the matrix row M. Its diagonal and off-diagonal slots are finite displayed entry rows, so the row-sum radius ledger R can only add the off-diagonal magnitude rows already exposed by the packet.

Next read the classifier D. It pairs the diagonal entry of row i with the radius recorded for the same row, producing the disk surface used by the bound route.

The route B is downstream-facing: it can constrain a spectral witness supplied elsewhere, but it cannot manufacture such a witness. Transport and replay through H and C, with provenance and naming through P and N, therefore preserve exactly the finite row-sum bound and no additional spectral principle. □

Theorem 1593.4 (Gershgorin disk matrix handoff)**.** *The* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *consumers may read a* $\mathsf{GershgorinDisk}^{\uparrow}$ *packet only through the matrix row of chapter 49, the linear-map interpretation of chapter 48, the vector-space boundary of chapter 47, and the displayed route*

$$R, D, B, H, C, P, N.$$

Thus the handoff supplies a finite spectral-bound surface for matrix-shaped linear maps while leaving eigenvalue existence, algebraic closure, spectral radius equality, analytic completeness, and host spectral decomposition outside the exported certificate.

Proof. Use Definition 1593.3. It places every displayed disk read on the route from matrix entries through row-sum radii, disk classification, and the bound route.

The matrix-to-linear-map handoff is read through chapter 49, chapter 48, and chapter 47. Those chapters supply the finite rows for entries, linear action, and vector slots; this chapter only adds the row-sum disk ledger on top of those rows.

The structural rows H, C, P, N preserve and name that ledger. Since the carrier lacks any coordinate for generating spectral points, completing scalar fields, or identifying spectral radii, the exported surface is only the finite Gershgorin bound handoff. □

Falsifiable prediction: Every $\mathsf{GershgorinDisk}^{\uparrow}$ spectral-bound read factors through a MatrixUp square-matrix row, a finite row-sum radius ledger, disk-center and radius classifiers, a displayed bound route, transport, continuation, provenance, and local naming.

Independence witness: matrix, linearmap, vecspace, real: $\mathsf{GershgorinDisk}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries matrix entries, finite row-sum radii, disk classifiers, a downstream bound route, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GershgorinDisk}^{\uparrow}}$.

Constructive story: A $\mathsf{GershgorinDisk}^{\uparrow}$ packet is generated from a MatrixUp square-matrix row, finite row-sum radius ledger, disk-center and radius classifier, displayed bound route, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GershgorinDisk}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1593.1, Definition 1593.2, Definition 1593.3, and Definition 1593.4 bind the finite row-sum spectral-bound packet for MatrixUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GershgorinDisk}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, eigenvalue existence over arbitrary fields, algebraic closure, spectral radius equality, analytic completeness, or host spectral decomposition.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, row-sum bound preservation, matrix handoff, and TasteGate field faithfulness for BEDC.Derived.GershgorinDiskUp.

Chapter 1594

A Concrete Naming Certificate for $\mathsf{CofinalSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CofinalSequence}^{\uparrow}$ packet records the constructive tail discipline for a sequence whose visible indices eventually dominate every displayed request. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular-rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It also provides the canonical carrier surface consumed by the specialized cofinal-tail selector of chapter 1263 and the cofinal-window Real-seal route of chapter 1174. The packet names a finite BEDC certificate for tail dominance and selected windows; it does not import countable choice, a host subsequence object, quotient stream equality, selected limits, or ambient completeness.

Definition 1594.1 (Cofinal sequence carrier)**.** *A* $\mathsf{CofinalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, D, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the unary index row, recording the visible natural-number requests and the displayed tail indices serving them.*
2. *D is the monotone cofinal dominance row, saying that whenever a displayed request is raised, the serving tail index is transported to a later displayed index and still dominates the request.*
3. *W is the finite family of selected* $\mathsf{StreamName}^{\uparrow}$ *windows attached to the serving indices in I.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row read from the selected windows in W.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the unary index row, dominance row, selected windows, and regular-rational handoff have been displayed.*
6. *H stores componentwise* hsame *transport for the index, dominance, window, regular handoff, Real seal, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a request to its serving tail index, from the serving index to the selected* $\mathsf{StreamName}^{\uparrow}$ *window, from the window to the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and from the handoff to the* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CofinalStreamTailSelector}^{\uparrow}$, $\mathsf{CofinalWindowRealSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CofinalSequence}^\uparrow}$ *row.*
The classifier compares accepted packets only through $I, D, W, Q, R, H, C, P, N$. It has no coordinate for countable choice, an external subsequence function, quotient equality of streams, a detached limit, an unbounded hidden selector, or ambient completeness.

Theorem 1594.2 (Cofinal sequence NameCert obligations)**.** *For every accepted* $\mathsf{CofinalSequence}^\uparrow$ *carrier $S = (I, D, W, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalSequence}^\uparrow}$ *surface consists exactly of carrier admission for the nine displayed rows, unary index exposure through I, monotone cofinal dominance through D, selected* $\mathsf{StreamName}^\uparrow$ *window exposure through W,* $\mathsf{RegSeqRat}^\uparrow$ *handoff through Q,* $\mathsf{Real}^\uparrow$ *seal ordering through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1594.1. The accepted packet exposes only the unary index row, the dominance row, the selected windows, the regular handoff, the Real seal, and the structural support rows.

Read the rows in dependency order. The index row I names the visible request and serving-tail indices; the dominance row D supplies the monotone cofinal comparison; the window row W selects exactly the finite $\mathsf{StreamName}^\uparrow$ observations allowed at those serving indices.

The remaining rows only continue that route. The regular handoff Q, Real seal R, transports H, continuations C, provenance P, and local name N do not add a hidden selector or host subsequence coordinate, so the listed obligations exhaust the certificate surface. □

Theorem 1594.3 (Cofinal sequence tail dominance)**.** *Every accepted* $\mathsf{CofinalSequence}^\uparrow$ *carrier exposes a finite dominance route*

$$I \longmapsto D \longmapsto W$$

before any $\mathsf{RegSeqRat}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *consumer may read the packet. Thus each displayed request is served by a displayed later index, and any transported read of the sequence factors through the same unary index row, monotone dominance row, selected window family, and structural rows H, C, P, N.*

Proof. Begin with a visible request in I. The dominance row D is the only row that relates the request to a later serving index, so every cofinal-tail read must pass through it.

After the serving index is fixed, the packet may expose only the selected windows in W. These windows are the finite observation surface consumed downstream, and they remain tied to the same request-serving route by H and C.

Since no coordinate of the carrier names an external subsequence function or a hidden unbounded selector, a transported consumer can only receive the displayed route I, D, W, H, C, P, N before any regular or Real-facing handoff. □

Theorem 1594.4 (Cofinal sequence RegSeqRat handoff)**.** *The* $\mathsf{RegSeqRat}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *consumers of an accepted* $\mathsf{CofinalSequence}^\uparrow$ *carrier receive the route*

$$I, D, W, Q, R, H, C, P, N.$$

The handoff first displays a cofinal tail index, then the selected $\mathsf{StreamName}^\uparrow$ *window at that index, then the* $\mathsf{RegSeqRat}^\uparrow$ *readback, and only then the* $\mathsf{Real}^\uparrow$ *seal. Consequently the specialized* $\mathsf{CofinalStreamTailSelector}^\uparrow$ *and* $\mathsf{CofinalWindowRealSeal}^\uparrow$ *chapters can consume a single canonical cofinal-sequence surface without importing host countable choice, quotient stream equality, selected limits, or ambient completeness.*

Proof. Use Definition 1594.3. It forces the downstream read to begin with the unary index row, dominance row, and selected window family.

The regular handoff is then exact: Q reads only the displayed windows in W, and R reads only the regular handoff Q. This is the same StreamName-to-RegSeqRat-to-Real ordering used by chapter 301 and chapter 38.

Finally compare the two cofinal consumers. The selector chapter chapter 1263 reads the tail-service part of the route, while chapter 1174 reads the terminal window-readback-seal part. Both reads are projections of the same displayed packet, so neither consumer receives an additional choice source or detached Real seal. □

Falsifiable prediction: Every $\mathsf{CofinalSequence}^{\uparrow}$ Real-facing read factors through a unary index row, monotone cofinal dominance, selected StreamName windows, RegSeqRat handoff, RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: streamname, regseqrat, real, cofinalstreamtailselector, cofinalwindowrealseal: $\mathsf{CofinalSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries unary tail-index dominance, selected windows, regular readback, Real seal ordering, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CofinalSequence}^{\uparrow}$ packet is generated from a unary index row, monotone cofinal dominance row, selected StreamName windows, RegSeqRat handoff, RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1594.1, Definition 1594.2, Definition 1594.3, and Definition 1594.4 bind the canonical cofinal tail-sequence packet for RegSeqRatUp, RealUp, CofinalStreamTailSelectorUp, and CofinalWindowRealSealUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, countable choice, external subsequence functions, quotient stream equality, selected limits, hidden unbounded selectors, or ambient completeness.

Upgrade path: Lean closure requires carrier admission, NameCert obligations, tail-dominance preservation, RegSeqRat handoff, Real seal ordering, and TasteGate field faithfulness for BEDC.Derived.CofinalSequenceUp.

Chapter 1595

A Concrete Naming Certificate for $\mathsf{LocatedCauchySequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedCauchySequence}^{\uparrow}$ packet records the sequence-level data of a located Cauchy name before any quotient stream, selected limit, countable-choice selector, trichotomy branch, or ambient completeness principle is available. It sits after the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located Cauchy source surface of chapter 340, the located real boundary of **??**, the terminal real-name seal of chapter 38, and the regular located Cauchy real packet of chapter 1454. The chapter packages finite sequence observations and located tail data; it does not assert that the sequence has a host-selected completed value.

Definition 1595.1 (Located Cauchy sequence carrier)**.** *A* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, D, T, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite observation windows of the sequence.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and tolerance ledger used to compare displayed sequence terms.*
4. *T is the located Cauchy tail row, recording the finite threshold at which later displayed terms are accepted inside the requested dyadic tolerance.*
5. *W is the located-window row inherited from chapter 340; it binds each tail request to the corresponding displayed located ball.*
6. *E is the downstream seal row toward* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$*, read only after S, Q, D, T, W have been exposed.*
7. *H records componentwise* hsame *transport for sequence windows, regular readback, dyadic tolerance data, tail rows, located windows, seal rows, continuation rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from each sequence precision request through $S, Q, D, T, W, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchySequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedCauchy}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the ten displayed rows. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, hidden infinite stream inspection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1595.2 (Located Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *carrier* $L = (S, Q, D, T, W, E, H, C, P, N)$*, the seed* $\mathsf{NameCert}_{\mathsf{LocatedCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, finite-window exposure through* S*, regular rational readback through* Q*, dyadic tolerance exposure through* D*, located tail exposure through* T*, located-window compatibility through* W*, downstream seal ordering through* E*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 1595.1. The visible analytic rows are precisely the stream schedule, regular readback, dyadic tolerance ledger, located tail row, located-window row, and downstream seal row.

Read the rows in dependency order. A precision request first reaches the finite windows S, then the regular rational observations Q, then the dyadic endpoint and tolerance ledger D. The located tail row T can be inspected only after those finite observations are fixed, and the located-window row W binds the same tail request to the displayed located ball.

The seal and structural rows add no further analytic object. The row E is downstream of S, Q, D, T, W, while H, C, P, N transport, replay, source, and name that same route. Since no listed row stores quotient stream equality, a selected host limit, countable choice, trichotomy, hidden stream inspection, or ambient completeness, the listed rows exhaust the seed obligation surface. □

Theorem 1595.3 (Located Cauchy sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read of an accepted* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *carrier factors through*

$$S \longmapsto Q \longmapsto D \longmapsto T \longmapsto W \longmapsto (H, C, P, N).$$

Thus the regular rational consumer receives a finite sequence window, the matching regular readback row, the dyadic tolerance ledger, and the located tail witness before any Real-facing seal can be inspected.

Proof. Start from a regular rational consumer request. By Definition 1595.1, the consumer reaches Q only through the finite schedule row S, so the read begins with a displayed sequence window rather than with an unbounded stream.

The dyadic and located rows then fix the tail evidence. The tolerance ledger D records the endpoint and radius data used by the request, while T and W attach the located Cauchy tail and located-window witness to that same finite readback.

The structural rows preserve this route. The rows H, C, P, N transport, replay, package, and name the displayed handoff, and none of them introduces a quotient of streams, a selected limit, a choice function, a trichotomy branch, or an ambient completeness witness. □

Theorem 1595.4 (Located Cauchy sequence Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{LocatedReal}^{\uparrow}$*-facing, or* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *carrier factors through the complete finite route*

$$S, Q, D, T, W, E, H, C, P, N.$$

Consequently a downstream consumer cannot read a terminal seal from the packet without also reading the sequence window, regular rational source, dyadic tolerance ledger, located Cauchy tail witness, located-window row, transport, continuation, provenance, and local NameCert rows.

Proof. Begin with a downstream seal read. The carrier exposes the seal only as E, and E is listed after the finite windows, regular readback, dyadic tolerance row, located tail row, and located-window row in Definition 1595.1.

The route to E is therefore finite. The sequence window S fixes the requested observations, Q records their regular rational readback, D budgets the dyadic comparisons, and T, W provide the located Cauchy and located-window evidence attached to that request.

Finally read the support rows. The rows H, C, P, N keep the same finite route stable under componentwise transport, continuation replay, package provenance, and local naming. Since the packet has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, hidden infinite stream inspection, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no downstream consumer can escape the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchySequence}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a located Cauchy tail row, a located-window witness, a downstream Real-facing seal row, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1595.1, Definition 1595.2, Definition 1595.3, and Definition 1595.4 seed the sequence-level located Cauchy packet between StreamName, RegSeqRat, DyadicRatCore, LocatedCauchy, LocatedReal, Real, and RegularLocatedCauchyReal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, trichotomy, hidden infinite stream inspection, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the NameCert obligation surface, the RegSeqRat handoff, and the Real seal nonescape theorem for BEDC.Derived.LocatedCauchySequenceUp.

Chapter 1596

A Concrete Naming Certificate for $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$

Following the Bishop–Bridges finite-cover reading, the $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ packet records the finite dyadic cover data used by interval compactness arguments without asserting a general open-cover compactness principle. It consumes the closed interval base of chapter 1591, the finite net surface of chapter 1250, the dyadic arithmetic rows of chapter 291, the finite name windows of chapter 126, the rational readback rows of chapter 301, the Real seal of chapter 38, and the downstream finite-net consumer route of chapter 388. The packet is a displayed certificate for one closed bounded interval row, one mesh scale, rational and dyadic cells, displayed centers, finite coverage rows, StreamName windows, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming.

Definition 1596.1 (Interval finite cover certificate carrier)**.** *An* $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, M, C, D, Z, F, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row, including its endpoint-order and containment ledgers;*
2. *M is the unary mesh scale naming the dyadic precision at which the cover is read;*
3. *C is the finite rational cell ledger attached to I at scale M;*
4. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell ledger refining the rational cells of C;*
5. *Z is the displayed center row, with one rational or dyadic center for each accepted cell;*
6. *F is the finite coverage row saying that every displayed interval containment read enters one of the cells in C together with its dyadic refinement in D;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to read the endpoints, centers, and cell boundaries at the mesh scale;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows;*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, M, C, D, Z, F, S, R have been displayed;*

10. *H records componentwise* hsame *transport for the interval row, mesh row, rational cells, dyadic cells, centers, coverage row, windows, readbacks, and Real seal;*
11. *K records displayed* Cont *replay from the closed interval row through mesh selection, cell coverage, finite windows, rational readback, and RealUp sealing;*
12. *P is the* Pkg *provenance over* $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}}$ *row.*

The carrier exports no arbitrary open cover, selected subcover, quotient real equality, trichotomy row, hidden point selector, or ambient Real completeness principle. Its public data are exactly the finite interval, mesh, cell, center, coverage, readback, seal, transport, replay, provenance, and local naming rows.

Definition 1596.2 (Interval finite cover certificate classifier)**.** *Two accepted* $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ *carriers*

$$J = (I, M, C, D, Z, F, S, R, E, H, K, P, N)$$

and

$$J' = (I', M', C', D', Z', F', S', R', E', H', K', P', N')$$

are classified together exactly when the displayed hsame *rows transport I to I', M to M', C to C', D to D', Z to Z', F to F', S to S', R to R', and E to E', and the displayed* Cont *rows replay the same interval-to-mesh-to-cover-to-seal route on both packets. The classifier has no coordinate for equality of ambient subsets of* $\mathsf{Real}^{\uparrow}$*, general cover refinement, choice of a finite subcover from an arbitrary family, trichotomy, or completed real-line compactness.*

Theorem 1596.3 (Interval finite cover certificate NameCert obligations)**.** *For the carrier of Definition 1596.1, the local* $\mathsf{NameCert}_{\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission for $J = (I, M, C, D, Z, F, S, R, E, H, K, P, N)$, closed bounded interval admission through I, unary mesh admission through M, rational cell admission through C, dyadic cell admission through D, center-row admission through Z, finite coverage admission through F, StreamName and RegSeqRat readback through S and R, RealUp sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier. Its rows display the closed interval, mesh scale, rational cells, dyadic cells, centers, finite coverage, windows, readbacks, Real seal, transport, replay, provenance, and local name as one finite packet.

Read the analytic rows before the seal. The interval row I, mesh row M, cell rows C, D, center row Z, and coverage row F are all exposed before the $\mathsf{Real}^{\uparrow}$ seal E, while S and R record the finite name windows and rational readbacks used by the same cells.

The remaining rows close the naming surface. The transport row H, replay row K, provenance row P, and local name N preserve exactly this finite route, so there is no additional NameCert obligation outside the listed rows. □

Theorem 1596.4 (Interval finite cover certificate nonescape)**.** *Every accepted* $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ *consumer reads interval coverage through the finite route*

$$I, M, C, D, Z, F, S, R, E, H, K, P, N.$$

The read receives a closed bounded interval row, a unary mesh scale, rational cells, dyadic refinements, displayed centers, a finite coverage ledger, StreamName windows, RegSeqRat readback, and

a RealUp seal. It receives no general open-cover compactness theorem, arbitrary subcover choice, quotient real equality, order trichotomy, or ambient Real completeness row.

Proof. Start with the displayed route. The carrier admits coverage only through the rows I, M, C, D, Z, F, so an interval read must pass through the accepted closed interval, the mesh scale, the rational and dyadic cells, the centers, and the finite coverage row.

The name-facing rows do not add escape coordinates. The StreamName windows, RegSeqRat readbacks, and RealUp seal are downstream of the finite cover rows, and the structural rows H, K, P, N only transport, replay, source, and name the same packet.

Since the carrier and classifier contain no coordinate for an arbitrary open cover, selected subcover, quotient real equality, trichotomy, or ambient completeness principle, a consumer cannot extract any such datum from an accepted packet. □

Theorem 1596.5 (Interval finite cover certificate compact-uniform handoff)**.** *Every* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ *packet receives the finite handoff*

$$I, M, C, D, Z, F, S, R, E, H, K, P, N$$

as a dyadic interval cover certificate. The handoff may be read as closed-bounded interval data from I, mesh and cell data from M, C, D, Z, F, finite name data from S, R, and a Real-facing seal from E, all transported and sourced by H, K, P, N. It does not assert Heine–Borel open-cover compactness, choose a subcover from an arbitrary family, identify quotient real names, decide trichotomy, or supply ambient Real completeness.

Proof. Use the NameCert obligations. They identify the accepted packet with the rows $I, M, C, D, Z, F, S, R, E, H,$ so a downstream consumer receives the same finite route rather than an ambient interval object.

Use the nonescape theorem. It prevents the handoff from being reinterpreted as a general open-cover compactness row, a selected subcover, quotient real equality, trichotomy, or Real completeness.

The compact-uniform-continuity consumer therefore sees exactly a finite dyadic cover certificate: interval base rows, mesh and cell rows, centers, coverage, finite name windows, rational readback, a Real-facing seal, transport, replay, provenance, and local naming. □

Falsifiable prediction: Every compact-uniform-continuity read of an IntervalFiniteCoverCertificateUp packet factors through the displayed finite dyadic cover route and not through an open-cover compactness row.

Independence witness: closedboundedinterval, closedboundedintervalnet, dyadicratcore, streamname, regseqrat, real, compactuniformcontinuity: $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a closed bounded interval row, unary mesh scale, rational and dyadic cells, displayed centers, finite coverage rows, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$ packet is generated from a closed bounded interval row, unary mesh scale, rational and dyadic cells, displayed centers, finite coverage rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1596.1, Definition 1596.2, Definition 1596.3, Definition 1596.4, and Definition 1596.5 bind the finite dyadic interval-cover certificate used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, general open-cover compactness, arbitrary subcover choice, quotient real equality, trichotomy, or ambient Real completeness.

Upgrade path: Lean closure requires carrier admission, classifier preservation, finite coverage nonescape, and compact-uniform handoff for BEDC.Derived.IntervalFiniteCoverCertificateUp.

Chapter 1597

A Concrete Naming Certificate for $\mathsf{DyadicOrder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicOrder}^{\uparrow}$ packet names the finite common-scale comparison certificate used before regular rational names or real-completion consumers read dyadic tolerance order data. It reads $\mathsf{DyadicRatCore}^{\uparrow}$ common-scale data through chapter 291, rational and integer endpoints through chapter 37 and chapter 31, and supplies the comparison ledger consumed by chapter 126, chapter 301, and chapter 38. The packet is a finite dyadic comparison route, not host rational order, trichotomy, quotient equality, a selected limit, ambient real completeness, or a standard bridge.

Definition 1597.1 (DyadicOrder carrier)**.** *A* $\mathsf{DyadicOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, R, S, C, H, K, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, carrying its signed mantissa, unary exponent, denominator ledger, and local dyadic provenance.*
2. *R is the right* $\mathsf{DyadicRatCore}^{\uparrow}$ *row with the same visible endpoint coordinates and provenance shape.*
3. *S is the common-scale row displaying the finite exponent into which L and R are transported before comparison.*
4. *C is the comparison bit and ledger row, recording the transported mantissas at scale S, the finite sign test, and the named outcome consumed by downstream tolerance checks.*
5. *H records componentwise* hsame *transport for the two endpoint rows, common-scale row, comparison ledger, continuation row, provenance row, and local naming row.*
6. *K records displayed* Cont *replay from the two dyadic inputs through the common-scale row and then through the comparison ledger.*
7. *P is the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host rational order, trichotomy, quotient rational equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Theorem 1597.2 (DyadicOrder NameCert obligations)**.** *For every accepted* $\mathsf{DyadicOrder}^{\uparrow}$ *carrier* $O = (L, R, S, C, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicOrder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the left and right* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, common-scale row, comparison ledger, transport, replay, provenance, and local naming row;*
2. *classifier transport, requiring componentwise* hsame *and displayed* Cont *replay to preserve the same left endpoint, right endpoint, common exponent, transported mantissas, and comparison bit;*
3. *common-scale comparison exactness, requiring every accepted comparison to read the exponent and transported mantissas in* S *before consuming the ledger row* C*;*
4. *radius-monotonicity compatibility, requiring downstream tolerance consumers to use the comparison only after the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window monotonicity route has exposed the same finite radius data;*
5. *non-escape, requiring* P *and* N *to name only the displayed rows and to export no host rational order, trichotomy, quotient equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification claim, or bridge certificate.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1597.1. The projection exposes the two dyadic endpoint rows, the common-scale row, the comparison ledger, and the structural rows H, K, P, N.

Classifier transport is read from H and K. Componentwise hsame carries the left and right endpoint rows, while displayed Cont replay repeats the route through S and then through C.

Common-scale exactness follows from the carrier order. The comparison ledger C is available only after the finite common exponent and the two transported mantissas have been displayed in S.

Radius-monotonicity compatibility is a downstream handoff condition. Since the package names only the $\mathsf{DyadicRatCore}^{\uparrow}$ rows and the displayed comparison route, every consumer must first read the scale-window monotonicity data already attached to those dyadic rows.

The exclusions follow from the row inventory. No host rational order, trichotomy, quotient equality, selected limit, ambient completeness, Lean verification row, or bridge row is present in O. □

Theorem 1597.3 (DyadicOrder common-scale comparison)**.** *Every accepted* $\mathsf{DyadicOrder}^{\uparrow}$ *comparison factors through a displayed finite ledger*

$$L, R \longmapsto S \longmapsto C$$

where S *names a common exponent and transported mantissas for the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *inputs, and* C *names the comparison bit read from those transported mantissas. The route cannot use host rational order, an ambient trichotomy principle, quotient equality of rationals, or any comparison object not carried by the finite rows of* O*.*

Proof. Start with an accepted comparison. Carrier admission gives only the left and right dyadic rows before a comparison can be inspected.

Replay K. The replay first transports the two endpoint rows into the common exponent displayed by S, and only then reaches the comparison ledger C.

The comparison bit in C is therefore a readback from the two transported mantissas at the displayed scale. Because the carrier contains no host rational order, trichotomy, quotient equality, or external comparison coordinate, the accepted route is exhausted by L, R, S, C, H, K, P, N. □

Theorem 1597.4 (DyadicOrder radius-monotonicity handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicOrder}^{\uparrow}$ *comparison receives only the finite dyadic comparison row* C *after the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale-window monotonicity route has exposed the relevant radius and endpoint rows. The handoff supplies no host rational order, trichotomy, quotient real equality, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean verification claim, or bridge component.*

Proof. Begin with a downstream consumer. The consumer can inspect the comparison only through the package row P and the local naming row N, both of which point back to the displayed dyadic inputs.

Read the $\mathsf{DyadicRatCore}^{\uparrow}$ scale-window monotonicity route attached to L and R. That route exposes the finite radius and endpoint rows before any comparison ledger is consumed.

After that exposure, replay K through the common-scale row S and the comparison ledger C. The consumer receives the same finite comparison bit and its endpoint provenance, transported by H.

Since the route ends at the displayed comparison ledger and structural rows, it cannot export host order, trichotomy, quotient equality, selected limits, ambient completeness, Lean verification, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicOrder}^{\uparrow}$ packet is generated from left and right DyadicRatCore rows, a common-scale row, a comparison bit and ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1597.1, Definition 1597.2, Definition 1597.3, and Definition 1597.4 seed the finite common-scale comparison object used by dyadic tolerance consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, host rational order, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, common-scale comparison exactness, radius-monotonicity compatibility, downstream handoff, and non-escape proofs for BEDC.Derived.DyadicOrderUp.

Chapter 1598

A Concrete Naming Certificate for $\mathsf{CompleteOrderedReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki's ordered-field language, the $\mathsf{CompleteOrderedReal}^{\uparrow}$ packet records the finite interface where $\mathsf{Real}^{\uparrow}$ names are read together with constructive ordered-field data and located least-upper-bound data. It sits over the ordered-field surface of chapter 1419, the located-supremum boundary of chapter 1427, and the inhabited bounded-family completion surface of chapter 1504. The packet is a BEDC NameCert interface for consumers that need these rows in one visible route; it is not a construction of all reals, not a quotient carrier, and not a classical completeness principle.

Definition 1598.1 (CompleteOrderedReal carrier)**.** *A* $\mathsf{CompleteOrderedReal}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$K = (R, A, S, D, W, H, C, P, N)$$

with the following displayed rows:

1. *R is the family of* $\mathsf{Real}^{\uparrow}$ *endpoint and witness rows read by the packet.*
2. *A is the* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *row supplying displayed addition, negation, multiplication, order, rational-bound, and Archimedean comparison data over the rows of R.*
3. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row supplying the upper-bound and lower-approximation faces for located bounded windows over R.*
4. *D is the* $\mathsf{DedekindCompleteReal}^{\uparrow}$ *row tying inhabited bounded* $\mathsf{Real}^{\uparrow}$ *family windows to the located-supremum witness and terminal real-seal readback.*
5. *W is the local compatibility ledger recording which real endpoints, order comparisons, bounded-family observations, and supremum precision requests are being inspected.*
6. *H records componentwise* hsame *transports for the real rows, ordered-field row, located-supremum row, completion row, compatibility ledger, continuation routes, provenance, and local naming row.*
7. *C records displayed* Cont *replay from real-name inspection through ordered-field comparison, located-supremum witnessing, completion handoff, and local compatibility.*
8. *P is the* Pkg *provenance over* $\mathsf{CompleteOrderedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{DedekindCompleteReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by replaying $R, A, S, D, W, H, C, P, N$. The carrier has no coordinate for a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, arbitrary subset completeness, trichotomy, quotient reals, countable choice, host real equality, or bridge checking.*

Theorem 1598.2 (CompleteOrderedReal NameCert obligations)**.** *For every accepted* $\mathsf{CompleteOrderedReal}^{\uparrow}$ *carrier $K = (R, A, S, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real endpoint and witness rows, ordered-field row, located-supremum row, completion row, compatibility ledger, transports, continuations, provenance, and local naming row;*
2. *ordered-field compatibility, requiring every operation, order, rational-bound, and Archimedean comparison read to pass through A over the same real rows listed in R and W;*
3. *located-supremum compatibility, requiring every upper-bound and lower-approximation read to pass through S and the bounded-family handoff in D at a precision listed in W;*
4. *transport and replay stability, requiring H and C to preserve the route R, A, S, D, W componentwise;*
5. *ledger exhaustion, requiring P and N to export no Dedekind-cut construction, arbitrary subset completeness, trichotomy, quotient real, countable-choice selector, host real equality, or bridge-checked standard object.*

No local obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1598.1. The accepted packet exposes exactly the real rows, ordered-field row, located-supremum row, completion row, compatibility ledger, transport row, replay row, provenance row, and local naming row.

Ordered-field compatibility is inherited through the displayed row A. Its operation, order, rational-bound, and Archimedean comparison reads are available only over the real rows recorded in R and selected by the local ledger W.

Located-supremum compatibility is inherited through the displayed rows S and D. The row S supplies the upper-bound and lower-approximation faces, while D records the inhabited bounded-family handoff that reaches the terminal $\mathsf{Real}^{\uparrow}$ seal.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, and Cont replay in C repeats the route R, A, S, D, W for every accepted request.

Finally, provenance is exhausted by P and N. Since the excluded principles are not coordinates of K, the local $\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}$ surface cannot export them. □

Theorem 1598.3 (CompleteOrderedReal Archimedean order route)**.** *Every order-facing read exported by an accepted* $\mathsf{CompleteOrderedReal}^{\uparrow}$ *carrier factors through*

$$R \longmapsto A \longmapsto W \longmapsto H, C, P, N.$$

Thus the packet presents constructive ordered-field behavior only through the existing $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *surface over displayed* $\mathsf{Real}^{\uparrow}$ *rows. It does not identify those rows with quotient reals, host real equality, trichotomy, a complete host ordered field, or a bridge-checked standard object.*

Proof. Start with an order-facing consumer of K. The only row carrying ordered-field operations and comparisons is A, and the row A is indexed by the displayed real rows in R.

The compatibility ledger W selects the particular endpoints, operation reads, rational-bound checks, and Archimedean comparisons in scope. No order read can bypass W, because the local carrier does not expose an independent host ordered-field coordinate.

The structural rows H, C, P, N then transport, replay, source, and name this route. Since quotient reals, host real equality, trichotomy, complete host ordered fields, and bridge-checked standard objects are absent from the carrier, the route exports none of them. □

Theorem 1598.4 (CompleteOrderedReal located-supremum boundary)**.** *Every supremum-facing read exported by an accepted* $\mathsf{CompleteOrderedReal}^{\uparrow}$ *carrier factors through*

$$R \longmapsto W \longmapsto S \longmapsto D \longmapsto H, C, P, N.$$

The read supplies only the displayed located upper-bound and lower-approximation boundary for the named inhabited bounded family window. It does not supply a Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, arbitrary subset completeness, trichotomy, quotient reals, countable choice, host real equality, or bridge checking.*

Proof. Begin with a supremum-facing request. The local ledger W fixes the family observations, bounds, and precision request currently under inspection over the real rows R.

The row S is then the only located-supremum coordinate. It supplies the upper-bound face and the lower-approximation face for the precision listed in W.

The row D is the only completion-facing handoff. It ties the located-supremum read to the inhabited bounded-family route and terminal $\mathsf{Real}^{\uparrow}$ seal without adding a separate arbitrary subset classifier.

Finally, H, C, P, N transport, replay, source, and name exactly this displayed route. The excluded completeness and classical-order principles have no carrier coordinate, so they cannot be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteOrderedReal}^{\uparrow}$ packet is generated from displayed $\mathsf{Real}^{\uparrow}$ endpoint and witness rows, an $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ ordered-field row, a $\mathsf{LocatedSupremum}^{\uparrow}$ least-upper-bound row, a $\mathsf{DedekindCompleteReal}^{\uparrow}$ inhabited bounded-family handoff, a compatibility ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1598.1, Definition 1598.2, Definition 1598.3, and Definition 1598.4 seed the finite ordered-field and located-supremum interface for displayed $\mathsf{Real}^{\uparrow}$ data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteOrderedReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut construction, arbitrary subset completeness, trichotomy, quotient reals, countable choice, bridge checking, host real equality, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, ordered-field compatibility, located-supremum compatibility, transport stability, replay stability, and ledger exhaustion for BEDC.Derived.CompleteOrderedRealUp.

Chapter 1599

A Concrete Naming Certificate for $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$

Following Birkhoff and Davey–Priestley on order completion, the $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ packet names the finite two-sided cut-closure surface that compares lower-row and upper-row observations without constructing $\mathsf{Real}^{\uparrow}$ by Dedekind cuts. It sits between the one-sided $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ ledgers, the located-cut boundary of chapter 404, the representation comparison of chapter 1424, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet supplies a finite order-completion comparison boundary for Real and Completion consumers; it is not a quotient-cut carrier, not a host supremum principle, and not a construction of $\mathsf{Real}^{\uparrow}$.

Definition 1599.1 (Dedekind–MacNeille completion carrier)**.** *A* $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, K, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the finite lower-row closure ledger. It records the displayed* $\mathsf{LowerReal}^{\uparrow}$ *observations, their downward stability, and the finite meet requests allowed by the packet.*
2. *U is the finite upper-row closure ledger. It records the displayed* $\mathsf{UpperReal}^{\uparrow}$ *observations, their upward stability, and the finite join requests allowed by the packet.*
3. *K is the* $\mathsf{LocatedCut}^{\uparrow}$ *handoff row. It ties the lower and upper ledgers by located separation data and records only the displayed gap witnesses used by finite consumers.*
4. *Q is the* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *boundary row. It compares cut-shaped observations from L, U, K with the regular Cauchy readback route accepted at the representation boundary.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal route. It is reached only after the lower closure, upper closure, located-cut handoff, and comparison boundary have been displayed.*
6. *H records componentwise* hsame *transport for the lower ledger, upper ledger, located-cut handoff, comparison boundary, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a two-sided cut request through L, U, K, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindMacNeilleCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, Q, E, H, C, P, N$. The carrier has no coordinate for quotient cuts, host suprema, arbitrary complete lattices, selected cuts, or a Dedekind construction of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DedekindMacNeilleCompletionUp.DedekindMacNeilleCompletionTasteGate single carrier alignment`

Theorem 1599.2 (Dedekind–MacNeille completion NameCert obligations)**.** *For every accepted* $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ *carrier $D = (L, U, K, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindMacNeilleCompletion}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed lower closure ledger, upper closure ledger, located-cut handoff, comparison boundary, Real seal, transport, replay, provenance, and local naming row;*
2. *two-sided closure compatibility, requiring every finite lower-row request in L and every finite upper-row request in U to meet only through the located separation witnesses recorded in K;*
3. *representation-boundary factorization, requiring every cut-shaped observation exposed by L, U, K to pass through Q before any* $\mathsf{Real}^{\uparrow}$*-facing read is available;*
4. *transport and replay stability, requiring* hsame *transport H and* Cont *replay C to preserve the ordered route L, U, K, Q, E componentwise;*
5. *ledger non-escape, requiring P and N to export no quotient cut, no host supremum, no arbitrary complete-lattice closure, no selected cut, and no Dedekind construction of* $\mathsf{Real}^{\uparrow}$.

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1599.1. The accepted packet exposes exactly the lower ledger, upper ledger, located-cut handoff, representation comparison, Real seal, and structural rows H, C, P, N.

The two-sided closure compatibility row is finite. A lower request reads only the displayed rows in L, an upper request reads only the displayed rows in U, and their common comparison is the located separation information stored in K.

Representation-boundary factorization is forced by the row order. Cut-shaped observations from L, U, K cannot reach a Real-facing seal directly; they first pass through the $\mathsf{DedekindCauchyComparison}^{\uparrow}$ row Q.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, and Cont replay in C repeats the same route L, U, K, Q, E for every accepted two-sided request.

Finally, provenance and naming are exhausted by P and N. Since quotient cuts, host suprema, arbitrary complete-lattice closure, selected cuts, and a Dedekind construction of $\mathsf{Real}^{\uparrow}$ are not coordinates of D, the local NameCert surface cannot export them. □

Lean4 ✓ `BEDC.Derived.DedekindMacNeilleCompletionUp.DedekindMacNeilleCompletionNameCert obligations`

Theorem 1599.3 (Dedekind–MacNeille finite cut closure)**.** *Let $D = (L, U, K, Q, E, H, C, P, N)$ be an accepted* $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ *carrier. Every finite two-sided cut-closure read exported by D factors through the route*

$$L, U \longmapsto K \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet supplies only the displayed lower-row closure, upper-row closure, located-cut handoff, representation comparison, and terminal Real-seal route. It supplies no quotient cut, no host supremum, no arbitrary Dedekind–MacNeille completion object, no selected cut, and no construction of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DedekindMacNeilleCompletionUp.DedekindMacNeilleCompletionCarrier cut closure`

Proof. Begin with a two-sided cut-closure read. The lower face is the finite ledger L, and the upper face is the finite ledger U; no ambient poset completion or host lattice is consulted.

The two faces meet only through K. This row records the located separation handoff, so a consumer may inspect the displayed gap witnesses but cannot form a quotient cut or selected cut from them.

The representation boundary is Q. Every cut-shaped observation from the displayed rows is compared with the regular Cauchy readback route before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

The rows H, C, P, N then transport, replay, source, and name exactly this route. Because the packet contains no coordinate for host suprema, arbitrary complete lattices, quotient cuts, selected cuts, or a Dedekind construction of $\mathsf{Real}^{\uparrow}$, none of those objects can be exported by the cut-closure read. □

Theorem 1599.4 (Dedekind–MacNeille Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ *packet must pass through the* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{DedekindCauchyComparison}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of the displayed carrier before reading the terminal seal. The packet cannot be read as a construction of* $\mathsf{Real}^{\uparrow}$*, a quotient cut, a host supremum, a selected cut, or an arbitrary complete-lattice closure.*

Lean4 ✓ `BEDC.Derived.DedekindMacNeilleCompletionUp.DedekindMacNeilleCompletionRealSealNonescape`

Proof. Project the carrier $D = (L, U, K, Q, E, H, C, P, N)$ from Definition 1599.1. The only Real-facing route first reads the lower ledger L and upper ledger U, then the located-cut handoff K, then the representation comparison boundary Q, and only then the Real seal row E.

Apply the cut-closure theorem. It identifies the exported finite route as

$$L, U \longmapsto K \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Row exhaustion gives the nonescape claim. The structural rows H, C, P, N transport, replay, source, and name the same displayed route, and no carrier coordinate stores quotient cuts, host suprema, selected cuts, arbitrary complete-lattice closure, or a Dedekind construction of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindMacNeilleCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DedekindMacNeilleCompletion}^{\uparrow}$ packet is generated from finite lower-row closure data, finite upper-row closure data, a LocatedCut handoff, a DedekindCauchyComparison boundary, a RealUp seal route, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindMacNeilleCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1599.1, Definition 1599.2, and Definition 1599.3 seed the finite two-sided cut-closure comparison boundary for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindMacNeilleCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient cuts, host suprema, arbitrary complete lattices, selected cuts, Lean verification, a standard bridge, or Dedekind construction of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, two-sided closure compatibility, LocatedCut handoff, DedekindCauchyComparison factorization, Real seal non-escape, transport stability, replay stability, and ledger non-escape for BEDC.Derived.DedekindMacNeilleCompletionUp.

Chapter 1600

A Concrete Naming Certificate for $\mathsf{DirectedSubnet}^{\uparrow}$

Following Kelley-style nets and Bourbaki general topology, the $\mathsf{DirectedSubnet}^{\uparrow}$ packet records cofinal subnet reindexing as finite BEDC rows rather than as a quotient of nets or a choice of subnet representatives. It sits between the source directed-window surface of chapter 589 and the downstream limit consumer of chapter 1561. Its displayed reindexing route also reads the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal boundary of chapter 38.

Definition 1600.1 (DirectedSubnet carrier)**.** *A* $\mathsf{DirectedSubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, J, \phi, K, L, S, R, D, A, H, C, P, N)$$

over $\mathsf{CauchyNet}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *with the following displayed rows:*

1. *I is the finite source directed-window ledger whose entries are the source windows inspected by the packet.*
2. *J is the finite target directed-window ledger whose entries name the subnet windows requested by downstream consumers.*
3. *$\phi : J \to I$ is the displayed cofinal map from target directed windows into source directed windows. Its cofinal coverage row records, for each displayed source window in I, a target window in J whose image lies above it inside the source directed order recorded by K.*
4. *K is the accepted source* $\mathsf{CauchyNet}^{\uparrow}$ *row read on the source windows named by I.*
5. *L is the target or subnet* $\mathsf{CauchyNet}^{\uparrow}$ *row whose windows are obtained by replaying K along ϕ.*
6. *S records the* $\mathsf{StreamName}^{\uparrow}$ *windows used to read both the source observations and their target replay.*
7. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the scheduled windows.*
8. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger consulted when the source and target readings are compared.*
9. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after K, ϕ, L, S, R, D have been replayed.*

10. *H records componentwise* hsame *transport for the source, target, map, window, readback, tolerance, seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a target-window request through ϕ into the source CauchyNet row and then back into the target CauchyNet row.*
12. *P is the* Pkg *provenance over the listed BEDC dependencies.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}$ *row.*

The carrier exports no quotient of nets, no selected limit, no cofinal choice operator, no host real equality, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetTasteGate single carrier alignment`

Theorem 1600.2 (DirectedSubnet NameCert obligations)**.** *For every accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *carrier $M = (I, J, \phi, K, L, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source ledger, target ledger, cofinal map, source CauchyNet row, target CauchyNet row, window, readback, tolerance, seal, transport, continuation, provenance, and local naming rows;*
2. *cofinal-map coverage, requiring every displayed source window requested by the packet to be reached by a target window through the row $\phi : J \to I$;*
3. *CauchyNet handoff, requiring the target row L to be read only by replaying the source row K along ϕ and the displayed window, readback, and tolerance rows S, R, D;*
4. *Real-seal non-escape, requiring the seal row A to remain downstream of K, ϕ, L, S, R, D;*
5. *ledger refusal, excluding quotient-net equality, selected subnet limits, cofinal choice outside ϕ, host real equality, and ambient completeness.*

No obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetNameCertObligations`

Proof. Carrier admission is projection from Definition 1600.1. The source and target ledgers, the displayed cofinal map, both CauchyNet rows, the StreamName windows, regular readback, dyadic tolerance ledger, Real seal, and structural rows are all visible coordinates of the same finite BHist packet.

Cofinal-map coverage is read from the ϕ row together with the source directed order recorded by K. For each source window the packet displays, the coverage row names a target window whose image lies above it, so the packet has no need for an ambient choice function selecting unseen target windows.

The CauchyNet handoff is then replayed through C and transported by H. A target-window request first reads J, applies ϕ into the source ledger I, consumes the source row K, and only then exports the target row L through the same S, R, D evidence.

The seal row A is downstream of this replay. Since the provenance and local naming rows package only those displayed coordinates, no quotient-net relation, selected limit, host real equality, or ambient completeness coordinate can be exported as a NameCert obligation. □

Theorem 1600.3 (DirectedSubnet CauchyNet handoff)**.** *Let $M = (I, J, \phi, K, L, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *packet. Every target* $\mathsf{CauchyNet}^{\uparrow}$ *row exported by L factors through the displayed route*

$$J, \phi, I, K, S, R, D \longmapsto L.$$

Thus the target CauchyNet packet is available only through the displayed cofinal map, the source CauchyNet row, the $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The route is a projection and replay over finite rows, not a quotient-net construction and not a choice argument.*

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetCarrier cauchynet handoff`

Proof. Start with a consumer of the target row L. Carrier admission exposes L only as a coordinate of the same packet that also contains J, ϕ, I, K, S, R, D.

Replay the continuation row C. The consumer's target-window request first lands in J, then follows the displayed map ϕ into the source ledger I. The source $\mathsf{CauchyNet}^{\uparrow}$ row K supplies the inspected source windows, while S, R, D supply the scheduled windows, regular rational readback, and dyadic tolerance comparisons needed for the replay.

Transport by H preserves this route componentwise. The package and local NameCert rows record exactly the same coordinates, so the target row L has no exported path through quotient-net equality, hidden cofinal choice, a selected limit, host real equality, or ambient completeness. □

Theorem 1600.4 (DirectedSubnet cofinal replay)**.** *Let $M = (I, J, \phi, K, L, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *packet. Any consumer replaying a target directed window must expose the target ledger J, the displayed cofinal map ϕ, the source ledger I, the source* $\mathsf{CauchyNet}^{\uparrow}$ *row K, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D before it reaches the target row L and the terminal* $\mathsf{Real}^{\uparrow}$ *seal A.*

Thus the cofinal replay route is the displayed finite chain

$$J, \phi, I, K, S, R, D \longmapsto L \longmapsto A.$$

It exports no quotient-net equality, hidden cofinal choice, selected subnet limit, host real equality, or ambient completeness principle.

Proof. Project the carrier tuple from Definition 1600.1. The target ledger J, map ϕ, source ledger I, source row K, target row L, window row S, readback row R, tolerance row D, and seal row A are all visible coordinates of the same finite packet.

Replay the continuation row C. A target-window request enters through J, follows ϕ into I, reads the source CauchyNet row K, and uses the same S, R, D evidence before producing the target row L.

Transport by H preserves each component of this chain. Since the package and local naming rows cite only the displayed coordinates, the terminal seal A remains downstream of the replay and no absent quotient, choice, limit, equality, or completeness coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetCofinalReplay`

Theorem 1600.5 (DirectedSubnet target Cauchy window)**.** *Every completion-facing consumer of an accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *packet must expose the target directed ledger J, the cofinal map ϕ, the target* $\mathsf{CauchyNet}^{\uparrow}$ *row L, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{Real}^{\uparrow}$ *seal A, componentwise transport H, displayed replay C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}$ *row N before it may read a limit or completion*

endpoint. The target window is therefore a displayed finite CauchyNet route, not a quotient net, selected subnet representative, countable-choice functional, or ambient Real completeness claim.

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetTargetCauchyWindow`

Proof. Project the carrier tuple from Definition 1600.1. The target ledger J, cofinal map ϕ, target row L, window row S, readback row R, tolerance row D, seal row A, and structural rows H, C, P, N are all visible coordinates of the same finite packet.

Apply the CauchyNet handoff of Definition 1600.3. Any read of L must already pass through the target request in J, the displayed map ϕ, the source route, and the same S, R, D evidence that makes the target row Cauchy-facing.

It remains to inspect a completion endpoint. The seal row A is downstream of the target CauchyNet route, while H, C, P, N only transport, replay, cite, and name the displayed coordinates. Hence a completion consumer receives no quotient-net equality, selected subnet representative, countable-choice data, or ambient Real completeness row. □

Theorem 1600.6 (DirectedSubnet target-filter nonescape)**.** *Every target-side Cauchy read exported by an accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *carrier must pass through the displayed source* $\mathsf{CauchyNet}^{\uparrow}$ *row, the cofinal replay row, the target Cauchy window, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame/Cont *transport,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}$ *row. It exports no selected subnet limit, quotient net, cofinal choice outside the displayed map, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetTargetFilterNonescape`

Proof. Start with the carrier of Definition 1600.1. The source CauchyNet row K, target CauchyNet row L, displayed cofinal map ϕ, stream windows S, readback R, tolerance ledger D, seal row A, and structural rows H, C, P, N are visible coordinates of the same packet.

Use the NameCert obligations of Definition 1600.2. They force a target-side read to cite the local naming row only after the carrier rows, transport row, replay row, and provenance row have all been exposed.

Apply the CauchyNet handoff of Definition 1600.3, then the cofinal replay of Definition 1600.4, and finally the target Cauchy window of Definition 1600.5. This route keeps the target read downstream of $K, \phi, L, S, R, D, A, H, C, P, N$, so the packet has no coordinate from which to export a selected subnet limit, quotient net, hidden cofinal choice, or ambient completeness claim. □

Theorem 1600.7 (DirectedSubnet obligation route)**.** *Let* $M = (I, J, \phi, K, L, S, R, D, A, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSubnet}^{\uparrow}$ *packet. Carrier admission, cofinal-map coverage, CauchyNet handoff, Real-seal nonescape, and ledger refusal are all projected from the same finite tuple* M*. Every obligation cites the visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of that tuple, and no obligation is delegated to quotient-net equality, hidden cofinal choice, selected subnet limits, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DirectedSubnetUp.DirectedSubnetObligationRoute`

Proof. Carrier admission is the projection of Definition 1600.1. The tuple displays the source ledger I, target ledger J, cofinal map ϕ, source and target CauchyNet rows K, L, the stream, readback, tolerance, and seal rows S, R, D, A, and the structural rows H, C, P, N.

Cofinal-map coverage is the same coverage row used by Definition 1600.4. It reads each requested source window through the displayed map $\phi : J \to I$ and never invokes an unseen cofinal selector.

The CauchyNet handoff is exactly the route of Definition 1600.3. A target-window request must expose J, ϕ, I, K, S, R, D before it exports L.

The Real-seal boundary is the target-window route of Definition 1600.5. The seal row A remains downstream of the target CauchyNet route, while H, C, P, N only transport, replay, source, and name the displayed coordinates. Since the refused quotient, choice, limit, equality, and completeness rows are absent from M, ledger refusal is a projection from the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1600.1, Definition 1600.2, Definition 1600.3, Definition 1600.4, Definition 1600.5, and Definition 1600.7 bind the finite carrier to the CauchyNet handoff, cofinal replay, target Cauchy window, target-filter nonescape, and displayed NameCert obligation route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirectedSubnet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected limits, cofinal choice outside the displayed map, host real equality, ambient completeness, or Lean verification.

Upgrade path: Public closure requires checked carrier admission, cofinal-map coverage, CauchyNet handoff, Real-seal non-escape, and ledger refusal for the same finite packet.

Constructive story: A $\mathsf{DirectedSubnet}^{\uparrow}$ packet is generated from source and target directed ledgers, a displayed cofinal map, source and target CauchyNet rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 1601

A Concrete Naming Certificate for $\mathsf{FilterBase}^{\uparrow}$

Following Kelley and Bourbaki on filters and filter bases, the $\mathsf{FilterBase}^{\uparrow}$ packet names the finite directed-base surface that completion routes may consume before any Cauchy specialization is chosen. It sits upstream of the $\mathsf{CauchyFilter}^{\uparrow}$ surface in chapter 333 and gives $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumers a common finite base interface. The packet records displayed base rows, directed lower-bound witnesses, refinement ledgers, transport, replay, provenance, and local naming only; it is not an ultrafilter, not a quotient filter, not a selected limit, and not an ambient completeness principle.

Definition 1601.1 (FilterBase finite carrier)**.** *A* $\mathsf{FilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, E, D, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the base packets that may be requested by a downstream filter or completion consumer.*
2. *E is the base-element row. Each entry stores the displayed* BHist *surface for one base packet and the local admissibility read attached to that packet.*
3. *D is the directed lower-bound row. For every two displayed base packets it names a third displayed packet together with the two containment reads back to the requested packets.*
4. *R is the refinement ledger. It records every inclusion, common-refinement, and replay edge used when a consumer narrows one displayed base packet to another.*
5. *H records componentwise* hsame *transport for index rows, base-element rows, lower-bound witnesses, refinement ledgers, continuation rows, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a base request through the selected element, through the directed lower-bound witness, and through the refinement ledger exposed to downstream consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through I, E, D, R, H, C, P, N. There is no coordinate for ultrafilters, Zorn arguments, choice functions, quotient filters, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseTasteGate single carrier alignment`

Theorem 1601.2 (FilterBase NameCert obligations)**.** *For every accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier* $B = (I, E, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FilterBase}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed finite index row, base-element row, directed lower-bound row, refinement ledger, componentwise transport, continuation replay, provenance, and local naming row;*
2. *directedness, requiring each pair of displayed base packets to read a lower-bound witness from D and the two containment reads recorded by R;*
3. *refinement stability, requiring every narrowed base request to remain inside the displayed ledger R after* hsame *transport by H and* Cont *replay by C;*
4. *downstream exposure, requiring every filter or completion consumer to receive only the displayed finite base row I, E, D, R together with its structural rows H, C, P, N;*
5. *ledger refusal, excluding ultrafilters, Zorn arguments, choice functions, quotient filters, selected limits, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1601.1. The projection exposes exactly the finite index row, the base-element row, the directed lower-bound row, the refinement ledger, and the structural rows H, C, P, N.

Directedness is the stored row D. A request for two base packets must read the selected lower-bound packet from the same finite carrier and then replay the two containment edges recorded in R.

Refinement stability is componentwise. Transport by H preserves the visible coordinates, while C replays the same base request through the selected element and the refinement ledger.

Finally, the provenance row P and the naming row N name only this carrier. Since ultrafilters, Zorn arguments, choice functions, quotient filters, selected limits, and ambient completeness are not coordinates of B, the local NameCert surface cannot export them. □

Theorem 1601.3 (FilterBase Cauchy-filter and completion handoff)**.** *Let $B = (I, E, D, R, H, C, P, N)$ be an accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer of the base route in chapter 333, chapter 1473, chapter 1506, and chapter 1569 factors through the finite route*

$$I, E, D, R \longmapsto H, C, P, N.$$

Thus completion consumers receive only the displayed finite base surface: base packets, directed lower-bound witnesses, refinement ledgers, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. They do not receive ultrafilters, choice data, quotient filters, selected limits, or ambient completeness.*

Proof. Begin with a downstream request. The carrier gives the consumer access to the base rows only through I and E, so the request first selects displayed base packets rather than an external filter object.

When the consumer needs common refinement, it reads D and the containment rows in R. These rows are finite coordinates of the same packet, and C records the replay route from the requested packets through the displayed lower-bound witness.

Transport and provenance preserve the same boundary. The row H moves the listed coordinates componentwise, while P and N expose the packet to $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ only through the displayed finite route.

No further coordinate remains. Hence the completion-facing handoff supplies a finite base surface, not an ultrafilter, a choice function, a quotient filter, a selected limit, or an ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier cauchyfilter completion handoff`

Theorem 1601.4 (FilterBase directed-refinement coverage)**.** *Let $B = (I, E, D, R, H, C, P, N)$ be an accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier. For every pair of displayed base packets requested through I and E, the directed lower-bound row D and the refinement ledger R expose a third displayed packet whose two containment reads replay through C and remain stable under H. This coverage is internal to the finite* BHist, hsame, Cont, Pkg, *and* NameCert *rows of B; it does not invoke ultrafilters, Zorn arguments, quotient filters, selected limits, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier directed refinement coverage`

Proof. Start from Definition 1601.1. A request for two displayed base packets is a read through the index row I and base-element row E, so the request has no access to an external filter object.

Apply Definition 1601.2. Its directedness row requires D to name a lower-bound packet and requires R to record the two containment reads back to the requested packets. The replay row C is therefore the only route by which the consumer can inspect the common refinement.

Apply Definition 1601.3 for downstream stability. The same finite route is handed to $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumers, while H transports the listed coordinates componentwise.

Since neither the carrier nor the handoff theorem contains an ultrafilter row, Zorn argument, quotient filter, selected limit, or ambient completeness witness, directed-refinement coverage is exactly the displayed finite lower-bound surface recorded by D, R, H, C, P, N. □

Theorem 1601.5 (FilterBase refinement stability)**.** *Let $B = (I, E, D, R, H, C, P, N)$ be an accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier. Every refinement of a displayed base packet that is admitted through I, E must remain in the same refinement ledger R after componentwise* hsame *transport by H and displayed* Cont *replay by C. The stable refinement read is therefore a finite row of B, not an external inclusion proof, ultrafilter argument, quotient filter, selected limit, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier refinement stability`

Proof. Start from the carrier definition. The only refinement coordinate is R, and every displayed base packet is first read through I and E.

Apply Definition 1601.2. Its refinement-stability obligation requires narrowed base requests to remain inside R after transport and replay. Thus H can only move the listed rows componentwise, while C replays the already displayed refinement route.

Finally apply Definition 1601.4. Directed lower-bound reads also return to the same ledger R, so refinement stability remains internal to the finite carrier and cannot introduce an external filter principle. □

Theorem 1601.6 (FilterBase classifier stability)**.** *If two accepted* $\mathsf{FilterBase}^{\uparrow}$ *carriers agree on* I, E, D, R, H, C, P, N*, then every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CauchyFilterCompletio}$ *consumer reads the same finite base surface from them. Classifier stability is determined by the displayed index, base-element, directed lower-bound, refinement, transport, replay, provenance, and naming rows; no hidden completion row or quotient filter can alter the comparison.*

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier classifier stability`

Proof. Use Definition 1601.1 to inspect the classifier. It compares accepted packets only through I, E, D, R, H, C, P, N.

Apply Definition 1601.3. Downstream consumers for $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ receive exactly the same displayed finite route.

The refinement-stability theorem keeps narrowed requests inside R under H and C. Therefore agreement on the carrier rows is enough to preserve every classifier read visible to the listed consumers. □

Theorem 1601.7 (FilterBase completion branch coverage)**.** *Every completion-facing branch that consumes an accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier factors through the same finite base route before reaching* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*. The branch may use displayed base packets, directed lower-bound witnesses, and refinement ledgers, but it receives no selected limit, quotient filter, ultrafilter, Zorn argument, or ambient completeness datum from the carrier.*

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier completion branch coverage`

Proof. Begin with a completion-facing request. By Definition 1601.3, the request is routed through I, E, D, R and then through H, C, P, N before any sibling completion chapter can consume it.

Directed-refinement coverage supplies the common lower-bound witness needed by the branch. It reads that witness from D and the two containment routes from R, with replay through C.

Refinement stability keeps subsequent narrowing inside the same finite ledger. Hence the branch coverage is exactly the displayed filter-base surface shared by the Cauchy-filter and completion siblings. □

Theorem 1601.8 (FilterBase ledger refusal)**.** *For every accepted* $\mathsf{FilterBase}^{\uparrow}$ *carrier* $B = (I, E, D, R, H, C, P, N)$*, the refinement ledger* R *refuses to act as an ultrafilter ledger, a choice-function ledger, a quotient-filter ledger, a selected-limit ledger, or an ambient-completeness ledger. Any consumer that asks for one of those rows receives only the displayed finite base route and the explicit absence of such a coordinate.*

Lean4 ✓ `BEDC.Derived.FilterBaseUp.FilterBaseCarrier ledger refusal`

Proof. Project the carrier rows from Definition 1601.1. The row list contains finite indices, base elements, directed lower-bound witnesses, refinement ledgers, transport, replay, provenance, and local naming only.

Use the ledger-refusal obligation in Definition 1601.2. It excludes ultrafilters, Zorn arguments, choice functions, quotient filters, selected limits, and ambient completeness from the local $\mathsf{NameCert}$ surface.

The handoff and completion-coverage theorems do not add a new coordinate. They only expose the same finite route to $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$, so the refusal remains visible to every downstream consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{FilterBase}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterBase}^{\uparrow}$ packet is generated from finite base rows, directed lower-bound witnesses, refinement ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterBase}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1601.2, Definition 1601.3, Definition 1601.4, Definition 1601.5, Definition 1601.6, Definition 1601.7, and Definition 1601.8 bind the directed-refinement, classifier-stability, completion-branch, and ledger-refusal obligations to the finite BHist, hsame, Cont, Pkg, and NameCert rows, with Definition 1601.3 and Definition 1601.7 giving the bridge surface to $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilters, Zorn arguments, choice, quotient filters, selected limits, Lean verification, bridge checking, or ambient completeness.

Upgrade path: No paper-side theory upgrade remains; Lean verification and bridge checking remain separate verification-axis work.

Chapter 1602

A Concrete Naming Certificate for $\mathsf{BaireMetric}^{\uparrow}$

Following Brouwer's finite-prefix presentation of choice sequences, the Bishop–Bridges metric treatment of countable product spaces, and Kelley's prefix topology, the $\mathsf{BaireMetric}^{\uparrow}$ packet records the standard prefix ultrametric on $\mathsf{BaireSpace}^{\uparrow}$ as finite BEDC rows. It sits between the prefix carrier of chapter 334, the metric parent surface of chapter 57, and the strong-triangle interface of chapter 1535. The packet names only displayed first-disagreement windows, dyadic radius comparisons, stream-name schedules, transport, replay, provenance, and a local certificate; it does not import completed streams, quotient equality of infinite names, real-valued host metrics, or external topology.

Definition 1602.1 (BaireMetric prefix-distance carrier)**.** *A* $\mathsf{BaireMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, W, D, R, U, S, H, C, P, N)$$

over $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{UltrametricSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *with the following displayed rows:*

1. *B is the accepted* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet whose finite stream windows supply the two names under comparison.*
2. *W is the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window. It records the inspected prefix length, the equality rows below that length, and the first displayed coordinate where the two prefix windows separate, when such a coordinate is present.*
3. *D is the dyadic radius row. It records the radius 2^{-k} attached to the first displayed disagreement at k, together with the zero-radius branch for displayed equality across the inspected window.*
4. *R is the parent* $\mathsf{Met}^{\uparrow}$ *row that reads the displayed point names, zero-distance boundary, symmetry row, and ordinary triangle route from B, W, D.*
5. *U is the* $\mathsf{UltrametricSpace}^{\uparrow}$ *witness whose strong max-radius comparison is read from the same first-disagreement window and dyadic radius row.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule controlling which finite prefixes of the two Baire names may be inspected by a consumer.*
7. *H records componentwise* hsame *transport for the BaireSpace row, Nat window, dyadic radius row, parent metric row, ultrametric witness, stream schedule, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a requested prefix comparison through S, then through B, W, D, and only then into the parent metric row R and ultrametric witness U.*
9. *P is the* Pkg *provenance over the listed BEDC dependencies.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireMetric}^{\uparrow}}$ *row.*

The classifier accepts the packet only when every distance comparison is read through the displayed first-disagreement window and dyadic radius row. It has no coordinate for a completed infinite stream, a quotient of Baire names, a host real-valued distance function, countable choice, or an external topological space.

Theorem 1602.2 (BaireMetric NameCert obligations)**.** *For every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier $M = (B, W, D, R, U, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireMetric}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed BaireSpace prefix packet, Nat first-disagreement window, dyadic radius row, parent MetricSpace row, UltrametricSpace witness, StreamName schedule, transport, replay, provenance, and local naming rows;*
2. *prefix-distance admission, requiring every positive radius claim to name the first displayed disagreement in W and every zero-radius claim to remain inside the inspected equality window;*
3. *metric compatibility, requiring the parent* $\mathsf{Met}^{\uparrow}$ *row R to read zero-distance, symmetry, and ordinary triangle evidence only from B, W, D;*
4. *ultrametric compatibility, requiring the witness U to read its strong max-radius comparison from the same first-disagreement and dyadic-radius rows;*
5. *ledger refusal, excluding completed streams, quotient equality of Baire names, host real-valued metrics, countable-choice selectors, and external topology.*

No obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricNamecertObligations`

Proof. Carrier admission is projection from Definition 1602.1. The tuple lists exactly the BaireSpace prefix packet, first-disagreement window, dyadic radius row, parent metric row, ultrametric witness, stream schedule, transport, continuation, provenance, and local name.

Prefix-distance admission is read from W and D. A positive comparison names the least displayed coordinate where the two scheduled prefixes disagree, while the zero-radius branch records only equality across the inspected finite window.

Metric compatibility is then replayed through R. The zero-distance boundary, symmetry row, and ordinary triangle route do not inspect a separate distance function; they use the same prefix windows and dyadic radius comparisons already displayed by B, W, D.

Ultrametric compatibility is read from U. Since the controlling radius is determined by the earliest displayed disagreement, the strong max-radius comparison factors through the same window and cannot bypass the dyadic ledger.

The structural rows H, C, P, N transport, replay, source, and name only those coordinates. Hence no completed stream, quotient equality of infinite names, host real metric, countable-choice selector, or external topological datum can appear as a NameCert obligation. □

Theorem 1602.3 (BaireMetric root observation carrier)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *root packet. Every root observation of M is carried by finite* BHist

prefix windows: the consumer first reads the $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, then the displayed prefix packet* B*, the finite first-disagreement window* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D*, before any* $\mathsf{Met}^{\uparrow}$ *row* R*,* $\mathsf{UltrametricSpace}^{\uparrow}$ *row* U*,* hsame *transport* H*,* Cont *replay* C*, provenance row* P*, or local* NameCert *row* N *is available. No root observation asserts equality of completed streams.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier root observation`

Proof. Project the accepted carrier of Definition 1602.1. The only observation source is the finite prefix surface S, B, W, D, with W recording inspected BHist prefix data and the displayed first-disagreement branch.

The parent $\mathsf{Met}^{\uparrow}$ and $\mathsf{UltrametricSpace}^{\uparrow}$ rows are downstream of that prefix surface. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so a root observation has no coordinate from which to read completed-stream equality. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootObservationCarrier`

Theorem 1602.4 (BaireMetric ultrametric window stability)**.** *For an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, extending a scheduled finite* BHist *prefix window through the same* $\mathsf{StreamName}^{\uparrow}$ *row preserves the ultrametric comparison route. The* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row remains read from the displayed first-disagreement window, the parent* $\mathsf{Met}^{\uparrow}$ *row* R *and* $\mathsf{UltrametricSpace}^{\uparrow}$ *row* U *consume that same radius data, and* hsame *transport with* Cont *replay preserves the local* NameCert *surface. The extension never supplies a completed-stream equality claim.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier ultrametric window stability`

Proof. Start with a scheduled prefix read. By Definition 1602.3, the read exposes only the finite window route S, B, W, D before the parent metric and ultrametric rows are consumed.

When the window is extended through the same schedule, the displayed first-disagreement branch is either preserved or moved to a later inspected coordinate. In both cases the dyadic radius is still read from D and from no other source.

The rows R and U consume that radius after the prefix window has been displayed. Componentwise hsame transport and Cont replay keep the route on the same carrier coordinates, while the local NameCert row names only that finite surface. □

Theorem 1602.5 (BaireMetric NameCert obligation surface)**.** *Every root-level* $\mathsf{NameCert}_{\mathsf{BaireMetric}^{\uparrow}}$ *obligation for an accepted packet* $M = (B, W, D, R, U, S, H, C, P, N)$ *is exhausted by finite* BHist *prefix-window admission,* $\mathsf{StreamName}^{\uparrow}$ *scheduling,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius exactness, parent* $\mathsf{Met}^{\uparrow}$ *compatibility,* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No additional obligation asserts completed-stream equality, quotient equality of Baire names, a host real-valued metric, or external topology.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricNamecertObligationSurface`

Proof. Apply the NameCert obligations of Definition 1602.2. The carrier admission, prefix-distance admission, metric compatibility, ultrametric compatibility, and ledger refusal rows list the entire obligation surface.

Use Definition 1602.3 to identify the root observation source with finite BHist prefix windows scheduled by $\mathsf{StreamName}^{\uparrow}$ and measured by the $\mathsf{DyadicRatCore}^{\uparrow}$ row. Then use Definition 1602.4 to keep parent metric and ultrametric consumers on the same finite route under window extension.

The remaining rows are structural: hsame transports the displayed coordinates, Cont replays them, Pkg records provenance, and the local NameCert row names them. Since the carrier has no other coordinate, completed-stream equality and external metric data are outside the obligation surface. □

Theorem 1602.6 (BaireMetric prefix-window admission)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* BaireMetric$^{\uparrow}$ *carrier. Every accepted distance read admits a finite prefix window: before the dyadic radius row is inspected, the continuation row displays a* StreamName$^{\uparrow}$ *schedule S, an accepted* BaireSpace$^{\uparrow}$ *prefix packet B, and a* Nat$^{\uparrow}$ *window W that records the inspected length and the equality rows below the first displayed separation.*

Proof. Project the carrier of Definition 1602.1. The continuation row C routes every comparison through the stream schedule S and the BaireSpace prefix packet B before reaching any radius coordinate.

The window row W is the only Nat-valued inspection coordinate in the carrier. Thus the accepted comparison must name the inspected finite length and the displayed equality rows below any first disagreement before D, R, or U can be read. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier prefix window admission`

Theorem 1602.7 (BaireMetric ultrametric window obligations)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* BaireMetric$^{\uparrow}$ *carrier. The ultrametric strong-triangle obligation is admitted only after the finite* StreamName$^{\uparrow}$ *schedule S has exposed the prefix packet B and the first-disagreement window W before dyadic radius inspection. The witness U must read the same W, D branch as the parent metric row R, so its max-radius comparison cannot be replaced by completed-stream equality, quotient names, or an external topology.*

Proof. Use Definition 1602.6. Every accepted comparison first exposes the schedule, prefix packet, and Nat window, so the dyadic row is not available before a finite first-disagreement read.

Apply Definition 1602.13. The row D is exact for the branch named by W, including the zero branch where the read remains inside finite inspected equality.

Finally apply Definition 1602.21. The ultrametric row U is downstream of S, B, W, D, R and therefore reads the same finite radius data as R. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricUltrametricWindowObligations`

Theorem 1602.8 (BaireMetric complete-metric handoff)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* BaireMetric$^{\uparrow}$ *carrier. A* CompleteMetric$^{\uparrow}$ *consumer can read M only as the finite prefix-window surface*

$$S, B, W, D, R, U, H, C, P, N.$$

The handoff exposes scheduled prefix agreement, first-disagreement radii, parent metric compatibility, and ultrametric strong-triangle data, but it does not expose a completed stream, quotient of Baire names, host real-valued metric, or countable-choice selector.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier complete metric handoff`

Proof. Start with the finite schedule supplied by Definition 1602.6. The consumer receives S, B, W before any dyadic or metric conclusion is public.

Use Definition 1602.20 and Definition 1602.7. These route R and U through the same displayed first-disagreement and dyadic-radius rows.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. Since the carrier has no coordinate outside the row family displayed in the statement, the complete-metric handoff remains finite prefix-window data. □

Theorem 1602.9 (BaireMetric obligation prefix completeness)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Every completeness-facing root obligation begins with a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S and a* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet B; it may inspect only the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window W and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row D before the parent* $\mathsf{Met}^{\uparrow}$ *row R is read. Thus a Cauchy or completeness consumer receives prefix completeness as displayed finite schedules and dyadic radius rows, not as completed streams, quotient equality of Baire names, or an ambient real-valued metric.*

Proof. Start with Definition 1602.8. It identifies the complete-metric consumer surface with the finite route $S, B, W, D, R, U, H, C, P, N$.

Use Definition 1602.6 and Definition 1602.13. The schedule and prefix packet display the inspected window, and the dyadic radius row is exact for that finite first-disagreement branch before the metric row is available.

The remaining rows U, H, C, P, N are downstream structural or ultrametric rows. Since no carrier coordinate stores completed streams or quotient equality, the prefix-completeness obligation is exhausted by the finite schedule, BaireSpace prefix packet, Nat window, and DyadicRatCore radius data. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier obligation prefix completeness`

Theorem 1602.10 (BaireMetric obligation ultrametric limit route)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Any ultrametric limit route exposed by M must keep each requested limit window on a finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, a* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet, a* $\mathsf{Nat}^{\uparrow}$ *first-disagreement or equality window, and the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row before it reaches R and U. The route is a finite sequence of displayed prefix-radius obligations, not a completed-stream object, quotient equality class, selected limit, or external topology.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier obligation ultrametric limit route`

Proof. Read the requested limit route as a family of finite distance comparisons. By Definition 1602.33, each displayed radius request determines a prefix thread through S, B, W, D.

Apply Definition 1602.7. The ultrametric strong-triangle row U consumes the same first-disagreement and dyadic-radius branch as the parent metric row R, so no separate limit coordinate can bypass the finite windows.

Finally use Definition 1602.29. It packages the consumer surface through R, U, H, C, P, N after the finite prefix-radius route, excluding completed streams, quotient names, selected limits, countable choice, and external topology. □

Theorem 1602.11 (BaireMetric obligation complete-metric consumer)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. A* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *consumer that reads a completeness obligation from M must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet,* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, then read the parent* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{UltrametricSpace}^{\uparrow}$ *handoff rows R, U, and only afterward pass through H, C, P, N. The consumer-facing obligation therefore contains no completed stream, quotient equality of names, host real metric, countable-choice selector, or external completeness theorem.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricObligationCompleteMetricConsumer`
↪ variant `BEDC.Derived.BaireMetricUp.BaireMetricCarrier obligation complete metric consumer`

Proof. Begin with prefix completeness, Definition 1602.9. It fixes the first four analytic rows available to a completeness consumer as S, B, W, D.

Next apply the ultrametric limit route theorem, Definition 1602.10. The parent metric and ultrametric rows are downstream of the same finite first-disagreement and dyadic-radius data, so the consumer cannot replace them by completed-stream equality or a selected limit.

Finally use Definition 1602.8. The structural rows H, C, P, N transport, replay, source, and name exactly the displayed route. Since the accepted carrier has no coordinate beyond $B, W, D, R, U, S, H, C, P, N$, the complete-metric consumer obligation is exactly the finite BEDC surface listed in the statement. □

Theorem 1602.12 (BaireMetric StreamName NameCert scope)**.** *For every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S *controls the local* $\mathsf{NameCert}_{\mathsf{BaireMetric}^{\uparrow}}$ *scope: every distance, ultrametric, or complete-metric read must pass through* S *before* $B, W, D, R, U, H, C, P, N$ *are inspected. Thus the certificate scope is a scheduled finite-prefix surface, not an equality theory of completed Baire names.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricStreamNameNameCertScope`

Proof. Project the carrier of Definition 1602.1. The schedule row is the only coordinate authorizing finite prefix inspection by a consumer.

Apply Definition 1602.22. It blocks every distance, metric, and ultrametric conclusion that tries to bypass the displayed schedule.

Then use Definition 1602.8. The complete-metric consumer receives only the scheduled finite row family, so the local NameCert scope contains no completed-stream equality route. □

Theorem 1602.13 (BaireMetric first-disagreement dyadic exactness)**.** *For an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, the dyadic radius row* D *is exact for the displayed first-disagreement window. If* W *records first disagreement at* k*, then* D *records the radius branch* 2^{-k}*; if* W *records only displayed equality across the inspected window, then* D *records the zero-radius branch for that finite read. No other radius source is available to the packet.*

Proof. The carrier admits dyadic radius data only through the row D. The row is downstream of S, B, W, so its positive branch is read from the first displayed disagreement already named by W.

In the equality branch, W exposes only the inspected finite equality window. The corresponding radius branch in D is therefore the displayed zero-radius read for that window, not a claim about completed infinite stream equality.

Because R and U are downstream consumers of D, neither can introduce an independent radius value. Exactness is consequently a projection of the single dyadic row attached to the first-disagreement window. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricFirstDisagreementDyadicRadiusExactness`

Theorem 1602.14 (BaireMetric prefix radius monotonicity)**.** *Let* $M = (B, W, D, R, U, S, H, C, P, N)$ *be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Extending the displayed inspected prefix through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule can only preserve the dyadic radius branch already read from* W, D *or move it to a smaller displayed dyadic branch. The extension cannot replace the first-disagreement route by a larger radius, a host metric, or a completed-stream comparison.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier prefix radius monotonicity`

Proof. Use carrier admission from Definition 1602.1 and the NameCert obligations of Definition 1602.2. Every distance read is routed through the scheduled prefix packet S, B and the finite window W before the dyadic row D is inspected.

By prefix-window admission in Definition 1602.6, an extended read only adds inspected coordinates after the displayed window already exposed. If the earlier first disagreement remains visible, first-disagreement exactness in Definition 1602.13 keeps the same 2^{-k} branch.

If the extension exposes a later first displayed disagreement after an equality branch, the dyadic branch read from D moves to that later index and is therefore no larger than the previous displayed prefix bound. Since R and U consume the same D row, no downstream metric or ultrametric consumer can enlarge the radius. □

Theorem 1602.15 (BaireMetric root ultrametric cylinder carrier)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *root carrier. The root cylinder surface of M is the finite* BHist *route in which the* $\mathsf{StreamName}^{\uparrow}$ *schedule S reads the prefix packet B, the first-disagreement window W reads that scheduled prefix, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row D reads W, the parent* $\mathsf{Met}^{\uparrow}$ *row R reads D, and the* $\mathsf{UltrametricSpace}^{\uparrow}$ *row U reads the same radius before the structural rows* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *are exposed through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootUltrametricCylinderCarrier`

Proof. Project the carrier of Definition 1602.1. The tuple $M = (B, W, D, R, U, S, H, C, P, N)$ lists the prefix packet, finite window, dyadic radius, metric handoff, ultrametric handoff, stream schedule, transport, replay, provenance, and local naming rows as the only available coordinates.

Use prefix-window admission and prefix radius monotonicity. Together they force every root cylinder read to pass through S, B, W, D before R or U can be consumed, and they keep every extension of the cylinder inside the same finite prefix schedule.

Finally read the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve the same order, so no root cylinder carrier can replace the displayed BHist packet by completed streams, quotient names, or a host metric. □

Theorem 1602.16 (BaireMetric root complete-metric nonescape)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *root carrier. Any* $\mathsf{CompleteMetric}^{\uparrow}$*-facing read from M must remain inside the displayed* BHist *route*

$$S, B, W, D, R, U, H, C, P, N.$$

It reads $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UltrametricSpace}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows from that tuple, and it has no coordinate for completed Baire streams, host completeness, quotient equality, or countable-choice selectors.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier root complete metric nonescape`

Proof. Start from a complete-metric handoff request. Carrier projection leaves only the tuple $M = (B, W, D, R, U, S, H, C, P, N)$, so every requested metric or completeness-facing row must be replayed through those displayed coordinates.

Apply Definition 1602.15. It routes the request through the $\mathsf{StreamName}^{\uparrow}$ schedule, finite prefix packet, first-disagreement window, $\mathsf{DyadicRatCore}^{\uparrow}$ radius, parent $\mathsf{Met}^{\uparrow}$ handoff, and $\mathsf{UltrametricSpace}^{\uparrow}$ strong-triangle row before any structural row is exported.

The structural rows H, C, P, N then preserve and name the same finite BHist route through hsame, Cont, Pkg, and NameCert. Since no coordinate outside the tuple is available, the complete-metric consumer cannot escape to completed streams, host completeness, quotient equality, or countable choice. □

Theorem 1602.17 (BaireMetric root StreamName prefix route)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* BaireMetric$^{\uparrow}$ *root carrier. Every root prefix read begins with the* StreamName$^{\uparrow}$ *schedule S and then consumes $B, W, D, R, U, H, C, P, N$ in the displayed* BHist *route. In particular, the* DyadicRatCore$^{\uparrow}$*,* Met$^{\uparrow}$*,* UltrametricSpace$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows are downstream of that scheduled prefix read, not independent root data.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootStreamNamePrefixRoute`

Proof. Read the prefix request through the tuple $M = (B, W, D, R, U, S, H, C, P, N)$. The schedule row S is the only row that can select finite prefixes before the Baire prefix packet B and first-disagreement window W are inspected.

Use Definition 1602.6 and Definition 1602.13. They place D after the scheduled window and ensure that each radius branch is read from the displayed finite prefix evidence.

Then apply Definition 1602.15. The parent Met$^{\uparrow}$ row R, UltrametricSpace$^{\uparrow}$ row U, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row are all downstream of the scheduled prefix route, so none can introduce an unscheduled prefix comparison. □

Theorem 1602.18 (BaireMetric zero-radius branch confinement)**.** *For an accepted* BaireMetric$^{\uparrow}$ *carrier $M = (B, W, D, R, U, S, H, C, P, N)$, the zero-radius branch of D is confined to displayed finite inspected-window equality. It is never a quotient equality of completed Baire streams, a global extensional equality theorem, or an ambient topological identification.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricZeroRadiusBranch`

Proof. Project the zero branch through the carrier of Definition 1602.1. The only equality evidence before D is the displayed finite equality row in W, scheduled by S and admitted through B.

Apply the NameCert obligations of Definition 1602.2. The prefix-distance row requires every zero-radius read to remain inside the inspected equality window, and the ledger-refusal row excludes completed streams, quotient equality of Baire names, host real metrics, and external topology.

Finally use Definition 1602.13. Its equality branch identifies the zero-radius read with that finite window alone. Transport by H and replay by C preserve the same row family and cannot turn finite inspected equality into equality of completed streams. □

Theorem 1602.19 (BaireMetric prefix radius transport)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ and $M' = (B', W', D', R', U', S', H', C', P', N')$ be accepted* BaireMetric$^{\uparrow}$ *carriers. If componentwise* hsame *transports the displayed Baire prefix packets, stream schedules, first-disagreement windows, dyadic rows, parent metric rows, ultrametric rows, provenance, and local names, and the displayed* Cont *replay preserves the prefix-read order, then M' exposes the same route*

$$S, B, W, D, R, U$$

as M for every accepted prefix-radius read.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricPrefixRadiusTransport`

Proof. Start from the accepted read in M. Prefix-window admission supplies the schedule, prefix packet, first-disagreement window, and dyadic row before any parent metric or ultrametric row is consumed.

Transport each displayed coordinate through the componentwise hsame rows recorded in H and H'. The carrier definition gives matching coordinates S', B', W', D', R', U' and the NameCert obligations require those coordinates to remain the only distance, metric, and ultrametric sources.

Replay the transported read through C'. First-disagreement exactness then reads D' from the transported W' branch, while the parent handoff and ultrametric handoff consume R' and U' downstream of that same branch. Thus the transported packet preserves the route and admits no alternate radius source. □

Theorem 1602.20 (BaireMetric parent MetricSpace handoff)**.** *Every parent* $\mathsf{Met}^{\uparrow}$ *read exported by an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *packet factors through*

$$S, B, W, D \longmapsto R.$$

The zero-distance boundary, symmetry row, and ordinary triangle route in R may read only the scheduled prefix packet, first-disagreement window, and dyadic radius row.

Proof. Start with a parent metric read. Carrier admission exposes the parent row only as R, and the definition places R downstream of the scheduled prefix and dyadic comparison rows.

The zero-distance, symmetry, and ordinary triangle components of R therefore receive their data from S, B, W, D. Componentwise transport by H and replay by C preserve this order, so the handoff cannot inspect a host metric on completed streams. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier parent metricspace handoff`

Theorem 1602.21 (BaireMetric UltrametricSpace strong-triangle handoff)**.** *Every* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle read exported by an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier factors through the same first-disagreement and dyadic-radius rows used by the parent metric handoff:*

$$S, B, W, D, R \longmapsto U.$$

The strong max-radius comparison is a finite prefix-window comparison, not a separate topology or completed-stream theorem.

Proof. The ultrametric witness appears in the carrier only as U. It is listed after the scheduled prefix packet, first-disagreement window, dyadic radius row, and parent metric row.

Thus a strong-triangle read must first replay the finite prefix comparisons through S, B, W, D and the parent handoff R. The row U then packages the max-radius comparison over those displayed radii, with no coordinate for external topology or completed Baire names. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricUltrametricSpaceStrongTriangleHandoff`

Theorem 1602.22 (BaireMetric stream-schedule nonescape)**.** *For every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier $M = (B, W, D, R, U, S, H, C, P, N)$, a consumer cannot read a distance, metric, or ultrametric conclusion outside the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule. Every such conclusion is replayed by C through S before it reaches $B, W, D, R,$ or U.*

Proof. The stream schedule S is the carrier coordinate controlling which finite prefixes may be inspected. The continuation row C names the replay order from a requested comparison through S before any BaireSpace, window, radius, metric, or ultrametric row is consumed.

Transport, provenance, and local naming preserve that replay order. Consequently a consumer has no slot from which to read an unscheduled prefix, completed stream, quotient equality of names, or external topological neighborhood. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricStreamScheduleNonescape`

Theorem 1602.23 (BaireMetric ultrametric prefix route)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *packet. Every accepted prefix-distance comparison exported by M factors through the displayed route*

$$B, S, W, D, R, U.$$

Equivalently, a consumer reads first the $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet, then the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the parent* $\mathsf{Met}^{\uparrow}$ *row, and finally the* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle witness. The route is a finite BEDC replay over prefix rows, not a host metric on completed streams.*

Proof. Start with an accepted prefix-distance comparison exported by M. Carrier admission exposes the comparison only through the coordinates B, S, W, D, R, U together with the structural rows.

The continuation row C first schedules finite prefix reads through S and then reads the accepted $\mathsf{BaireSpace}^{\uparrow}$ packet B. The $\mathsf{Nat}^{\uparrow}$ window W records how far the comparison has inspected the two prefixes and, when the positive branch is taken, which displayed coordinate is the first disagreement.

The dyadic radius row D converts that window into the displayed radius comparison. The parent $\mathsf{Met}^{\uparrow}$ row R receives only this radius evidence when supplying zero-distance, symmetry, and ordinary triangle data.

Finally the $\mathsf{UltrametricSpace}^{\uparrow}$ witness U reads the same first-disagreement and radius rows to supply the strong max-radius route. Componentwise transport by H and provenance by P preserve this order, so no consumer-facing path bypasses the displayed BaireSpace, Nat, StreamName, DyadicRatCore, MetricSpace, and UltrametricSpace rows. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCarrier ultrametric prefix route`

Theorem 1602.24 (BaireMetric complete-ultrametric consumer readiness)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *packet. Every* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{UltrametricSpace}^{\uparrow}$*, or* $\mathsf{BaireSpace}^{\uparrow}$ *consumer of M must first read the displayed route*

$$S, B, W, D, R, U, H, C, P, N.$$

The route schedules $\mathsf{StreamName}^{\uparrow}$ *reads before* $\mathsf{BaireSpace}^{\uparrow}$ *prefix rows, reads the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, reads the parent* $\mathsf{Met}^{\uparrow}$ *row before the* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle row, and reads transport, replay, provenance, and local naming before exporting any consumer-facing conclusion.*

Proof. Apply Definition 1602.6. It places the $\mathsf{StreamName}^{\uparrow}$ schedule S before the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet B and the $\mathsf{Nat}^{\uparrow}$ first-disagreement window W for every accepted distance read.

Apply Definition 1602.13. The dyadic radius row D is read only after the displayed first-disagreement window, so a downstream consumer cannot obtain a radius without the finite prefix-window evidence.

Apply Definition 1602.20 and Definition 1602.21. They order the parent $\mathsf{Met}^{\uparrow}$ row R before the $\mathsf{UltrametricSpace}^{\uparrow}$ strong-triangle row U, with both rows using the same scheduled prefix, window, and radius data.

Finally apply Definition 1602.23. It records that the consumer-facing prefix-distance route is finite and that componentwise transport H, replay C, provenance P, and local naming N preserve the same order before any $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{UltrametricSpace}^{\uparrow}$, or $\mathsf{BaireSpace}^{\uparrow}$ conclusion is exported. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCompleteUltrametricConsumerReadiness`

Theorem 1602.25 (BaireMetric root prefix schedule)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted $\mathsf{BaireMetric}^{\uparrow}$ root packet. The first read of the root packet exposes the finite prefix schedule*

$$S, B, W, D.$$

Thus the consumer first reads the $\mathsf{StreamName}^{\uparrow}$ schedule row S, the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet B, the $\mathsf{Nat}^{\uparrow}$ first-disagreement window W, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius row D before it may inspect the parent metric row, the strong-triangle row, transport, replay, provenance, or local naming.

Proof. Apply Definition 1602.6. It places every accepted distance read behind the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, and the finite first-disagreement window.

Apply Definition 1602.13. The dyadic radius row is determined only after that window is available, including the zero branch where the inspected prefixes remain equal on the displayed finite window.

Finally use Definition 1602.24. The complete-ultrametric route keeps the same prefix schedule as the initial root read and admits no earlier access to the metric, ultrametric, transport, replay, provenance, or local naming rows. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootPrefixSchedule`

Theorem 1602.26 (BaireMetric root strong-triangle route)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted $\mathsf{BaireMetric}^{\uparrow}$ root packet. Every $\mathsf{UltrametricSpace}^{\uparrow}$ consumer of the strong-triangle row U must first read the same finite prefix route*

$$S, B, W, D, R.$$

Only after this route has supplied the stream schedule, Baire prefix packet, Nat first-disagreement window, dyadic radius row, and parent metric row may the consumer read U as the strong-triangle witness.

Proof. Apply Definition 1602.25. It fixes the initial root read as the finite S, B, W, D schedule, so the strong-triangle consumer cannot start from completed streams, quotient names, or an external topology.

Next apply Definition 1602.20. The parent metric row R receives exactly the scheduled prefix, first-disagreement, and dyadic-radius data before exporting ordinary metric compatibility.

Finally apply Definition 1602.21. The strong-triangle row U reads the same finite prefix and radius route after R, so the ultrametric consumer cannot replace the displayed root schedule by a separate strong-triangle path. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootStrongTriangleRoute`

Theorem 1602.27 (BaireMetric root prefix ultrametric package)**.** *Every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *root package exposes exactly the route through* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{UltrametricSpace}^{\uparrow}$, *together with the structural rows* BHist, hsame, Cont, Pkg, *and* NameCert, *before any PolishSpace, BaireSpace, or ultrametric consumer reads it. In the notation of Definition 1602.1, the package surface is the finite row family*

$$B, S, W, D, R, U, H, C, P, N.$$

No consumer-facing read may replace this row family by completed streams, quotient equality of names, host real metrics, countable choice, or external topology.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricRootPrefixUltrametricPackage`

Proof. Apply Definition 1602.6. It forces every distance read to expose the $\mathsf{StreamName}^{\uparrow}$ schedule S, the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet B, and the $\mathsf{Nat}^{\uparrow}$ first-disagreement window W before any radius or metric row is inspected.

Apply Definition 1602.13. The dyadic radius row D is then determined by the displayed first-disagreement window, with the zero-radius branch remaining inside the inspected finite equality window.

Apply Definition 1602.20 and Definition 1602.21. The parent $\mathsf{Met}^{\uparrow}$ row R and the $\mathsf{UltrametricSpace}^{\uparrow}$ row U both read the same scheduled prefix, window, and dyadic-radius data.

Finally use Definition 1602.22 and Definition 1602.23. They place every metric, ultrametric, PolishSpace, BaireSpace, or downstream prefix consumer behind the same replay row C, with componentwise transport H, provenance P, and local naming N. Since the carrier has no coordinate for completed streams, quotient equality of names, host real metrics, countable choice, or external topology, the displayed root package is the whole exported surface. □

Theorem 1602.28 (BaireMetric StreamName ultrametric obligation)**.** *Let* $M = (B, W, D, R, U, S, H, C, P, N)$ *be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Every consumer that reads a complete-ultrametric or* $\mathsf{BaireSpace}^{\uparrow}$ *conclusion from* M *must first consume the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet* B*, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D*, the parent* $\mathsf{Met}^{\uparrow}$ *row* R*, the* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle witness* U*, and the structural rows* H, C, P, N *for* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming. Consequently the obligation surface is the finite route*

$$S, B, W, D, R, U, H, C, P, N,$$

and it exports no completed streams, quotient names, host real metric, countable choice, or external topology.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricStreamNameUltrametricObligation`

Proof. Apply Definition 1602.6. It places every distance read behind the displayed $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, and the finite first-disagreement window.

Apply Definition 1602.13. The dyadic radius row is therefore read from the finite window, with the zero branch remaining inside inspected prefix equality.

Next use Definition 1602.20 and Definition 1602.21. The parent metric row and the strong-triangle witness consume the same schedule, prefix packet, window, and dyadic radius data.

Finally apply Definition 1602.27. It packages exactly the displayed route with componentwise transport, replay, provenance, and local naming. Since no carrier coordinate supplies completed streams, quotient names, host real metrics, countable choice, or external topology, no complete-ultrametric or BaireSpace consumer can read such data from M. □

Theorem 1602.29 (BaireMetric complete-ultrametric consumer obligations)**.** *Let $M = (B, W, D, R, U, S, H, C, P,$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Every complete ultrametric consumer of M must display the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet B,* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{Nat}^{\uparrow}$ *first-disagreement window W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row D, parent* $\mathsf{Met}^{\uparrow}$ *row R,* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle row U, transport row H, replay row C, provenance row P, and local naming row N before it may read a* $\mathsf{CompleteMetric}^{\uparrow}$ *conclusion. No complete-ultrametric obligation is stored outside this row family.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCompleteUltrametricConsumerObligations`

Proof. Project the carrier. The rows $B, W, D, R, U, S, H, C, P, N$ are the only displayed coordinates available to a complete-ultrametric consumer.

Apply Definition 1602.6 and Definition 1602.13. They force every distance read through the stream schedule, Baire prefix packet, finite Nat window, and dyadic radius row.

Apply Definition 1602.21 and Definition 1602.28. The first theorem orders the strong-triangle row after the same prefix and radius data, while the second packages the complete-ultrametric consumer surface with the structural transport, replay, provenance, and local naming rows. □

Theorem 1602.30 (BaireMetric Cauchy-window completeness handoff)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Any* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff read from M receives its Cauchy-window evidence only through the finite prefix route*

$$S, B, W, D, R, U, H, C, P, N.$$

Thus completeness consumers may replay finite prefix agreement and dyadic-radius decay, but they cannot read completed streams, quotient names, host real metrics, or countable-choice selectors from the carrier.

Proof. Start from a handoff request to $\mathsf{CompleteMetric}^{\uparrow}$. Carrier projection leaves only the displayed prefix, window, radius, parent metric, ultrametric, and structural rows.

Use Definition 1602.25. It fixes the initial Cauchy-window read as the finite S, B, W, D route, so the handoff cannot begin from a completed stream.

Next apply Definition 1602.20 and Definition 1602.21. They pass the same window and radius data through R and U before any completeness-facing metric conclusion is exposed.

Finally use Definition 1602.29. The structural rows H, C, P, N replay and name exactly that route, so the handoff remains a finite Cauchy-window obligation. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCauchyWindowCompletenessHandoff`

Theorem 1602.31 (BaireMetric BaireSpace prefix-density route)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Every* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-density consumer of the complete ultrametric surface must pass through the same finite prefix schedule S, B, W, D before reading R, U, H, C, P, N. The route is therefore density over displayed finite prefixes, not a density theorem for completed Baire names.*

Proof. Project M and read the $\mathsf{BaireSpace}^{\uparrow}$ row B together with the $\mathsf{StreamName}^{\uparrow}$ schedule S. These rows are the only prefix source offered by the carrier.

Apply Definition 1602.6. It places all Baire prefix reads behind the finite schedule and first-disagreement window, with no quotient equality of names available.

Apply Definition 1602.30. The completeness-facing route uses the same $S, B, W, D, R, U, H, C, P, N$ row family, so a prefix-density consumer cannot bypass the finite window when reading the complete ultrametric surface. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricBaireSpacePrefixDensityRoute`

Theorem 1602.32 (BaireMetric prefix-cylinder coverage)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Every prefix-cylinder read exported by M is covered by the finite route*

$$S, B, W, D.$$

The $\mathsf{StreamName}^{\uparrow}$ *schedule selects the inspected prefix window, the* $\mathsf{BaireSpace}^{\uparrow}$ *row supplies the displayed cylinder base, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement row records the inspected length and branch, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row records the matching radius. No cylinder coverage claim may use completed streams, quotient equality of Baire names, or an external topology.*

Proof. Project the carrier. The only coordinates that can name a prefix cylinder before the metric rows are S, B, W, D.

The schedule S determines which finite prefixes are visible. The row B supplies the accepted BaireSpace prefix packet, and W records the finite length and first-disagreement branch inside that packet.

The radius row D then reads the branch named by W. Since R, U, H, C, P, N are downstream of this finite window route, no exported prefix-cylinder coverage can bypass S, B, W, D. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricPrefixCylinderCoverage`

Theorem 1602.33 (BaireMetric Cauchy prefix thread)**.** *For an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier $M = (B, W, D, R, U, S, H, C, P, N)$, every* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy read determines a finite prefix thread through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule. At each displayed radius request, the thread reads a scheduled* $\mathsf{BaireSpace}^{\uparrow}$ *prefix window, a* $\mathsf{Nat}^{\uparrow}$ *first-disagreement or equality branch, the corresponding* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, and then the parent* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{UltrametricSpace}^{\uparrow}$ *rows. The thread is not a completed stream and does not use countable choice.*

Proof. Start with a completeness-facing Cauchy request. By Definition 1602.30, the request sees only the finite route $S, B, W, D, R, U, H, C, P, N$.

For each displayed radius, the schedule S opens the relevant finite prefix window in B. The row W records whether the inspected prefixes agree on that window or where the first displayed disagreement lies, and D converts that finite branch into the dyadic radius read.

The parent metric row R and the ultrametric row U consume the same window and radius data. Transport, replay, provenance, and naming keep the finite thread attached to M, so the Cauchy read never becomes a selected infinite stream. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCauchyPrefixThread`

Definition 1602.34 (BaireMetric Cauchy-thread obligation surface)**.** *The* $\mathsf{BaireMetric}^{\uparrow}$ *Cauchy-thread obligation surface is the finite row inventory*

$$(S, B, W, D, R, U, H, C, P, N)$$

read by every completeness-facing consumer after Definition 1602.33: prefix-window admission through S, B, first-disagreement or equality admission through W, dyadic radius exactness through D, parent $\mathsf{Met}^{\uparrow}$ *handoff through R,* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle handoff through U, and structural preservation through H, C, P, N. It explicitly has no coordinate for completed-stream equality, quotient Baire names, host real metrics, countable choice, or an external completeness theorem.*

Lean4 (definition) `BEDC.Derived.BaireMetricUp.BaireMetricCauchyThreadObligationSurface`

Theorem 1602.35 (BaireMetric prefix Cauchy-thread completeness)**.** *For an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, every prefix Cauchy read that reaches a complete-ultrametric consumer is exhausted by the obligation surface of Definition 1602.34. The read factors through prefix-window admission, first-disagreement dyadic radius exactness, the parent* $\mathsf{Met}^{\uparrow}$ *handoff, the* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle handoff, and the complete-ultrametric exhaustion row before any limit-window conclusion is exported.*

Proof. Start with the Cauchy read supplied by Definition 1602.33. It already fixes each displayed radius request to a finite $\mathsf{StreamName}^{\uparrow}$ schedule, a $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, a $\mathsf{Nat}^{\uparrow}$ first-disagreement or equality branch, and a $\mathsf{DyadicRatCore}^{\uparrow}$ radius row.

The metric and ultrametric handoffs consume exactly that data. Apply Definition 1602.20 for the parent metric row R, and apply Definition 1602.21 for the strong-triangle row U.

Finally use Definition 1602.37. It says the complete-ultrametric surface has no remaining slot outside prefix coverage, Cauchy threading, the two handoffs, and H, C, P, N. Thus the prefix Cauchy read is complete at the displayed finite obligation surface and exports no completed stream, quotient name, host metric, countable-choice selector, or external completeness theorem. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricPrefixCauchyThreadCompleteness`

Theorem 1602.36 (BaireMetric prefix limit-window uniqueness)**.** *For an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, two limit-window reads that agree on the finite Cauchy-thread obligation surface of Definition 1602.34 agree as* $\mathsf{BaireMetric}^{\uparrow}$ *limit windows for every displayed prefix and dyadic radius. The uniqueness claim is window-local: it refuses completed-stream equality, quotient Baire-name equality, host real metrics, countable choice, and any external topological completeness principle.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricPrefixLimitWindowUniqueness`

Proof. Fix two limit-window reads and compare their displayed prefix requests. By Definition 1602.32, every such request is routed through S, B, W, D, so agreement on the obligation surface gives the same schedule, prefix packet, branch, and dyadic radius row.

The handoff rows preserve this window agreement. The parent metric row R and ultrametric row U read only those displayed prefix and radius rows, and their compatibility is fixed by Definition 1602.20 and Definition 1602.21.

Use Definition 1602.37 to close the route. Since all complete-ultrametric obligations are already exhausted by the finite prefix thread and structural rows H, C, P, N, there is no hidden coordinate where the two reads can diverge as limit windows. The conclusion is therefore finite-window uniqueness, not completed-stream equality or quotient equality. □

Theorem 1602.37 (BaireMetric complete-ultrametric exhaustion)**.** *Every complete-ultrametric obligation exported by an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$ *is exhausted by prefix-cylinder coverage, the Cauchy prefix thread, parent metric compatibility, ultrametric strong-triangle compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. There is no remaining obligation slot for a host real-valued metric, completed-stream equality, quotient Baire names, countable choice, or external topological completeness.*

Proof. Use Definition 1602.32 to identify every cylinder read with the finite S, B, W, D route. This covers the prefix side of the complete-ultrametric surface.

Use Definition 1602.33 for the completeness-facing Cauchy reads. It keeps every displayed radius request on the same StreamName schedule, BaireSpace prefix packet, Nat window, and dyadic radius row.

Finally apply Definition 1602.20 and Definition 1602.21. The rows R and U consume exactly that finite prefix data, while H, C, P, N transport, replay, source, and name it. The listed rows exhaust the carrier, so no external completeness or host metric datum remains. □

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCompleteUltrametricExhaustion`

Theorem 1602.38 (BaireMetric public prefix package)**.** *Every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *prefix-distance packet exports to* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{BaireSpace}^{\uparrow}$ *consumers only the finite public package*

$$W, \quad S, \quad B, \quad D, \quad R, \quad U, \quad H, \quad C, \quad P, \quad N.$$

Here W records first-disagreement or equality data, S records the $\mathsf{StreamName}^{\uparrow}$ *prefix schedule, B records the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-window packet, D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, R records the parent* $\mathsf{Met}^{\uparrow}$ *distance row, and U records the* $\mathsf{UltrametricSpace}^{\uparrow}$ *strong-triangle row. No public prefix-distance read exports completed streams, quotient Baire names, host real metrics, countable-choice selectors, or an external completeness theorem.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricObligationPublicPrefixPackage`

Proof. Start with prefix-cylinder coverage. It identifies each public prefix request with the finite S, B, W, D route, so the first-disagreement branch, prefix schedule, BaireSpace window, and dyadic radius are already displayed before any consumer reads the metric row.

Use first-disagreement dyadic exactness through the existing prefix-window and radius route. The dyadic row D is read from the branch recorded by W, not from a host real-valued metric or a quotient-name comparison.

Finally apply the parent metric handoff and the ultrametric strong-triangle handoff. The rows R and U consume only the same prefix and radius data, while H, C, P, N transport, replay, source, and name the package. Complete-metric and Baire-space consumers therefore receive the finite public package and no hidden completion or choice datum. □

Theorem 1602.39 (BaireMetric bridge route)**.** *Every bridge-facing* $\mathsf{BaireMetric}^{\uparrow}$ *consumer reads the finite prefix-distance route*

$$S, \quad B, \quad W, \quad D, \quad R, \quad U, \quad H, \quad C, \quad P, \quad N$$

before any standard metric interpretation is exposed. The route first displays the $\mathsf{StreamName}^{\uparrow}$ *schedule, then the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, the parent* $\mathsf{Met}^{\uparrow}$ *handoff, the* $\mathsf{UltrametricSpace}^{\uparrow}$ *witness, transport, replay, provenance, and the local* NameCert *row. Thus a bridge consumer receives the same finite prefix ledger named by the public package, not completed streams, quotient Baire names, selected limits, or an ambient topology.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricBridgeRoute`

Proof. Begin with the public prefix package of Definition 1602.38. It exposes the consumer-facing order of $S, B, W, D, R, U, H, C, P, N$ and keeps the metric read behind the finite prefix ledger.

Use Definition 1602.44 to identify this order with the whole paper-side obligation surface. The first-disagreement window and dyadic radius row are already displayed before the parent metric and ultrametric rows are available.

Finally apply the finite refinement control of Definition 1602.45. Any bridge-facing refinement stays on the same prefix-distance route, so the standard metric interpretation can only be read after the displayed rows have been consumed in that order. □

Theorem 1602.40 (BaireMetric formal target scope)**.** *Every* $\mathsf{BaireMetric}^{\uparrow}$ *consumer that reads a metric comparison must read it through the displayed* $\mathsf{BaireSpace}^{\uparrow}$ *stream row* B*,* $\mathsf{Nat}^{\uparrow}$ *finite-prefix row* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison row* U*,* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. In the carrier notation* $M = (B, W, D, R, U, S, H, C, P, N)$*, the metric comparison route is therefore*

$$S, \quad B, \quad W, \quad D, \quad R, \quad U, \quad H, \quad C, \quad P, \quad N.$$

It exposes no completed stream, quotient Baire-name equality, host real-valued metric, countable-choice selector, or external topology.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricFormalTargetScope`

Proof. Project the carrier of Definition 1602.1. The displayed route contains the stream schedule, BaireSpace prefix packet, Nat first-disagreement window, dyadic tolerance row, parent metric row, ultrametric comparison row, and structural transport, replay, provenance, and naming rows.

Use the NameCert obligations and the ultrametric prefix route, Definition 1602.2 and Definition 1602.23. They force each comparison through finite prefix admission, dyadic radius exactness, metric compatibility, ultrametric compatibility, and ledger refusal before any consumer-facing read is exported.

Finally apply complete-ultrametric exhaustion and the public prefix package theorem. Those results identify the CompleteMetric-facing handoff with the same $S, B, W, D, R, U, H, C, P, N$ route, so the formal target is exactly this finite carrier scope and not an external completion, topology, or host metric assertion. □

Theorem 1602.41 (BaireMetric ultrametric cylinder obligation)**.** *For every accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier* $M = (B, W, D, R, U, S, H, C, P, N)$*, each prefix-cylinder distance exported to a* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{UltrametricSpace}^{\uparrow}$ *consumer factors through the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row* B*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement or equality row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D*, and then the parent metric and ultrametric rows* R, U*. The route is preserved by the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows* H, C, P, N *and exports no completed stream, quotient Baire name, host real-valued metric, or external topology.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricUltrametricCylinderObligation`

Proof. Project the carrier of Definition 1602.1. The prefix-cylinder read has only the displayed window rows S, B, W, D before it reaches the parent metric and ultrametric rows.

Induct on the finite prefix window recorded by W. The equality branch keeps the dyadic radius at the displayed zero-or-tail row, while the first-disagreement branch exposes the first inspected coordinate and sends that coordinate to the $\mathsf{DyadicRatCore}^{\uparrow}$ radius row D.

The consumer rows cannot bypass this prefix thread. The metric row R and ultrametric row U are downstream from the same S, B, W, D route, and H, C, P, N transport, replay, source, and name that route without adding a second distance coordinate. □

Theorem 1602.42 (BaireMetric complete metric readback)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer reading an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier reads the same finite Cauchy prefix thread* $S, B, W, D, R, U, H, C, P, N$

already displayed by the prefix-cylinder obligation. The consumer may compare finite Cauchy windows and dyadic radii, but it receives no quotient Baire name, no selected infinite branch, no choice selector, no host stream equality, and no external completeness principle.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCompleteMetricReadback`

Proof. Begin with the Cauchy prefix thread of Definition 1602.33. It binds every completeness-facing radius request to a finite stream schedule, a Baire prefix row, a first-disagreement or equality row, and a dyadic radius row.

The complete metric readback uses the same carrier rows. Apply the parent metric handoff and ultrametric strong-triangle handoff to route the read through R and U after the prefix rows have been exposed.

Finally use complete-ultrametric exhaustion. Since H, C, P, N already account for transport, replay, provenance, and local naming, no quotient name, selector, host stream equality, or external completeness datum remains for a $\mathsf{CompleteMetric}^{\uparrow}$ consumer. □

Theorem 1602.43 (BaireMetric StreamName comeager ledger)**.** *The public comeager-style route available from an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *prefix schedule* S*, the* $\mathsf{BaireSpace}^{\uparrow}$ *dense-window ledger* B*, the first-disagreement row* W*, and the dyadic radius row* D*, together with* R, U, H, C, P, N*. Any dense-window or cylinder-consumer read must therefore remain inside the displayed finite prefix ledger and cannot import external topology, a completed Baire space, countable choice, or a quotient-open-set algebra.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricStreamNameComeagerLedger`

Proof. Read the prefix schedule first. The row S lists the inspected coordinates, and the row B records the BaireSpace window whose cylinder is being named.

Use finite prefix induction over that schedule. At each step, either the inspected coordinate is still common and the next prefix row is exposed, or W records the first disagreement and D records the corresponding dyadic radius.

The metric, ultrametric, and structural rows only package this ledger. Rows R, U, H, C, P, N pass the finite cylinder route to public consumers, so no external topology or completed Baire-space object is introduced. □

Theorem 1602.44 (BaireMetric obligation-closure route)**.** *Let* $M = (B, W, D, R, U, S, H, C, P, N)$ *be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. The paper-side obligation surface for* M *is exhausted by the finite route consisting of prefix-window admission, first-disagreement dyadic exactness, ultrametric window obligations, complete-metric handoff, StreamName schedule control, and the comeager-style ledger rows. Equivalently, every obligation-facing consumer of* M *reads only*

$$S, \quad B, \quad W, \quad D, \quad R, \quad U, \quad H, \quad C, \quad P, \quad N,$$

and no obligation row exports completed streams, quotient equality of Baire names, host real-valued metrics, countable-choice selectors, external topology, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricObligationClosureRoute`

Proof. Begin with prefix-window admission and dyadic exactness. The prefix-window theorem places every distance request behind the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, and the $\mathsf{Nat}^{\uparrow}$ first-disagreement row, while first-disagreement dyadic exactness forces the radius row D to be read from the branch recorded by W.

Next add the ultrametric and complete-metric routes. The ultrametric window-obligation theorem makes U consume the same W, D branch as the parent metric row R, and the complete-metric handoff theorem keeps completeness-facing consumers on the finite surface $S, B, W, D, R, U, H, C, P, N$.

Finally use the StreamName scope theorem and the comeager ledger. They ensure that dense-window, cylinder, metric, ultrametric, and completeness reads all pass through the displayed schedule and prefix ledger before any public conclusion is available. The structural rows transport, replay, source, and name that same finite packet, so the excluded completed-stream, quotient, choice, external-topology, and ambient-completeness data have no carrier coordinate. □

Theorem 1602.45 (BaireMetric cylindrical-refinement induction)**.** *Let $M = (B, W, D, R, U, S, H, C, P, N)$ be an accepted* $\mathsf{BaireMetric}^{\uparrow}$ *carrier. Any finite refinement of a Baire cylinder prefix schedule preserves the displayed route through*

$$S, \quad B, \quad W, \quad D, \quad R, \quad U, \quad H, \quad C, \quad P, \quad N.$$

For each added coordinate, the $\mathsf{StreamName}^{\uparrow}$ *schedule S inspects the coordinate, the* $\mathsf{Nat}^{\uparrow}$ *first-disagreement row W compares it with the current prefix window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D assigns the corresponding dyadic radius, and the result is passed to the parent* $\mathsf{Met}^{\uparrow}$ *row R and* $\mathsf{UltrametricSpace}^{\uparrow}$ *witness U through the same transport, replay, provenance, and local naming rows. No refinement step adds completed streams, quotient Baire names, countable-choice selectors, external topology, or an ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.BaireMetricUp.BaireMetricCylindricalRefinementInduction`

Proof. Proceed by primitive induction on the finite refinement spine. In the empty case, the route is exactly the prefix-window admission and first-disagreement exactness route already exposed by Definition 1602.44.

For the step case, append one coordinate to the inspected prefix schedule. The schedule row S exposes the new coordinate, the prefix-window admission theorem keeps the read inside the displayed $\mathsf{BaireSpace}^{\uparrow}$ cylinder packet, and the first-disagreement exactness theorem sends the comparison to W.

The dyadic radius is then read from D, after which the ultrametric prefix route passes the same branch to R and U. The structural rows H, C, P, N transport, replay, source, and name the refined packet without changing the carrier inventory.

Thus each finite refinement step preserves the same displayed dependency route, and induction over the refinement spine excludes completed streams, quotient Baire names, countable-choice selectors, external topology, and ambient completeness data from every refined cylinder read. □

Falsifiable prediction: Every accepted $\mathsf{BaireMetric}^{\uparrow}$ distance comparison factors through the $\mathsf{BaireSpace}^{\uparrow}$ prefix packet, $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{Nat}^{\uparrow}$ first-disagreement window, $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, parent $\mathsf{Met}^{\uparrow}$ row, $\mathsf{UltrametricSpace}^{\uparrow}$ witness, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any consumer reads a metric or ultrametric conclusion.

Independence witness: bairespace, streamname, nat, dyadicratcore, metricspace, ultrametricspace: $\mathsf{BaireMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries prefix windows, first-disagreement evidence, dyadic radii, parent metric compatibility, ultrametric strong-triangle evidence, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireMetric}^{\uparrow}}$.

Constructive story: A BaireMetric$^{\uparrow}$ packet is generated from a BaireSpace prefix packet, StreamName schedule, Nat first-disagreement window, DyadicRatCore radius row, parent MetricSpace row, UltrametricSpace witness, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireMetric}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1602.1, Definition 1602.2, Definition 1602.23, Definition 1602.38, and Definition 1602.44 expose the public finite prefix ultrametric interface on BaireSpace$^{\uparrow}$ through first-disagreement windows and dyadic radius rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed streams, quotient equality of Baire names, host real-valued metrics, countable-choice selectors, external topology, completeness, category arguments, or Lean verification.

Upgrade path: No paper-side theory upgrade remains beyond future Lean formalization and bridge checking.

Chapter 1603

A Concrete Naming Certificate for $\mathsf{SchurOrthogonality}^{\uparrow}$

Starting from Schur's orthogonality relations for finite-group characters, the $\mathsf{SchurOrthogonality}^{\uparrow}$ packet records the row and column orthogonality ledger as finite BEDC readback data. It refines the character ledger of chapter 245 and supplies a concrete downstream source for $\mathsf{RepresentationRing}^{\uparrow}$ and $\mathsf{InducedRep}^{\uparrow}$ without importing host character tables, quotient conjugacy-class carriers, or complete representation categories.

Definition 1603.1 (SchurOrthogonality carrier)**.** *A* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, V, X, \Phi, \Omega, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row for the finite group endpoint.*
2. *V is the consumed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row for the representation-space endpoints that carry the character reads.*
3. *X is the consumed* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *row, restricted to trace and class-function readback data already exposed by the character-theory packet.*
4. *Φ is the finite character-pairing ledger whose entries name the displayed pairings of character rows before any representation-ring or induced-representation consumer reads them.*
5. *Ω is the row and column orthogonality ledger, containing only finite rows for the displayed Schur row relations, Schur column relations, normalization reads, and zero-off-diagonal reads.*
6. *H records componentwise* hsame *transport for the group, vector-space, character-theory, pairing, orthogonality, continuation, provenance, and naming rows.*
7. *C records displayed* Cont *replay from G, V, X into Φ, from Φ into Ω, and from Ω into the downstream* $\mathsf{RepresentationRing}^{\uparrow}$ *and* $\mathsf{InducedRep}^{\uparrow}$ *readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{SchurOrthogonality}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}$ *row.*

The carrier exports only this finite orthogonality ledger. It contains no host character table, no quotient carrier of conjugacy classes, no ambient inner-product space, and no complete category of representations.

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityTasteGate single carrier alignment`

Theorem 1603.2 (SchurOrthogonality NameCert obligations)**.** *For every accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier* $S = (G, V, X, \Phi, \Omega, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed group, vector-space, character-theory, pairing, orthogonality, transport, continuation, provenance, and naming rows;*
2. *character-pairing route, requiring every orthogonality read to pass through the finite pairing ledger* Φ *before it is consumed by* $\mathsf{RepresentationRing}^{\uparrow}$ *or* $\mathsf{InducedRep}^{\uparrow}$*;*
3. *row and column orthogonality ledger, recording the displayed Schur row relations, Schur column relations, normalization reads, and zero-off-diagonal reads in* Ω*;*
4. *transport stability, requiring* hsame *and* Cont *replay to preserve the same displayed pairing and orthogonality entries under accepted source transport;*
5. *no-host-quotient boundary, excluding character-table computation, quotient conjugacy-class carriers, host inner products, and complete representation categories.*

No local obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1603.1. The only available data are the consumed G, V, X rows, the pairing ledger Φ, the orthogonality ledger Ω, and the structural rows H, C, P, N.

The character-pairing route is forced by the row order: downstream consumers read orthogonality only after the relevant character rows have been paired in Φ. The row and column relations are then read from Ω, whose entries are displayed finite ledger rows rather than host sums over a character table.

Transport stability is the role of H and C. They move accepted source rows componentwise and replay the same path from G, V, X through Φ into Ω.

Finally, the provenance and naming rows bind exactly this finite packet. Since no coordinate stores a host character table, quotient conjugacy-class carrier, host inner product, or representation category, such data remain outside the local NameCert surface. □

Theorem 1603.3 (SchurOrthogonality character-pairing route)**.** *Let* $S = (G, V, X, \Phi, \Omega, H, C, P, N)$ *be an accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier. Every orthogonality read exported from* S *to a* $\mathsf{RepresentationRing}^{\uparrow}$ *or* $\mathsf{InducedRep}^{\uparrow}$ *consumer factors through the displayed route*

$$G, V, X \longmapsto \Phi \longmapsto \Omega \longmapsto C, P, N.$$

Thus no consumer can read a Schur row relation, Schur column relation, normalization row, or zero-off-diagonal row before the corresponding character-pairing entry has appeared in Φ*.*

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityCharacterPairingRoute`

Proof. Start with a downstream orthogonality request. The group, vector-space, and character-theory sources available to the request are exactly the displayed rows G, V, X.

The request first enters the character-pairing ledger Φ. Only after that pairing entry is present can the continuation route reach the orthogonality ledger Ω and select the corresponding row relation, column relation, normalization row, or zero-off-diagonal row.

The structural rows C, P, N then expose that selected entry to the downstream consumer. Since every exported entry has this route as its displayed source, there is no direct read from a host character table, quotient class carrier, or ambient inner product. □

Theorem 1603.4 (SchurOrthogonality character inner-product ledger)**.** *Let $S = (G, V, X, \Phi, \Omega, H, C, P, N)$ be an accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier. Every character-inner-product read exposed by S is a finite ledger read from Φ over the displayed* $\mathsf{Group}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, *and* $\mathsf{VecSpace}^{\uparrow}$ *rows before any orthogonality row of Ω is selected. No exported inner-product read may bypass $G, V, X, \Phi, H, C, P, N$ or import an ambient Hilbert-space structure.*

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityCharacterInnerProduct`

Proof. Project the carrier. The group endpoint is G, the character source is X, and the representation-space endpoint is V.

The only row that pairs characters before orthogonality is Φ. Thus a character-inner-product request first reads the displayed group, character-theory, and vector-space rows, and then selects the corresponding finite pairing entry of Φ.

The orthogonality ledger Ω is downstream from this pairing row. It can consume the selected finite entry, but it cannot create a separate ambient inner product or read from an undeclared character table. The structural rows H, C, P, N transport, replay, source, and name the same finite route. □

Theorem 1603.5 (SchurOrthogonality representation-ring route)**.** *Let $S = (G, V, X, \Phi, \Omega, H, C, P, N)$ be an accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier. A* $\mathsf{RepresentationRing}^{\uparrow}$ *or* $\mathsf{InducedRep}^{\uparrow}$ *consumer may read only the displayed Schur ledger exported through Φ, Ω, C, P, N. Such a consumer cannot import host character tables, quotient conjugacy-class carriers, ambient character inner products, or a complete representation category as hidden sources for the same read.*

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityRepresentationRingRoute`

Proof. Begin with a downstream $\mathsf{RepresentationRing}^{\uparrow}$ or $\mathsf{InducedRep}^{\uparrow}$ request. By Definition 1603.3, every exported orthogonality read has already passed through G, V, X into Φ and then into Ω.

The continuation and provenance rows expose only this displayed ledger. They do not contain an independent coordinate for host character tables, quotient conjugacy classes, ambient inner products, or a complete representation category.

Therefore the downstream consumer receives the finite Schur ledger as named BEDC data. Any route requiring one of the excluded host objects is outside the accepted carrier. □

Theorem 1603.6 (SchurOrthogonality group-averaging ledger)**.** *Let $S = (G, V, X, \Phi, \Omega, H, C, P, N)$ be an accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier. The finite group-averaging row displayed inside Ω is the only averaging source available to Schur row relations, Schur column relations, normalization reads, and zero-off-diagonal reads. This row is preserved by componentwise* hsame *transport in H, displayed* Cont *replay in C,* Pkg *provenance in P, and the local* $\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityGroupAveragingLedger`

Proof. Inspect the orthogonality ledger Ω. Its displayed entries contain the finite averaging data used by the Schur row and column reads, together with normalization and zero-off-diagonal rows.

Transport acts componentwise through H. The averaging entry is moved with the same group, vector-space, character-theory, pairing, and orthogonality rows, so the transported packet still names the same finite source.

Replay and provenance then preserve this source. The continuation row C can only replay the displayed route, P can only cite its package provenance, and N names the local certificate. Hence no second averaging source is introduced. □

Theorem 1603.7 (SchurOrthogonality row and column orthogonality ledger)**.** *Let $S = (G, V, X, \Phi, \Omega, H, C, P, N)$ be an accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier. The row relations, column relations, normalization rows, and zero-off-diagonal rows exported by S are exposed only through $G, V, X, \Phi, \Omega, H, C, P, N$. A* $\mathsf{RepresentationRing}^{\uparrow}$ *or* $\mathsf{InducedRep}^{\uparrow}$ *consumer may read these rows only through the displayed finite pairing and orthogonality ledger, and cannot bypass the packet by importing a host character table, quotient conjugacy-class carrier, ambient inner product, or complete representation category.*

Proof. The carrier in Definition 1603.1 has only the displayed source rows G, V, X, the pairing ledger Φ, the orthogonality ledger Ω, and the structural rows H, C, P, N.

By Definition 1603.3, every exported orthogonality read first passes through the character-pairing ledger. By Definition 1603.4, the corresponding pairing is already a finite ledger read before any row of Ω is selected.

The downstream route is then governed by Definition 1603.5 and Definition 1603.6. These force $\mathsf{RepresentationRing}^{\uparrow}$ and $\mathsf{InducedRep}^{\uparrow}$ consumers to receive only the finite Schur row, column, normalization, and zero-off-diagonal entries named by Ω and replayed by C.

Since no coordinate of S stores a host character table, quotient conjugacy-class carrier, ambient inner product, or complete representation category, such objects cannot serve as alternate sources for the same consumer read. □

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityRowColumnOrthogonalityLedger`

Theorem 1603.8 (SchurOrthogonality scoped dependency package)**.** *For every accepted* $\mathsf{SchurOrthogonality}^{\uparrow}$ *carrier $S = (G, V, X, \Phi, \Omega, H, C, P, N)$, the carrier row, NameCert obligation surface, character-pairing route, character inner-product ledger, representation-ring route, group-averaging ledger, and row-column orthogonality ledger bind the finite scope over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

Proof. Use Definition 1603.1 to project the displayed rows $G, V, X, \Phi, \Omega, H, C, P, N$. The NameCert obligations of Definition 1603.2 admit exactly those rows as the local Schur orthogonality surface.

The character-pairing and inner-product routes are fixed by Definition 1603.3 and Definition 1603.4. They force every orthogonality read through G, V, X, Φ before the orthogonality ledger Ω is consumed.

The downstream package is then governed by Definition 1603.5, Definition 1603.6, and Definition 1603.7. These theorems expose only Ω, H, C, P, N as the finite route into $\mathsf{RepresentationRing}^{\uparrow}$ and $\mathsf{InducedRep}^{\uparrow}$ consumers, so the stated dependency package is the full scoped surface. □

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityScopedDependencyPackage`

Theorem 1603.9 (SchurOrthogonality character-basis public export)**.** *Every public* $\mathsf{SchurOrthogonality}^{\uparrow}$ *read exported to a* $\mathsf{RepresentationRing}^{\uparrow}$ *or* $\mathsf{InducedRep}^{\uparrow}$ *consumer is the finite character-basis ledger exposed through $G, V, X, \Phi, \Omega, H, C, P, N$. The public surface contains the row and column orthogonality entries, normalization rows, zero-off-diagonal rows, and character-pairing provenance named by the scoped dependency package, and it contains no host character table, quotient conjugacy-class carrier, ambient inner product, or complete representation category.*

Proof. Project the scoped dependency package of Definition 1603.8. It binds the carrier, pairing ledger, orthogonality ledger, transport, continuation, provenance, and NameCert rows as the whole visible surface.

The public consumer route is already fixed by Definition 1603.5. A $\mathsf{RepresentationRing}^{\uparrow}$ or $\mathsf{InducedRep}^{\uparrow}$ consumer reads only the displayed finite Schur ledger through Φ, Ω, C, P, N.

The row-column ledger of Definition 1603.7 supplies the exported character-basis entries. Since the carrier has no coordinate for the excluded host structures, the public read is exactly the finite character-basis ledger named in the statement. □

Lean4 ✓ `BEDC.Derived.SchurOrthogonalityUp.SchurOrthogonalityPublicCharacterBasis`

Closure status for $\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1603.8 and Definition 1603.9 bind the finite Schur orthogonality route over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{InducedRep}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows and expose its public character-basis ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchurOrthogonality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close character-table computation, quotient conjugacy classes, host inner products, complete representation categories, Lean verification, or bridge checking.

Upgrade path: The remaining work is Lean verification and bridge checking for the same finite Schur orthogonality packet.

Constructive story: A $\mathsf{SchurOrthogonality}^{\uparrow}$ packet is generated from displayed Group, VecSpace, and CharacterTheory sources, a finite character-pairing ledger, a finite row and column orthogonality ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1604

A Concrete Naming Certificate for $\mathsf{BaireOneFunction}^{\uparrow}$

Following Baire's function hierarchy and Bourbaki-style real analysis, the $\mathsf{BaireOneFunction}^{\uparrow}$ packet records pointwise limits of continuous real-valued maps as finite BEDC ledgers rather than as host function-space objects. It reads continuous approximants through chapter 130, indexes their displayed pointwise stream through chapter 126, checks finite rational readbacks through chapter 301, seals observed values through chapter 38, keeps the source context visible through chapter 334, routes the pointwise-limit handoff through chapter 2314, and exposes only a handoff suitable for lower-semicontinuous consumers in chapter 1536. The packet is therefore a certificate for a named pointwise-limit route, not a space of all functions and not a choice of approximating sequence outside the displayed ledger.

Definition 1604.1 (BaireOneFunction carrier)**.** *A* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, S, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{BaireSpace}^{\uparrow}$ *source-context row naming the inspected source points and the finite basic-open information available to the packet.*
2. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows $f_n : X \to \mathsf{Real}^{\uparrow}$ whose pointwise observations are being replayed.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting, for each displayed source point and precision request, the finite list of approximant indices inspected by the packet.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected real-value windows of the approximants.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after X, F, S, Q have been displayed.*
6. *L is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *and lower-semicontinuous handoff row recording the finite pointwise-limit observations and epigraph-facing observations that downstream* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumers may read from the ledger.*
7. *H records componentwise* hsame *transport for the source context, continuous approximant ledger, stream schedule, regular rational readback, real value seal, lower-semicontinuous handoff, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a source-point request through the continuous approximant ledger, the selected stream windows, the regular rational readback, the real value seal, and the lower-semicontinuous handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireOneFunction}^{\uparrow}}$ *row.*

The carrier exports no ambient function space, no arbitrary choice of approximating sequence, no quotient equality of real names, no full measurability theorem, and no completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.TasteGate.BaireOneFunctionTasteGate single carrier alignment`

Definition 1604.2 (BaireOneFunction classifier)**.** *Two accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carriers*

$$K = (X, F, S, Q, R, L, H, C, P, N) \quad \textit{and} \quad K' = (X', F', S', Q', R', L', H', C', P', N')$$

are classified together precisely when the displayed hsame *rows in H and H' identify the source-context rows, continuous-approximant ledgers, stream schedules, regular rational readbacks, real value seals, lower-semicontinuous handoff rows, continuation routes, provenance rows, and local naming rows componentwise. The classifier has no coordinate for extensional equality of host functions, no quotient of real names, and no comparison of undisplayed approximants.*

Lean4 (definition) `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionClassifier`
Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionClassifier component namecert transport`

Theorem 1604.3 (BaireOneFunction NameCert obligation surface)**.** *For every accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier $K = (X, F, S, Q, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireOneFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source-context row, continuous approximant ledger, stream schedule, regular rational readback, real value seal, lower-semicontinuous handoff, transport, continuation, provenance, and local naming rows;*

2. *continuous-approximant admission, requiring every displayed approximant read to pass through one row of F over the source context X;*

3. *pointwise-limit scheduling, requiring every value-seal read to factor through the selected windows in S and the corresponding readbacks in Q;*

4. *real-seal non-escape, requiring R to be downstream of X, F, S, Q and not of an ambient function space;*

5. *pointwise-limit and lower-semicontinuous handoff compatibility, requiring L to consume only the same displayed pointwise-limit ledger;*

6. *ledger refusal, excluding arbitrary approximating-sequence choice, quotient equality of real names, full measurability, ambient function-space topology, and completeness of* $\mathsf{Real}^{\uparrow}$.

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier namecert obligation surface`

Proof. Carrier admission is projection from Definition 1604.1. The accepted packet exposes exactly the source context, continuous approximant ledger, stream schedule, regular rational readback, real value seal, lower-semicontinuous handoff, transport, continuation, provenance, and local naming rows.

Continuous-approximant admission is read from F over X. Each displayed approximant contributes only its $\mathsf{ContinuousMap}^{\uparrow}$ row at the source points named by the packet, so no host function space or unlisted approximant can enter the certificate.

Pointwise-limit scheduling is then read through S and Q. A value request first chooses a displayed source point and precision window, then follows the finite stream schedule to the approximant indices recorded by S, and finally reads the regular rational windows recorded by Q.

The real value seal R is downstream of this route. Since R is reached only after the displayed approximant windows and rational readbacks have been replayed, it cannot export quotient equality of real names, a semantic function-space limit, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

The pointwise-limit and lower-semicontinuous row L is another downstream read of the same ledger. The componentwise transports and continuation routes in H and C preserve the route X, F, S, Q, R, L, while P and N record no excluded coordinate. Thus the listed rows exhaust the local NameCert surface. □

Theorem 1604.4 (BaireOneFunction pointwise-limit ledger)**.** *Every* $\mathsf{Real}^{\uparrow}$ *value exported by an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$X, F, S, Q \longmapsto R.$$

The route reads a source point from the $\mathsf{BaireSpace}^{\uparrow}$ *context, selects finitely many continuous approximant observations through the* $\mathsf{StreamName}^{\uparrow}$ *schedule, checks the selected windows through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and then exposes the* $\mathsf{Real}^{\uparrow}$ *value seal. It is not an ambient pointwise-convergence theorem, not a quotient of real names, and not a choice principle for approximating sequences.*

Proof. Start with a consumer of the value seal R. The carrier places R in the same packet as X, F, S, Q, and the continuation row C records the only displayed route into R.

The source request first reads the $\mathsf{BaireSpace}^{\uparrow}$ context X. The continuous approximant ledger F then supplies the relevant $\mathsf{ContinuousMap}^{\uparrow}$ rows over that context, while the stream schedule S selects the finite approximant windows currently being inspected.

The regular rational readback Q checks those selected windows before the seal row is available. Componentwise transport by H preserves this route, and the package and local naming rows record no ambient function-space coordinate. Therefore every exported value seal factors through X, F, S, Q. □

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier pointwise lowersemicontinuous handoff certificate`

Theorem 1604.5 (BaireOneFunction pointwise schedule obligations)**.** *For every accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier* $K = (X, F, S, Q, R, L, H, C, P, N)$*, every pointwise value read must expose the* $\mathsf{BaireSpace}^{\uparrow}$ *source row* X*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* R*, and the transport, replay, provenance, and local naming rows* H, C, P, N *before any* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *handoff row* L *is consumed. Thus the handoff cannot be read as a shortcut around the pointwise schedule.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier pointwise schedule obligations`

Proof. Project the carrier of Definition 1604.1. The rows X, F, S, Q, R appear before the lower-semicontinuous handoff row L in the displayed pointwise-limit ledger.

Apply Definition 1604.4. It forces every exported $\mathsf{Real}^{\uparrow}$ value through the source context, continuous approximant family, stream schedule, and regular rational readback before the value seal is available.

The handoff theorem Definition 1604.8 then consumes the same ledger. Componentwise transport H, replay C, provenance P, and local naming N preserve the route, so no $\mathsf{LowerSemicontinuous}^{\uparrow}$ consumer may bypass X, F, S, Q, R. □

Theorem 1604.6 (BaireOneFunction oscillation locality)**.** *Every oscillation or point-continuity consumer of an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier* $K = (X, F, S, Q, R, L, H, C, P, N)$ *factors through the displayed pointwise schedule, the* $\mathsf{Real}^{\uparrow}$ *observation rows, the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *source rows, and a finite precision window before reading lower-semicontinuous handoff or public-export data. Thus an oscillation read may inspect only the finite source context, selected approximant windows, regular-rational readbacks, real value seals, componentwise transport, replay, provenance, and local naming already visible in the carrier.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier oscillation locality`

Proof. Project the carrier and classifier rows of Definition 1604.1 and Definition 1604.2. The source row X, approximant ledger F, schedule row S, regular-rational readback Q, real seal R, transport H, replay C, provenance P, and local naming N are the rows available before any downstream handoff is read.

Use the pointwise-limit ledger of Definition 1604.4. It forces every $\mathsf{Real}^{\uparrow}$ value read through the route $X, F, S, Q \longmapsto R$, so an oscillation or point-continuity request must first expose the displayed source point, the selected continuous approximant rows, and the finite regular-rational windows used to seal the observed value.

Now apply the pointwise schedule obligations of Definition 1604.5. They keep the lower-semicontinuous handoff L downstream of the same pointwise schedule and prevent it from serving as a shortcut around $X, F, S, Q, R, H, C, P, N$.

The $\mathsf{Met}^{\uparrow}$ or $\mathsf{ContinuousMap}^{\uparrow}$ source rows enter only through the displayed source context and approximant ledger. Since the carrier has no coordinate for ambient function spaces, quotient real equality, hidden approximants, or an unrestricted continuity theorem, the requested oscillation or point-continuity read is local to the finite observation window already exposed by the packet. □

Theorem 1604.7 (BaireOneFunction pointwise oscillation handoff)**.** *Every pointwise oscillation consumer of an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier* $K = (X, F, S, Q, R, L, H, C, P, N)$ *must read the* $\mathsf{BaireSpace}^{\uparrow}$ *source row* X*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant ledger* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *value seal* R*, and the finite precision window determined by these rows before any lower-semicontinuous or oscillation-rank handoff is consumed. The handoff therefore exports a pointwise finite-observation route, not a host oscillation-rank theorem, hidden approximant selector, quotient of real names, or ambient function-space completeness principle.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier pointwise oscillation handoff`

Proof. Project the accepted carrier of Definition 1604.1. The source row X, approximant ledger F, stream schedule S, regular rational readback Q, value seal R, and structural rows H, C, P, N are displayed before a downstream lower-semicontinuous or oscillation-rank consumer can read any handoff row.

Apply Definition 1604.4. It forces every pointwise value request through the source context, selected continuous approximant observations, stream schedule, and regular rational readback before the $\mathsf{Real}^{\uparrow}$ seal is available.

Next use Definition 1604.5. The schedule obligations keep L downstream of the same pointwise ledger, so a lower-semicontinuous handoff cannot replace X, F, S, Q, R by an ambient convergence or measurability claim.

Finally apply Definition 1604.6. Oscillation reads are local to the finite source context, selected approximant windows, regular-rational readbacks, value seals, transport, replay, provenance, and local naming already visible in the carrier. Hence any oscillation-rank or lower-semicontinuous handoff must pass through the stated pointwise finite-observation route. □

Theorem 1604.8 (BaireOneFunction lower-semicontinuous handoff)**.** *Every* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer row exported by an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier reads only the displayed pointwise-limit ledger*

$$X, F, S, Q, R \longmapsto L.$$

The handoff may use finite epigraph-facing observations derived from the same source context, continuous approximant rows, stream schedule, regular rational readback, and real value seal. It receives no full measurability theorem, no host function-space topology, no arbitrary approximant selector, and no completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier lowersemicontinuous handoff`

Proof. Let a downstream consumer request the row L. By carrier admission, L is a coordinate of the same packet that contains the source context X, the approximant ledger F, the schedule S, the readback Q, and the value seal R.

Replay the continuation row C. The consumer can reach L only after the source context is read, the displayed continuous approximants are inspected, the selected stream windows are checked by the regular rational readback, and the real value seal is exposed.

Transport by H keeps the lower-semicontinuous handoff attached to those same rows. Since the provenance and local naming rows carry no measurability theorem, no ambient function-space topology, no arbitrary approximant selector, and no $\mathsf{Real}^{\uparrow}$ completeness coordinate, the handoff exports none of them. □

Theorem 1604.9 (BaireOneFunction lower-semicontinuous exhaustion)**.** *Every* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *public row exported by an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *carrier*

$$K = (X, F, S, Q, R, L, H, C, P, N)$$

is exhausted by the displayed source context X*, continuous approximant ledger* F*, stream schedule* S*, regular rational readback* Q*, real value seal* R*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*. The lower-semicontinuous row* L *may publish only the public observations already routed through these rows; it cannot introduce hidden approximants, quotient real equality, full measurability, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier lowersemicontinuous exhaustion`

Proof. Start with the pointwise ledger of Definition 1604.4. It binds every value observation to the route $X, F, S, Q \longmapsto R$, so a downstream lower-semicontinuous consumer cannot read a value that is detached from the displayed source context, approximant family, schedule, and readback.

Next apply the schedule obligations of Definition 1604.5. They force $X, F, S, Q, R, H, C, P, N$ to remain visible before L is consumed, and they rule out using the handoff as a shortcut around the finite pointwise schedule.

Finally use the lower-semicontinuous handoff of Definition 1604.8. The handoff row L is downstream of the same pointwise ledger, while H, C, P, N only transport, replay, source, and name the displayed packet. Therefore the public lower-semicontinuous export is exhausted by the stated rows and has no coordinate for hidden approximants, quotient real equality, full measurability, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1604.10 (BaireOneFunction public export)**.** *Every public* $\mathsf{BaireOneFunction}^{\uparrow}$ *consumer receives exactly the displayed route*

$$X, F, S, Q, R, L, H, C, P, N.$$

The public surface keeps the $\mathsf{BaireSpace}^{\uparrow}$ *source row,* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant ledger,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *handoff, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows visible. It exports no ambient function space, arbitrary approximant choice, quotient real equality, full measurability theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier public export`

Proof. Project the carrier of Definition 1604.1. Its public coordinates are exactly $X, F, S, Q, R, L, H, C, P, N$.

Apply the pointwise schedule obligations and lower-semicontinuous handoff, Definition 1604.5 and Definition 1604.8. These force every value and handoff read through the displayed BaireSpace, ContinuousMap, StreamName, RegSeqRat, Real, and LowerSemicontinuous rows before a public consumer can use the packet.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for ambient function spaces, arbitrary approximant selection, quotient real equality, full measurability, or $\mathsf{Real}^{\uparrow}$ completeness, none of those principles is part of the public export. □

Theorem 1604.11 (BaireOneFunction finite schedule nonescape)**.** *Every pointwise-limit or lower-semicontinuous consumer of an accepted* $\mathsf{BaireOneFunction}^{\uparrow}$ *packet must pass through the displayed finite schedule rows*

$$X, F, S, Q, R, L, H, C, P, N.$$

The route first exposes the $\mathsf{BaireSpace}^{\uparrow}$ *source row* X*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant ledger* F*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* R*, and the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *handoff* L*, then preserves the read through transport* H*, replay* C*, provenance* P*, and local naming* N*. It cannot introduce an undisplayed approximant selector, ambient function-space equality, quotient real equality, measurability theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireOneFunctionUp.BaireOneFunctionCarrier finite schedule nonescape`

Proof. Start with the pointwise-limit ledger of Definition 1604.4. It binds every value read to the displayed source row, approximant ledger, finite schedule, regular rational readback, and real seal.

Apply the pointwise schedule obligations of Definition 1604.5. They force the stream schedule S and readback row Q to remain visible before any pointwise consumer may inspect the value row.

Finally use the lower-semicontinuous handoff and public export, Definition 1604.8 and Definition 1604.10. The structural rows H, C, P, N preserve and name the same finite packet, so no hidden selector, ambient equality, quotient equality, measurability theorem, or $\mathsf{Real}^{\uparrow}$ completeness can enter the consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireOneFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireOneFunction}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1604.1, Definition 1604.2, Definition 1604.3, Definition 1604.4, Definition 1604.8, and Definition 1604.10 bind the finite pointwise-limit packet from continuous real-valued approximants to lower-semicontinuous public consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireOneFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function spaces, arbitrary choice of approximating sequence, quotient equality of real names, full measurability, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or a standard bridge.

Upgrade path: No paper-side theory upgrade remains; future work is Lean verification or bridge checking if not already recorded.

Constructive story: A $\mathsf{BaireOneFunction}^{\uparrow}$ packet is generated from a $\mathsf{BaireSpace}^{\uparrow}$ source context, a finite ledger of $\mathsf{ContinuousMap}^{\uparrow}$ approximants, $\mathsf{StreamName}^{\uparrow}$ schedule windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ value seal, a $\mathsf{LowerSemicontinuous}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1605

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Bourbaki–Kelley filter terminology, the $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ packet records the finite route by which the cofinite Frechet tail filter of a regular Cauchy source is read through Cauchy-filter basis rows, saturated Cauchy-filter evidence, stream windows, dyadic tolerances, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal. It refines the cofinite-tail filter surface of chapter 1564, the basis generator of chapter 1501, the saturation row of chapter 333, the filter-to-regular route of chapter 1445, the finite-window surface of chapter 126, the dyadic ledger of chapter 291, the regular rational interface of chapter 301, and the Real naming boundary of chapter 38. The packet is a finite tail-filter specialization; it is not a quotient filter, not a selected limit, not host real equality, not countable choice, and not an ambient completeness principle.

Definition 1605.1 (Regular Cauchy tail filter carrier)**.** *A* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, B, Q, S, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite-tail neighbourhood row, carrying the displayed tail threshold and finite-complement checks through which the regular Cauchy source is inspected.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row built from the same tail neighbourhoods, finite stream windows, regular readbacks, and dyadic tolerance requests.*
3. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturation row that consumes B before any filter-facing or Real-facing consumer can inspect the packet.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose late observations are admitted only after the Frechet tail row and basis row have been replayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to those late windows and to the Cauchy-filter basis inclusions.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from S, D after the saturated Cauchy-filter route F, B, Q has been displayed.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, B, Q, S, D, R have been replayed.*

8. *H records componentwise* hsame *transport for the Frechet tail row, basis row, saturation row, stream windows, dyadic tolerances, readback, Real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a tail-filter request through the Frechet tail neighbourhoods, CauchyFilterBasis generator, CauchyFilter saturation, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and then the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, host real equality, countable-choice tail selectors, quotient Cauchy names, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1605.2 (Regular Cauchy tail filter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier* $T = (F, B, Q, S, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Frechet tail row, CauchyFilterBasis generator, CauchyFilter saturation row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming row;*

2. *tail-neighbourhood exposure, requiring every accepted read to consume F before a stream window or basis element is inspected;*

3. *basis generation and saturation exactness, requiring B to be generated from the displayed tail windows and Q to consume that basis before any downstream filter route is exported;*

4. *regular readback order, requiring S and D to be replayed before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R is available;*

5. *Real seal boundary, requiring A to remain downstream of F, B, Q, S, D, R and refusing quotient filters, selected limits, host real equality, countable choice, quotient Cauchy names, and ambient completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1605.1. The visible rows are exactly $F, B, Q, S, D, R, A, H, C, P, N$.

The tail-neighbourhood obligation is forced by the replay row C. A request first consumes the cofinite-tail row F, then uses the same late-tail data to form the displayed basis row B.

Saturation exactness is the role of Q. Since Q is downstream of B, every filter-facing consumer receives the generated Cauchy-filter surface rather than a detached quotient filter.

The regular readback and Real boundary are then fixed by row order. The stream windows S and dyadic tolerances D feed the readback row R, and the seal row A is available only after R and the structural rows have been replayed. The excluded quotient, selector, host-equality, choice, and completeness objects are absent from the carrier, so the displayed rows exhaust the local obligation surface. □

Theorem 1605.3 (Regular Cauchy tail filter Real seal boundary)**.** *Let $T = (F, B, Q, S, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of T factors through the finite route*

$$F \longmapsto B \longmapsto Q \longmapsto S, D \longmapsto R \longmapsto A \longmapsto H, C, P, N.$$

Thus the consumer reads the Frechet tail neighbourhoods, generated Cauchy-filter basis, saturated Cauchy filter, finite stream windows, dyadic tolerances, and regular rational readback before reaching the Real seal. The exported surface contains no quotient filter, selected limit, host real equality, countable-choice selector, quotient Cauchy name, or ambient completeness assertion.

Proof. Start with a Real-facing or completion-facing consumer. Carrier admission exposes the seal row A, but only as a row downstream of the tail, basis, saturation, window, tolerance, and readback rows.

Replay the continuation row C. The route first reads F, generates B, saturates through Q, and then inspects the finite windows S with their dyadic ledger D.

The next public row is the regular rational readback R. Only after R has been replayed can the consumer read A.

Transport through H and provenance through P, N preserve the same route. Since the carrier contains no quotient-filter coordinate, selected limit, host real equality, countable-choice selector, quotient Cauchy name, or completeness witness, no Real-facing read can export one. □

Theorem 1605.4 (Regular Cauchy tail filter Real seal handoff)**.** *Let* $T = (F, B, Q, S, D, R, A, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier. Any* $\mathsf{DyadicTailBall}^{\uparrow}$ *or* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer using* T *must read the finite route*

$$\mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

in that order before exporting a Real-seal-facing conclusion.

The handoff is consumed by the downstream dyadic tail-ball route of chapter 1633 and by the Real Cauchy criterion route of chapter 1590. It cannot bypass through quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1605.1. The rows F, B, Q, S, D, R, A expose the Frechet tail filter, CauchyFilterBasis generator, CauchyFilter saturation, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and terminal Real seal.

Apply Definition 1605.3. It forces every Real-facing or completion-facing consumer to read F, then B, then Q, then S, D, then R, and only then A.

The $\mathsf{DyadicTailBall}^{\uparrow}$ consumer reads this same tail route before its dyadic ball containment row, while the $\mathsf{RealCauchyCriterion}^{\uparrow}$ consumer reads the same stream, readback, dyadic tolerance, and Real-seal surface before exposing its criterion row.

Since quotient stream equality, selected limits, countable choice, host real equality, trichotomy, and ambient completeness are absent from the carrier, neither downstream consumer can replace the displayed tail-filter route by one of those objects. □

Theorem 1605.5 (Real Cauchy criterion tail-filter handoff)**.** *Let* $T = (F, B, Q, S, D, R, A, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier. Every* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer of* T *must first read the tail-filter route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicTailBall}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it exports a Real Cauchy criterion conclusion. The handoff preserves the same stream, readback, dyadic tolerance, tail-ball, and Real-seal rows used by the local carrier and by chapter 1633.*

Proof. Project the accepted carrier of Definition 1605.1. Its consumer-facing rows display the Frechet tail row, CauchyFilterBasis generator, CauchyFilter saturation, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, terminal Real seal, transport, replay, provenance, and local naming.

Apply Definition 1605.4. It already places every $\mathsf{RealCauchyCriterion}^{\uparrow}$ consumer after the route through $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Now insert the dyadic tail-ball consumer route. The $\mathsf{DyadicTailBall}^{\uparrow}$ packet of chapter 1633 reads the same late stream windows, dyadic tolerance ledger, regular readback, and Real-seal surface before its containment row is public. Hence the Real Cauchy criterion consumer receives a named tail-filter handoff rather than an unnumbered prose dependency. □

Theorem 1605.6 (Regular Cauchy tail filter Bishop completion handoff)**.** *Let $T = (F, B, Q, S, D, R, A, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *carrier. Any* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *consumer that reads the tail-filter route must factor its request through*

$$F \longmapsto B \longmapsto Q \longmapsto S, D \longmapsto R \longmapsto A$$

before invoking the Bishop completion carrier rows of chapter 1639. The handoff consumes the displayed $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and terminal* $\mathsf{Real}^{\uparrow}$ *surface already exposed by T, then presents only the L10 completion rows governed by Definition 1639.2 and Definition 1639.15. It exports no quotient stream, selected limit, host real equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1605.1. A completion-facing read has access only to the rows $F, B, Q, S, D, R, A, H, C, P, N$, with F supplying tail exposure, B supplying the generated Cauchy-filter basis, and Q supplying saturation.

Apply Definition 1605.2. Tail exposure forces the consumer to read F before any stream window is inspected, while basis generation and saturation exactness force the route through B and Q before any filter-facing surface is available.

Apply Definition 1605.3. The only ordered Real-facing route then continues through the finite $\mathsf{StreamName}^{\uparrow}$ windows S, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal A.

The Bishop completion chapter can therefore receive the tail-filter data only as the displayed L10 rows. Its regularity envelope in Definition 1639.2 reads the finite stream, regular sequence, dyadic tolerance, common-tail, and Real-seal rows, while Definition 1639.15 prevents escape to quotient streams, selected limits, host real equality, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1605.1, Definition 1605.2, and Definition 1605.3 seed the finite route from Frechet tail neighbourhoods through CauchyFilterBasis generation, CauchyFilter saturation, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the downstream Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host real equality, countable-choice tail selectors, quotient Cauchy names, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-neighbourhood exposure, basis generation, CauchyFilter saturation exactness, RegSeqRat readback order, Real seal boundary, and ledger refusal proofs for the same packet.

Constructive story: A $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ packet is generated from a FrechetFilter tail row, a CauchyFilterBasis generator, a CauchyFilter saturation row, StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1606

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMinimality}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki and Kelley treatments of metric completion, the $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ packet records the minimality boundary for Cauchy completion through existing BEDC finite rows. It reads the finite completion carrier of chapter 1506, the dense isometric embedding surface of chapter 1512, the extension and uniqueness interface of chapter 397, the uniformly continuous extension handoff of chapter 769, and the Bishop-style constructive route of chapter 1556. Its dense-extension uniqueness and separated reflection rows also consume the complete-metric limit uniqueness route of chapter 5983 before the minimality packet exports uniqueness. The packet names the least completion-facing service boundary visible to later consumers: source metric data, dense embedding, universal extension, separated uniqueness, transport, provenance, and local naming.

Definition 1606.1 (CauchyCompletionMinimality carrier)**.** *A* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, C, E, U, F, S, H, R, P, N)$$

with the following visible rows:

1. *X is the source metric row together with the located source evidence used by the completion route.*
2. *C is the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *completion row over the same source histories.*
3. *E is the dense* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, including source-image coverage and distance reflection.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row that supplies the extension and uniqueness surface for completion-facing consumers.*
5. *F is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row that hands uniformly continuous dense-source maps to the completed packet.*
6. *S is the separated uniqueness row comparing two displayed completion reads only through zero-distance classifier evidence.*
7. *H records componentwise* hsame *transport for the source, completion, embedding, universal-property, extension, separatedness, replay, provenance, and local naming rows.*

8. *R records the displayed* Cont *replay through $X, C, E, U, F, S, H, P, N$, in the order in which a consumer may read the packet.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *row naming this packet and no larger completion principle.*

The carrier has no coordinate for quotient completions, selected representatives, host equality, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityTasteGate single carrier alignment`

Theorem 1606.2 (CauchyCompletionMinimality NameCert obligations)**.** *For an accepted* CauchyCompletionMinin *carrier $M = (X, C, E, U, F, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the source metric and located source row, displayed completion row, dense embedding row, universal-property row, uniformly continuous extension row, separated uniqueness row, componentwise transport, continuation replay, provenance, and local naming row;*

2. *dense-image coverage, requiring every completion-facing source approximation to pass through the embedding row E after the completion row C has been displayed;*

3. *universal factorization, requiring every completion-facing consumer to read the universal-property row U and extension handoff F before it can export a completed value;*

4. *separated uniqueness, requiring any comparison of two completed reads to pass through S and the componentwise transport ledger H;*

5. *ledger non-escape, stating that R, P, N expose no quotient completion, selected representative, host equality, countable-choice schedule, open-cover compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1606.1. The tuple displays exactly the source, completion, embedding, universal-property, extension, separatedness, transport, replay, provenance, and naming rows.

Dense-image coverage is read from E after C. The completion row gives the target-facing packet, and the embedding row supplies the source-image route through which every admitted approximation reaches that packet.

Universal factorization is the role of U and F. The universal-property row supplies extension and uniqueness data, while the uniformly continuous extension row records the finite handoff used by dense-source consumers.

Separated uniqueness is local to S and transported by H. The replay row R orders the same route, and the provenance rows P, N list no additional coordinate. Hence the excluded quotient, representative-selection, host-equality, choice, compactness, and ambient real-completeness principles cannot be read from the certificate. □

Theorem 1606.3 (CauchyCompletionMinimality dense-image coverage)**.** *Let $M = (X, C, E, U, F, S, H, R, P, N)$ be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Every completion-facing approximation exported by the minimality packet must expose the displayed completion row C and then the dense embedding row E before the universal-property, uniformly continuous extension, separated uniqueness, transport, replay, provenance, and local naming rows U, F, S, H, R, P, N can be consumed.*

The coverage route exports no quotient completion, selected representative, countable-choice schedule, host equality, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityDenseImageCoverage`

Proof. Project the carrier of Definition 1606.1. Its rows display the source metric X, completion row C, dense embedding row E, universal-property row U, extension row F, separated uniqueness row S, and structural rows H, R, P, N.

Apply the NameCert obligations of Definition 1606.2. The dense-image clause requires every completion-facing source approximation to pass through E after C has been displayed, while the ledger non-escape clause keeps quotient completions, selected representatives, choice schedules, host equality, open-cover compactness, and ambient real-completeness outside the packet.

Finally use Definition 1606.4. Since every downstream consumer must continue through U, F, S, H, R, P, N, the consumer cannot bypass E by reading the universal route directly. □

Theorem 1606.4 (CauchyCompletionMinimality universal factorization)**.** *Let $M = (X, C, E, U, F, S, H, R, P, N)$ be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Any completion-facing consumer sharing the same source metric X, dense embedding E, and separated target boundary S factors through the route*

$$X \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto (H, R, P, N).$$

The factorization exports no quotient completion, selected representative, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from the shared source metric row X. A completion-facing consumer cannot reach a completed read until the dense embedding row E has displayed the source image used by that read.

The universal row U is consumed next. It is the finite extension and uniqueness surface through which the consumer obtains a completion-facing map rather than a detached host-space operation.

The uniformly continuous handoff F then records the carried extension route. Its source and target sides are still the same rows displayed by X, E, and U.

Finally the separated row S compares any two resulting target reads, while H, R, P, N transport, replay, source, and name the same finite packet. Because none of the listed rows contains a quotient completion, selected representative, host equality, countable-choice schedule, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the factorization is minimal in exactly the displayed BEDC coordinates. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier universal factorization`

Theorem 1606.5 (CauchyCompletionMinimality reflector unit factorization)**.** *Let $M = (X, C, E, U, F, S, H, R, P,$ be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier whose source row is read through the* CauchyCompletionReflector *packet of chapter 8626. Then any minimality-facing consumer must expose the unit and dense-embedding rows before the universal factorization route*

$$X \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto (H, R, P, N).$$

Equivalently, the source read first passes through the reflector-unit carrier's source, unit embedding, reflector, dense-image, and triangle replay rows, and only then reaches the minimality carrier's dense embedding, universal-property, extension, separatedness, transport, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *rows.*

The factorization exports no quotient completion, selected representative, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the minimality carrier of Definition 1606.1. Its rows expose the source metric X, completion row C, dense embedding E, universal-property row U, extension row F, separated uniqueness row S, and structural rows H, R, P, N.

Now apply the local obligations of Definition 1606.2. Those obligations force dense-image coverage through E and universal factorization through U, F, S, H, R, P, N before any completion-facing consumer can export a result.

The universal factorization theorem Definition 1606.4 supplies the displayed route from X through E, U, F, S into the structural rows. Reading X through chapter 8626 adds only the finite unit embedding, dense-image, and triangle-replay rows before that same route.

Since neither the reflector-unit carrier nor the minimality carrier contains a quotient completion, selected representative, choice schedule, host equality row, or ambient Real-completeness theorem, the combined factorization remains inside the displayed BEDC packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityReflectorUnitFactorization`

Theorem 1606.6 (CauchyCompletionMinimality dense-extension uniqueness)**.** *Let* $M = (X, C, E, U, F, S, H, R, P$ *be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. If two completion-facing consumers share the same source metric row* X*, dense embedding row* E*, universal-property row* U*, uniformly continuous extension handoff* F*, and separated target boundary* S*, then their exported completed reads are identified only through the separated uniqueness row* S *and the shared transport, replay, provenance, and local naming rows* H, R, P, N*.*

The identification exports no quotient completion, selected representative, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityDenseExtensionUniqueness`

Proof. Apply Definition 1606.2. The accepted carrier exposes the source metric, completion, dense embedding, universal-property, extension, separatedness, transport, replay, provenance, and local naming rows as the complete local certificate surface.

Next use Definition 1606.4. Since the two consumers share X, E, U, F, S, both completion-facing reads pass through the same route from the source metric and dense embedding to the universal extension handoff and separated target boundary.

Apply Definition 1606.5 to the shared source read when it is presented through the reflector-unit surface. The unit and dense-embedding rows are prior to the same minimality route and do not add any separate equality principle for completed reads.

It remains only to compare the two exported target reads. The carrier permits that comparison through the separated uniqueness row S and then transports, replays, packages, and names the result through H, R, P, N. The excluded quotient completions, selected representatives, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot enter because no carrier coordinate or cited factorization theorem exports them. □

Theorem 1606.7 (CauchyCompletionMinimality separated reflection route)**.** *Let* $M = (X, C, E, U, F, S, H, R, P, N$ *be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Any completion-facing consumer comparing*

two completion reads must pass through the separated metric reflection row S after the dense embedding row E, the universal-property row U, and the uniformly continuous extension row F have been displayed. The consumer may then read the transport, replay, provenance, and local naming rows H, R, P, N as the terminal route. The route does not export quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier of Definition 1606.1. The displayed rows already place the dense embedding E, universal-property row U, extension row F, and separatedness row S on the same finite packet.

Apply Definition 1606.2. Its dense-image coverage and universal factorization clauses force any completion-facing consumer to pass through E, U, and F before the separated comparison can be read.

Now use Definition 1606.4 and Definition 1606.6. Together they keep the comparison route inside the displayed carrier and force the two completion reads to meet only at the separated row S.

Finally read H, R, P, N as transport, replay, provenance, and naming rows for the same finite packet. Since the carrier has no coordinate for quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no consumer can bypass the separated reflection row. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalitySeparatedReflectionRoute`

Theorem 1606.8 (CauchyCompletionMinimality replay non-escape)**.** *Every completion-minimality consumer of an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier must read the source metric row, the displayed completion row, the dense embedding row, the universal factorization row, the separated uniqueness row, the reflector-unit factorization row, the componentwise* hsame *row, the displayed* Cont *row, the* Pkg *row, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *row before it can read the minimal completion service.*

This route excludes quotient completions, selected representatives, host equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier from Definition 1606.1. The accepted packet already displays the source metric row X, the completion row C, the dense embedding row E, the universal-property row U, the extension row F, the separated row S, and the structural rows H, R, P, N.

Apply the NameCert obligations of Definition 1606.2. They force the consumer to pass through carrier admission, dense-image coverage, universal factorization, separated uniqueness, and ledger non-escape before any completion-facing read is exported.

The reflector-unit factorization of Definition 1606.5, together with Definition 1606.4 and Definition 1606.6, keeps the same route visible through $X, C, E, U, F, S, H, R, P, N$. Hence the consumer cannot bypass the displayed packet by quotient completion, selected representatives, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityReplayNonescape`

Theorem 1606.9 (CauchyCompletionMinimality finite route induction)**.** *Let* $M = (X, C, E, U, F, S, H, R, P, N)$ *be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Finite induction over the displayed service route*

$$X \longmapsto C \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto H \longmapsto R \longmapsto P \longmapsto N$$

preserves the following consumer invariant at every prefix: once a consumer has reached the current row, all earlier dense-image coverage, universal factorization, separated uniqueness, reflector-unit factoring, and replay non-escape obligations already forced by the prefix remain available to the consumer, and no later row can replace them by quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The base prefix is the source metric row X. By the carrier definition in Definition 1606.1, the source row is admitted only as part of the finite packet whose remaining rows are $C, E, U, F, S, H, R, P, N$; there is no separate coordinate for quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient real completeness.

The completion and dense-embedding steps are forced by the carrier and by Definition 1606.2. At the prefix ending in E, Definition 1606.3 supplies the dense-image clause, so later consumers inherit coverage through the displayed completion row rather than through an external host completion.

The universal-property and extension steps are the next induction moves. Applying Definition 1606.4 at the prefixes ending in U and F preserves the factorization route from the source and dense embedding through the universal extension handoff.

At the separated row S, the same induction invariant gains the comparison boundary supplied by the separated uniqueness clause of Definition 1606.2. When the source read is presented through the reflector-unit surface, Definition 1606.5 inserts only the unit and dense-image rows before the same minimality route, so the induction step still has the same consumer invariant.

The terminal structural rows H, R, P, N transport, replay, package, and name the already established prefix data. The replay non-escape theorem Definition 1606.8 closes the induction by showing that these rows expose no bypass around dense-image coverage, universal factorization, separated uniqueness, or reflector-unit factoring. Thus every prefix of the displayed finite route preserves the same minimal completion service boundary and refuses the excluded quotient, representative-selection, host-equality, choice, compactness, and ambient real-completeness principles. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier finite row induction`

Theorem 1606.10 (CauchyCompletionMinimality carrier admission)**.** *Let* $M = (X, C, E, U, F, S, H, R, P, N)$ *be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Carrier admission for* M *is exactly the finite admission of the source metric row* X*, displayed completion row* C*, dense embedding row* E*, universal-property row* U*, uniformly continuous extension row* F*, separated uniqueness row* S*, componentwise transport row* H*, replay row* R*, provenance row* P*, and local naming row* N*.*

No admitted coordinate of M *supplies quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1606.1. Its ten rows are precisely $X, C, E, U, F, S, H, R, P, N$, and the defining list assigns one completion-minimality role to each row.

Apply the NameCert obligations of Definition 1606.2. The first obligation is carrier admission for exactly those rows, while the ledger non-escape obligation forbids any additional completion principle from entering the packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier admission`

Theorem 1606.11 (CauchyCompletionMinimality transport stability)**.** *Let* $M = (X, C, E, U, F, S, H, R, P, N)$ *and* $M' = (X', C', E', U', F', S', H', R', P', N')$ *be accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carriers*

with componentwise hsame *witnesses for the source, completion, embedding, universal-property, extension, separatedness, replay, provenance, and naming rows. If the displayed* Cont *route reads* $X, C, E, U, F, S, H, R, P, N$*, then the transported route reads* $X', C', E', U', F', S', H', R', P', N'$ *with the same dense-image, universal-factorization, separated-uniqueness, replay non-escape, and carrier-admission obligations.*

The transport preserves the absence of quotient completion, selected representatives, host equality, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the componentwise transport row H of the first carrier and the corresponding row H' of the second carrier. These rows witness hsame for the components that can be read by the minimality packet, so each visible obligation can be moved component by component.

The displayed Cont row R orders the source, completion, embedding, universal-property, extension, separatedness, transport, provenance, and naming reads. Transporting along the matching hsame witnesses gives the route through R' with the same row order.

Finally apply Definition 1606.9. Since every prefix obligation is preserved before the terminal structural rows are read, the transported carrier keeps dense-image coverage, universal factorization, separated uniqueness, replay non-escape, and ledger non-escape without adding any excluded coordinate. □

Theorem 1606.12 (CauchyCompletionMinimality ledger exactness)**.** *Let* $M = (X, C, E, U, F, S, H, R, P, N)$ *be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. The ledger rows* R, P, N *are exact for the finite service boundary*

$$X \longmapsto C \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto H \longmapsto R \longmapsto P \longmapsto N.$$

They record the displayed replay, BEDC provenance, and local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *name, and they record no completion principle outside the rows already admitted by the carrier.*

Proof. Read the replay row R as the ordered Cont route through the admitted rows. The route is the same finite row sequence used by Definition 1606.9.

Read P as provenance for the displayed completion, embedding, universal-property, extension, separatedness, history, transport, continuation, package, and NameCert rows, and read N as the local name for this packet. These two rows do not add an independent service coordinate.

The replay non-escape theorem Definition 1606.8 and the carrier-admission theorem Definition 1606.10 together show that R, P, N only preserve the admitted service boundary. Hence the ledger is exact for the listed rows and excludes quotient completion, selected representatives, host equality, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityLedgerExactness`

Theorem 1606.13 (CauchyCompletionMinimality obligation export)**.** *An accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier exports exactly the carrier admission, dense-image coverage, universal factorization, reflector-unit factorization, dense-extension uniqueness, separated-reflection route, replay non-escape, finite-row induction, transport stability, and ledger exactness rows displayed in this chapter. The exported package is read through the same finite service route*

$$X \longmapsto C \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto H \longmapsto R \longmapsto P \longmapsto N.$$

It has no coordinate for quotient completion, selected limits, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1606.1. Its rows are exactly the source metric, displayed completion, dense embedding, universal-property, uniformly continuous handoff, separated uniqueness, componentwise transport, replay, provenance, and local naming rows. Carrier admission is therefore supplied by Definition 1606.10.

The NameCert obligations of Definition 1606.2 give dense-image coverage, universal factorization, separated uniqueness, and ledger non-escape. The dense-image, universal-factorization, reflector-unit, dense-extension uniqueness, separated-reflection, and replay non-escape theorems refine those clauses into the named rows.

Finally apply finite route induction, Definition 1606.9, transport stability, Definition 1606.11, and ledger exactness, Definition 1606.12. These rows preserve each previously displayed obligation along the route from X to N and prevent later rows from replacing them by quotient completion, selected limits, host equality, choice, compactness, or ambient real-completeness data. Thus the accepted carrier exports exactly the stated obligation package. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityObligationExport`

Theorem 1606.14 (CauchyCompletionMinimality public exported package)**.** *The carrier definition Definition 1606.1, the NameCert obligations Definition 1606.2, the carrier admission theorem Definition 1606.10, the transport stability theorem Definition 1606.11, the ledger exactness theorem Definition 1606.12, the finite-row induction theorem Definition 1606.9, and the obligation-export theorem Definition 1606.13 form the public BEDC package consumed by completion-minimality clients.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityPublicExportedPackage`

Proof. The carrier definition supplies the accepted source metric, displayed completion rows, dense embedding, universal-property rows, transport rows, replay rows, provenance rows, and local naming row. The NameCert obligations and carrier-admission theorem show that this row package is admissible.

Transport stability and ledger exactness keep the same public rows visible after finite route traversal, and finite-row induction ensures that no later coordinate replaces them. The obligation-export theorem packages those same rows as the chapter-level export, so the displayed data are exactly the public BEDC package used by completion-minimality clients. □

Theorem 1606.15 (CauchyCompletionMinimality bridge boundary)**.** *Let $M = (X, C, E, U, F, S, H, R, P, N)$ be an accepted* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *carrier. Any standard completion bridge for the public package of Definition 1606.14 must factor through the finite route*

$$X \longmapsto C \longmapsto E \longmapsto U \longmapsto F \longmapsto S \longmapsto H \longmapsto R \longmapsto P \longmapsto N.$$

Along this route the bridge preserves dense-image coverage, universal factorization, separated uniqueness, uniformly continuous transport, replay, provenance, and local naming. It does not replace the displayed package by quotient completions, selected representatives, countable choice, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 1606.1. Its displayed rows are exactly $X, C, E, U, F, S, H, R, P, N$, and the NameCert obligations of Definition 1606.2 expose dense-image coverage, universal factorization, separated uniqueness, and ledger non-escape on those rows.

Apply finite-row induction, Definition 1606.9. The bridge can traverse the public package only by reading the admitted route from the source metric through the local name. The dense-image

theorem, universal-factorization theorem, separated-reflection route, transport-stability theorem, and public exported package theorem preserve the corresponding rows at each stage.

Finally use ledger exactness, Definition 1606.12. The replay, provenance, and local naming rows record the same finite service boundary and no larger completion principle. Hence a standard bridge has a paper-visible boundary precisely at the displayed route and cannot appeal to quotient completions, selected representatives, countable choice, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityBridgeBoundary`

Theorem 1606.16 (CauchyCompletionMinimality bridge exactness)**.** *Let $M = (X, C, E, U, F, S, H, R, P, N)$ and $M' = (X', C', E', U', F', S', H', R', P', N')$ be BEDC completion candidates whose displayed dense embedding rows, reflector-unit factorization rows, dense-extension uniqueness rows, separated-reflection rows, and ledger-exactness rows are read through matching* hsame *transport. If both candidates expose the same public completion-minimality route from source metrics through dense embeddings, universal factorization, uniformly continuous extension, separated uniqueness, replay, provenance, and local naming, then they admit the same* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *bridge packet. The bridge exports only the displayed universal factorization and dense-extension uniqueness route; it exports no quotient representative, choice-selected limit, host real equality, countable-choice schedule, open-cover compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the reflector-unit theorem, Definition 1606.5. It places the source read, unit embedding, dense-image row, reflector row, and triangle replay before any minimality-facing consumer reaches the universal factorization surface.

Next apply dense-extension uniqueness, Definition 1606.6, together with the separated-reflection route, Definition 1606.7. These two rows force the two completion candidates to compare their exported reads only through the common dense embedding, universal-property row, uniformly continuous extension handoff, and separated boundary.

Use ledger exactness, Definition 1606.12. The replay, provenance, and local naming rows record exactly the same finite service boundary on both transported candidates, so no later row can introduce an additional representative selector, host equality, choice schedule, compactness witness, or ambient real-completeness coordinate.

Finally apply the bridge boundary theorem, Definition 1606.15. Since the public bridge can traverse only the displayed finite route and the preceding rows already identify the reflector unit, dense-extension uniqueness, separated comparison, and ledger boundary, the two candidates determine the same bridge packet and export only the stated universal factorization and uniqueness data. □

Theorem 1606.17 (CauchyCompletionMinimality formal-target contract)**.** *Any later Lean-facing formal target for the public* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *package must expose exactly the source metric row X, displayed completion row C, dense embedding row E, universal-property row U, uniformly continuous handoff row F, separated uniqueness row S, componentwise* hsame *transport row H, displayed* Cont *replay row R,* Pkg *provenance row P, and local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ *row N before it claims dense-image coverage, universal factorization, separated reflection, or the bridge boundary of this chapter. A target that omits one of these rows, merges two rows into a host completion object, or adds quotient completion, selected representatives, host equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is not the formal target for the public minimality package.*

Proof. Start with the NameCert obligations of Definition 1606.2. They list carrier admission, dense-image coverage, universal factorization, separated uniqueness, and ledger non-escape for the tuple $M = (X, C, E, U, F, S, H, R, P, N)$.

The public exported package of Definition 1606.14 is read through the same displayed rows. Finite-row induction in Definition 1606.9 prevents a later consumer from skipping from X to a completed value without traversing $C, E, U, F, S, H, R, P, N$.

Use ledger exactness, Definition 1606.12. The replay, provenance, and local naming rows record the same service boundary and do not add a quotient completion, selected representative, host equality, choice, compactness, or ambient real-completeness coordinate.

Finally apply the bridge boundary theorem Definition 1606.15. Since its bridge route preserves dense-image coverage, universal factorization, separated uniqueness, uniformly continuous transport, replay, provenance, and local naming along the displayed finite route, any Lean-facing target claiming those public conclusions must expose precisely the listed rows first. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier formal target contract`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}$ **.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1606.14 exports the public package over source metrics, displayed completion rows, dense embeddings, universal extension, uniformly continuous handoff, separated uniqueness, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMinimality}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityCarrier\_formal\_target\_contrac`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The remaining formal upgrade is bridge checking for the displayed public package.

Constructive story: A $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ packet is generated from a source metric with located source evidence, a displayed completion row, a dense completion embedding, a universal-property row, a uniformly continuous extension handoff, separated uniqueness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1607

A Concrete Naming Certificate for $\mathsf{DirectedSet}^{\uparrow}$

Following Moore–Smith and Kelley directed sets, and the Bishop–Bridges use of finite constructive evidence in analysis, the $\mathsf{DirectedSet}^{\uparrow}$ packet names the directed-index substrate consumed by net-based BEDC chapters. It sits upstream of the directed-window surface of chapter 589, the limit boundary of chapter 1561, and the convergence packet of chapter 1521. The packet records finite preorder and upper-window data; it is not a quotient of indices, not a global cofinal-choice operator, and not a host-level order-completeness principle.

Definition 1607.1 (DirectedSet carrier)**.** *A* $\mathsf{DirectedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, \preceq, W, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the inspected directed addresses.*
2. *$\preceq$ is the comparison row, recording the displayed preorder comparisons among the inspected addresses.*
3. *W is the finite window row, naming each finite subfamily of inspected addresses requested by a downstream net consumer.*
4. *U is the upper-window witness row, assigning to every displayed window in W a named address in I that is above every address in that window according to $\preceq$.*
5. *H records componentwise* hsame *transport for the index, comparison, window, upper-witness, continuation, provenance, and naming rows.*
6. *C records displayed* Cont *replay from a downstream finite-window request through W, $\preceq$, and U.*
7. *P is the* Pkg *provenance over* $\mathsf{DirectedSet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert, *together with the downstream consumers* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, *and* $\mathsf{MooreSmithConvergence}^{\uparrow}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectedSet}^{\uparrow}}$ *row.*

The carrier exports only this finite directed-index packet. It contains no total order extension, no selected cofinal sequence, no quotient on indices, and no infinite choice principle.

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetCarrier fields faithful`

Theorem 1607.2 (DirectedSet NameCert obligations)**.** *For every accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier* $D = (I, \preceq, W, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirectedSet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed index, comparison, window, upper-witness, transport, continuation, provenance, and naming rows;*
2. *preorder readability, stating that every comparison exported to a consumer is one of the displayed* $\preceq$ *rows on addresses in* I*;*
3. *finite-window coverage, stating that every requested finite index family is first named in* W*;*
4. *upper-window witnessing, stating that every displayed window has a named witness in* U *above each of its addresses by the displayed comparison row;*
5. *transport and continuation stability, stating that* H *and* C *preserve the same finite window and witness data under* hsame *replay;*
6. *ledger non-escape, excluding cofinal choice functions, quotient index objects, total-order completions, and infinite upper-bound selection.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1607.1. The finite index row, comparison row, displayed windows, upper-window witnesses, transports, continuations, provenance, and local naming row are visible coordinates of one BHist packet.

Preorder readability and finite-window coverage are forced by the row order. A consumer can read an address only through I, a comparison only through $\preceq$, and a finite request only after that request has been named in W.

The upper-window obligation is the defining role of U. For each displayed window, U supplies a named address and the comparison row supplies the finite list of relations showing that the address lies above every member of the window.

Transport and continuation stability use the structural rows H and C. They replay the same displayed windows, comparisons, and upper-witness entries, so they do not add a cofinal selector or an unrecorded index relation.

Finally, the package and naming rows name exactly the displayed finite packet. Since no coordinate stores a quotient of indices, a total-order completion, a selected cofinal sequence, or an infinite choice function, any request for such data remains outside the local ledger. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetTasteGate single carrier alignment`

Theorem 1607.3 (DirectedSet CauchyNet index handoff)**.** *Let* $D = (I, \preceq, W, U, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier. Any* $\mathsf{CauchyNet}^{\uparrow}$ *or* $\mathsf{CauchyNetLimit}^{\uparrow}$ *packet that reads its directed-window ledger from* D *receives index data only through the finite route*

$$I, \preceq, W, U \longmapsto H, C, P, N.$$

Thus the Cauchy-net carrier of Definition 589.1 and the limit carrier of Definition 1561.1 may share the same displayed directed-index substrate without adding quotient nets, selected limits, cofinal choice, or ambient completeness.

Proof. Start with a downstream Cauchy-net request. The directed addresses available to the consumer are the entries of I, and the comparisons available to the consumer are the displayed rows of $\preceq$.

The finite request is then named in W. The witness row U supplies the common upper-window address needed by the $\mathsf{CauchyNet}^{\uparrow}$ directed-window ledger, and the same witness row can be replayed before the $\mathsf{CauchyNetLimit}^{\uparrow}$ seal handoff is read.

The structural rows H and C transport and replay this route componentwise. The provenance and local naming rows record that the exported data is the finite directed-index packet, not a quotient net, selected host limit, cofinal-choice selector, or completeness assertion. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSet cauchynet index handoff certificate`

Theorem 1607.4 (DirectedSet filter-net handoff)**.** *Let $D = (I, \preceq, W, U, H, C, P, N)$ be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier. Any* $\mathsf{FilterConvergence}^{\uparrow}$ *or* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet that reads a directed index from D must expose the finite index row I, displayed preorder comparisons $\preceq$, finite request windows W, and upper-window witnesses U before the filter-base handoff, net tail ledger, or convergence consumer can be read. The resulting route exports finite directed-window data only; it does not supply a host directed set, cofinal choice, quotient index, or total-order completion.*

Proof. Project the carrier rows of Definition 1607.1. The index addresses, preorder comparisons, request windows, and upper-window witnesses are exactly the rows $I, \preceq, W, U$.

Apply the CauchyNet index handoff of Definition 1607.3. It already forces net-style consumers to read the directed-window ledger through the displayed index, comparison, window, and witness rows before transport, replay, provenance, and naming.

Use finite upper-window stability, Definition 1607.5, for retained filter and Moore–Smith windows. Retention keeps the upper witnesses inside the displayed subledger, so the filter-base or net-tail read cannot replace them by a cofinal selector.

Since the carrier contains no row for a host directed set, quotient index, total-order completion, or choice-indexed cofinal family, those objects cannot discharge the filter-net handoff. Every downstream read remains bounded by $I, \preceq, W, U, H, C, P, N$. □

Theorem 1607.5 (DirectedSet finite upper-window stability)**.** *Let $D = (I, \preceq, W, U, H, C, P, N)$ be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier, and let Ω be a retained finite subledger of I whose displayed windows in W_Ω keep their named upper witnesses in Ω. Then*

$$D_\Omega = (I_\Omega, \preceq_\Omega, W_\Omega, U_\Omega, H_\Omega, C_\Omega, P_\Omega, N_\Omega)$$

is an accepted $\mathsf{DirectedSet}^{\uparrow}$ *carrier for the retained windows. Every* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *consumer of D_Ω therefore reads the same finite upper-window witnesses, restricted to Ω.*

Proof. Restrict the index row to I_Ω and the comparison row to $\preceq_\Omega$. The hypothesis on retained windows says that every window kept in W_Ω still has its upper witness inside the retained index row.

The witness row U_Ω is the corresponding restriction of U. For each retained window, the required comparison entries are exactly the entries of $\preceq_\Omega$ that already certified the witness in the original carrier.

The transport, continuation, provenance, and naming rows restrict componentwise to H_Ω, C_Ω, P_Ω, and N_Ω. Their endpoints are among the retained rows, so replay cannot leave the finite subledger.

Apply the obligation surface of Definition 1607.2. Carrier admission is the displayed restricted tuple; preorder readability, finite-window coverage, upper-window witnessing, and structural stability are inherited by restriction. Ledger non-escape remains valid because no retained coordinate stores quotient indices, cofinal choice, total-order completion, or infinite upper-bound selection. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetPacket finite upper window stability`

Theorem 1607.6 (DirectedSet carrier obligation)**.** *For every accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier* $D = (I, \preceq, W, U, H, C, P, N)$*, the carrier obligation is the visible finite row family consisting of the index row* I*, the displayed preorder comparison row* $\preceq$*, the finite-window row* W*, and the upper-window witness row* U*, together with structural rows* H, C, P, N*. No carrier obligation may be discharged by quotient indices, a total-order completion, cofinal choice, or infinite upper-bound selection.*

Proof. Project the carrier of Definition 1607.1. The rows $I, \preceq, W, U, H, C, P, N$ are exactly the displayed coordinates of the accepted packet.

Apply the NameCert obligation theorem Definition 1607.2. Its carrier admission, preorder readability, finite-window coverage, and upper-window witnessing clauses identify the same visible rows.

The non-escape clause excludes the forbidden objects. Since none of them appears as a carrier coordinate, no downstream consumer can use one to discharge the carrier obligation. □

Theorem 1607.7 (DirectedSet preorder stability obligation)**.** *Let* $D = (I, \preceq, W, U, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier. Every displayed preorder comparison exported from* D *remains stable under the componentwise* hsame *transport row* H *and the displayed* Cont *replay row* C*. The stable comparison is still a row of* $\preceq$ *on addresses from* I*, not a host preorder, quotient relation, total-order completion, or hidden comparison outside the packet.*

Proof. Read the comparison row through Definition 1607.1. The only preorder comparisons available to a consumer are the displayed entries of $\preceq$ between addresses named by I.

Use the transport and continuation stability clause of Definition 1607.2. It requires H and C to preserve the same finite windows, comparisons, and upper-witness data under replay.

The CauchyNet handoff theorem Definition 1607.3 then sends net consumers through the same row family. Thus a transported or replayed comparison cannot be replaced by a quotient relation, a host preorder, or a total-order completion. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetCarrier preorder stability obligation`

Theorem 1607.8 (DirectedSet cofinal merge obligation)**.** *Let* $D = (I, \preceq, W, U, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier, and let two retained finite windows in* W *have displayed upper witnesses named by* U*. The cofinal merge obligation for their union is discharged only by naming that union as a finite window in* W *and reading its displayed upper witness through* U*. The resulting merge remains a finite-window witness and is not a cofinal sequence, infinite upper-bound selector, quotient index, or choice function.*

Proof. Begin with the two retained windows. By Definition 1607.2, each window is first named in W and each upper witness is read through U using displayed comparisons from $\preceq$.

For the merged request, use the same finite-window coverage clause on the union window. Once that union is displayed in W, the upper-window witnessing clause supplies its named upper witness in U.

Apply finite upper-window stability, Definition 1607.5, to retained subledgers. It keeps the merged witness inside the same finite route and prevents the merge from becoming a cofinal sequence, quotient index, infinite selector, or choice function. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetCarrier cofinal merge obligation`

Theorem 1607.9 (DirectedSet upper-window exhaustion)**.** *Let* $D = (I, \preceq, W, U, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier. Any* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, *or* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *consumer that reads a finite directed-window request from* D *factors through the displayed rows*

$$I, \preceq, W, U, H, C, P, N.$$

The consumer therefore cannot obtain a cofinal sequence, quotient index, total-order completion, infinite upper-bound selector, or choice function from the directed-window read.

Proof. Start with the carrier of Definition 1607.1. The only index addresses are the entries of I, the only comparisons are the displayed rows of $\preceq$, and the only finite requests are the windows named in W.

Apply Definition 1607.2. Its finite-window coverage and upper-window witnessing clauses force every accepted request to read its witness through U, while its transport and continuation clauses route every replay through H and C.

Now use Definition 1607.3 for $\mathsf{CauchyNet}^{\uparrow}$ and $\mathsf{CauchyNetLimit}^{\uparrow}$ consumers, and Definition 1607.5 for retained Moore–Smith windows. These results identify the same finite upper-window route after restriction or handoff.

Finally, P and N source and name only that route. Since the carrier and the cited obligations contain no row for a cofinal sequence, quotient index, total-order completion, infinite upper-bound selector, or choice function, no downstream consumer can extract one from the finite directed-window request. □

Lean4 ✓ `BEDC.Derived.DirectedSetUp.DirectedSetPacket upper window exhaustion`

Theorem 1607.10 (DirectedSet obligation closure package)**.** *The accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier*

$$D = (I, \preceq, W, U, H, C, P, N)$$

supports an obligationClosure package whose finite boundary consists of carrier admission for the displayed rows, preorder readability through $\preceq$*, finite-window coverage through* W*, upper-window witnessing through* U*, transport and* Cont *replay through* H, C*, retained-subledger stability, and non-escape from cofinal selectors, quotient indices, total-order completions, infinite upper-bound selection, and choice functions. Every* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, *and* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *consumer of the package reads only this finite directed-window substrate.*

Proof. Begin with the carrier definition, Definition 1607.1. It supplies the finite index row, comparison row, finite-window row, upper-witness row, transport row, continuation row, provenance row, and local naming row.

Apply the NameCert obligations of Definition 1607.2. Carrier admission, preorder readability, finite-window coverage, upper-window witnessing, transport replay, continuation replay, and ledger non-escape are exactly the displayed obligations.

Use the CauchyNet index handoff, Definition 1607.3, to identify how $\mathsf{CauchyNet}^{\uparrow}$ and $\mathsf{CauchyNetLimit}^{\uparrow}$ consumers read the same route through $I, \preceq, W, U, H, C, P, N$.

Use finite upper-window stability and upper-window exhaustion, Definition 1607.5 and Definition 1607.9. These keep retained subledgers inside the same finite route and show that Moore–Smith and Cauchy-net consumers cannot extract cofinal selectors, quotient indices, total-order completions, infinite upper-bound selection, or choice functions. □

Theorem 1607.11 (DirectedSet mature consumer package)**.** *Let* $D = (I, \preceq, W, U, H, C, P, N)$ *be an accepted* $\mathsf{DirectedSet}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, *and* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *consumer of* D *reads the directed-index substrate only through*

$$I, \preceq, W, U, H, C, P, N.$$

The package contains no cofinal sequence selector, quotient index object, total-order completion, infinite upper-bound selector, or choice function, and no mature consumer may replace the finite directed-window route by any such external object.

Proof. Begin with the carrier definition, Definition 1607.1. Its only public rows are the finite index row I, preorder row $\preceq$, finite-window row W, upper-witness row U, transport row H, continuation row C, provenance row P, and local naming row N.

Apply the NameCert obligations and the obligation closure package, Definition 1607.2 and Definition 1607.10. These identify the accepted directed-window boundary with exactly those rows and make the consumer-facing obligations exhaustive.

Next use the CauchyNet handoff and upper-window results, Definition 1607.3, Definition 1607.5, and Definition 1607.9. They show that Cauchy-net, Cauchy-net-limit, and Moore–Smith reads preserve the same finite route after restriction, handoff, and retained-window replay.

Since the cited carrier and obligations contain no row for a cofinal sequence selector, quotient index, total-order completion, infinite upper-bound selector, or choice function, a mature consumer has no source from which to extract one. Thus all named consumers read only $I, \preceq, W, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedSet}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DirectedSet}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1607.1, the NameCert obligations, Definition 1607.3, finite-window stability, Definition 1607.9, Definition 1607.10, and Definition 1607.11 bind the mature BEDC primitive boundary for the finite directed-index substrate consumed by Cauchy-net, Cauchy-net-limit, and Moore–Smith convergence packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DirectedSet}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.DirectedSetUp.DirectedSetCarrier\_fields\_faithful`

Bridge status: none

Not claimed: This chapter does not close quotient index objects, selected cofinal sequences, total-order completions, infinite choice principles, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a standard-schema bridge for the same directed-index packet.

Constructive story: A $\mathsf{DirectedSet}^{\uparrow}$ packet is generated from a finite index row, displayed preorder comparisons, finite requested windows, common upper-window witnesses, componentwise hsame transports, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 1608

A Concrete Naming Certificate for $\mathsf{RealAbs}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealAbs}^{\uparrow}$ packet records absolute value for displayed regular real names by finite readback rows rather than by a host real operation. It reads the real-name seal of chapter 38, the dyadic absolute-value windows of chapter 293, and the regular rational handoff of chapter 301. The packet also uses the apartness and order surface supplied by $\mathsf{RealAlgOrder}^{\uparrow}$, while withholding host trichotomy, quotient reals, selected limits, and completeness principles.

Definition 1608.1 (Real absolute-value carrier)**.** *A* $\mathsf{RealAbs}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, D, S, G, V, H, C, P, N)$$

with the following displayed coordinates:

1. *R is the displayed* $\mathsf{Real}^{\uparrow}$ *seal whose regular real name is being inspected.*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *apartness and order row used only as a displayed comparison surface for the real-name read.*
3. *D is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *window row attached to the dyadic observations opened from R.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation window through which finite requests to the real name are scheduled.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that supplies regular rational readback before the real seal is exported.*
6. *V is the finite nonnegative absolute-value ledger recording the endpoint and tolerance data available to downstream consumers.*
7. *H stores componentwise* hsame *transport for the real seal, order row, dyadic window, observation window, handoff, ledger, provenance, and local name.*
8. *C stores the displayed* Cont *routes from R through S and G into D, and from D into the nonnegative ledger V.*
9. *P records* Pkg *provenance over* $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealAbs}^{\uparrow}}$ *row.*

The carrier contains no coordinate for host trichotomy, quotient real equality, a selected limit, an external absolute-value operation, or a completeness witness.

Theorem 1608.2 (Real absolute-value NameCert obligations)**.** *For every accepted* $\mathsf{RealAbs}^{\uparrow}$ *carrier* $K = (R, A, D, S, G, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealAbs}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed real seal, apartness and order row, dyadic absolute-value window, observation window, regular rational handoff, nonnegative ledger, transport, continuation, provenance, and local naming rows;*
2. *classifier and transport stability, requiring accepted reads to move componentwise through the stored* hsame *rows and to replay through the displayed* Cont *routes;*
3. *dyadic-window handoff, requiring every absolute-value observation to pass from the* $\mathsf{Real}^{\uparrow}$ *seal through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows into the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *window row;*
4. *nonnegative real-seal export, requiring the output ledger* V *to expose only a nonnegative* $\mathsf{Real}^{\uparrow}$*-facing seal read from the displayed dyadic windows and regular handoff;*
5. *non-escape from trichotomy and completeness, excluding host order trichotomy, quotient real equality, selected limits, external absolute-value operations, and ambient completeness.*

No NameCert obligation is stored outside these five finite rows.

Proof. Carrier admission is projection from Definition 1608.1. The only available absolute-value data are the real seal R, the order row A, the dyadic window D, the observation window S, the regular handoff G, the nonnegative ledger V, and the structural rows H, C, P, N.

Classifier and transport stability are read from H and replayed through C. A transported carrier therefore keeps the same displayed route from the real seal through finite observation and regular rational readback into the dyadic absolute-value window.

The dyadic-window handoff uses chapter 293 as the only absolute-value window source. The output ledger V exports the nonnegative real-facing seal after that handoff, so the packet never opens a separate host absolute-value, trichotomy, quotient, selected-limit, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealAbs}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealAbs}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1608.1 and Definition 1608.2 seed the finite real absolute-value surface over displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealAbs}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host trichotomy, quotient real equality, selected limits, external absolute-value operations, or ambient completeness.

Upgrade path: Obligation closure requires formal carrier admission, classifier and transport stability, dyadic-window handoff, nonnegative real-seal export, and non-escape from trichotomy and completeness for the same finite packet.

Constructive story: A $\mathsf{RealAbs}^{\uparrow}$ packet is generated from a displayed Real seal, RealAlgOrder apartness and order row, DyadicAbsoluteValue window row, StreamName observation window, RegSeqRat handoff, nonnegative absolute-value ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1609

A Concrete Naming Certificate for $\mathsf{EventuallyZeroSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{EventuallyZeroSequence}^{\uparrow}$ packet records the finite-cutoff source of a null rational difference sequence: after one displayed cutoff row, every later observation is the zero dyadic row. It refines the zero-convergence consumer surface of chapter 380 by replacing an arbitrary modulus lookup with a visible cutoff induction over the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic zero row of chapter 291, the constant rational handoff of chapter 364, and the terminal equality seal of chapter 38.

Definition 1609.1 (Eventually-zero sequence carrier)**.** *An* $\mathsf{EventuallyZeroSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, G, D, Z, T, H, C, P, N)$$

with the following visible coordinates:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose accepted requests enumerate the displayed sequence positions.*
2. G *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the distinguished zero dyadic row and the finite comparison rows used by each inspected window.*
4. Z *is the* $\mathsf{StationaryRationalCauchy}^{\uparrow}$ *zero source whose constant rational readback supplies the zero row consumed after the cutoff.*
5. T *is the cutoff row: every accepted request at a position beyond* T *must replay the zero row of* D *through the stationary source* Z*.*
6. H *records componentwise* hsame *transport for stream windows, regular readbacks, dyadic rows, the stationary zero source, the cutoff row, continuation routes, provenance, and the local naming row.*
7. C *records displayed* Cont *replay from a sequence request through* S*, then* G*, then the dyadic row* D*, with the branch beyond* T *routed through* Z*.*
8. P *is the* Pkg *provenance over* $\mathsf{EventuallyZeroSequence}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StationaryRationalCauchy}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{EventuallyZeroSequence}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a quotient stream, selected host limit, trichotomy principle, or ambient completeness witness for $\mathsf{Real}^{\uparrow}$.

Theorem 1609.2 (Eventually-zero sequence NameCert obligations)**.** *For every accepted* $\mathsf{EventuallyZeroSequence}^{\uparrow}$ *carrier* $E = (S, G, D, Z, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EventuallyZeroSequence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the stream window, regular rational readback, dyadic zero ledger, stationary zero source, cutoff, transport, continuation, provenance, and local naming rows;*
2. *classifier and transport stability for the displayed rows under the componentwise* hsame *ledger* H*;*
3. *cutoff-window exactness, requiring every request beyond* T *to read the zero dyadic row of* D *through the stationary source* Z*;*
4. *null-sequence handoff, requiring the same finite cutoff packet to instantiate the* $\mathsf{NullSequence}^{\uparrow}$ *tolerance-window route;*
5. *non-escape, excluding quotient streams, selected limits, trichotomy, external real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1609.1. The displayed tuple supplies exactly the stream, readback, dyadic, stationary zero, cutoff, transport, continuation, package, and local naming rows.

Classifier and transport stability are componentwise. The rows in H move the finite window S, the readback G, the dyadic ledger D, the stationary source Z, and the cutoff T, while C replays the same request order after transport.

The cutoff-window obligation is the only analytic-looking row, but it is still finite: beyond T, each accepted window is routed through the stationary zero source and returns the zero dyadic row already listed in D.

The null-sequence handoff and non-escape rows then read the same finite packet. Since quotient streams, selected limits, trichotomy, external equality, and completeness are absent from the carrier, no local obligation can export them. □

Theorem 1609.3 (Eventually-zero cutoff-window induction)**.** *Let* $E = (S, G, D, Z, T, H, C, P, N)$ *be an accepted* $\mathsf{EventuallyZeroSequence}^{\uparrow}$ *carrier. For every finite* $\mathsf{StreamName}^{\uparrow}$ *window whose positions are all beyond the cutoff row* T*, induction over the listed window positions shows that each* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* G *is the zero dyadic row of* D *routed through the stationary source* Z*.*

Consequently every tail window admitted by S *beyond* T *is a finite zero window, transported only by* H *and replayed only through* C*.*

Proof. Use induction on the finite list of positions recorded by the accepted $\mathsf{StreamName}^{\uparrow}$ window.

The empty window has no readback request, so the zero-window claim is immediate from carrier admission.

For a window extended by one position beyond T, first apply the cutoff row. The replay route C sends the final request through S, then G, then D, and the beyond-cutoff branch forces the stationary zero source Z.

Apply the induction hypothesis to the earlier positions in the window. Transport through H preserves the same stream, readback, dyadic zero, stationary source, and cutoff rows, so adjoining

the final zero readback yields a finite window all of whose displayed values are the zero dyadic row. □

Theorem 1609.4 (Eventually-zero NullSequence handoff)**.** *Every accepted* $\mathsf{EventuallyZeroSequence}^{\uparrow}$ *carrier* $E = (S, G, D, Z, T, H, C, P, N)$ *determines a* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail packet for the same displayed regular rational stream: a tolerance request is answered by the cutoff row* T*, the post-cutoff window is read through* S *and* G*, the zero dyadic row is supplied by* D *and* Z*, and the resulting evidence is exported to the* $\mathsf{Real}^{\uparrow}$ *equality seal through the recorded* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Begin with a $\mathsf{NullSequence}^{\uparrow}$ tolerance request. The eventually-zero packet uses T as the displayed threshold row for that request.

Once the request is beyond the cutoff, apply Definition 1609.3. The selected finite window reads only the zero dyadic row through S, G, D, and Z.

Repack those rows as the null-sequence tolerance-window package. The stream row is G, the tolerance and zero ledger are read from D, the window is supplied by S, and the bound row is the cutoff T.

The remaining coordinates H, C, P, N provide the transport, replay, dependency, and local naming rows required by $\mathsf{NullSequence}^{\uparrow}$. The $\mathsf{Real}^{\uparrow}$ seal therefore consumes a finite zero-tail packet rather than a host limit or quotient stream. □

Theorem 1609.5 (Eventually-zero public certificate boundary)**.** *The public surface of an accepted* $\mathsf{EventuallyZeroSequence}^{\uparrow}$ *packet is exhausted by the finite rows*

$$S, G, D, Z, T, H, C, P, N.$$

It exposes the cutoff induction of Definition 1609.3 and the $\mathsf{NullSequence}^{\uparrow}$ *handoff of Definition 1609.4, and it exposes no quotient stream representative, selected limit, trichotomy principle, hidden real equality test, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 1609.1. Every public read is a read of the displayed stream, regular rational row, dyadic zero ledger, stationary zero source, cutoff, transport, continuation, provenance, or local naming row.

The cutoff-window induction is internal to those rows. It uses only the finite window S, the readback G, the dyadic ledger D, the stationary source Z, the cutoff T, and the displayed transport and continuation rows.

The handoff to $\mathsf{NullSequence}^{\uparrow}$ repacks the same data as a tolerance-window route. It does not add a separate convergence object, because the threshold is already the cutoff row and every tail value is the zero dyadic row.

Since no quotient stream, selected limit, trichotomy principle, hidden equality test, or completeness witness is a coordinate of the packet, none can be exported by the public certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{EventuallyZeroSequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EventuallyZeroSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1609.1, Definition 1609.2, Definition 1609.3, Definition 1609.4, and Definition 1609.5 seed the finite cutoff route from StreamName windows through RegSeqRat readback, DyadicRatCore zero rows, StationaryRationalCauchy zero source, NullSequence handoff, and the Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EventuallyZeroSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient streams, selected limits, trichotomy, hidden real equality tests, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier and transport stability, cutoff-window exactness, NullSequence handoff, and public non-escape for BEDC.Derived.EventuallyZeroSequenceUp.

Constructive story: An EventuallyZeroSequence$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, a DyadicRatCore zero ledger, a StationaryRationalCauchy zero source, a cutoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1610

A Concrete Naming Certificate for $\mathsf{HeineCantorModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{HeineCantorModulus}^{\uparrow}$ packet records the finite-net form of Heine–Cantor as explicit BEDC modulus data. It reads the compact-metric finite net of chapter 129, the finite compact-uniform-continuity route of chapter 388, the pointwise rows of chapter 130, the consumer modulus surface of chapter 369, and the final triangle handoff of chapter 1159. The chapter names the finite certificate route that turns local pointwise modulus rows on a displayed compact net into one tolerance-indexed uniform modulus.

Definition 1610.1 (Heine-Cantor modulus carrier)**.** *A* $\mathsf{HeineCantorModulus}^{\uparrow}$ *carrier is a finite BEDC packet*

$$H = (X, Y, F, \varepsilon, P, C, R, M, T, Q, S, K, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through* $\mathsf{NameCert}_{\mathsf{CompactMetric}^{\uparrow}}$*, with its finite* $\mathsf{ProbeBundle}$ *net family at the target precision scale.*
2. *Y is the target metric row whose triangle ledger receives the final estimate.*
3. *F is the continuous-map row from X to Y, read through* $\mathsf{NameCert}_{\mathsf{ContinuousMap}^{\uparrow}}$*, with pointwise modulus rows at the listed net centers.*
4. *ε is the positive target dyadic precision history whose uniform modulus is being named.*
5. *P is the finite* $\mathsf{ProbeBundle}$ *net selected from the compact metric certificate for the source scale required by ε.*
6. *C lists the center histories in P, together with their membership constructors and coverage rows.*
7. *R lists the pointwise continuity radii attached to those centers and the rational lower-bound rows used to fold them into one positive source precision.*
8. *M is the exported source precision history, classified componentwise by the* $\mathsf{Rat}^{\uparrow}$ *and source-metric classifiers.*
9. *T is the metric triangle handoff row, read through chapter 1159, which carries the compact-net source estimates and the two pointwise target estimates to the final ε bound.*
10. *Q records componentwise* hsame *transport for $X, Y, F, \varepsilon, P, C, R, M, T$.*

11. *S records the displayed* Cont *routes from compact metric data, pointwise continuity data, finite-fold data, and triangle data to the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer.*
12. *K records* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, *and* Cont *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineCantorModulus}^{\uparrow}}$ *row.*

The carrier accepts only these finite rows. Its displayed continuity condition is the continuation assertion

$$\mathsf{Cont}(H, \mathsf{NameCert}_{\mathsf{UniformModulus}^{\uparrow}})$$

through the rows $P, C, R, M, T, Q, S, K, N$; it has no coordinate for open-cover compactness, sequential compactness, choice-selected centers, trichotomy, or host real equality.

Theorem 1610.2 (Heine-Cantor modulus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{HeineCantorModulus}^{\uparrow}}$ *obligation surface consists exactly of the finite carrier exposure of Definition 1610.1, componentwise* hsame *transport for all displayed rows, the finite-net extraction row, the folded positive source-precision row, the metric triangle handoff row, and the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer route. No accepted obligation may cite an open-cover compactness theorem, a sequential compactness principle, a choice operation, a trichotomy split on host reals, or host equality of real points.*

Proof. Project the carrier packet H. The source, target, continuous-map, precision, net, center, radius, folded-modulus, triangle, transport, continuation, provenance, and local naming rows are the whole displayed surface.

Classifier stability is componentwise. The row Q transports each displayed history by the corresponding source, target, rational, finite-bundle, and triangle classifier, while S replays the same Cont route after transport.

The remaining obligations are the three constructive readings: extract the finite net from $\mathsf{CompactMetric}^{\uparrow}$, fold the pointwise radii in the finite list to obtain M, and apply the metric triangle handoff to deliver the target ε estimate. Since every reading uses a named coordinate of H, the excluded compactness and choice-theoretic routes have no readable slot in the local NameCert packet. □

Theorem 1610.3 (Heine-Cantor finite-net extraction)**.** *For an accepted* $\mathsf{HeineCantorModulus}^{\uparrow}$ *carrier H, the net rows P and C determine the finite list of centers used by the modulus construction. Each listed center is a source history carried by X, each membership row is a displayed* ProbeBundle *constructor, and each coverage row is a compact-metric distance ledger at the precision scale read from ε. The extraction is by finite-row projection and induction over the displayed net spine.*

Proof. Read P from the carrier. It is already a finite ProbeBundle row from the compact-metric certificate, so the only available centers are the histories listed in its finite spine.

Induct over that spine. At the empty boundary no center is produced and no coverage row is exported. At a successor row, project the displayed center history, its membership constructor, and the attached compact-metric coverage ledger, then append those rows to the extracted list.

The induction uses no cover refinement or external compactness principle. Every step reads the next stored row of P and C, and componentwise hsame transport in Q carries the extracted center and coverage rows to classified representatives. □

Theorem 1610.4 (Heine-Cantor triangle handoff)**.** *For an accepted* $\mathsf{HeineCantorModulus}^{\uparrow}$ *carrier H, the folded source precision M, the compact-net center rows C, and the pointwise continuity rows R determine the triangle handoff row T. Consequently the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer receives a*

single positive source precision whose validity for the target tolerance ε is witnessed by finite-net coverage, two pointwise target estimates at the selected center, and the target metric triangle ledger.

Proof. Begin with the finite-net extraction theorem. It gives the selected center route and compact-metric source coverage rows from the displayed net packet.

Next read the folded precision M from the rows R. The fold is finite: each pointwise radius attached to a center is compared through the rational ledger, and the accumulator remains a positive history classified by the source and rational classifiers.

Finally apply the triangle handoff row T. The source metric ledger places both source points inside the pointwise radius of the selected center, the continuous-map row supplies the two target estimates at that center, and the target metric triangle ledger returns the displayed ε estimate. The continuation row S sends exactly this finite package to $\mathsf{UniformModulus}^{\uparrow}$. □

Theorem 1610.5 (Heine-Cantor radius fold)**.** *For an accepted* $\mathsf{HeineCantorModulus}^{\uparrow}$ *carrier H, the folded source precision M is determined only by the finite center rows C, the pointwise radius rows R, and the rational lower-bound fold displayed in those rows. The extraction has no coordinate for open-cover compactness, a host minimum operator, a selector for centers, trichotomy on host reals, or host equality of real points.*

Proof. Project C, R, and M from the carrier. The finite-net extraction theorem identifies the center spine, so each source radius available to the fold is already attached to a displayed center row.

Induct over that finite center spine. At the empty boundary the fold reads the displayed positive source-precision base row. At a successor row, project the next center, its pointwise continuity radius, and its rational lower-bound certificate, then combine it with the accumulated positive history by the finite rational fold.

Componentwise hsame transport in Q carries the center row, radius row, and fold accumulator through the source-metric and rational classifiers. Thus the exported M is a finite-row object of C and R alone, with no appeal to open-cover data or host-side order case splits. □

Theorem 1610.6 (Heine-Cantor UniformModulus export)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer accepted by the* $\mathsf{HeineCantorModulus}^{\uparrow}$ *chapter factors through the finite rows $P, C, R, M, T, Q, S, K, N$. The route is finite-net extraction, radius fold, and metric triangle handoff, rather than an open-cover compactness argument, choice-selected center data, or host-side real comparison.*

Proof. First project the carrier rows. The finite-net extraction theorem reads P and C as the complete center and coverage surface for the source precision scale determined by ε.

Next apply the radius-fold theorem. It turns the pointwise rows R into the single positive source precision M by a finite rational lower-bound fold, and Q transports the resulting history componentwise across classified representatives.

Finally apply the triangle handoff theorem. The row T combines compact-net coverage, the two pointwise target estimates, and the target metric triangle ledger; the continuation row S presents exactly this package to the $\mathsf{UniformModulus}^{\uparrow}$ consumer. The provenance row K and local row N record that no further compactness, selector, or host comparison input is part of the accepted route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineCantorModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HeineCantorModulus}^{\uparrow}$ packet is generated from a compact metric finite-net row, pointwise continuous-map modulus rows at the displayed centers, a positive target dyadic precision, a finite rational lower-bound fold, metric triangle handoff data, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HeineCantorModulus}^{\uparrow}}$, seed)

Scope closed: The seed scope records the finite-net carrier, displayed pointwise-radius fold, and triangle handoff route from compact metric and continuous-map data to a UniformModulus$^{\uparrow}$ consumer.

Formal verification: FormalStatus(NameCert$_{\text{HeineCantorModulus}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.MetricTriangleUniformEstimateUp.MetricTriangleUniformEstimateCarrier\_heine\_cantor\_la`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, choice, trichotomy, host real equality, or a standalone Lean theorem for HeineCantorModulus$^{\uparrow}$.

Upgrade path: A stronger closure grade requires a dedicated Lean carrier for HeineCantorModulus$^{\uparrow}$ and checked projections from the compact-net fold to the uniform-modulus consumer.

Chapter 1611

A Concrete Naming Certificate for $\mathsf{RegularCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyCriterion}^{\uparrow}$ packet records the finite regular Cauchy criterion surface used by the Real and Completion route before any quotient stream or host-real completeness principle is available. It reads the finite stream windows of chapter 126, the regular rational readback of chapter 301, the tolerance ledger of chapter 292, the dyadic arithmetic core of chapter 291, the criterion surface of chapter 392, the convergence-facing criterion of chapter 1509, the Real-facing criterion of chapter 1590, and the terminal Real boundary of chapter 38. The packet is a finite RegSeqRat-level criterion; it is not a quotient stream, not a selected limit, not host real equality, not trichotomy, not countable choice, and not ambient Real completeness.

Definition 1611.1 (Regular Cauchy criterion carrier)**.** *A* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, M, D, Q, V, A, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose observations are admitted only as displayed stream segments.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same stream windows and carrying regular rational approximants.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *modulus row, recording the displayed stage bound used by each tolerance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row through which every requested radius is named before it is compared with a regular rational distance.*
5. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *row stating the finite Cauchy test over S, R, M, D.*
6. *V is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row that repacks the same finite test as a convergence-facing criterion without selecting a limit.*
7. *A is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary row, available only after S, R, M, D, Q, V have been replayed.*
8. *H records componentwise* hsame *transport for the stream, readback, modulus, dyadic, criterion, convergence criterion, Real boundary, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a criterion request through stream windows, regular rational readback, modulus lookup, dyadic tolerance naming, CauchyCriterion testing, CauchyConvergenceCriterion repacking, and the Real boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCriterion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterionTasteGate single carrier alignment`

Theorem 1611.2 (Regular Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *carrier* $K = (S, R, M, D, Q, V, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, CauchyModulus lookup, DyadicRatCore tolerances, CauchyCriterion test, CauchyConvergenceCriterion repacking, Real boundary, transport, continuation, provenance, and local naming row;*

2. *stream-readback alignment, requiring every criterion request to read finite windows S before the regular rational row R is used;*

3. *modulus-tolerance exposure, requiring every requested stage bound to pass through M and every radius comparison to pass through D;*

4. *criterion exactness, requiring Q to be the only exported Cauchy test and V to be the only convergence-facing repacking of that test;*

5. *Real seal nonescape, requiring A, H, C, P, N to expose no quotient stream, selected limit, host real equality, trichotomy, countable-choice selector, or ambient completeness principle.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion namecert obligations`
↪ variant `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterionNameCertObligations`

Proof. Project the carrier of Definition 1611.1. The displayed tuple contains exactly the stream, readback, modulus, dyadic, criterion, convergence criterion, Real boundary, transport, continuation, provenance, and local naming rows.

Stream-readback alignment is forced by the replay row C. A criterion request reads finite stream windows S before it can inspect the regular rational approximants stored in R.

The modulus route then consumes M and D. The stage bound is read from M, while the tolerance radius is named by the DyadicRatCore row D before any Cauchy comparison is exported.

The only criterion-facing rows are Q and V. Thus the CauchyCriterion test is exported through Q, and the convergence-facing repacking is exported through V without adding a selected limit.

The Real boundary is downstream of the same route. Since the carrier has no quotient-stream, selected-limit, host-equality, trichotomy, choice, or ambient-completeness coordinate, the listed rows exhaust the local obligation surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterionCarrier namecert obligations`

Theorem 1611.3 (Regular Cauchy criterion modulus route)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. Every criterion-facing consumer of K factors through the finite route*

$$S \longmapsto R \longmapsto M \longmapsto D \longmapsto Q \longmapsto V \longmapsto A \longmapsto H, C, P, N.$$

Consequently the consumer receives the modulus-bound and dyadic-tolerance evidence before it can read the CauchyCriterion row or the CauchyConvergenceCriterion row.

Proof. Begin with a criterion-facing consumer. Carrier admission exposes the criterion rows, but only as rows downstream of the finite stream, readback, modulus, and dyadic records.

Replay C. The consumer first reads the stream window row S and the regular rational readback row R.

The modulus row M is consumed next, followed by the dyadic tolerance row D. These two rows are the only places where the packet records the stage bound and radius naming needed by the Cauchy comparison.

Only after those reads can the consumer reach Q and then V. Transport and provenance through H, C, P, N preserve the same order, so no detached criterion route exists. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion modulus route`

Theorem 1611.4 (Regular Cauchy criterion window extraction)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. Every accepted criterion request extracts the finite* DyadicRatCore$^{\uparrow}$ *tolerance window and the finite* StreamName$^{\uparrow}$ *stream window before it inspects the completion-facing row V or the Real boundary row A.*

Proof. Use the carrier inventory of Definition 1611.1. The only stream-facing coordinate is S, and the only dyadic tolerance coordinate is D.

Apply the NameCert obligations of Definition 1611.2. Stream-readback alignment requires the request to display S before the regular rational row R is used, while modulus-tolerance exposure requires the requested radius to pass through D.

Finally apply the route of Definition 1611.3. The completion-facing row V and the Real boundary row A occur downstream of S, R, M, D, Q, so both finite windows are extracted before either downstream row is inspected. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion window extraction`

Theorem 1611.5 (Regular Cauchy criterion RegSeqRat route)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. Every Real-facing criterion route factors through the ordered rows*

$$D \longmapsto S \longmapsto R \longmapsto A.$$

Here D is the DyadicRatCore$^{\uparrow}$ *tolerance row, S is the* StreamName$^{\uparrow}$ *finite-window row, R is the* RegSeqRat$^{\uparrow}$ *readback row, and A is the* Real$^{\uparrow}$ *boundary row served by the* RealCauchyCriterion$^{\uparrow}$ *entry.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion regseqrat route`

Proof. Project the carrier in Definition 1611.1. The packet exposes the dyadic tolerance row D, stream window row S, regular rational readback row R, and Real-facing boundary row A as distinct coordinates.

The NameCert obligations of Definition 1611.2 force stream-readback alignment and modulus-tolerance exposure. Thus the dyadic tolerance row names the requested radius, the stream window row fixes the inspected finite segment, and the RegSeqRat row reads that segment as regular rational data.

The modulus route of Definition 1611.3 places the Real-facing boundary after the criterion and convergence-criterion rows. Since those rows are reached only after D, S, R have been displayed, every Real-facing route passes through the DyadicRatCore, StreamName, RegSeqRat, and Real rows in the stated order. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterionRegSeqRatRoute`

Theorem 1611.6 (Regular Cauchy criterion StreamName-RegSeqRat scope)**.** *Let $K = (S, R, M, D, Q, V, A, H, C,$ be an accepted* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *carrier. Any criterion request must expose* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyModulus}^{\uparrow}$ *bounds, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance naming before it can reach the* $\mathsf{CauchyCriterion}^{\uparrow}$ *row Q, the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row V, or the* $\mathsf{Real}^{\uparrow}$ *boundary row A.*

Proof. Start from the carrier definition. The rows S, R, M, D are the only coordinates for finite stream windows, regular rational readback, modulus bounds, and dyadic tolerance naming.

Apply the NameCert obligations of Definition 1611.2. They require stream-readback alignment and modulus-tolerance exposure, so a criterion request cannot inspect R without first displaying S, and cannot export a Cauchy comparison without first displaying M and D.

The modulus route of Definition 1611.3 and the RegSeqRat route of Definition 1611.5 place Q, V, A downstream of the displayed window, readback, modulus, and tolerance rows.

Finally use the Real seal nonescape theorem. The Real boundary contributes no quotient stream, selected limit, host equality, trichotomy, choice schedule, or ambient completeness coordinate, so the scope of every criterion request remains the finite StreamName-to-RegSeqRat route through S, R, M, D before Q, V, A. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion streamname regseqrat scope`

Theorem 1611.7 (Regular Cauchy criterion Real seal nonescape)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *carrier. Every Real-facing read of K reaches the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary only after the route through S, R, M, D, Q, V has been replayed. The exported surface contains no quotient stream, selected limit, host real equality, trichotomy, countable-choice selector, or ambient Real completeness assertion.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion real seal nonescape`

Proof. The Real boundary row A is part of the carrier, but its accessibility is ordered by the continuation row C. A Real-facing read must therefore replay the stream, readback, modulus, dyadic, criterion, and convergence-criterion rows before A is available.

The transport row H identifies componentwise changes only along the displayed rows. It cannot add a quotient stream, selected limit, host equality, trichotomy, choice schedule, or completeness theorem, because no such coordinate appears in the carrier.

The provenance and local naming rows P, N name the same finite packet. Hence the Real seal is a downstream boundary for the finite regular Cauchy criterion and not an escape into ambient Real completeness. □

Theorem 1611.8 (Regular Cauchy criterion window obligations)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. Every criterion request exposes a finite* StreamName$^{\uparrow}$ *window, a* RegSeqRat$^{\uparrow}$ *readback row, and a* DyadicRatCore$^{\uparrow}$ *tolerance row before it may inspect the CauchyCriterion, CauchyConvergenceCriterion, or Real boundary rows. The exposed window contains only displayed stage bounds, rational observations, and dyadic radii; it contains no quotient stream, selected tail, or ambient completeness witness.*

Proof. Project the carrier. The only finite-window coordinate is S, the only regular rational readback coordinate is R, and the only dyadic tolerance coordinate is D.

Apply the NameCert obligations. Stream-readback alignment forces S before R, while modulus-tolerance exposure forces every requested radius through D before a criterion comparison is exported.

The modulus route places Q, V, A downstream of S, R, M, D. Therefore every criterion request exposes the stated finite window before it can read the criterion, convergence, or Real boundary rows. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion window obligations`

Theorem 1611.9 (Regular Cauchy criterion tail transport)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P, N)$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. Componentwise* hsame *transport through H and displayed* Cont *replay through C preserve the tail criterion rows S, R, M, D, Q, V, A: a transported request still reads the same StreamName window, RegSeqRat readback, modulus bound, dyadic tolerance, CauchyCriterion row, CauchyConvergenceCriterion row, and Real boundary in the displayed order.*

Proof. Read the transport row H. It is componentwise over the stream, readback, modulus, dyadic, criterion, convergence criterion, Real boundary, continuation, provenance, and local naming rows.

Read the replay row C. It records the same order used by the original criterion request, from stream windows through regular readback, modulus lookup, dyadic tolerance naming, criterion testing, convergence repacking, and Real boundary.

Since neither H nor C contains a quotient stream, selected limit, or ambient completeness coordinate, transport can only preserve the displayed tail criterion rows and cannot replace them by an external tail principle. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterion tail transport`

Theorem 1611.10 (Regular Cauchy criterion Cauchy equivalence surface)**.** *Let $K = (S, R, M, D, Q, V, A, H, C, P,$ be an accepted* RegularCauchyCriterion$^{\uparrow}$ *carrier. The only exported equivalence surface between Cauchy-facing and convergence-facing reads is the pair of rows Q and V, namely the* CauchyCriterion$^{\uparrow}$ *row and the* CauchyConvergenceCriterion$^{\uparrow}$ *row. The carrier exports no selected limit, quotient stream equality, host real equality, or ambient* Real$^{\uparrow}$ *completeness as part of this equivalence surface.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCriterionUp.RegularCauchyCriterionCauchyEquivalence encode append`

Proof. Inspect the carrier coordinates. The Cauchy-facing test is stored in Q, and the convergence-facing repacking of that same finite test is stored in V.

Apply criterion exactness from the NameCert obligations. It states that Q is the only exported Cauchy test and V is the only convergence-facing repacking available from the packet.

The Real seal nonescape theorem excludes selected limits, quotient streams, host equality, choice, and ambient completeness from the Real boundary. Hence the exported equivalence surface is exactly Q, V and not an external completion principle. □

Theorem 1611.11 (Regular Cauchy criterion public route)**.** *Every public* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *consumer reads the displayed rows in the order*

$$S \longmapsto R \longmapsto M \longmapsto D \longmapsto Q \longmapsto V \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route packages the finite window obligations, tail transport, Cauchy equivalence surface, Real seal nonescape, and ledger refusal rows already exposed by the chapter. It cannot export quotient streams, selected limits, host equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-window surface. Apply Definition 1611.8 to expose the StreamName window, RegSeqRat readback, and DyadicRatCore tolerance row before any criterion-facing read proceeds.

Next apply Definition 1611.9. Componentwise hsame transport and displayed Cont replay preserve the same ordered tail route through S, R, M, D, Q, V, A and add no quotient-stream or selected-limit coordinate.

The Cauchy-facing and convergence-facing sides meet through Definition 1611.10. The only equivalence surface is Q, V, and the Real-facing boundary is kept behind the same route by Definition 1611.7.

Finally use the obligation inventory of Definition 1611.2. The structural rows H, C, P, N transport, replay, source, and name the public route, while the ledger refusal clauses leave quotient streams, selected limits, host equality, trichotomy, countable choice, and ambient Real completeness outside the exported surface. □

Theorem 1611.12 (Regular Cauchy criterion bridged consumer)**.** *Every downstream Real-and-completion consumer may read a public* $\mathsf{RegularCauchyCriterion}^{\uparrow}$ *packet only through the bridge-facing route of Definition 1611.11. The readable surface is the finite chain through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, *the* $\mathsf{Real}^{\uparrow}$ *seal row, and the structural rows* H, C, P, N*; it is not a quotient stream, selected limit, host equality, trichotomy, countable choice, or ambient completion theorem.*

Proof. Start with the public route. It exposes the rows $S, R, M, D, Q, V, A, H, C, P, N$ in order and leaves no separate consumer coordinate outside that route.

The finite-window and tail-transport obligations then force every Real-and-completion read to pass through the StreamName window, RegSeqRat readback, CauchyModulus bound, DyadicRatCore tolerance, and CauchyCriterion row before the Real seal is inspected.

Finally apply Real seal nonescape. Since the accepted carrier has no quotient-stream, selected-limit, host-equality, choice, trichotomy, or ambient-completion coordinate, the bridge-facing consumer receives only the displayed finite criterion route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCriterion}^{\uparrow}}, \mathsf{bridgedClosed})$

Scope closed: Definition 1611.1, Definition 1611.2, Definition 1611.3, Definition 1611.6, Definition 1611.7, Definition 1611.8, Definition 1611.9, Definition 1611.10, Definition 1611.11, and Definition 1611.12 bind the finite route from StreamName windows through RegSeqRat readback, CauchyModulus bounds, DyadicRatCore tolerances, CauchyCriterion testing, CauchyConvergenceCriterion repacking, transport, the Real boundary, and bridge-facing Real-and-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a Lean standard bridge for the public RegularCauchyCriterion route to a downstream Real-and-completion consumer.

Constructive story: A RegularCauchyCriterion$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, a CauchyModulus row, DyadicRatCore tolerances, CauchyCriterion testing, CauchyConvergenceCriterion repacking, a RealCauchyCriterion and RealUp boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1612

A Concrete Naming Certificate for $\mathsf{SeparatedCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Kelley-style separated uniform reasoning, the $\mathsf{SeparatedCompletion}^{\uparrow}$ packet records the finite NameCert boundary at which a metric-completion route becomes separated without becoming a quotient completion. It consumes the metric-completion surface of chapter 1506, the dense embedding route of chapter 1512, the extension-and-uniqueness surface of chapter 397, the zero-distance classifier of chapter 1411, the Hausdorff boundary of chapter 402, and the regular-Cauchy separated route of chapter 1513. The L10 spine supplying finite names is the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route of chapter 291, chapter 126, chapter 301, and chapter 38.

The packet is a general separated-completion interface. It exposes source metric rows, dense embedding rows, completion handoff rows, separated classifier rows, uniqueness-boundary rows, transport rows, continuation replay, package provenance, and local naming. It exports no quotient completion, no selected representative, no host equality of completed points, no countable-choice schedule, no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, and no hidden complete metric space.

Definition 1612.1 (SeparatedCompletion carrier)**.** *A* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, D, C, Z, U, H, R, T, P, N)$$

with the following displayed rows:

1. *M is the source metric row together with the* $\mathsf{MetricCompletion}^{\uparrow}$ *admission boundary for completion-facing reads.*
2. *D is the dense* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, recording how source observations enter the completion interface before any separated comparison is public.*
3. *C is the completion handoff row, read through the finite* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *L10 naming spine.*
4. *Z is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row that compares displayed completion readings by zero-distance evidence rather than by host equality.*
5. *U is the uniqueness boundary, supplied by the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, *and* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *rows over the same source, dense embedding, completion handoff, and separated classifier.*
6. *H records componentwise* hsame *transport for M, D, C, Z, U and for the local structural rows.*

7. *R records displayed* Cont *replay from source metric observations through dense embedding, completion handoff, separated classification, and uniqueness boundary.*
8. *T is the local transport ledger that states that hsame transport and Cont replay preserve the same ordered completion route.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, selected completion representatives, host equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external complete metric space.*

Lean4 (definition) `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionCarrier`

Theorem 1612.2 (SeparatedCompletion NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier $S = (M, D, C, Z, U, H, R, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the displayed source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, hsame transport, Cont replay, package provenance, and local naming rows;*
2. *dense-source compatibility, requiring every completion-facing consumer to read M and D before the handoff row C;*
3. *separated-classifier exactness, requiring every comparison of completion readings to pass through Z after C has been displayed;*
4. *uniqueness-boundary exactness, requiring every asserted uniqueness read to be exhausted by U over the same M, D, C, Z route;*
5. *transport replay, requiring H, R, T to preserve the ordered route under componentwise* hsame *transport and displayed* Cont *replay;*
6. *ledger non-escape, excluding quotient completions, selected representatives, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden complete metric spaces.*

Proof. Carrier admission is projection from Definition 1612.1. The tuple displays the source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, structural transport rows, provenance, and local naming.

Dense-source compatibility is the first dependency order. The rows M and D must be present before C can expose any completion-facing read, so no consumer can replace the dense embedding by a selected representative or external completion point.

Separated-classifier exactness is isolated in Z. Once C has been displayed, comparison of completion readings is admitted only through the zero-distance classifier row, not through host equality of completed points.

The uniqueness boundary is the row U. It is read over the same M, D, C, Z route and is therefore a local ledger comparison, not a quotient-completion uniqueness theorem outside the packet.

Finally, H, R, T, P, N record transport, replay, route preservation, provenance, and local naming for exactly those rows. Since the carrier has no quotient, representative, choice, ambient-completeness, host-equality, or external-complete-space coordinate, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionCarrier semantic name certificate`

Theorem 1612.3 (SeparatedCompletion source-metric obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every* BHist *or* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *read that exposes completion-facing metric data must first enter through the source metric row M. The later rows $D, C, Z, U, H, R, T, P, N$ may transport, replay, source, and name that read, but none of them supplies an independent source metric, an external complete metric space, a quotient completion, or host equality of completed points.*

Proof. Project the carrier of Definition 1612.1. The coordinate M is the displayed source metric row and is the only carrier coordinate with source-metric status.

Apply Definition 1612.2. Carrier admission and dense-source compatibility place M before the dense row D and the completion handoff C, while the remaining rows only compare, transport, replay, source, and locally name the same displayed route.

Since the carrier has no external complete-space, quotient-completion, host-equality, or second source-metric coordinate, every metric read is a read through M preserved by H, R, T, P, N. □

Theorem 1612.4 (SeparatedCompletion dense-embedding obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every dense source read consumed by a completion-facing row must pass through the dense embedding row D before the completion handoff row C is public. No row of the packet may treat C, Z, or U as a dense embedding unless the preceding M, D route has already been displayed in the same* BHist *packet.*

Proof. Read the displayed tuple in Definition 1612.1. The row D is the dense $\mathsf{CompletionEmbedding}^{\uparrow}$ row, and C is the downstream completion handoff row.

Use the dense-source compatibility clause of Definition 1612.2. It requires every completion-facing consumer to read M and D before C, so the handoff cannot be used as a substitute dense embedding.

The separated classifier Z and uniqueness row U are later rows over the same route. Thus a dense source read reaches C, Z, U only after the displayed M, D entrance has been recorded in BHist and replayed by the local Cont data. □

Theorem 1612.5 (SeparatedCompletion zero-distance obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every separated comparison must read the zero-distance classifier row Z before any uniqueness-boundary row U is consumed. The comparison is therefore a finite classifier route through M, D, C, Z, U with* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming, not host equality, selected representatives, quotient completion equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The row Z is the only separated zero-distance classifier row, while U records uniqueness over the same source, dense, completion, and classifier route.

Apply separated-classifier exactness from Definition 1612.2. A comparison of completion readings reaches Z only after C has been displayed, and uniqueness-boundary exactness reads U over that same M, D, C, Z route.

The structural rows H, R, T, P, N preserve, replay, source, and name the route without adding a second comparison principle. Because the excluded host-equality, representative, quotient, and ambient-completeness objects are absent from the carrier, every separated comparison must pass through Z before U. □

Definition 1612.6 (SeparatedCompletion kernel route)**.** *The separated-completion kernel route of an accepted carrier $S = (M, D, C, Z, U, H, R, T, P, N)$ is the ordered readback*

$$M, D \longmapsto C \longmapsto Z, U \longmapsto H, R, T, P, N.$$

Thus every completion-facing consumer first reads the source metric row M and dense embedding row D, then the completion handoff row C, then the separated classifier and uniqueness rows Z, U, and only then the transport, replay, provenance, and local naming rows H, R, T, P, N.

Theorem 1612.7 (SeparatedCompletion separated-limit handoff)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every separated completion comparison reaches the separated classifier row Z and uniqueness boundary row U only after the source metric row M, dense embedding row D, and completion handoff row C have been displayed.*

Proof. Use the carrier projection of Definition 1612.1 and the obligation order of Definition 1612.2. The rows M and D are the source and dense-entry surface, while C is the only completion handoff row.

The separated classifier Z and uniqueness boundary U are downstream rows in the same carrier. Since the carrier has no separate comparison coordinate, a separated-limit handoff cannot inspect Z or U before the M, D, C route has been read. □

Lean4 ✓ `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionCarrier separated limit handoff`

Theorem 1612.8 (SeparatedCompletion L10 readback exhaustion)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier $S = (M, D, C, Z, U, H, R, T, P, N)$, the handoff row C is exhausted by the finite L10 readbacks through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It exports no quotient completion, selected representative, host equality of completed points, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project C from the carrier of Definition 1612.1. The definition identifies C as the completion handoff row read through the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ naming spine.

Apply Definition 1612.2. The ledger non-escape clause states that quotient completions, selected representatives, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and hidden complete metric spaces are absent from the local surface.

Therefore every L10 readback of C is exhausted by the displayed dyadic, stream, regular-rational, and real-name route, and no additional completion principle is exported. □

Theorem 1612.9 (SeparatedCompletion obligation surface)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *read starts with the displayed source row M, dense row D, completion handoff C, separated classifier Z, and uniqueness row U, then preserves that route through* BHist, hsame, Cont, *provenance, and local naming data in H, R, T, P, N. The surface exports no quotient completion, selected representative, host equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier. The rows M, D, C, Z, U are the only source, dense, completion, separated-classifier, and uniqueness rows available to the local certificate.

The structural rows H, R, T then transport and replay the same ordered route through hsame and Cont inside BHist. The provenance and local naming rows P, N attach the route to $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ without adding another completion object.

Since the carrier has no quotient-completion, representative-selection, host-equality, countable-choice, or Real-completeness coordinate, every obligation read is exhausted by the displayed rows. □

Theorem 1612.10 (SeparatedCompletion source-density triad)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted $\mathsf{SeparatedCompletion}^{\uparrow}$ carrier. Every separated-completion consumer must read the source metric row M, dense embedding row D, and completion handoff row C before it can read the separated classifier Z, the uniqueness boundary U, or any Real-facing seal exported by the packet.*

Proof. Project the carrier of Definition 1612.1. Its first three analytic rows are exactly M, D, C, while Z and U are later rows over the same displayed packet.

Apply Definition 1612.2. Dense-source compatibility requires M, D before C, separated-classifier exactness requires C before Z, and uniqueness-boundary exactness reads U over the same M, D, C, Z route.

The Real-facing seal is part of the completion handoff read through C. Hence no separated classifier, uniqueness, or Real-facing consumer can bypass the source-density triad. □

Theorem 1612.11 (SeparatedCompletion source-density factorization)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted $\mathsf{SeparatedCompletion}^{\uparrow}$ carrier. Every completion-facing read that claims source density factors through the ordered prefix M, D, C before the separated classifier Z, uniqueness boundary U, transport row H, replay row R, route ledger T, provenance row P, or local naming row N is consumed. The factorization is the dense-source compatibility clause of the local NameCert obligations, not a quotient-completion read, selected representative, host equality of completed points, countable-choice schedule, or ambient Real-completeness assertion.*

Proof. Project the carrier of Definition 1612.1. The source metric row is M, the dense embedding row is D, and the completion handoff row is C; the remaining rows Z, U, H, R, T, P, N are downstream of that prefix.

Apply Definition 1612.2. Its dense-source compatibility clause requires every completion-facing consumer to read M and D before the handoff row C is public.

The structural rows H, R, T, P, N can only transport, replay, preserve, source, and name the same displayed route. Since the carrier has no quotient-completion, selected-representative, host-equality, choice, or ambient-completeness coordinate, source density can only be read through M, D, C. □

Theorem 1612.12 (SeparatedCompletion uniqueness boundary scope)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted $\mathsf{SeparatedCompletion}^{\uparrow}$ carrier. The uniqueness boundary row U is scoped only over the already displayed route M, D, C, Z and the structural rows H, R, T, P, N. A consumer may read U as the local separated-completion uniqueness boundary, but cannot use it as a quotient completion, a selected representative comparison, a host equality of completed points, a countable-choice selector, or an ambient Real-completeness theorem.*

Proof. First read the source-density factorization of Definition 1612.11. It places every completion-facing read on the prefix M, D, C.

Next apply the separated-classifier exactness and uniqueness-boundary exactness clauses of Definition 1612.2. They place Z after the handoff row C and read U over the same M, D, C, Z route.

Finally attach H, R, T, P, N. These rows keep the uniqueness comparison inside displayed BHist, hsame, Cont, Pkg, and local NameCert data. No listed row supplies an external completion object, a selected representative, host equality, choice, or ambient completeness. □

Theorem 1612.13 (SeparatedCompletion Real-seal nonescape)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* SeparatedCompletion$^{\uparrow}$ *carrier. Any Real-facing seal exported by the packet is read through the completion handoff row C after M, D and before the separated uniqueness surface Z, U, H, R, T, P, N is consumed. The seal remains a finite route through the displayed* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it cannot escape into quotient completion, selected representatives, host equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Begin with the carrier row C. By Definition 1612.1, C is the completion handoff read through the finite DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ L10 naming spine.

Use Definition 1612.11. The handoff row C is reachable only after the source metric row M and dense embedding row D have been displayed.

Use Definition 1612.12. The later separated rows Z, U, H, R, T, P, N preserve and name the same handoff route rather than adding an external Real-completion principle. Therefore every Real-facing seal stays inside the displayed carrier and cannot introduce the excluded quotient, representative, host-equality, choice, or completeness objects. □

Theorem 1612.14 (SeparatedCompletion seal uniqueness)**.** *For accepted packets $S = (M, D, C, Z, U, H, R, T, P, N$ and $S' = (M, D, C, Z, U', H', R', T', P, N')$ with the same displayed source, dense, completion, separated-classifier, and provenance rows, any equality-of-export read between their completion seals is determined by the local uniqueness rows U, U' together with the* BHist*,* hsame*,* Cont*, and* NameCert *data in H, R, T, N and H', R', T', N'. It is not a quotient completion, selected representative comparison, host equality, countable-choice selection, or ambient Real-completeness conclusion.*

Proof. The shared rows M, D, C, Z, P fix the source metric, dense embedding, completion handoff, separated classifier, and provenance surface for both packets.

Any remaining comparison must therefore inspect U, U' and the corresponding transport, replay, route-preservation, and local naming rows. These rows compare completion exports only over the already displayed route.

No carrier coordinate supplies quotient equality, selected representatives, host equality, choice, or Real completeness. Thus seal uniqueness is local to U, U' and the displayed BHist, hsame, Cont, and NameCert data. □

Theorem 1612.15 (SeparatedCompletion diagonal classifier obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* SeparatedCompletion$^{\uparrow}$ *carrier. Every diagonal classifier read comparing a completion-facing row with itself is forced through the separated classifier row Z after the dense route M, D, C has been displayed. The read is a* BHist*-visible zero-distance comparison preserved by* hsame *and* Cont *data, not a host equality, quotient-completion equality, selected representative, countable-choice schedule, or ambient* Real$^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 1612.1. The only separated comparison coordinate is Z, and it appears downstream from the source metric row M, dense embedding row D, and completion handoff row C.

Apply the NameCert obligations of Definition 1612.2. They require dense-source compatibility before C is consumed and separated-classifier exactness before any comparison is public.

The diagonal case does not add a new coordinate. Transport and replay are still the displayed rows H, R, T, so the comparison remains inside the local BHist, hsame, Cont, Pkg, and NameCert surface. Since the excluded objects have no carrier field, the diagonal classifier obligation is forced through Z. □

Theorem 1612.16 (SeparatedCompletion separated-limit handoff obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P,$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every separated-limit handoff read must pass through the source metric row M, dense embedding row D, completion handoff row C, separated classifier row Z, and uniqueness boundary row U before the structural rows H, R, T, P, N are consumed. No consumer can obtain that handoff from a quotient completion, a selected representative, host equality of completed points, countable choice, or an ambient Real-completeness coordinate.*

Proof. Start with the dense route. The carrier exposes M, D, C as the only entrance to completion-facing reads, and Definition 1612.19 makes every such read factor through that entrance.

Next read the separated boundary. The classifier row Z and uniqueness row U are later rows in the same accepted packet, so a separated-limit handoff can consume them only after the dense handoff is visible.

Finally attach H, R, T, P, N. These rows transport, replay, source, and name the same route rather than adding another completion object. The carrier therefore supplies no quotient, selected representative, host equality, choice, or ambient completeness coordinate from which a bypassing handoff could be read. □

Theorem 1612.17 (SeparatedCompletion nonescape obligation)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier $S = (M, D, C, Z, U, H, R, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *surface exports only the displayed rows $M, D, C, Z, U, H, R, T, P, N$. It exports no quotient completion, no selected representative, no countable-choice schedule, no host equality of completed points, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Use the carrier admission and obligation surface of Definition 1612.1 and Definition 1612.2. They enumerate exactly the rows available to the accepted packet.

The dense route M, D, C, the separated classifier Z, and the uniqueness boundary U account for the analytic part of the surface. The structural rows H, R, T, P, N account for componentwise hsame transport, displayed Cont replay, route preservation, provenance, and local naming.

There is no remaining coordinate to project a quotient completion, representative selector, choice schedule, host equality, or ambient Real-completeness theorem. Hence every exported obligation remains inside the displayed finite carrier. □

Theorem 1612.18 (SeparatedCompletion window transport)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. If a completion-facing window read is transported componentwise by the displayed* hsame *row H, then the transported read remains on the same M, D, C, Z, U route and is replayed by the displayed* Cont *row R and transport ledger T before the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *row N is inspected. The transport introduces no quotient completion, selected representative, host equality, countable choice, or ambient Real completeness.*

Proof. Start from the window read over M, D, C. These rows fix the source metric, dense source, and completion handoff before any separated comparison is available.

Apply the displayed hsame row H componentwise. The transport is a movement of the already displayed BHist rows, not a selection of a new representative or a quotient completion class.

The Cont replay row R and the transport ledger T then replay the same route through Z, U, P, N. Since all excluded principles are absent from the carrier, transported windows remain inside the local NameCert surface. □

Theorem 1612.19 (SeparatedCompletion dense factorization)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* SeparatedCompletion$^{\uparrow}$ *carrier. Every completion-facing consumer read factors through the finite dense route*

$$M, D \longmapsto C \longmapsto Z \longmapsto U \longmapsto H, R, T, P, N.$$

Thus source metric data enter the separated-completion boundary only through the displayed metric-completion and dense-embedding rows before any separated classifier or uniqueness boundary can be consumed.

Lean4 ✓ `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionDenseFactorization`

Proof. Begin with an accepted consumer read. Carrier admission gives the rows $M, D, C, Z, U, H, R, T, P, N$ and no other completion coordinate.

The source part is M, D. The metric-completion row admits the request, and the dense embedding row carries source observations into the completion-facing interface.

The handoff row C is read only after that source part is fixed. Its L10 route is the finite DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ naming spine, so the consumer cannot replace it by a quotient stream or selected host limit.

The separated classifier Z and uniqueness boundary U are downstream of the same handoff. Transport, replay, provenance, and local naming in H, R, T, P, N preserve this route and add no external completion object. □

Theorem 1612.20 (SeparatedCompletion uniqueness boundary)**.** *For accepted packets*

$$S = (M, D, C, Z, U, H, R, T, P, N) \quad \textit{and} \quad S' = (M, D, C, Z, U', H', R', T', P, N')$$

over the same source metric row, dense embedding, completion handoff, separated classifier, and package provenance, every comparison between their completion-facing exports is exhausted by the uniqueness rows U, U' together with the displayed transport and replay rows H, R, T and H', R', T'.

Proof. The shared coordinates M, D, C, Z, P fix the metric-completion source, the dense embedding route, the completion handoff, the separated classifier, and the provenance surface for both packets.

The remaining comparison data are U, H, R, T, N and U', H', R', T', N'. The uniqueness rows compare completion-facing presentations over the same shared route, while the transport and replay rows display the permitted movements of that route.

No further field remains from which a consumer could read quotient-completion equality, a selected representative, host equality, countable choice, ambient Real$^{\uparrow}$ completeness, or a hidden complete metric space. The uniqueness judgement is therefore exactly the displayed local boundary. □

Theorem 1612.21 (SeparatedCompletion Cauchy-equivalence source)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every comparison between separated-completion exports first reads the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *source rows carried by M, D, C before the separated classifier row Z is available. The comparison route is therefore sourced in displayed Cauchy-equivalence data over the same source metric and dense embedding, not in quotient completions, selected representatives, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a comparison read of an accepted packet. Carrier admission exposes $M, D, C, Z, U, H, R, T, P, N$ and no other completion coordinate.

The source of the read is the M, D, C route. The metric row and dense embedding row fix the source side, while the completion handoff records the regular-Cauchy separated completion and Cauchy-sequence equivalence data consumed by the comparison.

Only after that source route is displayed may Z classify separated completion readings. Since the carrier has no quotient-completion, selected-representative, host-equality, choice, Real-completeness, or hidden-complete-space coordinate, no comparison can bypass the displayed Cauchy-equivalence source. □

Theorem 1612.22 (SeparatedCompletion Cauchy reflector obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every Cauchy-reflector read exposed by S first displays the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route and the dense embedding route inside the M, D, C rows, then reads the separated comparison row Z. The obligation is a* BHist *and* $\mathsf{NameCert}$ *row route through $M, D, C, Z, H, R, T, P, N$, not parameter transport over an external completion object.*

Proof. Start from Definition 1612.21. It places every comparison behind the displayed Cauchy-equivalence and regular separated-completion source rows carried by M, D, C.

The reflector read is therefore not a free-standing map. Its source side is the displayed CauchyCompletionReflector row, while the dense embedding row D records the admission route into the completion handoff C.

Only after those rows are visible can Z compare separated completion readings. The structural rows H, R, T, P, N transport, replay, source, and locally name the same route, so the Cauchy reflector obligation is exhausted by the displayed packet. □

Theorem 1612.23 (SeparatedCompletion source-density obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every completion-facing obligation first reads the source metric row M and dense embedding row D, then passes through the completion handoff C before the separated zero-distance classifier row Z or the uniqueness row U is public. The source-density route is recorded by* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and it admits no quotient completion, selected representative, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted packet of Definition 1612.1. The source metric row M and dense embedding row D are the displayed entrance into every separated completion obligation.

Use dense factorization, Definition 1612.19. It places the completion handoff C downstream from the source metric and dense embedding route, so a consumer cannot start from separated comparison or uniqueness data alone.

The classifier and uniqueness rows are therefore later reads. The row Z compares only after the source-density route has been displayed, and U records uniqueness over that same route.

The structural rows H, R, T, P, N preserve, replay, source, and name the route. Since the carrier has no coordinate for quotient completions, selected representatives, host equality, countable choice, or ambient Real completeness, those objects are refused by the obligation surface. □

Theorem 1612.24 (SeparatedCompletion zero-distance classifier obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. The separated zero-distance classifier row Z is available only after the source metric route M, dense embedding route D, and completion handoff C have been displayed. It classifies completion reads through the separated zero-distance row of the packet, not through quotient completion equality, selected representatives, host equality of completed points, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the Cauchy reflector obligation of Definition 1612.22. It exposes the reflector route inside M, D, C before separated comparison can be read.

Apply zero-distance factorization, Definition 1612.29. The theorem identifies Z as the finite separated comparison boundary over the already displayed completion handoff.

The uniqueness and structural rows do not create a second classifier. The row U reads uniqueness downstream from Z, while H, R, T, P, N only transport, replay, source, and name the same comparison route.

Thus every zero-distance classifier obligation is carried by $M, D, C, Z, U, H, R, T, P, N$. The excluded quotient, representative, host-equality, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 1612.25 (SeparatedCompletion completion reflector route)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every completion reflector handoff exported by S follows the route from the source metric and dense embedding rows M, D through the completion handoff C, the separated zero-distance classifier Z, the uniqueness boundary U, and the structural rows H, R, T, P, N. A downstream reflector consumer receives that finite route only; it receives no quotient completion, selected representative, host equality, countable choice, hidden complete metric space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use the source-density obligation, Definition 1612.23, to fix the entrance of the reflector route at M, D, C. The completion reflector cannot be read before the source metric and dense embedding rows have been displayed.

Next use the zero-distance classifier obligation, Definition 1612.24. It makes Z the separated comparison boundary for the same handoff route, rather than an equality imported from a quotient completion.

Attach the existing reflector handoff through Definition 1612.39. The uniqueness row U and the rows H, R, T, P, N preserve, replay, source, and name the reflector route for downstream completion consumers.

No remaining coordinate names a selected representative, host equality, countable-choice schedule, hidden complete metric space, or ambient Real completeness theorem. The completion reflector handoff is therefore exactly the displayed finite packet route. □

Theorem 1612.26 (SeparatedCompletion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier remains downstream of the displayed* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows carried through M, D, C. The seal may be consumed by the separated classifier Z and uniqueness boundary U, but it cannot expose host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient completion equality, selected representatives, or countable choice.*

Proof. Read the source route M, D, C first. These rows carry the metric-completion source, dense embedding, regular rational readback, finite stream windows, and dyadic tolerance evidence before any separated comparison is inspected.

The $\mathsf{Real}^{\uparrow}$ seal is reached only after that finite route. It records terminal readback data for the displayed completion route rather than an ambient completeness theorem.

Now apply the separated classifier and uniqueness rows Z, U. They consume the seal downstream of the same packet, while H, R, T, P, N merely transport, replay, source, and name the route. Since no excluded real or completion principle is a carrier coordinate, the seal cannot escape the displayed rows. □

Theorem 1612.27 (SeparatedCompletion universal route exhaustion)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier, the universal completion route is exhausted by the displayed rows*

$$M, D, C, Z, U, H, R, T, P, N.$$

The route exports dense-source admission, Cauchy-equivalence sourcing, separated comparison, uniqueness boundary, transport, replay, provenance, and local naming, but it exports no quotient completion, selected representative, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or additional universal completion principle outside the carrier.*

Proof. Project the carrier of Definition 1612.1. Its displayed rows are exactly the source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, transport, replay, provenance, and local naming rows.

Use dense factorization and the Cauchy-equivalence source theorem. They place the universal read behind M, D, C before Z or U can be inspected.

The separated comparison and uniqueness rows then account for the completion-facing universal property available to consumers. No further coordinate remains for quotient completion, representative selection, host equality, choice, or ambient completeness.

Finally H, R, T, P, N preserve the same row list. Thus the displayed carrier exhausts the universal route and exports no completion principle beyond the finite packet. □

Theorem 1612.28 (SeparatedCompletion universal separated factorization)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every universal completion read accepted through* S *passes through* M, D, C *before the separated classifier* Z *or the uniqueness row* U *may be consumed. The separated equality read is zero-distance evidence over the same displayed route, not quotient equality, host equality of completed points, selected representative equality, or an ambient completion equality.*

Proof. Project the carrier of Definition 1612.1. The only universal completion route available to the packet is the displayed row list $M, D, C, Z, U, H, R, T, P, N$.

Use dense factorization, Definition 1612.19, and universal route exhaustion, Definition 1612.27. Together they place every universal read behind the source metric, dense embedding, and completion handoff rows M, D, C.

Now apply zero-distance factorization, Definition 1612.29. The separated classifier row Z reads equality as displayed zero-distance evidence over that same handoff, while U reads uniqueness only after the classifier route is exposed.

Finally use nonescape, Definition 1612.33. Since quotient equality, host equality, selected representatives, and ambient completion equality have no field in the accepted carrier, they cannot replace the displayed zero-distance route. □

Theorem 1612.29 (SeparatedCompletion zero-distance factorization)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. The separated classifier route factors through the* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *zero-distance evidence displayed by M, C, Z. Thus a separated-completion equality read is a finite zero-distance handoff through the local packet, not a quotient completion, a selected representative, host equality, countable choice, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Lean4 ✓ `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionZeroDistanceFactorization`

Proof. Read the route in order. The row M supplies the metric-completion admission boundary, and C supplies the finite L10 handoff through regular rational readback toward the Real-facing seal.

The row Z then reads that handoff as $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance evidence. Its classifier compares displayed completion readings by zero distance after the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows have been reached through C.

Transport, replay, provenance, and local naming in H, R, T, P, N preserve the same factorization. No field remains for quotient completion equality, selected representatives, host equality, countable choice, or ambient Real completeness. □

Theorem 1612.30 (SeparatedCompletion limit-uniqueness export)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier $S = (M, D, C, Z, U, H, R, T, P, N)$, the public limit-uniqueness export is exactly the local boundary U over the already displayed M, D, C, Z route. The export contains no separate uniqueness principle over quotient completions, selected representatives, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric spaces.*

Proof. Carrier admission makes U the only uniqueness-boundary row in the packet. Its hypotheses and conclusion are read over the same source metric, dense embedding, completion handoff, and separated classifier rows M, D, C, Z.

The transport and replay rows H, R, T can move this boundary componentwise, but they do not create a second uniqueness coordinate. The package and local naming rows P, N record the same source list.

Therefore every public uniqueness export is the local U boundary over M, D, C, Z. The excluded quotient, representative, host-equality, choice, Real-completeness, and hidden-complete-space objects are absent from the carrier and cannot be exported. □

Theorem 1612.31 (SeparatedCompletion dense uniqueness interface)**.** *Let*

$$S = (M, D, C, Z, U, H, R, T, P, N) \quad \textit{and} \quad S' = (M, D, C, Z, U', H', R', T', P, N')$$

be accepted $\mathsf{SeparatedCompletion}^{\uparrow}$ *carriers with the same dense-source route M, D, C and the same zero-distance separated classifier Z. Then any completion-facing comparison between their exports exposes the same limit-uniqueness boundary through U, U' together with the displayed transport and replay rows. It exports no quotient representative, no selected completion point, no host equality of completed points, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Use the dense factorization of Definition 1612.19. Every completion-facing read of either packet first passes through the shared source metric, dense embedding, and completion handoff rows M, D, C.

Use the zero-distance factorization of Definition 1612.29. Since the separated classifier row Z is also shared, the comparison is read as zero-distance evidence over the same finite route rather than as host equality of completed points.

Apply the limit-uniqueness export of Definition 1612.30 to both packets. The only remaining uniqueness data are U, U' and the displayed transport and replay rows H, R, T and H', R', T'. The carrier has no quotient representative, selected completion point, choice schedule, or ambient completeness coordinate, so the dense uniqueness interface exports exactly that local boundary. □

Theorem 1612.32 (SeparatedCompletion source-comparison exhaustion)**.** *Let*

$$S = (M, D, C, Z, U, H, R, T, P, N) \quad \textit{and} \quad S' = (M, D, C, Z, U', H', R', T', P, N')$$

be accepted $\mathsf{SeparatedCompletion}^{\uparrow}$ *carriers over the same source metric row, dense embedding, completion handoff, separated classifier, and provenance surface. Every accepted comparison between their separated-completion exports is exhausted by the shared* M, D, C, Z *source route, by the uniqueness rows* U, U'*, and by the displayed transport and replay rows* H, R, T *and* H', R', T'*. The comparison exports no quotient completion, selected representative, host equality of completed points, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or hidden complete metric space.*

Proof. First use the dense factorization of Definition 1612.19. It sends every completion-facing read through the shared source metric row M, dense embedding row D, and completion handoff row C before any separated classifier or uniqueness boundary is consumed.

Next use the zero-distance factorization of Definition 1612.29. The shared classifier Z reads the comparison as zero-distance evidence over the same finite route, so the comparison cannot be replaced by host equality of completed points.

Now apply the limit-uniqueness export of Definition 1612.30 to both packets. The only remaining comparison rows are U, U' together with the displayed transport and replay ledgers.

Finally attach the Cauchy-equivalence source theorem, Definition 1612.21. It keeps the comparison sourced in the displayed Cauchy-equivalence and dense-completion data carried by M, D, C. Since the carriers contain no excluded completion coordinate, the listed rows exhaust the accepted comparison surface. □

Theorem 1612.33 (SeparatedCompletion non-escape)**.** *Every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier exports only the rows*

$$M, D, C, Z, U, H, R, T, P, N$$

of Definition 1612.1. No consumer of the packet can obtain a quotient completion, selected representative, host equality of completed points, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or hidden complete metric space.*

Proof. The NameCert obligations of Definition 1612.2 enumerate the carrier, dense-source, separated-classifier, uniqueness-boundary, transport-replay, and refusal rows available to the packet.

The dense factorization of Definition 1612.19 covers every completion-facing consumer read. Such a read must traverse $M, D, C, Z, U, H, R, T, P, N$ and cannot bypass the finite route through a host completion object.

The uniqueness boundary of Definition 1612.20 covers every comparison between completion-facing exports. Since the carrier has no additional coordinate beyond the displayed rows, none of the excluded quotient, representative, host-equality, choice, Real-completeness, or hidden-complete-space objects can be projected from an accepted packet. □

Theorem 1612.34 (SeparatedCompletion choice-free source route)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every accepted source-to-uniqueness read exposes the*

source metric row M, dense embedding row D, completion handoff row C, and separated classifier row Z before any uniqueness row U is available. The route is carried by displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows and cannot expose choice data, quotient completion data, selected representatives, or host equality of completed points.*

Proof. Start with the carrier of Definition 1612.1. Its displayed rows are $M, D, C, Z, U, H, R, T, P, N$, so the source metric, dense embedding, completion handoff, and separated classifier are present before the uniqueness boundary can be projected.

Apply the dense factorization theorem, Definition 1612.19. It fixes the order $M, D \mapsto C \mapsto Z \mapsto U$ for completion-facing reads.

Apply the zero-distance factorization of Definition 1612.29 and the source-comparison exhaustion theorem, Definition 1612.32. These results keep the comparison in displayed Cauchy and zero-distance data rather than in quotient or host equality data.

The structural support rows are exactly H, R, T, P, N: componentwise hsame transport, displayed Cont replay, route preservation, Pkg provenance, and local NameCert admission. Since the carrier contains no choice, quotient completion, selected-representative, or host-equality coordinate, the source route is choice-free. □

Theorem 1612.35 (SeparatedCompletion regular-Cauchy source route)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every regular-Cauchy source read used by the separated-completion boundary factors through the displayed source metric row M, dense embedding row D, completion handoff row C, and separated classifier row Z before the uniqueness row U and the structural rows H, R, T, P, N are consumed. The route may use the dense factorization of Definition 1612.19, the zero-distance factorization of Definition 1612.29, and the source-comparison exhaustion of Definition 1612.32, but it cannot add quotient completions, selected representatives, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric spaces.*

Proof. Start with carrier admission. The visible rows are exactly $M, D, C, Z, U, H, R, T, P, N$, so any regular-Cauchy source read must begin inside the displayed carrier inventory.

Apply dense factorization, Definition 1612.19. It fixes M, D as the source prefix and C as the completion handoff reached from that prefix.

Apply zero-distance factorization, Definition 1612.29. The separated classifier row Z reads equality through zero-distance evidence over the same handoff, not through host equality of completed points.

Finally use source-comparison exhaustion, Definition 1612.32. It leaves only U, H, R, T, P, N as the downstream uniqueness, transport, replay, route, provenance, and naming rows, and the excluded completion and choice objects have no coordinate in the carrier. □

Theorem 1612.36 (SeparatedCompletion Hausdorff uniqueness route)**.** *For accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carriers over displayed rows $M, D, C, Z, U, H, R, T, P, N$, every Hausdorff-style uniqueness read is exhausted by the dense source route M, D, C, the zero-distance separated classifier Z, the uniqueness boundary U, and the transport, replay, route, provenance, and local naming rows H, R, T, P, N. The read is a local separated-completion boundary over the theorems Definition 1612.19, Definition 1612.29, and Definition 1612.32; it is not a quotient completion, a selected representative, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden complete metric space.*

Proof. Read the dense source route first. By Definition 1612.19, a uniqueness consumer must pass through M, D, C before reaching any separated classifier or uniqueness row.

Read the separated classifier next. By Definition 1612.29, Z supplies the zero-distance route that replaces host equality inside the accepted packet.

The Hausdorff-style uniqueness boundary is then the local row U together with the structural rows H, R, T, P, N. These rows transport, replay, preserve, source, and name the same displayed route.

Source-comparison exhaustion, Definition 1612.32, closes the comparison surface. Since quotient completions, selected representatives, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and hidden complete metric spaces are absent from $M, D, C, Z, U, H, R, T, P, N$, they cannot enter the uniqueness read. □

Theorem 1612.37 (SeparatedCompletion metric-completion handoff)**.** *Every metric-completion handoff exported by an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier factors through the displayed rows*

$$M, D, C, Z, U, H, R, T, P, N.$$

The handoff uses the dense route of Definition 1612.19, the zero-distance route of Definition 1612.29, and the exhausted comparison surface of Definition 1612.32. It exports no quotient completion, selected representative, host equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric space.*

Proof. Project the accepted carrier. The metric-completion source row M, dense embedding row D, and completion handoff row C form the only displayed metric-completion prefix.

Use Definition 1612.19. It sends every completion-facing handoff through M, D, C before any separated or uniqueness boundary is available.

Use Definition 1612.29. The row Z is the separated comparison route available after the completion handoff, and it is finite zero-distance evidence rather than host equality.

Use Definition 1612.32. The remaining rows U, H, R, T, P, N exhaust the uniqueness, transport, replay, route, provenance, and naming surface. Because the carrier has no quotient-completion, selected-representative, host-equality, choice, Real-completeness, or hidden-complete-space coordinate, the metric-completion handoff exports only the displayed finite route. □

Theorem 1612.38 (SeparatedCompletion filtered limit boundary)**.** *For an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, every filtered limit-boundary read is exhausted by the Cauchy-equivalence source rows displayed by* M, D, C*, the zero-distance separated classifier row* Z*, the local uniqueness row* U*, and the transport and replay rows* H, R, T*. It exports no additional limit selector, quotient class, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the source side first. The Cauchy-equivalence source theorem, Definition 1612.21, places every comparison over the displayed M, D, C route.

Read the classifier next. The zero-distance factorization theorem, Definition 1612.29, makes Z the only separated comparison row available after the completion handoff.

Read the boundary row. The limit-uniqueness export theorem, Definition 1612.30, identifies U as the public uniqueness boundary over the same M, D, C, Z route.

Finally apply source-comparison exhaustion, Definition 1612.32. It includes the displayed hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows and excludes any extra selector, quotient class, host equality, choice schedule, or ambient Real-completeness coordinate. □

Theorem 1612.39 (SeparatedCompletion completion-reflector handoff)**.** *Every handoff from an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier to* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *or* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *factors through the same finite source route*

$$M, D, C, Z, U, H, R, T, P, N.$$

The handoff may use the source metric, dense factorization, zero-distance factorization, limit-uniqueness export, source-comparison exhaustion, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it cannot add a quotient completion, selected representative, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier and apply dense factorization from Definition 1612.19. The handoff begins with M, D, C, so both reflector-facing and universal-property-facing reads share the same source metric, dense embedding, and completion handoff.

Apply the zero-distance factorization of Definition 1612.29. The separated classifier Z is the only comparison route available before any uniqueness boundary is read.

Apply the limit-uniqueness export of Definition 1612.30 and the source-comparison exhaustion theorem, Definition 1612.32. Together they identify U and the transport and replay rows H, R, T as the complete handoff boundary.

The provenance and local naming rows P, N attach the boundary to Pkg and NameCert data. Because the carrier contains no quotient-completion, selected-representative, host-equality, choice, or ambient-completeness coordinate, a $\mathsf{CauchyCompletionReflector}^{\uparrow}$ or $\mathsf{CompletionUniversalProperty}^{\uparrow}$ consumer receives only the displayed finite route. □

Theorem 1612.40 (SeparatedCompletion dense replay scope)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, every completion-facing read exposes the source metric row* M*, dense embedding row* D*, and completion handoff row* C *before the separated classifier* Z*, uniqueness boundary* U*, transport row* H*, replay row* R*, route ledger* T*, provenance row* P*, or local naming row* N *is consumed. Thus dense replay is a finite* M, D, C *prefix over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *data, not a selected representative or quotient-completion read.*

Proof. Start with the carrier of Definition 1612.1. The rows M, D, C are the only source metric, dense embedding, and completion handoff coordinates.

Apply dense factorization, Definition 1612.19. It places every completion-facing consumer read on the route $M, D \mapsto C$ before the separated classifier and uniqueness boundary are available.

Then use the choice-free source route, Definition 1612.34. The structural rows H, R, T, P, N preserve the same ordered route and do not introduce a choice coordinate, quotient completion, selected representative, or host equality of completed points. □

Theorem 1612.41 (SeparatedCompletion separated classifier scope)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, zero-distance classification through* Z *is the only separated equality route after the dense completion handoff* M, D, C *has been displayed. Any completion-facing equality read that bypasses* Z *leaves the* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *surface and cannot be recovered from* U, H, R, T, P, N *alone.*

Proof. Read the dense prefix first. By Definition 1612.40, a completion-facing equality read has already exposed M, D, C before it reaches any separated comparison row.

Now apply zero-distance factorization, Definition 1612.29. The classifier row Z reads equality as displayed zero-distance evidence over the $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, rather than as host equality or quotient completion equality.

Finally apply source-comparison exhaustion, Definition 1612.32. The uniqueness, transport, replay, provenance, and naming rows preserve a comparison once Z is present, but none of them supplies an alternate separated equality route. □

Theorem 1612.42 (SeparatedCompletion obligation closure package)**.** *The accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier*

$$S = (M, D, C, Z, U, H, R, T, P, N)$$

supports an obligationClosure package whose finite boundary is exactly dense factorization, zero-distance factorization, source-comparison exhaustion, the choice-free source route, the filtered limit boundary, and the completion-reflector handoff. The package closes the paper-side obligation surface over the displayed rows and refuses quotient completions, selected representatives, host equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, and hidden complete metric spaces.*

Proof. Build the source prefix from dense factorization, Definition 1612.19, and dense replay scope, Definition 1612.40. These results force every completion-facing read through M, D, C.

Build the classifier boundary from zero-distance factorization, Definition 1612.29, and separated classifier scope, Definition 1612.41. They make Z the only separated equality route after the dense completion handoff.

Add the comparison and source routes from Definition 1612.32 and Definition 1612.34. These theorems keep all comparison data in displayed BHist, hsame, Cont, Pkg, and NameCert rows.

Finally attach the filtered limit boundary, Definition 1612.38, and the completion-reflector handoff, Definition 1612.39. The resulting package contains no coordinate for quotient completions, selected representatives, host equality, countable choice, ambient Real completeness, or hidden complete metric spaces, so the obligation surface is closed at the paper level. □

Theorem 1612.43 (SeparatedCompletion choice-free endpoint package)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, any endpoint comparison exported to* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{HausdorffCompletion}^{\uparrow}$*,* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumers factors through the displayed rows*

$$M, D, C, Z, U, H, R, T, P, N.$$

The exported comparison is the finite endpoint package obtained from dense-source admission, filtered limit boundary, choice-free source transport, completion uniqueness, and Cauchy-reflector handoff. It exports no quotient completion, selected representative, host equality of completed points, countable choice, hidden complete metric space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the dense route. Dense factorization, Definition 1612.19, and dense replay scope, Definition 1612.40, force every endpoint-facing read through M, D, C before the separated classifier or uniqueness boundary is available.

Transport the limit boundary without a selector. The filtered limit boundary of Definition 1612.38 and the choice-free source route of Definition 1612.34 keep the read inside displayed BHist, hsame, Cont, Pkg, and NameCert rows. Thus the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff remains a finite route through C and cannot become a selected limit.

Attach the completion uniqueness package. The obligation closure package of Definition 1612.42 supplies the local uniqueness boundary over Z, U, H, R, T, P, N, while the Hausdorff uniqueness route of Definition 1612.36 keeps the comparison separated rather than quotient-based.

Finally use the Cauchy-reflector handoff, Definition 1612.39, together with the Cauchy reflector obligation of Definition 1612.22. These results route $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, and $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ consumers through the same displayed endpoint rows. Since the carrier has no quotient, representative-selection, host-equality, choice, hidden-complete-space, or ambient-completeness coordinate, the endpoint package exports exactly the rows stated above. □

Theorem 1612.44 (SeparatedCompletion carrier admission root obligation)**.** *Every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier exposes only the displayed* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *row is read. The admission surface contains no additional completion point, quotient class, selected representative, host equality of completed points, countable-choice schedule, or hidden complete metric space.*

Proof. Project the carrier of Definition 1612.1. The source, dense-embedding, completion handoff, separated classifier, and uniqueness rows are exactly M, D, C, Z, U.

Read these rows through the existing source route. The Cauchy-equivalence source theorem, Definition 1612.21, and the dense factorization theorem, Definition 1612.19, place the admission boundary over $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ data.

Then attach the uniqueness boundary from Definition 1612.20. It adds only the displayed $\mathsf{CompletionUniversalProperty}^{\uparrow}$ and $\mathsf{HausdorffCompletion}^{\uparrow}$ rows over the same finite route. Since H, R, T, P, N merely transport, replay, preserve, source, and name the displayed rows, the excluded completion objects have no carrier coordinate. □

Theorem 1612.45 (SeparatedCompletion classifier stability root obligation)**.** *For any accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carriers over componentwise* hsame*-transported* $M, D, C, Z, U, H, R, T, P, N$ *rows, the separated classifier exposed by* Z *is preserved by the displayed transport and replay rows. This stability cannot introduce quotient-completion equality, selected representatives, host equality of completed points, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the transported carrier rows. Componentwise hsame transport acts on the displayed source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, replay, provenance, and local naming rows named by the carrier.

Apply separated classifier scope, Definition 1612.41. After the dense completion handoff, Z is the only separated equality route available to a completion-facing read.

Finally apply source-comparison exhaustion, Definition 1612.32. It keeps transported comparisons inside $M, D, C, Z, U, H, R, T, P, N$. Because the transported packet has no quotient-completion or host-equality coordinate, classifier stability cannot create one. □

Theorem 1612.46 (SeparatedCompletion Real seal root obligation)**.** *Any Real-facing seal read exported by an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier factors through displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *rows before the separated completion handoff is consumed. The seal read is therefore a finite BEDC route through* $M, D, C, Z, U, H, R, T, P, N$*, not an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle or a chosen completion representative.*

Proof. Read the dense prefix first. By Definition 1612.40, every completion-facing read exposes M, D, C before any separated classifier, uniqueness boundary, or local naming row is available.

The opening carrier route identifies this prefix with the L10 spine through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the $\mathsf{MetricCompletion}^{\uparrow}$ handoff. The Real-facing seal is therefore downstream from finite stream windows and regular rational readback.

Now apply the choice-free source route, Definition 1612.34, and the completion-reflector handoff, Definition 1612.39. They keep the separated completion handoff in displayed BHist, hsame, Cont, Pkg, and NameCert rows and exclude ambient Real completeness, selected representatives, quotient completions, host equality, and countable choice. □

Theorem 1612.47 (SeparatedCompletion root obligation package)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier, the carrier-admission, classifier-stability, and Real-seal root obligations of Definition 1612.44, Definition 1612.45, and Definition 1612.46 jointly exhaust the finite* $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$ *surface over* $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *rows. Thus every downstream completion consumer receives only displayed source admission, separated classifier stability, Real-seal readback, componentwise transport, replay, provenance, and local naming.*

Proof. The carrier-admission root obligation identifies the finite row inventory available to the accepted carrier. It confines every completion-facing read to the displayed source, dense embedding, completion handoff, separated classifier, uniqueness, transport, replay, provenance, and local naming rows.

The classifier-stability root obligation then fixes the separated equality surface. Transport and replay preserve the classifier row inside that same finite inventory, so no extra quotient representative or host equality coordinate appears after the carrier is admitted.

The Real-seal root obligation closes the remaining completion-to-Real route through the StreamName, RegSeqRat, DyadicRatCore, and MetricCompletion rows. Together the three obligations cover the root surface exposed to CauchyCompletionReflector, CompletionEmbedding, MetricCompletion, RegSeqRat, Real, and StreamName consumers, and none of them introduces quotient representatives, host equality, countable choice, or an ambient completeness coordinate. □

Theorem 1612.48 (SeparatedCompletion separated witness triad)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every separated-completion equality consumer must expose the dense source row* D*, the zero-distance classifier row* Z*, and the uniqueness boundary row* U *in that order before any exported completion seal is read. The exposed triad is the root obligation witness over the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route; it is not a quotient completion, selected representative, host equality, countable-choice schedule, or completeness claim for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier. The rows D, Z, U are already part of the finite inventory listed in Definition 1612.1, with D recording dense source admission, Z recording separated zero-distance classification, and U recording the uniqueness boundary over the same route.

Use the NameCert obligations of Definition 1612.2. Dense-source compatibility forces the consumer to read D before the completion handoff can be inspected. Separated-classifier exactness then places Z after the handoff, and uniqueness-boundary exactness places U over the same M, D, C, Z route.

The root obligation package, Definition 1612.47, binds these rows to the carrier-admission, classifier-stability, and Real-seal obligations. Thus an exported completion seal can be read only after the dense source row, zero-distance classifier, and uniqueness boundary have been displayed in the stated order.

The carrier has no coordinate for quotient completions, selected representatives, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The witness triad is therefore induction-ready because later consumers can recurse over the displayed rows D, Z, U and their structural transport data, without selecting representatives or appealing to an external completion equality. □

Theorem 1612.49 (SeparatedCompletion filterbase admission)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier*

$$S = (M, D, C, Z, U, H, R, T, P, N),$$

any filterbase-facing admission read factors through the same finite source route as Definition 1612.47. The $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *rows receive the source metric M, dense embedding D, completion handoff C, separated classifier Z,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *windows, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows H, R, T, P, N before a filterbase consumer may inspect completion data. Thus filterbase admission is a* BHist *and* $\mathsf{NameCert}$ *route, not a quotient completion, selected representative, host equality, countable-choice selector, or ambient complete-space coordinate.*

Proof. Project the accepted carrier. The rows M, D, C admit the source metric, dense embedding, and completion handoff before any filterbase-facing consumer can read completion data.

The separated classifier row Z and uniqueness boundary U keep the filterbase comparison inside the same displayed separated route. Apply Definition 1612.45 to see that transport and replay cannot add a host equality coordinate.

Finally read the completion-facing spine. The $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are delivered through H, R, T, P, N, so the filterbase admission has no access to quotient completions, selected representatives, countable choice, or an ambient complete-space object. □

Theorem 1612.50 (SeparatedCompletion Cauchy-equivalence routing)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier, each Cauchy-equivalence read sent to* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the separated classifier and the Cauchy source rows before the terminal Real seal is exposed. In symbols, the public Cauchy-equivalence route is*

$$M, D, C, Z, U \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, R, T, P, N.$$

The route exports only displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows and does not export quotient stream equality, host equality of completed points, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 1612.21. It places every Cauchy-equivalence read behind the displayed Cauchy source and separated-completion rows, so the comparison is not a free-standing quotient relation.

Next apply Definition 1612.29 and Definition 1612.30. These theorems route zero-distance and uniqueness reads through Z and U before the regular-rational and stream windows can be consumed.

The Real-facing boundary is then supplied by Definition 1612.46. The structural rows preserve and replay the same path through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$, so no quotient stream equality, host equality, selected limit, choice principle, or ambient completeness coordinate is available. □

Theorem 1612.51 (SeparatedCompletion completion-reflector nonescape)**.** *Every* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier remains inside the finite route*

$$M, D, C, Z, U, H, R, T, P, N$$

together with the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff rows. The reflector may read the source metric, dense embedding, separated classifier, uniqueness boundary, transport, replay, provenance, and local naming, but it may not import quotient completions, selected representatives, host equality, countable choice, hidden complete metric spaces, or an ambient real-completeness theorem.*

Proof. Read the reflector handoff through Definition 1612.39. It already places the completion-reflector consumer downstream of the source metric, dense embedding, completion handoff, separated classifier, and Real-facing L10 spine.

The root package of Definition 1612.47 exhausts the carrier-admission, classifier-stability, and Real-seal obligations. Hence the reflector consumer can inspect only the rows listed there and the transport, replay, provenance, and naming rows H, R, T, P, N.

Nonescape follows from absence of coordinates. Quotient completions, selected representatives, host equality, countable choice, hidden complete metric spaces, and ambient real completeness are not fields of the carrier and cannot be produced by hsame transport, Cont replay, Pkg provenance, or local $\mathsf{NameCert}$ naming. □

Theorem 1612.52 (SeparatedCompletion Cauchy-reflector triangle)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier whose source route receives* $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Every endpoint-uniqueness read sent toward a Real-facing completion seal factors through the separated classifier row* Z *before it can be consumed by the reflector or universal-property rows. The triangle*

$$\mathsf{CauchyCompletionReflector}^{\uparrow} \longmapsto \mathsf{SeparatedCompletion}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow}$$

therefore commutes at the displayed $\mathsf{NameCert}$ *surface through* $M, D, C, Z, U, H, R, T, P, N$, *and exports no quotient completion, selected representative, host equality of completed points, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The dense source rows M, D, C are read first, so the reflector-facing route starts from the same finite metric, dense embedding, and completion handoff as the rest of the chapter.

Apply Definition 1612.19 and Definition 1612.20. They place endpoint uniqueness downstream of the dense factorization and force the uniqueness boundary to read the separated classifier row Z rather than a host equality coordinate.

Now apply Definition 1612.39. The CauchyCompletionReflector and CompletionUniversalProperty consumers receive the same route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, route-preservation, provenance, and local naming rows H, R, T, P, N preserve this factorization. Because the carrier contains no quotient-completion, selected-representative, host-equality, choice, or ambient-completeness coordinate, the only available triangle is the displayed separated-classifier route. □

Theorem 1612.53 (SeparatedCompletion root-consumer unblock package)**.** *Every downstream Real or Completion consumer of an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier must factor through the carrier-admission root obligation, classifier-stability root obligation, Real-seal root obligation, root obligation package, and Cauchy-reflector triangle before reading any completion-facing output. Equivalently, any consumer route over* $S = (M, D, C, Z, U, H, R, T, P, N)$ *must display the finite root surface*

$$M, D, C, Z, U, H, R, T, P, N$$

together with the carrier-admission, classifier-stability, Real-seal, root-package, and reflector-triangle witnesses. The package exports no quotient completion, selected representative, host equality of completed points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier into the rows $M, D, C, Z, U, H, R, T, P, N$. The source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, transport, route preservation, replay, provenance, and local naming rows are therefore the only displayed input surface available to downstream consumers.

Apply Definition 1612.47. It combines the carrier-admission root obligation, classifier-stability root obligation, and Real-seal root obligation, so the source, classifier, and seal sides of the accepted packet are exhausted before a completion-facing output can be read.

Next apply Definition 1612.52. The reflector and universal-property routes commute only through the separated classifier row and the same finite L10 source spine, so endpoint-uniqueness reads cannot bypass the root package.

The remaining structural rows H, R, T, P, N preserve, replay, source, and name that route. Since no quotient completion, selected representative, host equality, countable-choice schedule, or ambient Real-completeness coordinate appears among the carrier rows, the displayed package is the required unblock surface for all downstream Real and Completion consumers. □

Theorem 1612.54 (SeparatedCompletion dense limit route)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every accepted limit-facing read must pass through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the uniqueness row* U *is public. The route exposes dense source data, zero-distance classification, regular-Cauchy separated completion, and the Real-facing L10 spine, and it exports no quotient completion, selected limit representative, host equality of completed points, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the dense source prefix first. Dense factorization, Definition 1612.19, and dense replay scope, Definition 1612.40, force every completion-facing read through M, D, C before any separated classifier or uniqueness row is available.

Read the classifier and limit boundary next. Zero-distance factorization, Definition 1612.29, makes Z the separated equality route, and limit-uniqueness export, Definition 1612.30, makes U the public uniqueness row over M, D, C, Z.

Now attach source-comparison exhaustion, Definition 1612.32, and nonescape, Definition 1612.33. These results keep the source route inside the displayed MetricCompletion, CompletionEmbedding, CompletionUniversalProperty, SeparatedMetric, HausdorffCompletion, RegularCauchyCompletion-Separated, DyadicRatCore, StreamName, RegSeqRat, and Real rows. Since none of the excluded objects is a carrier coordinate, the dense limit route reaches uniqueness only through the displayed packet. □

Theorem 1612.55 (SeparatedCompletion choice-free limit transport)**.** *For accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carriers whose source, dense embedding, completion handoff, separated classifier, uniqueness boundary, transport, replay, route, provenance, and local naming rows are matched componentwise by* hsame *and replayed by* Cont*, the transported packet preserves the same dense source, zero-distance classifier, and uniqueness boundary. This preservation uses no countable choice, quotient completion, selected representative, host equality of completed points, hidden complete metric space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the two carriers and read their transported rows componentwise. The hsame rows act only on $M, D, C, Z, U, H, R, T, P, N$, and the Cont rows replay the same displayed path.

Apply dense factorization, Definition 1612.19, to both packets. The dense source remains the M, D, C route after transport, because no additional source coordinate is present.

Apply zero-distance factorization, Definition 1612.29, and limit-uniqueness export, Definition 1612.30. They identify Z as the classifier and U as the uniqueness boundary before any downstream completion consumer can read the transported packet.

Finally use source-comparison exhaustion, Definition 1612.32, together with nonescape, Definition 1612.33. The structural rows preserve the dense source, zero-distance classifier, and uniqueness boundary, while the excluded choice, quotient, representative, host-equality, hidden-space, and ambient-completeness objects remain absent from the carrier. □

Theorem 1612.56 (SeparatedCompletion completion uniqueness package)**.** *Every downstream* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{HausdorffCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier receives uniqueness only as the displayed row* U *over* M, D, C, Z *together with the structural rows* H, R, T, P, N*. The package may use the dense limit route of Definition 1612.54 and the choice-free transport route of Definition 1612.55, but it cannot export quotient completion equality, a selected representative, host equality of completed points, countable choice, a hidden complete metric space, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the dense route. By Definition 1612.54, a MetricCompletion or HausdorffCompletion consumer sees M, D, C before the separated classifier Z and uniqueness row U are public.

Read uniqueness through the local row U. The limit-uniqueness export theorem, Definition 1612.30, identifies that row as the whole uniqueness boundary over M, D, C, Z.

Transport and replay are then handled by Definition 1612.55. The rows H, R, T, P, N preserve, replay, source, and name the same boundary for downstream consumers.

Apply source-comparison exhaustion, Definition 1612.32, and nonescape, Definition 1612.33, to close the package. Since the carrier contains no quotient completion equality, selected representative, host equality, choice schedule, hidden complete metric space, or ambient completeness coordinate, the displayed U row over $M, D, C, Z, H, R, T, P, N$ is the only uniqueness package available. □

Theorem 1612.57 (SeparatedCompletion density separated route)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every dense limit read exported before uniqueness must factor through the displayed* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row* D*, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier* Z*, the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *route carried by* C*, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows in the completion handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before* U *may be consumed. Thus density and separation meet inside the finite route*

$$M, D, C, Z \longmapsto U, H, R, T, P, N,$$

not in a quotient completion, selected representative, host equality of completed points, countable-choice schedule, hidden complete metric space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier first. The source metric row M and dense embedding row D are the only displayed entrance into a dense limit read.

Read the completion handoff C next. Its regular-Cauchy separated route carries the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ seal rows before a separated comparison can be public.

Apply zero-distance factorization, Definition 1612.29. The separated classifier Z is the comparison boundary after C, so host equality of completed points and quotient equality are unavailable.

Only then may the uniqueness row U be consumed. The structural rows H, R, T, P, N transport, replay, source, and name the same dependency order, and the excluded completion objects have no coordinate in the carrier. □

Theorem 1612.58 (SeparatedCompletion filterbase reflector exhaustion)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every filterbase-facing reflector read exported by S factors through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *admission row, the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *dense row, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *reflector boundary, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier, the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated readback, the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *route, the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *L10 spine, and then through the structural* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. The exported read contains no quotient completion equality, selected representative, countable-choice schedule, host equality of completed points, hidden complete metric space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier. The only reflector-facing coordinates are the source metric row M, dense embedding row D, completion handoff row C, separated classifier row Z, uniqueness boundary U, and the structural rows H, R, T, P, N.

Read the filterbase entrance through the existing admission route of Definition 1612.49. That route already places filterbase data inside the displayed $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompletionEmbedding}^{\uparrow}$ prefix before a reflector consumer may inspect completion data.

Attach the reflector boundary using Definition 1612.39 and Definition 1612.52. The $\mathsf{CompletionUniversalProperty}$ $\mathsf{HausdorffCompletion}^{\uparrow}$, and $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ rows are consumed only after M, D, C, Z, U have been displayed.

Finally read the L10 spine carried by C: $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ supply finite tolerance, window, readback, and seal data. Since the carrier has no coordinate for quotient completion equality, selected representatives, choice schedules, host equality, hidden complete spaces, or ambient completeness, the reflector read is exhausted by the named rows. □

Theorem 1612.59 (SeparatedCompletion Cauchy-sequence equivalence lock)**.** *Let $S = (M, D, C, Z, U, H, R, T, P,$ be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every Cauchy-equivalence read exported by S is locked to the displayed route through* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. No exported equivalence may be replaced by quotient completion equality, a selected representative, countable choice, host equality of completed points, a hidden complete metric space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the Cauchy-equivalence source theorem Definition 1612.21. It places every comparison inside the displayed source, dense embedding, and completion handoff rows M, D, C before the separated classifier Z can be read.

Use zero-distance factorization, Definition 1612.29, to make Z the separated comparison boundary. The equivalence read is therefore a displayed zero-distance route through the $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows carried by M, D, C, Z.

Read the uniqueness and Hausdorff boundary through Definition 1612.30 and Definition 1612.36. These routes attach the CompletionUniversalProperty$^{\uparrow}$, HausdorffCompletion$^{\uparrow}$, and RegularCauchyCompletionSeparate rows without adding any equality object outside the carrier.

The structural rows H, R, T, P, N preserve, replay, source, and name the same route. Since quotient completions, selected representatives, choice schedules, host equality, hidden complete spaces, and ambient completeness are absent from the carrier, every exported Cauchy-equivalence read is locked to the displayed finite route. □

Theorem 1612.60 (SeparatedCompletion Hausdorff completion readback)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* SeparatedCompletion$^{\uparrow}$ *carrier. Every* HausdorffCompletion$^{\uparrow}$ *readback consumed by a downstream completion chapter factors through the same displayed* MetricCompletion$^{\uparrow}$*,* CompletionEmbedding$^{\uparrow}$*,* CompletionUniversalProperty$^{\uparrow}$*,* SeparatedMetric$^{\uparrow}$*,* RegularCauchyCompletionSeparated$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *route. The readback is a finite separated-completion boundary over S, not a quotient completion equality, selected representative, countable-choice schedule, host equality of completed points, hidden complete metric space, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier and read M, D, C as the dense completion prefix. By Definition 1612.57, the downstream completion consumer must see the dense embedding, separated classifier, regular-Cauchy separated route, stream and rational readback rows, and terminal Real seal before uniqueness may be consumed.

Now read the Hausdorff boundary through Definition 1612.36 and the reflector handoff through Definition 1612.39. These theorems place HausdorffCompletion$^{\uparrow}$ readback after CompletionUniversalProperty$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, and RegularCauchyCompletionSeparated$^{\uparrow}$ have been displayed over the same source route.

Finally apply source-comparison exhaustion, Definition 1612.32. The rows H, R, T, P, N transport, replay, preserve, source, and name the displayed route, while quotient completion equality, selected representatives, choice schedules, host equality, hidden complete spaces, and ambient real completeness have no coordinate in the carrier. □

Theorem 1612.61 (SeparatedCompletion source triad obligation)**.** *Let $S = (M, D, C, Z, U, H, R, T, P, N)$ be an accepted* SeparatedCompletion$^{\uparrow}$ *carrier. Every accepted packet reads the source metric row M, dense embedding row D, and completion handoff row C before any separated comparison through Z is public. The triad is preserved through* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg *provenance, and the local* NameCert *row, and no separated-comparison consumer can start from Z, U, H, R, T, P, N alone.*

Proof. Project the carrier of Definition 1612.1. The source metric, dense embedding, and completion handoff rows are exactly the coordinates M, D, C, and the separated classifier is a later coordinate Z over the same packet.

Apply dense factorization, Definition 1612.19. It places every completion-facing read on the route $M, D \mapsto C$ before separated comparison or uniqueness data is available.

Now apply zero-distance factorization, Definition 1612.29. The row Z reads equality as displayed zero-distance evidence over the already exposed dense completion handoff, not as an independent comparison source. The structural rows H, R, T, P, N only preserve, replay, source, and name this order. □

Theorem 1612.62 (SeparatedCompletion quotient-exhaustion refusal)**.** *For every accepted* SeparatedCompletion$^{\uparrow}$ *carrier $S = (M, D, C, Z, U, H, R, T, P, N)$, quotient completions, selected representatives, host*

equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, and hidden complete spaces are absent from every public route exported by* S*. Any public completion-facing, equality-facing, uniqueness-facing, or reflector-facing read is exhausted by* $M, D, C, Z, U, H, R, T, P, N$ *through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

Proof. Start with universal route exhaustion, Definition 1612.27. It identifies the public route with the displayed source, dense embedding, completion handoff, separated classifier, uniqueness boundary, transport, replay, provenance, and local naming rows.

Attach the dense source and comparison constraints. Dense factorization, Definition 1612.19, forces M, D, C to be read first, and zero-distance factorization, Definition 1612.29, forces equality reads through Z rather than through host equality or quotient equality.

Finally apply the limit-uniqueness export, Definition 1612.30. It makes U the whole public uniqueness boundary over M, D, C, Z. Since the carrier has no coordinate for quotient completions, selected representatives, choice schedules, ambient Real completeness, or hidden complete spaces, those objects are absent from every public route. □

Theorem 1612.63 (SeparatedCompletion scoped route)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier, the scoped route is exactly*

$$M, D, C, Z, U, H, R, T, P, N$$

read through BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The route admits the source metric, dense embedding, completion handoff, separated classifier, uniqueness boundary, transport, replay, route ledger, provenance, and local naming rows, and it admits no additional completion, equality, choice, Real-completeness, or hidden-space coordinate.*

Proof. Project the accepted carrier using Definition 1612.1. This gives the displayed row inventory $M, D, C, Z, U, H, R, T, P, N$ and no further public coordinate.

Use Definition 1612.61 to fix the source prefix. Every separated-completion read starts with M, D, C before it can inspect Z or U.

Use quotient-exhaustion refusal, Definition 1612.62, for the negative side. It excludes quotient completions, selected representatives, host equality, countable choice, ambient Real completeness, and hidden complete spaces from the route. The remaining rows are precisely the BHist, hsame, Cont, Pkg, and NameCert surface already displayed by the carrier. □

Definition 1612.64 (SeparatedCompletion scoped kernel route)**.** *For an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, the scoped kernel route is the finite BEDC read*

$$(\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}) \vdash M, D, C, Z, U, H, R, T, P, N.$$

It binds dense-source compatibility by M, D, C*, separated classifier exactness by* Z, U*, transport and replay by* H, R, T*, and provenance and local naming by* P, N*. The route has no coordinate for quotient completions, selected representatives, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric spaces.*

Theorem 1612.65 (SeparatedCompletion dense source scope)**.** *Every dense-source consumer of an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier must read* M, D, C *through the scoped kernel route before reading* Z, U, H, R, T, P, N*. Thus dense-source compatibility is a statement about the displayed source metric row, dense embedding row, and completion handoff row inside* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, not about an external completed space.*

Proof. Apply the scoped route theorem, Definition 1612.63. It identifies the public route with the rows $M, D, C, Z, U, H, R, T, P, N$ and no additional coordinate.

Now use the dense-limit route, Definition 1612.54. It places dense completion reads on the prefix M, D, C before separated comparison or uniqueness rows are consumed.

Finally apply quotient-exhaustion refusal, Definition 1612.62. Since quotient completions, selected representatives, host equality, choice, ambient Real completeness, and hidden complete spaces are absent, the dense-source scope is exactly the displayed kernel prefix. □

Theorem 1612.66 (SeparatedCompletion classifier nonescape scope)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier, the separated classifier and uniqueness boundary are exact only through* Z, U *over the dense-source prefix* M, D, C*. Any completion-facing, equality-facing, uniqueness-facing, or reflector-facing consumer reads the classifier route through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, and it cannot escape to quotient equality, selected representatives, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric spaces.*

Proof. Start from Definition 1612.64. It names the carrier rows available to the scoped kernel route and separates the dense-source prefix from the classifier and uniqueness rows.

Use Definition 1612.63 to identify the public consumer route. The theorem makes $M, D, C, Z, U, H, R, T, P, N$ the whole visible surface.

The exactness side is supplied by Definition 1612.54: the separated classifier is read only after the dense completion prefix has been exposed. The negative side is supplied by Definition 1612.62, which excludes quotient, representative, host equality, choice, Real-completeness, and hidden-space coordinates from every public route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1612.1, Definition 1612.2, Definition 1612.63, Definition 1612.62, the downstream routes in this file, and the source-density scope closure of Definition 1612.67, Definition 1612.68, and Definition 1612.69 bind the mature paper-side separated-completion semantic NameCert export to source density, Cauchy completion, separated metric comparison, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeparatedCompletion}^{\uparrow}}$, auditClean)

Lean target: `BEDC.Derived.SeparatedCompletionUp.SeparatedCompletionCarrier\_semantic\_name\_certificate`

Bridge status: paperBridge

Not claimed: This chapter does not claim quotient completions, selected representatives, host equality of completed points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, hidden complete metric spaces, or bridge checking.

Upgrade path: The remaining upgrade is a Lean-side standard bridge check for the same separated-completion semantic NameCert export.

Constructive story: A $\mathsf{SeparatedCompletion}^{\uparrow}$ packet is generated from a source metric row, a dense CompletionEmbedding row, a completion handoff through the DyadicRatCore-StreamName-RegSeqRat-Real L10 spine, a SeparatedMetric classifier, a CompletionUniversalProperty-HausdorffCompletion-RegularCauchyCompletionSeparated uniqueness boundary, componentwise hsame transport, Cont replay, route-preservation ledger, Pkg provenance, and a local NameCert row.

Theorem 1612.67 (SeparatedCompletion source-density scope)**.** *Let* $S = (M, D, C, Z, U, H, R, T, P, N)$ *be an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier. Every source-density read consumed by* S *is scoped by the source metric row* M*, dense embedding row* D*, completion handoff row* C*, separated classifier row* Z*, uniqueness row* U*, componentwise transport* H*, displayed replay and route ledger* R, T*,* Pkg *provenance* P*, and local* NameCert *row* N *through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The read may cite Definition 1612.1 and Definition 1612.2, but it exports no quotient completion, selected representative, host equality of completed points, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric space.*

Proof. Project the accepted carrier. The source metric row, dense embedding row, completion handoff row, separated classifier row, uniqueness row, transport, replay, route ledger, provenance, and local name are exactly $M, D, C, Z, U, H, R, T, P, N$.

Apply the NameCert obligations of Definition 1612.2. They force dense-source compatibility through M, D, C and keep every completion-facing read inside the same local packet.

Finally use the scoped route and quotient-exhaustion rows, Definition 1612.63 and Definition 1612.62. These identify the public route as the displayed kernel surface through BHist, hsame, Cont, Pkg, and NameCert, with the excluded completion objects absent from every coordinate. □

Theorem 1612.68 (SeparatedCompletion classifier scope)**.** *For every accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, Z, U, H, R, T, P, N)$*, a separated classifier read is scoped only after the source metric row* M*, dense embedding row* D*, and completion handoff row* C *have been displayed. The classifier row* Z*, uniqueness row* U*, transport row* H*, replay and route rows* R, T*, provenance* P*, and local* NameCert *row* N *bind the comparison to* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*, citing the carrier and obligation surfaces of Definition 1612.1 and Definition 1612.2.*

Proof. Read the source-density prefix first. By Definition 1612.67, the rows M, D, C are the only source, dense, and completion-handoff entrance for the accepted carrier.

Now inspect the classifier side. The separated classifier exactness clause of Definition 1612.2 places Z after C, and the uniqueness-boundary exactness clause reads U over the same M, D, C, Z route.

The structural rows H, R, T, P, N preserve, replay, source, and name that route. Hence the classifier scope is a local BEDC kernel route through BHist, hsame, Cont, Pkg, and NameCert, not a host equality or quotient-completion comparison. □

Theorem 1612.69 (SeparatedCompletion nonescape scope)**.** *Every public separated-completion read of an accepted* $\mathsf{SeparatedCompletion}^{\uparrow}$ *carrier is exhausted by the source metric row, dense embedding row, completion handoff row, separated classifier row, uniqueness row, transport, replay, route ledger, provenance, and local NameCert rows*

$$M, D, C, Z, U, H, R, T, P, N$$

through the kernel rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. Together with Definition 1612.67 and Definition 1612.68, this excludes quotient completions, selected representatives, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden complete metric spaces from the scoped public surface.*

Proof. Begin with the carrier and obligations. The carrier definition fixes $M, D, C, Z, U, H, R, T, P, N$ as the whole packet, and Definition 1612.2 assigns source-density, separated-classifier, uniqueness, transport, replay, provenance, and naming roles to those rows.

Apply the source-density and classifier scope theorems. They place source reads through M, D, C and separated comparisons through Z, U after that prefix, with H, R, T, P, N attached as structural BEDC rows.

Use quotient-exhaustion refusal, Definition 1612.62, for the negative side. Since none of the excluded objects is a carrier coordinate, every public read remains inside BHist, hsame, Cont, Pkg, and NameCert over the displayed row inventory. □

Theorem 1612.70 (SeparatedCompletion scope-closure route)**.** *Every public* $\mathsf{SeparatedCompletion}^{\uparrow}$ *read of an accepted carrier factors through the ordered rows*

$$M, D, C, Z, U, H, R, T, P, N.$$

The prefix M, D, C supplies the source metric, dense embedding, and completion handoff scope of Definition 1612.67; the rows Z, U supply the separated classifier and uniqueness scope of Definition 1612.68; and the rows H, R, T, P, N preserve the same public route through BHist, hsame, Cont, Pkg, *and* NameCert *as in Definition 1612.69. The route exports no quotient completion, selected representative, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric space.*

Proof. Start with the source-density theorem. It identifies M, D, C as the only entrance for source metric, dense embedding, and completion handoff data.

Next apply the classifier theorem. The separated classifier row Z and uniqueness row U are read only after that source-density prefix, so classifier comparison cannot detach from M, D, C.

Finally apply the nonescape theorem. It attaches H, R, T, P, N to the same ordered carrier route and excludes quotient completions, selected representatives, host equality, countable choice, ambient real completeness, and hidden complete metric spaces from the public surface. □

Chapter 1613

A Concrete Naming Certificate for $\mathsf{DyadicNormalForm}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicNormalForm}^{\uparrow}$ packet names the finite canonical presentation of a dyadic rational before the open Real and Completion route passes through regular rational streams. It sits downstream of the signed rows of chapter 31, the rational rows of chapter 37, and the dyadic core boundary of chapter 291; it supplies the displayed normal-form surface read by chapter 126, chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records canonical finite dyadic representation data, not quotient rational equality, completed real equality, order, completeness, or a standard bridge.

Definition 1613.1 (Dyadic normal-form carrier)**.** *A* $\mathsf{DyadicNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, M, E, Z, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the sign row, read through the signed* $\mathsf{Int}^{\uparrow}$ *surface and recording positive, negative, or zero orientation for the displayed dyadic.*
2. *M is the signed mantissa row, an* $\mathsf{Int}^{\uparrow}$*-read finite integer whose absolute part is visible before scale inspection.*
3. *E is the unary exponent row, carrying the nonnegative dyadic scale attached to the denominator 2^E.*
4. *Z is the zero classifier row, recording that the zero packet is represented by the distinguished row $M = 0$ with terminal scale $E = 0$.*
5. *R is the scale-minimality row, recording that a nonzero mantissa is odd at the displayed scale and hence cannot be divided by 2 while lowering E.*
6. *L is the dyadic readback ledger, exposing the finite rational row $M/2^E$ through* $\mathsf{Rat}^{\uparrow}$ *and the accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *boundary.*
7. *H records componentwise* hsame *transport for the sign, mantissa, exponent, zero classifier, scale-minimality row, readback ledger, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a dyadic input through sign extraction, mantissa readback, unary scale inspection, zero handling, scale-minimality verification, and dyadic readback.*

9. *P is the* Pkg *provenance over* $\mathsf{Int}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, M, E, Z, R, L, H, C, P, N$. It has no coordinate for quotient rational equality, host real equality, order, completion, a choice-selected representative, or a standard bridge.

Theorem 1613.2 (Dyadic normal-form NameCert obligations)**.** *For every accepted* $\mathsf{DyadicNormalForm}^{\uparrow}$ *carrier $D = (S, M, E, Z, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicNormalForm}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the sign, signed mantissa, unary exponent, zero classifier, scale-minimality row, dyadic readback ledger, transport, continuation, provenance, and local naming row;*
2. *source compatibility, requiring M and E to be read from an accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *packet through the displayed* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *surfaces;*
3. *zero exactness, requiring Z to identify the unique zero normal row $M = 0, E = 0$ before any downstream readback is exposed;*
4. *scale-minimality, requiring every nonzero accepted row to expose an odd mantissa at the displayed scale, so that no lower unary exponent represents the same dyadic row inside the packet;*
5. *dyadic readback stability, requiring L, H, C to preserve the rational readback $M/2^E$ under componentwise* hsame *transport and displayed* Cont *replay;*
6. *downstream handoff, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers to inspect only the finite normal-form row and its dyadic readback ledger;*
7. *ledger refusal, excluding quotient rational equality, host real equality, order, completion, choice-selected representatives, and standard-bridge data.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1613.1. The projection exposes exactly the sign row, mantissa row, unary exponent row, zero classifier, scale-minimality row, dyadic readback ledger, and structural rows H, C, P, N.

Source compatibility follows from the attachment to chapter 291. The mantissa is read through $\mathsf{Int}^{\uparrow}$, the rational readback is read through $\mathsf{Rat}^{\uparrow}$, and both rows are consumed only as finite dyadic-core data.

Zero exactness and scale-minimality are the two classifier rows. The zero case is forced by Z, while the nonzero case is forced by R, which records that no displayed factor of 2 remains in the mantissa at the chosen scale.

Replay C and then transport by H. The route preserves the same sign, mantissa, exponent, zero classifier, minimality row, and readback ledger before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer can inspect the packet.

The exclusions are immediate from the carrier inventory. Since quotient rational equality, host real equality, order, completion, choice-selected representatives, and bridge rows are absent, the local NameCert surface cannot export them. □

Theorem 1613.3 (Dyadic normal-form canonical readback)**.** *Let $D = (S, M, E, Z, R, L, H, C, P, N)$ and $D' = (S', M', E', Z', R', L', H', C', P', N')$ be accepted* $\mathsf{DyadicNormalForm}^{\uparrow}$ *carriers whose dyadic readback ledgers expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row through the displayed* $\mathsf{Rat}^{\uparrow}$ *surface. If both*

carriers satisfy the same zero classifier and scale-minimality rows, then their sign, mantissa, unary exponent, and dyadic readback rows agree under componentwise hsame *transport. Consequently every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumer reads a single canonical finite normal form rather than a family of scale-shifted dyadic representatives.*

Proof. Compare the two accepted packets through the displayed readback ledgers L and L'. The hypothesis says that the comparison is made inside the same $\mathsf{DyadicRatCore}^{\uparrow}$ boundary and through the same $\mathsf{Rat}^{\uparrow}$ readback surface.

If the common readback is zero, the classifier rows Z and Z' force both packets to use the distinguished zero row. Hence $M = M' = 0$ and $E = E' = 0$ after componentwise transport, and the sign rows agree with the zero orientation.

In the nonzero case, the scale-minimality rows R and R' force both mantissas to be odd at their displayed scales. A dyadic equality between two odd mantissas at different unary exponents would exhibit a remaining factor of 2 on one side after transporting to a common scale, contradicting the corresponding minimality row.

Thus the unary exponent rows agree, and then the displayed rational readback forces the signed mantissa rows and sign rows to agree. Transport by H and H' carries this agreement to the local naming rows, while C and C' replay the same finite route for downstream consumers.

The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff reads only the rows just compared. Therefore downstream consumers see the canonical finite normal form and not a separate scale-shifted representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicNormalForm}^{\uparrow}$ packet is generated from a sign row, signed mantissa, unary exponent, zero classifier, scale-minimality row, dyadic readback ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1613.1, Definition 1613.2, and Definition 1613.3 seed the canonical finite dyadic representation boundary feeding DyadicRatCore, StreamName, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, host real equality, order, completion, choice-selected representatives, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, zero exactness, scale-minimality, dyadic readback stability, canonical readback, downstream handoff, and ledger refusal proofs for BEDC.Derived.DyadicNormalFormUp.

Chapter 1614

A Concrete Naming Certificate for $\mathsf{Pseudometric}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{Pseudometric}^{\uparrow}$ packet records the pre-separated distance surface used by the constructive metric and completion route. It sits before the metric carrier of chapter 57, the separated reflection of chapter 1411, the located-distance interface of chapter 1533, and the completion boundary of chapter 95. The packet reads dyadic tolerance rows from chapter 291, finite windows from chapter 126, regular rational readback from chapter 301, and the terminal distance seal of chapter 38. Its purpose is to expose distance-zero evidence as finite NameCert data before any consumer turns that evidence into point identity.

Definition 1614.1 (Pseudometric carrier)**.** *A* $\mathsf{Pseudometric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, M, D, S, R, E, Z, H, C, N)$$

with the following displayed rows:

1. *X is the point carrier row. It records the admitted point names and their classifier, but it does not quotient points by zero distance.*
2. *M is the displayed distance witness. It names the finite comparison surface used to request a distance between two admitted point rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and triangle ledger. It records dyadic bounds, reflexive zero requests, symmetry swaps, and triangle estimates as displayed comparison rows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row controlling which point-name and distance-name windows may be inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the distance witness, giving regular rational approximants for requested windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *distance seal. It is reachable only after S, R, D have displayed the finite readback and tolerance evidence.*
7. *Z is the zero-distance boundary row. It records that zero distance is a boundary condition on the distance ledger, not an identification of point rows.*
8. *H records componentwise* hsame *transport for the point carrier, distance witness, dyadic ledger, stream windows, regular readback, Real seal, zero-distance boundary, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a consumer request through point admission, stream windows, regular readback, dyadic tolerance and triangle rows, Real distance sealing, and the zero-distance boundary.*

10. *N is both the* Pkg *provenance over the listed BEDC dependencies and the local* $\mathsf{NameCert}_{\mathsf{Pseudometric}^\uparrow}$ *row.*

The carrier has no coordinate for host equality of points, quotient completion, selected limits, countable choice, trichotomy, open-cover compactness, or a separated reflection.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricTasteGate single carrier alignment`

Definition 1614.2 (Pseudometric source carrier)**.** *A* $\mathsf{Pseudometric}^\uparrow$ *source carrier is a finite* BHist *packet*

$$P_s = (X, M, D, S, R, E, Z, H, C, N)$$

as in Definition 1614.1, with the additional source-side reading convention that D records metric-like distance rows while permitting zero distance between classifier-distinct point rows of X. The rows S, R, E expose the $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, *and* $\mathsf{Real}^\uparrow$ *readback of each requested distance, while Z records zero-distance evidence as a boundary row. Thus P_s is a source for separated reflection and completion consumers without identifying point names, quotienting the carrier, or importing ambient separated completion.*

Theorem 1614.3 (Pseudometric source obligation surface)**.** *For every accepted* $\mathsf{Pseudometric}^\uparrow$ *source carrier $P_s = (X, M, D, S, R, E, Z, H, C, N)$, the NameCert obligation surface is the conjunction of source carrier admission, dyadic distance readback through* $\mathsf{DyadicRatCore}^\uparrow$, *finite-window exposure through* $\mathsf{StreamName}^\uparrow$, *regular rational readback through* $\mathsf{RegSeqRat}^\uparrow$, *terminal distance sealing through* $\mathsf{Real}^\uparrow$, *zero-distance non-identification through Z, componentwise transport through H, replay through C, and local naming through N. The surface explicitly refuses quotient equality, host equality, and any ambient* $\mathsf{SeparatedMetric}^\uparrow$ *or* $\mathsf{Completion}^\uparrow$ *object not routed through these displayed rows.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier namecert obligation surface`

Proof. Start with the source carrier of Definition 1614.2. The tuple contains the point carrier, distance witness, dyadic ledger, finite stream windows, regular rational readback, Real seal, zero-distance row, transport, replay, and local naming row.

The distance side is read in order. A consumer may request a distance through M, but the observable data pass through $\mathsf{DyadicRatCore}^\uparrow$ in D, the $\mathsf{StreamName}^\uparrow$ windows in S, the $\mathsf{RegSeqRat}^\uparrow$ readback in R, and only then the $\mathsf{Real}^\uparrow$ seal in E.

The zero-distance side is deliberately separate from identity. The row Z records the boundary condition, while X retains classifier-distinct point rows and H, C, N merely transport, replay, and name the displayed packet.

Therefore a separated metric or completion consumer receives only this source surface. Quotient equality, host equality, and ambient separated completion are absent unless a later $\mathsf{SeparatedMetric}^\uparrow$ or $\mathsf{Completion}^\uparrow$ packet adds its own rows. □

Theorem 1614.4 (Pseudometric separated completion readback)**.** *Every read from a* $\mathsf{Pseudometric}^\uparrow$ *source carrier into a* $\mathsf{SeparatedMetric}^\uparrow$ *or* $\mathsf{Completion}^\uparrow$ *route factors through*

$$X, M, D, S, R, E, Z, H, C, N.$$

The SeparatedMetric$^{\uparrow}$ *route may consume Z as zero-distance evidence only after the* DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *distance rows have been displayed, and the* Completion$^{\uparrow}$ *route receives the same pre-separated source rows without quotient equality, host equality, selected representatives, selected limits, or ambient separated completion.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance boundary obligation`

Proof. Project the accepted source carrier. The only distance-read route is M, D, S, R, E, with D supplying dyadic tolerance and triangle rows, S supplying finite windows, R supplying regular rational readback, and E supplying the Real distance seal.

For the separated route, Z is read after that distance route as evidence of zero distance. It is not an equality constructor for point rows in X, so the separated packet must add its own reflection rows before point identity is available.

For the completion route, the consumer reads the same pre-separated source rows and the structural rows H, C, N. Since the carrier has no quotient, representative, limit selector, host equality, or ambient separated-completion coordinate, those objects cannot be exported from the pseudometric source. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated completion readback`

Theorem 1614.5 (Pseudometric separated-reflection factorization)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *source carrier. Any* SeparatedMetric$^{\uparrow}$ *or* Completion$^{\uparrow}$ *consumer that reads zero-distance evidence from P must first factor through the displayed rows*

$$X, M, D, S, R, E, Z, H, C, N.$$

Only a later separated-reflection packet may turn the zero-distance boundary row Z into point-identity data; the pseudometric source itself exports no quotient equality, host equality, selected representative, selected limit, or ambient separated completion.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance separated reflection exactness`

Proof. Project the source carrier of Definition 1614.2. The only available rows are the point carrier, distance witness, dyadic tolerance and triangle ledger, finite windows, regular readback, Real seal, zero-distance boundary, transport, replay, and local naming rows.

Apply Definition 1614.4. It forces both separated-metric and completion-facing reads through $X, M, D, S, R, E, Z, H, C, N$ before either consumer can inspect zero-distance evidence.

Use the zero-distance non-identification row. The row Z records boundary evidence on the displayed distance ledger, while point rows remain in X and are transported only by H and replayed only by C.

Finally apply the carrier exclusions. Since no quotient equality, host equality, selected representative, selected limit, or ambient separated-completion coordinate occurs in the source packet, identity data can appear only after a later separated-reflection packet supplies its own rows. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated reflection factorization`

Theorem 1614.6 (Pseudometric zero-distance scope)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *source carrier. Zero-distance evidence in P remains scoped to the displayed boundary row Z until a* SeparatedMetric$^{\uparrow}$ *consumer supplies its own reflection rows.* Completion$^{\uparrow}$ *and* LocatedMetric$^{\uparrow}$ *consumers may consume the distance witness, dyadic ledger, finite windows, regular readback, Real seal, transport, and replay rows, but they cannot turn Z into host equality, quotient equality, or selected representatives.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance scope`

Proof. Read the source carrier through Definition 1614.2. The row Z is part of the displayed pseudometric packet, while point admission remains in X and distance readback remains in M, D, S, R, E.

Apply Definition 1614.4. Completion-facing and separated-facing reads must factor through the displayed rows before zero-distance evidence is inspected, and the completion route receives no quotient equality, host equality, selected representative, selected limit, or ambient separated-completion coordinate.

Apply Definition 1614.5. The only route that may interpret zero distance as identity is a later separated-reflection packet with its own rows.

Thus a $\mathsf{Completion}^{\uparrow}$ or $\mathsf{LocatedMetric}^{\uparrow}$ consumer may use the distance and readback data exposed by M, D, S, R, E, H, C, N, but Z remains a scoped boundary row. Host equality, quotient equality, and selected representatives are absent until supplied by a distinct $\mathsf{SeparatedMetric}^{\uparrow}$ reflection surface. □

Theorem 1614.7 (Pseudometric NameCert obligations)**.** *For every accepted* $\mathsf{Pseudometric}^{\uparrow}$ *carrier* $P = (X, M, D, S, R, E, Z, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Pseudometric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the point carrier, displayed distance witness, dyadic tolerance and triangle ledger, StreamName windows, RegSeqRat readback, Real distance seal, zero-distance boundary, transport, continuation, provenance, and local naming rows;*
2. *distance readback, requiring every exported distance request to pass through* S, R, D *before the Real seal* E *is read;*
3. *reflexive, symmetric, and triangle stability, requiring zero self-distance, symmetry transport, and triangle estimates to be displayed as rows of* D *rather than imported as host metric laws;*
4. *classifier transport, requiring point and distance classifiers to move only by the componentwise* hsame *row* H *and replay row* C*;*
5. *zero-distance non-identification, requiring the row* Z *to record zero distance without equating the two point rows;*
6. *ledger non-escape, excluding quotient completion, host equality, selected limits, countable-choice selectors, trichotomy, open-cover compactness, and any separated reflection not routed through* $\mathsf{SeparatedMetric}^{\uparrow}$*.*

No pseudometric NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier namecert obligations`

Proof. Project the carrier of Definition 1614.1. The tuple lists exactly the point carrier, distance witness, dyadic ledger, stream windows, regular readback, Real seal, zero-distance boundary, transport, continuation, provenance, and local naming rows.

Distance readback follows the replay order in C. A consumer first reads the point names and finite stream windows, then the regular rational readback R and dyadic tolerance ledger D, and only then reaches the Real seal E.

Reflexive, symmetric, and triangle stability are ledger rows. The packet stores self-distance, symmetry, and triangle estimates inside D, so no host-level metric law can enter as a detached premise.

Classifier transport is componentwise. The row H transports exactly the displayed coordinates, while C replays them in the same order for consumers.

The zero-distance row Z is separate from point identity. Since X keeps point admission and Z keeps the boundary condition, the certificate exposes zero-distance evidence without identifying the point rows.

The remaining exclusions are immediate from the carrier coordinates. No quotient completion, host equality, selected limit, countable-choice selector, trichotomy, open-cover compactness, or separated reflection appears in P, so the listed rows exhaust the local NameCert surface. □

Theorem 1614.8 (Pseudometric separated reflection handoff)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Any consumer that wants point identity from zero distance must pass through the separated reflection boundary named by* $\mathsf{SeparatedMetric}^{\uparrow}$*. Consumers for* $\mathsf{Completion}^{\uparrow}$ *and* $\mathsf{LocatedMetric}^{\uparrow}$ *receive only the displayed pseudometric rows $X, M, D, S, R, E, Z, H, C, N$; they receive no quotient completion, host equality, selected limit, countable-choice selector, trichotomy, or open-cover compactness from the pseudometric packet itself.*

Proof. Start with a consumer of P. Carrier admission exposes the point carrier and distance rows, but the zero-distance row Z is only a boundary row on the displayed distance ledger.

If the consumer asks for point identity, the request is not answered inside P. The only named route from zero distance to point identity is the separated reflection interface of $\mathsf{SeparatedMetric}^{\uparrow}$.

Completion and located-distance consumers follow a different boundary. They may read the displayed point, distance, tolerance, stream, readback, Real seal, zero-distance, transport, replay, provenance, and local naming rows, but those rows do not identify points.

Because quotient completion, host equality, selected limits, countable-choice selectors, trichotomy, and open-cover compactness are absent from the carrier, no such datum can be exported along the completion or located-metric route. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated reflection handoff`

Theorem 1614.9 (Pseudometric zero-distance completion route)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Every zero-distance or completion-facing consumer must read the rows in the order*

$$X \longmapsto M \longmapsto D \longmapsto S \longmapsto R \longmapsto E \longmapsto Z \longmapsto H \longmapsto C \longmapsto N.$$

The only point-identity boundary available from the zero-distance row is the later $\mathsf{SeparatedMetric}^{\uparrow}$ *reflection route of Definition 1614.8; the pseudometric packet itself exports no quotient equality, host point equality, selected representative, or ambient separated completion.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance completion route`

Proof. Project the carrier of Definition 1614.1. The point admission row X is read before the distance witness M, the dyadic tolerance and triangle ledger D, the finite stream windows S, and the regular-rational readback R.

Use Definition 1614.7. It binds the terminal Real seal E, zero-distance boundary Z, componentwise transport H, replay C, and local naming row N to that same displayed distance route.

Now apply Definition 1614.8. A consumer may turn zero-distance evidence into point identity only after reading the separated metric boundary; the pseudometric carrier does not identify the source rows in X.

The exclusions follow from the displayed coordinate list. Since no coordinate stores quotient equality, host point equality, selected representatives, or ambient separated completion, a completion-facing consumer receives only the ordered finite route above. □

Theorem 1614.10 (Pseudometric quotient reflection route)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Every consumer that tries to turn zero-distance evidence from P into point identity must factor first through the displayed* SeparatedMetric$^{\uparrow}$ *reflection boundary and then through the existing* Met$^{\uparrow}$ *and* Completion$^{\uparrow}$ *consumers. The route preserves the rows $X, M, D, S, R, E, Z, H, C, N$ and exports no quotient completion, host equality, selected limit, countable choice, trichotomy, or open-cover compactness from the pseudometric packet itself.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier quotient reflection route`

Proof. Project the carrier of Definition 1614.1. Zero-distance evidence is stored in Z after the distance route through S, R, D, E, while point admission remains in X.

Apply Definition 1614.8. It forces any request for point identity from zero distance through the separated reflection boundary named by SeparatedMetric$^{\uparrow}$ before metric or completion consumers can read the result.

Use Definition 1614.11. The zero-distance read is a boundary ledger read through Z, H, C, N, so it preserves the displayed pseudometric rows rather than identifying points inside P.

Finally apply Definition 1614.13. Completion-facing consumers receive only $X, M, D, S, R, E, Z, H, C, N$ unless a later separated-metric boundary supplies stronger identity data, and the excluded quotient, choice, trichotomy, compactness, and host-equality objects are absent from the packet. □

Theorem 1614.11 (Pseudometric zero-distance boundary obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Every read that treats zero distance as available evidence must pass through the displayed rows Z, H, C, N after the distance request has been routed through S, R, D, E. The row Z records a boundary condition on the displayed distance ledger, H records componentwise* hsame *transport, C records replay through the same rows, and N records the local* NameCert *provenance. No such read identifies point rows in X or exports host equality from zero distance.*

Proof. Project the carrier of Definition 1614.1. The zero-distance coordinate is Z, while point admission remains in X and the distance route is displayed by M, D, S, R, E.

The replay row C forces a consumer to read stream windows, regular rational readback, the dyadic distance ledger, and the Real distance seal before reaching Z. The transport row H then moves only the displayed coordinates componentwise, and N names the local certificate row for the same packet.

Apply Definition 1614.8. It states that point identity from zero distance is not answered inside the pseudometric packet. Therefore Z, H, C, N expose zero-distance evidence as a boundary ledger and do not identify point rows in X. □

Theorem 1614.12 (Pseudometric triangle ledger obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Symmetry reads and triangle-estimate reads are ledger rows of D. Each such read is routed through the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the* StreamName$^{\uparrow}$ *finite windows, the* RegSeqRat$^{\uparrow}$ *readback row, and the terminal* Real$^{\uparrow}$ *distance seal before the transport, replay, provenance, and local naming rows are consumed. The packet provides no detached host symmetry law and no ambient metric triangle axiom outside this displayed ledger route.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier triangle ledger obligation`

Proof. Project the carrier of Definition 1614.1. The row D is explicitly the tolerance and triangle ledger, and the rows S, R, E are the only displayed route from finite windows and regular rational readback to the Real distance seal.

For a symmetry request, C replays the admitted point rows through S, R, D, E before any structural row is read. The symmetry datum is therefore a row of D, transported by H only after the same displayed distance route has been consumed.

For a triangle request, the same ledger row D supplies the dyadic estimates. The consumer then reaches E through S and R, and only afterward can read H, C, N as transport, replay, provenance, and naming rows.

Apply Definition 1614.8. Since separated identity and completion consumers receive only the displayed pseudometric rows, neither symmetry nor triangle data can enter as a detached host law. □

Theorem 1614.13 (Pseudometric completion nonescape obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Every* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{LocatedMetric}^{\uparrow}$ *consumer receives only the displayed point carrier, distance witness, dyadic ledger, stream windows, regular rational readback, Real distance seal, zero-distance boundary, componentwise transport, continuation replay, provenance, and local naming rows of P. The handoff exports no quotient completion, selected representative, trichotomy split, open-cover compactness datum, or separated point identity except through the named* $\mathsf{SeparatedMetric}^{\uparrow}$ *boundary.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier completion nonescape obligation`

Proof. Project the carrier of Definition 1614.1. Its coordinates are exactly $X, M, D, S, R, E, Z, H, C, N$, with no coordinate for a quotient completion, selected representative, trichotomy split, open-cover compactness datum, or separated point identity.

The completion-facing consumer can read distances only through S, R, D, E and can read zero-distance evidence only through Z, H, C, N. These are displayed pseudometric rows, not quotient-completion constructors or selectors.

The located-metric consumer receives the same surface. Its distance readbacks are routed through the dyadic, stream, regular rational, and Real rows, while the remaining structural information is componentwise transport, replay, provenance, and local naming.

Apply Definition 1614.8. Any conversion from zero distance to point identity is forced through $\mathsf{SeparatedMetric}^{\uparrow}$, so the pseudometric packet itself cannot export separated identity or any of the listed non-displayed principles. □

Theorem 1614.14 (Pseudometric scoped kernel dependency surface)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Its zero-distance boundary is consumed through chapter 1411, and its completion-facing route is consumed through chapter 95. The distance evidence underneath both consumers remains grounded in chapter 291, chapter 126, chapter 301, and chapter 38: X supplies the point carrier, M the displayed distance witness, D the dyadic tolerance and triangle ledger, S the finite window schedule, R the regular rational readback, E the terminal Real seal, Z the zero-distance boundary, H the componentwise* hsame *transport, C the replay route, and N the local* $\mathsf{NameCert}$ *row.*

Thus the separated metric sibling consumes Z, H, C, N only after the distance route M, D, S, R, E is visible, while the completion sibling consumes $X, M, D, S, R, E, Z, H, C, N$ as a pre-separated source. Neither sibling receives quotient classes, selected representatives, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the pseudometric packet.*

Proof. Project the carrier of Definition 1614.1. The displayed rows are exactly $X, M, D, S, R, E, Z, H, C, N$, so every sibling consumer starts from the same finite kernel surface.

Apply Definition 1614.15 and Definition 1614.12. These identify D, S, R, E as the dyadic, finite-window, regular-rational, and Real-seal route supporting all distance and triangle reads from M.

Use Definition 1614.8. A separated metric consumer may use Z as the zero-distance boundary only after the displayed distance route and the structural rows H, C, N have been read.

Finally apply Definition 1614.13. Completion-facing consumers receive the same pre-separated source rows and no quotient completion, selected representative, host equality, choice coordinate, or ambient completeness principle. The named sibling chapters therefore consume the displayed dependency surface rather than replacing it. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier scoped kernel dependency surface`

Theorem 1614.15 (Pseudometric distance readback obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Every consumer read of a distance witness must pass through the finite route S, R, D, E before any consumer may read the zero-distance boundary, componentwise transport, replay, provenance, or local naming rows Z, H, C, N. The route exposes stream windows, regular rational readback, the dyadic tolerance and triangle ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of Definition 1614.1; it exports no detached host distance or ambient metric law.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier distance readback obligation`

Proof. Project the carrier of Definition 1614.1. The distance witness is named by M, while the observable distance readback is routed through the stream window S, regular rational row R, dyadic ledger D, and Real seal E.

Apply Definition 1614.7. It lists distance readback as a local obligation requiring every exported distance request to pass through S, R, D before E is read.

The structural rows Z, H, C, N occur after the displayed distance route. Thus a consumer cannot read zero-distance evidence, componentwise transport, replay, provenance, or the local name before the finite distance route has been displayed.

Finally use Definition 1614.12. Symmetry and triangle estimates are rows of D, so the readback route does not import a detached host distance or ambient metric law. □

Theorem 1614.16 (Pseudometric classifier transport obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Point classifiers and distance classifiers transport only componentwise through the* hsame *row H and are replayed for consumers only through the displayed* Cont *row C. The transport reads no quotient completion, selected representative, separated point identity, host equality, or detached metric classifier outside the rows of Definition 1614.1.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier classifier transport obligation`

Proof. Start from the carrier definition. The row H is the only componentwise transport row for point admission, distance witness, dyadic ledger, stream windows, regular readback, Real seal, zero-distance boundary, continuation, provenance, and local naming.

Apply Definition 1614.7. Its classifier-transport clause requires point and distance classifiers to move only by H and replay row C.

The replay row C preserves the same order as the displayed distance route. It replays point admission, stream windows, regular readback, dyadic tolerance and triangle rows, Real sealing, and zero-distance boundary evidence before provenance and local naming are consumed.

Apply Definition 1614.8. Any conversion of zero distance into separated identity lies outside P and must pass through $\mathsf{SeparatedMetric}^{\uparrow}$, so transport inside P cannot smuggle in quotient completion, selected representatives, host equality, or a detached metric classifier. □

Theorem 1614.17 (Pseudometric ledger nonescape obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Quotient completion, host equality, selected limits, countable choice, trichotomy, open-cover compactness, and separated identity are absent from the pseudometric ledger unless a consumer routes the separated-identity request through* $\mathsf{SeparatedMetric}^{\uparrow}$*. The only ledger rows available to completion and located-metric consumers are the displayed coordinates $X, M, D, S, R, E, Z, H, C, N$.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier ledger nonescape obligation`

Proof. Project Definition 1614.1. The tuple contains the point carrier, displayed distance witness, dyadic ledger, stream windows, regular readback, Real seal, zero-distance boundary, transport, replay, provenance, and local naming rows, and it contains no coordinate for the excluded objects.

Apply Definition 1614.13. Completion and located-metric consumers receive only the displayed pseudometric rows and no quotient completion, selected representative, trichotomy split, open-cover compactness datum, or separated point identity except through $\mathsf{SeparatedMetric}^{\uparrow}$.

Apply Definition 1614.12. The metric-law surface is confined to the dyadic ledger D and the route through S, R, E, so host equality, countable choice, and ambient metric principles cannot enter through the distance side.

Finally apply Definition 1614.8. The only named route from zero distance to point identity is the separated reflection boundary, so separated identity is absent from the pseudometric ledger itself. □

Theorem 1614.18 (Pseudometric distance source obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. The displayed distance row M is the only source from which a consumer may request pseudometric distance data. Every such request is routed through the finite* BHist *rows S, R, D, E and is then accounted for by* hsame *transport H,* Cont *replay C, provenance, and the local* $\mathsf{NameCert}$ *row N. No consumer receives a detached host distance or an ambient metric law outside this displayed route.*

Proof. Project the carrier of Definition 1614.1. The distance witness is the row M, while finite windows, regular rational readback, dyadic ledger data, and the Real seal are displayed by S, R, D, E.

Apply Definition 1614.15. It forces every exported distance request through S, R, D, E before the structural rows are read.

The transport and replay rows are then H and C, and the local naming row is N. Since the carrier has no further distance coordinate, a consumer cannot obtain a detached host distance or ambient metric law from the pseudometric packet. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier distance source`

Theorem 1614.19 (Pseudometric kernel route obligation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Zero-distance reads and triangle-estimate reads remain kernel ledger reads until a consumer explicitly routes separated point identity through* $\mathsf{SeparatedMetric}^{\uparrow}$ *and real-valued distance sealing through* $\mathsf{Real}^{\uparrow}$*. The packet therefore exposes Z and D as displayed evidence rows, not as quotient completion, host equality, or separatedness.*

Proof. For zero-distance reads, use Definition 1614.11. It routes the read through S, R, D, E and then through Z, H, C, N, while leaving point rows in X unidentified.

For triangle-estimate reads, use Definition 1614.12. It keeps symmetry and triangle evidence inside the dyadic ledger D and routes those estimates through the same StreamName, RegSeqRat, and Real rows.

Finally apply Definition 1614.8. Point identity from zero distance is not answered inside P and must pass through $\mathsf{SeparatedMetric}^{\uparrow}$, so neither Z nor D can export quotient completion, host equality, or separatedness. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier kernel route`

Theorem 1614.20 (Pseudometric dependency-ready package)**.** *The* $\mathsf{Pseudometric}^{\uparrow}$ *chapter is dependency-ready for separated-completion and represented-space consumers at the level of displayed rows: it supplies the point carrier X, distance witness M, dyadic tolerance and triangle ledger D, StreamName windows S, RegSeqRat readback R, Real distance seal E, zero-distance boundary Z, componentwise transport H, replay C, provenance, and local naming N. This package does not claim separatedness, quotient completion, selected representatives, or completion uniqueness.*

Proof. Combine the carrier of Definition 1614.1 with the obligations of Definition 1614.7. Together they expose the point carrier, distance witness, dyadic ledger, stream windows, regular readback, Real seal, zero-distance boundary, transport, replay, provenance, and local naming rows.

Apply Definition 1614.18. It makes the displayed distance row the only source for distance requests and routes every such request through the finite BEDC rows.

Apply Definition 1614.19. It keeps zero-distance and triangle evidence in the displayed ledger until the separated reflection and Real sealing dependencies are read.

The exclusions follow from Definition 1614.13. The packet has no coordinate for separatedness, quotient completion, selected representatives, or completion uniqueness, so downstream consumers receive a dependency-ready surface without those stronger claims. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier dependency ready package`

Theorem 1614.21 (Pseudometric dependency unblock surface)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. The rows $X, M, D, S, R, E, Z, H, C, N$ are enough for dependency readiness of separated-completion and located-metric consumers: every such consumer can read point admission, distance source, dyadic tolerance and triangle evidence, finite windows, regular-rational readback, the Real distance seal, zero-distance boundary data, componentwise transport, replay, provenance, and local naming without receiving separated identity, quotient completion, selected representatives, or completion uniqueness.*

Proof. First project Definition 1614.1. It displays exactly $X, M, D, S, R, E, Z, H, C, N$ as the point, distance, tolerance, window, readback, Real-seal, zero-distance, transport, replay, provenance, and naming surface.

Apply Definition 1614.20. That theorem identifies the same rows as the dependency-ready package available to separated-completion and represented-space consumers.

Finally apply Definition 1614.13. It excludes separatedness, quotient completion, selected representatives, and completion uniqueness from the exported packet unless a later consumer routes a stronger claim through its own named boundary. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier dependency unblock surface`

Theorem 1614.22 (Pseudometric separated-reflection kernel saturation)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. The zero-distance kernel available to a separated-reflection consumer is saturated by the displayed route*

$$M, D, S, R, E, Z, H, C, N.$$

That is, every accepted zero-distance read must pass through the distance source, dyadic tolerance and triangle ledger, finite windows, regular-rational readback, Real seal, zero-distance boundary, componentwise hsame *transport, replay, and local naming rows before any* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer may inspect it; no quotient equality, host point identity, selected representative, or ambient separated metric row is supplied by* $\mathsf{Pseudometric}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated reflection kernel saturation`

Proof. Start with Definition 1614.21. It identifies the rows available to separated-completion and located-metric consumers and excludes separated identity and quotient completion from the packet itself.

Apply Definition 1614.8. A consumer that wants point identity from zero distance must pass through the later $\mathsf{SeparatedMetric}^{\uparrow}$ boundary after the zero-distance row has been read.

Use Definition 1614.9. The zero-distance route is already forced through $M, D, S, R, E, Z, H, C, N$, so the displayed rows saturate the kernel evidence available before separated reflection. □

Theorem 1614.23 (Pseudometric zero-distance classifier stability)**.** *Let* $P = (X, M, D, S, R, E, Z, H, C, N)$ *be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. If two zero-distance reads are transported by the packet's componentwise* hsame *row* H *and replayed by* C*, then their classifier-visible content remains the same displayed boundary route* $M, D, S, R, E, Z, H, C, N$*. The classifier may preserve the zero-distance evidence as a row of* Z*, but it cannot convert that evidence into point identity until a* $\mathsf{SeparatedMetric}^{\uparrow}$ *boundary is read.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance classifier stability`

Proof. Project the classifier transport obligation through Definition 1614.16. The only transport rows available to point and distance classifiers are the displayed hsame row H and replay row C.

Now use Definition 1614.11. Zero-distance evidence is a boundary row tied to the distance route through S, R, D, E and then to Z, H, C, N.

Finally apply Definition 1614.22. Since the saturated route contains no point-identity coordinate, classifier stability preserves the finite zero-distance row without adding separated equality. □

Theorem 1614.24 (Pseudometric completion universal handoff)**.** *Let* $P = (X, M, D, S, R, E, Z, H, C, N)$ *be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Every completion-facing consumer receives* P *through the universal pre-separated handoff consisting of the displayed rows* $X, M, D, S, R, E, Z, H, C, N$*, followed only by the external* $\mathsf{SeparatedMetric}^{\uparrow}$ *and* $\mathsf{Completion}^{\uparrow}$ *rows supplied by that consumer. The handoff carries zero-distance and distance-readback evidence, but it does not create completion points, identify quotient classes, select limits, or assert host equality inside* $\mathsf{Pseudometric}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCompletionUniversalHandoff`

Proof. Begin with Definition 1614.20. It packages the point carrier, distance witness, dyadic ledger, StreamName windows, RegSeqRat readback, Real seal, zero-distance boundary, transport, replay, provenance, and local naming rows as the dependency-ready surface.

Use Definition 1614.23. The zero-distance row remains stable under the displayed classifier transport, so completion consumers receive a coherent pre-separated source rather than a quotient construction.

Apply Definition 1614.25. Separated identity and completion points enter only through later $\mathsf{SeparatedMetric}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ boundaries. The pseudometric packet therefore provides the

universal handoff route without supplying selected limits, quotient classes, or host point equality. □

Theorem 1614.25 (Pseudometric separated-completion source)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Any separated-completion consumer that treats zero distance as point identity must route that identity claim through* SeparatedMetric$^{\uparrow}$ *after reading the displayed zero-distance boundary Z. The pseudometric packet itself supplies only the pre-separated source rows $X, M, D, S, R, E, Z, H, C, N$ and exports no host equality, quotient completion class, selected representative, or separated completion point.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated completion source`

Proof. Start with Definition 1614.11. It makes zero-distance evidence a read of Z, H, C, N after the distance route has passed through S, R, D, E.

Next use Definition 1614.8. It states that point identity from zero distance is not answered inside the pseudometric packet and must pass through the separated reflection boundary named by SeparatedMetric$^{\uparrow}$.

The completion-side exclusions then follow from Definition 1614.13. Since host equality, quotient completion classes, selected representatives, and separated completion points are absent from the carrier, the separated-completion consumer receives only the displayed pre-separated source rows. □

Theorem 1614.26 (Pseudometric separated-reflection source)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Every* SeparatedReflection$^{\uparrow}$ *consumer receives the source only through the displayed rows of Definition 1614.1: point admission X, distance witness M, dyadic tolerance and triangle ledger D, StreamName windows S, RegSeqRat readback R, Real seal E, zero-distance boundary Z, componentwise transport H, replay C, provenance, and local naming N. In particular, the consumer may use zero-distance evidence only through Definition 1614.11 and Definition 1614.8; it receives no point identity, quotient class, selected representative, or host equality before* SeparatedMetric$^{\uparrow}$ *supplies a stronger boundary.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated reflection source`

Proof. Project the carrier rows from Definition 1614.1. They are exactly $X, M, D, S, R, E, Z, H, C, N$, so a separated-reflection consumer has no other coordinate to inspect.

Apply Definition 1614.11. It keeps zero-distance evidence in the displayed Z, H, C, N route after the distance read has passed through S, R, D, E.

Finally use Definition 1614.8 and Definition 1614.13. The first theorem forces any identity request through SeparatedMetric$^{\uparrow}$, and the second excludes quotient classes, selected representatives, and host equality from the pseudometric packet. □

Theorem 1614.27 (Pseudometric pre-separated completion source)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. A completion consumer that has not yet read a* SeparatedMetric$^{\uparrow}$ *or* Completion$^{\uparrow}$ *boundary can use P only as a pre-separated source: it may read the displayed distance, tolerance, finite-window, regular-rational, Real-seal, zero-distance, transport, replay, provenance, and naming rows, but it cannot read separated completion points, completion uniqueness, quotient completion, selected limits, host equality, or ambient* Real$^{\uparrow}$ *completeness from P.*

Proof. The pre-separated source is the carrier of Definition 1614.1. Its displayed rows provide distance witnesses, tolerance and triangle ledgers, finite windows, regular-rational readback, Real sealing, zero-distance boundary data, transport, replay, provenance, and local naming.

Use Definition 1614.8. Any move from zero distance to point identity is outside P until the separated metric boundary is read.

Use Definition 1614.13. Completion uniqueness, quotient completion, selected limits, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the packet, so a completion consumer must wait for $\mathsf{Completion}^{\uparrow}$ or a later named completion boundary to supply them. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier completion universal handoff`

Theorem 1614.28 (Pseudometric located-metric source exhaustion)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted $\mathsf{Pseudometric}^{\uparrow}$ packet. Every located-metric consumer of P exhausts the available source data by reading the displayed distance source M, dyadic tolerance and triangle ledger D, StreamName windows S, RegSeqRat readback R, Real seal E, zero-distance boundary Z, and structural rows H, C, N. After these rows are read, no further located-metric structure remains in P; stronger separatedness, completion, host-equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness claims must come from $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, or another named consumer boundary.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier located metric source exhaustion`

Proof. Begin with Definition 1614.1. The only located-metric-facing coordinates are the displayed distance source, tolerance and triangle ledger, finite windows, regular-rational readback, Real seal, zero-distance boundary, and structural rows.

Apply Definition 1614.11. It identifies the zero-distance part of this surface as boundary evidence rather than separated identity.

Then apply Definition 1614.8 and Definition 1614.13. They force stronger separated or completion claims outside the pseudometric packet and leave the listed rows as the exhausted located-metric source surface. □

Theorem 1614.29 (Pseudometric located-metric readiness)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted $\mathsf{Pseudometric}^{\uparrow}$ packet. Every located-metric consumer of P reads only the displayed distance source M, the dyadic tolerance ledger D, the finite window row S, the regular-rational readback R, the terminal $\mathsf{Real}^{\uparrow}$ distance seal E, and the structural rows H, C, N. It receives no detached host distance, ambient metric law, trichotomy split, open-cover compactness datum, quotient completion, or separated point identity from the pseudometric packet.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier located metric readiness`

Proof. Apply Definition 1614.18. It names M as the only source for pseudometric distance requests and routes every such request through S, R, D, E before the structural rows are read.

Use Definition 1614.15. The readback route exposes finite windows, regular rational readback, dyadic tolerance and triangle data, and the terminal $\mathsf{Real}^{\uparrow}$ seal as displayed rows rather than as detached host distance data.

Finally apply Definition 1614.17. The located-metric consumer receives only the displayed ledger rows and cannot extract ambient metric laws, trichotomy, open-cover compactness, quotient completion, or separated point identity from the pseudometric packet. □

Theorem 1614.30 (Pseudometric finite-window distance route)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* Pseudometric$^{\uparrow}$ *packet. Every finite distance read exported by P is routed through the carrier row X, the displayed distance source M, the dyadic tolerance and triangle ledger D, the finite-window row S, the regular-rational readback row R, the terminal* Real$^{\uparrow}$ *seal E, the zero-distance boundary Z, and the structural rows H, C, N. The route is exactly the distance-readback surface of Definition 1614.15 together with the triangle-ledger control of Definition 1614.12, and it does not expose a detached host distance, separated point identity, quotient completion, selected limit, or ambient* Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier finite window distance route`

Proof. Start with the carrier projection of Definition 1614.1. The accepted packet gives the rows $X, M, D, S, R, E, Z, H, C, N$ and no additional distance coordinate.

Apply Definition 1614.15. A distance request must pass through the finite windows S, the regular-rational readback R, the dyadic ledger D, and the terminal seal E before the structural rows can package the read.

Apply Definition 1614.12. The same ledger row D is the only triangle-facing control row, so finite-window distance reads and triangle comparisons share the displayed tolerance route rather than splitting into a host metric law.

Finally use Definition 1614.13. Completion, selected-limit, host-equality, and ambient-completeness claims are outside the pseudometric packet, while Z remains only the zero-distance boundary available for later separated reflection. □

Theorem 1614.31 (Pseudometric zero-distance separated-reflection exactness)**.** *Let $P = (X, M, D, S, R, E, Z, H,$ be an accepted* Pseudometric$^{\uparrow}$ *packet. The zero-distance row Z is the exact boundary offered to separated-reflection consumers: it can be transported with H, replayed through C, packaged by the local naming row N, and compared with the distance-readback route M, D, S, R, E, but it becomes separated point identity only after the consumer reads the* SeparatedMetric$^{\uparrow}$ *boundary named in Definition 1614.8. The packet therefore exposes zero-distance evidence without quotient completion, selected limits, host equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricZeroDistanceSeparatedReflectionExactness`

Proof. Project the rows of P. The zero-distance row Z is present beside the distance source M, tolerance ledger D, finite windows S, regular readback R, and Real seal E.

Use Definition 1614.15 to align Z with the displayed distance route. The row can be inspected only against the same finite-window and dyadic readback surface that supplies distance evidence.

Now apply Definition 1614.8. That theorem places separated point identity beyond the pseudometric packet, at the later separated metric boundary, so Z is exact as boundary evidence and not as identity in X.

The non-escape clause follows from Definition 1614.13. Rows H, C, N preserve and name the boundary route, but no row stores quotient completion, selected limits, host equality, countable choice, or ambient completeness. □

Theorem 1614.32 (Pseudometric triangle separated-reflection compatibility)**.** *Let $P = (X, M, D, S, R, E, Z, H, C$ be an accepted* Pseudometric$^{\uparrow}$ *packet. The triangle information carried by D is compatible with separated reflection exactly along the displayed route $M, D, S, R, E, Z, H, C, N$: triangle comparisons are read through the same finite-window and regular-rational distance surface as zero-distance evidence, and any separated identity conclusion must pass through Definition 1614.8. The compatibility claim uses the distance-readback obligation, triangle-ledger obligation, and completion*

non-escape boundary, and it does not add an ambient metric law, quotient completion, selected limit, host equality, trichotomy split, or open-cover compactness datum.

Proof. Begin with Definition 1614.12. It identifies the triangle-facing content of P with the dyadic ledger row D, not with an external metric law.

Apply Definition 1614.15. The triangle comparison and the zero-distance comparison both read the finite windows S, regular-rational readback R, dyadic tolerance row D, and terminal Real seal E.

Use Definition 1614.8 for the separated boundary. If a consumer turns zero-distance evidence into point identity, that step is performed by the separated metric route, not by the pseudometric carrier itself.

Finally invoke Definition 1614.13. The same structural rows H, C, N preserve the displayed compatibility route while refusing quotient completion, selected limits, host equality, trichotomy, open-cover compactness, and ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier triangle separated reflection compatibility`

Theorem 1614.33 (Pseudometric zero-distance source reflection)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Every accepted zero-distance read from Z factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and triangle ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, the separated-reflection boundary of Definition 1614.31, and the completion source route of Definition 1614.27 before any* $\mathsf{Met}^{\uparrow}$ *consumer may read it as identity-like evidence. The source reflection is transported only by H, replayed only by C, and packaged only by the local* Pkg *and* $\mathsf{NameCert}$ *rows; it does not assert quotient identity, separated metric equality, host equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 1614.31. It identifies Z as exact boundary evidence and places separated point identity beyond the pseudometric packet.

Use Definition 1614.32. The zero-distance read and the triangle comparison share the displayed route through D, S, R, E, Z, H, C, N, so the source reflection remains tied to dyadic tolerance, finite windows, regular readback, and the Real seal.

Apply Definition 1614.27. Completion-facing consumers must first read the same pre-separated source rows before any metric or separated completion route can consume the evidence.

Finally project the structural rows. The BHist packet, H, C, local Pkg, and $\mathsf{NameCert}$ rows carry the source reflection, while the carrier has no coordinate for quotient identity, separated metric equality, host equality, selected limits, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier zero distance source reflection`

Theorem 1614.34 (Pseudometric kernel source carrier)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. The concrete source carrier available to every root-unblock consumer is exactly the finite* BHist *carrier row X together with the displayed distance source M, dyadic tolerance and triangle ledger D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, regular-rational readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, zero-distance boundary Z, componentwise* hsame *transport H, displayed* Cont *replay C, package provenance, and local* $\mathsf{NameCert}$ *row N. No consumer may replace this source by a host point set, quotient carrier, separated reflection, selected completion point, or ambient metric object.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier kernel source carrier`

Proof. Project the carrier of Definition 1614.1. The tuple displays exactly the rows $X, M, D, S, R, E, Z, H, C, N$, so the source carrier is a finite BEDC packet rather than an external set of points.

Apply Definition 1614.33. It routes every zero-distance source read through the dyadic ledger, stream windows, regular readback, Real seal, and structural rows before any metric-like consumer can inspect it.

Use Definition 1614.13. The excluded host point set, quotient carrier, selected completion point, separated reflection, and ambient metric object are absent from the carrier inventory, so the displayed rows exhaust the root-unblock source. □

Theorem 1614.35 (Pseudometric classifier stability)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Classifier transport for point rows and zero-distance rows is stable exactly along the displayed* hsame *and* Cont *route H, C: if two accepted reads share the same finite* BHist *carrier row X and the same transported zero-distance boundary Z, then any classifier comparison offered to a* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*, or completion-facing consumer must pass through $M, D, S, R, E, Z, H, C, N$. The comparison may cite the zero-distance row, but it cannot identify point names until the separated-reflection handoff is displayed.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier classifier stability`

Proof. Begin with Definition 1614.16. It confines point and distance classifier transport to the componentwise hsame row H and the displayed replay row C.

Next read the zero-distance route through Definition 1614.31. The row Z is boundary evidence aligned with M, D, S, R, E, not an equality row for point names in X.

Finally apply Definition 1614.8. A separated identity conclusion belongs to the later separated metric boundary. Thus classifier stability is available to consumers as a transported finite row, while point identification remains outside the pseudometric packet. □

Theorem 1614.36 (Pseudometric separated quotient handoff)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. The separated quotient handoff ledger offered by P is the displayed route from X through $M, D, S, R, E, Z, H, C, N$ to the separated-reflection boundary of Definition 1614.8 and the completion source of Definition 1614.27. The handoff supplies source rows for later separated reflection and completion consumers, but it does not form quotient classes, choose representatives, assert host equality, or close separated point identity inside* $\mathsf{Pseudometric}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier separated quotient handoff`

Proof. Start with Definition 1614.34. It fixes the source carrier as the displayed finite packet $X, M, D, S, R, E, Z, H, C, N$.

Use Definition 1614.35. Classifier comparisons and zero-distance boundary rows are transported only through H and replayed only through C, so the handoff is a ledger route rather than a quotient construction.

Now apply Definition 1614.8 and Definition 1614.27. These theorems route separated reflection and completion consumers through the displayed source rows while keeping quotient classes, selected representatives, host equality, and separated point identity outside the pseudometric packet. □

Theorem 1614.37 (Pseudometric bridged route)**.** *Let $P = (X, M, D, S, R, E, Z, H, C, N)$ be an accepted* $\mathsf{Pseudometric}^{\uparrow}$ *packet. Its public route bridges to separated-metric and completion consumers only through* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. The bridge refuses quotient equality,*

host point equality, selected representatives, selected limits, countable choice, and ambient separated completion.

Lean4 ✓ `BEDC.Derived.PseudometricUp.PseudometricCarrier bridged route`

Proof. Project the finite carrier of Definition 1614.1. The packet exposes point carrier data, distance data, dyadic tolerances, stream windows, regular readback, the Real seal, zero-distance boundary evidence, transport, replay, provenance, and local naming.

Use Definition 1614.36. It sends separated reflection and completion consumers through the displayed rows $X, M, D, S, R, E, Z, H, C, N$, so the later $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{Completion}^{\uparrow}$ reads receive a finite handoff rather than a quotient carrier.

Finally read the refusal side of the same carrier. Because no coordinate supplies host point equality, selected representatives, selected limits, countable-choice schedules, or ambient separated completion, the bridge-facing route is exactly the displayed BEDC packet route. □

Closure status for $\mathsf{NameCert}_{\mathsf{Pseudometric}^{\uparrow}}$.

Constructive story: A $\mathsf{Pseudometric}^{\uparrow}$ packet is generated from point carrier rows, displayed distance witnesses, DyadicRatCore tolerance and triangle ledgers, StreamName windows, RegSeqRat readback, a Real distance seal, a zero-distance boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Pseudometric}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1614.1, Definition 1614.7, Definition 1614.30, Definition 1614.31, Definition 1614.32, and Definition 1614.36 expose the public pre-separated distance, zero-boundary, separated-reflection, and quotient-handoff routes before completion or located-metric consumers read the packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Pseudometric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close separated point identity, quotient completion, host equality, selected limits, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: No further paper-theory upgrade remains; the remaining work is verification and bridge-axis checking for BEDC.Derived.PseudometricUp.

Chapter 1615

A Concrete Naming Certificate for $\mathsf{RiemannSum}^{\uparrow}$

Following Riemann and Darboux, the $\mathsf{RiemannSum}^{\uparrow}$ packet records the finite tagged-sum surface used before a consumer asks for an integral carrier. It sits between the dyadic mesh ledger of chapter 358, the finite partition surface of chapter 201, the stepwise approximation rows of chapter 428, the Darboux comparison carrier of chapter 1460, and the broader tagged-mesh handoff of chapter 1526. The packet names only the finite mesh, tag, value, width, and arithmetic rows of a displayed Riemann sum; it does not assert host integrability, a limiting process, quotient equality of functions, or choice-selected tags outside the packet.

Definition 1615.1 (Riemann sum carrier)**.** *A* $\mathsf{RiemannSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, T, F, W, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the mesh row, read through* $\mathsf{DyadicMesh}^{\uparrow}$ *and the finite* $\mathsf{Partition}^{\uparrow}$ *boundary, naming the ordered cells, endpoints, and adjacency ledger.*
2. *T is the tag-containment row, recording one displayed tag for each carried cell together with the dyadic or rational containment witness that places the tag inside that cell.*
3. *F is the integrand-read window, exposing exactly the finitely many carried values needed at the displayed tags.*
4. *W is the cell-width ledger, recording the width row attached to each mesh cell through the same dyadic or rational endpoint data.*
5. *S is the finite sum row, computed from the displayed tag values and cell widths by the carried arithmetic route.*
6. *H records componentwise* hsame *transport for mesh, tag containment, integrand reads, width rows, sum row, continuation row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from mesh cells to tag reads, from tag reads and width rows to summands, and from summands to the finite sum row.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, T, F, W, S, H, C, P, N$. It has no coordinate for a host limit over refinements, a measure-theoretic integral, a quotient function space, a tag-choice principle, or a completed $\mathsf{Real}^{\uparrow}$ *value.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumTasteGate single carrier alignment`

Theorem 1615.2 (Riemann sum NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSum}^{\uparrow}$ *carrier $Q = (M, T, F, W, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}$ *surface consists of carrier admission, mesh-tag containment, finite integrand readback, cell-width exposure, finite sum exactness, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged entirely by the displayed route*

$$M \longmapsto T \longmapsto F, W \longmapsto S$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier namecert obligations`

Proof. Carrier admission is the projection of the finite tuple in Definition 1615.1. The mesh row M fixes the displayed cell list before any tag or value row is read, and the tag-containment row T attaches one tag witness to each carried cell.

The value and width rows are then read componentwise. The window F exposes only the tag values named by T, while W exposes only the widths of the cells named by M. No external function equality or hidden refinement family is consulted.

Finite sum exactness is the arithmetic replay from these displayed rows to the row S. Since every summand is formed from one carried tag value and one carried width row, the sum is a finite BHist computation rather than a limiting assertion.

Transport and continuation use H and C to replay the same finite route after componentwise hsame movement. The provenance row P and local name N source the packet, and the absence of any host-limit, measure, quotient, or tag-choice coordinate gives ledger non-escape. □

Theorem 1615.3 (Riemann sum tagged-mesh exactness)**.** *Every accepted* $\mathsf{RiemannSum}^{\uparrow}$ *read factors through the displayed mesh row, the tag-containment row, the integrand-read window, the cell-width ledger, and the finite sum row. If two accepted packets agree on those rows under componentwise* hsame *transport, then every downstream consumer reads the same finite tagged-sum boundary through C, P, N.*

Proof. Project the accepted carrier through Definition 1615.1. The only route to a summand begins at a cell of M, passes through its tag witness in T, reads the corresponding value in F, and multiplies by the corresponding width in W.

Agreement under componentwise hsame transport identifies those rows cell by cell. Replaying C therefore rebuilds the same summand ledger and the same finite row S on both packets.

The downstream consumer has access only to S together with the replay, provenance, and local name rows. Hence agreement on the displayed tagged-mesh surface determines the public finite-sum boundary, with no extra channel for hidden tags or host function equality. □

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier tagged mesh exactness`

Theorem 1615.4 (Riemann sum Darboux handoff)**.** *An accepted* $\mathsf{RiemannSum}^{\uparrow}$ *packet can be handed to a* $\mathsf{DarbouxIntegral}^{\uparrow}$ *or* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer only as the finite tagged-mesh row consisting of* $M, T, F, W, S, H, C, P, N$*. The handoff supplies the sum side of a Darboux or Riemann comparison, but it does not supply an integral value, a convergence claim over all refinements, a quotient of integrands, or a choice-selected tag family.*

Proof. Start from the obligation surface of Definition 1615.2. It exhausts the packet as mesh, tag-containment, value, width, finite-sum, transport, replay, provenance, and local-name data.

A $\mathsf{DarbouxIntegral}^{\uparrow}$ consumer may compare this finite row with upper and lower step-function rows, and a $\mathsf{RiemannIntegral}^{\uparrow}$ consumer may place it inside a larger tagged-mesh integration packet. In both cases the consumer receives the same finite sum row S together with its visible construction route.

The excluded analytic claims are not carrier coordinates. The packet has no row for a completed integral, no family of all refinements, no host quotient of functions, and no choice-selected tag assignment outside T. Thus the handoff is exactly the finite tagged-sum side of the integration boundary. □

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier darboux handoff`

Theorem 1615.5 (Riemann sum mesh-tag obligation)**.** *Every obligation-level* $\mathsf{RiemannSum}^{\uparrow}$ *consumer of an accepted carrier* $Q = (M, T, F, W, S, H, C, P, N)$ *must expose the mesh row* M *and tag-containment row* T *before reading an integrand value, a cell-width ledger, or the finite sum row. The mesh row supplies the displayed cells, and* T *supplies exactly one carried tag-containment witness for each cell named by* M*.*

Proof. Project the carrier of Definition 1615.1. Its only mesh coordinate is M, and its only tag-containment coordinate is T.

Apply the NameCert obligations of Definition 1615.2. They place mesh-tag containment before finite integrand readback, width exposure, and sum exactness.

Use tagged-mesh exactness and the Darboux handoff, Definition 1615.3 and Definition 1615.4. Both force downstream comparison reads through the same displayed M, T boundary, so no consumer can substitute hidden cells or choice-selected tags. □

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier mesh tag obligation`

Theorem 1615.6 (Riemann sum width and sum exactness obligation)**.** *For every accepted* $\mathsf{RiemannSum}^{\uparrow}$ *carrier* $Q = (M, T, F, W, S, H, C, P, N)$*, the cell-width ledger* W *and finite sum row* S *must be replayed from the same mesh row* M *and tag-containment row* T*. The obligation surface admits a summand only when its tag value is read through* F *at a tag contained by* T *and its width is read through* W *at the corresponding cell of* M*.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier width sum exactness obligation`

Proof. Start with the carrier and NameCert obligation surface. The rows M, T, F, W, S are the only finite arithmetic rows available to the packet, and Definition 1615.2 orders them as mesh and tag data before value, width, and sum data.

Apply tagged-mesh exactness. Theorem 1615.3 says every summand route begins with a cell of M, passes through its tag witness in T, and then reads the matching value and width rows before rebuilding S.

The Darboux handoff preserves this same finite arithmetic boundary. A comparison consumer may inspect the sum side of the bracket, but it receives S only as the replay of M, T, F, W, not as a host limiting object. □

Theorem 1615.7 (Riemann sum ledger non-escape obligation)**.** *An accepted* $\mathsf{RiemannSum}^{\uparrow}$ *packet has no obligation-level route to host limits, quotient function equality, measure integration, or choice-selected tag families. Every downstream read remains inside the displayed finite ledger* $M, T, F, W, S, H, C, P, N$ *before any Darboux or Riemann integral consumer adds its own rows.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier ledger nonescape obligation`

Proof. Project the carrier of Definition 1615.1. The packet contains mesh, tag, finite value, width, sum, transport, replay, provenance, and local naming rows, and no coordinate for host analytic structure.

Apply the NameCert obligations and tagged-mesh exactness. These theorems force all accepted reads through the finite row route $M, T, F, W, S, H, C, P, N$.

Finally use the Darboux handoff. Theorem 1615.4 identifies the exported object as the finite tagged-sum side of a comparison, so host limits, quotient function equality, measure integration, and external tag choice remain outside the ledger. □

Theorem 1615.8 (Riemann sum obligation-closure package)**.** *The* $\mathsf{RiemannSum}^{\uparrow}$ *packet supports obligationClosure exactly by carrier admission, mesh-tag containment, finite integrand readback, cell-width exposure, finite-sum exactness, tagged-mesh exactness, Darboux/Riemann handoff, componentwise transport,* Cont *replay, provenance, and ledger non-escape. For an accepted carrier* $Q = (M, T, F, W, S, H, C, P, N)$*, every obligation-level consumer reads only the finite route through those nine rows and imports no integrability theorem, limiting process, quotient function equality, or tag-choice principle.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier obligation closure package`

Proof. Begin with the carrier of Definition 1615.1. It displays exactly $M, T, F, W, S, H, C, P, N$, so carrier admission and ledger non-escape are finite row projections rather than analytic closure assumptions.

Apply Definition 1615.2. The mesh row, tag-containment row, integrand-read window, cell-width ledger, finite sum row, transport, replay, provenance, and local naming rows give the obligation surface named in the statement.

Next use Definition 1615.3. It identifies downstream reads when the mesh, tags, finite values, widths, and sum row agree under componentwise hsame, giving the exact tagged-mesh boundary required by obligationClosure.

Finally apply Definition 1615.4. DarbouxIntegral and RiemannIntegral consumers receive only the finite tagged-sum side of the comparison. Since the packet has no row for integrability, all refinements, quotient function equality, or a choice-selected tag family, the obligation-closure package is exactly the listed finite surface. □

Theorem 1615.9 (Riemann sum public finite tagged-sum interface)**.** *The public export of an accepted* $\mathsf{RiemannSum}^{\uparrow}$ *carrier* $Q = (M, T, F, W, S, H, C, P, N)$ *is exactly the finite tagged-sum boundary* $M, T, F, W, S, H, C, P, N$*. Darboux and Riemann integral packets may consume that boundary as displayed mesh, tag-containment, integrand-read, cell-width, finite-sum, transport, replay, provenance, and local naming data, but the export contains no host limit, quotient function, measure integral, external tag selector, or completed integral value.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier public finite tagged sum interface`

Proof. Project the carrier through Definition 1615.1. Its displayed coordinates are precisely the finite tagged-sum rows $M, T, F, W, S, H, C, P, N$, with no additional public coordinate.

Apply the NameCert obligations and tagged-mesh exactness, Definition 1615.2 and Definition 1615.3. These identify the mesh, tag, value, width, and sum rows as the exact finite arithmetic route, preserved under componentwise transport and replay.

Use the Darboux handoff and obligation-closure package, Definition 1615.4 and Definition 1615.8. A Darboux or Riemann integral consumer receives only the displayed finite tagged-sum side of its comparison, so host limits, quotient functions, measure integrals, external tag selectors, and completed integral values remain outside the public export. □

Theorem 1615.10 (Riemann sum tagless refinement ledger)**.** *Let $Q = (M, T, F, W, S, H, C, P, N)$ be an accepted* $\mathsf{RiemannSum}^{\uparrow}$ *carrier. Any refinement-facing* $\mathsf{DarbouxIntegral}^{\uparrow}$ *or* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of Q must replay the same nine rows $M, T, F, W, S, H, C, P, N$. It may compare refinements only through displayed mesh-cell subdivision and transported tag-containment rows. The packet exports no external tag selector, quotient integrand equality, host limiting process, or measure integral.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier tagless refinement ledger`

Proof. Apply the mesh-tag obligation of Definition 1615.5. A refinement-facing consumer must first expose the displayed mesh row M and the tag-containment row T, so every subdivision comparison is anchored in the carried cells and carried tag witnesses.

Apply the width and sum exactness obligation of Definition 1615.6. The rows F, W, S are replayed from the same displayed mesh and tag route, hence a refined comparison may inspect only the subdivided cell-width ledger and the finite summand ledger produced from it.

Transport is componentwise through H and continuation replay is displayed through C. Thus transported tag-containment rows remain the only tag data available after refinement; no external selector can be projected from the packet.

Finally use ledger non-escape and the Darboux handoff, Definition 1615.7 and Definition 1615.4. They identify the exported object as the finite tagged-sum side of a Darboux or Riemann comparison, excluding quotient integrand equality, host limiting processes, and measure integrals. □

Theorem 1615.11 (Riemann sum mature finite mesh examples)**.** *The* $\mathsf{RiemannSum}^{\uparrow}$ *packet supports mature finite examples by displaying concrete rows for a mesh list, one tag-containment witness per cell, one carried value at each tag, one width row per cell, and the resulting finite sum row. A Darboux or Riemann consumer may read these examples as the finite comparison side of its own packet, but the example surface still stops before host integrability, convergence over all refinements, quotient equality of integrands, external tag choice, or a completed integral value.*

Lean4 ✓ `BEDC.Derived.RiemannSumUp.RiemannSumCarrier mature finite mesh examples`

Proof. Choose any displayed finite mesh row M and attach a tag-containment row T with one carried tag witness for each cell. The value window F and width ledger W then supply exactly the summand data named by the cells and tags.

The finite sum row S is obtained by replaying the carried arithmetic route from M, T, F, W. By Definition 1615.6, no summand is admitted unless its tag value and width are read from the same displayed cell route.

Use Definition 1615.9 and Definition 1615.10. They make the examples public as finite tagged-sum boundaries for Darboux and Riemann consumers while keeping completeness, all-refinement

convergence, quotient equality, external tag selection, and completed integral values outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSum}^{\uparrow}$ packet is generated from a finite mesh row, tag-containment row, integrand-read window, cell-width ledger, finite sum row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: The public finite tagged-sum interface and finite mesh example surface are closed for BHist mesh rows, $\mathsf{Partition}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}$ row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host integrability, convergence over all refinements, measure-theoretic integration, quotient function equality, choice-selected tags, Lean verification, or bridge checking.

Upgrade path: Bridged closure requires Lean verification and a checked standard bridge for the public finite tagged-sum interface of BEDC.Derived.RiemannSumUp.

Chapter 1616

A Concrete Naming Certificate for $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$ packet records the finite cofinal-rate surface used when a regular rational stream must answer every later precision request through an explicit late-window route. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the tolerance ledger of chapter 291, the effective threshold surface of chapter 415, and the terminal seal of chapter 38. The packet records a displayed cofinal-rate row and its finite handoff to the Real boundary; it is not a quotient of streams, a selected real value, a host metric limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1616.1 (Regular Cauchy cofinal-rate carrier)**.** *A* $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, K, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational approximants are the only numeric data read by the packet.*
2. *K is the cofinal rate row, read through the* $\mathsf{CauchyRate}^{\uparrow}$ *surface and assigning to each displayed dyadic tolerance a late stream stage sufficient for all later windows.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row reached at and beyond the stage named by K.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through which every requested radius is named before the cofinal row is inspected.*
5. *Q is the RegSeqRat readback row, exposing the regular rational comparison certified by R, W, D, K.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the source, rate, window, tolerance, and readback rows have been replayed.*
7. *H records componentwise* hsame *transport for the RegSeqRat source, cofinal rate, StreamName windows, DyadicRatCore ledger, readback, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through dyadic tolerance naming, cofinal-rate lookup, stream-window extraction, regular rational readback, and Real seal handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $R, K, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable-choice schedules, or ambient Real completeness.

Theorem 1616.2 (Regular Cauchy cofinal-rate NameCert obligations)**.** *For every accepted* RegularCauchyCofinalRa *carrier $G = (R, K, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalRate}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the RegSeqRat source, cofinal rate, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, transport, continuation, provenance, and local naming row;*
2. *tolerance-first routing, requiring each precision request to enter through D before the cofinal stage in K is read;*
3. *cofinal window exactness, requiring K to name a late stage after which W supplies every finite stream window consumed by the readback row;*
4. *readback stability, requiring Q, H, C to preserve the regular rational comparison under componentwise* hsame *transport and displayed* Cont *replay;*
5. *Real seal handoff, requiring E to be downstream of R, K, W, D, Q and to expose no quotient stream, selected limit, host equality, trichotomy, countable-choice selector, or ambient completeness principle.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1616.1. The projection exposes exactly the RegSeqRat source row, cofinal rate row, StreamName window row, DyadicRatCore ledger, readback row, Real seal, and structural rows H, C, P, N.

Tolerance-first routing follows from the order recorded by C. A precision request must name its radius through D before the cofinal rate row K can select the late stage.

The selected stage then controls the visible stream data. The window row W supplies only finite StreamName observations at and beyond the stage named by K, and the readback row Q consumes those observations with the source row R.

Transport and replay cannot add further obligations. The rows H and C preserve the displayed route, while P and N name the same finite packet.

Since quotient streams, selected limits, host equality, trichotomy, choice schedules, and ambient completeness data are absent from the carrier, the Real seal row E has no path to export them. □

Theorem 1616.3 (Regular Cauchy cofinal-rate downstream route)**.** *Let $G = (R, K, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$ *carrier. Every downstream consumer of the readback row factors through the finite route*

$$D \longmapsto K \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a dyadic tolerance name and a cofinal late-stage witness before it can read any regular rational comparison or Real seal.

Proof. Begin with a downstream consumer of the readback row. Carrier admission exposes Q only as a row in the displayed packet, not as an independent comparison principle.

Replay the continuation row C. The replay first names the requested tolerance by D and then reads the cofinal rate row K for the late stage attached to that tolerance.

The StreamName row W is available only after the cofinal stage is fixed. The RegSeqRat source row R is then read through those finite windows, and the readback row Q records the resulting regular rational comparison.

The Real seal row E and structural rows H, C, P, N are downstream of the same route. Thus no consumer reaches Q or E without first receiving the dyadic tolerance name and cofinal late-stage witness. □

Theorem 1616.4 (Regular Cauchy cofinal-rate Real seal handoff)**.** *Let $G = (R, K, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read of G reaches the seal row E only after replaying the RegSeqRat source, cofinal rate, StreamName windows, DyadicRatCore tolerance ledger, and RegSeqRat readback rows. The exported surface contains no quotient stream, selected limit, host real equality, trichotomy, countable-choice selector, or ambient Real completeness assertion.*

Proof. The Real seal row E is part of the carrier, but its accessibility is ordered by the continuation row C. A Real-facing read must therefore replay the tolerance ledger D, the cofinal rate row K, the finite window row W, the source row R, and the readback row Q before the seal is available.

Componentwise transport through H preserves the same row inventory. It can identify accepted packets along displayed hsame coordinates, but it cannot introduce a quotient stream, selected limit, host equality, trichotomy, choice schedule, or completeness theorem.

The provenance and local naming rows P, N certify the same finite packet. Hence the Real seal is a downstream handoff surface for regular Cauchy cofinal-rate data, not an appeal to ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalRate}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCofinalRate}^{\uparrow}$ packet is generated from a RegSeqRat source row, cofinal CauchyRate row, finite StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalRate}^{\uparrow}}$, seed)

Scope closed: Definition 1616.1, Definition 1616.2, Definition 1616.3, and Definition 1616.4 seed the finite cofinal-rate route from dyadic tolerance requests through StreamName windows, RegSeqRat readback, and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalRate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-first routing, cofinal window exactness, readback stability, Real seal handoff, and ledger refusal proofs for BEDC.Derived.RegularCauchyCofinalRateUp.

Chapter 1617

A Concrete Naming Certificate for $\mathsf{CauchyModulusComposition}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusComposition}^{\uparrow}$ packet records the finite bookkeeping needed to compose two displayed Cauchy modulus budgets before any RegSeqRat or Real consumer reads the resulting route. It sits between the tolerance ledger of chapter 292, the finite-window stream surface of chapter 126, the dyadic rational core of chapter 291, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet names only a finite composition ledger: it is not a quotient stream, not a selected limit, not host real equality, not trichotomy, not ambient completeness, and not a standard bridge assertion.

Definition 1617.1 (Cauchy modulus composition carrier)**.** *A* $\mathsf{CauchyModulusComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M_1, M_2, B, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *M_1 and M_2 are source* $\mathsf{CauchyModulus}^{\uparrow}$ *rows. The first row names the outer precision request, and the second row names the inner stage request read from the first row.*
2. *B is the composed precision-budget row. It records the finite request that first queries M_1, then queries M_2 at the stage exposed by M_1, and exports the resulting stage bound as a single budget.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window route through which the composed budget is allowed to read only displayed stream windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to name every radius comparison that appears during the two modulus lookups.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that receives the composed budget after W and D have been replayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *route row that may be read only after the RegSeqRat handoff has consumed the same composed budget.*
7. *H records componentwise* hsame *transport for M_1, M_2, B, W, D, R, E, the continuation row, the provenance row, and the local naming row.*
8. *C records displayed* Cont *replay from a precision request through M_1, M_2, the composed budget B, stream windows W, dyadic tolerances D, RegSeqRat handoff R, and the Real route E.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusComposition}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or standard bridge data.*

Theorem 1617.2 (Cauchy modulus composition NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusComposition}^{\uparrow}$ *carrier* $K = (M_1, M_2, B, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusCompos}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two source Cauchy modulus rows, composed budget, stream-window route, dyadic tolerance ledger, RegSeqRat handoff, Real route, transport, continuation, provenance, and local naming row;*
2. *budget composition, requiring B to read M_1 before M_2 and to export a single finite stage bound after the two lookups;*
3. *window and tolerance exposure, requiring the composed budget to pass through W and D before any regular rational or Real-facing consumer may read it;*
4. *transport stability, requiring H to preserve the same ordered rows under componentwise* hsame *replacement;*
5. *nonescape, requiring C, P, N to expose no quotient stream, selected limit, host real equality, trichotomy, ambient-completeness principle, or standard bridge claim.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1617.1. The displayed tuple contains exactly the two modulus rows, composed budget, stream-window route, dyadic ledger, RegSeqRat handoff, Real route, transport, continuation, provenance, and local naming row.

Budget composition is forced by the row B. It reads the outer request through M_1, uses the displayed stage from that lookup as the request for M_2, and stores the resulting stage as one finite budget row.

The consumer route is ordered by C. A regular rational or Real-facing read must replay W and D before it can consume R or E, so every exported comparison has passed through finite windows and dyadic tolerance naming.

Transport stability is componentwise. The row H replaces only displayed coordinates by hsame-classified coordinates and then replays the same continuation order.

Since the carrier has no quotient-stream, selected-limit, host-equality, trichotomy, completeness, or bridge coordinate, the listed rows exhaust the local obligation surface. □

Theorem 1617.3 (Cauchy modulus budget composition)**.** *Let $K = (M_1, M_2, B, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusComposition}^{\uparrow}$ *carrier. Every budget-facing consumer of K factors through the finite route*

$$M_1 \longmapsto M_2 \longmapsto B \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a single composed stage bound whose two source modulus lookups and dyadic tolerance rows remain visible.

Proof. Begin with a budget-facing consumer. Carrier admission exposes B only as the row downstream of the two source modulus lookups.

Replay the continuation row C. The first lookup reads M_1 at the requested precision; the second lookup reads M_2 at the stage request displayed by M_1. The row B stores the finite result of these two reads as one budget.

The route then passes through W and D. Thus stream-window access and dyadic tolerance naming are part of the same finite replay rather than external assumptions.

The RegSeqRat and Real rows are downstream consumers of this composed route. Transport and provenance through H, C, P, N preserve the order, so no detached budget can be read by R or E. □

Theorem 1617.4 (Cauchy modulus composition Real route)**.** *Let $K = (M_1, M_2, B, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusComposition}^{\uparrow}$ *carrier. Every Real-facing read of K reaches the* $\mathsf{Real}^{\uparrow}$ *route only after the composed modulus budget has passed through the StreamName windows, DyadicRatCore tolerance ledger, and RegSeqRat handoff. The exported surface contains no quotient stream, selected limit, host real equality, trichotomy, ambient Real completeness assertion, or standard bridge claim.*

Proof. The Real route row E is displayed in the carrier, but its accessibility is ordered by the continuation row C. A Real-facing read must therefore replay M_1, M_2, B, W, D, R before E is available.

The replay keeps the composed budget finite. The two source modulus rows provide the only stage lookups, W supplies the only stream-window route, D supplies the only tolerance ledger, and R supplies the only regular rational handoff.

The rows H, P, N name and transport that same finite packet. They cannot introduce a quotient stream, selected limit, host equality, trichotomy, completeness principle, or standard bridge assertion, because none of those data appear in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusComposition}^{\uparrow}$ packet is generated from two CauchyModulus source rows, a composed precision-budget row, a StreamName window route, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, a RealUp route, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1617.1, Definition 1617.2, Definition 1617.3, and Definition 1617.4 seed the finite ledger that composes two Cauchy modulus budgets before RegSeqRat and Real consumers read them.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, standard bridge data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget composition, window and tolerance exposure, RegSeqRat handoff, Real route nonescape, and transport stability for BEDC.Derived.CauchyModulusCompositionUp.

Chapter 1618

A Concrete Naming Certificate for $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ packet records finite tail-threshold selection for a regular Cauchy name before quotient streams, selected host limits, countable choice, trichotomy, or ambient Real completeness are available. It reads finite windows from chapter 126, regular rational observations from chapter 301, dyadic tolerance data from chapter 291, admissible index schedules from chapter 1568, explicit thresholds from chapter 292, and the downstream seal of chapter 38. The packet is a finite NameCert surface for choosing a displayed tail threshold inside a regular Cauchy route; it is not a quotient of streams and not a completed real-number object.

Definition 1618.1 (Cauchy tail selection criterion carrier)**.** *A* $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, M, D, I, Q, E, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose displayed observations are the only stream data admitted by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S and carrying the regular rational approximants inspected by the criterion.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row that names the finite stage bound for each displayed tolerance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through which every radius, endpoint, and comparison used by the tail criterion is named.*
5. *I is the* $\mathsf{CauchyReindexing}^{\uparrow}$ *admissible-schedule row, recording the monotone cofinal route along which the selected tail may be inspected.*
6. *Q is the cofinal tail request row, binding the requested precision to the displayed threshold and to the admissible schedule.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *boundary row, available only after S, R, M, D, I, Q have been replayed.*
8. *H records componentwise* hsame *transport for the stream windows, readback, modulus, dyadic tolerance ledger, admissible schedule, cofinal tail request, Real boundary, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a tail-selection request through $S, R, M, D, I, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailSelectionCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, hidden infinite stream inspection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1618.2 (Cauchy tail selection criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ *carrier $T = (S, R, M, D, I, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailSelectionCriterion}}$ *surface consists exactly of carrier admission for the eleven displayed rows, finite-window exposure through S, regular rational readback through R, threshold exposure through M, dyadic tolerance exposure through D, admissible-schedule exposure through I, cofinal tail-request binding through Q, Real-boundary ordering through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 1618.1. The visible analytic rows are precisely the stream windows, regular rational readback, CauchyModulus threshold, DyadicRatCore tolerance ledger, CauchyReindexing schedule, cofinal tail request, and Real boundary.

Read the rows in dependency order. A tail-selection request first reaches the finite windows S and their regular rational readback R, then consumes the threshold row M and dyadic tolerance ledger D that make the requested tail bound visible.

The schedule and request rows then close the local criterion. The admissible route I records how the selected tail may be inspected, while Q binds the cofinal request to the displayed threshold and schedule.

The rows E, H, C, P, N add only boundary, transport, replay, provenance, and local naming for the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, hidden-inspection, or ambient-completeness coordinate, the displayed rows exhaust the local NameCert obligation surface. □

Theorem 1618.3 (Cauchy tail selection criterion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ *carrier factors through*

$$S \longmapsto R \longmapsto M \longmapsto D \longmapsto I \longmapsto Q \longmapsto (H, C, P, N).$$

Thus a regular rational consumer receives the displayed stream window, the matching readback row, the explicit modulus threshold, the dyadic tolerance ledger, the admissible reindexing schedule, and the cofinal tail request before any Real-facing boundary can be inspected.

Proof. Start from a regular rational consumer request. By Definition 1618.1, the readback row R is attached to the finite stream windows S, so the consumer cannot begin from an unbounded stream or a quotient representative.

The threshold and tolerance rows are consumed next. The row M provides the displayed stage bound for the requested precision, while D records the dyadic endpoint and radius comparisons used by that bound.

The admissible schedule I and cofinal request Q then bind the same threshold to a tail route. Transport, replay, provenance, and local naming through H, C, P, N preserve that order, so no detached RegSeqRat handoff exists. □

Theorem 1618.4 (Cauchy tail selection criterion Real boundary nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ *carrier reaches the Real boundary row only through the complete finite route*

$$S, R, M, D, I, Q, E, H, C, P, N.$$

Consequently the packet cannot export quotient stream equality, a selected host limit, a countable-choice selector, a trichotomy branch, hidden infinite stream inspection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real-facing read. The carrier exposes the boundary only as the row E, and E is listed after the stream, readback, threshold, dyadic, admissible-schedule, and cofinal-request rows in Definition 1618.1.

Replay the continuation row C. The read therefore moves through S, R, M, D, I, Q before it can reach E, with H transporting only the displayed coordinates and P, N packaging and naming the same finite packet.

No row in the carrier stores quotient stream equality, a host-selected limit, a choice function, a trichotomy branch, an infinite-stream oracle, or an ambient completeness theorem. The Real boundary is therefore a downstream seal for the finite tail-selection criterion, not an escape from it. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailSelectionCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailSelectionCriterion}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a CauchyModulus threshold row, a DyadicRatCore tolerance ledger, a CauchyReindexing admissible schedule, a cofinal tail request, a RealUp boundary, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailSelectionCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 1618.1, Definition 1618.2, Definition 1618.3, and Definition 1618.4 seed the finite tail-selection route from StreamName windows through RegSeqRat readback, CauchyModulus thresholds, DyadicRatCore tolerances, CauchyReindexing schedules, cofinal tail requests, and the Real boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailSelectionCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, hidden infinite stream inspection, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, NameCert obligation coverage, RegSeqRat handoff, Real boundary nonescape, and ledger refusal proofs for BEDC.Derived.CauchyTailSelectionCriterionUp.

Chapter 1619

A Concrete Naming Certificate for $\mathsf{RealZero}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealZero}^{\uparrow}$ packet records the terminal zero real not as a host real equality claim but as the finite zero-source route already displayed by chapter 364, chapter 624, chapter 126, chapter 301, chapter 291, and chapter 38. The chapter packages that route as the public zero-real seal consumed by downstream positive-real, absolute-value, order, metric, and completion-facing certificates.

Definition 1619.1 (RealZero carrier)**.** *A* $\mathsf{RealZero}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (q, S, Z_0, D, R, H, C, P, N)$$

with the following displayed coordinates:

1. *q is the* $\mathsf{Rat}^{\uparrow}$ *zero seed row.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *window whose every admitted finite request reads q.*
3. *Z_0 is the* $\mathsf{RegularCauchyZero}^{\uparrow}$ *route, including the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *zero row and the zero regular-Cauchy source handoff.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger recording the endpoint, tolerance, and radius data used by the zero-source route.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the constant window, regular-rational readback, and dyadic zero ledger have been displayed.*
6. *H stores componentwise* hsame *transport for the rational seed, constant window, regular-Cauchy zero route, dyadic ledger, terminal seal, provenance, and local name.*
7. *C stores displayed* Cont *replay routes from q through S, from S through Z_0 and D, and from those rows into the terminal seal R.*
8. *P records* Pkg *provenance over* $\mathsf{RealZero}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{StationaryRationalCauchy}^{\uparrow}$, $\mathsf{RegularCauchyZero}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host zero equality, selected limits, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.RealZeroUp.RealZeroCarrier`

Theorem 1619.2 (RealZero NameCert obligations)**.** *For every accepted* $\mathsf{RealZero}^{\uparrow}$ *carrier* $Z = (q, S, Z_0, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the rational zero seed, constant StreamName window, RegularCauchyZero route, DyadicRatCore zero ledger, terminal RealUp seal, transport, replay, provenance, and local naming rows;*
2. *stationary zero-source route, requiring the rational seed to be read only through the constant window, the stationary rational Cauchy row, and the RegularCauchyZero handoff;*
3. *terminal RealUp seal handoff, requiring the zero real-facing row* R *to be exported only after the StreamName, RegSeqRat, DyadicRatCore, and RegularCauchyZero rows have been displayed;*
4. *componentwise transport and replay stability, requiring accepted zero-real packets to move through* H *and replay through* C *without changing the displayed zero-source chain;*
5. *ledger nonescape from quotient streams, host zero equality, selected limits, trichotomy, and ambient completeness.*

No NameCert obligation is stored outside these five finite rows.

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1619.1. The only visible data are the zero seed q, the constant window S, the regular-Cauchy zero route Z_0, the dyadic zero ledger D, the terminal seal R, and the structural rows H, C, P, N.

The stationary source row is forced by the sibling route. The packet reads q through the constant $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{StationaryRationalCauchy}^{\uparrow}$ surface, and the $\mathsf{RegularCauchyZero}^{\uparrow}$ route before any regular-rational or real-facing consumer can inspect the zero source.

The terminal seal is then read through R. Transport and replay are componentwise because H and C are the only rows that move or re-open the chain.

The carrier has no coordinate for quotient stream equality, host zero equality, selected limits, trichotomy, or ambient completeness. Thus the ledger cannot export any of those principles through the local $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ row. □

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroNameCertObligations`

Theorem 1619.3 (RealZero terminal seal handoff)**.** *Every accepted* $\mathsf{RealZero}^{\uparrow}$ *consumer must read the zero seed through the constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any downstream* $\mathsf{PositiveReal}^{\uparrow}$ *or* $\mathsf{RealAbs}^{\uparrow}$ *consumer may use it. Equivalently, the public zero-real boundary is the finite route*

$$q \longrightarrow S \longrightarrow Z_0 \longrightarrow D \longrightarrow R$$

together with H, C, P, N*, and no consumer may replace that route by host zero equality, quotient real equality, selected limits, trichotomy, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier terminal seal handoff`

Proof. Start with an accepted carrier $Z = (q, S, Z_0, D, R, H, C, P, N)$. The seed row q is not exported directly; it is first read through the constant window S and the regular-rational zero row supplied by Z_0.

Next the dyadic zero ledger D records the finite endpoint and tolerance data attached to that same route. The real-facing seal R is reached only after those rows are visible.

Downstream $\mathsf{PositiveReal}^{\uparrow}$ and $\mathsf{RealAbs}^{\uparrow}$ consumers therefore receive the zero source as the displayed terminal seal handoff, not as a host value. Since the carrier contains no excluded principle, the handoff cannot provide quotient real equality, selected limits, trichotomy, or ambient completeness. □

Theorem 1619.4 (RealZero stationary source obligation)**.** *Every accepted* $\mathsf{RealZero}^{\uparrow}$ *packet* $Z = (q, S, Z_0, D, R, H, C, P, N)$ *displays a stationary zero-source obligation: the rational seed* q *is read through the constant* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* BHist *rows of* Z_0*, the componentwise* hsame *support in* H*, the* Cont *replay support in* C*, the* Pkg *provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row* N *before any downstream consumer sees the terminal zero-real seal.*

This obligation exports no host zero equality, quotient stream equality, selected tail, trichotomy, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier stationary source obligation`

Proof. Project the carrier coordinates from Definition 1619.1. The only source row for the zero value is the rational seed q, and the only public route from q to a stream-facing object is the constant $\mathsf{StreamName}^{\uparrow}$ window S together with the zero route Z_0.

Now apply the terminal handoff of Definition 1619.3. It forces every consumer of the terminal $\mathsf{Real}^{\uparrow}$ seal to read the seed through the displayed StreamName, RegSeqRat, DyadicRatCore, and RealUp rows.

The structural rows H, C, P, N are part of the same carrier. Thus hsame transport, Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row are visible at the stationary source boundary, while excluded host and quotient principles have no coordinate in the packet. □

Theorem 1619.5 (RealZero constant-window uniqueness)**.** *Let* $Z = (q, S, Z_0, D, R, H, C, P, N)$ *and* $Z' = (q', S', Z_0', D', R', H', C', P', N')$ *be accepted* $\mathsf{RealZero}^{\uparrow}$ *packets. If the rational zero seed rows are the same displayed* $\mathsf{Rat}^{\uparrow}$ *zero row and the constant* $\mathsf{StreamName}^{\uparrow}$ *windows are related componentwise by the displayed* hsame *support, then every terminal* $\mathsf{Real}^{\uparrow}$ *seal read by a downstream consumer follows the same route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row.*

Thus constant-window agreement determines the terminal zero-real seal route as a finite packet route, not as host zero equality, quotient real equality, selected-limit equality, trichotomy, or ambient completeness.

Proof. Start with the carrier definition, Definition 1619.1. The only source row is the rational zero seed q, and the only stream-facing source is the constant window S.

Use the stationary-source obligation of Definition 1619.4. It forces every downstream consumer to read the zero seed through S, the rows of Z_0, the structural support H, C, P, N, and the local $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ row before the terminal seal is seen.

Apply the terminal seal handoff of Definition 1619.3. It fixes the public route through the constant window, the $\mathsf{RegSeqRat}^{\uparrow}$ zero row inside Z_0, the $\mathsf{DyadicRatCore}^{\uparrow}$ zero ledger D, and the terminal $\mathsf{Real}^{\uparrow}$ seal R.

Finally read the dyadic ledger obligation of Definition 1619.6. Since the two packets share the displayed zero seed and the constant-window rows componentwise, the downstream seal route must pass through the same RegSeqRat, DyadicRatCore, and local NameCert rows. The carrier has no coordinate for host equality, quotient equality, selected limits, trichotomy, or ambient completeness, so the uniqueness is exactly the displayed finite route. □

Theorem 1619.6 (RealZero dyadic ledger obligation)**.** *Every accepted* $\mathsf{RealZero}^{\uparrow}$ *packet* $Z = (q, S, Z_0, D, R, H, C, P, N)$ *displays a dyadic zero-ledger obligation: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *is the unique public tolerance ledger between the stationary source route* Z_0 *and the downstream* $\mathsf{Real}^{\uparrow}$ *seal* R*, and its* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support is read before any consumer may use the seal.*

The ledger is finite and does not introduce quotient real equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier dyadic ledger obligation`

Proof. Read D from the carrier in Definition 1619.1. Its role is fixed there as the zero ledger recording endpoint, tolerance, and radius data for the zero-source route.

Use Definition 1619.3 to locate D in the public route

$$q \longrightarrow S \longrightarrow Z_0 \longrightarrow D \longrightarrow R.$$

The downstream seal cannot be inspected before this DyadicRatCore row is displayed.

The same theorem keeps the route inside the finite StreamName, RegSeqRat, DyadicRatCore, and RealUp rows. Therefore the BHist ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert support are the whole dyadic obligation surface, with no coordinate for the excluded principles. □

Theorem 1619.7 (RealZero downstream consumer obligation)**.** *Every downstream consumer of an accepted* $\mathsf{RealZero}^{\uparrow}$ *packet must consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *only after the constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, the* BHist *display rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row have been displayed.*

Consequently, downstream certificates inherit a finite zero-real seal handoff and not a host real value, quotient real equality, selected limit, trichotomy, or ambient completeness premise.

Proof. Start with a downstream consumer and unfold the accepted packet by Definition 1619.1. The consumer can see the seal only through the carrier coordinate R and the visible support rows H, C, P, N.

Apply Definition 1619.3. It requires the route through S, the RegSeqRat zero row inside Z_0, the DyadicRatCore ledger D, and the terminal RealUp seal R before any downstream PositiveReal, RealAbs, order, metric, or completion-facing certificate may use the zero-real boundary.

Because the carrier has no coordinate for host equality, quotient equality, selected limits, trichotomy, or ambient completeness, those objects cannot be inherited by the downstream consumer. What is inherited is exactly the displayed finite handoff through BHist, hsame, Cont, Pkg, and NameCert support. □

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier downstream consumer obligation`

Theorem 1619.8 (RealZero positive-real and absolute-value handoff)**.** *Every* $\mathsf{PositiveReal}^{\uparrow}$ *or* $\mathsf{RealAbs}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealZero}^{\uparrow}$ *packet may use the zero seal only through the displayed finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, the consumer must first read the constant $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback supplied by the regular-Cauchy zero route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, the terminal*

$\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row.*

The handoff does not export host zero equality, quotient real equality, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness to either downstream chapter.*

Proof. Start with the terminal seal handoff of Definition 1619.3. It forces every zero-real read through the constant StreamName window, the RegSeqRat zero row inside the regular-Cauchy zero route, the DyadicRatCore zero ledger, and the terminal RealUp seal.

The stationary source obligation of Definition 1619.4 identifies the source side of that route. The rational zero seed is visible only through the constant window, the regular-rational zero readback, componentwise transport, continuation replay, package provenance, and the local NameCert row.

The dyadic ledger obligation of Definition 1619.6 supplies the tolerance-facing row consumed by order, metric, positive-real, and absolute-value chapters. It places the DyadicRatCore row between the stationary source route and the terminal RealUp seal, with no extra coordinate for quotient or selected-representative data.

Finally apply the downstream consumer obligation of Definition 1619.7. It states that downstream consumers inherit exactly the displayed finite handoff through BHist, hsame, Cont, Pkg, and NameCert support. Thus $\mathsf{PositiveReal}^{\uparrow}$ and $\mathsf{RealAbs}^{\uparrow}$ may consume the zero seal only through the listed route. □

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier positive abs handoff`

Theorem 1619.9 (RealZero zero-distance seal)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer using the zero real as a zero-distance or zero-radius boundary must first read the constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row of the accepted* $\mathsf{RealZero}^{\uparrow}$ *packet.*

The zero-distance seal is therefore the displayed finite route of the packet. It is not host zero equality, quotient real equality, selected-limit equality, trichotomy, or ambient completion.

Proof. First apply the terminal seal handoff, Definition 1619.3. It places the constant window, RegSeqRat zero row, DyadicRatCore zero ledger, and terminal RealUp seal upstream of any downstream read.

Next apply the dyadic ledger obligation, Definition 1619.6. The metric-facing zero-radius boundary must pass through the displayed tolerance ledger D rather than through a host equality or selected representative.

Then apply the downstream consumer obligation, Definition 1619.7. It covers the MetricSpace, QuasiMetric, Completion, UniformCompletion, and CompleteMetric consumers by requiring the same BHist, hsame, Cont, Pkg, and local NameCert support before the zero seal is used.

Finally use componentwise transport, Definition 1619.10. Transport preserves the finite zero-source route component by component, so the seal remains a BEDC packet boundary and does not become quotient real equality, trichotomy, selected-limit equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroCarrier zero distance seal`

Theorem 1619.10 (RealZero componentwise transport)**.** *Let* $Z = (q, S, Z_0, D, R, H, C, P, N)$ *and* $Z' = (q', S', Z'_0, D', R', H', C', P', N')$ *be accepted* $\mathsf{RealZero}^{\uparrow}$ *packets. If the displayed* hsame *rows in* H *identify the rational seed, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback inside* Z_0*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *support, replay, provenance, and local*

naming rows component by component, then every downstream read of Z' follows the same zero-source route as a read of Z.

The transported route remains a finite BEDC packet route; it does not become host real equality, quotient stream equality, selected-limit equality, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealZeroUp.RealZeroComponentwiseTransport`

Proof. Project the carrier rows with Definition 1619.1. The row H is the only coordinate that moves accepted zero-real packets, and it is specified componentwise for the seed, window, regular-Cauchy zero route, dyadic ledger, terminal seal, provenance, and local name.

Apply the terminal seal handoff of Definition 1619.3. It fixes the order

$$q \longrightarrow S \longrightarrow Z_0 \longrightarrow D \longrightarrow R$$

before any downstream consumer can inspect the zero-real boundary.

The stationary source obligation of Definition 1619.4 and the dyadic ledger obligation of Definition 1619.6 keep the transported seed and tolerance rows inside the same BHist, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. Since the carrier has no excluded coordinate, componentwise transport preserves the displayed packet without adding host or quotient principles. □

Theorem 1619.11 (RealZero replay stability)**.** *Every accepted* $\mathsf{RealZero}^{\uparrow}$ *replay through the* Cont *row C repeats the same stationary zero-source route through the constant* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback supplied by Z_0, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R. Replaying the packet cannot expose a shorter path to the zero-real seal and cannot introduce quotient streams, host zero equality, selected limits, trichotomy, or ambient completeness.*

Proof. Read C from Definition 1619.1. It records displayed Cont replay from q through S, from S through Z_0 and D, and from those rows into the terminal seal R.

Use Definition 1619.2 to identify replay stability as part of the local NameCert surface. The obligations do not permit a replay row outside C, and the terminal seal handoff forces every replayed read through the same finite route.

Finally apply Definition 1619.7. Downstream consumers inherit the displayed zero-real seal handoff through BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ support, so replay preserves the packet route rather than supplying an external equality or completeness principle. □

Theorem 1619.12 (RealZero ledger nonescape)**.** *The* $\mathsf{RealZero}^{\uparrow}$ *ledger rows in an accepted packet $Z = (q, S, Z_0, D, R, H, C, P, N)$ are exhausted by the stationary source, constant* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *zero readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *display, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealZero}^{\uparrow}}$ *row. No ledger read can escape from these rows to quotient stream equality, host zero equality, selected limits, trichotomy, countable choice, or ambient Real completeness.*

Proof. Carrier admission in Definition 1619.1 lists every ledger-bearing row. The finite packet contains $q, S, Z_0, D, R, H, C, P, N$ and no additional coordinate.

The dyadic ledger obligation of Definition 1619.6 places D between the stationary source route and the terminal RealUp seal. The downstream consumer obligation of Definition 1619.7 then requires every consumer to read the same BHist, transport, replay, provenance, and local naming support.

Since Definition 1619.2 explicitly excludes rows for quotient streams, host zero equality, selected limits, trichotomy, and ambient completeness, the ledger has no path to those principles. Countable

choice is likewise absent because the packet has only the displayed finite rows and no selector coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealZero}\uparrow}$**.**

Constructive story: A $\mathsf{RealZero}^{\uparrow}$ packet is generated from the rational zero seed, a constant StreamName window, a RegularCauchyZero route, a DyadicRatCore zero ledger, a terminal RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealZero}\uparrow}$, seed)

Scope closed: Definition 1619.1, Definition 1619.2, and Definition 1619.3 seed the finite zero-real seal route used by downstream PositiveReal and RealAbs consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealZero}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completeness, quotient real equality, host equality, trichotomy, selected limits, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stationary zero-source routing, terminal RealUp seal handoff, componentwise transport and replay stability, and ledger nonescape for the same finite packet.

Chapter 1620

A Concrete Naming Certificate for $\mathsf{RealClassifier}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealClassifier}^{\uparrow}$ packet records the public classifier surface of $\mathsf{Real}^{\uparrow}$ as a finite NameCert construction rather than as host equality of completed real numbers. It consumes the terminal real-name seal of chapter 38, the regular rational readback surface of chapter 301, the common finite-window discipline of chapter 126, the dyadic tolerance ledgers of chapter 291, the equality-inspection packet of chapter 1517, and the regular-name equivalence boundary of chapter 1404. The chapter gives downstream metric, algebraic, and order-facing certificates one classifier boundary for real names while keeping equality as displayed classifier data.

Definition 1620.1 (RealClassifier carrier)**.** *A* $\mathsf{RealClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$$

with the following displayed rows:

1. *X and Y are the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows whose public classifier relation is being exposed.*
2. *S_X and S_Y are the regular rational source rows read back from the two seals.*
3. *R_X and R_Y are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback rows.*
4. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *finite window through which both readbacks are inspected.*
5. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the finite precision, radius, and comparison budget used by the classifier read.*
6. *C is the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row comparing the two regular names through W and D.*
7. *E is the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row that records the regular-Cauchy equivalence boundary visible before the terminal real seals are consumed.*
8. *H stores componentwise* hsame *transports for the two seals, readbacks, common window, dyadic ledger, classifier rows, provenance, and local name.*
9. *K stores displayed* Cont *replay routes from X, Y through $S_X, S_Y, R_X, R_Y, W, D, C, E$ and back to the public seal boundary.*
10. *P records* Pkg *provenance over* $\mathsf{RealClassifier}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealNameClassifier}^{\uparrow}$, $\mathsf{RegularCauchyEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^\uparrow}$ *row.*

The packet has no coordinate for quotient equality of streams, host real equality, selected limits, trichotomy, countable choice, or ambient completeness.

Theorem 1620.2 (RealClassifier NameCert obligations)**.** *For every accepted* $\mathsf{RealClassifier}^\uparrow$ *carrier* $Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^\uparrow}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the two* $\mathsf{Real}^\uparrow$ *seals, regular rational readbacks, common window, dyadic tolerance ledger, classifier rows, transport, replay, provenance, and local name;*
2. *tolerance classifier stability, requiring every classifier read to pass through the common window W, the dyadic ledger D, and the* $\mathsf{RealNameClassifier}^\uparrow$ *row C;*
3. $\mathsf{RegSeqRat}^\uparrow$ *handoff, requiring every regular-rational consumer to factor through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before reaching the public classifier boundary;*
4. *real seal non-escape, requiring X and Y to be consumed only as displayed terminal seals rather than as host real values or quotient classes;*
5. *ledger exhaustion, requiring H, K, P, N to transport, replay, source, and name only the displayed finite packet.*

No local NameCert obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 1620.1. The accepted packet exposes exactly the two seal rows, the two regular rational source rows, the two $\mathsf{RegSeqRat}^\uparrow$ readbacks, the common window, the dyadic ledger, the two classifier rows, and the structural rows H, K, P, N.

Tolerance classifier stability is the route through W, D, C. Since W is the only common inspection schedule and D is the only tolerance ledger, a classifier read cannot inspect either side independently of those rows.

The regular-rational handoff uses S_X, S_Y, R_X, R_Y before it reaches C and E. The real seal row is then read only after that finite handoff has been displayed.

The remaining rows transport, replay, source, and locally name the same packet. Because the carrier omits quotient stream equality, host real equality, selected limits, trichotomy, countable choice, and ambient completeness, no NameCert obligation can export them. □

Lean4 ✓ `BEDC.Derived.RealClassifierCarrier namecert obligations`

Theorem 1620.3 (RealClassifier RegSeqRat window exhaustion)**.** *For every accepted* $\mathsf{RealClassifier}^\uparrow$ *carrier*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

every accepted public classifier comparison reaches the public $\mathsf{RealClassifier}^\uparrow$ *boundary only through the finite* $\mathsf{StreamName}^\uparrow$ *window W, the* $\mathsf{RegSeqRat}^\uparrow$ *readback rows R_X, R_Y, the* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledger D, the terminal* $\mathsf{Real}^\uparrow$ *seal rows X, Y, and the structural rows H, K, P, N. The route has no coordinate for quotient stream equality, trichotomy, selected representatives, or ambient Real completeness.*

Proof. Project the carrier of Definition 1620.1. Its public comparison rows are the two terminal Real seals, their source and RegSeqRat readbacks, the common StreamName window, the dyadic tolerance ledger, the classifier row, the regular-Cauchy boundary row, and the structural rows.

Use the NameCert obligations of Definition 1620.2. They force carrier admission, tolerance classifier stability, RegSeqRat handoff, Real seal non-escape, and ledger exhaustion to read through $S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$ before the public classifier boundary is exposed.

Now route the comparison through the existing handoff and seal the read. The RegSeqRat handoff of Definition 1620.5 supplies the regular-rational readback before the classifier row is consumed, while the real seal non-escape theorem Definition 1620.8 keeps X and Y as displayed terminal seals rather than host real values.

Finally apply ledger exhaustion from Definition 1620.9. The rows H, K, P, N transport, replay, source, and name only the displayed packet, so no quotient stream equality, trichotomy, selected representative, or ambient Real-completeness coordinate can be exported by the accepted comparison. □

Lean4 ✓ `BEDC.Derived.RealClassifierRegSeqRatWindowExhaustion`

Theorem 1620.4 (RealClassifier finite-window exactness target)**.** *For every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the public classifier relation is determined exactly by the common $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row* C*, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row* E*, and the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_X, R_Y*. Any consumer asking for host real equality, quotient stream equality, trichotomy, countable choice, or ambient completeness has no row in the carrier.*

Proof. Project the carrier rows from Definition 1620.1. The only public comparison data are the common finite window W, dyadic ledger D, classifier row C, regular-Cauchy boundary E, and regular rational readbacks R_X, R_Y over the terminal seals X, Y.

Use the NameCert obligations of Definition 1620.2. They force every accepted classifier read through W, D, C, E, R_X, R_Y before the public boundary is reached, while H, K, P, N only transport, replay, source, and locally name the same packet.

Finally inspect the absent coordinates of the carrier. Since no row stores host real equality, quotient stream equality, trichotomy, countable choice, or ambient completeness, no accepted consumer can extract one of those objects from the public classifier relation. □

Lean4 ✓ `BEDC.Derived.RealClassifierUp.RealClassifierCarrier finite window exactness target`

Theorem 1620.5 (RealClassifier RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing classifier read of an accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier factors through the displayed regular readbacks* S_X, S_Y, R_X, R_Y*, the common window* W*, the dyadic tolerance ledger* D*, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row* C*, and the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row* E*. Thus the public real-classifier read is a finite route through regular rational data, not a direct equality test on terminal real seals.*

Proof. Start with an accepted carrier $Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$. The two seal rows X and Y expose regular rational readbacks only through S_X, S_Y, R_X, R_Y.

The common window W supplies the shared finite requests, and the ledger D records the dyadic tolerance data for those requests. The classifier row C compares the two regular names through exactly those displayed rows.

The row E records the regular-Cauchy equivalence boundary before any real-facing consumer reaches the terminal seals. Hence every $\mathsf{RegSeqRat}^{\uparrow}$-facing read factors through the displayed finite chain and cannot be replaced by a host equality test on X and Y. □

Lean4 ✓ `BEDC.Derived.RealClassifierRegSeqRatHandoff`

Theorem 1620.6 (RealClassifier regular-real tail-equivalence readback)**.** *Every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *packet comparing two* $\mathsf{Real}^{\uparrow}$ *seals through*

$$S_X, S_Y, R_X, R_Y, W, D, C, E$$

factors any same-real read through the $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *finite tail route before it reaches the public classifier boundary. The readback therefore consumes the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, common* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RealNameClassifier}^{\uparrow}$ *comparison row, and visible* $\mathsf{Real}^{\uparrow}$ *seal rows as tail-equivalence data, not as host equality of completed reals.*

Proof. Apply the RegSeqRat handoff theorem Definition 1620.5. It forces every same-real read to expose $S_X, S_Y, R_X, R_Y, W, D, C, E$ before the public classifier boundary is reached.

Use the finite same-real tail route of chapter 5940. Its carrier reads two regular rational sources through one finite window and one dyadic tolerance ledger before the Real-facing seal is consumed.

Project the shared rows of the $\mathsf{RealClassifier}^{\uparrow}$ carrier. The rows R_X, R_Y supply the two regular rational readbacks, W supplies the common inspected window, D supplies the tolerance ledger, C records the name comparison, and E records the visible regular-Cauchy boundary before X, Y are read as terminal seals.

The structural rows H, K, P, N only transport, replay, source, and name that same finite packet. Hence a same-real read cannot bypass the regular-real tail-equivalence route by appealing to quotient streams, host real equality, selected limits, countable choice, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealClassifierUp.RealClassifierRegularRealTailEquivalenceReadback`

Theorem 1620.7 (RealClassifier regular-sequence induction)**.** *Let*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$$

be an accepted $\mathsf{RealClassifier}^{\uparrow}$ *carrier. Every classifier read whose* $\mathsf{StreamName}^{\uparrow}$ *requests lie in the common finite window* W *is consumed by induction over the displayed request rows of* W*: the base read uses the first dyadic tolerance row through* D*, compares the two regular names through* C*, and records the visible regular-Cauchy boundary through* E*; the step appends the next inspected request in* W *and replays the same finite packet through* H, K, P, N*. The induction exports only the rows of Definition 1620.1 and the obligations of Definition 1620.2; it does not export quotient stream equality, host real equality, trichotomy, countable choice, or ambient completeness.*

Proof. Apply the RegSeqRat handoff theorem Definition 1620.5. It forces every regular-rational classifier read to pass through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before the public classifier boundary is reached.

For the base request, read the first displayed row of the finite window W. The dyadic tolerance information is the row D, the comparison of the two regular names is the row C, and the regular-Cauchy boundary visible before the terminal seals is the row E.

For the induction step, append one further inspected request from W. The NameCert obligations of Definition 1620.2 keep the appended request inside the same carrier, while H transports the displayed rows, K replays the continuation route, P records provenance, and N locally names the same packet.

The carrier definition Definition 1620.1 has no coordinate for quotient stream equality, host real equality, trichotomy, countable choice, or ambient completeness. Thus the induction is a finite read over the common StreamName window and cannot be used as any of those ambient principles. □

Lean4 ✓ `BEDC.Derived.RealClassifierRegularSequenceInduction`

Theorem 1620.8 (RealClassifier real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier sees only the finite rows* $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$*. The consumer obtains no quotient equality of streams, host real equality, selected limit, trichotomy witness, countable-choice schedule, or ambient completeness principle from the classifier packet.*

Proof. Read the carrier of Definition 1620.1. The only real-facing rows are the displayed seals X and Y, and their only exposed classifier route is the finite chain through the regular rational readbacks, common window, dyadic ledger, and classifier rows.

The structural rows H, K, P, N do not add new analytic data. They identify, replay, source, and name the same finite packet.

Since no coordinate carries quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness, a consumer of the public $\mathsf{RealClassifier}^{\uparrow}$ boundary cannot extract any of those principles. □

Lean4 ✓ `BEDC.Derived.RealClassifierRealSealNonescape`

Theorem 1620.9 (RealClassifier ledger exhaustion)**.** *For every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

all public classifier reads are exhausted by the displayed structural rows H, K, P, N *together with the finite analytic rows* $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E$*. In particular, no public read can export quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness.*

Proof. Project the carrier rows of Definition 1620.1. The analytic access rows are exactly the two real seals, their regular rational sources and readbacks, the common stream window, the dyadic tolerance ledger, the classifier row, and the regular-Cauchy equivalence boundary.

Apply the NameCert obligations theorem Definition 1620.2. Its fifth row states that H, K, P, N transport, replay, source, and locally name only the displayed finite packet, so there is no hidden classifier read outside the carrier.

Finally combine the RegSeqRat handoff of Definition 1620.5 with the real seal non-escape theorem Definition 1620.8. The handoff forces every regular-rational read through $S_X, S_Y, R_X, R_Y, W, D, C, E$, and seal non-escape rules out the listed quotient, host-equality, choice, order, and completeness endpoints. □

Lean4 ✓ `BEDC.Derived.RealClassifierLedgerExhaustion`

Theorem 1620.10 (RealClassifier tolerance-window determinacy)**.** *Let*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$$

and

$$Q' = (X', Y', S'_X, S'_Y, R'_X, R'_Y, W', D, C', E', H', K', P', N')$$

be accepted $\mathsf{RealClassifier}^{\uparrow}$ *carriers. Suppose the two packets share the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, expose the same* $\mathsf{StreamName}^{\uparrow}$ *window through* W *and* W'*, use the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff surface through* S_X, S_Y, R_X, R_Y *and* S'_X, S'_Y, R'_X, R'_Y*, reach the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary through* X, Y *and* X', Y'*, and transport the displayed rows through the same* hsame*,* Cont*,* Pkg*, and local* NameCert *routes carried by* H, K, P, N *and* H', K', P', N'*. Then* Q *and* Q' *expose*

the same public classifier decision surface: every accepted classifier read of either packet is the finite route through the common tolerance ledger, common window, regular rational readback, real seal boundary, transport, replay, provenance, and local naming rows.

Proof. Apply the RegSeqRat handoff theorem Definition 1620.5 to both carriers. The hypotheses identify the tolerance ledger, finite window, and regular rational readback surfaces, so each classifier read reaches the public boundary through the same displayed D, W, S_X, S_Y, R_X, R_Y route.

Next apply the real seal non-escape theorem Definition 1620.8. It rules out quotient stream equality, host real equality, selected limits, trichotomy, countable choice, and ambient completeness as alternative endpoints, leaving only the displayed terminal seal boundary named in the hypotheses.

Finally apply ledger exhaustion from Definition 1620.9. The structural rows H, K, P, N and H', K', P', N' can only transport, replay, source, and locally name the displayed packet, so two accepted packets with the same tolerance, window, handoff, seal, transport, replay, provenance, and local naming routes expose the same classifier decision surface. □

Lean4 ✓ `BEDC.Derived.RealClassifierToleranceWindowDeterminacy`

Theorem 1620.11 (RealClassifier window scope)**.** *Every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *public read factors through the common* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *before a* $\mathsf{RealNameClassifier}^{\uparrow}$ *row or a* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row can be consumed. In carrier notation, for*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the public classifier boundary is the finite route through $S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$ *over the two terminal* $\mathsf{Real}^{\uparrow}$ *seals* X, Y*, not a direct equality test on completed real names.*

Proof. Project the carrier of Definition 1620.1. The only common inspection coordinate is W, and the only tolerance and comparison-budget coordinate is D.

Apply the RegSeqRat handoff theorem Definition 1620.5. It forces every regular-rational classifier read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before the public classifier boundary is reached.

Use real seal non-escape from Definition 1620.8. Since the terminal seals X, Y do not expose host real equality, quotient stream equality, selected limits, trichotomy, countable choice, or ambient completeness, the read cannot bypass the common window and ledger by inspecting the seals directly.

Finally apply ledger exhaustion from Definition 1620.9. The structural rows H, K, P, N only transport, replay, source, and name the displayed finite packet, so every accepted public read consumes W and D before it reaches C or E. □

Lean4 ✓ `BEDC.Derived.RealClassifierWindowScope`

Theorem 1620.12 (RealClassifier common-window quotient refusal)**.** *For accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carriers, a public equality or classifier read is admissible only when both terminal* $\mathsf{Real}^{\uparrow}$ *seals are compared through the same* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*. Any attempted read that identifies the terminal* $\mathsf{Real}^{\uparrow}$ *seals directly, without displaying both* W *and* D*, leaves the carrier surface and cannot be cited as* $\mathsf{RealClassifier}^{\uparrow}$ *evidence.*

Proof. Project an accepted carrier

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N).$$

The carrier exposes the two terminal seals only together with their regular rational source rows, readbacks, common window, tolerance ledger, classifier row, equivalence boundary, and structural rows.

Apply the RegSeqRat handoff of Definition 1620.5. It forces every public classifier read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before the terminal boundary is consumed.

Apply real seal non-escape from Definition 1620.8. It rules out a direct host equality or quotient equality read on X, Y as an endpoint supplied by the carrier.

Use ledger exhaustion from Definition 1620.9 and window scope from Definition 1620.11. These keep the structural rows H, K, P, N tied to the same finite packet and require the common window W and ledger D before any $\mathsf{RealNameClassifier}^{\uparrow}$ or $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ row is consumed. A read that omits either row is therefore outside the accepted $\mathsf{RealClassifier}^{\uparrow}$ evidence surface. □

Lean4 ✓ `BEDC.Derived.RealClassifierCommonWindowQuotientRefusal`

Theorem 1620.13 (RealClassifier source scope)**.** *Every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *public read factors through exactly the source scopes named by the carrier: the terminal* $\mathsf{Real}^{\uparrow}$ *seals* X, Y*, the regular rational source and readback rows* S_X, S_Y, R_X, R_Y*, the common* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row* C*, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row* E*, and the structural rows* H, K, P, N*. No quotient stream equality, host real equality, trichotomy, countable choice, selected limit, or ambient completeness principle can bypass those scopes.*

Proof. Project the carrier in Definition 1620.1. Its source rows are precisely the two terminal seals, the two regular rational source rows, the two readback rows, the common window, the dyadic ledger, the classifier row, the regular-Cauchy equivalence row, and the structural rows H, K, P, N.

Apply the NameCert obligations of Definition 1620.2. Carrier admission, tolerance classifier stability, the RegSeqRat handoff, real seal non-escape, and ledger exhaustion account for every accepted public read.

Use the RegSeqRat handoff of Definition 1620.5 and the real seal non-escape theorem Definition 1620.8. These force the read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ over the displayed seals X, Y and rule out host equality or quotient-stream shortcuts.

Finally combine ledger exhaustion from Definition 1620.9, tolerance-window determinacy from Definition 1620.10, and window scope from Definition 1620.11. The structural rows H, K, P, N only transport, replay, source, and name the same finite packet, so no listed non-source principle remains as a bypass. □

Lean4 ✓ `BEDC.Derived.RealClassifierSourceScope`

Theorem 1620.14 (RealClassifier tail-replacement stability)**.** *Let*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$$

be an accepted $\mathsf{RealClassifier}^{\uparrow}$ *carrier. Suppose a second displayed readback replaces* R_X *or* R_Y *by a* $\mathsf{RegSeqRat}^{\uparrow}$ *row with the same terminal* $\mathsf{Real}^{\uparrow}$ *seal, the same common* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the same* $\mathsf{RealNameClassifier}^{\uparrow}$ *row* C*, and a* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *tail-equivalence row matching* E*. Then the replacement preserves the public* $\mathsf{RealClassifier}^{\uparrow}$ *decision surface seen by completion-selector and completion-density consumers.*

Proof. Apply the RegSeqRat handoff theorem Definition 1620.5. It forces both the original and replacement readback to reach the classifier only through the displayed regular-rational rows, the common window W, the dyadic ledger D, the classifier row C, and the regular-Cauchy boundary E.

Use the regular-real tail-equivalence readback theorem Definition 1620.6. The replacement is admitted only as a tail-equivalent regular rational readback before the terminal Real seals are consumed, so it cannot introduce quotient stream equality or host real equality.

Apply window scope and public exactness, Definition 1620.11 and Definition 1620.16. Since the terminal seals, common window, dyadic ledger, classifier row, equivalence row, transport, replay, provenance, and local naming rows are shared, the public decision route remains the same finite route.

Finally read the downstream consumers through Definition 1620.20 and Definition 1620.21. These consumers inspect only the already exported classifier route, so replacing either regular rational readback by a tail-equivalent readback preserves the surface they see. □

Lean4 ✓ `BEDC.Derived.RealClassifierTailReplacementStability`

Theorem 1620.15 (RealClassifier dyadic-source embedding faithfulness)**.** *Let*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N)$$

and

$$Q' = (X', Y', S'_X, S'_Y, R'_X, R'_Y, W', D, C', E', H', K', P', N')$$

be accepted $\mathsf{RealClassifier}^{\uparrow}$ *carriers whose terminal* $\mathsf{Real}^{\uparrow}$ *seals arise from constant* $\mathsf{StreamName}^{\uparrow}$ *lifts of displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *source rows. Suppose the two packets share the same dyadic tolerance ledger D, expose the same common classifier window through W and W', and reach equal public* $\mathsf{RealClassifier}^{\uparrow}$ *boundary decisions. Then the equality at the classifier boundary preserves the displayed dyadic source equality surface: the only source equality exported to a* $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ *consumer is the one read through the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the constant* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the terminal* $\mathsf{Real}^{\uparrow}$ *seals.*

Proof. Apply the RegSeqRat handoff theorem Definition 1620.5 to both carriers. It forces each dyadic-source read to pass through the displayed regular-rational rows, the common window, the dyadic ledger, the classifier row, and the regular-Cauchy boundary before the public classifier boundary is reached.

Use common-window quotient refusal from Definition 1620.12. Since both packets share the classifier window and dyadic tolerance ledger, equality at the boundary cannot be replaced by host real equality or quotient stream equality.

Read the dyadic source ledger through chapter 291 and the constant-window route through chapter 126. These rows keep the two accepted RealClassifier packets tied to displayed dyadic sources rather than to unstated representatives.

Finally use the regular readback and tail-replacement surfaces, Definition 1620.6 and Definition 1620.14. The $\mathsf{RegSeqRat}^{\uparrow}$ readbacks and terminal $\mathsf{Real}^{\uparrow}$ seals may be transported only along the displayed finite route, so a $\mathsf{DyadicRealEmbeddingFaithful}^{\uparrow}$ consumer receives exactly the dyadic source equality surface and no selected limit, trichotomy, countable-choice schedule, or ambient-completeness coordinate. □

Lean4 ✓ `BEDC.Derived.RealClassifierDyadicSourceEmbeddingFaithfulness`

Theorem 1620.16 (RealClassifier public exactness surface)**.** *Every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *public classifier read is exactly the finite route through the two* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row. No accepted public read exports host real equality, quotient stream equality, a selected limit, trichotomy, or a completeness coordinate.*

Proof. Start with the carrier and NameCert obligations from Definition 1620.1 and Definition 1620.2. They list the two terminal seals, regular rational source and readback rows, common StreamName window, dyadic tolerance ledger, classifier row, equivalence boundary, and structural rows as the complete local surface.

Use the RegSeqRat handoff of Definition 1620.5 and the real seal non-escape theorem Definition 1620.8. These force every regular-rational read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ over the displayed seals X, Y and rule out host real equality, quotient stream equality, selected limits, trichotomy, countable choice, and ambient completeness as endpoints.

Finally apply ledger exhaustion from Definition 1620.9, tolerance-window determinacy from Definition 1620.10, and window scope from Definition 1620.11. The shared window and tolerance ledger determine the public classifier route, while H, K, P, N transport, replay, source, and locally name that exact finite packet. □

Lean4 ✓ `BEDC.Derived.RealClassifierPublicExactnessSurface`

Theorem 1620.17 (RealClassifier public exactness consumer reuse)**.** *Any* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *or completion-selector consumer that uses the public* $\mathsf{RealClassifier}^{\uparrow}$ *read receives the same finite route recorded by Definition 1620.16. The consumer route passes through the two* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row. It exports no selected representatives, quotient stream equality, countable choice, or ambient completeness.*

Proof. Start with the public exactness surface of Definition 1620.16. It already exhausts every accepted public classifier read by the two terminal seals, regular rational readbacks, common window, dyadic ledger, classifier row, equivalence row, and structural transport, replay, provenance, and naming rows.

Now read the consumer boundary. A $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ consumer can only reuse the public classifier route after the $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ row and componentwise hsame transport remain visible, while a completion-selector consumer receives the same route after the terminal seals and regular readbacks have been displayed.

Finally compare the excluded coordinates. Since neither consumer receives a row for selected representatives, quotient stream equality, countable choice, or ambient completeness, the public exactness surface is reused as the same finite route rather than replaced by a second classifier or completion endpoint. □

Lean4 ✓ `BEDC.Derived.RealClassifierPublicExactnessConsumerReuse`

Theorem 1620.18 (RealClassifier psame bridge consumption)**.** *The public exactness surface of Definition 1620.16 is exactly the RealClassifier input consumed by the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *psame bridge of chapter 1820. The bridge reads the same sealRead, equalityRead, and realClassifierRead route through the two* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row.*

Proof. Start with the public exactness surface. It already exhausts every accepted public classifier read by the finite route through the two Real seals, regular rational readbacks, shared window, dyadic ledger, classifier row, equivalence row, and structural rows.

Now read the CauchyEquivalenceSetoid$^{\uparrow}$ bridge chapter. Its psame-facing bridge consumes a RealClassifier boundary only after the same sealRead, equalityRead, and realClassifierRead rows are displayed, so the bridge receives no second classifier channel.

Finally compare the source lists. Both routes expose the same $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$ rows before any public equality-facing read is available. Thus the psame bridge consumes the RealClassifier public exactness surface without adding quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealClassifierUp.RealClassifierPsameBridgeConsumption`

Theorem 1620.19 (RealClassifier public export readback)**.** *Every public* RealClassifier$^{\uparrow}$ *read factors through the displayed* RegSeqRat$^{\uparrow}$, RegularCauchyEquivalence$^{\uparrow}$, RegularRealTailEquivalence$^{\uparrow}$, RealNameClassifier$^{\uparrow}$, *and* RealCompletionSelectorSeal$^{\uparrow}$ *rows before any completion-density or audit consumer can inspect the classifier. In carrier notation, for*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the exported read is the finite route through $S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$ *over the terminal* Real$^{\uparrow}$ *seals* X, Y*, not a direct equality read on completed real names.*

Proof. Start with the public exactness surface of Definition 1620.16. It already binds every public classifier read to the two terminal seals, the regular rational readbacks, the common StreamName window, the DyadicRatCore ledger, the RealNameClassifier row, the RegularCauchyEquivalence row, and the structural transport, replay, provenance, and local naming rows.

Apply the RegSeqRat handoff theorem Definition 1620.5 and the regular-real tail-equivalence readback theorem Definition 1620.6. These force the public read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before it reaches a same-real or tail-equivalence-facing consumer.

Next use the completion-selector-seal readback theorem Definition 1620.20. A RealCompletionSelectorSeal$^{\uparrow}$ consumer therefore receives the classifier only after the Real seals, regular readbacks, common window, tolerance ledger, classifier row, equivalence row, and structural rows have been displayed.

Finally apply the completion-density consumer theorem Definition 1620.21. Since density and audit consumers inspect only that already exported route, no consumer can replace it by host real equality, quotient stream equality, selected limits, trichotomy, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealClassifierPublicExportReadback`

Theorem 1620.20 (RealClassifier completion-selector-seal readback)**.** *Every accepted* RealClassifier$^{\uparrow}$ *public read that is consumed by a* RealCompletionSelectorSeal$^{\uparrow}$ *packet must pass through the same* Real$^{\uparrow}$ *seals,* RegSeqRat$^{\uparrow}$ *readbacks,* StreamName$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RealNameClassifier$^{\uparrow}$ *row, and* RegularCauchyEquivalence$^{\uparrow}$ *row before any selector-seal budget, window, or readback row is exposed. The handoff exports no selected limit, quotient stream equality, trichotomy, countable choice, or ambient-completeness coordinate.*

Proof. Start with the carrier of Definition 1620.1 and the NameCert obligations of Definition 1620.2. These list the two terminal Real$^{\uparrow}$ seals, regular rational rows, common StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ ledger, classifier row, equivalence row, and structural rows as the full accepted source.

Apply the RegSeqRat handoff of Definition 1620.5 and the real seal non-escape theorem Definition 1620.8. They force the read through $S_X, S_Y, R_X, R_Y, W, D, C, E$ before the terminal Real

boundary can be used, and they rule out host equality, quotient-stream equality, selected limits, trichotomy, countable choice, and ambient completeness as endpoints.

Use ledger exhaustion from Definition 1620.9 and the public exactness surface of Definition 1620.16. The shared window and dyadic ledger are already exhausted before the public classifier boundary is reached, so a selector-seal consumer receives the same finite classifier route rather than a second tolerance or equality channel.

Finally attach the sibling $\mathsf{RealCompletionSelectorSeal}^{\uparrow}$ route of chapter 1081. Its budget, window, and readback rows can only consume the displayed classifier read after the RealClassifier route has exposed the seals, readbacks, window, ledger, classifier row, equivalence row, and structural transport, replay, provenance, and naming rows. No coordinate remains from which to export a selected limit, quotient stream equality, trichotomy, countable choice, or ambient-completeness claim. □

Lean4 ✓ `BEDC.Derived.RealClassifierCompletionSelectorSealReadback`

Theorem 1620.21 (RealClassifier completion-density consumer)**.** *Every completion-density consumer of an accepted* $\mathsf{RealClassifier}^{\uparrow}$ *read must expose the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the rational-density route of chapter 7239 or the regular Cauchy dense-embedding route of chapter 1453, and the classifier transport rows before the read is used as a countable dense surface. In carrier notation, for*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the dense consumer may reuse only the displayed route through $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$ *together with the sibling density rows. It may not substitute selected representatives, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the public exactness surface of Definition 1620.16. It binds every accepted classifier read to the two terminal Real seals, the regular rational readbacks, the common StreamName window, the DyadicRatCore ledger, the comparison row, the regular-Cauchy equivalence boundary, and the structural transport, replay, provenance, and naming rows.

Next use the regular-real tail-equivalence readback of Definition 1620.6. It keeps a same-real read inside the shared regular-rational, finite-window, dyadic-ledger, and comparison route before the public classifier boundary is reached.

Now pass to the density sibling. A rational-density consumer reads chapter 7239; a completion-facing regular-Cauchy consumer reads chapter 1453. In either case, the dense surface receives the already displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{Real}^{\uparrow}$ seal route from the classifier packet.

Finally read H, K, P, N only as classifier transport, replay, provenance, and local naming. Since the carrier has no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness, a completion-density consumer cannot obtain any of those endpoints from the accepted $\mathsf{RealClassifier}^{\uparrow}$ read. □

Lean4 ✓ `BEDC.Derived.RealClassifierCompletionDensityConsumer`

Theorem 1620.22 (RealClassifier audit route)**.** *For every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the audit-facing classifier route is exactly the ordered read through the two $\mathsf{Real}^{\uparrow}$ *seals* X, Y*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_X, R_Y*, the shared* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance*

ledger D, the $\mathsf{RealNameClassifier}^{\uparrow}$ *row C, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row E, componentwise* hsame *transport H, displayed* Cont *replay K,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row N. No audit read may replace that route by quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness.*

Proof. Use the RegSeqRat handoff theorem Definition 1620.5. It forces every classifier read to expose the regular rational sources and readbacks before the public classifier boundary is reached.

Apply window scope and source scope, Definition 1620.11 and Definition 1620.13. These identify the shared StreamName window, dyadic tolerance ledger, classifier row, equivalence row, terminal Real seals, and structural rows as the complete source list for the read.

Finally use the public exactness surface of Definition 1620.16. Since that theorem already binds every public classifier read to the exact finite route through $X, Y, R_X, R_Y, W, D, C, E, H, K, P, N$, the audit route is the same ordered source read and has no ambient equality, choice, or completeness endpoint. □

Lean4 ✓ `BEDC.Derived.RealClassifierAuditRoute`

Theorem 1620.23 (RealClassifier audit-density coherence)**.** *Every audit-facing classifier read of an accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier that is later consumed by the rational-density route of chapter 7239 or the regular-Cauchy dense-embedding route of chapter 1453 must use the same ordered route through the* $\mathsf{Real}^{\uparrow}$ *seals,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, shared* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{RealNameClassifier}^{\uparrow}$ *row,* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row. No dense consumer may replace that route by selected representatives, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the audit route theorem Definition 1620.22. It fixes the audit-facing read as the ordered source route through the two terminal seals, regular readbacks, common window, dyadic ledger, classifier row, regular-Cauchy boundary, transport, replay, provenance, and local naming rows.

Apply the completion-density consumer theorem Definition 1620.21. It says that rational-density and regular-Cauchy dense-embedding consumers receive the already displayed classifier route together with their sibling density rows, rather than a second representative or a detached completion endpoint.

Finally use the public exactness surface of Definition 1620.16. Since every accepted public read is exhausted by the same finite rows, an audit-facing read and its dense consumer have one shared route through $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$. The excluded selected representatives, quotient stream equality, countable choice, and ambient completeness coordinates are absent from both surfaces. □

Lean4 ✓ `BEDC.Derived.RealClassifierAuditDensityCoherence`

Theorem 1620.24 (RealClassifier theorem-check target handoff)**.** *The theorem-check target for* $\mathsf{RealClassifier}^{\uparrow}$ *is exactly the public exactness surface route through the two* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row. The handoff targets that route only; it does not change the recorded formal status before Lean supplies the corresponding theorem-checked evidence.*

Proof. Start with the public exactness surface of Definition 1620.16. It already states the exact public classifier route through the Real seals, regular rational readbacks, shared window, dyadic ledger, classifier row, equivalence row, and structural rows.

Next use the audit route theorem Definition 1620.22. It confirms that the audit-facing read is the same ordered source route, not a second equality, representative, choice, or completeness channel.

Finally apply audit-density coherence, Definition 1620.23. Since density consumers receive the same displayed public route, the theorem-check target is precisely the public exactness surface named by the chapter leantarget. □

Lean4 ✓ `BEDC.Derived.RealClassifierTheoremCheckTargetHandoff`

Theorem 1620.25 (RealClassifier theorem-check readback route)**.** *For every accepted* $\mathsf{RealClassifier}^{\uparrow}$ *carrier*

$$Q = (X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N),$$

the theorem-check readback route is the same finite route through the two $\mathsf{Real}^{\uparrow}$ *seals* X, Y*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_X, R_Y*, one* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, the* $\mathsf{RealNameClassifier}^{\uparrow}$ *row* C*, the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* K*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$ *row* N*. The readback route exports no quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness beyond the theorem-check target handoff.*

Proof. Begin with public exactness, Definition 1620.16. It binds every public classifier read to the two terminal Real seals, regular rational readbacks, common StreamName window, DyadicRatCore tolerance ledger, RealNameClassifier row, RegularCauchyEquivalence row, and structural rows.

Use the audit route theorem Definition 1620.22. It identifies the audit-facing source order as the same route through $X, Y, S_X, S_Y, R_X, R_Y, W, D, C, E, H, K, P, N$, without a second equality channel or representative selector.

Apply audit-density coherence, Definition 1620.23. Density consumers receive the already displayed public and audit route, so the theorem-check readback cannot replace it with a quotient stream equality, selected limit, countable-choice, or ambient-completeness endpoint.

Finally apply the theorem-check target handoff Definition 1620.24. The handoff targets exactly this public exactness surface, and therefore the readback route for the theorem-check target is the same finite route through the two Real seals, regular readbacks, StreamName window, DyadicRatCore ledger, classifier row, equivalence row, transport, replay, provenance, and local NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1620.1, Definition 1620.2, Definition 1620.5, Definition 1620.8, Definition 1620.9, Definition 1620.10, Definition 1620.11, Definition 1620.13, Definition 1620.16, Definition 1620.19, Definition 1620.20, Definition 1620.21, Definition 1620.22, and Definition 1620.23 bind the mature route from RegSeqRat, RealNameClassifier, StreamName, DyadicRatCore, and RegularCauchyEquivalence rows into public, completion-facing, and audit-facing RealClassifier reads over the terminal RealUp seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealClassifier}^{\uparrow}}$, scaffoldChecked)

Lean target: `BEDC.Derived.RealClassifierPublicExactnessSurface`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, trichotomy, countable choice, ambient completeness, or machine-checked standard bridge evidence.

Upgrade path: The remaining work is formal audit-clean and bridge-checked coverage for the same mature public, completion-facing, and audit-facing route.

Constructive story: A RealClassifier$^{\uparrow}$ packet is generated from two RealUp seals, regular rational readbacks, one common StreamName window, DyadicRatCore tolerance data, a RealNameClassifier row, a RegularCauchyEquivalence row, componentwise transport, continuation replay, package provenance, and local naming.

Chapter 1621

A Concrete Naming Certificate for $\mathsf{StationaryDyadicStream}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{StationaryDyadicStream}^{\uparrow}$ packet records the finite constant dyadic stream route that feeds the open Real and Completion path. It begins at the dyadic source boundary of chapter 291, passes through the constant stream window of chapter 126, hands the regularity row to chapter 301, reaches the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and is read alongside the normal-form and zero-real boundaries of chapter 1613 and chapter 1619. The packet names a displayed constant dyadic stream, not a quotient stream, selected limit, host real equality, trichotomy, countable choice, or ambient completeness principle.

Definition 1621.1 (Stationary dyadic stream carrier)**.** *A* $\mathsf{StationaryDyadicStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, including the finite dyadic endpoint, tolerance ledger, and readback surface inspected before any stream request.*
2. *W is the constant* $\mathsf{StreamName}^{\uparrow}$ *window, recording that every admitted finite request reads the same dyadic row D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff, exposing the constant dyadic window as a regular rational sequence with zero internal drift across displayed finite windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, reached only after the dyadic source row, constant StreamName window, and regularity handoff have all been displayed.*
5. *H records componentwise* hsame *transport for D, W, R, E, C, P, N and for every displayed finite request window carried by W.*
6. *C records displayed* Cont *replay from D into W, from W into R, and from R into the terminal seal E.*
7. *P records* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicNormalForm}^{\uparrow}$, $\mathsf{RealZero}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{StationaryDyadicStream}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1621.2 (Stationary dyadic stream NameCert obligations)**.** *For every accepted* $\mathsf{StationaryDyadicStream}^{\uparrow}$ *carrier* $S = (D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StationaryDyadicStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the DyadicRatCore source row, constant StreamName window, RegSeqRat handoff, terminal RealUp seal, transport, continuation, provenance, and local naming rows;*
2. *constant-window exactness, requiring each displayed finite request in* W *to read the same source row* D*;*
3. *regularity handoff, requiring* R *to expose zero pairwise drift between all displayed windows admitted by the same constant source route;*
4. *terminal seal routing, requiring* E *to be exported only after* D*,* W*, and* R *have been displayed in that order;*
5. *componentwise transport and replay stability, requiring* H *and* C *to preserve the same dyadic source, constant window, regularity handoff, and real seal rows;*
6. *ledger refusal, excluding quotient streams, selected limits, host real equality, trichotomy, countable choice, and ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1621.1. The projection exposes only D, W, R, E, H, C, P, N, so carrier admission and the local naming row are finite.

For constant-window exactness, induct on the displayed finite request window stored in W. The empty window has no request to compare. In the successor step, the new request is read from the same DyadicRatCore row D, and the induction hypothesis preserves the earlier requests.

The RegSeqRat handoff is the same induction read through R. Each pair of displayed windows has identical dyadic readback because both are componentwise transported to D, so the regularity ledger records zero displayed drift without selecting a limit.

Replay by C gives the ordered route $D \to W \to R \to E$, and transport by H preserves the components of that route. Since the carrier contains no quotient stream, selected limit, host real equality, trichotomy, countable choice, or completeness coordinate, the local NameCert surface cannot export any such principle. □

Theorem 1621.3 (Stationary dyadic stream RegSeqRat handoff)**.** *Every accepted* $\mathsf{StationaryDyadicStream}^{\uparrow}$ *carrier* $S = (D, W, R, E, H, C, P, N)$ *exports its regular rational sequence boundary through the finite route*

$$D \longrightarrow W \longrightarrow R$$

where W *repeats the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row and* R *records regularity by componentwise equality of finite windows. The handoff does not require quotient stream equality, a chosen Cauchy limit, trichotomy, countable choice, or ambient completeness.*

Proof. Read D through the constant StreamName window W. Induction over any displayed finite window shows that every entry in the window is the same dyadic source row.

The row R consumes exactly that finite constant-window evidence. Componentwise hsame transport turns equality of entries in W into the regular-rational zero-drift ledger demanded by $\mathsf{RegSeqRat}^{\uparrow}$.

No further principle is available in the carrier. The handoff therefore remains the displayed route $D \to W \to R$, rather than a quotient-stream class or a selected limiting real. □

Theorem 1621.4 (Stationary dyadic stream constant embedding)**.** *Every accepted* $\mathsf{StationaryDyadicStream}^{\uparrow}$ *packet whose source row is a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *datum exports the same public route as the dyadic constant embedding into* $\mathsf{Real}^{\uparrow}$*:*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

For a carrier $S = (D, W, R, E, H, C, P, N)$*, this route is read through the rows* D, W, R, E, H, C, P, N*. It does not introduce quotient stream equality, a selected limit, host real equality, countable choice, or ambient completeness.*

Proof. Use the carrier definition to project the accepted packet. The source row D is the displayed dyadic rational datum, and the window row W records that every finite stream request repeats that same datum.

Apply the RegSeqRat handoff theorem. It identifies R as the regular rational readback of the constant finite windows, with zero displayed drift between the windows admitted by W.

Finally apply the Real seal boundary theorem. The seal E is available only after D, W, R have been displayed, and H, C, P, N only transport, replay, source, and name the same finite route. The packet therefore presents exactly the constant dyadic embedding route and no quotient stream, selected limit, host real equality, countable-choice schedule, or ambient completeness principle. □

Theorem 1621.5 (Stationary dyadic stream Real seal boundary)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{StationaryDyadicStream}^{\uparrow}$ *packet must inspect the dyadic source row, the constant StreamName window, and the RegSeqRat handoff before reading the terminal seal* E*. Consequently the public real-facing boundary is the finite route*

$$D \longrightarrow W \longrightarrow R \longrightarrow E$$

together with H, C, P, N*, and no consumer may replace that route by quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient Real completeness.*

Proof. Start with an accepted carrier $S = (D, W, R, E, H, C, P, N)$. The seal E is a terminal row in the carrier inventory, so it is not available before the source row D, the window W, and the handoff row R have been displayed.

Continuation replay C records the route from the dyadic source through the constant stream and regularity rows into E. Componentwise transport H preserves the same route for any accepted copy of the packet.

Thus a downstream $\mathsf{Real}^{\uparrow}$ or Completion-facing consumer receives the finite real-facing boundary $D \to W \to R \to E$ with its structural rows. The missing coordinates rule out quotient streams, host real equality, selected limits, trichotomy, countable choice, and ambient completeness as replacement routes. □

Closure status for $\mathsf{NameCert}_{\mathsf{StationaryDyadicStream}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StationaryDyadicStream}^{\uparrow}$ packet is generated from a DyadicRatCore source row, a constant StreamName finite-window row, a RegSeqRat regularity handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StationaryDyadicStream}^{\uparrow}}$, seed)

Scope closed: Definition 1621.1, Definition 1621.2, Definition 1621.3, and Definition 1621.5 seed the finite constant-dyadic stream bridge from DyadicRatCore to Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StationaryDyadicStream}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.StationaryDyadicStreamUp`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, host real equality, trichotomy, countable choice, ambient Real completeness, Lean theorem checking, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, constant-window exactness, RegSeqRat handoff, RealUp seal routing, componentwise transport and replay stability, and ledger refusal for the same finite packet.

Chapter 1622

A Concrete Naming Certificate for $\mathsf{KleeneTree}^{\uparrow}$

Following Kleene's computability-theoretic tree counterexample tradition and Bishop–Bridges constructive analysis, the $\mathsf{KleeneTree}^{\uparrow}$ packet records a decidable finite-prefix tree surface without turning that surface into an infinite path-selection principle. It sits between the binary prefix carrier of chapter 1475, the finite-bar discipline of chapter 400, the window schedule of chapter 126, the regular rational readback surface of chapter 301, and the terminal real boundary of chapter 38. The chapter makes explicit that finite tree rows and stream-name windows are not countable choice, not a selected branch, not quotient stream equality, not $\mathsf{Real}^{\uparrow}$ completeness, and not open-cover compactness.

The finite-prefix tree row is consumed through the $\mathsf{CantorSpace}^{\uparrow}$ prefix carriers of chapter 1475, the $\mathsf{FanTheorem}^{\uparrow}$ finite-bar discipline, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ boundary before any obstruction row can be interpreted by a downstream consumer. In particular, the finite-prefix obstruction is consumed by the $\mathsf{StreamName}^{\uparrow}$ window schedule of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback boundary of chapter 301 before any Real-facing consumer reaches chapter 38.

Concretely, the branch-facing read is not internal to the Kleene packet. It reads $\mathsf{CantorSpace}^{\uparrow}$ branch windows through chapter 1475 and compares those finite windows with the $\mathsf{FanTheorem}^{\uparrow}$ bar and modulus-failure boundary of chapter 400. The carrier therefore presents a horizontal dependency: finite prefix data can be tested against Cantor-space windows and fan-theorem bar failures, but the obstruction row still blocks any selected infinite branch.

Definition 1622.1 (Kleene tree carrier)**.** *A* $\mathsf{KleeneTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, B, L, S, O, H, C, P, N)$$

with the following displayed rows:

1. *T is a finite binary prefix-tree ledger, whose nodes are displayed finite $0/1$ words and whose extension relation is prefix-closed only for the listed words.*
2. *B is the $\mathsf{Bool}^{\uparrow}$ decidable node ledger classifying each listed finite word as admitted or refused inside the displayed tree packet.*
3. *L is the $\mathsf{List}^{\uparrow}$ prefix spine recording the parent chain for every listed node.*
4. *S is the $\mathsf{StreamName}^{\uparrow}$ window schedule specifying which finite prefixes a consumer may request.*

5. *O is the obstruction row: no computable infinite branch, no selected path, and no branch-producing choice functional is a coordinate of the packet.*
6. *H records componentwise* hsame *transport for the tree ledger, Bool ledger, List spine, StreamName window schedule, obstruction row, continuation row, provenance row, and local name.*
7. *C records displayed* Cont *traversal from a finite request through S, then through the prefix spine L, then through the node classifier B, and finally to the obstruction row O.*
8. *P is the* Pkg *provenance over* $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$ *row.*

The carrier stores no infinite path object, no countable-choice schedule, no quotient stream equality, no selected real name, no $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no open-cover compactness principle.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeTasteGate single carrier alignment`

Theorem 1622.2 (Kleene tree NameCert obligations)**.** *For every accepted* $\mathsf{KleeneTree}^{\uparrow}$ *carrier $K = (T, B, L, S, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite binary prefix tree, decidable Bool node ledger, List prefix spine, StreamName window schedule, obstruction row, transport, traversal, provenance, and local naming row;*
2. *prefix stability, requiring every accepted child row in T to have its displayed parent in L and its node decision in B;*
3. *window exactness, requiring every consumer request to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S before any tree node is inspected;*
4. *traversal stability, requiring* hsame *transport and* Cont *replay to preserve the same finite prefix ledger and the same obstruction row;*
5. *boundary refusal, requiring O to expose no computable infinite branch, no branch selector, no countable-choice functional, no quotient stream equality, no* $\mathsf{Real}^{\uparrow}$ *completeness, and no open-cover compactness principle.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeNameCert obligations`

Proof. Project the carrier of Definition 1622.1. The tuple contains exactly the finite tree ledger T, the Bool node classifier B, the List prefix spine L, the StreamName window schedule S, the obstruction row O, and the structural rows H, C, P, N.

Prefix stability is read from T, B, L. Since each listed child is accompanied by a displayed parent spine and a displayed Bool decision, a consumer cannot infer membership for an unlisted node or a branch beyond the finite ledger.

Window exactness is read from S and C. The continuation row routes each request through the finite StreamName window before the prefix ledger and classifier are inspected.

Transport and replay are componentwise. The rows H and C can move or traverse only the displayed finite packet, so the obstruction row O remains part of the same carrier throughout the read.

The carrier contains no coordinate for an infinite path, countable choice, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, or open-cover compactness. Thus the local $\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$ row cannot export any of those principles. □

Theorem 1622.3 (Kleene tree stream-name path boundary)**.** *Every consumer of an accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet reads only the finite route*

$$S \longmapsto L \longmapsto B \longmapsto T \longmapsto O \longmapsto H, C, P, N.$$

Consequently finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{CantorSpace}^{\uparrow}$ *prefix rows, and* $\mathsf{FanTheorem}^{\uparrow}$ *bar-facing tests supplied by the packet do not determine a computable infinite branch, a selected Cantor point, quotient stream equality, countable choice,* $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTree streamname path boundary`

Proof. Begin with an accepted carrier $K = (T, B, L, S, O, H, C, P, N)$. A consumer first receives the finite StreamName schedule S, not an infinite stream.

The traversal then passes through the List prefix spine L and the Bool node ledger B before reading the displayed prefix-tree ledger T. Each step concerns only listed finite words and their displayed parent chains.

The obstruction row O is downstream of the same finite traversal. It records that no computable infinite branch, branch selector, or countable-choice functional belongs to the packet.

The structural rows H, C, P, N preserve and name that route. Therefore the finite data may be reused by CantorSpace, FanTheorem, RegSeqRat, or Real-facing consumers only as finite prefix information, never as quotient stream equality, an open-cover compactness principle, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Theorem 1622.4 (Kleene tree Cantor–Fan dependency route)**.** *Every branch-facing read of an accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet first consumes the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix rows of chapter 1475 and the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar rows of chapter 400 before it can interpret the obstruction row as a stream-name, regular-rational, or real-facing boundary. The visible route remains*

$$S, L, B, T, O, H, C, P, N,$$

and no branch selector, selected Cantor point, countable-choice functional, quotient stream equality, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is produced.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.CantorFanDependencyRoute.KleeneTreeCantorFanDependencyRoute`

Proof. Project the carrier of Definition 1622.1. The only branch-facing data are the finite StreamName window S, the List prefix spine L, the Bool node ledger B, the finite tree ledger T, the obstruction row O, and the structural rows H, C, P, N.

Apply Definition 1622.3. It already forces any branch-facing read to remain inside the finite route and refuses infinite paths, selected Cantor points, and quotient stream equality.

The $\mathsf{CantorSpace}^{\uparrow}$ chapter supplies the finite prefix rows that a branch consumer may test, while the $\mathsf{FanTheorem}^{\uparrow}$ chapter supplies the finite-bar rows against which those prefixes are compared. Since both surfaces are read before the obstruction row is interpreted, the Kleene packet records a horizontal dependency through Cantor-space prefixes and fan-theorem bars rather than a free-standing path object. □

Theorem 1622.5 (Kleene tree Specker boundary route)**.** *Every* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *consumer using a* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction must first read the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix rows,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *boundary before it can consume the obstruction as a Specker-style boundary row. The public route factors through the Kleene carrier rows* $S, L, B, T, O, H, C, P, N$ *together with the Specker boundary route of chapter 3516.*

The consumer receives no branch selector, countable-choice schedule, quotient stream equality, selected Cantor point, path-coded real, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness principle from the Kleene packet.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeSpeckerBoundaryRoute`

The Specker sequence boundary of chapter 3516 consumes this Kleene finite-prefix obstruction as a nonlocated-supremum boundary: the obstruction row remains finite prefix data, and the composed route still refuses branch selectors, countable-choice schedules, quotient stream equality, selected Cantor points, and ambient completeness.

Proof. Start with the Kleene carrier of Definition 1622.1. Its public obstruction route is finite: $\mathsf{StreamName}^{\uparrow}$ schedule S, List prefix spine L, Bool node ledger B, finite prefix tree T, obstruction row O, and structural rows H, C, P, N.

Apply Definition 1622.3. It forces every tree read to pass through finite StreamName windows, CantorSpace prefix data, and FanTheorem bar-facing tests before the obstruction row is interpreted. No infinite branch or selected Cantor point is exposed there.

Now read the same obstruction through chapter 3516. The Specker boundary carrier consumes finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, and a Real seal or refusal row before any completion-facing consumer can cite the nonlocated-supremum obstruction.

Composing the two routes gives the advertised boundary. A Real or Completion obstruction consumer may cite the Kleene row only after the finite prefix, finite-bar, StreamName, RegSeqRat, and Real boundary rows are displayed. Since neither route contains a branch selector, countable-choice schedule, quotient stream equality, path-coded real, ambient Real completeness, or open-cover compactness, the composed Specker-facing read exports none of those objects. □

Theorem 1622.6 (Kleene tree Cantor-space fan boundary)**.** *Every Specker-facing* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction must pass through the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix windows and the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar rows before it can be read as a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *boundary. The route exposes only the finite rows* $S, L, B, T, O, H, C, P, N$*, so the later boundary read cannot recover an infinite branch, a selected Cantor point, quotient stream equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeCantorSpaceFanBoundary`

Proof. Start with Definition 1622.5. It already places every Specker-facing read behind the finite prefix windows, finite-bar rows, StreamName windows, RegSeqRat readback, and Real boundary.

Now isolate the horizontal dependency. The $\mathsf{CantorSpace}^{\uparrow}$ rows provide the finite prefix windows, and the $\mathsf{FanTheorem}^{\uparrow}$ rows provide the finite-bar tests attached to those windows. A Specker-facing consumer therefore reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ only after those two finite surfaces are displayed.

The carrier rows are still the finite rows $S, L, B, T, O, H, C, P, N$. Since none of them stores a branch selector, selected Cantor point, quotient stream equality, open-cover compactness principle, or ambient Real-completeness theorem, the composed boundary read exports none of those objects. □

Theorem 1622.7 (Kleene tree prefix-induction closure)**.** *Every accepted finite prefix-tree ledger in a* $\mathsf{KleeneTree}^{\uparrow}$ *carrier is closed under displayed parent-prefix induction. Each accepted non-root listed word has its displayed parent in the* $\mathsf{List}^{\uparrow}$ *prefix spine and its* $\mathsf{Bool}^{\uparrow}$ *node decision in the ledger, and* hsame/Cont *replay preserves this finite closure without selecting an infinite branch. The induction uses only the* $T, B, L, S, O, H, C, P, N$ *carrier rows.*

Proof. Induct on the displayed prefix spine L of Definition 1622.1. The root case is listed directly in the finite tree ledger T and has its Bool decision in B, so no parent row is required.

For a non-root listed word, the next row of L displays its parent prefix. The induction hypothesis places that parent inside the same finite spine, while the corresponding row of B records the accepted or refused node decision for the listed word.

The StreamName schedule S only chooses finite windows into this displayed spine. It cannot add a branch beyond the rows already present in T, B, L.

Finally H and C transport and replay the same finite packet, while P, N record provenance and local naming. Since the obstruction row O remains part of the carrier, the prefix induction closes the finite ledger without selecting an infinite path or invoking countable choice. □

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeCarrier prefix induction closure`

Theorem 1622.8 (Kleene tree fan boundary)**.** *For every accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet, any* $\mathsf{FanTheorem}^{\uparrow}$ *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer sees only the finite rows* $S, L, B, T, O, H, C, P, N$*. The consumer may use finite-bar tests and finite prefix checks along those rows, but it receives no infinite branch, no branch selector, no countable-choice functional, no quotient stream equality, no open-cover compactness principle, and no* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeCarrier fan boundary`

Proof. Project the carrier of Definition 1622.1. The displayed packet contains the finite StreamName schedule S, the List prefix spine L, the Bool node ledger B, the finite prefix-tree ledger T, the obstruction row O, and the structural rows H, C, P, N.

Apply Definition 1622.2. Prefix stability, window exactness, and traversal stability force every FanTheorem or CantorSpace read to remain inside the displayed finite rows.

It remains to identify what cannot pass through that read. Apply Definition 1622.3: the finite route is not a path selector, not a countable-choice functional, not quotient stream equality, not open-cover compactness, and not $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1622.9 (Kleene tree selector refusal)**.** *Every read from an accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet into* $\mathsf{CauchySelectionFunction}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*, or* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *factors through the finite rows* $S, L, B, T, O, H, C, P, N$*. The read may reuse finite StreamName windows, finite prefix ledgers, and finite obstruction data, but it cannot produce a branch selector, a countable-choice functional, or a sequential Cauchy selector row. The obstruction is carried by the row* O *together with the path and fan boundaries of Definition 1622.3 and Definition 1622.8.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeSelectorRefusal`

Proof. Project the accepted carrier of Definition 1622.1. The only rows exposed to a selector-facing consumer are the finite StreamName schedule S, the prefix spine L, the Bool ledger B, the tree ledger T, the obstruction row O, and the structural rows H, C, P, N.

Apply Definition 1622.3. A read through S, L, B, T supplies finite prefix information, not an infinite path, quotient stream equality, or a countable-choice functional.

Apply Definition 1622.8. Any CantorSpace or FanTheorem-facing use of the same packet remains inside the finite rows and reaches the obstruction row before it can export a selector.

The row O explicitly denies a computable infinite branch, a selected path, and a branch-producing choice functional as packet coordinates. A sequential Cauchy selector row would require a finite request ledger of selected Cauchy answers, but the Kleene packet contains only tree-prefix and stream-window rows. Therefore no read into $\mathsf{CauchySelectionFunction}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, or $\mathsf{SequentialCauchySelector}^{\uparrow}$ escapes the finite obstruction boundary. □

Theorem 1622.10 (Kleene tree Real-completion non-escape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion-facing read sourced from an accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet factors through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the obstruction row before any real seal can be inspected. Consequently the read cannot assert a selected Cantor point, a path-coded real, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Proof. First apply Definition 1622.8. Any CantorSpace or FanTheorem data exposed by the packet is finite row data, so a completion-facing consumer inherits only the same finite windows and the same obstruction row.

Next apply Definition 1622.9. A selector-facing read from the same packet must pass through $S, L, B, T, O, H, C, P, N$, and the obstruction row O blocks branch selectors, countable-choice functionals, and sequential Cauchy selector rows before any RegSeqRat or Real seal is inspected.

Now apply Definition 1622.3. The route through $S, L, B, T, O, H, C, P, N$ reaches the obstruction row before any Real-facing interpretation can treat a stream name as a completed real.

The obstruction row contains no selected Cantor point, no path-coded real, no open-cover compactness principle, and no $\mathsf{Real}^{\uparrow}$ completeness theorem. Thus a RegSeqRat or Real completion read cannot escape the finite StreamName boundary supplied by the Kleene tree packet. □

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeRealCompletionNonescape`

Theorem 1622.11 (Kleene tree uniform-structure boundary)**.** *Any* $\mathsf{RealUniformStructure}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction must first pass through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *boundary, and the* $\mathsf{RealUniformStructure}^{\uparrow}$ *L10 finite-source gate of Definition 1624.4. The composed route exports no quotient stream equality, selected path, selected limit, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Real-completion nonescape of Definition 1622.10. It forces every RegSeqRat or Real completion-facing read sourced from the Kleene obstruction to pass through finite StreamName windows and the obstruction row before any Real seal is inspected.

Next read the Real-uniform consumer through chapter 1624. Its L10 finite-source gate, Definition 1624.4, requires the Real endpoint and seal row, Metric distance row, DyadicRatCore radius ledger, StreamName finite-window row, and RegSeqRat readback row before any UniformSpace or CauchyFilter handoff is available.

Composing these two row orders gives the stated boundary. The Kleene side contributes only finite prefix and obstruction data, while the RealUniformStructure side contributes only the displayed L10 finite-source rows. Since neither route has a coordinate for quotient stream equality, selected paths, selected limits, open-cover compactness, or ambient Real completeness, the uniform-structure-facing consumer cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeUniformStructureBoundary`

Theorem 1622.12 (KleeneTree scoped prefix-choice refusal)**.** *Every accepted* $\mathsf{KleeneTree}^{\uparrow}$ *packet whose prefix induction, fan boundary, selector refusal, and Real-completion nonescape rows are displayed can answer a finite prefix-extension request only through the tree carrier, the* $\mathsf{CantorSpace}^{\uparrow}/\mathsf{FanTheorem}^{\uparrow}$ *boundary rows, the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the existing refusal ledger. The answer cannot export a branch selector, fan theorem, countable-choice functional, quotient path, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KleeneTreeUp.KleeneTreeCarrier scoped prefix choice refusal`

Proof. Project the accepted carrier $K = (T, B, L, S, O, H, C, P, N)$ of Definition 1622.1. A finite prefix-extension request first reads the finite tree ledger T, the Bool node ledger B, and the List prefix spine L.

Apply Definition 1622.7. It keeps every accepted extension inside the displayed parent-prefix induction over T, B, L and does not select an infinite branch.

Apply Definition 1622.8 and Definition 1622.9. The CantorSpace and FanTheorem reads remain finite boundary reads, while selector-facing consumers meet the obstruction row O before any branch-producing coordinate can appear.

Finally apply Definition 1622.10. A completion-facing read must pass through $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before any Real seal is inspected. Since the carrier inventory has no branch selector, fan theorem, countable-choice functional, quotient path, or ambient Real-completeness coordinate, the scoped prefix-extension answer exports none of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KleeneTree}^{\uparrow}$ packet is generated from a finite binary prefix-tree ledger, a Bool decidable node ledger, a List prefix spine, a StreamName window schedule, an obstruction row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$, obligation)

Scope closed: Definition 1622.1, Definition 1622.2, Definition 1622.3, Definition 1622.8, Definition 1622.9, and Definition 1622.10 seed the finite prefix-tree boundary separating StreamName, FanTheorem, CantorSpace, selector-facing, RegSeqRat, and Real-facing data from path-selection and completeness principles.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KleeneTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a computable infinite branch, selected Cantor point, quotient stream equality, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires an explicit dependency scope for the $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Chapter 1623

A Concrete Naming Certificate for $\mathsf{DyadicComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicComparison}^{\uparrow}$ packet names the finite comparison surface for dyadic rational rows before regular rational streams and completed real names read an order-facing observation. It sits after the dyadic core boundary of chapter 291, the canonical finite presentation of chapter 1613, and the finite schedule surface of chapter 126; it feeds the regular rational readback of chapter 301 and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records comparison rows for displayed dyadic windows, not host trichotomy, quotient rational equality, completed real order, located cuts, completeness, or a standard bridge.

Definition 1623.1 (Dyadic comparison carrier)**.** *A* $\mathsf{DyadicComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D_0, D_1, N_0, N_1, S, \Delta, Q, W, H, C, P, L)$$

with the following displayed rows:

1. D_0 *and* D_1 *are the two accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *source rows whose signed mantissas and unary exponents are visible to the packet.*
2. N_0 *and* N_1 *are the associated* $\mathsf{DyadicNormalForm}^{\uparrow}$ *readback rows, so the comparison is made after zero normalization and scale-minimality have been exposed.*
3. S *is the common-scale row, recording the finite unary exponent window used to compare the two normal forms.*
4. Δ *is the signed difference row at the common scale, read through the* $\mathsf{Int}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *surfaces already admitted by the dyadic core.*
5. Q *is the comparison classifier row, with outcomes negative, zero, or positive according to the sign of* Δ *inside the displayed finite window.*
6. W *is the finite window-stability ledger, recording how the classifier is preserved when both source rows are replayed through a larger common dyadic scale.*
7. H *records componentwise* hsame *transport for the source rows, normal-form rows, common-scale row, signed-difference row, classifier row, window ledger, continuation row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from the two input dyadic packets through normal-form readback, common-scale transport, signed-difference computation, classifier inspection, and window-stability export.*
9. *P is the* Pkg *provenance over* BMark, DyadicRatCore$^{\uparrow}$, DyadicNormalForm$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the rows $D_0, D_1, N_0, N_1, S, \Delta, Q, W, H, C, P, L$. It has no coordinate for host rational trichotomy, quotient rational equality, completed real order, located-cut data, ambient completeness, or a standard bridge.

Theorem 1623.2 (Dyadic comparison NameCert obligations)**.** *For every accepted* DyadicComparison$^{\uparrow}$ *carrier $K = (D_0, D_1, N_0, N_1, S, \Delta, Q, W, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two dyadic source rows, two normal-form readback rows, common-scale row, signed-difference row, comparison classifier, window-stability ledger, transport, continuation, provenance, and local naming row;*
2. *source compatibility, requiring D_0, D_1 to be accepted* DyadicRatCore$^{\uparrow}$ *packets and N_0, N_1 to be their displayed* DyadicNormalForm$^{\uparrow}$ *readbacks;*
3. *common-scale exactness, requiring S to transport the two mantissa-exponent pairs to one finite unary scale before Δ is inspected;*
4. *classifier exactness, requiring Q to read only the sign of the displayed finite difference Δ;*
5. *window stability, requiring W, H, C to preserve the classifier under componentwise* hsame *transport and displayed* Cont *replay into larger common dyadic windows;*
6. *regular-rational handoff, requiring* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *consumers to read only the stable classifier and its dyadic source route;*
7. *ledger refusal, excluding host rational trichotomy, quotient rational equality, completed real order, located cuts, ambient completeness, and standard-bridge data.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1623.1. The projection exposes exactly the two dyadic source rows, the two normal-form rows, the common-scale row, the signed-difference row, the classifier row, the window-stability ledger, and the structural rows H, C, P, L.

Source compatibility follows from the attachments to chapter 291 and chapter 1613. The packet never compares hidden representatives: it first reads the accepted dyadic core rows and then reads their zero-normalized, scale-minimal normal forms.

Common-scale exactness is the row S. Once both normal forms are transported to that row, the signed difference Δ is a finite integer-rational readback, and the classifier Q can inspect only its displayed sign.

Replay C and then transport by H. This preserves the same source rows, normal forms, common scale, signed difference, and classifier before any StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, or Real$^{\uparrow}$ consumer can inspect the packet.

The exclusions are immediate from the carrier inventory. Since host trichotomy, quotient equality, completed real order, located cuts, ambient completeness, and bridge rows are absent, the local NameCert surface cannot export them. □

Theorem 1623.3 (Dyadic comparison window stability)**.** *Let $K = (D_0, D_1, N_0, N_1, S, \Delta, Q, W, H, C, P, L)$ be an accepted* $\mathsf{DyadicComparison}^{\uparrow}$ *carrier. If S' is a displayed common dyadic scale extending S and the two normal-form rows are replayed into S' through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{DyadicNormalForm}^{\uparrow}$ *route, then the comparison classifier read at S' agrees with Q under componentwise* hsame *transport. Thus enlarging the finite comparison window changes neither the negative, zero, nor positive outcome exposed by the local* $\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$ *row.*

Proof. Start with the accepted carrier and the larger displayed scale S'. The continuation row C supplies the route from D_0, D_1 through N_0, N_1 to the original common scale S, while the hypothesis supplies the same route into S'.

The normal-form rows are scale-minimal by chapter 1613. Transporting both rows from S to S' multiplies both displayed mantissas by the same visible power of 2, so the signed difference row is multiplied by that same positive dyadic factor.

The sign of the finite difference is therefore unchanged. The zero case remains zero, the negative case remains negative, and the positive case remains positive because no row in the route introduces a hidden quotient or host-level comparison principle.

Apply the window-stability ledger W and componentwise transport H. They identify the classifier read at S' with the original row Q, and the local naming row L exports exactly that stable classifier. □

Theorem 1623.4 (Dyadic comparison RegSeqRat handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicComparison}^{\uparrow}$ *packet factors through the finite route*

$$D_0, D_1 \longmapsto N_0, N_1 \longmapsto S, \Delta, Q \longmapsto W, H, C, P, L.$$

Consequently the regular rational and real-seal phases receive a stable dyadic comparison row, not a direct order on completed real names, not quotient stream equality, not a located-cut comparison, and not an ambient completeness witness.

Proof. Take a consumer in the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ phase. Carrier admission exposes the local comparison row only through the displayed dyadic sources D_0, D_1.

The route then passes through the normal-form rows N_0, N_1. This is the point where zero normalization and scale-minimality are fixed before any regular rational readback can use the comparison.

Next the consumer reads the common-scale row S, the signed-difference row Δ, and the classifier row Q. By Definition 1623.3, any larger finite window used by a regular rational tail request returns the same classifier under hsame transport.

Finally the rows W, H, C, P, L package the stable comparison for the downstream phase. The displayed route is finite and contains no coordinate for direct completed-real order, quotient stream equality, located-cut comparison, selected host limit, ambient completeness, or a standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicComparison}^{\uparrow}$ packet is generated from two dyadic source rows, their normal-form readbacks, a common-scale row, a signed-difference row, a comparison classifier, a window-stability ledger, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$, seed)

Scope closed: Definition 1623.1, Definition 1623.2, Definition 1623.3, and Definition 1623.4 seed the finite dyadic comparison surface feeding StreamName, RegSeqRat, and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host rational trichotomy, quotient rational equality, completed real order, located cuts, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, common-scale exactness, classifier exactness, window stability, RegSeqRat handoff, and ledger refusal proofs for BEDC.Derived.DyadicComparisonUp.

Chapter 1624

A Concrete Naming Certificate for $\mathsf{RealUniformStructure}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges constructive analysis, the $\mathsf{RealUniformStructure}^{\uparrow}$ packet names the uniformity induced by the displayed $\mathsf{Real}^{\uparrow}$ metric surface through finite metric-ball entourage requests. It sits between the endpoint and seal row of chapter 38, the distance witness row of chapter 57, the entourage discipline of chapter 339, and the completion-facing Cauchy-filter route of chapter 333. The chapter records the Real-induced uniform structure as a finite NameCert boundary; it does not assert completeness of $\mathsf{Real}^{\uparrow}$ or choose limits of Cauchy data.

Definition 1624.1 (Real uniform-structure carrier)**.** *A* $\mathsf{RealUniformStructure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and seal row whose named points supply the source of every ball request.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance witness row read from the Real metric surface before any entourage is generated.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row generated by positive dyadic radius requests around the displayed metric balls.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row for completion-facing consumers that read Cauchy data through the induced entourage surface.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger used by the metric-ball generator.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by Real and Cauchy-filter consumers.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
8. *H records componentwise* hsame *transport for the Real, metric, uniform, Cauchy-filter, dyadic, window, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real endpoint request through distance, positive-radius selection, ball-entourage generation, Cauchy-filter compatibility, finite windows, and regular readback.*

10. *P is the* Pkg *provenance over* $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformStructure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, U, F, D, S, Q, H, C, P, N$. The carrier has no coordinate for quotient real equality, selected limits, trichotomy, countable choice, open-cover compactness, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureTasteGate single carrier alignment`

Theorem 1624.2 (Real uniform-structure NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *carrier $U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealUniformStructure}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Real endpoint and seal row, metric distance witness, uniform entourage row, Cauchy-filter compatibility row, dyadic radius ledger, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local naming row;*
2. *metric-ball entourage generation, requiring every entourage request in U to be generated from M by a positive dyadic radius read from D around endpoints displayed by R;*
3. *Cauchy-filter compatibility, requiring every completion-facing Cauchy-filter read in F to consume the induced entourage row U before inspecting stream windows or regular rational readbacks;*
4. *classifier transport, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the route R, M, D, U, F, S, Q;*
5. *ledger non-escape, requiring P and N to export no quotient real equality, selected limit, trichotomy, countable-choice selector, open-cover compactness, or ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1624.1. The projection exposes exactly the Real endpoint and seal row, the metric distance witness, the generated uniform entourage row, the Cauchy-filter compatibility row, the dyadic radius ledger, the finite-window and readback rows, and the structural rows H, C, P, N.

Metric-ball entourage generation is forced by the displayed route. An entourage request must first read endpoints from R, then read the distance witness in M, and then choose a positive dyadic radius from D before it can be recorded in the uniform row U.

Cauchy-filter compatibility is downstream of the induced uniformity. The row F is available only as a consumer of U, and the finite observations S and Q are read after that uniform row has been displayed.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, and Cont replay in C repeats the same route R, M, D, U, F, S, Q for every accepted request.

Finally, provenance is exhausted by P and N. Since quotient real equality, selected limits, trichotomy, countable choice, open-cover compactness, and ambient Real completeness are not rows of U_{Real}, the local NameCert surface cannot export them. □

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier namecert obligations`

Theorem 1624.3 (Real metric uniformity handoff)**.** *Every completion-facing or Cauchy-filter consumer of an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *carrier reads the Real uniformity only through the displayed rows*

$\mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{UniformSpace}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Nam}$

Consequently the packet exports no quotient real equality, selected limit, trichotomy, countable-choice selector, open-cover compactness, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier metric uniformity handoff`

Proof. Begin with a completion-facing or Cauchy-filter consumer. Carrier admission gives the consumer access to the local packet only through $R, M, U, F, D, S, Q, H, C, P, N$.

The Real metric uniformity is therefore read through the displayed metric-ball route. The Real row R supplies endpoint names, the metric row M supplies distance evidence, the dyadic row D supplies positive radii, and the uniform row U records the generated entourage.

Any Cauchy-filter use is then a downstream read of F. The window row S and the regular rational row Q expose finite observations only after U and F have been replayed.

The remaining rows do not add mathematical primitives. The BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows transport, replay, source, and name the same finite route, so the packet has no separate place from which quotient real equality, selected limits, trichotomy, countable choice, open-cover compactness, or ambient Real completeness can be exported. □

Theorem 1624.4 (Real uniform-structure L10 finite-source gate)**.** *Every L10 consumer of an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *carrier must inspect the* $\mathsf{Real}^{\uparrow}$ *endpoint and seal row, the* $\mathsf{Met}^{\uparrow}$ *distance row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before any* $\mathsf{UniformSpace}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff is available. No quotient real equality, selected limit, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness row is exported by this gate.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier l10 gate`

Proof. Project the carrier rows of Definition 1624.1. An L10 consumer can see the packet only through $R, M, U, F, D, S, Q, H, C, P, N$.

The finite-source gate is the displayed order of inspection. The Real endpoint and seal row R supplies named endpoints, M supplies distance observations, D supplies positive dyadic radii, S supplies finite windows, and Q supplies regular rational readback.

Only after those rows have been displayed can the existing metric-uniformity route of Definition 1624.3 expose U or F. The NameCert obligations of Definition 1624.2 then force the same route through H, C, P, N.

Since quotient real equality, selected limits, open-cover compactness, and $\mathsf{Real}^{\uparrow}$ completeness are not carrier rows, the L10 gate has no coordinate from which to export them. □

Theorem 1624.5 (Real uniform-structure located-filter compatibility)**.** *Every* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *consumer of the Real-induced uniformity must read the* $\mathsf{Real}^{\uparrow}$ *endpoint row* R*, metric row* M*, dyadic radius ledger* D*, uniform entourage row* U*, Cauchy-filter row* F*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*, and structural rows* H, C, P, N *before located-tail data is interpreted.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier located filter compatibility`

Proof. Project the carrier of Definition 1624.1. The Real-induced uniformity is available only through $R, M, U, F, D, S, Q, H, C, P, N$, so a located-filter consumer has no independent row from which to interpret located-tail data.

Use the NameCert obligations of Definition 1624.2. Metric-ball entourage generation forces the route through R, M, D, U, and Cauchy-filter compatibility forces the completion-facing filter row F to consume that entourage row before stream windows or rational readbacks are inspected.

Finally apply the L10 finite-source gate of Definition 1624.4. The rows S, Q, H, C, P, N preserve and name the same finite route, so a $\mathsf{LocatedCauchyFilter}^{\uparrow}$ consumer reads the Real-induced uniformity without receiving quotient real equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1624.6 (Real uniform-structure Cauchy-filter basis)**.** *Every accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet exports a* $\mathsf{CauchyFilter}^{\uparrow}$ *basis only through the* $\mathsf{Real}^{\uparrow}$ *endpoint and seal rows,* $\mathsf{Met}^{\uparrow}$ *distance witnesses, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radii, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The Cauchy-filter row is downstream of the metric-ball entourage row, and it cannot export quotient equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier cauchy filter basis`

Proof. Project the carrier of Definition 1624.1. The available coordinates are exactly $R, M, U, F, D, S, Q, H, C, P, N$, so a Cauchy-filter basis read starts from displayed Real endpoints, metric distance data, positive dyadic radii, stream windows, and regular rational readback.

Use the NameCert obligations of Definition 1624.2. They force every entourage request in U to be generated from the metric row M and positive dyadic radius ledger D around endpoints displayed by R.

The row F is therefore downstream of the metric-ball entourage row U. Its finite-window and regular-readback inspections pass through S and Q, while H, C, P, N transport, replay, source, and name the same route.

Since quotient equality, selected limits, and $\mathsf{Real}^{\uparrow}$ completeness are not rows of the carrier, the exported Cauchy-filter basis has no coordinate from which to supply them. □

Theorem 1624.7 (Real uniform-structure cofinal-refinement compatibility)**.** *Suppose an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet supplies its* $\mathsf{CauchyFilter}^{\uparrow}$ *route through a metric-ball entourage basis, and suppose an accepted* $\mathsf{FilterRefinement}^{\uparrow}$ *packet replaces that displayed basis by a cofinal refinement. Then every completion-facing Real-uniform consumer still reads through the same* $\mathsf{Real}^{\uparrow}$ *endpoint row,* $\mathsf{Met}^{\uparrow}$ *distance row, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger,* $\mathsf{CauchyFilter}^{\uparrow}$ *row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The refined route does not export quotient real equality, selected limits, countable choice, quotient filters, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier filter refinement compatibility`

Proof. Start with the Cauchy-filter basis of Definition 1624.6. It forces the Real-uniform route to expose the endpoint row R, metric row M, positive-radius ledger D, generated entourage row U, Cauchy-filter row F, finite windows, regular readbacks, and structural rows before any completion-facing consumer reads the basis.

Apply the Cauchy-preservation theorem for FilterRefinement$^{\uparrow}$, Definition 1626.2. The cofinal refinement replaces a displayed base request by a target-base request and a finite comparison row, while preserving the Cauchy route through the source basis.

Next use the completion route of Definition 1626.3. A completion-facing consumer receives only the refined finite base row, the cofinal refinement row, optional reverse comparison data when needed, and the transport, replay, provenance, and local naming rows.

Composing the two routes leaves the Real-uniform endpoint and metric data unchanged. The refinement acts on the displayed Cauchy-filter basis, so the consumer still reaches the same R, M, D, F, H, C, P, N surface and gains no coordinate for quotient real equality, selected limits, choice, quotient filters, or ambient completeness. □

Theorem 1624.8 (Real uniform-structure subbasis-completion handoff)**.** *Every completion-facing* RealUniformStructure$^{\uparrow}$ *consumer whose* CauchyFilter$^{\uparrow}$ *row is generated from an accepted* CauchyFilterSubbasis$^{\uparrow}$ *packet must read the* Real$^{\uparrow}$ *endpoint rows,* Met$^{\uparrow}$ *distance rows,* DyadicRatCore$^{\uparrow}$ *radius ledger, generated* UniformSpace$^{\uparrow}$ *entourage row, the* CauchyFilterSubbasis$^{\uparrow}$ *generated-base row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, and the terminal* Real$^{\uparrow}$ *seal before any completion use. The handoff exports no quotient filters, selected limits, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureSubbasisCompletionHandoff`

Proof. Start with the Cauchy-filter basis theorem Definition 1624.6. It forces the Real-uniform consumer to read endpoint rows R, metric data M, positive-radius ledger D, generated entourage row U, Cauchy-filter row F, windows S, readback Q, and structural rows before a completion-facing basis is available.

Apply the cofinal-refinement compatibility of Definition 1624.7. Cofinal replacement of the displayed basis preserves the same Real endpoint, metric, radius, Cauchy-filter, transport, replay, provenance, and local naming surface.

Now use the CauchyFilterSubbasis$^{\uparrow}$ generated-base route, Definition 4232.4 and Definition 4232.6. These force the subbasis-generated filter to expose finite intersections, generated events, the generation ledger, and the generated FilterBase$^{\uparrow}$ row before the CauchyFilter, StreamName, DyadicRatCore, RegSeqRat, and Real seal rows are read.

Composing the Real-uniform route with the subbasis route leaves a single finite handoff: Real endpoints and metric distances generate entourages, the subbasis packet supplies the cofinal generated base, and the windows, rational readback, and Real seal are inspected before completion use.

Since neither packet carries quotient filters, selected limits, countable choice, host equality, or ambient Real completeness, the combined handoff cannot export those objects. □

Theorem 1624.9 (Real uniform-structure subbasis refinement envelope)**.** *Every completion-facing* RealUniformStructure$^{\uparrow}$ *consumer whose* CauchyFilter$^{\uparrow}$ *row is generated from a* CauchyFilterSubbasis$^{\uparrow}$ *packet and then replaced by a displayed* FilterRefinement$^{\uparrow}$ *row still factors through Real endpoints,* Met$^{\uparrow}$ *distance,* DyadicRatCore$^{\uparrow}$ *radius data,* UniformSpace$^{\uparrow}$ *entourage generation, generated finite subbasis base,* CauchyFilter$^{\uparrow}$ *row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, and the terminal* Real$^{\uparrow}$ *seal. The envelope exports no quotient filters, selected limits, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier subbasis refinement envelope`

Proof. Start with the subbasis-completion handoff of Definition 1624.8. It forces the Real-uniform consumer to read Real endpoints, metric distances, dyadic radii, generated entourages, generated subbasis-base data, windows, readbacks, and the terminal Real seal before completion use.

Apply cofinal-refinement compatibility, Definition 1624.7. The displayed FilterRefinement$^{\uparrow}$ row may replace the generated base by a cofinal base request, but it preserves the same Real endpoint, metric, radius, Cauchy-filter, transport, replay, provenance, and local naming surface.

Now use the CauchyFilterSubbasis$^{\uparrow}$ route. The generated-base cofinality theorem Definition 4232.4 and the refinement stability theorem Definition 4232.6 keep finite subbasis tokens, finite intersections, generated events, the generation ledger, and the generated FilterBase$^{\uparrow}$ row visible before downstream Cauchy-filter and completion rows are read.

Composing these three routes leaves a single envelope through Real endpoints, Metric distance, DyadicRatCore radius data, UniformSpace entourages, generated finite subbasis bases, CauchyFilter rows, StreamName windows, RegSeqRat readbacks, and the Real seal. None of the displayed packets carries quotient filters, selected limits, countable-choice schedules, host real equality, or ambient Real completeness, so the envelope exports none of those objects. □

Theorem 1624.10 (Real uniform-structure finite-intersection subbasis)**.** *Any finite intersection of displayed* RealUniformStructure$^{\uparrow}$ *subbasis rows generated from an accepted* CauchyFilterSubbasis$^{\uparrow}$ *packet still factors through the same* Real$^{\uparrow}$ *endpoint row,* Met$^{\uparrow}$ *distance row, positive* DyadicRatCore$^{\uparrow}$ *radius ledger, generated* UniformSpace$^{\uparrow}$ *entourage row,* CauchyFilterSubbasis$^{\uparrow}$ *generated-base row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The finite-intersection route exports no selected limits, quotient filters, open-cover compactness, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureFiniteIntersectionSubbasis`

Proof. Start with the subbasis-completion handoff of Definition 1624.8. It forces every completion-facing subbasis consumer to read the Real endpoints, metric distances, positive dyadic radius ledger, generated entourage row, generated-base row, finite windows, regular readbacks, and structural rows before completion use.

Apply the completion entourage basis theorem, Definition 1624.14. The Real-uniform basis route remains attached to the same endpoint, metric, positive-radius, generated-entourage, Cauchy-filter, window, readback, transport, replay, provenance, and local naming surface.

Now use the generated-base and cofinality rows of CauchyFilterSubbasis$^{\uparrow}$, namely Definition 4232.4 and Definition 4232.6. A finite intersection is read as a displayed generated-base event, not as a new filter quotient or selected completion object.

Composing the three routes leaves the same finite Real-uniform surface. Since none of the displayed packets contains selected limits, quotient filters, open-cover compactness, countable choice, host equality, or ambient Real completeness, the finite-intersection route cannot export those objects. □

Theorem 1624.11 (Real uniform-structure cover-read handoff)**.** *Every compact-uniform or finite-cover consumer of an accepted* RealUniformStructure$^{\uparrow}$ *packet must read the* Real$^{\uparrow}$ *endpoint row,* Met$^{\uparrow}$ *distance row, positive* DyadicRatCore$^{\uparrow}$ *radius ledger, generated* UniformSpace$^{\uparrow}$ *entourage row,* CauchyFilter$^{\uparrow}$ *compatibility row,* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can hand data to* CompactUniformCoverRead$^{\uparrow}$ *or* UniformModulus$^{\uparrow}$*. The handoff exports no quotient real equality, selected limits, open-cover compactness, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCoverReadHandoff`

Proof. Project the carrier of Definition 1624.1. The compact-uniform consumer can see only the rows $R, M, U, F, D, S, Q, H, C, P, N$, so its cover read starts from named Real endpoints, metric distances, positive dyadic radii, generated entourages, Cauchy-filter compatibility, finite windows, and regular rational readbacks.

Apply the Cauchy-filter basis theorem Definition 1624.6. It places the Cauchy-filter row downstream of the metric-ball entourage row generated from R, M, D, and it keeps the window and readback rows S, Q attached to the same basis route.

Now use the cofinal-refinement compatibility of Definition 1624.7. A finite-cover consumer may refine the displayed basis, but the refinement remains attached to the same Real endpoint, metric, dyadic-radius, Cauchy-filter, and structural rows.

Therefore a $\mathsf{CompactUniformCoverRead}^{\uparrow}$ or $\mathsf{UniformModulus}^{\uparrow}$ handoff receives exactly the finite Real-uniform route already displayed by the packet. Since no carrier coordinate stores quotient real equality, selected limits, open-cover compactness, countable choice, or ambient Real completeness, none of those objects can be exported by the cover-read handoff. □

Theorem 1624.12 (RealUniformStructure basis refinement uniqueness)**.** *Any cofinal refinement accepted by the* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet must still factor through the same Real endpoint row, Metric distance row, positive DyadicRatCore radius ledger, generated UniformSpace entourage row, CauchyFilter row, StreamName windows, RegSeqRat readbacks, and structural rows. Thus a refined basis preserves the unique Real-induced basis route and exports no quotient real equality, selected limit, countable choice, quotient filter, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier basis refinement uniqueness`

Proof. Begin with the L10 finite-source gate of Definition 1624.4. It forces every Real-uniform consumer to inspect the Real endpoint row, Metric distance row, positive dyadic radius ledger, StreamName window row, and RegSeqRat readback before any UniformSpace or CauchyFilter handoff is available.

Apply the Cauchy-filter basis theorem Definition 1624.6. The CauchyFilter row is downstream of the generated metric-ball entourage row, and the structural rows transport, replay, source, and name that same route.

Now use the cofinal-refinement compatibility theorem Definition 1624.7. A cofinal refinement may replace the displayed basis request by a finite refined request, but it preserves the Real endpoint, metric, dyadic radius, CauchyFilter, and structural rows.

Combining the three routes leaves only one admitted Real-induced basis path:

$$R, M, D, U, F, S, Q \longmapsto H, C, P, N.$$

Since no carrier coordinate stores quotient real equality, selected limits, countable choice, quotient filters, open-cover compactness, or ambient Real completeness, no cofinal refinement can export those objects. □

Theorem 1624.13 (Real uniform-structure RealEquality filter boundary)**.** *Every completion-facing Cauchy-filter comparison over an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet must pass through the same* $\mathsf{Real}^{\uparrow}$ *endpoint row,* $\mathsf{Met}^{\uparrow}$ *distance row, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and then the* $\mathsf{RealEquality}^{\uparrow}$ *classifier boundary before any completion read is available. The route exports no quotient real equality, selected limit, countable-choice schedule, quotient filter, open-cover compactness, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureRealEqualityFilterBoundary`

Proof. Begin with the L10 finite-source gate of Definition 1624.4. It forces a Real-uniform consumer to inspect the Real endpoint row, Metric distance row, positive dyadic radius ledger, finite StreamName windows, and RegSeqRat readbacks before a uniform or filter-shaped read can occur.

Apply the Cauchy-filter basis theorem Definition 1624.6. The CauchyFilter row is downstream of the metric-ball entourage generated from the same Real endpoint, metric, and positive dyadic radius data.

Use the basis-refinement uniqueness theorem Definition 1624.12. Any completion-facing refinement keeps the same Real-induced basis route and cannot replace it by a quotient filter or a selected completion datum.

Finally apply the RealEquality classifier and NameCert boundary of Definition 1636.2 and Definition 1636.3. A Cauchy-filter comparison can read equality only through displayed finite windows, RegSeqRat readbacks, dyadic tolerance data, and the terminal Real boundary, so no hidden host equality or ambient completeness principle enters before the completion read. □

Theorem 1624.14 (Real uniform-structure completion entourage basis)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet reads an entourage basis only through the* $\mathsf{Real}^{\uparrow}$ *endpoint row, the* $\mathsf{Met}^{\uparrow}$ *distance row, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, the generated* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the structural rows* H, C, P, N*. The exported basis does not supply quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCompletionEntourageBasis`

Proof. Start with the NameCert obligations of Definition 1624.2. They admit exactly the rows $R, M, U, F, D, S, Q, H, C, P, N$, with every entourage request generated from Real endpoints, metric distance data, and a positive dyadic radius.

Apply the Cauchy-filter basis theorem Definition 1624.6. It places the $\mathsf{CauchyFilter}^{\uparrow}$ row downstream of the generated metric-ball entourage row and keeps the StreamName and RegSeqRat readbacks downstream of that same basis.

Finally use the cofinal-refinement compatibility of Definition 1624.7. A completion-facing consumer may replace a displayed basis by a cofinal finite refinement, but the route still reads through $R, M, D, U, F, S, Q, H, C, P, N$ and gains no coordinate for quotient real equality, selected limits, quotient filters, choice, open-cover compactness, or ambient Real completeness. □

Theorem 1624.15 (Real uniform-structure completion entourage subbasis)**.** *Every completion-facing subbasis request for an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet factors through the displayed Real endpoint row* R*, Metric row* M*, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger* D*, UniformSpace entourage row* U*, CauchyFilter row* F*, StreamName window row* S*, RegSeqRat readback row* Q*, and the structural rows* H, C, P, N *before any completion consumer reads it. Thus a subbasis request is only a finite entry point into the completion entourage basis; it cannot bypass the finite entourage basis, the positive-radius refinement route, or the Cauchy-filter basis rows.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCompletionEntourageSubbasis`

Proof. Start with the completion entourage basis theorem Definition 1624.14. It already orders every completion-facing entourage read through $R, M, D, U, F, S, Q, H, C, P, N$ before any completion consumer can inspect the basis.

Next apply the finite-entourage basis of Definition 1624.16. A subbasis request may name one generating metric-ball shape, but that shape is admitted only as a finite read of the same Real endpoint, metric, positive-radius, uniform, Cauchy-filter, window, readback, and structural rows.

Finally use the positive-radius entourage refinement of Definition 1624.17. Refining a subbasis request can narrow the positive radius in D, but it cannot add quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Therefore every completion-facing subbasis read remains a finite entrance to the displayed entourage and Cauchy-filter basis route. □

Theorem 1624.16 (Real uniform-structure finite entourage basis)**.** *Every accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet generates its entourage basis from finite reads of the* $\mathsf{Real}^{\uparrow}$ *endpoint row R, the* $\mathsf{Met}^{\uparrow}$ *distance row M, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger D, the generated* $\mathsf{UniformSpace}^{\uparrow}$ *row U, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the structural rows H, C, P, N. A completion-facing or Real-uniform consumer receives no basis element outside those displayed rows.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureFiniteEntourageBasis`

Proof. Start with the NameCert obligations of Definition 1624.2. They admit exactly the Real endpoint, metric distance, positive radius, uniform entourage, Cauchy-filter, finite-window, readback, transport, replay, provenance, and local naming rows.

Apply the metric-uniformity handoff of Definition 1624.3. It forces every entourage request to read endpoints from R, distance data from M, and a positive dyadic radius from D before the generated uniform row U is public.

Use the Cauchy-filter basis theorem Definition 1624.6. The row F is downstream of the metric-ball entourage row, while S and Q provide only finite windows and regular rational readbacks for that same basis.

Finally compose with the completion-entourage basis of Definition 1624.14. Completion-facing consumers may inspect the displayed basis, but the structural rows H, C, P, N only transport, replay, source, and name it. Thus no basis element is generated outside $R, M, D, U, F, S, Q, H, C, P, N$. □

Theorem 1624.17 (Real uniform-structure positive-radius entourage refinement)**.** *When an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet exposes two positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius rows in the same D ledger and the second is displayed as a refinement of the first, every refined entourage read still factors through the same* $\mathsf{Real}^{\uparrow}$ *endpoint row,* $\mathsf{Met}^{\uparrow}$ *distance row,* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row,* $\mathsf{CauchyFilter}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and structural rows. The refined request may narrow the finite metric-ball basis but cannot add quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructurePositiveRadiusEntourageRefinement`

Proof. Start with the NameCert obligations of Definition 1624.2. Both positive radii are read from the same D ledger, so the refinement remains attached to the displayed Real endpoint and Metric distance rows.

Apply the metric-uniformity handoff of Definition 1624.3. It sends the refined positive-radius request to the generated entourage row U without changing the endpoint row R or the distance row M.

Use the Cauchy-filter basis theorem Definition 1624.6 and the finite-entourage basis of Definition 1624.16. The refined request may narrow the finite metric-ball basis, but it remains downstream of $R, M, D, U, F, S, Q, H, C, P, N$.

Since none of those rows stores quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient Real completeness, no positive-radius refinement can export those objects. □

Definition 1624.18 (Real uniform-structure entourage shrink row)**.** *For an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet* $U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N)$*, an entourage-shrink row is a finite* BHist *entry*

$$\mathrm{shrink}(r, r', W)$$

read from the same positive-radius ledger D*, where* r *is the displayed positive Real uniform entourage radius,* r' *is a smaller positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *radius row, and* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window on which the metric-ball request is inspected. The row is admitted only when the generated entourage for* r' *is recorded in the same* $\mathsf{UniformSpace}^{\uparrow}$ *row* U *and is replayed through the same* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row* F*.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureEntourageShrink route closure`

Theorem 1624.19 (Real uniform-structure entourage shrink stability)**.** *Let* $U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N$ *and* $U'_{\mathsf{Real}} = (R', M', U', F', D', S', Q', H', C', P', N')$ *be accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packets. If an entourage-shrink row* $\mathrm{shrink}(r, r', W)$ *is displayed for* U_{Real} *and the endpoint, metric, radius-ledger, entourage, Cauchy-filter, finite-window, and readback rows transport componentwise under* hsame*, then the transported packet admits the corresponding shrink row through* D', U', F', S', Q' *and the same metric-uniformity handoff. The shrink cannot add quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the shrink row of Definition 1624.18. Its data are only the displayed positive radius r, the smaller dyadic or regular-rational radius r', and the finite window W read through D, S, Q before the generated entourage is recorded in U.

Transport the displayed rows componentwise under hsame. The transported radius ledger D' carries the smaller positive radius row, while S' and Q' carry the finite inspection window and readback. The displayed Cont replay through C' keeps the shrink request downstream of the same endpoint and metric rows.

Finally apply the metric-uniformity handoff of Definition 1624.3 together with the positive-radius refinement of Definition 1624.17. These force the transported shrink to land in U' and to remain compatible with F'. Since the carrier still has no coordinate for quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient Real completeness, the transported shrink exports none of those objects. □

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureEntourageShrinkStability`

Theorem 1624.20 (Real uniform-structure Cauchy-entourage shrink induction)**.** *Let* $U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N)$ *be an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet, and suppose a finite chain of entourage-shrink rows is displayed from the same positive-radius ledger* D*. Then iterated shrink rows preserve the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis* F*, the positive-radius entourage refinement through* D *and* U*, and the completion-entourage basis before any Real completion consumer reads a limit-shaped claim.*

The induction exports no quotient real equality, selected limits, quotient filters, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier cauchy entourage shrink induction`

Proof. Project the shrink definition of Definition 1624.18. Each shrink row is a finite read of a displayed positive radius, a smaller dyadic or regular-rational radius, and a finite StreamName window before the generated entourage is recorded in U and replayed through F.

Use shrink stability, Definition 1624.19, for the induction step. Transporting one shrink row preserves the endpoint, metric, radius-ledger, entourage, Cauchy-filter, finite-window, and readback rows componentwise.

Now apply the finite-entourage basis of Definition 1624.16 and the completion-entourage basis of Definition 1624.14. These keep every later entourage inside the same route through $R, M, D, U, F, S, Q, H, C, P, N$.

Induction over the finite shrink chain therefore preserves the Cauchy-filter basis, the positive-radius refinement, and the completion-entourage basis together. Since the carrier contains no quotient real equality, selected limit, quotient filter, choice schedule, open-cover compactness, or ambient Real-completeness coordinate, no limit-shaped completion claim can be read before those displayed rows are visible. □

Theorem 1624.21 (Real uniform-structure located Cauchy-filter handoff)**.** *Every* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *packet must read the* $\mathsf{Real}^{\uparrow}$ *endpoint row, the* $\mathsf{Met}^{\uparrow}$ *row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the structural rows* H, C, P, N *before any located-tail interpretation is available. The handoff exports no quotient filter, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier located cauchy filter handoff`

Proof. Start with the Cauchy-filter basis theorem Definition 1624.6. It places every filter-facing read downstream of the Real endpoint row, Metric row, positive dyadic radius ledger, generated entourage row, CauchyFilter row, StreamName windows, RegSeqRat readback, and the structural rows.

Apply the L10 finite-source gate Definition 1624.4. A located-tail consumer cannot inspect a tail predicate before the Real endpoint, metric distance, radius ledger, finite window, and regular readback rows have been displayed.

Finally use the positive-radius refinement route Definition 1624.17. Refining the entourage radius preserves the same $R, M, D, U, F, S, Q, H, C, P, N$ surface and adds no quotient filter, selected limit, choice schedule, host equality, or ambient completeness coordinate. Thus located-tail interpretation is a consumer of the displayed Real-uniform packet, not an independent completion principle. □

Theorem 1624.22 (Real uniform-structure obligation-closure package)**.** *For every accepted* $\mathsf{RealUniformStructure}^{\uparrow}$ *carrier*

$$U_{\mathsf{Real}} = (R, M, U, F, D, S, Q, H, C, P, N),$$

the displayed finite rows close carrier admission, metric-ball entourage generation, Cauchy-filter compatibility, classifier transport, and ledger non-escape for the Real-induced uniformity boundary. Every completion-facing, located-filter, or L10 consumer reads the packet through the same $R, M, U, F, D, S, Q, H, C, P, N$ *surface, and no quotient real equality, selected limit, countable-choice schedule, open-cover compactness, quotient filter, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row is exported.*

Lean4 ✓ `BEDC.Derived.RealUniformStructureUp.RealUniformStructureCarrier obligation closure package`

Proof. Carrier admission is the carrier definition, Definition 1624.1. It exposes exactly the Real endpoint row, Metric row, generated UniformSpace row, CauchyFilter row, DyadicRatCore radius ledger, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local naming rows.

Metric-ball entourage generation and the L10 inspection order are already fixed by Definition 1624.2, Definition 1624.3, and Definition 1624.4.

Together these theorems force every entourage request to pass through R, M, D, U and every finite-source consumer to display R, M, D, S, Q before using U or F.

Cauchy-filter compatibility is supplied by Definition 1624.6 and Definition 1624.21. These theorems place every Cauchy or located-tail read downstream of the same generated entourage surface and keep the finite window and readback rows visible.

Finally, classifier transport and ledger non-escape are the structural part of the same carrier. The rows H, C, P, N transport, replay, source, and name only the displayed route.

The packet therefore has no coordinate from which quotient real equality, selected limits, countable choice, open-cover compactness, quotient filters, or ambient Real completeness can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformStructure}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformStructure}^{\uparrow}$ packet is generated from a RealUp endpoint and seal row, a MetricUp distance witness row, a UniformSpace entourage row generated by positive DyadicRatCore radius requests, a CauchyFilter compatibility row, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformStructure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1624.1, Definition 1624.2, and Definition 1624.3 seed the Real-induced uniformity boundary for completion-facing and Cauchy-filter consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformStructure}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealUniformStructureUp`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, countable-choice selectors, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires carrier admission, metric-ball entourage generation, Cauchy-filter compatibility, classifier transport, and ledger non-escape proofs for the same packet.

Chapter 1625

A Concrete Naming Certificate for $\mathsf{RealIntervalEnclosure}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealIntervalEnclosure}^{\uparrow}$ packet names finite lower and upper rational or dyadic windows around a $\mathsf{Real}^{\uparrow}$ name. It sits after the dyadic finite arithmetic surface of chapter 291, the finite observation gateway of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the endpoint-ledger interface of chapter 394. The packet records an enclosure read for one real name through displayed finite windows; it is not a host interval, quotient real equality, trichotomy, a selected endpoint pair, or a completeness principle.

Definition 1625.1 (Real interval enclosure carrier)**.** *A* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, S, Q, D, L, U, A, W, H, C, P, N)$$

with the following displayed rows:

1. *R is the accepted* $\mathsf{Real}^{\uparrow}$ *name whose finite enclosure is being read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row specifying the finite observations of the real name used by the packet.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose dyadic radius bounds the displayed readback error.*
5. *L and U are the lower and upper rational or dyadic bound rows displayed in the packet.*
6. *A is the admission row saying that the readback Q and tolerance D place the inspected real-name window between L and U.*
7. *W is the window-stability ledger, recording that later finite windows refine the same enclosure route instead of replacing it by a host endpoint choice.*
8. *H records componentwise* hsame *transport for the real name, stream window, regular readback, dyadic tolerance, lower bound, upper bound, admission row, stability ledger, continuation row, provenance row, and local naming row.*
9. *C records the displayed* Cont *replay from the real-name seal through the finite window, regular readback, dyadic tolerance, bound rows, admission row, and stability ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalEnclosure}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite enclosure packet. It has no coordinate for host interval endpoints, quotient real equality, trichotomy, a selected limit value, open-cover compactness, Heine–Borel compactness, or Real completeness.

Theorem 1625.2 (Real interval enclosure NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *carrier $E = (R, S, Q, D, L, U, A, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalEnclosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real-name seal, stream window, regular rational readback, dyadic tolerance, lower bound, upper bound, admission, stability, transport, continuation, provenance, and local naming rows;*
2. *source compatibility, requiring S, Q, D to be read through the same* $\mathsf{Real}^{\uparrow}$ *route named by R;*
3. *bound admissibility, requiring L and U to be displayed rational or dyadic rows rather than selected host endpoint values;*
4. *enclosure exactness, requiring A to witness that the inspected regular-rational window with tolerance D lies between L and U;*
5. *window stability, requiring W, H, C to preserve the enclosure under componentwise transport and later finite-window replay;*
6. *located-interval handoff, allowing a* $\mathsf{LocatedInterval}^{\uparrow}$ *consumer to read only the displayed bound rows and the same real-name window route;*
7. *ledger refusal, excluding Heine–Borel compactness, open-cover compactness, trichotomy, quotient real equality, selected limits, host endpoint values, and Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1625.1. The projection exposes exactly the real-name seal, stream window, regular-rational readback, dyadic tolerance, lower and upper bound rows, admission row, stability ledger, and structural rows H, C, P, N.

Source compatibility follows by reading S, Q, D through the sibling route of chapter 126, chapter 301, and chapter 291. Since all three rows are attached to the same R, no second real-name route is available.

Bound admissibility and enclosure exactness are the rows L, U, A. The lower and upper rows are displayed rational or dyadic data, and A binds them to the inspected readback and tolerance window rather than to host endpoint values.

Replay C and transport by H. The replay preserves the same finite route into later windows, while W records that those later windows refine the displayed enclosure service. A $\mathsf{LocatedInterval}^{\uparrow}$ consumer therefore receives the same lower and upper rows through the same finite route.

The exclusions are immediate from the carrier inventory. The packet contains no row for Heine–Borel compactness, open-cover compactness, trichotomy, quotient real equality, selected limits, host endpoint values, Real completeness, or a standard bridge. □

Theorem 1625.3 (Real interval enclosure RegSeqRat window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{LocatedInterval}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *carrier factors through the finite*

route

$$R \longmapsto S \longmapsto Q, D \longmapsto L, U, A \longmapsto W, H, C, P, N.$$

Consequently the consumer receives a finite enclosure row around a real-name window, not a direct host interval, not quotient equality of real names, not trichotomy, not selected endpoint values, and not a completeness witness.

Proof. Start with an accepted carrier $E = (R, S, Q, D, L, U, A, W, H, C, P, N)$. The only terminal real-name coordinate is R, and the only finite observation coordinate attached to it is the stream window S.

The regular-rational readback Q and dyadic tolerance row D are then read from that same window. Thus any regular-rational consumer reaches the bound rows only after passing through the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route.

The lower and upper rows L, U are admitted through A. A $\mathsf{LocatedInterval}^{\uparrow}$ consumer may use them as displayed interval data, but only together with the same admission row and the same real-name source route.

Finally W, H, C, P, N package the handoff. Stability, transport, continuation replay, provenance, and local naming keep the consumer inside the finite packet, so no host interval, quotient real equality, trichotomy, selected endpoint value, completeness theorem, or bridge row can be inserted into the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalEnclosure}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalEnclosure}^{\uparrow}$ packet is generated from a real-name seal, stream window, regular rational readback, dyadic tolerance, displayed lower and upper rational or dyadic bounds, an enclosure admission row, a window-stability ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealIntervalEnclosure}^{\uparrow}}$, seed)

Scope closed: Definition 1625.1, Definition 1625.2, and Definition 1625.3 seed the finite interval-enclosure surface around Real, StreamName, RegSeqRat, DyadicRatCore, and LocatedInterval consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealIntervalEnclosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, open-cover compactness, trichotomy, quotient real equality, selected limits, host endpoint values, Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, bound admissibility, enclosure exactness, window stability, located-interval handoff, and ledger refusal proofs for BEDC.Derived.RealIntervalEnclosureUp.

Chapter 1626

A Concrete Naming Certificate for $\mathsf{FilterRefinement}^{\uparrow}$

Following Kelley and Bourbaki on finer filters, cofinal subfilters, and refinement of directed bases, the $\mathsf{FilterRefinement}^{\uparrow}$ packet names the finite L10 bridge by which a completion route replaces one displayed filter base by a cofinal refinement. It sits after the finite base surface of chapter 1601, refines the Cauchy-filter route of chapter 333, and keeps the completion-facing consumers of chapter 1473, chapter 1506, and chapter 1569 on the displayed finite route. The packet records cofinal comparison rows only; it is not a quotient filter, not an ultrafilter, not a selected limit, not countable choice, and not an ambient completeness principle.

The same cofinal replacement row is consumed horizontally by the Real-induced uniformity route of chapter 1624 and by the generated-base route of chapter 4232. A $\mathsf{RealUniformStructure}^{\uparrow}$ packet and a $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ consumer both read the displayed $\mathsf{FilterRefinement}^{\uparrow}$ source, target, refinement, and finite generated-base rows before any completion-facing route sees the replacement basis.

Definition 1626.1 (FilterRefinement NameCert surface)**.** *A* $\mathsf{FilterRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, \rho, \lambda, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{FilterBase}^{\uparrow}$ *row naming the base packets through which the original* $\mathsf{CauchyFilter}^{\uparrow}$ *or completion-facing route is read.*
2. *T is the target* $\mathsf{FilterBase}^{\uparrow}$ *row naming the refined base packets that will be exposed to downstream consumers.*
3. *ρ is the cofinal refinement map. For each displayed target base packet it names a source base packet and the containment read from the target packet into that source packet.*
4. *λ is the reverse comparison ledger when present. It records the displayed return comparisons needed for equivalence of two routes, but the carrier may leave this row empty when only one-way refinement is asserted.*
5. *H records componentwise* hsame *transport for the source base, target base, cofinal refinement map, optional reverse ledger, continuation rows, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from an original base request through the cofinal refinement map and, when present, through the reverse comparison ledger.*

7. *P is the* Pkg *provenance over* $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, \rho, \lambda, H, C, P, N$. There is no coordinate for quotient filters, ultrafilters, selected limits, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementTasteGate single carrier alignment`

Theorem 1626.2 (FilterRefinement Cauchy preservation)**.** *Let $R = (S, T, \rho, \lambda, H, C, P, N)$ be an accepted* $\mathsf{FilterRefinement}^{\uparrow}$ *carrier. If a* $\mathsf{CauchyFilter}^{\uparrow}$ *route is Cauchy through the displayed source base row S, then the route obtained by replaying every base request through the cofinal refinement map ρ is Cauchy through the displayed target base row T.*

The preserved Cauchy route uses only the finite refinement display

$$T \xrightarrow{\rho} S \longmapsto H, C, P, N.$$

It does not use a quotient filter, ultrafilter, selected limit, countable-choice selector, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinement cauchy preservation`

Proof. Project the accepted carrier of Definition 1626.1. The source and target bases are displayed $\mathsf{FilterBase}^{\uparrow}$ rows, and the cofinal map ρ is a finite row of the same packet.

Start with a Cauchy request on the refined route. The target base packet requested from T is sent by ρ to a source base packet in S, together with the displayed containment read from the refined packet into the source packet.

Apply the Cauchy evidence already available on the source route. Because the source route is Cauchy through S, every precision or late-window request is discharged by a displayed source base packet and its $\mathsf{CauchyFilter}^{\uparrow}$ evidence.

Replay the result along the containment read recorded by ρ. The componentwise hsame row H transports the visible base, refinement, continuation, provenance, and naming coordinates, while the Cont row C replays the request through the displayed target packet.

Thus every refined request is answered by a finite target-row request followed by the finite source-row Cauchy witness. No excluded filter, limit, choice, or completeness coordinate is available in the carrier, so Cauchy preservation remains internal to S, T, ρ, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCauchyPreservation`

Theorem 1626.3 (FilterRefinement completion route)**.** *Let $R = (S, T, \rho, \lambda, H, C, P, N)$ be an accepted* $\mathsf{FilterRefinement}^{\uparrow}$ *carrier. Every* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer that receives the refined route sees only the finite target base row T, the cofinal refinement row ρ, the optional reverse comparison ledger λ when finite route equivalence is required, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FilterRefinement}^{\uparrow}}$ *row.*

Equivalently, completion-facing reads factor through

$$S \xleftarrow{\rho} T \longmapsto \lambda, H, C, P, N,$$

with λ omitted for one-way refinement claims. When present, λ records only the finite return comparisons needed to identify the displayed source and target routes; it is not quotient-filter equality, an ultrafilter row, a limit selector, a selected completion point, countable choice, host equality, or ambient completeness.

Proof. Begin with a completion-facing consumer. Its route is required to read the refined base row T before it can inspect Cauchy-filter or completion data, so the first public object is a displayed finite $\mathsf{FilterBase}^{\uparrow}$ row.

The row ρ supplies the only mandatory comparison back to the source route. When a consumer must compare the refined route with the original route, it reads the optional ledger λ as a finite list of return comparisons for route equivalence; otherwise the one-way cofinal refinement row is the entire comparison surface.

Transport and replay do not enlarge the route. The row H moves the listed coordinates componentwise, and C replays base requests through the same displayed refinement rows. The provenance row P names the allowed completion consumers, while N names the local certificate surface.

Consequently $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumers receive a finite refinement route and no hidden completion object. Quotient-filter equality, ultrafilter data, selected completion points, limit selectors, countable-choice selectors, host equality, and ambient completeness are not coordinates of R, and S, T, ρ, H, C, P, N remain the only mandatory route rows. □

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCompletionRoute`

Theorem 1626.4 (FilterRefinement reverse-ledger comparison)**.** *Let $R = (S, T, \rho, \lambda, H, C, P, N)$ be an accepted* $\mathsf{FilterRefinement}^{\uparrow}$ *carrier. When λ is present, it supplies only finite return comparisons from the source route back to the target route. Thus source and target completion-facing reads may be compared through $S, T, \rho, \lambda, H, C, P, N$ without quotient-filter equality, ultrafilter data, selected limits, countable choice, host equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCarrier reverse ledger comparison`

Proof. Project the carrier surface of Definition 1626.1. The rows S and T are displayed filter-base rows, ρ is the cofinal refinement map, and λ is explicitly the optional reverse comparison ledger.

Use the completion route theorem Definition 1626.3. It identifies λ as route-equivalence data read only when a finite comparison between the source and target completion-facing routes is required.

Apply Cauchy preservation and the Cauchy-basis consumer, Definition 1626.2 and Definition 1626.6. Both directions remain inside displayed $\mathsf{FilterBase}^{\uparrow}$ and $\mathsf{CauchyFilterBase}^{\uparrow}$ rows, while H, C, P, N transport, replay, source, and name the same finite packet. No quotient filter, ultrafilter, selected limit, choice schedule, host equality, or ambient completeness row is available to the comparison. □

Theorem 1626.5 (FilterRefinement Cauchy base dependency)**.** *Every refinement accepted by* $\mathsf{FilterRefinement}^{\uparrow}$ *consumes the finite Cauchy filter base carrier of chapter 1915 before it may replay completion-facing cofinality. The refined route therefore exposes a* $\mathsf{CauchyFilterBase}^{\uparrow}$ *basis row, a cofinal comparison through ρ, and the same transport, replay, provenance, and local naming rows H, C, P, N before any* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer reads the refinement.*

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCarrier cauchy base dependency`

Proof. Start from the accepted carrier $R = (S, T, \rho, \lambda, H, C, P, N)$ of Definition 1626.1. The source and target rows are displayed filter-base rows, so a completion-facing refinement can only be consumed after the relevant Cauchy-filter base has been exposed.

Use the Cauchy-base carrier of chapter 1915 as the finite basis row read by the Cauchy route. The cofinal map ρ then sends target-base requests back to the source Cauchy base, while H and C transport and replay the same finite comparison.

Finally apply Definition 1626.3. It permits completion consumers to read only $T, \rho, \lambda, H, C, P, N$, so the Cauchy-base dependency is a displayed prerequisite for cofinality replay rather than a quotient filter, selected limit, countable-choice selector, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCauchyBaseDependency`

Theorem 1626.6 (FilterRefinement Cauchy-basis consumer)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer using a* $\mathsf{FilterRefinement}^{\uparrow}$ *route must first expose the* $\mathsf{FilterBase}^{\uparrow}$ *row* S*, the* $\mathsf{CauchyFilterBase}^{\uparrow}$ *row supporting the Cauchy route, and the cofinal comparison row* ρ *before any completion-facing read. The consumer then factors through the same transport, replay, provenance, and local naming rows* H, C, P, N*, with the optional return ledger* λ *read only when finite route equivalence is required.*

Lean4 ✓ `BEDC.Derived.FilterRefinementUp.FilterRefinementCauchyBasisConsumer`

Proof. Project the carrier $R = (S, T, \rho, \lambda, H, C, P, N)$ from Definition 1626.1. The source row S and target row T are displayed $\mathsf{FilterBase}^{\uparrow}$ rows, and ρ is the only mandatory cofinal comparison between them.

Use the Cauchy-base dependency theorem Definition 1626.5. It forces the Cauchy-filter route to expose its $\mathsf{CauchyFilterBase}^{\uparrow}$ basis before cofinality replay can be consumed.

Finally apply the completion route theorem Definition 1626.3. Completion-facing consumers therefore see the basis rows and ρ before they read any $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ boundary, and the structural rows H, C, P, N add no quotient filter, selected limit, countable choice, host equality, or ambient completeness coordinate. □

The completion-route theorem is read as a sibling dependency rule: a $\mathsf{FilterRefinement}^{\uparrow}$ packet consumes the $\mathsf{FilterBase}^{\uparrow}$ carrier of chapter 1601 and the $\mathsf{CauchyFilter}^{\uparrow}$ route of chapter 333 before any $\mathsf{UniformCompletion}^{\uparrow}$ consumer from chapter 1473, $\mathsf{MetricCompletion}^{\uparrow}$ consumer, or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer may read the refined route.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1626.1, Definition 1626.2, and Definition 1626.3 seed the finite filter-refinement bridge for Cauchy-filter and completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, ultrafilters, selected limits, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal refinement coverage, optional reverse comparison compatibility, Cauchy preservation, completion-route exposure, and ledger-refusal proofs for BEDC.Derived.FilterRefinementUp.

Constructive story: A $\mathsf{FilterRefinement}^{\uparrow}$ packet is generated from source and target FilterBase rows, a cofinal refinement map, an optional reverse comparison ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1627

A Concrete Naming Certificate for $\mathsf{LocatednessModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatednessModulus}^{\uparrow}$ packet names the finite budget by which a positive rational tolerance request is turned into displayed interval and real-name window data. It sits between the endpoint surface of chapter 394, the rational-cell ledger of chapter 354, the dyadic tolerance rows of chapter 291, the finite observation gateway of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The locatedness-modulus budget is consumed by chapter 1646 as the locatedness request row in the Real metric bracket route. The packet records an explicit locatedness service for constructive Real completion; it is not a host trichotomy, quotient real equality, selected endpoint functional, or ambient completeness theorem.

Definition 1627.1 (Locatedness modulus carrier)**.** *A* $\mathsf{LocatednessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (q, L, I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *q is the positive rational tolerance request supplied to the locatedness service.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint surface whose lower and upper endpoint rows are inspected under that request.*
3. *I is the finite* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger that records the rational windows admitted for the requested tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *or dyadic-interval refinement ledger that records the dyadic rows used to tighten the rational cells.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the endpoint names and the real-name route at precision q.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected stream window.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the readback and refinement budget have been displayed.*
8. *H records componentwise* hsame *transport for the tolerance request, located interval, rational cells, dyadic refinements, stream windows, readbacks, real-seal rows, continuation routes, provenance rows, and local naming row.*

9. *C records the displayed* Cont *replay from the tolerance request through the located interval, rational cells, dyadic refinement ledger, stream window, regular readback, and real-name seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatednessModulus}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite locatedness-budget packet. It has no coordinate for a host order trichotomy, quotient real equality, selected endpoints, a global apartness oracle, or ambient Real completeness.

Theorem 1627.2 (Locatedness modulus NameCert obligations)**.** *For every accepted* $\mathsf{LocatednessModulus}^{\uparrow}$ *carrier $M = (q, L, I, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatednessModulus}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission, requiring the displayed tolerance request, located interval, rational-cell ledger, dyadic refinement rows, stream windows, regular readbacks, real-seal rows, transport, continuation, provenance, and local naming data to inhabit one* BHist *packet;*

2. *tolerance-cell compatibility, requiring every rational cell in I to be admitted for the request q and for the endpoint surface L;*

3. *dyadic refinement stability, requiring every dyadic row in D to refine a displayed rational cell rather than replace it by a host endpoint choice;*

4. *window transport, requiring S, R, E, H, C to preserve the same tolerance request through finite endpoint windows and real-seal readbacks;*

5. *non-escape, excluding host trichotomy, quotient real equality, selected endpoints, global apartness choice, and ambient completeness from the exported certificate.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.TasteGate.LocatednessModulusTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.LocatednessModulusTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1627.1. The projection exposes exactly the tolerance request, located interval, rational cells, dyadic refinements, stream windows, readbacks, real-seal row, and structural rows H, C, P, N.

Carrier admission is the finite tuple M itself. Tolerance-cell compatibility is the assertion that the rational cells in I are displayed as the cells admitted for q and for the endpoint surface L.

Dyadic refinement stability reads the rows of D only through the rational cells in I. Thus a dyadic refinement tightens the displayed finite budget and cannot introduce a selected host endpoint.

Window transport is supplied by S, R, E, H, C. The stream window, regular readback, and real-seal replay preserve the same request route through componentwise transport.

The exclusions follow from the carrier inventory. Since M contains no coordinate for host trichotomy, quotient real equality, selected endpoints, a global apartness choice, or Real completeness, none can be exported by the local certificate. □

Theorem 1627.3 (Locatedness modulus window transport)**.** *Every finite consumer of an accepted* $\mathsf{LocatednessModulus}^{\uparrow}$ *carrier transports the tolerance request and endpoint windows through the route*

$$q \longmapsto L \longmapsto I, D \longmapsto S, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently a $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the same requested tolerance together with its finite endpoint-window route, not a new host comparison of completed real values.*

Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.LocatednessModulusCarrier window transport`

Proof. Start with an accepted carrier $M = (q, L, I, D, S, R, E, H, C, P, N)$. The only tolerance coordinate is q, and the only interval coordinate attached to it is the displayed $\mathsf{LocatedInterval}^{\uparrow}$ surface L.

The rational cells I and dyadic refinements D are then read as the finite response to that request. Their transport keeps each cell and refinement attached to q and L because no other request or interval row is present.

The stream window S and regular readback R inspect the endpoint data through that same route. The real-seal row E is reached only after those finite observations have been displayed.

Finally H, C, P, N package the transport. Componentwise hsame, continuation replay, provenance, and local naming keep the consumer inside the requested finite budget, so no host comparison of completed real values can be inserted. □

Theorem 1627.4 (Locatedness modulus Real-seal non-escape)**.** *For every accepted* $\mathsf{LocatednessModulus}^{\uparrow}$ *carrier* $M = (q, L, I, D, S, R, E, H, C, P, N)$*, every Real-facing read factors through the displayed tolerance request, located interval surface, rational-cell ledger, dyadic refinement rows, stream window, regular readback, real-seal row, transport row, continuation row, provenance row, and local naming row. It cannot export host trichotomy, quotient real equality, selected endpoints, a global apartness oracle, or ambient completeness.*

Proof. Read the Real-facing part of Definition 1627.1. The only real-name boundary is the seal row E, and it is preceded by the finite window S, readback R, dyadic refinement ledger D, rational cells I, located interval surface L, and tolerance request q.

Transport does not add a second Real-facing source. The componentwise rows H and displayed replay C move the same listed data through the same packet, while P and N record the package and local certificate provenance.

It remains to rule out non-displayed Real data. The carrier has no field for host trichotomy, quotient real equality, selected endpoints, a global apartness oracle, or ambient completeness; every Real-facing read is therefore a projection from M. □

Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.LocatednessModulusCarrier real seal non escape`

Theorem 1627.5 (Locatedness modulus window decision budget)**.** *For every accepted* $\mathsf{LocatednessModulus}^{\uparrow}$ *carrier* $M = (q, L, I, D, S, R, E, H, C, P, N)$*, a locatedness request at tolerance* q *is discharged only by the displayed route through* $L, I, D, S, R, E, H, C, P, N$*. Every rational-cell decision reads the dyadic refinement ledger* D *before the* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and every Real-facing output is sealed only after that budget is displayed. The route exports no host trichotomy, selected endpoints, quotient real equality, global apartness choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.LocatednessModulusObligationWindowDecisionBudget`

Proof. Project the carrier of Definition 1627.1. The tolerance request q is attached to the located interval L, rational-cell ledger I, dyadic refinement ledger D, StreamName window S, RegSeqRat readback R, Real seal E, and structural rows H, C, P, N.

Apply Definition 1627.3. It routes the request through L, then through the rational cells and dyadic refinements, and only then through S, R, E, H, C, P, N. Thus a rational-cell decision cannot bypass the displayed dyadic budget before the StreamName and RegSeqRat handoff.

Now apply Definition 1627.4. The Real-facing read factors through the same finite budget and reaches the seal row E only after the request, interval, cells, refinements, stream window, and readback are displayed.

The carrier has no coordinate for host trichotomy, selected endpoints, quotient real equality, global apartness choice, or ambient Real completeness. Therefore the locatedness request is discharged exactly by the displayed window decision budget and exports none of the excluded data. □

Theorem 1627.6 (Locatedness modulus public export)**.** *Every Real or completion consumer of an accepted* LocatednessModulus$^{\uparrow}$ *packet receives only the displayed tolerance request q, located interval surface L, rational-cell ledger I, dyadic-refinement ledger D,* StreamName$^{\uparrow}$ *window S,* RegSeqRat$^{\uparrow}$ *readback R,* Real$^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. This public export surface excludes host trichotomy, quotient real equality, selected endpoints, and ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LocatednessModulusUp.LocatednessModulusCarrier public export`

Proof. Project the accepted carrier of Definition 1627.1. Its public coordinates are exactly $q, L, I, D, S, R, E, H, C, P$

Apply the NameCert obligations of Definition 1627.2. They restrict certificate access to carrier admission, tolerance-cell compatibility, dyadic refinement stability, window transport, and non-escape.

Use Definition 1627.3 and Definition 1627.4. The first theorem fixes the consumer route through the displayed request, interval, rational cells, dyadic refinements, stream windows, readbacks, and structural rows; the second theorem prevents the Real-facing seal from exporting any non-displayed datum.

Finally apply the decision-budget theorem Definition 1627.5. A public consumer can only inspect the listed rows, so trichotomy, quotient equality, selected endpoints, and ambient Real completeness remain outside the exported surface. □

Closure status for NameCert$_{\text{LocatednessModulus}^{\uparrow}}$.

Constructive story: A LocatednessModulus$^{\uparrow}$ packet is generated from a rational tolerance request, a LocatedInterval$^{\uparrow}$ endpoint surface, RationalInterval$^{\uparrow}$ cells, DyadicRatCore$^{\uparrow}$ refinement rows, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, a Real$^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatednessModulus}^{\uparrow}}$, seed)

Scope closed: Definition 1627.1, Definition 1627.2, Definition 1627.3, and Definition 1627.4 seed the finite locatedness-budget surface for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{LocatednessModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient real equality, selected endpoints, global apartness choice, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires carrier admission, tolerance-window transport, dyadic refinement stability, locatedness decision budget, and Real-seal non-escape proofs for BEDC.Derived.LocatednessModulusUp.

Chapter 1628

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniformity}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges regular Cauchy names, the $\mathsf{RegularCauchyUniformity}^{\uparrow}$ packet names the dyadic-entourage uniformity on regular rational Cauchy names as finite BEDC rows. It reads the dyadic arithmetic core of chapter 291, the finite observation route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the ambient uniform-space interface of chapter 339, and the completion-facing route of chapter 1506. The packet is the finite dyadic-entourage surface used by Real and completion consumers; it is not quotient stream equality, not a selected limit, not host real equality, not countable choice, not open-cover compactness, and not Real completeness.

Definition 1628.1 (Regular Cauchy uniformity carrier)**.** *A* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *W is the finite window row recording the inspected stream prefix and the later common refinements admitted for the same regular Cauchy name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic radius used by the entourage request.*
4. *E is the dyadic-entourage row, asserting that two displayed regular rational windows are within the tolerance named by D at the inspected stage.*
5. *H records componentwise* hsame *transport for the regular rational name, window, dyadic tolerance, entourage row, continuation row, provenance row, and local naming row.*
6. *C records the displayed* Cont *replay from a regular Cauchy request through finite windows, dyadic tolerance naming, entourage admission, and the downstream Real or completion handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these eight rows form one finite dyadic-entourage uniformity packet. It contains no coordinate for quotient stream equality, selected limits, host real equality, countable choice, open-cover compactness, or Real completeness.

Lean4 ✓ BEDC.Derived.RegularCauchyUniformityUp.TasteGate.RegularCauchyUniformityTasteGate
single carrier alignment

Theorem 1628.2 (Regular Cauchy uniformity NameCert obligations)**.** *For every accepted* RegularCauchyUniformity *carrier* $U = (R, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational name, finite window, dyadic tolerance, entourage, transport, continuation, provenance, and local naming rows;*
2. *dyadic entourage admission, requiring* E *to be read only after the window row* W *and tolerance row* D *have been displayed;*
3. *window reindexing, requiring later common finite* $\mathsf{StreamName}^{\uparrow}$ *refinements to preserve the same regular rational source row* R*;*
4. *Real seal handoff, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to enter through the displayed route* R, W, D, E, H, C, P, N*;*
5. *transport, requiring* H *to identify only the listed rows componentwise under* hsame*;*
6. *replay, requiring* C *to reproduce the same finite path before any downstream uniform-space or completion read;*
7. *provenance, requiring* P *to name only the displayed RegSeqRat, StreamName, DyadicRatCore, Real, UniformSpace, and MetricCompletion dependencies;*
8. *non-escape, excluding quotient stream equality, selected limits, host real equality, countable choice, open-cover compactness, Real completeness, and any hidden host uniformity.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1628.1. The projection exposes exactly the regular rational name row, finite window row, dyadic tolerance row, entourage row, transport row, continuation row, provenance row, and local naming row.

Dyadic entourage admission is the row E downstream of W and D. Since D is the only tolerance coordinate and W is the only finite observation coordinate, the entourage cannot be read before both rows are displayed.

Window reindexing is recorded by the same W row and transported by H. A later common refinement changes only the inspected finite prefix, while the source row R and dyadic tolerance row D remain the rows named by the packet.

The Real and completion handoff is ordered by C. Any downstream consumer must replay R, W, D, E before reaching the structural rows H, C, P, N, so the consumer sees a finite dyadic-entourage route rather than a host uniformity.

The exclusions follow from the carrier inventory. There is no row for quotient stream equality, selected limits, host real equality, countable choice, open-cover compactness, Real completeness, or any hidden uniformity outside the displayed packet. □

Theorem 1628.3 (Regular Cauchy uniformity dyadic entourage stability)**.** *Let* $U = (R, W, D, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *carrier. If two finite* $\mathsf{StreamName}^{\uparrow}$ *windows are common refinements of the window row* W *and keep the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, then the dyadic-entourage row* E *remains the displayed entourage row for both refinements after componentwise* hsame *transport. In particular, common finite refinements preserve the same dyadic tolerance row* D *and do not create a second uniformity route.*

Proof. Start with the accepted carrier and the two common finite refinements of W. The carrier has only one source row R and only one dyadic tolerance row D, so both refinements are read as later windows for the same regular rational name and the same tolerance request.

Use the transport row H componentwise. It identifies the original window data and the refined window data only along the displayed regular rational, dyadic tolerance, and entourage rows.

Replay C after transport. The replay reaches the same entourage row E from each refined window, because no second tolerance coordinate, quotient stream coordinate, or host uniformity coordinate is present in the carrier.

Thus the dyadic-entourage row is stable under the common finite refinements, and the packet continues to expose a single finite route through R, W, D, E, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.RegularCauchyUniformityUp.RegularCauchyUniformityCarrier dyadic entourage stability`

Theorem 1628.4 (Regular Cauchy uniformity common-refinement transitivity)**.** *Let* $U = (R, W, D, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *carrier. For any displayed finite chain of* $\mathsf{StreamName}^{\uparrow}$ *common refinements*

$$W = W_0 \rightsquigarrow W_1 \rightsquigarrow \cdots \rightsquigarrow W_k$$

starting at the window row W*, if every step keeps the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, then the terminal window* W_k *carries the same dyadic-entourage row* E*. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of* W_k *still factors through the finite route*

$$R \longmapsto W \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

The chain introduces no quotient stream equality, selected limit, host real equality, countable choice, open-cover compactness, or Real completeness.

Proof. Induct on the displayed finite refinement chain. In the base case the terminal window is W itself, so the carrier projection of Definition 1628.1 supplies the source row R, tolerance row D, dyadic-entourage row E, and the replay route through H, C, P, N.

For the step, assume the current terminal window has already retained the same rows R, D, E. Apply Definition 1628.3 to the next common finite refinement. The hypothesis that the refinement keeps R and D makes the theorem applicable, and its conclusion preserves E as the displayed dyadic-entourage row.

Finally apply Definition 1628.5 to the terminal window. The consumer route remains the finite route through R, W, D, E, H, C, P, N, so the induction never creates quotient stream equality, a selected limit, host real equality, countable choice, open-cover compactness, Real completeness, or any host uniformity coordinate. □

Theorem 1628.5 (Regular Cauchy uniformity Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *carrier sees the uniformity only through the finite route*

$$R \longmapsto W \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a dyadic-entourage service for regular rational Cauchy names, not quotient stream equality, not a selected limit, not host real equality, not countable choice, not open-cover compactness, and not Real completeness.

Proof. Let $U = (R, W, D, E, H, C, P, N)$ be accepted. The only regular Cauchy source available to a downstream consumer is R, and the only observation surface attached to it is the finite window row W.

The tolerance request then passes through D, and the uniformity read is the dyadic-entourage row E. No Real or completion consumer can bypass those rows, because C records the replay order and H transports only the displayed coordinates.

The provenance and local naming rows P, N package the same route for $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ consumers. They do not add quotient stream equality, a selected limit, host real equality, countable choice, open-cover compactness, Real completeness, or a host uniformity coordinate. □

Theorem 1628.6 (Regular Cauchy uniformity scoped obligation package)**.** *The carrier of Definition 1628.1, the NameCert obligations of Definition 1628.2, dyadic-entourage stability of Definition 1628.3, common-refinement transitivity of Definition 1628.4, and the Real seal handoff of Definition 1628.5 form the complete* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *obligation package over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The package excludes quotient stream equality, selected limits, host real equality, countable choice, open-cover compactness, Real completeness, and hidden host uniformity.

Proof. Project the carrier of Definition 1628.1. Its rows are exactly the regular rational source, finite stream window, dyadic tolerance, dyadic entourage, componentwise transport, continuation replay, provenance, and local naming rows.

Apply Definition 1628.2 to read those rows as the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$ surface. The obligation rows include carrier admission, dyadic entourage admission, window reindexing, Real seal handoff, transport, replay, provenance, and non-escape.

Next apply Definition 1628.3 and Definition 1628.4. These two results compose the finite window route under common refinements while preserving the same source row, tolerance row, and entourage row.

Finally apply Definition 1628.5. Every $\mathsf{Real}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer still reaches the packet through the displayed finite route, so the listed BEDC rows exhaust the obligation surface and none of the excluded host-side data is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniformity}^{\uparrow}$ packet is generated from a regular rational name row, a finite StreamName window row, a DyadicRatCore tolerance row, a dyadic-entourage row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1628.1, Definition 1628.2, Definition 1628.3, Definition 1628.4, Definition 1628.5, and Definition 1628.6 close the finite dyadic-entourage obligation surface for RegSeqRat, StreamName, DyadicRatCore, Real, UniformSpace, and MetricCompletion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyUniformity}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.RegularCauchyUniformityUp.TasteGate.RegularCauchyUniformityTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, countable choice, open-cover compactness, Real completeness, or a public standard bridge.

Upgrade path: Public closure requires a public consumer theorem and a later bridge-checked standard interface for the displayed $\mathsf{RegularCauchyUniformity}^{\uparrow}$ dyadic-entourage route.

Chapter 1629

A Concrete Naming Certificate for $\mathsf{Upcrossing}^{\uparrow}$

Following Doob, the $\mathsf{Upcrossing}^{\uparrow}$ packet records the finite ledger used to count completed passages of a martingale path from a lower rational boundary to a higher rational boundary. It consumes the finite adapted-sequence surface of $\mathsf{Martingale}^{\uparrow}$ (chapter 190), the bounded stopped-window rows of $\mathsf{OptionalStopping}^{\uparrow}$ (chapter 383), and the probability, random-variable, and conditional-expectation sources $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{CondExp}^{\uparrow}$ already named in chapter 184, chapter 185, and chapter 188. The chapter names only the finite crossing-count carrier and its no-escape boundary; it does not assert martingale convergence or any almost-sure quotient theorem.

1629.1 Finite Crossing Carrier

Definition 1629.1 (Upcrossing finite ledger carrier)**.** *An* $\mathsf{Upcrossing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (\Omega, M, a, b, N, V, L, S, H, C, P, Q)$$

with the following visible rows:

1. Ω *is the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the probability context;*
2. M *is the consumed* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet over a finite unary time spine* $0, \ldots, N$*;*
3. a *and* b *are displayed* $\mathsf{Real}^{\uparrow}$ *boundary endpoints with a carried comparison row* $a <_R b$*;*
4. V *is the finite endpoint-row family of the martingale values read through* $\mathsf{RandomVar}^{\uparrow}$*;*
5. L *is the lower-boundary decision ledger recording every displayed time at which the current value is classified at or below* a *after no open crossing is active;*
6. S *is the upper-boundary decision ledger recording every displayed time at which the current value is classified at or above* b *after an open crossing is active;*
7. H *stores the componentwise* hsame *transports for the probability row, martingale row, boundary endpoints, endpoint rows, lower and upper decision rows, and crossing-count row;*
8. C *stores the finite* Cont *routes from the martingale endpoint ledger to the lower-boundary and upper-boundary decision ledgers;*
9. P *is the* Pkg *provenance row naming exactly* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$*, and the local crossing-count rows;*

10. *Q is the local* $\mathsf{NameCert}_{\mathsf{Upcrossing}^\uparrow}$ *row.*

The crossing count of U is the number of completed lower-to-upper pairs selected by the finite alternating ledger (L, S). The local classifier compares two carriers only through the transported finite time spine, boundary endpoints, decision rows, completed-count row, continuation routes, package provenance, and local certificate row.

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingTasteGate single carrier alignment`

1629.2 Certificate Obligations

Theorem 1629.2 (Upcrossing NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{Upcrossing}^\uparrow}$ *surface consists of five obligations:*

1. *carrier: formation of the finite packet of Definition 1629.1;*
2. *classifier: componentwise transport of the finite time spine, martingale endpoint rows, boundary endpoints, lower and upper decision ledgers, completed-count row,* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *row;*
3. *boundary-decision stability: every lower or upper decision is read from the displayed* $\mathsf{Real}^\uparrow$ *classifier on the corresponding* $\mathsf{RandomVar}^\uparrow$ *endpoint row and remains stable under the carried endpoint transport;*
4. *martingale-window handoff: every optional-stopping read used to isolate a lower or upper boundary hit factors through bounded* $\mathsf{OptionalStopping}^\uparrow$ *rows over the consumed finite* $\mathsf{Martingale}^\uparrow$ *packet;*
5. *ledger exactness: every probability, random-variable, conditional-expectation, martingale, stopped-window, decision, transport, continuation, package, and certificate row used by the crossing count appears in the displayed carrier.*

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingNamecertObligations`

Proof. Project the carrier tuple of Definition 1629.1. Carrier formation is exactly the finite package whose visible rows are Ω, M, a, b, N, V, L, S, H, C, P, and Q.

Classifier stability is componentwise. The transported martingale packet supplies the transported endpoint rows, while the boundary endpoints and decision ledgers are compared through the displayed $\mathsf{Real}^\uparrow$ and hsame rows already stored in the carrier.

For boundary-decision stability, each lower or upper decision is a finite read of one endpoint row against a or b. Transporting the endpoint row and the boundary endpoints transports the comparison row, so the selected lower and upper hits remain the same displayed finite decisions.

For the martingale-window handoff, each hit is read inside the finite time window of M. The bounded stopped-window rows of $\mathsf{OptionalStopping}^\uparrow$ package those finite reads without introducing unbounded stopping or a quotient of events.

Ledger exactness follows because the completed-count row is computed from the alternating lower and upper decision ledgers alone. No consumer of $\mathsf{NameCert}_{\mathsf{Upcrossing}^\uparrow}$ can read a probability row, endpoint row, conditional-expectation face, stopping window, transport, continuation route, package row, or local certificate outside the displayed carrier. □

Theorem 1629.3 (Upcrossing martingale dependency route)**.** *Every accepted* $\mathsf{Upcrossing}^\uparrow$ *carrier scopes its crossing-count read through the displayed finite* $\mathsf{Martingale}^\uparrow$ *adapted-sequence packet,*

the RandomVar$^{\uparrow}$ *endpoint rows carried by that packet, the* CondExp$^{\uparrow}$ *successor faces used by the martingale row, and the bounded* OptionalStopping$^{\uparrow}$ *windows that read lower and upper hits. The route contains no host stochastic process, no outside stopping-time theorem, and no convergence principle.*

Proof. Start with the consumed Martingale$^{\uparrow}$ packet M in Definition 1629.1. Its public surface supplies only the finite endpoint rows, conditional-expectation faces, filtration visibility rows, package provenance, and finite Cont/hsame ledger described in chapter 190.

The lower and upper decision ledgers L and S are then finite reads of those endpoint rows against a and b. Their comparison rows are carried by the displayed Real$^{\uparrow}$ endpoint classifier and by the RandomVar$^{\uparrow}$ rows already present in M.

Whenever the carrier isolates a first lower hit or the next upper hit, the read is bounded by the displayed finite horizon N. The relevant stopped-window route is therefore the bounded OptionalStopping$^{\uparrow}$ surface of chapter 383.

Thus the dependency route is exhausted by ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, CondExp$^{\uparrow}$, Martingale$^{\uparrow}$, OptionalStopping$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows. Since no other coordinate is present, the route cannot export a host stochastic process, an unbounded stopping theorem, or a convergence claim. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingMartingaleRoute`

Theorem 1629.4 (Upcrossing finite-ledger induction boundary)**.** *Every accepted* Upcrossing$^{\uparrow}$ *carrier computes its completed crossing count by induction on the displayed finite alternating lower and upper decision ledger* (L, S)*. The induction preserves boundary endpoints, endpoint rows, bounded* OptionalStopping$^{\uparrow}$ *windows, and crossing-count rows under componentwise* hsame *transport. An empty ledger gives zero completed crossings; a lower row opens a pending crossing; an upper row after a pending lower row adds exactly one completed count; and every case remains inside the displayed* Martingale$^{\uparrow}$*,* RandomVar$^{\uparrow}$*,* CondExp$^{\uparrow}$*,* OptionalStopping$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.UpcrossingUp.Upcrossing finite ledger induction boundary`

Proof. Inspect the finite ledger (L, S) in Definition 1629.1. The empty ledger has no lower row and no upper row, so the completed crossing count is the displayed zero count.

For the induction step with a lower row, the row is read from a displayed martingale endpoint against the boundary endpoint a. If no crossing is pending, the row opens one pending crossing and changes no completed count; if a pending crossing is already open, the finite state remains pending without exporting another object.

For the induction step with an upper row after a pending lower row, the row is read from the displayed endpoint against b. The pending lower row and this upper row form one completed lower-to-upper pair, so the completed-count row adds exactly one.

All transport is componentwise. The endpoint rows, boundary endpoints, stopped windows, decision rows, and crossing-count row are all coordinates of the same Upcrossing$^{\uparrow}$ carrier, so hsame transport and Cont replay preserve the finite induction without adding an unbounded stopping theorem or a convergence principle. □

Theorem 1629.5 (Upcrossing inequality handoff to martingale convergence)**.** *Every accepted* Upcrossing$^{\uparrow}$ *packet exports exactly the finite rows later consumed by a* MartingaleConvergence$^{\uparrow}$ *packet: displayed lower and upper* Real$^{\uparrow}$ *boundary rows, finite martingale endpoint rows, lower and*

upper decision ledgers, bounded OptionalStopping$^{\uparrow}$ *stopped-window rows, and the crossing-count ledger. The handoff is a finite dependency route, not a martingale convergence theorem.*

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingInequalityMartingaleConvergenceHandoff`

Proof. Project the carrier of Definition 1629.1. The boundary endpoints are a and b, the finite martingale endpoint family is V, the lower and upper decision ledgers are L and S, and the crossing-count read is computed from those displayed ledgers.

Apply Definition 1629.2. The certificate surface exposes the boundary-decision stability, martingale-window handoff, and ledger exactness rows required by any downstream convergence-preparation packet.

Apply Definition 1629.3. Every stopped-window read factors through the finite OptionalStopping$^{\uparrow}$ route over the consumed Martingale$^{\uparrow}$ packet, and no host stochastic process or limiting event is exported.

Thus the only data passed to MartingaleConvergence$^{\uparrow}$ is the displayed finite upcrossing handoff. Since no convergence coordinate is present in the carrier, the theorem records a dependency route rather than a convergence result. □

Theorem 1629.6 (Upcrossing MartingaleConvergence consumer route)**.** *Every* MartingaleConvergence$^{\uparrow}$ *consumer of an accepted* Upcrossing$^{\uparrow}$ *row, as named by chapter 2246, must pass through the finite lower and upper decision ledgers, the bounded* OptionalStopping$^{\uparrow}$ *stopped-window rows, and the Doob-upcrossing handoff of Definition 1629.5. The consumer receives only displayed lower and upper* Real$^{\uparrow}$ *boundary rows, finite martingale endpoint rows, lower and upper decision rows, bounded stopped-window routes, crossing-count data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows; it receives no limiting event, almost-sure quotient, convergence theorem, unbounded stopping principle, or completed measure carrier.*

Proof. Project the accepted Upcrossing$^{\uparrow}$ carrier. Its convergence-preparation data are the lower and upper boundary endpoints, finite martingale endpoint rows, lower and upper decision ledgers, and crossing-count row.

Apply the martingale route and NameCert obligations. The stopped-window reads are bounded by the displayed OptionalStopping$^{\uparrow}$ rows, and the decision ledgers are transported only componentwise and replayed only through the displayed continuation routes.

Finally use Definition 1629.5. The handoff identifies exactly the rows available to the downstream MartingaleConvergence$^{\uparrow}$ packet. Since those rows contain no limiting event or convergence coordinate, the consumer route is finite and no-escape rather than a convergence claim. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingMartingaleConvergenceConsumer`

Theorem 1629.7 (Upcrossing finite ledger handoff)**.** *For every accepted* Upcrossing$^{\uparrow}$ *carrier, the completed-crossing count, the bounded* OptionalStopping$^{\uparrow}$ *stopped-window ledger, and the lower and upper* Real$^{\uparrow}$ *boundary-rational rows form the only finite handoff consumed by the* MartingaleConvergence$^{\uparrow}$ *route of Definition 1629.6. The handoff is transported by the displayed* hsame *rows and replayed by the displayed* Cont *rows; it exports no almost-sure quotient equality, no unbounded stopping index, and no measure-completion carrier.*

Proof. Induct on the finite crossing ledger (L, S) of Definition 1629.1. In the empty case, the completed-crossing count is zero and the stopped-window ledger has no lower-to-upper pair to export.

For a lower-boundary row, the displayed boundary-rational data record a pending crossing against a. The stopped-window row is bounded by the current finite horizon, and no completed count is added.

For an upper-boundary row following a pending lower row, the displayed comparison against b closes exactly one finite pair. The completed-count row, the two boundary-rational rows, and the bounded stopped-window row are updated inside the same BHist packet.

The induction preserves the handoff under componentwise hsame transport and displayed Cont replay. Since the carrier contains no coordinate for almost-sure quotient equality, unbounded stopping, or measure completion, the martingale-convergence consumer can read only this finite ledger package. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingFiniteLedgerHandoff`

Theorem 1629.8 (Upcrossing scoped decision route)**.** *Every boundary-decision read from an accepted* Upcrossing$^{\uparrow}$ *carrier is scoped by* ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, CondExp$^{\uparrow}$, Martingale$^{\uparrow}$, OptionalStopping$^{\uparrow}$, *the finite lower and upper decision ledgers, and the structural rows* BHist, hsame, Cont, Pkg, *and* NameCert. *The read has no route through an unbounded stopping index, a host stochastic process, almost-sure quotient equality, or a convergence theorem.*

Proof. Project the carrier of Definition 1629.1. Its probability context, martingale packet, finite endpoint rows, lower ledger, upper ledger, transport rows, continuation rows, provenance row, and local certificate are the only coordinates available to a boundary decision.

Apply Definition 1629.2. Boundary-decision stability and ledger exactness force each lower or upper hit to be read from the displayed endpoint row against the displayed boundary endpoint, with transport and replay remaining componentwise.

Apply Definition 1629.3, Definition 1629.4, and Definition 1629.7. These route the endpoint rows through the finite martingale and optional-stopping windows, compute the completed count by the lower and upper decision ledgers, and expose only the finite handoff consumed downstream. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingScopedDecisionRoute`

Theorem 1629.9 (Upcrossing bounded optional-stopping route)**.** *Every lower or upper boundary hit used by the crossing ledger of an accepted* Upcrossing$^{\uparrow}$ *carrier factors through a bounded* OptionalStopping$^{\uparrow}$ *stopped-window row over the displayed finite* Martingale$^{\uparrow}$ *packet before the completed-count row is updated. Thus the lower ledger L and upper ledger S may update the count only after the corresponding finite endpoint row has been read through the displayed martingale-window route, with transport by H and replay by C.*

Proof. Project the carrier of Definition 1629.1. The only data available to a boundary hit are the probability row, the finite Martingale$^{\uparrow}$ packet M, the boundary endpoints a, b, the endpoint family V, the lower and upper ledgers L, S, and the support rows H, C, P, Q.

Apply Definition 1629.3. Every stopped-window read used by the chapter is bounded by the displayed finite horizon of M and factors through the OptionalStopping$^{\uparrow}$ rows carried by that route.

Apply Definition 1629.4. The completed-count row changes only in the finite induction step where a displayed lower row is followed by a displayed upper row; both rows are endpoint reads against the fixed boundaries.

Finally use Definition 1629.8. Boundary decisions are scoped by the same martingale packet, optional-stopping windows, transport rows, and continuation replay. Therefore each lower or upper hit must pass through the bounded stopped-window row before the ledger count is updated, and no unbounded stopping index or convergence theorem enters the route. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.UpcrossingBoundedOptionalStoppingRoute`

Theorem 1629.10 (Upcrossing no-escape boundary)**.** *The public boundary of* $\mathsf{Upcrossing}^{\uparrow}$ *exports only the finite crossing-count ledger for a displayed martingale window and fixed boundary pair* $a <_R b$*. It exports no martingale convergence theorem, no almost-sure quotient equality, no unbounded stopping principle, no measure completion, and no standard bridge beyond the finite rows explicitly carried by Definition 1629.1.*

Proof. Inspect the accepted carrier. Its time information is the finite horizon N, its value information is the finite endpoint family V, and its crossing data is the alternating lower and upper decision ledger (L, S).

Apply Definition 1629.2. The certificate surface reads only the listed probability, random-variable, conditional-expectation, martingale, optional-stopping, decision, transport, continuation, package, and local certificate rows.

Apply Definition 1629.3. Every martingale-facing read factors through the finite adapted-sequence packet and every stopped-window read is bounded by the displayed finite horizon.

None of the carrier coordinates names a limiting event, a quotient of almost-sure equality classes, an unbounded stopping index, a completed measure space, or a standard bridge. Therefore none of those objects can be exported by the public boundary of $\mathsf{Upcrossing}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.UpcrossingUp.Upcrossing nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{Upcrossing}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Upcrossing}^{\uparrow}$ packet is generated from a displayed probability row, a finite $\mathsf{Martingale}^{\uparrow}$ adapted-sequence packet, fixed lower and upper $\mathsf{Real}^{\uparrow}$ boundary endpoints, finite endpoint rows, lower and upper boundary decision ledgers, componentwise transports, finite continuation routes, package provenance, and a local NameCert row. The completed crossing count is read from the alternating finite decision ledger only.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Upcrossing}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1629.1, Definition 1629.2, Definition 1629.3, and Definition 1629.4 close the finite upcrossing packet through displayed probability, martingale, optional-stopping, boundary-decision, transport, replay, provenance, and finite-ledger induction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Upcrossing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a general optional-stopping theorem, martingale convergence, almost-sure quotient equality, unbounded stopping, measure-theoretic completeness, or a standard bridge.

Upgrade path: A checked formal axis requires Lean coverage for the scoped finite martingale-window dependency route and its no-escape boundary.

Chapter 1630

A Concrete Naming Certificate for $\mathsf{DoobUpcrossing}^{\uparrow}$

Following Doob, the $\mathsf{DoobUpcrossing}^{\uparrow}$ packet names the finite upcrossing-inequality surface obtained from a displayed martingale window. It consumes the probability, random-variable, conditional-expectation, martingale, optional-stopping, and crossing-count certificates of chapter 184, chapter 185, chapter 188, chapter 190, chapter 383, and chapter 1629. The finite Doob-bound row consumes the crossing-count ledger of chapter 1629 and the bounded stopped-window rows of chapter 383 before any martingale-convergence consumer may read it. Its carrier is not a convergence theorem; it is the finite ledger that relates a displayed crossing count to the corresponding expectation-bound row.

Definition 1630.1 (Doob upcrossing carrier)**.** *A* $\mathsf{DoobUpcrossing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Omega, M, a, b, N, U, E, B, H, C, P, Q)$$

with the following visible rows:

1. Ω *is the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the probability context;*
2. M *is the consumed* $\mathsf{Martingale}^{\uparrow}$ *finite adapted-sequence packet over the displayed time spine* $0, \ldots, N$*;*
3. a *and* b *are displayed* $\mathsf{Real}^{\uparrow}$ *endpoints with a carried strict comparison row* $a <_R b$*;*
4. U *is the consumed* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count ledger for* M, a, b, N*;*
5. E *is the finite expectation row read through the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint rows and the* $\mathsf{CondExp}^{\uparrow}$ *faces of* M*;*
6. B *is the displayed Doob bound row comparing the scaled completed-count read* $(b-a) \cdot U_N(a, b)$ *with the final negative-part expectation read;*
7. H *records componentwise* hsame *transport for the probability row, martingale row, boundary endpoints, upcrossing ledger, expectation row, bound row, continuation row, provenance row, and local naming row;*
8. C *records the finite* Cont *routes from the martingale endpoint rows and bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *windows to the crossing-count and expectation-bound reads;*
9. P *is the* Pkg *provenance row naming exactly* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, *and the local Doob-bound rows;*

10. *Q is the local* $\mathsf{NameCert}_{\mathsf{DoobUpcrossing}^{\uparrow}}$ *row.*

The local classifier compares two carriers only through these displayed rows. In particular, it compares the bound row as a finite relation between the consumed upcrossing ledger and the displayed expectation read, not as an ambient theorem about all stochastic processes.

Lean4 ✓ `BEDC.Derived.DoobUpcrossingUp.DoobUpcrossingTasteGate single carrier alignment`

Theorem 1630.2 (Doob upcrossing NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{DoobUpcrossing}^{\uparrow}}$ *surface consists of five obligations:*

1. *carrier: formation of the finite packet of Definition 1630.1;*
2. *classifier: componentwise transport of the probability row, martingale row, boundary endpoints, crossing-count ledger, expectation row, Doob-bound row,* hsame *rows,* Cont *routes,* Pkg *provenance, and local* NameCert *row;*
3. *upcrossing consumption: the completed-count coordinate is read only from the finite* $\mathsf{Upcrossing}^{\uparrow}$ *ledger of chapter 1629;*
4. *martingale-bound route: the expectation and stopped-window reads factor through the* $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, *and* $\mathsf{OptionalStopping}^{\uparrow}$ *rows named in the carrier;*
5. *ledger exactness: every row used by the inequality surface appears in the displayed carrier.*

Proof. Project the carrier tuple of Definition 1630.1. Carrier formation is exactly the finite package whose visible rows are Ω, M, a, b, N, U, E, B, H, C, P, and Q.

Classifier stability is componentwise. The probability, endpoint, martingale, upcrossing, expectation, and bound coordinates are compared by the displayed hsame rows, while the replay of the dependency route is confined to C and P.

The upcrossing-consumption obligation uses chapter 1629. Its finite ledger supplies the completed-count row, so the Doob packet does not construct a second crossing counter.

The martingale-bound route uses chapter 184, chapter 185, chapter 188, chapter 190, and chapter 383. These chapters provide the probability context, endpoint rows, conditional-expectation faces, finite martingale window, and bounded stopped-window reads consumed by E and B.

Ledger exactness follows because no coordinate outside D is named by the carrier or the local classifier. Every public inequality read therefore projects from one of the displayed rows. □

Theorem 1630.3 (Doob upcrossing martingale route)**.** *Every accepted* $\mathsf{DoobUpcrossing}^{\uparrow}$ *carrier scopes its finite inequality read through the displayed* $\mathsf{ProbSpace}^{\uparrow}$ *source, the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint rows, the* $\mathsf{CondExp}^{\uparrow}$ *martingale faces, the finite* $\mathsf{Martingale}^{\uparrow}$ *adapted-sequence packet, the bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *windows, and the consumed* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count ledger. The route contains no host stochastic process, no unbounded stopping theorem, no limiting event, and no martingale convergence principle.*

Proof. Start with the finite martingale packet M. Its public surface supplies only displayed endpoint rows, conditional-expectation faces, filtration visibility rows, and finite transport and provenance data as named by chapter 190.

The endpoint rows are interpreted inside the probability context through chapter 184, chapter 185, and chapter 188. The crossing-count coordinate is not recomputed here; it is the consumed $\mathsf{Upcrossing}^{\uparrow}$ row of chapter 1629.

The only stopped reads used to compare lower and upper visits with the final expectation row are bounded reads through chapter 383. Since the carrier fixes the horizon N and stores no unbounded index coordinate, the route cannot export an unbounded stopping theorem.

Thus the martingale-facing content of the packet is exhausted by the named finite dependency rows and the local Doob-bound row. No limiting event, almost-sure quotient, or convergence principle is present in the carrier. □

Theorem 1630.4 (Doob upcrossing no-escape boundary)**.** *The public boundary of* $\mathsf{DoobUpcrossing}^{\uparrow}$ *exports only the finite Doob upcrossing inequality row for a displayed martingale window, boundary pair* $a <_R b$*, consumed crossing-count ledger, and displayed expectation-bound read. It exports no martingale convergence theorem, no almost-sure quotient equality, no unbounded stopping principle, no measure completion, and no standard bridge beyond the rows explicitly carried by Definition 1630.1.*

Proof. Inspect the accepted carrier. Its time information is the finite horizon N, its stochastic information is the displayed probability and martingale dependency rows, and its crossing information is the consumed $\mathsf{Upcrossing}^{\uparrow}$ ledger.

Apply Definition 1630.2. The certificate surface reads the inequality only through the displayed carrier, classifier, upcrossing-consumption, martingale-bound-route, and ledger-exactness rows.

Apply Definition 1630.3. Every martingale-facing read factors through the finite adapted-sequence packet and every stopped-window read is bounded by the displayed finite horizon.

None of the carrier coordinates names a limiting event, a quotient of almost-sure equality classes, an unbounded stopping index, a completed measure space, or a standard bridge. Therefore none of those objects can be exported by the public boundary of $\mathsf{DoobUpcrossing}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{DoobUpcrossing}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DoobUpcrossing}^{\uparrow}$ packet is generated from a displayed probability row, a finite $\mathsf{Martingale}^{\uparrow}$ adapted-sequence packet, fixed lower and upper $\mathsf{Real}^{\uparrow}$ boundary endpoints, a consumed $\mathsf{Upcrossing}^{\uparrow}$ crossing-count ledger, endpoint expectation rows, a displayed Doob-bound comparison row, componentwise hsame transports, finite continuation routes, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoobUpcrossing}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope names the finite carrier, the five-row naming-certificate obligation surface, the martingale dependency route, and the no-escape boundary for finite Doob upcrossing inequality rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoobUpcrossing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close martingale convergence, almost-sure quotient equality, unbounded stopping, measure completion, or a standard bridge.

Upgrade path: Obligation closure requires Lean targets for carrier transport, upcrossing consumption, martingale-bound routing, finite inequality-row exactness, and the no-escape boundary.

Chapter 1631

A Concrete Naming Certificate for $\mathsf{ModulusContinuity}^{\uparrow}$

Following Bishop and Bridges on constructive continuity with explicit moduli, the $\mathsf{ModulusContinuity}^{\uparrow}$ packet names the function-specific route by which a tolerance request is converted into a finite source-window budget before a Real or Completion consumer reads the target seal. It sits between the graph surface of chapter 130, the tolerance-to-window discipline of chapter 292, the finite source observations of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet records the local modulus-continuity boundary used by Real and Completion routes; it is not an open-cover compactness principle, a selected limit, a quotient stream equality, a host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1631.1 (Modulus-continuity carrier)**.** *A* $\mathsf{ModulusContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, S, K, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the named function, including the displayed source and target endpoints that every consumer must inspect before reading continuity data.*
2. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose entries supply the domain-side observations governed by the modulus.*
3. *K is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tolerance-to-window row, sending each positive dyadic target tolerance to the displayed finite source window read through S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the target tolerance, the source-window budget, and the dyadic comparison rows used by K.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source and target windows inspected by the graph and modulus rows.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *target seal reached only after G, S, K, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the graph, source window, modulus row, dyadic ledger, regular readback, Real target seal, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a target tolerance request through D, then through the modulus row K, into the source window S, across the graph row G, through readback Q, and finally into the Real target seal R.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, S, K, D, Q, R, H, C, P, N$. It has no coordinate for open-cover compactness, selected limits, quotient stream equality, host real equality, a detached continuity oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.TasteGate.ModulusContinuityTasteGate single carrier alignment`

Theorem 1631.2 (Modulus-continuity NameCert obligations)**.** *For every accepted* $\mathsf{ModulusContinuity}^{\uparrow}$ *carrier $M = (G, S, K, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the continuous-map graph, source StreamName window, CauchyModulus tolerance-to-window row, dyadic tolerance ledger, RegSeqRat readback, Real target seal, transport, replay, provenance, and local naming row;*
2. *classifier transport, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the ordered route D, K, S, G, Q, R;*
3. *modulus-row stability, requiring every target tolerance read from D to select its source-window budget only through the displayed row K and never through an external continuity oracle;*
4. *continuity-window handoff, requiring every graph read in G and every regular readback in Q to consume the source window S selected by K before the target seal R is exposed;*
5. *ledger non-escape, requiring P and N to export no open-cover compactness, selected limit, quotient stream equality, host real equality, detached continuity oracle, or ambient Real completeness.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1631.1. The visible packet consists of the continuous-map graph G, the finite source-window row S, the Cauchy modulus row K, the dyadic ledger D, the readback row Q, the terminal Real seal R, and the structural rows H, C, P, N.

Classifier transport is structural. Componentwise hsame in H preserves each displayed coordinate, and displayed Cont replay in C repeats the same ordered route D, K, S, G, Q, R for every accepted tolerance request.

Modulus-row stability is forced by the placement of K. A target tolerance may be read from D, but the only row that converts it into a source-window budget is the displayed $\mathsf{CauchyModulus}^{\uparrow}$ row, so no detached continuity oracle can enter the carrier.

The continuity-window handoff follows next. The graph row G and the regular readback row Q are exposed only after the window S selected by K has been displayed, and the Real target seal R is reached only after those finite observations are replayed.

Finally, provenance is exhausted by P and N. Since the carrier has no coordinate for open-cover compactness, selected limits, quotient stream equality, host real equality, a detached continuity oracle, or ambient Real completeness, the local certificate cannot export any such object. □

Theorem 1631.3 (Modulus-continuity Real and Completion handoff)**.** *Every Real or Completion consumer of an accepted* ModulusContinuity$^{\uparrow}$ *carrier reads the function-specific modulus only through the displayed rows*

ContinuousMap$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg,

Consequently the consumer receives a finite tolerance-to-window route for the named function and no open-cover compactness, selected limit, quotient stream equality, host real equality, detached continuity oracle, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $M = (G, S, K, D, Q, R, H, C, P, N)$. Carrier projection gives a Real or Completion consumer access to the function-specific modulus only through those displayed rows.

The tolerance route starts in D. The target dyadic tolerance is converted into a finite source-window budget by K, and that budget identifies the source window S before any graph read is available.

The graph and readback route then passes through G and Q. These rows expose only the finite observations admitted by the selected window and the regular rational readback attached to it; the terminal Real target seal R is downstream of that replay.

The remaining rows do not add mathematical primitives. The BHist, hsame, Cont, Pkg, and NameCert rows transport, replay, source, and name the same finite route, so no excluded compactness, limit, quotient, host-equality, oracle, or completeness coordinate can be inserted. □

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityRealCompletion handoff`

Theorem 1631.4 (Modulus-continuity Cauchy route)**.** *For every accepted* ModulusContinuity$^{\uparrow}$ *carrier*

$$M = (G, S, K, D, Q, R, H, C, P, N),$$

every Cauchy, Real, or Completion tolerance request routed through M factors through the finite carrier path

$$D \to K \to S \to G \to Q \to R.$$

The rows H, C, P, N only transport, replay, source, and name that finite path. They do not add open-cover compactness, selected limits, quotient stream equality, host real equality, detached continuity oracles, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityCauchyRoute`

Proof. Project the accepted carrier into its displayed rows. By Definition 1631.2, carrier admission exposes only the graph row G, source-window row S, modulus row K, dyadic tolerance ledger D, regular readback row Q, Real target seal R, and the structural rows H, C, P, N.

The same obligation theorem fixes the order of use. Classifier transport preserves the displayed route, modulus-row stability forces each target tolerance read from D to pass through K, and continuity-window handoff sends the selected finite window through S before the graph and readback rows G and Q can reach the terminal seal R.

The structural coordinates cannot create an additional analytic source. Componentwise transport in H, displayed replay in C, provenance in P, and local naming in N only carry the already projected path $D \to K \to S \to G \to Q \to R$.

Finally apply Definition 1631.3. Any Cauchy, Real, or Completion consumer reads the function-specific modulus only through the displayed rows, so the excluded compactness, selected-limit,

quotient-stream, host-equality, detached-oracle, and ambient-completeness objects have no coordinate through which to enter the route. □

Theorem 1631.5 (Uniform Cauchy handoff for modulus continuity)**.** *Let*

$$M = (G, S, K, D, Q, R, H, C, P, N)$$

be an accepted $\mathsf{ModulusContinuity}^{\uparrow}$ *carrier over certified dyadic source windows. Suppose a family of source streams is presented through* $\mathsf{StreamName}^{\uparrow}$ *rows whose regular-rational readbacks are* $\mathsf{RegSeqRat}^{\uparrow}$ *streams and whose Cauchy witnesses share the same dyadic modulus row in* D*. Then the target streams obtained by routing each source stream through* M *are regular Cauchy target streams with the target modulus obtained by reindexing the shared dyadic budget through*

$$D \to K \to S \to G \to Q \to R.$$

The handoff uses only the chapter's non-escape and composition rows: it exports no selected limit, quotient stream equality, host real equality, detached continuity oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityUniformCauchyHandoff`

Proof. Project M through Definition 1631.1. The shared dyadic Cauchy modulus is read in D, and the carrier's modulus row K converts every target tolerance request into the same certified source-window budget.

Reindex the source streams along that budget. The $\mathsf{StreamName}^{\uparrow}$ row S selects the finite observations required by K, and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q supplies the regular-rational readback for those windows. Since all source streams share the same dyadic modulus row, the reindexed windows are uniform across the family.

Route the selected windows through the graph row G and then to the target seal R. The Cauchy route of Definition 1631.4 fixes the order of this handoff, so every target tolerance is discharged by the finite path $D \to K \to S \to G \to Q \to R$.

Finally apply the composition and non-escape rows. The structural coordinates H, C, P, N transport and name the same finite route, while Definition 1631.9 prevents a selected limit, quotient equality, host equality, detached continuity oracle, or ambient completeness row from entering the exported regular Cauchy target stream. □

Theorem 1631.6 (Modulus-continuity composition route)**.** *Let*

$$M_f = (G_f, S_f, K_f, D_f, Q_f, R_f, H_f, C_f, P_f, N_f)$$

and

$$M_g = (G_g, S_g, K_g, D_g, Q_g, R_g, H_g, C_g, P_g, N_g)$$

be accepted $\mathsf{ModulusContinuity}^{\uparrow}$ *carriers. Suppose the target seal exposed by* M_f *is the source window and readback surface consumed by* M_g*. Then every target tolerance request for the composed Real or Completion consumer factors through the finite route*

$$D_g \to K_g \to S_g \to G_g \to Q_g \to D_f \to K_f \to S_f \to G_f \to Q_f \to R_f.$$

The composed route uses only the displayed $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the two carriers.*

Proof. Project the second carrier first. By Definition 1631.2, the request for the outer target tolerance starts in D_g, is converted by K_g into the finite window S_g, and reaches the graph and readback rows only through G_g and Q_g.

Use the stated seal match to hand the consumed source surface of M_g to the target surface of M_f. The handoff theorem Definition 1631.3 says that this surface is not a free Real or Completion oracle; it is read only through the displayed rows of M_f.

Project the first carrier along the same finite pattern. The matching request therefore passes through D_f, then K_f, then S_f, then G_f and Q_f, and only after those finite rows have been replayed does it reach the terminal Real seal R_f.

The structural rows add no new primitive. The BHist, hsame, Cont, Pkg, and NameCert coordinates in M_f and M_g transport and name the two projected routes, so their concatenation is still a finite tolerance-to-window route built from the displayed carrier rows alone. □

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityCompositionRoute`

Theorem 1631.7 (Modulus-continuity double-composition route)**.** *Let*

$$M_f = (G_f, S_f, K_f, D_f, Q_f, R_f, H_f, C_f, P_f, N_f),$$

$$M_g = (G_g, S_g, K_g, D_g, Q_g, R_g, H_g, C_g, P_g, N_g),$$

and

$$M_h = (G_h, S_h, K_h, D_h, Q_h, R_h, H_h, C_h, P_h, N_h)$$

be accepted ModulusContinuity$^{\uparrow}$ *carriers whose target and source surfaces match consecutively. Then every target tolerance request for the triple composed Real or Completion consumer factors through the finite concatenated route*

$$D_h \to K_h \to S_h \to G_h \to Q_h \to D_g \to K_g \to S_g \to G_g \to Q_g \to D_f \to K_f \to S_f \to G_f \to Q_f \to R_f.$$

The route uses only the displayed ContinuousMap$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of the three carriers.*

Proof. Apply Definition 1631.6 first to M_g and M_h. The outer tolerance request is routed through D_h, K_h, S_h, G_h, Q_h and then through the matched surface of M_g.

Apply the same composition route to the resulting matched surface and M_f. The second handoff extends the displayed route through D_g, K_g, S_g, G_g, Q_g and then through $D_f, K_f, S_f, G_f, Q_f, R_f$.

Use Definition 1631.3 at both matched surfaces. The handoff theorem prevents either match from becoming a detached continuity oracle, an ambient completeness row, or a selected Real limit; each match is read only through the displayed carrier rows.

The structural rows only transport and source the same finite route. Componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows concatenate the three packet routes without adding any object beyond the displayed modulus-continuity inventories. □

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityDoubleCompositionRoute`

Theorem 1631.8 (Modulus-continuity composition associativity route)**.** *For three accepted* ModulusContinuity$^{\uparrow}$ *carriers with matching consecutive source and target surfaces, applying the composition route as* $(h \circ g) \circ f$ *or as* $h \circ (g \circ f)$ *exposes the same finite tolerance-to-window route through the displayed* D, K, S, G, Q, R *rows of the three carriers and the same structural* H, C, P, N *rows. Neither parenthesization exports open-cover compactness, selected limits, quotient stream equality, host real equality, detached continuity oracles, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Apply Definition 1631.6 twice in the parenthesization $(h \circ g) \circ f$. The outer request reads through the rows of M_h, then the rows of M_g, and finally the rows of M_f before reaching the terminal Real seal.

Apply the same theorem twice in the parenthesization $h \circ (g \circ f)$. The intermediate grouping changes which composed route is named first, but the displayed tolerance-to-window reads still pass through the same D, K, S, G, Q, R rows in the same carrier order.

Compare both readings with the double-composition route of Definition 1631.7. It records the common finite concatenation, while the structural rows H, C, P, N only transport, replay, source, and name that concatenation.

Finally use Definition 1631.9. Since the composed consumer has no coordinate for compactness, selected limits, quotient stream equality, host real equality, detached continuity oracles, or ambient completeness, neither parenthesization can export one. □

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityCompositionAssociativityRoute`

Theorem 1631.9 (Modulus-continuity composition non-escape)**.** *Under the hypotheses of Definition 1631.6, the composed Real or Completion consumer exports no open-cover compactness, selected limit, quotient stream equality, host real equality, detached continuity oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exported read is confined to the displayed rows of M_f and M_g and their concatenated finite tolerance-to-window route.*

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityCompositionNonescape`

Proof. Project both carriers through Definition 1631.1. The only rows available in each packet are the continuous-map graph, source window, modulus row, dyadic ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 1631.3 to each leg. For M_g, the outer request reads its target tolerance through D_g, K_g, S_g, G_g, Q_g before it reaches the matched surface. For M_f, that matched surface is consumed only through $D_f, K_f, S_f, G_f, Q_f, R_f$.

Now concatenate the two displayed routes using Definition 1631.6. Concatenation adds no coordinate beyond the rows already projected from M_f and M_g; it only orders the finite tolerance-to-window reads into one longer route.

The excluded analytic principles are absent from both carrier inventories. Since transport, replay, provenance, and local naming preserve the displayed rows rather than adding open-cover compactness, selected limits, quotient stream equality, host real equality, detached continuity oracles, or ambient Real completeness, none of those objects can be exported by the composed consumer. □

Theorem 1631.10 (Modulus-continuity public consumer surface)**.** *Every public* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or Completion consumer of an accepted* $\mathsf{ModulusContinuity}^{\uparrow}$ *carrier must read the finite path*

$$D \to K \to S \to G \to Q \to R$$

together with the structural rows H, C, P, N. The exported public surface is exactly the displayed tolerance-to-window route, componentwise transport, continuation replay, provenance, and local naming; it contains no open-cover compactness, selected limit, quotient stream equality, host real equality, detached continuity oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusContinuityUp.ModulusContinuityPublicConsumerSurface`

Proof. Use the Real and Completion handoff of Definition 1631.3. It confines every Real or Completion read to the displayed graph, modulus, source-window, regular readback, dyadic tolerance, Real seal, and structural rows.

The Cauchy route of Definition 1631.4 fixes the order of those analytic rows as $D \to K \to S \to G \to Q \to R$. The uniform Cauchy handoff of Definition 1631.5 gives the same ordered read to public regular-rational consumers.

Composition does not enlarge the public surface. By Definition 1631.6, composed consumers concatenate displayed modulus-continuity routes without adding analytic coordinates, and Definition 1631.9 excludes open-cover compactness, selected limits, quotient stream equality, host real equality, detached continuity oracles, and ambient completeness from the exported route.

Thus the public consumer receives only the finite path and the structural rows H, C, P, N named in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusContinuity}^{\uparrow}$ packet is generated from a ContinuousMap graph row, a source StreamName window, a CauchyModulus tolerance-to-window row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real target seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulusContinuity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1631.1, Definition 1631.2, Definition 1631.3, Definition 1631.4, Definition 1631.5, Definition 1631.6, Definition 1631.9, and Definition 1631.10 bind the public consumer surface as the bridge-facing route through ContinuousMap, CauchyModulus, StreamName, RegSeqRat, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulusContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: The mature paper-side treatment is complete for the displayed public consumer surface; formal and bridge verification remain separate verification-axis work.

Chapter 1632

A Concrete Naming Certificate for $\mathsf{CauchyContinuousExtension}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{CauchyContinuousExtension}^{\uparrow}$ packet records the finite handoff by which a Cauchy-continuous map on regular rational source data is read through the displayed completion route before it reaches a completed real-facing extension. It consumes the finite windows of chapter 126, the regular rational source rows of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the regular completion universal surface of chapter 1425, the extension and uniqueness interface of chapter 397, and the uniformly continuous extension route of chapter 769. The chapter names the Cauchy-continuous completion-extension boundary as finite BEDC data; it does not turn host completion, quotient representatives, or function equality into packet coordinates.

Definition 1632.1 (Cauchy-continuous extension carrier)**.** *A* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, W, D, F, U, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data and the regular source discipline of* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$*;*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule for source observations and comparison requests;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling Cauchy-continuous requests on those windows;*
4. *F is the Cauchy-continuous map row, recording that matching source windows inside the tolerance ledger produce matching target completion reads;*
5. *U is the completion-universal extension row, read through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *and aligned with the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *handoff when a uniform-modulus row is present;*
6. *L is the uniqueness ledger comparing two extension rows over the same regular source, finite windows, dyadic ledger, Cauchy-continuous map row, and completion-universal extension row;*
7. *H records componentwise* hsame *transport for S, W, D, F, U, L;*
8. *C records the displayed* Cont *replay route forcing every extension read through S, W, D, F, U, L in that order;*

9. *P is the* Pkg *provenance over* RegularCauchyCompletionUniversal$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, UniformContinuousExtension$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and local* NameCert *rows;*

10. *N is the local* NameCert$_{\text{CauchyContinuousExtension}^{\uparrow}}$ *row.*

The carrier stores no host function equality, quotient completion, selected limit, countable-choice schedule, or ambient completeness coordinate.

Theorem 1632.2 (CauchyContinuousExtension NameCert obligations)**.** *For the carrier of Definition 1632.1, the local* NameCert$_{\text{CauchyContinuousExtension}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed packet $E = (S, W, D, F, U, L, H, C, P, N)$;*

2. *regular source handoff, requiring each extension request to read S through the finite windows W and tolerance ledger D before any completion-facing row is available;*

3. *Cauchy-continuous map control, requiring the map row F to transport source Cauchy closeness into the target completion request consumed by U;*

4. *extension and uniqueness control, requiring the completion-universal row U and uniqueness ledger L to compare candidate extensions only over the same displayed source, window, tolerance, and map rows;*

5. *ledger non-escape, stating that H, C, P, N expose no host function equality, quotient completion, selected limit, countable choice, or ambient completeness principle.*

Proof. Project the packet in Definition 1632.1. Carrier admission is exactly the presence of the ten rows $S, W, D, F, U, L, H, C, P, N$ with a local naming row.

The source handoff reads the first three coordinates. The regular source S is visible only through the finite StreamName windows W and the dyadic tolerance ledger D, so a completion-facing request cannot bypass the regular rational source route.

The map and extension clauses then read F and U. The Cauchy-continuous map row F is the only route from source Cauchy closeness to a target completion request, and U is the only row that repacks that request through the completion-universal extension surface.

Finally the comparison and non-escape clauses inspect L, H, C, P, N. The uniqueness ledger L compares extensions over the same displayed coordinates, while H and C only transport and replay those coordinates. Since no excluded principle is a carrier coordinate, the five obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.CauchyContinuousExtensionNameCertObligations`

Theorem 1632.3 (CauchyContinuousExtension regular source handoff)**.** *Let $E = (S, W, D, F, U, L, H, C, P, N)$ be an accepted* CauchyContinuousExtension$^{\uparrow}$ *carrier. Every extension read exported by U factors through the regular-Cauchy source row S, the finite window schedule W, the dyadic tolerance ledger D, and the Cauchy-continuous map row F before it reaches the completion-universal extension surface.*

Proof. Start with the source coordinates. The rows S, W, D record the only regular-Cauchy observations available to the packet, so every admitted source read is a finite-window and tolerance-ledger read.

Apply the map row next. The row F consumes exactly those source reads and records the Cauchy-continuous target request; it is not accompanied by a separate host map or selected target limit.

The extension row U is therefore reached only after the route S, W, D, F has been replayed. The Cont row C preserves this order, and the provenance row P records the same sibling surfaces. Hence every exported extension factors through the regular source handoff before reading the completion-universal surface. □

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.TasteGate.CauchyContinuousExtensionRegularSourceHandof`

Theorem 1632.4 (CauchyContinuousExtension uniqueness ledger)**.** *Let $E = (S, W, D, F, U, L, H, C, P, N)$ and $E' = (S, W, D, F, U', L', H', C', P, N')$ be accepted* CauchyContinuousExtension$^{\uparrow}$ *carriers over the same regular source, finite windows, dyadic ledger, Cauchy-continuous map row, and provenance. Any comparison of the exported extension rows is exhausted by the displayed uniqueness ledgers L and L' together with componentwise* hsame *transport over S, W, D, F, U and U'. No host function equality, quotient completion, selected limit, countable choice, or ambient completeness row can contribute to the comparison.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.TasteGate.CauchyContinuousExtensionUniquenessLedger`

Proof. Read the shared coordinates S, W, D, F, P. They fix the regular source, window schedule, dyadic tolerance ledger, Cauchy-continuous map row, and provenance for both packets.

The only extension-specific coordinates are U, L, H, C, N and U', L', H', C', N'. The ledgers L and L' compare the two extension rows over the same shared source route, while H, H' provide only componentwise hsame transport and C, C' replay only the displayed coordinate order.

Because neither carrier contains host function equality, quotient completion, selected limits, countable-choice schedules, or ambient completeness witnesses, no external comparison principle is available. The uniqueness judgement is exactly the local ledger comparison over the displayed Cauchy-continuous extension route. □

Definition 1632.5 (Cauchy extension uniform-modulus sink)**.** *The uniform-modulus sink for a* CauchyContinuousExtension$^{\uparrow}$ *packet is the finite route*

$$S, W, D, F, U \longmapsto L, H, C, P, N$$

in which the StreamName$^{\uparrow}$ *windows W,* RegSeqRat$^{\uparrow}$ *regular source row S,* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, Cauchy-continuous map row F, and* UniformContinuousExtension$^{\uparrow}$ *modulus row recorded inside U are read before the completion-universal extension row can be consumed. The sink exports only the displayed completion-universal extension boundary and its uniqueness, transport, replay, provenance, and local naming rows; it exports no host function equality, quotient completion, selected limit, countable-choice schedule, or ambient completeness coordinate.*

Lean4 (definition) `BEDC.Derived.CauchyContinuousExtensionUp.CauchyExtensionUniformModulusSink`

Theorem 1632.6 (Cauchy extension regular-tail transport)**.** *Let $E = (S, W, D, F, U, L, H, C, P, N)$ be an accepted* CauchyContinuousExtension$^{\uparrow}$ *carrier. Componentwise* hsame *transport preserves the uniform-modulus sink of Definition 1632.5 only through the displayed rows $S, W, D, F, U, L, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.CauchyExtensionRegularTailTransport`

Proof. Project the accepted carrier. The sink begins with the regular source row S, finite windows W, dyadic tolerance ledger D, Cauchy-continuous map row F, and the extension/modulus handoff stored in U.

Componentwise hsame transport is recorded by H for exactly those rows and for the uniqueness ledger L. The replay row C then preserves the same order before provenance and local naming are read.

No other row is present in the carrier. Thus transport cannot use host function equality, quotient completion, selected limits, countable choice, or ambient completeness; it preserves the regular-tail sink precisely through $S, W, D, F, U, L, H, C, P, N$. □

Theorem 1632.7 (Cauchy extension completion consumer)**.** *Every* Real$^{\uparrow}$ *or* CompletionUniversalProperty$^{\uparrow}$ *consumer of an accepted* CauchyContinuousExtension$^{\uparrow}$ *packet reads the uniform-modulus sink of Definition 1632.5 before it can inspect the exported extension row.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.TasteGate.CauchyContinuousExtensionCarrier completion consumer`

Proof. Start with an accepted packet $E = (S, W, D, F, U, L, H, C, P, N)$. A real-facing or completion-universal consumer can reach the exported extension only through U, because the carrier contains no second extension coordinate.

The row U is available only after the source windows, regular source, dyadic tolerance ledger, Cauchy-continuous map row, and uniform-modulus handoff have been read. This is exactly the sink route named in Definition 1632.5.

The remaining rows L, H, C, P, N compare, transport, replay, source, and name the same route. Hence a Real$^{\uparrow}$ or CompletionUniversalProperty$^{\uparrow}$ consumer must read the uniform-modulus sink before inspecting the exported extension. □

Theorem 1632.8 (Cauchy-continuous extension choice-free density route)**.** *Every accepted* CauchyContinuousExten *extension read*

$$E = (S, W, D, F, U, L, H, C, P, N)$$

must expose the dense regular-Cauchy source row S, the finite StreamName$^{\uparrow}$ *windows W, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, and the Cauchy-continuous map row F before the extension row U is consumed. The displayed route reads through* RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RegularCauchyCompletionUniversal$^{\uparrow}$, *and* UniformContinuousExtension$^{\uparrow}$ *as finite packet data; it refuses selected representatives, host function equality, quotient completion, countable choice, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.CauchyContinuousExtensionCarrier choice free density route`

Proof. Project the carrier of Definition 1632.1. The rows $S, W, D, F, U, L, H, C, P, N$ are the only coordinates available to an accepted extension read.

Apply Definition 1632.3. It places the regular-Cauchy source row, finite windows, dyadic tolerance ledger, and Cauchy-continuous map row before the completion-universal extension surface can be read.

Use the sink of Definition 1632.5 and the consumer boundary of Definition 1632.7. Together they show that a Real$^{\uparrow}$ or completion-facing consumer reaches U only through the displayed finite-window route.

Thus density is carried by the visible regular-Cauchy source and its finite window ledger. Since no selected representative, host function equality, quotient completion, countable-choice schedule, or ambient completeness coordinate appears in the packet, the density route is choice-free BEDC data rather than an external selection principle. □

Theorem 1632.9 (Cauchy-continuous extension Real uniform handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *packet that asks for a uniform entourage must first read the sink rows* S, W, D, F, U *of Definition 1632.5 and then the positive-radius and metric-entourage route of* $\mathsf{RealUniformStructure}^{\uparrow}$ *from chapter 1624. The handoff cannot inspect selected limits, quotient filters, countable choice, host function equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.CauchyContinuousExtensionCarrier real uniform handoff`

Proof. Start with the completion consumer theorem Definition 1632.7. A Real-facing consumer reaches the exported extension row only through the sink displayed by S, W, D, F, U.

Apply the choice-free density route of Definition 1632.8. It forces the same consumer to read the dense regular-Cauchy source, finite StreamName windows, dyadic tolerance ledger, and Cauchy-continuous map row before the extension row is consumed.

Now read the Real uniform-structure chapter. The metric-uniformity handoff and finite entourage-basis route of chapter 1624 expose uniform entourages only through Real endpoints, metric distance data, positive dyadic radii, generated entourage rows, finite windows, and regular rational readback.

Composing these routes gives the claimed handoff. The Cauchy-continuous extension packet supplies the sink rows, and the Real uniform-structure packet supplies the positive-radius entourage route; neither carrier contains selected limits, quotient filters, countable choice, host function equality, or ambient Real completeness. □

Theorem 1632.10 (Cauchy-continuous extension Hausdorff pullback route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *packet that compares two exported extension rows through separated completion data must first read the sink rows* S, W, D, F, U *of Definition 1632.5, then the* $\mathsf{RealUniformStructure}^{\uparrow}$ *positive-radius route of chapter 1624, and then the* $\mathsf{HausdorffCompletionPullback}^{\uparrow}$ *packet of chapter 8585. The comparison cannot inspect selected limits, quotient filters, countable choice, host function equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousExtensionUp.CauchyContinuousExtensionCarrier hausdorff pullback route`

Proof. Start with Definition 1632.7. A consumer reaches an exported extension row only through the completion sink, so the rows S, W, D, F, U must be displayed before any separated comparison can be requested.

Apply Definition 1632.8. It keeps the dense regular-Cauchy source, finite windows, dyadic tolerance ledger, and Cauchy-continuous map row on the public route, excluding selected representatives and countable-choice schedules.

Use Definition 1632.9. This routes the Real-facing read through the positive-radius and metric-entourage surface of $\mathsf{RealUniformStructure}^{\uparrow}$ before any Real comparison is consumed.

Finally project the carrier of Definition 8585.1. The rows $S, W, D, P_0, P_1, H_0, H_1, U, T, C, Q, N$ force the two exported extension rows through the Hausdorff separation witnesses before the pullback comparison becomes public, and no carrier coordinate remains for quotient filters, host equality, selected limits, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuousExtension}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyContinuousExtension}^{\uparrow}$ packet is generated from a regular-Cauchy source row, finite StreamName windows, a DyadicRatCore tolerance ledger, a Cauchy-continuous map row, a CompletionUniversalProperty

extension row, a uniqueness hsame ledger, displayed transport and replay rows, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyContinuousExtension}\uparrow}$, seed)

Scope closed: Definition 1632.1, Definition 1632.2, Definition 1632.3, and Definition 1632.4 seed the finite Cauchy-continuous extension route over the L10 Real and Completion spine.

Formal verification: FormalStatus(NameCert$_{\text{CauchyContinuousExtension}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyContinuousExtensionUp.TasteGate.CauchyContinuousExtensionTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient completion, selected limits, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, regular-source handoff, Cauchy-continuous map control, completion-universal extension control, uniqueness-ledger exactness, and ledger non-escape for BEDC.Derived.CauchyContinuousExter

Chapter 1633

A Concrete Naming Certificate for $\mathsf{DyadicTailBall}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicTailBall}^{\uparrow}$ packet records the finite late-tail enclosure by which a regular Cauchy source enters a displayed dyadic ball before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads a seal. It binds the center-radius surface of chapter 311, the threshold and tolerance ledger of chapter 379, the late-tail route of chapter 1605, and the Real-specific Cauchy consumer of chapter 1590. The packet is a finite induction-friendly bridge from tail control to dyadic-ball containment; it is not a quotient stream, not a selected limit, not countable choice, not host real equality, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

The horizontal dependency route is substantive rather than decorative: chapter 379 supplies the late threshold and tolerance evidence, chapter 1605 turns that evidence into cofinite tail windows, chapter 311 records the resulting center-radius containment, and chapter 1590 is the downstream consumer that reads this finite late-tail containment bridge.

Definition 1633.1 (DyadicTailBall late-tail carrier)**.** *A* $\mathsf{DyadicTailBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, D, F, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{DyadicBall}^{\uparrow}$ *center-radius packet, carrying the dyadic center, positive radius, scheduled observation row, and containment ledger read from Definition 311.1.*
2. *D is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *threshold and tolerance ledger, recording the displayed late-tail threshold, dyadic tolerance budget, finite comparison rows, and seal-facing readback route.*
3. *F is the* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *route that exposes the cofinite late-tail neighbourhoods through the same threshold and tolerance rows before any downstream Cauchy consumer can inspect the packet.*
4. *R is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row, reading the late-tail windows, regular rational readback, dyadic tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal only after B, D, F have been displayed.*
5. *H records componentwise* hsame *transport for the dyadic ball row, tail-bound ledger, regular Cauchy tail-filter route, Real Cauchy criterion consumer, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a late-tail request through D, then through F, into the containment row of B, and finally into the consumer row R.*

7. *P is the* Pkg *provenance over* $\mathsf{DyadicBall}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{RegularCauchyTailFilter}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBall}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their componentwise hsame *transports, their displayed* Cont *routes, their* Pkg *provenance, and their local* NameCert *rows. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.TasteGate.DyadicTailBallTasteGate single carrier alignment`

Theorem 1633.2 (DyadicTailBall NameCert obligations)**.** *For every accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *carrier* $T = (B, D, F, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicTailBall}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic center-radius packet, dyadic tail-bound threshold and tolerance ledger, regular Cauchy tail-filter route, Real Cauchy criterion consumer, transport, replay, provenance, and local naming row;*

2. *tail-bound exactness, requiring the threshold and tolerance read by F and R to be the same displayed threshold and dyadic budget carried by D;*

3. *ball-containment exactness, requiring the late-tail observations admitted through F to enter the containment ledger of B before the consumer row R is exposed;*

4. *transport and replay stability, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the ordered route D, F, B, R;*

5. *provenance exhaustion, requiring P and N to name only the displayed rows and to export no quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1633.1. The visible packet consists of the dyadic ball row B, the tail-bound ledger D, the late-tail filter route F, the Real Cauchy criterion consumer R, and the structural rows H, C, P, N.

Tail-bound exactness is read from the order of D and F. The regular Cauchy tail-filter route may expose only the threshold and tolerance ledger already displayed by D, so no unrecorded tail selector or separate rational budget can enter the packet.

Ball-containment exactness is supplied by B. The late observations admitted through F are replayed through C into the center-radius containment ledger of the $\mathsf{DyadicBall}^{\uparrow}$ row before R can consume the packet.

Transport and replay are structural. Componentwise hsame in H preserves the four public rows D, F, B, R, and displayed Cont replay in C repeats the same route from late-tail request to dyadic-ball containment and then to the Real Cauchy criterion consumer.

Finally, provenance is exhausted by P and N. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness, the local certificate cannot export any such object. □

Theorem 1633.3 (DyadicTailBall regular Cauchy consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

Thus a consumer first reads the displayed late-tail threshold and dyadic tolerance ledger, then the regular Cauchy tail-filter route, then the dyadic center-radius containment packet, and only then the Real Cauchy criterion row with its terminal seal. The exported route contains no quotient stream, selected limit, countable-choice selector, host real equality, trichotomy, or ambient Real completeness assertion.

Proof. Start with an accepted packet $T = (B, D, F, R, H, C, P, N)$. Carrier projection exposes D as the only threshold and tolerance source available to the late-tail route.

Replay the continuation row C. The request first passes through D, then through F, so the regular Cauchy tail-filter reads the same finite threshold, tolerance, and comparison ledger rather than an external convergence predicate.

The next public row is the dyadic ball packet B. Its center-radius containment ledger receives the late-tail observations supplied by F, and its classifier reads only the displayed center, radius, schedule, observation, containment, transport, replay, provenance, and naming rows.

Only after this containment row has been replayed can the consumer inspect R. The Real Cauchy criterion row therefore receives a finite dyadic-ball late-tail packet rather than a quotient stream or a selected limit.

Transport through H and provenance through P, N preserve the same ordered route. The excluded quotient, choice, host-equality, trichotomy, and completeness objects are absent from the carrier inventory, so no consumer read can export them. □

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.TasteGate.DyadicTailBallCarrier regular cauchy consumer route`

Theorem 1633.4 (DyadicTailBall late-window stability)**.** *Let* $T = (B, D, F, R, H, C, P, N)$ *and* $T' = (B, D', F', R, H', C', P', N')$ *be accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packets sharing the same* $\mathsf{DyadicBall}^{\uparrow}$ *row* B *and the same* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row* R*. Suppose the second* $\mathsf{DyadicTailBound}^{\uparrow}$ *row* D' *refines the threshold and tolerance ledger of* D *through the* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *routes* F, F'*. Then every* $\mathsf{Real}^{\uparrow}$*-facing read available from* T *is available from* T' *through the same finite route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

The refinement exports no quotient stream, selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the two carriers. They share the dyadic ball row B and the Real Cauchy criterion consumer row R, so the only public late-tail change is the replacement of D, F by the refined pair D', F'.

Use the refinement hypothesis to replay every late-tail request through D' and F'. Because the refined threshold and tolerance ledger is served by the same regular Cauchy tail-filter route, each observation admitted by the first packet is still admitted before the shared containment ledger B is read.

The Real-facing consumer remains fixed. After B receives the refined late-tail observations, the same row R consumes the packet, so the public route has the same four stages as in Definition 1633.3.

Finally, replay H', C', P', N' componentwise. These rows transport, continue, source, and name the refined packet without adding any coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallWindowStability`

Theorem 1633.5 (DyadicTailBall monotone inclusion)**.** *Let $T = (B, D, F, R, H, C, P, N)$ and $T' = (B, D', F', R, H', C', P', N')$ be accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packets with the same* $\mathsf{DyadicBall}^{\uparrow}$ *row B and the same* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row R. Suppose D' refines the dyadic tail-bound ledger of D, and suppose the refined route preserves the same containment handoff from F' into B and the same Real Cauchy criterion handoff into R. Then every late-tail containment read accepted by T is accepted by T' through the same dyadic-ball containment row.*

Equivalently, tightening the displayed threshold or tolerance budget while preserving B and R gives a monotone inclusion of accepted containment reads. The inclusion exports no quotient stream, selected limit, countable-choice selector, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallMonotoneInclusion`

Proof. Project the two accepted carriers through Definition 1633.1. The public ball row B and Real Cauchy criterion row R are shared, so the containment target and terminal consumer boundary do not change.

Apply the regular Cauchy consumer route of Definition 1633.3. Every accepted containment read from T passes through the displayed order

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

The refinement hypothesis replaces D, F by D', F' without changing the ball containment row B or the consumer row R. Hence any late-tail observation already admitted through D, F is replayed through the refined threshold and tolerance ledger before it reaches the same containment row.

It remains to check that the refined packet exports only the same public route. This is exactly the late-window stability statement of Definition 1633.4: the refined rows H', C', P', N' transport, replay, source, and name the packet without adding quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness. □

Theorem 1633.6 (DyadicTailBall radius monotonicity)**.** *Let $T = (B, D, F, R, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packet. Suppose B' is obtained from the dyadic-ball row B by replacing the displayed radius row with a smaller displayed radius, while preserving the same center row, the same shared tail-bound ledger D, and the same* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row R. If the refined containment ledger still receives the late-tail observations admitted through F, then the refined packet factors through the same route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

Thus radius refinement is a finite containment-refinement read, not a new convergence principle, quotient stream, selected limit, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallRadiusMonotonicity`

Proof. Project the accepted packet. The only proposed change is the dyadic-ball radius coordinate inside B; the tail-bound ledger D, the regular late-tail route F, and the Real Cauchy criterion consumer R are shared.

Use the displayed radius refinement to form B'. Since the refined radius is smaller and the late-tail observations are assumed to enter the refined containment ledger, every containment read that reaches B' still arrives from the same D, F route.

Apply Definition 1633.5. It identifies the accepted containment reads after the radius replacement with reads through the same dyadic-ball containment row and the same downstream consumer boundary.

Finally, invoke late-window stability from Definition 1633.4. The structural rows transport, replay, source, and name the refined packet without adding quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness. □

Theorem 1633.7 (DyadicTailBall radius-monotone containment)**.** *Let $T = (B, D, F, R, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packet, and let $T' = (B', D, F', R, H', C', P', N')$ be an accepted refined packet with the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger D, the same* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *handoff shape, and the same* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row R. Suppose B' displays a smaller radius than B and the refined late-tail handoff still feeds the containment ledger of B' before R is read. Then every accepted refined containment read remains a read of the same finite* $\mathsf{DyadicTailBall}^{\uparrow}$ *route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

The containment refinement exports no quotient stream, selected limit, countable-choice selector, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with radius monotonicity from Definition 1633.6. The smaller displayed radius changes only the dyadic-ball radius coordinate, while the shared tail-bound ledger D and Real Cauchy criterion consumer R keep the public route fixed.

Apply monotone inclusion from Definition 1633.5. Since the refined handoff preserves the same dyadic tail-bound ledger and the same downstream consumer boundary, accepted containment reads for the refined packet are still reads through the displayed dyadic-ball containment row.

Use late-window stability from Definition 1633.4. The refined structural rows H', C', P', N' transport, replay, source, and name the same late-tail route without adding quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallRadiusMonotoneContainment`

Theorem 1633.8 (DyadicTailBall threshold refinement route)**.** *Let $T = (B, D, F, R, H, C, P, N)$ and $T' = (B, D', F', R, H', C', P', N')$ be accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packets with the same dyadic-ball row B, the same dyadic tolerance budget, the same regular Cauchy tail-filter handoff shape, and the same* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row R. Suppose D' tightens only the displayed threshold of D, and suppose the refined handoff still routes the admitted late-tail observations through F' before they enter B and R. Then every accepted containment read factors through the same finite route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

Thus threshold refinement is a late-window route refinement at fixed radius and fixed consumer boundary; it is not a quotient stream, selected limit, countable-choice selector, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the two accepted carriers. The dyadic-ball row B, the dyadic tolerance budget, and the Real Cauchy criterion row R are fixed, so the only public change is the displayed threshold inside D together with its regular Cauchy handoff F'.

Apply the regular Cauchy consumer route of Definition 1633.3. Both packets must read the same ordered route from tail-bound ledger to late-tail filter, then to dyadic-ball containment, and finally to the Real Cauchy criterion consumer.

Use late-window stability from Definition 1633.4. Tightening the threshold while keeping the tolerance budget and downstream rows fixed only delays the window at which the same containment evidence is replayed; it does not create a new radius, a new consumer row, or an external convergence predicate.

Finally apply monotone inclusion from Definition 1633.5. The refined threshold route preserves accepted containment reads through the same B and R, while H', C', P', N' transport, replay, source, and name only the displayed rows. Since quotient streams, selected limits, countable choice, trichotomy, host real equality, and ambient Real completeness are absent from the carrier inventory, the refined route cannot export them. □

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallThresholdRefinementRoute`

Theorem 1633.9 (Dyadic tail ball RealEquality consumer)**.** *Let $T = (B, D, F, R, H, C, P, N)$ be an accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packet. Suppose the packet is presented as a monotone late-tail ball route in which the finite tail window accepted by the* $\mathsf{DyadicTailBall}^{\uparrow}$ *carrier is the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window used by two regular-rational readbacks. Then T supplies exactly the dyadic tolerance ledger consumed by a* $\mathsf{RealEquality}^{\uparrow}$ *classifier row over those two readbacks.*

Equivalently, when both $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks are routed through the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the monotone tail-ball packet may be consumed as the tolerance side of the* $\mathsf{RealEquality}^{\uparrow}$ *classifier without adding quotient real equality, a selected limit, or any completeness datum.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallRealEqualityConsumer`

Proof. Start with the regular Cauchy consumer route of Definition 1633.3. It exposes the displayed order

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

Apply late-window stability from Definition 1633.4. Refining the late-tail window changes only the displayed threshold and tolerance ledger while preserving the same ball row and the same Real-facing consumer boundary.

Apply monotone inclusion from Definition 1633.5. The accepted containment reads of the original packet are still accepted after the refined route, so the dyadic ledger remains the public tolerance source for the shared window.

Finally route both regular-rational readbacks through the same displayed $\mathsf{StreamName}^{\uparrow}$ window and consume the classifier surface of chapter 1636. The $\mathsf{RealEquality}^{\uparrow}$ row receives the dyadic tolerance ledger already displayed by the $\mathsf{DyadicTailBall}^{\uparrow}$ packet; no quotient real equality, selected limit, or completeness datum is introduced. □

Theorem 1633.10 (Dyadic tail-ball RegSeqRat window consistency)**.** *Let $T_0 = (B_0, D, F_0, R, H_0, C_0, P_0, N_0)$ and $T_1 = (B_1, D, F_1, R, H_1, C_1, P_1, N_1)$ be accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packets feeding the same* $\mathsf{RegSeqRat}^{\uparrow}$ *late-tail consumer and sharing the displayed* $\mathsf{DyadicTailBound}^{\uparrow}$ *ledger D. If both packets route their*

ball-containment rows through the same displayed late-tail window before the Real-facing equality consumer is read, then the exposed tolerance route is the same finite route

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Thus the shared $\mathsf{RegSeqRat}^{\uparrow}$ *window is determined by the finite dyadic tail-bound and ball-containment rows, not by quotient streams, selected limits, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallRegSeqWindowConsistency`

Proof. Apply the regular Cauchy consumer route of Definition 1633.3. Both packets must first expose the same dyadic tail-bound ledger D before their late-tail filters and ball-containment rows can be consumed.

Use late-window stability from Definition 1633.4. Since the late-tail consumer is the same $\mathsf{RegSeqRat}^{\uparrow}$ read and the displayed tail-bound ledger is shared, both routes inspect the same finite window boundary before any Real-facing equality consumer appears.

Apply monotone inclusion from Definition 1633.5 and the threshold-refinement route of Definition 1633.8. These keep accepted containment reads fixed by the displayed dyadic tail-bound and ball-containment rows, even when the threshold presentation is refined.

Finally compare with Definition 1633.9. The Real-facing equality row can read only the already displayed tolerance ledger, so it cannot introduce quotient streams, selected limits, trichotomy, host real equality, or ambient completeness into the shared $\mathsf{RegSeqRat}^{\uparrow}$ window. □

Theorem 1633.11 (Dyadic tail-ball regular Cauchy apartness route)**.** *Let* $T_0 = (B_0, D, F_0, R, H_0, C_0, P_0, N_0)$ *and* $T_1 = (B_1, D, F_1, R, H_1, C_1, P_1, N_1)$ *be accepted* $\mathsf{DyadicTailBall}^{\uparrow}$ *packets sharing the displayed* $\mathsf{StreamName}^{\uparrow}$ *window used by two regular rational readbacks. Suppose the regular rational readbacks are separated by a displayed dyadic tolerance ledger carried through the shared tail-bound row* D*. Then every apartness-facing consumer of the pair factors through the finite route*

$$\mathsf{DyadicTailBound}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{DyadicBall}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

before any $\mathsf{RealEquality}^{\uparrow}$ *consumer is read. The route exports only the displayed tail threshold, dyadic tolerance ledger, regular Cauchy tail-filter handoff, dyadic-ball containment rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks; it exports no quotient stream equality, selected limits, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicTailBallUp.DyadicTailBallRegularCauchyApartnessRoute`

Proof. Start with the regular Cauchy consumer route of Definition 1633.3. Each accepted packet exposes its late-tail read in the order from $\mathsf{DyadicTailBound}^{\uparrow}$ through $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ and $\mathsf{DyadicBall}^{\uparrow}$ before a Real-facing row can inspect the packet.

Use the RegSeqRat window consistency theorem, Definition 1633.10. Because the two packets share the displayed $\mathsf{StreamName}^{\uparrow}$ window and the same dyadic tolerance ledger D, the regular rational readbacks are compared inside the common finite route ending at $\mathsf{RegSeqRat}^{\uparrow}$.

The displayed separation ledger is therefore read as a dyadic-tail and ball-containment fact before the Real-facing equality row appears. The RealEquality consumer of Definition 1633.9 can only consume that already displayed tolerance surface; it cannot insert a quotient stream equality or a selected limit.

Finally inspect the carrier inventory. The accepted packets contain no coordinate for quotient stream equality, selected limits, trichotomy, or ambient Real completeness, and the apartness-facing read is only a route through the displayed dyadic ledger and regular rational window. □

The late-window $\mathsf{DyadicTailBall}^{\uparrow}$ carrier is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301 and the $\mathsf{RealEquality}^{\uparrow}$ route of chapter 1636. Tail-ball reads expose their dyadic tail-bound, regular Cauchy tail-filter, and dyadic-ball containment rows before those sibling consumers inspect them, so they do not bypass the regular-rational or Real-equality rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTailBall}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicTailBall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1633.1, Definition 1633.2, and Definition 1633.3 seed the finite route from dyadic tail bounds through regular Cauchy late-tail filters into dyadic ball containment and the Real Cauchy criterion consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicTailBall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, tail-bound exactness, ball-containment exactness, transport stability, replay stability, Real Cauchy consumer factorization, and provenance exhaustion for BEDC.Derived.DyadicTailBallUp.

Constructive story: A $\mathsf{DyadicTailBall}^{\uparrow}$ packet is generated from a DyadicBall center-radius packet, a DyadicTailBound threshold and tolerance ledger, a RegularCauchyTailFilter route, a RealCauchyCriterion consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1634

A Concrete Naming Certificate for $\mathsf{WeierstrassApproximation}^{\uparrow}$

Following Weierstrass' approximation theorem in the Bishop–Bridges constructive reading, the $\mathsf{WeierstrassApproximation}^{\uparrow}$ packet records a finite polynomial-approximation surface for a continuous real-valued map on a compact real interval. It consumes the real-name endpoint surface of chapter 38, the interval-window rows of chapter 1625, the compact-uniform-continuity route of chapter 388, the polynomial code surface of chapter 50, the endpoint handoff discipline of chapter 463, and the finite tagged-sum mesh rows of chapter 1615. The packet names the inspected approximation ledger only; it is not an open-cover compactness theorem, a Stone–Weierstrass algebra theorem, a quotient function-space construction, a host supremum norm, a selector for infinite data, or a completeness principle for ambient $\mathsf{Real}^{\uparrow}$.

As a sibling dependency, the finite approximation ledger is a consumer of chapter 388 and chapter 50 before it is exposed to chapter 463.

Definition 1634.1 (Weierstrass approximation carrier)**.** *A* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, F, \varepsilon, P, S, M, E, T, C, R, Q)$$

with the following displayed rows:

1. *I is the compact real interval row, read through* $\mathsf{Real}^{\uparrow}$ *endpoints and the* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *interval-window service.*
2. *F is the continuous-map row over the same interval source, exposing only finite real-name windows on inspected inputs.*
3. *ε is the requested positive rational error row.*
4. *P is the displayed* $\mathsf{Poly}^{\uparrow}$ *row whose coefficient ledger is finite and whose evaluation route is read through the same* $\mathsf{Real}^{\uparrow}$ *arithmetic surface as F.*
5. *S is the finite interval-enclosure sample family, recording the inspected subintervals, sample points, and real-name windows used to compare F with P.*
6. *M is the compact-uniform-modulus row obtained from the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *route for F on I.*
7. *E is the uniform-error ledger saying that every displayed sample comparison is bounded by ε after the modulus row has selected the required mesh scale.*

8. *T stores componentwise* hsame *transport for the interval row, continuous-map row, error row, polynomial row, sample family, compact-uniform-modulus row, and uniform-error ledger.*

9. *C stores the displayed* Cont *routes from interval enclosures to samples, from samples to function and polynomial evaluations, and from those evaluations to the uniform-error ledger.*

10. *R is the* Pkg *provenance over* $\mathsf{WeierstrassApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{CompactUniformContinuit}$ $\mathsf{Poly}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *Q is the local* $\mathsf{NameCert}_{\mathsf{WeierstrassApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their displayed transports, their continuation routes, provenance, and local naming row. It has no coordinate for open-cover compactness, Stone–Weierstrass generality, quotient equality of functions, a host supremum norm, selected infinite sample data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1634.2 (Weierstrass approximation NameCert obligations)**.** *For every accepted* WeierstrassApproximatio *carrier*

$$W = (I, F, \varepsilon, P, S, M, E, T, C, R, Q),$$

the local $\mathsf{NameCert}_{\mathsf{WeierstrassApproximation}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the interval, continuous-map, error, polynomial, sample, modulus, uniform-error, transport, continuation, provenance, and local naming rows;*

2. *compact interval source compatibility, requiring every sample and endpoint read to pass through the same* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *route attached to I;*

3. *polynomial-row admissibility, requiring P to be a displayed* $\mathsf{Poly}^{\uparrow}$ *code with finite coefficient and evaluation ledgers rather than a host function;*

4. *uniform-error exactness, requiring E to bind the sample comparisons to the requested rational error row ε after the compact-uniform-modulus row M has fixed the mesh scale;*

5. *transport stability, requiring T and C to replay the same interval, sample, evaluation, and error rows under componentwise* hsame *movement;*

6. *ledger refusal, excluding open-cover compactness, Stone–Weierstrass algebra closure, quotient function spaces, host supremum norms, selected infinite data, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and any standard bridge beyond the displayed rows.*

Lean4 ✓ `BEDC.Derived.WeierstrassApproximationUp.WeierstrassApproximationTasteGate single carrier alignment`

Proof. Carrier admission is the projection of Definition 1634.1. The accepted packet exposes exactly the compact interval, continuous map, rational error, polynomial code, sample family, modulus, uniform-error ledger, transport, continuation, provenance, and local name rows.

Source compatibility starts with the interval row I. Every sample in S is read from the same finite interval-enclosure service, so the function row F and the polynomial row P are evaluated on the same inspected real-name windows.

Polynomial admissibility is the row P. Its coefficient and evaluation ledgers are finite $\mathsf{Poly}^{\uparrow}$ data, and the carrier gives no second coordinate for a host function or a quotient of functions.

Uniform-error exactness is the route through M and E. The compact-uniform-modulus row fixes the displayed mesh scale for the requested rational error, and E records the finite comparisons between the inspected F values and the inspected polynomial values.

Transport stability is componentwise. The rows in T move the interval, map, error, polynomial, sample, modulus, and error ledgers, while C replays the same comparison route after transport.

The refusal clauses follow from the inventory. No coordinate names an open cover, a Stone–Weierstrass algebra, a quotient function space, a host norm, an infinite selected sample family, ambient completeness, or a standard bridge. □

Theorem 1634.3 (Weierstrass approximation polynomial handoff)**.** *Every downstream approximation consumer of an accepted* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *carrier factors through the displayed route*

$$I, F, \varepsilon \longmapsto S, M \longmapsto P, E \longmapsto T, C, R, Q.$$

Consequently the consumer reads only the $\mathsf{RealIntervalEnclosure}^{\uparrow}$*,* $\mathsf{CompactUniformContinuity}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{UniformLimit}^{\uparrow}$*,* $\mathsf{RiemannSum}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows displayed by Definition 1634.1.*

Lean4 ✓ `BEDC.Derived.WeierstrassApproximationUp.WeierstrassApproximationPolynomialHandoff`

Proof. Begin with the accepted carrier $W = (I, F, \varepsilon, P, S, M, E, T, C, R, Q)$. The interval and map rows determine the inspected source windows, and the requested error row fixes the comparison threshold.

The sample family S is read through $\mathsf{RealIntervalEnclosure}^{\uparrow}$ and the compact-uniform-modulus row M is read through $\mathsf{CompactUniformContinuity}^{\uparrow}$. These rows determine the finite mesh on which the consumer may compare the continuous-map windows with the polynomial windows.

The polynomial row P supplies only the finite $\mathsf{Poly}^{\uparrow}$ coefficient and evaluation route. The error row E then records the comparison between those polynomial evaluations and the corresponding function evaluations, with $\mathsf{UniformLimit}^{\uparrow}$ and $\mathsf{RiemannSum}^{\uparrow}$ appearing only as downstream finite-window and mesh consumers named in the provenance row.

Finally T, C, R, Q transport, replay, source, and name the same route. Since the carrier has no other coordinate, every downstream approximation consumer factors through exactly these displayed BEDC rows and cannot read an open-cover proof, Stone–Weierstrass algebra closure, quotient function space, host supremum norm, selected infinite data, or ambient real completeness. □

Theorem 1634.4 (Weierstrass approximation UniformLimit consumer)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$*-facing read of an accepted* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *carrier first reads the compact interval row, continuous-map row, requested rational error, polynomial row, finite samples, compact-uniform modulus, and uniform-error ledger. The consumer boundary is therefore the route*

$$I, F, \varepsilon, P, S, M, E \longmapsto T, C, R, Q,$$

with $\mathsf{UniformLimit}^{\uparrow}$ *appearing only as the finite-window consumer named by the provenance row and by chapter 463.*

Lean4 ✓ `BEDC.Derived.WeierstrassApproximationUp.WeierstrassApproximationUniformLimitConsumer`

Proof. Start from the carrier of Definition 1634.1. Its rows $I, F, \varepsilon, P, S, M, E$ are the only interval, map, error, polynomial, sample, modulus, and uniform-error data exposed before structural replay.

Apply the polynomial handoff of Definition 1634.3. It forces every approximation consumer to pass through the same source-window, sample, modulus, polynomial, and error route before T, C, R, Q can transport, replay, source, and name the packet.

The $\mathsf{UniformLimit}^{\uparrow}$ consumer boundary is then exactly the sibling chapter chapter 463. Since the accepted carrier has no coordinate for a host supremum norm, quotient function space, selected infinite sample family, or ambient completeness principle, a uniform-limit-facing read receives only the displayed finite approximation ledger. □

Theorem 1634.5 (Weierstrass polynomial modulus ledger)**.** *Let $W = (I, F, \varepsilon, P, S, M, E, T, C, R, Q)$ be an accepted* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *packet. The packet exposes the polynomial approximation row P only after the compact-uniform-modulus row M fixes the finite mesh and the sample family S has been read through* $\mathsf{RealIntervalEnclosure}^{\uparrow}$*; the uniform-error ledger E then binds P to the requested rational error row ε.*

Lean4 ✓ `BEDC.Derived.WeierstrassApproximationUp.WeierstrassApproximationPolynomialModulusLedger`

Proof. Read the carrier of Definition 1634.1. The interval row I, continuous-map row F, requested error row ε, sample family S, compact-uniform-modulus row M, and error ledger E are all displayed coordinates of the accepted packet.

Apply the NameCert obligations of Definition 1634.2. Compact interval source compatibility forces S to be read through the same $\mathsf{RealIntervalEnclosure}^{\uparrow}$ service, while uniform-error exactness requires M to fix the finite mesh before E can compare the sampled function values with the polynomial row.

Finally use the polynomial handoff theorem Definition 1634.3. It places every downstream approximation read behind the route from I, F, ε through S, M and then through P, E. Thus E binds the displayed polynomial row P to the requested rational error, and no host supremum norm, quotient function space, selected infinite sample family, or ambient completeness row participates. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeierstrassApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{WeierstrassApproximation}^{\uparrow}$ packet is generated from a compact real interval row, continuous-map row, requested rational error row, displayed polynomial row, finite interval-enclosure samples, compact-uniform-modulus row, uniform-error ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeierstrassApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1634.1, Definition 1634.2, and Definition 1634.3 seed the finite polynomial-approximation surface over compact real intervals and routed continuous-map windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeierstrassApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Stone–Weierstrass generality, quotient function spaces, host supremum norms, selected infinite data, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval source compatibility, polynomial-row admissibility, uniform-error exactness, transport stability, ledger refusal, and polynomial handoff proofs for BEDC.Derived.WeierstrassAppro

Chapter 1635

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$

Following Bishop–Bridges constructive completion together with Kelley/Bourbaki completion terminology, the $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ packet records the finite BEDC certificate for comparing two ways of flattening a triple Cauchy-completion route. It sits over the monadic transformer of chapter 378, the unit route of chapter 1527, the counit route of chapter 1539, the idempotence boundary of **??**, and the minimality ledger of chapter 1606. It also consumes the finite-window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a route-comparison certificate for repeated completion, not a quotient-completion equality, not a selected representative, not host equality of completed objects, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

The same finite flattening route is consumed by the Bishop regular Cauchy completion handoff of chapter 1639, where the shared StreamName, RegSeqRat, DyadicRatCore, and Real rows are read before any completion-facing consumer can ask for a terminal seal.

Definition 1635.1 (CauchyCompletionAssociativity carrier)**.** *A* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, U, V, I, L, S, R, D, E, F, G, H, C, P, N)$$

with the following visible rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-transformer ledger used by all three displayed completion layers.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row admitting the source-facing data into the first completion surface.*
3. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning completion-facing reads to the displayed real-seal boundary.*
4. *I is the* $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$ *row flattening any adjacent pair of completion layers to the same finite route.*
5. *L is the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *row recording that the flattened route is the least displayed completion service boundary consumed by downstream readers.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the two flattening orders.*

7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplied after the shared stream windows and tolerance ledger have been consumed.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every precision request in the triple-completion route.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by both displayed flattening orders.*
10. *F records the left-associated route that first flattens the outer two completion layers and then flattens the result with the remaining layer.*
11. *G records the right-associated route that first flattens the inner two completion layers and then flattens the result with the remaining layer.*
12. *H records componentwise* hsame *transport for $M, U, V, I, L, S, R, D, E, F, G$ and the structural rows.*
13. *C records displayed* Cont *replay from a triple-completion read through either F or G and then through the shared U, V, I, L, S, D, R, E ledger.*
14. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}$ $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionMinimality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAssociativity}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected Cauchy representatives, host equality of completed objects, countable-choice schedules, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 1635.2 (CauchyCompletionAssociativity NameCert obligations)**.** *For an accepted* CauchyCompletionAss *carrier $A = (M, U, V, I, L, S, R, D, E, F, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAssociativity}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the monad, unit, counit, idempotence, minimality, stream-window, regular-readback, dyadic-tolerance, real-seal, left-route, right-route, transport, replay, provenance, and local naming rows;*
2. *left-associated flattening, requiring the route F to apply the idempotence row I and then the counit/minimality ledger V, L before the shared S, D, R, E surface is exposed;*
3. *right-associated flattening, requiring the route G to apply the same I, V, L ledger in the other finite order before the shared S, D, R, E surface is exposed;*
4. *classifier stability, requiring transported packets with the same $M, U, V, I, L, S, D, R, E, F, G$ rows to be identified by the displayed* hsame *and* Cont *data;*
5. *ledger non-escape, stating that no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness principle, ambient-completeness theorem, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported by the packet.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionAssociativityUp.CauchyCompletionAssociativityCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1635.1. The tuple lists exactly the completion transformer, unit, counit, idempotence, minimality, shared observation rows, the two association routes, transport, replay, provenance, and local name.

For the left-associated route, read F first. Its first replay flattens one adjacent pair of completion layers through I, then returns the resulting completion-facing read through V and the minimal service boundary L before the shared stream, dyadic, regular, and real-seal rows are consumed.

For the right-associated route, read G first. It uses the same idempotence, counit, and minimality rows in the other finite order, and the replay row C then sends the result to the same S, D, R, E surface.

Classifier stability is componentwise transport along H followed by replay through C. Since the carrier fields are exhausted by $M, U, V, I, L, S, R, D, E, F, G, H, C, P, N$, the excluded quotient, representative-selection, host-equality, choice, compactness, ambient-completeness, and real-completeness principles have no row from which a consumer could read them. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionAssociativityUp.TasteGate.CauchyCompletionAssociativityTasteGate single carrier alignment`

Theorem 1635.3 (CauchyCompletionAssociativity flattening route)**.** *Let $A = (M, U, V, I, L, S, R, D, E, F, G, H, C$ be an accepted* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ *carrier. The two finite routes from a triple completion to the single displayed completion factor through the same unit, counit, idempotence, and minimality ledger:*

$$F \longmapsto I \longmapsto V \longmapsto L \longmapsto S \longmapsto D \longmapsto R \longmapsto E,$$
$$G \longmapsto I \longmapsto V \longmapsto L \longmapsto S \longmapsto D \longmapsto R \longmapsto E.$$

Thus the visible associativity law is equality of the finite BEDC route through $M, U, V, I, L, S, D, R, E, H, C, P, N$, not equality of quotient completions, not host equality of completed metric objects, not a selected representative, and not countable choice or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionAssociativityUp.CauchyCompletionAssociativityFlatteningRoute`

Proof. Project the accepted carrier. The two route rows F and G are the only coordinates that distinguish the left-associated and right-associated flattening orders.

Read the ledger common to both routes. The idempotence row I supplies the adjacent-pair flattening step, the counit row V returns completion-facing reads to the displayed real-seal boundary, and the minimality row L records that downstream consumers receive the least displayed completion service boundary.

After that common ledger, both routes consume the same observation surface. The finite windows are named by S, each precision request is accounted for by D, the regular rational readback is R, and the terminal seal is E.

Finally M and U record the surrounding completion-transformer and source-admission rows, while H, C, P, N transport, replay, source, and name the same packet. No further coordinate remains for quotient equality, host equality, selected representatives, countable choice, or ambient completeness, so the two routes agree exactly on the displayed finite ledger. □

Theorem 1635.4 (CauchyCompletionAssociativity two-route coherence)**.** *Let $A = (M, U, V, I, L, S, R, D, E, F, G,$ and $A' = (M, U, V, I, L, S, R, D, E, F', G', H, C, P, N)$ be accepted* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ *carriers sharing the completion-transformer, unit, counit, idempotence, minimality, StreamName, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and local naming rows. Suppose the left and right route rows F, G, F', G' replay through the same displayed structural rows H, C, P, N. Then every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of either carrier receives the same finite ledger*

$$M, U, V, I, L, S, D, R, E, H, C, P, N$$

before any terminal seal is inspected.

Thus the comparison is a coherence statement about the displayed BEDC ledger. It is not quotient completion equality, not a selected representative comparison, not host equality of completed objects, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionAssociativityUp.CauchyCompletionAssociativityCarrier route coherence`

Proof. Project the two accepted carriers. By hypothesis, the non-route rows agree as displayed data: both packets expose the same $M, U, V, I, L, S, R, D, E, H, C, P, N$ rows before either association route is read.

Apply the NameCert obligations of Definition 1635.2. The obligations force carrier admission, left-associated flattening, right-associated flattening, classifier stability, and ledger non-escape to be read only from the displayed rows of each accepted packet.

Use the flattening-route theorem Definition 1635.3. It sends the left and right route rows through the same I, V, L, S, D, R, E ledger, while M, U, H, C, P, N supply the surrounding completion transformer, source admission, transport, replay, provenance, and local name.

Since F, G, F', G' replay through the same H, C, P, N rows, no Real-facing or Completion consumer can distinguish the carriers before the terminal seal by asking for quotient completion equality, a selected representative, host equality, countable choice, or ambient completeness. The only available comparison is the shared finite ledger in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionAssociativity}^{\uparrow}}$ ·

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAssociativity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1635.1, Definition 1635.2, and Definition 1635.3 seed the finite associativity certificate for flattening triple Cauchy-completion routes through the shared unit, counit, idempotence, minimality, StreamName, RegSeqRat, DyadicRatCore, and Real ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionAssociativity}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyCompletionAssociativityUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host equality of completed metric objects, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, left- and right-associated flattening, classifier stability, and ledger non-escape for BEDC.Derived.CauchyCompletionAssociativityUp.

Constructive story: The construction starts with a triple-completion read, compares the left- and right-associated finite flattening rows, routes both through the same unit, counit, idempotence, and minimality ledger, and then exposes the shared StreamName, DyadicRatCore, RegSeqRat, and Real seal surface.

Chapter 1636

Real Equality Name Certificate

Following Bishop and Bridges, the $\mathsf{RealEquality}^{\uparrow}$ packet records equality of real names as regular-Cauchy closeness witnessed through finite tolerance requests, not as quotient equality of completed streams. It sits between the terminal real-name seal of chapter 38, the regular-rational source route of chapter 301, the finite-window discipline of chapter 126, the dyadic tolerance core of chapter 291, and the regular-Cauchy uniformity boundary of chapter 1628. The chapter gives downstream RealAlgOrder, metric, and completion-facing certificates a named equality boundary that remains inside displayed BEDC rows.

Definition 1636.1 (RealEquality carrier)**.** *A* $\mathsf{RealEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, S_X, S_Y, R_X, R_Y, W, D, U, C, H, K, P, N)$$

with the following displayed rows:

1. *X and Y are the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows whose equality boundary is being inspected.*
2. *S_X and S_Y are the two* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows admitted by those seals.*
3. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback rows exposed through S_X and S_Y.*
4. *W is the shared finite window of precision requests on which both regular names are compared.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, and comparison budget for every request in W.*
6. *U is the* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *closeness row asserting that the two regular-rational readbacks meet the displayed dyadic tolerance schedule.*
7. *C records the classifier verdict obtained from W, D, U before either terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
8. *H stores componentwise* hsame *transport for the seals, windows, readbacks, tolerance ledger, uniformity row, classifier row, provenance, and local name.*
9. *K stores displayed* Cont *replay from X, Y through $S_X, S_Y, R_X, R_Y, W, D, U, C$ and back to the public equality boundary.*
10. *P records* Pkg *provenance over* $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$ *row.*

The packet has no coordinate for host real equality, quotient classes of streams, trichotomy, selected limits, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealEqualityUp.TasteGate.RealEqualityTasteGate single carrier alignment`

Theorem 1636.2 (RealEquality classifier stability)**.** *For every accepted* $\mathsf{RealEquality}^{\uparrow}$ *carrier* $E = (X, Y, S_X, S_Y, R_X, R_Y, W, D, U, C, H, K, P, N)$*, the classifier row* C *is stable under refinement of the displayed finite observation window whenever the refined window factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The public equality read therefore depends only on the finite regular-Cauchy closeness route* W, D, U*, not on host equality of* X *and* Y*.*

Lean4 ✓ `BEDC.Derived.RealEqualityUp.RealEqualityCarrier classifier stability`

Proof. Proceed by induction over the finite observation window W. The empty window contributes no classifier read, so the verdict is carried only by the displayed structural rows H, K, P, N.

For a one-request extension, the new request is compared by first reading the two $\mathsf{RegSeqRat}^{\uparrow}$ rows through R_X and R_Y, then consulting the dyadic tolerance ledger D. The row U supplies the regular-Cauchy closeness witness at that dyadic radius, so the classifier verdict is determined before either terminal $\mathsf{Real}^{\uparrow}$ seal can be inspected.

The induction step appends this one-request argument to the previous finite window. Componentwise hsame transport in H and replay in K preserve the displayed rows, while P and N record the same source and local name. Thus every finite refinement that factors through the same readbacks and dyadic ledger yields the same public equality boundary. □

Theorem 1636.3 (RealEquality NameCert boundary)**.** *For every accepted* $\mathsf{RealEquality}^{\uparrow}$ *carrier, the local* $\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window classifier stability, dyadic tolerance ledger exactness, regular-Cauchy closeness replay, real-seal non-escape, and provenance-local-name transport. No NameCert obligation may replace these rows by* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, quotient equality, selected limits, countable choice, or host real equality.*

Lean4 ✓ `BEDC.Derived.RealEqualityUp.RealEqualityCarrier namecert boundary`

Proof. Carrier admission is projection from Definition 1636.1. The only visible coordinates are the two $\mathsf{Real}^{\uparrow}$ seals, their $\mathsf{StreamName}^{\uparrow}$ windows, their $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the shared finite observation window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegularCauchyUniformity}^{\uparrow}$ closeness row, the classifier row, and the structural rows H, K, P, N.

Classifier stability is supplied by Definition 1636.2. Its induction over finite observation windows gives the finite-window part of the local certificate, and its dyadic ledger argument gives the exact tolerance route.

Regular-Cauchy closeness replay is the route through U after W and D have been displayed. The terminal seals X and Y are consumed only after that replay, so the public boundary cannot export host equality or quotient classes.

The provenance and local-name rows name the same packet and do not add analytic coordinates. Since the carrier contains no row for $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient equality, selected limits, countable choice, or host real equality, none of those principles can appear as a local $\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$ obligation. □

Theorem 1636.4 (RealEquality public boundary)**.** *Every downstream equality consumer of* $\mathsf{RealEquality}^{\uparrow}$ *receives exactly the finite route*

$$(X, Y, S_X, S_Y, R_X, R_Y, W, D, U, C, H, K, P, N)$$

and may use the public equality boundary only after the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *closeness row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row have been displayed.*

The boundary exports no host real equality, quotient class, selected limit, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealEqualityUp.RealEqualityCarrier public boundary`

Proof. Begin with classifier stability, Definition 1636.2. It fixes each public verdict by the finite observation window, the two regular readbacks, and the dyadic tolerance ledger rather than by host equality of the terminal seals.

Next apply the NameCert boundary of Definition 1636.3. That theorem exhausts the local certificate by the carrier admission, finite-window classifier stability, dyadic tolerance exactness, regular-Cauchy closeness replay, real-seal non-escape, provenance, and local-name transport rows.

Project the carrier of Definition 1636.1. It displays the two terminal Real seals only together with their finite StreamName windows, RegSeqRat readbacks, shared request window, dyadic tolerance ledger, uniformity row, classifier row, and structural rows. Thus every downstream public consumer receives the displayed BEDC boundary and no excluded analytic coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1636.1, Definition 1636.2, Definition 1636.3, and Definition 1636.4 bind the finite regular-Cauchy equality boundary for terminal RealUp names to one displayed carrier route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealEquality}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealEqualityUp.RealEqualityCarrier\_namecert\_boundary`

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, trichotomy, quotient equality, selected limits, countable choice, host real equality, or bridge checking.

Upgrade path: Public closure requires a public equality surface for the same carrier, classifier stability, dyadic tolerance, RegSeqRat readback, Real seal, hsame, Cont, Pkg, and NameCert rows.

Constructive story: A $\mathsf{RealEquality}^{\uparrow}$ packet is generated from two terminal RealUp seals, finite StreamName windows, RegSeqRat readbacks, one finite observation window, a DyadicRatCore tolerance ledger, a RegularCauchyUniformity closeness row, a classifier verdict, componentwise transport, continuation replay, package provenance, and local naming.

Chapter 1637

A Concrete Naming Certificate for $\mathsf{RiemannStieltjes}^{\uparrow}$

Following Riemann and Stieltjes, the $\mathsf{RiemannStieltjes}^{\uparrow}$ packet records a finite tagged-sum integration surface in which a regulated integrand is read against a bounded-variation integrator. It sits over the variation ledger of chapter 398, the regulated-integral handoff of chapter 1557, the finite tagged-sum surface of chapter 1615, and the dyadic step approximation rows of chapter 428. The packet names only the displayed finite rows needed for a constructive Riemann–Stieltjes read; it does not introduce a measure space, a completed integrator, a host function quotient, or an ambient choice principle. As a downstream integration consumer, this packet also reads the calculus root package of chapter 3632: the tagged-sum row is an integration-facing use of the finite derivative/integral/limit substrate, not a separate host integration principle.

Definition 1637.1 (Riemann–Stieltjes carrier)**.** *A* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, A, T, S, I, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the regulated integrand row, exposing only the finitely requested approximation windows and endpoint reads.*
2. *A is the bounded-variation integrator row, carrying the displayed partition-variation ledger and transported endpoint jumps.*
3. *T is the tagged partition and* $\mathsf{RiemannSum}^{\uparrow}$ *row, with each tag contained in its displayed interval cell.*
4. *S is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation row used to replace the regulated integrand on the displayed refinement.*
5. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row reached after the step approximation, variation ledger, and tagged-sum rows are present.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for the final accepted integral read.*
7. *H records componentwise* hsame *transport for the integrand, integrator, partition, step approximation, handoff, endpoint seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from partition cells to tag reads, from tag reads to step approximants, from variation jumps to finite sums, and from the handoff row to the* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *P is the* Pkg *provenance over* RiemannStieltjes$^{\uparrow}$, RegulatedIntegral$^{\uparrow}$, BoundedVariation$^{\uparrow}$, RiemannSum$^{\uparrow}$, DyadicStepFunction$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RiemannStieltjes}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, A, T, S, I, E, H, C, P, N$. It has no coordinate for Lebesgue integration, measure completion, selected limits, host real equality, a quotient of integrands or integrators, or a standard bridge.

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesTasteGate single carrier alignment`

Theorem 1637.2 (Riemann–Stieltjes NameCert obligations)**.** *For an accepted* RiemannStieltjes$^{\uparrow}$ *carrier $R = (F, A, T, S, I, E, H, C, P, N)$, the local* NameCert$_{\text{RiemannStieltjes}^{\uparrow}}$ *surface consists of exactly five rows:*

1. *carrier admission for the displayed tuple $F, A, T, S, I, E, H, C, P, N$;*
2. *partition-refinement stability for the tagged* RiemannSum$^{\uparrow}$ *row T and dyadic step row S;*
3. *bounded-variation ledger control for the integrator row A and its transported endpoint jumps;*
4. *regulated-integral handoff through I after the finite sum and variation rows have been displayed;*
5. *no-escape from host measure data, completion data, choice data, hidden function quotients, and ambient real equality.*

These rows are read through the finite route

$$F, A \longmapsto T \longmapsto S \longmapsto I \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesNameCertObligations`

Proof. Carrier admission is the projection of Definition 1637.1. The rows F and A expose the regulated integrand and bounded-variation integrator before any tagged sum is accepted.

Partition-refinement stability is checked on the displayed row T. A common refinement transports tags and cell containment through the visible hsame rows, and the dyadic step row S reads the same refined cells without adding a host-level function comparison.

Bounded-variation ledger control is supplied by A. The integrator contributes only transported endpoint jumps and finite variation ledgers, so every summand is controlled by displayed bounded-variation data rather than by a measure or completed variation object.

The handoff row I is downstream of T and S. Continuation replay through C rebuilds the same path, while P and N record provenance and local naming. Since the excluded host objects do not occur among the carrier coordinates, the listed five rows exhaust the certificate obligations. □

Theorem 1637.3 (Riemann–Stieltjes regulated-integral handoff)**.** *Every accepted* RiemannStieltjes$^{\uparrow}$ *sum read factors through the displayed dyadic-step and regulated-integral rows before reaching the* Real$^{\uparrow}$ *seal. Concretely, a consumer of the endpoint row E must first read the regulated integrand row F, the bounded-variation integrator row A, the tagged sum row T, the dyadic step approximation row S, and the regulated-integral handoff row I.*

Proof. Project an accepted carrier as $R = (F, A, T, S, I, E, H, C, P, N)$. The rows F and A supply the only integrand and integrator data available to the packet, and T records the finite tagged partition on which those data are read.

The row S is the dyadic step approximation surface attached to that displayed partition. Thus a downstream read cannot pass directly from tags to a real endpoint seal; it must first consume the step row that makes the regulated integrand finite on the refinement.

The row I then packages the regulated-integral handoff from the step approximation and bounded-variation ledger. Only after this handoff is displayed does E expose a $\mathsf{Real}^{\uparrow}$ endpoint seal. Componentwise transport through H and replay through C preserve this order, and the package and local naming rows add no bypass coordinate. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier regulated integral handoff`

Theorem 1637.4 (Riemann–Stieltjes Darboux mesh refinement)**.** *Every accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier refines a finite Darboux mesh through the displayed* $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. For a carrier*

$$R = (F, A, T, S, I, E, H, C, P, N),$$

any downstream Darboux or tagged-sum consumer must read the refined mesh from T and S, the integrator control from A, the regulated-integral handoff from I, and the endpoint seal from E. The packet exports no selected limit, quotient of tagged sums, host integrand quotient, host integrator quotient, or ambient real equality beyond those displayed rows.

Proof. Project the accepted carrier as $R = (F, A, T, S, I, E, H, C, P, N)$. The only partition data available to the consumer is the finite tagged-sum row T, and the only step approximation data available on a refinement is the dyadic step row S.

Refine the displayed mesh by replaying the continuation row C. Each cell containment, tag read, and dyadic step replacement is transported componentwise through H, so the refinement is a BEDC row calculation rather than a host-level limit selection.

The bounded-variation row A controls the integrator contribution on the refined cells. Its endpoint jumps and variation ledger are finite rows, and the regulated-integral handoff row I can be read only after A, T, S have all been displayed.

Finally, P and N route the refined mesh through the local package and classifier. Since equality is mediated only by H and by the local $\mathsf{NameCert}_{\mathsf{RiemannStieltjes}^{\uparrow}}$ comparison, no quotient of tagged sums, selected limit, or ambient real equality is exported by the carrier. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesDarbouxMeshRefinement`

Theorem 1637.5 (Riemann–Stieltjes Darboux integral consumer route)**.** *Every* $\mathsf{DarbouxIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier*

$$R = (F, A, T, S, I, E, H, C, P, N)$$

must read the displayed bounded-variation row A, tagged-sum row T, dyadic-step row S, regulated-integral handoff I, Real seal E, and structural rows H, C, P, N before it treats the packet as a Darboux-facing integral route. The consumer receives a finite Stieltjes mesh with bounded-variation control and regulated handoff; it receives no selected limiting partition, host integrand quotient, host integrator quotient, measure completion, or ambient real equality.

Proof. Start from the NameCert obligations of Definition 1637.2. They expose carrier admission, tagged-sum refinement stability, bounded-variation ledger control, regulated-integral handoff, and local nonescape as the finite rows of the accepted packet.

Apply the regulated-integral handoff theorem, Definition 1637.3. The Darboux-facing consumer cannot reach the endpoint seal E directly; it must first read the integrand row F, the bounded-variation row A, the tagged-sum row T, the dyadic-step row S, and the handoff row I.

Apply Darboux mesh refinement, Definition 1637.4. The mesh part of the consumer is therefore the displayed T, S refinement with integrator control supplied by A and the regulated route supplied by I.

Finally, transport, replay, provenance, and local naming are exactly H, C, P, N. Since the carrier has no coordinate for selected limiting partitions, host quotients, measure completion, or ambient real equality, every $\mathsf{DarbouxIntegral}^{\uparrow}$ consumer remains on the displayed Riemann–Stieltjes route. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesDarbouxIntegralConsumerRoute`

Theorem 1637.6 (Riemann–Stieltjes bounded-variation jump boundary)**.** *Every accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier reads integrator jump data only through the displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *row before the tagged-sum and regulated-integral rows are consumed. For*

$$R = (F, A, T, S, I, E, H, C, P, N),$$

the order of every jump-facing read is

$$A \longrightarrow T, S \longrightarrow I \longrightarrow E.$$

The route cannot replace bounded-variation evidence by a measure space, selected limit, host function quotient, or ambient choice.

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier bounded variation jump boundary`

Proof. Project the carrier rows $F, A, T, S, I, E, H, C, P, N$ from Definition 1637.1. The integrator row A is the only displayed source of bounded-variation evidence and transported endpoint jumps.

Use Definition 1637.2. It places bounded-variation ledger control before the regulated-integral handoff, so a tagged-sum read cannot consume jump data except through A.

Apply Definition 1637.4. The finite mesh and dyadic step rows T, S are read after the bounded-variation row and before the handoff row I, and the endpoint seal E remains downstream of that handoff.

The structural rows H, C, P, N transport, replay, source, and name the same order. Since the packet has no coordinate for measure spaces, selected limits, host function quotients, or ambient choice, the displayed bounded-variation jump boundary cannot be replaced by any of them. □

Theorem 1637.7 (Riemann–Stieltjes refinement-window exhaustion)**.** *For every accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier*

$$R = (F, A, T, S, I, E, H, C, P, N),$$

every downstream refinement-window read is exhausted by the displayed rows $A, T, S, I, E, H, C, P, N$*. In particular, no consumer can bypass the* $\mathsf{RiemannSum}^{\uparrow}$ *row* T*, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row* S*, the* $\mathsf{BoundedVariation}^{\uparrow}$ *row* A*, the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *row* I*, or the* $\mathsf{Real}^{\uparrow}$ *seal* E *when reading a refinement of the Riemann–Stieltjes packet.*

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier refinement window exhaustion`

Proof. Project the accepted carrier from Definition 1637.1. The only refinement-facing rows available to a consumer are the bounded-variation integrator row A, the tagged-sum row T, the dyadic step row S, the regulated-integral handoff row I, the endpoint seal E, and the structural rows H, C, P, N.

Apply Definition 1637.3. It forces every endpoint-facing read to pass through F, A, T, S, I before E is visible, so the endpoint seal cannot be used as a shortcut around the displayed finite rows.

Apply Definition 1637.4 and Definition 1637.6. The former confines mesh refinement to the tagged-sum and dyadic-step route, while the latter places every jump-facing integrator read behind the bounded-variation row A.

The rows H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no coordinate for a hidden refinement oracle, measure completion, selected limiting partition, host function quotient, or ambient real equality, the listed rows exhaust every downstream refinement-window read. □

Theorem 1637.8 (Riemann–Stieltjes regulated-integrator transport)**.** *Every* RiemannStieltjes$^{\uparrow}$ *read that is consumed as a regulated-integral handoff must pass through the bounded-variation integrator row A and the tagged-sum row T before the handoff row I is visible. For an accepted carrier*

$$R = (F, A, T, S, I, E, H, C, P, N),$$

the route to I is the displayed BEDC row route

$$F, A, T, S, H, C, P, N \longrightarrow I.$$

It cannot be replaced by a host measure integral, a quotient of integrators, a selected limiting partition, or an unrecorded real equality.

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier regulated integrator transport`

Proof. Project the carrier from Definition 1637.1. The regulated integrand row F and the bounded-variation integrator row A are the only sources from which the packet can read the two integration inputs.

Use the tagged-sum and step rows next. The row T displays the finite tagged partition, while S records the dyadic step approximation used by the regulated-integral handoff.

The handoff row I is therefore downstream of F, A, T, S. Componentwise transport through H and continuation replay through C preserve the same order, and P, N record provenance and local naming without adding any bypass coordinate.

Since the carrier has no coordinate for measure integration, integrator quotients, selected limiting partitions, or host real equality, every regulated-integrator transport consumed by the packet remains on this finite route. □

Theorem 1637.9 (Riemann–Stieltjes bounded-variation ledger)**.** *For every accepted* RiemannStieltjes$^{\uparrow}$ *carrier $R = (F, A, T, S, I, E, H, C, P, N)$, each integrator jump, variation read, and endpoint contribution used by the tagged sum is consumed as a visible* BHist *ledger row inside A before it is transported through H, C, P, N. The packet exports no hidden variation functional, measure-space control object, selected jump enumeration, or quotient of integrators.*

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier bounded variation ledger`

Proof. Read the bounded-variation component A from the carrier. By Definition 1637.6, jump-facing reads begin at A and cannot enter the tagged-sum route from any outside source.

The finite tagged-sum row T consumes only the displayed cells and tags, while the dyadic step row S supplies the corresponding finite integrand approximants. Thus every variation contribution used in the sum is already a visible row before the handoff row I is read.

Transport and replay do not create new variation data. The rows H and C move and replay the same ledger entries, while P and N identify the package and local NameCert surface. The excluded host objects have no coordinate in R, so the ledger is exactly the displayed bounded-variation row. □

Theorem 1637.10 (Riemann–Stieltjes mesh refinement stability)**.** *If an accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier is read along a mesh refinement, the refined packet preserves the same componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance route over* $F, A, T, S, I, E, H, C, P, N$*. Refinement may subdivide the visible tagged partition row, but it cannot introduce a new integrator, a new regulated-integral handoff, host function equality, selected limits, or a measure-space bridge.*

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier mesh refinement stability`

Proof. Start with the Darboux mesh-refinement theorem, Definition 1637.4. It confines every refinement-facing read to the displayed rows T, S, A, I, E together with the structural rows.

Refining the mesh changes only the visible tagged partition and the corresponding dyadic step rows. The bounded-variation ledger remains the row A, and the regulated-integral handoff remains the row I reached after the refined finite data are displayed.

The structural route is unchanged. Componentwise hsame transport is recorded by H, continuation replay by C, and provenance by P before the local naming row N accepts the refined packet. Because no host equality, selected limit, new integrator, or measure bridge occurs among the coordinates, refinement preserves the same finite BEDC route. □

Theorem 1637.11 (Riemann–Stieltjes bounded-variation partition coverage)**.** *For every accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier*

$$R = (F, A, T, S, I, E, H, C, P, N),$$

each cell of the finite BHist *partition row* T *is covered by a bounded-variation jump or oscillation row displayed inside* A *before the regulated-integral handoff* I *is public. The coverage is stable under componentwise* hsame *transport, replayed by* Cont *through* C*, sourced by the* Pkg *provenance row* P*, and named by the local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjes}^{\uparrow}}$ *row* N*.*

Proof. Project the carrier and read the finite partition row T. Its cells and tags are BHist entries, so coverage can be checked cell by cell rather than by a host partition object.

Induct over the displayed partition refinement rows. The base row is the original tagged partition, where Definition 1637.9 supplies the integrator jump and variation entries from A. In the refinement step, the subdivided cells inherit the same bounded-variation control by the mesh-refinement stability theorem and the bounded-variation jump boundary.

The structural rows preserve the coverage. Transport through H moves the displayed jump and oscillation rows componentwise, replay through C keeps the cell-to-jump route in the same order, and P, N record the package source and local certificate name. Thus I can only read a partition whose cells are already covered by the displayed bounded-variation rows. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier bounded variation partition coverage`

Theorem 1637.12 (Riemann–Stieltjes regulated-integral mesh handoff)**.** *Every mesh-facing read of an accepted* RiemannStieltjes$^{\uparrow}$ *carrier reaches the* RegulatedIntegral$^{\uparrow}$ *handoff row I only after the finite* BHist *partition row T, the bounded-variation jump and oscillation rows in A, and the dyadic step approximation row S have been displayed. The route is invariant under* hsame *transport, replayed by* Cont*, sourced by* Pkg *provenance, and accepted by the local* NameCert *row.*

Proof. Project $R = (F, A, T, S, I, E, H, C, P, N)$. The endpoint seal E is downstream of I, and the handoff theorem already places I after the integrand, integrator, tagged-sum, and dyadic-step rows.

Induct over the finite mesh refinement rows. At the base mesh, the partition row T and step row S are displayed directly, and the bounded-variation partition coverage theorem supplies the jump and oscillation rows from A. At a refinement step, Definition 1637.10 preserves the same A, T, S, I order while subdividing only the visible partition row.

Finally transport and replay do not change the route. The row H moves each displayed mesh, jump, oscillation, and step entry componentwise; C replays the handoff order; and P, N provide provenance and local naming. Hence a regulated-integral consumer cannot reach I without first reading the displayed finite mesh and bounded-variation control. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier regulated integral mesh handoff`

Theorem 1637.13 (Riemann–Stieltjes step-integrator exactness)**.** *For an accepted* RiemannStieltjes$^{\uparrow}$ *carrier $R = (F, A, T, S, I, E, H, C, P, N)$, each dyadic step contribution consumed by the handoff row I is exact with respect to the bounded-variation integrator row A: the value is computed from the displayed step value in S, the corresponding finite* BHist *partition cell in T, and the jump or oscillation ledger row in A. No hidden measure integral, selected limiting partition, host integrator quotient, or ambient real equality is used.*

Proof. Read the step row S together with the partition row T. Each step contribution names a displayed cell and a displayed step value, so the integrator contribution must be supplied by the bounded-variation ledger already present in A.

Proceed by induction over the displayed partition refinement rows. The base contribution is exact by direct projection from S, T, A: the step value is paired with the integrator jump or oscillation entry carried by the same cell. In the refinement step, the mesh-refinement stability theorem transports the subdivided step entries, while bounded-variation partition coverage supplies the matching integrator rows for every child cell.

The remaining structural rows only preserve this calculation. Componentwise hsame transport in H keeps step values, cells, and integrator ledger rows aligned; Cont replay through C repeats the same finite route; and P, N record provenance and local NameCert acceptance. The excluded host objects are absent from the carrier, so step-integrator exactness is a finite BEDC row calculation. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier step integrator exactness`

Theorem 1637.14 (Riemann–Stieltjes regulated-integrator variation exactness)**.** *For an accepted* RiemannStieltjes$^{\uparrow}$ *carrier $R = (F, A, T, S, I, E, H, C, P, N)$, every regulated-integrator variation contribution consumed by I is determined by the displayed bounded-variation row A, tagged partition row T, dyadic step row S, and structural rows H, C, P, N. No measure-space integral, selected limiting partition, host integrator quotient, or ambient real equality supplies variation data.*

Proof. Project the accepted carrier as $R = (F, A, T, S, I, E, H, C, P, N)$. The bounded-variation row A is the only component that carries integrator jumps, oscillation ledgers, and finite variation reads, while T and S identify the cells and step values on which those reads are consumed.

Use Definition 1637.9 and Definition 1637.11. They place every variation contribution inside A before the handoff row I can read the corresponding partition cell, and they keep each displayed cell of T covered by a visible bounded-variation ledger row.

Mesh-refinement stability then preserves the same rows under subdivision. Apply Definition 1637.10 to transport the refined cells through H and replay the finite route through C without adding a selected limiting partition.

Finally use Definition 1637.13. The step contribution is computed from the displayed rows S, T, A, and the remaining rows P, N only record provenance and local naming. Since the carrier has no coordinate for a measure-space integral, host quotient of integrators, or ambient real equality, the variation contribution is exact as a BEDC row calculation. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier regulated integrator variation exactness`

Theorem 1637.15 (Riemann–Stieltjes tagged-partition induction)**.** *For an accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier* $R = (F, A, T, S, I, E, H, C, P, N)$*, let a property of refinement-window reads be one that is evaluated only through the displayed rows* T, S, A, H, C, P, N*. If the property holds on the base tagged partition row* T*, and if it is preserved whenever one finite cell is subdivided through the same dyadic step row* S*, bounded-variation ledger* A*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N*, then the property holds for every displayed finite refinement consumed by the handoff row* I*.*

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier tagged partition induction`

Proof. Induct over the finite BHist refinement ledger consumed by I. The base case is the displayed tagged partition row T: the tag containment, dyadic step data, and integrator control are read by projection from T, S, A.

For the refinement step, subdivide one visible cell. By Definition 1637.10, the refined read preserves the same componentwise transport, replay, and provenance route over $F, A, T, S, I, E, H, C, P, N$.

Use Definition 1637.11 to keep each child cell covered by the bounded-variation ledger in A. Then apply Definition 1637.13 to identify the step contribution on the subdivided cells through the displayed rows rather than through a hidden limit.

The induction hypothesis applies to the remaining cells, and the stated preservation clause applies to the subdivided cell. Since I consumes only finite refinements produced by this ledger, every displayed refinement-window read satisfies the property. □

Theorem 1637.16 (Riemann–Stieltjes scoped ledger)**.** *For every accepted* $\mathsf{RiemannStieltjes}^{\uparrow}$ *carrier* $R = (F, A, T, S, I, E, H, C, P, N)$*, every integral-facing consumer of the tagged-partition result factors through the concrete BEDC dependency ledger*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{DyadicStepFunction}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{RegulatedIntegral}^{\uparrow}, \mathsf{RiemannSum}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

before the endpoint seal E *is public. In particular, the carrier read cannot bypass the bounded-variation integrator row* A*, the dyadic step row* S*, the tagged* $\mathsf{RiemannSum}^{\uparrow}$ *row* T*, the regulated-integral handoff row* I*, componentwise transport* H*, replay* C*, provenance* P*, or the local naming row* N*.*

Proof. Begin with the regulated-integral handoff theorem. It places every endpoint-facing read behind the ordered rows F, A, T, S, I, so a consumer cannot reach E directly from a tagged partition or from an ambient real object.

Next use the Darboux mesh-refinement and bounded-variation ledger rows. Mesh refinement confines the partition read to T and S, while the ledger theorem forces every jump, variation read, and endpoint contribution used by the sum to be a visible row inside A before it is transported downstream.

The exactness rows keep the calculation local. Step-integrator exactness computes each dyadic step contribution from S, T, A, and regulated-integrator variation exactness determines the variation contribution through A, T, S, H, C, P, N rather than through a measure-space integral or selected limiting partition.

Finally apply tagged-partition induction over the finite BHist refinement ledger. The base row and each subdivision step preserve the same RiemannSum$^{\uparrow}$, DyadicStepFunction$^{\uparrow}$, BoundedVariation$^{\uparrow}$, RegulatedIntegral$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert route, so every integral-facing consumer reads the tagged-partition result only through the stated dependency ledger. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier scope ledger`

Theorem 1637.17 (Riemann–Stieltjes public witness export)**.** *For every accepted* RiemannStieltjes$^{\uparrow}$ *carrier $R = (F, A, T, S, I, E, H, C, P, N)$, the regulated integrand row F, bounded-variation integrator row A, tagged partition and* RiemannSum$^{\uparrow}$ *row T, dyadic step approximation row S, regulated-integral handoff row I, and* Real$^{\uparrow}$ *endpoint seal E form the public finite witness packet consumed by downstream integration chapters. The exported packet is read only through H, C, P, N and contains no Lebesgue integral, measure completion, selected limiting partition, or host quotient of integrands or integrators.*

Proof. Start with the scoped ledger of Definition 1637.16. It already forces every integral-facing consumer to pass through the displayed dependency rows before the endpoint seal is visible.

The witness packet is the front face of that ledger. Rows F, A, T, S, I, E expose the regulated integrand, bounded-variation integrator, tagged Riemann sum, dyadic step approximation, handoff, and endpoint seal; rows H, C, P, N only transport, replay, source, and name the same finite data.

Because the carrier has no coordinate for measure completion, selected limiting partitions, or quotient equality of integrands and integrators, exporting these rows as public data does not add a stronger analytic claim. It only makes the finite Riemann–Stieltjes witness available to downstream integration chapters. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier public witness export`

Theorem 1637.18 (Riemann–Stieltjes regulated-integral public bridge)**.** *Any* RegulatedIntegral$^{\uparrow}$ *or* RiemannSum$^{\uparrow}$ *consumer of the public* RiemannStieltjes$^{\uparrow}$ *witness packet must pass through the rows*

$$F, A, T, S, I, E, H, C, P, N$$

before exporting an integration-facing read. The bridge exposes the regulated integrand, bounded-variation integrator, tagged partition, dyadic step approximation, regulated-integral handoff, Real endpoint seal, transport, replay, provenance, and local NameCert rows; it cannot substitute measure completion, selected limiting partitions, quotient integrand equality, quotient integrator equality, or a host limiting integral.

Proof. Start with the NameCert obligations of Definition 1637.2. They identify the accepted carrier as the finite row family $F, A, T, S, I, E, H, C, P, N$ and make the regulated integrand, bounded-variation integrator, tagged sum, dyadic step approximation, handoff, endpoint seal, transport, replay, provenance, and naming rows visible.

Apply the regulated-integral handoff of Definition 1637.3. It places each endpoint-facing read behind F, A, T, S, I, so a $\mathsf{RegulatedIntegral}^{\uparrow}$ consumer cannot bypass the displayed handoff row.

Use the Darboux mesh-refinement route, Definition 1637.4, together with the public witness export of Definition 1637.17. The former confines $\mathsf{RiemannSum}^{\uparrow}$ reads to the visible tagged partition and dyadic step rows, while the latter makes exactly F, A, T, S, I, E available as public witness data transported by H, C, P, N.

Since no carrier coordinate stores measure completion, a selected limiting partition, quotient equality of integrands or integrators, or a host limiting integral, the bridge remains a public finite-packet handoff and not a stronger integration theorem. □

Lean4 ✓ `BEDC.Derived.RiemannStieltjesUp.RiemannStieltjesCarrier regulated integral public bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannStieltjes}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannStieltjes}^{\uparrow}$ packet is generated from a regulated integrand row, bounded-variation integrator row, tagged partition and Riemann-sum row, dyadic step approximation row, regulated-integral handoff row, Real endpoint seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannStieltjes}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1637.1, Definition 1637.2, Definition 1637.3, Definition 1637.10, the bounded-variation jump boundary, Definition 1637.15, Definition 1637.17, and Definition 1637.18 bind the carrier, NameCert obligations, regulated-integral handoff, Darboux mesh refinement, bounded-variation control, refinement-window exhaustion, public finite witness export, and bridge-facing public handoff as the Riemann–Stieltjes route over $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannStieltjes}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lebesgue integration, measure completion, selected limits, host real equality, any quotient of integrands or integrators, Lean verification, or any standard bridge.

Upgrade path: No paper-side theory upgrade remains; bridgeCheckedV is the remaining formal-axis target for the public regulated-integral handoff.

Chapter 1638

A Concrete Naming Certificate for $\mathsf{RealMetric}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealMetric}^{\uparrow}$ packet names the standard metric on displayed regular real names by routing distance through finite absolute-value, stream-window, regular-rational, and dyadic-tolerance rows. It sits between the real-name seal of chapter 38, the absolute-value surface of chapter 1608, the generic metric obligations of chapter 57, the stream and regular-Cauchy readback interfaces of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. When a distance consumer asks for completion-facing tail data, the packet consumes the regular-Cauchy completion route of chapter 1639 while retaining the local $\mathsf{RealAbs}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ rows. The chapter records the finite NameCert boundary for real distance; it does not identify quotient reals, select limits, invoke trichotomy, or assert completeness.

Definition 1638.1 (Real metric carrier)**.** *A* $\mathsf{RealMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathsf{Real}} = (X, Y, A, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two displayed* $\mathsf{Real}^{\uparrow}$ *seals whose distance is being named.*
2. *A is the* $\mathsf{RealAbs}^{\uparrow}$ *distance row for the displayed difference read from X and Y.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to present finite distance requests and triangle budgets.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened from the two real-name seals before the distance row is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the displayed stream windows and dyadic tolerances.*
6. *H records componentwise* hsame *transport for the two real seals, absolute-value distance row, dyadic tolerance row, stream windows, regular readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a distance request through X, Y, then through S, R, D, into the* $\mathsf{RealAbs}^{\uparrow}$ *row A, and finally into the generic* $\mathsf{Met}^{\uparrow}$ *laws consumed downstream.*
8. *P is the* Pkg *provenance over* $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RealMetric}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, Y, A, D, S, R, H, C, P, N$. The carrier has no coordinate for host real equality, quotient real representatives, selected limits, trichotomy, or completeness of $\mathsf{Real}^\uparrow$.

Theorem 1638.2 (Real metric NameCert obligations)**.** *For every accepted* $\mathsf{RealMetric}^\uparrow$ *carrier* $M_{\mathsf{Real}} = (X, Y, A, D, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealMetric}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed endpoint seals, absolute-value distance row, dyadic tolerance row, stream windows, regular readback, transport, replay, provenance, and local naming row;*
2. *distance-row exposure, requiring every public distance read to pass from the endpoint seals X, Y through the finite windows S, the regular readback R, and the dyadic tolerance row D before it reaches the* $\mathsf{RealAbs}^\uparrow$ *distance row A;*
3. *symmetry route, requiring the classifier for the reversed endpoint order to reuse the same* $\mathsf{RealAbs}^\uparrow$ *row after componentwise transport of the two endpoint seals and their readback rows;*
4. *zero-distance classifier handoff, requiring a zero-distance request to be handed to the displayed* $\mathsf{Real}^\uparrow$ *seal classifier through the same S, R, D, A route rather than through host equality;*
5. *triangle-budget handoff, requiring any triangle use to present the two endpoint distance rows and the composed dyadic tolerance budget before reading the corresponding* $\mathsf{Met}^\uparrow$ *triangle law;*
6. *refusal of host real equality, quotient reals, selected limits, trichotomy, and completeness as NameCert data.*

No NameCert obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.RealMetricUp.RealMetricCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1638.1. The only available distance data are the endpoint seals X, Y, the absolute-value distance row A, the dyadic tolerance row D, the stream windows S, the regular readback R, and the structural rows H, C, P, N.

Distance-row exposure is forced by the displayed route. A public distance request first opens finite windows on the two real names, reads the corresponding regular rational data, records the dyadic tolerance request, and only then reaches the $\mathsf{RealAbs}^\uparrow$ row naming the absolute value of the displayed difference.

Symmetry and zero-distance reads use the same classifier surface. Reversing the endpoints transports X and Y componentwise through H and replays the same windows and readback through C; a zero-distance request then returns to the displayed $\mathsf{Real}^\uparrow$ seal classifier through S, R, D, A rather than by comparing host reals.

Triangle use is downstream of the same finite route. The generic $\mathsf{Met}^\uparrow$ triangle law is readable only after the three endpoint pairs have exposed their $\mathsf{RealAbs}^\uparrow$ distance rows and after the dyadic tolerance budgets have been composed in D.

Finally, provenance is exhausted by P and N. Since host real equality, quotient representatives, selected limits, trichotomy, and completeness are not coordinates of M_{Real}, the local NameCert surface cannot export them. □

Theorem 1638.3 (Real metric absolute-value distance handoff)**.** *Every accepted* $\mathsf{RealMetric}^\uparrow$ *distance consumer factors through the* $\mathsf{RealAbs}^\uparrow$ *row for the displayed endpoint difference and through the* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*, and* $\mathsf{DyadicRatCore}^\uparrow$ *readback route before it may read any* $\mathsf{Met}^\uparrow$

law. In particular, symmetry, zero-distance, and triangle consumers receive only the displayed rows

$\mathsf{Real}^{\uparrow}, \mathsf{RealAbs}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$

The packet exports no host real equality, quotient real representative, selected limit, trichotomy, or completeness coordinate.

Lean4 ✓ `BEDC.Derived.RealMetricUp.RealMetricCarrier absolute value distance handoff`

Proof. Let a distance consumer read an accepted packet $M_{\mathsf{Real}} = (X, Y, A, D, S, R, H, C, P, N)$. Carrier admission gives the consumer access only to the displayed endpoint, distance, tolerance, window, readback, transport, replay, provenance, and local naming rows.

The distance value is therefore obtained through the absolute-value route. The endpoint seals X, Y provide the displayed difference source, S opens the finite windows, R supplies regular rational readback, D records the dyadic tolerance request, and A names the resulting $\mathsf{RealAbs}^{\uparrow}$ distance row.

Only after that route is displayed may the consumer invoke a generic $\mathsf{Met}^{\uparrow}$ law. Symmetry reuses A under endpoint transport, zero-distance returns through the displayed $\mathsf{Real}^{\uparrow}$ seal classifier, and triangle reads the composed dyadic budget before applying the metric triangle row.

The remaining coordinates are structural support. The BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows transport, replay, source, and name the same route, with no row from which host real equality, quotient representatives, selected limits, trichotomy, or completeness can be read. □

Theorem 1638.4 (Real metric located-distance handoff)**.** *Every located-distance or completion-facing consumer of an accepted* $\mathsf{RealMetric}^{\uparrow}$ *packet must pass through the endpoint seals* X, Y*, the stream windows* S*, the regular readback* R*, the dyadic tolerance row* D*, and the* $\mathsf{RealAbs}^{\uparrow}$ *row* A *before it reads any* $\mathsf{Met}^{\uparrow}$ *law. The located-distance route is therefore*

$$X, Y \longmapsto S, R, D \longmapsto A \longmapsto \mathsf{Met}^{\uparrow}$$

inside the structural rows H, C, P, N*. It exports no host real equality, quotient representative, selected limit, trichotomy, or completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RealMetricUp.RealMetricLocatedDistanceHandoff`

Proof. Project the accepted carrier $M_{\mathsf{Real}} = (X, Y, A, D, S, R, H, C, P, N)$ of Definition 1638.1. A located-distance request first receives only the displayed endpoint seals X, Y, so every downstream read is still tied to the two $\mathsf{Real}^{\uparrow}$ names carried by the packet.

The request then follows the same finite readback route as Definition 1638.3. The stream windows S expose the finite observations, the regular readback row R presents the $\mathsf{RegSeqRat}^{\uparrow}$ data over those windows, and the dyadic tolerance row D records the requested distance precision before the $\mathsf{RealAbs}^{\uparrow}$ distance row A is visible.

Only after A is exposed may a completion-facing consumer read a $\mathsf{Met}^{\uparrow}$ law. The rows H, C, P, N preserve this route under componentwise transport, replay, provenance, and local naming, while no carrier row names host equality, quotient representatives, selected limits, trichotomy, or completeness. □

Theorem 1638.5 (Real metric triangle-window stability)**.** *Every triangle request for accepted* $\mathsf{RealMetric}^{\uparrow}$ *packets factors through shared finite* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealAbs}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *rows before a triangle-inequality read is visible. For endpoint seals* X, Y, Z*, the three*

pairwise distance reads must first expose their stream windows, regular rational readbacks, dyadic tolerance budgets, and absolute-value distance rows inside the same BHist, hsame, Cont, Pkg, *and* NameCert *route.*

Lean4 ✓ `BEDC.Derived.RealMetricUp.RealMetricCarrier triangle window stability`

Proof. Inspect the carrier of Definition 1638.1. A distance row is never a raw host value; it is read from endpoint $\mathsf{Real}^{\uparrow}$ seals through S, R, D, A before it reaches the generic $\mathsf{Met}^{\uparrow}$ row.

A triangle request therefore opens three copies of the same finite window route, one for each endpoint pair. The stream windows are shared at the level of displayed $\mathsf{StreamName}^{\uparrow}$ observations, the regular readbacks land in $\mathsf{RegSeqRat}^{\uparrow}$, and the dyadic budgets are composed in $\mathsf{DyadicRatCore}^{\uparrow}$ before the absolute-value rows are compared.

Only after these finite rows are present can the consumer read a $\mathsf{Met}^{\uparrow}$ triangle law. The structural rows H, C, P, N keep the three pairwise routes attached to the accepted packets, so no selected limit, quotient representative, trichotomy, or ambient completeness principle enters the triangle read. □

Theorem 1638.6 (Real metric identity-of-indiscernibles window)**.** *Every zero-distance read from an accepted* $\mathsf{RealMetric}^{\uparrow}$ *packet hands off to the displayed* $\mathsf{Real}^{\uparrow}$ *classifier through the same finite windows used for distance:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the* $\mathsf{RealAbs}^{\uparrow}$ *distance row, and the* $\mathsf{Met}^{\uparrow}$ *zero-distance law. The handoff identifies only the displayed real-name classifier data and does not use host equality, quotient representatives, selected limits, trichotomy, or completeness.*

Lean4 ✓ `BEDC.Derived.RealMetricUp.RealMetricCarrier identity of indiscernibles window`

Proof. Start with a zero-distance request for $M_{\mathsf{Real}} = (X, Y, A, D, S, R, H, C, P, N)$. The request first sees the endpoint seals X, Y, not host real representatives.

The distance route then exposes the finite windows S, readback R, tolerance row D, and absolute-value row A. These are the same rows used by located-distance and triangle consumers, so the zero-distance read has no separate equality channel.

The final handoff is to the displayed $\mathsf{Real}^{\uparrow}$ classifier, authenticated by H, C, P, N and then read through the generic $\mathsf{Met}^{\uparrow}$ zero-distance row. Since the carrier lacks host equality, quotient representatives, selected limits, trichotomy, and completeness coordinates, identity of indiscernibles is routed only through that finite window. □

Theorem 1638.7 (Real metric TasteGate scope)**.** *The accepted* $\mathsf{RealMetric}^{\uparrow}$ *carrier rows form a single-carrier* ChapterTasteGate *surface over endpoint* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{RealAbs}^{\uparrow}$ *absolute-value distance row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealMetric}^{\uparrow}}$ *row. The gate compares packets only through this displayed family and does not add host real equality, quotient representatives, selected limits, trichotomy, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RealMetricUp.taste gate`

Proof. Project the carrier of Definition 1638.1. Its displayed rows are exactly the endpoint seals X, Y, absolute-value distance row A, tolerance row D, windows S, readback R, and structural rows H, C, P, N.

Apply the NameCert obligations of Definition 1638.2. They force distance exposure, symmetry, zero-distance handoff, and triangle-budget handoff to read the same finite route before any $\mathsf{Met}^{\uparrow}$ law is visible.

Use the identity-of-indiscernibles window above. The zero-distance classifier also returns through the displayed $\mathsf{Real}^{\uparrow}$ seal and the same window/readback/tolerance/distance surface. Hence the TasteGate is a single-carrier surface over those rows, with no extra coordinate for the excluded host or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1638.1, Definition 1638.2, and Definition 1638.3 seed the finite real-metric surface over displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealMetric}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealMetricUp.taste\_gate`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient real representatives, selected limits, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires formal carrier admission, distance-row exposure, symmetry, zero-distance classifier handoff, triangle-budget handoff, and non-escape proofs for the same finite packet.

Constructive story: A $\mathsf{RealMetric}^{\uparrow}$ packet is generated from two displayed RealUp seals, a RealAbs distance row for their displayed difference, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1639

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive regular Cauchy completion, the $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ packet names the finite route by which dyadic tolerances, stream observations, regular rational sequences, and the terminal Real seal are assembled into a completion-facing object. It sits on the L10 spine supplied by the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational sequence route of chapter 301, and the displayed real-name seal of chapter 38. The packet records a regular Cauchy completion boundary for Real and Completion consumers; it does not assert completeness of $\mathsf{Real}^{\uparrow}$, identify quotient streams, select limits, invoke countable choice, or compare host real values.

Definition 1639.1 (Bishop regular Cauchy completion carrier)**.** *A* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, containing the dyadic source endpoints and tolerance comparisons used by the regular Cauchy reading.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation family opened from the named stream before any completion-facing consumer reads a tail claim.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the regular rational sequence data exposed by the stream windows and dyadic endpoint ledger.*
4. *D is the dyadic tail modulus row, assigning each displayed dyadic tolerance a finite tail index and the dyadic comparison budget used at that tail.*
5. *W is the common-tail window row, recording the shared finite window in which two accepted regular readings may be compared without quotienting their streams.*
6. *H records componentwise* hsame *transport for endpoints, observations, regularity, tail modulus, common-tail windows, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion tolerance request through D, then through W, S, R, E, and finally into the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*

8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, S, R, D, W, H, C, P, N$. The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionTasteGate single carrier alignment`

Theorem 1639.2 (Bishop regular Cauchy completion regularity envelope)**.** *For every accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier $B = (E, S, R, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyComplet}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint row, finite stream observations, regular rational sequence row, dyadic tail modulus, common-tail window, transport, replay, provenance, and local naming row;*

2. *classifier transport, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the ordered route D, W, S, R, E;*

3. *regularity envelope, requiring every regular Cauchy read to pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the dyadic endpoint ledger E before reaching the* $\mathsf{Real}^{\uparrow}$ *seal;*

4. *common-tail modulus stability, requiring any comparison of two accepted readings to expose the same displayed tail budget in D and the same common-tail window in W;*

5. *Real-seal handoff, requiring any Real consumer to read the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after S, R, E, D, W have been displayed;*

6. *ledger non-escape, requiring P and N to export no quotient stream equality, host real equality, selected limit, countable choice, or ambient Real completeness.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionRegularityEnvelope`

Proof. Carrier admission is projection from Definition 1639.1. The visible packet consists of the endpoint ledger E, finite observations S, regularity row R, dyadic tail modulus D, common-tail window W, and the structural rows H, C, P, N.

Classifier transport is structural. Componentwise hsame in H preserves each displayed coordinate, and displayed Cont replay in C repeats the same ordered route D, W, S, R, E for every accepted tolerance request.

The regularity envelope is forced by the placement of R and E. A consumer cannot read the completion boundary until the finite stream windows have exposed regular rational data and the dyadic ledger has supplied the corresponding endpoint and tolerance comparisons.

Common-tail stability is the row W governed by the tail modulus D. Two accepted readings are compared only inside the displayed shared window and under the same dyadic budget, so the packet never identifies their underlying streams by quotient equality.

Finally, the Real-seal handoff and ledger boundary are exhausted by C, P, N. Since no excluded quotient, host-equality, selected-limit, choice, or Real-completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1639.3 (Bishop regular Cauchy completion seal stability)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier reads only the displayed regular Cauchy completion rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Consequently the consumer receives finite endpoint, observation, regularity, tail-modulus, common-tail, transport, replay, provenance, and naming data, and receives no quotient stream equality, host real equality, selected limit, countable-choice schedule, or ambient Real completeness theorem.

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier seal stability`

Proof. Let a $\mathsf{Real}^{\uparrow}$ consumer read an accepted packet $B = (E, S, R, D, W, H, C, P, N)$. Carrier projection gives access only to the displayed rows of Definition 1639.1.

The Real-facing route starts with the tolerance request. The dyadic tail modulus D fixes the tail budget, W identifies the common window, S supplies finite observations, R records regular rational readback, and E supplies the dyadic endpoint comparisons before the terminal $\mathsf{Real}^{\uparrow}$ seal is reached.

The remaining rows transport and replay the same route. The BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ coordinates preserve the finite packet boundary and add no quotient, host-equality, selected-limit, choice, or completeness object. □

Theorem 1639.4 (Bishop regular Cauchy completion classifier transport)**.** *Let* $B = (E, S, R, D, W, H, C, P, N)$ *and* $B' = (E', S', R', D', W', H', C', P', N')$ *be accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carriers. If* H *records componentwise* hsame *transport from each row of* B *to the corresponding row of* B'*, and if* C *displays* Cont *replay of the route* D, W, S, R, E *into the terminal* $\mathsf{Real}^{\uparrow}$ *seal, then the local classifier transports acceptance from* B *to* B' *only through the displayed rows*

$$E, S, R, D, W, H, C, P, N.$$

In particular, the transport step may replace endpoint, observation, regularity, tail-modulus, common-window, replay, provenance, and local naming rows by their displayed hsame *companions, but it cannot introduce quotient stream equality, host real equality, a selected limit, countable choice, or ambient Real completeness.*

Proof. The classifier for Definition 1639.1 is row-local. Its admissible inputs are precisely the coordinates $E, S, R, D, W, H, C, P, N$, so a transported packet must be read by projecting these coordinates and comparing them with their displayed companions.

The componentwise hsame row H supplies the only equality-like transport available to the classifier. It moves endpoints, finite observations, regularity witnesses, dyadic tail budgets, common-tail windows, replay rows, provenance, and local naming rows without changing the ordered completion route.

The Cont row C then replays the same tolerance request through D, W, S, R, E before the $\mathsf{Real}^{\uparrow}$ seal is exposed. Since no excluded coordinate is present in either carrier, classifier transport has no path to quotient equality, host equality, selected limits, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier classifier transport`

Theorem 1639.5 (Bishop regular Cauchy completion cofinal-tail refinement)**.** *If an accepted* BishopRegularCauchyCompletion$^{\uparrow}$ *packet receives a* CauchyFilter$^{\uparrow}$ *row whose displayed base refines the packet's common-tail window, then every* Real$^{\uparrow}$*-facing completion read can be replayed through a* RegSeqRat$^{\uparrow}$ *tail window with the same* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* StreamName$^{\uparrow}$ *finite observations,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The replay exports no quotient filters, selected limits, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet from Definition 1639.1. Its comparison route exposes the dyadic tail modulus D, common-tail window W, StreamName observations S, RegSeqRat readback R, endpoint ledger E, and the structural rows H, C, P, N.

Read the incoming CauchyFilter$^{\uparrow}$ row only through its displayed cofinal-base refinement. By hypothesis that base refines the common-tail window W, so the filter request selects the finite window already named by the Bishop packet rather than introducing a quotient of filters.

Inside that window, replay the existing carrier route. The finite observations are read through S, the regular tail data through R, and the tolerance budget through D before the terminal Real$^{\uparrow}$ seal is available.

The rows H and C transport and replay the same route componentwise, while P and N keep the provenance and local naming attached to the displayed carrier. Since the packet contains no coordinate for quotient filters, selected limits, countable choice, host real equality, or ambient Real completeness, the cofinal-tail refinement adds no such data to the completion read. □

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCofinalTailRefinement`

Theorem 1639.6 (Bishop regular Cauchy completion common-tail modulus stability)**.** *For every accepted* BishopRegularCauchyCompletion$^{\uparrow}$ *carrier* $B = (E, S, R, D, W, H, C, P, N)$*, any comparison of two completion-facing regular readings must first expose the same displayed dyadic tail budget in* D *and the same displayed common-tail window in* W*. The comparison may then read finite observations from* S*, regularity data from* R*, and endpoint comparisons from* E*, but the terminal* Real$^{\uparrow}$ *seal is reached only after the shared* D *and* W *rows have been fixed.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier common tail modulus stability`

Proof. A completion-facing comparison begins with a dyadic tolerance request. The carrier has only one displayed tail-modulus row D, so both readings must use the same finite tail index and dyadic comparison budget attached to that request.

Once D is fixed, the common-tail row W supplies the shared finite window in which the two readings may be compared. The stream observations S, regularity data R, and endpoint ledger E are read inside that window, not by identifying the underlying streams.

The Real seal is therefore downstream of the displayed common-tail route. The packet cannot bypass D or W through quotient stream equality, selected limits, countable choice, or an ambient completeness theorem for Real$^{\uparrow}$. □

Theorem 1639.7 (Bishop regular Cauchy completion tail-diameter control)**.** *For every accepted* BishopRegularCauchyCompletion$^{\uparrow}$ *carrier* $B = (E, S, R, D, W, H, C, P, N)$*, a displayed dyadic tail budget* D *together with the common-tail window* W *determines a finite* CauchyTailDiameter$^{\uparrow}$ *row controlling any two* RegSeqRat$^{\uparrow}$ *readbacks visible to the completion comparison before the terminal* Real$^{\uparrow}$ *seal is inspected. The control row is obtained from the displayed* $D, W, S, R, E, H, C, P, N$

route and exports no quotient stream equality, selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionTailDiameterControl`

Proof. Begin with common-tail modulus stability, Definition 1639.6. It forces every comparison of two completion-facing regular reads to expose the same dyadic budget D and the same common-tail window W before the Real seal can be reached.

Next use diagonal-window stability, Definition 1639.8. Inside the fixed D, W window, both $\mathsf{RegSeqRat}^{\uparrow}$ readbacks are read through the finite StreamName observations and endpoint ledger rows, so their diameter claim is a row of the same displayed packet rather than an equality of underlying streams.

The row named by $\mathsf{CauchyTailDiameter}^{\uparrow}$ is therefore the finite tail-diameter ledger extracted from D, W, S, R, E and transported through H with replay through C. Provenance and local naming remain in P, N.

Finally apply nonescape exactness, Definition 1639.15. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness, the tail-diameter control closes before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected. □

Theorem 1639.8 (Bishop regular Cauchy completion diagonal-window stability)**.** *Let* $B = (E, S, R, D, W, H, C, P,$ *and* $B' = (E', S', R', D, W, H', C', P', N')$ *be accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packets sharing the same displayed dyadic tail modulus* D *and common-tail window* W*. Any diagonal comparison between the two packets must first read* D*, then* W*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows* S, S' *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* R, R'*, before either terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *or* E' *is available.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier diagonal window stability`

Proof. Project the carrier definition Definition 1639.1. The only comparison rows available to the diagonal read are the displayed tail modulus, common window, stream observations, regular readbacks, terminal seals, transport, replay, provenance, and naming rows.

Apply common-tail modulus stability from Definition 1639.6. It forces the comparison to expose the shared dyadic budget D and the shared finite window W before any Real seal can be inspected.

Use classifier transport from Definition 1639.4. Componentwise hsame and displayed Cont replay move only the visible S, S', R, R' rows along the already fixed D, W route.

Finally apply seal stability from Definition 1639.3. The Real seals are downstream of the displayed regular Cauchy route, so the diagonal comparison cannot bypass the ordered read through quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness. □

Theorem 1639.9 (Bishop regular Cauchy completion window induction)**.** *Every accepted completion-facing comparison over a* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier*

$$B = (E, S, R, D, W, H, C, P, N)$$

is obtained by induction on the displayed finite window row W*. The base case reads the first finite observation through* S, R, E *under the dyadic tail budget exposed by* D*. The step extends the same dyadic budget through* D, W *before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible, and then replays the enlarged window through* S, R, E, H, C, P, N*. The induction exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionWindowInduction

Proof. Start with the carrier definition Definition 1639.1. The only finite growth coordinate available to a completion-facing comparison is the common-tail window row W, and every read of that row is tied to the dyadic tail modulus D.

The base case is the first displayed observation. By common-tail modulus stability, Definition 1639.6, the comparison must expose the dyadic budget in D before it can read the finite observation row S, the regular readback row R, or the endpoint seal row E.

For the step, extend the same finite window rather than choosing a new stream representative. Diagonal-window stability, Definition 1639.8, keeps the comparison inside the already fixed D, W route as the next StreamName and RegSeqRat observations are displayed.

Finally apply seal stability and nonescape exactness, Definition 1639.3 and Definition 1639.15. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of the finite window recursion, while quotient stream equality, selected limits, countable choice, host equality, and ambient Real completeness have no carrier coordinate from which they could be read. □

Theorem 1639.10 (Bishop regular Cauchy completion finite-window induction)**.** *Every accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet consumes its finite observation windows by induction over the displayed* $\mathsf{StreamName}^{\uparrow}$ *sections. At each step the read preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The induction exports no selected limit, quotient representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionFiniteWindowInduction

Proof. Project the carrier $B = (E, S, R, D, W, H, C, P, N)$ from Definition 1639.1. The finite observation ledger is the $\mathsf{StreamName}^{\uparrow}$ row S, with regular readback in R, dyadic tolerance control in E, D, common-tail scope in W, and terminal sealing in E through the Real-facing route.

Run induction over the displayed window ledger. The base window reads the first finite StreamName section, then exposes the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ row and dyadic tolerance rows before any $\mathsf{Real}^{\uparrow}$ seal can be inspected.

For the step, extend only the displayed StreamName section. Common-tail modulus stability and diagonal-window stability keep the enlarged window inside the same D, W, S, R, E route, so the regular readback and dyadic tolerance rows are preserved rather than replaced by a chosen representative.

Finally replay the enlarged route through H, C, P, N and then through the Real seal discipline. Since the carrier has no coordinate for a selected limit, quotient representative, countable-choice schedule, or ambient Real completeness, the induction cannot export any of those objects. □

Theorem 1639.11 (Bishop regular Cauchy completion finite-stage induction)**.** *Every accepted finite-stage completion read of a* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet factors through the displayed sequence of rows*

$$D, W, S, R, E, H, C, P, N.$$

The stage induction first fixes the dyadic tail budget D and common-tail window W, then reads the finite $\mathsf{StreamName}^{\uparrow}$ *observation row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N. No induction step may replace the finite-stage route by quotient stream equality, a selected limit, a countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionFiniteStageInduction`

Proof. Induct on the finite stage named by the window row W. The base stage is the first displayed StreamName window, so Definition 1639.10 already forces the read through D, W, S, R, E before any terminal seal is visible.

For the successor stage, extend the displayed window while keeping the same dyadic budget. Common-tail modulus stability from Definition 1639.6 keeps the enlarged stage inside the fixed D, W route.

The regular readback and seal rows then follow the existing finite-window induction. Componentwise transport, replay, provenance, and naming are carried by H, C, P, N rather than by an external representative of the Cauchy stream.

Finally use diagonal-window stability from Definition 1639.8. It blocks any comparison that skips the displayed stage rows, so the induction cannot export quotient stream equality, a selected limit, countable choice, host real equality, or ambient Real completeness. □

Theorem 1639.12 (Bishop regular Cauchy completion universal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet factors through the single displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the packet's BHist*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The handoff is obtained by combining the common-tail, finite-window, diagonal-window, tail-diameter, seal-stability, and nonescape rows already displayed in this chapter; it exports no selected limit, quotient stream representative, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionUniversalHandoff`

Proof. Start with finite-window induction from Definition 1639.10. It forces the consumer to read $\mathsf{StreamName}^{\uparrow}$ windows by finite induction while preserving the $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, transport, replay, provenance, and terminal Real seal.

Next apply common-tail modulus stability and diagonal-window stability from Definition 1639.6 and Definition 1639.8. These rows fix the shared dyadic budget and common window before the stream, regularity, and Real-seal rows can be consumed.

Use tail-diameter control to keep the two regular readbacks inside the same finite completion comparison. Then apply seal stability and the abstract bridge route, Definition 1639.3 and Definition 1639.23, to send the same displayed packet to either a $\mathsf{Real}^{\uparrow}$ consumer or a $\mathsf{Completion}^{\uparrow}$-facing consumer.

Finally apply nonescape exactness from Definition 1639.15. Since the carrier contains only the displayed BHist, hsame, Cont, Pkg, and NameCert rows over the dyadic-stream-regular-Real route, no excluded host-level or choice-level object can be exported by the universal handoff. □

Theorem 1639.13 (Bishop regular Cauchy completion tail equivalence is quotient-free)**.** *Let* $B = (E, S, R, D, W, H, C, P, N)$ *and* $B' = (E', S', R', D, W, H', C', P', N')$ *be accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packets sharing the same dyadic tail modulus* D *and the same common-tail window* W*. Then their completion-facing comparison is carried only by the displayed rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

In particular, the comparison reads the common dyadic budget, the shared StreamName window, the two RegSeqRat readbacks, the Real seal boundaries, componentwise transport, replay, provenance,

and local naming data; it exports no quotient-stream equality, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier tail equivalence quotient free`

Proof. Begin with common-tail modulus stability from Definition 1639.6. The shared rows D and W must be exposed before either packet can reach its Real seal boundary.

Transport the classifier by Definition 1639.4. Componentwise hsame and displayed Cont replay compare only the listed endpoint, observation, regularity, tail-modulus, common-window, provenance, and naming rows.

Use seal stability from Definition 1639.3. The terminal $\mathsf{Real}^{\uparrow}$ boundaries are downstream of the displayed regular Cauchy route, so the comparison cannot be replaced by a selected limit or host real equality.

Finally apply nonescape exactness from Definition 1639.15. Since no quotient-stream equality, selected-limit, choice, host-equality, or Real-completeness coordinate is present in either carrier, the shared-tail comparison is quotient-free. □

Theorem 1639.14 (Bishop regular Cauchy completion tail determinacy)**.** *Let* $B = (E, S, R, D, W, H, C, P, N)$ *and* $B' = (E', S', R', D, W, H', C, P', N')$ *be accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packets sharing the same displayed dyadic tail modulus* D*, the same common-tail window* W*, componentwise transport through* H, H'*, and the same displayed* Cont *replay* C*. Then every completion-facing consumer reads the two packets through the same displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

before reaching the terminal Real-seal boundary. The determinacy is a rowwise read-route claim: it exports no quotient stream equality, no selected limit, no host real equality, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionTailDeterminacy`

Proof. Begin with common-tail modulus stability from Definition 1639.6. The shared rows D and W force both completion-facing reads to expose the same dyadic tail budget and the same finite comparison window before either Real seal can be inspected.

Next use classifier transport from Definition 1639.4. Componentwise hsame transport moves only the displayed endpoint, observation, regularity, tail-modulus, common-window, provenance, and naming rows, while the shared Cont replay C preserves the same ordered route.

Apply seal stability from Definition 1639.3. The terminal Real-facing boundary is downstream of the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint ledger, $\mathsf{StreamName}^{\uparrow}$ finite observations, and $\mathsf{RegSeqRat}^{\uparrow}$ readback, so the two reads reach their seals through the same visible dependency route.

Finally apply nonescape exactness from Definition 1639.15. Since neither carrier has a coordinate for quotient stream equality, selected limits, host real equality, countable choice, or ambient Real completeness, no such object can distinguish the completion-facing read route. □

Theorem 1639.15 (Bishop regular Cauchy completion nonescape exactness)**.** *Every completion-facing read of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier* $B = (E, S, R, D, W, H, C, P, N)$ *is exhausted by the rows* $E, S, R, D, W, H, C, P, N$*. Thus the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}$ *boundary exports only dyadic endpoints, finite stream observations, regular rational sequence data, dyadic tail moduli, common-tail windows, componentwise* hsame *transport, displayed* Cont *replay,*

Pkg *provenance, and local naming data, and exports no quotient stream equality, host real equality, selected limit, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionNonescapeExactness`

Proof. Carrier projection is exhaustive: the tuple $B = (E, S, R, D, W, H, C, P, N)$ lists every coordinate admitted by the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}$ surface. A completion-facing consumer can therefore read only those displayed rows.

The structural rows do not enlarge the carrier. The hsame row transports already displayed coordinates, the Cont row replays the displayed route, P records provenance for the named packet, and N records the local NameCert row.

The excluded objects are absent from the tuple and absent from the replay route. Since there is no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient Real completeness, no completion-facing read can escape the displayed carrier boundary. □

Theorem 1639.16 (Bishop regular Cauchy completion Real seal consumer)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier* $B = (E, S, R, D, W, H, C, P, N)$ *must pass through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto W \longmapsto H, C, P, N \longmapsto E.$$

In particular, the consumer first reads StreamName windows, then the RegSeqRat readback, dyadic tolerance budget, common-tail modulus, classifier transport, and seal stability route before the terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed. The read exports no quotient stream equality, selected limit, countable choice, host real equality, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier real seal consumer`

Proof. Begin with the regularity envelope. By Definition 1639.2, a completion-facing read must pass through the displayed finite stream observations S and regular rational sequence data R before it can be treated as a completion route.

Next fix the tolerance and tail data. The common-tail modulus stability theorem Definition 1639.6 forces the dyadic budget D and common-tail window W to be read before the Real seal is inspected.

Transport is then confined by Definition 1639.4. The componentwise hsame row and displayed Cont replay preserve the same route through S, R, D, W and do not introduce an outside real-equality coordinate.

Finally apply seal stability and nonescape exactness. The theorem Definition 1639.3 places the terminal seal E downstream of the displayed route, while Definition 1639.15 removes quotient equality, selected limits, countable choice, host equality, and ambient Real completeness from the accepted consumer boundary. □

Theorem 1639.17 (Bishop regular Cauchy completion density handoff)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer, and every countable-density consumer, of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet must first read the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows supplied by that completion packet before it may consume any dense embedding or countable dense surface. The handoff is the finite route*

$$\mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow},$$

where the first arrow is the quotient-free Real-seal route of this chapter and the second arrow is the RealSeparability hub route through regular rational readback and completion-facing dense embeddings. It exports no host dense subset, selected approximant, quotient real equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionCarrier density handoff`

Proof. Begin with the Real-seal consumer theorem Definition 1639.16. It forces every Real-facing read of the Bishop regular Cauchy completion packet through the finite StreamName window, RegSeqRat readback, dyadic tolerance, common-tail, transport, replay, provenance, and local naming rows before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible.

Next apply the universal handoff of Definition 1639.12. The same displayed route serves either a $\mathsf{Real}^{\uparrow}$ consumer or a completion-facing consumer, so a density consumer cannot bypass the packet by reading a quotient representative, selected limit, host equality coordinate, or choice schedule.

Finally use the $\mathsf{RealSeparability}^{\uparrow}$ hub route of chapter 1809, in particular the RegSeqRat forward route Definition 1810.35. That route admits completion-facing density only after the regular rational readback and the dense-embedding boundary have been exposed. Composing the two routes gives the stated density handoff and leaves no coordinate for host dense subsets, selected approximants, quotient real equality, countable choice, or ambient Real completeness. □

Theorem 1639.18 (Bishop regular Cauchy completion obligation package)**.** *The carrier definition together with the regularity-envelope, seal-stability, classifier-transport, common-tail-modulus-stability, nonescape-exactness, and Real-seal-consumer theorems form the obligation-closed* BishopRegularCauchyCo *packet over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. Its visible obligation surface consists exactly of carrier admission, the regularity envelope, Real-seal handoff, classifier transport, shared tail budget, ledger non-escape, and terminal Real consumer routing.*

Consequently no quotient stream equality, selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row remains as an unlisted local NameCert obligation for this packet.*

Proof. Project the carrier tuple from Definition 1639.1. Its coordinates are precisely the dyadic endpoint row, finite stream observations, regular rational sequence row, dyadic tail modulus, common-tail window, componentwise transport, replay, provenance, and local naming row, so carrier admission is already displayed by the definition.

The regularity and Real-facing rows are supplied by Definition 1639.2 and Definition 1639.3. These rows force every completion-facing read through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the finite stream observations before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed.

Classifier and tail-budget transport are supplied by Definition 1639.4 and Definition 1639.6. The former restricts transport to the displayed hsame and Cont rows, while the latter fixes the shared dyadic tail budget and common-tail window before any comparison reaches the Real seal.

Finally, Definition 1639.15 and Definition 1639.16 close the ledger boundary. Since the carrier tuple contains no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}$ obligations and the terminal Real consumer surface. □

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionObligationClosurePack`

Theorem 1639.19 (Bishop regular Cauchy completion scoped kernel route)**.** *Every accepted carrier read, classifier transport, common-tail comparison, and Real-seal handoff in the* BishopRegularCauchyCompletion *packet factors through exactly the displayed rows*

$$E, S, R, D, W, H, C, P, N.$$

Here E is grounded in $\mathsf{DyadicRatCore}^{\uparrow}$*, S in* $\mathsf{StreamName}^{\uparrow}$*, R in* $\mathsf{RegSeqRat}^{\uparrow}$*, the terminal handoff in* $\mathsf{Real}^{\uparrow}$*, and the structural support in* BHist, hsame, Cont, Pkg, *and* NameCert*. No accepted route factors through quotient stream equality, host real equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionScopedKernelRoute`

Proof. Carrier reads are forced by Definition 1639.1. The tuple exposes only $E, S, R, D, W, H, C, P, N$, with the endpoint, observation, regularity, terminal seal, history, transport, replay, provenance, and local naming roles named there.

The regularity and Real-facing passages are supplied by Definition 1639.2 and Definition 1639.3. They route every completion-facing request through the regular sequence row, dyadic endpoint ledger, finite observation family, tail budget, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Classifier and comparison steps are supplied by Definition 1639.4 and Definition 1639.6. The classifier reads componentwise hsame transport and displayed Cont replay, while the comparison first fixes the shared dyadic tail budget and common-tail window before reaching S, R, E.

The remaining boundary is Definition 1639.15 together with Definition 1639.16, and the rows are packaged by Definition 1639.18. Since those results exhaust the local NameCert obligations and identify no additional coordinate, the displayed kernel rows are exactly the dependency surface for the accepted routes. □

Theorem 1639.20 (Bishop regular Cauchy completion public Real-seal export)**.** *For every accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet, the public export available to* $\mathsf{Real}^{\uparrow}$ *and completion-facing consumers is exactly the quotient-free Real-seal handoff through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The export consists of the displayed dyadic endpoint ledger, finite stream observations, regular rational readback, dyadic tail modulus, common-tail window, seal-stability route, componentwise classifier transport, replay, package provenance, and local NameCert rows. It does not assert $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient stream equality, countable choice, host real equality, or any detached equality of Real names.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionPublicRealSeal`

Proof. Begin with the obligation package of Definition 1639.18. It identifies the carrier admission, regularity envelope, classifier transport, common-tail modulus, Real-seal handoff, nonescape boundary, and terminal Real consumer rows as the complete local obligation surface.

Next apply the scoped kernel route of Definition 1639.19. That theorem restricts every accepted carrier read, classifier transport, common-tail comparison, and Real-seal handoff to the displayed rows $E, S, R, D, W, H, C, P, N$ over the named DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert dependencies.

Seal stability supplies the public endpoint discipline. By Definition 1639.3, the terminal Real consumer reads finite endpoint, observation, regularity, tail-modulus, common-tail, transport, replay, provenance, and naming data before the $\mathsf{Real}^{\uparrow}$ seal is exposed.

The excluded objects are absent from all three ingredients. Since none of the carrier, obligation package, scoped route, or seal-stability row contains Real completeness, selected limits, quotient stream equality, countable choice, host real equality, or detached equality of Real names, the public export is exactly the quotient-free Real-seal handoff stated above. □

Theorem 1639.21 (Bishop regular Cauchy completion filter-Real handoff)**.** *A* BishopRegularCauchyCompletion$^{\uparrow}$ *packet receiving a* CauchyFilter$^{\uparrow}$ *basis reaches the terminal* Real$^{\uparrow}$ *seal only through the displayed CauchyFilter finite base,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *regular readback,* DyadicRatCore$^{\uparrow}$ *endpoint ledger, common-tail modulus, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. This handoff exports no* Real$^{\uparrow}$ *completeness, quotient stream equality, selected limits, host equality, or countable choice.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionFilterRealHandoff`

Proof. Start from the CauchyFilter Real-RegSeq handoff non-escape theorem Definition 333.8. It confines the incoming basis to displayed filter-base rows, stream schedules, regular readbacks, dyadic neighbourhood rows, real-ball bounds, refinement rows, modulus rows, ledger evidence, and provenance.

Apply the regularity envelope of Definition 1639.2. The accepted Bishop completion packet reads the incoming basis only as finite StreamName windows, RegSeqRat regularity data, dyadic endpoint ledgers, and the same local naming route.

Use common-tail modulus stability and seal stability from Definition 1639.6 and Definition 1639.3. These theorems force the terminal Real seal to remain over the displayed endpoint, tail modulus, common-tail window, transport, replay, provenance, and local NameCert rows.

Finally apply the public Real-seal export of Definition 1639.20. The export is already quotient-free, so the CauchyFilter-to-Real route cannot acquire Real completeness, quotient stream equality, selected limits, host equality, or countable choice. □

Theorem 1639.22 (Bishop regular Cauchy completion consumer exactness)**.** *Every* Completion$^{\uparrow}$*-facing consumer of an accepted* BishopRegularCauchyCompletion$^{\uparrow}$ *packet is exhausted by the displayed dyadic endpoint ledger,* StreamName$^{\uparrow}$ *finite observations,* RegSeqRat$^{\uparrow}$ *readback, dyadic tail budget, classifier transport, terminal* Real$^{\uparrow}$ *seal handoff, and local* NameCert *rows. Equivalently, the consumer-facing route is*

$$E, S, R, D, W, H, C, P, N \longmapsto \mathsf{Completion}^{\uparrow},$$

where the left side is the concrete regular Cauchy completion packet over DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The route exports no ambient* Real$^{\uparrow}$ *completeness, quotient stream equality, selected limit, countable-choice schedule, host real equality, or detached equality of Real names.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionConsumerExactness`

Proof. First apply seal stability from Definition 1639.3. It forces any terminal Real read to pass through the finite endpoint, stream-window, regular readback, tail budget, common-tail, transport, replay, provenance, and local naming rows.

Next use the universal handoff and nonescape exactness supplied by Definition 1639.16 and Definition 1639.15. These theorems keep completion-facing reads inside the displayed packet $E, S, R, D, W, H, C, P, N$ and leave no coordinate for quotient streams, selected limits, countable choice, host equality, or detached Real-name equality.

Then apply the public Real-seal export of Definition 1639.20. The public export is already the quotient-free handoff over DyadicRatCore endpoints, StreamName windows, RegSeqRat readback, the terminal Real seal, and the structural rows.

Finally use the filter-Real handoff of Definition 1639.21. It shows that even a CauchyFilter-facing basis reaches the terminal Real seal only through the same displayed route. Therefore every Completion-facing consumer sees exactly that finite regular Cauchy completion surface, and not an ambient completeness principle. □

Theorem 1639.23 (Bishop regular Cauchy completion abstract bridge route)**.** *Every abstract* $\mathsf{Completion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet must pass through the displayed carrier admission, classifier transport, regularity-envelope, common-tail modulus stability, Real-seal handoff, and nonescape rows before reading the abstract completion interface. The bridge-facing route is*

$$E, S, R, D, W, H, C, P, N \longmapsto \mathsf{Completion}^{\uparrow},$$

where the left side is the concrete DyadicRatCore, StreamName, RegSeqRat, Real-seal, transport, replay, provenance, and local NameCert packet. It exports no quotient stream equality, selected limit, host real equality, countable-choice schedule, or detached equality of Real names.

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionAbstractBridgeRoute`

Proof. Start with carrier admission from Definition 1639.1. The concrete packet exposes exactly the rows $E, S, R, D, W, H, C, P, N$, so any abstract completion-facing consumer must begin from those displayed coordinates.

Use the regularity envelope and common-tail modulus stability from Definition 1639.2 and Definition 1639.6. They force the StreamName observations, RegSeqRat regularity row, dyadic tail budget, and common-tail window to be read before the route can behave as a completion-facing object.

Apply classifier transport and the Real-seal handoff from Definition 1639.4 and Definition 1639.16. These rows keep the abstract-facing read tied to componentwise hsame transport, displayed Cont replay, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use nonescape exactness, the obligation package, and the public Real-seal export from Definition 1639.15, Definition 1639.18, and Definition 1639.20. They identify the concrete rows as the whole local NameCert surface and exclude quotient streams, selected limits, countable choice, host equality, and detached Real-name equality from the bridge-facing route. □

Theorem 1639.24 (Bishop regular Cauchy completion mature route)**.** *Every mature completion-facing read of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet factors, in order, through the obligation package, the scoped kernel route, the public Real-seal export, and the abstract bridge route. The resulting consumer boundary is over*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

It exports the finite regular Cauchy completion route to $\mathsf{Completion}^{\uparrow}$*-facing consumers without exporting* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, countable choice, host real equality, or detached equality of Real names.*

Proof. First apply the obligation package of Definition 1639.18. It exhausts the local rows by carrier admission, regularity envelope, Real-seal handoff, classifier transport, shared tail budget, ledger nonescape, and terminal Real consumer routing.

Next use the scoped kernel route of Definition 1639.19. It fixes the displayed dependency surface as the rows $E, S, R, D, W, H, C, P, N$ over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Then apply the public Real-seal export of Definition 1639.20. This makes the terminal Real read public only as a quotient-free seal over the same dyadic endpoint, stream-window, regular readback, tail, transport, replay, provenance, and local naming rows.

Finally apply the abstract bridge route of Definition 1639.23. It sends the same concrete packet toward $\mathsf{Completion}^{\uparrow}$ and adds no coordinate for Real completeness, quotient stream equality, selected limits, countable choice, host real equality, or detached equality of Real names. □

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionMatureRoute`

Theorem 1639.25 (Bishop regular Cauchy completion limit-witness route)**.** *Any* $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet must factor through dyadic tail modulus,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before reading a limit witness. The witness-facing route exports the mature regular Cauchy completion surface; it does not export quotient stream equality, selected limits outside the displayed packet, countable choice, host Real equality, detached Real-name equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionLimitWitnessRoute`

Proof. Start with the mature route of Definition 1639.24. It already factors every mature completion-facing read through the obligation package, scoped kernel route, public Real-seal export, and abstract bridge route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Apply the public Real-seal export of Definition 1639.20. The limit-witness consumer sees the terminal Real read only as a quotient-free seal after dyadic endpoints, stream windows, regular readback, tail data, transport, replay, provenance, and local naming have been displayed.

Next use the filter-Real handoff of Definition 1639.21. It keeps even filter-facing basis reads on the same displayed route before the terminal Real seal can be used, so a $\mathsf{RegularCauchyLimitWitness}^{\uparrow}$ consumer cannot bypass the finite tail and readback rows.

Finally apply nonescape exactness from Definition 1639.15. Since the carrier has no coordinate for quotient stream equality, selected limits outside the displayed packet, countable choice, host Real equality, detached Real-name equality, or ambient Real completeness, the witness-facing read remains exactly the finite regular Cauchy completion route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1639.1, Definition 1639.2, Definition 1639.3, Definition 1639.4, Definition 1639.6, Definition 1639.14, Definition 1639.15, Definition 1639.16, Definition 1639.18, Definition 1639.19, Definition 1639.20, Definition 1639.23, and Definition 1639.24 close the finite regular Cauchy completion dependency surface, public Real-seal export, mature consumer boundary, and bridge-facing route to the abstract completion interface over displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyCompletion}^{\uparrow}}, \mathsf{auditClean})$

Lean target: `BEDC.Derived.BishopRegularCauchyCompletionUp.BishopRegularCauchyCompletionMatureRoute`

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host real equality, selected limits, countable choice, or standard bridge checking.

Upgrade path: The next formal upgrade is a standard bridge theorem for the mature route over the same displayed BEDC primitives.

Constructive story: A BishopRegularCauchyCompletion$^{\uparrow}$ packet is generated from DyadicRatCore endpoints, StreamName finite observations, RegSeqRat regularity, a dyadic tail modulus, a common-tail window, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1640

A Concrete Naming Certificate for $\mathsf{UpperSemicontinuous}^{\uparrow}$

Following Bishop–Bridges constructive analysis, upper semicontinuity is read through upper-bound and sublevel windows rather than through a host topology. The $\mathsf{UpperSemicontinuous}^{\uparrow}$ packet is the upper-side companion of chapter 1536: it reads finite stream windows from chapter 126, regular rational readbacks from chapter 301, upper-real comparison rows from chapter 1447, located endpoint rows from **??**, and the terminal seal of chapter 38. The chapter therefore supplies the missing constructive dual interface for the L10 real-and-completion boundary without identifying it with the lower epigraph certificate.

Definition 1640.1 (UpperSemicontinuous carrier)**.** *An* $\mathsf{UpperSemicontinuous}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, F, S, W, R, O, H, C, P, N)$$

with the following visible rows:

1. *X is the source carrier row whose accepted points are read through displayed* BHist *histories and classifier transport;*
2. *F is the function graph or displayed value row whose finite reads are inspected only after a source window has been selected;*
3. *S is the sublevel ledger, consisting of finite rows (x, q) saying that the inspected value is bounded above by the rational threshold q at the displayed window;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite source, value, and threshold windows used by each sublevel inspection;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row giving the regular rational value windows and threshold witnesses consumed by S;*
6. *O is the* $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *comparison ledger used to read upper windows, located endpoint checks, terminal real seals, and the finite contrast with lower-side reads;*
7. *H records componentwise* hsame *transport for source histories, graph rows, sublevel rows, schedules, regular readbacks, comparison ledgers, provenance rows, and local naming rows;*
8. *C records displayed* Cont *replay from source admission through schedule selection, regular rational readback, sublevel inspection, located readback, and real-seal handoff;*

9. *P is the* Pkg *provenance over* UpperSemicontinuous$^{\uparrow}$, LowerSemicontinuous$^{\uparrow}$, UpperReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{UpperSemicontinuous}^{\uparrow}}$ *row.*

The classifier relates two accepted packets exactly when their source rows, finite sublevel windows, regular rational readbacks, upper-real comparisons, located readbacks, terminal Real$^{\uparrow}$ *seal rows, and provenance rows transport componentwise along H and replay through the same displayed routes in C. No host topology, quotient real equality, selected limit, trichotomy, countable choice, or* Real$^{\uparrow}$ *completeness principle is a carrier coordinate.*

Lean4 ✓ `BEDC.Derived.UpperSemicontinuousUp.UpperSemicontinuousTasteGate single carrier alignment`

Theorem 1640.2 (UpperSemicontinuous lower-dual boundary)**.** *The accepted* UpperSemicontinuous$^{\uparrow}$ *carrier is constructively dual to the* LowerSemicontinuous$^{\uparrow}$ *epigraph surface of chapter 1536: lower semicontinuity records finite lower-bound rows, while upper semicontinuity records finite upper-bound and sublevel rows over the same kind of stream schedule, regular rational readback, located endpoint check, transport, replay, provenance, and local naming route. This duality compares the two finite certificate surfaces and does not identify their carriers, classifiers, or exported* NameCert *rows.*

Lean4 ✓ `BEDC.Derived.UpperSemicontinuousUp.UpperSemicontinuousLowerDualBoundary`
↪ variant `BEDC.Derived.UpperSemicontinuousUp.UpperSemicontinuousCarrier lower dual boundary`

Proof. Read the lower-side packet through Definition 1536.1. Its distinguishing ledger is the finite epigraph row of lower rational bounds. Read the upper-side packet through Definition 1640.1. Its distinguishing ledger is the finite sublevel row of upper rational bounds.

The common route consists of displayed source histories, stream schedules, regular rational readbacks, located endpoint checks, Real$^{\uparrow}$ seal rows, componentwise hsame transport, Cont replay, package provenance, and a local name. These coordinates support a boundary comparison because every inspection on either side is finite and routed through named sibling certificates.

The two packets remain separate certificates. The lower carrier exports epigraph-facing rows; the upper carrier exports sublevel-facing rows. Since their local NameCert rows are different coordinates and neither carrier contains a quotient real equality or selected limit coordinate, the duality is a constructive boundary relation rather than an identification. □

Theorem 1640.3 (UpperSemicontinuous finite-window stability)**.** *Every upper-bound read from an accepted* UpperSemicontinuous$^{\uparrow}$ *carrier $U = (X, F, S, W, R, O, H, C, P, N)$ factors through the finite seal route*

$$W, R, O, H, C, P, N$$

where W is the StreamName$^{\uparrow}$ *schedule, R is the* RegSeqRat$^{\uparrow}$ *readback, O is the* UpperReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, *and* Real$^{\uparrow}$ *comparison ledger, and H, C, P, N are the transport, replay, provenance, and local naming rows. No consumer can read an upper bound by bypassing that finite route through host topology, quotient real equality, a selected limit, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with an upper-bound inspection of U. The source and value rows are not read directly: the schedule W first selects the finite source, value, and threshold windows that the inspection may observe.

The readback row R supplies the regular rational data attached to those windows. The sublevel ledger S records the upper-bound row (x, q) for that same inspection, and the comparison ledger O checks the row against the $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ surfaces.

Transport and replay are exhausted by H and C. They move the source, graph, sublevel, readback, comparison, and seal rows together, so the same finite route remains visible after classifier transport.

Finally, P and N expose exactly this route to downstream consumers. Since the carrier has no coordinate for host topology, quotient equality, selected limits, trichotomy, countable choice, or real completeness, those principles cannot be obtained from an upper-bound read. □

Lean4 ✓ `BEDC.Derived.UpperSemicontinuousUp.UpperSemicontinuousFiniteWindowStability`

Closure status for $\mathsf{NameCert}_{\mathsf{UpperSemicontinuous}^{\uparrow}}$.

Constructive story: A $\mathsf{UpperSemicontinuous}^{\uparrow}$ packet is generated from displayed source histories, finite sublevel rows, StreamName schedules, RegSeqRat readbacks, UpperReal and LocatedReal comparison ledgers, RealUp seal rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperSemicontinuous}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1640.1, Definition 1640.2, and Definition 1640.3 seed the finite upper-bound certificate over sublevel windows, stream schedules, regular rational readbacks, upper-real and located comparison rows, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpperSemicontinuous}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close host topology, quotient real equality, selected limits, trichotomy, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, sublevel-window exactness, upper-real classifier stability, located readback compatibility, transport replay, provenance, and real-seal non-escape for BEDC.Derived.UpperSemicontinuousUp.

Chapter 1641

A Concrete Naming Certificate for $\mathsf{SequentialCompactness}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{SequentialCompactness}^{\uparrow}$ packet records only explicit finite subsequence-selector and convergence-window data for compact-metric sequence consumers. It sits over the compact metric source of chapter 129, the sequence-window interface of chapter 126, the regular rational handoff of chapter 301, the terminal real-name seal of chapter 38, and the displayed selector route of chapter 1408. The chapter names the finite boundary that compactness consumers may inspect; it is not unrestricted sequential compactness and does not import choice-like subsequence selection.

Definition 1641.1 (Sequential compactness carrier)**.** *A* $\mathsf{SequentialCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, S, B, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite covering and metric inspection data bound the sequence consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *sequence-window row, recording the inspected source indices, target windows, and finite request schedule opened by the consumer.*
3. *B is the displayed* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *subsequence window, including the selected index row and the finite evidence that the selected windows stay inside the compact-metric source.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *convergence handoff row attached to the selected subsequence windows and rational tail budgets.*
5. *H is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed limit read after the regular rational handoff has been accepted.*
6. *C records componentwise* hsame *transports for K, S, B, R, H and displayed* Cont *replays from compact-metric source reads to sequence windows, selected subsequence windows, regular rational tail reads, and the terminal real seal.*
7. *P is the* Pkg *provenance over* $\mathsf{SequentialCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BolzanoWeierstrassSelecto}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through K, S, B, R, H, C, P, N. It has no coordinate for host sequential compactness, countable choice, quotient equality of streams, a subsequence selected outside B, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessTasteGate single carrier alignment`

Theorem 1641.2 (Sequential compactness NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompactness}^{\uparrow}$ *carrier*

$$Q = (K, S, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompactness}^{\uparrow}}$ *surface consists of the following finite obligation rows:*

1. *carrier admission for the compact-metric source row, sequence-window row, displayed selector window, regular rational convergence handoff, real seal, transports, provenance, and local naming row;*
2. *finite-window classifier exposure, requiring every sequence read to pass through S and every compactness read to pass through K before a selector row is inspected;*
3. *selector stability, requiring selected indices and selected windows to be exactly the rows displayed in B and stable under the componentwise transports in C;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness, requiring convergence reads to enter R only from the selected windows in B and the rational tail budgets displayed by the packet;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the terminal limit read to pass through H after R and forbidding any ambient completeness or quotient stream equality coordinate.*

No accepted packet carries a further obligation row for host sequential compactness, countable choice, external subsequence selection, quotient streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1641.1. The packet exposes exactly the compact source, sequence window, selector window, convergence handoff, real seal, transport, provenance, and local naming rows.

Finite-window classifier exposure follows from the route through K and S. A consumer first reads the compact metric source and the finite sequence windows; only then can it inspect the displayed selector window B.

Selector stability is internal to B and C. The selected indices and selected windows are the rows stored in B, and C transports those rows componentwise without adding a second selection principle.

The convergence handoff is the row R. It accepts only the selected windows displayed by B together with the rational tail budgets stored in the carrier, so the handoff cannot be replaced by host convergence of an unspecified subsequence.

Finally, the limit read is sealed by H. Since the carrier has no coordinate for host sequential compactness, countable choice, quotient stream equality, external selected subsequences, or ambient real completeness, none of those data can be exported by the local NameCert surface. □

Theorem 1641.3 (Sequential compactness selector refusal)**.** *No accepted* $\mathsf{SequentialCompactness}^{\uparrow}$ *consumer may replace the displayed selector rows by host sequential compactness, countable choice,*

quotient stream equality, a selected subsequence outside B, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Every accepted consumer factors through the displayed route*

$$K, S \longmapsto B \longmapsto R \longmapsto H, C, P, N.$$

Proof. Let a consumer read an accepted carrier $Q = (K, S, B, R, H, C, P, N)$. The only compactness source is K, and the only sequence access route is the finite window row S.

The only subsequence data are stored in B. Thus selected indices, selected windows, and the boundary evidence that they remain inside the compact-metric source must be read from B itself.

The only convergence route is then R, followed by the real seal H. The structural rows C, P, N transport, source, and name this same route; they do not add host sequential compactness, choice, quotient stream equality, an external subsequence, or ambient completeness.

Therefore every accepted consumer factors through K, S, B, R, H, C, P, N, and every attempted replacement by the listed external data lies outside the $\mathsf{SequentialCompactness}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessCarrier selector refusal`

Theorem 1641.4 (Sequential compactness compact-metric route)**.** *Every accepted* $\mathsf{SequentialCompactness}^{\uparrow}$ *packet*

$$Q = (K, S, B, R, H, C, P, N)$$

reads its compact source K through $\mathsf{CompactMetric}^{\uparrow}$*, then reads the finite sequence windows S, then reads the* $\mathsf{BolzanoWeierstrassSelector}^{\uparrow}$ *row B, and only then reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R and the* $\mathsf{Real}^{\uparrow}$ *seal H. No host sequential compactness theorem, choice-selected subsequence, quotient stream equality, or ambient completeness row can bypass the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source.*

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessCompactMetricRoute`

Proof. Project the carrier of Definition 1641.1. The first row K is the compact-metric source, and the sequence-window row S is the only route by which a consumer can inspect source indices and target windows.

Apply Definition 1641.2. Its finite-window classifier clause requires every sequence read to pass through S and every compactness read to pass through K before the selector row is inspected.

Now apply Definition 1641.3. It blocks host sequential compactness, countable choice, quotient stream equality, externally selected subsequences, and ambient real completeness. Therefore the only route to R and H is the displayed K, S, B, R, H, C, P, N route, with $\mathsf{CompactMetric}^{\uparrow}$ as the compact source. □

Theorem 1641.5 (Sequential compactness Bolzano-Weierstrass window)**.** *Every accepted* $\mathsf{SequentialCompactness}^{\uparrow}$ *packet consumed by a Bolzano-Weierstrass-facing route must factor through the finite window*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{BoundedRealSequence}^{\uparrow}, \quad \mathsf{RealSequenceClusterPoint}^{\uparrow}.$$

The route reads compact-metric finite nets, sequence windows, regular rational readback, RealUp seal rows, and the sibling surfaces of chapter 1415 and chapter 1580. It exports no unrestricted subsequence selector, countable choice, open-cover compactness theorem, quotient stream equality, or ambient Real completeness row.

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessBolzanoWeierstrassWindow`

Proof. Project the carrier of Definition 1641.1. The compact source K, sequence windows S, displayed selector row B, regular rational handoff R, and terminal Real seal H are the only rows available to a consumer.

Apply Definition 1641.4. It forces every route to R and H through the compact-metric source and the displayed sequence windows before the selector row can be inspected.

The bounded-real-sequence sibling supplies the finite bounded-window source surface, and the real-sequence-cluster-point sibling supplies the cluster-seal handoff. Reading those siblings through K, S, B, R, H, C, P, N gives the displayed Bolzano-Weierstrass-facing route while preserving the refusal of unrestricted selection, countable choice, open-cover compactness, quotient stream equality, and ambient Real completeness. □

Theorem 1641.6 (Sequential compactness finite-window exhaustion)**.** *Every subsequence-facing claim exported by an accepted* $\mathsf{SequentialCompactness}^{\uparrow}$ *carrier*

$$Q = (K, S, B, R, H, C, P, N)$$

must factor through the compact metric source K, the $\mathsf{StreamName}^{\uparrow}$ *windows S, the displayed selector window B, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal H before any structural row C, P, N is consumed. The export is exhausted by the finite route*

$$K, S \longmapsto B \longmapsto R \longmapsto H \longmapsto C, P, N.$$

It supplies no host sequential compactness theorem, no selector outside B, no quotient stream equality, no countable-choice schedule, no open-cover compactness theorem, and no ambient Real completeness row.

Lean4 ✓ `BEDC.Derived.SequentialCompactnessUp.SequentialCompactnessCarrier finite window exhaustion`

Proof. Project the carrier of Definition 1641.1. The only compact source row is K, the only source-window row is S, the only selector data are stored in B, and the only convergence and limit rows are R and H.

Use the NameCert obligations of Definition 1641.2. They force every sequence read through S, every compactness read through K, every selector read through the displayed window B, and every convergence read through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R before the $\mathsf{Real}^{\uparrow}$ seal H is available.

Apply the selector refusal theorem Definition 1641.3 and the compact-metric route theorem Definition 1641.4. Together they block replacement by host sequential compactness, choice, quotient streams, externally selected subsequences, and ambient completeness, while keeping $\mathsf{CompactMetric}^{\uparrow}$ as the source of the route.

Finally apply the Bolzano-Weierstrass window theorem Definition 1641.5. It identifies the downstream bounded-real-sequence and cluster-point reads as consumers of the same finite window. Thus every subsequence-facing export is exhausted by K, S, B, R, H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompactness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1641.1, Definition 1641.2, and Definition 1641.3 seed the finite selector-boundary surface for compact-metric sequence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly refuses unrestricted sequential compactness, host subsequence selection, countable choice, quotient stream equality, selected subsequences outside the displayed selector row, ambient Real completeness, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window classifier exposure, selector stability, RegSeqRat handoff exactness, RealUp seal nonescape, and selector-refusal proofs for BEDC.Derived.SequentialCompactnessUp.

Constructive story: A SequentialCompactness$^{\uparrow}$ packet is generated from a CompactMetric source row, StreamName sequence-window row, displayed BolzanoWeierstrassSelector subsequence window, RegSeqRat convergence handoff, Real seal row, componentwise transport and replay rows, provenance, and a local NameCert row.

Chapter 1642

A Concrete Naming Certificate for $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$

Following Bishop and Bridges, Cauchy continuity is read here as preservation of regular Cauchy names through explicit moduli. The $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ packet sits on the finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational source route of chapter 301, the pointwise modulus schedule of chapter 1486, the uniform compatibility boundary of chapter 391, the target tail-estimate route of chapter 1487, and the terminal real-name seal of chapter 38. It names the theorem surface that a modulus-controlled map sends displayed regular Cauchy source names to displayed regular Cauchy target names before any Real seal is consumed.

Definition 1642.1 (Cauchy continuity principle carrier)**.** *A* $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, M, U, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular rational Cauchy source data.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family through which source and target observations are inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested target tolerances, source-window budgets, and comparison rows.*
4. *M is the* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *data that converts each target tolerance into the source-window read used by the map.*
5. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *compatibility row, requiring the pointwise Cauchy-modulus route to agree with the available uniform read whenever a uniform packet is supplied.*
6. *T is the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *target handoff, recording the target tail estimate obtained after S, W, D, M, U have been replayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the target* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff carried by T is displayed.*
8. *H records componentwise* hsame *transport for S, W, D, M, U, T, E and for the replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a target tolerance request through D, then M, then the source row S and finite windows W, then U, then the target tail estimate T, and finally the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuityPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, M, U, T, E, H, C, P, N$. It has no coordinate for host function equality, quotient completion, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or equality of completed real values.*

Theorem 1642.2 (Cauchy continuity principle NameCert obligations)**.** *For every accepted* CauchyContinuityPrincip *carrier $Q = (S, W, D, M, U, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuityPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source regular Cauchy row, finite windows, dyadic tolerance ledger, Cauchy-continuity modulus, uniform compatibility row, target tail estimate, Real seal, transport, replay, provenance, and local naming row;*

2. *classifier transport, requiring H and C to preserve the ordered route D, M, S, W, U, T, E componentwise;*

3. *modulus exactness, requiring every target tolerance to reach its source-window budget only through the displayed row M and the dyadic ledger D;*

4. *regular-target handoff, requiring the target Cauchy read to pass through T before the Real seal E is exposed;*

5. *ledger non-escape, requiring P and N to export no host function equality, quotient completion, countable choice, selected limit, ambient Real completeness, or equality of completed real values.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1642.1. The projection exposes exactly the regular source row S, finite windows W, dyadic ledger D, modulus data M, uniform compatibility row U, target tail estimate T, Real seal E, and the structural rows H, C, P, N.

Classifier transport is componentwise. The transport row H moves the displayed coordinates to a common finite surface, and the replay row C repeats the same dependency order D, M, S, W, U, T, E for every accepted tolerance request.

Modulus exactness is exhausted by M and D. A target tolerance first reads the dyadic ledger, then the Cauchy-continuity modulus row, and only then obtains the source-window budget used to inspect S through W.

The regular-target handoff is the row T. It records the finite target tail estimate produced after the source row, windows, ledger, modulus, and compatibility row have been replayed, so the Real seal E is downstream of the regular target data.

Finally inspect P and N. Since the carrier contains no host function equality, quotient completion, countable-choice source, selected limit, ambient completeness witness, or equality of completed real values, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyContinuityPrincipleUp.TasteGate.CauchyContinuityPrincipleTasteGate single carrier alignment`

Theorem 1642.3 (Cauchy continuity principle RegSeqRat handoff)**.** *Let* $Q = (S, W, D, M, U, T, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ *carrier. Every target* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by the packet factors through the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *row* M*, the* $\mathsf{UniformContinuity}^{\uparrow}$ *compatibility row* U*, and the target* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *row* T*.*

Lean4 ✓ `BEDC.Derived.CauchyContinuityPrincipleUp.CauchyContinuityPrincipleCarrier regseqrat handoff`

Proof. Start with the source side. Carrier admission gives the source regular rational Cauchy row S and the finite window family W, so source observations are available only as displayed finite-window reads.

Next read the tolerance side. The dyadic ledger D supplies the requested target tolerance and comparison budget, and the modulus row M converts that request into the source-window budget used on S and W.

The compatibility row U does not add an external modulus. It records that the same finite route agrees with the available uniform-continuity packet whenever that packet is present.

The target handoff is then forced through T. This row records the regular Cauchy target tail estimate produced from the replayed route, and H, C, P, N transport, replay, source, and name only those displayed coordinates. Hence every target $\mathsf{RegSeqRat}^{\uparrow}$ read factors through the stated route and cannot be obtained from host function equality, quotient completion, countable choice, selected limits, or ambient Real completeness. □

Theorem 1642.4 (Cauchy continuity principle modulus-composition handoff)**.** *Let*

$$Q_1 = (S_1, W_1, D_1, M_1, U_1, T_1, E_1, H_1, C_1, P_1, N_1)$$

and

$$Q_2 = (S_2, W_2, D_2, M_2, U_2, T_2, E_2, H_2, C_2, P_2, N_2)$$

be accepted $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ *carriers. Suppose the target* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff displayed by* T_1 *is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S_2 *read by the second packet. Then every composite target* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by* Q_2 *factors through the finite route* $D_2, M_2, S_2, W_2, U_2, T_2$*, with the displayed occurrence of* S_2 *supplied by the first packet route* $D_1, M_1, S_1, W_1, U_1, T_1$*. The terminal* $\mathsf{Real}^{\uparrow}$ *seal is exposed only after both finite routes have been replayed.*

Lean4 ✓ `BEDC.Derived.CauchyContinuityPrincipleUp.CauchyContinuityPrincipleCarrier modulus composition handoff`

Proof. Apply Definition 1642.2 to Q_1 and Q_2. Carrier admission gives the two displayed rows $S_i, W_i, D_i, M_i, U_i, T_i, E_i, H_i, C_i, P_i, N_i$, while classifier transport and replay preserve the order $D_i, M_i, S_i, W_i, U_i, T_i,$ in each packet.

Apply Definition 1642.3 to Q_2. Its exported target $\mathsf{RegSeqRat}^{\uparrow}$ read must pass through $D_2, M_2, S_2, W_2, U_2, T_2$, so the only source regular Cauchy row available to that read is the displayed row S_2.

Use the assumed handoff identification. The row S_2 is not an ambient equality of completed values; it is the target $\mathsf{RegSeqRat}^{\uparrow}$ handoff displayed by T_1. Applying Definition 1642.3 to Q_1 therefore replaces that source read by the finite route $D_1, M_1, S_1, W_1, U_1, T_1$.

Finally replay the seal order from the two obligation surfaces. Since E_2 is downstream of T_2, and T_2 has already consumed the displayed S_2 supplied by T_1, the terminal $\mathsf{Real}^{\uparrow}$ seal is visible

only after the second route and the first route have both been displayed. The carrier coordinates contain no host function equality, quotient completion, countable choice, selected limit, ambient Real completeness, or completed-real equality, so there is no additional route for the composite read. □

Theorem 1642.5 (Cauchy continuity principle Real seal boundary)**.** *For every accepted* $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ *carrier* $Q = (S, W, D, M, U, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *consumer of the seal row* E *first reads the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CauchyModulusOfContinuity}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{RegularCauchyTail}$$

Consequently the consumer receives a finite source-name, tolerance, modulus, compatibility, target-tail, and seal route, and receives no host function equality, quotient completion, countable-choice selector, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or completed-real equality principle.*

Lean4 ✓ `BEDC.Derived.CauchyContinuityPrincipleUp.CauchyContinuityPrincipleCarrier real seal boundary`

Proof. Read the seal coordinate in Definition 1642.1. The row E is listed only as the terminal $\mathsf{Real}^{\uparrow}$ seal reached after the target tail estimate T is displayed.

Use Definition 1642.3. The target regular Cauchy read has already passed through S, W, D, M, U, T before a real-facing consumer can inspect the seal.

Replay the structural rows. The continuation row C preserves the dependency order, the transport row H preserves the same finite coordinates, and the provenance and naming rows P, N record no additional source.

Thus the Real consumer receives exactly the displayed finite route through the sibling surfaces. Because none of the excluded principles is present as a carrier coordinate, no consumer can bypass the route by reading host function equality, quotient completion, countable choice, a selected limit, ambient Real completeness, or equality of completed real values. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuityPrinciple}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContinuityPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1642.1, Definition 1642.2, Definition 1642.3, Definition 1642.4, and Definition 1642.5 seed the finite theorem surface connecting modulus-controlled maps to preservation of displayed regular Cauchy names before Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContinuityPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient completion, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, completed-real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier admission and handoff proofs for BEDC.Derived.CauchyContinuityPrincipleUp over the same $S, W, D, M, U, T, E, H, C, P, N$ rows.

Constructive story: A $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$ packet is generated from a source RegSeqRat row, finite StreamName windows, a DyadicRatCore tolerance ledger, CauchyModulusOfContinuity data, UniformContinuity compatibility, RegularCauchyTailEstimate target handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1643

A Concrete Naming Certificate for $\mathsf{RieszRepresentation}^{\uparrow}$

Following Riesz, the Riesz representation packet treats a bounded scalar-valued functional as data that can be read against a representing vector or representing measure only after the relevant Banach, Hilbert, norm, and measure surfaces have already supplied their carriers and classifiers. It depends on $\mathsf{NameCert}_{\mathsf{Banach}^{\uparrow}}$ (chapter 90), $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ (chapter 91), $\mathsf{NameCert}_{\mathsf{FunctionalAnalysis}^{\uparrow}}$ (chapter 94), $\mathsf{NameCert}_{\mathsf{Norm}^{\uparrow}}$ (chapter 89), and $\mathsf{NameCert}_{\mathsf{Measure}^{\uparrow}}$ (chapter 92). The packet is a cross-chapter representation surface: it names the carrier and obligations needed by later spectral and measure-representation chapters without asserting every classical representation theorem at once.

1643.1 Carrier definition

Definition 1643.1 (RieszRepresentation carrier)**.** *A* $\mathsf{RieszRepresentation}^{\uparrow}$ *carrier is a representation row*

$$\mathsf{RieszRep}(S, T, \lambda, r, \Lambda)$$

whose source S is either a Hilbert row or a locally measured Banach-function row supplied by the dependency certificates named above. The target T is the scalar $\mathsf{Real}^{\uparrow}$ *face used by the bounded-functional row of* $\mathsf{FunctionalAnalysis}^{\uparrow}$*, λ is the bounded scalar-valued functional row, r is the proposed representing datum, and Λ is the visible ledger binding the functional row to the representing datum.*

In the Hilbert branch, r is a carried Hilbert endpoint and the ledger records the row

$$\lambda(x) \sim_{\mathsf{Real}^{\uparrow}} \langle x, r \rangle$$

for every carried Hilbert endpoint x, using the inner-product and norm rows of $\mathsf{Hilbert}^{\uparrow}$*. In the measure branch, r is a finite signed or positive measure row over the displayed measurable carrier, and the ledger records scalar integration rows against bounded test endpoints. The classifier compares two representation rows only by inherited transport of the functional endpoint, representing datum, scalar value rows, and ledger entries; it does not identify host-level functions, host-level vectors, or host-level measures.*

1643.2 Certificate obligations

Theorem 1643.2 (RieszRepresentation NameCert obligation surface)**.** *The* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet exposes exactly the following NameCert obligation surface:*

1. *a source row from* $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, *and* $\mathsf{Measure}^{\uparrow}$ *identifying the carried functional domain;*
2. *a bounded scalar-valued functional row inherited from* $\mathsf{FunctionalAnalysis}^{\uparrow}$*;*
3. *a representing datum row, either a Hilbert endpoint or a measure endpoint, with its own carrier and classifier;*
4. *a scalar evaluation ledger comparing the functional value with the inner-product or integration value on every carried test endpoint;*
5. *a boundary row saying which representation branch is present and which branch is outside the current certificate.*

The classifier preserves the packet precisely when these five rows are transported by their inherited classifiers.

Proof. Read the bounded functional through the dual-space face of $\mathsf{FunctionalAnalysis}^{\uparrow}$. That face already fixes the source carrier, scalar target, pointwise classifier, norm-bound row, and visible ledger, so the representation packet begins with a stable functional endpoint rather than a host-level functional equality.

For the Hilbert branch, use the carrier-sharing rows of $\mathsf{Hilbert}^{\uparrow}$. The representing datum is a carried Hilbert endpoint, and the displayed scalar comparison is an inner-product value row transported by the same Hilbert classifier that transports the functional source endpoint.

For the measure branch, use the event and value rows of $\mathsf{Measure}^{\uparrow}$. The representing datum is a measure endpoint, and the displayed scalar comparison is an integration ledger over the carried test endpoints. Transport preserves the test endpoint, measure endpoint, scalar output, and evidence row componentwise.

These two branches share only the bounded-functional and scalar-output faces. The boundary row records which branch is active, so the certificate does not collapse vector representation and measure representation into one host-level equality. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationNameCertObligations`

Theorem 1643.3 (RieszRepresentation Hilbert functional bridge)**.** *For a Hilbert source already accepted by* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$*, a* $\mathsf{RieszRepresentation}^{\uparrow}$ *Hilbert branch row packages a bounded scalar-valued functional* λ *together with a representing endpoint* r *and the scalar ledger*

$$\lambda(x) \sim_{\mathsf{Real}^{\uparrow}} \langle x, r \rangle$$

on every carried endpoint x*. If the functional row, Hilbert endpoint row, or representing endpoint row is transported by the inherited classifiers, then the displayed scalar ledger transports to the new row and exposes the same* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *dual-space face.*

Proof. Start from the $\mathsf{Hilbert}^{\uparrow}$ dependency. It supplies a Banach surface, an inner-product surface, the norm-induced compatibility ledger, and classifier transport for carried endpoints.

Next read λ through the $\mathsf{FunctionalAnalysis}^{\uparrow}$ dual-space face. Its pointwise classifier compares scalar outputs on carried Hilbert inputs, so it is already stated in the form needed by the displayed ledger.

Finally transport the representing endpoint through the Hilbert classifier. Inner-product stability moves $\langle x, r\rangle$ along the transported endpoint rows, and the scalar $\mathsf{Real}^{\uparrow}$ classifier moves the value comparison. The repacked row is therefore the same dual-space face plus a transported representing endpoint ledger. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationHilbertFunctionalBridge`

Theorem 1643.4 (Riesz representation dual functional route)**.** *Every* $\mathsf{RieszRepresentation}^{\uparrow}$ *bounded-functional branch must read its source through the displayed* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *dual-space row, the* $\mathsf{Norm}^{\uparrow}$ *bound row, and the* $\mathsf{Banach}^{\uparrow}$ *carrier row before a representing datum is consumed. The route exposes the functional endpoint, scalar* $\mathsf{Real}^{\uparrow}$ *target, bound ledger, source carrier, and branch boundary as finite rows; it does not assert arbitrary dual identification, nonconstructive Hahn–Banach extension, uniqueness of representing data, or host-level equality of linear maps.*

Proof. Start with the source row of Definition 1643.2. It already requires the Banach, FunctionalAnalysis, and Norm certificates to be visible before a bounded scalar-valued functional row is read.

Next use the Hilbert bridge of Definition 1643.3 only for the branch in which the source is Hilbert. The same bounded-functional face remains the entry point, but the representing datum is then a carried Hilbert endpoint rather than an arbitrary dual object.

For the non-Hilbert branch, keep the boundary row active. Since no row supplies arbitrary dual identification, Hahn–Banach extension, representing uniqueness, or host-level linear-map equality, the route is exactly the finite dual-functional surface named in the theorem. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationDualFunctionalRoute`

Theorem 1643.5 (Riesz representation measure handoff)**.** *The measure branch of a* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet hands a bounded functional row to* $\mathsf{Measure}^{\uparrow}$ *only after the source carrier, scalar target, boundedness ledger, measurable endpoint, and test-endpoint integration ledger have been displayed. The handoff records a finite measure-facing representation surface; it does not claim uniqueness of representing measures, full measure representation, Radon regularity, countable choice, or ambient completeness of the scalar target.*

Proof. Read the bounded functional through Definition 1643.4. This fixes the source carrier, scalar target, and norm-bound ledger before the measure endpoint is inspected.

Now project the measure branch listed in Definition 1643.1. The representing datum is a measure endpoint, and the scalar comparison is an integration ledger over carried bounded test endpoints.

The branch boundary records that this is a handoff to $\mathsf{Measure}^{\uparrow}$, not a full representation theorem. Because the packet has no coordinate for uniqueness, Radon regularity, countable choice, or ambient scalar completeness, the handoff cannot export those endpoints. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationMeasureHandoff`

Theorem 1643.6 (Riesz representation Banach-Hilbert scope)**.** *The exact scope of the* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet is the finite family of rows supplied by* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{FunctionalAnalysis}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{Measure}^{\uparrow}$*: a source carrier, bounded scalar-valued functional, norm-bound ledger, optional Hilbert representing endpoint, optional measure endpoint, scalar comparison ledger, branch boundary, and inherited classifier transport. No accepted packet extends this scope to arbitrary nonconstructive duality, spectral representation, selected representing datum, quotient equality of functions, or uniqueness of representing measures.*

Proof. Combine the obligation surface of Definition 1643.2 with the Hilbert bridge of Definition 1643.3. These two rows display the source carrier, bounded functional, Hilbert representing endpoint when present, scalar comparison ledger, and branch boundary.

Add the dual-functional route and measure handoff above. They force the Banach, FunctionalAnalysis, Norm, and Measure rows to remain visible before either a Hilbert or measure representation read is consumed.

The classifier of Definition 1643.1 compares packets only through the inherited rows and ledger entries. Therefore the scope is exactly the five named dependency certificates and their branch rows, with the excluded representation theorems left outside the packet. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationBanachHilbertScope`

Theorem 1643.7 (Riesz representation branch-boundary nonescape)**.** *Every accepted* RieszRepresentation$^{\uparrow}$ *packet keeps the Hilbert representing-endpoint row and the measure representing-endpoint row separated by the branch boundary of Definition 1643.1. A consumer may project one branch only after reading the shared* FunctionalAnalysis$^{\uparrow}$ *bounded-functional row and the active branch row. It cannot infer representing uniqueness, Radon regularity, Hahn–Banach extension, spectral representation, or host-level equality from the inactive branch.*

Proof. Use the NameCert obligation surface of Definition 1643.2. Its boundary row is part of every accepted packet, so a consumer cannot read either representing endpoint before the active branch has been displayed.

For the Hilbert side, apply the Hilbert functional bridge of Definition 1643.3. It identifies the representing datum only as a carried Hilbert endpoint tied to the shared bounded-functional row and scalar comparison ledger.

For the measure side, apply the dual-functional route of Definition 1643.4 and the measure handoff of Definition 1643.5. These rows keep the bounded-functional face visible while exposing only the measure endpoint and its test-endpoint integration ledger.

Finally project the boundary coordinate from the carrier definition. Since the classifier compares packets through the active branch rows and the shared bounded-functional row, no inactive branch coordinate can export uniqueness, Radon regularity, Hahn–Banach extension, spectral representation, or host-level equality. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationBranchBoundaryNonescape`

Theorem 1643.8 (RieszRepresentation public export)**.** *The public export surface of a* RieszRepresentation$^{\uparrow}$ *packet is the finite certificate obtained by composing the obligation surface of Definition 1643.2, the Hilbert functional bridge of Definition 1643.3, the dual-functional route of Definition 1643.4, the measure handoff of Definition 1643.5, the Banach-Hilbert scope of Definition 1643.6, and the branch-boundary nonescape theorem of Definition 1643.7. A public consumer may read only the source carrier, bounded scalar-valued functional row, active representing datum row, scalar comparison ledger, inherited classifier transports, and branch boundary displayed by these certificates; it receives no Hahn–Banach extension, representing uniqueness, Radon regularity, spectral representation, selected representative, or host-level equality of functional rows.*

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationPublicExport`

Proof. Start with the obligation surface. It supplies the source row, bounded scalar-valued functional row, representing datum row, scalar evaluation ledger, and branch boundary as the local NameCert inventory.

For the Hilbert branch, use the Hilbert functional bridge to expose only a carried Hilbert endpoint and its inner-product scalar ledger. For the shared bounded-functional entry, use the dual-functional route so that every public read still passes through the displayed $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, and $\mathsf{Banach}^{\uparrow}$ rows.

For the measure branch, use the measure handoff to expose only the representing measure endpoint and the integration ledger over carried test endpoints. Then apply the Banach-Hilbert scope and branch-boundary nonescape theorems: together they keep the active branch public while preventing the inactive branch or any unlisted representation theorem from escaping the packet. □

Theorem 1643.9 (Riesz representation obligation closure package)**.** *The carrier, obligation surface, Hilbert bridge, dual-functional route, measure handoff, Banach-Hilbert scope, and branch-boundary nonescape of this chapter assemble the full obligationClosure surface for* $\mathsf{RieszRepresentation}^{\uparrow}$ *over* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{FunctionalAnalysis}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{Measure}^{\uparrow}$*. The package closes exactly the displayed finite representation obligations and does not claim a full representation theorem, nonconstructive duality, Hahn–Banach extension, selected representative, or uniqueness of representing data.*

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationObligationClosure`

Proof. Start with the carrier of Definition 1643.1 and the obligation surface of Definition 1643.2. Together they identify the source row, bounded scalar-valued functional row, representing datum row, scalar evaluation ledger, and branch boundary as the only public rows of the packet.

Apply the Hilbert bridge Definition 1643.3 and the dual-functional route Definition 1643.4. These results force every Hilbert-facing or bounded-functional read through the displayed $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, and $\mathsf{Banach}^{\uparrow}$ rows before a representing datum is consumed.

Use the measure handoff Definition 1643.5 for the measure branch, then combine the Banach-Hilbert scope theorem Definition 1643.6 with branch-boundary nonescape Definition 1643.7. The resulting package is exactly the finite obligation surface over the five named dependencies, while full representation, nonconstructive duality, Hahn–Banach extension, selected representatives, and uniqueness remain outside the carrier. □

Theorem 1643.10 (Riesz representation scope binding)**.** *Every accepted* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet scopes the bounded-functional route through* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{FunctionalAnalysis}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{Measure}^{\uparrow}$ *before either branch is projected. The branch-boundary rows keep Hilbert representing endpoints and measure endpoints separate, so a Hilbert projection cannot export a measure endpoint and a measure projection cannot export a Hilbert representing vector. The scoped packet exposes only the shared bounded-functional face, the active representing row, scalar comparison ledgers, inherited classifier transport, and the branch boundary.*

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationScopeBinding`

Proof. Start with the obligation surface of Definition 1643.2. It requires the source row, bounded scalar-valued functional row, representing datum row, scalar ledger, and branch boundary to be displayed as the local NameCert surface.

For the Hilbert branch, apply the Hilbert bridge Definition 1643.3. For the shared bounded-functional entry, use the dual-functional route Definition 1643.4. These bind the $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, and $\mathsf{Banach}^{\uparrow}$ rows before a representing endpoint is consumed.

For the measure branch, apply the measure handoff Definition 1643.5. Then combine Banach-Hilbert scope Definition 1643.6 with branch-boundary nonescape Definition 1643.7. The boundary

coordinate keeps the Hilbert and measure endpoints disjoint while preserving the shared bounded-functional face, so the accepted packet has exactly the scoped route stated above. □

Theorem 1643.11 (Riesz representation functional-carrier obligation)**.** *Every accepted* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet must display the functional-analysis, Banach, Hilbert, Norm, and Real rows that make the bounded scalar-valued functional readable before any representing datum is projected. The obligation is the finite row*

$$(F, A, H, N, R, B)$$

where F *is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *bounded-functional face,* A *is the* $\mathsf{Banach}^{\uparrow}$ *source carrier,* H *is the optional* $\mathsf{Hilbert}^{\uparrow}$ *branch source,* N *is the* $\mathsf{Norm}^{\uparrow}$ *bound ledger,* R *is the scalar* $\mathsf{Real}^{\uparrow}$ *seal, and* B *is the branch boundary. The row exports carrier admission and classifier transport only; it does not assert a host representation theorem, a selected representing vector, or equality of host linear maps.*

Proof. Read the carrier of Definition 1643.1. Its source and bounded-functional coordinates already require the Banach, Hilbert, FunctionalAnalysis, Norm, and Real faces before a representing datum can be consumed.

Apply the dual-functional route of Definition 1643.4. It keeps the bounded scalar-valued functional row visible as a finite NameCert coordinate rather than as a host-level function equality.

Finally use the branch-boundary nonescape theorem. Since B is part of the displayed row, the functional-carrier obligation cannot select a representing datum or identify host linear maps outside the active branch. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationFunctionalCarrierObligation`

Theorem 1643.12 (Riesz representation measure-row obligation)**.** *In the measure branch, every accepted* $\mathsf{RieszRepresentation}^{\uparrow}$ *packet must display the Measure and Real rows*

$$(M, T, I, R, L, B)$$

before a measure-facing representation read is exported. Here M *is the* $\mathsf{Measure}^{\uparrow}$ *representing endpoint,* T *is the finite test-endpoint schedule,* I *is the integration ledger over those tests,* R *is the scalar* $\mathsf{Real}^{\uparrow}$ *seal,* L *is the inherited bounded-functional ledger, and* B *records that the measure branch is active. This obligation exports only finite NameCert rows for the measure handoff; it does not choose a representing measure, prove uniqueness, assert Radon regularity, or close a host measure-representation theorem.*

Proof. Start from the measure branch in Definition 1643.1. The representing datum is a carried measure endpoint, and the scalar comparison is an integration ledger over displayed test endpoints.

Use the measure handoff theorem Definition 1643.5. It forces the source carrier, scalar target, boundedness ledger, measure endpoint, and test-endpoint integration ledger to be visible before the handoff is consumed.

The branch boundary then blocks the extra conclusions. Because no row carries a selected measure, uniqueness proof, Radon regularity theorem, or host-level representation theorem, the measure obligation remains a finite packet row. □

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationMeasureRowObligation`

Theorem 1643.13 (Riesz representation Real-seal obligation)**.** *Every scalar comparison exported by* $\mathsf{RieszRepresentation}^{\uparrow}$ *must pass through the terminal* $\mathsf{Real}^{\uparrow}$ *seal row shared by the Hilbert and measure branches. In the Hilbert branch the sealed comparison is the inner-product value ledger; in the measure branch it is the integration value ledger. The seal exports only scalar value rows and their inherited classifier transport, not Real completeness, selected representatives, equality of host functions, or a full representation theorem.*

Lean4 ✓ `BEDC.Derived.RieszRepresentationUp.RieszRepresentationRealSealObligation`

Proof. For the Hilbert branch, apply Definition 1643.3. The comparison row is a scalar ledger between the functional value and an inner-product value, both read through the same Real classifier.

For the measure branch, apply Definition 1643.5. The comparison row is an integration value ledger over displayed tests, again sealed by the scalar Real row.

Combine these branches with Definition 1643.6. The common Real seal is the only scalar export coordinate, so ambient Real completeness, selected representatives, host function equality, and full representation theorems remain outside the packet. □

Theorem 1643.14 (Riesz representation scoped-closure handoff)**.** *The scoped closure surface for* $\mathsf{RieszRepresentation}^{\uparrow}$ *is exactly the composition of the carrier of Definition 1643.1, the obligation surface of Definition 1643.2, the Hilbert functional bridge of Definition 1643.3, the dual-functional route of Definition 1643.4, the measure handoff of Definition 1643.5, the Banach-Hilbert scope of Definition 1643.6, the branch-boundary nonescape theorem of Definition 1643.7, the functional-carrier obligation of Definition 1643.11, the measure-row obligation of Definition 1643.12, and the Real-seal obligation of Definition 1643.13. Every consumer must display the* $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before projecting either the Hilbert branch or the measure branch.*

Proof. Begin with the carrier and NameCert obligation surface. They provide the source row, bounded functional row, representing datum row, scalar comparison ledger, inherited transport, and branch boundary as the only rows available to a consumer.

For the Hilbert branch, apply the Hilbert functional bridge together with the Banach-Hilbert scope. These force the $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, and $\mathsf{Norm}^{\uparrow}$ rows to remain visible before a representing endpoint is projected.

For the measure branch, apply the dual-functional route, the measure handoff, and the measure-row obligation. These bind the $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before a measure-facing representation read is exported.

Finally use branch-boundary nonescape, the functional-carrier obligation, and the Real-seal obligation. The branch boundary keeps the Hilbert and measure projections separate, while the Real seal is the only scalar export coordinate. Therefore scoped closure is exactly the displayed composition and not a full representation theorem, selected representative, uniqueness theorem, or host-level equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszRepresentation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RieszRepresentation}^{\uparrow}$ packet is generated from an inherited bounded scalar-valued functional row, a source row from the Banach, Hilbert, FunctionalAnalysis, Norm, and Measure certificates, a representing Hilbert endpoint or measure endpoint, scalar comparison ledgers for inner-product or integration values, componentwise classifier transport, and a branch-boundary row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszRepresentation}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1643.1, Definition 1643.2, Definition 1643.3, Definition 1643.5, Definition 1643.6, and Definition 1643.14 bind the carrier, NameCert obligation surface, Hilbert functional bridge, measure handoff, Banach-Hilbert scope, and scoped-closure handoff over the existing Banach, Hilbert, FunctionalAnalysis, Norm, Measure, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RieszRepresentation}\uparrow}$, auditClean)

Lean target: `BEDC.Derived.RieszRepresentationUp.RieszRepresentationNameCertObligations`

Bridge status: none

Not claimed: It does not claim full measure-representation, arbitrary nonconstructive duality, spectral representation, uniqueness of representing measures, Lean verification, or a standard bridge.

Upgrade path: Public closure requires a public export package that packages the scoped rows for downstream consumers without adding full representation, selected representatives, uniqueness, or bridge checking.

Chapter 1644

A Concrete Naming Certificate for $\mathsf{BoundedLinearOperator}^{\uparrow}$

Following Kreyszig and Conway, the $\mathsf{BoundedLinearOperator}^{\uparrow}$ packet records the standard bounded linear map surface as finite NameCert data over displayed Banach candidates. It is read from the Banach carrier of chapter 90, the linear-map obligations of chapter 48, the normed real-line rows of chapter 1500, the limit-witness interface of chapter 128, the stream window interface of chapter 126, and the regular rational handoff of chapter 301. The chapter exposes the public operator object already carried by the Banach bounded-linear-operator rows: endpoint action, linearity, norm bound, composition, identity and zero units, and transport of regular Cauchy and complete-metric limit witnesses.

Definition 1644.1 (Bounded linear operator carrier)**.** *For displayed* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D*, a* $\mathsf{BoundedLinearOperator}^{\uparrow}(C, D)$ *carrier is the public packaging of a Banach bounded-linear-operator row*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

from Definition 90.49. Its visible coordinates are the endpoint map T*, the RealUp norm-bound endpoint* K*, and the ledger* Λ *recording source and target endpoint rows, source and target norm rows, scalar rows, additive and homogeneous compatibility rows, classifier-respect rows, and bound-witness rows.*

The classifier compares two accepted representatives only by componentwise hsame *transport of their endpoint actions on carried source endpoints, RealUp transport of their bound endpoints, and componentwise transport of the displayed ledgers. It has no coordinate for host function extensionality, quotient equality of operators, selected limits, ambient completion data, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorTasteGate single carrier alignment`

Theorem 1644.2 (Bounded linear operator NameCert obligations)**.** *For displayed* $\mathsf{Banach}^{\uparrow}$ *candidates* C, D*, the carrier of Definition 1644.1 carries the public* $\mathsf{NameCert}_{\mathsf{BoundedLinearOperator}^{\uparrow}}$ *surface with the following obligation rows:*

1. *carrier admission for* $\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$*;*
2. *classifier stability under source and target Banach endpoint classifiers;*
3. *linearity exactness for additive and homogeneous endpoint action;*

4. *norm-bound exactness through the RealUp endpoint K;*
5. *Lipschitz and regular-Cauchy transport through the norm-induced metric rows;*
6. *zero, sum, scalar, identity, and composition rows when the corresponding Banach ledgers are displayed;*
7. *refusal of host function extensionality, operator quotients, selected limits, hidden Banach completeness, and bridge data.*

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorCarrier namecert obligations`

Proof. Carrier admission and classifier stability are projections of Definition 90.49. That definition already records the endpoint map, bound endpoint, and visible ledger, and it compares representatives only through displayed endpoint, bound, and ledger transports.

Linearity exactness is the conjunction of the additive and homogeneous compatibility rows in the same carrier. The rows are stated over carried Banach endpoints and carried RealUp scalars, so they are public operator data rather than host-level equations between functions.

The metric obligations are supplied by the Banach bounded-linear-operator surface. Apply Definition 90.50 for the displayed Lipschitz estimate, and apply Definition 90.51 for transport of regular Cauchy $\mathsf{StreamName}^{\uparrow}$ streams.

The algebraic operator rows are inherited from the Banach child files. The zero, pointwise sum, scalar, identity, and composition rows are accepted only when their visible Banach ledgers are present, and the public packet stores those rows as operator coordinates rather than as an ambient operator space quotient.

The refusal row follows from the carrier coordinates. Since the packet contains only T, K, Λ and their componentwise transports, there is no field from which host function extensionality, quotient operator equality, selected limits, hidden completeness, or a bridge can be read. □

Theorem 1644.3 (Bounded linear operator composition closure)**.** *Let C, D, E be displayed* $\mathsf{Banach}^{\uparrow}$ *candidates. If*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

and

$$\mathsf{BanachBLOp}(D, E, S, L, \Gamma)$$

are accepted $\mathsf{BoundedLinearOperator}^{\uparrow}$ *carrier rows, then their endpoint composite is accepted as*

$$\mathsf{BanachBLOp}(C, E, S \circ T, L \cdot K, \Gamma \star \Lambda).$$

The public composition ledger records exactly the two input operator ledgers together with the RealUp multiplication, monotonicity, associativity, and non-negativity rows needed for the product bound.

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorUpTasteGate single carrier alignment`

Proof. Read the two input rows through Definition 1644.1. They are precisely the Banach bounded-linear-operator rows required by Definition 90.64.

Apply that Banach composition theorem. It constructs the composite endpoint map $S \circ T$, the product bound $L \cdot K$, and the concatenated ledger $\Gamma \star \Lambda$, with classifier, additive, homogeneous, and norm-bound rows transported through the middle Banach carrier.

Repack the result as a $\mathsf{BoundedLinearOperator}^{\uparrow}(C, E)$ carrier. No further equality principle is used: endpoint comparison is pointwise over carried source endpoints, bound comparison is RealUp transport, and ledger comparison is the displayed concatenation row. □

Theorem 1644.4 (Bounded linear operator limit transport)**.** *Let C, D be displayed* $\mathsf{Banach}^{\uparrow}$ *candidates and let*

$$\mathsf{BanachBLOp}(C, D, T, K, \Lambda)$$

be an accepted $\mathsf{BoundedLinearOperator}^{\uparrow}(C, D)$ *carrier row. If a regular Cauchy* $\mathsf{StreamName}^{\uparrow}$ *stream over the norm-induced metric of C carries a* $\mathsf{CompleteMetric}^{\uparrow}$ *limit witness with limit endpoint ℓ, then the pointwise target stream carries a* $\mathsf{CompleteMetric}^{\uparrow}$ *limit witness over D with displayed limit endpoint $T(\ell)$.*

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorCarrier limit transport`

Proof. Read the accepted operator row as a Banach bounded-linear-operator carrier. The source limit witness is a complete-metric witness over the norm-induced metric attached to the source Banach candidate.

Apply Definition 90.75. Its construction first uses regular-Cauchy transport for the pointwise target stream and then applies the Lipschitz bound to late-distance estimates between stream observations and the source limit endpoint.

The resulting target witness has displayed limit endpoint $T(\ell)$ and its ledger is exactly the operator ledger together with the source and target complete-metric rows. Repacking that witness gives the public $\mathsf{BoundedLinearOperator}^{\uparrow}$ limit-transport surface. □

Theorem 1644.5 (Bounded linear operator nonescape boundary)**.** *Every accepted* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

In particular, the packet cannot export host function extensionality, quotient operator equality, selected limits, ambient Banach completeness beyond the displayed input rows, or bridge checking data.

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorCarrier nonescape boundary`

Proof. Let a consumer read an accepted carrier. By Definition 1644.1, the only operator coordinates are the endpoint map T, the RealUp bound endpoint K, and the visible ledger Λ.

The linear-map data are the additive and homogeneous rows inside Λ, the norm-bound data are the RealUp and NormUp rows inside Λ, and the metric data are obtained by the Lipschitz and regular-Cauchy transport rows cited in Definition 1644.2.

Composition and limit transport do not add hidden coordinates. Composition concatenates two displayed ledgers as in Definition 1644.3, and limit transport reuses the source complete-metric witness and target Banach rows as in Definition 1644.4.

Thus any public read factors through the displayed Banach, linear, norm, real, complete-metric, stream, regular-rational, history, transport, continuation, provenance, and naming rows. The excluded host and quotient objects are absent from the carrier and from the two transport constructions, so they cannot be exported by the packet. □

Theorem 1644.6 (Bounded linear operator scoped dependency route)**.** *Every accepted* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *packet reads its carrier, linearity row, norm-bound row, composition row, limit-transport row, and nonescape boundary only through*

$$\mathsf{Banach}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The route therefore binds the paper-side scope of the packet to displayed Banach, linear, norm, real, complete-metric, stream, regular-rational, history, transport, replay, provenance, and naming data.

Lean4 ✓ `BEDC.Derived.BoundedLinearOperatorUp.BoundedLinearOperatorCarrier scoped dependency route`

Proof. Read the accepted carrier through Definition 1644.1 and the local obligations in Definition 1644.2. These rows expose the endpoint action, linearity ledger, norm-bound endpoint, classifier transport, provenance, and local NameCert data.

Composition remains inside the same route by Definition 1644.3. The composite ledger is built from displayed source and target Banach rows, RealUp multiplication rows, and the two visible operator ledgers.

Limit transport remains inside the complete-metric and regular-rational route by Definition 1644.4. The proof consumes the source complete-metric witness, StreamName and RegSeqRat data, and the target Banach rows without adding a selected limit or hidden completeness coordinate.

Finally apply Definition 1644.5. Since that boundary excludes host function extensionality, quotient operator equality, selected limits, ambient Banach completeness beyond the displayed rows, and bridge checking, the listed dependencies are exactly the scoped route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearOperator}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearOperator}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1644.1, Definition 1644.2, Definition 1644.3, Definition 1644.4, Definition 1644.5, and Definition 1644.6 bind carrier admission, classifier stability, linearity and bound exactness, composition closure, limit transport, algebraic operator rows, nonescape, and the scoped dependency route over displayed Banach carriers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter explicitly excludes host function extensionality, quotient operator equality, selected limits, ambient Banach completeness beyond the input rows, and bridge checking.

Upgrade path: The theory axis has no higher closure grade; audit-clean and bridge-checked verification remain independent formal-axis obligations for BEDC.Derived.BoundedLinearOperatorUp.

Constructive story: A $\mathsf{BoundedLinearOperator}^{\uparrow}$ packet is generated from displayed source and target Banach candidates, an endpoint action, a RealUp norm-bound endpoint, visible linearity and bound ledgers, regular-Cauchy and complete-metric transport rows, componentwise transport, provenance, and a local NameCert row.

Chapter 1645

A Concrete Naming Certificate for $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ packet names the finite interval-operation surface obtained when both endpoints are displayed $\mathsf{DyadicRatCore}^{\uparrow}$ rows and every enclosure claim is routed through rational and real interval packets already present in the BEDC library. It sits after the dyadic endpoint core of chapter 291, the rational interval carrier of chapter 354, the metric-ball observation surface of chapter 1450, the located interval boundary of chapter 394, and the real interval enclosure route of chapter 1625. The packet records addition, multiplication, refinement, and containment comparison as finite endpoint ledgers; it is not a quotient interval type, a selected real representative, a trichotomy service, a countable-choice construction, an open-cover compactness argument, or an appeal to Real completeness.

Definition 1645.1 (Dyadic interval arithmetic carrier)**.** *A* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, U, O, +_{\Delta}, \cdot_{\Delta}, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are lower and upper endpoint rows accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *O is the endpoint-order ledger, recording that the lower endpoint row is below the upper endpoint row through the same dyadic comparison surface.*
3. *$+_{\Delta}$ is the addition-operation row, whose lower and upper outputs are computed from the displayed dyadic endpoint additions with outward endpoint witnesses.*
4. *$\cdot_{\Delta}$ is the multiplication-operation row, whose four endpoint products and selected lower and upper dyadic bounds are displayed as finite rows.*
5. *R is the refinement row, recording that a later dyadic endpoint pair tightens the current pair through endpoint comparisons over the same scale ledger.*
6. *E is the enclosure-comparison row, relating the dyadic endpoint pair to the* $\mathsf{RationalInterval}^{\uparrow}$ *and* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *containment rows consumed by downstream readers.*
7. *H records componentwise* hsame *transport for the endpoint, order, operation, refinement, enclosure, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the endpoint rows through the operation and enclosure rows.*

9. *P is the* Pkg *provenance over* DyadicIntervalArithmetic$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RationalInterval$^{\uparrow}$, MetricBall$^{\uparrow}$, LocatedInterval$^{\uparrow}$, RealIntervalEnclosure$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalArithmetic}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite dyadic endpoint interval-operation packet. It has no coordinate for quotient intervals, host real endpoints, selected real representatives, trichotomy, countable choice, open-cover compactness, or Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicIntervalArithmeticUp.DyadicIntervalArithmeticTasteGate single carrier alignment`

Theorem 1645.2 (Dyadic interval arithmetic NameCert obligations)**.** *For every accepted* DyadicIntervalArithmetic$^{\uparrow}$ *carrier $A = (L, U, O, +_{\Delta}, \cdot_{\Delta}, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalArithmetic}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two endpoint rows, endpoint-order ledger, addition row, multiplication row, refinement row, enclosure-comparison row, componentwise transport, continuation replay, provenance, and local naming row;*
2. *endpoint order stability, requiring O to transport along the displayed* hsame *and* Cont *rows without replacing the dyadic comparison by host order on reals;*
3. *operation enclosure soundness, requiring $+_{\Delta}$ and $\cdot_{\Delta}$ to compute their exported lower and upper endpoint rows from the displayed dyadic endpoint rows and outward comparison witnesses;*
4. *refinement monotonicity, requiring R to preserve containment when lower endpoints move upward and upper endpoints move downward through named dyadic comparison rows;*
5. *classifier transport, requiring every accepted operation and refinement row to remain accepted after componentwise* hsame *transport and displayed* Cont *replay;*
6. *ledger non-escape, excluding quotient intervals, selected real representatives, trichotomy, countable choice, open-cover compactness, Real completeness, and hidden host endpoint arithmetic.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.DyadicIntervalArithmeticUp.DyadicIntervalArithmeticCarrier namecert obligations`

Proof. Project the carrier of Definition 1645.1. The projection exposes exactly the two dyadic endpoint rows, endpoint-order ledger, operation rows, refinement row, enclosure-comparison row, and structural rows H, C, P, N.

Endpoint order stability is the row O transported by H and replayed by C. Since O is a dyadic comparison row, transport can only move the displayed endpoint histories and cannot insert an external order relation.

Operation enclosure soundness is read from $+_{\Delta}$ and $\cdot_{\Delta}$. The addition row uses the two endpoint additions with outward witnesses, while the multiplication row displays the four dyadic endpoint products and the selected lower and upper dyadic bounds. Each output endpoint is therefore read from the visible dyadic arithmetic ledger.

Refinement monotonicity is the row R. It records the tightening comparisons between the old and later endpoint rows, so containment can only be weakened along the displayed dyadic order witnesses.

Classifier transport follows by applying H and C componentwise to the same finite rows. The exclusions are immediate from the carrier inventory: there is no coordinate for quotient

intervals, selected real representatives, trichotomy, countable choice, open-cover compactness, Real completeness, or hidden host endpoint arithmetic. □

Theorem 1645.3 (Dyadic interval arithmetic endpoint-order transport)**.** *Let $A = (L, U, O, +_\Delta, \cdot_\Delta, R, E, H, C, P, N$ be an accepted* DyadicIntervalArithmetic$^\uparrow$ *carrier. If O is the displayed endpoint-order ledger and H, C are the componentwise* hsame *transport and displayed* Cont *replay rows, then lower-before-upper order is preserved for every transported dyadic interval packet read from A. The transport is a rowwise dyadic endpoint claim; it uses no host real order, trichotomy, selected representatives, quotient intervals, or* Real$^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 1645.1. The lower endpoint row L, upper endpoint row U, endpoint-order ledger O, structural transport row H, and replay row C are all visible before any downstream interval consumer is reached.

Read O as the finite dyadic comparison ledger asserting that the displayed lower endpoint precedes the displayed upper endpoint. This row is part of the carrier inventory, not a derived host-order statement about real endpoints.

Apply H rowwise to transport the endpoint histories and the order ledger together. Then apply C to replay the transported endpoint route through the same finite dyadic interval packet.

The carrier contains no coordinate for host real order, trichotomy, selected representatives, quotient intervals, or Real completeness. Thus lower-before-upper order is preserved exactly by the displayed O, H, C rows and cannot escape to an ambient interval principle. □

Lean4 ✓ `BEDC.Derived.DyadicIntervalArithmeticUp.DyadicIntervalArithmeticCarrier endpoint order transport`

Theorem 1645.4 (Dyadic interval arithmetic enclosure soundness)**.** *Every exported operation of an accepted* DyadicIntervalArithmetic$^\uparrow$ *carrier factors through the displayed dyadic endpoint rows and preserves the containment interfaces of* RationalInterval$^\uparrow$ *and* RealIntervalEnclosure$^\uparrow$*. More precisely, if an accepted packet exports an addition, multiplication, or refinement result, then every downstream* RationalInterval$^\uparrow$*,* MetricBall$^\uparrow$*,* LocatedInterval$^\uparrow$*, or* RealIntervalEnclosure$^\uparrow$ *consumer reaches that result through the finite route*

$$L, U, O \longmapsto +_\Delta, \cdot_\Delta, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a dyadic endpoint enclosure whose rational and real containment rows are the displayed rows in E, with no quotient interval, selected real representative, trichotomy, countable-choice row, open-cover compactness row, or Real$^\uparrow$ *completeness witness inserted into the route.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalArithmeticUp.DyadicIntervalArithmeticCarrier enclosure soundness`

Proof. Start with an accepted carrier $A = (L, U, O, +_\Delta, \cdot_\Delta, R, E, H, C, P, N)$. The only endpoint coordinates are the two DyadicRatCore$^\uparrow$ rows L and U, and the only order coordinate between them is O.

For addition, project $+_\Delta$. Its output lower and upper endpoint rows are formed from the displayed dyadic endpoint additions and outward comparison witnesses. For multiplication, project $\cdot_\Delta$. Its output bounds are chosen from the four displayed dyadic endpoint products by the finite comparison ledger. For refinement, project R, which names the tightening comparisons between endpoint pairs.

The enclosure-comparison row E is the sole handoff to $\mathsf{RationalInterval}^{\uparrow}$ and $\mathsf{RealIntervalEnclosure}^{\uparrow}$. A $\mathsf{MetricBall}^{\uparrow}$ or $\mathsf{LocatedInterval}^{\uparrow}$ reader can consume the same handoff only after the dyadic endpoints and operation rows have been displayed.

Finally H, C, P, N package the route. Componentwise transport and continuation replay preserve the same endpoint and enclosure rows, while provenance records the dependency anchors. Since every exported operation is reached through this route, there is no remaining coordinate from which a quotient interval, selected real representative, trichotomy, countable-choice row, open-cover compactness row, or $\mathsf{Real}^{\uparrow}$ completeness witness could be read. □

Theorem 1645.5 (Dyadic interval arithmetic refinement monotonicity)**.** *Let $A = (L, U, O, +_{\Delta}, \cdot_{\Delta}, R, E, H, C, P, N$ be an accepted* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *carrier. If the refinement row R displays a later lower endpoint above L and a later upper endpoint below U through named* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison rows, then every* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*, or* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *consumer reads a narrower or equal enclosure through the same endpoint arithmetic ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalArithmeticUp.DyadicIntervalArithmeticCarrier refinement monotonicity`

Proof. Project the carrier of Definition 1645.1. The endpoint rows L and U, the endpoint-order ledger O, the operation rows, the refinement row R, and the enclosure row E are all visible before any downstream interval consumer is reached.

Apply Definition 1645.4. Every downstream $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, or $\mathsf{RealIntervalEnclosure}^{\uparrow}$ read already factors through the displayed endpoint route

$$L, U, O \longmapsto +_{\Delta}, \cdot_{\Delta}, R \longmapsto E \longmapsto H, C, P, N.$$

The refinement row only strengthens the containment data. Moving the lower endpoint upward and the upper endpoint downward through displayed dyadic comparison rows leaves the old enclosure as an outer bound and presents the later pair as a narrower or equal bound.

Componentwise hsame transport and displayed Cont replay preserve the same endpoint arithmetic ledger and refinement comparisons. Therefore every listed consumer reads the refined enclosure through the original finite packet, with no quotient interval, selected real representative, trichotomy, countable choice, open-cover compactness, or Real completeness claim inserted. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ packet is generated from lower and upper $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, an endpoint-order ledger, displayed addition and multiplication rows, a refinement row, an enclosure-comparison row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1645.1, Definition 1645.2, and Definition 1645.4 seed the finite dyadic endpoint interval-operation surface over DyadicRatCore, RationalInterval, MetricBall, LocatedInterval, and RealIntervalEnclosure consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalArithmetic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient interval semantics, selected real representatives, trichotomy, countable choice, open-cover compactness, Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint order stability, operation enclosure soundness, refinement monotonicity, classifier transport, and ledger non-escape proofs for BEDC.Derived.DyadicIntervalArithmeticUp.

Chapter 1646

A Concrete Naming Certificate for $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ packet names the finite tolerance bracket in which real metric distance, locatedness requests, and real equality boundaries are read from the same displayed route. It sits between the metric distance packet of chapter 1638, the locatedness request surface of chapter 1627, the equality boundary of chapter 1636, the finite-window and regular-readback rows of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The bracket gives downstream Real and Completion consumers one displayed route for tolerance-sensitive reads; it does not identify quotient streams, select limits, invoke trichotomy, use countable choice, or assert ambient Real completeness.

Definition 1646.1 (Real metric locatedness bracket carrier)**.** *A* $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, L, E, S, R, D, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *distance row whose public read supplies the displayed distance boundary.*
2. *L is the* $\mathsf{LocatednessModulus}^{\uparrow}$ *tolerance-request row that records the finite locatedness budget attached to the distance read.*
3. *E is the* $\mathsf{RealEquality}^{\uparrow}$ *boundary row that records when the same tolerance bracket is read as an equality boundary.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened from the real-name seals before M, L, or E is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the windows in S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, comparison, and refinement rows used by the three sibling packets.*
7. *H records componentwise* hsame *transport for the metric row, locatedness row, equality row, stream windows, regular readback, dyadic ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through S, R, D, then through M, L, E as the consumer asks for distance, locatedness, or equality data.*

9. *P is the* Pkg *provenance over* $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealMetricLocatednessBracket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, E, S, R, D, H, C, P, N$. The carrier has no coordinate for host real equality, quotient streams, selected limits, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealMetricLocatednessBracketUp.RealMetricLocatednessBracketTasteGate single carrier alignment`

Theorem 1646.2 (Real metric locatedness bracket NameCert obligations)**.** *For every accepted* $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ *carrier $B = (M, L, E, S, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealMetricLocatednessBracke}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real metric row, locatedness request row, real equality boundary row, stream windows, regular readback, dyadic tolerance ledger, transport, replay, provenance, and local naming row;*
2. *distance-to-tolerance compatibility, requiring every* $\mathsf{RealMetric}^{\uparrow}$ *read from M to expose the same S, R, D route before a locatedness request in L is consumed;*
3. *tolerance-to-equality compatibility, requiring every equality-boundary read in E to use the displayed dyadic budget already carried by L and D;*
4. *window transport, requiring H and C to preserve the same finite bracket under componentwise transport of the metric, locatedness, equality, stream, readback, and dyadic rows;*
5. *non-escape, excluding host real equality, quotient streams, selected limits, trichotomy, countable choice, and ambient Real completeness from the exported certificate.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealMetricLocatednessBracketUp.RealMetricLocatednessBracketCarrier non escape`

Proof. Carrier admission is projection from Definition 1646.1. The only visible rows are $M, L, E, S, R, D, H, C, P, N$ so every obligation must be read through the same finite packet.

Distance-to-tolerance compatibility follows by applying the real-metric route of Definition 1638.3. A public distance read from M already factors through stream windows, regular readback, and the dyadic tolerance ledger, which are precisely the rows S, R, D carried by the bracket before L is consumed.

Tolerance-to-equality compatibility uses Definition 1627.3 and Definition 1636.2. The locatedness request keeps the tolerance budget attached to the same finite windows and readbacks, and the equality classifier remains stable under refinements that factor through the same regular-rational and dyadic rows.

Window transport is then supplied by H and C. Componentwise hsame transport moves M, L, E, S, R, D together, while displayed replay records that distance, locatedness, and equality consumers read the same bracket route rather than three unrelated certificates.

The exclusions follow from the carrier inventory. Since B contains no row for host real equality, quotient streams, selected limits, trichotomy, countable choice, or ambient Real completeness, none of those principles can be exported by the local $\mathsf{NameCert}_{\mathsf{RealMetricLocatednessBracket}^{\uparrow}}$ surface. □

Theorem 1646.3 (Real metric locatedness bracket window transport)**.** *Every finite consumer of an accepted* RealMetricLocatednessBracket$^{\uparrow}$ *carrier transports distance, locatedness, and equality reads through the common route*

$$S, R, D \longmapsto M, L, E \longmapsto H, C, P, N.$$

Consequently a Real, metric, equality, or completion-facing consumer receives one displayed tolerance bracket, not separate host comparisons of completed real values.

Lean4 ✓ `BEDC.Derived.RealMetricLocatednessBracketUp.RealMetricLocatednessBracketCarrier window transport`

Proof. Let $B = (M, L, E, S, R, D, H, C, P, N)$ be accepted. The stream row S, regular readback R, and dyadic ledger D are the only window and tolerance coordinates in the carrier.

The metric read is therefore transported through S, R, D before M is consumed. This is the same finite route used by the RealMetric$^{\uparrow}$ distance handoff.

The locatedness read is transported through the same rows because the LocatednessModulus$^{\uparrow}$ request has no second tolerance coordinate inside B. Its request is replayed through the displayed finite windows and dyadic refinements already present in the bracket.

The equality read is transported through the same route because E may read only the regular-rational and dyadic classifier rows exposed in the carrier. The structural rows H, C, P, N package this common route, so the consumer cannot split it into unrelated host comparisons. □

Theorem 1646.4 (Real metric locatedness bracket non-escape)**.** *For every accepted* RealMetricLocatednessBracket$^{\uparrow}$ *carrier* $B = (M, L, E, S, R, D, H, C, P, N)$*, every Real-facing read factors through the displayed metric, locatedness, equality, stream-window, regular-readback, dyadic-tolerance, transport, replay, provenance, and local naming rows. It cannot export host real equality, quotient streams, selected limits, trichotomy, countable choice, or ambient Real completeness.*

Proof. Read the Real-facing part of Definition 1646.1. The only analytic rows are the metric distance row M, the locatedness request row L, the equality boundary row E, the finite windows S, the regular readback R, and the dyadic tolerance ledger D.

Transport and replay do not add another analytic source. The rows H and C move the same listed data through componentwise equality and displayed continuation, while P and N record package provenance and local naming.

It remains to rule out non-displayed Real data. The carrier has no field for host real equality, quotient streams, selected limits, trichotomy, countable choice, or ambient Real completeness; every Real-facing read is therefore a projection from B. □

Lean4 ✓ `BEDC.Derived.RealMetricLocatednessBracketUp.RealMetricLocatednessBracketNonEscape`

Closure status for NameCert$_{\text{RealMetricLocatednessBracket}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RealMetricLocatednessBracket}^{\uparrow}}$, seed)

Scope closed: Definition 1646.1, Definition 1646.2, Definition 1646.3, and Definition 1646.4 seed the finite bracket joining real metric distance, locatedness requests, and real equality boundaries.

Formal verification: FormalStatus(NameCert$_{\text{RealMetricLocatednessBracket}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, distance-to-tolerance compatibility, tolerance-to-equality compatibility, window transport, and non-escape proofs for BEDC.Derived.RealMetricLocatednessBracketUp.

Constructive story: A RealMetricLocatednessBracket$^{\uparrow}$ packet is generated from a RealMetric$^{\uparrow}$ distance row, a LocatednessModulus$^{\uparrow}$ tolerance request, a RealEquality$^{\uparrow}$ boundary, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, a DyadicRatCore$^{\uparrow}$ tolerance ledger, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 1647

A Concrete Naming Certificate for $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ packet records the finite parity-tail criterion by which two regular rational subsequence rows are fused into a single Real-facing Cauchy route. It sits between the regular rational carrier of chapter 301, the finite schedule discipline of chapter 126, the Real-specific Cauchy consumer of chapter 1590, and the parity-selector interleaving surface of chapter 414. The packet is a parity-subsequence Cauchy certificate; it is not a Real completeness principle, quotient stream equality, selected limit, host real equality, trichotomy, or countable-choice source.

Definition 1647.1 (EvenOddCauchyCriterion carrier)**.** *An* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, one for the even subsequence and one for the odd subsequence.*
2. *S_A and S_B are the two* $\mathsf{StreamName}^{\uparrow}$ *schedules exposing the finite observation windows for the even and odd rows.*
3. *ϵ is the even–odd selector row recording which schedule is inspected at each displayed parity step.*
4. *M is the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger, carrying the common precision request, the two side budgets, and the late-tail threshold used by both parity rows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the rational radii, dyadic comparisons, and triangle-budget cells used to compare same-parity and cross-parity late windows.*
6. *F is the cross-parity tail-fusion row, tying every accepted even late window to an accepted odd late window under the same M, D budget.*
7. *R is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *handoff row, exposed only after $A, B, S_A, S_B, \epsilon, M, D, F$ have been displayed.*
8. *H records componentwise* hsame *transport for the two regular rows, two schedules, selector, shared modulus ledger, dyadic tolerance ledger, fusion row, Real Cauchy handoff, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from a precision request through the selector, the selected schedule, the selected regular row, the shared modulus ledger, the dyadic tolerance ledger, the cross-parity fusion row, and the Real Cauchy handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EvenOddCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionTasteGate single carrier alignment`

Theorem 1647.2 (EvenOddCauchyCriterion NameCert obligations)**.** *For an accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *carrier $K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EvenOddCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two regular rational rows, two stream schedules, parity selector, shared CauchyModulus ledger, DyadicRatCore tolerance ledger, cross-parity fusion row, Real Cauchy handoff, transport, replay, provenance, and local naming row;*
2. *parity schedule compatibility, requiring each even read to use A, S_A and each odd read to use B, S_B under the selector row ϵ;*
3. *shared modulus exactness, requiring both parity sides and every cross-parity late comparison to consume the same displayed M budget;*
4. *dyadic tolerance exactness, requiring same-parity and cross-parity tail comparisons to be recorded in D before the fusion row F can be exposed;*
5. *transport and replay stability, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the ordered route through ϵ, M, D, F, R;*
6. *provenance exhaustion, requiring P and N to name only this displayed route and to export no quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1647.1. The visible packet contains exactly the two regular rows, two schedules, selector, shared modulus ledger, dyadic tolerance ledger, fusion row, Real Cauchy handoff, and the structural rows H, C, P, N.

Parity schedule compatibility is read from ϵ, S_A, S_B. Even requests are routed through A, S_A, while odd requests are routed through B, S_B; the packet has no third schedule or unscheduled subsequence row.

Shared modulus and dyadic tolerance exactness are the roles of M and D. The same M budget is consumed on both parity sides, and D stores the dyadic comparisons needed before F may identify the cross-parity late-tail route.

Transport and replay are structural. Componentwise hsame in H preserves the displayed rows, and Cont replay in C repeats the precision-request route through the selector, selected schedule, shared budget, tolerance ledger, fusion row, and Real Cauchy handoff.

Finally, provenance is exhausted by P and N. Since the carrier has no row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness, the local certificate cannot export any such object. □

Theorem 1647.3 (EvenOddCauchyCriterion parity-tail fusion)**.** *Every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *carrier factors each late-tail comparison through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RealCauchyCriterion}^{\uparrow}.$$

The even side uses A, S_A and the odd side uses B, S_B, while the cross-parity row F records that both sides consume the same M, D budget before the handoff to R. Thus the packet fuses the parity tails without introducing quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionCarrier parity tail fusion`

Proof. Start with an accepted carrier K. The projection of Definition 1647.1 exposes the two regular rows and their two schedules before any tolerance comparison is available.

Use the selector row ϵ to split the finite read. The even case reads through A, S_A, and the odd case reads through B, S_B.

Both cases then pass through the same shared modulus row M. The dyadic tolerance ledger D records the corresponding same-parity and cross-parity comparison cells under that common budget.

The fusion row F is downstream of M, D. It can therefore identify the even and odd late-tail routes only after the shared budget and dyadic comparisons have been displayed.

Finally, the Real Cauchy handoff R receives the fused parity-tail route. The excluded quotient, choice, host-equality, trichotomy, and completeness objects are absent from the carrier inventory, so the fusion cannot export them. □

Theorem 1647.4 (EvenOddCauchyCriterion Real seal route)**.** *For every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N),$$

any Real-facing consumer of the packet must pass through the two $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules, the even–odd selector, the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the cross-parity fusion row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before it can consume the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *handoff R.*

Proof. Project the carrier rows of K. The handoff row R is not an independent Real seal; it is placed after the two regular rows, two schedules, selector, shared modulus, dyadic tolerance ledger, and cross-parity fusion row.

Definition 1647.2 supplies the certificate surface controlling carrier admission, schedule compatibility, shared modulus exactness, dyadic tolerance exactness, transport, replay, and provenance exhaustion.

Definition 1647.3 supplies the parity-tail route. Combining that route with the carrier projection shows that any Real-facing use of R first passes through the displayed even and odd windows and the common M, D comparison ledger.

The rows H, C, P, N preserve and name the same route. Since no quotient stream, selected limit, countable-choice selector, host real equality, trichotomy, or ambient completeness row appears in the carrier, no consumer can bypass the displayed $\mathsf{RealCauchyCriterion}^{\uparrow}$ handoff boundary. □

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionCarrier real seal route`

Theorem 1647.5 (EvenOddCauchyCriterion shared modulus transport)**.** *For every accepted* EvenOddCauchyCriter *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N),$$

the shared M budget and the D tolerance ledger are transported componentwise through the hsame *row H and replayed through the* Cont *row C before either the cross-parity fusion row F or the Real Cauchy handoff row R can be exposed. In particular, no accepted consumer can replace M by separate parity budgets or replace D by an undisplayed tolerance ledger while still reaching F or R.*

Proof. Begin with the carrier inventory of Definition 1647.1. The rows M and D are displayed before F, R, H, C, P, N, and the classifier compares packets only through the listed coordinates.

Definition 1647.2 identifies shared modulus exactness, dyadic tolerance exactness, transport stability, and replay stability as local NameCert obligations. Therefore H must preserve the same M, D cells on the even and odd sides rather than admitting parity-specific substitutes.

Next apply the replay row C. Its route starts from a precision request and passes through the selector, selected schedule, selected regular row, shared modulus ledger, dyadic tolerance ledger, fusion row, and Real Cauchy handoff. Thus replay cannot expose F or R until the transported M, D ledger has already been displayed.

Finally combine the parity-tail factorization of Definition 1647.3 with the Real seal boundary of Definition 1647.4. The only late-tail route reaching F or R is the one carrying the same shared budget and dyadic tolerance ledger through H and C. □

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionCarrier shared modulus transport`

Theorem 1647.6 (EvenOddCauchyCriterion cross-parity handoff non-escape)**.** *For every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N),$$

the cross-parity fusion row F and the Real Cauchy handoff row R can be consumed only after the packet has displayed the two $\mathsf{RegSeqRat}^{\uparrow}$ *rows A, B, the two* $\mathsf{StreamName}^{\uparrow}$ *schedules S_A, S_B, the selector ϵ, the shared modulus ledger M, the dyadic tolerance ledger D, componentwise transport H, displayed replay C, provenance P, and the local* NameCert *row N.*

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionCrossParityHandoffNonEscape`

Proof. The carrier definition Definition 1647.1 places F and R in a finite packet whose earlier coordinates name both regular rows, both schedules, the selector, the common budget, and the tolerance ledger. There is no independent coordinate from which a consumer could obtain a fusion row or a Real handoff.

Definition 1647.2 then fixes the admission and stability surface. Carrier admission accounts for $A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N$, while transport, replay, and provenance exhaustion require the later rows to preserve and name that same displayed route.

The factorization in Definition 1647.3 supplies the cross-parity ordering: F is downstream of the two parity tails under the shared M, D budget. Hence F cannot be consumed before the two regular rows, two schedules, selector, shared modulus, and dyadic ledger have been displayed.

The Real seal route in Definition 1647.4 adds the handoff boundary. Since R is downstream of F and is preserved only through H, C, P, N, a Real-facing consumer reaches R only through the complete displayed NameCert route, and no quotient stream, selected limit, countable-choice selector, host real equality, trichotomy, or ambient completeness object can provide an escape path. □

Theorem 1647.7 (EvenOddCauchyCriterion induction lock)**.** *For every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N),$$

finite induction over the displayed parity schedule ϵ routes every accepted read through exactly one of the two parity rows A, S_A or B, S_B, and then through the same shared ledger cells M, D. Thus an accepted finite read has no route to F or R that bypasses the selected parity schedule, the common $\mathsf{CauchyModulus}^{\uparrow}$ *ledger, or the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyInductionLock`

Proof. The base read is determined by the first displayed selector cell of ϵ. By Definition 1647.1, an even selector exposes A, S_A, and an odd selector exposes B, S_B; no third row is present in the carrier.

For the successor step, assume the accepted prefix has already been routed through the selected parity row and the shared M, D ledger. The next selector cell again chooses either A, S_A or B, S_B, and Definition 1647.2 requires parity schedule compatibility, shared modulus exactness, and dyadic tolerance exactness for that cell.

The induction therefore preserves the invariant that each accepted read first uses one parity schedule and then the common ledger. Definition 1647.3 and Definition 1647.5 place F and R downstream of the same M, D cells, so the completed finite schedule has no escaping handoff route. □

Theorem 1647.8 (EvenOddCauchyCriterion tail merge)**.** *For every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *packet K, once the late even tail and the late odd tail have both displayed the shared M, D cells, the cross-parity fusion row F merges the two parity tails into one BEDC route. The merge uses the same finite NameCert obligations as the carrier surface and does not introduce a quotient stream, selected limit, countable-choice selector, host real equality, trichotomy, or ambient Real completeness row.*

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyCriterionTailMerge`

Proof. Start from the two late tails exposed by Definition 1647.7. Each accepted read has already passed through either A, S_A or B, S_B and then through the shared modulus and tolerance cells.

Definition 1647.2 identifies those common cells as certificate obligations rather than auxiliary choices. Hence the two tails meet at displayed data: the same precision budget in M and the same dyadic comparison ledger in D.

Now apply Definition 1647.3. The row F is exactly the downstream fusion row for those displayed parity tails, and Definition 1647.6 prevents a consumer from reaching F through any undisplayed object. The late even and odd tails therefore merge through F as one finite route. □

Theorem 1647.9 (EvenOddCauchyCriterion single modulus)**.** *For every accepted* $\mathsf{EvenOddCauchyCriterion}^{\uparrow}$ *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N),$$

the RealCauchyCriterion$^{\uparrow}$ *handoff row R consumes one shared modulus route. It cannot split into separate even and odd Cauchy budgets after F has displayed the common M, D ledger.*

Proof. The carrier definition displays only one CauchyModulus$^{\uparrow}$ ledger M and one DyadicRatCore$^{\uparrow}$ tolerance ledger D. The classifier of Definition 1647.1 has no coordinate for parity-specific replacement budgets.

Definition 1647.7 routes each finite parity read through that same ledger, and Definition 1647.8 merges the late tails through F once the shared cells have been displayed.

Finally use the Real seal boundary of Definition 1647.4 together with the transport statement of Definition 1647.5. The handoff row R sits downstream of the fused parity route and of componentwise hsame and Cont replay, so it receives the single shared M, D route rather than two independent parity budgets. □

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchySingleModulus`

Theorem 1647.10 (EvenOddCauchyCriterion Real handoff exhaustion)**.** *Every* RealCauchyCriterion$^{\uparrow}$*-facing handoff from an accepted* EvenOddCauchyCriterion$^{\uparrow}$ *packet*

$$K = (A, B, S_A, S_B, \epsilon, M, D, F, R, H, C, P, N)$$

factors through exactly the displayed route consisting of the selector row ϵ, the selected parity RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *rows, the shared* CauchyModulus$^{\uparrow}$ *ledger M, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, the cross-parity fusion row F, and then the handoff row R. The route is preserved by H, C, P, N and exposes no second RealCauchyCriterion surface, no parity-specific completion datum, no quotient stream equality, no selected limit, no trichotomy, and no countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.EvenOddCauchyCriterionUp.EvenOddCauchyRealHandoffExhaustion`

Proof. First use the induction lock of Definition 1647.7. Every finite read of the parity schedule chooses exactly one displayed side: either A, S_A or B, S_B. The same induction keeps the chosen side tied to the shared M, D cells.

Next apply the tail merge theorem Definition 1647.8. Once both late parity tails have displayed the common modulus and dyadic tolerance rows, the cross-parity fusion row F is the unique downstream merge point available in the packet.

The single-modulus theorem Definition 1647.9 then prevents the handoff row R from splitting the route into separate even and odd Cauchy budgets. The Real seal route of Definition 1647.4 places R after the selector, selected parity row, shared ledger, dyadic tolerance ledger, and fusion row.

Finally the non-escape theorem Definition 1647.6 and the structural rows H, C, P, N show that no other packet coordinate can serve as a RealCauchyCriterion handoff. Thus the displayed route exhausts every Real-facing handoff surface. □

Closure status for NameCert$_{\text{EvenOddCauchyCriterion}^{\uparrow}}$**.**

Constructive story: An EvenOddCauchyCriterion$^{\uparrow}$ packet is generated from two RegSeqRat rows, two StreamName schedules, an even–odd selector, a shared CauchyModulus ledger, a DyadicRatCore tolerance ledger, a cross-parity fusion row, a RealCauchyCriterion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{EvenOddCauchyCriterion}^{\uparrow}}$, obligation)

Scope closed: Definition 1647.1, Definition 1647.2, Definition 1647.3, Definition 1647.4, Definition 1647.5, Definition 1647.6, Definition 1647.7, Definition 1647.8, and Definition 1647.9 seed the parity-tail criterion route through RegSeqRat, StreamName, CauchyModulus, DyadicRatCore, RealCauchyCriterion, and RegularCauchyInterleaving faces.

Formal verification: FormalStatus(NameCert$_{\mathsf{EvenOddCauchyCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, host real equality, trichotomy, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, parity schedule compatibility, shared modulus exactness, dyadic tolerance exactness, tail-fusion factorization, Real Cauchy handoff factorization, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.EvenOddCauchyCriterionUp.

Chapter 1648

A Concrete Naming Certificate for $\mathsf{FractionalPart}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FractionalPart}^{\uparrow}$ packet names the finite integer-part and residual-real interface used before decimal-prefix and comparison consumers may inspect a real name. It sits after the terminal real-name seal of chapter 38, the integer carrier of chapter 31, the public Archimedean bound surface of chapter 430, the integer-part bracket packet of chapter 1479, the decimal-prefix route of chapter 1417, the finite stream windows of chapter 126, the regular-rational readback surface of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet records the residual route as finite BEDC data; it is not a host fractional-part operation, a floor selector, a trichotomy service, a quotient-real equality witness, a selected representative, or an appeal to ambient Real completeness.

Definition 1648.1 (FractionalPart carrier)**.** *A* $\mathsf{FractionalPart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, I, F, B, D, S, G, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name row for the source real, exposed through the real-name seal and the regular-Cauchy readback route before any integer-part read is consumed.*
2. *I is the* $\mathsf{Int}^{\uparrow}$ *integer-part row, read through the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket packet and its displayed integer candidate.*
3. *F is the residual fractional* $\mathsf{Real}^{\uparrow}$ *row, recording the real-name route for the residual obtained from the source row and integer row.*
4. *B is the Archimedean bound and bracket row, binding the source real, the integer candidate, and the residual interval confinement to the same public* $\mathsf{ArchimedeanReal}^{\uparrow}$ *and* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *comparison data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii, slack comparisons, and refinements used to observe both the source row and residual row.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for decimal-prefix and comparison reads of the residual.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads the windows in S as regular rational data before the residual reaches a decimal-prefix consumer.*

8. *H records componentwise* hsame *transport for the source real, integer row, residual row, Archimedean bracket, dyadic tolerance ledger, stream windows, regular readback, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision or decimal-prefix request through R, I, F, B, D, S, G, without inserting another integer search or host arithmetic operation.*

10. *P is the* Pkg *provenance over* $\mathsf{FractionalPart}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FractionalPart}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite residual certificate. It has no coordinate for host floor, trichotomy over arbitrary real names, quotient real equality, selected representatives, countable choice, hidden Archimedean search, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FractionalPartUp.FractionalPartTasteGate single carrier alignment`

Theorem 1648.2 (FractionalPart NameCert obligations)**.** *For every accepted* $\mathsf{FractionalPart}^{\uparrow}$ *carrier $K = (R, I, F, B, D, S, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FractionalPart}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real row, integer-part row, residual real row, Archimedean bracket row, dyadic tolerance row, stream-window row, regular-rational handoff row, componentwise transport, continuation replay, provenance, and local naming row;*

2. *componentwise classifier transport, requiring the accepted status of R, I, F, B, D, S, G to remain stable under the* hsame *transports in H and the displayed* Cont *replay in C;*

3. *residual interval confinement, requiring the residual row F to stay within the displayed unit-bracket route supplied by B and observed through the tolerance ledger D;*

4. *decimal-prefix readback, requiring every decimal-prefix consumer to reach F only through the finite-window row S and the regular-rational handoff row G;*

5. *ledger exhaustion, excluding host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, and ambient Real completeness from the exported certificate.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FractionalPartUp.FractionalPartCarrier namecert obligations`

Proof. Project the accepted packet $K = (R, I, F, B, D, S, G, H, C, P, N)$. The projection exposes exactly the source real row, integer-part row, residual row, Archimedean bracket, dyadic tolerance ledger, stream windows, regular readback, transport row, continuation replay, provenance, and local naming row.

Componentwise classifier transport is read from H and C. The transports move the rows R, I, F, B, D, S, G together, and replay sends each precision request back through the same displayed route rather than through a detached host operation.

Residual interval confinement is the content of the bracket row B together with the tolerance row D. The bracket binds the source real, integer candidate, and residual to the same Archimedean comparison data, while the dyadic ledger records the finite tolerances under which the residual is observed.

Decimal-prefix readback is supplied by S and G. A decimal consumer first opens a finite $\mathsf{StreamName}^{\uparrow}$ window and then reads it through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, so the residual is never presented as an untracked completed real value.

The exclusions follow from the carrier inventory. Since no row stores host floor, trichotomy, quotient equality, selected representatives, countable choice, hidden Archimedean search, or ambient Real completeness, none of those principles can be exported by the local certificate. □

Theorem 1648.3 (FractionalPart Real-Int handoff)**.** *Every consumer of the fractional residual in an accepted* $\mathsf{FractionalPart}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{ArchimedeanReal}^{\uparrow} \longmapsto \mathsf{Int}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Concretely, the consumer receives the displayed source real row R, Archimedean bracket row B, integer-part row I, dyadic tolerance row D, stream-window row S, and regular-rational handoff row G before reading the residual row F. It receives no host floor function, trichotomy service, quotient real, selected representative, or $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.FractionalPartUp.FractionalPartCarrier real int handoff`

Proof. Let a downstream consumer request the fractional residual from an accepted carrier. The source real row R is the first analytic coordinate, so the route begins at the displayed $\mathsf{Real}^{\uparrow}$ name rather than at a host real value.

The Archimedean and integer-part data are then forced by B and I. The bracket row exposes the public Archimedean bound and comparison route, and the integer row is read through the displayed $\mathsf{Int}^{\uparrow}$ candidate supplied by the integer-part packet.

The residual can be inspected only through the dyadic and window rows. The tolerance ledger D records the finite dyadic observations of the residual, S opens the finite stream windows requested by a decimal or comparison consumer, and G converts those windows into the regular-rational readback surface.

Finally H, C, P, N package and name the same route. They transport, replay, source, and certify the displayed rows without adding a floor function, trichotomy service, quotient real, selected representative, or $\mathsf{Real}^{\uparrow}$ completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FractionalPart}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FractionalPart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1648.1, Definition 1648.2, and Definition 1648.3 seed the finite integer-part and residual-real interface for decimal-prefix and Real-comparison consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FractionalPart}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completeness, host floor, trichotomy, quotient equality, selected representatives, hidden Archimedean search, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, componentwise classifier transport, residual interval confinement, decimal-prefix readback, Real-Int handoff, and ledger-exhaustion proofs for BEDC.Derived.FractionalPartUp.

Constructive story: A $\mathsf{FractionalPart}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ source row, an $\mathsf{Int}^{\uparrow}$ integer-part row, a residual $\mathsf{Real}^{\uparrow}$ row, an Archimedean bracket row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1649

A Concrete Naming Certificate for $\mathsf{StableManifold}^{\uparrow}$

Hadamard–Perron stable manifold theory identifies the local invariant surface through a hyperbolic equilibrium by combining a local flow, a charted phase space, and a contraction estimate on the stable direction. The $\mathsf{StableManifold}^{\uparrow}$ packet records only the finite BEDC certificate needed by that local reading: an equilibrium row, a $\mathsf{DynSystem}^{\uparrow}$ flow row, an $\mathsf{Ode}^{\uparrow}$ local-flow row, a $\mathsf{Manifold}^{\uparrow}$ chart row, a $\mathsf{LipschitzMap}^{\uparrow}$ contraction ledger, a stable tangent cone row, componentwise hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. It does not import the full analytic theorem as a primitive.

Definition 1649.1 (StableManifold carrier)**.** *A* $\mathsf{StableManifold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, D, O, M, L, T, H, C, P, N)$$

with the following visible coordinates:

1. *E is the equilibrium point row on the phase carrier;*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *local flow packet whose source and endpoint rows are read in the same phase carrier;*
3. *O is the* $\mathsf{Ode}^{\uparrow}$ *local-flow row used by D as its generator-facing dependency;*
4. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row naming the local coordinate domain;*
5. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *contraction ledger on the stable chart coordinate;*
6. *T is a stable tangent cone row, recording the finite direction labels whose chart coordinates are governed by L;*
7. *H records componentwise* hsame *transports for equilibrium, flow, ODE, chart, contraction, tangent-cone, and endpoint rows;*
8. *C records the* Cont *routes by which local flow endpoints remain readable inside the charted stable packet;*
9. *P records* Pkg *provenance for* $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *dependency rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StableManifold}^{\uparrow}}$ *provenance row.*

The carrier accepts exactly those packets whose flow row fixes the displayed equilibrium, whose chart row admits the tangent-cone labels, and whose contraction ledger is read on the stable coordinate rows named by T.

Lean4 ✓ `BEDC.Derived.StableManifoldUp.StableManifoldTasteGate single carrier alignment`

Theorem 1649.2 (StableManifold NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{StableManifold}^\uparrow}$ *seed surface consists of six finite obligations:*

1. *carrier: the packet of Definition 1649.1;*
2. *classifier: componentwise comparison of the equilibrium, flow, ODE, chart, contraction, tangent-cone, transport, continuation, provenance, and local certificate rows;*
3. *equilibrium stability: the* $\mathsf{DynSystem}^\uparrow$ *flow row classifies the displayed equilibrium with its zero-time endpoint;*
4. *chart compatibility: the* $\mathsf{Manifold}^\uparrow$ *chart row carries every stable tangent-cone coordinate used by the packet;*
5. *contraction ledger: the* $\mathsf{LipschitzMap}^\uparrow$ *row supplies the finite contracting bound on stable chart coordinates;*
6. *ledger policy:* Pkg *and* NameCert *expose only the listed dynamical, ODE, manifold, Lipschitz, transport, continuation, and tangent-cone rows.*

Lean4 ✓ `BEDC.Derived.StableManifoldUp.StableManifoldCarrier namecert obligations`

Proof. Read the carrier coordinates of Definition 1649.1. The flow row is already a finite $\mathsf{DynSystem}^\uparrow$ packet, and chapter 195 identifies its public surface as a phase row, elapsed-time row, endpoint row, ODE dependency row, and displayed Cont ledger.

Read the generator dependency through chapter 193. It supplies the local-flow row and keeps vector-field evaluation, Picard data, and endpoint transport inside the ODE certificate surface.

Read chart and contraction data through chapter 96 and chapter 410. The chart row names the local coordinate domain, while the Lipschitz row supplies the finite bound ledger on the stable coordinate rows. Repacking these displayed rows with componentwise hsame, Cont, Pkg, and the local naming row gives the six advertised obligations. □

Theorem 1649.3 (StableManifold flow invariance)**.** *Let* $S = (E, D, O, M, L, T, H, C, P, N)$ *be an accepted* $\mathsf{StableManifold}^\uparrow$ *carrier. Suppose a local* $\mathsf{DynSystem}^\uparrow$ *flow segment in D starts at a point classified with the stable tangent-cone chart rows T and remains inside the chart domain named by M for the displayed time row. Then its endpoint is again classified with the same* $\mathsf{StableManifold}^\uparrow$ *carrier after transporting the equilibrium, flow, ODE, chart, contraction, and tangent-cone coordinates by the stored* hsame *and* Cont *rows.*

Proof. Project the local flow segment from D. By chapter 195, its endpoint is exposed by a displayed Cont route over the same $\mathsf{Manifold}^\uparrow$ phase row and $\mathsf{Ode}^\uparrow$ source row.

Use the $\mathsf{Manifold}^\uparrow$ chart row M to keep the source and endpoint chart coordinates in the same local coordinate surface. The stable tangent-cone row T is finite, so transport of its coordinate labels is componentwise through the stored hsame entries rather than through host equality of tangent spaces.

Finally read the $\mathsf{LipschitzMap}^\uparrow$ ledger L on those transported stable coordinates. The contraction bound is part of the same packet, so the endpoint coordinate repacks with the original

equilibrium, flow, ODE, chart, tangent, provenance, and local naming rows. This is exactly the classifier of Definition 1649.1. □

Lean4 ✓ `BEDC.Derived.StableManifoldUp.StableManifoldFlowInvariance`

Theorem 1649.4 (StableManifold local attraction)**.** *For an accepted* $\mathsf{StableManifold}^{\uparrow}$ *carrier* $S = (E, D, O, M, L, T, H, C, P, N)$*, any finite local orbit segment whose source row is classified with the stable tangent-cone coordinates* T *has every displayed successor endpoint read through the contraction ledger* L *and the same chart row* M*. Consequently the segment's endpoint-distance rows form a finite contraction ledger toward the equilibrium row* E *inside the local* $\mathsf{NameCert}_{\mathsf{StableManifold}^{\uparrow}}$ *surface.*

This is a local attraction certificate: it records the finite BEDC rows that make attraction readable, not a selected infinite trajectory or a global basin classification.

Lean4 ✓ `BEDC.Derived.StableManifoldUp.StableManifoldLocalAttraction`

Proof. Proceed along the finite orbit ledger. Each successor row is a $\mathsf{DynSystem}^{\uparrow}$ flow segment, so chapter 195 exposes its endpoint through the same phase carrier and displayed continuation route.

At each step, use the $\mathsf{Ode}^{\uparrow}$ local-flow source of chapter 193 to keep the generator-facing Picard and endpoint rows on the same local-flow surface. The ManifoldUp chart row then reads the source and endpoint coordinates in the chart named by M.

The $\mathsf{LipschitzMap}^{\uparrow}$ contraction row L supplies the finite comparison between the previous stable coordinate distance and the next one. Because the orbit segment is finite, iterating this ledger only concatenates the displayed comparison rows and their Cont routes. The endpoint-distance rows therefore remain a local contraction ledger toward E inside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StableManifold}^{\uparrow}}$**.**

Constructive story: A stable-manifold packet is generated from an equilibrium row, a DynSystemUp local-flow row, an OdeUp generator-facing local-flow row, a ManifoldUp chart row, a LipschitzMapUp contraction ledger, a finite stable tangent-cone row, componentwise hsame transport, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StableManifold}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed surface names the local finite carrier rows for equilibrium, charted flow, ODE source, stable tangent cone, contraction ledger, transport, continuation, provenance, and local naming, but it does not close the paper axis beyond this carrier presentation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StableManifold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global stable-manifold theorem, smooth foliation, spectral hyperbolicity proof, selected global solutions, or bridge verification.

Upgrade path: Scoped closure requires checked carrier admission, classifier laws, local flow invariance, contraction-ledger exactness, and endpoint attraction for the named stable-manifold packet.

Chapter 1650

A Concrete Naming Certificate for $\mathsf{RegularCauchySeparation}^{\uparrow}$

Following Bishop and Bridges-style regular Cauchy equality, the $\mathsf{RegularCauchySeparation}^{\uparrow}$ packet records how two regular rational names can be read as separated, or not yet separated at a displayed tolerance, before any $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the regular rational rows of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, the modulus surface of chapter 292, and the terminal real-name boundary of chapter 38. The certificate keeps separation evidence as finite NameCert data, not as quotient real equality, host real equality, trichotomy, or ambient completeness.

Definition 1650.1 (Regular Cauchy separation carrier)**.** *A* $\mathsf{RegularCauchySeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
2. *R is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger naming the left and right observation windows inspected by the separation read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and rational comparison rows used at those windows.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *separation-modulus evidence, stating which finite windows in W must be displayed before the tolerance rows in D may be read as equality or separation evidence.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after L, R, W, D, M have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, finite windows, dyadic tolerance rows, separation modulus evidence, Real seal boundary, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from every equality or separation read through $L, R, W, D, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySeparation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySeparation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, host real equality, trichotomy, selected limits, countable choice, ambient Real completeness, or a standard bridge.

Lean4 ✓ `BEDC.Derived.RegularCauchySeparationUp.RegularCauchySeparationTasteGate single carrier alignment`

Theorem 1650.2 (Regular Cauchy separation NameCert obligations)**.** *For an accepted carrier $K = (L, R, W, D, M, E, H, C, P, N)$ of Definition 1650.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySeparation}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows L, R, the* $\mathsf{StreamName}^{\uparrow}$ *window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{CauchyModulus}^{\uparrow}$ *separation-modulus row M, the* $\mathsf{Real}^{\uparrow}$ *boundary row E, and the structural rows H, C, P, N;*
2. *classifier exposure, stating that every equality or separation read of the packet is classified through the displayed route L, R, W, D, M before it can reach E;*
3. *finite-window stability, stating that transport through H and replay through C preserve the same left and right windows named by W;*
4. *dyadic tolerance exactness, stating that the rational comparison rows read from D are the only tolerance rows admitted for the displayed separation read;*
5. *modulus boundary, stating that M is the only row allowed to turn a requested equality or separation tolerance into finite windows of W;*
6. *Real seal non-escape, stating that E is a boundary row consumed only after $L, R, W, D, M, H, C, P, N$ have been retained;*
7. *package provenance and local naming, stating that P names exactly the listed sibling certificates and N names the same finite packet as a local* $\mathsf{NameCert}_{\mathsf{RegularCauchySeparation}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.RegularCauchySeparationUp.RegularCauchySeparationCarrier namecert obligations`

Proof. Project the accepted carrier. The rows L, R, W, D, M, E are the analytic coordinates, while H, C, P, N transport, replay, source, and name the same finite packet.

Read the classifier in dependency order. A separation or equality request first reads the two regular rational names L and R, then the finite observation windows W, then the dyadic tolerance ledger D, and then the modulus row M that binds the requested tolerance to those windows.

The Real-facing row is downstream of that route. Since E is stored only after L, R, W, D, M, a consumer that bypasses one of those rows has not obtained a read of the packet's Real seal boundary.

The structural rows add no hidden equality principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed route, so the listed rows exhaust the seed NameCert surface. □

Theorem 1650.3 (Regular Cauchy separation modulus boundary)**.** *For every accepted* $\mathsf{RegularCauchySeparation}^{\uparrow}$ *packet $K = (L, R, W, D, M, E, H, C, P, N)$, every equality or separation read factors through the finite route*

$$(L, R) \longmapsto W \longmapsto D \longmapsto M$$

before reaching the $\mathsf{Real}^{\uparrow}$ *seal boundary E. Thus the packet exports no equality or separation evidence whose source is a detached Real value, a host equality relation, trichotomy, a selected limit, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySeparationUp.RegularCauchySeparationCarrier modulus boundary`

Proof. Start from Definition 1650.2. The classifier exposure places every accepted equality or separation read on the displayed left and right regular names, their finite window ledger, the dyadic tolerance ledger, and the separation-modulus row.

The modulus row is the last analytic gate before the seal boundary. It converts the requested tolerance into the finite windows already named by W and the dyadic comparison rows already named by D; it does not select a completed real or import host equality.

Finally read E as a boundary row. The package reaches $\mathsf{Real}^{\uparrow}$ only after the finite route through L, R, W, D, M has been displayed, and the transport, continuation, provenance, and local naming rows preserve that route. Therefore every equality or separation read factors through W, D, M before reaching the Real seal boundary. □

Theorem 1650.4 (Regular Cauchy separation zero-distance reflection)**.** *Let $K = (L, R, W, D, M, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySeparation}^{\uparrow}$ *packet. Suppose the separation read uses the same displayed* $\mathsf{StreamName}^{\uparrow}$ *windows for L and R, and suppose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger collapses through the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M to the zero-distance boundary. Then the* $\mathsf{Real}^{\uparrow}$ *seal receives equality evidence only through the finite route*

$$(L, R) \longmapsto W \longmapsto D \longmapsto M \longmapsto E.$$

The packet exports no quotient real equality, host real equality, trichotomy, selected limit, countable choice, or ambient completeness assertion.

Lean4 ✓ `BEDC.Derived.RegularCauchySeparationUp.RegularCauchySeparationCarrier zero distance reflection`

Proof. Project the carrier from Definition 1650.1. The equality-facing read has access only to the two regular rational names L, R, the finite window ledger W, the dyadic tolerance ledger D, the modulus row M, the Real seal boundary E, and the structural rows H, C, P, N.

Use the NameCert obligations of Definition 1650.2. Classifier exposure forces the two displayed StreamName windows to be read before the dyadic tolerance ledger can be interpreted as equality evidence.

Apply the modulus boundary theorem Definition 1650.3. When the tolerance ledger collapses through M to the zero-distance boundary, the collapse is still routed through W, D, M before any Real seal read occurs.

The conclusion is therefore a finite route statement. The carrier has no quotient-real coordinate, no trichotomy coordinate, no selected-limit coordinate, and no ambient-completeness coordinate, while H, C, P, N preserve only the displayed rows. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchySeparation}^{\uparrow}$ consumer that reaches an equality or separation read without the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{CauchyModulus}^{\uparrow}$ separation-modulus row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, cauchymodulus, real: no listed sibling supplies the whole ten-row packet $K = (L, R, W, D, M, E, H, C, P, N)$, because each sibling contributes only regular-name rows, finite windows, dyadic tolerances, modulus evidence, or a Real seal boundary while this chapter binds the separation route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySeparation}^{\uparrow}}$.

Constructive story: A RegularCauchySeparation$^{\uparrow}$ packet is generated from left and right RegSeqRat$^{\uparrow}$ rows, a StreamName$^{\uparrow}$ finite-window ledger, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a CauchyModulus$^{\uparrow}$ separation-modulus row, a Real$^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchySeparation}^{\uparrow}}$, seed)

Scope closed: Definition 1650.1, Definition 1650.2, and Definition 1650.3 seed the certified separation boundary for regular Cauchy names over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyModulus$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchySeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, host real equality, trichotomy, selected limits, countable choice, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, finite-window stability, dyadic tolerance exactness, modulus boundary, Real seal non-escape, package provenance, and local naming for the displayed separation packet.

Chapter 1651

A Concrete Naming Certificate for $\mathsf{DenseUniformEmbedding}^{\uparrow}$

Following the Bourbaki and Kelley treatment of uniform completions, and the Bishop–Bridges constructive use of dense uniform embeddings, the $\mathsf{DenseUniformEmbedding}^{\uparrow}$ packet records the finite interface by which a source uniform or metric object enters a separated completion as a dense uniformly controlled subobject. It binds the dense embedding surface of chapter 1512, the completion target of chapter 1473, the modulus route of chapter 391, the extension consumer of chapter 769, and the separated target boundary of chapter 1612. The packet is a NameCert carrier for the L10 completion route; it is not a quotient completion, a representative selector, or an assertion of ambient completeness.

Definition 1651.1 (DenseUniformEmbedding carrier)**.** *A* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, T, I, Q, M, E, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source uniform or metric data row whose point histories, entourages, and metric observations supply the dense domain.*
2. *T is the target separated completion data row, read through the* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{SeparatedCompletion}^{\uparrow}$ *surfaces before any consumer treats target reads as separated completion reads.*
3. *I is the dense image witness row, recording that every displayed target tolerance window has a source-side image window.*
4. *Q is the uniform-continuity modulus row, read through* $\mathsf{UniformContinuity}^{\uparrow}$*, that transports source closeness to target closeness along the displayed image route.*
5. *M is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *handoff row by which source observations enter the completion-facing target route.*
6. *E is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *consumer row naming the downstream extension interface that may read the dense source and target separated completion together.*
7. *U is the local compatibility row stating that I, Q, M, E are read over the same source row S and target row T.*
8. *H records componentwise* hsame *transport for S, T, I, Q, M, E, U and for the structural rows.*

9. *C records displayed* Cont *replay from source histories through dense image, uniform modulus, completion embedding handoff, extension consumer, and separated completion target.*

10. *P is the* Pkg *provenance over* $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseUniformEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, host equality, countable choice, ambient completeness, or a detached target-completion object.

Lean4 ✓ `BEDC.Derived.DenseUniformEmbeddingUp.DenseUniformEmbeddingTasteGate single carrier alignment`

Theorem 1651.2 (DenseUniformEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *carrier $D = (S, T, I, Q, M, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseUniformEmbedding}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the source uniform or metric row, target separated completion row, dense image witness, uniform-continuity modulus, completion embedding handoff, extension consumer, compatibility row, hsame transport, Cont replay, package provenance, and local naming;*

2. *dense image coverage, requiring each target tolerance request in T to route through the source image witness I before any separated completion read is exported;*

3. *uniform-modulus stability, requiring every source closeness read consumed by the dense route to pass through Q before it reaches the target side;*

4. *completion-embedding compatibility, requiring the handoff row M to be the only route from source observations into the completion-facing target;*

5. *extension-consumer compatibility, requiring every* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *consumer to read the same S, T, I, Q, M route before using E;*

6. *transport and replay preservation, requiring H and C to preserve the ordered route under componentwise* hsame *transport and displayed* Cont *replay;*

7. *ledger non-escape, excluding quotient completions, selected representatives, host equality, countable choice, ambient completeness, and detached target-completion data.*

Lean4 ✓ `BEDC.Derived.DenseUniformEmbeddingUp.DenseUniformEmbeddingNameCertObligations`

Proof. Carrier admission is projection from Definition 1651.1. The tuple lists the source row, target separated completion row, dense image witness, modulus, completion handoff, extension consumer, compatibility row, transport, replay, provenance, and local naming.

Dense image coverage is read from I over the same target row T. A target tolerance request cannot be consumed until the packet has displayed a source-side image window for that request.

Uniform-modulus stability is the row Q. It is the only route by which source closeness is transported to target closeness, so every completion-facing dense read must pass through Q before reaching the target side.

The completion and extension boundaries are the rows M and E. The handoff row M admits source observations into the completion-facing route, and the extension row E is read only after S, T, I, Q, M have been fixed by U.

Finally, H, C, P, N record transport, replay, provenance, and local naming for exactly those rows. Since the carrier contains no quotient, selected representative, host equality, choice, ambient-completeness, or detached-completion coordinate, the listed obligations exhaust the NameCert surface. □

Theorem 1651.3 (DenseUniformEmbedding completion consumer route)**.** *Every completion or extension consumer of an accepted* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto I \longmapsto Q \longmapsto M \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a dense uniformly controlled source image inside the separated completion target, not a quotient-completion point, selected representative, host equality witness, countable-choice schedule, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DenseUniformEmbeddingUp.DenseUniformEmbeddingCarrier completion consumer route`

Proof. Begin with the source row S. The consumer may read only the displayed source uniform or metric histories supplied by the packet.

The dense image row I and uniform-modulus row Q are the next route components. They turn a target tolerance request into a source image window and transport source closeness to target closeness.

The completion embedding handoff M then sends the controlled source observations into the completion-facing boundary. The target row T is reached only after this dense uniform route has been recorded.

Finally the extension consumer row E may read the same route, while H, C, P, N preserve and name it. No excluded quotient, representative, host equality, choice, or ambient completeness object has a coordinate in the carrier, so no consumer can escape the displayed route. □

Theorem 1651.4 (DenseUniformEmbedding ledger non-escape)**.** *For every accepted* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *carrier* $D = (S, T, I, Q, M, E, U, H, C, P, N)$*, every exported read remains inside the rows*

$$S, T, I, Q, M, E, U, H, C, P, N$$

of Definition 1651.1. The packet exports no quotient completion, selected representative, host equality, countable-choice selector, ambient completeness witness, or detached completion object.

Lean4 ✓ `BEDC.Derived.DenseUniformEmbeddingUp.DenseUniformEmbeddingCarrier ledger nonescape`

Proof. The NameCert obligations of Definition 1651.2 enumerate every admissible carrier, coverage, modulus, handoff, extension, transport, replay, provenance, and naming row.

The completion consumer route of Definition 1651.3 covers each downstream read by forcing it through $S, I, Q, M, T, E, H, C, P, N$. The compatibility row U states that these route components are read over the same source and target rows.

There is no remaining carrier coordinate from which a consumer could project a quotient class, selected representative, host equality judgement, choice schedule, ambient-completeness theorem, or detached completion object. Therefore every exported read stays within the displayed finite ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{DenseUniformEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseUniformEmbedding}^{\uparrow}$ packet is generated from a source uniform or metric row, a target UniformCompletion and SeparatedCompletion row, a dense image witness, a UniformContinuity modulus, a CompletionEmbedding handoff, a UniformContinuousExtension consumer row, route compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseUniformEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1651.1, Definition 1651.2, Definition 1651.3, and Definition 1651.4 seed the dense uniform embedding interface for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseUniformEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DenseUniformEmbeddingUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host equality, countable choice, ambient completeness, or detached target-completion objects.

Upgrade path: Obligation closure requires checked carrier admission, dense image coverage, uniform-modulus stability, completion-embedding compatibility, extension-consumer compatibility, transport replay, and ledger non-escape for BEDC.Derived.DenseUniformEmbeddingUp.

Chapter 1652

A Concrete Naming Certificate for $\mathsf{UniformEmbedding}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges on constructive completion, the $\mathsf{UniformEmbedding}^{\uparrow}$ packet names the standard finite interface by which a uniformly controlled injective source enters a completion-facing target. It sits between the uniform-completion surface of chapter 1473, the metric-completion surface of chapter 1506, the source-to-completion handoff of chapter 1512, and the dense specialization of chapter 1651. The packet records the non-dense embedding route used by completion consumers; it is not quotient completion equality, a selected representative, host function equality, countable choice, or ambient completeness.

Definition 1652.1 (Uniform embedding carrier)**.** *A* $\mathsf{UniformEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, T, I, U, J, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source uniform or metric row whose point histories and finite entourage observations are admitted into the embedding route.*
2. *T is the target completion row, read through the* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *surface before any completion-facing consumer inspects target data.*
3. *I is the injective classifier row, recording that two displayed source histories are identified only when the target embedding route identifies their finite observations.*
4. *U is the uniform-modulus row that transports source closeness into target closeness and reflects the target-controlled closeness needed by the embedding interface.*
5. *J is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *handoff row by which source observations enter the completion-facing target route.*
6. *R is the local route-compatibility row stating that I, U, J are read over the same source row S and target completion row T.*
7. *H records componentwise* hsame *transport for S, T, I, U, J, R and for the structural rows.*
8. *C records displayed* Cont *replay from source histories through the injective classifier, uniform modulus, completion-embedding handoff, and target completion row.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host function equality, countable-choice schedules, ambient completeness, or a detached completed target object.

Lean4 ✓ `BEDC.Derived.UniformEmbeddingUp.UniformEmbeddingTasteGate single carrier alignment`

Theorem 1652.2 (Uniform embedding NameCert obligations)**.** *For every accepted* $\mathsf{UniformEmbedding}^{\uparrow}$ *carrier* $E = (S, T, I, U, J, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source row, target completion row, injective classifier, uniform-modulus row, completion-embedding handoff, route-compatibility row, hsame transport, Cont replay, package provenance, and local naming;*
2. *classifier transport, requiring I to be read through the same S, T route under componentwise* hsame *transport and displayed* Cont *replay;*
3. *uniform-modulus stability, requiring every source-to-target closeness read and every reflected embedding-control read to pass through U before target completion data are available;*
4. *embedding injectivity, requiring source identifications exported by the packet to be justified by the displayed injective classifier rather than by host function equality;*
5. *completion-handoff compatibility, requiring the row J to be the only route from source observations into the completion-facing target row T;*
6. *ledger non-escape, excluding quotient completion equality, selected representatives, countable choice, host function equality, ambient completeness, and detached completed target objects.*

Proof. Carrier admission is projection from Definition 1652.1. The tuple lists the source, target completion, injective classifier, uniform modulus, completion handoff, route compatibility, transport, replay, provenance, and local naming rows.

Classifier transport is read from I, H, C. The classifier is not a host equality predicate; it is transported componentwise along the displayed packet and replayed only through the same S, T route.

Uniform-modulus stability is the row U. Every target-facing closeness assertion used by the embedding route passes through this row before the completion handoff is available.

Embedding injectivity and completion compatibility use I and J together. The injective classifier supplies the finite source-identification test, and the handoff row J is the only route from source observations into the target completion row T.

Finally P and N record provenance and local naming for exactly those rows. Since no carrier coordinate stores quotient equality, selected representatives, host function equality, choice, ambient completeness, or detached target objects, the listed obligations exhaust the NameCert surface. □

Theorem 1652.3 (Uniform embedding completion handoff)**.** *Let* $E = (S, T, I, U, J, R, H, C, P, N)$ *be an accepted* $\mathsf{UniformEmbedding}^{\uparrow}$ *carrier. Every completion-facing consumer of E factors through the finite route*

$$S \longmapsto I, U \longmapsto J \longmapsto T \longmapsto R, H, C, P, N.$$

Consequently the consumer can use only the displayed uniform-control and completion-embedding rows. It cannot use quotient completion equality, selected representatives, countable choice, host function equality, or ambient completeness.

Proof. Start with a completion-facing consumer. Projection gives the source row S, but the packet exposes no completed point or representative selector at this stage.

The consumer must next read the classifier I and the uniform-modulus row U. These rows provide the finite injectivity and uniform-control surface needed before target completion data are public.

The completion-embedding handoff J then carries the controlled source observations into the target row T. The compatibility row R states that the classifier, modulus, and handoff are all read over the same S, T boundary.

Transport, replay, provenance, and local naming are exactly H, C, P, N. They preserve the displayed route and add no quotient equality, selected representative, host function equality, countable-choice schedule, or ambient-completeness witness, so every completion-facing consumer factors through the stated handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1652.1, Definition 1652.2, and Definition 1652.3 seed the non-dense uniform embedding interface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host function equality, countable choice, ambient completeness, dense image coverage, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, uniform-modulus stability, embedding injectivity, completion-handoff compatibility, and ledger non-escape for the same packet.

Constructive story: A $\mathsf{UniformEmbedding}^{\uparrow}$ packet is generated from a source uniform or metric row, a target UniformCompletion or MetricCompletion row, an injective classifier, a uniform-modulus row, a CompletionEmbedding handoff, route compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1653

A Concrete Naming Certificate for $\mathsf{DyadicSupremum}^{\uparrow}$

Following Bishop and Bridges-style constructive least-upper-bound reasoning, the $\mathsf{DyadicSupremum}^{\uparrow}$ packet records only the finite dyadic upper-bound and lower-approximation ledgers needed before a located supremum or Real-facing consumer reads the boundary. It sits over the dyadic endpoint core of chapter 291, the interval comparison surface of chapter 306, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the Real seal boundary of chapter 38, and the located-supremum packet of chapter 1427. The certificate is a finite boundary for dyadic supremum reads; it is not an arbitrary least-upper-bound principle, a quotient-cut construction, a selected supremum, trichotomy, host Real equality, countable choice, or Real completeness.

Definition 1653.1 (DyadicSupremum carrier)**.** *A* $\mathsf{DyadicSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, A, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded dyadic family row, recording the displayed family entries and the finite upper-bound ledger through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{DyadicInterval}^{\uparrow}$ *comparison rows.*
2. *A is the lower-approximation ledger, recording finite dyadic witnesses that every demanded lower approximation is met by an inspected family entry.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the finite family windows inspected by the upper-bound and lower-approximation ledgers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic scales at which the bound and approximation comparisons are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, translating the displayed dyadic ledgers into regular rational approximation data without selecting a completed real.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after B, A, W, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the bounded family, lower-approximation ledger, StreamName window, DyadicRatCore tolerance, RegSeqRat handoff, Real seal, continuation, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from every dyadic supremum read through $B, A, W, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicSupremum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary least-upper-bound completeness, quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, or Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicSupremumUp.TasteGate.DyadicSupremumTasteGate single carrier alignment`

Theorem 1653.2 (DyadicSupremum NameCert obligations)**.** *For an accepted carrier $S = (B, A, W, D, R, E, H, C,$ of Definition 1653.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the bounded dyadic family row B, lower-approximation ledger A,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R,* $\mathsf{Real}^{\uparrow}$ *seal boundary E, and the structural rows H, C, P, N;*
2. *upper-bound exposure, stating that every exported upper-bound read of the packet is classified through the displayed dyadic family ledger B and tolerance row D before it reaches the Real seal boundary;*
3. *lower-approximation exposure, stating that every exported lower read factors through a displayed witness in A inside one of the windows named by W;*
4. *finite-window stability, stating that transport through H and replay through C preserve the same inspected windows, dyadic scales, upper-bound rows, and lower-approximation witnesses;*
5. *dyadic tolerance exactness, stating that all supremum comparisons use only the dyadic scales and comparison rows displayed in D;*
6. *regular rational handoff, stating that R is the only row allowed to translate the dyadic ledgers into regular rational approximation data for downstream Real-facing consumers;*
7. *Real seal non-escape, stating that E is a boundary row consumed only after $B, A, W, D, R, H, C, P, N$ have been retained;*
8. *package provenance and local naming, stating that P names exactly the listed sibling certificates and N names the same finite packet as a local* $\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}$.

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.DyadicSupremumUp.DyadicSupremumNameCertObligations`

Proof. Project the accepted carrier. The rows B, A, W, D, R, E are the analytic coordinates, while H, C, P, N transport, replay, source, and name the same finite packet.

Read the classifier in dependency order. An upper-bound request first reads the bounded dyadic family row B and the dyadic tolerance ledger D. A lower-approximation request first reads the witness ledger A, the finite windows W, and the same dyadic tolerance row D.

The regular rational handoff is downstream of those finite ledgers. The row R can only translate the displayed dyadic comparisons into regular rational approximation data; it does not select a completed supremum or import an order-completeness principle.

The Real-facing row is the final boundary. Since E is stored only after B, A, W, D, R, a consumer that bypasses one of those rows has not obtained a read of the packet's Real seal boundary.

The structural rows add no hidden supremum principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed route, so the listed rows exhaust the seed NameCert surface. □

Theorem 1653.3 (DyadicSupremum located-supremum handoff)**.** *For every accepted* $\mathsf{DyadicSupremum}^{\uparrow}$ *packet* $S = (B, A, W, D, R, E, H, C, P, N)$*, every located-supremum or Real-facing read factors through the finite route*

$$B, A \longmapsto W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently any consumer from chapter 291, chapter 306, chapter 126, chapter 301, chapter 38, or chapter 1427 receives only the displayed dyadic upper-bound and lower-approximation ledgers, with no arbitrary least-upper-bound completeness, quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, or Real completeness inserted into the route.

Proof. Start with an accepted carrier $S = (B, A, W, D, R, E, H, C, P, N)$. The only family coordinate is B, the only lower-approximation coordinate is A, and the only inspected windows and comparison scales are W and D.

For an upper-bound read, project B and D. The exported bound is read from the displayed dyadic family ledger and the dyadic comparison rows at the named scales. For a lower-approximation read, project A, W, D. The witness is one of the inspected family entries named by W and compared at a displayed dyadic scale in D.

The row R is the sole handoff to $\mathsf{RegSeqRat}^{\uparrow}$ approximation data, and E is the sole boundary row for $\mathsf{Real}^{\uparrow}$ consumers. A $\mathsf{LocatedSupremum}^{\uparrow}$ reader can consume the same handoff only after the dyadic upper-bound and lower-approximation rows have been displayed.

Finally H, C, P, N package the route. Componentwise transport and continuation replay preserve the same finite ledgers, while provenance records the dependency anchors. Since every exported read is reached through this route, there is no remaining coordinate from which arbitrary least-upper-bound completeness, quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, or Real completeness could be read. □

Lean4 ✓ `BEDC.Derived.DyadicSupremumUp.DyadicSupremumLocatedCutHandoff`

Theorem 1653.4 (DyadicSupremum finite-window exhaustion)**.** *Every accepted* $\mathsf{DyadicSupremum}^{\uparrow}$ *consumer read of the lower-approximation ledger factors through a finite list of dyadic upper-bound and lower-approximation rows before any* $\mathsf{LocatedSupremum}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer sees the handoff.*

Proof. Project the accepted carrier of Definition 1653.1. The only rows available to a lower-approximation read are the bounded dyadic family row B, the lower-approximation ledger A, the finite window row W, and the dyadic tolerance row D.

Use Definition 1653.2. Its obligation surface exposes upper-bound rows through B, D, lower-approximation witnesses through A, W, D, finite-window stability through H, C, dyadic tolerance exactness through D, regular rational handoff through R, Real seal non-escape through E, provenance through P, and local naming through N.

Apply Definition 1653.3. The handoff to $\mathsf{LocatedSupremum}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ appears only after the finite dyadic ledgers have been displayed and routed through R and E.

Thus the consumer read is exhausted by the finite window named in W together with the dyadic rows displayed in B, A, D. The packet introduces no arbitrary least-upper-bound completeness, quotient cut, selected supremum, trichotomy, host Real equality, countable choice, or Real completeness coordinate. □

Lean4 ✓ `BEDC.Derived.DyadicSupremumUp.DyadicSupremumFiniteWindowExhaustion`

Falsifiable prediction: An accepted $\mathsf{DyadicSupremum}^{\uparrow}$ consumer that reaches a located-supremum or Real-facing read without the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: dyadicratcore, dyadicinterval, streamname, regseqrat, real, located-supremum: no listed sibling supplies the whole ten-row packet $S = (B, A, W, D, R, E, H, C, P, N)$, because each sibling contributes dyadic comparison rows, interval rows, finite windows, regular rational handoff data, a Real seal boundary, or a located-supremum consumer while this chapter binds the dyadic upper-bound and lower-approximation route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSupremum}^{\uparrow}$ packet is generated from a bounded dyadic family row, a lower-approximation ledger, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff row, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1653.1, Definition 1653.2, and Definition 1653.3 seed the finite dyadic supremum boundary over DyadicRatCore, DyadicInterval, StreamName, RegSeqRat, Real, and LocatedSupremum consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSupremum}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DyadicSupremumUp.DyadicSupremumLocatedCutHandoff`

Bridge status: none

Not claimed: This chapter does not claim arbitrary least-upper-bound completeness, quotient cuts, selected suprema, trichotomy, host Real equality, countable choice, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, upper-bound exposure, lower-approximation exposure, finite-window stability, dyadic tolerance exactness, regular rational handoff, Real seal non-escape, package provenance, and local naming for the displayed dyadic supremum packet.

Chapter 1654

A Concrete Naming Certificate for $\mathsf{RegularCauchyMetric}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMetric}^{\uparrow}$ packet names the metric-facing dyadic closeness route for two displayed regular rational Cauchy names before the completed $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the dyadic arithmetic core of chapter 291, the common finite windows of chapter 126, the regular rational readback rows of chapter 301, the terminal real-name seal of chapter 38, and the real metric consumer surface of chapter 1638. The packet exposes a finite distance route for regular Cauchy names; it does not identify quotient reals, assert host real equality, choose limits, use trichotomy, or claim completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1654.1 (Regular Cauchy metric carrier)**.** *A* $\mathsf{RegularCauchyMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{rc}} = (R_0, R_1, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
2. R_1 *is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
3. W *is the common* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger naming the paired observation windows inspected from* R_0 *and* R_1*.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius and the rational comparison budget for the two observed windows.*
5. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *distance row naming the displayed dyadic distance bound extracted from the same windows and tolerance ledger.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the regular rows, common windows, dyadic tolerance row, and dyadic distance row have been displayed.*
7. H *records componentwise* hsame *transport for the two regular names, common windows, dyadic tolerance row, dyadic distance row, Real seal row, replay row, provenance row, and local naming row.*
8. C *records displayed* Cont *replay from every metric-facing read through* $R_0, R_1, W, D, Q, E, H, P, N$*.*
9. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, W, D, Q, E, H, C, P, N$. The carrier has no coordinate for quotient real equality, host real equality, selected limits, trichotomy, ambient Real completeness, or a detached completed-real distance.

Lean4 ✓ `BEDC.Derived.RegularCauchyMetricUp.RegularCauchyMetricTasteGate single carrier alignment`

Theorem 1654.2 (Regular Cauchy metric NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMetric}^{\uparrow}$ *carrier $M_{\mathrm{rc}} = (R_0, R_1, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMetric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows R_0, R_1, the common* $\mathsf{StreamName}^{\uparrow}$ *window ledger W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *distance row Q, the* $\mathsf{Real}^{\uparrow}$ *seal boundary E, and the structural rows H, C, P, N;*
2. *common-window exposure, requiring every metric-facing read to inspect the paired windows named by W before any dyadic comparison row is read;*
3. *dyadic tolerance exactness, requiring D to supply the requested radius and comparison budget before the distance bound Q is exposed;*
4. *dyadic distance handoff, requiring Q to be the only displayed distance row handed to the downstream* $\mathsf{RealMetric}^{\uparrow}$ *consumer;*
5. *Real seal boundary, requiring E to be consumed only after the route through R_0, R_1, W, D, Q has been displayed;*
6. *transport and replay stability, requiring H and C to preserve the same two regular names, common windows, dyadic tolerance row, dyadic distance row, Real seal boundary, provenance, and local naming row;*
7. *package provenance and local naming, requiring P and N to name exactly the listed sibling certificates and the same finite packet;*
8. *refusal of quotient real equality, host real equality, selected limits, trichotomy, and Real completeness as NameCert data.*

No NameCert obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyMetricUp.RegularCauchyMetricCarrier namecert obligations`

Proof. Project the accepted carrier. The analytic coordinates are the two regular rational names R_0, R_1, the common window ledger W, the dyadic tolerance row D, the dyadic distance row Q, and the Real seal boundary E; the remaining rows transport, replay, source, and name the same packet.

Read the metric-facing route in dependency order. A distance request first opens the paired finite windows named by W from the two $\mathsf{RegSeqRat}^{\uparrow}$ rows, then reads the dyadic radius and comparison budget in D, and only then exposes the dyadic distance row Q.

The downstream Real-facing boundary is therefore ordered. The row E is reached after the regular names and dyadic rows have been displayed, and the $\mathsf{RealMetric}^{\uparrow}$ consumer receives Q as a finite distance row rather than a host completed-real distance.

Finally apply componentwise transport and replay. The rows H and C preserve the same displayed route, while P and N name exactly the sibling certificate dependencies. Since no carrier coordinate stores quotient equality, host equality, selected limits, trichotomy, or Real completeness, none of those principles can be exported by the local NameCert surface. □

Theorem 1654.3 (Regular Cauchy metric dyadic closeness route)**.** *Every accepted* $\mathsf{RegularCauchyMetric}^{\uparrow}$ *consumer factors a metric-facing closeness read through the finite route*

$$(R_0, R_1) \longmapsto W \longmapsto D \longmapsto Q \longmapsto E$$

before it may hand the read to a $\mathsf{RealMetric}^{\uparrow}$ *surface. Thus the consumer receives regular rational readback, common finite windows, dyadic tolerance data, a dyadic distance bound, and a Real seal boundary; it receives no quotient real equality, host real equality, selected limit, trichotomy, or Real completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMetricUp.RegularCauchyMetricCarrier dyadic closeness route`

Proof. Start with an accepted carrier $M_{\mathrm{rc}} = (R_0, R_1, W, D, Q, E, H, C, P, N)$. The only source rows for the closeness read are the two regular rational names R_0 and R_1, and the only observation route attached to them is the common finite window ledger W.

The dyadic request then passes through D and Q. The tolerance row D names the radius and comparison budget, while the distance row Q names the bound extracted from the same displayed windows; no second distance row is present.

Replay the route through C and transport it through H. The replay reaches the Real seal boundary E only after the regular rows, windows, tolerance row, and distance row have been displayed.

The provenance row P points to the sibling surfaces of chapter 291, chapter 126, chapter 301, chapter 38, and chapter 1638. Together with N, it packages the same finite route and adds no coordinate for quotient equality, host equality, selected limits, trichotomy, or completeness. □

Theorem 1654.4 (Regular Cauchy metric scoped kernel route)**.** *Every accepted* $\mathsf{RegularCauchyMetric}^{\uparrow}$ *read factors through* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and distance rows, the* $\mathsf{Real}^{\uparrow}$ *seal boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before any metric consumer inspects zero-distance or triangle-style data. The scoped kernel route exposes only the displayed regular Cauchy metric packet and no quotient real equality, host real equality, selected limit, trichotomy, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMetricUp.RegularCauchyMetricScopedKernelRoute`

Proof. Project the NameCert obligations of Definition 1654.2. They expose the two $\mathsf{RegSeqRat}^{\uparrow}$ rows, common $\mathsf{StreamName}^{\uparrow}$ windows, dyadic tolerance row, dyadic distance row, Real seal boundary, transport, replay, provenance, and local naming as the whole local surface.

Apply the dyadic closeness route of Definition 1654.3. It forces every metric-facing closeness read through $(R_0, R_1) \longmapsto W \longmapsto D \longmapsto Q \longmapsto E$ before any $\mathsf{RealMetric}^{\uparrow}$ surface is available.

The structural rows H, C, P, N preserve that same route for zero-distance and triangle-style consumers. Since the carrier contains no quotient equality, host equality, selected limit, trichotomy, or completeness coordinate, the scoped kernel route is exactly the displayed finite metric packet. □

Theorem 1654.5 (Regular Cauchy metric separated-zero route)**.** *Let*

$$M_{\mathrm{rc}} = (R_0, R_1, W, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMetric}^{\uparrow}$ *packet. If the dyadic distance row* Q *is routed through the zero-distance comparison at every displayed finite window, then every* $\mathsf{RealMetric}^{\uparrow}$ *zero-distance consumer still factors through* $R_0, R_1, W, D, Q, E, H, C, P, N$*. The consumer obtains classifier-compatible regular Cauchy equality only through those displayed rows, not through quotient real equality, host equality, a selected limit, trichotomy, or ambient Real completeness.*

Proof. Begin with Definition 1654.2. The local surface contains only the two regular rational names, the common finite window, the dyadic tolerance row, the dyadic distance row, the Real seal boundary, and the structural transport, replay, provenance, and naming rows.

Apply Definition 1654.3. It forces the zero-distance comparison to read Q only after R_0, R_1, W, D have been displayed and before the Real-facing seal boundary E is exposed.

Finally use Definition 1654.4. The zero-distance consumer is a metric-facing read of the same scoped kernel, so H, C, P, N preserve and name exactly the displayed packet. The resulting regular Cauchy equality is therefore classifier-compatible along the packet rows and is not quotient real equality or host equality. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMetricUp.RegularCauchyMetricCarrier separated zero route`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMetric}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyMetric}^{\uparrow}}$, seed)

Scope closed: Definition 1654.1, Definition 1654.2, and Definition 1654.3 seed the metric-facing finite dyadic closeness surface for two $\mathsf{RegSeqRat}^{\uparrow}$ rows over common $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and distance rows, $\mathsf{Real}^{\uparrow}$ seal data, and $\mathsf{RealMetric}^{\uparrow}$ consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality, host real equality, selected limits, trichotomy, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, common-window exposure, dyadic tolerance exactness, dyadic distance handoff, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMetricU

Constructive story: A $\mathsf{RegularCauchyMetric}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ rows, common $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and distance rows, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1655

A Concrete Naming Certificate for $\mathsf{MetricTopology}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{MetricTopology}^{\uparrow}$ packet names the metric-generated topology obtained from center-radius ball rows. It sits between the metric carrier of chapter 57, the center-radius ball surface of chapter 1450, the indexed-open topology carrier of chapter 88, and the radius ledgers of chapter 38 and chapter 37. The packet records the public bridge from metric data to topology data as finite NameCert rows; it does not select opens from a host powerset and does not quotient generated opens by ambient equality.

Definition 1655.1 (Metric topology carrier)**.** *A* $\mathsf{MetricTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G_{\mathsf{mtop}} = (M, B, T, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the certified* $\mathsf{Met}^{\uparrow}$ *carrier with its point classifier, distance witness, and displayed metric-law rows.*
2. *B is the finite* $\mathsf{MetricBall}^{\uparrow}$ *basis ledger whose singleton rows are center-radius ball data compatible with the metric carrier M.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier $\mathcal{T}_{d,\mathsf{ball}}$ generated from the finite ball indices of Definition 88.10.*
4. *R records the* $\mathsf{Real}^{\uparrow}$ *strict-comparison and positive-radius ledgers used by membership tests of the form $d(c, x) <_{\mathsf{R}} r$.*
5. *Q records the* $\mathsf{Rat}^{\uparrow}$ *radius ledger and its transport into the displayed positive* $\mathsf{Real}^{\uparrow}$ *radius rows.*
6. *H records componentwise* hsame *transport for metric points, ball centers, radius ledgers, singleton-ball predicates, finite ball lists, indexed-open witnesses, replay rows, provenance, and the local certificate.*
7. *C records displayed* Cont *replay from a public open request through the metric carrier M, the ball ledger B, the real and rational radius ledgers R, Q, and the indexed topology carrier T.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, T, R, Q, H, C, P, N$. The carrier has no coordinate for an ambient powerset of M, quotient equality of opens, hidden open selectors, selected centers outside B, or radius data outside R, Q.

Lean4 (definition) `BEDC.Derived.MetricTopologyUp.TasteGate.MetricTopologyUp`
Lean4 ✓ `BEDC.Derived.MetricTopologyUp.TasteGate.MetricTopologyTasteGate single carrier alignment`

Theorem 1655.2 (Metric topology NameCert obligations)**.** *For every accepted* $\mathsf{MetricTopology}^{\uparrow}$ *carrier* $G_{\mathsf{mtop}} = (M, B, T, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricTopology}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed metric carrier, ball basis ledger, indexed topology carrier, real and rational radius ledgers, transport, replay, provenance, and local naming row;*
2. *classifier transport for metric-basic opens, requiring each singleton ball predicate to move along the point classifier of M through the displayed comparison and radius ledgers;*
3. *singleton-ball coverage, requiring every public singleton metric-ball open of T to be read through a row of B before it is admitted as an indexed open;*
4. *finite intersection closure by ball-list concatenation, requiring the intersection of two finite ball-list opens to use the concatenated finite-list index named by the metric-generated topology source;*
5. *arbitrary-union ledger admission, requiring every union row to present an indexed family of already carried generated opens together with the supplied union witness of T;*
6. *boundary rows for the empty and whole opens, requiring the distinguished boundary indices of T and their visible* BHist *ledger rows;*
7. *ledger non-escape, requiring public topology consumers to read only $M, B, T, R, Q, H, C, P, N$ and never an ambient powerset, quotient-open representative, hidden open selector, or radius row outside the displayed ledgers.*

No NameCert obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.MetricTopologyUp.MetricTopologyCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1655.1. The only available public rows are the metric carrier M, the ball basis ledger B, the indexed topology carrier T, the radius ledgers R, Q, and the structural rows H, C, P, N.

Classifier transport and singleton coverage are the displayed route from $\mathsf{MetricBall}^{\uparrow}$ into $\mathsf{Topology}^{\uparrow}$. The singleton data are the ball rows of B, and Definition 88.11 supplies the indexed-open carrier and its membership transport along the point classifier of M.

Finite intersections are read through finite-list concatenation. The source of Definition 88.12 uses finite lists of ball data as indices, and Definition 88.17 identifies the concatenated list with the carried intersection predicate.

Arbitrary unions and boundary opens are the remaining displayed topology rows. The union witness and the empty and whole indices are part of T, so every public open admitted by the packet has a visible constructor ledger row.

Finally, provenance is exhausted by P and N. Since ambient powersets, quotient-open representatives, hidden open selectors, and detached radius rows are not coordinates of G_{mtop}, no local NameCert obligation can export them. □

Theorem 1655.3 (Metric topology ball-basis scope)**.** *Every public* $\mathsf{MetricTopology}^{\uparrow}$ *open carried by an accepted packet* $G_{\mathsf{mtop}} = (M, B, T, R, Q, H, C, P, N)$ *is read through the displayed* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *rows. More precisely, each public open is witnessed by one of the constructor rows of the metric-generated topology source: a singleton metric ball from* B*, a finite-list intersection of such balls, a binary generated-open meet, an indexed arbitrary union of already carried generated opens, the empty open, or the whole open. Thus every downstream consumer of the metric topology receives only the route*

$$\mathsf{Met}^{\uparrow}, \mathsf{MetricBall}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Rat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The packet exports no host powerset of metric points, quotient open, hidden open selector, or radius datum outside the displayed ledgers.

Lean4 ✓ `BEDC.Derived.MetricTopologyUp.MetricTopologyCarrier ball basis scope`

Proof. Let a public open be carried by G_{mtop}. By Definition 1655.1, the open can be read only through the indexed topology row T generated from the ball basis ledger B.

The constructor analysis is the metric-generated exactness surface of Definition 88.12. Singleton rows come from B; finite intersections are finite-list rows; binary meets, arbitrary unions, and boundary opens are the corresponding constructor rows of T.

Each constructor row preserves the same public route. The metric carrier M supplies points and distance witnesses, the ledgers R, Q supply positive-radius comparison data, H and C transport and replay the route, and P, N record provenance and local naming.

No other coordinate is present in the packet. Therefore a consumer cannot recover a host powerset, quotient open, hidden selector, or detached radius datum from a public $\mathsf{MetricTopology}^{\uparrow}$ open. □

Theorem 1655.4 (Metric topology ball-prebasis route)**.** *Every public* $\mathsf{MetricTopology}^{\uparrow}$ *open read from an accepted carrier* $G_{\mathsf{mtop}} = (M, B, T, R, Q, H, C, P, N)$ *factors first through* $\mathsf{MetricBall}^{\uparrow}$ *center-radius rows, then through finite-meet and indexed-union rows of* $\mathsf{Topology}^{\uparrow}$*, and then through the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *radius ledgers and structural* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. The route exports no host powerset of metric points, quotient-open representative, hidden open selector, detached radius datum, completeness theorem, or compactness theorem.*

Lean4 ✓ `BEDC.Derived.MetricTopologyUp.MetricTopologyBallPrebasisRoute`

Proof. Project the carrier. The only basic-open data available to a public read are the metric carrier M and the center-radius ball ledger B.

Apply Definition 1655.3. Singleton ball rows, finite-list intersections, binary meets, indexed unions, and boundary opens are exactly the constructor rows by which T admits public opens.

The radius rows are read with the ball data. The $\mathsf{Real}^{\uparrow}$ ledger R supplies strict comparison and positive-radius rows, while the $\mathsf{Rat}^{\uparrow}$ ledger Q supplies rational radius data and its transport into the displayed real-radius rows.

Finally H, C, P, N transport, replay, source, and name the same route. Since the packet has no ambient powerset, quotient-open representative, hidden selector, detached radius datum, completeness row, or compactness row, every public open factors through the displayed ball-prebasis route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricTopology}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1655.1, Definition 1655.2, and Definition 1655.3 seed the finite metric-topology bridge from $\mathsf{Met}^{\uparrow}$ and $\mathsf{MetricBall}^{\uparrow}$ to $\mathsf{Topology}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricTopology}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host powersets of metric points, quotient opens, hidden open selectors, completeness, compactness, or continuity theorems.

Upgrade path: Obligation closure requires formal carrier admission, singleton-ball coverage, classifier transport, finite-intersection concatenation, arbitrary-union admission, boundary rows, and ledger non-escape for the same finite packet.

Constructive story: A $\mathsf{MetricTopology}^{\uparrow}$ packet is generated from a MetricSpace carrier row, a MetricBall basis ledger, a Topology indexed-open carrier, RealUp and RatUp radius ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1656

A Concrete Naming Certificate for $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$

Bishop–Bridges constructive analysis treats a real name as usable only when the comparison route from fast modulus data to regular Cauchy readback is visible. The $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ packet records that route as finite BEDC data between the fast dyadic-modulus carrier of chapter 1589, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the completion-facing boundary of chapter 360. The packet is a two-way readback bridge; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1656.1 (Fast regular Cauchy equivalence carrier)**.** *A* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FastCauchyReal}^{\uparrow}$ *fast dyadic-modulus real-name row, including the visible precision schedule that every outgoing read must use.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose displayed observations are exactly the observations read by both the fast route and the regular route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, rounded comparison rows, and endpoint arithmetic for the shared stream windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from S and D without adding a quotient stream or a chosen limit.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the shared route through F, S, D, Q has been displayed.*
6. *H records componentwise* hsame *transport for the fast name, stream windows, dyadic ledger, regular readback, real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from every finite precision request through F, S, D, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$, $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FastRegularCauchyEquivalence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.FastRegularCauchyEquivalenceUp.TasteGate.FastRegularCauchyEquivalenceTasteGate single carrier alignment`

Theorem 1656.2 (Fast regular Cauchy equivalence NameCert obligations)**.** *For every accepted* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ *carrier* $K = (F, S, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FastRegularCauchyEquivalence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the fast dyadic-modulus row F, shared stream-window row S, dyadic tolerance ledger D, regular rational readback Q, real-seal row E, and structural rows H, C, P, N;*
2. *fast-to-regular handoff, requiring each finite precision read of F to pass through the displayed S, D rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q may be consumed;*
3. *regular-to-fast schedule extraction, requiring every regular Cauchy readback request on Q to recover a finite fast schedule in F from the same S, D rows;*
4. $\mathsf{Real}^{\uparrow}$*-seal nonescape, requiring E to remain terminal with respect to F, S, D, Q and forbidding any detached completion read;*
5. *ledger refusal, forbidding quotient stream equality, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and any read not routed through the displayed carrier rows.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FastRegularCauchyEquivalenceUp.FastRegularCauchyEquivalenceNameCert obligations`

Proof. Carrier admission is projection from Definition 1656.1. The tuple displays exactly the fast name, shared stream windows, dyadic tolerance ledger, regular readback, real seal, transport, replay, provenance, and local name.

For the fast-to-regular handoff, read a precision request in F. The request names a finite window in S and a dyadic tolerance row in D; those two rows are then repacked as the regular rational readback Q.

For the regular-to-fast schedule extraction, start with a finite regular Cauchy request on Q. Because Q is stored only as the readback of the same S, D route, the request recovers the stream windows and dyadic budget already admitted by F.

The real seal and refusal clauses follow by inspection of the carrier. The row E appears only after F, S, D, Q, while quotient streams, selected limits, countable choice, host real equality, ambient completeness, and detached completion reads are not coordinates of K. □

Theorem 1656.3 (Fast regular Cauchy two-way route)**.** *Every accepted* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ *carrier* $K = (F, S, D, Q, E, H, C, P, N)$ *proves both finite readback directions:*

1. *the fast dyadic-modulus row F yields the regular rational row Q by reading the shared* $\mathsf{StreamName}^{\uparrow}$ *row S and* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
2. *the regular rational row Q yields the fast schedule of F by reading the same S, D rows and the displayed* hsame *and* Cont *routes.*

Both directions reach the $\mathsf{Real}^{\uparrow}$ *seal only through E and the completion-facing provenance of* $\mathsf{RealCauchyCompletion}^{\uparrow}$*. The route constructs no quotient streams, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden completion object.*

Lean4 ✓ `BEDC.Derived.FastRegularCauchyEquivalenceUp.FastRegularCauchyEquivalenceTwoWayRoute`

Proof. The forward direction begins with the fast row F. Its precision schedule selects finite observations in S and dyadic tolerance rows in D; the regular readback row Q is precisely the repackaging of those displayed rows.

The reverse direction begins with the regular row Q. Since the carrier stores Q only downstream of S, D, every finite regular request traces back to the same stream windows and dyadic tolerances, and these rows determine the fast schedule already named by F.

Transport and replay do not change the route. The rows H and C preserve the componentwise reads through F, S, D, Q, E, while P records the source certificates and N binds the packet locally.

Finally, the $\mathsf{Real}^{\uparrow}$ seal is terminal. A completion-facing consumer may cite the $\mathsf{RealCauchyCompletion}^{\uparrow}$ provenance but receives only the displayed route through E, so no quotient stream, selected limit, countable choice, host real equality, or ambient completeness principle is exported. □

Theorem 1656.4 (Fast regular Cauchy schedule lock)**.** *Every accepted* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ *carrier* $K = (F, S, D, Q, E, H, C, P, N)$ *that serves either a* $\mathsf{FastCauchyReal}^{\uparrow}$ *consumer or a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer must read the same displayed* S, D *rows before* Q *or* E *may be consumed. Thus no consumer may substitute a detached fast schedule, quotient stream, selected limit, or hidden completion representative for the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carried by* K*.*

Lean4 ✓ `BEDC.Derived.FastRegularCauchyEquivalenceUp.FastRegularCauchyEquivalenceRegularScheduleLock`

Proof. Project the carrier of Definition 1656.1. The schedule row S and dyadic ledger D are the only rows connecting the fast dyadic-modulus name F to the regular readback Q.

Use the obligation surface of Definition 1656.2. Its fast-to-regular and regular-to-fast clauses both require reads to pass through the displayed S, D rows before the regular readback or real seal is available.

Apply the two-way route of Definition 1656.3. Since both directions recover their schedules from the same S, D rows and the carrier has no quotient stream, selected limit, detached fast schedule, or hidden completion coordinate, every accepted consumer is locked to the displayed schedule. □

Theorem 1656.5 (Fast regular Cauchy tail-window cofinality)**.** *Every accepted* $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ *carrier* $K = (F, S, D, Q, E, H, C, P, N)$ *serving a completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing tail request must read a cofinal family of displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows whose* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is the same* Q *used by the fast schedule. No detached tail selector, quotient stream, selected limit, or hidden completion representative can replace that cofinal family.*

Lean4 ✓ `BEDC.Derived.FastRegularCauchyEquivalenceUp.FastRegularCauchyEquivalenceTailWindowCofinality`

Proof. Project the rows $F, S, D, Q, E, H, C, P, N$ from Definition 1656.1. A completion-facing or $\mathsf{Real}^{\uparrow}$-facing tail request can reach the seal E only through the displayed fast row, stream windows, dyadic ledger, and regular readback.

Use the fast-to-regular and regular-to-fast obligations in Definition 1656.2. The forward direction forces every fast precision read to pass through S, D before Q is consumed, while the reverse direction recovers the fast schedule from the same S, D rows when the request starts from Q.

Apply the two-way route of Definition 1656.3 together with the schedule lock of Definition 1656.4. Both tail-facing directions therefore identify their finite requests with the same displayed S, D windows, and the readback remains the same row Q used by the fast schedule.

Finally, the structural rows H, C, P, N transport, replay, source, and name only this carrier. Since K has no coordinate for a detached tail selector, quotient stream, selected limit, or hidden completion representative, none can replace the cofinal displayed family. □

Falsifiable prediction: An accepted $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ consumer that obtains a $\mathsf{RegSeqRat}^{\uparrow}$ readback from $\mathsf{FastCauchyReal}^{\uparrow}$ data without the displayed $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows refutes the carrier surface.

Independence witness: fastcauchyreal, regseqrat, streamname, dyadicratcore, real, real-cauchy-completion: no listed sibling supplies the whole nine-row packet $K = (F, S, D, Q, E, H, C, P, N)$, because each sibling contributes only a fast name, regular readback, finite window, dyadic ledger, seal, or completion-facing boundary while this chapter binds both readback directions with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FastRegularCauchyEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{FastRegularCauchyEquivalence}^{\uparrow}$ packet is generated from a $\mathsf{FastCauchyReal}^{\uparrow}$ fast dyadic-modulus row, a shared $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ regular readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastRegularCauchyEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1656.1, Definition 1656.2, Definition 1656.3, Definition 1656.4, and Definition 1656.5 seed the finite two-way readback bridge between fast dyadic-modulus real names and regular Cauchy rational names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastRegularCauchyEquivalence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FastRegularCauchyEquivalenceUp.TasteGate.FastRegularCauchyEquivalenceTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fast-to-regular handoff, regular-to-fast schedule extraction, real-seal nonescape, ledger refusal, transport, replay, provenance, and local naming for the same displayed packet.

Chapter 1657

A Concrete Naming Certificate for $\mathsf{CauchyUniformNet}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Moore–Smith nets, the $\mathsf{CauchyUniformNet}^{\uparrow}$ packet records uniform Cauchy-net evidence as finite BEDC rows before any $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the uniform entourage surface of chapter 339, the directed Cauchy-net boundary of chapter 589, the Cauchy-filter surface of chapter 333, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite uniform-net completion route; it is not a quotient of nets, not a selected limit, not countable choice, not host equality, and not ambient completeness.

Definition 1657.1 (Cauchy uniform net carrier)**.** *A* $\mathsf{CauchyUniformNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, F, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the directed index row, recording the finite preorder and upper-tail witnesses used by the net.*
2. *F is the Cauchy-filter-base trace row, naming the finite tail filters generated from the displayed directed windows.*
3. *U is the uniform entourage and tolerance row, carrying the selected entourage, its symmetric or composed tolerance cells, and the finite request budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the windows of the net that have been requested through I, F, U.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows and tolerances.*
6. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, reachable only after I, F, U, W, R have been consumed.*
7. *H records componentwise* hsame *transport for the directed index row, filter-base trace, uniform tolerance row, finite windows, rational readbacks, seal boundary, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite entourage request through the directed tail, then through F, U, W, R, and finally to the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient net equality, selected limits, countable-choice subnet selection, host equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyUniformNetUp.CauchyUniformNetTasteGate single carrier alignment`

Theorem 1657.2 (Cauchy uniform net NameCert obligations)**.** *For every accepted* $\mathsf{CauchyUniformNet}^{\uparrow}$ *carrier* $Q = (I, F, U, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformNet}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the directed index row, filter-base trace, uniform tolerance row, finite windows, regular rational readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundary, transport, continuation, provenance, and local naming row;*
2. *directed-tail stability, requiring every later-window request to pass through the displayed upper-tail witness in I before it can read F;*
3. *uniform tolerance exactness, requiring the requested entourage and its tolerance cells to be read from U before any window, readback, or seal handoff is exported;*
4. *finite-window transport, requiring H and C to preserve the route through I, F, U, W, R, E under componentwise* hsame *and displayed replay;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring every seal-facing read to pass through F, U, W, R before it reaches E;*
6. *provenance and name exhaustion, requiring P and N to name only this displayed finite route and to export no quotient nets, selected limits, countable choice, host equality, or ambient completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1657.1. The displayed packet contains exactly the directed row, filter-base trace, uniform tolerance row, finite windows, regular rational readbacks, seal boundary, and the structural rows H, C, P, N.

For directed-tail stability, replay a finite request through C. The request first consumes the displayed directed upper-tail witness in I, then reads the filter-base trace F attached to that tail. A finite-window induction over the requested window list shows that every inspected window is still downstream of the same I, F route.

Uniform tolerance exactness is read from U. Each induction step carries the same selected entourage and tolerance cells from U into the next requested window W, and then into the regular rational readback R.

Transport is componentwise. The row H identifies corresponding directed, filter, uniform, window, readback, seal, provenance, and local-name coordinates, while C repeats the same ordered route after transport.

Finally, the seal boundary E appears only after F, U, W, R. Since the carrier has no row for quotient net equality, selected limits, countable-choice subnet selection, host equality, or ambient completeness, the local certificate is exhausted by the listed finite rows. □

Lean4 ✓ `BEDC.Derived.CauchyUniformNetUp.CauchyUniformNetNameCertObligations`

Theorem 1657.3 (Cauchy uniform net filter-real route)**.** *Let $Q = (I, F, U, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyUniformNet}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read of Q factors through the finite route*

$$I, F, U, W, R \longmapsto E \longmapsto H, C, P, N.$$

In particular, the read consumes the Cauchy-filter-base trace F, the uniform entourage and tolerance row U, the finite-window row W, and the regular rational readback row R before it reaches E. The packet exports no quotient net, selected limit, countable-choice selector, host equality, or ambient completeness principle.

Proof. Begin with an accepted carrier Q. Projection from Definition 1657.1 exposes the seal row E only in the same packet as $I, F, U, W, R, H, C, P, N$.

Apply the NameCert obligations of Definition 1657.2. Directed-tail stability forces the request through I and F, while uniform tolerance exactness forces the entourage budget through U before any window or readback is available.

The finite-window transport clause then carries the request through W and R under H and C. Thus the seal-facing read reaches E only after the filter-base trace, uniform tolerance row, finite windows, and regular rational readbacks have all been consumed.

The provenance and name rows P, N name only that route. Since no excluded quotient, selected limit, countable-choice, host-equality, or completeness row appears in the carrier, no $\mathsf{Real}^{\uparrow}$ consumer can bypass F, U, W, R or replace the finite route by an ambient limit principle. □

Lean4 ✓ `BEDC.Derived.CauchyUniformNetUp.CauchyUniformNetCarrier filter real route`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyUniformNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyUniformNet}^{\uparrow}$ packet is generated from a directed index row, Cauchy-filter-base trace, uniform entourage and tolerance row, StreamName finite windows, RegSeqRat readbacks, Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyUniformNet}^{\uparrow}}$, seed)

Scope closed: Definition 1657.1, Definition 1657.2, and Definition 1657.3 seed the finite uniform Cauchy-net route through directed tails, filter-base traces, uniform tolerance rows, StreamName windows, RegSeqRat readbacks, and the downstream Real seal boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyUniformNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient net equality, host limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail stability, uniform tolerance exactness, finite-window transport, Real-seal non-escape, and provenance/name exhaustion for BEDC.Derived.CauchyUniformNetUp.

Chapter 1658

A Concrete Naming Certificate for $\mathsf{Dyadic}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{Dyadic}^{\uparrow}$ interface records the finite dyadic naming surface used by constructive real-name and completion routes. It is the direct handoff object between $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal: every consumer receives displayed finite tolerance, stream-window, regular-sequence, transport, replay, provenance, and naming rows, rather than host real equality, selected binary expansions, trichotomy, or completed quotient data.

Definition 1658.1 (DyadicUp carrier)**.** *A* $\mathsf{Dyadic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, carrying the dyadic rational numerator, denominator, tolerance budget, and arithmetic ledger read by dyadic consumers.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which requested precision windows are opened.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row attached to the same window surface.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the displayed real-name boundary reached after Q, S, R have been read.*
5. *H records componentwise* hsame *transport for the dyadic core, stream window, regular sequence, real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a precision request through Q, S, R and then to the real seal E.*
7. *P is the* Pkg *provenance over* $\mathsf{Dyadic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$ *row.*

The carrier contains no coordinate for selected binary expansions, host real equality, countable choice, trichotomy, quotient-stream representatives, or ambient completeness.

Lean4 (definition) `BEDC.Derived.DyadicUp.DyadicUp`

Theorem 1658.2 (DyadicUp NameCert obligations)**.** *For every accepted* $\mathsf{Dyadic}^{\uparrow}$ *carrier*

$$D = (Q, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic core, stream window, regular sequence, real seal, transport, replay, provenance, and local naming rows;*
2. *dyadic core admission, requiring every tolerance and arithmetic read to pass through* Q*;*
3. *stream and regular-sequence admission, requiring the precision window* S *and regular rational row* R *to be displayed before any real-name consumer reads the dyadic packet;*
4. *real-seal handoff, requiring* E *to expose only the* $\mathsf{Real}^{\uparrow}$ *boundary reached from* Q, S, R*;*
5. *transport and replay stability, requiring* H *and* C *to preserve the same dyadic core, stream, regular sequence, and real-seal rows componentwise;*
6. *nonescape from selected binary expansions, host real equality, countable choice, trichotomy, quotient-stream representatives, and ambient completeness.*

No $\mathsf{Dyadic}^{\uparrow}$ *naming obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 1658.1. The accepted packet displays exactly the dyadic rational core Q, finite stream window S, regular rational row R, real seal E, and the structural rows H, C, P, N.

Dyadic core, stream, and regular-sequence admission follow from the displayed route order. A precision request first reads the dyadic tolerance ledger Q, opens the finite window S, and then reads the regular rational row R attached to that window.

The real-seal handoff is then forced through E. Since E is the only Real-facing coordinate of the carrier, a downstream Real or Completion consumer cannot replace the displayed handoff by host equality, a selected binary expansion, or a quotient-stream representative.

Transport and replay are componentwise. The hsame row H preserves each displayed coordinate, while the Cont row C repeats the same finite path from precision request to real seal.

Finally inspect P and N. Provenance and local naming mention only Q, S, R, E, H, C, so the excluded choice, trichotomy, quotient, host-equality, and completeness data have no projection from the packet. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicNameCertObligations`

Theorem 1658.3 (DyadicUp obligation-closure route)**.** *For every accepted* $\mathsf{Dyadic}^{\uparrow}$ *carrier* $D = (Q, S, R, E, H, C, P, N)$*, the rows* Q, S, R, E, H, C, P, N *listed in Definition 1658.1 and certified by Definition 1658.2 are sufficient to witness the* obligation *surface consumed by downstream* $\mathsf{Dyadic}^{\uparrow}$ *readers. The route binds* Q *to* $\mathsf{DyadicRatCore}^{\uparrow}$*,* S *to* $\mathsf{StreamName}^{\uparrow}$*,* R *to* $\mathsf{RegSeqRat}^{\uparrow}$*,* E *to* $\mathsf{Real}^{\uparrow}$*, and* H, C, P, N *to* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *before any Real-facing or Completion-facing read can inspect the dyadic packet.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUp obligation closure route`

Proof. Start from the carrier and NameCert obligations. The carrier displays exactly the dyadic core, stream window, regular rational readback, Real seal, transport, replay, provenance, and local naming rows, while the obligation theorem states that no local naming obligation is stored outside those rows.

Read the mathematical rows first. The dyadic tolerance ledger Q, the finite stream window S, the regular rational row R, and the terminal Real seal E appear before any Real-facing or Completion-facing consumer can use the packet.

The structural rows H, C, P, N then preserve, replay, source, and name the same route over BHist, hsame, Cont, Pkg, and NameCert. Since every obligation row is displayed before the consumer-facing reads and no excluded analytic or quotient object is a carrier coordinate, the displayed rows witness the obligation-closure surface for $\mathsf{Dyadic}^{\uparrow}$ consumers. □

Theorem 1658.4 (DyadicUp obligation-closure dependency surface)**.** *For every accepted* $\mathsf{Dyadic}^{\uparrow}$ *obligation read, the dependency surface factors through exactly the displayed rows*

$$Q, S, R, E, H, C, P, N.$$

Here Q is read from $\mathsf{DyadicRatCore}^{\uparrow}$*, S from* $\mathsf{StreamName}^{\uparrow}$*, R from* $\mathsf{RegSeqRat}^{\uparrow}$*, and E from* $\mathsf{Real}^{\uparrow}$ *before any Real or Completion consumer can use the dyadic packet. The rows H, C, P, N then transport, replay, source, and locally name that same route. No selected binary expansion, quotient stream, trichotomy, countable choice, host equality, or ambient completeness row is admitted by the dependency surface.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUp obligation closure dependency surface`

Proof. Start with Definition 1658.2. Its finite obligation list admits only the dyadic core, stream window, regular rational sequence, real seal, transport, replay, provenance, and local naming rows of the carrier.

Next apply Definition 1658.6. It forces every Real-facing read to expose Q, then S, then R before the terminal row E becomes visible.

Finally use Definition 1658.5. Since Real and Completion consumers can reach the packet only through that terminal factorization, the remaining rows H, C, P, N merely preserve and name the same finite route, and the excluded binary-expansion, quotient-stream, trichotomy, choice, host-equality, and completeness data have no coordinate from which they can be read. □

Theorem 1658.5 (DyadicUp real-seal factorization)**.** *Let $D = (Q, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{Dyadic}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of D must read the displayed dyadic core Q, the finite stream window S, the regular rational row R, and the terminal real seal E in that order before it can export a Real-facing name.*

Consequently a downstream Real or Completion route cannot introduce selected binary expansions, quotient stream representatives, trichotomy, host real equality, countable choice, or ambient completeness while still claiming to consume the same $\mathsf{Dyadic}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUpRealSealFactorization`

Proof. Project the carrier of Definition 1658.1. The only Real-facing coordinate is E, and it is displayed after the dyadic rational core Q, finite window row S, and regular rational row R.

Apply Definition 1658.2. Its dyadic core, stream, regular-sequence, and real-seal obligations force each precision request through Q, S, R, E before any local name can be read by a downstream Real consumer.

The structural rows H, C, P, N only preserve, replay, account for, and name that same displayed route. Since the excluded analytic and quotient objects are absent from the carrier, they cannot be inserted by a Real-facing consumer without leaving the accepted $\mathsf{Dyadic}^{\uparrow}$ NameCert surface. □

Theorem 1658.6 (DyadicUp StreamName RegSeqRat route)**.** *Let $D = (Q, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{Dyadic}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing* $\mathsf{Dyadic}^{\uparrow}$ *read first opens the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Q, then the* $\mathsf{StreamName}^{\uparrow}$ *finite window S, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is visible. Thus the dyadic route depends on the sibling chapters chapter 291, chapter 126, and chapter 301 as finite packet rows, not as ambient host-real completeness data.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUp streamname regseqrat route`

Proof. Project the carrier of Definition 1658.1. The displayed coordinates place Q, S, and R before the only terminal seal coordinate E.

Apply Definition 1658.5. A Real-facing consumer cannot reach E until the dyadic core, stream window, and regular rational row have all been read in the carrier order.

The sibling chapters provide exactly those finite rows: the tolerance ledger from chapter 291, the finite-window classifier from chapter 126, and the regular rational readback from chapter 301. The structural rows H, C, P, N preserve and name that route, so no selected binary expansion, quotient stream, trichotomy, host real equality, countable choice, or ambient completeness principle is exposed by the handoff. □

Theorem 1658.7 (Dyadic exponent-shift composition)**.** *Two displayed* $\mathsf{Dyadic}^{\uparrow}$ *exponent-shift tolerance ledgers compose only by common precision refinement. Any* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of the composed tolerance row must first read the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, refine them to a common finite precision window, expose the corresponding* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, and only then consume the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicExponentShiftComposition`

Proof. Start with the NameCert obligations of Definition 1658.2. Each exponent-shift read is a displayed dyadic core row together with its stream window, regular sequence row, Real seal boundary, transport, replay, provenance, and local naming.

Use Definition 1658.5 for both ledgers. A Real-facing read of either ledger is unavailable until the dyadic core, finite stream window, and regular rational row have all been displayed.

Now compare the two exponent-shift windows by common precision refinement. The refinement is finite because it is formed from the two displayed $\mathsf{StreamName}^{\uparrow}$ windows and the two displayed dyadic tolerance rows; no quotient stream or selected limit is introduced.

Finally apply Definition 1658.6. The composed tolerance row reaches $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers only through the refined finite route, so it exposes no host quotient, selected limit, trichotomy, countable choice, or ambient Real completeness. □

Theorem 1658.8 (DyadicUp common-exponent normalization)**.** *Let $D_0 = (Q_0, S_0, R_0, E_0, H_0, C_0, P_0, N_0)$ and $D_1 = (Q_1, S_1, R_1, E_1, H_1, C_1, P_1, N_1)$ be accepted* $\mathsf{Dyadic}^{\uparrow}$ *carriers whose dyadic tolerance ledgers are read by a common* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer. Then the two accepted reads admit a finite common exponent window before the consumer inspects either terminal row.*

Concretely, the consumer first projects the dyadic core rows Q_0, Q_1, refines the displayed stream windows S_0, S_1 to one finite precision request, transports the regular rational readbacks R_0, R_1 across the componentwise hsame *rows, and only then exposes E_0, E_1 to the terminal* $\mathsf{Real}^{\uparrow}$ *handoff. The normalization exports the shared finite exponent window; it does not add a selected binary expansion, quotient stream, host equality, trichotomy, countable choice, or ambient Real-completeness witness.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUpCommonExponentNormalization`

Proof. Project the two carriers of Definition 1658.1. Each accepted read exposes its dyadic core row, finite stream window, regular rational readback, terminal real seal, and the structural rows that transport and replay that route.

Apply Definition 1658.7 to the two dyadic core ledgers. The common exponent is obtained by finite precision refinement of the two displayed windows, so the construction remains inside the BHist rows already present in D_0 and D_1.

Use Definition 1658.6 after the refinement. Both regular rational readbacks are attached to the common window before the consumer reaches either $\mathsf{Real}^{\uparrow}$ seal.

Finally apply Definition 1658.5. Since the terminal rows are visible only after the refined finite route, the normalized handoff cannot read any quotient, choice, trichotomy, host-equality, selected-expansion, or ambient-completeness data outside the accepted packets. □

Theorem 1658.9 (Dyadic common-exponent totality for shared finite consumers)**.** *For any two accepted* $\mathsf{Dyadic}^{\uparrow}$ *carriers read by the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer, the displayed finite windows determine a common exponent and tolerance window before either terminal* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The total common window is read from the two dyadic core rows, the two finite stream windows, the two regular rational readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$ *rows.*

The route excludes selected binary expansions, quotient streams, trichotomy, choice, host equality, and ambient Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCommonExponentTotality`

Proof. Project the two accepted carriers through Definition 1658.1. The only rows available to the shared consumer are the two dyadic core ledgers, finite stream windows, regular rational readbacks, terminal seals, transport, replay, provenance, and local naming rows.

Apply dyadic exponent-shift composition from Definition 1658.7. The two displayed exponent-shift ledgers compose by finite precision refinement, so a common exponent and tolerance window is determined before any terminal seal can be read.

Use common-exponent normalization, Definition 1658.8. It places both $\mathsf{StreamName}^{\uparrow}$ windows and both $\mathsf{RegSeqRat}^{\uparrow}$ readbacks over the common finite window and keeps the route inside componentwise hsame transport and displayed Cont replay.

Finally apply the StreamName-RegSeqRat route of Definition 1658.6 and the real-seal factorization of Definition 1658.5. The shared consumer reaches the terminal Real seals only after the finite common window is visible, and no carrier row supplies a selected binary expansion, quotient stream, trichotomy, choice principle, host equality, or ambient Real-completeness witness. □

Theorem 1658.10 (DyadicUp common-window tail envelope)**.** *Let* $D_0 = (Q_0, S_0, R_0, E_0, H_0, C_0, P_0, N_0)$ *and* $D_1 = (Q_1, S_1, R_1, E_1, H_1, C_1, P_1, N_1)$ *be accepted* $\mathsf{Dyadic}^{\uparrow}$ *carriers read by one* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer. Then the two reads determine a shared finite tail-envelope window before either terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible.*

The envelope is read from the two dyadic core ledgers Q_0, Q_1*, the shared finite StreamName window refined from* S_0, S_1*, the two regular rational readbacks* R_0, R_1*, componentwise transport, displayed replay, provenance, and the two local* $\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$ *rows. It exposes no selected binary expansion, quotient stream, trichotomy branch, countable-choice selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUpCommonWindowTailEnvelope`

Proof. Project the two carriers through Definition 1658.1. The only rows available to the shared consumer are the two dyadic core ledgers, stream windows, regular rational readbacks, terminal seals, and structural rows.

Apply Definition 1658.7. The two dyadic ledgers compose by common finite precision refinement, so the consumer obtains a single candidate window before any terminal seal can be read.

Use Definition 1658.8. It places both StreamName windows and both RegSeqRat readbacks over the refined common exponent window, which is exactly the tail-envelope window seen by the shared consumer.

Finally apply Definition 1658.6. The terminal Real seals E_0, E_1 become visible only after the finite window and readback rows are displayed, and the excluded binary-expansion, quotient, trichotomy, choice, host-equality, and completeness data have no coordinate in either accepted carrier. □

Theorem 1658.11 (DyadicUp Real-RegSeqRat common-window readback)**.** *Any two accepted* $\mathsf{Dyadic}^{\uparrow}$ *packets consumed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *route expose their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers through one finite* $\mathsf{StreamName}^{\uparrow}$ *window before either* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is available.*

The readback route first projects the two dyadic core rows, refines their displayed StreamName windows to the common finite window, attaches the two regular rational readback rows to that window, and only then exposes the terminal Real seals. It exports no selected binary expansion, quotient stream, trichotomy branch, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Apply Definition 1658.6 to each accepted packet. A shared $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer must read the dyadic core row, then the StreamName window, then the regular rational readback row before any terminal Real seal can be used.

Use Definition 1658.8. The two dyadic core tolerance ledgers are refined to a common finite exponent window, and the two StreamName rows are transported into that same displayed window.

Now apply Definition 1658.10. It identifies the shared tail-envelope window as the finite route seen by the common consumer before either terminal row is visible.

Thus the RegSeqRat readbacks and Real seals are downstream of one finite StreamName window over the two DyadicRatCore ledgers. Since every step is a displayed carrier projection or finite refinement, no selected binary expansion, quotient stream, trichotomy, countable choice, host equality, or ambient Real completeness data is exposed. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUpRealRegSeqRatCommonWindowReadback`

Theorem 1658.12 (DyadicUp terminal Real-seal boundary)**.** *Let*

$$D = (Q, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{Dyadic}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of* D *factors through exactly the rows* Q, S, R, E, H, C, P, N*, with* Q *read from* $\mathsf{DyadicRatCore}^{\uparrow}$*,* S *from* $\mathsf{StreamName}^{\uparrow}$*,* R *from* $\mathsf{RegSeqRat}^{\uparrow}$*, and* E *from* $\mathsf{Real}^{\uparrow}$*. No selected binary expansion, quotient stream, host equality, trichotomy, countable choice, or ambient Real-completeness row is readable from the terminal handoff.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicUpTerminalRealSealBoundary`

Proof. Project the carrier of Definition 1658.1. It displays the mathematical route rows Q, S, R, E and the structural rows H, C, P, N, and it contains no additional Real-facing or Completion-facing coordinate.

Apply Definition 1658.2. The accepted NameCert surface forces dyadic core admission through Q, stream-window admission through S, regular-sequence admission through R, and real-seal admission through E.

Now use Definition 1658.5 and Definition 1658.6. Together they exhaust the route from the dyadic tolerance ledger through finite windows and regular rational readback before the terminal seal can be read.

The remaining rows H, C, P, N only transport, replay, source, and name that route. Since the excluded binary, quotient, host-equality, trichotomy, choice, and completeness data are absent from every displayed coordinate, no terminal consumer can read them while staying inside the accepted $\mathsf{Dyadic}^{\uparrow}$ packet. □

The $\mathsf{Dyadic}^{\uparrow}$ terminal route consumes the finite dyadic tolerance ledger of chapter 291, the finite window classifier of chapter 126, the regular-rational readback of chapter 301, and the terminal seal of chapter 38. These siblings are the displayed L10 dependency route for the dyadic packet, not external host-real infrastructure.

The dyadic Cauchy interface packets collect finite approximation and tail-envelope certificates for displayed real-name routes before Real and Completion consumers can read them. This interface consumes the DyadicRatCore tolerance ledger of chapter 291, the RegSeqRat regular-sequence handoff of chapter 301, and the RealUp terminal seal of chapter 38.

Chapter 1659

A Concrete Naming Certificate for $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ packet records the finite tail-envelope boundary that lets a regular rational Cauchy name expose dyadic tolerance data before the terminal real seal is read. It sits between the finite stream windows of chapter 126, the regular rational name surface of chapter 301, the dyadic tolerance core of chapter 291, the reindexing discipline of chapter 1568, and the real-name boundary of chapter 38. The packet is a finite envelope around displayed tails; it is not a quotient stream, a selected limit, countable choice, trichotomy, host real equality, or Real completeness.

Definition 1659.1 (DyadicCauchyTailEnvelope carrier)**.** *A* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, T, I, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected tail coordinates of the underlying sequence.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic precision budget for the tail comparison.*
4. *T is the tail-envelope ledger, recording the displayed assertion that all inspected post-bound rows of R remain inside the dyadic budget named by D.*
5. *I is the* $\mathsf{CauchyReindexing}^{\uparrow}$ *route row, recording the finite reindexing data used to move from one requested precision window to the tail window named by S.*
6. *H records componentwise* hsame *transport for the stream window, regular rational readback, dyadic tolerance, tail-envelope ledger, reindexing route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through I, S, R, D, T.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicCauchyTailEnvelopeUp.TasteGate.DyadicCauchyTailEnvelopeTasteGate single carrier alignment`

Theorem 1659.2 (DyadicCauchyTailEnvelope NameCert obligations)**.** *For every accepted* DyadicCauchyTailEnvelo *carrier* $E = (S, R, D, T, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, tail-envelope ledger,* $\mathsf{CauchyReindexing}^{\uparrow}$ *route, transport, replay, provenance, and local naming rows;*
2. *window admission, requiring every tail-envelope read to open the finite stream window* S *before reading regular rational rows;*
3. *tolerance admission, requiring the dyadic budget* D *to be displayed before the tail ledger* T *is consumed;*
4. *reindexing admission, requiring the row* I *to witness the finite precision-to-tail route used by the read;*
5. *tail containment, requiring* T *to compare the displayed rows of* R *inside the dyadic budget named by* D *after the route* I *has been fixed;*
6. *transport and replay stability, requiring* H *and* C *to preserve the same window, readback, tolerance, route, and ledger rows componentwise;*
7. *ledger nonescape from quotient streams, selected limits, countable choice, trichotomy, host real equality, and Real completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1659.1. The accepted packet displays exactly the finite stream window S, regular rational readback R, dyadic tolerance row D, tail ledger T, reindexing route I, and the structural rows H, C, P, N.

Window, tolerance, and reindexing admission follow from the route order. A precision request first fixes the finite reindexing row I, opens the stream window S, reads the regular rational row R, and displays the dyadic budget D before the tail-envelope ledger T can be consumed.

Tail containment is exactly the role of T. It compares only the displayed post-bound rows of R inside the dyadic budget named by D, so the envelope does not introduce a completed quotient stream or a selected limit.

Transport and replay are structural. The hsame row H moves each displayed coordinate componentwise, and the Cont row C repeats the same finite route for later consumers.

The nonescape clause follows because none of the excluded objects is a coordinate of the carrier. The local $\mathsf{NameCert}$ row N can therefore export only the displayed tail-envelope surface. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeNameCertObligations`

Theorem 1659.3 (DyadicCauchyTailEnvelope RegSeqRat Real route)**.** *Every accepted* DyadicCauchyTailEnvelope$^{\uparrow}$ *carrier* $E = (S, R, D, T, I, H, C, P, N)$ *exposes a finite route*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$$

from a precision request to the tail-envelope boundary used by $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers. Any downstream Real or Completion reader receives only the displayed reindexing, finite-window, regular-readback, dyadic-tolerance, and tail-ledger rows, with transport, replay, provenance, and local naming data attached.*

Proof. Start with an accepted carrier $E = (S, R, D, T, I, H, C, P, N)$. Projection gives the finite reindexing route I from the requested precision to the tail window, and then the stream window S opened along that route.

The regular rational row R is read only through this opened window. The dyadic tolerance row D supplies the comparison budget, and the tail-envelope ledger T records the displayed post-bound containment at that budget.

Thus the Real-facing consumer reaches the envelope only through I, S, R, D, T. The remaining rows H, C, P, N transport, replay, source, and name the same finite route, so no quotient stream, selected limit, countable-choice selector, trichotomy datum, host real equality, or Real-completeness principle is inserted into the handoff. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeRegSeqRatRealRoute`

Theorem 1659.4 (DyadicCauchyTailEnvelope common refinement)**.** *Let* $E = (S, R, D, T, I, H, C, P, N)$ *and* $E' = (S', R', D', T', I', H', C', P', N')$ *be accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *carriers serving the same displayed precision request. If the reindexing rows* I *and* I' *are transported through the same displayed* hsame *and* Cont *route, then there are a single finite tail window* S^* *and a single dyadic tolerance row* D^* *through which both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks factor before either* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Equivalently, the paired read has one finite common route

$$I^* \longrightarrow S^* \longrightarrow R, R' \longrightarrow D^* \longrightarrow T, T'$$

whose rows are all displayed in BHist *and named by the two local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}$ *surfaces. The common route exports no quotient stream, selected limit, countable choice, trichotomy, host real equality, or Real completeness.*

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeCommonRefinement`

Proof. Inspect the carrier rows of E and E'. The two accepted packets expose only their finite stream windows, regular rational readbacks, dyadic tolerance rows, tail ledgers, reindexing routes, componentwise transports, replay rows, provenance rows, and local naming rows.

Use the displayed Cont replay to compose the two reindexing routes over the common precision request. Since I and I' are transported through the same hsame route, their comparison is a finite reindexing comparison rather than a choice of a completed stream.

Take S^* to be the common displayed tail window reached by that composed route, and take D^* to be the displayed dyadic tolerance row read after the shared window has been opened. Transport both regular readbacks through the shared hsame rows, so R and R' are read only as rows over S^*.

Finally repack the data as the finite route

$$I^* \longrightarrow S^* \longrightarrow R, R' \longrightarrow D^* \longrightarrow T, T'.$$

Both tail ledgers are consumed before either $\mathsf{Real}^{\uparrow}$ seal boundary is inspected. Because the repacked object uses only carrier coordinates already listed in Definition 1659.1, it cannot expose quotient streams, selected limits, countable-choice selectors, trichotomy data, host real equality, or a Real-completeness principle. □

Theorem 1659.5 (DyadicCauchyTailEnvelope window handoff)**.** *Every accepted* DyadicCauchyTailEnvelope$^{\uparrow}$ *packet* $E = (S, R, D, T, I, H, C, P, N)$ *hands a precision request to any* Real$^{\uparrow}$ *or completion-facing consumer through the ordered route*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T.$$

The row I *is the* CauchyReindexing$^{\uparrow}$ *handoff,* S *is the opened* StreamName$^{\uparrow}$ *window,* R *is the* RegSeqRat$^{\uparrow}$ *readback on that window,* D *is the* DyadicRatCore$^{\uparrow}$ *tolerance row, and* T *is the displayed tail-envelope ledger consumed before the* Real$^{\uparrow}$ *seal is read.*

Proof. Project the carrier of Definition 1659.1. The precision request can reach a tail-envelope consumer only through the displayed rows I, S, R, D, T and the structural rows H, C, P, N.

Apply Definition 1659.3. It orders the Real-facing read from the reindexing row to the stream window, regular rational readback, dyadic tolerance, and tail ledger.

Apply Definition 1659.4 for paired consumers. The same ordering survives common refinement, so the window handoff remains a finite route through CauchyReindexing$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeWindowHandoff`

Theorem 1659.6 (DyadicCauchyTailEnvelope nonescape)**.** *Any* Real$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *consumer of an accepted* DyadicCauchyTailEnvelope$^{\uparrow}$ *packet receives no quotient stream, selected limit, countable choice, trichotomy, host real equality, or Real completeness row. It receives only the displayed* CauchyReindexing$^{\uparrow}$ *route,* StreamName$^{\uparrow}$ *window,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance, tail-envelope ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows.*

Proof. Project the accepted carrier. The packet has coordinates only for $S, R, D, T, I, H, C, P, N$, and none of the excluded objects is a coordinate of Definition 1659.1.

Apply Definition 1659.3. A Real-facing consumer receives the finite route through the reindexing row, stream window, regular readback, dyadic tolerance, and tail ledger, with no additional row inserted before the seal.

For paired reads, apply Definition 1659.4. The common route still uses only displayed carrier coordinates, so no quotient stream, selected limit, countable-choice schedule, trichotomy datum, host real equality, or Real-completeness row can escape to a Real$^{\uparrow}$ or CauchyLimitSeal$^{\uparrow}$ consumer. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeNonescape`

Theorem 1659.7 (DyadicCauchyTailEnvelope regular-seal handoff)**.** *Every accepted* DyadicCauchyTailEnvelope$^{\uparrow}$ *packet* $E = (S, R, D, T, I, H, C, P, N)$ *reaches any* Real$^{\uparrow}$ *or* CauchyLimitSeal$^{\uparrow}$ *consumer only through the ordered finite route*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$$

before the seal row is read. The route presents the reindexing request, finite stream window, regular rational readback, dyadic tolerance row, and tail-envelope ledger through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows, and it exposes no quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 1659.1. The accepted packet exposes only $S, R, D, T, I, H, C, P, N$, so a Real or Completion consumer has no route except the displayed envelope rows.

Apply Definition 1659.3. It orders the consumer-facing read as the finite path from the precision request through I, then S, then R, then D, and finally T before any Real-facing seal is consumed.

For paired tail reads, use Definition 1659.4. The common refinement supplies a shared finite route for the two reads, so neither branch can replace the displayed stream window, regular readback, dyadic budget, or tail ledger by a quotient stream, selected limit, countable-choice selector, trichotomy datum, host real equality, or ambient completeness proof. □

Theorem 1659.8 (DyadicCauchyTailEnvelope budget exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet* $E = (S, R, D, T, I, H, C, P, N)$ *must consume all budget rows* $I, S, R, D, T, H, C, P, N$*. There is no partial route that reads the Real seal after only* S, R, D *or only* I, S, R*; the seal is available only after the reindexing row, stream window, regular rational readback, dyadic budget, tail ledger, transport, replay, provenance, and local naming rows have all been displayed.*

Proof. Project the carrier of Definition 1659.1. Its budget surface is exactly $I, S, R, D, T, H, C, P, N$, with I routing the precision request, S opening the finite window, R reading regular rational data, D naming the dyadic tolerance, and T recording the tail containment ledger.

Apply the RegSeqRat Real route, window handoff, common refinement, and regular-seal handoff. These results force every consumer path to pass through $I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$ before any Real-facing or CauchyLimitSeal-facing boundary is read.

Finally use nonescape and the structural rows H, C, P, N. They transport, replay, source, and locally name the same finite route, so a consumer that stops at S, R, D or at I, S, R has not exhausted the carrier coordinates and has not reached the seal. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeBudgetExhaustion`

Theorem 1659.9 (DyadicCauchyTailEnvelope obligation completeness)**.** *For every accepted* DyadicCauchyTailEnve *packet* $E = (S, R, D, T, I, H, C, P, N)$*, the displayed rows*

$$I,\ S,\ R,\ D,\ T,\ H,\ C,\ P,\ N$$

exhaust carrier admission, window admission, tolerance admission, reindexing admission, tail containment, transport and replay stability, provenance, local naming, and nonescape. Thus every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of the envelope reads the same finite obligation surface before a seal row is available, and no quotient stream, selected limit, countable-choice schedule, trichotomy row, host real equality, or ambient Real-completeness row remains outside the listed packet.*

Proof. Project the carrier of Definition 1659.1. It lists exactly the stream window, regular rational readback, dyadic tolerance, tail ledger, reindexing route, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 1659.2. They assign those rows to carrier admission, window admission, tolerance admission, reindexing admission, tail containment, transport and replay stability, and ledger nonescape.

Use the RegSeqRat Real route and common refinement theorems, Definition 1659.3 and Definition 1659.4. They force the consumer path and paired refinements through the same ordered route $I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$.

Finally apply nonescape and budget exhaustion, Definition 1659.6 and Definition 1659.8. Nonescape excludes every row outside the displayed packet, while budget exhaustion requires the structural

rows H, C, P, N as well as I, S, R, D, T before any Real-facing or CauchyLimitSeal-facing boundary is read. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCauchyTailEnvelopeObligationCompleteness`

Theorem 1659.10 (DyadicCauchyTailEnvelope scoped dependency package)**.** *For every accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet* $E = (S, R, D, T, I, H, C, P, N)$*, the obligation surface displayed in this chapter closes exactly over*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$.

Every $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer must pass through those dependencies, in the order*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T,$$

before it reads the seal. The seal is therefore scoped by the listed dependencies and by no quotient stream, selected limit, countable-choice schedule, trichotomy row, host real equality, or ambient Real-completeness row.

Proof. Start with carrier projection and the NameCert obligations of Definition 1659.1 and Definition 1659.2. They expose exactly the stream window, regular rational readback, dyadic tolerance, tail ledger, reindexing route, transport, replay, provenance, and local naming rows.

Apply the RegSeqRat Real route, common refinement, and window handoff of Definition 1659.3, Definition 1659.4, and Definition 1659.5. These results force each consumer read to pass from the reindexing row through the finite stream window, regular rational row, dyadic budget, and tail ledger before any Real-facing boundary is available.

Finally use nonescape and the regular-seal handoff, Definition 1659.6 and Definition 1659.7. They exclude every row outside the listed dependency package, so a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can inspect the seal only after the displayed BEDC dependencies have already been read. □

Theorem 1659.11 (DyadicCauchyTailEnvelope obligation scope)**.** *Every accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *tail-containment read is scoped by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row before any Real-facing consumer can inspect the tail. Thus the only exported obligation route is the finite path*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T,$$

with structural rows H, C, P, N *attached. The route excludes quotient streams, selected limits, countable choice, trichotomy rows, host real equality, and ambient Real completeness.*

Proof. Start from the common refinement and window handoff, Definition 1659.4 and Definition 1659.5. They force each tail-containment read to open a finite $\mathsf{StreamName}^{\uparrow}$ window and read the $\mathsf{RegSeqRat}^{\uparrow}$ row before the dyadic budget can be consumed.

Next apply the regular-seal handoff of Definition 1659.7. It keeps the $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary behind the ordered route $I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$, so the seal is not available before the tolerance row and tail ledger have both been displayed.

Finally use nonescape and the scoped dependency package, Definition 1659.6 and Definition 1659.10. These results attach the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows to the same route and exclude quotient streams, selected limits, countable choice, trichotomy rows, host real equality, and ambient Real completeness from the obligation scope. □

Theorem 1659.12 (DyadicCauchyTailEnvelope obligation closure)**.** *Every accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet* $E = (S, R, D, T, I, H, C, P, N)$ *carries all rows required for obligation closure: the* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{CauchyReindexing}^{\uparrow}$ *route,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transport, displayed replay,* Pkg *provenance, local* $\mathsf{NameCert}$ *row, nonescape from excluded principles, and budget exhaustion before any Real-facing or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer can inspect the seal.*

Proof. Carrier projection and the NameCert obligations, Definition 1659.1 and Definition 1659.2, list the finite window, readback, tolerance, reindexing, transport, replay, provenance, and local naming rows.

Obligation completeness and the scoped dependency package, Definition 1659.9 and Definition 1659.10, identify those rows as the whole dependency surface over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$.

Finally apply nonescape and budget exhaustion, Definition 1659.6 and Definition 1659.8. Nonescape excludes quotient streams, selected limits, countable choice, trichotomy rows, host real equality, and ambient Real completeness, while budget exhaustion forces every consumer to display $I, S, R, D, T, H, C, P, N$ before any seal read. These clauses give precisely the obligation-closure surface for the accepted envelope. □

Theorem 1659.13 (DyadicCauchyTailEnvelope obligation route)**.** *Every accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet exposes its obligation route in the order*

$$\mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad T,\quad \mathsf{CauchyReindexing}^{\uparrow},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

Equivalently, a consumer first opens the finite stream window, reads the regular rational row, displays the dyadic tolerance, records the tail-envelope ledger, fixes the common reindexing route, and only then consumes transport, replay, provenance, and local naming. The route contains no quotient stream, selected limit, countable-choice schedule, trichotomy row, host real equality, or ambient Real-completeness row.

Proof. Project the carrier and the NameCert obligations. The carrier exposes only $S, R, D, T, I, H, C, P, N$, and the obligation list assigns these rows to window admission, tolerance admission, reindexing admission, tail containment, transport, replay, provenance, and local naming.

Use common refinement and nonescape. Common refinement keeps paired tail reads on one displayed finite route, while nonescape excludes every row outside the carrier inventory.

Finally apply budget exhaustion and the scoped dependency package. They force the structural rows H, C, P, N to be consumed with I, S, R, D, T, so the displayed route exhausts the obligation surface before any Real-facing seal can be read. □

Theorem 1659.14 (DyadicCauchyTailEnvelope public boundary)**.** *The public reader of an accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet receives exactly the finite path*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$$

with structural rows H, C, P, N*, before any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary is available. Thus the public surface is the reindexing route, finite stream window, regular rational readback, dyadic tolerance, tail ledger, transport, replay, provenance, and local naming, and it excludes quotient streams, selected limits, countable-choice schedules, trichotomy rows, host real equality, and ambient Real completeness.*

Proof. Begin with the RegSeqRat Real route and common refinement, Definition 1659.3 and Definition 1659.4. They order each public read through the reindexing row, finite window, and regular rational readback before the dyadic budget is consumed.

Next apply nonescape and budget exhaustion, Definition 1659.6 and Definition 1659.8. These results keep every visible read inside the rows I, S, R, D, T and attach no extra quotient, selected-limit, choice, trichotomy, host-equality, or completeness coordinate.

Finally use Definition 1659.13. It attaches H, C, P, N to the same route and exhausts the public obligation surface before a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumer can inspect a terminal seal. □

Theorem 1659.15 (DyadicCauchyTailEnvelope radius stability)**.** *Let $E = (S, R, D, T, I, H, C, P, N)$ be an accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet. If a displayed tail refinement replaces the stream window S by a later finite window along the same reindexing route I, then the dyadic radius row D remains the controlling tolerance for the refined tail ledger. The refined read still factors through*

$$I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$$

with structural rows H, C, P, N, and no new radius, quotient stream, selected limit, or Real-completeness row is introduced.

Proof. Start with the common refinement theorem. It supplies a single finite tail window and a single dyadic tolerance row for paired reads serving the same displayed precision request.

Apply the window handoff and regular-seal handoff. They keep every refined consumer path behind the ordered route $I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$, so the tolerance row is read before the tail ledger and remains the only dyadic budget available.

Use nonescape and budget exhaustion. Since the carrier has no coordinate for an additional radius row or completed stream object, displayed tail refinement preserves the original dyadic radius control. □

Theorem 1659.16 (DyadicCauchyTailEnvelope Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet reads the terminal seal only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *row and the dyadic tail rows have been displayed. In carrier notation, the consumer path must pass through R, D, T after the finite window and reindexing rows, then through H, C, P, N, before a Real-facing seal is available.*

Proof. Read the RegSeqRat Real route. It orders the consumer path from the reindexing row to the stream window, regular rational readback, dyadic tolerance, and tail ledger.

Apply the regular-seal handoff. The seal is reached only after the route $I \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow T$ has been displayed.

Finally use obligation completeness and the scoped dependency package. They attach transport, replay, provenance, and local naming to the same route, so a $\mathsf{Real}^{\uparrow}$ consumer cannot inspect the terminal seal before the RegSeqRat and dyadic tail rows have already been read. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a tail-envelope ledger, a $\mathsf{CauchyReindexing}^{\uparrow}$ route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1659.1, Definition 1659.2, Definition 1659.3, Definition 1659.4, Definition 1659.5, Definition 1659.6, Definition 1659.7, Definition 1659.8, Definition 1659.9, Definition 1659.10, Definition 1659.11, Definition 1659.13, Definition 1659.15, and Definition 1659.16 bind the cauchy-tail envelope obligation surface to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert through carrier admission, window admission, tolerance admission, reindexing admission, tail containment, common-tail refinement, RegSeqRat-to-Real handoff,

window handoff, transport and replay stability, nonescape, obligation completeness, regular-seal handoff, obligation route, radius stability, Real-seal handoff, and obligation scope.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, trichotomy, host real equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: The next grade requires Lean verification and bridge checking for the displayed carrier admission, window admission, tolerance admission, reindexing admission, tail containment, common-tail refinement, RegSeqRat-to-Real handoff, window handoff, transport and replay stability, nonescape, and regular-seal handoff rows.

Theorem 1659.17 (Dyadic core sibling route)**.** *Every dyadic hub route serving a* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *packet factors through the displayed sibling chain in the following order: chapter 1659, chapter 291, chapter 126, chapter 301, and chapter 38. The exported surface is only the finite tail-envelope read, dyadic tolerance ledger, finite stream window, regular rational readback, and terminal real-name seal, with transport, replay, provenance, and local naming rows attached. It exports no quotient stream, selected host limit, trichotomy, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the hub input chapter 1659. Its carrier exposes the finite stream window, regular rational readback, dyadic tolerance row, tail-envelope ledger, reindexing route, transport, replay, provenance, and local naming rows of an accepted $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ packet.

Next read the sibling sources named by that packet. The $\mathsf{DyadicRatCore}^{\uparrow}$ chapter supplies the tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ supplies the finite window, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal consumed by downstream Real and Completion readers.

The route is therefore exhausted by the five displayed chapter surfaces and the structural rows already present in the packet. Since neither the hub nor the tail-envelope carrier contains a quotient stream, selected host limit, trichotomy row, host real equality row, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, none of those objects can be exported by the dyadic hub route. □

Lean4 ✓ `BEDC.Derived.DyadicUp.DyadicCoreSiblingRoute`

Theorem 1659.18 (Dyadic Cauchy real-route sibling dependency)**.** *Every* $\mathsf{Dyadic}^{\uparrow}$ *Cauchy interface read consumed by Real and Completion routes factors through the dyadic-tail envelope before it reaches the sibling rows named by chapter 301, chapter 38, and chapter 291. The route exposes only the finite tail-envelope packet, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming rows.*

Proof. Start with a dyadic Cauchy read at the hub. The hub enters the route through the dyadic-tail envelope, so the first public packet is the finite tail envelope rather than a host Cauchy stream or a selected limit.

The envelope then reads the dyadic tolerance ledger from chapter 291. That ledger supplies the rational tolerance rows used by the regular readback without exporting quotient stream equality or ambient completeness.

Next the route passes through chapter 301 and then chapter 38. These sibling rows provide the regular rational schedule and terminal real-name seal consumed by downstream Real and Completion readers.

Transport, replay, provenance, and naming preserve exactly that ordered route. Since no coordinate of the carrier stores host real equality, a selected completed limit, trichotomy, or an external completion operator, none of those objects can be read from the dyadic Cauchy interface. □

Theorem 1659.19 (DyadicUp obligation-closure scope)**.** *Every accepted* $\mathsf{Dyadic}^{\uparrow}$ *obligation read is exhausted by the dyadic core ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling route, Real-route sibling dependency, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows. It exports no selected binary expansion, quotient stream, trichotomy, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1658.2. The local NameCert surface already restricts the dyadic packet to the dyadic core, stream window, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Use Definition 1659.12, Definition 1659.17, and Definition 1659.18. These sibling routes add the Cauchy tail-envelope obligation, DyadicRatCore dependency, and Real-facing handoff surface without introducing a second carrier coordinate.

The remaining rows are exactly componentwise hsame, displayed Cont, Pkg, and local NameCert data. Since none of the cited routes contains selected binary expansions, quotient streams, trichotomy, countable choice, host equality, or ambient completeness, the accepted obligation read is closed at the displayed obligation surface. □

Theorem 1659.20 (DyadicUp public consumer boundary)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *or Completion-facing* $\mathsf{Dyadic}^{\uparrow}$ *consumer must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *sibling route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling route, the Real-route sibling dependency, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows before exporting a dyadic public name. The boundary exports no selected binary expansion, quotient stream, trichotomy, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the real-seal factorization of Definition 1658.5. A Real-facing consumer cannot reach the terminal row E without first reading the dyadic core row Q, the finite window S, and the regular rational readback row R.

Next apply the StreamName-RegSeqRat route of Definition 1658.6. It identifies the visible route from Q through S and R as the only finite real-name path available to the consumer.

Use the terminal Real-seal boundary of Definition 1658.12. This pins every $\mathsf{Real}^{\uparrow}$ or Completion-facing read to the rows Q, S, R, E, H, C, P, N and excludes the selected binary, quotient, choice, trichotomy, host-equality, and ambient-completeness data.

Finally apply Definition 1659.19. The Cauchy tail-envelope route, DyadicRatCore sibling route, Real-route sibling dependency, componentwise hsame, displayed Cont, Pkg, and local NameCert rows are already part of the closed obligation surface, so the public consumer receives exactly that displayed package before it may export a dyadic public name. □

Closure status for $\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Dyadic}^{\uparrow}$ packet is generated from a DyadicRatCore ledger, a finite StreamName window, a RegSeqRat readback row, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 1658.1, Definition 1658.2, Definition 1658.5, and Definition 1659.19 exhaust the finite dyadic obligation surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dyadic}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.DyadicUp.DyadicNameCertObligations`

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected binary expansions, trichotomy, host real equality, ambient Real completeness, or a completed bridge.

Upgrade path: Scoped closure requires a public consumer boundary over the same dyadic core, stream window, regular readback, Real seal, sibling routes, transport, replay, provenance, and local naming rows.

The $\mathsf{Hyperspace}^{\uparrow}$ packet reads compact-subset and Hausdorff-distance requests through chapter 129 before any Vietoris or finite-net consumer can inspect the local hyperspace rows.

Every $\mathsf{Hyperspace}^{\uparrow}$ Hausdorff-distance or Vietoris consumer first reads the compact-metric finite-net source of chapter 129 and the metric witness route before inspecting the hyperspace rows.

Hausdorff-distance and Vietoris consumers read $\mathsf{Hyperspace}^{\uparrow}$ through chapter 2086 and chapter 5944 before completion-facing routes consume compact-subset data.

Chapter 1660

A Concrete Naming Certificate for $\mathsf{Hyperspace}^{\uparrow}$

Following Vietoris and Kelley on compact hyperspaces, the $\mathsf{Hyperspace}^{\uparrow}$ packet names the finite BEDC surface for nonempty compact subsets of a displayed compact metric source. It sits over the metric source of chapter 57, the compact finite-net row of chapter 129, the total-boundedness schedule of chapter 127, the completeness boundary of chapter 128, the diameter ledger of chapter 1503, the directed convergence row of chapter 1667, and the uniform Cauchy-net route of chapter 1657. The packet is a finite Hausdorff-distance ledger for compact-subset presentations; it is not a host powerset, not a quotient compactum, not a selected limit object, not open-cover compactness, and not a choice-selected representative.

Definition 1660.1 (Hyperspace carrier)**.** *A* $\mathsf{Hyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$$

with the following displayed rows:

1. *X is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, including its* $\mathsf{Met}^{\uparrow}$ *distance witnesses, finite-net schedules, total-boundedness row, and completeness boundary.*
2. *K_0 and K_1 are compact-subset presentations over X, each given by finite net schedules whose centers and radii are readable only through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *rows.*
3. *N_0 and N_1 are the requested net-resolution ledgers for K_0 and K_1, recording the finite witness lists used to inspect each subset presentation.*
4. *D_0 is the directed Hausdorff-distance coverage ledger from K_0 to K_1: every displayed center of K_0 is matched to a displayed K_1 center within the current tolerance after routing through the source metric row.*
5. *D_1 is the directed Hausdorff-distance coverage ledger from K_1 to K_0 with the same source row, tolerance budget, and finite-read discipline.*
6. *R is the dyadic and* $\mathsf{Real}^{\uparrow}$ *tolerance readback for the two directed ledgers, including the comparison cells that package the maximum of D_0 and D_1 as the finite Hausdorff-distance request.*
7. *H_s records componentwise* hsame *transport for the compact source, the two subset presentations, the two net ledgers, the two directed coverage ledgers, the tolerance readback, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a compact-subset request through $X, K_0, K_1, N_0, N_1, D_0, D_1, R$ and then to the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CauchyUniformNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *M is the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

The carrier has no coordinate for host powersets, quotient compacta, selected limits, open-cover compactness, or choice-selected representatives.

Lean4 (definition) `BEDC.Derived.HyperspaceUp.HyperspaceCarrier`

Theorem 1660.2 (Hausdorff-distance ledger stability)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$, classifier transport preserves the two-sided finite Hausdorff-distance ledger. More precisely, any componentwise* hsame *transport of the displayed source, subset presentations, net ledgers, directed coverage ledgers, and tolerance readback carries the finite route through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, *and* $\mathsf{CauchyUniformNet}^{\uparrow}$ *to a carrier with the same directed K_0-to-K_1 and K_1-to-K_0 coverage obligations.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace hausdorff distance stability`

Proof. Project the accepted carrier of Definition 1660.1. The source row X fixes the compact metric surface, while K_0, K_1, N_0, N_1 identify the finite subset-net data that any Hausdorff-distance read can inspect.

The two directed ledgers are then transported componentwise. The row H_s compares the source, subset presentations, net schedules, directed coverage rows, and tolerance readback, and the replay row C repeats the same route after transport.

The compact-source part of the route remains inside $\mathsf{CompactMetric}^{\uparrow}$. Diameter and tolerance comparisons are read through $\mathsf{MetricDiameter}^{\uparrow}$ and R, and the finite convergence and uniform-net inspections are routed through $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{CauchyUniformNet}^{\uparrow}$.

Thus each transported center match in D_0 and D_1 still cites the same kind of displayed source-distance witness and the same finite tolerance readback. Since the packet contains no host powerset, quotient compactum, selected limit, open-cover compactness, or choice-selected representative, no external coordinate can alter the two-sided ledger. □

Theorem 1660.3 (Hyperspace NameCert obligations)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *compact-source admission for the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source row X and its* $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *and* $\mathsf{CompleteMetric}^{\uparrow}$ *support;*

2. *finite subset-net admission for K_0, K_1, N_0, N_1, requiring every subset presentation and net resolution to be read through the displayed compact-source schedules;*

3. *bidirectional distance coverage, requiring both directed ledgers D_0 and D_1 to be present before the Hausdorff-distance tolerance readback R is public;*

4. *classifier and transport stability through H_s and C, preserving the source, subset-net, coverage, and readback rows under componentwise* hsame *and displayed replay;*

5. *provenance and name exhaustion through P and M, excluding host powersets, quotient compacta, selected limits, open-cover compactness, and choice-selected representatives.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1660.1. The packet exposes a compact metric source X and only the displayed finite rows attached to X.

Finite subset-net admission follows from the same projection. The rows K_0, K_1, N_0, N_1 are the only subset-presentation and resolution coordinates, so every compact-subset inspection passes through them.

Bidirectional coverage is exactly the pair D_0, D_1. A Hausdorff-distance read is not public until both directed ledgers have been consumed and the dyadic or $\mathsf{Real}^{\uparrow}$ tolerance readback R has compared them.

Transport is componentwise through H_s and replayed through C. These rows preserve the compact source, subset nets, directed coverage witnesses, and tolerance readbacks without adding a second classifier surface.

Finally, P and M name precisely the displayed packet. Because the carrier contains no row for host powersets, quotient compacta, selected limits, open-cover compactness, or choice-selected representatives, the five listed rows exhaust the local naming obligations. □

Theorem 1660.4 (Hyperspace Hausdorff-obligation surface)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, any public Hausdorff-distance obligation must expose compact-source admission through* X*, finite subset-net schedules through* K_0, K_1, N_0, N_1*, both directed ledgers* D_0, D_1*, tolerance readback through* R*, componentwise* hsame *transport through* H_s*, displayed* Cont *replay through* C*, provenance through* P*, and the local name through* M *before a consumer reads a symmetric distance.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHausdorffObligationSurface`

Proof. Start with the carrier projection in Definition 1660.1. It lists exactly the compact source, the two subset presentations, the two finite net schedules, the directed cover rows, the tolerance readback, transport, replay, provenance, and local naming rows.

Use Definition 1660.3. Its carrier-admission, finite subset-net, bidirectional coverage, transport-stability, and provenance clauses order the public obligation surface before any Hausdorff-distance consumer can read the symmetric value.

Now apply Definition 1660.6 and Definition 1660.7. They force each displayed hit through the compact source and finite schedules, and they require both directed ledgers D_0 and D_1 before the symmetric readback R is public.

The excluded objects are absent from the carrier. Host powersets, quotient compacta, selected limits, open-cover compactness, and choice-selected representatives therefore cannot supply a missing Hausdorff obligation or replace any of $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$. □

Theorem 1660.5 (Hyperspace carrier obligations)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier*

$$H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M),$$

carrier projection exposes exactly compact-source admission through X*, subset-net schedules through* K_0, K_1, N_0, N_1*, directed-coverage ledgers through* D_0, D_1*, tolerance readback through* R*, componentwise transport through* H_s*, displayed replay through* C*, provenance through* P*, and local* $\mathsf{NameCert}$ *rows through* M*. No root consumer receives any carrier row outside this list.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCarrierObligations`

Proof. Project the displayed carrier of Definition 1660.1. Its coordinates are precisely $X, K_0, K_1, N_0, N_1, D_0, D_1, R,$

Use Definition 1660.3. The compact-source, finite subset-net, bidirectional coverage, transport, replay, provenance, and local-name clauses account for each coordinate in the carrier.

The tolerance readback is R, and the directed-coverage ledgers are D_0, D_1. Since the excluded host powerset, quotient compactum, selected-limit, open-cover, and choice-selected coordinates are absent, no root consumer can read beyond the listed rows. □

Theorem 1660.6 (Hyperspace finite-net hit)**.** *Every directed Hausdorff hit exposed by an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must first pass through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row* X *and the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules carried by* K_0, K_1, N_0, N_1 *before the tolerance readback row* R *can be consumed. Thus a hit in* D_0 *or* D_1 *is a scheduled finite-net comparison over the displayed compact source, not a host point-selection event.*

Proof. Read Definition 1660.5. A directed hit is public only inside D_0 or D_1, and those rows occur after the compact-source and subset-net schedules have been exposed.

The schedules K_0, K_1, N_0, N_1 are attached to the same source row X. They carry the finite centers and resolution ledgers that a Hausdorff comparison may inspect.

Only after those rows are present can R compare the directed ledgers at the requested tolerance. Therefore every hit is routed through $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{TotallyBounded}^{\uparrow}$ finite-net data before tolerance readback, with no selected host point added to the carrier. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteNetHit`

Theorem 1660.7 (Hyperspace finite hit directed cover)**.** *Every finite Hausdorff hit accepted by a* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must expose both directed cover rows* D_0 *and* D_1 *before any symmetric Hausdorff-distance read is public. A consumer may read a hit from* K_0 *toward* K_1 *only as part of* D_0*, and a hit from* K_1 *toward* K_0 *only as part of* D_1*; neither direction can be replaced by a host compact-subset choice, quotient compactum, selected nearest point, or one-sided coverage oracle.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteHitDirectedCover`

Proof. Project the carrier of Definition 1660.1. The only directed coverage coordinates are D_0 and D_1, and both are listed before the tolerance readback row R.

Apply Definition 1660.6. Each displayed hit is already a finite-net comparison over the compact source and its total-bounded schedules.

Use Definition 1660.3. The bidirectional distance clause requires both directed ledgers before the public Hausdorff read is admitted, so no one-sided host selection can stand in for the missing direction. □

Theorem 1660.8 (Hyperspace tolerance readback exhaustion)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, the tolerance visible to a finite Hausdorff-distance consumer is exhausted by the readback row* R*. Once the compact source, subset-net schedules, and directed ledgers* D_0, D_1 *have been displayed, no downstream consumer may read an additional tolerance comparison from* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{NetConvergence}^{\uparrow}$*,* $\mathsf{CauchyUniformNet}^{\uparrow}$*, a selected limit, or a host real completeness coordinate.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceToleranceReadbackExhaustion`

Proof. Begin with Definition 1660.7. The symmetric Hausdorff read is not public until both directed cover rows D_0 and D_1 have been shown.

The carrier definition names R as the dyadic and $\mathsf{Real}^{\uparrow}$ tolerance readback for those two directed ledgers. Thus every comparison between the two directions passes through the same displayed row.

Finally apply Definition 1660.10. Completeness, diameter, convergence, and uniform-net rows support the finite route, but they do not export selected limits or an ambient tolerance object outside R. □

Theorem 1660.9 (Hyperspace compact source scope)**.** *Every compact-subset read of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet is scoped to the displayed rows*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CauchyUniformN}$

The read may inspect compact source admission, finite-net schedules, completion boundary, diameter or tolerance comparisons, net-convergence replay, and uniform Cauchy-net replay only through those rows and the carrier coordinates $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$*. It exports no host powerset of compact subsets, open-cover compactness coordinate, quotient compactum, selected limit, or choice-selected representative.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace compact source scope`

Proof. Read the source row X in Definition 1660.1. It already contains the compact metric source, total-bounded finite schedules, and complete metric boundary available to the packet.

Use Definition 1660.11. The public Hausdorff route passes through the diameter, directed convergence, and uniform Cauchy-net rows only as displayed support for D_0, D_1, R and replay C.

Apply Definition 1660.8. Since tolerance is exhausted by R and every directed hit is finite, the listed compact-source rows are the whole compact-subset scope. □

Theorem 1660.10 (Hyperspace CompleteMetric nonescape)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier, the* $\mathsf{CompleteMetric}^{\uparrow}$ *support carried by the source row* X *supplies only the displayed completion boundary needed by the finite compact-metric route. It cannot export selected limits, quotient compacta, open-cover compactness, choice-selected representatives, or an ambient hyperspace completion outside*

$$X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M.$$

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace complete metric nonescape`

Proof. Begin with Definition 1660.11. Completeness enters the route through the compact metric source row X before subset-net schedules, directed ledgers, and tolerance readback are public.

Apply Definition 1660.6. Any Hausdorff hit must pass through the finite-net schedules before it reaches R, so completeness cannot be read as an independent selected limit for a compact subset.

Finally use Definition 1660.5. The only remaining rows are transport, replay, provenance, and local naming. None is a quotient compactum, open-cover compactness proof, choice-selected representative, or ambient hyperspace completion, so the CompleteMetric support cannot escape the displayed boundary. □

Theorem 1660.11 (Hyperspace compact Hausdorff-distance route)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, the public Hausdorff-distance route is exactly the finite route through the* $\mathsf{CompactMetric}^{\uparrow}$ *source admission row* X*, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules in* K_0, K_1, N_0, N_1*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit boundary carried by* X*, the* $\mathsf{MetricDiameter}^{\uparrow}$ *tolerance readback in* R*, and the* $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{CauchyUniformNet}^{\uparrow}$ *replay named by the directed ledgers* D_0, D_1 *and the replay row* C*. No downstream compact-subset consumer receives a host powerset, open-cover compactness coordinate, quotient compactum, or selected limit object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompactHausdorffDistanceRoute`

Proof. Project the carrier in Definition 1660.1. The coordinate X is the only compact metric source row, and it already contains the displayed metric witnesses, total-boundedness support, and completeness boundary.

The subset coordinates K_0, K_1, N_0, N_1 provide the finite net schedules. Thus every center inspected by a Hausdorff-distance request is read through the same $\mathsf{TotallyBounded}^{\uparrow}$ schedule attached to the compact source rather than through an ambient compact-subset universe.

The directed ledgers D_0 and D_1 supply the two coverage directions. The tolerance row R packages their dyadic and $\mathsf{Real}^{\uparrow}$ readback, so the distance request can only compare the two directed ledgers through the displayed $\mathsf{MetricDiameter}^{\uparrow}$ boundary.

Apply Definition 1660.2 to transport the directed coverage rows and apply Definition 1660.3 to exhaust the local naming surface. The replay row C keeps every convergence read inside the displayed $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{CauchyUniformNet}^{\uparrow}$ route.

Finally, P and M certify that no further coordinates are public. The packet therefore exports the compact Hausdorff-distance route only through $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$. □

Theorem 1660.12 (Hyperspace finite Hausdorff carrier)**.** *Every public Hausdorff-distance read of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier is exactly the pair of directed finite ledgers* D_0, D_1 *together with the tolerance readback row* R*, over the displayed source and net rows*

$$X, K_0, K_1, N_0, N_1.$$

The read exposes no host powerset of compact subsets, quotient compactum, selected limit object, open-cover compactness coordinate, or choice-selected representative.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteHausdorffCarrier`

Proof. Project the carrier of Definition 1660.1. The compact source row X and the finite subset-net rows K_0, K_1, N_0, N_1 are the only rows that can be inspected before a Hausdorff-distance request.

Apply Definition 1660.11. The public distance route consumes both directed ledgers D_0, D_1 and then reads R as the dyadic and $\mathsf{Real}^{\uparrow}$ tolerance comparison.

Use Definition 1660.3 to exhaust the local naming surface. Since the forbidden host powerset, quotient, selected-limit, open-cover, and choice-selected coordinates are absent from the carrier, the finite directed-ledger read is the whole public Hausdorff surface. □

Theorem 1660.13 (Hyperspace complete-metric handoff)**.** *Completeness in an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet enters only through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source row* X *and its* $\mathsf{CompleteMetric}^{\uparrow}$ *support. It cannot be replaced by host compactness, selected limit data, open-cover compactness, or an ambient compact-subset universe outside* $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$.

Proof. Read the source coordinate X in Definition 1660.1. It is the only row that carries the compact metric source together with the displayed metric witnesses, finite-net schedules, total-boundedness row, and completeness boundary.

Apply Definition 1660.12. The subset and distance rows use X through finite net ledgers and directed coverage rows, so a Hausdorff read never asks for a host compact-subset universe or a selected limit.

Finally apply Definition 1660.2. Transport preserves the same source row and directed ledgers, while H_s, C, P, M add only structural comparison, replay, provenance, and local naming. Thus completeness is read only through the displayed CompactMetric and CompleteMetric source support. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompleteMetricHandoff`

Definition 1660.14 (Hyperspace finite-window carrier)**.** *The finite-window surface of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier is the same displayed packet*

$$H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$$

read with X *as the* $\mathsf{CompactMetric}^{\uparrow}$ *source,* K_0, K_1, N_0, N_1 *as finite subset-net schedules over* X*,* D_0, D_1 *as the two directed Hausdorff ledgers,* R *as the dyadic and* $\mathsf{Real}^{\uparrow}$ *readback row,* H_s *as componentwise* hsame *transport,* C *as displayed* Cont *replay,* P *as* Pkg *provenance, and* M *as the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. A finite-window read has no coordinate for a host hyperspace, selected compact subset, open-cover compactness proof, quotient compactum, or ambient completeness witness outside* X*.*

Lean4 (definition) `BEDC.Derived.HyperspaceUp.HyperspaceFiniteWindowCarrier`

Theorem 1660.15 (Hyperspace obligation surface)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, the root-unblock obligation surface consists exactly of the* $\mathsf{CompactMetric}^{\uparrow}$ *source admission in* X*, the finite subset-net schedules* K_0, K_1, N_0, N_1*, the two directed Hausdorff ledgers* D_0, D_1*, the readback row* R*, componentwise* hsame *transport through* H_s*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local* NameCert *rows through* M*. Every downstream compact-metric or Cauchy-uniform-net consumer must consume those rows before any Hausdorff-distance claim is public.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace namecert obligation surface`

Proof. Start from Definition 1660.14. It names the same twelve coordinates as the carrier of Definition 1660.1, but marks the finite-window roles needed by root consumers.

Apply Definition 1660.3. The carrier, subset-net, bidirectional coverage, transport, replay, provenance, and local-name clauses are already exhaustive for the local certificate.

The root-unblock surface is therefore obtained by reading the same clauses in consumer order. CompactMetric admission comes from X, finite subset-net schedules from K_0, K_1, N_0, N_1, directed Hausdorff coverage from D_0, D_1, and readback from R.

No further row can be inserted because H_s, C, P, M only transport, replay, source, and name the displayed packet. The forbidden host hyperspace, selected compact subset, quotient compactum, open-cover compactness proof, and ambient completeness witness are absent from the carrier. □

Theorem 1660.16 (Hyperspace CompactMetric distance lift)**.** *Let* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ *be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Any compact-metric distance consumer of* H *reads a Hausdorff-distance request only by lifting the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source row* X *through the finite subset-net schedules* K_0, K_1, N_0, N_1*, the two directed Hausdorff ledgers* D_0, D_1*, and the tolerance readback row* R*. The lift is stable under componentwise* hsame *through* H_s*, replayed by the displayed* Cont *row* C*, and named by the* Pkg *and local* NameCert *rows* P, M*.*

Proof. Project the finite-window carrier from Definition 1660.14. The only compact-metric source row is X, so every distance read starts there.

Use Definition 1660.11. It routes the Hausdorff request through the subset-net schedules, both directed coverage ledgers, and the tolerance readback row before any public distance value is available.

Finally apply Definition 1660.2. Componentwise hsame transport through H_s and displayed replay through C preserve that route, while P, M name it without adding a host compact-subset universe. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace compactmetric distance lift`

Theorem 1660.17 (Hyperspace totally bounded net lift)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every* $\mathsf{TotallyBounded}^{\uparrow}$ *net request exposed by the hyperspace packet is read through the finite subset-net schedules K_0, K_1, N_0, N_1 over the same* $\mathsf{CompactMetric}^{\uparrow}$ *source row X. The request may use D_0, D_1 and R to compare directed Hausdorff ledgers, but it exports no unscheduled net, selected center family, host powerset, or quotient compactum.*

Proof. Read the root-unblock obligation surface of Definition 1660.15. The finite subset-net schedules are named before any directed Hausdorff ledger can be public.

The total-boundedness source is part of X and is inspected through K_0, K_1, N_0, N_1. Thus a net request cannot detach a center family from the compact source row or bypass the finite schedules.

The directed ledgers D_0, D_1 and readback R compare only the scheduled data. Transport and replay through H_s, C preserve the same rows, and the local naming rows P, M exclude an unscheduled net, host powerset, and quotient compactum. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace totallybounded net lift`

Theorem 1660.18 (Hyperspace Vietoris compact-metric dependency)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every Vietoris-topology or Hausdorff-distance consumer of H must read the* $\mathsf{CompactMetric}^{\uparrow}$ *source row X, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules in K_0, K_1, N_0, N_1, the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary carried by X, the* $\mathsf{Met}^{\uparrow}$ *distance witnesses, and the two directed Hausdorff ledgers D_0, D_1 before any compact-subset topology route is public.*

The dependency route exports no host powerset, open-cover compactness proof, quotient compactum, selected limit, or choice-selected representative.

Proof. Project the carrier of Definition 1660.1. The compact metric source row X is the only source from which metric witnesses, total-boundedness schedules, and completeness support can be read.

Use Definition 1660.3. Its carrier-admission, finite subset-net, bidirectional coverage, transport, and provenance clauses force every public read to pass through $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$.

Apply Definition 1660.11 and Definition 1660.12. They identify the Hausdorff route as a finite two-direction ledger over the displayed compact source and exclude a host compact-subset universe.

Finally use Definition 1660.17. The Vietoris-facing finite-open or finite-hit information can inspect only the scheduled net rows attached to X, and the Hausdorff-facing read must retain both directed ledgers. Since no forbidden host coordinate is present in the carrier, every compact-subset topology consumer inherits exactly the compact-metric dependency named in the statement. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisCompactMetricDependency`

Theorem 1660.19 (Hyperspace CompleteMetric Cauchy lift)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{CauchyUniformNet}^{\uparrow}$ *consumer of the hyperspace route receives completeness only through the support carried by X, after the finite subset-net schedules and the two directed Hausdorff ledgers have been exposed. The lift is replayed through displayed* Cont*, transported componentwise by* hsame*, and named locally by P, M; it does not export selected limits, quotient completion representatives, open-cover compactness, or ambient hyperspace completeness.*

Proof. Begin with Definition 1660.13. Completeness enters the hyperspace packet only through the displayed compact metric source row X and its CompleteMetric support.

Use Definition 1660.17. A Cauchy-uniform-net consumer first sees finite subset-net schedules over the same source row and then the directed ledgers D_0, D_1 with tolerance readback R.

Apply Definition 1660.2 to transport the route. The rows H_s, C, P, M preserve, replay, source, and name the displayed packet, so no selected limit, quotient representative, open-cover compactness proof, or ambient hyperspace completeness coordinate is exported. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompleteMetricCauchyLift`

Theorem 1660.20 (Hyperspace finite-net carrier exhaustion)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* Hyperspace$^\uparrow$ *carrier. Every finite-net inspection of a compact-subset presentation in the* BHist *packet is exhausted by the* CompactMetric$^\uparrow$ *source row X, its* Met$^\uparrow$ *distance witnesses, its* CompleteMetric$^\uparrow$ *support, the displayed subset-net rows K_0, K_1, and the resolution ledgers N_0, N_1. Componentwise* hsame *transport through H_s, displayed* Cont *replay through C, and the* NameCert *row M preserve exactly those rows; no host compact-subset family, selected net, quotient compactum, or ambient completeness coordinate is public.*

Proof. Project the carrier of Definition 1660.1. The only compact source is X, and the only finite compact-subset presentations are K_0, K_1 with their resolution ledgers N_0, N_1.

Use Definition 1660.15. The root-unblock surface orders the public rows as compact-source admission, finite subset-net schedules, directed Hausdorff ledgers, readback, transport, replay, provenance, and local naming.

Thus any finite-net inspection must stop at X, K_0, K_1, N_0, N_1 before reading the directed ledgers. The rows H_s, C, M preserve and name that same visible packet, so no extra compact-subset carrier can be exposed. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace finite net carrier exhaustion`

Theorem 1660.21 (Hyperspace finite Vietoris carrier)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* Hyperspace$^\uparrow$ *carrier. Every Vietoris-facing carrier read of a compact-source finite subset is exposed only through the displayed finite-net rows $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ of Definition 1660.1. The read may inspect the* CompactMetric$^\uparrow$ *source row, the* TotallyBounded$^\uparrow$ *schedules, the* CompleteMetric$^\uparrow$ *support carried by X, and the* CauchyUniformNet$^\uparrow$ *replay row, but it does not export a host compact-subset universe, quotient compactum, open-cover compactness proof, choice-selected representative, or ambient hyperspace completion.*

Proof. Project the carrier of Definition 1660.1. The compact-source finite subsets are represented by K_0, K_1 and their finite resolution ledgers N_0, N_1 over the same source row X.

Use Definition 1660.20. It exhausts every finite-net inspection at X, K_0, K_1, N_0, N_1 before any directed Hausdorff ledger can be read.

Then apply Definition 1660.18 and Definition 1660.11. These force the Vietoris-facing route through the compact metric source, total-bounded schedules, complete-metric support, directed ledgers, and Cauchy-uniform-net replay. Since the carrier has no row for quotient compacta, open-cover compactness, selected representatives, host powersets, or ambient hyperspace completion, the finite Vietoris carrier is exactly the displayed packet. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteVietorisCarrier`

Theorem 1660.22 (Hyperspace hit-miss classifier)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* Hyperspace$^\uparrow$ *carrier. Every hit or miss comparison used by a Vietoris-facing consumer is read through the finite directed-distance rows D_0, D_1 and the tolerance readback row R,*

after the finite subset-net schedules K_0, K_1, N_0, N_1 have been exposed over X. The classifier is therefore a finite directed Hausdorff ledger in the BHist *packet, not a comparison inside a host compact-subset universe.*

Proof. Start from the carrier projection of Definition 1660.1. The only rows that can witness a hit or miss request are the finite subset-net schedules and the two directed coverage ledgers.

Apply Definition 1660.20. It prevents the classifier from reading a compact-subset carrier beyond X, K_0, K_1, N_0, N_1 before the directed ledgers are public.

Use Definition 1660.24. Componentwise hsame transport preserves the same directed ledgers D_0, D_1 and the same readback row R, so a transported classifier cannot replace them by a host distance, quotient compactum, selected representative, or ambient hyperspace metric. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHitMissClassifier`

Theorem 1660.23 (Hyperspace finite-cover ledger)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* Hyperspace$^{\uparrow}$ *carrier. Every finite-cover ledger consumed downstream is routed through the* CompactMetric$^{\uparrow}$ *source row X, the* TotallyBounded$^{\uparrow}$ *schedules in K_0, K_1, N_0, N_1, the* CompleteMetric$^{\uparrow}$ *support carried by X, and the* CauchyUniformNet$^{\uparrow}$ *replay visible through D_0, D_1, R, C. The ledger exports no open-cover compactness principle, quotient compactum, choice-selected representative, host compact-subset universe, selected limit, or ambient hyperspace completion.*

Proof. Project Definition 1660.1. The finite-cover information is not a separate host cover object; it is the displayed subset-net data K_0, K_1, N_0, N_1 attached to X.

Use Definition 1660.27. It identifies the complete-metric and total-boundedness obligations as the source support in X, the finite schedules, the two directed ledgers, and the tolerance readback.

Finally apply Definition 1660.19 and Definition 1660.25. These route downstream Cauchy-uniform-net and compact-complete consumers through the same finite ledger. No row remains from which to read open-cover compactness, quotient compacta, selected representatives, selected limits, host compact subsets, or ambient hyperspace completion. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteCoverLedger`

Theorem 1660.24 (Hyperspace Hausdorff classifier transport)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* Hyperspace$^{\uparrow}$ *carrier. Any Hausdorff-distance classifier comparison transported by* hsame *through the* BHist *packet must carry both directed ledgers D_0, D_1 over the same* CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, *and* CompleteMetric$^{\uparrow}$ *source rows, and must replay through the displayed* Cont *row C before the* NameCert *row M names the comparison. The transport cannot replace either directed ledger by a host distance, quotient compactum, selected limit, or external hyperspace metric.*

Proof. Apply Definition 1660.2. It states that componentwise hsame transport preserves the two directed Hausdorff ledgers and the tolerance readback.

The compact-source part of the comparison is fixed by Definition 1660.18. A classifier comparison must therefore retain the same CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, and CompleteMetric$^{\uparrow}$ source rows before any distance route is public.

Finally use Definition 1660.19. Displayed Cont replay keeps the transported comparison inside the finite Cauchy-uniform-net route, and the local NameCert row names only that packet. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHausdorffClassifierTransport`

Theorem 1660.25 (Hyperspace compact-complete handoff)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every downstream compact-complete consumer receives a handoff whose* BHist *rows are exactly the* $\mathsf{CompactMetric}^{\uparrow}$ *source X, the* $\mathsf{Met}^{\uparrow}$ *distance witnesses inside X, the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary carried by X, the finite subset-net rows K_0, K_1, N_0, N_1, the directed Hausdorff ledgers D_0, D_1, componentwise* hsame *transport H_s, displayed* Cont *replay C, and local* NameCert *naming through M. The handoff exports no ambient hyperspace completeness proof, open-cover compactness proof, selected Cauchy limit, quotient completion representative, or host powerset.*

Proof. Begin with Definition 1660.19. Completeness is handed to downstream Cauchy consumers only through the support carried by X after the finite subset-net schedules and directed Hausdorff ledgers are exposed.

Use Definition 1660.20. It identifies the compact-subset carrier rows that a downstream consumer may inspect before reading the distance ledgers.

Then apply Definition 1660.24. Transport and replay preserve the same compact-source, finite-net, and directed-ledger rows, and M names only the displayed packet. Therefore the compact-complete handoff is exactly the finite BEDC interface stated above. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompactCompleteHandoff`

Definition 1660.26 (Hyperspace finite-net carrier)**.** *The finite-net carrier of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet is the displayed subpacket*

$$F_H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$$

read with X as the $\mathsf{CompactMetric}^{\uparrow}$ *source row, K_0, K_1, N_0, N_1 as compact finite-net data, D_0, D_1 as the two Hausdorff coverage ledgers, R as dyadic and* $\mathsf{Real}^{\uparrow}$ *tolerance readback, H_s as componentwise* hsame *transport, C as displayed* Cont *replay, P as* Pkg *provenance, and M as the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. This carrier has no coordinate for host powersets, selected compact representatives, open-cover compactness, quotient compacta, or ambient hyperspace completeness.*

Lean4 (definition) `BEDC.Derived.HyperspaceUp.HyperspaceFiniteNetCarrier`

Theorem 1660.27 (Hyperspace complete and totally bounded obligations)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. The complete-metric and total-boundedness obligations exposed by the finite-net carrier are exactly the* $\mathsf{CompleteMetric}^{\uparrow}$ *support carried by the compact source row X, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules read by K_0, K_1, N_0, N_1, the two directed Hausdorff coverage ledgers D_0, D_1, the tolerance readback R, componentwise* hsame *transport H_s, displayed* Cont *replay C,* Pkg *provenance P, and local naming M.*

No downstream consumer receives a host powerset of compact subsets, a selected compact representative, open-cover compactness, quotient compactness, or a selected Cauchy limit.

Proof. Start from Definition 1660.26. It records the same finite rows as the accepted carrier and marks the compact finite-net roles used by root consumers.

Apply Definition 1660.13. Completeness is read only through the $\mathsf{CompactMetric}^{\uparrow}$ source row X and its displayed $\mathsf{CompleteMetric}^{\uparrow}$ support.

Apply Definition 1660.17. Total-boundedness is read through the finite schedules K_0, K_1, N_0, N_1 over the same source row. The directed ledgers and tolerance readback can compare only those scheduled rows.

The transport, replay, provenance, and local naming rows preserve this finite packet. Since the excluded host and selection coordinates are absent, the listed rows exhaust the complete and totally bounded obligations. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompleteTotallyBoundedObligations`

Theorem 1660.28 (Hyperspace Cauchy-uniform-net handoff)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyUniformNet}^{\uparrow}$ *handoff from H must pass through the compact finite-net carrier, inspect the* $\mathsf{CompactMetric}^{\uparrow}$ *source X, read the finite subset-net schedules K_0, K_1, N_0, N_1, consume both directed coverage ledgers D_0, D_1, and replay the same route through C after componentwise* hsame *transport in H_s.*

The handoff exports only finite Cauchy-uniform-net data named by P and M. It exports no host powerset, selected compact representative, selected limit, open-cover compactness proof, or quotient compactum.

Proof. Use Definition 1660.19. It sends every $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{CauchyUniformNet}^{\uparrow}$ consumer through the completeness support carried by X, after finite subset-net schedules and directed Hausdorff ledgers are exposed.

The finite-net carrier of Definition 1660.26 fixes the rows available to that handoff. The consumer reads $X, K_0, K_1, N_0, N_1, D_0, D_1, R$ and then follows the structural rows.

Transport and replay are supplied by H_s and C. The local naming and provenance rows P, M record the same route, so no selected compact representative, quotient compactum, open-cover proof, host powerset, or selected limit can be introduced at the handoff. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCauchyUniformNetHandoff`

Theorem 1660.29 (Hyperspace metric compactness unblock)**.** *Every compact-metric downstream consumer of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet can read metric compactness only through the finite-net carrier of Definition 1660.26. The unblocked surface consists of compact finite-net data,* $\mathsf{CompleteMetric}^{\uparrow}$ *support,* $\mathsf{TotallyBounded}^{\uparrow}$ *schedules, the* $\mathsf{CauchyUniformNet}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

The unblock supplies no host powerset, no selected compact representative, no open-cover compactness principle, no quotient compactum, and no ambient hyperspace completeness theorem.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceMetricCompactnessUnblock`

Proof. First read the complete and totally bounded obligations of Definition 1660.27. They give exactly the compact source support, finite schedules, directed ledgers, tolerance readback, transport, replay, provenance, and local naming rows.

Next apply the Cauchy-uniform-net handoff of Definition 1660.28. It routes every Cauchy-uniform-net consumer through the same compact finite-net carrier before any downstream compactness read is public.

Finally use Definition 1660.25. The compact-complete consumer receives only the displayed BHist rows over $X, K_0, K_1, N_0, N_1, D_0, D_1, H_s, C, M$. Thus metric compactness is unblocked as finite BEDC data and not as a host powerset, selected representative, open-cover principle, quotient compactum, or ambient completeness theorem. □

Theorem 1660.30 (Hyperspace compact-metric Vietoris unblock)**.** *Every Vietoris-facing downstream consumer of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must read compact-source support through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, *and* $\mathsf{CauchyUniformNet}^{\uparrow}$ *before any hit/miss or Hausdorff-distance classifier is public. The route exports no host powerset, quotient compactum, selected representative, open-cover compactness, or ambient hyperspace completion.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompactMetricVietorisUnblock`

Proof. Project the carrier of Definition 1660.1. Its rows expose the compact metric source, finite subset-net presentations, directed Hausdorff ledgers, tolerance readback, transport, replay, provenance, and local naming.

Use Definition 1660.20. It forces every compact-source finite-net read through the displayed rows before a downstream consumer can inspect distance or Vietoris data.

Apply Definition 1660.22 and Definition 1660.23. The hit/miss and finite-cover reads are both mediated by the same finite net schedules and directed ledgers, so neither supplies a host powerset or an open-cover compactness principle.

Finally use Definition 1660.29. Compact-metric readiness is public only through the compact source, total-bounded schedules, completeness support, metric-diameter tolerance, net-convergence route, and Cauchy-uniform-net handoff already present in the accepted carrier. The forbidden quotient, representative, powerset, open-cover, and ambient-completion coordinates are absent. □

Theorem 1660.31 (Hyperspace finite-cover selector refusal)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Every finite-cover read exported by H is mediated by the displayed compact finite-net rows, the Vietoris hit/miss rows, the two directed coverage ledgers D_0, D_1, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. The packet cannot export a selected compact representative, a selected point of a cover member, a host powerset element, a quotient compactum, or an open-cover compactness oracle.*

Proof. Project the carrier of Definition 1660.1. Its only compact-subset data are the compact metric source, the finite subset-net schedules, directed ledgers, tolerance readback, and structural rows.

Use Definition 1660.20. It forces every compact finite-net read through $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ so no selected representative or host powerset coordinate is available.

Apply Definition 1660.22, Definition 1660.23, and Definition 1660.30. The hit/miss and finite-cover classifiers are read from the same finite-net and Vietoris rows already required by the compact-metric route. Therefore the finite-cover surface is a refusal of selectors, not a source of hidden compact subsets. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteCoverSelectorRefusal`

Theorem 1660.32 (Hyperspace hit-miss coverage totality)**.** *Every hit/miss coverage read from an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, *and* $\mathsf{CauchyUniformNet}^{\uparrow}$ *rows before it is public. The coverage read is total over the accepted packet precisely because each available route is one of the finite-net, Vietoris, directed Hausdorff, metric-diameter, net-convergence, or Cauchy-uniform handoff rows already stored in the carrier.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHitMissCoverageTotality`

Proof. Start with Definition 1660.30. It lists the compact-source, total-bounded, complete-metric, metric-diameter, net-convergence, and Cauchy-uniform-net rows that must be displayed before a Vietoris-facing consumer can read hit/miss data.

Next use Definition 1660.36 and Definition 1660.28. The first theorem routes the compact-subset topology read through finite-net and Vietoris-hit rows, while the second routes the completion-facing handoff through the same finite carrier.

Finally apply Definition 1660.31. Since the finite-cover surface has no selector, host powerset, quotient compactum, or open-cover oracle, every public hit/miss coverage answer is accounted for by the displayed finite rows. That is the asserted totality of coverage reads over the accepted packet. □

Theorem 1660.33 (Hyperspace root compact-source totality)**.** *Every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet must expose the compact-source row, the metric-distance row, the total-bounded finite-net row, and the completeness-support row before any hit/miss read is admitted. Thus a hit/miss consumer reads the displayed* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *support through the carrier rows of Definition 1660.1, not through a host powerset, selected compact representative, open-cover oracle, or ambient hyperspace completion.*

Proof. Project the carrier of Definition 1660.1. The compact source, finite subset-net schedules, directed Hausdorff ledgers, tolerance readback, transport, replay, provenance, and local naming rows are the only rows exposed to a consumer.

Use Definition 1660.30 and Definition 1660.32. They require compact-metric support, total-bounded schedules, complete-metric support, metric-diameter tolerance, net-convergence routes, and Cauchy-uniform handoff before hit/miss coverage is public.

The root read is therefore total only after those source rows are visible. Since the carrier has no coordinate for host powersets, selected representatives, open-cover oracles, or ambient completion, every admitted hit/miss read factors through the displayed compact-source and metric rows. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootCompactSourceTotality`

Theorem 1660.34 (Hyperspace finite-net hit-miss determinacy)**.** *For an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, the finite-net data* K_0, K_1, N_0, N_1 *determine the public hit/miss ledger under componentwise* hsame *transport and displayed* Cont *replay. Any transported or replayed hit/miss read factors through the same finite-net schedules, directed ledgers* D_0, D_1*, readback row* R*, transport row* H_s*, replay row* C*, provenance row* P*, and local name* M*.*

Proof. Start with Definition 1660.36. It routes compact-subset topology reads through the compact source, finite-net schedules, and Vietoris hit rows before a consumer can inspect compact-subset data.

Apply Definition 1660.20. It identifies $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ as the complete finite carrier available for compact-subset reads.

Now transport or replay the read. Componentwise hsame transport preserves the displayed rows, and Cont replay repeats the same route through C. Therefore finite-net data determine the hit/miss ledger for every accepted transported or replayed read, without adding a selected compact set or host powerset element. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootFiniteNetHitMiss`

Theorem 1660.35 (Hyperspace Hausdorff transport root)**.** *Every Hausdorff-distance readback exported by an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet transports along the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *classifier rows. The readback must pass through the compact source, finite-net schedules, directed Hausdorff ledgers, metric-diameter tolerance row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row before it is public.*

Proof. Use Definition 1660.4. The two directed coverage ledgers D_0, D_1 are the only Hausdorff-facing ledgers in the accepted carrier.

Next apply Definition 1660.11 and Definition 1660.25. The readback tolerance is routed through the metric-diameter and compact-complete support rows, while completion-facing consumers are sent through the displayed $\mathsf{CompleteMetric}^{\uparrow}$ handoff.

Finally use Definition 1660.33. It fixes the compact-source and metric rows as prerequisites for hit/miss and Hausdorff-facing reads. Transport along $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ therefore remains a finite carrier read through $X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$, not a host Hausdorff metric on a powerset. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootHausdorffTransport`

Theorem 1660.36 (Hyperspace Vietoris finite-net obligations)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Before any downstream consumer reads a compact-subset topology route or a Hausdorff-distance route from H, the carrier must expose the* $\mathsf{CompactMetric}^{\uparrow}$ *source X, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules in K_0, K_1, N_0, N_1, and the Vietoris hit rows recorded by the directed ledgers D_0, D_1. Thus every compact-subset read is a finite-net and Vietoris-hit read over the displayed source, not a host powerset, selected compact representative, or open-cover compactness principle.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisFiniteNetObligations`

Proof. Start with Definition 1660.6. It forces every directed Hausdorff hit to pass through the compact source row and the finite-net schedules before tolerance readback is public.

Use Definition 1660.20. The only compact-subset rows available to a consumer are X, K_0, K_1, N_0, N_1 together with the directed ledgers and structural rows.

Finally apply Definition 1660.29. Since the unblocked surface is exactly the finite-net carrier with transport, replay, provenance, and local naming, a Vietoris compact-subset read cannot bypass those obligations. □

Theorem 1660.37 (Hyperspace Cauchy-uniform complete handoff)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. The only completion-facing handoff from H is the route that reads the* $\mathsf{CompleteMetric}^{\uparrow}$ *support carried by X, consumes the* $\mathsf{CauchyUniformNet}^{\uparrow}$ *replay through C, and then names the same finite-net carrier through P and M. No ambient hyperspace completion, selected Cauchy limit, quotient completion representative, or host equality of compact subsets is exported.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCauchyUniformCompleteHandoff`

Proof. Apply Definition 1660.19. It sends complete-metric and Cauchy-uniform-net consumers through the support carried by X after the finite subset-net schedules and directed Hausdorff ledgers are present.

Next use Definition 1660.28. It fixes the Cauchy-uniform handoff as a read of the compact finite-net carrier followed by componentwise hsame transport and displayed Cont replay.

The finite-net carrier of Definition 1660.26 supplies exactly those rows. Therefore completion enters the hyperspace packet only through $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{CauchyUniformNet}^{\uparrow}$ rows already named by the BEDC carrier. □

Theorem 1660.38 (Hyperspace Hausdorff classifier exactness)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$ be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. The Hausdorff classifier exposed by H is exactly componentwise* hsame *transport of the displayed source, compact finite-net rows, directed ledgers, tolerance*

readback, displayed Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *naming. It introduces no quotient equality, no extensional host equality of compact subsets, and no replacement distance outside the two directed ledgers* D_0, D_1*.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHausdorffClassifierExactness`

Proof. First apply Definition 1660.2. Componentwise hsame transport preserves the two directed Hausdorff ledgers and the tolerance readback.

Then use Definition 1660.24. The classifier comparison must retain the same compact-source, complete-metric, finite-net, transport, replay, and local naming rows.

Finally invoke Definition 1660.3. The $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ row names only the displayed packet, so the classifier has no coordinate on which quotient equality or host extensional equality could enter. □

Theorem 1660.39 (Hyperspace Vietoris hit-miss stability)**.** *Let* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ *be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. If a downstream Vietoris consumer reads a hit row or miss row from* H*, then the read is stable under the same componentwise* hsame *transport that preserves the Hausdorff classifier: the compact-source row* X*, finite-net rows* K_0, K_1, N_0, N_1*, directed ledgers* D_0, D_1*, tolerance readback* R*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row* M *remain the public coordinates of the read.*

The stability route exports no host powerset of compact subsets, no open-cover compactness principle, no selected representative of a compactum, and no quotient equality beyond the displayed transport rows.

Proof. First apply Definition 1660.22. It identifies every Vietoris-facing hit or miss read with finite carrier rows rather than with an ambient powerset.

Next use Definition 1660.6 and Definition 1660.23. These force the hit/miss read through compact-source support, finite-net schedules, directed Hausdorff ledgers, and tolerance readback.

Finally apply Definition 1660.38. The same componentwise hsame transport that preserves the Hausdorff classifier preserves the displayed Vietoris rows, so the transported hit/miss route remains a BEDC finite-net route. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisHitMissStability`

Theorem 1660.40 (Hyperspace finite Vietoris obligation basis)**.** *For an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, the finite obligation basis for every compact-metric Vietoris read is exactly the basis consisting of compact-source support through* X*, finite-net presentations through* K_0, K_1, N_0, N_1*, hit/miss and finite-cover ledgers through* D_0, D_1*, complete-metric support through* $\mathsf{CompleteMetric}^{\uparrow}$*, total-bounded schedules through* $\mathsf{TotallyBounded}^{\uparrow}$*, net-convergence routes through* $\mathsf{NetConvergence}^{\uparrow}$*, Cauchy-uniform-net replay through* $\mathsf{CauchyUniformNet}^{\uparrow}$*, componentwise transport through* H_s*, displayed replay through* C*, provenance through* P*, and local naming through* M*.*

No additional Vietoris obligation may be supplied by an ambient compact-subset topology, a host open-cover theorem, a selected compact representative, or a quotient compactum.

Proof. Read the compact-metric source obligations from Definition 1660.27. They provide the $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{TotallyBounded}^{\uparrow}$ support together with the finite compact-source rows.

Use Definition 1660.30 and Definition 1660.36. Together they force every Vietoris-facing read through finite-net presentations, hit/miss rows, finite-cover ledgers, and directed Hausdorff data before the compact-subset route is public.

The handoff through $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{CauchyUniformNet}^{\uparrow}$ is fixed by Definition 1660.28. Repacking those rows with H_s, C, P, M gives the stated finite obligation basis and leaves no coordinate for ambient topology or selected representatives. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteVietorisObligationBasis`

Theorem 1660.41 (Hyperspace Hausdorff-net completeness handoff)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s,$ be an accepted $\mathsf{Hyperspace}^{\uparrow}$ carrier. A completeness-facing consumer may pass from the Hausdorff classifier to a net-completeness read only by following the finite route*

$$\begin{gathered} X, K_0, K_1, N_0, N_1, D_0, D_1, R \\ \longrightarrow \mathsf{CompleteMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow}, \mathsf{NetConvergence}^{\uparrow}, \mathsf{CauchyUniformNet}^{\uparrow} \\ \longrightarrow H_s, C, P, M. \end{gathered}$$

Thus Hausdorff-distance convergence and Cauchy-uniform-net completeness are handed off through the displayed compact-metric rows, not through an ambient complete hyperspace, a selected compact limit, quotient completion, or host equality of compact subsets.

Proof. Start with Definition 1660.38. It fixes the Hausdorff classifier as transport of the displayed compact source, finite-net rows, directed ledgers, tolerance readback, replay, provenance, and local naming.

Next apply Definition 1660.37. That theorem identifies the completion-facing handoff as a read of $\mathsf{CompleteMetric}^{\uparrow}$ support and $\mathsf{CauchyUniformNet}^{\uparrow}$ replay over the same finite-net carrier.

Finally combine Definition 1660.29 with the finite-net carrier of Definition 1660.26. The complete, totally bounded, net-convergent, and Cauchy-uniform rows are all displayed before the handoff reaches H_s, C, P, M, so no ambient complete-hyperspace object or selected compact limit is exported. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHausdorffNetCompletenessHandoff`

Definition 1660.42 (Hyperspace finite carrier basis)**.** *For an accepted $\mathsf{Hyperspace}^{\uparrow}$ carrier $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$, the finite carrier basis is the* BHist *subpacket consisting exactly of the compact-source row X, the finite subset-net rows K_0, K_1, N_0, N_1, the two Hausdorff-distance window rows D_0, D_1, R, the total-bounded schedule and complete-metric handoff rows readable through X, and the structural rows H_s, C, P, M for* hsame, Cont, Pkg, *and* NameCert.

This basis has no coordinate for a host powerset of compact subsets, quotient compactum, selected compact limit, open-cover compactness proof, or external equality of compact sets.

Lean4 (definition) `BEDC.Derived.HyperspaceUp.HyperspaceFiniteCarrierBasis`

Theorem 1660.43 (Hyperspace Hausdorff distance window)**.** *Let $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$ be an accepted $\mathsf{Hyperspace}^{\uparrow}$ carrier. Every public Hausdorff-distance read from H is exhausted by the finite dyadic distance windows displayed in D_0, D_1, R together with the common refinements of the finite subset-net rows K_0, K_1, N_0, N_1. In particular, the read can inspect only source-metric cells from X, scheduled finite centers, the two directed coverage ledgers, the tolerance readback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

No public Hausdorff-distance read may appeal to a host powerset, quotient compactum, selected nearest point, selected compact limit, open-cover compactness theorem, or distance relation outside the displayed dyadic windows and their common refinements.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace hausdorff distance window`

Proof. Start from Definition 1660.42. The finite carrier basis lists the compact source, subset-net schedules, two directed ledgers, tolerance readback, and structural rows as the only readable coordinates.

Apply Definition 1660.38. It identifies the Hausdorff classifier with componentwise transport of those displayed rows, so any distance read must pass through D_0, D_1, R after the source metric and subset nets have been fixed.

Use Definition 1660.41. Completeness-facing consumers inherit the same finite route through $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, and $\mathsf{CauchyUniformNet}^{\uparrow}$, and the handoff never exports an ambient complete hyperspace or selected compact limit. □

Theorem 1660.44 (Hyperspace compact-metric handoff)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, a compact-metric downstream consumer receives only the compact finite-net rows of* X, K_0, K_1, N_0, N_1*, the total-bounded schedules readable through* $\mathsf{TotallyBounded}^{\uparrow}$*, the complete-metric support readable through* $\mathsf{CompleteMetric}^{\uparrow}$*, the metric-distance and Hausdorff-window rows* D_0, D_1, R*, and the structural rows* H_s, C, P, M*.*

The handoff exports no host powerset of compact subsets, no quotient compactum, no open-cover compactness principle, no selected limit, and no untracked distance object outside the finite carrier basis.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompactMetricHandoff`

Proof. Read the basis from Definition 1660.42. It exposes compact finite-net rows, Hausdorff-window rows, total-bounded and complete-metric support through the source row, and structural BEDC rows.

Next apply Definition 1660.40. Every compact-metric Vietoris read is forced through compact-source support, finite-net presentations, hit/miss ledgers, finite-cover ledgers, directed Hausdorff data, and the displayed net and Cauchy-uniform handoff.

Finally use Definition 1660.43. Since the public Hausdorff read is exhausted by finite dyadic windows and common refinements, the compact-metric consumer has no remaining coordinate from which to read host powersets, quotient compacta, selected limits, or open-cover compactness. □

Theorem 1660.45 (Hyperspace finite Vietoris subcover avoidance)**.** *For every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier* $H = (X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M)$*, every Vietoris hit-miss read used by a compact-metric consumer is a finite* BHist *route over the displayed rows* $K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M$*. The route may inspect finite hit rows, finite miss rows, compact-source cells from* X*, finite-cover ledgers internal to the subset nets, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

It has no coordinate for an arbitrary open cover of the hyperspace, a selected finite subcover, a host powerset compactness theorem, a quotient compactum, or a selected compact limit.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisFiniteSubcoverAvoidance`

Proof. Read the finite carrier basis from Definition 1660.42. The basis exposes the compact source, finite subset-net rows, Hausdorff-window rows, and structural rows before any Vietoris consumer can read hit or miss information.

Apply Definition 1660.40. Its obligation surface already lists hit rows, miss rows, finite-cover ledgers, directed Hausdorff data, net-convergence handoff, and Cauchy-uniform-net handoff as finite rows over the same packet.

Finally use Definition 1660.44. A compact-metric consumer receives only those displayed rows and their structural replay. Since no open-cover object or selected subcover occurs in the carrier, the Vietoris read remains a finite hit-miss route. □

Theorem 1660.46 (Hyperspace Cauchy-net compact handoff)**.** *Every Cauchy-net handoff from an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet to a compact-metric or completion-facing consumer factors through finite subset-net rows, the* $\mathsf{NetConvergence}^{\uparrow}$ *schedule readable from those rows, the* $\mathsf{CauchyUniformNet}^{\uparrow}$ *replay row, the* $\mathsf{CompleteMetric}^{\uparrow}$ *support carried by the compact source, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

The handoff may assert only that the displayed finite net route is available to the consumer. It does not select a limiting compact subset, prove open-cover compactness of the hyperspace, identify quotient compacta, or export an ambient complete-hyperspace object.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCauchyNetCompactHandoff`

Proof. Start with the accepted carrier and its finite basis. The rows K_0, K_1, N_0, N_1 display the finite subset nets, while D_0, D_1, R display the Hausdorff-window comparisons used to compare net stages.

Apply Definition 1660.41. The completion-facing read is already forced through $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, and $\mathsf{CauchyUniformNet}^{\uparrow}$ over the displayed finite-net carrier.

Use Definition 1660.44 to restrict the compact-facing consumer to the same rows. The structural rows H_s, C, P, M then transport, replay, source, and name the handoff, with no route to a selected compact limit or host complete hyperspace. □

Theorem 1660.47 (Hyperspace hit-miss directed refinement)**.** *Let H and H' be accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carriers whose compact-source rows and finite subset-net rows are related by componentwise* hsame *transport. If the hit rows of H' refine the hit rows of H and the miss rows of H' refine the miss rows of H by displayed finite witnesses, then every directed Hausdorff or Vietoris read of H' factors through the original finite hit-miss route of H together with the refinement witnesses.*

The refinement may replace a displayed finite row by a finer finite row, but it cannot add open-cover compactness, selected compact limits, host powersets, quotient compacta, or untracked distance objects.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace hit miss directed refinement`

Proof. Project both carriers to their finite bases. The hsame transport identifies the compact source, subset-net schedules, Hausdorff windows, and structural rows before the refinement witnesses are read.

The hit refinement sends every refined hit row to a displayed hit row in the original packet. The miss refinement does the same for finite avoidance rows, so a Vietoris read of H' returns to the hit-miss ledgers of H after inspecting only finite witnesses.

For directed Hausdorff reads, apply Definition 1660.43. The distance windows are exhausted by D_0, D_1, R and common finite refinements, and the structural rows replay the same route. Thus no excluded open-cover, host powerset, quotient, selected-limit, or untracked-distance coordinate is exported. □

Theorem 1660.48 (Hyperspace compact-source carrier obligations)**.** *Let H be an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier. Before any compact-source consumer can read a hyperspace row, the packet must expose the common* $\mathsf{CompactMetric}^{\uparrow}$ *source, the finite subset-net rows, the* $\mathsf{MetricDiameter}^{\uparrow}$ *and*

$\mathsf{TotallyBounded}^{\uparrow}$ *resolution rows, the* $\mathsf{CompleteMetric}^{\uparrow}$ *support carried by the source, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row.*

The exposed surface is a compact-source carrier for finite hyperspace observations only. It does not export a host powerset, quotient compactum, selected compact subset, selected limit, open-cover compactness theorem, or ambient complete-hyperspace object.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompactSourceCarrierObligations`

Proof. Project the accepted carrier. Its compact source is already the $\mathsf{CompactMetric}^{\uparrow}$ row, and the finite subset-net rows are the only hyperspace objects that downstream consumers can inspect.

Apply Definition 1660.44. The compact-facing handoff is forced through the displayed compact metric source, finite subset nets, and structural rows before any consumer receives hyperspace data.

Now use Definition 1660.46. The diameter, total-boundedness, complete-metric, Cauchy-uniform, replay, provenance, and naming rows are the same finite route. Since none of the forbidden host objects is a carrier coordinate, the compact-source obligation surface is exhausted by the displayed packet. □

Theorem 1660.49 (Hyperspace Vietoris root-unblock obligations)**.** *Every Vietoris hit-miss read from an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet factors through finite hit rows, finite miss rows, finite subset-net schedules, Hausdorff-window comparisons, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. Refinement of a Vietoris read is allowed only by replacing displayed finite hit or miss witnesses with finer displayed finite witnesses.*

Thus a Vietoris root consumer receives a finite hit-miss classifier and not a host powerset topology, quotient compactum, selected compact limit, open-cover compactness proof, or untracked distance object.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisRootUnblockObligations`

Proof. Start from the hit and miss rows of the accepted carrier. These rows are finite ledgers, so a Vietoris request can only inspect listed hit witnesses and listed avoidance witnesses.

Apply Definition 1660.39 and Definition 1660.47. Stability keeps the classifier on the displayed hit-miss route, while directed refinement replaces rows only by explicitly displayed finite refinements.

Finally apply Definition 1660.43. The Hausdorff side is already bounded by the finite distance windows and common refinements, so the Vietoris classifier cannot escape to host powersets, quotient compacta, selected limits, open-cover compactness, or untracked distance data. □

Theorem 1660.50 (Hyperspace completion-boundary obligations)**.** *Any completion-boundary read exported by an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet must pass through the finite subset-net carrier, the* $\mathsf{NetConvergence}^{\uparrow}$ *schedule, the* $\mathsf{CauchyUniformNet}^{\uparrow}$ *replay row, the* $\mathsf{CompleteMetric}^{\uparrow}$ *support inherited from the compact source, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row before a boundary-facing consumer receives data.*

The boundary handoff records only this finite Cauchy-uniform completion route. It does not select a limiting compact subset, identify quotient compacta, assert open-cover compactness, import a host powerset, or expose an ambient complete-hyperspace object.

Proof. Read the finite subset-net rows first. They provide the only net stages visible to the packet, and their schedules are the source for the $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{CauchyUniformNet}^{\uparrow}$ rows.

Use Definition 1660.41. The completion-facing read is forced through the displayed complete-metric support, net-convergence schedule, Cauchy-uniform replay, and Hausdorff-window carrier.

Then apply Definition 1660.46. The compact-source and completion-boundary routes are the same finite ledger route, so no selected limit, quotient compactum, host powerset, open-cover compactness proof, or ambient complete-hyperspace object is exported. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceCompletionBoundaryObligations`

Theorem 1660.51 (Hyperspace root Vietoris net basis)**.** *Every root Vietoris read from an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must expose the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedules carried by* K_0, K_1, N_0, N_1*, the finite hit and miss ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. The resulting basis is a finite hit/miss window over displayed nets, not a host powerset topology, open-cover compactness proof, selected subcover, quotient compactum, or selected compact limit.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootVietorisNetBasis`

Proof. Project the accepted carrier and its finite basis. The only compact-subset rows available to a Vietoris consumer are the compact source and the displayed finite subset-net schedules.

Apply Definition 1660.49. It routes every Vietoris hit-miss read through finite hit rows, finite miss rows, finite subset-net schedules, Hausdorff-window comparisons, and the structural rows.

Use Definition 1660.47 to account for refinement. Refinement can replace a displayed finite witness by a finer displayed finite witness, but it adds no open-cover compactness, selected subcover, host powerset, quotient compactum, or selected limit. □

Theorem 1660.52 (Hyperspace root Hausdorff finite window)**.** *Every root Hausdorff comparison exported by an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must pass through the* $\mathsf{Met}^{\uparrow}$ *distance witnesses inside the compact source, the terminal* $\mathsf{Real}^{\uparrow}$ *tolerance readback row* R*, and the finite net rows* $K_0, K_1, N_0, N_1, D_0, D_1$ *before any downstream distance consumer receives data. The comparison is a finite directed-window route; it is not a host Hausdorff metric on a powerset, quotient hyperspace equality, selected nearest-point operation, open-cover compactness, or ambient complete-hyperspace claim.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootHausdorffFiniteWindow`

Proof. Start from Definition 1660.43. It exhausts public Hausdorff reads by the displayed dyadic distance windows D_0, D_1, R and common refinements of the finite subset-net rows.

Apply Definition 1660.35. The comparison transports along the displayed $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ classifier rows, while the metric and Real tolerance readback stay inside the carrier.

Finally use Definition 1660.44. A downstream compact-metric consumer receives only compact finite-net rows, total-bounded schedules, complete-metric support, metric-distance windows, and structural rows, so no quotient equality, selected point, host powerset metric, open-cover proof, or ambient completion is exported. □

Theorem 1660.53 (Hyperspace root complete and totally bounded handoff)**.** *Any compact-hyperspace or completion-facing consumer of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet must read the* $\mathsf{CompleteMetric}^{\uparrow}$ *support and* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net rows carried by the compact source before making a compact-hyperspace claim. The handoff may then consume the finite subset-net rows, directed Hausdorff ledgers,* $\mathsf{NetConvergence}^{\uparrow}$ *schedule,* $\mathsf{CauchyUniformNet}^{\uparrow}$ *replay, componentwise transport, displayed*

replay, provenance, and local naming; it does not assert open-cover compactness, selected compact limits, quotient hyperspace equality, or an ambient complete hyperspace.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRootCompleteTotallyBoundedHandoff`

Proof. Read the compact-source carrier obligations in Definition 1660.48. They force the common $\mathsf{CompactMetric}^{\uparrow}$ source, finite subset-net rows, total-bounded resolution, complete-metric support, and structural rows to be visible before a compact-source consumer receives hyperspace data.

Next apply Definition 1660.50. A completion-boundary read must pass through the finite subset-net carrier, $\mathsf{NetConvergence}^{\uparrow}$ schedule, $\mathsf{CauchyUniformNet}^{\uparrow}$ replay, $\mathsf{CompleteMetric}^{\uparrow}$ support, and the structural rows.

Finally use Definition 1660.46. The compact-source and completion-boundary routes are the same finite ledger route, so the consumer receives only the displayed complete and total-bounded rows before any compact-hyperspace claim. □

Theorem 1660.54 (Hyperspace Vietoris refinement factorization)**.** *Every accepted* $\mathsf{Hyperspace}^{\uparrow}$ *Vietoris refinement factors through the displayed compact source row, finite subset-net schedules, hit and miss rows, directed Hausdorff ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. The factorization refines only finite displayed witnesses; it is not a host powerset refinement, selected subcover, quotient compactum, or open-cover compactness argument.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisRefinementFactorization`

Proof. Start from the finite Vietoris obligation basis of Definition 1660.49. It routes every hit-miss read through the listed finite hit rows, finite miss rows, subset-net schedules, Hausdorff-window comparisons, and structural rows.

Then apply the root Vietoris net basis of Definition 1660.51. The compact source and $\mathsf{TotallyBounded}^{\uparrow}$ schedules remain visible before a root consumer receives a Vietoris basis element.

Finally use directed refinement from Definition 1660.47. Refinement replaces a displayed finite witness by a finer displayed finite witness and therefore cannot import host powersets, selected subcovers, quotient compacta, or open-cover compactness. □

Theorem 1660.55 (Hyperspace Vietoris union stability)**.** *The finite union of two accepted finite Vietoris* $\mathsf{Hyperspace}^{\uparrow}$ *packets is routed through the union of their displayed hit rows and the intersection of their displayed miss rows. The resulting packet still reads only the compact source row, finite subset-net schedules, directed Hausdorff ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row; it does not import a host powerset union, open-cover compactness, selected subcover, quotient compactum, or selected compact limit.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisUnionStability`

Proof. Project the two accepted finite Vietoris packets. Their hit rows and miss rows are already exhausted by Definition 1660.49 and by the finite-net carrier data of Definition 1660.1.

Use Definition 1660.54. Directed refinement replaces displayed finite witnesses by displayed finer witnesses, so the union packet can combine hit evidence by finite union and miss evidence by common intersection without leaving the recorded net schedules.

Finally apply Definition 1660.47. The common refinement remains a finite displayed witness over the same Hausdorff ledgers and structural rows, and the excluded host, quotient, selected-limit, and open-cover data have no carrier coordinate. □

Theorem 1660.56 (Hyperspace Hausdorff union boundary)**.** *Every Hausdorff read of the finite union of two accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packets is bounded only by the displayed finite subset-net rows and directed coverage ledgers of the two packets. A downstream consumer may inspect the union through the common compact source, the net rows* $K_0, K_1, N_0, N_1, D_0, D_1$*, the terminal* $\mathsf{Real}^{\uparrow}$ *tolerance readback, and the structural rows, but it receives no host powerset metric, quotient hyperspace equality, selected nearest-point operation, open-cover compactness, or ambient complete-hyperspace object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceHausdorffUnionBoundary`

Proof. Start from the Hausdorff classifier transport of Definition 1660.52. It routes every comparison through the displayed compact source, finite net rows, directed Hausdorff ledgers, and Real tolerance readback.

Apply the finite-net exhaustion used by Definition 1660.53. The union read can only combine the two displayed net families and their coverage ledgers; it cannot select a new host-level compact subset.

Then use the Vietoris union stability theorem above. Since the union packet is already expressed through displayed hit, miss, net, and refinement rows, the Hausdorff read is bounded by those same rows and exports none of the excluded host or quotient data. □

Closure status for $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Hyperspace}^{\uparrow}$ packet is generated from a CompactMetricUp source, two finite compact-subset net presentations, two net-resolution ledgers, two directed Hausdorff-distance coverage ledgers, a dyadic and Real tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1660.1, Definition 1660.3, the root obligation rows of Definition 1660.59, the scoped lattice of Definition 1660.65, the public export witness of Definition 1660.66, the bridged interface route of Definition 1660.67, and the mature envelope of Definition 1660.70 close the paper-side compact-subset, directed Hausdorff, Vietoris-window, completion-facing, transport, replay, provenance, local naming, public-object, bridged-interface, and mature compact-metric Vietoris surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host powersets, quotient compacta, selected limits, open-cover compactness, choice-selected representatives, Lean verification, or bridge checking.

Upgrade path: Lean verification and bridge checking require a checked target for the public compact-source, finite-net, directed Hausdorff, Vietoris, and completion-facing Hyperspace interface.

Theorem 1660.57 (Hyperspace compact-subset root admission)**.** *Every root consumer admitted by an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *carrier must first read compact-source admission through Definition 1660.1 and the local obligation rows of Definition 1660.3. The compact-subset request is therefore represented by the concrete BEDC rows* X, K_0, K_1, N_0, N_1 *together with finite subset-net schedules, tolerance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Hyperspace}^{\uparrow}}$ *row. It is not a host powerset element, quotient compactum, selected compact limit, open-cover compactness proof, or choice-selected representative.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace compact subset root admission`

Proof. Project the carrier. The source row X supplies the displayed compact metric surface, while K_0, K_1, N_0, N_1 provide the finite compact-subset presentations and resolution schedules.

Apply Definition 1660.3. Its compact-source and finite subset-net clauses force every root consumer to read the subset request through those displayed rows before any Hausdorff or Vietoris data is public.

The remaining rows R, H_s, C, P, M account for tolerance readback, transport, replay, provenance, and local naming. Since the excluded host and quotient objects are absent from the carrier, compact-subset admission is exactly the concrete BEDC row surface listed above. □

Theorem 1660.58 (Hyperspace directed Hausdorff root coverage)**.** *Every directed Hausdorff root read exported by an accepted* Hyperspace$^{\uparrow}$ *packet is covered by the two displayed ledgers D_0 and D_1, the finite net rows K_0, K_1, N_0, N_1, and the tolerance readback row R. The coverage must be bidirectional before a symmetric Hausdorff-distance consumer receives data, and the read remains transport-stable through H_s, replayed through C, packaged by P, and named by M. No directed coverage row may be replaced by a selected nearest point, quotient compact-subset equality, host Hausdorff metric, open-cover compactness, or ambient complete-hyperspace object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace directed hausdorff root coverage`

Proof. Use Definition 1660.6. It routes each displayed hit through the compact source and the finite subset-net schedules before the tolerance row can be consumed.

Apply Definition 1660.7. That theorem requires both D_0 and D_1 before a symmetric Hausdorff read is public, so a one-sided host selection cannot stand in for the missing direction.

Finally use the transport and provenance clauses of Definition 1660.3. The rows H_s, C, P, M preserve the same finite coverage route and add no selected point, quotient equality, host metric, open-cover proof, or ambient completion coordinate. □

Theorem 1660.59 (Hyperspace finite Hausdorff root obligations)**.** *The finite Hausdorff root surface of* Hyperspace$^{\uparrow}$ *is exhausted by compact-source admission, finite subset-net schedules, bidirectional directed Hausdorff coverage, tolerance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *rows carried by*

$$X, K_0, K_1, N_0, N_1, D_0, D_1, R, H_s, C, P, M.$$

Equivalently, the root surface cited by Definition 1660.12 cannot be strengthened by host powersets, quotient compacta, selected limits, open-cover compactness, selected nearest points, or choice-selected representatives.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceFiniteHausdorffRootObligations`

Proof. Begin with Definition 1660.12. It identifies the public Hausdorff read as the two directed finite ledgers together with the tolerance row over the displayed compact source and net schedules.

Use Definition 1660.57 to account for the root carrier and finite subset-net admission, and use Definition 1660.58 for the bidirectional directed coverage and tolerance readback.

The structural rows are then fixed by Definition 1660.3. Componentwise transport, continuation replay, provenance, and local naming are the only remaining public rows, so the excluded host and quotient objects cannot add a further root obligation. □

Theorem 1660.60 (Hyperspace Vietoris finite-window obligations)**.** *Every Vietoris-style finite-window read exported by an accepted* Hyperspace$^{\uparrow}$ *packet must pass through compact-source admission, subset-net schedules, directed Hausdorff coverage, tolerance readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before a compact-subset consumer reads hit or miss data. The read is therefore exhausted by the same finite root surface as Definition 1660.59; it does not export a host Vietoris topology, arbitrary open-cover compactness, quotient compact-subset equality, selected nearest points, or an ambient complete hyperspace.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisFiniteWindowObligations`

Proof. Start with Definition 1660.57. It forces a compact-subset request to enter through the compact source and the finite subset-net schedules before any hit or miss observation is available.

Use Definition 1660.58. The hit and miss reads are routed through the two directed finite coverage ledgers and the shared tolerance readback row, so they cannot be supplied by a host Vietoris topology or by selected nearest points.

Finally apply Definition 1660.59. The remaining public rows are exactly componentwise transport, displayed replay, provenance, and local naming, which repack the same finite window without adding quotient compacta, open-cover compactness, or an ambient complete-hyperspace object. □

Theorem 1660.61 (Hyperspace Vietoris handoff)**.** *Every Vietoris-topology or Hausdorff-distance consumer of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet must first read the compact source row, the finite subset-net schedules, both directed coverage ledgers, and the tolerance readback row. In the notation of Definition 1660.1, the route is*

$$X, K_0, K_1, N_0, N_1, D_0, D_1, R \longmapsto H_s, C, P, M.$$

Thus the downstream interfaces of chapter 2375 and chapter 3653 consume the same compact-source and finite-net boundary supplied by chapter 129. When hyperspace packets are read through metric distance-to-closed-set witnesses, the Vietoris finite-hit and finite-miss surface also consumes the distance-function embedding route of chapter 7569; the consumers receive no host powerset topology, quotient compactum, selected compact limit, open-cover compactness proof, or untracked Hausdorff metric.

Proof. Project the accepted $\mathsf{Hyperspace}^{\uparrow}$ carrier. The source row X is the only compact metric source, and the finite subset-net schedules are displayed by K_0, K_1, N_0, N_1 before any directed distance or topology read is public.

Use the Hausdorff-distance ledger stability and finite Hausdorff carrier theorems of Definition 1660.2 and Definition 1660.12. They force the two directed coverage ledgers D_0, D_1 and the tolerance readback row R to be present before a symmetric Hausdorff consumer can read a distance value.

For Vietoris consumers, use the root Vietoris net basis and refinement factorization in Definition 1660.51 and Definition 1660.54. Those results route hit, miss, and refinement reads through the same compact source and finite subset-net schedules.

Both public reads therefore have the same source lock. The Vietoris-topology read cannot expose a hit or miss neighborhood until it has consumed the compact-metric source row X, the finite subset-net schedules K_0, K_1, N_0, N_1, and the directed coverage ledgers D_0, D_1 that witness how the scheduled finite subsets cover the compact source.

The Hausdorff-distance read is locked by the same rows. Before a symmetric distance value is public, the two directed coverage ledgers must already be visible over the displayed compact-metric source row and the finite subset-net schedules; otherwise the row would be an untracked host metric rather than the displayed Hausdorff-distance consumer of the hyperspace packet.

The structural rows H_s, C, P, M only transport, replay, source, and name this displayed packet. Since no forbidden host powerset, quotient compactum, selected compact limit, open-cover compactness proof, or untracked Hausdorff metric appears in the carrier, every Vietoris or Hausdorff consumer must use the stated handoff. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.Hyperspace vietoris handoff`

Lemma 1660.62 (Vietoris located-real tolerance dependency)**.** *Every Vietoris-topology or Hausdorff-distance consumer of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet that reads tolerance comparisons must route the tolerance through* $\mathsf{LocatedReal}^{\uparrow}$ *or the RealUp located-tolerance ledger before reading the symmetric Hausdorff value. In the notation of Definition 1660.1, the tolerance readback row* R *is therefore downstream of the compact source, finite subset-net schedules, and directed coverage ledgers named by Definition 1660.61, and upstream of the Hausdorff read consumed by chapter 3653.*

Proof. Project the accepted $\mathsf{Hyperspace}^{\uparrow}$ carrier. The compact source and finite subset-net schedules are the rows X, K_0, K_1, N_0, N_1, while D_0, D_1 record the directed coverage ledgers.

Apply Definition 1660.61. It forces both Vietoris and Hausdorff consumers to expose those rows before the tolerance readback row R becomes available.

The tolerance comparison itself must then be read through the displayed $\mathsf{LocatedReal}^{\uparrow}$ surface of chapter 1933 or through the located-tolerance ledger sealed by chapter 38. Since the carrier has no separate coordinate for an untracked Hausdorff metric or a host real comparison, the symmetric Hausdorff value can only be consumed after that located tolerance route is visible. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisLocatedRealDependency`

Vietoris–Hausdorff dependency route. The $\mathsf{Hyperspace}^{\uparrow}$ packet reads closed and compact subobjects through the $\mathsf{VietorisTopology}^{\uparrow}$ hit–miss surface of chapter 2375 and the $\mathsf{HausdorffDistance}^{\uparrow}$ scalar comparison surface of chapter 3653 before any compact-metric consumer inspects the hyperspace carrier. Thus the compact-subset route is not a kernel-only float: it exposes finite hit windows, finite miss windows, bidirectional Hausdorff ledgers, and the shared tolerance readback as the horizontal dependency surface between $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$.

Compact-metric dependency route. The Vietoris handoff of Definition 1660.61 is consumed only after the compact-source row of chapter 129 has exposed its finite-net and completeness support. Thus every $\mathsf{Hyperspace}^{\uparrow}$ route to Vietoris or Hausdorff data still factors through the displayed compact metric source, the finite subset-net schedules, the directed coverage ledgers, and the tolerance readback row; it receives no host powerset topology, selected compact subset, quotient compactum, or open-cover compactness proof outside the $\mathsf{CompactMetric}^{\uparrow}$ dependency.

Compact-metric Hausdorff dependency route. The $\mathsf{Hyperspace}^{\uparrow}$ packet exposes Vietoris and Hausdorff-distance consumers only after the compact source has passed through the $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{HausdorffDistance}^{\uparrow}$ rows. Thus a consumer may read hyperspace data only after the metric carrier, completeness witness, finite total-bounded schedules, compact-metric handoff, and Hausdorff comparison ledger have been displayed; it receives no host powerset topology, selected compact subset, quotient compactum, or open-cover compactness proof outside that dependency route.

Theorem 1660.63 (Hyperspace RealSeparability forward route)**.** *Every* $\mathsf{Hyperspace}^{\uparrow}$ *finite-net read over a Real-facing compact metric source may consume chapter 1809 only through the displayed density route*

$$\mathsf{Rat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The route exposes rational density windows, dyadic tolerance comparisons, stream-name schedules, regular rational readback, and the final real-name seal before any compact-subset finite net is consumed. It exports no host powerset, selected nearest point, quotient compactum, choice-selected dense enumeration, or ambient compactness proof.

Proof. Project the $\mathsf{Hyperspace}^{\uparrow}$ carrier through Definition 1660.1. A Real-facing finite-net read can inspect only the compact metric source row, the displayed subset-net schedules, the directed coverage ledgers, the tolerance readback, and the visible transport, replay, package, and naming rows.

Now route the source-side density request through chapter 1809. The rational windows supply the $\mathsf{Rat}^{\uparrow}$ row, the tolerance comparisons pass through $\mathsf{DyadicRatCore}^{\uparrow}$, the finite read schedule is a $\mathsf{StreamName}^{\uparrow}$ window, and the readback is a $\mathsf{RegSeqRat}^{\uparrow}$ row before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Repack those density rows with the hyperspace finite-net ledgers. Since neither carrier contains a host powerset, selected nearest point, quotient compactum, choice-selected dense enumeration, or ambient compactness proof, the forward route is exactly the displayed Real-separability dependency stack. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceRealSeparabilityForwardRoute`

Uniform-completion reflector forward link. The $\mathsf{Hyperspace}^{\uparrow}$ packet reads closed-set and compact-subset carriers through the Vietoris hit–miss route of chapter 2375, the Hausdorff metric comparison route of chapter 2086, and the compact metric source route of chapter 129 before any $\mathsf{UniformCompletionReflector}^{\uparrow}$ consumer can view the hyperspace as a completed uniform object. Thus completion-facing hyperspace data remains a displayed dependency lattice over finite Vietoris windows, Hausdorff comparison rows, compact-metric schedules, and the reflector packet of chapter 6282, with no host powerset topology, selected compact subset, quotient compactum, or ambient completion object.

1660.1 Downstream Dependency Lattice

Theorem 1660.64 (Hyperspace Vietoris downstream dependency)**.** *Every* $\mathsf{VietorisTopology}^{\uparrow}$ *or* $\mathsf{HausdorffDistance}^{\uparrow}$ *downstream read of an accepted* $\mathsf{Hyperspace}^{\uparrow}$ *packet consumes the compact source, finite subset-net schedules, and directed coverage ledgers of Definition 1660.61 before it may expose a topology row or distance row. In particular, the interfaces of chapter 2375 and chapter 3653 both pass through the compact-metric boundary of chapter 129; neither consumer receives a host powerset topology, quotient compactum, selected compact limit, open-cover compactness proof, or untracked Hausdorff metric.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisDownstreamDependency`

Proof. Read the handoff theorem Definition 1660.61. It forces any Vietoris-topology or Hausdorff-distance consumer to open the compact source row, finite subset-net schedules, directed coverage ledgers, and tolerance readback row before the structural transport, replay, provenance, and naming rows are public.

For a $\mathsf{VietorisTopology}^{\uparrow}$ consumer, the topology row is therefore downstream from the finite hit, miss, and refinement data already supplied by the compact source and subset-net schedules. It cannot be exposed as a host powerset topology independent of those rows.

For a $\mathsf{HausdorffDistance}^{\uparrow}$ consumer, the distance row is downstream from the two directed coverage ledgers and the same tolerance readback boundary. It cannot be exposed as an untracked metric, selected compact limit, quotient compactum, or open-cover compactness proof.

Thus the compact-metric dependency of chapter 129 sits below both downstream consumers in the same lattice: compact source first, finite nets and directed coverage next, topology or distance rows only after that displayed dependency path. □

Theorem 1660.65 (Hyperspace scoped closure route)**.** *Every accepted* Hyperspace$^{\uparrow}$ *consumer that reaches a compact-subset, directed Hausdorff, Vietoris, or completion-facing read is exhausted by the displayed carrier rows of Definition 1660.1, the compact source route through chapter 129, the metric-distance witnesses of chapter 57, the Hausdorff comparison route of chapter 2086, the Vietoris handoff of chapter 2375, and the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already named by the packet.*

Thus the scoped Hyperspace$^{\uparrow}$ *surface is the finite dependency lattice over compact-source admission, finite subset-net schedules, bidirectional Hausdorff windows, Vietoris hit–miss windows, completion-boundary handoffs, componentwise transport, displayed replay, provenance, and local naming. It exports no host powerset topology, selected compact subset, quotient compactum equality, open-cover compactness proof, selected nearest point, choice-selected subcover, or ambient complete-hyperspace object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceScopedClosureRoute`

Proof. Project the accepted packet through Definition 1660.1. The compact source, finite subset-net rows, directed Hausdorff ledgers, tolerance readback, transport, replay, provenance, and local naming rows are the only coordinates available to a consumer.

Use Definition 1660.57, Definition 1660.59, and Definition 1660.60. These root obligations force compact-subset, Hausdorff, and Vietoris reads to pass through the same finite carrier rows before any downstream surface is public.

For completion-facing consumers, apply Definition 1660.50, Definition 1660.53, and Definition 1660.46. They keep the completion route inside the displayed compact-source, net-convergence, complete-metric, Cauchy-uniform, and structural rows.

The sibling lattice citations then identify the external reads: CompactMetric supplies compact finite-net admission, Metric supplies distance witnesses, HausdorffMetric supplies the comparison route, and Vietoris supplies hit–miss consumption. Because the carrier contains no excluded host object or selected witness coordinate, the named finite lattice is the whole scoped surface. □

Theorem 1660.66 (Hyperspace public certificate route)**.** *The public* Hyperspace$^{\uparrow}$ *interface exports exactly the compact-metric source rows, finite compact-subset net schedules, directed Hausdorff-distance and Vietoris handoff rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming already visible in the scoped packet. A public reader may consume the route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{HausdorffDistance}^{\uparrow},\quad \mathsf{VietorisTopology}^{\uparrow},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

only through those displayed rows, and receives no quotient hyperspace, host powerset topology, selected compact representative, selected nearest point, open-cover compactness proof, or ambient complete-hyperspace object.

Proof. Project the accepted Hyperspace$^{\uparrow}$ packet through Definition 1660.1. The compact source, finite subset-net schedules, directed coverage ledgers, tolerance readback, transport, replay, provenance, and local naming rows are the only coordinates available to a public consumer.

Use Definition 1660.65. It already routes compact-subset, directed Hausdorff, Vietoris, and completion-facing reads through the same compact-source and finite-net dependency lattice before any downstream surface is public.

Finally apply the downstream dependency and Vietoris handoff theorems, Definition 1660.64 and Definition 1660.61. They identify the public Hausdorff-distance and Vietoris handoff rows with

the displayed finite ledgers, so the excluded quotient, host-topological, selected-representative, and open-cover objects have no public coordinate. □

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspacePublicCertificate`

Theorem 1660.67 (Hyperspace bridged interface route)**.** *The public* Hyperspace$^{\uparrow}$ *certificate of Definition 1660.66 is bridged by the displayed compact-metric finite-net rows, Hausdorff metric rows, Vietoris handoff rows, and completion-facing dependency routes already present in the hyperspace packet. More precisely, every bridged reader must pass through the compact-source route of chapter 129, the Hausdorff comparison route of chapter 2086, the Vietoris handoff of Definition 1660.61, the scoped lattice of Definition 1660.65, and the completion-facing handoff of Definition 1660.53 before consuming the public interface. The bridge exports no host powerset, quotient compactum, selected limit, open-cover compactness proof, selected nearest point, or ambient complete-hyperspace object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceBridgedInterfaceRoute`

Proof. Start with the public certificate route. It exposes only the compact-metric source rows, finite compact-subset schedules, directed Hausdorff ledgers, Vietoris handoff rows, structural transport, replay, provenance, and local naming rows.

The compact-metric and Hausdorff rows are then read through the sibling dependency files and the scoped closure route. These force finite-net schedules, bidirectional Hausdorff windows, and metric witnesses to remain visible before any public hyperspace reader can inspect compact-subset data.

For topology and completion-facing consumers, apply the Vietoris handoff and the root complete–totally-bounded handoff. Both routes reuse the same displayed compact source, finite net rows, directed Hausdorff ledgers, completion boundary, transport, replay, provenance, and naming rows. Hence the public certificate has a bridged interface using only the rows already present in the chapter. □

Theorem 1660.68 (Hyperspace bridged route formal target)**.** *The paper bridge of Definition 1660.67 is aimed at the checked public certificate target*

BEDC.Derived.HyperspaceUp.HyperspacePublicCertificate.

This ledger row records the formal target for the compact-metric, Hausdorff, Vietoris, and completion-facing public route while keeping the bridge-status claim separate from Lean bridge checking.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceBridgedRouteFormalTarget`

Proof. Use Definition 1660.67. Its route already factors every public reader through the compact-metric, Hausdorff, Vietoris, scoped-lattice, and completion-facing rows.

The named Lean target is the public certificate theorem for that same finite route. Recording it here identifies the formal endpoint of the paper bridge without asserting a checked standard bridge or changing the chapter closure block. □

Theorem 1660.69 (Compact-metric Vietoris maturity route)**.** *Every mature* Hyperspace$^{\uparrow}$ *Vietoris or Hausdorff-distance consumer first reads* CompactMetric$^{\uparrow}$ *finite-net data,* HausdorffMetric$^{\uparrow}$ *distance rows, and* KuratowskiHyperspace$^{\uparrow}$ *closure rows before it exports compact-subset data. More precisely, the consumer route factors through the compact-metric dependency route of chapter 129, the Vietoris handoff of Definition 1660.61, the compact-metric Vietoris unblock of Definition 1660.30,*

the bridged interface route of Definition 1660.67, the Hausdorff metric chapter of chapter 2086, and the Kuratowski hyperspace chapter of chapter 5944. The route exports no host powerset, quotient compactum, selected nearest point, open-cover compactness proof, or ambient complete hyperspace object.

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceMatureCompactMetricVietorisRoute`

Proof. Start with the compact-metric dependency route. It forces every Vietoris or Hausdorff-distance consumer to expose the compact source, finite-net schedules, directed coverage ledgers, and tolerance readback before the local $\mathsf{Hyperspace}^{\uparrow}$ rows are inspected.

Next apply the Vietoris handoff and the compact-metric Vietoris unblock. These routes identify the hit–miss rows and finite compact-subset windows that may be consumed, while keeping the compact-source row visible through chapter 129.

Finally apply the bridged interface route. The Hausdorff metric rows of chapter 2086 and the Kuratowski closure rows of chapter 5944 are read before any compact-subset export, so the mature consumer remains a finite dependency route rather than a host powerset or completion principle. □

Theorem 1660.70 (Hyperspace Vietoris mature envelope)**.** *The compact-metric finite-net row, Hausdorff metric row, Kuratowski hyperspace row, and Vietoris handoff together form the mature exported envelope for* $\mathsf{Hyperspace}^{\uparrow}$*. Every top-window consumer of compact-subset data must read those four rows after the local carrier and before any public compact-subset export, and the route exports no host powerset, quotient compactum, selected nearest point, open-cover compactness proof, or ambient complete hyperspace object.*

Lean4 ✓ `BEDC.Derived.HyperspaceUp.HyperspaceVietorisMatureEnvelope`

Proof. Apply the compact-metric Vietoris maturity route of Definition 1660.69. It already forces Vietoris and Hausdorff-distance consumers through compact-metric finite-net data before any compact-subset export.

Read the two sibling rows next. The Hausdorff metric chapter supplies the finite distance ledger, and the Kuratowski hyperspace chapter supplies the closure-facing row consumed by the same compact-source route.

Finally use the Vietoris handoff of Definition 1660.61. The handoff keeps hit–miss and compact-subset window data attached to the displayed finite-net route, so the four rows jointly form the mature envelope without adding any refused host object. □

Chapter 1661

A Concrete Naming Certificate for $\mathsf{LocatedCauchyFilter}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Kelley filterbase terminology, the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ packet records the filter-level located witness surface used by the L10 completion route before any quotient filter, selected limit, countable-choice selector, host equality, or $\mathsf{Real}^{\uparrow}$ completeness principle is available. It sits between the Cauchy-filter carrier of chapter 333, the basis-level witness rows of chapter 1501, the regular filter route of chapter 1445, and the sequence-level located witness of chapter 1595. Its finite readback path uses the stream windows of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the downstream real-name seal of chapter 38.

Definition 1661.1 (Located Cauchy filter carrier)**.** *A* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, B, R, S, Q, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row naming the accepted Cauchy filter surface.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row that exposes finite basis neighborhoods and refinement witnesses.*
3. *R is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *row carrying the regular tail budget attached to the basis surface.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by sequence-facing consumers.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for located ball comparisons.*
7. *T is the located tail row, recording the displayed witness that the filter basis eventually meets the requested dyadic ball.*
8. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, reachable only after F, B, R, S, Q, D, T have been consumed.*
9. *H records componentwise* hsame *transport for the filter, basis, regularity, window, readback, dyadic, located-tail, seal, continuation, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a located precision request through $F, B, R, S, Q, D, T, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient filters, selected limits, countable choice, hidden infinite basis inspection, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterTasteGate single carrier alignment`

Theorem 1661.2 (Located Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *carrier* $L = (F, B, R, S, Q, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Cauchy filter row, basis row, regular filter row, finite-window row, rational readback row, dyadic tolerance ledger, located-tail row, seal boundary, transport, continuation, provenance, and local naming row;*

2. *basis exposure, requiring every located request to read the displayed basis row B only after the Cauchy filter row F has been fixed;*

3. *regular-tail exactness, requiring the regular filter row R and located-tail row T to share the same basis request and dyadic tolerance ledger D;*

4. *sequence handoff compatibility, requiring S and Q to expose only finite windows and regular rational readbacks generated from the same located filter request;*

5. *transport replay, requiring H and C to preserve the route through F, B, R, S, Q, D, T, E under componentwise* hsame *and displayed continuation;*

6. *real-seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$*-facing read to pass through the filter, basis, regularity, window, readback, dyadic, and located-tail rows before it reaches E;*

7. *provenance and name exhaustion, requiring P and N to name only this finite route and to export no quotient filters, selected limits, countable choice, hidden basis inspection, host equality, or ambient completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterNameCertObligations`

Proof. Carrier admission is projection from Definition 1661.1. The visible analytic rows are precisely the Cauchy filter, basis, regularity, stream-window, rational readback, dyadic tolerance, located-tail, and downstream seal rows.

For basis exposure, replay a located precision request through C. The request first fixes the Cauchy filter row F, then reads the finite basis neighborhoods in B; no sequence window or real-facing seal is available before this basis route is displayed.

Regular-tail exactness is read from R, D, T. The regular row R records the tail budget over the same basis request, the dyadic ledger D supplies the requested ball data, and the located-tail row T witnesses that the basis eventually meets that displayed ball.

The sequence handoff uses only finite data. The rows S and Q expose the requested windows and regular rational readbacks downstream of F, B, R, D, T, so they cannot introduce hidden infinite basis inspection or a selected limit.

Finally, H and C preserve the same ordered route under transport and replay. The seal boundary E appears only after F, B, R, S, Q, D, T, and the provenance and name rows P, N have no coordinates for quotient filters, selected limits, countable choice, host equality, or ambient completeness. □

Theorem 1661.3 (Located Cauchy filter regular tail location)**.** *Let $L = (F, B, R, S, Q, D, T, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *carrier. Every regular-tail request in R factors through the finite located route*

$$F, B, D \longmapsto T \longmapsto R \longmapsto S, Q, H, C, P, N.$$

Thus the regular filter consumer receives a displayed Cauchy filter, a finite basis neighborhood, a dyadic tolerance ledger, and a located-tail witness before any sequence-facing readback or real-facing seal can be inspected.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterRegularTailLocation`

Proof. Start with a regular-tail request. By Definition 1661.1, the row R is packaged with the Cauchy filter row F, the basis row B, the dyadic ledger D, and the located-tail row T.

Apply the basis exposure and regular-tail exactness clauses of Definition 1661.2. The request fixes F and B, then reads the dyadic ball data in D, and only then consumes the located witness T attached to that same basis request.

The remaining rows are downstream support. The sequence windows S, rational readbacks Q, transport H, replay C, provenance P, and local name N preserve and package the same finite route; none of them adds a quotient filter, selected limit, countable-choice selector, hidden basis inspection, host equality, or ambient completeness witness. □

Theorem 1661.4 (Located Cauchy filter basis-refinement route)**.** *Every located-tail read T in an accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *packet*

$$L = (F, B, R, S, Q, D, T, E, H, C, P, N)$$

first fixes the $\mathsf{CauchyFilter}^{\uparrow}$ *row F, then refines through the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row B at the displayed dyadic tolerance D, and only then exposes the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *row R,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterBasisRefinementRoute`

Proof. Project the carrier. The located-tail coordinate T is packaged with exactly one Cauchy filter row F, one basis row B, one dyadic tolerance ledger D, and the downstream rows R, S, Q, E, H, C, P, N.

Use the basis exposure clause of Definition 1661.2. A located request fixes F before it may inspect the displayed basis row B.

Use the regular-tail exactness clause of the same theorem. The basis refinement is read at the dyadic tolerance recorded in D, and the located-tail row T shares that displayed request before R, S, Q or E are available.

The remaining rows only preserve or consume the finite route. Hence quotient filters, selected limits, countable choice, hidden basis inspection, host equality, and ambient completeness are absent from the refinement path. □

Theorem 1661.5 (Located Cauchy filter choice-free basis)**.** *Every accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *read factors through the displayed filter row F, basis row B, regular tail ledger R,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows S, Q, dyadic located-tail rows D, T, and the* $\mathsf{Real}^{\uparrow}$ *seal row E before*

any limit-facing consumer can inspect it. The route introduces no countable choice, hidden filter selection, quotient filter equality, selected limit, host equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterChoiceFreeBasis`

Proof. Project the carrier $L = (F, B, R, S, Q, D, T, E, H, C, P, N)$ using Definition 1661.1. The displayed filter and basis rows are F and B, and the regular tail and located dyadic rows are R, D, T.

Apply the basis-refinement route of Definition 1661.4. A located-tail read first fixes the Cauchy filter row and then refines through the displayed basis row at the requested dyadic tolerance before the downstream sequence or real-facing rows are visible.

Now combine the Real seal boundary of Definition 1661.6 with the regular sequence handoff of Definition 1661.8. The route through S, Q, D, T reaches the seal E only as displayed finite readback data, so no countable-choice selector or hidden filter selection is introduced. □

Theorem 1661.6 (Located Cauchy filter Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *carrier factors through the complete finite route*

$$F, B, R, S, Q, D, T, E, H, C, P, N.$$

Consequently a downstream consumer cannot read the terminal real-name seal without also reading the Cauchy filter row, basis row, regular tail budget, finite stream windows, regular rational readback, dyadic tolerance ledger, located-tail witness, componentwise transport, displayed continuation, provenance, and local NameCert row.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterRealSealBoundary`

Proof. Begin with a downstream real-facing read. The carrier exposes the seal only as E, and E is listed after the filter, basis, regularity, window, readback, dyadic, and located-tail rows in Definition 1661.1.

Use the real-seal non-escape clause of Definition 1661.2. Any request that reaches E must first pass through F, B, R, S, Q, D, T, so the terminal seal is a routed finite read rather than a host-selected limit.

The structural rows then preserve the route. The rows H, C, P, N transport, replay, source, and name the same displayed packet, and the carrier has no coordinate for quotient filters, countable choice, hidden basis inspection, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1661.7 (Located Cauchy filter Bishop completion handoff)**.** *Every accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *packet consumed by* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *must expose the rows*

$$F, B, R, S, Q, D, T, E, H, C, P, N$$

before the completion seal is read. The route uses the existing $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CauchyFilterBasis}^{\uparrow}$*,* $\mathsf{RegularCauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, and it exports no quotient filter, selected limit, countable choice, host equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterBishopCompletionHandoff`

Proof. Project the accepted carrier using Definition 1661.1. The visible rows are exactly the filter row F, basis row B, regular filter row R, finite windows S, regular readback Q, dyadic ledger D, located-tail row T, Real seal row E, and structural rows H, C, P, N.

Apply the basis-refinement route of Definition 1661.4. A completion-facing request fixes the Cauchy filter row and basis row at the displayed dyadic tolerance before the regular filter row, finite windows, or readback rows are available.

Now use the Real seal boundary of Definition 1661.6 together with the regular sequence handoff of Definition 1661.8. The $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ consumer reads the completion seal only after $F, B, R, S, Q, D, T, E, H, C, P, N$ have been displayed, so the route has no coordinate for quotient filters, selected limits, countable choice, host equality, or ambient completeness. □

Theorem 1661.8 (Located Cauchy filter regular sequence handoff)**.** *Every sequence-facing consumer of an accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *packet*

$$L = (F, B, R, S, Q, D, T, E, H, C, P, N)$$

factors through the finite sequence route

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}$$

before the downstream $\mathsf{Real}^{\uparrow}$ *seal row E can be inspected. Equivalently, a consumer that asks for a sequence presentation of the located filter must read the window row S, the regular rational readback Q, the dyadic tolerance ledger D, and the located-tail witness T before it reaches E.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterRegularSequenceHandoff`

Proof. Project the accepted carrier. The sequence-facing rows are exactly S and Q, and the dyadic tolerance needed to interpret their located readback is the displayed ledger D.

Use the sequence handoff compatibility clause of Definition 1661.2. It forces S and Q to expose only finite windows and regular rational readbacks generated from the same located filter request.

Replay the continuation route C. The request passes through S, Q, D, T before it can reach the seal boundary E, so the real-facing read is downstream from the sequence route rather than a selected limit.

Finally, apply the Real seal boundary of Definition 1661.6. The rows H, C, P, N preserve and name the same finite handoff, and no quotient filter, countable-choice selector, hidden basis inspection, host equality, or ambient completeness row is exported. □

Theorem 1661.9 (Located Cauchy filter Real extraction)**.** *An accepted* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *packet whose located base, Cauchy-filter row, modulus row, and regular-readback row are displayed determines a* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible Real-readback route. The extraction preserves the locatedness and Cauchy rows through F, B, D, T and the regular sequence readback through S, Q, and it does not select a host limit, quotient stream equality, or completed-real object.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterRealExtraction`

Proof. Read the carrier of Definition 1661.1. The located base is displayed by B, D, T over the Cauchy filter row F, while the sequence-facing readback is displayed by the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows S, Q.

Apply the regular-tail location and basis-refinement route, Definition 1661.3 and Definition 1661.4. These force every located request through the same Cauchy filter, basis, dyadic tolerance, and located-tail rows before the regular filter and sequence readback are visible.

Now use the regular sequence handoff together with the Real seal boundary, Definition 1661.8 and Definition 1661.6. The route reaches E only after S, Q, D, T have been displayed, so the Real-facing extraction is a finite $\mathsf{RegSeqRat}^{\uparrow}$-compatible readback rather than a selected limit, quotient stream equality, or completed-real object. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilter}\uparrow}$.

Constructive story: A $\mathsf{LocatedCauchyFilter}^{\uparrow}$ packet is generated from a CauchyFilter row, a CauchyFilterBasis row, a RegularCauchyFilter tail budget, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a located-tail witness, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilter}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 1661.1, Definition 1661.2, Definition 1661.3, Definition 1661.4, and Definition 1661.6 seed the filter-level located Cauchy packet between CauchyFilter, CauchyFilterBasis, RegularCauchyFilter, LocatedCauchySequence, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilter}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedCauchyFilterUp.LocatedCauchyFilterNameCertObligations`

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected limits, countable choice, hidden infinite basis inspection, host real equality, ambient Real completeness, or a public standard bridge.

Upgrade path: Public export or bridge work requires a standard interface for the checked located-filter route.

Chapter 1662

A Concrete Naming Certificate for $\mathsf{RealUniformEmbedding}^{\uparrow}$

Following Bishop and Bridges' constructive Cauchy completion and the Bourbaki-Kelley uniform-space terminology, the $\mathsf{RealUniformEmbedding}^{\uparrow}$ packet records the finite uniform embedding by which a regular Cauchy name enters the $\mathsf{Real}^{\uparrow}$ seal without becoming a quotient stream or a host real. It binds the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal seal of chapter 38, the dense uniform route of chapter 1651, and the regular completion boundary of chapter 1639. The packet is the Real and Completion handoff for finite regular Cauchy data; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, select representatives, or compare host real values.

Definition 1662.1 (Real uniform embedding carrier)**.** *A* $\mathsf{RealUniformEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, W, D, Q, E, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy name row, read as a finite regular rational stream rather than as a quotient class.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family recording the inspected prefixes and common refinements available to the embedding route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that names dyadic radii, tail budgets, and comparison obligations for the displayed windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracting regular rational sequence data from S through W and D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the finite regular Cauchy data have passed through the dyadic and readback route.*
6. *U is the uniform-control compatibility row stating that the source row, finite windows, dyadic tolerances, readback row, and Real seal are read over one displayed uniform embedding path.*
7. *H records componentwise* hsame *transport for S, W, D, Q, E, U and for the structural rows.*
8. *C records displayed* Cont *replay from a Real or completion tolerance request through W, D, Q, U and into the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

9. *P is the* Pkg *provenance over* $\mathsf{RealUniformEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformEmbedding}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite regular Cauchy uniform embedding packet. It contains no coordinate for quotient stream equality, host real equality, selected representatives, countable choice, ambient completeness, or a detached Real object.

Lean4 ✓ `BEDC.Derived.RealUniformEmbeddingUp.TasteGate.RealUniformEmbeddingTasteGate single carrier alignment`

Lean4 (definition) `BEDC.Derived.RealUniformEmbeddingUp`

Lean4 ✓ `BEDC.Derived.RealUniformEmbeddingUp.RealUniformEmbeddingTasteGate single carrier alignment`

Theorem 1662.2 (Real uniform embedding NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformEmbedding}^{\uparrow}$ *carrier* $R = (S, W, D, Q, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUniformEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, StreamName window row, DyadicRatCore tolerance ledger, RegSeqRat readback row, Real seal row, uniform-control compatibility row, transport row, replay row, provenance row, and local naming row;*

2. *uniform-window compatibility, requiring every inspected finite window in W to be read over the same source row S and dyadic tolerance ledger D before it reaches the embedding route;*

3. *RegSeqRat readback stability, requiring Q to preserve the regular rational reading of S across the displayed windows and dyadic comparison budgets;*

4. *Real seal handoff, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through S, W, D, Q, U before the seal row E is exposed;*

5. *classifier transport, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the same ordered route;*

6. *ledger non-escape, requiring P and N to export no quotient stream equality, host real equality, selected representative, countable choice, ambient completeness, or detached Real carrier.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealUniformEmbeddingUp.RealUniformEmbeddingUp single carrier alignment`

Proof. Carrier admission is projection from Definition 1662.1. The tuple lists the source row, finite windows, dyadic tolerance ledger, regular rational readback, Real seal, compatibility row, transport, replay, provenance, and local naming.

Uniform-window compatibility is the row U read together with W and D. A finite window cannot enter the embedding route until it has been tied to the same source regular Cauchy row and the same dyadic comparison budget.

RegSeqRat readback stability is the row Q. It reads S only through the displayed windows and tolerance ledger, so refinements of W preserve the same regular rational data rather than replacing the source by a quotient representative.

The Real seal handoff is ordered by C. Replay sends a Real or completion request through W, D, Q, U before exposing E, and H transports only the displayed rows along the same route.

Finally, P and N package exactly the named dependencies and local certificate. Since the carrier contains no excluded quotient, host-equality, representative, choice, completeness, or detached-Real coordinate, the listed rows exhaust the NameCert surface. □

Theorem 1662.3 (Real uniform embedding seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUniformEmbedding}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto W \longmapsto D \longmapsto Q \longmapsto U \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a uniformly controlled regular Cauchy entry into the Real seal, and receives no quotient stream equality, host real equality, selected representative, countable-choice schedule, ambient completeness theorem, or detached Real object.

Proof. Let $R = (S, W, D, Q, E, U, H, C, P, N)$ be accepted. Carrier projection gives the consumer only the displayed rows of Definition 1662.1.

The route begins with the source regular Cauchy row S and the finite windows W. The dyadic tolerance ledger D fixes the comparison budget for each inspected window before any Real-facing read is available.

The readback row Q then exposes regular rational data through the same window and tolerance route. The compatibility row U binds that readback to the uniform-control surface used by dense uniform embedding and Bishop regular Cauchy completion consumers.

Only after those rows have been displayed does the replay row C expose the Real seal row E. The transport row H, provenance row P, and local naming row N preserve and name the same finite path.

There is no remaining carrier coordinate from which a consumer could project quotient stream equality, host real equality, a selected representative, countable choice, ambient completeness, or a detached Real object. Hence every Real consumer factors through the displayed route. □

Lean4 ✓ `BEDC.Derived.RealUniformEmbeddingUp.RealUniformEmbeddingSealHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformEmbedding}^{\uparrow}$ packet is generated from a source regular Cauchy row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a Real seal row, uniform-control compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealUniformEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 1662.1, Definition 1662.2, and Definition 1662.3 seed the finite uniform embedding surface from regular Cauchy names into the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealUniformEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host real equality, selected representatives, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, uniform-window compatibility, RegSeqRat readback stability, Real seal handoff, classifier transport, and ledger non-escape proofs for BEDC.Derived.RealUniformEmbeddingUp.

Chapter 1663

A Concrete Naming Certificate for $\mathsf{ApartnessTopology}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{ApartnessTopology}^{\uparrow}$ packet names the topology generated by finite apartness-neighbourhood evidence rather than by host subsets of a completed real space. It sits over the positive gap handoff of chapter 1380, the located neighbourhood windows of chapter 1498, the metric ball basis route of chapter 1655, the located metric comparison interface of chapter 1533, and the located set distance surface of chapter 1541. The chapter isolates the constructive-topology frontier where finite gap evidence is repacked as neighbourhood data without importing powersets, quotient opens, trichotomy, selected limits, or open-cover compactness.

Definition 1663.1 (ApartnessTopology carrier)**.** *An* $\mathsf{ApartnessTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, O, T, M, S, W, R, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{LocatedApartness}^{\uparrow}$ *positive-gap row, including the two located sources, the shared finite window, the dyadic endpoint ledgers, and the gap evidence.*
2. *O is the* $\mathsf{LocatedOpen}^{\uparrow}$ *neighbourhood-window row that records the finite interval-entry or positive-radius witnesses read by a local neighbourhood consumer.*
3. *T is the* $\mathsf{MetricTopology}^{\uparrow}$ *basis row through which the generated neighbourhood packet is admitted as an indexed topology row.*
4. *M is the* $\mathsf{LocatedMetric}^{\uparrow}$ *comparison row tying the same observations to finite metric-distance and dyadic locatedness reads.*
5. *S is the* $\mathsf{LocatedSet}^{\uparrow}$ *membership row exposing the displayed set admission and located-distance route used by neighbourhood membership checks.*
6. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row on which the apartness, located-open, located-metric, and located-set reads are aligned.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations inspected by G, O, M, S.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the positive gap, endpoint, radius, and tolerance rows used by the neighbourhood request.*
9. *D is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular-rational, and dyadic rows have been displayed.*

10. *E records the local neighbourhood packet generated from the transported apartness gap and the located-open window.*

11. *H records componentwise* hsame *transport for the apartness row, located-open row, metric-topology row, located-metric row, located-set row, readbacks, generated neighbourhood packet, continuation, provenance, and local naming row.*

12. *C records displayed* Cont *replay from apartness gap requests to located neighbourhood windows, then to metric-topology basis rows and located-set membership reads.*

13. *P is the* Pkg *provenance over* $\mathsf{ApartnessTopology}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{LocatedOpen}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, O, T, M, S, W, R, Q, D, E, H, C, P, N$. It has no coordinate for host powersets, quotient opens, classical trichotomy, selected limits, open-cover compactness, or an independent $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.ApartnessTopologyUp.ApartnessTopologyTasteGate single carrier alignment`

Theorem 1663.2 (ApartnessTopology NameCert obligation surface)**.** *For every accepted* $\mathsf{ApartnessTopology}^{\uparrow}$ *carrier $A = (G, O, T, M, S, W, R, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ApartnessTopology}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier visibility: the packet displays the located-apartness gap, located-open window, metric-topology basis route, located-metric comparison row, located-set membership row, readbacks, generated neighbourhood packet, transport, replay, provenance, and local naming row;*

2. *gap-to-neighbourhood generation: the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row of G is aligned over W, R, Q, D and repacked as the neighbourhood packet E through the located-open row O;*

3. *topology admission: the generated neighbourhood packet is admitted only through the indexed basis route T and the comparison and membership rows M, S;*

4. *transport stability: componentwise* hsame *transport preserves the apartness gap, located window, metric basis, located metric comparison, located set membership, readbacks, and generated neighbourhood packet together;*

5. *boundary non-escape: the packet exports no host powerset, quotient open, trichotomy row, selected limit, open-cover compactness datum, or standalone* $\mathsf{Real}^{\uparrow}$ *completeness field.*

No local NameCert obligation is stored outside these five finite rows.

Lean4 ✓ `BEDC.Derived.ApartnessTopologyUp.ApartnessTopologyCarrier namecert obligation surface`

Proof. Carrier visibility is projection from Definition 1663.1. The displayed tuple lists the apartness, neighbourhood, topology, located-metric, located-set, readback, transport, replay, provenance, and local naming rows.

The generation row uses the shared finite window. The positive gap of G is transported over W, R, Q, D and then read by the located-open row O, so the generated neighbourhood packet E is determined by finite apartness evidence and the displayed neighbourhood window.

Topology admission is forced through T, M, S. The metric-topology row supplies the indexed basis route, while the located-metric and located-set rows supply the comparison and membership reads required by the same neighbourhood packet.

Transport stability follows componentwise from H and replay from C. These rows move the gap, window, basis, comparison, membership, readback, and generated-neighbourhood data together, so no detached witness can enter the local certificate.

The negative boundary is immediate from the carrier coordinates. Since no host powerset, quotient-open representative, trichotomy row, selected limit, compactness witness, or independent completeness field appears in A, the five listed rows exhaust the NameCert surface. □

Theorem 1663.3 (Apartness neighbourhood transport)**.** *Let $A = (G, O, T, M, S, W, R, Q, D, E, H, C, P, N)$ be an accepted* $\mathsf{ApartnessTopology}^{\uparrow}$ *packet. If the positive apartness gap row of G is transported componentwise along H over the shared finite window and readbacks W, R, Q, D, then the transported gap generates the same located neighbourhood packet E through O, T, M, S and the replay row C. This generation uses only the finite* BHist *route named by the carrier; it does not invoke host powersets, trichotomy, quotient opens, selected limits, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ApartnessTopologyUp.ApartnessTopologyCarrier neighborhood apartness transport`

Proof. Begin with the positive gap row inside G. The row is already attached to the shared finite window, regular-rational readbacks, dyadic endpoint and radius ledgers, and the real seal listed as W, R, Q, D.

Apply the transport row H componentwise. The transported packet still carries the same finite window indices, readback rows, dyadic gap evidence, and real-facing seal, so the transported apartness evidence remains aligned with the located-open window O.

Replay the neighbourhood request through C. The replay first reads O to form the located neighbourhood packet, then uses T for indexed topology admission, M for located metric comparison, and S for located set membership.

The resulting neighbourhood packet is therefore the same E named in the carrier, transported along the same finite rows. The construction never leaves $G, O, T, M, S, W, R, Q, D, E, H, C, P, N$, and hence cannot use any excluded powerset, quotient, trichotomy, selection, compactness, or completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessTopology}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ApartnessTopology}^{\uparrow}$ packet is generated from located-apartness positive gap evidence, located-open neighbourhood windows, metric-topology basis rows, located-metric comparison rows, located-set membership rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore ledgers, RealUp seals, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1663.1, Definition 1663.2, and Definition 1663.3 seed the finite apartness-generated neighbourhood carrier, obligation surface, and transport route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient topologies, host powersets, classical trichotomy, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, gap-to-neighbourhood generation, topology admission, transport stability, boundary non-escape, and neighbourhood transport for BEDC.Derived.ApartnessTopologyUp.

Chapter 1664

A Concrete Naming Certificate for $\mathsf{RealOne}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealOne}^{\uparrow}$ packet records the unit real as the visible rational-one route into the real-name surface, not as a host primitive. It sits beside the zero-real packet of chapter 1619 and reads its source through the rational embedding route of chapter 1491, the rational carrier of chapter 37, the constant window of chapter 126, the stationary regular-rational readback of chapter 301, the dyadic unit ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is the finite unit-real route consumed by later real algebra, order, metric, and completion-facing chapters.

Definition 1664.1 (RealOne carrier)**.** *A* $\mathsf{RealOne}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (q, S, R, D, E, H, C, P, N)$$

with the following displayed coordinates:

1. *q is the* $\mathsf{Rat}^{\uparrow}$ *unit seed row, read as the rational numerator-denominator packet for* 1*.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *window whose every admitted finite request reads the same seed q.*
3. *R is the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the constant unit window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *unit ledger recording the dyadic endpoint, tolerance, and radius rows used by the unit source.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the rational seed, constant window, stationary readback, and dyadic unit ledger have been displayed.*
6. *H stores componentwise* hsame *transport for the rational seed, constant window, stationary readback, dyadic ledger, terminal seal, provenance, and local name.*
7. *C stores displayed* Cont *replay routes from q through S, from S through R and D, and from those rows into the terminal seal E.*
8. *P records* Pkg *provenance over* $\mathsf{RealOne}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{RationalEmbedding}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, trichotomy, multiplication laws, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealOneUp.RealOneTasteGate single carrier alignment`

Theorem 1664.2 (RealOne NameCert obligations)**.** *For every accepted* $\mathsf{RealOne}^{\uparrow}$ *carrier* $O = (q, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the rational unit seed, constant StreamName window, stationary RegSeqRat readback, DyadicRatCore unit ledger, terminal RealUp seal, transport, replay, provenance, and local naming rows;*
2. *rational embedding route, requiring the unit seed to be read through the* $\mathsf{Rat}^{\uparrow}$ *row and the* $\mathsf{RationalEmbedding}^{\uparrow}$ *handoff before any real-facing consumer may inspect it;*
3. *stationary unit-source route, requiring the seed to be re-read only through the constant window, stationary regular-rational readback, and dyadic unit ledger;*
4. *terminal RealUp seal handoff, requiring the real-facing row* E *to be exported only after* q, S, R, D *and the structural rows have been displayed;*
5. *ledger nonescape from quotient streams, host real equality, selected limits, trichotomy, multiplication laws, and ambient completeness.*

No NameCert obligation is stored outside these five rows.

Lean4 ✓ `BEDC.Derived.RealOneUp.RealOneNameCertObligations`

Proof. Carrier admission is projection from Definition 1664.1. The visible data are the rational unit seed q, the constant window S, the stationary readback R, the dyadic unit ledger D, the terminal seal E, and the structural rows H, C, P, N.

The rational embedding row is forced before the real seal can be read. The packet exposes q as a $\mathsf{Rat}^{\uparrow}$ unit source and sends that source through the $\mathsf{RationalEmbedding}^{\uparrow}$ handoff rather than through a host real constant.

The stationary source row is then determined by S, R, D. Every admitted finite request rereads the same rational seed, the regular-rational readback is stationary, and the dyadic ledger records the corresponding unit endpoint and tolerance rows.

The terminal seal is reached only through E, with transport and replay supplied componentwise by H and C. Since the carrier contains no coordinate for quotient stream equality, host equality, selected limits, trichotomy, multiplication laws, or ambient completeness, none of those principles can be exported through the local $\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$ row. □

Theorem 1664.3 (RealOne rational embedding route)**.** *Every accepted* $\mathsf{RealOne}^{\uparrow}$ *packet exposes the rational-one source through the finite route*

$$q \longrightarrow S \longrightarrow R \longrightarrow D \longrightarrow E$$

together with H, C, P, N*. A real algebra, order, metric, or completion-facing consumer therefore receives the unit real only after the* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{RationalEmbedding}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed.*

Proof. Start with an accepted carrier $O = (q, S, R, D, E, H, C, P, N)$. The seed q is the rational unit row, so the first public source is the $\mathsf{Rat}^{\uparrow}$ carrier read through the rational embedding handoff.

The constant window S makes the route finite at every requested observation. The readback R records the same rational value as a stationary regular-rational row, and D records the dyadic unit ledger attached to that row.

Only after these rows are visible does the terminal seal E become available. Thus any downstream consumer receives the unit real through the displayed rational-to-real route and not through a host-level real constant. □

Lean4 ✓ `BEDC.Derived.RealOneUp.RealOneRationalEmbeddingRoute`

Theorem 1664.4 (RealOne terminal seal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RealOne}^{\uparrow}$ *packet exports exactly the displayed unit route* q, S, R, D, H, C, P, N*. It exports no quotient stream equality, host real equality, selected limit, trichotomy datum, multiplication unit law, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RealOneUp.RealOneTerminalSealNonescape`

Proof. Project the accepted packet. The only rows available before the terminal seal are the rational unit seed, the constant window, the stationary readback, the dyadic unit ledger, and the transport, replay, provenance, and local naming rows.

The seal E is downstream of those rows and has no independent coordinate carrying a quotient, a host equality proof, a selected limit, an order trichotomy decision, a multiplication law, or a completeness principle. Therefore the real-facing boundary preserves the visible unit route and cannot export any omitted principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealOne}^{\uparrow}$ packet is generated from the rational unit seed, a constant StreamName window, stationary RegSeqRat readback, a DyadicRatCore unit ledger, a terminal RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$, seed)

Scope closed: Definition 1664.1, Definition 1664.2, Definition 1664.3, and Definition 1664.4 seed the finite rational-one-to-RealUp unit route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealOne}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim real multiplication laws, quotient real equality, host equality, trichotomy, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational embedding routing, stationary unit-source routing, terminal RealUp seal handoff, componentwise transport and replay stability, and ledger nonescape for the same finite packet.

Chapter 1665

A Concrete Naming Certificate for $\mathsf{IshiharaTrick}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{IshiharaTrick}^{\uparrow}$ packet records the finite inspection boundary used when a real-name argument refines a disjunctive apartness situation without importing a global semi-classical principle. It is read against the constructive boundary example of chapter 1406, the regular-rational readback of chapter 301, the stream schedule of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a Real/Completion boundary object: it exhibits the rows a consumer may inspect and records that no LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness travels across the boundary.

Definition 1665.1 (Ishihara trick carrier)**.** *An* $\mathsf{IshiharaTrick}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, R, T, W, D, E, A, H, C, P, N)$$

with the following displayed coordinates:

1. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite observation requests admitted by the inspection.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row returning the rational approximants actually inspected from that schedule.*
3. *T is the displayed real-name test sequence, presented only through the finite schedule and regular-rational readback rows already named by S and R.*
4. *W is the finite inspection window naming the tested indices of T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, endpoint comparisons, and finite margin rows used by the inspection.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, T, W, D have been displayed.*
7. *A is the binary test outcome boundary: it records which finite tolerance row witnesses the local apartness alternative or, when no such row is displayed, which refinement request remains inside the schedule.*
8. *H stores componentwise* hsame *transport for the schedule, readback, test sequence, inspection window, dyadic ledger, real seal, boundary row, provenance, and local name.*

9. *C stores* Cont *replay from finite schedule requests through R, T, W, D, and from those rows into the sealed real-facing boundary A.*
10. *P records* Pkg *provenance over* $\mathsf{IshiharaTrick}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$ *row.*

The carrier has no coordinate for LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, countable choice, a selected limit, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickTasteGate single carrier alignment`

Theorem 1665.2 (Ishihara trick NameCert obligations)**.** *For every accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *carrier $O = (S, R, T, W, D, E, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the StreamName schedule, RegSeqRat readback, displayed real-name test sequence, finite inspection window, DyadicRatCore tolerance ledger, Real seal, inspected binary boundary, transport, replay, provenance, and local naming rows;*
2. *schedule-readback coherence, requiring every inspected rational value to be read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
3. *dyadic tolerance discipline, requiring every apartness or refinement branch to cite a displayed dyadic radius, endpoint comparison, and finite margin row from D within the window W;*
4. *real-boundary handoff, requiring a Real/Completion consumer to receive the branch only through E together with the displayed rows $S, R, T, W, D, A, H, C, P, N$;*
5. *ledger nonescape from LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, and ambient real completeness.*

No NameCert obligation is stored outside these five rows.

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1665.1. The visible data are the finite schedule S, readback R, displayed test sequence T, finite window W, tolerance ledger D, terminal real seal E, inspected boundary A, and structural rows H, C, P, N.

Schedule-readback coherence follows because the packet contains no readback coordinate independent of S and R. Every rational approximant inspected by the branch is therefore routed through the displayed stream and regular-rational rows.

The dyadic tolerance row is forced by D and the inspection window W. A branch may expose a local apartness witness only by naming a displayed dyadic radius, endpoint comparison, and finite margin inside the tested window; otherwise it records the finite refinement request still admitted by the schedule.

The real-boundary handoff is downstream of E, with transport and replay supplied componentwise by H and C. Since the carrier contains no coordinate for LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, or ambient completeness, none of those principles can be exported through the local $\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$ row. □

Theorem 1665.3 (Ishihara trick finite test boundary)**.** *Every accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *packet exposes its binary branch only through the displayed real-name test sequence T, finite inspection window W, and dyadic ledger D. Hence the classifier can distinguish the displayed local branch from continued refinement only by finite inspection of the named rows.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickCarrier finite test boundary`

Proof. The carrier presents the test sequence T through the already displayed stream and regular-rational rows. The window W names the inspected indices, so no classifier step can read outside the finite schedule.

The branch row A is binary at the packet boundary. If a dyadic separation is displayed, it cites a radius, endpoint comparison, and finite margin from D inside W; if no such row is displayed, A records only that the schedule continues by refinement.

Componentwise hsame and Cont replay preserve this finite route. Therefore the classifier cannot obtain a third outcome, a global ordering of reals, or a selected limit from the packet. □

Theorem 1665.4 (Ishihara trick branch stability)**.** *Let*

$$O = (S, R, T, W, D, E, A, H, C, P, N)$$

and

$$O' = (S', R', T', W', D', E', A', H', C', P', N')$$

be accepted $\mathsf{IshiharaTrick}^{\uparrow}$ *packets. Suppose the displayed schedule, regular-rational readback, test sequence, inspection window, dyadic ledger, and real seal rows of O and O' are related componentwise by the transport rows, and suppose the branch inspection is replayed through the same* Cont *route. Then the branch row is stable under that transport: a displayed apartness branch in A remains the corresponding displayed apartness branch in A' with transported dyadic radius, endpoint comparison, and margin, while a refinement branch in A remains a refinement request inside the transported schedule S'.*

Proof. Project the carrier rows of O and O'. The only rows available to a branch inspection are the schedules, readbacks, displayed test sequences, finite windows, dyadic ledgers, real seals, branch rows, componentwise transports, and continuation replays named in the two packets.

Apply Definition 1665.3 to the source packet. Every branch read is restricted to the displayed T, W, D rows, so an apartness branch can cite only a dyadic radius, endpoint comparison, and finite margin from the ledger inside the window, and a non-apartness branch can record only continued refinement through the schedule.

The row H transports the displayed schedule, readback, test sequence, window, dyadic ledger, real seal, and boundary row componentwise to the corresponding rows of O'. The row C replays the same continuation route through those transported rows, so the inspected branch cannot change shape during transport.

Thus a displayed apartness branch remains an apartness branch with transported dyadic data, and a refinement branch remains a refinement request inside the transported schedule. Since neither packet contains a coordinate for LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, a selected limit, countable choice, or ambient completeness, no outside principle can supply an additional branch case. □

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickCarrier branch stability`

Theorem 1665.5 (Ishihara trick binary branch boundary)**.** *Every accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *packet exposes exactly two branch readings at the finite Real/Completion boundary: either a displayed dyadic apartness branch whose radius, endpoint, and margin rows come from D inside W, or a refinement request inside the transported schedule. No LLPO, WLPO, Markov principle, trichotomy, quotient equality, selected limit, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness supplies a third case.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickCarrier binary branch boundary`

Proof. Project the carrier described in Definition 1665.1. The branch coordinate A is read only after the schedule S, readback R, test sequence T, finite window W, dyadic ledger D, and real seal E have been displayed.

Apply the finite-test boundary of Definition 1665.3. A positive branch can cite only a dyadic radius, endpoint comparison, and finite margin from D inside W; if no such row is displayed, the packet records only a refinement request still inside the schedule.

Apply branch stability from Definition 1665.4. Componentwise transport and continuation replay preserve the displayed apartness branch and the refinement-request branch, and the carrier has no coordinate for a third semi-classical or completion-supplied alternative. □

Theorem 1665.6 (Ishihara trick real seal nonescape)**.** *Every Real/Completion consumer of an accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *packet factors through the displayed finite rows*

$$S, R, T, W, D, E, A, H, C, P, N.$$

Consequently the consumer receives only a finite schedule, rational readback, displayed test sequence, finite inspection window, dyadic tolerance ledger, terminal real seal, inspected binary boundary, componentwise transport, continuation replay, provenance, and local NameCert row. It receives no LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickCarrier real seal nonescape`

Proof. Let a Real/Completion consumer accept the packet $O = (S, R, T, W, D, E, A, H, C, P, N)$. By the carrier definition, the only real-facing coordinate is the seal E, and that seal is reached after the schedule, readback, test sequence, finite window, tolerance ledger, inspected boundary, and structural rows have been displayed.

The consumer's branch information is therefore finite. It can inspect the schedule row, replay the regular-rational readback, cite the dyadic tolerance ledger inside the finite window, and use the componentwise hsame and Cont rows to transport those observations through the seal.

No omitted semi-classical or completion principle is a coordinate of the packet. Thus factorization through the displayed rows cannot export LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, a selected limit, countable choice, or ambient real completeness. □

Theorem 1665.7 (Ishihara trick bounded-sequence route)**.** *Any bounded real-sequence or Specker-boundary consumer using an accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *packet must pass through the displayed rows*

$$S, R, T, W, D, E, A, H, C, P, N$$

before interpreting a branch as bounded-sequence evidence. The route presents a finite schedule, regular-rational readback, displayed real-name test sequence, finite inspection window, dyadic tolerance ledger, terminal real seal, binary boundary row, componentwise transport, continuation replay, provenance, and local NameCert row. It does not expose LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickBoundedSequenceRoute`

Proof. Project the carrier of Definition 1665.1. Its coordinates are exactly the displayed rows $S, R, T, W, D, E, A, H, C, P, N$, so a bounded-sequence consumer has no independent sequence, bound, or completion coordinate to inspect.

Apply the finite-test boundary of Definition 1665.3. The branch can be read only after the test sequence T, window W, and dyadic ledger D are exposed through the schedule and regular-rational readback rows.

Apply the binary branch boundary of Definition 1665.5. A consumer may receive either the displayed dyadic apartness branch or the refinement request still inside the schedule; no third case is available for a bounded-sequence or Specker-boundary reading.

Finally use the real-seal nonescape theorem Definition 1665.6. The terminal $\mathsf{Real}^{\uparrow}$ handoff carries the same finite rows and no LLPO/WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limit, countable choice, or ambient completeness principle, so bounded-sequence evidence can be interpreted only along the displayed route. □

Theorem 1665.8 (Ishihara trick Specker-boundary handoff)**.** *Every Bishop–Specker or Markov-boundary consumer of an accepted* $\mathsf{IshiharaTrick}^{\uparrow}$ *packet must read the displayed rows*

$$S, R, T, W, D, E, A, H, C, P, N$$

before interpreting the branch as obstruction evidence for chapter 1406, chapter 6056, or chapter 1666. The handoff presents only the finite schedule, regular-rational readback, displayed real-name test sequence, finite inspection window, dyadic tolerance ledger, terminal real seal, binary boundary row, componentwise transport, continuation replay, provenance, and local NameCert row. It exports no LLPO, WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.IshiharaTrickUp.IshiharaTrickSpeckerBoundaryHandoff`

Proof. Project the carrier of Definition 1665.1. A Bishop–Specker or Markov-boundary reader receives exactly the tuple $S, R, T, W, D, E, A, H, C, P, N$, so the downstream obstruction interpretation has no independent supremum, negative-evidence rule, or completion row to inspect.

Apply the finite-test boundary of Definition 1665.3. The possible obstruction evidence is visible only after the displayed test sequence T, finite window W, and dyadic ledger D have been read through the schedule S and regular-rational row R.

Apply the binary branch boundary of Definition 1665.5. The consumer may interpret a displayed dyadic apartness branch, or it may continue with the refinement request still inside the schedule; no third branch is supplied by a Specker, Bishop–Specker, or Markov-sensitive boundary read.

Finally use the real-seal nonescape theorem Definition 1665.6. The terminal $\mathsf{Real}^{\uparrow}$ seal and the structural rows H, C, P, N preserve the same finite route, excluding LLPO/WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, and ambient completeness from the obstruction handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$.

Constructive story: An $\mathsf{IshiharaTrick}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat readback, displayed real-name test sequence, finite inspection window, DyadicRatCore tolerance ledger, Real seal, inspected binary boundary, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$, seed)

Scope closed: Definition 1665.1, Definition 1665.2, Definition 1665.3, Definition 1665.4, Definition 1665.5, Definition 1665.6, Definition 1665.7, and Definition 1665.8 seed the finite Real/Completion boundary for Ishihara-style inspection.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IshiharaTrick}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.IshiharaTrickUp`

Bridge status: none

Not claimed: This chapter does not claim LLPO/WLPO, Markov-principle leakage, trichotomy, quotient real equality, selected limits, countable choice, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-readback coherence, dyadic tolerance discipline, real-boundary handoff, componentwise transport and replay stability, and ledger nonescape for the same finite packet.

Chapter 1666

A Concrete Naming Certificate for $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$

Following Markov, Ishihara, and Bishop–Bridges constructive analysis, the $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ packet records the finite boundary at which an inspection argument may use Markov-style negative evidence as named data without accepting Markov's principle as a global rule. It is read beside the Ishihara inspection packet of chapter 1665, the Specker control example of chapter 1406, the stream schedule of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a Real/Completion boundary object: it exposes only the finite rows a consumer may inspect and refuses any route that would turn those rows into LPO, LLPO, trichotomy, countable choice, quotient stream equality, selected limits, or ambient real completeness.

The $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ dependency route consumes the finite branch-inspection row of chapter 1665 and the obstruction row of chapter 1406 before exporting any Markov-sensitive boundary read. Thus its horizontal lattice records Ishihara inspection and Specker obstruction as displayed sibling data, not as imported principles.

Definition 1666.1 (Markov principle boundary carrier)**.** *A* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, S, R, D, E, W, H, C, P, N)$$

with the following displayed coordinates:

1. *I is an* $\mathsf{IshiharaTrick}^{\uparrow}$ *inspection row naming the finite apartness/refinement test whose branch is being bounded.*
2. *K is a* $\mathsf{SpeckerSequence}^{\uparrow}$ *control row recording the nonconvergent obstruction that prevents the inspection from becoming a selected real limit.*
3. *S is a* $\mathsf{StreamName}^{\uparrow}$ *schedule row listing the finite stream requests admitted by the boundary.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational values actually inspected from S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the radii, endpoint comparisons, and finite margin rows used by the boundary.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the rows I, K, S, R, D have been displayed.*
7. *W is the Markov-boundary witness row: it records either a displayed finite failure of zerohood in the inspected schedule or the finite refusal row explaining why no global Markov rule is exported.*

8. *H stores componentwise* hsame *transport for the inspection, Specker control, schedule, readback, dyadic ledger, real seal, witness row, provenance, and local name.*
9. *C stores* Cont *replay from finite schedule requests through R and D, through the Ishihara branch I, and finally into the sealed real-facing witness W.*
10. *P records* Pkg *provenance over* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{IshiharaTrick}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovPrincipleBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for Markov's principle as an imported axiom, LPO, LLPO, trichotomy, countable choice, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MarkovPrincipleBoundaryUp.MarkovPrincipleBoundaryUpTasteGate single carrier alignment`

Theorem 1666.2 (Ishihara–Specker boundary route)**.** *For every accepted* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *carrier $M = (I, K, S, R, D, E, W, H, C, P, N)$, every real-facing inspection branch factors through the displayed route*

$$I, K, S, R, D, E, W, H, C, P, N.$$

Consequently an $\mathsf{IshiharaTrick}^{\uparrow}$ *refinement may be compared with a* $\mathsf{SpeckerSequence}^{\uparrow}$ *obstruction only by citing the finite stream schedule, regular-rational readback, dyadic tolerance ledger, real seal, witness row, componentwise transport, replay, provenance, and local NameCert row displayed by M.*

Lean4 ✓ `BEDC.Derived.MarkovPrincipleBoundaryUp.MarkovPrincipleBoundaryIshiharaSpeckerRoute`

Proof. The carrier definition lists the only coordinates admitted by the boundary. The Ishihara inspection row I supplies the finite branch being tested, while the Specker row K supplies the obstruction preventing that branch from being promoted to a selected limit.

The schedule-readback-tolerance part is carried by S, R, D. Thus every inspected rational value and every finite margin used by the branch is named before the real seal E is reached.

The witness row W is downstream of those finite rows. Componentwise hsame transport in H and Cont replay in C allow a consumer to replay the route without adding any unlisted classifier. The provenance and local NameCert rows P, N then repack the same displayed data as the public boundary certificate. □

Theorem 1666.3 (Markov boundary nonescape)**.** *Every Real/Completion consumer of an accepted* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *packet receives only finite inspection data and the sealed witness row W. It receives no Markov's principle as an imported axiom, no LPO, no LLPO, no trichotomy, no countable choice, no quotient stream equality, no selected limit, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Let a Real/Completion consumer accept $M = (I, K, S, R, D, E, W, H, C, P, N)$. By Definition 1666.1, the only real-facing coordinate is the seal E, and the only Markov-sensitive coordinate is the displayed witness row W.

The route theorem Definition 1666.2 forces the consumer to reach W through the finite Ishihara inspection, Specker control, stream schedule, regular-rational readback, and dyadic tolerance ledger. None of these coordinates contains a global principle or a selected completion operator.

Since the carrier has no coordinate for Markov's principle as an imported axiom, LPO, LLPO, trichotomy, countable choice, quotient stream equality, selected limits, or ambient real completeness, transport and replay cannot export any such principle through the local $\mathsf{NameCert}_{\mathsf{MarkovPrincipleBoundary}^{\uparrow}}$ row. □

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovPrincipleBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ packet is generated from an Ishihara inspection row, Specker control row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, Markov-boundary witness row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovPrincipleBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1666.1, Definition 1666.2, and Definition 1666.3 seed the finite Real/Completion boundary for Markov-sensitive inspection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovPrincipleBoundary}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MarkovPrincipleBoundaryUp`

Bridge status: none

Not claimed: This chapter does not claim Markov's principle as an imported axiom, LPO, LLPO, trichotomy, countable choice, quotient stream equality, selected limits, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Ishihara–Specker route factorization, stream-readback tolerance discipline, real-boundary handoff, componentwise transport and replay stability, and ledger nonescape for the same finite packet.

Chapter 1667

A Concrete Naming Certificate for $\mathsf{NetConvergence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style convergence of nets, the $\mathsf{NetConvergence}^{\uparrow}$ packet names only the finite directed-tail evidence a completion-facing consumer can inspect. It sits between the Moore–Smith net carrier of chapter 3403, the directed-window surface of chapter 589, the finite filterbase handoff of chapter 333, the entourage discipline of chapter 339, and the L10 real-and-completion route of section 9467.12. The Moore–Smith net row supplies the directed-tail and value-window route that this convergence interface consumes before filter and Real handoff rows are read. The packet is a convergence interface for Real and completion arguments; it is not a selected limit, a quotient of nets, or an ambient completeness theorem.

Definition 1667.1 (Net convergence carrier)**.** *A* $\mathsf{NetConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (D, T, E, A, F, S, R, L, H, C, P, M)$$

with the following displayed rows:

1. *D is the directed-index window, naming the finite index vertices, the displayed upper-window witnesses, and the finite tail cut used by every convergence read.*
2. *T is the tail-neighbourhood ledger, recording the neighbourhood requests whose tails are inspected inside D.*
3. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, giving the finite entourage request through which the tail-neighbourhood read is interpreted.*
4. *A is an optional* $\mathsf{CauchyNet}^{\uparrow}$ *source row; when present, its common-window classifier is read only through the same D, T, E rows, and when absent the packet records no Cauchy-net source.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row obtained from the displayed tail-neighbourhood family.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite observation windows used by the directed tail.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those scheduled observations and dyadic comparisons.*
8. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, T, E, F, S, R have been displayed.*
9. *H records componentwise* hsame *transport for the directed window, tail-neighbourhood ledger, entourage row, optional source row, filter handoff, stream schedule, regular-rational readback, Real seal, provenance, and local naming rows.*

10. *C records the displayed* Cont *replay for every consumer read through the same coordinates.*
11. *P is the* Pkg *provenance over* $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *M is the local* $\mathsf{NameCert}_{\mathsf{NetConvergence}^{\uparrow}}$ *row.*

The carrier exports only these finite rows. It contains no selected limit point, quotient net, countable-choice selector, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceTasteGate single carrier alignment`

Theorem 1667.2 (Net convergence NameCert obligations)**.** *For an accepted carrier $N = (D, T, E, A, F, S, R, L, H,$ of Definition 1667.1, the local* $\mathsf{NameCert}_{\mathsf{NetConvergence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the directed-index window, tail-neighbourhood ledger, uniform-entourage row, optional CauchyNet source row, CauchyFilter handoff row, StreamName schedule, RegSeqRat readback, Real seal boundary, and structural rows;*
2. *tail classification, stating that every convergence read names a displayed tail in D before it can consume a neighbourhood row in T;*
3. *entourage interpretation, stating that the neighbourhood read is routed through the finite* $\mathsf{UniformSpace}^{\uparrow}$ *row E rather than a host topology or ambient uniformity;*
4. *optional-source discipline, stating that a present* $\mathsf{CauchyNet}^{\uparrow}$ *row is read only through the same directed window and that an absent source row exports no hidden Cauchy-net data;*
5. *CauchyFilter handoff, stating that the filter-facing consumer sees only the tail-neighbourhood family and refinement rows displayed by F;*
6. *Real seal non-escape, stating that L is consumed only after the stream schedule S and regular-rational readback R have been displayed;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, M.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceNameCert obligations`

Proof. Project the accepted packet of Definition 1667.1. The analytic coordinates are D, T, E, A, F, S, R, L, and the structural coordinates are H, C, P, M.

A convergence read first names a directed tail in D. The tail-neighbourhood ledger T can then be inspected only at that displayed tail, and the neighbourhood is interpreted by the finite entourage row E.

The optional source row does not add a second convergence surface. If A is present, its common-window classifier is read through the same directed window and entourage data; if A is absent, there is no coordinate from which source-net data can be exported.

The downstream routes are likewise fixed by visible rows. The Cauchy-filter consumer reads F, while Real-facing consumers read S and R before reaching the seal L. Transport, replay, provenance, and naming are componentwise through H, C, P, M, so the listed rows exhaust the local naming surface. □

Theorem 1667.3 (Moore–Smith dependency for net convergence)**.** *Every* $\mathsf{NetConvergence}^{\uparrow}$ *consumer that is sourced from a* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet reads the Moore–Smith directed-index row,*

value-window row, and finite tail ledger before it reads the $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row or the* $\mathsf{Real}^{\uparrow}$ *seal row. The dependency is confined to displayed finite directed-tail evidence and does not introduce selected subnets, cofinal choice, quotient nets, or ambient completeness.*

Proof. Use the Moore–Smith net obligations of Definition 3403.2. They expose directed indices, value windows, and tail requests as the finite packet consumed by convergence-side readers.

Now apply the NetConvergence carrier discipline of Definition 1667.1. The optional source row A can contribute source-net material only through the same directed window, tail ledger, and entourage rows D, T, E.

The downstream rows are read after that directed-tail route. The $\mathsf{CauchyFilter}^{\uparrow}$ handoff row F and the $\mathsf{Real}^{\uparrow}$ seal row L therefore consume displayed Moore–Smith evidence only through the accepted $\mathsf{NetConvergence}^{\uparrow}$ packet, leaving no coordinate for a selected subnet, cofinal-choice operator, quotient net, or ambient completeness theorem. □

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier moore smith dependency`

Theorem 1667.4 (Net convergence CauchyFilter handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier factors through the displayed route*

$$D \longmapsto T \longmapsto E \longmapsto F$$

and every $\mathsf{Real}^{\uparrow}$ *read after that handoff factors through the same* F *row together with the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule* S *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*. No selected limit, quotient net, countable-choice selector, or ambient completeness row can alter the handoff.*

Proof. Start from Definition 1667.2. The tail classification obligation places every convergence read on a displayed directed tail in D, and the tail-neighbourhood ledger T supplies the finite neighbourhood family inspected at that tail.

The entourage interpretation obligation routes that neighbourhood family through the uniform row E. Thus the filter-facing datum is not a host filter; it is exactly the finite family that the handoff row F exposes to $\mathsf{CauchyFilter}^{\uparrow}$ consumers.

For a Real-facing read, the packet must continue through S and R before reaching L. These rows are part of the same accepted carrier and are transported only by the displayed hsame and Cont data in H and C.

There is no remaining coordinate in N for a selected limit point, a quotient of nets, a countable-choice selector, or an ambient completeness theorem. Consequently the handoff is determined by the displayed route through D, T, E, F, S, R, L. □

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier cauchy filter handoff`

Theorem 1667.5 (Net convergence directed-window admission)**.** *Every accepted* $\mathsf{NetConvergence}^{\uparrow}$ *convergence read first exposes the directed-index window* D*, the tail-neighbourhood ledger* T*, and the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row* E *of Definition 1667.1. No CauchyNet source row, CauchyFilter handoff row, StreamName schedule, RegSeqRat readback, or Real seal can be consumed before this directed window has been displayed.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier directed window admission`

Proof. Use the carrier projection of Definition 1667.1. The rows D, T, E are the only coordinates that name a directed tail, the neighbourhood requests inspected at that tail, and the finite entourage row interpreting those requests.

Apply the tail classification and entourage interpretation clauses of Definition 1667.2. They force every convergence read to pass through D and T, then through E, before any downstream row can be inspected.

The optional source row A, the filter handoff F, the stream and rational readback rows S, R, and the Real seal L are downstream coordinates. Since the carrier has no alternative directed-tail coordinate, they cannot be consumed before the displayed directed window is exposed. □

Theorem 1667.6 (Net convergence CauchyNetLimit handoff)**.** *Every CauchyNetLimit-facing consumer of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier receives the same directed tail* D, T, E *before it can inspect the optional* $\mathsf{CauchyNet}^{\uparrow}$ *row* A *or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* L*. The handoff is therefore a finite directed-tail route rather than a selected limit, host convergence predicate, quotient net, or ambient completeness witness.*

Proof. Start from Definition 1667.5. It places D, T, E before every convergence read, independently of which downstream consumer is asking for the data.

A CauchyNetLimit-facing consumer can use the optional CauchyNet source only through A. By the optional-source discipline of Definition 1667.2, a present A is read through the same directed window, and an absent A exports no source-net data.

The Real-facing part of the handoff reaches L only after the stream schedule S and the regular-rational readback R have been displayed. Thus the consumer receives one finite directed-tail route before either the CauchyNet source or the Real seal is consumed. □

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier cauchy net limit handoff`

Theorem 1667.7 (Net convergence Moore–Smith filter boundary)**.** *Moore–Smith consumers and* $\mathsf{CauchyFilter}^{\uparrow}$ *consumers of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier meet only through the displayed finite directed-tail evidence* D, T, E, F *and the structural rows* H, C, P, M*. The boundary exports no selected subnet, cofinal-choice operator, quotient net, host filter equality, or ambient completeness row.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier moore smith filter boundary`

Proof. Read the Moore–Smith dependency theorem Definition 1667.3. It says that source-net consumers contribute directed-index, value-window, and tail-ledger evidence through the finite directed-tail route.

Read the CauchyFilter handoff theorem Definition 1667.4. It routes every filter-facing datum through D, T, E, F before any Real-facing rows are inspected.

The structural rows H, C, P, M transport, replay, package, and locally name that same route. Since neither theorem supplies a coordinate for selected subnets, cofinal choice, quotient nets, host filter equality, or ambient completeness, the Moore–Smith and filter consumers meet only at the displayed finite boundary. □

Theorem 1667.8 (NetConvergence Moore–Smith filter handoff)**.** *Every completion-facing read of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier* $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ *that compares Moore–Smith and filter-base evidence first factors through the finite directed-set windows* D*, the Moore–Smith convergence boundary rows* T, E, A*, the* $\mathsf{FilterBase}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *tail rows carried by* F*, the* $\mathsf{StreamName}^{\uparrow}$ *observation windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal* L*, and then the transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows* H, C, P, M*. The handoff is a finite route inspection; it does not select a subnet, replace the filter by a host equality, or appeal to an ambient limit theorem.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier moore smith real seal handoff`

Proof. Expose the Moore–Smith side first. By Definition 1667.3, a source-net consumer can contribute only directed-index, value-window, and tail-ledger evidence, and by Definition 1667.5 those rows enter the carrier through D, T, E before any downstream row is read.

Next expose the filter side. The CauchyFilter handoff theorem Definition 1667.4 routes every filter-facing datum through D, T, E, F, while Definition 1667.9 identifies the $\mathsf{FilterBase}^{\uparrow}$ boundary with that same displayed handoff rather than with a host filter equality.

Finally pass to completion-facing readback. The stream route Definition 1667.12 and real-seal nonescape theorem Definition 1667.10 force the handoff through S, R, L, and the rows H, C, P, M preserve only this displayed route under transport, replay, provenance, and local naming. Thus the Moore–Smith/filter comparison is exhausted by finite carrier inspection, with no selected subnet, quotient net, cofinal-choice operator, host convergence predicate, or ambient completeness witness. □

Theorem 1667.9 (Net convergence filterbase finality)**.** *Every* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier factors through the displayed directed tail* D, T, E *and the handoff row* F*. No host filter equality, selected limit, quotient net, ambient completeness witness, or alternate filter coordinate can refine that handoff beyond the finite route already displayed by the carrier.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier filterbase finality`

Proof. Project the carrier of Definition 1667.1. The rows D, T, E are the directed-index window, tail-neighbourhood ledger, and entourage row, while F is the only row that names the filter-facing handoff.

Apply Definition 1667.4. It sends every CauchyFilter-facing read through D, T, E, F before any Real-facing rows can contribute. Apply also Definition 1667.7, which identifies the Moore–Smith and filter boundary with the same displayed rows.

The carrier has no second filterbase coordinate, no selected limit coordinate, and no quotient-net coordinate. Thus every filterbase or CauchyFilter consumer factors through the displayed handoff, and that handoff is final among the routes accepted by the packet. □

Theorem 1667.10 (Net convergence Real seal nonescape)**.** *Let* $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ *be an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier. The terminal* $\mathsf{Real}^{\uparrow}$ *seal* L *is reachable only after the directed window* D*, tail-neighbourhood ledger* T*, entourage row* E*, CauchyFilter handoff* F*, StreamName schedule* S*, and RegSeqRat readback* R *have been displayed. The seal cannot be selected as a host limit, quotient-net representative, ambient convergence witness, or completeness object.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier real seal nonescape`

Proof. Project the carrier of Definition 1667.1. The rows preceding the Real seal are exactly the directed-tail rows D, T, E, the filter-facing row F, and the Real-facing readback rows S, R.

Use Definition 1667.2. Its carrier admission and Real seal non-escape clauses force L to be read only after the displayed analytic rows have appeared.

Use also Definition 1667.5, Definition 1667.4, and Definition 1667.6. Together they leave no route from a host limit, quotient net, ambient convergence predicate, or completeness theorem into the seal coordinate. □

Theorem 1667.11 (Net convergence entourage-tail stability)**.** *Let $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ and $N' = (D', T', E', A', F', S', R', L', H', C', P', M')$ be accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carriers related by the componentwise* hsame *transport recorded by their structural rows. Transport of D, T, E preserves the same tail-neighbourhood and entourage route for every* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer, so the handoff remains a read through D, T, E, F rather than a new host filter or ambient uniformity.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier entourage tail stability`

Proof. Read the structural transport row in Definition 1667.1. It transports the directed window, tail-neighbourhood ledger, entourage row, optional source row, filter handoff, stream schedule, readback, seal, provenance, and local naming component by component.

Apply Definition 1667.2 and Definition 1667.5. A convergence read must first expose D, T, E, and the transported carrier exposes the corresponding D', T', E' through the same structural coordinate.

Finally apply Definition 1667.4 and Definition 1667.6. Since every filter-facing read is already routed through the directed tail and entourage rows before F is consumed, componentwise hsame transport preserves the route and adds no host filter equality or ambient uniformity coordinate. □

Theorem 1667.12 (Net convergence stream readback route)**.** *Let $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ be an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier. After the CauchyFilter handoff row F has been displayed, the only Real-facing route available to the packet is the* $\mathsf{StreamName}^{\uparrow}$ *schedule S followed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and the terminal seal L. No alternate real-name route, selected limit, quotient-net representative, or ambient completeness row is exported.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier stream readback route`

Proof. Start with the carrier projection of Definition 1667.1. The rows downstream of F on the Real-facing side are S, R, L, together with the structural rows H, C, P, M.

Apply Definition 1667.2. The listed obligations include the CauchyFilter handoff, Real seal non-escape, componentwise transport, replay, provenance, and local naming; none of them introduces a second real-name coordinate.

Apply Definition 1667.5, Definition 1667.4, and Definition 1667.6. They force the route through D, T, E, F before any Real-facing read and then through S, R, L, so the stream schedule and regular readback are the only Real-facing route after the handoff. □

Theorem 1667.13 (Net convergence tail-window admission)**.** *Let $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ be an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier. Every accepted convergence read first exposes a directed tail window in D together with the tail-neighbourhood request recorded in T and the entourage interpretation row E. Only after that route is displayed may the packet read a present* $\mathsf{CauchyNet}^{\uparrow}$ *row A, the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff F, or the Real-facing schedule S, R, L.*

Proof. Read the carrier rows of Definition 1667.1. The only coordinates that name a directed tail and interpret a neighbourhood request are D, T, E.

Apply the tail classification and entourage interpretation clauses of Definition 1667.2. They put every convergence read on the displayed directed-tail route before the optional source row, filter handoff, stream schedule, regular-rational readback, or Real seal can be consumed.

The downstream rows therefore depend on the tail-window admission route. Since the carrier has no second directed-tail coordinate, no consumer can bypass D, T, E by reading a host convergence predicate or a cofinal-choice operator. □

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier tail window admission`
Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier tail window real seal bridge`

Theorem 1667.14 (Net convergence boundary readback)**.** *For an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier* $N = (D, T, E, A, F, S, R, L, H, C, P, M)$*, every Cauchy-facing read and every Real-facing read meet at the displayed boundary*

$$D, T, E \longmapsto F \longmapsto S, R, L.$$

The $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *handoff rows may be read only through the same directed tail, and the* $\mathsf{Real}^{\uparrow}$ *seal is read only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *boundary row has been displayed.*

Proof. Start with the Cauchy-facing side. By Definition 1667.4, the filter handoff F is reached through D, T, E, and by Definition 1667.6, a present source-net row A is read through that same directed route.

Now read the Real-facing side. The stream readback theorem Definition 1667.12 forces every Real read after F through S and R before it reaches the terminal seal L.

Thus the boundary is visible on both sides: directed tails supply the Cauchy handoff, and regular-rational readback supplies the Real seal. The packet does not expose a selected limit, quotient net, host convergence predicate, or ambient completeness row that could replace this boundary. □

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier boundary readback`

Theorem 1667.15 (Net convergence ledger nonescape)**.** *Every consumer of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *packet factors through the finite ledger rows* $D, T, E, A, F, S, R, L, H, C, P, M$*. The packet refuses quotient nets, selected limits, cofinal-choice operators, host filter equality, ambient convergence predicates, and ambient completeness; none of these objects can be recovered from transport, replay, provenance, or local naming.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier ledger nonescape`

Proof. Project the accepted packet. Its analytic coordinates are the directed window, tail ledger, entourage row, optional source row, filter handoff, stream schedule, regular-rational readback, and Real seal; its structural coordinates are transport, replay, provenance, and local naming.

Apply Definition 1667.13 and Definition 1667.14. Together they place every accepted read on the finite directed-tail and Cauchy-to-Real boundary route.

The structural rows cannot add analytic content. Componentwise hsame transport and displayed Cont replay preserve only the listed coordinates, while P and M source and name only the same packet. Hence quotient nets, selected limits, cofinal choice, host filter equality, ambient convergence, and completeness remain outside the certificate. □

Theorem 1667.16 (Net convergence to sequential compact dependency)**.** *When an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier* $N = (D, T, E, A, F, S, R, L, H, C, P, M)$ *is consumed by the sequential-compact cluster route of chapter 2372, the directed-tail rows* D, T, E, F, S, R, L *must be displayed before any* $\mathsf{SequentialCompact}^{\uparrow}$ *finite-window cluster row is read. The consumer therefore sees the directed window, tail-neighbourhood ledger, entourage interpretation, CauchyFilter handoff, StreamName schedule, RegSeqRat readback, and Real seal as the dependency surface for the sequential-compact finite-window packet. It does not receive a selected subnet, quotient net, countable-choice operator, host convergence predicate, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier sequentialcompact dependency`

Proof. Project the accepted carrier. The directed-tail rows D, T, E are the only rows that expose the Moore–Smith window and entourage request, while F, S, R, L are the only rows that hand the same route to CauchyFilter, StreamName, RegSeqRat, and Real-facing consumers.

Apply Definition 1667.5, Definition 1667.4, and Definition 1667.12. These place every downstream read behind the displayed directed-tail route and then behind the filter and stream-readback handoff.

The $\mathsf{SequentialCompact}^{\uparrow}$ finite-window cluster rows can therefore read N only through D, T, E, F, S, R, L and the structural rows H, C, P, M. Since no carrier coordinate stores a selected subnet, quotient-net representative, countable-choice operator, host convergence predicate, or ambient completeness witness, none of those objects is exported to the sequential-compact route. □

Theorem 1667.17 (Net convergence to SequentialCompact bridge)**.** *The displayed* $\mathsf{NetConvergence}^{\uparrow}$ *directed-tail route bridges to the* $\mathsf{SequentialCompact}^{\uparrow}$ *finite-window consumer only through*

$$D, T, E, F, S, R, L, H, C, P, M.$$

The bridge exports no selected subnet, quotient net, countable-choice operator, host convergence predicate, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier sequentialcompact bridge`

Proof. First expose the directed-tail route. Apply Definition 1667.5 to place every convergence read behind D, T, E, and apply Definition 1667.4 to route the filter-facing handoff through F.

Next expose the Real-facing boundary. Apply Definition 1667.12 and Definition 1667.14; they force the handoff through the StreamName schedule S, RegSeqRat readback R, and terminal Real seal L before any downstream finite-window consumer reads the route.

Finally bind the non-escape and downstream dependency. Apply Definition 1667.15 and Definition 1667.16. The only structural rows left for transport, replay, provenance, and naming are H, C, P, M, so the SequentialCompact consumer receives the displayed bridge and no selected subnet, quotient net, choice operator, host convergence predicate, or ambient completeness principle. □

Theorem 1667.18 (Net convergence mature source package)**.** *The* $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers of an accepted* $\mathsf{NetConvergence}^{\uparrow}$ *carrier all meet at the finite route*

$$D, T, E, F, S, R, L, H, C, P, M.$$

The package exports no selected limit, quotient net, cofinal-choice operator, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.NetConvergenceUp.NetConvergenceCarrier mature source package`

Proof. Start with the Moore–Smith dependency theorem Definition 1667.3. It supplies the directed window and tail evidence consumed by the source side of the packet, and Definition 1667.5 places every later read behind D, T, E.

Apply the Cauchy-facing handoff results. The CauchyFilter handoff theorem Definition 1667.4, the CauchyNetLimit handoff theorem Definition 1667.6, and filterbase finality Definition 1667.9 route the filter and optional source reads through the same D, T, E, F boundary.

Now apply the Real-facing non-escape theorem Definition 1667.10. The Real seal is reachable only after S, R, L have been displayed downstream of the filter handoff.

The structural rows H, C, P, M transport, replay, source, and name this finite package. Since Definition 1667.15 excludes quotient nets, selected limits, cofinal choice, host convergence, and ambient completeness, the listed source-package route is the mature public surface available to downstream consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{NetConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{NetConvergence}^{\uparrow}$ packet is generated from a directed-index window, a tail-neighbourhood ledger, a UniformSpace entourage row, an optional CauchyNet source row, a CauchyFilter handoff row, StreamName windows, RegSeqRat readback, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NetConvergence}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1667.1, Definition 1667.2, Definition 1667.4, Definition 1667.13, Definition 1667.14, Definition 1667.15, Definition 1667.16, Definition 1667.17, and Definition 1667.18 bind the displayed obligations and the public downstream route from $\mathsf{NetConvergence}^{\uparrow}$ to $\mathsf{SequentialCompact}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NetConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient nets, countable choice, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Bridge checking requires a Lean standard-bridge theorem for the displayed NetConvergence to SequentialCompact route; the paper-side mature source package is closed here.

Chapter 1668

A Concrete Naming Certificate for $\mathsf{RealRationalApproximation}^{\uparrow}$

Bishop and Bridges-style constructive analysis reads a real name through explicit rational approximants at finite precision, not through a host real value or a completed quotient stream. The $\mathsf{RealRationalApproximation}^{\uparrow}$ packet records that finite approximation window as BEDC data between the displayed real-name seal of chapter 38, the regular rational readback of chapter 301, the finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the rational boundedness interface of chapter 430. It is a rational-approximation face for Real and Completion consumers; it is not host real equality, a best-approximant selector, trichotomy, or ambient completeness.

Definition 1668.1 (RealRationalApproximation carrier)**.** *A* $\mathsf{RealRationalApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, Q, D, S, G, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal whose finite rational approximation window is being displayed.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *approximation row exported at the requested precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic budget under which Q is accepted as an approximant.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window opened from R before the rational approximation is read.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S, recording the regular rational observations used to justify Q.*
6. *A is the approximation ledger comparing Q with the displayed readback rows under the dyadic tolerance budget D.*
7. *H stores componentwise* hsame *transport for the real seal, rational approximation row, dyadic tolerance row, stream window, regular readback, approximation ledger, replay, provenance, and local name.*
8. *C stores displayed* Cont *replay from a finite precision request through S, G, D, A, Q and then into the terminal real seal R.*
9. *P records* Pkg *provenance over* $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealRationalApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient streams, selected best approximants, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealRationalApproximationUp.TasteGate.RealRationalApproximationTasteGate single carrier alignment`

Theorem 1668.2 (RealRationalApproximation NameCert obligations)**.** *For every accepted* RealRationalApproxima *carrier $W = (R, Q, D, S, G, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealRationalApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Rat}^{\uparrow}$ *approximation row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, approximation ledger, transport, replay, provenance, and local naming rows;*
2. *window admission, requiring every approximation request to open the finite stream window S from the real seal R before reading any rational row;*
3. *tolerance admission, requiring the dyadic budget D to be displayed before the rational approximant Q may be accepted;*
4. *readback comparison, requiring the ledger A to compare Q against the regular rational observations in G under the same dyadic budget D;*
5. *transport and replay stability, requiring H and C to preserve the ordered route R, S, G, D, A, Q componentwise;*
6. *ledger nonescape from host real equality, quotient streams, selected best approximants, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealRationalApproximationUp.RealRationalApproximationTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1668.1. The accepted packet displays exactly the real seal R, rational row Q, dyadic tolerance row D, stream window S, regular readback G, approximation ledger A, and the structural rows H, C, P, N.

Window admission and tolerance admission are forced by the ordering of the packet. A finite approximation request first opens S from the displayed real seal, reads the regular rational row G, and fixes the dyadic budget D before Q can be consumed as an approximant.

The comparison row is the ledger A. It records the finite check that the displayed rational row Q matches the readback observations in G at the tolerance named by D, so the approximation surface is exhausted by the rows already present in the carrier.

Transport and replay are structural. The hsame row H moves each displayed coordinate componentwise, and the Cont row C repeats the same finite route for later consumers.

The nonescape clause follows because the carrier has no row for host real equality, quotient streams, selected best approximants, trichotomy, or ambient completeness. The local $\mathsf{NameCert}$ row N can therefore export only the displayed approximation window. □

Theorem 1668.3 (RealRationalApproximation window extraction)**.** *Every accepted* $\mathsf{RealRationalApproximation}^{\uparrow}$ *carrier $W = (R, Q, D, S, G, A, H, C, P, N)$ exposes a finite rational approximation window*

$$R \longrightarrow S \longrightarrow G \longrightarrow D \longrightarrow A \longrightarrow Q$$

for the displayed $\mathsf{Real}^{\uparrow}$ *name. A Real or Completion consumer receives the rational approximant only after the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows have been displayed with their transport, replay, provenance, and local naming data.*

Lean4 ✓ `BEDC.Derived.RealRationalApproximationUp.RealRationalApproximationCarrier window extraction`

Proof. Start from an accepted packet $W = (R, Q, D, S, G, A, H, C, P, N)$. Projection gives the terminal real seal R and the finite stream window S opened from it.

The regular readback row G is then read through the same window. The dyadic tolerance row D supplies the precision budget for the request, and the ledger A compares the candidate rational row Q with the displayed observations under that budget.

Thus the consumer obtains Q only through the ordered finite route R, S, G, D, A. The remaining rows H, C, P, N transport, replay, source, and name the same route, so no detached real value or quotient stream is required to extract the approximation window. □

Theorem 1668.4 (RealRationalApproximation Real seal handoff)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RealRationalApproximation}^{\uparrow}$ *packet hands off exactly the displayed approximation rows* $Q, D, S, G, A, H, C, P, N$. *It exports no host real equality, quotient stream, selected best approximant, trichotomy datum, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.RealRationalApproximationUp.RealRationalApproximationRealSealHandoff`

Proof. The real-facing boundary is the coordinate R of Definition 1668.1. Every public approximation read attached to R must pass through the finite window S, the regular readback G, the dyadic tolerance row D, the ledger A, and the rational row Q.

The structural rows do not enlarge the boundary. They preserve the same displayed route by componentwise transport, continuation replay, package provenance, and local naming.

Since no excluded object is a coordinate of the carrier, the terminal seal has no path to host real equality, quotient streams, selected best approximants, trichotomy, or ambient completeness. The handoff is exactly the finite rational-approximation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealRationalApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{RealRationalApproximation}^{\uparrow}$ packet is generated from a displayed RealUp seal, a RatUp approximation row, a DyadicRatCoreUp tolerance row, a StreamNameUp finite window, a RegSeqRatUp readback row, an approximation ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealRationalApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1668.1, Definition 1668.2, Definition 1668.3, and Definition 1668.4 seed the finite rational approximation window over displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealRationalApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient streams, selected best approximants, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window admission, tolerance admission, readback comparison, RealUp seal handoff, componentwise transport and replay stability, and ledger nonescape for the same finite packet.

Chapter 1669

RealLocatorUp NameCert Construction

Following Bishop and Bridges, the $\mathsf{RealLocator}^{\uparrow}$ packet records constructive located comparison data for a displayed $\mathsf{Real}^{\uparrow}$ name without turning that data into trichotomy or Dedekind-cut reconstruction. It sits beside the terminal real-name seal of chapter 38, the regular rational readback rows of chapter 301, and the dyadic tolerance ledger of chapter 291. The public role is a finite comparison service for later real algebra, metric locatedness, and apartness consumers.

Definition 1669.1 (Real locator carrier)**.** *A* $\mathsf{RealLocator}^{\uparrow}$ *carrier is finite BEDC data that answers each dyadic rational comparison request by a displayed lower or upper apartness window for the named real. Concretely it is a finite* BHist *packet*

$$L = (X, Q, S, R, D, A, H, C, P, N)$$

with the following rows:

1. *X is the terminal* $\mathsf{Real}^{\uparrow}$ *name whose seal is inspected by the locator.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison request naming the queried dyadic rational and the requested finite precision.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation window opened from the real name for that request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the observed stream window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger that records the dyadic lower and upper rational bounds used by the comparison.*
6. *A is the displayed apartness-window row, read either as a lower witness or as an upper witness for the request, with the undecided boundary left uncollapsed.*
7. *H records componentwise* hsame *transport for the real seal, dyadic request, stream window, regular readback, dyadic ledger, apartness row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay through $X, Q, S, R, D, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocator}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite locator packet. It has no coordinate for Dedekind-cut completeness, real trichotomy, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealLocatorUp.RealLocatorTasteGate single carrier alignment`

Theorem 1669.2 (Real locator NameCert obligations)**.** *For every accepted* $\mathsf{RealLocator}^{\uparrow}$ *carrier* $L = (X, Q, S, R, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocator}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier stability, requiring the real seal, dyadic request, stream observation window, regular readback, dyadic ledger, apartness row, transport, replay, provenance, and local naming data to inhabit one finite* BHist *packet;*
2. *classifier stability, requiring the lower or upper apartness verdict in* A *to be read only through the displayed request* Q *and observation route* S, R, D*;*
3. *transport along* hsame*, requiring* H *to preserve the same request, window, readback, dyadic ledger, and apartness row under componentwise transport;*
4. *dyadic ledger exactness, requiring* D *to record the lower and upper dyadic bounds used by the displayed comparison and not a host-order comparison of completed reals;*
5. $\mathsf{Real}^{\uparrow}$ *seal compatibility, requiring* X *to be consumed only through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows before the apartness window is read.*

No NameCert obligation may replace these rows by Dedekind-cut completeness, real trichotomy, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealLocatorUp.RealLocatorNameCertObligations`

Proof. Project the accepted carrier L. Its analytic coordinates are exactly the real seal X, dyadic request Q, stream window S, regular readback R, dyadic ledger D, and apartness row A; the remaining rows transport, replay, source, and name the same packet.

Classifier stability follows from the order of projection. The apartness row A is reached only after the request and finite observation route Q, S, R, D have been displayed, so its verdict cannot be detached from those rows.

Transport stability is supplied by H and replay stability by C. They preserve the same listed coordinates under hsame transport and displayed continuation replay.

The dyadic ledger and real seal boundaries close the obligation surface. Since D is the only dyadic comparison ledger and X is consumed only through S, R, D, the local NameCert has no field from which Dedekind-cut completeness, trichotomy, quotient equality, countable choice, or ambient completeness could be read. □

Theorem 1669.3 (Real locator apartness decision window)**.** *For every accepted* $\mathsf{RealLocator}^{\uparrow}$ *carrier* $L = (X, Q, S, R, D, A, H, C, P, N)$*, a displayed locator decision in* A *yields a finite apartness readback window through*

$$Q \longmapsto S \longmapsto R \longmapsto D \longmapsto A \longmapsto H, C, P, N.$$

The readback exposes only the lower or upper apartness window carried by A *for the request* Q*. It does not assert trichotomy for all real comparisons and does not identify quotient representatives of* X*.*

Proof. Start with the displayed decision row A. By the carrier definition, A is not an isolated truth value; it is attached to the dyadic request Q, the stream observation window S, the regular readback R, and the dyadic ledger D.

The finite readback window is therefore obtained by replaying that route. The rows H and C transport and replay the same request, observation, ledger, and apartness data, while P and N package the resulting local certificate.

The conclusion is intentionally local. Because the packet has no coordinate for a global order classifier, quotient representative equality, countable choice, or ambient completeness, the displayed decision supplies only the finite apartness window for Q. □

Lean4 ✓ `BEDC.Derived.RealLocatorUp.RealLocatorApartnessDecisionWindow`

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocator}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealLocator}\uparrow}$, seed)

Scope closed: Definition 1669.1, Definition 1669.2, and Definition 1669.3 seed the finite located comparison surface for $\mathsf{Real}^{\uparrow}$ names through stream, regular rational, and dyadic readback rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealLocator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut completeness, real trichotomy, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, transport along hsame, dyadic ledger exactness, RealUp seal compatibility, and apartness-window readback for BEDC.Derived.RealLocatorUp.

Constructive story: A $\mathsf{RealLocator}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{DyadicRatCore}^{\uparrow}$ comparison request, a $\mathsf{StreamName}^{\uparrow}$ observation window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a dyadic lower-upper ledger, a displayed apartness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1670

A Concrete Naming Certificate for $\mathsf{RegularCauchyMidpoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ packet names midpoint formation for two regular rational Cauchy names by finite dyadic arithmetic and displayed modulus rows before the completed $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the dyadic arithmetic core of chapter 291, the stream-window schedule of chapter 126, the regular rational readback rows of chapter 301, the pointwise midpoint ledger of chapter 445, and the terminal real-name seal of chapter 38. The packet supplies the finite midpoint surface needed by interval bisection and Real-facing finite-window algebra; it does not import host real arithmetic, quotient streams, selected limits, trichotomy, or ambient completeness.

Definition 1670.1 (Regular Cauchy midpoint carrier)**.** *A* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{mid}} = (R_0, R_1, S_0, S_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. R_1 *is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
3. S_0 *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the left name, recording exactly which rational approximants are inspected for each requested midpoint precision.*
4. S_1 *is the matching* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the right name, synchronized with* S_0 *at the common precision ledger.*
5. D *is the* $\mathsf{DyadicMidpoint}^{\uparrow}$ *pointwise midpoint row, reading the two displayed rational approximants through the dyadic endpoint and common-denominator midpoint calculation.*
6. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, common denominator budget, and regularity allowance used to certify that the pointwise midpoint row is again regular Cauchy.*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the two regular names, synchronized schedules, dyadic midpoint row, and tolerance ledger have been displayed.*
8. H *records componentwise* hsame *transport for the two regular rows, two stream schedules, pointwise midpoint row, tolerance ledger, Real seal row, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *routes from every midpoint-facing read through $R_0, R_1, S_0, S_1, D, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyMidpoint$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicMidpoint$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyMidpoint}^\uparrow}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, S_0, S_1, D, Q, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, host real midpoint arithmetic, selected limits, trichotomy, ambient completeness, or a detached completed-real midpoint.

Lean4 ✓ `BEDC.Derived.RegularCauchyMidpointUp.RegularCauchyMidpointTasteGate single carrier alignment`

Theorem 1670.2 (Regular Cauchy midpoint NameCert obligations)**.** *For every accepted* RegularCauchyMidpoint$^\uparrow$ *carrier*

$$M_{\text{mid}} = (R_0, R_1, S_0, S_1, D, Q, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMidpoint}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* RegSeqRat$^\uparrow$ *source rows R_0, R_1, the synchronized* StreamName$^\uparrow$ *schedules S_0, S_1, the* DyadicMidpoint$^\uparrow$ *pointwise midpoint row D, the* DyadicRatCore$^\uparrow$ *tolerance ledger Q, the* Real$^\uparrow$ *seal row E, and the structural rows H, C, P, N;*
2. *synchronized-window exposure, requiring every midpoint-facing read to inspect the two stream schedules at a common finite precision before the dyadic midpoint row is read;*
3. *pointwise midpoint exactness, requiring D to be obtained through the displayed* DyadicMidpoint$^\uparrow$ *common-denominator calculation on the two rational approximants exposed by S_0 and S_1;*
4. *regularity tolerance accounting, requiring Q to record the dyadic radius and the allowance that turns the two input regularity rows into the output regularity row;*
5. *Real seal boundary, requiring E to be consumed only after the route through R_0, R_1, S_0, S_1, D, Q has been displayed;*
6. *transport and replay stability, requiring H and C to preserve the same two regular names, synchronized schedules, midpoint row, tolerance ledger, Real seal boundary, provenance, and local naming row;*
7. *package provenance and local naming, requiring P and N to name exactly the listed sibling certificates and the same finite midpoint packet;*
8. *refusal of quotient streams, host real midpoint arithmetic, selected limits, trichotomy, and ambient completeness as NameCert data.*

No NameCert obligation is stored outside these finite rows.

Proof. Project the accepted carrier. The analytic coordinates are the two regular rational names R_0, R_1, the synchronized schedules S_0, S_1, the dyadic midpoint row D, the tolerance ledger Q, and the Real seal row E; the remaining rows transport, replay, source, and name the same finite packet.

Read the midpoint route in dependency order. A precision request first opens the paired finite schedules S_0 and S_1 from the two RegSeqRat$^\uparrow$ rows. The displayed rational approximants then enter the DyadicMidpoint$^\uparrow$ row D, where the common-denominator midpoint calculation is the only pointwise midpoint source.

The regularity ledger is the next step. The row Q records the dyadic radius and the allowance inherited from the two input regular Cauchy rows, so the midpoint readback remains a regular Cauchy name before the $\mathsf{Real}^{\uparrow}$ seal is reached.

Finally apply componentwise transport and replay. The rows H and C preserve the same displayed route, while P and N name exactly the sibling certificate dependencies. Since no carrier coordinate stores quotient stream equality, host real arithmetic, selected limits, trichotomy, or ambient completeness, none of those principles can be exported by the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMidpointUp.RegularCauchyMidpointNameCertObligations`

Theorem 1670.3 (Regular Cauchy midpoint Real seal handoff)**.** *Every accepted* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *carrier factors its Real-facing midpoint read through the finite route*

$$(R_0, R_1) \longmapsto (S_0, S_1) \longmapsto D \longmapsto Q \longmapsto E$$

before it may hand the result to a $\mathsf{Real}^{\uparrow}$ *consumer. Thus the consumer receives two regular rational source rows, synchronized finite schedules, a pointwise* $\mathsf{DyadicMidpoint}^{\uparrow}$ *row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and a Real seal row; it receives no quotient stream equality, host real midpoint operation, selected limit, trichotomy, or ambient completeness coordinate.*

Proof. Start with an accepted carrier $M_{\mathrm{mid}} = (R_0, R_1, S_0, S_1, D, Q, E, H, C, P, N)$. The only source rows for the midpoint read are the two regular rational names R_0 and R_1, and the only observation schedules attached to them are S_0 and S_1.

The pointwise midpoint then passes through D. That row consumes the displayed rational approximants through the same finite common-denominator midpoint calculation named by chapter 445; no second midpoint row is present.

The tolerance and seal rows finish the handoff. The row Q records the dyadic budget that witnesses regularity of the midpoint name, and E is reached only after this finite ledger has been displayed.

Replay the route through C and transport it through H. The provenance row P points to chapter 291, chapter 445, chapter 126, chapter 301, and chapter 38. Together with N, it packages the same finite route and adds no coordinate for quotient stream equality, host arithmetic, selected limits, trichotomy, or completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMidpointUp.RegularCauchyMidpointRealSealHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMidpoint}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMidpoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1670.1, Definition 1670.2, and Definition 1670.3 seed the finite midpoint surface for two $\mathsf{RegSeqRat}^{\uparrow}$ rows over synchronized $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{DyadicMidpoint}^{\uparrow}$ pointwise midpoint rows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMidpoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real arithmetic, selected limits, trichotomy, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, pointwise midpoint exactness, regularity tolerance accounting, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMidpointUp.

Constructive story: A $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ rows, synchronized $\mathsf{StreamName}^{\uparrow}$ schedules, a $\mathsf{DyadicMidpoint}^{\uparrow}$ pointwise midpoint row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1671

A Concrete Naming Certificate for $\mathsf{CauchyRateComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRateComparison}^{\uparrow}$ packet records the finite comparison surface for two effective Cauchy-rate reads before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer may inspect a seal. It sits over the effective-rate route of chapter 415, the modulus schedule surface of chapter 292, the dyadic tail-bound ledger of chapter 379, the common finite-window interface of chapter 126, the regular rational readback boundary of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later regular-Cauchy, completion, and phase-exit consumers a named common-refinement certificate instead of requiring each consumer to coordinate two rate rows informally.

Definition 1671.1 (Cauchy rate comparison carrier)**.** *A* $\mathsf{CauchyRateComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{cmp}} = (R_0, R_1, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two accepted* $\mathsf{CauchyRate}^{\uparrow}$ *rows whose effective reads are compared.*
2. W *is the common* $\mathsf{StreamName}^{\uparrow}$ *window row through which both rate reads expose the same finite observation schedule.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the shared precision budget, dyadic radius, and tail allowance used by the comparison.*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback comparison row obtained after both rate rows have been routed through* W *and* D*.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after* R_0, R_1, W, D, Q *have all been displayed.*
6. H *records componentwise* hsame *transports for the two rate rows, common window, dyadic tolerance ledger, readback comparison row, seal boundary, replay row, provenance row, and local naming row.*
7. C *records displayed* Cont *routes from the two accepted rate reads to the common window, from the window to the dyadic tolerance ledger, from the ledger to the regular rational comparison row, and from that row to the Real seal boundary.*
8. P *is the* Pkg *provenance over* $\mathsf{CauchyRateComparison}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRateComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed rows $R_0, R_1, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real comparison, selected limits, countable choice, trichotomy, ambient completeness, or an unrecorded completion object.

Lean4 ✓ `BEDC.Derived.CauchyRateComparisonUp.CauchyRateComparisonTasteGate single carrier alignment`

Theorem 1671.2 (Cauchy rate comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateComparison}^{\uparrow}$ *carrier*

$$K_{\mathrm{cmp}} = (R_0, R_1, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRateComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two* $\mathsf{CauchyRate}^{\uparrow}$ *rows, the common* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback comparison row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, and the structural rows H, C, P, N;*
2. *classifier exactness, requiring accepted comparison packets to be compared only by componentwise* hsame *transport and displayed* Cont *routes over those coordinates;*
3. *common-window refinement, requiring both rate reads to expose the same finite observation window W before either read reaches D, Q, or E;*
4. *transport, requiring the stored* hsame *rows in H to preserve the two rate reads, the common window, the tolerance ledger, the readback row, the seal boundary, provenance, and local naming;*
5. *non-escape, requiring every Real-facing or completion-facing read to pass through R_0, R_1, W, D, Q, E and refusing quotient stream equality, host real comparison, selected limits, countable choice, trichotomy, and ambient completeness as NameCert data.*

No comparison obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.CauchyRateComparisonUp.CauchyRateComparisonNameCertObligations`

Proof. Project the accepted carrier. Its analytic coordinates are exactly the two $\mathsf{CauchyRate}^{\uparrow}$ rows, the common $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ comparison readback, and the $\mathsf{Real}^{\uparrow}$ seal boundary.

Read the route in dependency order. The two rate rows must first expose the same finite window W; the dyadic ledger D is then read against that shared window, and only then can the regular rational comparison row Q be consumed.

Transport and replay are structural. The rows H and C preserve the displayed route under componentwise sameness and continuation reads, while P and N name exactly the sibling certificate dependencies and the local packet.

Since the carrier has no extra coordinate for quotient stream equality, host real comparison, selected limits, countable choice, trichotomy, ambient completeness, or an unrecorded completion object, none of those principles can enter the local NameCert surface. □

Theorem 1671.3 (Cauchy rate comparison common refinement)**.** *Let two accepted effective-rate reads be admitted into a* $\mathsf{CauchyRateComparison}^{\uparrow}$ *carrier*

$$K_{\mathrm{cmp}} = (R_0, R_1, W, D, Q, E, H, C, P, N).$$

If the two reads share the displayed window row W, then both reads factor through the same W, D, Q, E route before any $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer can read a seal. Thus the comparison exports a common finite refinement of the two* $\mathsf{CauchyRate}^{\uparrow}$ *rows, not a quotient equality of streams, a selected host limit, an ambient completeness principle, or a host-level real comparison.*

Proof. Start from the shared window hypothesis. Both accepted reads are carried by R_0 and R_1, and the carrier contains a single $\mathsf{StreamName}^{\uparrow}$ row W as the only finite observation window available to either read.

The tolerance and readback rows are downstream of that window. The displayed route in C sends the shared window to the dyadic ledger D and then to the regular rational comparison row Q, so the two reads cannot diverge into separate tolerance ledgers or separate readback rows inside the packet.

The seal boundary is last. The row E is reachable only after R_0, R_1, W, D, Q have been displayed, so any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the same common-refinement route before it can inspect a seal.

Finally apply the classifier and provenance rows. Componentwise transport through H preserves the same route, and P, N name no quotient stream equality, selected limit, ambient completeness principle, or host-level real comparison. The exported object is therefore exactly the finite common refinement of the two effective-rate reads. □

Lean4 ✓ `BEDC.Derived.CauchyRateComparisonUp.CauchyRateComparisonCommonRefinement`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1671.1, Definition 1671.2, and Definition 1671.3 seed the finite common-refinement surface for two $\mathsf{CauchyRate}^{\uparrow}$ rows over a shared $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback comparison row, and $\mathsf{Real}^{\uparrow}$ seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateComparison}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyRateComparisonUp.CauchyRateComparisonCommonRefinement`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real comparison, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, common-window refinement, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchyRateComparisonUp.

Constructive story: A $\mathsf{CauchyRateComparison}^{\uparrow}$ packet is generated from two $\mathsf{CauchyRate}^{\uparrow}$ rows, a common $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback comparison row, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1672

A Concrete Naming Certificate for $\mathsf{DyadicArchimedean}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicArchimedean}^{\uparrow}$ packet names the finite scale-domination certificate used before a dyadic tolerance route may choose a displayed scale large enough to dominate a displayed dyadic bound. It sits after the dyadic endpoint core of chapter 291, the common-scale order surface of chapter 1597, the comparison replay of chapter 1623, the endpoint enclosure handoff of chapter 1645, and the dyadic density route of chapter 1484. It is also consumed by the finite stream windows of chapter 126, the regular rational route of chapter 301, and the terminal real-name seal of chapter 38. The packet records scale domination as unary induction over displayed dyadic rows; it is not trichotomy, quotient equality, host rational order, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1672.1 (DyadicArchimedean carrier)**.** *A* $\mathsf{DyadicArchimedean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, B, K, S, C, I, H, T, P, N)$$

with the following displayed rows:

1. *D is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row whose denominator and numerator histories are the source of the tolerance comparison.*
2. *B is the target dyadic bound row, displayed over the same dyadic core surface as the quantity to be dominated.*
3. *K is a unary scale witness, naming a finite number of successor-scale steps from the source tolerance scale to a scale whose dyadic unit dominates B.*
4. *S is the common-scale replay row, transporting D and B to the comparison scale named by K without invoking host rational order.*
5. *C is the* $\mathsf{DyadicComparison}^{\uparrow}$ *and* $\mathsf{DyadicOrder}^{\uparrow}$ *domination row, recording the finite comparison that the replayed scale dominates the replayed bound.*
6. *I is the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *enclosure handoff, exposing the same dominated scale as an endpoint enclosure for downstream tolerance consumers.*
7. *H records componentwise* hsame *transport for the dyadic source row, bound row, unary scale witness, common-scale replay, domination comparison, enclosure handoff, continuation, provenance, and local naming rows.*

8. *T records displayed* Cont *replay from scale requests through $D, B, K, S, C, I, H, P, N$.*
9. *P is the* Pkg *provenance over* DyadicArchimedean$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicOrder$^{\uparrow}$, DyadicComparison$^{\uparrow}$, DyadicIntervalArithmetic$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DyadicArchimedean}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite dyadic scale-domination packet. It has no coordinate for trichotomy, quotient equality, host rational order, selected limits, countable choice, or Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicArchimedeanUp.TasteGate.DyadicArchimedeanTasteGate single carrier alignment`

Theorem 1672.2 (DyadicArchimedean NameCert obligations)**.** *For every accepted* DyadicArchimedean$^{\uparrow}$ *carrier $A = (D, B, K, S, C, I, H, T, P, N)$, the local* NameCert$_{\text{DyadicArchimedean}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic source row D, target bound row B, unary scale witness K, common-scale replay S, domination comparison C, enclosure handoff I, and structural rows H, T, P, N;*
2. *scale-witness well-foundedness, requiring K to be read as a finite unary successor ledger rather than as an ambient Archimedean principle;*
3. *common-scale exactness, requiring S to replay D and B at the scale named by K through displayed dyadic denominator rows;*
4. *domination exposure, requiring C to certify domination only through the* DyadicComparison$^{\uparrow}$ *and* DyadicOrder$^{\uparrow}$ *rows produced by S;*
5. *enclosure compatibility, requiring I to export the dominated scale as a* DyadicIntervalArithmetic$^{\uparrow}$ *endpoint enclosure without changing the comparison route;*
6. *classifier transport, requiring H and T to preserve the same source row, bound row, scale witness, replay row, comparison row, and enclosure row;*
7. *ledger non-escape, excluding trichotomy, quotient equality, host rational order, selected limits, countable choice, and* Real$^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the accepted carrier. The analytic rows are D, B, K, S, C, I, while H, T, P, N transport, replay, source, and name the same finite packet.

The scale-witness obligation is read from K. Since K is unary, the only available eliminator is finite induction over the displayed successor ledger. The base row compares the initial dyadic unit with the replayed bound when the bound is already dominated at the source scale.

For the successor step, assume the preceding scale has been replayed through S and compared through C. The next unary constructor doubles the displayed dyadic unit at the common scale, and the comparison row is recomputed through the same DyadicComparison$^{\uparrow}$ and DyadicOrder$^{\uparrow}$ surface. No host rational order or trichotomy is introduced in this step.

The enclosure handoff is then read from I. It packages the dominated scale as a dyadic endpoint enclosure, so downstream tolerance consumers see the same comparison route rather than a separate real-order argument.

Finally H and T transport and replay the listed rows componentwise, while P and N record provenance and local naming. The carrier contains no coordinate from which trichotomy, quotient equality, host rational order, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness could be read. □

Lean4 ✓ `BEDC.Derived.DyadicArchimedeanUp.DyadicArchimedeanNameCertObligations`

Theorem 1672.3 (DyadicArchimedean scale existence)**.** *Every accepted* $\mathsf{DyadicArchimedean}^{\uparrow}$ *carrier* $A = (D, B, K, S, C, I, H, T, P, N)$ *exposes a finite dyadic scale that dominates its displayed target bound. More precisely, induction on the unary witness* K *produces a scale row* k *such that the replay route*

$$D, B, K \longmapsto S \longmapsto C$$

certifies that the dyadic unit at scale k *dominates the bound row* B *through displayed* $\mathsf{DyadicComparison}^{\uparrow}$ *and* $\mathsf{DyadicOrder}^{\uparrow}$ *data.*

Lean4 ✓ `BEDC.Derived.DyadicArchimedeanUp.DyadicArchimedeanScaleExistence`

Proof. Induct on the unary scale witness K.

In the base case, the carrier supplies the common-scale replay row S at the initial scale. The comparison row C is then the displayed domination certificate for the source dyadic unit against the replayed target bound.

In the successor case, apply the induction hypothesis to the predecessor witness. The successor constructor extends the scale ledger by one displayed doubling step, and S replays both D and B at the extended common scale. The comparison row C records the corresponding domination through the same dyadic comparison and order surfaces.

The induction returns the terminal scale row named by K. Since each step used only S and C, the resulting scale is finite and carries no appeal to host rational order, trichotomy, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1672.4 (DyadicArchimedean StreamName scale transport)**.** *For every accepted* $\mathsf{DyadicArchimedean}^{\uparrow}$ *carrier* $A = (D, B, K, S, C, I, H, T, P, N)$*, a* $\mathsf{StreamName}^{\uparrow}$ *tolerance consumer transports the unary scale witness* K *only through finite stream windows before* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumers read the enclosure row* I*. More precisely, the consumer first requests a finite* $\mathsf{StreamName}^{\uparrow}$ *window for the tolerance read, then reads the dyadic source and bound rows* D, B*, the unary scale ledger* K*, the common-scale replay* S*, and the domination row* C *inside that window, and only afterward receives the endpoint enclosure handoff* I *through* H, T, P, N*.*

Lean4 ✓ `BEDC.Derived.DyadicArchimedeanUp.DyadicArchimedeanStreamnameScaleTransport`

Proof. Begin with the scale existence theorem, Definition 1672.3. Its induction on K builds the dominating scale by finite successor steps and routes every comparison through S and C.

Now place the same induction inside a $\mathsf{StreamName}^{\uparrow}$ tolerance request. The stream window fixes the finite boundary of the read: it displays which tolerance row is being inspected before the dyadic source D, bound B, scale witness K, common-scale replay S, and domination comparison C are consumed.

Finally apply the L10 handoff theorem, Definition 1672.5. The enclosure row I is downstream of the displayed domination route, so $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive only the finite stream-window transport of the scale witness, with no trichotomy, quotient equality, host rational order, selected limit, countable-choice row, or Real-completeness witness. □

Theorem 1672.5 (DyadicArchimedean L10 handoff)**.** *For every accepted* $\mathsf{DyadicArchimedean}^{\uparrow}$ *carrier* $A = (D, B, K, S, C, I, H, T, P, N)$*, every L10 dyadic tolerance consumer reaches the domination certificate through the finite route*

$$D, B, K \longmapsto S \longmapsto C \longmapsto I \longmapsto H, T, P, N.$$

Consequently a $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicDensityReal}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer receives a displayed scale-domination and enclosure row, with no trichotomy, quotient equality, host rational order, selected limit, countable-choice row, or Real-completeness witness inserted into the route.*

Proof. Start with an accepted carrier $A = (D, B, K, S, C, I, H, T, P, N)$. The only source tolerance row is D, the only target bound row is B, and the only scale-selection data is the unary witness K.

By Definition 1672.3, finite induction on K yields a displayed dominating scale. The proof of that theorem uses the common-scale replay S and the comparison row C, so the exported domination certificate is already routed through $\mathsf{DyadicComparison}^{\uparrow}$ and $\mathsf{DyadicOrder}^{\uparrow}$.

The row I is the sole handoff to $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ enclosure data. It exports the dominated scale as an endpoint enclosure that a $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicDensityReal}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer may read after the dyadic comparison has been displayed.

The structural rows H, T, P, N preserve and name the route. Componentwise transport and continuation replay keep the same source row, bound row, scale witness, common-scale replay, comparison row, and enclosure handoff. Since every L10 tolerance read is reached through this route, there is no remaining coordinate from which trichotomy, quotient equality, host rational order, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness could be read. □

Lean4 ✓ `BEDC.Derived.DyadicArchimedeanUp.DyadicArchimedeanCarrier l10 handoff`

Theorem 1672.6 (DyadicArchimedean obligation closure ledger)**.** *For every accepted* $\mathsf{DyadicArchimedean}^{\uparrow}$ *carrier* $A = (D, B, K, S, C, I, H, T, P, N)$*, the obligation package is exactly carrier admission, unary scale-witness induction, common-scale replay, domination exposure, interval enclosure compatibility, classifier transport, and ledger non-escape. These rows jointly bind the full local* $\mathsf{NameCert}_{\mathsf{DyadicArchimedean}^{\uparrow}}$ *surface; no additional obligation is supplied by trichotomy, quotient equality, host rational order, selected limits, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the obligation list from Definition 1672.2. Carrier admission exposes the ten rows $D, B, K, S, C, I, H, T, P, N$, while the ledger non-escape clause excludes any off-packet analytic supply.

The scale-witness part is discharged by Definition 1672.3. Its induction is unary on K: the base and successor steps produce the displayed domination scale only through S and C.

The finite transport part is supplied by Definition 1672.4. A $\mathsf{StreamName}^{\uparrow}$ consumer reads the same D, B, K, S, C rows inside a finite window before $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumers receive the enclosure row I.

Finally Definition 1672.5 identifies the downstream route $D, B, K \longmapsto S \longmapsto C \longmapsto I \longmapsto H, T, P, N$. Thus the listed rows are not merely an inventory: they assemble the full obligation-closure ledger for the packet. □

Falsifiable prediction: An accepted $\mathsf{DyadicArchimedean}^{\uparrow}$ consumer that reaches an L10 dyadic tolerance scale without the unary witness K, common-scale replay S, domination row C, and enclosure handoff I refutes the carrier surface.

Independence witness: dyadicratcore, dyadicorder, dyadiccomparison, dyadicintervalarithmetic, dyadicdensityreal, streamname, regseqrat, real: no listed sibling supplies the whole ten-row packet $A = (D, B, K, S, C, I, H, T, P, N)$, because each sibling contributes dyadic rows, comparison rows, enclosure rows, finite windows, regular rational readback, density data, or a Real seal while this chapter binds finite scale domination with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicArchimedean}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicArchimedean}^{\uparrow}$ packet is generated from a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ source row, target dyadic bound row, unary scale witness, common-scale replay, dyadic domination comparison, interval-arithmetic enclosure handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicArchimedean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1672.1, Definition 1672.2, Definition 1672.3, and Definition 1672.5 seed the finite dyadic scale-domination surface over DyadicRatCore, DyadicOrder, DyadicComparison, DyadicIntervalArithmetic, DyadicDensityReal, StreamName, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicArchimedean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient equality, host rational order, selected limits, countable choice, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, scale-witness well-foundedness, common-scale exactness, domination exposure, enclosure compatibility, classifier transport, and ledger non-escape proofs for BEDC.Derived.DyadicArchimedeanUp.

Chapter 1673

A Concrete Naming Certificate for $\mathsf{CauchyRateRealization}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRateRealization}^{\uparrow}$ packet records the finite route by which an explicit Cauchy-rate datum realizes a terminal $\mathsf{Real}^{\uparrow}$ seal without hiding a selected limit. It sits over the effective-rate surface of chapter 415, the real-modulus route of chapter 1423, the finite-window surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet gives later Real-and-Completion consumers one named realization bridge from explicit rate data to the seal, while keeping quotient streams, selected limits, countable choice, host real equality, and ambient completeness outside the carrier.

Definition 1673.1 (Cauchy rate realization carrier)**.** *A* $\mathsf{CauchyRateRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{real}} = (p, r, d, s, q, m, e, H, C, P, N)$$

with the following displayed rows:

1. *p is the displayed precision-request row whose requested dyadic tolerance must be served.*
2. *r is the accepted* $\mathsf{CauchyRate}^{\uparrow}$ *row exposing the effective threshold schedule for the request p.*
3. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational endpoint and radius comparisons read at that threshold.*
4. *s is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the rate row r for the request p.*
5. *q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream window and dyadic tolerance ledger.*
6. *m is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *route row showing that the same rate, window, tolerance, and readback data serve the Real-facing modulus request.*
7. *e is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after p, r, d, s, q, m have all been displayed.*
8. *H records componentwise* hsame *transport for the precision request, rate row, dyadic ledger, stream window, readback row, modulus route, terminal seal, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from the precision request through r, s, d, q, m, e and then through the structural rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRateRealization}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed rows $p, r, d, s, q, m, e, H, C, P, N$. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, countable choice, host real equality, trichotomy, or an unrecorded completion object.*

Lean4 ✓ `BEDC.Derived.CauchyRateRealizationUp.CauchyRateRealizationTasteGate single carrier alignment`

Theorem 1673.2 (Cauchy rate realization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateRealization}^{\uparrow}$ *carrier*

$$K_{\mathrm{real}} = (p, r, d, s, q, m, e, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRateRealization}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed precision request, effective rate row, dyadic tolerance ledger, stream window, regular rational readback, Real-facing modulus route, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, package provenance, and local naming. Every accepted Real-facing read is classified only by those rows, and no obligation is stored in a quotient stream, selected limit, countable-choice selector, host real equality proof, ambient completeness theorem, or detached seal row.*

Proof. Project the accepted carrier. Its analytic coordinates are exactly p, r, d, s, q, m, e, and its structural coordinates are exactly H, C, P, N.

Read the analytic rows in dependency order. The precision request p first consumes the effective rate row r; the stream window s and dyadic ledger d are then displayed at the scheduled threshold; the regular readback row q is obtained from those finite observations; and the Real-facing modulus row m records that the same route serves the terminal seal request.

The structural rows add no analytic source. Componentwise transport in H preserves the displayed coordinates, continuation replay in C follows the same route, and P, N only source and name the packet.

Since the carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, ambient completeness, trichotomy, or a detached seal row, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.CauchyRateRealizationUp.CauchyRateRealizationNameCertObligations`

Theorem 1673.3 (Cauchy rate realization Real seal route)**.** *Every accepted* $\mathsf{CauchyRateRealization}^{\uparrow}$ *carrier routes a displayed precision request to the terminal* $\mathsf{Real}^{\uparrow}$ *seal through the finite chain*

$$p \longmapsto r \longmapsto s, d \longmapsto q \longmapsto m \longmapsto e \longmapsto (H, C, P, N).$$

Thus the seal read is the realization of explicit Cauchy-rate data through finite window, tolerance, readback, and modulus rows, not an appeal to $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, countable choice, or host real equality.*

Lean4 ✓ `BEDC.Derived.CauchyRateRealizationUp.CauchyRateRealizationRealSealRoute`

Proof. Start with the obligation surface of Definition 1673.2. It places the precision row p and effective rate row r before every downstream read.

The finite observations are then forced through the scheduled window and tolerance rows. The stream row s and dyadic ledger d are the only coordinates from which the regular rational readback q is available.

The modulus route and seal are downstream. The row m records the Real-facing modulus use of the same finite data, and e is reached only after p, r, s, d, q, m have been displayed.

Finally replay the structural rows. The route through H, C, P, N preserves and names the same packet, so no ambient completion theorem, selected host limit, quotient stream, countable-choice row, or host equality proof can substitute for the displayed chain. □

Theorem 1673.4 (Cauchy rate realization no-choice boundary)**.** *An accepted* $\mathsf{CauchyRateRealization}^{\uparrow}$ *packet cannot manufacture a selected limit, choose a representative stream, invoke countable choice, or identify host real values. All consumer-visible data are already present in the finite rows* $p, r, d, s, q, m, e, H, C, P, N$*, and every Real-facing read of the packet factors through those rows before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Lean4 ✓ `BEDC.Derived.CauchyRateRealizationUp.CauchyRateRealizationCarrier no choice boundary`

Proof. Inspect the carrier in Definition 1673.1. The only analytic rows are the precision request, rate schedule, dyadic tolerance ledger, stream window, regular readback, Real-facing modulus route, and terminal seal.

Apply the route theorem Definition 1673.3. It forces every consumer-visible seal read through those analytic rows before the structural replay, provenance, and local naming rows are read.

It remains to compare this route with the excluded principles. A selected limit, representative stream choice, countable-choice sequence, host real equality proof, or ambient completeness theorem would require an additional coordinate or an alternate continuation route. The accepted packet has neither, so those principles cannot be manufactured by the local certificate. □

Theorem 1673.5 (Cauchy rate realization public certificate)**.** *The public certificate exported by an accepted* $\mathsf{CauchyRateRealization}^{\uparrow}$ *packet is exactly the finite realization bridge from explicit rate data to the terminal Real seal:*

$$\mathsf{CauchyRate}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{RealCauchyModulus}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Every downstream Real-and-Completion consumer sees the same displayed rows $p, r, d, s, q, m, e, H, C, P, N$*, and the packet exports no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream, selected limit, countable-choice selector, or host real equality proof.*

Lean4 ✓ `BEDC.Derived.CauchyRateRealizationUp.CauchyRateRealizationPublicCertificate`

Proof. Project the accepted packet and read its provenance row. The row P names exactly the sibling certificate surfaces for the effective rate, finite stream window, dyadic tolerance ledger, regular rational readback, Real-facing modulus route, and terminal Real seal.

Use Definition 1673.2 to identify the local certificate obligations. They are exhausted by the displayed analytic and structural rows, so a public consumer cannot ask the packet for data outside $p, r, d, s, q, m, e, H, C, P, N$.

Apply Definition 1673.4. The excluded principles have no coordinate and no continuation route inside the packet. Therefore the public certificate is precisely the finite realization bridge through the listed sibling surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateRealization}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1673.1, Definition 1673.2, Definition 1673.3, Definition 1673.4, and Definition 1673.5 seed the finite realization surface from explicit $\mathsf{CauchyRate}^{\uparrow}$ data through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateRealization}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyRateRealizationUp`

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, countable choice, host real equality, Lean theorem coverage, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, seal-route factorization, no-choice boundary, public certificate export, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRateRealizationUp.

Constructive story: A $\mathsf{CauchyRateRealization}^{\uparrow}$ packet is generated from a precision request, an explicit $\mathsf{CauchyRate}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{StreamName}^{\uparrow}$ finite-window row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{RealCauchyModulus}^{\uparrow}$ route row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1674

A Concrete Naming Certificate for $\mathsf{RealMinMax}^{\uparrow}$

Following Bishop and Bridges constructive analysis, finite minimum and maximum of two displayed real names are read through located comparison and equality rows, not through trichotomy, quotient equality, host real order, or completeness. The $\mathsf{RealMinMax}^{\uparrow}$ packet sits over the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance core of chapter 291, the equality boundary of chapter 1636, the located comparison row of chapter 1381, and the algebraic order ledger of chapter 766. It supplies a finite order-operation boundary for RealAlgOrder consumers without selecting representatives or asserting ambient completeness.

Definition 1674.1 (RealMinMax carrier)**.** *A* $\mathsf{RealMinMax}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, R_X, R_Y, S_X, S_Y, D, E, C, O, L, U, H, T, P, N)$$

with the following displayed rows:

1. *X and Y are the two displayed* $\mathsf{Real}^{\uparrow}$ *seal rows whose finite minimum and maximum are being named.*
2. *R_X and R_Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed by the real seals before any order operation is consumed.*
3. *S_X and S_Y are finite* $\mathsf{StreamName}^{\uparrow}$ *windows selecting the observations used by R_X and R_Y.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-tolerance ledger, recording the dyadic radius, slack, and finite comparison budget for the windows.*
5. *E is the* $\mathsf{RealEquality}^{\uparrow}$ *boundary row, used when the displayed comparison reaches an equality or overlap boundary.*
6. *C is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *row, recording the located comparison verdict obtained from the readbacks and the dyadic tolerance ledger.*
7. *O is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *order ledger, exposing the nonnegative and apartness-facing rows that downstream algebraic-order consumers may read.*
8. *L is the minimum row. It is admitted only after $X, Y, R_X, R_Y, S_X, S_Y, D, E, C, O$ have been displayed.*

9. *U is the maximum row. It is admitted only after the same comparison, equality, and order rows have been displayed.*

10. *H records componentwise* hsame *transport for the two real seals, readbacks, finite windows, tolerance ledger, equality row, comparison row, order ledger, minimum row, maximum row, continuation row, provenance row, and local naming row.*

11. *T records displayed* Cont *replay from a finite comparison request through $S_X, S_Y, R_X, R_Y, D, E, C, O, L, U$ and back to the public min/max boundary.*

12. *P is the* Pkg *provenance over* $\mathsf{RealMinMax}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealMinMax}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite constructive min/max packet. It has no coordinate for real trichotomy, quotient real equality, host real order, selected representatives, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealMinMaxUp.RealMinMaxUp single carrier alignment`

Theorem 1674.2 (RealMinMax NameCert obligations)**.** *For every accepted* $\mathsf{RealMinMax}^{\uparrow}$ *carrier $M = (X, Y, R_X, R_Y, S_X, S_Y, D, E, C, O, L, U, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealMinMax}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{Real}^{\uparrow}$ *seals, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, two finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledger, equality boundary, comparison row, order ledger, min row, max row, transport, replay, provenance, and local naming rows;*

2. *classifier transport, requiring H and T to preserve the same two seals, readbacks, windows, tolerance row, equality row, comparison row, order row, and min/max rows componentwise;*

3. *finite comparison handoff, requiring every order-facing consumer to receive L and U only through the displayed route D, E, C, O;*

4. *min/max row stability, requiring replacement of displayed windows by* hsame*-transported windows to preserve the same public minimum and maximum rows;*

5. *non-escape, excluding trichotomy, quotient real equality, host real order, selected representatives,* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.RealMinMaxUp.RealMinMaxTasteGate single carrier alignment`

Proof. Project the accepted carrier. The analytic coordinates are the two real seals, their readbacks, their finite windows, the tolerance ledger, the equality boundary, the comparison row, the algebraic order row, and the two output rows L and U.

Carrier admission is exactly admission of those displayed coordinates together with H, T, P, N. The local certificate has no additional field from which a host real value or quotient representative could be recovered.

Classifier transport is structural. Componentwise hsame in H keeps the two input seals, readbacks, windows, comparison data, and output rows aligned, while Cont replay in T repeats the same finite comparison route for later consumers.

The handoff obligation is forced by the order of the rows. A consumer can inspect L or U only after the dyadic tolerance row D, equality row E, located comparison row C, and algebraic order ledger O have already been displayed.

Min/max stability follows because changing finite windows along the recorded hsame transport does not change the comparison route presented to D, E, C, O. The same route therefore names the same public minimum and maximum rows.

Finally, the non-escape clause is immediate from the carrier surface. None of trichotomy, quotient real equality, host real order, selected representatives, $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking is a coordinate of M. □

Theorem 1674.3 (RealMinMax finite comparison handoff)**.** *For every accepted* $\mathsf{RealMinMax}^{\uparrow}$ *carrier* $M = (X, Y, R_X, R_Y, S_X, S_Y, D, E, C, O, L, U, H, T, P, N)$*, every* $\mathsf{RealAlgOrder}^{\uparrow}$ *consumer receives the minimum row* L *and maximum row* U *only through the displayed finite route*

$$X, Y \longmapsto S_X, S_Y \longmapsto R_X, R_Y \longmapsto D, E, C, O \longmapsto L, U \longmapsto H, T, P, N.$$

Consequently the handoff supplies min/max data only through displayed finite comparison, equality, and order rows; it does not supply trichotomy, quotient real equality, host real order, selected representatives, $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge-checked standard object.*

Lean4 ✓ `BEDC.Derived.RealMinMaxUp.RealMinMaxCarrier finite comparison handoff`

Proof. Start from an accepted carrier. The only input real rows are X and Y, and their only readback route in the packet passes through the finite windows S_X, S_Y and regular rational rows R_X, R_Y.

The comparison data are then fixed by D, E, C, O. The dyadic row D supplies the finite tolerance budget, E records the equality boundary, C records the Bishop-style located comparison read, and O records the algebraic order ledger consumed by RealAlgOrder-facing requests.

The output rows L and U occur after those rows in the carrier, so they cannot be read before the comparison route is displayed. They are therefore public min/max rows over that route rather than representatives selected from a host ordered real.

The rows H, T, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for trichotomy, quotient real equality, host real order, selected representatives, completeness, or bridge checking, every $\mathsf{RealAlgOrder}^{\uparrow}$ consumer receives only the displayed finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealMinMax}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealMinMax}^{\uparrow}$ packet is generated from two displayed RealUp seals, two RegSeqRat readbacks, finite StreamName windows, a DyadicRatCore comparison-tolerance ledger, a RealEquality boundary row, a BishopRealComparison comparison row, a RealAlgOrder order ledger, min and max rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealMinMax}^{\uparrow}}$, seed)

Scope closed: Definition 1674.1, Definition 1674.2, and Definition 1674.3 seed the finite constructive min/max boundary for displayed RealUp names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealMinMax}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, host real order, selected representatives, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite comparison handoff, min/max row stability, and non-escape proofs for BEDC.Derived.RealMinMaxUp.

Chapter 1675

A Concrete Naming Certificate for $\mathsf{BrouwerBarInduction}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on spreads, bars, and bar induction, the $\mathsf{BrouwerBarInduction}^{\uparrow}$ packet records the finite barred-spread boundary needed by constructive Real and Completion work. It is positioned after the finite fan surface of chapter 400, the prefix-law surface of chapter 1532, and the finite choice-sequence surface of chapter 1482. Downstream Real and Completion consumers may cite it only through the $\mathsf{StreamName}^{\uparrow}$ window discipline, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the terminal $\mathsf{Real}^{\uparrow}$ seal; they do not receive countable choice, open-cover compactness, sequential compactness, selected infinite branches, quotient stream equality, or ambient real completeness.

Definition 1675.1 (Brouwer bar induction carrier)**.** *A* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *carrier is a finite barred-spread packet*

$$K = (T, B, M, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite prefix-tree row for the currently inspected spread. It records the displayed root, prefix closure, and finite extension rows, not a completed infinite branch.*
2. *B is the decidable bar row on prefixes of T, including the displayed Bool decisions for every prefix exposed by the current packet.*
3. *M is the monotone coverage row, asserting that whenever a displayed prefix is barred, every displayed extension remains barred through the same prefix-tree route.*
4. *I is the induction-step replay row. It records the local backwards step from barred immediate extensions to the displayed parent prefix, using only the finite children visible in T.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row by which Real and Completion consumers may request finite observations of the spread.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows when a regular-rational consumer reads the packet.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after T, B, M, I, W, R and the structural rows have been displayed.*

8. *H stores componentwise* hsame *transport for the prefix tree, bar decisions, monotone coverage, induction replay, StreamName windows, RegSeqRat readback, Real handoff, continuation, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from prefix inspection through bar decisions, monotone closure, induction-step replay, StreamName windows, RegSeqRat readback, and the Real handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerBarInduction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, B, M, I, W, R, E, H, C, P, N$. It has no coordinate for countable choice, open-cover compactness, sequential compactness, quotient equality of streams, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BrouwerBarInductionUp.BrouwerBarInductionTasteGate single carrier alignment`

Theorem 1675.2 (Brouwer bar induction prefix stability)**.** *Let $K = (T, B, M, I, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *carrier. Every displayed prefix subtree of T, with the restricted bar row, monotone coverage row, induction-step replay row, StreamName windows, RegSeqRat readback, Real handoff, and structural rows inherited from K, repacks as an accepted* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *carrier. The restricted packet has no access to tail values outside the displayed prefix subtree.*

Lean4 ✓ `BEDC.Derived.BrouwerBarInductionUp.BrouwerBarInductionPrefixStability`

Proof. Project the carrier of Definition 1675.1. A displayed prefix subtree is formed by retaining only the prefix rows already present in T together with the corresponding decisions in B.

The monotone row M restricts because every retained extension is still an extension inside the original finite tree row. The induction-step row I restricts for the same reason: the local backwards step is read only over the displayed children of the retained parent prefix.

The rows W, R, E remain downstream of the restricted prefix surface. Componentwise transport, continuation replay, provenance, and naming are supplied by H, C, P, N, so the restricted tuple is again the same finite packet type and exposes no hidden tail data. □

Theorem 1675.3 (Brouwer bar induction choice and completeness boundary)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing read of an accepted* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *carrier factors through*

$$T, B, M, I, W, R, E, H, C, P, N.$$

The read may use finite barred-prefix information and the displayed induction-step replay, but it cannot extract countable choice, open-cover compactness, sequential compactness without choice-like data, quotient stream equality, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BrouwerBarInductionUp.BrouwerBarInductionChoiceBoundary`

Proof. Start with a downstream read. The only analytic-facing rows of the carrier are the StreamName window W, the RegSeqRat readback R, and the Real handoff E.

Replay the continuation row C. The route first exposes the prefix tree T, the decidable bar row B, monotone coverage M, and the induction-step row I; only then can it open W, read R, and reach E.

The structural rows H, P, N preserve and name that same route. Since the carrier contains no selector for infinite branches, no quotient stream equality row, no open-cover row, no sequential

compactness row, and no ambient completeness row, none of those principles can be exported by the packet. □

Theorem 1675.4 (Brouwer bar induction NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *carrier* $K = (T, B, M, I, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BrouwerBarInduction}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the prefix tree, decidable bar, monotone coverage, induction-step replay, StreamName window, RegSeqRat readback, Real handoff, transport, continuation, provenance, and naming rows;*
2. *prefix stability, requiring every displayed prefix subtree to repack through Definition 1675.2;*
3. *monotone barred-spread replay, requiring bar decisions and extension closure to be read only through* T, B, M*;*
4. *induction-step exactness, requiring every backwards step to pass through the displayed local replay row* I *rather than through a host infinite branch;*
5. *analytic boundary nonescape, requiring every Real or Completion-facing read to use the route of Definition 1675.3.*

No local obligation is stored outside these five rows.

Lean4 ✓ `BEDC.Derived.BrouwerBarInductionUp.BrouwerBarInductionNameCertObligations`

Proof. Carrier admission is the tuple displayed in Definition 1675.1. The classifier can inspect only the listed coordinates, so the first obligation is exactly the projection of $T, B, M, I, W, R, E, H, C, P, N$.

Prefix stability is Definition 1675.2. It shows that a shorter displayed subtree keeps the same bar, monotone, replay, analytic handoff, transport, continuation, provenance, and naming structure.

The monotone barred-spread and induction-step clauses are read directly from B, M, I. The bar row gives only displayed Bool decisions, the monotone row preserves those decisions along displayed extensions, and the replay row supplies only the finite backwards step over visible children.

The analytic boundary is Definition 1675.3. It forces every StreamName, RegSeqRat, Real, or Completion consumer through W, R, E after the barred-spread rows have been displayed.

The packet has no further coordinate. Therefore countable choice, open-cover compactness, sequential compactness, quotient stream equality, selected infinite branches, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{BrouwerBarInduction}^{\uparrow}$ Real or Completion-facing read factors through a displayed finite prefix tree, decidable bar row, monotone coverage row, induction-step replay row, StreamName window, RegSeqRat readback, Real handoff, transport, continuation, provenance, and local naming.

Independence witness: fantheorem, spread-law, choice-sequence, streamname, regseqrat, real: $\mathsf{BrouwerBarInduction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite barred-spread prefix tree, decidable bar, monotone coverage, induction-step replay, analytic window handoff, transport, continuation, provenance, and local naming while refusing branch selection and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerBarInduction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BrouwerBarInduction}^{\uparrow}$ packet is generated from a finite prefix tree, a decidable bar row, monotone coverage, induction-step replay, StreamName windows, RegSeqRat readback, a Real handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BrouwerBarInduction}^{\uparrow}}$, seed)

Scope closed: Definition 1675.1, Definition 1675.2, Definition 1675.3, and Definition 1675.4 seed the finite barred-spread boundary for Real and Completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{BrouwerBarInduction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, open-cover compactness, sequential compactness, quotient stream equality, selected infinite branches, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix stability, monotone barred-spread replay, induction-step exactness, analytic boundary nonescape, and TasteGate field faithfulness for BEDC.Derived.BrouwerBarInductionUp.

Chapter 1676

A Concrete Naming Certificate for $\mathsf{BrouwerOperation}^{\uparrow}$

Following Troelstra–van Dalen on neighbourhood functions, the $\mathsf{BrouwerOperation}^{\uparrow}$ packet records a finite prefix-tree operation surface over countable-name data. It sits between the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126, the $\mathsf{BaireSpace}^{\uparrow}$ finite cylinder surface of chapter 334, the prefix-law rows of chapter 1532, the barred-prefix boundary of chapter 1675, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite neighbourhood-function certificate for downstream Real and Completion consumers; it is not a host function space, countable choice, quotient stream equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1676.1 (BrouwerOperation finite neighbourhood carrier)**.** *A* $\mathsf{BrouwerOperation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite prefix-tree row over countable-name observations. It records the retained prefixes, root row, immediate-extension rows, and the finite cylinder boundary shared with* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{SpreadLaw}^{\uparrow}$*.*
2. *O is the output-row ledger. For each accepted prefix at which the neighbourhood function has stabilized, O records the displayed output cell and the finite justification prefix that produced it.*
3. *M is the monotone extension ledger. It requires every displayed extension of a stabilized prefix to retain the same output row through the same prefix-tree route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row by which a consumer reads only the finite prefix needed by the current operation cell.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed after the relevant window has been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the prefix tree, output ledger, monotone extension ledger, StreamName window, and RegSeqRat readback have been displayed.*
7. *H stores componentwise* hsame *transport for the prefix tree, output rows, monotone ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from finite prefix inspection through output stabilization, monotone extension, StreamName window, RegSeqRat readback, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerOperation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerOperation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, O, M, W, R, E, H, C, P, N$. It has no coordinate for host function extensionality, countable choice, quotient stream equality, selected infinite branches, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BrouwerOperationUp.BrouwerOperationTasteGate single carrier alignment`

Theorem 1676.2 (BrouwerOperation NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerOperation}^{\uparrow}$ *carrier $K = (T, O, M, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BrouwerOperation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed prefix tree, output-row ledger, monotone extension ledger, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *prefix-tree stability, requiring accepted restrictions to retain only rows already displayed in T and to compare them through the* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{SpreadLaw}^{\uparrow}$ *cylinder classifiers;*
3. *output exactness, requiring each stabilized output in O to name its finite justification prefix and no hidden tail of the input name;*
4. *monotone extension, requiring M to preserve the output row along every displayed extension of a stabilized prefix;*
5. *analytic readback, requiring W, R, E to be consumed in that order before a Real or Completion-facing route may read the operation value;*
6. *provenance and local naming, requiring P and N to package exactly the displayed rows and no additional function-space datum.*

No BrouwerOperation obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 1676.1. The visible packet consists exactly of the prefix tree T, output ledger O, monotone extension ledger M, analytic rows W, R, E, and structural rows H, C, P, N.

Prefix-tree stability is inherited from the retained finite rows of T. A restriction keeps only prefixes already displayed, so its comparisons still pass through the same BaireSpace cylinder and SpreadLaw prefix surfaces.

Output exactness and monotone extension are the local contents of O and M. The output row names the finite justification prefix, while the monotone ledger says that displayed extensions of that prefix preserve the same output cell.

The analytic readback row is forced by W, R, E. The StreamName window must be opened before RegSeqRat readback, and the Real seal is terminal after that readback.

Finally, H, C, P, N transport, replay, package, and name the same route without adding a host function, stream quotient, choice selector, branch selector, selected limit, or Real-completeness coordinate. Hence the six listed rows exhaust the local NameCert surface. □

Theorem 1676.3 (BrouwerOperation prefix-continuity handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing consumer of an accepted* $\mathsf{BrouwerOperation}^{\uparrow}$ *carrier receives only the finite prefix-continuity route*

$$T \longmapsto O \longmapsto M \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer may cite a stabilized finite prefix and its monotone extension ledger, but it cannot bypass the displayed route by importing a host function space, countable choice, quotient stream equality, a selected infinite branch, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and a downstream consumer. The only input-inspection coordinate is the prefix tree T, and the only operation-value coordinate is the output ledger O.

The monotone extension ledger M is the route from a stabilized prefix to its displayed extensions. It prevents the consumer from replacing the finite justification prefix by an unstated tail property of the input name.

The analytic part of the handoff is then forced through W, R, E. A StreamName window opens the finite observation, RegSeqRat reads that window as regular rational data, and the Real seal is reached only after those rows have been consumed.

The structural rows H, C, P, N preserve and name this same route. Since the carrier has no coordinate for a host function space, countable choice, quotient stream equality, selected branch, selected limit, or ambient completeness theorem, none of those objects can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{BrouwerOperation}^{\uparrow}$ Real or Completion-facing read factors through a finite prefix tree, output-row ledger, monotone extension ledger, StreamName window, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, bairespace, spread-law, brouwer-bar-induction, regseqrat, real: $\mathsf{BrouwerOperation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite prefix-tree operation data, stabilized output rows, monotone extension, analytic readback, transport, replay, provenance, and local naming while refusing host functions, countable choice, quotient stream equality, selected branches, selected limits, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerOperation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerOperation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1676.1, Definition 1676.2, and Definition 1676.3 seed the finite neighbourhood-function route from prefix-tree operation data through StreamName windows, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerOperation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, countable choice, quotient stream equality, selected infinite branches, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-tree stability, output exactness, monotone extension, analytic readback, provenance, and local naming for BEDC.Derived.BrouwerOperationUp.

Constructive story: A $\mathsf{BrouwerOperation}^{\uparrow}$ packet is generated from finite prefix-tree neighbourhood data, stabilized output rows, a monotone extension ledger, a StreamName window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1677

A Concrete Naming Certificate for $\mathsf{BrouwerSpread}^{\uparrow}$

Following Brouwer and Bishop–Bridges on spreads and finite choice-sequence observation, the $\mathsf{BrouwerSpread}^{\uparrow}$ packet records the finite prefix carrier that mediates between displayed choice-sequence reads and later bar, fan, Real, and Completion consumers. It sits after the finite answer discipline of chapter 1482, the binary tree surface of chapter 1475, the prefix-cylinder surface of chapter 334, the window schedule of chapter 126, and the admissibility law of chapter 1532. The packet names finite admitted prefixes; it is not a completed infinite branch, countable choice, quotient stream equality, fan compactness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1677.1 (Brouwer spread carrier)**.** *A* $\mathsf{BrouwerSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, S, W, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the decidable prefix-tree row. It records the displayed root, prefix relation, prefix-closure checks, and spread-law decisions inherited from the finite admissibility surface of* $\mathsf{SpreadLaw}^{\uparrow}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule. It records the finite order in which admitted prefixes may be requested, replayed, and compared by downstream name readers.*
3. *W is the finite displayed-prefix row. Every entry of W is a prefix already listed in T and already scheduled by S.*
4. *A is the admissible answer row. It contains only displayed* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Bool}^{\uparrow}$ *answers whose finite prefixes pass the decision row of T.*
5. *H stores componentwise* hsame *transport for the tree, schedule, displayed prefixes, admissible answers, continuation replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *prefix replay from a request for a finite choice-sequence read to the corresponding T, S, W, A rows.*
7. *P is the* Pkg *provenance over* $\mathsf{BrouwerSpread}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{SpreadLaw}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerSpread}^{\uparrow}}$ *row.*

The carrier has no coordinate for a total host function, a completed branch through T, a quotient stream, a countable-choice selector, compactness, or a $\mathsf{Real}^{\uparrow}$ *completeness seal.*

Lean4 ✓ `BEDC.Derived.BrouwerSpreadUp.BrouwerSpreadTasteGate single carrier alignment`

Definition 1677.2 (Brouwer spread classifier)**.** *Two accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *carriers*

$$B = (T, S, W, A, H, C, P, N)$$

and

$$B' = (T', S', W', A', H', C', P', N')$$

are classified together exactly when their common admitted prefixes agree through the displayed rows T, S, W, A *and* T', S', W', A'*, and when every branching decision visible in one carrier is admissible through the corresponding spread-law row of the other carrier wherever both packets display the prefix. The classifier compares finite common-prefix admission, branching admissibility, schedule replay, transport, continuation, provenance, and local naming. It never compares host functions, completed infinite streams, quotient representatives, or extensional equality of unseen tails.*

Theorem 1677.3 (Brouwer spread prefix closure)**.** *Let*

$$B = (T, S, W, A, H, C, P, N)$$

be an accepted $\mathsf{BrouwerSpread}^{\uparrow}$ *carrier. Every admitted prefix subwindow of* W*, with the restricted prefix tree, restricted StreamName schedule, restricted admissible* $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Bool}^{\uparrow}$ *answer rows, inherited componentwise transport, inherited continuation replay, inherited provenance, and inherited local naming, repacks as an accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *carrier.*

Proof. Project the carrier of Definition 1677.1. An admitted prefix subwindow keeps only entries of W already licensed by T and already scheduled by S.

The restricted tree remains prefix closed because T stores prefix-closure checks for every displayed prefix. The restricted answer row remains admissible because A contains only displayed $\mathsf{Nat}^{\uparrow}$ and $\mathsf{Bool}^{\uparrow}$ answers attached to prefixes accepted by T.

The structural rows restrict componentwise. The transport row H compares the retained coordinates, the continuation row C replays the same prefix requests over the shorter window, and P, N name the same local package. The resulting tuple has the same carrier shape and no access to any tail outside the displayed subwindow. □

Theorem 1677.4 (Brouwer spread choice-sequence factorization)**.** *Every public* $\mathsf{ChoiceSequence}^{\uparrow}$ *read of an accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *carrier factors through the admitted finite prefix rows*

$$T, S, W, A, H, C, P, N.$$

The read may inspect only the finite $\mathsf{Nat}^{\uparrow}$ *and* $\mathsf{Bool}^{\uparrow}$ *answers admitted by the prefix tree and scheduled by the StreamName row; it cannot extract a completed infinite choice sequence, a host function, quotient stream equality, countable choice, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a public $\mathsf{ChoiceSequence}^{\uparrow}$ read. The carrier exposes choice-sequence data only through the displayed-prefix row W and the admissible answer row A.

Replay the continuation row C. The route first checks the decidable prefix tree T, then opens the StreamName schedule S, and only then reads the finite answers recorded in W, A.

The structural rows H, P, N preserve and name that route. Since the packet has no coordinate for a total sequence, quotient stream, choice selector, compactness principle, or real-completion seal, every public read factors through the finite prefix rows and cannot export the excluded data. □

Theorem 1677.5 (Brouwer spread NameCert obligation surface)**.** *For every accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *carrier* $B = (T, S, W, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BrouwerSpread}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the decidable prefix tree, StreamName schedule, displayed-prefix row, admissible answer row, transport, continuation, provenance, and local naming;*
2. *classifier agreement on common admitted prefixes and branching admissibility as in Definition 1677.2;*
3. *prefix closure through Definition 1677.3;*
4. *choice-sequence factorization through Definition 1677.4;*
5. *componentwise* hsame *transport for every displayed row;*
6. *displayed* Cont *replay for every prefix request;*
7. *exact* Pkg *provenance and local* NameCert *naming;*
8. *non-escape, requiring every downstream read to stay inside* T, S, W, A, H, C, P, N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1677.1. The visible packet contains exactly T, S, W, A, H, C, P, N.

Classifier agreement is Definition 1677.2. It compares common admitted prefixes and branching admissibility without invoking equality of host functions or unseen stream tails.

Prefix closure and choice-sequence factorization are supplied by Definition 1677.3 and Definition 1677.4. These two clauses force every retained subwindow and every public choice-sequence read through the same finite prefix rows.

The remaining clauses are the structural projections. The row H gives transport, C gives replay, and P, N give provenance and local naming. Because the carrier inventory has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{BrouwerSpread}^{\uparrow}$ public ChoiceSequence, StreamName, Real, or Completion-facing read factors through a decidable prefix tree, StreamName schedule, finite displayed-prefix row, admissible Nat/Bool answer row, componentwise transport, continuation replay, provenance, and local NameCert row.

Independence witness: choice-sequence, cantorspace, bairespace, streamname, spread-law: $\mathsf{BrouwerSpread}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries decidable prefix admission, StreamName scheduling, displayed prefix windows, admissible Nat/Bool answer rows, transport, continuation replay, provenance, and local naming while refusing completed branches, countable choice, quotient streams, compactness, and ambient Real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerSpread}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BrouwerSpread}^{\uparrow}$ packet is generated from a decidable prefix tree, a StreamName schedule, finite displayed prefixes, admissible Nat/Bool answer rows, componentwise hsame transport, displayed Cont prefix replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BrouwerSpread}^{\uparrow}}$, seed)

Scope closed: Definition 1677.1, Definition 1677.2, Definition 1677.3, Definition 1677.4, and Definition 1677.5 seed the finite-prefix spread carrier for L10 Real and Completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BrouwerSpread}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed infinite branches, countable choice, quotient stream equality, fan compactness, open-cover compactness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-prefix classification, prefix closure, choice-sequence factorization, transport, continuation, provenance, and non-escape for BEDC.Derived.BrouwerSpreadUp.

Chapter 1678

A Concrete Naming Certificate for $\mathsf{Limit}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's finite-neighborhood reading of convergence, the $\mathsf{Limit}^{\uparrow}$ packet names the generic finite limit surface used by the Real and Completion chapters. It is not a host limit operator and not a quotient of convergent streams. It is the visible route through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and a downstream $\mathsf{Real}^{\uparrow}$ seal row, with transport, continuation replay, provenance, and local naming retained as BEDC rows.

The sibling dependency route is deliberately shared with the specialized limit chapters. The $\mathsf{RealLimit}^{\uparrow}$ packet of chapter 347 and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ boundary of chapter 386 consume the same finite faces supplied by chapter 126, chapter 301, chapter 291, and chapter 38. The generic $\mathsf{Limit}^{\uparrow}$ carrier records that common surface before any specialized real-limit or completion-facing packet chooses a narrower role.

Definition 1678.1 (Limit carrier)**.** *A* $\mathsf{Limit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, A, T, C, H, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that supplies the finite observation schedule for the limit request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radius, slack, and comparison budget for each finite observation.*
4. *A is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, read only after S, R, D have been replayed.*
5. *T is the limit transport row binding the same S, R, D, A surface to any specialized* $\mathsf{RealLimit}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer.*
6. *C records displayed* Cont *replay from a finite limit request through the window row, rational readback, tolerance ledger, seal row, and transport row.*
7. *H stores componentwise* hsame *transport for the window, readback, tolerance ledger, seal, limit transport, continuation, provenance, and local naming rows.*

8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLimit}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows, their displayed transports, their continuation replay, package provenance, and local naming provenance. The carrier has no coordinate for quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.LimitUp.LimitTasteGate single carrier alignment`

Theorem 1678.2 (Limit NameCert obligations)**.** *For every accepted* $\mathsf{Limit}^{\uparrow}$ *carrier* $L = (S, R, D, A, T, C, H, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed window, readback, tolerance ledger, seal, transport, continuation, provenance, and local naming rows;*
2. *finite-window exactness, requiring every limit-facing read to inspect S before it can reach the regular rational readback row R;*
3. *tolerance-ledger exactness, requiring every readback comparison to pass through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D before the* $\mathsf{Real}^{\uparrow}$ *seal row A is exposed;*
4. *specialized-consumer handoff, requiring every* $\mathsf{RealLimit}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer to reuse the same S, R, D, A route through T, C, H, P, N;*
5. *non-escape from quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, and bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 1678.1. The projection exposes exactly the finite window row, regular rational readback, dyadic tolerance ledger, Real seal, limit transport, continuation replay, componentwise transport, package provenance, and local naming row.

Finite-window exactness follows from the route order stored in C. A limit-facing request enters through S before it can obtain R, and the comparison budget is then read from D before the seal row A can be consumed.

The specialized handoff is supplied by T, H, P, N. These rows retain the same displayed route for $\mathsf{RealLimit}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers, so the packet has no remaining coordinate from which quotient equality, selected representatives, host equality, trichotomy, choice, completeness, Lean verification, or bridge checking could be exported. □

Lean4 ✓ `BEDC.Derived.LimitUp.LimitUpTasteGate single carrier alignment`

Theorem 1678.3 (Limit Real and Completion handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

as displayed by the packet rows S, R, D, A, then through the transport, continuation, provenance, and local naming rows T, C, H, P, N. Consequently the generic limit surface can be cited by $\mathsf{RealLimit}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *chapters without claiming quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with a Real-facing or Completion-facing consumer. Carrier admission gives access to the seal row A only as a row downstream of the finite window, readback, and tolerance rows.

Replay the continuation row C. The consumer first reads S, obtains the regular rational observations through R, consults the dyadic tolerance ledger D, and only then reaches the $\mathsf{Real}^{\uparrow}$ seal row A.

Finally apply the handoff row T together with the structural rows H, P, N. They repack the same finite route for $\mathsf{RealLimit}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers, while adding no quotient, representative selector, host equality, trichotomy, choice, completeness, Lean target, or bridge row. □

Lean4 ✓ `BEDC.Derived.LimitUp.LimitRealCompletionHandoff`

Theorem 1678.4 (Limit RegSeqRat Bishop handoff)**.** *Any Bishop-style* $\mathsf{Limit}^{\uparrow}$ *read in the current L10 route must pass through finite* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a downstream* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ *naming. For an accepted carrier* $L = (S, R, D, A, T, C, H, P, N)$*, the only exported route is*

$$S \longmapsto R \longmapsto D \longmapsto A \longmapsto T, C, H, P, N.$$

The packet cannot export selected representatives, quotient stream equality, trichotomy, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Proof. Unfold the finite carrier of Definition 1678.1. The window row S, regular rational readback row R, dyadic tolerance ledger D, Real seal A, and specialized handoff row T are the only content rows available to a Bishop-style limit read.

Apply the NameCert obligations of Definition 1678.2. They force finite-window exactness before R, tolerance-ledger exactness before A, and specialized-consumer handoff through T, C, H, P, N.

The Real and Completion handoff of Definition 1678.3 identifies the same route as the shared L10 naming spine for downstream consumers. Since the carrier contains no selected representative, quotient stream equality, trichotomy, countable-choice, host-equality, completeness, Lean-verification, or bridge-checking coordinate, no such object can be exported by the handoff. □

Lean4 ✓ `BEDC.Derived.LimitUp.LimitRegSeqRatBishopHandoff`

Definition 1678.5 (Limit obligation surface)**.** *For an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier* $L = (S, R, D, A, T, C, H, P, N)$*, the* obligation surface *is the finite projection ledger consisting of:*

1. *the* $\mathsf{StreamName}^{\uparrow}$ *window exactness row* S *and its replay into the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *binding every displayed comparison before the downstream* $\mathsf{Real}^{\uparrow}$ *seal row* A *can be consumed;*
3. *the consumer handoff row* T *to* $\mathsf{RealLimit}^{\uparrow}$ *and* $\mathsf{CauchyLimitSeal}^{\uparrow}$*;*
4. *the structural replay rows* C, H, P, N *for* Cont*, componentwise* hsame*,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ *naming.*

This surface records only finite projection and replay obligations over the carrier rows. It is not quotient stream equality, a selected representative, a host limit operator, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1678.6 (Limit finite-window exactness)**.** *Let $L = (S, R, D, A, T, C, H, P, N)$ be an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier. Every finite limit read that reaches the regular rational row R factors through the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window row S, and the factorization is replayed by the* Cont *row C and transported componentwise by H. Thus the window obligation of Definition 1678.5 is a finite projection route through chapter 126 and chapter 301.*

Lean4 ✓ `BEDC.Derived.LimitUp.LimitFiniteWindowExactness`

Proof. Project the carrier L and isolate the rows S, R, C, H. The carrier definition exposes R only as the readback row downstream of the displayed window row S.

Replay the request through C. The replay ledger opens the window schedule first, then reads the regular rational observations recorded by R. Componentwise transport by H keeps the same S and R rows under hsame.

No further row remains from which a host stream representative, quotient equality, or ambient limit operator could be read. The exactness statement is therefore the finite projection route from $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 1678.7 (Limit tolerance-ledger exactness)**.** *Let $L = (S, R, D, A, T, C, H, P, N)$ be an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier. Every readback comparison that reaches the* $\mathsf{Real}^{\uparrow}$ *seal row A passes through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, with replay and transport recorded by C, H, P, N. Thus the tolerance obligation of Definition 1678.5 is a finite replay route through chapter 291 and chapter 38.*

Proof. Start from a comparison exported by the accepted carrier. The rows R, D, A are ordered as readback, tolerance ledger, and downstream Real seal, so a comparison cannot inspect A before the dyadic budget row D is displayed.

Use C to replay the comparison route and H to transport the same row package. The provenance row P records that the route is sourced in $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, while N names the local packet.

The carrier has no coordinate for trichotomy, countable choice, host real equality, or ambient completeness. The tolerance obligation is exactly the finite ledger route from D to A. □

Lean4 ✓ `BEDC.Derived.LimitUp.LimitToleranceLedgerExactness`

Theorem 1678.8 (Limit consumer handoff exactness)**.** *Let $L = (S, R, D, A, T, C, H, P, N)$ be an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier. Every* $\mathsf{RealLimit}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *consumer of L receives only the finite route*

$$S \longmapsto R \longmapsto D \longmapsto A \longmapsto T, C, H, P, N.$$

Consequently the consumer obligation of Definition 1678.5 is a finite handoff through chapter 347 and chapter 386, not a claim of quotient stream equality, selected representatives, host real equality, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.LimitUp.LimitConsumerHandoffExactness`

Proof. Read the consumer through the transport row T. That row binds $\mathsf{RealLimit}^{\uparrow}$ and $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers to the same upstream rows S, R, D, A displayed by the generic $\mathsf{Limit}^{\uparrow}$ packet.

Replay the route using C and transport it using H. The provenance and local naming rows P, N record the same dependency spine through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLimit}^{\uparrow}$, and $\mathsf{CauchyLimitSeal}^{\uparrow}$.

Because the carrier is exhausted by these rows, the consumer cannot acquire an external stream quotient, selected representative, host real equality proof, completeness principle, Lean target, or bridge row. The handoff exactness is therefore a finite replay obligation. □

Theorem 1678.9 (Limit scoped kernel route)**.** *Let $L = (S, R, D, A, T, C, H, P, N)$ be an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier. Every scoped read of L is bound to the BEDC primitive route*

$$\mathsf{StreamName}^{\uparrow}(S) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(R) \longmapsto \mathsf{DyadicRatCore}^{\uparrow}(D) \longmapsto \mathsf{Real}^{\uparrow}(A) \longmapsto \mathsf{hsame}(H), \mathsf{Cont}(C), \mathsf{Pkg}(P), \mathsf{Name}$$

The transport row T may hand this same route to $\mathsf{RealLimit}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, but it cannot add a quotient stream equality, selected representative, host real equality, trichotomy, countable choice, ambient completeness, or bridge-checking coordinate.*

Lean4 ✓ `BEDC.Derived.LimitUp.LimitScopedKernelRoute`

Proof. Project the carrier and read the primitive rows in order. The only nonstructural rows before the handoff are the finite StreamName window S, the RegSeqRat readback R, the DyadicRatCore tolerance ledger D, and the Real seal A.

Apply finite-window exactness and tolerance-ledger exactness. They force any scoped request to inspect S before R and D before A, so the read is already confined to the displayed BEDC primitives.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. The handoff row T only repacks it for the two specialized consumers, so no excluded coordinate remains available to a scoped reader. □

Theorem 1678.10 (Limit RegSeqRat-to-Real scope)**.** *For an accepted* $\mathsf{Limit}^{\uparrow}$ *carrier $L = (S, R, D, A, T, C, H, P, N$ every finite RegSeqRat readback consumed as a Real-facing limit observation passes through the displayed tolerance route*

$$S \longmapsto R \longmapsto D \longmapsto A$$

and keeps H, C, P, N attached to that same route. Thus the RegSeqRat-to-Real scope is finite readback plus tolerance-ledger replay, not a host equality of streams and not ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LimitUp.LimitRegseqratRealScope`

Proof. Start with a Real-facing observation exported by L. The carrier definition makes A downstream of the regular rational row R and the dyadic tolerance ledger D, with S supplying the finite observation windows.

Use Definition 1678.7. It prevents a comparison from reaching A before the same D ledger is displayed, and it keeps the replay and transport rows attached.

The remaining rows P, N record provenance and local naming only. Hence the scope is exactly the finite route from StreamName windows through RegSeqRat readback and DyadicRatCore tolerance to the Real seal. □

Theorem 1678.11 (Limit completion-scope refusal)**.** *No accepted* $\mathsf{Limit}^{\uparrow}$ *carrier exports quotient stream equality, a selected representative, host real equality, trichotomy, countable choice, or ambient completeness to a completion-facing consumer. Any such export would require a coordinate outside the scoped primitive rows $S, R, D, A, T, C, H, P, N$ and outside the local* $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ *surface.*

Lean4 ✓ `BEDC.Derived.LimitUp.LimitCarrier completion scope refusal`

Proof. Read the scoped kernel route of Definition 1678.9. It exhausts the carrier by StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a Real seal, the handoff row, transport, replay, provenance, and local naming.

Each refused object needs an additional coordinate. Quotient stream equality and selected representatives require a stream quotient or selector row; host real equality and trichotomy require external Real comparison data; countable choice and ambient completeness require global principles not stored in the finite carrier.

Because none of those coordinates occurs in the accepted packet, every completion-facing read remains inside the displayed scoped route. □

Theorem 1678.12 (Limit standard bridge envelope)**.** *The public finite route of an accepted* $\mathsf{Limit}^{\uparrow}$ *packet can be read by a standard bridge only as the same displayed envelope through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The bridge-facing read remains valid only while quotient streams, selected representatives, host real equality, countable choice, and ambient completeness stay excluded from the packet.*

Proof. Apply the RegSeqRat Bishop handoff, the scoped kernel route, and the completion-scope refusal theorems, Definition 1678.4, Definition 1678.9, and Definition 1678.11. Together they expose the same finite public route and rule out every excluded host coordinate.

A standard bridge may therefore read the route only as this finite envelope. Adding quotient streams, representatives, host equality, countable choice, or ambient completeness would require rows absent from the accepted carrier and from the local $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1678.4, Definition 1678.3, Definition 1678.8, and Definition 1678.12 supply the bridged consumer envelope for the finite Limit route through StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, Pkg, and NameCert primitives.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Limit}^{\uparrow}}$, axiomClean)

Lean target: `BEDC.Derived.LimitUp.LimitRegSeqRatBishopHandoff`

Bridge status: paperBridge

Not claimed: This chapter does not claim quotient stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: No further paper-side theory closure remains; Lean standard-bridge checking is a separate formal-axis obligation over the public Limit handoff.

Constructive story: A $\mathsf{Limit}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealUp seal row, a limit transport row for RealLimitUp and CauchyLimitSealUp consumers, continuation replay, componentwise transport, package provenance, and local naming.

Chapter 1679

A Concrete Naming Certificate for $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$ packet names the finite regular-Cauchy route for the affine expression $qx+(1-q)y$ before any completed $\mathsf{Real}^{\uparrow}$ consumer is allowed to read the result. It sits over the two-source sum certificate of chapter 417, the rational scale certificate of chapter 465, the midpoint specialization of chapter 1670, the dyadic arithmetic core of chapter 291, the stream-window discipline of chapter 126, the regular rational sequence boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet gives later completion and real-algebra consumers one finite algebraic surface instead of forcing each consumer to rethread scale, complementary scale, sum, regular readback, and seal ledgers separately.

Definition 1679.1 (Regular Cauchy affine combination carrier)**.** *A* $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aff}} = (Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *Q is the coefficient row, recording a rational coefficient q together with the dyadic window in which the complementary coefficient $1-q$ is computed.*
2. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows.*
3. *W_X and W_Y are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules for the two source names, read at the same precision request and observation index.*
4. *D_Q and $D_{\bar{Q}}$ are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for q and $1-q$ at the common exponent used by the packet.*
5. *S_X is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row for the scaled source qx, reading Q, X, W_X, D_Q and its scale-error ledger.*
6. *S_Y is the* $\mathsf{RegularCauchyScale}^{\uparrow}$ *row for the scaled source $(1-q)y$, reading the complementary coefficient row, $Y, W_Y, D_{\bar{Q}}$, and its scale-error ledger.*
7. *Σ is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row that adds the two scaled regular names.*
8. *E is the finite error-budget ledger combining the two scale ledgers, the sum tolerance, the complementary-coefficient tolerance, and the final output tolerance assigned to the affine readback.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose scheduled point row is supplied by Σ with regularity evidence supplied by E.*

10. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the coefficient, two sources, two synchronized schedules, two dyadic coefficient endpoints, two scale rows, sum row, finite ledger, and regular readback have been displayed.*

11. *H stores componentwise* hsame *transports for the coefficient, source rows, schedules, dyadic endpoint rows, scale rows, sum row, ledger, readback, Real seal, replay row, provenance row, and local naming row.*

12. *C stores displayed* Cont *replay routes for coefficient exposure, source-window exposure, the two scale reads, the sum read, the ledger read, the regular readback, and the Real seal handoff.*

13. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{RegularCauchyMidpoint}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAffineCombination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R,$ The carrier has no coordinate for quotient stream equality, host real affine arithmetic, a selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completed-real affine operation.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAffineCombinationUp.RegularCauchyAffineCombinationTasteGate`
`single carrier alignment`

Theorem 1679.2 (Regular Cauchy affine combination obligation surface)**.** *For every accepted* $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$ *carrier*

$$A_{\text{aff}} = (Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyAffineCombination}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the coefficient row, two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *schedules, dyadic endpoint rows for q and $1-q$, two* $\mathsf{RegularCauchyScale}^{\uparrow}$ *rows, one* $\mathsf{RegularCauchySum}^{\uparrow}$ *row, the finite ledger, readback, Real seal, and structural rows;*

2. *coefficient complement exposure, requiring D_Q and $D_{\bar{Q}}$ to be read from the same finite dyadic coefficient window before either scale row is consumed;*

3. *synchronized-window exposure, requiring W_X and W_Y to be read at a common requested precision before the two scale rows are formed;*

4. *scale exactness for S_X and S_Y, requiring the displayed rows for qx and $(1-q)y$ to consume their coefficient endpoint, source window, scale endpoint, and scale-error ledger;*

5. *sum exactness for Σ, requiring the two scaled regular names to be added through the* $\mathsf{RegularCauchySum}^{\uparrow}$ *surface before the affine readback row is exposed;*

6. *ledger accounting, requiring E to combine the complementary-coefficient budget, the two scale-error budgets, the sum tolerance, and the final output tolerance before R or Z is read;*

7. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the displayed finite affine packet;*

8. *refusal of quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No affine-combination obligation is stored outside these finite rows.

Proof. Project the accepted carrier. The coefficient row, two source rows, synchronized schedules, two dyadic coefficient endpoints, two scale rows, the sum row, the ledger, readback, and Real seal are explicit coordinates of the packet.

Read the coefficient and source windows first. The complementary coefficient $1-q$ is displayed in the same dyadic endpoint discipline as q, while W_X and W_Y expose the two source observations at a common finite precision.

Apply the sibling scale and sum surfaces. The rows S_X and S_Y name the two scaled regular Cauchy names, and Σ names their pointwise sum before any regular readback or Real seal row is consumed.

Finally read the ledger and structural rows. The finite row E accounts for coefficient complement, scale, sum, and output tolerance budgets, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient equality, host affine arithmetic, selected limits, trichotomy, ambient completeness, Lean verification, or a bridge, none can become a local obligation. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAffineCombinationUp.RegularCauchyAffineCombinationObligationSurface`

Theorem 1679.3 (Regular Cauchy affine combination classifier stability)**.** *Let two accepted* RegularCauchyAffineCo *carriers have coefficient rows, source rows, stream schedules, dyadic endpoint rows, scale rows, sum rows, ledgers, readbacks, Real seals, package provenance, and local naming rows matched componentwise by their displayed* hsame *classifiers. Then every consumer route supplied by the obligation surface of Definition 1679.2 transports to the corresponding route of the second packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAffineCombinationUp.RegularCauchyAffineCombinationCarrierstability`

Proof. Start with the componentwise classifier data in H. It transports the coefficient row, the complementary coefficient endpoint, the two regular source rows, and the synchronized schedules without changing the finite precision route.

Transport the scale and sum rows next. The classified scale rows still read qx and $(1-q)y$ through the corresponding coefficient endpoints and source windows, and the classified sum row still combines those two scaled names through the displayed $\mathsf{RegularCauchySum}^{\uparrow}$ surface.

The ledger route is preserved by the same componentwise transport. The row E remains the only place where complement, scale, sum, and output tolerances are combined before R and Z are exposed.

Replay through C and repack through P and N. Every consumer route in the obligation surface is one of these displayed projections, so classifier transport cannot add quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data. □

Theorem 1679.4 (Regular Cauchy affine combination ledger nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$ *packet reads the affine result only after the finite ledger E has combined the coefficient-complement row, the two scale-error rows, the sum-tolerance row, and the final output tolerance. The packet has no alternate ledger for quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAffineCombinationUp.RegularCauchyAffineCombinationLedger`

Proof. Read a consumer request through the displayed affine route. The request first reaches the coefficient row, the two source rows, the synchronized stream windows, and the dyadic endpoint rows for q and $1-q$.

The two scale rows are then consumed. Their finite scale ledgers are the only sources for the tolerances attached to qx and $(1-q)y$.

The sum row Σ and ledger E finish the regular handoff. The row E combines the two scale ledgers with the sum tolerance and final output tolerance before the readback row R or Real seal row Z can be read.

Thus every consumer-facing affine route passes through the finite ledger. Since the carrier inventory contains no second ledger or host-level affine object, no consumer can escape to quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Theorem 1679.5 (Regular Cauchy affine combination convex route)**.** *Assume the coefficient endpoint rows of an accepted* $\mathsf{RegularCauchyAffineCombination}^{\uparrow}$ *carrier display the nonnegative coefficient row for q and the complementary nonnegative coefficient row for $1-q$ inside the same dyadic window. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the affine packet factors through the synchronized route*

$$Q, X, Y, W_X, W_Y, D_Q, D_{\bar{Q}}, S_X, S_Y, \Sigma, E, R, Z.$$

This route is a finite convex-combination certificate. It exposes neither quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, nor bridge data.*

Proof. Start with the obligation surface of Definition 1679.2. It places the coefficient row, the two source rows, the synchronized StreamName windows, the dyadic endpoint rows for q and $1-q$, the two scale rows, the sum row, the ledger, readback, and Real seal in the displayed carrier order.

Read the coefficient-complement data next. The nonnegative row for q and the nonnegative complementary row for $1-q$ are both exposed through D_Q and $D_{\bar{Q}}$ inside the same dyadic window, so no consumer can split the affine read into two unrelated coefficient schedules.

Apply classifier stability from Definition 1679.3. The two source windows W_X, W_Y, the scale rows S_X, S_Y, and the sum row Σ transport componentwise along the same synchronized route.

Finally apply the ledger theorem Definition 1679.4. The row E combines the coefficient-complement budget, the two scale-error budgets, the sum tolerance, and the final output tolerance before R or Z is visible, so the consumer receives exactly the finite convex-combination certificate and no excluded quotient, host arithmetic, limit, trichotomy, completeness, or bridge data. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAffineCombinationUp.RegularCauchyAffineCombinationConvexRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAffineCombination}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAffineCombination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1679.1, Definition 1679.2, Definition 1679.3, and Definition 1679.4 seed the finite affine-combination surface over coefficient, two $\mathsf{RegSeqRat}^{\uparrow}$ sources, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, dyadic coefficient endpoints, $\mathsf{RegularCauchyScale}^{\uparrow}$ rows, a $\mathsf{RegularCauchySum}^{\uparrow}$ row, finite ledger data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAffineCombination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real affine arithmetic, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient complement exposure, synchronized-window exposure, scale exactness, sum exactness, ledger accounting, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyAffineCombinationUp.

Constructive story: A RegularCauchyAffineCombination$^{\uparrow}$ packet is generated from a coefficient row, two RegSeqRat$^{\uparrow}$ sources, synchronized StreamName$^{\uparrow}$ windows, dyadic endpoint rows for q and $1 - q$, two RegularCauchyScale$^{\uparrow}$ rows, a RegularCauchySum$^{\uparrow}$ row, a finite ledger, RegSeqRat$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1680

A Concrete Naming Certificate for $\mathsf{CauchyRepresentation}^{\uparrow}$

Following Bishop and Bridges and the computable-analysis convention of representing reals by effective Cauchy names, the $\mathsf{CauchyRepresentation}^{\uparrow}$ packet records the finite interface through which stream windows, regular rational readbacks, dyadic tolerances, and a terminal real seal are presented to represented-space consumers. It sits between the stream-name certificate of chapter 126, the regular rational sequence boundary of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the represented-space surface of chapter 1494. The packet supplies a standard Cauchy-representation boundary without identifying it with $\mathsf{ComputableReal}^{\uparrow}$, $\mathsf{BishopReal}^{\uparrow}$, or the completed $\mathsf{Real}^{\uparrow}$ object.

Definition 1680.1 (CauchyRepresentation carrier)**.** *A* $\mathsf{CauchyRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, R, D, A, G, H, C, P, E, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row. It records the certified stream name whose finite windows are inspected.*
2. *W is the finite window schedule opened from S at the requested observation indices and precision levels.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row turning the displayed stream windows into regular rational observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the regular rational observations at the displayed scales.*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, R, D have been replayed.*
6. *G is the* $\mathsf{RepresentedSpace}^{\uparrow}$*-facing source row, recording that the same finite name is being offered as a represented-space source rather than as a host real or quotient stream.*
7. *H records componentwise* hsame *transport for the stream source, window schedule, regular readback, dyadic tolerance ledger, real seal, represented-space source row, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from source-window inspection through regular readback, dyadic tolerance comparison, represented-space exposure, and the final seal handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *E is the local evidence row recording that every represented-space read of the packet factors through the same S, W, R, D, A route.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRepresentation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite Cauchy-representation packet. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyRepresentationUp.CauchyRepresentationTasteGate single carrier alignment`

Theorem 1680.2 (CauchyRepresentation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRepresentation}^{\uparrow}$ *carrier $K = (S, W, R, D, A, G, H, C, P, E, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRepresentation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream source, finite window schedule, regular rational readback, dyadic tolerance ledger, real seal, represented-space source row, transport, replay, provenance, evidence, and local naming rows;*
2. *window exactness, requiring each finite observation to be read first through S and W before any regular rational row is consulted;*
3. *regular readback exactness, requiring R to expose only the rational observations generated by the displayed stream windows;*
4. *dyadic tolerance exactness, requiring D to record the finite tolerance comparisons used to justify the regular readback route;*
5. *real-seal non-escape, requiring A to remain downstream of S, W, R, D and never to serve as a host-level selected limit;*
6. *represented-space source compatibility, requiring G and E to expose the same finite route to* $\mathsf{RepresentedSpace}^{\uparrow}$ *consumers;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the displayed packet;*
8. *refusal of quotient streams, selected limits, host real equality, countable choice, and ambient completeness as local NameCert data.*

No Cauchy-representation obligation is stored outside these rows.

Proof. Project the accepted carrier. The finite analytic route is exactly the stream row S, the window schedule W, the regular rational readback R, the dyadic tolerance ledger D, and the seal row A.

Window exactness and regular readback exactness follow from this route order. A consumer cannot reach R without first displaying the finite window schedule opened from S, and the tolerance ledger D reads only the regular observations exposed by R.

The represented-space boundary is carried by G and E. These rows expose the same finite route to represented-space consumers, so they do not add a second source of limits or equality.

Finally, H, C, P, N transport, replay, package, and name the packet without adding rows. Since the carrier contains no coordinate for quotient streams, selected limits, host real equality, countable choice, or ambient completeness, none of those readings can become a local obligation. □

Lean4 ✓ `BEDC.Derived.CauchyRepresentationUp.CauchyRepresentationCarrier namecert obligations`

Theorem 1680.3 (CauchyRepresentation stream and regular handoff)**.** *Let* $K = (S, W, R, D, A, G, H, C, P, E, N)$ *be an accepted* $\mathsf{CauchyRepresentation}^{\uparrow}$ *carrier. Every* $\mathsf{RepresentedSpace}^{\uparrow}$*- or* $\mathsf{Real}^{\uparrow}$*-facing read of* K *factors through the finite handoff route*

$$S, W \longmapsto R \longmapsto D \longmapsto A \longmapsto G, E, H, C, P, N.$$

Thus the stream name and regular rational readback are the only source rows by which the real seal and represented-space exposure are reached. The route contains no quotient stream, selected limit, host real equality, countable-choice selector, or ambient completeness principle.

Proof. Start with a consumer read of the packet. Carrier admission gives access to the seal and represented-space rows, but only as rows in the same finite packet as S, W, R, D.

Replay the continuation row C. The read first opens the $\mathsf{StreamName}^{\uparrow}$ source and its finite schedule, then consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and then consults the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger.

Only after this replay can the $\mathsf{Real}^{\uparrow}$ seal row and the $\mathsf{RepresentedSpace}^{\uparrow}$-facing row be read. The evidence, transport, provenance, and local naming rows preserve that same route, so the consumer receives no quotient stream, selected limit, host real equality, countable choice, or ambient completeness data. □

Lean4 ✓ `BEDC.Derived.CauchyRepresentationUp.CauchyRepresentationStreamRegSeqRatHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRepresentation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRepresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1680.1, Definition 1680.2, and Definition 1680.3 seed the finite Cauchy-representation boundary over $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, a $\mathsf{Real}^{\uparrow}$ seal row, and $\mathsf{RepresentedSpace}^{\uparrow}$ exposure.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRepresentation}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyRepresentationUp.CauchyRepresentationStreamRegSeqRatHandoff`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, regular readback exactness, dyadic tolerance exactness, seal non-escape, represented-space compatibility, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRepresentationUp.

Constructive story: A $\mathsf{CauchyRepresentation}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ source, finite stream windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, a $\mathsf{RepresentedSpace}^{\uparrow}$ source row, componentwise hsame transports, displayed Cont replay, Pkg provenance, a local evidence row, and a local NameCert row.

Chapter 1681

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ packet names a finite located-comparison surface for two regular Cauchy rational names before any completed-real order principle is exposed. It sits over the regular rational rows of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, the Real seal boundary of chapter 38, the regular Cauchy separation route of chapter 1650, and the apartness-real boundary of chapter 304. The certificate records a located order read as finite NameCert data; it does not import constructive-real trichotomy, host real equality, quotient stream equality, selected limits, countable choice, or ambient completeness.

Definition 1681.1 (Regular Cauchy located order carrier)**.** *A* $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, R, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
2. *R is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row.*
3. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger shared by the left and right observations.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the dyadic radius, slack, and rational comparison budget used by the located read.*
5. *Q is the located comparison row, displaying the finite evidence that the inspected windows place the two names to the left, to the right, or still inside the unresolved tolerance band named by D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, R, W, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the two regular rational names, the common window, dyadic tolerance row, located comparison row, Real seal, continuation, provenance, and local naming row.*
8. *C records displayed* Cont *replay from each located-order request through $L, R, W, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySeparation}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host order on completed reals, quotient stream equality, trichotomy, a selected limit, countable choice, Real completeness, or a standard bridge.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedOrderUp.RegularCauchyLocatedOrderUp single carrier alignment`

Theorem 1681.2 (Regular Cauchy located order NameCert obligations)**.** *For every accepted carrier $O = (L, R, W, D, Q, E, H, C, P, N)$ of Definition 1681.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedOrder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *rows L, R, the shared* $\mathsf{StreamName}^{\uparrow}$ *window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the located comparison row Q, the* $\mathsf{Real}^{\uparrow}$ *seal row E, and the structural rows H, C, P, N;*
2. *classifier stability, requiring every accepted located-order read to inspect the same displayed route L, R, W, D, Q before the Real seal row is exposed;*
3. *common-window transport, requiring* hsame *transport through H and replay through C to preserve the same left and right observation windows named by W;*
4. *comparison-row exactness, requiring the left, right, and unresolved-band outcomes to be read only from the finite dyadic comparison data stored in D and Q;*
5. *Real-seal handoff, requiring E to be consumed only after the source names, common window, dyadic tolerance, and located comparison rows have been retained;*
6. *ledger refusal, excluding trichotomy, host real order, quotient stream equality, selected limits, countable choice, ambient completeness, Lean verification, and bridge checking from the local NameCert surface.*

No obligation row is stored outside the displayed packet.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedOrderUp.RegularCauchyLocatedOrderCarrier namecert obligations`

Proof. Project the carrier. The rows L, R, W, D, Q, E are the analytic coordinates, while H, C, P, N transport, replay, source, and name the same finite packet.

Read the classifier in route order. A located-order request first reads the two regular rational names L and R, then aligns them over the common finite window W, then consults the dyadic tolerance ledger D, and only then reads the comparison row Q.

The common-window condition follows from the structural rows. Componentwise hsame in H preserves the left and right windows together, and Cont replay in C repeats the same order of reads rather than allowing a detached comparison row.

The Real-facing row is downstream of the comparison route. Since E appears only after L, R, W, D, Q, any consumer that bypasses those rows has not obtained the displayed packet's Real seal boundary.

The excluded principles have no carrier coordinate. Thus the listed carrier admission, classifier stability, common-window transport, comparison exactness, Real-seal handoff, and ledger-refusal rows exhaust the local NameCert surface. □

Theorem 1681.3 (Regular Cauchy located order Real seal boundary)**.** *For every accepted* RegularCauchyLocatedOr *packet $O = (L, R, W, D, Q, E, H, C, P, N)$, every Real-facing read factors through the finite route*

$$(L, R) \longmapsto W \longmapsto D \longmapsto Q$$

before reaching the $\mathsf{Real}^{\uparrow}$ *seal boundary* E*. Consequently the packet exports no trichotomy, host real equality, quotient stream, selected limit, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the obligation surface of Definition 1681.2. Carrier admission exposes the seal row only as part of the displayed finite packet, and classifier stability places every located-order read on the route through L, R, W, D, Q.

The comparison row is the last analytic gate before the seal boundary. It reads only the dyadic tolerance and rational comparison data already stored in D over the common window W, so it cannot select a completed real, quotient a stream, or invoke a host order.

Finally consume E as a boundary row. The transport, continuation, provenance, and local naming rows preserve the same route, so any Real-facing read must pass through the finite located-comparison packet before it reaches the seal. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedOrderUp.RegularCauchyLocatedOrderCarrier real seal boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ packet is generated from left and right RegSeqRat rows, a shared StreamName window, a DyadicRatCore tolerance ledger, a located comparison row, a RealUp seal boundary, componentwise hsame transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1681.1, Definition 1681.2, and Definition 1681.3 seed the finite located-order boundary for regular Cauchy rational names over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySeparation}^{\uparrow}$, and $\mathsf{ApartnessReal}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full $\mathsf{RealAlgOrder}^{\uparrow}$, trichotomy, host real equality, quotient stream equality, selected limits, countable choice, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, common-window transport, comparison-row exactness, Real-seal handoff, and ledger refusal for the displayed located-order packet.

Chapter 1682

A Concrete Naming Certificate for $\mathsf{ToeplitzLemma}^{\uparrow}$

Following Toeplitz and Hardy on summability transforms, the $\mathsf{ToeplitzLemma}^{\uparrow}$ packet records the finite matrix-window route by which a regular real-name sequence and a Toeplitz row ledger produce a regular transformed sequence before any host limit theorem is read. It sits between the series and uniform-limit consumers of chapter 1581 and chapter 463, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic row-sum ledger of chapter 291, and the terminal real-name seal of chapter 38. The same matrix-window transform is the sibling ledger consumed by chapter 1765 and chapter 1827 before any host summability or selected-limit reading is made. The packet is a finite certificate for a regular summability transform; it is not a host summability theorem, a selected limit, quotient equality of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1682.1 (Toeplitz lemma carrier)**.** *A* $\mathsf{ToeplitzLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the matrix-coefficient window row, listing the finite Toeplitz rows and columns currently inspected together with their row-sum and entry-size requests.*
2. *W is the selected source-window row, recording the finite* $\mathsf{StreamName}^{\uparrow}$ *observations of the input regular real-name sequence used by the inspected matrix rows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected source windows and for the transformed finite row produced by the same matrix window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for row sums, coefficient truncation, source-window tolerances, small-entry estimates, and accumulated arithmetic error.*
5. *T is the transformed finite partial row, obtained by applying the displayed coefficients of A to the selected source values in W and reading the result through R and D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the transformed row T, regular readback R, and dyadic ledger D have been displayed.*
7. *H stores componentwise* hsame *transport for the matrix window, source windows, readback, dyadic ledger, transformed row, real seal, continuation, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from matrix-row inspection to selected source windows, from source windows to regular readback, from readback and dyadic accounting to the transformed row, and from the transformed row to the real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ToeplitzLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, R, D, T, E, H, C, P, N$. It has no coordinate for an infinite matrix object, a host summability method, a selected limit of the source sequence, a quotient of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ToeplitzLemmaUp.ToeplitzLemmaTasteGate single carrier alignment`

Theorem 1682.2 (Toeplitz lemma NameCert obligations)**.** *For every accepted* $\mathsf{ToeplitzLemma}^{\uparrow}$ *carrier $K = (A, W, R, D, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ToeplitzLemma}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the displayed matrix-window, source-window, readback, ledger, transformed-row, seal, transport, continuation, provenance, and naming rows;*

2. *row-sum boundedness, read from A and checked inside the dyadic ledger D before any transformed row is exposed;*

3. *small-entry and window-vanishing control, requiring every displayed coefficient request outside the selected finite source window to pass through the same A, W, D route;*

4. *regular transformed handoff, requiring every transformed-sequence read to pass through T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the* $\mathsf{Real}^{\uparrow}$ *seal E;*

5. *ledger nonescape, excluding host summability methods, selected source limits, quotient real-name equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet surface.*

Lean4 ✓ `BEDC.Derived.ToeplitzLemmaUp.ToeplitzLemmaNamecertObligations`

Proof. Project the carrier of Definition 1682.1. Carrier admission is exactly the inventory of $A, W, R, D, T, E, H, C, P, N$, and the matrix-row information available to the classifier is confined to the finite row A.

The row-sum and small-entry clauses are read through the same dyadic ledger D. The selected source windows W provide the only input sequence observations, so any window-vanishing claim must use the displayed coefficient requests in A together with the finite source reads in W.

The transformed row T is then exposed through R and E. The structural rows H, C, P, N transport, replay, source, and name this route, and the carrier contains no coordinate for the excluded host principles. □

Theorem 1682.3 (Toeplitz lemma regular sequence handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the transformed sequence exported by an accepted* $\mathsf{ToeplitzLemma}^{\uparrow}$ *carrier factors through*

$$A, W, R, D, T, E, H, C, P, N.$$

Thus the consumer reads a finite Toeplitz matrix window, selected source windows, regular rational readback, dyadic row-sum and error accounting, transformed finite partial row, real seal, transport, continuation, provenance, and local naming before it can use the transformed regular sequence.

Lean4 ✓ `BEDC.Derived.ToeplitzLemmaUp.ToeplitzLemmaRegularSequenceHandoff`

Proof. Start with a transformed-sequence consumer. By Definition 1682.1, the only transformed row coordinate is T, and it is built from the matrix-window row A, selected source windows W, regular readback R, and dyadic ledger D.

For a $\mathsf{RegSeqRat}^{\uparrow}$ read, the handoff is therefore already forced through R after A, W, D have supplied the coefficient, source, and error data. For a $\mathsf{Real}^{\uparrow}$ read, the same route continues to the seal row E.

The rows H, C, P, N preserve the route under transport, replay the same continuations, record package provenance, and bind the local NameCert. Since no second transformed-sequence coordinate is present, every downstream consumer factors through the displayed packet. □

Theorem 1682.4 (ToeplitzLemma Cesaro consumer route)**.** *Any* $\mathsf{HardyCesaroMean}^{\uparrow}$ *or* $\mathsf{CesaroConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{ToeplitzLemma}^{\uparrow}$ *packet must read the finite route*

$$A, W, R, D, T, E, H, C, P, N.$$

Thus the consumer receives the finite matrix window, source windows, $\mathsf{DyadicRatCore}^{\uparrow}$ *row-sum and error ledger, transformed finite partial row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ToeplitzLemma}^{\uparrow}}$ *row before it can use the Toeplitz transform as a Cesaro-facing input.*

The route exports no host summability theorem, selected source limit, quotient equality of real-name streams, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.ToeplitzLemmaUp.ToeplitzLemmaCesaroConsumerRoute`

Proof. Start with the NameCert obligations of Definition 1682.2. They force carrier admission, row-sum boundedness, small-entry and window-vanishing control, regular transformed handoff, and ledger nonescape to be read from $A, W, R, D, T, E, H, C, P, N$.

Apply the regular sequence handoff Definition 1682.3. It places every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read of the transformed sequence behind the matrix window, selected source windows, dyadic ledger, transformed row, and real seal.

The Cesaro-facing chapters consume only that transformed regular sequence. A $\mathsf{HardyCesaroMean}^{\uparrow}$ or $\mathsf{CesaroConvergence}^{\uparrow}$ reader therefore receives the same displayed Toeplitz packet, with H, C, P, N preserving transport, replay, provenance, and local naming. Since no host summability, selected limit, quotient stream equality, choice, or completeness row occurs in the carrier, none is exported to the Cesaro consumer. □

Falsifiable prediction: Every $\mathsf{ToeplitzLemma}^{\uparrow}$ transformed-sequence read factors through a finite matrix-coefficient window, selected source windows, RegSeqRat readback, a DyadicRatCore row-sum and error ledger, a transformed finite partial row, a RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: realseries, uniformlimit, streamname, regseqrat, dyadicratcore, real: $\mathsf{ToeplitzLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries matrix-coefficient windows, selected source windows, regular readback, dyadic row-sum and small-entry accounting, transformed-row construction, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ToeplitzLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ToeplitzLemma}^{\uparrow}$ packet is generated from finite Toeplitz matrix windows, selected source windows, RegSeqRat readback, a DyadicRatCore row-sum and error ledger, a transformed finite partial row, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ToeplitzLemma}^{\uparrow}}$, seed)

Scope closed: Definition 1682.1, Definition 1682.2, Definition 1682.3, and Definition 1682.4 bind the finite summability-transform certificate surface and the Cesaro-facing consumer route.

Formal verification: FormalStatus(NameCert$_{\mathsf{ToeplitzLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host summability theorems, selected source limits, quotient equality of real-name streams, countable choice, ambient Real$^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, row-sum boundedness, small-entry and window-vanishing control, regular transformed handoff, ledger nonescape, and TasteGate field faithfulness for BEDC.Derived.ToeplitzLemmaUp.

Chapter 1683

A Concrete Naming Certificate for $\mathsf{SequentialCompletion}^{\uparrow}$

Following Bishop and Bridges' regular-sequence completion language and Kelley's sequence-completion terminology, the $\mathsf{SequentialCompletion}^{\uparrow}$ packet names the finite L10 route by which displayed stream schedules, regular rational readbacks, dyadic tolerance ledgers, Cauchy criterion rows, and terminal real seals are offered to Real and Completion consumers. It sits on the same human-origin spine as chapter 126, chapter 301, chapter 291, chapter 1509, chapter 386, and chapter 38. The packet records a sequential completion interface without asserting ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient-stream equality, selected limits, countable choice, trichotomy, or open-cover compactness.

Definition 1683.1 (SequentialCompletion carrier)**.** *A* $\mathsf{SequentialCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, Q, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It records the finite stream-name schedule inspected by a Real-facing or Completion-facing request.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed schedule windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, slack rows, and finite comparison budgets used by the readback.*
4. *K is the criterion row, either a* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *exposure row or a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row, read only after S, Q, D have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached downstream of the schedule, readback, tolerance, and criterion rows.*
6. *H records componentwise* hsame *transport for the schedule, readback, tolerance ledger, criterion or limit-seal row, terminal real seal, continuation row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Real or Completion request through S, Q, D, K, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite sequential-completion packet. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.SequentialCompletionUp.TasteGate.SequentialCompletionTasteGate single carrier alignment`

Theorem 1683.2 (SequentialCompletion NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompletion}^{\uparrow}$ *carrier* $J = (S, Q, D, K, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletion}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed schedule, regular readback, dyadic tolerance ledger, Cauchy criterion or limit-seal row, terminal real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *criterion exposure, requiring* K *to be read only after the same* S, Q, D *route has been displayed;*
3. *regular readback, requiring each completion-facing read to obtain its rational observations through* Q *and the dyadic ledger* D *before the terminal seal is exposed;*
4. *Real seal non-escape, requiring* E *to remain downstream of* S, Q, D, K *and never to serve as a quotient stream, selected limit, host real equality witness, choice-selected representative, trichotomy decision, compactness witness, or ambient completeness proof;*
5. *transport, provenance, and local naming, requiring* H, C, P, N *to preserve and name exactly the displayed packet.*

No sequential-completion obligation is stored outside these five rows.

Proof. Project the accepted carrier of Definition 1683.1. The projection exposes exactly the schedule row S, the regular readback row Q, the dyadic tolerance ledger D, the Cauchy criterion or limit-seal row K, the terminal real seal E, and the structural rows H, C, P, N.

Criterion exposure follows from the route order displayed by C. A consumer cannot reach K without first opening the stream schedule through S, reading the regular rational data through Q, and consulting the dyadic ledger D.

The regular readback and Real-seal rows are therefore locked to the same finite path. Since E is downstream of S, Q, D, K, the seal cannot introduce a quotient stream, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient completeness.

Finally, H, C, P, N transport, replay, package, and name the same packet. They add no analytic coordinates beyond the rows already projected, so the listed five rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SequentialCompletionUp.SequentialCompletionNamecertObligations`

Theorem 1683.3 (SequentialCompletion RegSeqRat and Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read of an accepted* $\mathsf{SequentialCompletion}^{\uparrow}$ *carrier* $J = (S, Q, D, K, E, H, C, P, N)$ *factors through the finite route*

$$S \longmapsto Q \longmapsto D \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the sequence-completion handoff exports the displayed $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *or* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows. It exports no quotient stream, selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompletionUp.SequentialCompletionRegseqratRealHandoff`
↪ variant `BEDC.Derived.SequentialCompletionUp.SequentialCompletionRegSeqRatRealHandoff`

Proof. Begin with a Real-facing or Completion-facing consumer of J. Carrier admission gives access to the terminal seal only as a row in the same finite packet as the schedule, readback, tolerance, and criterion rows.

Replay the continuation row C. The consumer first reads the $\mathsf{StreamName}^{\uparrow}$ schedule S, then obtains the regular rational observations through Q, then consults the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and only then reaches the criterion or limit-seal row K.

The terminal $\mathsf{Real}^{\uparrow}$ seal E is exposed after that route. Componentwise transport, package provenance, and the local naming row preserve the same factorization, so no quotient stream, selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient completeness data can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1683.1, Definition 1683.2, and Definition 1683.3 seed the finite sequence-completion route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exposure, regular readback, Real-seal non-escape, transport, provenance, and local naming for BEDC.Derived.SequentialCompletionUp.

Constructive story: A $\mathsf{SequentialCompletion}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ or $\mathsf{CauchyLimitSeal}^{\uparrow}$ row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1684

A Concrete Naming Certificate for $\mathsf{CauchySchwarzReal}^{\uparrow}$

Following Hardy–Littlewood–Polya and standard inner-product space treatments, the $\mathsf{CauchySchwarzReal}^{\uparrow}$ packet records the finite route by which a Cauchy–Schwarz inequality read over displayed finite real-vector rows is exposed to downstream analysis consumers. It sits over the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the finite observation windows of chapter 126, the vector-space source of chapter 47, and the norm boundary of chapter 89. The packet is a finite inequality certificate for RealUp row consumers; it is not a host real inner product theorem, quotient stream equality, ambient completeness, an infinite-dimensional Hilbert assertion, or bridge checking.

Definition 1684.1 (CauchySchwarzReal carrier)**.** *A* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *finite-vector source row. It records the displayed vector carrier, scalar action, vector addition, zero, negation, and classifier rows used by the two finite rows.*
2. *X and Y are the two finite vector rows. Their coordinates are exposed as finite families of* $\mathsf{Real}^{\uparrow}$ *seal rows, one family for each inspected coordinate.*
3. *I is the inner-product row. It records the finite coordinatewise multiplication-and-sum read that produces the scalar row $\langle X, Y\rangle$.*
4. *A and B are the two norm-square rows. They record the finite coordinatewise reads for $\|X\|^2$ and $\|Y\|^2$ through the* $\mathsf{Norm}^{\uparrow}$ *boundary.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *square-and-tolerance ledger. It stores the dyadic square comparison budget, slack rows, and finite positivity comparisons used to read $(\langle X, Y\rangle)^2 \leq \|X\|^2\|Y\|^2$.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the real scalar observations in I, A, B, D.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by Q and by the displayed coordinate observations of X and Y.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed inequality value. It is reached only after $V, X, Y, I, A, B, D, Q, S$ have been displayed.*

9. *H records componentwise* hsame *transport for the vector-space source, vector rows, inner-product row, norm-square rows, dyadic ledger, readback, windows, Real seal, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay from a finite coordinate request through S, Q, I, A, B, D, E and back to the public inequality boundary.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzReal}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite Cauchy–Schwarz packet. It has no coordinate for host real order, quotient stream equality, ambient completeness, non-finite vector-space semantics, or a bridge-checked standard theorem.

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.TasteGate.CauchySchwarzRealTasteGate single carrier alignment`

Theorem 1684.2 (CauchySchwarzReal NameCert obligations)**.** *For every accepted* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier $K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space source, two finite real-vector rows, inner-product row, two norm-square rows, dyadic square-and-tolerance ledger, regular rational readback, finite windows, Real seal, transport, replay, provenance, and local naming rows;*

2. *classifier transport, requiring H and T to preserve the same vector rows, scalar reads, square ledger, readback, windows, Real seal, and local naming rows componentwise;*

3. *finite inequality route, requiring every downstream read of Cauchy–Schwarz data to pass through $V, X, Y, I, A, B, D, Q, S, E$ before the certificate row is available;*

4. *dyadic square exactness, requiring the displayed square comparison in D to be the only route by which $(\langle X, Y\rangle)^2 \leq \|X\|^2\|Y\|^2$ is exported;*

5. *ledger non-escape, excluding host real order, quotient stream equality, ambient completeness, non-finite vector-space semantics, hidden square-root extraction, and bridge checking.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealCarrier namecert obligations`

Proof. Project the accepted carrier. The analytic coordinates are the vector-space source V, the displayed finite vector rows X and Y, the scalar inner-product row I, the two norm-square rows A and B, the dyadic square-and-tolerance ledger D, the regular readback Q, the finite window row S, and the terminal Real seal E.

Carrier admission is exactly admission of those displayed coordinates together with H, T, P, N. The local certificate has no additional field from which a host real value, quotient-stream representative, or ambient vector-space object could be recovered.

Classifier transport is structural. Componentwise hsame in H keeps the vector source, vector rows, scalar reads, square ledger, readback, windows, and Real seal aligned, while Cont replay in T repeats the same finite coordinate route for later consumers.

The finite inequality route is forced by the order of the rows. A consumer can inspect the Cauchy–Schwarz boundary only after the vector rows, scalar rows, dyadic square ledger, readback, windows, and Real seal have already been displayed.

Dyadic square exactness follows because D is the only square-comparison coordinate in the carrier. Thus the exported inequality is the displayed square comparison over the finite readback route, not a host-order theorem imported from outside the packet.

Finally, the non-escape clause is immediate from the carrier surface. None of host real order, quotient stream equality, ambient completeness, non-finite vector-space semantics, hidden square-root extraction, or bridge checking is a coordinate of K. □

Theorem 1684.3 (CauchySchwarzReal dyadic square handoff)**.** *For every accepted* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier* $K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$*, every* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, or real-analysis consumer receives the Cauchy–Schwarz inequality only through the displayed finite route*

$$V, X, Y \longmapsto I, A, B \longmapsto D \longmapsto Q, S \longmapsto E \longmapsto H, T, P, N.$$

Consequently the handoff supplies the square-form inequality $(\langle X, Y\rangle)^2 \leq \|X\|^2\|Y\|^2$ *through finite vector rows, norm-square rows, dyadic tolerance comparisons, regular rational readback, finite stream windows, and a RealUp seal; it does not supply host real order, quotient stream equality, ambient completeness, hidden square-root extraction, or a bridge-checked standard object.*

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealCarrier dyadic square handoff`

Proof. Start from an accepted carrier. The only vector data are the source row V and the two finite vector rows X and Y, and the only scalar data derived from them are I, A, B.

The comparison data are then fixed by D. The dyadic row supplies the square comparison budget, positivity slack, and finite tolerance ledger used to compare the inner-product square with the product of the two norm-square rows.

The readback route is fixed by Q, S, E. The regular rational row Q reads the scalar observations through the finite windows S, and the terminal Real row E seals that same displayed route before any downstream consumer receives the inequality boundary.

The rows H, T, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for host real order, quotient stream equality, ambient completeness, hidden square-root extraction, or bridge checking, every consumer receives only the displayed finite dyadic square handoff. □

Theorem 1684.4 (CauchySchwarzReal norm-bound handoff)**.** *For every accepted* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier* $K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$*, every finite consumer that reads the norm-bound shape of Cauchy–Schwarz receives it only through the displayed finite vector rows* X, Y*, the inner-product row* I*, the norm-square rows* A, B*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *square ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* S*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. Thus the consumer-facing read is the finite norm-bound boundary generated by those rows and transported through* H, T, P, N*.*

The handoff has no coordinate for host real order, hidden square-root extraction, quotient streams, ambient completeness, or bridge checking. In particular the norm-bound form is not obtained by selecting square roots inside a host ordered real field; it is read as the finite RealUp-facing consequence of the existing dyadic square handoff.

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealCarrier norm bound handoff`

Proof. Project the carrier definition of Definition 1684.1. The only vector inputs are X and Y over the source row V, the only scalar comparison inputs are I, A, B, D, and the only real readback inputs are Q, S, E.

Apply the NameCert obligations of Definition 1684.2. They force every downstream Cauchy–Schwarz read to expose the finite vector rows, inner-product row, norm-square rows, dyadic ledger, regular readback, stream windows, Real seal, transport, replay, provenance, and local naming rows.

Finally use the dyadic square handoff of Definition 1684.3. It is the unique route exporting the square-form comparison to RealUp consumers, so the norm-bound read can only be the consumer-facing boundary obtained from that finite square ledger and seal. Since none of the excluded analytic coordinates is present in the carrier, no host real order, hidden square-root extraction, quotient stream, ambient completeness, or bridge row can supply an alternate handoff. □

Theorem 1684.5 (CauchySchwarzReal inner-product bound route)**.** *For every accepted* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier* $K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$*, the complete route by which downstream* $\mathsf{HolderInequality}^{\uparrow}$ *and* $\mathsf{MinkowskiInequality}^{\uparrow}$ *consumers read the inner-product bound is the row sequence*

$$V, X, Y \longmapsto I, A, B \longmapsto D \longmapsto Q, S \longmapsto E \longmapsto H, T, P, N.$$

The row I *supplies the finite inner-product read,* A *and* B *supply the two norm-square reads,* D *supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *square ledger,* Q *supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* S *supplies the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and* E *supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No additional row is available between the square ledger and the downstream Holder or Minkowski consumer.*

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealInnerproductBound`

Proof. Project the carrier. The displayed vector information is exhausted by V, X, Y, and the scalar inequality information is exhausted by I, A, B, D.

The readback and sealing rows are then forced. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q and $\mathsf{StreamName}^{\uparrow}$ row S expose the finite observations used by the dyadic comparison, and the $\mathsf{Real}^{\uparrow}$ row E is the only terminal seal available before the transport, replay, provenance, and naming rows H, T, P, N.

Apply Definition 1684.2 and Definition 1684.4. Together they state that every downstream read of the Cauchy–Schwarz boundary passes through the displayed finite rows and that the norm-bound form is the consumer-facing consequence of the same square ledger. Thus Holder and Minkowski consumers receive exactly this route and no intermediate analytic object. □

Theorem 1684.6 (CauchySchwarzReal non-escape boundary)**.** *For every accepted* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *carrier* $K = (V, X, Y, I, A, B, D, Q, S, E, H, T, P, N)$*, the packet exports no host inner-product theorem, no quotient stream equality, no infinite-dimensional Hilbert theorem, no selected limit, no countable choice principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem. Every exported Cauchy–Schwarz row is read through the finite carrier coordinates of Definition 1684.5.*

Lean4 ✓ `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealCarrier nonescape boundary`

Proof. Inspect the carrier coordinates in Definition 1684.1. They name finite vector rows, scalar reads, a dyadic square comparison ledger, regular readback, finite windows, a terminal Real seal, and structural transport data.

None of the listed excluded objects is a coordinate of the carrier. There is no host inner-product theorem row, no quotient-stream equality row, no Hilbert-space theorem row, no selected-limit row, no countable-choice row, and no ambient completeness row.

The only exported inequality route is therefore the one isolated in Definition 1684.5. Since every downstream read factors through that finite route, the excluded objects cannot be recovered from the packet as public data. □

Falsifiable prediction: Every $\mathsf{CauchySchwarzReal}^{\uparrow}$ inequality read factors through a VecSpaceUp finite-vector source, two RealUp finite vector rows, an inner-product row, two NormUp square rows, a DyadicRatCore square-and-tolerance ledger, RegSeqRat readback, StreamName windows, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: real, dyadicratcore, regseqrat, streamname, vecspace, norm: $\mathsf{CauchySchwarzReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite vector rows, inner-product readback, norm-square rows, dyadic square comparison, regular readback, stream windows, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchwarzReal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySchwarzReal}^{\uparrow}$ packet is generated from a VecSpaceUp finite-vector source, two finite RealUp vector rows, an inner-product row, two NormUp square rows, a DyadicRatCore square-and-tolerance ledger, RegSeqRat readback, StreamName windows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySchwarzReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1684.1, Definition 1684.2, and Definition 1684.3 seed the finite Cauchy–Schwarz inequality boundary for displayed RealUp finite-vector rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySchwarzReal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchySchwarzRealUp.CauchySchwarzRealCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close host real order, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, non-finite vector-space semantics, hidden square-root extraction, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite inequality routing, dyadic square exactness, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchySchwarzRealUp.

Chapter 1685

A Concrete Naming Certificate for $\mathsf{RealBaireCylinder}^{\uparrow}$

Following Brouwer's finite-prefix reading of choice sequences and the Bishop–Bridges insistence that real-number data be displayed by effective finite approximations, the $\mathsf{RealBaireCylinder}^{\uparrow}$ packet names a Baire-prefix cylinder only through the finite rows that a Real-facing consumer can inspect. It sits after the prefix-cylinder surface of chapter 334, the finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite readback boundary for the open Real and Completion route; it is not a quotient stream, a host function equality, a countable-choice selector, a completed Baire branch, or a Real completeness theorem.

Definition 1685.1 (Real Baire-cylinder carrier)**.** *A* $\mathsf{RealBaireCylinder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is a* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet. It records the finite natural-index window, displayed unary answer rows, prefix-cylinder classifier, and local Baire ledger.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected from B. It exposes the scheduled requests that a Real-facing read is allowed to replay.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It converts the selected stream window into the regular rational approximation rows visible to the Real route.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal. It is reached only after the Baire prefix, StreamName window, and RegSeqRat readback have been displayed.*
5. *H records componentwise* hsame *transport for the Baire prefix packet, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a finite Real-facing request back through E, Q, S, B.*
7. *P is the* Pkg *provenance over* $\mathsf{RealBaireCylinder}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBaireCylinder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, S, Q, E, H, C, P, N. It has no coordinate for host sequence equality, quotient real equality, countable choice, ambient completeness, or a tail of the Baire stream outside the displayed prefix window.

Lean4 ✓ `BEDC.Derived.RealBaireCylinderUp.TasteGate.RealBaireCylinderTasteGate single carrier alignment`

Theorem 1685.2 (Real Baire-cylinder NameCert obligations)**.** *For every accepted* $\mathsf{RealBaireCylinder}^{\uparrow}$ *carrier* $R = (B, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealBaireCylinder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed BaireSpace prefix packet, StreamName finite window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *prefix-cylinder stability, requiring every transported read to remain inside the finite Baire prefix packet B and its displayed classifier rows;*
3. *StreamName handoff, requiring every Real-facing request to select its finite schedule through S before any regular rational readback is exposed;*
4. *RegSeqRat readback, requiring the scheduled window to pass through Q before the Real seal can be read;*
5. *Real-seal non-escape, requiring E to seal only the displayed readback route and to expose no host real equality, selected limit, countable choice, or ambient completeness principle;*
6. *provenance exhaustion, requiring P and N to name only B, S, Q, E, H, C and to export no principle absent from those rows.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1685.1. The projected rows are exactly the Baire prefix packet B, the StreamName finite window S, the RegSeqRat readback Q, the Real seal E, and the structural rows H, C, P, N.

Prefix-cylinder stability is inherited from the BaireSpace packet. Transport may change the displayed rows only through the componentwise hsame data named by H and replayed by C, so a transported request remains inside the same finite prefix-cylinder surface.

The StreamName handoff and RegSeqRat readback are ordered by the carrier itself. A Real-facing request first chooses the finite window S from B, then reads the regular rational approximation data through Q.

Finally the seal row E is downstream of Q. Since P and N name only this route, there is no carrier coordinate from which a tail stream, quotient equality, countable-choice selector, selected limit, or ambient completeness theorem can be exported. □

Lean4 ✓ `BEDC.Derived.RealBaireCylinderUp.RealBaireCylinderCarrier namecert obligations`

Theorem 1685.3 (Real Baire-cylinder stream-real handoff)**.** *Let $R = (B, S, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RealBaireCylinder}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by R factors through the finite route*

$$B \longmapsto S \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus a consumer receives a $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet, then a* $\mathsf{StreamName}^{\uparrow}$ *finite window, then a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No read can bypass this route through host function equality, quotient streams, countable choice, or Real completeness.*

Lean4 ✓ `BEDC.Derived.RealBaireCylinderUp.RealBaireCylinderStreamRealHandoff`

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing read of R. By Definition 1685.1, the only Real coordinate is E, and E is placed after the prefix packet B, the finite window S, and the regular rational readback Q.

The Baire prefix supplies only the displayed finite cylinder rows. The StreamName row selects the schedule coordinates available to the consumer, and the RegSeqRat row converts that scheduled window into the rational readback visible to the Real route.

The rows H, C, P, N transport, replay, name, and record the same dependency order. Therefore every Real-facing read factors through the displayed prefix-window route and cannot expose any excluded host or completeness datum. □

Falsifiable prediction: Every $\mathsf{RealBaireCylinder}^{\uparrow}$ Real-facing read factors through a BaireSpace prefix packet, StreamName finite window, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming row.

Independence witness: bairespace, streamname, regseqrat, real: $\mathsf{RealBaireCylinder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Baire prefix-cylinder data, finite stream-window selection, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBaireCylinder}^{\uparrow}}$.

Constructive story: A $\mathsf{RealBaireCylinder}^{\uparrow}$ packet is generated from a BaireSpace prefix packet, a StreamName finite window, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBaireCylinder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1685.1, Definition 1685.2, and Definition 1685.3 seed the finite Baire-cylinder readback surface for Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBaireCylinder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, host function equality, countable choice, completed Baire branches, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, prefix-cylinder stability, StreamName handoff, RegSeqRat readback, Real-seal non-escape, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.RealBaireCylinderUp.

Chapter 1686

A Concrete Naming Certificate for $\mathsf{MooreOsgood}^{\uparrow}$

Following Moore and Osgood on iterated-limit interchange, the $\mathsf{MooreOsgood}^{\uparrow}$ packet records the finite route by which a two-parameter real-name window may hand a first-index limit row through a uniform second-index tail ledger before a downstream consumer reads the opposite iterated limit. It sits between the double-sequence surface of chapter 460, the uniform-limit discipline of chapter 463, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite certificate for an iterated-limit handoff; it is not a host double-limit convergence theorem, quotient equality of arrays, a selected limiting value, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1686.1 (Moore-Osgood carrier)**.** *A* $\mathsf{MooreOsgood}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, F, S, U, Q, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the two-parameter window family, listing the finite rectangle of inspected indices together with the requested first-index and second-index tolerances.*
2. *F is the first-limit row, recording the displayed regular readback of the first iterated direction over the windows in W.*
3. *S is the second-limit row, recording the displayed regular readback of the second iterated direction over the same window family.*
4. *U is the uniform tail ledger in one index, binding a finite tolerance request to a tail bound that is stable across the displayed cofinal side of the other index.*
5. *Q is the cofinal schedule row, naming the finite schedule that routes window enlargement, tail entry, and iterated-limit handoff.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the two iterated regular-rational reads exposed by F and S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for rectangle tolerances, uniform tails, cofinal schedule bounds, and accumulated arithmetic error.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the window family, iterated rows, uniform ledger, schedule, regular readback, and dyadic accounting have been displayed.*

9. *H stores componentwise* hsame *transport for the window, iterated-limit, uniform-ledger, schedule, readback, dyadic, real-seal, continuation, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from rectangle inspection to first-limit and second-limit rows, from the uniform tail ledger to the cofinal schedule, from regular readback and dyadic accounting to the real seal, and from the seal to the local NameCert row.*

11. *P is the* Pkg *provenance over* $\mathsf{MooreOsgood}^{\uparrow}$, $\mathsf{CauchyDoubleSequence}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreOsgood}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, F, S, U, Q, R, D, E, H, C, P, N$. It has no coordinate for a completed double limit, quotient array equality, a selected limiting representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1686.2 (Moore-Osgood NameCert obligations)**.** *For every accepted* $\mathsf{MooreOsgood}^{\uparrow}$ *carrier $K = (W, F, S, U, Q, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreOsgood}^{\uparrow}}$ *surface consists exactly of the following six finite rows:*

1. *carrier admission for the displayed two-parameter window family, first-limit row, second-limit row, uniform tail ledger, cofinal schedule, regular readback, dyadic ledger, real seal, transport, continuation, provenance, and naming rows;*

2. *rectangle-window stability, requiring every first-index or second-index observation to pass through the same W row before either iterated read is exposed;*

3. *uniform-tail control in one index, requiring U to bind the tail bound before the cofinal schedule Q may route an interchange handoff;*

4. *iterated regular readback, requiring the first-limit and second-limit rows F and S to be exported only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the dyadic tolerance ledger D;*

5. *real-seal admission, requiring the* $\mathsf{Real}^{\uparrow}$ *seal E to occur after W, F, S, U, Q, R, D have all been displayed;*

6. *ledger nonescape, excluding host double-limit convergence, quotient equality of arrays, selected limiting values, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet surface.*

Lean4 ✓ `BEDC.Derived.MooreOsgoodUp.MooreOsgoodTasteGate single carrier alignment`

Proof. Project the carrier of Definition 1686.1. Carrier admission is exactly the inventory of $W, F, S, U, Q, R, D, E, H, C, P, N$, and the only available two-parameter observations are the finite rectangle windows listed by W.

The stability and uniform-tail clauses are then forced by the placement of U and Q. The packet does not allow either iterated row to choose an unrecorded tail; the uniform ledger must first bind the relevant tolerance, and the cofinal schedule must record the finite route by which the chosen rectangle is enlarged.

The two iterated rows pass through R and D before the real seal E is visible. The structural rows H, C, P, N transport, replay, source, and name this same route, while the carrier contains no coordinate for the excluded host principles. □

Theorem 1686.3 (Moore-Osgood iterated-limit handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an iterated-limit interchange exported by an accepted* $\mathsf{MooreOsgood}^{\uparrow}$ *carrier factors through*

$$W, F, S, U, Q, R, D, E, H, C, P, N.$$

Thus the consumer reads a finite two-parameter window family, first-limit row, second-limit row, uniform tail ledger, cofinal schedule, regular rational handoff, dyadic accounting, real seal, transport, continuation, provenance, and local naming before it can use the interchanged iterated-limit surface.

Lean4 ✓ `BEDC.Derived.MooreOsgoodUp.MooreOsgoodIteratedLimitHandoff`

Proof. Start with a consumer of the interchange surface. By Definition 1686.1, the only visible array data is W, and the only visible iterated rows are F and S.

A $\mathsf{RegSeqRat}^{\uparrow}$ read is therefore forced through R after U and Q have supplied the uniform tail and cofinal schedule data. The dyadic ledger D accounts for the displayed tolerances and schedule bounds, so no consumer can bypass the finite ledger route while reading either iterated row.

For a $\mathsf{Real}^{\uparrow}$ read, the same route continues to the seal E. The rows H, C, P, N preserve the handoff under transport, replay the displayed continuations, record package provenance, and bind the local NameCert, so every downstream consumer factors through the displayed packet. □

Theorem 1686.4 (Moore-Osgood uniform tail lock)**.** *For every accepted* $\mathsf{MooreOsgood}^{\uparrow}$ *carrier* $K = (W, F, S, U, Q, R, D, E, H, C, P, N)$*, any consumer that uses both the first-limit row* F *and the second-limit row* S *must read the uniform tail ledger* U *and the cofinal schedule* Q *before reaching the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*.*

If the source two-parameter window is served by $\mathsf{CauchyDoubleSequence}^{\uparrow}$*, this read factors through Definition 460.8 and Definition 460.9 before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is visible. Thus no unscheduled rectangle, quotient array, selected limit, countable-choice row, or ambient real-completeness row can enter the Moore-Osgood handoff.*

Lean4 ✓ `BEDC.Derived.MooreOsgoodUp.MooreOsgoodUniformTailLock`

Proof. Project K along Definition 1686.1. The iterated rows F and S are exposed as rows of the same finite packet, while the carrier places the uniform tail ledger U and the cofinal schedule Q between the two-parameter window family W and the regular readback row R.

Apply Definition 1686.3. A consumer of the interchanged surface can reach R only through the displayed route $W, F, S, U, Q, R, D, E, H, C, P, N$, so the row pair F, S has no independent path around U or Q.

When the window family is supplied by $\mathsf{CauchyDoubleSequence}^{\uparrow}$, use Definition 460.8 to force every rectangle read through the scheduled tail, then use Definition 460.9 to keep completion-facing consumption inside that scheduled surface. Composing these two constraints with the Moore-Osgood handoff leaves only the displayed ledger and schedule route before E, excluding the non-carrier rows named in the statement. □

Theorem 1686.5 (Moore-Osgood Real seal nonescape)**.** *In every accepted* $\mathsf{MooreOsgood}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is downstream of the two-parameter window family, both iterated rows, the uniform tail ledger, the cofinal schedule, the regular readback row, and the dyadic tolerance ledger. Consequently a Real-facing consumer cannot obtain from the packet a host double-limit theorem, quotient equality of arrays, a selected limiting value, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier order in Definition 1686.1. The seal row E is not an independent coordinate; it is exposed after W, F, S, U, Q, R, D have already supplied the finite observations, iterated rows, uniform tail data, schedule data, regular readback, and dyadic accounting.

The nonescape statement follows from the classifier boundary. Accepted packets are compared only through $W, F, S, U, Q, R, D, E, H, C, P, N$, and none of those rows stores a completed double limit, quotient array equality, a selected limit representative, countable-choice selector, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.MooreOsgoodUp.MooreOsgoodCarrier real seal nonescape`

Falsifiable prediction: Every $\mathsf{MooreOsgood}^{\uparrow}$ iterated-limit read factors through a finite two-parameter window family, first-limit and second-limit rows, a uniform tail ledger in one index, a cofinal schedule, RegSeqRat handoff, DyadicRatCore accounting, a RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: cauchydoublesequence, uniformlimit, streamname, regseqrat, dyadicratcore, real: $\mathsf{MooreOsgood}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two-parameter windows, paired iterated-limit rows, uniform tail control, cofinal scheduling, regular readback, dyadic tolerance accounting, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreOsgood}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreOsgood}^{\uparrow}$ packet is generated from finite two-parameter windows, first-limit and second-limit rows, a uniform tail ledger in one index, a cofinal schedule, RegSeqRat handoff, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreOsgood}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1686.1, Definition 1686.2, Definition 1686.3, Definition 1686.4, and Definition 1686.5 bind the finite iterated-limit interchange certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreOsgood}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host double-limit convergence, quotient array equality, selected limiting values, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, rectangle-window stability, uniform-tail control, cofinal scheduling, iterated regular readback, real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.MooreOsgoodUp.

Chapter 1687

A Concrete Naming Certificate for $\mathsf{CompactMetricCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactMetricCompletion}^{\uparrow}$ packet records the finite route by which a complete totally bounded metric completion is exposed as a compact-facing certificate. It reads the finite completion route of chapter 1573, the Cauchy-filter convergence surface of chapter 1588, the finite-net ledger of chapter 127, the compact certificate surface of chapter 60, and the terminal real-name seal of chapter 38. The packet is a finite NameCert surface for complete totally bounded metric completions; it is not open-cover compactness, a quotient completion, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1687.1 (Compact metric completion carrier)**.** *A* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, U, B, K, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *row. It records the source metric carrier, finite-net completion route, completion embedding, separated equality handoff, and total-bounded completion ledger consumed by the compact-facing certificate.*

2. *U is the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *row. It stores the finite entourage, Cauchy-filter basis, convergence boundary, readback, and seal data through which completeness is read.*

3. *B is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net ledger. For each displayed positive tolerance it records the center bundle, membership rows, center-carrier rows, distance witnesses, refinement rows, and rational bound ledger.*

4. *K is the* $\mathsf{Compact}^{\uparrow}$ *export row. It exposes compactness only as a downstream finite-net completion read, not as an independent open-cover or host-topological compactness object.*

5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite-net completion row T, complete uniform-space row U, total-bounded ledger B, and compact export row K have been displayed.*

6. *H records componentwise* hsame *transport for the completion route, uniform-space convergence route, total-bounded ledger, compact export, Real seal, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a finite-net tolerance request through total-bounded completion, complete uniform-space convergence, compact export, and the Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{CompactMetricCompletion}^{\uparrow}$, $\mathsf{TotallyBoundedCompletion}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, U, B, K, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient representatives, selected limits, countable-choice schedules, host real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompactMetricCompletionUp.TasteGate.CompactMetricCompletionTasteGate single carrier alignment`

Theorem 1687.2 (Compact metric completion NameCert obligations)**.** *For every accepted* CompactMetricComplet*carrier $M = (T, U, B, K, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed total-bounded completion route, complete uniform-space convergence row, total-bounded finite-net ledger, compact export, Real seal, transport, replay, provenance, and local naming rows;*

2. *finite-net completion handoff, requiring every compact-facing tolerance request to read the* TotallyBoundedComple*row T together with the* $\mathsf{TotallyBounded}^{\uparrow}$ *ledger B before any compact export is available;*

3. *complete uniform-space seal, requiring every completion convergence read to pass through the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *row U and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

4. *compact export, requiring the* $\mathsf{Compact}^{\uparrow}$ *row K to be downstream of T, U, B, E and to expose no independent open-cover, quotient, selected-limit, countable-choice, host-equality, or ambient-completeness datum;*

5. *transport and replay stability, requiring H and C to preserve the same ordered route under componentwise sameness and displayed continuation replay;*

6. *ledger non-escape, requiring P and N to name only the displayed rows and to export no principle absent from them.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1687.1. The projected rows are exactly the total-bounded completion route T, complete uniform-space route U, finite-net ledger B, compact export K, Real seal E, and structural rows H, C, P, N.

The finite-net completion handoff reads chapter 1573 together with chapter 127. The row T supplies the completion-facing route, while B supplies the tolerance-indexed centers, membership rows, distance witnesses, and refinement ledger consumed by that route.

The complete uniform-space seal reads chapter 1588 before it reaches chapter 38. Thus the convergence request is replayed through U and only then sealed by E.

The compact export reads chapter 60 as a downstream consumer of the same route. Since K is placed after T, U, B, E, it cannot supply an open-cover compactness object or a host-level compactness theorem independently of the displayed finite-net completion path.

Transport and replay stability are precisely the roles of H and C. The provenance and naming rows P, N name only this packet, so the excluded principles have no carrier coordinate, replay edge, or ledger row from which to escape. □

Lean4 ✓ `BEDC.Derived.CompactMetricCompletionUp.CompactMetricCompletionCarrier namecert obligations`

Theorem 1687.3 (Compact metric completion finite-net seal)**.** *Let $M = (T, U, B, K, E, H, C, P, N)$ be an accepted* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Compact}^{\uparrow}$ *read exported by M factors through the finite-net completion route*

$$B \longmapsto T \longmapsto U \longmapsto E \longmapsto K \longmapsto H, C, P, N.$$

Consequently a compact-facing consumer receives total-bounded finite nets, their completion handoff, complete uniform-space convergence, the Real seal, compact export, transport, replay, provenance, and local naming before it can read compactness.

Proof. Begin with a $\mathsf{Compact}^{\uparrow}$ read of M. By Definition 1687.1, the only compact coordinate is K, and the definition places it downstream of the finite-net ledger B, the total-bounded completion route T, the complete uniform-space row U, and the Real seal E.

The first part of the route is fixed by chapter 127 and chapter 1573. A tolerance request reads finite centers, membership rows, distance witnesses, and refinements through B, and the completion-facing handoff then passes through T rather than through a quotient completion or selected limit.

The second part of the route is fixed by chapter 1588 and chapter 38. Completion convergence is read through U, and the terminal Real-facing datum is sealed by E before the compact export is exposed.

Finally, chapter 60 is reached only at K. The rows H, C, P, N preserve and name the same dependency order, so every exported compact read factors through the displayed finite-net completion route. □

Lean4 ✓ `BEDC.Derived.CompactMetricCompletionUp.CompactMetricCompletionFiniteNetSeal`

Theorem 1687.4 (Compact metric completion uniform-modulus seal)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{CompactModulusRealizer}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *packet must first read the finite-net ledger B, completion route T, complete-uniform row U, Real seal E, and compact export K. The handoff composes the finite-net seal of Definition 1687.3 with the compact-uniform finite fold surface of Definition 9237.5:* $\mathsf{TotallyBounded}^{\uparrow}$ *supplies finite centers and refinements,* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *supplies the completion convergence row,* $\mathsf{Real}^{\uparrow}$ *supplies the terminal seal, and* $\mathsf{Compact}^{\uparrow}$ *supplies the compact-facing export before any uniform-modulus read is public.*

Proof. Begin with an accepted $\mathsf{CompactMetricCompletion}^{\uparrow}$ carrier. By Definition 1687.3, every compact-facing read factors through B, T, U, E, K before the structural rows are consumed.

Now let the downstream consumer be a $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{CompactModulusRealizer}^{\uparrow}$ packet. The compact-uniform finite fold surface of Definition 9237.5 needs compact finite-net data before it can fold pointwise radii into a uniform precision.

The only compact finite-net data exposed by the completion packet are the displayed $\mathsf{TotallyBounded}^{\uparrow}$ ledger B, the $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ route T, the $\mathsf{CompleteUniformSpace}^{\uparrow}$ row U, the $\mathsf{Real}^{\uparrow}$ seal E, and the $\mathsf{Compact}^{\uparrow}$ export K. Therefore any uniform-modulus consumer must read those rows first, and it cannot replace them by open-cover compactness, quotient completion, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CompactMetricCompletionUp.CompactMetricCompletionUniformModulusSeal`

Theorem 1687.5 (Compact metric completion Lebesgue-number consumer route)**.** *Every Lebesgue-number, finite-cover, or cover-radius consumer of an accepted* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *packet must first read the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net rows, the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *convergence rows,*

the $\mathsf{Real}^{\uparrow}$ *sealing row, and the* $\mathsf{Compact}^{\uparrow}$ *export row. In the notation of Definition 1687.1, the route is*

$$B \longmapsto T \longmapsto U \longmapsto E \longmapsto K \longmapsto H, C, P, N.$$

Thus cover-radius data are available only as a finite-net completion read, not as primitive open-cover compactness, quotient completion, selected limits, or countable choice.

Lean4 ✓ `BEDC.Derived.CompactMetricCompletionUp.CompactMetricCompletionLebesgueNumberConsumerRoute`

Proof. Start with an accepted carrier $M = (T, U, B, K, E, H, C, P, N)$. By Definition 1687.3, every compact-facing read already factors through the finite-net ledger B, the completion route T, the complete-uniform row U, the Real seal E, and the compact export K.

A Lebesgue-number or finite-cover consumer is a compact-facing read with a radius request. The only radius source exposed by the packet is the tolerance-indexed $\mathsf{TotallyBounded}^{\uparrow}$ finite-net ledger B, and the only convergence source that validates the completed metric read is the $\mathsf{CompleteUniformSpace}^{\uparrow}$ row U.

Apply Definition 1687.4. It keeps downstream modulus consumers behind the same B, T, U, E, K route, so a cover-radius read cannot bypass the Real seal or detach itself from the CompactUp export row.

The structural rows H, C, P, N only transport, replay, source, and name this route. Since the carrier has no coordinate for open-cover compactness, quotient completions, selected limits, or countable choice, the finite-net route is the entire public consumer surface. □

Falsifiable prediction: Every $\mathsf{CompactMetricCompletion}^{\uparrow}$ compact read factors through a TotallyBounded finite-net ledger, a TotallyBoundedCompletion route, a CompleteUniformSpace convergence row, a RealUp seal, CompactUp export, transport, replay, provenance, and local naming.

Independence witness: totallyboundedcompletion, completeuniformspace, totallybounded, compact, real: $\mathsf{CompactMetricCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite-net completion handoff, complete uniform-space sealing, compact export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricCompletion}^{\uparrow}$ packet is generated from a TotallyBounded finite-net ledger, a TotallyBoundedCompletion route, a CompleteUniformSpace convergence row, a CompactUp export, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1687.1, Definition 1687.2, and Definition 1687.3 seed the finite NameCert surface for complete totally bounded metric completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient completions, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-net completion handoff, complete uniform-space seal, compact export, transport and replay stability, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.CompactMetric

Chapter 1688

A Concrete Naming Certificate for $\mathsf{MinimalCauchyFilter}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and Kelley on filter bases, the $\mathsf{MinimalCauchyFilter}^{\uparrow}$ packet records the finite boundary at which a Cauchy filterbase is trimmed to the rows actually consumed by regular rational and Real-facing names. It reads the directed Cauchy-filter surface of chapter 333, the regular handoff of chapter 1445, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet is a human-known L10 Real and Completion route object: it names a minimal finite Cauchy-filter basis before regular rational and Real-seal consumers read it, without adding quotient filters, selected limits, countable choice, host equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 1688.1 (MinimalCauchyFilter carrier)**.** *A* $\mathsf{MinimalCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, S, R, T, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase row, including the displayed base packets, refinement rows, Cauchy-modulus row, ledger rows, and provenance boundary.*
2. *S is the minimal support row. For every base packet in F that is read by a downstream regular-rational or Real-facing consumer, S records the retained tail window and its inclusion into F; for every unread base packet, S records that no downstream read cites it.*
3. *R is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *handoff row obtained from the retained support, not from the whole ambient filterbase.*
4. *T is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule listing exactly the retained tail windows requested from S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger on the same retained windows, with endpoints, radii, comparison rows, and refinement bounds visible in the packet.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, S, R, T, D have been displayed.*
7. *H records componentwise* hsame *transport for the source filterbase, minimal support, regular handoff, stream schedule, dyadic ledger, Real seal, and structural rows.*

8. *C records displayed* Cont *replay from source refinements to retained support rows, from retained support rows to stream windows and dyadic tolerances, and from the regular handoff to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MinimalCauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimalCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, R, T, D, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable-choice selectors, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MinimalCauchyFilterUp.TasteGate.MinimalCauchyFilterUpTasteGate single carrier alignment`

Theorem 1688.2 (MinimalCauchyFilter NameCert obligations)**.** *For every accepted* $\mathsf{MinimalCauchyFilter}^{\uparrow}$ *carrier $Q = (F, S, R, T, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinimalCauchyFilter}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the displayed ten-row packet;*
2. *CauchyFilter source exposure through F and its finite filterbase rows;*
3. *minimal support exposure through S, retaining precisely the base packets cited by downstream regular-rational or Real-facing reads;*
4. *StreamName and DyadicRatCore readback through T and D on the retained windows;*
5. *regular handoff through R after the retained support has been displayed;*
6. *Real seal handoff through E after F, S, R, T, D have been read;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No NameCert obligation may be discharged by a quotient filter, selected limit, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.MinimalCauchyFilterUp.MinimalCauchyFilterCarrier namecert obligations`

Proof. Project the carrier of Definition 1688.1. The rows $F, S, R, T, D, E, H, C, P, N$ are the only coordinates available to the local classifier.

The source side is inherited from chapter 333. Its finite base packets, refinement rows, Cauchy-modulus row, ledger rows, and provenance boundary enter the packet as F.

The minimality step is the row S. It records exactly which base packets are read by the downstream regular-rational and Real-facing route, and it records non-use for the remaining displayed base packets without deleting the source CauchyFilter row.

The handoff rows close the obligation surface. The schedule T and tolerance ledger D are read only on retained windows, R is obtained from that retained support, and E is downstream from F, S, R, T, D. Since the excluded principles are absent from the carrier, the seven listed obligations exhaust the local NameCert surface. □

Theorem 1688.3 (MinimalCauchyFilter regular handoff)**.** *Let $Q = (F, S, R, T, D, E, H, C, P, N)$ be an accepted* $\mathsf{MinimalCauchyFilter}^{\uparrow}$ *packet. Every* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *read exported by Q factors through the retained support route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MinimalCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchy}$$

Concretely, the consumer reads F, then the support row S, then the retained stream schedule T and dyadic ledger D, and only then the regular handoff row R with its transport, replay, provenance, and local naming data.

Proof. Start with a $\mathsf{RegularCauchyFilter}^{\uparrow}$ read of Q. By Definition 1688.1, the only regular handoff coordinate is R, and the carrier places R downstream from the source filterbase F, minimal support S, stream schedule T, and dyadic ledger D.

The first part of the route reads chapter 333. The row F supplies finite base packets, refinement rows, Cauchy-modulus rows, ledger rows, and provenance; the row S restricts downstream access to the support actually cited by the regular-rational consumer.

The second part of the route reads chapter 126, chapter 291, and chapter 301. The schedule T and ledger D expose only retained windows, and the regular rational readback is the one used by R.

Finally apply the regular-filter boundary of chapter 1445. The rows H, C, P, N transport, replay, package, and name the same route, so no second regular handoff can be read from an ambient filterbase. □

Lean4 ✓ `BEDC.Derived.MinimalCauchyFilterUp.TasteGate.MinimalCauchyFilterCarrier regular handoff`

Theorem 1688.4 (MinimalCauchyFilter non-escape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MinimalCauchyFilter}^{\uparrow}$ *carrier* $Q = (F, S, R, T, D, E, H, C, P, N)$ *reads only the displayed CauchyFilter source row, retained support row, StreamName schedule, DyadicRatCore tolerance ledger, regular handoff row, Real seal row, componentwise transport, continuation replay, provenance, and local NameCert row. The handoff exports no quotient filter, selected limit, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.MinimalCauchyFilterUp.TasteGate.MinimalCauchyFilterCarrier nonescape`

Proof. Project the downstream rows from Definition 1688.1. The only rows exposed to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumers are $F, S, R, T, D, E, H, C, P, N$.

The regular-rational part is forced by the retained support. The row S decides which base packets of F are visible downstream, while T and D attach stream-window and dyadic-tolerance data only to those retained packets.

The Real-facing part is downstream from the same support. The row E is not an independent completeness assertion; it is reached after the CauchyFilter source, retained support, regular handoff, stream schedule, and dyadic ledger have already been displayed.

The structural rows do not add hidden coordinates. They only transport, replay, package, and name the displayed data. Therefore a quotient filter, selected limit, countable-choice selector, host equality proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle has no carrier row, continuation edge, or provenance slot from which to escape. □

Falsifiable prediction: Every $\mathsf{MinimalCauchyFilter}^{\uparrow}$ regular or Real-facing read factors through a displayed CauchyFilter source row, retained support row, StreamName schedule, DyadicRatCore tolerance ledger, RegularCauchyFilter handoff, Real seal, transport, replay, provenance, and local naming.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, regseqrat, dyadicratcore, real: $\mathsf{MinimalCauchyFilter}^{\uparrow}$ is not bijective to any listed sibling because it carries the retained-support boundary between a finite CauchyFilter source and the regular rational or Real-facing handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalCauchyFilter}^{\uparrow}$ packet is generated from a displayed CauchyFilter finite-filterbase row, a retained support row, a RegularCauchyFilter handoff row, StreamName tail windows, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1688.1, Definition 1688.2, Definition 1688.3, and Definition 1688.4 seed the finite retained-support boundary from Cauchy filterbase data to regular rational and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, CauchyFilter source exposure, retained support exactness, StreamName and DyadicRatCore readback, regular handoff, Real seal handoff, transport and replay stability, and ledger non-escape for BEDC.Derived.MinimalCauchyFilterUp.

Chapter 1689

A Concrete Naming Certificate for $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ packet records the finite route by which a double regular-Cauchy window can be read along a cofinal diagonal schedule without importing a choice principle or a host-level double-limit theorem. It sits on the Real and Completion spine of chapter 291, chapter 126, chapter 301, chapter 38, chapter 1639, and chapter 1561. The packet is a finite-window diagonal extraction surface; it is not quotient-stream equality, a selected host limit, countable choice, Moore–Smith completeness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1689.1 (DoubleCauchyDiagonal carrier)**.** *A* $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, W, D, K, H, C, P, N)$$

with the following displayed rows:

1. *R is the double regular-Cauchy row. It records the two-index regular rational observations, the rowwise and columnwise Cauchy ledgers, and the finite compatibility readings used by diagonal consumers.*
2. *W is the rectangular* $\mathsf{StreamName}^{\uparrow}$ *window row. For each displayed precision request it stores the inspected rectangle of stream names before any diagonal readback is exposed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records dyadic radii, slack comparisons, row and column budgets, and the terminal comparison budget used by the diagonal read.*
4. *K is the cofinal diagonal schedule. It records a finite schedule of diagonal windows whose entries are cofinal for the displayed precision requests and whose selected rectangles remain within the windows named by W.*
5. *H records componentwise* hsame *transport for the double row, rectangular windows, dyadic ledgers, diagonal schedule, continuation replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a Real-facing or Completion-facing request through R, W, D, K, H, P, N and then into the downstream* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *P is the* Pkg *provenance over* $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DoubleCauchyDiagonal}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite double-Cauchy diagonal packet. It has no coordinate for quotient streams, selected limits, countable choice, Moore–Smith completeness, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DoubleCauchyDiagonalUp.TasteGate.DoubleCauchyDiagonalTasteGate single carrier alignment`

Theorem 1689.2 (DoubleCauchyDiagonal NameCert obligations)**.** *For every accepted* $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ *carrier* $G = (R, W, D, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DoubleCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed double regular-Cauchy row, rectangular StreamName windows, dyadic tolerance ledger, cofinal diagonal schedule, componentwise transport, replay, provenance, and local naming rows;*
2. *rectangular window coverage, requiring every diagonal read to inspect its row and column data through* W *before the regular rational observations in* R *are consumed;*
3. *dyadic tolerance accounting, requiring the rowwise, columnwise, and terminal diagonal comparisons to be recorded in* D *before any Real-facing seal is exposed;*
4. *cofinal schedule selection, requiring* K *to provide only finite diagonal windows that are cofinal for the displayed requests and internal to the rectangles already named by* W*;*
5. *transport and replay stability, requiring* H *and* C *to preserve the same route under componentwise sameness and displayed continuation replay;*
6. *ledger non-escape, requiring* P *and* N *to name only the displayed rows and to export no quotient stream, selected limit, countable-choice rule, Moore–Smith completeness theorem, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No DoubleCauchyDiagonal obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DoubleCauchyDiagonalUp.DoubleCauchyDiagonalNamecertObligations`

Proof. Carrier admission is projection from Definition 1689.1. The projected rows are exactly the double regular-Cauchy row R, the rectangular StreamName window row W, the dyadic ledger D, the cofinal diagonal schedule K, and the structural rows H, C, P, N.

Rectangular window coverage reads chapter 126 together with the regular rational readback surface of chapter 301. A diagonal consumer may inspect only the finite row and column windows already displayed by W, and the regular observations in R are read through those windows.

Dyadic tolerance accounting is supplied by chapter 291. The rowwise, columnwise, and terminal diagonal comparisons are ledger rows in D, so a Real-facing consumer cannot obtain a seal before the displayed dyadic budget has been replayed.

The schedule condition is the role of K. Since K stores only finite cofinal windows internal to W, the schedule is a displayed diagonal route rather than a countable-choice function or a selected host subsequence.

Finally, H and C transport and replay the same route, while P and N package and name only the displayed rows. The excluded principles have no carrier coordinate, continuation edge, or ledger row from which to be exported. □

Theorem 1689.3 (DoubleCauchyDiagonal cofinal window selection)**.** *Let $G = (R, W, D, K, H, C, P, N)$ be an accepted* $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ *carrier. Every Real-facing diagonal read exported by G factors through the finite route*

$$W \longmapsto R \longmapsto D \longmapsto K \longmapsto H, C, P, N.$$

Consequently the selected diagonal window is read only after the rectangular $\mathsf{StreamName}^{\uparrow}$ *windows, double regular-Cauchy observations, dyadic tolerance ledger, and cofinal schedule rows have been displayed.*

Lean4 ✓ `BEDC.Derived.DoubleCauchyDiagonalUp.DoubleCauchyDiagonalCofinalWindowSelection`

Proof. Begin with a Real-facing diagonal read of G. By Definition 1689.1, the only window coordinate available to the read is the rectangular row W, and the only regular observation coordinate is the double row R.

Replay the continuation row C. The request first opens W, then reads the regular observations through R, then consults the dyadic ledger D, and only then consumes the cofinal schedule K.

The transport, provenance, and local naming rows preserve that order. Therefore the selected diagonal window is a finite displayed row of the packet, not a host-selected subsequence, quotient stream, or countable-choice object. □

Theorem 1689.4 (DoubleCauchyDiagonal Real seal boundary)**.** *Let $G = (R, W, D, K, H, C, P, N)$ be an accepted* $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ *carrier. Any downstream* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal read attributed to G is bounded by the displayed route through* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *and* $\mathsf{CauchyNetLimit}^{\uparrow}$*:*

$$R, W, D, K \longmapsto \mathsf{BishopRegularCauchyCompletion}^{\uparrow}, \mathsf{CauchyNetLimit}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The boundary exports the diagonal extraction surface only as finite NameCert data. It does not assert quotient streams, selected limits, countable choice, Moore–Smith completeness, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.DoubleCauchyDiagonalUp.DoubleCauchyDiagonalRealSealBoundary`

Proof. The accepted carrier supplies the finite diagonal surface R, W, D, K, H, C, P, N. By the carrier definition, its provenance row names the Bishop regular-Cauchy completion route and the Cauchy-net limit boundary as downstream consumers, not as hidden coordinates.

Read the completion side through chapter 1639. The double regular-Cauchy row, window row, and dyadic ledger provide the finite data consumed by that route; they do not create a quotient stream or selected completion point.

Read the net-limit side through chapter 1561 and then through the terminal $\mathsf{Real}^{\uparrow}$ surface of chapter 38. The seal is therefore downstream of the displayed diagonal route and remains bounded by the same finite NameCert packet.

Since the packet has no coordinate for Moore–Smith completeness, countable choice, selected limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be exported by the seal boundary. □

Falsifiable prediction: Every $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ diagonal read factors through rectangular StreamName windows, double regular-Cauchy observations, a DyadicRatCore tolerance ledger, a cofinal diagonal schedule, transport, replay, provenance, and local naming before any Real-facing seal can be consumed.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishopregularcauchycompletion, cauchynetlimit: $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a double regular-Cauchy row, rectangular windows, dyadic tolerance ledgers, a cofinal diagonal schedule, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DoubleCauchyDiagonal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoubleCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1689.1, Definition 1689.2, Definition 1689.3, and Definition 1689.4 seed the finite double-Cauchy diagonal extraction surface through rectangular windows, dyadic ledgers, cofinal schedules, and downstream Real and Completion seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoubleCauchyDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, Moore–Smith completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rectangular window coverage, dyadic tolerance accounting, cofinal schedule selection, transport and replay stability, ledger non-escape, and Real seal boundary proofs for BEDC.Derived.DoubleCauchyDiagonalUp.

Constructive story: A $\mathsf{DoubleCauchyDiagonal}^{\uparrow}$ packet is generated from a double regular-Cauchy row, rectangular $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a cofinal diagonal schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1690

A Concrete Naming Certificate for $\mathsf{DyadicCeiling}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicCeiling}^{\uparrow}$ packet records the finite upper-grid endpoint needed when a regular-Cauchy or real-facing read is enclosed from above at a displayed dyadic scale. It sits over the dyadic arithmetic surface of chapter 291, the lower-grid dual of chapter 1416, the adjacent endpoint surface of chapter 433, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite certificate for the upper endpoint of a dyadic enclosure, not a host ceiling function, order trichotomy decision, quotient real equality, countable-choice selection, chosen representative, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, or hidden Archimedean search.

Definition 1690.1 (DyadicCeiling carrier)**.** *A* $\mathsf{DyadicCeiling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (x, k, c, p, \ell, u, M, R, E, H, T, P, N)$$

with the following displayed rows:

1. *x is the source real-facing request, read only through the regular rational or terminal real rows carried by the packet.*
2. *k is the requested dyadic scale.*
3. *c is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid point claimed as the upper dyadic ceiling row at scale k.*
4. *p is the predecessor row $c - 2^{-k}$, recorded as a second* $\mathsf{DyadicRatCore}^{\uparrow}$ *row rather than as host integer arithmetic.*
5. *ℓ is the predecessor-lower comparison evidence that the inspected request lies above the predecessor row p at scale k.*
6. *u is the ceiling-upper comparison evidence that the inspected request is bounded above by c at the same scale.*
7. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule supplying the finite observation threshold needed to read both displayed comparisons.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular rational window selected by M.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dyadic rows and regular readback have been displayed.*

10. *H records componentwise* hsame *transport for the request, scale, ceiling row, predecessor row, comparison rows, modulus, readback, seal, continuation, provenance, and local naming rows.*
11. *T records the displayed* Cont *replay from a precision request through $k, c, p, \ell, u, M, R, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{DyadicCeiling}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicFloor}^{\uparrow}$, $\mathsf{DyadicRounding}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCeiling}^{\uparrow}}$ *row.*

The carrier has no coordinate for host ceiling, trichotomy, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden Archimedean search.*

Lean4 ✓ `BEDC.Derived.DyadicCeilingUp.TasteGate.DyadicCeilingTasteGate single carrier alignment`

Theorem 1690.2 (DyadicCeiling NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{DyadicCeiling}^{\uparrow}}$ *surface for Definition 1690.1 consists exactly of carrier exposure for C, upper-grid exposure through c, predecessor exposure through p, predecessor-lower comparison through ℓ, ceiling-upper comparison through u, modulus exposure through M, regular rational readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through T, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DyadicCeilingUp.DyadicCeilingNameCertObligations`

Proof. Project the carrier tuple $C = (x, k, c, p, \ell, u, M, R, E, H, T, P, N)$. The definition displays every analytic and structural coordinate available to the local naming row.

The upper-grid clauses are finite projections. The row c is the claimed ceiling endpoint, p is its predecessor at the same scale, and the comparison rows ℓ and u bind the inspected request between those two dyadic rows.

The remaining coordinates preserve the same packet. The schedule M and readback R supply the finite observation window, the seal E is terminal, and H, T, P, N transport, replay, source, and name the displayed rows. Since the excluded host operations are not coordinates of C, the listed rows exhaust the seed obligation surface. □

Theorem 1690.3 (DyadicCeiling floor-dual route)**.** *Every accepted* $\mathsf{DyadicCeiling}^{\uparrow}$ *packet $C = (x, k, c, p, \ell, u, M, R, E, H, T, P, N)$ exposes the upper-grid dual of the* $\mathsf{DyadicFloor}^{\uparrow}$ *bounding primitive: the predecessor row p and ceiling row c form the finite dyadic interval*

$$p <_k x \leq_k c$$

at the displayed scale k, where ℓ reads the predecessor-lower side and u reads the ceiling-upper side. The route factors through

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and exports no host ceiling, trichotomy branch, quotient real equality, countable-choice row, selected representative, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden Archimedean search.*

Proof. Start from the carrier of Definition 1690.1. The only dyadic endpoint rows available at scale k are the predecessor row p and the ceiling row c.

Read the two comparison rows next. The row ℓ supplies the predecessor-lower evidence, while u supplies the ceiling-upper evidence, so the accepted packet exposes the displayed interval from below by p and from above by c.

Finally compare the route with chapter 1416. The floor packet stores a lower grid row and its successor; the ceiling packet stores an upper grid row and its predecessor. Both routes pass through M, R before the dyadic comparison rows and reach E only after the finite regular-rational window has been displayed. Thus the ceiling packet is the upper-grid dual needed for dyadic interval enclosure, without importing any of the excluded ambient operations. □

Lean4 ✓ `BEDC.Derived.DyadicCeilingUp.DyadicCeilingFloorDualRoute`

Falsifiable prediction: An accepted $\mathsf{DyadicCeiling}^{\uparrow}$ consumer that reads an upper dyadic enclosure without the displayed scale k, ceiling row c, predecessor row p, predecessor-lower evidence ℓ, ceiling-upper evidence u, modulus schedule M, and regular readback row R refutes the carrier surface.

Independence witness: dyadicratcore, dyadicfloor, dyadicrounding, regseqrat, real: no listed sibling is bijective to $\mathsf{DyadicCeiling}^{\uparrow}$ because this packet jointly binds a scale, an upper dyadic grid point, its predecessor, two comparison rows, a modulus schedule, regular readback, terminal seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCeiling}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCeiling}^{\uparrow}$ packet is generated from a real-facing request, a dyadic scale, an upper dyadic grid point, its predecessor, predecessor-lower and ceiling-upper comparison evidence, a finite modulus schedule, a regular rational readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCeiling}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1690.1, Definition 1690.2, and Definition 1690.3 seed the finite upper-grid dyadic enclosure primitive for the L10 Real and regular-Cauchy route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCeiling}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host ceiling, trichotomy, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, hidden Archimedean search, or a Lean theorem.

Upgrade path: A formal route can encode the carrier and prove the NameCert obligation surface and floor-dual enclosure theorem for BEDC.Derived.DyadicCeilingUp.

Chapter 1691

A Concrete Naming Certificate for $\mathsf{FastCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion with explicit moduli, the $\mathsf{FastCauchyCompletion}^{\uparrow}$ packet records the finite route by which a modulus-controlled Cauchy name is exposed to Real and Completion consumers. It sits on the L10 spine supplied by the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Real-facing modulus route of chapter 1423, and the rate-comparison boundary of chapter 1671. The packet is a finite NameCert surface for modulus-controlled completion reads; it is not a quotient stream, a selected limit, trichotomy, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1691.1 (Fast Cauchy completion carrier)**.** *A* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (M_0, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. M_0 *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row. It records the finite precision request, the displayed tail index, and the modulus budget that every downstream completion or Real read must consume.*
2. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the threshold named by* M_0*.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the same window, including dyadic radii, endpoint comparisons, rounding rows, and tolerance arithmetic.*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from* S *and* D *under the budget exposed by* M_0*.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after* M_0, S, D, Q *have been displayed.*
6. H *records componentwise* hsame *transport for the modulus row, stream windows, dyadic ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a finite completion or Real precision request through* M_0, S, D, Q, E, H, P, N*.*
8. P *is the* Pkg *provenance over* $\mathsf{FastCauchyCompletion}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyRateComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M_0, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, trichotomy, countable choice, host equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 1691.2 (Fast Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *carrier $M = (M_0, S, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Cauchy-modulus threshold, stream-window row, dyadic tolerance ledger, regular rational readback, Real seal handoff, and structural rows H, C, P, N;*
2. *modulus discipline, requiring every finite precision request to read the threshold row M_0 before any stream window, dyadic tolerance, regular readback, or Real seal is exposed;*
3. *stream and dyadic readback, requiring S and D to be read under the same displayed modulus budget before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q may be consumed;*
4. *Real-seal handoff, requiring E to be terminal with respect to M_0, S, D, Q and to expose no detached Real object or ambient completeness theorem;*
5. *transport and replay stability, requiring H and C to preserve the same ordered route under componentwise sameness and displayed continuation replay;*
6. *ledger non-escape, requiring P and N to name only the displayed rows and to export no quotient stream, selected limit, trichotomy, countable-choice, host-equality, or ambient-completeness coordinate.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FastCauchyCompletionUp.FastCauchyCompletionUp single carrier alignment`

Proof. Carrier admission is projection from Definition 1691.1. The visible packet consists of the modulus threshold M_0, finite stream window S, dyadic tolerance ledger D, regular readback Q, Real seal E, and structural rows H, C, P, N.

Modulus discipline follows from the placement of M_0 at the front of the route. A completion-facing precision request first fixes the displayed threshold, and every later observation is read inside the finite budget named by that threshold.

The stream and dyadic readback obligation reads chapter 126, chapter 291, and chapter 301 in order. The window S and ledger D are consumed before Q is available, so regular rational data cannot be detached from the displayed modulus budget.

The Real-seal handoff reads chapter 1423, chapter 1671, and chapter 38. The row E is terminal and downstream of M_0, S, D, Q, while H, C, P, N transport, replay, package, and name exactly that route.

Since quotient streams, selected limits, trichotomy, countable choice, host equality, and ambient Real completeness are absent from the carrier, no local NameCert obligation can export them. □

Theorem 1691.3 (Fast Cauchy completion modulus Real seal)**.** *Let $M = (M_0, S, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *carrier. Every Real-facing read exported by M factors through the displayed modulus route*

$$M_0 \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently a Real or Completion consumer receives a Cauchy-modulus threshold, finite stream windows, dyadic tolerance arithmetic, regular rational readback, the terminal Real seal, transport,

replay, provenance, and local naming before it can read the Real-facing datum. The route cannot use quotient streams, selected limits, trichotomy, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchyCompletionUp.FastCauchyCompletionModulusRealSeal`

Proof. Begin with a Real-facing read of M. By Definition 1691.1, the only Real coordinate is E, and the definition places it downstream of the modulus threshold M_0, stream window S, dyadic ledger D, and regular readback row Q.

The first part of the route is fixed by the modulus and finite-window rows. The threshold M_0 supplies the finite tail budget, S exposes only the corresponding stream observations, and D supplies the dyadic tolerances and endpoint comparisons for those observations.

The second part of the route is fixed by regular readback and the Real seal. The row Q is consumed only after S and D, and E seals the Real-facing datum only after the regular row has been displayed.

Finally, H, C, P, N preserve and name the same dependency order. Because the excluded objects have no coordinate in the carrier and no replay edge in the displayed route, every Real-facing read factors through the modulus route shown in the statement. □

Theorem 1691.4 (Fast Cauchy completion finite induction)**.** *Let* $M = (M_0, S, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *carrier. Every accepted precision-tail read of* M *is obtained by finite induction on the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* M_0*: the base read opens the first* $\mathsf{StreamName}^{\uparrow}$ *window* S *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*; the successor step advances only by* Cont *replay through the same* S, D, Q *route; and the terminal read reaches* E *only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal. No induction step uses quotient streams, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchyCompletionUp.FastCauchyCompletion precision tail finite induction`

Proof. Use the carrier of Definition 1691.1 to expose the ordered rows $M_0, S, D, Q, E, H, C, P, N$. The NameCert obligations of Definition 1691.2 make the threshold row the first row read by any completion-facing precision request.

For the base threshold, read the first displayed stream window through S and the dyadic tolerance ledger through D. These rows are already inside the accepted carrier, so the base read does not require a quotient stream, a selected limit, or host equality.

For the successor threshold, replay the previous route through the Cont row C. The replay can advance only the displayed finite window and tolerance rows, and the readback row Q remains downstream of the same S, D data.

The terminal step is the modulus Real-seal route of Definition 1691.3. It reaches E through Q, then preserves the route through H, C, P, N. Since the excluded principles have no carrier coordinate and no replay edge, the displayed induction exhausts the accepted precision-tail reads. □

Falsifiable prediction: Every $\mathsf{FastCauchyCompletion}^{\uparrow}$ Real-facing read factors through a CauchyModulus threshold, StreamName finite windows, DyadicRatCore tolerance arithmetic, RegSeqRat readback, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: real-cauchy-modulus, cauchy-modulus, streamname, dyadicratcore, regseqrat, cauchy-rate-comparison, real: $\mathsf{FastCauchyCompletion}^{\uparrow}$ is not bijective to any listed sibling

because it jointly carries the modulus threshold, stream window, dyadic tolerance ledger, regular readback, terminal Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyCompletion}^{\uparrow}$ packet is generated from a CauchyModulus threshold row, StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1691.1, Definition 1691.2, and Definition 1691.3 seed the finite modulus-controlled completion route from CauchyModulus thresholds to terminal $\mathsf{Real}^{\uparrow}$ reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus discipline, stream and dyadic readback, Real-seal handoff, transport and replay stability, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.FastCauchyCompletionU

Chapter 1692

A Concrete Naming Certificate for $\mathsf{LocatedUniformCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive completion language and Kelley's uniform-space terminology, the $\mathsf{LocatedUniformCompletion}^{\uparrow}$ packet records the finite route by which located Cauchy-filter data, uniform completion data, regular Cauchy readbacks, finite stream windows, and terminal real seals are offered to Real and Completion consumers. It sits between the located filter surface of chapter 1661, the uniform completion carrier of chapter 1473, the Bishop regular Cauchy completion route of chapter 1639, the stream-name observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real boundary of chapter 38. The packet names a located uniform-completion certificate, not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$, a quotient-limit object, countable choice, selected representatives, or host equality.

Definition 1692.1 (Located uniform completion carrier)**.** *A* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, U, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row. It records the located filter basis, explicit neighborhood decision ledger, and Cauchy-filter handoff inspected by the completion-facing request.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row. It stores the finite uniform completion route through which the located filter is admitted as completion data.*
3. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *row. It repacks the located uniform route as a Bishop-style regular Cauchy completion handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the displayed completion name.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows and regular Cauchy handoff.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, U, B, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the located filter, uniform completion, regular Cauchy completion, stream window, regular rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion request through $F, U, B, S, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located uniform-completion packet. It has no coordinate for quotient limits, host real equality, selected representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedUniformCompletionUp.TasteGate.LocatedUniformCompletionTasteGate single carrier alignment`

Theorem 1692.2 (Located uniform completion NameCert obligations)**.** *For every accepted* LocatedUniformComple *carrier $L = (F, U, B, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed located Cauchy filter, uniform completion route, Bishop regular Cauchy completion row, stream window, regular rational readback, terminal Real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *located filter exposure, requiring every completion-facing request to read the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row F before it reaches the uniform completion row U;*
3. *uniform completion routing, requiring U to be the only completion route between the located filter data and the Bishop regular Cauchy completion row B;*
4. *finite readback, requiring every Real-facing read to inspect the* $\mathsf{StreamName}^{\uparrow}$ *row S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the terminal seal E is exposed;*
5. *transport and replay stability, requiring H and C to preserve the same ordered route under componentwise sameness and displayed continuation replay;*
6. *ledger non-escape, requiring P and N to name only the displayed packet and to export no quotient limit, host equality, selected representative, countable-choice schedule, or ambient completeness coordinate.*

No located uniform-completion obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedUniformCompletionUp.LocatedUniformCompletionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1692.1. The projected rows are exactly the located Cauchy filter row F, uniform completion row U, Bishop regular Cauchy completion row B, stream window S, regular rational readback R, terminal Real seal E, and structural rows H, C, P, N.

Located filter exposure reads chapter 1661 before any completion-facing coordinate is available. Uniform completion routing then reads chapter 1473, so the route from F to the completion surface is carried by U rather than by an unrecorded quotient or host-level limit.

The Bishop regular Cauchy completion handoff is supplied by chapter 1639. It repacks the located uniform route into the regular Cauchy completion row B before any Real-facing read can continue.

Finite readback uses chapter 126 and chapter 301. The terminal boundary is then read through chapter 38 as the row E.

The rows H and C preserve this order under transport and replay. The provenance and naming rows P, N name only the displayed packet, so the excluded quotient, choice, selected-limit, host-equality, and completeness data have no coordinate from which to enter. □

Theorem 1692.3 (Located uniform completion density route)**.** *Every completion-facing read of an accepted* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *carrier* $L = (F, U, B, S, R, E, H, C, P, N)$ *factors through the finite route*

$$F \longmapsto U \longmapsto B \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the dense located source is exposed only through a located Cauchy filter row, a uniform completion row, a Bishop regular Cauchy completion row, finite stream windows, regular rational readback, the terminal Real seal, transport, replay, provenance, and local naming.

Lean4 ✓ `BEDC.Derived.LocatedUniformCompletionUp.LocatedUniformCompletionCarrier density route`

Proof. Begin with a completion-facing read of L. The carrier definition gives no direct completion coordinate except the displayed route $F, U, B, S, R, E, H, C, P, N$.

The located source enters through F, and the uniform completion interface is reached through U. This is the density-facing part of the certificate: the source is visible by located filter decisions and by the finite uniform completion route, not by a selected dense image or quotient-limit representative.

The regular Cauchy handoff then passes through B, and the finite readback passes through S and R. The terminal boundary row E is downstream of those rows, while H, C, P, N transport, replay, package, and name the same route.

Therefore every completion-facing read factors through the displayed finite route and receives no host equality, countable-choice schedule, selected limit, quotient limit, or ambient completeness theorem. □

Theorem 1692.4 (Located uniform completion Real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *boundary read exported by an accepted* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *carrier* $L = (F, U, B, S, R, E, H, C, P, N)$ *is downstream of the located uniform completion route*

$$\mathsf{LocatedCauchyFilter}^{\uparrow}(F), \quad \mathsf{UniformCompletion}^{\uparrow}(U), \quad \mathsf{BishopRegularCauchyCompletion}^{\uparrow}(B), \quad \mathsf{StreamName}^{\uparrow}(S),$$

No $\mathsf{Real}^{\uparrow}$ *consumer receives a quotient limit, host real equality, selected representative, countable-choice schedule, or ambient completeness datum from this packet.*

Lean4 ✓ `BEDC.Derived.LocatedUniformCompletionUp.LocatedUniformCompletionRealBoundary`

Proof. Project the terminal boundary row from Definition 1692.1. The row E is present only after F, U, B, S, R have been displayed.

The preceding route is fixed by the six cited boundary chapters: located filters supply F, uniform completions supply U, Bishop regular Cauchy completions supply B, stream names supply S, regular rational names supply R, and the Real chapter supplies the terminal seal discipline for E.

Since the carrier contains no quotient-limit, host-equality, selected-representative, choice, or ambient-completeness coordinate, the $\mathsf{Real}^{\uparrow}$ consumer can inspect only the displayed route and the structural rows H, C, P, N. The claimed boundary factorization follows. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1692.1, Definition 1692.2, Definition 1692.3, and Definition 1692.4 seed the located uniform-completion route through located Cauchy filters, uniform completion, Bishop regular Cauchy completion, stream windows, regular rational readback, and the Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, quotient limits, countable choice, host equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located filter exposure, uniform completion routing, finite readback, transport and replay stability, ledger non-escape, density routing, and Real-boundary factorization for BEDC.Derived.LocatedUniformCompletionUp.

Constructive story: A LocatedUniformCompletion$^{\uparrow}$ packet is generated from a located Cauchy filter row, uniform completion route, Bishop regular Cauchy completion row, finite StreamName window, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Chapter 1693

A Concrete Naming Certificate for $\mathsf{FanBarRoute}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on finite fans and barred spreads, the $\mathsf{FanBarRoute}^{\uparrow}$ packet packages the finite route from $\mathsf{FanTheorem}^{\uparrow}$ bar-depth rows and $\mathsf{BrouwerBarInduction}^{\uparrow}$ barred-spread replay into the consumer surface used by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It sits after the finite fan certificate of chapter 400, the barred-spread induction certificate of chapter 1675, the stream-window surface of chapter 126, the regular-rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers one finite fan/bar handoff route; it is not open-cover compactness, sequential compactness, countable choice, selected infinite branches, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1693.1 (FanBarRoute carrier)**.** *A* $\mathsf{FanBarRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, I, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite fan-tree row. It records the displayed binary prefix tree, prefix closure, finite successor rows, and the source fan surface consumed from* $\mathsf{FanTheorem}^{\uparrow}$*.*
2. *B is the decidable bar-depth ledger. It records the displayed barred prefixes, Bool decisions, unary depth bound, and exact branch-prefix coverage rows delivered by the finite fan theorem.*
3. *I is the barred-spread induction replay row. It records the local backwards replay consumed from* $\mathsf{BrouwerBarInduction}^{\uparrow}$*, using only displayed children and barred prefixes from T and B.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It is the only route by which stream-facing consumers may request finite observations of the fan/bar packet.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It turns the displayed stream windows into regular rational observations before a Real or Completion consumer can inspect the handoff.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic precision, radius, comparison, and refinement ledger attached to the readback route.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, B, I, W, Q, D and the structural rows have been displayed.*

8. *H records componentwise* hsame *transport for the fan tree, bar-depth ledger, induction replay, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from finite fan inspection through bar-depth exactness, barred-spread replay, stream windows, regular-rational readback, dyadic tolerance, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{FanBarRoute}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FanBarRoute}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, B, I, W, Q, D, E, H, C, P, N$. It has no coordinate for a selected infinite branch, open-cover compactness, sequential compactness, quotient stream equality, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FanBarRouteUp.TasteGate.FanBarRouteTasteGate single carrier alignment`

Theorem 1693.2 (FanBarRoute NameCert obligations)**.** *For every accepted* $\mathsf{FanBarRoute}^{\uparrow}$ *carrier $F = (T, B, I, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FanBarRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite fan tree, decidable bar-depth ledger, induction replay, StreamName window, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, continuation, provenance, and local naming rows;*

2. *finite bar-depth exactness, requiring every branch-prefix request to meet the displayed bar through T and B by the named unary depth;*

3. *barred-spread replay, requiring every backwards induction step to pass through I and the displayed children of T rather than through a selected infinite branch;*

4. *stream and readback compatibility, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window read to pass through W and every regular-rational read to pass through Q before any dyadic or Real-facing seal is visible;*

5. *Real seal nonescape, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row E to be downstream of T, B, I, W, Q, D and to export no ambient completeness or host real equality;*

6. *transport and provenance stability, requiring H, C, P, N to preserve the same route under componentwise sameness, displayed continuation replay, package provenance, and local naming.*

No local obligation is stored outside these six rows.

Lean4 ✓ `BEDC.Derived.FanBarRouteUp.FanBarRouteNameCertObligations`

Proof. Carrier admission is projection from Definition 1693.1. The projected packet is exactly the tuple $T, B, I, W, Q, D, E, H, C, P, N$.

Finite bar-depth exactness reads the $\mathsf{FanTheorem}^{\uparrow}$ surface. The fan tree T supplies the displayed prefix tree, and B supplies the Bool bar decisions together with the unary depth ledger, so the bar claim is checked over visible prefixes only.

Barred-spread replay reads the $\mathsf{BrouwerBarInduction}^{\uparrow}$ surface. The row I contains the backwards local replay, and its children are the displayed successors already present in T, so no branch selector is introduced.

The analytic handoff uses the ordered rows W, Q, D, E. A stream request first opens the finite $\mathsf{StreamName}^{\uparrow}$ window W, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Finally, H, C, P, N transport, replay, name, and record the same route. Since the carrier contains no further coordinate, the excluded compactness, choice, quotient, branch-selection, host-equality, and ambient-completeness principles cannot be exported. □

Theorem 1693.3 (FanBarRoute finite bar handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{FanBarRoute}^{\uparrow}$ *carrier factors through*

$$T, B, I, W, Q, D, E, H, C, P, N.$$

The consumer receives a finite fan tree, a decidable bar-depth ledger, a barred-spread replay row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, the Real seal, transport, continuation, provenance, and local naming. It receives no selected infinite branch, open-cover compactness, sequential compactness, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FanBarRouteUp.FanBarRouteFiniteBarHandoff`

Proof. Start with a Real or Completion-facing read of the carrier. By Definition 1693.1, the only analytic-facing rows are the stream window W, the regular-rational readback Q, the dyadic tolerance row D, and the Real seal E.

Replay the continuation row C. Before any analytic-facing row is visible, the route exposes the finite fan tree T, the decidable bar-depth ledger B, and the barred-spread induction replay I.

After those rows have been displayed, the consumer can read W, then Q, then D, and then E. Componentwise transport and package provenance are fixed by H and P, and the local naming row is N.

Thus every consumer-visible route factors through the displayed tuple. The carrier has no row for branch selection, open-cover compactness, sequential compactness, quotient stream equality, countable choice, or ambient real completeness, so none of those principles can be extracted from the handoff. □

Falsifiable prediction: Every $\mathsf{FanBarRoute}^{\uparrow}$ Real or Completion-facing read factors through a finite fan tree, decidable bar-depth ledger, barred-spread replay, StreamName window, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, continuation, provenance, and local naming.

Independence witness: fantheorem, brouwer-bar-induction, streamname, regseqrat, dyadicratcore, real: $\mathsf{FanBarRoute}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite fan tree, bar-depth ledger, barred-spread replay, analytic handoff rows, transport, continuation, provenance, and local naming while refusing branch selection, compactness, choice, quotient equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{FanBarRoute}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FanBarRoute}^{\uparrow}$ packet is generated from a finite fan tree, a decidable bar-depth ledger, barred-spread replay, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanBarRoute}^{\uparrow}}$, seed)

Scope closed: Definition 1693.1, Definition 1693.2, and Definition 1693.3 seed the finite fan/bar handoff surface for Real and Completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanBarRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, open-cover compactness, sequential compactness, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar-depth exactness, barred-spread replay, stream/readback compatibility, Real seal nonescape, transport and provenance stability, and TasteGate field faithfulness for BEDC.Derived.FanBarRouteUp.

Chapter 1694

A Concrete Naming Certificate for $\mathsf{UniformCauchyFilter}^{\uparrow}$

Following Bourbaki uniform spaces and Bishop–Bridges constructive completion, the $\mathsf{UniformCauchyFilter}^{\uparrow}$ packet records the current-state uniform Cauchy filter route as finite BEDC data. It sits between the entourage interface of chapter 339, the finite base interface of chapter 1601, the directed-window interface of chapter 589, and the completion boundary of chapter 1473. Its rational readback is routed through chapter 126, chapter 301, and chapter 660.

Definition 1694.1 (Uniform Cauchy filter carrier)**.** *A* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(U, F, C, T, S, R, E, H, K, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage ledger whose finite entourages, diagonal rows, symmetry rows, and composition refinements are available to the packet;*
2. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *base row naming the displayed filter-base indices, base elements, directed lower-bound witnesses, and refinement ledger;*
3. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row used when a downstream route asks to read the same Cauchy evidence as finite net windows;*
4. *T is the entourage-Cauchy stability row: for each displayed entourage request in U, it names a base row in F whose paired elements are read inside that entourage;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window readback row attached to the accepted base windows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the same stream windows, with the regularity budget displayed as part of the packet;*
7. *E is the completion handoff row for* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumers, exposing only the rows U, F, C, T, S, R and their structural support;*
8. *H is the componentwise* hsame *transport row over U, F, C, T, S, R, E;*
9. *K is the displayed* Cont *replay route for base requests, entourage requests, stream windows, regular rational reads, and completion handoffs;*

10. *P is the* Pkg *provenance over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchySequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, choice-selected subnet limits, host completeness, open-cover compactness, ultrafilters, or Zorn-style maximality.

Theorem 1694.2 (Uniform Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *carrier* $(U, F, C, T, S, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed uniform-space, filter-base, Cauchy-net, stability, stream-window, regular rational, completion, transport, replay, provenance, and naming rows;*
2. *filter-base compatibility, requiring every filter request to read the displayed* $\mathsf{FilterBase}^{\uparrow}$ *row F and its directed lower-bound and refinement ledger before any Cauchy claim is exported;*
3. *entourage-Cauchy stability, requiring each displayed entourage request from U to be witnessed by the base row selected in T and replayed through the same refinement ledger;*
4. *stream-window readback, requiring finite observations to pass through the* $\mathsf{StreamName}^{\uparrow}$ *row S rather than through an external sequence object;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring regular rational data to be read from R on the same finite windows and budgets already displayed by S;*
6. *uniform-completion handoff, requiring* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumers to read only the displayed rows U, F, C, T, S, R, E and the structural rows H, K, P, N;*
7. hsame *transport, requiring componentwise transport through H for all displayed rows;*
8. Cont *replay, requiring the replay route K to preserve the base request, entourage request, finite window, rational readback, and completion handoff order;*
9. Pkg *provenance, requiring P to name exactly the sibling packets listed in the carrier and no external completeness or compactness principle;*
10. *local naming, requiring N to expose the certificate only as* $\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}$.

Proof. Carrier admission is projection from Definition 1694.1. The rows $U, F, C, T, S, R, E, H, K, P, N$ are the visible coordinates of one finite BHist packet, so the local certificate has no detached filter, subnet, limit, or completeness coordinate.

Filter-base compatibility starts with F. A downstream request must read the base indices, base elements, directed lower-bound witnesses, and refinement ledger before it can inspect the Cauchy row T.

Entourage-Cauchy stability then reads U and T together. The entourage request comes from the displayed uniform-space ledger U, while T names the base row in F whose paired observations are read inside that entourage.

The stream and rational readbacks are the next two fixed faces. The row S supplies the finite $\mathsf{StreamName}^{\uparrow}$ windows, and R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational data on those same windows and budgets.

Finally, completion-facing consumers reach E only after the preceding rows have been displayed. Componentwise transport through H, replay through K, package provenance through P, and local

naming through N preserve that route and do not introduce quotient filters, selected subnet limits, host completeness, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.UniformCauchyFilterUp.UniformCauchyFilterTasteGate single carrier alignment`

Theorem 1694.3 (Uniform Cauchy filter completion handoff)**.** *Let* $(U, F, C, T, S, R, E, H, K, P, N)$ *be an accepted* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *carrier. Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of this route may read only the displayed filter-base row* F*, the Cauchy-net row* C*, the stream-window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the completion row* E*, and the structural rows* H, K, P, N*. It cannot read a quotient filter, a choice-selected subnet limit, host completeness, open-cover compactness, an ultrafilter, or a maximal-filter principle from the packet.*

Proof. Project the accepted carrier. The completion-facing coordinate is E, but its admissible inputs are already fixed by the carrier: the filter-base row F, the directed-window row C, the stream-window row S, and the regular rational row R.

The entourage stability row T mediates between the uniform-space ledger U and the base row F. Thus any completion request that asks for Cauchy evidence must pass through the displayed filter-base and Cauchy-net faces before reading E.

The remaining rows are structural. Transport by H, replay by K, provenance by P, and local naming by N expose the same finite handoff route and no further coordinate. Since quotient filters, selected subnet limits, host completeness, open-cover compactness, ultrafilters, and maximal-filter principles are not coordinates of the carrier, no $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer can obtain them from this certificate. □

Lean4 ✓ `BEDC.Derived.UniformCauchyFilterUp.UniformCauchyFilterCarrier completion handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyFilter}^{\uparrow}$ packet is generated from displayed uniform-space entourage rows, filter-base rows, Cauchy-net windows, stream windows, RegSeqRat readback, a completion handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1694.1, Definition 1694.2, and Definition 1694.3 seed the finite uniform Cauchy filter route from uniform-space entourages and filter bases to StreamName, RegSeqRat, and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, choice-selected subnet limits, host completeness, open-cover compactness, ultrafilters, maximal-filter principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base compatibility, entourage-Cauchy stability, stream-window readback, RegSeqRat readback, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyFilterUp.

Chapter 1695

A Concrete Naming Certificate for $\mathsf{RegularCauchyAddition}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyAddition}^{\uparrow}$ packet names addition of two regular rational Cauchy real names by finite synchronized-window arithmetic before a completed $\mathsf{Real}^{\uparrow}$ consumer is allowed to read the sum. It sits over the regular rational source rows of chapter 301, the stream-window discipline of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later real-algebra and completion consumers a licensed BEDC route for addition; it does not import host real arithmetic, quotient Cauchy classes, selected limits, trichotomy, or ambient completeness.

Definition 1695.1 (RegularCauchyAddition carrier)**.** *A* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_+ = (R_0, R_1, W_0, W_1, T_0, T_1, D, S, E, Z, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *input rows.*
2. W_0 *and* W_1 *are paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedules, read at a common precision request and observation index.*
3. T_0 *and* T_1 *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows assigned to the two source observations.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *summed-endpoint row, recording the dyadic readback of the pointwise sum exposed by the two windows.*
5. S *is the summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose regularity is supplied by the two input regularity rows, the paired window schedules, and the dyadic tolerance rows.*
6. E *is the finite error-budget ledger combining the two source tolerances, the dyadic addition tolerance, and the output tolerance needed by the summed readback row.*
7. Z *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the two source rows, paired windows, dyadic tolerance rows, summed readback row, and finite ledger have been displayed.*
8. H *records componentwise* hsame *transport for the two source rows, paired stream windows, tolerance rows, summed endpoint row, summed readback, Real seal, replay row, provenance row, and local naming row.*
9. C *records displayed* Cont *replay from every addition-facing precision request through* $R_0, R_1, W_0, W_1, T_0, T_1, D, E,$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, W_0, W_1, T_0, T_1, D, S, E, Z, H, C, P, N$. The carrier has no coordinate for host real addition, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached completed-real addition operation.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAdditionUp.RegularCauchyAdditionUp single carrier alignment`

Theorem 1695.2 (Regular Cauchy addition NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *carrier*

$$A_+ = (R_0, R_1, W_0, W_1, T_0, T_1, D, S, E, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *input rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance rows, summed endpoint row, finite error ledger, summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural rows;*

2. *paired-window exposure, requiring each addition-facing precision request to inspect W_0 and W_1 at the same finite precision before any summed readback is available;*

3. *dyadic endpoint addition, requiring D to be read from the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows T_0, T_1 and the two window observations;*

4. *finite regularity accounting, requiring E to combine the two input tolerance rows, the dyadic addition tolerance, and the output tolerance before S is consumed;*

5. *Real seal boundary, requiring Z to be read only after the route through $R_0, R_1, W_0, W_1, T_0, T_1, D, E, S$ has been displayed;*

6. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite addition packet;*

7. *refusal of host real addition, quotient Cauchy classes, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No addition obligation is stored outside these finite rows.

Proof. Project the accepted carrier. The analytic rows are the two regular rational names R_0, R_1, the paired stream schedules W_0, W_1, the dyadic tolerance rows T_0, T_1, the summed endpoint row D, the finite ledger E, the summed readback S, and the Real seal Z.

Read the windows before the sum by finite-window induction on the displayed precision schedule. The base window exposes the first paired observations, and the successor window preserves the same synchronized request before any pointwise dyadic arithmetic is used.

Form the dyadic endpoint and the regular readback next. The row D consumes the two displayed observations and their tolerance rows through the $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic route, while E records the finite tolerance accounting that turns the pointwise sum into the $\mathsf{RegSeqRat}^{\uparrow}$ readback row S.

Finally seal and package the route. The row Z is downstream of the summed readback, and the structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores host real addition, quotient classes, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data, none can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.RegularCauchyAdditionUp.RegularCauchyAdditionCarrier namecert obligations`

Theorem 1695.3 (Regular Cauchy addition window-sum stability)**.** *Let two accepted* RegularCauchyAddition$^{\uparrow}$ *carriers have input rows, paired windows, dyadic tolerance rows, summed endpoint rows, finite ledgers, readbacks, Real seals, package provenance, and local naming rows matched componentwise by their displayed* hsame *classifiers. Then every summed-window route supplied by the obligation surface of Definition 1695.2 transports to the corresponding route of the second packet.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAdditionUp.RegularCauchyAdditionCarrier window sum stability`

Proof. Start with the componentwise classifier data in H. A finite-window induction on the paired schedules transports the two RegSeqRat$^{\uparrow}$ source rows and the StreamName$^{\uparrow}$ windows without changing the common precision request.

Transport the dyadic addition row. The matched tolerance rows T_0, T_1 still feed the same DyadicRatCore$^{\uparrow}$ summed endpoint route D, so the pointwise addition step remains tied to the displayed finite windows.

The ledger and readback rows are preserved by the same componentwise transport. The row E remains the only place where source tolerances, dyadic addition tolerance, and output tolerance are combined before S and Z are exposed.

Replay through C and repack through P and N. Every consumer route in the obligation surface is one of these displayed projections, so classifier transport cannot add host real addition, quotient Cauchy classes, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data. □

Theorem 1695.4 (Regular Cauchy addition uniform modulus)**.** *If two accepted* RegularCauchyAddition$^{\uparrow}$ *packets share the same* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, and* DyadicRatCore$^{\uparrow}$ *addition tolerance ledger through their displayed* hsame *rows, then their* Real$^{\uparrow}$ *sum seal is controlled by the same displayed modulus row. The control is the finite route through* $R_0, R_1, W_0, W_1, T_0, T_1, D, E, S, Z, H, C, P, N$*, not host real addition, quotient Cauchy classes, selected limits, trichotomy, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAdditionUp.RegularCauchyAddition uniform modulus`

Proof. Project the carrier of Definition 1695.1. The two packets expose the same source readbacks, paired finite windows, dyadic tolerance rows, summed endpoint row, and finite ledger row through componentwise transport.

Use Definition 1695.3. It binds the finite windows and dyadic addition route, so the displayed ledger E remains the common place where the two source budgets, the addition tolerance, and the output tolerance are combined.

Apply Definition 1695.5. The terminal row Z is read only after the common ledger route has fixed S, so both sum seals are controlled by the same displayed modulus row and no host real addition, quotient class, selected limit, trichotomy, or ambient Real completeness is read. □

Theorem 1695.5 (Regular Cauchy addition Real seal handoff)**.** *Every accepted* RegularCauchyAddition$^{\uparrow}$ *carrier factors its Real-facing addition read through the finite route*

$$(R_0, R_1) \longmapsto (W_0, W_1) \longmapsto (T_0, T_1, D) \longmapsto E \longmapsto S \longmapsto Z$$

before it may hand the result to a Real$^{\uparrow}$ *consumer. Thus the consumer receives two regular rational source rows, paired finite schedules, dyadic tolerance and summed endpoint rows, a finite addition ledger, a summed* RegSeqRat$^{\uparrow}$ *readback row, and a Real seal row; it receives no host real addition, quotient Cauchy class, selected limit, trichotomy, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAdditionUp.RegularCauchyAdditionCarrier real seal handoff`

Proof. Begin with an accepted carrier A_+. The only source coordinates for the addition read are R_0 and R_1, and finite-window induction over the attached schedules W_0, W_1 supplies the paired observations in dependency order.

The pointwise sum then passes through the dyadic rows. The rows T_0 and T_1 attach the source tolerances to the displayed observations, while D records the finite dyadic addition readback.

The finite ledger finishes the regular handoff. The row E combines the two source budgets with the dyadic addition and output budgets before the summed $\mathsf{RegSeqRat}^{\uparrow}$ readback row S can be consumed.

Only then is the $\mathsf{Real}^{\uparrow}$ seal row Z exposed. The rows H, C, P, N preserve and name this same dependency order, so the Real-facing consumer receives a finite BEDC addition route rather than host arithmetic, quotient classes, selected limits, trichotomy, or ambient completeness. □

Theorem 1695.6 (Regular Cauchy addition sibling dependency route)**.** *Every downstream Real algebra or completion consumer of an accepted* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *packet factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *summed-endpoint row, the finite addition ledger, the summed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}$ *row. The route is exactly the dependency package certified by the NameCert obligations, window-sum stability, and Real seal handoff; it excludes host real addition, quotient Cauchy classes, selected limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligation surface of Definition 1695.2. It lists the two regular rational source rows, paired stream windows, dyadic tolerance rows, summed endpoint row, finite ledger, summed readback, Real seal, and structural rows as the complete carrier-facing surface.

Use Definition 1695.3 to keep the same rows under componentwise classifier transport. No downstream consumer can replace the paired windows or dyadic addition rows while preserving the accepted packet.

Finally apply the Real seal handoff of Definition 1695.5. The seal is exposed only after the displayed addition route, so Real algebra and completion consumers inherit precisely this sibling dependency route and none of the excluded host or completion objects. □

Theorem 1695.7 (Regular Cauchy addition window-fusion boundary)**.** *Let*

$$A_+ = (R_0, R_1, W_0, W_1, T_0, T_1, D, S, E, Z, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyAddition}^{\uparrow}$ *packet whose two source windows are fused through the public finite route of* $\mathsf{RegularCauchyWindowFusion}^{\uparrow}$*. Before any* $\mathsf{Real}^{\uparrow}$ *algebra consumer reads the sum seal* Z*, the fused addition packet exports the same finite L10 route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. The fusion handoff changes only the shared terminal-window presentation; it does not add host real addition, quotient Cauchy classes, selected limits, trichotomy, ambient completeness, or a second sum seal.*

Proof. Start with Definition 1695.2. It identifies the two source rows, paired windows, dyadic tolerance rows, summed endpoint row, finite ledger, summed readback, Real seal, and structural rows as the complete addition-facing packet.

Apply Definition 1695.5. The sum seal Z can be read only after the route through $R_0, R_1, W_0, W_1, T_0, T_1, D, E, S$ has been displayed, so a downstream Real algebra consumer cannot bypass the finite addition rows.

Finally use Definition 693.19. The fusion packet exposes the terminal window through the same finite L10 route over regular readback, stream window, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows. Substituting that public route for the two displayed source windows preserves the addition handoff and contributes no host arithmetic or ambient completion data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1695.1, Definition 1695.2, Definition 1695.3, and Definition 1695.5 seed the finite addition surface over two $\mathsf{RegSeqRat}^{\uparrow}$ inputs, paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and summed endpoint rows, summed $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAddition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real addition, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-window exposure, dyadic endpoint addition, finite regularity accounting, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyAddition

Constructive story: A $\mathsf{RegularCauchyAddition}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ rows, paired $\mathsf{StreamName}^{\uparrow}$ finite-window schedules, two $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ summed endpoint row, a finite error ledger, a summed $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1696

A Concrete Naming Certificate for $\mathsf{RegularCauchySubtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySubtraction}^{\uparrow}$ packet records binary subtraction of displayed regular rational Cauchy real names as a finite sign-flip-plus-sum route. It sits over the regular rational source rows of chapter 301, the synchronized window discipline of chapter 126, the dyadic tolerance core of chapter 291, the addition route of chapter 1695, the sign-flip route of chapter 459, the algebra packet of chapter 1470, and the terminal real-name seal of chapter 38. The packet is the binary subtraction surface consumed by later L10 real and completion algebra routes; it is not host real subtraction, a quotient Cauchy-class operation, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1696.1 (RegularCauchySubtraction carrier)**.** *A* $\mathsf{RegularCauchySubtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B_{-} = (X, Y, G, A, W, D, E, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for regular rational Cauchy real names.*
2. *G is the sign-flip route for Y through the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *carrier, exposing only the displayed source row, finite windows, dyadic sign-flip ledger, transported classifier, and* $\mathsf{Real}^{\uparrow}$ *seal handoff of that packet.*
3. *A is the sum route for $X + (-Y)$ through the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *carrier, using X as the first summand and the regular readback exported by G as the second summand.*
4. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window ledger aligning the source reads of X, the source reads of Y, the sign-flipped reads in G, and the paired summand reads consumed by A.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the source tolerance rows, the sign-flipped dyadic endpoint rows, and the dyadic addition rows used by the final subtraction readback.*
6. *E is the finite error-budget accounting row combining the tolerance consumed by the sign flip, the paired summand tolerance, and the output tolerance required by the subtraction readback.*
7. *S is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed binary subtraction route.*
8. *H records componentwise* hsame *transport for the two source rows, the sign-flip route, the sum route, synchronized windows, dyadic ledger, finite budget row, readback, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from any subtraction-facing precision request through* $X, Y, G, A, W, D, E, S, H, P$
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubtraction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, G, A, W, D, E, S, H, C, P, N$. The carrier has no coordinate for host real subtraction, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubtractionUp.RegularCauchySubtractionTasteGate single carrier alignment`

Theorem 1696.2 (Regular Cauchy subtraction NameCert obligations)**.** *For every accepted* RegularCauchySubtracti *carrier*

$$B_{-} = (X, Y, G, A, W, D, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *sign-flip route, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *sum route, synchronized windows, dyadic tolerance rows, finite error-budget row, resulting readback, and structural rows;*
2. *negation handoff, requiring G to expose the sign-flipped row of Y only through the finite sign-flip ledger of chapter 459;*
3. *addition handoff, requiring A to consume X and the readback exported by G only through the finite addition route of chapter 1695;*
4. *finite error-budget accounting, requiring E to combine the sign-flip tolerance, paired summand tolerance, dyadic addition tolerance, and output tolerance before S is available;*
5. *Real seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to read the route through X, Y, G, A, W, D, E, S before any terminal seal is exposed;*
6. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite subtraction packet;*
7. *refusal of host real subtraction, quotient Cauchy classes, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No subtraction obligation is stored outside these finite rows.

Theorem 1696.3 (Regular Cauchy subtraction Real seal handoff)**.** *Every Real-facing subtraction read exported by an accepted* $\mathsf{RegularCauchySubtraction}^{\uparrow}$ *carrier factors through the finite route*

$$(X, Y) \longmapsto G \longmapsto A \longmapsto W \longmapsto D \longmapsto E \longmapsto S \longmapsto \mathsf{Real}^{\uparrow}$$

before it may be consumed as a completed real-name boundary. Thus the consumer receives two source rows, the sign-flip row for Y, the sum row for $X + (-Y)$, synchronized $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. It receives no host real subtraction, quotient Cauchy class, selected limit, trichotomy, ambient completeness coordinate, Lean verification, or bridge checking.*

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubtraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1696.1, Definition 1696.2, and Definition 1696.3 seed the finite binary subtraction route through $\mathsf{RegSeqRat}^{\uparrow}$ source rows, $\mathsf{RegularCauchyNegation}^{\uparrow}$ sign flip, $\mathsf{RegularCauchyAddition}^{\uparrow}$ sum, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, resulting $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real subtraction, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, negation handoff, addition handoff, finite error-budget accounting, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubt

Constructive story: A $\mathsf{RegularCauchySubtraction}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a $\mathsf{RegularCauchyNegation}^{\uparrow}$ sign-flip route for the second source, a $\mathsf{RegularCauchyAddition}^{\uparrow}$ sum route for the first source and the sign-flipped readback, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, a finite error-budget row, a resulting $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1697

A Concrete Naming Certificate for $\mathsf{BoundedRegularCauchy}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet records the bounded regular Cauchy surface needed before Real and Completion consumers ask for completeness, compactness, or limit extraction. It realizes the Real and Completion route of chapter 9238 as explicit finite packet data: the dyadic endpoint and tolerance ledger is read from chapter 291, finite observations are read through chapter 126, regular rational readback is supplied by chapter 301, the terminal real-name boundary is sealed by chapter 38, and the tail schedule is exposed through chapter 292. The packet is a boundedness-and-tail witness for regular Cauchy names; it is not a host completeness theorem, a quotient stream representative, a selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real equality.

Definition 1697.1 (BoundedRegularCauchy carrier)**.** *A* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window family, listing the finite observation windows that may be opened before any Real or Completion consumer inspects a tail claim.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail schedule, assigning each displayed dyadic tolerance request a finite threshold and the corresponding late-window budget.*
3. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *boundedness ledger, containing the dyadic center, radius, endpoint comparisons, and tail-bound inequalities used to keep all late reads inside the displayed bound.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows, the modulus schedule, and the dyadic boundedness ledger.*
5. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, M, B, Q have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, modulus schedule, dyadic bound, regular readback, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from any bounded-regular-Cauchy request through M, then through S and B, into Q, and finally into the* $\mathsf{Real}^{\uparrow}$ *seal row A.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedRegularCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, M, B, Q, A, H, C, P, N$. The carrier has no coordinate for host completeness, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real equality.

Lean4 ✓ `BEDC.Derived.BoundedRegularCauchyUp.BoundedRegularCauchyUpTasteGate single carrier alignment`

Theorem 1697.2 (BoundedRegularCauchy NameCert obligation surface)**.** *For every accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window family, tail schedule, dyadic boundedness ledger, regular rational readback, Real seal row, componentwise transport, continuation replay, provenance, and local naming;*
2. *tail-bound exactness, requiring every requested dyadic tolerance to read its threshold from M and its late-window dyadic inequalities from B before the readback row Q is exposed;*
3. *regular-readback compatibility, requiring Q to be obtained only from the displayed S, M, B rows and to remain inside the bound recorded by B;*
4. *classifier transport, requiring H and C to replay the same displayed route M, S, B, Q, A under componentwise* hsame*;*
5. *Real-seal handoff, requiring any* $\mathsf{Real}^{\uparrow}$ *consumer to read the terminal seal row A only after stream windows, modulus data, dyadic tail bounds, and regular readback have all been displayed;*
6. *ledger non-escape, requiring P and N to export no host completeness, quotient stream equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real equality.*

No bounded-regular-Cauchy obligation is stored outside these rows.

Proof. Carrier admission is row projection from Definition 1697.1. The displayed packet has exactly the stream-window, modulus, dyadic-bound, readback, seal, transport, replay, provenance, and local naming rows listed there.

Tail-bound exactness is the joint content of M and B. A tolerance request first fixes a threshold in the modulus schedule, and the dyadic ledger then records the late-window inequalities that keep the corresponding observations inside the displayed bound.

Regular-readback compatibility follows by replaying the sibling surfaces in order. The $\mathsf{StreamName}^{\uparrow}$ windows expose finite observations, the $\mathsf{CauchyModulus}^{\uparrow}$ row chooses the late threshold, the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the bounded dyadic comparisons, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the resulting regular rational readback.

Classifier transport is componentwise. The hsame row H transports each displayed coordinate, while C repeats the same M, S, B, Q, A route for any accepted request. Since no excluded coordinate is present in the carrier, the transport cannot introduce one.

Finally, the Real-seal handoff and non-escape rows are exhausted by A, P, N. A Real consumer reaches A only through the displayed route, and the provenance and naming rows record that the packet exports no host completeness, quotient stream equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real equality. □

Theorem 1697.3 (BoundedRegularCauchy Real-seal boundary)**.** *Every Real-facing or Completion-facing read exported by an accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier factors through the finite route*

$$M \longmapsto S \longmapsto B \longmapsto Q \longmapsto A$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Thus the consumer receives explicit dyadic tail-bound data, finite stream windows, a modulus threshold, regular rational readback, and the displayed Real seal, and receives no host completeness theorem, quotient stream representative, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, or ambient Real equality.*

Proof. Begin with an accepted packet $R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N)$ and project its displayed rows. The only threshold source is M, so every Real-facing or Completion-facing request must first read a finite tail budget from the modulus row.

The finite observation and boundedness rows are downstream of that budget. The stream-window row S supplies the observations that may be inspected at the selected threshold, and the dyadic ledger B records the explicit bound and late-window inequalities attached to those observations.

The regular rational readback and Real seal are last. The row Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from S, M, B, and the row A is the terminal $\mathsf{Real}^{\uparrow}$ seal exposed only after that readback has been displayed.

The structural rows preserve this route without enlarging it. Componentwise transport in H, replay in C, provenance in P, and local naming in N add no coordinate for host completeness, quotient stream representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real equality. □

Theorem 1697.4 (BoundedRegularCauchy tail-window induction)**.** *For every accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N)$$

and every unary tolerance-depth request read through M, induction on the finite threshold chain displayed by M shows that each late window is first read through S, then bounded through B, then repacked through the same $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, and only after those stages reaches the terminal* $\mathsf{Real}^{\uparrow}$ *row A. No endpoint, selected limit, quotient stream, countable choice, trichotomy, open-cover compactness, or ambient Real equality can enter the induction.*

Lean4 ✓ `BEDC.Derived.BoundedRegularCauchyUp.BoundedRegularCauchyCarrier tail window induction`

Proof. The base stage projects the first displayed threshold from M. At that stage the carrier exposes only the finite stream-window row S attached to the threshold, the dyadic bound row B attached to the same late window, and the regular readback row Q obtained from those displayed rows.

For the induction step, assume the route has been established for the preceding displayed thresholds. The next threshold is supplied by the same modulus row M; the continuation row C replays the route through the next S-window and the next B-bound, while componentwise transport in H carries the displayed rows without changing the classifier surface.

The repacking stage preserves the $\mathsf{RegSeqRat}^{\uparrow}$ handoff because Q has no coordinate beyond the stream windows, modulus schedule, and dyadic boundedness ledger. Thus each successor window reaches A only after the finite route through S, B, Q has been displayed.

The conclusion follows from Definition 1697.3 together with ledger non-escape in P and N. Since those rows export no endpoint choice, selected limit, quotient stream, countable choice, trichotomy, open-cover compactness, or ambient Real equality, the induction remains confined to the displayed threshold windows. □

Theorem 1697.5 (BoundedRegularCauchy modulus-chain induction)**.** *For every accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N),$$

induction on the finite modulus chain displayed by M shows that each tolerance request first reads its threshold from M, then reads the dyadic boundedness ledger B, then repacks through the $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, and only after those rows reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal A. The induction exports no quotient stream, selected limit, countable choice, trichotomy, open-cover compactness, or ambient Real equality.*

Proof. The base stage is the first displayed tolerance request. By Definition 1697.1, the request has no threshold coordinate except the modulus row M, and no bound coordinate except the dyadic ledger B.

For the induction step, assume all earlier displayed requests have followed the route through M, B, Q. The successor request is still a row of the same finite chain M, so the continuation replay in C can only expose the corresponding B-entry before the readback row Q is consumed.

The terminal stage is fixed by Definition 1697.3. That theorem places A after the displayed modulus, boundedness, and regular-readback rows, so the induction cannot reach the Real seal before M, B, Q have appeared.

Finally the structural rows H, P, N add transport, provenance, and local naming without adding excluded coordinates. Thus the whole finite modulus-chain induction remains confined to the displayed carrier rows. □

Theorem 1697.6 (BoundedRegularCauchy tail-modulus extraction)**.** *For every accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N),$$

each displayed dyadic tolerance request exports a finite tail-modulus threshold row by the route

$$M \longrightarrow B \longrightarrow Q \longrightarrow A.$$

The request first reads its threshold from the modulus schedule M, validates the late-window dyadic bound in B, repacks the bounded late observations through the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal A only after those rows have been displayed. The exported threshold row is therefore a reusable Real and Completion handoff, not a quotient stream representative, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, or ambient Real equality.*

Proof. Carrier projection from Definition 1697.1 gives the only available rows for a tolerance request: the stream windows S, modulus schedule M, dyadic bound B, regular readback Q, and terminal Real seal A.

The Real-seal boundary of Definition 1697.3 orders any Real-facing or Completion-facing read through the displayed finite route before A can be consumed. Hence no terminal seal can appear before the modulus, boundedness, and readback rows have been exposed.

Apply the tail-window induction of Definition 1697.4 to keep each late window attached to its finite threshold in M and its displayed observations in S. Then apply Definition 1697.5 to read the threshold chain through M, B, Q, A.

The extracted threshold row is the finite row produced by that chain. The structural rows H, C, P, N transport, replay, source, and name the same route, so the handoff cannot introduce a quotient representative, selected limit, choice schedule, trichotomy principle, open-cover compactness proof, or ambient Real equality. □

Theorem 1697.7 (BoundedRegularCauchy two-stage tail envelope)**.** *For every accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *carrier*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N),$$

there is a reusable two-stage tail envelope exported by the displayed rows. In the first stage, M, S, B *produce the finite bounded late-window envelope: the modulus schedule chooses the late threshold, the stream-window row supplies the finite observations at that threshold, and the dyadic boundedness ledger records the corresponding endpoint and radius inequalities. In the second stage,* Q, A *repack that finite envelope as a* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *seal handoff. The envelope is therefore available to Real and Completion consumers as finite BEDC data, not as a quotient stream, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, or ambient Real equality.*

Proof. Start with the accepted packet and project the route of Definition 1697.3. Every Real-facing read reaches A only after the displayed finite path through the modulus, stream-window, boundedness, and regular-readback rows.

The first stage is finite induction over late windows. By Definition 1697.4, each requested late window stays attached to its threshold in M, its finite observations in S, and its dyadic bound in B. By Definition 1697.5, the same finite route is stable as the tolerance request moves along the displayed modulus chain.

The second stage is the readback and seal handoff. Apply Definition 1697.6: the bounded late-window data are repacked through Q and then exposed through the terminal $\mathsf{Real}^{\uparrow}$ seal A. The structural rows H, C, P, N transport, replay, source, and name the same packet, so this handoff remains reusable without adding any excluded host principle. □

Theorem 1697.8 (BoundedRegularCauchy tail-window dependency)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet*

$$R_{\mathrm{bd}} = (S, M, B, Q, A, H, C, P, N)$$

must expose the dependency rows in the order

$$M \longrightarrow S \longrightarrow B \longrightarrow Q \longrightarrow A.$$

Here M *is the modulus threshold row,* S *is the finite stream-window row,* B *is the dyadic boundedness ledger,* Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus no Real or Completion use of the packet may bypass the finite tail-window route by appealing to hidden completeness, quotient-stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real equality.*

Proof. The Real-seal boundary in Definition 1697.3 fixes the outer route: a consumer can reach A only after the displayed finite path has passed through the modulus, stream-window, boundedness, and readback rows.

The late-window content of that route is forced by Definition 1697.4. Once a tolerance request has read its threshold from M, every inspected late window is opened through S, bounded through B, and repacked through Q before the seal row can be consumed.

The threshold chain itself is forced by Definition 1697.5. It keeps every tolerance request attached to the displayed M-chain and prevents the readback from skipping directly to the terminal seal.

Finally, Definition 1697.7 packages the same data as the reusable finite envelope M, S, B followed by the handoff Q, A. Since the structural rows only transport, replay, source, and name that envelope, the consumer receives the ordered rows M, S, B, Q, A and no excluded host principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchy}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegularCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1697.1, Definition 1697.2, Definition 1697.3, Definition 1697.4, Definition 1697.7, and Definition 1697.8 seed the finite bounded regular Cauchy packet over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegularCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-bound exactness, regular-readback compatibility, classifier transport, Real-seal handoff, provenance, local naming, and ledger non-escape for BEDC.Derived.BoundedRegularCauchyUp.

Constructive story: A $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{CauchyModulus}^{\uparrow}$ threshold schedule, a $\mathsf{DyadicRatCore}^{\uparrow}$ boundedness ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1698

A Concrete Naming Certificate for BoundedRegularCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the BoundedRegularCauchySubsequence$^{\uparrow}$ packet records the bounded tail-subsequence interface needed on the L10 Real and Completion route before any consumer asks for compactness, completeness, or a chosen limit. It refines the bounded tail ledger of chapter 1697 through the monotone-cofinal surface of chapter 381, the criterion route of chapter 1552, the inspected fast route of chapter 1518, the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38.

The packet is a finite NameCert handoff. It stores bounded tail data, a displayed monotone-cofinal schedule, finite observation windows, dyadic tolerance rows, RegSeqRat$^{\uparrow}$ readback, RegularCauchySubsequenc handoff, and a Real$^{\uparrow}$ seal; it stores no selected infinite subsequence, quotient stream, countable choice, host equality, or Real completeness principle.

Definition 1698.1 (BoundedRegularCauchySubsequence carrier)**.** *A* BoundedRegularCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, T, W, D, Q, R, F, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* BoundedRegularCauchy$^{\uparrow}$ *row, including the bounded tail schedule and its finite dyadic endpoint ledger;*
2. *T is a displayed monotone-cofinal unary schedule over the inspected tail indices;*
3. *W records the source and selected* StreamName$^{\uparrow}$ *windows admitted along T;*
4. *D records the* DyadicRatCore$^{\uparrow}$ *tolerance rows, tail bounds, and late-window comparisons attached to those windows;*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback determined by B, T, W, D;*
6. *R is the* RegularCauchySubsequence$^{\uparrow}$ *handoff row built from the same schedule, windows, tolerance rows, and readback;*
7. *F is the* FastCauchySubsequence$^{\uparrow}$ *or* CauchySubsequenceCriterion$^{\uparrow}$ *criterion witness read only from the displayed bounded subsequence route;*
8. *E is the terminal* Real$^{\uparrow}$ *seal exposed after Q, R, F have been displayed;*

9. *H records componentwise* hsame *transport for all displayed analytic and support rows;*
10. *C records* Cont *replay through the route B, T, W, D, Q, R, F, E;*
11. *P is the* Pkg *provenance over* BoundedRegularCauchySubsequence$^{\uparrow}$, BoundedRegularCauchy$^{\uparrow}$, RegularCauchySubsequence$^{\uparrow}$, CauchySubsequenceCriterion$^{\uparrow}$, FastCauchySubsequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{BoundedRegularCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, T, W, D, Q, R, F, E, H, C, P, N$. The carrier has no coordinate for a selected infinite subsequence, quotient stream, countable choice, host equality, or Real completeness.

↪variant BEDC.Derived.BoundedRegularCauchySubsequenceUp.BoundedRegularCauchySubsequenceTasteGate single carrier alignment

Theorem 1698.2 (BoundedRegularCauchySubsequence selection factorization)**.** *For every accepted* BoundedRegularCauchySubsequence$^{\uparrow}$ *carrier*

$$L = (B, T, W, D, Q, R, F, E, H, C, P, N),$$

each bounded regular Cauchy subsequence read factors through bounded tail data, the monotone-cofinal schedule, finite windows, dyadic tolerance rows, RegSeqRat$^{\uparrow}$ *readback,* RegularCauchySubsequence$^{\uparrow}$ *handoff, the criterion witness, and the* Real$^{\uparrow}$ *seal:*

$$B, T, W, D, Q, R, F, E.$$

Thus the consumer receives a finite BEDC route from the bounded regular Cauchy source to the real-facing seal, and receives no selected infinite subsequence, quotient stream, countable choice, host equality, or Real completeness.

Lean4 ✓ BEDC.Derived.BoundedRegularCauchySubsequenceUp.BoundedRegularCauchySubsequenceTasteGate single carrier alignment

Proof. Project the accepted carrier of Definition 1698.1. Its only source row for boundedness is B, and its only source row for subsequence movement is the displayed monotone-cofinal schedule T.

The finite windows and tolerance rows are downstream of those two coordinates. The row W records exactly the source and selected StreamName$^{\uparrow}$ windows inspected along T, while D records the DyadicRatCore$^{\uparrow}$ bounds and dyadic tolerance comparisons attached to the same windows.

The analytic handoff rows consume that displayed route. The RegSeqRat$^{\uparrow}$ readback Q is determined by B, T, W, D, the RegularCauchySubsequence$^{\uparrow}$ row R repacks the same schedule and window data, and the witness row F records the fast or criterion-side acceptance of the displayed subsequence.

Finally, the seal and support rows cannot enlarge the interface. The Real$^{\uparrow}$ row E is reached only after Q, R, F, and the rows H, C, P, N transport, replay, source, and name the same displayed coordinates. Since the classifier contains no remaining coordinate, the excluded subsequence, quotient, choice, host-equality, and completeness data cannot be read. □

Theorem 1698.3 (Bounded regular Cauchy subsequence tail seal)**.** *Every accepted* BoundedRegularCauchySubseque *read exports exactly the rows*

$$B, T, W, D, Q, R, F, E, H, C, P, N.$$

These rows are, respectively, the bounded tail data, the monotone-cofinal schedule, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff, the criterion witness, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}}$ *row. No consumer of the packet can read a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the accepted read.*

Proof. Project the carrier of Definition 1698.1. Its displayed coordinates are precisely $B, T, W, D, Q, R, F, E, H, C, P$ and the classifier compares accepted packets only through those rows.

Apply Definition 1698.2. The displayed route identifies the bounded tail data, monotone-cofinal schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, criterion witness, and terminal $\mathsf{Real}^{\uparrow}$ seal as the exported analytic surface.

The remaining rows only transport, replay, source, and name that same surface. Since the carrier has no coordinate outside $B, T, W, D, Q, R, F, E, H, C, P, N$, there is no exported place from which a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle can be recovered. □

Theorem 1698.4 (Regular Cauchy subsequence modulus readback)**.** *For every accepted* $\mathsf{BoundedRegularCauchySub}$ *carrier*

$$L = (B, T, W, D, Q, R, F, E, H, C, P, N),$$

the complete modulus-readback route is exactly the bounded tail data B*, monotone-cofinal schedule* T*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff* R*, criterion witness* F*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and structural rows* H, C, P, N*. No completion-facing consumer can read a selected subsequence, quotient stream equality, countable choice, host real equality, or ambient Real-completeness principle outside this route.*

Proof. Project the carrier of Definition 1698.1. Its analytic rows are precisely the bounded tail data, schedule, windows, tolerances, regular readback, subsequence handoff, criterion witness, and Real seal.

Apply Definition 1698.2. It factors each subsequence read through B, T, W, D, Q, R, F, E, so the modulus information is already tied to the displayed bounded tail, monotone-cofinal schedule, finite windows, and dyadic tolerances.

Apply Definition 1698.3. The exported rows are exactly $B, T, W, D, Q, R, F, E, H, C, P, N$, and the support rows only transport, replay, source, and name that surface. Therefore the displayed route is the complete modulus readback for completion-facing consumers. □

Theorem 1698.5 (Bounded regular Cauchy subsequence Cauchy-hub forward route)**.** *Every bounded regular Cauchy subsequence modulus read is a downstream consumer of the Cauchy hub route in chapter 3413. The forward route passes through bounded tail data, a monotone-cofinal schedule, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchySubsequenceCriteri}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the local* $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}}$ *row is exported.*

Proof. Project the carrier of Definition 1698.1. Its analytic rows are B, T, W, D, Q, R, F, E, and these are the only rows that can feed the modulus readback.

Use the existing route blocks in this chapter. The selection factorization of Definition 1698.2 ties the read to bounded tail data, monotone-cofinal schedule, windows, tolerances, regular readback,

criterion witness, and Real seal. The tail seal of Definition 1698.3 adds only H, C, P, N, and Definition 1698.4 identifies that same list as the complete modulus-readback route.

The upstream analytic source is the Cauchy hub. The route named in chapter 3413 supplies the Cauchy-facing family consumed here before the bounded tail, subsequence criterion, regular rational readback, and Real seal are displayed. Therefore every accepted modulus read in this chapter is a downstream Cauchy-hub consumer rather than an independent selected-subsequence or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1698.1, Definition 1698.2, and Definition 1698.3 seed the finite bounded regular Cauchy subsequence route over displayed bounded tail data, monotone-cofinal schedule, finite windows, dyadic tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, criterion witness, and $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient streams, countable choice, host equality, Real completeness, Lean verification, or standard bridge checking.

Upgrade path: Formal closure requires checked carrier admission, bounded-tail exactness, monotone-cofinal schedule admission, finite-window transport, dyadic tolerance replay, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, criterion witness routing, and Real-seal non-escape for BEDC.Derived.BoundedRegularCauchySubsequenceUp.

Constructive story: A $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a displayed monotone-cofinal schedule, finite StreamName windows, dyadic tolerance rows, RegSeqRat readback, RegularCauchySubsequence handoff, a criterion witness, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1699

A Concrete Naming Certificate for $\mathsf{UniformCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformCauchySubsequence}^{\uparrow}$ packet records the common dyadic tail-budget subsequence interface needed on the L10 Real and Completion route. It refines the bounded subsequence route of chapter 1698 by requiring every real-facing subsequence read to pass through one shared $\mathsf{DyadicRatCore}^{\uparrow}$ budget row before the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal are visible.

The packet is finite. It stores a bounded regular Cauchy subsequence source, displayed tail indices, a shared uniform dyadic budget row, finite observation windows, local dyadic comparisons, regular rational readback, seal routing, transport, replay, provenance, and a local certificate; it stores no selected infinite subsequence, quotient stream, countable choice, host equality, or Real completeness principle.

Definition 1699.1 (UniformCauchySubsequence carrier)**.** *A* $\mathsf{UniformCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, T, U, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *row, including the bounded tail ledger, displayed monotone-cofinal schedule, and finite subsequence route already admitted by its local* $\mathsf{NameCert}$*;*
2. *T is the exposed tail-index schedule read from B, with no independent choice coordinate;*
3. *U is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *uniform tail-budget row, assigning one dyadic tolerance budget to all late windows that feed the subsequence readback;*
4. *W records the* $\mathsf{StreamName}^{\uparrow}$ *source and subsequence windows inspected under T and budgeted by U;*
5. *D records the local dyadic comparisons, endpoint rows, and late-window inequalities read from U, W;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback determined by B, T, U, W, D;*
7. *R records the real-facing regular subsequence handoff obtained from the same rows;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after Q and R have been displayed;*

9. *H records componentwise* hsame *transport for all displayed analytic and support rows;*
10. *C records* Cont *replay through the route B, T, U, W, D, Q, R, E;*
11. *P is the* Pkg *provenance over* UniformCauchySubsequence$^{\uparrow}$, BoundedRegularCauchySubsequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{UniformCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, T, U, W, D, Q, R, E, H, C, P, N$. In particular, U is a named coordinate of the carrier, not a family reconstructed separately for each consumer read.

Lean4 ✓ `BEDC.Derived.UniformCauchySubsequenceUp.UniformCauchySubsequenceTasteGate single carrier alignment`

Theorem 1699.2 (Uniform tail-budget factorization)**.** *For every accepted* UniformCauchySubsequence$^{\uparrow}$ *carrier*

$$L = (B, T, U, W, D, Q, R, E, H, C, P, N),$$

every Real$^{\uparrow}$*-facing subsequence read factors through*

$$B, T, U, W, D, Q, R, E$$

before any seal coordinate is visible. Thus the common dyadic tail-budget row U is read before the finite StreamName$^{\uparrow}$ *windows, the* RegSeqRat$^{\uparrow}$ *readback, the regular subsequence handoff, and the terminal* Real$^{\uparrow}$ *seal.*

Proof. Project the accepted carrier of Definition 1699.1. Its source of subsequence data is B, and its exposed tail schedule is T; neither row contains a separate selected infinite subsequence.

The next mandatory row is U. By definition U is the shared DyadicRatCore$^{\uparrow}$ tail-budget coordinate, and the window row W is admitted only as the collection of StreamName$^{\uparrow}$ windows inspected under T and budgeted by U.

The local dyadic comparison row D is downstream of U, W. The readback row Q is then determined by B, T, U, W, D, and the handoff row R repacks the same displayed route for the real-facing subsequence read.

Finally, the seal row E is exposed only after Q and R. The remaining rows H, C, P, N transport, replay, source, and certify the displayed coordinates without adding a second route around U, so every visible seal read has the stated factorization. □

Theorem 1699.3 (UniformCauchySubsequence NameCert obligations)**.** *The local* NameCert$_{\text{UniformCauchySubsequence}^{\uparrow}}$ *row is generated by exactly the following five obligation rows:*

1. *carrier admission for the finite packet $B, T, U, W, D, Q, R, E, H, C, P, N$;*
2. *common tail-budget stability, asserting that* hsame *transport preserves the shared* DyadicRatCore$^{\uparrow}$ *row U and every late-window comparison read from it;*
3. *window and readback transport, carrying the* StreamName$^{\uparrow}$ *windows through T, U, D into the* RegSeqRat$^{\uparrow}$ *row Q and the regular subsequence handoff R;*
4. Real$^{\uparrow}$*-seal non-escape, asserting that no consumer reaches E without passing through B, T, U, W, D, Q, R;*
5. *provenance exactness for the displayed* Pkg *sources* UniformCauchySubsequence$^{\uparrow}$, BoundedRegularCauchySubseque StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$.

No additional row may name a selected infinite subsequence, quotient stream, countable choice witness, host-equality proof, or Real completeness principle.

Proof. Read the local certificate row N through the classifier of Definition 1699.1. The admissible coordinates of the carrier are precisely $B, T, U, W, D, Q, R, E, H, C, P, N$, so carrier admission is the first row and every later obligation must be a statement about these coordinates.

The common-budget row is forced by U. Since all window data in W is budgeted by U, componentwise hsame transport can preserve the packet only by preserving U and the dyadic comparisons recorded in D.

The transport and readback rows follow the displayed dependency order. Windows in W are read along T, comparisons in D are read from U, W, the rational readback Q is determined by B, T, U, W, D, and R is the same route repacked for the subsequence handoff.

The non-escape and provenance rows are the remaining classifier checks. The seal E has no incoming coordinate except Q, R, while P lists exactly the named sources of the packet. Therefore the five rows listed above generate the local certificate, and no excluded object has a coordinate from which a NameCert obligation could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchySubsequence}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformCauchySubsequence}^{\uparrow}}$, seed)

Scope closed: Definition 1699.1, Definition 1699.2, and Definition 1699.3 seed the finite common tail-budget route from bounded regular Cauchy subsequence data to $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and a $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformCauchySubsequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient streams, countable choice, host equality, Real completeness, Lean verification, or standard bridge checking.

Upgrade path: Formal closure requires checked carrier admission, common tail-budget stability, window and readback transport, Real-seal non-escape, and provenance exactness for BEDC.Derived.UniformCauchySubsequenceUp.

Constructive story: A $\mathsf{UniformCauchySubsequence}^{\uparrow}$ packet is generated from bounded regular Cauchy subsequence data, an exposed tail schedule, one shared DyadicRatCore budget row, StreamName windows, dyadic comparison rows, RegSeqRat readback, regular subsequence handoff, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1700

A Concrete Naming Certificate for $\mathsf{AbsoluteConvergence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{AbsoluteConvergence}^{\uparrow}$ packet records absolute domination of a real-valued series by a finite majorant route rather than by an ambient summability predicate. It sits over the real-series surface of chapter 1581, the fast tail budget of chapter 367, the Cauchy-majorant schedule of chapter 1774, the uniform Cauchy route of chapter 1736, the regular readback of chapter 301, the finite stream observations of chapter 126, the dyadic tolerance ledger of chapter 291, and the endpoint seal of chapter 38. The packet gives Weierstrass-style and real-power-series consumers one finite certificate for absolute domination, majorant-tail control, and $\mathsf{Real}^{\uparrow}$ nonescape without importing host infinite sums or ambient real completeness.

Definition 1700.1 (Absolute convergence carrier)**.** *An* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, M, B, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the real-valued term-stream row, read only through the finite* $\mathsf{RealSeries}^{\uparrow}$ *observations selected by the packet.*
2. *M is the majorant-stream row, recording the nonnegative comparison terms, displayed majorant partial sums, and the tail index attached to each tolerance request.*
3. *B is the absolute-domination ledger: each selected term observation from T is paired with its displayed majorant observation from M and the dyadic budget witnessing the absolute comparison.*
4. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *route obtained from the displayed domination ledger and the Cauchy-majorant budget.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected term, majorant, partial-sum, and tail-window observations opened by the packet.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after T, M, B, U, R have been displayed.*
7. *H stores componentwise* hsame *transport for the term stream, majorant stream, domination ledger, uniform Cauchy route, readback, Real seal, continuation, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from term reads to absolute comparisons, from comparisons to the majorant budget, from the budget to the uniform Cauchy route, from that route to regular readback, and from readback to the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{AbsoluteConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, B, U, R, E, H, C, P, N$. It has no coordinate for selected sums, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1700.2 (Absolute convergence NameCert obligations)**.** *For every accepted* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *carrier*

$$A = (T, M, B, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbsoluteConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the term stream, majorant stream, absolute-domination ledger, uniform Cauchy route, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *term and majorant window exactness, requiring T and M to expose only the selected finite observations and displayed partial-sum ledgers;*

3. *absolute-domination exactness, requiring B to pair each selected term read with its majorant read and its* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budget;*

4. *Cauchy-majorant handoff, requiring U to be read only after the domination ledger and majorant-tail budget have been displayed;*

5. *regular readback, requiring R to expose only the selected* $\mathsf{StreamName}^{\uparrow}$ *windows opened by the term, majorant, domination, and uniform-Cauchy rows;*

6. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to be terminal with respect to T, M, B, U, R and to export no selected sum, quotient stream equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge data;*

7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No absolute-convergence obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1700.1. The accepted packet displays exactly the term stream T, majorant stream M, domination ledger B, uniform Cauchy route U, regular readback R, Real seal E, and structural rows H, C, P, N.

The term and majorant rows provide the only series observations. The domination row B then records the absolute comparison between each selected term observation and its displayed majorant observation, using the dyadic budget already visible to the packet.

The Cauchy-facing rows are downstream of that ledger. The row U consumes the domination ledger and the majorant-tail budget before any uniform Cauchy conclusion is available, and R reads back only the finite windows opened by this route.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since selected sums, quotient stream equality, countable choice, trichotomy, ambient completeness, Lean

verification, and bridge checking do not occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.AbsoluteConvergenceUp.AbsoluteConvergenceTasteGate single carrier alignment`

Theorem 1700.3 (Absolute convergence Cauchy-majorant handoff)**.** *Every Cauchy-facing read from an accepted* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *carrier factors through the route*

$$T \longmapsto M \longmapsto B \longmapsto U \longmapsto R.$$

Consequently the read must display the term stream, majorant stream, absolute-domination ledger, Cauchy-majorant budget, uniform Cauchy threshold, selected $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before it can serve a Completion or Real consumer.*

Proof. Start with a Cauchy-facing request. The carrier first exposes the term stream T and the majorant stream M, so the request has only selected finite observations of the series and the comparison stream.

The row B is then the unique absolute-domination ledger. It pairs the selected term reads with their displayed majorant reads and the dyadic comparison budget, so the request cannot bypass the majorant route.

The handoff to Cauchy control occurs through U and R. The row U records the uniform Cauchy route supported by the Cauchy-majorant budget, while R supplies the regular rational readback for the finite windows opened along that route.

Thus every Completion or Real consumer sees the displayed domination ledger, majorant budget, uniform threshold, selected stream windows, and regular readback before any endpoint-facing seal is available. □

Theorem 1700.4 (Absolute convergence Real seal nonescape)**.** *Every Real-facing read from an accepted* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *carrier reaches the terminal seal only through*

$$T \longmapsto M \longmapsto B \longmapsto U \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The read therefore exports no selected sum, quotient stream equality, countable-choice selector, trichotomy principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Use Definition 1700.3 to place every Real-facing request behind the term stream, majorant stream, absolute-domination ledger, uniform Cauchy route, and regular readback.

The terminal row E is a $\mathsf{Real}^{\uparrow}$ seal for this already displayed route. It is not a selected infinite sum, a quotient equality of streams, or a completeness theorem for arbitrary real names.

The structural rows H, C, P, N preserve and replay the same route componentwise. Because the carrier has no coordinate for the excluded principles, none of them can escape through the Real-facing boundary. □

Falsifiable prediction: Every $\mathsf{AbsoluteConvergence}^{\uparrow}$ Real-facing read factors through a real-valued term stream, a majorant stream, an absolute-domination ledger, a Cauchy-majorant budget, a uniform Cauchy threshold, selected StreamName windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, fastconvergentseries, cauchy-majorant-series, uniform-cauchy-series, regseqrat, streamname, dyadicratcore, real: $\mathsf{AbsoluteConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries term-stream access, a majorant stream, an absolute-domination

ledger, Cauchy-majorant handoff, uniform Cauchy thresholding, regular readback, Real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsoluteConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1700.1, Definition 1700.2, Definition 1700.3, and Definition 1700.4 seed the finite absolute-domination route from real-valued series data through majorant-tail control, uniform Cauchy thresholding, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsoluteConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, term and majorant window exactness, absolute-domination exactness, Cauchy-majorant handoff, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteConvergenceUp.

Constructive story: An $\mathsf{AbsoluteConvergence}^{\uparrow}$ packet is generated from a real-valued term stream, a majorant stream, an absolute-domination ledger, a uniform Cauchy route, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1701

A Concrete Naming Certificate for $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ packet records the finite route by which displayed convergence data for a real sequence supplies Cauchy readiness. It sits between the displayed-limit surface of chapter 1413, the Cauchy-consumer boundary of chapter 1590, the oscillation route of chapter 448, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is an induction-style L10 handoff from a displayed limit row to a late-pair Cauchy-test row; it is not a Real completeness theorem and it does not choose limits for arbitrary streams.

Definition 1701.1 (Convergent real sequence Cauchy carrier)**.** *A* $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (L, W, Q, D, T, E, H, R, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row, containing the displayed source sequence, the displayed limit* $\mathsf{Real}^{\uparrow}$ *row, the convergence tail schedule, and the local convergence classifier.*
2. *W stores the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected from the sequence and from the displayed limit row.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the sequence windows, limit windows, and late-pair rational observations opened by W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the split tolerance, triangle budget, and comparison entries used to turn two convergence reads into one late-pair Cauchy test.*
5. *T is the late-pair Cauchy-test ledger, pairing the two sufficiently late sequence indices with the shared limit row and the dyadic split budget.*
6. *E is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row, exposed only after L, W, Q, D, T have been displayed.*
7. *H records componentwise* hsame *transport for the limit row, stream windows, rational readback, dyadic ledger, late-pair test, Cauchy-consumer row, replay row, provenance row, and local naming row.*

8. *R records displayed* Cont *replay from a convergence request through the two late reads, the common limit, the dyadic split ledger, and the Cauchy-consumer row.*
9. *P is the* Pkg *provenance over* ConvergentRealSequenceCauchy$^{\uparrow}$, RealSequenceLimit$^{\uparrow}$, RealCauchyCriterion$^{\uparrow}$, CauchyOscillation$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{ConvergentRealSequenceCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, W, Q, D, T, E, H, R, P, N$. It has no coordinate for Real completeness, quotient streams, selected limits, trichotomy, countable choice, host sequence equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ConvergentRealSequenceCauchyUp.ConvergentRealSequenceCauchyTasteGate single carrier alignment`

Theorem 1701.2 (Convergent real sequence Cauchy NameCert obligations)**.** *For every accepted* ConvergentRealSequenceCauchy$^{\uparrow}$ *carrier*

$$C = (L, W, Q, D, T, E, H, R, P, N),$$

the local NameCert$_{\text{ConvergentRealSequenceCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* RealSequenceLimit$^{\uparrow}$ *row, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *ledger, late-pair Cauchy-test ledger,* RealCauchyCriterion$^{\uparrow}$ *consumer row, transport, replay, provenance, and local naming rows;*
2. *limit-row exactness, requiring L to be the only source of the displayed sequence, displayed limit, and convergence tail schedule;*
3. *window and readback alignment, requiring W and Q to expose only the sequence, limit, and late-pair observations selected by L and T;*
4. *dyadic split exactness, requiring D to store the tolerance split and triangle comparison used by the late-pair test;*
5. *late-pair Cauchy-test exactness, requiring T to use two convergence reads from L against the same displayed limit row before any Cauchy consumer is exposed;*
6. *criterion handoff, requiring E to be read only after L, W, Q, D, T have been replayed;*
7. *componentwise transport, displayed replay, provenance, and local naming through H, R, P, N;*
8. *refusal of Real completeness, quotient streams, selected limits, trichotomy, countable choice, host sequence equality, Lean verification, and bridge checking as local certificate data.*

No convergent-real-sequence Cauchy obligation is stored outside these rows.

Proof. Project the carrier of Definition 1701.1. The visible packet consists of the displayed convergence row L, finite windows W, rational readback Q, dyadic ledger D, late-pair test T, Cauchy-consumer row E, and structural rows H, R, P, N.

The limit row is the only source of convergence data. It supplies the displayed sequence, the displayed limit, and the tail schedule that selects the two late reads used by the Cauchy test.

The window, readback, and dyadic rows align those two reads. The row W exposes only the selected sequence and limit windows, Q reads back the corresponding regular rational observations, and D records the split tolerance and triangle comparison.

The late-pair ledger T then forms the Cauchy test by comparing both late sequence values to the same displayed limit row. Only after that finite test is displayed can the $\mathsf{RealCauchyCriterion}^{\uparrow}$ row E be read.

Finally, H, R, P, N transport, replay, record, and name the same finite route. Since none of the excluded objects occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 1701.3 (Convergent real sequence Cauchy criterion handoff)**.** *Every* $\mathsf{RealCauchyCriterion}^{\uparrow}$*-facing read from an accepted* $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ *carrier*

$$C = (L, W, Q, D, T, E, H, R, P, N)$$

factors through the displayed finite route

$$L \longmapsto W \longmapsto Q \longmapsto D \longmapsto T \longmapsto E \longmapsto H, R, P, N.$$

Consequently the consumer receives one displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ *row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows for the sequence and limit,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance split, a late-pair Cauchy-test ledger, a* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *consumer row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The route exports no Real completeness theorem, quotient stream, selected limit, trichotomy principle, countable-choice selector, host sequence equality, Lean verification, or bridge checking.*

Proof. Begin with a Cauchy-facing read. The carrier has no direct Cauchy coordinate before E, so the read must first expose the displayed convergence row L.

The two late sequence observations are then opened through W and Q. These rows provide only the finite windows and rational readbacks selected by the convergence tail schedule and the late-pair request.

The tolerance accounting is supplied by D. Its split budget and triangle comparison turn the two reads against the common displayed limit into the late-pair ledger T.

The consumer row E is downstream of that ledger. Thus any $\mathsf{RealCauchyCriterion}^{\uparrow}$ use sees L, W, Q, D, T first, and the structural rows H, R, P, N only preserve and name the same route.

Because Real completeness, quotient streams, selected limits, trichotomy, countable choice, host sequence equality, Lean evidence, and bridge data are absent from the carrier, none of them can be exported by the handoff. □

Theorem 1701.4 (Convergent real sequence Cauchy late-pair exhaustion)**.** *Let*

$$C = (L, W, Q, D, T, E, H, R, P, N)$$

be an accepted $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ *carrier. Every accepted late-pair Cauchy-test ledger* T *is exhausted by two convergence reads from the same* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row* L*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* Q*, and one* $\mathsf{DyadicRatCore}^{\uparrow}$ *split ledger* D*. No third limit row, alternate window family, selected limit, quotient stream equality, trichotomy principle, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can participate in the Cauchy handoff.*

Proof. Project the accepted carrier. The only limit coordinate is L, the only finite-window coordinate is W, the only rational readback coordinate is Q, and the only tolerance-split coordinate is D.

Apply the limit-row exactness, window and readback alignment, and dyadic split exactness clauses of Definition 1701.2. These clauses force T to use two late reads selected by the same tail schedule in L, opened through W and Q, and compared with the split ledger recorded in D.

Now apply the criterion handoff theorem Definition 1701.3. The consumer row E is downstream of exactly the displayed route through L, W, Q, D, T.

Since the carrier has no coordinate for a third limit row, a second window family, a selected host limit, quotient stream equality, trichotomy, countable choice, or ambient completeness, none of those objects can enter the late-pair ledger or the downstream Cauchy read. □

Falsifiable prediction: Every $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ criterion-facing read factors through a displayed RealSequenceLimit row, finite StreamName windows for the sequence and limit, RegSeqRat readback, a DyadicRatCore tolerance split, a late-pair Cauchy-test ledger, a RealCauchyCriterion consumer row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real-sequence-limit, real-cauchy-criterion, cauchyoscillation, streamname, regseqrat, dyadicratcore, real: $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a displayed convergence row, selected sequence-and-limit windows, regular rational readback, dyadic split accounting, a late-pair Cauchy-test ledger, criterion handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}$ packet is generated from a RealSequenceLimit row, finite StreamName windows for the source sequence and displayed limit, RegSeqRat readback, a DyadicRatCore tolerance ledger, a late-pair Cauchy-test ledger, a RealCauchyCriterion consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1701.1, Definition 1701.2, Definition 1701.3, and Definition 1701.4 seed the finite route from displayed convergence data to Cauchy readiness through late-pair dyadic accounting, readback, criterion handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvergentRealSequenceCauchy}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConvergentRealSequenceCauchyUp.ConvergentRealSequenceCauchyTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, trichotomy, countable choice, host sequence equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, limit-row exactness, window and readback alignment, dyadic split exactness, late-pair Cauchy-test exactness, criterion handoff, transport, replay, provenance, and local naming for BEDC.Derived.ConvergentRealSequenceCauchyUp.

Chapter 1702

A Concrete Naming Certificate for $\mathsf{RegularCauchyMultiplication}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ packet records binary multiplication of displayed regular rational Cauchy real names as a finite bounded-window product route. It sits over the regular rational source rows of chapter 301, the synchronized window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the addition route of chapter 1695, the subtraction route of chapter 1696, the sign-flip route of chapter 459, the algebra packet of chapter 1470, and the terminal real-name seal of chapter 38. The packet is the finite product surface consumed by later Real algebra and completion routes; it is not host real multiplication, a quotient Cauchy-class operation, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1702.1 (RegularCauchyMultiplication carrier)**.** *A* $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B_{\times} = (X, Y, W, D, B, E, M, Z, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for regular rational Cauchy real names.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window ledger aligning the finite observations of X and Y at every product-facing precision request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the two source tolerance rows, the displayed endpoint products, and the dyadic product tolerance row.*
4. *B is the finite bounding ledger used to choose a product window large enough for the two source magnitudes and the requested output tolerance.*
5. *E is the finite error-budget accounting row combining source tolerance, bounding data, dyadic product tolerance, and the output tolerance required by the product readback.*
6. *M is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed binary multiplication route.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the two source rows, synchronized windows, dyadic tolerance rows, bounding ledger, finite error budget, and product readback have been displayed.*
8. *H records componentwise* hsame *transport for the source rows, synchronized windows, dyadic ledger, bounding ledger, finite budget row, readback, seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from any multiplication-facing precision request through* $X, Y, W, D, B, E, M, Z, H$
10. *P is the* Pkg *provenance over* RegularCauchyMultiplication$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegularCauchyAddition$^\uparrow$, RegularCauchySubtraction$^\uparrow$, RegularCauchyNegation$^\uparrow$, RegularCauchyRing$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyMultiplication}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, B, E, M, Z, H, C, P, N$. The carrier has no coordinate for host real multiplication, quotient Cauchy classes, selected limits, trichotomy, ambient Real$^\uparrow$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMultiplicationUp.TasteGate.RegularCauchyMultiplicationTasteGate`
`single carrier alignment`

Theorem 1702.2 (Regular Cauchy multiplication NameCert obligations)**.** *For every accepted* RegularCauchyMultiplication$^\uparrow$ *carrier*

$$B_\times = (X, Y, W, D, B, E, M, Z, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMultiplication}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* RegSeqRat$^\uparrow$ *source rows, synchronized* StreamName$^\uparrow$ *windows, dyadic tolerance rows, finite bounding ledger, finite error-budget row, product readback,* Real$^\uparrow$ *seal row, and structural rows;*
2. *synchronized-window exposure, requiring each multiplication-facing precision request to inspect the two source windows before any endpoint product row is available;*
3. *dyadic endpoint product, requiring D to be read from the displayed source observations and tolerance rows through the* DyadicRatCore$^\uparrow$ *arithmetic route;*
4. *bounding handoff, requiring B to record enough finite source-magnitude information to allocate the product tolerance without appealing to a completed real norm or global order;*
5. *finite error-budget accounting, requiring E to combine the source tolerance, bounding data, dyadic product tolerance, and output tolerance before M is available;*
6. *Real seal nonescape, requiring every* Real$^\uparrow$ *consumer to read the route through X, Y, W, D, B, E, M before any terminal seal is exposed;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite multiplication packet;*
8. *refusal of host real multiplication, quotient Cauchy classes, selected limits, trichotomy, ambient* Real$^\uparrow$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No multiplication obligation is stored outside these finite rows.

Theorem 1702.3 (Regular Cauchy multiplication Real seal handoff)**.** *Every Real-facing product read exported by an accepted* RegularCauchyMultiplication$^\uparrow$ *carrier factors through the finite route*

$$(X, Y) \longmapsto W \longmapsto D \longmapsto B \longmapsto E \longmapsto M \longmapsto Z$$

before it may be consumed as a completed real-name boundary. Thus the consumer receives two regular rational source rows, synchronized StreamName$^\uparrow$ *windows,* DyadicRatCore$^\uparrow$ *tolerance and*

endpoint-product rows, a finite bounding ledger, a finite product-budget ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ *product readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. It receives no host real multiplication, quotient Cauchy class, selected limit, trichotomy, ambient completeness coordinate, Lean verification, or bridge checking.*

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMultiplication}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMultiplication}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1702.1, Definition 1702.2, and Definition 1702.3 seed the finite binary multiplication route through $\mathsf{RegSeqRat}^{\uparrow}$ source rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and endpoint-product rows, finite bounding and error-budget rows, resulting $\mathsf{RegSeqRat}^{\uparrow}$ product readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMultiplication}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyMultiplicationUp.TasteGate.RegularCauchyMultiplicationTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not close host real multiplication, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic endpoint product, finite bounding, finite error-budget accounting, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMultiplicationUp.

Constructive story: A $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, synchronized $\mathsf{StreamName}^{\uparrow}$ finite-window schedules, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and endpoint-product rows, a finite bounding ledger, a finite error-budget row, a resulting $\mathsf{RegSeqRat}^{\uparrow}$ product readback, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1703

A Concrete Naming Certificate for $\mathsf{PrimitiveRecursion}^{\uparrow}$

Kleene/Dedekind-style primitive recursion grounding reads a function over unary depth by naming the seed row, the displayed step row, and the finite trace that iterates the step along a $\mathsf{Nat}^{\uparrow}$ spine. The $\mathsf{PrimitiveRecursion}^{\uparrow}$ packet sits after the unary-history surface of chapter 28, the continuation addition route of chapter 29, the finite-window schedule surface of chapter 126, the regular rational readback route of chapter 301, and the regular Cauchy subtraction consumer of chapter 1696. It gives L10 Real and Completion chapters a displayed schedule generator over unary indices; it is not host natural-number recursion, a partial-recursive minimisation principle, quotient stream equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1703.1 (PrimitiveRecursion carrier)**.** *A* $\mathsf{PrimitiveRecursion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, T, U, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Nat}^{\uparrow}$ *unary-depth row. It is the only index spine along which the recursion may advance.*
2. *S is the seed row. It records the value exposed at depth zero before any step row is replayed.*
3. *T is the step row. It records one displayed successor transition, with its predecessor value, successor depth, and output value carried as finite* BHist *data.*
4. *U is the recursive trace row. It is a finite UnaryHistory list of seed-and-step applications from S along D to the terminal value named by the packet.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule handoff. It converts a requested unary index into the finite window schedule that downstream stream consumers may inspect.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule consumer row. It records the regular rational readback that consumes the schedule exported by W.*
7. *H records componentwise* hsame *transport for the unary depth, seed, step, trace, stream schedule, regular rational consumer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from any schedule-facing request through $D, S, T, U, W, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{PrimitiveRecursion}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{Add}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchySubtraction}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveRecursion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, T, U, W, Q, H, C, P, N$. The carrier has no coordinate for host natural-number recursion, unbounded minimisation, quotient stream equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PrimitiveRecursionUp.TasteGate.PrimitiveRecursionTasteGate single carrier alignment`

Theorem 1703.2 (PrimitiveRecursion NameCert obligations)**.** *For every accepted* $\mathsf{PrimitiveRecursion}^{\uparrow}$ *carrier*

$$R = (D, S, T, U, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimitiveRecursion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{Nat}^{\uparrow}$ *unary-depth row, seed row, step row, recursive trace,* $\mathsf{StreamName}^{\uparrow}$ *schedule handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *schedule consumer, and structural rows;*
2. *seed exactness, requiring the trace U to expose S at the zero depth before any successor transition is read;*
3. *step exactness, requiring each successor cell of U to be obtained by replaying the displayed step row T from the previous trace value;*
4. *unary-depth coverage, requiring the terminal trace cell to occur at the displayed depth D and not at an ambient host index;*
5. *schedule handoff, requiring every stream-facing request to pass through W before any* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer row Q is inspected;*
6. *componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite recursion packet;*
7. *refusal of host natural-number recursion, unbounded minimisation, quotient stream equality, countable choice, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No primitive-recursion obligation is stored outside these rows.

Proof. Carrier admission is the projection of Definition 1703.1. The projected rows are precisely the unary depth D, seed S, step row T, trace U, schedule W, regular consumer Q, and structural rows H, C, P, N.

Seed exactness is the zero-depth face of the UnaryHistory trace. Before any successor cell is present, the trace has only the displayed seed row, so the certificate cannot read a hidden initial value from a host recursion operator.

Step exactness is checked cell by cell. Each successor cell in U names its predecessor value, successor depth, and output value through T, while $\mathsf{Add}^{\uparrow}$ supplies only the unary continuation discipline needed to move from one displayed depth to the next.

The terminal-depth and schedule obligations then follow from the order of the rows. The trace reaches the displayed $\mathsf{Nat}^{\uparrow}$ row D before W is available, and Q is downstream of W, so regular rational consumers receive only the finite schedule generated by the packet.

Finally, H, C, P, N transport, replay, package, and name the same rows. Since none of these structural rows contains host recursion, minimisation, quotient equality, choice, selected limits, completeness, Lean verification, or bridge checking, the seven listed rows exhaust the local NameCert surface. □

Theorem 1703.3 (PrimitiveRecursion unary schedule realization)**.** *Let $R = (D, S, T, U, W, Q, H, C, P, N)$ be an accepted* $\mathsf{PrimitiveRecursion}^{\uparrow}$ *carrier. The schedule row W is realized by induction on the UnaryHistory trace U: the zero cell reads the seed row S, and each successor cell reads the preceding trace value together with the displayed step row T. Consequently every schedule exported by W is indexed by the* $\mathsf{Nat}^{\uparrow}$ *depth D and by no host natural-number recursion coordinate.*

Lean4 ✓ `BEDC.Derived.PrimitiveRecursionUp.PrimitiveRecursionCarrier unary schedule realization`

Proof. Induct on the UnaryHistory trace row U. In the base case the only admitted trace cell is the displayed seed S, and the schedule opened by W has zero-depth source D.

For a successor trace, apply the induction hypothesis to the predecessor trace. The next cell is admitted only by replaying the displayed step row T against that predecessor value and the successor depth supplied by the $\mathsf{Nat}^{\uparrow}$ spine.

The $\mathsf{Add}^{\uparrow}$ dependency is only the continuation route between adjacent unary depths. It contributes no independent recursion principle, so the terminal schedule remains the finite trace schedule already named by U and W. □

Theorem 1703.4 (PrimitiveRecursion StreamName and RegSeqRat handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{PrimitiveRecursion}^{\uparrow}$ *carrier factors through the finite route*

$$D \longmapsto S \longmapsto T \longmapsto U \longmapsto W \longmapsto Q \longmapsto H, C, P, N.$$

Thus downstream L10 Real and Completion consumers, including routes that later use $\mathsf{RegularCauchySubtraction}^{\uparrow}$*, receive a displayed unary schedule, seed, step, trace, stream window schedule, regular rational consumer row, transport, replay, provenance, and local naming. They receive no host natural-number recursion, unbounded minimisation, quotient stream equality, countable choice, selected limit, ambient completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PrimitiveRecursionUp.PrimitiveRecursionStreamScheduleHandoff`

Proof. Begin with a stream-facing or regular-rational-facing read of R. By Definition 1703.1, the only stream schedule coordinate is W, and the only regular rational consumer coordinate is Q.

The row W is downstream of the unary depth D, seed S, step row T, and trace U. By Definition 1703.3, this means the schedule has already been generated by the displayed UnaryHistory induction before any stream window may be inspected.

The row Q then consumes W through the regular rational readback surface of chapter 301. Later real and completion routes may pass that scheduled read to sibling consumers such as chapter 1696, but they cannot bypass the displayed route.

Componentwise transport, replay, provenance, and local naming preserve the same order. Hence every exported read factors through the stated finite route and exposes none of the excluded ambient data. □

Theorem 1703.5 (PrimitiveRecursion RegSeqRat consumer induction)**.** *Let $R = (D, S, T, U, W, Q, H, C, P, N)$ be an accepted* $\mathsf{PrimitiveRecursion}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of R is obtained by induction on the displayed UnaryHistory trace U before the consumer row Q is read: the base step reads the seed row S, the successor step reads the displayed step row T against the predecessor trace cell, the schedule step reads W only after that induction has generated the requested finite window, and the regular rational consumer step reads Q only after D, S, T, U, W are visible.*

Consequently L10 Real and Completion consumers inherit exactly the UnaryHistory induction route of the packet. They do not receive host recursion, unbounded minimisation, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as part of the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Project the carrier in Definition 1703.1. The only recursion-bearing row is the displayed trace U, whose zero face is S and whose successor face is replay of T along the unary depth row D.

Induct on U. In the base case, the consumer sees only the seed row S before any stream schedule exists. In the successor case, the predecessor trace value is already available by the induction hypothesis, and the next value is admitted only through the displayed step row T.

After the trace cell is produced, apply Definition 1703.3. The schedule row W is therefore a finite window generated from D, S, T, U, not a host recursion coordinate.

Finally apply Definition 1703.4. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q consumes that schedule and cannot move ahead of it, while H, C, P, N only transport, replay, source, and name the same finite packet. The excluded principles are absent from the carrier and hence absent from the consumer induction route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveRecursion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1703.1, Definition 1703.2, Definition 1703.3, and Definition 1703.4 seed the finite unary-history recursion schedule over $\mathsf{Nat}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RegularCauchySubtraction}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host natural-number recursion, unbounded minimisation, quotient stream equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seed exactness, step exactness, unary-depth coverage, StreamName schedule handoff, RegSeqRat consumer handoff, componentwise transport, replay, provenance, and local naming for BEDC.Derived.PrimitiveRecursionUp.

Constructive story: A $\mathsf{PrimitiveRecursion}^{\uparrow}$ packet is generated from a $\mathsf{Nat}^{\uparrow}$ unary-depth row, a seed row, a displayed step row, a finite UnaryHistory recursive trace, a $\mathsf{StreamName}^{\uparrow}$ schedule handoff, a $\mathsf{RegSeqRat}^{\uparrow}$ schedule consumer, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1704

A Concrete Naming Certificate for $\mathsf{SternBrocot}^{\uparrow}$

Following Stern–Brocot and Farey rational approximation, the $\mathsf{SternBrocot}^{\uparrow}$ packet records the finite rational tree whose adjacent boundary rows have determinant one and whose mediant steps refine rational intervals without choosing a host real representative. It sits above the bracketing approximation packet of chapter 1431 and the breadth-first positive-rational enumeration packet of chapter 1474. Its L10 real-readback neighbors are the continued-fraction route of chapter 342 and the rational-stream route of chapter 350; the packet itself stores only finite tree addresses, rational boundary rows, mediant rows, transport, replay, provenance, and naming.

Definition 1704.1 (Stern–Brocot tree carrier)**.** *A* $\mathsf{SternBrocot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, L, U, M, F, B, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite address row, recording the left/right path from the root interval to the inspected tree node.*
2. *L and U are the lower and upper* $\mathsf{Rat}^{\uparrow}$ *boundary rows bracketing the inspected node.*
3. *M is the mediant row computed from L and U at the addressed node.*
4. *F is the Farey-neighbor ledger, recording the determinant-one boundary check for the displayed pair L, U.*
5. *B is the finite branch ledger, recording which boundary is replaced by M after each inspected step.*
6. *Q is the rational-tree output row exposed to the sibling* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *and* $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ *packets.*
7. *R is the* $\mathsf{RationalStream}^{\uparrow}$ *readback row used when finite tree nodes are consumed as a rational source for L10 real and completion routes.*
8. *H records componentwise* hsame *transport for address, boundary, mediant, Farey-ledger, branch, output, rational-stream, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an address request through boundary inspection, mediant construction, Farey-neighbor checking, branch replacement, tree output, and rational-stream readback.*

10. *P is the* Pkg *provenance over* $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{CalkinWilfEnumeration}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, U, M, F, B, Q, R, H, C, P, N$. The carrier has no coordinate for Dedekind cuts, host trichotomy, quotient rational equality, selected real representatives, ambient completeness, or Lean verification.

Lean4 ✓ `BEDC.Derived.SternBrocotUp.TasteGate.SternBrocotTasteGate single carrier alignment`

Theorem 1704.2 (Stern–Brocot NameCert obligations)**.** *For every accepted* $\mathsf{SternBrocot}^{\uparrow}$ *carrier*

$$S = (A, L, U, M, F, B, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SternBrocot}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the address, boundary, mediant, Farey-neighbor, branch, rational-tree output, rational-stream readback, and structural rows;*
2. *mediant exactness, requiring M to be the displayed numerator-denominator sum of the currently displayed boundary rows L and U;*
3. *Farey-neighbor boundary, requiring F to record determinant-one adjacency for every displayed boundary pair before and after a branch step;*
4. *branch discipline, requiring B to replace exactly one displayed boundary by the displayed mediant at each finite address step;*
5. *sibling readback, requiring every* $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{CalkinWilfEnumeration}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, *or* $\mathsf{RationalStream}^{\uparrow}$ *consumer to pass through Q and R before using the tree node as rational data;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming, requiring H, C, P, N to preserve and name the same finite tree packet;*
7. *refusal of Dedekind cuts, host trichotomy, quotient rational equality, selected representatives, ambient completeness, and Lean verification as local NameCert data.*

No Stern–Brocot obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SternBrocotUp.SternBrocotTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1704.1. The accepted tuple exposes exactly the address, boundary, mediant, Farey-ledger, branch, output, rational-stream, transport, replay, provenance, and local naming rows listed in the statement.

Mediant exactness is local to M. The row records the numerator-denominator sum of the displayed boundary rows L and U, so no host rational search or quotient-rational comparison is needed to name the inspected node.

The Farey-neighbor and branch obligations are read together. The row F records determinant-one adjacency for the current displayed pair, and B records the single boundary replacement by M that advances the finite address. Thus the next inspected pair is still a displayed tree boundary pair.

The sibling readback obligation follows from the order of the carrier rows. The approximation and enumeration siblings consume the same rational-tree output row Q, while continued-fraction and rational-stream consumers read through R before passing data to L10 real routes.

Finally, H, C, P, N transport, replay, package, and name the same finite inventory. Since the inventory has no coordinate for Dedekind cuts, trichotomy, quotient rational equality, selected representatives, ambient completeness, or Lean verification, the listed obligations exhaust the local certificate. □

Theorem 1704.3 (Stern–Brocot Farey neighbor boundary)**.** *Let $S = (A, L, U, M, F, B, Q, R, H, C, P, N)$ be an accepted* $\mathsf{SternBrocot}^{\uparrow}$ *carrier. At every finite prefix of the address row A, the displayed boundary rows L and U named by the corresponding branch prefix are Farey neighbors according to F, and the mediant row M lies between them as the next displayed tree node. Every public boundary read factors through*

$$A \longmapsto L, U \longmapsto M \longmapsto F \longmapsto B \longmapsto Q \longmapsto R.$$

The boundary claim is only about the finite Stern–Brocot tree surface. It does not assert Dedekind cut locatedness, host trichotomy, quotient-real equality, selected representatives, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SternBrocotUp.FareyNeighborBoundary.SternBrocotFareyNeighborBoundary`

Proof. Induct on the finite address prefix in A. At the root prefix the carrier supplies the displayed boundary rows L and U together with the Farey-neighbor ledger F, so the determinant-one boundary is read from the packet itself.

For a successor prefix, use the branch row B. It records whether the left or right boundary is replaced by the mediant row M, and it leaves the other boundary unchanged as displayed BHist data.

The new boundary ledger is again the row F for the successor prefix. Because F is stored as part of the same finite carrier, the determinant-one check is replayed by displayed data rather than by an ambient rational-order principle.

The public readback route then follows the output rows. The rational-tree node is exposed through Q, and L10 real-readback consumers receive it only after the $\mathsf{RationalStream}^{\uparrow}$ row R has been inspected. □

Theorem 1704.4 (Stern–Brocot sibling route to real readback)**.** *Every accepted* $\mathsf{SternBrocot}^{\uparrow}$ *packet supplies a finite rational-tree source for the approximation route of chapter 1431, the enumeration route of chapter 1474, the continued-fraction comparison route of chapter 342, and the rational-stream source route of chapter 350. Each such route reads the same displayed address, boundary, mediant, Farey-neighbor, branch, output, rational-stream, transport, replay, provenance, and local naming rows, and no route receives an independent real representative or ambient completion field.*

Lean4 ✓ `BEDC.Derived.SternBrocotUp.SternBrocotL10RealReadbackRoute`

Proof. Start with an accepted carrier $S = (A, L, U, M, F, B, Q, R, H, C, P, N)$. The tree-facing data are exactly A, L, U, M, F, B, Q, and the rational-stream handoff is exactly R.

The Stern–Brocot approximation sibling reads Q as finite bracketing data and then applies its own tolerance and real-seal rows. The Calkin-Wilf enumeration sibling reads the same finite tree output as positive-rational enumeration data, but it contributes its own breadth-first schedule outside the $\mathsf{SternBrocot}^{\uparrow}$ carrier.

The continued-fraction and rational-stream siblings are readback consumers. They receive the finite rational node only through R, so the real-facing path is a rational-source path rather than a selected host real.

Componentwise transport and replay preserve this route for classified packets. The package and local naming rows record exactly the same dependency surface, so no independent representative, quotient completion, ambient real order, or completeness field is exported. □

Theorem 1704.5 (Stern–Brocot rational interval-tree consumer)**.** *Every* $\mathsf{RationalIntervalTree}^{\uparrow}$ *consumer using a* $\mathsf{SternBrocot}^{\uparrow}$ *source reads finite boundary pairs, mediant rows, Farey-neighbor ledgers, dyadic tolerances,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows before any Real-facing approximation is public. The handoff factors through the* $\mathsf{SternBrocot}^{\uparrow}$ *route of Definition 1704.4 and the rational interval-tree carrier of Definition 7691.1; it exports no selected real representative, quotient equality, host trichotomy, or ambient Real completeness.*

Proof. Start with the accepted $\mathsf{SternBrocot}^{\uparrow}$ carrier. By Definition 1704.4, every real-readback consumer sees the finite address, boundary, mediant, Farey-neighbor, branch, output, rational-stream, transport, replay, provenance, and local naming rows before any real-facing use.

Now attach the rational interval-tree carrier of Definition 7691.1. Its boundary rows L, U, split row M, branch ledger B, dyadic tolerance ledger Q, stream window S, regular-rational readback R, and Real seal E are precisely the finite interval-refinement rows consumed downstream.

The bisection-window theorem Definition 7691.3 orders those rows before any public Real-facing approximation. Composing the two finite routes gives the stated handoff from Stern–Brocot boundary data to the rational interval-tree consumer surface.

Both routes refuse selected representatives, quotient equality, host trichotomy, and ambient completeness as carrier coordinates. Hence the composed consumer receives only finite boundary, mediant, Farey-neighbor, tolerance, window, readback, and seal rows. □

Lean4 ✓ `BEDC.Derived.SternBrocotUp.SternBrocotRationalIntervalTreeConsumer`

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocot}^{\uparrow}}$.

Constructive story: A $\mathsf{SternBrocot}^{\uparrow}$ packet is generated from a finite address row, two displayed rational boundary rows, a mediant row, a Farey-neighbor ledger, a finite branch ledger, a rational-tree output row, a $\mathsf{RationalStream}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1704.1, Definition 1704.2, Definition 1704.3, and Definition 1704.4 seed the finite Stern–Brocot rational-tree certificate above the approximation and enumeration siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SternBrocot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, host trichotomy, quotient rational equality, selected representatives, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mediant exactness, Farey-neighbor boundary, branch discipline, sibling readback, transport, replay, provenance, and local naming for the same finite packet.

Chapter 1705

A Concrete Naming Certificate for $\mathsf{EffectiveCauchySequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{EffectiveCauchySequence}^{\uparrow}$ packet records a Cauchy sequence whose convergence is exposed by an explicit modulus before any $\mathsf{RegSeqRat}^{\uparrow}$ handoff or terminal $\mathsf{Real}^{\uparrow}$ seal is read. It sits on the finite-window route of chapter 126, the modulus schedule surface of chapter 292, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet is a source-level certificate for explicit-modulus Cauchy sequence data on the current Real and Completion route; it is not quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1705.1 (Effective Cauchy sequence carrier)**.** *An* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row. It names the displayed source of observations whose finite windows are inspected by Real-facing or Completion-facing consumers.*
2. *M is the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row. For each requested positive tolerance it gives the finite stage after which the source row is to be read within that tolerance.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the actual inspected source windows downstream of the modulus schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the dyadic radii, slack rows, comparison budgets, and tolerance conversions used by M and W.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It records the regular rational readback obtained only after the source, modulus, window, and dyadic ledger rows have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached downstream of S, M, W, D, Q.*
7. *H records componentwise* hsame *transport for the source, modulus, window, dyadic ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion request through $S, M, W, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequence}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite explicit-modulus Cauchy sequence packet. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.TasteGate.EffectiveCauchySequenceUpTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceTasteGate single carrier alignment`

Theorem 1705.2 (Effective Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier $G = (S, M, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed sequence source, explicit modulus, finite stream windows, dyadic tolerance ledger, regular rational handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *modulus exposure, requiring every tolerance request to read M before the corresponding window row W is consumed;*
3. *dyadic tolerance accounting, requiring every use of the modulus to pass through the dyadic radii, slack rows, comparison budgets, and tolerance conversions stored in D;*
4. *regular readback, requiring Q to be downstream of the source, modulus, window, and dyadic rows before any regular rational observation is exported;*
5. *Real seal boundary, requiring E to remain downstream of S, M, W, D, Q and never to serve as quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient completeness;*
6. *transport, replay, provenance, and local naming stability, requiring H, C, P, N to preserve and name exactly the displayed packet.*

No effective-Cauchy-sequence obligation is stored outside these rows.

Proof. Carrier admission is the projection of Definition 1705.1. The projected rows are exactly the sequence source S, explicit modulus M, finite windows W, dyadic ledger D, regular readback Q, Real seal E, and structural rows H, C, P, N.

Modulus exposure reads chapter 292 together with the finite-window discipline of chapter 126. A tolerance request cannot reach the inspected window row W until the corresponding stage has first been supplied by M.

Dyadic tolerance accounting is supplied by chapter 291. The row D is the only displayed place where radii, slack, comparison budgets, and tolerance conversions live, so every modulus-guided read must pass through it before reaching the regular readback.

The regular readback and Real boundary then follow the route of chapter 301 and chapter 38. Since Q and E are downstream of S, M, W, D, neither row can introduce a quotient stream, selected limit, countable choice, trichotomy, host real equality, or ambient completeness datum.

Finally, H, C, P, N transport, replay, package, and name the same displayed rows. They add no analytic coordinate beyond the carrier, so the six listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceCarrier namecert obligations`

Theorem 1705.3 (Effective Cauchy sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier* $G = (S, M, W, D, Q, E, H, C, P, N)$ *factors through the finite route*

$$S \longmapsto M \longmapsto W \longmapsto D \longmapsto Q \longmapsto H, C, P, N.$$

Thus a regular rational consumer receives the displayed source, explicit modulus, finite stream windows, dyadic tolerance ledger, regular readback, transport, replay, provenance, and local naming before it can inspect the handoff.

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$ read of G. By Definition 1705.1, the only regular readback coordinate is Q, and Q is placed after the source, modulus, window, and dyadic ledger rows.

The first part of the route reads S through the window interface of chapter 126. The explicit stage discipline is then supplied by M through chapter 292, and the inspected finite windows are recorded by W.

The tolerance conversion and comparison work is read through D using chapter 291. Only after that dyadic ledger is fixed does chapter 301 expose the regular rational handoff Q.

The rows H, C, P, N preserve and name the same order. Therefore every regular rational read factors through the displayed route and has no bypass through quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceCarrier regseqrat handoff`

Theorem 1705.4 (Effective Cauchy sequence modulus readback)**.** *Let* $G = (S, M, W, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier. The explicit modulus row* M *determines the dyadic tolerance budget consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* Q *and by the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary* E*: every such read must pass through the same finite route*

$$S \longmapsto M \longmapsto W \longmapsto D \longmapsto Q \longmapsto E$$

before transport, replay, provenance, or local naming can export it.

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceModulusReadback`

Proof. Start with the carrier inventory of Definition 1705.1. The row M is the only explicit modulus coordinate, and the row D is the only dyadic tolerance ledger available to the packet.

The NameCert obligations of Definition 1705.2 require every tolerance request to expose M before the corresponding finite windows W are consumed. The same obligations require all radii, slack rows, comparison budgets, and tolerance conversions to pass through D.

Apply the RegSeqRat handoff theorem Definition 1705.3. It places the regular rational readback Q downstream of S, M, W, D, so the dyadic budget consumed by Q is exactly the one selected by the displayed modulus route.

The Real seal boundary of Definition 1705.5 then places E downstream of the same handoff. By the non-escape theorem Definition 1705.7, no quotient, selected limit, choice, trichotomy, host equality, ambient completeness, Lean, or bridge datum can replace that budget outside the displayed rows. □

Theorem 1705.5 (Effective Cauchy sequence Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier* $G = (S, M, W, D, Q, E, H, C, P, N)$ *factors through*

$$S \longmapsto M \longmapsto W \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the terminal Real seal is available only after the source, explicit modulus, finite windows, dyadic tolerance ledger, and regular rational handoff have been displayed.

Proof. Start from a $\mathsf{Real}^{\uparrow}$-facing consumer. The carrier definition exposes the terminal seal only as the row E in the same finite packet as $S, M, W, D, Q, H, C, P, N$.

Replay the continuation row C. The consumer first reads the source row S, obtains the requested stage from the explicit modulus M, inspects the finite stream windows W, and checks the dyadic tolerance ledger D before any regular rational handoff can be consumed.

The regular readback Q then supplies the only route into the terminal seal of chapter 38. Since E is downstream of Q, the Real-facing read cannot export quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Componentwise transport, replay, provenance, and local naming preserve the route. Hence every Real-facing read factors through the displayed source-to-seal boundary. □

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceRealSealBoundary`

Theorem 1705.6 (Effective Cauchy sequence extractor handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier* $G = (S, M, W, D, Q, E, H, C, P, N)$ *that reads through a* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *route must expose the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M *before any finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{Real}^{\uparrow}$ *seal row is consumed.*

Proof. Project the carrier of Definition 1705.1. The explicit modulus row M is listed before the finite window row W, dyadic ledger D, regular readback Q, and terminal seal E.

The NameCert obligations of Definition 1705.2 require modulus exposure for every tolerance request. Thus a consumer routed through a $\mathsf{CauchyModulusExtractor}^{\uparrow}$ packet must read M before it can open the corresponding finite StreamName window.

The RegSeqRat handoff theorem Definition 1705.3 then places Q downstream of S, M, W, D. The Real seal boundary theorem Definition 1705.5 places E downstream of the same route.

Therefore the extractor handoff cannot consume a finite window, dyadic tolerance ledger, regular readback, or Real seal before the explicit Cauchy modulus row has been displayed. □

Theorem 1705.7 (Effective Cauchy sequence non-escape)**.** *Let* $G = (S, M, W, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *carrier. No consumer of* G *can read quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from the packet unless that datum is already one of the displayed rows* $S, M, W, D, Q, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceUp.EffectiveCauchySequenceNonescape`

Proof. Inspect the carrier rows in Definition 1705.1. The analytic rows are only the source, explicit modulus, finite windows, dyadic ledger, regular readback, and terminal Real seal.

The structural rows H, C, P, N provide sameness transport, replay, provenance, and local naming for those same rows. They do not add a quotient relation, limit selector, choice principle, order decision, host equality predicate, completeness theorem, Lean target, or bridge certificate.

Therefore the excluded data have no carrier coordinate, replay edge, provenance entry, or local NameCert row from which to be read. Any consumer of G is confined to the displayed explicit-modulus route. □

Falsifiable prediction: Every $\mathsf{EffectiveCauchySequence}^{\uparrow}$ regular or Real-facing read factors through a sequence source row, explicit CauchyModulus row, finite StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, RealUp seal, transport, replay, provenance, and local naming.

Independence witness: streamname, cauchymodulus, dyadicratcore, regseqrat, real: $\mathsf{EffectiveCauchySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source row, explicit modulus, finite windows, tolerance ledger, regular handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1705.1, Definition 1705.2, Definition 1705.3, Definition 1705.5, and Definition 1705.7 seed the finite explicit-modulus Cauchy sequence route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exposure, dyadic tolerance accounting, regular readback, Real seal boundary, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.EffectiveCauchySequenceU

Constructive story: An $\mathsf{EffectiveCauchySequence}^{\uparrow}$ packet is generated from a displayed sequence source, explicit $\mathsf{CauchyModulus}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1706

A Concrete Naming Certificate for $\mathsf{EffectivePolishSpace}^{\uparrow}$

Following Alexandrov-Urysohn and Weihrauch, the $\mathsf{EffectivePolishSpace}^{\uparrow}$ packet records complete separable metric data with a visible dense enumeration and explicit Cauchy realizer route before any Polish or Real consumer reads the completed object. It sits over the Polish-space boundary of chapter 1490, the indexed dense-window surface of chapter 1566, the explicit-modulus Cauchy route of chapter 1705, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names effective complete-separable metric data on the current Real and Completion route; it is not a choice-selected dense set, quotient completion equality, open-cover compactness, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1706.1 (Effective Polish space carrier)**.** *An* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (P, D, Q, S, R, M, L, H, C, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row, carrying the complete separable metric surface that downstream Polish consumers are allowed to inspect.*
2. *D is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *row, exposing the indexed dense family and the finite dense windows cited by precision requests.*
3. *Q is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *row, carrying the explicit-modulus Cauchy data generated from the displayed dense indices.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, listing the finite observation windows through which dense indices and Cauchy stages are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the dense index, stream window, and effective Cauchy rows have been displayed.*
6. *M is the dyadic and Real tolerance ledger, storing radii, slack rows, comparison budgets, and seal thresholds used by D, Q, S, R.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the displayed readback route.*
8. *H records componentwise* hsame *transport for the Polish row, dense row, effective Cauchy row, stream windows, regular readback, tolerance ledger, Real seal, and structural rows.*

9. *C records displayed* Cont *replay from a Polish or Real request through $D, S, Q, R, M, L, H, G, N$.*

10. *G is the* Pkg *provenance over* EffectivePolishSpace$^{\uparrow}$, PolishSpace$^{\uparrow}$, CountableDenseSubset$^{\uparrow}$, EffectiveCauchySequer StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{EffectivePolishSpace}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite effective complete-separable metric packet. It has no coordinate for choice-selected dense points, quotient completion equality, open-cover compactness, ambient Real completeness, host equality, trichotomy, Lean verification, or bridge checking.

Theorem 1706.2 (Effective Polish space NameCert obligations)**.** *For every accepted* EffectivePolishSpace$^{\uparrow}$ *carrier $E = (P, D, Q, S, R, M, L, H, C, G, N)$, the local* NameCert$_{\text{EffectivePolishSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Polish, dense, effective Cauchy, StreamName, RegSeqRat, tolerance-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *dense enumerator visibility, requiring every finite dense-window request to read the indexed row D before it reaches S, Q, R, or L;*

3. *effective Cauchy realizer handoff, requiring every Cauchy realization to pass through the explicit row Q and its displayed stream windows before regular rational readback;*

4. *complete-separable metric compatibility, requiring the Polish row P to be read only through the same dense family, explicit Cauchy route, tolerance ledger, and Real seal used by downstream consumers;*

5. *nonescape, excluding choice-selected dense points, quotient completion equality, open-cover compactness, ambient* Real$^{\uparrow}$ *completeness, host equality, trichotomy, Lean verification, and bridge checking.*

No effective-Polish-space obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.EffectivePolishSpaceUp.EffectivePolishSpaceUpTasteGate single carrier alignment`

Proof. Carrier admission is the projection of Definition 1706.1. The projected rows are exactly the Polish-space row P, dense enumerator D, effective Cauchy row Q, finite stream windows S, regular readback R, tolerance ledger M, Real seal L, and structural rows H, C, G, N.

Dense enumerator visibility is supplied by chapter 1566. A finite approximation request must first cite the displayed index row D before the corresponding window can be served by chapter 126.

The effective Cauchy handoff uses chapter 1705. Once the dense index and stream window have been displayed, the row Q supplies the explicit modulus route, and chapter 301 reads the regular rational handoff only downstream of that route.

Metric compatibility is read through chapter 1490 and the terminal seal of chapter 38. Since P and L are carried in the same finite packet as D, Q, S, R, M, the complete-separable metric surface cannot introduce a second dense family, hidden completion equality, or ambient completeness coordinate.

Finally, H, C, G, N transport, replay, package, and name the same displayed rows. They add no analytic coordinate beyond the carrier, so the five listed obligations exhaust the local NameCert surface. □

Theorem 1706.3 (Effective Polish space realizer handoff)**.** *Every Polish-space or Real-facing consumer of an accepted* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carrier* $E = (P, D, Q, S, R, M, L, H, C, G, N)$ *factors through the displayed route*

$$D \longmapsto S \longmapsto Q \longmapsto R \longmapsto M \longmapsto L \longmapsto P, H, C, G, N.$$

Thus the consumer receives the displayed dense index, finite StreamName windows, EffectiveCauchySequence modulus route, RegSeqRat readback, dyadic and Real tolerance ledger, Real seal, Polish row, transport, replay, provenance, and local naming before it can inspect the completed object.

Lean4 ✓ `BEDC.Derived.EffectivePolishSpaceUp.EffectivePolishSpaceRealizerHandoff`

Proof. Start with a Polish-space or Real-facing read of E. By Definition 1706.1, the only dense-coordinate row is D, and the only effective Cauchy route is Q.

The finite observation part reads D through chapter 1566 and then reads S through chapter 126. These rows expose the dense index and the finite windows without selecting a hidden dense point.

The realization part is then forced through chapter 1705. The row Q supplies the explicit modulus route; the row R records the regular rational readback from chapter 301; and the ledger M records the tolerance data used before the terminal seal L is read.

The Polish row P and the Real seal L are therefore downstream of the same finite route. The structural rows H, C, G, N preserve and name that route, so every consumer factors through the displayed handoff and has no bypass through choice, quotient completion equality, open-cover compactness, host equality, trichotomy, or ambient Real completeness. □

Theorem 1706.4 (Effective Polish space complete-separable handoff)**.** *Every* $\mathsf{CompleteSeparableMetric}^{\uparrow}$*-facing read of an accepted* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carrier* $E = (P, D, Q, S, R, M, L, H, C, G, N)$ *factors through*

$$P, D, Q, S, R, M, L, H, C, G, N.$$

In particular, the dense index row D *and effective Cauchy realizer row* Q *are displayed before the complete-separable metric surface is consumed.*

Lean4 ✓ `BEDC.Derived.EffectivePolishSpaceUp.EffectivePolishSpaceCompleteSeparableHandoff`

Proof. Project the carrier in Definition 1706.1. The complete-separable metric consumer sees the Polish row P only inside the same packet as the dense row D, effective Cauchy row Q, StreamName windows S, RegSeqRat readback R, tolerance ledger M, Real seal L, and structural rows H, C, G, N.

Apply Definition 1706.3. The handoff orders the displayed dense index, StreamName windows, effective Cauchy realizer, regular rational readback, tolerance ledger, and Real seal before any completed-object read is available.

The sibling consumer is chapter 1735. Its route can therefore consume the complete-separable metric surface only after P, D, Q, S, R, M, L have been displayed, with H, C, G, N preserving and naming the same route.

No step introduces a second dense family, a hidden Cauchy realizer, quotient completion equality, open-cover compactness, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1706.5 (EffectivePolishSpace dense realizer transport)**.** *Let $E = (P, D, Q, S, R, M, L, H, C, G, N)$ and $E' = (P', D', Q', S', R', M', L', H', C', G', N')$ be accepted* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carriers. Suppose the Polish rows, countable dense families, effective Cauchy realizer rows, StreamName windows, RegSeqRat readback rows, tolerance ledgers, and Real seals transport componentwise under* hsame *and replay through displayed* Cont *routes. Then E' remains an accepted effective Polish packet over the transported rows, with the same dense-realizer handoff to the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.EffectivePolishSpaceUp.EffectivePolishSpaceCarrier dense realizer transport`

Proof. Project the carrier tuple in Definition 1706.1. The analytic route is exactly the Polish row P, dense row D, effective Cauchy row Q, StreamName windows S, RegSeqRat readback R, tolerance ledger M, and Real seal L.

Transport the dense index and realizer rows first. The componentwise hsame hypotheses identify D with D' and Q with Q', while displayed Cont replay keeps the dense windows and effective Cauchy stages on the same finite route.

Next transport the readback surface. The rows S, R, M, L and S', R', M', L' provide the same finite StreamName windows, regular rational readback, tolerance ledger, and Real seal after replay.

Apply Definition 1706.3. The accepted consumer route is already forced through dense indices, finite windows, effective Cauchy data, regular readback, tolerance rows, and the Real seal, so repacking the transported rows yields the accepted packet over E'.

No step introduces choice-selected dense points, quotient completion equality, open-cover compactness, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The transport stays inside the displayed dense-realizer route. □

Theorem 1706.6 (Effective Polish space dense realizer consumer route)**.** *Let $E = (P, D, Q, S, R, M, L, H, C, G, N)$ be an accepted* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carrier. Every Polish-space or Real-facing consumer first reads the displayed Polish row P, indexed dense family D, effective Cauchy realizer Q, StreamName windows S, RegSeqRat readback R, tolerance ledger M, Real seal L, and only then the local* $\mathsf{NameCert}_{\mathsf{EffectivePolishSpace}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.EffectivePolishSpaceUp.EffectivePolishSpace dense realizer consumer route`

Proof. Project the carrier in Definition 1706.1. Its analytic rows are exactly the Polish row, dense-index row, effective Cauchy row, StreamName windows, RegSeqRat readback, tolerance ledger, and Real seal.

Apply Definition 1706.3. The handoff orders the dense index, finite windows, effective Cauchy modulus route, regular rational readback, tolerance ledger, and Real seal before any completed-object read is available.

The local naming row is therefore last. The rows H, C, G, N transport, replay, source, and name the already displayed route; they do not add a second dense enumerator, a hidden realizer, a quotient completion equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 1706.7 (Effective Polish space non-escape)**.** *Let $E = (P, D, Q, S, R, M, L, H, C, G, N)$ be an accepted* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *carrier. No consumer of E can read choice-selected dense points, quotient completion equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, or trichotomy from the packet unless that datum is already one of the displayed rows $P, D, Q, S, R, M, L, H, C, G, N$.*

Proof. Inspect the carrier rows in Definition 1706.1. The analytic rows are only the Polish-space row, countable dense row, effective Cauchy row, finite stream windows, regular readback, tolerance ledger, and terminal Real seal.

The structural rows H, C, G, N provide sameness transport, replay, provenance, and local naming for those same rows. They do not add a selector for dense points, a quotient-equality principle for completions, an open-cover compactness theorem, an ambient completeness theorem, a host equality predicate, or trichotomy.

Therefore the excluded data have no carrier coordinate, replay edge, provenance entry, or local NameCert row from which to be read. Any consumer of E is confined to the displayed effective complete-separable route. □

Falsifiable prediction: Every $\mathsf{EffectivePolishSpace}^{\uparrow}$ Polish-space or Real-facing read factors through a displayed dense index, StreamName finite windows, EffectiveCauchySequence modulus route, RegSeqRat readback, dyadic and Real tolerance ledger, RealUp seal, Polish row, transport, replay, provenance, and local naming.

Independence witness: polishspace, countable-dense-subset, effective-cauchy-sequence, streamname, regseqrat, real: $\mathsf{EffectivePolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Polish row, indexed dense family, effective Cauchy realizer, stream windows, regular readback, tolerance ledger, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectivePolishSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectivePolishSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1706.1, Definition 1706.2, Definition 1706.3, and Definition 1706.7 seed the finite effective complete-separable metric route through $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectivePolishSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completion equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host equality, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense enumerator visibility, effective Cauchy realizer handoff, complete-separable metric compatibility, transport and replay stability, and nonescape for BEDC.Derived.EffectivePolishSpaceUp

Constructive story: An $\mathsf{EffectivePolishSpace}^{\uparrow}$ packet is generated from a displayed PolishSpace row, CountableDenseSubset row, EffectiveCauchySequence row, StreamName finite windows, RegSeqRat readback, dyadic and Real tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1707

A Concrete Naming Certificate for $\mathsf{BishopFanModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen's treatment of fan bars, the $\mathsf{BishopFanModulus}^{\uparrow}$ packet records the finite route by which a barred fan depth ledger becomes an explicit Real and Completion window modulus. It sits after the finite fan certificate of chapter 400, the Cantor-space naming surface of chapter 1475, the StreamName window interface of chapter 126, the RegSeqRat readback boundary of chapter 301, the DyadicRatCore tolerance ledger of chapter 291, the Real seal of chapter 38, and the downstream uniform-modulus consumer of chapter 369. The packet is a finite modulus bridge for L10 Real and Completion routes; it is not open-cover compactness, countable choice, a selected branch, a selected limit, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1707.1 (Bishop fan modulus carrier)**.** *A* $\mathsf{BishopFanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, A, B, S, Q, D, R, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row. It records the displayed binary fan, prefix closure, decidable branch-prefix inspection, and the finite bar-depth bound.*
2. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *address row. It names the finite binary-prefix domain on which the fan row is read, without selecting an infinite branch.*
3. *B is the bar-to-modulus row. For each displayed bar depth from F, it records the finite observation depth that bounds all branch-prefix reads used by the Real or Completion consumer.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records exactly the source windows inspected after the bar-to-modulus row has fixed the required depth.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores dyadic radii, comparison budgets, and refinement slack attached to the same window schedule.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, A, B, S, Q, D have been displayed.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row. It exposes the same depth schedule as an explicit modulus for later uniform, Real, and Completion-facing routes.*

9. *H records componentwise* hsame *transport for the fan, Cantor-address, bar-to-modulus, stream-window, regular-readback, dyadic-tolerance, Real-seal, uniform-modulus, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a consumer request through $F, A, B, S, Q, D, R, U, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopFanModulus}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFanModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, A, B, S, Q, D, R, U, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected infinite branches, selected limits, quotient stream equality, host real equality, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BishopFanModulusUp.TasteGate.BishopFanModulusUp`

Lean4 ✓ `BEDC.Derived.BishopFanModulusUp.TasteGate.BishopFanModulusTasteGate single carrier alignment`

Theorem 1707.2 (Bishop fan modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopFanModulus}^{\uparrow}$ *carrier $M = (F, A, B, S, Q, D, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopFanModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed fan, Cantor-address, bar-to-modulus, stream-window, regular-readback, dyadic-tolerance, Real-seal, uniform-modulus, transport, replay, provenance, and local naming rows;*

2. *fan-bar coverage, requiring every branch-prefix request to be bounded by the finite bar-depth row read from F over the address surface A;*

3. *modulus extraction, requiring the depth row B to convert that finite bar-depth data into the window depth consumed by S and U;*

4. *Real window handoff, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window to pass through S, every regular readback to pass through Q, every dyadic tolerance check to pass through D, and every terminal real read to pass through R;*

5. *uniform-modulus handoff, requiring U to expose only the same finite schedule already displayed by F, A, B, S, Q, D, R;*

6. *transport, replay, provenance, and local naming stability, requiring H, C, P, N to preserve and name exactly the displayed packet.*

No Bishop fan modulus obligation is stored outside these six rows.

Lean4 ✓ `BEDC.Derived.BishopFanModulusUp.BishopFanModulusUpTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1707.1. The projected finite packet is exactly $F, A, B, S, Q, D, R, U, H, C, P, N$.

Fan-bar coverage reads the finite fan row and the Cantor-address row together. The bar depth is supplied by F, and the binary-prefix domain on which it is interpreted is supplied by A, so the argument ranges only over displayed finite prefixes.

The modulus row B converts this bar depth into a window schedule. Once B has fixed the depth, the stream row S can expose finite observations and the uniform row U can expose the same schedule as a $\mathsf{UniformModulus}^{\uparrow}$ consumer surface.

The Real-facing route then passes through Q, D, R. Regular readback is read from Q, dyadic tolerance accounting is read from D, and the terminal $\mathsf{Real}^{\uparrow}$ seal is read from R only after the window schedule has already been displayed.

Finally, H, C, P, N transport, replay, record, and name the same route. Since the carrier contains no further coordinate, the excluded compactness, choice, branch-selection, limit-selection, quotient, host-equality, verification, bridge, and ambient-completeness data cannot be exported. □

Theorem 1707.3 (Bishop fan modulus Real window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read of an accepted* $\mathsf{BishopFanModulus}^{\uparrow}$ *carrier* $M = (F, A, B, S, Q, D, R, U, H, C, P, N)$ *factors through*

$$F \longmapsto A \longmapsto B \longmapsto S \longmapsto Q \longmapsto D \longmapsto R \longmapsto U \longmapsto H, C, P, N.$$

Thus the consumer receives a finite fan-bar depth, a Cantor-prefix address surface, an explicit window schedule, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a Real seal, a UniformModulus handoff, and the structural support rows before it can inspect the Real or Completion boundary.

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ or Completion-facing request. By Definition 1707.1, the request can reach the terminal Real row R only through the displayed packet.

The route first reads F and A. These two rows fix the finite fan surface and the finite Cantor-prefix domain, so the bar-depth request has no access to a selected infinite branch.

The row B then turns the finite bar-depth data into the observation depth consumed by S. The regular readback Q and dyadic tolerance row D are downstream of that same window schedule.

Only after S, Q, D are displayed can the Real seal R and the uniform-modulus row U be read. The rows H, C, P, N preserve the same order as transport, continuation, provenance, and local naming data.

Therefore every Real or Completion-facing consumer factors through the displayed finite route. The carrier has no bypass through open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.BishopFanModulusUp.BishopFanModulusRealWindowHandoff`

Theorem 1707.4 (Bishop fan modulus non-escape)**.** *Let* $M = (F, A, B, S, Q, D, R, U, H, C, P, N)$ *be an accepted* $\mathsf{BishopFanModulus}^{\uparrow}$ *carrier. No consumer of* M *can read open-cover compactness, countable choice, a selected infinite branch, a selected limit, quotient stream equality, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from the packet unless that datum is already one of the displayed rows* $F, A, B, S, Q, D, R, U, H, C, P, N$.

Proof. Inspect the accepted carrier. Its analytic coordinates are the fan row F, Cantor-address row A, bar-to-modulus row B, StreamName window row S, RegSeqRat readback row Q, DyadicRatCore tolerance row D, Real seal row R, and UniformModulus row U.

The remaining coordinates H, C, P, N provide componentwise sameness transport, displayed continuation replay, package provenance, and local naming for those same rows. They do not add a compactness object, choice principle, branch selector, limit selector, quotient relation, host equality predicate, completeness theorem, Lean target, or bridge certificate.

Thus the excluded data have no carrier coordinate, continuation edge, provenance entry, or local NameCert row from which to be read. Every consumer is confined to the displayed fan-bar-to-window-modulus route. □

Lean4 ✓ `BEDC.Derived.BishopFanModulusUp.BishopFanModulusNonescape`

Falsifiable prediction: Every $\mathsf{BishopFanModulus}^{\uparrow}$ Real or Completion-facing read factors through a finite fan row, Cantor-prefix address row, bar-to-modulus row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, UniformModulus handoff, transport, replay, provenance, and local naming.

Independence witness: fantheorem, cantorspace, streamname, regseqrat, dyadicratcore, real, uniformmodulus: $\mathsf{BishopFanModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries fan-bar depth, Cantor-prefix addressing, explicit window scheduling, analytic readback rows, a Real seal, a UniformModulus handoff, and structural support while refusing compactness, choice, branch selection, selected limits, quotient stream equality, host equality, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFanModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFanModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1707.1, Definition 1707.2, Definition 1707.3, and Definition 1707.4 seed the finite fan-bar-to-window-modulus route through $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFanModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-bar coverage, modulus extraction, Real window handoff, uniform-modulus handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.BishopFanModulusUp.

Constructive story: A $\mathsf{BishopFanModulus}^{\uparrow}$ packet is generated from a finite $\mathsf{FanTheorem}^{\uparrow}$ row, $\mathsf{CantorSpace}^{\uparrow}$ address row, bar-to-modulus row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{UniformModulus}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1708

A Concrete Naming Certificate for $\mathsf{CauchyQuotientBoundary}^{\uparrow}$

Following Bishop and Bridges' constructive completion, the $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ packet records quotient-facing Cauchy completion requests only as finite BEDC boundary data. It sits where the $\mathsf{CauchySequenceSpace}^{\uparrow}$ common-window route of chapter 406 meets the $\mathsf{QuotientStreamRefusal}^{\uparrow}$ refusal surface of chapter 998, downstream of the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ regular-completion route of chapter 1639, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite refusal boundary for quotient-style completion reads; it is not a quotient construction, selected representative, host equality, countable-choice principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1708.1 (Cauchy quotient boundary carrier)**.** *A* $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, F, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *source row. It records the finite family, common-window schedule, cofinal Cauchy-window ledger, tolerance row, completion handoff, and local classifier data that are already visible before any quotient-facing request is inspected.*
2. *Q is the quotient-request row. It names the requested equality of Cauchy streams, quotient class, representative, or quotient-completion read without admitting any of those objects as carrier coordinates.*
3. *F is the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *route row. It records the refusal path used when Q asks for a quotient-style completion datum outside the displayed finite route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores dyadic radii, slack rows, comparison budgets, and tolerance conversions consumed by the source and refusal routes.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the inspected source windows that make the Cauchy request visible before refusal is issued.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational observations available after the finite windows and dyadic ledger have been displayed.*

7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row. It is visible only downstream of S, W, D, R, F and does not identify quotient streams or choose a representative.*

8. *H records componentwise* hsame *transport for the source, quotient request, refusal route, dyadic ledger, stream windows, regular readback, Real seal boundary, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a quotient-facing completion request through $S, Q, F, D, W, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyQuotientBoundary}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{QuotientStreamRefusal}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientBoundary}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite quotient-facing refusal-boundary packet. It has no coordinate for quotient stream equality, quotient classes, selected representatives, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyQuotientBoundaryUp.TasteGate.CauchyQuotientBoundaryUpTasteGate single carrier alignment`

↪ variant `BEDC.Derived.CauchyQuotientBoundaryUp.CauchyQuotientBoundaryUp single carrier alignment`

Theorem 1708.2 (Cauchy quotient boundary NameCert obligations)**.** *For every accepted* CauchyQuotientBoundary *carrier $B = (S, Q, F, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientBoundary}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Cauchy-window source, quotient request, refusal route, dyadic tolerance ledger, finite stream windows, regular rational readback, Real seal boundary, transport, replay, provenance, and local naming rows;*

2. *quotient-request routing, requiring every quotient-facing completion request to expose Q and then enter the displayed refusal route F unless it is already a read of the finite Cauchy source route S, W, D, R;*

3. *refusal factorization, requiring F to route off-route quotient-style requests through the* $\mathsf{QuotientStreamRefusal}^{\uparrow}$ *surface before any* $\mathsf{Real}^{\uparrow}$ *seal boundary is visible;*

4. *Real-seal non-escape, requiring E to remain downstream of the CauchySequenceSpace and refusal rows and never to serve as quotient stream equality, a quotient class, selected representative, host equality, countable choice, or ambient completeness;*

5. *transport and replay stability, requiring H and C to preserve the displayed row order $S, Q, F, D, W, R, E, P, N$;*

6. *provenance exhaustion, requiring P and N to name exactly the displayed packet and no quotient object outside it.*

No Cauchy quotient boundary obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyQuotientBoundaryUp.CauchyQuotientBoundaryCarrier namecert obligations`

Proof. Carrier admission is the projection of Definition 1708.1. The accepted packet exposes only the CauchySequenceSpace source row, quotient-request row, refusal route, dyadic ledger, finite windows, regular readback, Real seal boundary, and structural rows.

Quotient-request routing begins at Q. If the request is a finite Cauchy-window read, it remains inside the displayed source route S, W, D, R. If it asks for quotient equality, a quotient class, a

representative, or quotient-completion data beyond that route, the only displayed next row is the refusal row F.

The refusal factorization is supplied by chapter 998. That chapter routes off-route quotient-style requests through the refusal ledger before any Real-completion read can reach a seal, so E stays downstream of S, W, D, R, F.

Finally, componentwise hsame transport, displayed Cont replay, provenance, and local naming preserve the same rows. Since no excluded quotient or completeness coordinate occurs in B, the listed obligations exhaust the local NameCert surface. □

Theorem 1708.3 (Cauchy quotient boundary refusal factorization)**.** *Every quotient-facing Cauchy completion read of an accepted* $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ *carrier* $B = (S, Q, F, D, W, R, E, H, C, P, N)$ *factors through the displayed route*

$$\mathsf{CauchySequenceSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{QuotientStreamRefusal}^{\uparrow} \longmapsto \mathsf{Re}$$

Thus the read consumes the source row S, finite windows W, dyadic ledger D, regular readback R, refusal route F, and only then the Real seal boundary E. It exports no quotient class, selected representative, host equality, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyQuotientBoundaryUp.CauchyQuotientBoundaryCarrier refusal factorization`

Proof. Start with a quotient-facing completion read. The carrier gives the request only as Q, and Definition 1708.1 gives no coordinate where quotient stream equality, a quotient class, or a selected representative could already live.

The Cauchy source part of the read is forced through S. By the $\mathsf{CauchySequenceSpace}^{\uparrow}$ route, the consumer then opens finite $\mathsf{StreamName}^{\uparrow}$ windows, reads the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and obtains the $\mathsf{RegSeqRat}^{\uparrow}$ readback before any completion-facing handoff is visible.

When the same consumer asks for quotient-style data beyond those rows, the next displayed row is F. The $\mathsf{QuotientStreamRefusal}^{\uparrow}$ route sends that request through the refusal ledger, and only after that route is fixed can the terminal $\mathsf{Real}^{\uparrow}$ seal boundary E be inspected.

The structural rows H, C, P, N transport, replay, package, and name this route. They add no quotient class, selected representative, host equality, countable-choice schedule, or ambient completeness witness, so the stated factorization is exhaustive. □

Theorem 1708.4 (Cauchy quotient boundary Real-seal non-escape)**.** *Let* $B = (S, Q, F, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ *carrier. Any consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal boundary E has already consumed the displayed CauchySequenceSpace source, quotient-request row, QuotientStreamRefusal route, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, transport, replay, provenance, and local naming rows. No consumer can use E to recover quotient stream equality, a quotient class, a selected representative, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyQuotientBoundaryUp.CauchyQuotientBoundaryRealSealNonescape`

Proof. The Real seal boundary appears only as the row E in the carrier. Its position in the packet is downstream of S, Q, F, D, W, R, so a consumer cannot reach it before the Cauchy source route and the refusal route have both been displayed.

Replay the request through C. The replay first exposes the source and quotient request, then the finite-window, dyadic-ledger, regular-readback, and refusal rows. Componentwise transport in H preserves those rows rather than replacing them by quotient data.

The provenance and local naming rows P, N name the same finite boundary. Since quotient stream equality, quotient classes, selected representatives, host equality, countable choice, and ambient Real completeness are absent from the carrier, E cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyQuotientBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyQuotientBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1708.1, Definition 1708.2, Definition 1708.3, and Definition 1708.4 seed the finite refusal boundary for quotient-facing Cauchy completion reads through $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{QuotientStreamRefusal}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyQuotientBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, quotient classes, selected representatives, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-request routing, refusal factorization, Real-seal non-escape, transport and replay stability, and provenance exhaustion for BEDC.Derived.CauchyQuotientBoundaryUp.

Constructive story: A $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ packet is generated from a CauchySequenceSpace source row, quotient-request row, QuotientStreamRefusal route, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, Real seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1709

Located Real Interval NameCert Construction

Following Bishop and Bridges, the $\mathsf{LocatedRealInterval}^{\uparrow}$ packet names a finite located real interval whose endpoints, radius budget, locator reads, and locatedness moduli are displayed as one BEDC NameCert surface. It sits after the dyadic endpoint arithmetic of chapter 1645, the tolerance bracket of chapter 1646, the comparison service of chapter 1669, the locatedness budget of chapter 1627, and the quotient boundary of chapter 1708. The packet gives later Real and Completion consumers one named interval target; it is not a host closed interval, a selected pair of quotient representatives, a trichotomy oracle, a compactness theorem, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 1709.1 (Located real interval carrier)**.** *A* $\mathsf{LocatedRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, \rho, \Delta, \Lambda, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *L and U are endpoint* $\mathsf{Real}^{\uparrow}$ *rows, each read through the same real-name seal discipline as the* $\mathsf{RealLocator}^{\uparrow}$ *comparison service.*

2. *ρ is the radius row. It records the dyadic rational slack, positive tolerance request, and endpoint-radius compatibility budget consumed by located interval readers.*

3. *Δ is the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *row that stores endpoint arithmetic, refinement, containment comparison, and outward dyadic enclosure data for the endpoint-radius packet.*

4. *Λ is the* $\mathsf{RealLocator}^{\uparrow}$ *locator row. It records the finite lower or upper apartness windows used when a consumer asks whether a rational probe lies outside or inside the interval boundary.*

5. *M is the* $\mathsf{LocatednessModulus}^{\uparrow}$ *row. It records the tolerance-to-window budget used to turn a positive rational request into endpoint windows, rational cells, dyadic refinements, and real-name readbacks.*

6. *Q is the* $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ *row. It records that distance, locatedness, and equality reads use the same finite tolerance bracket rather than unrelated host comparisons.*

7. *H records componentwise* hsame *transport for endpoints, radius, dyadic interval arithmetic, locator rows, locatedness modulus, metric bracket, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an interval membership, containment, or separation request through $L, U, \rho, \Delta, \Lambda, M, Q, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$, $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealInterval}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one endpoint-plus-radius located interval history. It has no coordinate for host interval equality, selected quotient representatives, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedRealIntervalUp.TasteGate.LocatedRealIntervalUp`
Lean4 ✓ `BEDC.Derived.LocatedRealIntervalUp.TasteGate.LocatedRealIntervalTasteGate single carrier alignment`

Definition 1709.2 (Located real interval classifier)**.** *Two accepted* $\mathsf{LocatedRealInterval}^{\uparrow}$ *carriers*

$$I = (L, U, \rho, \Delta, \Lambda, M, Q, H, C, P, N)$$

and

$$I' = (L', U', \rho', \Delta', \Lambda', M', Q', H', C', P', N')$$

are classified as the same located interval exactly when the endpoint rows are the same $\mathsf{Real}^{\uparrow}$ *endpoints through the displayed real equality boundary, the radius rows are compatible through the shared dyadic tolerance ledger, and the dyadic interval arithmetic, locator, locatedness modulus, and metric bracket rows transport along those endpoint and radius witnesses. The classifier is therefore endpoint* $\mathsf{Real}^{\uparrow}$ *equality plus radius compatibility, replayed through $\Delta, \Lambda, M, Q, H, C, P, N$; it is not equality of host closed intervals and does not collapse undecided boundary probes.*

Theorem 1709.3 (Located real interval NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealInterval}^{\uparrow}$ *carrier $I = (L, U, \rho, \Delta, \Lambda, M, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint rows, radius row, dyadic interval arithmetic row, locator row, locatedness modulus row, metric locatedness bracket, transport, replay, provenance, and local naming row;*

2. *endpoint-radius compatibility, requiring ρ to bind the lower and upper endpoint reads to the dyadic interval ledger Δ without replacing them by host real representatives;*

3. *classifier exactness, requiring sameness to be endpoint* $\mathsf{Real}^{\uparrow}$ *equality plus radius compatibility and to transport through $\Delta, \Lambda, M, Q, H, C, P, N$;*

4. *locatedness stability, requiring Λ and M to answer finite boundary and tolerance requests through displayed windows and dyadic refinements;*

5. *metric bracket stability, requiring Q to keep distance, equality, and locatedness reads on one common tolerance route;*

6. *ledger exhaustion, excluding host interval equality, selected quotient representatives, trichotomy, countable choice, compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No located real interval obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1709.1. The projection exposes exactly the two endpoint rows, radius row, dyadic interval arithmetic row, locator row, locatedness modulus row, metric bracket, and structural rows.

Endpoint-radius compatibility is read from ρ and Δ. The radius row carries the positive dyadic slack and endpoint-radius budget, while the dyadic interval arithmetic row supplies the outward endpoint enclosures and refinement comparisons.

Classifier exactness follows from Definition 1709.2. Sameness first compares the endpoint $\mathsf{Real}^{\uparrow}$ rows through the displayed equality boundary, then checks the radius compatibility ledger, and only then transports the dyadic arithmetic, locator, locatedness, and metric bracket rows.

Locatedness stability is supplied by Λ and M. The locator row answers rational probes by finite apartness windows, and the modulus row turns tolerance requests into endpoint windows, rational cells, dyadic refinements, and real-name readbacks.

Metric bracket stability is supplied by Q. It records that distance, equality, and locatedness requests consume the same bracket route. The remaining structural rows H, C, P, N transport, replay, package, and name the displayed packet, so no excluded host interval or completeness coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalUp.LocatedRealIntervalCarrier namecert obligations`

Theorem 1709.4 (Located real interval exactness route)**.** *Every membership, containment, separation, or refinement read of an accepted* $\mathsf{LocatedRealInterval}^{\uparrow}$ *carrier factors through the displayed route*

$$L, U, \rho \longmapsto \Delta, \Lambda, M, Q \longmapsto H, C, P, N.$$

Consequently the consumer receives endpoint $\mathsf{Real}^{\uparrow}$ *equality data and radius compatibility data before any locatedness answer is read. The route exports no host interval equality, selected quotient representative, trichotomy, countable-choice schedule, compactness witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalUp.LocatedRealIntervalCarrier exactness route`

Proof. Start with an accepted packet $I = (L, U, \rho, \Delta, \Lambda, M, Q, H, C, P, N)$. The only interval coordinates are the endpoint rows L, U and the radius row ρ.

The dyadic interval arithmetic row Δ is the first computational handoff from those coordinates. It records outward endpoint enclosures, containment comparison, and refinement data using displayed dyadic ledgers.

The locator and modulus rows Λ and M then supply the finite boundary and tolerance services. They can only read the endpoint-radius data through the displayed windows, rational readbacks, and dyadic refinements already named by the packet.

The metric bracket Q binds the distance, equality, and locatedness routes together. Componentwise transport, continuation replay, provenance, and local naming then package the same route. Since the carrier contains no host interval equality or completeness coordinate, the listed route is exhaustive. □

Theorem 1709.5 (Located real interval Cauchy quotient boundary route)**.** *Every quotient-boundary-facing read of an accepted* $\mathsf{LocatedRealInterval}^{\uparrow}$ *carrier must pass through the endpoint* $\mathsf{Real}^{\uparrow}$ *rows* L, U*, the radius compatibility row* ρ*, the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *row* Δ*, the* $\mathsf{RealLocator}^{\uparrow}$ *row* Λ*, the* $\mathsf{LocatednessModulus}^{\uparrow}$ *row* M*, and the* $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ *row* Q *before it can reach the* $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ *sibling. Thus the quotient boundary consumes the same endpoint-plus-radius located interval route and cannot replace it by selected quotient representatives, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalUp.LocatedRealIntervalCarrier cauchy quotient boundary route`

Proof. Project the carrier of Definition 1709.1. The endpoint rows L, U and the radius row ρ are the only interval coordinates available before any arithmetic, locator, modulus, or metric bracket data are read.

Apply Definition 1709.4. It forces every membership, containment, separation, or refinement request through the route from endpoints and radius to Δ, Λ, M, Q and then to the structural rows.

The $\mathsf{CauchyQuotientBoundary}^{\uparrow}$ sibling can therefore appear only as a downstream consumer of that same endpoint and radius route. Its quotient-facing read receives the dyadic interval arithmetic, real locator, locatedness modulus, and metric locatedness bracket already displayed by the interval packet.

Since the carrier has no coordinate for selected quotient representatives, trichotomy, countable choice, host interval equality, or ambient real completeness, the boundary-facing read cannot expose any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealInterval}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1709.1, Definition 1709.2, Definition 1709.3, Definition 1709.4, and Definition 1709.5 close the finite endpoint-plus-radius obligation surface over dyadic interval arithmetic, real locator, locatedness modulus, metric locatedness bracket, and Cauchy quotient boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected quotient representatives, trichotomy, countable choice, compactness, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Scoped closure requires a public consumer boundary or bridge theorem for the endpoint-plus-radius located interval surface.

Constructive story: A $\mathsf{LocatedRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{Real}^{\uparrow}$ endpoint rows, a dyadic radius row, a $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ endpoint ledger, a $\mathsf{RealLocator}^{\uparrow}$ comparison row, a $\mathsf{LocatednessModulus}^{\uparrow}$ tolerance row, a $\mathsf{RealMetricLocatednessBracket}^{\uparrow}$ route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1710

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ packet records the positive comparison and apartness reads available from regular Cauchy real names before any trichotomy, quotient-real equality, selected representative, or ambient completeness principle is admitted. It sits after the finite budget route of chapter 991, the comparison boundary of chapter 1381, and the positive-gap interface of chapter 1514. Its public role is an L10 constructive order/apartness boundary over displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Definition 1710.1 (Regular Cauchy apartness-order carrier)**.** *A* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, A, O, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source whose regular rational observations supply the two comparison endpoints or the single signed difference stream under inspection.*
2. *A is the* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *row. It records the finite positive lower-bound budget that must be displayed before any strict comparison or apartness read is exported.*
3. *O is the comparison-direction row. It records whether the packet is being read as positive apartness, $<$, $>$, or a one-sided order read, and it stores only the displayed direction tag and its local classifier data.*
4. *M is the modulus row selecting the finite precision at which the budgeted positive separation is inspected.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selected from X by A, M, and O.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-bound readback row obtained from that finite window.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row tying D and W back to the regular source X.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the budget, direction, window, dyadic positive-bound, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for the source, apartness budget, comparison direction, modulus, window, positive dyadic bound, regular readback, Real seal, replay, provenance, and local name.*

10. *C records displayed* Cont *replay from a comparison or apartness request through $X, A, O, M, W, D, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$, $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessOrder}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for trichotomy, quotient real equality, host real inequality, a selected representative, total order decision, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1710.2 (Regular Cauchy apartness-order NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *carrier $Q = (X, A, O, M, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApart}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the regular source, apartness budget, comparison direction, modulus, finite window, positive dyadic bound, regular readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *classifier stability, requiring transported packets to preserve the same direction tag O, budget row A, window W, positive-bound row D, readback R, and seal E componentwise through H and C;*

3. *positive-boundary exactness, requiring every exported strict comparison or apartness read to pass through the displayed* $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ *row before reading the* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive bound;*

4. *Real-seal non-escape, requiring the* $\mathsf{Real}^{\uparrow}$ *seal E to remain downstream of A, O, M, W, D, and R and never to serve as quotient equality, trichotomy, selected representative, host inequality, or completeness data;*

5. *ledger exclusions, requiring P and N to name exactly the displayed packet and to export no total order decision, no comparison by host real values, no quotient representative, and no ambient completion theorem.*

No regular Cauchy apartness-order obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessOrderUp.RegularCauchyApartnessOrderUpTasteGate`
`single carrier alignment`

Proof. Carrier admission is the projection of Definition 1710.1. The accepted packet exposes the source, budget, direction, modulus, window, positive-bound, readback, seal, and structural rows, and it has no additional comparison coordinate.

Classifier stability is componentwise. The transport row H moves exactly the rows already displayed, and the replay row C repeats the same comparison route after transport instead of recomputing it from host real values.

Positive-boundary exactness follows from the row order. A strict comparison or apartness read first exposes the budget row A, then the direction row O and modulus row M, then the finite window W, and only then the dyadic positive-bound row D.

The Real-seal and ledger clauses are inventory consequences. Since E is the only $\mathsf{Real}^{\uparrow}$ row and it is downstream of A, O, M, W, D, R, it cannot supply quotient equality, trichotomy, selected

representatives, host inequality, or completeness. The provenance and local naming rows therefore exhaust the packet rather than opening any further order principle. □

Theorem 1710.3 (Regular Cauchy apartness-order positive boundary)**.** *Every exported comparison or apartness read of an accepted* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegularCauchyApartnessBudget}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, a packet $Q = (X, A, O, M, W, D, R, E, H, C, P, N)$ *exposes its comparison direction through* O*, consumes the positive budget* A*, reads the finite window* W*, obtains the positive dyadic boundary* D*, reattaches it to the regular source through* R*, and only then reaches the Real seal* E*. The sibling surfaces of chapter 991, chapter 1381, and chapter 1514 are used only as displayed certificate rows, not as sources of trichotomy, quotient real equality, host inequality, selected representatives, total order decisions, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessOrderUp.RegularCauchyApartnessOrderPositiveBoundary`

Proof. Start with an exported comparison or apartness read. By Definition 1710.2, it must be a read of the finite packet Q and must preserve the displayed comparison direction row O.

The positive part of the read is forced through A. The budget row is the only public source of positive lower-bound data, so the consumer cannot reach a strict comparison or apartness assertion before the $\mathsf{RegularCauchyApartnessBudget}^{\uparrow}$ route is displayed.

The finite observation route then passes through W and D. The window row is a $\mathsf{StreamName}^{\uparrow}$ inspection of the regular source, and the positive-bound row is a $\mathsf{DyadicRatCore}^{\uparrow}$ read from that finite window.

Finally, R reattaches the dyadic bound to the $\mathsf{RegSeqRat}^{\uparrow}$ source and E provides the terminal $\mathsf{Real}^{\uparrow}$ seal. The comparison rows supplied by the BishopRealComparison and ApartnessSpace siblings remain boundary rows on this route; they do not add total comparison, quotient equality, host inequality, selected representatives, or completeness data. □

Theorem 1710.4 (Regular Cauchy apartness-order located comparison scope)**.** *Every accepted* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *comparison that is read as located apartness or located order evidence factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Equivalently, the packet exports located comparison evidence only after the dyadic tolerance row D*, the finite observation window* W*, the regular readback row* R*, the Real seal* E*, and the kernel rows* H, C, P, N *have all been displayed. It does not export trichotomy, quotient real equality, selected representatives, total order decisions, host real inequality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessOrderUp.RegularCauchyApartnessOrderLocatedComparisonScope`

Proof. Project the accepted packet as $Q = (X, A, O, M, W, D, R, E, H, C, P, N)$. The carrier and NameCert-obligation rows expose only the displayed comparison direction, budget, modulus, finite window, positive dyadic bound, regular readback, Real seal, transport, replay, provenance, and local naming data.

Apply Definition 1710.3. Any strict comparison or apartness read must first consume the budget row and then pass through the finite window W, positive dyadic boundary D, regular readback R, and Real seal E.

The located comparison scope is obtained by replaying that route through the kernel rows. The BHist packet supplies the finite ledger, H supplies componentwise hsame transport, C supplies displayed Cont replay, P records Pkg provenance, and N records the local NameCert row.

Finite ledger induction over the displayed observation windows keeps the exported evidence located: each read is tied to an explicit dyadic tolerance and finite stream observation before it reaches the regular source and Real seal. Since no packet coordinate contains trichotomy, quotient equality, selected representatives, total order decisions, host real inequality, or completeness, none of those principles is exported by the located comparison route. □

Theorem 1710.5 (Regular Cauchy apartness-order refusal boundary)**.** *An accepted* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *packet exports only the finite surface*

$$X, A, O, M, W, D, R, E, H, C, P, N.$$

Any consumer asking for trichotomy, quotient real equality, host real inequality, selected representatives, total order decisions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking leaves the carrier surface.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessOrderUp.RegularCauchyApartnessOrderRefusalBoundary`

Proof. Read the carrier inventory. The packet contains one regular source, one budget row, one comparison-direction row, one modulus row, one finite window, one positive dyadic bound, one regular readback, one Real seal, and the structural rows H, C, P, N.

Apply Definition 1710.2. The local NameCert obligations admit and transport exactly those fields, while positive-boundary exactness and Real-seal non-escape force all comparison reads through the same finite route.

The excluded objects are not packet coordinates. A consumer that asks for trichotomy, quotient equality, host inequality, a representative, a total order decision, choice, completeness, Lean verification, or bridge checking therefore asks for data outside $X, A, O, M, W, D, R, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1710.1, Definition 1710.2, Definition 1710.3, Definition 1710.4, and Definition 1710.5 seed the constructive comparison boundary for regular Cauchy real names through finite apartness-budget, window, dyadic positive-bound, regular readback, Real-seal, and kernel replay rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host real inequality, selected representatives, total order decisions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, positive-boundary exactness, located-comparison scope, Real-seal non-escape, ledger exclusions, and refusal-boundary proofs for BEDC.Derived.RegularCauchyApartnessOrd

Constructive story: A $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a RegularCauchyApartnessBud budget row, a displayed comparison-direction row, a modulus row, a $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ positive-bound readback, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1711

A Concrete Naming Certificate for $\mathsf{LocatedIntervalBisection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalBisection}^{\uparrow}$ packet names the finite bisection route for a located real interval without replacing it by a host closed interval or by a completed choice of midpoint decisions. It consumes the located endpoint-plus-radius surface of chapter 1709, the dyadic branch ledger of chapter 361, the retained-window discipline of chapter 320, the downstream root-location consumer of chapter 1402, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records one requested finite bisection history: located interval data, dyadic midpoint rows, refinement signs or locator decisions, retained nested windows, and a Real-facing seal route.

Definition 1711.1 (Located interval bisection carrier)**.** *A* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, D, M, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row. It supplies lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the radius budget, locator reads, and the locatedness modulus exposed by the source interval.*
2. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger. It records the finite left-or-right branch prefix and the dyadic enclosure scale used by the requested bisection depth.*
3. *M is the midpoint row. It stores the displayed dyadic midpoint cell, the two child interval cells, and the endpoint-radius compatibility budget inherited from I.*
4. *S is the sign or refinement row. It records the finite locator, sign, or containment decision that selects the retained child interval at each branch of D.*
5. *W is the retained nested-window row. It stores the visible lower and upper windows after each retained branch and exposes them to chapter 320.*
6. *R is the* $\mathsf{Real}^{\uparrow}$*-facing seal row. It is reached only after the retained window has passed through the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route.*
7. *H records componentwise* hsame *transport for I, D, M, S, W, R and the local structural rows.*

8. *C records displayed* Cont *replay from a bisection request through interval inspection, midpoint construction, branch retention, nested-window exposure, regular readback, and the Real-facing seal.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalBisection}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located-interval bisection packet. It has no coordinate for host interval equality, trichotomy, selected quotient representatives, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a choice-selected root.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalBisectionUp.LocatedIntervalBisectionTasteGate single carrier alignment`

Theorem 1711.2 (Located interval bisection NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *carrier* $B = (I, D, M, S, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalBisection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval row, dyadic branch ledger, midpoint row, sign or refinement row, retained nested window, Real-facing seal, transports, replay, provenance, and local naming;*

2. *interval-source discipline, requiring every endpoint, radius, locator, and locatedness read to factor through I before midpoint data are inspected;*

3. *midpoint branch discipline, requiring M and S to attach each retained child interval to the branch prefix recorded by D;*

4. *nested-window exposure, requiring W to store only the retained lower and upper windows produced by the displayed branch ledger;*

5. *Real-facing seal discipline, requiring R to be reached through the retained* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rather than through a host interval object;*

6. *ledger exhaustion, excluding host interval equality, trichotomy, selected quotient representatives, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a choice-selected root, Lean verification, and a standard bridge.*

No located interval bisection obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedIntervalBisectionUp.LocatedIntervalBisectionNameCertObligations`

Proof. Carrier admission is projection from Definition 1711.1. The projection exposes the located source interval, the finite dyadic branch ledger, midpoint data, branch-retention data, nested-window data, and structural rows.

Interval-source discipline is supplied by I. Every endpoint, radius, locator, and locatedness request starts from the accepted $\mathsf{LocatedRealInterval}^{\uparrow}$ row, so the bisection packet cannot replace the source by a host closed interval.

Midpoint branch discipline is supplied by D, M, and S. The branch ledger names the finite prefix, the midpoint row supplies the displayed child cells at each step, and the sign or refinement row selects the retained child from data already attached to that prefix.

Nested-window exposure is supplied by W. It records only the lower and upper windows retained by the displayed branch route, with H and C transporting and replaying those windows componentwise.

The Real-facing seal is therefore downstream of the retained windows and the regular readback route. Provenance and local naming exhaust the remaining coordinates, so no excluded host equality, compactness, completeness, or root-selection coordinate can be exported. □

Theorem 1711.3 (Located interval bisection nested-window route)**.** *Every nested-window,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *carrier factors through the displayed route*

$$I \longmapsto D, M, S \longmapsto W \longmapsto R, H, C, P, N.$$

The consumer first reads the located interval source, then the finite midpoint and branch-retention ledger, then the retained nested window, and only then the regular rational readback and Real-facing seal.

Proof. Start with an accepted packet $B = (I, D, M, S, W, R, H, C, P, N)$. The only source interval coordinate is I, so every endpoint, radius, and locator observation begins at the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ row.

The branch stage is D, M, S. The dyadic ledger records the finite prefix, the midpoint row gives the child cells, and the sign or refinement row records which child is retained. No nested-window row is available before this stage has been displayed.

The retained-window stage is W. It reads the visible lower and upper windows produced by the branch stage and presents them to the nested-interval and stream-name surfaces.

Finally, R, H, C, P, N route the retained windows through regular readback and the Real-facing seal. Since the route is exhausted by the listed rows, a consumer cannot bypass the branch ledger to obtain an independent host interval or selected limit. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalBisectionUp.LocatedIntervalBisectionNestedWindowRoute`

Theorem 1711.4 (Located interval bisection non-escape)**.** *Every public read of an accepted* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *carrier is exhausted by the located interval row* I*, the branch ledger* D*, the midpoint row* M*, the sign or refinement row* S*, the retained window* W*, the Real-facing seal* R*, componentwise transport* H*, continuation replay* C*, provenance* P*, and the local naming row* N*. In particular, the certificate exports no host interval equality, no trichotomy row, no selected quotient representatives, no open-cover compactness witness, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, and no choice-selected root.*

Proof. Apply the NameCert obligation theorem. It lists the carrier, source discipline, midpoint branch discipline, nested-window exposure, Real-facing seal discipline, and ledger exhaustion rows.

Apply the nested-window route theorem for any nested-window, regular-rational, or Real-facing read. Such a read must pass through I, then D, M, S, then W, and finally the structural and seal rows.

The remaining public reads are structural: transport, replay, provenance, and local naming. These are exactly H, C, P, N and do not add analytic coordinates beyond the displayed bisection packet.

Thus every public read is one of the listed coordinates or a route through them. The excluded host equality, trichotomy, quotient-representative selection, compactness, completeness, and root-selection principles have no row from which they could be projected. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalBisection}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1711.1, Definition 1711.2, Definition 1711.3, and Definition 1711.4 seed the finite located-interval bisection surface over source interval, dyadic branch, midpoint, retained window, regular readback, and Real-facing seal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedIntervalBisection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, trichotomy, selected quotient representatives, open-cover compactness, ambient RealUp completeness, a choice-selected root, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, midpoint branch discipline, nested-window routing, seal discipline, and non-escape proofs for BEDC.Derived.LocatedIntervalBisectionUp.

Constructive story: A LocatedIntervalBisection$^{\uparrow}$ packet is generated from a LocatedRealInterval$^{\uparrow}$ source row, a DyadicBisection$^{\uparrow}$ branch ledger, displayed midpoint rows, sign or refinement rows, retained nested windows, StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1712

A Concrete Naming Certificate for $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ packet records the finite midpoint route for a displayed located real interval before an interval-refinement or completion consumer may inspect child cells. It consumes the located endpoint and radius discipline of chapter 1709, the common-denominator midpoint ledger of chapter 445, the child-cell handoff of chapter 1465, the finite mesh route of chapter 1250, and the nested completion consumer of chapter 1421. The packet is a midpoint certificate for one located interval; it is not host real equality, a selected quotient representative, an interval completeness principle, or a branch-choice oracle.

Definition 1712.1 (Located interval midpoint carrier)**.** *A* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, E_0, E_1, B, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the parent* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, including the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoints, radius budget, locator route, locatedness modulus, and metric bracket rows.*
2. *D is the* $\mathsf{DyadicMidpoint}^{\uparrow}$ *row that computes the midpoint from the displayed endpoint windows by a common-denominator dyadic calculation.*
3. *E_0 and E_1 are endpoint transport rows. They record the lower-to-midpoint and midpoint-to-upper endpoint reuse checks consumed by child located intervals.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *handoff row that exposes the two child interval cells after the midpoint row has been displayed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the parent endpoint reads, midpoint read, and child endpoint observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite windows, including the regular rational midpoint approximation used by Real-facing readers.*
7. *Q is the pair of* $\mathsf{Real}^{\uparrow}$ *seal rows reached by the midpoint and child endpoint reads through S, R, D, E_0, E_1.*
8. *H records componentwise* hsame *transport for the parent interval, midpoint row, endpoint transports, bisection handoff, windows, readbacks, real seals, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a midpoint, bisection, finite-net, or nested-interval request through* $I, D, E_0, E_1, B, S, R, Q, H, P, N$.
10. *P is the* Pkg *provenance over* LocatedIntervalMidpoint$^\uparrow$, LocatedRealInterval$^\uparrow$, DyadicMidpoint$^\uparrow$, IntervalBisection$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocatedIntervalMidpoint}^\uparrow}$ *row.*

The carrier predicate asserts that these rows form one parent-interval-to-midpoint packet. It has no coordinate for host real midpoint arithmetic, quotient real equality, trichotomy, selected branch choice, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedIntervalMidpointUp.LocatedIntervalMidpointTasteGate single carrier alignment`

Theorem 1712.2 (Located interval midpoint NameCert obligations)**.** *For every accepted* LocatedIntervalMidpoint$^\uparrow$ *carrier* $K = (I, D, E_0, E_1, B, S, R, Q, H, C, P, N)$*, the local* NameCert$_{\text{LocatedIntervalMidpoint}^\uparrow}$ *surface consists exactly of carrier admission for those twelve rows, parent located-interval admission through* I*, dyadic midpoint exactness through* D*, endpoint transport through* E_0, E_1*, interval-bisection handoff through* B*, StreamName and RegSeqRat readback through* S, R*, Real seal exposure through* Q*, componentwise classifier transport through* H*, consumer replay through* C*, package provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalMidpointUp.NameCertObligations.LocatedIntervalMidpointCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1712.1. The displayed tuple exposes the parent located interval, midpoint row, endpoint transport rows, bisection handoff, finite windows, regular readbacks, real seals, and structural rows as finite BHist coordinates.

Parent admission is read through I. The lower and upper endpoint rows, radius budget, locator route, locatedness modulus, and metric bracket remain the only source of interval data available to the midpoint packet.

Dyadic midpoint exactness is read through D. The row is the common-denominator calculation already named by the DyadicMidpoint$^\uparrow$ surface, so the midpoint is obtained from displayed endpoint windows rather than from host real arithmetic.

Endpoint transport is read through E_0, E_1 and then passed to B. These rows record that the left child reuses the parent lower endpoint with the midpoint upper endpoint, while the right child reuses the midpoint lower endpoint with the parent upper endpoint. The IntervalBisection$^\uparrow$ handoff can therefore expose two child cells without selecting a branch.

The finite-window and seal obligations are read through S, R, Q. Componentwise transport, continuation replay, provenance, and local naming are then supplied by H, C, P, N. Since the carrier has no coordinate for quotient equality, trichotomy, selected branch choice, countable choice, or completeness, no such datum can be exported by the local NameCert surface. □

Theorem 1712.3 (Located interval midpoint refinement handoff)**.** *Every interval-refinement, finite-net, or Cantor-intersection consumer of an accepted* LocatedIntervalMidpoint$^\uparrow$ *packet factors its midpoint request through the displayed route*

$$I, D, E_0, E_1, B \longmapsto S, R, Q \longmapsto H, C, P, N.$$

Consequently later IntervalBisection$^\uparrow$, ClosedBoundedIntervalNet$^\uparrow$, *and* CantorIntersection$^\uparrow$ *reads receive a finite located midpoint packet, not host real equality, selected branch choice, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalMidpointUp.Handoff.LocatedIntervalMidpointCarrier refinement handoff`

Proof. Start with the parent row I. It fixes the displayed located real interval before any midpoint or child-cell request is inspected.

Next read D, E_0, E_1, B. The dyadic midpoint row supplies the common-denominator midpoint calculation, the endpoint transport rows bind that midpoint to the two inherited endpoint routes, and the bisection handoff exposes the two child cells as finite interval data.

Now pass to S, R, Q. The finite-window and regular-rational rows provide the readback required by Real-facing consumers, and the seal rows expose only the midpoint and child endpoint names already routed through the displayed packet.

Finally replay the route through H, C, P, N. A finite-net consumer may use the child cells as mesh data, and a Cantor-intersection consumer may use them in a nested interval chain, but both consumers receive the same finite midpoint route. The packet contains no selector for an infinite branch, no quotient real equality, and no completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalMidpoint}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalMidpoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1712.1, Definition 1712.2, and Definition 1712.3 seed the finite midpoint route for one displayed located real interval over dyadic midpoint, endpoint transport, bisection handoff, finite-window readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalMidpoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real midpoint arithmetic, quotient real equality, trichotomy, selected branch choice, countable choice, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic midpoint exactness, endpoint transport, bisection handoff, finite-window readback, Real seal nonescape, and consumer-route proofs for BEDC.Derived.LocatedIntervalMidpointUp.

Constructive story: A $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ packet is generated from a parent LocatedRealIntervalUp row, a DyadicMidpointUp common-denominator midpoint row, endpoint transport rows, an IntervalBisectionUp handoff row, StreamNameUp windows, RegSeqRatUp readbacks, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1713

A Concrete Naming Certificate for $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ packet records the finite located nested-interval bridge from displayed interval bisections to regular Cauchy real names. It consumes the located endpoint-plus-radius surface of chapter 1709, the branch route of chapter 1711, the midpoint handoff of chapter 1712, the retained-window discipline of chapter 320, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite BEDC route for nested located intervals; it is not a host interval-intersection theorem, a selected quotient representative, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1713.1 (Located real nested intervals carrier)**.** *A* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, B, M, D, J, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, including endpoint names, radius data, locator reads, locatedness modulus, and metric bracket rows.*
2. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *row that records the finite retained branch route from the source interval.*
3. *M is the* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ *row that supplies the midpoint and child-cell handoff used by each retained branch.*
4. *D is the diameter-budget row. It records the finite dyadic contraction schedule, the positive tolerance requests, and the proof rows that each retained child interval has a smaller displayed diameter than its predecessor.*
5. *J is the nested-containment row. It records that the retained located intervals are nested through displayed endpoint, midpoint, and branch witnesses rather than through a host intersection object.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row obtained from the retained lower and upper endpoint windows at the requested precisions.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same retained windows and diameter budgets.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, J, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the located interval rows, bisection rows, midpoint rows, diameter budgets, containment witnesses, stream windows, regular readbacks, real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a nested-interval or point-name request through $I, B, M, D, J, S, R, E, H, P,$*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealNestedIntervals}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located nested-interval history. It has no coordinate for host interval intersection, selected quotient representatives, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealNestedIntervalsUp.LocatedRealNestedIntervalsTasteGate single carrier alignment`

Theorem 1713.2 (Located real nested intervals diameter contraction)**.** *For every accepted* LocatedRealNestedInterv *carrier $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$, every retained interval row exposed by J is downstream of the diameter-budget row D. In particular, any finite precision request reaching the stream-window row S first reads a displayed contraction witness in D, then the nested containment witness in J, and only then the retained endpoint windows.*

Proof. Carrier admission is projection from Definition 1713.1. The projection exposes the source located interval, the finite bisection route, midpoint handoffs, diameter budgets, containment witnesses, stream windows, regular readback, and structural rows.

The contraction data are stored in D. Each requested precision has a displayed dyadic tolerance budget and a retained-child diameter row, so a consumer cannot inspect a retained window before the corresponding budget has been read.

The containment data are stored in J. Its rows bind each retained located interval to the previous one through the endpoint, midpoint, and branch witnesses supplied by I, B, M, D.

Finally S is downstream of D and J. The finite stream windows are therefore exposed only after the contraction and containment rows have fixed the retained interval route, and no host intersection or compactness coordinate is available from the carrier. □

Theorem 1713.3 (Located real nested intervals point-name route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing point-name read of an accepted* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *carrier factors through the finite route*

$$I \longmapsto B, M \longmapsto D, J \longmapsto S, R \longmapsto E, H, C, P, N.$$

Thus the read first consumes the located interval source, then the finite bisection and midpoint rows, then the diameter-contraction and nested-containment rows, then the $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no host interval intersection, selected quotient representative, open-cover compactness, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$. The only source interval coordinate is I, so endpoint, radius, locator, and locatedness observations begin at the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ row.

The next stage is B, M. The bisection row records the retained finite branch route, while the midpoint row supplies the child-cell handoff for each retained branch.

The diameter and containment stage is D, J. These rows state that the retained intervals contract at the requested precisions and that each retained row is nested inside the previous displayed row.

The readback stage is S, R. The stream windows are taken from the retained endpoint windows fixed by D, J, and the regular rational row repacks exactly those windows as the point-name readback.

Finally the seal stage is E, H, C, P, N. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the regular readback, and the structural rows transport, replay, package, and name the same finite route. Since the carrier has no host intersection, quotient-representative selector, compactness, choice, or completeness coordinate, the displayed route is exhaustive. □

Definition 1713.4 (Located real nested intervals classifier)**.** *For an accepted* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *carrier* $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$*, the local classifier is the finite predicate that admits exactly the reads transported through the following rows:*

1. *the source transported located interval row* I*;*
2. *the bisection branch row* B *and midpoint row* M*;*
3. *the diameter-budget row* D *and nested-containment row* J*;*
4. *the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*;*
5. *the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*;*
6. *the transport, replay, provenance, and local naming rows* H, C, P, N*.*

The classifier rejects every read that bypasses one of these rows or asks for a host interval intersection, a selected quotient representative, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1713.5 (Located real nested intervals NameCert obligations)**.** *For every accepted* LocatedRealNestedInte *carrier* $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealNestedIntervals}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located interval row, bisection row, midpoint row, diameter budget, containment row, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *classifier stability under componentwise* hsame *transport of* I, B, M, D, J, S, R, E*;*
3. *diameter contraction for every retained interval row exposed by* J*;*
4. *nested containment from the source located interval through the retained branch and midpoint rows;*
5. $\mathsf{StreamName}^{\uparrow}$ *window exposure using only the retained endpoint windows fixed by* D, J*;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *readback from the same retained windows;*
7. $\mathsf{Real}^{\uparrow}$ *seal nonescape through* S, R, E*;*
8. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
9. *ledger refusal for any consumer that omits a located interval row, bisection row, midpoint row, diameter budget, containment row, stream window, regular readback, Real seal, transport row, replay row, provenance row, or local naming row.*

Proof. Project the carrier of Definition 1713.1. This gives the displayed rows $I, B, M, D, J, S, R, E, H, C, P, N$ and no additional host-level intersection, quotient, compactness, choice, or completeness coordinate.

The classifier is stable because Definition 1713.4 is defined rowwise from the transported located interval, bisection, midpoint, diameter, containment, stream, readback, and seal rows. Componentwise hsame transport therefore preserves exactly the admitted reads.

The diameter and containment obligations are the two gates of Definition 1713.2. Every retained interval window downstream of J first reads the budget row D, then the containment row J.

The readback and seal obligations are chained through Definition 1713.3. The route reaches S, R only after D, J, and it reaches E only after the StreamName and RegSeqRat rows have been displayed.

Finally H, C, P, N transport, replay, package, and name the same finite route. Any consumer missing one of the listed rows fails the local classifier and is refused by the NameCert ledger. □

Theorem 1713.6 (Located real nested intervals Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read accepted by the* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *classifier passes through*

$$I, B, M, D, J, S, R, E, H, C, P, N.$$

In particular, no accepted read reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal from a host interval-intersection object, a selected quotient representative, an open-cover compactness proof, a countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a read be accepted by the classifier of Definition 1713.4. By definition, the read is classified only when it is transported through the source located interval, bisection, midpoint, diameter-budget, containment, stream-window, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

Apply Definition 1713.2 to the retained interval stage. The read cannot reach S before the contraction row D and containment row J have been displayed.

Apply Definition 1713.3 to the terminal route. The read reaches E only after I, B, M, D, J, S, R and remains packaged by H, C, P, N.

The carrier of Definition 1713.1 contains no alternate coordinate for host intersection, quotient selection, compactness, choice, or completeness. Hence every accepted $\mathsf{Real}^{\uparrow}$-facing read is forced through the displayed finite route. □

Theorem 1713.7 (Located real nested intervals Cauchy witness route)**.** *For every accepted* LocatedRealNestedInter *carrier* $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$*, the diameter-budget row* D *and nested-containment row* J *force each requested precision through the finite witness route*

$$D, J \longmapsto S, R \longmapsto E, H, C, P, N.$$

Consequently the packet supplies a regular-Cauchy witness route into $\mathsf{RegSeqRat}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal, rather than only a sealed Real-name row.*

Proof. Apply Definition 1713.2. It places every retained interval window downstream of the displayed diameter budget D and nested-containment row J.

Next apply Definition 1713.3. The point-name route sends the retained endpoint windows through S and repacks them through the $\mathsf{RegSeqRat}^{\uparrow}$ row R before the Real seal E can be read.

Finally use Definition 1713.6. The seal remains downstream of D, J, S, R and is transported only through H, C, P, N, so each requested precision carries an explicit finite Cauchy witness route into the terminal $\mathsf{Real}^{\uparrow}$ boundary. □

Theorem 1713.8 (Located nested intervals as a constructive IVT modulus substrate)**.** *If an accepted* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *packet is used as the bisection substrate for a* $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ *request, then the IVT modulus route must pass through the same diameter row* D*, containment row* J*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *named by the located nested-interval carrier. In particular, the substrate contributes a finite window route*

$$D, J \longmapsto S, R \longmapsto E$$

and not a host interval intersection, selected root, quotient real equality, open-cover compactness, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted located nested-interval carrier $Z = (I, B, M, D, J, S, R, E, H, C, P, N)$. The rows D and J are the displayed diameter-budget and containment gates of the retained interval route.

Apply Definition 1713.7. It forces every requested precision through D, J before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows are read.

Now apply Definition 6431.3 to the consuming $\mathsf{ConstructiveIVTModulus}^{\uparrow}$ request. The IVT modulus bisection route opens only the finite windows supplied by its substrate and then proceeds through regular readback and the terminal Real seal.

Combining the two routes identifies the substrate rows used by the modulus request as the same D, J, S, R, E already named in the located nested-interval carrier. Since neither route carries a host intersection, root selector, quotient equality, compactness proof, choice schedule, or completeness coordinate, the conclusion follows by finite-route exhaustion. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealNestedIntervals}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealNestedIntervals}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1713.1, Definition 1713.2, and Definition 1713.3 seed the finite located nested-interval bridge from located interval bisection and midpoint rows to StreamName, RegSeqRat, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealNestedIntervals}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval intersection, selected quotient representatives, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, diameter contraction, nested containment, StreamName window exposure, RegSeqRat readback, Real seal nonescape, and point-name route proofs for BEDC.Derived.LocatedRealNestedI

Constructive story: A $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ packet is generated from a source LocatedRealIntervalUp row, LocatedIntervalBisectionUp branch rows, LocatedIntervalMidpointUp child-cell rows, dyadic diameter budgets, nested containment witnesses, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1714

A Concrete Naming Certificate for $\mathsf{SeparableCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki-style separability, the $\mathsf{SeparableCompletion}^{\uparrow}$ packet records the finite L10 route by which a separable metric source, an enumerated dense schedule, a completion seal, finite stream windows, dyadic or Real tolerances, and regular rational readbacks are offered to completion-facing consumers. It sits between the metric and separability surfaces of chapter 57, chapter 1458, and chapter 1566; the completion interfaces of chapter 1506, chapter 128, and chapter 1687; and the finite observation spine of chapter 126, chapter 291, chapter 301, and chapter 38. The packet names a completion interface controlled by displayed countable density data, not a quotient-completion construction, a selected sequence of dense points, countable choice, host equality of completion points, or ambient Real completeness.

Definition 1714.1 (SeparableCompletion carrier)**.** *A* $\mathsf{SeparableCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, D, C, W, T, R, E, H, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{SeparableMetric}^{\uparrow}$ *source row. It records the metric carrier, distance classifier, and separable-metric countable-name surface over the same point histories.*
2. *D is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *schedule row. It stores the unary indexed dense candidates and the finite request windows through which density is read.*
3. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row. It records the completion-facing admissibility route and the displayed limit or completion-seal handoff.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect dense indices, Cauchy windows, and completion-facing name segments.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance row carrying the dyadic scale, Real distance boundary, and comparison ledger attached to the window W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed dense-window and tolerance data before the completion seal is consulted.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary or completion-seal row reached only after M, D, C, W, T, R have been displayed.*

8. *H records componentwise* hsame *transport for the metric and separable source, dense schedule, completion seal, finite windows, tolerance row, readback, terminal seal, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* SeparableCompletion$^\uparrow$, Met$^\uparrow$, SeparableMetric$^\uparrow$, CountableDenseSubset$^\uparrow$, MetricCompletion$^\uparrow$, CompleteMetric$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{SeparableCompletion}^\uparrow}$ *row.*

The carrier predicate asserts exactly that these rows form one finite separable-completion packet. It has no coordinate for quotient-completion equality, host equality of completed points, a selected dense sequence, countable choice, open-cover compactness, or ambient completeness of Real$^\uparrow$.

Lean4 ✓ `BEDC.Derived.SeparableCompletionUp.SeparableCompletionUp single carrier alignment`

Theorem 1714.2 (SeparableCompletion NameCert obligations)**.** *For every accepted* SeparableCompletion$^\uparrow$ *carrier $S = (M, D, C, W, T, R, E, H, P, N)$, the local* NameCert$_{\text{SeparableCompletion}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier exposure for the displayed metric and separable source, countable dense schedule, completion seal, stream windows, tolerance ledger, regular rational readback, terminal seal, componentwise transport, provenance, and local naming rows;*
2. *classifier transport, requiring comparisons of accepted packets to transport M, D, C, W, T, R, E together before any completion-facing read is exposed;*
3. *dense-window admission, requiring each approximation request to pass through the indexed dense schedule D and the finite* StreamName$^\uparrow$ *row W;*
4. *completion-seal handoff, requiring the* MetricCompletion$^\uparrow$ *or* CompleteMetric$^\uparrow$ *row C to be read only after the metric source and dense-window route are displayed;*
5. *approximation-route exactness, requiring the tolerance row T and regular readback row R to mediate every finite approximation before the terminal seal E is inspected;*
6. *nonescape, excluding quotient completion equality, host equality, selected dense points, countable-choice selectors, open-cover compactness, and ambient Real completeness from the packet.*

No separable-completion obligation is stored outside these rows.

Proof. Project the carrier of Definition 1714.1. The projection exposes exactly the metric and separable source row M, countable dense schedule D, completion seal C, finite windows W, tolerance ledger T, regular readback R, terminal seal E, and structural rows H, P, N.

Classifier transport is componentwise. The row H transports the metric source, dense schedule, completion seal, windows, tolerance ledger, readback, and terminal seal together, so a comparison cannot replace the dense schedule while keeping the same completion-facing packet.

Dense-window admission comes from D and W. A consumer sees a dense candidate only by reading the displayed unary index schedule and the finite stream window attached to the request.

The completion-seal handoff is downstream of that route. The row C records only the displayed MetricCompletion$^\uparrow$ or CompleteMetric$^\uparrow$ admission path, and the rows T and R mediate the approximation window before E can be inspected.

Finally, P and N package and name the same finite route. Since none of the carrier coordinates stores quotient-completion equality, host equality, selected dense points, countable choice, open-cover compactness, or ambient Real completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SeparableCompletionUp.SeparableCompletionCarrier namecert obligations`

Theorem 1714.3 (SeparableCompletion density handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{SeparableCompletion}^{\uparrow}$ *carrier* $S = (M, D, C, W, T, R, E, H, P, N)$ *factors through the finite route*

$$M, D \longmapsto W \longmapsto T \longmapsto R \longmapsto C \longmapsto E \longmapsto H, P, N.$$

Thus the consumer reads the countable dense schedule, finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and completion seal before it reads the public packet. The route exports no quotient completion equality, no host equality of completed points, no selected dense point, no countable-choice selector, and no ambient Real-completeness datum.*

Proof. Begin with a completion-facing read of S. Carrier admission gives the consumer access to the source row and dense schedule only as the displayed pair M, D.

Replay the finite observation route. The request opens a $\mathsf{StreamName}^{\uparrow}$ window W over the dense schedule, reads the dyadic or Real tolerance row T, and obtains the regular rational handoff R before the completion seal C is available.

The terminal row E is downstream of the same route. Transport, provenance, and local naming through H, P, N preserve the ordered dependency chain rather than adding a second completion object.

Therefore every completion-facing consumer factors through the countable dense schedule, finite stream window, tolerance ledger, regular readback, completion seal, and terminal boundary before it reads the public packet. The excluded quotient, host-equality, selected-point, choice, and ambient-completeness data have no coordinate in the carrier. □

Lean4 ✓ `BEDC.Derived.SeparableCompletionUp.SeparableCompletionDensityHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparableCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1714.1, Definition 1714.2, and Definition 1714.3 seed the finite separable-completion route through metric and separable source data, countable dense schedules, stream windows, dyadic or Real tolerances, regular rational readbacks, and completion seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparableCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected dense points, countable choice, host equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier transport, dense-window admission, completion-seal handoff, approximation-route exactness, nonescape, and density handoff for BEDC.Derived.SeparableCompletionUp.

Constructive story: A $\mathsf{SeparableCompletion}^{\uparrow}$ packet is generated from a MetricUp and SeparableMetricUp source row, a CountableDenseSubsetUp schedule, a MetricCompletionUp or CompleteMetricUp seal, finite StreamName windows, DyadicRatCore or RealUp tolerance data, a RegSeqRat readback, a terminal RealUp or completion seal, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Chapter 1715

A Concrete Naming Certificate for $\mathsf{CompletionPreservesSeparable}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley separability tradition, the $\mathsf{CompletionPreservesSeparable}^{\uparrow}$ packet records the finite BEDC preservation bridge from a separable source metric to separable completion data. It reads the source surfaces of chapter 1458 and chapter 1566, the existing route of chapter 1714, and the completion interface of chapter 1506. Its finite observation spine is the L10 route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the bridge preserves displayed density through windows and tolerance ledgers rather than through quotient equality, countable choice, selected representatives, or host equality of completed points.

Definition 1715.1 (CompletionPreservesSeparable carrier)**.** *A* $\mathsf{CompletionPreservesSeparable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, D, C, W, T, R, I, H, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{SeparableMetric}^{\uparrow}$ *row. It records the metric carrier, distance classifier, and countable-name separability surface over the source histories.*
2. *D is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *row. It stores the indexed dense schedule and the finite request windows through which source density is observed.*
3. *C is the* $\mathsf{SeparableCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *row. It records the completion-facing admission route reached from the same source metric and the same dense schedule.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect dense indices, source Cauchy observations, and completion-facing name segments.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance ledger attached to W. It carries dyadic scales, Real distance boundaries, and comparison budgets for the dense-image request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from D, W, T before the completion seal is consulted.*
7. *I is the dense-image handoff row. It records that a completion request is answered by first exposing an indexed source dense point through D, W, T, R and then passing through the completion seal C.*

8. *H records componentwise* hsame *transport for the source metric, dense schedule, completion row, stream windows, tolerance ledger, readback, image handoff, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompletionPreservesSeparable}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{SeparableCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionPreservesSeparable}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite preservation packet. It has no coordinate for quotient completion equality, selected dense representatives, countable choice, host equality of completion points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionPreservesSeparableUp.CompletionPreservesSeparableTasteGate`
`single carrier alignment`

Theorem 1715.2 (CompletionPreservesSeparable NameCert obligations)**.** *For every accepted* $\mathsf{CompletionPreservesSeparable}^{\uparrow}$ *carrier* $Q = (M, D, C, W, T, R, I, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionPreservesSeparabl}}$ *surface consists exactly of the following obligation rows:*

1. *carrier exposure for the displayed source metric and separable row, countable dense schedule, separable-completion and metric-completion row, stream windows, tolerance ledger, regular readback, dense-image handoff, transport, provenance, and local naming rows;*

2. *source-density admission, requiring each completion-density request to start from the displayed countable dense schedule D over the source metric row M;*

3. *finite-window inspection, requiring dense indices and completion-facing name segments to pass through the same* $\mathsf{StreamName}^{\uparrow}$ *row W;*

4. *tolerance and readback discipline, requiring* $\mathsf{DyadicRatCore}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison data in T and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R before the completion row is exposed;*

5. *completion-image handoff, requiring the row I to bind the selected source dense observation to the completion route C without storing a host-level completed point;*

6. *classifier transport, requiring comparisons of accepted packets to transport M, D, C, W, T, R, I together through H;*

7. *nonescape, excluding quotient completion equality, selected dense representatives, countable-choice selectors, host equality of completion points, open-cover compactness, and ambient Real completeness from the packet.*

No completion-preservation obligation is stored outside these rows.

Proof. Project the carrier of Definition 1715.1. The projection exposes exactly the source metric row M, countable dense schedule D, completion row C, finite windows W, tolerance ledger T, regular readback R, dense-image handoff I, and structural rows H, P, N.

Source-density admission is read before the completion row. A density request first displays an index in D over the same source metric row M, then opens the finite $\mathsf{StreamName}^{\uparrow}$ window W attached to that request.

The tolerance and readback rows mediate the request. The packet reads the dyadic or Real comparison data T and the regular rational handoff R before it exposes the completion-facing row C.

The dense-image handoff is the row I. It binds the displayed source dense observation to the completion route without adding a selected completed representative, quotient equality, or host-level equality of completion points.

Finally, H, P, N transport, source, and name the same finite route. Since no carrier coordinate stores the excluded quotient, choice, host-equality, compactness, or ambient-completeness data, the listed rows exhaust the local NameCert surface. □

Theorem 1715.3 (CompletionPreservesSeparable dense-image handoff)**.** *Every complete-separable or completion-facing consumer of an accepted* $\mathsf{CompletionPreservesSeparable}^{\uparrow}$ *carrier* $Q = (M, D, C, W, T, R, I, H, P$ *reads the finite route*

$$M, D \longmapsto W \longmapsto T \longmapsto R \longmapsto I \longmapsto C \longmapsto H, P, N$$

before it reads the public packet. Thus separability of the completion is exported through displayed source dense data, finite windows, tolerance checks, regular readback, and the dense-image handoff. The route exports no quotient completion equality, no selected dense representative, no countable-choice selector, no host equality of completed points, and no ambient Real-completeness datum.

Proof. Begin with a complete-separable or completion-facing read of Q. Carrier admission gives the consumer the source metric and source dense schedule only as the displayed pair M, D.

Replay the finite observation route. The request opens the $\mathsf{StreamName}^{\uparrow}$ window W, reads the dyadic or Real tolerance ledger T, and obtains the $\mathsf{RegSeqRat}^{\uparrow}$ readback R before any completion-facing image is available.

The image row I is downstream of those source observations. It records the handoff from the displayed source dense point to the completion route C, so the consumer sees density in the completion only through the same finite source data.

Transport, provenance, and local naming through H, P, N preserve that route rather than adding a second completion object. Therefore every downstream consumer factors through source density, finite observation, tolerance, readback, image handoff, and completion admission before it reads the public packet.

The excluded quotient, selected-representative, choice, host-equality, and ambient-completeness data have no coordinate in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionPreservesSeparable}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionPreservesSeparable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1715.1, Definition 1715.2, and Definition 1715.3 seed the finite preservation bridge from source separability to separable completion data through dense schedules, stream windows, dyadic or Real tolerances, regular readbacks, and dense-image handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionPreservesSeparable}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected dense representatives, countable choice, host equality of completion points, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-density admission, finite-window inspection, tolerance and readback discipline, completion-image handoff, classifier transport, nonescape, and dense-image handoff for BEDC.Derived.CompletionPreservesSeparableUp.

Constructive story: A $\mathsf{CompletionPreservesSeparable}^{\uparrow}$ packet is generated from a MetricUp and SeparableMetricUp source row, a CountableDenseSubsetUp schedule, a SeparableCompletionUp and MetricCompletionUp row, finite StreamName windows, DyadicRatCore or RealUp tolerance data, a RegSeqRat readback, a dense-image handoff row, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Chapter 1716

A Concrete Naming Certificate for $\mathsf{CauchyRealOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealOrder}^{\uparrow}$ packet names the constructive order and apartness surface for Cauchy real data by displaying the finite comparison route before any quotient equality or trichotomy principle is available. It sits after the stream, dyadic, regular-rational, real-name, quotient-boundary, and located-interval surfaces of chapter 126, chapter 291, chapter 301, chapter 38, chapter 1708, and chapter 1709. The packet gives Real and Completion consumers one named ordered-real route; it is not a quotient real equality theorem, a representative selector, a trichotomy oracle, a countable-choice schedule, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 1716.1 (Cauchy real order carrier)**.** *A* $\mathsf{CauchyRealOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, T, W, D, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S and T are the two source rows. Each is read through* $\mathsf{StreamName}^{\uparrow}$ *windows and a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff before it is sealed as a* $\mathsf{Real}^{\uparrow}$ *name.*
2. *W is the common finite-window row. It records the shared observation lengths, synchronised stream cuts, and comparison windows used for both source rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and sign ledger. It stores the dyadic radii, positive gap requests, sign cells, and slack bookkeeping consumed by apartness and order reads.*
4. *Q is the quotient-boundary row. It records that equality-sensitive reads remain inside the displayed Cauchy quotient boundary and are not replaced by host quotient equality.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the two accepted real names after S, T, W, D, Q have been displayed.*
6. *A is the apartness and order verdict row. It records a positive apartness route, a negative apartness route when the displayed finite windows do not certify a positive gap, and the oriented order verdict exported to consumers.*
7. *H records componentwise* hsame *transport for the two source rows, common windows, dyadic tolerance ledger, quotient-boundary row, real seals, verdict row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each order or apartness request through $S, T, W, D, Q, R, A, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyQuotientBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealOrder}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive ordered-real comparison packet. It has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRealOrderUp.CauchyRealOrderUpTasteGate single carrier alignment`

Theorem 1716.2 (Cauchy real order NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealOrder}^{\uparrow}$ *carrier* $O = (S, T, W, D, Q, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealOrder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two source rows, common finite windows, dyadic tolerance and sign ledger, quotient-boundary row, real seal rows, verdict row, transport, replay, provenance, and local naming row;*
2. *classifier exactness, requiring accepted packets to be compared by the displayed stream windows, regular-rational readbacks, dyadic tolerance ledger, real seal rows, and componentwise transport;*
3. *positive apartness route, requiring every positive verdict to exhibit a finite dyadic gap certified by W, D, Q, R, A;*
4. *negative apartness route, requiring every refusal of positive apartness to record the exhausted finite windows and dyadic slack rather than a quotient-equality proof;*
5. *order transport, requiring H and C to replay the oriented verdict along componentwise* hsame *witnesses;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring both source rows to pass through regular-rational readback before the real seal and verdict rows are inspected;*
7. *real seal non-escape, requiring R to expose only the displayed* $\mathsf{Real}^{\uparrow}$ *rows and to export no selected representative, quotient equality, or ambient completeness datum;*
8. *ledger exhaustion, excluding trichotomy, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Cauchy real order obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyRealOrderUp.TasteGate.CauchyRealOrderTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1716.1. The projection exposes exactly the two source rows, common window row, dyadic tolerance ledger, quotient-boundary row, real seal row, verdict row, and structural rows.

Classifier exactness is componentwise. The comparison first reads the two stream and regular-rational source rows through S, T, W, then consumes the dyadic sign ledger D, quotient-boundary row Q, real seal row R, and verdict row A before transport or replay can package the result.

The positive apartness route is the row of certified dyadic gaps. A positive verdict must display a finite pair of windows, a dyadic tolerance request, and a sign cell whose slack remains positive after the recorded refinements.

The negative apartness route is also finite. It records that the displayed windows and dyadic slack do not certify a positive gap at the requested precision. This is a refusal of the positive route, not a proof of quotient equality.

Order transport uses H and C. The componentwise hsame row transports the source, window, tolerance, seal, and verdict data together, while the replay row preserves the order in which a consumer is allowed to inspect them. The remaining provenance and naming rows therefore exhaust the local certificate without exporting any excluded principle. □

Lean4 ✓ `BEDC.Derived.CauchyRealOrderUp.CauchyRealOrderCarrier namecert obligations`

Theorem 1716.3 (Cauchy real order RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyRealOrder}^{\uparrow}$ *carrier factors each order or apartness request through the route*

$$S, T \longmapsto W \longmapsto D \longmapsto Q \longmapsto R \longmapsto A \longmapsto H, C, P, N.$$

In particular, both Cauchy real source rows pass through $\mathsf{RegSeqRat}^{\uparrow}$ *readback and common finite windows before any* $\mathsf{Real}^{\uparrow}$ *seal or oriented verdict is exported.*

Proof. Start with an accepted packet $O = (S, T, W, D, Q, R, A, H, C, P, N)$. The only source coordinates are S and T, and the carrier requires them to be inspected through stream windows and regular-rational readback.

The common row W synchronises those observations. The tolerance and sign ledger D then records the dyadic scale at which a consumer asks for comparison data, while Q keeps equality-sensitive reads at the quotient-boundary surface.

The real seal R is downstream of this displayed route. The verdict row A can therefore see only the readbacks and tolerances already named by S, T, W, D, Q, R.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no earlier row contains a selected representative or host equality witness, the handoff route is exhaustive. □

Theorem 1716.4 (Cauchy real order Real seal non-escape)**.** *For every accepted* $\mathsf{CauchyRealOrder}^{\uparrow}$ *carrier* $O = (S, T, W, D, Q, R, A, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R *can be used only through the displayed comparison route* $S, T, W, D, Q, A, H, C, P, N$*. Hence a consumer of the order surface cannot extract trichotomy, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from* R*.*

Lean4 ✓ `BEDC.Derived.CauchyRealOrderUp.CauchyRealOrderCarrier real seal nonescape`

Proof. Inspect the carrier coordinates. The row R is recorded as the terminal real-name seal for the two source rows after their stream windows, regular-rational readbacks, dyadic tolerances, and quotient-boundary data have been displayed.

Any order consumer must then pass through A. The verdict row records only positive apartness data, negative apartness refusal data at a finite request, and the oriented order verdict tied to the same dyadic sign ledger.

Transport and replay do not add analytic content. The rows H and C preserve the displayed dependencies, while P and N package and name them. Therefore the real seal has no path to export any excluded principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1716.1, Definition 1716.2, Definition 1716.3, and Definition 1716.4 seed the finite constructive order and apartness route through two stream-backed regular-rational source rows, common windows, a dyadic tolerance and sign ledger, quotient-boundary data, real seals, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealOrder}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyRealOrderUp.CauchyRealOrderUpTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, positive and negative apartness routes, order transport, RegSeqRat handoff, real seal non-escape, and ledger exhaustion for BEDC.Derived.CauchyRealOrderUp.

Constructive story: A CauchyRealOrder$^{\uparrow}$ packet is generated from two StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ source rows, a common finite-window row, a DyadicRatCore$^{\uparrow}$ tolerance and sign ledger, a CauchyQuotientBoundary$^{\uparrow}$ row, terminal Real$^{\uparrow}$ seals, an apartness and order verdict row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1717

A Concrete Naming Certificate for $\mathsf{CauchyRealField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealField}^{\uparrow}$ packet names the finite field-facing surface for Cauchy real data by joining the regular-Cauchy operation rows and constructive order rows before later Real and Completion consumers read them as one algebraic interface. It sits after the real-name boundary of chapter 38, the regular rational source surface of chapter 301, the stream-window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the addition surface of chapter 1695, the product surface of chapter 461, the reciprocal surface of chapter 1471, and the constructive order surface of chapter 1716. The packet gives Real and Completion consumers a single finite field route; it is not host real arithmetic, quotient equality, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1717.1 (Cauchy real field carrier)**.** *A* $\mathsf{CauchyRealField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, S, D, Z, A, M, I, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source family row for the regular rational Cauchy names exposed to field-facing consumers.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window discipline used to synchronize every algebraic and order read of the source family.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic and tolerance ledger. It records the dyadic additions, products, reciprocal denominator bounds, sign cells, and slack budgets consumed by the operation rows.*
4. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row for the accepted real names reached only after R, S, D and the relevant operation or order row have been displayed.*
5. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *operation row, including the paired source windows, dyadic endpoint sum, finite addition ledger, and summed readback.*
6. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *operation row, including the paired source windows, dyadic endpoint product, product-error ledger, and product readback.*
7. *I is the* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *partial reciprocal row, including the apartness window, reciprocal budget, readback row, and terminal real seal for inputs whose displayed order row keeps the denominator away from zero.*

8. *O is the* CauchyRealOrder$^{\uparrow}$ *apartness and order row used to expose positive-gap witnesses, refusal of positive apartness at a finite request, and oriented order verdicts.*

9. *H records componentwise* hsame *transport for the source, window, dyadic, seal, addition, product, reciprocal, order, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from each field-facing request through $R, S, D, Z, A, M, I, O, H, P, N$.*

11. *P is the* Pkg *provenance over* CauchyRealField$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, RegularCauchyReciprocal$^{\uparrow}$, CauchyRealOrder$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CauchyRealField}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive field-facing packet. It has no coordinate for host real arithmetic, quotient equality, trichotomy, selected representatives, countable choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldTasteGate single carrier alignment`

Theorem 1717.2 (Cauchy real field NameCert obligations)**.** *For every accepted* CauchyRealField$^{\uparrow}$ *carrier*

$$F = (R, S, D, Z, A, M, I, O, H, C, P, N),$$

the local NameCert$_{\text{CauchyRealField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational source family, synchronized stream windows, dyadic arithmetic and tolerance ledger, real seal row, addition row, product row, reciprocal row, order row, transport, replay, provenance, and local naming row;*

2. *addition projection, requiring the A row to be read through the accepted* RegularCauchyAddition$^{\uparrow}$ *source windows, dyadic endpoint sum, finite addition ledger, regular readback, and Real seal handoff;*

3. *product projection, requiring the M row to be read through the accepted* RegularCauchyProduct$^{\uparrow}$ *source windows, dyadic endpoint product, finite product-error ledger, regular readback, and Real seal handoff;*

4. *reciprocal projection, requiring the I row to be read only after the displayed apartness window and reciprocal budget of* RegularCauchyReciprocal$^{\uparrow}$ *have been exposed;*

5. *order projection, requiring the O row to supply the finite positive-gap, negative-apartness, and oriented-order verdict routes of* CauchyRealOrder$^{\uparrow}$*;*

6. *shared seal discipline, requiring Z to be downstream of the same R, S, D rows and the selected operation or order row rather than an independent completed-real carrier;*

7. *transport and replay, requiring H and C to preserve the componentwise source, window, ledger, operation, order, seal, provenance, and local naming rows together;*

8. *ledger exhaustion, excluding host real arithmetic, quotient equality, trichotomy, selected representatives, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Cauchy real field obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1717.1. The projection exposes the regular rational source row, synchronized stream windows, dyadic ledger, real seal, three operation rows, order row, and structural rows.

The operation projections are inherited from the sibling surfaces. The addition row uses the finite addition route of chapter 1695; the product row uses the finite product route of chapter 461; and the reciprocal row uses the apartness-window route of chapter 1471. Each projection keeps its source windows, dyadic arithmetic, regular readback, and seal handoff visible.

The order projection is the comparison route of chapter 1716. It supplies the positive-gap, negative-apartness, and oriented-order rows used by reciprocal eligibility and ordered-field consumers.

Finally, the rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for the excluded principles, none can become a local NameCert obligation. □

Theorem 1717.3 (Cauchy real field operation composition)**.** *Every accepted* $\mathsf{CauchyRealField}^{\uparrow}$ *carrier routes addition, multiplication, and reciprocal reads through the common finite chain*

$$R \longmapsto S \longmapsto D \longmapsto (A, M, I) \longmapsto Z \longmapsto H, C, P, N.$$

The reciprocal branch additionally passes through the order row O before the reciprocal row I is allowed to expose a readback.

Lean4 ✓ `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldCarrier operation composition`

Proof. Start from the accepted packet $F = (R, S, D, Z, A, M, I, O, H, C, P, N)$. The only regular rational source family is R, and every operation read is synchronized by the same stream-window row S before the dyadic ledger D is consumed.

Project the addition and product branches. The rows A and M are the only addition and product coordinates, so their readbacks are exactly the sibling regular-Cauchy operation routes named in the carrier.

For the reciprocal branch, first read O. Its apartness route supplies the positive denominator information required by the reciprocal row I, and a missing positive-gap witness leaves the reciprocal route at the boundary surface rather than exporting an inverse.

The real seal and structural rows are downstream. The row Z is reached only after the selected branch has been displayed, and H, C, P, N preserve this dependency order for every field-facing consumer. □

Theorem 1717.4 (Cauchy real field order-compatible reciprocal boundary)**.** *For every accepted* $\mathsf{CauchyRealField}^{\uparrow}$ *carrier, a reciprocal read from I can be consumed only after the* $\mathsf{CauchyRealOrder}^{\uparrow}$ *row O displays a finite apartness route for the denominator source. Therefore the field surface exports a partial reciprocal boundary tied to constructive order data, not a total reciprocal operation on* $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldCarrier order compatible reciprocal boundary`

Proof. Inspect the carrier coordinates. The reciprocal row I is listed as a partial reciprocal row whose apartness window is supplied by the operation packet and whose eligibility is tied to the order row O.

Use the sibling reciprocal surface. chapter 1471 requires an apartness window before any reciprocal dyadic row, budget row, readback row, or Real seal can be consumed.

Use the sibling order surface. chapter 1716 is the only row in the field packet that exports positive-gap data for real-name comparisons. Hence a reciprocal read without O has no displayed denominator-apartness source.

The remaining rows only transport and package this route. They cannot turn the partial boundary into total completed-real reciprocal data, because no such coordinate is present in the carrier. □

Theorem 1717.5 (Cauchy real field Real and Completion consumer boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyRealField}^{\uparrow}$ *packet reads one finite public surface consisting of* $R, S, D, Z, A, M, I, O, H, C, P, N$*. The consumer may project the addition, product, reciprocal, or order branch, but it receives no separate operation packet outside the field surface and no host real arithmetic, quotient equality, trichotomy, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. The consumer begins with the field carrier, not with independent operation packets. Projecting A, M, I, or O only exposes rows already named by Definition 1717.1.

The shared source and seal rows keep the projections tied together. Addition, product, reciprocal, and order reads all pass through the same R, S, D boundary before reaching Z or the structural rows.

The structural rows do not add analytic content. They transport, replay, source, and name the displayed finite packet. Consequently the Real and Completion consumer receives a unified NameCert surface rather than separate host-level arithmetic or completeness data. □

Lean4 ✓ `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldRealCompletionConsumerBoundary`

Theorem 1717.6 (Cauchy real field scoped operation route)**.** *Every field-facing read of an accepted* $\mathsf{CauchyRealField}^{\uparrow}$ *packet is exhausted by the concrete rows*

$\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauc}$

The route reads the regular rational source row, synchronized finite windows, dyadic arithmetic ledger, addition row, product row, reciprocal row, order row, and terminal $\mathsf{Real}^{\uparrow}$ *seal before any Real or Completion consumer receives the field surface. It exports no host real arithmetic, quotient equality, trichotomy, selected representatives, or ambient completeness coordinate.*

Proof. Project the accepted carrier. The only source, window, ledger, seal, operation, and order rows are R, S, D, Z, A, M, I, O, with H, C, P, N preserving and naming the same finite packet.

Use operation composition. It places addition, product, and reciprocal reads behind the common R, S, D route, with the reciprocal branch additionally reading the order row before the reciprocal row is available.

Use the order-compatible reciprocal boundary and the Real and Completion consumer boundary. These theorems keep reciprocal eligibility tied to constructive order data and keep downstream consumers inside the unified packet.

Thus the algebraic surface is exactly the displayed L10 operation route: RealUp seal, RegSeqRat source, StreamName windows, DyadicRatCore ledger, regular-Cauchy addition, product, reciprocal, and Cauchy-real order rows, with no host arithmetic or ambient completeness export. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealField}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1717.1, Definition 1717.2, Definition 1717.3, Definition 1717.4, and Definition 1717.5 seed the finite field-facing route over regular rational source rows, stream windows, dyadic arithmetic, real seals, regular-Cauchy addition, regular-Cauchy product, regular-Cauchy reciprocal, constructive order, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealField}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyRealFieldUp.CauchyRealFieldTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host real arithmetic, quotient equality, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation projections, reciprocal eligibility, order projection, shared seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRealFieldUp.

Constructive story: A $\mathsf{CauchyRealField}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source family, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic ledger, a $\mathsf{Real}^{\uparrow}$ seal row, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, and $\mathsf{CauchyRealOrder}^{\uparrow}$ rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1718

A Concrete Naming Certificate for $\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}$ packet records the finite regular-Cauchy boundary for the distributive-law expression $x \cdot (y+z)$ and $x \cdot y + x \cdot z$ before any completed real consumer reads a field law. It sits over the two-source addition surface of chapter 417, the two-source product surface of chapter 461, and the field-facing operation package of chapter 1717. The packet gives Real and Completion routes a finite BEDC-internal distributivity obligation; it is not host real equality, quotient Cauchy classes, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1718.1 (Regular Cauchy field distributivity carrier)**.** *A* $\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, Z, W, D_X, D_Y, D_Z, S_L, M_L, M_Y, M_Z, S_R, E_L, E_R, R, H, C, P, N)$$

with the following displayed rows:

1. X, Y, Z *are the three* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are read as the variables* x, y, z*.*
2. W *is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite common request at which the three source rows are inspected, and it provides the shared window discipline for both distributive branches.*
3. D_X, D_Y, D_Z *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from* X, Y, Z *through* W*.*
4. S_L *is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *comparison boundary for the displayed source sum* $Y + Z$ *used by the left branch.*
5. M_L *is the first* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *branch, reading* X *and the readback of* S_L *as the product route for* $x \cdot (y + z)$*.*
6. M_Y *and* M_Z *are the two* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *branches for* $x \cdot y$ *and* $x \cdot z$*, each consuming the same* X, W, D_X *row with the corresponding* Y *or* Z *row.*
7. S_R *is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *comparison boundary that adds the two product readbacks from* M_Y *and* M_Z*.*
8. E_L *and* E_R *are finite error-budget ledgers. They record the endpoint addition budget for* $Y + Z$*, the product budgets for* M_L, M_Y, M_Z*, the final sum budget for* S_R*, and the tolerance comparison between the two readback branches.*

9. *R is the terminal* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison readback row, displayed only after both branches have exposed their regular readbacks and the ledgers have matched the requested tolerance.*

10. *H records componentwise* hsame *transport for the three source rows, synchronized windows, dyadic endpoint rows, the two product branches on the right, the left product branch, the two sum boundaries, ledger rows, readback row, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a distributivity request through $X, Y, Z, W, D_X, D_Y, D_Z, S_L, M_L, M_Y, M_Z, S$*

12. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{FiniteErrorBudget}^{\uparrow}$, $\mathsf{CauchyRealField}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite distributivity packet. It has no coordinate for host real equality, completed-real multiplication, quotient Cauchy classes, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFieldDistributivityUp.RegularCauchyFieldDistributivityTasteGate`
`single carrier alignment`

Theorem 1718.2 (Regular Cauchy field distributivity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}$ *carrier*

$$D = (X, Y, Z, W, D_X, D_Y, D_Z, S_L, M_L, M_Y, M_Z, S_R, E_L, E_R, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFieldDistributivity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the three* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, dyadic endpoint rows, one left* $\mathsf{RegularCauchySum}^{\uparrow}$ *boundary, three* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *branches, one right* $\mathsf{RegularCauchySum}^{\uparrow}$ *comparison boundary, finite ledgers, readback, transport, replay, provenance, and local naming;*

2. *window synchronization, requiring all four operation rows S_L, M_L, M_Y, M_Z and the final comparison row S_R to read their source endpoints through the same displayed finite window discipline W;*

3. *left-branch exactness, requiring S_L to expose $Y + Z$ before M_L may expose the product branch $x \cdot (y + z)$;*

4. *right-branch exactness, requiring M_Y and M_Z to expose the two products $x \cdot y$ and $x \cdot z$ before S_R may expose the sum branch $x \cdot y + x \cdot z$;*

5. *ledger comparison, requiring E_L and E_R to account for every addition, product, magnitude, slack, and tolerance-comparison entry used to place the two branch readbacks inside the same requested regular-Cauchy boundary;*

6. *transport and replay stability, requiring H and C to preserve the source rows, windows, endpoint rows, operation rows, ledgers, readback, provenance, and local naming without adding host real equality or completed-real field data;*

7. *field consumer confinement, requiring any* $\mathsf{CauchyRealField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion consumer to read the distributivity obligation only through this finite packet.*

No distributivity obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1718.1. The source, window, dyadic endpoint, operation, ledger, readback, and structural rows are explicit coordinates of the packet.

The branch obligations are inherited from the sibling operation surfaces. Use chapter 417 for S_L and S_R, and use chapter 461 for M_L, M_Y, M_Z. Each operation is read only after its source windows and dyadic endpoint rows have been displayed.

The ledgers compare the two routes at the regular-Cauchy boundary. They record the finite endpoint arithmetic, product magnitude bounds, slack conversions, and output tolerances needed for the common request. The structural rows then transport and replay the same displayed packet. Since no carrier coordinate stores host real equality, quotient classes, selected representatives, countable choice, trichotomy, ambient completeness, Lean verification, or bridge data, none can become a local obligation. □

Theorem 1718.3 (Regular Cauchy field distributivity window handoff)**.** *Every accepted* RegularCauchyFieldDistribu *carrier routes a distributivity request through the finite handoff*

$$X, Y, Z \longmapsto W \longmapsto D_X, D_Y, D_Z \longmapsto S_L, M_L, M_Y, M_Z, S_R \longmapsto E_L, E_R \longmapsto R, H, C, P, N.$$

The left branch reads $Y + Z$ and then multiplies by X; the right branch multiplies X by Y and Z separately and then reads the sum comparison boundary. Both branches share the same source windows, dyadic endpoint rows, finite ledgers, componentwise hsame *transports,* Cont *routes,* Pkg *provenance, and local* NameCert *row.*

Proof. Begin with the three regular sources. The single window coordinate W supplies the synchronized observation request, so every operation branch reads from the same finite source-window boundary.

The left branch first consumes S_L and then M_L. The right branch consumes M_Y and M_Z and then S_R. These are exactly the operation coordinates listed in the carrier, and they are the only available addition and product routes.

Finally read the two ledger rows and the structural rows. The ledgers account for the visible tolerance comparison between branch readbacks, while H, C, P, N preserve and name the displayed packet. No alternate completed-real multiplication or equality route is exposed. □

Theorem 1718.4 (Regular Cauchy field distributivity Real seal boundary)**.** *Every* CauchyRealField$^{\uparrow}$, Real$^{\uparrow}$, *or Completion consumer of an accepted* RegularCauchyFieldDistributivity$^{\uparrow}$ *packet reaches a field-law read only after the three* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows, dyadic endpoint rows, both* RegularCauchyProduct$^{\uparrow}$ *branches on the right, the left* RegularCauchyProduct$^{\uparrow}$ *branch, the* RegularCauchySum$^{\uparrow}$ *comparison boundaries, finite error-budget ledgers,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The consumer receives a finite distributivity obligation and receives no host real equality, quotient stream equality, selected representative, countable-choice selector, trichotomy witness, or ambient* Real$^{\uparrow}$ *completeness datum.*

Proof. Project the consumer read through Definition 1718.3. The consumer cannot reach R before the two branches and the ledger comparison have been displayed.

The field-facing position is supplied by chapter 1717. That chapter collects regular-Cauchy operation rows for downstream Real and Completion consumers; the present packet adds only the finite distributivity boundary between those operation rows.

The excluded data have no source coordinate. Transport, replay, provenance, and local naming preserve the finite packet, and the packet has no row for host equality, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness. □

Theorem 1718.5 (Regular Cauchy field distributivity NameCert surface)**.** *The* RegularCauchyFieldDistributivity$^{\uparrow}$ *NameCert surface is exactly the finite packet of Definition 1718.1 together with the obligation rows of Definition 1718.2, the window handoff of Definition 1718.3, and the Real seal boundary of Definition 1718.4. It is therefore a paper-side theorem target for a future Lean declaration under* $BEDC.Derived.RegularCauchyFieldDistributivityUp$*, not a current Lean marker.*

Proof. The carrier supplies all data-bearing rows. The obligation theorem identifies the local NameCert clauses, the handoff theorem fixes the finite read order, and the Real seal theorem states the downstream consumer boundary.

These components exhaust the packet because every row named by the local NameCert is already one of $X, Y, Z, W, D_X, D_Y, D_Z, S_L, M_L, M_Y, M_Z, S_R, E_L, E_R, R, H, C, P, N$. The statement therefore records a finite BEDC distributivity surface without asserting host real equality or adding a Lean-side marker. □

Closure status for NameCert$_{\text{RegularCauchyFieldDistributivity}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyFieldDistributivity}^{\uparrow}}$, seed)

Scope closed: Definition 1718.1, Definition 1718.2, Definition 1718.3, Definition 1718.4, and Definition 1718.5 seed the finite distributivity boundary over three RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, dyadic endpoint rows, RegularCauchyProduct$^{\uparrow}$ branches, RegularCauchySum$^{\uparrow}$ comparison boundaries, finite error-budget ledgers, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyFieldDistributivity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real equality, completed-real multiplication, quotient Cauchy classes, selected representatives, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window handoff, left-branch exactness, right-branch exactness, ledger comparison, transport, replay, provenance, local naming, and Real seal boundary for BEDC.Derived.RegularCauchyFieldDistributivityUp.

Constructive story: A RegularCauchyFieldDistributivity$^{\uparrow}$ packet is generated from three RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, dyadic endpoint rows, one RegularCauchySum$^{\uparrow}$ left comparison boundary, three RegularCauchyProduct$^{\uparrow}$ branches, one RegularCauchySum$^{\uparrow}$ right comparison boundary, finite error-budget ledgers, a RegSeqRat$^{\uparrow}$ comparison readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1719

A Concrete Naming Certificate for $\mathsf{RaabeDuhamelTest}^{\uparrow}$

Following Raabe and Duhamel, and Bishop–Bridges constructive analysis, the $\mathsf{RaabeDuhamelTest}^{\uparrow}$ packet records the finite first-order ratio-defect test that turns displayed tail estimates into a regular rational convergence route before any Real-facing consumer reads a seal. It sits beside the ratio-test route of chapter 1461, the comparison-test route of chapter 1459, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet supplies a finite convergence-test boundary for the Phase Real and Completion route; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1719.1 (Raabe–Duhamel test carrier)**.** *A* $\mathsf{RaabeDuhamelTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, Q, M, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source series row, exposing only the inspected term windows and partial-sum windows used by the current tail request.*
2. *Q is the positive-tail quotient row, recording the finite adjacent-term quotient windows on which the Raabe–Duhamel defect is read.*
3. *M is the displayed defect-modulus row, assigning each requested tail budget a finite threshold and a rational margin witnessing that the inspected quotient defect stays beyond the chosen comparison boundary.*
4. *D is the dyadic domination ledger, converting the defect and threshold rows into a finite majorant comparison usable by the regular Cauchy comparison-test surface.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the displayed partial sums and comparison majorants.*
6. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes D and W before any regular Cauchy conclusion is exposed.*
7. *E is the endpoint route through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after T has been displayed.*
8. *H stores componentwise* hsame *transport for the source, quotient, defect-modulus, domination, window, comparison, endpoint, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from each tail request through M, D, W, T, E.*

10. *P is the* Pkg *provenance over* $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RaabeDuhamelTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Q, M, D, W, T, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RaabeDuhamelTestUp.RaabeDuhamelTestTasteGate single carrier alignment`

Theorem 1719.2 (Raabe–Duhamel test NameCert obligations)**.** *For every accepted* $\mathsf{RaabeDuhamelTest}^{\uparrow}$ *carrier*

$$R = (A, Q, M, D, W, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RaabeDuhamelTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source series row, adjacent-term quotient row, defect-modulus row, dyadic domination ledger, finite window row, comparison-test handoff, endpoint route, transport, replay, provenance, and local naming row;*

2. *quotient-window exactness, requiring every accepted defect read to pass through the displayed source and adjacent-term quotient windows A, Q;*

3. *defect-modulus stability, requiring the threshold and rational margin in M to be the only data used to choose the tail budget;*

4. *domination routing, requiring D and W to convert the displayed defect data into the same finite comparison windows consumed by T;*

5. *regular and Real handoff, requiring every endpoint-facing read to pass through T and then E before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*

6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 1719.1. Carrier admission is exactly the inventory $A, Q, M, D, W, T, E, H, C$, so the local certificate has no detached row for an infinite sum, quotient series equality, or ambient completeness.

The quotient-window and defect-modulus clauses are read before the comparison handoff. The source row A exposes the inspected terms and partial sums, the row Q exposes only adjacent-term quotient windows, and M records the finite threshold and rational margin used by the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed defect data into a majorant comparison, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, E supplies the regular rational and Real-facing handoff. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 1719.3 (Raabe–Duhamel test tail handoff)**.** *Every accepted* $\mathsf{RaabeDuhamelTest}^{\uparrow}$ *read reaching a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$A, Q \longmapsto M \longmapsto D, W \longmapsto T \longmapsto E \longmapsto (H, C, P, N).$$

Thus a Phase Real or Completion consumer receives tail convergence data only after the source series row, adjacent-term quotient row, defect threshold, dyadic domination ledger, finite windows, comparison-test handoff, endpoint route, transport, replay, provenance, and local NameCert have all been displayed.

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 1719.1, the only term-level data available before the defect estimate are A and Q, so the read first consumes the displayed source windows and adjacent-term quotient windows.

The tail budget is then forced through M. This row names the finite threshold and rational margin attached to the requested tail, and it contributes no selected infinite sum or host summability theorem.

Next the comparison route passes through D, W, T. The dyadic domination ledger D and finite window row W prepare exactly the rows consumed by $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ in T, matching the sibling route named by chapter 1459.

Only after that comparison row has been displayed can the packet reach E, the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve this same route by transport, replay, provenance, and local naming, so no consumer can bypass it through quotient series equality, selected sums, countable choice, trichotomy, host summability, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{RaabeDuhamelTest}^{\uparrow}$ endpoint-facing read factors through source series windows, adjacent-term quotient windows, a defect-modulus row, dyadic domination, StreamName windows, a RegularCauchyComparisonTest handoff, RegSeqRat and RealUp endpoint routing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: regularcauchyratiotest, regularcauchycomparisontest, regseqrat, real: $\mathsf{RaabeDuhamelTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries adjacent-term quotient windows, a defect-modulus threshold, dyadic domination, comparison-test handoff, regular rational endpoint routing, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RaabeDuhamelTest}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RaabeDuhamelTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1719.1, Definition 1719.2, and Definition 1719.3 seed the finite Raabe–Duhamel convergence-test boundary over displayed quotient-defect, domination, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RaabeDuhamelTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, defect-modulus stability, domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.RaabeDuhamelTestUp.

Constructive story: A $\mathsf{RaabeDuhamelTest}^{\uparrow}$ packet is generated from source series windows, adjacent-term quotient windows, a displayed defect modulus, dyadic domination, StreamName windows, a RegularCauchyComparisonTest handoff, RegSeqRat and RealUp endpoint routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1720

A Concrete Naming Certificate for $\mathsf{CauchyRealizerSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealizerSequence}^{\uparrow}$ packet records a finite sequence-level window over explicit Cauchy realizers before any completion-facing consumer is allowed to read a terminal real-name seal. It sits over the single-realizer route of chapter 1771, the finite stream-window gateway of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite sequence surface for the L10 Real and Completion route; it is not quotient stream equality, a selected limit, countable choice, RealUp completeness, trichotomy, host real equality, Lean verification, or a standard bridge.

Definition 1720.1 (Cauchy realizer sequence carrier)**.** *A* $\mathsf{CauchyRealizerSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of sequence indices, precision requests, and visible observation windows consumed by the sequence route.*
2. *W is the finite family of* $\mathsf{CauchyRealizer}^{\uparrow}$ *rows, one row for each displayed sequence window in S.*
3. *R is the family of* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks obtained from the displayed windows of S and the realizer rows of W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested radii, slack comparisons, and finite compatibility checks for every displayed readback in R.*
5. *E is the family of terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the corresponding rows of S, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, realizer rows, readbacks, dyadic ledgers, real seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each Real or Completion request through the finite schedule S, the realizer family W, the readback family R, the ledger family D, the seal family E, and the structural rows H, P, N.*

8. *P is the* Pkg *provenance over* CauchyRealizerSequence$^{\uparrow}$, CauchyRealizer$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{CauchyRealizerSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, W, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, ambient RealUp completeness, trichotomy, host real equality, Lean verification, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CauchyRealizerSequenceUp.CauchyRealizerSequenceTasteGate single carrier alignment`

Theorem 1720.2 (Cauchy realizer sequence NameCert obligations)**.** *For every accepted* CauchyRealizerSequence$^{\uparrow}$ *carrier*

$$Z = (S, W, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyRealizerSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the finite* StreamName$^{\uparrow}$ *schedule S, the* CauchyRealizer$^{\uparrow}$ *family W, the* RegSeqRat$^{\uparrow}$ *readback family R, the* DyadicRatCore$^{\uparrow}$ *ledger family D, the terminal* Real$^{\uparrow}$ *seal family E, and the structural rows H, C, P, N;*
2. *schedule exactness, requiring each displayed sequence index and precision request to enter through S before any realizer row is read;*
3. *rowwise realizer exposure, requiring every member of W to be an exposed* CauchyRealizer$^{\uparrow}$ *row attached to the corresponding finite window of S;*
4. *regular-readback exactness, requiring R to consume only the displayed schedule and realizer rows S, W;*
5. *dyadic tolerance accounting, requiring D to store precisely the visible radii, slack rows, and comparison evidence consumed by R and E;*
6. *Real-seal routing, requiring every terminal* Real$^{\uparrow}$ *row in E to be downstream of the matching S, W, R, D rows;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *non-escape for quotient stream equality, selected limits, countable choice, RealUp completeness, trichotomy, host real equality, Lean verification, and standard-bridge data.*

No sequence-level Cauchy realizer obligation is stored outside these rows.

Proof. Project the carrier of Definition 1720.1. The finite schedule S is the only source of displayed sequence indices and precision requests, while the family W is the only source of CauchyRealizer rows attached to that schedule.

For each displayed window, the readback and ledger rows are then forced. The row R records exactly the regular rational observations obtained from S and W, and D records exactly the dyadic tolerances and comparison evidence used to certify those observations.

The terminal seal rows are downstream. The family E has no entrance other than the matching S, W, R, D rows, while H, C, P, N transport, replay, source, and name the same finite route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 1720.3 (Cauchy realizer sequence window factorization)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{CauchyRealizerSequence}^{\uparrow}$ *carrier factors each displayed sequence request through the finite route*

$$S \longmapsto W \longmapsto R \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a finite StreamName schedule, a family of CauchyRealizer rows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, terminal RealUp seals, transport, replay, provenance, and local naming. The factorization does not supply quotient stream equality, selected limits, countable choice, RealUp completeness, trichotomy, host real equality, Lean verification, or a standard bridge.

Proof. Begin with the accepted carrier and inspect its schedule row. A downstream request has no direct access to a sequence member; it first lands in the finite $\mathsf{StreamName}^{\uparrow}$ schedule S.

Now proceed by finite-window induction over the displayed schedule. In the empty window there is no member to read, so the route is exhausted by the structural rows. In the step case, the new displayed request exposes exactly one attached $\mathsf{CauchyRealizer}^{\uparrow}$ row in W, then its $\mathsf{RegSeqRat}^{\uparrow}$ readback in R, then the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger row in D, and then the matching $\mathsf{Real}^{\uparrow}$ seal in E.

The induction hypothesis transports the already displayed windows through H, C, P, N, and the step row uses the same structural transport and replay. Thus every consumed sequence request factors through the displayed route, and the excluded quotient, selector, choice, completeness, order, equality, verification, and bridge objects remain outside the packet. □

Theorem 1720.4 (Cauchy realizer sequence Real seal non-escape)**.** *Let* $Z = (S, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyRealizerSequence}^{\uparrow}$ *carrier. Any downstream use of a terminal* $\mathsf{Real}^{\uparrow}$ *seal in* E *must pass through the finite schedule-realizer-readback-ledger route*

$$S, W, R, D$$

and the structural rows H, C, P, N*. Thus no Real seal in the sequence packet can be used as a hidden endpoint for quotient stream equality, selected limits, countable choice, RealUp completeness, trichotomy, host real equality, Lean verification, or a standard bridge.*

Proof. The only $\mathsf{Real}^{\uparrow}$ rows in the carrier are the rows of E. By Definition 1720.1, each of them is placed after a displayed schedule window, an attached CauchyRealizer row, a regular rational readback, and a dyadic tolerance ledger.

A downstream consumer therefore reaches such a seal only by replaying the corresponding finite route through C and transporting it componentwise through H. The package row P and local name N record the same route and do not add another terminal endpoint.

Because no carrier coordinate stores quotient stream equality, a selected limit, choice data, RealUp completeness, trichotomy, host real equality, Lean evidence, or bridge data, the terminal Real seals cannot escape the finite route named above. □

Falsifiable prediction: Every $\mathsf{CauchyRealizerSequence}^{\uparrow}$ Real or Completion read factors through a finite StreamName schedule, CauchyRealizer rows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-realizer, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyRealizerSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite sequence schedule, rowwise

CauchyRealizer data, regular readbacks, dyadic tolerance ledgers, terminal Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealizerSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealizerSequence}^{\uparrow}$ packet is generated from a finite StreamName schedule, CauchyRealizer rows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, terminal RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealizerSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1720.1, Definition 1720.2, Definition 1720.3, and Definition 1720.4 seed the finite sequence-level realizer route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealizerSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, RealUp completeness, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, schedule exactness, rowwise realizer exposure, regular-readback exactness, dyadic tolerance accounting, Real-seal routing, finite-window factorization, and Real-seal non-escape for BEDC.Derived.CauchyRealizerSequenceUp.

Chapter 1721

A Concrete Naming Certificate for $\mathsf{RegularCauchyShuffle}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyShuffle}^{\uparrow}$ packet records the finite even/odd shuffle of two regular Cauchy rational names before any Real or Completion consumer is allowed to read the shuffled real-name boundary. It sits over the parity discipline of chapter 414, the finite schedule-change route of chapter 1568, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive finite schedule-combination certificate for the L10 Real and Completion route; it is not quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Definition 1721.1 (Regular Cauchy shuffle carrier)**.** *A* $\mathsf{RegularCauchyShuffle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (A, B, S, E, O, D, R_A, R_B, L_A, L_B, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose finite regular Cauchy observations are to be shuffled.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *shuffle schedule, recording the displayed requests and whether each request reads the even branch or the odd branch.*
3. *E is the even projection ledger, exposing exactly those schedule windows of S that read from the A branch.*
4. *O is the odd projection ledger, exposing exactly those schedule windows of S that read from the B branch.*
5. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the visible radii, slack rows, and branchwise comparison evidence used by both projections.*
6. *R_A and R_B are the regular readbacks of the even and odd projection routes, read only from A, E, D and from B, O, D respectively.*
7. *L_A and L_B are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows for the two projected regular Cauchy names.*
8. *H records componentwise* hsame *transport for the source rows, shuffle schedule, projection ledgers, tolerance ledger, readbacks, real seals, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each Real or Completion request through S, the selected projection ledger E or O, the shared ledger D, the corresponding readback R_A or R_B, the matching seal L_A or L_B, and the structural rows H, P, N.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyShuffle}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyShuffle}^{\uparrow}}$ *row.*

The classifier compares accepted shuffle packets only through the displayed rows $A, B, S, E, O, D, R_A, R_B, L_A, L_B, H$ It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyShuffleUp.RegularCauchyShuffleTasteGate single carrier alignment`

Theorem 1721.2 (Regular Cauchy shuffle NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyShuffle}^{\uparrow}$ *carrier*

$$Z = (A, B, S, E, O, D, R_A, R_B, L_A, L_B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyShuffle}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows A, B, the* $\mathsf{StreamName}^{\uparrow}$ *shuffle schedule S, the even and odd projection ledgers E, O, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the regular readbacks R_A, R_B, the* $\mathsf{Real}^{\uparrow}$ *seals L_A, L_B, and the structural rows H, C, P, N;*

2. *schedule exactness, requiring every displayed request in S to choose exactly one of the even or odd projection ledgers before any regular readback is consumed;*

3. *even projection exactness, requiring E to read only the A branch through the even windows of S;*

4. *odd projection exactness, requiring O to read only the B branch through the odd windows of S;*

5. *shared dyadic accounting, requiring D to store precisely the visible tolerances and slack comparisons used by both projection readbacks;*

6. *regular-readback routing, requiring R_A to consume only A, E, D and R_B to consume only B, O, D;*

7. *Real-seal routing, requiring L_A and L_B to be downstream of their corresponding source, projection, ledger, and readback rows;*

8. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*

9. *non-escape for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard-bridge data.*

No shuffle obligation is stored outside these rows.

Proof. Project the carrier of Definition 1721.1. The only source rows are A and B, and the only schedule row is S.

Now inspect each displayed request of S. The schedule sends it to exactly one projection ledger: even requests enter E and then read from A, while odd requests enter O and then read from B. The shared dyadic ledger D records the tolerance evidence used by both branches and does not introduce another branch.

The readback and seal rows are downstream. The row R_A is attached to A, E, D, the row R_B is attached to B, O, D, and the seals L_A, L_B are reached only after those corresponding readbacks. The structural rows H, C, P, N transport, replay, source, and name the same finite routes.

Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 1721.3 (Regular Cauchy shuffle even/odd projection handoff)**.** *Let*

$$Z = (A, B, S, E, O, D, R_A, R_B, L_A, L_B, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyShuffle}^{\uparrow}$ *carrier. Every downstream Real or Completion request for an even projection factors through*

$$S \longmapsto E \longmapsto A \longmapsto D \longmapsto R_A \longmapsto L_A \longmapsto H, C, P, N,$$

and every downstream request for an odd projection factors through

$$S \longmapsto O \longmapsto B \longmapsto D \longmapsto R_B \longmapsto L_B \longmapsto H, C, P, N.$$

The handoff is the finite parity projection surface supplied by the shuffle packet. It exports no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyShuffleUp.RegularCauchyShuffleEvenOddProjectionHandoff`

Proof. Begin with a downstream request and read the finite schedule S. If the schedule marks the request as even, the carrier exposes the projection ledger E; if it marks the request as odd, the carrier exposes the projection ledger O.

In the even case, the projection ledger has only the A branch as source. The request then passes through the shared dyadic ledger D, the regular readback R_A, and the seal L_A before the structural rows H, C, P, N record transport, replay, provenance, and local naming.

In the odd case, the same argument uses O in place of E, the B branch in place of A, the readback R_B in place of R_A, and the seal L_B in place of L_A.

These two routes are exactly the finite even and odd projections of the carrier. No third route exists in the displayed packet, and the excluded quotient, selector, choice, equality, order, completeness, verification, and bridge objects remain outside the handoff. □

Falsifiable prediction: Every $\mathsf{RegularCauchyShuffle}^{\uparrow}$ Real or Completion read factors through a finite StreamName shuffle schedule, an even or odd projection ledger, the selected RegSeqRat source row, a shared DyadicRatCore tolerance ledger, the corresponding regular readback, the corresponding RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-interleaving, cauchy-reindexing, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyShuffle}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular Cauchy source rows, one finite shuffle schedule, two projection ledgers, a shared dyadic tolerance ledger, two readbacks, two terminal real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyShuffle}^{\uparrow}}$.

Constructive story: A RegularCauchyShuffle$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, a finite StreamName$^{\uparrow}$ shuffle schedule, even and odd projection ledgers, a shared DyadicRatCore$^{\uparrow}$ tolerance ledger, two regular readbacks, two terminal Real$^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyShuffle}^{\uparrow}}$, seed)

Scope closed: Definition 1721.1, Definition 1721.2, and Definition 1721.3 seed the finite even/odd shuffle and projection route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyShuffle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, schedule exactness, even and odd projection exactness, shared dyadic accounting, regular-readback routing, Real-seal routing, and projection handoff for BEDC.Derived.RegularCauchyShuffle

Chapter 1722

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$

Following Bishop and Bridges' constructive Cauchy sequence discipline, the $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ packet records the finite BEDC route by which a cofinal subsequence of a regular Cauchy row preserves the same Real-facing seal. It consumes the subsequence interface of chapter 381, the cofinal selector surface of chapter 405, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet names invariance as displayed finite routing data, not as quotient stream equality, selected limit choice, host real equality, or ambient Real completeness.

Definition 1722.1 (Regular-Cauchy subsequence invariance carrier)**.** *A* $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (Q, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source regular Cauchy row whose Real-facing seal is being preserved.*
2. *F is the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *selector row, carrying only displayed monotone and cofinal access to source indices.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the source and selected subsequence windows inspected along F.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the selected windows and the regular Cauchy source.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the source row, selected windows, readback row, and Real-facing seal.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal exposed after Q, F, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the source row, cofinal selector, stream windows, readback row, dyadic ledger, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from any finite subsequence read back through F, S, R, D before the seal row E is reached.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, S, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceInvarianceUp.RegularCauchySubsequenceInvarianceTasteGate`
`single carrier alignment`

Theorem 1722.2 (Regular-Cauchy subsequence invariance NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ *carrier*

$$I = (Q, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, cofinal selector, stream-window ledger, regular rational readback, dyadic tolerance ledger, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *cofinal access, requiring every selected index read to pass through the displayed selector row F rather than through an external choice function;*
3. *window alignment, requiring every subsequence observation to be one of the finite* $\mathsf{StreamName}^{\uparrow}$ *windows named by S;*
4. *readback exactness, requiring R to read only the selected windows from S and the source regular Cauchy row Q;*
5. *tolerance preservation, requiring the dyadic ledger D to be the same ledger consumed by Q, S, R, and the seal row E;*
6. *Real-seal non-escape, requiring E to be reached only after F, S, R, D have displayed the subsequence route from Q.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 1722.1. The visible analytic rows are exactly the source row Q, the cofinal selector F, stream windows S, regular rational readback R, dyadic ledger D, and Real seal E.

Cofinal access is read from F. Since every selected index must enter through that row, the certificate never calls on a detached infinite choice function or host subsequence object.

Window alignment and readback exactness are read from S and R. The selected observations are the finite windows named by S, and R can only replay those windows against the source row Q.

Tolerance preservation is the projection to D. The source row, selected windows, readback row, and Real seal all consume that same dyadic ledger, so no second tolerance budget can alter the subsequence read.

Finally, H, C, P, N transport, replay, source, and name only the displayed packet rows. The excluded quotient, choice, selected-limit, host-equality, completeness, Lean, and bridge data therefore have no coordinate in the local NameCert surface. □

Theorem 1722.3 (Regular-Cauchy subsequence invariance cofinal route)**.** *Let $I = (Q, F, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ *carrier. Every subsequence read from I factors through the cofinal route*

$$Q, F, S, R, D, E.$$

In particular, the read reaches the Real-facing seal only after the source regular Cauchy row, cofinal selector, finite stream windows, regular rational readback, and dyadic tolerance ledger have all been displayed.

Proof. Begin with a subsequence read of the accepted packet. The only selector coordinate is F, so the read first factors from the source row Q through the displayed cofinal access row.

The selected observation is then forced into the finite window ledger S. Since the carrier has no host stream coordinate, each inspected value is read through one of the displayed $\mathsf{StreamName}^{\uparrow}$ windows.

The regular rational readback and tolerance rows come next. The row R replays the selected windows as regular rational data, and D supplies the shared dyadic tolerance ledger used by both the source row and the selected readback.

The Real-facing seal is downstream of those rows. Thus the read reaches E only through Q, F, S, R, D, and the support rows H, C, P, N cannot add another analytic route. □

Theorem 1722.4 (Regular-Cauchy subsequence invariance Real seal non-escape)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ *carrier*

$$I = (Q, F, S, R, D, E, H, C, P, N),$$

the public export is the displayed finite route from the source regular Cauchy row to the Real-facing seal. It exports no quotient stream equality, selected limit, countable choice principle, host real equality, ambient Real completeness, independent Real carrier, Lean credential, or bridge credential.

Proof. Use Definition 1722.3. Every Real-facing subsequence read has already passed through the source row, cofinal selector, finite stream windows, regular rational readback, and dyadic tolerance ledger before it reaches E.

The carrier has no remaining analytic coordinate. The transport, replay, provenance, and naming rows preserve the displayed route, but they do not introduce quotient representatives, selected limits, choice principles, host equality witnesses, completeness principles, or independent Real carriers.

Therefore the public export is exactly the finite seal-preserving route described above. The omitted Lean and bridge credentials are also not exported, because this seed packet records only the paper-side finite NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}$ packet is generated from a source regular Cauchy row, a cofinal selector row, finite StreamName windows, RegSeqRat readback, a shared DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1722.1, Definition 1722.2, Definition 1722.3, and Definition 1722.4 seed the finite cofinal-subsequence route preserving the same Real-facing regular Cauchy seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceInvariance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Real completeness, selected limits, quotient stream equality, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Formal closure requires checked carrier admission, cofinal-route factorization, tolerance preservation, and Real-seal non-escape theorems under BEDC.Derived.RegularCauchySubsequenceInvarianceUp.

Chapter 1723

A Concrete Naming Certificate for $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ packet records the shared finite tail-threshold data by which a Cauchy realizer sequence serves Real and Completion requests before any terminal real-name seal is read. It refines the sequence route of chapter 1720, the single-realizer route of chapter 1771, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite tail-modulus surface for the L10 Real and Completion route; it is not quotient stream equality, a selected limit, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, host real equality, Lean verification, or a standard bridge.

Definition 1723.1 (Cauchy realizer tail modulus carrier)**.** *A* $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the shared tail-threshold and modulus row, recording the finite tail index used by every displayed precision request in the packet.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule of sequence indices, precision requests, and visible observation windows governed by M.*
3. *W is the finite family of* $\mathsf{CauchyRealizer}^{\uparrow}$ *rows attached to the windows of S after the common tail threshold has been displayed.*
4. *R is the family of* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks obtained from the displayed schedule S, the realizer rows W, and the shared threshold row M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the common dyadic radii, slack comparisons, and finite compatibility checks served by M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the corresponding rows of M, S, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the shared threshold, window schedule, realizer rows, readbacks, dyadic ledger, real seals, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each Real or Completion request through M, S, W, R, D, E and the structural rows H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$, $\mathsf{CauchyRealizerSequence}^{\uparrow}$, $\mathsf{CauchyRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealizerTailModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $M, S, W, R, D, E, H, C, P, N$. It has no coordinate for private tail thresholds, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, host real equality, Lean verification, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.CauchyRealizerTailModulusUp.CauchyRealizerTailModulusTasteGate single carrier alignment`

Theorem 1723.2 (Cauchy realizer tail modulus NameCert obligations)**.** *For every accepted* CauchyRealizerTailMod *carrier*

$$T = (M, S, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealizerTailModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the shared threshold row M, the* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{CauchyRealizer}^{\uparrow}$ *family W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the terminal* $\mathsf{Real}^{\uparrow}$ *seal family E, and the structural rows H, C, P, N;*
2. *shared-tail admission, requiring every displayed precision request in S to use the same threshold row M before any realizer row is read;*
3. *rowwise realizer exposure, requiring every member of W to be attached to a window of S whose tail threshold is supplied by M;*
4. *regular-readback exactness, requiring R to consume only the displayed rows M, S, W;*
5. *dyadic tolerance accounting, requiring D to store precisely the common radii, slack rows, and comparison evidence served by M and consumed by R and E;*
6. *Real-seal routing, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *row in E to be downstream of the matching M, S, W, R, D rows;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *non-escape for private thresholds, quotient stream equality, selected limits, countable choice,* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, host real equality, Lean verification, and standard-bridge data.*

No tail-modulus obligation for this carrier is stored outside these rows.

Proof. Project the carrier of Definition 1723.1. The shared row M is the only displayed source of tail-threshold data, and S is the only displayed source of sequence indices, precision requests, and observation windows governed by that row.

For each displayed window, finite-window induction determines the remaining route. The empty window has no realizer or readback to expose. In the step case, the new request first reads the already displayed threshold row M, then its scheduled window in S, its $\mathsf{CauchyRealizer}^{\uparrow}$ row in W, its $\mathsf{RegSeqRat}^{\uparrow}$ readback in R, and its dyadic accounting row in D.

The terminal seal rows are downstream of the same route. The family E has no entrance other than the matching M, S, W, R, D rows, while H, C, P, N transport, replay, source, and name that finite route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 1723.3 (Cauchy realizer shared-tail exactness)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ *carrier factors each displayed precision request through the common route*

$$M \longmapsto S \longmapsto W \longmapsto R \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives one shared tail-threshold row, a finite StreamName schedule, CauchyRealizer rows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, terminal RealUp seals, transport, replay, provenance, and local naming. The factorization does not supply private tail thresholds, quotient stream equality, selected limits, countable choice, $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, host real equality, Lean verification, or a standard bridge.*

Proof. Begin with an accepted carrier and inspect the common row M. A downstream request has no direct access to a private tail threshold, because the carrier exposes only the single shared threshold and modulus row.

Proceed by finite-window induction over S. In the empty window the consumer can only see the structural rows H, C, P, N. In the step case, the displayed request uses M before its scheduled window, realizer row, readback row, dyadic ledger row, and terminal seal are available.

The induction hypothesis transports the earlier windows through the same structural rows, and the step row uses the same replay route. Therefore every displayed request factors through the common route above, and no omitted object can be read as part of the accepted packet. □

Theorem 1723.4 (Cauchy realizer tail modulus Real seal non-escape)**.** *Let* $T = (M, S, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ *carrier. Any downstream use of a terminal* $\mathsf{Real}^{\uparrow}$ *seal in* E *must pass through the shared tail-modulus route*

$$M, S, W, R, D$$

and the structural rows H, C, P, N*. Thus no Real seal in the packet can be used as a hidden endpoint for private thresholds, quotient stream equality, selected limits, countable choice,* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, host real equality, Lean verification, or a standard bridge.*

Proof. The only $\mathsf{Real}^{\uparrow}$ rows in the carrier are the rows of E. By Definition 1723.1, each of them is placed after the displayed shared threshold row, a scheduled window, an attached CauchyRealizer row, a regular rational readback, and a dyadic tolerance ledger.

A downstream consumer therefore reaches such a seal only by replaying the corresponding finite route through C and transporting it componentwise through H. The package row P and local name N record the same route and do not add another terminal endpoint.

Because no carrier coordinate stores a private threshold, quotient stream equality, a selected limit, choice data, $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, host real equality, Lean evidence, or bridge data, the terminal Real seals cannot escape the finite route named above. □

Falsifiable prediction: Every $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ Real or Completion read factors through one shared tail-threshold row, a finite StreamName schedule, CauchyRealizer rows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-realizer-sequence, cauchy-realizer, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a shared tail threshold, a finite sequence schedule, rowwise CauchyRealizer data, regular

readbacks, a dyadic tolerance ledger, terminal Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealizerTailModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealizerTailModulus}^{\uparrow}$ packet is generated from one shared tail-threshold row, a finite StreamName schedule, CauchyRealizer rows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, terminal RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealizerTailModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1723.1, Definition 1723.2, Definition 1723.3, and Definition 1723.4 seed the shared finite tail-modulus route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealizerTailModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, shared-tail admission, rowwise realizer exposure, regular-readback exactness, dyadic tolerance accounting, Real-seal routing, finite-window factorization, and Real-seal non-escape for BEDC.Derived.CauchyRealizerTailModulusUp.

Chapter 1724

A Concrete Naming Certificate for $\mathsf{DecimalNormality}^{\uparrow}$

Following Borel on normal numbers, the $\mathsf{DecimalNormality}^{\uparrow}$ packet records only finite decimal digit-frequency evidence as BEDC NameCert data. It consumes the finite decimal-prefix readback surface of chapter 1417, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the finite stream-window interface of chapter 126, the regular rational handoff of chapter 301, and the probability and frequency surfaces of chapter 184, chapter 185, and chapter 92 where those rows are explicitly displayed. The packet distinguishes decimal digit-frequency certificates from plain decimal expansions while refusing to turn Borel's measure-theoretic normality theorem into a local finite certificate.

Definition 1724.1 (Decimal normality carrier)**.** *A* $\mathsf{DecimalNormality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *row, carrying the inspected base-ten digit prefix and its endpoint-normalized decimal readback route.*
2. *W is the finite prefix-window row, listing the displayed prefix lengths, subword windows, and digit blocks whose frequencies are being inspected.*
3. *F is the digit-frequency ledger, recording for each displayed digit or finite digit block the counted occurrences in W, the displayed denominator, and the rational comparison row for the target frequency.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same decimal window and frequency ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the D, W, F, R rows have been displayed.*
6. *H records componentwise* hsame *transport for the decimal expansion, prefix-window, frequency, regular-readback, real-seal, continuation, provenance, and local naming coordinates.*
7. *C records displayed* Cont *routes that replay finite prefix-window frequency rows without opening an infinite host stream or a measure quotient.*
8. *P is the* Pkg *provenance over* $\mathsf{DecimalNormality}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the displayed probability-frequency surfaces,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalNormality}^{\uparrow}}$ *row.*

The carrier has no coordinate for a measure-theoretic almost-everywhere normality theorem, random real selector, quotient stream equality, selected host decimal, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.DecimalNormalityUp.DecimalNormalityTasteGate single carrier alignment`

Theorem 1724.2 (Decimal normality NameCert obligations)**.** *For every accepted* $\mathsf{DecimalNormality}^{\uparrow}$ *carrier* $K = (D, W, F, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DecimalNormality}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the decimal expansion row, finite prefix-window row, digit-frequency ledger, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*
2. *digit-window frequency stability, requiring each transported frequency claim to read the same displayed prefix windows and the same counted digit or finite-block rows;*
3. *decimal endpoint normalization compatibility, requiring endpoint conventions already displayed by* $\mathsf{DecimalExpansion}^{\uparrow}$ *to be read before any frequency row is compared;*
4. $\mathsf{Real}^{\uparrow}$ *readback, requiring the terminal real seal to remain downstream of the decimal expansion, prefix-window, frequency, and regular-readback rows;*
5. *ledger non-escape, requiring every consumer of the packet to read only* $D, W, F, R, E, H, C, P, N$*.*

No obligation is stored in a hidden infinite decimal stream, an almost-everywhere normality theorem, a random real selector, quotient stream equality, or ambient real completeness.

Proof. Carrier admission is projection from Definition 1724.1. The tuple displays the decimal expansion, finite prefix windows, frequency counts, regular readback, real seal, transport, replay, provenance, and local naming rows as one finite packet.

Frequency stability is componentwise. A transported row in H preserves the prefix-window coordinate W and the frequency coordinate F, so a digit or block count can be compared only against the same displayed finite window data.

Endpoint normalization compatibility follows because the decimal expansion row D is read before F. Any terminating or repeating endpoint convention must therefore be supplied by the decimal-expansion packet before a frequency ledger can count the displayed digits.

For real readback, the row E appears only downstream of D, W, F, R. The continuation and provenance rows replay that same order and add no external coordinates. Hence ledger non-escape follows from the carrier list itself. □

Theorem 1724.3 (Decimal normality frequency-window stability)**.** *Let* $K = (D, W, F, R, E, H, C, P, N)$ *and* $K' = (D', W', F', R', E', H', C', P', N')$ *be accepted* $\mathsf{DecimalNormality}^{\uparrow}$ *carriers. If the displayed* hsame *rows in* H, H' *identify the decimal-expansion rows, finite prefix-window rows, and digit-frequency ledger rows, then every finite prefix-window frequency row read from* K *transports to the corresponding finite prefix-window frequency row read from* K' *through the displayed* Cont *routes. The transported read uses no host infinite stream, quotient stream equality, random selector, measure-completion theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. Read the frequency row through the source carrier. By Definition 1724.1, the row is not an isolated predicate: it is attached to the decimal expansion row D and the finite prefix-window row W.

The displayed hsame data identify the corresponding decimal, window, and frequency coordinates in K' componentwise. Transport therefore carries the counted digit or block row, its displayed denominator, and its rational frequency comparison to the matching F' coordinate.

The continuation row then replays the same finite read on the target side. Because the replay route uses only D', W', F' and the structural rows R', E', H', C', P', N', none of the excluded external objects can enter the transported frequency-window claim. □

Theorem 1724.4 (Decimal normality RealUp readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DecimalNormality}^{\uparrow}$ *packet first reads the finite* $\mathsf{DecimalExpansion}^{\uparrow}$ *windows, then the displayed prefix-window frequency ledger, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then the terminal real seal. The consumer receives no measure-theoretic almost-everywhere normality theorem, random real selector, quotient stream equality, selected host decimal, or host completeness principle from this readback.*

Proof. Start with an accepted carrier $K = (D, W, F, R, E, H, C, P, N)$. The only real-facing coordinate is E, and the carrier places it after the decimal, window, frequency, and regular-readback rows.

A consumer must therefore read D before it can inspect any digit data. The finite prefix-window row W bounds the displayed digit positions, and the ledger F counts only those displayed positions or finite blocks.

The regular-readback row R is reached from the same finite route, and the structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming. Since no excluded object is a coordinate of the packet, the terminal real readback cannot export it. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalNormality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalNormality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1724.1, Definition 1724.2, Definition 1724.3, and Definition 1724.4 seed the finite decimal digit-frequency certificate surface over $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalNormality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic Borel normality as a theorem, random real selection, quotient streams, selected host decimals, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, digit-window frequency stability, decimal endpoint normalization compatibility, RealUp readback, and ledger non-escape for BEDC.Derived.DecimalNormalityUp.

Constructive story: A $\mathsf{DecimalNormality}^{\uparrow}$ packet is generated from a $\mathsf{DecimalExpansion}^{\uparrow}$ row, finite prefix-window rows, digit-frequency ledger rows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1725

A Concrete Naming Certificate for $\mathsf{CompletionExtractor}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompletionExtractor}^{\uparrow}$ packet records the finite handoff by which explicit Cauchy modulus data is extracted into the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the explicit source route of chapter 1705, the threshold surface of chapter 1423, the completion handoff of chapter 1691, the finite window rows of chapter 126, the regular readback surface of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a reviewable L10 completion-extraction object; it is not quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1725.1 (Completion extractor carrier)**.** *A* $\mathsf{CompletionExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (G, M, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row. It names the explicit Cauchy sequence data from which every completion-facing read begins.*
2. *M is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row. It records the finite precision request, the displayed threshold, and the Real-facing modulus budget read by the extractor.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It records the inspected windows admitted by the threshold row M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the dyadic radii, slack rows, comparison budgets, and tolerance conversions used by M and S.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational handoff obtained from the displayed source, threshold, finite windows, and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *seal handoff reached only after G, M, S, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source, threshold, window, dyadic ledger, regular readback, seal handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion or Real precision request through $G, M, S, D, Q, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CompletionExtractor}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{FastCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionExtractor}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite completion-extractor packet. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompletionExtractorUp.CompletionExtractorUpTasteGate single carrier alignment`

Theorem 1725.2 (Completion extractor NameCert obligations)**.** *For every accepted* $\mathsf{CompletionExtractor}^{\uparrow}$ *carrier* $X = (G, M, S, D, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionExtractor}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier exposure, requiring the displayed source, threshold, finite-window, dyadic-ledger, regular-readback, seal-handoff, transport, replay, provenance, and local naming rows to be inspectable in one finite packet;*
2. *classifier stability, requiring accepted packets to be compared only through* $G, M, S, D, Q, E, H, C, P, N$*;*
3. *modulus-window induction, requiring each finite precision request to read* M *before the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window row* S *is consumed;*
4. *regular-readback admission, requiring* Q *to be downstream of the explicit source, modulus threshold, finite windows, and dyadic tolerance ledger;*
5. *Real-seal nonescape, requiring* E *to remain downstream of* G, M, S, D, Q *and never to serve as quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient completeness.*

No completion-extractor obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CompletionExtractorUp.CompletionExtractorNameCertObligations`

Proof. Carrier exposure is the projection of Definition 1725.1. The displayed packet consists of the explicit source G, threshold row M, finite windows S, dyadic ledger D, regular readback Q, seal handoff E, and structural rows H, C, P, N.

Classifier stability is forced by the same projection. Since the carrier has no coordinate beyond those rows, any accepted comparison can inspect only the displayed source, threshold, window, dyadic, readback, seal, transport, replay, provenance, and naming data.

Modulus-window induction reads the Real-facing threshold through chapter 1423 and then opens the finite windows through chapter 126. The tolerance arithmetic used by that induction is confined to the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger of chapter 291.

The regular-readback and Real-seal rows follow chapter 301 and chapter 1691. The row Q is admitted only after G, M, S, D, and E is terminal after Q; therefore the listed five obligations exhaust the local NameCert surface. □

Theorem 1725.3 (Completion extractor regular readback route)**.** *Every accepted regular readback of a* $\mathsf{CompletionExtractor}^{\uparrow}$ *carrier* $X = (G, M, S, D, Q, E, H, C, P, N)$ *factors through the finite route*

$$G \longmapsto M \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

The route cannot use quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionExtractorUp.CompletionExtractorRegularReadbackRoute`

Proof. Start from an accepted regular readback. By Definition 1725.1, the only regular rational coordinate is Q, and it is placed after the explicit source row G, modulus threshold row M, finite-window row S, and dyadic ledger row D.

The source row is supplied by chapter 1705. The threshold row then reads the Real-facing modulus surface of chapter 1423, and the corresponding finite windows are the $\mathsf{StreamName}^{\uparrow}$ windows named by S.

The dyadic ledger D supplies the tolerance conversions used by the same window. Only after that ledger is fixed does chapter 301 admit the regular readback row Q.

Finally, the seal handoff E is downstream of the readback through chapter 1691. The excluded quotient, selector, choice, order-decision, host-equality, and ambient-completeness data have no carrier row and no replay edge, so every accepted regular readback factors through the displayed route. □

Falsifiable prediction: Every $\mathsf{CompletionExtractor}^{\uparrow}$ regular or Real-facing read factors through an EffectiveCauchySequence source, a RealCauchyModulus threshold, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp/FastCauchyCompletion seal handoff, transport, replay, provenance, and local naming.

Independence witness: effective-cauchy-sequence, realcauchymodulus, fast-cauchy-completion, streamname, regseqrat, dyadicratcore: $\mathsf{CompletionExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source, threshold, finite-window, tolerance-ledger, regular-readback, seal-handoff, transport, replay, provenance, and local NameCert rows in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionExtractor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1725.1, Definition 1725.2, and Definition 1725.3 seed the finite completion-extraction route from explicit Cauchy modulus data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionExtractor}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CompletionExtractorUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked carrier obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier stability, modulus-window induction, regular-readback admission, and Real-seal nonescape for BEDC.Derived.CompletionExtractorUp.

Constructive story: A $\mathsf{CompletionExtractor}^{\uparrow}$ packet is generated from an EffectiveCauchySequence source row, RealCauchyModulus threshold row, StreamName finite windows, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp/FastCauchyCompletion seal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1726

A Concrete Naming Certificate for $\mathsf{LocallyCompact}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop–Bridges local compactness practice, the $\mathsf{LocallyCompact}^{\uparrow}$ packet records a compact neighbourhood certificate at one inspected point before a properness or completion consumer may read any global compactness boundary. It sits between the closed-ball row of chapter 1451, the compact metric witness of chapter 129, the located compact handoff of chapter 1488, and the radius-window comparison of **??**. The certificate is local: it names one displayed neighbourhood window and its finite compact witness, not a host-selected neighbourhood base, an open-cover compactness theorem, or a global proper metric structure.

Definition 1726.1 (LocallyCompact carrier)**.** *A* $\mathsf{LocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, x, r, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *row whose distance, point, and transport data are already admitted as finite BEDC carrier rows;*
2. *x is the inspected point row in X, named as local data rather than selected from an ambient host set;*
3. *r is a positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius ledger for the neighbourhood window around x;*
4. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row around x at radius r, exposing the closed comparison $d(x, y) \leq_{\mathsf{R}} r$ through the center-radius certificate surface of chapter 1451;*
5. *K is a* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed ball B or for the named neighbourhood window carried by B;*
6. *A is an optional* $\mathsf{LocatedCompact}^{\uparrow}$ *handoff, either a located compact row for the same window or an explicit empty handoff row when no located boundary is claimed;*
7. *H records componentwise* hsame *transports for the metric source, inspected point, positive radius ledger, closed-ball row, compact witness, located handoff, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from a local neighbourhood request through X, x, r, B, K, A, H and into the compactness or completion consumer;*

9. *P is the* Pkg *provenance over* $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for an open cover, a host-selected neighbourhood selector, quotient point equality, countable choice, ambient real completeness, or a global properness assertion.

Lean4 (definition) `BEDC.Derived.LocallyCompactUp.TasteGate.LocallyCompactUp`

Theorem 1726.2 (LocallyCompact NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *carrier* $L = (X, x, r, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$ *surface consists exactly of carrier admission, local neighbourhood indexing, closed-ball compact witness transport, located compact handoff when the row A is present,* $\mathsf{ProperMetric}^{\uparrow}$ *comparison boundary, and non-escape for the displayed rows $X, x, r, B, K, A, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactUpTasteGate single carrier alignment`

Proof. Project the accepted packet $L = (X, x, r, B, K, A, H, C, P, N)$. Carrier admission reads only the metric source, inspected point, radius ledger, closed-ball row, compact witness, located handoff row, transports, replay, provenance, and local naming row.

Local neighbourhood indexing is the positive-radius part of the packet. The request first names x and r, then reaches the closed-ball row B; no selector ranging over host neighbourhoods occurs.

Closed-ball compact witness transport is componentwise. The compact witness K is attached to the displayed ball or window named by B, and every transported read passes through H before the continuation replay C reaches a consumer.

The located compact handoff is optional and exact. When A is a $\mathsf{LocatedCompact}^{\uparrow}$ row, it is read through the same B, K, H, C, P, N support. When the empty handoff row is displayed, the packet makes no locatedness claim.

The $\mathsf{ProperMetric}^{\uparrow}$ comparison boundary is one-way. A proper metric packet may use the local row as one indexed closed-ball window, but this local row does not supply a radius-indexed family for all centers and radii.

Finally, non-escape follows from the carrier definition. Since the packet has no row for open covers, host neighbourhood selectors, quotient point equality, countable choice, or ambient real completeness, those objects cannot be recovered by any local $\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$ consumer. □

Theorem 1726.3 (LocallyCompact closed-ball neighbourhood base)**.** *Every local compactness consumer of an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet factors through*

$$X, x, r, B, K, A, H, C, P, N$$

and receives only the displayed metric source, inspected point, positive radius ledger, closed-ball row, compact witness, optional located compact handoff, transports, replay, provenance, and local naming data. It receives no open-cover compactness, host neighbourhood selector, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactClosedBallNeighborhoodBase`

Proof. Start from a consumer request for local compactness at the inspected point. The only admissible entry point is the carrier projection, so the request first reads X, x, r and then the closed-ball row B.

The compactness data are exactly K. A consumer may transport that witness along H and replay it along C, but the route never leaves the displayed closed-ball or neighbourhood window named by B.

If located compactness is needed, the consumer reads A only as a handoff row over the same window. If no such handoff is displayed, the packet exposes no located compact claim.

The provenance row P names the sibling certificates that may be cited, and N names the local packet. Therefore every downstream compactness, completion, or properness comparison factors through the displayed tuple above, with no additional host compactness or completeness datum. □

Theorem 1726.4 (LocallyCompact located handoff boundary)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every* $\mathsf{LocatedCompact}^{\uparrow}$ *consumer of L first reads the* $\mathsf{Met}^{\uparrow}$ *row X, inspected point x,* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius ledger r,* $\mathsf{MetricClosedBall}^{\uparrow}$ *row B, and* $\mathsf{CompactMetric}^{\uparrow}$ *witness K before the located handoff row A can be inspected. The handoff is transported through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows H, C, P, N, and it does not claim open-cover compactness, a neighbourhood-base selector, countable choice, global properness, or ambient Real completeness.*

Proof. Use the carrier of Definition 1726.1. The located handoff row A is listed only after the metric source, inspected point, positive radius ledger, closed-ball row, and compact witness.

Apply the NameCert obligations of Definition 1726.2. They require located compact handoff to be exact when A is present and to remain empty when no located boundary is displayed.

Finally use the closed-ball neighbourhood-base theorem, Definition 1726.3. Every consumer factors through $X, x, r, B, K, A, H, C, P, N$, so the located handoff cannot bypass the displayed MetricSpace, MetricClosedBall, CompactMetric, Rat or Real radius, BHist, hsame, Cont, Pkg, and NameCert route. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactLocatedHandoffBoundary`

Theorem 1726.5 (LocallyCompact properness comparison non-escape)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Any* $\mathsf{ProperMetric}^{\uparrow}$ *comparison that consumes L receives one displayed closed-ball window over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricClosedBall}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{LocatedCompact}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It receives no global properness theorem, no radius-indexed family for all centers, no neighbourhood-base selector, no countable choice, and no ambient Real completeness.*

Proof. Project L through Definition 1726.1. The properness-facing data are the single inspected point x, radius ledger r, closed-ball row B, compact witness K, and optional located handoff A.

The NameCert obligations of Definition 1726.2 make the $\mathsf{ProperMetric}^{\uparrow}$ comparison boundary one-way. A proper metric packet may read this local row as an indexed closed-ball window, but L does not contain the global family required to prove properness.

The closed-ball neighbourhood-base theorem of Definition 1726.3 exhausts the consumer route through H, C, P, N. Since no coordinate ranges over all centers and radii, the properness comparison cannot escape the displayed local packet. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactPropernessComparisonNonescape`

Theorem 1726.6 (LocallyCompact completion consumer route)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every completion-facing consumer reads local compactness only through the displayed* $\mathsf{Met}^{\uparrow}$ *source, inspected point,* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *radius ledger,* $\mathsf{MetricClosedBall}^{\uparrow}$

window, CompactMetric$^{\uparrow}$ *witness, optional* LocatedCompact$^{\uparrow}$ *handoff,* BHist *packet, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route supplies a finite local window and does not import open-cover compactness, a host neighbourhood-base selector, countable choice, global properness, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompletionConsumerRoute`

Proof. Begin with Definition 1726.3. It identifies the only local compactness route available to downstream compactness, completion, or properness consumers as $X, x, r, B, K, A, H, C, P, N$.

Use the located handoff boundary of Definition 1726.4 for consumers that require located compact data. They can read A only after the same closed-ball and compact-metric rows have been displayed.

Use the properness comparison non-escape of Definition 1726.5 for completion consumers that compare against proper metric structure. The comparison supplies one local window, not a global properness proof or ambient completeness theorem. □

Theorem 1726.7 (LocallyCompact located-ball Real-seal boundary)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* LocallyCompact$^{\uparrow}$ *packet. Every* Real$^{\uparrow}$*-facing local compactness read through L must first display the metric source X, inspected point x, radius ledger r, closed-ball row B, compact witness K, and located handoff row A, and then pass through H, C, P, N. The* Rat$^{\uparrow}$ *or* Real$^{\uparrow}$ *radius ledger is a visible row and cannot be replaced by ambient* Real$^{\uparrow}$ *completeness, a selected neighbourhood base, or global properness.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactLocatedBallRealSealBoundary`

Proof. Project the carrier of Definition 1726.1. The radius ledger r, closed-ball row B, compact witness K, and located row A are explicit fields between the metric source data X, x and the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 1726.2. They bind local neighbourhood indexing to the displayed radius row, bind closed-ball compact witness transport to B, K, H, C, and keep located handoff exact when A is present.

Use the closed-ball neighbourhood-base theorem, Definition 1726.3, and the located handoff boundary, Definition 1726.4. Together they force every located-ball read through X, x, r, B, K, A before a downstream consumer may inspect the compactness boundary.

Finally use the completion consumer route of Definition 1726.6. Completion-facing and Real$^{\uparrow}$-facing reads pass through the same H, C, P, N route, so the visible radius ledger cannot be replaced by completeness, a host neighbourhood selector, or global properness. □

Theorem 1726.8 (LocallyCompact local compact-neighbourhood obligations)**.** *Let $L = (X, x, r, B, K, A, H, C, P,$ be an accepted* LocallyCompact$^{\uparrow}$ *packet. The local compact-neighbourhood obligation for L is exactly the finite closed-ball compact window named by $X, x, r, B, K, A, H, C, P, N$: the metric source and inspected point are X, x, the positive window is r, the closed boundary is B, the compact witness is K, the located handoff is A when present, and all consumer reads pass through H, C, P, N. No obligation in this packet asserts a host neighbourhood base, open-cover compactness, quotient point equality, countable choice, global properness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1726.1. The only rows available to a local compact-neighbourhood request are $X, x, r, B, K, A, H, C, P, N$.

Apply the NameCert obligations of Definition 1726.2. They identify local neighbourhood indexing with the positive-radius route through x, r, B and closed-ball compact witness transport through K, H, C.

Use the closed-ball neighbourhood-base theorem, Definition 1726.3. It forces every consumer through the displayed tuple, so the local obligation is a finite closed-ball compact window rather than a host neighbourhood base.

The non-escape clauses in the carrier and the same closed-ball theorem exclude the missing rows. Since no open-cover, quotient, choice, global properness, or ambient completeness coordinate is present, no such obligation can be recovered from L. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactLocalCompactNeighborhoodObligations`

Theorem 1726.9 (LocallyCompact compact-window root admission)**.** *For an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet* $L = (X, x, r, B, K, A, H, C, P, N)$*, any root consumer that admits local compactness admits only the compact window consisting of the closed-ball row* B *and compact witness* K *over the displayed source rows* X, x, r*. The located row* A *may refine the same window, and* H, C, P, N *may transport, replay, source, and name it, but the root admission does not become an open-cover theorem, a selected neighbourhood system, a global properness theorem, or an ambient completeness assertion.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompactWindowRootAdmission`

Proof. Start with the carrier projection. The root consumer can see the metric source X, inspected point x, radius ledger r, closed-ball row B, compact witness K, optional located handoff A, and structural rows H, C, P, N.

Apply Definition 1726.3. It states that downstream compactness, completion, and properness comparisons factor through this displayed tuple and receive no host compactness or completeness datum.

If the consumer asks for located compact information, apply Definition 1726.4. The row A is read only after B and K, so it refines the same compact window rather than opening a second root route.

Finally apply Definition 1726.5. Properness can read the window as one boundary datum, but the packet lacks the family over all centers and radii required for global properness. □

Theorem 1726.10 (LocallyCompact properness-locality root boundary)**.** *Let* $L = (X, x, r, B, K, A, H, C, P, N)$ *be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. A* $\mathsf{ProperMetric}^{\uparrow}$ *or completion-facing root may consume* L *only as a local closed-ball compactness boundary: it receives one source* X*, one inspected point* x*, one positive radius* r*, one closed-ball row* B*, one compact witness* K*, the optional located handoff* A*, and the transport, replay, provenance, and naming rows* H, C, P, N*. It receives no all-centers/all-radii properness family and no host neighbourhood-base structure.*

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactPropernessLocalityRootBoundary`

Proof. Project the accepted packet through Definition 1726.1. The properness-facing boundary is the finite row tuple $X, x, r, B, K, A, H, C, P, N$.

Apply Definition 1726.5. It states that a $\mathsf{ProperMetric}^{\uparrow}$ comparison receives one displayed closed-ball window and not the radius-indexed family for all centers.

Use Definition 1726.6 for completion-facing roots. They read local compactness through the same metric source, radius ledger, closed-ball window, compact witness, located handoff, transport, replay, provenance, and local NameCert route.

The carrier projection and both non-escape results leave no row for a host neighbourhood base or global properness theorem. Therefore the root boundary is locality itself: a finite closed-ball compact window with exact sibling handoffs. □

Theorem 1726.11 (LocallyCompact compact-ball window)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every root local compactness read exposes the metric source X, inspected point x, positive radius r, closed-ball row B, and compact witness K before any completion-facing consumer can read the packet.*

Proof. Project the carrier of Definition 1726.1. The compact window is not an inferred object: it is the displayed row sequence X, x, r, B, K inside the finite packet.

Apply Definition 1726.2 and Definition 1726.3. These obligations require the metric source, point, radius ledger, closed-ball row, and compact witness to be visible before the neighbourhood-base surface is read.

Finally use Definition 1726.6. A completion consumer receives the same window only through H, C, P, N, so it cannot bypass X, x, r, B, K or replace them by a host compactness theorem. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompactBallWindow`

Theorem 1726.12 (LocallyCompact located compact route)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. The located compact row A is a refinement of the same closed-ball compact window B, K over X, x, r, not a second compactness source and not an independent located compact object.*

Proof. Read the carrier rows in Definition 1726.1. The located handoff coordinate A occurs after the metric source, inspected point, radius ledger, closed-ball row, and compact witness.

Apply Definition 1726.4. It binds A to the same displayed closed-ball compact witness rather than to an external compactness source.

The NameCert obligations of Definition 1726.2 then transport and replay A through the same H, C, P, N boundary. Thus the handoff refines B, K and cannot introduce a second compactness route. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactLocatedCompactRoute`

Theorem 1726.13 (LocallyCompact proper-metric nonescape)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every* $\mathsf{ProperMetric}^{\uparrow}$ *consumer receives one local closed-ball compactness boundary through $X, x, r, B, K, A, H, C, P, N$, not an all-centers/all-radii properness theorem.*

Proof. Begin with the compact-ball window of Definition 1726.11. It exposes exactly one inspected source, point, radius, closed-ball row, and compact witness before downstream consumption.

Use the located compact route of Definition 1726.12 to attach A only as a refinement of that same window. The located handoff therefore cannot widen the packet into a global properness family.

Apply Definition 1726.5 and Definition 1726.10. Together they state that properness-facing reads remain inside the displayed local boundary, with no coordinate for all centers, all radii, a host neighbourhood base, or an unrestricted proper metric theorem. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactProperMetricNonescape`

Theorem 1726.14 (LocallyCompact compact-window transport)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Any* hsame *transport of the displayed metric source X, inspected point x, positive radius r, closed-ball row B, compact witness K, and optional located handoff A preserves the same local compact window before any* $\mathsf{ProperMetric}^{\uparrow}$ *or completion consumer reads it. The transported read still factors through $X, x, r, B, K, A, H, C, P, N$ and does not upgrade the packet to global properness, open-cover compactness, a host neighbourhood-base selector, quotient point equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier in Definition 1726.1. The transport row H is componentwise over the metric source, point, radius, closed-ball row, compact witness, located handoff, replay, provenance, and local naming data.

Use the closed-ball neighbourhood-base theorem, Definition 1726.3. It identifies the visible local compactness window as exactly $X, x, r, B, K, A, H, C, P, N$, so an hsame read can only transport that displayed finite window.

Now apply the properness-locality root boundary of Definition 1726.10. A properness or completion consumer may consume the transported window as one local boundary, but it receives no all-centers/all-radii family and no host neighbourhood-base structure.

Therefore compact-window transport preserves locality itself. The optional located handoff remains attached to the same closed-ball compact witness, and the excluded global compactness, properness, quotient, choice, and completeness rows remain absent. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompactWindowTransport`

Theorem 1726.15 (LocallyCompact local properness-radius exhaustion)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every* $\mathsf{ProperMetric}^{\uparrow}$ *or completion-facing consumer that asks for a properness-radius read from L must first display exactly one inspected point x, one positive radius ledger r, one* $\mathsf{MetricClosedBall}^{\uparrow}$ *row B, one* $\mathsf{CompactMetric}^{\uparrow}$ *witness K, the optional* $\mathsf{LocatedCompact}^{\uparrow}$ *handoff A, and the existing transport, replay, provenance, and naming rows H, C, P, N over the source row X. Before those rows are displayed, no properness-radius read is available. After they are displayed, the read remains local and supplies no all-centers/all-radii properness family, open-cover compactness, countable choice, quotient point equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the proper-metric nonescape theorem, Definition 1726.13. It states that a $\mathsf{ProperMetric}^{\uparrow}$ consumer receives one local closed-ball compactness boundary through $X, x, r, B, K, A, H, C, P, N$, not an all-centers/all-radii theorem.

Apply the properness-locality root boundary of Definition 1726.10. This fixes the root read as one source, one inspected point, one positive radius, one closed-ball row, one compact witness, the optional located handoff, and the structural rows H, C, P, N.

Now use compact-window transport, Definition 1726.14. Transport may preserve the displayed window, but it cannot introduce a second point, a second radius, a global radius-indexed family, or a host neighbourhood-base route.

Therefore properness-radius access is exhausted by the displayed local packet. Completion-facing consumers may reuse that boundary through the same rows, while the absent open-cover, choice, quotient, global properness, and ambient-completeness coordinates remain unavailable. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactLocalPropernessRadiusExhaustion`

Theorem 1726.16 (LocallyCompact compact-neighbourhood basis)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* LocallyCompact$^{\uparrow}$ *packet. The neighbourhood basis exported by L is exactly the finite center-radius route displayed by the metric source X, inspected point x, positive radius ledger r,* MetricClosedBall$^{\uparrow}$ *row B,* CompactMetric$^{\uparrow}$ *witness K, optional* LocatedCompact$^{\uparrow}$ *handoff A,* ProperMetric$^{\uparrow}$ *comparison boundary, componentwise transport H, replay C, provenance P, and local* NameCert *row N. Every accepted basis read factors through those rows, and no read introduces a host-selected neighbourhood family, open-cover compactness, quotient point equality, countable choice, all-centers/all-radii properness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 1726.1. Its only local neighbourhood coordinates are the displayed center x, radius ledger r, closed-ball row B, compact witness K, optional located handoff A, and structural rows H, C, P, N over the metric source X.

Apply the closed-ball neighbourhood-base theorem, Definition 1726.3. It already forces every compactness, completion, or properness consumer to factor through $X, x, r, B, K, A, H, C, P, N$, so the basis is the finite center-radius row rather than an external selector.

Use the local properness-radius exhaustion theorem, Definition 1726.15. Properness-facing reads can compare against ProperMetric$^{\uparrow}$ only after the same closed-ball compact window has been displayed, and they still receive no all-centers/all-radii family.

Finally combine the located handoff boundary and compact-window transport, Definition 1726.4 and Definition 1726.14. They keep A and H attached to the same MetricClosedBall$^{\uparrow}$ and CompactMetric$^{\uparrow}$ rows, so replay and provenance cannot create open-cover compactness, host neighbourhoods, quotient equality, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompactNeighbourhoodBasis`

Theorem 1726.17 (LocallyCompact public neighbourhood-base export)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* LocallyCompact$^{\uparrow}$ *packet. The public neighbourhood-base read exported by L is exactly the ten-row local carrier route: compact-ball access through B, K, local-cover access through the displayed center-radius window X, x, r, componentwise* hsame *transport through H,* Cont *replay through C, provenance through P, and local* NameCert *naming through N, with the optional located handoff A attached to the same compact window. Every public read factors through those rows and exports no host-selected neighbourhood base, open-cover compactness, quotient point equality, countable choice, all-centers/all-radii properness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 1726.1. The public read has only the displayed source, point, radius, compact-ball, compact witness, located handoff, transport, replay, provenance, and local naming rows.

Apply the compact-neighbourhood basis theorem, Definition 1726.16. It identifies the neighbourhood basis as the finite center-radius route through $X, x, r, B, K, A, H, C, P, N$ and refuses external selectors.

Use compact-window transport and the local properness-radius exhaustion route, Definition 1726.14 and Definition 1726.15. These keep hsame transport, properness comparison, and completion-facing reads inside the same compact-ball window rather than turning the packet into a global properness family.

Finally use the completion consumer route, Definition 1726.6. The public export is therefore the carrier route itself with provenance and local NameCert rows visible; it is not open-cover compactness, quotient equality, choice, ambient completeness, or a host neighbourhood-base structure. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactPublicNeighbourhoodBaseExport`

Theorem 1726.18 (LocallyCompact abstract schema bridge surface)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. The public neighbourhood-base export of Definition 1726.17 is the bridge surface from the ten-row LocallyCompact carrier to the abstract local-compactness schema: it exports compact-ball access through B, K, local window access through X, x, r, optional located refinement through A, and structural transport through H, C, P, N. It exports no open-cover compactness, host neighbourhood-base selector, quotient point equality, countable choice, all-centers/all-radii properness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1726.1. The abstract schema can only see the displayed metric source, inspected point, positive radius, closed ball, compact witness, optional located handoff, transport, replay, provenance, and local naming rows.

Apply the public neighbourhood-base export, Definition 1726.17. It already identifies the exported public read with compact-ball access, center-radius local-cover access, componentwise transport, continuation replay, provenance, and local NameCert naming through exactly those rows.

Therefore the bridge surface is the public export itself. Since neither the carrier nor the public export contains open-cover compactness, a host selector, quotient equality, countable choice, global properness, or ambient Real completeness, the abstract local-compactness schema receives only the finite local compact-neighbourhood boundary. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactAbstractSchemaBridgeSurface`

Theorem 1726.19 (LocallyCompact mature compact-completion route)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every mature consumer that asks for compactness or completion first reads the local closed-ball window X, x, r, B, the compact witness K, the optional* $\mathsf{LocatedCompact}^{\uparrow}$ *handoff A, the properness nonescape boundary, and the compact-window transport rows H, C, P, N. Only after those rows are displayed may the consumer reuse the packet as a compact-completion route, and the route exports no open-cover compactness, host neighbourhood base, quotient point equality, countable choice, all-centers/all-radii properness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the compact-ball window of Definition 1726.11. It exposes one metric source, one inspected point, one positive radius ledger, one closed-ball row, and one compact witness before any compactness or completion-facing read is available.

Attach the located handoff using Definition 1726.12. The row A is optional, and when present it refines the same displayed closed-ball window rather than introducing an open-cover compactness principle or host neighbourhood base.

Next apply proper-metric nonescape, Definition 1726.13. A properness or completion consumer can reuse the compact window only through the displayed local packet, so the route does not become an all-centers/all-radii properness family.

Finally use compact-window transport, Definition 1726.14. The transport rows H, C, P, N preserve the same source, point, radius, closed-ball, compact witness, and located handoff, and therefore cannot add quotient equality, countable choice, or ambient Real completeness. The mature compact-completion route is exactly this composed finite window. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactMatureCompactCompletionRoute`

Theorem 1726.20 (LocallyCompact compact-window handoff)**.** *Let $L = (X, x, r, B, K, A, H, C, P, N)$ be an accepted* $\mathsf{LocallyCompact}^{\uparrow}$ *packet. Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$,

$\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *consumer that asks for compactness or completion must read the same displayed local window*

$$X, \quad x, \quad r, \quad B, \quad K, \quad A, \quad H, \quad C, \quad P, \quad N$$

before using the compactness or completion-facing surface. The handoff exports one compact closed-ball window with transport, replay, provenance, and local naming; it does not export open-cover compactness, global properness, a host neighbourhood base, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the compact-ball window, Definition 1726.11. It exposes the metric source, inspected point, positive radius, closed-ball row, and compact witness before any compactness-facing consumer can read the packet.

Attach the located row using Definition 1726.12. The $\mathsf{LocatedCompact}^{\uparrow}$ handoff is tied to the same X, x, r, B, K window, so it cannot introduce a second compact source or an open-cover compactness principle.

Next use proper-metric nonescape, Definition 1726.13. A $\mathsf{ProperMetric}^{\uparrow}$ or completion-facing consumer receives one displayed local compactness boundary, not an all-centers/all-radii properness theorem.

Finally compose compact-window transport with the mature compact-completion route, Definition 1726.14 and Definition 1726.19. The rows H, C, P, N preserve the same local window through transport, replay, provenance, and naming, so compactness and completion consumers can reuse only the finite window stated above. □

Lean4 ✓ `BEDC.Derived.LocallyCompactUp.LocallyCompactCompactWindowHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompact}^{\uparrow}$ packet is generated from a metric source row, an inspected point row, a positive RatUp or RealUp radius ledger, a MetricClosedBall row, a CompactMetric witness for the displayed window, an optional LocatedCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1726.1, Definition 1726.2, Definition 1726.3, Definition 1726.9, Definition 1726.10, Definition 1726.6, Definition 1726.16, Definition 1726.17, Definition 1726.18, and Definition 1726.19 bind the compact-neighbourhood public surface, abstract schema bridge, and downstream compact-completion example to displayed $\mathsf{Topology}^{\uparrow}$ neighbourhood rows, $\mathsf{CompactMetric}^{\uparrow}$ and finite-net rows, $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ radius rows, BHist, hsame, Cont, Pkg, and NameCert support.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocallyCompact}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, a host neighbourhood base, quotient point equality, countable choice, ambient Real completeness, global properness, Lean verification, or bridge checking.

Upgrade path: No further theory-axis upgrade is claimed beyond future bridge checking and formalization.

Chapter 1727

A Concrete Naming Certificate for $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ packet records the constructive zero-distance boundary for two regular Cauchy names by routing the displayed difference through a finite $\mathsf{NullSequence}^{\uparrow}$ tolerance-window witness before any $\mathsf{RealEquality}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits between the source rows of chapter 301, the pointwise difference surface of chapter 429, the zero-tail consumer of chapter 380, the equality boundary of chapter 1636, the finite windows of chapter 126, and the dyadic tolerance ledger of chapter 291.

The certificate is a finite equality boundary. It does not identify quotient streams, select limits, compare host reals, or invoke completeness; it records only the displayed route by which the regular-Cauchy difference is certified as a null sequence and then handed to the equality classifier.

Definition 1727.1 (RegularCauchyZeroDistance carrier)**.** *A* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, D, Z, E, W, T, H, C, P, N)$$

with the following displayed coordinates:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are compared.*
2. *D is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row displaying the pointwise difference stream $X - Y$ through aligned finite windows.*
3. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row consuming the displayed difference stream before equality is sealed.*
4. *E is the* $\mathsf{RealEquality}^{\uparrow}$ *classifier-consumption row that reads the null-difference packet and exposes the corresponding* $\mathsf{Real}^{\uparrow}$ *equality boundary.*
5. *W stores the* $\mathsf{StreamName}^{\uparrow}$ *observation windows used by the two source rows, the displayed difference row, and the null-sequence consumer.*
6. *T stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance requests, slack rows, and comparison budgets consumed by the difference and zero-tail rows.*
7. *H records componentwise* hsame *transport for source rows, difference rows, zero-tail rows, equality classifier rows, windows, tolerance ledgers, continuation routes, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a zero-distance request through $X, Y, D, Z, W, T, H, P, N$ and then into E.*
9. *P is the* Pkg *provenance over* RegularCauchyZeroDistance$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyDifference$^{\uparrow}$, NullSequence$^{\uparrow}$, RealEquality$^{\uparrow}$, Real$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyZeroDistance}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite zero-distance packet. It has no coordinate for quotient stream equality, host real equality, trichotomy, selected limits, countable choice, Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyZeroDistanceUp.RegularCauchyZeroDistanceTasteGate single carrier alignment`

Theorem 1727.2 (RegularCauchyZeroDistance NameCert obligations)**.** *For every accepted* RegularCauchyZeroDist *carrier $Q = (X, Y, D, Z, E, W, T, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyZeroDistance}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two* RegSeqRat$^{\uparrow}$ *source rows, the* RegularCauchyDifference$^{\uparrow}$ *displayed difference row, the* NullSequence$^{\uparrow}$ *zero-tail row, the* RealEquality$^{\uparrow}$ *classifier-consumption row, finite* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerances, transport, continuation, provenance, and local naming rows;*
2. *componentwise classifier stability for source rows, aligned windows, dyadic tolerances, the displayed difference route, the zero-tail route, the equality handoff, and the structural rows H, C, P, N;*
3. *zero-tail exactness, requiring Z to consume precisely the difference row D through the windows W and tolerance ledger T before the equality classifier is read;*
4. *RealEquality handoff, requiring E to read the* NullSequence$^{\uparrow}$ *packet as the equality evidence for the two regular Cauchy names and not as a host equality proof;*
5. *non-escape, requiring every public zero-distance read to factor through Q and forbidding quotient stream equality, host real equality, trichotomy, selected limits, countable choice,* Real$^{\uparrow}$ *completeness, and bridge checking from the local surface.*

No zero-distance obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1727.1. The packet displays the two regular rational source rows, the difference row, the null-sequence row, the equality classifier row, finite windows, dyadic tolerances, and structural provenance before any equality read is visible.

Classifier stability is componentwise. The source rows transport their regular rational observations, the windows transport the finite reads, the tolerance ledger transports the requested dyadic bounds, and the displayed difference row transports before the null-sequence row is consumed.

Zero-tail exactness comes next. The row Z can read only the displayed difference D through W and T, so the equality handoff has already passed through the finite null-difference package before E is inspected.

Finally inspect the carrier inventory. Since no coordinate of Q is a quotient stream equality, host real equality proof, trichotomy principle, selected limit, countable-choice schedule, completeness theorem, or bridge certificate, none can occur in the local NameCert surface. □

Theorem 1727.3 (RegularCauchyZeroDistance NullSequence criterion)**.** *Let $Q = (X, Y, D, Z, E, W, T, H, C, P, N)$ be an accepted* RegularCauchyZeroDistance$^{\uparrow}$ *carrier. A zero-distance read for the two regular Cauchy*

source rows is admitted exactly when the displayed difference route D factors through the NullSequence$^{\uparrow}$ *packet Z using the finite* StreamName$^{\uparrow}$ *windows W, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger T, the transport ledger H, the continuation replay C, the provenance row P, and the local naming row N before the* RealEquality$^{\uparrow}$ *classifier row E and the downstream* Real$^{\uparrow}$ *seal are consumed.*

Thus zero distance is the finite null-sequence-mediated equality boundary for the displayed difference stream, not a quotient equality of streams, host real equality, trichotomy comparison, selected limit, countable-choice construction, or completeness assertion.

Lean4 ✓ `BEDC.Derived.RegularCauchyZeroDistanceUp.RegularCauchyZeroDistanceNullsequenceCriterion`

Proof. For the forward direction, begin with an admitted zero-distance read. The NameCert obligations of Definition 1727.2 force the read through the carrier coordinates of Q.

The displayed difference row D is read before any equality classifier row. By zero-tail exactness, the consumer must then pass through the NullSequence$^{\uparrow}$ row Z, using exactly the windows W and tolerance ledger T transported by H and replayed by C.

Only after that finite null-difference package is visible may the classifier row E consume it as RealEquality$^{\uparrow}$ evidence. The provenance and local naming rows P, N record the same route and add no analytic data.

For the reverse direction, suppose the displayed difference route factors through Z with W, T, H, C, P, N before E is consumed. Then the carrier admission, zero-tail exactness, and RealEquality handoff clauses of Definition 1727.2 accept this route as the local zero-distance read.

The non-escape clause rules out all stronger interpretations. Since the packet contains no quotient stream equality, host real equality, trichotomy, selected limit, countable-choice row, or completeness theorem, the accepted read is exactly the stated finite boundary. □

Closure status for NameCert$_{\text{RegularCauchyZeroDistance}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyZeroDistance}^{\uparrow}}$, seed)

Scope closed: Definition 1727.1, Definition 1727.2, and Definition 1727.3 seed the finite zero-distance boundary for two regular Cauchy source rows through displayed difference, NullSequence zero-tail evidence, RealEquality classifier consumption, StreamName windows, and DyadicRatCore tolerance rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyZeroDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, host real equality, trichotomy, selected limits, countable choice, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, componentwise classifier stability, zero-tail exactness, RealEquality handoff, and non-escape for BEDC.Derived.RegularCauchyZeroDistanceUp.

Constructive story: A RegularCauchyZeroDistance$^{\uparrow}$ packet is generated from two RegSeqRat source rows, one RegularCauchyDifference route, one NullSequence zero-tail row, one RealEquality classifier-consumption row, finite StreamName windows, DyadicRatCore tolerances, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1728

A Concrete Naming Certificate for $\mathsf{RealPolynomial}^{\uparrow}$

Following Bishop and Bridges on constructive real functions and the standard polynomial-function construction, the $\mathsf{RealPolynomial}^{\uparrow}$ packet names finite polynomial evaluation over displayed $\mathsf{Real}^{\uparrow}$ names. It consumes the coefficient-list surface of chapter 50, the terminal real-name seal of chapter 38, the regular-Cauchy algebra route of chapter 1470, the finite window discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, and the continuous-map handoff of chapter 130. The chapter records polynomial evaluation as displayed BEDC finite data rather than host real-function syntax, quotient real functions, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The same finite Horner evaluation surface is consumed downstream by chapter 2269 for sign-window root isolation and by chapter 6454 for finite polynomial test rows. The exported rows are coefficient-list provenance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{ContinuousMap}^{\uparrow}$ handoff data; they do not expose host polynomial functions or selected roots.

Definition 1728.1 (Real polynomial carrier)**.** *A* $\mathsf{RealPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R_P = (A, X, Q, S, G, W, D, E, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite coefficient-list provenance row read from* $\mathsf{Poly}^{\uparrow}$*. It records the polynomial arity, coefficient order, constant term, successor-coefficient cells, and the finite list boundary used by the evaluator.*
2. *X is the input* $\mathsf{Real}^{\uparrow}$ *row whose regular name is read only through the displayed real-name seal.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the input real name and to every real coefficient name requested by the coefficient list.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It records the shared precision window used to inspect the input name and the finitely many coefficient names.*
5. *G is the* $\mathsf{RegularCauchyRing}^{\uparrow}$ *algebra row. It records the finite addition, multiplication, and powering schedule used by Horner evaluation, with no host real-operation coordinate.*
6. *W is the finite-window support row binding the coefficient list, input stream windows, regular readbacks, dyadic tolerances, and regular-Cauchy algebra operations to one evaluation request.*

7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores dyadic radii, endpoint refinements, multiplication slack, addition slack, and the final output tolerance budget.*
8. *E is the evaluation ledger. It records the finite Horner spine from the last coefficient toward the constant coefficient, with each intermediate value read through Q, S, G, W, D.*
9. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row. It exposes the same finite evaluation route as a continuous-map consumer surface once the evaluation ledger has been displayed.*
10. *H records componentwise* hsame *transport for coefficient provenance, input real names, regular readbacks, finite windows, regular-Cauchy algebra rows, dyadic tolerances, the evaluation ledger, the continuous-map handoff, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from each polynomial-evaluation request through $A, X, Q, S, G, W, D, E, M, H, P,$*
12. *P is the* Pkg *provenance over* $\mathsf{RealPolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyRing}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealPolynomial}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite polynomial-evaluation history. The classifier compares accepted packets only through $A, X, Q, S, G, W, D, E, M, H, C, P, N$, so it has no coordinate for host polynomial functions, quotient real functions, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealPolynomialUp.RealPolynomialUpTasteGate single carrier alignment`

Theorem 1728.2 (Real polynomial NameCert obligations)**.** *For every accepted* $\mathsf{RealPolynomial}^{\uparrow}$ *carrier*

$$R_P = (A, X, Q, S, G, W, D, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealPolynomial}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the coefficient-list provenance, input* $\mathsf{Real}^{\uparrow}$ *row, regular readbacks, finite windows, regular-Cauchy algebra, dyadic tolerance row, evaluation ledger, continuous-map handoff, transport, replay, provenance, and local naming rows;*
2. *coefficient normalization, requiring A to expose a finite* $\mathsf{Poly}^{\uparrow}$ *coefficient list before any evaluator row consumes coefficients;*
3. *finite evaluation route, requiring the Horner spine in E to read only Q, S, G, W, D and the already displayed coefficient cells of A;*
4. *classifier transport, requiring H to preserve componentwise sameness for coefficient provenance, input real names, finite windows, regular readbacks, regular-Cauchy algebra rows, dyadic tolerances, the evaluation ledger, and structural rows;*
5. *continuous-map handoff, requiring M to export only the same finite evaluation route already displayed by A, X, Q, S, G, W, D, E;*
6. *non-escape, refusing quotient real functions, host polynomial functions, selected limits, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No real polynomial obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 1728.1. The projection exposes exactly the coefficient-list row, input real row, regular readbacks, finite windows, regular-Cauchy algebra row, dyadic tolerance row, evaluation ledger, continuous-map handoff, and structural rows.

Coefficient normalization is read from A using the polynomial carrier of chapter 50. The row supplies the finite coefficient order, so evaluation never ranges over an unbounded host polynomial.

The finite evaluation route is the Horner spine recorded in E. At each step the input and coefficient rows are read through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, regular readbacks from chapter 301, finite windows from chapter 126, dyadic tolerances from chapter 291, and algebra rows from chapter 1470.

Classifier transport is componentwise. The row H preserves the coefficient provenance, input row, regular readbacks, finite windows, algebra schedule, tolerance row, and ledger before C, P, N replay, source, and name the same packet.

Finally use the continuous-map surface of chapter 130. The handoff row M receives only the displayed finite evaluation route, so quotient real functions, host polynomial functions, selected limits, trichotomy, countable choice, ambient completeness, Lean verification, and bridge checking have no local obligation row. □

Theorem 1728.3 (Real polynomial evaluation stability)**.** *Every accepted* $\mathsf{RealPolynomial}^{\uparrow}$ *evaluation route reads only the displayed coefficient provenance, input* $\mathsf{Real}^{\uparrow}$ *name, regular-Cauchy algebra, finite-window support, regular rational readback, dyadic tolerance, evaluation ledger, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, continuous-map handoff, and local* NameCert *rows. Consequently two accepted carriers matched componentwise by their displayed classifier data have the same finite evaluation route.*

Proof. Inspect an accepted carrier $R_P = (A, X, Q, S, G, W, D, E, M, H, C, P, N)$. The coefficient source is A, the real input is X, and the only analytic readback rows are Q, S, G, W, D, E, M.

The evaluation ledger E fixes the dependency order. It reads coefficient cells from chapter 50, real-name rows from chapter 38, algebra operations from chapter 1470, finite stream windows from chapter 126, regular rational readbacks from chapter 301, dyadic tolerance rows from chapter 291, and the continuous-map handoff from chapter 130.

Componentwise sameness in H transports each displayed row and no other row. The replay row C preserves the same dependency order, while P and N record provenance and local naming for the transported packet.

Thus any consumer route accepted by the local certificate is already one of the displayed projections. Since the carrier contains no coordinate for quotient real functions, host polynomial functions, selected limits, trichotomy, countable choice, or ambient completeness, stability cannot acquire such data under transport. □

Lean4 ✓ `BEDC.Derived.RealPolynomialUp.RealPolynomialEvaluationStability`

Closure status for $\mathsf{NameCert}_{\mathsf{RealPolynomial}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1728.1, Definition 1728.2, and Definition 1728.3 seed the finite polynomial-evaluation route over $\mathsf{Poly}^{\uparrow}$ coefficient provenance, $\mathsf{Real}^{\uparrow}$ input rows, $\mathsf{RegularCauchyRing}^{\uparrow}$ algebra, $\mathsf{ContinuousMap}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real functions, host polynomial functions, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient normalization, finite evaluation route, classifier transport, continuous-map handoff, and non-escape for BEDC.Derived.RealPolynomialUp.

Constructive story: A RealPolynomial$^{\uparrow}$ packet is generated from finite Poly$^{\uparrow}$ coefficient-list provenance, an input Real$^{\uparrow}$ name, RegSeqRat$^{\uparrow}$ readbacks, StreamName$^{\uparrow}$ windows, RegularCauchyRing$^{\uparrow}$ algebra rows, DyadicRatCore$^{\uparrow}$ tolerance rows, a finite Horner evaluation ledger, a ContinuousMap$^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1729

A Concrete Naming Certificate for $\mathsf{RealSequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealSequenceAlgebra}^{\uparrow}$ packet records the finite pointwise algebra surface for displayed $\mathsf{Real}^{\uparrow}$ sequences. It sits between the limit interface of chapter 1413, the limit-law consumers of chapter 1480, the bounded-source surface of chapter 1415, and the regular Cauchy addition route of chapter 1695. The packet names the finite sequence-algebra carrier used by later limit and boundedness consumers; it is not host sequence equality, a quotient stream construction, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1729.1 (Real sequence algebra carrier)**.** *A* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S_X, S_Y, W_X, W_Y, R_X, R_Y, D, O, E, H, C, P, N)$$

with the following displayed rows:

1. *S_X and S_Y are the two source* $\mathsf{Real}^{\uparrow}$ *sequence rows. Each source term is offered only through a displayed sequence route, not through host sequence equality.*
2. *W_X and W_Y are finite* $\mathsf{StreamName}^{\uparrow}$ *window rows for the two sources at the requested index and precision scale.*
3. *R_X and R_Y are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the displayed source windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation and error ledger. It records source tolerances, the operation tolerance, output tolerance, and the displayed budget for sum or product under the supplied bounds.*
5. *O is the pointwise operation row. It records whether the public read is a pointwise sum or a pointwise product under displayed bounds, and it cites the rows $S_X, S_Y, W_X, W_Y, R_X, R_Y, D$ before exposing an output readback.*
6. *E is the output* $\mathsf{Real}^{\uparrow}$ *seal reached only after the two source rows, finite windows, regular readbacks, dyadic ledger, and operation row have been displayed.*
7. *H records componentwise* hsame *transport for the two source sequence rows, both finite-window rows, both readback rows, the dyadic ledger, the operation row, the output seal, continuation replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a sequence-algebra request through the source rows, finite windows, readbacks, ledger, operation row, output seal, transport, package provenance, and local name.*

9. *P is the* Pkg *provenance over* $\mathsf{RealSequenceAlgebra}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RealSequenceLimitLaws}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequenceAlgebra}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite pointwise algebra packet. It has no coordinate for host equality of real sequences, quotient streams, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached host operation on sequences.*

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraTasteGate single carrier alignment`

Theorem 1729.2 (Real sequence algebra NameCert obligations)**.** *For every accepted* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *carrier*

$$A = (S_X, S_Y, W_X, W_Y, R_X, R_Y, D, O, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed source sequence rows, two finite window rows, two regular readback rows, dyadic operation and error ledger, pointwise operation row, output* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*

2. *classifier stability, requiring componentwise* hsame *transport to move the two source rows, windows, readbacks, ledger, operation row, output seal, and structural rows together;*

3. *operation-row exactness, requiring every pointwise sum or bounded pointwise product read to cite the dyadic budget stored in D and the operation choice stored in O;*

4. *seal handoff, requiring E to be read only after $S_X, S_Y, W_X, W_Y, R_X, R_Y, D, O$ have been displayed;*

5. *nonescape, excluding host sequence equality, quotient streams, selected limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking from the local packet.*

No pointwise sequence-algebra obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are the two source sequence rows, two finite windows, two regular rational readbacks, dyadic operation ledger, pointwise operation row, and output seal. The structural coordinates are the componentwise transport, continuation replay, package provenance, and local naming rows.

Classifier stability is componentwise. The row H transports exactly the displayed coordinates of the carrier, so a classified packet cannot exchange one source sequence, window, readback, ledger, or output seal while keeping the same public sequence-algebra read.

Operation-row exactness is the role of D and O. The ledger stores the source and output tolerances, while the operation row records whether the read is a pointwise sum or a pointwise product under displayed bounds. Thus every arithmetic read is mediated by the finite dyadic budget before an output seal is exposed.

Finally, C, P, N replay, source, and name the same route. Since no carrier coordinate stores host sequence equality, quotient streams, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraCarrier namecert obligations`

Theorem 1729.3 (Real sequence algebra limit consumer)**.** *Every* $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ *consumer that uses pointwise algebra on displayed* $\mathsf{Real}^{\uparrow}$ *sequences factors through an accepted* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *carrier. In the notation of Definition 1729.1, the consumer reads the finite route*

$$(S_X, S_Y) \longmapsto (W_X, W_Y) \longmapsto (R_X, R_Y) \longmapsto D \longmapsto O \longmapsto E \longmapsto H, C, P, N$$

before it invokes the limit-law handoff. The route exports no host sequence equality, quotient stream, selected limit, trichotomy, or ambient completeness coordinate.

Proof. Start with a limit-law consumer that asks for a pointwise algebra row. The source data visible to that consumer are the displayed sequence rows S_X, S_Y, not host functions on an ambient sequence type.

The finite windows and readbacks are opened next. The rows W_X, W_Y select the requested finite observations, while R_X, R_Y supply the regular rational readbacks used by the operation ledger.

The operation route then passes through D and O. These rows determine the tolerance accounting and the selected pointwise operation before the output seal E is available.

The remaining rows H, C, P, N transport, replay, source, and name this same finite packet. Thus the consumer reaches the $\mathsf{RealSequenceLimitLaws}^{\uparrow}$ handoff only through the sequence-algebra carrier, with no coordinate from which to read host sequence equality, quotient streams, selected limits, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraLimitConsumer`

Theorem 1729.4 (Real sequence algebra bounded consumer)**.** *Every* $\mathsf{BoundedRealSequence}^{\uparrow}$ *consumer of pointwise algebra reads bounds only through the displayed bounded-sequence surface and the regular Cauchy rows named by an accepted* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *carrier. In particular, a bounded pointwise product read must pass through the source sequence rows, finite windows, regular readbacks, dyadic bound ledger, operation row, output seal, and componentwise structural rows of Definition 1729.1.*

Proof. Begin with a bounded-sequence consumer of a pointwise algebra read. The boundedness information available to the consumer is the displayed $\mathsf{BoundedRealSequence}^{\uparrow}$ row and the regular rational readback data attached to the finite windows.

For a pointwise sum, the consumer reads the same source windows and regular readbacks before the dyadic ledger records the output tolerance. For a pointwise product, the consumer additionally reads the displayed bound entries through D and O before any output seal is exposed.

The output remains a $\mathsf{Real}^{\uparrow}$ seal named by E and transported through H, C, P, N. Since the carrier has no host sequence equality, quotient stream, selected limit, trichotomy, ambient completeness, Lean verification, or bridge coordinate, the bounded consumer can obtain bounds only from the displayed bounded-sequence and regular-Cauchy rows. □

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraBoundedConsumer`

Theorem 1729.5 (Real sequence algebra operation-row exactness)**.** *Every accepted* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *pointwise sum or bounded-product read factors through the two source* $\mathsf{Real}^{\uparrow}$ *sequence rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation and error ledger, the operation row, the output* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RealSequenceAlgebra}^{\uparrow}}$ *row. Thus the operation row is*

exact: it records the selected pointwise sum or bounded product together with its dyadic budget before any output seal or limit-law handoff is available.

Proof. Project the accepted carrier $A = (S_X, S_Y, W_X, W_Y, R_X, R_Y, D, O, E, H, C, P, N)$ of Definition 1729.1. The two source sequence rows are S_X, S_Y, and the only finite observations of them are the windows W_X, W_Y with readbacks R_X, R_Y.

The dyadic operation and error ledger is D. For a pointwise sum it records the source and output tolerances; for a bounded product it also records the displayed bound data needed by the product budget.

The operation row O is downstream of those source and ledger rows. It therefore cannot expose a sum or bounded product read without first naming the two sources, their finite windows, their regular readbacks, and the dyadic tolerance accounting.

Apply Definition 1729.3 and Definition 1729.4. The first confines limit-law consumers to the same route, and the second confines bounded consumers to the displayed bound and regular-Cauchy rows.

Finally E, H, C, P, N seal, transport, replay, source, and locally name this same route. Since no carrier coordinate stores host sequence equality, quotient streams, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the operation-row read is exactly the displayed finite ledger. □

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraCarrier operation row exactness`

Theorem 1729.6 (Real sequence algebra pointwise zero unit)**.** *Let $A = (S_X, S_Y, W_X, W_Y, R_X, R_Y, D, O, E, H, C,$ be an accepted* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *carrier. Suppose the second source row S_Y is the displayed zero* $\mathsf{Real}^{\uparrow}$ *sequence row and the operation row O is the pointwise sum row. Then the public output read exports the first source sequence through the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget, output* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route identifies the zero source only by displayed* $\mathsf{Real}^{\uparrow}$ *zero rows, not by host sequence equality, and it exports no quotient stream, selected limit, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The first source route is S_X, W_X, R_X, while the second source route is S_Y, W_Y, R_Y.

Use the displayed zero hypothesis only on S_Y. It says that the second source is read through zero $\mathsf{Real}^{\uparrow}$ rows inside the carrier; it does not identify host sequences or supply an ambient equality of functions.

Apply operation-row exactness, Definition 1729.5. Since O is the pointwise sum row, the output seal can be opened only after the dyadic ledger D has compared the first source route with the displayed zero route.

The dyadic budget and output seal therefore replay the first source finite windows through the same W_X, R_X, D, E path, with H, C, P, N transporting, replaying, sourcing, and naming that path. The zero row contributes no additional quotient, limit, trichotomy, or completeness coordinate.

Thus every consumer of the pointwise zero-unit read sees the first source sequence through the original finite StreamName and RegSeqRat rows together with the same dyadic ledger and output seal. No host sequence equality or ambient completeness principle is introduced. □

Lean4 ✓ `BEDC.Derived.RealSequenceAlgebraUp.RealSequenceAlgebraCarrier pointwise zero unit`

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceAlgebra}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSequenceAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1729.1, Definition 1729.2, Definition 1729.3, and Definition 1729.4 seed the finite pointwise sequence-algebra surface over displayed $\mathsf{Real}^{\uparrow}$ sequence rows, finite windows, regular readbacks, dyadic operation ledgers, bounded operation rows, and output seals.

Formal verification: FormalStatus(NameCert$_{\text{RealSequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host sequence equality, quotient streams, selected limits, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, operation-row exactness, seal handoff, nonescape, limit-law consumer factoring, and bounded-consumer bound routing for BEDC.Derived.RealSequenceAlgebraUp.

Constructive story: A RealSequenceAlgebra$^{\uparrow}$ packet is generated from two displayed Real$^{\uparrow}$ sequence rows, two finite StreamName$^{\uparrow}$ windows, two RegSeqRat$^{\uparrow}$ readbacks, a DyadicRatCore$^{\uparrow}$ operation and error ledger, a pointwise sum or bounded-product operation row, an output Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1730

A Concrete Naming Certificate for $\mathsf{LocatedRealComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedRealComparison}^{\uparrow}$ packet records the finite comparison surface for two located real presentations. It sits between **??**, chapter 1381, chapter 1380, chapter 304, chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the finite row surface by which a later order, density, or apartness consumer compares two displayed located-real endpoints without importing trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Definition 1730.1 (Located real comparison carrier)**.** *A* $\mathsf{LocatedRealComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, C, P, N)$$

with the following displayed rows:

1. *L_0 and L_1 are the two* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows. Each endpoint is read as a located real presentation before any comparison row is opened.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the common finite observation schedule used to inspect both endpoint presentations at the requested precision.*
3. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the windows of L_0 and L_1.*
4. *D_0 and D_1 are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows. They record the dyadic enclosures, slack budgets, and comparison tolerances used by the common window.*
5. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *row. It records the finite comparison request, its rational bracket data, and the displayed sign or overlap evidence available at the chosen window.*
6. *A is the optional positive-gap handoff row. When the comparison exposes a positive separation, A routes the same gap through the* $\mathsf{LocatedApartness}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$ *surfaces; when no positive gap is displayed, A records the absent handoff and exports no apartness witness.*
7. *H records componentwise* hsame *transport for both located-real endpoints, the shared window, both regular readbacks, both dyadic endpoint rows, the Bishop comparison row, the optional apartness handoff, continuation replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a comparison request through $L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet for two displayed located real endpoints. The classifier compares accepted packets only through $L_0, L_1, W, R_0, R_1, D_0, D_1, B, A,$ so it has no coordinate for trichotomy, total order of constructive reals, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonUp.LocatedRealComparisonUpTasteGate single carrier alignment`

Theorem 1730.2 (Located real comparison NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealComparison}^{\uparrow}$ *carrier*

$$K = (L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, shared finite window, two regular readbacks, two dyadic endpoint and tolerance rows, Bishop comparison row, optional apartness handoff, transport, replay, provenance, and local naming rows;*
2. *shared-window discipline, requiring both endpoints to be inspected through the same W before the comparison row B is public;*
3. *readback exactness, requiring R_0, R_1, D_0, D_1 to supply the rational and dyadic data consumed by B without exposing host endpoint values;*
4. *classifier transport, requiring H to preserve componentwise sameness for the endpoint rows, window, readbacks, dyadic ledgers, comparison row, optional handoff row, and structural rows;*
5. *positive-gap handoff, requiring A to export a* $\mathsf{LocatedApartness}^{\uparrow}$ *or* $\mathsf{ApartnessReal}^{\uparrow}$ *witness only when the same displayed comparison data in B and dyadic budget in D_0, D_1 exhibit a positive gap;*
6. *non-escape, refusing trichotomy, total order of constructive reals, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No located-real comparison obligation is stored outside these six rows.

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonUp.LocatedRealComparisonCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1730.1. The projection exposes exactly the two located-real endpoints, the shared window, the two regular readbacks, the two dyadic ledgers, the Bishop comparison row, the optional apartness handoff, and the structural rows.

The shared-window obligation is read from W. Since both endpoint presentations are inspected through the same finite schedule before B is public, a comparison consumer cannot compare one endpoint at a hidden precision or through an unrecorded stream request.

Readback exactness is the role of R_0, R_1, D_0, D_1. The regular rational rows supply finite endpoint observations, while the dyadic rows store the slack budgets and endpoint enclosures consumed by the Bishop comparison row.

Classifier transport is componentwise. The row H preserves the endpoint, window, readback, dyadic, comparison, and optional handoff rows before C, P, N replay, source, and name the same packet.

Finally, the positive-gap handoff is governed by A. A handoff to $\mathsf{LocatedApartness}^{\uparrow}$ or $\mathsf{ApartnessReal}^{\uparrow}$ may occur only when the already displayed comparison and dyadic rows exhibit the gap; otherwise the row records absence. Thus trichotomy, total order, quotient real equality, selected representatives, countable choice, ambient completeness, Lean verification, and bridge checking never become local obligations. □

Theorem 1730.3 (Located real comparison shared-window stability)**.** *Every accepted* $\mathsf{LocatedRealComparison}^{\uparrow}$ *packet reads both endpoints through the same displayed* $\mathsf{StreamName}^{\uparrow}$ *window before it exposes the comparison row. Consequently two accepted carriers matched componentwise by* $L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H,$ *have the same finite comparison route and the same available positive-gap handoff status.*

Proof. Inspect an accepted carrier $K = (L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, C, P, N)$. The only window coordinate is W, and both endpoint readbacks cite this same finite observation schedule.

The comparison row B is downstream of the endpoint rows, readbacks, and dyadic ledgers. Therefore changing the public comparison route would require changing at least one displayed coordinate among $L_0, L_1, W, R_0, R_1, D_0, D_1, B$.

The transport row H moves these coordinates componentwise, while C, P, N replay, source, and name the transported packet. The optional handoff A is part of the same displayed route, so its present or absent status is stable under the same classifier match. □

Theorem 1730.4 (Located real comparison apartness handoff)**.** *If an accepted* $\mathsf{LocatedRealComparison}^{\uparrow}$ *carrier exposes a positive-gap handoff row, then every downstream* $\mathsf{LocatedApartness}^{\uparrow}$ *or* $\mathsf{ApartnessReal}^{\uparrow}$ *consumer of that gap factors through the displayed route*

$$(L_0, L_1) \longmapsto W \longmapsto (R_0, R_1) \longmapsto (D_0, D_1) \longmapsto B \longmapsto A \longmapsto H, C, P, N.$$

If the handoff row is absent, the packet exports no apartness witness.

Proof. Start with a downstream consumer asking for apartness evidence from the comparison packet. The endpoint data available to that consumer are the displayed located-real rows L_0 and L_1, not host endpoint values.

The consumer must next read the common window W, the two regular readbacks R_0, R_1, and the dyadic endpoint budgets D_0, D_1. These rows supply the finite observations consumed by the Bishop comparison row B.

When B and the dyadic budgets exhibit a positive gap, the handoff row A routes that same evidence to the $\mathsf{LocatedApartness}^{\uparrow}$ and $\mathsf{ApartnessReal}^{\uparrow}$ surfaces. When the row is absent, no consumer can project an apartness witness from the packet.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Hence the handoff exports exactly the displayed positive-gap evidence and no trichotomy, total order, quotient real equality, selected representative, countable-choice selector, ambient completeness witness, or bridge data. □

Theorem 1730.5 (Located real comparison Real seal factorization)**.** *Every accepted* $\mathsf{LocatedRealComparison}^{\uparrow}$ *carrier*

$$K = (L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, C, P, N)$$

first reads the two $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows* L_0, L_1 *and then factors their terminal* $\mathsf{Real}^{\uparrow}$ *seals through the shared finite readback route*

$$(L_0, L_1) \longmapsto W \longmapsto (R_0, R_1) \longmapsto (D_0, D_1)$$

before exposing either the $\mathsf{BishopRealComparison}^{\uparrow}$ *row* B *or the optional positive-gap handoff row* A*. Thus any downstream order, density, or apartness consumer of* B *or* A *reaches the two real seals only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonUp.LocatedRealComparisonCarrier real seal factorization`

Proof. Project the carrier definition. The only endpoint coordinates are L_0 and L_1, and the comparison packet contains no separate host endpoint row or quotient-real representative from which B or A could be opened directly.

Apply Definition 1730.3. The shared W coordinate is the common $\mathsf{StreamName}^{\uparrow}$ window for both endpoints, and the comparison route is stable only through the displayed coordinates $L_0, L_1, W, R_0, R_1, D_0, D_1, B, A, H, C, P, N$.

The two regular readbacks R_0, R_1 and the two dyadic ledgers D_0, D_1 are the finite rows that connect the endpoint presentations to the comparison row. Hence the terminal $\mathsf{Real}^{\uparrow}$ seals of the located endpoints are visible to B only after this common readback route has been traversed.

Apply Definition 1730.4 for the positive-gap branch. If A is present, it exports only the same displayed gap already routed through W, R_0, R_1, D_0, D_1 and B; if A is absent, it exports no apartness witness.

Consequently the factorization refuses host endpoint equality, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and bridge data. The public comparison and handoff surfaces are exactly the finite certified route named by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1730.1, Definition 1730.2, Definition 1730.3, Definition 1730.4, and Definition 1730.5 seed the finite comparison route over two $\mathsf{LocatedReal}^{\uparrow}$ endpoints, a shared $\mathsf{StreamName}^{\uparrow}$ window, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, a $\mathsf{BishopRealComparison}^{\uparrow}$ row, and an optional $\mathsf{LocatedApartness}^{\uparrow}$ or $\mathsf{ApartnessReal}^{\uparrow}$ positive-gap handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparison}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedRealComparisonUp.LocatedRealComparisonUpTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, total order of constructive reals, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, readback exactness, classifier transport, positive-gap handoff, non-escape, and downstream apartness routing for BEDC.Derived.LocatedRealComparisonUp.

Constructive story: A $\mathsf{LocatedRealComparison}^{\uparrow}$ packet is generated from two displayed $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, one shared $\mathsf{StreamName}^{\uparrow}$ window, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance ledgers, a $\mathsf{BishopRealComparison}^{\uparrow}$ row, an optional positive-gap handoff through $\mathsf{LocatedApartness}^{\uparrow}$ or $\mathsf{ApartnessReal}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1731

A Concrete Naming Certificate for $\mathsf{KuratowskiEmbedding}^{\uparrow}$

Following Bourbaki's and Kelley's metric-space treatment of distance functions, the $\mathsf{KuratowskiEmbedding}^{\uparrow}$ packet records the classical embedding of a metric space into a space of distance-to-point functions as finite BEDC data. It sits over the metric source of chapter 57, the real-distance rows of chapter 38, and the distance-preserving interface of chapter 439. The packet exposes a selected basepoint or displayed family row, distance-function target rows, and the isometric route; it does not import quotient completion, host function extensionality, or a detached real-valued function space.

Definition 1731.1 (Kuratowski embedding carrier)**.** *A* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, B, D, T, I, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose point and distance-witness data supply the domain of the embedding.*
2. *B is the selected basepoint or displayed point-family row used to normalize and compare distance functions.*
3. *D is the* $\mathsf{Real}^{\uparrow}$ *distance-witness row recording the finite reads $d(x, b)$ and $d(y, b)$ against the selected basepoint or family.*
4. *T is the distance-function target row naming the displayed family of* $\mathsf{Real}^{\uparrow}$ *distance observations rather than a host function space.*
5. *I is the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *route row witnessing that source distances are preserved by the displayed distance-function target.*
6. *H records componentwise* hsame *transport for M, B, D, T, I.*
7. *C records displayed* Cont *replay from metric-distance reads to target distance-function reads and then to the isometric route.*
8. *P is the* Pkg *provenance over* $\mathsf{KuratowskiEmbedding}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{IsometricEmbedding}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, host function extensionality, ambient Banach-space structure, or a selected completion representative.

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingTasteGate single carrier alignment`

Theorem 1731.2 (Kuratowski embedding NameCert obligations)**.** *For the carrier of Definition 7569.1, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$ *surface consists of six finite rows:*

1. *carrier admission: every accepted packet exposes* $K = (M, B, D, T, I, H, C, P, N)$ *and no hidden function-space coordinate;*
2. *metric-source exposure: every source point and source-distance read enters through the* $\mathsf{Met}^{\uparrow}$ *row* M*;*
3. *basepoint or family selection: the row* B *is the only displayed source for the normalizing point or comparison family;*
4. *real-distance target exposure: the rows* D *and* T *turn source-distance reads into* $\mathsf{Real}^{\uparrow}$ *distance-function observations before any target comparison is made;*
5. *isometric route: the row* I *is the only distance-preserving handoff, and it is read after* M, B, D, T*;*
6. *structural replay: the rows* H, C, P, N *preserve, replay, source, and name the same route without exporting quotient completion or host function extensionality.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 7569.1. The accepted packet displays the metric source, basepoint or family row, real-distance witness row, target row, isometric route, transport, continuation, provenance, and local naming row.

Read the dependency rows in order. The metric source row M supplies point and distance-witness data; the row B supplies the selected basepoint or displayed family; and the real-distance row D records the corresponding $\mathsf{Real}^{\uparrow}$ observations.

The target row T is therefore a displayed distance-function surface, not a host function space. The isometric row I can only be read after those source and target observations are visible, and H, C, P, N preserve that same finite route. □

The route consumes the metric source of chapter 57, the real-distance rows of chapter 38, and the distance-preserving handoff of chapter 439.

Theorem 1731.3 (Kuratowski embedding distance route)**.** *Every accepted* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *packet routes source-distance comparison through*

$$M \to B \to D \to T \to I \to H \to C \to P \to N.$$

Thus a consumer reaches the $\mathsf{IsometricEmbedding}^{\uparrow}$ *handoff only after reading the* $\mathsf{Met}^{\uparrow}$ *source, the selected basepoint or family row, and the* $\mathsf{Real}^{\uparrow}$ *distance-function target rows. No consumer receives quotient completion, host function extensionality, ambient Banach-space structure, or a selected completion representative from the packet.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbeddingCarrier distance route`

Proof. Start with an accepted carrier $K = (M, B, D, T, I, H, C, P, N)$. A source-distance query must read M, and a distance-function query must next read the basepoint or family row B.

The real-valued observations are then stored in D and organized by T. Because T is part of the finite carrier, it supplies only displayed $\mathsf{Real}^{\uparrow}$ distance observations and not host-level function extensionality.

Finally apply the isometric route row I. The structural rows H, C, P, N keep the same route under transport and replay, so the only exported embedding data are the displayed metric, basepoint or family, real-distance, target, and isometric rows. □

Theorem 1731.4 (Kuratowski embedding distance readback exactness)**.** *Every accepted* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *packet reads source metric distance exactly through the displayed route*

$$M \to D \to T \to I.$$

The $\mathsf{Met}^{\uparrow}$ *row* M *supplies the source distance query, the* $\mathsf{Real}^{\uparrow}$ *row* D *supplies the distance witnesses used by the distance-function target* T*, and the* $\mathsf{IsometricEmbedding}^{\uparrow}$ *row* I *is the first row from which a completion, hyperspace, or distance-function consumer may inspect the resulting target comparison. No consumer may read a distance-function target as exact before the metric source, real-distance witnesses, and isometric handoff have all been displayed.*

Lean4 ✓ `BEDC.Derived.KuratowskiEmbeddingUp.KuratowskiEmbedding distance readback exactness`

Proof. Project the accepted carrier $K = (M, B, D, T, I, H, C, P, N)$ from Definition 7569.1. The source-distance query starts in M, while the basepoint or displayed family row B fixes the finite comparison context.

Apply the NameCert obligations of Definition 7569.2. They force every source-distance read through M and every real-valued distance observation through D before the target row T can be inspected.

Finally use the source-distance route of Definition 7569.3. The isometric handoff I is downstream of M, B, D, T, and the structural rows H, C, P, N only preserve that route. Thus completion, hyperspace, and distance-function consumers receive exactness through the displayed isometric readback rather than from an ambient function space. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KuratowskiEmbedding}^{\uparrow}$ packet is generated from a MetricUp source row, a selected basepoint or displayed family row, RealUp distance-witness rows, a distance-function target row, an IsometricEmbeddingUp route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 7569.1, Definition 7569.2, and Definition 7569.3 seed the finite distance-function embedding surface over MetricUp, RealUp, and IsometricEmbeddingUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion, host function extensionality, ambient Banach-space structure, selected completion representatives, or bridge checking.

Upgrade path: Lean closure requires checked carrier admission, basepoint or family exposure, RealUp distance target exposure, isometric-route preservation, and structural replay proofs for BEDC.Derived.KuratowskiEmbeddingUp.

Chapter 1732

A Concrete Naming Certificate for $\mathsf{HellySelection}^{\uparrow}$

Following Helly's selection theorem in the Bishop–Bridges constructive reading, the $\mathsf{HellySelection}^{\uparrow}$ packet records the finite subsequence-selection surface available for bounded-variation real-sequence data. It consumes the partition ledger of chapter 398, the displayed sequence algebra of chapter 1729, the finite selector handoff of chapter 1641, and the regular rational stream names of chapter 301. The packet names only finite windows, variation budgets, selector rows, regular rational readbacks, transport, replay, provenance, and local naming; it does not import open-cover compactness, trichotomy, quotient stream equality, ambient completeness, or choice-selected subsequences.

Definition 1732.1 (Helly selection carrier)**.** *A* $\mathsf{HellySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, A, W, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *row, including the finite partition-variation ledger and the displayed uniform variation budget read by the selection request.*
2. *A is the* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *row that exposes the real-sequence source only through pointwise algebra windows, not through host equality of streams.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger recording the requested source indices, rational scales, and finite observation schedule.*
4. *S is the* $\mathsf{SequentialCompactness}^{\uparrow}$ *selector row whose selected indices and selected windows are displayed as data of the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected windows and rational tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the selected-window read after B, A, W, S, R have been displayed.*
7. *T records componentwise* hsame *transport for the bounded-variation ledger, sequence-algebra row, finite windows, selector row, regular readback, and terminal seal.*
8. *C records displayed* Cont *replay from a bounded-variation selection request through the sequence-algebra row, finite windows, selector row, regular readback, real seal, and transport data.*
9. *P is the* Pkg *provenance over* $\mathsf{HellySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RealSequenceAlgebra}^{\uparrow}$, $\mathsf{SequentialCompactness}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, W, S, R, E, T, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, or choice-selected subsequences.

Lean4 ✓ `BEDC.Derived.HellySelectionUp.HellySelectionTasteGate single carrier alignment`

Theorem 1732.2 (Helly selection NameCert obligations)**.** *For every accepted* $\mathsf{HellySelection}^{\uparrow}$ *carrier*

$$H = (B, A, W, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, sequence-algebra row, finite stream windows, displayed selector row, regular rational readback, real seal, transport, replay, provenance, and local naming;*
2. *bounded-variation exposure, requiring the variation budget read by the selection request to pass through B before any selected window is inspected;*
3. *sequence-window exposure, requiring every source observation to pass through A and W before the selector row S is read;*
4. *selector stability, requiring the selected indices and selected windows to be exactly the rows displayed in S and transported componentwise by T;*
5. *regular readback exactness, requiring every selected-window convergence read to pass from S to R before the terminal seal E is available;*
6. *nonescape, excluding open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, Lean verification, bridge checking, and choice-selected subsequences.*

No Helly-selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are the bounded-variation row, the sequence-algebra row, the finite stream windows, the displayed selector row, the regular readback, and the terminal real seal. The structural coordinates are the componentwise transport, continuation replay, package provenance, and local naming rows.

The bounded-variation and window obligations are read in order. A consumer must first inspect the variation budget through B, then expose the real-sequence source through A and the finite observation schedule W.

Selector stability is internal to S and T. The selected indices and selected windows are named rows of the carrier, and the transport coordinate moves only those rows together with the visible source, window, readback, and seal data.

The regular readback and terminal seal are then forced by R and E. Since no carrier coordinate stores open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, Lean verification, bridge data, or choice-selected subsequences, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.HellySelectionUp.HellySelection namecert obligations`

Theorem 1732.3 (Helly selection bounded-variation window)**.** *Every accepted* $\mathsf{HellySelection}^{\uparrow}$ *packet reads its bounded-variation information through a finite window route*

$$B \longmapsto A \longmapsto W \longmapsto S.$$

The route exposes the source variation budget, sequence-algebra observations, requested stream windows, and displayed selector windows before any $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is available. It exports no open-cover compactness, trichotomy, quotient stream equality, ambient completeness, or choice-selected subsequence coordinate.*

Lean4 ✓ `BEDC.Derived.HellySelectionUp.HellySelection bounded variation window`

Proof. Begin with the bounded-variation source row B. Its visible data are finite partition ledgers and variation budgets, so the selection request starts from displayed bounded-variation information rather than an ambient compactness principle.

The sequence source is then opened through A and W. The row A supplies the pointwise real-sequence algebra surface, while W records the finite stream-name requests and rational scales used by the selector.

Only after these rows have been displayed can the selector row S be inspected. Thus the bounded-variation window route reaches the selected windows without introducing open-cover compactness, trichotomy, quotient stream equality, ambient completeness, or any choice-selected subsequence. □

Theorem 1732.4 (Helly selection finite-variation window handoff)**.** *Every accepted* $\mathsf{HellySelection}^{\uparrow}$ *packet must expose the* $\mathsf{BoundedVariation}^{\uparrow}$ *row, the* $\mathsf{RealSequenceAlgebra}^{\uparrow}$ *row, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any selected-subsequence consumer reads the route*

$$B, A, W \longmapsto S \longmapsto R, E.$$

The handoff is therefore a finite variation-window route: the consumer sees the displayed variation budget, source algebra observations, finite window schedule, selector row, regular rational tail budgets, and real seal in that order, not an ambient compactness principle or a choice-selected subsequence.

Lean4 ✓ `BEDC.Derived.HellySelectionUp.HellySelection finite variation window handoff`

Proof. Project the carrier of Definition 1732.1. The rows B, A, W are the only source-facing rows available before the selector is inspected, so a selected-subsequence consumer first reads the bounded-variation budget, sequence-algebra observations, and finite window schedule.

Apply Definition 1732.3. It forces the displayed route from B through A and W into S, and prevents a consumer from replacing the finite variation window by open-cover compactness, trichotomy, quotient stream equality, ambient completeness, or a choice-selected subsequence.

Finally apply Definition 1732.5. Once S has been displayed, every selected-window read passes through R and then E, with T, C, P, N preserving the same route. Thus the finite variation-window handoff is visible before any downstream subsequence read is exported. □

Theorem 1732.5 (Helly selection subsequence handoff)**.** *Every selected subsequence read exported by an accepted* $\mathsf{HellySelection}^{\uparrow}$ *carrier factors through the displayed selector row, regular rational handoff, terminal real seal, and structural rows*

$$S \longmapsto R \longmapsto E \longmapsto T, C, P, N.$$

The handoff names a finite selected-window route over $\mathsf{SequentialCompactness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *not a host-chosen subsequence or a quotient stream.*

Lean4 ✓ `BEDC.Derived.HellySelectionUp.HellySelection subsequence handoff`

Proof. Project the selector row S. Its selected indices and selected windows are part of the finite carrier, so a consumer cannot replace them by an external selection function.

The selected windows enter R together with the rational tail budgets. This is the only route by which the packet exposes a regular rational convergence read.

The terminal seal E is read after R, and the rows T, C, P, N transport, replay, source, and name the same route. Hence the subsequence handoff is finite and displayed, with no coordinate for a host-chosen subsequence or quotient stream equality. □

Theorem 1732.6 (Helly selection public export)**.** *The public boundary of* $\mathsf{HellySelection}^{\uparrow}$ *exports exactly the bounded-variation source row, real-sequence algebra row, finite stream windows, displayed selector row, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}$ *row of Definition 1732.1. It exports no open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, or choice-selected subsequences.*

Proof. Inspect the carrier coordinates. The public rows are precisely $B, A, W, S, R, E, T, C, P, N$.

The NameCert obligation theorem shows that every local obligation is attached to those rows. The bounded-variation window theorem shows that the variation and window data are read before the selector is inspected, and the subsequence handoff theorem shows that every selected-window read passes through S, R, E, T, C, P, N.

No additional row remains from which open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, or a choice-selected subsequence could be exported. Therefore the public boundary is exactly the finite displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1732.1, Definition 1732.2, Definition 1732.3, Definition 1732.5, and Definition 1732.6 seed the finite bounded-variation subsequence-selection surface over displayed variation ledgers, sequence-algebra windows, selector rows, regular rational readbacks, and terminal real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient stream equality, ambient completeness, host sequence equality, Lean verification, bridge checking, or choice-selected subsequences.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation exposure, sequence-window exposure, selector stability, regular readback exactness, nonescape, handoff factoring, and public-export proofs for BEDC.Derived.HellySelection

Constructive story: A $\mathsf{HellySelection}^{\uparrow}$ packet is generated from a bounded-variation source row, real-sequence algebra row, finite StreamName windows, displayed SequentialCompactness selector row, RegSeqRat readback, Real seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1733

A Concrete Naming Certificate for $\mathsf{SequenceFilterBridge}^{\uparrow}$

Following Bishop–Bridges regular-sequence completions and Kelley-style filterbase completions, the $\mathsf{SequenceFilterBridge}^{\uparrow}$ packet records the finite bridge by which displayed regular sequence completion data and displayed Cauchy-filter completion data read the same completion surface. It sits between the sequence carrier of chapter 406, the finite filterbase surface of chapter 1601, the filter completion handoff of chapter 1569, the sequential completeness boundary of chapter 1515, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a human-origin L10 Real and Completion bridge, not a quotient filter construction, selected limit, countable-choice selector, host equality comparison, or ambient completeness assumption.

Definition 1733.1 (Sequence-filter bridge carrier)**.** *A* $\mathsf{SequenceFilterBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, F, Q, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *row, including the displayed sequence entries, Cauchy-modulus rows, tail-comparison rows, and provenance boundary.*
2. *F is the accepted* $\mathsf{FilterBase}^{\uparrow}$ *row generated from the same finite tail windows, with base packets indexed only by displayed precision requests and eventual-tail witnesses.*
3. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion handoff read from F after the filterbase directedness and Cauchy rows have been displayed.*
4. *T is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule common to the sequence side and the filterbase side.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the sequence windows and reused by the filterbase completion handoff.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, F, Q, T, R have been displayed.*
7. *H records componentwise* hsame *transport for the sequence row, filterbase row, completion handoff, stream schedule, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from sequence-completion reads to filter-completion reads and back along the same finite window schedule.*

9. *P is the* Pkg *provenance over* $\mathsf{SequenceFilterBridge}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}$ $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequenceFilterBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, Q, T, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable-choice selectors, host equality, or ambient completeness of the target metric or $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequenceFilterBridgeUp.SequenceFilterBridgeUpTasteGate single carrier alignment`

Theorem 1733.2 (Sequence-filter bridge NameCert obligations)**.** *For every accepted* $\mathsf{SequenceFilterBridge}^{\uparrow}$ *carrier $B = (S, F, Q, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequenceFilterBridge}^{\uparrow}}$ *surface consists exactly of:*

1. *carrier admission for the displayed ten-row packet;*
2. *sequence exposure through S, including displayed entries, Cauchy-modulus rows, tail-comparison rows, and source provenance;*
3. *filterbase exposure through F, with base packets generated from the same finite tail windows and precision requests;*
4. *completion handoff through Q, requiring the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *read to cite the displayed filterbase rows of F;*
5. *shared-window readback through T and R, requiring both directions of the bridge to use the same* $\mathsf{StreamName}^{\uparrow}$ *schedule and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*
6. *Real and sequential-completeness compatibility through E and the* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *boundary, after the sequence, filterbase, completion, and readback rows have been displayed;*
7. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No NameCert obligation may be discharged by a quotient filter, selected limit, countable choice, host equality, or ambient completeness principle.

Proof. Project the carrier of Definition 1733.1. The rows $S, F, Q, T, R, E, H, C, P, N$ are the only coordinates available to the local classifier.

The sequence side is the row S. It exposes the finite Cauchy sequence data through chapter 406, while T and R make the inspected windows and regular rational readback visible before any completion-facing handoff is read.

The filter side is generated from the same windows. The row F records the $\mathsf{FilterBase}^{\uparrow}$ packets built from displayed tails, and Q reads the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ handoff only after those packets have been exposed.

The compatibility rows close the obligation surface. The row E seals the Real-facing read after the sequence, filterbase, completion, window, and readback rows have been displayed, while H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles have no coordinate in the carrier, the seven listed obligations exhaust the local NameCert surface. □

Theorem 1733.3 (Sequence-to-filter route)**.** *Every accepted sequence-completion read of a* $\mathsf{SequenceFilterBridge}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchySequenceSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow}.$$

In the notation of Definition 1733.1, the consumer reads S, then the shared schedule T and readback R, then the filterbase row F, and only then the completion handoff Q with its structural rows.

Proof. Start with a sequence-completion read of the bridge packet. By Definition 1733.1, the only sequence source coordinate is S, and every completion-facing observation from that coordinate is routed through the finite schedule T and readback R.

The route then forms the filterbase row. The base packets in F are indexed by the same displayed tail windows and precision requests, so no independent filter selector or quotient-filter equality is introduced.

Finally the completion handoff Q reads chapter 1569. The structural rows H, C, P, N record that this is the same finite packet seen from the sequence side, so every accepted sequence-completion read reaches the filter completion surface through the displayed route. □

Lean4 ✓ `BEDC.Derived.SequenceFilterBridgeUp.SequenceFilterBridgeSequenceToFilterRoute`

Theorem 1733.4 (Filter-to-sequence route)**.** *Every accepted Cauchy-filter completion read of a* $\mathsf{SequenceFilterBridge}^{\uparrow}$ *carrier factors back through the displayed sequence route*

$$\mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchySequenceSpace}^{\uparrow} \longmapsto \mathsf{SequentiallyCompleteMetric}^{\uparrow}.$$

The filter-facing consumer reads F, then the shared window schedule T and readback R, then the sequence row S, and then the sequential-completeness boundary named in P.

Proof. Begin with a filter-completion read. The row F supplies only the displayed finite filterbase packets, and Q records the completion handoff attached to those packets.

The bridge exposes the same finite observations through T and R. These rows determine which sequence windows and regular rational readbacks are visible, so the filter-facing route cannot choose an unrelated sequence representative.

The row S then supplies the $\mathsf{CauchySequenceSpace}^{\uparrow}$ packet, and P records the $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ boundary used by the completion consumer. The replay row C makes the reverse traversal explicit, while H keeps the sequence, filterbase, completion, and readback rows componentwise transported together. □

Lean4 ✓ `BEDC.Derived.SequenceFilterBridgeUp.SequenceFilterBridgeFilterToSequenceRoute`

Theorem 1733.5 (Sequence-filter bridge nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, *or* $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequenceFilterBridge}^{\uparrow}$ *carrier* $B = (S, F, Q, T, R, E, H, C, P$ *reads only the displayed sequence row, filterbase row, completion handoff, stream schedule, regular readback, Real seal, componentwise transport, continuation replay, provenance, and local NameCert row. The bridge exports no quotient filter, selected limit, countable-choice selector, host equality proof, or ambient completeness coordinate.*

Proof. Project the downstream rows from Definition 1733.1. The only rows exposed to Real, regular-rational, filter-completion, or sequential-completeness consumers are $S, F, Q, T, R, E, H, C, P, N$.

The sequence and filter presentations share their finite observations. The rows T and R attach the inspected windows and regular readbacks to both S and F, so neither side can escape to an unlisted representative, a quotient filter, or a selected limit.

The completion and Real-facing rows are downstream from the same finite route. The row Q is read only after F, and the row E is read only after S, F, Q, T, R have been displayed.

The structural rows do not add hidden analytic data. They only transport, replay, source, and name the displayed rows. Therefore countable choice, host equality, and ambient completeness have no carrier row, continuation edge, or provenance slot from which to enter the bridge. □

Falsifiable prediction: Every $\mathsf{SequenceFilterBridge}^{\uparrow}$ completion read factors through a displayed CauchySequenceSpace source row, shared StreamName schedule, RegSeqRat readback, FilterBase row, CauchyFilterCompletion handoff, Real seal, transport, replay, provenance, and local naming.

Independence witness: cauchysequencespace, filterbase, cauchyfiltercompletion, sequentially-completemetric, streamname, regseqrat, real: $\mathsf{SequenceFilterBridge}^{\uparrow}$ is not bijective to any listed sibling because it carries the two directed bridge routes between displayed sequence completion data and displayed Cauchy-filter completion data in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequenceFilterBridge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequenceFilterBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1733.1, Definition 1733.2, Definition 1733.3, Definition 1733.4, and Definition 1733.5 seed the finite bridge between regular sequence completion data and Cauchy-filter completion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequenceFilterBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence exposure, filterbase exposure, completion handoff, shared-window readback, Real and sequential-completeness compatibility, transport and replay stability, and nonescape for BEDC.Derived.SequenceFilterBridgeUp.

Constructive story: A $\mathsf{SequenceFilterBridge}^{\uparrow}$ packet is generated from a CauchySequenceSpace row, a FilterBase row on the same finite tail windows, a CauchyFilterCompletion handoff, a StreamName schedule, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1734

A Concrete Naming Certificate for $\mathsf{CauchyFilterCluster}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyFilterCluster}^{\uparrow}$ packet records the finite bridge by which displayed Cauchy-filter evidence is read as sequence-cluster data before a Real-facing seal may be consumed. It sits after the filter route of chapter 333, the retained-support boundary of chapter 1688, the uniform-window route of chapter 1694, and the cluster-row surface of chapter 1580. Its readback route is the L10 spine through chapter 301, chapter 126, chapter 291, and chapter 38. The packet gives Phase Real and Completion consumers a named finite handoff from filter data to cluster-window data without asserting a selected limit, quotient stream equality, countable choice, host real equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 1734.1 (CauchyFilterCluster carrier)**.** *A* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (F, M, U, W, Q, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *filter-base source row, including displayed base packets, refinements, Cauchy-modulus rows, and source provenance.*
2. *M is the* $\mathsf{MinimalCauchyFilter}^{\uparrow}$ *exposure row, retaining exactly the source base packets that a downstream cluster or Real-facing consumer reads.*
3. *U is the* $\mathsf{UniformCauchyFilter}^{\uparrow}$ *uniform-window row, naming the displayed uniform Cauchy windows attached to the retained support.*
4. *W is the cluster witness-window row. It lists the finite windows through which the same retained support is read as a* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *witness surface.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained support and witness windows.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used by W and Q.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger on the same windows, with endpoints, radii, slack rows, comparison rows, and refinement budgets visible in the packet.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after F, M, U, W, Q, S, D have been displayed.*
9. *H records componentwise* hsame *transport for the filter source, retained support, uniform windows, cluster windows, regular readback, stream schedule, dyadic ledger, Real seal, and structural rows.*

10. *C records displayed* Cont *replay from filter refinements to retained support rows, from retained support rows to uniform and cluster windows, and from regular readback to the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MinimalCauchyFilter}^{\uparrow}$, $\mathsf{UniformCauchyFilter}^{\uparrow}$, RealSequenceClusterPoi RegSeqRat$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, M, U, W, Q, S, D, E, H, C, P, N$. It has no coordinate for selected limits, quotient stream equality, host real equality, countable choice, ambient completeness, open-cover compactness, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyFilterClusterUp.CauchyFilterClusterTasteGate single carrier alignment`

Theorem 1734.2 (CauchyFilterCluster NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *carrier $Z = (F, M, U, W, Q, S, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed filter source, retained support, uniform-window, cluster-window, regular-readback, stream-schedule, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*
2. *filter-source exposure through F, with no downstream read before the displayed Cauchy-filter base and refinement rows have been inspected;*
3. *minimal-filter exposure through M, requiring each cluster or Real-facing window to cite a retained support row rather than the whole ambient filter base;*
4. *uniform Cauchy-window admission through U, on the same retained support used by the cluster route;*
5. *cluster witness-window admission through W, requiring every cluster witness window to read the retained support, stream schedule, dyadic ledger, and regular readback rows;*
6. *RegSeqRat, StreamName, and DyadicRatCore readback through Q, S, D on the same finite windows;*
7. *Real-seal handoff through E, only after the filter, support, uniform-window, cluster-window, readback, schedule, and tolerance rows have been displayed;*
8. *componentwise transport, continuation replay, package provenance, and local naming through H, C, P, N.*

No listed obligation may be discharged by selected limits, quotient stream equality, host real equality, countable choice, ambient completeness, open-cover compactness, or bridge checking.

Proof. Carrier admission is projection from Definition 1734.1. The rows $F, M, U, W, Q, S, D, E, H, C, P, N$ are the visible coordinates of one finite BHist packet, so the local certificate has no detached limit, quotient, choice, host-equality, compactness, or bridge coordinate.

The filter side is read first. The source row F supplies the Cauchy-filter base and refinement data, and the retained-support row M exposes exactly the base packets that later cluster or Real-facing consumers are allowed to inspect.

The window side is then fixed by U and W. Uniform Cauchy windows in U and cluster witness windows in W are attached to the same retained support, so the packet cannot route a cluster witness through a second ambient filter base.

The readback rows Q, S, D close the finite observation surface. They expose regular rational data, stream windows, and dyadic tolerance budgets on the same witness windows already listed by W.

Finally the Real-facing row E is downstream from those displayed rows, while H, C, P, N only transport, replay, package, and name the same route. Therefore the eight obligations exhaust the local $\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.CauchyFilterClusterUp.CauchyFilterCluster namecert obligations`

Theorem 1734.3 (CauchyFilterCluster cluster route)**.** *Every accepted* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *read exported by a* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *carrier* $Z = (F, M, U, W, Q, S, D, E, H, C, P, N)$ *factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{MinimalCauchyFilter}^{\uparrow} \longmapsto \mathsf{UniformCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegS}$$

Concretely, the consumer reads F, then M, then the uniform windows U, then the cluster witness windows W, and only then the stream, dyadic, and regular-rational readback rows S, D, Q with their structural support.

Proof. Start from an accepted cluster read of Z. By Definition 1734.1, the only cluster witness coordinate is W, and it is placed after the Cauchy-filter source F, retained support M, and uniform-window row U.

The source route reads chapter 333, chapter 1688, and chapter 1694. These chapters supply the filter-base rows, retained-support boundary, and uniform Cauchy windows that the cluster witness is permitted to cite.

The observation route then reads the finite schedule and tolerance rows. The row S supplies $\mathsf{StreamName}^{\uparrow}$ windows, D supplies $\mathsf{DyadicRatCore}^{\uparrow}$ budgets on those windows, and Q supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by the cluster row.

The structural rows preserve the same factorization. Componentwise transport through H, replay through C, provenance through P, and local naming through N do not create a second cluster route, so every accepted $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ read factors through the displayed chain. □

Lean4 ✓ `BEDC.Derived.CauchyFilterClusterUp.CauchyFilterClusterClusterRoute`

Theorem 1734.4 (CauchyFilterCluster Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *carrier* $Z = (F, M, U, W, Q, S, D, E, H, C, P, N)$ *reads the terminal seal only through the displayed filter source, retained support, uniform Cauchy windows, cluster witness windows, stream schedule, dyadic tolerance ledger, regular rational readback, componentwise transport, continuation replay, provenance, and local NameCert row. The packet exports no selected limit, quotient stream equality, host real equality, countable choice, ambient completeness, open-cover compactness, or bridge-checked standard schema.*

Proof. Project the Real-facing row from Definition 1734.1. The only seal coordinate is E, and the carrier places it downstream from F, M, U, W, Q, S, D.

The route to E is finite. The filter source and retained support determine which base packets may be inspected; the uniform and cluster windows determine which finite observations are visible; and the stream, dyadic, and regular-rational rows determine the readback consumed before the seal.

The structural coordinates cannot enlarge the seal boundary. The rows H, C, P, N transport, replay, package, and name the displayed route, but they contain no coordinate for a selected limit,

quotient stream equality, host real equality, countable choice, ambient completeness, open-cover compactness, or checked standard bridge.

Thus every $\mathsf{Real}^{\uparrow}$ consumer reaches the seal through the displayed route and through no external completion principle. □

Falsifiable prediction: Every $\mathsf{CauchyFilterCluster}^{\uparrow}$ cluster or Real-facing read factors through a displayed CauchyFilter source, MinimalCauchyFilter retained support, UniformCauchyFilter window row, RealSequenceClusterPoint witness-window row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming.

Independence witness: cauchyfilter, minimal-cauchy-filter, uniform-cauchy-filter, real-sequence-cluster-point, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyFilterCluster}^{\uparrow}$ is not bijective to any listed sibling because it carries the finite bridge from retained Cauchy-filter support to cluster witness windows before the Real seal is read.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCluster}^{\uparrow}$ packet is generated from a displayed CauchyFilter source row, MinimalCauchyFilter retained support row, UniformCauchyFilter window row, cluster witness-window row, RegSeqRat readback row, StreamName schedule, DyadicRatCore tolerance ledger, RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1734.1, Definition 1734.2, Definition 1734.3, and Definition 1734.4 seed the finite bridge from Cauchy-filter evidence to sequence-cluster and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCluster}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-source exposure, retained-support exactness, uniform-window admission, cluster witness-window admission, StreamName, DyadicRatCore, and RegSeqRat readback, Real-seal boundary, transport and replay stability, provenance, and local naming for BEDC.Derived.CauchyFilterClusterUp.

Chapter 1735

A Concrete Naming Certificate for $\mathsf{CompleteSeparableMetric}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{CompleteSeparableMetric}^{\uparrow}$ packet records the traditional complete separable metric surface as one finite L10 NameCert route. It joins the separable metric data of chapter 1458, the completion route of chapter 1714, the complete-metric boundary of chapter 128, and the metric-completion interface of chapter 1506. When the underlying metric carrier is Real-facing, the dense schedule can also be read through the $\mathsf{RealSeparability}^{\uparrow}$ route of chapter 1809. Its observable route is read through the finite windows of chapter 126, the rational readback discipline of chapter 301, the real boundary of chapter 38, and the downstream complete-separable consumer surface of chapter 1490. The same finite complete-separable metric surface is the handoff consumed by chapter 1490 and by the effective dense-realizer route of chapter 1706.

Definition 1735.1 (Complete separable metric carrier)**.** *A* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, K, D, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row. It records the metric carrier, distance classifier, and the* $\mathsf{SeparableMetric}^{\uparrow}$ *countable-name surface over the same point histories.*
2. *K is the complete-metric seal. It records the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness handoff or the* $\mathsf{MetricCompletion}^{\uparrow}$ *admission route that supplies completion-facing convergence data.*
3. *D is the separable dense schedule. It stores the countable dense enumeration, the request indices, and the finite visibility evidence needed to use density.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It is the only route through which dense indices, Cauchy observations, and completion-facing name segments are inspected.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *tolerance ledger. It carries dyadic scales, Real distance boundaries, comparison budgets, and the finite tolerance rows attached to W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the dense schedule and tolerance ledger before the complete-metric seal is read.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *boundary row. It records terminal distance or limit-boundary readings and has no coordinate for ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

8. *H records componentwise* hsame *transport for the metric source, complete-metric seal, dense schedule, finite windows, tolerance ledger, regular readback, Real boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from metric admission through dense-window reads, tolerance checks, regular readback, and complete-metric seal inspection.*
10. *P is the* Pkg *provenance over* $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{SeparableCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteSeparableMetric}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite complete-separable metric packet. It does not store quotient completion equality, selected dense points, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteSeparableMetricUp.CompleteSeparableMetricTasteGate single carrier alignment`

Theorem 1735.2 (Complete separable metric NameCert obligations)**.** *For every accepted* CompleteSeparableMetric *carrier $Q = (M, K, D, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteSeparableMetric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed metric source, complete-metric seal, dense schedule, finite windows, tolerance ledger, regular readback, Real boundary, transport, replay, provenance, and local naming rows;*
2. *complete-metric handoff, requiring the convergence or completion-facing seal K to be read from the same metric source that supplies M and never from a second host completion object;*
3. *countable-density visibility, requiring every density request to pass through the displayed schedule D and a finite* $\mathsf{StreamName}^{\uparrow}$ *window W;*
4. *tolerance discipline, requiring dyadic and Real comparison budgets in T before any completion-facing approximation is exposed;*
5. *regular readback, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to mediate finite rational observations before the complete-metric seal is inspected;*
6. *Real-seal nonescape, requiring the boundary row E to remain a terminal distance or limit reading rather than a proof of ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
7. *ledger nonescape, excluding quotient completion equality, selected dense points, countable-choice selectors, open-cover compactness, and any hidden completion equality from the packet.*

No complete-separable metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 1735.1. The projection exposes exactly the metric source M, complete-metric seal K, dense schedule D, finite window W, tolerance ledger T, regular readback R, Real boundary E, and structural rows H, C, P, N.

Carrier admission and complete-metric handoff are tied to the same metric source. The rows M and K are transported together by H, and the replay row C permits the completion-facing seal only after the source metric route has been displayed.

Density visibility is finite. A consumer reaches a dense candidate through the indexed schedule D and a displayed $\mathsf{StreamName}^{\uparrow}$ window W, then reads the tolerance ledger T and regular rational row R before consulting K or E.

The nonescape clauses follow from the carrier shape. None of the rows stores quotient completion equality, selected dense points, a countable-choice selector, open-cover compactness, hidden completion equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the listed obligations exhaust the local NameCert surface. □

Theorem 1735.3 (Complete separable metric density-completion handoff)**.** *Every* $\mathsf{PolishSpace}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *carrier* $Q = (M, K, D, W, T, R, E, H, C, P, N)$ *reads the finite route*

$$M, D \longmapsto W \longmapsto T \longmapsto R \longmapsto K \longmapsto E \longmapsto H, C, P, N$$

before it reads the public packet. Thus density, finite observation, tolerance, regular readback, and complete-metric seal data remain visible to the consumer, while quotient completion equality, selected dense points, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness remain outside the packet.*

Lean4 ✓ `BEDC.Derived.CompleteSeparableMetricUp.CompleteSeparableMetricCarrier density completion handoff`

Proof. Begin with a $\mathsf{PolishSpace}^{\uparrow}$ or completion-facing read of Q. Carrier admission gives the consumer the metric source and dense schedule only as the displayed pair M, D.

Replay the observation route. The request opens the finite $\mathsf{StreamName}^{\uparrow}$ window W, reads the dyadic or Real tolerance ledger T, and obtains the $\mathsf{RegSeqRat}^{\uparrow}$ readback R before the complete-metric seal K is available.

The Real boundary and public packaging are downstream. The row E records only terminal distance or limit-boundary data, while H, C, P, N transport, replay, source, and name the same finite route.

Therefore every downstream consumer reads the dense schedule, finite windows, tolerance ledger, regular readback, and complete-metric seal before the public packet. The excluded quotient, selected-point, choice, compactness, hidden-completion, and ambient-Real-completeness data have no coordinate in the carrier. □

Theorem 1735.4 (Complete separable metric finite-window density seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{PolishSpace}^{\uparrow}$*-facing read of an accepted* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *carrier* $Q = (M, K, D, W, T, R, E, H, C, P, N)$ *must expose the finite-window route*

$$M, D \longmapsto W \longmapsto T \longmapsto R \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

The row E *is only a terminal distance or limit-boundary row for the displayed route; it is not ambient* $\mathsf{Real}^{\uparrow}$ *completeness and cannot replace the complete-metric seal* K*.*

Lean4 ✓ `BEDC.Derived.CompleteSeparableMetricUp.CompleteSeparableMetricCarrier finite window density seal`

Proof. Project the carrier in Definition 1735.1. The projection exposes the metric source M, dense schedule D, finite window W, tolerance ledger T, regular readback R, complete-metric seal K, Real boundary row E, and structural rows H, C, P, N.

Apply Definition 1735.2. Density requests must pass through D and W, tolerance data must pass through T, and regular rational observations must pass through R before either completion-facing or Real-facing data are available.

Apply Definition 1735.3. The downstream consumer route already orders $M, D, W, T, R, K, E, H, C, P, N$, so every $\mathsf{Real}^{\uparrow}$ or $\mathsf{PolishSpace}^{\uparrow}$ read inherits the same finite-window density seal.

Finally inspect the role of E. The carrier assigns E only to terminal distance or limit-boundary readings, while K carries the complete-metric seal. Thus E cannot be read as ambient $\mathsf{Real}^{\uparrow}$ completeness or as a substitute for K. □

Theorem 1735.5 (Complete separable metric finite-density induction)**.** *For every accepted* CompleteSeparableMetr *carrier $Q = (M, K, D, W, T, R, E, H, C, P, N)$ and every displayed finite dense-prefix schedule inside the* $\mathsf{StreamName}^{\uparrow}$ *window row W, the prefix is consumed by induction through the same displayed rows. The empty prefix exposes only the metric source M, dense schedule D, and finite window W. If a prefix has already been consumed through M, D, W, T, R, then extending it by one dense index adds exactly that index from D, its dyadic or* $\mathsf{Real}^{\uparrow}$ *tolerance row in T, and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R before the same complete-metric seal K and terminal* $\mathsf{Real}^{\uparrow}$ *boundary row E are read. Thus every finite dense schedule prefix reaches completion-facing consumers through the route*

$$M, D, W \longmapsto T \longmapsto R \longmapsto K \longmapsto E$$

and not through quotient completion equality, selected dense points, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteSeparableMetricUp.CompleteSeparableMetricCarrier finite density induction`

Proof. Start with the carrier projection in Definition 1735.1. The displayed rows identify the only visible metric source, dense schedule, finite window, tolerance ledger, regular readback, complete-metric seal, and Real boundary rows.

The base case is the empty finite prefix. It makes no dense request, so Definition 1735.2 leaves visible only the source rows M, D and the finite window row W.

For the induction step, assume a finite prefix has already been consumed through M, D, W, T, R. Extending the prefix by one dense index opens one more displayed request in D through the same $\mathsf{StreamName}^{\uparrow}$ window W, then reads the attached dyadic or $\mathsf{Real}^{\uparrow}$ tolerance row in T and the matching $\mathsf{RegSeqRat}^{\uparrow}$ readback in R.

The completion-facing part does not change with the prefix length. By Definition 1735.3, the extended prefix still reaches the complete-metric seal through K and the terminal Real boundary through E before any public packet is exposed.

Apply Definition 1735.4. Since every finite-window read already has the route $M, D, W, T, R, K, E, H, C, P, N$, the induction consumes each finite dense schedule prefix through those rows and through no quotient completion equality, selected dense point, countable-choice selector, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteSeparableMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteSeparableMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1735.1, Definition 1735.2, Definition 1735.3, and Definition 1735.5 seed the finite complete-separable route through metric source data, complete-metric seals, dense schedules, finite windows, tolerance ledgers, regular readbacks, finite-prefix induction, and Real boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteSeparableMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected dense points, countable choice, open-cover compactness, hidden completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric handoff, countable-density visibility, tolerance discipline, regular readback, Real-seal nonescape, ledger nonescape, and density-completion handoff for BEDC.Derived.CompleteSeparableMetricUp.

Constructive story: A CompleteSeparableMetric$^{\uparrow}$ packet is generated from a metric and separable source row, a complete-metric or metric-completion seal, a countable dense schedule, finite StreamName windows, DyadicRatCore or RealUp tolerance data, a RegSeqRat readback, a terminal RealUp boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1736

A Concrete Naming Certificate for $\mathsf{UniformCauchySeries}^{\uparrow}$

Bishop–Bridges constructive-analysis grounding treats convergence of a real series as effective Cauchy control on finite partial sums rather than as an ambient infinite-sum object. The $\mathsf{UniformCauchySeries}^{\uparrow}$ packet records that shared-uniform-tail route as finite BEDC data, sitting between the finite series surface of chapter 1581, the fast tail-budget schedule of chapter 367, and the shared-threshold criterion of chapter 435. It routes all endpoint-facing reads through selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seal, with no host infinite sum, selected limit, quotient stream equality, countable choice, or ambient real completeness.

Definition 1736.1 (Uniform Cauchy series carrier)**.** *A* $\mathsf{UniformCauchySeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, S, W, Q, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the real-valued term stream, read only through selected finite observations and never as a completed host sequence.*
2. *S is the finite partial-sum window row, listing displayed initial segments, accumulated* $\mathsf{Real}^{\uparrow}$ *sum seals for those segments, and adjacent term contributions.*
3. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row for the terms, partial sums, and endpoint-facing observations used by the packet.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every selected term window and partial-sum window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording finite addition budgets, comparison radii, and requested tail tolerances.*
6. *M is the shared uniform Cauchy tail threshold row: one displayed threshold governs every accepted partial-sum comparison requested by the packet.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the term stream, partial-sum windows, selected stream names, regular readbacks, dyadic ledger, and shared threshold have been displayed.*
8. *H stores componentwise* hsame *transport for the term stream, partial-sum windows, selected stream windows, regular readbacks, dyadic ledger, tail threshold, endpoint seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from term reads to partial sums, from partial sums to regular readback, from readback to dyadic tolerance accounting, from the ledger to the shared uniform threshold, and from that threshold to the endpoint seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, W, Q, D, M, E, H, C, P, N$. It has no coordinate for a host infinite sum, a selected limit of partial sums, quotient equality of streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1736.2 (Uniform Cauchy series NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchySeries}^{\uparrow}$ *carrier $U = (T, S, W, Q, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed term-stream, partial-sum, stream-window, regular readback, dyadic ledger, shared-threshold, endpoint-seal, transport, replay, provenance, and naming rows;*

2. *classifier transport, requiring every accepted comparison to preserve $T, S, W, Q, D, M, E, H, C, P, N$ componentwise and to preserve no hidden summability datum;*

3. *uniform tail threshold, requiring the single row M to govern every displayed comparison of partial-sum windows beyond the requested tolerance;*

4. *partial-sum window exactness, requiring each finite endpoint-facing observation to pass through T, S, W, Q, and D before any tail or seal row is read;*

5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the endpoint row E to consume only the displayed finite route and never a host sum, chosen limit, quotient stream equality, countable choice, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.UniformCauchySeriesUp.TasteGate.UniformCauchySeriesTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1736.1. The packet displays exactly the rows $T, S, W, Q, D, M, E, H, C, P, N$, so the local certificate has no detached coordinate for a completed series or a choice-selected endpoint.

Classifier transport is componentwise. The row H stores the permitted hsame comparisons, while C replays only the displayed route from term reads through partial sums, readbacks, dyadic budgets, the shared tail threshold, and the endpoint seal.

The tail-threshold row is unique inside the packet. A tolerance request may ask for many finite partial-sum comparisons, but those comparisons all read the same displayed threshold row M before reaching the endpoint-facing seal.

Partial-sum exactness follows from the order of the carrier rows. The term stream T and partial-sum windows S are exposed through the selected windows W, then through the regular readback Q and dyadic ledger D.

Finally, the $\mathsf{Real}^{\uparrow}$ seal row E is downstream of the same finite route. Since no host infinite sum, selected limit, quotient stream equality, countable-choice selector, or ambient completeness theorem appears among the carrier coordinates, none can be exported by the local $\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}$ surface. □

Theorem 1736.3 (Uniform tail-threshold nonescape)**.** *Every endpoint-facing read from an accepted* UniformCauchySeries$^{\uparrow}$ *carrier factors through the displayed* RealSeries$^{\uparrow}$, FastConvergentSeries$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows named by the packet. Consequently the read cannot escape through a host infinite sum, selected limit, quotient stream equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchySeriesUp.TailThreshold.UniformCauchySeriesTailThresholdSemanticCertif`

Proof. Start with an endpoint-facing request. The carrier definition forces the request to read the real-valued term stream and finite partial-sum windows through the RealSeries$^{\uparrow}$ provenance row and the displayed rows T and S.

Next the request reads the fast tail-budget and shared criterion surfaces. The FastConvergentSeries$^{\uparrow}$ provenance supplies the finite tail-budget discipline, while the UniformCauchyCriterion$^{\uparrow}$ provenance supplies the single shared threshold row M for all accepted partial-sum comparisons.

The finite observations are then exposed only through W, Q, and D. These are respectively the selected StreamName$^{\uparrow}$ windows, the RegSeqRat$^{\uparrow}$ readbacks, and the DyadicRatCore$^{\uparrow}$ tolerance ledger.

The endpoint row E is a Real$^{\uparrow}$ seal for this already displayed route. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any additional endpoint coordinate.

Because every stage of the read is one of the displayed sibling rows, the packet has no place from which to read a host infinite sum, selected limit, quotient stream equality, countable-choice selector, or ambient completeness theorem. □

Theorem 1736.4 (Uniform Cauchy series majorant sum handoff)**.** *Let* $U = (T, S, W, Q, D, M, E, H, C, P, N)$ *be an accepted* UniformCauchySeries$^{\uparrow}$ *carrier. Every endpoint-facing read whose term stream is supplied by the* RealSeries$^{\uparrow}$ *surface and whose tail control is supplied by a displayed* CauchyMajorantSeries$^{\uparrow}$ *or* FastConvergentSeries$^{\uparrow}$ *route factors through*

$$T \longmapsto S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

Here M is read as the shared tail threshold for the majorant-backed partial-sum comparisons, and E is read only after the selected StreamName$^{\uparrow}$ *windows, the* RegSeqRat$^{\uparrow}$ *readbacks, and the* DyadicRatCore$^{\uparrow}$ *ledger have already displayed the finite route to the endpoint* Real$^{\uparrow}$ *seal. Consequently the handoff exports the endpoint Real sum route without exporting a host infinite sum, selected limit, quotient stream equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $U = (T, S, W, Q, D, M, E, H, C, P, N)$. The RealSeries$^{\uparrow}$ surface supplies the visible term stream and partial-sum observations, so the endpoint request begins at T and S rather than at an external summability predicate.

The majorant-backed tail control is then read through the displayed sibling route. The CauchyMajorantSeries$^{\uparrow}$ surface contributes the finite majorant schedule, while the FastConvergentSeries$^{\uparrow}$ surface contributes the fast tail-budget discipline; both are consumed by the single shared threshold row M named in the UniformCauchySeries$^{\uparrow}$ carrier.

Before the endpoint row is visible, the request must pass through W, Q, and D. These rows open the selected StreamName$^{\uparrow}$ windows, read them as RegSeqRat$^{\uparrow}$ data, and account for the requested dyadic tolerances through DyadicRatCore$^{\uparrow}$.

Finally apply Definition 1736.3. Since the endpoint row E is downstream of the displayed threshold and ledger route, the structural rows H, C, P, N can only transport, replay, source, and

name the same finite handoff. No carrier row remains from which a host infinite sum, selected limit, quotient stream equality, countable-choice selector, or ambient completeness theorem could be read. □

Falsifiable prediction: Every $\mathsf{UniformCauchySeries}^{\uparrow}$ endpoint-facing read factors through finite partial-sum windows, one shared uniform Cauchy tail threshold, selected StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, fastconvergentseries, uniformcauchycriterion, regseqrat, streamname, dyadicratcore, real: $\mathsf{UniformCauchySeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a real-valued term stream, finite partial-sum windows, one shared uniform tail threshold, selected stream windows, regular readbacks, dyadic tolerance accounting, endpoint sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchySeries}^{\uparrow}$ packet is generated from a real-valued term stream, finite partial-sum windows, selected StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, one shared uniform Cauchy tail threshold, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1736.1, Definition 1736.2, Definition 1736.3, and Definition 1736.4 seed the finite shared-uniform-tail series route from real-valued term streams and displayed majorant control to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchySeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, uniform tail thresholding, partial-sum window exactness, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.UniformCauchySeriesUp.

Chapter 1737

A Concrete Naming Certificate for $\mathsf{CauchyApproximationSequence}^{\uparrow}$

Bishop–Bridges constructive analysis treats a real name as a precision-indexed approximation practice: a requested tolerance is answered by displayed rational data and an explicit regularity route, not by a completed host value. The $\mathsf{CauchyApproximationSequence}^{\uparrow}$ packet records that finite request-to-seal surface between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the regular completion boundary of chapter 1639. It gives later Real and Completion consumers one named route from precision requests to finite approximants, while refusing quotient stream equality, selected limits, countable choice, host real equality, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1737.1 (Cauchy approximation sequence carrier)**.** *A* $\mathsf{CauchyApproximationSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the requested-precision row, listing the finite tolerance questions currently being answered by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximant ledger, recording dyadic approximants, dyadic radius comparisons, and the finite error budgets attached to each requested precision.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window schedule opened for the requested precisions and no other hidden observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row obtained from the displayed stream windows and dyadic approximant ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, D, S, R have been exposed and replayed.*
6. *H records componentwise* hsame *transport for requested precisions, dyadic approximants, stream-window schedules, regular readbacks, the Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through D, then through S and R, and finally into the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, S, R, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1737.2 (Cauchy approximation sequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyApproximationSequence}^{\uparrow}$ *carrier $A = (Q, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the requested-precision row, dyadic approximant ledger, stream-window schedule, regular-rational readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *request exactness, requiring each precision question to be read from Q before any dyadic approximant in D is accepted;*

3. *dyadic ledger exactness, requiring D to record the approximant and tolerance comparison used for the displayed request, not an ambient real value;*

4. *window and readback exactness, requiring S and R to expose only the finite stream windows and regular rational data demanded by Q and justified by D;*

5. *Real-seal boundary, requiring E to remain downstream of Q, D, S, R and never to serve as a selected limit or host equality object;*

6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed route from requested precision to Real seal;*

7. *provenance and local naming, requiring P and N to export the finite route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-sequence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyApproximationSequenceUp.CauchyApproximationSequenceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1737.1. The accepted packet displays exactly the requested precisions Q, dyadic ledger D, stream windows S, regular readback R, Real seal E, and structural rows H, C, P, N.

Request exactness and dyadic ledger exactness follow from the route order. A consumer first reads a finite precision question from Q; only then may it consult D for the dyadic approximant and tolerance comparison attached to that question.

The window and readback rows are likewise downstream of the request and ledger. The schedule S opens only the finite windows demanded by Q, and R records the regular-rational data read through those windows under the displayed dyadic budget.

Finally, the Real seal row E is terminal for this finite route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 1737.3 (Cauchy approximation sequence Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationSequence}^{\uparrow}$ *carrier reads only the displayed request-to-seal rows*

$$Q, D, S, R, E, H, C, P, N.$$

Consequently the consumer receives finite requested precisions, dyadic approximants, stream-window schedules, regular-rational readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It receives no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a $\mathsf{Real}^{\uparrow}$ consumer read an accepted packet $A = (Q, D, S, R, E, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 1737.1.

The read begins with a requested precision in Q. The dyadic ledger D supplies the corresponding approximant and tolerance comparison; the stream schedule S opens the finite observation windows; and the regular readback R exposes the rational data that the Real seal may consume.

The terminal row E is therefore reached only after the finite route through Q, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationSequence}^{\uparrow}$ Real-facing read factors through finite requested precisions, a DyadicRatCore approximant ledger, selected StreamName windows, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishopregularcauchycompletion: $\mathsf{CauchyApproximationSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries requested precisions, dyadic approximant rows, finite stream-window schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one request-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyApproximationSequence}^{\uparrow}$ packet is generated from requested precision rows, a DyadicRatCore approximant ledger, selected StreamName windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1737.1, Definition 1737.2, and Definition 1737.3 seed the finite request-indexed route from dyadic precision questions to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exactness, dyadic ledger exactness, window and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationSequenceUp.

Chapter 1738

A Concrete Naming Certificate for $\mathsf{CauchyTailEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyTailEquivalence}^{\uparrow}$ packet records equality of regular Cauchy real names by a shared finite tail route rather than by quotienting streams. It sits over the regular rational source rows of chapter 301, the selected-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the synchronized tail criterion of chapter 435, and the terminal real-name seal of chapter 38. The packet names the symmetric finite carrier consumed by the Real and Completion route: two regular Cauchy sources, one shared tail threshold, finite tolerance accounting, componentwise transport, and a Real-facing seal handoff with no quotient stream equality, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1738.1 (CauchyTailEquivalence carrier)**.** *A* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E_{\mathrm{tail}} = (X, Y, W, M, D, T, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational Cauchy observations are compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming exactly the finite source observations used on both sides of the comparison.*
3. *M is the shared* $\mathsf{StreamName}^{\uparrow}$ *tail threshold row; every accepted tolerance request for X and Y must read this one row before comparing tail windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested precision, the source radii, the tail slack, and the final comparison budget.*
5. *T is the finite tail-equivalence row asserting that beyond M the displayed X and Y observations meet the dyadic budget recorded in D.*
6. *H stores componentwise* hsame *transport for the two source rows, shared windows, shared tail threshold, dyadic ledger, tail-equivalence row, continuation, provenance, and local naming.*
7. *C stores displayed* Cont *replay from a Real-facing equality request through $X, Y, W, M, D, T, H, P, N$ before any seal is consumed.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, M, D, T, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyTailEquivalenceUp.CauchyTailEquivalenceUpTasteGate single carrier alignment`

Theorem 1738.2 (CauchyTailEquivalence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *carrier*

$$E_{\text{tail}} = (X, Y, W, M, D, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger, the shared tail-threshold row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the tail-equivalence row, and the structural rows;*
2. *symmetric source stability, requiring interchange of X and Y to preserve the same shared window, threshold, tolerance, tail-equivalence, transport, replay, provenance, and naming rows;*
3. *shared-threshold exactness, requiring every accepted comparison to read the single threshold row M before the dyadic budget D is discharged by T;*
4. *componentwise* hsame *transport, requiring H to preserve the two source rows, the shared windows, the threshold row, the tolerance ledger, and the tail-equivalence row without adding a quotient coordinate;*
5. *displayed* Cont *replay, requiring C to route each Real-facing request through $X, Y, W, M, D, T, H, P, N$ before the consumer reaches a* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *refusal of quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No tail-equivalence obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.CauchyTailEquivalenceUp.TasteGate.CauchyTailEquivalenceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1738.1. The packet displays exactly the two regular Cauchy source rows, shared windows, one shared threshold, dyadic tolerance accounting, and structural naming rows.

Symmetry is finite because the carrier stores no oriented endpoint datum. Interchanging X and Y leaves W, M, D, T, H, C, P, N as the same displayed route, so the local classifier has no extra quotient or host-equality coordinate to transport.

Shared-threshold exactness follows from the order of reads in the carrier. A tolerance request must pass through W, then through the single row M, then through the dyadic ledger D before the tail-equivalence row T is available.

The transport and replay clauses are componentwise. The row H preserves the named source, window, threshold, tolerance, and tail rows, while C replays the same route for a Real-facing consumer.

Finally inspect the carrier inventory. Since quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from E_{tail}, none can appear in the local $\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}$ surface. □

Theorem 1738.3 (CauchyTailEquivalence tail transport)**.** *Let $E_{\text{tail}} = (X, Y, W, M, D, T, H, C, P, N)$ be an accepted* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *carrier. Any finite tail-equivalence read admitted for X and Y transports along componentwise* hsame *data exactly when the transported read preserves the shared* $\mathsf{StreamName}^{\uparrow}$ *windows W, the shared tail threshold M, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, the tail-equivalence row T, the displayed* Cont *replay C, the* Pkg *provenance P, and the local naming row N.*

Consequently the transported read remains a finite tail comparison between the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows. It is not a quotient equality of streams, a selected limit, a countable-choice construction, a trichotomy comparison, or an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyTailEquivalenceUp.CauchyTailEquivalenceTailTransport`

Proof. For the forward direction, begin with an admitted tail-equivalence read. The NameCert obligations of Definition 1738.2 force the read through the carrier rows $X, Y, W, M, D, T, H, C, P, N$.

The componentwise transport row H can preserve only those displayed rows. Therefore an accepted transported read keeps the same selected windows, shared threshold, dyadic ledger, tail-equivalence row, continuation replay, provenance, and local naming.

For the reverse direction, suppose those rows are preserved by componentwise hsame data. Carrier admission and displayed replay then reconstruct the same finite tail route through W, M, D, T, C, P, N, so the local classifier admits the transported comparison.

The nonescape clause rules out stronger interpretations. Because the carrier contains no quotient coordinate, selected limit, choice selector, trichotomy row, or completeness theorem, transport cannot export any of them. □

Theorem 1738.4 (CauchyTailEquivalence Real seal handoff)**.** *Every Real-facing equality read exported by an accepted* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *carrier factors through the finite route*

$$(X, Y) \longmapsto W \longmapsto M \longmapsto D \longmapsto T \longmapsto H \longmapsto C \longmapsto P \longmapsto N \longmapsto \mathsf{Real}^{\uparrow}$$

before it may be consumed by a Real or Completion endpoint. Thus the consumer receives two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, one shared* $\mathsf{StreamName}^{\uparrow}$ *tail threshold, selected source windows, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data. It receives no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with a Real-facing equality request. The carrier definition requires the request to identify the two regular Cauchy source rows X and Y before any endpoint seal is available.

The request then reads the selected source windows W and the shared tail threshold M. Only after that shared threshold is displayed may the dyadic ledger D and tail-equivalence row T certify the requested comparison.

The structural rows do not add analytic data. They transport the finite route componentwise, replay it as a displayed continuation, record its provenance, and name it locally.

The final handoff to $\mathsf{Real}^{\uparrow}$ is therefore downstream of the finite route just listed. Since no quotient stream equality, selected limit, countable-choice selector, trichotomy principle, host real equality, or ambient completeness theorem appears among the rows, none is delivered to the Real or Completion consumer. □

Theorem 1738.5 (CauchyTailEquivalence Real-seal nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *packet must consume the terminal seal only after the finite route*

$$X, Y, W, M, D, T \longmapsto H, C, P, N.$$

The seal therefore cannot be reinterpreted as quotient stream equality, selected limit data, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the handoff theorem Definition 1738.4. It already routes every Real-facing equality read through the two regular rational source rows, selected windows, shared threshold, dyadic ledger, tail-equivalence row, transport, replay, provenance, and local naming before the Real endpoint is used.

Use the NameCert obligations of Definition 1738.2. They keep $X, Y, W, M, D, T, H, C, P, N$ as the complete local certificate surface and store no quotient, selector, trichotomy, host equality, or completeness coordinate.

Thus the Real seal is a terminal row of the displayed finite packet. Any consumer that treats it as an external equality or completion principle has left the accepted $\mathsf{CauchyTailEquivalence}^{\uparrow}$ route. □

Falsifiable prediction: Every $\mathsf{CauchyTailEquivalence}^{\uparrow}$ Real-facing equality read factors through two RegSeqRat source rows, selected StreamName windows, one shared StreamName tail threshold, a DyadicRatCore tolerance ledger, finite tail-equivalence data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, uniformcauchycriterion, real: $\mathsf{CauchyTailEquivalence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular Cauchy sources, one shared tail threshold, symmetric tail comparison, dyadic tolerance accounting, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, selected $\mathsf{StreamName}^{\uparrow}$ windows, one shared $\mathsf{StreamName}^{\uparrow}$ tail threshold, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite tail-equivalence row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1738.1, Definition 1738.2, Definition 1738.3, and Definition 1738.4 seed the finite symmetric tail-equivalence carrier for the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailEquivalence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyTailEquivalenceUp.CauchyTailEquivalenceUpTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, symmetric source stability, shared-threshold exactness, componentwise transport, displayed replay, and Real seal handoff for BEDC.Derived.CauchyTailEquivalenceUp.

Chapter 1739

A Concrete Naming Certificate for $\mathsf{UniqueChoicePrinciple}^{\uparrow}$

Following Bishop–Bridges constructive analysis, unique relation readback is admitted only through displayed existence and uniqueness rows. The $\mathsf{UniqueChoicePrinciple}^{\uparrow}$ packet records that finite surface as a NameCert carrier: a source row and relation row are not turned into a selected representative or host function, but into a deterministic readback row whose use is justified by visible existence evidence and a uniqueness ledger. It sits beside the extension consumers of chapter 769 and chapter 1632, the completion interface of chapter 397, and the separated comparison boundary of chapter 1411.

The packet is a finite relation-to-readback certificate. It does not assert countable choice, selected representatives, host function equality, quotient completion, real trichotomy, or ambient completeness; it records only the displayed route by which an accepted source and relation row expose one deterministic readback.

Definition 1739.1 (UniqueChoicePrinciple carrier)**.** *A* $\mathsf{UniqueChoicePrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, E, U, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the source carrier row whose accepted elements may be queried.*
2. *R is the relation row over S and the readback surface, stored as finite classifier data rather than as a host function.*
3. *E is the existence witness row. For each accepted source read, it displays a relation output row before deterministic readback is visible.*
4. *U is the uniqueness ledger. It compares any two displayed relation outputs for the same accepted source by componentwise* hsame *transport and classifier agreement.*
5. *D is the deterministic readback row obtained only after E and U have been displayed.*
6. *H records componentwise* hsame *transport for source rows, relation rows, existence rows, uniqueness-ledger rows, deterministic readback rows, continuation routes, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a source query through S, R, E, U, D, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{UniqueChoicePrinciple}^{\uparrow}$, BHist, hsame, *classifier data,* Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniqueChoicePrinciple}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite unique-relation readback packet. It has no coordinate for countable choice, selected representatives, host function equality, quotient completion, real trichotomy, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.UniqueChoicePrincipleUp.UniqueChoicePrincipleTasteGate single carrier alignment`

Theorem 1739.2 (UniqueChoicePrinciple NameCert obligations)**.** *For every accepted* $\mathsf{UniqueChoicePrinciple}^{\uparrow}$ *carrier $Q = (S, R, E, U, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniqueChoicePrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source row S, relation row R, existence witness row E, uniqueness ledger U, deterministic readback row D, transport, continuation, provenance, and local naming rows;*

2. *existence visibility, requiring each accepted source query to display an E row before any deterministic readback row is consumed;*

3. *uniqueness-ledger determinacy, requiring two relation outputs over the same accepted source to be compared only through U by* hsame *transport and classifier agreement;*

4. *readback stability under* hsame *transport, requiring the deterministic row D to transport componentwise with S, R, E, U, H, C, P, N;*

5. *ledger non-escape, requiring every deterministic readback to factor through the displayed existence and uniqueness rows and forbidding countable choice, selected representatives, host function equality, quotient completion, real trichotomy, and ambient completeness from the local surface.*

No unique-choice obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.UniqueChoicePrincipleUp.UniqueChoicePrincipleCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1739.1. The packet displays the source, relation, existence witness, uniqueness ledger, deterministic readback, transport, continuation, provenance, and local naming rows before any readback can be consumed.

Existence visibility follows from the position of E in the carrier route. A source query can reach D only through the replay row C, and that replay passes through the relation row and the existence witness row first.

Uniqueness-ledger determinacy is the role of U. If two relation outputs are displayed for the same accepted source row, the packet compares them by the hsame and classifier data recorded in U, not by host equality or a hidden selector.

Readback stability is componentwise. The transport ledger H moves the source, relation, existence, uniqueness, deterministic readback, replay, provenance, and naming rows together, so the deterministic row has no independent equality principle.

Finally inspect the carrier inventory. Since no coordinate of Q is a countable-choice schedule, selected representative, host function equality proof, quotient-completion object, trichotomy decision, or ambient-completeness theorem, none can occur in the local NameCert surface. □

Theorem 1739.3 (UniqueChoicePrinciple deterministic readback)**.** *Let $Q = (S, R, E, U, D, H, C, P, N)$ be an accepted* UniqueChoicePrinciple$^{\uparrow}$ *carrier. Any two deterministic readbacks over the same accepted source row and relation row are compared only through the displayed uniqueness ledger U and the componentwise* hsame *transport recorded by H.*

Thus deterministic readback is a finite relation-ledger boundary. It is not countable choice, a selected representative, host function equality, quotient completion, real trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniqueChoicePrincipleUp.UniqueChoicePrincipleCarrier deterministic readback`

Proof. Start with two accepted deterministic readbacks over the same source row S and relation row R. By Definition 1739.1, each readback must be the row D reached through displayed existence evidence E and replay C inside the same packet.

The NameCert obligations of Definition 1739.2 force comparison of the corresponding relation outputs through the uniqueness ledger U. That ledger exposes only hsame transport and classifier agreement for the displayed rows.

The transport ledger H then carries this comparison componentwise across S, R, E, U, D, C, P, N. There is no separate host function equality or representative selector available to the readback.

The excluded principles have no coordinate in Q and no replay edge in C. Therefore any accepted comparison of deterministic readbacks is exactly the displayed uniqueness-ledger comparison transported by H. □

Closure status for NameCert$_{\text{UniqueChoicePrinciple}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{UniqueChoicePrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 1739.1, Definition 1739.2, and Definition 1739.3 seed the finite unique-relation readback surface through source, relation, existence, uniqueness-ledger, deterministic readback, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{UniqueChoicePrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close countable choice, selected representatives, host function equality, quotient completion, real trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, existence visibility, uniqueness-ledger determinacy, readback stability under hsame transport, and ledger non-escape for BEDC.Derived.UniqueChoicePrincipleUp.

Constructive story: A UniqueChoicePrinciple$^{\uparrow}$ packet is generated from one source carrier row, one relation row, one existence witness row, one uniqueness ledger, one deterministic readback row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1740

A Concrete Naming Certificate for $\mathsf{RegularCauchyInterpolation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyInterpolation}^{\uparrow}$ packet records finite interpolation and reindexing of regular Cauchy rational names before a completed real-name consumer is allowed to read the interpolated row. It sits over the dyadic mesh discipline of chapter 291, the finite stream-window surface of chapter 126, the regular rational source rows of chapter 301, the terminal real-name seal of chapter 38, and the paired-row handoff discipline of chapter 414. The packet gives Real and completion consumers a finite BEDC route for interpolation and schedule change; it is not host interpolation, quotient-stream equality, a selected real, a host limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1740.1 (Regular Cauchy interpolation carrier)**.** *A* $\mathsf{RegularCauchyInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, D, M, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational observations are to be interpolated or reindexed.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row containing the dyadic interpolation nodes, endpoint tolerances, and local mesh-width bounds used by the displayed route.*
3. *M is the finite interpolation ledger, recording for each requested output precision the source indices, dyadic weights, endpoint observations, and tolerance split consumed by that request.*
4. *W is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window family exposing exactly the finite source observations read by M.*
5. *S is the regular readback schedule whose output row is obtained by replaying the finite interpolation ledger against R, D, M, W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source row, dyadic mesh row, interpolation ledger, window family, and readback schedule have been displayed.*
7. *H records componentwise* hsame *transport for the source row, mesh row, ledger, windows, readback schedule, Real seal, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from every interpolation-facing precision request through $R, D, M, W, S, E, H, P,$*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyInterpolation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyInterpolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, M, W, S, E, H, C, P, N$. The carrier has no coordinate for host interpolation algorithms, quotient-stream equality, selected reals, host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyInterpolationUp.RegularCauchyInterpolationTasteGate single carrier alignment`

Theorem 1740.2 (Regular Cauchy interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyInterpolation}^{\uparrow}$ *carrier*

$$I = (R, D, M, W, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyInterpolation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, finite interpolation ledger, displayed* $\mathsf{StreamName}^{\uparrow}$ *windows, regular readback schedule,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural rows;*
2. *dyadic mesh exposure, requiring every interpolation node, endpoint tolerance, and mesh-width bound used by M to be read from D before an interpolated readback is available;*
3. *finite window exposure, requiring W to display exactly the source observations requested by the ledger and no completed stream equality;*
4. *regular readback accounting, requiring S to consume only the displayed rows R, D, M, W and the finite tolerance split stored in the ledger;*
5. *Real seal nonescape, requiring E to be exposed only after the source, mesh, ledger, window, and readback rows have been displayed;*
6. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite interpolation packet;*
7. *refusal of host interpolation theorems, quotient-stream equality, selected reals, host limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No interpolation obligation is stored outside these finite rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyInterpolationUp.RegularCauchyInterpolationNameCertObligations`

Proof. Project the accepted carrier. The analytic rows are the source R, the dyadic mesh D, the finite interpolation ledger M, the displayed windows W, the readback schedule S, and the Real seal E.

Read the mesh and windows before forming any interpolated observation. The row D supplies the finite dyadic nodes, endpoint tolerances, and mesh-width bounds, while W exposes exactly the source observations named by the corresponding ledger entry of M.

Build the readback by replaying the ledger. Each requested output precision uses only the displayed source observations, dyadic weights, and tolerance split already present in R, D, M, W, so S is a regular readback schedule over finite data rather than an appeal to an ambient interpolation theorem.

Finally transport and package the packet through H, C, P, N. Since no carrier coordinate stores quotient-stream equality, host interpolation, selected reals, host limits, ambient completeness, Lean verification, or bridge data, none of those objects can become a local NameCert obligation. □

Theorem 1740.3 (Regular Cauchy interpolation dyadic-window handoff)**.** *Every readback exported by an accepted* $\mathsf{RegularCauchyInterpolation}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto D \longmapsto M \longmapsto W \longmapsto S$$

before it may be consumed as a regular Cauchy output row. Thus the consumer receives a regular rational source row, a dyadic mesh row, a finite interpolation ledger, displayed stream windows, and a regular readback schedule; it receives no quotient-stream equality, host interpolation theorem, selected real, host limit, or ambient completeness coordinate.

Proof. Begin with the accepted carrier and an interpolation-facing precision request. The request first projects the source row R and the mesh row D, so every endpoint and tolerance used downstream is already displayed.

The ledger row M then selects a finite set of source observations and dyadic weights. The window family W exposes precisely those observations, and componentwise replay through C preserves the same finite dependency order.

The row S is therefore downstream of the displayed mesh and windows. Because the carrier offers no coordinate for quotient-stream equality, host interpolation, selected reals, host limits, or ambient completeness, the handoff to the regular readback cannot escape this finite route. □

Theorem 1740.4 (Regular Cauchy interpolation Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RegularCauchyInterpolation}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto D \longmapsto M \longmapsto W \longmapsto S \longmapsto E$$

before it may be consumed as a completed real-name boundary. Thus the Real-facing consumer receives the source row, dyadic mesh row, interpolation ledger, displayed windows, regular readback schedule, and seal row; it receives no host interpolation theorem, quotient-stream equality, selected real, host limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the terminal row E of the accepted packet. By carrier admission, E is not independent data; it is downstream of the displayed source row, dyadic mesh row, interpolation ledger, stream windows, and regular readback schedule.

The only route into E passes through S. The row S is itself read from R, D, M, W, so the Real-facing boundary inherits the same finite interpolation route and the same componentwise transport recorded by H.

No additional coordinate remains to supply host interpolation, quotient-stream equality, selected reals, host limits, or ambient completeness. The $\mathsf{Real}^{\uparrow}$ consumer therefore sees only the finite BEDC seal produced by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyInterpolation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1740.1, Definition 1740.2, Definition 1740.3, and Definition 1740.4 seed the finite interpolation route from a $\mathsf{RegSeqRat}^{\uparrow}$ source through $\mathsf{DyadicRatCore}^{\uparrow}$ mesh data, finite interpolation ledger, displayed $\mathsf{StreamName}^{\uparrow}$ windows, regular readback schedule, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, host interpolation theorems, quotient-stream equality, selected reals, host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exposure, window handoff, regular readback accounting, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyInterpolationUp.

Constructive story: A RegularCauchyInterpolation$^{\uparrow}$ packet is generated from a RegSeqRat$^{\uparrow}$ source row, a DyadicRatCore$^{\uparrow}$ mesh row, a finite interpolation ledger, displayed StreamName$^{\uparrow}$ windows, a regular readback schedule, a Real$^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1741

A Concrete Naming Certificate for $\mathsf{UniformCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformCauchyModulus}^{\uparrow}$ packet records the finite shared Cauchy-window budget needed by the Real and completion route before any quotient stream, selected limit, or ambient completeness theorem is available. It sits between the selected-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the endpoint seal of chapter 38. It also reads the shared-bound interface of chapter 462 and the filter-side uniformity route of chapter 1694.

Definition 1741.1 (Uniform Cauchy modulus carrier)**.** *A* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, D, M, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row whose finite stream observations are visible to the packet;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected windows in S;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording every requested rational radius and comparison budget;*
4. *M is the shared Cauchy-window modulus row: one displayed budget assigns a finite tail window to each requested dyadic tolerance;*
5. *W is the finite witness row saying that all accepted pairs of selected windows beyond the budget in M are read inside the tolerance recorded in D;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after S, R, D, M, W have been displayed;*
7. *H stores componentwise* hsame *transport for the stream windows, rational readbacks, dyadic ledgers, shared modulus, witness row, handoff, provenance, and local naming rows;*
8. *C stores displayed* Cont *replay from selected windows to regular readback, from regular readback to dyadic tolerance accounting, from the tolerance ledger to the shared modulus, and from the witness row to the* $\mathsf{Real}^{\uparrow}$ *handoff;*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{UniformCauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, W, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, open-cover compactness, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyModulusUp.UniformCauchyModulusUpTasteGate single carrier alignment`

Theorem 1741.2 (Uniform Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier $U = (S, R, D, M, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed selected-window, regular-readback, dyadic-ledger, shared-modulus, witness, handoff, transport, replay, provenance, and naming rows;*
2. *classifier transport, requiring accepted comparisons to preserve $S, R, D, M, W, E, H, C, P, N$ componentwise and to preserve no hidden completion datum;*
3. *shared-window budget exactness, requiring the single row M to govern every displayed Cauchy comparison requested by D;*
4. *finite witness replay, requiring each accepted comparison beyond the budget in M to pass through S, R, D, and W before any endpoint-facing handoff is read;*
5. $\mathsf{Real}^{\uparrow}$ *handoff nonescape, requiring E to consume only the displayed finite route and never quotient stream equality, selected limits, countable choice, open-cover compactness, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyModulusUp.UniformCauchyModulusTasteGate e0 row round trip`

Proof. Carrier admission is projection from Definition 1741.1. The displayed packet has exactly the rows $S, R, D, M, W, E, H, C, P, N$, so the local certificate has no detached limit selector, quotient stream coordinate, or ambient completeness witness.

Classifier transport is componentwise. The row H stores the permitted hsame comparisons, while C replays only the displayed route from selected windows through regular readback, dyadic budgets, the shared modulus row, the finite witness row, and the endpoint handoff.

The shared-window budget is unique inside the packet. A tolerance request may ask for many finite window comparisons, but those comparisons all read the same displayed modulus row M before the witness row W can be exported.

Finally, the $\mathsf{Real}^{\uparrow}$ handoff row E is downstream of the same finite route. Since none of the excluded principles is a coordinate of the carrier, none can be exported by the local $\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}$ surface. □

Theorem 1741.3 (Uniform Cauchy modulus Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier may read only the displayed selected-window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the shared modulus row M, the finite witness row W, and the structural rows H, C, P, N before reading the handoff row E. Such a consumer cannot read quotient stream equality, selected limits, countable choice, open-cover compactness, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Project the accepted carrier. The endpoint-facing row is E, but its admissible inputs are already fixed by the preceding rows: selected windows in S, rational readback in R, tolerance accounting in D, the shared budget in M, and finite comparison witnesses in W.

Transport, replay, provenance, and local naming do not add an endpoint object. They only preserve the same route through H, C, P, N.

Thus every $\mathsf{Real}^{\uparrow}$ or completion-facing read must pass through the finite carrier surface before reaching E. Since the carrier has no coordinate for the excluded principles, no such principle can be extracted from this handoff. □

Theorem 1741.4 (UniformCauchyModulus shared-window budget)**.** *For every accepted* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier* $U = (S, R, D, M, W, E, H, C, P, N)$*, a finite precision request must first read the selected* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the shared modulus row* M*, and the finite witness row* W*. If two endpoint-facing consumers share* M *and* W*, then their* $\mathsf{Real}^{\uparrow}$ *handoff row* E *is reached through the same displayed budget route*

$$S \longrightarrow R \longrightarrow D \longrightarrow M \longrightarrow W \longrightarrow E$$

as replayed by H *and* C*. No quotient stream, selected limit, countable choice, ambient completeness row, host real equality, open-cover compactness, or trichotomy coordinate can enter this budget.*

Lean4 ✓ `BEDC.Derived.UniformCauchyModulusUp.UniformCauchyModulusSharedWindowBudget`

Proof. Project the accepted carrier. The only rows available before the endpoint handoff are the selected-window row S, regular readback row R, dyadic tolerance row D, shared modulus row M, finite witness row W, transport row H, replay row C, provenance row P, and local naming row N.

Induct over the displayed witness row W. The empty witness route exposes no endpoint-facing read, so no consumer can reach E without the selected windows, readbacks, tolerance ledger, and shared modulus already displayed. In the step case, the next witness cell is read only after its selected windows have been transported through H and replayed through C along the route from S to R, from R to D, and from D to M.

Now take two endpoint-facing consumers that share M and W. The induction gives the same finite witness route for both consumers, and the carrier has only one handoff row E downstream of that route. Hence both consumers reach E through the displayed shared-window budget.

The excluded principles are not coordinates of U and are not produced by H or C. Therefore the finite budget cannot import quotient stream equality, a selected limit, countable choice, ambient completeness, host real equality, open-cover compactness, or trichotomy. □

Theorem 1741.5 (Uniform Cauchy modulus single-route exhaustion)**.** *For every accepted* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier* $U = (S, R, D, M, W, E, H, C, P, N)$*, any two* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumers that use the same precision request and the displayed witness row* W *factor through the same ordered route*

$$S \longrightarrow R \longrightarrow D \longrightarrow M \longrightarrow W \longrightarrow E.$$

The route is exhausted by the displayed rows of U*: no second modulus row, detached tail selector, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness can be inserted between the precision request and the endpoint handoff.*

Proof. Project the accepted carrier. By Definition 1741.1, the endpoint-facing order is already displayed as selected windows S, regular rational readbacks R, dyadic tolerance accounting D, one shared Cauchy-window modulus row M, finite witness replay W, and handoff row E.

The single-row obligation is fixed by Definition 1741.2: accepted comparisons must preserve the same row M for every displayed Cauchy comparison requested by D. Thus two consumers with the same precision request cannot split into distinct modulus rows or detached tail selectors before the witness row is read.

Apply Definition 1741.4 to the common witness row W. Its witness induction gives the same finite route through S, R, D, M, W for both consumers, with transport and replay supplied only by H and C.

Finally, Definition 1741.3 gives the nonescape boundary at E. Since the excluded principles are not coordinates of the carrier and are not produced by H or C, neither consumer can insert quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness into the route. □

Theorem 1741.6 (Uniform Cauchy modulus scoped kernel binding)**.** *For every accepted* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *carrier* $U = (S, R, D, M, W, E, H, C, P, N)$*, the selected windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, shared modulus row, witness replay, and* $\mathsf{Real}^{\uparrow}$ *handoff bind only through the displayed* BHist*,* hsame*,* Cont*,* Pkg*,* $\mathsf{NameCert}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. No scoped kernel consumer of* U *can insert a second route between the finite precision request and the endpoint handoff.*

Proof. Start from the carrier definition. The rows $S, R, D, M, W, E, H, C, P, N$ are the only displayed coordinates, and the carrier contains no detached tail selector or hidden completion witness.

The NameCert obligations bind those coordinates componentwise. Carrier admission fixes the selected windows, regular readbacks, tolerance ledger, shared modulus, witness row, handoff, transport, replay, provenance, and local naming rows.

Apply the shared-window budget theorem to the middle of the route. It forces every precision request to pass through the same S, R, D, M, W budget before the endpoint row is visible.

Finally apply single-route exhaustion. The route from finite precision request to E is already exhausted by the displayed rows, so the scoped kernel surface is exactly the listed BHist, hsame, Cont, Pkg, $\mathsf{NameCert}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ binding surface. □

Theorem 1741.7 (Uniform Cauchy modulus compact-tail router)**.** *Any* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer reading a* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *endpoint must route through selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the shared modulus row* M*, finite witness replay* W*, and the* $\mathsf{Real}^{\uparrow}$ *handoff* E*. The route excludes selected subsequences, open-cover compactness, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1741.1. The endpoint-facing data are the selected windows, regular readbacks, dyadic tolerance ledger, shared modulus row, finite witness row, and handoff row.

Apply the shared-window budget theorem, Definition 1741.4. It forces every precision request to read the same route through S, R, D, M, W, E before a consumer receives endpoint data.

Use Definition 1741.5 and Definition 1741.6. They prevent a compact, complete, or continuous-map consumer from inserting a second route outside the displayed kernel rows.

Thus the compact-tail handoff remains a finite BEDC route. Since the carrier has no selected subsequence, open-cover compactness, quotient stream equality, countable-choice schedule, or ambient completeness coordinate, none can be exported. □

Falsifiable prediction: Every $\mathsf{UniformCauchyModulus}^{\uparrow}$ endpoint-facing read factors through selected StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, one shared Cauchy-window modulus row, finite witness replay, a RealUp handoff row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchymodulusmeet, uniformcauchyfilter, real: $\mathsf{UniformCauchyModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries selected windows, regular rational readbacks, dyadic tolerance accounting, one shared finite Cauchy-window budget, witness replay, endpoint handoff, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyModulus}^{\uparrow}$ packet is generated from selected StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, one shared finite Cauchy-window modulus row, finite witness replay, a RealUp handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1741.1, Definition 1741.2, Definition 1741.3, Definition 1741.4, Definition 1741.5, and Definition 1741.6 close the finite shared Cauchy-window budget and scoped kernel binding for Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformCauchyModulusUp.UniformCauchyModulusTasteGate\_e0\_row\_round\_trip`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public-object surface and bridge alignment for BEDC.Derived.UniformCauchyModulusUp.

Chapter 1742

A Concrete Naming Certificate for $\mathsf{RegularCauchySquare}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySquare}^{\uparrow}$ packet records diagonal multiplication of a displayed regular rational Cauchy real name by itself as a finite bounded-window route. It sits over the regular rational source rows of chapter 301, the synchronized window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the binary product carrier of chapter 461, the product-budget ledger of chapter 676, and the terminal real-name seal of chapter 38. The packet is the diagonal square surface consumed by later real algebra routes; it is not host real multiplication, quotient stream equality, a selected limit, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1742.1 (Regular Cauchy square carrier)**.** *A* $\mathsf{RegularCauchySquare}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W_L, W_R, D_L, D_R, M, E, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *A is the single* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose diagonal square is being read.*
2. *W_L and W_R are duplicated aligned* $\mathsf{StreamName}^{\uparrow}$ *window rows opened from the same source row A at the same product-facing precision request and observation index.*
3. *D_L and D_R are identical* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from the duplicated windows; their equality is displayed as a diagonal source-lock row rather than inferred from quotient stream equality.*
4. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *diagonal product row obtained by feeding the two product source slots with the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row A and the two aligned endpoint rows D_L, D_R.*
5. *E is the* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *ledger row that records the paired source-window radii, dyadic square row, product rounding allowance, and remaining readback budget for the diagonal route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from M and E.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source row, duplicated windows, dyadic endpoint rows, diagonal product row, product-budget ledger, and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for the source row, duplicated windows, dyadic endpoints, diagonal product row, product-budget ledger, readback, seal, continuation routes, package provenance, and local naming.*

9. *C records displayed* Cont *replay from any square-facing precision request through* $A, W_L, W_R, D_L, D_R, M, E, R, Z$
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySquare}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyProductBudget}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySquare}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W_L, W_R, D_L, D_R, M, E, R, Z, H, C, P, N$. The carrier has no coordinate for host multiplication, quotient stream equality, selected limits, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySquareUp.RegularCauchySquareTasteGate single carrier alignment`

Theorem 1742.2 (Regular Cauchy square NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySquare}^{\uparrow}$ *carrier*

$$S = (A, W_L, W_R, D_L, D_R, M, E, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySquare}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, duplicated aligned* $\mathsf{StreamName}^{\uparrow}$ *windows, identical* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, diagonal product row, product-budget ledger, regular readback,* $\mathsf{Real}^{\uparrow}$ *seal row, and structural rows;*
2. *diagonal source-lock, requiring both product source slots to read the same source row A before any square endpoint is available;*
3. *duplicated-window admission, requiring W_L and W_R to expose the same finite source window at the same product-facing precision request and observation index;*
4. *dyadic square row, requiring D_L and D_R to be identical endpoint rows read from the duplicated windows before the dyadic product observation is formed;*
5. *product-budget exactness, requiring E to be the finite* $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ *ledger for the diagonal product row M and the source-window radii already displayed;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the readback row R to consume only the diagonal product row and the displayed product-budget ledger;*
7. $\mathsf{Real}^{\uparrow}$ *seal boundary, requiring every real-facing read to pass through R and the visible square provenance before the seal row Z is exposed;*
8. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite square packet;*
9. *non-escape from host multiplication, quotient stream equality, selected limits, ambient completeness,* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert data.*

No diagonal square obligation is stored outside these finite rows.

Theorem 1742.3 (Regular Cauchy square product handoff)**.** *Every accepted* $\mathsf{RegularCauchySquare}^{\uparrow}$ *packet factors through the existing* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route with both product source rows identified as the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row. More precisely, for*

$$S = (A, W_L, W_R, D_L, D_R, M, E, R, Z, H, C, P, N),$$

the product-facing component is the diagonal specialization of the product packet whose source rows are A, A, whose aligned windows are W_L, W_R, whose endpoint rows are D_L, D_R, whose product route is M, and whose product-budget handoff is E. The readback R and seal Z therefore consume the public product and budget boundary before any square-facing consumer can read the result.

Proof. Project the carrier rows of Definition 1742.1. The single source row A is copied into the two product source slots, and the duplicated windows W_L, W_R expose the same finite observation boundary before the dyadic endpoint rows D_L, D_R are available.

Apply the public product route of Definition 461.9. Its product carrier accepts two regular source rows, aligned windows, dyadic endpoint rows, a finite product ledger, a regular readback, transports, continuations, provenance, and local naming. In the square packet these product source rows are the displayed pair A, A, so the product component is the diagonal specialization of the same route.

Next apply the product-budget handoff rows of Definition 676.10 and Definition 676.11. They force the readback and real-facing seal to consume the finite product-budget ledger before any downstream consumer receives the result.

The remaining rows H, C, P, N are structural rows of the same finite packet. Since the carrier contains no coordinate for host multiplication, quotient stream equality, selected limits, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, the factorization cannot escape the displayed $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ routes. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySquare}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySquare}^{\uparrow}$ packet is generated from one $\mathsf{RegSeqRat}^{\uparrow}$ source row, duplicated aligned $\mathsf{StreamName}^{\uparrow}$ windows, identical $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, a diagonal $\mathsf{RegularCauchyProduct}^{\uparrow}$ row, a $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySquare}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1742.1, Definition 1742.2, and Definition 1742.3 seed the finite diagonal square route through one $\mathsf{RegSeqRat}^{\uparrow}$ source row, duplicated aligned $\mathsf{StreamName}^{\uparrow}$ windows, identical $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, the existing $\mathsf{RegularCauchyProduct}^{\uparrow}$ route, the $\mathsf{RegularCauchyProductBudget}^{\uparrow}$ ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySquare}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchySquareUp.RegularCauchySquareTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host multiplication, quotient stream equality, selected limits, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal source-lock, product-budget exactness, RegSeqRat handoff, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySquareUp.

Chapter 1743

A Concrete Naming Certificate for $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$

Following Kreisel, Lacombe, and Shoenfield on effective continuity, and Bishop–Bridges on constructive real names, the $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ packet records the finite-prefix route by which a computable functional over choice-sequence data can hand a real-name consumer only inspected prefix evidence. It sits between the Baire-prefix surface of chapter 334, the observation discipline of chapter 1482, the finite-window carrier of chapter 126, the regular rational readback of chapter 301, and the endpoint seal of chapter 38. The chapter names the finite certificate boundary for prefix continuity; it does not complete a choice sequence, select a real limit, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1743.1 (Kreisel-Lacombe-Shoenfield carrier)**.** *A* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, S, R, E, M, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-context row naming the finite input prefix currently inspected by the functional;*
2. *O is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *observation row recording the choice-sequence queries whose answers are visible in that prefix context;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the output name is requested;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the output windows in S;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the prefix context, observations, window schedule, and regular readback have been displayed;*
6. *M is the finite modulus row assigning to each displayed output-window request the input-prefix length sufficient for replaying the read;*
7. *L is the finite continuity ledger recording every accepted pair of matching prefixes, their matching observations, and the common output-window readback justified by M;*
8. *H records componentwise* hsame *transport for the Baire prefix context, choice-sequence observations, stream-window schedule, regular readback, Real seal, finite modulus, continuity ledger, replay, provenance, and local naming rows;*

9. *C records displayed* Cont *replay from Baire prefixes to choice-sequence observations, from observations to the output window schedule, from the window schedule to regular rational readback, from the finite modulus to the ledger, and from the ledger to the* $\mathsf{Real}^{\uparrow}$ *seal;*
10. *P is the* Pkg *provenance over* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, S, R, E, M, L, H, C, P, N$. It has no coordinate for countable choice, completed host functions, quotient stream equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KreiselLacombeShoenfieldUp.KreiselLacombeShoenfieldUpTasteGate single carrier alignment`

Theorem 1743.2 (Kreisel-Lacombe-Shoenfield NameCert obligations)**.** *For every accepted* KreiselLacombeShoenfie *carrier $K = (B, O, S, R, E, M, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Baire prefix context, choice-sequence observations, StreamName window schedule, RegSeqRat readback, Real seal, finite modulus, continuity ledger, transport, replay, provenance, and local naming rows;*
2. *prefix-observation exactness, requiring every query in O to be read inside the displayed prefix context B;*
3. *output-window exactness, requiring every scheduled output request in S to be read back through R before the Real seal row E is available;*
4. *finite modulus exactness, requiring M to assign the input-prefix length used by each output-window request and by no hidden total function;*
5. *continuity-ledger replay, requiring L to compare accepted matching prefixes only through the observations, window schedule, readback row, and modulus row already displayed;*
6. *nonescape for countable choice, completed host functions, quotient stream equality, selected limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. The only computational rows are the Baire prefix context B, the choice-sequence observation row O, the output window schedule S, the regular rational readback R, the Real seal E, the finite modulus row M, and the continuity ledger L.

Carrier admission and classifier transport are componentwise. The row H moves exactly the listed coordinates, while the replay row C repeats the same route from prefixes to observations, output windows, rational readback, modulus entries, continuity ledger entries, and the endpoint seal.

The finite modulus obligation is local to displayed requests. A consumer can ask for an output window, but the packet answers by opening the corresponding prefix budget in M and then checking the ledger row L against visible observations.

The excluded principles are absent from the carrier. Since no row names countable choice, a completed host function, quotient stream equality, a selected limit, trichotomy, or ambient Real completeness, none of them can become a local $\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}$ obligation. □

Theorem 1743.3 (Kreisel-Lacombe-Shoenfield prefix continuity)**.** *Every output-window read from an accepted* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier factors through a finite prefix route*

$$B, O \longmapsto M \longmapsto S, R \longmapsto L \longmapsto E, H, C, P, N.$$

Thus two accepted carriers that agree on the displayed prefix context and choice-sequence observations up to the modulus row M have the same scheduled output-window readback through S and R for the ledger entry in L.

Proof. Start with an output-window read. The schedule row S is not available as a detached total output function; it is paired with the displayed prefix context B and observation row O.

Open the modulus row M. It assigns the finite input-prefix length required by the requested output window, so matching prefixes and matching observations are the only data the ledger may compare.

Replay the ledger row L. Its accepted entries bind the prefix agreement, observation agreement, scheduled output window, and regular rational readback into one finite route.

The structural rows H, C, P, N preserve this route and add no hidden input or output object. Therefore agreement on the displayed finite prefix data up to M determines the scheduled readback for the ledger entry under consideration. □

Theorem 1743.4 (Kreisel-Lacombe-Shoenfield StreamName modulus)**.** *Every displayed* $\mathsf{StreamName}^{\uparrow}$ *output-window request answered by an accepted* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier first opens the finite modulus row M. The answer then reads only the matching Baire prefix context B, choice-sequence observations O, regular rational readback R, and continuity ledger L before the* $\mathsf{Real}^{\uparrow}$ *seal row E is exposed. The handoff exports no completed host function, countable choice, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows from Definition 1743.1. The output-window request is represented by the schedule row S, while the only row assigning an input-prefix budget to that request is M.

Apply prefix continuity from Definition 1743.3. Once M is opened, the request is replayed through the displayed Baire prefix context B and the choice-sequence observations O, and the corresponding output data are read through S, R, L.

The Real seal row is downstream of that finite route. A consumer may inspect E only after the modulus entry, matching prefix context, observations, regular readback, and ledger entry have all been displayed.

The excluded principles are not carrier coordinates. The structural rows H, C, P, N preserve the same windowed handoff and cannot add a completed host function, countable-choice selector, quotient stream equality, selected limit, or ambient Real completeness theorem. □

Theorem 1743.5 (Kreisel-Lacombe-Shoenfield modulus-ledger exactness)**.** *For every accepted* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier $K = (B, O, S, R, E, M, L, H, C, P, N)$, each entry of the continuity ledger L is justified by exactly one displayed finite modulus row M, Baire prefix row B, choice-observation row O, StreamName window row S, and RegSeqRat readback row R. Componentwise transport and replay through H and C can preserve this displayed justification, but they cannot add countable choice, quotient stream equality, a completed host function, selected Real limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows. The ledger L is not a detached list of continuity assertions; by carrier admission it is downstream of the displayed prefix context B, observation row O, output-window schedule S, regular rational readback R, and finite modulus row M.

Apply Definition 1743.3. Matching prefixes and observations determine the scheduled output readback only after the relevant modulus entry has been opened.

Apply Definition 1743.4. Every StreamName request answered by the packet first reads M and then replays the route through B, O, S, R, L before the Real seal E is exposed.

It remains to rule out hidden support for the ledger. This is exactly the nonescape surface of Definition 1743.7: the transport, replay, provenance, and naming rows preserve displayed carrier coordinates and have no coordinate for countable choice, quotient stream equality, completed host functions, selected limits, or ambient Real completeness. □

Theorem 1743.6 (Kreisel-Lacombe-Shoenfield Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier must consume the displayed Baire prefix context, choice-sequence observation row, StreamName window schedule, RegSeqRat readback, finite modulus, and continuity ledger before reading the Real seal row. The handoff supplies finite real-name evidence only; it supplies no countable choice, completed host function, quotient stream equality, selected limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the endpoint-facing row E. By carrier admission it is downstream of B, O, S, R, M, L, so a Real consumer cannot read E without the prefix, observation, schedule, readback, modulus, and ledger rows.

The readback row R is the only rational-name information supplied to the endpoint. It is scheduled by S and justified by the ledger entries whose prefix budgets are recorded in M.

Transport and replay do not add a completion theorem. The rows H and C preserve the displayed route, while P and N source and name it as the local packet.

Consequently the Real handoff exports finite real-name evidence. The absent principles have no coordinate in the packet and cannot be extracted from the handoff. □

Theorem 1743.7 (Kreisel-Lacombe-Shoenfield nonescape)**.** *No accepted* $\mathsf{KreiselLacombeShoenfield}^{\uparrow}$ *carrier exports countable choice, a completed host function on Baire space, quotient stream equality, a selected real limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every consumer-visible datum is one of the displayed rows* $B, O, S, R, E, M, L, H, C, P, N$ *or a componentwise replay of those rows.*

Proof. List the carrier rows from Definition 1743.1. They are finite prefix, observation, window, readback, seal, modulus, ledger, transport, replay, provenance, and naming rows.

Now inspect each consumer route. Prefix-continuity reads pass through B, O, M, S, R, L, while Real handoff reads pass through B, O, S, R, M, L, E before the structural rows preserve the packet.

There is no remaining export surface. The classifier compares packets only through the displayed rows, and the structural rows H, C, P, N cannot introduce a principle absent from their inputs.

Thus the listed principles are refused by construction. Any consumer needing one of them must obtain it outside the local $\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1743.1, Definition 1743.2, Definition 1743.3, Definition 1743.6, and Definition 1743.7 seed the finite-prefix continuity certificate for computable functionals from Baire and choice-sequence observations into real-name data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KreiselLacombeShoenfield}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, completed host functions, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-observation exactness, output-window exactness, finite modulus exactness, continuity-ledger replay, Real handoff, and nonescape for BEDC.Derived.KreiselLacombeShoenfieldUp.

Constructive story: A KreiselLacombeShoenfield$^{\uparrow}$ packet is generated from Baire prefix context, choice-sequence observation, StreamName finite-window schedule, RegSeqRat readback, Real seal, finite modulus, continuity ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1744

Cauchy Modulus Monotone Normalization NameCert Construction

Following Bishop and Bridges on constructive analysis with explicit moduli, the CauchyModulusMonotoneNormalizat packet records the finite certificate by which an inspected Cauchy-modulus row is replaced by a displayed monotone threshold schedule without changing the tail-window claims already exposed to Real and Completion consumers. It sits on the explicit source route of chapter 1705, the separation boundary of chapter 1650, the finite completion route of chapter 1691, and the shared-budget surface of chapter 1741. The packet is a normalization certificate for later diagonal and tail-window arguments; it is not Real completeness, quotient completion, quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or a standard bridge.

Definition 1744.1 (Cauchy modulus monotone normalization carrier)**.** *A* $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, M, D, T, U, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row: a displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *surface whose Cauchy-window comparisons are being normalized.*
2. *M is the input* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, assigning a finite tail budget to each requested dyadic tolerance in the source route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the finite precision requests, slack conversions, and comparison budgets consumed by M.*
4. *T is the finite prefix-maximum table. For each displayed tolerance request it records the largest threshold among the finitely many tighter requests visible in D.*
5. *U is the normalized monotone modulus row obtained by reading T as the replacement threshold schedule.*
6. *W is the tail-window witness row saying that every accepted pair of source observations beyond the threshold named by U is still admitted by the original row M at the requested tolerance.*
7. *E is the endpoint handoff row available to* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing consumers only after S, M, D, T, U, W have been displayed.*

8. *H records componentwise* hsame *transport for the source, input modulus, dyadic ledger, prefix-maximum table, normalized modulus, witness, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through D, M, T, U, W, and the endpoint handoff E.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, RegularCauchySeparation $\mathsf{FastCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, D, T, U, W, E, H, C, P, N$. It has no coordinate for Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyModulusMonotoneNormalizationUp.CauchyModulusMonotoneNormalizationUpTasteGate`
`single carrier alignment`

Theorem 1744.2 (Cauchy modulus monotone normalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$ *carrier $Z = (S, M, D, T, U, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source, input modulus, dyadic tolerance ledger, prefix-maximum table, normalized modulus, tail-window witness, endpoint handoff, transport, replay, provenance, and local naming rows;*

2. *prefix-maximum exactness, requiring T to be read only from the finite tolerance requests displayed in D and the input thresholds displayed in M;*

3. *monotone schedule exposure, requiring U to expose the normalized threshold for each displayed tolerance request through T and no hidden cofinal tail;*

4. *tail-window preservation, requiring W to show that every comparison admitted beyond U is already a comparison admitted by M at the requested tolerance;*

5. *consumer handoff nonescape, requiring E to be downstream of S, M, D, T, U, W and to export no Real completeness, quotient completion, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or bridge datum;*

6. *componentwise transport, replay, provenance, and local naming stability for exactly the displayed rows.*

No monotone-normalization obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1744.1. The packet exposes the source row, input modulus, dyadic ledger, prefix-maximum table, normalized modulus, tail-window witness, endpoint handoff, and structural rows as one finite carrier.

Prefix-maximum exactness is finite. For each displayed tolerance request in D, the table T reads only the finitely many tighter requests already visible to the packet and the corresponding thresholds named by M.

The normalized modulus row U is therefore monotone by construction of the table T. Moving to a tighter displayed tolerance can only enlarge the inspected finite prefix, so the threshold exposed by U does not move backward.

Tail-window preservation is the witness row W. A consumer using U first reads the table entry in T, then reads the input threshold row M for the corresponding finite requests, and only then obtains the same tail-window comparison.

Finally, E, H, C, P, N hand off, transport, replay, source, and name the same finite route. Since Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and bridge data are absent from the carrier, none can become a local NameCert obligation. □

Theorem 1744.3 (Cauchy modulus monotone normalization tail-window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or completion-facing read of an accepted* $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$ *carrier factors through*

$$S, M, D \longmapsto T \longmapsto U \longmapsto W \longmapsto E, H, C, P, N.$$

Thus later diagonal, separation, and tail-window consumers may read a monotone threshold schedule only through the displayed input modulus, dyadic tolerance ledger, prefix-maximum table, normalized row, and preservation witness. The route exports no Real completeness, quotient completion, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or standard bridge.

Proof. Start with a consumer read. The only normalized schedule coordinate is U, and Definition 1744.1 places U after the source, input modulus, dyadic ledger, and prefix-maximum table.

The consumer first reads S, M, D. This fixes the inspected source, the original Cauchy thresholds, and the finite tolerance requests whose prefix maximum is allowed to be formed.

The table T then supplies the displayed maximum for the relevant finite prefix, and U reads that table as the monotone schedule. The witness row W checks that using this threshold remains a valid tail-window read for the input modulus M.

Endpoint and structural rows do not add a completion object. They only pass the same normalized route to E and preserve it through H, C, P, N, so the excluded objects have no coordinate from which a consumer can read them. □

Falsifiable prediction: Every $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$ consumer read factors through a source row, an input CauchyModulus threshold row, a DyadicRatCore tolerance ledger, a finite prefix-maximum table, a normalized monotone modulus row, a tail-window preservation witness, an endpoint handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: effectivecauchysequence, regularcauchyseparation, fastcauchycompletion, uniformcauchymodulus: $\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the input modulus, dyadic tolerance ledger, prefix-maximum table, normalized monotone schedule, preservation witness, endpoint handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1744.1, Definition 1744.2, and Definition 1744.3 seed the finite normalization packet that turns displayed Cauchy-modulus threshold rows into monotone threshold schedules for later Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusMonotoneNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-maximum exactness, monotone schedule exposure, tail-window preservation, consumer handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusMonotoneNormalizationUp.

Constructive story: A CauchyModulusMonotoneNormalization$^{\uparrow}$ packet is generated from a displayed source row, an input CauchyModulus threshold row, a DyadicRatCore tolerance ledger, a finite prefix-maximum table, a normalized monotone modulus row, a tail-window witness, an endpoint handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1745

A Concrete Naming Certificate for $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ packet names the finite boundary where cut-shaped observations may be compared with regular Cauchy readback without constructing $\mathsf{Real}^{\uparrow}$ by Dedekind cuts. It sits beside the lower and upper real ledgers, the located-cut handoff of chapter 404, the comparison surface of chapter 1424, the stream and regular-rational readback route, the dyadic tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet supplies a boundary for Phase Real and Completion consumers: it allows cut evidence to be inspected, routed, and refused as finite BEDC data while keeping the regular Cauchy route as the only displayed Real-facing readback.

Definition 1745.1 (Dedekind cut Cauchy boundary carrier)**.** *A* $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, K, Q, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *row. It records exactly the lower cut observations admitted by the boundary and their finite downward-use discipline.*
2. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *row. It records exactly the upper cut observations admitted by the boundary and their finite upward-use discipline.*
3. *K is the* $\mathsf{LocatedCut}^{\uparrow}$ *row. It records the located separation and displayed gap witnesses tying the lower and upper observations together.*
4. *Q is the* $\mathsf{DedekindCauchyComparison}^{\uparrow}$ *row. It is the only comparison row through which cut-shaped observations may approach a Cauchy-facing read.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule. It names the inspected stream windows used by the regular readback branch.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational observations exposed after the finite windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the dyadic radii, comparison margins, and finite slack rows consumed by the comparison and readback branches.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal. It is visible only after the located-cut row, comparison row, stream windows, regular-rational readback, and dyadic tolerance ledger have all been displayed.*
9. *H records componentwise* hsame *transport for the lower row, upper row, located-cut row, comparison row, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a cut-shaped request through $L, U, K, Q, S, R, D, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutCauchyBoundary}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite cut-to-Cauchy boundary packet. It has no coordinate for quotient cuts, selected cuts, host suprema, a Dedekind construction of $\mathsf{Real}^{\uparrow}$*, ambient completeness, trichotomy, Classical choice, Quot.sound, or propext-dependent equality.*

Lean4 (definition) `BEDC.Derived.DedekindCutCauchyBoundaryUp.DedekindCutCauchyBoundaryUp`
Lean4 ✓ `BEDC.Derived.DedekindCutCauchyBoundaryUp.TasteGate.DedekindCutCauchyBoundaryUpTasteGate`
`single carrier alignment`
Lean4 ✓ `BEDC.Derived.DedekindCutCauchyBoundaryUp.TasteGate.DedekindCutCauchyBoundaryTasteGate`
`single carrier alignment`

Theorem 1745.2 (Dedekind cut Cauchy boundary NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ *carrier $B = (L, U, K, Q, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutCauchyBo}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed lower row, upper row, located-cut row, comparison row, stream schedule, regular readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming row;*
2. *cut-side discipline, requiring every lower-row read in L and every upper-row read in U to meet only through the located separation and gap witnesses displayed in K;*
3. *Cauchy-readback factorization, requiring every cut-shaped observation exposed by L, U, K to pass through Q and then through S, R, D before the* $\mathsf{Real}^{\uparrow}$ *seal E is visible;*
4. *Real seal non-escape, requiring E to remain a terminal seal for the regular Cauchy route and never to serve as a quotient cut, selected cut, host supremum, Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, ambient completeness, trichotomy, Classical choice, Quot.sound, or propext-dependent surface;*
5. *transport and replay stability, requiring H and C to preserve the displayed route $L, U, K, Q, S, R, D, E, P, N$ componentwise;*
6. *ledger exhaustion, requiring P and N to name exactly the displayed finite packet and no object outside it.*

No Dedekind cut Cauchy boundary obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1745.1. The accepted packet exposes only the lower row, upper row, located-cut row, comparison row, stream schedule, regular readback, dyadic ledger, Real seal, and structural rows H, C, P, N.

Cut-side discipline is finite. A lower request reads only L, an upper request reads only U, and the shared located evidence is the row K; there is no ambient cut object in which the two sides can be merged.

Cauchy-readback factorization is forced by the row order. The comparison row Q receives the cut-shaped observations, the rows S, R, D provide the regular stream, rational, and dyadic readback data, and only then is the $\mathsf{Real}^{\uparrow}$ seal E available.

Transport and replay are structural. Componentwise hsame in H preserves each displayed coordinate, and Cont replay in C repeats the same route for every accepted cut-shaped request.

Finally, provenance and local naming are exhausted by P and N. Since none of the excluded surfaces is a coordinate of B, the local NameCert surface cannot export such a surface. □

Theorem 1745.3 (Dedekind cut Cauchy boundary nonescape)**.** *Let $B = (L, U, K, Q, S, R, D, E, H, C, P, N)$ be an accepted $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ carrier. Every Real/Completion consumer of B receives only the displayed lower-row evidence, upper-row evidence, located-cut witnesses, comparison row, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming row. It receives no quotient cut, selected cut, host supremum, Dedekind construction of $\mathsf{Real}^{\uparrow}$, ambient completeness, trichotomy, Classical choice, Quot.sound, or propext-dependent surface.*

Proof. Start with an accepted consumer read. The carrier presents the cut side only as L, U, K: lower observations, upper observations, and located witnesses. Because no quotient cut or selected cut is a coordinate, the consumer cannot recover such an object before entering the comparison row.

The comparison row is Q. It sends the cut-shaped evidence into the regular readback branch S, R, D, where finite stream windows, regular rational observations, and dyadic tolerance data are displayed before the Real seal is visible.

The Real-facing coordinate is only E. Its position downstream of Q, S, R, D prevents it from becoming a host supremum, Dedekind construction of $\mathsf{Real}^{\uparrow}$, ambient completeness principle, or trichotomy principle.

The structural rows H, C, P, N transport, replay, source, and name the same packet. They add no Classical choice, no Quot.sound, and no propext-dependent equality surface, so the listed exclusions cannot escape through the local NameCert. □

Theorem 1745.4 (Dedekind cut Cauchy boundary located refusal)**.** *Let $B = (L, U, K, Q, S, R, D, E, H, C, P, N)$ be an accepted $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ carrier. Any Real/Completion consumer that asks B for a located cut-side equality, interval boundary, or gap conclusion must first read the located-cut row K, then the comparison row Q, then the stream, regular-rational, and dyadic rows S, R, D, and only then the $\mathsf{Real}^{\uparrow}$ seal E.*

Conversely, any route that omits one of K, Q, S, R, D is refused as a located or equality-facing route to E. Such a route may still read the displayed lower and upper rows, transport row, replay row, provenance row, and local naming row, but those rows only transport, replay, source, and name the same finite boundary packet.

Proof. Project the accepted carrier along Definition 1745.1. The only cut-side coordinates are L, U, K, and the NameCert obligations of Definition 1745.2 require lower-row and upper-row reads to meet through the located separation and gap witnesses displayed in K.

The same obligation theorem gives the Cauchy-readback factorization. A cut-shaped observation that faces the $\mathsf{Real}^{\uparrow}$ seal must pass from K to the comparison row Q, then through the stream schedule S, the regular rational readback row R, and the dyadic tolerance ledger D, before E is visible.

For the converse, suppose a consumer omits one of K, Q, S, R, D. Then the factorization just described is incomplete, so the consumer has no displayed BEDC route from a located cut-side

question to E as an equality-facing conclusion. Apply Definition 1745.3: the missing route cannot be replaced by a quotient cut, selected cut, host supremum, Dedekind construction of $\mathsf{Real}^{\uparrow}$, ambient completeness, trichotomy, Classical choice, Quot.sound, or propext-dependent surface.

It remains only to account for the structural rows. The row H transports the displayed coordinates componentwise, C replays the same finite route, P records its provenance, and N names the local certificate. None of these rows adds a separate located or equality-facing route to E, so the refusal is exactly the refusal of every route that bypasses the regular Cauchy branch. □

Theorem 1745.5 (Dedekind cut Cauchy boundary regular readback dominance)**.** *Let $B = (L, U, K, Q, S, R, D, E,$ be an accepted* $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ *carrier. Every Real/Completion consumer of B that faces equality, located boundary, interval gap, or* $\mathsf{Real}^{\uparrow}$ *seal data must reach E through the finite route*

$$K, Q, S, R, D, E.$$

The lower and upper cut rows L, U remain visible cut observations, but they do not construct $\mathsf{Real}^{\uparrow}$ *by Dedekind cuts and they do not dominate equality-facing reads. The regular Cauchy branch through located evidence, comparison, stream windows, regular rational readback, and dyadic tolerance data dominates every equality-facing read of the boundary packet.*

Proof. Project the carrier of Definition 1745.1. The lower and upper rows L, U are observation rows, while K, Q, S, R, D, E are the displayed route from located cut evidence to a Real-facing seal.

Apply the NameCert obligations of Definition 1745.2. They require cut-side reads to meet through K and require every Cauchy-facing read to pass from K through Q, S, R, D before the seal E is available.

Use the nonescape theorem Definition 1745.3. Since quotient cuts, selected cuts, host suprema, and a Dedekind construction of $\mathsf{Real}^{\uparrow}$ are absent, no consumer can replace the regular branch by an unlisted cut construction.

Finally apply the located-refusal theorem Definition 1745.4. Any route omitting one of K, Q, S, R, D is refused as a located or equality-facing route to E, so the regular Cauchy branch is the dominant readback surface for every equality-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutCauchyBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutCauchyBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1745.1, Definition 1745.2, Definition 1745.3, and Definition 1745.4 seed the finite boundary comparing cut-shaped observations with regular Cauchy readback for Phase Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutCauchyBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient cuts, selected cuts, host suprema, Dedekind construction of $\mathsf{Real}^{\uparrow}$, ambient completeness, trichotomy, Classical choice, Quot.sound, propext-dependent equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-side discipline, Cauchy-readback factorization, Real seal non-escape, transport and replay stability, and ledger exhaustion for BEDC.Derived.DedekindCutCauchyBoundaryUp.

Constructive story: A $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ packet is generated from lower and upper real rows, a LocatedCut handoff, a DedekindCauchyComparison row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1746

A Concrete Naming Certificate for $\mathsf{CauchyTailDiameter}^{\uparrow}$

Following Bishop and Bridges regular Cauchy analysis, the $\mathsf{CauchyTailDiameter}^{\uparrow}$ packet records the finite witness that a displayed tail window has bounded diameter at a requested dyadic tolerance. It sits between the dyadic tolerance core of chapter 291, the stream-window source of chapter 126, the regular rational readback discipline of chapter 301, the terminal real-name seal of chapter 38, and the interval enclosure consumer of chapter 1625. The packet supplies the finite tail-diameter row later regular-sequence and real-completion arguments consume; it is not a completeness theorem, a quotient-real equality principle, or a point selector.

Definition 1746.1 (Cauchy tail diameter carrier)**.** *A* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, I, J, D, B, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the source stream or* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose tail is being inspected.*
2. *T is the finite tail threshold row, naming the displayed index after which the diameter claim is allowed to be read.*
3. *I and J are two visible tail-index rows, both certified to lie beyond the threshold T in the inspected window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row requested by the consumer.*
5. *B is the dyadic diameter-bound row, recording the displayed comparison that the two tail observations are within the bound attached to D.*
6. *W is the window-stability row asserting that every read of I, J, D, B uses the same finite* $\mathsf{StreamName}^{\uparrow}$ *window family and no hidden cofinal tail.*
7. *H records componentwise* hsame *transport for the source, threshold, two indices, tolerance, diameter bound, window-stability row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from the source row through the threshold, the two visible indices, the dyadic tolerance, the diameter bound, and the window-stability row.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailDiameter}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealIntervalEnclosure}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDiameter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, I, J, D, B, W, H, C, P, N$. It has no coordinate for a completed real limit, quotient real equality, trichotomy, point selection, open-cover compactness, or a standard bridge.

Theorem 1746.2 (Cauchy tail diameter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *carrier $K = (S, T, I, J, D, B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDiameter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source stream or* $\mathsf{RegSeqRat}^{\uparrow}$ *row, tail threshold, two visible tail indices, dyadic tolerance, diameter bound, window stability, transport, replay, provenance, and local naming rows;*
2. *threshold exactness, requiring both displayed indices I and J to be certified as tail indices beyond T before the diameter row is read;*
3. *dyadic bound exactness, requiring B to compare only the two observations exposed by I and J against the tolerance row D;*
4. *window stability, requiring all source and tail observations to use the same finite* $\mathsf{StreamName}^{\uparrow}$ *window family recorded by W;*
5. *componentwise transport and replay, requiring H and C to preserve the same source, threshold, indices, tolerance, bound, stability, provenance, and local naming rows;*
6. *refusal of completed real limits, quotient real equality, trichotomy, point selection, open-cover compactness, and standard-bridge data as local NameCert obligations.*

No Cauchy tail-diameter obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyTailDiameterUp.CauchyTailDiameterUpTasteGate single carrier alignment`

Proof. Project the accepted carrier. The only analytic rows are the source S, threshold T, visible indices I, J, tolerance D, diameter bound B, and window-stability row W.

Read the tail claim in dependency order. The threshold row is opened first, then the two indices are checked as visible tail indices in the same finite window family, and only then is the dyadic comparison row B available.

The structural rows preserve this route. Componentwise transport H can move only the listed coordinates, while the replay row C replays the same source-threshold-index-tolerance-bound-window sequence.

The provenance and local naming rows package the same finite data. Since the carrier contains no coordinate for a completed real limit, quotient real equality, trichotomy, point selection, open-cover compactness, or standard bridge, none of those objects can become a local obligation. □

Theorem 1746.3 (Cauchy tail diameter tail-window handoff)**.** *Every tail-diameter read from an accepted* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *carrier factors through the finite route*

$$S, T \longmapsto I, J \longmapsto D \longmapsto B \longmapsto W \longmapsto H, C, P, N.$$

Thus a consumer sees the source row, one threshold, two visible tail indices, a dyadic tolerance, a dyadic diameter bound, and the window-stability row before it can use the bound. The route exports no completed real limit, quotient real equality, trichotomy, point selection, open-cover compactness, or standard bridge.

Proof. Begin with a consumer read of the carrier. Carrier admission exposes $S, T, I, J, D, B, W, H, C, P, N$ as one finite packet, so the consumer cannot read a detached tail object.

Replay the continuation row C. The source row S supplies the finite observation family, T supplies the tail threshold, and I, J identify the two visible tail positions within that family.

The dyadic comparison is downstream. The tolerance row D fixes the requested radius, and the bound row B records the displayed diameter comparison for the two indexed observations.

Finally, W, H, P, N stabilize, transport, source, and name the same packet. No step in the route opens a completed real limit, quotient equality, trichotomy row, selected point, open-cover compactness witness, or standard bridge. □

Theorem 1746.4 (Cauchy tail diameter RegSeqRat consumption)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *consumer that uses a tail-diameter bound from an accepted* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *carrier must consume the displayed source row, tail threshold, two visible tail indices, dyadic tolerance, diameter bound, and window-stability row of Definition 1746.1. In particular, the consumer receives a finite diameter witness for the displayed tail window, not Real completeness, quotient real equality, trichotomy, point selection, open-cover compactness, or a standard bridge.*

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{RealIntervalEnclosure}^{\uparrow}$ consumer. The consumer-facing evidence is the displayed carrier $K = (S, T, I, J, D, B, W, H, C, P, N)$, so the source and tail threshold are the only rows from which the tail request can begin.

The visible tail indices then determine the readback. The consumer reads I and J as finite rows beyond T, consults the dyadic tolerance row D, and obtains the diameter comparison only from B.

Window stability prevents replacement of the route by a hidden cofinal tail. The row W keeps the source observations, two indices, and bound inside the same finite $\mathsf{StreamName}^{\uparrow}$ window family.

Transport, replay, provenance, and local naming finish the consumption route. They preserve the same rows for $\mathsf{RegSeqRat}^{\uparrow}$ and interval-enclosure consumers and add no coordinate for completeness, quotient equality, trichotomy, point selection, compactness, or a standard bridge. □

Theorem 1746.5 (Cauchy tail diameter fast-modulus consumption)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer that uses a* $\mathsf{FastCauchyModulus}^{\uparrow}$ *threshold to read a tail-diameter bound from an accepted* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *carrier must consume the displayed source row, threshold row, two visible tail indices, dyadic tolerance, diameter bound, and window-stability row of Definition 1746.1. The threshold route must pass through the fast or doubled threshold discipline of chapter 1781, so the selector is displayed in the finite packet and is not hidden in a completed Real limit, quotient stream equality, countable choice, trichotomy, point selection, open-cover compactness, or bridge datum.*

Proof. Project the accepted $\mathsf{CauchyTailDiameter}^{\uparrow}$ carrier. Its analytic rows are S, T, I, J, D, B, W, and no consumer can obtain the diameter bound without first reading the source, threshold, visible tail indices, tolerance, bound, and window-stability rows.

Apply the tail-window handoff and RegSeqRat consumption theorems above. Together they force every regular-rational or Real-facing read to pass through the same finite tail-window route before the bound can be used.

Now read the $\mathsf{FastCauchyModulus}^{\uparrow}$ route from chapter 1781. Its doubling-window theorem exposes the fast or doubled threshold discipline, and its NameCert obligations require that threshold selector to be part of the displayed carrier surface.

Thus a fast-modulus consumer supplies only a visible threshold discipline for the same tail-diameter rows. Since neither carrier contains a hidden selector, completed limit, quotient equality,

choice schedule, trichotomy row, point selector, compactness witness, or bridge datum, none is exported by the consumption route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailDiameter}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailDiameter}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1746.1, Definition 1746.2, Definition 1746.3, Definition 1746.4, and Definition 1746.5 seed the finite tail-diameter witness packet over source streams or $\mathsf{RegSeqRat}^{\uparrow}$ rows, displayed thresholds, visible tail indices, dyadic tolerances, diameter bounds, window-stability rows, and fast-modulus threshold discipline.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailDiameter}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient real equality, trichotomy, point selection, open-cover compactness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, dyadic bound exactness, window stability, transport, replay, provenance, local naming, tail-window handoff, and RegSeqRat consumption for BEDC.Derived.CauchyTailDiameterUp.

Constructive story: A $\mathsf{CauchyTailDiameter}^{\uparrow}$ packet is generated from a source stream or RegSeqRat row, a tail threshold, two visible tail indices, a DyadicRatCore tolerance row, a dyadic diameter-bound row, a StreamName window-stability row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1747

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailComposition}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ packet records the finite rule by which two displayed regular-Cauchy tail budgets are composed before a Real-facing consumer may read a seal. It sits over the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, the shared budget route of chapter 1741, the symmetric comparison surface of chapter 1738, and the diameter witness of chapter 1746. The packet gives Real and Completion consumers one finite tail-composition surface instead of a hidden cofinal tail, quotient stream, selected limit, or choice-like threshold selector.

Definition 1747.1 (Regular Cauchy tail composition carrier)**.** *A* $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T_1, T_2, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. T_1 *is the first regular-Cauchy tail threshold row, naming the displayed point after which the source observations may be read inside the first budget.*
2. T_2 *is the second regular-Cauchy tail threshold row, naming the displayed point after which the refinement observations may be read inside the second budget.*
3. B *is the composed tail-budget row; it records the finite threshold obtained by reading* T_1 *and* T_2 *in order and taking the displayed common tail region exposed by the packet.*
4. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row, requiring all source, intermediate, and refined reads to use the same finite window family.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested tolerance, the intermediate slack, and the final comparison budget.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that connects the window reads and dyadic tolerances to the regular rational sequence surface.*
7. E *is the tail-evidence row; it records that every accepted read beyond* B *is admitted through the two displayed tail refinements rather than through a quotient stream or a hidden limit.*

8. *H stores componentwise* hsame *transport for the two thresholds, composed budget, windows, dyadic ledger, regular-rational handoff, evidence, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from T_1 to T_2, from the two thresholds to B, from B to W, D, R, E, and then to the provenance and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailComposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T_1, T_2, B, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, countable choice, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailCompositionUp.RegularCauchyTailCompositionTasteGate`
`single carrier alignment`

Theorem 1747.2 (Regular Cauchy tail composition NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *carrier*

$$Q = (T_1, T_2, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed tail thresholds, composed tail budget, shared window row, dyadic ledger, regular-rational handoff, tail-evidence row, transport, replay, provenance, and local naming rows;*
2. *threshold ordering, requiring T_1 to be read before T_2 and requiring the composed budget B to be downstream of both threshold rows;*
3. *dyadic budget exactness, requiring the tolerance ledger D to record the requested tolerance, the intermediate slack, and the final comparison budget used by B;*
4. *window stability, requiring W to keep the source, intermediate, and refined observations inside one finite* $\mathsf{StreamName}^{\uparrow}$ *window family;*
5. *regular-rational handoff, requiring R to expose the same readback route used by the* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer before any* $\mathsf{Real}^{\uparrow}$ *seal is read;*
6. *componentwise transport and replay, requiring H and C to preserve the two thresholds, composed budget, windows, dyadic ledger, handoff, evidence, provenance, and local naming rows;*
7. *refusal of quotient stream equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, countable choice, Lean verification, and bridge checking as local obligations.*

No tail-composition obligation is stored outside these displayed rows.

Proof. Project the accepted carrier. Its analytic rows are exactly the two thresholds T_1, T_2, composed budget B, shared windows W, dyadic ledger D, regular-rational handoff R, and tail evidence E.

The threshold obligations are read in order. A consumer first opens T_1, then opens the refinement threshold T_2, and only then may read the composed budget B as the displayed common tail region.

The numerical and window obligations are finite. The tolerance ledger D records the requested tolerance together with the intermediate slack, while W requires the source, intermediate, and refined observations to remain in one StreamName window family.

The structural rows preserve only this route. Componentwise transport H and displayed replay C carry the same coordinates through provenance P and local naming N. Since none of the excluded objects is a coordinate of Q, none can become a local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}$ obligation. □

Theorem 1747.3 (Regular Cauchy tail-budget associativity)**.** *Let*

$$Q = (T_1, T_2, B, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *carrier. Two successive finite tail refinements through* T_1 *and then* T_2 *compose into the single displayed threshold* B*. Consequently, any third finite tail read that consumes the result may read the composed budget either as the already displayed row* B *or as the replayed route*

$$T_1 \longrightarrow T_2 \longrightarrow B,$$

and the two readings preserve the same W, D, R, E, H, C, P, N *rows. The comparison is a finite threshold-composition statement; it is not a quotient equality of streams, a selected limit, a countable-choice schedule, trichotomy, or an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the successive refinement route. Carrier admission exposes T_1 and T_2 as two threshold rows in one packet, so the second refinement is not a detached cofinal tail.

Replay the continuation row C. Its threshold part reads T_1 before T_2 and then exposes the single composed budget row B. Thus a consumer that asks for the combined tail sees one displayed threshold after the two finite inputs have been read.

Now compare with the direct read of B. The transport row H preserves $T_1, T_2, B, W, D, R, E, P, N$ componentwise, and the dyadic ledger D and window row W are shared by both readings.

Therefore the two readings are the same local NameCert route through the carrier. Since the route never opens a quotient coordinate, selected limit, choice schedule, trichotomy row, or completeness theorem, associativity is only the finite composition of displayed tail thresholds. □

Theorem 1747.4 (Regular Cauchy tail composition Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read admitted by an accepted* $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ *carrier factors through the finite route*

$$(T_1, T_2) \longmapsto B \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a Real or Completion consumer must first read the two displayed tail thresholds, the composed tail budget, the finite $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the tail-evidence row. No such read bypasses this route through quotient stream equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, countable choice, Lean verification, or bridge checking.*

Proof. Start with a Real-facing request. The carrier gives no immediate Real seal; the request must first identify the two threshold rows T_1, T_2 and the composed budget B.

The finite observation route then passes through W, D, R, E. The shared StreamName windows determine which source and refined observations are visible, the DyadicRatCore ledger fixes the tolerance budget, and the RegSeqRat handoff supplies the regular rational readback used by the tail-evidence row.

The structural rows finish the route. Componentwise transport and displayed replay preserve the same data into provenance and local naming before the terminal $\mathsf{Real}^{\uparrow}$ consumer may read the result.

The nonescape clause follows from the carrier inventory. Quotient equality, trichotomy, ambient completeness, selected limits, countable choice, Lean verification, and bridge checking are not coordinates of Q, so no Real-facing read can extract them from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1747.1, Definition 1747.2, Definition 1747.3, and Definition 1747.4 seed the finite composition carrier for regular-Cauchy tail budgets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold ordering, dyadic budget exactness, window stability, regular-rational handoff, componentwise transport, displayed replay, tail-budget associativity, and Real route nonescape for BEDC.Derived.RegularCauchyTailCompositionUp.

Constructive story: A $\mathsf{RegularCauchyTailComposition}^{\uparrow}$ packet is generated from two regular-Cauchy tail thresholds, one composed tail-budget row, a shared StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat handoff, tail evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1748

A Concrete Naming Certificate for $\mathsf{UniformLimitContinuous}^{\uparrow}$

Following Bishop–Bridges and Kelley-style uniform convergence of continuous maps, the $\mathsf{UniformLimitContinuous}^{\uparrow}$ packet records the standard constructive handoff from a uniformly convergent displayed family to a continuous displayed endpoint. It refines the uniform-limit certificate of chapter 463 by adding the continuity-modulus row that every endpoint consumer must read before it treats the sealed limit as a $\mathsf{ContinuousMap}^{\uparrow}$ object.

The chapter uses the $\mathsf{StreamName}^{\uparrow}$ finite-window surface, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ rational handoff, $\mathsf{Real}^{\uparrow}$ seal rows, and the $\mathsf{ContinuousMap}^{\uparrow}$ modulus discipline as anchors. It does not introduce quotient-function equality, selected host limits, open-cover compactness, countable choice, or ambient completeness.

Definition 1748.1 (Uniform-limit-continuous carrier)**.** *A* $\mathsf{UniformLimitContinuous}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, U, M, W, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. *F is the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *family, read only through certified source-window, target-window, and modulus rows;*
2. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *row tying the family to one uniform-limit endpoint surface;*
3. *M is the shared continuity-modulus ledger used by the family and by the endpoint handoff;*
4. *W is the finite window schedule containing source requests, target tolerances, and the uniform threshold rows read from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the same finite windows and uniform-limit comparisons;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal exposed by the uniform-limit route;*
7. *H stores componentwise* hsame *transport for the family, uniform-limit row, shared modulus, finite windows, regular-rational handoff, endpoint seal, continuation routes, provenance, and local naming row;*
8. *C stores the displayed* Cont *routes for reading the continuous-family modulus, entering the uniform-limit ledger, applying the shared window threshold, reaching the regular handoff, and exposing the endpoint as a continuous map read;*

9. *P is the* Pkg *provenance over* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLimitContinuous}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows, their transports, their continuation routes, package provenance, and local naming provenance.

Lean4 ✓ `BEDC.Derived.UniformLimitContinuousUp.UniformLimitContinuousTasteGate single carrier alignment`

Theorem 1748.2 (Uniform-limit-continuous NameCert obligations)**.** *For the carrier of Definition 1748.1, the local* $\mathsf{NameCert}_{\mathsf{UniformLimitContinuous}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet $Q = (F, U, M, W, R, E, H, C, P, N)$;*
2. *uniform-window stability, requiring every endpoint-facing read to use the same* $\mathsf{UniformLimit}^{\uparrow}$ *row U, finite window schedule W, and shared threshold rows;*
3. *continuity-modulus handoff, requiring the endpoint continuity read to factor through the displayed modulus ledger M before it is exported as a* $\mathsf{ContinuousMap}^{\uparrow}$ *read;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the regular-rational handoff R and endpoint seal E to be reached only through the uniform-limit row and the finite windows;*
5. *ledger refusal for quotient-function equality, selected limits, open-cover compactness, countable choice, ambient completeness, and any continuity proof not routed through F, U, M, W, R, E.*

Proof. Carrier admission is projection from Definition 1748.1. The packet lists the continuous family, uniform-limit row, shared modulus, finite windows, regular handoff, endpoint seal, transports, continuations, provenance, and local certificate.

Uniform-window stability follows because the only endpoint-facing route in C enters the uniform-limit row U and reads the same window schedule W. The transport rows in H move those coordinates componentwise, so the accepted classifier cannot switch to a pointwise or externally selected threshold.

The continuity-modulus handoff is the displayed route through M. The packet has no coordinate for an ambient open-cover proof, quotient of functions, selected host limit, countable-choice selector, or completeness theorem, so those objects cannot serve as NameCert obligations. □

Theorem 1748.3 (Uniform-limit-continuous continuity handoff)**.** *Every continuity read of the endpoint map exported by an accepted* $\mathsf{UniformLimitContinuous}^{\uparrow}$ *carrier factors through the finite route*

$$F, U, M, W \longmapsto R \longmapsto E.$$

Thus the endpoint is available as a $\mathsf{ContinuousMap}^{\uparrow}$ *read only after the displayed continuous-family row, uniform-limit row, shared continuity modulus, finite window schedule, regular-rational handoff,* $\mathsf{Real}^{\uparrow}$ *seal, and visible* hsame, Cont, Pkg, *and* NameCert *provenance have been consumed. No quotient-function equality, selected limit, open-cover compactness, countable choice, or ambient completeness is exported by the packet.*

Proof. Begin with a consumer asking for continuity of the endpoint. By Definition 1748.1, the only continuity-facing entrance is the displayed family row F together with the shared modulus ledger M.

Next the request enters the uniform-limit row U and the finite window schedule W. These rows supply the uniform threshold and finite comparisons needed before the regular-rational handoff R can be read.

Finally the endpoint seal E is exposed through the same continuation and provenance rows. Applying Definition 1748.2, there is no exported coordinate for quotient-function equality, selected host limits, open-cover compactness, countable choice, or ambient completeness, so every continuity read has the stated finite factorization. □

Theorem 1748.4 (Uniform-limit-continuous public export)**.** *Let $Q = (F, U, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{UniformLimitContinuous}^{\uparrow}$ *carrier. The public endpoint-continuity export of Q is exactly the* $\mathsf{ContinuousMap}^{\uparrow}$ *endpoint read obtained through the rows*

$$F, U, M, W, R, E, H, C, P, N.$$

Every consumer first reads the displayed continuous-family row F, the uniform-limit row U, the shared modulus row M, the finite window schedule W, the regular-rational handoff R, and the endpoint seal E, and then transports and replays that endpoint through H, C, P, N. The export refuses quotient-function equality, selected limits, countable choice, ambient completeness, and any continuity coordinate not routed through these rows.

Proof. Apply Definition 1748.2. It identifies the accepted packet rows as the whole local NameCert surface and makes the shared modulus, finite windows, regular handoff, endpoint seal, transport, replay, provenance, and naming rows visible.

Next apply Definition 1748.3. The endpoint continuity read factors through F, U, M, W, then through R, and finally through E, with H, C, P, N preserving the same route.

The two statements have the same carrier rows. Since neither statement supplies quotient-function equality, selected host limits, countable-choice selectors, ambient completeness, or an external continuity proof coordinate, the public export is precisely the displayed $\mathsf{ContinuousMap}^{\uparrow}$ endpoint read. □

Theorem 1748.5 (Uniform-limit-continuous obligation-closure package)**.** *The carrier admission of Definition 1748.1, the NameCert obligation row of Definition 1748.2, the continuity handoff of Definition 1748.3, and the public export of Definition 1748.4 give the full local obligation surface for* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformLimit}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *reads inside an accepted* $\mathsf{UniformLimitContinuous}^{\uparrow}$ *packet. The package excludes quotient-function equality, selected host limits, open-cover compactness, countable choice, and ambient completeness.*

Proof. Project the accepted carrier. The definition exposes only the rows $F, U, M, W, R, E, H, C, P, N$, so every local obligation must be read through the displayed continuous-family, uniform-limit, modulus, window, handoff, endpoint, transport, replay, provenance, and naming coordinates.

Apply Definition 1748.2. It identifies the carrier-admission and refusal duties attached to those rows, including the BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ provenance.

Next apply Definition 1748.3 and Definition 1748.4. The first theorem routes endpoint continuity through F, U, M, W, R, E, and the second theorem exports that route as the public $\mathsf{ContinuousMap}^{\uparrow}$ read. Since neither route adds quotient-function equality, selected host limits, open-cover compactness, countable choice, or ambient completeness, the four displayed rows form exactly the stated obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitContinuous}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformLimitContinuous}^{\uparrow}$ packet is generated from a displayed ContinuousMap family, a UniformLimit row, a shared continuity modulus, finite StreamName and DyadicRatCore windows, a RegSeqRat handoff, a RealUp endpoint seal, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitContinuous}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 1748.1, Definition 1748.2, Definition 1748.3, Definition 1748.4, and Definition 1748.5 form the finite obligation surface for uniform limits of continuous maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLimitContinuous}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient-function equality, selected host limits, open-cover compactness, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-window stability, continuity-modulus handoff, RealUp seal nonescape, and ledger refusal for the same finite packet.

Chapter 1749

A Concrete Naming Certificate for $\mathsf{UniformLimitInterchange}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{UniformLimitInterchange}^{\uparrow}$ packet records the finite handoff by which continuity data for a finite family of displayed real-valued maps is preserved when the family is consumed through a uniform-limit route. It consumes the uniform-limit endpoint packet of chapter 463, the continuous-map rows of chapter 130, the shared-window modulus service of chapter 1741, the regular rational readback discipline of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes the Real and Completion handoff for finite uniform-limit continuity consumers; it does not claim host function-space completeness, quotient equality of maps, selected limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1749.1 (UniformLimitInterchange carrier)**.** *A* $\mathsf{UniformLimitInterchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, L, M, R, S, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose source and target Real windows are visible to the packet.*
2. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *row that reads the same finite family through its shared uniform-tail route.*
3. *M is the* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *evidence row, recording the single modulus ledger used by the continuity rows and by the uniform-limit route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite window observations after the shared modulus row has been displayed.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed from the regular readback and made available to Real and Completion consumers.*
6. *T is the interchange transport row, asserting that continuity-facing reads of F and endpoint-facing reads of L use the same M, R, S route.*
7. *H records componentwise* hsame *transport for the continuous-map ledger, uniform-limit row, shared modulus evidence, regular readback, Real seal, interchange transport, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from F to L, from both rows to M, from M to R, from R to S, and from this route to the transport row T.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformLimitInterchange}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformCauchyModulus}$ $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLimitInterchange}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, M, R, S, T, H, C, P, N$. It has no coordinate for host function-space completeness, quotient equality of maps, selected host limits, open-cover compactness, pointwise-only convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformLimitInterchangeUp.UniformLimitInterchangeTasteGate single carrier alignment`

Theorem 1749.2 (UniformLimitInterchange NameCert obligations)**.** *For every accepted* $\mathsf{UniformLimitInterchange}^{\uparrow}$ *carrier $I = (F, L, M, R, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformLimitInterchange}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger,* $\mathsf{UniformLimit}^{\uparrow}$ *route, shared* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *evidence,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, interchange transport, replay, provenance, and local naming rows;*
2. *continuity-route exactness, requiring every continuity-facing read of F to use the same displayed family rows consumed by L;*
3. *shared-modulus exactness, requiring the continuity rows and the uniform-limit row to consult the single modulus evidence row M before a readback is exposed;*
4. *RegSeqRat readback exactness, requiring R to be obtained only after F, L, M have been displayed;*
5. *Real-seal handoff, requiring S to be consumed only after R and only along the route replayed by C;*
6. *componentwise transport and displayed replay for F, L, M, R, S, T, P, N through H and C;*
7. *refusal of host function-space completeness, quotient equality of maps, selected host limits, open-cover compactness, pointwise-only convergence, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No uniform-limit interchange obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 1749.1. The only analytic rows are the finite continuous-map ledger F, the uniform-limit route L, the shared modulus row M, the regular readback R, the Real seal S, and the transport row T.

The continuity and limit routes meet at the shared modulus. A continuity-facing read cannot bypass M, because the carrier contains no second modulus row, and a uniform-limit read cannot expose a readback before F, L, M have been displayed.

The Real-facing boundary is downstream. The replay row C places R after F, L, M and places S after R, while the transport row H moves only the listed coordinates.

The provenance and local naming rows package the same finite route. Since no coordinate names host function-space completeness, quotient map equality, selected limits, open-cover compactness, pointwise-only convergence, or ambient Real completeness, none of those objects is a local NameCert obligation. □

Theorem 1749.3 (UniformLimitInterchange Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read exported by an accepted* $\mathsf{UniformLimitInterchange}^{\uparrow}$ *carrier factors through the finite route*

$$F, L \longmapsto M \longmapsto R \longmapsto S \longmapsto T, H, C, P, N.$$

Thus the consumer receives a finite uniform-limit continuity handoff from continuous-map rows to a Real seal through the shared modulus evidence, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the displayed transport and replay rows. The route exports no host function-space completeness, quotient equality of maps, selected host limits, open-cover compactness, pointwise-only convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real or Completion consumer of the accepted packet. The endpoint seal S is not an entrance row; the carrier first exposes the finite continuous-map ledger F and the $\mathsf{UniformLimit}^{\uparrow}$ route L.

The shared modulus row is then forced. By the shared-modulus obligation in Definition 1749.2, both the continuity rows and the limit row consult the same M before the regular readback row R is available.

After R is displayed, the Real seal S can be consumed. The interchange transport row T states that the continuity-facing and endpoint-facing requests follow this same route, and the structural rows H, C, P, N preserve and name it.

Composing these route projections gives the displayed factorization. Because every step is a row of the finite packet and the packet has no coordinate for the excluded analytic objects, the handoff cannot export any stronger host-level principle. □

Theorem 1749.4 (UniformLimitInterchange shared modulus determinacy)**.** *Let* $I = (F, L, M, R, S, T, H, C, P, N)$ *be an accepted* $\mathsf{UniformLimitInterchange}^{\uparrow}$ *carrier. Every consumer route from the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *rows of* F *and the* $\mathsf{UniformLimit}^{\uparrow}$ *route* L *to* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the* $\mathsf{Real}^{\uparrow}$ *seal factors through the same displayed shared-modulus row* M *before it reaches* R *and* S*.*

Equivalently, the finite gate for the interchange is determined by the carrier rows $F, L, M, R, S, T, H, C, P, N$*: no host function-space completeness, quotient equality of maps, selected limit, open-cover compactness, pointwise-only convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness can replace the row* M *in a certified readback route.*

Proof. Project the accepted carrier through Definition 1749.1. Its analytic surface contains the continuous-map ledger F, the uniform-limit row L, the shared modulus evidence M, the regular readback R, the Real seal S, and the transport row T.

Apply Definition 1749.2. The shared-modulus exactness row identifies M as the single row consulted by both the continuity-facing reads of F and the limit-facing read of L before any $\mathsf{RegSeqRat}^{\uparrow}$ readback is exposed.

Apply Definition 1749.3. Since every Real or Completion consumer reaches S only through the displayed route

$$F, L \longmapsto M \longmapsto R \longmapsto S \longmapsto T, H, C, P, N,$$

both R and S are downstream of that same M.

The remaining excluded objects are absent from the classifier coordinates. They therefore cannot supply an alternate gate for the finite readback route or replace the shared modulus row inside the local $\mathsf{NameCert}_{\mathsf{UniformLimitInterchange}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitInterchange}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitInterchange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1749.1, Definition 1749.2, Definition 1749.3, and Definition 1749.4 seed the finite continuity handoff from continuous-map rows and a uniform-limit route to RegSeqRat readback and a Real seal.

Formal verification: FormalStatus(NameCert$_{\text{UniformLimitInterchange}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function-space completeness, quotient equality of maps, selected host limits, open-cover compactness, pointwise-only convergence, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuity-route exactness, shared-modulus exactness, RegSeqRat readback exactness, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformLimitInte

Constructive story: A UniformLimitInterchange$^{\uparrow}$ packet is generated from a finite ContinuousMap family ledger, a UniformLimit route, shared UniformCauchyModulus evidence, RegSeqRat readback, a RealUp seal, an interchange transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1750

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$

Following Bishop–Bridges regular Cauchy sequence discipline, the $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ packet records the finite fact that once a regular rational name is accepted at an earlier tail window, later tail windows may reuse the same tolerance ledger without opening a new completion object. It sits between the selected-window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the finite equality route of chapter 1738, the tail-threshold determinacy surface of chapter 1423, and the terminal real-name seal of chapter 38. The packet supplies the monotone tail-extension certificate consumed by the Real and Completion route; it is not quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1750.1 (Regular Cauchy tail monotonicity carrier)**.** *A* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{tail}} = (E, L, R, D, T, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the earlier* $\mathsf{StreamName}^{\uparrow}$ *tail-window row whose accepted regular Cauchy read is already visible.*
2. *L is the later* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, displayed as an extension of the same source schedule beyond E.*
3. *R is the single* $\mathsf{RegSeqRat}^{\uparrow}$ *source row read through both E and L.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the earlier and later reads.*
5. *T is the monotone tail-extension row, asserting that the accepted tail comparison at E remains accepted when the later window L is substituted under the same source row R and tolerance ledger D.*
6. *A is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold-determinacy row used to confirm that later windows do not change the chosen tail threshold for the displayed tolerance.*
7. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row available only after E, L, R, D, T, A have been displayed.*
8. *H records componentwise* hsame *transport for the earlier and later windows, source row, tolerance ledger, monotone extension row, threshold-determinacy row, seal handoff, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from the earlier window to the later window, from both windows to the shared regular source, from the source to the dyadic tolerance ledger, from the ledger to the monotone extension row, and from the extension row through threshold determinacy to the* $\mathsf{Real}^{\uparrow}$ *handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, R, D, T, A, Q, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, hidden cofinal tails, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMonotonicityUp.RegularCauchyTailMonotonicityUpTasteGate`
`single carrier alignment`

Definition 1750.2 (Regular Cauchy tail monotonicity obligation surface)**.** *For an accepted* RegularCauchyTailMono *carrier*

$$M_{\text{tail}} = (E, L, R, D, T, A, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the earlier* $\mathsf{StreamName}^{\uparrow}$ *window, later* $\mathsf{StreamName}^{\uparrow}$ *window, one* $\mathsf{RegSeqRat}^{\uparrow}$ *source, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, monotone tail-extension row,* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold-determinacy row,* $\mathsf{Real}^{\uparrow}$ *seal handoff, transport, replay, provenance, and local naming rows;*

2. *common-source exactness, requiring E and L to read the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*

3. *tolerance-ledger exactness, requiring the earlier and later tail reads to consult the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*

4. *monotone extension exactness, requiring T to be read only after E, L, R, D are visible and only as the later-window extension of the earlier accepted read;*

5. *threshold-determinacy replay, requiring A to reuse the displayed tolerance and source data rather than choosing a new threshold outside the packet;*

6. *componentwise transport and displayed replay for $E, L, R, D, T, A, Q, P, N$ through H and C;*

7. *refusal of quotient stream equality, selected limits, countable choice, trichotomy, host real equality, hidden cofinal tails, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No monotone tail-extension obligation is stored outside these rows.

Theorem 1750.3 (Regular Cauchy tail monotone window extension)**.** *Let $M_{\text{tail}} = (E, L, R, D, T, A, Q, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ *carrier. Any accepted regular Cauchy tail read at the earlier window E extends to the later window L exactly through the displayed route*

$$E \longmapsto L \longmapsto R \longmapsto D \longmapsto T \longmapsto A \longmapsto H, C, P, N.$$

Thus the later-window read remains a finite $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *computation over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It does not introduce quotient stream equality, selected limits, countable choice, trichotomy, host real equality, hidden cofinal tails, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier of Definition 1750.1. Carrier admission exposes the earlier window E, later window L, shared source R, tolerance ledger D, monotone extension row T, threshold-determinacy row A, and structural rows.

The earlier read cannot move to L by changing source data. Common-source exactness in Definition 1750.2 forces both windows to consult R, while tolerance-ledger exactness forces both reads to consult D.

The monotone extension is the only analytic transition row. It is read as T after E, L, R, D are already visible, and the threshold-determinacy row A confirms that the later window has not introduced a second tail threshold for the same displayed tolerance.

Transport and replay preserve this finite route. The rows H, C, P, N move, replay, source, and name the same packet, so no excluded quotient, limit, choice, order-comparison, hidden-tail, or completeness coordinate is available. □

Theorem 1750.4 (Regular Cauchy tail monotonicity Real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read exported by an accepted* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ *carrier factors through the earlier window E, the later window L, the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the monotone tail-extension row T, the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold-determinacy row A, and the structural rows H, C, P, N before the seal handoff row Q may be consumed. The consumer therefore receives a finite monotone-tail certificate, not quotient stream equality, selected limits, countable choice, trichotomy, host real equality, hidden cofinal tails, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ or completion-facing request. The handoff row Q is downstream of the displayed carrier, so the request first reads the earlier and later $\mathsf{StreamName}^{\uparrow}$ windows E, L and the shared $\mathsf{RegSeqRat}^{\uparrow}$ source R.

The tolerance accounting is then fixed by D. The monotone tail-extension row T uses that same ledger to pass from the accepted earlier tail read to the later-window read, while A records that the tail threshold is determined by the displayed source and tolerance data.

The structural rows do not create a new endpoint. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same route through E, L, R, D, T, A.

Only after this route has been displayed may the consumer read Q. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, trichotomy, host equality, hidden cofinal tails, or ambient completeness, the seal handoff cannot export any of them. □

Falsifiable prediction: Every $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ Real-facing read factors through earlier and later StreamName windows, one RegSeqRat source row, a DyadicRatCore tolerance ledger, a monotone tail-extension row, RealCauchyModulus threshold determinacy, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, cauchytailequivalence, realcauchymodulus, real: $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an earlier tail window, a later tail window, one regular rational source, one tolerance ledger, a monotone extension row, threshold determinacy, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1750.1, Definition 1750.2, Definition 1750.3, and Definition 1750.4 seed the finite monotone-tail extension certificate between regular Cauchy names and Real seals.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailMonotonicity}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyTailMonotonicityUp.RegularCauchyTailMonotonicityUpTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, hidden cofinal tails, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source exactness, tolerance-ledger exactness, monotone extension exactness, threshold-determinacy replay, Real seal factorization, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyTailMonotonicityUp.

Constructive story: A RegularCauchyTailMonotonicity$^{\uparrow}$ packet is generated from earlier and later StreamName tail windows, one RegSeqRat source row, a DyadicRatCore tolerance ledger, a monotone tail-extension row, RealCauchyModulus threshold determinacy, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Chapter 1751

A Concrete Naming Certificate for $\mathsf{UniformLimitContinuity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformLimitContinuity}^{\uparrow}$ packet records the finite BEDC surface for the theorem that a uniform limit of continuous real-valued maps is continuous. It reads the $\mathsf{UniformLimit}^{\uparrow}$ route of chapter 463, the $\mathsf{ContinuousMap}^{\uparrow}$ rows of chapter 130, the $\mathsf{UniformModulus}^{\uparrow}$ consumer surface, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows, and the terminal $\mathsf{Real}^{\uparrow}$ seal rows. The packet claims no pointwise-only convergence theorem, selected host limit, quotient function space, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1751.1 (UniformLimitContinuity carrier)**.** *A* $\mathsf{UniformLimitContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, M, T, R, E, G, U, H, C, P, N)$$

with the following displayed coordinates:

1. *F is the* $\mathsf{UniformLimit}^{\uparrow}$ *family row naming the uniformly convergent real-valued map family and its limiting endpoint route.*
2. *M is the shared modulus row that every continuity request and every uniform-tail request reads before a tolerance is consumed.*
3. *T is the tail ledger row inherited from the uniform-limit packet, exposing only the accepted shared-threshold comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular handoff row reached from the same family, modulus, and tail rows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the limiting endpoint after the regular handoff has been displayed.*
6. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the approximating maps and the limiting graph surface visible to downstream consumers.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *or uniform-continuity consumer row that receives the shared modulus data before reading continuity evidence.*
8. *H stores componentwise* hsame *transports for F, M, T, R, E, G, U, package provenance, and the local naming row.*
9. *C stores the displayed* Cont *replay route from uniform-tail exposure through regular handoff, real seal, continuous-map graph, and uniform-continuity consumption.*

10. *P is the* Pkg *provenance over* UniformLimitContinuity$^\uparrow$, UniformLimit$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformLimitContinuity}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these displayed rows, their componentwise transports, their continuation replay routes, package provenance, and local naming provenance.

Theorem 1751.2 (UniformLimitContinuity NameCert obligations)**.** *For the carrier of Definition 1751.1, the local* NameCert$_{\text{UniformLimitContinuity}^\uparrow}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the packet $K = (F, M, T, R, E, G, U, H, C, P, N)$;*
2. *uniform-tail exposure, requiring continuity-facing reads to pass through the shared modulus row M and tail ledger row T before any limiting endpoint is exposed;*
3. *continuity handoff, requiring every* ContinuousMap$^\uparrow$ *consumer to read the graph row G only after the uniform-limit route has exposed R and E;*
4. *transport, replay, and provenance stability through the displayed H, C, P, N rows;*
5. *nonescape from pointwise-only convergence, host-selected limits, quotient function equality, open-cover compactness, and ambient* Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformLimitContinuityUp.UniformLimitContinuityUpTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1751.1. The displayed tuple names the uniform-limit family, shared modulus, tail ledger, regular handoff, real seal, continuous-map graph, uniform-continuity consumer row, transports, continuations, provenance, and local certificate.

For the route obligations, read an endpoint-facing continuity request through C. The route first exposes F, M, T, then reaches R and E, and only then allows the continuous-map graph row G and uniform-continuity consumer row U to be used.

The transport and provenance clauses are componentwise in H and P, with replay recorded by C and local naming by N. Since the carrier contains no coordinate for the excluded analytic structures, those structures cannot be exported by the local certificate. □

Theorem 1751.3 (UniformLimitContinuity continuous-map handoff)**.** *Every* ContinuousMap$^\uparrow$ *consumer of the limiting endpoint of an accepted* UniformLimitContinuity$^\uparrow$ *carrier factors through the* UniformLimit$^\uparrow$ *shared threshold and seal route*

$$F \longmapsto M \longmapsto T \longmapsto R \longmapsto E$$

before reading the continuous-map graph row G or the uniform-continuity consumer row U. Thus the limiting continuity data are consumed only after the shared modulus, uniform tail ledger, regular handoff, and real seal have been displayed.

Proof. Start with a ContinuousMap$^\uparrow$ consumer of the limiting endpoint. By the carrier definition, the only limiting endpoint row is E, and the only route to it is the displayed replay path through F, M, T, R.

Apply the uniform-tail exposure and continuity handoff clauses of Definition 1751.2. They force the consumer to read the shared threshold row and tail ledger before the regular handoff and seal are available, and they place the graph row G after that route.

The remaining rows H, C, P, N record transport, replay, package provenance, and naming provenance for the same path. Hence a continuous-map consumer factors through the uniform-limit threshold and seal route before it reads continuity data. □

Theorem 1751.4 (UniformLimitContinuity modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or uniform-continuity consumer of an accepted* $\mathsf{UniformLimitContinuity}^{\uparrow}$ *carrier* $K = (F, M, T, R, E, G, U, H, C, P, N)$ *must read the shared modulus row* M *before using the tail ledger* T*, regular handoff* R*, real seal* E*, continuous-map graph* G*, or consumer row* U*. Thus all uniform-continuity consumption in the packet factors through the same displayed modulus row before it reaches any endpoint, graph, or downstream consumer surface.*

Proof. Project the accepted carrier along Definition 1751.1. The only modulus-bearing coordinate available to a uniform-continuity request is the shared row M, while T, R, E, G, U occur later in the displayed carrier route.

Apply the uniform-tail exposure clause of Definition 1751.2. It requires continuity-facing reads to pass through M and the tail ledger before the limiting endpoint route is exposed.

Apply Definition 1751.3 to the endpoint-facing part of the same consumer. That theorem places the route through F, M, T, R, E before G or U can be read, so a $\mathsf{UniformModulus}^{\uparrow}$ or uniform-continuity consumer cannot reach the later rows without first passing through M.

The remaining rows H, C, P, N do not introduce a second route. They transport, replay, package, and name the same carrier path, so they preserve the modulus handoff rather than bypassing it. □

Theorem 1751.5 (UniformLimitContinuity Cauchy criterion handoff)**.** *Every accepted* $\mathsf{UniformLimitContinuity}^{\uparrow}$ *carrier* $K = (F, M, T, R, E, G, U, H, C, P, N)$ *sends its shared modulus row* M *and tail ledger* T *through the* $\mathsf{UniformLimit}^{\uparrow}$ *row, the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *or* $\mathsf{CauchyCriterion}^{\uparrow}$ *surface, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before any* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer reads continuity. The route exports no pointwise-only convergence theorem and no selected host limit.*

Proof. Project the accepted carrier through Definition 1751.1. The rows F, M, T, R, E are the only uniform-limit, shared-modulus, tail-ledger, regular-readback, and Real-seal coordinates available before the continuity-facing rows G and U.

Apply Definition 1751.4. It forces every $\mathsf{UniformModulus}^{\uparrow}$ or uniform-continuity consumer to read M before T, R, E, G, U can be used.

Apply Definition 1751.3. That handoff places the endpoint-facing continuity read after the route through F, M, T, R, E, so a $\mathsf{ContinuousMap}^{\uparrow}$ consumer also waits for the shared modulus, tail ledger, regular readback, and Real seal.

The $\mathsf{UniformCauchyCriterion}^{\uparrow}$ or $\mathsf{CauchyCriterion}^{\uparrow}$ surface reads the same tail ledger and regular-seal route rather than a second limit object. Since the carrier has no coordinate for pointwise-only convergence or a selected host limit, those excluded objects cannot mediate the continuity read. □

Theorem 1751.6 (UniformLimitContinuity nonescape boundary)**.** *An accepted* $\mathsf{UniformLimitContinuity}^{\uparrow}$ *carrier exports only the displayed uniform-limit family row, shared modulus row, tail ledger, regular handoff, real seal, continuous-map graph, uniform-continuity consumer row, componentwise transports, continuation replay, package provenance, and local naming row of Definition 1751.1. It exports no pointwise-only convergence theorem, no host limit selector, no quotient function equality, no open-cover compactness principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Inspect the carrier tuple. Its visible rows are precisely $F, M, T, R, E, G, U, H, C, P, N$, and its consumer order is the replay route recorded by C.

Use Definition 1751.3. Any continuity-facing endpoint read passes through the shared threshold, tail ledger, regular handoff, and real seal before the continuous-map graph is read.

The excluded objects would require an additional carrier coordinate or a replay route outside C. Neither is present in the packet, so the public surface is exhausted by the displayed finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformLimitContinuity}^{\uparrow}$ packet is generated from a uniform-limit family row, a shared modulus row, a uniform tail ledger, a regular handoff, a real seal, continuous-map graph rows, a uniform-continuity consumer row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1751.1, Definition 1751.2, Definition 1751.3, and Definition 1751.6 bind the seed carrier, obligation surface, continuous-map handoff, and nonescape boundary for the finite uniform-limit-continuity packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLimitContinuity}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.UniformLimitContinuityUp`

Bridge status: none

Not claimed: This chapter does not close pointwise-only convergence, selected host limits, quotient function equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a checked Lean certificate.

Upgrade path: A formal carrier theorem and bridge theorem remain verification-axis work over the same finite uniform-limit-continuity surface.

Chapter 1752

A Concrete Naming Certificate for $\mathsf{CantorCompleteness}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Cantor's nested-interval principle, the $\mathsf{CantorCompleteness}^{\uparrow}$ packet records the finite naming boundary for a descending sequence of closed located intervals whose displayed diameters tend to zero and whose regular-rational readback reaches a terminal $\mathsf{Real}^{\uparrow}$ seal. It sits between the nested interval surfaces of chapter 320, the located endpoint discipline of chapter 394, the tail-diameter witness of chapter 1746, the window and readback route of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The completion-facing consumer route is shared with chapter 1421, which packages the same nested-interval width-window, RegSeqRat handoff, and RealUp seal as an explicit intersection consumer. The chapter isolates the constructive Cantor-completeness interface as NameCert data; it does not assert Heine–Borel compactness, open-cover compactness, choice-selected points, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1752.1 (Cantor completeness carrier)**.** *A* $\mathsf{CantorCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{Cantor}} = (I, L, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{NestedInterval}^{\uparrow}$ *row listing the nested closed interval spine, with each later interval displayed as a subinterval of the earlier row.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row for the left and right endpoints of every displayed interval in the inspected finite window.*
3. *R is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger asserting that the visible interval diameters are eventually bounded by each requested dyadic tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the diameter ledger and by the regular-rational readback route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row identifying which finite prefix of the nested interval spine is inspected at the requested tolerance.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracting the compatible regular rational name from the endpoint and diameter rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, R, D, W, Q have been displayed.*

8. *H records componentwise* hsame *transport for the nested intervals, located endpoints, diameter ledger, dyadic tolerances, window row, regular readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the nested interval spine through located endpoints, diameter-to-zero evidence, dyadic tolerances, StreamName windows, RegSeqRat readback, and the terminal Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorCompleteness}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, R, D, W, Q, E, H, C, P, N$. It has no coordinate for an open cover, a Heine–Borel subcover, a choice-selected point, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorCompletenessUp.CantorCompletenessTasteGate single carrier alignment`

Theorem 1752.2 (Cantor completeness NameCert obligations)**.** *For every accepted* $\mathsf{CantorCompleteness}^{\uparrow}$ *carrier*

$$K_{\mathrm{Cantor}} = (I, L, R, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested closed interval spine, located endpoint rows, diameter-to-zero ledger, dyadic tolerance row, StreamName window row, RegSeqRat readback row, Real seal, transport, replay, provenance, and local naming rows;*

2. *nesting exactness, requiring each visible later interval in I to be read as a closed subinterval of the preceding visible interval;*

3. *endpoint locatedness, requiring L to display the left and right endpoint rows used by every visible interval comparison;*

4. *diameter-tail exactness, requiring R to answer each requested tolerance only through the dyadic row D and the finite window row W;*

5. *readback exactness, requiring Q to extract a regular rational name only from the compatible endpoint and diameter rows exposed by I, L, R, D, W;*

6. *seal exactness, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to be downstream of the displayed RegSeqRat readback rather than an ambient completeness assumption;*

7. *componentwise transport, displayed replay, package provenance, and local naming for the same rows through H, C, P, N;*

8. *refusal of open-cover compactness, Heine–Borel subcovers, choice-selected points, quotient real equality, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No Cantor-completeness obligation is stored outside these rows.

Proof. Project the accepted carrier. Carrier admission exposes the nested interval spine I, located endpoint rows L, diameter ledger R, dyadic tolerance row D, StreamName window row W, RegSeqRat readback Q, Real seal E, and structural rows H, C, P, N.

The interval information is read first. Nesting exactness keeps each visible later interval inside the preceding closed interval, and endpoint locatedness ensures that the comparisons are made only through the displayed endpoint rows.

The diameter and window rows control the limiting read. For each requested dyadic tolerance, the ledger R consults D and the finite StreamName window W before any readback can be produced.

The regular-rational and Real rows are downstream. The row Q repacks the compatible endpoint and diameter data as a regular rational name, and E seals that name as the displayed $\mathsf{Real}^{\uparrow}$ output.

Transport, replay, provenance, and local naming preserve this same route. Since no coordinate carries an open cover, Heine–Borel subcover, choice-selected point, quotient equality, trichotomy, or ambient completeness principle, none of those objects can become a local NameCert obligation. □

Theorem 1752.3 (Cantor completeness nested diameter handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{CantorCompleteness}^{\uparrow}$ *carrier factors through the finite route*

$$I, L \longmapsto R, D, W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives the nested closed interval rows, located endpoint rows, diameter-to-zero ledger, dyadic tolerances, finite StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, package provenance, and local NameCert. The route exports no open-cover compactness, Heine–Borel subcover, choice-selected point, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer request for the Real-facing output. The request can enter the packet only through the accepted carrier, so the first visible data are the nested interval spine I and the located endpoint row L.

Next read the diameter route. The ledger R answers the requested tolerance by consulting the dyadic row D and the finite window row W, keeping the argument inside the displayed tail of the interval spine.

The readback row then supplies the candidate name. Because Q is downstream of I, L, R, D, W, the regular rational name is produced from the same compatible endpoint and diameter data rather than from an external point selector.

Finally, the Real seal E and structural rows H, C, P, N package the same finite route for consumers. No step in the route opens an open cover, invokes a subcover theorem, chooses a point from the nested family, quotients real names, decides trichotomy, or assumes ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorCompleteness}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1752.1, Definition 1752.2, and Definition 1752.3 seed the finite nested closed interval, diameter-tail, regular-rational readback, and Real-seal packet for Cantor-style completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Heine–Borel compactness, choice-selected points, quotient real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nesting exactness, endpoint locatedness, diameter-tail exactness, readback exactness, seal exactness, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.CantorCompletenessUp.

Constructive story: A $\mathsf{CantorCompleteness}^{\uparrow}$ packet is generated from nested closed interval rows, located endpoint rows, a diameter-to-zero ledger, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1753

A Concrete Naming Certificate for $\mathsf{BishopModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopModulusReal}^{\uparrow}$ packet records the explicit-modulus real-name carrier used when a constructive real name is read through finite precision data rather than through a detached equality principle. It sits over the dyadic tolerance arithmetic of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the fast real-name route of chapter 1589, and the completion-facing modulus route of chapter 1691. The packet makes the Bishop modulus visible as finite BEDC data before any FastCauchyReal or FastCauchyCompletion consumer treats the name as Real-facing data.

Definition 1753.1 (BishopModulusReal carrier)**.** *A* $\mathsf{BishopModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the explicit Bishop modulus row. It records, for each finite precision request, the displayed index bound and the local tail discipline that every downstream read must consume.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened only at the windows named by M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for those windows, including dyadic radii, endpoint comparisons, rounding rows, and tolerance arithmetic.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S and D under the modulus budget exposed by M.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, S, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the modulus row, stream windows, dyadic ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each finite Real-facing precision request through M, S, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopModulusReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{FastCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopModulusRealUp.BishopModulusRealTasteGate single carrier alignment`

Theorem 1753.2 (BishopModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{BishopModulusReal}^{\uparrow}$ *carrier $B = (M, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopModulusReal}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed Bishop modulus, finite stream-window row, dyadic tolerance ledger, regular rational readback, terminal Real seal, and structural rows H, C, P, N;*
2. *modulus exactness, requiring every finite precision request to read the row M before any stream window, dyadic tolerance, regular readback, or Real seal is exposed;*
3. *window and readback compatibility, requiring S and D to be consumed under the same displayed modulus budget before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R may be read;*
4. *Real-seal nonescape, requiring E to remain terminal with respect to M, S, D, R and to expose no detached Real object, selected limit, host equality, or ambient completeness theorem;*
5. *componentwise transport and provenance, requiring H, C, P, N to preserve and name only the displayed route under* hsame *transport,* Cont *replay, package provenance, and local certification.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1753.1. The visible packet consists of the Bishop modulus row M, finite stream window S, dyadic tolerance ledger D, regular readback R, Real seal E, and structural rows H, C, P, N.

Modulus exactness follows from the placement of M at the front of the route. A Real-facing precision request first fixes the displayed bound, and every later observation is read inside the finite budget named by that bound.

Window and readback compatibility read chapter 126, chapter 291, and chapter 301 in order. The window S and ledger D are consumed before R is available, so regular rational data cannot be detached from the displayed Bishop modulus.

The Real-seal nonescape clause reads chapter 38. The row E is terminal and downstream of M, S, D, R, while H, C, P, N transport, replay, package, and name exactly that route.

Since quotient stream equality, selected limits, countable choice, host real equality, trichotomy, and ambient Real completeness are absent from the carrier, no local NameCert obligation can export them. □

Theorem 1753.3 (BishopModulusReal fast-Cauchy handoff)**.** *Let $B = (M, S, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopModulusReal}^{\uparrow}$ *carrier. Every* $\mathsf{FastCauchyReal}^{\uparrow}$ *or* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *consumer reads the packet only through the displayed modulus route*

$$M \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently those consumers receive an explicit Bishop modulus, finite stream windows, dyadic tolerance arithmetic, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming before any Real-facing datum is available. The route cannot use quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a FastCauchyReal or FastCauchyCompletion read of B. By Definition 1753.1, the only Real-facing coordinate is E, and the definition places it downstream of the modulus row M, stream window S, dyadic ledger D, and regular readback row R.

The first part of the route is fixed by the modulus and finite-window rows. The row M supplies the explicit finite bound, S exposes only the corresponding stream observations, and D supplies the dyadic tolerances and endpoint comparisons for those observations.

The second part of the route is fixed by regular readback and the Real seal. The row R is consumed only after S and D, and E seals the Real-facing datum only after the regular row has been displayed.

Finally, H, C, P, N preserve and name the same dependency order. Because the excluded objects have no coordinate in the carrier and no replay edge in the displayed route, every FastCauchyReal or FastCauchyCompletion read factors through the modulus route shown in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopModulusReal}^{\uparrow}$ packet is generated from an explicit Bishop modulus row, StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopModulusReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1753.1, Definition 1753.2, and Definition 1753.3 seed the finite explicit-modulus Real-name route consumed by FastCauchyReal and FastCauchyCompletion packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopModulusReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, window and readback compatibility, Real-seal nonescape, componentwise transport, provenance, and local naming for the same displayed packet.

Chapter 1754

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ packet records the finite precision-to-tail threshold service used by regular Cauchy real names before any completed limit, quotient stream equality, or choice-selected modulus is available. It reads the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational source route of chapter 301, the terminal real-name seal of chapter 38, the displayed tail-diameter witness of chapter 1746, and the shared-window budget of chapter 1741. The packet names the one-source threshold row that later $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, and $\mathsf{UniformCauchyModulus}^{\uparrow}$ consumers may cite when a requested precision must be turned into a visible finite tail service.

Definition 1754.1 (Regular Cauchy tail modulus carrier)**.** *A* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, T, W, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular Cauchy tail service is being queried.*
2. *Q is the precision-request row, recording the requested dyadic tolerance as a visible consumer input.*
3. *T is the displayed tail-threshold row returned for the request Q on the source S.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window service showing which source observations are visible once the threshold T has been read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row against which the displayed request and tail observations are compared.*
6. *H records componentwise* hsame *transport for the source, request, threshold, window service, dyadic tolerance, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the source and precision request through the threshold, the finite window service, and the dyadic tolerance row.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, T, W, D, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1754.2 (Regular Cauchy tail modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *carrier $K = (S, Q, T, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular source, precision request, displayed threshold, finite window service, dyadic tolerance, transport, replay, provenance, and local naming rows;*
2. *request exactness, requiring Q to be the precision request consumed by the source row S and by no detached schedule;*
3. *tail-threshold exactness, requiring T to be the displayed tail threshold returned for the pair (S, Q);*
4. *window handoff, requiring W to expose only finite* $\mathsf{StreamName}^{\uparrow}$ *observations available after the threshold T is read;*
5. *dyadic tolerance compatibility, requiring the request row Q and all tail-window comparisons exported through W to be checked against the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
6. *transport and replay stability, requiring H and C to preserve the same source-request-threshold-window-tolerance route;*
7. *provenance and local naming, requiring P and N to package exactly the displayed rows S, Q, T, W, D, H, C;*
8. *refusal of quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert obligations.*

No regular Cauchy tail-modulus obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailModulusUp.RegularCauchyTailModulusTasteGate single carrier alignment`

Theorem 1754.3 (Regular Cauchy tail modulus window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that uses a tail threshold from an accepted* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *carrier must read the rows in the order*

$$S, Q, T, W, D, H, C, P, N.$$

Thus the consumer sees the regular source, the requested precision, the displayed threshold, the finite window service, the dyadic tolerance row, transport, replay, provenance, and local naming before using the threshold. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailModulusUp.RegularCauchyTailModulusWindowHandoff`

Definition 1754.4 (Regular Cauchy tail modulus diameter-control carrier)**.** *A diameter-control* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K^{\diamondsuit} = (S, Q, T, W, D, B, U, H, C, P, N)$$

whose first five rows are the regular source S, precision request Q, displayed threshold T, finite window W, and $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance D of Definition 1754.1. The row B is a* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *certificate row over the same S, Q, T, W, D route, and the row U is a* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *consumer row over the same source, request, threshold, window, tolerance, and diameter-control row. The remaining rows H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}$ *packaging for exactly these displayed rows.*

The classifier compares accepted packets only through $S, Q, T, W, D, B, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1754.5 (Regular Cauchy tail modulus diameter control)**.** *If a diameter-control* RegularCauchyTailModulu *carrier*

$$K^{\diamondsuit} = (S, Q, T, W, D, B, U, H, C, P, N)$$

is accepted, then every two displayed tail indices beyond the threshold row T read their diameter bound through the $\mathsf{CauchyTailDiameter}^{\uparrow}$ *row B after the shared route S, Q, T, W, D and before the structural rows H, C, P, N. Thus the threshold is a concrete finite diameter-control service: the same source, precision request, threshold, finite window, and dyadic tolerance that admit the tail handoff also admit the displayed tail-diameter bound.*

The conclusion uses no quotient stream equality, selected limit, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge data.*

Proof. Read the accepted packet by projection. The first five projections recover the route S, Q, T, W, D, and Definition 1754.3 forces any consumer of the threshold to pass through that route before the structural rows are available.

Apply the finite-window reading of chapter 1746 to the displayed row B. Since B is part of the same accepted packet and its carrier coordinates are exactly S, Q, T, W, D, every pair of visible tail indices beyond T obtains its bound from the same finite window and the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row.

The remaining rows H, C, P, N only transport, replay, package, and name the displayed data. They add no equality of quotient streams, no selected limit, no choice principle, no trichotomy, no compactness principle, no ambient completeness claim, and no bridge datum. □

Theorem 1754.6 (Regular Cauchy tail modulus uniform window consumption)**.** *Every* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *consumer using the threshold T of an accepted diameter-control* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *carrier must first consume the same route*

$$S, Q, T, W, D, B$$

before reading its local uniform-modulus row U or the structural rows H, C, P, N. Consequently the uniform modulus is not a detached schedule: it is a finite-window consumer of the displayed source, request, threshold, window, dyadic tolerance, and $\mathsf{CauchyTailDiameter}^{\uparrow}$ *diameter-control row.*

Proof. Project the accepted packet to the threshold route and then to the diameter-control row. The preceding theorem supplies the finite tail-diameter service through B, so any consumer that claims to use the same threshold already has the route S, Q, T, W, D, B in view.

Apply the shared-window budget described by chapter 1741. The $\mathsf{UniformCauchyModulus}^{\uparrow}$ row U is admitted only as a consumer over the displayed route, so its scheduling data is read after the diameter-control row and cannot be separated from the finite window that witnesses the tail bound.

The structural rows preserve this dependency by transport, replay, provenance, and local naming. They do not introduce quotient stream equality, selected limits, choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 1754.7 (Regular Cauchy tail modulus RegSeqRat consumer)**.** *Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read that consumes a* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *threshold from an accepted diameter-control carrier*

$$K^{\diamondsuit} = (S, Q, T, W, D, B, U, H, C, P, N)$$

must first expose the same $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*,* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *diameter row* B*, and* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *budget row* U *already named by the packet. Consequently the threshold is consumed only as the finite route*

$$S, Q, T, W, D, B, U, H, C, P, N,$$

and not as quotient stream equality, a selected limit, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge data.*

Proof. Start with Definition 1754.3. It forces any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ threshold consumer to read the regular source, precision request, displayed threshold, finite window, and dyadic tolerance rows before the structural rows.

Apply Definition 1754.5. Once the same threshold route is visible, the tail-diameter row B supplies the finite diameter-control service over S, Q, T, W, D.

Then use Definition 1754.6. The uniform Cauchy modulus row U is downstream of S, Q, T, W, D, B, so a RegSeqRat or Real consumer cannot detach the budget from the visible window or from the dyadic tolerance ledger.

The remaining rows H, C, P, N only transport, replay, package, and name this route. Since the carrier contains no excluded host-side coordinate, the displayed rows exhaust the consumer service. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1754.1, Definition 1754.2, Definition 1754.3, Definition 1754.4, Definition 1754.5, and Definition 1754.6 seed the finite precision-to-tail threshold packet over RegSeqRat source rows, dyadic precision requests, displayed thresholds, finite StreamName window services, DyadicRatCore tolerances, CauchyTailDiameter rows, UniformCauchyModulus rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exactness, tail-threshold exactness, window handoff, dyadic tolerance compatibility, transport, replay, provenance, and local naming for future BEDC.Derived.RegularCauchyTailModul obligations.

Constructive story: A $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, a precision request, a displayed tail threshold, a finite StreamName window service, a DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1755

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$

Bishop–Bridges constructive regular Cauchy analysis treats a modulus as reusable finite evidence: once a displayed precision has produced a tail threshold, later consumers may cite that threshold through monotone window data rather than choose another modulus. The $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ packet records this threshold-reuse service over the precision-to-tail carrier of chapter 1754, the later-window discipline of chapter 1750, the finite observation surface of chapter 126, the regular rational source route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The certificate gives Real and uniform-limit consumers one displayed handoff for reusing a tail threshold; it is not quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1755.1 (Regular Cauchy tail modulus reuse carrier)**.** *A* $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R_{\text{reuse}} = (S, Q, T, E, L, W, D, M, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular Cauchy tail service is being reused.*
2. *Q is the displayed precision-request row already consumed by the source.*
3. *T is the* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *threshold row returned for (S, Q).*
4. *E is the earlier* $\mathsf{StreamName}^{\uparrow}$ *tail window on which the threshold T was first read.*
5. *L is the later* $\mathsf{StreamName}^{\uparrow}$ *tail window that asks to reuse the same threshold and precision request.*
6. *W is the shared window-family row, requiring E and L to expose observations from the same finite stream-name service.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by both the original tail request and the reused later-window comparison.*
8. *M is the* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ *handoff row, showing that the later window may read the earlier tail evidence without changing S, Q, T, D.*
9. *U is the reuse row; it records that a downstream Real or uniform-limit consumer cites T through M rather than selecting a fresh threshold.*

10. *H records componentwise* hsame *transport for the source, request, threshold, earlier window, later window, shared window family, dyadic tolerance, monotone handoff, reuse row, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from S, Q to T, from T to E, L, W, D, M, U, and then through provenance and local naming.*
12. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$, $\mathsf{RegularCauchyTailModulus}^{\uparrow}$, $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, T, E, L, W, D, M, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, hidden cofinal tails, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailModulusReuseUp.RegularCauchyTailModulusReuseTasteGate`
`single carrier alignment`

Theorem 1755.2 (Regular Cauchy tail modulus reuse NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ *carrier*

$$R_{\text{reuse}} = (S, Q, T, E, L, W, D, M, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational source, precision request, reused threshold, earlier and later stream windows, shared window family, dyadic tolerance, monotone handoff, reuse row, transport, replay, provenance, and local naming rows;*
2. *threshold provenance, requiring T to be the displayed* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *row returned from the same (S, Q) pair stored in the carrier;*
3. *window compatibility, requiring E and L to belong to the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window family W;*
4. *tolerance reuse, requiring the original threshold read and the later-window read to consult the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
5. *monotone handoff admission, requiring M to witness that L extends the accepted tail read from E without changing S, Q, T, D;*
6. *reuse exactness, requiring U to cite the displayed threshold T through M rather than produce a new threshold row;*
7. *componentwise transport and displayed replay for $S, Q, T, E, L, W, D, M, U, P, N$ through H and C;*
8. *refusal of quotient stream equality, selected limits, countable choice, trichotomy, hidden cofinal tails, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No tail-modulus reuse obligation is stored outside these rows.

Theorem 1755.3 (Regular Cauchy tail modulus reuse monotone handoff)**.** *Let*

$$R_{\text{reuse}} = (S, Q, T, E, L, W, D, M, U, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ *carrier. Any Real-facing or uniform-limit-facing read that reuses the threshold* T *must factor through the finite route*

$$S, Q \longmapsto T \longmapsto E, L, W, D \longmapsto M \longmapsto U \longmapsto H, C, P, N.$$

Consequently the consumer receives the same source row, precision request, tail threshold, window family, and dyadic tolerance that generated the original modulus row. The handoff is monotone threshold reuse; it is not a quotient equality of streams, a selected limit, a countable-choice schedule, trichotomy, a hidden cofinal tail, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Falsifiable prediction: Every $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ Real-facing or uniform-limit-facing read factors through one RegSeqRat source row, one precision request, one displayed RegularCauchyTailModulus threshold, earlier and later StreamName windows, one shared window family, one DyadicRatCore tolerance row, a RegularCauchyTailMonotonicity handoff, a reuse row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-tail-modulus, regular-cauchy-tail-monotonicity, stream-name, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the original threshold row and the later-window monotone handoff that authorizes reuse by downstream consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1755.1, Definition 1755.2, and Definition 1755.3 seed the finite threshold-reuse certificate over a regular rational source, displayed precision request, reused tail threshold, earlier and later finite windows, dyadic tolerance, monotone handoff, reuse row, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, hidden cofinal tails, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold provenance, window compatibility, tolerance reuse, monotone handoff admission, reuse exactness, componentwise transport, displayed replay, and consumer route nonescape for BEDC.Derived.RegularCauchyTailModulusReuseUp.

Constructive story: A $\mathsf{RegularCauchyTailModulusReuse}^{\uparrow}$ packet is generated from one RegSeqRat source row, one precision request, one displayed RegularCauchyTailModulus threshold, earlier and later StreamName windows, one shared window family, one DyadicRatCore tolerance row, a RegularCauchyTailMonotonicity handoff, a reuse row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1756

A Concrete Naming Certificate for $\mathsf{RegularCauchySquareRoot}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ packet records the finite square-root readback service available for a displayed nonnegative regular Cauchy rational name before any completed real quotient, trichotomy principle, or host real square-root theorem is invoked. It sits beside the regular rational source discipline of chapter 301, the finite stream-window service of chapter 126, the dyadic arithmetic ledger of chapter 291, the monotone and modulus tail services of chapter 1750 and chapter 1754, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a visible route for a constructive square-root approximation while keeping total square root on arbitrary reals, quotient real equality, selected limits, and ambient completeness outside the local certificate.

Definition 1756.1 (Regular Cauchy square-root carrier)**.** *A* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, N, W, D, B, T, M, S, E, H, C, P, A)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose observations form the regular rational name to be square-rooted.*
2. *N is the nonnegativity witness row, recording finite lower-bound data sufficient for the displayed square-root route and no trichotomy coordinate.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window service exposing exactly the source observations used by the square-root ledger.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row containing dyadic interval endpoints, midpoint arithmetic, squaring comparisons, tolerance splits, and rational order certificates used by the ledger.*
5. *B is the dyadic bisection or Newton ledger, recording for each requested precision the finite interval or Newton step, the squared-endpoint check, and the selected dyadic output approximation.*
6. *T is the* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *request-to-threshold row used to determine which source tail must be visible for the requested square-root precision.*
7. *M is the* $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$ *row showing that later source windows preserve the accepted tail and tolerance ledger.*

8. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by replaying B against R, N, W, D, T, M.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the source, nonnegativity witness, windows, dyadic ledger, square-root ledger, tail rows, and readback row have been displayed.*
10. *H records componentwise* hsame *transport for the source, nonnegativity witness, windows, dyadic ledger, square-root ledger, tail rows, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every square-root precision request through $R, N, W, D, B, T, M, S, E, H, P,$*
12. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyTailModulus}^{\uparrow}$, $\mathsf{RegularCauchyTailMonotonicity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySquareRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, N, W, D, B, T, M, S, E, H, C, P, A$. It has no coordinate for full $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, total square root on arbitrary reals, quotient real equality, selected limits, host real square-root functions, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySquareRootUp.TasteGate.RegularCauchySquareRootTasteGate single carrier alignment`

Theorem 1756.2 (Regular Cauchy square-root NameCert obligations)**.** *For every accepted* RegularCauchySquareRo *carrier*

$$Q = (R, N, W, D, B, T, M, S, E, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySquareRoot}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational source, nonnegativity witness, finite StreamName windows, DyadicRatCore arithmetic ledger, square-root ledger, tail modulus row, tail monotonicity row, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *nonnegativity exactness, requiring N to certify the displayed source row R rather than an external representative or completed real;*
3. *window exactness, requiring W to expose precisely the source observations read by the square-root ledger and no quotient stream equality;*
4. *dyadic square-root accounting, requiring B to use only dyadic interval bisection or Newton steps whose endpoint squares, midpoint arithmetic, and tolerance splits are stored in D;*
5. *tail-modulus compatibility, requiring T to supply the visible threshold for each requested square-root precision;*
6. *monotone-tail stability, requiring M to preserve the accepted source and tolerance ledger when later windows are read;*
7. *regular readback exactness, requiring S to be obtained by replaying the displayed ledger rows R, N, W, D, B, T, M;*
8. *Real seal nonescape, requiring E to be downstream of the regular readback and not a host square-root value;*
9. *componentwise transport, displayed replay, provenance, and local naming, requiring H, C, P, A to preserve exactly the same square-root packet;*

10. *refusal of full* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, total square root on arbitrary reals, quotient real equality, selected limits, host real square-root functions, Lean verification, and bridge checking as local certificate data.*

No regular Cauchy square-root obligation is stored outside these rows.

Theorem 1756.3 (Regular Cauchy square-root Newton handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that reads a square-root approximation from an accepted* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *carrier must factor through the displayed route*

$$R \longmapsto N \longmapsto W \longmapsto D \longmapsto B \longmapsto T \longmapsto M \longmapsto S \longmapsto E.$$

Thus the consumer sees the nonnegative source row, finite observation windows, dyadic bisection or Newton ledger, tail modulus threshold, monotone tail stability row, regular readback, and Real seal before using the square-root output. The handoff exports no full $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, total square root on arbitrary reals, quotient real equality, selected limit, or host real square-root function.*

Falsifiable prediction: Every $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ Real-facing read factors through a nonnegative RegSeqRat source row, StreamName finite windows, a DyadicRatCore bisection or Newton ledger, RegularCauchyTailModulus threshold service, RegularCauchyTailMonotonicity stability, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regularcauchytailmodulus, regularcauchytailmonotonicity, real: $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries nonnegative-source data, finite source windows, a dyadic square-root ledger, tail threshold and monotonicity rows, regular readback, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySquareRoot}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySquareRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1756.1, Definition 1756.2, and Definition 1756.3 seed the finite square-root packet over a nonnegative regular rational source, finite StreamName windows, DyadicRatCore bisection or Newton data, tail modulus and monotonicity rows, regular readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySquareRoot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, total square root on arbitrary reals, quotient real equality, selected limits, host real square-root functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nonnegativity exactness, window exactness, dyadic square-root accounting, tail-modulus compatibility, monotone-tail stability, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySquareRootUp.

Constructive story: A $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ packet is generated from a nonnegative RegSeqRat source row, StreamName finite windows, a DyadicRatCore bisection or Newton ledger, RegularCauchyTailModulus threshold service, RegularCauchyTailMonotonicity stability, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1757

A Concrete Naming Certificate for $\mathsf{BoundedSet}^{\uparrow}$

Following Hausdorff–Kelley metric-space practice and Bishop–Bridges constructive analysis, the $\mathsf{BoundedSet}^{\uparrow}$ packet records bounded subset data as a finite NameCert surface rather than as an ambient host-set fact. It sits over the metric carrier of chapter 57, the algebraic and order-facing real rows of chapter 766 and chapter 38, the ball-containment ledger of chapter 1450, and the consumer routes of chapter 1503 and chapter 339. The packet exposes exactly the center, radius, subset classifier, and ball-containment evidence needed by diameter, finite-net, precompactness, and completion-facing arguments.

Definition 1757.1 (BoundedSet carrier)**.** *A* $\mathsf{BoundedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, S, c, r, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier whose distance rows are the only metric data available to the packet.*
2. *S is the bounded subset classifier row. It records accepted membership reads as finite classifier entries over X.*
3. *c is the displayed center row, or the displayed witness row naming the local center used by the boundedness certificate.*
4. *r is the* $\mathsf{Real}^{\uparrow}$ *or rational bound row read through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface.*
5. *B is the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger asserting that every accepted membership read of S is routed through the displayed ball with center c and bound r.*
6. *H records componentwise* hsame *transport for the metric carrier, subset classifier, center or witness, real or rational bound, ball-containment ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a membership request through X, S, c, r, B, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{MetricDiameter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, c, r, B, H, C, P, N$. It has no coordinate for an ambient host set, a supremum row, compactness, open-cover compactness, a selected maximum, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier semantic name certificate`

Theorem 1757.2 (BoundedSet NameCert obligations)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{Met}^{\uparrow}$ *carrier, bounded subset classifier, displayed center or witness,* $\mathsf{Real}^{\uparrow}$ *or rational bound read through* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger, transport, replay, provenance, and local naming rows;*
2. *subset classifier stability, requiring membership reads of S to be preserved componentwise by the displayed transport row H;*
3. *ball containment, requiring each accepted membership read to pass through the* $\mathsf{MetricBall}^{\uparrow}$ *ledger B with center c and bound r;*
4. *real-bound nonescape, requiring the bound row r to be read only through the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface and not as a host supremum, selected maximum, trichotomy proof, or completeness theorem;*
5. *transport and replay stability for the same X, S, c, r, B, P, N route under* hsame *transport and* Cont *replay.*

No BoundedSet NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 1757.1. The only analytic rows are the metric carrier X, subset classifier S, displayed center or witness c, bound row r, and ball-containment ledger B.

Carrier admission is therefore rowwise. A membership request first names the metric carrier and classifier, then reads the center or witness and the bound comparison row before any containment claim is available.

Subset classifier stability and transport stability are forced by the single transport row H. Since H moves only the listed coordinates, it cannot replace the displayed classifier by a host subset or by an unrecorded predicate.

Ball containment is forced by B. The continuation row C replays membership reads through X, S, c, r, B, so a downstream consumer sees the $\mathsf{MetricBall}^{\uparrow}$ containment route before it can use boundedness.

Finally, r is a displayed $\mathsf{Real}^{\uparrow}$ or rational bound read through $\mathsf{RealAlgOrder}^{\uparrow}$. Because the carrier contains no coordinate for a supremum, selected maximum, trichotomy proof, compactness principle, open-cover argument, or ambient Real completeness, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetTasteGate single carrier alignment`

Theorem 1757.3 (BoundedSet ball-containment route)**.** *Every accepted subset membership read from a* $\mathsf{BoundedSet}^{\uparrow}$ *carrier factors through the displayed route*

$$X, S \longmapsto c, r \longmapsto B \longmapsto H, C, P, N.$$

Consequently every MetricDiameter$^{\uparrow}$ *or* UniformSpace$^{\uparrow}$ *consumer receives the* MetricBall$^{\uparrow}$ *containment ledger before it can read bounded-subset data. The route exports no host set, supremum, compactness, open-cover compactness, selected maximum, trichotomy, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Begin with an accepted membership read. By Definition 1757.1, the read is a classifier row of S over the metric carrier X and is not an ambient host-set membership fact.

The center and bound rows are opened next. The row c supplies the displayed center or witness, and r supplies the real or rational radius through the order-facing comparison surface.

The containment claim is downstream of those rows. The ledger B is the only coordinate that relates the membership read to a MetricBall$^{\uparrow}$ packet, so every boundedness read passes through B before diameter or uniform-space data can be consumed.

The structural rows preserve the same order. Componentwise transport, replay, provenance, and local naming add no coordinate for any stronger analytic principle, so the displayed factorization is the only exported route. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier ball containment route`

Theorem 1757.4 (BoundedSet real-bound nonescape)**.** *An accepted* BoundedSet$^{\uparrow}$ *carrier exports the displayed bound row r only as a* Real$^{\uparrow}$ *or rational comparison read through* RealAlgOrder$^{\uparrow}$ *and the* MetricBall$^{\uparrow}$ *containment ledger. It exports no host set, supremum, compactness, open-cover compactness, selected maximum, trichotomy, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Inspect the carrier rows. The bound coordinate r is paired with the center or witness c and the containment ledger B; it is not paired with a host-order supremum row, a selected maximal point, a cover family, or a completeness witness.

Every replay of r is governed by C. The replay starts at a membership request, opens X, S, c, r, and then reads the ball-containment ledger B before the structural rows H, P, N package the result.

Thus r is available only as part of the displayed comparison-and-containment route. Since the excluded objects have no carrier coordinate and no replay edge, they cannot escape through the bound row. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier real bound nonescape`

Theorem 1757.5 (BoundedSet obligation closure rows)**.** *For every accepted* BoundedSet$^{\uparrow}$ *carrier $Q = (X, S, c, r, B, H, C, P, N)$ of Definition 1757.1, the obligation-facing surface is exhausted by the concrete rows $X, S, c, r, B, H, C, P, N$. In particular, every* MetricDiameter$^{\uparrow}$ *or* UniformSpace$^{\uparrow}$ *consumer that reads boundedness must pass through the* MetricBall$^{\uparrow}$ *containment ledger B and the* RealAlgOrder$^{\uparrow}$ *comparison read of r before it can consume the bounded-subset claim.*

Proof. Project the carrier named in Definition 1757.1. The subset classifier is S, the displayed center or witness is c, the bound is r, and the containment evidence is B.

The only route from membership to boundedness passes through these rows. The consumer first reads X, S, c, r, then reads B as the MetricBall containment ledger, and only then reaches the replay and naming rows H, C, P, N.

Since no other analytic coordinate occurs in the carrier, diameter and uniform-space consumers cannot bypass either the ball-containment row or the real-order comparison row. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier obligation closure rows`

Theorem 1757.6 (BoundedSet MetricBall carrier obligation)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, each membership read of* S *must pass through the displayed* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B *over the same metric carrier* X*, center row* c*, and bound row* r *before it can be consumed as bounded-set data. The carrier exports no host set, subset comprehension oracle, selected maximal point, supremum row, trichotomy proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier metric ball carrier obligation`

Proof. Project the carrier of Definition 1757.1. The subset classifier S is paired with the metric carrier X, center row c, bound row r, and containment ledger B inside the same finite tuple.

Apply Definition 1757.3. It forces every accepted membership read to follow the route from X, S through c, r and then through B before the structural rows are available.

Use Definition 1757.4. The excluded host set, supremum, selected maximum, trichotomy, and ambient completeness rows have no coordinate in the packet, so the $\mathsf{MetricBall}^{\uparrow}$ ledger is the only boundedness carrier obligation. □

Theorem 1757.7 (BoundedSet RealAlgOrder radius obligation)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the radius or bound row* r *is readable only through the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface before the* $\mathsf{MetricBall}^{\uparrow}$ *containment row* B *can witness membership containment. A diameter, uniform-space, finite-net, or completion-facing consumer cannot replace* r *by a host supremum, selected maximum, trichotomy decision, compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the bound row in Definition 1757.1. It is explicitly a $\mathsf{Real}^{\uparrow}$ or rational bound read through $\mathsf{RealAlgOrder}^{\uparrow}$, and it is paired with c and B rather than with a host-order extremum.

Apply Definition 1757.4. That theorem confines every replay of r to the displayed comparison-and-containment route.

Then use Definition 1757.11. Downstream consumers receive boundedness only after the $\mathsf{MetricBall}^{\uparrow}$ containment and real-order comparison rows are displayed, so no stronger analytic principle can replace the radius obligation. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier real order radius obligation`

Theorem 1757.8 (BoundedSet RealSeparability forward link)**.** *When an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *packet over a* $\mathsf{Real}^{\uparrow}$ *metric carrier is consumed by a* $\mathsf{RealSeparability}^{\uparrow}$ *route, the consumer may read only the displayed center row* c*, radius row* r*,* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B*, and countable dense witness rows from chapter 1809. The handoff cannot infer compactness, a supremum, a selected maximum, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $Q = (X, S, c, r, B, H, C, P, N)$ using Definition 1757.1. The bounded-set side exposes the metric carrier X, membership classifier S, center row c, radius row r, containment ledger B, and structural rows H, C, P, N.

Route membership through the containment ledger as in Definition 1757.6. A separability consumer therefore receives bounded membership only after the $\mathsf{MetricBall}^{\uparrow}$ row B has paired the membership classifier with the displayed center and radius rows.

Use the radius obligation Definition 1757.7. The radius remains a displayed $\mathsf{RealAlgOrder}^{\uparrow}$ comparison input, so it cannot become a host supremum, selected maximum, trichotomy decision, compactness witness, or ambient completeness theorem.

Finally pass the finite ball-cover request to the countable dense witness surface of chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ route may consume the displayed ball data and dense rows, but the structural rows H, C, P, N preserve only that finite handoff. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier real separability forward link`

Theorem 1757.9 (BoundedSet NameCert ledger nonescape)**.** *Let $Q = (X, S, c, r, B, H, C, P, N)$ be an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier. The local* $\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}$ *ledger consumes exactly the membership classifier S,* $\mathsf{MetricBall}^{\uparrow}$ *containment row B,* $\mathsf{RealAlgOrder}^{\uparrow}$ *radius comparison row r, and structural rows H, C, P, N over the metric carrier X and center row c. No accepted ledger read can escape into host sets, suprema, selected maxima, trichotomy, compactness, finite-net existence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier namecert ledger nonescape`

Proof. Begin with Definition 1757.2. It identifies carrier admission, subset classifier stability, ball containment, real-bound nonescape, and transport-replay stability as the whole local certificate surface.

Use Definition 1757.6 for the membership side. Every accepted member of S must pass through the displayed $\mathsf{MetricBall}^{\uparrow}$ ledger B over X, c, r.

Use Definition 1757.7 for the radius side. The bound row r remains a $\mathsf{RealAlgOrder}^{\uparrow}$ comparison input and not a host extremal object or completeness theorem.

The structural rows H, C, P, N only transport, replay, source, and name this same route. Since the carrier has no coordinate for the excluded analytic objects, the NameCert ledger cannot export them. □

Theorem 1757.10 (BoundedSet subset-ball replay stability)**.** *For an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier $Q = (X, S, c, r, B, H, C, P, N)$ of Definition 1757.1, componentwise replay preserves the order*

$$X, S \longmapsto c, r \longmapsto B \longmapsto H, C, P, N.$$

Thus a transported $\mathsf{MetricDiameter}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *consumer still reads boundedness only after the subset membership row has passed through* $\mathsf{MetricBall}^{\uparrow}$ *containment and the bound row has passed through* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier subset ball replay stability`

Proof. The transport row H is componentwise, so it can move the metric carrier, subset classifier, center or witness, bound, containment ledger, replay, provenance, and local name only along their displayed coordinates.

The continuation row C replays the same dependency order. A transported membership request starts at X, S, opens c, r, reads B, and then packages the result through H, P, N.

Therefore replay stability does not create a new boundedness route. The consumer boundary remains the MetricBall containment and RealAlgOrder comparison route recorded in the carrier. □

Theorem 1757.11 (BoundedSet diameter consumer boundary)**.** *Every* $\mathsf{MetricDiameter}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier $Q = (X, S, c, r, B, H, C, P, N)$ from Definition 1757.1 receives only the finite rows $X, S, c, r, B, H, C, P, N$. Its diameter read must factor through* $\mathsf{MetricBall}^{\uparrow}$ *containment for S around c and through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface for r before any boundedness datum is available. The same boundary applies to* $\mathsf{UniformSpace}^{\uparrow}$ *consumers that use boundedness as a uniform-control input.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier diameter consumer boundary`

Proof. Start with the diameter-facing read. The carrier offers the metric row X and the classifier row S, but neither row by itself contains a diameter bound.

The center and bound rows supply the displayed comparison data. The row c names the center or witness, while r is available only through the order-facing comparison surface.

The containment ledger B is then required before the consumer can treat membership as bounded. The remaining rows H, C, P, N transport, replay, package, and name the same finite route, so no supremum, selected maximum, compactness theorem, or ambient completeness row can be read by a diameter or uniform-space consumer. □

Theorem 1757.12 (BoundedSet ball-radius ledger factorization)**.** *Every boundedness read from an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier*

$$Q = (X, S, c, r, B, H, C, P, N)$$

factors through the displayed ball-radius ledger

$$S \longmapsto c, r \longmapsto B \longmapsto H, C, P, N.$$

Here r is read only through the $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface, and B is read only as the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger. Thus no finite-net, diameter, or uniform-space consumer can use a bounded-subset claim before receiving the displayed center, radius, ball-containment, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier ball radius ledger factorization`

Proof. Project the carrier of Definition 1757.1. The subset classifier is S, the displayed center or witness is c, the radius row is r, and the containment ledger is B.

Apply Definition 1757.2. It places carrier admission, subset classifier stability, ball containment, real-bound nonescape, transport, replay, provenance, and local naming in the same finite obligation surface.

Now use Definition 1757.11. A consumer can treat the subset as bounded only after the $\mathsf{MetricBall}^{\uparrow}$ containment row and the $\mathsf{RealAlgOrder}^{\uparrow}$ comparison read have been displayed. The structural rows H, C, P, N preserve that order, so the factorization is exactly the route shown above. □

Theorem 1757.13 (BoundedSet radius ledger)**.** *Every accepted bounded-set read from a* $\mathsf{BoundedSet}^{\uparrow}$ *carrier $Q = (X, S, c, r, B, H, C, P, N)$ exposes a finite radius ledger over the displayed* $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, *and metric rows before boundedness is exported. The ledger consists of the metric carrier X, subset classifier S, center or witness row c, radius row r, and* $\mathsf{MetricBall}^{\uparrow}$ *containment row B, with transport, replay, provenance, and naming through H, C, P, N.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier radius ledger`

Proof. Project the carrier of Definition 1757.1. The radius coordinate is r, and it is paired with the metric row X, subset classifier S, center or witness row c, and containment ledger B.

Apply Definition 1757.2 and Definition 1757.4. They force every radius read through the displayed Real or rational comparison surface and exclude suprema, selected maxima, trichotomy, compactness, and ambient completeness.

Use Definition 1757.12. The resulting ledger is finite and rowwise: S leads to c, r, then to B, and finally to H, C, P, N before any boundedness datum is exported. □

Theorem 1757.14 (BoundedSet MetricBall handoff)**.** *Every membership or boundedness consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$ *must pass through the* $\mathsf{MetricBall}^{\uparrow}$ *row* B *and the* $\mathsf{MetricDiameter}^{\uparrow}$ *consumer boundary before a bound is exported. The handoff names the finite ball-radius route over* $X, S, c, r, B, H, C, P, N$ *and not a host set, selected center family, supremum, compactness theorem, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier metric ball handoff`

Proof. Start with Definition 1757.3. It sends every accepted membership read through X, S, then c, r, and then the $\mathsf{MetricBall}^{\uparrow}$ containment ledger B.

Apply Definition 1757.11 and Definition 1757.16. These theorems require a diameter-facing consumer to read the same displayed ball and radius rows before any $\mathsf{MetricDiameter}^{\uparrow}$ bound is public.

Finally use Definition 1757.20. The envelope contains only the metric, classifier, center, radius, ball, and structural rows, so the handoff cannot export the excluded host-level objects. □

Theorem 1757.15 (BoundedSet uniform-space boundary)**.** *Every* $\mathsf{UniformSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier receives only the displayed finite bound and entourage or metric rows routed through* $X, S, c, r, B, H, C, P, N$*. The carrier does not provide total boundedness, compactness, selected centers, open-cover data, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier uniform space boundary`

Proof. Project the carrier and read Definition 1757.2. The available analytic coordinates are X, S, c, r, B, while H, C, P, N only transport, replay, source, and name those rows.

Use Definition 1757.3 and Definition 1757.11. A uniform-space consumer receives boundedness only after the membership row has passed through $\mathsf{MetricBall}^{\uparrow}$ containment and the bound row has passed through the displayed Real or rational comparison surface.

Apply Definition 1757.17. Finite nets, compactness, selected centers, and completeness are downstream obligations not stored in the bounded-set packet, so the uniform-space boundary stops at the displayed finite rows. □

Theorem 1757.16 (BoundedSet diameter consumer handoff)**.** *Every diameter-facing consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$ *must receive the rows* $X, S, c, r, B, H, C, P, N$ *before it can read bounded-subset data. In particular, the consumer must pass through the* $\mathsf{Met}^{\uparrow}$ *carrier* X*, the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B*, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison row* r *before any* $\mathsf{MetricDiameter}^{\uparrow}$ *bound is public.*

The handoff exports no supremum, selected maximum, compactness theorem, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier diameter consumer handoff`

Proof. Project the carrier of Definition 1757.1. The only subset data is S, and the only bounding data is the displayed center or witness c, bound row r, and containment ledger B.

Apply Definition 1757.3. It places every membership read behind the route from X, S through c, r and then through B.

Apply Definition 1757.4. The bound row is read only through the displayed Real or rational comparison surface, so a diameter consumer cannot replace it by a supremum or selected maximum.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since compactness, trichotomy, and ambient Real completeness have no carrier coordinate, the handoff stops at the displayed bounded-set rows. □

Theorem 1757.17 (BoundedSet finite-net consumer boundary)**.** *Every finite-net or precompactness consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$ *must read boundedness through* $X, S, c, r, B, H, C, P, N$*, with* X *supplying the* $\mathsf{Met}^{\uparrow}$ *carrier,* B *supplying the* $\mathsf{MetricBall}^{\uparrow}$ *boundary, and* r *supplying the* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison data. The consumer receives a bounded-subset boundary for later finite-net construction; it does not receive compactness, an open-cover subcover, a selected finite net, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier finite net consumer boundary`

Proof. Start from a finite-net request. The carrier does not contain a finite-net row, so the request can only consume boundedness through the subset classifier S and the displayed ball data c, r, B.

Use Definition 1757.3. It requires each accepted member of S to pass through the $\mathsf{MetricBall}^{\uparrow}$ ledger before bounded-subset data is available.

Use Definition 1757.4. It keeps the radius row within the Real or rational comparison route and refuses selected maxima, suprema, and trichotomy.

The structural rows H, C, P, N export only this bounded boundary. Since a finite net, compactness theorem, open-cover subcover, and ambient completeness principle are absent from Q, the consumer must obtain those obligations elsewhere. □

Theorem 1757.18 (BoundedSet total-bounded consumer boundary)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$ *must receive, in addition to* Q*, a separate finite-net or located-net row. The bounded-set packet contributes the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger, the* $\mathsf{MetricDiameter}^{\uparrow}$ *radius or diameter boundary, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison read for* r*; it does not contribute the finite cover, center family, located membership rows, compactness theorem, open-cover subcover, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness required by total boundedness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier totally bounded consumer`

Proof. Project the bounded-set carrier. Its analytic rows are the metric source X, membership classifier S, center or witness c, radius row r, and ball-containment ledger B.

Use chapter 1450, chapter 1503, and chapter 766 to read those rows as a radius or diameter ledger: membership passes through $\mathsf{MetricBall}^{\uparrow}$, while the bound is read through $\mathsf{RealAlgOrder}^{\uparrow}$ before any downstream consumer sees it.

Now compare with chapter 127. A total-boundedness consumer needs finite centers and coverage witnesses for each displayed tolerance, or a located-net row that supplies the same finite coverage data. No such row occurs among $X, S, c, r, B, H, C, P, N$.

The structural rows H, C, P, N preserve the bounded route, but they cannot synthesize finite cover data. Therefore a $\mathsf{TotallyBounded}^{\uparrow}$ consumer may use the bounded packet only after an independent finite-net or located-net row is supplied, and it receives no compactness or finite-net existence from boundedness alone. □

Theorem 1757.19 (BoundedSet completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$ *must pass through* $X, S, c, r, B, H, C, P, N$*, the* $\mathsf{Met}^{\uparrow}$ *carrier, the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison row before it can use bounded-subset data. The carrier supplies a boundedness boundary for completion arguments, not a completed subset, not a selected completion point, not compactness, not trichotomy, and not ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the bounded-set carrier. Its analytic rows are the metric carrier X, subset classifier S, displayed center or witness c, comparison row r, and containment ledger B.

Apply Definition 1757.3. A completion-facing consumer must see membership routed through the displayed ball before it may consume boundedness.

Apply Definition 1757.4. The comparison row remains a Real or rational bound read through $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$, not a completeness assertion.

Finally transport and replay through H, C, P, N. Since the carrier has no completed-subset row, selected completion point, compactness proof, trichotomy row, or ambient Real-completeness theorem, boundedness cannot escape into those stronger completion claims. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier completion nonescape`

Theorem 1757.20 (BoundedSet closure obligation envelope)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the bounded-set obligation envelope is exactly the displayed* $\mathsf{Met}^{\uparrow}$ *carrier row* X*, subset classifier row* S*, center or witness row* c*, radius row* r *read through* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B*, and the* $\mathsf{MetricDiameter}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *consumer boundaries routed through those same rows. The structural rows* H, C, P, N *transport, replay, source, and name this envelope; they do not add compactness, a selected finite net, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1757.1. The analytic data are precisely X, S, c, r, B, while H, C, P, N are structural rows for transport, replay, provenance, and local naming.

Use Definition 1757.3 and Definition 1757.4. They force each membership and bound read through the $\mathsf{MetricBall}^{\uparrow}$ ledger and the $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ comparison surface before any consumer can use boundedness.

Finally apply Definition 1757.16 and Definition 1757.19. Diameter, uniform, and completion-facing consumers receive only that displayed envelope, so the stronger compactness and completeness claims remain outside the packet. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier closure obligation envelope`

Theorem 1757.21 (BoundedSet radius classifier stability)**.** *If an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier transports its radius data to a consumer, the transported radius classifier is stable exactly along the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison row* r *and the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B*. No consumer may replace this transport by a selected maximum, supremum, host-order comparison, trichotomy decision, or ambient completeness argument.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier radius classifier stability`

Proof. Start with the radius row in Definition 1757.1. The carrier pairs r with the displayed center or witness c and containment ledger B, not with a host extremum or selected maximum.

Apply Definition 1757.4. It confines every read of r to the Real or rational comparison route and excludes suprema, selected maxima, trichotomy, and ambient completeness.

Use Definition 1757.3. The comparison row becomes a bounded-set classifier only through the metric-ball ledger B, and componentwise transport through H can replay only that same classifier. □

Theorem 1757.22 (BoundedSet located diameter readback)**.** *Every located-diameter consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier must read diameter control through the displayed ledger*

$$X, S, c, r, B \longmapsto H, C, P, N.$$

In particular, the consumer may use the located ball and radius rows as bounded-diameter input for $\mathsf{MetricDiameter}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$*, but it cannot infer a selected diameter, a least upper bound, compactness, finite-net existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier located diameter readback`

Proof. Begin with Definition 1757.16. A diameter consumer receives boundedness only through the metric carrier, subset classifier, center or witness, bound, containment ledger, and structural rows.

Use Definition 1757.20. It identifies the same rows as the full obligation envelope for metric, ball, diameter, real-order, and uniform consumers.

Finally apply Definition 1757.21. The located ball and radius data remain classifier reads through r and B, so selected diameters, suprema, compactness, finite nets, trichotomy, and ambient completeness have no carrier coordinate. □

Theorem 1757.23 (BoundedSet MetricBall containment dependency)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the* $\mathsf{MetricBall}^{\uparrow}$ *containment dependency is an explicit L10 row: each accepted membership read of* S *must pass through the displayed carrier coordinates* X, c, r*, the radius row* r*, and the subset-ball replay ledger* B, H, C, P, N *before any boundedness claim is exported. Thus the packet supplies a dependency row for* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*, and* $\mathsf{UniformSpace}^{\uparrow}$ *consumers, not a host-set containment fact.*

Proof. Project the displayed carrier of Definition 1757.1. The only rows available to a membership read are the metric carrier X, classifier S, center or witness c, radius row r, containment ledger B, and structural rows H, C, P, N.

Apply Definition 1757.3 and Definition 1757.6. These force every accepted member of S through the same $\mathsf{MetricBall}^{\uparrow}$ containment route before the read can be consumed as bounded-set data.

Use Definition 1757.13 and Definition 1757.10. The radius row and replay row preserve the order from X, S through c, r, B and then through H, C, P, N.

Finally invoke Definition 1757.22. Located diameter and uniform-space consumers receive only that displayed ledger, so the containment dependency is an L10 row rather than an ambient host predicate. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier metric ball containment dependency`

Theorem 1757.24 (BoundedSet Real radius ledger dependency)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the radius read* r *is ledger-controlled: it is read through the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *comparison surface and the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger before boundedness reaches a consumer. No consumer may replace* r *by host real order, a selected supremum, trichotomy, compactness, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier real radius ledger`

Proof. Start with the radius coordinate in Definition 1757.1. The row r is paired with the displayed center or witness c and with the ball-containment ledger B, not with a host-order extremum.

Apply Definition 1757.7. It requires the radius or bound row to be read through the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surface before containment is available.

Use Definition 1757.13 and Definition 1757.21. They keep the radius classifier on the finite ledger $X, S, c, r, B, H, C, P, N$ and prevent replacement by a selected maximum, supremum, host-order comparison, trichotomy decision, or ambient completeness argument.

Finally apply Definition 1757.22. A located-diameter or uniform-space consumer may use the ledger as bounded-diameter input, but it cannot read any stronger real-order object from the packet. □

Theorem 1757.25 (BoundedSet real-clamping stability)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, a bounded-set radius read clamped to a located interval must pass through the displayed* $\mathsf{RealClamping}^{\uparrow}$ *clamp row, the* $\mathsf{LocatedUpperBound}^{\uparrow}$ *upper-bound witness, the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger* B*, and the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison surface before boundedness reaches finite-net, precompactness, diameter, or completion consumers. The resulting route is a finite bounded-radius handoff through* $X, S, c, r, B, H, C, P, N$*, not a selected supremum, host interval order, compactness assertion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier real clamping stability`

Proof. Start with the radius coordinate of Definition 1757.1. The row r is readable only together with X, S, c and the ball-containment ledger B.

Apply Definition 1757.24. It places every radius read behind the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ comparison surface and the $\mathsf{MetricBall}^{\uparrow}$ containment route before boundedness is public.

The clamped interval read is therefore an additional consumer of that same radius ledger. It may pass through $\mathsf{RealClamping}^{\uparrow}$ and $\mathsf{LocatedUpperBound}^{\uparrow}$ to present a located bounded interval, but those rows consume the already displayed comparison and ball-containment data rather than replacing them.

Finally use Definition 1757.17 and Definition 1757.19. Finite-net, precompactness, diameter, and completion consumers receive only the routed bounded-radius handoff, so selected suprema, host interval order, compactness, and ambient completeness remain outside the packet. □

Theorem 1757.26 (BoundedSet obligation carrier admission)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *packet* $Q = (X, S, c, r, B, H, C, P, N)$*, obligation-level carrier admission is rowwise over the displayed metric carrier* X*, subset classifier* S*, center or witness* c*, real or rational bound row* r*, ball-containment ledger* B*, componentwise transport* H*, replay row* C*, package provenance* P*, and local name* N*. No obligation consumer may admit the packet from an ambient host subset, a hidden supremum, a selected finite net, compactness, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1757.1. The displayed rows $X, S, c, r, B, H, C, P, N$ are the whole packet, so carrier admission is obtained by checking those coordinates and no external host-set predicate.

Use Definition 1757.2. Its carrier-admission clause already lists the metric, classifier, center or witness, bound, containment, transport, replay, provenance, and local naming rows as the local $\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}$ surface.

Finally apply Definition 1757.20. The envelope sends diameter, uniform, and completion-facing consumers through the same rows. Since compactness, finite-net selection, trichotomy, suprema, and ambient Real completeness do not occur in the carrier, they cannot be added at the obligation-admission boundary. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier obligation carrier admission`

Theorem 1757.27 (BoundedSet obligation classifier transport)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, subset-classifier transport at obligation level is exactly the componentwise* hsame *movement of* S *along the same* X, c, r, B *rows. A transported membership*

read remains a bounded-subset read only when it replays through H and C to the displayed ball-containment ledger; it cannot be replaced by an unrecorded host predicate, a new radius classifier, a selected maximum, or a completion-side subset equality.

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier obligation classifier transport`

Proof. Start with the subset classifier row S in Definition 1757.1. The row classifies accepted membership reads over the displayed metric carrier X, not over an ambient host set.

Apply Definition 1757.3. Any membership read must pass from X, S to c, r and then to the ledger B before the structural rows are used.

Use Definition 1757.21. It fixes the radius classifier along the comparison row r and containment ledger B. Therefore H and C may transport and replay only the same classifier route, and no selected maximum, replacement predicate, or completion equality can enter the transported obligation. □

Theorem 1757.28 (BoundedSet obligation ledger nonescape)**.** *For every accepted* $\mathsf{BoundedSet}^{\uparrow}$ *carrier* $Q = (X, S, c, r, B, H, C, P, N)$*, the obligation ledger for ball containment and real bounds is the finite route*

$$X, S, c, r, B \longmapsto H, C, P, N.$$

The row B records containment in the displayed metric ball and the row r records the real or rational bound through $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*; together they supply boundedness to* $\mathsf{MetricDiameter}^{\uparrow}$*, finite-net, precompactness, and completion-facing consumers without exporting least bounds, compactness, open-cover compactness, trichotomy, quotient equality, or host Real completeness.*

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier obligation ledger`

Proof. First read the containment side. By Definition 1757.3, membership data must pass through X, S, c, r, B before boundedness is available to a consumer.

Next read the bound side. The real-bound nonescape theorem confines r to the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ comparison route and prevents a supremum, selected maximum, trichotomy proof, or completeness theorem from being inferred.

Finally combine the consumer handoff theorems. The diameter handoff, completion nonescape, closure-obligation envelope, and located-diameter readback all reuse the same H, C, P, N structural rows. Thus metric diameter, finite-net, precompactness, and completion consumers receive the ball-containment and bound ledger only as finite boundedness evidence, not as stronger compactness or completeness data. □

Theorem 1757.29 (BoundedSet obligation consumer package)**.** *Every finite-net, precompactness,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *packet* $Q = (X, S, c, r, B, H, C, P, N)$ *receives boundedness only through the displayed carrier route*

$$X, S, c, r, B \longmapsto H, C, P, N.$$

The consumer may project metric membership, the subset classifier, the center or witness, the bound row, and the ball-containment ledger, and may then replay the same rows through componentwise transport, continuation, provenance, and local naming. It cannot replace the route by host-set boundedness, hidden compactness, a finite-net selection, a supremum, trichotomy, quotient equality, or ambient completion data.

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier obligation consumer package`

Proof. Start with carrier admission, Definition 1757.26. It admits the packet only from $X, S, c, r, B, H, C, P, N$, so a downstream consumer has no independent host-set boundedness coordinate to inspect.

Next use classifier transport, Definition 1757.27. Membership and subset movement remain tied to S over the same X, c, r, B rows and are replayed only through H and C.

Finally apply ledger nonescape, Definition 1757.28. The boundedness evidence delivered to finite-net, precompactness, diameter, uniform-space, and completion-facing consumers is exactly the displayed ball-containment and real-bound ledger followed by H, C, P, N. Since the packet contains no compactness, supremum, trichotomy, quotient, or completion coordinate, no consumer can obtain one from the obligation package. □

Theorem 1757.30 (BoundedSet metric-bornology consumer)**.** *Let $Q = (X, S, c, r, B, H, C, P, N)$ be an accepted* $\mathsf{BoundedSet}^{\uparrow}$ *packet. Every ?? consumer of Q reads exactly the route*

$$X, S, c, r, B \longmapsto H, C, P, N$$

before hereditary or finite-union bornology rows can consume the bounded-set data. The radius row r is read through $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$*, the containment row B is read through the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger, and the downstream metric-bornology packet is the bounded-family surface of chapter 4091.*

Consequently the consumer receives metric carrier data, subset-classifier data, a displayed center or witness, a real or rational radius comparison, ball containment, transport, replay, provenance, and local naming, but no host set, supremum, selected maximum, compactness proof, open-cover compactness, trichotomy, quotient equality, or ambient completion datum.

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier metric bornology consumer`

Proof. Project the bounded-set carrier of Definition 1757.1. Its analytic rows are exactly X, S, c, r, B, and its support rows are H, C, P, N.

Use Definition 1757.3. It forces every membership read through the center-radius path before boundedness reaches any downstream consumer. The radius component is the same $\mathsf{Real}^{\uparrow}$ or rational comparison read through $\mathsf{RealAlgOrder}^{\uparrow}$, and the containment component is the same $\mathsf{MetricBall}^{\uparrow}$ ledger.

Finally apply the metric-bornology surface of chapter 4091. Hereditary and finite-union bornology rows may consume bounded-set data only after the displayed route has been opened, so the consumer obtains exactly the finite bounded-family handoff and none of the excluded host principles. □

Theorem 1757.31 (BoundedSet standard-interface bridge)**.** *The public bounded-set consumer package over* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *??*, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *is exactly the bridge input for the bounded-set standard interface. An accepted carrier*

$$Q = (X, S, c, r, B, H, C, P, N)$$

exports metric carrier data, subset classification, a displayed center or witness, a real or rational bound, ball containment, metric-bornology handoff, transport, replay, provenance, and local naming. It exports no compactness proof, supremum, trichotomy, quotient equality, or completion coordinate.

Proof. Start with Definition 1757.29. It identifies the public consumer route as the displayed carrier path $X, S, c, r, B \longmapsto H, C, P, N$ and excludes hidden host-set boundedness.

Apply Definition 1757.30. This places the public route against the **??** consumer surface while preserving the $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows used by the carrier.

Finally use Definition 1757.28. The ledger supplies only ball containment and real-bound data followed by transport, replay, provenance, and naming. These rows are precisely the standard-interface bridge input, and no excluded compactness, supremum, trichotomy, quotient, or completion coordinate is present. □

Lean4 ✓ `BEDC.Derived.BoundedSetUp.BoundedSetCarrier standard interface bridge`

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1757.1, Definition 1757.2, Definition 1757.3, Definition 1757.4, Definition 1757.11, Definition 1757.15, Definition 1757.30, and Definition 1757.31 bind the mature public consumer surface and standard-interface bridge over $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, **??**, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close compactness, Heine–Borel, open-cover compactness, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Mature closure requires worked examples and completeness-facing cross-chapter treatment for the bounded-set interface.

Constructive story: A $\mathsf{BoundedSet}^{\uparrow}$ packet is generated from a MetricUp carrier, a bounded subset classifier row, a displayed center or witness, a RealUp or rational bound read through RealAlgOrderUp, a MetricBallUp containment ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1758

A Concrete Naming Certificate for $\mathsf{CauchyNullDifference}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyNullDifference}^{\uparrow}$ packet records the finite route by which two regular Cauchy rational names expose equality of their $\mathsf{Real}^{\uparrow}$ endpoints: first form the displayed regular-Cauchy difference, then certify that displayed difference as a $\mathsf{NullSequence}^{\uparrow}$, and only then allow the terminal real-name equality seal to be read. It sits between the regular rational source rows of chapter 301, the pointwise difference route of chapter 429, the zero-tail classifier of chapter 380, the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the endpoint surface of chapter 38.

Definition 1758.1 (Cauchy null-difference carrier)**.** *A* $\mathsf{CauchyNullDifference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (X, Y, D, Z, W, T, E, H, K, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are compared.*
2. *D is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row displaying the aligned pointwise difference stream $X - Y$.*
3. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-shrinkage row that consumes the displayed difference stream.*
4. *W stores the finite* $\mathsf{StreamName}^{\uparrow}$ *windows used by the source rows, the difference row, and the null-sequence consumer.*
5. *T stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance requests, slack rows, and comparison budgets used by D and Z.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *equality-seal row, readable only after the displayed difference has passed through Z.*
7. *H records componentwise* hsame *transport for source rows, difference rows, null-sequence rows, windows, tolerance ledgers, the equality seal, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from the two source rows through D, then through Z, and finally into the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyNullDifference}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNullDifference}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, D, Z, W, T, E, H, K, P, N$. It has no coordinate for quotient stream equality, host equality of reals, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyNullDifferenceUp.CauchyNullDifferenceTasteGate single carrier alignment`

Theorem 1758.2 (Cauchy null-difference NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNullDifference}^{\uparrow}$ *carrier* $C = (X, Y, D, Z, W, T, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyNullDifference}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *displayed difference row, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-shrinkage row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal, transport, continuation, provenance, and local naming rows;*
2. *source stability, requiring accepted comparisons to preserve the two regular Cauchy source rows componentwise;*
3. *difference handoff exactness, requiring D to be the displayed pointwise difference route generated from X and Y through the finite windows W;*
4. *null-sequence consumption, requiring Z to consume precisely the displayed difference row D with the tolerance ledger T before any equality seal is read;*
5. *real-seal nonescape, requiring E to read only the null-difference route $X, Y, D, Z, W, T, H, K, P, N$ and never a quotient stream equality, host real equality, selected limit, countable-choice schedule, trichotomy principle, ambient completeness theorem, or bridge certificate.*

No Cauchy null-difference obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1758.1. The accepted packet displays exactly the two source rows, the difference row, the null-sequence row, windows, tolerance ledger, equality seal, transport, continuation, provenance, and local naming rows.

The source and difference clauses are then forced by the order of the displayed route. The two $\mathsf{RegSeqRat}^{\uparrow}$ rows X, Y are read first; the $\mathsf{RegularCauchyDifference}^{\uparrow}$ row D is the only coordinate that can display their pointwise difference; and the windows W are the finite observations shared by the source and difference reads.

The null-sequence clause is the next step. The row Z consumes D together with the dyadic tolerance ledger T, so the zero-shrinkage evidence is visible before the endpoint row E is inspected.

Finally inspect the carrier inventory. Since quotient stream equality, host equality of reals, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, and bridge data are not coordinates of C, none can become local NameCert obligations. □

Theorem 1758.3 (Cauchy null-difference Real seal boundary)**.** *Let* $C = (X, Y, D, Z, W, T, E, H, K, P, N)$ *be an accepted* $\mathsf{CauchyNullDifference}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *equality-facing consumer of the two regular Cauchy source rows must first read the finite route*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow E$$

as replayed by W, T, H, K, P, N. Thus the endpoint seal is exposed only after the regular-Cauchy difference has been displayed and certified as a null sequence; it is not a quotient equality of streams, a host equality proof, a selected limit, a countable-choice construction, a trichotomy comparison, or an ambient completeness assertion.

Proof. Start with an equality-facing consumer. By Definition 1758.2, the consumer can read the source data only through the accepted carrier C.

The RegularCauchyDifference handoff is the first major step. The rows X, Y provide the regular rational observations, while D displays their aligned difference through the finite windows W and the dyadic tolerance ledger T.

The NullSequence classifier is then consumed before the endpoint is available. The row Z reads the same displayed difference route, and the transport and continuation rows H, K preserve that route under componentwise comparison and replay.

Only after this null-difference package has been displayed may the consumer reach E. The provenance and local naming rows P, N record the same finite route and add no analytic coordinate.

The non-escape statement follows from the carrier inventory. No quotient stream equality, host equality proof, selected limit, countable-choice row, trichotomy comparison, or ambient completeness theorem is present, so the $\mathsf{Real}^{\uparrow}$ equality seal is exactly the stated finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNullDifference}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNullDifference}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1758.1, Definition 1758.2, and Definition 1758.3 seed the finite route from two regular Cauchy source rows through displayed difference, NullSequence zero-shrinkage, StreamName windows, DyadicRatCore tolerance rows, and a terminal Real equality seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNullDifference}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host equality of reals, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source stability, difference handoff exactness, NullSequence consumption, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyNullDifferenceUp.

Constructive story: A $\mathsf{CauchyNullDifference}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one RegularCauchyDifference route, one NullSequence zero-shrinkage row, finite StreamName windows, DyadicRatCore tolerances, a terminal Real equality-seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1759

A Concrete Naming Certificate for $\mathsf{FareySequence}^{\uparrow}$

Following Hardy–Wright's standard treatment of Farey sequences and the Stern–Brocot rational tree, the $\mathsf{FareySequence}^{\uparrow}$ packet records finite ordered rational boundary sequences whose adjacent rows are determinant-one neighbors. It sits between the tree-facing $\mathsf{SternBrocot}^{\uparrow}$ ledger of chapter 1704, the rational sources $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ of chapter 37 and chapter 291, the finite observation route $\mathsf{StreamName}^{\uparrow}$ of chapter 126, the regular readback $\mathsf{RegSeqRat}^{\uparrow}$ of chapter 301, the approximation window $\mathsf{RealRationalApproximation}^{\uparrow}$ of chapter 1668, and the terminal real-name seal $\mathsf{Real}^{\uparrow}$ of chapter 38; the packet stores only finite rational rows, denominator bounds, dyadic tolerances, adjacency ledgers, stream windows, approximation handoffs, transport, replay, provenance, and naming.

Definition 1759.1 (Farey sequence carrier)**.** *A* $\mathsf{FareySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite ordered boundary row, listing rational endpoints as displayed* $\mathsf{Rat}^{\uparrow}$ *rows.*
2. *A is the adjacent determinant-one ledger, recording that each consecutive pair of boundary rows is a Farey-neighbor pair.*
3. *M is the mediant and refinement row, recording the displayed numerator-denominator sum inserted between adjacent boundary rows when a refinement step is requested.*
4. *L is the denominator-bound row, recording the finite Farey level or displayed denominator cap that bounds every rational row inspected by the packet.*
5. *T is the dyadic tolerance row, read by* $\mathsf{DyadicRatCore}^{\uparrow}$ *when a downstream approximation request asks for a finite error budget.*
6. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *handoff row, sending each adjacent-pair inspection to the finite tree address and Farey-neighbor ledger used by the Stern–Brocot packet.*
7. *D is the rational-density and approximation handoff row, exposing the finite sequence as a rational bracket source for L10 approximation routes.*
8. *Q is the sibling rational-source row read by* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, *and* $\mathsf{ContinuedFraction}^{\uparrow}$ *consumers when those consumers are present in the surrounding route.*

9. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which sequence entries are requested and replayed.*
10. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same ordered boundary and refinement data.*
11. *G is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff row, packaging the displayed rational bracket, denominator bound, dyadic tolerance, stream window, and regular rational readback for real-facing consumers.*
12. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the rational-source, stream-window, and regular-rational rows have been displayed.*
13. *H records componentwise* hsame *transport for boundary sequence, adjacency ledger, mediant row, denominator bound, dyadic tolerance, Stern–Brocot handoff, density handoff, rational-source row, stream window, regular-rational readback, RealRationalApproximation handoff, real seal, replay, provenance, and local naming rows.*
14. *C records displayed* Cont *replay from a finite observation request through boundary inspection, determinant-one adjacency checking, mediant refinement, denominator-bound inspection, dyadic tolerance readback, Stern–Brocot handoff, rational-source handoff, stream-window readback, regular-rational readback, RealRationalApproximation handoff, and the real seal.*
15. *P is the* Pkg *provenance over* $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
16. *N is the local* $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$. The carrier has no coordinate for host real order, quotient real equality, selected best approximants, trichotomy, countable choice, ambient completeness, or a completed real representative.

Lean4 ✓ `BEDC.Derived.FareySequenceUp.TasteGate.FareySequenceTasteGate single carrier alignment`

Theorem 1759.2 (Farey sequence NameCert obligations)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the ordered boundary sequence, adjacent determinant-one ledger, mediant refinement row, denominator-bound row, dyadic tolerance row, Stern–Brocot handoff, density handoff, rational-source row, stream-window row, regular-rational readback, RealRationalApproximation handoff, real seal, transport, replay, provenance, and naming rows;*
2. *adjacency stability, requiring A to record determinant-one adjacency for every consecutive pair before and after each displayed refinement step;*
3. *monotone finite sequence order, requiring the displayed row B to keep the finite boundary sequence in the same rational order after a mediant row is inserted;*
4. *finite level control, requiring L and T to record the displayed denominator cap and dyadic tolerance consumed by the approximation route;*
5. *mediant handoff, requiring M and S to send each refinement to the corresponding* $\mathsf{SternBrocot}^{\uparrow}$ *finite tree address and Farey-neighbor ledger;*

6. *approximation handoff, requiring G to expose the bracket, denominator-bound, dyadic-tolerance, stream-window, and regular-rational rows to* $\mathsf{RealRationalApproximation}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read;*
7. *ledger non-escape, requiring every* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer to read through Q, W, R, G, E rather than through a host real order, quotient real equality, selected best approximant, trichotomy, countable choice, or ambient completeness field.*

No Farey-sequence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceNameCertObligations`

Proof. Carrier admission is projection from Definition 1759.1. The accepted tuple exposes exactly the ordered boundary sequence, adjacency ledger, mediant refinement, denominator-bound, dyadic-tolerance, Stern–Brocot handoff, density handoff, rational-source, stream-window, regular-rational, RealRationalApproximation, real-seal, transport, replay, provenance, and local naming rows listed in the statement.

Adjacency stability is local to A. Each consecutive pair of rows in B has its determinant-one check recorded before it can be used as a Farey-neighbor pair, and each refinement inserts only the mediant row M between the inspected adjacent pair.

The order and level obligations are finite. The sequence row B names the displayed left-to-right rational boundary order, the mediant row is inserted between the two rows that generated it, and L, T record the denominator cap and dyadic tolerance without invoking a host trichotomy or quotient-real comparison.

The handoff obligation is read from S. It translates the same adjacent-pair inspection to the $\mathsf{SternBrocot}^{\uparrow}$ address and Farey-neighbor ledger, while D and Q expose the resulting finite rational data to approximation consumers.

Finally, W, R, G, E, H, C, P, N keep the real-facing route downstream of finite rational rows. Since the carrier has no coordinate for host real order, quotient real equality, selected best approximants, trichotomy, countable choice, ambient completeness, or completed representatives, the listed rows exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier namecert obligations`

Theorem 1759.3 (Farey sequence Stern–Brocot adjacency obligations)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each determinant-one adjacency read from A is routed through the $\mathsf{SternBrocot}^{\uparrow}$ *row S before any mediant refinement row M, approximation handoff row G, or terminal* $\mathsf{Real}^{\uparrow}$ *seal E can be consumed. The route is preserved by the NameCert obligations of Definition 1759.2, the approximation-window factorization of Definition 1759.6, and the mediant refinement stability of Definition 1759.9.*

Proof. Project the accepted carrier. The adjacent determinant-one ledger is A, and the only tree-facing handoff coordinate attached to that ledger is S.

Apply the local NameCert obligations of Definition 1759.2. They require each adjacent-pair inspection to be recorded in A and then sent through the $\mathsf{SternBrocot}^{\uparrow}$ handoff before the packet treats the pair as a Farey-neighbor route.

Next apply approximation-window factorization. The rows G and E are downstream of the finite ordered boundary, adjacency ledger, mediant row, denominator and tolerance rows, Stern–Brocot handoff, stream window, and regular readback rows, so an approximation consumer cannot inspect G or E before the A, S route is visible.

Finally apply mediant refinement stability. The inserted mediant is read only between the two boundary rows already inspected by A, and the same finite tree address is carried by S. Thus every mediant or real-approximation read remains behind the determinant-one Stern–Brocot adjacency obligation, with no host order, selected best approximant, quotient-real equality, trichotomy, choice, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier stern brocot adjacency obligations`

Theorem 1759.4 (Farey sequence carrier admission)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

carrier admission exposes exactly the finite ordered boundary row B, adjacent determinant-one ledger A, mediant refinement row M, denominator-bound row L, dyadic tolerance row T, Stern–Brocot handoff S, density handoff D, rational-source row Q, stream window W, regular rational readback R, RealRationalApproximation handoff G, terminal $\mathsf{Real}^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance P, and local naming row N.*

Proof. Project the accepted tuple through Definition 1759.1. The carrier definition lists the rows $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$ and no additional coordinates.

The analytic rows are read first. The ordered boundary sequence, determinant-one adjacency ledger, mediant refinement row, denominator-bound row, dyadic tolerance row, Stern–Brocot handoff, density handoff, rational-source row, stream window, regular rational readback, approximation handoff, and real seal are exactly $B, A, M, L, T, S, D, Q, W, R, G, E$.

The structural rows are H, C, P, N. They transport, replay, source, and name the same displayed packet, so carrier admission cannot expose host real order, quotient real equality, selected best approximants, trichotomy, countable choice, ambient completeness, or a completed real representative. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier admission`

Theorem 1759.5 (Farey sequence classifier transport)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

componentwise classifier transport through H, C, P, N preserves the displayed rows $B, A, M, L, T, S, D, Q, W, R, G, E$. Any transported Farey-sequence packet therefore has the same boundary-sequence classifier, adjacency classifier, mediant classifier, denominator and tolerance classifiers, rational-source classifier, stream-window classifier, regular-readback classifier, approximation classifier, and real-seal classifier, read through the same local $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *route.*

Proof. Start with the componentwise transport row H. Its coordinates compare the boundary sequence, adjacency ledger, mediant row, denominator bound, dyadic tolerance, handoff rows, stream window, regular readback, approximation row, and real seal one field at a time.

The displayed replay row C then preserves the observation route through the same fields. A request transported from one accepted packet to another still reads $B, A, M, L, T, S, D, Q, W, R, G, E$ in the same order before it reaches the local naming row.

Finally P and N package and name the transported route. Because the classifier has no extra coordinate beyond the carrier rows, transport through H, C, P, N preserves the stated componentwise classifiers without adding a selected approximant, host comparison, or ambient completion witness. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequence classifier transport`

Theorem 1759.6 (Farey sequence approximation-window factorization)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every $\mathsf{RealRationalApproximation}^{\uparrow}$ *read from* F *factors through the ordered boundary row* B*, determinant-one adjacency ledger* A*, mediant refinement row* M*, denominator-bound row* L*, dyadic tolerance row* T*, Stern–Brocot handoff* S*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, approximation handoff* G*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, with* H, C, P, N *preserving the same route under transport, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceApproximationWindowFactorization`

Proof. Project the carrier through Definition 1759.1. The approximation-facing rows are displayed in the tuple, and carrier admission from Definition 1759.4 exposes exactly those rows before any Real-facing read is available.

Apply the NameCert obligations of Definition 1759.2. They force adjacency stability through A, mediant refinement through M, finite denominator and tolerance control through L, T, and the Stern–Brocot handoff through S before the approximation handoff can be consumed.

Use classifier transport from Definition 1759.5. The rows H, C, P, N preserve the boundary, adjacency, refinement, denominator, tolerance, window, readback, approximation, and Real-seal classifiers componentwise, so the transported packet keeps the same ordered route.

Finally read the approximation side through W, R, G, E. The finite stream window supplies the displayed observations, the regular-rational row repackages the same window, the $\mathsf{RealRationalApproximation}^{\uparrow}$ row receives the bracket with its denominator and dyadic budget, and the $\mathsf{Real}^{\uparrow}$ seal is reached only after those rows. Thus no quotient Real equality, selected best approximant, trichotomy, countable choice, or ambient completeness coordinate enters the factorization. □

Theorem 1759.7 (Farey sequence denominator-bound window)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, the denominator cap* L *and dyadic tolerance* T *must be displayed before any mediant refinement row* M *or approximation consumer reads* G, E*. The route through* L, T *is a finite* BHist *budget, not a host real order, choice principle, selected best approximant, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier denominator bound window`

Proof. Read the approximation-window factorization theorem. It places L, T before W, R, G, E in every Real-facing route.

The mediant refinement stability theorem keeps M attached to the same finite denominator and tolerance rows. Thus refinement and approximation consumers receive the displayed budget before they inspect either the inserted mediant or the terminal Real-facing handoff. □

Theorem 1759.8 (Farey sequence stream-window stability)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *preserve the boundary row* B *and adjacency row* A *under classifier*

transport before the Real-facing seal E is read. The stream window therefore cannot introduce a host sequence, selected subsequence, quotient Real equality, choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use classifier transport. The rows H, C, P, N preserve the boundary, adjacency, stream-window, and regular-rational classifiers componentwise.

Then read approximation-window factorization. It routes every Real-facing approximation through W, R, G before E, so the transported stream window and regular readback remain attached to the original Farey boundary and adjacency rows. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier stream window stability`

Theorem 1759.9 (Farey sequence mediant refinement stability)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the mediant row M preserves adjacent determinant-one boundary exposure through A, S, L, T: the inserted mediant is read between the two displayed adjacent boundary rows named by B, the adjacency ledger A records the determinant-one checks for the neighboring pairs created by the insertion, the Stern–Brocot handoff S reads the same finite tree address, and L, T keep the denominator and dyadic tolerance budgets finite. The refinement exports no selected best approximant, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the adjacent pair selected from B. The determinant-one exposure for that pair is the row A, and the mediant row M is admitted only as the displayed numerator-denominator sum inserted between those two boundary rows.

After insertion, the two new neighboring pairs are still checked by A. The Stern–Brocot handoff S reads the same finite tree address route, so the refinement remains a finite Farey-neighbor exposure rather than a host-order comparison.

The rows L and T keep the denominator cap and dyadic tolerance budget attached to the refinement. Since Definition 1759.1 contains no coordinate for selected best approximants, trichotomy, countable choice, ambient completeness, or a completed real representative, mediant refinement cannot export any of those principles. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceMediantRefinementStability`

Theorem 1759.10 (Farey sequence denominator-refinement induction)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the finite refinement route induced by the denominator-bound row L is obtained by induction over the displayed level budget and the mediant row M. The base level reads the boundary row B unchanged; each successor level inspects adjacent entries through A, inserts exactly the mediant row admitted by M when L permits the denominator, and transports the resulting boundary through T, S, H, C, P, N before any D, Q, W, R, G, E consumer is available.

Proof. Use the carrier of Definition 1759.1 to expose the finite rows $B, A, M, L, T, S, H, C, P, N$. The denominator-bound window of Definition 1759.7 makes L visible before a refinement or approximation consumer can read the route.

The induction is a sibling-consuming route through chapter 1704 for adjacent mediants and through chapter 291 for the dyadic tolerance budget before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can read the bracket.

The base level is the displayed boundary row B. No adjacent insertion is required, so carrier admission and the NameCert obligations of Definition 1759.2 supply the whole packet at that level.

For the successor level, inspect one adjacent pair through A. The mediant refinement stability theorem Definition 1759.9 identifies M as the inserted numerator-denominator row, while denominator and tolerance control from Definition 1759.21 keeps the insertion under L, T.

Apply the induction hypothesis to the remaining displayed boundary. Approximation-window factorization from Definition 1759.6 keeps D, Q, W, R, G, E downstream of the finite route, so the induction never imports host order, selected approximants, quotient Real equality, trichotomy, choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceDenominatorRefinementInduction`
Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier denominator refinement induction`

Theorem 1759.11 (Farey sequence finite mediant coverage)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, each boundary inserted by a displayed mediant refinement remains routed through* A, S, L, T*. Consequently every inserted rational boundary is covered by the same determinant-one adjacency ledger, Stern–Brocot handoff, denominator cap, and dyadic tolerance rows before it can be read by* D, Q, W, R, G, E*.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceFiniteMediantCoverage`

Proof. Start with an inserted boundary row. By mediant refinement stability, the row is produced from an adjacent pair already named by B and checked by A.

The finite tree-facing route is supplied by S. The Stern–Brocot handoff lock of Definition 1759.14 keeps every successor or approximation read that consumes M behind A, L, T, S.

The budget rows are L, T. Denominator and tolerance control says that no second host bound or selected modulus can enter any read of M, S, D, Q, W, R, G, E.

Finally apply approximation-window factorization. The downstream rational, stream, regular-readback, approximation, and Real-seal rows remain after A, S, L, T, so each inserted boundary has finite mediant coverage through the displayed packet rows. □

Theorem 1759.12 (Farey sequence Real-bracket consumer lock)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

any Real-facing bracket consumer must read $B, A, M, L, T, S, Q, W, R, G, E$ *before the terminal seal is consumed. In particular, no* $\mathsf{RealRationalApproximation}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can bypass the displayed boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, rational-source, stream-window, regular-readback, approximation, and seal rows.*

Proof. Begin with the approximation-window factorization theorem. It places each Real-facing read behind the ordered boundary, determinant-one adjacency ledger, mediant row, denominator and tolerance rows, Stern–Brocot handoff, stream window, regular readback, approximation handoff, and terminal seal.

Use finite mediant coverage above for inserted boundaries. Every boundary created by refinement remains under A, S, L, T, so the bracket read cannot detach a refined endpoint from its finite denominator and tolerance budget.

The rational-source and readback rows are then Q, W, R, G, E. The Real readback root obligation of Definition 1759.15 and the approximation ledger non-escape theorem of Definition 1759.26 keep those rows inside the same displayed Farey packet.

Thus every Real-bracket consumer reads $B, A, M, L, T, S, Q, W, R, G, E$ before the seal. The carrier contains no quotient Real equality, selected best approximant, host order, trichotomy, countable choice, or ambient completeness coordinate that could supply an alternate bracket route. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier real bracket consumer lock`

Theorem 1759.13 (Farey sequence finite-density root package)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each finite-density read exported to an L10 approximation or completion consumer factors through one displayed root package: the boundary row B, adjacency ledger A, mediant row M, denominator and tolerance rows L, T, $\mathsf{SternBrocot}^{\uparrow}$ *handoff S,* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{RealRationalApproximation}^{\uparrow}$ *row G, and terminal* $\mathsf{Real}^{\uparrow}$ *seal E, with H, C, P, N preserving the same route. The package exports no selected best approximant, host order, quotient real equality, trichotomy, countable choice, or ambient completeness principle.*

Proof. Start with the Farey carrier and NameCert obligations. The carrier of Definition 1759.1 and the obligations of Definition 1759.2 expose exactly the finite boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, rational-source, stream, readback, approximation, Real-seal, transport, replay, provenance, and naming rows.

The density route is then locked to finite denominator data. The denominator-bound window theorem Definition 1759.7 and the denominator refinement induction of Definition 1759.10 place each inspected density window behind B, A, M, L, T before any real-facing row can be read.

Next apply the Stern–Brocot handoff lock of Definition 1759.14. It keeps each mediant or successor-level read attached to A, L, T, S rather than to a host-order branch choice.

Finally use the approximation-window factorization theorem Definition 1759.6. It sends the resulting finite bracket through W, R, G, E, while H, C, P, N transport, replay, source, and name the same packet. Hence every finite-density read factors through the displayed root package and cannot export any of the refused ambient objects. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceFiniteDensityRootPackage`

Theorem 1759.14 (Farey sequence Stern–Brocot handoff lock)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every denominator-level successor or approximation read that consumes a mediant row M must pass through the adjacent determinant-one ledger A, the denominator cap L, the dyadic tolerance T, and the $\mathsf{SternBrocot}^{\uparrow}$ *handoff S before the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *row G or terminal* $\mathsf{Real}^{\uparrow}$ *row E is read. The locked route exports no selected best approximant, host order, quotient real equality, trichotomy, countable choice, or ambient completeness principle.*

Proof. Start with a successor or approximation read that consumes M. Carrier admission exposes M only as the mediant row for an adjacent pair already displayed in B, so the determinant-one ledger A is the first required witness for that read.

Apply mediant refinement stability from Definition 1759.9. The inserted row remains tied to the same adjacent-pair inspection, and the Stern–Brocot handoff S records the corresponding finite tree route rather than a host-order branch choice.

Next use denominator and tolerance control from Definition 1759.21. The denominator cap L and dyadic tolerance T are the only budget rows available to the mediant and approximation route, so no second host bound or selected modulus can enter before the read reaches G or E.

Finally apply approximation-window factorization from Definition 1759.6. The real-facing rows G, E are downstream of A, M, L, T, S, W, R and the structural rows H, C, P, N. Since the carrier has no coordinate for selected best approximants, host order, quotient real equality, trichotomy, countable choice, or ambient completeness, the handoff is locked to the finite Stern–Brocot route. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceSternBrocotHandoffLock`

Theorem 1759.15 (Farey sequence Real readback root obligation)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-facing Farey read factors through the finite route*

$$B, A, M, L, T, S, Q, W, R, G, H, C, P, N \longmapsto E.$$

Thus the ordered boundary, adjacent determinant-one ledger, mediant row, denominator cap, dyadic tolerance, Stern–Brocot handoff, rational-source row, StreamName window, RegSeqRat readback, approximation handoff, transport, replay, provenance, and local naming rows are all visible before the terminal Real seal is consumed. The route supplies no selected best approximant, host order, quotient real equality, trichotomy, countable choice, or ambient completeness principle.

Proof. Start from a Real-facing Farey read. Carrier admission exposes the only Real seal as E, and the approximation-window factorization theorem places E after the finite boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, window, readback, and approximation rows.

The rational and stream stages are read before the seal. The rational-source row Q, StreamName window W, RegSeqRat readback R, and approximation handoff G give the displayed route to the terminal Real boundary.

Finally transport, replay, provenance, and local naming are carried by H, C, P, N. Since no row of the carrier stores a selected approximant, host order, quotient Real equality, trichotomy, choice schedule, or ambient completeness datum, the listed route exhausts the Real-facing readback obligation. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRealReadbackRootObligation`

Theorem 1759.16 (Farey sequence RegSeqRat density root obligation)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every $\mathsf{RegSeqRat}^{\uparrow}$ *consumer and rational-density consumer receives only the displayed boundary row* B*, adjacent determinant-one ledger* A*, mediant row* M*, denominator cap* L*, dyadic tolerance row* T*, density handoff* D*, rational-source row* Q*, StreamName window* W*, and regular readback row* R*, with* H, C, P, N *preserving that same finite route. No such consumer receives a host order, selected best approximant, quotient real equality, trichotomy, countable choice, or ambient completeness coordinate.*

Proof. Read the density side of the carrier. The finite rational data available before any regular readback are B, A, M, L, T, D, Q, namely the displayed boundary, adjacency, mediant, denominator, tolerance, density, and rational-source rows.

The window and regular readback are then W, R. The StreamName row fixes the finite observation window, and the RegSeqRat row replays that window as a regular rational readback without adding a second rational-density witness.

Transport and replay preserve the same finite route through H, C, P, N. Because the carrier has no coordinate for host order, selected approximants, quotient real equality, trichotomy, choice, or ambient completeness, the RegSeqRat and rational-density consumers can receive only the displayed rows listed in the statement. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRegSeqRatDensityRootObligation`

Theorem 1759.17 (Farey sequence level-successor induction)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the successor from one displayed denominator level to the next is generated by induction on the finite boundary row B. The base boundary is admitted unchanged, and each successor step reads one adjacent pair through A, inserts the permitted mediant through M when the denominator cap in L admits it, and carries the resulting row through T, S, H, C, P, N. The construction uses no parameter-transport equivalence, selected best approximant, host order, quotient real equality, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceLevelSuccessorInduction`

Proof. Induct on the displayed boundary row B. For the empty and singleton boundaries there is no adjacent pair to inspect, so carrier admission supplies the whole next-level row and no mediant insertion is performed.

For the successor step, expose the first adjacent pair through the determinant-one ledger A. The mediant row M supplies exactly the numerator-denominator sum for that pair, while the denominator cap L decides whether that finite row belongs to the next displayed level.

Apply the induction hypothesis to the remaining boundary suffix. The tolerance row T and Stern–Brocot handoff S keep the inserted row on the same finite route, and H, C, P, N transport, replay, source, and name the resulting level.

Since each step consumes only $B, A, M, L, T, S, H, C, P, N$, the construction is an induction over the displayed finite row. The excluded host-order, quotient, trichotomy, choice, selected-approximant, and ambient-completeness data never become available. □

Theorem 1759.18 (Farey sequence mediant density window)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each density window opened by a $\mathsf{RegSeqRat}^{\uparrow}$ *or rational approximation consumer is refined by finite mediant insertion through A, M, L, T, S before it is handed to D, Q, W, R, G, E. In particular, a requested window between adjacent displayed rationals is covered by the boundary row already present in B or by the inserted mediant row admitted by M under the denominator cap L and tolerance row T.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceMediantDensityWindow`

Proof. Begin with a density-window request. Carrier admission exposes the boundary row B and the adjacent determinant-one ledger A before any rational-source, stream, or regular-readback row can be consumed.

If the requested window is already covered by an adjacent boundary endpoint, the route stays inside B, A and proceeds to the density handoff D. No mediant is inserted in this case.

If the window lies strictly between an adjacent pair, use the level-successor induction of Definition 1759.17. The induction step inserts the unique displayed mediant row through M and keeps it under the denominator and tolerance rows L, T.

The Stern–Brocot row S records the same adjacent-pair route, after which D, Q, W, R, G, E read the density, rational-source, stream-window, RegSeqRat, approximation, and Real-seal data. Thus the window is refined by finite mediant insertion rather than by trichotomy, selected approximants, host order, quotient equality, choice, or ambient completeness. □

Theorem 1759.19 (Farey sequence RegSeqRat density induction)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *density route is obtained by induction over the displayed denominator-level successor rows. At each level the finite boundary row is refined by* A, M, L, T, S*, the density handoff is read through* D, Q, W*, and the regular rational readback* R *is produced before* G, E *are available. The route preserves the no-trichotomy, no-choice, no-host-order, no-quotient-equality, and no-ambient-completeness exclusions.*

Proof. Induct on the displayed denominator-level successor row supplied by Definition 1759.17. The base level supplies only the boundary data already admitted by the carrier, so the RegSeqRat density read is the finite readback through D, Q, W, R for that boundary.

For the successor level, apply the mediant density window theorem Definition 1759.18. Each new gap is either already covered by the boundary or refined by the mediant row generated through A, M, L, T, S.

After the finite refinement step, read D, Q, W as the density, rational-source, and StreamName window rows. The regular rational row R packages exactly that windowed data, and the approximation and Real rows G, E remain downstream.

The structural rows H, C, P, N preserve the same induction route under transport, replay, provenance, and local naming. Since the carrier contains no coordinate for trichotomy, countable choice, host order, quotient real equality, selected best approximants, or ambient completeness, the RegSeqRat density derivation is internal to the displayed Farey packet. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRegSeqRatDensityInduction`

Theorem 1759.20 (Farey sequence root-unblock package)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the root-facing Farey package is the conjunction of carrier admission, the local NameCert obligations, approximation-window factorization, the Stern–Brocot handoff lock, the Real readback root obligation, and the RegSeqRat density root obligation. Consequently downstream $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers read the same finite Farey packet through* $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$*, with no selected best approximant, host order, quotient real equality, trichotomy, countable choice, or ambient completeness principle.*

Proof. Begin with carrier admission and the local NameCert obligations. These expose exactly the boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, density, rational-source, stream-window, readback, approximation, Real-seal, transport, replay, provenance, and naming rows.

Next combine approximation-window factorization with the Stern–Brocot handoff lock. The first routes the windowed approximation read through W, R, G, E, while the second keeps mediant and denominator-level reads on the finite A, M, L, T, S route.

Apply the two root obligations above. The Real readback obligation keeps terminal Real consumers behind the same finite route, and the RegSeqRat density obligation restricts rational-density consumers to the displayed boundary, mediant, tolerance, stream-window, and readback rows.

The structural rows H, C, P, N preserve that package under transport, replay, provenance, and local naming. Since the accepted carrier stores none of the excluded host-order, selection, quotient, choice, or completeness data, the package is a finite root-facing unblock certificate rather than an analytic completion principle. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier root unblock package`

Theorem 1759.21 (Farey sequence denominator and tolerance control)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the denominator-bound row L and the dyadic-tolerance row T are the only finite denominator and tolerance rows consumed by the mediant refinement and approximation route. In particular, every read of M, S, D, Q, W, R, G, E that depends on a denominator cap or tolerance budget factors through L, T and through no additional host bound, selected modulus, quotient-real comparison, or ambient completeness coordinate.

Proof. Project the carrier of Definition 1759.1. The displayed packet contains exactly one denominator-bound coordinate L and exactly one dyadic-tolerance coordinate T.

The mediant row M reads denominator data only through the displayed boundary and adjacent-pair inspection together with L. The Stern–Brocot and rational-source rows S, D, Q inherit the same finite cap, so they cannot introduce a second denominator witness.

The approximation side is then forced through W, R, G, E. The finite stream window and regular rational readback use the tolerance row T before the $\mathsf{RealRationalApproximation}^{\uparrow}$ handoff G and the terminal $\mathsf{Real}^{\uparrow}$ seal E are available.

The componentwise transport and replay rows H, C preserve the same two coordinates. Since the carrier has no coordinate for a host bound, selected modulus, quotient-real comparison, or ambient completeness principle, L, T exhaust the denominator and tolerance control consumed by the route. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceDenominatorToleranceControl`

Theorem 1759.22 (Farey sequence level successor)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the denominator-bound row L and mediant row M produce the next finite Farey level by induction on the displayed boundary row B. Each successor step inspects one adjacent pair recorded by

A, inserts exactly the mediant row admitted by M when the denominator cap in L permits it, and transports the resulting boundary through S, H, C, P, N without invoking selected best approximants, host order, quotient real equality, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceLevelSuccessor`

Proof. Induct on the finite boundary row B. The empty and singleton boundaries have no adjacent pair, so the next level is the displayed row already admitted by carrier admission and no mediant insertion is available.

For the inductive step, expose the first adjacent pair through A. The mediant refinement stability theorem Definition 1759.9 identifies the only permitted insertion as the numerator-denominator sum stored in M, and the denominator and tolerance control theorem Definition 1759.21 keeps that insertion under the cap recorded by L.

Apply the induction hypothesis to the remaining boundary suffix. The Stern–Brocot handoff S records the same adjacent-pair route, while H, C, P, N transport, replay, source, and name the resulting finite level. Since every step reads only $B, A, M, L, S, H, C, P, N$, no excluded order, quotient, selection, choice, or completeness object enters the successor construction. □

Theorem 1759.23 (Farey sequence adjacency density)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each requested adjacent gap in the displayed boundary row is refined through A, M, S and then handed to the density and approximation rows D, Q, W, R, G, E. The refinement is a finite adjacent-pair inspection and not a selected best approximant, host real-order comparison, quotient-real equality, trichotomy, countable-choice selector, or ambient completeness principle.

Proof. Begin with a requested adjacent gap. Carrier admission and the NameCert obligations identify the gap by its two boundary entries in B and by the determinant-one ledger entry in A.

Apply mediant refinement stability. The row M supplies the displayed mediant for that adjacent pair, and the handoff row S records the same finite Stern–Brocot route without selecting a branch outside the packet.

Finally read the density and approximation surface. The rows D, Q expose the finite rational source, W, R expose the stream window and regular-rational readback, and G, E expose the RealRationalApproximation handoff and terminal Real seal. The classifier transport theorem keeps this route under H, C, P, N, so the density claim remains finite and cannot import a best-approximant selector, trichotomy, choice, quotient equality, host order, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier adjacency density`

Theorem 1759.24 (Farey sequence refinement coverage)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every $\mathsf{RealRationalApproximation}^{\uparrow}$ *consumer reads each Farey refinement through the finite route $B, A, M, L, T, S, W, R, G, E$ together with transport, replay, provenance, and local naming through H, C, P, N. Thus refinement coverage is exhaustive over the displayed packet and contains no quotient real equality, selected approximant, trichotomy, countable choice, host sequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from an accepted approximation-window request. The approximation-window factorization theorem routes the request through the ordered boundary B, adjacency ledger A, mediant row M, denominator and tolerance rows L, T, Stern–Brocot handoff S, stream window W, regular readback R, approximation handoff G, and Real seal E.

Use adjacency density for the local refinement step. It shows that each requested adjacent gap is refined by the finite A, M, S route before D, Q, W, R, G, E are consumed by rational-window or Real-facing readers.

The real approximation ledger non-escape theorem then confines the consumer to the same displayed packet. Transport through H, replay through C, provenance through P, and local naming through N preserve the coverage route, so every refinement visible to a $\mathsf{RealRationalApproximation}^{\uparrow}$ consumer is accounted for by $B, A, M, L, T, S, W, R, G, E, H, C, P, N$ and by no excluded ambient object. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRefinementCoverage`

Theorem 1759.25 (Farey sequence stream-window denominator lock)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every stream-window read W consumed by R, G, E carries the denominator-bound row L and the dyadic-tolerance row T from the same accepted carrier. Consequently, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *consumers cannot replace the displayed Farey denominator cap by a host denominator bound, an external modulus, a quotient-real comparison, or an ambient completeness coordinate.*

Proof. Project the accepted carrier through Definition 1759.1. The rows L, T, W, R, G, E are coordinates of the same finite packet, so a window read cannot be detached from the denominator and tolerance rows that classify the packet.

Apply the denominator and tolerance control of Definition 1759.21. It identifies L, T as the only finite bound and tolerance rows available to any read of W, R, G, E that depends on such data.

Finally use the approximation ledger non-escape theorem Definition 1759.26. Since real-facing reads remain inside $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$, no downstream consumer can substitute a host cap, external modulus, quotient-real comparison, or ambient completeness witness for the displayed rows L, T. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier stream window denominator lock`

Theorem 1759.26 (Farey sequence approximation ledger non-escape)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each $\mathsf{RealRationalApproximation}^{\uparrow}$ *read and each terminal* $\mathsf{Real}^{\uparrow}$ *read factors through the finite ledger $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$. The exported surface contains no quotient real equality, no selected best approximant, no trichotomy, no countable-choice selector, and no ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier real approximation ledger nonescape`

Proof. Begin with a real-facing read of the accepted carrier. By the handoff clause in Definition 1759.2, the read must pass through the rational bracket, denominator-bound row, dyadic-tolerance row, stream window, regular rational readback, and $\mathsf{RealRationalApproximation}^{\uparrow}$ handoff before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

The finite rational data are exactly B, A, M, L, T, S, D, Q. These rows display the boundary sequence, determinant-one adjacency ledger, mediant refinement, denominator cap, tolerance budget, Stern–Brocot handoff, density handoff, and sibling rational-source handoff.

The real-facing route is exactly W, R, G, E. The finite stream window reads entries, the regular rational readback packages the sequence row, the approximation handoff exposes the displayed bracket and budget, and the real seal records only the downstream endpoint allowed by the preceding rows.

Transport, replay, provenance, and naming are supplied by H, C, P, N. They preserve the same finite inventory and do not add a coordinate whose classifier could select a best approximant or compare quotient real names.

Therefore every $\mathsf{RealRationalApproximation}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read factors through the listed packet coordinates. The absence of host real order, quotient real equality, selected best approximants, trichotomy, countable choice, ambient completeness, and completed representatives in Definition 1759.1 gives the stated non-escape boundary. □

Theorem 1759.27 (Farey sequence RealRationalApproximation handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing approximation read of an accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FareySequence}^{\uparrow},\ \mathsf{SternBrocot}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{RationalStream}^{\uparrow},\ \mathsf{ContinuedFraction}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeq}$$

with the optional siblings omitted only when the surrounding packet does not expose them. The route uses the ordered boundary sequence, adjacent determinant-one ledger, denominator-bound row, dyadic tolerance row, mediant refinement row, Stern–Brocot handoff, rational-source row, stream window, regular-rational readback, RealRationalApproximation handoff, and real seal. It supplies no host real order, quotient real equality, selected best approximant, trichotomy, countable choice, ambient completeness, or independent completed-real representative.

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRealApproximationHandoff`

Proof. Start with an accepted carrier $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$. The Farey-facing data are exactly the finite boundary sequence B, the determinant-one ledger A, the mediant refinement row M, the denominator-bound row L, and the dyadic tolerance row T.

The Stern–Brocot handoff is the row S. It reads each adjacent pair and refinement through the same finite tree address and Farey-neighbor ledger used by chapter 1704.

The rational-source handoff is then carried by D and Q. Dyadic, rational, rational-stream, and continued-fraction consumers can inspect the displayed finite rational rows together with the tolerance and denominator-bound rows, but they receive no additional real-order field.

The approximation route is carried by W, R, G, E. A finite $\mathsf{StreamName}^{\uparrow}$ window requests entries from the sequence, $\mathsf{RegSeqRat}^{\uparrow}$ reads the resulting regular rational row, $\mathsf{RealRationalApproximation}^{\uparrow}$ receives the bracket with its dyadic tolerance and denominator bound, and $\mathsf{Real}^{\uparrow}$ receives only the downstream seal row.

Componentwise transport and replay preserve the same factorization for classified packets. The provenance and local naming rows name the same finite inventory, so no quotient real equality, selected limit, ambient completeness, or independent representative can enter the route. □

Theorem 1759.28 (Farey sequence regular-readback approximation bridge)**.** *For every accepted* FareySequence$^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the RegSeqRat$^{\uparrow}$ *readback row* R *and the* RealRationalApproximation$^{\uparrow}$ *handoff row* G *are forced through the same ordered boundary row* B*, adjacency ledger* A*, denominator-bound row* L*, dyadic-tolerance row* T*, and stream-window row* W*. The bridge exports a regular rational readback and a real-approximation handoff only along this common finite boundary, not through a host sequence, selected approximant, quotient-real equality, countable choice, or ambient completion.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRegularReadbackApproximationBridge`

Proof. Start from the carrier of Definition 1759.1. The regular readback R and approximation handoff G are not independent rows; they are read after the same boundary, adjacency, denominator, tolerance, and stream-window rows B, A, L, T, W are displayed.

Apply the approximation-window factorization of Definition 1759.6. It routes each RealRationalApproximation$^{\uparrow}$ read through the ordered boundary, determinant-one adjacency ledger, denominator and tolerance rows, stream window, regular readback, and approximation handoff before the terminal real seal is available.

Use the RealRationalApproximation handoff theorem Definition 1759.27. The handoff keeps R and G downstream of the same finite Farey packet, so the bridge cannot introduce a host sequence, selected approximant, quotient-real equality, countable-choice selector, or ambient completion coordinate. □

Theorem 1759.29 (Farey sequence continued-fraction window)**.** *Every* ContinuedFraction$^{\uparrow}$ *consumer of an accepted* FareySequence$^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$$

must read the rational-source row Q*, the finite* StreamName$^{\uparrow}$ *window* W*, the* RegSeqRat$^{\uparrow}$ *readback* R*, the denominator-bound row* L*, and the dyadic tolerance row* T *before any continued-fraction comparison is exported. The comparison route therefore exposes only displayed finite rational-source and window rows, with no selected tail, host real order, quotient real equality, trichotomy, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the accepted carrier through Definition 1759.1. A continued-fraction consumer can find its rational-source row only at Q, and that row is already tied to the displayed boundary, mediant, denominator, and tolerance data.

The next visible access is the finite window row W. It requests only displayed entries of the Farey packet, so a comparison cannot bypass the packet through a host rational sequence.

The regular readback row R packages the same windowed data before any continued-fraction comparison leaves the packet. The rows L and T keep the denominator cap and dyadic tolerance attached to that read.

Transport, replay, provenance, and naming through H, C, P, N preserve this route. Since the carrier has no selected continued-fraction tail, host real order, quotient equality, trichotomy, choice selector, or ambient completeness coordinate, every exported continued-fraction comparison is confined to Q, W, R, L, T and the displayed support rows. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceContinuedFractionWindow`

Theorem 1759.30 (Farey sequence Stern–Brocot mediant density route)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the mediant refinement row M and the Stern–Brocot handoff row S jointly preserve determinant-one adjacency through A and keep the denominator and tolerance data attached through L, T. The resulting density route is finite and exports no selected best approximant, branch selector, host real order, quotient real equality, trichotomy, choice selector, or ambient completeness coordinate.

Proof. Begin with an adjacent pair from the ordered boundary row B. The determinant-one check for that pair is the ledger row A, and the mediant row M inserts only the displayed numerator-denominator sum between those two boundary entries.

The Stern–Brocot handoff S reads the same adjacent-pair inspection. It therefore sends the refinement to the finite tree-facing ledger without selecting an independent branch or importing a host-order comparison.

The denominator and tolerance side is locked by L, T. Every density read that uses the mediant or the Stern–Brocot handoff keeps the displayed denominator cap and dyadic error budget attached to the same finite refinement step.

The downstream rows $D, Q, W, R, G, E, H, C, P, N$ can only consume the resulting finite route. Because none of the carrier rows stores a best-approximant selector, branch selector, host real order, quotient equality, trichotomy, choice, or completeness witness, the mediant density route remains the finite one stated above. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceSternBrocotMediantDensityRoute`

Theorem 1759.31 (Farey sequence Real approximation density route)**.** *Every* $\mathsf{RealRationalApproximation}^{\uparrow}$ *density read of an accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier factors through the full finite inventory*

$$B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N.$$

In particular, the read first uses the ordered boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, and density rows B, A, M, L, T, S, D, then the rational-source, window, regular-readback, approximation, and Real-seal rows Q, W, R, G, E, and finally transport, replay, provenance, and naming through H, C, P, N. It has no host order, quotient Real equality, trichotomy, choice selector, selected best approximant, or ambient completeness coordinate.

Proof. Start with a density read aimed at $\mathsf{RealRationalApproximation}^{\uparrow}$. The Farey-side part of the carrier is B, A, M, L, T, S, D: the boundary row, adjacency ledger, mediant refinement, denominator cap, tolerance budget, Stern–Brocot handoff, and density handoff.

The rational-source and windowed-readback part is Q, W, R. These rows ensure that the density read reaches rational and regular rational consumers through displayed finite windows rather than through a host rational or real sequence.

The real-facing endpoint is then G, E. The handoff row G supplies the $\mathsf{RealRationalApproximation}^{\uparrow}$ packet, and E exposes only the terminal $\mathsf{Real}^{\uparrow}$ seal reached from the preceding finite rows.

Finally H, C, P, N transport, replay, source, and name the same inventory. Since Definition 1759.1 contains no coordinate for host order, quotient Real equality, trichotomy, choice, selected best approximants, or ambient completeness, every Real approximation density read factors through exactly the displayed finite rows. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier real approximation density route`

Theorem 1759.32 (Farey sequence stream-window adjacency scope)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the $\mathsf{StreamName}^{\uparrow}$ *window* W *reads only the displayed ordered boundary row* B *and the adjacency ledger* A *when it exports a finite Farey window. In particular, every windowed entry request is classified by the displayed rational boundary row, every neighboring-pair request is classified by the determinant-one rows in* A*, and no host real order, quotient real comparison, selected branch, or ambient completion datum is available to the window classifier.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceStreamWindowAdjacencyScope`

Proof. Project the accepted packet to B, A, W, H, C, P, N. The row W is a finite stream window whose entries are requested from the displayed boundary sequence B, so the entry classifier is already bounded by the ordered rational rows of B.

Neighboring-pair reads then use A. Each adjacent request in the window points to the determinant-one ledger row for the displayed consecutive pair, not to a host-order comparison or to an external real representative.

The rows H, C, P, N transport, replay, package, and name the same finite data. Since the carrier has no coordinate for host real order, quotient real comparison, selected branches, or ambient completion, the stream-window classifier cannot escape B, A, W, H, C, P, N. □

Theorem 1759.33 (Farey sequence rational-source handoff scope)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RationalStream}^{\uparrow}$*, and* $\mathsf{ContinuedFraction}^{\uparrow}$ *consumer factors through the rational-source row* Q*, the stream-window row* W*, the regular-rational readback* R*, the approximation handoff* G*, and the real seal* E *whenever those sibling consumers are present. The handoff exposes only finite rational rows, dyadic tolerance data, windowed sequence requests, regular readback, and the named approximation route.*

Proof. Start with a sibling rational consumer. The accepted carrier exposes rational-source data through Q, which packages the displayed rational boundary, dyadic tolerance, and denominator information already recorded by B, L, T.

The consumer then reads finite sequence access through W and regular rational replay through R. These rows keep the request inside the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces before any real-facing approximation row is visible.

The rows G and E provide the only approximation and real-seal endpoints. Thus $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, and $\mathsf{ContinuedFraction}^{\uparrow}$ consumers factor through Q, W, R, G, E and receive no independent host rational carrier, selected continued-fraction tail, quotient real comparison, or ambient completion witness. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRationalSourceHandoffScope`

Theorem 1759.34 (Farey sequence local naming replay scope)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the componentwise transport H*, displayed replay* C*, package provenance* P*, and local naming row* N *preserve the same finite packet* $B, A, M, L, T, S, D, Q, W, R, G, E$*. They do not add host real order,*

quotient real equality, selected approximants, trichotomy, countable choice, ambient completeness, or a completed real representative to the Farey sequence route.

Proof. Project the accepted carrier to the transport and replay rows. The row H compares packets componentwise across the displayed boundary, adjacency, mediant, bound, tolerance, handoff, window, readback, approximation, and real-seal rows.

The row C replays observation requests through the same components. It has no separate source from which a host order, quotient equality, or selected approximant could be read.

Finally, P and N package and name the already displayed packet. Because the listed carrier coordinates exhaust the local certificate, transport, replay, provenance, and naming preserve the finite Farey packet without adding trichotomy, countable choice, ambient completeness, or a completed real representative. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceLocalNamingReplayScope`

Theorem 1759.35 (Farey sequence mediant window coverage)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each displayed denominator-bounded mediant window is covered by a finite adjacent-row inspection of B, A, M, L*. The coverage row names the two adjacent rational boundary entries, the inserted mediant, and the level cap before any* $\mathsf{RealRationalApproximation}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may read the window. It exports no host real order, quotient real equality, selected best approximant, trichotomy, countable choice, or ambient completeness coordinate.*

Proof. Project the accepted carrier to the boundary, adjacency, mediant, and level rows. The ordered row B supplies the finite list of rational boundary entries, A supplies the determinant-one neighboring checks, M supplies the displayed numerator-denominator sum, and L supplies the finite denominator cap.

For an inspected adjacent pair, the only admitted refinement is the mediant row placed between the two named endpoints. Thus the displayed window is bounded by the same adjacent-row inspection that created it, and every denominator-bounded request is accounted for by a finite segment of B, A, M, L.

The downstream rows $T, S, D, Q, W, R, G, E, H, C, P, N$ preserve that finite coverage route. Since none of those rows contains a host-order comparison, quotient equality, selected best approximant, trichotomy, choice selector, or completeness witness, the mediant window cannot escape the displayed adjacent-row coverage. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceMediantWindowCoverage`

Theorem 1759.36 (Farey sequence stream approximation totality)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every accepted approximation request that reaches the $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff first routes through the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*. The handoff then reads* G *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from the same displayed packet; it does not choose a best approximant or identify quotient real names.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceStreamApproximationTotality`

Proof. Start with an accepted approximation request. The finite rational data available to it are the boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, density, and rational-source rows B, A, M, L, T, S, D, Q.

The request must next pass through the window row W. That row supplies only finite $\mathsf{StreamName}^{\uparrow}$ access to displayed sequence entries, so the approximation cannot bypass the stream window to a host rational sequence.

After the window is displayed, the readback row R packages the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ surface. Only then can G expose the $\mathsf{RealRationalApproximation}^{\uparrow}$ handoff and E expose the terminal $\mathsf{Real}^{\uparrow}$ seal.

Because the route is $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N$ and the carrier has no selected-best-approximant or quotient-real coordinate, the accepted approximation request is total only along this finite stream and regular-rational route. □

Theorem 1759.37 (Farey sequence adjacent dyadic window)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

an adjacent Farey window inspected through B, A, M, L *gives a dyadic tolerance route only through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *rows in* Q *together with the tolerance row* T*. The route may feed* $\mathsf{RationalStream}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *consumers through* W*, but it cannot introduce a host real-order comparison, selected best approximant, or quotient-real equality witness.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceAdjacentDyadicWindow`

Proof. Start with an adjacent-pair inspection. The finite boundary row B names the rational endpoints, the adjacency ledger A records determinant-one neighboring, the mediant row M supplies the displayed refinement, and L bounds the denominators.

The dyadic tolerance read is then forced through T and Q. These rows are the only coordinates that expose $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ data to downstream rational consumers.

The stream-facing part of the same request uses W for the $\mathsf{StreamName}^{\uparrow}$ window and may expose $\mathsf{RationalStream}^{\uparrow}$ through the rational-source handoff. Since no host-order, selected-approximant, or quotient-equality coordinate appears in Definition 1759.1, the adjacent window remains a finite dyadic-tolerance route. □

Theorem 1759.38 (Farey sequence rational stream readback)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the $\mathsf{RationalStream}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback of an adjacent Farey window preserves the same classifier rows* B, A, M, L, T, Q, W*. When the readback is repackaged as a* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, it still records the same finite window classifier and does not add a new rational sequence, choice selector, or ambient completion coordinate.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRationalStreamReadback`

Proof. Read the stream-facing route from Q, W. The rational-source row Q exposes only the displayed rational data already controlled by B, A, M, L, T, and the finite stream window W requests entries from that displayed sequence.

The regular rational readback row R repackages the same window for $\mathsf{RegSeqRat}^{\uparrow}$. It is downstream of Q, W, so it cannot replace the classifier by a hidden rational stream or by a selected subsequence.

Transport and replay through H, C preserve the same rows before provenance and local naming are read. Thus the classifier remains B, A, M, L, T, Q, W, R throughout the rational-stream readback. □

Theorem 1759.39 (Farey sequence real approximation boundary)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the Real-facing approximation boundary is the route from adjacent Farey rows through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and then* G, E *for* $\mathsf{RealRationalApproximation}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$. *The boundary exports finite approximation data only; it cannot export quotient equality of real names, a selected limit, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the adjacent window and dyadic tolerance route of Definition 1759.37. It confines the approximation request to B, A, M, L, T, Q, W before the regular readback is available.

Apply the readback scope of Definition 1759.38. The row R gives the $\mathsf{RegSeqRat}^{\uparrow}$ surface for the same displayed window, preserving the rational and stream classifiers.

Only after those rows are displayed can G expose the $\mathsf{RealRationalApproximation}^{\uparrow}$ handoff and E expose the terminal $\mathsf{Real}^{\uparrow}$ seal. The carrier contains no quotient-real equality, selected-limit, trichotomy, choice, or ambient-completeness coordinate, so the real-facing boundary is finite approximation data and nothing stronger. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRealApproximationBoundary`

Theorem 1759.40 (Farey sequence Stern–Brocot refinement scope)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the Stern–Brocot-facing scope of the packet is exactly the refinement handoff through S *from an adjacent-row inspection of* B, A, M, L. *No other row exports a tree address, branch selector, or refinement path to a tree-facing consumer.*

Proof. Read the tree-facing data from the carrier. The only row named as a Stern–Brocot handoff is S, and its input is the same finite adjacent-pair inspection supplied by B, A, M, L.

The rational-source and approximation rows D, Q, W, R, G, E are downstream of that handoff. They can consume the finite rational and window data, but they do not contain a separate tree address or branch-selection coordinate.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the carrier stores no hidden branch selector or independent refinement path, every tree-facing use of $\mathsf{FareySequence}^{\uparrow}$ is scoped by the single Stern–Brocot handoff S. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceSternBrocotRefinementScope`

Theorem 1759.41 (Farey sequence neighbor determinacy)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

an adjacent determinant-one inspection in A *determines a unique displayed Stern–Brocot refinement route inside the carrier: the boundary rows are read from* B, *the mediant insertion is read from* M, *the tree-facing address is read from* S, *and every transported or replayed use of that route remains under* H, C, P, N. *The determinacy is internal to the displayed rows and does not assert quotient-real equality, a selected best approximant, trichotomy, choice, or an ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier neighbor determinacy`

Proof. Start with the finite adjacent-pair data. Theorems 1759.4, 1759.2, and 1759.9 identify B, A, M as the boundary, determinant-one, and mediant rows for a displayed refinement.

The tree-facing route is then forced by S. Theorems 1759.30 and 1759.40 say that the Stern–Brocot consumer reads only the handoff row attached to that adjacent-pair inspection, with no separate branch selector or refinement path.

Finally transport and replay through H, C, P, N preserve the same route by Theorems 1759.5 and 1759.34. Since the carrier has no coordinate for quotient-real equality, selected approximants, trichotomy, choice, or ambient completeness, the displayed Stern–Brocot route is unique at the level of the packet rows. □

Theorem 1759.42 (Farey sequence denominator ledger exactness)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the denominator bounds and dyadic tolerances consumed by any $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff are read from the same finite ledger rows* L, T *before the handoff reaches* W, R, G, E*. No consumer may replace* L, T *by a host denominator bound, selected modulus, quotient-real comparison, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier denominator ledger`

Proof. Read the ledger rows first. Theorems 1759.21 and 1759.37 identify L, T as the displayed denominator and dyadic-tolerance source for adjacent Farey windows.

Next pass to stream and regular-rational consumers. Theorems 1759.32, 1759.38, and 1759.36 route W and R through the same adjacent-window data rather than through an independent bound.

The real-facing handoff is downstream of that route. By Theorems 1759.6, 1759.39, and 1759.26, G, E receive only the displayed ledger, window, and readback rows, so no host bound, selected modulus, quotient comparison, or ambient completeness coordinate can enter. □

Theorem 1759.43 (Farey sequence dyadic mediant bracket)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

each adjacent determinant-one packet displayed by B, A *determines a dyadic bracket row whose mediant refinement remains inside the displayed rational stream route* Q, W, R*. The same denominator and tolerance ledger* L, T *is the budget consumed by* $\mathsf{RealRationalApproximation}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *through* G, R*, with no selected best approximant, trichotomy, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier dyadic mediant bracket`

Proof. Read the adjacent Farey packet first. Theorems 1759.2, 1759.9, and 1759.42 place the boundary row, determinant-one ledger, mediant row, denominator cap, and dyadic tolerance in B, A, M, L, T.

Next pass to the rational stream route. Theorems 1759.38 and 1759.44 keep the displayed bracket inside Q, W, R, so the mediant row cannot be replaced by a hidden stream element or selected best approximant.

Finally hand the same budget to real-approximation consumers. Theorems 1759.39 and 1759.26 route G, E from the displayed denominator and tolerance rows, while the carrier has no coordinate for trichotomy, quotient real equality, countable choice, or ambient completeness. □

Theorem 1759.44 (Farey sequence rational coverage)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

every rational-window approximation consumer factors through the rows already present in the carrier: $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *are read through* Q, T*,* $\mathsf{RationalStream}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *through* Q, W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *through* R*,* $\mathsf{RealRationalApproximation}^{\uparrow}$ *through* G*, and* $\mathsf{Real}^{\uparrow}$ *through* E*, with* $B, A, M, L, S, H, C, P, N$ *preserving the finite Farey-neighbor context. The coverage statement adds no quotient-real equality, selected best approximant, trichotomy, choice, or Real-completeness claim.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceRationalCoverage`

Proof. Begin with the rational-source rows. Theorems 1759.33, 1759.37, and 1759.35 place $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalStream}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ reads inside Q, T, W over the same finite boundary and adjacency rows.

The regular and approximation rows are then read in order. Theorems 1759.38, 1759.27, and 1759.39 route R, G, E from the rational window to $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The context rows $B, A, M, L, S, H, C, P, N$ are preserved by carrier admission, classifier transport, and local naming replay. Thus every rational-window approximation consumer factors through the displayed Farey packet rows, and the excluded quotient, selection, trichotomy, choice, and completeness objects remain absent from the carrier. □

Theorem 1759.45 (Farey sequence denominator-monotone coverage)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, the displayed denominator row* L *covers each boundary entry of* B *monotonically along the finite row order before any* W, R, G, E *approximation consumer is read. The coverage is a* BHist *and* $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *fact over* $B, A, M, L, T, H, C, P, N$*, transported only by the displayed* hsame *rows and replayed only through the displayed* Cont *rows.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceDenominatorMonotoneCoverage`

Proof. Induct over the displayed boundary row B. The first and last rows are read from carrier admission, while each adjacent step is checked by the determinant-one row A and the denominator budget row L.

At a mediant insertion, the row M supplies the inserted numerator-denominator data and T keeps the dyadic tolerance request attached to the same budget. Componentwise transport through H and replay through C preserve that finite budget row, so downstream W, R, G, E reads cannot introduce a larger hidden denominator source. □

Theorem 1759.46 (Farey sequence Stern–Brocot cofinal bracket)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, every displayed Stern–Brocot bracket consumed through* S *is cofinal with a finite adjacent bracket already present in* B, A, M, L, T*. The cofinality certificate is the carried* BHist *route through* S, H, C, P, N*, not a host order-completeness or infinite-branch selector.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceSternBrocotCofinalBracket`

Proof. Induct over the finite Stern–Brocot handoff row S. The base bracket is one of the adjacent pairs in B checked by A, with its denominator budget read from L.

For a refinement step, the mediant row M inserts the next bracket inside the previous adjacent pair and the tolerance row T records the requested depth. Since H, C, P, N transport, replay, source, and name only those rows, the bracket exported to a consumer is cofinal inside the displayed finite Farey context. □

Theorem 1759.47 (Farey sequence real rational-approximation unblock)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier* $F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N)$*, a* $\mathsf{RealRationalApproximation}^{\uparrow}$ *consumer may read an approximation route only after the finite coverage rows* $B, A, M, L, T, S, D, Q, W, R, G$ *have been displayed. The terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is therefore reached through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *rows, with no selected best approximant, quotient real equality, trichotomy, choice, or ambient completeness coordinate.*

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier real rational approximation unblock`

Proof. Start with the rational-source rows. The boundary row B, adjacency row A, mediant row M, denominator row L, tolerance row T, Stern–Brocot row S, density row D, and rational source row Q form the finite rational bracket inventory.

The approximation route then passes through W, R, G in order: stream window, regular readback, and real-rational handoff. Transport through H, replay through C, and provenance through P, N keep the terminal seal E attached to that inventory, so a Real or Completion consumer receives a finite rational-approximation route and no hidden selector. □

Theorem 1759.48 (Farey sequence adjacency obligation)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the determinant-one adjacency obligation of $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *is bound exactly by the rows* A *and* S*. The row* A *records the adjacent determinant-one checks for consecutive entries of* B*, and the row* S *carries the same adjacent-pair inspection to the* $\mathsf{SternBrocot}^{\uparrow}$ *handoff before any* W, R, G, E *approximation read is available.*

Proof. Carrier admission from Definition 1759.4 exposes B, A, S as displayed rows of the same accepted packet. The classifier transport theorem Definition 1759.5 keeps these rows componentwise fixed under H, C, P, N.

The local adjacency content is supplied by A. The adjacent dyadic window theorem Definition 1759.37 starts each approximation window with the displayed boundary row and the determinant-one ledger before tolerance or stream data can be consumed.

The tree-facing continuation is supplied by S. The Stern–Brocot refinement scope theorem Definition 1759.40 and the neighbor determinacy theorem Definition 1759.41 route the same adjacent inspection through the Stern–Brocot handoff without adding a branch selector, host order, quotient real equality, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier adjacency obligation`

Theorem 1759.49 (Farey sequence refinement obligation)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the mediant refinement obligation of $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *is bound exactly by the rows* M, L, T *together with the adjacency and Stern–Brocot rows* A, S*. The inserted mediant is read from* M*, its finite denominator and tolerance budgets are read from* L, T*, and its refinement route remains attached to the adjacent-pair handoff through* A, S*.*

Proof. Start with the mediant row exposed by carrier admission. The theorem Definition 1759.9 identifies M as the displayed numerator-denominator sum inserted between the two boundary rows named by B and checked by A.

The finite budget is supplied by L, T. The denominator and tolerance control theorem Definition 1759.21 says that every mediant, Stern–Brocot, rational-source, stream, readback, and approximation read that depends on such data factors through those two rows.

The refinement handoff is then routed through S. Theorems 1759.14, 1759.30, and 1759.24 keep the inserted row on the finite A, M, L, T, S route before any downstream approximation consumer can read W, R, G, E. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier refinement obligation`

Theorem 1759.50 (Farey sequence approximation obligation)**.** *For every accepted* $\mathsf{FareySequence}^{\uparrow}$ *carrier*

$$F = (B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P, N),$$

the $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff obligation of* $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$ *is bound exactly by the rows* W, R, G, E *over the finite Farey context* B, A, M, L, T, S, D, Q*. A real-facing approximation read must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the handoff row* G*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Proof. Begin with the windowed approximation route. The approximation-window factorization theorem Definition 1759.6 routes every $\mathsf{RealRationalApproximation}^{\uparrow}$ read through the ordered boundary, adjacency, mediant, denominator, tolerance, Stern–Brocot, stream, readback, handoff, and real-seal rows.

The stream and rational readback stages are finite. The adjacent dyadic window theorem Definition 1759.37 supplies Q, T, W from the same adjacent Farey window, and the rational stream readback theorem Definition 1759.38 repackages that window as the row R without adding a new rational sequence or choice selector.

The real-facing boundary is then G, E. The theorem Definition 1759.39 and the real approximation ledger non-escape theorem Definition 1759.26 keep the handoff inside $B, A, M, L, T, S, D, Q, W, R, G, E, H, C, P$ with no quotient real equality, selected limit, trichotomy, countable choice, host sequence, or ambient completeness coordinate. □

Lean4 ✓ `BEDC.Derived.FareySequenceUp.FareySequenceCarrier approximation obligation`

Closure status for $\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FareySequence}^{\uparrow}$ packet is generated from a finite ordered rational boundary sequence, an adjacent determinant-one ledger, a mediant refinement row, a denominator-bound row, a dyadic tolerance row, a Stern–Brocot handoff, rational-source and density handoffs, a finite StreamName window, a RegSeqRat readback, a RealRationalApproximation handoff, a RealUp seal, componentwise transports, displayed continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1759.44, Definition 1759.46, and Definition 1759.47 bind the bridge scope through finite Farey approximation, Stern–Brocot refinement, RegSeqRat density readback, and the Real-rational approximation route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FareySequence}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not close quotient real equality, selected best approximants, trichotomy, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: No theory-axis upgrade remains; bridge checking and Lean verification are separate obligations.

Chapter 1760

A Concrete Naming Certificate for $\mathsf{AdmissibleRepresentation}^{\uparrow}$

Following Weihrauch and the represented-spaces tradition in computable analysis, the $\mathsf{AdmissibleRepresentation}^{\uparrow}$ packet records the finite finality interface by which Cauchy names become sufficient for represented-space consumers. It sits over the represented-space surface of chapter 1494, the Cauchy-representation handoff of chapter 1680, the finite stream windows of chapter 126, the regular rational readback boundary of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet expresses admissibility as a local NameCert surface: a consumer may use the displayed Cauchy route, but receives no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 1760.1 (AdmissibleRepresentation carrier)**.** *An* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, K, W, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose consumers read only the finite represented-space interface.*
2. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row supplying the admissible Cauchy route for the same represented-space observations.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened by both X and K.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the displayed stream windows become regular rational observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare those observations at the displayed precisions.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, W, R, D have been replayed.*
7. *F is the continuous-consumer finality row: every represented-space consumer displayed in the packet factors through the same K, W, R, D, E route before reading its output.*
8. *H records componentwise* hsame *transport for the represented-space row, Cauchy source row, stream-window schedule, regular readback row, dyadic tolerance ledger, real seal, finality row, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from represented-space access through the Cauchy source, finite windows, regular readback, dyadic tolerance comparison, real seal, and consumer-finality handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AdmissibleRepresentation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite admissible-representation packet. Its classifier compares accepted carriers only after transporting the displayed rows to a common finite surface, and it contains no coordinate for quotient streams, selected limits, host real equality, countable choice, metric completeness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.AdmissibleRepresentationUp.AdmissibleRepresentationTasteGate single carrier alignment`

Theorem 1760.2 (AdmissibleRepresentation NameCert obligations)**.** *For every accepted* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *carrier $A = (X, K, W, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AdmissibleRepresentation}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the represented-space source, Cauchy source, finite stream windows, regular readback, dyadic tolerance ledger, real seal, consumer-finality row, transport, replay, provenance, and local naming rows;*

2. *represented-space compatibility, requiring X to expose only the represented-space interface already admitted by the packet;*

3. *Cauchy handoff, requiring K, W, R, D, E to be the only displayed route from names to the real-facing seal;*

4. *consumer finality, requiring every displayed represented-space consumer to factor through K, W, R, D, E before reading output data;*

5. *transport, replay, and provenance, requiring H, C, P to preserve and source exactly the displayed packet;*

6. *local naming, requiring N to certify precisely those rows and no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

No admissible-representation obligation is stored outside these rows.

Proof. Project the accepted carrier. Carrier admission exposes $X, K, W, R, D, E, F, H, C, P, N$ and no further row.

Represented-space compatibility is the X projection. It is read only as the represented-space source already present in the packet, so it cannot introduce a second representation relation or host-level equality.

The Cauchy handoff is the ordered route K, W, R, D, E. The source row supplies the Cauchy representation, the stream schedule supplies the finite observations, the regular readback exposes rational data, the dyadic ledger compares tolerances, and the real seal is reached only at the end of that replay.

Consumer finality is the F projection. Any displayed represented-space consumer must pass through the same handoff route before it reads output data, so its computation has no independent access to a quotient stream, selected limit, or completeness principle.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same packet, and therefore add no extra analytic coordinate. The listed rows exhaust the local NameCert surface. □

Theorem 1760.3 (AdmissibleRepresentation real-Cauchy handoff)**.** *Let $A = (X, K, W, R, D, E, F, H, C, P, N)$ be an accepted* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing represented-space consumer displayed by A factors through the route*

$$X \longmapsto K \longmapsto W \longmapsto R \longmapsto D \longmapsto E \longmapsto F, H, C, P, N.$$

Thus the consumer receives the displayed $\mathsf{CauchyRepresentation}^{\uparrow}$ *route and no quotient stream, selected limit, host real equality, countable-choice row, metric-completeness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with a $\mathsf{Real}^{\uparrow}$-facing represented-space consumer of A. The represented-space source is X, but the only real-facing access row attached to X inside the packet is the Cauchy source K.

Replay the Cauchy route. The consumer opens the finite $\mathsf{StreamName}^{\uparrow}$ schedule W, reads the $\mathsf{RegSeqRat}^{\uparrow}$ row R, consults the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and reaches the $\mathsf{Real}^{\uparrow}$ seal E only after those rows have been consumed.

Now apply finality. The row F states that every displayed consumer factors through this same route, while H, C, P, N merely transport, replay, source, and name it. Since the carrier has no coordinate for quotient streams, selected limits, host real equality, countable choice, metric completeness, or ambient real completeness, none of those data can be delivered to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{AdmissibleRepresentation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AdmissibleRepresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1760.1, Definition 1760.2, and Definition 1760.3 seed the finite represented-space finality interface from $\mathsf{CauchyRepresentation}^{\uparrow}$ names to $\mathsf{Real}^{\uparrow}$-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AdmissibleRepresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representations, selected limits, host real equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, represented-space compatibility, Cauchy handoff, consumer finality, transport, replay, provenance, and local naming for BEDC.Derived.AdmissibleRepresentationUp.

Constructive story: An $\mathsf{AdmissibleRepresentation}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal, a consumer-finality row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1761

A Concrete Naming Certificate for DyadicErrorBudget$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicErrorBudget$^{\uparrow}$ packet records the finite epsilon-budget split used when two visible dyadic approximation errors must be joined before any quotient stream equality, selected limit, or ambient Real$^{\uparrow}$ completeness principle is available. It sits on the dyadic arithmetic core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the tail-radius ledger of chapter 757. The packet gives RealCompletionExactBoundaryUp and RealCauchyCompletion$^{\uparrow}$ consumers a named finite cross-route triangle-budget carrier instead of forcing each consumer to restate the same dyadic slack accounting.

Definition 1761.1 (Dyadic error-budget carrier)**.** *A* DyadicErrorBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, T, H, C, P, N)$$

with the following displayed rows:

1. *Q is the requested dyadic tolerance for the downstream comparison.*
2. *A and B are the two displayed sub-budget rows whose dyadic sum is required to fit inside Q.*
3. *S is the split certificate saying that the two sub-budgets are the only local epsilon pieces consumed by the packet.*
4. *W_X and W_Y are the two finite* StreamName$^{\uparrow}$ *windows opened for the compared approximation routes.*
5. *D_X and D_Y are the two* DyadicRatCore$^{\uparrow}$ *error rows read from the visible windows.*
6. *D_Z is the* DyadicRatCore$^{\uparrow}$ *triangle row recording the displayed dyadic sum of the two error rows against Q.*
7. *R_X and R_Y are the* RegSeqRat$^{\uparrow}$ *readback rows attached to the two windows.*
8. *T is the* DyadicTailRadiusLedger$^{\uparrow}$ *handoff row that records the same request and split inside the tail-radius accounting route.*
9. *H records componentwise* hsame *transport for the request, split, windows, dyadic rows, readbacks, tail-radius handoff, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from the request through the split, the two windows, the two dyadic error rows, the triangle row, the regular readbacks, and the tail-radius handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{DyadicErrorBudget}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicTailRadiusLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicErrorBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, T, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicErrorBudgetUp.DyadicErrorBudgetUpTasteGate single carrier alignment`

Theorem 1761.2 (Dyadic error-budget triangle accounting)**.** *For every accepted* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *carrier*

$$E = (Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, T, H, C, P, N),$$

any downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBoundaryUp, *or* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *consumer that uses the packet's error bound must read the finite route*

$$Q, A, B, S \longmapsto W_X, W_Y \longmapsto D_X, D_Y, D_Z \longmapsto R_X, R_Y, T, H, C, P, N.$$

Thus the requested tolerance Q is split only through A, B, S, the two visible approximation errors are read only through the finite windows and dyadic rows W_X, W_Y, D_X, D_Y, and the exported triangle budget is exactly the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *row D_Z together with the tail-radius handoff T.*

Lean4 ✓ `BEDC.Derived.DyadicErrorBudgetUp.DyadicErrorBudgetCarrier triangle accounting`

Proof. Project the carrier of Definition 1761.1. The request row, the two sub-budget rows, and the split row are the only coordinates that can supply an epsilon split.

Read the two approximation routes through the displayed windows. The rows W_X, W_Y open the finite StreamName observations, D_X, D_Y record the two dyadic error contributions, and R_X, R_Y attach the corresponding RegSeqRat readbacks.

The triangle step is the row D_Z. It reads the same request and the same two sub-budgets, so a consumer obtains a bound only by passing through the displayed dyadic sum row and not through a hidden stream quotient or selected limit.

Finally the tail-radius handoff T and the structural rows H, C, P, N preserve the same finite route. Since no other coordinate is present, every listed downstream consumer receives exactly the displayed triangle-budget packet. □

Theorem 1761.3 (Dyadic error-budget NameCert obligations)**.** *For every accepted* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *carrier $E = (Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicErrorBudget}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the request, two sub-budgets, split certificate, two StreamName windows, three DyadicRatCore rows, two RegSeqRat readbacks, tail-radius handoff, transport, replay, provenance, and local naming rows;*

2. *request exactness, requiring Q to be the downstream tolerance consumed by the packet;*

3. *split exhaustion, requiring A and B to be the only local epsilon pieces and S to bind them to Q;*

4. *window exactness, requiring W_X and W_Y to be the finite windows from which the two error rows are read;*

5. *dyadic triangle exactness, requiring D_Z to record the displayed dyadic sum of D_X and D_Y against Q;*
6. *regular readback and tail-radius handoff, requiring R_X, R_Y, T to consume the same windows, dyadic rows, request, and split;*
7. *transport and replay stability, requiring H and C to preserve the same finite route;*
8. *provenance and local naming, requiring P and N to package exactly the displayed rows $Q, A, B, S, W_X, W_Y, D_X,$*
9. *refusal of quotient streams, selected limits, host equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local NameCert obligations.*

No dyadic error-budget obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1761.1. The carrier lists the full request, split, window, dyadic, readback, handoff, transport, replay, provenance, and local naming inventory.

Request exactness and split exhaustion follow from the rows Q, A, B, S. The packet gives the downstream tolerance one split certificate, and the carrier has no third sub-budget coordinate.

Window exactness and dyadic triangle exactness follow from W_X, W_Y, D_X, D_Y, D_Z. The two error rows are visible only through their finite windows, and the exported triangle bound is the displayed dyadic row D_Z.

The rows R_X, R_Y, T, H, C, P, N then carry the same route to regular readback, tail-radius accounting, transport, replay, provenance, and naming. Since quotient streams, selected limits, host equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge data are absent from the carrier, they cannot become local obligations. □

Theorem 1761.4 (Dyadic error-budget ladder composition)**.** *Composing two accepted* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *packets whose downstream and upstream tolerance rows match yields a single displayed ladder route. The exported tolerance is still read through the two split certificates, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *triangle rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoffs, tail-radius rows, and structural support of the original packets; no quotient stream, selected limit, host equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate is introduced.*

Proof. Project the two carriers of Definition 1761.1. Each carrier supplies its own request row, two sub-budget rows, split certificate, finite windows, dyadic error rows, triangle row, regular readbacks, tail-radius handoff, transport, replay, provenance, and local name.

Identify the shared tolerance request where the downstream row of the first packet is the upstream row of the second. The composition has no new request source; it reads that shared row through the two displayed split certificates.

Apply triangle accounting from Definition 1761.2 to each packet. The first packet exports its bound through its finite windows, dyadic rows, regular readbacks, and tail-radius handoff, and the second packet reads the matching tolerance through the same kind of displayed route.

Concatenate the displayed rows and use the NameCert obligations of Definition 1761.3 for both packets. The resulting ladder route is sourced and named by the two original structural supports, and the excluded host, quotient, choice, order, and completeness principles remain absent. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicErrorBudget}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicErrorBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1761.1, Definition 1761.2, and Definition 1761.3 seed the finite epsilon-split packet over a dyadic request, two displayed sub-budgets, two StreamName windows, three DyadicRatCore rows, two RegSeqRat readbacks, tail-radius handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{DyadicErrorBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exactness, split exhaustion, window exactness, dyadic triangle exactness, regular readback, tail-radius handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicErrorBudgetUp.

Constructive story: A DyadicErrorBudget$^{\uparrow}$ packet is generated from a downstream dyadic request, two displayed sub-budgets, a split certificate, two finite StreamName windows, two dyadic error rows, one dyadic triangle row, two RegSeqRat readbacks, a DyadicTailRadiusLedger handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1762

A Concrete Naming Certificate for BishopModulusProductClosure$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopModulusProductClosure$^{\uparrow}$ packet records the explicit-modulus product closure route for two constructive real names before the result is sealed as a Real-facing product. It binds the source-modulus surface of chapter 1753, the dyadic split of chapter 1761, the regular-Cauchy multiplication route of chapter 1702, the Cauchy product sibling of chapter 464, and the terminal Real seal of chapter 38. The packet is a finite certificate that product reads consume displayed moduli, displayed dyadic budgets, and displayed regular-Cauchy product rows rather than quotient stream equality, selected limits, host real multiplication, countable choice, trichotomy, or ambient completeness.

Definition 1762.1 (Bishop modulus product-closure carrier)**.** *A* BishopModulusProductClosure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, U, V, E, H, C, P, N)$$

with the following displayed rows:

1. *M_X and M_Y are the two* BishopModulusReal$^{\uparrow}$ *source routes. Each row exposes one explicit Bishop modulus, its StreamName window discipline, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.*
2. *Q is the requested dyadic tolerance for the product read.*
3. *A and B are the two sub-budget rows assigned to the two source-modulus error contributions.*
4. *S is the* DyadicErrorBudget$^{\uparrow}$ *split row certifying that A and B exhaust the local error split for Q.*
5. *W_X and W_Y are the finite* StreamName$^{\uparrow}$ *factor-window rows opened through the source moduli M_X and M_Y.*
6. *D_X and D_Y are the two* DyadicRatCore$^{\uparrow}$ *factor-error rows read from the displayed windows.*
7. *D_Z is the* DyadicRatCore$^{\uparrow}$ *product-error row that combines D_X, D_Y under the split Q, A, B, S.*
8. *R_X and R_Y are the two* RegSeqRat$^{\uparrow}$ *source readbacks exposed by the source-modulus rows before multiplication.*
9. *U is the* RegularCauchyMultiplication$^{\uparrow}$ *row consuming $R_X, R_Y, W_X, W_Y, D_X, D_Y, D_Z$ under the displayed budget split.*

10. *V is the* $\mathsf{CauchyProduct}^{\uparrow}$ *sibling row that presents the same product read as the Cauchy-product NameCert surface.*

11. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *product seal reached only after $M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, U, V$ have been displayed.*

12. *H records componentwise* hsame *transport for the two source moduli, the error-budget split, windows, dyadic rows, regular readbacks, product rows, Real seal, replay, provenance, and local naming.*

13. *C records displayed* Cont *replay from each Real-facing product request through the same finite route.*

14. *P is the* Pkg *provenance over* $\mathsf{BishopModulusProductClosure}^{\uparrow}$, $\mathsf{BishopModulusReal}^{\uparrow}$, $\mathsf{DyadicErrorBudget}^{\uparrow}$, $\mathsf{RegularCauchyMultiplication}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

15. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopModulusProductClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, U, V, E,$ It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, host real multiplication, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1762.2 (Bishop modulus product-closure NameCert obligations)**.** *For every accepted* $\mathsf{BishopModulusProductClosure}^{\uparrow}$ *carrier*

$$Z = (M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, U, V, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopModulusProductClosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{BishopModulusReal}^{\uparrow}$ *source routes, requested tolerance, two sub-budgets, split certificate, StreamName windows, DyadicRatCore factor and product rows, RegSeqRat source readbacks, RegularCauchyMultiplication row, CauchyProduct row, Real seal, transport, replay, provenance, and local naming;*

2. *source-modulus exactness, requiring M_X and M_Y to be read before any factor window, dyadic factor row, regular readback, multiplication row, CauchyProduct row, or Real product seal is exposed;*

3. *error-budget split exhaustion, requiring Q, A, B, S to be the only local product-error budget route and to factor through the* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *split surface;*

4. *product-row compatibility, requiring $W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y$ to be the common rows consumed by both U and V;*

5. *Real-seal nonescape, requiring E to remain terminal with respect to the displayed source-modulus, budget, window, dyadic, regular-readback, and product rows;*

6. *transport and replay stability, requiring H and C to preserve the same finite route under* hsame *transport and* Cont *replay;*

7. *provenance and local naming, requiring P and N to package exactly the displayed rows and no additional quotient, selected-limit, choice, trichotomy, host-equality, host-multiplication, completeness, Lean, or bridge data.*

No Bishop modulus product-closure obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1762.1. The packet displays the two source-modulus rows, the product tolerance and split, the two factor windows, the dyadic factor and product ledgers, the regular readbacks, the two product rows, the Real seal, and the structural rows.

Source-modulus exactness follows by reading chapter 1753 twice. The rows M_X and M_Y expose the two Bishop modulus routes before $W_X, W_Y, D_X, D_Y, R_X, R_Y$ can be consumed.

The error-budget clause is the sibling route through chapter 1761. The request Q, sub-budgets A, B, and split row S are the only budget coordinates in the carrier, so the split is exhausted by those displayed rows.

Product-row compatibility reads chapter 1702 and chapter 464 over the same source data. Both product rows consume the displayed StreamName windows, DyadicRatCore rows, and RegSeqRat readbacks, so neither row can introduce a second product source.

Finally apply the Real seal discipline from chapter 38. The row E is terminal and downstream of the two source-modulus routes, the budget split, the finite windows, the dyadic rows, the regular readbacks, and the product rows, while H, C, P, N transport, replay, package, and name exactly that route. □

Lean4 ✓ `BEDC.Derived.BishopModulusProductClosureUp.BishopModulusProductClosureTasteGate single carrier alignment`

Theorem 1762.3 (Bishop modulus product-closure Real product seal)**.** *For every accepted* BishopModulusProductC *carrier*

$$Z = (M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D_Y, D_Z, R_X, R_Y, U, V, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-facing product read factors through the finite route*

$$M_X, M_Y \longmapsto Q, A, B, S \longmapsto W_X, W_Y \longmapsto D_X, D_Y, D_Z \longmapsto R_X, R_Y \longmapsto U, V \longmapsto E, H, C, P, N.$$

Consequently the read factors through $\mathsf{BishopModulusReal}^{\uparrow}$*,* $\mathsf{DyadicErrorBudget}^{\uparrow}$*,* $\mathsf{RegularCauchyMultiplication}^{\uparrow}$*,* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before the Real product seal is available.*

Proof. Begin with a Real-facing product read of Z. By Definition 1762.1, the only Real-facing coordinate is the terminal row E.

The source part of the route is forced by the two $\mathsf{BishopModulusReal}^{\uparrow}$ rows of chapter 1753. A consumer reaches factor windows, dyadic factor rows, and regular readbacks only after M_X and M_Y have displayed the corresponding modulus routes.

The product-budget part is forced by chapter 1761. The rows Q, A, B, S expose the finite budget split before D_Z can be used as the product-error row.

The multiplication part is forced by chapter 1702 and chapter 464. The rows U and V consume the same windows, dyadic rows, and RegSeqRat readbacks, and neither row precedes the source-modulus or error-budget routes.

The final seal is the Real boundary of chapter 38. Since E is downstream of $M_X, M_Y, Q, A, B, S, W_X, W_Y, D_X, D$ every Real-facing product read factors through the route in the statement and through the structural rows H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopModulusProductClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopModulusProductClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1762.1, Definition 1762.2, and Definition 1762.3 seed the finite explicit-modulus product route over two BishopModulusReal source rows, a DyadicErrorBudget split, RegularCauchyMultiplication, CauchyProduct, and a Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopModulusProductClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-modulus exactness, error-budget split exhaustion, product-row compatibility, Real-seal nonescape, transport and replay stability, provenance, and local naming for BEDC.Derived.BishopModulusProductClosureUp.

Constructive story: A $\mathsf{BishopModulusProductClosure}^{\uparrow}$ packet is generated from two BishopModulusReal source routes, one dyadic product request, two displayed sub-budgets, a DyadicErrorBudget split, two StreamName factor windows, DyadicRatCore factor and product ledgers, RegSeqRat readbacks, RegularCauchyMultiplication and CauchyProduct rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1763

A Concrete Naming Certificate for $\mathsf{AbelSummation}^{\uparrow}$

Following Abel and Dirichlet summation by parts in Bishop–Bridges constructive analysis, the $\mathsf{AbelSummation}^{\uparrow}$ packet isolates the finite identity consumed by the Abel-test route of chapter 1496 and the Dirichlet-test route of chapter 1497. Its source rows sit over the finite series windows of chapter 121; its regular handoff and dyadic estimates use chapter 301 and chapter 291; and its seal-facing boundary is the $\mathsf{Real}^{\uparrow}$ route of chapter 38. The certificate records a displayed finite transform and the handoff rows that downstream convergence tests may read; it does not choose an infinite sum or appeal to host completeness.

Definition 1763.1 (AbelSummation carrier)**.** *An* $\mathsf{AbelSummation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, A, D, B, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite partial-sum row, displaying the inspected source values and their cumulative sums on the admitted interval.*
2. *A is the coefficient-window row, displaying the scalar coefficients read at the same finite endpoints.*
3. *D is the finite-difference row, displaying the adjacent coefficient differences used by the summation-by-parts calculation.*
4. *B is the boundary-term row, displaying the two endpoint products and the finite interior products determined by S, A, D.*
5. *T is the tail-estimate row, assigning a requested dyadic tolerance to a finite source window, coefficient window, and difference window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed boundary and tail rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to that regular rational handoff.*
8. *H records componentwise* hsame *transport for partial sums, coefficient windows, finite differences, boundary terms, tail estimates, regular handoff, real seal, continuation replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a precision request through T to the rows S, A, D, B, then through R to the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelSummation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven displayed rows. It has no coordinate for a selected infinite sum, quotient series equality, host real completeness, countable choice, trichotomy, open-cover compactness, or an ambient summation operator.

Lean4 (definition) `BEDC.Derived.AbelSummationUp.AbelSummationUp`
Lean4 ✓ `BEDC.Derived.AbelSummationUp.AbelSummationTasteGate single carrier alignment`

Theorem 1763.2 (AbelSummation NameCert obligations)**.** *For every accepted* $\mathsf{AbelSummation}^{\uparrow}$ *carrier* $Q = (S, A, D, B, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AbelSummation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *partial-sum exposure for S, preserving the inspected source values and cumulative sums along componentwise* hsame *transport;*
3. *coefficient-window exposure for A, preserving the endpoint coefficients read by the finite identity;*
4. *finite-difference exposure for D, preserving the adjacent coefficient differences used by the interior products;*
5. *boundary exactness for B, stating that the displayed endpoint and interior terms are the finite summation-by-parts row determined by S, A, D;*
6. *tail-handoff exactness for T, R, E, forcing every completion-facing consumer through the displayed boundary row and regular rational handoff;*
7. *transport stability, continuation replay, provenance, and local naming for exactly the route recorded by H, C, P, N.*

These are obligations over displayed finite rows, not obligations over a host summation operator or a selected completed series.

Proof. Project the accepted packet $Q = (S, A, D, B, T, R, E, H, C, P, N)$. The projection exposes exactly the partial-sum row, coefficient-window row, finite-difference row, boundary-term row, tail-estimate row, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

Read the finite identity through S, A, D, B. The row S supplies cumulative sums on the inspected interval, A supplies the endpoint coefficients, and D supplies the adjacent coefficient differences. The boundary row B records the endpoint products and interior products obtained from those rows.

Now read a precision request through T. The schedule names only finite windows, so the handoff row R receives a regular rational estimate whose data are already displayed in S, A, D, B, T.

Finally apply the structural rows. Componentwise transport in H preserves the analytic rows together, while C, P, N replay, source, and name that same route. Since the carrier has no further coordinate, the seven listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{AbelSummation}^{\uparrow}}$ surface. □

Theorem 1763.3 (AbelSummation partial-sum handoff)**.** *Every accepted* $\mathsf{AbelSummation}^{\uparrow}$ *carrier sends a displayed coefficient-weighted finite sum to a completion-facing consumer only through*

$$S, A, D, B, T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may read the summation-by-parts calculation only after reading the finite partial sums, coefficient windows, adjacent differences, boundary terms, and tail estimate that determine the handoff.*

Proof. Begin with an accepted carrier and a completion-facing read. By the carrier definition, the only rows visible to such a read are the regular handoff row R, the seal row E, and the structural rows H, C, P, N.

Trace R backward through the packet. Its input is the tail-estimate row T, and the estimate row is defined over the displayed finite rows S, A, D, B.

The partial sums, coefficient values, finite differences, and boundary terms are therefore read before any regular rational or real seal row can be consumed. No separate coordinate supplies a host sum, a quotient series equality, or a choice-selected limit.

Transport and replay preserve this route componentwise. Hence every completion-facing read factors through the displayed chain and cannot bypass the finite summation-by-parts packet. □

Lean4 ✓ `BEDC.Derived.AbelSummationUp.AbelSummationPartialSumHandoff`

Theorem 1763.4 (AbelSummation Dirichlet-test route)**.** *Any* $\mathsf{DirichletTest}^{\uparrow}$ *or* $\mathsf{AbelTest}^{\uparrow}$ *consumer that uses a summation-by-parts row from an accepted* $\mathsf{AbelSummation}^{\uparrow}$ *carrier factors through the same finite identity rows* S, A, D, B, T*. The consumer may combine those rows with its own bounded-partial-sum or monotone-null ledger, but it cannot import selected infinite sums, quotient series equality, host real completeness, countable choice, trichotomy, or open-cover compactness from the* $\mathsf{AbelSummation}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.AbelSummationUp.AbelSummationDirichletTestRoute`

Proof. Project the shared summation-by-parts input used by the convergence-test consumer. The $\mathsf{AbelSummation}^{\uparrow}$ carrier exports only the rows S, A, D, B, T to an analytic handoff, followed by R, E, H, C, P, N for regular readback, seal, transport, replay, provenance, and local naming.

For an $\mathsf{AbelTest}^{\uparrow}$ read, the downstream packet may add its monotone null coefficient ledger and uniform-limit row. For a $\mathsf{DirichletTest}^{\uparrow}$ read, the downstream packet may add its bounded partial-sum and null-tail schedule rows. In both cases the summation-by-parts input itself remains the displayed finite row B determined by S, A, D.

Because the source carrier contains no prohibited coordinate, neither consumer can receive such data through hsame transport or Cont replay. The route is exactly the finite BEDC identity plus the consumer's own convergence-test rows. □

Falsifiable prediction: An accepted $\mathsf{AbelSummation}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{AbelTest}^{\uparrow}$ row, or $\mathsf{DirichletTest}^{\uparrow}$ row without reading the partial-sum row, coefficient-window row, finite-difference row, boundary-term row, and tail-estimate row refutes the carrier surface.

Independence witness: series, abel-test, dirichlet-test, regseqrat, dyadicratcore, real: no listed sibling supplies the whole eleven-row packet $Q = (S, A, D, B, T, R, E, H, C, P, N)$, because each sibling contributes only a source, convergence-test consumer, regular readback, dyadic estimate, or seal face while this chapter binds the finite summation-by-parts identity with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelSummation}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelSummation}^{\uparrow}$ packet is generated from finite partial-sum rows, displayed coefficient windows, adjacent finite-difference rows, boundary-term rows, a finite tail-estimate row, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{AbelSummation}^{\uparrow}}$, seed)

Scope closed: Definition 1763.1, Definition 1763.2, Definition 1763.3, and Definition 1763.4 seed the finite summation-by-parts boundary over Series$^{\uparrow}$, AbelTest$^{\uparrow}$, DirichletTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{AbelSummation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected infinite sums, quotient series equality, host real completeness, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite NameCert obligation surface, partial-sum handoff theorem, and Dirichlet-test route theorem.

Chapter 1764

A Concrete Naming Certificate for $\mathsf{UniformLimitMetric}^{\uparrow}$

Following Bourbaki and Bishop–Bridges, the $\mathsf{UniformLimitMetric}^{\uparrow}$ packet records the traditional metric-valued uniform-limit handoff as a finite L10 NameCert route. It sits between the uniform-limit continuity boundary of chapter 1748, the interchange route of chapter 1749, the complete-metric consumer surface of chapter 128, and the displayed metric source of chapter 57. Its distance rows are read through the rational and real route of chapter 301, chapter 38, and chapter 291.

The packet exposes the finite metric data needed by a complete-metric consumer of a uniformly convergent metric-valued family. It does not introduce host completeness, quotient streams, selected limits, open-cover compactness, countable choice, or a second metric completion object.

Definition 1764.1 (Uniform-limit-metric carrier)**.** *A* $\mathsf{UniformLimitMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, F, U, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier, distance classifier, metric-law ledger, and the finite distance observations available to the packet.*
2. *F is the metric-valued family row. It records the displayed family entries whose pairwise distances are inspected only through M and through finite windows.*
3. *U is the uniform modulus row, giving one shared threshold schedule for the family and for the endpoint-facing read.*
4. *B is the finite dyadic distance-budget row. It stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, request indices, late-window bounds, and distance comparison budgets consumed by U.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-comparison row obtained after the metric distance observations and dyadic budgets have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for terminal distance readings reached from the regular tail comparison.*
7. *H records componentwise* hsame *transport for the metric source, family row, uniform modulus, dyadic budgets, regular tail comparison, Real seal, continuation routes, package provenance, and local naming row.*
8. *C records displayed* Cont *replay from the metric source through the family row, uniform modulus, dyadic budget, regular tail comparison, Real seal, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* UniformLimitMetric$^{\uparrow}$, UniformLimitContinuous$^{\uparrow}$, UniformLimitInterchange$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{UniformLimitMetric}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, F, U, B, R, E, H, C, P, N$. It has no coordinate for host completeness, quotient streams, selected limits, open-cover compactness, countable-choice selectors, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 1764.2 (Uniform-limit-metric NameCert obligations)**.** *For every accepted* UniformLimitMetric$^{\uparrow}$ *carrier $Q = (M, F, U, B, R, E, H, C, P, N)$, the local* NameCert$_{\text{UniformLimitMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, metric-valued family, uniform modulus, dyadic distance budgets, regular tail comparison, Real seal, transport, replay, provenance, and local naming rows;*
2. *classifier exactness, requiring every metric-valued family read to use the point and distance classifiers displayed by M;*
3. *uniform-modulus stability, requiring endpoint-facing reads and family-facing reads to consult the same threshold schedule U before any tail comparison is exported;*
4. *dyadic budget exactness, requiring every distance estimate to pass through the finite* DyadicRatCore$^{\uparrow}$ *budget row B;*
5. *RegSeqRat tail comparison, requiring R to be obtained only after the metric distance observations and dyadic budgets have been displayed;*
6. *Real seal handoff, requiring E to be consumed only after R and only along the route replayed by C;*
7. *ledger nonescape, excluding host completeness, quotient streams, selected limits, open-cover compactness, countable-choice selectors, hidden completion objects, and ambient* Real$^{\uparrow}$ *completeness as local certificate data.*

No uniform-limit metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 1764.1. The analytic rows are exactly the metric source M, the metric-valued family row F, the uniform modulus U, the dyadic budget row B, the regular tail comparison R, and the Real seal E.

Classifier exactness follows from the shape of M and F. The family row contains no independent point equality or distance relation, so every family-facing read must return to the displayed point and distance classifiers carried by M.

The modulus and budget rows then determine the tail comparison route. The replay row C places U before B, places B before R, and places R before E, while H transports only the same listed coordinates.

The refusal clauses are forced by absence of coordinates. Since the carrier has no host completeness, quotient stream, selected limit, open-cover compactness, choice selector, hidden completion object, or ambient Real-completeness row, those objects cannot become local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.UniformLimitMetricUp.UniformLimitMetricTasteGate single carrier alignment`

Theorem 1764.3 (Uniform-limit-metric complete handoff)**.** *Every complete-metric consumer of an accepted* $\mathsf{UniformLimitMetric}^{\uparrow}$ *carrier reads the finite route*

$$M, F \longmapsto U \longmapsto B \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives the displayed metric classifier, uniform modulus, finite dyadic distance budgets, $\mathsf{RegSeqRat}^{\uparrow}$ *tail comparison,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it reads the packet. The route never exports host completeness, quotient streams, selected limits, open-cover compactness, countable choice, hidden completion objects, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a complete-metric consumer of Q. By Definition 1764.1, the consumer can enter the packet only through the displayed metric source M and the metric-valued family row F.

The uniform threshold route is visible before any completion-facing read. The row U supplies the shared modulus, and the dyadic budget row B supplies the finite distance tolerances attached to the inspected windows.

After those rows have been displayed, the regular tail comparison R and the Real seal E become available. The structural rows H, C, P, N transport, replay, source, and name that same finite route.

Composing these projections gives the displayed factorization. Applying Definition 1764.2, a complete-metric consumer cannot replace the route by host completeness, quotient streams, selected limits, open-cover compactness, countable choice, hidden completion data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1764.1, Definition 1764.2, and Definition 1764.3 seed the finite metric-valued uniform-limit route through displayed metric classifiers, uniform modulus rows, dyadic distance budgets, RegSeqRat tail comparisons, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLimitMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full metric completeness, open-cover compactness, host completeness, quotient streams, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, uniform-modulus stability, dyadic budget exactness, RegSeqRat tail comparison, Real seal handoff, ledger nonescape, and complete-metric consumer handoff for BEDC.Derived.UniformLimitMetricUp.

Constructive story: A $\mathsf{UniformLimitMetric}^{\uparrow}$ packet is generated from a displayed MetricUp source, a metric-valued family row, a shared uniform modulus, finite DyadicRatCore distance budgets, a RegSeqRat tail comparison, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1765

A Concrete Naming Certificate for $\mathsf{HardyCesaroMean}^{\uparrow}$

Following Hardy on divergent series and Bishop–Bridges constructive analysis, the $\mathsf{HardyCesaroMean}^{\uparrow}$ packet records finite Cesaro averaging of a displayed real-name sequence before any Real-completion consumer reads a transformed endpoint. It sits between the summability-transform route of chapter 1682, the shared finite-tail context of chapter 1736, the finite sequence windows of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names a finite averaging route; it does not import a host summability theorem, a selected limit, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1765.1 (Hardy–Cesaro mean carrier)**.** *A* $\mathsf{HardyCesaroMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, P, U, D, R, E, T, C, Q, N)$$

with the following displayed rows:

1. *S is the source real-name sequence window, listing only the finite* $\mathsf{StreamName}^{\uparrow}$ *observations of the input sequence requested by the current Cesaro prefix.*
2. *P stores finite prefix sums, each obtained from the displayed source rows in S and never from a completed host sequence.*
3. *U stores the unary denominator and prefix-length rows, including the visible length used to divide each prefix sum by its Cesaro denominator.*
4. *D stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division rows and tolerance budgets for prefix summation, denominator inversion, averaging, and endpoint-facing error accounting.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite Cesaro means exposed by P, U, D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, prefix sums, denominator rows, dyadic budgets, and regular readback have been displayed.*
7. *T stores componentwise* hsame *transport for source windows, prefix sums, denominator rows, dyadic budgets, regular readback, real seal, continuation, provenance, and local naming.*
8. *C stores displayed* Cont *replay from source-window reads to prefix sums, from prefix sums and denominator rows to dyadic division, from dyadic division to regular readback, and from regular readback to the real seal.*

9. *Q is the* Pkg *provenance over* $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{Seq}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FastConvergentSeries}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyCesaroMean}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, P, U, D, R, E, T, C, Q, N$. It has no coordinate for a host summability method, selected limit, quotient equality of real-name streams, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.HardyCesaroMeanUp.HardyCesaroMeanUp`
Lean4 ✓ `BEDC.Derived.HardyCesaroMeanUp.HardyCesaroMeanTasteGate single carrier alignment`

Theorem 1765.2 (Hardy–Cesaro mean NameCert obligations)**.** *For every accepted* $\mathsf{HardyCesaroMean}^{\uparrow}$ *carrier $H = (S, P, U, D, R, E, T, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HardyCesaroMean}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the displayed source-window, prefix-sum, denominator, dyadic-division, regular-readback, real-seal, transport, continuation, provenance, and naming rows;*
2. *finite Cesaro averaging exactness, requiring each exposed mean to be computed from the displayed prefix sum P, visible prefix length U, and dyadic division ledger D;*
3. *transformed regular-readback handoff, requiring every transformed sequence read to pass through R after the source, prefix, denominator, and dyadic rows have been displayed;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring the endpoint row E to consume only the displayed finite averaging route;*
5. *transport and provenance, requiring T, C, Q, N to preserve, replay, source, and name the same packet without adding a hidden host limit or summability coordinate.*

Proof. Project the carrier of Definition 1765.1. Carrier admission is the finite inventory $S, P, U, D, R, E, T, C, Q, N$, so the local certificate has no detached coordinate for an infinite source sequence or a completed mean sequence.

Finite Cesaro averaging exactness is read from the order of the displayed rows. The source window S feeds the finite prefix sums P; the visible denominator row U supplies the prefix length; and the dyadic ledger D records the finite division and tolerance budget used by the exposed mean.

The transformed readback is then forced through R. A $\mathsf{Real}^{\uparrow}$ endpoint can only continue from R to the seal row E, with no second route through a host summability theorem, selected limit, quotient equality, countable choice, trichotomy, or ambient completeness principle.

Finally, T, C, Q, N transport, replay, source, and name exactly the displayed route. They are structural rows over the same finite packet and therefore cannot contribute an additional analytic principle. □

Theorem 1765.3 (Hardy–Cesaro regular transform handoff)**.** *Every Real-completion-facing read from an accepted* $\mathsf{HardyCesaroMean}^{\uparrow}$ *carrier factors through the displayed source route in* $\mathsf{Series}^{\uparrow}$ *or* $\mathsf{Seq}^{\uparrow}$*, the summability-transform context of* $\mathsf{ToeplitzLemma}^{\uparrow}$ *and* $\mathsf{UniformCauchySeries}^{\uparrow}$*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets, the* $\mathsf{FastConvergentSeries}^{\uparrow}$ *tail discipline where present, and the* $\mathsf{Real}^{\uparrow}$ *seal. Consequently the read exposes only*

$$S, P, U, D, R, E, T, C, Q, N$$

before it reaches a completion-facing consumer.

Proof. Start with a completion-facing request for the Cesaro-transformed sequence. By Definition 1765.1, the request first sees the source rows through S and the provenance row Q, so the source is available only as displayed $\mathsf{Series}^{\uparrow}$ or $\mathsf{Seq}^{\uparrow}$ data.

The averaging route is the finite matrix-row instance of the summability-transform surface named by chapter 1682. When a uniform tail or summable-series context is present, chapter 1736 supplies the sibling route that keeps all tail information inside finite partial-sum and threshold rows.

The actual Cesaro mean is then read through P, U, D. The selected source windows are the $\mathsf{StreamName}^{\uparrow}$ observations in S, the finite divisions and tolerance budgets are the $\mathsf{DyadicRatCore}^{\uparrow}$ rows in D, and the transformed regular sequence reaches consumers only through the $\mathsf{RegSeqRat}^{\uparrow}$ row R.

If a fast tail discipline is part of the displayed provenance, it is consumed only through the $\mathsf{FastConvergentSeries}^{\uparrow}$ rows named by Q before the endpoint reaches E. The endpoint is therefore a $\mathsf{Real}^{\uparrow}$ seal for the already displayed route, not a selected limit or an ambient completeness witness.

The rows T, C, Q, N preserve the same route under transport, continuation replay, package provenance, and local naming. Since every completion-facing read has already passed through those finite rows, no host summability principle, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can be recovered from the packet. □

Lean4 ✓ `BEDC.Derived.HardyCesaroMeanUp.HardyCesaroMeanRegularTransformHandoff`

Falsifiable prediction: Every $\mathsf{HardyCesaroMean}^{\uparrow}$ completion-facing read factors through finite source windows, prefix sums, unary denominator rows, DyadicRatCore division budgets, RegSeqRat readback, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: series, seq, toeplitzlemma, uniformcauchyseries, streamname, regseqrat, dyadicratcore, fastconvergentseries, real: $\mathsf{HardyCesaroMean}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source sequence windows, finite prefix sums, unary denominator rows, dyadic averaging budgets, regular transformed readback, optional tail-discipline provenance, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HardyCesaroMean}^{\uparrow}}$.

Constructive story: A $\mathsf{HardyCesaroMean}^{\uparrow}$ packet is generated from source real-name sequence windows, finite prefix sums, unary denominator and prefix-length rows, dyadic division and tolerance budgets, RegSeqRat readback for Cesaro means, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HardyCesaroMean}^{\uparrow}}$, seed)

Scope closed: Definition 1765.1, Definition 1765.2, and Definition 1765.3 seed the finite Cesaro averaging route from real-name sequence windows to Real-completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HardyCesaroMean}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host summability, selected limits, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite Cesaro averaging exactness, transformed regular-readback handoff, Real seal nonescape, transport and provenance, and TasteGate field faithfulness for BEDC.Derived.HardyCesaroMeanUp.

Chapter 1766

A Concrete Naming Certificate for $\mathsf{HolderContinuity}^{\uparrow}$

Following Holder and Bishop–Bridges constructive analysis, the $\mathsf{HolderContinuity}^{\uparrow}$ packet records the classical power-law continuity surface as finite BEDC data. It sits between the displayed metric source of chapter 57, the terminal real coefficient route of chapter 38, the graph-facing map surface of chapter 130, the exponent-one boundary of chapter 410, and the explicit modulus routes of chapter 1631 and chapter 369. The packet gives later Real and Completion consumers a finite power-law modulus layer without forcing all continuity data to be either Lipschitz or arbitrary-modulus data.

Definition 1766.1 (Holder continuity carrier)**.** *A* $\mathsf{HolderContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, G, \alpha, K, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, carrying the source point classifier, source distance classifier, and the finite source-distance observations inspected by the packet.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, carrying the target point classifier, target distance classifier, and the finite target-distance observations exposed by the graph row.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from X to Y, including only those finite source and target observations consumed by the continuity read.*
4. *α is the rational exponent row. It records a positive rational exponent, its displayed comparison data, and the finite power-law read discipline used by the packet.*
5. *K is the* $\mathsf{Real}^{\uparrow}$ *coefficient ledger for the displayed Holder constant and the real-valued distance estimates used by the power-law bound.*
6. *M is the induced* $\mathsf{ModulusContinuity}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *row obtained from X, Y, G, α, K by reading the power-law budget as a displayed modulus.*
7. *H records componentwise* hsame *transport for the two metric rows, graph row, rational exponent, coefficient ledger, induced modulus row, continuation route, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite source-distance request through G, α, K, M, H, P, N to the target-distance estimate.*

9. *P is the* Pkg *provenance over* HolderContinuity$^\uparrow$, Met$^\uparrow$, Real$^\uparrow$, ContinuousMap$^\uparrow$, ModulusContinuity$^\uparrow$, UniformModulus$^\uparrow$, LipschitzMap$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{HolderContinuity}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, Y, G, \alpha, K, M, H, C, P, N$. It has no coordinate for a host real power function, unrestricted exponent calculus, ambient metric completion, quotient metric equality, selected limit, choice sequence, or a continuity principle detached from the displayed graph.

Lean4 ✓ `BEDC.Derived.HolderContinuityUp.HolderContinuityTasteGate single carrier alignment`

Theorem 1766.2 (Holder continuity NameCert obligations)**.** *For every accepted* HolderContinuity$^\uparrow$ *carrier $Q = (X, Y, G, \alpha, K, M, H, C, P, N)$, the local* NameCert$_{\text{HolderContinuity}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source and target* Met$^\uparrow$ *rows,* ContinuousMap$^\uparrow$ *graph row, rational exponent row,* Real$^\uparrow$ *coefficient ledger, induced modulus row, transport, replay, provenance, and local naming rows;*
2. *exponent and coefficient stability, requiring α and K to be transported only componentwise and to remain attached to the same graph row G;*
3. *power-law metric-bound read, requiring every accepted source-distance request to read the target-distance estimate through X, Y, G, α, K before any modulus row is exported;*
4. *induced modulus handoff, requiring M to be consumed as the displayed* ModulusContinuity$^\uparrow$ *or* UniformModulus$^\uparrow$ *row generated by the power-law budget;*
5. *transport and replay, requiring H and C to preserve the same displayed dependency order under* hsame *transport and* Cont *replay;*
6. *ledger nonescape, excluding host real power functions, unrestricted exponent calculus, ambient metric completion, quotient metric equality, selected limits, choice sequences, and detached continuity principles as local certificate data.*

No Holder continuity obligation is stored outside these rows.

Proof. Project the carrier of Definition 1766.1. Carrier admission exposes exactly the two metric rows X, Y, the continuous graph row G, the exponent row α, the coefficient ledger K, the induced modulus row M, and the structural rows H, C, P, N.

Exponent and coefficient stability are componentwise clauses. The row α is read only as the displayed rational exponent, while K is read only as the Real coefficient ledger attached to the same graph row G.

The power-law metric-bound read is forced by the order of the rows. A source-distance request first reads the metric source X, then the graph row G, then the target metric row Y, and only then consults α and K for the displayed power-law estimate.

The modulus handoff is the row M. Since M is downstream of X, Y, G, α, K, every ModulusContinuity$^\uparrow$ or UniformModulus$^\uparrow$ consumer receives the displayed power-law budget rather than an arbitrary continuity witness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. The excluded objects have no coordinate in the carrier, so they cannot become local NameCert obligations. □

Theorem 1766.3 (Holder continuity modulus handoff)**.** *Every* ModulusContinuity$^\uparrow$ *or* UniformModulus$^\uparrow$ *consumer of an accepted* HolderContinuity$^\uparrow$ *carrier reads the Holder packet only through* Met$^\uparrow$, Real$^\uparrow$,

ContinuousMap$^{\uparrow}$, ModulusContinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, LipschitzMap$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. Equivalently, the consumer factors through the displayed route*

$$X, Y, G, \alpha, K \longmapsto M \longmapsto H, C, P, N.$$

The handoff exports no host real power function, unrestricted exponent calculus, ambient metric completion, quotient metric equality, selected limit, choice sequence, or detached continuity principle.

Proof. Begin with a modulus consumer of Q. By Definition 1766.1, the only modulus-facing coordinate is M, and M is displayed only after the two metric rows, graph row, exponent row, and coefficient ledger have been read.

The first part of the route is analytic. The source and target distance classifiers come from X and Y, the graph-facing continuity data come from G, the rational exponent comes from α, and the Real coefficient and distance estimates come from K.

The second part of the route is the continuity handoff. The row M is read as a ModulusContinuity$^{\uparrow}$ or UniformModulus$^{\uparrow}$ row, while the LipschitzMap$^{\uparrow}$ dependency is visible only as the exponent-one boundary named in provenance, not as an independent replacement for the Holder packet.

The structural rows H, C, P, N preserve the same factorization under transport, replay, provenance, and local naming. Since every consumer-facing read has already passed through those rows, none of the excluded host-level or choice-level objects can be recovered from the packet. □

Closure status for NameCert$_{\text{HolderContinuity}^{\uparrow}}$.

Constructive story: A HolderContinuity$^{\uparrow}$ packet is generated from source and target MetricUp rows, a ContinuousMap graph row, a positive rational exponent row, a RealUp coefficient ledger, an induced ModulusContinuity or UniformModulus row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HolderContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 1766.1, Definition 1766.2, and Definition 1766.3 seed the finite power-law continuity route from displayed metric and graph data to modulus consumers.

Formal verification: FormalStatus(NameCert$_{\text{HolderContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted exponent calculus, host real power functions, ambient metric completion, quotient metric equality, selected limits, choice sequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponent and coefficient stability, power-law metric-bound read, induced modulus handoff, transport, replay, ledger nonescape, and modulus-consumer handoff for BEDC.Derived.HolderContinuityUp.

Chapter 1767

A Concrete Naming Certificate for $\mathsf{FiniteEpsilonNet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ packet records the finite epsilon-net witness used before total-bounded, compact-metric, and continuous-map consumers read a metric handoff. It sits over the displayed metric source of chapter 57, the tolerance-indexed total-boundedness surface of chapter 127, the compact-metric consumer row of chapter 129, and the finite rational tolerance route supplied by $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$. The packet keeps centers, coverage witnesses, and transport ledgers as finite BEDC rows rather than as a host-level finite subset or an open-cover compactness assertion.

Definition 1767.1 (Finite epsilon-net carrier)**.** *A* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, \varepsilon, P, C, D, H, R, S, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier, distance classifier, and metric-law ledger read by the packet.*
2. *ε is the positive tolerance row, read as a* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cell together with the displayed positivity witness used by downstream coverage claims.*
3. *P is the closed* $\mathsf{ProbeBundle}$ *of centers. Its membership rows are the only center-admission data exported by the packet.*
4. *C is the center-carrier ledger, recording for each accepted center of P the corresponding point-classifier admission in X.*
5. *D is the distance and coverage ledger. For each accepted source point inspected by the packet it displays an admitted center of P and a metric-distance row in X bounded by ε.*
6. *H records componentwise* hsame *transport for the metric source, tolerance row, center bundle, center-carrier ledger, distance and coverage ledger, replay row, provenance row, and local naming row.*
7. *R records displayed* Cont *replay from a coverage request through $X, \varepsilon, P, C, D, H, S, N$.*
8. *S is the* Pkg *provenance over* $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, \varepsilon, P, C, D, H, R, S, N$. It has no coordinate for an ambient finite subset, a host enumeration, an open cover, a selected compactness proof, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1767.2 (Finite epsilon-net NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier $Q = (X, \varepsilon, P, C, D, H, R, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, positive tolerance, closed center bundle, center-carrier ledger, coverage-distance ledger, transport, replay, provenance, and local naming rows;*
2. *center visibility, requiring every center used by a coverage witness to enter through the displayed* $\mathsf{ProbeBundle}$ *row P and the center-carrier ledger C;*
3. *finite coverage, requiring each inspected source point to be paired by D with an admitted center and a displayed distance row bounded by ε;*
4. *tolerance transport, requiring the same* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row to be preserved by H before any coverage conclusion is exported;*
5. *metric-distance handoff, requiring every coverage distance to be read through the point and distance classifiers carried by X;*
6. *ledger nonescape, excluding ambient finite subsets, host enumerations, open covers, selected compactness proofs, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No finite epsilon-net obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 1767.1. The analytic rows are exactly the metric source X, positive tolerance ε, closed center bundle P, center-carrier ledger C, and coverage-distance ledger D.

Carrier admission and center visibility are rowwise. A center can be used only after it is visible in P and admitted by C as a point of the metric source X.

The coverage claim is then forced by D. Each inspected source point is paired with one displayed center and one distance comparison, and that distance comparison is read through the metric classifiers of X before it is bounded by ε.

The transport and replay rows preserve the same route. The row H moves no coordinate beyond $X, \varepsilon, P, C, D, R, S, N$, and R replays coverage requests through the same ordered data.

The refusal clauses follow from absence of coordinates. Since the carrier has no host finite subset, host enumeration, open-cover family, compactness selector, choice schedule, or ambient Real-completeness row, none of those objects can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.FiniteEpsilonNetUp.FiniteEpsilonNetTasteGate single carrier alignment`

Theorem 1767.3 (Finite epsilon-net coverage obligation)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier $Q = (X, \varepsilon, P, C, D, H, R, S, N)$ must read the metric source row X, the positive tolerance row ε, the center bundle P, the center-carrier ledger C, and the coverage-distance ledger D before it may use the net as a covering witness. The consumer receives no ambient finite subset, host enumeration, open-cover family, selected compactness proof, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows. The only coverage data are the metric source X, positive tolerance ε, center bundle P, center-carrier ledger C, and coverage-distance ledger D.

The center rows are prerequisite data. A consumer cannot use a proposed center before the bundle P displays it and the ledger C admits it as a point of the metric source X.

The coverage row is then read through D. Each inspected source point is paired with an admitted center and a displayed distance comparison bounded by ε.

Apply Definition 1767.5. That handoff places every total-bounded or compact-metric read downstream of the same finite center and distance rows, so the covering witness cannot be replaced by a host finite subset or open-cover argument. □

Theorem 1767.4 (Finite epsilon-net center-transport obligation)**.** *For every accepted* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier* $Q = (X, \varepsilon, P, C, D, H, R, S, N)$*, componentwise* hsame *transport preserves exactly the same center rows of* P*, the same center-carrier admissions of* C*, and the same coverage-distance ledger* D*. A transported finite-net read may replay the route through* H *and* R*, but it may not introduce a new center, tolerance, coverage comparison, host enumeration, open-cover proof, or ambient completeness row.*

Proof. Project the transport row H. It acts componentwise on $X, \varepsilon, P, C, D, R, S, N$ and has no coordinate for additional centers or detached coverage witnesses.

The center data are therefore unchanged. Transport can move the displayed bundle row P and the admission ledger C, but it cannot add a center outside the bundle or drop the metric-source admission attached to that center.

The coverage-distance ledger is transported in the same way. Each row of D keeps the inspected source point, admitted center, and distance comparison bounded by ε.

Apply Definition 1767.5. Downstream total-bounded and compact-metric consumers still read the same X, ε, P, C, D route after transport, so no host enumeration, open-cover proof, selected compactness witness, or ambient completeness row is exported. □

Theorem 1767.5 (Finite epsilon-net totally-bounded handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier reads the finite route*

$$X, \varepsilon \longmapsto P, C, D \longmapsto H, R, S, N.$$

Thus the consumer receives the finite center bundle, positive tolerance row, coverage witnesses, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *before any total-bounded conclusion is available. The handoff exports no ambient finite subset, host enumeration, open cover, selected compactness proof, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a total-bounded or compact-metric consumer of Q. By Definition 1767.1, the consumer can enter the packet only through the metric source X and the displayed tolerance row ε.

The finite center data are opened next. The closed bundle P names the centers, C admits those centers as points of X, and D supplies the corresponding distance and coverage witnesses.

The structural rows then package exactly that finite route. Componentwise transport H preserves the same tolerance and center data, replay R exposes the route to downstream consumers, provenance S records the package lineage, and N names the local certificate.

Therefore the total-bounded handoff factors through the displayed finite-net route before any consumer can use total boundedness. Applying Definition 1767.2, the consumer cannot replace the route by a host finite subset, host enumeration, open-cover argument, selected compactness proof, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1767.6 (Finite epsilon-net Lebesgue-number handoff)**.** *Every* $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *carrier reads the finite route*

$$X, \varepsilon, P, C, D \longmapsto H, R, S, N$$

before it may inspect a cover-stability tolerance. Thus the consumer receives center rows, membership rows, dyadic or rational tolerance rows, the $\mathsf{TotallyBounded}^{\uparrow}$ *refinement surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The handoff exports no selected subcover, open-cover compactness theorem, host* $\mathsf{Real}^{\uparrow}$ *equality, countable-choice schedule, or ambient completeness assertion.*

Proof. Project the carrier of Definition 1767.1. The only tolerance-facing rows available to a Lebesgue-number finite-net consumer are the metric source X, the positive tolerance row ε, the closed center bundle P, the center-carrier ledger C, and the coverage-distance ledger D.

Use Definition 1767.2. Center visibility and finite coverage force every center and membership read through P, C, D, while tolerance transport keeps the same rational or dyadic tolerance row visible under H.

Apply Definition 1767.5 for the refinement surface consumed by $\mathsf{TotallyBounded}^{\uparrow}$ and compact-metric routes. The Lebesgue-number finite-net handoff may read that finite refinement surface before a cover-stability tolerance, but it cannot replace the finite route by an open-cover compactness proof.

The nonescape clauses are inherited from the same rows. Since the packet has no coordinate for a selected subcover, host Real equality, countable-choice schedule, or ambient completeness principle, no $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ consumer can obtain one from this handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEpsilonNet}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1767.1, Definition 1767.2, and Definition 1767.5 seed the finite epsilon-net packet over displayed metric sources, positive rational or dyadic tolerances, closed center bundles, coverage ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteEpsilonNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full total boundedness, compact metric completeness, uniform continuity, open-cover compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center visibility, finite coverage, tolerance transport, metric-distance handoff, ledger nonescape, and total-bounded consumer handoff for BEDC.Derived.FiniteEpsilonNetUp.

Constructive story: A $\mathsf{FiniteEpsilonNet}^{\uparrow}$ packet is generated from a MetricUp source row, a positive RatUp or DyadicRatCoreUp tolerance, a closed ProbeBundle of centers, a center-carrier ledger, distance and coverage rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1768

A Concrete Naming Certificate for $\mathsf{DyadicToleranceScale}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicToleranceScale}^{\uparrow}$ packet records the single dyadic precision scale row that a regular rational or real-name consumer must read before turning finite stream observations into endpoint-facing data. It sits over the dyadic tolerance arithmetic of chapter 291, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the Cauchy threshold discipline of chapter 292, and the terminal real-name seal of chapter 38. The packet gives Real and Completion routes one visible scale certificate instead of spreading the same precision ledger across adjacent dyadic, modulus, and readback chapters.

Definition 1768.1 (Dyadic tolerance scale carrier)**.** *A* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (s, d, w, r, m, e, h, c, p, n)$$

with the following displayed rows:

1. *s is the scale row. It records the finite dyadic precision index, the displayed tolerance requested at that index, and the monotone relation from coarser requests to finer requests.*
2. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to s, including the dyadic radius, rounding, comparison, and containment rows used by the scale.*
3. *w is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the precision named by s and budgeted by d.*
4. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the scale row, dyadic ledger, and stream window have been displayed.*
5. *m is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row saying that the same scale controls the finite tail bound consumed by regular readback.*
6. *e is the terminal* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after s, d, w, r, m have been read.*
7. *h records componentwise* hsame *transport for the scale row, dyadic ledger, stream window, regular readback, Cauchy threshold, Real handoff, replay, provenance, and local naming rows.*
8. *c records displayed* Cont *replay from the scale request through the dyadic ledger, stream window, regular readback, Cauchy threshold, Real handoff, transport, provenance, and local naming rows.*
9. *p is the* Pkg *provenance over* $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *n is the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, d, w, r, m, e, h, c, p, n$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 (definition) `BEDC.Derived.DyadicToleranceScaleUp.DyadicToleranceScaleUp`

Theorem 1768.2 (Dyadic tolerance scale NameCert obligations)**.** *For every accepted* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier $T = (s, d, w, r, m, e, h, c, p, n)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed scale row, DyadicRatCore tolerance ledger, StreamName window, RegSeqRat readback, CauchyModulus threshold, Real handoff, transport, replay, provenance, and local naming rows;*
2. *scale monotonicity, requiring the row s to order coarser and finer dyadic precision requests before any ledger, window, readback, threshold, or seal row can be consumed;*
3. *classifier transport, requiring accepted comparisons to preserve $s, d, w, r, m, e, h, c, p, n$ componentwise and to expose no hidden precision coordinate;*
4. *RegSeqRat handoff, requiring the readback row r to consume the displayed scale, dyadic ledger, and stream window rather than a detached stream quotient or selected limit;*
5. *Real-seal non-escape, requiring the row e to remain terminal with respect to s, d, w, r, m and to export no host equality, countable choice, trichotomy, ambient completeness theorem, Lean verification, or bridge data;*
6. *provenance and local naming, requiring p and n to package exactly the displayed rows and no additional completion datum.*

No dyadic tolerance-scale obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1768.1. The visible packet consists exactly of the scale row s, dyadic ledger d, stream window w, regular readback r, Cauchy threshold m, Real handoff e, and structural rows h, c, p, n.

Scale monotonicity is the local content of s. Since s is the only row carrying the precision index and the coarser-to-finer relation, every downstream precision comparison must read it before any ledger, window, readback, threshold, or seal can be named.

Classifier transport is componentwise through h and replayed through c. These rows preserve the displayed route from s to d, from d to w, from w to r, from r to m, and from m to e.

The RegSeqRat handoff and Real-seal clauses follow from the placement of r and e. The readback row is downstream of the displayed scale, ledger, and window, while the Real handoff is terminal after the same route and the Cauchy threshold row.

The excluded objects are absent from the carrier and are not produced by transport, replay, provenance, or local naming. Hence no local NameCert obligation can be stored outside the displayed rows. □

Theorem 1768.3 (Dyadic tolerance scale RegSeqRat handoff)**.** *Let $T = (s, d, w, r, m, e, h, c, p, n)$ be an accepted* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of T must read the finite route*

$$s \longmapsto d \longmapsto w \longmapsto r \longmapsto m \longmapsto e \longmapsto h, c, p, n$$

before the regular readback and before the Real seal are available. Consequently the dyadic scale row is the unique precision gate for those consumers, and no consumer can bypass it through quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicToleranceScaleUp.DyadicToleranceScaleRegSeqRatHandoff`

Proof. Project the accepted carrier. The only readback coordinate is r, and the only Real-facing coordinate is e.

The readback coordinate is not free-standing. By Definition 1768.1, r is obtained after the scale row s, the DyadicRatCore ledger d, and the StreamName window w have been displayed.

The Real-facing coordinate is also not free-standing. The row e is downstream of r and the CauchyModulus threshold row m, so a Real consumer reaches it only after the same scale, ledger, window, and regular readback route has already been read.

Finally, h, c, p, n preserve and name that route without adding a second precision coordinate. Since the carrier has no excluded coordinate, every RegSeqRat or Real consumer must pass through the dyadic scale row before readback and seal. □

Theorem 1768.4 (Dyadic tolerance scale precision-gate exhaustion)**.** *Let* $T = (s, d, w, r, m, e, h, c, p, n)$ *be an accepted* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of* T *must pass through the scale row* s*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* d*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* w*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* m*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r*, the* $\mathsf{Real}^{\uparrow}$ *seal* e*, and the structural rows* h, c, p, n*. No quotient stream equality, selected limit, host equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate is available to such a consumer.*

Proof. Project the carrier of Definition 1768.1. The visible rows are exactly the scale, dyadic ledger, finite window, regular readback, Cauchy threshold, Real seal, and structural rows $s, d, w, r, m, e, h, c, p, n$.

Apply Definition 1768.2. The scale monotonicity row must be read before any ledger, window, readback, threshold, or seal row, and the classifier transport exposes no hidden precision coordinate.

Now apply Definition 1768.3. It forces each RegSeqRat or Real consumer through the route from s to d, then w, r, m, e, and finally h, c, p, n.

Because the carrier and the handoff route contain no quotient, selected-limit, host-equality, choice, trichotomy, or completeness row, the scale row exhausts the precision gate for every accepted consumer. □

Theorem 1768.5 (Dyadic tolerance scale obligation-closure scope)**.** *Let* $T = (s, d, w, r, m, e, h, c, p, n)$ *be an accepted* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier. Every accepted obligation read of* T *is exhausted by the scale row* s*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* d*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* w*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row* m*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* r*, the* $\mathsf{Real}^{\uparrow}$ *seal* e*, componentwise* hsame *transport* h*, displayed* Cont *replay* c*,* Pkg *provenance* p*, and the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$ *row* n*. The obligation surface exports no quotient stream equality, selected limit, countable choice, trichotomy, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Proof. Project the accepted carrier through Definition 1768.1. Its visible coordinates are exactly $s, d, w, r, m, e, h, c, p, n$, with s naming the dyadic precision scale and d, w, m, r, e forming the finite route toward regular readback and the Real seal.

Apply Definition 1768.2. The obligation rows are carrier admission, scale monotonicity, classifier transport, RegSeqRat handoff, Real-seal non-escape, provenance, and local naming, and each of those rows is stored on the displayed coordinates of the carrier.

Use Definition 1768.3 together with Definition 1768.4. They force every RegSeqRat or Real consumer through the same scale, DyadicRatCore ledger, StreamName window, CauchyModulus threshold, RegSeqRat readback, Real seal, transport, replay, package, and naming route.

It remains only to identify the boundary. The carrier has no coordinate for quotient stream equality, a selected limit, countable choice, trichotomy, host equality, ambient Real completeness, or bridge data, and the obligation rows do not construct such a coordinate. Therefore the displayed rows exhaust the obligation-closure surface. □

Theorem 1768.6 (Dyadic tolerance scale Real regular readback)**.** *Let $T = (s, d, w, r, m, e, h, c, p, n)$ be an accepted* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read from T factors through the dyadic scale row s, the* $\mathsf{StreamName}^{\uparrow}$ *finite window w, the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row m, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row r, the terminal* $\mathsf{Real}^{\uparrow}$ *seal e, componentwise* hsame *transport h, displayed* Cont *replay c,* Pkg *provenance p, and the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$ *row n. Consequently a Real-facing consumer receives a regular rational readback only through the displayed dyadic scale route, not through quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Proof. Start with Definition 1768.3. It places every RegSeqRat and Real consumer downstream of the route from the scale row s through the dyadic ledger d, the finite stream window w, the regular readback r, the Cauchy threshold m, the Real seal e, and the structural rows h, c, p, n.

Apply Definition 1768.4. The scale row is the unique precision gate for those consumers, and the displayed dyadic ledger, StreamName window, Cauchy threshold, RegSeqRat readback, Real seal, transport, replay, provenance, and naming rows are the only rows available for the read.

Finally use Definition 1768.5. Its closure scope says that the same rows exhaust every accepted obligation read of the packet. Therefore a Real-facing read factors through the regular readback and Real seal exactly along the local NameCert route, with no bypass through an excluded quotient, selected limit, host-equality, choice, completeness, or bridge coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$, obligation)

Scope closed: Definition 1768.1, Definition 1768.2, Definition 1768.3, Definition 1768.4, and Definition 1768.5 close the finite dyadic precision-scale obligation surface from a scale row through dyadic ledger, StreamName window, Cauchy threshold, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicToleranceScale}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: The next step is Lean verification of the displayed obligation-closure surface for BEDC.Derived.DyadicToleranceScaleUp.

Constructive story: A $\mathsf{DyadicToleranceScale}^{\uparrow}$ packet is generated from one dyadic scale row, a DyadicRatCore tolerance ledger, a StreamName finite window, RegSeqRat readback, a CauchyModulus threshold, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1769

A Concrete Naming Certificate for $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ packet records the finite located-tail envelope by which a regular located Cauchy source becomes visible to $\mathsf{Real}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers. It sits over the regular located source boundary of chapter 1454, the paired tail-agreement seal of chapter 699, the limit handoff surface of chapter 386, and the nested interval compactness boundary of chapter 1584. The envelope keeps the tail window, dyadic tolerance, agreement seal, limit seal, and terminal Real seal in one finite NameCert route before any consumer treats the route as completed real data.

Definition 1769.1 (LocatedRealTailEnvelope carrier)**.** *A* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, A, L, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *source row, including the regular rational source, located enclosure, and visible convergence-rate data.*
2. *W is the finite tail-window row naming the source indices inspected by the consumer.*
3. *D is the dyadic tolerance ledger for W, including the requested radii, endpoint comparisons, and readback bounds.*
4. *A is the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row certifying that the displayed tail window agrees with the Real-facing read at the named tolerance.*
5. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row consuming the same finite tail window and dyadic ledger as the limit-seal handoff.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after R, W, D, A, L have been displayed.*
7. *H records componentwise* hsame *transport for the source, tail window, dyadic ledger, agreement seal, limit seal, terminal Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *precision request through $R, W, D, A, L, S, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{RealTailAgreementSeal}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealTailEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted envelopes only through $R, W, D, A, L, S, H, C, P, N$. It has no coordinate for quotient streams, selected host limits, countable choice, open-cover compactness, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.LocatedRealTailEnvelopeUp.LocatedRealTailEnvelopeTasteGate single carrier alignment`

Theorem 1769.2 (LocatedRealTailEnvelope NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ *carrier $E = (R, W, D, A, L, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealTailEnvelope}^{\uparrow}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier exposure for the displayed located Cauchy source, finite tail window, dyadic tolerance ledger, agreement seal, limit seal, terminal Real seal, and structural rows H, C, P, N;*
2. *located-tail route stability, requiring every consumer precision request to read the source row R, the finite window W, and the dyadic ledger D in the same displayed order;*
3. *agreement-seal compatibility, requiring the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *row A to consume the same W, D route as the located source;*
4. *CauchyLimitSeal handoff, requiring the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L to read no tail data outside R, W, D, A;*
5. *Real-seal routing, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row S to be downstream of the source, window, tolerance, agreement, and limit-seal rows;*
6. *ledger non-escape, requiring H, C, P, N to preserve and name only the displayed route and to export no quotient stream, selected host limit, countable-choice selector, open-cover compactness object, or ambient Real completeness principle.*

No NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1769.1. The admitted packet contains the located Cauchy source R, the finite tail window W, the dyadic ledger D, the agreement row A, the Cauchy limit seal L, the terminal Real seal S, and the structural rows H, C, P, N.

Located-tail route stability follows from the order of the displayed rows. A consumer precision request first fixes the located source, then reads only the finite window named for that source, and then reads the dyadic tolerance ledger attached to that window.

Agreement-seal compatibility is forced because A has no window or tolerance coordinate other than W and D. The same route is therefore used for the located source and for the tail-agreement read.

The CauchyLimitSeal handoff is the next row in the packet. The row L consumes the already displayed R, W, D, A route and has no separate source for an unbounded tail, selected limit, or host convergence predicate.

Finally S is terminal, and the rows H, C, P, N only transport, replay, source, and name the same finite route. Since the excluded objects are absent from the carrier inventory, no local obligation can export them. □

Theorem 1769.3 (LocatedRealTailEnvelope Real-seal factorization)**.** *Let $E = (R, W, D, A, L, S, H, C, P, N)$ be an accepted* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer read of E factors through the displayed located tail route*

$$R \longmapsto W \longmapsto D \longmapsto A \longmapsto L \longmapsto S \longmapsto H, C, P, N$$

before reading the terminal Real seal.

Consequently the consumer receives a regular located Cauchy source, finite tail window, dyadic tolerance ledger, RealTailAgreementSeal row, CauchyLimitSeal row, terminal $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. The route cannot use quotient streams, selected host limits, countable choice, open-cover compactness, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier by Definition 1769.1. The only Real-facing endpoint is S, and the carrier places S after the located source, tail window, dyadic ledger, agreement seal, and limit seal.

The first part of the route is the located-tail read. The row R supplies the regular located Cauchy source, W selects the finite tail window inspected for the request, and D supplies the dyadic tolerance ledger for that window.

The second part of the route is the seal read. The row A certifies agreement on the same window and ledger, and L packages the CauchyLimitSeal handoff before S can be exposed.

The structural rows preserve the route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming do not add a second route to $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$. Hence every consumer read factors through the displayed located tail route, and the excluded quotient, selector, compactness, host-equality, and ambient-completeness coordinates remain unavailable. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealTailEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ packet is generated from a RegularLocatedCauchyReal source row, a finite tail-window row, a dyadic tolerance ledger, a RealTailAgreementSeal row, a CauchyLimitSeal row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealTailEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1769.1, Definition 1769.2, and Definition 1769.3 seed the finite located-tail envelope for RealUp and CompletionUp consumers over displayed agreement and limit-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealTailEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, located-tail route stability, agreement-seal compatibility, CauchyLimitSeal handoff, Real-seal routing, and ledger non-escape for the same displayed packet.

Chapter 1770

A Concrete Naming Certificate for $\mathsf{RegularCauchyDistance}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyDistance}^{\uparrow}$ packet records the finite distance ledger between two regular Cauchy names before any $\mathsf{Real}^{\uparrow}$ metric handoff treats the distance as a completed real. It sits over the regular rational readback of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The chapter supplies the distance-account row consumed by Real and metric-facing packets; it does not identify quotient streams, compare host reals by trichotomy, select limits, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1770.1 (Regular Cauchy distance carrier)**.** *A* $\mathsf{RegularCauchyDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\mathrm{dist}} = (X, Y, S, T, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are compared.*
2. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *window row exposing aligned finite observations of X and Y.*
3. *T is the tolerance-window row, recording the displayed tail threshold at which the two names are read for a requested distance precision.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *distance ledger, containing the dyadic absolute-difference bounds and slack rows used by the finite comparison.*
5. *Q is the precision-query row consumed jointly by S, T, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *distance seal row, read only after the finite ledger has displayed the distance account.*
7. *H records componentwise* hsame *transport for the two source rows, paired windows, threshold row, dyadic ledger, precision query, distance seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the two source rows through S, T, D, Q and then into E, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDistance}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDistanceUp.RegularCauchyDistanceUpTasteGate single carrier alignment`

Theorem 1770.2 (Regular Cauchy distance NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDistance}^{\uparrow}$ *carrier*

$$L_{\mathrm{dist}} = (X, Y, S, T, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDistance}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, tolerance-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *distance ledger, precision query, terminal* $\mathsf{Real}^{\uparrow}$ *distance seal, transport, replay, provenance, and local naming rows;*
2. *source alignment, requiring the windows in S to expose observations from the same two source rows X, Y named by the carrier;*
3. *query exactness, requiring Q to be the displayed precision request consumed by the threshold row T and the dyadic ledger D;*
4. *dyadic distance exactness, requiring D to store precisely the absolute-difference bounds and slack rows read through S, T, Q;*
5. *Real metric handoff, requiring E to consume the finite dyadic distance ledger as a* $\mathsf{Real}^{\uparrow}$ *distance seal after X, Y, S, T, D, Q are visible;*
6. *componentwise transport and displayed replay for $X, Y, S, T, D, Q, E, P, N$ through H and C;*
7. *refusal of quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No regular Cauchy distance obligation is stored outside these rows.

Theorem 1770.3 (Regular Cauchy distance classifier stability)**.** *Let*

$$L_{\mathrm{dist}} = (X, Y, S, T, D, Q, E, H, C, P, N)$$

and

$$L'_{\mathrm{dist}} = (X', Y', S', T', D', Q', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyDistance}^{\uparrow}$ *carriers. If their source rows, paired windows, threshold rows, dyadic distance ledgers, precision queries, terminal seals, transport rows, replay rows, provenance rows, and local naming rows are componentwise* hsame*, then the local classifier gives the same distance certificate verdict to L_{dist} and L'_{dist}.*

Consequently classifier stability is ledgerwise: it follows only from the displayed source-window-threshold-ledger-query-seal route and not from quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1770.4 (Regular Cauchy distance Real metric handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *metric consumer that reads a distance between the two source rows of an accepted* $\mathsf{RegularCauchyDistance}^{\uparrow}$ *carrier must factor through the finite route*

$$X, Y \longmapsto S, T, Q \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

The consumer therefore receives the two regular rational source rows, aligned finite observation windows, the requested tolerance threshold, the dyadic absolute-difference ledger, and the terminal real-name distance seal before using the distance as metric data. This handoff is a finite distance account between regular Cauchy names; it is not a quotient equality of streams, host real trichotomy, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Falsifiable prediction: Every $\mathsf{RegularCauchyDistance}^{\uparrow}$ metric-facing read factors through two RegSeqRat source rows, paired StreamName windows, one tolerance-window row, one DyadicRatCore distance ledger, one precision query, one terminal RealUp distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyDistance}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular Cauchy source rows, aligned finite windows, dyadic absolute-difference bounds, and the terminal RealUp distance seal that metric consumers read as one ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDistance}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1770.1, Definition 1770.2, Definition 1770.3, and Definition 1770.4 seed the finite distance ledger over two regular Cauchy names, paired StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readbacks, a terminal RealUp distance seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDistance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source alignment, query exactness, dyadic distance exactness, Real metric handoff, componentwise transport, displayed replay, classifier stability, and metric-consumer nonescape for BEDC.Derived.RegularCauchyDistanceUp.

Constructive story: A $\mathsf{RegularCauchyDistance}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName windows, one tolerance-window row, one DyadicRatCore distance ledger, one precision query, one terminal RealUp distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1771

A Concrete Naming Certificate for $\mathsf{CauchyRealizer}^{\uparrow}$

Following Weihrauch and Bishop–Bridges, the $\mathsf{CauchyRealizer}^{\uparrow}$ packet records the explicit finite route by which effective Cauchy data is decoded, inspected through stream windows, read back as regular rational data, budgeted by dyadic tolerances, and sealed for the current Real and Completion route. It sits over the effective source surface of chapter 1705, the representation decoding boundary of chapter 1680, the represented-space naming interface of chapter 1494, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is the finite NameCert route for explicit Cauchy realization; it is not Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or a standard bridge.

Definition 1771.1 (Cauchy realizer carrier)**.** *A* $\mathsf{CauchyRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (E, R, V, W, Q, D, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row, naming the effective Cauchy input, its visible request schedule, and the finite source evidence used by the realizer.*
2. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoding row, read together with the* $\mathsf{RepresentedSpace}^{\uparrow}$ *row V so that the source name is interpreted through the displayed representation interface.*
3. *V is the represented-space row that fixes the admissible naming surface consumed by R and exported to downstream Real and Completion consumers.*
4. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the realizer for the displayed precision requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the regular rational observations that the Real route may consume.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the requested radii, slack rows, and finite comparison evidence used to certify the readback.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after the source, representation, represented-space, stream-window, regular-readback, and dyadic-ledger rows have been displayed.*

8. *H records componentwise* hsame *transport for the effective source, representation decoder, represented-space row, stream windows, regular readback, dyadic ledger, terminal Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each Real or Completion precision request through $E, R, V, W, Q, D, S, H, P,$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyRealizer}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted realizers only through $E, R, V, W, Q, D, S, H, C, P, N$. It has no coordinate for Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CauchyRealizerUp.CauchyRealizerUpTasteGate single carrier alignment`

Theorem 1771.2 (Cauchy realizer NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealizer}^{\uparrow}$ *carrier*

$$Z = (E, R, V, W, Q, D, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source,* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoder,* $\mathsf{RepresentedSpace}^{\uparrow}$ *naming surface,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows H, C, P, N;*
2. *source-decoder compatibility, requiring the effective Cauchy source E to be read through the representation row R and the represented-space row V before any stream window is exposed;*
3. *finite-window exactness, requiring every precision request to inspect only the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows W attached to E, R, V;*
4. *regular-readback exactness, requiring Q to consume exactly the displayed windows W and to export no second rational-name route;*
5. *dyadic tolerance accounting, requiring D to store precisely the visible tolerance radii, slack rows, and comparison evidence consumed by Q and S;*
6. *Real-seal routing, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row S to be downstream of E, R, V, W, Q, D;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *non-escape for Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, and standard-bridge data.*

No Cauchy realizer obligation is stored outside these rows.

Proof. Project the carrier of Definition 1771.1. Its visible computational source is E, and the only interpretation rows available for that source are the decoder R and represented-space row V.

The finite observation route is then forced by the displayed order. A precision request reaches W only after E, R, V are fixed, reaches the regular readback Q only through those windows, and reaches the dyadic ledger D only as the tolerance account for that readback.

The terminal seal is downstream. The row S has no entrance other than E, R, V, W, Q, D, while H, C, P, N transport, replay, source, and name the same route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 1771.3 (Cauchy realizer representation handoff)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{CauchyRealizer}^{\uparrow}$ *carrier factors its source request through the displayed representation route*

$$E \longmapsto R \longmapsto V \longmapsto W \longmapsto Q \longmapsto D \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer receives an effective Cauchy source, a CauchyRepresentation decoder, a RepresentedSpace naming row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal Real seal, transport, replay, provenance, and local naming. The handoff does not supply quotient stream equality, selected limits, countable choice, host real equality, trichotomy, Real completeness, Lean verification, or a standard bridge.

Proof. Begin with the accepted carrier and read its first three rows. The source row E is not consumed directly by Real or Completion; it is interpreted through the Cauchy representation row R and the represented-space row V.

Next inspect the finite data available after decoding. The only observation surface is the displayed family of stream windows W, and the only regular rational output attached to those windows is Q.

Finally read the tolerance and seal rows. The dyadic ledger D accounts for the requested precision before the terminal Real seal S is exposed, and H, C, P, N preserve that same route. This proves the stated factorization and leaves the excluded quotient, selector, choice, equality, trichotomy, completeness, verification, and bridge objects outside the handoff. □

Theorem 1771.4 (Cauchy realizer Real seal non-escape)**.** *Let* $Z = (E, R, V, W, Q, D, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchyRealizer}^{\uparrow}$ *carrier. Any downstream use of the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S *must pass through the finite source-decoder-window-readback-ledger route*

$$E, R, V, W, Q, D$$

and the structural rows H, C, P, N*. Thus the Real seal cannot be used as a hidden endpoint for Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or a standard bridge.*

Proof. The only $\mathsf{Real}^{\uparrow}$ row in the carrier is S. By Definition 1771.1, S is placed after the effective Cauchy source, the representation decoder, the represented-space row, stream windows, regular readback, and dyadic tolerance ledger.

A downstream consumer therefore reaches S only by replaying the displayed route through C and transporting it componentwise through H. The package row P and local name N record the same route and do not add another endpoint.

Because no carrier coordinate stores completeness, quotient equality, a selected limit, a choice schedule, trichotomy, host real equality, Lean evidence, or bridge data, the Real seal cannot escape the finite route named above. □

Falsifiable prediction: Every $\mathsf{CauchyRealizer}^{\uparrow}$ Real or Completion read factors through an EffectiveCauchySequence source, a CauchyRepresentation decoder, a RepresentedSpace row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: effective-cauchy-sequence, cauchy-representation, representedspace, streamname, regseqrat: $\mathsf{CauchyRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an

effective Cauchy source, representation decoding, represented-space naming, finite stream windows, regular rational readback, dyadic tolerance accounting, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealizer}\uparrow}$.

Constructive story: A $\mathsf{CauchyRealizer}^{\uparrow}$ packet is generated from an EffectiveCauchySequence source row, a CauchyRepresentation decoder row, a RepresentedSpace naming row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealizer}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1771.1, Definition 1771.2, Definition 1771.3, and Definition 1771.4 seed the finite explicit-Cauchy realizer route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealizer}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-decoder compatibility, finite-window exactness, regular-readback exactness, dyadic tolerance accounting, Real-seal routing, representation handoff, and Real-seal non-escape for BEDC.Derived.CauchyRealizerUp.

Chapter 1772

A Concrete Naming Certificate for $\mathsf{RegularCauchySqueeze}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{RegularCauchySqueeze}^{\uparrow}$ packet records the finite squeeze route by which a middle regular Cauchy real name is enclosed between lower and upper regular Cauchy names before any $\mathsf{Real}^{\uparrow}$ consumer treats the enclosure as completed real data. It sits over the regular rational readback of chapter 301, the finite window interface of chapter 126, the shared tail handoff of chapter 1738, and the terminal real-name seal of chapter 38. The theorem-level comparison sandwich of chapter 2163 consumes this regular Cauchy squeeze packet before the displayed limit route of chapter 1413 reaches the terminal Real seal. The packet supplies a non-quotient L10 route for squeeze arguments: lower, middle, and upper names are compared through displayed windows, dyadic tolerances, and a tail-enclosure ledger, not through host completeness, selected limits, trichotomy, or quotient equality.

Definition 1772.1 (Regular Cauchy squeeze carrier)**.** *A* $\mathsf{RegularCauchySqueeze}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q_{\mathrm{sq}} = (L, M, U, W, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. L, M, U *are the lower, middle, and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose regular rational observations form the squeeze boundary.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row exposing aligned finite observations of* L, M, U *for a requested precision.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii, slack budget, and endpoint comparison bounds used by the squeeze read.*
4. T *is the tail-enclosure ledger, certifying that the displayed middle observations remain between the displayed lower and upper observations on the shared tail window at the dyadic budget in* D*.*
5. E *is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row, used only after* L, M, U, W, D, T *have exposed the finite enclosure route.*
6. H *records componentwise* hsame *transport for the three source rows, shared windows, dyadic tolerance row, tail-enclosure ledger, tail-equivalence handoff, replay, provenance, local naming, and terminal seal rows.*

7. *C records displayed* Cont *replay from each Real-facing squeeze request through $L, M, U, W, D, T, E, H, P, N$ before the seal row is read.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySqueeze}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySqueeze}^{\uparrow}}$ *row carrying the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary named by the displayed route.*

The classifier compares accepted packets only through $L, M, U, W, D, T, E, H, C, P, N$ and the terminal seal boundary carried by N. It has no coordinate for quotient stream equality, host real trichotomy, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySqueezeUp.RegularCauchySqueezeTasteGate single carrier alignment`

Theorem 1772.2 (Regular Cauchy squeeze NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySqueeze}^{\uparrow}$ *carrier*

$$Q_{\mathrm{sq}} = (L, M, U, W, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySqueeze}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower, middle, and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the tail-enclosure ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, transport, replay, provenance, local naming, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary carried by N;*

2. *source alignment, requiring W to expose observations from precisely the three source rows L, M, U named by the carrier;*

3. *tolerance exactness, requiring D to be the displayed dyadic budget consumed by the window row W and the tail-enclosure row T;*

4. *squeeze enclosure exactness, requiring T to store only the lower-to-middle and middle-to-upper tail comparisons read through W at the budget in D;*

5. *tail-equivalence handoff, requiring E to consume the same displayed L, M, U, W, D, T route before any Real-facing squeeze conclusion is exposed;*

6. *componentwise transport, displayed replay, provenance, local naming, and terminal seal preservation through H, C, P, N;*

7. *refusal of quotient stream equality, host real trichotomy, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No regular Cauchy squeeze obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1772.1. The packet displays exactly the three regular rational source rows, one shared finite window row, one dyadic tolerance row, one tail-enclosure ledger, one tail-equivalence handoff, and the structural rows.

Source alignment and tolerance exactness follow from the row order. The window row W has no source coordinate other than L, M, U, and the tolerance row D is the only dyadic budget available to the tail-enclosure row T.

The squeeze enclosure clause is therefore finite. The row T can compare the displayed lower-to-middle and middle-to-upper tail observations, but it cannot appeal to a completed real order, a trichotomy case split, or a selected limit.

The handoff row E is downstream of the same route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and the terminal seal preserve that route without adding analytic coordinates.

Inspecting the carrier inventory gives the refusal clause. Since quotient stream equality, host trichotomy, selected limits, countable choice, ambient completeness, Lean verification, and bridge checking are absent from Q_{sq}, none can appear in the local $\mathsf{NameCert}_{\mathsf{RegularCauchySqueeze}^{\uparrow}}$ surface. □

Theorem 1772.3 (Regular Cauchy squeeze tail enclosure route)**.** *Let*

$$Q_{\mathrm{sq}} = (L, M, U, W, D, T, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySqueeze}^{\uparrow}$ *carrier. Every finite squeeze read for the middle row M factors through the displayed route*

$$L, M, U \longmapsto W \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently the read receives three regular Cauchy source rows, shared StreamName windows, one dyadic tolerance budget, one tail-enclosure ledger, one CauchyTailEquivalence handoff, componentwise transport, displayed replay, provenance, and local naming before it can be consumed as squeeze evidence.

Proof. Begin with an accepted squeeze read. Carrier admission forces the read to name the three source rows L, M, U before any window or tolerance data is available.

The read then passes through the shared window row W and the dyadic budget D. These two rows determine the only observations and tolerance bounds that the enclosure ledger T may inspect.

The row T supplies the lower-to-middle and middle-to-upper tail comparisons on that displayed data. Only after those comparisons are present can the handoff row E expose the tail-equivalence route to a Real-facing consumer.

The remaining rows are structural. They transport, replay, source, and name the same route, so the displayed factorization is exhaustive for the local classifier. □

Theorem 1772.4 (Regular Cauchy squeeze Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySqueeze}^{\uparrow}$ *carrier reads the terminal seal only after the finite route*

$$L, M, U \longmapsto W \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N \longmapsto \mathsf{Real}^{\uparrow}$$

has been displayed. The exported seal is therefore a routed finite squeeze certificate for regular Cauchy real names. It is not quotient stream equality, a host real order comparison, trichotomy, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier by Definition 1772.1. The terminal $\mathsf{Real}^{\uparrow}$ seal boundary carried by N is not available before the lower, middle, upper, window, tolerance, enclosure, and handoff rows have been displayed.

The Real-facing read first receives the same squeeze route established in Definition 1772.3. Thus the seal is downstream of L, M, U, W, D, T, E and cannot bypass the tail-enclosure ledger.

The structural rows H, C, P, N preserve the displayed route. They add transport, replay, provenance, and local naming, but no quotient coordinate, host order predicate, trichotomy principle, selector, choice operator, or ambient completeness theorem.

Therefore every exported $\mathsf{Real}^{\uparrow}$ seal remains a finite squeeze certificate over regular Cauchy names, with exactly the boundary claimed. □

Theorem 1772.5 (Regular Cauchy squeeze RealSequenceLimit handoff)**.** *Any* RealSequenceLimit$^{\uparrow}$ *consumer of a* RegularCauchySqueeze$^{\uparrow}$ *packet must read the lower, middle, and upper regular names, the shared* StreamName$^{\uparrow}$ *windows, the* DyadicRatCore$^{\uparrow}$ *tolerance row, the* RegSeqRat$^{\uparrow}$ *readback route, and the terminal* Real$^{\uparrow}$ *seal in the same finite route. The handoff may then be consumed by the* SqueezeTheorem$^{\uparrow}$ *comparison sandwich, but it does not supply host sequence limits, quotient stream equality, trichotomy, selected limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $Q_{\mathrm{sq}} = (L, M, U, W, D, T, E, H, C, P, N)$. The rows L, M, U are the only lower, middle, and upper regular names, and W is the only shared window route through which their observations can be compared.

The dyadic and rational stages are forced by the carrier order. The tolerance row D supplies the budget for the tail-enclosure ledger T, and the same displayed route reaches the RegSeqRat$^{\uparrow}$ readback through the handoff row E before the terminal seal boundary carried by N can be read.

The theorem-level consumer in chapter 2163 therefore receives this packet as a finite lower-middle-upper route, and the displayed limit route of chapter 1413 can consume it only after those rows have been opened. Transport, replay, provenance, and local naming preserve the same route, so no excluded host principle can enter the handoff. □

Theorem 1772.6 (Regular Cauchy squeeze even-odd boundary)**.** *Suppose the lower and upper squeeze witnesses of an accepted* RegularCauchySqueeze$^{\uparrow}$ *packet are supplied by the even and odd parity rows of an accepted* EvenOddCauchyCriterion$^{\uparrow}$ *packet. Then the middle* RegularCauchySqueeze$^{\uparrow}$ *Real-facing read must pass through the parity selector, the shared* CauchyModulus$^{\uparrow}$ *ledger, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and the* RealCauchyCriterion$^{\uparrow}$ *handoff before it reaches the squeeze seal.*

Proof. Start with the squeeze route of Definition 1772.3. It places the middle read between the lower and upper regular rows through the shared window row, dyadic tolerance row, tail-enclosure ledger, and tail-equivalence handoff.

Now apply Definition 1647.3. The even and odd rows are fused only through the parity selector, shared CauchyModulus ledger, DyadicRatCore tolerance ledger, and RealCauchyCriterion handoff carried by the accepted EvenOddCauchyCriterion$^{\uparrow}$ packet.

Composing the two routes gives the boundary. The lower and upper witnesses may enter the squeeze packet from parity rows, but the middle Real-facing read cannot reach the squeeze seal until the parity selector and shared budget have already been displayed. □

Falsifiable prediction: Every RegularCauchySqueeze$^{\uparrow}$ Real-facing read factors through three RegSeqRat source rows, one shared StreamName window route, one DyadicRatCore tolerance row, one tail-enclosure ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and the terminal RealUp seal boundary carried by N.

Independence witness: regseqrat, streamname, cauchy-tail-equivalence, real: RegularCauchySqueeze$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower, middle, and upper regular Cauchy source rows, a shared finite window route, dyadic squeeze tolerance data, a tail-enclosure ledger, and a Real-facing seal boundary.

Closure status for NameCert$_{\text{RegularCauchySqueeze}^{\uparrow}}$**.**

Constructive story: A RegularCauchySqueeze$^{\uparrow}$ packet is generated from lower, middle, and upper RegSeqRat rows, shared StreamName windows, one DyadicRatCore tolerance row, one tail-enclosure ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and the terminal RealUp seal boundary carried by the NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchySqueeze}\uparrow}$, seed)

Scope closed: Definition 1772.1, Definition 1772.2, Definition 1772.3, Definition 1772.4, and Definition 1772.6 seed the finite squeeze route for regular Cauchy real names through displayed windows, dyadic tolerances, tail enclosure, tail equivalence, parity-tail fusion, and a terminal RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchySqueeze}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real order, trichotomy, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source alignment, tolerance exactness, squeeze enclosure exactness, tail-equivalence handoff, componentwise transport, displayed replay, Real seal boundary, and nonescape for BEDC.Derived.RegularCauchySqueezeUp.

Chapter 1773

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ packet records the finite route by which a regular Cauchy family of regular rational names is read along a displayed diagonal selector before any $\mathsf{Real}^{\uparrow}$ consumer treats the result as a completed real name. It sits over the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the parity-aware schedule handoff of chapter 414. The packet supplies a Real and Completion boundary for diagonal extraction; it does not assert quotient stream equality, choose a host limit, invoke countable choice, use trichotomy, claim ambient $\mathsf{Real}^{\uparrow}$ completeness, provide Lean verification, or record a checked bridge.

Definition 1773.1 (Regular Cauchy diagonal extraction carrier)**.** *A* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G_{\mathrm{diag}} = (F, \delta, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source-family row, naming a finite inspection boundary for a family of* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose members are regular Cauchy rational names.*
2. *δ is the unary diagonal selector row, recording the displayed rule that sends a requested diagonal precision to the source index and source-window coordinate read at that precision.*
3. *W is the aligned* $\mathsf{StreamName}^{\uparrow}$ *window family, exposing only the finite observations of F selected by δ and aligned with the requested diagonal tolerance.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic radius, slack, and comparison rows that justify the selected diagonal read.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row, built from the selected windows W and the tolerance ledger D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the family, selector, windows, tolerance ledger, and diagonal readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the source family, unary selector, aligned windows, tolerance ledger, diagonal readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from each Real-facing diagonal request through* $F, \delta, W, D, Q, E, H, P, N$.
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, \delta, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient streams, a selected host limit, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalExtractionUp.RegularCauchyDiagonalExtractionTasteGate`
`single carrier alignment`

Theorem 1773.2 (Regular Cauchy diagonal extraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *carrier*

$$G_{\mathrm{diag}} = (F, \delta, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source family, unary diagonal selector, aligned* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *selector exactness, requiring δ to be the only row that chooses which source-family coordinate and which finite window coordinate are read for a requested diagonal precision;*
3. *window alignment, requiring W to expose precisely the selected source-family observations and no independent schedule;*
4. *dyadic tolerance exactness, requiring D to store the displayed radius, slack, and comparison rows consumed by the selected windows and by the diagonal readback;*
5. *regular readback exactness, requiring Q to be assembled from F, δ, W, D as the diagonal regular rational name;*
6. *Real seal routing, requiring E to be downstream of the finite source-selector-window-ledger-readback route;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *refusal of quotient stream equality, selected host limits, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No diagonal extraction obligation is stored outside these rows.

Proof. Project the carrier of Definition 1773.1. Its only source coordinate is the family row F, and its only selector coordinate is the unary diagonal row δ.

The finite observation route is therefore forced. A Real-facing precision request can reach an observed source-family value only through δ and then through the aligned window family W.

The tolerance and readback rows are downstream of that route. The dyadic ledger D accounts for the requested radius and slack before the regular readback row Q can be formed.

Finally the seal row E and the structural rows H, C, P, N preserve the same route. Since the excluded quotient, selector, choice, trichotomy, completeness, verification, and bridge objects do not occur as coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 1773.3 (Regular Cauchy diagonal extraction Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing diagonal extraction read from an accepted* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *carrier*

$$G_{\mathrm{diag}} = (F, \delta, W, D, Q, E, H, C, P, N)$$

factors through the displayed finite route

$$F \longmapsto \delta \longmapsto W \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a source family, one unary diagonal selector, aligned StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat diagonal readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. The route exports no quotient stream, selected host limit, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Proof. Begin with the accepted carrier and read the source-family row F. A diagonal request has no direct access to a member of the family; it must first pass through the unary selector δ.

The selector determines the finite observations exposed by W. The dyadic ledger D is the only tolerance row attached to those observations, and the diagonal readback row Q is formed only after W and D are visible.

The terminal Real seal E is downstream of F, δ, W, D, Q. The transport, replay, provenance, and naming rows H, C, P, N record the same finite route and do not add a second Real-facing path.

Because the carrier has no quotient-stream coordinate, selected host limit, countable-choice schedule, trichotomy principle, ambient completeness theorem, Lean evidence, or bridge data, every Real-facing read factors through exactly the route displayed above. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ Real-facing read factors through a source family, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-interleaving: $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source family, a unary diagonal selector, aligned finite windows, a dyadic tolerance ledger, a regular rational diagonal readback, and a terminal RealUp seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ packet is generated from a source family, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1773.1, Definition 1773.2, and Definition 1773.3 seed the finite diagonal extraction route over regular Cauchy real names through displayed source-family, selector, window, tolerance, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector exactness, window alignment, dyadic tolerance exactness, regular readback exactness, Real seal routing, componentwise transport, displayed replay, and Real-facing nonescape for BEDC.Derived.RegularCauchyDiagonalExtractionUp.

Chapter 1774

A Concrete Naming Certificate for $\mathsf{CauchyMajorantSeries}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyMajorantSeries}^{\uparrow}$ packet records the finite majorant-series ledger used to turn tail estimates into regular rational and real-name control without selecting a completed sum. It sits over the dyadic tolerance scale of chapter 1768, the finite StreamName observation surface of chapter 126, the RegSeqRat readback route of chapter 301, the Bishop regular Cauchy completion boundary of chapter 1639, and the terminal Real seal of chapter 38. The packet gives later Real and Completion tail-control arguments one visible majorant schedule rather than asking them to recover a bound from quotient streams, chosen limits, or ambient completeness.

Definition 1774.1 (Cauchy majorant series carrier)**.** *A* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, D, S, R, B, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the majorant schedule row. It records the finite list of nonnegative rational or dyadic majorant terms, their displayed partial sums, and the tail index attached to each requested tolerance.*
2. *D is the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *tolerance row consumed by A, including the dyadic precision scale and comparison budget for the displayed tail request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the tail index chosen inside A and budgeted by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite window after the majorant schedule and tolerance scale have been displayed.*
5. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, recording that the regular readback has reached the completion-facing Cauchy boundary by the displayed tail route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after A, D, S, R, B have been read.*
7. *H records componentwise* hsame *transport for the majorant schedule, dyadic tolerance scale, StreamName window, RegSeqRat readback, Bishop regular Cauchy completion boundary, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail-control request through $A, D, S, R, B, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMajorantSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, D, S, R, B, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyMajorantSeriesUp.CauchyMajorantSeriesTasteGate single carrier alignment`

Theorem 1774.2 (Cauchy majorant series NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *carrier*

$$M = (A, D, S, R, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMajorantSeries}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the majorant schedule, dyadic tolerance scale, StreamName finite window, RegSeqRat readback, Bishop regular Cauchy completion boundary, Real seal, transport, replay, provenance, and local naming rows;*
2. *schedule exactness, requiring A to store the displayed partial-sum ledger and the tail index attached to each dyadic tolerance request;*
3. *tolerance compatibility, requiring the row D to be the same dyadic scale consumed by A and later by the finite window S;*
4. *readback handoff, requiring S and R to expose only the finite StreamName window and RegSeqRat data opened by the scheduled tail request;*
5. *completion-boundary handoff, requiring B to consume A, D, S, R before any* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
6. *Real-seal non-escape, requiring E to be terminal with respect to A, D, S, R, B and to export no quotient equality, selected limit, countable-choice schedule, host equality, trichotomy, ambient completeness, Lean verification, or bridge data;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No Cauchy majorant-series obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1774.1. The visible packet consists exactly of the majorant schedule A, dyadic tolerance scale D, finite window S, regular readback R, completion boundary B, Real seal E, and structural rows H, C, P, N.

Schedule exactness is the local content of A. A tail-control request must name its finite partial-sum ledger and tail index before it may open a tolerance scale or a stream window.

Tolerance compatibility then follows from the placement of D. The same dyadic scale row budgets the scheduled tail request and the finite observations in S, so the route cannot split into an unstated precision ledger.

The readback and completion handoffs are downstream. The row R reads only the displayed StreamName window opened by S, and B reads only the majorant, tolerance, window, and readback rows before exposing the completion-facing Cauchy boundary.

Finally, E, H, C, P, N seal, transport, replay, and name the same route. Since the excluded objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1774.3 (Cauchy majorant series tail control)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *carrier must factor its tail-control request through the finite route*

$$A \longmapsto D \longmapsto S \longmapsto R \longmapsto B \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed majorant schedule, dyadic tolerance scale, finite StreamName window, RegSeqRat readback, Bishop regular Cauchy completion boundary, terminal Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy principle, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Project the accepted carrier $M = (A, D, S, R, B, E, H, C, P, N)$. The only row carrying tail-control data is the majorant schedule A, and the only row carrying the precision budget for that schedule is D.

The finite observation route is forced by the tuple order. The StreamName row S opens the scheduled window, the RegSeqRat row R reads back that window, and the Bishop regular Cauchy completion row B records the completion-facing boundary after both rows are visible.

The Real or Completion consumer reaches the terminal seal only through E and then through the structural rows H, C, P, N. These rows preserve and replay the displayed route; they do not add quotient stream equality, a selected limit, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every $\mathsf{CauchyMajorantSeries}^{\uparrow}$ tail-control read factors through one majorant schedule row, one DyadicToleranceScale row, one StreamName finite window, one RegSeqRat readback row, one BishopRegularCauchyCompletion boundary row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadic-tolerance-scale, streamname, regseqrat, bishop-regular-cauchy-completion, real: $\mathsf{CauchyMajorantSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a majorant partial-sum schedule, a dyadic tolerance scale, the finite observation window, the regular rational readback, and the completion-facing tail boundary read by Real and Completion consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMajorantSeries}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMajorantSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1774.1, Definition 1774.2, and Definition 1774.3 seed the finite majorant-series route from a displayed majorant schedule through dyadic tolerance scale, StreamName finite window, RegSeqRat readback, Bishop regular Cauchy completion boundary, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMajorantSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exactness, tolerance compatibility, readback handoff, completion-boundary handoff, Real-seal non-escape, transport, replay, provenance, and tail-control factorization for BEDC.Derived.CauchyMajorantSeriesUp.

Constructive story: A $\mathsf{CauchyMajorantSeries}^{\uparrow}$ packet is generated from one majorant schedule row, one DyadicToleranceScale row, one StreamName finite window, one RegSeqRat readback row, one BishopRegularCauchyCompletion boundary row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1775

A Concrete Naming Certificate for $\mathsf{CauchyContinuousImage}^{\uparrow}$

Bishop–Bridges style constructive analysis treats continuity on Cauchy names through displayed moduli and finite tail data rather than through an ambient host function. The $\mathsf{CauchyContinuousImage}^{\uparrow}$ packet records the finite route by which a regular Cauchy source name is read through a continuous-map modulus and then sealed as a Real-facing image. It sits over the StreamName finite-window interface of chapter 126, the RegSeqRat readback route of chapter 301, the DyadicRatCore tolerance arithmetic of chapter 291, the ContinuousMap modulus surface of chapter 130, and the terminal Real seal of chapter 38.

Definition 1775.1 (CauchyContinuousImage carrier)**.** *A* $\mathsf{CauchyContinuousImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, R, D, M, J, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source Cauchy name, including the displayed observation window used by the image computation.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the source windows and their regularity budget.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for source and image estimates, including dyadic radii, endpoint comparisons, and rounding rows.*
4. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row. It records the displayed input tolerance required for each output tolerance request and names the finite continuity route used by the packet.*
5. *J is the image-window row. It records the finite image observations computed only after S, R, D, M have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *image seal reached only after the source readback, dyadic ledger, continuity modulus, and image window are visible.*
7. *H records componentwise* hsame *transport for the source window, RegSeqRat readback, dyadic ledger, continuous-map modulus, image window, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each image precision request through $M, D, S, R, J, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyContinuousImage}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuousImage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, J, E, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyContinuousImageUp.TasteGate.CauchyContinuousImageTasteGate single carrier alignment`

Theorem 1775.2 (CauchyContinuousImage NameCert obligations)**.** *For every accepted* $\mathsf{CauchyContinuousImage}^{\uparrow}$ *carrier*

$$I = (S, R, D, M, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyContinuousImage}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName source window, RegSeqRat readback, DyadicRatCore tolerance ledger, ContinuousMap modulus row, image window, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *source-readback exactness, requiring S and R to expose only the displayed regular Cauchy source data before any image row is available;*
3. *tolerance compatibility, requiring D to be the shared dyadic ledger consumed by the source readback, the continuous-map modulus, and the image-window computation;*
4. *modulus exactness, requiring M to answer each requested image tolerance by a displayed source tolerance before J or E may be read;*
5. *image-seal nonescape, requiring J and E to remain downstream of S, R, D, M and to export no selected limit, quotient stream equality, host function extensionality, ambient Real completeness, or countable-choice selector;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No CauchyContinuousImage obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1775.1. The accepted packet displays exactly the source window S, regular readback R, dyadic ledger D, continuity modulus M, image window J, Real seal E, and structural rows H, C, P, N.

Source-readback exactness follows from the order of the displayed rows. A consumer first opens the finite StreamName window and then reads the RegSeqRat row attached to that window; the image side has no coordinate before this source route is fixed.

Tolerance compatibility and modulus exactness are the middle of the route. The DyadicRatCore ledger D supplies the common precision arithmetic, while M converts each requested output tolerance into a displayed source tolerance consumed by S and R before J is available.

The image-seal clause is terminal. The row J records only the finite image observations admitted by S, R, D, M, and E seals the resulting Real-facing datum only after those rows have been displayed.

Finally, H, C, P, N transport, replay, package, and name the same finite route. Since selected limits, quotient stream equality, host function extensionality, ambient Real completeness, and countable choice are absent from the carrier, no local obligation can export them. □

Theorem 1775.3 (CauchyContinuousImage tail stability)**.** *Let $I = (S, R, D, M, J, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyContinuousImage}^{\uparrow}$ *carrier. Every Real-facing consumer of the image seal factors through the finite tail route*

$$S \longmapsto R \longmapsto D \longmapsto M \longmapsto J \longmapsto E \longmapsto H, C, P, N.$$

Consequently replacing a source observation by a later observation inside the same displayed RegSeqRat tail changes no exported Real-facing image datum unless the replacement changes one of the displayed rows $S, R, D, M, J, E, H, C, P, N$. The consumer receives no selected limit, quotient stream equality, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, countable-choice selector, Lean verification, or bridge checking.*

Proof. Project the accepted carrier. The only Real-facing row is E, and Definition 1775.1 places it downstream of the source window, regular readback, dyadic ledger, continuity modulus, and image window.

A same-tail replacement inside the displayed RegSeqRat route preserves the rows S and R by hypothesis. The shared dyadic ledger D and the continuity modulus M therefore read the same source tolerance budget for every requested image tolerance.

The image window J is computed only from those displayed rows. If S, R, D, M are unchanged, then the finite image observations named by J and the terminal Real seal E are unchanged as exported packet coordinates.

The structural rows H, C, P, N preserve the same factorization under componentwise transport, displayed replay, package provenance, and local naming. No unlisted principle can distinguish the two tails, because selected limits, quotient stream equality, host function extensionality, ambient completeness, and countable choice are not coordinates of the carrier. □

Falsifiable prediction: Every $\mathsf{CauchyContinuousImage}^{\uparrow}$ image read factors through one StreamName source window, one RegSeqRat readback row, one DyadicRatCore tolerance ledger, one ContinuousMap modulus row, one finite image-window row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, continuousmap, real: $\mathsf{CauchyContinuousImage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source finite-window route, regular Cauchy readback, dyadic tolerance ledger, continuous-map modulus, finite image window, and Real-facing image seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuousImage}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContinuousImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1775.1, Definition 1775.2, and Definition 1775.3 seed the finite Cauchy-image route from a displayed source window through RegSeqRat readback, DyadicRatCore tolerance arithmetic, ContinuousMap modulus data, image-window computation, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContinuousImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-readback exactness, tolerance compatibility, modulus exactness, image-seal nonescape, transport, replay, provenance, and tail-stability factorization for BEDC.Derived.CauchyContinuousImageUp.

Constructive story: A $\mathsf{CauchyContinuousImage}^{\uparrow}$ packet is generated from a StreamName source window, RegSeqRat readback, DyadicRatCore tolerance ledger, ContinuousMap modulus row, finite image-window row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1776

A Concrete Naming Certificate for $\mathsf{BishopCompletionInduction}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopCompletionInduction}^{\uparrow}$ packet records the finite induction principle by which a proof or construction defined on a dense source is extended across the completion-facing Real route without invoking ambient completeness. It sits over the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the extension-and-uniqueness surface of chapter 397, the regular Cauchy universal route of chapter 1425, and the located completion boundary of chapter 1455. The packet gives Real and Completion consumers a visible dense-source extension route rather than a quotient stream, a selected host limit, countable choice, host function equality, or a hidden completeness theorem.

Definition 1776.1 (Bishop completion induction carrier)**.** *A* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (Q, F, D, R, E, U, K, H, C, P, N)$$

with the following displayed rows:

1. *Q is the dense-source row, naming the source data on which the induction step is first displayed.*
2. *F is the finite-window row, recording the* $\mathsf{StreamName}^{\uparrow}$ *observations inspected from the dense source and from the completion-facing name.*
3. *D is the dyadic tolerance ledger used to compare dense approximants, finite windows, and the terminal Real seal.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after Q, F, D, R have been displayed.*
6. *U is the extension face, storing the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *extension-and-uniqueness data read from the same source route.*
7. *K is the uniqueness ledger, requiring two extension reads to agree once they coincide on the displayed dense-source row and preserve the same finite-window and tolerance route.*
8. *H records componentwise* hsame *transport for dense source, finite windows, dyadic tolerance ledger, readback, Real seal, extension face, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each dense-source proof step through $Q, F, D, R, E, U, K, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, D, R, E, U, K, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopCompletionInductionUp.BishopCompletionInductionTasteGate single carrier alignment`

Theorem 1776.2 (Bishop completion induction NameCert obligations)**.** *For every accepted* BishopCompletionInduc *carrier*

$$I = (Q, F, D, R, E, U, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dense source, finite windows, dyadic tolerance ledger, RegSeqRat readback, Real seal, extension face, uniqueness ledger, transport, replay, provenance, and local naming rows;*

2. *dense-source visibility, requiring every induction premise to be displayed first on the source row Q;*

3. *finite-window discipline, requiring every completion-facing read to pass through the same* $\mathsf{StreamName}^{\uparrow}$ *windows F and dyadic tolerance ledger D;*

4. *readback and seal routing, requiring R to consume only F, D and requiring E to be terminal with respect to Q, F, D, R;*

5. *extension-face exactness, requiring U to read the sibling universal-property and regular Cauchy completion rows from the same route;*

6. *uniqueness-ledger exactness, requiring K to compare two extension reads only by their dense-source agreement and their shared finite-window, tolerance, readback, and seal rows;*

7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No Bishop completion-induction obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1776.1. The accepted packet exposes exactly the dense source Q, finite windows F, dyadic tolerance ledger D, regular readback R, Real seal E, extension face U, uniqueness ledger K, and structural rows H, C, P, N.

The dense-source and finite-window clauses are forced by the displayed order. A proof step has no source coordinate except Q, and any completion-facing observation must pass through the same finite windows F and tolerance ledger D before the readback row can be inspected.

The readback and seal clauses are downstream. The row R consumes only the finite-window and dyadic-tolerance route, and the terminal $\mathsf{Real}^{\uparrow}$ seal E is visible only after Q, F, D, R have been displayed.

The extension and uniqueness rows are then local. The extension face U reads the sibling universal-property data from the same route, while K compares two reads only through dense-source agreement and the shared finite route already present in the carrier.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the excluded objects are absent from the carrier, the displayed rows exhaust the local NameCert surface. □

Theorem 1776.3 (Bishop completion induction dense-source route)**.** *Every accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier routes a completion-facing proof step through the finite dense-source chain*

$$Q \longmapsto F \longmapsto D \longmapsto R \longmapsto E \longmapsto U \longmapsto K \longmapsto H, C, P, N.$$

Consequently the proof step receives a dense-source row, finite $\mathsf{StreamName}^{\uparrow}$ *windows, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, an extension face, a uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. It receives no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected host limit, countable-choice selector, host function equality, Lean verification, or bridge checking.*

Proof. Project the accepted carrier $I = (Q, F, D, R, E, U, K, H, C, P, N)$. The dense-source row Q is the only coordinate on which the induction premise can be opened.

The finite observation route is then fixed. The window row F stores the inspected StreamName observations, the dyadic ledger D budgets those observations, and the RegSeqRat row R is the only readback coordinate available before the terminal Real seal E.

After the seal is visible, the extension and uniqueness rows preserve the same route. The row U exposes the completion universal-property face, while K records agreement of extension reads only through the dense-source row and the displayed finite route.

The structural rows add no analytic coordinate. Thus every completion-facing proof step factors through the stated dense-source chain and cannot obtain any excluded completeness, quotient, choice, host-equality, Lean, or bridge datum. □

Lean4 ✓ `BEDC.Derived.BishopCompletionInductionUp.BishopCompletionInductionCarrier dense source route`

Theorem 1776.4 (Bishop completion induction extension principle)**.** *Let* $I = (Q, F, D, R, E, U, K, H, C, P, N)$ *be an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier. Any consumer that extends a construction or proof from the dense source to the completion-facing Real route must read* U *only after the dense-source route*

$$Q \longmapsto F \longmapsto D \longmapsto R \longmapsto E$$

has been displayed, and must read uniqueness through K *before exporting the result as a local* $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *claim. The exported result is therefore an extension along finite dense-source evidence, not an appeal to ambient completeness, quotient stream identity, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.*

Proof. Begin with an extension consumer of I. By Definition 1776.2, the consumer can open the extension face only inside the carrier inventory.

The source route comes first. The dense-source row Q displays the premise, the finite windows F and dyadic ledger D give the inspected completion-facing observations, and R records the regular rational readback before the Real seal E is available.

Only then can the extension face be read. The row U is downstream of the displayed dense-source route, so the consumer obtains extension data as a finite handoff from the same Q, F, D, R, E rows.

The uniqueness export is the final step. The row K compares extension reads only by dense-source agreement and their shared finite route, while H, C, P, N transport, replay, source, and name the same evidence. Since no excluded object is present in the carrier, the exported local NameCert claim has exactly the stated constructive extension form. □

Theorem 1776.5 (Bishop completion induction extension uniqueness)**.** *Let $I = (Q, F, D, R, E, U, K, H, C, P, N)$ be an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier. Suppose two extension faces are read through U and agree on the dense-source row Q, and suppose both preserve the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows F,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal E. Then the two faces induce the same Real seal route under the* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *packet, and any consumer export through H, C, P, N names a single local* $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *extension route.*

Proof. The dense-source agreement is visible inside the packet by Definition 1776.3. Both faces start from the same row Q, so their proof reads enter the completion-facing route through the same displayed source.

Apply the extension principle of Definition 1776.4. It permits an extension face to be read only after the common route Q, F, D, R, E has been displayed and requires the uniqueness ledger K before a local NameCert export is made.

The shared finite readback data identify the route consumed by the two faces. The window row F, tolerance ledger D, regular readback R, and Real seal E are the same for both reads, so transport exactness in Definition 1776.8 preserves the same componentwise hsame data.

Finally, consumer nonescape in Definition 1776.9 prevents either face from exporting an ambient completeness, quotient, choice, host-equality, Lean, or bridge datum that could distinguish the two routes outside the displayed packet. The local export through H, C, P, N therefore names one Real seal route under the $\mathsf{BishopCompletionInduction}^{\uparrow}$ carrier. □

Theorem 1776.6 (Bishop completion induction obligation closure)**.** *For every accepted* BishopCompletionInduction *carrier $I = (Q, F, D, R, E, U, K, H, C, P, N)$, the carrier definition, NameCert obligation theorem, dense-source route theorem, and extension principle together close the paper-side obligation package for carrier admission, dense-source visibility, finite-window discipline, readback and seal routing, extension-face exactness, uniqueness-ledger exactness, componentwise transport, displayed replay, provenance, and local naming. No obligation row for ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected host limits, countable choice, host function equality, Lean verification, or bridge checking is exported by this package.*

Proof. The carrier admission and row inventory are exactly Definition 1776.1. It displays $Q, F, D, R, E, U, K, H, C,$ and no additional analytic or verification coordinate.

The local obligation surface is supplied by Definition 1776.2. That theorem assigns dense-source visibility, finite-window discipline, readback and seal routing, extension-face exactness, uniqueness-ledger exactness, transport, replay, provenance, and local naming to the same rows.

The routing and export conditions are supplied by Definition 1776.3 and Definition 1776.4. Together they force every completion-facing proof step to pass through the displayed dense-source chain before the extension and uniqueness rows can be read.

Since each cited result uses only the displayed carrier rows and each excluded object is absent from those rows, the four cited sites form the claimed paper-side obligation package. □

Theorem 1776.7 (Bishop completion induction scoped kernel route)**.** *Every accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier read, extension read, uniqueness comparison, classifier transport, and consumer handoff is*

scoped to the displayed kernel surface

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniver}$

No route in the packet factors through an unlisted completion principle or a host-level equality service.

Proof. Project the carrier of Definition 1776.1. The finite-window row F, dyadic tolerance row D, RegSeqRat readback row R, Real seal E, extension face U, uniqueness ledger K, and structural rows H, C, P, N name exactly the displayed kernel objects.

Definition 1776.2 keeps each NameCert obligation on those same rows. In particular, dense-source visibility begins at Q, finite-window discipline uses F, D, readback and seal routing uses R, E, and extension-face exactness uses U, K.

Definition 1776.3 and Definition 1776.4 then fix the consumer path. A proof step reaches extension and uniqueness only after the finite dense-source chain has been displayed.

Thus the packet has no route through an unlisted completion theorem, quotient stream service, host equality service, or choice principle. The scoped kernel surface is exactly the one stated. □

Theorem 1776.8 (Bishop completion induction transport exactness)**.** *For an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier* $I = (Q, F, D, R, E, U, K, H, C, P, N)$*, classifier transport is exactly componentwise* hsame *on the displayed carrier rows together with displayed* Cont *replay of the same dense-source route. Transport may preserve dense-source rows, finite windows, dyadic tolerances, readback, Real seal, extension face, uniqueness ledger, provenance, and local naming; it may not replace them by ambient Real completeness, quotient stream equality, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.*

Proof. The transport coordinates are named in Definition 1776.1: row H records componentwise hsame for the carrier rows, and row C records displayed Cont replay from each dense-source proof step through the same packet.

Definition 1776.2 makes those transport and replay rows part of the local NameCert surface. It does not provide a second classifier coordinate or an external equality principle.

The dense-source route theorem fixes what H and C may preserve. They transport and replay $Q, F, D, R, E, U, K, P, N$ along the displayed route, not along an ambient completion or quotient-stream interface.

Therefore classifier transport is exactly componentwise hsame plus displayed Cont replay for the accepted packet, with all excluded replacement routes outside the carrier. □

Theorem 1776.9 (Bishop completion induction consumer nonescape)**.** *No consumer of an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier may export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, countable choice, host function equality, Lean verification, or bridge checking. The consumer may export only the finite dense-source extension route named by* $Q, F, D, R, E, U, K, H, C,$ *and the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *evidence attached to those rows.*

Proof. Start with the carrier inventory of Definition 1776.1. The tuple contains rows for dense source, finite windows, dyadic tolerance, RegSeqRat readback, Real seal, extension face, uniqueness ledger, transport, replay, provenance, and local naming only.

The NameCert obligations of Definition 1776.2 exhaust the accepted packet. Each allowed export is tied to one of the displayed rows, and no excluded row is introduced by the obligation theorem.

The extension principle Definition 1776.4 forces every consumer to read U after Q, F, D, R, E and to read uniqueness through K before a local NameCert claim is exported.

Thus any exported consumer result remains inside the finite dense-source route. The listed ambient completeness, quotient, choice, host-equality, Lean, and bridge objects are absent from the carrier and cannot escape through the local packet. □

Theorem 1776.10 (Bishop completion induction regular-Cauchy route)**.** *Any extension consumer of an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier* $I = (Q, F, D, R, E, U, K, H, C, P, N)$ *must read* Q, F, D, R, E *in order before* U *and* K *may export the completion-facing extension and uniqueness rows. Here* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the displayed finite windows and dyadic tolerance ledger, and* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route refuses ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, countable choice, host function equality, Lean verification, and bridge checking.*

Proof. Project the accepted carrier of Definition 1776.1. The dense source Q, finite windows F, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal E are the only rows available before the extension face and uniqueness ledger.

Apply the NameCert obligations of Definition 1776.2. They require dense-source visibility, finite-window discipline, and readback and seal routing before any extension-face exactness or uniqueness-ledger exactness can be consumed.

The dense-source route of Definition 1776.3 fixes the displayed order from Q through E and then to U, K, H, C, P, N. The extension principle of Definition 1776.4 then allows U and K to export the completion-facing extension only after the same route has been displayed.

Since the carrier and the cited route theorems contain no ambient completeness, quotient-stream, selected-limit, countable-choice, host-function-equality, Lean-verification, or bridge-checking coordinate, the regular-Cauchy handoff is exactly the finite route stated. □

Theorem 1776.11 (Bishop completion induction limit-seal factorization)**.** *Let* $I = (Q, F, D, R, E, U, K, H, C, P, N$ *be an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier. Every completion-facing* $\mathsf{Real}^{\uparrow}$ *seal read factors through*

$$Q \longmapsto F \longmapsto D \longmapsto R \longmapsto E$$

before either U *or* K *may be consumed. Thus the terminal seal route is a finite dense-source, StreamName-window, DyadicRatCore-budget, RegSeqRat-readback, Real-seal factorization. It imports no quotient stream, selected host limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier of Definition 1776.1. The rows before the extension face are exactly the dense-source row Q, finite-window row F, dyadic tolerance ledger D, regular-rational readback R, and terminal Real seal E.

Use the dense-source route of Definition 1776.3. It fixes the consumer path through Q, F, D, R, E before any extension, uniqueness, transport, replay, provenance, or local naming row is exported.

Apply the extension principle of Definition 1776.4 and the regular-Cauchy route of Definition 1776.10. These results permit U and K only after the same displayed route has exposed the RegSeqRat readback and terminal Real seal.

Finally invoke consumer nonescape from Definition 1776.9. Since quotient streams, selected host limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no hidden coordinate can bypass the factorization. □

Theorem 1776.12 (Bishop completion induction finite-window recursion)**.** *Let $I = (Q, F, D, R, E, U, K, H, C, P, N$ be an accepted* $\mathsf{BishopCompletionInduction}^{\uparrow}$ *carrier. Every dense-source recursive step exported from I must first expose the* $\mathsf{StreamName}^{\uparrow}$ *window F, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *handoff inside U, the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *route inside U, the* $\mathsf{BishopLocatedCompletionBoundary}^{\uparrow}$ *boundary row named by P, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. Only after these rows are displayed may the step be exported through K, H, C, P, N as a local* $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$ *induction claim.*

Consequently Bishop completion induction transports dense-source proof data through finite StreamName windows, regular rational readback, dyadic tolerance control, completion-universal handoff, regular-Cauchy routing, located-boundary provenance, and Real sealing. It does not export a quotient stream, a selected host limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host function equality, Lean verification, or bridge checking.*

Proof. Inspect the accepted carrier rows in Definition 1776.1. The dense-source row Q is the only row on which the recursive premise is initially visible, while F, D, R, E are the displayed finite-window, tolerance, readback, and terminal seal rows available before any extension export.

The dense-source and finite-window route is fixed by Definition 1776.3. A recursive step cannot jump directly from Q to a completion-facing result; it must pass through the StreamName window F, the DyadicRatCore ledger D, the RegSeqRat readback R, and the Real seal E.

The extension handoff is then governed by Definition 1776.4, Definition 1776.10, and Definition 1776.11. These results place the $\mathsf{CompletionUniversalProperty}^{\uparrow}$ and $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ data in U downstream of Q, F, D, R, E, with the located-boundary provenance named in the package row P rather than supplied by an external completion principle.

Finally apply uniqueness and nonescape. The uniqueness ledger K compares exports only along the shared dense-source and finite-window route, while H, C, P, N transport, replay, source, and name the same rows. By Definition 1776.5 and Definition 1776.9, no quotient stream, selected host limit, countable-choice selector, ambient completeness theorem, host function equality, Lean verification, or bridge datum can replace the displayed recursion route. □

Falsifiable prediction: Every $\mathsf{BishopCompletionInduction}^{\uparrow}$ completion-facing proof read factors through one dense-source row, one finite-window row, one DyadicRatCore tolerance ledger, one RegSeqRat readback row, one terminal RealUp seal, one extension face, one uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, completionuniversalproperty, regular-cauchy-completion-universal, bishoplocatedcompletionboundary: $\mathsf{BishopCompletionInduction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a dense-source induction row, finite windows, dyadic tolerances, readback, terminal seal, extension face, and uniqueness ledger for proof transport across completion-facing Real data.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1776.1, Definition 1776.2, Definition 1776.3, Definition 1776.4, Definition 1776.6, Definition 1776.7, Definition 1776.8, and Definition 1776.9 seed the finite dense-source induction route through StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, extension face, uniqueness ledger, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected host limits, countable choice, host function equality, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed carrier admission, dense-source visibility, finite-window discipline, readback and seal routing, extension-face exactness, uniqueness-ledger exactness, transport, replay, provenance, consumer nonescape, and extension-principle factorization to the listed L10 kernel scope for BEDC.Derived.BishopCompletionInductionUp.

Constructive story: A BishopCompletionInduction$^{\uparrow}$ packet is generated from a dense-source row, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, a CompletionUniversalProperty and RegularCauchyCompletionUniversal extension face, a uniqueness ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1777

A Concrete Naming Certificate for $\mathsf{ContinuedFractionConvergents}^{\uparrow}$

Following Perron, Khinchin, and the Hardy–Wright treatment of continued fractions, the ContinuedFractionConverge packet names finite convergent data before any completion-facing consumer reads it as a regular rational sequence or as a real-name seal. It refines the partial-quotient surface of chapter 342 by separating the convergent numerator and denominator rows, the rational normalization rows of chapter 37, the finite-window route of chapter 126, the regular-rational handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet is a finite approximation certificate between continued fractions and Real or Completion routes; it is not a completeness theorem and not a quotient equality principle.

Definition 1777.1 (ContinuedFractionConvergents carrier)**.** *A* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, N, D, Q, W, E, R, S, H, C, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *partial-quotient source row, including the finite prefix of partial quotients and the local recurrence ledger that names which prefix is being read.*
2. *N and D are the* $\mathsf{Rat}^{\uparrow}$ *numerator and denominator convergent rows. They display the finite recurrence steps and the denominator-nonzero side condition needed to read each convergent as a rational endpoint.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *normalized rational-convergent row obtained from N and D by the displayed rational arithmetic route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the convergent indices requested by the downstream consumer.*
5. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-ledger row. It records the dyadic radius, endpoint enclosure, and precision budget attached to each displayed finite window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row built from Q, W, E and the recurrence ledger of A, N, D.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational row R has been displayed.*
8. *H records componentwise* hsame *transport for partial quotients, numerator rows, denominator rows, normalized rationals, stream windows, dyadic ledgers, regular handoff, the Real seal, provenance, and local naming.*

9. *C records displayed* Cont *replay from each finite window request through* $A, N, D, Q, W, E, R, S, H, P, L$.
10. *P is the* Pkg *provenance row over* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{ContinuedFractionConvergents}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite continued-fraction tail, host rational equality, quotient real equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ContinuedFractionConvergentsUp.ContinuedFractionConvergentsTasteGate`
`single carrier alignment`

Theorem 1777.2 (ContinuedFractionConvergents NameCert obligations)**.** *For every accepted* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *carrier* $K = (A, N, D, Q, W, E, R, S, H, C, P, L)$, *the local* $\mathsf{NameCert}_{\mathsf{ContinuedFractionCo}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier admission for the displayed partial-quotient source, rational numerator and denominator rows, normalized rational convergent row, finite stream windows, dyadic error ledger, regular handoff, Real seal, and structural rows* H, C, P, L;
2. *recurrence exactness, requiring each displayed numerator and denominator row to read the finite recurrence ledger of* A *before the rational row* Q *is available;*
3. *rational-window compatibility, requiring* Q *and* W *to expose the same finite convergent indices and no unrequested tail data;*
4. *dyadic error compatibility, requiring* E *to be attached to the same finite windows as* W *and to supply the precision budget consumed by* R;
5. *Real-route nonescape, requiring* S *to remain terminal with respect to* A, N, D, Q, W, E, R *and to expose no detached real object;*
6. *componentwise transport, replay, provenance, and naming, requiring* H, C, P, L *to preserve only the displayed route under* hsame *transport and* Cont *replay.*

No local obligation is stored outside these six rows.

Lean4 ✓ `BEDC.Derived.ContinuedFractionConvergentsUp.ContinuedFractionConvergentsNameCertObligations`

Proof. Carrier admission is projection from Definition 1777.1. The visible packet contains the partial-quotient source, the rational numerator and denominator rows, the normalized rational convergent row, finite windows, dyadic error data, regular handoff, Real seal, and the structural rows H, C, P, L.

Recurrence exactness reads the source row A before N and D are accepted. The rational row Q is therefore downstream of the finite recurrence ledger, not a detached rational sequence.

Window and dyadic compatibility read W and E over the same finite index set. The stream row exposes only requested convergent windows, and the dyadic row supplies the precision budget that the regular handoff row R consumes.

The Real-route clause follows from the placement of S after R. Since the carrier has no separate coordinate for completed tails, quotient equality, or selected limits, H, C, P, L can transport and name only the displayed finite route. □

Theorem 1777.3 (ContinuedFractionConvergents RegSeqRat handoff)**.** *Let $K = (A, N, D, Q, W, E, R, S, H, C, P,$ be an accepted* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer reads the packet through the displayed route*

$$A, N, D \longmapsto Q \longmapsto W, E \longmapsto R \longmapsto H, C, P, L.$$

Consequently the consumer receives finite continued-fraction prefixes, rational numerator and denominator rows, normalized rational convergents, finite stream windows, dyadic error ledgers, componentwise hsame *transport,* Cont *replay, package provenance, and the local NameCert row before any regular-rational data is available. The route cannot use quotient real equality, a choice-selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$ read of K. By Definition 1777.1, the only regular rational coordinate is R, and it is downstream of Q, W, E together with the recurrence ledger carried by A, N, D.

The rational part of the route is fixed by N, D, Q. The numerator and denominator rows expose the finite recurrence data, while Q records the normalized rational convergent surface read through chapter 37.

The window and precision part of the route is fixed by W and E. The finite stream surface of chapter 126 chooses the displayed windows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies the error budget consumed by the regular-rational row of chapter 301.

Finally, H, C, P, L transport, replay, package, and name the same route. Since the excluded objects have no coordinate in the carrier, every handoff to $\mathsf{RegSeqRat}^{\uparrow}$ factors through the displayed finite convergent packet. □

Theorem 1777.4 (ContinuedFractionConvergents Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *carrier factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R before it reaches the Real seal row S. The boundary exposes the same finite partial-quotient source, rational convergent rows, finite windows, dyadic error ledger, transport, replay, provenance, and local naming; it does not assert full real completeness, quotient equality of real names, or a choice-selected limit of the convergent sequence.*

Proof. Read the Real-facing coordinate of Definition 1777.1. The seal row is S, and the definition places S after the regular handoff row R.

Apply Definition 1777.3. The handoff row already factors through the finite partial-quotient source, rational numerator and denominator rows, normalized rational convergents, stream windows, dyadic error ledgers, and structural rows.

Thus any Real-facing consumer receives the same finite route before the seal is visible. No coordinate can bypass R to provide quotient equality, choice-selected limits, or ambient completeness for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuedFractionConvergents}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ packet is generated from a ContinuedFraction partial-quotient source, Rat numerator and denominator convergent rows, StreamName finite windows, a DyadicRatCore error ledger, RegSeqRat handoff, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ContinuedFractionConvergents}^{\uparrow}}$, seed)

Scope closed: Definition 1777.1, Definition 1777.2, Definition 1777.3, and Definition 1777.4 seed the finite continued-fraction convergent route consumed by RegSeqRat and Real packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ContinuedFractionConvergents}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full real completeness, quotient equality of real names, choice-selected limits of convergent sequences, total continued-fraction expansion for every real, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recurrence exactness, rational-window compatibility, dyadic error compatibility, Real-route nonescape, componentwise transport, provenance, and local naming for the same displayed packet.

Chapter 1778

A Concrete Naming Certificate for $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ packet records the finite closed-interval route by which endpoint bounds, interval nets, stream windows, regular rational readbacks, dyadic tolerance ledgers, and the Bishop regular Cauchy completion boundary are assembled before a Real-facing seal is consumed. It sits over the closed bounded interval net of chapter 1250, the finite observation surface of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, the Bishop regular Cauchy completion packet of chapter 1639, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a finite closed-interval completion surface; it does not import Heine–Borel compactness, open-cover compactness, selected limits, quotient streams, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1778.1 (Closed interval Cauchy completion carrier)**.** *A* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, I, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower closed endpoint row and B is the upper closed endpoint row, both read through the same finite endpoint discipline as the source closed interval.*
2. *I is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row, recording the finite interval net, mesh scale, endpoint containment route, and interval-local coverage data consumed by the completion request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the interval-net request before any regular rational readback is exposed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed StreamName windows and endpoint-bounded interval route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radius, endpoint comparison budget, common denominator budget, and tail allowance used by the interval Cauchy read.*
6. *Q is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *handoff row, consuming A, B, I, S, R, D before any Real-facing completion boundary is visible.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the endpoint rows, interval-net row, finite windows, regular readback, dyadic tolerance ledger, and Bishop completion handoff have been displayed.*

8. *H records componentwise* hsame *transport for endpoints, interval-net data, StreamName windows, RegSeqRat readbacks, dyadic tolerance ledgers, Bishop completion handoff, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval completion request through $A, B, I, S, R, D, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, I, S, R, D, Q, E, H, C, P, N$. The carrier has no coordinate for open-cover compactness, selected limits, quotient streams, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalCauchyCompletionUp.ClosedIntervalCauchyCompletionTasteGate`
`single carrier alignment`

Theorem 1778.2 (Closed interval Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *carrier*

$$K = (A, B, I, S, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower endpoint, upper endpoint, ClosedBoundedIntervalNet row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Bishop regular Cauchy completion handoff, Real seal, transport, replay, provenance, and local naming rows;*

2. *endpoint-bounded interval admission, requiring the interval-net row I to be read from the displayed endpoint rows A, B rather than from a host closed interval;*

3. *finite-window locality, requiring S to expose exactly the stream observations opened by the interval-net request and the dyadic tolerance ledger D;*

4. *regular readback locality, requiring R to be read only from the displayed StreamName windows and endpoint-bounded interval route;*

5. *completion handoff ordering, requiring Q to consume A, B, I, S, R, D before E exposes the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*

6. *componentwise classifier transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N;*

7. *ledger non-escape, requiring the packet to export no open-cover compactness, selected limit, quotient stream, countable-choice schedule, trichotomy branch, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No closed-interval Cauchy completion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1778.1. The visible packet consists exactly of the two endpoint rows A, B, the interval-net row I, finite windows S, regular readback R, dyadic tolerance ledger D, Bishop completion handoff Q, terminal seal E, and structural rows H, C, P, N.

Endpoint-bounded interval admission is upstream of every completion read. The row I is the only interval-net coordinate, and it is attached to the displayed endpoints A, B, so a consumer cannot replace it by a host closed interval or by an unrecorded compactness theorem.

Finite-window and regular readback locality are the middle route. The StreamName row S opens exactly the observations allowed by I and D, and the RegSeqRat row R reads only those displayed windows before any completion handoff is available.

The completion handoff then follows the displayed order. The row Q consumes A, B, I, S, R, D, and the terminal Real seal E is exposed only after that handoff has been displayed.

Finally, H, C, P, N transport, replay, package, and name the same finite route. Since none of the excluded objects occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1778.3 (Closed interval Cauchy completion route)**.** *Every Real or Completion consumer that reads an accepted* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *carrier factors its request through the finite route*

$$A, B \longmapsto I \longmapsto S \longmapsto R \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives displayed closed endpoints, a ClosedBoundedIntervalNet row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Bishop regular Cauchy completion handoff, a terminal RealUp seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. The route supplies no open-cover compactness, selected limit, quotient stream, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $K = (A, B, I, S, R, D, Q, E, H, C, P, N)$. The endpoint rows A, B and the interval-net row I are the only closed-interval source coordinates available to a consumer.

The finite observation route is then forced by the packet. The StreamName row S opens the window selected by the interval-net request, and the RegSeqRat row R reads back that window inside the displayed endpoint bounds.

The tolerance and completion boundary are downstream. The DyadicRatCore row D records the dyadic radius and comparison budget, the Bishop regular Cauchy completion row Q consumes the endpoint, net, window, readback, and tolerance rows, and the Real seal E is reached only after that handoff.

The remaining rows preserve the same route under componentwise hsame transport and displayed Cont replay, with P and N recording provenance and local naming. Since no excluded compactness, selection, quotient, choice, trichotomy, host-equality, or ambient-completeness coordinate is present, every public read is exhausted by the displayed route. □

Theorem 1778.4 (Closed interval Cauchy completion finite-net Real seal)**.** *For every accepted* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *carrier*

$$K = (A, B, I, S, R, D, Q, E, H, C, P, N),$$

every Real-facing read reaches the terminal seal row E only after the lower endpoint row A, upper endpoint row B, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row I,* $\mathsf{StreamName}^{\uparrow}$ *finite windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *handoff Q have been displayed. No open-cover compactness, selected limit, quotient stream, countable-choice*

schedule, trichotomy branch, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate is available to the read.*

Proof. Start with the carrier definition. It lists the terminal Real seal as the single row E, and every row preceding it in the completion route is explicitly displayed as A, B, I, S, R, D, Q.

Apply the NameCert obligations. Endpoint-bounded interval admission fixes A, B, I, finite-window locality fixes S, regular readback locality fixes R, and completion handoff ordering fixes D, Q before E can be exposed.

Apply the route theorem. Every Real or Completion consumer factors through the ordered path from endpoints to interval net, windows, readback, tolerance ledger, Bishop handoff, and only then the Real seal.

The excluded objects have no carrier coordinate and no NameCert obligation row. Thus the finite-net-to-Real-seal boundary is exactly the displayed route and cannot leak open-cover compactness, selected limits, quotient streams, countable choice, trichotomy, host real equality, or ambient Real completeness. □

Theorem 1778.5 (Closed interval Cauchy completion finite-net induction)**.** *For every accepted* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *packet*

$$K = (A, B, I, S, R, D, Q, E, H, C, P, N),$$

each inspected finite interval-net stage is exposed through the same displayed route

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BishopRegularCauchyCompl}$$

before the terminal seal row E is read. The induction over displayed net stages uses only the carrier rows $I, S, R, D, Q, E, H, C, P, N$, the route theorem, and the finite-net Real seal theorem; it exports no open-cover compactness, selected limit, quotient stream, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct over the finite list of net stages displayed in the row I of Definition 1778.1. The base stage is the first inspected closed bounded interval net cell. Its only available observations are the interval-net row I, the StreamName windows S, the regular readback R, and the dyadic tolerance ledger D.

For the induction step, assume the route has been exposed for all earlier displayed stages. The next stage is another row of the same finite interval-net ledger I, so the consumer again opens only the windows recorded by S, reads them back through R, and accounts for the tolerance through D.

Apply the route theorem to the stage just inspected. It places the Bishop regular Cauchy completion handoff Q after I, S, R, D and before the Real seal E.

Finally apply the finite-net Real seal theorem. It prevents the terminal seal from being read before the displayed interval-net, window, readback, tolerance, and completion rows have appeared, and the structural rows H, C, P, N transport, replay, source, and name the same finite induction route. □

Falsifiable prediction: Every $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ read factors through displayed closed endpoints, one ClosedBoundedIntervalNet row, finite StreamName windows, one RegSeqRat readback row, one DyadicRatCore tolerance ledger, one BishopRegularCauchyCompletion handoff, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closedboundedintervalnet, streamname, regseqrat, dyadicratcore, bishop-regular-cauchy-completion, real: $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint-bounded interval-net data, finite observation windows, regular rational readback, dyadic tolerance accounting, the Bishop completion handoff, and the terminal Real seal as one closed-interval completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1778.1, Definition 1778.2, and Definition 1778.3 seed the finite closed-interval Cauchy completion route from displayed endpoints and interval-net rows through StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Bishop regular Cauchy completion handoff, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient streams, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-bounded interval admission, finite-window locality, regular readback locality, completion handoff ordering, classifier transport, replay, provenance, local naming, and route factorization for BEDC.Derived.ClosedIntervalCauchyCompletionUp.

Constructive story: A $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ packet is generated from displayed closed endpoint rows, one ClosedBoundedIntervalNet row, finite StreamName windows, one RegSeqRat readback row, one DyadicRatCore tolerance ledger, one BishopRegularCauchyCompletion handoff, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1779

A Concrete Naming Certificate for $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ packet names the finite closed-interval criterion by which a uniformly Cauchy family of interval-local real windows is admitted before any uniform-limit endpoint or completion-facing seal is read. It sits after the closed-interval completion route of chapter 1778, uses the shared threshold surface of chapter 435, hands off to the uniform-limit packet of chapter 463, and keeps its finite observations inside the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling surfaces of chapter 126, chapter 301, and chapter 38. The chapter records a criterion packet for closed-interval Cauchy data; it does not import open-cover compactness, selected limits, quotient function-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1779.1 (Closed interval Cauchy criterion carrier)**.** *A* $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, I, F, W, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower closed endpoint row and B is the upper closed endpoint row, both read through the same finite closed-interval discipline as the source interval.*
2. *I is the closed interval row that records endpoint containment, interval-local request scope, and the closed-interval completion handoff consumed from* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$*.*
3. *F is the function-window family, recording the finite interval-local real-valued windows visible to the criterion.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by F for the finite family of interval requests.*
5. *T is the uniform Cauchy threshold row, a single shared threshold surface from* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *for every displayed tolerance and every retained interval-local window.*
6. *S is the criterion ledger asserting that all displayed pairs of windows beyond T are compared through the same interval row I and the same function-window family F.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W, T, S before any limit or completion-facing consumer reads the terminal row.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, F, W, T, S, R have been displayed.*

9. *H records componentwise* hsame *transport for endpoints, the closed-interval row, function-window rows, StreamName windows, uniform Cauchy threshold, criterion ledger, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a closed-interval Cauchy criterion request through $A, B, I, F, W, T, S, R, E, H$*
11. *P is the* Pkg *provenance over* ClosedIntervalCauchyCriterion$^{\uparrow}$, ClosedIntervalCauchyCompletion$^{\uparrow}$, UniformCauchyCriterion$^{\uparrow}$, UniformLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{ClosedIntervalCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, I, F, W, T, S, R, E, H, C, P, N$. The carrier has no coordinate for open-cover compactness, selected limits, quotient function-space equality, countable choice, host function equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalCauchyCriterionUp.ClosedIntervalCauchyCriterionTasteGate`
`single carrier alignment`

Theorem 1779.2 (Closed interval Cauchy criterion NameCert obligations)**.** *For every accepted* ClosedIntervalCauchyCriterion$^{\uparrow}$ *carrier*

$$J = (A, B, I, F, W, T, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{ClosedIntervalCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint rows, closed-interval row, function-window family, StreamName windows, uniform Cauchy threshold, criterion ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *closed-interval anchoring, requiring every criterion read to use the displayed row I obtained from the* ClosedIntervalCauchyCompletion$^{\uparrow}$ *route rather than an ambient compact interval object;*
3. *function-window locality, requiring F and W to expose exactly the finite interval-local windows inspected by the criterion request;*
4. *uniform-threshold exactness, requiring the single threshold row T from* UniformCauchyCriterion$^{\uparrow}$ *to govern all retained displayed window pairs;*
5. *readback ordering, requiring R and the terminal* Real$^{\uparrow}$ *seal E to be exposed only after I, F, W, T, S have been displayed;*
6. *componentwise classifier transport, displayed replay, provenance, and local naming through H, C, P, N;*
7. *ledger non-escape, requiring the packet to export no open-cover compactness, selected limit, quotient function-space equality, countable-choice schedule, host function equality, or ambient* Real$^{\uparrow}$ *completeness theorem.*

No closed-interval Cauchy criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1779.1. The visible packet consists exactly of the endpoint rows A, B, closed-interval row I, function-window family F, StreamName windows W, threshold row T, criterion ledger S, readback row R, terminal seal E, and structural rows H, C, P, N.

Closed-interval anchoring is upstream of every criterion read. The row I is the only closed-interval coordinate, and it is supplied by the $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ sibling route, so no host compact interval object can replace it.

Function-window locality and uniform-threshold exactness are the middle route. The rows F and W expose the finite interval-local windows, while T is the single shared $\mathsf{UniformCauchyCriterion}^{\uparrow}$ threshold governing all displayed window pairs recorded by S.

The readback order is then forced. The $\mathsf{RegSeqRat}^{\uparrow}$ row R and the terminal $\mathsf{Real}^{\uparrow}$ seal E occur only after the closed-interval, window, threshold, and criterion-ledger rows have been displayed.

Finally, H, C, P, N transport, replay, package, and name the same finite route. Since none of the excluded objects occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1779.3 (Closed interval Cauchy criterion uniform-window handoff)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$ *or completion-facing consumer that reads an accepted* $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ *carrier factors its request through the finite route*

$$A, B \longmapsto I \longmapsto F, W \longmapsto T, S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives displayed closed endpoints, the closed-interval completion row, interval-local function windows, StreamName windows, one shared UniformCauchyCriterion threshold, one criterion ledger, one RegSeqRat readback row, one RealUp seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. The handoff supplies no open-cover compactness, selected limit, quotient function-space equality, countable choice, host function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $J = (A, B, I, F, W, T, S, R, E, H, C, P, N)$. The endpoint rows A, B and the closed-interval row I are the only interval source coordinates available to the consumer.

The finite observation route then opens through F and W. These rows expose only interval-local function windows and their displayed $\mathsf{StreamName}^{\uparrow}$ readings.

Uniformity is supplied before any readback. The threshold row T is shared across all retained windows, and the criterion ledger S records the displayed pairwise comparisons under that same threshold.

The downstream handoff reaches R and then E. Thus a $\mathsf{UniformLimit}^{\uparrow}$ or completion-facing consumer reads the $\mathsf{RegSeqRat}^{\uparrow}$ route and the terminal $\mathsf{Real}^{\uparrow}$ seal only after the closed-interval, window, and uniform-threshold rows are in scope.

The structural rows H, C, P, N preserve and name the same route. Because the carrier has no coordinate for the excluded compactness, selection, quotient, choice, host-equality, or completeness objects, every public read is exhausted by the displayed handoff. □

Falsifiable prediction: Every $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ read factors through displayed closed endpoints, one closed-interval completion row, interval-local function windows, StreamName windows, one UniformCauchyCriterion threshold, one criterion ledger, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closed-interval-cauchy-completion, uniformcauchycriterion, uniformlimit, streamname, regseqrat, real: $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries closed-interval completion anchoring, interval-local function windows, a shared uniform Cauchy threshold, the criterion ledger, regular rational readback, and the terminal Real seal as one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCriterion}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCriterion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 1779.1, Definition 1779.2, and Definition 1779.3 seed the finite closed-interval Cauchy criterion route from displayed endpoints and closed-interval completion rows through interval-local function windows, StreamName windows, a shared UniformCauchyCriterion threshold, criterion ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalCauchyCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient function-space equality, countable choice, host function equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-interval anchoring, function-window locality, uniform-threshold exactness, readback ordering, classifier transport, replay, provenance, local naming, and uniform-window handoff for BEDC.Derived.ClosedIntervalCauchyCriterionUp.

Constructive story: A $\mathsf{ClosedIntervalCauchyCriterion}^{\uparrow}$ packet is generated from displayed closed endpoint rows, one closed-interval completion row, interval-local function windows, finite StreamName windows, one UniformCauchyCriterion threshold row, one criterion ledger, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1780

A Concrete Naming Certificate for $\mathsf{RationalCauchyGap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalCauchyGap}^{\uparrow}$ packet records a finite rational Cauchy-gap witness that must pass through the rational, dyadic, stream-window, regular-sequence, and real-name route before any Completion consumer treats it as a completed real. It sits over the rational source schedule of chapter 37, the late-pair dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The chapter supplies the current-state witness that rational Cauchy information is not already a completed real object: the completion route is visible as finite NameCert data rather than quotient equality, a selected limit, countable choice, or host completeness.

Definition 1780.1 (Rational Cauchy gap carrier)**.** *A* $\mathsf{RationalCauchyGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source schedule, recording the rational observations and precision requests that generate the candidate Cauchy gap.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *late-pair tolerance ledger, storing the dyadic bounds, slack rows, and comparison budgets used to test the source schedule at later stages.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by Q and budgeted by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, recording the regular rational readback obtained from the displayed windows and dyadic late-pair ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after Q, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the rational source schedule, dyadic tolerance ledger, finite windows, regular rational handoff, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the rational precision request through Q, D, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalCauchyGap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyGap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, W, R, E, H, C, P, N$. It has no coordinate for quotient equality of rational streams, a selected real limit, countable choice, host real comparison, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RationalCauchyGapUp.RationalCauchyGapTasteGate single carrier alignment`

Theorem 1780.2 (Rational Cauchy gap NameCert obligations)**.** *For every accepted* $\mathsf{RationalCauchyGap}^{\uparrow}$ *carrier*

$$G = (Q, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCauchyGap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{Rat}^{\uparrow}$ *source schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *late-pair tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *rational-source exactness, requiring Q to display the rational observations and precision requests that the gap read is allowed to inspect;*
3. *late-pair tolerance exactness, requiring D to store the dyadic bounds and slack rows used when later elements of the rational schedule are compared;*
4. *finite-window compatibility, requiring W to expose only the StreamName observations opened by the source schedule and budgeted by the late-pair ledger;*
5. *regular handoff exactness, requiring R to consume Q, D, W before any Real seal is visible;*
6. *Real-seal nonescape, requiring E to be terminal with respect to the route Q, D, W, R and to export no quotient stream equality, selected limit, countable-choice schedule, host comparison, ambient completeness theorem, Lean verification, or bridge data;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No rational Cauchy-gap obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1780.1. The accepted packet displays exactly the rational source schedule, dyadic late-pair tolerance ledger, finite windows, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming rows.

The source and tolerance clauses are then forced by the route. The row Q is the only coordinate that carries rational observations and precision requests, while D is the only coordinate that carries the late-pair dyadic comparison budget.

The finite-window and regular handoff clauses are downstream. The row W reads only the observations opened by Q and budgeted by D, and R is the first row that repacks those finite observations as regular rational readback.

Finally, E, H, C, P, N seal, transport, replay, and name the same finite route. Since quotient stream equality, selected limits, countable choice, host comparison, ambient completeness, Lean verification, and bridge data are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1780.3 (Rational Cauchy gap Real handoff)**.** *Every accepted rational Cauchy gap read that reaches a* $\mathsf{Real}^{\uparrow}$ *consumer must factor through the finite route*

$$Q \longmapsto D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives the rational source schedule, dyadic late-pair tolerance ledger, finite StreamName windows, regular rational handoff, terminal Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives no quotient equality of rational streams, selected limit, countable-choice schedule, host real comparison, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Project the accepted carrier $G = (Q, D, W, R, E, H, C, P, N)$. The rational gap data starts in Q, and the only late-pair tolerance data available to the read is the dyadic ledger D.

The observation route is finite. The StreamName row W exposes the bounded windows licensed by Q and D, and the RegSeqRat row R is the first point where those windows are repacked as regular rational readback.

The Real consumer reaches the terminal seal only through E and then through the structural rows H, C, P, N. These rows preserve and replay the same route; they do not add quotient stream equality, a selected limit, countable choice, host real comparison, ambient completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every $\mathsf{RationalCauchyGap}^{\uparrow}$ read into Real or Completion factors through one RatUp source schedule, one DyadicRatCore late-pair tolerance ledger, one StreamName finite-window row, one RegSeqRat handoff row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalCauchyGap}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a rational source schedule, late-pair dyadic tolerance ledger, finite stream windows, regular rational readback, and the terminal RealUp seal that witnesses why completion is a genuine route rather than an already available rational object.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCauchyGap}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalCauchyGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1780.1, Definition 1780.2, and Definition 1780.3 seed the finite rational Cauchy-gap route through RatUp source schedule, DyadicRatCore late-pair tolerance ledger, StreamName finite windows, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalCauchyGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of rational streams, selected limits, countable choice, host real comparison, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-source exactness, late-pair tolerance exactness, finite-window compatibility, regular handoff exactness, Real-seal nonescape, transport, replay, provenance, and Real-consumer factorization for BEDC.Derived.RationalCauchyGapUp.

Constructive story: A $\mathsf{RationalCauchyGap}^{\uparrow}$ packet is generated from one RatUp source schedule, one DyadicRatCore late-pair tolerance ledger, one StreamName finite-window row, one RegSeqRat handoff row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1781

A Concrete Naming Certificate for $\mathsf{FastCauchyModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FastCauchyModulus}^{\uparrow}$ packet records the fast-threshold discipline that turns an ordinary Cauchy modulus into the doubled-window schedule expected by regular Cauchy and real-name consumers. It sits over the generic modulus schedule of chapter 292, the finite observation windows of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives downstream RealCauchyModulus and RegSeqRat consumers one visible fast-modulus carrier before any Real or Completion route reads a seal.

Definition 1781.1 (Fast Cauchy modulus carrier)**.** *A* $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{CauchyModulus}^{\uparrow}$ *row, read as a finite threshold schedule for Cauchy observations.*
2. *F is the fast or doubled threshold discipline. It records, for each displayed precision request, whether the carrier uses the direct fast threshold or the doubled-window threshold, and it stores the comparison row showing that the selected threshold is strong enough for the requested tail estimate.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the threshold selected by F from M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger consumed by the selected window and by the regularity check.*
5. *Q is the precision-query row whose dyadic request is read jointly by M, F, S, D.*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after M, F, S, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source modulus, fast discipline, stream window, dyadic ledger, precision query, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through $M, F, S, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, countable choice, trichotomy, selected limits, unbounded stream reads, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FastCauchyModulusUp.FastCauchyModulusUpTasteGate single carrier alignment`

Theorem 1781.2 (Fast Cauchy modulus doubling window)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. Every regular-Cauchy or Real-facing consumer that requests a precision row Q must read the finite route*

$$Q \longmapsto M, F \longmapsto S, D \longmapsto E \longmapsto H, C, P, N.$$

If F selects the direct fast threshold, the opened window S is already the thresholded finite window supplied by M. If F selects the doubled threshold, S is the displayed enlarged window and D records the dyadic slack showing that the doubled read implies the requested tail estimate. In both cases the consumer receives no unbounded stream read, quotient stream equality, host real equality, countable-choice schedule, trichotomy principle, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge data.*

Proof. Project the accepted carrier. The precision query Q can reach a stream window only through the displayed source modulus row M and the fast discipline row F, so the carrier exposes no separate threshold selector.

For the direct fast case, F records that the threshold read from M already satisfies the requested tail estimate. The window row S is therefore the finite StreamName window opened at that threshold, and the dyadic ledger D only records the requested tolerance account.

For the doubled case, F records the doubled threshold before any window is opened. The row S contains that enlarged finite window, and D stores the dyadic slack proving that the doubled read is strong enough for the original precision request.

Finally E, H, C, P, N hand the same displayed route to regular-Cauchy and Real-facing consumers. Since the excluded objects are absent from the carrier, no consumer can bypass the finite threshold-window-ledger path. □

Theorem 1781.3 (Fast Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FastCauchyModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source* $\mathsf{CauchyModulus}^{\uparrow}$ *row, fast threshold discipline,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, precision query,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, transport, replay, provenance, and local naming rows;*

2. *threshold exactness, requiring F to select either the direct fast threshold or the doubled-window threshold from the displayed source modulus row M for the same precision query Q;*

3. *finite-window exactness, requiring S to be precisely the finite StreamName window opened by the threshold selected in F;*

4. *dyadic slack exactness, requiring D to store the tolerance and slack row that makes the selected threshold sufficient for the requested tail estimate;*

5. *regular and Real handoff, requiring E to consume M, F, S, D, Q before exposing any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N;*
7. *refusal of quotient stream equality, host real equality, countable choice, trichotomy, selected limits, unbounded stream reads, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No Fast Cauchy modulus obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FastCauchyModulusUp.FastCauchyModulusTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1781.1. The visible packet consists exactly of the source modulus, fast threshold discipline, finite window, dyadic ledger, precision query, handoff, and structural rows.

Threshold exactness is finite precision induction on the displayed query row. At the base precision, F records the selected threshold directly from M; at the next displayed precision, the same row either preserves the fast threshold discipline or opens the doubled threshold together with its comparison row.

Finite-window exactness follows by reading S only after F has selected the threshold. The induction step transports the selected threshold into the StreamName window, and the dyadic ledger D records the slack needed to interpret that finite window at the requested precision.

The handoff row E is downstream of M, F, S, D, Q, so regular-rational and Real consumers see only the displayed finite route. Componentwise hsame transport and displayed Cont replay preserve each coordinate through H and C, and P, N record the same carrier as package provenance and local name.

The excluded objects have no carrier coordinate. Therefore the listed rows exhaust the local NameCert obligation surface. □

Theorem 1781.4 (Fast Cauchy modulus threshold dichotomy)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. For the displayed precision query Q, the threshold discipline row F exposes exactly one of the following rows read from the same source modulus row M: the direct fast threshold row, or the doubled-window threshold row. The choice is part of F itself and is not supplied by a hidden selector, quotient stream equality, selected limit, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge datum.*

Proof. Project the accepted carrier using Definition 1781.1. The only coordinates that mention thresholds are the source modulus row M, the fast discipline row F, and the precision query row Q.

Apply Definition 1781.2. A consumer can open a StreamName window only after Q has passed through M and F, and F records whether the direct fast threshold or the doubled threshold is used.

Apply Definition 1781.3. Its threshold exactness row requires the selected threshold to be either the direct fast threshold or the doubled-window threshold from M for the same Q. Since no other selector is present in the carrier, the two alternatives are exclusive and exhaustive on accepted packets. □

Theorem 1781.5 (Fast Cauchy modulus window exactness)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. The* $\mathsf{StreamName}^{\uparrow}$ *finite window row* S *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger row* D *are determined by the threshold selected in* F *for the precision query* Q*. If* F *selects the direct fast threshold, then* S *is the finite window opened at that threshold and* D *records the requested tolerance account. If* F *selects the doubled-window threshold, then* S *is the enlarged finite window and* D *records the dyadic slack proving that the doubled read preserves the requested tail estimate.*

Proof. Project the carrier and read the route displayed in Definition 1781.2. The row S appears only after the source modulus and threshold discipline rows have been read, so it is opened at the threshold selected in F.

In the direct case, the threshold stored by F already supplies the requested tail estimate. Thus the StreamName row is the finite window at that threshold, and the DyadicRatCore row stores the tolerance ledger consumed by that read.

In the doubled case, F stores the enlarged threshold before S is opened. The same theorem identifies S as the enlarged finite window and identifies D as the dyadic slack ledger that transports the doubled read back to the original precision request.

The NameCert obligations of Definition 1781.3 state that finite-window exactness and dyadic slack exactness are stored in those same displayed rows. No further window or tolerance coordinate is available. □

Theorem 1781.6 (Fast Cauchy modulus threshold minimality)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. The selected threshold row in* F *is exhausted by the direct fast threshold or the doubled-window threshold for the same precision query* Q*, and every downstream* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *read must use that selected row before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff is consumed.*

Proof. Project the accepted carrier from Definition 1781.1. The only threshold-bearing rows are the source modulus row M, the fast discipline row F, and the precision query row Q.

Apply Definition 1781.4. For the displayed query, F exposes either the direct fast threshold or the doubled-window threshold from the same source modulus row.

Apply Definition 1781.5. The finite window S and dyadic ledger D are determined by that selected threshold, and no additional window or tolerance coordinate is present.

Thus every downstream StreamName or DyadicRatCore read uses the selected row in F before the regular rational or Real handoff row E is consumed. The carrier has no third selector from which a competing threshold could be read. □

Theorem 1781.7 (Fast Cauchy modulus handoff nonescape)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff row* E *must consume the displayed rows* M, F, S, D, Q *before exposing a regular-rational readback or Real seal. The handoff exports no quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, host real equality, Lean verification row, or bridge datum.*

Proof. Project the carrier from Definition 1781.1. The handoff coordinate E is listed downstream of the source modulus, threshold discipline, finite window, dyadic ledger, and precision query rows.

Apply Definition 1781.2. Every regular-Cauchy or Real-facing consumer follows the finite route through Q, then M, F, then S, D, and only then reaches E.

Apply Definition 1781.3. Its regular and Real handoff row requires E to consume M, F, S, D, Q before any readback or seal is exposed, while its refusal row excludes quotient stream equality, selected limits, choice, trichotomy, ambient completeness, Lean verification, and bridge data from the local certificate surface.

Therefore the handoff cannot escape the displayed fast-modulus route. It exposes only the readback or seal certified by the finite carrier rows. □

Theorem 1781.8 (Fast Cauchy modulus consumer factorization)**.** *Let*

$$M_{\text{fast}} = (M, F, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{FastCauchyModulus}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of this carrier factors through the displayed route*

$$\mathsf{CauchyModulus}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{N}$$

In particular, the consumer reads the precision query, threshold discipline, finite window, dyadic ledger, and handoff rows before any Real-facing seal. No hidden selector, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, Lean verification row, or bridge datum remains available outside the displayed carrier.*

Proof. Project the accepted carrier from Definition 1781.1. The visible rows are exactly the source modulus M, fast discipline F, finite window S, dyadic ledger D, precision query Q, handoff E, transport H, replay C, provenance P, and local name N.

Apply Definition 1781.4. The precision query can select only the direct fast threshold or the doubled-window threshold displayed by F from the same source modulus row M.

Apply Definition 1781.5. The StreamName window S and DyadicRatCore ledger D are determined by the selected threshold and the same query row Q.

Finally use Definition 1781.7. The handoff row E is downstream of M, F, S, D, Q, while H, C, P, N preserve the same route as transport, replay, provenance, and local naming. Since the refused objects have no carrier coordinate, every RegSeqRat or RealUp consumer factors through the displayed threshold, window, ledger, and handoff rows. □

Falsifiable prediction: Every $\mathsf{FastCauchyModulus}^{\uparrow}$ precision read factors through one CauchyModulus row, one fast or doubled threshold discipline row, one StreamName finite window, one DyadicRatCore tolerance ledger, one precision query, one RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{FastCauchyModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source modulus row, the fast or doubled threshold choice, the opened finite window, the dyadic slack ledger, and the regular-rational or Real-facing handoff as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1781.1, Definition 1781.2, and Definition 1781.3 seed the finite fast-modulus route from a displayed CauchyModulus row through fast or doubled threshold discipline, StreamName finite window, DyadicRatCore tolerance ledger, precision query, RegSeqRat and Real handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{FastCauchyModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, countable choice, trichotomy, selected limits, unbounded stream reads, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, finite-window exactness, dyadic slack exactness, regular and Real handoff, componentwise transport, displayed replay, provenance, and finite consumer factorization for BEDC.Derived.FastCauchyModulusUp.

Constructive story: A FastCauchyModulus$^{\uparrow}$ packet is generated from one CauchyModulus row, one fast or doubled threshold discipline row, one StreamName finite window, one DyadicRatCore tolerance ledger, one precision query, one RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1782

A Concrete Naming Certificate for RegularCauchyCompletionUniqueness$^{\uparrow}$

Following Bishop and Bridges' constructive regular Cauchy completion, the RegularCauchyCompletionUniqueness$^{\uparrow}$ packet isolates the finite uniqueness boundary for comparing two completion handoff routes before any quotient or ambient completeness principle is available. It sits on the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Bishop completion handoff of chapter 1639, and the frontier route of chapter 1180. The packet records only the displayed common-tail comparison by which two accepted regular-Cauchy completion routes expose the same terminal seal boundary.

Definition 1782.1 (Regular Cauchy completion uniqueness carrier)**.** *A* RegularCauchyCompletionUniqueness$^{\uparrow}$ *carrier is a paired finite* BHist *packet*

$$U = (L, R, D, S, Q, E, T, H, C, P, N)$$

with the following displayed rows:

1. *L is the left completion handoff route, read through the* BishopRegularCauchyCompletion$^{\uparrow}$ *boundary before any terminal seal is compared.*
2. *R is the right completion handoff route, read through the* RegularCauchyCompletionFrontier$^{\uparrow}$ *boundary with the same consumer-facing precision request.*
3. *D is the shared* DyadicRatCore$^{\uparrow}$ *tolerance row used by both routes.*
4. *S is the shared* StreamName$^{\uparrow}$ *finite-window row opened by the tolerance row D.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row exposed inside the same finite window.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by both displayed handoffs after D, S, Q have been fixed.*
7. *T is the shared-tail comparison row, recording that the two routes are compared only after the common dyadic tolerance, stream window, and regular rational readback have been displayed.*
8. *H records componentwise* hsame *transport for the paired handoff rows, shared tolerance, window, readback, seal, tail comparison, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion-facing precision request through D, S, Q, T into the paired handoff rows L, R and the terminal seal row E.*

10. *P is the* Pkg *provenance over* RegularCauchyCompletionUniqueness$^{\uparrow}$, BishopRegularCauchyCompletion$^{\uparrow}$, RegularCauchyCompletionFrontier$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyCompletionUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, R, D, S, Q, E, T, H, C, P, N$. The carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 1782.2 (Regular Cauchy completion uniqueness NameCert obligation surface)**.** *For every accepted* RegularCauchyCompletionUniqueness$^{\uparrow}$ *carrier*

$$U = (L, R, D, S, Q, E, T, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyCompletionUniqueness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the paired completion handoff routes, shared dyadic tolerance row, stream-window row, regular rational readback row, terminal Real seal row, shared-tail comparison row, transport, replay, provenance, and local naming row;*
2. *paired-route exposure, requiring both L and R to be displayed before the local classifier compares their completion-facing terminal seal routes;*
3. *shared-tail comparison, requiring the comparison to pass through the same D, S, Q rows before either handoff route reaches E;*
4. *terminal-seal coincidence, requiring the consumer-visible seal row E to be read only as the common endpoint of the two displayed routes after T has fixed the shared tail surface;*
5. *classifier transport, requiring componentwise* hsame *in H and displayed* Cont *replay in C to preserve the ordered route D, S, Q, T, L, R, E;*
6. *ledger non-escape, requiring P and N to export no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient Real completeness theorem.*

No uniqueness obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUniquenessUp.RegularCauchyCompletionUniquenessUpTasteGate`
`single carrier alignment`

Proof. Carrier admission is projection from Definition 1782.1. The displayed tuple supplies exactly the two handoff rows, the shared tolerance-window-readback rows, the terminal seal, the comparison row, and the structural transport, replay, provenance, and naming rows.

Paired-route exposure is forced by the placement of L and R. A consumer cannot compare completion handoffs until both routes have been opened as finite BHist data, one through the Bishop completion boundary and the other through the frontier boundary.

The shared-tail comparison is the row T governed by D, S, Q. It fixes the common dyadic tolerance, the same finite StreamName window, and the same regular rational readback before either route is allowed to claim the terminal seal row.

Terminal-seal coincidence is therefore a displayed endpoint fact about E, not an equality of quotient streams or host real values. The seal is reached only after the common tail surface has already been read.

Finally, componentwise hsame in H and displayed Cont replay in C preserve the ordered route D, S, Q, T, L, R, E. Since the carrier contains no quotient, selected-limit, choice, host-equality, or Real-completeness coordinate, the provenance and local NameCert rows cannot export any such object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1782.1 and Definition 1782.2 seed the finite uniqueness boundary for paired regular-Cauchy completion handoffs through shared dyadic tolerance, stream-window, regular rational readback, terminal Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionUniquenessUp.RegularCauchyCompletionUniquenessUpTasteGate\_singl`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, paired-route exposure, shared-tail comparison, terminal-seal coincidence, classifier transport, and ledger non-escape for BEDC.Derived.RegularCauchyCompletionUniquenessUp.

Constructive story: A $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$ packet is generated from two completion handoff rows, one shared dyadic tolerance, one finite StreamName window, one RegSeqRat readback, one terminal Real seal, one shared-tail comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1783

A Concrete Naming Certificate for $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of real numbers by regular Cauchy data, the $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ packet records the finite comparison-principle surface used before a completed real seal is consumed. It sits over the regular rational source rows of chapter 301, the shared finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the equality-facing comparison row of chapter 411, the domination-test row of chapter 1459, and the terminal seal surface of chapter 38. The packet is a human-origin L10 comparison-principle boundary for composing equality and domination comparison rows before $\mathsf{Real}^{\uparrow}$ seal handoff; it is not quotient stream equality, host real comparison, trichotomy, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1783.1 (Regular Cauchy comparison principle carrier)**.** *A* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Y, W, D, E, T, Q, R, H, C, K, N)$$

with the following displayed rows:

1. *X is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source row.*
2. *Y is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source row inspected at the same tolerance requests as X.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family on which X, Y, the dyadic ledger, and both comparison rows are read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed dyadic scales, endpoint observations, and rational tolerance rows attached to the finite windows in W.*
5. *E is the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *equality row, recording that equality-facing inspection of X and Y is mediated by the same W, D surface.*
6. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *domination row, recording the finite majorant or target-test comparison on the same windows and tolerances.*
7. *Q is the comparison verdict row obtained only after E and T have both been displayed.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row consumed after the verdict row Q is fixed.*

9. *H records componentwise* hsame *transport for the two regular Cauchy sources, shared windows, dyadic ledger, equality row, domination row, verdict row, seal row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a comparison request through W, D, E, T, Q, R and then through the structural rows.*

11. *K is the* Pkg *provenance over* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through these twelve displayed rows. There is no coordinate for quotient stream equality, host real comparison, trichotomy, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonPrincipleUp.TasteGate.RegularCauchyComparisonPrincipleTaste`
`single carrier alignment`

Theorem 1783.2 (Regular Cauchy comparison principle NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier*

$$P = (X, Y, W, D, E, T, Q, R, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the equality comparison row, the domination-test row, the comparison verdict, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, transport, replay, provenance, and local naming row;*

2. *shared-window exposure, requiring X and Y to be inspected only on the windows displayed by W;*

3. *tolerance-ledger exactness, requiring every comparison tolerance to be read from D before either comparison row is consumed;*

4. *equality-row routing, requiring the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row E to use the same X, Y, W, D surface;*

5. *domination-row routing, requiring the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *row T to use the same window and tolerance rows before contributing to the verdict Q;*

6. *seal handoff, requiring R to be reached only after E, T, Q have been displayed;*

7. *structural stability, requiring H, C, K, N to transport, replay, source, and name exactly the displayed route.*

No comparison-principle obligation is stored outside these rows.

Proof. Project the carrier of Definition 1783.1. The projection exposes exactly two regular Cauchy source rows, one shared window family, one dyadic tolerance ledger, one equality comparison row, one domination-test row, one verdict row, one seal handoff, and the structural transport, replay, provenance, and naming rows.

Shared-window exposure and tolerance-ledger exactness are forced by W and D. A comparison request first opens a finite StreamName window and then reads the dyadic tolerance ledger attached to that window, so neither E nor T can inspect a source outside the displayed finite surface.

The equality and domination rows are then read on the same surface. The row E contributes the equality-facing comparison route of Definition 411.3, while T contributes the domination-test route of Definition 1459.2. Since both rows are attached to X, Y, W, D, the verdict row Q is a finite comparison-principle read rather than a host real comparison.

Finally R, H, C, K, N close the route. The Real seal row is reachable only after E, T, Q have been displayed, and the structural rows only transport, replay, source, and name that route. Thus the local NameCert surface contains no hidden quotient, trichotomy, selected-limit, choice, completeness, Lean, or bridge coordinate. □

Theorem 1783.3 (Regular Cauchy comparison principle carrier admission)**.** *An accepted* RegularCauchyComparison *packet is admitted only as the twelve-row carrier*

$$P = (X, Y, W, D, E, T, Q, R, H, C, K, N)$$

of Definition 1783.1. Thus carrier admission displays the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *equality row,* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *domination row, verdict,* $\mathsf{Real}^{\uparrow}$ *seal handoff,* BHist *transport through* hsame, Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Project the carrier definition. The displayed tuple supplies exactly $X, Y, W, D, E, T, Q, R, H, C, K, N$ and no additional comparison coordinate.

The source, window, and tolerance rows are read first, so admission is already finite before either comparison row is consumed. The equality and domination rows then attach to the same displayed data, and the verdict row has no independent source.

Finally R, H, C, K, N only seal, transport, replay, source, and name the admitted route. This proves that carrier admission is precisely the displayed NameCert surface and not a host comparison, quotient equality, trichotomy, selected-limit, choice, completeness, Lean, or bridge datum. □

Theorem 1783.4 (Regular Cauchy comparison principle shared-window exposure)**.** *For every accepted* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier* $P = (X, Y, W, D, E, T, Q, R, H, C, K, N)$*, the equality row* E*, domination row* T*, and verdict row* Q *inspect* X *and* Y *only through the shared* $\mathsf{StreamName}^{\uparrow}$ *window row* W *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*. In particular, no comparison-principle consumer can read a tail-order verdict before* W *and* D *have been exposed.*

Proof. Apply Definition 1783.2. Its shared-window and tolerance-ledger rows require a comparison request to open W and then read D before either comparison row is consumed.

Use Definition 1783.6. The displayed route places E, T, Q downstream of W, D, so equality and domination comparisons share the same finite inspection surface.

The Real seal handoff of Definition 1783.7 is later than Q. Hence a consumer that reaches a verdict or seal has already exposed the common windows and dyadic tolerances, and no hidden tail, host comparison, or ambient completeness row is available. □

Theorem 1783.5 (Regular Cauchy comparison principle equality-domination verdict)**.** *Every verdict exported by an accepted* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier is routed through both the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *equality row* E *and the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *domination row* T *on the same* X, Y, W, D *surface before the* $\mathsf{Real}^{\uparrow}$ *seal row* R *is read. The verdict row* Q *therefore records a finite equality-domination comparison and not a host real comparison, trichotomy, selected limit, countable choice, quotient equality, ambient completeness, Lean verification, or bridge check.*

Proof. Start from the carrier projection and read X, Y, W, D. These rows expose the two regular Cauchy sources, the shared finite windows, and the dyadic tolerance ledger.

Now read the equality and domination rows. The row E is attached to the $\mathsf{RegularCauchyComparison}^{\uparrow}$ equality-facing route, while T is attached to the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ domination route. Since both rows use the same X, Y, W, D data, the verdict Q is a comparison-principle row over a single finite surface.

Finally apply Definition 1783.7. The Real seal row R consumes the verdict only after E, T, Q have been displayed, so the exported verdict cannot be an unrecorded host comparison or any of the excluded escape data. □

Theorem 1783.6 (Regular Cauchy comparison principle tail-order route)**.** *Let*

$$P = (X, Y, W, D, E, T, Q, R, H, C, K, N)$$

be an accepted $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier. Every tail-order comparison exported by P factors through the finite route*

$$X, Y \longmapsto W, D \longmapsto E, T \longmapsto Q \longmapsto R, H, C, K, N.$$

The route first binds the two regular Cauchy sources to the shared StreamName windows and DyadicRatCore tolerance ledger, then composes the $\mathsf{RegularCauchyComparison}^{\uparrow}$ *equality row with the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *domination row before any Real-facing seal consumer can read a verdict.*

Proof. Begin with a tail-order comparison request. The carrier projection gives the source rows X, Y but admits no direct comparison between them; the request must first pass through the shared finite window family W.

The dyadic ledger D fixes the displayed tolerance rows on those windows. This pins the numerical inspection used by both comparison siblings to the same finite data.

Now read the two comparison rows. The equality-facing row E is the $\mathsf{RegularCauchyComparison}^{\uparrow}$ route over X, Y, W, D, and the domination row T is the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ route over the same windows and tolerances. Their combination supplies the verdict row Q.

The remaining rows only export the verdict. The seal handoff R consumes Q, and H, C, K, N preserve, replay, source, and name the displayed route. A tail-order consumer therefore sees the ordered finite route above and no quotient stream, host real comparison, trichotomy, selected limit, choice, completeness, Lean, or bridge datum. □

Theorem 1783.7 (Regular Cauchy comparison principle Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ *carrier is downstream of the equality and domination comparison rows:*

$$W, D, E, T, Q \longmapsto R.$$

Consequently the seal handoff consumes a finite comparison-principle verdict before it reaches $\mathsf{Real}^{\uparrow}$*, and it does not assert quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Use Definition 1783.2 to place every accepted packet on the displayed NameCert route. The only seal row in the carrier is R, and the obligations require it to be read after the comparison rows E, T and verdict row Q.

Apply Definition 1783.6. Its factorization shows that E and T are already downstream of the common window and dyadic tolerance ledger, so the Real-facing consumer cannot bypass W, D or inspect an unrecorded tail.

Finally read R as the terminal handoff row. Since R is reached only through the finite route W, D, E, T, Q, the exported seal is a BEDC comparison-principle handoff and not a host equality, trichotomy, selected-limit, choice, completeness, Lean, or bridge assertion. □

Falsifiable prediction: A $\mathsf{Real}^{\uparrow}$ consumer that reaches a comparison verdict from accepted $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ data without first reading the shared $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegularCauchyComparison}^{\uparrow}$ equality row, and $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ domination row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-comparison, regular-cauchy-comparison-test, real: no listed sibling supplies the whole twelve-row packet $P = (X, Y, W, D, E, T, Q, R, H,$ because each sibling contributes only source, window, tolerance, equality-comparison, domination-test, or seal data while this chapter binds all six faces with a verdict row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}$ packet is generated from two RegSeqRat source rows, shared StreamName windows, a DyadicRatCore tolerance ledger, a RegularCauchyComparison equality row, a RegularCauchyComparisonTest domination row, a comparison verdict, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1783.1, Definition 1783.2, Definition 1783.6, and Definition 1783.7 seed the finite comparison-principle surface over two regular Cauchy source rows, shared windows, dyadic tolerances, equality and domination comparison rows, verdict, seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host real comparison, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window exposure, tolerance-ledger exactness, equality-row routing, domination-row routing, Real seal handoff, and structural stability for BEDC.Derived.RegularCauchyComparisonPrin

Chapter 1784

A Concrete Naming Certificate for $\mathsf{RealIntegerPart}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntegerPart}^{\uparrow}$ packet records the finite boundary at which a displayed real name may expose an integer-part read to decimal, approximation, and completion-facing consumers. It sits after the real-name seal of chapter 38, the Archimedean bound surface of chapter 430, the integer-bracketing certificate of chapter 1479, the residual interface of chapter 1648, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, and the regular-rational readback route of chapter 301. The packet names the public RealUp integer-part boundary; it is not a host floor operation, a trichotomy oracle, quotient real equality, a selected representative, countable choice, or ambient Real completeness.

Definition 1784.1 (RealIntegerPart carrier)**.** *A* $\mathsf{RealIntegerPart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, B, Z, F, D, S, G, H, C, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose integer part is being read.*
2. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row attached to the same source real.*
3. *B is the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row, displaying the finite lower integer candidate and successor comparison for R.*
4. *Z is the* $\mathsf{Int}^{\uparrow}$ *integer candidate exposed by the bracket row.*
5. *F is the* $\mathsf{FractionalPart}^{\uparrow}$ *residual row tied to the same source real and integer candidate.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the tolerances and dyadic refinements used to observe the source, bracket, and residual rows.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for decimal, approximation, and completion-facing reads.*
8. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets the windows in S as regular rational data before any consumer receives the integer-part value.*
9. *H records componentwise* hsame *transport for the source real, Archimedean bound, integer bracket, integer candidate, residual, dyadic scale, stream window, readback, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from every precision, decimal-prefix, or approximation request through R, A, B, Z, F, D, S, G without inserting another integer search.*

11. *P is the* Pkg *provenance over* $\mathsf{RealIntegerPart}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{FractionalPart}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for host floor, trichotomy over arbitrary real names, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1784.2 (RealIntegerPart NameCert obligations)**.** *For every accepted* $\mathsf{RealIntegerPart}^{\uparrow}$ *carrier*

$$K = (R, A, B, Z, F, D, S, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the source* $\mathsf{Real}^{\uparrow}$ *row,* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row,* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row,* $\mathsf{Int}^{\uparrow}$ *candidate row,* $\mathsf{FractionalPart}^{\uparrow}$ *residual row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and structural rows H, C, P, N;*

2. *bracket-route exactness, requiring the integer candidate Z to be read only through the displayed source row R, Archimedean bound row A, and ArchimedeanIntegerPart bracket row B;*

3. *residual handoff, requiring F to consume the same R, B, Z route before any fractional residual is exposed;*

4. *classifier transport, requiring the accepted status of R, A, B, Z, F, D, S, G to remain stable under the componentwise* hsame *transports in H and the displayed* Cont *replay in C;*

5. *ledger non-escape, excluding host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, ambient Real completeness, Lean verification, and bridge checking from the local certificate surface.*

No RealIntegerPart obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealIntegerPartUp.RealIntegerPartTasteGate single carrier alignment`

Proof. Project the accepted packet $K = (R, A, B, Z, F, D, S, G, H, C, P, N)$. The projection exposes exactly the real source, Archimedean bound, integer bracket, integer candidate, residual, dyadic scale, stream window, regular readback, transport, replay, provenance, and naming rows.

Bracket-route exactness follows from the sibling surface of chapter 1479. The row B is downstream of the source real and Archimedean bound rows, and the integer candidate Z is exposed only as the integer endpoint carried by that bracket.

Residual handoff follows from chapter 1648. The residual row F records the same source real and integer candidate before it opens any fractional read, so the residual cannot introduce a second integer-part route.

The transport and replay clauses are stored in H and C. They carry the visible rows componentwise and send each downstream request back through R, A, B, Z, F, D, S, G.

The excluded objects have no carrier coordinate and no classifier clause. Therefore the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}$ obligation surface. □

Theorem 1784.3 (RealIntegerPart bracket route)**.** *Every Real, decimal, or approximation consumer that reads an integer part from an accepted* $\mathsf{RealIntegerPart}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{ArchimedeanReal}^{\uparrow} \longmapsto \mathsf{ArchimedeanIntegerPart}^{\uparrow} \longmapsto \mathsf{Int}^{\uparrow} \longmapsto \mathsf{FractionalPart}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Str}$$

Concretely, the consumer first receives the source real row R, then the Archimedean bound row A, the ArchimedeanIntegerPart bracket row B, the integer candidate row Z, the fractional residual row F, the dyadic scale row D, the finite window row S, the regular readback row G, and finally the structural rows H, C, P, N.

Lean4 ✓ `BEDC.Derived.RealIntegerPartUp.RealIntegerPartCarrier bracket route`

Proof. Begin with a precision, decimal-prefix, or approximation request. The replay row C sends the request to the displayed $\mathsf{Real}^{\uparrow}$ source row R, so the route starts from a named real carrier rather than a host real value.

The Archimedean data then enter through A and B. The row A provides the finite bound surface, and the row B supplies the bracket route named by chapter 1479.

The integer candidate Z is the endpoint exported by that bracket. The residual row F is then read through chapter 1648, tying the integer endpoint and source real to the same fractional handoff.

The consumer can inspect the route only through the dyadic, window, and readback rows D, S, G. These rows expose finite tolerances, finite StreamName windows, and RegSeqRat readback before the integer-part value is usable by a Real or decimal consumer.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since no other coordinate is present, every integer-part read factors through the stated boundary. □

Falsifiable prediction: Every $\mathsf{RealIntegerPart}^{\uparrow}$ integer-part read factors through one $\mathsf{Real}^{\uparrow}$ source row, one $\mathsf{ArchimedeanReal}^{\uparrow}$ bound row, one $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, one $\mathsf{Int}^{\uparrow}$ candidate row, one $\mathsf{FractionalPart}^{\uparrow}$ residual row, one $\mathsf{DyadicRatCore}^{\uparrow}$ scale row, one $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, archimedeanreal, archimedean-integer-part, int, fractional-part, dyadicratcore, streamname, regseqrat: no listed sibling is bijective to $\mathsf{RealIntegerPart}^{\uparrow}$ because this packet jointly binds the source real, Archimedean bound, integer bracket, integer candidate, fractional residual, finite tolerance scale, window read, readback route, transport, replay, provenance, and local naming for the Real-facing integer-part boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1784.1, Definition 1784.2, and Definition 1784.3 seed the finite Real-facing integer-part boundary from a displayed Real source through Archimedean bound, ArchimedeanIntegerPart bracket, Int candidate, FractionalPart residual, DyadicRatCore scale, StreamName window, RegSeqRat readback, and structural rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntegerPart}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host floor, trichotomy, quotient real equality, selected representatives, countable choice, hidden Archimedean search, unbounded stream reads, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, bracket-route exactness, residual handoff, classifier transport, and ledger non-escape for BEDC.Derived.RealIntegerPartUp.

Constructive story: A $\mathsf{RealIntegerPart}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ source row, an $\mathsf{ArchimedeanReal}^{\uparrow}$ bound row, an $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ bracket row, an $\mathsf{Int}^{\uparrow}$ candidate row, a $\mathsf{FractionalPart}^{\uparrow}$ residual row, a $\mathsf{DyadicRatCore}^{\uparrow}$

scale row, a StreamName$^{\uparrow}$ window row, a RegSeqRat$^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1785

A Concrete Naming Certificate for $\mathsf{NewtonKantorovich}^{\uparrow}$

Following Kantorovich and Bishop–Bridges constructive analysis, the $\mathsf{NewtonKantorovich}^{\uparrow}$ packet records the finite Newton data that turns a displayed derivative source, inverse witness, and contraction neighbourhood into a seal-facing real root row without asserting host root existence. It consumes the finite step discipline of chapter 223, the contraction handoff of chapter 396, the derivative source of chapter 122, the regular rational readback route of chapter 301, and the terminal Real seal of chapter 38. The chapter is a Real-and-Completion frontier consumer: it exposes the finite certificate route from Newton ledgers to a $\mathsf{Real}^{\uparrow}$ root boundary while leaving convergence, completeness, and equality claims inside the named BEDC rows that provide them.

Definition 1785.1 (Newton-Kantorovich carrier)**.** *A* $\mathsf{NewtonKantorovich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, I, U, L, Q, B, N, R, E, H, C, P, Z)$$

with the following displayed coordinates:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *source row, including the point context, function-value history, derivative-answer history, and local derivative classifier consumed by the Newton data.*
2. *I is the inverse witness row, recording the displayed inverse or one-sided inverse budget for the derivative row named by D.*
3. *U is the contraction neighbourhood row, storing the radius, local Lipschitz or majorant budget, and the nonescape side condition for all displayed Newton steps inside the named neighbourhood.*
4. *L is the finite* $\mathsf{NewtonIteration}^{\uparrow}$ *ledger, recording source points, function-value rows, derivative-answer rows, inverse-witness reads, next-step rows, and splice-compatible step transports.*
5. *Q is the Newton residual and contraction-ratio ledger obtained from D, I, U, L by finite rational comparisons.*
6. *B is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *handoff row attached to the Newton map determined by D, I, U, Q, exposing the contraction packet used downstream.*
7. *N is the finite Newton-to-Banach replay row, stating that the displayed Newton step ledger L is read as the Picard ledger consumed by B after the inverse and contraction rows have been opened.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the Banach handoff and the finite Newton residual ledger.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing root row exposed only after D, I, U, L, Q, B, N, R have been displayed.*
10. *H records componentwise* hsame *transport for the derivative source, inverse witness, contraction neighbourhood, Newton ledger, residual ledger, Banach handoff, replay, RegSeqRat readback, Real seal, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from the derivative source through the inverse witness, contraction neighbourhood, finite Newton ledger, Banach handoff, RegSeqRat readback, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{NewtonKantorovich}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *Z is the local* $\mathsf{NameCert}_{\mathsf{NewtonKantorovich}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, U, L, Q, B, N, R, E, H, C, P, Z$. It has no coordinate for host root existence, quotient-real equality, global invertibility, ambient completeness, selected limits, or bridge checking.

Lean4 ✓ `BEDC.Derived.NewtonKantorovichUp.NewtonKantorovichTasteGate single carrier alignment`

Theorem 1785.2 (Newton-Kantorovich NameCert obligations)**.** *For every accepted* $\mathsf{NewtonKantorovich}^{\uparrow}$ *carrier*

$$K = (D, I, U, L, Q, B, N, R, E, H, C, P, Z),$$

the local $\mathsf{NameCert}_{\mathsf{NewtonKantorovich}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, derivative-source stability for D, inverse-witness stability for I, contraction-neighbourhood nonescape for U, finite Newton-ledger stability for L, residual-ledger exactness for Q, Banach handoff compatibility for B, Newton-to-Banach replay for N, RegSeqRat readback for R, Real-seal non-escape for E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through Z.*

Proof. Carrier admission is projection from Definition 1785.1. Each listed row is a visible coordinate of the finite packet, so the local naming surface starts with those coordinates and cannot read a hidden root object.

The derivative, inverse, and neighbourhood obligations are the source-side checks. The row D supplies the derivative histories, I supplies the inverse witness attached to those histories, and U supplies the displayed radius and nonescape side condition used by the Newton rows.

The Newton ledger and residual rows are finite. The row L carries only displayed $\mathsf{NewtonIteration}^{\uparrow}$ steps, and Q records the rational comparisons that make those steps compatible with the contraction neighbourhood.

The downstream obligations are then forced in order. The Banach handoff B consumes the Newton map data; the replay row N identifies the Newton ledger as the Picard ledger accepted by B; the readback row R exposes the regular rational name; and the seal row E gives the terminal $\mathsf{Real}^{\uparrow}$ boundary. The transport, replay, provenance, and local naming rows H, C, P, Z repack the same route. □

Theorem 1785.3 (Newton-Kantorovich Banach handoff)**.** *Every accepted* $\mathsf{NewtonKantorovich}^{\uparrow}$ *carrier factors its root-read route through the finite handoff*

$$D, I, U, L, Q \longmapsto B \longmapsto N \longmapsto R \longmapsto E.$$

The row B is the only contraction-completion handoff available to the packet; the row N is the only bridge from the displayed $\mathsf{NewtonIteration}^{\uparrow}$ *ledger to the Picard ledger consumed by B; and the rows R, E are reached only after this handoff has been displayed.*

Proof. Read the coordinates of K. The derivative row D, inverse witness I, neighbourhood row U, Newton ledger L, and residual ledger Q are the only rows that describe the Newton map before a contraction packet is opened.

The Banach handoff row B is therefore the first row that may invoke the fixed-point surface of chapter 396. The replay row N records the finite identification between the Newton ledger and the Picard ledger that B consumes, so there is no second completion route outside the displayed handoff.

After B and N are present, the packet may read R and then E. This yields a regular rational readback and a terminal $\mathsf{Real}^{\uparrow}$ seal, but it does not add a host root, a quotient-real equality, a chosen limit, or an ambient completeness theorem. □

Theorem 1785.4 (Newton-Kantorovich nonescape boundary)**.** *Let $K = (D, I, U, L, Q, B, N, R, E, H, C, P, Z)$ be an accepted* $\mathsf{NewtonKantorovich}^{\uparrow}$ *carrier. Every public read of the seal row E must pass through the neighbourhood row U and the finite Newton ledger L before it reaches the Banach handoff B. Consequently the root boundary exposed by E is confined to the displayed contraction neighbourhood and cannot export a global root-existence theorem, global inverse theorem, host equality of roots, quotient-real equality, or ambient completeness beyond the carried rows.*

Proof. The only row carrying a neighbourhood boundary is U. Since E is downstream of the handoff route in Definition 1785.3, any read of E must first pass through the data that certify the Newton steps remain inside that neighbourhood.

The finite ledger L supplies those displayed Newton steps. The residual ledger Q and Banach handoff B may use them to produce the contraction route, but neither row introduces an additional domain or a global inverse principle.

Thus the exposed root boundary is local to U and to the rows transported by H and replayed by C. Because the carrier lacks coordinates for global root existence, quotient-real equality, host equality, selected limits, and ambient completeness, no public consumer can read those objects from E. □

Falsifiable prediction: Every $\mathsf{NewtonKantorovich}^{\uparrow}$ root-boundary read factors through one Derivative row, one inverse witness row, one contraction neighbourhood row, one finite NewtonIteration ledger, one residual ledger, one BanachFixedPoint handoff row, one Newton-to-Banach replay row, one RegSeqRat readback row, one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: derivative, newtoniteration, banachfixedpoint, regseqrat, real: $\mathsf{NewtonKantorovich}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the derivative source, inverse witness, contraction neighbourhood, finite Newton ledger, Banach handoff, regular readback, and seal-facing root boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonKantorovich}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NewtonKantorovich}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1785.1, Definition 1785.2, Definition 1785.3, and Definition 1785.4 seed the finite route from derivative data and inverse witness through a contraction neighbourhood, Newton finite ledger, Banach handoff, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NewtonKantorovich}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host root existence, quotient-real equality, global invertibility, host equality of roots, selected limits, ambient completeness beyond the carried rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-source stability, inverse-witness stability, contraction-neighbourhood nonescape, Newton-ledger stability, residual-ledger exactness, Banach handoff compatibility, Newton-to-Banach replay, RegSeqRat readback, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.NewtonKantorovichUp.

Constructive story: A NewtonKantorovich$^{\uparrow}$ packet is generated from one Derivative row, one inverse witness row, one contraction neighbourhood row, one finite NewtonIteration ledger, one residual ledger, one BanachFixedPoint handoff row, one Newton-to-Banach replay row, one RegSeqRat readback row, one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1786

A Concrete Naming Certificate for $\mathsf{RealRootBracket}^{\uparrow}$

Following Bishop and Bridges constructive analysis, a root bracket is read as a finite sign-change interval together with the displayed continuity and bisection evidence that keeps the interval meaningful at a requested tolerance. The $\mathsf{RealRootBracket}^{\uparrow}$ packet packages that route as BEDC data: endpoint interval and sign rows are inspected through the intermediate-value surfaces of chapter 1402 and chapter 1422, retained windows are supplied by chapter 1407, image transport is routed through chapter 1775, and the terminal readback uses chapter 126, chapter 301, chapter 291, chapter 130, and chapter 38. The certificate names a bracket surface, not a selected root.

Definition 1786.1 (RealRootBracket carrier)**.** *A* $\mathsf{RealRootBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, S, M, V, K, W, A, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the endpoint interval row. It records the lower and upper endpoint windows, rational or dyadic cells, endpoint order evidence, and the interval identity consumed by the bracket.*
2. *S is the endpoint sign row. It stores the displayed lower and upper sign data and the sign-change orientation used by the bracket packet.*
3. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus row for the inspected function, including the input tolerance demanded by each bracket tolerance request.*
4. *V is the* $\mathsf{IntermediateValue}^{\uparrow}$ *ledger row. It records the finite intermediate-value obligation surface attached to the interval, signs, and displayed continuity data.*
5. *K is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *ledger row. It records the finite branch prefix, midpoint sign rows, and regular-window handoff used to refine the initial sign-change interval.*
6. *W is the* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *retained-window row. It stores the lower and upper retained endpoint windows after the requested branch prefix.*
7. *A is the* $\mathsf{CauchyContinuousImage}^{\uparrow}$ *image route row. It records how endpoint and midpoint function reads are replayed through the displayed continuity modulus and image-window route.*
8. *R is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained lower and upper endpoint names.*

9. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback ledger, containing the regular rational endpoint readbacks, dyadic tolerances, endpoint comparisons, branch radii, and rounding rows used by the bracket.*

10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the retained bracket endpoints and for any Real-facing bracket consumer.*

11. *H records componentwise* hsame *transport for the endpoint interval, endpoint signs, modulus row, intermediate-value ledgers, retained windows, image route, stream windows, regular and dyadic readback, Real seal, replay, provenance, and local naming rows.*

12. *C records displayed* Cont *replay from a bracket request through endpoint inspection, sign inspection, modulus use, intermediate-value admission, bisection refinement, retained-window exposure, image routing, regular readback, and Real-seal consumption.*

13. *P is the* Pkg *provenance over* $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{BisectionRootIsolation}^{\uparrow}$, $\mathsf{CauchyContinuousImage}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{RealRootBracket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, M, V, K, W, A, R, D, E, H, C, P, N$. It has no coordinate for a selected root, quotient real equality, real trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host root equality.*

Lean4 ✓ `BEDC.Derived.RealRootBracketUp.RealRootBracketTasteGate single carrier alignment`

Theorem 1786.2 (RealRootBracket NameCert obligations)**.** *For every accepted* $\mathsf{RealRootBracket}^{\uparrow}$ *carrier*

$$B = (I, S, M, V, K, W, A, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealRootBracket}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint interval, endpoint signs, continuous-map modulus, intermediate-value ledger, intermediate-value bisection ledger, retained isolating windows, continuous-image route, stream windows, regular and dyadic readback, Real seal, transports, replay, provenance, and local naming;*

2. *endpoint discipline, requiring I and S to expose the bracket interval and opposite displayed sign rows before any bisection or Real-facing row is available;*

3. *modulus discipline, requiring M to answer each requested bracket tolerance by a displayed source tolerance consumed by the endpoint and image rows;*

4. *intermediate-value discipline, requiring V and K to attach every sign-change branch and midpoint sign row to the already displayed endpoint and modulus data;*

5. *retained-window discipline, requiring W to store only windows retained by the finite bisection ledger and to expose them through R and D before the Real seal is read;*

6. *image-route discipline, requiring A to replay function-value reads through the* $\mathsf{CauchyContinuousImage}^{\uparrow}$ *surface rather than through host function extensionality;*

7. *seal nonescape, requiring E, H, C, P, N to export only the displayed bracket rows and no selected root, quotient real equality, real trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host root equality.*

No RealRootBracket obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1786.1. The accepted packet displays exactly the endpoint interval, sign rows, modulus, intermediate-value ledgers, retained windows, image route, regular readback, Real seal, and structural rows.

Endpoint discipline is the first stage of the route. The interval row I and sign row S must be opened before the intermediate-value or bisection rows can be inspected, so the packet cannot replace the bracket by a host interval or by a completed root.

The modulus and intermediate-value disciplines are the middle stage. The continuous-map row M supplies the tolerance conversion, the row V records the finite intermediate-value surface, and the row K records the branch prefix and midpoint signs that retain a sign-change window.

Retained-window discipline and image-route discipline are downstream of that branch prefix. The row W exposes only the retained windows; R and D read them as StreamName, RegSeqRat, and DyadicRatCore data; A replays function-value reads through the CauchyContinuousImage route.

Finally, E, H, C, P, N seal, transport, replay, package, and name the same finite route. Since selected roots, quotient real equality, real trichotomy, open-cover compactness, ambient Real completeness, and host root equality are absent from the carrier, no local obligation can export them. □

Theorem 1786.3 (RealRootBracket bisection handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealRootBracket}^{\uparrow}$ *carrier factors through the displayed route*

$$I, S, M, V \longmapsto K \longmapsto W \longmapsto R, D \longmapsto E, H, C, P, N.$$

The consumer first reads the endpoint interval, sign rows, modulus data, and intermediate-value ledger; then the finite IntermediateValueBisection branch prefix; then the retained BisectionRootIsolation windows; then the regular rational and dyadic readback; and only then the Real-facing bracket seal. The handoff exports a finite bracket window, not a selected root or host equality theorem.

Proof. Start with an accepted packet $B = (I, S, M, V, K, W, A, R, D, E, H, C, P, N)$. The only endpoint and sign coordinates are I and S, and the only continuity coordinate is M, so every bracket read begins with those rows.

The intermediate-value stage is recorded by V and K. The row V admits the sign-change obligation attached to the displayed endpoint and modulus data, while K records the finite branch prefix and midpoint sign rows used by the bisection route.

The retained-window stage is W. It is downstream of K and supplies the visible lower and upper windows that are then exposed through R and D as StreamName, RegSeqRat, and DyadicRatCore data at the requested tolerance.

The Real-facing stage is terminal. The seal E is read only after the retained windows and readback ledger are fixed, while H, C, P, N transport, replay, package, and name the same route. The image row A is also bound to this route, because its function-value reads consume the displayed modulus and retained endpoint windows.

Thus every listed consumer factors through the finite handoff route. No step introduces a selected root, quotient real equality, real trichotomy, open-cover compactness, ambient Real completeness, or host root equality. □

Falsifiable prediction: Every $\mathsf{RealRootBracket}^{\uparrow}$ read factors through endpoint interval and sign rows, one $\mathsf{ContinuousMap}^{\uparrow}$ modulus row, $\mathsf{IntermediateValue}^{\uparrow}$ and $\mathsf{IntermediateValueBisection}^{\uparrow}$ ledgers,

one BisectionRootIsolation$^{\uparrow}$ retained-window row, one CauchyContinuousImage$^{\uparrow}$ image route row, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ and DyadicRatCore$^{\uparrow}$ readback, one Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, dyadicratcore, continuousmap, intermediatevalue, intermediate-value-bisection, bisectionrootisolation, cauchy-continuous-image: RealRootBracket$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint sign data, a continuity modulus, intermediate-value admission, bisection branch data, retained isolating windows, continuous-image replay, regular and dyadic readback, and a Real-facing bracket seal.

Closure status for NameCert$_{\text{RealRootBracket}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RealRootBracket}^{\uparrow}}$, seed)

Scope closed: Definition 1786.1, Definition 1786.2, and Definition 1786.3 seed the finite constructive root-bracketing route from endpoint signs and continuity data through intermediate-value bisection, retained isolating windows, continuous-image replay, regular and dyadic readback, and Real-facing seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RealRootBracket}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected roots, quotient reals, real trichotomy, open-cover compactness, ambient Real$^{\uparrow}$ completeness, host root equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint discipline, modulus discipline, intermediate-value ledger discipline, retained-window handoff, image-route replay, seal nonescape, and bisection-handoff factorization for BEDC.Derived.RealRootBracketUp.

Constructive story: A RealRootBracket$^{\uparrow}$ packet is generated from endpoint interval and sign rows, a ContinuousMap$^{\uparrow}$ modulus row, IntermediateValue$^{\uparrow}$ and IntermediateValueBisection$^{\uparrow}$ ledgers, BisectionRootIsolation$^{\uparrow}$ retained windows, a CauchyContinuousImage$^{\uparrow}$ image route, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ and DyadicRatCore$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1787

A Concrete Naming Certificate for $\mathsf{ApartnessCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ApartnessCompletion}^{\uparrow}$ packet records the generic separated completion boundary for sources whose equality information is exposed through apartness evidence. It sits between the real-specific boundary of chapter 1567, the Hausdorff separation surface of chapter 402, the constructive metric completion route of chapter 1556, the finite windows of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite boundary object for Real and Completion consumers; it is not a quotient completion, a host equality principle, a selected limit, countable choice, trichotomy, or ambient completeness.

Definition 1787.1 (Apartness completion carrier)**.** *An* $\mathsf{ApartnessCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, Q, M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the apartness source row, carrying the positive separation evidence used by the completion boundary.*
2. *Q is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion route attached to the same source row.*
3. *M is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *completion route through which the separated source is replayed.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposed to consumers of the completion read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback extracted from those stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested positive gaps, radii, and comparison cells.*
7. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal handoff, present only when the generic completion route is consumed by a Real-facing boundary.*
8. *H records componentwise* hsame *transport for the apartness source, separated route, completion route, window schedule, readback row, tolerance ledger, optional seal, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from apartness inspection through separated completion, constructive metric completion, stream windows, regular readback, dyadic tolerance, and the optional Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{ApartnessCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion classes, host equality of completed points, selected limits, countable-choice schedules, trichotomy rows, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ApartnessCompletionUp.ApartnessCompletionTasteGate single carrier alignment`

Theorem 1787.2 (Apartness completion NameCert obligations)**.** *For every accepted* $\mathsf{ApartnessCompletion}^{\uparrow}$ *carrier $A = (X, Q, M, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ApartnessCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed apartness source row, separated completion route, constructive metric completion route, stream windows, regular readback, dyadic tolerance ledger, optional Real seal, transport, replay, provenance, and local naming rows;*

2. *apartness-source exactness, requiring the positive separation row X to be displayed before any separated completion route Q or completion route M is consumed;*

3. *finite readback discipline, requiring each consumer observation to pass through S, R, and D before the optional seal E is available;*

4. *componentwise stability, requiring H and C to transport and replay only the displayed rows $X, Q, M, S, R, D, E, P, N$;*

5. *provenance exhaustion, requiring P and N to export no quotient completion class, host equality, selected limit, countable-choice schedule, trichotomy row, or ambient-completeness principle.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1787.1. The tuple displays the apartness source, separated completion route, constructive metric completion route, finite windows, regular readback, dyadic tolerance ledger, optional Real seal, transport, replay, provenance, and local name.

Apartness-source exactness follows from the order of the displayed route. A consumer can inspect Q or M only after the positive separation row X has been named, so separated completion cannot hide its apartness source.

The readback discipline is carried by S, R, D, E. Each completion observation first opens a finite stream window, then reads the regular rational row, then consults the dyadic tolerance ledger; the optional Real seal is downstream of that finite route.

Componentwise stability and provenance exhaustion are the roles of H, C, P, N. These rows transport, replay, package, and name only the displayed packet, and therefore cannot add quotient completion classes, host equality, selected limits, countable choice, trichotomy, or ambient completeness. □

Theorem 1787.3 (Apartness completion Real seal boundary)**.** *Let $A = (X, Q, M, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{ApartnessCompletion}^{\uparrow}$ *packet. Every Real-facing completion read of A, when the optional seal row E is present, factors through the finite route*

$$X, Q, M, S, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal receives only the displayed apartness source, Hausdorff separated route, constructive metric completion route, stream windows, regular readback, dyadic tolerance ledger, transport rows, continuation replay, provenance, and local NameCert row. It receives no quotient completion, host equality of completed points, selected limit, countable-choice schedule, trichotomy row, or ambient-completeness coordinate.

Proof. Start with a Real-facing completion read. Projection of the carrier gives a single ordered route from the apartness source row X through the separated route Q and constructive metric completion route M.

The finite observation rows then exhaust the readback path. The window schedule S, regular readback R, and dyadic tolerance ledger D must be displayed before the Real seal row E can be inspected.

Finally, the transport, replay, provenance, and local name rows H, C, P, N carry the same route without adding coordinates. Row exhaustion therefore blocks quotient completions, host equality, selected limits, countable choice, trichotomy, and ambient completeness from entering the Real seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1787.1, Definition 1787.2, and Definition 1787.3 seed the generic separated apartness-completion boundary over HausdorffCompletion, ConstructiveMetricCompletion, StreamName, RegSeqRat, DyadicRatCore, optional Real, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host equality of completed points, selected limits, countable choice, trichotomy, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, apartness-source exactness, finite readback discipline, componentwise stability, provenance exhaustion, and Real seal boundary proofs for BEDC.Derived.ApartnessCompletionUp.

Constructive story: An $\mathsf{ApartnessCompletion}^{\uparrow}$ packet is generated from an apartness source row, a HausdorffCompletion separated route, a ConstructiveMetricCompletion route, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an optional RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1788

A Concrete Naming Certificate for ApartnessCotransitivity$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the ApartnessCotransitivity$^{\uparrow}$ packet records the finite cotransitive apartness read by which an apart pair of real names separates through a third displayed name on either the left branch or the right branch. It sits over the Bishop-style apartness surface of chapter 304, the located apartness classifier of chapter 1380, the regular Cauchy order/apartness route of chapter 1710, the regular rational handoff of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the terminal real-name seal of chapter 38. Its role is to expose cotransitivity as finite NameCert data without importing trichotomy, ambient real equality, quotient streams, selected representatives, countable choice, or Real$^{\uparrow}$ completeness.

Definition 1788.1 (Apartness cotransitivity carrier)**.** *An* ApartnessCotransitivity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, L, R, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the left* Real$^{\uparrow}$ *name whose apartness from Y is being tested by a displayed regular Cauchy route.*
2. *Y is the right* Real$^{\uparrow}$ *name in the source apart pair.*
3. *Z is the third* Real$^{\uparrow}$ *name through which cotransitivity is read.*
4. *L is the left-branch row. When active, it records the witnessed apartness route from X to Z through the located apartness classifier and the regular Cauchy apartness-order surface.*
5. *R is the right-branch row. When active, it records the witnessed apartness route from Z to Y through the same classifier and regular Cauchy apartness-order surface.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row. It records the positive dyadic separation budget, slack, and finite precision data consumed by whichever branch is displayed.*
7. *H records componentwise* hsame *transport for the three real-name rows, the two branch rows, the dyadic tolerance row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an apartness-cotransitivity request through $X, Y, Z, L, R, D, H, P, N$.*
9. *P is the* Pkg *provenance over* ApartnessCotransitivity$^{\uparrow}$*,* ApartnessReal$^{\uparrow}$*,* LocatedApartness$^{\uparrow}$*,* RegularCauchyApartne RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessCotransitivity}^{\uparrow}}$ *row.*

The classifier admits exactly the finite packet in which the source apartness of X and Y is witnessed by one displayed branch, either L or R, together with the tolerance and structural rows needed to replay that branch. It has no coordinate for trichotomy, host real equality, quotient streams, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1788.2 (Apartness cotransitivity NameCert obligations)**.** *For every accepted* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *carrier*

$$K = (X, Y, Z, L, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApartnessCotransitivity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two source real names, the third real name, the left and right branch rows, the dyadic tolerance row, transport, replay, provenance, and local naming rows;*
2. *branch stability under classifier transport, requiring H and C to preserve the displayed left-or-right branch tag and to transport the active branch through the same located-apartness and regular Cauchy apartness-order route;*
3. *dyadic tolerance preservation, requiring D to carry the positive separation budget, slack, and finite precision data consumed by the active branch before any* $\mathsf{Real}^{\uparrow}$ *seal is read;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring each active branch to pass through a displayed regular rational readback before it is treated as a real-name apartness read;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring the terminal real-name rows X, Y, Z to remain endpoints and branch parameters rather than sources of host equality, quotient equality, selected representatives, trichotomy, countable choice, or completeness;*
6. *ledger exclusion, requiring P and N to name exactly the displayed cotransitivity packet and to export no trichotomy, no host real equality, no quotient streams, no countable choice, no selected representatives, and no* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No apartness-cotransitivity obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.ApartnessCotransitivityUp.ApartnessCotransitivityTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1788.1. The accepted packet displays the source names X, Y, the third name Z, the two branch rows L, R, the dyadic tolerance row D, and the structural rows H, C, P, N.

Branch stability is componentwise. Transport may move the displayed names and structural rows, but it must preserve the branch tag and replay the same left-or-right route through the located apartness classifier and the regular Cauchy apartness-order surface.

The dyadic and regular handoff clauses are the finite middle of the route. The active branch consumes D before it reaches a regular rational readback, and the $\mathsf{RegSeqRat}^{\uparrow}$ row is the point at which finite Cauchy data is handed to the real-name seal.

The Real-seal and ledger clauses follow from the inventory. Since the only available rows are $X, Y, Z, L, R, D, H, C, P, N$, the packet cannot export trichotomy, host equality, quotient streams, choice, selected representatives, or completeness. □

Theorem 1788.3 (Apartness cotransitivity Real boundary)**.** *Every accepted* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *read with source apartness between* X *and* Y *and third real name* Z *reaches a Real-facing consumer only through the finite route*

$$\mathsf{ApartnessReal}^{\uparrow} \longmapsto \mathsf{LocatedApartness}^{\uparrow} \longmapsto \mathsf{RegularCauchyApartnessOrder}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto$$

The consumer receives exactly one displayed cotransitive branch: either the left branch witnessing X *apart from* Z*, or the right branch witnessing* Z *apart from* Y*. The consumer receives no trichotomy of* X, Z, Y*, no host real equality, no quotient stream equality, no countable-choice selection of branches, no selected representatives, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier $K = (X, Y, Z, L, R, D, H, C, P, N)$. The source apartness and third real name are visible only as displayed real-name rows and branch parameters.

The active branch is finite. If the left row is active, the read follows L through the located apartness classifier, the regular Cauchy apartness-order route, the regular rational handoff, and the dyadic tolerance row. If the right row is active, the same route is followed through R.

In both cases the Real-facing consumer reaches the endpoint names only after the displayed branch, regular handoff, and dyadic tolerance data have been opened. The structural rows H, C, P, N transport, replay, package, and name that same branch.

No unlisted principle can be read from the packet. Trichotomy, host equality, quotient streams, countable-choice branch selection, selected representatives, and ambient completeness are absent from the carrier and from the local NameCert obligations. □

Falsifiable prediction: Every $\mathsf{ApartnessCotransitivity}^{\uparrow}$ read factors through three RealUp name rows, one left-or-right branch row, one DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: apartnessreal, located-apartness, regular-cauchy-apartness-order, regseqrat, dyadicratcore, real: $\mathsf{ApartnessCotransitivity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source apart pair, a third RealUp name, an explicit left-or-right branch witness, dyadic tolerance data, regular rational handoff, and the ledger exclusions that keep cotransitivity separate from trichotomy and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessCotransitivity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessCotransitivity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1788.1, Definition 1788.2, and Definition 1788.3 seed the finite Bishop-style cotransitivity boundary for Real and RegSeqRat apartness through displayed branch, dyadic tolerance, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessCotransitivity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, host real equality, quotient stream equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch stability under classifier transport, dyadic tolerance preservation, RegSeqRat handoff, Real-seal non-escape, and ledger exclusion for BEDC.Derived.ApartnessCotransitivityUp.

Constructive story: An $\mathsf{ApartnessCotransitivity}^{\uparrow}$ packet is generated from two source RealUp names, one third RealUp name, a displayed left-or-right apartness branch, one DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1789

A Concrete Naming Certificate for $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$

Following Kelley and Bourbaki on Moore–Smith nets, filters, and uniform spaces, the CauchyNetFilterCorrespondence packet records the finite correspondence between Cauchy-net and Cauchy-filter completion routes as BEDC naming data. It sits over the directed-net surface of chapter 589, the filter-base surface of chapter 333, the uniform entourage surface of chapter 339, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the uniform-net route of chapter 1657. The chapter packages the correspondence as one finite NameCert object; it is not a quotient of nets, not a quotient of filters, not a selected limit, not countable choice, not host equality, and not ambient completeness.

Definition 1789.1 (Cauchy net-filter correspondence carrier)**.** *A* $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, F, U, W, R, E, J, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyNet}^{\uparrow}$ *row, recording the directed index surface, displayed tail witnesses, and finite Cauchy-net requests available before any filter read is made.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, recording the filter-base trace generated from the displayed tails and the reverse filter request surface that can be read back as net data.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, storing the selected entourage requests, symmetric or composed tolerance cells, and finite comparison budget shared by the net and filter reads.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite windows inspected by the displayed K, F, U route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from those windows and tolerance cells.*
6. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, reachable only after K, F, U, W, R have been consumed.*
7. *J is the common-refinement ledger, listing the finite tail/filter-base refinements through which the net-to-filter and filter-to-net reads are compared.*
8. *H records componentwise* hsame *transport for the Cauchy-net row, Cauchy-filter row, uniform entourage row, window row, readback row, seal boundary, common-refinement ledger, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite Cauchy request through K, F, U, W, R, J and then to the seal row E.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyUniformNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient net equality, quotient filter equality, selected limits, countable-choice refinements, host equality, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyNetFilterCorrespondenceUp.CauchyNetFilterCorrespondenceTasteGate`
`single carrier alignment`

Theorem 1789.2 (Cauchy net-filter correspondence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$ *carrier*

$$Q = (K, F, U, W, R, E, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier exposure: the accepted object exposes only the displayed packet Q with rows $K, F, U, W, R, E, J, H, C, P, N$*
2. *net-to-filter routing: every read from a displayed Cauchy-net tail to a filter-base trace passes through K, F, U before any finite window is inspected;*
3. *filter-to-net routing: every read from a displayed filter-base request back to a net tail passes through the same F, K, U rows before any finite window is inspected;*
4. *common-refinement comparison: both directions compare their visible outputs only through the ledger J, then through W, R, E when a real or completion-facing seal is requested;*
5. *provenance and name exhaustion: H, C, P, N transport, replay, name, and exhaust exactly this finite route and export no quotient nets, quotient filters, selected limits, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1789.1. The packet displays the Cauchy-net row, Cauchy-filter row, uniform entourage row, finite windows, regular rational readback, Real seal boundary, common-refinement ledger, and structural rows H, C, P, N.

The two routing clauses are read from the same finite packet. A net-to-filter read starts at the displayed tail witnesses in K, opens the filter-base trace F generated by those tails, and carries the selected entourage budget through U before any window in W is available. A filter-to-net read starts at the displayed request in F, returns to the net-tail row K, and uses the same U budget before it reaches W.

The comparison clause is forced by J. The ledger records the common refinements already displayed by both directions, so the two reads can be compared only at those visible refinements and then transported through the same W, R, E route when a seal-facing consumer asks for a real or completion boundary.

Finally, H and C transport and replay the listed route componentwise, while P and N name only those rows. Since the carrier has no excluded quotient, choice, host-equality, completeness, verification, or bridge coordinate, the five obligation rows exhaust the local certificate surface. □

Theorem 1789.3 (Cauchy net-filter common refinement)**.** *Let $Q = (K, F, U, W, R, E, J, H, C, P, N)$ be an accepted* $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$ *carrier. Every net-to-filter read and every filter-to-net read of Q compare their visible outputs only through displayed common refinements in J. In particular, a comparison has the finite route*

$$K, F, U \longmapsto J \longmapsto W, R, E, H, C, P, N,$$

and no comparison may use quotient net equality, quotient filter equality, a selected subnet, a selected ultrafilter, countable choice, host equality, or ambient completeness.

Proof. Project the accepted carrier. The rows K and F are both present before a comparison can be made, and the uniform budget row U is shared by the two directions.

Apply the routing obligations of Definition 1789.2. Each direction must reach the same finite comparison boundary before reading windows or rational readbacks. That boundary is exactly the ledger J.

Once J is fixed, the downstream route is unique in the carrier: windows are read through W, regular rational data through R, and the seal boundary through E, with H, C, P, N carrying transport, replay, provenance, and local naming. No row outside this displayed route can witness a comparison, so quotient, choice, host-equality, and completeness principles are unavailable. □

Theorem 1789.4 (Cauchy net-filter real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$ *carrier must factor through the same finite route*

$$K, F, U, J, W, R \longmapsto E$$

before it reads the seal boundary. Thus the consumer receives a Cauchy-net row, a Cauchy-filter row, a uniform entourage row, the displayed common-refinement ledger, finite StreamName windows, regular rational readback, and the Real seal, but receives no quotient net, quotient filter, selected limit, countable-choice selector, host equality, ambient completeness, Lean verification, or bridge checking.

Proof. The seal row E is a downstream coordinate of the carrier in Definition 1789.1. It is not available before the rows K, F, U, W, R, J have been displayed.

Use Definition 1789.3. Any route that compares the net and filter presentations must pass through J, and the only seal-facing route after that comparison is through the finite windows W and regular rational readback R.

Therefore a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reaches E only by the displayed route $K, F, U, J, W, R \longmapsto E$. The structural rows H, C, P, N preserve and name that route, and the absence of quotient, choice, host-equality, completeness, verification, and bridge rows prevents any alternate seal handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1789.1, Definition 1789.2, Definition 1789.3, and Definition 1789.4 seed the finite correspondence route tying Cauchy-net and Cauchy-filter completion reads through a shared uniform budget, common-refinement ledger, StreamName windows, RegSeqRat readback, and Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, quotient filters, selected limits, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, net-to-filter routing, filter-to-net routing, common-refinement comparison, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyNetFilterCorr

Constructive story: A CauchyNetFilterCorrespondence$^{\uparrow}$ packet is generated from one CauchyNet row, one CauchyFilter row, one UniformSpace entourage row, finite StreamName windows, RegSeqRat readback, a Real seal boundary, a common-refinement ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1790

A Concrete Naming Certificate for $\mathsf{SeparatedMetricReflection}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Bourbaki universal-mapping language, the $\mathsf{SeparatedMetricReflection}^{\uparrow}$ packet records the separated-reflection route consumed by L10 completion uniqueness. It reads the separated classifier surface of chapter 1411, the completion adjunction route of chapter 1456, the extension boundary of chapter 397, the reflection consumer of chapter 403, and the terminal real-name seal of chapter 38. The packet records this route as finite NameCert data before any consumer treats zero-distance evidence as host equality or quotient-space structure.

Definition 1790.1 (SeparatedMetricReflection carrier)**.** *A* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, S, A, U, Z, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier and metric-distance witness surface available before separation is read.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier row, exposing only the finite route from a displayed zero-distance witness to point-classifier equivalence.*
3. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *completion route that carries dense-row admission, completion-seal access, extension, and uniqueness by dense agreement.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row read by completion-facing consumers after the separated classifier has been routed through A.*
5. *Z is the reflected classifier output, obtained by reading S through the completion route and then returning the classifier evidence at the separated-reflection surface.*
6. *H stores componentwise* hsame *transport for the metric source, separated classifier, completion adjunction, universal row, reflected classifier, continuation row, package row, and local naming row.*
7. *C stores the* Cont *replay from metric-distance witnesses to zero-distance readings, from those readings to the separated classifier, and from the separated classifier through the completion universal route to Z.*

8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, CompletionUniversalPropert $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedMetricReflection}^{\uparrow}}$ *row.*

The carrier exposes separated reflection only as a finite classifier route. It carries no quotient metric space, no host equality coordinate, no trichotomy row, no open-cover compactness principle, no countable-choice selector, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionTasteGate single carrier alignment`

Theorem 1790.2 (SeparatedMetricReflection NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{SeparatedMetricReflection}^{\uparrow}}$ *surface for Definition 1790.1 consists exactly of five finite obligation rows:*

1. *carrier exposure: an accepted object exposes only the packet $R = (X, S, A, U, Z, H, C, P, N)$;*
2. *zero-distance classifier exactness: every classifier read begins with the metric-distance witness in X, passes through the zero-distance classifier row S, and reaches the reflected classifier output Z;*
3. *adjunction compatibility: the completion-facing part of the route is read through the dense-row and extension coordinates carried by A;*
4. *universal-route preservation: the extension row U preserves the order* $\mathsf{Met}^{\uparrow} \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{CauchyCompletionAdjunction}^{\uparrow} \to \mathsf{CompletionUniversalProperty}^{\uparrow}$ *before a consumer reads Z;*
5. *ledger non-escape: the provenance row P exports no quotient metric space, host equality, trichotomy, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Thus the packet binds the separated-reflection boundary used by completion uniqueness without adding quotient, host-equality, choice, compactness, or ambient-completeness data.

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier namecert obligations`

↪ variant `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionNameCertObligations`

Proof. Carrier exposure is projection from the tuple in Definition 1790.1. The source row is X, the separated classifier is S, the completion adjunction is A, the universal row is U, the reflected classifier is Z, and the structural rows are H, C, P, N.

Zero-distance classifier exactness follows by reading the displayed route in order. A metric-distance witness starts in X, the zero-distance classifier row S converts that witness into point-classifier evidence, and the output coordinate Z records the reflected classifier read.

Adjunction compatibility and universal-route preservation are the completion-facing clauses. The row A is the only dense-row and extension route carried by the packet, while U keeps the order of the route fixed before any consumer reads the reflected classifier.

Ledger non-escape is an inventory check on P. Since quotient metric spaces, host equality, trichotomy, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking are not fields of R, no local NameCert obligation can use them. □

Theorem 1790.3 (SeparatedMetricReflection zero-distance exactness)**.** *For every accepted* SeparatedMetricReflecti *carrier $R = (X, S, A, U, Z, H, C, P, N)$, any reflected classifier read starts with metric-distance evidence in X, passes through the zero-distance classifier row S, then through the completion route*

rows A, U, and reaches the reflected classifier output Z. The read exports this finite classifier route only; it does not conclude quotient metric space structure or host equality.

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier zero distance exactness`

Proof. Project the carrier of Definition 1790.1. The row X is the only metric-distance source, S is the only zero-distance classifier row, and Z is the only reflected classifier output.

Apply the zero-distance classifier exactness clause of Definition 1790.2. It forces every classifier read to begin at X and pass through S before Z is available.

The universal route theorem then fixes the completion-facing segment. Rows A and U preserve the separated classifier through the completion adjunction and universal extension route, while the provenance row contains no quotient metric space or host-equality coordinate. □

Theorem 1790.4 (SeparatedMetricReflection universal route)**.** *Every completion-facing consumer of an accepted* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Met}^{\uparrow} \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{CauchyCompletionAdjunction}^{\uparrow} \to \mathsf{CompletionUniversalProperty}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{NameCert}$$

before reading the reflected classifier Z. The consumer may inspect the metric source row, the zero-distance classifier row, the completion adjunction route, the universal extension row, the real-name seal, the structural hsame *and* Cont *rows, the package provenance, and the local naming row. It may not bypass the route by reading quotient metric spaces, host equality, trichotomy, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier of Definition 1790.1. Its public consumer-facing rows occur in the order X, S, A, U, Z, with H, C, P, N supplying transport, replay, provenance, and local admission.

The metric source row X supplies distance witnesses, and S turns the relevant zero-distance readings into separated classifier evidence. The row A moves that evidence to the completion-facing boundary, while U records the universal extension surface a consumer is allowed to read.

The real-name seal and the local NameCert row are read only after this route has been preserved. Because P names no quotient, host-equality, trichotomy, open-cover compactness, choice, or ambient-completeness field, every accepted completion-facing read factors through the displayed chain before reaching Z. □

Theorem 1790.5 (SeparatedMetricReflection Cauchy uniqueness)**.** *Let*

$$R = (X, S, A, U, Z, H, C, P, N)$$

be an accepted $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *packet. Suppose two completion-facing reads of R agree on the separated zero-distance classifier row Z and on the completion adjunction and universal rows A, U. Then the two reads have the same reflected classifier output along the displayed route*

$$H, C, P, N.$$

Equivalently, the packet supplies Cauchy uniqueness only through its separated zero-distance classifier, adjunction route, universal route, componentwise transport, replay, provenance, and local naming rows; it supplies no quotient metric-space equality or host equality coordinate.

Proof. Project the carrier $R = (X, S, A, U, Z, H, C, P, N)$ from Definition 1790.1. The relevant completion-facing rows are the zero-distance classifier route through S, Z and the adjunction and universal rows A, U.

Apply Definition 1790.3. Agreement on the separated zero-distance classifier row forces both reads to pass through the same X, S, Z classifier route before any completion-facing output is available.

Next apply Definition 1790.4. Agreement on A, U keeps the completion adjunction and universal extension segment fixed, so the two reads reach the reflected classifier through the same displayed completion route.

Finally transport componentwise through H and replay the same displayed continuation through C, with provenance and local naming supplied by P, N. Since the packet contains no quotient metric-space row and no host equality coordinate, the common reflected classifier output is the only uniqueness conclusion available. □

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier cauchy uniqueness`
Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier universal route`

Theorem 1790.6 (SeparatedMetricReflection completion-uniqueness route)**.** *Every completion uniqueness consumer of an accepted* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *carrier*

$$R = (X, S, A, U, Z, H, C, P, N)$$

must expose the zero-distance classifier S, completion adjunction A, universal extension row U, reflected classifier Z, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before using uniqueness by separated reflection. The consumer receives the displayed separated-reflection route only; it receives no quotient metric space, host equality, trichotomy, choice principle, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with carrier projection. The completion uniqueness consumer can inspect only the rows $X, S, A, U, Z, H, C, P, N$ displayed in Definition 1790.1.

Use the NameCert obligations and zero-distance exactness. The obligations force the consumer to read the zero-distance classifier S as a finite row of the same packet, while Definition 1790.3 routes the classifier read from X through S and onward to Z.

Next apply Definition 1790.4. It places the completion-facing part of the read through A and U before the reflected classifier is consumed by a uniqueness argument.

Finally read the structural rows H, C, P, N. Transport and replay preserve the displayed route, provenance records the packet source, and local naming admits the route as $\mathsf{SeparatedMetricReflection}^{\uparrow}$ data. Since the carrier has no quotient, host-equality, trichotomy, choice, open-cover, or ambient-real-completeness coordinate, the uniqueness consumer cannot import any of those objects. □

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier completion uniqueness route`

Theorem 1790.7 (SeparatedMetricReflection obligation-closure package)**.** *Let*

$$R = (X, S, A, U, Z, H, C, P, N)$$

be an accepted $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *packet. Then Definition 1790.1, Definition 1790.2, and Definition 1790.4 assemble the five finite obligation rows required for obligation closure: carrier exposure, zero-distance classifier exactness, adjunction compatibility, universal-route preservation, and ledger non-escape for the same packet R.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier obligation closure package`

Proof. Project $R = (X, S, A, U, Z, H, C, P, N)$ from Definition 1790.1. The carrier exposure row is the displayed tuple itself: metric source X, separated classifier S, completion adjunction A, universal route U, reflected classifier Z, transport H, replay C, provenance P, and local naming N.

Zero-distance classifier exactness is the first named obligation in Definition 1790.2. A distance witness is read from $\mathsf{Met}^{\uparrow}$ through X, routed to the zero-distance classifier in $\mathsf{SeparatedMetric}^{\uparrow}$ through S, and returned as the reflected classifier output Z.

Adjunction compatibility and universal-route preservation are supplied by the completion-facing rows. The route passes through $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ in A and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ in U, so completion consumers read dense-row, extension, and uniqueness data only after the separated classifier route has been displayed.

Finally apply the universal-route theorem. It keeps every completion-facing read on the chain through $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while the provenance row P excludes quotient metric spaces, host equality, trichotomy, open-cover compactness, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1790.8 (SeparatedMetricReflection scoped dependency route)**.** *Every accepted* SeparatedMetricReflectio*packet whose obligation rows are read for completion uniqueness factors through the displayed route*

$$\mathsf{Met}^{\uparrow} \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{CauchyCompletionAdjunction}^{\uparrow} \to \mathsf{CompletionUniversalProperty}^{\uparrow} \to \mathsf{Real}^{\uparrow} \to \mathsf{BHist} \to \mathsf{hs}$$

The route may read the metric source, separated classifier, completion adjunction, universal extension, real-name seal, history packet, transport, continuation, package provenance, and local naming rows of the same carrier. It has no quotient metric space, host equality, trichotomy, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row available.*

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier scope`

Proof. Project the carrier $R = (X, S, A, U, Z, H, C, P, N)$ from Definition 1790.1. The rows X, S, A, U, Z provide the metric, separated classifier, completion adjunction, universal extension, and reflected output coordinates, while H, C, P, N provide transport, replay, provenance, and local naming.

Use the NameCert obligations of Definition 1790.2. They force carrier exposure, zero-distance classifier exactness, adjunction compatibility, universal-route preservation, and ledger non-escape for the same packet.

Now apply the universal route theorem Definition 1790.4. It places every completion-facing consumer on the chain through $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the reflected classifier is read.

It remains only to read the structural rows. The history, hsame, Cont, Pkg, and NameCert coordinates are exactly R's finite support rows, and ledger non-escape from P excludes quotient metric spaces, host equality, trichotomy, choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.SeparatedMetricReflectionUp.SeparatedMetricReflectionCarrier scoped dependency route`

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedMetricReflection}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{SeparatedMetricReflection}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ source row, a $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance classifier row, a $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ completion route, a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ extension

row, a reflected classifier output, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SeparatedMetricReflection}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1790.1, Definition 1790.2, Definition 1790.4, Definition 1790.5, and Definition 1790.6 bind the completion-uniqueness and universal-route coverage for separated reflection over the displayed metric, separated classifier, completion-adjunction, universal-property, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparatedMetricReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient metric spaces, host equality, trichotomy, open-cover compactness, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridged closure requires an explicit standard bridge for the separated-reflection schema, preserving the displayed route through Met$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, CauchyCompletionAdjunction$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert without adding quotient or host-equality data.

Chapter 1791

A Concrete Naming Certificate for $\mathsf{CauchyEntourageFilter}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures and filter bases, and Bishop–Bridges on constructive Cauchy completion, the $\mathsf{CauchyEntourageFilter}^{\uparrow}$ packet names the finite boundary between displayed entourage data and Cauchy-filter completion consumers. It sits over the uniform entourage surface of chapter 339, the basis-generator surface of chapter 1501, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion consumer boundary of chapter 1588. The packet is a finite induction surface for Cauchy-entourage requests; it is not a quotient filter, not a selected limit, not countable choice, not host equality, and not an ambient completeness principle.

Definition 1791.1 (Cauchy entourage-filter carrier)**.** *A* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row. It records the finite entourage filterbase, diagonal, refinement, symmetry, and composition data from which every Cauchy-entourage request starts.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row. It turns displayed entourage requests into finite Cauchy-filter basis elements before any completion-facing consumer can read them.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by those generated basis elements.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite windows in S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare entourage radii with the regular rational readbacks.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after U, B, S, R, D have been replayed.*
7. *H records componentwise* hsame *transport for the entourage row, basis generator, finite windows, regular readback, dyadic ledger, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an entourage request through U, then through B, S, R, D, and finally to the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyEntourageFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyEntourageFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed route $U, B, S, R, D, E, H, C, P, N$. The carrier has no coordinate for quotient filters, selected limits, countable-choice selectors, host equality, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyEntourageFilterUp.CauchyEntourageFilterTasteGate single carrier alignment`

Theorem 1791.2 (Cauchy entourage-filter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *carrier*

$$Q = (U, B, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyEntourageFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the displayed uniform entourage row, CauchyFilterBasis generator, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming row;*
2. *entourage-to-basis routing, requiring every Cauchy request read from U to pass through the finite generator row B before any window or rational readback is available;*
3. *dyadic readback comparison, requiring every generated basis element to inspect S and R before the tolerance ledger D compares the requested entourage scale;*
4. *completion handoff, requiring every* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer named by P to read the seal boundary E only after the route U, B, S, R, D has been replayed;*
5. *transport, replay, and provenance exhaustion, requiring H, C, P, N to preserve and name exactly this finite route and to export no quotient filter, selected limit, countable-choice selector, host equality, ambient completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1791.1. The projection exposes exactly the uniform entourage row, the basis generator, the finite windows, the regular readback, the dyadic tolerance ledger, the Real seal, and the structural rows H, C, P, N.

The entourage-to-basis route is fixed by the displayed row order. A Cauchy request begins at the finite entourage filterbase in U and is admitted to completion-facing use only after the generator row B has produced a displayed basis element.

The dyadic comparison is downstream of that generator. The basis element can inspect windows only through S, can read regular rational data only through R, and can compare the requested scale only through the ledger D.

The completion handoff is terminal for this packet. The provenance row P may mention $\mathsf{CompleteUniformSpace}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only as consumers of the already replayed route $U, B, S, R, D \longmapsto E$.

Finally, H transports each listed coordinate componentwise and C replays the same dependency order. Since the carrier has no quotient, choice, selected-limit, host-equality, completeness, verification, or bridge coordinate, P and N exhaust the local certificate surface. □

Theorem 1791.3 (Cauchy entourage-filter dyadic basis handoff)**.** *Let $Q = (U, B, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *carrier. Every generated Cauchy-filter basis read factors through the finite route*

$$U \longmapsto B \longmapsto S \longmapsto R \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

Thus an entourage request reaches the Cauchy-filter basis surface only with its StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance ledger displayed. It cannot bypass D by selecting a real representative, quotienting a filter, choosing a limit, or invoking ambient completeness.

Proof. Start with a generated basis read of the packet. The row U supplies only finite uniform-space entourage data, so the request first names a displayed entourage demand and its refinement ledger.

The row B consumes that demand as a finite Cauchy-filter basis generator. It does not create a completed filter object and does not expose a limit by itself.

The regular readback route is then forced. The generated basis element inspects finite windows through S, reads regular rational observations through R, and checks the dyadic scale and inclusion data through D.

Only after this route has been replayed can the seal row E and the structural rows H, C, P, N be read. Hence no selected representative, quotient filter, selected limit, choice schedule, host equality, or ambient completeness witness is available as a bypass. □

Theorem 1791.4 (Cauchy entourage-filter Real seal non-escape)**.** *For every accepted* $\mathsf{CauchyEntourageFilter}^{\uparrow}$ *carrier* $Q = (U, B, S, R, D, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{CompleteUniformSpace}^{\uparrow}$*-facing consumer exposed by* P *factors through the displayed route*

$$U, B, S, R, D \longmapsto E.$$

Consequently the packet cannot export a Real seal, completion boundary, quotient filter, selected limit, countable-choice selector, host equality, or ambient completeness principle from any earlier row.

Proof. Project the provenance row P. Its $\mathsf{Real}^{\uparrow}$ and $\mathsf{CompleteUniformSpace}^{\uparrow}$ occurrences are recorded as downstream consumers of the displayed entourage-filter route, not as separate coordinates.

The row E is available only after the route through U, B, S, R, D has been replayed. The rows S and R expose finite observations, and D supplies the dyadic tolerance comparison that ties those observations back to the original entourage request.

Transport and replay preserve this order. Componentwise hsame in H cannot introduce a new seal coordinate, and Cont replay in C repeats the same route for every accepted request.

Thus the local naming row N names a finite boundary between entourage data and completion consumers. It has no earlier point from which a Real seal, completion boundary, quotient filter, selected limit, choice schedule, host equality, or ambient completeness principle can escape. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyEntourageFilter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyEntourageFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1791.1, Definition 1791.2, Definition 1791.3, and Definition 1791.4 seed the finite boundary through which UniformSpace entourage data reaches CauchyFilterBasis, StreamName, RegSeqRat, DyadicRatCore, Real, and CompleteUniformSpace consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyEntourageFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, entourage-to-basis routing, dyadic readback comparison, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyEntourageFilterUp.

Constructive story: A $\mathsf{CauchyEntourageFilter}^{\uparrow}$ packet is generated from one UniformSpace entourage row, one CauchyFilterBasis generator row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1792

A Concrete Naming Certificate for $\mathsf{FiniteIntersectionProperty}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style compactness practice, the $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ packet records only the finite closed-family certificate used by the L10 Real-and-Completion route. It reads the closed-interval completion surface of chapter 1778, the nested-window compactness handoff of chapter 1584, the Cantor nested-intersection consumer of chapter 1421, the finite epsilon-net witness of chapter 1767, and the terminal real-name seal of chapter 38. The packet names finite closed-family intersection data; it does not import open-cover compactness, arbitrary closed-family compactness, countable choice, selected endpoints, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1792.1 (Finite intersection property carrier)**.** *A* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (J, F, A, Q, N, C, E, R, H, T, P, M)$$

with the following displayed rows:

1. *J is the finite family index row, recording the inspected closed-family cells and the finite subfamily request consumed by the packet.*
2. *F is the closed-family row, giving each displayed member as a closed interval or closed interval subpacket over the same Real-and-Completion route.*
3. *A is the finite-acceptance ledger, requiring every index in J to have a displayed closed-family member in F and requiring the finite subfamily request to be read only through those cells.*
4. *Q is the* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *row, supplying the endpoint-bounded Cauchy completion handoff for the displayed closed interval cells.*
5. *N is the* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *row, used when the finite family is routed through a nested-window refinement rather than through a general cover argument.*
6. *C is the* $\mathsf{CantorIntersection}^{\uparrow}$ *row, recording the finite Cantor-style nested-intersection handoff downstream of the displayed nested-window data.*
7. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row or interval-net witness row, recording the finite net, mesh, or interval-window data used to keep the finite family request inside visible compactness-shaped data.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after J, F, A, Q, N, C, E have been displayed.*

9. *H records componentwise* hsame *transport for the finite index, closed-family, completion, nested-window, Cantor handoff, finite-net, Real seal, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay from a finite closed-family request through $J, F, A, Q, N, C, E, R, H, P, M$.*
11. *P is the* Pkg *provenance over* FiniteIntersectionProperty$^{\uparrow}$, ClosedIntervalCauchyCompletion$^{\uparrow}$, NestedIntervalCompa CantorIntersection$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *M is the local* NameCert$_{\text{FiniteIntersectionProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J, F, A, Q, N, C, E, R, H, T, P, M$. The carrier has no coordinate for open-cover compactness, arbitrary closed-family compactness, countable choice, selected endpoints, quotient real equality, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.FiniteIntersectionPropertyUp.FiniteIntersectionPropertyUp`

Theorem 1792.2 (Finite intersection property NameCert obligations)**.** *For every accepted* FiniteIntersectionProper *carrier*

$$K = (J, F, A, Q, N, C, E, R, H, T, P, M),$$

the local NameCert$_{\text{FiniteIntersectionProperty}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite family index, closed-family cells, finite-acceptance ledger, closed-interval Cauchy completion row, nested-window compactness row, Cantor handoff row, finite epsilon-net or interval-net witness, Real seal, transport, replay, provenance, and local naming rows;*
2. *finite-family exactness, requiring every accepted intersection request to name a finite subfamily through J and to read each member through the displayed closed-family row F;*
3. *closed-interval completion handoff, requiring Q to be consumed before any nested-window, Cantor, finite-net, or Real-facing conclusion is exported;*
4. *compactness-shape locality, requiring N, C, E to be read as displayed finite nested-window, Cantor handoff, and finite-net or interval-net rows rather than as an open-cover compactness theorem;*
5. *Real seal routing, requiring R to remain downstream of J, F, A, Q, N, C, E and upstream only of H, T, P, M;*
6. *componentwise classifier transport, displayed replay, provenance, and local naming for the same rows through H, T, P, M;*
7. *ledger non-escape, requiring the packet to export no open-cover compactness, arbitrary closed-family compactness, countable-choice schedule, selected endpoint, quotient real equality, trichotomy branch, or ambient* Real$^{\uparrow}$ *completeness theorem.*

No finite intersection property obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FiniteIntersectionPropertyUp.FiniteIntersectionPropertyNameCertObligations`

Proof. Carrier admission is projection from Definition 1792.1. The visible packet consists exactly of the finite family index J, the closed-family row F, the finite-acceptance ledger A, the completion row Q, the nested-window row N, the Cantor handoff row C, the finite-net or interval-net witness E, the Real seal R, and the structural rows H, T, P, M.

Finite-family exactness is rowwise. Every accepted request first names its finite subfamily through J, and every named member is read through F before any downstream intersection certificate is visible.

The completion and compactness-shaped handoffs are then ordered. The row Q supplies the closed-interval Cauchy completion boundary; the rows N, C, E consume that displayed route as nested-window, Cantor-style, and finite-net data rather than as a general open-cover theorem.

Finally, R, H, T, P, M seal, transport, replay, source, and name the same finite route. Since none of the excluded compactness, choice, endpoint-selection, quotient-equality, trichotomy, or ambient-completeness objects occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1792.3 (Closed interval completion finite intersection handoff)**.** *Every Real-and-Completion consumer that reads an accepted* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *carrier factors its closed-family intersection request through the finite route*

$$J, F, A \longmapsto Q \longmapsto N, C, E \longmapsto R \longmapsto H, T, P, M.$$

Consequently the consumer receives a displayed finite family index, displayed closed-family rows, finite-acceptance data, a closed-interval Cauchy completion handoff, nested-window and Cantor intersection handoffs, a finite epsilon-net or interval-net witness, a terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* $\mathsf{NameCert}$ *row. The route supplies no open-cover compactness, arbitrary closed-family compactness, countable choice, selected endpoint, quotient real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $K = (J, F, A, Q, N, C, E, R, H, T, P, M)$. The finite family index J, closed-family row F, and acceptance ledger A are the only source coordinates available to a finite intersection request.

The completion boundary is downstream of those rows. By the closed-interval completion handoff obligation of Definition 1792.2, the row Q must be consumed before the packet exposes nested-window, Cantor, finite-net, or Real-facing data.

The compactness-shaped data remain finite. The rows N, C, E route the request through displayed nested-window, Cantor-style, and finite-net or interval-net witnesses, and none of these rows contains an open-cover compactness theorem or an arbitrary closed-family principle.

The Real-facing seal and structural rows preserve that order. The row R is reachable only after J, F, A, Q, N, C, E, while H, T, P, M transport, replay, source, and name the same displayed route. Thus every consumer factors through the stated finite handoff. □

Theorem 1792.4 (Finite intersection property Real seal nesting)**.** *For any accepted* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *carrier*

$$K = (J, F, A, Q, N, C, E, R, H, T, P, M),$$

the terminal $\mathsf{Real}^{\uparrow}$ *seal row R is accessible only after the finite family rows J, F, A, the* $\mathsf{ClosedIntervalCauchyCompletio}$ *row Q, and the compactness-shaped rows N, C, E. Therefore the packet cannot export open-cover compactness, arbitrary closed-family compactness, selected endpoints, quotient real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1792.1. The finite family index J, closed-family row F, finite-acceptance ledger A, closed-interval completion row Q, compactness-shaped rows N, C, E, and terminal seal R are the only analytic rows available before transport and replay.

Use the Real seal routing clause in Definition 1792.2. It requires R to remain downstream of J, F, A, Q, N, C, E and upstream only of H, T, P, M.

Finally apply Definition 1792.3. The handoff route places the finite family rows before Q, then the nested-window, Cantor, and finite-net rows before R. Since the excluded compactness, endpoint-selection, quotient, trichotomy, and ambient-completeness objects are absent from the carrier, the Real seal read cannot export them. □

Theorem 1792.5 (Finite intersection property compact-filter route)**.** *Whenever a* $\mathsf{Compact}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer reads an accepted* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *carrier*

$$K = (J, F, A, Q, N, C, E, R, H, T, P, M),$$

any compact-filter conclusion must factor through the displayed finite-family route

$$J, F, A \longmapsto Q \longmapsto N, C, E \longmapsto R \longmapsto H, T, P, M.$$

The consumer must therefore expose the finite subfamily index, closed-family rows, closed-interval completion handoff, finite-net or interval-net witness, and terminal $\mathsf{Real}^{\uparrow}$ *seal before any compact-filter use is public. The packet exports no open-cover compactness theorem, arbitrary closed-family compactness principle, countable-choice schedule, selected endpoint, quotient real equality, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 1792.1. The only source rows for a compact-filter read are the finite family index J, the closed-family row F, and the finite-acceptance ledger A.

Apply Definition 1792.2. The NameCert surface forces closed-interval completion through Q before any nested-window, Cantor, finite-net, or Real-facing conclusion is exported.

Use Definition 1792.3 and Definition 1792.4. These route the compactness-shaped rows through N, C, E and place the terminal seal R downstream of the finite-family and completion data.

The structural rows H, T, P, M transport, replay, source, and name only that finite route. Since no excluded compactness, choice, endpoint-selection, quotient-equality, trichotomy, or ambient-completeness coordinate occurs in the carrier, a $\mathsf{Compact}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$ consumer receives only the displayed compact-filter boundary. □

Theorem 1792.6 (Finite intersection property compact subcover extraction)**.** *For an accepted* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *carrier*

$$K = (J, F, A, Q, N, C, E, R, H, T, P, M),$$

any compact finite-window consumer requesting a subcover or intersection witness must first expose the finite family index J, the closed-family row F, and the finite-acceptance ledger A; then the $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *row Q; then the compactness-shaped rows N, C, E; and only then the* $\mathsf{Real}^{\uparrow}$ *seal R and the structural rows H, T, P, M. The extracted witness is a finite row inside E, N, C, not an open-cover compactness theorem, arbitrary closed-family principle, or countable-choice schedule.*

Proof. Project the carrier of Definition 1792.1. The finite source data are J, F, A, so a consumer cannot ask for a subcover or intersection witness before the finite index, closed-family rows, and acceptance ledger are displayed.

Apply the NameCert obligations of Definition 1792.2. They force the closed-interval completion handoff through Q before any compactness-shaped row can be exported.

Use the compact-filter route of Definition 1792.5. It places N, C, E after Q and before the terminal $\mathsf{Real}^{\uparrow}$ seal R, so the witness available to a compact finite-window consumer is a finite nested-window, Cantor-handoff, or finite-net row.

Finally apply the closed interval completion handoff of Definition 1792.3. The structural rows H, T, P, M only transport, replay, source, and name the same finite route. Since open-cover compactness, arbitrary closed-family compactness, and countable choice have no carrier coordinate, the extracted witness remains inside E, N, C. □

Falsifiable prediction: Every $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ read factors through one displayed finite family index, one closed-family row, one finite-acceptance ledger, one ClosedIntervalCauchyCompletion row, NestedIntervalCompactness and CantorIntersection handoff rows, one FiniteEpsilonNet or interval-net witness, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closed-interval-cauchy-completion, nestedintervalcompactness, cantor-intersection, finite-epsilon-net, real: $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite closed-family indexing, closed-interval completion, compactness-shaped nested and Cantor handoffs, finite-net or interval-net witnesses, and the terminal Real seal as one finite intersection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntersectionProperty}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntersectionProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1792.1, Definition 1792.2, and Definition 1792.3 seed the finite closed-family intersection certificate from displayed finite family indices and closed interval rows through closed-interval Cauchy completion, nested-window and Cantor handoffs, finite-net or interval-net witnesses, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntersectionProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary closed-family compactness, countable choice, selected endpoints, quotient real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-family exactness, closed-interval completion handoff, compactness-shape locality, Real seal routing, classifier transport, replay, provenance, local naming, and finite route factorization for BEDC.Derived.FiniteIntersectionPropertyUp.

Constructive story: A $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ packet is generated from a finite family index, displayed closed-family rows, a finite-acceptance ledger, a ClosedIntervalCauchyCompletion row, NestedIntervalCompactness and CantorIntersection handoff rows, a FiniteEpsilonNet or interval-net witness, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1793

A Concrete Naming Certificate for CauchyCompletionReflectiveSubcategory$^{\uparrow}$

Following Mac Lane's reflective-subcategory formulation and Bishop–Bridges constructive completion, the CauchyCompletionReflectiveSubcategory$^{\uparrow}$ packet records completion as a finite BEDC reflective-subcategory surface. It reads the reflector packet of chapter 1585, the functorial packet of chapter 423, the universal-property packet of chapter 397, the separated-reflection boundary of chapter 1790, the stream-name windows of chapter 126, the regular rational rows of chapter 301, and the terminal real-name surface of chapter 38. The chapter names the public boundary where completion is consumed as a reflective inclusion and reflector route without importing host completed spaces.

Definition 1793.1 (CauchyCompletionReflectiveSubcategory carrier)**.** *A* CauchyCompletionReflectiveSubcategory$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, J, U, Q, E, Z, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and name-data row, including the displayed source classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before completion is read.*
2. *M is the completed-object surface, exposed through the* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFuncto *packets and not through a quotient carrier or selected host limit.*
3. *J is the reflective inclusion row naming the completed objects as the full completion-facing subcategory surface visible to downstream consumers.*
4. *U is the reflector unit row admitting source data from S into M.*
5. *Q is the counit and idempotence row, recording return along already completed reads and reuse of the same completed-object surface after a second completion pass.*
6. *E is the universal-property factorization row for finite-window continuous consumers, read through* CompletionUniversalProperty$^{\uparrow}$ *after U, J, Q have been displayed.*
7. *Z is the separated-reflection boundary row, read through* SeparatedMetricReflection$^{\uparrow}$ *before any zero-distance or completion-facing classifier is exposed.*

8. *T is the local transport row through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *carrying source windows, regular rational readback, dyadic tolerances, and the real-name seal used by S, M, U, Q, E, Z.*

9. *H records componentwise* hsame *transport for the source metric/name data, completed-object surface, reflective inclusion, unit, counit/idempotence, universal-property factorization, separated-reflection boundary, local transport, continuation, provenance, and naming rows.*

10. *C records displayed* Cont *replay from source observations through unit admission, completed-object access, reflective inclusion, counit return, idempotent replay, universal-property factorization, separated reflection, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}}$ *row.*

The carrier exposes only the finite reflective-subcategory boundary. It contains no quotient completion, no selected limit, no host equality for completed objects, no countable-choice selector, no ambient completeness theorem, and no full completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveSubcategoryUp.TasteGate.CauchyCompletionReflectiveSubcat`
`single carrier alignment`

Theorem 1793.2 (CauchyCompletionReflectiveSubcategory NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$ *carrier $B = (S, M, J, U, Q, E, Z, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, completed-object surface, reflective inclusion, unit, counit/idempotence, universal-property factorization, separated-reflection boundary, local transport, continuation, provenance, and local naming rows;*

2. *reflective inclusion exactness, requiring every completed-object read to pass through J after the completed-object surface M has been displayed;*

3. *unit and counit compatibility, requiring source admission to use U and already-completed return or repeated completion to use Q over the same M, J, T route;*

4. *universal-property factorization, requiring each finite-window consumer to reach its completion-facing read through E after the unit and inclusion rows have been projected;*

5. *separated-reflection boundary preservation, requiring every zero-distance or completion-facing classifier read to pass through Z before the local real-name route is consumed;*

6. *local transport discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *reads to occur only through the row T;*

7. *ledger non-escape, requiring P and N to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1793.1. The packet displays the source row, completed-object surface, reflective inclusion, unit, counit/idempotence, universal-property

factorization, separated-reflection boundary, local transport, continuation, provenance, and local name.

Reflective inclusion exactness reads the completed-object surface in order. A consumer may inspect M only as a completion-facing object admitted into the inclusion row J, so the public subcategory boundary is the displayed finite row rather than a host collection of completed spaces.

The unit, counit, and idempotence obligations are the two reflector directions. Source data enter through U, already-completed reads return through Q, and repeated completion reuses the same M, J, T route.

Universal-property factorization and separated-reflection preservation are downstream of that route. The row E is the only extension surface for finite-window consumers, and the row Z is the only boundary from which zero-distance or completion-facing classifier evidence can be read.

Finally, H, C, P, N transport, replay, package, and name exactly the displayed rows. Since quotient completions, selected limits, host equality, countable choice, ambient completeness, and full $\mathsf{Real}^{\uparrow}$ completeness are absent from those rows, no local NameCert obligation can export them. □

Theorem 1793.3 (CauchyCompletionReflectiveSubcategory reflection law)**.** *Let $B = (S, M, J, U, Q, E, Z, T, H, C,$ be an accepted* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$ *carrier. Every public completion consumer factors through the finite reflective route*

$$S \longmapsto U \longmapsto M \longmapsto J \longmapsto Q \longmapsto E, Z \longmapsto T \longmapsto H, C, P, N.$$

Thus the unit, counit, idempotence, universal-property factorization, separated-reflection boundary, and local StreamName–RegSeqRat–DyadicRatCore–Real transport are read as one BEDC reflective-subcategory surface. The route exposes no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a public completion consumer and project the carrier rows of Definition 1793.1. The source row S is visible only through unit admission U, and unit admission leads to the completed-object surface M.

The reflective inclusion row J is then the public subcategory boundary. The counit and idempotence row Q compares already-completed reads and repeated completion requests only over the same M, J, T coordinates.

The universal-property row E and separated-reflection row Z are consumed after the reflector route is fixed. They supply extension factorization and classifier boundary data without adding a second completed-object carrier.

The local transport row T and structural rows H, C, P, N close the route. Because these rows exhaust the packet, the listed non-escape claims follow directly from the absence of quotient, selected-limit, host-equality, choice, ambient-completeness, and full real-completeness coordinates. □

Theorem 1793.4 (CauchyCompletionReflectiveSubcategory L10 handoff)**.** *Every L10 completion read exported by an accepted* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$ *packet is obtained by combining the sibling reflector, functor, universal-property, separated-reflection, stream-name, regular-rational, dyadic-tolerance, and real-name rows named in P. The handoff is exhausted by the carrier of Definition 1793.1, the obligations of Definition 1793.2, and the reflection law of Definition 1793.3.*

Proof. The provenance row P names exactly the sibling packets available to the handoff: CauchyCompletionReflector CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, SeparatedMetricReflection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$.

Apply Definition 1793.2 to read those packets through the carrier rows $S, M, J, U, Q, E, Z, T, H, C, P, N$. Each read is assigned to the source, reflector, inclusion, extension, separated boundary, local transport, or naming part of the same finite packet.

Apply Definition 1793.3 to place the consumer on the displayed reflective route. The route is finite and public, so the L10 handoff has no remaining coordinate from which a quotient completion, selected limit, host equality, countable choice, ambient completeness, or full Real$^{\uparrow}$ completeness theorem could be obtained. □

Closure status for NameCert$_{\text{CauchyCompletionReflectiveSubcategory}^{\uparrow}}$.

Constructive story: A CauchyCompletionReflectiveSubcategory$^{\uparrow}$ packet is generated from source metric/name data, a completed-object surface, reflective inclusion, reflector unit, counit and idempotence rows, universal-property factorization, separated-reflection boundary, local StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionReflectiveSubcategory}^{\uparrow}}$, seed)

Scope closed: Definition 1793.1, Definition 1793.2, Definition 1793.3, and Definition 1793.4 seed the finite reflective-subcategory boundary for the L10 completion route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionReflectiveSubcategory}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality, countable choice, ambient completeness, full Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflective inclusion exactness, unit-counit-idempotence compatibility, universal-property factorization, separated-reflection preservation, local transport discipline, and ledger non-escape for BEDC.Derived.CauchyCompletionReflectiveSubcategoryUp.

Chapter 1794

A Concrete Naming Certificate for $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive regular-Cauchy completion, the $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ packet names the finite sequential-completeness surface that sits between regular rational names and the downstream real-facing limit consumer. It reads the dyadic endpoint ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence row of chapter 301, and the terminal real-name seal of chapter 38. It also records the completion-side compatibility supplied by chapter 1639, the real-facing consumer boundary of chapter 1544, and the extension-and-uniqueness surface of chapter 397. The packet is a finite NameCert bridge from regular Cauchy sequence data to a displayed limit seal; it is not a quotient stream construction, a selected host limit, countable choice, open-cover compactness, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 1794.1 (RegularCauchySequentialCompleteness carrier)**.** *A* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, M, L, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic endpoints and comparison budgets used to inspect the regular Cauchy sequence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the inspected stream windows needed by the sequential-completeness request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row read from the windows in S and the tolerance ledger in D.*
4. *M is the regular Cauchy modulus row, assigning each displayed dyadic tolerance a finite tail threshold for the sequence row R.*
5. *L is the displayed* $\mathsf{Real}^{\uparrow}$ *limit seal reached from R through the same D, S, M route.*
6. *U is the completion-universal compatibility row, read through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *after the Bishop regular-Cauchy completion surface has supplied the source and limit routes.*
7. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular sequence row, modulus row, limit seal, universal-property row, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from a sequential-completeness request through M, D, S, R, L, U.*
9. *P is the* Pkg *provenance over* RegularCauchySequentialCompleteness$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BishopRegularCauchyCompletion$^{\uparrow}$, RealSequentialCompleteness$^{\uparrow}$, CompletionUniversalProperty BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchySequentialCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, M, L, U, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected host limits, countable choice, open-cover compactness, host real equality, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ BEDC.Derived.RegularCauchySequentialCompletenessUp.RegularCauchySequentialCompletenessUpTasteGa
single carrier alignment

Theorem 1794.2 (RegularCauchySequentialCompleteness NameCert obligations)**.** *For every accepted* RegularCauchySequentialCompleteness$^{\uparrow}$ *carrier $Q = (D, S, R, M, L, U, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchySequentialCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance ledger, stream-window row, regular rational sequence row, regular Cauchy modulus, limit seal, completion-universal compatibility row, transport, replay, provenance, and local naming row;*
2. *classifier stability, requiring componentwise* hsame *transport in H to preserve the rows D, S, R, M, L, U before the local classifier identifies two accepted packets;*
3. *regular-sequence tail control, requiring each tolerance request to read its finite threshold from M and then inspect the corresponding S, R, D rows;*
4. *limit-seal handoff, requiring the displayed* Real$^{\uparrow}$ *row L to be reached only after the regular Cauchy route M, D, S, R has been replayed;*
5. *completion compatibility, requiring the* CompletionUniversalProperty$^{\uparrow}$ *row U to be read only after the Bishop regular-Cauchy completion source route and the displayed limit seal have both been exposed;*
6. *ledger non-escape, requiring P and N to export no quotient streams, selected host limits, countable choice, open-cover compactness, host real equality, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1794.1. The visible packet consists of the dyadic ledger D, finite windows S, regular sequence row R, modulus row M, limit seal L, universal-property row U, and the structural rows H, C, P, N.

Classifier stability is componentwise. The transport row H preserves the displayed analytic rows and the structural rows, so a classified replacement cannot change the tolerance ledger, stream windows, regular readback, threshold schedule, limit seal, or universal-property compatibility route.

Regular-sequence tail control is forced by the order of the replay row C. A tolerance request first reads its threshold from M, then inspects the finite windows S, the regular rational readback R, and the dyadic ledger D attached to that threshold.

The limit-seal handoff and completion compatibility are downstream of the same route. The Real$^{\uparrow}$ seal L is reached only after M, D, S, R have been displayed, and the row U is then read as the completion-universal compatibility surface attached to that visible source-and-limit pair.

Finally, P and N record only provenance and local naming for the listed rows. Since no excluded coordinate occurs in the carrier, the local NameCert surface is exhausted by the six obligations displayed above. □

Theorem 1794.3 (RegularCauchy sequential tail seal)**.** *Let $Q = (D, S, R, M, L, U, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier. Any consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal L must first expose a dyadic tolerance in D, the finite* $\mathsf{StreamName}^{\uparrow}$ *window in S, the regular rational readback in R, and the threshold/modulus row M. Componentwise* hsame *transport and displayed* Cont *replay preserve that route before L is consumed, so the carrier exports no quotient stream equality, selected host limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. The rows available before the terminal seal are the dyadic ledger D, stream-window row S, regular rational readback R, modulus row M, and the structural rows that transport and replay them.

Use the replay order in Definition 1794.2. A tolerance request reads its dyadic budget in D, exposes the inspected finite window in S, obtains the regular rational readback in R, and then reads the threshold/modulus row M before the terminal row L is available.

The transport and replay rows do not add a second route. Componentwise hsame in H preserves the displayed rows, while Cont replay in C repeats the same tolerance-window-readback-threshold path.

Finally apply the ledger non-escape rows P, N. Since the excluded quotient, host-limit, choice, host-equality, and ambient-completeness objects are not coordinates of the carrier, no consumer of L can export them. □

Theorem 1794.4 (RegularCauchySequentialCompleteness limit seal)**.** *Let $Q = (D, S, R, M, L, U, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier. Every real-facing sequential-completeness consumer of Q reaches the displayed limit only through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BishopRegularCauchyCompletion}^{\uparrow},\ \mathsf{CompletionUniversalProperty}$$

The limit seal is the row L read after M, D, S, R and before U is consumed; it is not a selected host limit, quotient representative, countable-choice output, open-cover compactness consequence, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier Q. The only rows available to a real-facing consumer before the limit read are M, D, S, R together with the transport and replay rows that preserve their displayed order.

The consumer first reads the regular Cauchy request through M. The threshold supplied by M selects the finite stream windows in S, the regular rational readback in R, and the dyadic tolerance comparisons in D.

Only after that route is displayed can the consumer read L as the $\mathsf{Real}^{\uparrow}$ limit seal. The completion-universal row U is then available as compatibility data for the same visible route, not as a second source of limits.

The excluded objects are absent from the tuple. Hence the limit seal cannot be obtained by quotient streams, host representatives, choice, open covers, or an ambient completeness principle. □

Theorem 1794.5 (RegularCauchySequentialCompleteness modulus-limit determinacy)**.** *Let $Q = (D, S, R, M, L, U, H, C, P, N)$ and $Q' = (D', S', R', M', L', U', H', C', P', N')$ be accepted* RegularCauchySequentialCo

carriers. Suppose the transport rows display componentwise hsame *comparison between D and D', between S and S', between R and R', and between M and M'. Suppose also that C and C' give compatible displayed* Cont *replay for the same dyadic tolerance request through those transported rows. Then the two real-facing sequential-completeness reads reach the same displayed limit-seal route before either U or U' is consumed.*

Equivalently, the displayed seal comparison is determined by the transported modulus-source route M, D, S, R. No quotient stream, selected host limit, countable-choice row, host representative, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate can separate the seal route of Q from the seal route of Q'.*

Proof. Project both accepted carriers. By Definition 1794.2, classifier stability sees the dyadic ledger, stream-window row, regular rational readback, and modulus row only through componentwise hsame transport. Thus the displayed source route for the fixed tolerance request is the transported route M, D, S, R on the first carrier and the matching route M', D', S', R' on the second carrier.

The compatible Cont replay rows preserve the order of that route. The request first reads the modulus threshold, then the finite stream windows, then the regular rational sequence row and the dyadic comparison ledger. The two replays therefore expose the same finite source information before any completion-universal compatibility row is available.

Apply Definition 1794.4 to each carrier. In both reads, the $\mathsf{Real}^{\uparrow}$ seal is reached exactly after the displayed M, D, S, R route and strictly before the corresponding U row is consumed. Since the source route has already been identified by transport and compatible replay, the two reads display the same limit-seal route.

The remaining excluded data are not coordinates of either carrier before the seal. Quotient streams, selected host limits, countable choice, host representatives, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness therefore have no local row by which they could distinguish the two displayed seal routes. □

Theorem 1794.6 (RegularCauchySequentialCompleteness nonescape)**.** *Every completion-facing or real-facing read of an accepted* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier $Q = (D, S, R, M, L, U, H, C, P, N)$ is exhausted by the rows $D, S, R, M, L, U, H, C, P, N$. Thus the local packet exports only dyadic tolerances, finite stream windows, regular rational sequence data, regular Cauchy modulus thresholds, a displayed* $\mathsf{Real}^{\uparrow}$ *limit seal, completion-universal compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming data.*

It exports no quotient streams, selected host limits, countable choice, open-cover compactness, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. The carrier tuple lists every row admitted by the local certificate. A completion-facing or real-facing consumer can therefore obtain only $D, S, R, M, L, U, H, C, P, N$ by projection.

The structural rows do not enlarge the analytic surface. The hsame row transports already displayed coordinates, the Cont row replays the same ordered route, and the provenance and naming rows record the finite source of the packet.

Since the excluded objects are not coordinates of the carrier and are not produced by the replay route, no accepted read can escape the displayed regular-Cauchy sequential-completeness boundary. □

Theorem 1794.7 (RegularCauchySequentialCompleteness scoped kernel route)**.** *Every accepted* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier factors through the displayed L10 finite rows*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longrightarrow \mathsf{RealSequer}$$

The factorization reads the dyadic tolerance rows, finite stream windows, regular rational readback, terminal limit seal, Bishop regular-Cauchy completion route, real-sequential-completeness consumer boundary, and completion-universal handoff in that order. It exports no quotient stream, selected limit, countable-choice witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier definition. The available analytic rows are the dyadic ledger D, finite windows S, regular rational sequence row R, modulus row M, displayed limit seal L, and completion-universal row U.

Use the limit-seal theorem to locate the $\mathsf{Real}^{\uparrow}$ read. It places L after the displayed M, D, S, R route and before the completion-universal row is consumed.

Use modulus-limit determinacy and nonescape. These results keep every real-facing read inside the transported finite rows and prevent quotient streams, selected limits, countable choice, and ambient completeness from entering as hidden coordinates.

Finally read the sibling dependencies in the order named in the chapter introduction: DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, RealUp seal, Bishop regular-Cauchy completion, RealSequentialCompleteness boundary, and CompletionUniversalProperty handoff. □

Theorem 1794.8 (RegularCauchySequentialCompleteness obligation surface)**.** *For every accepted* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *carrier*

$$Q = (D, S, R, M, L, U, H, C, P, N),$$

the rows $D, S, R, M, L, U, H, C, P, N$ *jointly supply the obligation surface for carrier admission, classifier stability, regular tail control, limit-seal handoff, completion compatibility, and ledger non-escape. This surface is exactly the displayed regular-Cauchy sequential-completeness route: it reads dyadic tolerances, finite windows, regular rational sequence data, modulus thresholds, a displayed* $\mathsf{Real}^{\uparrow}$ *limit seal, completion-universal compatibility, transport, replay, provenance, and local naming, and no quotient stream, selected host limit, countable-choice witness, open-cover compactness row, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission and classifier stability are the first two obligations of Definition 1794.2. They identify $D, S, R, M, L, U, H, C, P, N$ as the whole accepted packet and make classifier transport componentwise through those rows.

Regular tail control and the limit-seal handoff are supplied by the same obligation theorem together with Definition 1794.4. These results place each tolerance request through M, D, S, R before L is exposed.

Completion compatibility is the downstream part of Definition 1794.7. It routes the displayed limit through the Bishop regular-Cauchy completion, real-sequential-completeness, and completion-universal surfaces without adding another analytic row.

Ledger non-escape is given by Definition 1794.5 and Definition 1794.6. They keep all transported reads inside the finite tuple and exclude quotient streams, selected host limits, choice, open covers, host equality, and ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, a RegSeqRat regular rational sequence row, a regular Cauchy modulus row, a displayed RealUp limit seal, a CompletionUniversalProperty compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1794.1, Definition 1794.2, Definition 1794.4, Definition 1794.5, and Definition 1794.6 seed the finite regular-Cauchy sequential-completeness surface between $\mathsf{RegSeqRat}^{\uparrow}$ data and the displayed $\mathsf{Real}^{\uparrow}$ limit seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySequentialCompleteness}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchySequentialCompletenessUp.RegularCauchySequentialCompletenessUpTasteGate\_s`

Bridge status: none

Not claimed: This chapter excludes quotient streams, selected host limits, countable choice, open-cover compactness, Real completeness as an ambient principle, Lean verification, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, regular-sequence tail control, limit-seal handoff, completion compatibility, and ledger non-escape for BEDC.Derived.RegularCauchySequentialCompletenessUp.

Chapter 1795

A Concrete Naming Certificate for $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$

Following Bishop and Bridges constructive completion and the Bourbaki-Kelley universal-mapping language, the packet records the two triangle composites for the Cauchy-completion adjunction as finite BEDC routes. It reads the monad packet of chapter 378, the unit packet of chapter 1527, the counit packet of chapter 1539, the adjunction packet of chapter 1456, the idempotence packet of **??**, the stream-name surface of chapter 126, the regular rational route of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter gives later completion associativity, functoriality, and Real-facing handoff packets a finite triangle-identity surface rather than a host categorical equality.

Definition 1795.1 (CauchyCompletionTriangleIdentity carrier)**.** *A* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, U, V, A, I, L, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row, including the finite unit-and-bind ledger used by completion consumers before any triangle composite is inspected.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row, admitting source* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data into the completion route.*
3. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row, returning already-completed reads without selecting representatives or host limits.*
4. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row, naming the unit-counit correspondence and its finite universal-mapping ledger.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *row, recording that a second completion pass reuses the same completion surface rather than adding a new quotient carrier.*
6. *L is the left triangle route ledger, from source admission through completion, counit return, adjunction comparison, and idempotent reuse.*
7. *R is the right triangle route ledger, from completed input through unit replay, monad bind, counit return, adjunction comparison, and idempotent reuse.*
8. *S is the* $\mathsf{StreamName}^{\uparrow}$ *handoff row, exposing only finite stream windows used by M, U, V, A, I, L, R.*

9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff row, recording tolerance budgets and dyadic comparison steps available to the two route ledgers.*
10. *E is the joint* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, sealing regular rational reads into the real-name boundary after the two triangle ledgers have been displayed.*
11. *H records componentwise* hsame *transport for the monad, unit, counit, adjunction, idempotence, left route, right route, stream, dyadic, real-regular handoff, continuation, provenance, and naming rows.*
12. *C records displayed* Cont *replay from source windows and regular rational reads through the unit, counit, adjunction, idempotence, both triangle ledgers, dyadic tolerance, real-name seal, and local naming rows.*
13. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}}$ *row.*

The carrier exposes only the finite triangle-coherence boundary. It contains no quotient completion, no selected representative, no host function equality, no countable-choice selector, no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, and no bridge-checked route.*

Lean4 (definition) `BEDC.Derived.CauchyCompletionTriangleIdentityUp.CauchyCompletionTriangleIdentityUp`

Theorem 1795.2 (CauchyCompletionTriangleIdentity NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *carrier* $T = (M, U, V, A, I, L, R, S, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed monad, unit, counit, adjunction, idempotence, left route, right route, stream, dyadic, real-regular, transport, continuation, provenance, and naming rows;*
2. *left triangle exactness, requiring every source-side triangle read to pass through U, M, V, A, I, L before it can be exported;*
3. *right triangle exactness, requiring every completed-side triangle read to pass through M, U, V, A, I, R before it can be exported;*
4. *handoff discipline, requiring all finite observations to use S, all dyadic budgets to use D, and all real or regular rational reads to use E;*
5. *route exhaustion, requiring H, C, P, N to replay and name no row except the displayed carrier rows;*
6. *non-escape, requiring the packet to export no quotient completion, selected representative, host function equality, countable choice, ambient real completeness, or bridge-checked coherence theorem.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1795.1. The packet displays the monad, unit, counit, adjunction, idempotence, two route ledgers, three handoff rows, transport, continuation, provenance, and local name.

The left and right triangle obligations are read separately. The left route ledger L starts from source admission through U, then reads the completion surface through M, returns through V, compares through A, and reuses completion through I.

The right route ledger R starts from completed input, replays unit and bind through M and U, returns through V, compares through A, and records repeated completion through I. Thus each triangle composite has a named finite route before any consumer inspects it.

The handoff rows are then forced by projection. Finite stream observations are in S, dyadic tolerances are in D, and real or regular rational reads are in E.

Finally, H, C, P, N transport, replay, package, and name only the displayed rows. Since quotient completions, selected representatives, host function equality, countable choice, ambient real completeness, and bridge checking are absent from those rows, the non-escape obligation follows by route exhaustion. □

Theorem 1795.3 (CauchyCompletionTriangleIdentity left route)**.** *Let* $T = (M, U, V, A, I, L, R, S, D, E, H, C, P, N$ *be an accepted* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *carrier. Every source-side completion read exported by the packet factors through the finite left triangle route*

$$S, E \longmapsto U \longmapsto M \longmapsto V \longmapsto A \longmapsto I \longmapsto L \longmapsto H, C, P, N.$$

The route identifies the source admission followed by completed return with the original finite completion-facing read only through the displayed rows.

Proof. Begin with a source-side read. The only source windows available to the packet are the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ coordinates displayed in S and E.

Project the unit row U to admit those windows into the completion route. The monad row M supplies the completion surface, while the counit row V supplies the return from the completed side.

The adjunction row A is the comparison ledger for this composite, and the idempotence row I records that no second completion carrier is created. The left route row L stores exactly this source-side order.

The structural rows H, C, P, N then transport, replay, package, and name the same route. Because no other row is displayed, the source-side triangle read cannot appeal to representative selection, host equality, choice, ambient completeness, or bridge checking. □

Theorem 1795.4 (CauchyCompletionTriangleIdentity right route)**.** *Let* $T = (M, U, V, A, I, L, R, S, D, E, H, C, P,$ *be an accepted* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *carrier. Every completed-side read exported by the packet factors through the finite right triangle route*

$$E, D \longmapsto M \longmapsto U \longmapsto V \longmapsto A \longmapsto I \longmapsto R \longmapsto H, C, P, N.$$

The route identifies completed input followed by unit replay and counit return with the original completed read only through the displayed rows.

Proof. Begin with a completed-side read. The available completed observation is the real and regular-rational handoff row E, together with the dyadic tolerance row D needed by the completion ledger.

Project M to expose the completion surface, then replay unit information through U and return through V. These projections are interpreted through the adjunction row A.

The idempotence row I records that the repeated completion request reuses the same surface. The right route row R stores exactly this completed-side order.

The rows H, C, P, N close the route by transport, continuation, provenance, and local naming. Since the route is exhausted by displayed rows, the packet exports no quotient completion, selected representative, host function equality, countable choice, ambient real completeness, or bridge-checked theorem. □

Theorem 1795.5 (CauchyCompletionTriangleIdentity Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *packet is obtained after the left or right triangle route has passed through the shared* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *handoff rows. The handoff is exhausted by the carrier of Definition 1795.1, the obligations of Definition 1795.2, the left route of Definition 1795.3, and the right route of Definition 1795.4.*

Proof. The provenance row P names the available sibling packets: $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 1795.3 for source-side consumers and Definition 1795.4 for completed-side consumers. In both cases the route reaches the handoff rows before the local name is consumed.

Now project S, D, E. The finite stream window is read from S, the dyadic budget is read from D, and the regular rational plus real-name seal is read from E.

By Definition 1795.2, no additional handoff row exists. Hence every Real-facing read is routed through the displayed finite triangle-coherence packet and cannot obtain quotient completion data, selected representatives, host function equality, countable choice, ambient real completeness, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ packet is generated from Cauchy-completion monad, unit, counit, adjunction, idempotence, left-route, right-route, StreamName, DyadicRatCore, Real-RegSeqRat handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1795.1, Definition 1795.2, Definition 1795.3, Definition 1795.4, and Definition 1795.5 seed the finite triangle-identity coherence boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier and route proofs for BEDC.Derived.CauchyCompletionTriangleIdentityUp, including both triangle ledgers and the StreamName-RegSeqRat-DyadicRatCore-Real handoff.

Chapter 1796

A Concrete Naming Certificate for $\mathsf{MetricCompletionUniqueness}^{\uparrow}$

Following Bishop–Bridges separated completion uniqueness and the Bourbaki presentation of completions by universal dense maps, the $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ packet records the finite BEDC boundary at which two completion presentations over the same metric source may be compared. It consumes the dense embedding route of chapter 1512, the finite-window completion handoff of chapter 1559, the separated classifier route of chapter 1790, and the general separated-completion boundary of chapter 1612. The packet names uniqueness only as a routed comparison between presentations; it does not introduce quotient completion objects, selected representatives, or host equality of completed points.

Definition 1796.1 (MetricCompletionUniqueness carrier)**.** *A* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, E_0, F_0, R_0, S_0, E_1, F_1, R_1, S_1, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the common* $\mathsf{Met}^{\uparrow}$ *source row whose point classifier and distance witnesses are shared by both completion-facing presentations.*
2. *E_i is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *dense row for presentation $i \in \{0, 1\}$, recording how source observations enter that presentation.*
3. *F_i is the* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *handoff row for presentation i, exposing only finite-window completion reads downstream of M and E_i.*
4. *R_i is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier route for presentation i, comparing completion-facing reads by separated zero-distance evidence.*
5. *S_i is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *uniqueness boundary for presentation i, read over the same M, E_i, F_i, R_i route.*
6. *Q is the local comparison ledger. It records that both presentations read the same source row M, that dense rows are consumed before finite handoff rows, that classifier routes are consumed before uniqueness rows, and that the comparison never leaves the displayed separated routes.*
7. *H stores componentwise* hsame *transport for M, the two dense rows, the two finite handoff rows, the two separated-reflection rows, the two separated-completion rows, the comparison ledger, and the structural rows.*

8. *C stores displayed* Cont *replay from source metric observations through each dense row, finite handoff, separated classifier, separated-completion boundary, and then through Q.*

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{MetricCompletionFinite}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$ $\mathsf{SeparatedCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionUniqueness}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected representatives, host equality of completed points, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUniquenessUp.MetricCompletionUniquenessTasteGate single carrier alignment`

Theorem 1796.2 (MetricCompletionUniqueness NameCert obligations)**.** *For every accepted* MetricCompletionUnic *carrier*

$$U = (M, E_0, F_0, R_0, S_0, E_1, F_1, R_1, S_1, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionUniqueness}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the common source metric row, the two dense embedding rows, the two finite completion handoff rows, the two separated-reflection classifier routes, the two separated-completion uniqueness boundaries, the comparison ledger, transport, replay, provenance, and local naming rows;*

2. *common-source agreement, requiring both presentations to read the same M before any completion-facing comparison is admitted;*

3. *dense-to-finite order, requiring each E_i to precede the corresponding finite handoff F_i;*

4. *classifier-before-uniqueness order, requiring each separated-reflection route R_i to precede the corresponding separated-completion boundary S_i;*

5. *comparison exactness, requiring Q to compare only the two displayed routes M, E_i, F_i, R_i, S_i and their structural rows;*

6. *ledger non-escape, excluding quotient completion, selected representatives, host equality of completed points, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Lean4 ✓ `BEDC.Derived.MetricCompletionUniquenessUp.TasteGate.MetricCompletionUniquenessTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1796.1. The tuple lists the common source, the two completion-facing routes, the comparison ledger, and the structural rows.

Common-source agreement is the first dependency. Both presentations expose the same M, so a comparison cannot be formed from unrelated metric sources or from an external completed host space.

Dense-to-finite order is read separately on the two sides. For each i, the dense row E_i is displayed before the finite handoff F_i, so no selected representative or quotient point can replace the source-to-completion route.

Classifier-before-uniqueness order is also sidewise. The separated-reflection route R_i supplies the zero-distance classifier boundary before the separated-completion row S_i is read as a uniqueness boundary.

Finally Q, H, C, P, N compare, transport, replay, source, and name only those displayed rows. Since the carrier has no quotient, representative, host-equality, choice, compactness, ambient-completeness, Lean, or bridge coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1796.3 (MetricCompletionUniqueness separated route)**.** *Every comparison exported by an accepted* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *carrier factors through the finite separated route*

$$M \longmapsto (E_0, F_0, R_0, S_0) \quad \textit{and} \quad M \longmapsto (E_1, F_1, R_1, S_1) \longmapsto Q, H, C, P, N.$$

Thus separated uniqueness is read only after both completion presentations have consumed the same source metric row, dense embedding route, finite completion handoff, separated classifier route, and separated-completion boundary.

Proof. Begin with a consumer comparison admitted by the packet. Carrier admission exposes the two routes and the common source row M.

On each side, the dense row precedes the finite handoff. The consumer therefore reaches F_i only through E_i and never through a selected representative, quotient class, or host completion point.

The separated-reflection row R_i then supplies the classifier route used by the corresponding separated-completion row S_i. This places zero-distance classifier evidence before any uniqueness read.

The comparison ledger Q is downstream of both sidewise routes. The transport and continuation rows H, C preserve the same route, while P, N record provenance and local naming. Hence the exported comparison factors through the displayed separated route and nothing else. □

Theorem 1796.4 (MetricCompletionUniqueness separated classifier exactness)**.** *Let*

$$U = (M, E_0, F_0, R_0, S_0, E_1, F_1, R_1, S_1, Q, H, C, P, N)$$

be an accepted $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *packet. Every uniqueness-facing comparison is classifier-exact only when both presentations read, in order, the common source row M, their dense embedding rows E_0, E_1, their finite handoff rows F_0, F_1, their separated-reflection classifier routes R_0, R_1, their separated-completion boundaries S_0, S_1, and the comparison ledger Q, H, C, P, N. Thus the two completion presentations are compared through the same separated zero-distance classifier route, not through quotient completion equality or host equality of completed points.*

Proof. Apply the NameCert obligations of Definition 1796.2. They require common-source agreement at M, dense-to-finite order along each E_i, F_i side, classifier-before-uniqueness order along each R_i, S_i side, and comparison exactness through Q, H, C, P, N.

Use the separated route theorem, Definition 1796.3. It orders every exported comparison through the two displayed sidewise routes over the same source before the comparison ledger and structural rows can be consumed.

Finally invoke the non-escape theorem, Definition 1796.5. Since quotient completion, selected representatives, host equality of completed points, choice, compactness, ambient completeness, Lean evidence, and bridge evidence are not carrier coordinates, classifier exactness is exhausted by the shared separated zero-distance routes R_0, R_1 and the displayed comparison ledger. □

Theorem 1796.5 (MetricCompletionUniqueness non-escape)**.** *An accepted* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *packet exports no quotient completion, no selected representative, no host equality of completed*

points, no countable-choice schedule, no open-cover compactness row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, no Lean verification, and no bridge checking. Every uniqueness-facing read is confined to the displayed rows*

$$M, E_0, F_0, R_0, S_0, E_1, F_1, R_1, S_1, Q, H, C, P, N.$$

Proof. Use the NameCert obligations of Definition 1796.2. They enumerate the common source row, the two dense-to-finite completion routes, the two separated classifier and uniqueness routes, the comparison ledger, transport, replay, provenance, and local naming as the whole public surface.

Use the separated route theorem, Definition 1796.3, for any uniqueness-facing comparison. It forces the comparison through M, E_i, F_i, R_i, S_i on both sides before Q, H, C, P, N can be consumed.

No remaining coordinate can supply quotient completion data, representatives, host equality, choice, compactness, ambient completeness, Lean evidence, or bridge evidence. Therefore those excluded objects cannot be projected from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionUniqueness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1796.1, Definition 1796.2, Definition 1796.3, and Definition 1796.5 seed the separated uniqueness boundary for two metric-completion presentations over the same source metric row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, host equality of completed points, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source agreement, dense-to-finite order, classifier-before-uniqueness order, comparison exactness, and ledger non-escape for BEDC.Derived.MetricCompletionUniquenessUp.

Constructive story: A $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ packet is generated from a common Metric source row, two CompletionEmbedding dense rows, two MetricCompletionFinite handoff rows, two SeparatedMetricReflection classifier routes, two SeparatedCompletion uniqueness boundaries, a local comparison ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1797

A Concrete Naming Certificate for $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$ packet records the uniqueness of the displayed dyadic embedding route from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ into the $\mathsf{Real}^{\uparrow}$ seal. It sits between the finite dyadic observation ledger of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is a finite route certificate for dyadic constants entering Real-facing completion consumers; it is not a quotient stream, host real equality, a selected limit, trichotomy, ambient Real completeness, or Lean verification.

Definition 1797.1 (Dyadic embedding uniqueness carrier)**.** *A* $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row naming the dyadic constant, its denominator scale, its tolerance cells, and the finite comparison data used by the route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row obtained from D, recording the inspected prefixes through which the dyadic constant is presented as a constant stream.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S, recording the regular rational sequence data read from the same dyadic observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from Q by the displayed finite embedding route.*
5. *E is the uniqueness ledger stating that any second accepted route with the same D, S, Q, R rows is componentwise the same BEDC route after* hsame *transport and* Cont *replay.*
6. *H records componentwise* hsame *transport for D, S, Q, R, E and for the structural rows of the packet.*
7. *C records displayed* Cont *replay from a dyadic observation request through stream windows and regular readback into the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite route certificate for the dyadic embedding into the Real seal. It contains no coordinate for quotient stream equality, host real equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or Lean verification.*

Lean4 ✓ `BEDC.Derived.DyadicEmbeddingUniquenessUp.DyadicEmbeddingUniquenessTasteGate single carrier alignment`

Theorem 1797.2 (Dyadic embedding uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$ *carrier* $U = (D, S, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic observation row, stream-window row, regular rational readback row, Real seal row, uniqueness ledger, transport row, replay row, provenance row, and local naming row;*
2. *dyadic source exactness, requiring every window in* S *to be read from the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
3. *constant-stream readback, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *to be obtained through* S *without changing the dyadic source or inserting a quotient stream representative;*
4. *Real seal route admission, requiring the* $\mathsf{Real}^{\uparrow}$ *row* R *to be exposed only after* D, S, Q *have been displayed in that order;*
5. *uniqueness stability, requiring* E, H, C *to identify any second accepted route by componentwise* hsame *transport and displayed* Cont *replay over the same rows;*
6. *ledger nonescape, requiring* P *and* N *to export no quotient stream equality, host real equality, selected limit, trichotomy, ambient Real completeness, or Lean verification witness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1797.1. The tuple displays exactly the dyadic observation row, stream windows, regular rational readback, Real seal, uniqueness ledger, componentwise transport, continuation replay, provenance, and local name.

Dyadic source exactness is the ordering of D before S. A consumer can inspect a constant-stream window only after the finite dyadic observation row has named the denominator scale, tolerance cells, and comparison data used by that window.

Constant-stream readback is the row Q read through S. Since Q is attached to the same displayed windows and the carrier has no quotient-stream coordinate, the readback cannot replace the dyadic source by an unshown representative.

The Real seal route admission is the displayed path D, S, Q, R. The Cont row C replays requests through that order, and the hsame row H transports only the same listed coordinates.

Uniqueness stability is the role of E. If another accepted packet presents the same dyadic observation, stream-window, readback, and Real-seal rows, E permits only componentwise hsame transport and the replay already stored in C, so no second route can add a distinct witness.

Finally, P and N package only the named finite dependencies. Because the carrier has no coordinate for quotient streams, host real equality, selected limits, trichotomy, ambient completeness, or Lean verification, the listed rows exhaust the local NameCert surface. □

Theorem 1797.3 (Dyadic embedding uniqueness Real seal route)**.** *Every accepted* $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$ *carrier* $U = (D, S, Q, R, E, H, C, P, N)$ *exposes a unique finite BEDC route*

$$D \longmapsto S \longmapsto Q \longmapsto R \longmapsto E, H, C, P, N$$

from the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *observations into the* $\mathsf{Real}^{\uparrow}$ *seal. Any Real-facing consumer receives the dyadic constant only through this route and receives no quotient stream equality, host real equality, selected limit, trichotomy, ambient Real completeness, or Lean verification witness.*

Proof. Start with an accepted packet $U = (D, S, Q, R, E, H, C, P, N)$. Projection gives the finite dyadic observation row D as the only source coordinate for the dyadic constant.

The stream-window row S is downstream of D, and the regular rational readback Q is downstream of those windows. Thus every Real-facing read first passes through the displayed dyadic observations and their finite constant-stream presentation.

The Real seal row R is exposed only after Q has been read. The continuation row C records this replay order, while the transport row H preserves the same coordinates componentwise.

The uniqueness ledger E then blocks competing finite routes over the same displayed rows. A second accepted route with the same D, S, Q, R data is identified by the hsame and Cont rows already present in the packet, rather than by a host equality principle or an external selected limit.

The provenance and local naming rows P, N do not enlarge the surface. They name the finite route and exclude quotient streams, host real equality, selected limits, trichotomy, ambient completeness, and Lean verification as source witnesses. Hence the Real seal receives exactly the displayed unique finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}}$, seed)

Scope closed: Definition 1797.1, Definition 1797.2, and Definition 1797.3 seed the finite uniqueness route from DyadicRatCore observations through StreamName and RegSeqRat into the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, trichotomy, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic source exactness, constant-stream readback, Real seal route admission, uniqueness stability, and ledger nonescape proofs for BEDC.Derived.DyadicEmbeddingUniquenessUp.

Constructive story: A $\mathsf{DyadicEmbeddingUniqueness}^{\uparrow}$ packet is generated from a DyadicRatCore observation row, StreamName windows, RegSeqRat readback, a RealUp seal, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1798

A Concrete Naming Certificate for $\mathsf{BisectionRootCauchySeal}^{\uparrow}$

Following Bishop–Bridges constructive bisection, the $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ packet records the finite route by which sign-change bisection data become a regular Cauchy read and then a Real-facing seal. It sits after the located-sign surface of chapter 1402 and the bracket surface of chapter 1786, and it hands its retained windows to the finite stream, regular rational, dyadic tolerance, and real-name interfaces of chapter 126, chapter 301, chapter 291, and chapter 38. The certificate names the bisection-to-Cauchy seal route; it does not select a root, quotient real names, or import ambient completeness.

Definition 1798.1 (BisectionRootCauchySeal carrier)**.** *A* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, S, M, V, B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row. It records the initial lower and upper endpoint windows, their rational or dyadic cells, the endpoint order evidence, and the interval identity inherited by each retained branch.*
2. *S is the sign row. It stores the displayed opposite endpoint signs and the branch orientation used by the bisection route.*
3. *M is the modulus row. It records the requested regular Cauchy tolerance, the bisection depth sufficient for that tolerance, and the endpoint precision consumed by sign and midpoint reads.*
4. *V is the* $\mathsf{IntermediateValue}^{\uparrow}$ *ledger row. It records the finite located-sign obligation surface that admits the sign-change interval and its displayed continuity data.*
5. *B is the branch ledger. It stores the finite bisection choices, midpoint sign rows, retained subintervals, and regularity checks for the requested depth.*
6. *W is the retained-window row. It records the lower and upper windows retained after the branch prefix and the diameter comparison demanded by the regular Cauchy read.*
7. *R is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It exposes the retained bisection windows as finite stream observations, one request depth at a time.*

8. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row. It stores the regular rational readback, dyadic tolerance rows, midpoint-radius bounds, and rounding discipline used by completion-facing consumers.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after the retained windows and regular readback rows have been read.*

10. *H records componentwise* hsame *transport for the interval, signs, modulus, intermediate-value ledger, branch ledger, retained windows, stream windows, regular and dyadic readback, Real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a root-location or completion request through endpoint inspection, sign inspection, modulus use, intermediate-value admission, finite bisection, retained-window exposure, regular readback, and Real-seal consumption.*

12. *P is the* Pkg *provenance over* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionRootCauchySeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, M, V, B, W, R, D, E, H, C, P, N$. It has no coordinate for selected roots, quotient reals, real trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host root equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BisectionRootCauchySealUp.TasteGate.BisectionRootCauchySealTasteGate`
`single carrier alignment`

Theorem 1798.2 (BisectionRootCauchySeal NameCert obligations)**.** *For every accepted* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *carrier*

$$X = (I, S, M, V, B, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionRootCauchySeal}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the interval, signs, modulus, intermediate-value ledger, branch ledger, retained windows, stream windows, regular and dyadic readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *endpoint, sign, and modulus discipline, requiring I, S, M to display the initial sign-change interval, the precision budget, and the bisection depth before any retained window is exposed;*

3. *finite bisection branch regularity, requiring V and B to keep every midpoint sign row, branch choice, retained interval, and diameter comparison attached to the same displayed sign-change packet;*

4. *StreamName, RegSeqRat, and Dyadic handoff, requiring W, R, D to expose retained windows first as finite stream observations and then as regular rational readback with dyadic tolerance rows;*

5. *Real-seal nonescape, requiring E, H, C, P, N to export only the displayed bisection-to-Cauchy route and no selected root, quotient real equality, real trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host root equality, Lean verification, or bridge checking.*

No BisectionRootCauchySeal obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1798.1. The accepted packet displays exactly the interval, signs, modulus, intermediate-value ledger, branch ledger, retained windows, regular readback, Real seal, and structural rows.

The endpoint, sign, and modulus discipline is the entry stage. The rows I, S, and M must be read before a branch prefix can be inspected, so the packet cannot replace the finite sign-change data by a host interval or selected root.

Finite bisection branch regularity is the middle stage. The row V admits the located-sign surface, while B records only midpoint signs and retained intervals produced from that surface at the depth requested by M.

The StreamName, RegSeqRat, and Dyadic handoff is the readback stage. The retained windows W are exposed through R as finite stream observations and through D as regular rational and dyadic tolerance rows before the Real seal is available.

Finally, E, H, C, P, N seal, transport, replay, package, and name the same route. Since the omitted principles have no coordinates in the carrier, none of the five obligation rows can export them. □

Theorem 1798.3 (BisectionRootCauchySeal regular handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing root-location read of an accepted* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *carrier factors through the displayed finite bisection route*

$$I, S, M, V \longmapsto B \longmapsto W \longmapsto R, D \longmapsto E, H, C, P, N$$

before consuming the $\mathsf{Real}^{\uparrow}$ *seal. The read first opens the sign-change interval and modulus rows, then the finite branch ledger, then the retained windows, then the StreamName, RegSeqRat, and Dyadic readback rows, and only then the Real-facing seal. The handoff exports a regular Cauchy seal surface rather than a selected root or host equality theorem.*

Proof. Start with an accepted packet $X = (I, S, M, V, B, W, R, D, E, H, C, P, N)$. The only entry coordinates for a root-location read are the interval, sign, and modulus rows I, S, M, together with the intermediate-value ledger V.

The bisection stage is then forced through B. This row stores the finite branch prefix, midpoint signs, retained intervals, and diameter comparisons, so no downstream consumer can read a regular Cauchy window before the branch data have been fixed.

The regular readback stage is W, R, D. The retained windows in W become finite stream observations through R, and the regular rational and dyadic tolerance rows in D supply the completion-facing read.

The Real-facing stage is terminal. The seal E is consumed only after the regular readback route is fixed, while H, C, P, N transport, replay, package, and name the same finite route. Thus every listed read factors through the displayed bisection route before it reaches the $\mathsf{Real}^{\uparrow}$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionRootCauchySeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ packet is generated from an interval row, endpoint sign rows, a modulus row, an $\mathsf{IntermediateValue}^{\uparrow}$ ledger, a finite bisection branch ledger, retained windows, $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionRootCauchySeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1798.1, Definition 1798.2, and Definition 1798.3 seed the finite route from constructive bisection data through regular Cauchy readback and Real-facing seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionRootCauchySeal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BisectionRootCauchySealUp.TasteGate.BisectionRootCauchySealTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close selected roots, quotient reals, real trichotomy, open-cover compactness, ambient $\mathrm{Real}^{\uparrow}$ completeness, host root equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint/sign/modulus discipline, finite bisection branch regularity, StreamName/RegSeqRat/Dyadic handoff, Real-seal nonescape, and regular-handoff factorization for BEDC.Derived.BisectionRootCauchySealUp.

Chapter 1799

A Concrete Naming Certificate for $\mathsf{SummableTailModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SummableTailModulus}^{\uparrow}$ packet records the finite tail-summability modulus by which a summable error schedule is turned into dyadic tolerance control before any completion-facing or Real-facing consumer reads a boundary. It sits over the majorant-series route of chapter 1774, the squeeze route of chapter 1772, the dyadic tolerance scale of chapter 1768, the finite StreamName interface of chapter 126, the RegSeqRat readback route of chapter 301, the Bishop regular Cauchy completion boundary of chapter 1639, and the terminal Real seal of chapter 38. The packet gives Real and Completion tail-control arguments one BEDC-finite modulus surface rather than letting them infer a hidden summability witness from quotient streams, selected limits, or ambient completeness.

Definition 1799.1 (Summable tail modulus carrier)**.** *A* $\mathsf{SummableTailModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, D, S, R, B, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the summable tail modulus row. It records the displayed summability schedule, the finite partial-tail ledger, and the tail index assigned to each requested tolerance.*
2. *D is the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *tolerance row consumed by M, including the dyadic precision budget and comparison slack for the displayed tail request.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened at the tail index named by M and budgeted by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite window after the summability schedule and tolerance scale have been displayed.*
5. *B is the Bishop regular Cauchy completion boundary row, recording that the regular readback reaches a completion-facing boundary only by the displayed M, D, S, R route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after M, D, S, R, B have been read.*
7. *H records componentwise* hsame *transport for the summable tail modulus, dyadic tolerance, StreamName window, RegSeqRat readback, completion boundary, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail-control request through $M, D, S, R, B, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{RegularCauchySqueeze}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SummableTailModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, S, R, B, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SummableTailModulusUp.SummableTailModulusTasteGate single carrier alignment`

Theorem 1799.2 (Summable tail modulus NameCert obligations)**.** *For every accepted* $\mathsf{SummableTailModulus}^{\uparrow}$ *carrier*

$$T = (M, D, S, R, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummableTailModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the summable tail modulus row, dyadic tolerance row, StreamName finite window, RegSeqRat readback, Bishop regular Cauchy completion boundary, Real seal, transport, replay, provenance, and local naming rows;*
2. *summability schedule exactness, requiring M to store the displayed summability ledger and the tail index attached to each dyadic tolerance request;*
3. *monotone tail-index stability, requiring later displayed tolerances to reuse or enlarge the tail index recorded by M without replacing it by a hidden selector;*
4. *tolerance compatibility, requiring D to be the same dyadic scale consumed by M and later by the finite window S;*
5. *readback handoff, requiring S and R to expose only the finite StreamName window and RegSeqRat data opened by the scheduled tail request;*
6. *completion-boundary handoff, requiring B to consume M, D, S, R before any completion-facing boundary is visible;*
7. *Real-seal non-escape, requiring E to be terminal with respect to M, D, S, R, B and to export no quotient equality, selected limit, countable-choice schedule, trichotomy, host equality, ambient completeness, Lean verification, or bridge data;*
8. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No summable tail modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1799.1. The visible packet consists exactly of the summable tail modulus row M, dyadic tolerance row D, finite StreamName window S, RegSeqRat readback R, completion boundary B, Real seal E, and structural rows H, C, P, N.

Summability schedule exactness is the local content of M. The row stores the finite partial-tail ledger and the tail index for each displayed tolerance request, so the schedule is not recovered from an unrecorded convergence theorem or selected limit.

Monotone tail-index stability follows from the same row. Later displayed tolerances may reuse or enlarge the named tail index, but the carrier has no separate selector that could replace the finite schedule after the request has entered M.

Tolerance compatibility and readback handoff are downstream. The row D is the only dyadic budget consumed by the scheduled request and the window row S, while R reads back only the observations opened by that window.

Finally, B, E, H, C, P, N expose the completion boundary, terminal seal, transport, replay, provenance, and local name along the same displayed route. Since the excluded objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1799.3 (Summable tail modulus regular Cauchy criterion)**.** *Let*

$$T = (M, D, S, R, B, E, H, C, P, N)$$

be an accepted $\mathsf{SummableTailModulus}^{\uparrow}$ *carrier. Every accepted summable tail request routes through the finite chain*

$$M \longmapsto D \longmapsto S \longmapsto R$$

before any completion boundary is exposed. Consequently the request reaches B only after it has displayed the summability schedule, dyadic tolerance budget, finite StreamName window, and RegSeqRat readback.

Proof. Project the accepted carrier. The only coordinate that stores summability data is M, and the only coordinate that stores the tolerance budget for that data is D.

The finite observation route is then forced by the tuple order. The StreamName row S opens the window named by the modulus row and budgeted by the tolerance row, and the RegSeqRat row R reads back that finite window.

The completion boundary B is downstream of M, D, S, R. Therefore a completion-facing consumer cannot expose B before the summable tail request has passed through the displayed modulus, tolerance, window, and readback rows. □

Theorem 1799.4 (Summable tail modulus Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SummableTailModulus}^{\uparrow}$ *carrier reads the terminal seal E only after the finite route*

$$M \longmapsto D \longmapsto S \longmapsto R \longmapsto B \longmapsto E$$

has been displayed. The consumer receives the routed summable tail modulus, dyadic tolerance, finite StreamName window, RegSeqRat readback, completion boundary, and terminal Real seal, but receives no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Read the Real-facing request through Definition 1799.3. Before the request reaches any completion boundary, it has already displayed the modulus, tolerance, window, and readback rows.

The seal row E is listed after the completion boundary B in Definition 1799.1. Thus every Real-facing consumer reaches E only after the displayed M, D, S, R, B route.

The structural rows H, C, P, N transport, replay, source, and name that route. They add no quotient coordinate, selected limit, choice operator, trichotomy principle, host equality predicate, ambient completeness theorem, Lean verification row, or bridge datum. □

Falsifiable prediction: Every $\mathsf{SummableTailModulus}^{\uparrow}$ Real-facing tail-control read factors through one summable tail modulus row, one DyadicToleranceScale row, one StreamName finite window, one RegSeqRat readback row, one Bishop regular Cauchy completion boundary row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-majorant-series, regular-cauchy-squeeze, dyadic-tolerance-scale, streamname, regseqrat, real: $\mathsf{SummableTailModulus}^{\uparrow}$ is not bijective to any listed sibling because

it jointly carries a summable tail schedule, monotone tail-index stability, dyadic tolerance control, finite observation window, regular rational readback, and a completion-to-Real seal handoff as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SummableTailModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummableTailModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1799.1, Definition 1799.2, Definition 1799.3, and Definition 1799.4 seed the finite summable tail modulus route through dyadic tolerance, StreamName finite window, RegSeqRat readback, completion boundary, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummableTailModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, summability schedule exactness, monotone tail-index stability, tolerance compatibility, readback handoff, completion-boundary handoff, Real-seal non-escape, transport, replay, provenance, and finite route factorization for BEDC.Derived.SummableTailModulusUp.

Constructive story: A $\mathsf{SummableTailModulus}^{\uparrow}$ packet is generated from one summable tail modulus row, one DyadicToleranceScale row, one StreamName finite window, one RegSeqRat readback row, one Bishop regular Cauchy completion boundary row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1800

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ packet records the finite diagonal-limit seal that joins the budget surface of chapter 853, the diagonal-limit surface of chapter 374, the representative surface of chapter 1205, the regular Cauchy extraction route of chapter 1773, and the terminal real-name boundary of chapter 38. It is the public L10 packet by which separate diagonal budget, limit, representative, extraction, and Real seal rows are consumed as one finite NameCert route without importing host completion.

Definition 1800.1 (Cauchy diagonal limit carrier)**.** *A* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\mathrm{diag}} = (R, B, D, V, E, S, Q, Y, T, H, C, P, N)$$

with the following displayed rows:

1. *R is the precision-request row, naming the Real-facing tolerance request and the finite diagonal window it is permitted to inspect.*
2. *B is the* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *row, recording the budget, slack, and comparison ledger attached to the request R.*
3. *D is the* $\mathsf{DiagonalLimit}^{\uparrow}$ *row, storing the displayed diagonal-limit coordinate reached after the budget row has been read.*
4. *V is the* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *row, fixing the representative readback used by the diagonal-limit coordinate.*
5. *E is the* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *row, routing the representative through the source family, selector, aligned windows, dyadic ledger, and regular rational diagonal readback.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the finite windows selected by the diagonal route.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the selected windows and the displayed tolerance data.*
8. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row containing the dyadic radius, slack, and comparison entries consumed by B, E, S, Q.*
9. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after R, B, D, V, E, S, Q, Y have been displayed.*

10. *H records componentwise* hsame *transport for the precision request, budget, diagonal-limit row, representative row, extraction row, StreamName readback, RegSeqRat readback, DyadicRatCore ledger, Real seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from each Real-facing precision request through $R, B, D, V, E, S, Q, Y, T, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$, $\mathsf{CauchyDiagonalBudget}^{\uparrow}$, $\mathsf{DiagonalLimit}^{\uparrow}$, $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, B, D, V, E, S, Q, Y, T, H, C, P, N$. It has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalLimitUp.TasteGate.CauchyDiagonalLimitTasteGate single carrier alignment`

Theorem 1800.2 (Cauchy diagonal limit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ *carrier*

$$L_{\mathrm{diag}} = (R, B, D, V, E, S, Q, Y, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyDiagonalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the precision request,* $\mathsf{CauchyDiagonalBudget}^{\uparrow}$ *budget row,* $\mathsf{DiagonalLimit}^{\uparrow}$ *diagonal-limit row,* $\mathsf{DiagonalLimitRepresentative}^{\uparrow}$ *representative row,* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *extraction row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *budget exactness, requiring B to be the only row that assigns radius, slack, and comparison data to the precision request R;*
3. *diagonal-limit routing, requiring D to be read only after B and before any representative, extraction, or Real-seal coordinate is consumed;*
4. *representative exactness, requiring V to be the only representative row attached to the diagonal-limit coordinate D;*
5. *extraction exactness, requiring E to route the representative through the displayed regular Cauchy diagonal extraction surface;*
6. *readback alignment, requiring S, Q, Y to expose only the StreamName windows, RegSeqRat diagonal readback, and DyadicRatCore ledger selected by R, B, D, V, E;*
7. *Real seal non-escape, requiring T to be downstream of the finite budget-limit-representative-extraction-readback route;*
8. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
9. *refusal of quotient streams, selected host limits, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No Cauchy diagonal limit obligation is stored outside these rows.

Proof. Project the carrier of Definition 1800.1. The precision request R is the only public entry point, and the budget row B is the only row that assigns the dyadic radius, slack, and comparison data for that request.

The diagonal-limit route is then forced. The row D reads the limit coordinate only after B, while V fixes the representative attached to that coordinate before any extraction or seal can be inspected.

The extraction and readback rows are downstream of the representative. The row E supplies the regular Cauchy diagonal extraction route, and S, Q, Y expose exactly the selected finite windows, regular rational readback, and dyadic ledger.

Finally, the Real seal T and structural rows H, C, P, N preserve the same finite route. Since quotient streams, selected host limits, countable choice, trichotomy, ambient completeness, Lean evidence, and bridge data do not occur as coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 1800.3 (Cauchy diagonal limit Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing diagonal limit read from an accepted* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ *carrier*

$$L_{\mathrm{diag}} = (R, B, D, V, E, S, Q, Y, T, H, C, P, N)$$

factors through the displayed finite route

$$R \longmapsto B \longmapsto D \longmapsto V \longmapsto E \longmapsto S, Q, Y \longmapsto T \longmapsto H, C, P, N.$$

Consequently the consumer receives one precision request, one CauchyDiagonalBudget row, one DiagonalLimit row, one DiagonalLimitRepresentative row, one RegularCauchyDiagonalExtraction row, aligned StreamName windows, one RegSeqRat readback, one DyadicRatCore ledger, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. The route exports no quotient stream, selected host limit, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Proof. Begin with the accepted carrier and read the precision request row R. A Real-facing consumer has no direct access to a diagonal limit or a representative; it must first pass through the budget row B.

After the budget is fixed, the diagonal-limit coordinate D and representative row V determine the only extraction route available to the consumer. The extraction row E then supplies the regular Cauchy diagonal route.

The readback rows S, Q, Y are aligned with that route. They expose the finite StreamName windows, the RegSeqRat readback, and the DyadicRatCore ledger selected by the budget, limit, representative, and extraction rows.

The terminal Real seal T is therefore downstream of the entire route, and H, C, P, N merely transport, replay, record provenance, and name the same packet. Because the carrier has no quotient-stream coordinate, selected host limit, countable-choice schedule, trichotomy principle, ambient completeness theorem, Lean evidence, or bridge data, every Real-facing read factors through exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ packet is generated from a precision request, a CauchyDiagonalBudget row, a DiagonalLimit row, a DiagonalLimitRepresentative row, a RegularCauchyDiagonalExtraction row, StreamName windows, a RegSeqRat readback, a DyadicRatCore ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyDiagonalLimit}^{\uparrow}}$, seed)

Scope closed: Definition 1800.1, Definition 1800.2, and Definition 1800.3 seed the finite diagonal-limit route joining budget, limit, representative, extraction, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyDiagonalLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget exactness, diagonal-limit routing, representative exactness, extraction exactness, readback alignment, Real seal non-escape, componentwise transport, displayed replay, and local naming for BEDC.Derived.CauchyDiagonalLimitUp.

Chapter 1801

A Concrete Naming Certificate for $\mathsf{LocatedSequenceModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedSequenceModulus}^{\uparrow}$ packet records the finite data by which a located rational sequence exposes a reusable modulus surface before it is consumed as a regular Cauchy or real-name object. It reads the located bounds of **??**, the stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the modulus surface of chapter 292, and the real-name seal of chapter 38. The chapter isolates the constructive sequence-modulus packet used by the current real and completion route without assuming quotient streams, selected limits, trichotomy, countable choice, or ambient real completeness.

Definition 1801.1 (Located sequence modulus carrier)**.** *A* $\mathsf{LocatedSequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located sequence-window row. It records the finite rational interval or ball witnesses available at each inspected sequence index, together with the BEDC marks that say which windows are presently exposed.*
2. *M is the modulus schedule row. It assigns each displayed tolerance request to a finite index bound and records monotone reuse when a tighter or later request is inspected.*
3. *D is the dyadic tolerance-scale row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and carrying the rational error budgets against which the sequence windows are compared.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It names exactly the finite stream coordinates that may be queried before the regular rational or real-facing read is formed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular Cauchy-facing rational sequence obtained from the located windows, the schedule, the dyadic tolerance scale, and the stream-name window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It packages the real-facing consumer boundary after the regular rational readback has been displayed, without adding a selected limit or quotient-stream coordinate.*
7. *H records componentwise* hsame *transport for the located windows, modulus schedule, dyadic tolerance scale, stream-name window, regular rational readback, real seal, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from located finite windows through the modulus schedule, dyadic tolerances, stream-name windows, regular rational readback, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedSequenceModulus}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceModulus}^{\uparrow}}$ *row.*

The carrier exposes only the finite located-window, modulus, tolerance, stream, readback, and seal boundary. It contains no quotient stream, selected limit, trichotomy principle, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification certificate, or checked bridge.*

Lean4 ✓ `BEDC.Derived.LocatedSequenceModulusUp.LocatedSequenceModulusTasteGate single carrier alignment`

Theorem 1801.2 (Located sequence modulus NameCert obligations)**.** *For every accepted* LocatedSequenceModulus *carrier $B = (L, M, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located sequence windows, modulus schedule, dyadic tolerance scale, stream-name window, regular rational readback, real seal, transport, continuation, provenance, and local naming rows;*
2. *located-window coverage, requiring every sequence coordinate used by a consumer to be witnessed in L before it can be compared against the tolerance scale;*
3. *monotone modulus stability, requiring a bound stored in M to remain valid under later index requests and tighter finite tolerance requests displayed by the packet;*
4. *dyadic tolerance compatibility, requiring each comparison between sequence windows and scheduled bounds to use the row D rather than an ambient rational approximation convention;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the readback row R to be formed only after the located-window, modulus, tolerance, and stream-name rows have been projected;*
6. *real-seal non-escape, requiring E, P, N to export no quotient stream, selected limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification certificate, or checked bridge.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1801.1. The packet displays the located-window row L, modulus schedule M, dyadic tolerance row D, stream-name row S, regular rational readback R, real seal E, and the structural rows H, C, P, N.

Located-window coverage is forced before any readback can occur. A consumer that inspects a sequence coordinate must obtain the corresponding finite witness from L, and the witness is compared only after the tolerance request has been routed through D.

Monotone modulus stability is stored in the schedule row. The same scheduled bound may be reused for later index windows or tighter displayed tolerances only through M, so no external choice of a bound is available to the consumer.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is then determined by the visible route L, M, D, S, R. The readback row R cannot be projected until the located windows, schedule, dyadic tolerance scale, and stream-name window have already been displayed.

Finally, the real seal E is downstream of that readback. Since the remaining rows only transport, replay, package, and name the displayed finite data, the local NameCert has no coordinate

from which a quotient stream, selected limit, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification certificate, or checked bridge could be obtained. □

Theorem 1801.3 (Located sequence modulus regular Cauchy handoff)**.** *Let $B = (L, M, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedSequenceModulus}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing consumer of B factors through the finite route*

$$L \longmapsto M \longmapsto D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer reads located sequence windows, a monotone modulus schedule, dyadic tolerances, a $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal as one BEDC packet. The route exposes no quotient stream, selected limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification certificate, or checked bridge.*

Proof. Project the accepted carrier rows. A regular Cauchy-facing consumer first reads the located sequence data through L and obtains its finite bound discipline through M.

The tolerance and stream coordinates are then fixed by D and S. These rows determine which dyadic error budgets and stream-name windows can participate in the readback, so the consumer has no second route to an unrecorded approximation convention.

The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, and the row E is the only $\mathsf{Real}^{\uparrow}$ seal surface. Any real-facing consumer must therefore pass through R before reading E.

The structural rows H, C, P, N close the displayed route by transporting, replaying, packaging, and naming exactly those finite coordinates. Because the packet contains no additional completion or choice coordinate, the non-escape claims follow from exhaustion of the carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSequenceModulus}^{\uparrow}$ packet is generated from located sequence windows, a monotone modulus schedule, dyadic tolerance data, a StreamName window, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1801.1, Definition 1801.2, and Definition 1801.3 seed the finite located sequence modulus boundary for the current real and completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window coverage, monotone modulus stability, dyadic tolerance compatibility, RegSeqRat handoff, real-seal non-escape, and the regular Cauchy handoff for BEDC.Derived.LocatedSequenceModulusUp.

Chapter 1802

Rational Interval Enclosure NameCert Construction

Following Bishop–Bridges constructive analysis, the $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ packet records the finite rational bracket by which a regular Cauchy real name becomes observable before any quotient-real or completeness principle is invoked. It sits over the rational carrier of chapter 37, the dyadic tolerance and rounding discipline of chapter 291, the regular rational stream boundary of chapter 301, the terminal real-name seal of chapter 38, and the located interval surface of chapter 1709. The packet names finite endpoint rows, not a completed equality class of real numbers.

Definition 1802.1 (RationalIntervalEnclosure carrier)**.** *A* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, U, Q, D, S, R, I, T, H, C, P, N)$$

with the following displayed rows:

1. *L is the rational lower-endpoint row, read through* $\mathsf{Rat}^{\uparrow}$ *and carrying the displayed lower bracket value.*
2. *U is the rational upper-endpoint row, read through* $\mathsf{Rat}^{\uparrow}$ *and carrying the displayed upper bracket value.*
3. *Q is the rational order ledger, recording $L \leq U$ and the rational comparisons requested by the enclosure consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic radius, rounding rows, and rational-to-dyadic comparison data used to justify the displayed bracket width.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, naming the regular Cauchy rational stream inspected by the bracket.*
6. *R is the regular readback row, recording the finite index and tolerance request at which the source stream is read into the bracket.*
7. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *interval row, exposing the bracket as located interval data without adding a host interval object.*
8. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, Q, D, S, R, I have been displayed.*
9. *H records componentwise* hsame *transport for rational endpoints, rational order, dyadic tolerance, regular source, regular readback, located interval, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an enclosure request through $L, U, Q, D, S, R, I, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalEnclosure}^{\uparrow}}$ *row.*

The classifier compares accepted enclosures only through $L, U, Q, D, S, R, I, T, H, C, P, N$. It has no coordinate for Dedekind-cut equivalence, host real equality, trichotomy, quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RationalIntervalEnclosureUp.RationalIntervalEnclosureUp`

Theorem 1802.2 (RationalIntervalEnclosure NameCert obligations)**.** *For every accepted* RationalIntervalEnclosure *carrier*

$$E = (L, U, Q, D, S, R, I, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalEnclosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational endpoint rows, rational order ledger, dyadic tolerance ledger, regular Cauchy source, regular readback, located interval row, Real seal, transports, replay, provenance, and local naming;*
2. *rational endpoint exactness, requiring every enclosure read to expose L, U, Q before any dyadic tolerance or real-facing row is available;*
3. *dyadic tolerance compatibility, requiring D to justify the displayed bracket radius and endpoint comparisons through* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows rather than through host real arithmetic;*
4. *regular-source readback, requiring S and R to identify the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *name, index, and tolerance request from which the bracket is read;*
5. *located-interval routing, requiring I to expose the same rational endpoints and tolerance ledger as located interval data;*
6. *Real-seal ordering, requiring T to be read only after L, U, Q, D, S, R, I have been displayed;*
7. *ledger non-escape, requiring H, C, P, N to export no Dedekind-cut equivalence, host real equality, trichotomy, quotient stream, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No RationalIntervalEnclosure obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1802.1. The accepted packet displays exactly the lower and upper rational endpoint rows, rational order ledger, dyadic tolerance ledger, regular Cauchy source, regular readback, located interval row, terminal Real seal, and structural rows.

Rational endpoint exactness is the first stage. The rows L, U, Q are the only endpoint and order coordinates available to the classifier, so a consumer cannot replace the bracket by a host real interval or a completed equality class.

Dyadic tolerance compatibility and regular-source readback form the middle stage. The row D records the finite dyadic radius and rounding comparisons, while S and R identify the regular Cauchy rational name, index, and tolerance request that produced the displayed endpoint bracket.

The located-interval and Real-seal rows are downstream. The row I exposes the same endpoint and tolerance data as located interval evidence, and T is terminal because it appears only after the endpoint, dyadic, source, and interval rows.

Finally, H, C, P, N transport, replay, package, and name the same finite route. Since the excluded cut, equality, trichotomy, quotient, selection, choice, and completeness objects are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1802.3 (RationalIntervalEnclosure RegSeqRat readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read of an accepted* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ *carrier factors through the displayed route*

$$S \longmapsto R \longmapsto D \longmapsto L, U, Q \longmapsto I \longmapsto T, H, C, P, N.$$

Consequently the consumer receives one regular Cauchy rational source, one finite readback row, one dyadic tolerance ledger, rational lower and upper endpoint rows with their order ledger, one located interval row, one terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. The readback supplies a finite bracket around the name and does not identify quotient streams or host real equality.*

Proof. Project the accepted carrier $E = (L, U, Q, D, S, R, I, T, H, C, P, N)$. The only regular Cauchy source coordinate is S, and the only finite readback coordinate attached to it is R.

The tolerance stage then passes through D. This row records the dyadic radius and rounding comparisons that make the rational endpoints readable at the requested precision.

The endpoint stage is L, U, Q. These rows expose the lower endpoint, upper endpoint, and rational order ledger after the source and tolerance request have been fixed.

The downstream stage is I, T, H, C, P, N. The located interval row and terminal Real seal consume the displayed bracket, while the structural rows preserve and name the same route. No quotient-stream or host-real-equality coordinate is present, so every $\mathsf{RegSeqRat}^{\uparrow}$ readback factors through the displayed finite bracket. □

Theorem 1802.4 (RationalIntervalEnclosure Real-seal nonescape)**.** *Let* $E = (L, U, Q, D, S, R, I, T, H, C, P, N)$ *be an accepted* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *consumer read of* E *must pass through the rational endpoint rows, dyadic tolerance ledger, regular-source readback, and located interval row before the terminal seal* T *is available. Therefore the seal exports only the displayed finite rational bracket and its structural replay; it exports no Dedekind-cut equivalence, host real equality, trichotomy, quotient streams, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The terminal Real-facing coordinate of the carrier is T. By the displayed packet order in Definition 1802.1, T is downstream of L, U, Q, D, S, R, I.

A $\mathsf{LocatedRealInterval}^{\uparrow}$ consumer must therefore read I through the same endpoint and tolerance rows that produced the bracket. There is no second interval coordinate and no host interval object.

A $\mathsf{Real}^{\uparrow}$ consumer must also read through the same route before reaching T. The structural rows H, C, P, N then transport, replay, package, and name only that route.

Since the carrier inventory omits Dedekind-cut equivalence, host real equality, trichotomy, quotient streams, selected limits, countable choice, and ambient Real completeness, none of those objects can escape through the terminal seal. □

Falsifiable prediction: Every $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ read factors through rational lower and upper endpoint rows, one rational order ledger, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ source row, one finite regular readback row, one $\mathsf{LocatedRealInterval}^{\uparrow}$ interval row, one $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: rat, dyadicratcore, regseqrat, located-real-interval, real: $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rational endpoints, rational order,

dyadic tolerance, regular Cauchy readback, located interval exposure, and a terminal Real-facing seal as one finite bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalEnclosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalEnclosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1802.1, Definition 1802.2, Definition 1802.3, and Definition 1802.4 seed the finite rational interval enclosure route from regular Cauchy rational names through rational endpoint rows, dyadic tolerance, located interval exposure, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalEnclosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut equivalence, host real equality, trichotomy, quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational endpoint exactness, dyadic tolerance compatibility, regular-source readback, located-interval routing, Real-seal ordering, ledger non-escape, and consumer factorization for BEDC.Derived.RationalIntervalEnclosureUp.

Constructive story: A $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ packet is generated from rational lower and upper endpoint rows, a rational order ledger, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ source row, one finite readback row, a $\mathsf{LocatedRealInterval}^{\uparrow}$ interval row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1803

A Concrete Naming Certificate for $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley on filters and Moore–Smith nets, the $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ packet records the finite handoff between Cauchy-filter evidence and Cauchy-net evidence before either route is consumed by completion. It sits over the Cauchy-filter surface of chapter 333, the Cauchy-net directed-window surface of chapter 589, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the metric completion consumer of chapter 1506, the uniform completion consumer of chapter 1473, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite L10 bridge: it records displayed evidence routes, not quotient filters, selected host nets, countable choice, selected limits, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1803.1 (Cauchy filter-net correspondence carrier)**.** *A* $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, D, W, R, T, M, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, recording the filter-base requests, late-set witnesses, and finite Cauchy evidence available before a directed net is read.*
2. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, recording the directed index surface and finite upper-window witnesses generated from the displayed filter requests.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by both the filter read and the directed-window read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from those windows.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by the filter and net evidence routes.*
6. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row, visible only after F, D, W, R, T have been displayed.*
7. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer row, visible only after the same filter-net handoff route has been displayed.*
8. *H records componentwise* hsame *transport for $F, D, W, R, T, M, U, C, P, N$.*
9. *C records displayed* Cont *replay from a finite Cauchy-filter request through the directed-window handoff and then to the completion consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected host nets, countable-choice schedules, selected limits, ambient completion, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyFilterNetCorrespondenceUp.CauchyFilterNetCorrespondenceTasteGate`
`single carrier alignment`

Theorem 1803.2 (Cauchy filter-net NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ *carrier*

$$Q = (F, D, W, R, T, M, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}}$ *surface consists exactly of four finite obligation rows:*

1. *carrier exposure: the accepted object exposes only the displayed packet Q with rows $F, D, W, R, T, M, U, H, C, P, N$*
2. *filter-to-net routing: every filter-base Cauchy request reaches the directed-window row D only through the displayed filter row F and the shared tolerance row T;*
3. *net-to-filter routing: every directed-window read back to filter evidence returns through the same D, F, T boundary before any completion consumer sees the route;*
4. *consumer non-escape: the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *rows read only the displayed F, D, W, R, T route, with H, C, P, N supplying transport, replay, provenance, and local naming.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1803.1. The accepted packet contains the filter row, directed-window row, window row, readback row, tolerance row, the two completion consumer rows, and the structural rows H, C, P, N.

For the filter-to-net obligation, start with a finite Cauchy request in F. The only directed output coordinate in the carrier is D, and the route to it is mediated by the shared dyadic tolerance row T before any window in W or readback in R can be inspected.

For the net-to-filter obligation, reverse the displayed route at the level of evidence rows. A directed-window read in D returns to the same filter boundary F and uses the same tolerance row T, so the reverse handoff cannot introduce a second base, a host subnet, or a choice-selected branch.

Finally, M and U are downstream consumer rows. Since they occur after F, D, W, R, T in the carrier and have only H, C, P, N as structural support, they receive exactly the displayed handoff route and no quotient, choice, selected-limit, ambient-completion, Real-completeness, verification, or bridge coordinate. □

Theorem 1803.3 (Cauchy filter-to-net finite handoff)**.** *Let $Q = (F, D, W, R, T, M, U, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ *carrier. Every displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *request in F that is admitted to a completion-facing route factors through a finite directed-window handoff*

$$F, T \longmapsto D \longmapsto W, R.$$

Thus the filter branch supplies Cauchy evidence to $\mathsf{CauchyNet}^{\uparrow}$ *as displayed directed-window data, not as a quotient filter, selected host net, countable-choice selector, selected limit, or ambient completion object.*

Proof. Project the accepted carrier and read the filter row F. A completion-facing filter request has no direct edge to M or U inside the packet; the displayed bridge route first records its tolerance demand in T.

The directed-window row D is then the only net-facing coordinate. The replay row C sends the finite request from F, T to D, and H keeps the row identities stable under componentwise transport.

After D is displayed, the route may inspect finite windows W and regular rational readback R. The carrier contains no excluded quotient, host-net, choice, selected-limit, or ambient-completion coordinate, so the handoff remains the finite route shown above. □

Theorem 1803.4 (Cauchy net-to-filter finite handoff)**.** *Let $Q = (F, D, W, R, T, M, U, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ *carrier. Every displayed* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window read in D that is compared with filter Cauchy evidence factors through*

$$D, T \longmapsto F \longmapsto W, R.$$

The reverse read returns to the displayed $\mathsf{CauchyFilter}^{\uparrow}$ *row and does not export quotient-filter equality, selected host nets, countable choice, selected limits, ambient completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the directed-window row D. The carrier makes no autonomous completion claim from D alone; any comparison with filter evidence must use the same dyadic tolerance row T that the forward handoff used.

The filter row F is the unique filter-facing coordinate. Replaying the route through C therefore returns the directed-window read to F before windows or regular rational readback are exposed.

The downstream rows W and R are shared by both directions. Since P and N name only this packet and H transports only its displayed rows, the reverse handoff cannot create quotient-filter equality, host-net selection, countable choice, selected limits, ambient completion, or Real completeness. □

Theorem 1803.5 (Cauchy filter-net consumer non-escape)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ *carrier factors through the displayed route*

$$F, D, T, W, R \longmapsto M, U.$$

Consequently the completion consumers may read finite Cauchy-filter evidence, finite directed-window Cauchy-net evidence, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance data, but they receive no quotient filters, selected host nets, countable choice, selected limits, ambient completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The consumer rows M and U are coordinates of the carrier in Definition 1803.1. They are not source rows; each is reached only after the filter row F, directed-window row D, tolerance row T, window row W, and readback row R have been displayed.

Apply Definition 1803.3 for reads that start from filter evidence, and apply Definition 1803.4 for reads that start from directed-window evidence. Both routes meet the same finite F, D, T, W, R boundary before any completion consumer is visible.

The $\mathsf{Real}^{\uparrow}$-facing part of the route is therefore mediated by the same readback and completion rows. Because the carrier has no excluded quotient, selected-host, choice, selected-limit, ambient-completion, or Real-completeness coordinate, every consumer read remains inside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1803.1, Definition 1803.2, Definition 1803.3, Definition 1803.4, and Definition 1803.5 seed the finite L10 correspondence surface between filter and net Cauchy evidence for completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected host nets, countable choice, selected limits, ambient completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, filter-to-net routing, net-to-filter routing, consumer non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterNetCorrespondenceUp.

Constructive story: A $\mathsf{CauchyFilterNetCorrespondence}^{\uparrow}$ packet is generated from one CauchyFilter row, one CauchyNet directed-window row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance row, MetricCompletion and UniformCompletion consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1804

A Concrete Naming Certificate for $\mathsf{BisectionConvergence}^{\uparrow}$

Bishop–Bridges constructive bisection/interval-halving analysis treats convergence as finite retained-window data at each requested precision, not as a hidden completed choice of a limit point. The $\mathsf{BisectionConvergence}^{\uparrow}$ packet names that finite route over the dyadic branch surface of chapter 361, the located interval bisection surface of chapter 1711, the retained nested-window discipline of chapter 320, the finite StreamName interface of chapter 126, the RegSeqRat readback route of chapter 301, and the terminal Real seal of chapter 38. It supplies a named convergence surface for intermediate-value and root-bracketing consumers without exporting selected limits, quotient stream equality, real trichotomy, countable choice, or compactness principles.

Definition 1804.1 (Bisection convergence carrier)**.** *A* $\mathsf{BisectionConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, D, W, T, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row. It records the initial lower and upper endpoint windows, locatedness reads, endpoint order data, and the interval identity consumed by the finite bisection request.*
2. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger. It stores the finite left-or-right branch prefix, midpoint cells, child-window records, and branch depth used by the requested tolerance.*
3. *W is the retained* $\mathsf{NestedInterval}^{\uparrow}$ *window row. It records the lower and upper retained windows after each branch listed in D.*
4. *T is the tail-diameter schedule. It assigns to each requested tolerance a displayed branch depth and the retained-window diameter bound available at that depth.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened from W at the depth named by T.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite StreamName windows and the displayed diameter budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the retained windows, tail-diameter schedule, StreamName windows, and RegSeqRat readback have been displayed.*

8. *H records componentwise* hsame *transport for the located interval source, branch ledger, retained windows, tail schedule, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a convergence request through interval inspection, finite branch retention, tail-diameter scheduling, finite window opening, regular readback, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{BisectionConvergence}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, W, T, S, Q, E, H, C, P, N$. It has no coordinate for selected limits, quotient stream equality, real trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host interval equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BisectionConvergenceUp.BisectionConvergenceTasteGate single carrier alignment`

Theorem 1804.2 (Bisection convergence NameCert obligations)**.** *For every accepted* $\mathsf{BisectionConvergence}^{\uparrow}$ *carrier*

$$B = (I, D, W, T, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, dyadic branch ledger, retained nested windows, tail-diameter schedule, StreamName windows, RegSeqRat readback, Real seal, transports, replay, provenance, and local naming;*

2. *source discipline, requiring every endpoint, locatedness, and order read to factor through I before any branch or tail row is inspected;*

3. *branch-retention discipline, requiring D to display the finite branch prefix and requiring W to store only the retained windows produced by that prefix;*

4. *tail-diameter exactness, requiring T to attach each requested tolerance to a displayed branch depth and a displayed retained-window diameter bound;*

5. *finite readback discipline, requiring S and Q to consume the retained window and tail schedule before any regular rational observation is visible;*

6. *Real-seal non-escape, requiring E, H, C, P, N to export only the displayed finite route and no selected limit, quotient stream equality, real trichotomy, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host interval equality, Lean verification, or bridge data.*

No bisection convergence obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1804.1. The visible packet consists exactly of the source interval row I, branch ledger D, retained-window row W, tail-diameter schedule T, StreamName row S, RegSeqRat readback Q, Real seal E, and structural rows H, C, P, N.

Source discipline is the first stage of the route. The endpoint windows, locatedness reads, and order data are supplied by I, so a convergence request cannot replace the source by a host interval equality theorem or by a completed limit object.

Branch-retention discipline is the middle stage. The ledger D displays the finite branch prefix and midpoint cells, while W records only the retained windows produced by that ledger.

Tail-diameter exactness is local to T. For each requested tolerance, the schedule stores the branch depth and the corresponding retained-window diameter bound; no unrecorded convergence principle or countable choice operator chooses the depth.

Finally, S, Q, E, H, C, P, N expose finite windows, regular readback, terminal seal, transport, replay, provenance, and naming along the same displayed route. Since the excluded objects have no carrier coordinate, no local NameCert obligation can export them. □

Theorem 1804.3 (Bisection convergence tail-diameter route)**.** *Let*

$$B = (I, D, W, T, S, Q, E, H, C, P, N)$$

be an accepted $\mathsf{BisectionConvergence}^{\uparrow}$ *carrier. Every accepted convergence request factors through the finite route*

$$I \longmapsto D \longmapsto W \longmapsto T$$

before any $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is exposed. Consequently each requested tolerance is answered by a displayed branch depth and retained-window diameter bound rather than by a selected limit or ambient completeness principle.*

Proof. Project the accepted carrier. The only source interval coordinate is I, and the only branch coordinate is D, so the request first reads the located interval data and then the finite branch prefix.

The retained-window row W is downstream of that branch prefix. It stores the lower and upper windows that remain after the displayed bisection choices, and it provides the visible window data consumed by the tail schedule.

The tail-diameter row T then answers the tolerance request. It names the branch depth and diameter bound available for the retained window at that depth.

Since S and Q occur only after T in the carrier route, no StreamName or RegSeqRat readback is visible before the finite diameter schedule has been displayed. The route therefore supplies finite convergence data without selected limits or ambient completeness. □

Theorem 1804.4 (Bisection convergence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BisectionConvergence}^{\uparrow}$ *carrier reads the terminal seal* E *only after the finite route*

$$I \longmapsto D \longmapsto W \longmapsto T \longmapsto S \longmapsto Q \longmapsto E$$

has been displayed. The consumer receives the located interval source, dyadic branch ledger, retained nested windows, tail-diameter schedule, StreamName windows, RegSeqRat readback, and terminal Real seal, but receives no selected limit, quotient stream equality, real trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host interval equality.*

Proof. Read the Real-facing request through Definition 1804.3. Before any readback is exposed, the request has already displayed the located interval source, finite branch ledger, retained windows, and tail-diameter schedule.

The finite observation stage is S followed by Q. The StreamName row opens the retained windows at the scheduled depth, and the RegSeqRat row reads back only those displayed finite observations under the recorded diameter budget.

The Real seal E is terminal with respect to that route. Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed data and introduce no analytic coordinate beyond it.

Thus every Real-facing consumer receives the named finite bisection convergence handoff. The excluded limit-selection, quotient-equality, trichotomy, countable-choice, compactness, completeness, and host-equality principles have no row from which they could be projected. □

Falsifiable prediction: Every $\mathsf{BisectionConvergence}^{\uparrow}$ Real-facing convergence read factors through one located interval source row, one DyadicBisection branch ledger, one retained NestedInterval window row, one tail-diameter schedule, one StreamName finite-window row, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicbisection, located-interval-bisection, nestedinterval, streamname, regseqrat, real, real-root-bracket: $\mathsf{BisectionConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite branch ledger, retained nested windows, a tolerance-indexed tail-diameter schedule, finite StreamName windows, RegSeqRat readback, and a Real-seal handoff as one convergence certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1804.1, Definition 1804.2, Definition 1804.3, and Definition 1804.4 seed the finite bisection convergence route through located interval source data, dyadic branch ledger, retained nested windows, tail-diameter schedule, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, real trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host interval equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source discipline, branch-retention discipline, tail-diameter exactness, finite readback discipline, Real-seal non-escape, and route factorization for BEDC.Derived.BisectionConvergenceUp.

Constructive story: A $\mathsf{BisectionConvergence}^{\uparrow}$ packet is generated from one located interval source row, one DyadicBisection branch ledger, one retained NestedInterval window row, one tail-diameter schedule, one StreamName finite-window row, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1805

A Concrete Naming Certificate for $\mathsf{LocatedCompletionSeal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompletionSeal}^{\uparrow}$ packet names the finite seal by which located Cauchy source data reaches the terminal Real route without turning that route into an ambient completeness theorem. It sits over the regular Cauchy completion boundary of chapter 1639, the located regular source of chapter 1454, the tail envelope of chapter 1769, the regular rational route of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite located-completion boundary for $\mathsf{Real}^{\uparrow}$ consumers; it is not Real completeness, quotient stream equality, a selected limit, countable choice, trichotomy, or host real equality.

The same seal is consumed by chapter 3703, where the dense unit and completion seal become the extension-and-uniqueness source row.

Definition 1805.1 (Located completion seal carrier)**.** *A* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, Q, E, A, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance row, containing the dyadic radii, comparisons, and interval endpoints read by the located completion route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before any regular rational or located-tail read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, recording the regular rational observations available on the windows named by S and measured by D.*
4. *Q is the* $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$ *source row, binding the regular rational source to the displayed locatedness and modulus data.*
5. *E is the* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ *row, recording the finite tail window through which the located source is exposed to completion consumers.*
6. *A is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *completion-facing boundary row, consuming the same dyadic, stream, and regular rational data as D, S, R.*
7. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after D, S, R, Q, E, A have been displayed.*
8. *H records componentwise* hsame *transport for dyadic, stream, regular rational, located-source, tail-envelope, completion-boundary, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each located-completion precision request through $D, S, R, Q, E, A, T, H, P, N$*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionSeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularLocatedCauchyReal}^{\uparrow}$, $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, Q, E, A, T, H, C, P, N$. The carrier has no coordinate for Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.LocatedCompletionSealUp.LocatedCompletionSealTasteGate single carrier alignment`

Theorem 1805.2 (Located completion seal NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *carrier*

$$L = (D, S, R, Q, E, A, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompletionSeal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the dyadic ledger, finite stream windows, regular rational source, regular located Cauchy source, located tail envelope, Bishop regular completion boundary, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *located-source routing, requiring every completion-facing request to read D, S, R before it may read the regular located source row Q;*
3. *tail-envelope compatibility, requiring E to consume the same dyadic, stream, regular rational, and located-source rows as the request that enters the packet;*
4. *completion-boundary compatibility, requiring A to expose only the displayed regular Cauchy completion boundary over D, S, R and not a selected point of a completed space;*
5. *Real handoff, requiring T to be downstream of D, S, R, Q, E, A for every* $\mathsf{Real}^{\uparrow}$ *consumer named by P;*
6. *ledger non-escape, requiring H, C, P, N to preserve and name exactly this finite route and to export no Real completeness, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCompletionSealUp.LocatedCompletionSealNameCert obligation rows exact`

Proof. Carrier exposure is projection from Definition 1805.1. The displayed packet contains the dyadic row D, stream windows S, regular rational row R, located source row Q, located tail envelope E, Bishop regular completion boundary A, terminal Real seal T, and the structural rows H, C, P, N.

The located-source route is fixed by the row order. A precision request first opens the dyadic tolerance ledger, then the finite stream windows, then the regular rational observations, and only then the regular located Cauchy source.

The tail envelope and completion boundary are compatible because they read the same preceding rows. The row E packages the finite located tail route over D, S, R, Q, while the row A packages the regular Cauchy completion-facing boundary over D, S, R.

The Real handoff is terminal. The row T is exposed only after the located source, tail envelope, and completion boundary have all been displayed.

Finally, the rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded objects appears as a coordinate of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1805.3 (Located completion seal Real handoff)**.** *Let $L = (D, S, R, Q, E, A, T, H, C, P, N)$ be an accepted* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer read factors through the finite route*

$$D \longmapsto S \longmapsto R \longmapsto Q \longmapsto E \longmapsto A \longmapsto T \longmapsto H, C, P, N.$$

Consequently the consumer receives a dyadic tolerance ledger, finite stream windows, regular rational observations, regular located Cauchy source data, a located tail envelope, a Bishop regular Cauchy completion boundary, a terminal Real seal, transport, replay, provenance, and local naming. The handoff cannot use Real completeness, quotient stream equality, a selected limit, countable choice, trichotomy, or host real equality.

Proof. Project the accepted carrier by Definition 1805.1. The only Real-facing endpoint is T, and the carrier places T after the dyadic, stream, regular rational, located-source, tail-envelope, and completion-boundary rows.

The analytic route starts with D, S, R. These rows fix the tolerance ledger, finite observation windows, and regular rational readback before any located completion claim is visible.

The located completion seal is then read through Q, E, A. The row Q binds the source as a regular located Cauchy real, E exposes the finite located tail envelope, and A records the completion-facing boundary without selecting a completed point.

The structural rows preserve the same order. Componentwise hsame transport and displayed Cont replay cannot introduce a second route to T, and the provenance and local naming rows add no excluded coordinate. Hence every $\mathsf{Real}^{\uparrow}$ read factors through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionSeal}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCompletionSeal}^{\uparrow}}$, seed)

Scope closed: Definition 1805.1, Definition 1805.2, and Definition 1805.3 seed the finite located-completion seal through which DyadicRatCore, StreamName, RegSeqRat, RegularLocatedCauchyReal, LocatedRealTailEnvelope, BishopRegularCauchyCompletion, and Real rows are read in one displayed route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCompletionSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, located-source routing, tail-envelope compatibility, completion-boundary compatibility, Real handoff, and ledger non-escape for BEDC.Derived.LocatedCompletionSealUp.

Constructive story: A $\mathsf{LocatedCompletionSeal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, a RegSeqRat source row, a RegularLocatedCauchyReal source row, a LocatedRealTailEnvelope row, a BishopRegularCauchyCompletion boundary row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1806

A Concrete Naming Certificate for BishopRealUniformCompletion$^{\uparrow}$

Following Bishop and Bridges and the Bourbaki–Kelley uniform-completion viewpoint, the BishopRealUniformComp packet records the finite route by which the Real uniformity is consumed by Cauchy completion without turning that consumption into a Real-completeness theorem. It reads the Real-induced entourage surface of chapter 1624, the reflective completion inclusion of chapter 1793, the triangle route of chapter 1795, and the terminal real-name seal of chapter 38. The chapter names the L10 bridge between uniform Real reads and completion-facing Cauchy routes while keeping quotient streams, selected limits, host real equality, countable choice, ambient completeness, and bridge credentials outside the packet.

Definition 1806.1 (BishopRealUniformCompletion carrier)**.** *A* BishopRealUniformCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, U, J, T, S, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* Real$^{\uparrow}$ *endpoint and seal row, exposing only named Real observations and the local seal consumed by the bridge.*
2. *U is the* RealUniformStructure$^{\uparrow}$ *entourage row, generated from Real metric-ball requests before any completion-facing read is available.*
3. *J is the reflective completion inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$ *as the public completion route for the Real-uniform data.*
4. *T is the* CauchyCompletionTriangleIdentity$^{\uparrow}$ *route row, recording that both completion composites reuse the same finite Real-uniform and reflective-inclusion surface.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected by Real and completion consumers.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used by the Real-uniform entourage request and by the completion-side route comparison.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the stream windows and dyadic tolerances.*
8. *H records componentwise* hsame *transport for the Real seal, Real-uniform entourage, reflective completion inclusion, triangle route, stream, dyadic, regular-rational, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Real endpoint request through the Real-uniform entourage row, reflective inclusion, triangle route, stream window, dyadic tolerance, regular rational readback, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealUniformCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, U, J, T, S, D, Q, H, C, P, N$. The carrier contains no quotient stream, selected limit, host real equality, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, metric-completeness datum, or bridge credential.*

Lean4 ✓ `BEDC.Derived.BishopRealUniformCompletionUp.BishopRealUniformCompletionTasteGate single carrier alignment`

Theorem 1806.2 (BishopRealUniformCompletion NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *carrier $B = (R, U, J, T, S, D, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealUniformComplet}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Real endpoint and seal row, Real-uniform entourage row, reflective completion inclusion row, triangle route row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, transport, replay, provenance, and local naming row;*

2. *uniform entourage handoff, requiring every completion-facing Real read to consume U after projecting R and before inspecting J or T;*

3. *reflective route exactness, requiring every completion inclusion read to pass through J over the same R, U, S, D, Q coordinates;*

4. *triangle route exactness, requiring both completion composites to use T after the reflective route has fixed the finite inclusion surface;*

5. *L10 handoff discipline, requiring StreamName, DyadicRatCore, and RegSeqRat observations to be read only through S, D, Q along the displayed Real-uniform and completion route;*

6. *non-escape, requiring H, C, P, N to replay and name no row from which quotient streams, selected limits, host real equality, countable choice, ambient Real completeness, metric-completeness data, or bridge credentials can be exported.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1806.1. The packet displays the Real seal, Real-uniform entourage, reflective inclusion, triangle route, finite-window, tolerance, readback, transport, replay, provenance, and local naming rows.

The uniform entourage handoff is ordered by the carrier. A Real-facing request first projects R, then reads the Real-induced entourage row U, and only then may continue toward completion through J or T.

Reflective exactness is the completion-side projection. The row J is the only displayed inclusion surface, so every completion inclusion read is forced to reuse the same R, U, S, D, Q coordinates rather than selecting a host completion object.

Triangle exactness is then downstream of J. The row T records both completion composites over the fixed inclusion route, and it has no separate quotient, representative, or host-equality coordinate.

Finally, the L10 handoff and non-escape clauses are exhausted by S, D, Q, H, C, P, N. These rows transport, replay, source, and name the displayed route, and none of them contains quotient streams, selected limits, host real equality, countable choice, ambient Real completeness, metric-completeness data, or bridge credentials. □

Theorem 1806.3 (BishopRealUniformCompletion L10 handoff)**.** *Every completion-facing* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *carrier factors through the finite rows*

$$R \longmapsto U \longmapsto J \longmapsto T \longmapsto S, D, Q \longmapsto H, C, P, N.$$

Consequently the packet exports no quotient stream, selected limit, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or bridge credential.*

Proof. Start with a completion-facing Real read and project the carrier rows. The Real endpoint and seal data are visible only through R, and the uniformity consumed by the read is visible only through U.

The reflective completion route then passes through J. This fixes the finite inclusion surface before the triangle route is inspected, so the read cannot branch into a host completion carrier or selected limit.

Project T to record the two completion composites over the same route. The finite observation rows S, D, Q supply StreamName windows, dyadic tolerances, and regular rational readback for that route and no other one.

The structural rows H, C, P, N close the handoff by transport, replay, provenance, and local naming. Since these rows exhaust the packet, there is no coordinate from which a quotient stream, selected limit, host real equality, countable choice, ambient Real completeness, metric-completeness datum, or bridge credential could be exported. □

Theorem 1806.4 (BishopRealUniformCompletion diagonal seal ledger)**.** *For every accepted* $\mathsf{BishopRealUniformCor}$ *carrier* $B = (R, U, J, T, S, D, Q, H, C, P, N)$*, any diagonal or completion-facing seal read factors through*

$$R \longmapsto U \longmapsto J \longmapsto T \longmapsto S, D, Q \longmapsto H, C, P, N.$$

Thus the diagonal route consumes the $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage, reflective completion inclusion, triangle route,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any* $\mathsf{Real}^{\uparrow}$ *seal is public.*

Proof. Project the accepted carrier and apply Definition 1806.2. The obligation surface orders the Real endpoint row R, the Real-uniform entourage row U, the reflective route J, the triangle route T, and the L10 handoff through S, D, Q.

Apply Definition 1806.3. It identifies the completion-facing read as the finite route

$$R \longmapsto U \longmapsto J \longmapsto T \longmapsto S, D, Q \longmapsto H, C, P, N.$$

The diagonal seal ledger is this same route read from the Real-uniform side toward the completion-facing seal boundary.

The carrier has no quotient stream, selected limit, host real equality, countable-choice selector, ambient completeness theorem, metric-completeness datum, or bridge credential coordinate. Therefore the diagonal or completion-facing seal cannot leave the displayed RealUniformStructure, reflective completion, triangle, StreamName, DyadicRatCore, and RegSeqRat rows before the local seal is consumed. □

Theorem 1806.5 (BishopRealUniformCompletion obligation rows)**.** *Let $B = (R, U, J, T, S, D, Q, H, C, P, N)$ be an accepted* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *carrier. The obligation surface exposed by B is exactly the six rows named in Definition 1806.2: carrier admission; uniform-entourage handoff; reflective-route exactness; triangle-route exactness; L10 handoff through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$*; and non-escape from quotient streams, selected limits, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, and bridge credentials.*

Proof. Project the accepted carrier. The rows $R, U, J, T, S, D, Q, H, C, P, N$ are the only displayed coordinates, so any local obligation must be read through the Real endpoint, Real-uniform entourage, reflective inclusion, triangle route, finite stream window, dyadic tolerance, regular rational readback, transport, replay, provenance, and local naming rows.

Apply Definition 1806.2. It lists exactly carrier admission, uniform-entourage handoff, reflective-route exactness, triangle-route exactness, L10 handoff discipline, and non-escape, and it states that no local obligation is stored outside those rows.

Finally apply Definition 1806.3. The handoff identifies the route order

$$R \longmapsto U \longmapsto J \longmapsto T \longmapsto S, D, Q \longmapsto H, C, P, N,$$

so the L10 row is precisely the StreamName, DyadicRatCore, and RegSeqRat readback surface downstream of the Real-uniform and completion rows. □

Theorem 1806.6 (BishopRealUniformCompletion density handoff)**.** *Every accepted* $\mathsf{BishopRealUniformCompletion}$ *packet whose* $\mathsf{Real}^{\uparrow}$ *endpoint row is read through a dyadic or regular-rational dense approximation route exposes that route through*

$$R, U, J, T, S, D, Q, H, C, P, N$$

before the completion seal is public. The packet therefore records the density handoff needed by completion consumers, but it does not claim ambient $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit, a quotient stream, host equality, countable choice, metric-completeness data, or a bridge credential.*

Proof. Start from the accepted carrier of Definition 1806.1. The endpoint row R is not read in isolation; the Real-uniform entourage row U, reflective completion route J, and triangle route T determine how a completion-facing consumer reaches it.

Apply Definition 1806.2. The obligation rows require L10 handoff through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$, so any dyadic or regular-rational dense approximation route must be displayed through S, D, Q before the endpoint row is public.

Apply Definition 1806.3 and Definition 1806.4. Together they identify the route

$$R \longmapsto U \longmapsto J \longmapsto T \longmapsto S, D, Q \longmapsto H, C, P, N.$$

The structural rows then transport, replay, source, and name the displayed density handoff, while the carrier still contains no coordinate for completeness, selected limits, quotient streams, host equality, countable choice, metric-completeness data, or bridge credentials. □

Theorem 1806.7 (BishopRealUniformCompletion uniform-filter roundtrip)**.** *Let $B = (R, U, J, T, S, D, Q, H, C, P,$ be an accepted* $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ *packet. Any completion-facing consumer of B whose read begins from a Cauchy-filter basis must factor first through the* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage row U, then through the reflective completion inclusion J, the triangle route T, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and finally the structural rows H, C, P, N. The roundtrip records the same finite Real-uniform basis named by Definition 1624.6; it exports no quotient filters, selected limits, countable choice, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or bridge credential.*

Proof. Project the carrier of Definition 1806.1. A Cauchy-filter read has no independent row in B; it must be read through the Real-uniform entourage coordinate U together with the Real endpoint row R.

Apply Definition 1624.6. The Cauchy-filter basis available to a Real-uniform consumer is downstream of the displayed RealUniformStructure entourage row, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, transport, replay, provenance, and local naming.

Apply Definition 1806.2 and Definition 1806.3. The completion-side route therefore proceeds from U to the reflective inclusion J, then to the triangle row T, and then through S, D, Q, H, C, P, N before the completion-facing seal is public.

Finally apply Definition 1806.6. The density handoff reuses the same finite route, so the consumer cannot bypass the RealUniformStructure row or obtain quotient filters, selected limits, countable choice, host equality, ambient Real completeness, metric-completeness data, or bridge credentials from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealUniformCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealUniformCompletion}^{\uparrow}$ packet is generated from a Real endpoint and seal row, a RealUniformStructure entourage row, a CauchyCompletionReflectiveSubcategory inclusion row, a CauchyCompletionTriangleIdentity route row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1806.1, Definition 1806.2, and Definition 1806.3 seed the finite Real-uniform completion bridge for L10 consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform entourage handoff, reflective route exactness, triangle route exactness, L10 handoff, and non-escape proofs for BEDC.Derived.BishopRealUniformCompletionUp.

Chapter 1807

A Concrete Naming Certificate for $\mathsf{FiniteBracketingInterval}^{\uparrow}$

Following Bishop and Bridges' constructive interval methods, the $\mathsf{FiniteBracketingInterval}^{\uparrow}$ packet records the finite BEDC surface at which a real approximation route is bracketed by dyadic endpoints before any $\mathsf{Real}^{\uparrow}$ read is exported. It sits on the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the finite density route of chapter 1809. The packet is a bracketing certificate for displayed endpoint windows; it is not a quotient real interval, selected host limit, trichotomy row, ambient completeness principle, or open-cover compactness argument.

Definition 1807.1 (Finite bracketing interval carrier)**.** *A* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, W, R, E, S, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint rows, both named inside the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
2. *D is the dyadic-radius containment row, recording the requested tolerance and the finite comparison budget for $L \leq U$.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation window inspected under the same tolerance request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window handoff row that reads W as regular rational data between the endpoint rows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after L, U, D, W, R have been displayed.*
6. *S is the* $\mathsf{RealSeparability}^{\uparrow}$ *density-support row, recording that a finite rational candidate can be inspected inside the same bracket when a density consumer requests one.*
7. *H stores componentwise* hsame *transport for endpoint, radius, window, readback, seal, density-support, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a bracketing request through L, U, D, W, R, E, S.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteBracketingInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteBracketingInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient stream equality, selected limits, countable choice, ambient completeness, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.FiniteBracketingIntervalUp.FiniteBracketingIntervalTasteGate single carrier alignment`

Theorem 1807.2 (Finite bracketing endpoint order)**.** *For every accepted* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *carrier*

$$I = (L, U, D, W, R, E, S, H, C, P, N),$$

the endpoint order read is exhausted by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ *rows* L, U, D*. A consumer may read the bracket only after the same finite comparison budget has recorded* $L \leq U$*, and no host order on* $\mathsf{Real}^{\uparrow}$*, trichotomy row, quotient stream equality, selected midpoint, or ambient completeness datum can replace that dyadic endpoint ledger.*

Proof. Project the carrier of Definition 1807.1. The only endpoint rows are L and U, and the only comparison budget attached to them is D.

Read these rows through the dyadic tolerance surface of chapter 291. The comparison is therefore a finite ledger entry over named dyadic endpoints, not an external order assertion about host reals.

The remaining rows W, R, E, S, H, C, P, N preserve, replay, source, and name the same endpoint packet. Since none of them contains host real equality, quotient stream equality, selected midpoint data, trichotomy, or ambient completeness, the endpoint order is exhausted by L, U, D. □

Theorem 1807.3 (Finite bracketing dyadic-radius containment)**.** *Every accepted* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *containment read is witnessed by the same finite dyadic-radius row* D *together with the endpoint rows* L, U *and the opened* $\mathsf{StreamName}^{\uparrow}$ *window* W*. The read states that the inspected window is contained in the displayed bracket at the requested dyadic radius before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff or* $\mathsf{Real}^{\uparrow}$ *seal is available.*

Proof. Begin with an accepted containment read. By Definition 1807.1, the tolerance request has only one dyadic-radius row D, and it is tied to the endpoint rows L, U.

The finite window stage is then W. The StreamName surface of chapter 126 supplies exactly the inspected observations at the requested radius, so containment is checked before regular rational readback.

Only after L, U, D, W are displayed may the route pass to R and then E. Thus the containment read cannot be supplied by a quotient stream, selected host limit, countable choice, ambient completeness, or a detached $\mathsf{Real}^{\uparrow}$ equality. □

Theorem 1807.4 (Finite bracketing RegSeqRat handoff)**.** *Every finite-window consumer of an accepted* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *packet factors through the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads L, U, D*, then the finite window* W*, then the regular rational handoff* R*, and only then the terminal seal* E*.*

Proof. Project the carrier. The dyadic endpoint and radius rows L, U, D precede the finite observation row W, and W precedes the readback row R.

Use the RegSeqRat chapter chapter 301. Its public handoff reads finite StreamName windows as regular rational data before any Real-facing seal can consume them.

The terminal row is E, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The route has no coordinate for quotient stream equality, selected limits, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, so the displayed handoff is the whole public consumer surface. □

Theorem 1807.5 (Finite bracketing RealUp seal non-escape)**.** *An accepted* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *packet exports a* $\mathsf{Real}^{\uparrow}$ *read only through the terminal seal row E after the endpoint order, dyadic-radius containment, finite-window read, RegSeqRat handoff, and RealSeparability density-support rows have already been displayed. No consumer can escape from the bracket into host real equality, selected limits, quotient streams, countable choice, ambient completeness, open-cover compactness, or a detached dense subset.*

Proof. Apply the preceding endpoint, containment, and handoff theorems. They force a Real-facing read to pass through L, U, D, W, R before E is available.

The density-support row S is read through chapter 1809. It supplies only the finite rational-window support for a density request inside the same bracket, not an externally chosen dense point.

Finally inspect the structural rows H, C, P, N. They transport, replay, source, and name the displayed packet, and the carrier contains no coordinate for host real equality, selected limits, quotient streams, countable choice, ambient completeness, open-cover compactness, or detached dense subsets. Therefore the terminal seal row is a non-escape boundary. □

Theorem 1807.6 (Finite bracketing interval sibling route)**.** *Every bracket accepted by the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *carrier reads the located interval endpoint surface, dyadic endpoint ledger,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{RealSeparability}^{\uparrow}$ *density-support row in that order. The route consumes chapter 1709 for the endpoint surface and chapter 1809 for the finite density support before the local structural rows H, C, P, N package the bracket.*

Proof. Project the carrier. The endpoint rows L, U and dyadic-radius ledger D provide the finite endpoint surface, and the local introduction ties that surface to the located real interval route.

Use the finite-window RegSeqRat handoff theorem. It forces the consumer through the $\mathsf{StreamName}^{\uparrow}$ window W and $\mathsf{RegSeqRat}^{\uparrow}$ readback R before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be read.

Use the RealUp seal non-escape theorem. It requires the density-support row S to remain the $\mathsf{RealSeparability}^{\uparrow}$ row cited from chapter 1809, not a detached dense subset or chosen point.

Therefore the sibling route is exactly the located endpoint surface, dyadic endpoint ledger, finite window, regular rational handoff, real seal, and separability support, followed only by transport, replay, provenance, and local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBracketingInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteBracketingInterval}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, a dyadic-radius containment ledger, a StreamName finite observation window, a RegSeqRat handoff row, a RealUp terminal seal, a RealSeparability density-support row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBracketingInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1807.1, Definition 1807.2, Definition 1807.3, Definition 1807.4, and Definition 1807.5 seed the finite bracketing route over dyadic endpoints, StreamName windows, RegSeqRat readback, RealUp seal, and RealSeparability density support.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBracketingInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, countable choice, ambient completeness, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint order, dyadic-radius containment, finite-window RegSeqRat handoff, and RealUp seal non-escape for BEDC.Derived.FiniteBracketingIntervalUp.

Chapter 1808

A Concrete Naming Certificate for $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$

Following Bishop and Bridges' constructive interval compactness and finite-net methods, the $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ packet names the finite BEDC surface at which a located closed interval admits a displayed mesh row before any completion-facing consumer reads a cover claim. It sits after the located endpoint surface of chapter 1709, the bracketing packet of chapter 1807, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet records a finite net for a displayed located interval; it is not a general Heine-Borel theorem, an open-cover compactness principle, a quotient interval, a selected host limit, or an ambient completeness coordinate.

Definition 1808.1 (Located closed interval finite-net carrier)**.** *A* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, M, N, D, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, displaying the lower and upper endpoint names together with the located interval boundary that consumers may inspect.*
2. *M is the mesh-scale row, a finite dyadic tolerance request read through the same endpoint boundary.*
3. *N is the finite net row, listing the displayed mesh points admitted between the located endpoints at scale M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *coverage ledger, recording endpoint comparisons, mesh-step arithmetic, and the finite containment checks for the net row.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row opened only for the endpoint and mesh requests already displayed by I, M, D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets the opened windows as regular rational data along the same finite mesh route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the located interval, mesh, net, dyadic ledger, windows, and readback rows have been displayed.*

8. *H records componentwise* hsame *transport for the interval source, mesh scale, net row, coverage ledger, window row, readback row, seal row, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from a finite-net request through I, M, N, D, W, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{FiniteBracketingInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, general Heine-Borel compactness, host interval equality, quotient representatives, selected real limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedClosedIntervalFiniteNetUp.LocatedClosedIntervalFiniteNetTasteGate`
`single carrier alignment`

Theorem 1808.2 (Located closed interval finite-net coverage)**.** *For every accepted* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *carrier*

$$F = (I, M, N, D, W, R, E, H, C, P, L),$$

each point read by the local finite-net consumer is covered only through the displayed route

$$\mathsf{LocatedRealInterval}^{\uparrow} \longrightarrow \mathsf{FiniteBracketingInterval}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}.$$

Concretely, the consumer first reads the located interval source I, then the mesh scale M, then the finite net row N, then the dyadic coverage ledger D, and only then opens the finite windows W and readback row R.

Proof. Project the accepted carrier. The source interval row is I, and the only mesh data are the scale M, the net row N, and the dyadic coverage ledger D.

Use chapter 1709 for the endpoint surface and chapter 1807 for the finite bracketing route. The bracket read is therefore a displayed interval-and-endpoint computation before any stream window is opened.

The remaining coverage work is finite. The dyadic ledger D records the mesh-step arithmetic and containment checks, while W and R expose only the StreamName windows and RegSeqRat readback attached to that displayed ledger.

The excluded compactness principles have no coordinate in F. Thus the coverage read is a finite-net route over the listed rows, not an open-cover or ambient-completeness argument. □

Theorem 1808.3 (Located closed interval finite-net Real seal non-escape)**.** *An accepted* LocatedClosedIntervalFinite *packet exports a* $\mathsf{Real}^{\uparrow}$ *read only through the terminal seal row E after I, M, N, D, W, R have already been displayed. No consumer can escape from the finite net into general Heine-Borel compactness, open-cover compactness, host interval equality, quotient representatives, selected real limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply the finite-net coverage theorem above. It forces a consumer read through the located source, mesh scale, finite net row, dyadic coverage ledger, finite windows, and regular rational readback before the terminal seal is available.

The seal row E is downstream of that finite route. It records the $\mathsf{Real}^{\uparrow}$ boundary attached to the displayed interval-net computation and does not supply an independent completed-real object.

Finally inspect the structural rows H, C, P, L. They transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for the excluded principles or host objects, the terminal seal is a non-escape boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ packet is generated from a located closed interval source, a dyadic mesh scale, a finite net row, a DyadicRatCore coverage ledger, StreamName finite windows, RegSeqRat readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1808.1, Definition 1808.2, and Definition 1808.3 seed the finite L10 interval-net route over located endpoints, bracketing rows, dyadic coverage, finite windows, regular rational readback, and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.LocatedClosedIntervalFiniteNetUp.LocatedClosedIntervalFiniteNetTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close general Heine-Borel compactness, open-cover compactness, host interval equality, quotient representatives, selected real limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedClosedIntervalFiniteNetUp.

The $\mathsf{RealSeparability}^{\uparrow}$ hub routes rational density through chapter 1820, chapter 291, chapter 126, chapter 301, and chapter 38. Completion-facing density exports pass through chapter 1453 before any metric-completion consumer reads a countable dense surface.

The $\mathsf{RealSeparability}^{\uparrow}$ hub consumes the Cauchy-equivalence setoid route of chapter 1820 before dense real-name reads are public.

The dense schedule is read through $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rather than through quotient stream equality. The setoid route compares regular names by displayed finite classifier rows, the regular-rational route supplies the readback window, and the Real seal is consumed only after those rows are visible to the separability carrier.

Chapter 1809

A Concrete Naming Certificate for $\mathsf{RealSeparability}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of the real line by rational approximants, and the Bourbaki description of separability by a countable dense subset, the $\mathsf{RealSeparability}^{\uparrow}$ packet records the finite BEDC surface at which rational density is available to $\mathsf{Real}^{\uparrow}$ consumers. It sits on the rational naming row of chapter 37, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the countable density boundary of chapter 1566. Thus $\mathsf{RealSeparability}^{\uparrow}$ consumes the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ source surfaces of chapter 126 and chapter 301 before the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names rational dense enumeration as finite Real data; it does not introduce host real equality, choice over balls, quotient streams, or ambient completeness.

Definition 1809.1 (Real separability carrier)**.** *A* $\mathsf{RealSeparability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *enumeration row. It exposes rational candidates by finite index windows and records the order in which a density request may inspect them.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same request window, supplying the dyadic precision against which the rational candidates are tested.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row opened by D, so the consumer reads only the finite name segment required by the requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, transporting the inspected rational candidate and window data into the regular rational route used by the real-name interface.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It records that the dense rational window is consumed as Real data only after Q, D, W, R have already been displayed.*
6. *T is the dense-window exactness ledger. It records that every density read first cites an index in Q, then a dyadic tolerance in D, then a finite window in W, then the regular readback in R, and only then the terminal seal E.*
7. *H stores componentwise* hsame *transport for the rational enumeration, tolerance, window, readback, seal, exactness ledger, replay, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from a Real-facing density request through Q, D, W, R, T and into the terminal seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, a choice-selected dense point, quotient stream equality, ambient completeness, trichotomy, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.RealUp.RealSeparabilityCarrier`

The sibling lattice route is explicit: $\mathsf{RealSeparability}^{\uparrow}$ consumes chapter 1484 for the dyadic tolerance witness and chapter 1409 for the rational interval witness before $\mathsf{CountableDenseSubset}^{\uparrow}$ names the dense source. This places the separability packet downstream of the dyadic and rational density witnesses rather than treating density as a detached Real-side fact.

The same dyadic tolerance witness and rational interval witness feed the finite precision requests of chapter 1423: a modulus consumer may read the dyadic tolerance row D as its requested precision and the rational interval witness carried by Q, T as the finite interval evidence before opening the StreamName and RegSeqRat rows.

Theorem 1809.2 (Real separability NameCert obligations)**.** *For every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational enumeration, dyadic tolerance, StreamName window, RegSeqRat readback, Real seal, dense-window exactness ledger, transport, replay, provenance, and local naming rows;*
2. *rational enumeration exposure, requiring every density read to cite the displayed* $\mathsf{Rat}^{\uparrow}$ *index row Q before any tolerance, window, readback, or Real seal is consumed;*
3. *dense-window exactness, requiring T to bind the same request through Q, D, W so that the inspected rational candidate is tested only inside the opened finite window;*
4. *RegSeqRat handoff, requiring the finite rational data exposed by Q, D, W to pass through R before it can be read by the Real-facing boundary;*
5. *RealUp seal non-escape, requiring E, P, N to export no host real equality, choice-selected dense point, quotient stream equality, ambient completeness, trichotomy, Lean verification, or bridge checking.*

No separability obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1809.1. The tuple lists exactly the rational enumeration, tolerance, finite observation window, regular rational readback, Real seal, exactness ledger, and structural rows.

Rational enumeration exposure is forced by the placement of Q. A density request cannot inspect a candidate until the displayed $\mathsf{Rat}^{\uparrow}$ index row has been read as finite packet data.

Dense-window exactness is the row T governed by Q, D, W. It fixes the same rational index, dyadic tolerance, and finite StreamName window before any Real-facing read is admitted.

The RegSeqRat handoff is then the row R. It transports the inspected rational candidate and window data into the regular rational route, so the Real boundary receives a regular readback rather than a host real point.

Finally E, H, C, P, N seal, transport, replay, source, and name only the displayed route. Since the carrier contains no coordinate for host equality, selected dense points, quotient streams, ambient completeness, trichotomy, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealUp.RealSeparabilityNameCertObligations`

Theorem 1809.3 (Real separability rational dense handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *density consumer admitted by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before any separability read is exported. In particular, the consumer reads the same displayed rational enumeration row Q, dyadic tolerance row D, finite window row W, regular rational readback row R, and terminal seal row E before it can consume the local $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *surface.*

Proof. Begin with a density consumer admitted by the carrier of Definition 1809.1. Projection exposes the finite rows $Q, D, W, R, E, T, H, C, P, N$.

The first analytic row is Q, the rational enumeration. The consumer therefore starts with a displayed $\mathsf{Rat}^{\uparrow}$ index window, not with a choice-selected dense point or an ambient real value.

The tolerance and observation rows follow. The dyadic row D supplies the requested precision, and the StreamName row W opens exactly the finite window read at that precision.

The handoff to R places the inspected rational candidate into the regular rational route. Only after that readback does the terminal seal E present the datum as $\mathsf{Real}^{\uparrow}$.

The exactness ledger T, componentwise transport H, replay C, provenance P, and local naming row N preserve this order. Hence every exported density read factors through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before separability is consumed. □

Theorem 1809.4 (Real separability Cauchy-equivalence forward consumption)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that compares two regular rational windows must consume the finite classifier route of chapter 1820 before the rational dense handoff is exported. Thus an accepted separability carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

receives a finite $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison surface rather than quotient stream equality, selected representatives, host real equality, countable-choice selectors, or ambient completeness.*

Proof. Project the carrier of Definition 1809.1. A density read has only the rational enumeration row Q, dyadic tolerance row D, finite window row W, regular readback row R, terminal seal row E, exactness ledger T, transport H, replay C, provenance P, and local name N.

Apply Definition 1809.3. The rational dense handoff is exported only after the read has passed through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

When two regular rational windows are compared, read the classifier route of chapter 1820 before that export. The classifier supplies the finite Cauchy and RegSeqRat comparison surface, so the separability packet cannot replace it with quotient stream equality, selected representatives, host real equality, countable-choice selectors, or ambient completeness. □

Theorem 1809.5 (Real separability L10 density certificate)**.** *Every dense dyadic schedule exported by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it can serve a countable-density consumer. The route uses the Cauchy-equivalence classifier of chapter 1820 to compare regular rational windows, and it exports no quotient stream equality, selected dense point, host real equality, countable-choice selector, or ambient completeness witness.*

Proof. Project the carrier of Definition 1809.1. A dense dyadic schedule can read only the displayed rational enumeration row Q, dyadic tolerance row D, finite window row W, regular readback row R, terminal seal row E, exactness ledger T, transport H, replay C, provenance P, and local name N.

Apply Definition 1809.3. The exported density read must pass through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any separability consumer can use it.

Next apply Definition 1809.4. When the schedule compares two regular rational windows, the comparison is routed through $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ and not through quotient stream equality or a selected representative.

Finally route the displayed schedule to the countable-density boundary by Definition 1809.12. The same rows $Q, D, W, R, E, T, H, C, P, N$ carry the dyadic schedule to the separability export, so no host equality, selected dense point, countable-choice selector, or ambient completeness witness is introduced. □

Theorem 1809.6 (Real separability dyadic-rational density handoff)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density consumer whose tolerance row is served by an accepted* $\mathsf{DyadicDensityReal}^{\uparrow}$ *witness and whose interval witness is served by* $\mathsf{RationalDensityReal}^{\uparrow}$ *factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{DyadicDensityReal}^{\uparrow} \longrightarrow \mathsf{RationalDensityReal}^{\uparrow} \longrightarrow \mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow$$

The route uses only the displayed carrier rows $Q, D, W, R, E, T, H, C, P, N$ together with the dyadic-density surface of chapter 1484 and the rational-density surface of chapter 1409. It exports no host real equality, choice-selected dense point, quotient stream, trichotomy, or ambient completeness witness.

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier as

$$S = (Q, D, W, R, E, T, H, C, P, N).$$

The row D is the only tolerance row exposed to the density consumer, while Q, W, R, E are the rational enumeration, finite window, regular readback, and terminal Real seal rows consumed by the existing rational dense handoff.

Read the tolerance request through chapter 1484. An accepted $\mathsf{DyadicDensityReal}^{\uparrow}$ witness names the dyadic source inside $\mathsf{DyadicRatCore}^{\uparrow}$ before the Real-facing window and readback rows are inspected, so the request supplies a displayed dyadic witness rather than a selected host real point.

Next read the interval witness through chapter 1409. The accepted $\mathsf{RationalDensityReal}^{\uparrow}$ packet turns the displayed interval and dyadic tolerance into a visible $\mathsf{Rat}^{\uparrow}$ witness row, with the rational witness routed through its own finite window and ledger rows before any Real seal is available.

Compose those two witness routes with Definition 1809.3. The consumer first sees the $\mathsf{DyadicRatCore}^{\uparrow}$ source, then the dyadic-density witness, then the rational-density witness, and only then the rational enumeration, StreamName window, RegSeqRat readback, and Real seal recorded by Q, W, R, E.

The remaining rows T, H, C, P, N preserve the same order as exactness ledger, componentwise transport, displayed replay, provenance, and local naming. Since none of the three packets contains a coordinate for host real equality, a choice-selected dense point, quotient stream equality, trichotomy, or ambient completeness, the composed handoff cannot export any such data. □

Theorem 1809.7 (Real separability dyadic witness family)**.** *An accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density read determines a finite family of rational witnesses indexed only by the displayed dyadic tolerance row* D *and finite* $\mathsf{StreamName}^{\uparrow}$ *window* W *of its carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N).$$

Every witness in that family is routed through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before separability is exported. The family contains no choice-selected dense point, host real equality, quotient stream, trichotomy row, or ambient completeness witness.*

Proof. Project the carrier of Definition 1809.1. The only rows that can index a finite density family are the dyadic tolerance row D and the finite StreamName window row W, with the rational enumeration row Q supplying the visible rational candidates.

Apply Definition 1809.3. Each candidate read starts in $\mathsf{Rat}^{\uparrow}$, passes through the dyadic tolerance and finite window rows, and reaches $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Next apply Definition 1809.6. The dyadic-density and rational-density witnesses attach the family to the same displayed route, so the candidate family is indexed by D, W rather than by a choice function over Real balls.

Finally apply Definition 1809.12. The $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary receives exactly the finite rational family already routed through $Q, D, W, R, E, T, H, C, P, N$. Since the carrier and handoff theorems expose no host equality, quotient stream, trichotomy, selected dense point, or ambient completeness coordinate, the witness family exports none of them. □

Theorem 1809.8 (Real separability dyadic countable-dense scope)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense read exporting a countable dense witness must display the dyadic tolerance request, rational interval locator,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the countable-dense consumer reads the witness. For an accepted carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the countable-dense route is scoped by the same rational enumeration row Q*, dyadic tolerance row* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, regular rational readback* R*, terminal seal* E*, exactness ledger* T*, transport* H*, replay* C*, provenance* P*, and local naming row* N*. It exports no host real equality, quotient stream, countable-choice selector, selected dense point, or ambient completeness witness.*

Proof. Project the carrier rows from Definition 1809.1. A countable-dense export has no data beyond the rational enumeration Q, dyadic tolerance D, finite window W, readback R, Real seal E, exactness ledger T, transport H, replay C, provenance P, and local naming row N.

Use Definition 1809.7. The exported witness family is indexed by the displayed dyadic tolerance request D and finite StreamName window W, with the rational enumeration row Q supplying the visible rational candidates. The rational interval locator is therefore consumed as part of the finite rational-density handoff rather than as a choice over Real balls.

Next use Definition 1809.13. Every comparison of regular-rational names passes through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before the dense rational row is exported, so the readback R is classified by displayed Cauchy-setoid data and not by quotient stream equality.

Finally apply Definition 1809.12. The $\mathsf{CountableDenseSubset}^{\uparrow}$ consumer receives the witness only after the route through $Q, D, W, R, E, T, H, C, P, N$ has reached the terminal $\mathsf{Real}^{\uparrow}$ seal. Since the carrier and cited routes have no coordinate for host real equality, quotient streams, countable-choice selectors, selected dense points, or ambient completeness, the countable-dense export cannot expose any of them. □

Theorem 1809.9 (Real separability RegSeqRat dense-window induction)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window read for a carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

can be consumed by induction over the displayed finite $\mathsf{StreamName}^{\uparrow}$ *window* W *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *are opened. The induction exposes only the rational enumeration row* Q*, dyadic tolerance row* D*, finite window* W*, dense-window exactness ledger* T*, transport* H*, replay* C*, provenance* P*, and local naming row* N *until the readback and seal rows are explicitly displayed. It exports no selected dense point, quotient stream, trichotomy branch, ambient completeness witness, host real equality, or countable-choice selector.*

Proof. Project the carrier of Definition 1809.1. Before R and E are opened, the dense-window read has exactly the rational enumeration, dyadic tolerance, finite StreamName window, exactness ledger, transport, replay, provenance, and local naming rows available.

Apply the rational dense handoff Definition 1809.3. It orders every density consumer through Q, D, W before the RegSeqRat readback and Real seal can be used.

Use the Cauchy-equivalence forward consumption theorem Definition 1809.4. Any comparison of regular rational windows is deferred to the displayed classifier route, so the induction over W cannot become quotient stream equality or host real equality.

Finally apply the dyadic witness family theorem Definition 1809.7. The rational witnesses form a finite family indexed by D and W, hence the dense-window read is consumed by finite-window induction before R and E are opened. The refused principles are absent from the carrier inventory. □

Theorem 1809.10 (Real separability finite-density selector)**.** *Every finite density selector exported by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

factors through the displayed route

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The selector first reads the rational enumeration row Q*, then the dyadic tolerance row* D*, then the finite StreamName window* W*, then the RegSeqRat readback row* R*, and only then the terminal Real seal* E*. It cannot expose a chosen dense point, quotient stream, host real equality, ambient completeness witness, selected nearest point, or detached dense subset.*

Proof. Project the accepted carrier. The rows available to a finite density selector are exactly the rational enumeration Q, dyadic tolerance D, finite window W, readback R, Real seal E, exactness ledger T, transport H, replay C, provenance P, and local naming row N.

Apply Definition 1809.3. It forces the selector to begin at the $\mathsf{Rat}^{\uparrow}$ enumeration row and to pass through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any separability boundary is exported.

Next apply Definition 1809.6. The dyadic tolerance and rational-density witness surfaces attach the selector to the same finite request, so the selected datum is displayed as a routed rational witness rather than as a point chosen from an ambient Real ball.

Finally apply Definition 1809.7. The admissible witnesses form a finite family indexed by D and W, and each family member is consumed through Q, R, E, T, H, C, P, N. Since no listed row contains a quotient stream, host equality, ambient completeness witness, selected nearest point, or detached dense subset, the selector cannot export any such object. □

Theorem 1809.11 (Real separability stream-window stability)**.** *Let*

$$S = (Q, D, W, R, E, T, H, C, P, N)$$

and

$$S' = (Q', D', W', R', E', T', H', C', P', N')$$

be accepted $\mathsf{RealSeparability}^{\uparrow}$ *packets whose rational enumeration, dyadic tolerance, and finite stream-window rows are componentwise transported by displayed* hsame *data*

$$Q\mathsf{hsame}Q', \qquad D\mathsf{hsame}D', \qquad W\mathsf{hsame}W',$$

and whose replay rows display the same Cont *route from a Real-facing density request through those transported rows. Then the finite rational witness family determined by S and the family determined by S' are the same displayed density route. In particular, the RegSeqRat readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff agree as packet data, and no choice-selected dense point, host real equality, quotient stream equality, trichotomy row, or ambient completeness witness is introduced by replacing S with S'.*

Proof. Project the two carriers. The hypotheses identify the only rows that index the finite rational witness family: the rational enumeration rows Q, Q', the dyadic tolerance rows D, D', and the StreamName windows W, W'.

Apply Definition 1809.3 to both packets. Since the displayed Cont replay carries the same request through transported Q, D, W rows, the inspected rational candidates are read along the same finite route before any RegSeqRat or Real seal row is visible.

Apply Definition 1809.6. The dyadic-density and rational-density surfaces attach to the transported tolerance and window rows, so the two packets expose the same dyadic-rational density witness route.

Finally apply Definition 1809.7 and Definition 1809.12. The witness family is exhausted by the displayed route through $Q, D, W, R, E, T, H, C, P, N$ and the transported route through $Q', D', W', R', E', T', H', C', P', N'$, with no additional coordinate available for a selected dense point, host equality, quotient stream, trichotomy, or ambient completeness. □

Theorem 1809.12 (Real separability finite-density exhaustion)**.** *Every admitted* $\mathsf{RealSeparability}^{\uparrow}$ *density read is exhausted by the displayed finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{CountableDenseSubset}^{\uparrow}.$$

Equivalently, the consumer reads only the carrier rows $Q, D, W, R, E, T, H, C, P, N$ together with the finite rational-density and dyadic-density handoff surfaces before the local separability certificate is exported.

The route exports no choice-selected dense point, quotient stream, trichotomy row, ambient completeness witness, host real equality, selected nearest point, or detached dense subset.

Proof. Project the carrier of Definition 1809.1. The rows Q, D, W, R, E give the rational enumeration, dyadic tolerance, StreamName window, RegSeqRat readback, and terminal Real seal, while T, H, C, P, N provide exactness, transport, replay, provenance, and the local naming row.

Apply Definition 1809.3. It forces every density consumer through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any separability read is available.

Next apply Definition 1809.6. The dyadic and rational density witnesses are inserted into the same finite route, so the density source is displayed by the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary and not by an external dense point.

The structural rows T, H, C, P, N preserve that order and add no analytic coordinate. Since the carrier and the two handoff routes contain no choice-selected dense point, quotient stream, trichotomy, ambient completeness, host real equality, selected nearest point, or detached dense subset, the displayed finite route exhausts the admitted density read. □

Theorem 1809.13 (Real separability Cauchy-setoid density stability)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that compares two regular-rational names must pass through the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier of chapter 1820 before the dense rational row is exported. For an accepted separability carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the exported density route is therefore the composite of $Q, D, W, R, E, T, H, C, P, N$ *with the displayed Cauchy-equivalence classifier rows, and it exposes no quotient stream equality, selected representative, countable choice, host real equality, or ambient completeness.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier. The rows $Q, D, W, R, E, T, H, C, P, N$ list the rational enumeration, dyadic tolerance, finite window, regular rational readback, Real seal, dense-window exactness, transport, replay, provenance, and local naming available to the density consumer.

When the consumer compares two regular-rational names, the comparison is not a free equality on streams. It must open the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier of chapter 1820, where the two StreamName windows, RegSeqRat readbacks, dyadic tolerance ledger, setoid-test row, Real seal boundary, transport, replay, provenance, and local naming rows are displayed.

Compose that classifier route with Definition 1809.6. The dense rational row is exported only after the regular readback R, dyadic row D, window row W, and terminal seal E have been compared through the Cauchy-setoid surface.

The structural rows T, H, C, P, N preserve this composite route. Since neither the separability carrier nor the Cauchy-setoid classifier carries quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness, the density read cannot export any of those objects. □

Theorem 1809.14 (Real separability completion-facing density handoff)**.** *Every completion-facing* $\mathsf{RealSeparability}^{\uparrow}$ *density read factors through* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformSpaceCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, *or other completion consumer reads it. For an accepted carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the exported density data is exactly the displayed finite route through the rational enumeration row Q, dyadic tolerance row D, finite stream window W, regular rational readback R, Real seal E, dense-window ledger T, transport H, replay C, provenance P, and local naming row N. It exports no selected nearest point, quotient stream equality, countable-choice selector, host dense subset, or ambient Real completeness.

Proof. Project the carrier of Definition 1809.1. The rows $Q, D, W, R, E, T, H, C, P, N$ are the only data available to a density read before it reaches a completion-facing consumer.

Apply Definition 1809.12. The density read is already exhausted by the route through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CountableDenseSubset}^{\uparrow}$; no selected dense point or external dense subset is introduced.

Next apply Definition 1809.13. Any comparison of regular rational names required by the completion-facing read passes through the displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before the dense row is exported.

Finally hand the same finite route to the completion consumer. Since the carrier rows and cited sibling routes contain no selected nearest point, quotient stream equality, countable-choice selector, host dense subset, or ambient Real completeness coordinate, the completion-facing consumer receives only the displayed density handoff. □

Theorem 1809.15 (Real separability seed-density ledger)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density read exposes the finite L10 seed route*

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{CountableDenseSub}$$

in that order before the read is exported as $\mathsf{RealSeparability}^{\uparrow}$ *data. For an accepted carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the ledger rows $Q, D, W, R, E, T, H, C, P, N$ supply respectively the rational enumeration, dyadic tolerance, finite stream window, regular-rational readback, terminal real seal, dense-window exactness, componentwise transport, replay, provenance, and local naming. The route exports no host $\mathsf{Real}^{\uparrow}$ *equality, selected dense point, quotient stream, countable-choice selector, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier of Definition 1809.1. The rational enumeration row Q, dyadic tolerance row D, finite $\mathsf{StreamName}^{\uparrow}$ window W, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, terminal $\mathsf{Real}^{\uparrow}$ seal E, exactness ledger T, transport H, replay C, provenance P, and local name N are the only rows available to a density consumer.

Apply the rational dense handoff of Definition 1809.3. It places every accepted density read first on the displayed $\mathsf{Rat}^{\uparrow}$ enumeration and then through the dyadic tolerance, finite window, regular readback, and terminal seal rows.

Next apply the dyadic-rational density handoff of Definition 1809.6. The $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance source and rational-density witness are inserted before the separability export, so the route is a finite density route rather than a selected point from an ambient Real ball.

Use Definition 1809.12 to pass the same finite family to the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary. Then use Definition 1809.13 to force every comparison of regular-rational names through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before that boundary is read.

The structural rows T, H, C, P, N preserve the displayed order and add no analytic principle. Since the carrier and the cited routes contain no coordinate for host Real equality, selected dense points, quotient streams, countable choice, trichotomy, or ambient Real completeness, the exported $\mathsf{RealSeparability}^{\uparrow}$ data contains none of them. □

Theorem 1809.16 (Real separability obligation-closure route)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density export factors through the finite L10 seed chain*

$$\mathsf{Rat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{CountableDenseSub}$$

before any public separability data is read. For an accepted carrier

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the obligation surface consists only of the rational dense handoff, dyadic-rational density handoff, finite-density exhaustion, Cauchy-setoid density stability, and seed-density ledger rows. It exports no selected dense point, quotient stream, countable-choice selector, trichotomy branch, host real equality, or ambient completeness witness.

Proof. Project the carrier of Definition 1809.1. The rows $Q, D, W, R, E, T, H, C, P, N$ are the only displayed data available to a density consumer.

Apply the rational dense handoff and the dyadic-rational density handoff. These two routes place every density export behind the $\mathsf{Rat}^{\uparrow}$ enumeration, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before the separability row is visible.

Next apply finite-density exhaustion. It packages the same finite family as the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary rather than as a selected point from an ambient real ball.

Use Cauchy-setoid density stability for every comparison of regular-rational names. The comparison must pass through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before the dense row is exported, so the route never becomes quotient stream equality or host real equality.

Finally apply the seed-density ledger. The structural rows T, H, C, P, N preserve the displayed order and add no analytic coordinate, so the excluded selected dense point, quotient stream, countable-choice, trichotomy, host-equality, and ambient-completeness data cannot be exported by the accepted $\mathsf{RealSeparability}^{\uparrow}$ packet. □

The dense Real route also reads Cauchy-equivalent regular names through the classifier of chapter 1820 before export. Classifier stability keeps equivalent regular names on the same displayed density-consumer route, so the separability packet transports dense witnesses through finite RegSeqRat and Real rows rather than through quotient real equality or ambient completeness.

Chapter 1810

A Concrete Naming Certificate for $\mathsf{RealClamping}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealClamping}^{\uparrow}$ packet records the finite interval-clamp surface by which a displayed real name is forced through a located rational interval before any classical order principle is available. It sits over the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the equality boundary of chapter 1636, the located interval surface of chapter 1709, the finite min/max operation of chapter 1674, and the positive-radius discipline of chapter 1587. The packet gives interval-enclosure and bisection consumers one bounded Real-facing clamp route; it does not import host trichotomy, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1810.1 (RealClamping carrier)**.** *A* $\mathsf{RealClamping}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (X, L, U, D, W, R, E, I, M, A, H, T, P, N)$$

with the following displayed rows:

1. *X is the displayed* $\mathsf{Real}^{\uparrow}$ *source seal whose interval clamp is being named.*
2. *L and U are the displayed lower and upper bound rows, read as* $\mathsf{Real}^{\uparrow}$ *data and constrained by the same located interval request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radius, endpoint slack, and rational comparison budget used by the clamp.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the precision named by D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from X, L, U, W, D before any clamp value is exposed.*
6. *E is the* $\mathsf{RealEquality}^{\uparrow}$ *boundary row used only when the source and an endpoint are indistinguishable at the displayed tolerance.*
7. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *interval row exposing the same lower endpoint, upper endpoint, positive radius, and containment request as finite interval data.*
8. *M is the* $\mathsf{RealMinMax}^{\uparrow}$ *operation row, displaying the constructive min/max route used to form the bounded clamp from X, L, U, R, E, I.*
9. *A is the* $\mathsf{PositiveReal}^{\uparrow}$ *radius row. It records that the interval has a positive displayed separation budget, not a trichotomy witness.*

10. *H records componentwise* hsame *transport for the source seal, endpoints, tolerance ledger, finite window, readback, equality boundary, interval row, min/max row, positive-radius row, replay, provenance, and local naming rows.*

11. *T records displayed* Cont *replay from an interval-clamp request through D, W, R, E, I, M, A and back to the public clamp boundary.*

12. *P is the* Pkg *provenance over* $\mathsf{RealClamping}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealMinMax}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive interval-clamp packet. It has no coordinate for host trichotomy, quotient equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1810.2 (RealClamping NameCert obligations)**.** *For every accepted* $\mathsf{RealClamping}^{\uparrow}$ *carrier*

$$C = (X, L, U, D, W, R, E, I, M, A, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Real seal, endpoint Real rows, dyadic tolerance ledger, finite window, regular rational readback, equality boundary, located interval row, min/max row, positive-radius row, transport, replay, provenance, and local naming;*

2. *interval-bound exposure, requiring L, U, I, A to be displayed before the clamp row can be consumed as bounded Real data;*

3. *finite readback exactness, requiring every clamp request to pass through D, W, R before either equality-boundary or min/max data are inspected;*

4. *equality-boundary confinement, requiring E to serve only as the displayed endpoint-overlap boundary and not as quotient equality of reals;*

5. *min/max routing, requiring M to form the clamp through the displayed* $\mathsf{RealMinMax}^{\uparrow}$ *row rather than through host trichotomy or host order;*

6. *non-escape, requiring H, T, P, N to export no host trichotomy, quotient equality, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

No RealClamping obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1810.1. The accepted packet displays exactly the source Real seal, two endpoint rows, the dyadic tolerance ledger, finite window, regular rational readback, equality boundary, located interval row, min/max row, positive-radius row, and structural rows.

Interval-bound exposure is forced by the rows L, U, I, A. A consumer cannot read a bounded clamp value before the endpoints, located interval row, and positive-radius budget have been displayed as finite packet data.

Finite readback exactness is the middle route D, W, R. The tolerance ledger fixes the requested precision, the StreamName row opens the finite observations, and the RegSeqRat row repacks those observations before equality or min/max data are available.

The equality-boundary and min/max obligations are separate. The row E records only the displayed overlap boundary at the requested tolerance, while M records the constructive min/max route that forms the bounded clamp.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier inventory omits host trichotomy, quotient equality, selected limits, countable choice, ambient Real completeness, Lean verification, and bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 1810.3 (RealClamping interval seal)**.** *Let $C = (X, L, U, D, W, R, E, I, M, A, H, T, P, N)$ be an accepted* $\mathsf{RealClamping}^{\uparrow}$ *carrier. Every* $\mathsf{LocatedRealInterval}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of the clamp factors through the displayed route*

$$X, L, U \longmapsto D, W, R \longmapsto E, I, A \longmapsto M \longmapsto H, T, P, N.$$

Consequently the exported clamp is sealed only as a bounded interval value over the displayed Real, RegSeqRat, StreamName, DyadicRatCore, RealEquality, LocatedRealInterval, RealMinMax, and PositiveReal rows. It is not a host-order projection, quotient representative, selected limit, or ambient completeness consequence.

Proof. Start with the accepted carrier and project the rows $X, L, U, D, W, R, E, I, M, A, H, T, P, N$. The only Real-facing inputs are the source seal and the two endpoint rows.

The readback route then passes through D, W, R. These rows determine the finite tolerance request, the observation window, and the regular rational readback before any interval or clamp output can be consumed.

The interval seal is E, I, A. The equality row handles endpoint overlap at the displayed tolerance, the located interval row exposes the bounded containment request, and the positive-radius row records the separation budget.

The min/max row M is downstream of those interval rows. It forms the clamp through the displayed constructive operation and then hands the result to H, T, P, N for transport, replay, provenance, and local naming.

No other coordinate is present. Hence every consumer receives only the displayed bounded interval seal and cannot extract host trichotomy, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1810.4 (RealClamping Cauchy stability)**.** *For every accepted* $\mathsf{RealClamping}^{\uparrow}$ *carrier $C = (X, L, U, D, W, R, E, I, M, A, H, T, P, N)$, replacing X, L, U, W, R by componentwise* hsame-*transported* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows preserves the same public clamp boundary whenever the transported packet keeps the displayed D, E, I, M, A, T, P, N rows. The stabilized boundary is again the finite route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RealEquality}^{\uparrow}$*,* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{RealMinMax}^{\uparrow}$*,* $\mathsf{PositiveReal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*; it does not require a selected limit or ambient Real completeness theorem.*

Proof. Componentwise transport is supplied by the row H. It relates the source seal, endpoint rows, finite window, and regular readback without adding any coordinate outside the carrier.

The fixed rows D, E, I, M, A keep the analytic boundary unchanged. They preserve the same dyadic tolerance ledger, equality boundary, located interval request, min/max operation row, and positive-radius budget.

Displayed replay is supplied by T. The transported request therefore follows the same finite route from Real seals through StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, RealEquality boundary, LocatedRealInterval containment, RealMinMax operation, and PositiveReal radius.

The provenance and local naming rows P, N repack that route as the same public $\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}$ surface. Since no selected limit, quotient equality, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness field is present, Cauchy stability is exactly stability of the displayed finite packet. □

Falsifiable prediction: Every $\mathsf{RealClamping}^{\uparrow}$ read factors through one $\mathsf{Real}^{\uparrow}$ source seal, two $\mathsf{Real}^{\uparrow}$ endpoint rows, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{StreamName}^{\uparrow}$ finite window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{RealEquality}^{\uparrow}$ boundary row, one $\mathsf{LocatedRealInterval}^{\uparrow}$ interval row, one $\mathsf{RealMinMax}^{\uparrow}$ operation row, one $\mathsf{PositiveReal}^{\uparrow}$ radius row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore, real-equality, located-real-interval, real-min-max, positivereal: $\mathsf{RealClamping}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source and endpoint Real rows, finite tolerance readback, equality-boundary handling, located interval exposure, min/max clamping, positive-radius data, and one public bounded clamp boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1810.1, Definition 1810.2, Definition 1810.3, and Definition 1810.4 seed the finite constructive interval-clamp route from displayed Real rows through dyadic tolerance, finite windows, regular rational readback, equality boundary, located interval data, min/max operation, positive-radius budget, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealClamping}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-bound exposure, finite readback exactness, equality-boundary confinement, min/max routing, interval sealing, and Cauchy stability for BEDC.Derived.RealClampingUp.

Constructive story: A $\mathsf{RealClamping}^{\uparrow}$ packet is generated from one RealUp source seal, two endpoint RealUp rows, a DyadicRatCore tolerance ledger, StreamName finite window, RegSeqRat readback, RealEquality boundary, LocatedRealInterval row, RealMinMax operation row, PositiveReal radius row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{RealSeparability}^{\uparrow}$ rational dense enumeration is read through the dyadic-density surface of chapter 1484, the rational-density surface of chapter 1409, and the countable dense subset boundary of chapter 1566 before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Theorem 1810.5 (RealSeparability finite-density lattice)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *density packet presents its sibling dependency lattice as the finite route through chapter 37, chapter 291, chapter 126, chapter 301, chapter 38, and chapter 1566. A density consumer reaches the local separability row only after these sibling certificates expose the rational source, dyadic tolerance, finite window, regular readback, terminal Real seal, and countable dense support.*

Proof. Project the carrier of Definition 1809.1. Its displayed rows are the rational enumeration, dyadic tolerance, StreamName window, RegSeqRat readback, Real seal, dense-window exactness ledger, transport, replay, provenance, and local naming row.

Read those rows through the sibling chapters named in the statement. The rational source is supplied by $\mathsf{Rat}^{\uparrow}$, tolerance by $\mathsf{DyadicRatCore}^{\uparrow}$, finite observation by $\mathsf{StreamName}^{\uparrow}$, regular readback by $\mathsf{RegSeqRat}^{\uparrow}$, the terminal public exit by $\mathsf{Real}^{\uparrow}$, and dense support by $\mathsf{CountableDenseSubset}^{\uparrow}$.

Since the carrier has no coordinate for host real equality, choice-selected dense points, quotient streams, trichotomy, or ambient completeness, this displayed sibling lattice is the whole finite-density dependency route. □

Theorem 1810.6 (Finite Cauchy density nets extract RealUp separability)**.** *Let* $N = (D_{\mathrm{net}}, W_{\mathrm{net}}, R_{\mathrm{net}}, E_{\mathrm{net}}, M_{\mathrm{net}},$ *be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet over dyadic and regular-Cauchy windows, and let*

$$S = (Q, D, W, R, E, T, H, C, P, N_S)$$

be the $\mathsf{RealSeparability}^{\uparrow}$ *density carrier whose finite observation request reads the same dyadic mesh, finite stream window, regular-Cauchy readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then the finite density-net certificate induces the finite observation row needed by* $\mathsf{RealSeparability}^{\uparrow}$*:* $D_{\mathrm{net}}, W_{\mathrm{net}}, R_{\mathrm{net}}, E_{\mathrm{net}}, M_{\mathrm{net}}$ *supply the dyadic witness family, finite window, regular readback, Real seal, and precision budget for* Q, D, W, R, E*, while* $T_{\mathrm{net}}, C_{\mathrm{net}}, P_{\mathrm{net}}, L_{\mathrm{net}}$ *transport, replay, source, and name the same separability route. Classifier stability is inherited from regular-Cauchy tail equivalence and dyadic mesh refinement, so the induced observation row exports no host countability assumption, selected dense point, quotient stream equality, trichotomy branch, or ambient completeness witness.*

Proof. First project the finite density-net packet. Its rows display the dyadic net ledger, finite StreamName window, RegSeqRat readback, Real seal, and precision budget before any density read is exported.

Next align those rows with the $\mathsf{RealSeparability}^{\uparrow}$ carrier. The rational enumeration and dyadic tolerance rows Q, D consume the net ledger and budget, the finite window W consumes W_{net}, and the regular readback and Real seal rows R, E consume $R_{\mathrm{net}}, E_{\mathrm{net}}$.

The classifier comparison is stable because regular-Cauchy tail equivalence preserves the readback row, while dyadic mesh refinement preserves the finite net ledger and its tolerance budget. The transport and replay rows of the density-net packet therefore induce the transport and replay required by the separability carrier.

Finally apply the finite-density exhaustion route for $\mathsf{RealSeparability}^{\uparrow}$. The separability read receives only the displayed finite observation rows and their provenance. Since neither packet contains host countability, a selected dense point, quotient stream equality, trichotomy, or ambient completeness, the induced observation row cannot export any of those surfaces. □

Every $\mathsf{RealSeparability}^{\uparrow}$ density read is consumed through the sibling lattice $\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$, with substantive producer and consumer rows supplied by $\mathsf{CountableDenseSubset}^{\uparrow}$ in chapter 1566, $\mathsf{DyadicDensityReal}^{\uparrow}$ in chapter 1484, and $\mathsf{RationalDensityReal}^{\uparrow}$ in chapter 1409. The $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is the only public exit, the finite windows are those of chapter 126, regular readback is the route of chapter 301, and the tolerance ledger is the dyadic surface of chapter 291; thus a separability consumer reads density data as finite rational-window evidence before it reaches the real seal, with no host equality, selected dense point, quotient stream, or ambient completeness row.

Theorem 1810.7 (RealSeparability dependency route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density consumer reaches the local* NameCert *row only after passing through the rational source of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. Thus the separability packet exposes density as a finite rational-window route over displayed BEDC rows, not as a host dense subset, quotient real equality, selected nearest point, or ambient completeness principle.*

Proof. Project the $\mathsf{RealSeparability}^{\uparrow}$ carrier through its local construction file. The density consumer first reads the rational enumeration row, so the route begins with the $\mathsf{Rat}^{\uparrow}$ source surface rather than an external dense set.

The dyadic tolerance and finite-window stages are then forced by the displayed tolerance ledger. The $\mathsf{DyadicRatCore}^{\uparrow}$ row fixes the requested rational scale, and the $\mathsf{StreamName}^{\uparrow}$ row supplies exactly the inspected finite windows.

The readback and seal stages are $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Regular rational readback consumes the same finite windows, and the terminal real-name seal is reached only after that readback is displayed.

The local NameCert row packages this route with its hsame, Cont, and Pkg provenance. Since the route has no coordinate for a host dense subset, quotient real equality, selected nearest point, or completeness theorem, every density consumer factors through the listed sibling chapters before the local certificate row. □

Theorem 1810.8 (Real separability dyadic lattice link)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read consumes the dyadic density lattice through chapter 38, chapter 37, and chapter 291 before any separability observation is exported. The* $\mathsf{Real}^{\uparrow}$ *row supplies the terminal real-name seal, the* $\mathsf{Rat}^{\uparrow}$ *row supplies the rational dense source, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row supplies the finite tolerance lattice that controls each displayed approximation window.*

The same dependency route reads dense dyadic witnesses through chapter 291, regular-sequence readback through chapter 301, and the completed real seal through chapter 38 before the $\mathsf{RealSeparability}^{\uparrow}$ *carrier exports its finite density row.*

Proof. Project the accepted separability packet to its displayed density route. A consumer first reads the real-name seal so that the observation is a $\mathsf{Real}^{\uparrow}$-facing read rather than a host dense-subset claim.

The rational source is then supplied by the $\mathsf{Rat}^{\uparrow}$ row. Its enumeration is meaningful for separability only after the dyadic tolerance lattice has fixed the finite scale at which the approximation is inspected.

The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies that finite scale and feeds the same request into the existing density route. Hence the separability observation cannot float as a kernel-only island: it is exported only after the real, rational, and dyadic lattice rows have all been consumed. □

Theorem 1810.9 (Real separability Cauchy completion route)**.** *Every dense dyadic witness exported by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet feeds* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{BishopCompletionCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *only through finite windows and dyadic readback. The route first displays the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then the completion or Real-facing seal. It has no coordinate for quotient stream equality, ambient completeness, selected host limits, or an unbounded dense-set search.*

Proof. Use the density carrier and finite density lattice already present in the $\mathsf{RealSeparability}^{\uparrow}$ packet. They expose dense dyadic witnesses as finite certificate rows rather than as a global choice function.

The dyadic lattice link places each witness on a $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance ledger. The sibling route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ then reads that ledger through a finite window before any completion-facing consumer can use it.

Finally the completion consumers receive the same finite readback. Since the packet has no row for quotient streams, ambient completeness, selected host limits, or unbounded dense search, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ and $\mathsf{BishopCompletionCriterion}^{\uparrow}$ can consume the witness only through the displayed finite-window and dyadic-readback route. □

Theorem 1810.10 (Real separability Cauchy-equivalence lattice route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *finite density read that is consumed by a Cauchy-equivalence or completion classifier must pass through the displayed* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$

sibling rows. The route exposes finite density windows and dyadic readback before the Cauchy-equivalence classifier is read, and it exports no quotient real equality, selected dense enumeration, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the density routes of Definition 1809.12 and Definition 1809.3, whose dyadic-rational handoff is recorded in Definition 1809.6. They read the rational dense enumeration through finite dyadic and rational-density surfaces before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Next use the completion-facing sibling route, Definition 1810.9. It forces the same density witness through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ finite windows before any completion-facing consumer receives the read.

Finally attach the Cauchy-equivalence classifier of chapter 1820. The classifier may consume only the displayed finite windows and sibling rows already supplied by the separability route. Since neither route includes quotient real equality, a selected dense enumeration, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the reciprocal lattice edge cannot export any of those claims. □

The Cauchy-equivalence route for $\mathsf{RealSeparability}^{\uparrow}$ is the finite density-row consumer in which $\mathsf{RegSeqRat}^{\uparrow}$ windows and the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier are both visible before the $\mathsf{Real}^{\uparrow}$ density row is exported. A separability consumer reads chapter 301 for the regular rational windows, then chapter 1820 for the comparison surface that keeps equivalent regular names on the same finite density route, and only then chapter 38 for the terminal Real seal. Thus the dependency edge records finite readback and classifier data, not quotient real equality, a selected dense enumeration, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Lemma 1810.11 (Real separability Cauchy-equivalence dependency route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that is consumed through a Cauchy-equivalence classifier must display the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback windows and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier before the* $\mathsf{Real}^{\uparrow}$ *density row is exported. The route is a child dependency of chapter 1809, chapter 301, and chapter 1820; it exports no quotient stream equality, selected dense point, host real equality, or ambient completeness.*

Proof. Project the Real separability density read through the displayed finite route. The rational candidate and tolerance request are not yet Real data; they must first be read through the regular rational windows supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

Next read the Cauchy-equivalence classifier. It compares the regular names only through the displayed finite classifier rows of $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, so equivalent names replay the same density route before the Real seal is available.

Finally export the terminal $\mathsf{Real}^{\uparrow}$ density row. Since the preceding rows contain no quotient stream equality, selected dense point, host real equality, or ambient completeness coordinate, the exported separability read contains none of them. □

Every $\mathsf{RealSeparability}^{\uparrow}$ finite-density read that compares two regular real names first consumes the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier surface of chapter 1820 before the density row is exported. The route displays the $\mathsf{RegSeqRat}^{\uparrow}$ readback windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before any separability consumer reads density; it exports no quotient real equality, selected dense enumeration, countable choice, host equality, or ambient completeness. The hub status route for $\mathsf{RealSeparability}^{\uparrow}$ is the child closurestatus block in Definition 1810.19, whose scoped surface is recorded inside the routed child file before the hub exposes the Real and completion dependency lattice. The $\mathsf{RealSeparability}^{\uparrow}$ hub is seeded by its displayed child routes: density, Cauchy-equivalence, dyadic-lattice, completion, RegSeqRat, MetricCompletion, and countable-dense siblings jointly expose the finite separability surface without adding theorem content to the

hub. Its seed route binds $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, density, completion, and countable-dense routes as the visible finite rows through which dense Real names are read, and the hub-level seed registration records exactly this child-route family. The adjacent obligation marker on the hub points to the same child-route family: the route becomes obligation-level only through the carrier, route exactness, density coverage, Cauchy compatibility, dyadic compatibility, completion handoff, transport, replay, provenance, and local naming rows named in this child status block.

Theorem 1810.12 (Real separability seed status)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *seed status binds the existing Cauchy-equivalence,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *density routes as the displayed child-route family through which dense Real names are read. It does not claim countable choice, quotient Real equality, full Real completeness, or any verified density theorem beyond the rows named in the seed ledger.*

Proof. The seed child lists the visible route family and fixes the route at seed with $\mathsf{unformalized}$ formal status. Thus the record is a ledger statement about displayed child routes, not a proof object for the formal axis.

The closure and verification axes remain separate: the former says which paper rows are named, while the latter is still bounded by the absence of a checked carrier admission, route exactness theorem, density coverage theorem, and completion handoff for $\mathsf{RealSeparability}^{\uparrow}$. □

Theorem 1810.13 (Real separability finite dense-window seed target)**.** *Finite dense-window packets over* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *provide the paper-side seed target for* $\mathsf{RealSeparability}^{\uparrow}$*. The target reads dyadic tolerance rows, finite stream observations, regular-rational readback, and the terminal Real seal as displayed packet data before any dense-name consumer is exposed. It does not claim classical separability of* $\mathsf{Real}^{\uparrow}$*, a chosen countable dense subset, quotient Real equality, or countable choice.*

Proof. Read the finite-window rows already displayed by the seed status block. The dyadic row supplies the tolerance ledger, the StreamName row supplies bounded observations, the RegSeqRat row supplies regular rational readback, and the Real row supplies the terminal seal.

The conclusion is therefore a paper-side seed target over those finite packets. It remains on the theory axis: no `\leanchecked{}` marker is attached here, and the formal verification axis is still the one recorded in the surrounding closurestatus block.

The refused endpoints follow from the same row inspection. None of the displayed finite-window rows contains a classical separability theorem, a selected dense enumeration, quotient Real equality, or a countable-choice schedule. □

Theorem 1810.14 (Real separability scoped density route)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density read factors through the L10 source chain*

$$\mathsf{CauchyEquivalenceSetoid}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before exposing a countable-dense seal. The route reads the Cauchy-equivalence classifier, dyadic tolerance ledger, finite stream windows, regular-rational readback, and terminal Real seal as displayed rows, and it exports no quotient stream equality, selected dense enumeration, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the seed route named by Definition 1810.12. Its child-route family keeps density, Cauchy-equivalence, dyadic approximation, finite StreamName windows, RegSeqRat readback, and the Real seal visible as separate rows.

Use the Cauchy-equivalence routes, including Definition 1810.10 and Definition 1810.21. They place the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier on the density read without converting it into quotient stream equality.

Next compose with the dyadic-density and L10 source routes, including Definition 1809.6 and Definition 1809.5. These keep the dyadic tolerance, stream-window, RegSeqRat, and Real rows visible before a countable-dense consumer receives the seal.

The refused endpoints follow from the same route discipline. Because the seed packet exposes only the listed child rows, it supplies neither a selected dense enumeration nor countable choice nor ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1810.15 (RealSeparability seed closure route)**.** *The seed closure route for* $\mathsf{RealSeparability}^{\uparrow}$ *is the child-file route that binds dyadic and rational density,* $\mathsf{RegSeqRat}^{\uparrow}$ *windows,* $\mathsf{StreamName}^{\uparrow}$ *witnesses, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any dense-name consumer reads a separability row. It is the critical-path to-seed rotation witness over the displayed child ledger of Definition 1810.12 and Definition 1810.14; it exports no host dense enumeration, quotient Real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1810.12 to identify the seed ledger as the child-route family rather than as theorem content on the hub. The visible rows include density, Cauchy-equivalence, dyadic approximation, StreamName windows, RegSeqRat readback, completion-facing handoff, and the Real seal.

Then apply Definition 1810.14. It orders the Cauchy-equivalence, dyadic, StreamName, RegSeqRat, and Real rows before the countable-dense seal is exposed, so the seed route is exactly the displayed finite closure route stated above. □

Theorem 1810.16 (Real separability seed status route)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *seed-status route is the displayed L10 chain through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal, with* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *naming kept visible before any dense-name or countable-dense consumer reads the packet. It records the existing separability rows as a finite child-route family and exports no quotient Real equality, selected dense enumeration, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 1810.12. It identifies the seed surface as a child-route family rather than as data stored on the hub.

Apply the scoped density route and seed closure route, Definition 1810.14 and Definition 1810.15. They order the dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, and terminal Real seal before the countable-dense row is visible.

Finally read the structural rows componentwise. The hsame, Cont, Pkg, and NameCert entries preserve and name the same child ledger, so the seed-status route contains exactly the displayed separability rows and none of the refused endpoints. □

Theorem 1810.17 (Real separability Cauchy-setoid dyadic density obligation)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density request must expose the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any countable-dense row is consumed. The obligation is exactly the finite child-route coverage named by Definition 1810.12, Definition 1810.14, and Definition 1810.15; it exports no quotient Real equality, selected dense enumeration, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 1810.12. It identifies the $\mathsf{RealSeparability}^{\uparrow}$ seed surface as the displayed child-route family rather than as content stored on the hub.

Apply Definition 1810.14. The route orders the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier, dyadic approximation, finite StreamName windows, RegSeqRat readback, and terminal Real seal before the countable-dense seal is visible.

Finally use Definition 1810.15. It records the same child ledger as the seed closure route, so the density request is covered precisely by those finite rows. Since the route contains no quotient Real equality, selected enumeration, countable-choice schedule, or ambient completeness row, those endpoints are explicitly outside the obligation. □

Theorem 1810.18 (Real separability dyadic enumeration route)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *dense read that exports a countable-dense surface factors first through the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *density ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The exported surface is therefore a finite dyadic schedule read from the displayed child ledger, not a countable-choice selector, host enumeration, quotient Real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 1810.12. It fixes the dense read as a child-route family over Cauchy equivalence, dyadic approximation, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Apply Definition 1810.14 to order those rows before the countable-dense seal is visible. Then use Definition 1810.17; it requires the Cauchy-equivalence classifier and dyadic tolerance row before any dense consumer can inspect the exported surface.

The remaining endpoints are absent from the packet. Since the route contains only the displayed finite rows, it cannot supply a countable-choice selector, a host enumeration, quotient Real equality, or an ambient completeness theorem. □

Theorem 1810.19 (Real separability seed closure status block)**.** *The scoped closurestatus for* $\mathsf{RealSeparability}^{\uparrow}$ *is read from the displayed child-route family of Definition 1810.12, Definition 1810.14, Definition 1810.15, Definition 1810.17, and Definition 1810.18. It records only the paper-side scope over dyadic density, StreamName windows, RegSeqRat readback, the Real seal, and regular-Cauchy dense embedding routes, and it does not upgrade the formal verification axis.*

Proof. Read the displayed child-route theorems in order. They identify the seed ledger, the scoped density route, the seed closure route, the Cauchy-setoid dyadic density obligation, and the dyadic enumeration route as paper rows under the $\mathsf{RealSeparability}^{\uparrow}$ hub.

The closurestatus block records that same child-route family at scopedClosed. Its unformalized formalstatus remains unchanged, so the displayed block is a scope ledger and not a Lean verification claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSeparability}^{\uparrow}$ packet is generated from the existing child routes for density, Cauchy-equivalence, dyadic approximation, StreamName windows, RegSeqRat readback, Real seal, MetricCompletion handoff, completion-facing handoff, and countable dense Real-facing schedules.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The scoped surface binds dyadic density, StreamName finite windows, RegSeqRat readback, the Real seal, and the regular-Cauchy dense embedding route under chapter 1809, without claiming classical separability for arbitrary spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This seed status does not claim classical separability of arbitrary spaces, $\mathsf{Real}^{\uparrow}$ completeness, a countable-choice dense enumeration, a checked Lean carrier, bridge checking, or a standalone separability theorem beyond the child-route packet surface.

Upgrade path: Public closure requires checked carrier admission, route exactness, density coverage, Cauchy and dyadic compatibility, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealSeparabilityUp.

Theorem 1810.20 (Real separability Cauchy-setoid dependency)**.** *The* RealSeparability$^{\uparrow}$ *finite dense-read surface consumes the* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier route of chapter 1820 and the regular rational readback route of chapter 301 before any countable dense* Real$^{\uparrow}$ *read is exported. Thus the separability packet may expose finite dense witnesses only after the setoid comparison, regular-rational readback, tolerance ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows are visible.*

Proof. Read the dependency route in order. The Cauchy-equivalence setoid fixes when two regular Cauchy names may be treated as the same dense-read source, and the RegSeqRat$^{\uparrow}$ row supplies the finite rational windows that the separability packet later counts.

Only after those two sibling routes are visible may the RealSeparability$^{\uparrow}$ packet expose its countable dense Real readback. The route therefore exports no quotient real equality, selected dense enumeration, countable choice, or ambient completeness theorem. □

The RealSeparability dense-read route exposes the Cauchy-equivalence classifier as a downstream consumer edge, not as a free equality principle. A dense-read consumer receives the classifier data of chapter 1820 only after the finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance rows, and Real$^{\uparrow}$ seal rows of the separability route have been displayed.

Theorem 1810.21 (Real separability Cauchy-equivalence dense-read route)**.** *Every* RealSeparability$^{\uparrow}$ *dense-read consumer that uses* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier data must first display the* StreamName$^{\uparrow}$ *windows, the* RegSeqRat$^{\uparrow}$ *readback row, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and the terminal* Real$^{\uparrow}$ *seal carried by the RealSeparability route. Thus the classifier is consumed as finite Cauchy-equivalence data after the L10 source rows have been shown, not as quotient stream equality, selected dense enumeration, host real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the displayed RealSeparability route. The density read begins with finite source windows and rational readback; those rows are not yet a Cauchy-equivalence classifier.

Next expose the dyadic tolerance and Real-seal rows. They place the dense-read request on the same L10 route used by the existing RealSeparability children, so the classifier cannot be consumed before the source data are displayed.

Only then read the Cauchy-equivalence classifier of chapter 1820. Since the route has no coordinate for quotient stream equality, selected dense enumeration, host real equality, countable choice, or ambient completeness, the consumer receives only the finite classifier data. □

The RealSeparability$^{\uparrow}$ dense reads that compare equivalent regular Cauchy names pass through chapter 1820 after the RegSeqRat$^{\uparrow}$ window route has displayed regular rational readback and before the terminal Real$^{\uparrow}$ density export is consumed.

Theorem 1810.22 (RealSeparability forward Cauchy-equivalence route)**.** *Every* RealSeparability$^{\uparrow}$ *dense-read consumer that uses* CauchyEquivalenceSetoid$^{\uparrow}$ *data factors through the finite classifier boundary of chapter 1820. The route first displays the* RegSeqRat$^{\uparrow}$ *readback rows, the* DyadicRatCore$^{\uparrow}$ *tolerance rows, the finite* StreamName$^{\uparrow}$ *windows, and the terminal* Real$^{\uparrow}$ *seal before exporting a dense-read surface. Thus the Cauchy-equivalence classifier is consumed as finite BEDC data, not as quotient stream equality, selected dense enumeration, countable choice, host real equality, or ambient completeness.*

Proof. Read the RealSeparability hub through its existing dense-read children. Those files expose finite stream windows, regular rational readback, dyadic tolerance rows, and the terminal real seal before any dense surface is exported.

Only after those rows are displayed may a consumer call the Cauchy-equivalence classifier. The classifier boundary is therefore the chapter named by chapter 1820, with $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ providing the finite route into it.

Since the route has no coordinate for quotient stream equality, selected dense enumeration, countable choice, host real equality, or ambient completeness, it exports only the displayed dense-read classifier data. □

Theorem 1810.23 (RealSeparability completion-facing display route)**.** *Every completion-facing* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reaches* $\mathsf{Real}^{\uparrow}$ *only after the rational enumeration of chapter 37, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route of chapter 301, the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *source of chapter 1566, and the terminal seal of chapter 38 have all been displayed. The completion-facing route exports finite density data only through that lattice, not through quotient real equality, selected nearest points, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the rational source row in the $\mathsf{RealSeparability}^{\uparrow}$ carrier. It provides the enumerated rational data through the $\mathsf{Rat}^{\uparrow}$ sibling surface rather than through an external dense-set selector.

The dyadic tolerance and finite-window stages then force the request through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$. The $\mathsf{RegSeqRat}^{\uparrow}$ row reads back the same displayed windows at the requested precision.

Finally the dense-source row is named by $\mathsf{CountableDenseSubset}^{\uparrow}$, and the terminal public boundary is $\mathsf{Real}^{\uparrow}$. Since no carrier row stores quotient real equality, a selected nearest point, countable choice, or ambient completeness, completion-facing Real routes can consume only the displayed lattice. □

The dense-embedding route is the completion-facing dependency between the density siblings and the public separability row. A $\mathsf{RealSeparability}^{\uparrow}$ density consumer first reads the dyadic and rational density routes of Definition 1809.6 and Definition 1809.12; when the same consumer asks for completion-facing reuse, the finite dense image-window and approximation ledger of chapter 3617 are read before the $\mathsf{CountableDenseSubset}^{\uparrow}$ row is exported.

Theorem 1810.24 (Real separability dense-embedding route)**.** *Every completion-facing* $\mathsf{RealSeparability}^{\uparrow}$ *density read that consumes canonical* $\mathsf{DenseEmbedding}^{\uparrow}$ *data factors through the displayed dense image-window and approximation ledger before it reaches the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *boundary. Thus the separability packet may reuse dense-embedding data only as finite source, window, tolerance, approximation, and handoff rows, not as host topological density, a selected dense point, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite density exhaustion of Definition 1809.12. The RealSeparability consumer already passes through $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary.

Now insert the canonical dense-embedding packet of chapter 3617. Its completion handoff exposes only the dense image-window, approximation ledger, and local handoff rows before a completion-facing read is available.

Composing the two routes leaves the CountableDenseSubset row downstream of finite density data and finite dense-embedding data. Since neither route carries host topological density, selected

dense points, quotient real equality, countable choice, or ambient completeness, the composed separability read exports none of those objects. □

Theorem 1810.25 (Real separability RegSeqRat dependency)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read consumes the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window route and the dyadic-density witness before it exports separability. The read first reaches the terminal real-name surface of chapter 38, then routes the requested rational approximation through chapter 301, and finally checks the tolerance witness through chapter 1484. The resulting separability observation is therefore a finite regular-sequence and dyadic-density handoff, not a host dense-subset assertion, selected nearest point, quotient stream, or ambient completeness claim.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier through the local density route. The Real-facing consumer can only read the terminal $\mathsf{Real}^{\uparrow}$ seal after the rational enumeration, dyadic tolerance, StreamName window, and regular rational readback rows have been displayed.

The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the finite-window readback. It carries the inspected rational candidate and the opened observation window as regular sequence data, so the separability consumer does not receive a detached host rational point.

The dyadic-density sibling then supplies the tolerance witness. It binds the displayed dyadic scale to the same Real-facing request before any separability row is exported.

At the horizontal dependency level, the finite density rows are read through the dyadic endpoints of chapter 291 and the regular-rational windows of chapter 301 before the $\mathsf{Real}^{\uparrow}$ seal can present them as separability evidence. This makes the route a concrete dyadic-window handoff rather than a detached dense-subset assertion.

The structural hsame, Cont, Pkg, and local NameCert rows replay only those displayed rows. Since none of the routes contains a coordinate for a host dense subset, selected nearest point, quotient stream, or ambient completeness, every $\mathsf{RealSeparability}^{\uparrow}$ density read depends on the RegSeqRat finite-window route and dyadic-density witness before export. □

Theorem 1810.26 (RealSeparability RegSeqRat density pullback)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *countable-density export factors first through the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readback and the dyadic-density witness before it reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular, the dependency route of Definition 1810.25 and the countable dense route of Definition 1810.39 force each exported dense read to display its regular rational window, dyadic tolerance witness, and Real seal boundary; no host dense subset, countable-choice selector, or ambient completeness coordinate can supply the density export.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier. Its density row can expose only the displayed rational or dyadic schedules, finite readback rows, Real-facing seal, transport, replay, provenance, and local naming data.

Use the RegSeqRat dependency of Definition 1810.25. It places the finite-window readback before any Real-facing separability export, so the density read cannot start from an untracked host subset.

Next use the dyadic-density sibling route and Definition 1810.39. The dyadic witness supplies the tolerance row consumed by the countable dense schedule before the terminal $\mathsf{Real}^{\uparrow}$ seal is read.

Thus the Real seal receives only the displayed RegSeqRat windows, dyadic witness, and certificate rows. Since the carrier has no coordinate for a host dense subset, choice selector, or ambient completeness theorem, every countable-density export factors through the stated pullback route. □

Theorem 1810.27 (RealSeparability located RegSeqRat density readback)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density-window consumer must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route of chapter 301, the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *locatedness route of chapter 2376, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before exporting a dense-read handoff. The exported handoff is finite regular-sequence and locatedness data; it is not quotient real equality, a selected dense point, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the RealSeparability carrier. Its density-window route names finite observations and regular-rational reads before any real-facing handoff is available.

Apply the RegSeqRat schedule route. The schedule supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback and finite $\mathsf{StreamName}^{\uparrow}$ windows, while the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells record the displayed comparison budgets consumed by the density request.

The located regular-sequence route then supplies the $\mathsf{LocatedRegSeqRat}^{\uparrow}$ locatedness data. Only after these rows are displayed can the handoff reach the terminal $\mathsf{Real}^{\uparrow}$ seal.

Thus the dense-read handoff is a finite readback route through regular sequences, locatedness, stream windows, dyadic tolerances, and the Real seal. Since the carrier exports none of the excluded coordinates, no quotient real equality, selected dense point, countable choice, or ambient completeness coordinate is exported. □

Theorem 1810.28 (Real separability dyadic density dependency)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read factors through the regular rational windows of chapter 301, the dyadic tolerance surface of chapter 291, and the terminal seal of chapter 38 before it exports a separability witness. The dense dyadic candidate is visible only as a finite* $\mathsf{RegSeqRat}^{\uparrow}$ *window with a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and a downstream* $\mathsf{Real}^{\uparrow}$ *seal, not as a detached host dense subset or selected nearest point.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ packet to its density route. The requested approximation is first read through the finite $\mathsf{RegSeqRat}^{\uparrow}$ window, so the consumer receives a displayed regular rational observation rather than an unlogged rational point.

The $\mathsf{DyadicRatCore}^{\uparrow}$ row then supplies the density scale and the tolerance comparison used by that same window. This makes the dyadic witness part of the finite dependency lattice, not an external density theorem.

Finally the $\mathsf{Real}^{\uparrow}$ seal consumes the regular window and dyadic tolerance rows before the separability row is public. Hence separability reads dense dyadic witnesses only through RegSeqRat windows and RealUp sealing. □

Theorem 1810.29 (Real separability dyadic density induction)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *rational-density read used by a completion-facing consumer factors through a finite induction on dyadic windows before reaching* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. At each induction stage the consumer reads the displayed dyadic tolerance row, the finite stream window, the regular rational readback, and the terminal real-name seal already carried by the RealSeparability route; it receives no classical countable choice, no selected dense subsequence, no quotient equality of Real names, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with Definition 1810.28. It places every density request behind the dyadic tolerance surface and the finite regular-rational window before the terminal real-name seal can be read.

Use induction over the displayed dyadic window length. The base window is the empty or initial tolerance request already visible in the $\mathsf{DyadicRatCore}^{\uparrow}$ row, so no density witness can be consumed before the dyadic scale is named.

For the extension step, append one displayed dyadic refinement window and replay the existing RegSeqRat and StreamName dependencies. The dependency route keeps the new rational candidate inside the same finite window ledger rather than selecting a point from an ambient dense subset.

Finally compose with Definition 1810.25 and Definition 1810.76. These routes pass the finite dyadic-window induction through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal, while excluding countable choice, selected dense subsequences, quotient Real equality, and ambient completeness. □

Theorem 1810.30 (Real separability dyadic density scope)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that consumes a dyadic rational witness factors through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *numerator and exponent rows, the finite observation windows of* $\mathsf{StreamName}^{\uparrow}$*, the regular readback surface of* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports a scoped finite density witness only after these four surfaces have been read, and it exports no selected representative, quotient stream, host-countable enumeration, nearest-dyadic selector, or ambient completeness theorem.*

Proof. Start with the dyadic-density dependency and induction routes. Theorem 1810.28 places each density request behind the dyadic tolerance ledger, while Definition 1810.29 keeps each refinement step inside a finite dyadic window.

Next read the stream and regular-rational routes. The finite $\mathsf{StreamName}^{\uparrow}$ observation window records which prefixes are inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ dependency route supplies the regular rational readback before the separability ledger is exported.

Finally compose with the forward dyadic-density link and the terminal $\mathsf{Real}^{\uparrow}$ seal. The resulting route is the single finite chain

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

inside the $\mathsf{RealSeparability}^{\uparrow}$ packet, so no selected representative, quotient stream, host-countable enumeration, nearest-dyadic selector, or ambient completeness theorem can be recovered from the accepted density read. □

Theorem 1810.31 (Real separability regular Cauchy completion criterion route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that reaches the L10 completion tail consumes the* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *route of chapter 3452 through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The dyadic-density dependency of Definition 1810.28 supplies the finite dyadic witness, while the RegSeqRat dependency of Definition 1810.25 keeps that witness on the displayed regular-rational readback route before the terminal Real seal exports separability.*

Proof. Read the accepted $\mathsf{RealSeparability}^{\uparrow}$ packet through its existing density dependencies. The dyadic-density row names the finite tolerance witness, and the RegSeqRat row keeps the witness inside a displayed regular rational window rather than a host dense subset.

Now pass to the completion tail. The regular Cauchy completion criterion consumes precisely those finite windows, dyadic tolerance rows, regular readback rows, and local $\mathsf{Real}^{\uparrow}$ seal rows. Therefore the separability consumer can cite the completion criterion only through the same finite dyadic witness and RegSeqRat route already named by the RealSeparability packet.

The route adds no selected nearest point, quotient real equality, ambient completeness, or detached dense subset. It records a sibling edge from RealSeparability to the regular Cauchy completion criterion along the finite dyadic and regular-rational surface. □

The $\mathsf{RealSeparability}^{\uparrow}$ dependency lattice consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the dyadic-density embedding route of chapter 1484, and the located $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 2376 before exposing its separability ledger. Thus the separability packet reads rational candidates through regular sequence windows, dyadic density through the real-density sibling, and located regular readback through the located route already carried by the hub before any $\mathsf{Real}^{\uparrow}$ density observation is named. Every $\mathsf{RealSeparability}^{\uparrow}$ density witness reads through finite $\mathsf{StreamName}^{\uparrow}$ windows before it reaches any Real-facing separability row. The route first opens the displayed window of chapter 126, then transports the inspected rational data through the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, checks the tolerance against the $\mathsf{DyadicRatCore}^{\uparrow}$ rows of chapter 291, and only then consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Thus the separability packet exports a finite density route, not a detached dense subset, selected nearest point, quotient real equality, countable choice, or ambient completeness.

Theorem 1810.32 (Real separability L10 Cauchy-net dependency)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *countable dense read that is presented to an L10 Cauchy-net consumer first opens the dyadic tolerance rows of chapter 291, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301. Only after those finite-window rows are visible does the read reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 and become available as countable dense data for Cauchy-net or completion-facing routes.*

Proof. Project the separability source route. A dense-read consumer sees rational or dyadic candidates only through displayed finite windows and regular rational readback, so the dyadic ledger, stream window, and RegSeqRat rows are upstream of the Real-facing density claim.

The Real seal is terminal. It receives the inspected dense candidate after the finite rows have been displayed; it does not create a host dense subset, selected nearest point, quotient equality, or ambient completion coordinate.

Thus an L10 Cauchy-net consumer reads $\mathsf{RealSeparability}^{\uparrow}$ through the same DyadicRatCore, StreamName, RegSeqRat, and Real rows before any countable dense handoff is public. □

The $\mathsf{RealSeparability}^{\uparrow}$ dense-read route is not a free-standing lattice float. Every countable-density packet reads $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names from chapter 301, $\mathsf{StreamName}^{\uparrow}$ pointwise observation windows from chapter 126, and $\mathsf{DyadicRatCore}^{\uparrow}$ dyadic approximation rows from chapter 291 before exporting its dense enumeration evidence.

Proposition-style dependency spine: if a $\mathsf{RealSeparability}^{\uparrow}$ consumer claims countable-density evidence while omitting the regular-rational source, the stream-window observation row, or the dyadic approximation ledger, then the read is outside the displayed finite BEDC route. The admissible route is regular Cauchy naming, finite stream observation, dyadic approximation, transport, replay, provenance, and only then the countable-density handoff.

Theorem 1810.33 (Real separability Cauchy locator forward link)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that feeds the locator surface of chapter 3410 passes through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyRealLocator}^{\uparrow}.$$

Concretely, the dyadic tolerance row is read before the finite StreamName window, the regular rational readback is read before the terminal Real seal, and the Cauchy locator receives only this finite-window density route. The link exports no quotient stream equality, selected nearest dense point, countable-choice search, trichotomy branch, or ambient Real completeness.

Proof. Start with the density carrier of Definition 1809.1. Its rows $Q, D, W, R, E, T, H, C, P, N$ place every density request behind the rational enumeration, dyadic tolerance, finite window, regular readback, and Real seal.

Use the existing separability routes. The rational dense handoff, dyadic-rational density handoff, and Cauchy completion route keep the density witness inside the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows already named by the packet.

Finally pass to chapter 3410. Its locator requests consume dyadic tolerances, finite windows, regular readbacks, and the terminal Real seal in that order. Therefore a RealSeparability density read can feed $\mathsf{CauchyRealLocator}^{\uparrow}$ only through the displayed finite route, with no hidden selector, quotient equality, choice schedule, trichotomy, or ambient completeness coordinate.

The same forward link is a consumer-side input for chapter 3895: its apartness request first reads the $\mathsf{CauchyRealLocator}^{\uparrow}$ row, so the density support supplied by $\mathsf{RealSeparability}^{\uparrow}$ remains attached to the finite locator route rather than becoming an independent search principle. □

The $\mathsf{RealSeparability}^{\uparrow}$ packet supplies the dense finite rational rows consumed by finite-cover and uniformly located finite-cover routes. The real carrier of chapter 38, the regular rational sequence route of chapter 301, and the stream-window naming surface of chapter 126 provide the finite rational windows that chapter 2377 reads before coverage and radius ledgers are exposed.

Proposition 1810.34 (Real separability finite-cover forward link)**.** *Every finite-cover consumer of* $\mathsf{RealSeparability}^{\uparrow}$ *receives dense rational data only as displayed finite windows over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$*. The handoff may feed* $\mathsf{FiniteRealCover}^{\uparrow}$ *and later uniformly located finite-cover packets through those rows, together with* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *evidence; it does not provide a choice-selected dense subsequence, an arbitrary open-cover refinement, quotient real equality, trichotomy, or ambient completeness.*

The forward route from $\mathsf{RealSeparability}^{\uparrow}$ to completion-facing density data is finite: a separability consumer first reads the regular rational handoff of chapter 301, then the source-to-seal discipline of chapter 1453, and only then the dense-image packet of chapter 3617.

Theorem 1810.35 (Real separability RegSeqRat forward route)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet that reuses dense-embedding data factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the finite* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *source-to-seal route, and the* $\mathsf{DenseEmbedding}^{\uparrow}$ *image-window ledger before it exports a dense source to completion, finite-net, compact-metric, or* $\mathsf{CountableDenseSubset}^{\uparrow}$ *consumers. The route supplies no host dense subset, selected nearest point, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the RegSeqRat dependency of Definition 1810.25. It places the rational candidate, opened StreamName window, dyadic tolerance row, and regular readback before any Real-facing separability export.

Next read the regular Cauchy dense-embedding boundary of chapter 1453. Its source-to-seal route keeps the same $\mathsf{RegSeqRat}^{\uparrow}$ row visible while the $\mathsf{Real}^{\uparrow}$ seal is reached.

Finally pass that visible source row through the dense-embedding packet of chapter 3617. The image-window and approximation ledger are then the only completion-facing dense data available to downstream finite-net, compact-metric, and countable-density consumers.

Because each stage exposes only displayed rows and none carries a host dense subset, selected nearest point, quotient real equality, countable-choice schedule, or ambient completeness witness, the composed forward route exports no such coordinate. □

Theorem 1810.36 (Real separability RegSeqRat forward dependency)**.** *Any* RealSeparability$^{\uparrow}$ *dense-completion export must first expose the* RegSeqRat$^{\uparrow}$ *rational candidate, the* StreamName$^{\uparrow}$ *finite window, the* DyadicRatCore$^{\uparrow}$ *tolerance row, and the regular readback before it uses* RegularCauchyDenseEmbeddin *and* DenseEmbedding$^{\uparrow}$*. Consequently completion, compact-metric, finite-net, and countable-density consumers receive only the displayed regular dense route; they receive no host dense subset, selected nearest point, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 1810.35. It already places every completion-facing dense export downstream of the RegSeqRat readback row, the regular Cauchy dense-embedding source-to-seal route, and the DenseEmbedding image-window ledger.

Unpack the first step of that route. The RegSeqRat dependency exposes a rational candidate, an opened StreamName window, a DyadicRatCore tolerance row, and a regular readback before any dense-completion consumer can reach the embedding rows.

The later RegularCauchyDenseEmbedding$^{\uparrow}$ and DenseEmbedding$^{\uparrow}$ rows therefore consume only displayed finite data. Since neither row introduces a host dense subset, nearest-point selector, quotient equality oracle, choice schedule, or ambient completeness witness, the export has exactly the stated dependency surface. □

Theorem 1810.37 (Real separability dyadic density forward link)**.** *Every* RealSeparability$^{\uparrow}$ *density route that consumes a dyadic-density witness must pass that witness through the finite* RegSeqRat$^{\uparrow}$ *readback surface and the regular Cauchy completion route before it exports a separability ledger. In particular, the dyadic witness of chapter 1484 and Definition 1810.28, the RegSeqRat dependency of Definition 1810.25, the regular Cauchy completion criterion route of Definition 1810.31, and the terminal* Real$^{\uparrow}$ *seal of chapter 38 form one displayed dependency chain rather than four detachable facts.*

The exported separability ledger therefore records a finite dyadic-density route through DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, regular Cauchy completion, and* Real$^{\uparrow}$*. It does not select nearest dyadics, identify quotient Real representatives, invoke ambient completeness, or detach the dense set from the displayed regular-rational readback route.*

Proof. Start with the dyadic-density dependency. Theorem 1810.28 exposes the dyadic witness only through a finite tolerance row and a regular rational observation window.

Next read the RegSeqRat route. Theorem 1810.25 keeps that witness on the displayed regular-rational surface before any RealSeparability$^{\uparrow}$ ledger is exported.

Now pass through regular Cauchy completion. Theorem 1810.31 consumes the same finite dyadic and regular-rational rows before the terminal Real$^{\uparrow}$ seal is read.

Thus the separability ledger receives one finite dependency chain through dyadic density, RegSeqRat readback, regular Cauchy completion, and Real sealing. Since none of those rows is a nearest-point selector, quotient-real equality, ambient completeness theorem, or detached dense subset, the forward link exports only the displayed separability route. □

The RealSeparability$^{\uparrow}$ packet supplies the dense dyadic window route consumed by MetricCompletion$^{\uparrow}$ in chapter 1506 and by the Cauchy comparison surface of chapter 1820. A completion-facing read first observes the existing Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ route, then uses those finite dyadic windows as the separability source for regular readbacks and Cauchy-equivalence comparisons.

The ledger edge is theorem-free: Real separability is used only as a displayed dense-window supplier for metric completion and Cauchy equivalence consumers, not as a host countable dense subset, selected enumeration, quotient real equality, or ambient completeness principle.

Theorem 1810.38 (RealSeparability forward metric-completion route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *consumer of a* $\mathsf{RealSeparability}^{\uparrow}$ *packet first reads the* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *dense-embedding, and* $\mathsf{CountableDenseSubset}^{\uparrow}$ *rows displayed by the separability carrier. The route exports finite density and readback data only through those rows; it exports no host dense subset, selected nearest point, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the Cauchy completion route of Definition 1810.9. It places every dense dyadic witness on finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback rows before a completion-facing consumer receives it.

Next use the completion-facing display route of Definition 1810.23. That route adds the dense source and $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary required by metric-completion consumers.

Finally apply the RegSeqRat dependency of Definition 1810.25. It keeps the witness on the displayed regular-rational surface, so a $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ consumer cannot obtain a host dense subset, selected nearest point, quotient real equality, countable choice, or ambient completeness principle from the separability packet. □

The forward density route from $\mathsf{RealSeparability}^{\uparrow}$ to substantive sibling consumers is the L10 corridor through chapter 1566, the dyadic-density surface of chapter 1484, the rational-density surface of chapter 1409, the finite window discipline of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. A density read therefore opens a countable dense-subset row, checks dyadic and rational density against displayed stream windows, routes the tolerance through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$, and reaches the Real seal only after the same finite rows have been exposed.

This route makes the hub dependency visible before any Real-facing consumer treats separability as an available dense surface. It exports no selected nearest point, host dense subset, quotient real equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Lemma 1810.39 (RealSeparability countable dense route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that is exported as a countable dense surface factors through the countable dense subset boundary of chapter 1566, the dyadic-density witness surface of chapter 1484, and the rational-density witness surface of chapter 1409 before reaching any terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

The route reads the displayed rational enumeration, dyadic tolerance, finite $\mathsf{StreamName}^{\uparrow}$ *window, regular rational readback, and local* $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *rows as one finite dependency chain. It exports no host dense subset, selected nearest point, quotient real equality, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier to its rational enumeration, dyadic tolerance, finite window, regular readback, Real seal, exactness ledger, transport, replay, provenance, and local naming rows. The countable dense subset boundary supplies only the displayed indexed dense-source row, so the separability read first cites chapter 1566.

The tolerance and witness rows then pass through the two density siblings. The dyadic-density surface of chapter 1484 supplies the dyadic tolerance witness, and the rational-density surface of chapter 1409 supplies the rational interval witness.

Finally the existing $\mathsf{RealSeparability}^{\uparrow}$ route consumes those witnesses through the finite $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the terminal $\mathsf{Real}^{\uparrow}$ seal. Since every row is displayed in that chain, no host dense subset, selected nearest point, quotient equality, choice selector, or ambient completeness object is available. □

Theorem 1810.40 (RealSeparability countable dense window budget)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window read that is consumed as countable dense data factors through the same finite L10 source route:*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Concretely, the read must display the $\mathsf{DyadicRatCore}^{\uparrow}$ *rational-density row, the* $\mathsf{StreamName}^{\uparrow}$ *finite request schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the* BHist *transport ledger, the displayed* Cont *replay route, the* Pkg *provenance row, and the local* $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *row before any countable-dense consumer can use the window.*

The theorem does not assert full $\mathsf{Real}^{\uparrow}$ *completeness, a choice-selected enumeration of reals, a host dense subset, quotient real equality, or a selected nearest rational for each real name.*

Proof. Start with the countable dense route of Definition 1810.39. It already orders every exported dense surface through the countable dense subset boundary, dyadic-density witness surface, rational-density witness surface, finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Use the dyadic-density dependency and RegSeqRat schedule rows in this hub. They force each dense-window request to carry a displayed dyadic tolerance row and a finite stream request schedule before the regular rational readback is available.

Now route the read through the Real seal route. The $\mathsf{Real}^{\uparrow}$ seal is downstream of the same dyadic, stream, and regular-rational rows, and the structural BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows retain that dependency for the countable-dense consumer.

Thus the window budget is exactly the displayed L10 source route. Since none of those rows stores real completeness, a choice-selected enumeration, a host dense subset, quotient real equality, or nearest-rational selection, the exported countable dense read cannot claim any of those principles. □

Theorem 1810.41 (Real separability choice-free dense modulus)**.** *For every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet* $S = (Q, D, W, R, E, T, H, C, P, N)$ *and every finite precision request, the dyadic density row,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal produce a finite rational witness and modulus ledger without a selector over all reals. The witness is read from the displayed route*

$$Q, D, W, R, E, T, H, C, P, N$$

and not from a host dense subset, countable-choice schedule, quotient stream, selected nearest point, or ambient completeness principle.

Proof. Project the accepted carrier of Definition 1809.1. The precision request first reads the rational enumeration row Q and the dyadic tolerance row D.

Use the dyadic witness family of Definition 1809.7. It supplies the finite rational witnesses indexed by the displayed dyadic request and the finite StreamName window, not by a selector over all real points.

Push the witness through the StreamName and RegSeqRat boundary. Window stability keeps the read inside W, and the regular-rational readback R carries the same finite rational datum toward the Real-facing boundary.

Finally seal via E and record the modulus ledger through T, H, C, P, N. The rows only replay, transport, source, and name the finite density route, so no Classical.choice, quotient construction, host sequence selection, selected nearest point, or ambient completeness witness is introduced. □

The $\mathsf{RealSeparability}^{\uparrow}$ connected-local route supplies dense finite rational and dyadic reads to connected and locally connected consumers only after the source has passed through $\mathsf{Real}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$. A connected-space consumer may compare these reads with chapter 2353, while a local neighbourhood-component consumer may use the same finite density surface in chapter 5809. Thus the hub records a substantive horizontal dependency from real separability to connected-space and local-connectedness reads, without selecting a nearest rational, quotient real representative, host dense subset, path, arc, or ambient completeness principle. Every $\mathsf{RealSeparability}^{\uparrow}$ density consumer that reuses interval data first reads the rational interval carrier of chapter 354, then the dyadic-density comparison route of chapter 3513, and finally the regular rational readback surface of chapter 301. Density consumers needing explicit lower and upper rational brackets also pass through the locator route of chapter 4202 before the $\mathsf{Real}^{\uparrow}$-facing readback is exported. Thus the separability route exports a finite rational-interval, locator, and dyadic-density ledger rather than a detached dense subset, selected nearest point, quotient real equality, countable-choice schedule, or ambient completeness coordinate.

Theorem 1810.42 (Real separability closed bounded interval route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density route that feeds a closed bounded interval consumer factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the consumer named by chapter 1591 may inspect its interval certificate. The route exposes dyadic rational cells, regular rational readback, terminal Real seals, and the shared transport, replay, provenance, and local naming rows; it does not expose a selected dense point, quotient real equality, hidden endpoint chooser, open-cover compactness, or ambient completeness row.*

Proof. Start with the carrier of Definition 1809.1. Its density data is finite and is already attached to dyadic rational cells, regular rational readback, terminal Real seals, transport, replay, provenance, and local naming.

Route the density data to chapter 1591. A closed bounded interval consumer can use the separability packet only after the dyadic cell and the regular rational readback have been displayed.

The $\mathsf{Real}^{\uparrow}$ seal is then read as the terminal boundary of the same finite route. Since no coordinate of the separability packet contains a selected dense point, quotient equality, endpoint chooser, open-cover compactness, or ambient completeness row, the closed bounded interval handoff is exactly the displayed dependency route. □

Theorem 1810.43 (Real separability to closed bounded interval forward route)**.** *Every* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density packet reads the finite dense windows only through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route. In particular, the endpoint and containment consumers of chapter 1591 receive rational cells, dyadic tolerance rows, finite stream windows, regular rational readback, and terminal Real seals before any local interval row is inspected.*

The forward route exports no selected dense point, quotient real equality, ambient completeness row, hidden endpoint chooser, or open-cover compactness principle.

Proof. Start with the carrier of Definition 1809.1. Its finite density rows expose rational candidates together with the dyadic tolerance ledger, stream-window schedule, regular rational readback, terminal Real seal, componentwise transport, replay, provenance, and local naming.

Next read the interval-facing route through chapter 354. A closed bounded interval consumer can inspect endpoint and containment data only after the rational cell row has been displayed, and the dyadic refinement remains attached to that same finite rational ledger.

Now pass through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. They supply the finite endpoint windows, the regular rational readbacks, and the terminal Real seals that chapter 1591 may consume when forming its local interval certificate.

Since the separability packet contains no selected dense point, quotient real equality, ambient completeness row, hidden endpoint chooser, or open-cover compactness principle, the handoff is exactly the displayed finite route. □

Lemma 1810.44 (Real separability to uniform convergence forward link)**.** *Every* $\mathsf{UniformConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density packet first reads the* $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows displayed by the separability carrier. In particular, the consumer reaches the uniform-convergence boundary of chapter 3417 only after the finite dense source, dyadic tolerance ledger, stream-window schedule, regular rational readback, terminal Real seal, componentwise transport, replay, provenance, and local naming rows have all been exposed.*

The forward link exports no selected dense point, quotient real equality, host dense subset, countable-choice search, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the carrier of Definition 1809.1. Its rows $Q, D, W, R, E, T, H, C, P, N$ expose the rational source, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, dense-source row, transport, replay, provenance, and local naming.

Use the completion-facing and density routes already displayed by the separability chapter. They place the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary downstream of the dyadic and rational density rows, while the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows remain the only route by which Real-facing consumers inspect the dense packet.

Now pass to chapter 3417. A uniform-convergence consumer may reuse the density packet only through finite tolerance and window data, so it must read the same $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before reaching its own uniform boundary.

Since the separability carrier contains no host dense subset, selected dense point, quotient real equality, countable-choice search, or ambient completeness row, the forward link supplies only the displayed finite lattice edge. □

Every completion-facing read of a $\mathsf{RealSeparability}^{\uparrow}$ density packet that needs filter-completion data must pass through the finite rows already displayed by the separability hub before it reaches chapter 1569. The route first exposes the dyadic tolerance row, StreamName window, RegSeqRat readback, and terminal Real seal; only then may the downstream consumer inspect the Cauchy-filter completion packet.

Theorem 1810.45 (Real separability to CauchyFilterCompletion forward link)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet factors through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 1809.1 before the consumer may read the completion boundary named by chapter 1569. The handoff exports only finite density windows, dyadic tolerances, regular rational readback, terminal Real seals, transport, replay, provenance, and local naming; it exports no selected dense point, quotient filter, selected completion limit, host real equality, countable choice, or ambient completeness row.*

Proof. Project the carrier of Definition 1809.1. Its rows $Q, D, W, R, E, T, H, C, P, N$ expose the rational enumeration, dyadic tolerance, finite StreamName window, regular rational readback, terminal Real seal, dense-window exactness ledger, transport, replay, provenance, and local naming.

Use the existing dyadic, StreamName, RegSeqRat, and Real handoff route. A completion-facing consumer cannot reach completion data until the density request has passed through D, W, R, E and the exactness ledger T has tied those rows to the same finite rational read.

Now pass to chapter 1569. The completion packet reads Cauchy-filter source data, finite windows, dyadic tolerances, regular readback, and a terminal Real seal, so the separability packet can feed it only through the rows already displayed in the separability carrier.

Since neither carrier contains a coordinate for a selected dense point, quotient filter, selected limit, host equality, countable choice, or ambient completeness theorem, the forward link supplies exactly the finite horizontal route to the Cauchy-filter completion consumer. □

The formalization target is the finite handoff theorem for the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier rows $Q, D, W, R, E, T, H, C, P, N$. The statement should require the downstream $\mathsf{CauchyFilterCompletion}^{\uparrow}$ read to display, in order, the dyadic tolerance row D, the finite $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the exactness ledger T before any completion-facing row is consumed. Its conclusion is only that this displayed route feeds the completion packet through finite windows; it is not a selected dense point, quotient filter, selected completion limit, host equality, countable choice, or ambient completeness theorem. The $\mathsf{RealSeparability}^{\uparrow}$ source ledger consumes the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ route of chapter 1569 and the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ interval route of chapter 1591 through the same finite dense rational and dyadic witness rows. Concretely, the density request reads the rational candidate row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, the finite stream window, and the regular readback before the Cauchy-filter completion consumer sees a completion-facing dense witness, while the closed bounded interval consumer reads the same witness as an endpoint-contained interval-window row. Thus both source routes consume the displayed dense-window ledger, not a detached host dense subset, selected nearest point, quotient real equality, or ambient completeness principle.

Proposition 1810.46 (RealSeparability metric-completion reverse route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *consumer that uses* $\mathsf{RealSeparability}^{\uparrow}$ *dense data first reads the displayed* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. After those rows are exposed, the consumer may pass through chapter 1506 and chapter 1585 as a finite completion-facing route. The route exports no selected dense points, quotient* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the rational and dyadic rows from the separability dependency route before any completion-facing consumer receives dense data. The $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows identify the finite rational windows, while the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows keep the readback inside the displayed real-name corridor.

Then pass the displayed data through the metric-completion and reflector chapters named in the statement. Since the route has already exposed all dense rows and has no coordinate for selecting points, quotienting real names, invoking countable choice, or importing ambient completeness, every completion-facing use remains a finite BEDC route. □

The $\mathsf{RealSeparability}^{\uparrow}$ density packet exposes its regular-Cauchy-facing route only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301 and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291 have been displayed. Those rows feed the existing real-density and finite-window surfaces before any downstream regular Cauchy limit uniqueness consumer may inspect a terminal $\mathsf{Real}^{\uparrow}$ seal; the handoff exports no selected dense point, quotient real equality, countable-choice row, or ambient completeness coordinate.

Theorem 1810.47 (Real separability compact-continuous sup-norm forward link)**.** *Every compact-continuous sup-norm or uniform-modulus consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density packet first reads the displayed dyadic-density route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the compact-uniform surfaces of chapter 388 and chapter 369. In particular, the consumer may reach chapter 3807 only through those finite density, readback, window, Real-seal, compact-continuity, and modulus rows.*

The forward link exports no selected dense point, host dense subset, quotient Real equality, open-cover compactness, selected supremum, countable-choice search, or ambient completeness coordinate.

Proof. Project the carrier of Definition 1809.1. Its rows expose the rational source, dyadic tolerance ledger, StreamName windows, regular rational readback, terminal Real seal, density exactness, transport, replay, provenance, and local naming.

Use the existing dyadic-density and RegSeqRat routes. They keep every density request on the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ surface before any compact-continuous consumer can inspect the data.

Now pass to the compact-uniform route. The compact-continuity and uniform-modulus packets consume only displayed finite windows, modulus ledgers, and Real-sealed comparison rows, and the $\mathsf{ContinuousSupNorm}^{\uparrow}$ packet reads those rows as its finite bound source.

Thus the separability packet supplies a dependency edge to compact-continuous norm and modulus consumers only through the named BEDC rows. The excluded host dense subset, selected point, quotient equality, open-cover, supremum, choice, and completeness objects are absent from the route. □

Theorem 1810.48 (Real separability Cauchy-dense forward link)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *packet routes finite dyadic density witnesses through the dense Cauchy route of chapter 1453 and the Cauchy completion reflector of chapter 1585 before a downstream metric-completion consumer reads a countable dense surface. The read exposes the rational enumeration, dyadic tolerance row, finite stream window, regular rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RealSeparability}^{\uparrow}}$ *row; it exports no selected dense point, quotient completion class, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier to its rational enumeration, dyadic tolerance, finite window, regular readback, terminal Real seal, transport, replay, provenance, and local naming rows. The dyadic witness is therefore a finite row of the separability packet rather than a detached dense subset.

The dense-embedding consumer may read that witness only through chapter 1453. After this route has displayed the dense Cauchy source, the Cauchy completion reflector named by chapter 1585 supplies the completion-facing handoff.

Finally, componentwise hsame transport and Cont replay preserve the same displayed route into downstream metric-completion consumers. Since no row in the packet names quotient equality, a selected dense point, countable choice, or ambient completeness, none of those objects is exported by the forward link. □

Definition 1810.49 (Real separability RegSeqRat schedule)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *RegSeqRat schedule is the finite route row that presents a dense-read request through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal*

$\mathsf{Real}^{\uparrow}$ *seal in that order. A schedule instance is a* BHist *surface*

$$\Sigma = (S, R, D, E, H, C, P, N)$$

where S opens the finite stream window, R records the regular rational candidate read on that window, D fixes the dyadic tolerance used by the density request, E is the $\mathsf{Real}^{\uparrow}$ *seal reached only after those rows have been displayed, and H, C, P, N provide componentwise transport, replay, provenance, and local naming. The schedule has no coordinate for a host dense subset, selected nearest point, quotient real equality, countable choice, or ambient Real completeness.*

Theorem 1810.50 (Real separability RegSeqRat schedule handoff)**.** *Every dense-read consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet factors through the schedule of Definition 1810.49. In particular, the consumer first opens the finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, then reads the regular rational row of chapter 301, then fixes the dyadic tolerance row of chapter 291, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. No dense-read handoff exports a host dense subset, selected nearest point, quotient real equality, countable-choice schedule, or ambient Real completeness coordinate.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ density route. The local carrier already exposes density through finite observation rows, so the consumer cannot reach the Real-facing seal before the stream window, regular rational readback, and dyadic tolerance data are displayed.

The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite window S. The $\mathsf{RegSeqRat}^{\uparrow}$ row R then reads the rational candidate on that window, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D binds the tolerance used by the density request.

Only after those rows are fixed does the consumer reach the $\mathsf{Real}^{\uparrow}$ seal E. The structural rows H, C, P, N transport, replay, source, and name the same finite schedule.

Because the schedule contains no host dense subset, selected nearest point, quotient real equality, countable-choice schedule, or ambient completeness coordinate, every dense-read consumer factors through the displayed RegSeqRat schedule before separability is exported. □

Theorem 1810.51 (Real separability Cauchy realizer link)**.** *Every density read exported by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet to a* $\mathsf{CauchyRealizer}^{\uparrow}$ *consumer factors through the finite rational-window route of chapter 1771, chapter 301, chapter 126, and chapter 291. The read first displays a stream window, then a regular rational candidate, then the dyadic tolerance ledger used by the Cauchy realizer, and only then reaches the terminal Real-facing density seal. It exports no host dense subset, selected nearest point, quotient real equality, countable-choice schedule, or ambient Real completeness coordinate.*

Proof. Begin with the RegSeqRat schedule handoff of Definition 1810.50. It routes every density read through the finite $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal.

The $\mathsf{CauchyRealizer}^{\uparrow}$ consumer reads exactly that kind of finite request-to-window route by chapter 1771. Feeding the density read to the realizer therefore changes only the downstream consumer, not the rows inspected by the separability packet.

Since neither the schedule nor the Cauchy realizer chapter supplies host dense subsets, selected nearest points, quotient equality, countable choice, or ambient completeness as carrier coordinates, the linked route remains a finite rational-window handoff. □

The $\mathsf{RealSeparability}^{\uparrow}$ countable dense route is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ schedules of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, and the Cauchy-modulus and

majorant schedule of chapter 292 and chapter 4087 before any $\mathsf{Real}^{\uparrow}$ separability read becomes public.

Theorem 1810.52 (Real separability RegSeqRat and dyadic-density schedule dependency)**.** *Every countable dense route exported by an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet consumes the regular-Cauchy schedule through chapter 301, the dyadic-density witness surface of chapter 1484, and the terminal seal of chapter 38. The route is therefore the existing finite schedule through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicDensityReal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert rows; it exports no selected nearest point, host dense subset, quotient real equality, countable-choice selector, or ambient completeness coordinate.*

Proof. Use Definition 1810.49 to expose the finite schedule rows S, R, D, E, H, C, P, N. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback consumed by the density request, while the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row fixes the scale at which that request is inspected.

Next read the dyadic-density surface through chapter 1484. Its witness is attached to the same finite window and tolerance rows before the route reaches the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and local naming rows are the same rows already carried by the accepted $\mathsf{RealSeparability}^{\uparrow}$ packet. Since no carrier coordinate names a selected nearest point, host dense subset, quotient equality, countable-choice selector, or ambient completeness theorem, the schedule dependency is exactly the displayed RegSeqRat and dyadic-density route. □

Theorem 1810.53 (RealSeparability RegSeqRat density-window handoff)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read consumed by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing route factors through a finite* $\mathsf{StreamName}^{\uparrow}$ *window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximant row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the regular-Cauchy dense embedding route before the consumer may read the separability surface. The handoff is the displayed finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the regular-Cauchy dense embedding certificate; it exports no quotient stream equality, selected host dense subset, countable-choice selector, or ambient completeness coordinate.

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ packet through the existing schedule dependency of Definition 1810.52. The density request first opens a finite $\mathsf{StreamName}^{\uparrow}$ window and reads a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row at that window.

The $\mathsf{RegSeqRat}^{\uparrow}$ row then supplies the regular rational readback attached to the same finite window. This is the only route by which a rational approximant reaches the terminal $\mathsf{Real}^{\uparrow}$ seal in the separability packet.

Finally use the dense embedding route of chapter 1453. A completion-facing consumer receives the already displayed regular-Cauchy dense embedding certificate after the window, dyadic, and readback rows have been fixed.

Since the carrier has no coordinate for quotient stream equality, selected host dense subsets, countable-choice selectors, or ambient completeness, every such density read remains inside the finite density-window handoff. □

Theorem 1810.54 (RealSeparability RegSeqRat forward anchor)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that is forwarded to a Real-facing consumer first consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route of chapter 38. The read is therefore anchored in displayed regular-rational windows and Real seal data before the separability row is exported.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier to its density route. The consumer can inspect rational density only after the regular readback rows have exposed the finite $\mathsf{RegSeqRat}^{\uparrow}$ window data.

The same route then passes through the $\mathsf{Real}^{\uparrow}$ seal before any density observation is exported. Thus the forward read is a child route of the RegSeqRat and Real chapters, not a detached dense subset, quotient real equality, selected dense point, countable-choice schedule, or ambient completeness principle. □

The $\mathsf{RealSeparability}^{\uparrow}$ packet supplies countable dense readback to the $\mathsf{BishopCompleteMetric}^{\uparrow}$ completion surface of chapter 3424. A Bishop-complete-metric consumer first reads the separability carrier's dyadic density windows, $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal, then passes those finite rows to the complete-metric route already mediated by the existing Real separability completion links.

Theorem 1810.55 (Real separability to BishopCompleteMetric forward link)**.** *Every* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet factors through the displayed countable dense readback rows of Definition 1809.1 before it may read the complete-metric surface of chapter 3424. The handoff exports finite density windows, dyadic tolerances, regular rational readback, Real sealing, transport, replay, provenance, and local naming; it exports no selected dense point, quotient real equality, host completion object, countable-choice selector, or ambient completeness theorem.*

Proof. Project the Real separability carrier. Its dense-readback rows expose the rational enumeration, dyadic tolerance, StreamName window, regular rational readback, terminal Real seal, exactness ledger, transport, replay, provenance, and local naming.

Use the existing Real separability routes to metric completion and Cauchy-filter completion. Those child routes already force completion-facing reads through finite dyadic windows, regular readback, and the Real seal.

Now pass to chapter 3424. The Bishop-complete-metric packet consumes metric-completion, Cauchy-filter, regular readback, separated metric, complete-metric, and Real-seal rows, so Real separability can feed it only through the displayed countable dense readback surface.

Since neither packet contains selected dense points, quotient real equality, host completion objects, countable-choice selectors, or ambient-completeness coordinates, the link is exactly the finite forward handoff stated above. □

The $\mathsf{RealSeparability}^{\uparrow}$ dependency lattice also factors through the Bishop regular-real route of chapter 4058. A separability read that enumerates finite dyadic approximants must first expose the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the dyadic tolerance rows of chapter 291, and the regular-real handoff surface before treating the resulting $\mathsf{Real}^{\uparrow}$ seal as part of a countable dense family.

This keeps the separability route inside displayed L10 data: dyadic density supplies finite approximation addresses, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational names, and $\mathsf{BishopRegularReal}^{\uparrow}$ records the bounded regular real carrier that the separability certificate may consume. No host density theorem, quotient equality of real names, or ambient completion principle is introduced by this child route. The $\mathsf{RealSeparability}^{\uparrow}$ finite dyadic dense-family reads consume the Bishop regular-real handoff of chapter 4058, after the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301, the dyadic tolerance rows of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Thus the separability packet reaches its dense-family read through displayed L10 regular-sequence and dyadic data before it treats a Bishop regular real as a source for finite real-density evidence. The $\mathsf{RealSeparability}^{\uparrow}$ metric-density route reads its countable dense surface through the finite regular-rational and metric witnesses already exposed by sibling chapters. A density consumer first reads the $\mathsf{RegSeqRat}^{\uparrow}$ schedule of chapter 301, the

$\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{StreamName}^{\uparrow}$ finite windows; only then may it inspect the $\mathsf{Met}^{\uparrow}$ distance witnesses of chapter 57. Thus the separability packet exports a finite density readback over regular rational windows and metric distance rows, not a selected dense point, quotient real equality, countable-choice selector, host dense subset, or ambient completeness principle.

Theorem 1810.56 (Real separability metric density route)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density read that is consumed by a metric route factors through* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{Met}^{\uparrow}$ *distance witnesses before the local separability row is exported.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier through its rational enumeration, dyadic tolerance, finite stream-window, regular readback, terminal real-seal, transport, replay, provenance, and local naming rows. The metric consumer can read no dense datum before those rows have been displayed.

The regular rational schedule supplies the dense candidate as a finite readback rather than as a host dense subset. The dyadic tolerance and stream-window rows record the precision and observed entries used by that candidate.

The metric row is then read only as a displayed distance witness. Transport and replay preserve the same route, so the exported density surface remains finite and carries no selected nearest point, quotient real equality, countable-choice selector, or ambient completeness theorem. □

Lemma 1810.57 (RealSeparability Polish-space forward link)**.** *Every* $\mathsf{PolishSpace}^{\uparrow}$ *complete-separable consumer that reads a* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window route consumes the RealSeparability packet of chapter 1809 only after the* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal route of chapter 38 have been displayed. The resulting read is a forward edge into chapter 1490, not a selected dense point, quotient completion equality, countable-choice schedule, or ambient completeness claim.*

Proof. Project the $\mathsf{RealSeparability}^{\uparrow}$ density route through its real-facing sibling rows. The forward read first exposes finite stream windows, regular rational observations, dyadic tolerance checks, and the terminal real seal before the dense-window data are exported.

The $\mathsf{PolishSpace}^{\uparrow}$ carrier then consumes that route through its own S, R, W observation ledger and complete-separable boundary. Since neither carrier has a coordinate for selected dense points, quotient completion equality, countable choice, or ambient completeness, the forward link preserves only the displayed finite rows. □

Theorem 1810.58 (RealSeparability Polish forward criterion)**.** *A* $\mathsf{PolishSpace}^{\uparrow}$ *consumer may read* $\mathsf{RealSeparability}^{\uparrow}$ *only when the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal route, and the dense-window route of* $\mathsf{RealSeparability}^{\uparrow}$ *are all displayed. The criterion exports no completeness theorem, no choice principle, no selected dense point, and no quotient-completion equality.*

Proof. Project the forward link of Definition 1810.57. It already orders the PolishSpace read through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, and the terminal Real seal before the dense-window route is consumed.

The local RealSeparability route files identify the same dense-window data as finite BEDC rows. Repacking those rows gives the stated criterion for PolishSpace consumers, while the excluded completeness, choice, selected-point, and quotient-completion coordinates remain absent from the displayed carrier surfaces. □

Theorem 1810.59 (RealSeparability dense Real forward link)**.** *Every accepted* RealSeparability$^{\uparrow}$ *dense finite schedule that exposes* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows feeds both the* PolishSpace$^{\uparrow}$ *countable dense surface of chapter 1490 and the* SeparatedCompletion$^{\uparrow}$ *zero-distance completion route of chapter 1612. The forward link carries only displayed finite schedules, stream windows, regular rational readbacks, dyadic tolerances, and Real seals; it does not select dense points, invoke countable choice, identify quotient completion classes, or use ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted RealSeparability$^{\uparrow}$ schedule. Its dense rows are finite: a consumer sees named windows, regular rational readbacks, dyadic tolerance comparisons, and terminal Real seals before any dense-surface claim is public.

For the PolishSpace branch, read the same finite schedule as a countable dense surface. The route supplies a displayed dense inventory to the PolishSpace$^{\uparrow}$ consumer, but no point is selected outside the schedule and no countable-choice operator is introduced.

For the SeparatedCompletion branch, pass the Real-seal comparisons through the zero-distance completion route. The SeparatedCompletion$^{\uparrow}$ consumer reads the same StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ evidence before the separated classifier is available.

The two branches therefore share one dense Real source. Since the carrier has no coordinate for chosen dense points, quotient completion equality, countable choice, or ambient completeness, the forward link is exhausted by the displayed finite RealSeparability route. □

Theorem 1810.60 (Real separability Cauchy approximation-tree forward link)**.** *Every* CauchyApproximationTree$^{\uparrow}$ *consumer that reads a countable dense* RealSeparability$^{\uparrow}$ *surface does so through the finite dense-readback route named by this chapter. The consumed branch window first reads the* DyadicRatCore$^{\uparrow}$ *tolerance row of chapter 291, then the finite* StreamName$^{\uparrow}$ *observation surface of chapter 126, then the* RegSeqRat$^{\uparrow}$ *readback row of chapter 301, and finally the* Real$^{\uparrow}$ *seal of chapter 38 before entering the finite tree packet of chapter 5716. Thus the dense route exports finite approximation-tree windows, not a detached dense subset, a selected real representative, quotient stream equality, countable choice, or ambient completeness.*

Proof. Use the dense readback rows already displayed by RealSeparability$^{\uparrow}$. They expose countable dense data only through dyadic tolerances, finite stream observations, regular rational readbacks, and Real seals.

Now read the CauchyApproximationTree$^{\uparrow}$ carrier. Its branch windows consume exactly those four rows before a Real-facing endpoint is sealed, so the dense surface is passed forward as finite tree-window data.

The carrier has no coordinate for detached dense subsets, selected representatives, quotient stream equality, countable choice, or ambient completeness. The forward link therefore preserves the concrete dense readback surface and adds no independent completion principle. □

The countable dense RealSeparability$^{\uparrow}$ rows are consumed by the PolishSpace$^{\uparrow}$ root surface of chapter 1490 only after the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ readbacks are displayed. This route supplies a finite consumer edge from real separability to the Polish complete-separable package, but it does not provide selected dense points, quotient equality, or ambient completeness. The countable dense rows of RealSeparability$^{\uparrow}$ give sequential consumers a displayed source of rational and regular-Cauchy probes before any topology or limit read is accepted. A SequentialSpace$^{\uparrow}$ packet may consume those rows through chapter 4112 only after the finite density route has passed through the StreamName$^{\uparrow}$ and LimitSpace$^{\uparrow}$ surfaces of chapter 126 and chapter 3845. When the same read is completion-facing, the CauchySpace$^{\uparrow}$ comparison boundary of chapter 1847 remains

part of the route, so the dense source is not reread as a selected limit, quotient real equality, countable-choice selector, or ambient completeness principle. The $\mathsf{RealSeparability}^{\uparrow}$ Cauchy-setoid modulus route records the density-side dependency that keeps regular rational names comparable before a countable dense handoff is claimed. It reads chapter 301 for finite regular windows, chapter 1820 for the classifier that identifies equivalent regular Cauchy presentations, and the RealUp L10 input factorization before exporting a density route.

Theorem 1810.61 (Real separability Cauchy setoid route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that consumes* $\mathsf{RegSeqRat}^{\uparrow}$ *regular names factors through the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier before it claims a countable dense route. The consumer may then use the RealUp L10 route, but only after the regular readback, Cauchy-equivalence classifier, dyadic tolerance, terminal seal, transport, replay, provenance, and local naming rows are visible; it exports no quotient real equality, selected dense enumeration, host equality, countable choice, or ambient completeness.*

Proof. Start with the regular rational names. A density read that consumes $\mathsf{RegSeqRat}^{\uparrow}$ data first opens the finite windows and dyadic tolerances described by chapter 301; those rows are not yet a Real-facing countable dense route.

Next pass through the Cauchy-equivalence classifier of chapter 1820. Equivalent regular Cauchy presentations replay the same density read only through the displayed classifier row, so the consumer cannot replace the route by quotient real equality or a selected representative.

Finally use the RealUp L10 input factorization. The factorization keeps the terminal Real seal downstream from finite stream windows, regular readback, and dyadic tolerance data, so the countable dense route is claimed only after the setoid comparison has already fixed the regular-name boundary. □

The RealSeparability dense-modulus route records the classifier boundary that a regular Cauchy dense-read consumer must cross before a displayed dense modulus is exported. It reads the Cauchy-equivalence classifier of chapter 1820 together with the existing RealSeparability Cauchy-setoid and dense-read routes, including Definition 1810.61 and Definition 1810.21.

Theorem 1810.62 (RealSeparability Cauchy-setoid dense-modulus route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-read consumer that uses a regular Cauchy name must first consume the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{Real}^{\uparrow}$ *seal before exporting the dense modulus. The exported modulus is therefore a finite classifier-mediated RealSeparability route, not quotient real equality, host stream equality, selected dense enumeration, countable choice, or ambient completion.*

Proof. Start with the regular Cauchy name boundary. The classifier route of chapter 1820 compares regular presentations through finite displayed rows, so the dense-read consumer cannot begin from a quotient real object or a selected representative.

Next compose with the existing RealSeparability Cauchy-setoid route of Definition 1810.61. It places the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before any countable dense handoff and keeps the comparison inside the displayed regular-name boundary.

Finally use the dense-read route of Definition 1810.21. The consumer then reads $\mathsf{RegSeqRat}^{\uparrow}$ regular readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{StreamName}^{\uparrow}$ windows, and the terminal $\mathsf{Real}^{\uparrow}$ seal before the dense modulus is exported, leaving quotient equality, host equality, selected dense enumerations, countable choice, and ambient completion outside the packet. □

The $\mathsf{RealSeparability}^{\uparrow}$ countable-dense route depends on the classifier surface of chapter 1820 before dense readback can be consumed. This child route pins the separability lattice to the dis-

played Cauchy-equivalence setoid rows, so the dense enumeration reads a finite classifier boundary rather than quotient path equality or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 1810.63 (Real separability Cauchy-setoid forward route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-read consumer that identifies regular Cauchy names must expose the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row before the* $\mathsf{Real}^{\uparrow}$ *separability seal is exported. The route passes through the regular rational readback and dyadic tolerance rows already visible in Definition 1810.61 and the local Real-separability carrier, then reaches chapter 1820 as a classifier boundary rather than as quotient stream equality.*

Proof. Project the Real-separability dense-read rows from Definition 1809.1. The consumer first reads the displayed density schedule, StreamName windows, RegSeqRat readback, dyadic tolerance row, terminal Real seal, transport, replay, provenance, and local naming.

Apply the Cauchy-setoid route of Definition 1810.61. It places the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier between regular Cauchy readback and the public dense-route handoff, so the consumer cannot export the $\mathsf{Real}^{\uparrow}$ separability seal before the classifier row is visible.

Finally use the sibling chapter chapter 1820. Its carrier supplies finite StreamName, RegSeqRat, DyadicRatCore, setoid-test, and Real-seal rows, not a quotient stream equality, selected representative, countable-choice selector, host real equality, or ambient completeness witness. □

The $\mathsf{RealSeparability}^{\uparrow}$ density route reads finite regular windows through chapter 301 and identifies comparable presentations through chapter 1820 before exporting countable dense readback. $\mathsf{RealSeparability}^{\uparrow}$ consumes the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ routes of chapter 301 and chapter 126; its displayed dense source is the finite rational row family supplied by $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$, with chapter 291 fixing the dyadic tolerance ledger used by each L10 read. The same dense dyadic and rational source supplies downstream $\mathsf{Pseudometric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ distance-read routes through chapter 1614 and chapter 1506.

Theorem 1810.64 (RealSeparability L10 source route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer that reads finite dense-window data through the L10 source chain must display the same* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows cited by chapter 126, chapter 301, chapter 291, and chapter 38. The route exposes finite stream windows, regular rational sequence readback, dyadic tolerance control, and the terminal Real seal before any separability-density handoff is consumed.*

Proof. Project the RealSeparability packet through its displayed density route. The source side can read only finite window data, so the first visible dependency is the $\mathsf{StreamName}^{\uparrow}$ row named by the sibling source chapter.

The window data are then interpreted through $\mathsf{RegSeqRat}^{\uparrow}$ and bounded by the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. These rows supply the finite regular-name and dyadic-readback surface used by separability consumers.

The terminal read is the $\mathsf{Real}^{\uparrow}$ seal. Since the route has no coordinate for quotient real equality, ambient completeness, or a choice-selected dense sequence, the L10 source chain is exhausted by the four displayed sibling chapters. □

Theorem 1810.65 (RealSeparability countable-dense Cauchy handshake)**.** *Every* $\mathsf{CountableDenseSubset}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSeparability}^{\uparrow}$ *packet must first display finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance control, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and only then read rational density. The handshake routes the dense subset surface of chapter 1566 through the L10 source chain of Definition 1810.64; it exports no selected dense sequence, quotient stream equality, ambient completeness, or host enumeration of completed reals.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier and apply Definition 1810.64. The source route already forces finite $\mathsf{StreamName}^{\uparrow}$ windows to be visible before any density read is consumed.

The windows are then read back by $\mathsf{RegSeqRat}^{\uparrow}$ and bounded by the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. These rows supply the finite Cauchy-facing data needed by the countable dense subset consumer.

Only after the terminal $\mathsf{Real}^{\uparrow}$ seal is displayed may the consumer read rational density from chapter 1566. Since no selected dense sequence, quotient stream equality, ambient completeness, or host enumeration row appears in the displayed route, none can be exported by the handshake. □

Theorem 1810.66 (RealSeparability nested-set density handoff)**.** *Every nested located closed-set packet over* $\mathsf{Real}^{\uparrow}$ *that consumes rational density must expose the* $\mathsf{RealSeparability}^{\uparrow}$ *countable dense row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it passes diameter-modulus data to completion consumers. In particular, the handoff to chapter 7594 is routed through the L10 source chain of Definition 1810.64 and the countable-dense handshake of Definition 1810.65; it exports no sequential compactness, open-cover compactness, countable-choice selector, quotient stream equality, or ambient completeness row.*

Proof. First read the density source through Definition 1810.64. It forces a density consumer to expose finite $\mathsf{StreamName}^{\uparrow}$ windows before the rational density row is consumed.

Next use Definition 1810.65. The handshake routes the countable dense surface through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any downstream consumer treats density as Real-facing data.

For a nested located closed-set packet, the diameter-modulus row is downstream of the same finite density surface. Thus the completion-facing handoff can pass only the displayed countable dense row, dyadic approximation row, regular rational readback row, Real seal, transport, replay, provenance, and local naming rows.

Since none of these rows contains sequential compactness, open-cover compactness, countable choice, quotient stream equality, or ambient completeness, the nested-set consumer cannot export any of those claims. □

The dense dyadic schedule of $\mathsf{RealSeparability}^{\uparrow}$ also consumes the located-comparison readback of chapter 1730 and the rate-extraction route of chapter 2085. The schedule is therefore not a free host enumeration: it is a finite $\mathsf{NameCert}$ ledger whose comparison cells are inherited from $\mathsf{LocatedRealComparison}^{\uparrow}$ and whose precision rates are inherited from $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ before any dense real-name consumer can read them. Every $\mathsf{RealSeparability}^{\uparrow}$ dense-countable read that is consumed by a located real completion argument first passes through the finite $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ comparison surface of chapter 1820 and the $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ route of chapter 6045. The separability packet exports dense reads only after the displayed regular-name comparison, located Cauchy completeness handoff, hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows have been shown; it exports no quotient real equality, selected dense enumeration, countable choice, host completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completion. The child dependency route for $\mathsf{RealSeparability}^{\uparrow}$ is the finite L10 path through chapter 301 and chapter 38: a density request first reads the existing separability schedule rows of Definition 1810.25, then consumes the dense dyadic and rational rows of Definition 1809.6, and only after those rows reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. Thus the hub surface records a substantive horizontal dependency from separability to regular-rational readback and Real sealing, with no route through a host dense subset, selected nearest point, quotient stream, or ambient completeness claim.

Theorem 1810.67 (Real separability L10 certificate route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read that consumes the* $\mathsf{Real}^{\uparrow}$ *seal factors through the displayed L10 route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{CountableDenseSubset}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route is grounded in chapter 301, chapter 126, chapter 291, chapter 37, chapter 1566, and chapter 38. Thus a separability consumer receives regular-rational readback, finite stream windows, dyadic tolerance rows, rational endpoint rows, countable dense subset naming, and the terminal Real seal as displayed sibling data, with no host dense subset, selected nearest point, quotient real equality, countable-choice enumeration, or ambient completeness coordinate.

Proof. Read the $\mathsf{RealSeparability}^{\uparrow}$ consumer as a request for a displayed density route. The regular-rational row supplies the source readback, while the stream row supplies the finite windows on which the density request is inspected.

The dyadic and rational rows then expose the tolerance and endpoint data used by the dense subset certificate. The $\mathsf{CountableDenseSubset}^{\uparrow}$ row names the countable dense boundary as a sibling certificate rather than as an external host enumeration.

Only after those rows are displayed can the consumer reach the $\mathsf{Real}^{\uparrow}$ seal. Since the route has no coordinate for a host dense subset, selected nearest point, quotient equality, choice enumeration, or ambient completeness, none of those objects can be exported by the separability packet. □

The $\mathsf{RealSeparability}^{\uparrow}$ route reads dense rational windows through chapter 301 and dyadic tolerances through chapter 291 before the terminal $\mathsf{Real}^{\uparrow}$ seal. This places the L10 dependency on displayed regular-rational schedules and dyadic modulus rows, rather than on an external countable dense subset or a choice-selected enumeration. The $\mathsf{RealSeparability}^{\uparrow}$ source lattice is not a kernel-only island: its rational enumeration is read from chapter 37, its dyadic tolerance route is read from chapter 291, and its density handoff reaches $\mathsf{CountableDenseSubset}^{\uparrow}$ only after the StreamName, RegSeqRat, and Real route has displayed finite windows, regular rational readback, and the terminal real-name seal. Every $\mathsf{RealSeparability}^{\uparrow}$ density read that reaches the public countable-dense boundary factors through $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{DyadicDensityReal}^{\uparrow}$, and $\mathsf{RationalDensityReal}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is available. The route is grounded in chapter 1566, chapter 1484, chapter 1409, and chapter 38, while the L10 source lattice remains the finite chain through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$; no host dense subset, selected nearest point, quotient real equality, countable choice, or ambient completeness row is exported by this seal route.

Theorem 1810.68 (RealSeparability countable dense choice-free filter)**.** *An accepted* $\mathsf{RealSeparability}^{\uparrow}$ *dense read is exported as a countable dense filter only after the dyadic rational candidate row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are all displayed. The route exports the countable dense filter ledger carried by those rows and exports no selected representative, quotient stream equality, trichotomy, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted separability carrier to its finite density rows. The dyadic-density dependency of Definition 1810.28 supplies the dyadic rational candidate and tolerance ledger through $\mathsf{DyadicRatCore}^{\uparrow}$, so the dense read begins from displayed rational data rather than from a selected host representative.

Next use the countable dense route of Definition 1810.39. Its schedule is finite-window data, and the $\mathsf{StreamName}^{\uparrow}$ row records exactly the windows inspected by the density ledger.

Finally apply the RegSeqRat pullback of Definition 1810.26 and the countable-dense seal route immediately above this theorem. These bind the regularity readback and the terminal $\mathsf{Real}^{\uparrow}$ seal before the filter is public. Since no row in the chain carries quotient stream equality, trichotomy, countable choice, or ambient completeness, the exported object is precisely the displayed countable dense filter ledger. □

Theorem 1810.69 (RealSeparability regular Cauchy dense countable route)**.** *Every accepted* $\mathsf{RealSeparability}^{\uparrow}$ *density request that reaches a countable dense row first factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The request reads dyadic tolerance rows, finite stream windows, regular rational readback, the Cauchy-equivalence classifier, and the terminal real-name seal before the countable dense row is consumed. The route exports no quotient $\mathsf{Real}^{\uparrow}$ *equality, selected dense enumeration, countable-choice branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with an accepted density request for $\mathsf{RealSeparability}^{\uparrow}$. The hub route already places such a request downstream of the dyadic rational core, stream-name window, regular-rational readback, Cauchy-equivalence classifier, and terminal real-name seal.

The dyadic and stream rows provide the finite observation data. The $\mathsf{RegSeqRat}^{\uparrow}$ readback attaches those observations to a regular Cauchy name, and the Cauchy-equivalence classifier is the only equality classifier inspected before the name reaches the $\mathsf{Real}^{\uparrow}$ seal.

Only after that seal is fixed may the countable dense row be read. Since none of the displayed rows contains quotient real equality, a selected dense enumeration, countable choice, or an ambient completeness coordinate, the countable-density export remains exactly the finite regular-Cauchy route stated above. □

Every $\mathsf{RealSeparability}^{\uparrow}$ density read that is forwarded to a public dense-source consumer first consumes $\mathsf{CountableDenseSubset}^{\uparrow}$ through chapter 1566, $\mathsf{DyadicDensityReal}^{\uparrow}$ through chapter 1484, and $\mathsf{RationalDensityReal}^{\uparrow}$ through chapter 1409. Only after those sibling rows have been displayed may the route pass through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal; it exports no host dense subset, selected nearest point, quotient real equality, countable choice, or ambient real completeness. The $\mathsf{RealSeparability}^{\uparrow}$ route supplies the countable dense schedule read by the complete-separable consumers of chapter 1490 and chapter 1458. A Polish-space or separable-metric consumer reaches that schedule only after the Real-facing density packet has exposed its $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic tolerance rows, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row. The dependency is therefore a finite L10 dense-schedule handoff, not a host dense subset, selected nearest point, quotient real equality, countable-choice selector, or ambient completeness claim.

Theorem 1810.70 (Real separability lawless-sequence forward link)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *density consumer that reuses lawless or Baire finite-prefix observations must pass through chapter 1495, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the dyadic tolerance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it exports separability. For an accepted separability carrier*

$$S = (Q, D, W, R, E, T, H, C, P, N),$$

the lawless or Baire observation is admitted only as a finite prefix read routed through W*, tested against* D*, transported through* R*, and sealed by* E*.*

The route refuses host infinite sequences, countable choice, quotient streams, selected dense points, and $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted $\mathsf{RealSeparability}^{\uparrow}$ carrier. The density consumer has access only to the rational enumeration row Q, dyadic tolerance row D, finite StreamName window W, RegSeqRat readback R, terminal Real seal E, exactness ledger T, transport H, replay C, provenance P, and local naming N.

If the consumer cites a lawless or Baire finite-prefix observation, the citation first enters through chapter 1495. That chapter exposes only finite StreamName windows, Bool digit rows, Nat prefix indices, transport, replay, provenance, and local naming; it does not export an infinite sequence or a countable-choice source.

Compose that finite-window scope with the separability density route. The cited prefix window is read as the separability row W, the requested precision is read through the dyadic row D, the observed rational data are transported through the RegSeqRat row R, and the terminal Real boundary is reached only at E.

The rows T, H, C, P, N preserve the same route as exactness, componentwise transport, displayed replay, provenance, and local naming. Since neither packet contains host infinite sequences, countable choice, quotient streams, selected dense points, or ambient Real completeness, the composed density consumer cannot export any of them. □

The $\mathsf{RealSeparability}^{\uparrow}$ route consumes the Cauchy readback boundary of chapter 3413 and the completion universal-property boundary of chapter 2124 before exporting dense countable real reads. This pins the separability packet to its Cauchy-window source and completion-facing universal route, rather than letting the hub appear as a kernel-only index. The $\mathsf{RealSeparability}^{\uparrow}$ density windows used by compact interval or uniform-continuity consumers pass through chapter 5863 when the consumer asks for a finite prefix bar. The fan route supplies the finite window bound; ordinary dyadic density still comes from the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 and the dyadic arithmetic route of chapter 291. Thus the separability lattice distinguishes density supply from window certification without introducing a selected dense subsequence, open-cover compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The dense-window reads exported by $\mathsf{RealSeparability}^{\uparrow}$ consume the finite cofinal tail-window service of chapter 4060 before presenting rational density windows to $\mathsf{Real}^{\uparrow}$ consumers. This route keeps the separability dossier tied to displayed regular-Cauchy windows, cofinal tail schedules, rational-density rows, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert naming rather than to an ambient countable dense subset or a selected completion limit.

Theorem 1810.71 (RealSeparability hub sibling route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density read exported by chapter 1809 factors first through the regular rational schedule of chapter 301, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, and the terminal real-name seal of chapter 38. Only after those sibling rows are displayed may the packet expose a countable dense surface to a downstream consumer.*

Proof. Read the existing RealSeparability carrier and sibling-lattice files through the hub. The regular rational schedule gives the rational approximants, the dyadic core gives the tolerance rows, and the stream-name route gives the finite observation windows that the density read may inspect.

The terminal $\mathsf{Real}^{\uparrow}$ seal is then read before the density surface is exported. Thus the hub route records a concrete L10 dependency through RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, and RealUp rather than a detached dense subset, selected representative, quotient real equality, countable-choice selector, or ambient completeness row. □

Theorem 1810.72 (RealSeparability countable-dense forward read)**.** *Every* $\mathsf{CountableDenseSubset}^{\uparrow}$ *or* $\mathsf{RealCauchyModulus}^{\uparrow}$ *consumer reading* $\mathsf{RealSeparability}^{\uparrow}$ *must pass through the sibling route*

named in Definition 1810.71, then through the countable-dense source of chapter 1566 before exporting density. The route exports no selected dense point, quotient real equality, countable-choice selector, or ambient completeness row.

Proof. Start with Definition 1810.71. It places every density read behind the RegSeqRat, DyadicRatCore, StreamName, and RealUp sibling rows before the hub exports a countable dense surface.

Project the carrier rows of Definition 1809.1. The rational enumeration, dyadic tolerance, finite window, regular readback, terminal Real seal, exactness, transport, replay, provenance, and local naming rows are the only data available to the separability consumer.

The countable-dense source of chapter 1566 is therefore read after those sibling rows and before density is exported to either a $\mathsf{CountableDenseSubset}^{\uparrow}$ or $\mathsf{RealCauchyModulus}^{\uparrow}$ consumer. Since the carrier contains no selected dense point, quotient real equality, countable-choice selector, or ambient completeness row, the forward read cannot export any of them. □

Theorem 1810.73 (RealSeparability Cauchy setoid forward link)**.** *Every countable-dense* $\mathsf{RealSeparability}^{\uparrow}$ *route that exports rational names to a Real-facing consumer supplies a forward read of the Cauchy equivalence classifier in chapter 1820. Dense rational names remain compared through the displayed setoid ledger after the RegSeqRat, DyadicRatCore, StreamName, and RealUp sibling rows are exposed, rather than by a separate real-name equality predicate.*

Proof. Begin with Definition 1810.72. It forces each density export through the regular rational schedule, dyadic tolerance ledger, finite observation windows, terminal Real seal, and countable-dense source.

Now read the Cauchy equivalence setoid boundary. The same rational names that witness density are compared by the displayed classifier ledger of chapter 1820, so the separability route has no extra coordinate for host real equality or a separate real-name equality predicate. □

Theorem 1810.74 (Real separability public status route)**.** *Every public* $\mathsf{RealSeparability}^{\uparrow}$ *density read is exported only after the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal are displayed. The route presents a finite density surface for Real and Completion consumers; it exports no quotient real equality, selected dense enumeration, countable choice, hidden host dense subset, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the density rows already routed through chapter 1809. A public consumer may read a candidate only as a finite density witness, not as a selected host enumeration.

Next pass the witness through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows. These rows expose the finite observation windows and regular-rational readback before any Real-facing density claim is public.

The $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger and the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier then fix the comparison boundary for the read. Equivalent regular presentations replay the same displayed density route rather than becoming quotient real equality.

Only after those rows are visible does the consumer reach the terminal $\mathsf{Real}^{\uparrow}$ seal. Since no row contains a selected dense enumeration, countable-choice schedule, hidden host dense subset, quotient real equality, or ambient completeness principle, the public status route is exhausted by the displayed finite data. □

Theorem 1810.75 (RealSeparability rotation-selected public route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *public-route consumer selected by the rotation frontier must pass through* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *the countable-dense seal, transport, replay, provenance, and local naming before any separability read is exported. The route exposes the finite Cauchy classifier, dyadic tolerance, stream window, regular-rational readback, terminal Real seal, and countable dense support; it exports no quotient Real equality, selected dense point, countable-choice selector, or ambient completeness coordinate.*

Proof. Read the hub route through Definition 1810.74. Its public status row already orders the density request through the finite L10 source route before a public countable-dense read is available.

Next use the Cauchy-equivalence and countable-dense child routes. They force the consumer to display the CauchyEquivalenceSetoid$^{\uparrow}$ classifier, dyadic tolerance, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, terminal Real$^{\uparrow}$ seal, transport, replay, provenance, and local naming before the separability read is exported.

Since the route names only those displayed rows, the refused quotient equality, selected dense point, countable-choice selector, and ambient completeness coordinates have no public export path. □

Every RealSeparability$^{\uparrow}$ countable-density read consumes the RationalDensityReal$^{\uparrow}$ route of chapter 1409 before a rational witness can be treated as dense in the displayed real carrier. The sibling route keeps rational interval witnesses, finite stream windows, RegSeqRat$^{\uparrow}$ readback, and the terminal Real$^{\uparrow}$ seal inside the L10 Real packet, with no host dense subset, selected nearest point, quotient real equality, countable choice, or ambient completeness row. The RealSeparability$^{\uparrow}$ packet supplies countable dense dyadic reads for the same finite Real-facing corridor used by effective finite-cover compactness and compact-uniform modulus routes. Its dense readback surface is grounded in the existing L10 siblings chapter 301 and chapter 38, and it is consumed by chapter 6035 as the dyadic observation side of a finite-cover handoff.

Theorem 1810.76 (Real separability dyadic lattice forward route)**.** *Every accepted* RealSeparability$^{\uparrow}$ *dense-read route that exports a countable dense ledger factors first through the dyadic rational density rows of chapter 291, then through the regular rational readback surface of chapter 301, and finally through the terminal real-name seal of chapter 38. The exported ledger is therefore a displayed dyadic, regular-sequence, and Real-sealed route, not a host countability claim, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Read the existing RealSeparability child routes. The density route and dyadic lattice link expose rational candidates only after the dyadic tolerance rows have fixed the finite approximation scale.

The RegSeqRat forward route then supplies the regular rational readback. It keeps the StreamName window and dyadic tolerance visible before a dense-completion or countable-density consumer can inspect the exported row.

Finally the Real seal route closes the read at Real$^{\uparrow}$. Since the hub child routes carry no host countability oracle, quotient equality token, or ambient completeness coordinate, the countable dense ledger is exported only through the displayed dyadic, RegSeqRat, and Real-sealed dependency chain. □

RealSeparability$^{\uparrow}$ supplies the countable dense real-window rows consumed by the compact-uniform-continuity finite-net route. The downstream packet of chapter 388 may read those windows only after the Real$^{\uparrow}$ seal of chapter 38, the RegSeqRat$^{\uparrow}$ regularity schedule of chapter 301, and the DyadicRatCore$^{\uparrow}$ observation rows of chapter 291 remain visible through the RealSeparability$^{\uparrow}$ dense-window surface. The edge therefore contributes a finite source ledger for compact finite-net

consumers, not an ambient separability theorem or a choice of all real points. The $\mathsf{RealSeparability}^{\uparrow}$ separability packet consumes the equality-facing boundary of chapter 1820 before it reads the countable dense real route through the regular-rational readback of chapter 301 and the dyadic tolerance core of chapter 291. Thus a dense-read consumer first identifies regular-Cauchy presentations through $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, then reads the dyadic density windows exposed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$. The route exports no host countable choice principle and no selected enumeration outside the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows. The $\mathsf{RealSeparability}^{\uparrow}$ hub reads the classifier boundary of chapter 1820 as the sibling surface that supplies separability readback for regular Cauchy presentations; this ledger line exposes the local dependency without adding a separate theorem claim. The $\mathsf{RealSeparability}^{\uparrow}$ route consumes the CauchyEquivalenceSetoid classifier of chapter 1820 together with the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 when it turns countable dyadic density into a Real-facing separability packet. This link records that separability is read through the setoid classifier and regular-rational presentation before it is exposed to Real consumers.

Theorem 1810.77 (Real separability StreamName witness route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *countable dense witness consumed by an L10* $\mathsf{Real}^{\uparrow}$ *route must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before it is read as density data. The witness route is grounded in chapter 126, chapter 301, and chapter 38; when the dense witness is compared as a regular Cauchy presentation, it reads density only through the Cauchy-equivalence classifier route of chapter 1820 before the Real seal. It supplies no host dense subset, selected enumeration, quotient real equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with the L10 certificate route of Definition 1810.67. It places the density read on displayed sibling rows rather than on a host dense subset or selected enumeration.

The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite windows at which the candidate dense witness is inspected. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback attached to those windows, so the density datum is not available as a detached rational list.

When a consumer compares the resulting regular presentation, the comparison passes through the Cauchy-equivalence classifier of chapter 1820. This keeps density attached to displayed classifier rows before the Real seal is available.

Only after those two rows have been displayed can the route consume the terminal $\mathsf{Real}^{\uparrow}$ seal. Therefore every L10 separability witness read exposes StreamName windows, RegSeqRat readback, and the Real seal before it is read as density data, with all excluded choice and quotient coordinates remaining absent. □

Theorem 1810.78 (Real separability seed registration)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *seed route is registered by the displayed finite lattice*

$$\mathsf{CountableDenseSubset}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Every dense-read consumer of the seed must first expose countable-dense source rows, dyadic tolerance rows, finite stream windows, regular-rational readback, and the terminal Real seal before any separability datum is exported. The hub marker points to the seed-status theorem Definition 1810.12, while this child records the finite registration text that keeps the $\mathsf{RealSeparability}^{\uparrow}$ *route at the same scoped marker site. The registration claims only seed-level routing; it does not claim scoped closure, selected dense points, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the countable-dense route from the existing child lattice. The separability packet exposes density through countable source rows and dyadic tolerance rows before a Real-facing consumer can inspect a dense witness.

Next use the StreamName and RegSeqRat sibling routes. They place the finite observation window and regular-rational readback between the dyadic tolerance ledger and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The seed-status child records the resulting closure axis at seed level. Therefore this registration is a finite route marker for $\mathsf{RealSeparability}^{\uparrow}$ and not a stronger separability theorem or a completeness principle. □

Theorem 1810.79 (Real separability seed child route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *density request factors through the displayed child route consisting of a* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any countable-dense row is consumed. The route is the seed-level child of Definition 1810.12, Definition 1810.14, and Definition 1810.15; it exports no quotient Real equality, selected dense enumeration, countable choice, host dense subset, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the seed status Definition 1810.12. It identifies the separability seed as the displayed child-route family rather than theorem content stored on the hub.

Apply the scoped density route Definition 1810.14. The dyadic approximation row, finite StreamName windows, RegSeqRat readback, Cauchy-equivalence comparison, and Real seal are exposed before the countable-dense row can be read.

Finally use the seed closure route Definition 1810.15. It records the same child ledger as the seed closure surface, so the density request remains inside the finite route and cannot supply quotient Real equality, selected enumeration, countable choice, a host dense subset, or ambient completeness. □

Theorem 1810.80 (RealSeparability to-seed status route)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *to-seed status is recorded by the displayed child route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the existing Cauchy-equivalence, density, completion-facing, transport, replay, provenance, and local naming rows under chapter 1809. The route records only seed-level paper scope for dense-window reads: it does not upgrade the verification axis, does not assert mature $\mathsf{Real}^{\uparrow}$ *completeness, and does not introduce a chosen dense enumeration or countable-choice schedule.*

Proof. Read the hub registration Definition 1810.12. It identifies $\mathsf{RealSeparability}^{\uparrow}$ as a child-route family rather than as theorem body stored on the hub.

Next use Definition 1810.16 and Definition 1810.79. They keep the dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, and terminal Real seal visible before any density consumer receives the separability packet.

The status conclusion is therefore a theory-axis statement about the seed route. The formal verification row remains the one recorded by the surrounding closurestatus block, and the route contains no mature Real-completeness coordinate, selected dense enumeration, or countable-choice data. □

Theorem 1810.81 (Real separability additive seed-status route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *seed-status consumer reads the displayed child route*

$\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbed}$

before it treats the separability packet as a dense Real source. The Cauchy-equivalence classifier is read from chapter 1820; dyadic tolerance, stream-window, regular-rational, and terminal Real rows are read through chapter 291, chapter 126, chapter 301, and chapter 38; and completion-facing density is handed to chapter 1453. The route exports no quotient Real equality, selected dense enumeration, countable-choice schedule, host dense subset, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the seed-status surface of Definition 1810.12. It identifies $\mathsf{RealSeparability}^{\uparrow}$ as a child-route family, so the hub itself contributes no theorem body and no independent density object.

Read the classifier and approximation rows next. The Cauchy-equivalence setoid row fixes the comparison boundary, while the dyadic, StreamName, RegSeqRat, and Real rows display the finite tolerance, observation, readback, and seal route that a dense-name consumer must traverse.

Finally pass through the regular-Cauchy dense embedding boundary. This keeps completion-facing density attached to the same finite seed route, leaving no coordinate for quotient Real equality, selected enumeration, choice, host dense subsets, or ambient completeness. □

Theorem 1810.82 (RealSeparability public closure handoff)**.** *Every public* $\mathsf{RealSeparability}^{\uparrow}$ *countable-dense read is exported through the displayed L10 source route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

inside the separability packet. The handoff may expose dyadic tolerances, finite stream windows, regular rational readback, terminal Real seals, transport, replay, provenance, and local naming, but it does not expose host enumerations of completed reals, selected dense points, quotient real equality, countable choice, completed-real compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the countable dense route of Definition 1810.39 and the public status route of Definition 1810.74. Together they force the density consumer to pass through the displayed finite density rows before any public Real-facing read is available.

The dyadic tolerance row is the first public source row. It controls the approximation window and keeps the density read inside the finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than inside a host enumeration.

The stream and regular-rational rows then expose the inspected finite windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback. Only after those rows are displayed may the consumer inspect the terminal $\mathsf{Real}^{\uparrow}$ seal.

Thus the public handoff is the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Since no row contains a selected dense point, host enumeration, quotient equality, countable-choice selector, compactness theorem, or ambient completeness principle, those objects cannot be exported by the public separability surface. □

Theorem 1810.83 (RealSeparability public density)**.** *Every public countable dense* $\mathsf{RealSeparability}^{\uparrow}$ *read factors through*

$$\mathsf{CountableDenseSubset}^{\uparrow}, \quad \mathsf{DyadicDensityReal}^{\uparrow}, \quad \mathsf{RationalDensityReal}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSe}$$

before any Real-facing seal is exported. The public density surface exposes only the countable-dense boundary, dyadic and rational density routes, dyadic tolerance rows, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming. It exports no selected nearest point, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 1810.82. It already forces any public separability read through the displayed L10 route before the Real-facing public seal can be inspected.

Next read the countable-dense route of Definition 1810.39. That route places the $\mathsf{CountableDenseSubset}^{\uparrow}$ boundary downstream of finite stream windows, regular rational readback, dyadic tolerance data, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally combine the density routes of Definition 1809.6 and Definition 1809.12. They expose the $\mathsf{DyadicDensityReal}^{\uparrow}$ and $\mathsf{RationalDensityReal}^{\uparrow}$ rows before the public density read reaches $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The resulting public density export is therefore the displayed finite route through the countable-dense, dyadic-density, rational-density, dyadic-rational, stream, regular-rational, and Real rows. Since none of these routes contains a selected nearest point, quotient real equality, countable-choice selector, or ambient completeness coordinate, the public density theorem exports none of them. □

Chapter 1811

A Concrete Naming Certificate for $\mathsf{NestedDyadicCauchy}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the interval computations of Moore–Kearfott–Cloud, the $\mathsf{NestedDyadicCauchy}^{\uparrow}$ packet records the finite data by which a nested chain of dyadic intervals becomes a regular rational Cauchy name before the terminal real seal is consumed. It refines the finite-chain surface of chapter 328, the ambient interval discipline of chapter 320, the dyadic endpoint core of chapter 291, the finite schedule interface of chapter 126, the regular rational handoff of chapter 301, and the seal boundary of chapter 38. The packet is a finite-means Cauchy route for L10 real and completion work; it is not a completed intersection point, quotient stream equality, host real equality, countable choice, or ambient completeness.

Definition 1811.1 (Nested dyadic Cauchy carrier)**.** *A* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested dyadic interval window. It records a nonempty finite prefix of dyadic intervals, their endpoint rows through* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the displayed refinement relation inherited from* $\mathsf{NestedDyadicInterval}^{\uparrow}$*.*

2. *B is the shrinking-width dyadic budget. It assigns each displayed tolerance request to a prefix index whose interval width is bounded by the requested dyadic scale.*

3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite interval-prefix entries and endpoint observations that may be queried before regular rational readback.*

4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row formed from the selected endpoint or midpoint observations, the shrinking-width budget, and the stream schedule.*

5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, B, S, R have been displayed.*

6. *H records componentwise* hsame *transport for interval prefixes, dyadic budgets, stream schedules, regular rational readback, real seals, replay rows, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a finite interval-window request through B, S, R, E and then through the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{NestedDyadicInterval}^{\uparrow}$*,* $\mathsf{NestedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{NestedDyadicCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for completed intersections, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge checking.

↪ variant `BEDC.Derived.NestedDyadicCauchyUp.NestedDyadicCauchyTasteGate single carrier alignment`

Theorem 1811.2 (Nested dyadic Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier*

$$K = (I, B, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedDyadicCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested dyadic interval prefix, shrinking-width budget, stream schedule, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*
2. *nested-prefix exposure, requiring each interval read to be one of the displayed prefix rows of I;*
3. *dyadic-width exactness, requiring every Cauchy tolerance request to be discharged by a width comparison stored in B and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the interval-prefix and endpoint observations used by B and R;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be formed only after the prefix, budget, and schedule rows have been projected;*
6. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to be consumed only after R and the structural rows have fixed the finite route.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1811.1. The projection exposes one finite nested dyadic interval prefix, one shrinking-width dyadic budget, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Nested-prefix exposure is forced by I. A consumer may inspect only interval rows already present in the finite prefix, and each such row carries its endpoints through the DyadicRatCore surface named by the packet.

Dyadic-width exactness is then forced by B. Each tolerance request is answered by a displayed prefix index and a displayed dyadic comparison of the corresponding interval width against that tolerance.

The schedule and handoff rows are downstream. The row S records the finite prefix and endpoint observations that may be opened, and R repacks exactly those observations as regular rational Cauchy data.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 1811.3 (Nested dyadic Cauchy regularity-readback obligation)**.** *For every accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier* $K = (I, B, S, R, E, H, C, P, N)$*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer*

may read regularity only after it has displayed the $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The readback route is exactly*

$$S \longmapsto R \longmapsto E$$

together with I, B, H, C, P, N *as structural support; it exports no completed intersection, quotient stream equality, host real equality, countable choice, or ambient completeness.*

Proof. Project the carrier rows. The schedule row S is the only source of endpoint or midpoint observations available to a regular rational consumer, so regularity cannot be read before the schedule is displayed.

Apply Definition 1811.6. The handoff theorem forces every regular rational read through the finite interval prefix I, the shrinking-width budget B, the schedule S, and then the handoff row R.

The Real-facing step is downstream. By Definition 1811.9, the seal row E is opened only after R has fixed the regular rational route.

Tail compatibility is stable under Definition 1811.7: two compatible carriers share the displayed finite route before either seal is consumed. Thus the regularity-readback obligation is exactly the carrier projection through S, R, E and the structural rows. □

Theorem 1811.4 (Nested dyadic Cauchy shrinking-width obligation)**.** *For every accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier* $K = (I, B, S, R, E, H, C, P, N)$*, each Cauchy tolerance request is discharged by a row of the shrinking-width budget* B *whose endpoint comparison is read through* $\mathsf{DyadicRatCore}^{\uparrow}$*. No* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, or completion-facing consumer may use the request until that dyadic width comparison and its prefix index have been displayed.*

Proof. The row B is the only shrinking-width coordinate of the carrier. It assigns each displayed tolerance request to a prefix index and to the dyadic width comparison for the corresponding interval.

The comparison is not external arithmetic. The endpoint rows of I and the width comparison stored in B are read through the $\mathsf{DyadicRatCore}^{\uparrow}$ surface named in the carrier provenance, so the request is answered by displayed dyadic data.

Apply Definition 1811.6. Since R is formed only after I, B, S are projected, the RegSeqRat handoff cannot consume a tolerance request without first opening the dyadic width row.

The same restriction persists through Definition 1811.7. Common-tail gluing compares two requests by a shared dyadic budget row B^* before any shared regular rational row is exported. □

Theorem 1811.5 (Nested dyadic Cauchy schedule-compatibility obligation)**.** *For every accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier* $K = (I, B, S, R, E, H, C, P, N)$*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S *exposes exactly the finite interval-prefix entries and endpoint observations used by* B *and* R*. Any consumer that reads* R *or* E *must therefore show the displayed schedule row first, and the structural rows* H, C, P, N *may transport only that scheduled route.*

Proof. Project S from the carrier. Its entries are the finite interval-prefix and endpoint observations that the packet permits downstream consumers to inspect.

The budget row B and the handoff row R are compatible with that same row. A budget comparison may cite only a scheduled prefix entry, and the regular rational handoff may repack only scheduled endpoint or midpoint observations.

Apply Definition 1811.6. The displayed route $I \mapsto B \mapsto S \mapsto R$ makes the schedule row a prerequisite for every RegSeqRat read.

Finally use Definition 1811.7. When two carriers are glued, the common schedule S^* restricts both reads before the shared handoff R^* is formed. Hence H, C, P, N can only transport, replay, source, and name the scheduled route already displayed. □

Theorem 1811.6 (Nested dyadic Cauchy RegSeqRat handoff)**.** *Let $K = (I, B, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of K factors through the finite route*

$$I \longmapsto B \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite nested dyadic interval prefix, a shrinking-width budget, and the scheduled endpoint observations before it exports the regular rational Cauchy row. Any $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *consumer of the same packet must therefore read the route through I, B, S, R before its completion-facing tail modulus, common-tail window, and terminal Real seal rows are available, as in chapter 1639.*

Proof. Start with the displayed interval prefix I. Its rows give the only endpoint data available to a regular rational consumer, and the refinement relation keeps later prefix rows inside earlier rows.

Use the budget row B to answer a displayed Cauchy tolerance request. The selected prefix index and the dyadic width comparison are part of the same finite packet, so no external modulus or choice of tail is needed.

The schedule row S then fixes which interval endpoints or midpoint observations are opened. The handoff row R is obtained by reading those scheduled observations under the budget discipline and repacking them as the regular rational Cauchy surface.

The rows H, C, P, N transport, replay, source, and name that handoff. Since the carrier has no additional tail selector or quotient coordinate, every RegSeqRat read factors through the displayed route.

Apply the Bishop regular Cauchy completion carrier route of chapter 1639. Its $E, S, R, D, W, H, C, P, N$ rows cannot be opened for this packet until the NestedDyadicCauchy handoff has supplied the finite interval prefix I, dyadic width budget B, StreamName schedule S, and RegSeqRat row R. Thus the completion consumer adds no quotient stream equality, selected limit, countable choice, host real equality, or ambient Real completeness beyond the displayed finite handoff. □

Theorem 1811.7 (Nested dyadic Cauchy tail gluing)**.** *Let $K = (I, B, S, R, E, H, C, P, N)$ and $K' = (I', B', S', R', E', H', C', P', N')$ be accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carriers. Suppose their finite interval prefixes and dyadic budgets admit a displayed common refinement through the StreamName schedules: there are rows I^*, B^*, and S^* such that I^* refines the displayed tails of I and I', B^* compares both dyadic tolerance budgets against that common prefix through chapter 291, and S^* restricts both scheduled reads through chapter 126. Then every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read of K and K' factors through one common finite route*

$$I^* \longmapsto B^* \longmapsto S^* \longmapsto R^*$$

before either $\mathsf{Real}^{\uparrow}$ *seal row E or E' is available. Here R^* is the shared regular rational handoff row supplied by the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface of chapter 301, and the terminal seal boundary remains the* $\mathsf{Real}^{\uparrow}$ *boundary of chapter 38.*

Lean4 ✓ `BEDC.Derived.NestedDyadicCauchyUp.NestedDyadicCauchyTasteGate single carrier alignment`

Proof. Project both accepted carriers by Definition 1811.1. The projections expose only the finite interval-prefix rows, shrinking-width budgets, StreamName schedule rows, RegSeqRat handoff rows, Real seal rows, and structural transport, replay, provenance, and naming rows.

Use the assumed displayed common refinement to choose the common prefix row I^*. The nested-prefix rows identify the tail entries of I and I' that are read by the two consumers, and the common prefix refines both of them before any downstream handoff row is opened.

Next use B^* to compare the two tolerance budgets against the same prefix. The comparisons are dyadic endpoint and width comparisons in the sense of chapter 291, so no external tail selector or ambient completeness principle is needed to align the two requests.

The schedule row S^* restricts both reads to the same finite endpoint observations. By the StreamName discipline of chapter 126, those observations are exactly the endpoint or midpoint rows that the two regular rational consumers may inspect.

Apply Definition 1811.6 to each carrier and compare the two handoffs through the shared rows I^*, B^*, S^*. The RegSeqRat interface of chapter 301 repacks those shared finite observations as one row R^*, so both reads factor through $I^* \mapsto B^* \mapsto S^* \mapsto R^*$ before the Real seal boundary of chapter 38 is consumed.

The construction exports no completed intersection point, quotient stream equality, host real equality, countable-choice selector, or ambient completeness. It only glues two compatible finite tails through the common route available before E or E'. □

Theorem 1811.8 (Nested dyadic Cauchy common-tail uniqueness)**.** *Let $K = (I, B, S, R, E, H, C, P, N)$ and $K' = (I', B', S', R', E', H', C', P', N')$ be accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carriers. Suppose the two packets admit a displayed common refinement I^*, B^*, S^* and that the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R^* is the same displayed route for both packets. Then every* $\mathsf{Real}^{\uparrow}$*-facing read of either packet reaches the same terminal seal boundary through*

$$I^* \longmapsto B^* \longmapsto S^* \longmapsto R^*,$$

without quotient stream equality, selected intersection point, host real equality, countable choice, or ambient completeness.

Proof. Apply tail gluing, Definition 1811.7. The displayed common refinement I^*, B^*, S^* forces both accepted packets through one regular-rational route before either Real seal is available.

Use the RegSeqRat handoff theorem Definition 1811.6 for each carrier. Since the handoff row R^* is the same displayed route for both packets, the two reads share the same finite route $I^* \mapsto B^* \mapsto S^* \mapsto R^*$.

Finally apply the Real seal boundary theorem Definition 1811.9. A Real-facing consumer can inspect a terminal seal only after the regular rational handoff has been displayed, so both packets reach the same terminal boundary through the common route.

The carrier inventory supplies the refusals. None of the cited theorems introduces quotient stream equality, a selected intersection point, host real equality, countable choice, or ambient completeness; they only compare the displayed finite tail route. □

Theorem 1811.9 (Nested dyadic Cauchy Real seal boundary)**.** *For every accepted* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *carrier $K = (I, B, S, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the regular rational handoff row R before the seal row E is available. The consumer receives the finite interval prefix, shrinking-width budget, StreamName schedule, RegSeqRat handoff, Real seal, and structural rows, but receives no completed intersection point, quotient stream equality, host real equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted carrier. The only real-facing coordinate is the seal row E, and E is downstream of the prefix, budget, schedule, and handoff rows by Definition 1811.1.

Apply Definition 1811.6. The regular rational handoff has already forced every read through I, B, S, R, H, C, P, N, so a Real-facing consumer cannot bypass the displayed Cauchy route.

The seal row E packages that handoff as the terminal Real boundary. Because the carrier contains no separate coordinate for an intersection point, a quotient stream, host real equality, a choice-selected tail, or ambient completeness, those objects cannot be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedDyadicCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedDyadicCauchy}^{\uparrow}$ packet is generated from finite nested dyadic interval windows, a shrinking-width dyadic budget, a StreamName schedule, a RegSeqRat handoff, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedDyadicCauchy}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1811.1, Definition 1811.2, Definition 1811.3, Definition 1811.4, Definition 1811.5, Definition 1811.6, Definition 1811.7, and Definition 1811.9 close the finite nested dyadic Cauchy obligation surface from interval prefixes to RegSeqRat readback and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedDyadicCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed intersections, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires Lean-checked carrier admission, nested-prefix exposure, dyadic-width exactness, schedule compatibility, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.NestedDyadicCauchyUp.

Chapter 1812

A Concrete Naming Certificate for $\mathsf{DyadicIntervalNesting}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{DyadicIntervalNesting}^{\uparrow}$ packet names the finite interval-nesting route that turns dyadic endpoint data into a regular rational real-name boundary. It sits between the endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational handoff of chapter 301, the terminal seal of chapter 38, and the finite nested-chain sibling of chapter 328. The packet records a constructive nesting route for L10 Real and Completion work; it does not assert an infinite intersection point, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1812.1 (DyadicIntervalNesting carrier)**.** *A* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (D, S, R, E, I, H, C, P, N)$$

with five displayed NameCert source rows and four structural rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and width ledger for the displayed dyadic intervals, including lower endpoint, upper endpoint, and dyadic comparison reads.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation schedule naming the interval-prefix windows that may be opened by a consumer.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from scheduled endpoint or midpoint reads after the dyadic width ledger has answered the current tolerance request.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R and the nesting row have been displayed.*
5. *I is the existing* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *sibling row, carrying the finite decreasing-chain witness and adjacent refinement ledger.*
6. *H stores componentwise* hsame *transport for the dyadic endpoint ledger, stream schedule, regular rational handoff, real seal, nested-chain row, continuation row, provenance row, and local naming row.*
7. *C stores the* Cont *replay from a finite interval-prefix request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalNesting}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalNesting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, I, H, C, P, N$. The carrier has no coordinate for completed intersections, quotient real names, host real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalNestingUp.DyadicIntervalNestingTasteGate single carrier alignment`

Theorem 1812.2 (DyadicIntervalNesting NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier $J = (D, S, R, E, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalNesting}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *dyadic endpoint admission through D, requiring every interval endpoint and width comparison to be read from* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *finite observation scheduling through S, requiring every public interval-prefix read to be one of the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows;*
3. *regular rational handoff through R, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row to be formed only after the endpoint ledger and finite schedule have been projected;*
4. *real seal nonescape through E, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through D, S, R and the nested-chain row I;*
5. *sibling nesting compatibility through I, requiring the displayed interval chain to be the finite* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *refinement surface before any regular rational or real-name consumer reads it.*

The structural rows H, C, P, N transport, replay, source, and name these five rows and add no further mathematical coordinate.

Proof. Project the carrier of Definition 1812.1. The five mathematical source rows are precisely D, S, R, E, I, and each is named by a concrete sibling chapter in the L10 Real and Completion path.

The dyadic endpoint row supplies the lower endpoint, upper endpoint, and width comparison data. The stream row restricts all public reads to finite interval-prefix windows, so no hidden tail can be inspected.

The regular rational handoff is downstream of those two rows. It repacks scheduled endpoint or midpoint observations only after the dyadic tolerance ledger has selected the displayed comparison used by the current request.

The nested-chain row I supplies the adjacent refinement ledger already exposed by $\mathsf{NestedDyadicInterval}^{\uparrow}$. The real seal E is reached after this finite route, and the structural rows only preserve the same route under hsame transport and Cont replay. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 1812.3 (DyadicIntervalNesting regular name route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier factors through the concrete route*

$$\mathsf{NestedDyadicInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the consumer reads a finite nested dyadic chain, its dyadic endpoint and width ledger, the scheduled finite windows, the regular rational handoff, and finally the real seal. It does not receive an infinite intersection point, quotient stream equality, host real equality, selected limit, countable-choice selector, or ambient completeness theorem.

Proof. Start with the sibling nesting row I. It supplies a finite decreasing dyadic chain and adjacent refinement rows, so every downstream read begins with displayed interval data rather than with a completed point.

Read D next. The endpoint and width comparisons used by the nesting route are dyadic rational rows, and all tolerance comparisons are made inside that displayed ledger.

The schedule row S then fixes the finite interval windows available to the consumer. The handoff row R repacks exactly those scheduled reads as a regular rational name, and the seal row E exposes the corresponding $\mathsf{Real}^{\uparrow}$ boundary only after the handoff has occurred.

The rows H, C, P, N transport, replay, source, and name this route. They do not add quotient equality, host equality, choice, a selected limit, or completeness, so every regular-name and real-name consumer remains inside the displayed chain. □

Theorem 1812.4 (DyadicIntervalNesting width monotonicity)**.** *Let*

$$J = (D, S, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier. When a later interval-prefix window is read from the same* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *row I, the dyadic width ledger in D is no larger than the earlier displayed width, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R remains downstream of that monotone ledger. The route uses only I, D, S, R, E and introduces no completed intersection, host real equality, selected limit, or ambient completeness principle.*

Proof. Start with the nested-chain row I. It is the only carrier row that names the finite decreasing interval chain and its adjacent refinement relation, so a later interval-prefix window must be a later displayed read of the same sibling row.

Read the dyadic ledger D. Its endpoint and width comparison entries record the earlier and later displayed widths, and the later prefix is admitted only through the refinement data supplied by I.

Apply Definition 1812.3. Every regular-name consumer reads the nested row, then the dyadic endpoint and width ledger, then the scheduled finite windows before reaching the RegSeqRat handoff.

Thus the later width is no larger than the earlier displayed width inside the dyadic ledger, and R remains downstream of that monotone ledger. The seal row E is reached only after this route, so the argument never uses a completed intersection or host real equality. □

Theorem 1812.5 (DyadicIntervalNesting obligation scope)**.** *For an accepted* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier $J = (D, S, R, E, I, H, C, P, N)$, the obligation scope is exactly the* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *row I, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and width ledgers in D, finite* $\mathsf{StreamName}^{\uparrow}$ *windows in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, the* $\mathsf{Real}^{\uparrow}$ *seal E, the ambient* BHist *packet, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N. No completed intersection, quotient stream equality, host real equality, selected limit, countable choice, or ambient completeness coordinate lies in the scope.*

Proof. Project the carrier rows $I, D, S, R, E, H, C, P, N$ from Definition 1812.1. The nested interval row, endpoint-width ledger, scheduled windows, regular handoff, and seal are exactly the mathematical source rows, while H, C, P, N preserve and name the same packet inside BHist.

Apply Definition 1812.2. It identifies the local NameCert clauses with dyadic endpoint admission, finite observation scheduling, regular rational handoff, real seal nonescape, and sibling nesting compatibility.

Apply Definition 1812.3 and Definition 1812.4. These keep every regular or Real consumer downstream of I, D, S, R, E and keep width reads inside the dyadic ledger. Since the carrier has no further coordinate, the listed rows exhaust the obligation scope. □

Theorem 1812.6 (DyadicIntervalNesting Real seal handoff)**.** *Every accepted* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier*

$$J = (D, S, R, E, I, H, C, P, N)$$

reaches a $\mathsf{Real}^{\uparrow}$ *boundary only through the dyadic endpoint ledger* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observations* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name route* R*, the width-monotonicity row carried by* D *and* I*, and the sibling* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *chain packet* I*. Thus the Real seal* E *is a finite nesting-to-real route and not an infinite-intersection selector, quotient stream equality, host real equality, selected limit, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Apply Definition 1812.3. It sends every regular-name or Real consumer through the sibling nested chain, the dyadic endpoint-width ledger, finite stream windows, and the RegSeqRat handoff before the seal is read.

Next use Definition 1812.4. The later interval windows remain inside the same dyadic ledger and the same sibling chain, so the route to R carries monotone displayed width data rather than a completed intersection point.

Finally apply Definition 1812.5. The scope theorem lists exactly $I, D, S, R, E, H, C, P, N$ and excludes any extra host coordinate. Therefore the Real seal is reached only through the finite nesting route, with no quotient equality, host equality, selected limit, choice principle, or ambient completeness theorem. □

Theorem 1812.7 (DyadicIntervalNesting terminal seal boundary)**.** *For every accepted* $\mathsf{DyadicIntervalNesting}^{\uparrow}$ *carrier*

$$J = (D, S, R, E, I, H, C, P, N),$$

each $\mathsf{Real}^{\uparrow}$*-facing consumer must read exactly the finite nesting route* $I, D, S, R, E, H, C, P, N$*. The route begins with the nested-chain row* I*, then the dyadic endpoint and width ledger* D*, the finite observation schedule* S*, the regular rational handoff* R*, and the terminal Real seal* E*, with* H, C, P, N *only transporting, replaying, sourcing, and naming the same packet. It cannot be replaced by a completed intersection, quotient stream equality, selected limit, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 1812.1. The rows I, D, S, R, E are the only mathematical source rows, and H, C, P, N are the only structural rows available to a consumer.

Apply Definition 1812.2. It forces endpoint admission through D, finite observation through S, regular rational handoff through R, Real-seal nonescape through E, and sibling nesting compatibility through I.

Use Definition 1812.3 and Definition 1812.4. These keep the regular-name route downstream of the finite nested-chain row and keep every width comparison inside the dyadic ledger rather than inside a completed intersection.

Finally apply Definition 1812.5 and Definition 1812.6. The scope theorem exhausts the readable rows, and the handoff theorem makes E terminal. Since the excluded intersection, quotient, selected-limit, choice, host-equality, and completeness data have no carrier coordinate, no Real-facing consumer can substitute them for the finite terminal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalNesting}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalNesting}^{\uparrow}$ packet is generated from a finite NestedDyadicInterval row, DyadicRatCore endpoint and width ledgers, StreamName windows, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNesting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1812.1, Definition 1812.2, Definition 1812.3, and Definition 1812.4 seed the finite dyadic interval nesting route from endpoint ledgers and scheduled windows to regular rational readback and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNesting}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DyadicIntervalNestingUp.DyadicIntervalNestingTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close infinite interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Lean verification for every obligation row, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint ledger exactness, finite-window scheduling, regular rational handoff, sibling nesting compatibility, and Real seal nonescape for BEDC.Derived.DyadicIntervalNestingUp.

Chapter 1813

A Concrete Naming Certificate for $\mathsf{DyadicExponentShift}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicExponentShift}^{\uparrow}$ packet records the finite exponent-shift normalization route used before dyadic data is handed to regular rational and real-name consumers. It sits between the dyadic core boundary of chapter 291, the precision-scale packet of chapter 1768, the canonical normal-form surface of chapter 1613, the finite-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a BEDC-native finite normalization ledger for shifting dyadic exponents; it is not quotient equality of dyadics, host arithmetic on powers of two, or an appeal to ambient real completeness.

Definition 1813.1 (Dyadic exponent-shift carrier)**.** *A* $\mathsf{DyadicExponentShift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, U, K, S, D, T, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the signed mantissa row, read through the finite integer and rational surfaces already admitted by the dyadic core.*
2. *U is the unary exponent row attached to M, recording the current dyadic denominator scale before any shift is applied.*
3. *K is the shift amount row, recording a finite signed shift request as BEDC data rather than as host exponent arithmetic.*
4. *S is the common-scale row, displaying the single finite scale at which the shifted mantissa and shifted exponent are compared.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row for the mantissa-exponent pair before and after shifting.*
6. *T is the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *handoff row, recording the precision-scale ledger that downstream regular rational and real-name consumers may inspect.*
7. *F is the* $\mathsf{DyadicNormalForm}^{\uparrow}$ *handoff row, recording the canonical finite normal-form surface reached after the common-scale comparison.*
8. *H records componentwise* hsame *transport for the mantissa, exponent, shift amount, common scale, dyadic readback, tolerance-scale handoff, normal-form handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the input dyadic packet through shift inspection, common-scale comparison, dyadic readback, tolerance-scale handoff, normal-form handoff, transport, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{DyadicExponentShift}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{DyadicNormalForm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicExponentShift}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, K, S, D, T, F, H, C, P, N$. It has no coordinate for quotient dyadic equality, host powers of two, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

↪variant BEDC.Derived.DyadicExponentShiftUp.DyadicExponentShiftTasteGate single carrier alignment

Theorem 1813.2 (Dyadic exponent-shift NameCert obligations)**.** *For every accepted* $\mathsf{DyadicExponentShift}^{\uparrow}$ *carrier $E = (M, U, K, S, D, T, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicExponentShift}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the signed mantissa, unary exponent, shift amount, common scale, dyadic readback, tolerance-scale handoff, normal-form handoff, transport, replay, provenance, and local naming rows;*

2. *shift-ledger exactness, requiring K to be consumed as the only shift request and requiring S to record the finite common scale before any shifted readback is exported;*

3. *common-scale classifier stability, requiring accepted comparisons to preserve $M, U, K, S, D, T, F, H, C, P, N$ componentwise under* hsame *transport and displayed* Cont *replay;*

4. *normalization handoff, requiring* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *and* $\mathsf{DyadicNormalForm}^{\uparrow}$ *rows to be exposed before* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumers read the shifted dyadic;*

5. *Real-and-Completion nonescape, requiring P and N to export no quotient dyadic equality, host powers of two, selected limits, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or public bridge component.*

No exponent-shift NameCert obligation is stored outside these rows.

Lean4 ✓ BEDC.Derived.DyadicExponentShiftUp.DyadicExponentShiftTasteGate single carrier alignment

Proof. Carrier admission is projection from Definition 1813.1. The finite packet exposes exactly the mantissa row, unary exponent row, shift row, common-scale row, dyadic readback row, two normalization handoff rows, and structural rows H, C, P, N.

Shift-ledger exactness is the role of K and S. Since K is the only row carrying a shift amount and S is the only row carrying the common comparison scale, a shifted readback cannot be named before both rows have been displayed.

Classifier stability is componentwise through H and replayed through C. These rows preserve the route from M, U, K into S, then into the DyadicRatCore readback D, and then into the tolerance-scale and normal-form rows T and F.

The normalization handoff follows from the placement of T and F. A stream, regular rational, or real-name consumer reaches the shifted dyadic only after those two rows have been exposed, so the consumer reads a displayed finite normalization route rather than an implicit exponent manipulation.

The nonescape row is immediate from the carrier inventory. Quotient dyadic equality, host powers of two, selected limits, trichotomy, countable choice, ambient real completeness, and bridge

components are absent from the tuple and are not produced by transport, replay, provenance, or local naming. □

Theorem 1813.3 (Dyadic exponent-shift normalization handoff)**.** *Let* $E = (M, U, K, S, D, T, F, H, C, P, N)$ *be an accepted* $\mathsf{DyadicExponentShift}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an exponent-shifted dyadic read factors through the finite route*

$$M, U, K \longmapsto S \longmapsto D \longmapsto T, F \longmapsto H, C, P, N$$

over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{DyadicNormalForm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. Consequently the shifted dyadic is exported only as a normalized finite packet, not through quotient dyadic equality, host exponent arithmetic, selected limits, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a regular rational or real-name consumer of the shifted dyadic. By Definition 1813.1, the consumer can reach dyadic readback only at the row D.

The readback row D is downstream of the visible route M, U, K, S. Thus the mantissa, unary exponent, requested shift, and common-scale comparison must all be displayed before the shifted dyadic is available.

The rows T and F then route the same readback through the tolerance-scale and normal-form surfaces. These rows are the only downstream normalization interface presented to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers.

Finally, H, C, P, N transport, replay, package, and name that same route. Since no excluded coordinate appears in the packet, the consumer receives only the displayed finite normalization route over the listed BEDC rows. □

Theorem 1813.4 (Dyadic exponent-shift canonical window)**.** *Let* $E = (M, U, K, S, D, T, F, H, C, P, N)$ *be an accepted* $\mathsf{DyadicExponentShift}^{\uparrow}$ *packet. The packet exposes the same canonical finite observation window through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{DyadicNormalForm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*: exponent shifting changes only the displayed normalization ledger* K, S, D, T, F*, not the downstream* $\mathsf{Real}^{\uparrow}$ *seal route through* H, C, P, N*.*

Proof. Project the carrier rows from Definition 1813.1. The rows M, U, K, S, D, T, F display the mantissa, unary exponent, shift request, common-scale comparison, dyadic readback, tolerance-scale handoff, and normal-form handoff before any downstream consumer reads the packet.

Apply the normalization handoff of Definition 1813.3. It forces every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ read through the same finite route from M, U, K into S, D, T, F and then into the structural rows.

Finally read the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows componentwise through H and replay them through C. The exponent shift therefore changes the displayed normalization ledger only; the downstream real-name seal remains the same transported and replayed finite window. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicExponentShift}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicExponentShift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1813.1, Definition 1813.2, and Definition 1813.3 seed the finite exponent-shift route from mantissa and unary exponent data through common-scale comparison, DyadicRatCore readback, tolerance-scale handoff, and normal-form handoff into regular rational and real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicExponentShift}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient dyadic equality, host powers of two, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shift-ledger exactness, common-scale classifier stability, normalization handoff, and Real-and-Completion nonescape for BEDC.Derived.DyadicExponentShiftUp.

Constructive story: A DyadicExponentShift$^{\uparrow}$ packet is generated from a signed mantissa row, unary exponent row, shift amount row, common-scale row, DyadicRatCore readback, DyadicToleranceScale and DyadicNormalForm handoff rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1814

A Concrete Naming Certificate for $\mathsf{CauchyRateDominance}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRateDominance}^{\uparrow}$ packet records the finite certificate by which one displayed Cauchy-modulus schedule is dominated by another before a Real-facing or completion-facing read consumes the resulting seal. It sits over the explicit modulus surface of chapter 292, the Real-facing modulus boundary of chapter 1423, the finite-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the domination-test consumer of chapter 1459. The packet gives later Real and Completion consumers a reusable finite domination row instead of requiring each consumer to repeat the modulus comparison inside its own carrier.

Definition 1814.1 (Cauchy rate dominance carrier)**.** *A* $\mathsf{CauchyRateDominance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule. It records the precision requests, source thresholds, and monotone reuse rows whose domination is being certified.*
2. *D is the displayed dominating modulus row. It records the target thresholds, comparison witnesses, and finite ledger entries showing that each accepted source request is bounded by the dominating schedule on the same displayed precision surface.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the dominated schedule and inspected only at requests listed in M and D.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from those windows, thresholds, and dyadic tolerance entries.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after M, D, S, Q have all been displayed.*
6. *H records componentwise* hsame *transport for the source modulus schedule, dominating row, finite windows, regular readback, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through M, D, S, Q, E and then through the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyRateDominance}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateDominance}^{\uparrow}}$ *row.*

The classifier accepts only packets whose reads pass through these nine rows. It has no coordinate for quotient stream equality, a selected limit, host real comparison, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRateDominanceUp.CauchyRateDominanceTasteGate single carrier alignment`

Theorem 1814.2 (Cauchy rate dominance NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateDominance}^{\uparrow}$ *carrier*

$$R = (M, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRateDominance}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, dominating modulus row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*

2. *dominance ledger exactness, requiring each accepted precision request to read the source threshold in M and the displayed dominating threshold in D before a window or readback can be consumed;*

3. *componentwise transport, requiring H to preserve the source schedule, dominating row, selected windows, regular readback, terminal seal, replay, provenance, and naming rows together;*

4. *selected-window stability, requiring every window opened by S to be one of the windows licensed by the displayed domination ledger D for the source schedule M;*

5. *non-escape, requiring E, P, N to export no quotient stream equality, selected limit, host real comparison, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No dominance obligation is stored outside these rows.

Proof. Project the carrier of Definition 1814.1. The projection exposes exactly one source modulus schedule, one displayed dominating row, one finite window family, one regular rational readback, one terminal seal, and the structural transport, replay, provenance, and naming rows.

Dominance ledger exactness is forced before any analytic read occurs. A precision request first cites its source threshold in M, then reads the dominating threshold and comparison evidence in D. Only after both rows are displayed may the request open a StreamName window.

Selected-window stability follows from the order of the route. The window row S is downstream of M and D, so any window consumed by Q is a window licensed by the domination ledger for the same precision request.

Finally H, C, P, N only transport, replay, source, and name the displayed route, while E is the unique terminal seal row. Since the carrier has no additional analytic coordinate, the local NameCert surface cannot export quotient stream equality, selected limits, host comparison, choice, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Theorem 1814.3 (Cauchy rate dominance Real seal handoff)**.** *Every Real-facing read through the dominated modulus in an accepted* $\mathsf{CauchyRateDominance}^{\uparrow}$ *carrier factors through the same finite route*

$\mathsf{CauchyModulus}^{\uparrow}$, D, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Equivalently, the read first consumes the source schedule M, the dominating row D, the selected window row S, and the regular readback Q, and only then reaches the terminal seal E and the structural rows H, C, P, N.

Proof. Begin with a Real-facing read admitted by the carrier. By Definition 1814.2, the read has no direct route to E; it must first read the source schedule M and the displayed domination ledger D.

The finite window and readback rows are then fixed. The row S supplies exactly the StreamName windows licensed by D, while the dyadic tolerance entries named in D determine the regular rational readback Q.

The terminal seal is downstream of that readback. Thus E is reached only through the route M, D, S, Q, and the remaining rows H, C, P, N transport, replay, source, and name the same displayed packet. This gives the stated factorization through the CauchyModulus, dominating row, StreamName, DyadicRatCore, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert surfaces. □

Falsifiable prediction: An accepted $\mathsf{CauchyRateDominance}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first reading the source $\mathsf{CauchyModulus}^{\uparrow}$ schedule, the displayed dominating row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: cauchymodulus, realcauchymodulus, streamname, dyadicratcore, regseqrat, regular-cauchy-comparison-test, real: no listed sibling supplies the whole nine-row packet $R = (M, D, S, Q, E, H, C, P, N)$, because each sibling contributes only schedule, window, tolerance, readback, comparison-test, or seal data while this chapter binds explicit domination, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateDominance}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRateDominance}^{\uparrow}$ packet is generated from a source CauchyModulus schedule, a displayed dominating modulus row, finite StreamName windows, DyadicRatCore tolerance entries, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateDominance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1814.1, Definition 1814.2, and Definition 1814.3 seed the finite domination surface for explicit Cauchy-modulus schedules through the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateDominance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real comparison, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dominance ledger exactness, componentwise transport, selected-window stability, non-escape, and Real seal handoff for BEDC.Derived.CauchyRateDominanceUp.

Chapter 1815

A Concrete Naming Certificate for $\mathsf{UniformHomeomorphism}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{UniformHomeomorphism}^{\uparrow}$ packet presents a uniform homeomorphism as a bidirectional uniform-continuity certificate between two $\mathsf{UniformSpace}^{\uparrow}$ rows. It is grounded by the uniform-space surface of chapter 339, the modulus discipline of chapter 391, the graph and pointwise route of chapter 130, the embedding boundary of chapter 1651, and the completion consumer surface of chapter 1473. The chapter records the L10 equivalence route used by completion comparisons without exposing quotient completion equality, selected representatives, host equality, countable choice, or ambient completeness.

Definition 1815.1 (UniformHomeomorphism carrier)**.** *A* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, f, g, F, G, M, N, H, C, P, L)$$

with the following displayed rows:

1. *X and Y are the source and target* $\mathsf{UniformSpace}^{\uparrow}$ *rows, including only the entourage requests and refinement ledgers visible to the certificate.*
2. *f and g are the forward and inverse graph rows, read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface and carrying no host function object beyond the displayed graph data.*
3. *F and G are the forward and inverse uniform-continuity rows, each exposing the source tolerance request, the target tolerance request, and the corresponding finite modulus route.*
4. *M and N are the forward and inverse modulus rows, respectively, so that each direction has its own displayed tolerance transformer before any completion consumer may read the route.*
5. *H records inverse-route compatibility: the composites $g \circ f$ and $f \circ g$ are accepted only through displayed graph rows, displayed uniform-continuity rows, and componentwise* hsame *transport.*
6. *C records* Cont *replay from a source or target entourage request through the appropriate graph row, the matching modulus row, inverse-route compatibility, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, f, g, F, G, M, N, H, C, P, L$. The carrier contains no quotient completion equality, selected representative, host equality of points or maps, countable-choice selector, ambient completeness datum, or hidden completion object.

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismCarrier field projection round trip`

Theorem 1815.2 (UniformHomeomorphism NameCert obligations)**.** *For every accepted* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *carrier $U = (X, Y, f, g, F, G, M, N, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source and target* $\mathsf{UniformSpace}^{\uparrow}$ *rows, forward and inverse graph rows, forward and inverse uniform-continuity rows, both modulus rows, inverse-route compatibility, replay, provenance, and local naming;*
2. *classifier bidirectionality, requiring rowwise comparison to preserve both directions at once rather than accepting a one-sided uniform-continuity certificate;*
3. *stability of both moduli, requiring M and N to transport with their matching graph and uniform-continuity rows under componentwise* hsame*;*
4. *inverse-route exactness, requiring the two composites to be read only through H after f, g, F, G, M, N have fixed the displayed bidirectional route;*
5. *ledger non-escape, requiring C, P, L to replay, source, and name no row from which quotient completion equality, selected representatives, host equality, countable choice, ambient completeness, or a hidden completion object can be exported.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1815.1. The packet displays the two uniform-space rows, the two graph rows, the two uniform-continuity rows, both modulus rows, inverse-route compatibility, replay, provenance, and local naming.

Classifier bidirectionality follows because the classifier reads the whole tuple $X, Y, f, g, F, G, M, N, H, C, P, L$. A packet cannot be accepted by comparing only f, F, M or only g, G, N; both directions are part of the same displayed row comparison.

Stability of both moduli is the componentwise transport clause. The forward modulus row M travels with f and F, while the inverse modulus row N travels with g and G, and both transports are recorded in H and replayed through C.

Inverse-route exactness is forced after the two directed routes have been fixed. The row H is the only displayed location where the composites $g \circ f$ and $f \circ g$ are named, so a consumer cannot introduce a separate host inverse or quotient-completion equality witness.

Finally, ledger non-escape is exhausted by C, P, L. These rows replay, source, and name only the displayed route, and none of them contains selected representatives, host equality, countable choice, ambient completeness, or a hidden completion object. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismNameCertObligations`

Theorem 1815.3 (UniformHomeomorphism bidirectional modulus route)**.** *Every accepted* $\mathsf{UniformHomeomorphism}$ *consumer route factors through both displayed moduli M and N along the finite route*

$$X, Y \longmapsto f, g \longmapsto F, G \longmapsto M, N \longmapsto H \longmapsto C, P, L.$$

Consequently the packet cannot read quotient completion equality, selected representatives, host equality, countable choice, or ambient completeness.

Proof. Start with a consumer route for an accepted packet and project the carrier rows. The source and target uniformities are visible only as X and Y, and the directed graphs are visible only as f and g.

The route must then read the two uniform-continuity rows F and G. The forward read exposes the modulus row M, and the inverse read exposes the modulus row N; omitting either one would leave only a one-sided continuity route rather than a uniform homeomorphism packet.

After both moduli have been read, the inverse-route row H records the displayed composites. This is the only route by which the consumer sees bidirectionality, so quotient completion equality, selected representatives, and host equality are not available as alternate witnesses.

The structural rows C, P, L close the consumer route by replay, provenance, and local naming. Since these rows name only $X, Y, f, g, F, G, M, N, H$, the packet exports no countable-choice selector, ambient completeness datum, or hidden completion object. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismBidirectionalModulusRoute`

Theorem 1815.4 (UniformHomeomorphism classifier bidirectionality)**.** *For every accepted* UniformHomeomorphism *carrier* $U = (X, Y, f, g, F, G, M, N, H, C, P, L)$*, rowwise classifier agreement preserves the forward graph row* f*, inverse graph row* g*, forward uniform-continuity row* F*, inverse uniform-continuity row* G*, forward modulus row* M*, and inverse modulus row* N *simultaneously. In particular, an agreement that preserves only* f, F, M *or only* g, G, N *is not a* UniformHomeomorphism$^{\uparrow}$ *classifier agreement.*

Proof. Project the classifier surface from Definition 1815.1. The accepted packet is compared through the full tuple $X, Y, f, g, F, G, M, N, H, C, P, L$, and Definition 1815.2 lists classifier bidirectionality as a local obligation.

The displayed route of Definition 1815.3 then forces both directed graph/modulus chains to remain visible before the inverse-route row H and the structural rows C, P, L can be read. A one-sided comparison omits one of the directed chains and therefore fails to compare the accepted carrier rows. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismClassifierBidirectionality`

Theorem 1815.5 (UniformHomeomorphism inverse-route exactness)**.** *For every accepted* UniformHomeomorphism *carrier* $U = (X, Y, f, g, F, G, M, N, H, C, P, L)$*, the composites* $g \circ f$ *and* $f \circ g$ *are readable only through the displayed graph rows* f, g*, both uniform-continuity rows* F, G*, both modulus rows* M, N*, and the inverse-route row* H*. No consumer may replace this route by a host inverse, quotient completion equality, selected representative, or ambient equality of maps.*

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismCarrier inverse route exactness`

Proof. Use the carrier definition to isolate the only rows that mention the directed maps and their inverse route. The graph rows are f, g, the uniform-continuity rows are F, G, the modulus rows are M, N, and the composite-read row is H.

The NameCert obligations of Definition 1815.2 require inverse-route exactness after both directed routes have been fixed. The bidirectional modulus route of Definition 1815.3 gives the order

$$X, Y \longmapsto f, g \longmapsto F, G \longmapsto M, N \longmapsto H.$$

Since the carrier has no additional coordinate for a host inverse, quotient completion equality, selected representative, or ambient equality of maps, the two composites can be read only through that displayed route. □

Theorem 1815.6 (UniformHomeomorphism completion consumer boundary)**.** *Any* UniformCompletion$^{\uparrow}$ *consumer reading an accepted* UniformHomeomorphism$^{\uparrow}$ *packet may use it only as the bidirectional modulus route through* $X, Y, f, g, F, G, M, N, H, C, P, L$*. The consumer may not read quotient completion equality, selected completion representatives, host equality, countable choice, or ambient completeness from the packet.*

Proof. Start with the completion-facing provenance row P in Definition 1815.1. It names UniformCompletion$^{\uparrow}$ only as a downstream consumer of the displayed uniform-space, graph, uniform-continuity, modulus, inverse-route, replay, provenance, and local naming rows.

Apply the NameCert obligation surface and the bidirectional modulus route. They expose exactly the two directed moduli and the inverse-route row before any completion-facing read is available.

Finally use the carrier exclusions. Because no row stores quotient completion equality, selected representatives, host equality, countable choice, ambient completeness, or a hidden completion object, a UniformCompletion$^{\uparrow}$ consumer can import only the finite bidirectional modulus route. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismCompletionConsumerBoundary`

Theorem 1815.7 (UniformHomeomorphism bidirectional modulus nonescape)**.** *Every accepted* UniformHomeomorphism$^{\uparrow}$ *carrier read that exposes either directed modulus row must expose the same displayed route through* UniformSpace$^{\uparrow}$*,* UniformContinuity$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* DenseUniformEmbedding$^{\uparrow}$*,* UniformCompletion$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The read may use the forward modulus* M *only together with* X, Y, f, F *and the inverse modulus* N *only together with* Y, X, g, G*, and both routes must pass through* H, C, P, L *before any completion-facing consumer can inspect them. No selected inverse, quotient completion equality, host function equality, countable-choice selector, or ambient completeness datum is exported by either direction.*

Proof. Project the carrier of Definition 1815.1. The forward route displays X, Y, f, F, M, and the inverse route displays Y, X, g, G, N; neither modulus is a free-standing row.

Apply the bidirectional modulus route of Definition 1815.3. It forces both directed chains to pass through the inverse-route compatibility row H and then through the structural rows C, P, L.

The completion consumer boundary of Definition 1815.6 is reached only after those rows have been displayed. Since the carrier has no coordinate for a selected inverse, quotient completion equality, host function equality, countable-choice selector, or ambient completeness datum, no such datum can escape from either modulus route. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismBidirectionalModulusNonescape`

Theorem 1815.8 (UniformHomeomorphism completion classifier stability)**.** *Completion-facing classifier comparisons for accepted* UniformHomeomorphism$^{\uparrow}$ *carriers are stable only along the displayed forward and inverse modulus rows. If two carriers are compared by a* UniformCompletion$^{\uparrow}$ *consumer, then the comparison must preserve* X, Y, f, F, M *and* Y, X, g, G, N *before it reads* H, C, P, L*; a comparison that identifies completion endpoints while dropping one directed modulus is not a* UniformHomeomorphism$^{\uparrow}$ *classifier comparison.*

Proof. Start with a completion-facing comparison. By Definition 1815.6, the consumer sees the packet only as the finite route $X, Y, f, g, F, G, M, N, H, C, P, L$.

Use classifier bidirectionality from Definition 1815.4. It requires the forward graph, inverse graph, both uniform-continuity rows, and both modulus rows to be preserved simultaneously.

It remains to exclude completion-only identifications. The carrier contains no quotient completion equality or host function equality row, and Definition 1815.7 keeps both directed modulus routes visible before H, C, P, L are consumed. Therefore completion-facing stability is exactly stability along the displayed forward and inverse modulus rows. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismCompletionClassifierStability`

Theorem 1815.9 (UniformHomeomorphism scoped L10 route)**.** *Every L10 completion-facing read of an accepted* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *packet factors through* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before any completion consumer inspects it. The route may expose the displayed bidirectional modulus and inverse-route compatibility rows, but it never exposes quotient completion equality, selected representatives, countable choice, host map equality, or ambient completeness.*

Proof. Start with the carrier of Definition 1815.1. Its source and target uniformities, directed graph rows, directed uniform-continuity rows, modulus rows, inverse-route compatibility, replay, provenance, and local naming are the only rows available to a completion-facing read.

Apply the bidirectional modulus route of Definition 1815.3. It forces a consumer to read both directed moduli before the inverse-route row can be inspected.

Use inverse-route exactness from Definition 1815.5. The composites are readable only through the displayed graph, continuity, modulus, and compatibility rows, so no host inverse or quotient-completion equality can be substituted.

Finally apply the completion consumer boundary and completion classifier stability of Definition 1815.6 and Definition 1815.8. They place every L10 completion-facing comparison downstream of the same finite route through $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, while selected representatives, countable choice, host map equality, and ambient completeness remain outside the packet. □

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismScopedRoute`

Theorem 1815.10 (UniformHomeomorphism public NameCert export)**.** *The carrier, NameCert obligations, bidirectional modulus route, classifier bidirectionality, inverse-route exactness, completion consumer boundary, bidirectional modulus nonescape, completion classifier stability, and scoped L10 route assemble the public* $\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}$ *surface over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. The exported surface preserves the exclusions of quotient completion equality, selected representatives, countable choice, host map equality, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.UniformHomeomorphismUp.UniformHomeomorphismPublicExport`

Proof. Start from the carrier of Definition 1815.1 and its obligation list in Definition 1815.2. These two sites supply the displayed source and target uniform spaces, forward and inverse graph rows, forward and inverse uniform-continuity rows, both modulus rows, inverse-route compatibility, replay, provenance, and local naming.

The directed modulus chain is fixed by Definition 1815.3. Classifier bidirectionality from Definition 1815.4 then keeps the forward and inverse graph, continuity, and modulus rows visible together, while inverse-route exactness from Definition 1815.5 prevents a host inverse or ambient map equality from replacing the displayed compatibility row.

The completion-facing boundary is supplied by Definition 1815.6, refined by the nonescape clause of Definition 1815.7, and stabilized by Definition 1815.8. The scoped L10 route of Definition 1815.9 repacks those rows as the public completion-facing surface.

Because each cited site keeps quotient completion equality, selected representatives, countable choice, host map equality, and ambient completeness outside the displayed carrier rows, their composite public export keeps the same exclusions. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformHomeomorphism}^{\uparrow}$ packet is generated from two UniformSpace rows, forward and inverse graph rows, forward and inverse UniformContinuity rows, two displayed modulus rows, inverse-route compatibility, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1815.1, Definition 1815.2, Definition 1815.3, Definition 1815.4, Definition 1815.5, Definition 1815.6, Definition 1815.7, Definition 1815.8, and Definition 1815.9 close the carrier, NameCert obligations, bidirectional modulus route, completion consumer boundary, completion classifier stability, and scoped L10 route for finite completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformHomeomorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public-object surface and Lean verification for the bidirectional modulus, classifier stability, inverse-route exactness, and ledger non-escape rows of BEDC.Derived.UniformHomeomorphismUp.

Chapter 1816

A Concrete Naming Certificate for $\mathsf{RealExponential}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealExponential}^{\uparrow}$ packet records the exponential function by the displayed power series with coefficients $1/n!$ and by uniform Cauchy control on bounded real windows. It is a finite NameCert route through the $\mathsf{RealPowerSeries}^{\uparrow}$ carrier of chapter 1583, the shared tail discipline of $\mathsf{UniformCauchySeries}^{\uparrow}$ in chapter 1736, the series-product handoff of $\mathsf{CauchyProduct}^{\uparrow}$ in chapter 464, the arithmetic seal of $\mathsf{CauchyRealField}^{\uparrow}$ in chapter 1717, the classifier boundary of $\mathsf{RealEquality}^{\uparrow}$ in chapter 1636, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names exponential as a uniformly Cauchy real power-series packet; it does not import host exp, host transcendental-function laws, analytic continuation, quotient real equality, selected infinite sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1816.1 (RealExponential carrier)**.** *A* $\mathsf{RealExponential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, F, B, U, M, K, P, S, R, Q, T, C, L, N)$$

with the following displayed rows:

1. *A is the coefficient ledger fixing the exponential power series by the finite readback of the coefficient rule $a_n = 1/n!$.*
2. *F is the factorial and reciprocal dyadic budget row, recording the inspected natural-number factorial windows, reciprocal dyadic enclosures, and denominator-growth bounds used by the coefficient ledger.*
3. *B is the bounded real-window row on which the packet accepts a request for evaluation.*
4. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *modulus row showing that the displayed coefficient windows and powers form a uniformly Cauchy series on B.*
5. *M is the majorant and tail-budget row used to discharge the uniform convergence request without selecting a host infinite sum.*
6. *K is the* $\mathsf{CauchyProduct}^{\uparrow}$ *law-route row that records the Cauchy-product passage used for the addition law handoff.*
7. *P is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *evaluation row, exposing the coefficient ledger, argument window, partial sums, and endpoint seal only as power-series data.*

8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal reached after the coefficient ledger, dyadic budget, uniform Cauchy modulus, product-law route, and power-series evaluation rows have all been displayed.*
9. *R is the* $\mathsf{RealEquality}^{\uparrow}$ *classifier row used to compare accepted exponential readbacks through the displayed real-equality boundary.*
10. *Q is the* $\mathsf{CauchyRealField}^{\uparrow}$ *arithmetic row used for finite addition, multiplication, and reciprocal-budget accounting inside the route.*
11. *T stores componentwise* hsame *transport for the coefficient ledger, factorial budget, bounded window, uniform modulus, majorant, product-law route, power-series row, real seal, equality classifier, arithmetic row, continuation, provenance, and local naming rows.*
12. *C stores displayed* Cont *replay from coefficient and factorial reads to partial sums, from partial sums to uniform Cauchy control, from the uniform route to the* $\mathsf{RealPowerSeries}^{\uparrow}$ *endpoint, from Cauchy product rows to the addition-law handoff, and from the endpoint to the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealEquality}^{\uparrow}$ *consumers.*
13. *L is the* Pkg *provenance over* $\mathsf{RealExponential}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{CauchyRealField}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{RealExponential}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, F, B, U, M, K, P, S, R, Q, T, C, L, N$. It has no coordinate for a host transcendental function, analytic continuation, quotient real equality, selected infinite sum, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1816.2 (RealExponential NameCert obligations)**.** *For every accepted* $\mathsf{RealExponential}^{\uparrow}$ *carrier $E = (A, F, B, U, M, K, P, S, R, Q, T, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealExponential}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier exposure for the displayed coefficient ledger, factorial and reciprocal budget, bounded real window, uniform Cauchy modulus, majorant, Cauchy-product law route, power-series evaluation row, real seal, real-equality classifier, arithmetic row, transport, continuation, provenance, and local naming rows;*
2. *classifier exposure through R, requiring every accepted comparison to preserve the displayed* $\mathsf{RealEquality}^{\uparrow}$ *route and no hidden host equality;*
3. *uniform Cauchy convergence through U and M, requiring every endpoint-facing read to use the displayed tail modulus and majorant budget before the endpoint seal is read;*
4. *product and addition law handoff through K and Q, requiring the exponential addition law to be read only through the* $\mathsf{CauchyProduct}^{\uparrow}$ *and* $\mathsf{CauchyRealField}^{\uparrow}$ *rows;*
5. *non-escape through T, C, L, N, requiring transport, replay, provenance, and local naming to preserve the same finite route and to export no host* $\exp$*, analytic-continuation datum, quotient real equality, selected infinite sum, countable-choice selector, or ambient completeness witness.*

Proof. Carrier exposure is projection from Definition 1816.1. The accepted packet displays exactly the rows $A, F, B, U, M, K, P, S, R, Q, T, C, L, N$, so there is no detached coordinate from which a consumer could read a host exponential function.

The classifier exposure is the row R. Since R is declared as the $\mathsf{RealEquality}^{\uparrow}$ boundary and the transport row T is componentwise, accepted comparisons can preserve only the displayed real-equality route attached to the packet.

Uniform convergence is read through U and M. The coefficient ledger and factorial budget first produce finite partial-sum data; the shared tail modulus and majorant budget then place those partial sums inside the $\mathsf{UniformCauchySeries}^{\uparrow}$ route before P or S can be consumed.

The product and addition handoff is read through K and Q. The Cauchy-product row supplies the series-product route, while the Cauchy-real-field row supplies the finite arithmetic surface that the addition law may inspect.

Finally, T, C, L, N transport, replay, source, and name the same displayed rows. Because all endpoint, equality, and arithmetic reads are already forced through the listed rows, none of the excluded host or quotient principles is exported by the local $\mathsf{NameCert}_{\mathsf{RealExponential}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.RealExponentialUp.RealExponentialTasteGate single carrier alignment`

Theorem 1816.3 (RealExponential series route)**.** *Every public read of an accepted* $\mathsf{RealExponential}^{\uparrow}$ *packet factors through*

$$\mathsf{RealPowerSeries}^{\uparrow},\quad \mathsf{UniformCauchySeries}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{CauchyRealField}^{\uparrow},\quad \mathsf{RealEquality}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

before any exponential value is consumed. Consequently the public route names the exponential series by finite coefficient, modulus, product, arithmetic, equality, and real-seal data, not by a host transcendental primitive.

Proof. Start with a public exponential read. The coefficient ledger A and factorial budget F identify the displayed $1/n!$ coefficients, while the bounded window B fixes the finite evaluation scope.

The read then enters the real power-series route through P. The power-series packet exposes the coefficient stream, selected powers, partial sums, and endpoint request without selecting a host infinite sum.

Uniform convergence is supplied next by U and M. These rows force the partial sums to pass through $\mathsf{UniformCauchySeries}^{\uparrow}$ and its displayed tail budget before any endpoint-facing real seal is available.

For law-facing reads, K and Q provide the only product and addition handoff. The Cauchy-product route supplies the multiplication of series windows, and the Cauchy-real-field route supplies the finite real-arithmetic operations that compare the result.

The final public comparison reads R and S. Thus equality is mediated by $\mathsf{RealEquality}^{\uparrow}$, and value consumption is mediated by the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows T, C, L, N preserve this route as transport, replay, provenance, and local naming, so no host transcendental function or ambient completeness principle can be read. □

Falsifiable prediction: Every $\mathsf{RealExponential}^{\uparrow}$ public read factors through a coefficient ledger for $1/n!$, factorial and reciprocal dyadic budgets, a bounded real-window row, a uniform Cauchy-series modulus, a majorant tail budget, a Cauchy-product law route, a Cauchy-real-field arithmetic row, a RealPowerSeries evaluation row, a RealEquality classifier row, a RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: realpowerseries, uniform-cauchy-series, cauchyproduct, cauchy-real-field, real-equality, real: $\mathsf{RealExponential}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the fixed exponential coefficient ledger, factorial reciprocal budget, bounded-window uniform convergence route, Cauchy-product addition-law handoff, arithmetic seal, equality classifier, endpoint real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealExponential}^{\uparrow}}$.

Constructive story: A RealExponential$^{\uparrow}$ packet is generated from the coefficient ledger $a_n = 1/n!$, factorial and reciprocal dyadic budgets, a bounded real-window row, a uniform Cauchy-series modulus, a majorant tail budget, a Cauchy-product law route, Cauchy-real-field arithmetic rows, a RealPowerSeries evaluation row, a RealEquality classifier row, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealExponential}^{\uparrow}}$, seed)

Scope closed: Definition 1816.1, Definition 1816.2, and Definition 1816.3 seed the finite exponential power-series route through existing BEDC real-series and Cauchy-series infrastructure.

Formal verification: FormalStatus(NameCert$_{\text{RealExponential}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.RealExponentialUp`

Bridge status: none

Not claimed: This chapter does not close host transcendental functions, analytic continuation, quotient real equality, selected infinite sums, countable choice, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, uniform Cauchy convergence, product and addition law handoff, nonescape, and TasteGate field faithfulness for BEDC.Derived.RealExponentialUp.

Chapter 1817

A Concrete Naming Certificate for $\mathsf{ExpLogBisection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ExpLogBisection}^{\uparrow}$ packet records the finite bisection route that makes logarithm a constructive inverse read for exponential on positive real data. It consumes the exponential packet of chapter 1816, the logarithm packet of chapter 2315, the dyadic bracket ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the inverse route by finite brackets and comparison rows, not by a host exponential, host logarithm, quotient real equality, selected infinite bisection branch, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1817.1 (ExpLogBisection carrier)**.** *An* $\mathsf{ExpLogBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, I, B, E, L, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the positive* $\mathsf{Real}^{\uparrow}$ *input row whose logarithm is being named.*
2. *I is the finite dyadic initial bracket row for the inverse search.*
3. *B is the retained bisection ledger, recording midpoint comparisons and selected subintervals.*
4. *E is the* $\mathsf{RealExponential}^{\uparrow}$ *comparison row used to evaluate the displayed midpoint endpoints.*
5. *L is the* $\mathsf{RealLogarithm}^{\uparrow}$ *endpoint row read only after the bisection ledger and exponential comparisons are displayed.*
6. *W records the* $\mathsf{StreamName}^{\uparrow}$ *finite windows for the retained dyadic endpoints.*
7. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *width, midpoint, and comparison tolerances used by the bisection ledger.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the dyadic endpoint windows.*
9. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row for the inverse endpoint and its exponential comparison with X.*
10. *H records componentwise* hsame *transport for all displayed inverse-route rows.*
11. *C records displayed* Cont *replay from positive input and brackets through exponential comparison, regular readback, logarithm row, and real seal.*

12. *P is the* Pkg *provenance over* $\mathsf{ExpLogBisection}^{\uparrow}$, $\mathsf{RealExponential}^{\uparrow}$, $\mathsf{RealLogarithm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ExpLogBisection}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, I, B, E, L, W, D, R, Q, H, C, P, N$. It has no coordinate for host analytic functions, quotient real equality, selected infinite branches, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.ExpLogBisectionUp`

Theorem 1817.2 (ExpLogBisection NameCert obligations)**.** *For every accepted* $\mathsf{ExpLogBisection}^{\uparrow}$ *carrier $G = (X, I, B, E, L, W, D, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ExpLogBisection}^{\uparrow}}$ *surface consists exactly of positive input exposure, finite initial bracket exposure, retained bisection rows,* $\mathsf{RealExponential}^{\uparrow}$ *comparison rows,* $\mathsf{RealLogarithm}^{\uparrow}$ *endpoint routing,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the accepted carrier. The positive input and bracket data are exactly X, I, B, so the inverse search begins with a displayed finite bracket ledger.

The comparison and window rows are E, W, D, R. Exponential comparisons are evaluated through E, while the dyadic widths, midpoint tolerances, finite windows, and regular readback are exposed through D, W, R.

The endpoint and seal rows are L, Q. They identify the logarithm endpoint and the real equality-facing seal only after the bisection comparison rows have been displayed.

Finally H, C, P, N preserve the same rows under transport, replay, provenance, and naming. Since the carrier has no host analytic-function or selected-branch coordinate, the listed rows exhaust the local certificate. □

Theorem 1817.3 (ExpLogBisection inverse handoff)**.** *Every public inverse read of an accepted* $\mathsf{ExpLogBisection}^{\uparrow}$ *carrier factors through the finite route*

$$X \longmapsto I, B \longmapsto E, D, W \longmapsto R \longmapsto L \longmapsto Q.$$

Thus the consumer of $\exp(\log(X)) \sim_{\mathsf{Real}^{\uparrow}} X$ *receives a positive real input, dyadic brackets, exponential comparison rows, StreamName windows, RegSeqRat readback, logarithm endpoint row, and Real seal before any inverse conclusion is inspected.*

Proof. Start with the carrier projection. The inverse route has no access to a completed host logarithm; it first displays the positive input X and the finite bracket rows I, B.

The retained bracket is justified through E, D, W, R. These rows provide the exponential comparisons, dyadic width budgets, finite endpoint windows, and regular rational readback that make the bisection endpoint visible.

The logarithm and real-seal handoff is then L, Q. The endpoint row records the logarithm packet, and the real seal records the equality-facing inverse comparison with the original input.

The structural rows transport and replay exactly that route. No quotient equality, selected infinite branch, countable-choice schedule, or ambient completeness theorem is exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ExpLogBisection}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ExpLogBisection}^{\uparrow}$ packet is generated from a positive RealUp input, dyadic initial and retained bracket rows, RealExponential comparison rows, a RealLogarithm endpoint row, StreamName windows, DyadicRatCore

tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ExpLogBisection}\uparrow}$, seed)

Scope closed: Definition 1817.1, Definition 1817.2, and Definition 1817.3 seed the finite bisection inverse route linking logarithm and exponential over positive RealUp data.

Formal verification: FormalStatus(NameCert$_{\text{ExpLogBisection}\uparrow}$, formalTarget)

Lean target: `BEDC.Derived.ExpLogBisectionUp`

Bridge status: none

Not claimed: This chapter does not close host exponential or logarithm functions, quotient real equality, selected infinite bisection branches, countable choice, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-domain exposure, dyadic bracket exactness, exponential comparison routing, StreamName and RegSeqRat readback, logarithm endpoint handoff, Real seal nonescape, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.ExpLogBisectionUp.

Chapter 1818

A Concrete Naming Certificate for $\mathsf{CauchyRateLattice}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRateLattice}^{\uparrow}$ packet records the finite meet and join surface for Cauchy-rate schedules before any Real or completion consumer may read a terminal seal. It sits over the displayed domination route of chapter 1814, the shared-bound interface of chapter 462, the uniform schedule surface of chapter 1741, the finite-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives later Real and Completion certificates one finite lattice carrier for comparing and combining displayed Cauchy-rate schedules.

Definition 1818.1 (Cauchy rate lattice carrier)**.** *A* $\mathsf{CauchyRateLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, M, J, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source Cauchy-rate schedule rows whose precision requests, source thresholds, and monotone reuse entries are visible to the packet.*
2. *M is the meet or shared-bound row. It records the finite threshold cells that both source schedules may use as a common Cauchy window before a downstream read consumes either source.*
3. *J is the displayed join or dominating-rate row. It records the finite threshold cells that dominate both A and B on the same precision surface.*
4. *D is the pair of* $\mathsf{CauchyRateDominance}^{\uparrow}$ *ledgers witnessing that M is readable from the two sources and that J dominates both source schedules through displayed comparison rows.*
5. *S is the selected* $\mathsf{StreamName}^{\uparrow}$ *window family opened only at requests certified by A, B, M, J, D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the selected windows and the visible dyadic tolerance entries.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source schedules, meet row, join row, dominance ledgers, selected windows, and regular readback have all been displayed.*
8. *H records componentwise* hsame *transport for the two source schedules, meet row, join row, dominance ledgers, selected windows, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through A, B, M, J, D, S, R, E and then through the structural rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRateLattice}^{\uparrow}$, $\mathsf{CauchyRateDominance}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateLattice}^{\uparrow}}$ *row.*

The classifier accepts only packets whose reads pass through these twelve rows. It has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRateLatticeUp.CauchyRateLatticeTasteGate single carrier alignment`

Theorem 1818.2 (Cauchy rate lattice NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateLattice}^{\uparrow}$ *carrier*

$$L = (A, B, M, J, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRateLattice}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two source schedules, meet row, join row, dominance ledgers, selected windows, regular readback, terminal seal, transport, replay, provenance, and local naming rows;*

2. *meet ledger exactness, requiring M to be read as the displayed shared-bound row licensed by the source schedules and the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *surface;*

3. *join ledger exactness, requiring J to be read as the displayed dominating-rate row whose comparisons to A and B pass through the* $\mathsf{CauchyRateDominance}^{\uparrow}$ *ledgers in D;*

4. *uniform-window compatibility, requiring selected windows in S to be licensed by the same finite precision surface used by M, J, and the* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *schedule row;*

5. *componentwise transport, requiring H to preserve $A, B, M, J, D, S, R, E, C, P, N$ together;*

6. *non-escape, requiring the certificate to export no quotient stream equality, selected host limit, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No lattice obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyRateLatticeUp.CauchyRateLatticeCarrier namecert obligations`

Proof. Project the carrier of Definition 1818.1. The projection exposes exactly two source schedules, one shared-bound row, one dominating-rate row, the dominance ledgers, the selected windows, the regular rational readback, the terminal seal, and the structural rows.

The meet obligation is read first from A and B. A precision request cites both source thresholds, then reads M as the common row supplied by the CauchyModulusMeet surface. No downstream window is opened until this shared-bound row is visible.

The join obligation is read through J and D. The dominance ledgers in D display the comparisons from both source schedules to the same dominating row, so a consumer cannot replace J by a detached rate selector.

Uniform-window compatibility follows from the displayed route. The selected windows in S are downstream of M, J, D, and the uniform schedule row fixes the finite precision surface on which the meet and join are consumed. The readback row R and terminal seal E therefore see the same schedule surface.

Finally, H, C, P, N only transport, replay, source, and name the displayed packet. Since the carrier has no additional analytic coordinate, the local NameCert cannot export quotient stream equality, selected host limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Theorem 1818.3 (Cauchy rate lattice meet-join stability)**.** *Let*

$$L = (A, B, M, J, D, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyRateLattice}^{\uparrow}$ *carrier. Any Real-facing or completion-facing consumer that reads the meet row M or the join row J must factor through the same displayed source pair A, B, dominance ledger row D, selected-window row S, regular readback row R, and structural rows H, C, P, N. Thus the meet and join operations are stable under componentwise* hsame *transport and displayed* Cont *replay inside the packet.*

Proof. Begin with a consumer of M. By Definition 1818.2, the meet row is not an independent tail selector; it is read only after the source pair A, B has been displayed and only through the shared-bound ledger.

Now consider a consumer of J. The join row is likewise downstream of the same source pair, and its dominance evidence is stored in D. Hence the join read has no route that bypasses A, B, D.

The selected-window and readback rows are common to both reads. Since S is opened only after the meet, join, and dominance ledgers are visible, and since R reads only those selected windows, both operations reach the endpoint through the same finite window and readback route.

Transport stability is componentwise. The row H preserves all carrier coordinates together, and C replays only the displayed order from precision request to seal. Therefore both meet and join reads remain stable under the permitted hsame and Cont structure. □

Theorem 1818.4 (Cauchy rate lattice Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read out of an accepted* $\mathsf{CauchyRateLattice}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyRateDominance}^{\uparrow},\ \mathsf{CauchyModulusMeet}^{\uparrow},\ \mathsf{UniformCauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqR}$$

Equivalently, the read first consumes the two source schedules A, B, the meet row M, the join row J, the dominance ledgers D, the selected windows S, and the regular readback R, and only then reaches the terminal seal E and the structural rows H, C, P, N.

Proof. Take an endpoint-facing read admitted by the carrier. Projection of the carrier gives no direct coordinate from a precision request to E; the only endpoint route is through the displayed rows A, B, M, J, D, S, R.

The meet and join rows determine the lattice surface. The meet row reads the common Cauchy window supplied by the shared-bound surface, while the join row reads the dominating schedule supplied by the dominance ledgers. The uniform modulus row fixes the common finite precision surface used by both.

The selected windows and regular readback are then forced. The row S opens only windows licensed by the displayed lattice rows, and R reads those windows through the dyadic tolerance ledger before a $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Thus E is reached only through the stated sequence of concrete surfaces. The remaining rows H, C, P, N transport, replay, source, and name the same route, and no quotient stream, selected host limit, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checker can be inserted. □

Falsifiable prediction: An accepted $\mathsf{CauchyRateLattice}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first reading the source Cauchy-rate schedules, the shared meet row, the displayed join row, the dominance ledgers, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: cauchy-rate-dominance, cauchymodulusmeet, uniform-cauchy-modulus, streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole twelve-row packet $L = (A, B, M, J, D, S, R, E, H, C, P, N)$, because each sibling contributes only domination, shared-bound, uniform-schedule, window, tolerance, readback, or seal data while this chapter binds meet, join, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRateLattice}^{\uparrow}$ packet is generated from two source Cauchy-rate schedules, a displayed meet row, a displayed join row, dominance ledgers, selected StreamName windows, DyadicRatCore tolerance entries, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1818.1, Definition 1818.2, Definition 1818.3, and Definition 1818.4 seed the finite meet and join surface for Cauchy-rate schedules through the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, meet ledger exactness, join ledger exactness, uniform-window compatibility, componentwise transport, non-escape, meet-join stability, and Real route factorization for BEDC.Derived.CauchyRateLatticeUp.

Chapter 1819

A Concrete Naming Certificate for $\mathsf{EntourageBase}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on uniform structures, metric entourages, and constructive completion, the $\mathsf{EntourageBase}^{\uparrow}$ packet records the finite basis interface used by the L10 completion route before Cauchy filters or quotient completions are allowed to read it. It is anchored in the uniform entourage surface of chapter 339, the metric-ball naming surface of chapter 1510, and the finite observation windows of chapter 126. The packet names a displayed entourage-basis row and its dyadic radius ledger; it is not a quotient completion, not a selected limit, not an open-cover compactness argument, not countable choice, and not ambient completeness.

The finite basis route consumes the $\mathsf{UniformSpace}^{\uparrow}$ source row of chapter 339, the $\mathsf{Met}^{\uparrow}$ ball row of chapter 57, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301 before any completion-facing consumer may inspect the packet.

Definition 1819.1 (EntourageBase carrier)**.** *An* $\mathsf{EntourageBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, M, B, R, W, D, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, exposing only the finite entourage data and its displayed containment and refinement witnesses.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row from which metric-ball entourage requests may be read before any completion-facing consumer is present.*
3. *B is the finite entourage-basis row. It lists the basis entourages accepted by the packet, their finite-intersection refinement witnesses, and the containment route back to U.*
4. *R is the dyadic radius ledger, recording the admitted radii, the smaller-radius relation, and the monotone route from each radius request to the row B.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by finite completion-route consumers that need named observations rather than host-level points.*
6. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to the dyadic radius comparisons in R.*
7. *H records componentwise* hsame *transport for the uniform source, metric source, basis row, radius ledger, finite windows, readback row, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a requested entourage scale through* $U, M, B, R, W, D, H, P, N$.
9. *P is the* Pkg *provenance over* $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EntourageBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $U, M, B, R, W, D, H, C, P, N$. *The carrier has no coordinate for quotient completion, selected limits, open-cover compactness, countable-choice selectors, ambient completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EntourageBaseUp.EntourageBaseTasteGate single carrier alignment`

Theorem 1819.2 (EntourageBase NameCert obligations)**.** *For every accepted* $\mathsf{EntourageBase}^{\uparrow}$ *carrier*

$$E = (U, M, B, R, W, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EntourageBase}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed UniformSpace source, MetricSpace source, entourage-basis row, dyadic radius ledger, StreamName window row, RegSeqRat readback, transport, replay, provenance, and local naming row;*
2. *basis containment, requiring every basis entourage in B to carry a displayed containment witness into the UniformSpace entourage row U;*
3. *finite-intersection refinement, requiring every accepted pair of basis rows to refine through a displayed row of B before any completion-route consumer may inspect it;*
4. *radius monotonicity, requiring every smaller-radius request recorded by R to factor through the same displayed basis row B and not through an ambient metric completion;*
5. *transport, requiring H to preserve the source rows, basis row, radius ledger, finite windows, readback, replay, provenance, and local naming componentwise under* hsame*;*
6. *replay, requiring C to repeat the route from a requested entourage scale through U, M, B, R, W, D before any named consumer is exposed;*
7. *provenance, requiring P to source only the listed rows and to export no quotient completion, selected limit, open-cover compactness, countable choice, or ambient completeness principle;*
8. *local naming, requiring N to name exactly this finite surface and no external completion object.*

No local NameCert obligation is stored outside these rows.

Theorem 1819.3 (EntourageBase finite radius refinement)**.** *Let $E = (U, M, B, R, W, D, H, C, P, N)$ be an accepted* $\mathsf{EntourageBase}^{\uparrow}$ *carrier. Every accepted smaller-radius request in R factors through the same displayed route*

$$U, M \longmapsto B \longmapsto R \longmapsto W, D \longmapsto H, C, P, N.$$

In particular, the refinement is witnessed by the finite entourage-basis row and dyadic ledger of the packet. It does not factor through quotient completion, selected limits, open-cover compactness, countable choice, or ambient completeness.

Theorem 1819.4 (EntourageBase UniformSpace dependency route)**.** *Every accepted* $\mathsf{EntourageBase}^{\uparrow}$ *completion-facing consumer must first display the same* $\mathsf{UniformSpace}^{\uparrow}$ *source row,* $\mathsf{Met}^{\uparrow}$ *ball row,*

$\mathsf{StreamName}^{\uparrow}$ *window row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carried by* U, M, W, D*. The consumer then reaches the basis row* B*, radius ledger* R*, transport* H*, replay* C*, provenance* P*, and local name* N *through that finite dependency route, and not through an ambient completion object.*

Closure status for $\mathsf{NameCert}_{\mathsf{EntourageBase}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EntourageBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1819.1, Definition 1819.2, Definition 1819.3, and Definition 1819.4 seed the finite uniform-basis interface through which UniformSpace and MetricSpace entourage data reaches StreamName and RegSeqRat completion-route consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EntourageBase}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EntourageBaseUp.EntourageBaseTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, open-cover compactness, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis containment, finite-intersection refinement, radius monotonicity, transport, replay, provenance, and local naming for BEDC.Derived.EntourageBaseUp.

Constructive story: An $\mathsf{EntourageBase}^{\uparrow}$ packet is generated from one UniformSpace source row, one MetricSpace source row, a finite entourage-basis row, a dyadic radius ledger, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ is consumed by chapter 301 and chapter 1809 through regular-rational classifier transport and dense readback; the seed closurestatus is carried by the child namecert construction file, while this hub only routes RegSeqRat and Real classifier dependencies.

Chapter 1820

A Concrete Naming Certificate for $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ packet records the finite classifier surface by which two regular Cauchy names may be compared before the $\mathsf{Real}^{\uparrow}$ seal exposes any equality-facing consumer. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a non-quotient equality surface for the L10 Real and Completion route; it is not quotient stream equality, host real equality, a selected limit, countable choice, or ambient completeness.

The same classifier is consumed by chapter 1809 and chapter 1847 as the finite comparison route that must be displayed before density or Cauchy-space reads are exported.

The setoid dependency lattice also consumes the finite regular-rational readback of chapter 301, the Cauchy-name comparison surface of chapter 1412, and the terminal Real seal route of chapter 38; those sibling rows are displayed before the setoid relation becomes public to any Real or Completion consumer.

It is also a horizontal dependency of chapter 301 and the regular-Cauchy uniqueness route before any $\mathsf{Real}^{\uparrow}$-facing seal is read. A consumer first reads the finite classifier row T, the open-tail route through $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks R_0, R_1, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D; only then may regular-Cauchy uniqueness consume the comparison. The route refuses quotient stream equality, so representative independence remains a displayed classifier handoff rather than a host quotient.

Definition 1820.1 (CauchyEquivalenceSetoid carrier)**.** *A* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows. Each row lists only the visible observation windows that a classifier request may inspect.*

2. *R_0 and R_1 are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows. They expose the regular rational values attached to the stream windows before any real-name seal is consumed.*

3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the admitted dyadic precision requests, comparison witnesses, and refinement rows used to compare the two regular names.*

4. *T is the setoid-test row. It stores reflexivity, symmetry, and transitivity checks only as finite classifier rows over S_0, S_1, R_0, R_1, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-boundary row reached after the stream windows, readback rows, dyadic ledger, and setoid-test row have been displayed.*
6. *H records componentwise* hsame *transport for the two stream rows, the two readback rows, the dyadic ledger, the setoid-test row, the seal row, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from an equality-facing tolerance request through $S_0, S_1, R_0, R_1, D, T, E, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidCarrier semantic name certificate`

The regular-sequence classifier route of chapter 301 supplies the readback side of the setoid test, while the terminal seal discipline of chapter 38 fixes where equality-facing consumers may leave the comparison surface. Thus $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ is a displayed classifier over those sibling rows, not a quotient object; every public read keeps the two $\mathsf{RegSeqRat}^{\uparrow}$ presentations and the downstream $\mathsf{Real}^{\uparrow}$ seal visible before completion or equality consumers inspect the packet.

Theorem 1820.2 (CauchyEquivalenceSetoid obligation surface)**.** *For every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N),$$

the obligation surface consists exactly of carrier admission, regular-rational classifier transport, tail-stability of the displayed setoid tests, the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *readback boundary, and ledger non-escape through $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$. No obligation may be discharged by quotient stream equality, host real equality, a selected representative, countable choice, or ambient completeness.*

Proof. Carrier admission is the projection of Definition 1820.1. The projection supplies the two stream windows, two regular-rational readbacks, dyadic tolerance ledger, setoid-test row, Real seal boundary, and structural rows.

Regular-rational classifier transport is read from the rows R_0, R_1, T, H, C. The displayed transport row H and replay row C carry the same classifier tests over the two $\mathsf{RegSeqRat}^{\uparrow}$ presentations, so the comparison never becomes a host quotient relation.

Tail-stability is supplied by the classifier-law rows and by the regular-Cauchy handoff of Definition 1820.3 and Definition 1820.6. These force reflexive, symmetric, transitive, and uniqueness-facing reads to remain on $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$.

Finally the RegSeqRat and Real dependency routes of Definition 1820.20 and Definition 1820.8 place R_0, R_1 before the Real seal E. Since the carrier has no coordinate for any excluded equality, selection, choice, or completeness object, every accepted obligation is a ledger projection from the displayed packet. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidCarrier obligation surface`

Theorem 1820.3 (CauchyEquivalenceSetoid classifier laws)**.** *For every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier*

$$Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N),$$

the classifier row T *supplies exactly the reflexive, symmetric, and transitive finite tests visible to the local* $\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}$ *surface:*

1. *reflexivity tests compare a stream window with itself through the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and the dyadic zero-tolerance entries of* D*;*
2. *symmetry tests reverse the two displayed stream-readback sides and reuse the same dyadic comparison ledger* D *with no second equality relation;*
3. *transitivity tests concatenate two displayed comparison witnesses through a common readback side and discharge the resulting tolerance request by the refinement rows of* D*;*
4. *every test is replayed by* C*, transported componentwise by* H*, sourced by* P*, and named by* N *before the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is consumed.*

No classifier law is stored outside the finite rows of Q*.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidClassifierLaws`

Proof. Project the carrier of Definition 1820.1. The projection exposes two finite StreamName windows, two RegSeqRat readback rows, one DyadicRatCore tolerance ledger, one setoid-test row, the Real seal boundary, and the structural rows.

For reflexivity, use the same stream row and the same readback row on both sides. The dyadic ledger supplies the displayed zero-tolerance and refinement entries, so the test is a row of T rather than a quotient assertion.

For symmetry, exchange the two stream-readback sides. The comparison witness remains in the same ledger D, and T records the reversed finite row as the visible classifier test.

For transitivity, concatenate the two displayed comparison rows through their shared middle readback side. The resulting tolerance request is discharged by the refinement entries already listed in D.

Finally H, C, P, N transport, replay, source, and name these finite rows, while E is available only after the rows have been displayed. Since Q has no quotient or host-equality coordinate, the local certificate contains exactly the listed classifier laws. □

Theorem 1820.4 (CauchyEquivalenceSetoid transport)**.** *Let* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier. Every* hsame *transport of an equality-facing classifier read is componentwise over*

$$S_0, S_1, R_0, R_1, D, T, E, C, P, N.$$

Thus transport preserves the StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal boundary, displayed replay, package provenance, and local NameCert rows before any equality consumer may inspect the result.

Proof. Start with a classifier read and its displayed hsame witness in H. By the carrier definition, H has no global equality coordinate; it stores only componentwise transports for the rows of Q.

Apply the transport row to $S_0, S_1, R_0, R_1, D, T, E$ in order. The stream windows remain finite, the readback rows remain the same regular rational interface, the tolerance ledger remains the same dyadic comparison surface, and the seal row remains downstream of those rows.

The replay, provenance, and local naming rows are transported last. Since C, P, N only replay, source, and name the displayed route, the transported read still factors through the same finite classifier surface before any equality-facing consumer can inspect it. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidTransport`

Theorem 1820.5 (CauchyEquivalenceSetoid NameCert surface)**.** *For every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *carrier admission for the two StreamName finite-window rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, setoid-test row, Real seal boundary, componentwise transport, replay, provenance, and local naming row;*
2. *StreamName window exactness, requiring every equality-facing observation to be one of the displayed windows in* S_0 *or* S_1*;*
3. *RegSeqRat readback exactness, requiring every rational comparison to pass through* R_0 *and* R_1 *before it reaches the dyadic ledger;*
4. *DyadicRatCore tolerance exactness, requiring every accepted equality test to be discharged by an entry of* D*;*
5. *classifier-law stability, requiring reflexivity, symmetry, and transitivity to be exactly the finite tests stored in* T*;*
6. *Real seal nonescape, requiring* E *to be consumed only after the window, readback, tolerance, and classifier-law rows have been displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*.*

The surface exports no quotient stream equality, host real equality, selected limit, countable-choice selector, ambient completeness witness, Lean verification, or bridge check.

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidNameCertSurface`

Proof. Carrier admission is the displayed tuple of Definition 1820.1. It gives exactly the stream, readback, dyadic, classifier, seal, transport, replay, provenance, and naming rows.

Window exactness and readback exactness are immediate from the source rows. Every observation is first a finite StreamName window in S_0 or S_1, and every rational comparison then reads the corresponding RegSeqRat row R_0 or R_1.

Tolerance exactness is supplied by D. A classifier request can only use dyadic precision entries and refinement witnesses present in that ledger.

Use Definition 1820.3 for the setoid tests and Definition 1820.4 for the transport discipline. These place the law rows and transported reads inside the same finite packet.

The Real seal row E is downstream of the displayed classifier route, and C, P, N only replay, source, and name that route. Therefore the local NameCert surface is exhausted by the listed obligations and cannot export any of the excluded completion or equality objects. □

Theorem 1820.6 (CauchyEquivalenceSetoid regular-Cauchy uniqueness handoff)**.** *Any regular-Cauchy uniqueness consumer comparing two* $\mathsf{Real}^{\uparrow}$*-facing regular rational names through an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ *must first read the two* $\mathsf{StreamName}^{\uparrow}$ *windows* S_0, S_1*, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_0, R_1*, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance ledger D, the setoid-test row T, the Real$^{\uparrow}$ *seal boundary E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N. Representative independence is exported only through this non-quotient classifier route.*

Proof. Project the carrier of Definition 1820.1. The two regular rational presentations are visible only through the two finite StreamName windows, the two RegSeqRat readbacks, the dyadic tolerance ledger, and the setoid-test row.

Apply the classifier laws of Definition 1820.3. Reflexivity, symmetry, and transitivity remain finite tests in T over S_0, S_1, R_0, R_1, D, so the uniqueness consumer receives classifier evidence rather than quotient stream equality.

Use the transport theorem and NameCert surface, Definition 1820.4 and Definition 1820.5. They keep the comparison stable under componentwise hsame, replay the same displayed route through C, and source and name it through P, N before the Real seal can be consumed.

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient completeness. Thus representative independence is a displayed handoff through $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$ and not an appeal to any excluded equality or limit principle. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRegularCauchyUniquenessHandoff`

Theorem 1820.7 (CauchyEquivalenceSetoid RealEquality consumer route)**.** *Every* RealEquality$^{\uparrow}$ *consumer of an accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *packet $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ must read the two* StreamName$^{\uparrow}$ *windows S_0, S_1, the two* RegSeqRat$^{\uparrow}$ *readbacks R_0, R_1, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, the setoid-test row T, the* Real$^{\uparrow}$ *seal boundary E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N before it reaches the* RealEquality$^{\uparrow}$ *classifier boundary of chapter 1636. The consumer route has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRealEqualityConsumerRoute`

Proof. Project the carrier of Definition 1820.1. The visible inputs are exactly the two stream windows, two regular-rational readbacks, one dyadic ledger, the setoid-test row, the real seal, and the structural support rows.

Apply the classifier laws of Definition 1820.3. They require every reflexive, symmetric, or transitive setoid read to remain a finite test over S_0, S_1, R_0, R_1, D before the seal row is consumed.

Use Definition 1820.4 for the componentwise handoff and Definition 1820.5 for local certification. The rows H, C, P, N transport, replay, source, and name the same route, while E remains the downstream Real$^{\uparrow}$ boundary feeding the RealEquality$^{\uparrow}$ classifier.

Since the accepted packet lists no quotient, host-equality, selected-limit, choice, or ambient-completeness coordinate, a RealEquality$^{\uparrow}$ consumer can inspect only the displayed finite route before reaching the sibling equality classifier. □

Theorem 1820.8 (CauchyEquivalenceSetoid Real seal forward route)**.** *Every accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *read reaches a* Real$^{\uparrow}$ *consumer only after StreamName finite windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the classifier row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row are visible. Concretely, the Real-facing route is*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto H, C, P, N \longmapsto E,$$

where E is the terminal $\mathsf{Real}^{\uparrow}$ *seal and no quotient stream equality, host real equality, selected representative, countable-choice selector, or ambient completeness row is exported.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRealSealForwardRoute`

Proof. Project the carrier of Definition 1820.1. The projection displays the StreamName windows S_0, S_1, RegSeqRat readbacks R_0, R_1, DyadicRatCore ledger D, classifier row T, Real seal E, and structural rows H, C, P, N.

Apply the classifier laws and NameCert surface, Definition 1820.3 and Definition 1820.5. They keep every comparison test inside the finite windows, readbacks, tolerance ledger, and classifier row before the Real seal can be consumed.

Use the transport theorem and RealEquality consumer route, Definition 1820.4 and Definition 1820.7. The transport row keeps the route componentwise stable, while the equality consumer route confirms that the Real-facing seal is reached through the same displayed packet rather than through quotient equality or host selection principles. □

Theorem 1820.9 (CauchyEquivalenceSetoid downstream equality dependency)**.** *Let $Q = (S_0, S_1, R_0, R_1, D, T, E,$ be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier. Every* RealClassifier *or* $\mathsf{MetricCompletion}^{\uparrow}$ *equality-facing consumer of Q reads the setoid surface only through the route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before any completion-facing read is available. In particular, the consumer first inspects $\mathsf{StreamName}^{\uparrow}$ *windows, then* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then the* $\mathsf{Real}^{\uparrow}$ *seal boundary that feeds Real equality or completion consumers.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidDownstreamEqualityDependency`

Proof. Start from the NameCert surface of Definition 1820.5. It says that every equality-facing observation is one of the displayed windows in S_0 or S_1, every rational comparison passes through R_0 and R_1, every accepted test is discharged by D, and the $\mathsf{Real}^{\uparrow}$ seal row E is downstream of those rows.

Use Definition 1820.3 for the finite equality tests stored in T. Reflexivity, symmetry, and transitivity therefore remain classifier rows over the same windows, readback rows, and dyadic tolerance ledger rather than a detached host equality.

Use Definition 1820.4 for the componentwise handoff to downstream readers. The sibling dependency paragraph below the surface theorem places this same route next to the StreamName, RegSeqRat, DyadicRatCore, and Real chapters, so a RealClassifier or $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives only the routed packet $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$ before any completion-facing read. □

Theorem 1820.10 (CauchyEquivalenceSetoid psame RealClassifier bridge)**.** *Let $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C$ be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier packet. Any accepted classifier row consumed as a* psame *comparison reaches the* RealClassifier *boundary only through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E, H, C, P, N.$$

Conversely, any RealClassifier*-facing read backed by exactly those displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, setoid-test row, Real seal boundary, componentwise transport, replay, provenance, and local naming rows is a legitimate* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *consumer route. The bridge exports no quotient stream equality, host real equality, selected limit, countable choice, or ambient completeness.*

Lean4 ✓ BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidPsameRealClassifierBridge

Proof. For the forward direction, begin with the classifier laws of Definition 1820.3. A psame comparison can use only the reflexive, symmetric, and transitive finite tests in T over S_0, S_1, R_0, R_1, D.

Use the transport theorem, Definition 1820.4, to carry the comparison componentwise through H while preserving the same StreamName windows, RegSeqRat readbacks, dyadic ledger, classifier row, seal boundary, replay, provenance, and local naming rows.

Apply the NameCert surface and RealEquality consumer route, Definition 1820.5 and Definition 1820.7. They place the Real-facing classifier boundary downstream of the displayed finite route and exclude any detached quotient or host-equality coordinate.

For the converse direction, suppose a RealClassifier-facing read is backed by the rows $S_0, S_1, R_0, R_1, D, T, E, H, C$ in the displayed order. The NameCert surface admits exactly those rows as the local certificate surface, and the downstream equality dependency theorem, Definition 1820.9, identifies the same route as the equality-facing consumer route. Therefore the read is a legitimate $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ consumer route. □

Theorem 1820.11 (CauchyEquivalenceSetoid separated-completion forward link)**.** *Every* RegularCauchyCompletio *or* $\mathsf{SeparatedMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P,$ *first reads the two* $\mathsf{StreamName}^{\uparrow}$ *windows* S_0, S_1*, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_0, R_1*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the setoid-test row* T*, the* $\mathsf{Real}^{\uparrow}$ *seal boundary* E*, and the structural rows* H, C, P, N*. The route feeds the separated-completion corridor of chapter 1513, the sequence-equivalence boundary of chapter 1412, and the separated-metric classifier of chapter 1411; it exposes no quotient stream equality, selected representative, host real equality, countable choice, or ambient completion.*

Proof. Begin with the downstream equality dependency of Definition 1820.9. It orders every equality-facing completion read through the two stream windows, regular rational readbacks, dyadic tolerance ledger, classifier row, real seal, and structural rows.

Use the classifier laws of Definition 1820.3. They say that reflexive, symmetric, and transitive comparison data remain finite tests in T over S_0, S_1, R_0, R_1, D, which is the same comparison route consumed by the sequence-equivalence boundary of chapter 1412.

Apply the NameCert surface of Definition 1820.5 and the transport theorem Definition 1820.4. These keep the separated-completion and separated-metric consumers inside E, H, C, P, N after the classifier route, matching the corridor of chapter 1513 and chapter 1411.

No carrier row remains from which a quotient equality, selected representative, host equality, countable-choice schedule, or ambient completion can be read. Thus every separated-completion or separated-metric consumer first traverses the displayed setoid route. □

Lean4 ✓ BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidSeparatedCompletionForwardLink

Theorem 1820.12 (CauchyEquivalenceSetoid separated metric factorization)**.** *Every* $\mathsf{SeparatedMetric}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet factors its equality-facing read through the displayed route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before the separated metric classifier is available. Thus the consumer first reads StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the classifier row, and the RealUp seal, and only then reaches the separated metric classifier boundary.

Proof. Project the carrier of Definition 1820.1. The projection exposes exactly $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$, so every consumer starts from the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore ledger, classifier row, Real seal, and structural rows.

Apply the NameCert surface and downstream equality dependency, Definition 1820.5 and Definition 1820.9. These force equality-facing reads through the windows, readbacks, tolerance ledger, classifier row, and Real seal before a completion-facing read can be inspected.

Apply the separated-completion forward link, Definition 1820.11. It sends $\mathsf{SeparatedMetric}^{\uparrow}$ and $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ consumers through the same displayed route before the separated metric classifier of chapter 1411 is reached.

No row of the carrier stores quotient stream equality, a selected representative, host equality, countable choice, or ambient completion. Therefore the separated metric factorization is exhausted by the displayed packet and cannot be replaced by any of those excluded objects. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidSeparatedMetricFactorization`

Theorem 1820.13 (CauchyEquivalenceSetoid RealSeparability backward link)**.** *Let $Q = (S_0, S_1, R_0, R_1, D, T, E, H$ be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier packet whose displayed rows include the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R_0, R_1 and the* $\mathsf{Real}^{\uparrow}$ *seal row E. Then the* $\mathsf{RealSeparability}^{\uparrow}$ *dense-route consumer can read Q through the same finite classifier surface*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

without importing quotient equality of Cauchy names, host real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1820.1. The rows available to a dense-route consumer are precisely the StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, setoid-test row, Real seal, componentwise transport, replay, provenance, and local naming.

Apply the NameCert surface and classifier laws, Definition 1820.5 and Definition 1820.3. They keep every comparison used by the dense route inside S_0, S_1, R_0, R_1, D, T before the seal boundary is reached.

Use the downstream equality dependency of Definition 1820.9. The same finite route is already the path supplied to Real-facing consumers, and the opening sibling route to chapter 1809 reads it as classifier data rather than as quotient equality.

Since Q contains no row for quotient Cauchy-name equality, host real equality, selected limits, countable choice, or ambient completeness, the $\mathsf{RealSeparability}^{\uparrow}$ consumer receives only the displayed dense-route classifier packet. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRealSeparabilityBackwardLink`

Theorem 1820.14 (CauchyEquivalenceSetoid RealSeparability stable-density handoff)**.** *Let $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier packet. Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-route consumer of Q preserves the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently any two Cauchy-equivalent $\mathsf{RegSeqRat}^{\uparrow}$ *presentations accepted by the same packet feed the same rational density window to the* $\mathsf{RealSeparability}^{\uparrow}$ *consumer through that route, without quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRealSeparabilityStableDensity`

Proof. Start with the NameCert surface of Definition 1820.5. It exhausts the accepted packet by the two stream windows, two regular-rational readbacks, dyadic tolerance ledger, classifier row, real seal, transport, replay, provenance, and local naming rows.

Apply the classifier laws of Definition 1820.3. Reflexive, symmetric, and transitive comparisons are finite tests in T over S_0, S_1, R_0, R_1, D, so a Cauchy-equivalent pair of regular rational presentations reaches the density consumer through the displayed classifier surface.

Use the downstream equality dependency of Definition 1820.9. It orders every Real-facing consumer read through the same route before the seal row E is available, so the rational density window cannot be chosen from a detached quotient equality or host real equality.

Finally apply the $\mathsf{RealSeparability}^{\uparrow}$ backward link of Definition 1820.13. The dense-route consumer already reads the classifier packet as finite route data; therefore the handoff preserves the same rational density window and excludes selected representatives, countable choice, and ambient completeness. □

Theorem 1820.15 (CauchyEquivalenceSetoid RealSeparability forward anchor)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-read consumer that uses an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must pass through the displayed classifier route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before it reads the separability surface of chapter 1809. Thus the density-facing handoff reaches $\mathsf{RealSeparability}^{\uparrow}$ *only through the two* $\mathsf{StreamName}^{\uparrow}$ *windows, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, dyadic tolerance ledger, classifier row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidRealSeparabilityForwardAnchor`

Proof. Project the accepted packet of Definition 1820.1. The only available dense-read data are the two stream windows, two regular-rational readbacks, dyadic ledger, classifier row, Real seal, and structural rows.

Apply the stable-density handoff and the backward link, Definition 1820.14 and Definition 1820.13. They identify the route consumed by a $\mathsf{RealSeparability}^{\uparrow}$ reader and place it before the downstream separability surface is inspected.

The local dependency route has no quotient equality, selected representative, countable-choice schedule, host real equality, or ambient completeness coordinate. Therefore the forward anchor into chapter 1809 is exactly the displayed classifier route. □

Theorem 1820.16 (CauchyEquivalenceSetoid tail transitivity)**.** *Let accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *tests share compatible* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers through the displayed classifier row* T*. Any downstream* $\mathsf{RealClassifier}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *equality-facing read receives the composed finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before any seal boundary is available. The composition is a finite classifier route through $S_0, S_1, R_0, R_1, D, T, E, H, C$ *not quotient stream equality, host real equality, a selected limit, countable choice, or ambient completeness.*

Proof. Use the classifier laws of Definition 1820.3. The transitivity row in T concatenates compatible finite comparison witnesses through the shared middle readback side, and the dyadic refinement rows in D discharge the resulting tolerance request.

Next apply the transport theorem Definition 1820.4. It keeps the two stream windows, the two regular-rational readbacks, the dyadic ledger, the classifier row, and the seal row componentwise stable under hsame handoff.

Finally use the downstream equality dependency theorem. It orders every RealClassifier or MetricCompletion$^{\uparrow}$ equality-facing read through the displayed stream, readback, dyadic, classifier, seal, transport, replay, provenance, and naming rows before any completion-facing boundary is reached. The carrier has no coordinate for quotient equality or host selection principles, so the composed tail-transitivity read remains inside the finite route. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidTailTransitivity`

Theorem 1820.17 (CauchyEquivalenceSetoid completion-tail composition)**.** *Let two compatible accepted equality-facing* CauchyEquivalenceSetoid$^{\uparrow}$ *tests share their* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, and* DyadicRatCore$^{\uparrow}$ *tolerance ledgers through the displayed classifier row. Their composed read reaches any completion-facing classifier only through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before any RealClassifier *or* MetricCompletion$^{\uparrow}$ *consumer may inspect the seal. The composition is a finite classifier route over the shared rows, not quotient stream equality, host real equality, a selected limit, countable choice, or ambient completeness.*

Proof. Use Definition 1820.3. The compatible tests compose by the transitivity row in T, and their reflexive and symmetric boundary reads remain finite classifier rows over the same windows, readbacks, and dyadic tolerance ledger.

Apply Definition 1820.4. It keeps the shared StreamName windows, RegSeqRat readbacks, DyadicRatCore ledger, classifier row, and seal boundary componentwise stable under the displayed hsame handoff.

Now use Definition 1820.16. It orders the composed finite route through $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$ before any downstream seal boundary is available.

Finally apply Definition 1820.9. Every RealClassifier or MetricCompletion$^{\uparrow}$ consumer receives only that displayed route, so completion-facing inspection cannot bypass the shared windows, readbacks, tolerance ledger, or classifier row. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidCompletionTailComposition`

Theorem 1820.18 (CauchyEquivalenceSetoid finite-chain composition)**.** *Any finite displayed chain of compatible* CauchyEquivalenceSetoid$^{\uparrow}$ *packets composes through repeated tail transitivity into one classifier route over the shared* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *tolerance ledgers, the terminal* Real$^{\uparrow}$ *seal row, componentwise transport, displayed replay, provenance, and local* NameCert *rows.*

The composed route is a finite classifier surface for downstream Real equality and completion consumers. It is not quotient equality of Cauchy streams, host real equality, a selected representative, a selected limit, countable choice, or ambient completeness.

Proof. Induct over the displayed finite chain. In the empty or singleton case, the route is the classifier row supplied by Definition 1820.3, transported componentwise by Definition 1820.4.

For the successor step, assume the initial segment has been composed into one classifier route over the displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming rows. Compose that route with the next compatible packet using Definition 1820.16.

Apply Definition 1820.17 to order the extended route before any completion-facing consumer inspects the seal. Since each step stays on the displayed classifier surface, no quotient equality, host equality, selected representative, selected limit, countable choice, or ambient completeness coordinate is introduced. □

Theorem 1820.19 (CauchyEquivalenceSetoid regular-chain induction)**.** *Any finite chain of accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier comparisons factors through repeated displayed rows*

$$S_i, R_i, D_i, T_i, E_i, H_i, C_i, P_i, N_i.$$

Its composite comparison remains a finite classifier route for Real equality and completion consumers. The induction never exports quotient stream equality, a selected representative, a selected limit, host real equality, countable choice, or ambient completeness.

Proof. Induct over the finite chain of accepted classifier comparisons. In the base case, the empty or singleton chain is the reflexive classifier row supplied by Definition 1820.3, with the carrier rows transported componentwise by Definition 1820.4.

For the successor step, assume the initial chain has already been factored through displayed rows $S_i, R_i, D_i, T_i, E_i, H_i, C_i, P_i, N_i$ and has produced one finite classifier route. The next accepted comparison contributes the same kind of rows, and the shared regular side is matched componentwise through the StreamName, RegSeqRat, and DyadicRatCore entries.

Apply the tail-transitivity and completion-tail composition theorems, Definition 1820.16 and Definition 1820.17. They compose the accumulated route with the next comparison while preserving the displayed windows, readbacks, tolerance ledgers, classifier rows, Real seal rows, transport, replay, provenance, and local naming.

Finally use the Real seal nonescape supplied by the downstream equality dependency and finite-chain composition theorems. The composite comparison remains a routed classifier packet before any Real equality or completion consumer may inspect it, so none of the excluded quotient, selection, host-equality, choice, or completeness objects can appear. □

Theorem 1820.20 (CauchyEquivalenceSetoid RegSeqRat route)**.** *Every completion-tail comparison exported by an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must pass through the sibling* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *routes of chapter 301 and chapter 38. The exported comparison is the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

rather than quotient stream equality, host real equality, a selected limit, countable choice, or ambient completeness.

Proof. Start from the classifier laws of Definition 1820.3. They force reflexive, symmetric, and transitive comparison reads to stay on the displayed S_0, S_1, R_0, R_1, D, T surface.

Use Definition 1820.4. It preserves the same StreamName windows, RegSeqRat readbacks, dyadic ledger, classifier row, Real seal, replay, provenance, and local naming under componentwise hsame comparison.

Apply Definition 1820.9 and Definition 1820.17. The first identifies the Real-facing equality consumer route, and the second orders completion-tail reads through the composed classifier surface before any completion consumer can inspect the seal.

Thus every exported completion-tail comparison must use the sibling RegSeqRat readback and Real seal routes. Since no carrier row records quotient equality, host real equality, selected limits, countable choice, or ambient completeness, the route cannot leave the finite packet. □

Theorem 1820.21 (CauchyEquivalenceSetoid finite sibling route)**.** *Every equality-facing consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must pass through the finite sibling route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyModulus}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

as displayed regular-Cauchy readback, finite tolerance and modulus rows, terminal Real seal, completion-facing classifier, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}$ *row. The route is concrete over the carrier rows* $S_0, S_1, R_0, R_1, D, T, E, H, C, P,$ *and exports no quotient equality, host equality, selected representative, selected limit, countable choice, or ambient completeness.*

Proof. Start from the NameCert surface of Definition 1820.5. It forces every equality-facing observation through the displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, setoid-test row, Real seal boundary, transport, replay, provenance, and local naming rows.

Use the classifier laws and transport theorem, Definition 1820.3 and Definition 1820.4. The comparison remains a finite classifier route over $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$, and hsame can move only those displayed coordinates.

Apply the RegSeqRat route of Definition 1820.20. It places the regular-Cauchy readback and terminal Real seal before any completion-tail consumer can inspect the comparison, so the route reads $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ as sibling rows rather than as a quotient carrier.

Finally use the no-quotient boundary of Definition 1820.22. Since the accepted packet has no coordinate for quotient equality, host equality, selected representatives, selected limits, countable choice, or ambient completeness, the completion-facing consumer receives only the displayed finite sibling route through $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, replay, provenance, and local naming. □

Theorem 1820.22 (CauchyEquivalenceSetoid no-quotient boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet reads equality-facing data through the displayed classifier laws, componentwise transport, completion-tail composition, and finite-chain composition. The consumer receives the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

and cannot replace that route by quotient equality of Cauchy streams, selected representatives, host real equality, selected limits, countable choice, or ambient completion.

Proof. Project the carrier and apply Definition 1820.3. The equality-facing read begins with the two $\mathsf{StreamName}^{\uparrow}$ windows, the two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the dyadic tolerance ledger, and the classifier-law row.

Use Definition 1820.4. Componentwise hsame transport preserves the same displayed rows before any downstream Real-facing seal is available.

Apply Definition 1820.17 and Definition 1820.18. These order every completion-tail and finite-chain read through the classifier surface before the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can inspect the terminal seal.

The carrier has no coordinate for quotient equality, selected representatives, host real equality, selected limits, countable choice, or ambient completion. Therefore a downstream consumer cannot replace the displayed route by any such datum. □

Lean4 ✓ `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidNoQuotient`

The forward dependency into $\mathsf{RealSeparability}^{\uparrow}$ is the density-facing use of the Real seal route. Every $\mathsf{RealSeparability}^{\uparrow}$ consumer of an accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ packet must read the displayed comparison surface

$$S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$$

through the downstream $\mathsf{Real}^{\uparrow}$ seal of Definition 1820.8. Thus the density or completion-facing consumer reaches chapter 1809 only after the two stream windows, two regular rational readbacks, dyadic tolerance ledger, classifier row, Real seal, componentwise transport, replay, provenance, and local naming row are all visible. The route exports no quotient equality of Cauchy streams, selected representative, countable-choice schedule, host real equality, or ambient completeness witness.

The setoid-test row is consumed through the finite-window surface of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. Downstream Cauchy locatedness and selection-free real consumers therefore read the equality-facing classifier through those concrete sibling edges rather than through a detached kernel-only equality object.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a finite classifier-law row, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: The scoped surface binds the displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, Real seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows behind the non-quotient classifier route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyEquivalenceSetoid}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyEquivalenceSetoidUp.CauchyEquivalenceSetoidNameCertSurface`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked classifier-law stability, componentwise transport, replay, provenance, and local naming beyond the displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ NameCert surface.

The downstream $\mathsf{RealClassifier}^{\uparrow}$ consumer of chapter 1620 reads $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ only after the finite comparison surface of chapter 1412 and the regular rational readback rows of chapter 301 have exposed their visible classifier data. Thus the setoid packet supplies the equality-facing sibling lattice route into the Real classifier without becoming a detached quotient equality, a selected limit, or a host real-equality oracle. The Real classifier route of chapter 1620 treats chapter 1820 as a visible sibling dependency, not as a hidden quotient boundary. Its consumer first exposes the regular rational readback discipline of chapter 301, then opens the StreamName windows and DyadicRatCore tolerance rows that make the Cauchy setoid finite before the Real-facing classifier seal is read.

Theorem 1820.23 (Cauchy equivalence setoid Real classifier forward dependency)**.** *Every* $\mathsf{RealClassifier}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must pass through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows before any Real equality-facing seal or classifier boundary is consumed; it supplies no selected representative, quotient equality oracle, countable-choice schedule, ambient Real completeness witness, Lean verification, or bridge check.

Proof. Start from the Cauchy setoid packet and the Real classifier consumer. The packet exposes equality only through finite StreamName windows and the regular rational rows that are read on those windows.

Read the RegSeqRat and DyadicRatCore data next. These rows fix the rational readback and tolerance ledger before the classifier can ask for a Real-facing boundary.

Finally pass the visible setoid route to the Real classifier. Since the classifier receives only the displayed finite route, no quotient oracle, selected representative, choice schedule, ambient completeness datum, Lean proof, or bridge coordinate is available. □

Theorem 1820.24 (Cauchy equivalence setoid Real classifier consumer)**.** *Every* $\mathsf{Real}^{\uparrow}$ *classifier consumer of the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route must pass through the Cauchy equivalence carrier of Definition 1820.1, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the Real classifier dependency route before it can read the terminal Real classifier boundary of chapter 38. The consumer receives the two finite* $\mathsf{StreamName}^{\uparrow}$ *windows, regular rational readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, setoid-test row, Real seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it receives no quotient stream equality, host real equality, selected limit, countable choice, trichotomy, ambient completeness theorem, or detached Real classifier oracle.*

Proof. Project the carrier of Definition 1820.1. Any equality-facing consumer first sees the displayed stream windows, regular rational readbacks, dyadic tolerance ledger, setoid-test row, Real seal boundary, and structural rows.

Use the local Real classifier dependency route. That route sends downstream Real classifier reads through the finite comparison surface and the regular rational readback rows before chapter 38 can be consumed.

The structural rows then transport, replay, source, and name the same path. Since the carrier has no quotient, host-equality, selected-limit, choice, trichotomy, ambient-completeness, or detached-classifier coordinate, every $\mathsf{Real}^{\uparrow}$ classifier consumer remains on the displayed setoid route. □

Theorem 1820.25 (CauchyEquivalenceSetoid RegSeqRat sibling route)**.** *Every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *equality-facing read consumes the finite comparison surface of* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *from chapter 1412, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before exposing the* $\mathsf{Real}^{\uparrow}$ *seal. Completion-facing consumers must read this whole route before any equality read is public, so the setoid packet is horizontally linked to those sibling certificates rather than being a detached equality object, a quotient stream, or a host real-equality relation.*

Proof. Begin with the carrier of Definition 1820.1. Its equality-facing route starts with stream-window rows and regular rational readback rows before the dyadic tolerance, classifier-law, and Real seal rows are visible.

Apply the classifier laws and the NameCert surface. These theorems force reflexivity, symmetry, transitivity, readback exactness, and tolerance exactness to remain finite rows over the same comparison packet.

Use the downstream equality dependency theorem. It identifies the route through the Cauchy comparison surface, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic tolerance ledger, classifier row, and terminal $\mathsf{Real}^{\uparrow}$ seal before any completion-facing consumer can read the result.

The completion-facing boundary is therefore not a separate equality service. A $\mathsf{Real}^{\uparrow}$ or completion consumer first sees $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows, then the classifier laws and terminal seal; only after that finite dependency route has been displayed can the public equality read be consumed.

Therefore the sibling route is substantive and non-quotient: the setoid packet exposes equality only after the RegSeqRat and CauchySequenceEquivalence dependencies have been consumed, and no detached host equality coordinate is present. □

The completion-facing dependency lattice for $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ routes equality reads through the regular-Cauchy classifier before any metric-completion consumer sees them. In particular, chapter 301 supplies the regular rational readback rows that make the setoid comparison visible as finite stream data, and chapter 1506 consumes that comparison only after the metric-completion carrier has exposed its tolerance and completion rows. Thus the setoid packet is a child dependency of the regular-Cauchy and metric-completion route, not a free-standing quotient equality or detached host equivalence relation.

Theorem 1820.26 (RegSeqRat completion classifier route)**.** *Every completion-facing* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *equality read first factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows of chapter 301 and only then through the* $\mathsf{MetricCompletion}^{\uparrow}$ *tolerance and completion rows of chapter 1506. Thus a downstream completion consumer receives the stream-window comparison, regular rational readback, dyadic tolerance ledger, setoid test, and completion-facing handoff in that order; it receives no detached quotient equality, selected representative, host real equality, or independent completion oracle.*

Proof. Start with an accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. Its equality-facing rows expose two stream windows, their regular rational readbacks, the dyadic tolerance ledger, the setoid-test row, and the structural transport, replay, provenance, and naming rows.

The first completion-facing classifier available to a consumer is the $\mathsf{RegSeqRat}^{\uparrow}$ readback route. Only after those regular rational rows and the dyadic tolerance ledger have been displayed can the $\mathsf{MetricCompletion}^{\uparrow}$ tolerance and completion rows read the comparison.

Since the carrier contains no quotient equality coordinate, selected representative, host real equality, or independent completion oracle, every completion-facing equality read follows the stated route. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier is consumed only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback route has exposed the regular rational observation and before the $\mathsf{Real}^{\uparrow}$ seal comparison is read, so the setoid row remains an L10 dependency between chapter 301 and chapter 38 rather than an isolated quotient or host-equality surface. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier route depends substantively on the regular rational readback of chapter 301 and the terminal seal of chapter 38. The completion and Real classifier child routes already display how those dependencies are consumed before any downstream equality-facing reader sees the setoid packet, so the dependency lattice remains a routed finite comparison surface rather than quotient stream equality, host real equality, a selected limit, or ambient completeness. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ packet is consumed by the regular-rational to real-name seal route: a $\mathsf{RegSeqRat}^{\uparrow}$ readback supplies the displayed Cauchy representative boundary of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ classifier of chapter 38 can then read equality only through that setoid boundary. This route keeps the setoid as a finite dependency row for the Real seal rather than as host quotient equality, selected representatives, countable choice, or ambient real completeness. Every $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier consumer that serves $\mathsf{MetricCompletion}^{\uparrow}$ or separated-completion readers first reads the regular-rational comparison route of chapter 301 and the terminal Real seal route of chapter 38. The setoid classifier is therefore consumed only through displayed regular-rational windows, finite $\mathsf{StreamName}^{\uparrow}$ observations, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seal rows before completion-facing data is public. Quotient stream equality and selected representatives are excluded from the route, since

no completion consumer receives a classifier except through those displayed rows and their hsame, Cont, Pkg, and NameCert handoff.

Theorem 1820.27 (CauchyEquivalenceSetoid RegSeqRat–Real dependency anchor)**.** *Every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier read consumed by chapter 301 or chapter 38 routes through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and dyadic tolerance ledgers before any terminal Real-facing equality boundary is exposed. Thus the setoid surface supplies a displayed dependency anchor for regular rational readback and Real sealing, not a quotient-stream object, selected representative, host equality relation, or ambient completeness principle.*

The $\mathsf{RegSeqRat}^{\uparrow}$ consumer edge of chapter 301 reads $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ only as a displayed finite classifier. The route is forced through the setoid carrier, the open-tail classifier, and the regular-Cauchy uniqueness boundary before any regular rational readback is exported.

Theorem 1820.28 (RegSeqRat forward link from Cauchy equivalence setoid)**.** *Every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier read used by a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer factors through the finite classifier of Definition 1820.1, the open-tail route of Definition 1820.32, and the regular-Cauchy uniqueness route of Definition 1820.55. Thus a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer receives stream windows, regular rational readback rows, the dyadic tolerance ledger, the setoid-test row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming as one finite route, not as quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted carrier $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ from Definition 1820.1. The only visible comparison surface begins with the two $\mathsf{StreamName}^{\uparrow}$ windows and their two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows.

Apply the open-tail classifier route. It places the dyadic tolerance ledger and setoid-test row after those regular rational readbacks and before the terminal $\mathsf{Real}^{\uparrow}$ seal, so the consumer cannot replace the route by a detached equality of streams.

Finally apply the regular-Cauchy uniqueness route. That route consumes the same displayed classifier rows before exporting any uniqueness-facing regular Cauchy readback, and therefore gives the forward link into chapter 301 without selected representatives, host equality, or an ambient completion principle. □

Theorem 1820.29 (Cauchy equivalence setoid RegSeqRat classifier gate)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must pass through the displayed regular-Cauchy classifier row before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The consumer may then use the forward link to chapter 301 and the seal boundary of chapter 38, but it receives no quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient completion principle outside the displayed classifier route.*

Proof. Project the setoid classifier row from the carrier of Definition 1820.1. This is the only public row that compares two regular Cauchy presentations before a Real-facing read is allowed.

Next apply the RegSeqRat forward link, Definition 1820.28. It sends the consumer through the stream windows, regular rational readback rows, dyadic tolerance ledger, and setoid-test row before any regular-rational data are exported.

Finally read the Real-seal nonescape boundary along the same route. Since the terminal seal is downstream of the displayed classifier and the RegSeqRat handoff, a consumer cannot bypass those rows by using a quotient stream, selected representative, host equality, choice selector, or ambient completion witness. □

The forward equality route for $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ is a finite dependency path through chapter 301, chapter 291, and chapter 38. It keeps the sibling edge outside the hub and records that equality-facing reads are downstream of regular rational and dyadic tolerance data.

Theorem 1820.30 (Cauchy equivalence setoid RegSeqRat–Real forward route)**.** *Every equality-facing read exported by* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *first consumes finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the setoid route exposes regular rational observations and dyadic comparison rows before any Real-facing equality handoff, and it exports no selected representatives, quotient stream object, host real equality, or ambient completeness principle.*

Proof. Project the accepted Cauchy-equivalence carrier from Definition 1820.1. Its comparison surface begins with finite stream windows and regular rational readback rows.

Apply the dyadic tolerance route. The tolerance ledger is read before the Real seal, so a Real-facing equality handoff cannot bypass the dyadic comparison cells.

Finally apply the existing Real classifier dependency route for $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$. It reaches $\mathsf{Real}^{\uparrow}$ only after the regular rational and dyadic rows have been displayed, which gives the stated forward route without selected representatives, quotient stream objects, host real equality, or ambient completeness. □

Theorem 1820.31 (Cauchy equivalence setoid RegSeqRat sibling forward route)**.** *Every* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier read consumed by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing route factors through the displayed sibling rows of* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any Real seal is read. Concretely, the consumer first follows the RegSeqRat sibling route of Definition 1820.25, then the forward link of Definition 1820.28, and finally the RegSeqRat–Real forward route of Definition 1820.30; no quotient stream object, selected representative, host real equality, or detached completion oracle is available outside that finite route.*

Proof. Project the accepted carrier of Definition 1820.1. Its comparison surface already begins with finite stream windows and regular rational readback rows rather than a quotient object.

Apply the sibling route of Definition 1820.25. This forces every equality-facing read to expose the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger before the terminal seal can be consumed.

Now use the forward link of Definition 1820.28. The link identifies the same displayed rows as the route read by RegSeqRat-facing consumers, so a completion-facing consumer cannot replace the dependency by selected representatives or host equality.

Finally apply Definition 1820.30. The Real-facing handoff occurs only after the regular rational and dyadic rows have been displayed, which proves the sibling forward route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any Real seal is read. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ packet is also read in reverse from the regular-Cauchy and Real completion side: consumers in chapter 301 and chapter 38 cite its equality-facing classifier only after the displayed regular rational readback, dyadic tolerance ledger, and terminal Real seal have all been exposed. This records the horizontal dependency edge without turning the setoid into quotient stream equality, selected representatives, host real equality, or ambient completeness. When a $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ comparison is restricted to bounded regular Cauchy packets, its equality route consumes the bounded tail-modulus and regular-readback surface of chapter 1697. The comparison first reads the finite threshold row of Definition 1697.6, then keeps the two compared names on the same displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and $\mathsf{NameCert}$ route before any setoid classifier may be applied.

Theorem 1820.32 (Open-tail Cauchy equivalence classifier)**.** *Every open-tail classifier read from an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet is the exact sibling route consumed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *completion readback and the terminal* $\mathsf{Real}^{\uparrow}$ *classifier boundary. It first exposes the two* $\mathsf{StreamName}^{\uparrow}$ *finite windows, then the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and setoid-test row, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. Completion-facing consumers may read that route through chapter 1506, but they receive no quotient stream equality, selected representative, host real equality, or detached completion oracle.*

Proof. Project the carrier of Definition 1820.1. An open-tail read begins with the two visible stream windows and their regular rational readback rows.

Use the RegSeqRat completion classifier route of Definition 1820.26. That route places the regular rational readbacks before the metric-completion tolerance and completion rows, so the comparison is already a displayed finite classifier before any completion consumer can inspect it.

Finally use the Real classifier consumer of Definition 1820.24. The terminal Real boundary is reached only after the dyadic tolerance ledger and setoid-test row are visible, and the packet has no coordinate for quotient equality or selected representatives. □

Theorem 1820.33 (Cauchy equivalence setoid open-tail forward link)**.** *Every accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet that is consumed by an open-tail classifier route sends its setoid comparison forward through the displayed route of Definition 1820.32. The consumer first reads the two* $\mathsf{StreamName}^{\uparrow}$ *finite windows, then the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback rows, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and local setoid-test row, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *completion seal. Thus the open-tail route is a child dependency of the setoid packet rather than a detached completion oracle.*

Proof. Project the accepted carrier from Definition 1820.1. Its visible comparison data are the two stream names, their regular-rational readbacks, the dyadic tolerance ledger, the local setoid-test row, transport, replay, provenance, and local naming.

Apply the open-tail classifier route of Definition 1820.32. That route already orders the finite windows before the $\mathsf{RegSeqRat}^{\uparrow}$ rows and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the terminal $\mathsf{Real}^{\uparrow}$ seal.

Consequently a consumer that asks for the open-tail classifier consumes the same setoid comparison surface. Since the carrier has no coordinate for quotient stream equality, selected representatives, host real equality, or a detached completion point, the forward link is exactly the displayed child route. □

Theorem 1820.34 (Open-tail Cauchy equivalence transitivity route)**.** *Every open-tail classifier route from an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet composes Cauchy-equivalence transitivity with the* $\mathsf{Real}^{\uparrow}$ *completion readback as a finite NameCert route. If two displayed comparisons share the middle regular-Cauchy read, then the consumer first reads the two* $\mathsf{StreamName}^{\uparrow}$ *window pairs, then the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and transitivity row, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *completion seal of chapter 38. The route is available to completion consumers through chapter 1506, but it exports no quotient stream equality, selected representative, host real equality, or detached completion point.*

Proof. Project the accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. The two open-tail comparisons expose their finite stream windows and regular rational readbacks before the setoid classifier can be consumed.

Use the open-tail classifier route of Definition 1820.32. Each comparison is already constrained to the same StreamName, RegSeqRat, DyadicRatCore, and RealUp route, so the shared middle read is a displayed row rather than a quotient representative.

Now compose the two displayed tolerance ledgers by the transitivity row carried by the setoid packet. The resulting comparison remains in the same BHist surface and is transported by the packet hsame row and replayed by its Cont row.

Finally pass the composed classifier to the Real completion readback. Since the terminal seal is downstream from the displayed windows, readbacks, dyadic ledger, transitivity row, transport, replay, provenance, and local naming, no host equality or detached completion point is introduced. □

Theorem 1820.35 (Cauchy equivalence setoid tail transitivity choice-free)**.** *Composing two open-tail comparisons in an accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *packet with a shared middle read uses only the displayed finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks,* DyadicRatCore$^{\uparrow}$ *tolerance transitivity,* Real$^{\uparrow}$ *completion readback, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows. The composite exports no selected representative, quotient stream equality, host real equality, detached completion point, countable-choice schedule, or ambient completeness row.*

Proof. Start with the open-tail route of Definition 1820.34. The two comparisons are already displayed as finite StreamName windows and RegSeqRat readbacks with one shared middle read.

Next read the DyadicRatCore tolerance rows attached to those two comparisons. Their transitivity row composes the displayed tail bounds inside the same BHist packet, so no representative of a quotient stream is selected.

Now transport the composite through the packet hsame row and replay it through the displayed Cont row. The Pkg provenance and local NameCert rows keep the composite attached to the same CauchyEquivalenceSetoid surface.

Finally send the composed comparison to the Real completion readback named in the open-tail route. Since the terminal Real seal consumes only the displayed windows, readbacks, dyadic transitivity, transport, replay, provenance, and local naming rows, it cannot add quotient equality, host real equality, detached completion points, choice data, or ambient completeness. □

Theorem 1820.36 (CauchyEquivalenceSetoid tail stability)**.** *Let* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ *be an accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *carrier. Replacing both compared regular rational windows by later displayed tails preserves the local* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier whenever the later windows are read through the same* StreamName$^{\uparrow}$ *schedules,* RegSeqRat$^{\uparrow}$ *readbacks, dyadic tolerance ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

The preserved classifier is still the finite row T *over the displayed tails. It exports no quotient stream equality, selected representatives, host real equality, countable choice, hidden cofinal tail, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 1820.1. The classifier can inspect only the two StreamName$^{\uparrow}$ finite windows, the two RegSeqRat$^{\uparrow}$ readbacks, the DyadicRatCore$^{\uparrow}$ tolerance ledger, the setoid-test row, the Real seal boundary, and the structural rows.

Pass to later displayed tails by finite tail projection on S_0 and S_1. The corresponding readbacks remain rows of R_0 and R_1, and every admitted comparison still uses the same dyadic tolerance ledger D.

Transport the projected windows componentwise through H and replay the read order through C. The rows P and N keep the tail comparison attached to the same package provenance and local NameCert surface.

Thus the setoid-test row T is preserved over the later displayed tails. Since no carrier row names quotient stream equality, selected representatives, host real equality, countable choice, hidden

cofinal tails, or ambient completeness, tail stability is exactly the displayed finite classifier route. □

Theorem 1820.37 (CauchyEquivalenceSetoid uniform completion consumer)**.** *Every* $\mathsf{UniformSpaceCompletion}^{\uparrow}$*-facing read from an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet factors through the displayed regular-Cauchy classifier, the completion dependency lattice, the* $\mathsf{RegSeqRat}^{\uparrow}$ *completion classifier route, and the* $\mathsf{Real}^{\uparrow}$ *classifier consumer before it can be consumed by chapter 3560. Thus the consumer receives stream-window comparison, regular rational readback, dyadic tolerance, setoid-test, completion handoff, Real classifier boundary, transport, replay, provenance, and local naming rows; it cannot replace those classifier rows by host quotient equality or by a selected representative.*

Proof. Start with an accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. The completion dependency lattice forces the equality read through the regular-Cauchy classifier before any completion-facing consumer can see it.

Next use the $\mathsf{RegSeqRat}^{\uparrow}$ completion classifier route. This route exposes the regular rational readback and dyadic tolerance ledger before the comparison reaches a completion handoff, so the uniform-completion consumer cannot receive a detached quotient equality.

Finally use the $\mathsf{Real}^{\uparrow}$ classifier consumer. It sends the same finite comparison through the Real classifier boundary and then repacks the result through the packet projection, transport, replay, provenance, and local NameCert rows. Since none of these rows contains a selected representative or host quotient coordinate, the $\mathsf{UniformSpaceCompletion}^{\uparrow}$ read remains on the displayed classifier route. □

Theorem 1820.38 (Cauchy equivalence setoid Real separability route)**.** *Let* $Q = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$ *be an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *carrier. Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer that exports a* $\mathsf{Real}^{\uparrow}$ *read from* Q *factors through the displayed Cauchy equivalence row, the regular rational readback rows* R_0, R_1*, the terminal Real seal row* E*, and the finite density route of chapter 1809. In particular, the consumer may inspect only*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D, T \longmapsto E \longmapsto H, C, P, N$$

before it reaches the Real-separability density read.

Proof. Start with an accepted consumer of Q that exports a Real-separability read. By Definition 1820.1, the only equality-facing rows available to that consumer are the two StreamName windows, the two RegSeqRat readback rows, the dyadic tolerance ledger, the setoid-test row, the Real seal boundary, and the structural rows.

Apply Definition 1820.3 and Definition 1820.5. These force the Cauchy equivalence row to be read as finite classifier data over S_0, S_1, R_0, R_1, D, T, with no quotient-equality or host-equality coordinate.

Next use the sibling route through chapter 301, chapter 38, and chapter 1809. The regular rational readback is inspected before E, and the finite density route of $\mathsf{RealSeparability}^{\uparrow}$ is available only after that Real seal has been displayed.

This is the forward dependency that moves the density surface of chapter 1809 through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier before $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are consumed.

Finally, Definition 1820.4 keeps the route componentwise under H, while C, P, N replay, source, and name the same rows. Thus every Real-separability consumer factors through the displayed Cauchy equivalence, RegSeqRat readback, Real seal, and density route. □

Theorem 1820.39 (Cauchy equivalence setoid Real-separability forward link)**.** *The* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier route is the forward source consumed by the* RealSeparability$^{\uparrow}$ *RegSeqRat dependency lattice. Any consumer using chapter 1809 together with the regular rational surface of chapter 301 must first read the displayed Cauchy-equivalence classifier rows, the two RegSeqRat readback rows, the dyadic tolerance ledger, and the Real seal boundary supplied by the accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *packet.*

Consequently the RealSeparability density surface receives a finite Cauchy/RegSeqRat comparison route rather than a quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient completeness row.

Proof. Begin with the accepted CauchyEquivalenceSetoid$^{\uparrow}$ classifier route. Its visible source rows are the two StreamName windows, the two RegSeqRat readback rows, the dyadic tolerance ledger, the setoid-test row, the Real seal boundary, and the hsame, Cont, Pkg, and local NameCert support.

Apply Definition 1820.38. That route places the RealSeparability consumer after the Cauchy-equivalence classifier, the RegSeqRat readback rows, and the Real seal boundary, so the density read cannot bypass the displayed setoid comparison.

Now read the RegSeqRat dependency through chapter 301. The same regular rational rows that feed the Cauchy-equivalence classifier are the rows consumed by the RealSeparability lattice, and chapter 1809 receives them as finite classifier-facing data.

The route is consumed by chapter 1809 because the Cauchy-equivalence setoid classifier rows provide the dense-read equality surface used by RealSeparability$^{\uparrow}$.

The excluded objects have no source row in this route. Quotient stream equality, selected representatives, host real equality, countable-choice selectors, and ambient completeness data are not among the displayed classifier, readback, tolerance, seal, transport, continuation, package, or naming rows, so they cannot serve as a second forward dependency. □

The backward dependency from RealSeparability$^{\uparrow}$ to CauchyEquivalenceSetoid$^{\uparrow}$ is the classifier-facing half of the finite density route. Before a separability packet exports a countable dense display as in chapter 1809, it reads the Cauchy-equivalence classifier to keep equivalent regular names on the same displayed density window. The dependency is therefore a finite comparison and density handoff, not quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness.

Proposition 1820.40 (RealSeparability backward dependency)**.** *Every* RealSeparability$^{\uparrow}$ *consumer that exports a countable dense display through chapter 1809 reads the accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier before the dense display is exposed. Consequently the exported density row is stable under the displayed Cauchy-equivalence comparison, and the route exports no quotient stream, selected representative, countable-choice selector, host real equality, or ambient completeness row.*

Proposition 1820.41 (Cauchy equivalence setoid dense-readback consumer route)**.** *The* CauchyEquivalenceSetoid$^{\uparrow}$ *packet supplies the classifier boundary consumed by the dense-readback routes of chapter 1809. In particular, every dense* RealSeparability$^{\uparrow}$ *read that identifies a rational or regular-rational approximation through Cauchy-tail agreement must keep the equivalence-setoid row upstream of the* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*, and* Real$^{\uparrow}$ *readback rows, so the dense consumer receives a finite classifier boundary rather than quotient stream equality, host real equality, or a selected completion representative.*

Proof. Project the CauchyEquivalenceSetoid$^{\uparrow}$ carrier through the existing real-classifier and uniform-completion routes. Those routes expose the Cauchy-tail classifier before any regular rational or Real-facing seal is consumed.

The dense-readback packet of chapter 1809 uses that same boundary when it names approximating rationals. Since the setoid row is already upstream of the StreamName, RegSeqRat, DyadicRatCore, and Real rows, the dense consumer cannot replace it by quotient stream equality or by a host-selected representative. □

Theorem 1820.42 (CauchyEquivalenceSetoid RealSeparability forward scope)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-modulus consumer that reads through an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must first pass through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *schedule of chapter 5728, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before exporting density through chapter 1809. The route exports no quotient equality, selected representative, countable choice, or ambient completeness principle.*

Proof. Project the local CauchyEquivalenceSetoid carrier of Definition 1820.1. Its visible rows begin with two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the setoid-test row, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Next route the dense-modulus request through chapter 5728. The bridge supplies the explicit Cauchy-modulus schedule before the RealSeparability consumer may export density.

Finally read the separability endpoint of chapter 1809. Because the carrier and bridge expose only the displayed finite rows, the route cannot appeal to quotient equality, selected representatives, countable choice, or ambient completeness. □

Theorem 1820.43 (CauchyEquivalenceSetoid RealSeparability route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-readback consumer of* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *first reads the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks of chapter 301, dyadic tolerance data, the displayed setoid classifier, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming rows. Thus the downstream route to chapter 1809 is a finite dependency row through the Cauchy-equivalence carrier, not quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Theorem 1820.44 (Cauchy equivalence setoid Real-separability sibling route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-route consumer that reads an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must first traverse the finite sibling route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RealSeparability}^{\uparrow}.$$

Concretely, the consumer reads the two $\mathsf{StreamName}^{\uparrow}$ *windows, the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the setoid-test row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and then the componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows before density is public through chapter 1809. The route exports no quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient completeness row.*

Proof. Start with the carrier and surface theorems Definition 1820.1 and Definition 1820.5. They list exactly the two stream windows, two regular-rational readbacks, dyadic ledger, setoid-test row, Real seal, transport, replay, provenance, and local naming rows.

Apply the classifier laws and transport discipline, Definition 1820.3 and Definition 1820.4. These keep the setoid-test row finite over the displayed StreamName, RegSeqRat, and DyadicRatCore rows before the Real seal is consumed.

Now use the existing Real-separability routes Definition 1820.38, Definition 1820.39, and Definition 1820.14. They place the dense-route consumer of chapter 1809 after the same classifier and Real-seal surface.

Thus the public density read is available only after the sibling route has displayed every row in order. Since none of the cited carrier rows contains quotient equality, selected representatives, host equality, choice, or ambient completeness, the Real-separability consumer receives only the finite sibling edge. □

Theorem 1820.45 (Cauchy equivalence setoid Real-separability forward lattice repair)**.** *Every downstream* $\mathsf{RealSeparability}^{\uparrow}$ *dense-route consumer that cites chapter 1809 from the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *side must first read the displayed sibling rows*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CauchyEquivalenceSetoid}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Thus the RealSeparability citation is a horizontal sibling edge after the finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid test row, and RealUp seal have all been displayed.

Proof. Apply Definition 1820.44. It already forces the dense-route consumer through the StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and RealUp rows before any $\mathsf{RealSeparability}^{\uparrow}$ read is public.

The explicit endpoint is the sibling chapter chapter 1809. Therefore the citation to RealSeparability is not a self-reference inside the CauchyEquivalenceSetoid packet, but a finite lattice edge whose source rows have already been displayed. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier is the forward lattice edge used when $\mathsf{RealSeparability}^{\uparrow}$ reads Cauchy-tail agreement as dense-readback data. It is placed before the stream, regular-sequence, dyadic, and real readback rows are treated as a dense route.

Theorem 1820.46 (Cauchy equivalence setoid Real-separability forward lattice)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense-readback consumer that uses Cauchy-tail agreement must read the displayed* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier before the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *readback rows can be treated as a dense route. The consumer receives a finite classifier edge from the accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet to chapter 1809; it receives no quotient stream equality, selected representative, countable-choice selector, host real equality, or ambient completeness row.*

Proof. Start with the carrier surface of Definition 1820.1. Its displayed rows expose the two stream windows, two regular-rational readbacks, dyadic tolerance ledger, setoid-test row, terminal Real seal, componentwise transport, continuation replay, provenance, and local naming data.

The dense-readback consumer may use Cauchy-tail agreement only after the setoid-test row has been inspected. Therefore the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier is the row that turns the surrounding $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readbacks into a route toward $\mathsf{RealSeparability}^{\uparrow}$.

Finally apply the existing Real-separability sibling and route statements, Definition 1820.44 and Definition 1820.38. They place chapter 1809 after the same finite classifier boundary, so the forward lattice edge adds no quotient equality, selected representative, countable-choice selector, host real equality, or ambient completeness claim. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ family also supplies the forward classifier route consumed by $\mathsf{RealSeparability}^{\uparrow}$ dense-read and setoid-modulus schedules. The route in Definition 1820.44 keeps the two $\mathsf{StreamName}^{\uparrow}$ windows, the two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the Cauchy-equivalence classifier, and the terminal $\mathsf{Real}^{\uparrow}$ seal visible before the dense consumer reaches chapter 1809. Thus the separability lattice receives a finite classifier boundary

rather than quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness.

Theorem 1820.47 (Cauchy equivalence setoid Real-separability reverse route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *dense read that uses regular Cauchy representatives must preserve the same* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier before any dense-seal comparison is exported. Thus a consumer returning from chapter 1809 to the Cauchy-equivalence surface must keep the two StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, setoid-test row, terminal Real seal, transport, replay, provenance, and local naming rows visible.*

Proof. Start with the accepted Cauchy-equivalence carrier and local classifier route. The route in Definition 1820.38 sends a Real-separability consumer through the displayed StreamName, RegSeqRat, DyadicRatCore, setoid-test, and Real-seal rows before density is public.

Now read the dense-seal comparison back through chapter 1809. A Real-separability dense read supplies a countable dense surface only after its regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The reverse route therefore preserves the original classifier row. If the consumer dropped it, the returned dense-seal comparison would be a quotient stream equality, selected representative, countable-choice selector, host real equality, or ambient completeness claim, none of which is present in the accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. □

Theorem 1820.48 (Cauchy equivalence setoid Real-separability backlink)**.** *The* $\mathsf{RealSeparability}^{\uparrow}$ *dense surface consumes the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier through the regular Cauchy and real-name routes. Concretely, any citation from chapter 1809 back to the setoid packet must first display the regular rational readback of chapter 301, the Cauchy equivalence classifier row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38; only then may the dense surface read the setoid route as a horizontal dependency.*

Proof. Start with the accepted $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. Its classifier row is not a free equality principle; it is read through StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, the setoid test row, and the terminal Real seal.

The RealSeparability surface can therefore consume the setoid packet only after those same rows have been displayed. The regular Cauchy route supplies the readback discipline, the RealUp route supplies the name seal, and the dense surface records the resulting edge without selecting representatives or quotienting completed streams. □

Theorem 1820.49 (CauchyEquivalenceSetoid RegSeqRat Real classifier bridge)**.** *Every equality-facing comparison accepted by* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *displays two* $\mathsf{StreamName}^{\uparrow}$ *windows, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier before any* $\mathsf{Real}^{\uparrow}$ *seal or separated-completion consumer may read equality. Thus the bridge from regular rational readback to Real-facing and completion-facing equality is a finite classifier route, not quotient stream equality, selected representatives, host real equality, countable choice, or ambient Real completeness.*

Proof. Start with the carrier and local surface of Definition 1820.1 and Definition 1820.5. They expose exactly the two stream windows, two regular-rational readback rows, dyadic tolerance ledger, setoid-test row, Real seal boundary, transport, replay, provenance, and local naming.

Apply the RegSeqRat–Real route of Definition 1820.30. It orders finite $\mathsf{StreamName}^{\uparrow}$ observations before $\mathsf{RegSeqRat}^{\uparrow}$ readback and places the dyadic comparison ledger before the terminal $\mathsf{Real}^{\uparrow}$ seal.

Use the separated-completion route of Definition 1820.56. A completion-facing reader receives equality only after the same displayed classifier rows have been consumed.

Since the carrier contains no quotient stream, selected representative, host equality, choice, or ambient-completeness coordinate, the composed bridge exports only the finite classifier route. □

Theorem 1820.50 (CauchyEquivalenceSetoid concrete RegSeqRat Real route)**.** *Every completion-facing consumer that uses the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier as a RegSeqRat-to-Real equality route must first read the regular rational comparison surface of chapter 301 and then the terminal seal boundary of chapter 38. The consumed packet is the displayed route through* $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$*, not quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Start with the carrier and NameCert surface of Definition 1820.1 and Definition 1820.5. They display the two StreamName windows, two RegSeqRat readback rows, dyadic tolerance ledger, setoid-test row, Real seal boundary, and structural rows before any equality-facing consumer may inspect the result.

Apply the RegSeqRat Real classifier bridge of Definition 1820.49. It orders the regular rational readback through chapter 301 before the Real-facing seal route of chapter 38.

Finally use the Real seal forward route of Definition 1820.8. The downstream consumer receives the same finite classifier route and no excluded quotient, selected representative, host equality, choice, or ambient-completeness object. □

The Cauchy-space route for $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ is a child dependency of the finite regular-rational comparison surface. A consumer in chapter 1847 reads the setoid only after the regular rational windows of chapter 301 have exposed the comparison rows, and before the completion-facing seals of chapter 1506 or chapter 38 may consume them.

Theorem 1820.51 (CauchySpace route for Cauchy-equivalence setoids)**.** *Every* $\mathsf{CauchySpace}^{\uparrow}$ *consumer that uses a* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison must read it through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison route before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is available. Thus the setoid supplies a routed comparison row for Cauchy-space and metric-completion consumers, not a detached quotient equality or a host representative selector.*

Proof. Read the setoid comparison as a regular-rational route. The sibling surface of chapter 301 supplies the finite windows through which two Cauchy names become comparable.

The Cauchy-space consumer of chapter 1847 can then import that comparison as one row in its finite filter-uniform route. It has not yet reached a metric-completion seal or a terminal real seal.

Only downstream of that route may chapter 1506 and chapter 38 consume the comparison. Because the route stores no quotient equality, selected representative, or host equality test, the setoid remains a finite dependency edge in the L10 completion lattice. □

The $\mathsf{CauchySpace}^{\uparrow}$ consumers of chapter 1847 read $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ from chapter 1820 only after the $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, setoid-test row, terminal $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, and NameCert rows are visible. Thus the setoid packet reaches Cauchy-space as a routed finite comparison surface, not as a detached quotient representative or host equality selector.

Theorem 1820.52 (Completion and real dependency readback)**.** *Every* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *read consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchySpace}^{\uparrow}$*,* $\mathsf{UniformSpaceCompletion}^{\uparrow}$*, or* $\mathsf{RealSeparability}^{\uparrow}$

is available only through the displayed classifier and readback rows of the accepted packet. More precisely, a consumer may inspect the two StreamName$^{\uparrow}$ *windows, the two* RegSeqRat$^{\uparrow}$ *readback rows of chapter 301, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the setoid-test classifier, the terminal* Real$^{\uparrow}$ *seal of chapter 38, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The completion-facing consumers of chapter 3560 and the density-facing consumers of chapter 1809 receive exactly that route, so the setoid remains a child dependency of the Real-completion lattice rather than a top-level quotient stream, selected representative, countable-choice selector, host real equality, or ambient completeness principle.*

Proof. Start with an accepted CauchyEquivalenceSetoid$^{\uparrow}$ carrier. Its public coordinates are the two stream windows, the two regular-rational readback rows, the dyadic tolerance ledger, the setoid-test classifier, the Real seal boundary, transport, replay, provenance, and local naming.

The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers read these coordinates in the order already exposed by the classifier route. The CauchySpace$^{\uparrow}$ consumer of chapter 1847 receives the same comparison row only after the regular-rational readback and Real seal have been displayed.

For completion and separability consumers, apply the uniform-completion and real-separability routes. They consume the displayed comparison after the same readback and classifier rows, and neither route adds a quotient equality, selected representative, countable-choice selector, host real equality, or ambient completeness coordinate. Hence every listed sibling reads the setoid only through the displayed dependency route. □

The CauchyEquivalenceSetoid$^{\uparrow}$ equality surface is consumed by chapter 1569 and chapter 1506 only through the completion-filter route that first exposes StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, the DyadicRatCore$^{\uparrow}$ tolerance ledger, the terminal Real$^{\uparrow}$ seal, and then the displayed BHist, hsame, Cont, Pkg, and NameCert rows. The route supplies equality inspection for completion consumers without exporting quotient representatives, selected limits, ambient metric completeness, countable choice, or host real equality. The CauchyEquivalenceSetoid$^{\uparrow}$ completion reads consume the finite Cauchy-filter route of chapter 333 before the regular-sequence and real-name consumers identify tail-equivalent regular Cauchy names. In this route, the filter base fixes the displayed tail packets and Cauchy-modulus rows, chapter 301 reads them as regular rational sequence evidence, and chapter 38 seals the same data as a terminal real-name consumer without adding quotient filters, selected limits, or ambient completeness.

Theorem 1820.53 (CauchyEquivalenceSetoid sibling route)**.** *Every setoid-classifier read exported by an accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *packet consumes the finite sibling route through chapter 126, chapter 301, chapter 291, and chapter 38 before any* RealEquality$^{\uparrow}$ *or* MetricCompletion$^{\uparrow}$ *consumer may inspect it. After the* RealEquality$^{\uparrow}$ *boundary is displayed,* MetricCompletion$^{\uparrow}$ *and* SeparatedCompletion$^{\uparrow}$ *consumers routed through chapter 1612 may consume the classifier only after the* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, *and* RealEquality$^{\uparrow}$ *rows are displayed. The route reads finite stream windows, regular rational readbacks, the dyadic tolerance ledger, and the terminal real-name seal as displayed rows; it does not export host real equality, quotient setoid classes, selected representatives, countable choice, or ambient completion.*

Proof. Project the accepted setoid packet to its classifier route. The sequence-facing entry is a displayed StreamName$^{\uparrow}$ row, so a consumer first reads finite windows rather than a completed stream or selected representative.

The regular rational readback is then consumed through the RegSeqRat$^{\uparrow}$ sibling route. Its tolerance comparisons are read only through the DyadicRatCore$^{\uparrow}$ ledger, which supplies the finite dyadic budget used by the classifier.

Only after those rows are fixed can the terminal $\mathsf{Real}^{\uparrow}$ seal be inspected. The $\mathsf{RealEquality}^{\uparrow}$ consumer of chapter 1636 reads the setoid classifier only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows have been fixed.

The completion consumers are downstream of that equality boundary. $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{SeparatedCompletion}^{\uparrow}$ may consume the classifier only after the RealEquality row is displayed, so they receive a routed setoid classifier over displayed rows, not host real equality or a quotient completion object. □

Theorem 1820.54 (CauchyEquivalenceSetoid quotient-boundary completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet must read quotient-boundary refusal through chapter 1708 and completion readiness through chapter 3452 before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The setoid route therefore reaches Real readback only after the quotient boundary, regular Cauchy completion criterion, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed; it does not export quotient classes, selected representatives, host real equality, countable choice, or ambient completion.*

Proof. Project the accepted setoid packet to its completion-facing classifier rows. The quotient-boundary chapter supplies the refusal surface, so the consumer first learns that the route is not a quotient-class construction.

Next read the regular Cauchy completion criterion. It exposes the finite completion readiness rows before any Real seal can be inspected, and it keeps the route on displayed regular-Cauchy data rather than a selected representative.

The structural rows then transport, replay, source, and name the same route. Thus the terminal Real readback is downstream of quotient-boundary refusal and completion readiness, not a replacement for them. □

The $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ seal route of chapter 3390 may consume the finite classifier supplied by $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ before it reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. This makes the setoid packet a horizontal dependency for regular Cauchy uniqueness rather than an isolated equality-facing classifier.

Theorem 1820.55 (RegularCauchyUniqueness CauchyEquivalenceSetoid route)**.** *Every* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *seal read that consumes a* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet first traverses the finite classifier route of Definition 1820.1, the classifier laws of Definition 1820.3, and the NameCert surface of Definition 1820.5. Thus the uniqueness consumer reads* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the setoid-test row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, transport, replay, provenance, and local naming before it treats the regular Cauchy seal as independent of representative choice.*

Proof. Project the local $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier. Its visible route is the pair of stream windows, pair of regular readback rows, dyadic tolerance ledger, setoid-test row, Real seal boundary, and structural rows.

Apply the classifier laws and NameCert surface. Reflexivity, symmetry, transitivity, tolerance exactness, seal nonescape, transport, replay, provenance, and local naming are all supplied by the same finite rows, so the uniqueness consumer receives classifier data rather than quotient stream equality.

Finally compare with chapter 3390. The $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ carrier already reads paired $\mathsf{RegSeqRat}^{\uparrow}$ names, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a comparison row, and a terminal $\mathsf{Real}^{\uparrow}$ seal. The setoid packet supplies that comparison row through the displayed finite

classifier route, with no selected representative, host real equality, countable choice, or ambient completeness. □

Theorem 1820.56 (CauchyEquivalenceSetoid separated-completion forward route)**.** *Every* $\mathsf{SeparatedCompletion}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *packet consumes the setoid boundary through chapter 1612 after the displayed* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been exposed. The route is a finite comparison handoff into separated completion, not quotient completion equality, selected representatives, host real equality, countable choice, or ambient Real completeness.*

Proof. Start with the CauchyEquivalenceSetoid carrier of Definition 1820.1. It exposes two finite StreamName windows, two RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the setoid-test row, and the Real seal boundary.

Use the downstream equality dependency and completion dependency routes already displayed in this chapter. They place completion-facing consumers behind the same finite comparison data rather than behind a quotient of Cauchy names.

Now read the separated-completion carrier. Its Cauchy-equivalence source row consumes the displayed CauchySequenceEquivalence, RegSeqRat, StreamName, DyadicRatCore, and Real rows before separated comparison is available. Hence the forward route into $\mathsf{SeparatedCompletion}^{\uparrow}$ is exactly the finite setoid boundary and not a quotient-completion equality. □

Theorem 1820.57 (SeparatedCompletion displayed setoid dependency)**.** *Every* $\mathsf{SeparatedCompletion}^{\uparrow}$ *read that cites the Cauchy-equivalence component of its carrier reads it only as displayed classifier evidence from the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *surface of Definition 1820.1. Thus the route through chapter 1612 consumes the finite setoid rows* $S_0, S_1, R_0, R_1, D, T, E, H, C, P, N$ *before separated-completion equality is exported, and it does not replace those rows by quotient classes, selected representatives, host real equality, or ambient completeness.*

Proof. Project the separated-completion carrier through its Cauchy-equivalence source row. The row is a consumer of displayed classifier evidence, so it must read the local $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ carrier before the separated-completion export is available.

The carrier of Definition 1820.1 lists exactly the two stream windows, two regular-rational readbacks, dyadic tolerance ledger, setoid-test row, real seal boundary, componentwise transport, replay, provenance, and local naming rows. These are the rows read by the separated-completion route.

Since no quotient class, selected representative, host real equality, or ambient-completeness witness appears in that carrier, the reverse dependency records only the displayed classifier route into chapter 1612. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier is consumed by separated-completion and metric-completion readers only after the regular rational stream rows and terminal real-name rows have been displayed. The route reads chapter 301 for regular-Cauchy comparison data, chapter 38 for the Real seal that preserves the same comparison window, and chapter 1506 for the completion-facing tolerance consumer.

Route statement. A separated-completion or metric-completion consumer may use $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ as a sibling lattice edge through the sequence

$$\mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow},$$

but it receives no quotient real equality, selected representative, ambient completion theorem, or host equivalence relation beyond the displayed classifier rows. The equality-facing reads of $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ are also consumed by the cut-to-stream comparison route of chapter 2312. Before the setoid route is compared with a located lower–upper cut packet, the exposed cut rows must display $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and the $\mathsf{Real}^{\uparrow}$ seal boundary, so the comparison remains a finite L10 dependency rather than quotient equality or a selected limit.

Theorem 1820.58 (Cauchy equivalence setoid Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *lattice must route through the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The setoid boundary may identify regular Cauchy names only after those sibling rows are displayed, and it exports no isolated quotient token that can bypass the L10 lattice.*

Proof. Start from the existing Cauchy-equivalence setoid packet and read its classifier route through the sibling dependency lattice. The displayed equality surface is attached to finite stream observations, regular rational readback, and dyadic tolerance comparisons before any real-facing consumer can inspect a seal.

The $\mathsf{StreamName}^{\uparrow}$ windows determine which finite observations are compared. The $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational source names, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the tolerance ledger that makes the Cauchy comparison meaningful.

Only after those rows have been retained may the $\mathsf{Real}^{\uparrow}$ seal be read. Therefore the setoid cannot export a detached quotient credential, selected representative, host real equality, or ambient completeness row; every Real-facing read stays on the displayed StreamName, RegSeqRat, DyadicRatCore, and Real route. □

The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row feeds the Cauchy–Dedekind gap-router comparison of chapter 5906 only after the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ classifier rows have been displayed. This keeps the equality-facing setoid as an L10 dependency route into the router rather than a quotient stream, selected representative, host real equality, or ambient completeness surface.

Theorem 1820.59 (Cauchy setoid modulus extraction route)**.** *Every* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *consumer that needs an explicit representative-tail schedule must route the tail read through* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *before it asks* $\mathsf{RegSeqRat}^{\uparrow}$ *for finite-window readback or* $\mathsf{Real}^{\uparrow}$ *for terminal seal rows. The extraction route is*

$$\mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{CauchyModulusBridge}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the setoid interface remains a finite schedule route for displayed representative tails, not a quotient-only sink and not a source of selected representatives, quotient Real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ consumer to the representative-tail rows it actually reads. An explicit tail schedule is not a bare setoid verdict: it contains a displayed request for the point at which the two representative reads are compared.

Route that request through chapter 5728. The $\mathsf{CauchyModulusBridge}^{\uparrow}$ ledger is the finite surface that mediates the schedule before any regular rational readback can be consumed.

After the bridge row is visible, the consumer may read the RegSeqRat$^{\uparrow}$ finite windows and then the terminal Real$^{\uparrow}$ seal rows. Since no step introduces a quotient representative, a selected limit, a choice schedule, or an ambient completeness coordinate, the setoid cannot be used as a quotient-only sink. □

The CauchyEquivalenceSetoid$^{\uparrow}$ lattice reads CauchyModulus$^{\uparrow}$ bridge data before any Real$^{\uparrow}$ seal consumer may treat tail equivalence as a usable completion classifier. The sibling dependency on chapter 5728 keeps representative-tail schedules visible as finite modulus rows, so downstream completion reads pass through displayed tail thresholds, regular rational windows, and the terminal Real seal rather than through an unrecorded quotient-equivalence sink.

Theorem 1820.60 (CauchyEquivalenceSetoid RealModulusExtraction forward link)**.** *Every* RealModulusExtractio *consumer that treats a regular-Cauchy witness as a stable real-facing modulus must first read the* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier rows of Definition 1820.1. The required route exposes two* StreamName$^{\uparrow}$ *windows, two* RegSeqRat$^{\uparrow}$ *readbacks, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the setoid-test row, the* Real$^{\uparrow}$ *seal boundary,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row before the downstream chapter 3848 consumer may name the modulus as stable.*

Proof. Project the carrier of Definition 1820.1. Its displayed coordinates include the paired stream windows and paired regular rational readbacks whose comparison is measured by the dyadic tolerance ledger and the setoid-test row.

The RealModulusExtraction$^{\uparrow}$ consumer reads a regular-Cauchy witness only as real-facing data. Therefore the route must pass through the same Real$^{\uparrow}$ seal boundary and preserve the comparison by the componentwise hsame transport and displayed Cont replay rows.

When the downstream modulus is read as an extracted Cauchy threshold, the same forward link also factors through chapter 641 before the Real-facing modulus row is named.

Finally, the provenance and local naming rows keep this as a CauchyEquivalenceSetoid$^{\uparrow}$ classifier dependency rather than a host equality or selected representative. The downstream chapter 3848 chapter can then consume a stable modulus route only after the listed classifier rows have been displayed. □

Theorem 1820.61 (CauchyEquivalenceSetoid located completion forward link)**.** *Every* LocatedCompactCompletio *or completion-facing consumer that reads a regular-Cauchy stream as a stable located real route must first consume the* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier side of Definition 1820.1. The route passes through the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located compact completion handoff of chapter 6172. It exposes paired stream windows, paired* RegSeqRat$^{\uparrow}$ *tails, the* Real$^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before the downstream completion packet may treat the classifier as stable.*

Proof. Project the classifier rows from Definition 1820.1. The setoid side compares two regular-Cauchy names by paired stream windows and paired regular rational tails, so a located completion consumer cannot read a single representative without first displaying the comparison rows.

The RegSeqRat and Real sibling chapters provide the forward handoff. The regular rational rows are read through chapter 301, and the terminal seal is read through chapter 38; the located compact completion consumer then receives the same classifier route through chapter 6172.

Transport, replay, provenance, and local naming keep the route inside the displayed CauchyEquivalenceSetoid$^{\uparrow}$ classifier. Thus the forward link supplies the classifier side for located completion consumers without selecting representatives, identifying quotient completions, choosing limits, or asserting host real equality. □

The CauchyEquivalenceSetoid$^{\uparrow}$ lattice reads RegSeqRat comparison rows through chapter 301 and terminal Real classifier rows through chapter 38. When the located dyadic real route of chapter 6192 is present, it supplies the finite located tolerance handoff between those rows without adding quotient stream equality, selected limits, or ambient Real completeness. The compact-metric continuous-image route reads CauchyEquivalenceSetoid$^{\uparrow}$ only through the displayed RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ classifier rows before any image-net or compactness consumer is reached. In particular, the finite image-net handoff of chapter 3605 and the total-bounded image carrier of chapter 3924 consume Cauchy equivalence as a regular-rational comparison surface with Real-seal readback, not as quotient stream equality, selected limits, host real equality, or ambient completeness. The CauchyEquivalenceSetoid$^{\uparrow}$ route is consumed by the Real$^{\uparrow}$ completion classifier of chapter 38. A regular Cauchy name routed through RegSeqRat$^{\uparrow}$ reaches the Real seal only after equality of regular Cauchy names has been read through the displayed setoid; the classifier exported by the Real completion surface therefore depends on this sibling row rather than on host quotient equality or a selected representative. The CauchyEquivalenceSetoid$^{\uparrow}$ route is consumed by the RealSeparability$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ classifier routes. A dense Real-name read in chapter 1809 and a regular-rational readback in chapter 301 both use this displayed setoid boundary before any equality-facing classifier can pass to a Real$^{\uparrow}$ seal; neither route receives host quotient equality, a selected representative, or an ambient equality theorem.

Theorem 1820.62 (Cauchy equivalence setoid seed closure for Real separability)**.** *Every* RealSeparability$^{\uparrow}$ *dense-read packet that consumes a Cauchy-name classifier reads the* CauchyEquivalenceSetoid$^{\uparrow}$ *route before exporting dense real rows. The consumed data are the displayed stream comparison, regular rational readback, metric-completion boundary, componentwise* hsame *transport,* Cont *replay, provenance, and local* NameCert *rows of this hub; the export does not acquire quotient real equality, selected representatives, countable choice, or ambient* Real$^{\uparrow}$ *completeness from the setoid classifier.*

Proof. Read the hub through its regular-rational and Real-separability sibling routes. They expose the Cauchy classifier as finite comparison data before the dense real consumer receives any exported row.

The dense read therefore factors through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, MetricCompletion$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert support already displayed by the setoid packet. Since none of those rows stores quotient real equality, a chosen representative, countable choice, or ambient completeness, the RealSeparability consumer cannot export such data from this dependency. □

The CauchyEquivalenceSetoid$^{\uparrow}$ hub records its substantive classifier dependencies through the regular rational readback of chapter 301, the terminal seal discipline of chapter 38, and the Cauchy-name comparison boundary of chapter 1412. These sibling chapters supply the visible rows that make the setoid a finite comparison route rather than a detached quotient or host equality object.

Theorem 1820.63 (CauchyEquivalenceSetoid forward dependency)**.** *Every* Real$^{\uparrow}$ *or completion-facing equality consumer of an accepted* CauchyEquivalenceSetoid$^{\uparrow}$ *classifier must expose the* StreamName$^{\uparrow}$ *finite windows, the* RegSeqRat$^{\uparrow}$ *readback rows, the* DyadicRatCore$^{\uparrow}$ *tolerance rows, and the terminal* Real$^{\uparrow}$ *seal before it consumes the setoid classifier. The route is the sibling-facing dependency surface of chapter 126, chapter 301, chapter 291, and chapter 38; it exports no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient completion principle.*

Proof. Project the carrier of Definition 1820.1. Its equality-facing rows begin with two StreamName windows, two RegSeqRat readbacks, and the DyadicRatCore tolerance ledger before the setoid-test row is available.

Apply the classifier-law and NameCert-surface theorems, Definition 1820.3 and Definition 1820.5. They keep reflexivity, symmetry, transitivity, tolerance exactness, transport, replay, provenance, and local naming inside the displayed packet.

Use the Real seal forward route of Definition 1820.8. It places the terminal $\mathsf{Real}^{\uparrow}$ seal after the visible stream, readback, tolerance, classifier, transport, replay, provenance, and naming rows.

Thus any Real or completion-facing equality consumer reads the classifier only after those sibling rows have been exposed. Since the carrier has no quotient, selected-representative, choice, host-equality, or ambient-completion coordinate, none of those objects can replace the displayed dependency route. □

Chapter 1821

A Concrete Naming Certificate for $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$

Following Bishop and Bridges constructive regular-Cauchy completion, the $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$ packet isolates the finite limit-seal surface consumed by the L10 completion route. It reads the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It also points to the sequential-completeness route of chapter 1794, the uniqueness boundary of chapter 1782, and the universal route of chapter 1425. The packet names only the displayed finite handoff from regular rational Cauchy data into a Real-facing seal; it does not assert quotient streams, selected host limits, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1821.1 (Bishop regular Cauchy limit seal carrier)**.** *A* $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, M, E, U, V, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to ask finite regular-Cauchy precision questions.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the tolerance ledger.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row exposed inside the displayed windows.*
4. *M is the regular tail-control row witnessing that each dyadic request reads a finite threshold before inspecting S and R.*
5. *E is the displayed* $\mathsf{Real}^{\uparrow}$ *limit seal reached from D, S, R, M.*
6. *U is the* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *handoff row that records the finite route from regular sequence data to the displayed seal.*
7. *V is the completion comparison row, read through* $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$ *and* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *after the seal row E has been exposed.*
8. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, tail-control row, seal row, sequential-completeness handoff, completion comparison row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a limit-seal request through D, S, R, M, E, U, V.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, M, E, U, V, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected host limits, countable-choice selectors, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness data, or bridge credentials.*

Theorem 1821.2 (Bishop regular Cauchy limit seal NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$ *carrier*

$$L = (D, S, R, M, E, U, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance ledger, stream-window row, regular rational readback row, regular tail-control row, Real limit seal, sequential-completeness handoff, completion comparison row, transport, replay, provenance, and local naming row;*
2. *regular-tail discipline, requiring every tolerance request to obtain its threshold from M before it reads the finite rows D, S, R;*
3. *seal handoff, requiring the displayed* $\mathsf{Real}^{\uparrow}$ *row E to be reached only after the regular route D, S, R, M has been replayed;*
4. *sequential-completeness compatibility, requiring U to consume the same D, S, R, M, E coordinates and no separate completion carrier;*
5. *uniqueness and universal compatibility, requiring V to compare completion-facing uses only after the displayed seal row and the common finite regular-Cauchy route have been fixed;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient stream equality, selected host limit, countable-choice selector, host real equality, ambient Real completeness theorem, metric-completeness datum, or bridge credential.*

No NameCert obligation is stored outside these rows.

```
Lean4 ✓  BEDC.Derived.BishopRegularCauchyLimitSealUp.BishopRegularCauchyLimitSealTasteGate
single carrier alignment
```

Proof. Carrier admission is projection from Definition 1821.1. The accepted packet displays the dyadic ledger, finite windows, regular readback, tail-control row, seal row, sequential-completeness handoff, completion comparison row, and the structural rows H, C, P, N.

The regular-tail discipline is forced by the order of the replay row. A tolerance request first reads M, then inspects the corresponding dyadic, window, and regular rational rows D, S, R.

The seal handoff is downstream of that same route. The row E is the only displayed $\mathsf{Real}^{\uparrow}$ limit seal, so a consumer cannot obtain a Real-facing endpoint before D, S, R, M have been exposed.

Sequential-completeness compatibility follows because U is a row over the displayed route, not an independent completion object. The completion comparison row V is likewise read after E and can compare only the already displayed finite route.

Finally, H, C, P, N preserve and name the listed rows. Since the carrier contains none of the excluded coordinates, the local certificate cannot export quotient stream equality, selected host

limits, choice, host real equality, ambient Real completeness, metric-completeness data, or bridge credentials. □

Theorem 1821.3 (Bishop regular Cauchy limit seal nonescape)**.** *Every Real-facing or completion-facing read of an accepted* $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchySequentialCompleteness}^{\uparrow},\ \mathsf{RegularCauchyCompletionUni}$$

The exported limit seal is the displayed row E reached from D, S, R, M and then consumed by U, V, H, C, P, N. It exports no quotient stream equality, selected host limit, countable-choice selector, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, metric-completeness datum, or bridge credential.*

Proof. Project the accepted carrier. A Real-facing or completion-facing read can access only the rows $D, S, R, M, E, U, V, H, C, P, N$.

The read first follows the regular-Cauchy data route. The dyadic tolerance ledger D opens the finite windows S, the row R supplies regular rational readback, and M supplies the finite tail-control threshold for the same request.

Only after this route is displayed can the read reach E as the $\mathsf{Real}^{\uparrow}$ limit seal. The rows U and V then record sequential-completeness and completion comparison uses of that same seal, rather than selecting a host limit or a quotient representative.

The structural rows close the route by componentwise transport, replay, provenance, and local naming. Because no excluded coordinate occurs in the packet, every accepted read remains inside the finite limit-seal surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, a RegSeqRat regular rational readback row, a regular tail-control row, a displayed RealUp limit seal, RegularCauchySequentialCompleteness handoff, RegularCauchyCompletionUniqueness and RegularCauchyCompletionUniversal comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1821.1, Definition 1821.2, and Definition 1821.3 seed the finite L10 limit-seal surface from regular rational Cauchy data into the displayed $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, metric-completeness data, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding carrier admission, regular-tail discipline, seal handoff, sequential-completeness compatibility, uniqueness and universal compatibility, and ledger non-escape to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Chapter 1822

A Concrete Naming Certificate for $\mathsf{CompactMetricProduct}^{\uparrow}$

Following Bishop and Bridges constructive metric compactness and Kelley-style product metrics, the $\mathsf{CompactMetricProduct}^{\uparrow}$ packet records binary finite-product compactness for L10 compact metric spaces as a displayed BEDC certificate. It reads the product-metric ledger of chapter 424, the compact metric source rows of chapter 129, the finite-net surface of chapter 127, the complete-limit surface of chapter 128, the finite epsilon-net rows of chapter 1767, and the binary product carrier of chapter 34. The packet names the standard constructive route: build finite product nets componentwise, pair complete-limit witnesses componentwise, and expose the resulting compact-metric package only through the displayed product metric.

Definition 1822.1 (Compact metric product carrier)**.** *A* $\mathsf{CompactMetricProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, M, P, T, L, H, C, Q, N)$$

with the following displayed rows:

1. *X is the first compact metric source row. It exposes its* $\mathsf{CompactMetric}^{\uparrow}$ *packet only through the associated* $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *and* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *rows.*
2. *Y is the second compact metric source row, with the same displayed compact-metric boundary for the second component.*
3. *M is the* $\mathsf{ProductMetric}^{\uparrow}$ *row on the binary product carrier. It stores the two projection routes, the component distance reads, and the product-distance ledger.*
4. *P is the* $\mathsf{Prod}^{\uparrow}$ *pairing row carrying the product points inspected by M, together with the two displayed projection histories.*
5. *T is the finite epsilon-net product witness. A tolerance request is split into component tolerances, the component* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witnesses are read from X and Y, and their paired centers form the product finite-net row.*
6. *L is the paired complete-limit witness row. A regular Cauchy stream in the product metric is projected through M to the two component* $\mathsf{CompleteMetric}^{\uparrow}$ *witnesses, then repacked through P as the displayed product limit row.*
7. *H records componentwise* hsame *transport for the two compact metric sources, the product metric row, the product carrier row, the finite-net product witness, the paired limit witness, replay, provenance, and local name.*

8. *C records displayed* Cont *replay from a compactness request through*

$$X, Y \longmapsto M, P \longmapsto T, L \longmapsto H, Q, N.$$

9. *Q is the* Pkg *provenance over* $\mathsf{CompactMetricProduct}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, M, P, T, L, H, C, Q, N$. The carrier has no coordinate for open-cover compactness, sequential compactness, quotient product equality, selected limits, countable choice, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompactMetricProductUp.TasteGate.CompactMetricProductTasteGate single carrier alignment`

Theorem 1822.2 (Compact metric product NameCert obligations)**.** *For every accepted* $\mathsf{CompactMetricProduct}^{\uparrow}$ *carrier*

$$K = (X, Y, M, P, T, L, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricProduct}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *carrier admission for the two compact metric sources, product metric row, product carrier row, finite-net product witness, paired complete-limit witness, transport, replay, provenance, and local name;*
2. *product-metric exactness, requiring every product distance and projection read to pass through the displayed* $\mathsf{ProductMetric}^{\uparrow}$ *row M and* $\mathsf{Prod}^{\uparrow}$ *row P;*
3. *finite-net product coverage, requiring T to be assembled from the two component* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *rows and the product-distance ledger of M;*
4. *complete-limit pairing, requiring L to project every product Cauchy stream to the two component* $\mathsf{CompleteMetric}^{\uparrow}$ *witnesses before repacking the limit through P;*
5. *non-escape provenance, requiring H, C, Q, N to preserve the displayed rows and to export no open-cover compactness, sequential compactness, quotient product equality, selected limit, countable-choice selector, or external compactness theorem.*

No local NameCert obligation is stored outside these five rows.

Lean4 ✓ `BEDC.Derived.CompactMetricProductUp.CompactMetricProductTasteGate single carrier alignment`

Proof. The carrier row already lists the only admissible coordinates. Carrier admission is therefore the tuple inspection of $X, Y, M, P, T, L, H, C, Q, N$, while product-metric exactness is the projection and distance replay inherited from M and P.

For total boundedness, read the positive tolerance through the component tolerance split recorded by T. The two component compact metric sources expose their finite epsilon-net rows, and the product row pairs the chosen centers through P. The product-distance ledger in M then gives the displayed coverage comparison.

For completeness, project a product Cauchy stream through the two projection routes of M. The component complete-limit rows in X and Y supply the two limits, and L repacks those limits through the binary product carrier P.

Finally, H, C, Q, N only replay and name the rows already displayed. Since no listed coordinate stores an open-cover argument, sequential compactness theorem, quotient equality, selected limit, or countable-choice selector, none can be exported by the local NameCert surface. □

Theorem 1822.3 (Compact metric product component handoff)**.** *Let $K = (X, Y, M, P, T, L, H, C, Q, N)$ be an accepted* $\mathsf{CompactMetricProduct}^{\uparrow}$ *carrier. Every consumer read of compactness for the product factors through the two component* $\mathsf{CompactMetric}^{\uparrow}$ *rows X, Y and the* $\mathsf{ProductMetric}^{\uparrow}$ *projection ledger M. In particular, the only compactness data visible to the consumer are the component finite-net rows, the component complete-limit rows, their paired transport through P, and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *rows of K.*

Lean4 ✓ `BEDC.Derived.CompactMetricProductUp.CompactMetricProductComponentHandoff`

Proof. Start with an arbitrary product compactness consumer. The carrier has no compactness coordinate except T and L, so the consumer must ask for finite coverage or complete-limit data through those rows.

The finite-coverage request reaches T. That row is defined by reading the finite epsilon-net witnesses from X and Y, then pairing the selected centers through P and measuring coverage through M.

The complete-limit request reaches L. That row is defined by projecting the product Cauchy stream through M, invoking the component complete-limit witnesses carried by X and Y, and repacking the two limits through P.

Thus every compactness read is componentwise before it is product-facing. The remaining rows H, C, Q, N transport, replay, source, and name this same route without adding any external compactness principle. □

Theorem 1822.4 (CompactMetric package for binary products)**.** *Every accepted* $\mathsf{CompactMetricProduct}^{\uparrow}$ *carrier*

$$K = (X, Y, M, P, T, L, H, C, Q, N)$$

supplies a $\mathsf{CompactMetric}^{\uparrow}$ *package for the binary product named by P and metrized by M. Its total-bounded component is the finite epsilon-net product witness T, and its complete component is the paired complete-limit witness L. The package is exposed only through the displayed rows $X, Y, M, P, T, L, H, C, Q, N$.*

Proof. The product metric row M supplies the metric carrier for the binary product row P. This gives the target metric surface on which a $\mathsf{CompactMetric}^{\uparrow}$ package can be read.

The total-bounded half is T. For each admitted tolerance, T reads finite epsilon-net witnesses from the component compact metric rows, pairs their centers, and uses the product-distance ledger to prove product coverage.

The complete half is L. A product Cauchy stream is projected to the two component streams, the component complete-limit witnesses produce component limits, and the binary product row repacks those limits as the product limit.

Combining these two rows gives the compact metric package for the product. The package has no route to open-cover compactness, sequential compactness, quotient product equality, selected limits, countable choice, Lean verification, or bridge checking, because those data are absent from the carrier and from its replay and provenance rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricProduct}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1822.1, Definition 1822.2, Definition 1822.3, and Definition 1822.4 seed the constructive finite-product compactness certificate for L10 compact metric spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, quotient product equality, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-metric exactness, finite-net product coverage, complete-limit pairing, component handoff, provenance non-escape, and local naming for BEDC.Derived.CompactMetricProductUp.

Constructive story: A CompactMetricProduct$^{\uparrow}$ packet is generated from two CompactMetricUp sources, one ProductMetricUp row, one ProdUp product carrier, finite epsilon-net product witnesses, paired CompleteMetricUp limit witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1823

A Concrete Naming Certificate for $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$ packet records the standard passage from a displayed convergent real sequence to a displayed Cauchy criterion as finite BEDC route data. It sits over the displayed limit row of chapter 1413, the criterion row of chapter 1509, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic triangle ledger of chapter 291, and the terminal seal boundary of chapter 38.

Definition 1823.1 (Convergent sequence Cauchy carrier)**.** *A* $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, W, D, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row containing the displayed source sequence, candidate limit, tail-window schedule, dyadic tolerance ledger, and convergence classifier.*
2. *W is the paired finite tail-window schedule. For each requested dyadic tolerance it stores two tail observations of the displayed sequence and the matching observation windows of the displayed limit row.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *triangle ledger. It records the two sequence-to-limit comparisons from L and the finite triangle comparison routing the two tail observations through the displayed limit.*
4. *K is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row obtained from the same paired tail-window schedule and dyadic triangle ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled sequence observations and the displayed limit observations used by L, W, D, K.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the displayed limit row, paired tail windows, dyadic ledger, criterion row, and regular rational readback have been exposed.*
7. *H records componentwise* hsame *transports for L, W, D, K, R, E and the structural rows.*
8. *C records displayed* Cont *replay from a Cauchy request through the limit row, paired tail windows, dyadic ledger, criterion handoff, regular rational readback, and terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvergentSequenceCauchy}^{\uparrow}}$ *row.*

The carrier stores a finite route from displayed convergence data to displayed Cauchy data. It contains no host sequence convergence theorem, selected limit, quotient stream equality, countable-choice selector, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.ConvergentSequenceCauchyUp.ConvergentSequenceCauchyTasteGate single carrier alignment`

Theorem 1823.2 (Convergent sequence Cauchy NameCert obligations)**.** *For every accepted* ConvergentSequenceCa *carrier $Q = (L, W, D, K, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConvergentSequenceCauchy}^{\uparrow}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier admission for the displayed* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row, paired tail-window schedule, dyadic triangle ledger,* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local name;*
2. *limit-source exposure, requiring every tail comparison to read the displayed convergence classifier stored in L;*
3. *paired-window synchronization, requiring the two sequence observations and the two limit observations to be taken from the same tolerance request in W;*
4. *triangle-ledger exactness, requiring the Cauchy comparison to be assembled only from the two sequence-to-limit comparisons recorded in D;*
5. *criterion handoff, requiring the row K to reuse the same finite windows and dyadic ledger rather than introducing a second sequence representative;*
6. *seal boundary, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to remain downstream of L, W, D, K, R and the structural rows.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 1823.1. Its visible coordinates are exactly the displayed limit row, paired tail-window schedule, dyadic triangle ledger, criterion row, regular rational readback, terminal seal, transport, replay, provenance, and local naming row.

The limit-source and paired-window obligations are forced by L and W. A Cauchy request first selects one tolerance request, then W exposes two tail observations together with the matching limit observations certified by the displayed convergence row L.

The triangle-ledger obligation is read from D. The ledger contains the two sequence-to-limit comparisons and the finite triangle route comparing the two tail observations through the displayed limit, so no detached comparison row is available.

The criterion handoff is the projection to K. Since K is part of the same packet and reuses W, D, R, the local certificate names the Cauchy criterion without adding a second stream representative or a selected host limit.

The seal boundary follows from the ordering of the rows. The terminal seal E can be consumed only after the displayed limit, paired windows, dyadic ledger, criterion handoff, and regular rational readback have been exposed, while H, C, P, N transport, replay, source, and name the same finite route. □

Theorem 1823.3 (Convergent sequence Cauchy tail-window factorization)**.** *Let $Q = (L, W, D, K, R, E, H, C, P, N$ be an accepted* $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *consumer*

of Q factors through the finite route

$$L \longmapsto W \longmapsto D \longmapsto K$$

before it can inspect the regular rational readback R, the terminal $\mathsf{Real}^{\uparrow}$ *seal E, or the structural rows H, C, P, N. The route compares two tail observations only by routing each observation through the displayed limit row and then through the dyadic triangle ledger.*

Proof. Begin with a criterion-facing consumer of Q. Carrier admission gives the consumer no sequence source except the displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ row L.

The request then passes to the paired window schedule W. This schedule supplies two tail observations for the same dyadic tolerance request, together with the limit-side observation windows needed to read the convergence classifier in L.

Next read the dyadic triangle ledger D. The ledger is the only row comparing the two tail observations, and it does so by composing the two sequence-to-limit comparisons through the displayed limit.

The handoff to K now has exactly the data required by the $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ surface: finite windows, dyadic tolerances, and a regular rational readback route. Only after this handoff may the consumer inspect R, E, H, C, P, N, so the criterion read factors through L, W, D, K. □

Theorem 1823.4 (Convergent sequence Cauchy Real-seal boundary)**.** *For an accepted* $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$ *carrier $Q = (L, W, D, K, R, E, H, C, P, N)$, every terminal* $\mathsf{Real}^{\uparrow}$*-facing read is downstream of the displayed convergence row L, the paired tail-window schedule W, the dyadic triangle ledger D, the criterion row K, and the regular rational readback R. The read exports only the finite NameCert route from displayed convergence to displayed Cauchy data, not host sequence convergence, a selected limit, quotient stream equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Start from the accepted packet. The terminal seal row E is not a source coordinate; it is listed after the displayed limit row, paired window schedule, dyadic ledger, criterion row, and regular rational readback.

A terminal consumer must therefore traverse L, W, D, K, R in order. The rows L, W, D supply the finite convergence-to-Cauchy triangle route, while K names the Cauchy criterion handoff and R supplies the regular rational observations used by that handoff.

The structural rows preserve the same route. Transport H is componentwise, replay C follows the displayed request path, provenance P records the same dependencies, and N names the local certificate.

Since the carrier has no coordinate for host sequence convergence, selected limits, quotient stream equality, countable choice, trichotomy, ambient real completeness, Lean verification, or bridge checking, the terminal seal read exports none of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvergentSequenceCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConvergentSequenceCauchy}^{\uparrow}$ packet is generated from a displayed RealSequenceLimit row, paired finite tail-window schedule, DyadicRatCore triangle ledger, CauchyConvergenceCriterion handoff, RegSeqRat readback, terminal RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvergentSequenceCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1823.1, Definition 1823.2, Definition 1823.3, and Definition 1823.4 seed the finite route from displayed convergence data to displayed Cauchy criterion data through paired tail windows and a dyadic triangle ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvergentSequenceCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence convergence, selected limits, quotient stream equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, limit-source exposure, paired-window synchronization, triangle-ledger exactness, criterion handoff, and Real-seal boundary for the same displayed packet.

Chapter 1824

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$ packet records the finite endpoint handoff by which regular Cauchy completion data may be read by public $\mathsf{Real}^{\uparrow}$ and Completion consumers without importing a quotient stream, a selected representative, or an ambient completeness principle. It sits after the regular-completion carrier of chapter 1639, the limit-seal route of chapter 386, the exact-boundary row of chapter 1120, the source-to-endpoint unit of chapter 1527, and the terminal real-name seal of chapter 38. The packet is an endpoint boundary: it names the handoff surface consumed by later Real and Completion routes, not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1824.1 (Regular Cauchy completion endpoint carrier)**.** *A* $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, L, X, U, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *row containing the regular rational sequence route, dyadic tail modulus, common-tail window, and completion-facing carrier boundary.*
2. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *route that seals the displayed regular Cauchy handoff before any endpoint consumer reads it.*
3. *X is the* RealCompletionExactBoundaryUp *exact-boundary row binding the StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, and terminal Real faces into one finite completion route.*
4. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-to-endpoint row, recording how the displayed source name enters the endpoint boundary without selecting a representative.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after B, L, X, U have been displayed.*
6. *H records componentwise* hsame *transport for the regular-completion row, limit-seal route, exact-boundary row, unit row, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an endpoint request through B, L, X, U, R and then through the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, RealCompletionExactBoundaryUp, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}}$ *row.*

The classifier accepts only packets read through $B, L, X, U, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyCompletionEndpointUp.RegularCauchyCompletionEndpointUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEndpointUp.RegularCauchyCompletionEndpointTasteGate`
`single carrier alignment`

Theorem 1824.2 (Regular Cauchy completion endpoint NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$ *carrier $E = (B, L, X, U, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCor}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the regular-completion row, limit-seal route, exact-boundary row, source-to-endpoint unit, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *classifier transport, requiring H to preserve B, L, X, U, R, C, P, N componentwise and requiring C to replay the same endpoint route;*
3. *endpoint seal handoff, requiring every endpoint-facing read to pass through B, L, X, U before exposing R;*
4. *unit and embedding compatibility, requiring the unit row U to reuse the displayed source route already admitted by B and sealed by L, X;*
5. *nonescape, requiring the packet to export no quotient stream equality, selected representative, countable choice, host real equality, or ambient Real completeness.*

No endpoint obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEndpointUp.RegularCauchyCompletionEndpointCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 1824.1. The projection exposes exactly the regular-completion row, limit-seal route, exact-boundary row, unit row, terminal seal, and structural rows.

Carrier admission is therefore rowwise. The completion-facing source is admitted through B, the seal route through L, the exact endpoint boundary through X, the unit handoff through U, and the terminal Real surface through R.

Classifier transport is componentwise. The row H preserves each visible coordinate, while C replays the ordered route from an endpoint request through B, L, X, U, R.

The endpoint seal handoff and unit compatibility are forced by the placement of R downstream of B, L, X, U. A consumer cannot read the terminal $\mathsf{Real}^{\uparrow}$ seal before the displayed regular-completion source, limit-seal route, exact boundary, and unit row have all been exposed.

Finally, the carrier inventory contains no excluded coordinate. Since quotient stream equality, selected representatives, countable choice, host real equality, and ambient Real completeness are absent from the tuple and from the replay route, the nonescape obligation is exhausted by $B, L, X, U, R, H, C, P, N$. □

Theorem 1824.3 (Regular Cauchy completion endpoint Real-seal handoff)**.** *Let $E = (B, L, X, U, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or Completion endpoint consumer of E factors through the displayed route*

$$B \longmapsto L \longmapsto X \longmapsto U \longmapsto R$$

together with the structural rows H, C, P, N. Consequently the consumer receives only the regular-completion carrier, CauchyLimitSeal route, RealCompletionExactBoundary row, CauchyCompletionUnit handoff, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert rows, and receives no quotient stream equality, selected representative, countable choice, host real equality, or ambient Real completeness.

Proof. Begin with an endpoint consumer of the accepted packet. By carrier projection, its only completion-facing source is B, the displayed $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ row.

The consumer next reads L and X. The row L supplies the Cauchy-limit seal for the same regular-completion route, while X binds that sealed route to the exact Real-completion boundary before an endpoint row is public.

The source-to-endpoint step is U. This row records the unit handoff from the displayed source route into the endpoint boundary and does not add a second representative or a hidden selector.

Only after B, L, X, U have been read may the terminal seal R be exposed. The rows H, C, P, N transport, replay, source, and name the same ordered route, so every endpoint consumer factors through the listed packet.

The excluded principles have no carrier slot and no replay insertion point. Hence the consumer receives no quotient stream equality, selected representative, countable choice, host real equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}$ packet is generated from a BishopRegularCauchyCompletion row, CauchyLimitSeal route, RealCompletionExactBoundary row, CauchyCompletionUnit source-to-endpoint row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1824.1, Definition 1824.2, and Definition 1824.3 seed the endpoint handoff from regular Cauchy completion packets to public RealUp and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEndpoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires future Lean work under BEDC.Derived.RegularCauchyCompletionEndpointUp for carrier admission, classifier transport, endpoint seal handoff, unit and embedding compatibility, and nonescape.

Chapter 1825

A Concrete Naming Certificate for $\mathsf{CauchyEstimate}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyEstimate}^{\uparrow}$ packet records finite Cauchy error-estimate data for regular rational real names. It consumes the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy threshold discipline of chapter 292. The packet gives the L10 Real and Completion route one visible estimate ledger before any $\mathsf{Real}^{\uparrow}$-facing consumer can read a completed seal.

Definition 1825.1 (Cauchy estimate carrier)**.** *A* $\mathsf{CauchyEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, L, M, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row containing the regular rational name whose estimates are being read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It lists exactly the observation windows of R that an estimate request may inspect.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic scale, comparison arithmetic, and tolerance conversions used by the estimate ledger.*
4. *L is the finite estimate ledger. Each row pairs a requested tolerance with the two scheduled rational observations, the dyadic comparison witnessing their distance bound, and the local row in which the bound is stored.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning every admitted tolerance request to the window threshold used by W and L.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, L, M have been displayed.*
7. *H records componentwise* hsame *transport for the source row, finite windows, dyadic tolerance row, estimate ledger, modulus row, real seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from an estimate request through $R, W, D, L, M, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyEstimate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient real equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyEstimateUp.CauchyEstimateTasteGate single carrier alignment`

Theorem 1825.2 (Cauchy estimate NameCert obligations)**.** *For every accepted* $\mathsf{CauchyEstimate}^{\uparrow}$ *carrier*

$$E = (R, W, D, L, M, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyEstimate}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, estimate ledger,* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *seal row, transport, replay, provenance, and local name;*
2. *estimate-window exactness, requiring every ledger entry in L to use only the finite windows displayed by W for the admitted tolerance request;*
3. *dyadic tolerance stability, requiring every estimate comparison to be read through the dyadic scale and comparison arithmetic stored in D;*
4. *classifier transport, requiring H and C to preserve the source, window, tolerance, ledger, modulus, seal, provenance, and local naming rows componentwise under* hsame *and displayed replay;*
5. *ledger non-escape, requiring the packet to export no estimate row outside $R, W, D, L, M, S, H, C, P, N$.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1825.1. The projection exposes one regular rational source, one finite-window schedule, one dyadic tolerance row, one estimate ledger, one modulus row, one terminal real seal, and the structural rows.

Carrier admission is therefore exactly admission of those displayed coordinates. Since the source row R and the window row W are the only observation coordinates, an estimate request cannot read a rational observation outside the finite windows named by W.

The dyadic tolerance obligation is read from D. Each ledger row in L cites the dyadic scale and comparison arithmetic stored there, so stability of the bound is a property of the displayed tolerance row rather than of an ambient real-comparison principle.

Transport and replay are structural. The row H transports the same displayed coordinates componentwise, and the row C repeats the route from an estimate request through the source, windows, dyadic tolerance row, ledger, modulus, seal, provenance, and local name.

The remaining non-escape clause follows from the coordinate list. The packet has no additional estimate surface and no coordinate for real completeness, quotient real equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking. □

Theorem 1825.3 (Cauchy estimate Real seal handoff)**.** *Let $E = (R, W, D, L, M, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyEstimate}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing estimate read factors through the finite route*

$$W \longmapsto R \longmapsto D \longmapsto M \longmapsto L$$

before the real seal row S is visible. The read exports only the displayed StreamName windows, RegSeqRat observations, DyadicRatCore tolerance comparisons, CauchyModulus threshold row, estimate ledger, real seal, and structural rows, not $\mathsf{Real}^{\uparrow}$ *completeness, quotient real equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking.*

Proof. Start with a real-facing estimate consumer of the accepted packet. The only real-facing coordinate is S, and it is listed downstream of the regular rational source, finite windows, dyadic tolerance row, estimate ledger, and modulus row.

The consumer first reads the finite windows W and the source observations R. These rows determine which rational observations can participate in an estimate request.

Next the consumer reads D and M. The row D supplies the dyadic tolerance arithmetic, while M supplies the threshold discipline saying which windows discharge the admitted tolerance.

Only then can the estimate ledger L be inspected. The ledger binds the selected windows, regular rational observations, dyadic comparisons, and modulus threshold into the displayed estimate row.

Thus the seal row S is visible only after the finite StreamName, RegSeqRat, DyadicRatCore, modulus, and ledger rows have been read. Since the carrier has no coordinate for completeness, quotient equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking, none of those principles can be exported by the real-facing read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyEstimate}^{\uparrow}$ packet is generated from one RegSeqRat source row, one StreamName finite-window row, one DyadicRatCore tolerance row, a finite estimate ledger, one CauchyModulus threshold row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1825.1, Definition 1825.2, and Definition 1825.3 seed the finite error-estimate ledger route from regular rational observations through dyadic tolerances and a Cauchy modulus before the Real seal is consumed.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyEstimate}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyEstimateUp`

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, estimate-window exactness, dyadic tolerance stability, classifier transport, ledger non-escape, and Real-seal handoff for BEDC.Derived.CauchyEstimateUp.

Chapter 1826

A Concrete Naming Certificate for $\mathsf{DiagonalSubsequence}^{\uparrow}$

Following Cantor's diagonal method and Bishop–Bridges constructive analysis, the $\mathsf{DiagonalSubsequence}^{\uparrow}$ packet records the finite-window diagonal subsequence extraction route used by the L10 Real and Completion spine. It sits over the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the regular Cauchy extraction route of chapter 1773, and the downstream diagonal-limit seal of chapter 1800. The packet names diagonal subsequence extraction as displayed BEDC route data, not as a quotient stream, a countable-choice selector, a host limit, trichotomy, or an ambient completeness theorem.

Definition 1826.1 (Diagonal subsequence carrier)**.** *A* $\mathsf{DiagonalSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, W, J, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence window row. It exposes only finite* $\mathsf{StreamName}^{\uparrow}$ *windows of the displayed family from which the diagonal subsequence is read.*
2. *W is the finite diagonal window ledger. For each requested finite diagonal stage it stores the source window, target window, tolerance budget, and compatibility check visible at that stage.*
3. *J is the monotone unary selector row. It records a displayed finite selector trace whose successive entries are strictly later source positions and whose order is checked inside the packet.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windows selected by S, W, J.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic radii, slack entries, and comparison budgets used by W, J, R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, diagonal ledger, selector row, regular rational readback, and dyadic tolerance ledger have been displayed.*
7. *H records componentwise* hsame *transport for S, W, J, R, Q, E and the structural rows.*
8. *C records displayed* Cont *replay from a finite diagonal request through $S, W, J, R, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DiagonalSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, J, R, Q, E, H, C, P, N$. It has no coordinate for a quotient stream, countable-choice selector, host limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DiagonalSubsequenceUp.TasteGate.DiagonalSubsequenceTasteGate single carrier alignment`

Theorem 1826.2 (Diagonal subsequence NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalSubsequence}^{\uparrow}$ *carrier*

$$D = (S, W, J, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiagonalSubsequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source-sequence windows, finite diagonal window ledger, monotone unary selector row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *finite-window diagonal compatibility, requiring each diagonal stage in W to expose only the source and target windows selected from S for the same displayed tolerance request;*
3. *selector monotonicity, requiring J to witness that later diagonal stages read strictly later source positions by a finite unary trace;*
4. *readback exactness, requiring R to be assembled only from the selected windows and tolerance entries recorded in S, W, J, Q;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E to be downstream of the finite source-window, diagonal-ledger, selector, readback, and tolerance route;*
6. *componentwise* hsame *transport through H for all displayed analytic and structural rows;*
7. *displayed* Cont *replay through C for every finite diagonal request;*
8. Pkg *provenance through P for the same dependency surface;*
9. *local naming through N and refusal of quotient streams, countable-choice selectors, host limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as certificate data.*

No diagonal subsequence obligation is stored outside these rows.

Proof. Project the carrier of Definition 1826.1. The visible analytic rows are exactly the source windows S, finite diagonal ledger W, monotone selector row J, regular rational readback R, dyadic tolerance ledger Q, and terminal real seal E.

Finite-window diagonal compatibility is read from S and W. A diagonal request chooses one finite stage, and the ledger exposes only the source and target windows assigned to that same displayed tolerance request.

Selector monotonicity is read from J. Since J is a finite unary trace inside the packet, the local certificate can check that each later diagonal stage uses a later source position without importing a countable-choice schedule.

Readback exactness is the projection to R and Q. The regular rational row is assembled from the selected windows and dyadic tolerance entries already displayed by S, W, J, Q, so no quotient stream or detached representative is available.

The seal, transport, replay, provenance, and naming rows preserve the same route. The terminal seal E is downstream of S, W, J, R, Q, while H, C, P, N transport, replay, record, and name only the listed packet rows. □

Theorem 1826.3 (Diagonal subsequence regular readback route)**.** *Every accepted* $\mathsf{DiagonalSubsequence}^{\uparrow}$ *packet*

$$D = (S, W, J, R, Q, E, H, C, P, N)$$

routes each $\mathsf{Real}^{\uparrow}$*-facing diagonal-subsequence read through the finite chain*

$$S \longmapsto W \longmapsto J \longmapsto R \longmapsto Q \longmapsto E$$

before any structural row is consumed. Consequently the read exposes source windows, a finite diagonal window ledger, a monotone unary selector trace, a $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it exports no quotient stream, countable-choice selector, host limit, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing consumer of the accepted packet. The only source data visible before a seal read are the finite windows in S.

The diagonal ledger W then binds the consumer to one finite diagonal stage. It supplies the source and target windows for that stage and records the tolerance budget that the later readback must respect.

The selector row J is next. It certifies that the selected source positions form a monotone unary trace, so the consumer can read a subsequence route without selecting an infinite function outside the packet.

The rows R and Q assemble the regular rational readback and the dyadic tolerance ledger from the same displayed windows and selector trace. Thus the terminal seal E is reached only after S, W, J, R, Q have been consumed.

Since the carrier contains no quotient stream, countable-choice selector, host limit, trichotomy row, or ambient completeness theorem, the displayed route exhausts every real-facing diagonal-subsequence read. □

Theorem 1826.4 (Diagonal subsequence Cauchy-window extraction)**.** *Every accepted* $\mathsf{DiagonalSubsequence}^{\uparrow}$ *packet*

$$D = (S, W, J, R, Q, E, H, C, P, N)$$

extracts each $\mathsf{RegSeqRat}^{\uparrow}$ *readback window from the finite source window row* S*, diagonal ledger* W*, monotone selector row* J*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Q*. The terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is downstream of that extracted Cauchy window and cannot be read from a quotient stream or from a countable-choice selector.*

Proof. Project the carrier of Definition 1826.1. The source windows S, diagonal ledger W, selector trace J, readback row R, tolerance ledger Q, and seal E are the only analytic rows available to a real-facing diagonal-subsequence consumer.

Use Definition 1826.2. Its readback exactness clause requires R to be assembled only from the selected windows and tolerance entries recorded in S, W, J, Q.

Now apply Definition 1826.3. The ordered route places E after S, W, J, R, Q, so the terminal seal is reached only after the extracted Cauchy window has been displayed. Since the carrier has no quotient-stream or countable-choice coordinate, no consumer can bypass that finite extraction route. □

Bounded-sequence compactness consumers read this diagonal route through chapter 1254 and chapter 1415 when a retained finite-window subsequence is repacked as a cluster witness or compactness witness. The diagonal packet therefore supplies the finite extraction surface consumed by the L10 bounded-sequence lattice rather than floating only as a kernel-facing route.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalSubsequence}^{\uparrow}$ packet is generated from source-sequence windows, a finite diagonal window ledger, a monotone unary selector row, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1826.1, Definition 1826.2, and Definition 1826.3 seed the finite-window diagonal subsequence extraction route through source windows, diagonal ledgers, monotone selector traces, regular readback, dyadic tolerances, and a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, countable-choice selectors, host limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and TasteGate field faithfulness for BEDC.Derived.DiagonalSubseque

Chapter 1827

A Concrete Naming Certificate for $\mathsf{CesaroConvergence}^{\uparrow}$

Following Hardy's classical treatment of Cesaro summability, the $\mathsf{CesaroConvergence}^{\uparrow}$ packet records ordinary convergence of the displayed Cesaro means as a BEDC finite-window certificate over regular real-sequence data. It consumes the finite averaging surface of chapter 1765, keeps the quotient-tail comparison discipline of chapter 1439 available as a sibling route, routes every transformed read through the regular rational interface of chapter 301, and reaches the terminal real-name seal only through chapter 38. The chapter names the carrier that a convergence consumer may inspect; it does not import a host convergence theorem or an ambient limit object.

Definition 1827.1 (Cesaro convergence carrier)**.** *A* $\mathsf{CesaroConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, M, A, L, R, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the source real-sequence row, exposed only through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows requested by the current convergence tolerance.*
2. *M is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row carrying the displayed prefix sums, unary denominator rows, dyadic division budgets, and finite Cesaro mean windows generated from S.*
3. *A is the finite averaging ledger, recording for each inspected tolerance the source windows, prefix rows, denominator rows, and dyadic error budgets used by the displayed Cesaro means.*
4. *L is the convergence-window ledger, containing the finite tail requests that classify the Cesaro mean rows as a regular real sequence.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from M, A, L before any completed real-name seal is read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source, Cesaro mean, averaging ledger, convergence-window ledger, and regular handoff rows have been displayed.*
7. *H stores componentwise* hsame *transports for the sequence source, Cesaro mean rows, averaging ledger, convergence ledger, regular handoff, real seal, continuation rows, provenance, and local naming.*

8. *T stores displayed* Cont *routes from source-window reads to Cesaro mean rows, from Cesaro mean rows to convergence-window classification, from that classification to R, and from R to E.*

9. *P is the* Pkg *provenance over* $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{StolzCesaro}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CesaroConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, A, L, R, E, H, T, P, N$. It has no coordinate for ordinary pointwise convergence of the source sequence, a host limit, quotient stream equality, a choice-selected subsequence, trichotomy, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CesaroConvergenceUp.TasteGate.CesaroConvergenceTasteGate single carrier alignment`

Theorem 1827.2 (Cesaro convergence NameCert obligations)**.** *For every accepted* $\mathsf{CesaroConvergence}^{\uparrow}$ *carrier $C = (S, M, A, L, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CesaroConvergence}^{\uparrow}}$ *surface consists exactly of the following six finite obligation rows:*

1. *carrier admission for the displayed source row, Hardy–Cesaro mean row, averaging ledger, convergence-window ledger, regular handoff, real seal, transport, continuation, provenance, and naming rows;*

2. *mean-window classification, requiring each accepted Cesaro mean read to be generated by M from the finite source windows and averaging ledger A;*

3. *stability, requiring the displayed* hsame *transports H and* Cont *routes T to preserve the same source, averaging, convergence-window, regular handoff, and real-seal rows;*

4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring every accepted convergence read to pass through R after M, A, L have been displayed;*

5. $\mathsf{Real}^{\uparrow}$ *seal factorization, requiring the seal row E to consume only the regular handoff obtained from the displayed Cesaro averaging route;*

6. *ledger non-escape, excluding ordinary pointwise convergence of S, host limits, quotient stream equality, choice-selected subsequences, and any detached real-completeness coordinate.*

Proof. Project the carrier of Definition 1827.1. Carrier admission is exactly the displayed packet $S, M, A, L, R, E, H, T, P, N$, so the certificate has no free coordinate for a source-sequence limit or an external convergence oracle.

Mean-window classification is supplied by the order of the rows. The source windows in S feed the $\mathsf{HardyCesaroMean}^{\uparrow}$ row M, and A records the finite averaging ledger that produced each displayed Cesaro mean from prefix sums, denominator rows, and dyadic budgets.

The convergence side is then forced through L and R. The ledger L records the finite tail requests witnessing regularity of the Cesaro mean sequence, while R is the only $\mathsf{RegSeqRat}^{\uparrow}$ handoff reachable from those requests before any real seal is visible.

Finally, E is downstream of R, and H, T, P, N merely transport, replay, source, and name the same route. Since no listed row stores ordinary pointwise convergence, a host limit, quotient stream equality, a selected subsequence, or ambient completeness, the six obligations exhaust the local NameCert surface. □

Theorem 1827.3 (Cesaro convergence RegSeqRat handoff)**.** *Every accepted* $\mathsf{CesaroConvergence}^{\uparrow}$ *read reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *through the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row and the displayed averaging ledger before the* $\mathsf{Real}^{\uparrow}$ *seal is available. More precisely, for every accepted packet* $C = (S, M, A, L, R, E, H, T, P, N)$*, a completion-facing consumer can reach* E *only along the route*

$$S \longmapsto M \longmapsto A \longmapsto L \longmapsto R \longmapsto E \longmapsto (H, T, P, N).$$

The sibling $\mathsf{StolzCesaro}^{\uparrow}$ *route may certify compatible quotient-tail estimates in* P*, but it does not bypass* M, A, L, R *for the Cesaro-convergence read.*

Proof. Begin with a completion-facing read of C. By Definition 1827.1, the read first exposes the source row S only as finite stream windows.

The next analytic surface is M. This is the $\mathsf{HardyCesaroMean}^{\uparrow}$ packet of chapter 1765; it supplies the finite prefix-sum, denominator, dyadic-division, and Cesaro mean rows. The averaging ledger A records the same rows as the local route consumed by the convergence classifier.

The tail classifier then reads L and only then reaches the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R. The sibling $\mathsf{StolzCesaro}^{\uparrow}$ edge named in P can provide quotient-tail compatibility data, but the displayed route for this carrier still passes through the Cesaro mean row and averaging ledger before R.

The $\mathsf{Real}^{\uparrow}$ seal E is downstream of R. The structural rows H, T, P, N preserve and name the route, so a consumer that reaches E has already consumed S, M, A, L, R and receives no ordinary pointwise convergence witness, host limit, quotient stream equality, selected subsequence, or ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{CesaroConvergence}^{\uparrow}$ Real-facing read factors through source sequence windows, a HardyCesaroMean row, a finite averaging ledger, a convergence-window ledger, RegSeqRat handoff, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: hardycesaromean, stolzcesaro, streamname, regseqrat, dyadicratcore, real: $\mathsf{CesaroConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source sequence row, Hardy–Cesaro mean rows, an averaging ledger, a convergence-window ledger, RegSeqRat handoff, Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CesaroConvergence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CesaroConvergence}^{\uparrow}$ packet is generated from a source real-sequence row, finite HardyCesaroMean averaging rows, an averaging ledger, convergence-window ledger, RegSeqRat handoff, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CesaroConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1827.1, Definition 1827.2, and Definition 1827.3 seed the finite Cesaro-convergence surface from regular real-sequence windows to the Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CesaroConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ordinary pointwise convergence, host limits, quotient stream equality, choice-selected subsequences, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, mean-window classification, stability, RegSeqRat handoff, RealUp seal factorization, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.CesaroConvergenceUp.

Chapter 1828

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocationWitness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocationWitness}^{\uparrow}$ packet records the finite location witness by which a regular Cauchy rational name may be consumed as a located real-name route. It sits over the dyadic tolerance ledger of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the sequence-level located Cauchy surface of chapter 340, and the filter-level located witness rows of chapter 1661. The certificate supplies the missing sequence-level location surface for later $\mathsf{Real}^{\uparrow}$ consumers without asserting Dedekind cuts, quotient streams, selected limits, countable choice, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1828.1 (Regular Cauchy location witness carrier)**.** *A* $\mathsf{RegularCauchyLocationWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, S, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular source row, namely the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name whose finite windows are being located.*
2. *S is the finite window schedule, namely the* $\mathsf{StreamName}^{\uparrow}$ *row that exposes exactly the inspected windows of R.*
3. *D is the dyadic tolerance ledger, recording requested radii, comparison budgets, endpoint approximants, and window tolerances.*
4. *W is the location-witness row, recording for each displayed dyadic cut request a finite left-or-right locatedness witness over the windows selected by S and the tolerances in D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, reachable only after R, S, D, W have been displayed.*
6. *H records componentwise* hsame *transport for the regular source, schedule, dyadic ledger, location witness, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a location request through R, S, D, W, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocationWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedCauchy}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocationWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, W, E, H, C, P, N$. The carrier contains no Dedekind cut, quotient stream equality, real trichotomy branch, countable-choice selector, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge checking datum.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocationWitnessUp.TasteGate.RegularCauchyLocationWitnessTasteGate single carrier alignment`

Theorem 1828.2 (Regular Cauchy location witness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocationWitness}^{\uparrow}$ *carrier*

$$L = (R, S, D, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocationWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular source row, finite window schedule, dyadic tolerance ledger, location-witness row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, transport, replay, provenance, and local name;*
2. *regular-source exposure, requiring every location request to read the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R only through the displayed schedule S;*
3. *dyadic-budget exactness, requiring each locatedness comparison to consume the radius, endpoint, and tolerance data stored in D before the witness row W is inspected;*
4. *witness routing, requiring W to record only finite left-or-right locatedness rows over the windows selected by S and the dyadic budget supplied by D;*
5. *seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *read to remain downstream of R, S, D, W;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name exactly the displayed route and no off-packet completeness or choice object.*

No local NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier from Definition 1828.1. The analytic rows are exactly the regular rational source, the finite schedule, the dyadic tolerance ledger, the location witness, and the terminal seal boundary.

Regular-source exposure is forced by the schedule row. A location request cannot inspect R directly; it must pass through the finite windows named by S.

The dyadic-budget and witness rows are then read together. The ledger D supplies the radius, endpoint, and tolerance data for the displayed cut request, and W records the finite locatedness witness over those same scheduled windows.

Finally, seal non-escape and structural stability are componentwise. The seal row E is reached only after R, S, D, W, while H, C, P, N preserve, replay, source, and name that same route. Since no excluded object is a coordinate of the packet, the local certificate has no place from which to export Dedekind cuts, quotient streams, trichotomy, choice, selected limits, ambient completeness, Lean verification, or bridge checking. □

Theorem 1828.3 (Regular Cauchy location witness regular readback)**.** *Let*

$$L = (R, S, D, W, E, H, C, P, N)$$

be an accepted RegularCauchyLocationWitness$^{\uparrow}$ *carrier. Every location read of L factors through the displayed regular-Cauchy route*

$$R \longmapsto S \longmapsto D \longmapsto W$$

before any Real$^{\uparrow}$ *seal or structural row is inspected.*

Proof. Begin with a location read. Carrier admission supplies R as the only regular rational source row, so no second stream representative or quotient stream can be selected.

The read then passes through S. This schedule determines the finite windows on which the regular source is inspected, and it is the only window source available to the packet.

Next the read consumes D and W. The dyadic ledger fixes the tolerance and endpoint budget, while the witness row records the finite locatedness decision over the scheduled windows.

Only after this route is displayed can the read reach E or the structural rows H, C, P, N. Thus every location read is a regular-Cauchy readback route rather than a host-level located real construction. □

Theorem 1828.4 (Regular Cauchy location witness Real seal non-escape)**.** *For every accepted* RegularCauchyLocationWitness$^{\uparrow}$ *carrier $L = (R, S, D, W, E, H, C, P, N)$, the exported* Real$^{\uparrow}$ *seal E is read only as the terminal boundary of the displayed finite route through R, S, D, W, H, C, P, N. It exports no Dedekind cuts, quotient stream equality, real trichotomy, countable-choice selector, selected limit, ambient Real completeness theorem, Lean verification, or bridge checking datum.*

Proof. Project the seal row from the carrier. The row E is listed after the regular source, schedule, dyadic ledger, and witness rows, so a seal-facing consumer has no direct entry to E.

Use Definition 1828.3. Any seal-facing location route first reads R, S, D, W, and those rows exhaust the finite analytic data available to the packet.

The structural rows preserve only this route. Transport H, replay C, provenance P, and local naming N do not add new analytic coordinates; they keep the same displayed packet available under componentwise comparison and continuation.

The excluded principles are absent from the carrier inventory and from the provenance row. Therefore the terminal seal remains a finite located regular-Cauchy boundary, not a proof of Dedekind completeness, quotient equality, trichotomy, choice, selected limits, Lean verification, or bridge checking. □

Closure status for NameCert$_{\text{RegularCauchyLocationWitness}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyLocationWitness}^{\uparrow}}$, seed)

Scope closed: Definition 1828.1, Definition 1828.2, Definition 1828.3, and Definition 1828.4 seed the finite location-witness route from regular rational source rows through StreamName windows, DyadicRatCore tolerances, displayed locatedness witnesses, and the terminal Real seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyLocationWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyLocationWitnessUp.TasteGate.RegularCauchyLocationWitnessTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient streams, real trichotomy, countable choice, selected limits, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-source exposure, dyadic-budget exactness, witness routing, regular readback, Real-seal non-escape, and structural stability for BEDC.Derived.RegularCauchyLocationWitnessUp.

Constructive story: A RegularCauchyLocationWitness$^{\uparrow}$ packet is generated from a RegSeqRat regular source row, finite StreamName windows, a DyadicRatCore tolerance ledger, displayed locatedness witnesses, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1829

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$

Following Mac Lane's monad multiplication terminology together with Bishop–Bridges and Kelley completion language, the $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ packet records the finite BEDC join route that flattens a completion of completion into the displayed completion boundary. It sits after the monad surface of chapter 378, the idempotence row of **??**, and the associativity ledger of chapter 1635; its public L10 face is read through chapter 126, chapter 301, chapter 291, and chapter 38. The packet names monad multiplication as finite carrier data; it is not a quotient completion, selected representative, host equality comparison, countable-choice construction, ambient completeness principle, or completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1829.1 (Cauchy completion multiplication carrier)**.** *A* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, O, I, A, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *transformer row that records the completion endofunctor, unit, and bind surface consumed by the join.*
2. *O is the outer completion row, presenting the completion-of-completion input before flattening.*
3. *I is the* $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$ *row that identifies a completed completion boundary with the displayed completion boundary through finite certificate data.*
4. *A is the* $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$ *ledger that keeps nested bind and flattening routes in the same displayed order.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed only after the nested completion route has passed through M, O, I, A.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same flattened stream windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the flattened readback and terminal seal.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the displayed StreamName, RegSeqRat, and DyadicRatCore faces have been fixed.*
9. *H records componentwise* hsame *transports for the monad, outer-completion, idempotence, associativity, window, readback, tolerance, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an outer completion request through the flattening route and then through S, R, D, E.*
11. *P is the* Pkg *provenance over* CauchyCompletionMultiplication$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionIde CauchyCompletionAssociativity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionMultiplication}^{\uparrow}}$ *row.*

The classifier of accepted packets compares only these twelve displayed rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, ambient Real completeness, completeness of Real$^{\uparrow}$*, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.TasteGate.CauchyCompletionMultiplicationTasteGate single carrier alignment`

Theorem 1829.2 (Cauchy completion multiplication NameCert obligations)**.** *For every accepted* CauchyCompletionMultiplication$^{\uparrow}$ *carrier $K = (M, O, I, A, S, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionI}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed monad transformer, outer completion, idempotence, associativity, stream-window, readback, tolerance, real-seal, transport, replay, provenance, and local naming rows;*
2. *outer-to-inner flattening, requiring every completion-of-completion request to pass through M, O, I, A before any public L10 row is exposed;*
3. *shared window, readback, and seal routing, requiring S, R, D, E to be read as one flattened route rather than as detached representatives;*
4. *componentwise transport, requiring H to preserve the twelve displayed coordinates and no additional completion witness;*
5. *replay, requiring C to reproduce the same route from outer completion request through M, O, I, A, S, R, D, E;*
6. *provenance, requiring P to name only the cited monad, idempotence, associativity, stream, regular rational, dyadic, Real, and BEDC structural sources;*
7. *ledger nonescape, requiring N and the classifier to export no quotient completion, selected representative, countable choice, host equality, ambient Real completeness,* Real$^{\uparrow}$ *completeness theorem, Lean verification claim, or bridge credential.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1829.1. The tuple lists exactly the monad transformer, outer completion, idempotence, associativity ledger, public L10 rows, transport, replay, provenance, and local name.

The outer-to-inner flattening obligation is forced by the tuple order. A request addressed to the outer completion first reads M and O, then uses I to display the boundary as a single completion row, and uses A to keep the nested bind route coherent.

Shared routing is the projection to S, R, D, E. These rows are downstream of M, O, I, A, so the packet cannot expose a stream window, regular rational readback, dyadic tolerance, or Real seal detached from the displayed flattening route.

Componentwise transport and replay are read through H and C. The transport row identifies accepted packets coordinate by coordinate, while the continuation row reproduces the same request path from the outer completion through the public L10 face.

Finally, provenance and nonescape are exhausted by P, N and by the classifier. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge credentials, none of these can be exported by a local obligation. □

Theorem 1829.3 (Cauchy completion multiplication flattening route)**.** *Every accepted* CauchyCompletionMultiplica *packet flattens the outer completion through the route*

$$M \longmapsto O \longmapsto I \longmapsto A$$

before exposing the public L10 face

$$S \longmapsto R \longmapsto D \longmapsto E.$$

Thus a completion-of-completion consumer can read the StreamName window, RegSeqRat readback, DyadicRatCore tolerance, and Real seal only after the monad transformer, outer completion row, idempotence row, and associativity ledger have been displayed.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplication flattening route`

Proof. Start with an accepted packet $K = (M, O, I, A, S, R, D, E, H, C, P, N)$. Projection gives the monad transformer M and the outer completion row O before any public L10 coordinate is available.

The idempotence row I supplies the displayed route from the completed completion boundary back to the single completion boundary. The associativity ledger A records that nested bind and flattening reads follow the same route.

Only after M, O, I, A have been read does the carrier expose S, R, D, E. The continuation row C replays this order, and componentwise transport by H preserves it for transported packets.

Consequently every public read from an accepted packet factors through the stated flattening route before the StreamName, RegSeqRat, DyadicRatCore, and Real-seal rows can be consumed. □

Theorem 1829.4 (Cauchy completion multiplication Real seal nonescape)**.** *Let $K = (M, O, I, A, S, R, D, E, H, C,$ be an accepted* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of K receives only the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N after the flattening route through M, O, I, A.*

In particular, the consumer receives no quotient completion, selected representative, countable-choice selector, host equality comparison, ambient Real completeness datum, or completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationRealSealNonescape`
↪ variant `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationCarrier real seal nonescape`

Proof. Project the accepted packet. A $\mathsf{Real}^{\uparrow}$-facing read reaches the terminal seal only through the flattened public face S, R, D, E.

The structural rows H, C, P, N do not add data to that face. They transport it componentwise, replay the displayed route, record the cited sources, and name the local certificate.

The excluded objects are absent from the carrier and from the classifier. There is no coordinate from which a quotient completion, selected representative, countable-choice selector, host equality comparison, ambient Real completeness datum, or $\mathsf{Real}^{\uparrow}$ completeness theorem could be read.

Therefore the Real-facing consumer receives exactly the displayed rows listed in the statement, downstream of the flattening route through M, O, I, A. □

Theorem 1829.5 (Cauchy completion multiplication obligation closure package)**.** *For every accepted* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *carrier* $K = (M, O, I, A, S, R, D, E, H, C, P, N)$*, the obligation surface named by* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMultiplication}^{\uparrow}}$ *is the package obtained by projecting exactly the rows* $M, O, I, A, S, R, D, E, H, C, P, N$*, applying Definition 1829.2 for carrier admission and local obligation exhaustion, applying Definition 1829.3 for the outer-to-inner route, and applying Definition 1829.4 for the terminal Real-facing boundary.*

The package has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, ambient Real completeness, a completeness theorem for $\mathsf{Real}^{\uparrow}$*, Lean verification status, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationCarrier obligation closure package`

Proof. Project the accepted carrier. The rows M, O, I, A provide the monad transformer, outer-completion, idempotence, and associativity data; the rows S, R, D, E provide the displayed StreamName, RegSeqRat, DyadicRatCore, and Real-seal face; and H, C, P, N provide transport, replay, provenance, and local naming.

Apply Definition 1829.2. It lists the local carrier admission, flattening, shared routing, transport, replay, provenance, and ledger nonescape obligations and says that no local obligation is stored outside those rows.

Apply Definition 1829.3 to bind the order M, O, I, A before S, R, D, E. Apply Definition 1829.4 to show that the terminal Real-facing consumer receives only the displayed L10 face and structural rows.

The excluded objects are absent from the carrier and from all three cited obligation routes, so the projected package is exactly the local obligation surface. □

Theorem 1829.6 (Cauchy completion multiplication outer-inner replay)**.** *Let* $K = (M, O, I, A, S, R, D, E, H, C, P,$ *be an accepted* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *carrier. Every displayed* Cont *replay in* C *factors an outer-completion request through*

$$M \longmapsto O \longmapsto I \longmapsto A$$

before it reads

$$S \longmapsto R \longmapsto D \longmapsto E.$$

The replay then returns through H, C, P, N *as structural transport, displayed continuation, provenance, and local naming, and it never exposes a detached completion-of-completion representative.*

Proof. Project K and read the replay row C together with the transport row H. The carrier order gives the monad transformer M and outer completion row O before any public L10 row can be inspected.

Use Definition 1829.3. It forces the idempotence row I and associativity ledger A to mediate the route from the outer completion to the displayed completion boundary before the route reaches S, R, D, E.

Use Definition 1829.2 for replay as a local NameCert obligation. The replay obligation names the same route and no hidden representative.

Finally, Definition 1829.4 prevents the terminal read from exporting quotient completion equality, selected representatives, countable choice, host equality, or ambient completeness data. Thus every replay is the stated outer-inner route. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationCarrier outer inner replay`

Theorem 1829.7 (Cauchy completion multiplication provenance exhaustion)**.** *For every accepted* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ *carrier* $K = (M, O, I, A, S, R, D, E, H, C, P, N)$*, the provenance row* P *and local naming row* N *source only* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionAssociativity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*. Any accepted consumer of* K *that reaches provenance or local naming has already passed through the carrier admission, flattening, replay, and Real-seal nonescape obligations.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMultiplicationUp.CauchyCompletionMultiplicationProvenanceExhaustio`

Proof. Project the carrier to P and N. By Definition 1829.1, P names only the displayed monad, idempotence, associativity, stream, regular rational, dyadic, Real, and BEDC structural sources, while N names the local certificate row.

Apply Definition 1829.2. Its provenance and ledger-nonescape clauses require the local surface to source only these rows and to export no additional completion witness.

Apply Definition 1829.3 and Definition 1829.4. The first forces provenance-facing reads to pass through the displayed outer-to-inner route, and the second prevents the terminal seal from adding Real-completeness or selected-representative data.

Therefore P, N exhaust the cited sources and no provenance-facing consumer receives data outside the listed carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMultiplication}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMultiplication}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1829.1, Definition 1829.2, Definition 1829.3, Definition 1829.4, Definition 1829.5, Definition 1829.6, and Definition 1829.7 enumerate the displayed carrier admission, outer-to-inner flattening, componentwise transport, replay, provenance, and Real-seal nonescape obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMultiplication}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits or representatives, host equality, countable choice, ambient completeness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, outer-to-inner flattening, componentwise transport, replay, provenance, and Real-seal nonescape obligations for BEDC.Derived.CauchyCompletionMultiplicationUp.

Constructive story: A $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$ packet is generated from a CauchyCompletionMonad transformer row, an outer completion row, idempotence and associativity rows, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1830

A Concrete Naming Certificate for $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ packet records uniformly Cauchy real-valued function data as a finite BEDC route rather than as host function equality or a completed function space. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the uniform Cauchy modulus surface of chapter 1741, and the terminal real-name seal of chapter 38. The packet gives later uniform-limit and compact-continuity consumers one named route for real-valued function families before any quotient stream, selected pointwise limit, countable-choice schedule, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 1830.1 (Real-function uniform Cauchy carrier)**.** *A* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, S, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the real-valued function source row. It stores only the displayed argument window, the displayed function-value row, and the local classifier for the finite function observation.*
2. *S is the finite source-window ledger. For each accepted tolerance request it records the finite argument windows and the paired value-observation rows that will be compared.*
3. *M is the* $\mathsf{UniformCauchyModulus}^{\uparrow}$ *row. It assigns one displayed modulus request to all source-window rows in S and never selects a host-level pointwise limit.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested tolerance, the smaller dyadic tolerance supplied by M, and the finite descent route used by the paired value comparisons.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations exposed by S, M, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, uniform modulus row, dyadic tolerance ledger, and regular rational readback have been exposed.*
7. *H records componentwise* hsame *transport for the function source row, source-window ledger, modulus row, dyadic ledger, readback row, seal row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a uniform Cauchy request through $A, S, M, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{UniformCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealFunctionUniformCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $A, S, M, D, R, E, H, C, P, N$. The carrier has no coordinate for host function equality, quotient streams, selected pointwise limits, countable-choice selectors, ambient real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RealFunctionUniformCauchyUp.TasteGate.RealFunctionUniformCauchyTasteGate single carrier alignment`

Theorem 1830.2 (Real-function uniform Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *carrier*

$$F = (A, S, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealFunctionUniformCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real-valued function row, finite source-window ledger, uniform Cauchy modulus row, dyadic tolerance ledger, regular rational readback, terminal real seal, transport, replay, provenance, and local name;*
2. *finite-window stability, requiring every comparison of function values to be read from the windows listed in S;*
3. *uniform-modulus transport, requiring the same modulus row M to serve all accepted source-window rows for the requested tolerance;*
4. *dyadic tolerance descent, requiring the comparison tolerance used by the value rows to be obtained from the dyadic ledger D;*
5. *regular rational readback, requiring every numeric observation consumed by the packet to pass through R before the terminal seal is inspected;*
6. *Real-seal non-escape, requiring E to remain downstream of A, S, M, D, R and the structural rows;*
7. *provenance exhaustion, requiring P to source only the listed rows and to export no host function equality, quotient stream, selected pointwise limit, countable-choice schedule, or ambient real completeness principle;*
8. *local naming, requiring N to name exactly this finite surface and no external function-space object.*

No local NameCert obligation is stored outside these rows.

Theorem 1830.3 (Real-function uniform Cauchy Real-completion handoff)**.** *Let $F = (A, S, M, D, R, E, H, C, P, N$ be an accepted* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read of F factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as exposed by the source windows S, the readback row R, the tolerance ledger D, and the terminal seal E. In particular, a completion-facing consumer can inspect real-valued function data only

through the displayed finite windows, regular rational observations, dyadic tolerance descent, uniform modulus transport, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It cannot read host function equality, quotient streams, countable choice, selected pointwise limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Closure status for $\mathsf{NameCert}_{\mathsf{RealFunctionUniformCauchy}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealFunctionUniformCauchy}\uparrow}$, seed)

Scope closed: Definition 1830.1, Definition 1830.2, and Definition 1830.3 seed the finite route from uniform Cauchy real-valued function data to StreamName, RegSeqRat, DyadicRatCore, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealFunctionUniformCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient streams, countable choice, selected pointwise limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window stability, uniform-modulus transport, dyadic tolerance descent, regular rational readback, Real-seal non-escape, provenance exhaustion, and local naming for BEDC.Derived.RealFunctionUniformCauchyUp.

Constructive story: A $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ packet is generated from one real-valued function source row, finite source-window rows, one UniformCauchyModulus row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1831

A Concrete Naming Certificate for $\mathsf{DyadicIntervalRefinement}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Moore–Kearfott–Cloud interval analysis, the $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ packet isolates one finite dyadic subdivision step before any infinite nested route is read. It sits over the dyadic endpoint core of chapter 291, the interval surface of chapter 306, the midpoint selector of chapter 1712, the finite nesting discipline of chapter 328, the Cauchy handoff of chapter 1811, the finite schedule interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a one-step finite-means carrier for L10 real and completion work; it is not an open-cover compactness theorem, a selected branch, a quotient stream equality, a trichotomy principle, countable choice, ambient real completeness, a Lean theorem, or a bridge check.

Definition 1831.1 (Dyadic interval refinement carrier)**.** *A* $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, J_0, J_1, E, R, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the parent* $\mathsf{DyadicInterval}^{\uparrow}$ *row. It exposes the ordered dyadic endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and the parent containment ledger read by later interval consumers.*

2. *J_0 and J_1 are the two child* $\mathsf{DyadicInterval}^{\uparrow}$ *rows produced by the displayed midpoint row. Their outer endpoints are reused from I, and their shared endpoint is the midpoint supplied by* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$*.*

3. *E is the endpoint-reuse ledger. It records that the lower endpoint of J_0 is the lower endpoint of I, the upper endpoint of J_1 is the upper endpoint of I, and the upper endpoint of J_0 is the lower endpoint of J_1.*

4. *R is the refinement-containment ledger. It records both child intervals as dyadic subintervals of I, and records that any retained child row is a finite step admissible for the* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *surface.*

5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It names only the parent endpoint observations, child endpoint observations, midpoint observation, and retained-child flag opened by this one refinement step.*

6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It repacks the scheduled dyadic observations from S as regular rational data for downstream Cauchy consumers.*

7. *H records componentwise* hsame *transport for the parent interval, child intervals, endpoint-reuse ledger, refinement-containment ledger, stream window, regular rational readback, replay rows, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a refinement request through $I, J_0, J_1, E, R, S, Q, H, P, N$ without adding a branch selector, quotient stream, or completed real point.*
9. *P is the* Pkg *provenance over* DyadicIntervalRefinement$^\uparrow$, DyadicRatCore$^\uparrow$, DyadicInterval$^\uparrow$, LocatedIntervalMidpoi NestedDyadicInterval$^\uparrow$, NestedDyadicCauchy$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DyadicIntervalRefinement}^\uparrow}$ *row.*

The classifier compares accepted packets only through these eleven displayed rows. The carrier has no coordinate for open-cover compactness, selected infinite branches, quotient stream equality, trichotomy, countable choice, ambient Real$^\uparrow$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalRefinementUp.DyadicIntervalRefinementTasteGate single carrier alignment`

Theorem 1831.2 (Dyadic interval refinement NameCert obligations)**.** *For every accepted* DyadicIntervalRefinement *carrier*

$$K = (I, J_0, J_1, E, R, S, Q, H, C, P, N),$$

the local NameCert$_{\text{DyadicIntervalRefinement}^\uparrow}$ *surface consists exactly of carrier admission, parent dyadic interval exposure, two child dyadic interval exposures, endpoint-reuse exactness, dyadic containment and retained-step exactness, finite StreamName window exposure, RegSeqRat readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. No local NameCert obligation is stored outside those rows.*

Proof. Project the carrier of Definition 1831.1. The projection exposes one parent interval row, two child interval rows, one endpoint-reuse row, one refinement-containment row, one stream window, one regular rational readback, and the structural rows.

The parent and child exposure obligations are forced by I, J_0, J_1. Endpoint-reuse exactness is forced by E, and containment exactness is forced by R; a consumer can inspect no other interval or containment object through the classifier.

The downstream read rows are S and Q. The StreamName row lists exactly the finite observations opened by this subdivision step, while the RegSeqRat row repacks only those observations as regular rational data.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since the carrier has no extra coordinate, the listed rows exhaust the NameCert surface. □

Theorem 1831.3 (Dyadic interval refinement window handoff)**.** *Let $K = (I, J_0, J_1, E, R, S, Q, H, C, P, N)$ be an accepted* DyadicIntervalRefinement$^\uparrow$ *carrier. Every* NestedDyadicInterval$^\uparrow$, StreamName$^\uparrow$, *or* RegSeqRat$^\uparrow$ *consumer of the refinement factors through the finite route*

$$I, J_0, J_1 \longmapsto E \longmapsto R \longmapsto S \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer receives a parent interval, two displayed child intervals, endpoint reuse, finite containment, the scheduled observation window, and regular rational readback, but receives no infinite branch selector or quotient stream equality.

Proof. Start with the interval rows. The parent row I and child rows J_0, J_1 provide the only endpoint data available to any nested-window consumer.

Use E to identify the reused endpoints and shared midpoint. Then use R to read the finite containment relation that makes either child an admissible retained step for a nested dyadic interval prefix.

The schedule row S fixes which endpoint, midpoint, and retained-child observations may be opened. The readback row Q then repacks those finite observations for regular rational consumers.

The structural rows H, C, P, N only transport, replay, source, and name this route. They do not add an infinite branch, a quotient stream, or an external real point, so every listed consumer factors through the displayed finite handoff. □

Theorem 1831.4 (Dyadic interval refinement Real boundary)**.** *For every accepted* $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ *carrier* $K = (I, J_0, J_1, E, R, S, Q, H, C, P, N)$*, every* $\mathsf{NestedDyadicCauchy}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing read must pass through the finite StreamName window* S *and the RegSeqRat readback* Q *before any real-completion consumer can inspect the row. The refinement exports a finite dyadic subdivision boundary, not open-cover compactness, a selected infinite branch, quotient stream equality, trichotomy, countable choice, ambient real completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier. The only schedule coordinate is S, and the only regular rational readback coordinate is Q.

Apply Definition 1831.3. A NestedDyadicCauchy or Real-facing consumer must first read the displayed parent and child intervals, endpoint reuse, containment, and scheduled observations before the readback row is available.

The readback row Q is therefore the boundary through which real-completion consumers see the refinement. Because the carrier contains no open-cover, selected-branch, quotient, trichotomy, choice, completeness, Lean, or bridge coordinate, none of those objects can be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ packet is generated from one parent dyadic interval, two child dyadic intervals, endpoint reuse, dyadic containment and retained-step rows, finite StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1831.1, Definition 1831.2, Definition 1831.3, and Definition 1831.4 seed the finite one-step dyadic refinement route from interval subdivision to StreamName, RegSeqRat, NestedDyadicCauchy, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, quotient stream equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-reuse exactness, containment and retained-step exactness, finite window exposure, RegSeqRat readback, Real-boundary nonescape, provenance exhaustion, and local naming for BEDC.Derived.DyadicIntervalRefinementUp.

Chapter 1832

A Concrete Naming Certificate for $\mathsf{RegularCauchyMinimum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMinimum}^{\uparrow}$ packet records the finite Bishop-style route for the lower member of two regular Cauchy real names without importing a host minimum operation or an ambient order-completeness principle. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located comparison modulus of chapter 1627, and the Archimedean comparison budget of chapter 430. The certificate gives downstream real-order and completion-facing chapters one finite route for binary lower-bound formation before quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness is available.

Definition 1832.1 (Regular Cauchy minimum carrier)**.** *A* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, T, L, D, A, B, W, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* $\mathsf{Real}^{\uparrow}$ *rows presented by regular Cauchy names. Each row exposes only its finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, dyadic tolerance ledger, and terminal real seal.*

2. *T is the dyadic comparison tolerance row. It stores the requested output tolerance, the comparison tolerance used to separate the two names when a side is selected, and the smaller dyadic budget consumed by the exported lower-bound row.*

3. *L is the* $\mathsf{LocatednessModulus}^{\uparrow}$ *row. It records the finite located comparison request for the two displayed regular Cauchy names and never supplies a host trichotomy or a global order decision.*

4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for the two readbacks, the comparison tolerance, and the lower dyadic approximation chosen for the output name.*

5. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *budget row, used only to align dyadic comparison scale, regular Cauchy tail scale, and the requested output tolerance.*

6. *B is the finite branch ledger. It records whether the displayed located comparison routes the lower-bound output through the left name, through the right name, or through the overlap tolerance when the two windows are indistinguishable at the requested scale.*

7. *W is the exported regular Cauchy output row. Every window of W is copied from X or Y under the branch ledger B and the tolerance rows T, L, D, A.*

8. *H records componentwise* hsame *transport for the two input names, tolerance, locatedness, dyadic arithmetic, Archimedean budget, branch, output, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a lower-bound request through $X, Y, T, L, D, A, B, W, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMinimum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the tuple $X, Y, T, L, D, A, B, W, H, C, P, N$. The carrier has no coordinate for a host minimum function, trichotomy, quotient stream equality, selected limits, countable-choice schedules, ambient real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyMinimumUp.RegularCauchyMinimumTasteGate single carrier alignment`

Theorem 1832.2 (Regular Cauchy minimum NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carrier*

$$M = (X, Y, T, L, D, A, B, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMinimum}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, tolerance row, locatedness row, dyadic arithmetic ledger, Archimedean budget, branch ledger, output row, transport, replay, provenance, and local name;*

2. *input regularity, requiring every exported output window to be copied from the displayed regular Cauchy windows of X or Y at the scale fixed by T, D, A;*

3. *located comparison, requiring the side ledger B to be read only after the locatedness row L and the dyadic tolerance row T have been displayed;*

4. *left-bound exactness, requiring a left-routed branch to expose an output window no larger than the displayed window of X at the requested tolerance;*

5. *right-bound exactness, requiring a right-routed branch to expose an output window no larger than the displayed window of Y at the requested tolerance;*

6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, L, D, A, B, W, C, P, N$ to preserve the accepted lower-bound route;*

7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no host minimum, trichotomy, quotient stream equality, selected limit, countable choice, ambient Real completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1832.1. Its displayed rows are exactly the two regular Cauchy real inputs, the dyadic tolerance row, locatedness row, dyadic arithmetic ledger, Archimedean budget, branch ledger, output row, and structural rows.

Input regularity follows because W has no independent numeric coordinate. Each exported window is copied from X or Y after the tolerance, locatedness, dyadic, and Archimedean rows fix the scale.

The left-bound and right-bound rows are then read from B. A left-routed branch consumes the left input window and the overlap tolerance; a right-routed branch consumes the right input window and the same displayed tolerance surface.

Classifier stability and non-escape are ledger exhaustion. The only permitted transports and replays are H and C, and the provenance and naming rows P, N cite no excluded host or quotient object. □

Theorem 1832.3 (Regular Cauchy minimum left bound)**.** *Let $M = (X, Y, T, L, D, A, B, W, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carrier. Every finite window exported by W is a displayed lower-bound window for the left input row X at the tolerance recorded in T. The witness is read entirely from the branch ledger B, the locatedness row L, the dyadic arithmetic ledger D, the Archimedean budget A, the* hsame *transport H, and the replay row C.*

Proof. Project the displayed output window from W and inspect the corresponding branch row in B.

In a left-routed branch, the output window is copied from X, so the claimed left-bound row is the reflexive displayed comparison transported by H and replayed by C.

In a right-routed or overlap branch, the locatedness row L supplies the finite comparison certificate at the dyadic scale fixed by T, D, A. The output window is copied from Y only under that certificate, so the same ledger gives the displayed lower-bound comparison against X.

No other comparison source exists in the carrier. The proof is therefore exhausted by B, L, T, D, A, H, C. □

Lean4 ✓ `BEDC.Derived.RegularCauchyMinimumUp.RegularCauchyMinimumLeftBound`

Theorem 1832.4 (Regular Cauchy minimum right bound)**.** *Let $M = (X, Y, T, L, D, A, B, W, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carrier. Every finite window exported by W is a displayed lower-bound window for the right input row Y at the tolerance recorded in T. The witness is read entirely from the branch ledger B, the locatedness row L, the dyadic arithmetic ledger D, the Archimedean budget A, the* hsame *transport H, and the replay row C.*

Proof. Project the displayed output window from W and inspect its branch row in B.

In a right-routed branch, the output window is copied from Y, so the claimed right-bound row is the reflexive displayed comparison transported by H and replayed by C.

In a left-routed or overlap branch, the locatedness row L supplies the finite comparison certificate at the dyadic scale fixed by T, D, A. The output window is copied from X only under that certificate, so the same ledger gives the displayed lower-bound comparison against Y.

The carrier contains no independent host order proof. Thus the right-bound witness is exactly the local projection of B, L, T, D, A, H, C. □

Theorem 1832.5 (Regular Cauchy minimum classifier stability)**.** *If two accepted* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carriers*

$$M = (X, Y, T, L, D, A, B, W, H, C, P, N)$$

and

$$M' = (X', Y', T', L', D', A', B', W', H', C', P', N')$$

are componentwise hsame *on the two input names, tolerance, locatedness, dyadic arithmetic, Archimedean budget, branch ledger, output row, replay, provenance, and local naming row, then their classifiers agree. In particular, every accepted lower-bound window in W is transported to the corresponding accepted lower-bound window in W' without adding a host minimum or quotient-stream equality.*

Proof. Unfold the classifier described in Definition 1832.1. It reads only $X, Y, T, L, D, A, B, W, H, C, P, N$.

The assumed componentwise hsame rows identify each coordinate with its primed counterpart. Transport along those rows preserves the side ledger, the copied output window, and the displayed dyadic comparison scale.

Since neither classifier has a hidden coordinate, there is no remaining source from which disagreement could arise. The transported packet is accepted exactly when the original packet is accepted. □

Theorem 1832.6 (Regular Cauchy minimum non-escape)**.** *Every accepted* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *carrier exports only the finite lower-bound route described by Definition 1832.1. No consumer of the packet can recover a host minimum function, trichotomy of real names, quotient stream equality, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean verification, or bridge checking credential.*

Proof. Read the carrier coordinate by coordinate. Numeric data are confined to the two input rows, the tolerance row, the locatedness row, the dyadic ledger, the Archimedean budget, the branch ledger, and the output row.

Transport and replay are confined to H and C. Provenance and local naming are confined to P and N, and those rows cite only the finite BEDC names listed in the carrier.

The excluded principles have no coordinate, no provenance entry, and no replay edge. Therefore a consumer can inspect the displayed lower-bound certificate but cannot escape to any host or quotient object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMinimum}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1832.1, Definition 1832.2, Definition 1832.3, Definition 1832.4, Definition 1832.5, and Definition 1832.6 seed a finite lower-bound route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host minimum, trichotomy, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, input regularity, located comparison, left-bound exactness, right-bound exactness, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyMinimumUp.

Constructive story: A $\mathsf{RegularCauchyMinimum}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a dyadic comparison tolerance, a LocatednessModulus row, a DyadicRatCore arithmetic ledger, an ArchimedeanReal budget, a branch ledger, an output regular Cauchy row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1833

A Concrete Naming Certificate for $\mathsf{RegularCauchyMaximum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyMaximum}^{\uparrow}$ packet records the constructive upper member of two regular rational Cauchy real names by reducing the public route to already displayed regular Cauchy arithmetic. It reads the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic surface of chapter 291, the absolute-value lift of chapter 1499, the addition surface of chapter 417, the subtraction surface of chapter 429, the sibling lower-bound packet of chapter 1832, and the terminal real-name seal of chapter 38. The certificate gives downstream $\mathsf{Real}^{\uparrow}$ consumers a finite maximum route without assuming quotient streams, selected limits, trichotomy, countable choice, or ambient completeness.

Definition 1833.1 (Regular Cauchy maximum carrier)**.** *A* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, W_X, W_Y, D, S, A, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* $\mathsf{Real}^{\uparrow}$ *rows presented by regular Cauchy names; each exposes only its finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, dyadic tolerance ledger, and terminal real seal.*

2. *W_X and W_Y are aligned finite observation windows for the two names at the requested output tolerance.*

3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subtraction ledger for the displayed difference $X - Y$ read through W_X and W_Y.*

4. *S is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row repacking the pointwise dyadic subtraction route as a regular Cauchy difference name.*

5. *A is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *row applied to the displayed difference route S.*

6. *U is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row adding the two source names and the absolute-difference row on the shared observation scale.*

7. *R is the final regular Cauchy readback row for the half-sum maximum route, with the dyadic division-by-two ledger displayed inside the same finite tolerance budget.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R has been exposed.*

9. *H records componentwise* hsame *transport for the two inputs, windows, dyadic subtraction, difference row, absolute-difference row, sum row, half-sum readback, real seal, provenance, and local name.*

10. *C records displayed* Cont *replay from a maximum request through $X, Y, W_X, W_Y, D, S, A, U, R, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMaximum}^{\uparrow}$, $\mathsf{RegularCauchyMinimum}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMaximum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W_X, W_Y, D, S, A, U, R, E, H, C, P, N$. The carrier has no coordinate for a host maximum function, trichotomy, quotient stream equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMaximumUp.RegularCauchyMaximumTasteGate single carrier alignment`

Theorem 1833.2 (Regular Cauchy maximum NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *carrier*

$$M = (X, Y, W_X, W_Y, D, S, A, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMaximum}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, aligned stream windows, dyadic subtraction ledger, difference row, absolute-difference row, sum row, half-sum readback, real seal, transport, replay, provenance, and local name;*

2. *aligned-window exposure, requiring every output request to display matching* $\mathsf{StreamName}^{\uparrow}$ *windows for X and Y before any dyadic subtraction, absolute value, sum, or seal row is consumed;*

3. *arithmetic route exactness, requiring D, S, A, U, R to be read in the order subtraction, regular Cauchy difference, regular Cauchy absolute value, regular Cauchy sum, and dyadic half-sum readback;*

4. *classifier stability, requiring componentwise* hsame *transport of $X, Y, W_X, W_Y, D, S, A, U, R, E, C, P, N$ to preserve the accepted maximum route;*

5. *non-escape, requiring H, C, P, N to transport, replay, source, and name no host maximum, trichotomy, quotient stream equality, selected limit, countable choice, ambient Real completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1833.1. Its displayed rows are exactly the two regular Cauchy real inputs, aligned stream windows, dyadic subtraction ledger, regular Cauchy difference row, absolute-difference row, sum row, half-sum readback, real seal, and structural rows.

Aligned-window exposure follows because D has no independent source coordinate. The subtraction ledger can read endpoints only through W_X and W_Y, so the later rows S, A, U, R, E remain downstream of the displayed finite windows.

Arithmetic route exactness is ledger order. The difference row S consumes D, the absolute-difference row A consumes S, the sum row U consumes X, Y, A, and the readback row R records the dyadic division by two before E seals the result as a real name.

Classifier stability and non-escape are coordinate exhaustion. The only permitted transports and replays are H and C, and the provenance and naming rows P, N cite no excluded host or quotient object. □

Theorem 1833.3 (Regular Cauchy maximum abs-sum route)**.** *Let*

$$M = (X, Y, W_X, W_Y, D, S, A, U, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the maximum row reaches the seal E only through the constructive route*

aligned windows for X and Y,
dyadic subtraction $X - Y$,
regular Cauchy difference S,
absolute difference A,
regular Cauchy sum of X, Y, A,
dyadic half-sum readback R,
final real seal E.

Equivalently, the packet presents the maximum of X and Y by the displayed finite route for $(X + Y + |X - Y|)/2$, and it exposes no quotient stream equality, selected limit, trichotomy, countable choice, host maximum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a real-facing read of the carrier. By the obligation rows of Definition 1833.2, the read cannot reach the seal E before it has consumed the half-sum readback row R.

The row R is downstream of the sum row U. The sum row consumes the two source names together with the absolute-difference row A, and the row A is downstream of the regular Cauchy difference row S formed from the dyadic subtraction ledger D.

The ledger D is read only after the aligned windows W_X and W_Y are displayed. Thus the route is the listed sequence of aligned windows, dyadic subtraction, regular Cauchy difference, absolute difference, sum, half-sum readback, and real seal.

No excluded principle has a coordinate in the carrier. The provenance and replay rows merely name and transport this finite route, so the consumer receives the abs-sum maximum certificate and nothing outside it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMaximum}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1833.1, Definition 1833.2, and Definition 1833.3 seed a finite upper-member route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMaximum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host maximum, trichotomy, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, aligned-window exposure, arithmetic route exactness, classifier stability, non-escape, and the abs-sum RealUp handoff for BEDC.Derived.RegularCauchyMaximumUp.

Constructive story: A $\mathsf{RegularCauchyMaximum}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, aligned StreamName windows, a DyadicRatCore subtraction ledger, RegularCauchyDifference, RegularCauchyAbs, and RegularCauchySum rows, a dyadic half-sum readback, a RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1834

A Concrete Naming Certificate for $\mathsf{ImplicitFunction}^{\uparrow}$

Following the classical implicit function theorem tradition of Dieudonne and Rudin, the $\mathsf{ImplicitFunction}^{\uparrow}$ packet records the local-solution data that a constructive BEDC consumer may inspect when an equation row is solved by a displayed derivative and an invertible linear approximation. The packet consumes the derivative source of chapter 122, the linear approximation surface of chapter 48, the finite coordinate ledger of chapter 49, the completeness boundary of chapter 128, the Picard contraction route of chapter 395, and the terminal real-name seal of chapter 38. It is not a host-level theorem asserting differentiability of an ambient function space or selecting a solution germ; it is a finite NameCert route from displayed analytic rows to a local graph window.

Definition 1834.1 (Implicit function carrier)**.** *An* $\mathsf{ImplicitFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, B, D, L, M, Q, G, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the displayed equation row. It stores the finite source variables, target variable, equation-value history, and local classifier for the read $F(x, y) = 0$.*
2. *B is the base-point row. It records the displayed point (x_0, y_0), the equation read at that point, and the finite neighbourhood name in which all later rows are interpreted.*
3. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row for the equation source at B, including the displayed partial-derivative histories consumed by the linear approximation.*
4. *L is the invertible* $\mathsf{LinMap}^{\uparrow}$ *approximation row attached to the target-variable derivative, together with its finite inverse-witness ledger.*
5. *M is the* $\mathsf{Mat}^{\uparrow}$ *coordinate row that displays the same approximation and inverse witness in finite component form for comparison and tolerance bookkeeping.*
6. *Q is the* $\mathsf{PicardContraction}^{\uparrow}$ *local-solution row. It records the displayed fixed-point map, contraction budget, iterate ledger, and local nonescape side condition obtained from D, L, M.*
7. *G is the local graph and window row. It records the source window, target window, graph readback, and uniqueness window exported by the Picard route.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the equation, base point, derivative, linear approximation, matrix ledger, Picard route, and graph window have been displayed.*

9. *H records componentwise* hsame *transport for the equation row, base-point row, derivative row, linear approximation row, matrix row, Picard local-solution row, graph-window row, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from an equation request through $E, B, D, L, M, Q, G, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{ImplicitFunction}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, B, D, L, M, Q, G, R, H, C, P, N$. The carrier has no coordinate for host differentiability, quotient function spaces, selected solution germs, the classical inverse-function theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionTasteGate single carrier alignment`

Theorem 1834.2 (Implicit function root equation carrier)**.** *Every accepted* $\mathsf{ImplicitFunction}^{\uparrow}$ *root packet must display the equation row, base point row, derivative row, linear map row, matrix row, Picard row, graph row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}$ *row. No root-equation consumer may read a local solution window before those rows are all visible in the same finite* BHist *packet.*

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionRootEquationCarrier`

Proof. Project the accepted packet through Definition 1834.1. The carrier inventory is exactly $E, B, D, L, M, Q, G, R, H, C, P, N$, so the displayed equation, base point, derivative, linear approximation, matrix ledger, Picard route, graph window, Real seal, transport, replay, provenance, and local name are the only rows available.

A root-equation read must therefore open E and B before any analytic route is interpreted. The rows D, L, M, Q, G then expose the derivative, linearization, component ledger, contraction route, and graph window in carrier order.

The terminal rows R, H, C, P, N seal, transport, replay, source, and name that same route. Since the carrier contains no second equation row or detached solution object, no local solution window can be consumed before the displayed rows have been read. □

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionCarrier root equation carrier`

Theorem 1834.3 (Implicit function linearization Picard route)**.** *The local graph route of an accepted* $\mathsf{ImplicitFunction}^{\uparrow}$ *packet can be consumed only after the* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *and* $\mathsf{PicardContraction}^{\uparrow}$ *rows are visible through D, L, M, Q and the completeness boundary named by the Picard row. Thus the graph row G is a contraction-route endpoint, not a host differentiability theorem or selected solution germ.*

Proof. Read the linearization rows from the carrier. The derivative row D supplies the equation source data, the linear-map row L supplies the invertible approximation, and the matrix row M supplies the finite component ledger for the same approximation and inverse witness.

The Picard row Q is the only fixed-point coordinate in the packet. It names the complete-metric boundary and the contraction budget before any graph endpoint is exported.

The graph row G is downstream of D, L, M, Q. Consequently a consumer of the local graph must pass through the displayed derivative, linear map, matrix, complete-metric, and Picard rows, and cannot extract host differentiability or a selected germ from an absent carrier coordinate. □

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionLinearizationPicardRoute`

Theorem 1834.4 (Implicit function local graph seal handoff)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal* R *of an accepted* $\mathsf{ImplicitFunction}^{\uparrow}$ *packet is a boundary for the displayed graph window* G*. It exports the equation, base point, derivative, linearization, matrix ledger, Picard route, graph window, transport, replay, provenance, and local name, but exports no host differentiability, selected solution germ, quotient function space, or ambient completeness principle.*

Proof. Apply the root equation carrier theorem above to expose the finite packet rows. The Real seal R is listed after the equation, base point, derivative, linearization, matrix, Picard, and graph-window rows.

Use the linearization Picard route theorem. It places G downstream of the derivative, linear map, matrix, complete-metric, and Picard data, so the seal reads only the displayed contraction endpoint.

The structural rows H, C, P, N then transport, replay, source, and name the sealed graph window. The excluded host differentiability, selected germ, quotient function space, and ambient completeness principles have no coordinate in Definition 1834.1, so the seal cannot export them. □

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionLocalGraphSealHandoff`

Theorem 1834.5 (Implicit function NameCert obligations)**.** *For every accepted* $\mathsf{ImplicitFunction}^{\uparrow}$ *carrier*

$$I = (E, B, D, L, M, Q, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed equation row, base-point row, derivative row, linear approximation row, matrix row, Picard row, graph-window row, Real seal, transport, replay, provenance, and local name;*
2. *derivative and invertibility ledger compatibility, requiring* D, L, M *to display the same target-variable approximation and inverse witness;*
3. *Picard local-solution handoff, requiring* Q *to be the only fixed-point route available to the packet;*
4. *graph-window transport, requiring* G *to expose local reads only through the source and target windows named by* B *and certified by* Q*;*
5. *Real-seal non-escape, requiring every read of* R *to remain downstream of* E, B, D, L, M, Q, G *and the structural rows;*
6. *provenance exactness, requiring* P *and* N *to name only the listed rows and no host differentiability theorem, quotient function space, selected solution germ, classical inverse-function theorem, or ambient completeness principle.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.ImplicitFunctionUp.ImplicitFunctionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1834.1. The finite packet lists the equation, base point, derivative source, invertible approximation, matrix ledger, Picard route, graph window, Real seal, transport, replay, provenance, and local name.

The derivative and invertibility obligations are local to D, L, M. The derivative row supplies the displayed partial-derivative histories, the linear-map row records the inverse witness, and the matrix row exposes the same witness in component form for finite tolerance comparison.

The local solution is then routed through Q. Since Q is the only Picard coordinate of the carrier, any fixed-point read must use the displayed contraction budget, iterate ledger, and nonescape row recorded there.

The graph-window and seal obligations are downstream. The row G reads the local graph only inside the carried source and target windows, and the row R is reached only after the equation, derivative, invertibility, Picard, and graph-window rows have been opened. The structural rows H, C, P, N transport, replay, source, and name this same route without adding a host solution object. □

Theorem 1834.6 (Implicit function Picard local-solution route)**.** *Every accepted* $\mathsf{ImplicitFunction}^{\uparrow}$ *carrier factors each local-solution read through the route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{Mat}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{PicardContraction}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by D, L, M, Q, R *and the graph-window row* G*. In particular, the* $\mathsf{Real}^{\uparrow}$ *seal is consumed only after the derivative row of chapter 122, the linear-map row of chapter 48, the matrix row of chapter 49, the complete-metric boundary of chapter 128, the Picard contraction route of chapter 395, and the real-name seal of chapter 38 have been displayed.*

Proof. Read the carrier coordinates. The equation and base-point rows E, B provide only the local problem statement and the neighbourhood in which the route is interpreted.

The analytic source then passes through D, L, M. The derivative row D supplies the local derivative data, the linear-map row L supplies the invertible approximation, and the matrix row M supplies finite componentwise comparisons for the same approximation and inverse witness.

The completeness and fixed-point part is exposed only by Q. Its contraction budget and nonescape side condition are the carried route to the complete-metric boundary and Picard local-solution ledger, so no alternate selected solution germ is available.

Finally G repacks the Picard endpoint as a local graph-window read, and R provides the terminal $\mathsf{Real}^{\uparrow}$ seal. Because R is downstream of D, L, M, Q, G, every Real-facing local-solution read factors through the listed chapters before the seal is consumed. □

Falsifiable prediction: Every $\mathsf{ImplicitFunction}^{\uparrow}$ local-solution read factors through one displayed equation row, one base-point row, one Derivative row, one invertible LinearMap approximation row, one Matrix component ledger, one PicardContraction local-solution row, one local graph-window row, one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: derivative, linearmap, matrix, completemetric, picardcontraction, real: $\mathsf{ImplicitFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries equation data, base-point data, derivative data, an invertible linear approximation, matrix coordinates, a Picard local-solution route, a local graph window, and a terminal Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1834.1, Definition 1834.5, Definition 1834.6, Definition 1834.3, and Definition 1834.4 bind the scoped BEDC primitives for derivative-window exposure, local inverse approximation, Picard local-solution routing, graph-window transport, Real-seal nonescape, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ImplicitFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, quotient function spaces, selected solution germs, the classical inverse-function theorem, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires checked carrier admission, derivative and invertibility ledger compatibility, Picard local-solution handoff, graph-window transport, Real-seal non-escape, provenance exactness, and local naming for BEDC.Derived.ImplicitFunctionUp.

Constructive story: An ImplicitFunction$^{\uparrow}$ packet is generated from one displayed equation row, one base-point row, one Derivative row, one invertible LinearMap approximation row, one Matrix component ledger, one PicardContraction local-solution row, one local graph-window row, one Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1835

A Concrete Naming Certificate for $\mathsf{UniformBoundedness}^{\uparrow}$

Following the Banach–Steinhaus theorem as presented in Bourbaki and Kreyszig, the $\mathsf{UniformBoundedness}^{\uparrow}$ packet records the finite certificate surface by which a pointwise bounded family of displayed bounded linear operators exposes a uniform bound on a displayed Baire-complete source window. It consumes the operator carrier of chapter 1644, the prefix-bar completeness handoff of chapter 1602, the norm rows of chapter 89, the functional-analysis consumer face of chapter 94, the stream scheduler of chapter 126, and the regular rational readback surface of chapter 301. The packet is a NameCert reconstruction of the traditional theorem's visible route; it is not an unrestricted host-level Banach–Steinhaus principle.

Definition 1835.1 (Uniform boundedness carrier)**.** *A* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, B, N, R, S, Q, H, C, P, L)$$

with the following displayed rows:

1. *F is a finite family ledger of* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows* $\mathsf{BanachBLOp}(X, Y_i, T_i, K_i, \Lambda_i)$ *sharing the displayed source carrier X and carrying target rows Y_i.*
2. *W is the pointwise boundedness window. For each carried source endpoint x inspected by the schedule, it records a* $\mathsf{Norm}^{\uparrow}$ *bound row for the finite family of target reads $T_i(x)$ and the local rational budget used for that inspection.*
3. *B is the* $\mathsf{BaireMetric}^{\uparrow}$ *bar and complete-metric handoff row on the source-side observation space. It records the displayed bar of windows on which pointwise boundedness is visible and the complete-metric route used to pass from local windows to a source neighbourhood.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *envelope ledger. It records the source norm window, the target norm rows, and the candidate uniform operator-bound endpoint.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the rational bound budget used to compare the local pointwise windows with the single displayed uniform bound.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule controlling which source endpoints, operator rows, norm rows, and rational comparisons may be inspected.*
7. *Q records componentwise* hsame *transport for the operator family, pointwise windows, Baire handoff, norm envelope, regular-rational readback, stream schedule, continuation, provenance, and local name.*

8. *H is the visible history row binding the operator-family reads, pointwise-boundedness witnesses, Baire bar handoff, norm-envelope rows, and readback comparisons into one* BHist *packet.*
9. *C records displayed* Cont *replay from a requested family-bound read through $F, W, B, N, R, S, Q, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformBoundedness}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the listed rows. It has no coordinate for host function extensionality, choice-selected uniform constants, quotient operator equality, a hidden operator space, or the unrestricted classical Banach–Steinhaus theorem.

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessUpTasteGate single carrier alignment`

Theorem 1835.2 (Uniform boundedness NameCert obligations)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier*

$$U = (F, W, B, N, R, S, Q, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{UniformBoundedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the shared-source family of* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, pointwise boundedness windows, Baire bar handoff, norm envelope, regular-rational readback, stream schedule, transport, history, replay, provenance, and local naming;*
2. *family coherence, requiring every operator row in F to share the displayed source carrier and to expose its endpoint action and norm bound through Definition 1644.1;*
3. *pointwise boundedness admission, requiring W to give only scheduled source endpoints and scheduled operator reads;*
4. *Baire handoff compatibility, requiring B to be the only route from pointwise windows to a source neighbourhood;*
5. *norm-envelope exactness, requiring N and R to display the same rational budget and the same uniform operator-bound endpoint;*
6. *ledger refusal, excluding host function extensionality, choice-selected uniform constants, quotient operator equality, hidden operator-space completion, and an unrestricted classical Banach–Steinhaus theorem.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessPacket namecert obligations`
↪ variant `BEDC.Derived.UniformBoundednessUp.UniformBoundednessTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1835.1. The tuple lists the finite operator family, pointwise boundedness window, Baire handoff, norm envelope, regular-rational readback, stream schedule, transport, history, replay, provenance, and local name.

Family coherence is read from F. Each member is a $\mathsf{BoundedLinearOperator}^{\uparrow}$ carrier over the same displayed source, so the endpoint actions, bound endpoints, and operator ledgers are exactly the rows admitted by Definition 1644.2.

Pointwise boundedness is local to W and S. The schedule chooses the finite source reads and the finite operator reads, while W records only the corresponding target-norm comparisons and rational budgets.

The passage from local pointwise windows to a source neighbourhood is routed through B. By Definition 1602.2, the handoff may inspect only displayed prefix windows, bar data, metric rows, transport, replay, provenance, and local naming.

Finally N and R bind the visible norm envelope to the regular-rational readback. Since the classifier sees only $F, W, B, N, R, S, Q, H, C, P, L$, none of the excluded host or quotient principles can be exported as a NameCert obligation. □

Theorem 1835.3 (Uniform boundedness carrier admission)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier*

$$U = (F, W, B, N, R, S, Q, H, C, P, L),$$

the admission surface is exactly the projection of the eleven displayed rows $F, W, B, N, R, S, Q, H, C, P, L$*. These rows expose the shared-source operator family, pointwise boundedness window, Baire handoff, norm envelope, regular-rational readback, stream schedule, componentwise transport, visible history, replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundedness carrier admission`

Proof. Open Definition 1835.1. The tuple presentation lists precisely $F, W, B, N, R, S, Q, H, C, P, L$, so every admitted coordinate is one of those rows.

Conversely each listed row appears as a displayed coordinate of the same carrier. Thus carrier admission is neither an ambient Banach-space assertion nor an unrestricted theorem schema; it is the finite inventory read from the accepted packet. □

Theorem 1835.4 (Uniform boundedness family coherence)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier, each row of* F *shares the displayed source carrier and reads its endpoint action and norm bound through the existing* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *envelope. No family member may be read as a host function, quotient operator, or independently completed operator-space point.*

Proof. The row F is specified in Definition 1835.1 as a finite family ledger of $\mathsf{BoundedLinearOperator}^{\uparrow}$ rows over the same source carrier. Reading a member of the family is therefore a read of the bounded-operator packet, not a read of an external function object.

Apply Definition 1644.2 and Definition 1644.5. They confine endpoint action, bound endpoint, linearity ledger, norm ledger, history, replay, provenance, and local naming to the displayed operator envelope, which gives the required family coherence. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessFamilyCoherence`

Theorem 1835.5 (Uniform boundedness pointwise window)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier, the pointwise boundedness evidence is scheduled only through* W *and* S*. A source endpoint, operator row, target norm comparison, or rational pointwise budget is visible to the packet only when the stream schedule* S *names the read and the window* W *stores the corresponding boundedness row.*

Proof. The carrier definition gives S as the finite schedule and W as the pointwise boundedness window. Thus a pointwise read has two required coordinates: the schedule that permits the source and operator inspection, and the window that stores the target-norm comparison.

No other row in U has pointwise boundedness as its role. The Baire row consumes windows downstream, while N and R record the envelope and rational readback; hence unscheduled pointwise data cannot be read from the packet. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessPointwiseWindow`

Theorem 1835.6 (Uniform boundedness Baire handoff compatibility)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier, a source neighbourhood may be read only after the pointwise windows have passed through the* $\mathsf{BaireMetric}^{\uparrow}$ *row* B*. The handoff uses* B *as the unique route from scheduled pointwise data to a displayed Baire-complete source window.*

Proof. Start with the scheduled pointwise data supplied by W and S. These rows identify finite source inspections and finite operator reads, but they do not themselves expose a source neighbourhood.

The source-neighbourhood passage is stored in B. By Definition 1602.2 and Definition 1602.8, the Baire row supplies the displayed bar and complete-metric route; since no other coordinate carries that route, compatibility is forced through B. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessBaireHandoffCompatibility`

Theorem 1835.7 (Uniform boundedness norm-envelope exactness)**.** *For every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier, the norm-envelope row* N *and the regular-rational readback row* R *name the same uniform-bound endpoint and the same rational comparison budget. A consumer cannot read a bound endpoint from* N *while substituting an unrelated rational budget from outside* R*.*

Proof. The row N records the source norm window, the target norm rows, and the candidate uniform operator-bound endpoint. The row R is the regular-rational readback used to compare the local pointwise budgets with that single displayed endpoint.

Because both rows occur in one carrier and are replayed by the same C row, the comparison endpoint and the rational budget are tied to the same finite history H. Thus exactness is the shared readback of N and R, not a separately chosen bound. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessNormEnvelopeExactness`

Theorem 1835.8 (Uniform boundedness ledger refusal)**.** *Every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier refuses host function extensionality, choice-selected uniform constants, quotient operator equality, hidden operator-space completion, and unrestricted Banach–Steinhaus. These principles are not coordinates of* $F, W, B, N, R, S, Q, H, C, P, L$ *and cannot be reconstructed by transport, replay, provenance, or local naming.*

Proof. Inspect the carrier inventory from Definition 1835.3. The visible data are the finite family ledger, pointwise windows, Baire handoff, norm envelope, rational readback, schedule, transport, history, replay, provenance, and local name.

The structural rows Q, H, C, P, L preserve and replay that inventory. They do not add host function extensionality, a selected uniform constant, quotient equality of operators, an operator-space completion, or the unrestricted classical theorem, so the ledger refuses each listed endpoint. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessLedgerRefusal`

Theorem 1835.9 (Uniform boundedness Baire handoff)**.** *Every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier factors the passage from pointwise boundedness to a uniform source-window bound through the displayed route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{BaireMetric}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Norm}^{\uparrow}.$$

The stream schedule first names finite source and operator reads, the Baire row supplies the bar and complete-metric handoff, the regular-rational row reads the finite bound budget, and the norm row records the source and target bound comparisons. No completed host space or countable-choice selector is introduced by the route.

Proof. Read an accepted carrier as $U = (F, W, B, N, R, S, Q, H, C, P, L)$. A pointwise boundedness read must begin with S, because W has no unscheduled source endpoint or unscheduled operator coordinate.

The Baire step is then forced through B. It is the only carrier coordinate naming a bar and complete-metric handoff, and Definition 1602.23 confines such handoffs to displayed prefix, schedule, radius, metric, and ultrametric rows.

The rational and norm comparisons are downstream of that handoff. The row R reads the regular rational budget attached to the selected window, and N records the matching norm envelope and uniform-bound endpoint.

Thus the source-window bound is obtained only by replaying S, B, R, N with the structural rows Q, H, C, P, L. The carrier contains no coordinate for a completed ambient host space or a countable-choice selector, so neither can be read from the handoff. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessBaireHandoff`

Theorem 1835.10 (Uniform boundedness bounded-operator envelope)**.** *Let U be an accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *carrier. Every uniform operator-bound conclusion exported by U reads each family member through a* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *carrier row and reads the common bound through* $\mathsf{Norm}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. In particular, the functional-analysis consumer face receives only the displayed operator ledgers, pointwise boundedness windows, Baire handoff, rational readback, and norm-envelope rows named in Definition 1835.1.*

Proof. Open the family row F. By construction its entries are $\mathsf{BoundedLinearOperator}^{\uparrow}$ packets sharing the source carrier, so Definition 1644.5 makes the endpoint action, bound endpoint, linearity ledger, norm ledger, history, replay, provenance, and local naming the only readable operator data.

The common envelope is not an additional operator-space object. It is the row N together with the rational readback R, so the comparison between individual operator bounds and the displayed uniform bound is carried by $\mathsf{Norm}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ evidence.

The $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer face therefore receives the same bounded-operator rows and the same norm-bound ledgers described in Definition 94.3. The uniform boundedness packet adds only the scheduled pointwise windows and the Baire handoff that justify reading one envelope from the finite family ledger. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessBoundedOperatorEnvelope`

Theorem 1835.11 (Uniform boundedness public boundary)**.** *Every accepted* $\mathsf{UniformBoundedness}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{BaireMetric}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{FunctionalAnalysis}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pk}$$

The public packet explicitly excludes host function extensionality, choice-selected uniform constants, quotient operator equality, hidden completion of an operator space, and an unrestricted classical Banach–Steinhaus theorem.

Proof. Let a consumer read an accepted carrier. By Definition 1835.1, the only visible coordinates are the operator family, pointwise window, Baire handoff, norm envelope, rational readback, stream schedule, componentwise transport, history, replay, provenance, and local name.

The operator data are confined by Definition 1835.10. The Baire passage is confined by Definition 1835.9, and the finite NameCert obligation surface is exactly the one listed in Definition 1835.2.

Combining these three readings gives the displayed public route and no other route. Since no carrier coordinate stores a host function-extensionality principle, a selected uniform constant, quotient equality of operators, hidden operator-space completion, or the full classical theorem, none of those objects can be exported by a public $\mathsf{UniformBoundedness}^{\uparrow}$ consumer. □

Lean4 ✓ `BEDC.Derived.UniformBoundednessUp.UniformBoundednessPublicBoundary`

Falsifiable prediction: Every accepted $\mathsf{UniformBoundedness}^{\uparrow}$ uniform-bound read factors through one finite family of $\mathsf{BoundedLinearOperator}^{\uparrow}$ rows, one pointwise boundedness window, one $\mathsf{BaireMetric}^{\uparrow}$ bar handoff, one $\mathsf{Norm}^{\uparrow}$ envelope, one $\mathsf{RegSeqRat}^{\uparrow}$ rational readback, one $\mathsf{StreamName}^{\uparrow}$ schedule, componentwise hsame transport, one BHist ledger, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: boundedlinearoperator, bairemetric, norm, functionalanalysis, streamname, regseqrat: $\mathsf{UniformBoundedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a shared-source operator family, pointwise boundedness windows, a Baire bar handoff, a norm envelope, regular-rational readback, scheduling, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformBoundedness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformBoundedness}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1835.1, Definition 1835.2, Definition 1835.9, Definition 1835.10, and Definition 1835.11 seed the finite Banach–Steinhaus NameCert surface over displayed bounded-operator families, Baire handoffs, norm envelopes, scheduled pointwise windows, and regular-rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformBoundedness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, choice-selected uniform constants, quotient operator equality, hidden completion of an operator space, an unrestricted classical Banach–Steinhaus theorem, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires the displayed BoundedLinearOperator, BaireMetric, Norm, FunctionalAnalysis, StreamName, RegSeqRat, BHist, hsame, Cont, Pkg, and NameCert rows to be bound into one public UniformBoundednessUp consumer scope.

Constructive story: A $\mathsf{UniformBoundedness}^{\uparrow}$ packet is generated from a finite shared-source family of $\mathsf{BoundedLinearOperator}^{\uparrow}$ rows, scheduled pointwise boundedness windows, a $\mathsf{BaireMetric}^{\uparrow}$ bar and complete-metric handoff, $\mathsf{Norm}^{\uparrow}$ envelope rows, $\mathsf{RegSeqRat}^{\uparrow}$ rational readback, $\mathsf{StreamName}^{\uparrow}$ scheduling, componentwise transport, a visible history ledger, replay, provenance, and a local NameCert row.

Chapter 1836

A Concrete Naming Certificate for $\mathsf{RegularCauchyComposition}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyComposition}^{\uparrow}$ packet records the finite route by which one regular Cauchy name is read through another regular Cauchy name before any completed real equality or function-space object is consulted. It sits over the shared finite windows of chapter 126, the rational regularity rows of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the nested dyadic Cauchy route of chapter 1811, and the Cauchy-rate lattice discipline of chapter 1818. The packet gives Real and Completion consumers one named finite composition surface instead of forcing each consumer to reassemble the StreamName, RegSeqRat, DyadicRatCore, and Real rows separately.

Definition 1836.1 (Regular Cauchy composition carrier)**.** *A* $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, W, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row. It stores the displayed rational observations, the source regularity rate, and the source-side classifier used by the composed read.*
2. *T is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *row. It stores the displayed rational observations, the target regularity rate, and the target-side classifier reached after the source row has supplied its accepted window.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger. It records the finite source windows, the finite target windows, and the handoff cells in which an accepted source observation licenses the target observation consumed next.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the outer tolerance request, the inner tolerance used to read S, the target tolerance used to read T, and the finite dyadic descent route binding these tolerances together.*
5. *K is the composition route. It records the displayed order S, W, D, T and the local compatibility checks saying that each target read is opened only after the matching source window and dyadic tolerance cells have been exposed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is reachable only after both regular rational rows, the shared window ledger, the dyadic tolerance ledger, and the composition route have all been displayed.*

7. *H records componentwise* hsame *transport for the source row, target row, shared window ledger, tolerance ledger, composition route, seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from an outer regular Cauchy request through $S, T, W, D, K, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyComposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NestedDyadicCauchy}^{\uparrow}$, $\mathsf{CauchyRateLattice}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $S, T, W, D, K, E, H, C, P, N$. The carrier has no coordinate for completed real equality, quotient stream equality, host-level function composition, selected limits, countable choice, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompositionUp.RegularCauchyCompositionTasteGate single carrier alignment`

Theorem 1836.2 (Regular Cauchy composition NameCert obligations)**.** *For every accepted* RegularCauchyCompos *carrier*

$$Q = (S, T, W, D, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular rational row, target regular rational row, shared stream-window ledger, dyadic tolerance ledger, composition route, terminal seal, transport, replay, provenance, and local naming row;*
2. *source-window exactness, requiring every source-side observation consumed by the composed read to be listed in S and opened through W;*
3. *target-window exactness, requiring every target-side observation to be opened from the handoff cells in W after the matching source observation has been accepted;*
4. *dyadic tolerance compatibility, requiring the outer request, source tolerance, target tolerance, and descent route to be the displayed cells of D;*
5. *composition-route determinacy, requiring K to name the displayed order S, W, D, T and to forbid any detached host-level composition map;*
6. *Real-seal non-escape, requiring E to remain downstream of S, T, W, D, K and the structural rows;*
7. *componentwise structural preservation, requiring H and C to transport and replay the whole packet without changing the handoff cells;*
8. *provenance exhaustion, requiring P and N to source and name exactly the displayed packet and no completed real equality, quotient stream equality, selected limit, countable-choice schedule, Lean verification, or bridge checker.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1836.1. The projection exposes exactly two regular rational rows, one shared stream-window ledger, one dyadic tolerance ledger, one composition route, one terminal seal, and the structural rows.

The source-window and target-window obligations are read from S, T, W. A source observation must first appear in the source row and in the shared window ledger. Only then may the corresponding handoff cell open the target observation.

The tolerance obligation is read from D. The outer request, the source-side tolerance, the target-side tolerance, and the dyadic descent cells are all visible in the same ledger, so a consumer cannot replace the route by an unrecorded modulus.

The composition route is then forced by K. It records the order S, W, D, T and therefore blocks any direct jump from a source rational observation to a completed real equality or a host-level composition map.

Finally, E, H, C, P, N only seal, transport, replay, source, and name the displayed packet. Since the carrier has no further analytic coordinate, the local NameCert exports exactly the listed finite obligations and no external completion principle. □

Theorem 1836.3 (Regular Cauchy composition stream handoff)**.** *Let*

$$Q = (S, T, W, D, K, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier. Every target-side* $\mathsf{StreamName}^{\uparrow}$ *read in the composed route is opened from a handoff cell of W whose source-side cell has already been accepted by S and whose tolerance cell is displayed in D. Thus the stream handoff is stable under componentwise* hsame *transport and displayed* Cont *replay inside the packet.*

Proof. Begin with a target-side stream read. The carrier gives no independent target-window selector; the only target-window coordinates are the handoff cells recorded in W.

The source row must be consumed first. The accepted source observation is listed in S, and the same finite window ledger W records the cell that authorizes the target read.

The tolerance ledger supplies the remaining compatibility condition. The cell in D displays the outer request and the inner tolerances used by both regular rational rows, so the handoff cannot be detached from the dyadic descent route.

Transport and replay preserve the handoff because H acts componentwise on $S, T, W, D, K, E, C, P, N$, while C replays only the displayed order. Therefore every target-side stream read remains downstream of the accepted source cell and the displayed tolerance cell. □

Definition 1836.4 (Regular Cauchy composition modulus carrier)**.** *A modulus-refined* $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q_\mu = (S_0, S_1, W, D, \mu, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows whose regular rational observations are composed.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule used by both regular rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, displaying the outer tolerance, the intermediate tolerance, and the endpoint tolerance consumed by the composition.*
4. *μ is the composed modulus row. It records the finite route from an outer precision request through the shared window schedule and both regular rational rows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after S_0, S_1, W, D, μ have all been displayed.*
6. *H records componentwise* hsame *transport for the two regular rows, shared schedule, tolerance ledger, composed modulus row, seal, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from an outer composed-modulus request through $S_0, S_1, W, D, \mu, E, H, P, N$.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyComposition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$ *row.*

The classifier compares modulus-refined packets only through the displayed tuple $S_0, S_1, W, D, \mu, E, H, C, P, N$. It has no coordinate for a host-level modulus function, quotient stream equality, selected limit, completed real equality, Lean verification, or bridge checking.

Theorem 1836.5 (Regular Cauchy composition modulus stability)**.** *Let*

$$Q_\mu = (S_0, S_1, W, D, \mu, E, H, C, P, N)$$

be an accepted modulus-refined $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier. Componentwise* hsame *transport of the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal preserves the composed modulus route μ. Hence every transported packet exposes the same finite outer-request to endpoint-read route through $S_0, S_1, W, D, \mu, E, H, C, P, N$.*

Proof. Project the modulus carrier of Definition 1836.4. The projection exposes the two regular rational rows, the shared window schedule, the tolerance ledger, the composed modulus row, and the terminal seal.

Transport is componentwise by the row H. It moves the displayed histories of $S_0, S_1, W, D, \mu, E, C, P, N$ together, so the composed modulus row cannot be separated from the two source regular rows or from the dyadic tolerance cells that justify it.

Replay is restricted by C. It repeats the displayed order from an outer composed-modulus request through the shared window schedule, both regular rows, the dyadic tolerance ledger, and the terminal seal.

Therefore the transported packet still exposes the same finite composed modulus route. No host-level modulus function, quotient stream equality, selected limit, or completed real equality is introduced by the transport. □

Theorem 1836.6 (Regular Cauchy composition modulus Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read out of an accepted modulus-refined* $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mu \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the shared schedule, the two regular rational rows, the dyadic tolerance ledger, the composed modulus row, and the terminal seal. Thus every Real and Completion consumer reads the composed name through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *$\mu$, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Take a Real-facing read admitted by the modulus-refined carrier. Projection of Q_μ gives no coordinate from an outer request directly to the terminal seal E.

The read first opens the shared StreamName schedule and the two regular rational rows. The schedule W fixes the finite windows, while S_0 and S_1 display the rational observations consumed at those windows.

The tolerance and modulus rows then determine the Real-facing handoff. The ledger D displays the outer, intermediate, and endpoint tolerances, and μ records the composed route through those cells.

Only after those rows are displayed can E be consumed. The structural rows H, C, P, N preserve, replay, source, and name the same route, so every Real-facing read factors through the stated finite packet. □

Theorem 1836.7 (Regular Cauchy composition Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read out of an accepted* $\mathsf{RegularCauchyComposition}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyComposition}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by W*,* S, T*,* D*,* K*, and* E*. Equivalently, a Real or Completion consumer can inspect the composed name only through the shared stream windows, the two regular rational rows, the dyadic tolerance ledger, the composition route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Take a Real-facing read admitted by the carrier. Projection of Q gives no coordinate from an outer request directly to the terminal seal E.

The read first passes through the shared stream windows and the two regular rational rows. The row W displays the source and target windows, while S and T display the regular rational observations consumed at those windows.

The dyadic tolerance ledger and composition route then determine the endpoint. The row D records the tolerance descent, and K records the order in which the source row, handoff cells, tolerance cells, and target row are read.

Only after those rows are displayed can the terminal seal E be consumed. The rows H, C, P, N preserve, replay, source, and name the same route, so no completed real equality, quotient stream equality, host-level function composition, selected limit, countable choice, Lean verification, or bridge checker can be inserted. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyComposition}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first reading the shared $\mathsf{StreamName}^{\uparrow}$ windows, both $\mathsf{RegSeqRat}^{\uparrow}$ rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the displayed composition route refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, nested-dyadic-cauchy, cauchy-rate-lattice, real: no listed sibling supplies the whole ten-row packet $Q = (S, T, W, D, K, E, H, C, P, N)$, because each sibling contributes only window, regularity, tolerance, nested route, rate discipline, or terminal seal data while this chapter binds source and target regular Cauchy name reads into one finite composition carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyComposition}^{\uparrow}$ packet is generated from source and target RegSeqRat rows, a shared StreamName window ledger, a DyadicRatCore tolerance ledger, a displayed composition route, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$, seed)

Scope closed: Definition 1836.1, Definition 1836.2, Definition 1836.3, and Definition 1836.7 seed the finite route for composing regular Cauchy name reads through StreamName, RegSeqRat, DyadicRatCore, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyComposition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed real equality, quotient stream equality, host-level function composition, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, target-window exactness, dyadic tolerance compatibility, composition-route determinacy, Real-seal non-escape, stream handoff stability, and Real seal factorization for BEDC.Derived.RegularCauchyCompositionUp.

Chapter 1837

A Concrete Naming Certificate for $\mathsf{RealRootExistence}^{\uparrow}$

Following Bishop and Bridges on constructive intermediate-value reasoning, the $\mathsf{RealRootExistence}^{\uparrow}$ packet turns a displayed sign-change bracket and its bisection-to-Cauchy seal into a public finite BEDC root-existence surface. It is grounded by the intermediate-value row of chapter 1402, the sign-change bracket of chapter 1786, the bisection seal of chapter 1798, the finite stream interface of chapter 126, the regular rational and dyadic readback of chapter 301 and chapter 291, and the terminal real-name boundary of chapter 38. The chapter names the public existence certificate; it does not select a root.

Definition 1837.1 (RealRootExistence carrier)**.** *A* $\mathsf{RealRootExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, F, S, B, R, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval row. It records the lower and upper endpoint windows, endpoint order evidence, interval identity, and tolerance requests consumed by the public existence read.*
2. *F is the endpoint sign row. It stores the displayed opposite-sign endpoint data and the sign orientation used by the root-existence packet.*
3. *S is the continuity modulus row. It records the source tolerance demanded by each endpoint, midpoint, retained-window, and Real-facing read.*
4. *B is the* $\mathsf{IntermediateValue}^{\uparrow}$ *ledger row. It stores the finite intermediate-value obligation surface attached to the displayed interval, endpoint signs, and modulus data.*
5. *R is the* $\mathsf{RealRootBracket}^{\uparrow}$ *row. It records the finite bracket surface that keeps the sign-change interval visible as a BEDC certificate rather than as a host theorem.*
6. *Q is the* $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ *row. It records the finite bisection branch ledger, retained windows, regular Cauchy readback, and public seal handoff.*
7. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row. It contains the regular rational observations, dyadic tolerances, endpoint comparisons, branch radii, and rounding rows used by the seal.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after the sign-change bracket, bisection branch, retained windows, and regular readback have been exposed.*

9. *H records componentwise* hsame *transport for the interval, signs, modulus, intermediate-value ledger, bracket row, bisection seal, regular and dyadic readback, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a public root-existence request through located interval inspection, endpoint sign inspection, modulus use, intermediate-value admission, bracket exposure, bisection sealing, regular and dyadic readback, and Real-seal consumption.*

11. *P is the* Pkg *provenance over* $\mathsf{RealRootExistence}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{BisectionRootCauchySea}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealRootExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, S, B, R, Q, D, E, H, C, P, N$. It has no coordinate for a selected root, quotient stream equality, real trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host root equality.*

Lean4 ✓ `BEDC.Derived.RealRootExistenceUp.TasteGate.RealRootExistenceTasteGate single carrier alignment`

Theorem 1837.2 (RealRootExistence NameCert obligations)**.** *For every accepted* $\mathsf{RealRootExistence}^{\uparrow}$ *carrier*

$$X = (I, F, S, B, R, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealRootExistence}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the located interval, endpoint signs, continuity modulus, intermediate-value ledger, real-root bracket, bisection-root Cauchy seal, regular and dyadic readback, Real seal, transports, replay, provenance, and local naming;*

2. *sign-change stability, requiring I and F to expose the same located interval and opposite displayed endpoint signs before any public existence row is available;*

3. *continuity and modulus compatibility, requiring S to answer each bracket, bisection, and seal request by source precision consumed by the same finite route;*

4. *bisection branch regularity, requiring B, R, Q to keep the intermediate-value ledger, bracket row, branch prefix, midpoint sign rows, retained windows, and Cauchy readback attached to the displayed sign-change packet;*

5. *Real-seal non-escape, requiring D, E, H, C, P, N to export only the displayed existence route and no selected root, quotient real equality, real trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host root equality.*

No RealRootExistence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealRootExistenceUp.RealRootExistenceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1837.1. The accepted packet displays exactly the located interval, endpoint signs, modulus row, intermediate-value ledger, bracket row, bisection seal, regular and dyadic readback, Real seal, and structural rows.

Sign-change stability is the entry stage. The rows I and F must be opened together, so the public existence read cannot replace the displayed interval and endpoint signs by an ambient real interval or by a host sign theorem.

Continuity and modulus compatibility is the precision stage. The row S supplies the source precision used by endpoint inspection, midpoint inspection, retained-window requests, and the terminal seal, so all later rows consume one displayed continuity route.

Bisection branch regularity is the middle stage. The row B admits the intermediate-value ledger, R exposes the bracket surface, and Q records the finite branch prefix, midpoint sign rows, retained windows, and regular Cauchy handoff.

Finally, D, E, H, C, P, N perform readback, sealing, transport, replay, provenance, and naming for the same route. Since selected roots, quotient real equality, real trichotomy, open-cover compactness, ambient Real completeness, and host root equality are absent from the carrier, none of the five obligation rows can export them. □

Theorem 1837.3 (RealRootExistence bisection seal handoff)**.** *Every accepted* $\mathsf{RealRootExistence}^{\uparrow}$ *read factors through* $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{BisectionRootCauchySeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any public root seal is visible. In row form the route is*

$$I, F, S \longmapsto B \longmapsto R \longmapsto Q \longmapsto D \longmapsto E, H, C, P, N.$$

The handoff exports a finite public existence certificate and a terminal seal boundary, not a selected root, quotient stream, trichotomy decision, or ambient Real-completeness principle.

Proof. Start with an accepted carrier $X = (I, F, S, B, R, Q, D, E, H, C, P, N)$. The only entry rows are the located interval, endpoint sign, and modulus rows I, F, S.

The intermediate-value and bracket stages are forced next. The row B records the $\mathsf{IntermediateValue}^{\uparrow}$ obligation surface attached to I, F, S, and the row R records the $\mathsf{RealRootBracket}^{\uparrow}$ surface that keeps the sign-change bracket visible as finite data.

The bisection seal stage is Q. It records the $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ branch ledger, retained windows, StreamName-facing observations, and regular Cauchy handoff, all downstream of the same intermediate-value and bracket rows.

The public seal stage is terminal. The row D reads the route as $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ data, the row E seals the resulting boundary as $\mathsf{Real}^{\uparrow}$ data, and H, C, P, N transport, replay, package, and name that same route.

Thus every accepted read factors through the listed siblings before the public seal is visible. Because the carrier has no coordinate for a selected root, quotient stream, trichotomy row, or ambient Real-completeness theorem, the handoff cannot export any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealRootExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealRootExistence}^{\uparrow}$ packet is generated from a located interval row, endpoint sign row, continuity modulus row, $\mathsf{IntermediateValue}^{\uparrow}$ ledger, $\mathsf{RealRootBracket}^{\uparrow}$ row, $\mathsf{BisectionRootCauchySeal}^{\uparrow}$ row, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealRootExistence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1837.1, Definition 1837.2, and Definition 1837.3 seed the finite public sign-change root-existence route from located interval and endpoint sign rows through intermediate-value, bracket, bisection-seal, regular readback, dyadic readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealRootExistence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected roots, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host root equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-change stability, bisection branch regularity, Cauchy-seal handoff, and Real-seal non-escape for BEDC.Derived.RealRootExistenceUp.

Chapter 1838

A Concrete Naming Certificate for BishopCompletionUniversalBoundary$^{\uparrow}$

Following Bishop and Bridges' constructive treatment of completion, the BishopCompletionUniversalBoundary$^{\uparrow}$ packet isolates the finite L10 boundary at which a Bishop regular Cauchy completion route can be consumed by a universal-property extension and uniqueness surface. It reads the completion carrier of chapter 1639, the extension and uniqueness face of chapter 397, and the regular-Cauchy universal seed of chapter 1425. Its source observations remain the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readback of chapter 301, the DyadicRatCore$^{\uparrow}$ tolerance ledger of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

The packet is only a boundary certificate. It names the finite rows that let a completion consumer move from regular Cauchy data to an extension route and a uniqueness ledger, while excluding quotient streams, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, and any bridge credential.

Definition 1838.1 (Bishop completion universal boundary carrier)**.** *A* BishopCompletionUniversalBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, R, D, A, U, E, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* BishopRegularCauchyCompletion$^{\uparrow}$ *row, exposing the regular Cauchy completion carrier and its common-tail route.*
2. *S is the* StreamName$^{\uparrow}$ *finite-window row read by the completion source and by downstream extension consumers.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the windows in S.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used by the regular Cauchy route, the extension request, and the uniqueness comparison.*
5. *A is the terminal* Real$^{\uparrow}$ *seal reached only after Q, S, R, D have been displayed.*
6. *U is the* CompletionUniversalProperty$^{\uparrow}$ *face restricted to the displayed Bishop regular Cauchy route.*
7. *E is the extension route row, read through U after the rows Q, S, R, D, A have fixed the source and seal surface.*

8. *L is the uniqueness ledger, a displayed* hsame *comparison between two extension rows over the same Q, S, R, D, A, U, E route.*
9. *H records componentwise* hsame *transport for the completion row, stream windows, regular readback, dyadic ledger, Real seal, universal-property face, extension route, uniqueness ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a completion-universal request through $Q, S, R, D, A, U, E, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, R, D, A, U, E, L, H, C, P, N$. The carrier contains no quotient stream, selected limit, countable-choice selector, host real equality row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, or standard-bridge credential.*

Lean4 ✓ `BEDC.Derived.BishopCompletionUniversalBoundaryUp.BishopCompletionUniversalBoundaryTasteGate`
`single carrier alignment`

Theorem 1838.2 (Bishop completion universal extension route)**.** *Let $B = (Q, S, R, D, A, U, E, L, H, C, P, N)$ be an accepted* $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$ *carrier. Every extension read exported by E first displays the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *row Q, then the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal A, and only then reads the restricted* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *face U.*

Proof. Project the carrier of Definition 1838.1. The extension row E is downstream of the completion carrier Q, the finite source windows S, the regular rational readback R, the dyadic tolerance ledger D, the terminal seal A, and the universal-property face U.

The sibling completion chapters supply exactly these rows. The Bishop regular Cauchy completion packet fixes the completion-facing source, while the StreamName, RegSeqRat, DyadicRatCore, and Real chapters expose the finite observation route consumed by the universal-property face.

The continuation row C can replay only the displayed route, and the provenance row P names no selected limit, quotient stream, host real equality, choice schedule, or ambient completeness object. Therefore every extension read factors through Q, S, R, D, A, U before E is exported. □

Theorem 1838.3 (Bishop completion universal uniqueness ledger)**.** *Let*

$$B = (Q, S, R, D, A, U, E, L, H, C, P, N)$$

and

$$B' = (Q, S, R, D, A, U, E', L', H', C', P, N')$$

be accepted $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$ *carriers over the same completion row, stream windows, regular readback, dyadic ledger, Real seal, universal-property face, and provenance. Any comparison between the exported extension rows is exhausted by the displayed uniqueness ledgers L, L' together with their componentwise* hsame *transports and displayed* Cont *replays. No quotient-stream equality, selected limit, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, or bridge credential can contribute to the comparison.*

Proof. The shared coordinates Q, S, R, D, A, U, P fix the completion source, finite windows, readback, tolerance ledger, terminal seal, universal-property face, and provenance available to both packets.

The only extension-sensitive coordinates are E, L, H, C, N and E', L', H', C', N'. The ledgers L and L' compare extension rows over the fixed displayed source route, while H, H' and C, C' provide the permitted transports and replays.

Since neither accepted carrier contains any excluded quotient, selected-limit, choice, host-equality, completeness, or bridge coordinate, the comparison cannot leave the displayed uniqueness surface. It is exactly the local hsame ledger over the Bishop regular Cauchy universal boundary. □

Theorem 1838.4 (Bishop completion universal NameCert obligations)**.** *For every accepted* BishopCompletionUnive *carrier* $B = (Q, S, R, D, A, U, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Bishop regular Cauchy completion row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, CompletionUniversalProperty face, extension route, uniqueness ledger, transport, replay, provenance, and local naming row;*
2. *source-route exactness, requiring every universal-property request to pass through* Q, S, R, D, A *before* U, E, L *can be inspected;*
3. *extension-route exactness, requiring* E *to be read only as the extension route attached to the displayed* Q, S, R, D, A, U *surface;*
4. *uniqueness-ledger exactness, requiring every comparison of extension rows to use* L *over the same displayed source, seal, universal-property, and extension coordinates;*
5. *regular-Cauchy compatibility, requiring the boundary to reuse the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *extension and uniqueness seed without adding a second completion carrier;*
6. *non-escape, requiring* H, C, P, N *to transport, replay, source, and name no row from which quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, or bridge credentials can be exported.*

No local obligation is stored outside these rows.

```
Lean4 ✓  BEDC.Derived.BishopCompletionUniversalBoundaryUp.BishopCompletionUniversalBoundaryCarrier
namecert obligations
```

Proof. Carrier admission is projection from Definition 1838.1. The accepted packet displays exactly the completion carrier, finite windows, readback, tolerance ledger, Real seal, universal-property face, extension route, uniqueness ledger, and structural rows.

Source-route exactness follows from the order of the carrier. A universal-property request first reads Q and then the finite observation route S, R, D, A before it can inspect the rows U, E, L.

Extension-route exactness is Definition 1838.2. The extension row E is attached to the displayed Bishop regular Cauchy source and cannot be read as a detached host completion map.

Uniqueness-ledger exactness is Definition 1838.3. The comparison is carried by the displayed ledgers L, L' and their transports and replays, over the same visible source and extension coordinates.

Regular-Cauchy compatibility is the provenance constraint in P. The packet cites the RegularCauchyCompletionU seed as a source for the extension and uniqueness shape, but it stores no second completion carrier beyond the displayed row Q.

Finally, non-escape is exhausted by H, C, P, N. These rows transport, replay, source, and name the displayed route only, and none contains quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, or bridge credentials. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1838.1, Definition 1838.2, Definition 1838.3, and Definition 1838.4 seed the finite L10 boundary from Bishop regular Cauchy completion data to extension and uniqueness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-route exactness, extension-route exactness, uniqueness-ledger exactness, regular-Cauchy compatibility, and non-escape proofs for BEDC.Derived.BishopCompletionUniversalBoundary

Constructive story: A $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$ packet is generated from a BishopRegularCauchyCompletion row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, a CompletionUniversalProperty face, an extension route, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1839

A Concrete Naming Certificate for $\mathsf{RealOscillation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealOscillation}^{\uparrow}$ packet records the finite oscillation interface by which a real-facing consumer reads tail variation, interval clamping, and squeeze enclosure before it treats a displayed name as Real or Completion data. It refines the tail ledger of chapter 448, the interval route of chapter 1810, and the envelope route of chapter 1772. Its numerical rows are grounded in the dyadic tolerance surface of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet is therefore a finite L10 interface for oscillation arguments, not an assertion of ambient Real completeness or host equality.

Every $\mathsf{RealOscillation}^{\uparrow}$ read consumes the clamped interval route of chapter 1810 and the squeeze envelope of chapter 1772 before the terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

Definition 1839.1 (Real oscillation carrier)**.** *A* $\mathsf{RealOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q_{\mathrm{osc}} = (X, O, C, S, D, W, R, H, T, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *seal whose oscillation-facing reads are being routed.*
2. *O is the* $\mathsf{CauchyOscillation}^{\uparrow}$ *row exposing the finite tail-variation ledger.*
3. *C is the* $\mathsf{RealClamping}^{\uparrow}$ *interval row recording the clamped bounded-window route.*
4. *S is the* $\mathsf{RegularCauchySqueeze}^{\uparrow}$ *envelope row recording the lower, middle, and upper regular Cauchy squeeze data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row supplying the dyadic radius and endpoint slack consumed by O, C, S.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened at the precision named by D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from X, O, C, S, D, W before any Real or Completion-facing oscillation conclusion is exposed.*
8. *H records componentwise* hsame *transport for the Real seal, oscillation ledger, clamp interval, squeeze envelope, dyadic tolerance, window, readback, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay from each Real or Completion-facing oscillation request through $O, C, S, D, W, R, H, P, N$ before X is consumed as a terminal seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyOscillation}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, $\mathsf{RegularCauchySqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, O, C, S, D, W, R, H, T, P, N$ and the displayed routes among those rows. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, host trichotomy, selected limits, countable choice, Lean verification, or bridge checking.*

Lean4 (definition) `BEDC.Derived.RealOscillationUp.RealOscillationCarrier`

Theorem 1839.2 (Real oscillation NameCert obligations)**.** *For every accepted* $\mathsf{RealOscillation}^{\uparrow}$ *carrier*

$$Q_{\mathrm{osc}} = (X, O, C, S, D, W, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealOscillation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Real seal, CauchyOscillation ledger, RealClamping interval, RegularCauchySqueeze envelope, dyadic tolerance, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming rows;*

2. *oscillation-ledger exposure, requiring O to be displayed before any Real or Completion-facing oscillation read can consume the seal X;*

3. *clamp interval exposure, requiring C, D, W, R to display the bounded interval route used by the oscillation packet;*

4. *squeeze-envelope compatibility, requiring S to consume the same tolerance, window, and regular rational readback route as the oscillation and clamp rows;*

5. *ledger nonescape, requiring H, T, P, N to export no ambient Real completeness, quotient stream equality, host trichotomy, selected limit, countable-choice schedule, Lean verification, standard bridge credential, or row outside Q_{osc}.*

No Real oscillation obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealOscillationUp.RealOscillationTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1839.1. The accepted packet displays exactly the Real seal, oscillation ledger, clamp interval, squeeze envelope, tolerance, finite window, readback, transport, replay, provenance, and local naming rows.

Oscillation-ledger exposure is forced by the row O. A consumer cannot inspect the terminal seal X as oscillation evidence before the CauchyOscillation ledger has displayed the tail-variation data attached to the same packet.

The clamp and squeeze obligations are the two bounded routes around that ledger. The clamp row C is read through D, W, R, while the squeeze row S uses the same tolerance, window, and readback route to align its envelope with the displayed oscillation ledger.

The structural rows H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier inventory omits ambient completeness, quotient stream equality, host trichotomy, selected limits, countable choice, Lean verification, bridge credentials, and external rows, the listed obligations exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.RealOscillationUp.RealOscillationUpTasteGate single carrier alignment`

Theorem 1839.3 (Real oscillation clamped-window factorization)**.** *Let*

$$Q_{\mathrm{osc}} = (X, O, C, S, D, W, R, H, T, P, N)$$

be an accepted $\mathsf{RealOscillation}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read of the oscillation packet factors through the displayed route*

$$O \longmapsto C \longmapsto S \longmapsto D, W, R \longmapsto H, T, P, N \longmapsto X.$$

Consequently the read receives the CauchyOscillation tail ledger, RealClamping interval route, RegularCauchySqueeze envelope, dyadic tolerance, StreamName window, RegSeqRat readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it reaches the Real seal.*

Proof. Start with the accepted carrier and project the rows $X, O, C, S, D, W, R, H, T, P, N$. The terminal seal X is present in the packet, but the classifier exposes it to oscillation consumers only through the displayed rows that name the oscillation route.

The read first reaches the tail ledger O, then the clamp interval row C, and then the squeeze envelope row S. These rows supply tail variation, bounded interval routing, and lower-middle-upper enclosure as finite data.

The rows D, W, R provide the shared numerical route. They fix the dyadic tolerance, open the finite StreamName window, and repack the observations as RegSeqRat readback before any Real or Completion-facing conclusion is exposed.

The remaining rows H, T, P, N preserve the same route by transport, replay, provenance, and local naming. No coordinate in the packet can replace that route by ambient Real completeness, quotient stream equality, host trichotomy, a selected limit, or a standard bridge credential, so the displayed factorization is exhaustive. □

Falsifiable prediction: Every $\mathsf{RealOscillation}^{\uparrow}$ Real or Completion-facing read factors through one CauchyOscillation tail ledger, one RealClamping interval route, one RegularCauchySqueeze envelope, one DyadicRatCore tolerance row, one StreamName window, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert, and the terminal RealUp seal.

Independence witness: cauchyoscillation, real-clamping, regular-cauchy-squeeze, dyadicratcore, streamname, regseqrat, real: $\mathsf{RealOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Real seal, an oscillation ledger, a clamp interval route, a squeeze envelope, one shared dyadic-window-readback route, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealOscillation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealOscillation}^{\uparrow}$ packet is generated from one RealUp seal, one CauchyOscillation ledger, one RealClamping interval row, one RegularCauchySqueeze envelope, one DyadicRatCore tolerance row, one StreamName window, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealOscillation}^{\uparrow}}$, seed)

Scope closed: Definition 1839.1, Definition 1839.2, and Definition 1839.3 seed the finite Real and Completion-facing oscillation route through tail variation, clamping, squeeze enclosure, dyadic tolerance, finite windows, regular rational readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealOscillation}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.RealOscillationUp.RealOscillationCarrier`

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, host trichotomy, selected limits, countable choice, Lean theorem checking, or standard bridge credentials.

Upgrade path: Obligation closure requires checked carrier admission, oscillation-ledger exposure, clamp interval exposure, squeeze-envelope compatibility, clamped-window factorization, and ledger nonescape for BEDC.Derived.RealOscillationUp.RealOscillation

Chapter 1840

A Concrete Naming Certificate for $\mathsf{MeanErgodic}^{\uparrow}$

Following von Neumann on Hilbert-space mean convergence and Birkhoff on orbit averages, the $\mathsf{MeanErgodic}^{\uparrow}$ packet records only the finite Cesaro-average ledger visible to a BEDC consumer. It consumes the measure-preserving transformation surface of chapter 196, the finite transition rows of chapter 195, the event and preservation rows of chapter 92, the finite transfer-row handoff of chapter 456, and the Cesaro averaging route of chapter 1765. The chapter names a finite averaging NameCert surface; it does not claim a host Hilbert projection, pointwise convergence theorem, invariant-function quotient, or ambient completed orbit space.

Definition 1840.1 (Mean ergodic carrier)**.** *A* $\mathsf{MeanErgodic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (E, D, W, T, H, A, P, R, C, Q, N)$$

with the following displayed rows:

1. *E is the measure-preserving* $\mathsf{Ergodic}^{\uparrow}$ *source row. It displays the carried transformation, the invariant-row ledger, the measure-preservation witness, and the finite dynamic and measure endpoints already visible through* $\mathsf{Ergodic}^{\uparrow}$*.*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *step row for the same transformation, recording the source point row, successor row, and finite iterate endpoints used by the orbit window.*
3. *W stores finite orbit windows $x, Tx, \ldots, T^{n-1}x$ together with the visible unary window length n.*
4. *T is the* $\mathsf{TransferOperator}^{\uparrow}$ *or transition-row handoff. It records the finite row by which each displayed step is transported into the averaging ledger when the packet is read through a transfer-style consumer.*
5. *H is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *averaging row. It stores the prefix sums, denominator row, and finite Cesaro division route for the orbit-window observations.*
6. *A stores the averaged endpoint rows exposed to downstream consumers, each obtained only after W, T, H have been displayed.*
7. *P is the* $\mathsf{Measure}^{\uparrow}$ *preservation row, recording that the same measure ledger is carried across the displayed transformation and its finite orbit-window reads.*
8. *R stores componentwise* hsame *transport for the ergodic source, dynamic steps, finite windows, transfer rows, Cesaro averages, averaged endpoints, measure preservation, replay, provenance, and local naming.*

9. *C stores displayed* Cont *replay from an orbit-window request through $E, D, W, T, H, A, P, R, Q, N$.*
10. *Q is the* Pkg *provenance over* $\mathsf{MeanErgodic}^{\uparrow}$, $\mathsf{Ergodic}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{TransferOperator}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MeanErgodic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, W, T, H, A, P, R, C, Q, N$. It has no coordinate for a host Hilbert-space projection, pointwise ergodic convergence, completed invariant subspace, quotient orbit space, spectral theorem, or ambient Banach-space limit.

Lean4 ✓ `BEDC.Derived.MeanErgodicUp.MeanErgodicTasteGate single carrier alignment`

Theorem 1840.2 (Mean ergodic NameCert obligations)**.** *For every accepted* $\mathsf{MeanErgodic}^{\uparrow}$ *carrier $M = (E, D, W, T, H, A, P, R, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MeanErgodic}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed ergodic source, dynamic step row, finite orbit windows, transfer handoff, Hardy–Cesaro averaging row, averaged endpoint rows, measure-preservation row, transport, replay, provenance, and local name;*
2. *orbit-window exactness, requiring every entry of W to be read from the displayed* $\mathsf{DynSystem}^{\uparrow}$ *step ledger and the measure-preserving* $\mathsf{Ergodic}^{\uparrow}$ *source row;*
3. *Cesaro-average exactness, requiring every exposed average in A to be computed by the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row from the displayed finite orbit window and denominator row;*
4. *preservation stability, requiring the* $\mathsf{Measure}^{\uparrow}$ *preservation row to be transported with the same* hsame *and* Cont *routes as the orbit-window source;*
5. *nonescape, requiring the packet to export no Hilbert projection, no pointwise convergence theorem, no invariant-function quotient, no completed orbit space, and no ambient spectral or Banach-space limit.*

Proof. Carrier admission is projection from Definition 1840.1. The visible packet lists exactly the ergodic source, dynamic steps, orbit windows, transfer rows, Cesaro averaging row, averaged endpoints, preservation row, transport rows, replay rows, provenance, and local name.

Orbit-window exactness is read from D and W. Each window entry is generated by replaying the displayed $\mathsf{DynSystem}^{\uparrow}$ step row over the transformation already carried by E, so there is no independent trajectory object.

Cesaro-average exactness is read from H and A. The averaging row consumes the finite window W, prefix sums, the visible denominator, and dyadic division data supplied by $\mathsf{HardyCesaroMean}^{\uparrow}$ before an averaged endpoint can be exposed.

Preservation stability is then read from P, R, C. The same measure-preservation row is transported along the displayed hsame and replay routes used by the orbit-window source, so a measure claim cannot detach from the carried transformation.

Finally, nonescape follows from the carrier inventory. Since none of the coordinates names a Hilbert projection, pointwise convergence theorem, invariant quotient, completed orbit space, spectral theorem, or Banach limit, the local NameCert obligations are exhausted by the finite rows above. □

Theorem 1840.3 (Mean ergodic Cesaro-average handoff)**.** *Every averaged endpoint exported by an accepted* $\mathsf{MeanErgodic}^{\uparrow}$ *packet factors through the route*

$$\mathsf{Ergodic}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow} \longmapsto \mathsf{TransferOperator}^{\uparrow} \longmapsto \mathsf{HardyCesaroMean}^{\uparrow} \longmapsto \mathsf{Measure}^{\uparrow}$$

as instantiated by E, D, T, H, P and the finite orbit-window row W. Consequently, the consumer sees only the displayed transformation, finite orbit windows, transfer or step handoff rows, Hardy–Cesaro prefix averages, measure-preservation ledger, hsame *transport,* Cont *replay,* Pkg *provenance, and local naming before reading A.*

Proof. Start with an averaged-endpoint request. The request first opens E, so the transformation and preservation source are the carried $\mathsf{Ergodic}^{\uparrow}$ rows rather than a host measurable dynamical system.

The finite orbit is then read through D and W. These rows display the step ledger and the selected finite window length, so the consumer has only the requested finite prefix of the orbit.

If the packet is viewed through a transfer-style consumer, T records the handoff from the step rows into the finite transfer ledger. This handoff is still a finite row route and does not add an analytic transfer operator or a spectral datum.

The averaging itself is supplied by H. By chapter 1765, the prefix sum, denominator, and division rows are the only route from the finite window to a Cesaro mean.

The preservation row P is finally replayed along the same transported source. Thus every exported endpoint in A factors through the displayed finite route and no alternate host projection or convergence theorem can be read. □

Theorem 1840.4 (Mean ergodic nonescape boundary)**.** *An accepted* $\mathsf{MeanErgodic}^{\uparrow}$ *carrier exports no data beyond the finite rows $E, D, W, T, H, A, P, R, C, Q, N$. In particular, it does not export a Hilbert-space orthogonal projection onto invariant vectors, a pointwise almost-everywhere convergence theorem, an invariant-function quotient, an infinite orbit measure, a completed orbit closure, a spectral decomposition, or a Banach-space limit operator.*

Proof. Project the accepted carrier to the tuple of Definition 1840.1. The only readable coordinates are the displayed finite rows $E, D, W, T, H, A, P, R, C, Q, N$.

The first group E, D, W, T, H, A, P accounts for the mathematical content: carried ergodic source, displayed dynamic steps, finite windows, transfer handoff, Cesaro averaging, averaged endpoints, and measure preservation. Each row has a named sibling source and a finite replay route.

The second group R, C, Q, N accounts for transport, continuation replay, package provenance, and local naming. These structural rows preserve and name the same packet; they do not add a projection, quotient, completion, spectral, or Banach-space coordinate.

Therefore any downstream consumer is confined to the finite rows in the carrier. The listed host-level objects are not merely unproved; they are absent from the packet boundary. □

Falsifiable prediction: Every $\mathsf{MeanErgodic}^{\uparrow}$ averaged endpoint read factors through one ErgodicUp measure-preserving source row, one DynSystemUp finite step ledger, one finite orbit-window row, a TransferOperatorUp or step handoff row, one HardyCesaroMeanUp averaging row, one MeasureUp preservation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ergodic, dynsystem, transferoperator, hardycesaromean, measure: $\mathsf{MeanErgodic}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a measure-preserving ergodic source, finite orbit windows, transition handoff rows, Cesaro prefix averages, preservation stability, transport, replay, provenance, and local naming while excluding host projection and convergence coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{MeanErgodic}^{\uparrow}}$.

Constructive story: A MeanErgodic$^{\uparrow}$ packet is generated from an ErgodicUp measure-preserving source row, DynSystemUp finite step rows, finite orbit-window rows, TransferOperatorUp or step handoff rows, HardyCesaroMeanUp prefix-average rows, MeasureUp preservation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MeanErgodic}^{\uparrow}}$, seed)

Scope closed: Definition 1840.1, Definition 1840.2, Definition 1840.3, and Definition 1840.4 seed the finite Cesaro-average ergodic packet from measure-preserving orbit windows to averaged endpoint reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{MeanErgodic}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.MeanErgodicUp.MeanErgodicTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Hilbert-space projection, pointwise ergodic convergence, invariant-function quotients, infinite orbit measures, completed orbit closures, spectral decompositions, Banach-space limit operators, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, orbit-window exactness, Cesaro-average handoff, preservation stability, nonescape, and local naming for BEDC.Derived.MeanErgodicUp.

Chapter 1841

A Concrete Naming Certificate for $\mathsf{RegularCauchyBall}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyBall}^{\uparrow}$ packet records finite-window membership in a rational-radius ball around a displayed regular Cauchy real name. It sits over the finite observation windows of chapter 126, the regular rational readback rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later metric and completion consumers a local-neighbourhood surface before any ambient real ball, selected limit, quotient equality, trichotomy, countable choice, compactness principle, or $\mathsf{Real}^{\uparrow}$ completeness claim is available.

Definition 1841.1 (Regular Cauchy ball carrier)**.** *A* $\mathsf{RegularCauchyBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, C, R, W, D, Q, S, H, T, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row for the point being tested.*
2. *C is the* $\mathsf{RegSeqRat}^{\uparrow}$ *centre row for the displayed ball.*
3. *R is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, including the rational endpoint and dyadic scale used by the membership test.*
4. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger from which X and C are inspected.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the requested window and radius budget.*
6. *Q is the comparison row recording that the displayed window distance is inside the radius budget after the dyadic tolerance descent in D.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after X, C, R, W, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the point row, centre row, radius row, window ledger, tolerance row, comparison row, seal row, replay row, provenance row, and local naming row.*
9. *T records displayed* Cont *replay from a ball-membership request through $X, C, R, W, D, Q, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBall}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBall}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed tuple $X, C, R, W, D, Q, S, H, T, P, N$. The carrier has no coordinate for ambient real balls, quotient equality of real names, selected limits, trichotomy, countable-choice schedules, metric-space compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyBallUp.TasteGate.RegularCauchyBallTasteGate single carrier alignment`

Theorem 1841.2 (Regular Cauchy ball window stability)**.** *Let*

$$B = (X, C, R, W, D, Q, S, H, T, P, N)$$

be an accepted $\mathsf{RegularCauchyBall}^{\uparrow}$ *carrier. Every ball-membership read of B factors through the same finite route*

$$X, C, R \longmapsto W, D \longmapsto Q \longmapsto S, H, T, P, N.$$

Consequently any two reads transported by H and replayed by T use the same point row, centre row, radius row, finite window ledger, dyadic tolerance descent, comparison row, and seal boundary.

Proof. Project the accepted carrier. The point row X, centre row C, and radius row R are the only analytic inputs available to a membership request, and the request has no direct access to an ambient real ball.

The finite observation step is therefore forced through W and D. The window ledger W names the finite observations of X and C, while the tolerance row D supplies the dyadic descent needed to compare those observations against the radius row R.

The comparison row Q is read only after those finite rows are displayed. The seal boundary S is downstream of Q, so a Real-facing consumer receives a window-stable membership route rather than a host-level ball assertion.

Finally transport and replay are componentwise. The rows H and T preserve the same displayed inputs and the same route through W, D, Q, S, while P and N source and name that finite packet. Thus repeated reads cannot introduce a second window, hidden radius, selected limit, quotient equality, compactness principle, or $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 1841.3 (Regular Cauchy ball NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBall}^{\uparrow}$ *carrier $B = (X, C, R, W, D, Q, S, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBall}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the point* $\mathsf{RegSeqRat}^{\uparrow}$ *row, centre* $\mathsf{RegSeqRat}^{\uparrow}$ *row, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger, dyadic tolerance row, comparison row,* $\mathsf{Real}^{\uparrow}$ *seal boundary, transport, replay, provenance, and local name;*
2. *finite-window exposure, requiring membership to be tested only on the windows displayed by W;*
3. *radius-budget exactness, requiring the radius row R and tolerance row D to be read before the comparison row Q;*
4. *comparison routing, requiring Q to bind the point and centre observations to the radius budget through the same dyadic ledger;*
5. *seal non-escape, requiring S to remain downstream of X, C, R, W, D, Q;*
6. *structural stability, requiring H, T, P, N to transport, replay, source, and name exactly the displayed route.*

No local NameCert obligation is stored outside these rows.

Proof. Read the carrier projection from Definition 1841.1. It exposes exactly the two regular rational names, one positive dyadic radius row, one finite window ledger, one dyadic tolerance row, one comparison row, one seal boundary, and the structural rows.

Finite-window exposure is the first obligation. A membership request can inspect X and C only through W, and the dyadic tolerance row D supplies the visible descent used with the radius row R.

The comparison row then records the local ball test. Since Q is downstream of R, W, D, it is a finite-window comparison row and not an ambient real-ball predicate.

The remaining obligations are structural. The seal row S cannot be read before the comparison route is displayed, while H, T, P, N only preserve, replay, source, and name the same finite packet. The excluded principles are absent because no carrier coordinate stores Real completeness, trichotomy, selected limits, countable choice, metric compactness, or quotient equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBall}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1841.1, Definition 1841.2, and Definition 1841.3 seed the finite-window ball-membership surface for two $\mathsf{RegSeqRat}^{\uparrow}$ rows, a positive $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, shared $\mathsf{StreamName}^{\uparrow}$ windows, dyadic tolerance descent, comparison routing, and terminal $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, selected limits, countable choice, metric-space compactness, quotient equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, radius-budget exactness, comparison routing, seal non-escape, structural stability, and local naming for BEDC.Derived.RegularCauchyBallUp.

Constructive story: A $\mathsf{RegularCauchyBall}^{\uparrow}$ packet is generated from a point RegSeqRat row, a centre RegSeqRat row, a positive DyadicRatCore radius row, shared StreamName windows, a dyadic tolerance row, a comparison row, a terminal RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1842

A Concrete Naming Certificate for $\mathsf{PrecompactCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PrecompactCompletion}^{\uparrow}$ packet reads $\mathsf{PrecompactMetric}^{\uparrow}$ finite-net and filter rows through $\mathsf{CompletionEmbedding}^{\uparrow}$ and $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ without quotient completion, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits between the precompact-metric bridge of chapter 1531, the totally bounded completion route of chapter 1573, and the constructive metric completion route of chapter 1556; its role is to preserve the finite BEDC ledger that completion consumers inspect.

Definition 1842.1 (Precompact completion carrier)**.** *A* $\mathsf{PrecompactCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, N, F, E, C, S, H, R, Q, L)$$

with the following displayed rows:

1. *X is the source metric carrier row, including accepted point histories, metric-distance witnesses, and classifier transports already exposed by the* $\mathsf{PrecompactMetric}^{\uparrow}$ *packet.*
2. *N is the finite-net ledger inherited from* $\mathsf{PrecompactMetric}^{\uparrow}$*. For each positive tolerance it records the closed* $\mathsf{ProbeBundle}$ *of centers,* $\mathsf{InBundle}$ *membership rows, center histories, distance witnesses, rational comparison ledgers, and net-refinement reads.*
3. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *replay row generated from the same metric and finite-net data, including finite basis packets, common-refinement rows, regular tail windows, and dyadic neighbourhood witnesses.*
4. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that inserts source observations into the dense completion boundary while preserving the displayed finite-net and filter-basis ledgers.*
5. *C is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route through regular Cauchy names, dyadic tolerances, separated comparison, finite stream windows, regular rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *naming boundary.*
6. *S is the stream-and-real seal row. It records the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *reads that completion-facing consumers may inspect after F, E, C have been replayed.*
7. *H records componentwise* hsame *transport for the source metric, finite nets, filter-basis replay, embedding row, constructive completion route, stream-and-real seal, provenance, and local naming rows.*

8. *R is the route ledger of displayed* Cont *reads, requiring every consumer to pass through N, F, E, C, S in that order before it reaches H, Q, L.*
9. *Q is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{PrecompactMetric}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}$ *row naming the displayed packet and the permitted reads from it.*

The classifier on such packets is componentwise transport through H together with replay through the displayed route row R. No coordinate stores a quotient completion, a selected completed point, a choice schedule, an open-cover compactness argument, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PrecompactCompletionUp.PrecompactCompletionTasteGate single carrier alignment`

Theorem 1842.2 (PrecompactCompletion NameCert obligations)**.** *For every accepted* $\mathsf{PrecompactCompletion}^{\uparrow}$ *carrier $P = (X, N, F, E, C, S, H, R, Q, L)$, the local* $\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed source metric, finite-net ledger, Cauchy-filter basis, completion embedding, constructive completion route, stream-and-real seal, transport ledger, route ledger, provenance, and local naming row;*
2. *finite-net persistence, requiring every completion tolerance request to read the centers, membership constructors, distance witnesses, rational bounds, and refinement rows recorded in N;*
3. *filter-basis replay, requiring each completion-facing read to consume the finite basis packets, common-refinement rows, regular tail windows, and dyadic neighbourhood witnesses in F before it reaches the embedding row;*
4. *dense-embedding handoff, requiring source observations to pass through E before the constructive completion route C may be inspected;*
5. *constructive-completion non-escape, requiring the completed read to continue through C and S using only* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *naming rows exposed by the finite route;*
6. *ledger exactness, excluding quotient completion, selected limits, countable choice, open-cover compactness, detached* $\mathsf{Real}^{\uparrow}$ *completeness, and ambient completeness.*

No NameCert obligation is stored outside these displayed rows.

Proof. Carrier admission is projection from Definition 1842.1. The packet lists exactly the source metric, finite-net ledger, Cauchy-filter basis, dense embedding, constructive completion route, stream-and-real seal, transport ledger, route ledger, provenance, and local naming row.

Finite-net persistence is read from N. A tolerance request inspects the closed ProbeBundle, its InBundle constructors, center histories, metric witnesses, rational comparisons, and refinement rows; no later row can replace this ledger by an open cover or by a host finite set.

Filter-basis replay and dense-embedding handoff follow the route order in R. The basis row F consumes the displayed finite-net data, and the embedding row E receives only those basis reads before placing source observations in the dense completion boundary.

The constructive-completion read is then local to C and S. Regular Cauchy names, dyadic tolerances, separated comparison, finite stream windows, regular rational readbacks, and the terminal real-name seal are the only completion-facing rows exposed downstream.

Finally, H, Q, L record transport, provenance, and local naming for the same coordinates. Since the excluded principles have no carrier coordinate, no replay edge, and no provenance row, the local $\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}$ surface is exactly the listed obligation surface. □

Theorem 1842.3 (PrecompactCompletion density handoff)**.** *Let $P = (X, N, F, E, C, S, H, R, Q, L)$ be an accepted* $\mathsf{PrecompactCompletion}^{\uparrow}$ *packet. Every completion-facing consumer factors through*

$\mathsf{PrecompactMetric}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Re

Equivalently, the consumer reads the finite route

$$X, N \longmapsto F \longmapsto E \longmapsto C \longmapsto S \longmapsto H, R, Q, L$$

rather than a quotient completion class, a selected completed point, a choice schedule, an open-cover compactness proof, or an ambient completeness theorem.

Proof. Start with the source metric row X and the finite-net ledger N. These are precisely the precompact metric data of chapter 1531: point histories, distance witnesses, finite centers, membership constructors, rational bounds, and refinement reads.

The consumer next reaches the Cauchy-filter basis row F. This step keeps the handoff finite because the basis packets, common refinements, regular tail windows, and dyadic neighbourhood witnesses are generated from the same rows already exposed by $\mathsf{PrecompactMetric}^{\uparrow}$.

Density enters only through E. The $\mathsf{CompletionEmbedding}^{\uparrow}$ row receives source observations after the basis replay and places them in the completion boundary without adding quotient representatives or selected limits.

The constructive completion route C and seal row S complete the factorization. They expose the regular Cauchy, stream-name, regular-rational, and real-name reads of $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ while preserving the displayed route ledger.

Transport, provenance, and local naming through H, R, Q, L do not add new data; they preserve the same dependency order. Thus every completion-facing consumer factors through the named route and cannot inspect any of the excluded nonconstructive or ambient-completeness objects. □

Theorem 1842.4 (PrecompactCompletion ledger exactness)**.** *For every accepted* $\mathsf{PrecompactCompletion}^{\uparrow}$ *packet $P = (X, N, F, E, C, S, H, R, Q, L)$, every exported completion read remains inside the finite ledger*

$$X, N, F, E, C, S, H, R, Q, L.$$

It cannot export a quotient completion, a selected completed point, countable choice, open-cover compactness, detached $\mathsf{Real}^{\uparrow}$ *completeness, or ambient completeness.*

Proof. Project an exported completion read from the carrier. By Definition 1842.1, the read has access only to the ten displayed coordinates and their componentwise transport rows.

Apply Definition 1842.2. Finite-net persistence keeps the read inside N, filter-basis replay moves it through F, dense-embedding handoff moves it through E, and constructive-completion non-escape moves it through C and S.

The remaining rows H, R, Q, L record transport, route order, provenance, and local naming for the same data. Every excluded object is absent from these rows, so no exported completion read can leave the displayed finite ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1842.1, Definition 1842.2, Definition 1842.3, and Definition 1842.4 seed the finite precompact-completion route from explicit precompact metric rows through dense constructive completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrecompactCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PrecompactCompletionUp.PrecompactCompletionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, open-cover compactness, full $\mathsf{Real}^{\uparrow}$ completeness, ambient completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-net persistence, filter-basis replay, dense-embedding handoff, constructive-completion non-escape, and ledger exactness for the same packet.

Constructive story: A $\mathsf{PrecompactCompletion}^{\uparrow}$ packet starts from a precompact metric source, attaches finite Cauchy-filter basis replay, routes source observations through a dense completion embedding and constructive metric completion, then exposes only stream-name, regular-rational, and Real naming rows to downstream consumers.

Chapter 1843

Compact Baire NameCert Construction

Beginning from the Baire category theorem for complete metric spaces, as developed by Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{CompactBaire}^{\uparrow}$ packet records the finite BEDC route by which a compact metric certificate may feed a dense-open Baire certificate without importing open-cover compactness or host intersections. It reads the compact metric carrier of chapter 129, the dense-open handoff of chapter 1542, the enumerated dense surface of chapter 1566, the complete-metric witness of chapter 128, the finite windows of chapter 126, and the regular rational readback of chapter 301. The chapter names the compact-metric Baire consequence as certificate composition, not as a host theorem about arbitrary intersections.

Definition 1843.1 (Compact Baire carrier)**.** *A* $\mathsf{CompactBaire}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, B, D, C, S, R, T, H, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, exposing the metric carrier, total-bounded finite-net data, and complete-limit route that will be inspected by the Baire argument.*
2. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open row, restricted to the displayed compact-metric source row X.*
3. *D is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *row used to enumerate the finite dense probes that a scheduled dense-open request may read.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row inherited from the compact metric certificate and presented as the only completion route visible to the Baire row.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule controlling which compact probes, dense-open rows, countable dense indices, and completion windows may be inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tolerance and radius comparisons used by the finite dense-open route.*
7. *T is the compact Baire trace ledger, recording the finite passage from compact-metric data through the complete-metric handoff and dense-open route to the displayed readback.*
8. *H records componentwise* hsame *transport for X, B, D, C, S, R, T and the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactBaire}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactBaire}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no open-cover compactness object, countable-choice selector, host set intersection, ambient Real completeness row, Lean verification credential, or standard bridge credential.

Lean4 ✓ `BEDC.Derived.CompactBaireUp.CompactBaireTasteGate single carrier alignment`

Theorem 1843.2 (Compact Baire NameCert obligations)**.** *For every accepted* $\mathsf{CompactBaire}^{\uparrow}$ *carrier* $K = (X, B, D, C, S, R, T, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactBaire}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric row, Baire dense-open row, countable dense row, complete-metric handoff, stream schedule, regular rational readback, trace ledger, transport, provenance, and local naming;*
2. *compact-source exposure, requiring every dense-open and finite-probe read to enter through the same* $\mathsf{CompactMetric}^{\uparrow}$ *row* X*;*
3. *complete-metric handoff, requiring the Baire row to consume the inherited* $\mathsf{CompleteMetric}^{\uparrow}$ *route* C *rather than an ambient completion principle;*
4. *dense-open finite routing, requiring the* $\mathsf{BaireCategory}^{\uparrow}$ *row* B *to use only the displayed schedule* S*, dense row* D*, readback row* R*, and trace ledger* T*;*
5. *nonescape, excluding open-cover compactness, countable choice, host set intersections, ambient Real completeness, Lean verification, and bridge checking.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1843.1. The accepted tuple displays exactly the compact metric row, Baire dense-open row, countable dense row, complete-metric handoff, stream schedule, regular rational readback, trace ledger, transport, provenance, and local name.

Compact-source exposure is forced by X. The dense-open row B, dense index row D, complete handoff C, schedule S, and readback R are all presented downstream of the same compact metric certificate.

The complete-metric handoff is the row C. Since C is inherited from the compact metric certificate, the packet has no coordinate from which an ambient completion theorem or separate Real-completeness principle can be selected.

Dense-open finite routing is read from B, S, R, T. The route names finite windows, regular rational comparisons, and the compact Baire trace ledger before any dense-open conclusion is exported.

Finally, H, P, N transport, source, and name only the displayed rows. Hence the carrier exports no open-cover compactness object, countable-choice selector, host set intersection, ambient Real completeness row, Lean verification credential, or bridge credential. □

Theorem 1843.3 (Compact Baire complete metric handoff)**.** *Let* $K = (X, B, D, C, S, R, T, H, P, N)$ *be an accepted* $\mathsf{CompactBaire}^{\uparrow}$ *carrier. Every Baire read exported by* K *reaches* $\mathsf{BaireCategory}^{\uparrow}$ *only after the compact metric row* X *has exposed its inherited* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff* C*.*

Proof. Read the carrier from its source row. The compact metric coordinate X is the first row that can supply the metric carrier, total-bounded finite-net surface, and complete-limit route.

The Baire row B is attached to that same source. Its complete-metric requirement is not stored independently; it is the displayed row C inherited from X.

Thus any Baire read accepted by the packet passes through X and C before consuming the dense-open rows. The remaining rows S, R, T, H, P, N only schedule, read back, trace, transport, source, and name that route. □

Theorem 1843.4 (Compact Baire dense-open finite route)**.** *Every dense-open request accepted by a* $\mathsf{CompactBaire}^{\uparrow}$ *carrier factors through the displayed finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CountableDenseSubset}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The route records compact-source admission, completion handoff, dense-open refinement, countable dense probing, scheduled windows, and rational readback before the local name is exported.

Proof. Begin with a dense-open request. Carrier admission sends the request first to X, and Definition 1843.3 identifies the displayed completion row C as the only complete-metric route available to the Baire read.

The dense-open step is the row B. It is read with the countable dense row D, so the request sees only displayed finite probes rather than a choice-selected dense subset.

The schedule and readback are then supplied by S and R. The trace ledger T records the same compact-source, completion, dense-open, dense-probe, window, and rational-readback route, while H, P, N transport, source, and name it. □

Theorem 1843.5 (Compact Baire nonescape boundary)**.** *For an accepted* $\mathsf{CompactBaire}^{\uparrow}$ *carrier* $K = (X, B, D, C, S, R, T, H, P, N)$*, every exported comparison, replay, provenance read, and local name is exhausted by the displayed rows* $X, B, D, C, S, R, T, H, P, N$*. In particular, the packet exports no open-cover compactness object, countable-choice selector, host set intersection, ambient Real completeness theorem, Lean verification credential, or bridge credential.*

Proof. Project the accepted carrier. The classifier compares two packets only through the displayed compact metric, Baire, countable dense, complete-metric, stream, regular-rational, trace, transport, provenance, and local naming rows.

The possible escape points are exactly the compactness, density, completion, and dense-open interfaces. The compactness row X is read as the $\mathsf{CompactMetric}^{\uparrow}$ certificate of chapter 129; the complete row C is its displayed completion handoff; the dense row D is the finite index surface of chapter 1566; and the Baire row B is the dense-open route of chapter 1542.

None of these coordinates stores an open-cover compactness object, countable-choice selector, host set intersection, ambient Real completeness theorem, Lean verification credential, or bridge credential. The structural rows only replay and name the same route, so no exported read can leave the displayed boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactBaire}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactBaire}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1843.1, Definition 1843.2, Definition 1843.3, Definition 1843.4, and Definition 1843.5 seed the compact-metric Baire certificate route over displayed compactness, completeness, countable dense probing, dense-open refinement, finite windows, and regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactBaire}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, host set intersections, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, complete-metric handoff, dense-open finite routing, trace exactness, and nonescape proofs for BEDC.Derived.CompactBaireUp.

Constructive story: A CompactBaire$^{\uparrow}$ packet is generated from a CompactMetric$^{\uparrow}$ row, a BaireCategory$^{\uparrow}$ dense-open row, a CountableDenseSubset$^{\uparrow}$ dense-probe row, an inherited CompleteMetric$^{\uparrow}$ handoff, StreamName$^{\uparrow}$ scheduling, RegSeqRat$^{\uparrow}$ readback, a trace ledger, componentwise transport, package provenance, and a local NameCert row.

Chapter 1844

A Concrete Naming Certificate for $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ packet records the finite four-object boundary at which rational approximants, stream names, regular rational sequences, and the terminal Real seal can be cited together without asserting a completeness theorem. It reads the dyadic endpoint and tolerance ledger of chapter 291, the finite-window stream surface of chapter 126, the regular-rational readback of chapter 301, the terminal seal of chapter 38, and the Bishop regular Cauchy completion carrier of chapter 1639.

The packet is an interface certificate. It names the rows that a completion-facing consumer may inspect when moving among the L10 rational, stream, regular-sequence, and Real surfaces, while excluding quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, and bridge checking.

Definition 1844.1 (Cauchy real completion interface carrier)**.** *A* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, S, R, B, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance row used by the rational boundary, stream-window requests, regular-sequence comparisons, and Real-seal reads.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the displayed windows that a completion consumer may replay.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational sequence row attached to the same dyadic tolerance and stream-window data.*
4. *B is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after D, S, R have fixed the rational, stream, and regular-sequence faces.*
5. *T is the completion-boundary tail ledger, recording the finite handoff from dyadic tolerances through stream windows and regular rational readback to the terminal Real seal.*
6. *H records componentwise* hsame *transports for the dyadic, stream, regular-sequence, Real-seal, tail-ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-boundary request through D, S, R, B, T, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealCompletionInterface}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, B, T, H, C, P, N$. The carrier contains no quotient stream equality, selected limit, countable-choice selector, host real equality row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, or standard-bridge credential.*

Theorem 1844.2 (Cauchy real completion interface NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carrier $I = (D, S, R, B, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealCompletionInterface}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed DyadicRatCore endpoint and tolerance row, StreamName finite windows, RegSeqRat readback, Real seal, completion-boundary tail ledger, transport, replay, provenance, and local naming row;*
2. *dyadic endpoint exposure, requiring every interface read to project D before inspecting stream windows, regular rational readback, or the terminal Real seal;*
3. *stream-window exactness, requiring S to expose only finite windows tied to the displayed dyadic tolerance row;*
4. *regular-sequence handoff, requiring R to reuse the same D, S surface rather than selecting a second sequence representative;*
5. *Real-seal boundary, requiring every* $\mathsf{Real}^{\uparrow}$ *read to remain downstream of D, S, R and the tail ledger T;*
6. *non-escape, requiring H, C, P, N to transport, replay, source, and name no row from which quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or bridge credentials can be exported.*

No local obligation is stored outside these rows.

```
Lean4 ✓  BEDC.Derived.CauchyRealCompletionInterfaceUp.CauchyRealCompletionInterfaceTasteGate
single carrier alignment
```

Proof. Carrier admission is projection from Definition 1844.1. The accepted packet displays exactly the dyadic endpoint and tolerance row, finite stream windows, regular-rational readback, terminal Real seal, completion-boundary tail ledger, and structural rows.

The dyadic exposure and stream-window exactness clauses are forced by the order of the carrier. A completion-boundary request first reads D, then obtains only the finite windows in S that are tied to that displayed tolerance row.

The regular-sequence handoff is the projection to R. Since R is stored after D, S, the interface has no coordinate from which a second stream representative or detached rational sequence can be selected.

The Real-seal boundary is carried by B and T. The terminal seal is read only after the rational, stream, and regular-sequence faces have been displayed, and the tail ledger records that same handoff.

Finally, non-escape is exhausted by H, C, P, N. These rows transport, replay, source, and name the displayed interface only, and none contains quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or bridge credentials. □

Theorem 1844.3 (Cauchy real completion four-face handoff)**.** *Let $I = (D, S, R, B, T, H, C, P, N)$ be an accepted* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carrier. Every completion-facing read exported by I factors through the four displayed faces*

$$D \longmapsto S \longmapsto R \longmapsto B$$

before it can inspect the tail ledger T or the structural rows H, C, P, N.

Proof. Begin with a completion-facing read of I. Carrier admission provides no source row before the dyadic endpoint and tolerance ledger D.

The read then passes to the finite-window row S. These windows are the only stream observations exposed by the packet, so no quotient stream equality or host stream representative can enter the route.

Next the read projects the regular-rational row R. This row is attached to the displayed dyadic and stream data, and it does not select a second regular sequence.

The terminal Real seal B is therefore reached only after D, S, R. The tail ledger T and structural rows H, C, P, N replay the same route, so every exported completion-facing read factors through the four displayed faces in order. □

Lean4 ✓ `BEDC.Derived.CauchyRealCompletionInterfaceUp.CauchyRealCompletionInterfaceUpTasteGate`
`single carrier alignment`

Theorem 1844.4 (Cauchy real completion interface Real-seal route determinacy)**.** *Let $I = (D, S, R, B, T, H, C, P, N)$ and $I' = (D', S', R', B', T', H', C', P', N')$ be accepted* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carriers. Suppose the two packets expose componentwise matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. Then every completion-facing consumer read of I and every matching read of I' reaches the same terminal Real-seal route after the four-face handoff through D, S, R, B and D', S', R', B'. Any remaining exported comparison is exhausted by the shared T, H, C, P, N rows, so there is no second seal coordinate, quotient stream equality, selected limit, countable-choice selector, host real equality, ambient completeness datum, or bridge credential along which the two reads could diverge.*

Proof. Project both accepted carriers to their displayed rows. The componentwise matching assumptions identify the dyadic endpoint and tolerance rows, finite stream windows, regular-rational readbacks, and terminal Real seals as the only route data available to the two completion-facing reads.

Apply Definition 1844.3 to each carrier. Both reads must pass through the four displayed faces in order, so the matched D, S, R, B and D', S', R', B' rows force the same terminal Real-seal route before any tail-ledger or structural row can be inspected.

It remains to compare the residual rows. The carrier projection leaves only T, H, C, P, N for completion-boundary tail data, componentwise transport, replay, provenance, and local naming. These rows replay the already matched handoff and contain no second seal coordinate, quotient stream equality, selected limit, countable-choice selector, host real equality, ambient completeness datum, or bridge credential. Hence no further route choice remains. □

Theorem 1844.5 (Cauchy real completion interface non-escape)**.** *For an accepted* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carrier $I = (D, S, R, B, T, H, C, P, N)$, every exported comparison, replay, provenance read, or local name is exhausted by the displayed rows $D, S, R, B, T, H, C, P, N$. In particular, the packet exports no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit, countable-choice selector, host real equality, or bridge credential.*

Proof. Project the accepted packet. Its classifier compares carriers only through the displayed rows $D, S, R, B, T, H, C, P, N$.

The comparison-sensitive rows are T, H, C, P, N. The tail ledger records the finite handoff across D, S, R, B, transport is componentwise, replay follows the displayed request path, provenance names the same source chapters, and the local NameCert row names only this packet.

Since no accepted coordinate stores a quotient stream equality, selected limit, choice selector, host real equality, ambient Real-completeness datum, or bridge credential, no exported comparison or replay can use such a row. The packet therefore remains a boundary interface rather than a completeness assertion. □

Theorem 1844.6 (Cauchy real completion interface tail-ledger determinacy)**.** *Let $I = (D, S, R, B, T, H, C, P, N)$ and $I' = (D', S', R', B', T', H', C', P', N')$ be accepted* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *carriers. If the two packets expose matching* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows D, D', matching* $\mathsf{StreamName}^{\uparrow}$ *window rows S, S', matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows R, R', and matching terminal* $\mathsf{Real}^{\uparrow}$ *seal rows B, B', then every completion-boundary read through I and every matching read through I' is forced onto the same tail-ledger route through T, H, C, P, N and T', H', C', P', N'. The comparison exports no second Real seal, quotient stream equality, selected limit, host real equality, ambient completeness datum, or bridge credential.*

Proof. Project both carriers to the displayed rows of Definition 1844.1. The assumptions identify the four analytic faces D, S, R, B and D', S', R', B' as matching dyadic, stream, regular-rational, and terminal Real-seal data.

Apply the four-face handoff theorem, Definition 1844.3, to each packet. A completion-boundary read cannot inspect T, H, C, P, N or T', H', C', P', N' until the matched faces have already been displayed in order.

Use Real-seal route determinacy, Definition 1844.4. The terminal seal route is already fixed by the matched four faces, so any remaining comparison can only replay the tail ledger, componentwise transport, continuation, provenance, and local naming rows.

Finally invoke non-escape, Definition 1844.5. Those residual rows contain no second seal coordinate, quotient stream equality, selected limit, host real equality, ambient completeness datum, or bridge credential. Hence the tail-ledger route is determined by the matching D, S, R, B faces. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealCompletionInterface}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealCompletionInterface}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1844.1, Definition 1844.2, Definition 1844.3, Definition 1844.4, Definition 1844.5, and Definition 1844.6 bind the finite L10 interface obligations to $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealCompletionInterface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, stream-window exactness, regular-sequence handoff, Real-seal boundary, four-face handoff, Real-seal route determinacy, and non-escape proofs for BEDC.Derived.CauchyRealCompletionInterfaceUp.

Constructive story: A $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ packet is generated from a DyadicRatCore endpoint and tolerance row, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal row, a completion-boundary tail ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1845

A Concrete Naming Certificate for $\mathsf{CauchyCompletionKleisli}^{\uparrow}$

Following Mac Lane's language of Kleisli categories for monads, and Bishop–Bridges and Kelley completion terminology for Cauchy completion interfaces, the $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ packet names the finite consumer boundary where unit and bind routes of the L10 completion monad are handed to downstream completion readers. It is downstream of the completion monad of chapter 378, the unit certificate of chapter 1527, the multiplication certificate of chapter 1829, the stream surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38.

The packet is a consumer boundary rather than a construction of selected limits. It records exactly the rows by which a downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ reader can inspect a unit source, a bind source, a reassociated bind source, the shared schedule-window-observation-readback route, and the terminal seal.

Definition 1845.1 (Cauchy completion Kleisli carrier)**.** *A* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, B, A, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the unit source row inherited from* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*, displaying the constant regular-rational input packet before any bind consumer is read.*
2. *B is the bind source row inherited from* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*, displaying the source-family row used by a single Kleisli bind request.*
3. *A is the reassociated bind source row inherited from* $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$*, displaying the source route for two successive bind requests after reassociation.*
4. *S is the shared unary schedule row used by U, B, A before any finite window, dyadic observation, readback, or seal coordinate is inspected.*
5. *W is the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the schedule row.*
6. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and tolerance row attached to the same finite window.*
7. *R is the shared diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
8. *E is the public* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been displayed.*

9. *H records componentwise* hsame *transport for the unit source, bind source, reassociated bind source, schedule, window, dyadic observation, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Kleisli consumer request through $U, B, A, S, W, D, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, B, A, S, W, D, R, E, H, C, P, N$. The carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, hidden representatives, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1845.2 (Cauchy completion Kleisli NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *carrier $K = (U, B, A, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unit source, bind source, reassociated bind source, shared schedule, finite stream window, dyadic observation and tolerance row, diagonal readback, Real seal, transport, replay, provenance, and local naming row;*
2. *unit-source admission, requiring every left-unit or right-unit consumer read to pass through U before it can inspect S, W, D, R, E;*
3. *bind-source admission, requiring every single bind consumer read to pass through B before it can inspect the shared downstream rows;*
4. *reassociated-bind admission, requiring every two-step bind consumer read to pass through A and then the same S, W, D, R, E rows;*
5. *seal coherence, requiring the unit, bind, and reassociated bind routes to reach the same public* $\mathsf{Real}^{\uparrow}$ *seal row under componentwise* hsame *transport and displayed* Cont *replay;*
6. *non-escape, requiring P, N to source and name no quotient stream, selected limit, countable-choice selector, host equality, hidden representative, ambient completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

```
Lean4 ✓  BEDC.Derived.CauchyCompletionKleisliUp.TasteGate.CauchyCompletionKleisliUpTasteGate
single carrier alignment
```

Proof. Carrier admission is projection from Definition 1845.1. The accepted packet displays exactly the unit source, bind source, reassociated bind source, shared schedule, finite window, dyadic observation, diagonal readback, Real seal, and structural rows.

Unit-source admission is supplied by U and by the unit boundary of chapter 1527. A unit consumer reaches the downstream rows only after the constant source packet has been displayed.

Bind-source admission is supplied by B and by the bind route of chapter 378. The route fixes the source family before opening the shared schedule, window, dyadic observation, readback, and seal.

Reassociated-bind admission is supplied by A and by the multiplication boundary of chapter 1829. Two successive bind requests are exposed as one reassociated source route before the same downstream rows are read.

The seal coherence and non-escape clauses are structural. Componentwise transport H, continuation replay C, provenance P, and the local name N preserve and source only the displayed Kleisli packet, so no excluded completion coordinate is available. □

Theorem 1845.3 (Cauchy completion Kleisli source triangle)**.** *For every accepted* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *carrier* $K = (U, B, A, S, W, D, R, E, H, C, P, N)$*, the route that reads a unit source, then a bind source, and then a downstream completion consumer has the same displayed source triangle as the reassociated route. Both routes read only*

$$U, B, A \longmapsto S \longmapsto W \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N*, and neither route inserts a second completion source between bind and consumer.*

Proof. Start with the unit source row U. It displays the constant regular-rational source before any bind-facing coordinate is read.

Next attach the bind source row B. Since the carrier admits no row between B and the shared schedule, every single-bind consumer proceeds through S, W, D, R, E.

Now compare with the reassociated bind source row A. The carrier stores A as the only source route for two successive bind requests, and it is followed by the same schedule, window, observation, readback, and seal rows.

The structural rows H, C, P, N transport, replay, source, and name this same route. Thus the unit-bind-consumer path and the reassociated consumer path have the same displayed source triangle. □

Theorem 1845.4 (Cauchy completion Kleisli seal coherence)**.** *In every accepted* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *carrier, the unit route, the single-bind route, and the reassociated-bind route reach the same public* $\mathsf{Real}^{\uparrow}$ *seal row* E *after componentwise* hsame *transport of the source, schedule, window, dyadic-observation, diagonal-readback, replay, provenance, and local naming rows. The Kleisli consumer boundary therefore has one seal-facing route for unit, bind, and reassociated bind reads.*

Proof. Read the unit, bind, and reassociated bind rows U, B, A. Each row is followed in the carrier by the same schedule, finite window, dyadic observation, diagonal readback, and seal coordinates.

The componentwise transport row H compares these routes without introducing a detached representative. Displayed replay C preserves the order in which the downstream consumer reads the shared rows.

The public seal coordinate is the single row E. Since provenance P and the local name N source the same packet, every unit, bind, or reassociated bind presentation reaches the same seal-facing route. □

Theorem 1845.5 (Cauchy completion Kleisli nonescape)**.** *Every consumer-facing read of an accepted* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *carrier* $K = (U, B, A, S, W, D, R, E, H, C, P, N)$ *is exhausted by the displayed rows* $U, B, A, S, W, D, R, E, H, C, P, N$*. Thus the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}$ *boundary exports no quotient stream, selected limit, countable-choice selector, host equality, hidden representative, ambient completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

Proof. Project the accepted carrier. The tuple $K = (U, B, A, S, W, D, R, E, H, C, P, N)$ lists every coordinate admitted by the local Kleisli consumer surface.

The source rows U, B, A and the shared analytic rows S, W, D, R, E give the whole consumer-facing route. They contain the unit source, bind source, reassociated bind source, schedule, finite stream window, dyadic observation, diagonal readback, and public Real seal.

The structural rows do not enlarge the boundary. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row only preserve, replay, source, and name the same displayed packet.

Since no excluded object occurs as a coordinate or replay target, no consumer-facing read can escape to quotient streams, selected limits, countable choice, host equality, hidden representatives, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1845.1, Definition 1845.2, Definition 1845.3, Definition 1845.4, and Definition 1845.5 seed the finite Kleisli consumer boundary for the L10 completion monad.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host equality, countable choice, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-source admission, bind-source admission, reassociated-bind admission, source-triangle, seal-coherence, and non-escape proofs for BEDC.Derived.CauchyCompletionKleisliUp.

Constructive story: A $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ packet is generated from a unit source row, a bind source row, a reassociated bind source row, a shared schedule, finite StreamName windows, DyadicRatCore observations, RegSeqRat diagonal readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1846

A Concrete Naming Certificate for $\mathsf{CauchyPrecompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley constructive completion, the $\mathsf{CauchyPrecompletion}^{\uparrow}$ packet collects displayed regular Cauchy names before quotient, separated completion, or ambient completeness is read. It sits after the dyadic rational endpoint and tolerance surface of chapter 291, the finite stream windows of chapter 126, and the regular rational readback of chapter 301, while remaining upstream of the terminal Real seal of chapter 38, the completion-embedding interface of chapter 1512, the separated completion boundary of chapter 1612, and the constructive metric completion surface of chapter 1556.

The packet is a pre-separated collection boundary. It names the finite rows a completion consumer may inspect when assembling regular Cauchy data, but it does not quotient names, select representatives, invoke countable choice, identify host reals, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, close compactness by open covers, or provide a standard bridge.

Definition 1846.1 (CauchyPrecompletion carrier)**.** *A* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, R, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the source window family, listing the finite* $\mathsf{StreamName}^{\uparrow}$ *requests whose displayed entries are admitted before any quotient or separated completion step.*
2. *R is the regular Cauchy readback row, carrying the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence read from those source windows.*
3. *D is the dyadic tolerance ledger, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints and tolerance budgets used by W and R.*
4. *S is the finite stream-observation row, exposing only the displayed stream entries attached to W, D, R.*
5. *E is the completion-embedding handoff row, naming the downstream* $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, *and* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *consumers without giving them a quotient, a selected representative, or a completeness proof.*
6. *H records componentwise* hsame *transport for the source-window, regular-readback, dyadic-ledger, stream-observation, handoff, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a precompletion request through W, R, D, S, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyPrecompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPrecompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, S, E, H, C, P, N$. The carrier contains no quotient completion row, selected representative, countable-choice schedule, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, open-cover compactness datum, Lean verification credential, or standard bridge.*

Lean4 ✓ `BEDC.Derived.CauchyPrecompletionUp.TasteGate.CauchyPrecompletionTasteGate single carrier alignment`

Theorem 1846.2 (CauchyPrecompletion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *carrier $Q = (W, R, D, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyPrecompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source window family, regular Cauchy readback, dyadic tolerance ledger, finite stream observations, completion-embedding handoff, transport, replay, provenance, and local naming row;*
2. *source-window locality, requiring every precompletion request to read only the displayed finite windows in W before it can inspect the regular Cauchy readback;*
3. *regular-readback attachment, requiring R to be tied to the same source windows and dyadic ledger rather than to a second stream name or host representative;*
4. *dyadic-tolerance exactness, requiring D to provide the endpoint and tolerance budget used by both the source windows and the readback row;*
5. *finite-observation exactness, requiring S to expose only displayed stream observations attached to W, R, D;*
6. *downstream handoff, requiring E to name only completion-embedding, separated-completion, and constructive-metric-completion consumers without exporting their quotient, representative, completeness, compactness, or bridge data;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no row outside $W, R, D, S, E, H, C, P, N$.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1846.1. The accepted packet displays exactly the source window family, regular Cauchy readback, dyadic tolerance ledger, finite stream observations, completion-embedding handoff, and structural rows.

Source-window locality and regular-readback attachment are forced by the order of the carrier. A precompletion request first reads W, then obtains the regular rational readback R attached to the same windows rather than to a second stream name or host representative.

Dyadic-tolerance and finite-observation exactness are supplied by D and S. The dyadic ledger records the endpoint and tolerance budget for both the source windows and the readback row, while S exposes only the displayed observations tied to that same data.

The downstream handoff is the row E. It names completion-embedding, separated-completion, and constructive-metric-completion consumers as later readers, but it has no coordinate from

which to export a quotient completion, selected representative, completeness theorem, open-cover compactness datum, or bridge credential.

Finally, non-escape is exhausted by H, C, P, N. These rows transport, replay, source, and name the displayed precompletion packet only, so no local obligation is stored outside $W, R, D, S, E, H, C, P, N$. □

Theorem 1846.3 (CauchyPrecompletion regular handoff)**.** *Every completion-facing read of an accepted* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *carrier* $Q = (W, R, D, S, E, H, C, P, N)$ *factors through the displayed regular Cauchy route*

$$W, D, S \longmapsto R \longmapsto E$$

before it can be replayed, transported, sourced, or locally named. Consequently the packet hands regular Cauchy names to completion interfaces without quotienting them, selecting representatives, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $Q = (W, R, D, S, E, H, C, P, N)$. The source window family W is the only row from which a completion-facing request may obtain stream entries.

The read then exposes the shared dyadic tolerance ledger D and the finite observation row S. These rows fix the endpoint budget and the displayed stream observations before any regular readback is used.

The regular Cauchy readback R is therefore reached through W, D, S. Since R is attached to the same displayed data, the packet has no coordinate for a second stream name, quotient class, selected representative, choice schedule, or host real equality.

The handoff row E names downstream completion consumers only after the regular route has been displayed. Transport, replay, provenance, and local naming preserve that same route, so the handoff remains a pre-separated regular Cauchy boundary rather than an ambient completeness assertion. □

Theorem 1846.4 (CauchyPrecompletion nonescape)**.** *Every exported comparison, replay, provenance read, or local name of an accepted* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *carrier* $Q = (W, R, D, S, E, H, C, P, N)$ *is exhausted by the displayed rows* $W, R, D, S, E, H, C, P, N$*. In particular, the packet exports no quotient completion, selected representative, countable-choice selector, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, open-cover compactness principle, Lean verification marker, or bridge credential.*

Proof. Project the accepted packet. Its classifier compares carriers only through the tuple $W, R, D, S, E, H, C, P, N$ named in Definition 1846.1.

The analytic rows are W, R, D, S, E. They display the source windows, regular readback, dyadic tolerance ledger, finite observations, and downstream handoff surface, without adding a quotient row or an ambient completeness coordinate.

The structural rows do not enlarge the boundary. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row only preserve, replay, source, and name the same displayed packet.

Since no excluded object appears as a coordinate or replay target, no exported comparison, provenance read, or local name can escape to quotient completion, selected representatives, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPrecompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyPrecompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1846.1, Definition 1846.2, Definition 1846.3, and Definition 1846.4 seed the finite pre-separated regular Cauchy collection boundary over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, SeparatedCompletion$^{\uparrow}$, and ConstructiveMetricCompletion$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyPrecompletion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyPrecompletionUp.TasteGate.CauchyPrecompletionTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, ambient Real$^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window locality, regular-readback attachment, dyadic-tolerance exactness, finite-observation exactness, completion-embedding handoff, regular handoff, and non-escape proofs for BEDC.Derived.CauchyPrecompletionUp.

Constructive story: A CauchyPrecompletion$^{\uparrow}$ packet is generated from a source window family, regular Cauchy readback, DyadicRatCore tolerance ledger, finite StreamName observations, completion-embedding handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1847

A Concrete Naming Certificate for $\mathsf{CauchySpace}^{\uparrow}$

Following Kelley and Bishop and Bridges, the $\mathsf{CauchySpace}^{\uparrow}$ packet records the finite L10 interface at which filter Cauchy data, uniform entourage data, regular rational Cauchy names, and Real completion consumers can be read through one NameCert surface. It refines the filter route of chapter 333, the entourage route of chapter 339, and the regular rational readback of chapter 301, while remaining upstream of the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 and the pre-separated completion boundary of chapter 1846.

The packet is an interface certificate rather than a completion theorem. It names the finite rows that let later metric and completion chapters cite one displayed Cauchy-space boundary, without asking those chapters to reassemble the filter, uniform, regular-sequence, and Real-consumer handoff separately.

Definition 1847.1 (CauchySpace carrier)**.** *A* $\mathsf{CauchySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, U, R, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, listing the finite filter requests and finite stability witnesses that are allowed to be read by the Cauchy-space interface.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, listing the finite entourage requests, refinements, and composition witnesses used to test the same Cauchy data.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row attached to the displayed filter and entourage data.*
4. *Q is the precompletion row, pointing to the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *packet that accepts the displayed regular Cauchy name before quotienting or representative selection.*
5. *T is the Real-consumer row, naming the downstream* $\mathsf{Real}^{\uparrow}$ *completion consumers allowed to read F, U, R, Q without importing ambient completeness.*
6. *H records componentwise* hsame *transport for the filter, uniform, regular-name, precompletion, Real-consumer, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Cauchy-space request through F, U, R, Q, T, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyPrecompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}$ *row.*

The carrier contains no quotient completion row, selected representative, countable-choice schedule, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, open-cover compactness datum, Lean verification credential, or standard bridge.*

Lean4 (definition) `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier`
Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceTasteGate single carrier alignment`

Definition 1847.2 (CauchySpace classifier)**.** *Two accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carriers*

$$S = (F, U, R, Q, T, H, C, P, N) \quad \textit{and} \quad S' = (F', U', R', Q', T', H', C', P', N')$$

are classified as the same Cauchy-space packet exactly when the displayed component comparisons hold:

$$F\mathsf{hsame}F', \quad U\mathsf{hsame}U', \quad R\mathsf{hsame}R', \quad Q\mathsf{hsame}Q', \quad T\mathsf{hsame}T',$$
$$H\mathsf{hsame}H', \quad C\mathsf{hsame}C', \quad P\mathsf{hsame}P', \quad N\mathsf{hsame}N'.$$

No classifier branch may inspect quotient completions, selected representatives, countable-choice selectors, host real equality, ambient completeness, open-cover compactness, Lean verification, or bridge credentials.

Lean4 (definition) `BEDC.Derived.CauchySpaceUp.CauchySpaceClassifier`

Theorem 1847.3 (CauchySpace filter stability)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier* $S = (F, U, R, Q, T, H, C, P, N)$*, every exported filter comparison or replay factors through the displayed Cauchy-filter row F and is stable under the componentwise transport recorded by H. In particular, changing the surrounding uniform, regular-name, precompletion, or Real-consumer rows by displayed* hsame *transport cannot create a new filter obligation outside the carrier rows.*

Proof. Project the accepted carrier. The only row whose coordinates can answer a filter request is F, so carrier admission places every exported filter comparison inside the displayed Cauchy-filter boundary.

The transport row H is componentwise. Its filter component transports F to an hsame filter row while leaving the remaining rows available only as later readers, not as additional filter witnesses.

Replay by C follows the same displayed request route. Since C, P, N replay, source, and name the carrier S rather than an external filter object, no replay step can introduce a second filter obligation.

The remaining rows U, R, Q, T may consume the stabilized filter data, but none contains an independent filter classifier. Thus every exported filter comparison is exhausted by F and its displayed transport through H. □

Theorem 1847.4 (CauchySpace filter-uniform compatibility)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier* $S = (F, U, R, Q, T, H, C, P, N)$*, any consumer that compares filter Cauchy data with a uniform entourage test must first read F and U through the same componentwise* hsame *transport row H. Only after that shared transport may the consumer pass through R, Q, T, C, P, N before reaching any* $\mathsf{Real}^{\uparrow}$ *consumer. The route refuses quotient completion, selected representatives, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier filter uniform compatibility`

Proof. Use the carrier of Definition 1847.1 and the classifier of Definition 1847.2. The rows F and U are separate visible coordinates, but their comparison is classified only after the componentwise transport row H has carried both sides of the packet.

Apply the filter-stability result of Definition 1847.3. It keeps every filter read inside F and its H-transported copy, so an entourage test cannot introduce an independent filter witness.

The uniform read is then aligned with the same transported packet. The regular-name row R, precompletion row Q, Real-consumer row T, replay row C, provenance row P, and local naming row N are downstream readers of that aligned finite packet.

Since no carrier row stores a quotient completion, selected representative, countable-choice schedule, host real equality test, or ambient completeness datum, the shared filter-uniform comparison reaches $\mathsf{Real}^{\uparrow}$ consumers only through the displayed finite route. □

Theorem 1847.5 (CauchySpace uniform regular-sequence interface)**.** *Let $S = (F, U, R, Q, T, H, C, P, N)$ be an accepted $\mathsf{CauchySpace}^{\uparrow}$ carrier. Every regular rational name exported by S is read through the finite interface*

$$F, U \longmapsto R$$

where F supplies the Cauchy-filter stability data and U supplies the uniform entourage tests. The packet therefore exposes regular Cauchy names to completion consumers without selecting representatives or invoking countable choice.

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier uniform regseq interface`

Proof. Start with an exported regular-name read. Carrier admission exposes the filter row F and the uniform row U before the regular rational row R can be inspected.

The filter row supplies the finite stability witnesses, while the uniform row supplies the finite entourage requests and refinements. These two rows are the only analytic inputs allowed upstream of R.

The regular rational row is attached to the displayed pair F, U. Hence a downstream reader obtains R as a regular Cauchy name certified by finite filter and entourage data, not as a chosen representative of a quotient class.

Transport, replay, provenance, and local naming preserve the same route. Since no accepted row stores a choice schedule or representative selector, the interface from uniform Cauchy data to regular rational names remains constructive and finite. □

Theorem 1847.6 (CauchySpace Real completion consumer boundary)**.** *Every $\mathsf{Real}^{\uparrow}$ completion consumer named by an accepted $\mathsf{CauchySpace}^{\uparrow}$ carrier $S = (F, U, R, Q, T, H, C, P, N)$ reads only the displayed route*

$$F, U, R \longmapsto Q \longmapsto T$$

before structural replay, provenance, or local naming. Consequently the packet exports a Cauchy-space consumption boundary, not quotient completion, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier real completion consumer`

Proof. The consumer row T is downstream of the analytic rows. A $\mathsf{Real}^{\uparrow}$ completion consumer first reads the finite filter row F, the finite uniform row U, and the attached regular rational name row R.

The precompletion row Q is then the only permitted handoff into completion-facing data. It accepts the displayed regular Cauchy name while still excluding quotienting, representative selection, host equality, and ambient completeness.

The consumer row T names later $\mathsf{Real}^{\uparrow}$ readers of that precompletion handoff. It has no coordinate from which to export a real-equality theorem, a completeness theorem, an open-cover compactness principle, or a bridge credential.

Finally, H, C, P, N preserve, replay, source, and name this same finite route. Since each structural row is attached to F, U, R, Q, T, every exported completion-consumer read remains inside the displayed Cauchy-space boundary. □

Theorem 1847.7 (CauchySpace Real completion provenance route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion read exported by an accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier can be named only after the displayed route*

$$F, U \longmapsto R \longmapsto N \longmapsto P$$

has exposed the filter-uniform replay, regular rational readback, local naming row, and provenance row. Thus the Real completion consumer boundary feeds the local $\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}$ *provenance route, rather than a detached completion object or selected host representative.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier real completion provenance route`

Proof. Start with Definition 1847.6. It places the Real-facing read downstream of the displayed filter-uniform route and the regular rational readback.

The local naming row N is then the only accepted row that names this Real-facing read for the packet. The provenance row P records the source of that named read, so a consumer cannot replace the route by a quotient completion, selected representative, host equality, or ambient completeness datum.

Because H transports the same displayed packet and C replays only that route, the resulting provenance read remains attached to F, U, R, N, P and exports no external completion object. □

Theorem 1847.8 (CauchySpace filter-uniform Real consumer obligation)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier* $S = (F, U, R, Q, T, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *consumer named by* T *must first traverse the shared filter and uniform rows* F, U *together with the componentwise transport row* H*. Only after that shared route is displayed may the consumer read the regular-name row* R*, the precompletion row* Q*, the Real-consumer row* T*, and the structural replay, provenance, and naming rows* C, P, N*.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier filter uniform real consumer obligation`

Proof. Project the carrier in Definition 1847.1. The Real-consumer row T is not an independent analytic source; it is downstream of the finite filter row, the uniform entourage row, the regular rational row, and the precompletion row.

Use the filter-uniform compatibility result of Definition 1847.4. It forces any comparison between filter Cauchy data and uniform entourage data through F, U, H before a later row can consume the comparison.

Thus a Real-facing read reaches R, Q, T only after the shared filter-uniform transport has been displayed. The support rows C, P, N replay, source, and name this same route, so no Real consumer can bypass F, U, H. □

Theorem 1847.9 (CauchySpace precompletion nonescape obligation)**.** *Let $S = (F, U, R, Q, T, H, C, P, N)$ be an accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier. The precompletion row Q can export only the displayed handoff from the regular rational name R to the Real-consumer row T; it cannot export quotient completion, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read Q through Definition 1847.6. The theorem places Q between the regular-name row R and the Real-consumer row T, so Q is a handoff row rather than a completed quotient object.

The excluded principles are absent from the carrier inventory in Definition 1847.1. Since neither Q nor the structural rows H, C, P, N store those principles, replay and naming cannot introduce them.

Consequently any precompletion consumer receives only the finite route $F, U, R, Q, T, H, C, P, N$ and no stronger completion principle. □

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier precompletion nonescape obligation`

Theorem 1847.10 (CauchySpace regular-name handoff obligation)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *carrier $S = (F, U, R, Q, T, H, C, P, N)$, the row R is the sole* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff from finite filter and uniform data. Any completion-facing read must obtain its regular rational name through $F, U \longmapsto R$ before the rows Q and T can be inspected.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier regular name handoff obligation`

Proof. Use the uniform regular-sequence interface of Definition 1847.5. It states that exported regular rational names are attached to the displayed filter row F and uniform row U.

No second regular-name coordinate appears in the carrier. The rows Q and T are downstream consumers of R, while H, C, P, N only transport, replay, source, and name the existing route.

Therefore every completion-facing handoff reads R as the unique $\mathsf{RegSeqRat}^{\uparrow}$ row generated from the finite filter-uniform data, before any precompletion or Real-consumer row is exposed. □

Definition 1847.11 (CauchySpace local filter carrier)**.** *A* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier is a finite* BHist *subpacket*

$$L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$$

inside an accepted carrier $S = (F, U, R, Q, T, H, C, P, N)$ such that F_0 lists the local $\mathsf{CauchyFilter}^{\uparrow}$ *requests, F_1 lists their finite meet and tail-stability witnesses, U_0 lists the matching* $\mathsf{UniformSpace}^{\uparrow}$ *entourage requests, R_0 is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from those local requests, T_0 is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer handoff, H_0 records componentwise* hsame *transport for all displayed rows, C_0 records the* Cont *replay route, P_0 records* Pkg *provenance, and N_0 is the local* $\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.CauchySpaceUp.CauchySpaceLocalFilterCarrier`

Theorem 1847.12 (CauchySpace RegSeqRat uniform obligation)**.** *Let $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$ be a* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exported by R_0 is generated only after the local* $\mathsf{CauchyFilter}^{\uparrow}$ *rows F_0, F_1 and the* $\mathsf{UniformSpace}^{\uparrow}$ *row U_0 have been transported componentwise by H_0 and replayed through C_0. The route therefore exposes a regular rational Cauchy name from finite local-filter and uniform-entourage rows, not from quotient completion, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier regseqrat uniform obligation`

Proof. Project the local filter carrier. The rows F_0 and F_1 contain the finite filter requests, finite meets, and tail-stability data, while U_0 contains the matching uniform-entourage requests.

Transport these rows by the corresponding components of H_0. Since H_0 is componentwise, it moves the local filter data and uniform data together without adding a second filter object or a second uniform-space object.

Replay the transported route by C_0. The regular rational readback R_0 is reached only after this replay, so the resulting $\mathsf{RegSeqRat}^{\uparrow}$ row is generated from the displayed local Cauchy and uniform rows.

The remaining rows T_0, P_0, N_0 consume, source, and name the same route. None stores a quotient completion, selected representative, choice schedule, host equality test, or ambient completeness datum. □

Theorem 1847.13 (CauchySpace completion handoff obligation)**.** *For every* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$*, every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer named by* T_0 *receives the handoff*

$$F_0, F_1, U_0 \longmapsto R_0 \longmapsto T_0$$

through the displayed hsame *and* Cont *rows. The consumer receives no quotient-completion object, no selected Cauchy representative, no host real equality, no countable-choice selector, and no ambient completeness theorem from the Cauchy-space packet.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier completion handoff obligation`

Proof. Begin with the consumer row T_0. By Definition 1847.11, T_0 is downstream of the local filter rows, the uniform row, and the regular rational readback row.

Use Definition 1847.12 to obtain the generated $\mathsf{RegSeqRat}^{\uparrow}$ row R_0 from F_0, F_1, U_0 after componentwise transport and replay. This places R_0 between the local Cauchy-space data and any completion-facing consumer.

The displayed route then passes to T_0 and is recorded by P_0 and N_0. Since the carrier inventory contains no quotient-completion row, selected representative, choice schedule, host equality, or ambient completeness row, the completion handoff remains the finite route shown above. □

Theorem 1847.14 (CauchySpace filter-uniform completion nonescape)**.** *For every* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$*, every completion-facing CauchySpace read must traverse the shared* $\mathsf{CauchyFilter}^{\uparrow}$ *rows* F_0, F_1*, the* $\mathsf{UniformSpace}^{\uparrow}$ *row* U_0*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R_0*, the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer row* T_0*, the componentwise* hsame *transport row* H_0*, the* Cont *replay row* C_0*, the* Pkg *provenance row* P_0*, and the local* $\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}$ *row* N_0 *before exposing a completion boundary. The route exports no quotient completion, selected representative, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, Lean credential, or bridge claim.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier filter uniform completion nonescape`

Proof. Project the local carrier of Definition 1847.11. Its visible rows are exactly $F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0$, so every completion-facing read is sourced by this local inventory.

Use Definition 1847.12 for the filter-uniform compatibility and regular-name handoff. It places the $\mathsf{RegSeqRat}^{\uparrow}$ row R_0 downstream of the $\mathsf{CauchyFilter}^{\uparrow}$ rows F_0, F_1, the $\mathsf{UniformSpace}^{\uparrow}$ row U_0, componentwise transport, and displayed replay.

Use Definition 1847.13 for the completion handoff. The consumer row T_0 is reached only after R_0 and is then recorded by the same transport, replay, provenance, and local naming rows.

The excluded principles do not occur in the local carrier. Hence the completion boundary exposes only the displayed finite route and no quotient completion, selected representative, choice principle, host equality, ambient completeness, compactness, Lean credential, or bridge claim. □

Theorem 1847.15 (CauchySpace NameCert ledger)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchySpace}^\uparrow}$ *ledger for Definition 1847.11 consists of the finite* BHist *rows* F_0, F_1, U_0, R_0, T_0*, their componentwise* hsame *transports* H_0*, their* Cont *replay route* C_0*, and their* Pkg *provenance* P_0*. Any downstream* $\mathsf{CauchyFilter}^\uparrow$*,* $\mathsf{UniformSpace}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{Real}^\uparrow$*, or* $\mathsf{UniformCompletion}^\uparrow$ *read must factor through this ledger before inspecting a completion-facing row.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier namecert ledger`

Proof. Project the local carrier and record its rows in order. The first block, F_0, F_1, U_0, is the local Cauchy-filter and uniform-entourage surface; the second block, R_0, T_0, is the regular-name and completion-consumer surface.

The transport row H_0 and replay row C_0 bind the two blocks into one displayed route. Apply Definition 1847.12 for the regular-name handoff and Definition 1847.13 for the completion handoff.

Finally, P_0 and N_0 source and name exactly those rows. Thus the ledger contains every permitted read by the listed downstream packets and excludes any hidden quotient, selected representative, host equality, choice, or ambient completeness coordinate. □

Theorem 1847.16 (CauchySpace obligation coverage route)**.** *For every accepted* $\mathsf{CauchySpace}^\uparrow$ *local carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$ *of Definition 1847.11, every downstream Cauchy-completion consumer must first expose the* $\mathsf{CauchyFilter}^\uparrow$ *rows* F_0, F_1*, the* $\mathsf{UniformSpace}^\uparrow$ *entourage row* U_0*, the* $\mathsf{RegSeqRat}^\uparrow$ *regular-name row* R_0*, the* $\mathsf{CauchyPrecompletion}^\uparrow$ *handoff inside* T_0*, the* $\mathsf{Real}^\uparrow$ *completion-consumer row inside* T_0*, the componentwise* hsame *transport row* H_0*, the* Cont *replay row* C_0*, the* Pkg *provenance row* P_0*, and the local* $\mathsf{NameCert}_{\mathsf{CauchySpace}^\uparrow}$ *row* N_0 *before reading any completion boundary. The route exports no quotient completion, selected representative, host equality, countable choice, ambient* $\mathsf{Real}^\uparrow$ *completeness, open-cover compactness, Lean credential, or bridge claim.*

Proof. Project the local carrier L. Its displayed coordinates already list the Cauchy-filter rows, uniform row, regular-name row, completion-consumer handoff, transport, replay, provenance, and local naming row.

Apply Definition 1847.14 to force completion-facing reads through the shared filter and uniform surface. Apply Definition 1847.12 to place the $\mathsf{RegSeqRat}^\uparrow$ row downstream of that surface and upstream of any completion consumer.

Apply Definition 1847.13 for the handoff into the $\mathsf{CauchyPrecompletion}^\uparrow$ and $\mathsf{Real}^\uparrow$ consumer row. The remaining coordinates H_0, C_0, P_0, N_0 provide the componentwise transport, replay, provenance, and local NameCert rows for the same route.

The excluded principles are absent from the carrier and from the three cited routes. Hence every downstream Cauchy-completion consumer sees only the displayed finite obligation surface before any completion boundary is read. □

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier obligation coverage route`

Theorem 1847.17 (CauchySpace precompletion factorization)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *local carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$ *of Definition 1847.11, every* $\mathsf{Real}^{\uparrow}$ *completion consumer factors through the displayed precompletion route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyPrecompletion}^{\uparrow}$$

before it can read T_0, H_0, C_0, P_0, N_0*. Concretely, the consumer first exposes the filter rows* F_0, F_1*, then the uniform-entourage row* U_0*, then the regular rational name row* R_0*, then the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *handoff inside* T_0*, and only afterward the* $\mathsf{Real}^{\uparrow}$ *completion boundary inside the same row.*

The factorization exports no quotient completion, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceLocalFilterCarrier precompletion factorization`

Proof. Project the local carrier. Its public rows are ordered as F_0, F_1 for the $\mathsf{CauchyFilter}^{\uparrow}$ surface, U_0 for the $\mathsf{UniformSpace}^{\uparrow}$ surface, R_0 for the $\mathsf{RegSeqRat}^{\uparrow}$ regular-name readback, and T_0 for the precompletion and Real completion consumer handoff.

Apply Definition 1847.14. It keeps every completion-facing read downstream of the shared Cauchy-filter and uniform-entourage surface and prevents a direct jump to a completed host object.

Apply Definition 1847.12. The regular rational readback R_0 is generated from F_0, F_1, U_0 under the componentwise transport and displayed replay rows, so the readback cannot be replaced by a selected representative.

Finally apply Definition 1847.13 and Definition 1847.16. The handoff row T_0 exposes the $\mathsf{CauchyPrecompletion}^{\uparrow}$ boundary before the $\mathsf{Real}^{\uparrow}$ consumer boundary, while H_0, C_0, P_0, N_0 transport, replay, source, and name the same finite route. Hence quotient completion, selected representatives, countable choice, host equality, ambient completeness, and bridge credentials remain outside the carrier. □

Theorem 1847.18 (CauchySpace public consumer boundary)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$*, the public CauchySpace boundary exposed to completion-facing consumers is exactly the route that combines the local filter carrier, the precompletion nonescape obligation, the regular-name handoff obligation, and the completion handoff obligation. A consumer must read* F_0, F_1, U_0*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R_0*, then the* $\mathsf{CauchyPrecompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff recorded in* T_0*, and finally the* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The public boundary exports no quotient completion, selected representative, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, Lean credential, or bridge claim.*

Proof. Start with Definition 1847.11. It lists the only local filter, uniform-entourage, regular-name, completion-consumer, transport, replay, provenance, and naming rows available to a public consumer.

Use Definition 1847.9 to keep the precompletion row from becoming a quotient completion or selected representative. Use Definition 1847.10 and Definition 1847.12 to force every regular-name read through the displayed filter and uniform rows before R_0 is consumed.

Use Definition 1847.13 and Definition 1847.17 for the completion-facing handoff. These theorems place T_0 downstream of R_0 and keep the $\mathsf{Real}^{\uparrow}$ or $\mathsf{UniformCompletion}^{\uparrow}$ consumer inside the displayed route.

The structural rows H_0, C_0, P_0, N_0 transport, replay, source, and name the same carrier. Since the excluded principles have no row in the local carrier or in the cited obligations, the public consumer boundary is exactly the finite route stated above. □

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier public consumer boundary`

Theorem 1847.19 (CauchySpace public completion interface)**.** *For every accepted* $\mathsf{CauchySpace}^{\uparrow}$ *local filter carrier* $L = (F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0)$*, any* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *consumer reading the public* $\mathsf{CauchySpace}^{\uparrow}$ *surface must pass through the local filter carrier, the precompletion nonescape route, the regular-name handoff, the completion handoff, and the public consumer boundary already displayed in this chapter. Thus the completion-facing interface exposes only the finite rows* $F_0, F_1, U_0, R_0, T_0, H_0, C_0, P_0, N_0$ *and exports no quotient completion, selected representative, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, Lean credential, or bridge claim.*

Proof. Start with Definition 1847.11. It is the unique local inventory from which a completion-facing consumer may read filter rows, uniform rows, regular-name rows, completion handoff rows, transport, replay, provenance, and local naming.

Apply Definition 1847.9. This keeps the precompletion-facing read inside the displayed handoff row and blocks quotient completions, selected representatives, and host equality from entering the interface.

Next use Definition 1847.10. The regular-name read R_0 is reached only through the local filter and uniform rows, so a $\mathsf{Completion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{CauchyPrecompletion}^{\uparrow}$ consumer cannot bypass the displayed rational-name route.

Finally apply Definition 1847.13 and Definition 1847.18. The first theorem places the completion-facing row T_0 after the regular-name handoff, and the second repacks the entire route as the public boundary. Hence the stated public completion interface is exactly the same finite carrier surface. □

Lean4 ✓ `BEDC.Derived.CauchySpaceUp.CauchySpaceCarrier public completion interface`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1847.1, Definition 1847.2, Definition 1847.16, Definition 1847.17, Definition 1847.18, and Definition 1847.19 close the displayed finite CauchySpace public completion interface for downstream completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: The next grade requires Lean verification and bridge checking for the displayed public completion interface, without adding quotient completion, selected representatives, countable choice, or ambient completeness.

Constructive story: A $\mathsf{CauchySpace}^{\uparrow}$ packet is generated from a CauchyFilter row, UniformSpace entourage row, RegSeqRat regular-name row, CauchyPrecompletion handoff row, RealUp completion-consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1848

A Concrete Naming Certificate for $\mathsf{UniformEntourageRegularName}^{\uparrow}$

Following Kelley, Bourbaki, and Bishop–Bridges, the $\mathsf{UniformEntourageRegularName}^{\uparrow}$ packet records the finite L10 route by which uniform entourage data are read as regular rational names before any separated completion, representative selection, or host real equality is invoked. It sharpens the Cauchy-space interface of chapter 1847 by isolating the route from the $\mathsf{UniformSpace}^{\uparrow}$ entourage row of chapter 339 and the $\mathsf{CauchyFilter}^{\uparrow}$ requests of chapter 333 through finite $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126 into the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301. The same route then hands the result to the $\mathsf{CauchyPrecompletion}^{\uparrow}$ boundary of chapter 1846 and later $\mathsf{Real}^{\uparrow}$ consumers from chapter 38.

The packet is a naming certificate for the entourage-to-regular-name handoff. It does not assert that the handoff has already been quotient-completed, represented by a selected stream, identified with host real equality, or upgraded to ambient real completeness.

Definition 1848.1 (Uniform entourage regular-name carrier)**.** *A* $\mathsf{UniformEntourageRegularName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, F, W, R, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage-data row, listing the finite entourage requests, refinement witnesses, diagonal witnesses, and composition witnesses available to the route.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *request row, listing the finite filter requests and stability witnesses whose Cauchy behavior is tested against the entourage data in U.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the displayed entourage and Cauchy-filter requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same U, F, W data.*
5. *Q is the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *handoff row that accepts the displayed regular rational name before quotient completion or representative selection.*
6. *T is the* $\mathsf{Real}^{\uparrow}$ *consumer row, naming only downstream real-name readers of Q and R without importing ambient completeness.*
7. *H records componentwise* hsame *transport for the uniform, Cauchy-filter, stream-window, regular-readback, precompletion, Real-consumer, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an entourage-regular-name request through $U, F, W, R, Q, T, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformEntourageRegularName}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyPrecompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformEntourageRegularName}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, W, R, Q, T, H, C, P, N$ and the displayed routes among those rows. It contains no quotient completion row, selected representative, countable-choice schedule, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, open-cover compactness datum, Lean verification credential, or standard bridge.*

Lean4 ✓ `BEDC.Derived.UniformEntourageRegularNameUp.UniformEntourageRegularNameTasteGate single carrier alignment`

Theorem 1848.2 (Uniform entourage regular-name NameCert obligations)**.** *For every accepted* $\mathsf{UniformEntourageRegularName}^{\uparrow}$ *carrier*

$$E = (U, F, W, R, Q, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformEntourageRegularName}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the uniform entourage data, Cauchy-filter requests, finite stream windows, regular rational readback, precompletion handoff, Real-consumer row, transport, replay, provenance, and local naming row;*
2. *entourage locality, requiring every regular-name request to read only the displayed* $\mathsf{UniformSpace}^{\uparrow}$ *entourages in U before it can inspect a stream window;*
3. *Cauchy-filter compatibility, requiring the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row F to be the only filter source paired with the entourage tests in U;*
4. *finite-window readback, requiring W and R to expose a* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the same U, F route;*
5. *completion-facing handoff, requiring Q and T to name only the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers of that regular name;*
6. *non-escape, requiring H, C, P, N to transport, replay, source, and name no row outside $U, F, W, R, Q, T, H, C, P, N$.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1848.1. The accepted packet displays exactly the uniform, filter, window, readback, handoff, consumer, and structural rows.

Entourage locality is forced by the first coordinate U. Any regular-name request starts by reading the finite entourage data already accepted by the $\mathsf{UniformSpace}^{\uparrow}$ row, so no external uniform structure can be inspected.

Cauchy-filter compatibility is the role of F. The packet pairs the finite $\mathsf{CauchyFilter}^{\uparrow}$ requests with the same entourage data, and there is no second filter coordinate from which another Cauchy source could be read.

The finite-window and readback obligations are the rows W and R. They expose the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ regular rational name attached to U, F, not to a selected representative or host real.

The rows Q and T only hand the displayed regular name to $\mathsf{CauchyPrecompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. Finally H, C, P, N preserve, replay, source, and name the same finite carrier, so the listed rows exhaust the local certificate surface. □

Theorem 1848.3 (Uniform entourage RegSeqRat handoff)**.** *Every regular rational name exported by an accepted* $\mathsf{UniformEntourageRegularName}^{\uparrow}$ *carrier*

$$E = (U, F, W, R, Q, T, H, C, P, N)$$

factors through the displayed route

$$U, F \longmapsto W \longmapsto R$$

before it reaches precompletion or Real-facing consumers. Consequently the packet hands uniform Cauchy data to $\mathsf{RegSeqRat}^{\uparrow}$ *without selecting representatives, invoking countable choice, or asserting host real equality.*

Proof. Begin with an exported regular rational name. The carrier exposes the uniform entourage row U and the Cauchy-filter request row F before any stream observation is available.

The stream-window row W is therefore opened only by the displayed U, F data. It contains the finite observations permitted by the route and no selected infinite representative.

The readback row R repacks those observations as the $\mathsf{RegSeqRat}^{\uparrow}$ name attached to the same route. Thus the exported regular rational name is generated by the displayed entourage and filter data rather than by quotient completion or host equality.

Rows Q, T, H, C, P, N appear only after this readback. They hand the name to precompletion and Real-facing consumers, then preserve and name the route without adding countable choice or representative selection. □

Theorem 1848.4 (Uniform entourage CauchySpace boundary)**.** *Every* $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{CauchyPrecompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformEntourageRegularName}^{\uparrow}$ *carrier*

$$E = (U, F, W, R, Q, T, H, C, P, N)$$

reads only the displayed route

$$U, F, W, R \longmapsto Q, T \longmapsto H, C, P, N$$

before it treats the packet as completion-facing data. The packet therefore supplies a boundary between uniform entourage Cauchy data and regular rational names, not a quotient completion, selected representative, countable-choice principle, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, open-cover compactness theorem, Lean verification, or bridge check.*

Proof. Project the accepted carrier. The only analytic rows exposed before completion-facing data are U, F, W, R: uniform entourage data, Cauchy-filter requests, finite stream windows, and regular rational readback.

The precompletion and Real-facing rows are Q and T. They consume the displayed regular name and name later $\mathsf{CauchyPrecompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers, but neither row contains a quotient, selected representative, host equality, completeness proof, compactness proof, or bridge credential.

The structural rows H, C, P, N are attached to the same displayed route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming cannot enlarge the packet beyond its accepted coordinates.

Since no excluded object appears as a coordinate or replay target, every $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{CauchyPrecompletion}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer remains inside the finite uniform-entourage-to-regular-name boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEntourageRegularName}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEntourageRegularName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1848.1, Definition 1848.2, Definition 1848.3, and Definition 1848.4 seed the finite L10 route from $\mathsf{UniformSpace}^{\uparrow}$ entourage data and $\mathsf{CauchyFilter}^{\uparrow}$ requests through $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback to $\mathsf{CauchyPrecompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEntourageRegularName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage locality, Cauchy-filter compatibility, finite-window readback, RegSeqRat handoff, CauchySpace boundary, completion-facing handoff, and non-escape proofs for BEDC.Derived.UniformEntourageRegularNameUp.

Constructive story: A $\mathsf{UniformEntourageRegularName}^{\uparrow}$ packet is generated from UniformSpace entourage data, CauchyFilter requests, finite StreamName windows, RegSeqRat readback, CauchyPrecompletion handoff, RealUp consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1849

A Concrete Naming Certificate for $\mathsf{RationalRealEmbedding}^{\uparrow}$

Following Bishop and Bridges' construction of reals from rational Cauchy data, the $\mathsf{RationalRealEmbedding}^{\uparrow}$ packet names the constant-rational route from the arithmetic source $\mathsf{Rat}^{\uparrow}$ to the regular Cauchy and terminal real-name layers. It sits over the rational certificate of chapter 37, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet is the finite source boundary for rational constants inside the L10 completion route; it does not quotient streams, select limits, assert host real equality, invoke countable choice, or prove ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1849.1 (Rational-real embedding carrier)**.** *A* $\mathsf{RationalRealEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, S, R, D, T, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row, containing the rational constant together with the finite arithmetic source data allowed to be read by the embedding packet.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *window row: each displayed window reads the same rational source Q and contains no varying tail schedule.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *constant readback row, recording that every regular rational name coordinate is read back as the rational source from Q through the windows in S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the zero-tail and dyadic comparison budget used to certify that the constant readback satisfies every displayed regularity request.*
5. *T is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from Q, S, R, D, naming the terminal real-facing certificate for the rational constant.*
6. *H records componentwise* hsame *transport for the rational source, constant windows, regular readback, dyadic tolerance ledger, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a rational-real embedding request through Q, S, R, D, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalRealEmbedding}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $Q, S, R, D, T, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RationalRealEmbeddingUp.RationalRealEmbeddingTasteGate single carrier alignment`

Theorem 1849.2 (Rational-real embedding NameCert obligations)**.** *For every accepted* $\mathsf{RationalRealEmbedding}^{\uparrow}$ *carrier $E = (Q, S, R, D, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational source, constant stream windows, regular rational constant readback, dyadic tolerance ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *rational-source admission, requiring every exported constant to pass through the displayed* $\mathsf{Rat}^{\uparrow}$ *row Q;*
3. *constant-window admission, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in S to replay the same rational source and no independent tail schedule;*
4. *regular-readback admission, requiring R to expose the constant rational name through the displayed windows in S;*
5. *tolerance admission, requiring D to discharge the regularity requests by the zero-tail dyadic ledger attached to Q, S, R;*
6. *seal admission, requiring the* $\mathsf{Real}^{\uparrow}$ *row T to be read only after Q, S, R, D have been fixed;*
7. *non-escape, requiring H, C, P, N to export no quotient-stream equality, selected limit, countable-choice selector, host real equality, trichotomy datum, ambient completeness theorem, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1849.1. The accepted packet displays exactly the rational source, constant windows, regular readback, dyadic tolerance ledger, terminal real seal, and structural rows.

The rational-source and constant-window obligations are then forced by row order. A constant exported by the packet first reads Q, and every window in S is attached to that same rational source rather than to a separate stream tail.

The regular-readback and tolerance obligations use the next two rows. The row R reads the constant rational name through S, while D supplies the dyadic zero-tail ledger that answers the finite regularity tests.

The terminal seal obligation is downstream. The $\mathsf{Real}^{\uparrow}$ row T can be cited only after Q, S, R, D are already displayed, so the real-facing name is a seal of the finite constant route rather than a selected host limit.

The remaining rows are structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve and source the same displayed packet; none contains a quotient stream, selected limit, choice selector, host equality, trichotomy datum, ambient completeness theorem, Lean credential, or bridge credential. □

Theorem 1849.3 (Rational-real embedding real-seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{RationalRealEmbedding}^{\uparrow}$ *carrier* $E = (Q, S, R, D, T, H, C, P, N)$ *factors through the finite route*

$$Q, S, R, D \longmapsto T$$

before transport, replay, provenance, or local naming can be read. Consequently the embedding of a rational constant into the real-name layer is witnessed by a displayed constant StreamName route and a RegSeqRat readback, not by quotienting streams or selecting a limit.

Proof. Start with an exported real seal. The carrier has only one real-facing coordinate, namely T, and that row is listed after the rational source, constant-window, regular-readback, and dyadic-ledger rows.

The source row Q fixes the rational constant. The stream row S then opens only constant windows over Q, so no later step can introduce a variable Cauchy tail or an independent schedule.

The regular readback row R reads those windows as a constant regular rational name. The dyadic ledger D records the zero-tail tolerance facts needed by the displayed regularity requests.

Only after these four rows are fixed may T be read as a $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming preserve that same order, so the exported seal factors through Q, S, R, D and never through a quotient stream or selected limit. □

Theorem 1849.4 (Rational-real embedding constant-window coherence)**.** *For every accepted* $\mathsf{RationalRealEmbedding}^{\uparrow}$ *carrier* $E = (Q, S, R, D, T, H, C, P, N)$*, any two finite* $\mathsf{StreamName}^{\uparrow}$ *window reads exposed by* S *over inspected unary indices are classifier-equal through the same rational source row* Q*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* T *may consume only that constant-window surface.*

Proof. Project the accepted carrier. By the carrier definition, the row S is the constant $\mathsf{StreamName}^{\uparrow}$ window row attached to the rational source Q, and it contains no coordinate for a varying tail schedule.

Now compare two inspected unary-index windows in S. Each window reads the same displayed rational source row Q, so their classifier comparison is mediated by that source row rather than by an external stream representative.

Transport preserves the comparison componentwise through H. Replay through C can only cite the displayed route Q, S, R, D, T, H, P, N, so it reuses the same constant-window surface instead of creating a second tail schedule.

The downstream rows are therefore constrained by source order. The readback row R reads the constant windows already exposed by S, the tolerance row D evaluates the zero-tail ledger attached to that readback, and the seal row T is reached only after those rows have been fixed. □

Theorem 1849.5 (Rational-real embedding nonescape)**.** *Every rational-source, stream-window, regular-readback, dyadic-tolerance, or real-seal read of an accepted* $\mathsf{RationalRealEmbedding}^{\uparrow}$ *carrier* $E = (Q, S, R, D, T, H, C, P, N)$ *is exhausted by the displayed rows* $Q, S, R, D, T, H, C, P, N$*. Thus the local* $\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$ *boundary exports only the rational source, constant StreamName windows, RegSeqRat constant readback, DyadicRatCore tolerance ledger, RealUp seal row, componentwise transport, continuation replay, provenance, and local naming data.*

It exports no quotient-stream equality, selected limit, countable-choice selector, host real equality, trichotomy datum, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

Proof. Project the accepted carrier. The tuple $E = (Q, S, R, D, T, H, C, P, N)$ lists every coordinate admitted by the local $\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$ surface.

The analytic route is Q, S, R, D, T. It fixes a rational source, opens constant finite windows, reads them as a regular rational name, supplies the dyadic tolerance ledger, and then seals the result as a $\mathsf{Real}^{\uparrow}$ name.

The structural route is H, C, P, N. These rows transport, replay, source, and name the same displayed carrier; they do not add analytic coordinates or external equality principles.

Since no excluded object occurs as a coordinate or replay target, no read can escape to quotient-stream equality, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or a bridge credential. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 1849.1, Definition 1849.2, Definition 1849.3, Definition 1849.4, and Definition 1849.5 seed the finite rational-constant embedding route from $\mathsf{Rat}^{\uparrow}$ through constant $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalRealEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-source admission, constant-window admission, regular-readback admission, dyadic-tolerance admission, real-seal factorization, transport stability, and non-escape proofs for BEDC.Derived.RationalRealEmbeddingUp.

Constructive story: A $\mathsf{RationalRealEmbedding}^{\uparrow}$ packet is generated from a RatUp source row, constant StreamName windows, RegSeqRat constant readback, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1850

A Concrete Naming Certificate for $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ packet records the finite BEDC boundary at which a located real interval and a regular Cauchy real name can be read together. It sits between the regular completion route of chapter 1639, the located interval surface of chapter 1709, the rational approximation window of chapter 1668, and the four-face completion interface of chapter 1844. The packet gives Real and Completion consumers a located Cauchy interface; it is not a completeness theorem for $\mathsf{Real}^{\uparrow}$, a quotient-stream construction, a selected limit, countable choice, host real equality, or a standard bridge.

Definition 1850.1 (Bishop located Cauchy real carrier)**.** *A* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, B, D, S, R, A, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, supplying lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, radius compatibility, locator reads, locatedness moduli, and the metric bracket route.*
2. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, carrying the regular Cauchy completion-facing route through dyadic endpoints, stream windows, regular rational readback, and the terminal Real seal.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger shared by the interval radius budget and the regular Cauchy tail budget.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the Cauchy name before any locatedness request reads the terminal seal.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same stream windows and dyadic tolerance ledger.*
6. *A is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *row, displaying the rational approximation window used to compare the Cauchy readback with the located interval endpoints.*
7. *W is the located-Cauchy witness ledger, recording that the regular Cauchy tail route and the located interval route use the same dyadic tolerance request before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read.*

8. *H records componentwise* hsame *transport for the interval, completion boundary, dyadic tolerance, stream window, regular readback, approximation, witness, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a located Cauchy request through $I, B, D, S, R, A, W, H, P, N$.*

10. *P is the* Pkg *provenance over* BishopLocatedCauchyReal$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, BishopRegularCauchyCompletion$^{\uparrow}$ DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealRationalApproximation$^{\uparrow}$, CauchyRealCompletionInterface$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BishopLocatedCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, D, S, R, A, W, H, C, P, N$. The carrier has no coordinate for Real$^{\uparrow}$ *completeness, quotient stream equality, selected limits, countable choice, host real equality, trichotomy, compactness, or bridge checking.*

Theorem 1850.2 (Bishop located Cauchy real NameCert obligations)**.** *For every accepted* BishopLocatedCauchyRe *carrier $L = (I, B, D, S, R, A, W, H, C, P, N)$, the local* NameCert$_{\text{BishopLocatedCauchyReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval row, regular Cauchy completion boundary row, dyadic tolerance ledger, stream window, regular rational readback, rational approximation row, located-Cauchy witness ledger, transport, replay, provenance, and local naming row;*

2. *located-interval admission, requiring every endpoint, radius, locator, locatedness, and metric-bracket request to pass through the displayed* LocatedRealInterval$^{\uparrow}$ *row I;*

3. *regular-Cauchy admission, requiring every completion-facing read to pass through B, D, S, R before the terminal* Real$^{\uparrow}$ *seal can be cited;*

4. *approximation compatibility, requiring A to compare the regular rational readback with the interval endpoint data under the same dyadic tolerance row D;*

5. *witness stability, requiring W, H, C to preserve the shared located-Cauchy route componentwise and replay it through displayed continuations only;*

6. *non-escape, requiring P, N to export no* Real$^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy, compactness principle, or bridge credential.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BishopLocatedCauchyRealUp.BishopLocatedCauchyRealTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1850.1. The accepted packet displays exactly the located interval row, regular Cauchy completion boundary, shared dyadic tolerance ledger, finite stream window, regular rational readback, rational approximation row, witness ledger, and structural rows.

Located-interval admission is supplied by I. Endpoint, radius, locator, locatedness, and metric-bracket requests have no route except the displayed LocatedRealInterval$^{\uparrow}$ carrier of chapter 1709.

Regular-Cauchy admission is supplied by B, D, S, R. The completion-facing read first fixes the dyadic tolerance ledger, then opens the stream window and regular rational readback before the terminal Real$^{\uparrow}$ seal can be used.

Approximation compatibility is the row A. It compares the regular readback with the located interval endpoint data under the same tolerance row D, so the packet does not introduce a detached rational approximation or a host real representative.

The remaining obligations are structural. The witness ledger W, componentwise transport H, continuation replay C, provenance P, and local name N preserve and source only the displayed located-Cauchy route; none contains a completeness theorem, quotient equality, selected limit, choice selector, host equality, trichotomy, compactness principle, or bridge credential. □

Theorem 1850.3 (Bishop located Cauchy real regular-Cauchy handoff)**.** *Every completion-facing read of an accepted* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *carrier* $L = (I, B, D, S, R, A, W, H, C, P, N)$ *factors through the regular Cauchy route*

$$B, D, S, R \longmapsto A \longmapsto W, H, C, P, N$$

before it can be compared with the located interval row I*. Consequently the packet hands off regular Cauchy data to the located interface without selecting a limit, quotienting streams, invoking countable choice, or asserting ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted packet $L = (I, B, D, S, R, A, W, H, C, P, N)$. The regular completion boundary B is the only completion-facing row admitted by the carrier.

The read then exposes the shared dyadic tolerance ledger D, the finite stream window S, and the regular rational readback R. These rows are the displayed regular Cauchy route inherited from chapter 1639.

The rational approximation row A is read only after D, S, R have been fixed. It gives the finite comparison surface that can be tested against the located interval endpoint data, while the witness ledger W records that the same tolerance request is being used.

Transport, replay, provenance, and local naming then preserve this route. Since the carrier has no selected limit, quotient equality, countable-choice schedule, host real equality, or completeness coordinate, the handoff remains a finite regular Cauchy interface rather than a Real-completeness assertion. □

Theorem 1850.4 (Bishop located Cauchy real shared-tolerance handoff)**.** *Any accepted* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *packet* $L = (I, B, D, S, R, A, W, H, C, P, N)$ *that compares the regular Cauchy readback with the located interval endpoints must expose the same dyadic row* D *before both the interval row* I *and the regular route* B, S, R, A, W *can be consumed. Thus the located side and the Cauchy side cannot be read under unrelated tolerance budgets.*

Proof. Project the accepted carrier. The only dyadic tolerance coordinate available to the packet is the row D of Definition 1850.1.

The regular Cauchy side is governed by Definition 1850.3. That route reads B, D, S, R before the approximation row A and the witness row W are consumed.

The located side is governed by Definition 1850.6. It keeps interval endpoint, radius, locator, locatedness-modulus, and metric-bracket data attached to the same displayed tolerance route through D, S, R, A, W.

Finally apply non-escape through the carrier boundary: no second tolerance row, host interval equality, quotient representative, selected limit, or ambient Real completeness coordinate exists. Hence every comparison between the Cauchy readback and the interval endpoints uses the shared row D. □

Theorem 1850.5 (Bishop located Cauchy real shared-tolerance exactness)**.** *For every accepted* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *packet* $L = (I, B, D, S, R, A, W, H, C, P, N)$*, any comparison between the regular Cauchy readback and the located interval endpoints is exhausted by the single displayed dyadic row* D*. The interval side may not substitute a second radius budget, and the Cauchy side may not substitute a second tail budget, host interval equality, quotient representative, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier. The only dyadic tolerance coordinate listed by Definition 1850.1 is D, and both the regular Cauchy boundary row and the located interval row are attached to it through the witness ledger W.

Apply the regular-Cauchy handoff of Definition 1850.3. It forces the completion-facing side to read B, D, S, R before the approximation row A and the witness row W can be consumed.

Apply located-interval stability from Definition 1850.6. Endpoint, radius, locator, locatedness-modulus, and metric-bracket data remain tied to the same displayed route through $I, B, D, S, R, A, W, H, C, P, N$ under componentwise transport.

Finally use nonescape from Definition 1850.7. Since the accepted packet has no coordinate for a second tolerance ledger, host interval equality, quotient representative, selected limit, countable choice, trichotomy, compactness, or ambient Real completeness, the row D is the whole comparison budget. □

Theorem 1850.6 (Bishop located Cauchy real located-interval stability)**.** *Let* $L = (I, B, D, S, R, A, W, H, C, P, N)$ *and* $L' = (I', B', D', S', R', A', W', H', C', P', N')$ *be accepted* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *carriers. If* H *records componentwise* hsame *transport from each row of* L *to the corresponding row of* L'*, then the local classifier transports locatedness only through the displayed rows*

$$I, B, D, S, R, A, W, H, C, P, N.$$

In particular, interval endpoint, radius, locator, locatedness-modulus, and metric-bracket data remain attached to the same regular Cauchy tolerance route and cannot be replaced by host interval equality, trichotomy, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The classifier is row-local by Definition 1850.1. Its accepted inputs are precisely the displayed tuple $I, B, D, S, R, A, W, H, C, P, N$.

Componentwise hsame transport in H moves the located interval row together with the regular Cauchy boundary, tolerance ledger, stream window, readback row, approximation row, witness ledger, replay row, provenance row, and local name. It has no coordinate that could identify host intervals or choose quotient representatives.

Locatedness therefore remains attached to I and to the shared tolerance route through D, S, R, A, W. The transported packet may replay the same finite route through C, P, and N, but it cannot introduce trichotomy, compactness, host equality, or ambient Real completeness. □

Theorem 1850.7 (Bishop located Cauchy real nonescape)**.** *Every located or completion-facing read of an accepted* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *carrier* $L = (I, B, D, S, R, A, W, H, C, P, N)$ *is exhausted by the displayed rows* $I, B, D, S, R, A, W, H, C, P, N$*. Thus the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyReal}^{\uparrow}}$ *boundary exports only located interval data, regular Cauchy completion-boundary data, shared dyadic tolerances, finite stream windows, regular rational readback, rational approximation windows, located-Cauchy witness data, componentwise transport, continuation replay, provenance, and local naming data.*

It exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy datum, compactness principle, or bridge credential.*

Proof. Project the accepted carrier. The tuple $L = (I, B, D, S, R, A, W, H, C, P, N)$ lists every coordinate admitted by the local $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyReal}^{\uparrow}}$ surface.

The analytic rows are I, B, D, S, R, A, W. They give the located interval packet, the regular completion boundary, the shared tolerance ledger, finite windows, regular readback, rational approximation data, and the witness ledger tying the two routes together.

The structural rows do not enlarge the boundary. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row only preserve, replay, source, and name the same displayed packet.

Since no excluded object occurs as a coordinate or replay target, no located or completion-facing read can escape to $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host real equality, trichotomy, compactness, or a bridge credential. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyReal}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyReal}^{\uparrow}}$, seed)

Scope closed: Definition 1850.1, Definition 1850.2, Definition 1850.3, Definition 1850.4, Definition 1850.5, Definition 1850.6, and Definition 1850.7 seed the finite located Cauchy real interface over $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, and $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host real equality, trichotomy, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval admission, regular-Cauchy admission, approximation compatibility, witness stability, regular-Cauchy handoff, located-interval transport, and non-escape proofs for BEDC.Derived.BishopLocatedCauchyRealUp.

Constructive story: A $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ packet is generated from a LocatedRealInterval row, a BishopRegularCauchyCompletion boundary row, a shared DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular readback, a RealRationalApproximation row, a located-Cauchy witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1851

A Concrete Naming Certificate for $\mathsf{LocatedArchimedean}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedArchimedean}^{\uparrow}$ packet records the finite located Archimedean search boundary used before a constructive real-completion consumer may treat a locatedness request, an integer-part bracket, an Archimedean approximation row, and a dyadic-density witness as one displayed route. It sits over the locatedness budget of chapter 1627, the integer-part boundary of chapter 1784, the approximation surface of chapter 1507, the dyadic-density witness of chapter 1484, the finite window interface of chapter 126, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite search boundary for completion-facing RealUp work; it is not trichotomy, quotient real equality, countable choice, selected integer search, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 1851.1 (LocatedArchimedean carrier)**.** *A* $\mathsf{LocatedArchimedean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, I, A, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatednessModulus}^{\uparrow}$ *row receiving the positive tolerance request and exposing the finite locatedness budget.*
2. *I is the* $\mathsf{RealIntegerPart}^{\uparrow}$ *row that supplies the integer-part and bracket route attached to the same real-name source.*
3. *A is the* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *row that records the finite bound, rational source, dyadic tolerance, and approximation window used by the search.*
4. *D is the* $\mathsf{DyadicDensityReal}^{\uparrow}$ *witness row available at the displayed radius after the locatedness and approximation rows have been opened.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the tolerance request, integer-part bracket, approximation row, and dyadic-density witness.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S before any Real-facing consumer receives the search boundary.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, I, A, D, S, R have been displayed in the same history.*

8. *H records componentwise* hsame *transport for the locatedness request, integer-part bracket route, Archimedean approximation row, dyadic-density witness, stream windows, readbacks, real seal, continuation rows, provenance, and local naming.*

9. *C records displayed* Cont *replay from every precision, interval, or completion-facing request through L, I, A, D, S, R, E without inserting a second Archimedean search.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedArchimedean}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$, $\mathsf{RealIntegerPart}^{\uparrow}$, $\mathsf{ArchimedeanApproximation}^{\uparrow}$, $\mathsf{DyadicDensityReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedArchimedean}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for trichotomy, quotient real equality, selected integer search, countable choice, hidden completion data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedArchimedeanUp.TasteGate.LocatedArchimedeanTasteGate single carrier alignment`

Theorem 1851.2 (LocatedArchimedean NameCert obligations)**.** *For every accepted* $\mathsf{LocatedArchimedean}^{\uparrow}$ *carrier*

$$K = (L, I, A, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedArchimedean}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the locatedness request L, integer-part bracket route I, Archimedean approximation row A, dyadic-density witness D, stream windows S, regular readback R, real seal E, and structural rows H, C, P, N;*

2. *located search compatibility, requiring L to provide the tolerance budget before any integer-part or approximation row is consumed;*

3. *bracket and approximation exactness, requiring I and A to name the same finite search boundary rather than independent host searches;*

4. *dyadic witness handoff, requiring D to be read only after the locatedness, bracket, approximation, window, and readback rows have been displayed;*

5. *transport and replay stability, requiring H and C to preserve the accepted status of L, I, A, D, S, R, E under componentwise movement and downstream requests;*

6. *non-escape, excluding trichotomy, quotient real equality, selected integer search, countable choice, hidden completion data, ambient Real completeness, Lean verification, and bridge checking from the local certificate surface.*

No LocatedArchimedean obligation is stored outside these rows.

Proof. Project the accepted packet $K = (L, I, A, D, S, R, E, H, C, P, N)$. The projection exposes exactly the locatedness request, integer-part bracket route, approximation row, dyadic witness, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

The located search compatibility clause is the first gate. The tolerance and locatedness budget is already fixed by L, so the later integer-part and approximation rows cannot choose a separate search request.

The bracket and approximation clauses use the sibling surfaces of chapter 1784 and chapter 1507. The rows I and A are displayed in the same BHist packet and are transported together by H, so their search data are read as one finite boundary.

The dyadic witness clause then applies chapter 1484. The witness row D is downstream of the locatedness, bracket, and approximation rows, and the windows S with readback R expose the same finite observation surface before the seal E is reached.

Finally, C replays all downstream requests through the listed rows, while P and N provide only provenance and local naming. Since the carrier has no field for the excluded principles, the displayed rows exhaust the local $\mathsf{NameCert}_{\mathsf{LocatedArchimedean}^{\uparrow}}$ surface. □

Theorem 1851.3 (LocatedArchimedean dyadic-density handoff)**.** *Every dyadic-density witness exported by an accepted* $\mathsf{LocatedArchimedean}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{LocatednessModulus}^{\uparrow} \longmapsto \mathsf{RealIntegerPart}^{\uparrow} \longmapsto \mathsf{ArchimedeanApproximation}^{\uparrow} \longmapsto \mathsf{DyadicDensityReal}^{\uparrow} \longmapsto \mathsf{StreamNam}$$

Concretely, the consumer first receives the locatedness budget L, then the integer-part bracket route I, the Archimedean approximation row A, the dyadic witness D, the finite windows S, the regular readback R, the Real seal E, and finally the structural rows H, C, P, N.

Proof. Begin with a dyadic-density read from an accepted carrier. The only dyadic-density coordinate is D, and it appears after the locatedness, integer-part, and approximation rows in the displayed packet.

The locatedness row L fixes the tolerance budget. The integer-part route I and approximation row A then bind the finite Archimedean search boundary used by that budget, so the witness cannot arise from a detached host search.

The witness is exposed only through the window and readback rows S and R. These are the same StreamName and RegSeqRat surfaces used by chapter 126 and chapter 301, and they precede the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The rows H, C, P, N preserve, replay, source, and name the same route. Hence every dyadic-density witness factors through the stated finite handoff and exports none of the excluded host or quotient data. □

Theorem 1851.4 (LocatedArchimedean real search boundary)**.** *Every completion-facing* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedArchimedean}^{\uparrow}$ *carrier receives exactly the finite search boundary*

$$(L, I, A, D, S, R, E, H, C, P, N)$$

as carrier data. The consumer may read locatedness budget, integer bracket, Archimedean approximation, dyadic-density witness, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming, but it cannot read trichotomy, quotient real equality, selected integer search, countable choice, hidden completion data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the completion-facing read. Its only Real-facing coordinate is the seal row E, and E is reached after L, I, A, D, S, R have been displayed.

The rows L, I, A, D bind the analytic part of the search boundary. Locatedness supplies the request budget, the integer-part row supplies the bracket route, the Archimedean approximation row supplies the finite bound and tolerance route, and the dyadic-density row supplies the displayed witness.

The rows S and R provide the finite observation and readback surface for the same boundary. Thus the consumer receives StreamName windows and RegSeqRat readback before it can inspect the Real seal named by chapter 38.

Transport, replay, provenance, and local naming are exactly H, C, P, N. No carrier field remains from which trichotomy, quotient real equality, selected integer search, countable choice, hidden completion data, or Real completeness could be read. □

Falsifiable prediction: Every LocatedArchimedean$^{\uparrow}$ completion-facing read factors through one LocatednessModulus$^{\uparrow}$ row, one RealIntegerPart$^{\uparrow}$ bracket route, one ArchimedeanApproximation$^{\uparrow}$ row, one DyadicDensityReal$^{\uparrow}$ witness, finite StreamName$^{\uparrow}$ windows, one RegSeqRat$^{\uparrow}$ readback row, one Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locatedness-modulus, real-integer-part, archimedean-approximation, dyadic-density-real, streamname, regseqrat, real: no listed sibling is bijective to LocatedArchimedean$^{\uparrow}$ because this packet jointly binds locatedness budget, integer bracket, Archimedean approximation, dyadic-density witness, finite observation, regular readback, Real seal, transport, replay, provenance, and local naming for the finite search boundary.

Closure status for NameCert$_{\text{LocatedArchimedean}\uparrow}$.

Theory closure: TheoryGate(NameCert$_{\text{LocatedArchimedean}\uparrow}$, seed)

Scope closed: Definition 1851.1, Definition 1851.2, Definition 1851.3, and Definition 1851.4 seed the finite located Archimedean search boundary for Real and completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{LocatedArchimedean}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.LocatedArchimedeanUp.TasteGate.LocatedArchimedeanTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected integer search, countable choice, hidden completion data, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, located search compatibility, bracket and approximation exactness, dyadic witness handoff, transport and replay stability, and Real-seal non-escape for BEDC.Derived.LocatedArchimedeanUp.

Constructive story: A LocatedArchimedean$^{\uparrow}$ packet is generated from a LocatednessModulus$^{\uparrow}$ request row, a RealIntegerPart$^{\uparrow}$ bracket route, an ArchimedeanApproximation$^{\uparrow}$ row, a DyadicDensityReal$^{\uparrow}$ witness, StreamName$^{\uparrow}$ windows, a RegSeqRat$^{\uparrow}$ readback row, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1852

A Concrete Naming Certificate for $\mathsf{ApartnessContinuity}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{ApartnessContinuity}^{\uparrow}$ packet isolates the finite L10 route by which a displayed continuous map carries positive apartness information through an explicit modulus before any Real or order-facing consumer reads the target separation. It sits between the graph rows of chapter 130, the positive-separation rows of chapter 1380 and chapter 304, the tolerance discipline of chapter 1631, the finite source windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the constructive continuity handoff; it does not assert trichotomy, quotient real equality, selected limits, host equality, countable choice, or ambient Real completeness.

Definition 1852.1 (Apartness continuity carrier)**.** *An* $\mathsf{ApartnessContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, L, R, M, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, exposing the source and target observation cells read by the continuity route.*
2. *L is the* $\mathsf{LocatedApartness}^{\uparrow}$ *positive-separation row on displayed source windows.*
3. *R is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-separation row on displayed real names, read only through the same source and target cells as G.*
4. *M is the* $\mathsf{ModulusContinuity}^{\uparrow}$ *tolerance row converting each requested target separation budget into a finite source-window budget.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule consumed by the source apartness row, the graph row, and the modulus row.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the windows in S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording positive gaps, target tolerances, source budgets, endpoint readbacks, and comparison cells.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal visible only after G, L, R, M, S, Q, D have been displayed.*
9. *H records componentwise* hsame *transport for the graph, apartness rows, modulus row, windows, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an apartness-continuity request through graph inspection, source separation, modulus conversion, target-separation handoff, finite readback, dyadic ledger inspection, and Real-seal consumption.*

11. *P is the* Pkg *provenance over* $\mathsf{ApartnessContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ApartnessContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, L, R, M, S, Q, D, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected limits, host equality, countable choice, ambient Real completeness, or a detached semantic-continuity theorem.

Lean4 ✓ `BEDC.Derived.ApartnessContinuityUp.TasteGate.ApartnessContinuityTasteGate single carrier alignment`

Theorem 1852.2 (Apartness continuity NameCert obligations)**.** *For every accepted* $\mathsf{ApartnessContinuity}^{\uparrow}$ *carrier*

$$A = (G, L, R, M, S, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApartnessContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the continuous-map graph, located apartness row, real apartness row, modulus-continuity row, StreamName windows, RegSeqRat readback, DyadicRatCore ledger, Real seal, transport, replay, provenance, and local naming;*

2. *apartness-source stability, requiring L and R to expose positive separation only over the same displayed source and target cells opened by G;*

3. *modulus-to-target-separation handoff, requiring M and D to turn every target-separation request into the finite source budget consumed by S, Q before R or E can be read;*

4. *transport and replay stability, requiring H and C to carry only the displayed rows $G, L, R, M, S, Q, D, E, P, N$ through componentwise* hsame *and displayed* Cont *routes;*

5. *ledger non-escape, requiring D, P, N to export no trichotomy, quotient real equality, selected limit, host equality, countable choice, ambient Real completeness, or detached host continuity fact.*

No local apartness-continuity obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1852.1. The accepted tuple displays exactly the graph row, the two apartness rows, the modulus row, the finite observation rows, the Real seal, and the structural transport, replay, provenance, and naming rows.

Apartness-source stability is the entry condition. The rows L and R may be inspected only with the graph row G, so positive separation cannot be detached from the displayed source and target cells.

The handoff from modulus to target separation is carried by M, S, Q, D. A target-separation request first reads the modulus row M, then opens the StreamName windows S, the regular readback Q, and the dyadic ledger D before the real apartness row R or the terminal seal E is visible.

Transport and replay stability follow from H and C. These rows move the displayed packet componentwise and replay the same finite route; they introduce no independent equality, limit, choice, completeness, or semantic-continuity coordinate.

Finally, ledger non-escape is the exhaustion clause for D, P, N. The dyadic ledger, provenance, and local NameCert rows name only the displayed route, so the excluded nonconstructive principles cannot become local obligations. □

Theorem 1852.3 (Apartness continuity Real handoff)**.** *Every Real or order consumer of an accepted* $\mathsf{ApartnessContinuity}^{\uparrow}$ *packet factors through the displayed apartness and modulus rows*

$$G, L, M, S, Q, D \longmapsto R \longmapsto E, H, C, P, N.$$

The consumer receives the continuous graph, positive-separation evidence, tolerance-to-source budget, finite windows, regular readback, dyadic ledger, terminal Real seal, transport, replay, provenance, and local NameCert rows. It receives no trichotomy, quotient real equality, selected limit, host equality, countable choice, ambient Real completeness, or detached host continuity theorem.

Proof. Start with an accepted packet $A = (G, L, R, M, S, Q, D, E, H, C, P, N)$. Projection exposes the continuous graph row G and the source apartness row L before any target-facing read is available.

The tolerance stage is forced next. The modulus row M converts the target request into finite source windows S, regular readback Q, and the dyadic ledger D, so the target apartness row R is read only after the displayed budget has been consumed.

The Real and order-facing stage is terminal. The seal E and the structural rows H, C, P, N transport, replay, source, and name the same route, with no coordinate for the excluded principles.

Therefore every Real or order consumer factors through the displayed apartness and modulus rows before receiving the terminal surface. Since trichotomy, quotient real equality, selected limits, host equality, countable choice, ambient Real completeness, and detached semantic continuity are absent from the carrier, none can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApartnessContinuity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApartnessContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1852.1, Definition 1852.2, and Definition 1852.3 seed the finite L10 continuity route from graph, apartness, modulus, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApartnessContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-source stability, modulus-to-target-separation handoff, transport and replay stability, ledger non-escape, and Real handoff proofs for BEDC.Derived.ApartnessContinuityUp.

Constructive story: An $\mathsf{ApartnessContinuity}^{\uparrow}$ packet is generated from a $\mathsf{ContinuousMap}^{\uparrow}$ graph row, $\mathsf{LocatedApartness}^{\uparrow}$ and $\mathsf{ApartnessReal}^{\uparrow}$ positive-separation rows, a $\mathsf{ModulusContinuity}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{CoveringDimension}^{\uparrow}$ packet is read through the compact-metric cover route of chapter 129 and the metric observation surface of chapter 57; the child files below record the finite cover and Real-separability handoffs.

Chapter 1853

A Concrete Naming Certificate for $\mathsf{CoveringDimension}^{\uparrow}$

Following Hurewicz–Wallman and Engelking, the $\mathsf{CoveringDimension}^{\uparrow}$ packet records finite-cover dimension as BEDC refinement data. The classical statement is read here through compact metric finite rows: a finite cover is displayed as BHist cover entries, a refinement is a finite map of rows, the order bound is a visible multiplicity ledger, and the Lebesgue-number evidence is the compact finite ledger already exposed by chapter 129, chapter 1767, and chapter 1302. Thus the packet speaks about a named finite cover-refinement surface, not about arbitrary open-cover compactness.

The dependency route is horizontal rather than purely kernel-local. The finite cover-refinement rows consumed by $\mathsf{CoveringDimension}^{\uparrow}$ are read from the compact metric packet of chapter 129, while the pointwise inspection of cover members uses metric ball observations from chapter 57. Thus the dimension carrier enters the dossier lattice through displayed compact metric and metric-ball rows before it records its refinement and order-bound ledgers.

Definition 1853.1 (Covering dimension carrier)**.** *A* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, E, C, R, O, L, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the compact metric classifier and the finite-net read route consumed by the packet;*
2. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, exposing the finite* ProbeBundle *of centers and tolerance-indexed coverage witnesses used to sample the cover;*
3. *C is the finite cover row, a displayed list of* BHist *cover members together with membership witnesses readable through the compact metric source;*
4. *R is the finite refinement map row, assigning each refined cover member to a displayed parent member of C and recording the local containment witness;*
5. *O is the order-bound witness row, recording that no inspected point-history is carried by more than the displayed bound of refined members;*
6. *L is the* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *row, tying the cover and refinement to positive radius ledgers and finite Lebesgue-style readback;*

7. *H records componentwise* hsame *transport for the compact metric source, finite epsilon-net row, cover row, refinement map, order-bound witness, Lebesgue ledger, replay row, provenance row, and local naming row;*
8. *T records displayed* Cont *routes through which compact-metric, finite-net, cover, refinement, order-bound, and Lebesgue-ledger consumers inspect the packet;*
9. *P is the* Pkg *provenance over* $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, CompactCoverLebesgueLe ProbeBundle, BHist, hsame, Cont, *and local naming rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, a choice-selected refinement, a host finite-subcover theorem, a classical covering-dimension theorem, quotient real equality, or an external multiplicity oracle.

Lean4 ✓ `BEDC.Derived.CoveringDimensionUp.CoveringDimensionTasteGate single carrier alignment`

Theorem 1853.2 (Covering dimension NameCert obligations)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for D, compact metric admission through K, finite epsilon-net admission through E, finite cover admission through C, refinement containment through R, order-bound verification through O, Lebesgue-ledger compatibility through L, componentwise* hsame *stability through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 1853.1. The analytic rows are exactly the compact metric source K, finite epsilon-net row E, finite cover row C, refinement row R, order-bound row O, and Lebesgue-ledger row L.

Carrier admission and compact admission are rowwise. The finite-net row E supplies the only center bundle and coverage witnesses read by the packet, so every cover and refinement check is anchored in displayed compact metric data.

The finite cover row C is refined only through R. Each refined member carries its displayed parent and containment witness before the order row O may count overlaps.

The Lebesgue-ledger row L ties the same cover route to positive radius data. Its compatibility obligation forces the refinement and order rows to be read with the finite lower-radius ledger rather than with a detached open-cover argument.

Finally H, T, P, N transport, replay, package, and name the same finite surface. Since the carrier contains no arbitrary open-cover coordinate, choice-selected refinement, host subcover theorem, classical dimension theorem, quotient equality, or external multiplicity oracle, none of those objects can enter the local NameCert obligations. □

Definition 1853.3 (Covering dimension compact-net root)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the compact-net root is the finite source surface consisting of K, E together with the downstream cover, refinement, order, Lebesgue-ledger, transport, replay, provenance, and naming rows. It is the reusable BHist *root through which every compact finite-cover consumer must enter before reading any dimension-bound data.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier compact net admission`

Theorem 1853.4 (Covering dimension compact-net admission)**.** *Every compact finite-cover consumer of an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N)$$

must first admit the compact-net root through the compact metric row K and the finite epsilon-net row E. The consumer cannot read the finite cover row C, refinement row R, order row O, Lebesgue ledger L, or structural rows H, T, P, N as a dimension-bound packet until the BHist *source of K, E is visible.*

Proof. Project the carrier of Definition 1853.1. The compact source row K and finite epsilon-net row E precede every cover or refinement row in the displayed packet.

Use the NameCert obligations of Definition 1853.2. They require compact metric admission and finite epsilon-net admission before cover admission, refinement containment, order-bound verification, and Lebesgue-ledger compatibility are available.

Thus the compact-net root of Definition 1853.3 is not optional bookkeeping. It is the entry surface for later finite-cover consumers, while H, T, P, N only transport, replay, source, and name the same root. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetAdmission`

Theorem 1853.5 (Covering dimension refinement multiplicity window)**.** *For every accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any refinement-order read must pass from the finite cover row C through the refinement map R before the order-bound witness row O is inspected. The multiplicity window counted by O is therefore a finite BHist *ledger over displayed refined members and their parents in C, not a host overlap oracle or an arbitrary open-cover dimension predicate.*

Proof. Read the cover and refinement rows from Definition 1853.1. The row C lists the cover members, while R assigns each refined member to one displayed parent and records the containment witness.

Apply the NameCert obligations. Order-bound verification through O is downstream of cover admission and refinement containment, so O can count only refined members already visible through C and R.

The structural rows H, T, P, N preserve this finite multiplicity window. Since no carrier coordinate stores a host overlap oracle, choice-selected refinement, or classical dimension predicate, the order read is exactly the displayed C, R, O window. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRefinementMultiplicityWindow`

Theorem 1853.6 (Covering dimension Lebesgue-ledger handoff)**.** *Every compact-cover Lebesgue read exported by an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N)$$

must pass through the finite ledger route

$$L \longmapsto H, T, P, N$$

after the compact-net root K, E and the finite cover-refinement rows C, R, O have been displayed. The handoff exports the same $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *evidence, componentwise transport, replay, provenance, and local naming; it does not export open-cover compactness, selected subcovers, quotient real equality, or host dimension data.*

Proof. Start with the compact-net admission theorem, Definition 1853.4. It places K, E before every downstream cover and dimension-bound read.

Then use Definition 1853.5. The finite cover, refinement map, and order ledger are read through C, R, O before a Lebesgue-style handoff can certify compatibility with the same finite cover route.

Finally apply the NameCert obligations of Definition 1853.2. They keep the Lebesgue ledger L attached to componentwise transport H, displayed replay T, provenance P, and local naming N, excluding open-cover compactness, selected subcovers, quotient real equality, and host dimension data from the handoff. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionLebesgueLedgerHandoff`

Theorem 1853.7 (Covering dimension root finite-cover carrier)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *packet admits a dimension read only through the finite carrier rows*

$$K, E, C, R, O, L, H, T, P, N.$$

The admitted rows are exactly compact metric source data, finite epsilon-net support, finite BHist *cover entries, finite refinement membership, order-bound witnesses, compact-cover Lebesgue ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *naming. The carrier has no coordinate for arbitrary open-cover compactness, choice-selected refinements, host countable dense sets, quotient completion data, or an external dimension oracle.*

Proof. Project the carrier of Definition 1853.1. The compact and metric source rows are read through $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$, with finite sampling supplied by chapter 129, chapter 57, and chapter 1767.

The cover, refinement, order, and radius rows are exactly C, R, O, L. Their finite lower-bound support is the $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ and $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ route of chapter 1302 and chapter 452, not a host compactness theorem.

Finally H, T, P, N preserve the same displayed packet. Since the carrier has no row for the excluded objects, a root dimension read cannot admit them as local obligations. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverCarrier`

Theorem 1853.8 (Covering dimension root refinement obligation)**.** *Every root order comparison exported by an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier factors through the finite refinement ledger before any dimension bound is inspected. In carrier notation, a consumer must read*

$$C \longmapsto R \longmapsto O$$

after the $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *source rows have been displayed and before any* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff reuses the bound.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootRefinementObligation`

Proof. Use Definition 1853.5. It already places the refinement map R between the finite cover row C and the order-bound row O.

Attach the root carrier admission of Definition 1853.7. The compact source rows from $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{FiniteEpsilonNet}^{\uparrow}$ precede C, R, O, while the finite Lebesgue and dyadic support rows are downstream consumers of the same displayed route.

The completion and separability siblings can therefore reuse only the finite order ledger already exposed by C, R, O. They cannot replace the refinement step by a selected subcover or by a host order comparison. □

Theorem 1853.9 (Covering dimension root separability route)**.** *When a root* $\mathsf{CoveringDimension}^{\uparrow}$ *packet uses* $\mathsf{RealSeparability}^{\uparrow}$*, that row supplies only the finite cover-selection route needed by the displayed refinement ledger. It contributes rational windows, dyadic tolerances, and regular-rational readback through*

$$\mathsf{RealSeparability}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}$$

before the order row is exported; it does not supply a host countable dense set, selected dense sequence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite-density route Definition 1853.17. It routes dense Real witnesses through displayed rational and dyadic density rows before O is exposed.

The dyadic and rational readback rows are the siblings chapter 291, chapter 301, and chapter 1809. They refine finite cover selection and tolerance inspection, but they do not add a separate dense-set carrier.

Combine this with Definition 1853.8. The separability route is inserted before the displayed order export and remains attached to C, R, O, L, H, T, P, N, so no selected dense sequence or host completeness principle is available. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier root separability route`

Theorem 1853.10 (Covering dimension root completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or Cauchy-facing read of a root* $\mathsf{CoveringDimension}^{\uparrow}$ *packet is consumed only after the finite cover and refinement ledgers have been displayed. The handoff route first exposes*

$$K, E, C, R, O, L$$

and only then permits completion-facing rows from $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the regular Cauchy support of chapter 301 to inspect the same bound.*

Proof. Apply Definition 1853.8. It fixes C, R, O downstream of the compact source rows and upstream of every order-bound export.

Next apply the Lebesgue-ledger handoff Definition 1853.6. The row L ties the displayed finite cover and refinement to compact-cover lower-bound data before structural transport and replay are read.

Only after those rows are visible may the completion-facing siblings of chapter 1506, chapter 301, and chapter 291 consume the bound. The completion handoff therefore receives a finite displayed cover-refinement packet, not a selected completion object or a hidden Cauchy limit. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier root completion handoff`

Theorem 1853.11 (Covering dimension refinement order)**.** *Every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *refinement factors through the finite route*

$$K, E, C \longmapsto R, O, L \longmapsto H, T, P, N.$$

Thus a downstream dimension consumer receives a displayed finite cover, a displayed refinement map into that cover, an order-bound witness over the refined rows, and the compact Lebesgue ledger before it may read the packet as a dimension bound.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRefinementOrder`

Proof. Begin with a downstream consumer of an accepted carrier D. By Definition 1853.1, the consumer enters through the compact metric row K and the finite epsilon-net row E before it can inspect any cover data.

The cover row C is then opened as a finite list of named cover members. The refinement row R maps each refined member back to this list and records the containment evidence needed to read the refined cover as a refinement of C.

The order-bound row O counts only refined members that are already visible through R. Hence the order claim is a finite overlap ledger over the displayed refinement, not a global theorem about all open covers.

The Lebesgue-ledger row L supplies the positive radius readback compatible with the same finite cover route. Transport, replay, provenance, and naming through H, T, P, N preserve that route for the dimension consumer. □

Theorem 1853.12 (Covering dimension compact-metric dependency)**.** *Every finite-cover refinement read exported as a* $\mathsf{CoveringDimension}^{\uparrow}$ *dimension bound must expose the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *carrier rows, the finite cover row, the finite refinement row, the order-bound row, any* $\mathsf{RealSeparability}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *rows used by the read, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *provenance before the bound is exported. In carrier terms, an accepted packet*

$$D = (K, E, C, R, O, L, H, T, P, N)$$

is consumed through compact metric source K, finite-net row E, finite cover row C, refinement row R, order row O, compact Lebesgue ledger L, transport H, replay T, package provenance P, and local naming N, with the dependency routes of the compact-metric and metric siblings remaining visible.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactMetricDependency`

Proof. Project the carrier of Definition 1853.1. The rows $K, E, C, R, O, L, H, T, P, N$ are the only compact source, finite-net, cover, refinement, order, Lebesgue-ledger, transport, replay, provenance, and naming rows available to the dimension consumer.

Use the compact metric route and metric dependency route, Definition 1853.22 together with the displayed route in `compact_metric_dependency_route.tex`. They force the cover-refinement read through $\mathsf{CompactMetric}^{\uparrow}$ finite-net data and $\mathsf{Met}^{\uparrow}$ ball observations before the refinement map and order ledger are inspected.

If the read uses dense real witnesses or completion-facing metric data, apply Definition 1853.17 and the completion dependency child routes. Those routes place $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ rows before the order-bound export rather than after it.

Finally H, T, P, N keep the route componentwise stable, replayed, sourced, and locally named. Since no carrier row stores arbitrary open-cover compactness, a choice-selected refinement, a host

dimension theorem, quotient real equality, or an external multiplicity oracle, a dimension bound can be exported only after the listed dependency rows are visible. □

Theorem 1853.13 (Covering dimension ledger non-escape)**.** *No accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet can replace its finite rows* $K, E, C, R, O, L, H, T, P, N$ *by arbitrary open-cover compactness, choice-selected refinements, a host multiplicity oracle, or a classical dimension theorem. Any exported dimension-bound claim must be read through the displayed cover, refinement, order-bound, Lebesgue-ledger, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionLedgerNonEscape`

Proof. Inspect the carrier inventory of Definition 1853.1. The only compactness data are the compact metric row K, the finite epsilon-net row E, and the finite Lebesgue-ledger row L.

The only cover data are the finite cover row C and the refinement map row R. The only multiplicity data are the order-bound witnesses in O.

The structural rows H, T, P, N preserve and expose these displayed rows. They do not add an open-cover family, a choice operator, a host overlap counter, or an external theorem.

Therefore any consumer that exports a dimension-bound claim without reading $K, E, C, R, O, L, H, T, P, N$ leaves the carrier surface and is not reading a $\mathsf{CoveringDimension}^{\uparrow}$ certificate. □

Theorem 1853.14 (CoveringDimension nerve order bound)**.** *An accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover-refinement packet may export an order-bound ledger to* $\mathsf{SimplicialComplex}^{\uparrow}$ *only through displayed finite cover members, refinement rows, overlap and multiplicity rows, and a finite face inventory. In carrier coordinates, the route from* $D = (K, E, C, R, O, L, H, T, P, N)$ *to a nerve consumer first reads the* $\mathsf{CompactMetric}^{\uparrow}$ *source* K*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row* E*, finite cover row* C*, refinement map* R*, order-bound ledger* O*, and* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *row* L *before the* $\mathsf{SimplicialComplex}^{\uparrow}$ *face rows can receive a dimension bound.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionNerveOrderBound`

Proof. Project the accepted carrier. The compact metric source K and finite epsilon-net row E are the only rows from which cover members can be sampled.

The cover and refinement rows are then opened as finite data. The row C supplies displayed cover members, and R assigns each refined member to a parent row before any overlap count is meaningful.

The order ledger O counts only overlaps among those displayed refined members. Hence the nerve face inventory sent to $\mathsf{SimplicialComplex}^{\uparrow}$ is finite: its vertices are cover rows, and its faces are exactly the inspected nonempty overlap rows bounded by O.

Finally L, H, T, P, N keep the compact-cover Lebesgue ledger, transport, replay, provenance, and local naming attached to the same finite route. The density and completion neighbours, $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$, may be cited only through their displayed finite rows, so the nerve export remains non-parametric and concrete. □

Theorem 1853.15 (CoveringDimension compact-cover nerve nonescape)**.** *No accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *nerve route can export ambient open-cover compactness, a selected subcover, or a host dimension predicate to* $\mathsf{SimplicialComplex}^{\uparrow}$*. Every compact-cover nerve read must remain inside the displayed* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, finite cover, refinement, overlap, multiplicity, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactCoverNerveNonescape`

Proof. Start with Definition 1853.13. It rules out replacing the carrier rows by arbitrary open-cover compactness, choice-selected refinements, host multiplicity oracles, or classical dimension theorems.

Now read the nerve route of Definition 1853.14. The $\mathsf{SimplicialComplex}^{\uparrow}$ consumer receives only the finite face inventory determined by displayed cover members, refinement rows, overlap rows, and the order-bound ledger.

The compactness source is still the compact metric finite route. $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, and $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ provide sampled cover data and positive-radius ledger rows, while $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ may contribute only their finite density or completion-facing rows when the read asks for them.

Since no carrier coordinate stores an ambient open cover, a selected subcover, or a host dimension predicate, none of those objects can be exported to the nerve. Any such export leaves the accepted $\mathsf{CoveringDimension}^{\uparrow}$ surface. □

Theorem 1853.16 (CoveringDimension scope-closure witness)**.** *Every scoped* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer that exports a dimension-bound claim must first read the displayed compact metric source row K, finite epsilon-net row E, finite cover row C, finite refinement map R, order-bound row O, and compact-cover Lebesgue ledger L. When the read uses dense or completion-facing support, the* $\mathsf{RealSeparability}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *rows must be displayed before O is exported. When the read exports a nerve, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *face rows must be displayed after the compact cover, refinement, overlap, and multiplicity rows. In every case, the local H, T, P, N rows for* hsame*,* Cont*,* Pkg*, and* NameCert *must be read before the bound leaves the packet, so the scoped kernel dependencies are exactly the finite cover, compact metric, Real separability, metric completion, transport, replay, provenance, and naming rows displayed in this chapter.*

Proof. Start with Definition 1853.2. It identifies $K, E, C, R, O, L, H, T, P, N$ as the complete local obligation surface, so a scoped consumer cannot read a dimension-bound claim before those rows are available.

Next apply Definition 1853.12. It keeps the compact metric source, finite-net row, finite cover, refinement, order, optional $\mathsf{RealSeparability}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ rows, and local provenance visible before any finite-cover refinement read is exported.

The completion side is governed by Definition 1853.10. It allows completion-facing rows only after K, E, C, R, O, L have been displayed, so completion support cannot replace the finite cover-refinement route.

Finally apply Definition 1853.15. A nerve consumer receives only the displayed finite face inventory determined by compact cover, refinement, overlap, multiplicity, transport, replay, provenance, and local naming rows. Combining the cited routes gives the stated scoped-closure witness for every dimension-bound export. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionScopeClosureWitness`

Theorem 1853.17 (Covering dimension finite-density route)**.** *Every compact metric cover-refinement consumer that uses dense* $\mathsf{Real}^{\uparrow}$ *witnesses to read a* $\mathsf{CoveringDimension}^{\uparrow}$ *packet must consume* $\mathsf{RealSeparability}^{\uparrow}$ *only through finite rational and dyadic density rows before the order-bound ledger is exposed. For an accepted carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the route first displays the compact metric source K, finite epsilon-net row E, finite cover row C, refinement map R, RealSeparability density handoff, and only then the order-bound row O,

Lebesgue ledger L, transport H, replay T, provenance P, and local naming row N. It exports no arbitrary open-cover compactness, selected dense points, choice-selected refinements, or host dimension theorem.

Proof. Project the accepted $\mathsf{CoveringDimension}^{\uparrow}$ carrier. The rows $K, E, C, R, O, L, H, T, P, N$ are the only compact metric source, finite-net, cover, refinement, order-bound, Lebesgue-ledger, transport, replay, provenance, and local naming rows available to the dimension consumer.

If the cover-refinement read uses dense Real witnesses, the density source is not selected from an ambient dense set. It is read through chapter 1809, where rational enumeration, dyadic tolerance, finite StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal are displayed as finite rows.

Insert that RealSeparability route before O is exposed. The finite cover and refinement rows C, R may inspect dense rational and dyadic witnesses, but the overlap count in O is read only after those witnesses have passed through the separability packet rather than through a choice-selected point.

Finally L, H, T, P, N replay and name the same route. Since the covering-dimension carrier has no arbitrary open-cover coordinate, choice-selected refinement, or host dimension theorem, and the RealSeparability route has no selected dense point or ambient completeness coordinate, the composed finite-density route exports none of them. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteDensityRoute`

Theorem 1853.18 (CoveringDimension finite-order refinement bridge)**.** *Every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-order refinement bridge is read through finite cover rows, compact-net rows,* $\mathsf{RealSeparability}^{\uparrow}$ *budget rows, and the structural route*

$$\mathsf{BHist},\ \mathsf{Cont},\ \mathsf{hsame},\ \mathsf{NameCert}.$$

For a carrier $D = (K, E, C, R, O, L, H, T, P, N)$, the bridge first displays the compact metric row K, finite epsilon-net row E, finite cover row C, refinement map row R, and order-bound row O; if dense real witnesses are used, their $\mathsf{RealSeparability}^{\uparrow}$ *finite rational and dyadic budget rows must be displayed before O is exported as a dimension-bound ledger.*

Proof. Apply the compact-net admission theorem, Definition 1853.4. It places K, E before every finite cover and refinement read.

Use the refinement-order theorem, Definition 1853.11. The finite cover row C, refinement map R, and order ledger O are read as a single displayed multiplicity route.

Finally apply the finite-density route, Definition 1853.17. It inserts the $\mathsf{RealSeparability}^{\uparrow}$ rational and dyadic budget rows before the order-bound export, while H, T, P, N preserve the route through hsame, Cont, provenance, and $\mathsf{NameCert}$. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteOrderRefinementBridge`

Theorem 1853.19 (CoveringDimension compact-net simplicial nerve)**.** *Every* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve exported by an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet consumes compact-net rows and finite cover rows before it forms its finite face inventory. In carrier notation, K, E supply the compact-net source, C, R supply the finite cover and refinement rows, O supplies the order-budget ledger, and any* $\mathsf{RealSeparability}^{\uparrow}$ *density budget is read before the nerve face rows are named through* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

Proof. Start with the nerve order-bound theorem, Definition 1853.14. It sends only displayed cover members, refinement rows, overlap rows, and the order-bound ledger to the simplicial consumer.

Then use compact-cover nerve nonescape, Definition 1853.15. It keeps the nerve route inside CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactCoverLebesgueLedger$^{\uparrow}$, and the optional RealSeparability$^{\uparrow}$ finite rows rather than inside an ambient open-cover theorem.

The structural rows finish the handoff. Componentwise hsame transport, displayed Cont replay, provenance, and local NameCert naming preserve the compact-net and finite-cover rows as the only source of nerve faces. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetSimplicialNerve`

Theorem 1853.20 (CoveringDimension RealSeparability finite-cover budget)**.** *Whenever an accepted* CoveringDimension$^{\uparrow}$ *packet uses* RealSeparability$^{\uparrow}$ *to inspect a finite cover, the separability data is a finite budget row, not a selected dense sequence. The consumer must display rational enumeration rows, dyadic tolerance rows, compact-net rows K, E, finite cover rows C, R, the order-bound row O, and the structural* BHist*,* Cont*,* hsame*, and* NameCert *rows before the finite-cover budget can support a dimension-bound export.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityFiniteCoverBudget`

Proof. Read the finite-density route, Definition 1853.17. It already places RealSeparability$^{\uparrow}$ rational and dyadic rows before the order-bound ledger is exposed.

Attach the compact metric dependency theorem, Definition 1853.12. It keeps the compact metric source, finite-net row, finite cover, refinement, order-bound ledger, and local NameCert provenance visible throughout the read.

Finally use the ledger nonescape theorem, Definition 1853.13. Since neither CoveringDimension$^{\uparrow}$ nor RealSeparability$^{\uparrow}$ stores a selected dense point, arbitrary open-cover compactness, or a host dimension theorem, the finite-cover budget remains the displayed row family $K, E, C, R, O, L, H, T, P, N$ with the separability rows inserted before O. □

Falsifiable prediction: A consumer that claims a CoveringDimension$^{\uparrow}$ bound while omitting the finite cover row, refinement map row, order-bound witness row, compact Lebesgue ledger, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 1853.1.

Independence witness: compactmetric, finite-epsilon-net, compact-cover-lebesgue-ledger: CoveringDimension$^{\uparrow}$ binds a finite cover row, a refinement map, an order-bound ledger, compact finite-net provenance, and Lebesgue-ledger compatibility in one NameCert packet; each named sibling lacks at least one of these rows as a local dimension-obligation coordinate.

Closure status for NameCert$_{\text{CoveringDimension}^{\uparrow}}$.

Constructive story: A CoveringDimension$^{\uparrow}$ packet is generated from a CompactMetricUp source, a FiniteEpsilonNetUp center bundle, a finite BHist cover row, a finite refinement map, an order-bound witness ledger, a CompactCoverLebesgueLedgerUp row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CoveringDimension}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1853.1, Definition 1853.2, Definition 1853.11, Definition 1853.12, Definition 1853.9, Definition 1853.10, Definition 1853.16, and Definition 1853.20 bind finite cover, compact metric, Real separability, metric completion, hsame, Cont, Pkg, and NameCert dependencies before any dimension-bound export, with BEDC.Derived.CoveringdimensionUp.CoveringD as the bridge-facing witness.

Formal verification: FormalStatus(NameCert$_{\text{CoveringDimension}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CoveringdimensionUp.CoveringDimensionScopeClosureWitness`

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, choice-selected refinements, classical covering-dimension theorems, bridge checking, quotient real equality, or host multiplicity oracles.

Upgrade path: Mature closure requires a checked standard-bridge surface for the public finite cover-refinement exports.

The finite cover-refinement rows of $\mathsf{CoveringDimension}^{\uparrow}$ are consumed by chapter 3877 for scale-cover counting and by chapter 4119 for gauge-cover comparison. This forward route records the dimension-theory children as finite cover consumers rather than as detached scale ledgers. The finite cover-refinement rows of $\mathsf{CoveringDimension}^{\uparrow}$ enter compact-metric consumers only after the compact source row of chapter 129 and the finite face-inventory row of chapter 238 have been displayed. A downstream reader therefore consumes cover members, refinement witnesses, and order-bound ledgers through $\mathsf{CompactMetric}^{\uparrow}$ finite-net data and $\mathsf{SimplicialComplex}^{\uparrow}$ finite-face rows before it reaches the local $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ surface; it does not receive an arbitrary open-cover compactness principle, an ambient nerve construction, or a detached host dimension predicate.

Theorem 1853.21 (Covering dimension compact metric forward link)**.** *Every compact-cover or finite-net read exported by* $\mathsf{CoveringDimension}^{\uparrow}$ *first passes through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row and the* $\mathsf{Met}^{\uparrow}$ *ball-observation row before a dimension-bound consumer may inspect the finite cover row, refinement map, order-bound ledger, compact Lebesgue ledger, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactMetricForwardLink`

Proof. Project the carrier of Definition 1853.1. The compact source row K is the only entry point for compact-cover and finite-net data, and its pointwise observations are read through the metric ball surface before a cover member can be tested.

Only after this compact-metric and metric route is visible may the finite cover row C, refinement row R, order-bound row O, and compact Lebesgue ledger L be consumed as a dimension-bound surface. The structural rows H, T, P, N then transport, replay, source, and name the same route without adding an arbitrary open-cover compactness principle. □

The $\mathsf{CoveringDimension}^{\uparrow}$ lattice reads finite cover order only after the compact metric source of chapter 129, the finite epsilon-net schedule of chapter 1767, the compact-cover Lebesgue ledger of chapter 1302, the completion-facing rows of chapter 1506, the rational dense windows of chapter 1809, and the finite face rows of chapter 238 have been displayed. Thus a dimension-bound consumer receives cover multiplicity through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, and $\mathsf{SimplicialComplex}^{\uparrow}$ rows, not through an ambient open-cover compactness predicate.

Theorem 1853.22 (Covering dimension metric dependency)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *cover-refinement read enters through the compact metric source row and the metric ball observation surface before it records the refinement and order ledgers. Concretely, for an accepted carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the row K is read as a $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source, its pointwise cover tests are inspected through* $\mathsf{Met}^{\uparrow}$ *ball observations, and only then may the consumer read the finite cover row C, refinement row R, order-bound row O, compact Lebesgue ledger L, transport row H, replay row T, provenance row P, and local naming row N.*

Proof. Project the carrier of Definition 1853.1. The first row K is the compact metric source, and the carrier description already binds it to finite-net and Lebesgue-ledger rows rather than to an arbitrary open-cover principle.

The cover row C is inspected pointwise. Each such inspection uses metric ball observations from the metric source before a refined member can be assigned to a parent cover member by R.

The compact-source read is specifically the $\mathsf{CompactMetric}^{\uparrow}$ finite-net route of chapter 129: its center bundle is the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ surface of chapter 1767, and only after those centers and radii are visible does the packet export the refinement and order-bound rows used as the dimension bound.

After those metric reads, the order-bound row O counts only displayed refined members, while L records the compact finite-radius ledger for the same cover route. The structural rows H, T, P, N transport and name that route without adding a second dependency surface.

Thus every cover-refinement read reaches the dimension ledger through $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ first. A consumer that skips those rows is not reading the displayed $\mathsf{CoveringDimension}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionMetricDependency`

Theorem 1853.23 (CoveringDimension Cauchy-filter completion route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier must read the finite cover-refinement rows of Definition 1853.1 through the compact metric source of chapter 129, the finite sampling row of chapter 1767, the compact cover Lebesgue ledger of chapter 1302, the rational-window surface of chapter 1809, and then the Cauchy-filter completion boundary of chapter 1569 before any completion-facing readback is exported.*

Proof. Project the carrier $D = (K, E, C, R, O, L, H, T, P, N)$ of Definition 1853.1. The cover row C, refinement row R, order-bound row O, and Lebesgue ledger L are available only after the compact metric source K and finite epsilon-net row E have been displayed.

Use the metric-completion dependency route already carried by the $\mathsf{CoveringDimension}^{\uparrow}$ sibling lattice. It sends completion-facing reads through compact metric finite covers, $\mathsf{RealSeparability}^{\uparrow}$ rational windows, finite Lebesgue ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row.

Finally route the completion boundary through chapter 1569. The Cauchy-filter completion packet exposes the source filter, finite windows, tolerance, regular readback, and Real seal rows before completion consumers inspect the handoff, so the dimension route cannot bypass the displayed finite-cover sampling surface. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCauchyFilterCompletionRoute`

Theorem 1853.24 (CoveringDimension finite refinement window)**.** *Under the completion handoff of Definition 1853.23, the finite refinement and order-bound window of an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet remains the displayed* BHist *route*

$$K, E, C \longmapsto R, O, L \longmapsto H, T, P, N.$$

The handoff exports no arbitrary open-cover theorem, no choice-selected refinement, no hidden completion point, and no detached multiplicity oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteRefinementWindow`

Proof. Start from Definition 1853.11. It factors every dimension consumer through the compact metric row K, finite epsilon-net row E, finite cover row C, refinement row R, order-bound witness O, and compact Lebesgue ledger L before the structural rows are read.

Apply Definition 1853.23. Completion-facing consumers inherit the same finite route through $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{CauchyFilterCompletion}^{\uparrow}$, but those sibling rows add only rational windows, Cauchy-filter windows, regular readback, and completion-facing seal data.

Now use Definition 1853.13. Since the local carrier has no coordinate for arbitrary open covers, selected refinements, hidden completion points, or external multiplicity oracles, transporting the route through completion boundaries preserves the displayed BHist refinement window. □

The finite cover-refinement rows of $\mathsf{CoveringDimension}^{\uparrow}$ can be consumed by $\mathsf{RealSeparability}^{\uparrow}$ only after the compact metric source of chapter 129, the finite epsilon-net schedule of chapter 1767, and the compact-cover Lebesgue ledger of chapter 1302 have been displayed. Thus the separability-facing route of chapter 1809 receives finite cover data through compact metric, finite-net, and Lebesgue-ledger rows rather than through an arbitrary open-cover compactness claim.

Lemma 1853.25 (Covering dimension separability cover-refinement route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer of a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover-refinement row factors through the compact metric row K, finite epsilon-net row E, cover row C, refinement row R, order-bound row O, compact-cover Lebesgue ledger L, and structural rows H, T, P, N of Definition 1853.1. No such consumer receives a detached host open cover, choice-selected refinement, or external multiplicity oracle.*

Proof. Project the carrier of Definition 1853.1. The cover and refinement rows are visible only after the compact metric source K and finite epsilon-net row E have supplied the finite sampling surface.

The Lebesgue ledger L then binds the cover and refinement rows to displayed positive-radius data from chapter 1302. Hence a $\mathsf{RealSeparability}^{\uparrow}$ consumer can read the refinement only through compact metric, finite-net, and Lebesgue-ledger dependencies.

Finally the structural rows H, T, P, N preserve the same route under hsame transport, Cont replay, package provenance, and local naming. Since the carrier has no detached host open cover, choice-selected refinement, or external multiplicity oracle, none can be exported to the separability-facing consumer. □

Lean4 ✓ `BEDC.Derived.CoveringDimensionUp.CoveringDimensionSeparabilityCoverRefinementRoute`

Theorem 1853.26 (Covering dimension real-separability refinement bound)**.** *When* $\mathsf{CoveringDimension}^{\uparrow}$ *uses* $\mathsf{RealSeparability}^{\uparrow}$ *to sample finite cover members, the resulting dimension-bound read must expose the compact metric source K, finite epsilon-net schedule E, finite cover row C, refinement row R, finite* $\mathsf{RealSeparability}^{\uparrow}$ *density witnesses, order-bound row O, and compact-cover Lebesgue ledger L before any order bound is consumed. Thus the separability-facing refinement route uses chapter 1809 only through finite density rows and the compact-cover ledger of chapter 1302, not through arbitrary open-cover or selected dense-point data.*

Lean4 ✓ `BEDC.Derived.CoveringDimensionUp.CoveringDimensionRealSeparabilityRefinementBound`

Proof. Apply Definition 1853.25. It already forces every separability-facing refinement consumer through the compact metric row K, finite epsilon-net row E, cover row C, refinement row R, order-bound row O, and Lebesgue ledger L.

Insert the $\mathsf{RealSeparability}^{\uparrow}$ read only at the finite-density position. The consumer may sample cover members through the finite density witnesses supplied by chapter 1809, but those witnesses are read after K, E, C, R have displayed the finite cover-refinement surface.

The order-bound row O and Lebesgue ledger L are then consumed after the finite density witnesses are visible. Since the route has no coordinate for arbitrary open-cover compactness, selected dense points, or an external refinement oracle, the resulting dimension-bound read is exactly the displayed finite-density handoff. □

Every RealSeparability$^{\uparrow}$ density-window use of CoveringDimension$^{\uparrow}$ must consume the finite cover-refinement rows of Definition 1853.1 through displayed metric and naming data rather than through an ambient open-cover principle.

Theorem 1853.27 (Covering dimension real-separability cover obligation)**.** *Every* CoveringDimension$^{\uparrow}$ *cover-refinement row consumed by a* RealSeparability$^{\uparrow}$ *density-window argument must expose finite cover rows, dyadic or rational density windows, metric carrier rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and* NameCert *rows on the same finite BEDC surface. The consumer receives no detached host cover, choice-selected refinement, external density oracle, quotient point equality, or ambient separability theorem.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityCoverObligation`

Proof. Project the covering-dimension carrier of Definition 1853.1. Its cover and refinement rows are finite rows attached to the compact metric source, finite epsilon-net schedule, order-bound row, Lebesgue ledger, transport, replay, provenance, and local naming.

Use the separability cover-refinement route, Definition 1853.25. It already forces any RealSeparability$^{\uparrow}$ consumer to pass through the compact metric row, finite epsilon-net row, cover row, refinement row, order-bound row, compact-cover Lebesgue ledger, and structural rows before the separability-facing read is available.

The density-window requirement is therefore the same finite obligation surface read from the separability side. Dyadic or rational windows may be consumed only as displayed finite rows over the metric carrier and structural rows, while detached host covers, choice-selected refinements, external density oracles, quotient point equality, and ambient separability are not carrier coordinates. □

Theorem 1853.28 (Covering dimension real-separability cover scope)**.** *Every* RealSeparability$^{\uparrow}$ *density-window consumer of* CoveringDimension$^{\uparrow}$ *must cite* CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactCoverLebesg DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows before reading a cover-refinement row. The consumer receives the finite L10 cover route through the displayed carrier, not through open-cover compactness, quotient equality, or external separability.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityCoverObligationScope`

Proof. Project the CoveringDimension$^{\uparrow}$ carrier of Definition 1853.1. Its finite cover-refinement rows expose compact metric source data, finite epsilon-net witnesses, compact-cover Lebesgue ledger data, order rows, transport, replay, provenance, and local naming before a separability-facing consumer can read a refinement.

Apply the cover-obligation theorem Definition 1853.27. It already requires finite cover rows, dyadic or rational density windows, metric carrier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and NameCert rows on the same BEDC surface.

The scope condition is therefore just the explicit consumer boundary for that route. A RealSeparability$^{\uparrow}$ density-window read must display CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactCoverLebesgueLedge DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert before it reads the cover-refinement row. Since none of those rows is open-cover compactness, quotient equality, or an external separability theorem, those excluded objects remain outside the accepted packet. □

Theorem 1853.29 (Covering dimension real-separability cover nonescape)**.** *Every* RealSeparability$^{\uparrow}$ *density-window consumer of a* CoveringDimension$^{\uparrow}$ *cover-refinement row that drops one of the* CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactCoverLebesgueLedger$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *or* NameCert *rows leaves the accepted cover-refinement surface. Such a consumer cannot replace the finite route with open-cover compactness, quotient equality, external separability, or a choice-selected refinement.*

Proof. Project the covering-dimension carrier of Definition 1853.1. Its accepted cover-refinement surface displays the compact metric source, finite epsilon-net schedule, compact-cover Lebesgue ledger, dyadic or rational density windows, regular readback, Real-facing seal, history row, transport, replay, provenance, and local naming before a separability-facing read is available.

Apply the cover-obligation theorem Definition 1853.27 and the scope theorem Definition 1853.28. Together they force the RealSeparability$^{\uparrow}$ density-window consumer to cite the same finite rows before reading a CoveringDimension$^{\uparrow}$ cover-refinement row.

If any listed row is dropped, the consumer no longer has the displayed carrier coordinates required by those two theorems. Open-cover compactness, quotient equality, external separability, and choice-selected refinements are not carrier coordinates, so none of them can repair the missing finite row. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityCoverNonescape`

Theorem 1853.30 (Covering dimension real-separability sampling)**.** *Every finite-cover dimension consumer may use* RealSeparability$^{\uparrow}$ *only through displayed rational dense windows before compact metric cover checks reduce to finite sampled rows. The* CoveringDimension$^{\uparrow}$ *carrier exports only the finite cover row, refinement map, order-bound witness, compact Lebesgue ledger,* hsame, Cont, Pkg, *and* NameCert *rows of Definition 1853.1.*

Proof. Read the dense-window evidence from chapter 1809. It supplies sampled rational names for inspected balls and mesh witnesses before any compact metric cover check is evaluated.

This is the downstream consumer edge for the RealSeparability$^{\uparrow}$ packet: chapter 1809 supplies rational dense windows that chapter 1853 consumes before finite-cover order checks.

Now project the CoveringDimension$^{\uparrow}$ carrier of Definition 1853.1. The only exported rows are the finite cover row, refinement map, order-bound witness, compact Lebesgue ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Thus the dependency is a finite-data sampling interface. It does not add arbitrary open-cover compactness, a selected infinite refinement, or any host dimension theorem. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilitySampling`

Theorem 1853.31 (Covering dimension real-separability route scope)**.** *Every accepted* CoveringDimension$^{\uparrow}$ *packet that uses the* RealSeparability$^{\uparrow}$ *density route must read finite dense-window data through*

$$\mathsf{RealSeparability}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{FiniteEpsilonNet}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow}$$

before it exposes any covering-dimension classifier row. The exposed classifier is therefore bounded by displayed dense samples, metric balls, compact-metric finite-net rows, and regular rational readback; it does not supply open-cover compactness, choice-selected covers, selected limits, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the CoveringDimension$^{\uparrow}$ carrier of Definition 1853.1. A density-dependent read first reaches the RealSeparability$^{\uparrow}$ row, then the metric and compact-metric rows that turn sampled balls into finite cover windows.

Next read the FiniteEpsilonNet$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows. They provide the finite net and regular rational window data consumed by the dimension route before the classifier row is visible.

The carrier has no hidden coordinate for arbitrary open covers or a choice-selected refinement. Since the Real-facing dependency is already exhausted by the displayed finite rows, the route cannot import ambient Real$^{\uparrow}$ completeness or a host compactness theorem. □

Lean4 ✓ `BEDC.Derived.CoveringDimensionUp.CoveringDimensionRealSeparabilityRouteScope`

Theorem 1853.32 (Covering dimension Real separability forward bound)**.** *For every accepted* CoveringDimension$^{\uparrow}$ *carrier, a* Real$^{\uparrow}$ *separability read supplies a finite cover-bound ledger when it is combined with the metric-completion dependency route. Concretely, the rational dense windows of chapter 1809 and the completion-facing rows of chapter 1506 determine a finite ledger of sampled balls, refinement witnesses, and order-bound rows for the* CoveringDimension$^{\uparrow}$ *NameCert surface. The conclusion is only this finite NameCert ledger; it is not a classical open-cover compactness theorem or a selected infinite refinement.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityForwardBound`

Proof. Project the CoveringDimension$^{\uparrow}$ carrier of Definition 1853.1. Its public data are finite cover rows, refinement maps, order-bound witnesses, compact Lebesgue ledger rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Read the Real$^{\uparrow}$ separability route through chapter 1809. The rational dense windows supply the sampled centers and mesh witnesses that can be attached to the finite cover rows.

Next use the metric-completion dependency route through chapter 1506. Completion-facing data may refine the sampled balls, but it must do so through displayed completion rows and the same structural transport and replay rows.

Combining the two routes yields a finite cover-bound ledger: sampled balls from separability, completion-compatible refinement rows, and order-bound witnesses inside the covering-dimension packet. Since every row is displayed and finite, the theorem exports no open-cover compactness principle and no selected infinite refinement. □

Theorem 1853.33 (Real separability covering-dimension forward route)**.** *A* RealSeparability$^{\uparrow}$ *rational-window read can enter* CoveringDimension$^{\uparrow}$ *only through the finite cover row, refinement map, order-bound witness, compact Lebesgue ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows of Definition 1853.1. It does not supply selected dense points, arbitrary open covers, or choice-selected refinements.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.RealSeparabilityCoveringDimensionForwardRoute`
↪ variant `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityForwardRoute`

Proof. Use Definition 1853.32. The existing forward bound identifies the rational dense windows as sampled finite cover data after they are combined with the metric-completion dependency route.

Project the carrier of Definition 1853.1. Its only covering-dimension entry rows for that separability read are the finite cover row, refinement map, order-bound witness, compact Lebesgue ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

The projected route has no coordinate for selected dense points, arbitrary open covers, or choice-selected refinements. Hence every RealSeparability$^{\uparrow}$ read enters CoveringDimension$^{\uparrow}$ through the displayed finite-cover surface alone. □

The $\mathsf{RealSeparability}^{\uparrow}$ route of chapter 1809 supplies finite rational dense windows that the $\mathsf{CoveringDimension}^{\uparrow}$ carrier of chapter 1853 may consume when sampling finite covers. Thus the dimension packet reads separability only as displayed rational-window data before order-bound and finite-cover checks are evaluated. The $\mathsf{CoveringDimension}^{\uparrow}$ carrier consumes $\mathsf{RealSeparability}^{\uparrow}$ through chapter 1809 when countable dense windows are refined into finite cover samples, and it consumes $\mathsf{MetricCompletion}^{\uparrow}$ through chapter 1506 when those finite refinements are transported across completion-stable metric rows. Thus the dimension packet records a real separability and metric completion dependency route without turning dense covers into selected points or completion stability into ambient compactness.

Theorem 1853.34 (Covering dimension real-separability forward link)**.** *Every finite refinement row used by the* $\mathsf{CoveringDimension}^{\uparrow}$ *packet may be read as a consumer of the dense* $\mathsf{RealSeparability}^{\uparrow}$ *route of chapter 1809 when the covering-dimension argument is applied to* $\mathsf{Real}^{\uparrow}$*-derived metric carriers. The dependency is the finite dense-window refinement surface: a displayed dense real window supplies sample points, a finite cover row supplies the cover cells inspected at those samples, and the refinement row records only the finite multiplicity bound visible at that window.*

The link does not assert ambient separability, countable choice of dense points, a host open-cover compactness principle, selected real representatives, or a detached dimension oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityForwardLink`

Proof. Project the $\mathsf{CoveringDimension}^{\uparrow}$ carrier through Definition 1853.1. The refinement row is visible only together with the compact metric source, finite epsilon-net row, cover row, order-bound row, Lebesgue ledger, transport, replay, provenance, and local naming rows.

Use the RealSeparability route of chapter 1809. For a $\mathsf{Real}^{\uparrow}$-derived metric carrier, the dense-window row supplies the finite real samples that the covering-dimension refinement consumes; the refinement is therefore read from displayed dense windows rather than from an ambient separability theorem.

Finally apply the existing separability cover-refinement route, Definition 1853.25. It identifies the finite compact-metric, epsilon-net, cover, refinement, order-bound, Lebesgue-ledger, and structural rows through which the consumer reads the dependency, and it excludes detached host covers, choice-selected refinements, and external multiplicity oracles. □

The $\mathsf{CoveringDimension}^{\uparrow}$ packet reaches $\mathsf{RealSeparability}^{\uparrow}$ only through the metric inspection route of chapter 57. A finite cover-refinement read first exposes metric-ball rows, then uses the rational dense windows of chapter 1809 to sample those displayed balls before the order-bound ledger is read.

Proposition 1853.35 (CoveringDimension metric separability dependency route)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer that uses* $\mathsf{RealSeparability}^{\uparrow}$ *data factors through the metric-ball surface of chapter 57. The exported route is only the finite chain*

$$\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{RealSeparability}^{\uparrow} \longrightarrow \mathsf{CoveringDimension}^{\uparrow}$$

read through the carrier of Definition 1853.1; it does not select dense points, arbitrary open covers, or external refinements.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionMetricRealSeparabilityRoute`

Proof. Project the carrier of Definition 1853.1. The finite cover row and refinement row can be inspected only through displayed metric balls and the compact metric source already named by the packet.

Read the $\mathsf{RealSeparability}^{\uparrow}$ sibling through rational dense windows. Those windows contribute finite sample rows for the displayed metric balls, not selected points outside the certificate surface.

Conversely, any dense-window use of $\mathsf{RealSeparability}^{\uparrow}$ in this chapter must be tied back to a displayed metric-ball refinement. The covering-dimension consumer may cite the separability row only after the relevant ball, radius, and refinement rows have been exposed through $\mathsf{Met}^{\uparrow}$.

The order-bound ledger is then read from the same finite refinement packet. Since the route has no coordinate for arbitrary open covers or external refinements, the dependency remains the displayed metric-to-separability route. □

Theorem 1853.36 (CoveringDimension metric-real dependency)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-refinement read that uses* $\mathsf{RealSeparability}^{\uparrow}$ *must expose, in order, the* $\mathsf{Met}^{\uparrow}$ *ball rows, the* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window rows, and the local order-bound ledger of the covering-dimension packet. The route premise is the metric separability dependency of Definition 1853.35; no consumer may read the dense-window row before the metric-ball surface or the order-bound ledger before both preceding rows are displayed.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionMetricRealDependency`

Proof. Project the carrier of Definition 1853.1. The finite cover, refinement, and order-bound rows are local to the same packet, while Definition 1853.35 supplies the only route from the covering-dimension read to $\mathsf{RealSeparability}^{\uparrow}$.

Use that route premise to expose the $\mathsf{Met}^{\uparrow}$ ball rows first. The dense-window rows of $\mathsf{RealSeparability}^{\uparrow}$ sample only those displayed metric balls, so they cannot be read before the ball centers, radii, and finite refinement rows have been named.

The order-bound ledger is then a downstream read of the covering-dimension packet. Since neither the carrier nor the route premise contains a coordinate for selected dense points, arbitrary open covers, or external refinements, the dependency order remains metric balls, dense windows, and the local order-bound ledger. □

Theorem 1853.37 (Covering dimension Real and Metric forward route)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer that enters the Real and Metric dependency lattice reads* $\mathsf{RealSeparability}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *only through the finite cover, finite-net, dyadic-density, compact-metric, and Lebesgue-ledger rows already displayed by the accepted carrier. The route exposes compact metric source rows, metric ball inspection, finite epsilon-net centers, dyadic density witnesses, cover-refinement rows, and the order-bound ledger before any dimension bound is consumed.*

It exports no ambient metric completion, selected dense sequence, host open-cover compactness theorem, quotient real equality, or external multiplicity oracle.

Proof. Project the accepted carrier $D = (K, E, C, R, O, L, H, T, P, N)$ from Definition 1853.1. The compact metric source K, finite epsilon-net row E, cover row C, refinement row R, order-bound row O, and Lebesgue ledger L are the only analytic rows visible to the packet.

The Real-facing information is read through the existing $\mathsf{RealSeparability}^{\uparrow}$ routes in this chapter. Those routes supply rational and dyadic density witnesses as finite rows before the cover and refinement rows are inspected; they do not select an ambient dense sequence.

The Metric-facing information is read through the compact metric and metric ball rows used by K and E. Thus pointwise cover inspection is finite and row-local, and the order-bound ledger cannot be replaced by a host compactness or multiplicity theorem.

Finally H, T, P, N transport, replay, package, and name the same finite route. Since no extra completion, selected dense sequence, host open-cover theorem, quotient real equality, or external oracle occurs in the carrier, the Real and Metric forward route is exactly the displayed finite dependency surface. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealMetricForwardRoute`

Theorem 1853.38 (CoveringDimension RealSeparability MetricCompletion cover route)**.** *Every finite-cover dimension read over Real-facing metric data must expose* RealSeparability$^{\uparrow}$ *finite dense windows and* MetricCompletion$^{\uparrow}$ *completion boundary rows before a* CoveringDimension$^{\uparrow}$ *consumer may inspect cover order. The route presents compact metric source rows, finite cover and refinement rows, separability witnesses, completion-stable metric rows, the order-bound ledger, transport, replay, provenance, and local naming, but it exports no arbitrary open-cover compactness, selected dense points, choice-selected refinements, quotient point equality, or ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityMetricCompletionCoverRoute`

Proof. Begin with the carrier of Definition 1853.1. Its cover-order read is available only through the compact metric source, finite epsilon-net schedule, finite cover row, refinement row, order-bound witness, compact-cover Lebesgue ledger, transport, replay, provenance, and local naming.

Apply the RealSeparability routes Definition 1853.25 and Definition 1853.32. They place finite rational and dyadic dense windows before any cover-refinement or order-bound row is consumed.

Use the metric-completion route Definition 1853.23. It sends completion-facing cover reads through the displayed completion boundary rather than through an ambient completion theorem.

Together the cited routes force finite dense windows and completion boundary rows to appear before cover order is inspected. Since none of the rows contains arbitrary open covers, selected dense points, choice-selected refinements, quotient equality, or ambient completion data, the composed route exports only the finite cover dependency. □

Theorem 1853.39 (CoveringDimension finite epsilon-net carrier)**.** *For an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the finite epsilon-net row E is a root cover obligation: every cover sample exported by C, every refinement membership read by R, and every order-bound count recorded by O must be read through the displayed ProbeBundle *centers, dyadic tolerance witnesses, and* BHist *coverage rows of E as described in Definition 1853.1. The local NameCert surface of Definition 1853.2 therefore admits finite cover inspection only through $K, E, C, R, O, L, H, T, P, N$; it admits no arbitrary open-cover compactness, selected dense subfamily, or host finite-subcover theorem.*

Proof. Project the carrier rows of Definition 1853.1. The source row K supplies the compact metric packet, and E supplies the finite ProbeBundle of centers and tolerance-indexed coverage witnesses.

The cover row C can be inspected only by reading those displayed finite witnesses. The refinement row R and order-bound row O then operate over the same BHist cover entries, so their counts and memberships remain finite BEDC rows rather than external compactness data.

The rows L, H, T, P, N replay, transport, package, and name the same route. By Definition 1853.2, the NameCert obligation surface has no coordinate for an arbitrary cover or a selected finite subcover, so the finite epsilon-net carrier obligation is root-local. □

Theorem 1853.40 (CoveringDimension compact-cover forward dependency)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the cover-refinement rows C, R, O *consume the finite-cover lower-bound ledgers of chapter 1302 before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealSeparability}^{\uparrow}$ *consumer may read a dimension bound. Concretely, the Lebesgue-ledger row* L *must display the same compact finite-net centers, positive radius lower bounds, and finite fold route used by the cover row* C *and refinement row* R*; only then can* O *record an order bound and expose it through* H, T, P, N*. Thus the dimension-bound surface is downstream of a substantive* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *finite-cover dependency, not an isolated root cover assertion.*

Proof. Start with the finite epsilon-net carrier of Definition 1853.39. It exposes the compact metric source row K, probe centers, tolerance witnesses, and finite cover row C before any order-bound count is visible.

The compact-cover lower-bound ledger then enters through row L. By chapter 1302, this ledger is a finite route through compact finite-net data, positive rational lower bounds, and finite lower-bound folding; it cannot supply an arbitrary open-cover compactness theorem.

Finally use Definition 1853.41 and Definition 1853.43. The order row O is read after C, R, L, and the MetricCompletion and RealSeparability consumers see that same displayed route before receiving a dimension-bound handoff. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactCoverForwardDependency`

Theorem 1853.41 (CoveringDimension order refinement obligation)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the order row O *is valid only after the refinement row* R *has displayed, for each refined* BHist *cover member, its parent member in the finite cover row* C *and its containment witness. Thus the order-refinement obligation exposed by Definition 1853.1 and Definition 1853.2 is a finite computation over* $K, E, C, R, O, L, H, T, P, N$*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row.*

Proof. Open the accepted carrier. The finite cover row C is the only list of cover members, and the refinement row R is the only map from refined members back into that list.

Before O may count overlaps, each refined member must therefore pass through the parent and containment data carried by R. The count is made over finite BHist rows visible through C and R, not over a choice-selected refinement or an ambient multiplicity oracle.

Finally H, T, P, N preserve the same finite route under transport, replay, provenance, and naming. The NameCert obligations in Definition 1853.2 require exactly these rows, so the order refinement claim is a root obligation of the displayed carrier. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionOrderRefinementObligation`

Theorem 1853.42 (CoveringDimension compact metric handoff obligation)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any compact metric consumer that receives a dimension-bound handoff must first read the compact metric source row K*, the finite epsilon-net row* E*, the finite cover row* C*, the refinement row* R*, the*

order-bound row O, and the Lebesgue-ledger row L, and then pass through H, T, P, N. Consequently the handoff certified by Definition 1853.1 and Definition 1853.2 remains a finite BEDC route over BHist, ProbeBundle, hsame, Cont, Pkg, *and* NameCert, *not a claim of open-cover compactness or choice-selected refinement.*

Proof. Read the compact metric handoff in row order. The source row K anchors the carrier, while E displays the finite probes used to sample the cover row C.

The refinement and order data are then exposed by R and O. The Lebesgue-ledger row L ties the same finite cover route to positive radius readback, so the consumer sees a compact metric handoff only after the finite cover-refinement surface has been displayed.

The rows H, T, P, N preserve this route as componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Since the carrier and NameCert obligation theorem list no arbitrary open-cover coordinate or choice-selected refinement coordinate, the handoff obligation is exactly the finite BEDC route $K, E, C, R, O, L, H, T, P, N$. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactMetricHandoffObligation`

Theorem 1853.43 (CoveringDimension RealSeparability root completion route)**.** *For an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any root cover read that samples finite covers through dense real windows must display the RealSeparability$^{\uparrow}$ *rational and dyadic windows of chapter 1809 and the completion-facing rows of* MetricCompletion$^{\uparrow}$ *in chapter 1506 after the finite epsilon-net row E and cover-refinement rows C, R, and before the order-bound row O is consumed. Thus the root cover route is the finite chain*

$$K, E, C, R, \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, O, L, H, T, P, N,$$

not a selected dense sequence, arbitrary open-cover compactness principle, or ambient completion theorem.

Proof. Begin with the compact metric handoff of Definition 1853.42. The rows K, E, C, R display the compact metric source, finite epsilon-net probes, finite cover entries, and refinement memberships before any order count is public.

When dense real samples are used, the sample row must enter through chapter 1809. Those rational and dyadic windows are finite rows, so they can refine the displayed cover samples but cannot supply a selected dense subfamily outside E, C, R.

The completion-facing read is then routed through chapter 1506. Its completion boundary rows are attached after the separability windows and before O, so the order-bound ledger counts only the finite cover-refinement data already exposed through the root cover packet.

Finally L, H, T, P, N replay the same route as the Lebesgue ledger, componentwise hsame transport, displayed Cont replay, package provenance, and local naming rows. Hence the RealSeparability and MetricCompletion dependency is a substantive root-cover obligation of the finite carrier, not a bare citation or an ambient compactness argument. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityRootRoute`

Theorem 1853.44 (CoveringDimension finite-cover root boundary)**.** *For an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

every downstream dimension-bound read must display the compact source row K, finite epsilon-net row E, finite cover row C, refinement row R, order-bound row O, and Lebesgue-ledger row L before any cover consumer can read the order bound. Thus the root boundary for the order row is the finite chain

$$K, E, C, R, O, L,$$

together with the structural rows H, T, P, N, and not an arbitrary open-cover compactness theorem, selected refinement, or ambient separability argument.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootBoundary`

Proof. Start with the carrier of Definition 1853.1. Its dimension-bound surface is a finite row family $K, E, C, R, O, L, H, T, P, N$, so an order-bound read has no source outside those coordinates.

Use Definition 1853.2 and Definition 1853.39. The NameCert obligations require carrier admission through the displayed rows, while the finite epsilon-net carrier theorem makes E the root cover witness for C, R, O.

Finally apply Definition 1853.43. Dense real windows and MetricCompletion rows enter only after E, C, R and before O is consumed, so the consumer sees the same finite root boundary K, E, C, R, O, L rather than a hidden compactness or selection principle. □

Theorem 1853.45 (CoveringDimension refinement order root unblock)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier $D = (K, E, C, R, O, L, H, T, P, N)$, refinement containment through R is the only admissible source for the order row O. A downstream finite-cover consumer may read O only after R has displayed, for each refined* BHist *cover member, its parent member in C and its containment witness over the finite epsilon-net support E.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRefinementOrderRootUnblock`

Proof. Project the refinement and order rows from Definition 1853.1. The row R is the only coordinate that relates refined cover members to parent cover members in C, and O is the only coordinate that records the resulting order bound.

Apply Definition 1853.2. The local NameCert surface assigns order validity to the displayed refinement route, so no external multiplicity oracle or selected refinement can provide the order row.

Use Definition 1853.39 and Definition 1853.43. The finite epsilon-net row E supplies the root cover support, and any dense real or completion-facing sample is routed through that same support before O is visible. Therefore refinement containment through R is the root source that unblocks the order-bound consumer. □

Theorem 1853.46 (CoveringDimension root cover scope route)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any accepted root cover read must first display the finite epsilon-net row E, the finite cover row C, the refinement row R, the $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *radius ledger in L, the* $\mathsf{RealSeparability}^{\uparrow}$ *dyadic and rational windows, and the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-facing boundary before the order row O is consumed. Thus the public root route is the scoped finite chain*

$$K, E, C, R, L, \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, O, H, T, P, N,$$

not an arbitrary open-cover compactness theorem, selected dense subfamily, quotient completion witness, or ambient dimension oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverScopeRoute`

Proof. Begin with Definition 1853.39. It makes E the root finite witness for all cover samples exported by C, refinement memberships in R, and order-bound counts later recorded by O.

Next apply Definition 1853.40. The Lebesgue-ledger row L must display the compact finite-net centers, positive radius lower bounds, and finite fold route before a dimension-bound consumer may inspect the order surface.

Use Definition 1853.43. Dense real sampling is then routed through the $\mathsf{RealSeparability}^{\uparrow}$ rational and dyadic windows, and the completion-facing boundary is routed through $\mathsf{MetricCompletion}^{\uparrow}$ before O becomes visible.

Finally apply Definition 1853.45. It identifies refinement containment through R as the root source that unblocks O, so the complete accepted read is exactly the displayed finite chain through $K, E, C, R, L, \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, O, H, T, P, N$. □

Theorem 1853.47 (CoveringDimension root cover order-ledger exhaustion)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

every order-bound ledger read in O is exhausted by the finite chain

$$K, E, C, R, L, \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, O, H, T, P, N.$$

No later consumer may replace that chain by arbitrary open-cover compactness, a selected dense subfamily, a quotient completion witness, or an ambient dimension oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverOrderLedgerExhaustion`

Proof. First read E through Definition 1853.39. This fixes the root cover support before any order-bound count can be inspected.

Next apply Definition 1853.40. The compact-cover row is routed through the Lebesgue ledger L, so the order read cannot import an unlisted open-cover compactness principle.

Use Definition 1853.43. The RealSeparability and MetricCompletion windows are downstream of the same finite root data rather than selected dense subfamilies or quotient completion witnesses.

Finally apply Definition 1853.46. It identifies the complete scoped route through $K, E, C, R, L, \mathsf{RealSeparability}^{\uparrow}$, and the carrier has no coordinate for an ambient dimension oracle. Therefore every accepted order-bound ledger read is exhausted by the displayed chain. □

Theorem 1853.48 (CoveringDimension root cover primitive route)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the accepted root cover route is scoped only by the finite epsilon-net row E, finite cover row C, refinement row R, Lebesgue-ledger row L, $\mathsf{RealSeparability}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *finite windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row. In particular, the route may read* ProbeBundle *centers and tolerance witnesses through E, but it may not replace those primitive rows by arbitrary open-cover compactness, selected dense subfamilies, quotient completion witnesses, or an ambient dimension oracle.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverScopePrimitiveRoute`

Proof. Begin with Definition 1853.39. It binds every cover sample to the finite ProbeBundle centers, dyadic tolerance witnesses, and BHist coverage rows displayed by E.

Next apply Definition 1853.40. The finite cover row C, refinement row R, and Lebesgue-ledger row L are read as one compact-cover route before any dimension-bound consumer sees the order row.

Use Definition 1853.43. The RealSeparability$^{\uparrow}$ rational and dyadic windows and the MetricCompletion$^{\uparrow}$ completion-facing boundary attach to that finite prefix rather than replacing it by selected dense data or quotient completion data.

Finally apply Definition 1853.45. It keeps the order read tied to the displayed refinement row, while H, T, P, N preserve the same surface as componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Thus the accepted primitive route is exactly the listed finite row family. □

Theorem 1853.49 (CoveringDimension root-cover scope audit)**.** *For an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any order-bound read exported to a downstream cover consumer must expose, in order, the finite epsilon-net row E, finite cover row C, refinement row R, Lebesgue-ledger row L, the RealSeparability$^{\uparrow}$ *dyadic and rational windows, the* MetricCompletion$^{\uparrow}$ *completion-facing boundary, the order row O, componentwise transport H, displayed replay T, provenance P, and local naming N. No selected dense subfamily, quotient completion witness, or ambient dimension oracle is available to that consumer.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverScopeAudit`

Proof. First read the finite support. The finite epsilon-net carrier of Definition 1853.39 makes E the root witness for cover samples, while Definition 1853.40 binds C, R, L to the same compact-cover route before an order bound can be read.

Next attach the real and completion boundary. Apply Definition 1853.43 to route rational and dyadic sampling through RealSeparability$^{\uparrow}$, and use Definition 1853.48 to keep the MetricCompletion$^{\uparrow}$ boundary on the same finite prefix.

Finally expose the consumer surface. The primitive route already carries O, H, T, P, N as the order, transport, replay, provenance, and local-name rows after E, C, R, L, RealSeparability$^{\uparrow}$, MetricCompletion$^{\uparrow}$. Since the carrier has no coordinate for a selected dense subfamily, quotient completion witness, or ambient dimension oracle, every downstream cover consumer receives exactly the displayed audit route. □

Theorem 1853.50 (CoveringDimension root-cover scoped dependency)**.** *For an accepted* CoveringDimension$^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

every accepted root-cover read exposing the order row O must pass through the finite epsilon-net row E, compact-cover Lebesgue ledger row L, RealSeparability$^{\uparrow}$ *dyadic and rational windows, and* MetricCompletion$^{\uparrow}$ *completion-facing boundary before the row is exported through H, T, P, N. Equivalently, the scoped dependency route is the finite chain*

$$K, E, C, R, L, \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, O, H, T, P, N,$$

with no appeal to arbitrary open-cover compactness, selected dense subfamilies, quotient completion, or ambient dimension oracles.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverScopedDependency`

Proof. Start with Definition 1853.39. It makes the finite epsilon-net row E the root source for cover samples, refinement memberships, and later order counts.

Apply Definition 1853.40. The cover row C, refinement row R, and Lebesgue-ledger row L form the compact-cover dependency before any dimension-bound export is available.

Use Definition 1853.43 and Definition 1853.48. The rational and dyadic windows of $\mathsf{RealSeparability}^{\uparrow}$ and the completion-facing boundary of $\mathsf{MetricCompletion}^{\uparrow}$ attach to the same finite prefix before O is consumed.

Finally H, T, P, N replay and name this route as componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. Since each cited route explicitly excludes open-cover compactness, selected dense subfamilies, quotient completion, and ambient dimension oracles, the scoped dependency exports only the displayed finite chain. □

Theorem 1853.51 (CoveringDimension root-cover ledger factorization)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

every accepted root-cover order read factors through the $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *finite-net radius exhaustion of Definition 1302.5 before* $\mathsf{RealSeparability}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *rows are allowed to feed the order ledger* O*. Thus the order-facing route is the finite factorization*

$$K, E, C, R, L \longmapsto \mathsf{CompactCoverLebesgueLedger}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow} \longmapsto O, H, T, P, N,$$

not an open-cover oracle, selected dense subfamily, quotient completion witness, or ambient dimension oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverLedgerFactorization`

Proof. Start with Definition 1853.39. The finite epsilon-net row E supplies the displayed ProbeBundle centers, dyadic tolerance witnesses, and BHist coverage rows for the cover and refinement data.

Next apply Definition 1853.40. The rows C, R, O cannot be read as an isolated dimension assertion; the compact-cover route must first pass through the Lebesgue ledger row L.

Use Definition 1302.5. The ledger exhausts every radius export through the finite-net, pointwise-radius, positive-rational, and finite-fold rows, so the order read has no source in an arbitrary open-cover oracle.

Finally apply Definition 1853.43. The $\mathsf{RealSeparability}^{\uparrow}$ windows and $\mathsf{MetricCompletion}^{\uparrow}$ boundary attach only after the finite ledger has been displayed, and H, T, P, N transport, replay, source, and name that same route. Hence the accepted order ledger exposes no selected dense subfamily, quotient completion witness, or ambient dimension oracle. □

Theorem 1853.52 (CoveringDimension root-cover ledger realization)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

any accepted root-cover order read that exposes O *realizes the* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *finite-net radius exhaustion before the order row is inspected. The realized route displays* K, E, C, R, L *and the radius-exhaustion ledger of Definition 1302.5 before* O, H, T, P, N *can be exported, and it has no source in an open-cover oracle, selected dense subfamily, quotient completion witness, or ambient dimension oracle.*

Proof. Start with Definition 1853.51. It already places the order-facing read behind the finite epsilon-net row E, cover row C, refinement row R, and Lebesgue ledger L.

Apply Definition 1302.5. The compact-cover ledger realizes each radius read through finite-net, pointwise-radius, positive-rational, and finite-fold rows before any order count is available.

The rows H, T, P, N then transport, replay, source, and name that realized finite route. Since no carrier coordinate stores an open-cover oracle, selected dense subfamily, quotient completion witness, or ambient dimension oracle, the accepted read of O is exactly the displayed ledger realization. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverLedgerRealization`

Theorem 1853.53 (CoveringDimension separability-completion lock)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, the* $\mathsf{RealSeparability}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *rows can feed the order ledger* O *only after the finite epsilon-net row* E*, compact-cover ledger* L*, and refinement row* R *are visible on the same root-cover route. Thus separability and completion data are locked behind the finite route*

$$E, C, R, L \longmapsto \mathsf{RealSeparability}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow} \longmapsto O, H, T, P, N,$$

not behind selected dense sequences, quotient completion equality, arbitrary open-cover compactness, or an ambient dimension oracle.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverSeparabilityCompletionLock`

Proof. Read Definition 1853.39 first. It makes E the finite source for the cover samples and tolerance windows used by the root-cover route.

Next use Definition 1853.40 together with Definition 1853.52. The cover row C, refinement row R, and compact-cover ledger L must be visible before the order row is read.

Finally apply Definition 1853.43. The $\mathsf{RealSeparability}^{\uparrow}$ windows and $\mathsf{MetricCompletion}^{\uparrow}$ boundary attach only to that already displayed finite route, and H, T, P, N preserve it as the named order ledger. Therefore the separability and completion rows cannot bypass the finite epsilon-net, compact-cover ledger, or refinement rows. □

Theorem 1853.54 (CoveringDimension public root-cover export)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the public order-bound root-cover surface exports only through K, E, C, R, L*, the* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *finite-net radius route,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and the rows* O, H, T, P, N*. The export has no source in an open-cover oracle, selected dense subfamily, quotient completion witness, or ambient dimension oracle.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionPublicRootCoverExport`

Proof. Start with Definition 1853.39. The root-cover read begins with the finite sample and tolerance data in K and E, not with an arbitrary open-cover oracle.

Apply Definition 1853.40 and Definition 1853.46. These force the cover row C and refinement row R to pass through the compact-cover ledger L before the order row can be inspected.

Next use Definition 1853.51 and Definition 1853.52. The factorization and realization steps route every order-bound read through the $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ finite-net radius exhaustion, so no selected dense subfamily or ambient dimension oracle is available.

Finally apply Definition 1853.53. The $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ rows attach only after the finite ledger route is visible, and O, H, T, P, N export exactly that named route. Therefore the public surface is the displayed finite ledger chain and not a quotient completion witness or external dimension assertion. □

Definition 1853.55 (CoveringDimension root cover carrier)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the root cover-dimension carrier is the finite subpacket

$$D_{\mathrm{root}} = (K, E, C, R, O, S, V, H, T, P, N)$$

where K exposes the $\mathsf{CompactMetric}^{\uparrow}$ *source and its* $\mathsf{Met}^{\uparrow}$ *distance observations, E exposes the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *refinement witnesses, C exposes finite open-cover rows, R exposes containment rows for the refinement of C, O exposes the order-count ledger, S exposes the* $\mathsf{RealSeparability}^{\uparrow}$ *density rows used when covers are sampled through rational or dyadic windows, and V exposes the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve rows attached to the finite refined cover. The structural rows H, T, P, N transport, replay, source, and name exactly these finite rows.*

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverCarrierPacket`

Theorem 1853.56 (CoveringDimension root cover NameCert obligations)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier $D = (K, E, C, R, O, L, H, T, P, N)$, the local root cover-dimension obligations consist exactly of finite open-cover admission through C, compact metric finite-net admission through K, E, metric distance observation through the* $\mathsf{Met}^{\uparrow}$ *part of K, refinement containment through R, density-window compatibility through* $\mathsf{RealSeparability}^{\uparrow}$*, nerve-row compatibility through* $\mathsf{SimplicialComplex}^{\uparrow}$*, order counting through O, and structural transport through H, T, P, N. No root cover-dimension read may bypass these rows by using arbitrary open-cover compactness, a selected dense subfamily, a host refinement theorem, or an external simplicial nerve construction.*

Proof. Project the root carrier of Definition 1853.55. The finite open-cover rows are the displayed entries in C, and their only compact source is the compact metric row K together with the finite epsilon-net row E.

The metric observations are therefore local to K, E, C, R. Distance comparisons can inspect the $\mathsf{Met}^{\uparrow}$ part of K, but they cannot introduce a second metric space, a host open-cover theorem, or a detached refinement.

Density and nerve reads remain downstream. The $\mathsf{RealSeparability}^{\uparrow}$ windows sample the same finite cover route before O is consumed, and the $\mathsf{SimplicialComplex}^{\uparrow}$ nerve rows are attached only to the displayed refined cover.

Finally H, T, P, N preserve and name the same finite data. Since the carrier has no coordinate for arbitrary open-cover compactness, selected dense subfamilies, host refinements, or external nerve construction, projection and route exhaustion give the stated root obligations. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverNameCertObligations`

Theorem 1853.57 (CoveringDimension compact-net refinement handoff)**.** *Every compact-net refinement read exported by an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet factors through*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow} \longrightarrow \mathsf{FiniteEpsilonNet}^{\uparrow} \longrightarrow C, R, O \longrightarrow \mathsf{RealSeparability}^{\uparrow},\ \mathsf{SimplicialComplex}^{\uparrow} \longrightarrow H, T, P, N.$$

The handoff gives downstream cover-dimension consumers finite open-cover rows, metric distance observations, compact finite-net refinements, density windows, nerve rows, order counts, transport, replay, provenance, and local naming, and it gives no ambient compactness theorem, selected dense sequence, external refinement oracle, or host nerve construction.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetRefinementHandoff`

Proof. Begin with the compact-net entry. The rows K and E expose the compact metric source, metric observations, finite ProbeBundle centers, and tolerance-indexed coverage witnesses before any finite cover row is read.

Next read C, R, O. The cover row C supplies finite open-cover entries, the refinement row R records containment into those entries, and the order row O counts only refined rows already visible through R.

The downstream consumers then inspect only attached finite rows. $\mathsf{RealSeparability}^{\uparrow}$ contributes density windows for the same cover samples, while $\mathsf{SimplicialComplex}^{\uparrow}$ contributes nerve rows for the same refined finite cover.

The rows H, T, P, N transport, replay, source, and name the route. Thus the compact-net refinement handoff is exhausted by the displayed chain and cannot export arbitrary open-cover compactness, selected dense sequences, external refinement oracles, or host nerve constructions. □

Theorem 1853.58 (CoveringDimension root cover density-nerve lock)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, every accepted root cover-dimension read must expose* K, E, C, R, O *before either the* $\mathsf{RealSeparability}^{\uparrow}$ *density row* S *or the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row* V *is consumed. Both rows must refer to the same finite refined cover displayed by* C, R, O*, so density windows and nerve data cannot be read as detached external witnesses.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverDensityNerveLock`

Proof. Project the root cover carrier of Definition 1853.55. The rows K, E, C, R, O display the compact metric source, finite epsilon-net refinements, finite open-cover entries, containment rows, and order ledger before the downstream rows S and V appear.

Apply the root cover NameCert obligations, Definition 1853.56. They make density-window compatibility and nerve-row compatibility part of the same local root cover surface rather than separate external certificates.

Use the compact-net refinement handoff, Definition 1853.57. It routes both $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{SimplicialComplex}^{\uparrow}$ through the same chain

$$K, E, C, R, O, H, T, P, N.$$

Thus any accepted density row S and nerve row V are attached to the same finite refined cover already exposed by C, R, O. Since no carrier coordinate supplies a second cover, selected dense subfamily, or host nerve construction, the two downstream rows remain locked to that displayed route. □

Theorem 1853.59 (CoveringDimension root cover RealSeparability simplicial lock)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, every accepted root-cover read that consumes a* $\mathsf{RealSeparability}^{\uparrow}$ *density window and a* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row must expose the same refined-cover route*

$$K, E, C, R, O$$

first. The density and nerve rows are then transported only by the shared structural rows H, T, P, N, so neither row can be replaced by a selected dense subfamily, a detached sampling schedule, or an external nerve construction.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverRealSeparabilitySimplicialLock`

Proof. Project the carrier through Definition 1853.55. The only root-cover route available before downstream sampling is the displayed compact metric source K, finite epsilon-net refinement E, finite cover row C, containment row R, and order ledger O.

Apply Definition 1853.56. Its density-window and nerve-row clauses attach $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{SimplicialComplex}^{\uparrow}$ to that local root surface rather than to independent witnesses.

Use Definition 1853.57. The handoff orders both downstream rows after the same finite refined cover and before the same structural rows H, T, P, N perform transport, replay, provenance, and local naming.

Thus the two consumers are locked to one refined-cover route. Since the carrier contains no coordinate for selected dense subfamilies or external nerves, no such object can be introduced after the shared route has been exposed. □

Theorem 1853.60 (CoveringDimension root cover public scope)**.** *Every public* $\mathsf{CoveringDimension}^{\uparrow}$ *root-cover consumer factors through the same finite chain*

$$\mathsf{CompactMetric}^{\uparrow}/\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{FiniteEpsilonNet}^{\uparrow} \longrightarrow C, R, O \longrightarrow \mathsf{RealSeparability}^{\uparrow}/\mathsf{SimplicialComplex}^{\uparrow} \longrightarrow H, T, P, N.$$

The public route gives compact metric source rows, metric observations, finite epsilon-net refinements, finite cover rows, containment rows, order counting, density-window rows, nerve rows, transport, replay, provenance, and local naming. It gives no arbitrary open-cover compactness theorem, selected dense subfamily, host refinement theorem, or external nerve construction.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverPublicScope`

Proof. First project the public consumer through the root carrier of Definition 1853.55. The only visible rows are the compact metric source K, finite epsilon-net row E, cover row C, containment row R, order ledger O, density row S, nerve row V, and the structural rows H, T, P, N.

Apply Definition 1853.56. It makes compact metric admission, metric observation, finite cover admission, finite-net refinement, density compatibility, nerve compatibility, order counting, and structural transport part of one local NameCert surface.

Use Definition 1853.57 to order the route from $\mathsf{CompactMetric}^{\uparrow}/\mathsf{Met}^{\uparrow}$ through $\mathsf{FiniteEpsilonNet}^{\uparrow}$ and C, R, O before the downstream $\mathsf{RealSeparability}^{\uparrow}/\mathsf{SimplicialComplex}^{\uparrow}$ rows are read.

Finally apply Definition 1853.58. The density and nerve rows are locked to the same finite refined cover, so projection from the root carrier exhausts the public route and leaves no coordinate for arbitrary open-cover compactness, selected dense subfamilies, host refinements, or external nerve constructions. □

Theorem 1853.61 (CoveringDimension root cover scope lock)**.** *Every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *root cover packet is scoped exactly by the compact finite-net rows, the* $\mathsf{RealSeparability}^{\uparrow}$ *density witnesses, and the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve rows displayed in*

$$K, E, C, R, O, S, V, H, T, P, N.$$

The compact side is read through $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*; the sampling side is read through* $\mathsf{RealSeparability}^{\uparrow}$*; and the combinatorial side is read through* $\mathsf{SimplicialComplex}^{\uparrow}$*. No arbitrary open-cover compactness theorem, selected subcover, host dimension primitive, detached dense family, or external nerve construction enters the root-cover scope.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverScopeLock`
↪ variant `BEDC.Derived.CoveringDimensionUp.CoveringDimensionRootCoverScopePackage`

Proof. Use the compact-net refinement handoff. It orders the compact metric source, metric observations, finite epsilon-net rows, finite cover rows, containment rows, and order ledger before any downstream density or nerve consumer can read the packet.

Apply the density-nerve lock. The rows S and V are attached to the same displayed refined cover, so the $\mathsf{RealSeparability}^{\uparrow}$ witnesses and $\mathsf{SimplicialComplex}^{\uparrow}$ nerve rows cannot be replaced by detached sampling or external combinatorial data.

Finally use the RealSeparability simplicial lock and the public-scope theorem. Together they route every public root-cover consumer through $K, E, C, R, O, S, V, H, T, P, N$, with H, T, P, N transporting, replaying, sourcing, and naming only those rows. The excluded compactness, selected-subcover, host-dimension, dense-family, and external-nerve primitives are absent from the carrier and therefore outside the scope. □

Theorem 1853.62 (CoveringDimension simplicial star-refinement scope)**.** *Every simplicial star-refinement read exported by* $\mathsf{CoveringDimension}^{\uparrow}$ *must expose the same finite root-cover route*

$$K, E, C, R, O \longrightarrow \mathsf{RealSeparability}^{\uparrow} \longrightarrow \mathsf{SimplicialComplex}^{\uparrow} \longrightarrow H, T, P, N$$

before any structural row is consumed. Thus the star-refinement read first displays the compact metric source row K, the finite epsilon-net row E, the finite cover row C, the containment row R, and the order ledger O; then it displays the $\mathsf{RealSeparability}^{\uparrow}$ *density window and the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row; only after those rows are visible may transport, replay, provenance, and local naming be read.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionSimplicialStarRefinementScope`

Proof. Start with the root cover carrier of Definition 1853.55. The carrier makes $K, E, C, R, O, S, V, H, T, P, N$ the only rows available to a root-cover consumer, with S serving the density window and V serving the nerve row.

Apply the root cover NameCert obligations of Definition 1853.56. They require compact metric admission, finite epsilon-net refinement, finite cover admission, containment, order counting, density compatibility, nerve compatibility, and structural transport to remain part of one local surface.

Use the compact-net refinement handoff and the root cover scope lock, Definition 1853.57 and Definition 1853.61. These force every downstream read to pass through K, E, C, R, O before the $\mathsf{RealSeparability}^{\uparrow}$ density window and $\mathsf{SimplicialComplex}^{\uparrow}$ nerve row are used.

Therefore a simplicial star-refinement read cannot skip directly to structural transport or to an external nerve construction. It exports only the displayed finite root-cover route followed by H, T, P, N, and no arbitrary open-cover compactness, selected dense subfamily, host star-refinement theorem, or detached simplicial primitive enters the scope. □

Theorem 1853.63 (CoveringDimension root scope ledger)**.** *Every accepted root-cover read of a* $\mathsf{CoveringDimension}^{\uparrow}$ *packet that can be consumed by a covering-dimension boundary theorem factors through the compact-net refinement row, the* $\mathsf{RealSeparability}^{\uparrow}$ *row, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row, the* $\mathsf{Met}^{\uparrow}/\mathsf{CompactMetric}^{\uparrow}$ *source rows, and the L10 tolerance route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The read is therefore scoped by displayed tolerance, readback, finite-cover, density, nerve, transport, replay, provenance, and local naming rows. It does not import ambient open-cover compactness, quotient real equality, selected dense subfamilies, host dimension primitives, or external nerve construction.

Proof. Project the root cover carrier of Definition 1853.55. The accepted read begins with $K, E, C, R, O, S, V, H, T,$ so compact metric, finite epsilon-net, finite cover, containment, order, density, nerve, transport, replay, provenance, and naming data are all visible before a boundary theorem can consume the packet.

Use the compact-net refinement handoff, Definition 1853.57. It places the $\mathsf{Met}^{\uparrow}/\mathsf{CompactMetric}^{\uparrow}$ source and $\mathsf{FiniteEpsilonNet}^{\uparrow}$ refinement before C, R, O, and it keeps the downstream $\mathsf{RealSeparability}^{\uparrow}$ and $\mathsf{SimplicialComplex}^{\uparrow}$ rows attached to the same finite refined cover.

Apply the density-nerve lock, RealSeparability simplicial lock, public scope, and scope lock theorems, Definition 1853.58, Definition 1853.59, Definition 1853.60, and Definition 1853.61. These force every public root-cover consumer to stay inside the displayed finite route.

Finally read the L10 tolerance rows through the dyadic tolerance ledger, regular rational sequence readback, and terminal real-name seal. Since those rows are displayed as tolerance and readback data rather than quotient real equality or ambient completeness, the resulting scope is exactly the finite boundary route and no stronger compactness or real-equality principle is exported. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootScopeLedger`

Theorem 1853.64 (CoveringDimension compact-net dependency route)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *read that starts from compact finite-net data must expose the finite cover row, refinement row, order-bound ledger,* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *source,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff when completion data are consumed,* BHist *carrier, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *before a dimension-bound consumer may inspect the order row. The route is a compact-net certificate over the accepted carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, not an arbitrary open-cover compactness theorem or selected refinement.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetDependencyRoute`

Proof. Project the carrier of Definition 1853.1. The compact metric source K, finite epsilon-net row E, finite cover row C, refinement row R, order-bound ledger O, and Lebesgue ledger L are displayed before the structural rows are available.

Read the compact-net source through Definition 1853.39. It makes the finite cover row a consequence of the displayed probe bundle and tolerance-indexed coverage witnesses rather than of host compactness.

Then route completion-facing uses through the existing MetricCompletion dependency children. The handoff may consume the same cover and refinement rows, but it does not add a second source for O or a selected completion representative.

Finally H, T, P, N preserve the projected route under transport, replay, provenance, and local naming. This gives the root dependency route as a finite compact-net certificate. □

Theorem 1853.65 (CoveringDimension refinement-order root route)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, the refinement-order root route requires the finite cover row* C*, refinement row* R*, order-bound ledger* O*,* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *source rows* K, E*, any* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff used by the read,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *to remain visible as one packet before the order bound is exported.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRefinementOrderRootRoute`

Proof. Start from Definition 1853.41. It makes each order count in O depend on the displayed parent and containment witnesses stored in R over the finite cover row C.

Use Definition 1853.42 to keep K, E, C, R, O, L in the compact metric route. If the consumer asks for completion-facing data, the MetricCompletion child route is inserted after the finite cover and refinement rows and before the order export.

The structural rows H, T, P, N then replay and name the same data. The order-bound ledger is therefore a projected root obligation of the displayed carrier, not parameter transport through an external multiplicity oracle. □

Theorem 1853.66 (CoveringDimension metric-completion root route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of a* $\mathsf{CoveringDimension}^{\uparrow}$ *bound must first read the finite cover row C, refinement row R, order-bound ledger O,* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *source K, E,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff,* BHist *carrier rows, componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row of the accepted packet. The completion boundary receives the displayed finite cover-refinement route; it receives no arbitrary open-cover compactness, host multiplicity theorem, quotient completion point, or selected representative.*

Proof. Begin with the compact-net dependency route above. It exposes $K, E, C, R, O, L, H, T, P, N$ before a dimension-bound consumer can inspect the order ledger.

Now insert the completion handoff through the existing MetricCompletion dependency route. The handoff is downstream of the compact metric and finite epsilon-net source rows, so it can replay the cover-refinement certificate but cannot replace it by an ambient completion object.

Finally apply the refinement-order root route. It keeps the order ledger attached to the same finite cover and refinement rows after the completion boundary is visible.

Thus the completion-facing export is a projection and replay of the accepted carrier. It is not parameter transport, quotient topology, selected representatives, or host-level general compactness. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionMetricCompletionRootRoute`

Theorem 1853.67 (CoveringDimension Cauchy forward link)**.** *Let $D = (K, E, C, R, O, L, H, T, P, N)$ be an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover-refinement packet. Any metric-completion or Real-facing consumer that uses the dimension bound must pass through the* $\mathsf{Cauchy}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite windows of chapter 3413 and chapter 301, together with the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows of chapter 129, before the* $\mathsf{Real}^{\uparrow}$*-facing bound is consumed.*

Thus the cover-refinement read reaches completion consumers as a finite route through compact finite nets, Cauchy windows, regular-rational readback, transport, replay, provenance, and local naming. It does not export open-cover compactness, selected subcovers, quotient completion points, host multiplicity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existing cover-refinement routes of Definition 1853.64 and Definition 1853.65. They expose the compact metric source K, finite-net row E, finite cover row C, refinement row R, order-bound ledger O, compact Lebesgue row L, and structural rows H, T, P, N before the dimension bound can be read.

Attach the Cauchy and regular-rational windows. A completion-facing consumer reads the Cauchy route through chapter 3413 and the regular-rational readback through chapter 301; these rows supply finite windows and tolerance ledgers, not quotient completion points or selected limits.

Now use the compact metric finite-net source of chapter 129. The finite-net row determines the sampled cover data that the refinement and order ledgers may consume, so the Real-facing dimension bound remains downstream of compact finite data.

Finally exhaust the packet rows. The consumer has only compact finite nets, Cauchy windows, RegSeqRat readback, cover refinement, order ledger, transport, replay, provenance, and local naming. Open-cover compactness, selected subcovers, quotient completion points, host multiplicity, and ambient Real completeness have no row in the accepted carrier. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCauchyForwardLink`

Definition 1853.68 (CoveringDimension finite cover carrier)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet, the finite cover carrier is the displayed* BHist *subpacket*

$$F = (K, E, M, C, R, O, S, A, H, T, P, N)$$

cut out of the local carrier of Definition 1853.1. Here K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* E *is the finite-net row,* M *is the* $\mathsf{Met}^{\uparrow}$ *distance-read ledger used by the cover tests,* C *is the finite* BHist *cover row,* R *is the refinement membership row,* O *is the order-bound ledger,* S *is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row,* A *records any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal rows consumed by a metric read, and* H, T, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *data.*

The carrier is finite at every exposed face. It contains no arbitrary open cover, selected subcover, host multiplicity oracle, quotient completion point, or ambient compactness theorem.

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverCarrier`

Theorem 1853.69 (CoveringDimension finite cover admission)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *read that admits finite cover data must first expose the finite cover carrier of Definition 1853.68. In particular, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows* K, E*, the* $\mathsf{Met}^{\uparrow}$ *distance reads* M*, the finite* BHist *cover row* C*, the refinement row* R*, the order ledger* O*, any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal rows* A*, and the structural rows* H, T, P, N *are all visible before a dimension-bound consumer may read an order conclusion.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier finite cover admission`

Proof. Project the accepted carrier through Definition 1853.1. The compact source, finite-net row, cover row, refinement row, order ledger, and structural rows are already part of the displayed CoveringDimension packet.

Insert the metric and Real-facing reads only as rows of the same finite carrier. The $\mathsf{Met}^{\uparrow}$ ledger records distance tests for cover membership, while the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows are read only when those tests are Real-facing.

Since no arbitrary open-cover coordinate occurs in F, admission is a finite row projection. The dimension-bound consumer receives the displayed finite cover route and none of the excluded host principles. □

Theorem 1853.70 (CoveringDimension order refinement route)**.** *For a finite cover carrier* $F = (K, E, M, C, R, O, S, A, H, T, P, N)$*, the order-bound row* O *is readable only after each member counted by* O *has been traced through the finite cover row* C *and the refinement membership row* R*. The route uses the same* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{Met}^{\uparrow}$ *distance reads, and* BHist *cover entries that admitted the carrier, and it exports no selected refinement or host overlap oracle.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionOrderRefinementRoute`

Proof. Start with the finite cover admission theorem. It places K, E, M, C, R before O in the public route.

For every refined member, the row R names a parent in the finite cover row C and records the containment witness used by the distance-read ledger M. Thus the overlap count in O is computed over displayed BHist rows.

The structural rows H, T, P, N transport, replay, source, and name that finite computation. They do not add a second order source, so the order refinement route remains a finite CoveringDimension obligation. □

Theorem 1853.71 (CoveringDimension simplicial nerve handoff)**.** *Every* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve read consumed by a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier must use the same finite cover and refinement data*

$$C, R, O \longmapsto S$$

after the compact finite-net rows K, E and metric distance ledger M have been displayed. The nerve row S receives vertices and incidence rows from the finite cover carrier; it receives no classical nerve theorem, selected locally finite cover, or ambient topological realization.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverSimplicialNerveHandoff`

Proof. Read the carrier F. The cover row C supplies the finite vertex list for the nerve, while R supplies the refinement memberships that determine incidence.

The order ledger O bounds the displayed overlap count before the nerve row is consumed. Therefore the $\mathsf{SimplicialComplex}^{\uparrow}$ row S is downstream of the same finite cover-refinement route, not a detached simplicial object.

The compact finite-net and metric rows K, E, M remain attached to the cover tests. Hence the handoff to the nerve surface is constructive row replay over the accepted carrier. □

Theorem 1853.72 (CoveringDimension Cauchy metric scope)**.** *When a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier is read by a Cauchy, RegSeqRat, or Real-facing metric consumer, the consumer may use only the displayed route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the carrier rows K, E, M, A. The finite cover, refinement, order, and nerve rows C, R, O, S remain the only dimension-facing rows, and no quotient completion point, selected limit, host metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is exposed.*

Proof. Project K, E, M, A from Definition 1853.68. These are the only rows through which metric or Real-facing reads enter the finite cover route.

The regular rational and Real seal data in A are downstream of the metric distance ledger M. They can certify the numerical side of a cover test, but they do not replace the finite cover row C or refinement row R.

Finally use the order refinement and nerve handoff routes. They keep C, R, O, S as the dimension-facing surface, so completion or Real-facing metric data remain scoped to the displayed finite carrier. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCauchyMetricScope`

Theorem 1853.73 (CoveringDimension finite cover root metric-completion route)**.** *Every metric-completion read of a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier*

$$F = (K, E, M, C, R, O, S, A, H, T, P, N)$$

factors through the displayed finite-cover route $K, E, M, C, R, O, S, A, H, T, P, N$*. The completion-facing rows may inspect the compact source* K*, finite-net row* E*, metric distance ledger* M*, and regular or Real seal row* A*, but the dimension conclusion remains routed through the finite cover row* C*, refinement row* R*, order ledger* O*, nerve row* S*, and structural rows* H, T, P, N*. No quotient completion point, selected Cauchy limit, arbitrary open cover, selected subcover, host metric equality, or ambient compactness principle is exported.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootMetricCompletionRoute`

Proof. Start with Definition 1853.72. It places every completion-facing metric read inside the rows K, E, M, A of the finite cover carrier.

Then apply finite cover admission and order refinement. The rows C, R, O, S remain the only dimension-facing route, so the completion read cannot replace them by a quotient point, a selected limit, or an external open-cover family.

The transport, replay, provenance, and naming rows H, T, P, N keep both readings attached to the same displayed carrier. Thus metric completion consumes the finite-cover root rather than an ambient completion construction. □

Theorem 1853.74 (CoveringDimension finite cover root Real separability route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *separability read that consumes a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier*

$$F = (K, E, M, C, R, O, S, A, H, T, P, N)$$

must pass through the same finite-cover root $K, E, M, C, R, O, S, A, H, T, P, N$*. The separability consumer may read rational or regular Real-facing data only through the metric ledger* M *and the seal row* A*, while cover membership, refinement, order, and nerve data remain the displayed rows* C, R, O, S*. The route exports no arbitrary dense sequence, selected real representative, quotient completion point, host open ball, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier and read the finite-net and metric rows K, E, M. These rows are the only source from which a Real-facing separability consumer can obtain numerical cover tests.

The seal row A is downstream of those tests and does not introduce a separate dense sequence or selected Real representative. By the finite cover admission theorem, cover data still enter through C and refinement through R.

Finally the order and nerve rows O, S are replayed through H, T, P, N. Hence Real separability reads factor through the displayed finite-cover root and do not expose any ambient dense-set or completeness principle. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootRealSeparabilityRoute`

Theorem 1853.75 (CoveringDimension finite cover root Lebesgue ledger route)**.** *Every finite Lebesgue-ledger read attached to a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier*

$$F = (K, E, M, C, R, O, S, A, H, T, P, N)$$

is computed from the displayed route $K, E, M, C, R, O, S, A, H, T, P, N$*. The ledger may consume the compact finite-net rows* K, E*, the metric test row* M*, and any regular or Real seal row* A *only to*

certify finite cover membership; its covering, refinement, order, and nerve obligations are exactly C, R, O, S transported and replayed by H, T, P, N. It exports no arbitrary open cover, selected Lebesgue number, host compactness theorem, chosen finite subcover, quotient completion point, or ambient Real-completeness witness.

Proof. Use finite cover admission to display the rows $K, E, M, C, R, O, S, A, H, T, P, N$. A finite Lebesgue ledger can read K, E, M, A only as the numerical side of membership tests.

Order refinement then forces every counted overlap through C and R, and the simplicial nerve handoff routes incidence through S. Thus the ledger consumes the same finite cover, refinement, order, and nerve rows as the dimension packet.

The structural rows H, T, P, N transport, replay, source, and name that computation. Since no selected Lebesgue number, chosen subcover, host compactness theorem, quotient completion point, or ambient Real-completeness witness is present in the carrier, the ledger factors through the finite-cover root. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootLebesgueLedgerRoute`

Theorem 1853.76 (CoveringDimension finite cover root Cauchy selection route)**.** *Every Cauchy or metric-completion cover-selection read attached to a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite cover carrier*

$$F = (K, E, M, C, R, O, S, A, H, T, P, N)$$

factors through the displayed finite-cover route. The compact metric and metric rows K, E, M are displayed before any Cauchy read, the regular and Real-facing row A is displayed before any completion-facing numerical seal, and the finite cover, refinement, order, and nerve rows C, R, O, S are displayed before selection data may be inspected. The route exports no quotient completion point, selected Cauchy limit, arbitrary open cover, host metric equality, or ambient compactness principle.

Proof. Use finite cover admission to expose $K, E, M, C, R, O, S, A, H, T, P, N$. This places compact metric data, metric distance tests, finite cover membership, refinement rows, order bounds, and nerve rows in the finite carrier before any selection read.

Apply the Cauchy metric scope and metric-completion route. They force completion-facing reads to remain inside K, E, M, A and prevent a quotient completion point or selected Cauchy limit from replacing the displayed finite-cover rows.

Use the Real separability route for the numerical seal side. It keeps regular rational and Real-facing data downstream of the metric ledger, so arbitrary open covers, host metric equality, and ambient compactness never enter the exported CoveringDimension root. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootCauchySelection`

Theorem 1853.77 (CoveringDimension finite-cover scope grounding)**.** *Each* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-cover root obligation must expose the* $\mathsf{Met}^{\uparrow}$ *ball row,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row,* $\mathsf{Real}^{\uparrow}$ *radius seal,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window row before any covering-dimension consumer reads a refinement conclusion. The refinement remains a finite-cover root route over the displayed rows $K, E, M, C, R, O, S, A, H, T, P, N$.*

Proof. Project the finite-cover carrier of Definition 1853.68 and the admission route of Definition 1853.69. These expose the compact source, finite-net row, metric distance ledger, finite cover row, refinement row, order ledger, and structural rows before the dimension-facing conclusion is read.

Use the metric-completion, Real-separability, and Lebesgue-ledger routes, Definition 1853.73, Definition 1853.74, and Definition 1853.75. Together they force radius seals, regular readback, dyadic tolerances, and dense-window data to remain inside the same finite-cover root.

The dependency routes in this subdirectory keep metric, compact-metric, Real, RealSeparability, RegSeqRat, and DyadicRatCore reads explicit. Since no row names open-cover compactness, a selected subcover, a host dimension theorem, or quotient real equality, every refinement consumer is grounded in the displayed finite BEDC scope. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverScopeGrounding`

Definition 1853.78 (CoveringDimension finite cover order carrier)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet, the finite-cover order carrier is the displayed subpacket*

$$O_{\text{fin}} = (K, E, M, R, Q, S, A, H, T, P, N)$$

where K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* E *is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *cover support,* M *is the* $\mathsf{Met}^{\uparrow}$ *distance-read ledger,* R *records* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison rows,* Q *records the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows used by cover tests,* S *is the finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row, and* A *is the finite order table assigning each displayed cover row its overlap count. The rows* H, T, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local NameCert data.*

The carrier admits only displayed finite cover rows and their order table. It has no coordinate for arbitrary open-cover compactness, ambient dimension theory, or choice-selected refinements.

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderCarrier`

Theorem 1853.79 (CoveringDimension finite cover order refinement)**.** *Every refinement read from* $O_{\text{fin}} = (K, E, M, R, Q, S, A, H, T, P, N)$ *must preserve the displayed order table* A*. A refined cover member is accepted only when its parent cover row is named by* E*, its metric containment test is read through* M, R, Q*, and its nerve incidence is replayed through* S *before the order count is exported.*

Proof. Begin with the finite epsilon-net support E. It lists the only cover rows available to a refinement consumer, so a refined member must name a parent in that finite table.

The containment test is then read through the metric ledger M and the regular rational and Real seal rows R, Q. These rows can compare displayed radii and distances, but they cannot select a new cover outside the finite support.

Finally replay the incidence row through the simplicial nerve S and consult the order table A. Since every step is indexed by the displayed finite cover rows, the refinement preserves the order carrier and exposes no host dimension theorem or choice-selected refinement. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderRefinement`

Theorem 1853.80 (CoveringDimension finite cover order non-escape)**.** *Every downstream order-bound consumer of* $O_{\text{fin}} = (K, E, M, R, Q, S, A, H, T, P, N)$ *remains inside the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SimplicialComplex}^{\uparrow}.$$

The consumer may replay displayed cover rows, metric comparisons, Real seal checks, nerve incidence, transport, provenance, and local naming, but it cannot invoke arbitrary open-cover compactness, ambient dimension theory, or choice-selected refinements.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderNonescape`

Proof. Project the consumer request to the rows of O_{fin}. The only source of cover data is E over the compact metric row K, and the only numerical tests are the metric, regular rational, and Real seal rows M, R, Q.

The nerve row S and order table A are downstream of those finite tests. They can replay incidence and count overlaps over displayed rows, but they do not introduce a second cover source.

The structural rows H, T, P, N preserve the same route under transport, replay, provenance, and naming. Because the carrier lacks coordinates for open-cover compactness, host dimension theory, and choice-selected refinement, the order-bound read cannot escape the finite carrier. □

Theorem 1853.81 (CoveringDimension finite cover order root)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-cover order read first exposes the root rows*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SimplicialComplex}^{\uparrow}$$

before the order table is consumed. Equivalently, an order-bound consumer may reach the table A of $O_{\text{fin}} = (K, E, M, R, Q, S, A, H, T, P, N)$ only after reading the compact metric source K, finite epsilon-net support E, metric ledger M, regular rational comparison rows R, terminal Real seal rows Q, and simplicial nerve row S.

Proof. Project the read to the finite-cover order carrier of Definition 1853.78. The coordinates before the table are exactly K, E, M, R, Q, S, followed by the order table A and the structural rows.

The refinement theorem Definition 1853.79 forces every refined member to name a parent cover row in E, test containment through M, R, Q, and replay incidence through S before an order count is exported.

The nonescape theorem Definition 1853.80 then prevents the consumer from bypassing those root rows through arbitrary open-cover compactness, ambient dimension theory, or choice-selected refinements. Thus the order table is reachable only after the displayed root route has been exposed. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderRoot`

Theorem 1853.82 (CoveringDimension finite cover order scope)**.** *Every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-cover order consumer must display the source rows*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SimplicialComplex}^{\uparrow}$$

before the order table of $O_{\text{fin}} = (K, E, M, R, Q, S, A, H, T, P, N)$ is public. The table is scoped only to those six source rows together with the componentwise hsame *transport row H, the displayed* Cont *replay row T, the* Pkg *provenance row P, and the local* NameCert *row N.*

Proof. Project the carrier of Definition 1853.78. It displays exactly the compact metric row K, finite epsilon-net support E, metric ledger M, regular rational comparison rows R, terminal Real seal rows Q, simplicial nerve row S, order table A, and structural rows H, T, P, N.

Apply the refinement theorem Definition 1853.79. It requires each refined member to read the finite support, metric containment test, rational and Real comparison rows, and simplicial incidence row before exporting an order count.

Use the nonescape theorem Definition 1853.80 and the root theorem Definition 1853.81. Together they rule out arbitrary open-cover compactness, ambient dimension theory, and choice-selected refinements, so the public order table remains scoped to the displayed root rows and the structural hsame, Cont, Pkg, and NameCert rows. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderScope`

Theorem 1853.83 (Finite covering-dimension sibling dependency ledger)**.** *Every finite covering-dimension read carried by the* $\mathsf{CoveringDimension}^{\uparrow}$ *family is admitted only through the sibling ledger*

$\mathsf{CompactMetric}^{\uparrow}, \mathsf{FiniteEpsilonNet}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SimplicialComplex}^{\uparrow}, \mathsf{CompactCoverLebesgueLedger}^{\uparrow}, \mathsf{Rea}$

The finite cover rows, refinement rows, and order table of O_{fin} *remain the controlling dependency surface for that read. A downstream consumer may use those siblings to inspect compact finite-net support, metric containment, rational or Real seals, nerve incidence, Lebesgue-radius compatibility, Real-separability windows, and completion-facing scale data, but it cannot replace the displayed finite cover/order ledger by open-cover compactness, a selected refinement, host dimension data, or ambient completeness.*

Proof. Begin with the finite-cover order carrier of Definition 1853.78. It exposes the compact source row, finite epsilon-net support, metric ledger, regular-rational comparison rows, Real seal rows, simplicial nerve row, finite order table, and structural transport, replay, provenance, and naming rows.

Apply Definition 1853.82. This forces every public order-table read to pass through the displayed source rows before the table is public. The finite cover and refinement information therefore remains a rowwise BHist ledger, not a host overlap oracle.

Now read the surrounding sibling routes: the finite-cover root admission and Lebesgue-ledger route bind compact-cover radius evidence, the Real-separability route binds rational windows, and the metric-completion route binds completion-facing scale data. These routes attach their evidence to the same O_{fin} surface and do not introduce another cover selector.

Thus the sibling dependency ledger is exactly the finite BEDC route listed in the statement. It supports finite covering-dimension consumers while excluding arbitrary open-cover compactness, selected refinements, host dimension predicates, and ambient completion principles. □

Definition 1853.84 (Covering dimension finite cover-order carrier)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet*

$$D = (K, E, C, R, O, L, H, T, P, N),$$

the finite cover-order carrier is the displayed root subpacket

$$F_{\mathrm{dim}} = (K, E, C, R, O, L, M, S, Q, A, H, T, P, N).$$

Here K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* E *is the finite compact metric cover-support row,* C *is the displayed finite cover,* R *is the finite refinement map,* O *is the order-bound ledger,* L *is the finite Lebesgue-number and compact-cover ledger,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-facing metric row,* S *is the* $\mathsf{RealSeparability}^{\uparrow}$ *finite density row,* Q *records the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks used by cover tests, and* A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. The rows* H, T, P, N *transport, replay, source, and name the same finite packet.*

The carrier has no coordinate for arbitrary open-cover compactness, choice-selected subcovers, quotient equality, selected dense points, host dimension theory, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverOrderAdmission`

Theorem 1853.85 (Covering dimension finite refinement obligations)**.** *Every accepted finite cover-order carrier* F_{dim} *discharges its refinement obligations through the displayed rows* $K, E, C, R, O, L, M, S, Q, A, H, T$ *A downstream cover consumer must read compact metric cover rows, finite Lebesgue-number or compact-cover ledger rows, MetricCompletion rows, RealSeparability rows, RegSeqRat and DyadicRatCore comparison rows, and the terminal Real seal before it exports the order-bound ledger.*

Proof. Project the carrier of Definition 1853.84. The rows K, E, C, R, O, L are the compact source, finite cover, refinement, order, and finite Lebesgue-ledger coordinates already visible in the parent $\mathsf{CoveringDimension}^{\uparrow}$ packet.

Read the completion and separability rows before the order bound is exported. The $\mathsf{MetricCompletion}^{\uparrow}$ row M supplies only displayed completion-facing metric data, while the $\mathsf{RealSeparability}^{\uparrow}$ row S supplies finite rational or dyadic density witnesses; neither row chooses a subcover or a dense point outside the packet.

The comparison rows Q, A then attach the RegSeqRat, DyadicRatCore, and Real seal reads to the same finite cover tests. Transport, replay, provenance, and local naming through H, T, P, N preserve that route.

Thus the refinement obligation is exactly the finite row family named in the statement. Since no excluded open-cover, choice, quotient, host-dimension, or completeness coordinate appears in the carrier, none can be exported by a downstream consumer. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteRefinementObligations`

Theorem 1853.86 (Covering dimension Real-separability refinement route)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of a* $\mathsf{CoveringDimension}^{\uparrow}$ *finite refinement row must factor through*

$$\mathsf{CompactMetric}^{\uparrow},\ C,\ R,\ \mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{RealSeparability}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}$$

The route exposes compact metric cover rows, finite cover order rows, finite Lebesgue-number and refinement siblings, completion-facing metric rows, finite density rows, regular-rational and dyadic comparison rows, and the Real seal boundary before any dimension-bound order row is consumed. It exports no arbitrary open-cover compactness, choice-selected subcover, quotient equality, selected approximating sequence, selected dense point, host dimension theorem, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityRefinementRoute`

Proof. Start with the finite refinement obligations of Definition 1853.85. They force the consumer to open the compact source K, finite cover C, refinement map R, order ledger O, and finite Lebesgue ledger L as displayed cover-order rows.

Next read the completion and density dependencies. The $\mathsf{MetricCompletion}^{\uparrow}$ row is a finite completion-facing metric handoff, and the $\mathsf{RealSeparability}^{\uparrow}$ row is a finite density handoff; both are inserted before the order ledger is consumed, so neither can act as an external selector.

Then apply the RegSeqRat, DyadicRatCore, and Real seal rows. They provide the comparison and terminal seal data used by the cover tests, but they do not add quotient Real equality or ambient completeness.

The structural rows H, T, P, N replay the same finite route for downstream cover and compactness chapters. Because each step is a displayed carrier row, the resulting route refuses open-cover compactness, choice-selected subcovers, quotient equality, selected approximating sequences, selected dense points, host dimension theorems, and ambient Real completeness. □

Theorem 1853.87 (Covering dimension cover-order directedness)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet with finite cover-order carrier* $F_{\mathrm{dim}} = (K, E, C, R, O, L, M, S, Q, A, H, T, P, N)$*, any two displayed finite cover refinements over the same compact metric source row* K *admit a common later finite refinement whose order row is still bounded by the same* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier. The common refinement is read through the displayed finite cover rows, refinement maps, order ledger, finite Lebesgue ledger, transport, replay, provenance, and local naming; it introduces no arbitrary*

open-cover compactness, choice-selected subcover, quotient equality, host dimension theorem, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCoverOrderDirectedness`

Proof. Start with the finite refinement obligations of Definition 1853.85. They keep both displayed refinements inside the same carrier rows K, E, C, R, O, L before any downstream dimension read is public.

Form the common later refinement by reading the displayed finite cover rows and the refinement map rows together. The order ledger O supplies the same carrier-bound order control because both input refinements are attached to the same compact metric source and finite Lebesgue ledger.

Transport and replay through H, T preserve the common refinement as a later finite row, and P, N source and name it locally. Since the carrier contains no arbitrary open-cover, choice-selected, quotient, host-dimension, or ambient-completeness coordinate, directedness stays inside the displayed finite cover-order surface. □

Theorem 1853.88 (Covering dimension compact-net Real-separability handoff)**.** *Every public finite-cover read of a* $\mathsf{CoveringDimension}^{\uparrow}$ *packet that uses compact-net rows must also read the* $\mathsf{RealSeparability}^{\uparrow}$ *finite density row before the dimension order row is exported. In carrier notation, the compact-net support in* K, E, C, L *and the Real-separability row* S *jointly supply the finite cover handoff through* M, Q, A, H, T, P, N*, so no dimension consumer can bypass the compact-net and Real-separability route by selecting a dense set, a subcover, quotient Real equality, a host dimension theorem, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetRealSeparabilityHandoff`

Proof. Apply the Real-separability refinement route of Definition 1853.86. It places compact metric cover rows, finite cover rows, and finite Lebesgue-ledger rows before completion-facing metric data and RealSeparability data are consumed.

The compact-net contribution is displayed by K, E, C, L, while S supplies the finite density handoff. The rows M, Q, A then attach completion-facing metric reads, RegSeqRat and DyadicRatCore comparison data, and the terminal Real seal to the same finite cover test.

Only after this route is visible may the order ledger O be exported. Transport, replay, provenance, and local naming through H, T, P, N preserve the same handoff, and the missing selected-dense-set, selected-subcover, quotient, host-dimension, and ambient-completeness objects cannot be read from the carrier. □

Theorem 1853.89 (Covering dimension finite-cover obligations)**.** *Let* $D = (K, E, C, R, O, L, H, T, P, N)$ *be an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet. Every finite-cover obligation exported by* D *must first display the* $\mathsf{CompactMetric}^{\uparrow}$ *source row* K*, the* $\mathsf{Met}^{\uparrow}$ *observation route carried by that source, the finite cover row* C*, the refinement row* R*, the order-bound row* O*, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *transport, replay, provenance, and naming rows* H, T, P, N*. If the obligation is read by* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$*, those rows are inserted as finite cover-support rows before the order-bound export and not as ambient compactness or completeness principles.*

Proof. Project the accepted carrier. The rows K, E, C, R, O, L are the finite compact-metric cover, finite-net, refinement, order, and compact-cover ledger rows already present in the $\mathsf{CoveringDimension}^{\uparrow}$ packet.

The metric and downstream support rows are read before export. A $\mathsf{Met}^{\uparrow}$ observation is available only through the compact metric source row, while $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$,

and $\mathsf{MetricCompletion}^{\uparrow}$ supply finite density, finite net, or completion-facing support rows attached to the same cover.

Finally, H, T, P, N transport, replay, source, and name the same finite packet. Since arbitrary open-cover compactness, selected limits, host equality, and ambient completeness are not carrier coordinates, the finite-cover obligation cannot escape the displayed route. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverObligations`

Theorem 1853.90 (Covering dimension Real-separability refinement)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *refinement read of an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet is stable only as a finite-cover route through*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{RealSeparability}^{\uparrow},\ C,\ R,\ O,\ L,\ H,\ T,\ P,\ N.$$

The refinement may tighten finite cover members and rational or dyadic density windows, but it must preserve the same compact metric source, finite epsilon-net support, refinement membership row, order-bound ledger, compact-cover ledger, and local naming surface. It exports no selected dense sequence, arbitrary open cover, quotient point equality, host dimension theorem, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Start with Definition 1853.89. It places compact metric source data, metric observations, finite cover rows, refinement membership, and order ledgers before any downstream refinement can be consumed.

Read the Real-separability route as finite density data. The $\mathsf{RealSeparability}^{\uparrow}$ rows may refine rational or dyadic windows, but the compact metric source K and finite epsilon-net row E remain the cover support used by C, R, O, L.

The structural rows H, T, P, N preserve this cover route under componentwise transport and displayed replay. Because no selected dense sequence, arbitrary open cover, quotient point equality, host theorem, selected limit, or ambient completeness row is present, refinement stability is exactly the finite route stated above. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityRefinement`

Theorem 1853.91 (Covering dimension compact-net order route)**.** *Every compact-net order read of an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet must pass through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *support rows before a finite cover order bound is exported. The order row may be consumed only after the finite cover row, refinement row, compact-cover ledger, completion-facing metric row, density row, transport row, replay row, provenance row, and local* NameCert *row have been displayed. The route supplies carrier admission and refinement stability, and it does not import open-cover compactness, selected subcovers, selected limits, host equality, quotient completion, or ambient Real completeness.*

Proof. Use the finite-cover obligations of Definition 1853.89. They expose the compact metric source, metric observations, finite cover, refinement row, order ledger, and structural rows as the carrier boundary.

Insert the compact-net support. The $\mathsf{FiniteEpsilonNet}^{\uparrow}$ row supplies finite centers and coverage witnesses, while the $\mathsf{RealSeparability}^{\uparrow}$ row supplies finite density windows; both are read before the order row is public.

For completion-facing consumers, read the $\mathsf{MetricCompletion}^{\uparrow}$ row as displayed metric support for the same finite cover route. It does not add a completed point or quotient completion coordinate.

The order export is therefore downstream of the finite cover, compact-net, separability, completion, transport, replay, provenance, and naming rows. Since the excluded host and choice objects have no carrier slot, the route remains inside finite-cover compactness and selected-limit nonescape. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetOrderRoute`

Definition 1853.92 (CoveringDimension compact net order carrier)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet, the compact-net order carrier is the finite subpacket*

$$O_{\mathrm{net}} = (K, M, E, U, R, Q, S, A, H, T, P, N)$$

where K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* M *is the* $\mathsf{Met}^{\uparrow}$ *distance-read ledger,* E *is the finite compact-net cover support,* U *is the uniform radius table,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row,* Q *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* S *is the finite nerve row, and* A *is the order table counting displayed overlap rows. The structural rows* H, T, P, N *are the componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows.*

The carrier has no coordinate for arbitrary open-cover compactness, ambient dimension theory, choice-selected subcovers, or host topological refinement.

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetOrderCarrier`

Theorem 1853.93 (CoveringDimension compact net order admission)**.** *Every compact-net order read admitted by* $O_{\mathrm{net}} = (K, M, E, U, R, Q, S, A, H, T, P, N)$ *must be generated from the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *row* K*, the* $\mathsf{Met}^{\uparrow}$ *distance ledger* M*, the finite net support* E*, and the uniform radius table* U*. The exported order count is then replayed through* R, Q, S, A, H, T, P, N *and cannot be replaced by an arbitrary compactness theorem or a choice-selected refinement.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetOrderAdmission`

Proof. Project the read to the compact metric row K and the metric ledger M. These rows are the only sources from which distances and bounded cover tests can be read.

Next consume the finite net support E and uniform radius table U. The regular rational and Real seal rows R, Q compare the displayed radii and distances without selecting a new cover.

Finally replay nerve incidence through S and consult the order table A. The transport, replay, provenance, and naming rows H, T, P, N preserve that same finite route, so the admitted order count remains compact-net data. □

Theorem 1853.94 (CoveringDimension RealSeparability cover pullback)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *cover pullback used by the compact-net order carrier must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row* R *and the* $\mathsf{Real}^{\uparrow}$ *source seal row* Q *before it reads the finite net support* E*, the uniform radius table* U*, the nerve row* S*, and the order table* A*. The pullback therefore records a displayed cover schedule, not a countable-choice enumeration, quotient stream equality, or ambient separability theorem.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityCoverPullback`

Proof. Begin with the Real-facing comparison rows R, Q. They provide only regular rational comparisons and terminal Real seal checks for the finite cover tests.

The cover itself is then pulled back to the compact-net rows E, U over the compact metric and metric ledgers K, M. This pins each cover member to a displayed radius and a displayed source row.

Use the nerve and order rows S, A to replay the overlap count. Since the carrier lacks coordinates for countable choice, quotient stream equality, and ambient separability, the RealSeparability consumer receives only the finite cover schedule carried by O_{net}. □

Theorem 1853.95 (CoveringDimension compact net order scope)**.** *For an accepted compact-net order carrier*

$$O_{\mathrm{net}} = (K, M, E, U, R, Q, S, A, H, T, P, N),$$

any order count read from O_{net} is scoped by the displayed $\mathsf{CompactMetric}^{\uparrow}$ *row K, the* $\mathsf{Met}^{\uparrow}$ *distance ledger M, the finite net support E, the uniform radius table U, the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row R, the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q, the finite nerve row S, and the order table A. The structural rows H, T, P, N may transport, replay, source, and name that finite route, but they do not promote the count to arbitrary open-cover dimension theory.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetOrderScope`

Proof. Project the carrier of Definition 1853.92. The compact metric, metric, finite-net, uniform-radius, regular rational, Real, nerve, and order rows are all displayed entries of O_{net}.

Apply Definition 1853.93. It forces the order read to be generated from K, M, E, U before the replay rows R, Q, S, A, H, T, P, N expose the count.

Finally use Definition 1853.94. The Real-facing comparison remains on the finite route through R, Q, E, U, S, A, so the carrier supplies a compact-net order count rather than an ambient open-cover dimension theorem. □

Theorem 1853.96 (CoveringDimension cover refinement monotonicity)**.** *Let $D = (K, E, C, R, O, L, H, T, P, N)$ be an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier and let C' be a displayed finite refinement row read from the same* BHist *packet through the finite cover order carrier of Definition 1853.78. If every member of C' names a parent in C, every containment witness is a compact-cover or* MetricDistanceWitness *row transported by H, and the order table of C' is bounded by the displayed order ledger O, then replacing C by C' is monotone for the local NameCert ledger: no accepted read can increase the exported finite cover order or escape the* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *rows.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCoverRefinementMonotonicity`

Proof. Project the root carrier of Definition 1853.55 and the finite order carrier of Definition 1853.78. The only cover rows visible to a refinement consumer are the displayed BHist cover entries, their compact-metric source rows, and the metric containment witnesses.

The refinement map names a parent for every refined member before the order table is read. Thus each overlap counted for C' is already attached to a row counted by O, and componentwise hsame transport only replays the same containment witnesses.

Finally apply the finite-cover order non-escape route of Definition 1853.80. Since there is no coordinate for a host refinement theorem, selected open subcover, or external multiplicity oracle, the refined cover preserves the displayed order bound and remains inside the local NameCert ledger. □

Theorem 1853.97 (CoveringDimension Lebesgue-ledger stability)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier $D = (K, E, C, R, O, L, H, T, P, N)$, suppose a finite cover read uses a compact-cover Lebesgue ledger row L' whose positive-radius and containment entries are* hsame*-transported from L and whose compact-cover rows are read through K, E, C, R. Then the finite cover order exported by O*

is stable under replacing L by L': the accepted read still factors through compact metric admission, finite cover order, displayed replay, Pkg *provenance, and the same* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *nonescape boundary.*

Proof. Start with the Lebesgue-ledger handoff of Definition 1853.6. It places every compact-cover ledger read downstream of the compact-net root K, E and the finite cover-refinement rows C, R, O.

The transported ledger L' has the same positive-radius and containment entries as L under H. Therefore it can change only the presentation of the finite ledger row, not the compact metric source, the cover entries, the refinement parents, or the order table.

Use Definition 1853.56 to close the route. The structural rows T, P, N replay, source, and name the same finite data, so the ledger replacement cannot add arbitrary open-cover compactness, a selected dense subfamily, or a host Lebesgue-number theorem. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionLebesgueLedgerStability`

Theorem 1853.98 (CoveringDimension metric completion cover transport)**.** *Let an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier $D = (K, E, C, R, O, L, H, T, P, N)$ be read by a metric-completion consumer through the dependency route of chapter 1506 and the separability route of Definition 1853.25. If each completion-side cover member is backed by a source* BHist *cover row, a* MetricDistanceWitness *or compact-cover containment row,* hsame *transport, and the finite cover order ledger O, then completion transport preserves the CoveringDimension NameCert boundary: the consumer receives the same finite cover order and no quotient completion, selected limit, or external dimension predicate.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCarrier metric completion cover transport`

Proof. Read the compact-net refinement handoff of Definition 1853.57. It sends compact metric and metric rows into finite epsilon-net, cover, refinement, order, Real-separability, and nerve rows before the structural rows are consumed.

Metric completion can observe the transported cover only through the source BHist rows and their metric or compact-cover containment witnesses. The hsame row keeps those witnesses aligned with the source packet, while O remains the only finite order ledger.

The dependency route through chapter 1506 and the Real-separability children therefore transports a finite cover certificate, not an ambient completion theorem. Since the carrier lacks coordinates for quotient completions, selected limits, and external dimension predicates, the local $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ ledger remains nonescaping. □

Theorem 1853.99 (CoveringDimension root cover refinement tree)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier $D = (K, E, C, R, O, L, H, T, P, N)$, any root read that exports a finite cover-refinement tree must display the* BHist *root K, E, C before it displays refinement successors. Each successor row is named by R, transported by the componentwise* hsame *row H, replayed by the displayed* Cont *row T, and sourced by the same* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row N. Thus the exported tree is a finite cover-refinement object over the existing compact metric and metric rows, not a selected open-cover subtree or an external refinement theorem.*

Proof. Project the accepted carrier from Definition 1853.1. The source rows K, E, C are the only compact metric, finite-net, and finite cover rows available before refinement data are read.

The refinement row R can therefore add only named successor rows over the displayed cover C. Componentwise hsame transport through H keeps each successor attached to its parent cover row, while Cont replay through T fixes the same finite inspection route after transport.

Finally the local NameCert row N names this finite tree together with the provenance row P. Since the carrier has no coordinate for selected subcovers or host refinement theorems, every exported refinement tree remains inside the displayed BHist cover packet. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCoverRefinementTree`

Theorem 1853.100 (CoveringDimension root finite-order compatibility)**.** *Let $D = (K, E, C, R, O, L, H, T, P, N)$ be an accepted* CoveringDimension$^{\uparrow}$ *carrier. If a root refinement read exports an order bound, then the order row O is compatible with the refinement tree exactly when every counted overlap is a displayed* BHist *row whose cover parent is named by R, whose transport is supplied by H, and whose replay route is supplied by T. The order read cannot inspect a host multiplicity oracle, an arbitrary open-cover family, or quotient equality of real-valued scales.*

Proof. Read the finite cover and refinement rows through Definition 1853.99. The only members that can enter an overlap count are the displayed successor rows already attached to parents in C by R.

The order row O counts those displayed overlaps. Transport by H preserves the parent assignments and containment witnesses componentwise, and replay by T reuses the same finite read rather than opening a separate multiplicity source.

The NameCert row N therefore binds finite-order compatibility to the local carrier. No row in D stores a host overlap counter, arbitrary open-cover compactness, or quotient-real equality, so the compatibility check is the finite C, R, O, H, T, N surface. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootFiniteOrderCompatibility`

Theorem 1853.101 (CoveringDimension root Real-separability window)**.** *Every root* CoveringDimension$^{\uparrow}$ *read that reuses* RealSeparability$^{\uparrow}$ *evidence must expose a finite window*

$$K, E, C, R, O, L, H, T, P, N \longmapsto \mathsf{RealSeparability}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before the cover order is consumed by a Real-facing downstream packet. The window records rational dense reads, regular-rational scale rows, dyadic tolerance comparisons, and the terminal real seal only as support for the displayed finite cover-refinement packet; it does not claim Dedekind completeness, quotient equality of real names, or a selected dense sequence.

Lean4 ✓ `BEDC.Derived.CoveringDimensionUp.CoveringDimensionRootRealSeparabilityWindow`

Proof. Start with the local cover-refinement tree and finite-order compatibility results above. They fix the finite $K, E, C, R, O, L, H, T, P, N$ packet before any Real-facing route can consume the order bound.

The Real-separability side then supplies only displayed window evidence. The RealSeparability$^{\uparrow}$ row contributes finite dense-window reads, RegSeqRat$^{\uparrow}$ packages the scale readback, DyadicRatCore$^{\uparrow}$ supplies tolerance comparisons, and Real$^{\uparrow}$ seals the resulting read without adding an ambient completeness principle.

Because the same BHist, hsame, Cont, and NameCert rows remain visible across the handoff, the downstream consumer receives a finite cover-order window. It does not receive selected dense points, quotient equality of real names, or an external separability theorem. □

Theorem 1853.102 (CoveringDimension root finite-cover refinement obligation)**.** *Every root* CoveringDimension$^{\uparrow}$ *read that asks for a finite-cover refinement must expose its* BHist *carrier rows through the compact-metric source of chapter 129, the finite-net boundary of chapter 1767, and the*

local root finite-cover siblings already routed by the hub. The resulting $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row is transported by* hsame *and replayed by* Cont *before any dimension consumer may read an order-bound witness.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootFiniteCoverRefinementObligation`

Proof. Project the root carrier rows recorded by the covering-dimension child files. A refinement request first reads the compact-metric source and its finite epsilon-net boundary, so the cover data are finite before a dimension classifier can inspect them.

Apply the existing root finite-cover refinement and stability siblings. They keep the refinement row inside the displayed BHist packet and make the local NameCert row the only public endpoint of the finite-cover refinement.

Finally transport and replay the same rows through hsame and Cont. Since the route has no hidden open-cover compactness or selected subcover coordinate, the order-bound witness remains a finite-cover NameCert obligation. □

Theorem 1853.103 (CoveringDimension root Lebesgue ledger obligation)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *read that consumes a Lebesgue-cover ledger must pass through the compact-cover ledger of chapter 1302, the compact-metric source of chapter 129, and the finite epsilon-net boundary of chapter 1767. The* BHist *rows, componentwise* hsame *transport, displayed* Cont *replay, and local* NameCert *row together certify only the finite ledger needed by covering dimension, not an ambient open-cover compactness theorem.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootLebesgueLedgerObligation`

Proof. Read the compact-cover Lebesgue ledger before the root dimension row. Its radius and cover entries are supplied by the compact-metric and finite epsilon-net siblings, so the ledger is a finite row in the same BHist packet.

The covering-dimension root siblings then consume that ledger as support for the order-bound classifier. This keeps the carrier, finite cover, Lebesgue radius, and local NameCert data on one displayed route.

Transport by hsame and replay by Cont preserve the finite ledger rows. Therefore a downstream dimension consumer receives the compact-cover Lebesgue ledger as a finite NameCert obligation, not as an unrestricted host compactness principle. □

Theorem 1853.104 (CoveringDimension root Real-separability window obligation)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *read that uses Real-separability window data must factor through the Real-separability sibling of chapter 1809, the compact-metric finite-cover route of chapter 129, and the finite epsilon-net route of chapter 1767. The read is transported by* hsame*, replayed by* Cont*, and named by the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row before any dimension or compact-cover consumer may use it.*

Proof. Start with the Real-separability cover-window siblings already attached to the covering-dimension hub. They supply the finite real-window data used to compare compact metric cover rows.

Use the compact-metric and finite epsilon-net siblings to bind those real windows to displayed finite-cover refinement data. The BHist carrier therefore contains the cover source, the separability window, and the dimension-facing classifier row together.

Finally apply componentwise hsame transport and displayed Cont replay. The local NameCert row names the resulting route, so no ambient separability theorem, selected dense sequence, or unrestricted Real completeness is exported. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootRealSeparabilityWindowObligation`

Theorem 1853.105 (CoveringDimension finite cover carrier obligation)**.** *Every root* CoveringDimension$^{\uparrow}$ *carrier exposes finite cover rows over the displayed* CompactMetric$^{\uparrow}$ *and* Met$^{\uparrow}$ *witnesses before any dimension read may inspect an order or refinement conclusion. In particular, the consumer must first see the compact finite-net row, the metric distance ledger, the finite* BHist *cover row, the refinement membership row, and the local* NameCert *row carried by the existing compactmetric route.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverCarrierObligation`

Proof. Read the hub route through Definition 1853.69 and Definition 1853.64. These place the compact metric source, finite-net row, metric distance ledger, finite cover row, refinement row, and order ledger in the displayed carrier before any dimension-facing consumer reads a bound.

The structural rows then preserve the same route. Componentwise hsame transport keeps the compact and metric witnesses attached to the finite cover rows, displayed Cont replay fixes the order of the read, and the local NameCert row names only that finite surface. □

Theorem 1853.106 (CoveringDimension Real separability scale obligation)**.** *Every scale-refinement read in a* CoveringDimension$^{\uparrow}$ *packet that touches Real-facing data must pass through the displayed* RealSeparability$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* DyadicRatCore$^{\uparrow}$ *tolerance rows before the finite cover order is consumed. The scale row may refine cover radii and rational dense windows, but it may not replace the finite cover carrier by a host separability theorem or an ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRealSeparabilityScaleObligation`

Proof. Use the Real-separability route of Definition 1853.26 together with the metric-completion cover route of Definition 1853.38. These routes display rational dense windows, dyadic tolerance comparisons, and regular-rational seal rows before cover refinement and order-bound rows are read.

The same finite carrier is preserved by the BHist and Cont rows. Since the scale rows are downstream of the metric ledger and upstream of the finite order read, they cannot introduce a separate separability oracle, selected dense subfamily, or ambient Real completeness export. □

Theorem 1853.107 (CoveringDimension compactmetric non-escape obligation)**.** *The* CoveringDimension$^{\uparrow}$ *packet does not export open-cover compactness, choice-selected refinements, or host topological dimension. Its compactmetric-facing export is limited to displayed compact finite-net rows, metric distance ledgers, finite* BHist *cover entries, refinement memberships, order rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactMetricNonescapeObligation`

Proof. Inspect the compactmetric and metric routes already named by the hub. The finite cover admission, compact-net dependency, and finite-cover order non-escape results keep compactness, refinement, and order data inside the displayed carrier rows.

No row in that carrier names arbitrary open-cover compactness, a selected refinement outside the finite BHist cover, or a host topological-dimension theorem. The structural rows transport and replay only the finite packet, so every downstream read remains inside the CoveringDimension NameCert surface. □

Theorem 1853.108 (CoveringDimension compact-net cover root route)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *cover read over an accepted carrier*

$$D = (K, E, C, R, O, L, H, T, P, N)$$

must expose the compact finite-net route

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow} \longmapsto K, E \longmapsto C, R, O, L \longmapsto H, T, P, N$$

before a dimension-bound consumer may inspect the order row. The rows $K, E, C, R, O, L, H, T, P, N$ *are the complete finite cover route:* K, E *expose compact metric and metric finite-net data,* C, R, O *expose cover membership, refinement, and order,* L *exposes the* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$ *support, and* H, T, P, N *transport, replay, source, and name the same packet.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootCompactNetCoverRoute`

Proof. Start with the compactmetric non-escape obligation, Definition 1853.107. It keeps compactness and metric information inside the displayed finite carrier rows rather than in an ambient open-cover principle.

Then read the finite cover carrier obligation, Definition 1853.105. It places K, E before C, R, O, so the order row cannot be consumed before compact finite-net and metric evidence are visible.

Finally use the Lebesgue-ledger dependency already carried by L. Since H, T, P, N only transport, replay, source, and name the packet, the resulting root route is the displayed finite chain through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, and $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$. □

Theorem 1853.109 (CoveringDimension refinement-order root route)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier* $D = (K, E, C, R, O, L, H, T, P, N)$*, the refinement-order read is available only along the displayed route*

$$K, E, C, R, \mathsf{CompactCoverLebesgueLedger}^{\uparrow}, O, L, H, T, P, N.$$

In particular, a consumer must read the finite cover row C*, the refinement membership row* R*, the order-bound row* O*, and the compact-cover ledger* L *as one packet after the* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *source rows have been displayed, and before any* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *completion-facing route may reuse the bound.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootRefinementOrderRoute`

Proof. Project the carrier coordinates. The refinement data are not a separate theorem: they are the rows C, R, O, L of the accepted packet, downstream of K, E and upstream of the structural rows.

Apply Definition 1853.108. It supplies the compact finite-net and metric prefix required before C, R, O, L can be read.

The completion-facing siblings may consume the order only after this local refinement route is fixed. Thus $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ receive a finite refinement-order packet, not an external order oracle or selected refinement. □

Theorem 1853.110 (CoveringDimension Real-separability completion root route)**.** *Any root* $\mathsf{CoveringDimension}^{\uparrow}$ *read that reaches Real-facing completion data must factor through*

$$K, E, C, R, O, L, H, T, P, N \longmapsto \mathsf{MetricCompletion}^{\uparrow}, \mathsf{RealSeparability}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

The first segment is the finite cover-refinement packet over $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{CompactCoverLebesgueLedge}$ *the second segment supplies only completion-facing metric rows, finite rational or dyadic density windows, regular-rational readback, dyadic tolerance checks, and terminal real seals.*

Proof. Begin with the scale obligation, Definition 1853.106. It places $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before a Real-facing scale read may consume the finite cover order.

Next use Definition 1853.109. That theorem fixes the local packet $K, E, C, R, O, L, H, T, P, N$ before any completion-facing consumer reuses the bound.

The $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows therefore receive only the displayed finite route. They cannot introduce arbitrary open-cover compactness, selected dense points, quotient completion equality, selected refinements, host dimension data, or ambient real completeness. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootRealSeparabilityCompletionRoute`

Theorem 1853.111 (CoveringDimension root obligation surface)**.** *Every root* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer over an accepted packet* $D = (K, E, C, R, O, L, H, T, P, N)$ *must read a finite obligation surface with rows over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *finite* BHist *covers,* $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$. *The rows* H, T, P, N *then preserve the same surface by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming; no root consumer may replace the surface by open-cover compactness, selected refinement, host simplicial realization, or ambient separability.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootObligationSurface`

Proof. Project the accepted packet. The compactmetric and metric prefix gives K, E, the finite cover and refinement rows give C, R, O, and the Lebesgue ledger gives L before any completion-facing or nerve-facing read occurs.

The completion-facing rows are only consumers of that finite prefix. They may attach $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ scale data, but they do not create a second cover carrier or a selected dense subcover.

Finally read the structural rows H, T, P, N. They transport, replay, source, and name the same finite surface, so the visible obligation set is exactly the displayed CoveringDimension packet. □

Theorem 1853.112 (CoveringDimension finite cover ledger)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet, the finite cover ledger is the ordered BEDC row*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{BHist}, \mathsf{CompactCoverLebesgueLedger}^{\uparrow}, \mathsf{RealSeparability}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, H, T, P$$

It records finite cover entries, refinement membership, order bounds, rational or dyadic scale windows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row before the dimension bound is exported.*

Proof. Start from Definition 1853.109. It fixes the finite rows K, E, C, R, O, L before the order bound can be read by any downstream consumer.

Then apply the Real-separability completion route of Definition 1853.110. The rational and dyadic scale rows are attached after the finite cover ledger, so they refine the scale of the same cover rather than replacing it.

The structural rows H, T, P, N preserve ledger identity. Thus the export is a finite cover ledger over displayed compactmetric, metric, rational, dyadic, transport, replay, provenance, and naming data. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverLedger`

Theorem 1853.113 (CoveringDimension compact-net obligation surface)**.** *Every* CoveringDimension$^{\uparrow}$ *root obligation surface that exports a dimension-bound consumer must first read* CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, CompactCoverLebesgueLedger$^{\uparrow}$, *and the local finite cover, refinement, and order rows. The exposed consumer is a finite packet over those rows together with the displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data; it is not a host compactness theorem, selected refinement principle, ambient dimension predicate, or completion witness.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionCompactNetObligationSurface`

Proof. Begin with Definition 1853.105. It fixes the local finite cover carrier and the transport, replay, provenance, and naming rows before any dimension-bound surface is exported.

Apply the compact-net cover route of Definition 1853.108. This supplies the FiniteEpsilonNet$^{\uparrow}$ row as a finite cover witness read from the same CompactMetric$^{\uparrow}$ and Met$^{\uparrow}$ source rows.

Next use Definition 1853.109. The refinement membership and order rows are attached to the compact-net cover rather than chosen outside the packet.

Finally apply Definition 1853.110. The rational, dyadic, and completion-facing scale rows consume the finite cover/refinement/order surface, so the resulting dimension-bound consumer remains a displayed BEDC packet over the listed rows. □

Theorem 1853.114 (CoveringDimension simplicial nerve handoff)**.** *If a* CoveringDimension$^{\uparrow}$ *consumer hands a finite cover bound to* SimplicialComplex$^{\uparrow}$, *the handoff must pass through the finite* BHist *cover row, the* CompactCoverLebesgueLedger$^{\uparrow}$ *refinement-order ledger, the* CompactMetric$^{\uparrow}$ *and* Met$^{\uparrow}$ *source rows, the* RealSeparability$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* DyadicRatCore$^{\uparrow}$ *scale rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The nerve row receives only this finite packet and does not receive host open-cover compactness, selected vertices, or ambient topological dimension.*

Proof. Read the root obligation surface of Definition 1853.111. It exposes the finite cover, refinement, order, scale, transport, replay, provenance, and naming rows before any downstream handoff is available.

Now restrict the downstream consumer to the nerve route. The SimplicialComplex$^{\uparrow}$ row may read cover incidences and finite intersection data from the same BHist cover ledger, but it has no coordinate for selected vertices outside the packet or for an ambient realization theorem.

Therefore the handoff is finite and constructive. It is a consumer route from the displayed CoveringDimension ledger to the displayed simplicial-complex packet, not a separate compactness or dimension principle. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionSimplicialNerveHandoff`

Definition 1853.115 (CoveringDimension finite-refinement carrier)**.** *A root finite-refinement carrier for* CoveringDimension$^{\uparrow}$ *is the accepted packet*

$$D = (K, E, C, R, O, L, S, M, A, G, H, T, P, N)$$

whose visible rows extend the root cover ledger by finite downstream handoff data. The rows K and E are the CompactMetric$^{\uparrow}$ *and* Met$^{\uparrow}$ *source rows; C is the finite cover row; R records refinement membership; O is the finite order-bound row; L is the* CompactCoverLebesgueLedger$^{\uparrow}$ *support row; S is the* SimplicialComplex$^{\uparrow}$ *nerve handoff row; M is the* MetricCompletion$^{\uparrow}$ *consumer row; A, G are the* RealSeparability$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* DyadicRatCore$^{\uparrow}$ *scale windows grouped as finite rational and dyadic data; and H, T, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert$_{\text{CoveringDimension}^{\uparrow}}$ *row.*

The carrier admits only finite cover rows, finite refinement memberships, finite order bounds, finite scale windows, and local naming rows. It has no coordinate for arbitrary open-cover compactness, selected refinements, a host dimension predicate, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteRefinementCarrier`

Theorem 1853.116 (CoveringDimension finite-refinement NameCert obligations)**.** *For every accepted finite-refinement carrier $D = (K, E, C, R, O, L, S, M, A, G, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^\uparrow}$ *root obligation surface first exposes* $\mathsf{CompactMetric}^\uparrow$, $\mathsf{Met}^\uparrow$, *finite cover rows, refinement membership,* $\mathsf{CompactCoverLebesgueLedger}^\uparrow$, $\mathsf{RealSeparability}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming before any order-bound export is available. The certificate surface is exactly this finite packet and does not import arbitrary open-cover compactness, selected refinement, host dimension data, or ambient completion.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteRefinementNameCertObligations`

Proof. Project the carrier of Definition 1853.115. The rows K, E, C, R, O, L are the finite cover, refinement, order, and Lebesgue-ledger rows already isolated by Definition 1853.111.

Next read the scale and completion-facing rows. By Definition 1853.110, $\mathsf{RealSeparability}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, and $\mathsf{MetricCompletion}^\uparrow$ consume the displayed finite prefix rather than replacing it by a separate completeness or separability theorem.

Finally use the structural rows H, T, P, N. They transport, replay, source, and name the same finite packet, so every root obligation remains inside the displayed carrier. □

Theorem 1853.117 (CoveringDimension finite-order handoff)**.** *If an accepted* $\mathsf{CoveringDimension}^\uparrow$ *finite-refinement packet exports an order bound to* $\mathsf{SimplicialComplex}^\uparrow$ *or* $\mathsf{MetricCompletion}^\uparrow$*, the export must pass through*

$$\mathsf{CompactMetric}^\uparrow, \mathsf{Met}^\uparrow, C, R, O, \mathsf{CompactCoverLebesgueLedger}^\uparrow, \mathsf{RealSeparability}^\uparrow, \mathsf{RegSeqRat}^\uparrow, \mathsf{DyadicRatCore}^\uparrow, H, T,$$

The $\mathsf{SimplicialComplex}^\uparrow$ *consumer receives only the finite nerve and incidence data determined by C, R, O, L, and the* $\mathsf{MetricCompletion}^\uparrow$ *consumer receives only the finite scale and rational or dyadic tolerance windows attached to the same packet. Neither handoff receives arbitrary open-cover compactness, selected refinement, a host dimension predicate, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteOrderHandoff`

Proof. Start with Definition 1853.116. It fixes the finite cover, refinement, order, scale, transport, replay, provenance, and naming rows before any order-bound export is visible.

For the simplicial route, apply Definition 1853.114. The nerve receives finite cover incidences and the finite order ledger, not selected vertices or a host realization theorem.

For the completion route, apply Definition 1853.110. The completion-facing packet receives the same finite order bound together with rational and dyadic scale windows. Thus both downstream consumers read the finite order handoff from the displayed carrier and cannot escape to arbitrary compactness, selected refinement, host dimension, or ambient completeness. □

Theorem 1853.118 (CoveringDimension root finite-cover Real-window exactness)**.** *A Real-facing root read of a* $\mathsf{CoveringDimension}^\uparrow$ *finite-cover packet is exact only when the compact finite-cover rows, refinement row, compact-cover Lebesgue ledger,* $\mathsf{RealSeparability}^\uparrow$ *window,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance row, and structural H, T, P, N rows are all visible in the same finite*

route. The read may export a finite cover bound for downstream consumers, but it cannot replace those rows by open-cover compactness, selected dense sequences, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootFiniteCoverRealWindowExactness`

Proof. Start with the Real-separability window obligation, Definition 1853.104. It places the Real-facing scale read behind the compactmetric source, finite cover rows, regular-rational readback, dyadic tolerance, and structural route.

Next apply the finite-cover ledger, Definition 1853.112. The ledger binds the compact finite-cover rows, refinement row, compact-cover Lebesgue ledger, and finite order data before any downstream order-bound export is available.

Finally use the Real-separability completion route, Definition 1853.110. It keeps the Real-window, RegSeqRat readback, DyadicRatCore tolerance, and completion-facing rows attached to the same finite cover packet.

The exactness claim is therefore an inventory statement over the displayed carrier. Since the route already accounts for compact finite covers, refinement, Lebesgue-ledger support, Real-separability windows, rational and dyadic readback, and H, T, P, N, no open-cover compactness theorem, selected dense sequence, or ambient Real-completeness principle can be introduced by the root read. □

Definition 1853.119 (CoveringDimension finite-cover root)**.** *A finite-cover root for* $\mathsf{CoveringDimension}^{\uparrow}$ *is the displayed carrier*

$$R = (K, C, A, M, D, Q, G, H, T, P, N)$$

where K *is a* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* C *is a finite* BHist *cover and refinement row,* A *is the* $\mathsf{RealSeparability}^{\uparrow}$ *scale-admission row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-facing row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* G *is the finite local NameCert row family for cover membership and order, and* H, T, P, N *are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row.*

The carrier admits only displayed finite covers, displayed refinements, finite scale windows, completion-facing metric rows, dyadic and regular-rational readback, and local naming data. It has no coordinate for arbitrary open-cover compactness, selected refinements, host separability, quotient completion equality, or ambient dimension.

Lean4 (definition) `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootCarrier`

Theorem 1853.120 (CoveringDimension finite-cover root admission)**.** *Every accepted finite-cover root* $R = (K, C, A, M, D, Q, G, H, T, P, N)$ *for* $\mathsf{CoveringDimension}^{\uparrow}$ *admits a root obligation surface only after the* $\mathsf{CompactMetric}^{\uparrow}$ *source, finite cover row,* $\mathsf{RealSeparability}^{\uparrow}$ *scale row,* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and local NameCert rows have all been displayed. The admitted surface is the finite BEDC packet of Definition 1853.119, not an external compactness, separability, completion, or dimension theorem.*

Proof. Project the carrier of Definition 1853.119. The rows K, C, A, M, D, Q, G are exactly the compact source, finite cover, scale, completion, dyadic, regular-rational, and local naming rows.

Apply the root obligation surface theorem Definition 1853.111. It already forces every root consumer to read the compactmetric source, finite cover ledger, Real-separability scale rows, regular-rational and dyadic readback, and local naming before a dimension bound is exported.

Finally read the finite-refinement NameCert obligations of Definition 1853.116. They repack the same finite rows through H, T, P, N, so the admission is the displayed packet and no external theorem enters the root surface. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootAdmission`

Theorem 1853.121 (CoveringDimension finite-cover root stability)**.** *Let* $R = (K, C, A, M, D, Q, G, H, T, P, N)$ *and* $R' = (K', C', A', M', D', Q', G', H', T', P', N')$ *be accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *finite-cover roots. If componentwise* hsame *transport identifies the* $\mathsf{CompactMetric}^{\uparrow}$*, finite cover,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and local NameCert rows of* R *with the corresponding rows of* R'*, then the admitted finite-cover root obligation surface is stable under that transport.*

Proof. Begin with finite-cover root admission for R. It identifies the admitted obligation surface as the displayed rows $K, C, A, M, D, Q, G, H, T, P, N$ and excludes external compactness, selected refinement, host separability, and quotient completion data.

Use the componentwise hsame hypotheses to transport the compactmetric, cover, scale, completion, dyadic, regular-rational, and local naming rows to R'. The structural rows H', T', P', N' replay and name the transported packet.

Apply Definition 1853.117. Downstream $\mathsf{SimplicialComplex}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ consumers receive only the same finite order handoff, so the root obligation surface remains stable under the displayed transport. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionFiniteCoverRootStability`

Theorem 1853.122 (CoveringDimension root metric-real window)**.** *For every accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *carrier, any root cover read that touches a metric-real window must expose finite* BHist *cover rows, the displayed* $\mathsf{Met}^{\uparrow}$ *distance ledger, the* $\mathsf{CompactMetric}^{\uparrow}$ *cover window, the* $\mathsf{RealSeparability}^{\uparrow}$ *finite density row, and the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row before a dimension-order conclusion is consumed. The route may refine a finite cover by rational or dyadic scale windows, but it may not replace the displayed carrier by open-cover compactness, an arbitrary dense subset, a selected refinement, or an ambient topological-dimension theorem.*

Proof. Start with the finite-refinement carrier of Definition 1853.115. It displays the compactmetric source, metric ledger, finite cover row, refinement membership row, order row, simplicial handoff row, and Real-facing scale windows in one packet.

Apply the root obligation surface of Definition 1853.111. That theorem forces the compactmetric, metric, finite cover, Real-separability, rational, dyadic, and simplicial rows to remain visible before a root consumer reads the order bound.

The structural rows then preserve the same finite window. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming do not add an open-cover compactness oracle, selected dense subset, selected refinement, or host dimension predicate. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootMetricRealWindow`

Theorem 1853.123 (CoveringDimension root simplicial-cover handoff)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *root handoff from a finite cover window to* $\mathsf{SimplicialComplex}^{\uparrow}$ *must pass through the finite* BHist *cover row, refinement-membership row, order-bound row,* $\mathsf{CompactMetric}^{\uparrow}$ *cover window,* $\mathsf{Met}^{\uparrow}$ *distance ledger,* $\mathsf{RealSeparability}^{\uparrow}$ *finite density row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CoveringDimension}^{\uparrow}}$ *row. The* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row receives only finite cover incidence and order data; it does not receive host realization data, selected vertices outside the carrier, arbitrary compactness, or ambient dimension information.*

Proof. First read the simplicial nerve handoff of Definition 1853.114. It identifies the finite cover incidences and order ledger as the only data exported to the simplicial consumer.

Next use Definition 1853.122. The metric and Real-separability windows are already attached to the finite cover packet before the nerve row is read, so the handoff does not create a second cover carrier.

Finally repack the handoff through the structural rows H, T, P, N named in the root packet. Since the packet contains no coordinate for selected vertices, host realization, arbitrary compactness, or ambient dimension, the simplicial consumer receives exactly the finite cover handoff stated above. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootSimplicialCoverHandoff`

Theorem 1853.124 (CoveringDimension root dimension ledger exactness)**.** *For an accepted* $\mathsf{CoveringDimension}^{\uparrow}$ *packet, the root dimension ledger is exact only when the order-bound row is read after the finite* BHist *cover row, the refinement-membership row, the* $\mathsf{CompactMetric}^{\uparrow}$ *cover window, the* $\mathsf{Met}^{\uparrow}$ *distance ledger, the* $\mathsf{RealSeparability}^{\uparrow}$ *finite density row, and the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row have all been displayed. Any downstream Real, completion, or simplicial consumer may reuse that ledger only as the same finite packet with* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming still attached.*

Proof. Project the finite-refinement NameCert obligations in Definition 1853.116. They place the cover, refinement, order, scale, transport, replay, provenance, and naming rows inside one accepted carrier.

Use the metric-real window theorem to fix the metric, compactmetric, and Real-separability prefix. Use the simplicial-cover handoff theorem to fix the nerve-facing read. Together these results leave the order-bound row as a ledger coordinate of the same finite carrier rather than an external dimension theorem.

The downstream consumers only replay that coordinate. Real-facing and completion-facing rows may attach scale windows to the finite order read, and simplicial rows may read the nerve incidence, but neither route can detach the ledger from the displayed BHist packet or replace it by ambient compactness, selected refinement, or host dimension data. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootDimensionLedgerExactness`

Theorem 1853.125 (CoveringDimension root metric-real window separation)**.** *An accepted finite cover packet for* $\mathsf{CoveringDimension}^{\uparrow}$ *cannot read a dimension ledger until the* $\mathsf{Met}^{\uparrow}$ *distance window,* $\mathsf{Real}^{\uparrow}$ *separability scale, compact-net row, and simplicial-cover handoff have all been displayed in the same finite* BHist *route. The route exports no second cover carrier, host real equality, open-cover compactness theorem, selected refinement, or quotient representative.*

Proof. First apply the root metric-real window theorem Definition 1853.122. It fixes the $\mathsf{Met}^{\uparrow}$ distance ledger, $\mathsf{CompactMetric}^{\uparrow}$ cover window, and $\mathsf{RealSeparability}^{\uparrow}$ finite density row before any dimension-order conclusion is consumed.

Next apply the root simplicial-cover handoff Definition 1853.123. The simplicial consumer receives only finite cover incidence and order data from the displayed packet, so it cannot introduce a second cover carrier.

Finally apply dimension ledger exactness Definition 1853.124. The order-bound row is read only after the finite cover, refinement-membership, metric-real, compact-net, and simplicial rows are already present. Projecting those shared finite rows leaves no coordinate for host real equality, open-cover compactness, selected refinement, quotient representatives, or a detached dimension theorem. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootMetricRealWindowSeparation`

Theorem 1853.126 (CoveringDimension root Real-separability scale budget)**.** *Every accepted* CoveringDimension$^{\uparrow}$ *finite-cover packet that consumes* RealSeparability$^{\uparrow}$ *must expose the finite density scale row before the simplicial nerve and dimension-order ledger are read. The scale budget is transported by* hsame *and replayed by* Cont *through the same root packet; it cannot be replaced by an arbitrary dense subset, an open-cover compactness theorem, host real equality, or a quotient representative.*

Proof. Apply the root metric-real window theorem Definition 1853.122. It places the RealSeparability$^{\uparrow}$ finite density row in the finite cover packet before a dimension-order conclusion is consumed.

Use the root simplicial-cover handoff Definition 1853.123. The simplicial nerve receives only finite cover incidence and order data after the displayed metric-real and separability rows are already present.

Apply root dimension ledger exactness, Definition 1853.124, together with root metric-real window separation, Definition 1853.125. These theorems keep the order-bound ledger attached to the same finite BHist route and rule out a second cover carrier, host real equality, compactness shortcut, selected refinement, or quotient representative.

The structural rows then preserve the scale budget. Componentwise hsame transport and displayed Cont replay carry the same finite density row forward, so a consumer cannot substitute an arbitrary dense subset for the RealSeparability$^{\uparrow}$ scale row. □

Lean4 ✓ `BEDC.Derived.CoveringdimensionUp.CoveringDimensionRootRealSeparabilityScaleBudget`

Chapter 1854

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ packet records the finite BEDC route by which displayed Cauchy data and an explicit modulus determine a regular observation schedule. It consumes the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy-sequence boundary of chapter 406, and the terminal Real seal of chapter 38. The packet names explicit modulus-based subsequence extraction as displayed certificate data, not as a host subsequence function, countable-choice schedule, selected limit, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 1854.1 (Cauchy subsequence extraction carrier)**.** *A* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, M, D, I, W, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the Cauchy input stream.*
2. *M is the explicit Cauchy/modulus row. It records the displayed tolerance request and the stage after which all later source windows satisfy that request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic radii, slack comparisons, and budget transfers used by the modulus and extracted windows.*
4. *I is the increasing index schedule. It is a finite unary trace whose entries are checked to be strictly later and to dominate the stage demanded by M for each displayed tolerance.*
5. *W is the extracted finite-window ledger, pairing each selected index in I with the source window in S and the dyadic tolerance entry in D.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff assembled from the extracted windows and the same dyadic tolerance ledger.*
7. *H records componentwise* hsame *transport for the source windows, modulus row, dyadic ledger, index schedule, extracted windows, regular readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite extraction request through $S, M, D, I, W, R, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, D, I, W, R, H, C, P, N$. It has no coordinate for a host subsequence function, countable-choice selector, selected limit, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchySubsequenceExtractionUp.CauchySubsequenceExtractionTasteGate single carrier alignment`

Theorem 1854.2 (Cauchy subsequence extraction obligation surface)**.** *For every accepted* CauchySubsequenceExtra *carrier*

$$E = (S, M, D, I, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source stream windows, explicit Cauchy/modulus row, dyadic tolerance ledger, increasing index schedule, extracted window ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *modulus exposure, requiring each finite extraction request to project the displayed Cauchy/modulus row M before reading a selected source window;*
3. *dyadic budget compatibility, requiring the tolerance entry in D used by M to be the same entry consumed by the extracted window ledger W and the regular readback R;*
4. *schedule monotonicity, requiring I to be a finite increasing unary trace whose chosen indices dominate the displayed modulus stage for the relevant tolerance;*
5. *extracted-window exactness, requiring W to pair only windows from S with indices admitted by I and tolerances recorded in D;*
6. *regular handoff, requiring R to be assembled from the extracted windows and tolerance ledger rather than from a detached stream representative;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no host subsequence function, countable-choice selector, selected limit, quotient equality, ambient Real-completeness datum, Lean credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 1854.1. The analytic rows are exactly the source windows S, modulus row M, dyadic ledger D, increasing index schedule I, extracted window ledger W, and regular readback row R.

The modulus and dyadic obligations are row-local. A finite extraction request must first read M, and the tolerance entry it names is the same dyadic budget recorded in D for the extracted windows and the downstream regular readback.

Schedule monotonicity is read from I. Since I is a finite unary trace inside the packet, the certificate checks increasingness and domination of the displayed modulus stage without selecting an infinite subsequence function.

Extracted-window exactness and the regular handoff are the projections to W and R. The ledger W can pair only indices admitted by I with source windows from S, and R is assembled from those displayed windows under the same tolerance data.

The remaining rows are structural. Componentwise transport, continuation replay, provenance, and the local name preserve only the displayed packet rows, so no excluded host selector, selected limit, quotient equality, ambient completeness claim, Lean credential, or bridge credential can enter the local surface. □

Theorem 1854.3 (Cauchy subsequence extraction regular window bound)**.** *Let $E = (S, M, D, I, W, R, H, C, P, N)$ be an accepted* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *carrier. For every displayed dyadic tolerance request in D, each extracted window in W whose selected index is admitted by I and dominates the corresponding stage of M is bounded by the same regular tolerance budget used by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R.*

Proof. Fix a displayed dyadic tolerance request in D. The carrier admits no tolerance row other than D, so both the modulus row and the extracted-window ledger must read the same budget.

The row M supplies the stage after which the source stream windows meet the requested Cauchy bound. The row I supplies a finite increasing trace whose selected entry dominates that stage.

Now project W. Each extracted window recorded in W is paired with an admitted index from I and with the same dyadic tolerance entry from D, so the Cauchy bound supplied by M applies to that extracted window.

Finally project R. The regular rational handoff is assembled from the extracted windows and tolerance ledger already read by W, hence it uses exactly the same regular budget and cannot tighten or replace it through a detached representative. □

Theorem 1854.4 (Cauchy subsequence extraction RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *carrier*

$$E = (S, M, D, I, W, R, H, C, P, N)$$

factors through the finite chain

$$S \longmapsto M \longmapsto D \longmapsto I \longmapsto W \longmapsto R$$

before it can inspect the structural rows H, C, P, N. Consequently the handoff exports finite stream windows, an explicit modulus row, dyadic tolerance data, an increasing index schedule, extracted windows, and a regular rational readback, but no host subsequence function, countable-choice selector, selected limit, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$-facing read of the accepted packet. The source surface available to the read is the finite stream-window row S.

The request then projects M and D. These rows display the Cauchy/modulus stage and the dyadic tolerance ledger that the extracted schedule must respect.

The increasing schedule I is consumed next. It supplies only a finite unary trace, so the read obtains selected positions for the displayed request without importing a host subsequence function or countable-choice selector.

The extracted-window ledger W pairs those selected positions with source windows and dyadic tolerances. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is assembled from that ledger, and no coordinate allows a detached stream representative or quotient equality to replace it.

Only after this chain has been read may the structural rows H, C, P, N transport, replay, source, and name the packet. They preserve the same displayed route and do not export selected limits or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1854.1, Definition 1854.2, Definition 1854.3, and Definition 1854.4 seed the finite explicit-modulus extraction route from stream windows through dyadic tolerances, increasing schedules, extracted windows, and RegSeqRat readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtraction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchySubsequenceExtractionUp.CauchySubsequenceExtractionTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close host subsequence functions, countable choice, selected limits, quotient equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exposure, dyadic budget compatibility, schedule monotonicity, extracted-window exactness, regular-window bounds, RegSeqRat handoff, and non-escape proofs for BEDC.Derived.CauchySubsequenceExtractionUp.

Constructive story: A $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ packet is generated from source StreamName windows, an explicit Cauchy/modulus row, a DyadicRatCore tolerance ledger, an increasing index schedule, extracted finite windows, a RegSeqRat handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1855

StreamMap NameCert Construction

Following Bishop and Bridges, the $\mathsf{StreamMap}^{\uparrow}$ packet records the finite pointwise route by which a certified countable name is mapped to another certified countable name before any regular rational or real-facing consumer reads the result. It sits after the finite window classifier of chapter 126, the tail-window handoff of chapter 301, the dyadic tolerance ledgers of chapter 291, and the terminal real-name seal of chapter 38. The packet names the map surface itself; it does not assert quotient stream equality, host function extensionality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1855.1 (Stream map carrier)**.** *A* $\mathsf{StreamMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, F, W, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *classifier row, exposing only finite source windows and their pointwise name cells.*
2. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *classifier row, exposing only finite target windows and their pointwise name cells.*
3. *F is the point-map row, assigning each displayed source cell in a requested window to the corresponding displayed target cell.*
4. *W records the finite-window output observations produced by F from S and consumed by T.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tail-window handoff row attached to the same source and target windows when a regular rational consumer asks for mapped readback.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic budgets, endpoint readbacks, and comparison cells used by S, T, F, W, Q.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, visible only after S, T, F, W, Q, D have fixed the finite map route.*
8. *H records componentwise* hsame *transport for source windows, target windows, point-map rows, output observations, regular readback, dyadic ledgers, Real seals, replay rows, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a stream-map request through source inspection, pointwise mapping, target-window inspection, regular rational handoff, dyadic tolerance inspection, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{StreamMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, F, W, Q, D, R, H, C, P, N$. It has no coordinate for quotient stream equality, host function extensionality, selected limits, countable choice, ambient Real completeness, or a detached semantic map between host functions.

Lean4 ✓ `BEDC.Derived.StreamMapUp.TasteGate.StreamMapTasteGate single carrier alignment`

Theorem 1855.2 (Stream map pointwise stability)**.** *Let $M = (S, T, F, W, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier. If two accepted requests inspect the same displayed finite source window through the* $\mathsf{StreamName}^{\uparrow}$ *classifier row S and transport its cells by the componentwise row H, then the point-map row F produces the same displayed target cells through T and the same output observations through W.*

Proof. Project the accepted carrier. The source row S exposes only finite windows and pointwise cells, so equality of inspected requests is read from the same finite classifier rows rather than from a quotient of streams.

The componentwise transport row H moves those displayed cells into the point-map row F. Since F has no access to an unbounded stream representative, it can act only on the cells opened by S and named by the request.

The target row T and output row W are downstream of that same point-map read. Their cells are therefore determined by the displayed finite route S, H, F, and the packet contains no host function extensionality coordinate that could introduce a second comparison. □

Lean4 ✓ `BEDC.Derived.StreamMapUp.TasteGate.StreamMapPointwiseStability`

Definition 1855.3 (Stream map pointwise carrier transport)**.** *For an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier $M = (S, T, F, W, Q, D, R, H, C, P, N)$, the pointwise carrier transport datum is the finite relation*

$$\mathsf{Transport}_M(s, t)$$

whose rows first read source $\mathsf{StreamName}^{\uparrow}$ *cell transport through H, then apply the point-map row F to the transported source cell, and finally produce the target* $\mathsf{StreamName}^{\uparrow}$ *cell transport in T before any W, Q, D, or R row is read. The relation is stored only over displayed finite windows, transported source cells, point-map outputs, target-cell rows, replay, provenance, and local naming.*

Theorem 1855.4 (Stream map pointwise carrier transport stability)**.** *The datum of Definition 1855.3 is stable under the finite-window functoriality route of Definition 1855.5. In particular, two accepted reads related by the source-cell* hsame *rows of H produce related target-cell* hsame *rows through F and T, and the transport exports no host function extensionality, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the pointwise transport relation. Its source side is exactly the source-cell component of H over the finite window opened by S.

Apply Definition 1855.2. The point-map row F consumes the transported displayed source cell and determines the target cell exposed through T and W.

Finally apply Definition 1855.5. Since every downstream consumer factors through S, F, T, W before reaching Q, D, or R, the transport remains a finite StreamMap datum and cannot expose host function extensionality or quotient-stream data. □

Theorem 1855.5 (Stream map finite-window functoriality)**.** *For an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, every finite-window consumer of the mapped stream factors through*

$$S \longmapsto F \longmapsto T \longmapsto W$$

before it can inspect the regular rational handoff Q*, the dyadic tolerance ledger* D*, or the terminal Real seal* R*.*

Proof. Begin with a finite-window request for the mapped stream. Carrier admission exposes the source classifier row S before any target-facing read is available.

The point-map row F is next. It converts the displayed source cells into displayed target cells, and no row in the carrier selects a new infinite representative or compares host functions.

The target classifier T and the output observations W then package exactly the finite target window produced by F. Only after this finite route is visible may a consumer inspect the regular rational handoff Q, the dyadic ledger D, or the Real seal R.

Thus every mapped finite-window read has the displayed factorization $S \longmapsto F \longmapsto T \longmapsto W$, and downstream consumers inherit that route rather than a quotient-stream or selected-limit principle. □

Lean4 ✓ `BEDC.Derived.StreamMapUp.TasteGate.StreamMapFiniteWindow functoriality`

Theorem 1855.6 (Stream map window functoriality)**.** *For an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, the pointwise carrier transport of Definition 1855.3 preserves finite* $\mathsf{StreamName}^{\uparrow}$ *windows through the route of Definition 1855.5. Consequently every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *handoff reads the mapped window through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *route fixed by* S, F, T, W, Q, D*, and* R*.*

Proof. Project the carrier of Definition 1855.1 and the pointwise transport datum of Definition 1855.3. The only source cells available to the map are the finite $\mathsf{StreamName}^{\uparrow}$ window cells exposed by S and transported through H.

Apply Definition 1855.5. The mapped observation factors through S, F, T, W before any regular rational, dyadic, or Real-facing row can be read.

Finally apply Definition 1855.14. The handoff to $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumes the same mapped window route and the same structural rows H, C, P, N, so it cannot import host function extensionality, quotient streams, selected limits, or ambient Real completeness. □

Theorem 1855.7 (Stream map constant-window stability)**.** *Let* $M = (S, T, F, W, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier applied to a constant certified point stream. Then the displayed finite observation window is preserved by the point-map row: the constant point row read from* S *is routed through* F *to the same target-window shape in* T *and* W*, and any downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer must first read the same* hsame*,* Cont*,* Pkg*, and local* NameCert *support rows.*

Proof. Project the $\mathsf{StreamMap}^{\uparrow}$ carrier. The source classifier row S exposes the constant point stream only through a displayed finite window and its point cells.

Use the constant point row together with the finite-window row W. Since every source cell in the inspected window is the certified constant point, the point-map row F produces a target row with the same displayed window shape before Q, D, or R is available.

Componentwise transport is then read through H. The target cells, output observations, regular handoff, dyadic ledger, and Real seal boundary are moved only along the displayed source and target windows.

Finally the replay and provenance rows C, P, N preserve that route for downstream consumers. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer therefore receives the mapped constant window only after the finite StreamMap support has been displayed, and cannot replace it by quotient stream equality, host function extensionality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1855.8 (Stream map composition-window route)**.** *Let two accepted* $\mathsf{StreamMap}^{\uparrow}$ *carriers be composable by identifying the first target window with the second source window. The composed mapped stream can be exposed only through the first source* $\mathsf{StreamName}^{\uparrow}$ *row, the first point-map row, the intermediate target/source window, the second point-map row, and the final target window before any* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row is consumed.*

Proof. Project the two accepted carriers. The first carrier exposes its mapped finite-window read through its source row, point-map row, target row, and output row before any regular rational, dyadic, or Real-facing row is available.

Apply Definition 1855.5 to the first carrier. The intermediate target window is therefore a displayed $\mathsf{StreamName}^{\uparrow}$ window, not a selected stream representative or a host-function value.

Apply the same theorem to the second carrier with that intermediate window serving as its source window. The second point-map row consumes only the displayed cells of the intermediate window and exports only the final displayed target window.

Thus the composed route is the concatenation of the two finite-window routes. It reaches the regular rational handoff, dyadic ledger, or Real seal only after the first source window, first point-map row, intermediate window, second point-map row, and final target window have all been displayed. □

Lean4 ✓ `BEDC.Derived.StreamMapUp.TasteGate.StreamMapCompositionWindowRoute`

Theorem 1855.9 (Stream map pointwise composition)**.** *Let* M_1 *and* M_2 *be accepted* $\mathsf{StreamMap}^{\uparrow}$ *carriers whose first target* $\mathsf{StreamName}^{\uparrow}$ *window is the second source* $\mathsf{StreamName}^{\uparrow}$ *window. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of the pointwise composite must read the first source window, the first point-map row, the intermediate target window, the second point-map row, the final target window, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before any terminal Real seal is available.*

Proof. Apply Definition 1855.5 to M_1. The first source window is opened through the first source $\mathsf{StreamName}^{\uparrow}$ row, the first point-map row consumes only those displayed cells, and the intermediate target window is exposed as a finite $\mathsf{StreamName}^{\uparrow}$ row.

Apply Definition 1855.5 again to M_2. The second point-map row reads the same intermediate window as its source and exposes only the final displayed target window.

Now use Definition 1855.8. It concatenates the two finite-window routes, so the composite can reach a downstream consumer only after both point-map rows and both source/target window boundaries are visible.

Finally apply Definition 1855.11. The dyadic tolerance ledger and $\mathsf{RegSeqRat}^{\uparrow}$ handoff remain downstream of the displayed windows, and the terminal Real seal cannot import host function extensionality, quotient streams, selected limits, countable choice, or ambient Real completeness. □

Theorem 1855.10 (Stream map tail-window determinacy)**.** *For an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$, *any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the mapped stream must first read the same source* $\mathsf{StreamName}^{\uparrow}$ *row* S, *point-map row* F, *target* $\mathsf{StreamName}^{\uparrow}$ *row* T, *output-window row* W, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *fixed by the finite-window route.*

If two consumer reads agree on those rows, then they agree on the exposed tail-window handoff Q before either terminal Real seal R is read.

Proof. Project the carrier of Definition 1855.1. The rows S, F, T, W are the finite map route, while Q and D are downstream rows attached to that same displayed route.

Apply Definition 1855.5. Every consumer read of the mapped stream factors through S, F, T, W before it can inspect Q, D, or R, so two reads agreeing on the source, point-map, target, and output-window rows have the same finite route.

Now read the tolerance and tail rows. The ledger D records the dyadic budgets for the same finite output observations, and the handoff row Q is the regular rational tail read attached to those observations.

Therefore agreement on S, F, T, W, D fixes the classifier-facing handoff Q before the terminal Real seal is available. Since no quotient-stream coordinate or selected-limit coordinate appears in the carrier, there is no second tail-window handoff for the consumer to select. □

Theorem 1855.11 (Stream map tail-ledger nonescape)**.** *For an accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, any consumer that reads the regular rational handoff Q, the dyadic tolerance ledger D, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal R must first read the same finite route S, F, T, W. The structural rows H, C, P, N preserve only that displayed route and cannot introduce quotient stream equality, host function extensionality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 1855.5. Every finite-window consumer reaches S, F, T, W before the downstream rows Q, D, or R are available, so a tail-facing read cannot bypass the displayed source window, point-map row, target window, or output-window row.

Apply Definition 1855.10. Once two tail-facing reads agree on S, F, T, W, D, the exposed regular rational handoff Q is fixed before the terminal Real seal is read.

It remains to inspect the structural rows. The rows H, C, P, N transport, replay, cite provenance, and name the same finite route; none stores a quotient-stream comparison, host-function extensionality proof, selected-limit witness, countable-choice selector, or ambient Real-completeness principle. Thus every tail-ledger consumer remains inside the displayed StreamMap ledger. □

Theorem 1855.12 (Stream map NameCert obligation surface)**.** *For every accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier*

$$M = (S, T, F, W, Q, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for source and target StreamName classifier rows, the point-map row, output observations, RegSeqRat handoff, DyadicRatCore ledger, Real seal boundary, transport, replay, provenance, and local naming;*
2. *source-window exactness, requiring every mapped request to project S before F, T, W, Q, D, or R can be read;*
3. *pointwise stability, requiring F to consume only displayed source cells and to export only displayed target cells;*
4. *finite-window functoriality, requiring every output observation to factor through S, F, T, W before any regular rational, dyadic, or Real-facing consumer can inspect it;*
5. *tail-window handoff, requiring Q to reuse the same finite windows rather than selecting a second stream representative;*

6. *ledger non-escape, requiring D, R, H, C, P, N to export no quotient stream equality, host function extensionality, selected limit, countable choice, ambient Real completeness, or detached host map.*

No local stream-map obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.StreamMapUp.TasteGate.StreamMapNameCertObligationSurface`

Proof. Carrier admission is projection from Definition 1855.1. The accepted packet displays exactly the source and target StreamName rows, the point-map row, finite output observations, the regular rational handoff, the dyadic ledger, the Real seal boundary, and the structural rows.

Source-window exactness and pointwise stability use the same finite inspection route. A request must first open S, then F consumes only the cells displayed by that window and exports only the corresponding target cells.

Finite-window functoriality follows from Definition 1855.5. The mapped observation reaches T and W before any Q, D, or R read, so downstream regular rational, dyadic, and Real-facing consumers inherit the same finite map route.

The tail-window handoff is the projection to Q. Since Q is stored after S, T, F, W, it records the regular rational readback for the displayed windows and has no coordinate for choosing another stream representative.

Finally, ledger non-escape is exhausted by D, R, H, C, P, N. These rows record tolerances, seal boundaries, componentwise transport, displayed replay, provenance, and local naming for the same finite route, and none stores quotient stream equality, host function extensionality, selected limits, countable choice, ambient Real completeness, or a detached host map. □

Theorem 1855.13 (Stream map obligation-closure package)**.** *For every accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, the carrier row of Definition 1855.1, the pointwise stability of Definition 1855.2, the carrier transport datum of Definition 1855.3, the carrier transport stability of Definition 1855.4, the finite-window functoriality of Definition 1855.5, the composition-window route of Definition 1855.8, the tail-window determinacy of Definition 1855.10, the tail-ledger nonescape of Definition 1855.11, and the NameCert obligation surface of Definition 1855.12 jointly form the obligation-closed finite* $\mathsf{StreamMap}^{\uparrow}$ *packet over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

Proof. Project the carrier row first. It displays exactly the source and target $\mathsf{StreamName}^{\uparrow}$ rows, the point-map row, finite output windows, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the $\mathsf{Real}^{\uparrow}$ seal, and the structural rows H, C, P, N.

Apply the pointwise and transport clauses next. The rows cited in Definition 1855.2, Definition 1855.3, and Definition 1855.4 keep the map action inside displayed finite source and target cells.

Now use the window and tail clauses. The route of Definition 1855.5, Definition 1855.8, Definition 1855.10, and Definition 1855.11 forces every downstream regular-rational, dyadic, or Real-facing consumer to pass through the same finite StreamMap window route before any tail or seal row is visible.

Finally apply Definition 1855.12. Its listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$ surface, so the paper closure claim here is an obligation package over the displayed finite rows only. It does not assert Lean verification, quotient stream equality, host function extensionality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1855.14 (Stream map RegSeqRat and Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a mapped stream accepted by a* $\mathsf{StreamMap}^{\uparrow}$ *carrier must first read source-window exactness, pointwise map stability, finite-window functoriality, tail-window determinacy, and tail-ledger nonescape. The handoff is therefore a finite pointwise stream route from source windows to regular rational readback and the terminal real seal, not a quotient stream equality, selected limit, host function extensionality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with pointwise stability from Definition 1855.2. It keeps each target cell attached to the displayed source cell and point-map row rather than to a detached host function.

Apply Definition 1855.5. The mapped observation reaches the target window through the finite route S, F, T, W before any downstream Q, D, or R row can be consumed.

Now apply Definition 1855.10. A regular rational or real-facing read that agrees on the source window, point-map row, target window, output window, and dyadic ledger must expose the same regular handoff before the terminal real seal is read.

Finally apply Definition 1855.11. The tail-facing consumer cannot bypass S, F, T, W, D and cannot import quotient stream equality, selected limits, host function extensionality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1855.15 (Stream map scoped closure package)**.** *Every accepted* $\mathsf{StreamMap}^{\uparrow}$ *packet routes pointwise map data through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows before any* $\mathsf{Real}^{\uparrow}$*-facing consumer reads it. The scoped route is the finite package assembled by the obligation-closure theorem and the RegSeqRat/Real handoff theorem; it exports no quotient stream equality, host function extensionality, selected limit, countable choice, or ambient real-completeness datum.*

Proof. Start with the obligation package of Definition 1855.13. It exposes the carrier row, source-window exactness, pointwise stability, carrier transport, finite-window functoriality, composition-window routing, tail-window determinacy, tail-ledger nonescape, and the local NameCert surface over the displayed $\mathsf{StreamMap}^{\uparrow}$ rows.

Next apply Definition 1855.14. That theorem places every regular-rational or Real-facing read behind the same source window, point-map row, target window, output observation, dyadic ledger, regular rational handoff, and terminal Real seal.

The structural rows H, C, P, N then preserve the scoped route as componentwise transport, displayed replay, provenance, and local naming. Since the carrier has no coordinate for quotient stream equality, host function extensionality, selected limits, countable choice, or ambient Real completeness, the scoped closure package is exactly the displayed finite route. □

Theorem 1855.16 (Stream map scoped obligation route)**.** *For every accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, the carrier definition, pointwise stability, carrier transport, finite-window functoriality, composition-window route, tail-window determinacy, tail-ledger nonescape, NameCert obligation surface, and* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff jointly bind the exact rows*

$$S, T, F, W, Q, D, R, H, C, P, N$$

over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. Every consumer read of the scoped route must pass through those rows before reaching a regular-rational or Real-facing handoff.*

The route exports no quotient stream equality, host function extensionality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 1855.1. It displays $S, T, F, W, Q, D, R, H, C, P, N$ as the only source, target, point-map, output-window, regular handoff, dyadic ledger, Real seal, transport, replay, provenance, and local-name rows.

Apply Definition 1855.2, Definition 1855.3, and Definition 1855.4. These rows keep the pointwise map action and its transport inside displayed source and target windows.

Next use Definition 1855.5, Definition 1855.8, Definition 1855.10, and Definition 1855.11. They force composition, tail, dyadic, and Real-facing reads through S, T, F, W, Q, D, R before any consumer can inspect a downstream result.

Finally apply Definition 1855.12, Definition 1855.13, and Definition 1855.14. The local NameCert surface and handoff theorem exhaust the structural rows H, C, P, N, so the scoped obligation route has no coordinate for quotient stream equality, host function extensionality, selected limits, countable choice, or ambient Real completeness. □

Theorem 1855.17 (Stream map scope kernel route)**.** *For every accepted* $\mathsf{StreamMap}^{\uparrow}$ *carrier* $M = (S, T, F, W, Q, D, R, H, C, P, N)$*, every consumer read that reaches* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *factors through the displayed rows* $S, T, F, W, Q, D, R, H, C, P, N$ *and the kernel primitives* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The route exports no quotient stream equality, host function extensionality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 1855.2. It keeps the source cell, target cell, and point-map row attached to the accepted carrier rather than to an external function.

Use Definition 1855.3. The transport row is componentwise hsame over the displayed carrier cells, so a transported read still names $S, T, F, W, Q, D, R, H, C, P, N$.

Next apply Definition 1855.5. It forces each mapped observation through the finite source and target windows before any regular-rational, dyadic, or Real-facing row is public.

Finally use Definition 1855.14 and Definition 1855.12. The handoff accounts for the Q, D, R rows, while the NameCert surface accounts for H, C, P, N as BHist-based transport, displayed replay, provenance, and local naming. Since no carrier coordinate stores quotient stream equality, host function extensionality, selected limits, countable choice, or ambient completeness, every such consumer read factors through the stated kernel route. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$, obligation)

Scope closed: Definition 1855.1, Definition 1855.13, Definition 1855.14, and Definition 1855.16 give the obligation witness set for the finite L10 pointwise map route through StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host function extensionality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, pointwise stability, finite-window functoriality, tail-window handoff, and ledger non-escape proofs for BEDC.Derived.StreamMapUp.stream_map_namecert_obliga

Constructive story: A $\mathsf{StreamMap}^{\uparrow}$ packet is generated from source and target $\mathsf{StreamName}^{\uparrow}$ classifier rows, a point-map row, finite-window output observations, a $\mathsf{RegSeqRat}^{\uparrow}$ tail-window handoff, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1856

A Concrete Naming Certificate for $\mathsf{BishopIntervalNestLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ packet records the finite nested-interval limit route through chapter 320, chapter 328, chapter 126, chapter 301, chapter 38, and chapter 1639. The packet gives L10 Real and Completion work a displayed induction-style handoff from nested interval data to a terminal Real seal, while remaining below any claim of general Heine–Borel compactness, selected intersection points, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1856.1 (Bishop interval-nest limit carrier)**.** *A* $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested-interval induction row. It records the inspected prefix, the containment step from each displayed interval to its predecessor, and the nonempty interval surface inherited from* $\mathsf{NestedInterval}^{\uparrow}$*.*
2. *D is the* $\mathsf{NestedDyadicInterval}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger used to read the inspected interval endpoints at the requested dyadic scales.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It exposes only the finite interval-prefix observations and endpoint windows that the induction row permits.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled endpoint observations and the shrinking-width ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, D, S, R have been displayed.*
6. *M is the Bishop regular Cauchy completion handoff row. It records that the same I, D, S, R, E route is the one consumed by* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for the interval induction row, endpoint ledger, stream schedule, regular rational readback, Real seal, completion handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a nested-interval request through I, D, S, R, E, M.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalNestLimit}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedDyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNestLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, M, H, C, P, N$. The carrier contains no general Heine–Borel compactness row, selected intersection point, quotient stream equality, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, host real equality, or bridge credential.*

Lean4 ✓ `BEDC.Derived.BishopIntervalNestLimitUp.BishopIntervalNestLimitTasteGate single carrier alignment`

Theorem 1856.2 (Bishop interval-nest limit NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ *carrier*

$$L = (I, D, S, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalNestLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed nested-interval induction row, dyadic endpoint ledger, stream schedule, regular rational readback, Real seal, Bishop regular Cauchy completion handoff, transport, replay, provenance, and local naming row;*
2. *induction-step exposure, requiring every interval read to be one of the displayed prefix rows of I and every successor read to pass through the displayed containment step;*
3. *dyadic endpoint exactness, requiring D to supply the endpoint and shrinking-width comparisons used by the scheduled windows;*
4. *stream and regular readback discipline, requiring S and R to reuse the same displayed prefix and endpoint ledger rather than selecting a second nested chain or regular sequence;*
5. *Real-seal handoff, requiring E to be reached only after the nested, dyadic, stream, and regular rational rows have been projected;*
6. *completion-interface non-escape, requiring M, H, C, P, N to transport, replay, source, and name no general Heine–Borel compactness row, selected intersection point, quotient stream equality, countable-choice selector, ambient Real completeness datum, host real equality, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1856.1. The accepted packet displays exactly the interval induction row, dyadic endpoint ledger, stream schedule, regular rational readback, Real seal, completion handoff, and the structural rows.

The induction-step exposure clause is forced by I. A consumer can inspect only the finite prefix already present in the packet, and each displayed successor interval is read together with its containment step into the preceding interval.

Dyadic endpoint exactness is then forced by D. The endpoint and shrinking-width comparisons are the only numerical rows available to the schedule, so the packet cannot introduce an external point selector or compactness witness.

The rows S and R are downstream of the same displayed prefix and endpoint ledger. They schedule finite observations and repack them as regular rational readback without replacing the nested chain by a second representative.

Finally, E, M, H, C, P, N close the route. The Real seal and completion handoff are reached only after I, D, S, R, while the structural rows transport, replay, source, and name the same finite data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 1856.3 (Bishop interval-nest limit convergence route)**.** *Let $L = (I, D, S, R, E, M, H, C, P, N)$ be an accepted* $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ *carrier. Every Real-facing or completion-facing read exported by L factors through the induction route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto E \longmapsto M$$

before it can inspect the transport, replay, provenance, or local naming rows. The route is a finite nested-interval handoff into the $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with a Real-facing or completion-facing read of L. The carrier gives no source row before the displayed nested-interval induction row I.

The read then projects the dyadic endpoint ledger D. This fixes the endpoint observations and shrinking-width comparisons that the later schedule may use.

Next the read passes through S and R. The schedule exposes only finite interval-prefix windows, and the regular rational row repacks those same observations as the Cauchy-facing readback.

Only after this route has been displayed can the read reach the Real seal E and the Bishop regular Cauchy completion handoff M. The remaining rows H, C, P, N preserve and name that route, so every exported read factors through the displayed induction chain. □

Theorem 1856.4 (Bishop interval-nest limit nonescape)**.** *For an accepted* $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ *carrier $L = (I, D, S, R, E, M, H, C, P, N)$, every exported comparison, replay, provenance read, completion handoff, or local name is exhausted by the displayed rows $I, D, S, R, E, M, H, C, P, N$. In particular, the packet exports no general Heine–Borel compactness theorem, selected intersection point, quotient stream equality, countable-choice selector, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Project the accepted packet. Its classifier compares carriers only through the displayed interval induction row, endpoint ledger, schedule, regular readback, Real seal, completion handoff, and structural rows.

The comparison-sensitive rows are M, H, C, P, N. The completion handoff records the same finite route into the Bishop regular Cauchy completion surface, transport is componentwise, replay follows the displayed request path, provenance names the same source chapters, and the local NameCert row names only this packet.

Since no accepted coordinate stores a compactness theorem, an intersection selector, quotient stream equality, choice data, host real equality, ambient Real completeness, or a bridge credential, no exported comparison or replay can use such a row. The packet therefore remains an induction-style nested-interval handoff rather than a completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNestLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopIntervalNestLimit}^{\uparrow}$ packet is generated from a finite nested-interval induction row, a nested dyadic endpoint ledger, StreamName finite windows, RegSeqRat readback, a terminal RealUp seal, a BishopRegularCauchyCompletion handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalNestLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1856.1, Definition 1856.2, Definition 1856.3, and Definition 1856.4 seed the finite nested-interval induction route from displayed interval prefixes through RegSeqRat readback to the terminal Real seal and Bishop regular Cauchy completion handoff.

Formal verification: FormalStatus(NameCert$_{\text{BishopIntervalNestLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, selected intersection points, quotient stream equality, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, induction-step exposure, dyadic endpoint exactness, stream and regular readback discipline, Real-seal handoff, completion-interface non-escape, and convergence-route proofs for BEDC.Derived.BishopIntervalNestLimitUp.

Chapter 1857

A Concrete Naming Certificate for $\mathsf{CozeroSet}^{\uparrow}$

Following Kelley and Bishop–Bridges, cozero sets are read as the positive-apartness locus of a continuous real-valued map, with BEDC sources chapter 38, chapter 130, chapter 88, and chapter 111. The $\mathsf{CozeroSet}^{\uparrow}$ packet names the finite route from a displayed continuous $\mathsf{Real}^{\uparrow}$-valued map to an open and set-like membership surface. It does not assert zero-set completeness, classical complements, excluded middle, host $\mathsf{Real}^{\uparrow}$ equality, or an ambient semantic preimage theorem.

Definition 1857.1 (Cozero set carrier)**.** *A* $\mathsf{CozeroSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (F, R, A, O, S, H, K, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for a displayed map into* $\mathsf{Real}^{\uparrow}$*, exposing only the source observation, target real-name, and graph cells used by the cozero read.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *observation row attached to the target cell of F.*
3. *A is the positive-apartness witness row certifying that the displayed target observation is apart from zero through the same real-name surface as R.*
4. *O is the induced* $\mathsf{Topology}^{\uparrow}$ *open-row, available only after F, R, A have fixed the positive witness.*
5. *S is the* $\mathsf{Set}^{\uparrow}$ *membership row exported by the open-row O and read only through the same displayed source cell.*
6. *H records componentwise* hsame *transport for the continuous-map row, Real observation row, positive-apartness row, open-row, set-membership row, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a cozero membership request through F, R, A, O, S, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CozeroSet}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Set}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CozeroSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, A, O, S, H, K, P, N$. It has no coordinate for the classical complement of a zero set, excluded middle, host real equality, quotient real equality, or a detached semantic preimage theorem.

Theorem 1857.2 (Cozero set NameCert obligations)**.** *For every accepted* $\mathsf{CozeroSet}^{\uparrow}$ *carrier*

$$C = (F, R, A, O, S, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CozeroSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the continuous-map row into* $\mathsf{Real}^{\uparrow}$*, the target Real observation row, the positive-apartness witness row, the induced topological open-row, the set-membership row, transport, replay, provenance, and local naming;*
2. *positive witness exactness, requiring every accepted membership read to use the same target real observation* R *and positive-apartness row* A *opened by* F*;*
3. *continuous-preimage stability, requiring componentwise* hsame *and displayed* Cont *transport of the continuous map and positive witness to preserve the cozero membership surface;*
4. *open-set handoff, requiring* O *to be read only after* F, R, A *have displayed the positive witness;*
5. *set-membership handoff, requiring* S *to export only the membership surface induced by* O *on the same displayed source cell;*
6. *non-escape, requiring* H, K, P, N *to export no host real equality, quotient real equality, classical complement, excluded middle, zero-set completeness, or detached semantic preimage theorem.*

No local cozero-set obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CozeroSetUp.TasteGate.CozeroSetUpTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1857.1. The accepted tuple displays exactly the continuous-map row, target Real observation row, positive-apartness witness, open-row, set-membership row, and structural transport, replay, provenance, and naming rows.

Positive witness exactness is forced by the order of the carrier. A membership read first opens F, then reads the target observation R, and only then obtains the positive-apartness witness A for the same displayed real-name surface.

The open-set and set-membership handoffs are the projections to O and S. Since both rows are downstream of F, R, A, they cannot introduce an independent complement, zero-set equality test, or host real comparison.

Transport and replay stability are exhausted by H and K. These rows move the displayed map and positive witness componentwise and replay the same finite route, so the cozero membership surface is preserved without adding excluded middle, host equality, or a detached semantic preimage principle.

Finally, non-escape follows from the absence of any further classifier coordinate. The provenance and local NameCert rows name only this finite route, and no row stores zero-set completeness, classical complements, excluded middle, or host $\mathsf{Real}^{\uparrow}$ equality. □

Theorem 1857.3 (Cozero set positive witness route)**.** *Every membership read exported by an accepted* $\mathsf{CozeroSet}^{\uparrow}$ *packet factors through*

$$F \longmapsto R \longmapsto A$$

before it can inspect the topological open-row O *or the set-membership row* S*.*

Proof. Begin with a membership request for the displayed cozero surface. Carrier admission exposes no set row before the continuous-map row F.

The request then reads the target Real observation row R. This fixes the real-name observation used by the positive test, and the packet contains no host equality coordinate from which another target value could be substituted.

The positive-apartness witness row A is therefore the gate to membership. Only after F, R, A have been displayed may the request inspect the induced open-row O and the set row S.

Thus every membership read has the displayed factorization $F \longmapsto R \longmapsto A$ before any topological or set-like export. □

Theorem 1857.4 (Cozero set continuous-preimage stability)**.** *Let $C = (F, R, A, O, S, H, K, P, N)$ be an accepted* $\mathsf{CozeroSet}^{\uparrow}$ *carrier. Componentwise* hsame *transport and displayed* Cont *replay of the continuous-map row F, Real observation row R, and positive-apartness witness row A preserve the induced cozero membership surface exported through O and S.*

Proof. Project the accepted carrier. The rows F, R, A determine the positive witness route for the displayed source cell and target real observation.

The transport row H moves these three rows componentwise. It does not compare host real values or select a classical complement, so it preserves exactly the displayed positive-apartness witness.

The replay row K then replays the same finite route through the open-row O and set-membership row S. Since O and S are downstream of F, R, A, the transported packet exports the same cozero membership surface and no additional zero-set or complement principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CozeroSet}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CozeroSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1857.1, Definition 1857.2, Definition 1857.3, and Definition 1857.4 seed the finite L10 route from continuous $\mathsf{Real}^{\uparrow}$-valued maps through positive-apartness witnesses into topological and set-like membership surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CozeroSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close zero-set completeness, classical complements, excluded middle, host $\mathsf{Real}^{\uparrow}$ equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive witness exactness, continuous-preimage stability, open-set handoff, set-membership handoff, and non-escape proofs for BEDC.Derived.CozeroSetUp.

Constructive story: A $\mathsf{CozeroSet}^{\uparrow}$ packet is generated from a $\mathsf{ContinuousMap}^{\uparrow}$ row into $\mathsf{Real}^{\uparrow}$, a Real observation row, a positive-apartness witness row, an induced $\mathsf{Topology}^{\uparrow}$ open-row, a $\mathsf{Set}^{\uparrow}$ membership row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1858

A Concrete Naming Certificate for $\mathsf{ZeroSet}^{\uparrow}$

Following Kelley and Bishop–Bridges, zero sets are read as displayed zero-loci of continuous real-valued maps, with the constructive cozero companion in chapter 1857 and BEDC sources chapter 38, chapter 130, chapter 88, and chapter 111. The $\mathsf{ZeroSet}^{\uparrow}$ packet names the finite route from a displayed continuous $\mathsf{Real}^{\uparrow}$-valued map through an exact zero witness into a closed topological and set-like membership surface. It does not identify this surface with the classical complement of a cozero set, and it does not add excluded middle, host $\mathsf{Real}^{\uparrow}$ equality, quotient equality, or an ambient semantic preimage theorem.

Definition 1858.1 (Zero set carrier)**.** *A* $\mathsf{ZeroSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (F, R, E, T, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for a displayed map into* $\mathsf{Real}^{\uparrow}$*, exposing only the source observation, target real-name, and graph cells used by the zero-locus read.*
2. *R is the displayed* $\mathsf{Real}^{\uparrow}$ *observation row attached to the target cell of F.*
3. *E is the zero-witness row certifying that the displayed target observation is zero through the same real-name surface as R.*
4. *T is the induced* $\mathsf{Topology}^{\uparrow}$ *closed-locus row, available only after F, R, E have fixed the zero witness.*
5. *S is the* $\mathsf{Set}^{\uparrow}$ *membership row exported by the closed-locus row T and read only through the same displayed source cell.*
6. *H records componentwise* hsame *transport for the continuous-map row, Real observation row, zero-witness row, closed-locus row, set-membership row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a zero-set membership request through F, R, E, T, S, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ZeroSet}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Set}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ZeroSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, E, T, S, H, C, P, N$. It has no coordinate for the classical complement of $\mathsf{CozeroSet}^{\uparrow}$*, excluded middle, host Real equality, quotient equality, or a detached semantic preimage theorem.*

Lean4 ✓ `BEDC.Derived.ZeroSetUp.ZeroSetTasteGate single carrier alignment`

Theorem 1858.2 (Zero set NameCert obligations)**.** *For every accepted* $\mathsf{ZeroSet}^{\uparrow}$ *carrier*

$$Z = (F, R, E, T, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ZeroSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the continuous-map row into* $\mathsf{Real}^{\uparrow}$*, the displayed target Real observation row, the zero-witness row, the induced topological closed-locus row, the set-membership row, transport, replay, provenance, and local naming;*
2. *zero-witness exactness, requiring every accepted membership read to use the same target real observation R and exact zero row E opened by F;*
3. *continuous-preimage stability, requiring componentwise* hsame *and displayed* Cont *transport of the continuous map and zero witness to preserve the zero-locus membership surface;*
4. *topology and set handoff, requiring T to be read only after F, R, E have displayed the zero witness and requiring S to export only the membership surface induced by T on the same displayed source cell;*
5. *CozeroSet boundary separation, requiring the packet to share the continuous-map and Real observation vocabulary of* $\mathsf{CozeroSet}^{\uparrow}$ *while refusing to identify the zero row E with a negated positive-apartness witness;*
6. *non-escape, requiring H, C, P, N to export no classical complement of* $\mathsf{CozeroSet}^{\uparrow}$*, excluded middle, host Real equality, quotient equality, or detached semantic preimage theorem.*

No local zero-set obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1858.1. The accepted tuple displays exactly the continuous-map row, target Real observation row, zero-witness row, closed-locus row, set-membership row, and structural transport, replay, provenance, and naming rows.

Zero-witness exactness is forced by the order of the carrier. A membership read first opens F, then reads the target observation R, and only then obtains the exact zero witness E for the same displayed real-name surface.

The topology and set handoffs are the projections to T and S. Since both rows are downstream of F, R, E, they cannot introduce an independent complement, a cozero-set negation rule, or a host real comparison.

Continuous-preimage stability is exhausted by H and C. These rows move the displayed map and zero witness componentwise and replay the same finite route, so the zero-locus membership surface is preserved without adding excluded middle, host equality, quotient equality, or a detached semantic preimage principle.

CozeroSet boundary separation follows from the absence of any row converting E into the negation of a positive-apartness witness. The packet may cite the sibling vocabulary of chapter 1857, but it exports only the displayed zero route named by $F, R, E, T, S, H, C, P, N$.

Finally, non-escape follows from the absence of any further classifier coordinate. The provenance and local NameCert rows name only this finite route, and no row stores classical complements, excluded middle, host $\mathsf{Real}^{\uparrow}$ equality, quotient equality, or ambient preimage semantics. □

Theorem 1858.3 (Zero set positive-boundary separation)**.** *Every membership read exported by an accepted* $\mathsf{ZeroSet}^{\uparrow}$ *packet factors through*

$$F \longmapsto R \longmapsto E$$

before it can inspect the closed-locus row T *or the set-membership row* S*. The same read cannot be replaced by a* $\mathsf{CozeroSet}^{\uparrow}$ *positive-apartness route or by the classical complement of such a route.*

Proof. Begin with a membership request for the displayed zero-locus surface. Carrier admission exposes no set row before the continuous-map row F.

The request then reads the target Real observation row R. This fixes the real-name observation used by the zero test, and the packet contains no host equality coordinate from which another target value could be substituted.

The zero-witness row E is therefore the gate to membership. Only after F, R, E have been displayed may the request inspect the induced closed-locus row T and the set row S.

The carrier has no row that turns the absence of a $\mathsf{CozeroSet}^{\uparrow}$ positive witness into a zero witness. Thus the displayed factorization through $F \longmapsto R \longmapsto E$ is a zero-locus route, not a complement route. □

Theorem 1858.4 (Zero set nonclassical boundary)**.** *An accepted* $\mathsf{ZeroSet}^{\uparrow}$ *packet supplies no classical complement of* $\mathsf{CozeroSet}^{\uparrow}$*, no excluded middle, no host* $\mathsf{Real}^{\uparrow}$ *equality, no quotient equality, and no ambient semantic preimage. Its exported surface is only the displayed finite route*

$$F, R, E, T, S, H, C, P, N.$$

Proof. Project the accepted carrier. The listed rows are the only rows available to a downstream read.

The rows F, R, E determine the displayed zero witness, while T and S expose only the closed-locus and set-membership surfaces induced by that witness. None of these rows compares host real values or reasons by excluded middle.

The transport and replay rows H and C preserve the displayed route componentwise. They do not add quotient equality, a semantic preimage theorem, or a complement operation against $\mathsf{CozeroSet}^{\uparrow}$.

The provenance and local naming rows P, N name the finite package and no further classifier. Hence every exported read remains inside $F, R, E, T, S, H, C, P, N$ and cannot escape to the classical boundary listed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{ZeroSet}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ZeroSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1858.1, Definition 1858.2, Definition 1858.3, and Definition 1858.4 seed the finite route from continuous $\mathsf{Real}^{\uparrow}$-valued maps through displayed zero witnesses into closed topological and set-like membership surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ZeroSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical complements of $\mathsf{CozeroSet}^{\uparrow}$, excluded middle, host $\mathsf{Real}^{\uparrow}$ equality, quotient equality, ambient semantic preimage theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-witness exactness, continuous-preimage stability, topology and set handoff, CozeroSet boundary separation, and non-escape proofs for BEDC.Derived.ZeroSetUp.

Constructive story: A $\mathsf{ZeroSet}^{\uparrow}$ packet is generated from a $\mathsf{ContinuousMap}^{\uparrow}$ row into $\mathsf{Real}^{\uparrow}$, a Real observation row, an exact zero-witness row, an induced $\mathsf{Topology}^{\uparrow}$ closed-locus row, a $\mathsf{Set}^{\uparrow}$ membership row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1859

A Concrete Naming Certificate for $\mathsf{LocatedUniformContinuity}^{\uparrow}$

Following Bishop and Bridges on constructive continuity with explicit moduli, the $\mathsf{LocatedUniformContinuity}^{\uparrow}$ packet records the finite L10 route by which a located source row and a located-apartness witness feed a uniform-continuity modulus before any Real or completion consumer reads the target seal. It sits over the modulus discipline of chapter 1631, the Cauchy-modulus row of chapter 292, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the completion handoff of chapter 1639, and the located-source surface of chapter 1541. The packet names the constructive located-uniform-continuity handoff; it is not open-cover compactness, host function equality, selected limits, quotient stream equality, countable choice, or ambient Real completeness.

Definition 1859.1 (Located uniform continuity carrier)**.** *A* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, M, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the located source row, exposing the finite* $\mathsf{LocatedSet}^{\uparrow}$ *cells and source-location requests that the continuity route may inspect.*
2. *A is the located-apartness witness row, recording the positive separation and located decision cells used to keep source inspection constructive.*
3. *M is the* $\mathsf{ModulusContinuity}^{\uparrow}$ *row, turning each requested target tolerance into a finite source-window budget and citing the Cauchy-modulus discipline when completion-facing reads ask for tail control.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row consumed by the located source, apartness, and modulus rows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording target tolerances, source budgets, dyadic endpoints, comparison cells, and positive-gap readbacks.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *seal handoff, visible only after X, A, M, S, D, R have fixed the displayed route.*

8. *H records componentwise* hsame *transport for the located source, apartness witness, modulus row, windows, dyadic ledger, regular readback, Real and completion handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a located-uniform-continuity request through source location, apartness inspection, modulus conversion, finite-window readback, dyadic ledger inspection, regular readback, and completion handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformContinuity}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, M, S, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, host function equality, selected limits, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedUniformContinuityUp.LocatedUniformContinuityTasteGate single carrier alignment`

Theorem 1859.2 (Located uniform continuity NameCert obligations)**.** *For every accepted* LocatedUniformContinui *carrier*

$$L = (X, A, M, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedUniformContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located source, located-apartness witness, modulus-continuity row, StreamName finite windows, DyadicRatCore tolerance ledger, RegSeqRat readback, Real and completion seal handoff, transport, replay, provenance, and local naming;*

2. *located-source stability, requiring X and A to expose only displayed source cells and positive-separation witnesses before any target tolerance can be read;*

3. *modulus-window adequacy, requiring M, S, D to turn every target tolerance into a finite source-window budget recorded in the dyadic ledger;*

4. *readback and handoff coherence, requiring R and E to consume the same windows and budgets before any Real or completion-facing consumer receives a seal;*

5. *transport, replay, and non-escape, requiring H, C, P, N to preserve only the displayed rows and to export no open-cover compactness, host function equality, selected limit, quotient stream equality, countable choice, or ambient Real completeness.*

No local located-uniform-continuity obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1859.1. The accepted packet displays exactly the located source row, apartness witness, modulus row, finite windows, dyadic ledger, regular readback, Real and completion handoff, and the structural rows.

Located-source stability is forced by the order of the displayed route. The rows X and A are read before any tolerance request reaches M, so source inspection remains located and positive-separation evidence cannot be replaced by a host equality test.

The modulus-window obligation is the finite handoff through M, S, D. A target tolerance first reads the modulus row, then opens only the source windows in S and the dyadic cells in D that were displayed for that request.

Readback and handoff coherence are projections to R and E. The regular rational row and the Real or completion seal consume the same windows and budgets, so neither can select a limit, compare quotient streams, or import ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the excluded principles have no coordinate in the carrier, they cannot become local NameCert obligations. □

Theorem 1859.3 (Located uniform continuity completion handoff)**.** *Every Real or completion consumer of an accepted* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *carrier factors through the displayed route*

$$X, A, M, S, D \longmapsto R \longmapsto E$$

before componentwise transport, replay, provenance, or local naming can be inspected. The consumer receives a located source, located-apartness evidence, an explicit modulus, finite windows, a dyadic tolerance ledger, regular rational readback, and a Real or Bishop-regular-Cauchy completion seal, but no selected limit, quotient stream equality, countable choice, host function equality, open-cover compactness, or ambient Real completeness.

Proof. Start with an accepted carrier $L = (X, A, M, S, D, R, E, H, C, P, N)$. Projection exposes the located source row X and the apartness witness row A before any completion-facing read is available.

The modulus stage is next. The row M turns the target request into source-window data, and those windows are exactly the finite cells recorded by S and budgeted by the dyadic ledger D.

The regular readback row R consumes the same finite data. It does not choose a stream representative or a limiting value; it records the regular rational readback allowed by the displayed source windows and tolerance ledger.

The handoff row E is terminal for Real and completion consumers. It receives X, A, M, S, D, R as finite BEDC rows and has no coordinate for selected limits, quotient equality, choice, host function equality, open-cover compactness, or ambient completeness.

The structural rows H, C, P, N preserve this route by componentwise transport, displayed replay, provenance, and local naming. Therefore every Real or completion consumer factors through the displayed handoff before it can inspect the terminal certificate surface. □

Theorem 1859.4 (Located uniform continuity non-escape)**.** *No accepted* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *packet exports open-cover compactness, host function equality, selected limits, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exported comparison, replay, or consumer handoff is exhausted by the rows* $X, A, M, S, D, R, E, H, C, P, N$.

Proof. Inspect the classifier of Definition 1859.1. Accepted packets are compared only componentwise through the displayed located source, apartness, modulus, window, ledger, readback, seal, transport, replay, provenance, and naming rows.

The analytic rows X, A, M, S, D, R, E provide located source inspection, positive separation, tolerance conversion, finite windows, dyadic budgets, regular readback, and terminal Real or completion handoff. None stores an open cover, a host function graph equality, a chosen limit, a quotient-stream equality proof, a choice schedule, or an ambient completeness principle.

The structural rows H, C, P, N cannot add new analytic coordinates. They transport, replay, source, and name only the accepted tuple, so any exported comparison or consumer handoff must remain inside the same finite row list.

Thus the excluded principles have no entry point in the carrier, classifier, transport, replay, provenance, or local NameCert surface, and the packet is non-escaping in precisely the stated sense. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1859.1, Definition 1859.2, Definition 1859.3, and Definition 1859.4 seed the finite Real and Completion route for located-source uniform continuity through located source, apartness, modulus, finite-window, dyadic-ledger, regular-readback, Real and completion handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host function equality, selected limits, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires the later BEDC.Derived.LocatedUniformContinuityUp carrier, modulus-window, completion-handoff, transport, and nonescape obligations.

Constructive story: A $\mathsf{LocatedUniformContinuity}^{\uparrow}$ packet is generated from a $\mathsf{LocatedSet}^{\uparrow}$ source row, a located-apartness witness row, a $\mathsf{ModulusContinuity}^{\uparrow}$ row citing $\mathsf{CauchyModulus}^{\uparrow}$ tail control, $\mathsf{StreamName}^{\uparrow}$ finite windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ and $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ handoff seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1860

A Concrete Naming Certificate for $\mathsf{SierpinskiSpace}^{\uparrow}$

Following Sierpinski and Scott on observable opens, the $\mathsf{SierpinskiSpace}^{\uparrow}$ packet records the two-observation classifier that turns open-set membership into finite BEDC data. It sits after the finite truth rows of chapter 32, the open-set discipline of chapter 88, the finite countable-name windows of chapter 126, and the Real-facing uniform-structure boundary of chapter 1624. The same observable-open truth row is the input surface for later topology consumers such as chapter 3338, **??**, and compact-uniform cover routes, where an open-membership read must remain a displayed Sierpinski classifier row rather than a host proposition. The packet names observable truth of opens as a classifier surface; it does not import excluded middle, selected limits, quotient streams, host propositions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1860.1 (Sierpinski space carrier)**.** *A* $\mathsf{SierpinskiSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, O, W, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *truth row, exposing the two accepted observable truth marks and their classifier transport.*
2. *O is the* $\mathsf{Topology}^{\uparrow}$ *open-set row, recording the open under inspection as a displayed observation row rather than as a host proposition.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row, recording the finite windows through which a countable name reports whether the open has been observed.*
4. *H records componentwise* hsame *transport for truth rows, open rows, finite-observation windows, replay rows, provenance, and the local naming row.*
5. *C records displayed* Cont *replay from an observable-open request through B, O, W, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, O, W, H, C, P, N. It contains no coordinate for host propositions, excluded middle, selected limits, quotient stream equality, countable choice, or a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SierpinskiSpaceUp.SierpinskiSpaceTasteGate single carrier alignment`

Theorem 1860.2 (Observable open-set classifier)**.** *Let $S = (B, O, W, H, C, P, N)$ be an accepted* $\mathsf{SierpinskiSpace}^{\uparrow}$ *carrier. Any two accepted observable-open requests are compared only through the displayed open-truth rows B, O, W and the specialization-order readback they determine. The classifier has no access to host propositions, proof irrelevance for ambient opens, excluded middle, or a hidden truth-value evaluator outside the packet.*

Lean4 ✓ `BEDC.Derived.SierpinskiSpaceUp.SierpinskiSpaceCarrier open set classifier`

Proof. Project the accepted carrier. The truth row B supplies the only accepted observable truth marks, and the open row O records the open under inspection as displayed topology data.

The finite-observation row W is the only route by which a countable name can report an observation of that open. Thus a comparison may read whether one displayed observation is below another in the specialization order generated by the observable truth rows, but it cannot inspect an external proposition asserting open membership.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since none of them stores excluded middle, proof irrelevance, a hidden truth evaluator, selected limits, quotient streams, or ambient Real completeness, the classifier is exhausted by observable open-truth readback. □

Theorem 1860.3 (Sierpinski space NameCert obligation surface)**.** *For every accepted* $\mathsf{SierpinskiSpace}^{\uparrow}$ *carrier*

$$S = (B, O, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Bool truth row, Topology open-set row, StreamName finite-observation row, transport, replay, provenance, and local naming row;*
2. *open-truth classifier exactness, requiring every observable-open comparison to read B, O, W before any structural row can be consumed;*
3. *specialization-order stability, requiring* hsame *transport to preserve the observable order read from the two truth marks and the displayed open row;*
4. *finite-observation stability, requiring every StreamName window in W to report only the displayed observable truth row attached to O;*
5. *ledger non-escape, requiring H, C, P, N to export no excluded middle, selected limits, quotient streams, countable choice, host proposition equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local observable-open obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1860.1. The accepted packet displays exactly the Bool truth row, the Topology open-set row, finite StreamName observations, and the structural transport, replay, provenance, and local naming rows.

Open-truth classifier exactness follows from the order of the displayed rows. A request can compare observable opens only after reading the truth row B, the open row O, and the finite observation row W.

Specialization-order stability is the componentwise transport clause in H. Transport preserves the displayed truth marks and the open row, so the specialization readback determined by them is preserved as well.

Finite-observation stability is the projection to W. Each window reports the observable truth row attached to O, and no row selects a second stream representative or a detached open-membership proposition.

Finally, ledger non-escape is exhausted by H, C, P, N. These rows transport, replay, source, and name the displayed observable-open route only, and none stores excluded middle, selected limits, quotient streams, countable choice, host proposition equality, ambient Real completeness, Lean verification, or a bridge credential. □

Theorem 1860.4 (Sierpinski space consumer handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{TZeroReflection}^{\uparrow}$*,* $\mathsf{UrysohnLemma}^{\uparrow}$*, or observable-open-filter consumer of an accepted* $\mathsf{SierpinskiSpace}^{\uparrow}$ *packet must read the* $\mathsf{Bool}^{\uparrow}$ *truth row* B*, the* $\mathsf{Topology}^{\uparrow}$ *open row* O*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, and the structural rows* H, C, P, N *before treating open membership as available. The consumer receives only displayed observable-open truth and its replay route; it receives no host proposition for membership, no excluded middle over opens, no proof-irrelevant open equality, no selected limit, and no quotient stream representative.*

Proof. Project the carrier of Definition 1860.1. The Bool truth row, the Topology open row, and the finite StreamName windows are the only rows that expose observable open membership.

Apply Definition 1860.2. It forces any membership comparison to read B, O, W rather than a detached host proposition or hidden evaluator.

Apply Definition 1860.3. The structural rows H, C, P, N then transport, replay, source, and name that same displayed route before any downstream topology, T0, Urysohn, or observable-open-filter consumer may use it.

Since the carrier has no coordinate for excluded middle, proof-irrelevant open equality, selected limits, quotient stream representatives, or ambient Real completeness, none of those objects can be supplied by the handoff. □

Theorem 1860.5 (Sierpinski observable-open induction)**.** *Let* $S = (B, O, W, H, C, P, N)$ *be an accepted* $\mathsf{SierpinskiSpace}^{\uparrow}$ *carrier, and let* ω *be a finite observable-open request history addressed to the displayed* $\mathsf{StreamName}^{\uparrow}$ *window list* W*. The request is exhausted by induction on that finite window list. The base case reads no positive observation beyond the* $\mathsf{Bool}^{\uparrow}$ *truth row* B *and the* $\mathsf{Topology}^{\uparrow}$ *open row* O*. In the step case, consuming one displayed window from* W *preserves the specialization-order readback through the transport row* H*, then replays the same observable-open route through* C, P, N*. Thus every finite observable-open request over* S *remains inside* B, O, W, H, C, P, N *before any topology-facing consumer can read open membership.*

Proof. Read the observable-open classifier from Definition 1860.2. It restricts every membership comparison to the displayed truth row B, the open row O, the StreamName window row W, and the specialization-order readback determined by those rows.

Induct on the displayed finite window list inside W. In the empty case, the request history contains no positive window report, so the only available observable information is the pair B, O together with the structural rows that name the route.

For a successor window, consume the first displayed window and apply the obligation surface of Definition 1860.3. Finite-observation stability keeps the window attached to the same open row O, and specialization-order stability transports the readback componentwise through H.

After the window is consumed, apply the consumer handoff of Definition 1860.4. The remaining request is replayed through C and sourced and named by P, N, so the induction step produces the same finite route for the tail of the window list. No host open-membership proposition, hidden

evaluator, selected limit, quotient stream, or ambient Real-completeness coordinate is introduced. □

Theorem 1860.6 (Sierpinski observable-open filter link)**.** *Every* $\mathsf{ObservableOpenFilter}^{\uparrow}$ *read through chapter 6399 that consumes an accepted* $\mathsf{SierpinskiSpace}^{\uparrow}$ *carrier*

$$S = (B, O, W, H, C, P, N)$$

must first expose the Sierpinski observable truth row B*, the* $\mathsf{Topology}^{\uparrow}$ *open row* O*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*, and the structural rows* H, C, P, N*. The downstream filter therefore receives observable-open truth as displayed Sierpinski data, not as a host proposition, selected limit, quotient stream, excluded-middle witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the accepted carrier. The rows B, O, W are the only rows that expose observable-open truth, open-set data, and finite StreamName windows.

Apply Definition 1860.5. It exhausts every finite observable-open request by induction on W and preserves the same route through H, C, P, N.

The ObservableOpenFilter consumer named in chapter 6399 can therefore read only that displayed Sierpinski route. Since no carrier row stores host proposition membership, selected limits, quotient stream representatives, excluded middle, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects is exported to the downstream filter. □

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1860.1, Definition 1860.2, Definition 1860.3, and Definition 1860.6 seed the finite L10 observable-open classifier route through Bool truth rows, Topology open rows, StreamName finite observations, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close excluded middle, choice, selected limits, quotient streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-truth classifier exactness, specialization-order stability, finite-observation stability, and ledger non-escape proofs for BEDC.Derived.SierpinskiSpaceUp.

Constructive story: A $\mathsf{SierpinskiSpace}^{\uparrow}$ packet is generated from a Bool truth row, a Topology open-set row, StreamName finite-observation windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1861

A Concrete Naming Certificate for $\mathsf{CousinLemma}^{\uparrow}$

Following Bishop-style treatments of Cousin's lemma for gauge partitions, the $\mathsf{CousinLemma}^{\uparrow}$ packet names the finite route from a located closed interval and a displayed positive gauge to a tagged interval subdivision. It sits between the finite cover surface of chapter 1596, the positive-radius handoff of chapter 452, and the bisection discipline of chapter 1711. Its Real-facing rows are read through the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter records the traditional finite-subdivision certificate used by later integration and compact-interval cover arguments; it is not general open-cover compactness, a choice-selected partition, or host real completeness.

Definition 1861.1 (Cousin lemma carrier)**.** *A* $\mathsf{CousinLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, G, T, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located closed interval row, including the named endpoints, endpoint order, and the interval boundary that later bisection rows may inspect.*
2. *G is the gauge row, assigning to each displayed tag request a positive local radius read through finite rational and Real-facing witnesses.*
3. *T is the finite tagged-subinterval row. Each entry records a tag, a left endpoint, a right endpoint, adjacency to the neighbouring row, containment in I, and the local gauge admission for that tag.*
4. *M is the mesh ledger recording the endpoint comparisons and mesh bounds used to certify that consecutive tagged subintervals cover exactly the displayed interval row.*
5. *W is the located bisection ledger. It records the finite bisection induction tree that refines any row whose current mesh is not yet admitted by the corresponding gauge radius.*
6. *R is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposed for the endpoint, tag, radius, and mesh observations in I, G, T, M, W.*
7. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite window schedule has been displayed.*

8. *H records componentwise* hsame *transport for the interval, gauge, tagged-subdivision, mesh, bisection, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a gauge-partition request through $I, G, W, T, M, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, T, M, W, R, E, H, C, P, N$. It has no coordinate for a general open-cover compactness theorem, a choice-selected partition, a host real completeness theorem, quotient stream equality, or an external real interval object.

Lean4 ✓ `BEDC.Derived.CousinLemmaUp.TasteGate.CousinLemmaTasteGate single carrier alignment`

Theorem 1861.2 (Cousin lemma NameCert obligations)**.** *For every accepted* $\mathsf{CousinLemma}^{\uparrow}$ *carrier*

$$K = (I, G, T, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CousinLemma}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, gauge, tagged-subdivision, mesh, bisection, window, readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *gauge positivity and locality, requiring every tag row of T to read its radius from G and to prove the corresponding tagged subinterval is contained in that local radius window;*
3. *subdivision adjacency and coverage, requiring the ordered entries of T to start at the left endpoint of I, end at the right endpoint of I, and share consecutive endpoints without gaps or overlaps;*
4. *mesh and bisection exactness, requiring M and W to be the only ledgers by which failed local gauge tests are refined into smaller interval rows;*
5. *StreamName, RegSeqRat, and Real-seal discipline, requiring R and E to read only the finite endpoint, tag, radius, and mesh windows already exposed by I, G, T, M, W;*
6. *structural non-escape, requiring H, C, P, N to transport, replay, source, and name no general open-cover compactness theorem, choice-selected partition, host real completeness theorem, quotient stream equality, or external real interval object.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1861.1. The accepted packet displays the located interval, gauge, tagged subdivision, mesh ledger, bisection ledger, finite window schedule, readback and Real seal, and the structural rows.

Gauge positivity and locality are read from G and T. Each tag row carries its own local radius test, and the classifier has no branch that can replace that test by an arbitrary open cover or by a selected point from the ambient host interval.

Subdivision adjacency and coverage are read from T and M. The first and last endpoint comparisons bind the subdivision to I, while consecutive endpoint equality is recorded in the finite mesh ledger.

The refinement surface is exhausted by W. Whenever a current interval row does not satisfy its local gauge test, the next row is produced by the displayed bisection ledger and then re-tested through the same gauge row.

Finally R, E, H, C, P, N preserve this route. The StreamName windows and RegSeqRat readback use only the displayed finite observations, the Real seal is terminal for those observations, and the structural rows transport and name the same packet. Since no excluded coordinate occurs in the carrier, the listed rows are exactly the NameCert obligations. □

Theorem 1861.3 (Cousin tagged partition existence)**.** *Let $K = (I, G, T, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CousinLemma}^{\uparrow}$ *carrier. The tagged-subinterval row T is produced by finite bisection induction over the displayed located interval row I: the base row accepts an interval whose chosen tag radius from G covers the current subinterval, and the induction row bisects a failed interval by W, applies the same local gauge test to both children, and concatenates the accepted child rows with the adjacency ledger in M. Thus T is a finite tagged partition of I whose every tagged subinterval is admitted by its own gauge row.*

Proof. Proceed by finite induction on the bisection tree displayed in W. The base case is an interval row already admitted by its local gauge radius in G. Its single tagged subinterval is entered into T, and M records that its endpoints are the endpoints of the current interval.

For the induction step, take a displayed interval row whose local gauge test has not yet been admitted. The bisection ledger W supplies the midpoint row and the two child interval rows, both still contained in the parent row of I.

Apply the induction hypothesis separately to the left and right child rows. The left accepted list covers the left child, the right accepted list covers the right child, and every row in both lists carries its own tag and gauge-radius admission from G.

Concatenate the two finite lists. The mesh ledger M identifies the shared midpoint endpoint and preserves the outer endpoints inherited from the parent interval. This gives a finite tagged partition of the parent row.

Applying this construction to the root interval row I yields the displayed row T. The proof uses only the finite bisection induction ledger, not open-cover compactness or countable choice. □

Theorem 1861.4 (Cousin finite gauge-partition handoff)**.** *Let $K = (I, G, T, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CousinLemma}^{\uparrow}$ *carrier. The tagged-subinterval row T produced by finite bisection induction is exactly the finite gauge-partition handoff consumed by* $\mathsf{FiniteGaugeCover}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, *and* $\mathsf{HenstockKurzweil}^{\uparrow}$*: every such consumer receives the ordered finite rows*

$$I, G, W, T, M \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

with no additional open-cover compactness, selected partition, host real completeness, quotient stream equality, or external interval equality coordinate.

Proof. Project the accepted carrier to its displayed rows $I, G, T, M, W, R, E, H, C, P, N$. By Definition 1861.3, the finite bisection ledger W produces T over the interval row I, and every entry of T carries its local gauge admission from G with endpoint and adjacency checks recorded in M.

Those are precisely the finite gauge-partition rows inspected by $\mathsf{FiniteGaugeCover}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, and $\mathsf{HenstockKurzweil}^{\uparrow}$. The downstream read cannot start from a separate cover theorem or a selected partition, because the classifier of Definition 1861.1 exposes only the rows $I, G, T, M, W, R, E, H, C, P, N$.

Finally apply Definition 1861.5. It sends the same finite subdivision through R, E, and then H, C, P, N, so the consumer-facing packet is exactly the displayed gauge-partition handoff and not any stronger ambient compactness or completeness principle. □

Theorem 1861.5 (Cousin finite Lebesgue handoff)**.** *Every* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$*, Riemann-style, Darboux-style, compact-interval-cover, or Real-completion consumer of an accepted* $\mathsf{CousinLemma}^{\uparrow}$ *carrier reads the packet through the ordered route*

$$I, G, W, T, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The exported handoff is a finite gauge-tagged subdivision certificate over displayed interval rows, StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal. It does not export general open-cover compactness, choice-selected partitions, host real completeness, quotient stream equality, or external interval equality.*

Proof. Start with a downstream consumer read. Carrier admission exposes the located interval row I and the gauge row G before any subdivision data can be inspected.

The bisection ledger W and tagged-subinterval row T then provide the finite subdivision route. The mesh ledger M records the endpoint comparisons and adjacency facts consumed by finite Lebesgue-number, Riemann-style, Darboux-style, and compact-interval-cover arguments.

The Real-facing part of the read passes next through R and E. These rows expose only finite StreamName windows, RegSeqRat readback, and the terminal Real seal attached to the interval, tag, radius, and mesh observations already displayed.

The remaining rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for open-cover compactness, a choice-selected partition, host real completeness, quotient stream equality, or external interval equality, no downstream consumer can read those principles from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{CousinLemma}^{\uparrow}$ packet is generated from a located closed interval row, a positive gauge row, finite tagged-subinterval rows, mesh and bisection ledgers, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CousinLemma}^{\uparrow}}$, seed)

Scope closed: Definition 1861.1, Definition 1861.2, Definition 1861.3, Definition 1861.4, and Definition 1861.5 seed the finite gauge-tagged subdivision route from a located closed interval through bisection induction, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CousinLemma}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CousinLemmaUp`

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, choice-selected partitions, host real completeness, quotient stream equality, external interval equality, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge locality, subdivision adjacency, finite bisection induction, finite gauge-partition handoff, finite Lebesgue handoff, Real-seal non-escape, and downstream consumer boundary proofs for BEDC.Derived.CousinLemmaUp.

Chapter 1862

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$ packet records the finite BEDC route by which a regular-Cauchy subsequence read reaches a terminal real-name seal. It consumes the regular-Cauchy subsequence surface of chapter 381, the subsequence-limit consumer boundary of chapter 1467, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the handoff from extracted regular subsequence data to a Real-facing seal as displayed certificate data, not as arbitrary subsequence choice, quotient stream equality, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1862.1 (Regular-Cauchy subsequence limit carrier)**.** *A* $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *row, exposing the selected finite windows and the regular-Cauchy discipline inherited from the subsequence extraction route.*
2. *L is the existing* $\mathsf{SubsequenceLimit}^{\uparrow}$ *consumer boundary, recording the finite limit-facing route through which selected subsequence data is read.*
3. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window ledger, containing only the finite windows inspected by the regular-Cauchy subsequence and the limit consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from those selected windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the regular-Cauchy row, the selected windows, the regular rational readback, and the limit boundary.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after S, L, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the subsequence row, limit boundary, selected windows, regular rational readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite Real-facing read through $S, L, W, R, D, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{SubsequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, W, R, D, E, H, C, P, N$. It has no coordinate for arbitrary subsequence choice, quotient stream equality, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceLimitUp.RegularCauchySubsequenceLimitTasteGate`
`single carrier alignment`

Theorem 1862.2 (Regular-Cauchy subsequence limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$ *carrier*

$$T = (S, L, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular-Cauchy subsequence row, subsequence-limit consumer boundary, selected stream windows, regular rational readback, dyadic tolerance ledger, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *selected-window alignment, requiring every window read by L and R to be one of the finite windows named by W and admitted by the regular-Cauchy subsequence row S;*
3. *tolerance preservation, requiring the dyadic ledger D to be the same ledger consumed by S, W, R, and the limit-facing row L;*
4. *Real-seal handoff, requiring the seal row E to be exposed only after the regular-Cauchy subsequence row, limit boundary, selected windows, regular rational readback, and tolerance ledger have been displayed;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name no arbitrary subsequence choice, quotient stream equality, host sequence equality, ambient Real-completeness datum, Lean credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 1862.1. The visible analytic rows are exactly the regular-Cauchy subsequence row S, the subsequence-limit boundary L, selected windows W, regular rational readback R, dyadic ledger D, and terminal seal E.

Selected-window alignment is read from the triple (S, L, W). The row S admits the finite subsequence windows, while L can consume only the windows displayed by the same ledger W.

Tolerance preservation is the projection to D. The regular-Cauchy row, selected-window row, regular rational readback, and limit boundary all name the same dyadic tolerance ledger, so no second budget can alter the limit read.

The Real-seal handoff follows the row order S, L, W, R, D, E. A Real-facing consumer reaches E only after the selected subsequence data, finite windows, regular rational readback, and dyadic tolerance ledger are already visible.

The remaining rows are structural. Componentwise transport, continuation replay, provenance, and local naming preserve only the displayed packet rows, so the excluded host choices, equality principles, completeness claim, Lean credential, and bridge credential cannot enter the local surface. □

Theorem 1862.3 (Regular-Cauchy subsequence limit Real seal boundary)**.** *Let* $T = (S, L, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read from* T *factors through*

$$S, L, W, R, D, E, H, C, P, N.$$

Consequently the terminal seal is reached only through the regular-Cauchy subsequence row, the existing subsequence-limit boundary, selected stream windows, regular rational readback, and the shared dyadic tolerance ledger.

Proof. Begin with a $\mathsf{Real}^{\uparrow}$-facing read of the accepted packet. The carrier places the terminal seal E downstream of S, L, W, R, D, so the read must first inspect the regular-Cauchy subsequence row and the existing subsequence-limit consumer boundary.

The selected windows and readback rows are then forced by Definition 1862.2. Every window consumed by the limit boundary is recorded in W, and every regular rational observation in R uses the same selected-window ledger.

The dyadic tolerance ledger is shared by all analytic rows. Since D is the only displayed tolerance coordinate, the Real-facing read cannot switch to a detached modulus, quotient representative, or host sequence equality before reaching the seal.

Finally H, C, P, N transport, replay, source, and name the same finite route. Thus every Real-facing read factors through the listed rows and exports no arbitrary subsequence choice, quotient stream equality, host sequence equality, ambient completeness theorem, Lean credential, or bridge credential. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}$ packet is generated from a RegularCauchySubsequence row, an existing SubsequenceLimit consumer boundary, selected StreamName windows, RegSeqRat readback, a shared DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1862.1, Definition 1862.2, and Definition 1862.3 seed the finite route from regular-Cauchy subsequence data through the subsequence-limit boundary to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary subsequence choice, quotient stream equality, host sequence equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-window alignment, tolerance preservation, Real-seal handoff, and non-escape proofs for BEDC.Derived.RegularCauchySubsequenceLimitUp.

Chapter 1863

A Concrete Naming Certificate for $\mathsf{DyadicPartition}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the standard dyadic partition use in Riemann and Darboux integration, the $\mathsf{DyadicPartition}^{\uparrow}$ packet records a finite ordered family of dyadic interval cells over chapter 306. It sits between the interval tree route of **??**, the bisection route of chapter 1489, the finite-cover handoff of chapter 1596, and the step-function consumer of chapter 428. The packet gives the L10 Real and Completion route a finite partition interface without importing open-cover compactness, quotient-style real equality, selected mesh limits, or ambient completeness.

Definition 1863.1 (Dyadic partition carrier rows)**.** *A* $\mathsf{DyadicPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (B, I, A, R, F, H, C, G, N)$$

with the following displayed rows:

1. *B is the boundary row. It records the lower and upper dyadic endpoints of the inspected interval, read through the* $\mathsf{DyadicInterval}^{\uparrow}$ *surface.*
2. *I is the finite ordered list of* $\mathsf{DyadicInterval}^{\uparrow}$ *cell rows. Each cell exposes its two dyadic endpoints and its local endpoint ledger.*
3. *A is the adjacency and coverage row. It records that consecutive cells share the displayed endpoint and that the first and last cells meet the boundary row B.*
4. *R is the common-refinement row. It names the finite interval rows produced when two accepted partition carriers are compared through a displayed refinement.*
5. *F is the finite-cover handoff row. It exports the same cells as a finite interval-cover certificate for downstream compactness-facing consumers.*
6. *H records componentwise* hsame *transport for the boundary, cells, adjacency, common-refinement, finite-cover, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a partition request through B, I, A, R, F, H, G, N.*
8. *G is the* Pkg *provenance over* $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicIntervalTree}^{\uparrow}$, $\mathsf{IntervalHalving}^{\uparrow}$, $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for open-cover compactness, quotient-style real equality, selected limiting mesh, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicPartitionUp.TasteGate.DyadicPartitionTasteGate single carrier alignment`

Definition 1863.2 (Dyadic partition classifier rows)**.** *Two accepted* $\mathsf{DyadicPartition}^{\uparrow}$ *carriers are classified together exactly when their boundary rows agree through the displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint classifier, their cell lists admit the finite common-refinement row* R*, every refined cell transports to a parent cell through the adjacency ledger* A*, and the finite-cover handoff row* F *exports the same displayed cells. The comparison is finite and ordered: every read is mediated by* $B, I, A, R, F, H, C, G, N$*.*

Theorem 1863.3 (Dyadic partition NameCert obligation rows)**.** *For every accepted* $\mathsf{DyadicPartition}^{\uparrow}$ *carrier* $P = (B, I, A, R, F, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicPartition}^{\uparrow}}$ *surface consists of exactly the following obligation rows:*

1. *carrier admission for the displayed boundary, cell list, adjacency, common-refinement, finite-cover, transport, replay, provenance, and local naming rows;*
2. *boundary and cell exposure, requiring every endpoint read to pass through* B *or one of the finitely many cells of* I*;*
3. *adjacency and coverage exactness, requiring* A *to account for shared endpoints and boundary contact;*
4. *common-refinement stability, requiring comparisons of two partition carriers to use the finite row* R *and no hidden partition representative;*
5. *finite-cover handoff, requiring* F *to export exactly the displayed cells to the interval finite-cover route before any step-function or completion-facing consumer reads the packet.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DyadicPartitionUp.TasteGate.DyadicPartitionNamecertObligationRows`

Proof. Project the carrier rows of Definition 1863.1. The accepted packet displays exactly the boundary row, finite ordered cell list, adjacency ledger, common-refinement row, finite-cover handoff, and the structural rows.

Boundary and cell exposure are forced by B and I. A downstream read can inspect an endpoint only through the boundary interval or one of the listed dyadic interval cells.

Adjacency and coverage exactness are forced by A. The shared endpoint and boundary-contact reads are finite rows, so the packet does not select an external subdivision or an ambient real equality witness.

Common-refinement stability is forced by R. When two carriers are compared, the classifier first displays the finite refinement and then transports each refined cell back to its parent cell through A.

Finally, F, H, C, G, N export, transport, replay, source, and name the same rows. Since the carrier has no additional coordinate, the five listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicPartition}^{\uparrow}}$**.**

Constructive story: A DyadicPartition$^{\uparrow}$ packet is generated from a boundary dyadic interval row, a finite ordered list of dyadic interval cells, an adjacency and coverage ledger, a common-refinement row, a finite-cover handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicPartition}\uparrow}$, seed)

Scope closed: Definition 1863.1, Definition 1863.2, and Definition 1863.3 seed the finite dyadic partition interface from displayed dyadic interval cells to common refinement, finite-cover handoff, and step-function consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicPartition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient-style real equality, selected limiting meshes, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary and cell exposure, adjacency and coverage exactness, common-refinement stability, finite-cover handoff, provenance exhaustion, and local naming for BEDC.Derived.DyadicPartitionUp.

Chapter 1864

A Concrete Naming Certificate for $\mathsf{DyadicLipschitzModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ packet records the finite error-propagation route for Lipschitz estimates whose tolerances are read through dyadic rational data. It sits over the dyadic ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the function-bound surface of chapter 410. The packet names a non-quotient certificate for moving a displayed dyadic input tolerance through a displayed Lipschitz bound before any Real-facing consumer can inspect the sealed output.

Definition 1864.1 (DyadicLipschitzModulus carrier)**.** *A* $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, B, G, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the inspected dyadic input radius, the dyadic output allowance, and the finite comparisons used to certify that the input radius is small enough for the displayed Lipschitz bound.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite input-name and output-name windows whose observations may be used by the modulus computation.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same dyadic ledger and stream windows.*
4. *B is the* $\mathsf{LipschitzMap}^{\uparrow}$ *bound ledger. It stores the displayed Lipschitz constant row and the finite multiplication or scaling comparisons needed to move a dyadic input tolerance to a dyadic output tolerance.*
5. *G is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It may be inspected only after D, S, R, B have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular rational readback, Lipschitz bound ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a modulus request through D, S, R, B, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicLipschitzModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicLipschitzModulus}^{\uparrow}}$ *row.*

The carrier contains no Real completeness row, host function extensionality row, quotient stream equality, selected limit, trichotomy datum, countable-choice selector, Lean verification credential, or standard bridge.

Lean4 ✓ `BEDC.Derived.DyadicLipschitzModulusUp.DyadicLipschitzModulusTasteGate single carrier alignment`

Theorem 1864.2 (DyadicLipschitzModulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ *carrier*

$$M = (D, S, R, B, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicLipschitzModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance ledger, stream-window row, regular rational readback row, Lipschitz bound ledger, Real seal, componentwise transport, replay, provenance, and local naming row;*
2. *classifier exactness, requiring accepted carriers to be compared only through* $D, S, R, B, G, H, C, P, N$*;*
3. *bound admission, requiring* B *to expose the displayed Lipschitz constant and the finite dyadic scaling comparisons used by the output tolerance;*
4. *finite-window admission, requiring* S *to expose only the input and output windows licensed by* D *and* B*;*
5. *RegSeqRat handoff, requiring* R *to read back the same dyadic and stream-window observations rather than a second regular sequence;*
6. *Real seal nonescape, requiring* G *to be reached only after* D, S, R, B *and to export no Real completeness, host function extensionality, quotient stream equality, selected limit, trichotomy, or bridge credential;*
7. *transport and replay stability, requiring* H, C, P, N *to preserve, replay, source, and name the same finite rows without adding an external analytic premise.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1864.1. The accepted packet displays exactly the dyadic tolerance ledger, stream windows, regular rational readback, Lipschitz bound ledger, Real seal, and structural rows.

Classifier exactness is componentwise. Two accepted carriers can be compared only by the displayed coordinates $D, S, R, B, G, H, C, P, N$, so no hidden function equality, quotient-stream equality, selected limit, or trichotomy branch can enter the classifier.

The bound and finite-window clauses are forced by B and S. The Lipschitz row supplies the displayed constant and finite dyadic scaling comparisons, while the stream row supplies only the input and output observations licensed by those dyadic comparisons.

The regular rational and Real-facing clauses are downstream. The row R reads back the same observations as a regular rational name, and G seals the resulting Real-facing value only after D, S, R, B have been displayed.

Finally H, C, P, N preserve the route. Transport is componentwise, replay follows the displayed request path, provenance names the same source packets, and the local NameCert row names only this packet. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 1864.3 (DyadicLipschitzModulus RegSeqRat handoff)**.** *Let $M = (D, S, R, B, G, H, C, P, N)$ be an accepted* $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ *carrier. Every consumer route exported by M factors through the finite path*

$$D \longmapsto S \longmapsto R \longmapsto B \longmapsto G$$

before it can inspect the transport, replay, provenance, or local naming rows. Thus every Real-facing Lipschitz error read passes through dyadic tolerance data, StreamName windows, RegSeqRat readback, the Lipschitz bound ledger, and the terminal $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with an exported consumer route. The carrier has no analytic source row before the dyadic tolerance ledger D, so the input tolerance and the output allowance are read first as displayed dyadic data.

The route then passes through S and R. The stream row exposes only the finite input and output windows licensed by the dyadic ledger, and the regular rational row repacks those same observations as the readback available to later consumers.

Next the route reads B. The displayed Lipschitz constant and dyadic scaling comparisons are the only rows allowed to propagate the input tolerance into an output tolerance.

Only after this finite path has been displayed can the route reach the Real seal G. The remaining rows H, C, P, N transport, replay, source, and name that path, so every exported consumer route factors through the dyadic tolerance, StreamName, RegSeqRat, Lipschitz-bound, and $\mathsf{Real}^{\uparrow}$-seal stages. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicLipschitzModulus}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicLipschitzModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1864.1, Definition 1864.2, and Definition 1864.3 seed the finite L10 route from dyadic tolerance data through StreamName windows, RegSeqRat readback, Lipschitz bound propagation, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicLipschitzModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, host function extensionality, quotient streams, selected limits, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, bound admission, finite-window admission, RegSeqRat handoff, Real seal nonescape, transport stability, and replay stability for BEDC.Derived.DyadicLipschitzModulus

Constructive story: A $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a LipschitzMap bound ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1865

A Concrete Naming Certificate for $\mathsf{SignedDyadicNormalForm}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ packet records the finite signed dyadic normal-form boundary used between the arithmetic surface of chapter 291 and the stream, regular-rational, and real-name handoff surfaces of chapter 126, chapter 301, and chapter 38. It reads signed mantissas through the rational and integer certificates of chapter 37 and chapter 31, records the positive unary exponent and denominator ledger as finite rows, and exposes canonical comparison only after a common-scale replay has been displayed.

The packet is a normal-form interface for L10 real and completion work. It gives downstream chapters a single finite object to cite when they need dyadic comparison, distance, radius, or scale data, without importing quotient rational equality, host integer normal forms, quotient streams, host real equality, trichotomy, choice, or Real completeness.

Definition 1865.1 (Signed dyadic normal-form carrier)**.** *A* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, E, L, C, K, D, T, H, R, P, N)$$

with the following displayed rows:

1. *M is the signed mantissa row, read through the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$ *certificate surfaces rather than through a host integer normal form.*
2. *E is the positive unary exponent row, recording the denominator scale as finite* $\mathsf{UnaryHistory}$ *data.*
3. *L is the denominator ledger row, listing the finite denominator witnesses and the scale requests consumed by the packet.*
4. *C is the common-scale row, displaying the single scale at which two signed dyadic rows may be compared or subtracted.*
5. *K is the normalized comparison row, recording sign, zero, distance, and radius comparisons only after the common-scale row has been projected.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *provenance row for the mantissa, exponent, denominator ledger, common scale, and comparison data.*
7. *T is the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff row, exposing precisely the normal-form data that downstream completion consumers may read.*

8. *H records componentwise* hsame *transport for the mantissa, exponent, denominator ledger, common scale, comparison, dyadic provenance, handoff, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from a signed dyadic request through mantissa inspection, exponent inspection, denominator ledger lookup, common-scale comparison, normalized comparison, DyadicRatCore provenance, downstream handoff, transport, packaging, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, L, C, K, D, T, H, R, P, N$. It has no coordinate for quotient rational equality, host integer canonicalization, quotient stream equality, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SignedDyadicNormalFormUp.SignedDyadicNormalFormTasteGate single carrier alignment`

Theorem 1865.2 (Signed dyadic normal-form NameCert obligations)**.** *For every accepted* SignedDyadicNormalForm *carrier $S = (M, E, L, C, K, D, T, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicNormalForm}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the signed mantissa, positive unary exponent, denominator ledger, common scale, normalized comparison, DyadicRatCore provenance, completion handoff, transport, replay, provenance, and local naming rows;*
2. *signed-mantissa exactness, requiring each mantissa read to pass through the displayed* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$ *rows before any dyadic comparison is exported;*
3. *denominator-ledger exactness, requiring every denominator and scale request to be discharged by E and L before the common-scale row is opened;*
4. *common-scale classifier stability, requiring C and K to be preserved componentwise under* hsame *transport and displayed* Cont *replay;*
5. *completion-handoff nonescape, requiring the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers in T to read only the displayed normal-form route.*

No signed dyadic normal-form obligation is stored outside these rows.

Proof. Project the carrier of Definition 1865.1. The finite packet exposes exactly the signed mantissa, positive unary exponent, denominator ledger, common scale, normalized comparison, DyadicRatCore provenance, handoff, and structural rows.

The mantissa and denominator obligations are forced by the order of the displayed rows. A comparison cannot read the mantissa without passing through M, and it cannot read a denominator scale without the exponent row E and ledger row L.

Common-scale stability is the role of C, K, H, R. The common-scale row fixes the scale, the normalized comparison row records the comparison at that scale, the transport row preserves those coordinates componentwise, and the replay row repeats the same route.

The downstream handoff is then only a reader of the normal form. Since T is downstream of M, E, L, C, K, D and the structural rows contain no excluded coordinate, the local NameCert surface exports no quotient rational equality, host integer normal form, quotient stream equality, host real equality, trichotomy, choice, ambient real completeness, Lean credential, or bridge datum. □

Theorem 1865.3 (Signed dyadic common-scale canonicality)**.** *Let $S = (M, E, L, C, K, D, T, H, R, P, N)$ be an accepted* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *carrier. Every normalized comparison exported by S factors through the finite route*

$$M, E, L \longmapsto C \longmapsto K \longmapsto D, H, R, P, N.$$

Thus the comparison is canonical at the displayed common scale, and two accepted replays with componentwise hsame *rows export the same normalized comparison surface without appealing to quotient equality or trichotomy.*

Proof. Start with an exported normalized comparison. Carrier admission places the only comparison coordinate at K, and K is downstream of the common-scale row C.

The row C is downstream of the signed mantissa, positive unary exponent, and denominator ledger rows. Hence the compared signed dyadics are first read through M, E, L, and the common comparison scale is part of the finite packet before K can be inspected.

Componentwise transport through H preserves M, E, L, C, K as displayed rows, while replay through R repeats the same access path. Provenance and local naming through P, N source and name that route but do not add a second comparison rule.

Therefore a downstream reader receives the normalized comparison at the displayed common scale. The conclusion is canonical for this finite carrier route and uses neither quotient equality nor a trichotomy principle. □

Theorem 1865.4 (Signed dyadic Real handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer named by an accepted* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *carrier reads the route*

$$M, E, L, C, K, D \longmapsto T \longmapsto H, R, P, N.$$

Consequently the handoff supplies signed dyadic normal-form data for stream windows, regular rational Cauchy names, and real-name consumers without importing quotient streams, host real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the handoff row T. It is the only row whose purpose is to expose the signed dyadic packet to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers.

The row T is downstream of the arithmetic and normal-form rows M, E, L, C, K, D. A consumer therefore sees signed mantissa data, positive unary exponent data, denominator ledger data, common-scale comparison data, normalized comparison data, and DyadicRatCore provenance before any stream or completion-facing read is available.

Transport, replay, provenance, and local naming preserve the same route. They do not store quotient streams, host real equality, representative selectors, choice schedules, or completeness principles.

Thus the downstream consumer receives a finite signed dyadic normal-form boundary. The packet is sufficient for stream-window, regular rational Cauchy, and real-name handoffs, but it does not claim any completed real object or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDyadicNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDyadicNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1865.1, Definition 1865.2, Definition 1865.3, and Definition 1865.4 seed the finite signed dyadic normal-form boundary between DyadicRatCore arithmetic and StreamName, RegSeqRat, and Real handoff consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDyadicNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, host integer normal forms, quotient streams, host real equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-mantissa exactness, denominator-ledger exactness, common-scale classifier stability, completion-handoff nonescape, and Real-and-Completion replay for BEDC.Derived.SignedDyadicNormalFormUp.

Constructive story: A SignedDyadicNormalForm$^{\uparrow}$ packet is generated from signed Rat and Int mantissa rows, a positive UnaryHistory exponent row, a denominator ledger, a common-scale row, a normalized comparison row, DyadicRatCore provenance, StreamName, RegSeqRat, and Real handoff support, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1866

A Concrete Naming Certificate for $\mathsf{NormalSpace}^{\uparrow}$

Following Kelley and Munkres, with Urysohn and Bourbaki as the separation-theory background, normal spaces are read here as the finite BEDC surface that separates two displayed closed-set rows by disjoint open-neighborhood rows. The $\mathsf{NormalSpace}^{\uparrow}$ packet sits after the open-set discipline of chapter 88, beside the point-separation surface of chapter 1925, before the separation-function route of **??**, and also beside the positive-open route of chapter 1857, the observable-open classifier of chapter 1860, and the Cauchy-facing topology interface of chapter 1847. It names only the local separation route; it does not import a selected complement, excluded middle, open-cover compactness, choice-selected functions, host powersets, or a function extension theorem.

Definition 1866.1 (Normal space carrier)**.** *A* $\mathsf{NormalSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the ambient open-row discipline and the finite observation surface used by the separation read.*
2. *S is the* $\mathsf{HausdorffSpace}^{\uparrow}$ *compatibility row when a point-separation consumer supplies separated names before closed-pair separation is requested; it is read only as displayed neighbourhood data and is not part of the closed-pair classifier.*
3. *F_0 and F_1 are the two closed-set witness rows, displayed as ledger entries over the same topological source rather than as host complements.*
4. *D is the disjointness ledger, recording that the two closed-set witnesses have no accepted common membership cell inside the displayed packet.*
5. *U_0 and U_1 are the separating open-neighborhood rows, with F_0 contained in U_0, F_1 contained in U_1, and no accepted overlap between U_0 and U_1.*
6. *H records componentwise* hsame *transport for the topology source, closed-set witnesses, disjointness ledger, open-neighborhood rows, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a separation request through $T, F_0, F_1, D, U_0, U_1, H, P, L$.*
8. *P is the* Pkg *provenance over* $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{CozeroSet}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{NormalSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F_0, F_1, D, U_0, U_1, H, K, P, L$, with S available only as a compatibility handoff from the sibling Hausdorff surface. It has no coordinate for selected complements, host closed-set predicates, host powersets, excluded middle, Urysohn functions, choice-selected functions, Tietze extension data, paracompact refinements, classical separation principles not displayed by the rows, or open-cover compactness.

Lean4 (definition) `BEDC.Derived.NormalSpaceUp.NormalSpaceUp`

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpaceTasteGate single carrier alignment`

Theorem 1866.2 (Normal space NameCert obligations)**.** *For every accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier*

$$N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{NormalSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, the two closed-set witnesses, the disjointness ledger, the separating open-neighborhood pair, transport, replay, provenance, and local naming;*
2. *Hausdorff compatibility, requiring any point-separation input to arrive through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *neighbourhood row of chapter 1925 before it can be compared with the closed-pair request;*
3. *closed-pair exposure, requiring every separation request to display F_0, F_1 over the same topology source before any open-neighborhood row can be consumed;*
4. *open-separation exactness, requiring U_0, U_1 to contain the displayed closed witnesses and to carry the packet's own disjointness read for the two open neighborhoods;*
5. hsame *and* Cont *transport, requiring componentwise transport and replay to preserve the topology source, the closed pair, the disjointness ledger, and the two separating open rows;*
6. *Urysohn handoff discipline, requiring any later* $\mathsf{UrysohnLemma}^{\uparrow}$ *consumer of* **??** *to receive only the displayed closed pair, disjointness ledger, and separating open-neighborhood rows;*
7. *no excluded-middle or classical-complement escape, requiring H, K, P, L to export no selected complement, host closed-set predicate, host powerset, excluded middle, choice-selected function, Urysohn function, Tietze extension datum, paracompact refinement, classical separation principle not displayed by the rows, or open-cover compactness principle.*

No local normal-space obligation is stored outside these displayed rows.

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket semantic name certificate`

Proof. Carrier admission is projection from Definition 1866.1. The accepted packet displays exactly the topology source, the closed-pair witnesses, the disjointness ledger, the open-neighborhood pair, and the structural transport, replay, provenance, and naming rows.

Closed-pair exposure follows from the order of the carrier. A separation request first reads T and then the two closed-set witnesses F_0, F_1 over that same source; no open-neighborhood row is available before the closed pair has been displayed.

Open-separation exactness is the projection to D, U_0, U_1. The ledger D records disjointness for the displayed closed pair, while U_0 and U_1 record the two open neighborhoods, the two containment reads, and the open-neighborhood disjointness read internal to the packet.

Transport and replay are exhausted by H and K. These rows move the topology source, closed-pair witnesses, disjointness ledger, and open-neighborhood pair componentwise and replay the same finite route.

The Hausdorff compatibility row is only a sibling handoff. It can supply displayed separated-neighbourhood evidence from chapter 1925, but it cannot replace the closed-pair route or add host point equality.

The Urysohn handoff is also downstream only. A later consumer from **??** receives $T, F_0, F_1, D, U_0, U_1, H, K, P, L$ as displayed rows, not a selected separating function.

Finally, non-escape follows from the absence of any further classifier coordinate. The provenance and local NameCert rows name only this finite separation route, so no row stores a selected complement, excluded middle, host closed-set predicate, host powerset, choice-selected function, Urysohn function, Tietze extension datum, paracompact refinement, classical separation principle not displayed by the rows, or open-cover compactness principle. □

Theorem 1866.3 (Normal space open-separation route)**.** *Every separation read exported by an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *packet factors through the displayed closed-pair row and the two open-neighborhood rows before the local naming row can be inspected.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket open separation route`

Proof. Begin with a separation request for the displayed normal-space surface. Carrier admission exposes the topology source T, but it exports no local normal-space name before the closed-set witnesses are read.

The request then reads the displayed closed pair F_0, F_1 and the disjointness ledger D. This fixes the pair to be separated and prevents substitution by an external complement or host predicate.

Only after F_0, F_1, D have been displayed may the request inspect the open-neighborhood rows U_0, U_1. These rows supply the two containment reads and the disjointness read for the open neighborhoods.

The replay and transport rows then carry this same factorization to the local naming row L. Thus every exported separation read passes through the displayed closed pair and the two separating open-neighborhood rows before local naming. □

Theorem 1866.4 (Normal space Urysohn and cozero localization)**.** *Every* $\mathsf{UrysohnLemma}^{\uparrow}$ *or* $\mathsf{CozeroSet}^{\uparrow}$ *consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier* $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$ *localizes to the displayed topology source, the two closed-set rows, the disjointness ledger, the separating open-neighborhood rows, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local NameCert rows. The consumer receives its normal-space input through* $T, F_0, F_1, D, U_0, U_1, H, K, P, L$ *and not through selected complements, host powersets, choice-selected Urysohn functions, detached cozero predicates, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket urysohn cozero localization`

Proof. Project the carrier of Definition 1866.1. The only normal-space rows exposed to a downstream consumer are the topology source, the closed pair, the disjointness ledger, the separating open-neighborhood pair, and the structural transport, replay, provenance, and naming rows.

Apply Definition 1866.2. Its Urysohn handoff discipline and non-escape row say that the consumer reads closed-pair exposure and open-separation exactness as displayed NameCert data, not as an ambient function-selection principle.

Use Definition 1866.3. It forces the read through F_0, F_1, D before U_0, U_1, and through those open-neighborhood rows before L.

Thus a $\mathsf{CozeroSet}^{\uparrow}$ consumer may reuse the positive-open boundary already displayed by U_0, U_1, and a $\mathsf{UrysohnLemma}^{\uparrow}$ consumer may reuse the same closed-pair and open-neighborhood route. Neither consumer obtains selected complements, host powersets, choice-selected Urysohn functions, detached cozero predicates, or open-cover compactness from the $\mathsf{NormalSpace}^{\uparrow}$ packet. □

Theorem 1866.5 (Normal space closed-set separation route)**.** *For every accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier* $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$*, any closed-set separation consumer must read* F_0 *and* F_1 *over the displayed topology source* T*, then the disjointness ledger* D*, then the separating open-neighborhood rows* U_0 *and* U_1*, before it can inspect provenance or the local NameCert row. The route is preserved by* H *and* K *and exports no selected complement, host closed-set predicate, host powerset, or open-cover compactness witness.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket closed set separation route`

Proof. Start with carrier admission from Definition 1866.1. The closed-set rows F_0 and F_1 are coordinates of the same packet as the topology source T, so they are read over that source rather than over an external host predicate.

Apply the NameCert obligation theorem Definition 1866.2. Closed-pair exposure requires F_0, F_1 before any open-neighborhood row is consumed, and open-separation exactness requires U_0, U_1 to contain the displayed closed witnesses while carrying the packet's own disjointness read.

Now apply Definition 1866.3. It places the local name downstream of the closed pair, the disjointness ledger, and the two open-neighborhood rows.

The componentwise hsame row H and displayed Cont replay row K preserve that order. Since no selected complement, host closed-set predicate, host powerset, or open-cover compactness witness appears as a carrier coordinate, the closed-set separation consumer cannot read such data from the packet. □

Theorem 1866.6 (Normal space Cauchy and Hausdorff consumer boundary)**.** *Every* $\mathsf{CauchySpace}^{\uparrow}$ *or* $\mathsf{HausdorffSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier factors its topology-facing read through the displayed source* T*, the closed-set rows* F_0, F_1*, the disjointness ledger* D*, the separating open-neighborhood rows* U_0, U_1*, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local NameCert rows. Hausdorff point-separation data may enter only as the displayed compatibility handoff described by the carrier, and Cauchy-facing topology reads receive no selected complements, host powersets, choice-selected Urysohn functions, quotient completion data, or open-cover compactness from the normal-space packet.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket cauchyspace hausdorff consumer boundary`

Proof. Project the accepted packet. The topology source T and the optional $\mathsf{HausdorffSpace}^{\uparrow}$ compatibility handoff are displayed before the closed-pair route, while the closed-set rows, disjointness ledger, and open-neighborhood rows remain the only normal-space separation coordinates.

Use Definition 1866.2. The Hausdorff compatibility obligation allows point-separation information only as displayed neighbourhood data from the sibling Hausdorff surface; it does not replace the closed-pair classifier or add host point equality.

Apply Definition 1866.3. Any Cauchy-facing topology read that consumes normal-space separation must pass through F_0, F_1, D, U_0, U_1 before reaching provenance or local naming, and H, K replay the same finite route after transport.

Therefore a $\mathsf{CauchySpace}^{\uparrow}$ consumer receives only the topology and separation rows needed for its local topology interface, while a $\mathsf{HausdorffSpace}^{\uparrow}$ consumer receives only the displayed compatibility handoff and the closed-pair route. Selected complements, host powersets, choice-selected

Urysohn functions, quotient completion data, and open-cover compactness are absent from the carrier and cannot be exported. □

Theorem 1866.7 (Normal space closed-pair stability)**.** *For every accepted* NormalSpace$^{\uparrow}$ *carrier* $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$*, componentwise* hsame *transport and displayed* Cont *replay preserve the closed-pair and separating-open route. The transported packet still reads* F_0, F_1 *over the transported topology source, still reads* D *as the disjointness ledger for that displayed pair, still reads* U_0, U_1 *as the separating open-neighborhood rows, and still reaches the local name only after those rows have been replayed.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket closed pair stability`

Proof. Project the carrier of Definition 1866.1. The closed-pair rows, disjointness ledger, separating open-neighborhood rows, componentwise transport row, and replay row are all explicit coordinates of the accepted packet.

Apply the open-separation route Definition 1866.3. It fixes the order in which a separation read reaches local naming: first the topology source and closed pair, then the disjointness ledger, then the two open-neighborhood rows.

The transport row H moves those displayed coordinates componentwise, while the replay row K replays the same finite route after transport. Since no carrier coordinate stores a selected complement, external closed-set predicate, or host powerset, stability can preserve only the displayed closed-pair and separating-open rows. □

Theorem 1866.8 (Normal space cozero-Urysohn handoff)**.** *Every* CozeroSet$^{\uparrow}$ *or* UrysohnLemma$^{\uparrow}$ *consumer of an accepted* NormalSpace$^{\uparrow}$ *packet can read only the displayed topology source, closed-pair rows* F_0, F_1*, disjointness ledger* D*, separating open-neighborhood rows* U_0, U_1*, componentwise transport, displayed replay, provenance, and local naming. Such a consumer receives no selected complement, host powerset, choice-selected separating function, detached cozero predicate, or ambient classical separation principle from the normal-space packet.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket cozero urysohn handoff`

Proof. Start with the NameCert obligation surface of Definition 1866.2. It exposes closed-pair data, open-separation exactness, transport, replay, provenance, local naming, and the Urysohn handoff discipline as displayed rows.

Use Definition 1866.3 and Definition 1866.7. These force downstream reads to pass through the same closed-pair, disjointness, and separating-open rows even after componentwise transport and replay.

A CozeroSet$^{\uparrow}$ consumer may therefore read only the positive-open boundary already contained in those displayed open-neighborhood rows. A UrysohnLemma$^{\uparrow}$ consumer may read the same closed-pair and open-neighborhood route, but it must supply any separating-function certificate separately.

The carrier contains no coordinate for selected complements, host powersets, choice-selected functions, detached cozero predicates, or classical separation principles. Hence neither downstream consumer can obtain those data from the NormalSpace$^{\uparrow}$ packet. □

Theorem 1866.9 (Normal space closed-pair transport)**.** *For every accepted* NormalSpace$^{\uparrow}$ *carrier* $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$*, componentwise* hsame *transport preserves the displayed closed-pair surface. A transported separation read still exposes* F_0 *and* F_1 *over the same topology source* T*, and the transported disjointness row is the image of the displayed ledger* D *rather than an external closed-set predicate or complement choice.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket closed pair transport`

Proof. Project the finite rows from Definition 1866.1. The carrier stores T, F_0, F_1, D before the open-neighborhood rows and before the local naming row.

Apply the transport obligation from Definition 1866.2. The row H moves the topology source, the two closed-set witnesses, and the disjointness ledger componentwise, so the transported request still reads the same displayed closed pair over the same transported topology source.

Finally apply the route theorem Definition 1866.3. Since local naming is unavailable before F_0, F_1, D have been displayed, the transported packet cannot replace the closed-pair data by a host complement, selected predicate, or external disjointness witness. □

Theorem 1866.10 (Normal space open-neighborhood stability)**.** *For every accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier* $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$*, the separating open-neighborhood rows remain stable under the same componentwise transport. The transported rows* U_0 *and* U_1 *still contain the transported closed witnesses, still carry the packet's own disjointness read, and still precede every inspection of the local naming row* L*.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket open neighborhood stability`

Proof. The NameCert obligation surface in Definition 1866.2 identifies open-separation exactness as a displayed row, not as an ambient theorem about normal spaces. Thus U_0 and U_1 store the two containment reads and the open-neighborhood disjointness read inside the carrier.

Use Definition 1866.9 to transport the closed witnesses and the disjointness ledger. The componentwise transport row H then moves U_0 and U_1 with those witnesses, preserving which closed row each neighborhood contains.

The factorization of Definition 1866.3 forces the transported read through the closed pair and the two separating open rows before local naming. Therefore the open-neighborhood stability is part of the displayed finite route and does not import Urysohn, Tietze, paracompact, or open-cover data. □

Theorem 1866.11 (Normal space Urysohn handoff)**.** *Every* $\mathsf{UrysohnLemma}^{\uparrow}$ *consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *packet must read the route*

$$\mathsf{Topology}^{\uparrow}, \mathsf{HausdorffSpace}^{\uparrow}, F_0, F_1, D, U_0, U_1, H, K, P, L \longmapsto \mathsf{UrysohnLemma}^{\uparrow}$$

through the displayed rows of Definition 1866.1. The handoff exports the topology source, optional Hausdorff neighbourhood compatibility, the closed-set pair, the disjointness ledger, the two separating open-neighborhood rows, componentwise transport, displayed replay, provenance, and local naming; it exports no choice-selected separating function, host powerset of opens, selected complement, excluded-middle branch, or classical separation principle not already displayed by those rows.

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket topology consumer boundary`

Proof. Start with the carrier projection. The topology source T is the only ambient open-row discipline available to the packet, and any point-separation compatibility has already been confined to the displayed $\mathsf{HausdorffSpace}^{\uparrow}$ handoff described in Definition 1866.1.

The closed-pair request then reads F_0, F_1, D. These rows expose the two closed-set witnesses and their disjointness ledger over the same topology source, so the request cannot be replaced by a host powerset predicate or by a selected complement.

Only after that read does the packet expose U_0 and U_1. Those rows are precisely the separating open neighborhoods consumed by the sibling $\mathsf{UrysohnLemma}^{\uparrow}$ route of **??**.

The structural rows H, K, P, L preserve and name this route. Because the carrier has no coordinate for a selected separating function, a host powerset of opens, excluded middle, or a classical separation principle outside the displayed rows, the Urysohn-facing consumer receives only the stated finite handoff. □

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket urysohn handoff`

Theorem 1866.12 (Normal space topology-consumer boundary)**.** *Every topology-facing consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier can inspect only the rows* $T, F_0, F_1, D, U_0, U_1, H, K, P, L$*. Such a consumer may read the topology source, the displayed closed pair, the disjointness ledger, the separating open-neighborhood rows, componentwise transport, replay, provenance, and local naming, but it cannot import Urysohn functions, Tietze extension data, paracompact refinements, selected complements, excluded middle, or open-cover compactness.*

Proof. Start with the carrier projection of Definition 1866.1. The accepted packet has exactly the listed rows $T, F_0, F_1, D, U_0, U_1, H, K, P, L$ available to a topology-facing consumer.

The obligation theorem Definition 1866.2 says that all local normal-space obligations are exhausted by carrier admission, closed-pair exposure, open-separation exactness, transport, replay, provenance, local naming, and non-escape. The closed-pair transport theorem and the open-neighborhood stability theorem keep those same rows stable under componentwise hsame transport.

It remains only to rule out extra topology principles. The open-separation route Definition 1866.3 exposes no coordinate beyond the displayed rows, so no consumer can read Urysohn, Tietze, paracompact, selected-complement, excluded-middle, or open-cover compactness data from this packet. □

Theorem 1866.13 (Normal space Tietze extension refusal)**.** *Every accepted* $\mathsf{NormalSpace}^{\uparrow}$ *packet exposes only the displayed topology source, closed-pair rows, disjointness ledger, and two separating open-neighborhood rows, together with componentwise transport, replay, provenance, and local naming. It does not expose a continuous extension row, a selected function, a host powerset of closed or open sets, a Tietze extension theorem, or any choice-selected extension datum.*

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket tietze extension refusal`

Proof. Start with the carrier of Definition 1866.1. Its public rows are the topology source, the two closed-set witnesses, the disjointness ledger, the separating open-neighborhood pair, and the structural rows.

Apply Definition 1866.2 and Definition 1866.3. These identify the local obligation surface with closed-pair exposure and open-separation exactness, not with extension of functions.

Use Definition 1866.10 and Definition 1866.12. Stability preserves only the displayed open-neighborhood route, while the topology boundary refuses Urysohn functions, Tietze extension data, host powersets, selected complements, excluded middle, and open-cover compactness. Hence no Tietze extension row is exported by the packet. □

Theorem 1866.14 (Normal space Urysohn lemma consumer exactness)**.** *An* $\mathsf{UrysohnLemma}^{\uparrow}$ *consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *packet may read the displayed closed pair* F_0, F_1*, disjointness*

ledger D, and separating open-neighborhood rows U_0, U_1 through Definition 1866.11. It must supply any separating-function certificate separately; the normal-space packet itself exports no choice-selected function, host powerset, excluded-middle branch, Tietze extension datum, or classical separation principle beyond those displayed rows.

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket urysohn lemma consumer exactness`

Proof. Project the carrier in Definition 1866.1. The Urysohn-facing data available to a consumer are exactly the topology source, the closed pair, the disjointness ledger, the open-neighborhood pair, and the structural rows.

Apply Definition 1866.11. The handoff lets the consumer read the displayed closed pair and separating open neighborhoods, but it names no separating function row.

The exactness of that boundary is fixed by Definition 1866.3, Definition 1866.10, and Definition 1866.12. These results keep the consumer on the displayed normal-space route and exclude host powersets, selected functions, Tietze data, excluded middle, and classical separation principles not present in the carrier. □

Theorem 1866.15 (Normal space scoped kernel grounding)**.** *The* $\mathsf{NormalSpace}^{\uparrow}$ *obligation surface is scoped over the displayed closed-pair rows F_0, F_1, D, separating open-neighborhood rows U_0, U_1, cozero-Urysohn handoff rows, topology-consumer rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows of Definition 1866.1. The scope imports neither unrestricted Tietze extension nor host topological equality.*

Proof. Begin with Definition 1866.2. It identifies the local certificate obligations with carrier admission, closed-pair exposure, open-separation exactness, Hausdorff compatibility, Urysohn handoff discipline, transport, replay, provenance, local naming, and non-escape.

Use the route and stability results next. The open-separation route of Definition 1866.3, closed-pair stability of Definition 1866.7, closed-pair transport of Definition 1866.9, and open-neighborhood stability of Definition 1866.10 keep every separation read on the displayed $F_0, F_1, D, U_0, U_1, H, K, P, L$ rows.

For downstream consumers, apply the cozero-Urysohn handoff of Definition 1866.11 and the topology-consumer boundary of Definition 1866.12. They allow only the displayed separation route to feed Urysohn-facing, cozero-facing, and topology-facing consumers.

Finally apply the Tietze refusal of Definition 1866.13. Since unrestricted Tietze extension, selected functions, host powersets, and host topological equality are not carrier coordinates, the scoped kernel is exactly the displayed normal-space route and no larger ambient separation principle. □

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket scoped kernel grounding`

Theorem 1866.16 (Normal space public separation certificate)**.** *Every public topology-facing consumer of an accepted* $\mathsf{NormalSpace}^{\uparrow}$ *carrier $N = (T, F_0, F_1, D, U_0, U_1, H, K, P, L)$ receives one separation certificate surface consisting of closed-pair stability, closed-pair transport, open-neighborhood stability, the* $\mathsf{UrysohnLemma}^{\uparrow}$ *handoff, and the topology-consumer boundary. The public read exposes only*

$$T, F_0, F_1, D, U_0, U_1, H, K, P, L$$

and exports no Tietze extension row, selected complement, choice-selected separating function, host powerset, excluded-middle branch, or open-cover compactness principle.

Lean4 ✓ `BEDC.Derived.NormalSpaceUp.NormalSpacePacket public separation certificate`

Proof. Start with the NameCert obligation surface and the scoped kernel grounding. Theorems 1866.2 and 1866.15 identify the finite separation rows as the topology source, closed pair, disjointness ledger, separating open neighborhoods, transport, replay, provenance, and local naming.

Apply the stability and transport rows. Theorems 1866.7, 1866.9, and 1866.10 keep the closed witnesses, the disjointness ledger, and the two open neighborhoods on the displayed route after componentwise hsame transport and Cont replay.

For downstream consumers, use Definition 1866.11 and Definition 1866.12. The Urysohn-facing read receives the closed-pair and open-neighborhood data as input only, while the general topology-facing boundary permits no extra coordinate beyond $T, F_0, F_1, D, U_0, U_1, H, K, P, L$.

Finally apply Definition 1866.13 and Definition 1866.14. Since Tietze extension, selected complements, host powersets, excluded middle, selected separating functions, and open-cover compactness are absent from the carrier, the public certificate exports only the finite normal-space separation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalSpace}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NormalSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: The consumer-boundary and topology-route theorems Definition 1866.3, Definition 1866.9, Definition 1866.10, Definition 1866.11, Definition 1866.14, Definition 1866.12, and Definition 1866.16 bind the closed-pair, open-neighborhood, Urysohn-facing, cozero-facing, and topology-facing consumer lattice.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NormalSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Tietze extension, selected complements, choice-selected functions, host powersets, excluded middle, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a Lean bridge witness for the displayed closed-pair and open-neighborhood handoff; this chapter does not claim that checked bridge.

Constructive story: A $\mathsf{NormalSpace}^{\uparrow}$ packet is generated from a Topology source row, a Hausdorff compatibility handoff when present, two closed-set witness rows, a disjointness ledger, two separating open-neighborhood rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1867

A Concrete Naming Certificate for $\mathsf{ProductFilter}^{\uparrow}$

Following Bourbaki and Kelley on product filters and filter bases, the $\mathsf{ProductFilter}^{\uparrow}$ packet records a finite BEDC product-base route over the filter-base surface of chapter 1601, the Cauchy-filter surface of chapter 333, the stream-window surface of chapter 126, and the regular rational read-back surface of chapter 301. It also keeps the dyadic tolerance and real-facing boundary visible through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, so completion consumers can read a product filter base without importing an ultrafilter, a quotient filter, a selected limit, choice data, host real equality, or ambient completeness.

Definition 1867.1 (Product filter finite carrier)**.** *A* $\mathsf{ProductFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B_X, B_Y, I, \pi_X, \pi_Y, D, R, H, C, P, N)$$

with the following displayed rows:

1. *B_X is the first* $\mathsf{FilterBase}^{\uparrow}$ *row. It stores the finite base packets, their directed lower-bound witnesses, and their local refinement ledger for the first component.*
2. *B_Y is the second* $\mathsf{FilterBase}^{\uparrow}$ *row, with the same displayed filter-base boundary for the second component.*
3. *I is the product index row. Each entry names a finite product-base request assembled from one displayed request in B_X and one displayed request in B_Y.*
4. *π_X and π_Y are the projection rows. They replay each product-base request back to its two component base requests before any Cauchy or completion-facing consumer can inspect the packet.*
5. *D is the directed lower-bound witness row. For two displayed product-base requests it reads component lower bounds from B_X and B_Y, pairs them in I, and records the two projection-compatible containment reads.*
6. *R is the refinement ledger. It records product refinement only through component refinements and the projection rows π_X, π_Y.*
7. *H records componentwise* hsame *transport for the two component bases, the product index row, both projection rows, the lower-bound witness row, the refinement ledger, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a product-base request through*

$$B_X, B_Y \longmapsto I, \pi_X, \pi_Y \longmapsto D, R \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{ProductFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ProductFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B_X, B_Y, I, \pi_X, \pi_Y, D, R, H, C, P, N$. There is no coordinate for ultrafilters, quotient filters, selected limits, countable choice, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ProductFilterUp.ProductFilterTasteGate single carrier alignment`

Theorem 1867.2 (Product filter NameCert obligations)**.** *For every accepted* $\mathsf{ProductFilter}^{\uparrow}$ *carrier*

$$F = (B_X, B_Y, I, \pi_X, \pi_Y, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProductFilter}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the two component filter-base rows, the product index row, the projection rows, the directed lower-bound row, the refinement ledger, transport, replay, provenance, and local naming;*
2. *projection stability, requiring every product-base request to replay through π_X and π_Y to the same component requests used to assemble it;*
3. *directed lower-bound coverage, requiring D to build product lower bounds only from displayed component lower bounds and to record their two containment reads in R;*
4. $\mathsf{CauchyFilter}^{\uparrow}$ *handoff, requiring every Cauchy-filter consumer to receive the product filter only through the finite product-base route $B_X, B_Y, I, \pi_X, \pi_Y, D, R$ and the structural rows H, C, P, N;*
5. *completion-facing nonescape, requiring the same route to export no ultrafilter, quotient filter, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.*

No local product-filter obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1867.1. The accepted packet exposes exactly the two component base rows, the product index row, the two projection rows, the lower-bound witness row, the refinement ledger, and the structural rows H, C, P, N.

Projection stability follows from the order of the carrier. A product-base request is admitted by I only together with the component requests replayed by π_X and π_Y, so later transport cannot replace the component base data by an external filter object.

Directed lower-bound coverage is componentwise. The row D reads lower-bound witnesses from B_X and B_Y, pairs them as another entry of I, and records the two product containment reads in R.

The Cauchy-filter handoff is exhausted by the displayed route. A downstream consumer sees the same finite base route after hsame transport by H and Cont replay by C; the provenance row P and naming row N name that route without adding any further coordinate.

Since ultrafilters, quotient filters, selected limits, countable-choice schedules, host real equality, and ambient completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 1867.3 (Product filter completion handoff)**.** *Let* $F = (B_X, B_Y, I, \pi_X, \pi_Y, D, R, H, C, P, N)$ *be an accepted* $\mathsf{ProductFilter}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$*-facing consumer of the product filter factors through the displayed product-base route*

$$B_X, B_Y \longmapsto I, \pi_X, \pi_Y \longmapsto D, R \longmapsto H, C, P, N.$$

Consequently the consumer cannot read an ultrafilter, quotient filter, selected limit, choice row, host real equality, or ambient completeness witness from the product filter packet.

Proof. Start with a consumer request. The carrier has no product-filter coordinate except the product index row and its projections, so the request first reaches I, π_X, π_Y and thereby names the two component base requests.

If the consumer asks for refinement or common lower bounds, the route continues through D and R. These rows use only displayed lower bounds and refinements from the component filter bases B_X and B_Y, then repack the result as another product-base row.

A $\mathsf{CauchyFilter}^{\uparrow}$ consumer therefore receives a directed finite product base, not a completed filter. The same finite route is the only surface exposed to $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$-facing provenance reads.

Transport and replay do not enlarge the boundary. The rows H and C preserve the listed coordinates componentwise, while P and N package and name only this product-base route.

Thus every completion-facing read factors through the displayed carrier. Because the carrier contains no ultrafilter, quotient filter, selected limit, choice row, host real equality, or ambient completeness witness, none can be read downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProductFilter}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProductFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1867.1, Definition 1867.2, and Definition 1867.3 seed the finite product-filter base route used by Cauchy-filter, completion, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProductFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilters, quotient filters, selected limits, countable choice, host real equality, Lean verification, bridge checking, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, projection stability, directed lower-bound coverage, CauchyFilter handoff, completion-facing nonescape, and ledger-refusal proofs for BEDC.Derived.ProductFilterUp.

Constructive story: A $\mathsf{ProductFilter}^{\uparrow}$ packet is generated from two FilterBase rows, one product index row, two projection rows, directed lower-bound witnesses, a refinement ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1868

A Concrete Naming Certificate for $\mathsf{CauchyTailBasis}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy terminology and Kelley's filter-basis terminology, the $\mathsf{CauchyTailBasis}^{\uparrow}$ packet names the finite tail-basis surface that sits between tail-index requests and the L10 Real and Completion route. It refines the finite cofinal-tail filter of chapter 1119, reads finite windows from chapter 126, uses dyadic tolerance ledgers from chapter 291, hands regular rational readback to chapter 301, and remains upstream of the terminal Real seal of chapter 38.

The packet is a basis certificate rather than a completion theorem. It gives downstream Cauchy and Real consumers one named finite route for tail-index refinement, window comparison, tolerance budgeting, and regular readback, while excluding quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking.

Definition 1868.1 (CauchyTailBasis carrier)**.** *A* $\mathsf{CauchyTailBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the tail-index basis row, listing the finite tail requests, tail refinements, and common-tail meets allowed to act as basis elements.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row read over the displayed tail-index basis.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same tail requests and finite windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, carrying only the regular rational observations supported by T, W, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after the tail basis, finite windows, dyadic tolerances, and regular readback have been displayed.*
6. *H records componentwise* hsame *transport for the tail-index basis, finite-window, dyadic-ledger, regular-readback, Real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail-basis request through T, W, D, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyTailBasis}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, W, D, R, E, H, C, P, N$. The carrier contains no quotient stream equality, selected limit, countable-choice selector, host real equality row, trichotomy row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, Lean verification credential, or standard bridge.*

Lean4 ✓ `BEDC.Derived.CauchyTailBasisUp.TasteGate.CauchyTailBasisTasteGate single carrier alignment`

Theorem 1868.2 (CauchyTailBasis NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailBasis}^{\uparrow}$ *carrier $B = (T, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailBasis}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed tail-index basis, finite StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming row;*
2. *tail-basis locality, requiring every request to enter through a displayed element of T before it can inspect W, D, R, or E;*
3. *finite-window exactness, requiring W to expose only windows supported by the admitted tail basis;*
4. *tolerance-ledger exactness, requiring D to record only the dyadic budgets attached to those admitted windows;*
5. *regular-readback attachment, requiring R to reuse the same T, W, D surface rather than selecting a second stream representative;*
6. *Real-seal boundary, requiring E to remain downstream of T, W, D, R;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no row from which quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge credentials can be exported.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1868.1. The accepted packet displays exactly the tail-index basis, finite-window, dyadic-ledger, regular-readback, Real-seal, and structural rows.

Tail-basis locality is forced by the row order. A consumer reads T before it can inspect a StreamName window, a dyadic tolerance, a regular rational readback, or a Real seal.

The window, tolerance, and readback clauses are then component projections. The row W contains only finite windows supported by T, the row D contains only budgets attached to those windows, and the row R is tied to T, W, D rather than to a detached representative.

The Real-seal boundary is the projection to E. Since E is downstream of T, W, D, R, it cannot introduce a quotient stream, a selected limit, a choice schedule, host equality, trichotomy, or ambient completeness.

Finally, H, C, P, N preserve, replay, source, and name the same displayed packet. These rows have no coordinate outside $T, W, D, R, E, H, C, P, N$, so the local NameCert obligations are exhausted by the listed finite rows. □

Theorem 1868.3 (CauchyTailBasis window directedness)**.** *Let $B = (T, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyTailBasis}^{\uparrow}$ *carrier. Any two finite window reads exported by W and admitted over tail-basis elements of T have a displayed common-tail read in T whose dyadic tolerance ledger in D supports both windows before the route reaches R or E.*

Proof. Start with two exported finite-window reads. Since W is read only after the tail-index basis row, each window has an admitted tail-basis element in T.

The row T is a basis row, so its displayed data include finite tail refinements and common-tail meets for the admitted requests. Project the meet row supplied for the two chosen tail elements.

Read D at the displayed meet. The tolerance ledger is attached to the same tail-basis and window rows, so the dyadic budgets for both windows are recorded before any regular readback or Real seal can be inspected.

The structural rows preserve this directed route. Componentwise hsame transport and displayed Cont replay may move the accepted packet along the recorded comparison, but they do not add a second window source or an external tail oracle. □

Theorem 1868.4 (CauchyTailBasis RegSeqRat handoff)**.** *Every regular rational readback exported by an accepted* $\mathsf{CauchyTailBasis}^{\uparrow}$ *carrier* $B = (T, W, D, R, E, H, C, P, N)$ *factors through the finite route*

$$T, W, D \longmapsto R \longmapsto E$$

before any Real-facing consumer can read the terminal seal. Thus the packet hands tail-basis Cauchy data to $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an exported regular rational readback. Carrier admission provides no analytic row before T, so the read first chooses an admitted finite tail-basis element.

The route next reads the finite-window row W and the dyadic tolerance ledger D. These two rows fix the observed stream entries and the tolerance budget for the admitted tail-basis element.

The regular readback row R is attached to the displayed T, W, D data. It therefore cannot select a second stream representative, a quotient class, a detached limit, or a countable-choice schedule.

The Real seal row E is downstream of the same readback. Since the structural rows H, C, P, N only transport, replay, source, and name this finite route, every Real-facing consumer receives the displayed handoff and no ambient completeness, trichotomy, or host equality coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailBasis}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailBasis}^{\uparrow}}$, seed)

Scope closed: Definition 1868.1, Definition 1868.2, Definition 1868.3, and Definition 1868.4 seed the finite tail-basis route over $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligation exactness, window directedness, regular rational handoff, Real-seal boundary, and non-escape proofs for BEDC.Derived.CauchyTailBasisUp.

Constructive story: A $\mathsf{CauchyTailBasis}^{\uparrow}$ packet is generated from a tail-index basis row, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1869

A Concrete Naming Certificate for $\mathsf{CauchyCompletionContinuation}^{\uparrow}$

Following Mac Lane's Kleisli category terminology for continuation-style monad consumers, and Bishop–Bridges and Kelley completion terminology for Cauchy completion interfaces, the CauchyCompletionContinu packet names the finite route by which a continuation request reads a bind source before it can reach the L10 completion face. It sits after the completion monad of chapter 378, the Kleisli consumer boundary of chapter 1845, the unit certificate of chapter 1527, the multiplication certificate of chapter 1829, the stream surface of chapter 126, the regular rational readback of chapter 301, the dyadic observation ledger of chapter 291, and the terminal Real seal of chapter 38.

The packet is a continuation-facing certificate rather than a choice of completed limits. It records the finite request row, bind source row, schedule, stream window, dyadic observation, regular readback, Real seal, and structural transport by which downstream completion consumers replay a bind request.

Definition 1869.1 (Cauchy completion continuation carrier)**.** *A* $\mathsf{CauchyCompletionContinuation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, B, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the continuation request row, naming the finite consumer request that asks for a completed value through a Kleisli bind route.*
2. *B is the bind source row inherited from* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *and* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*, displaying the source family before any downstream window or seal is read.*
3. *S is the finite unary schedule row attached to the request and bind source.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the schedule.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation and tolerance row read from the same finite window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to W and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, B, S, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the continuation request, bind source, schedule, stream window, dyadic observation, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the continuation request through* $Q, B, S, W, D, R, E, H, P, N$.

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionContinuation}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, CauchyCompletionMona $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMultiplication}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionContinuation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, S, W, D, R, E, H, C, P, N$. The carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, hidden representatives, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionContinuationUp.CauchyCompletionContinuationTasteGate`
`single carrier alignment`

Theorem 1869.2 (Cauchy completion continuation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionContinuation}^{\uparrow}$ *carrier $T = (Q, B, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionConti}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the continuation request, bind source, schedule, finite stream window, dyadic observation and tolerance row, regular rational readback, Real seal, transport, replay, provenance, and local naming row;*

2. *request admission, requiring every continuation consumer to display Q before it can read a bind source or public L10 row;*

3. *bind-source admission, requiring every continuation replay to pass through B before opening S, W, D, R, E;*

4. *schedule and window exactness, requiring S, W to be the only finite stream-window surface exposed by the request;*

5. *readback and seal coherence, requiring D, R, E to be read in the displayed order and to remain stable under componentwise* hsame *transport and displayed* Cont *replay;*

6. *non-escape, requiring P, N to source and name no quotient stream, selected limit, countable-choice selector, host equality, hidden representative, ambient completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1869.1. The accepted packet displays exactly the continuation request, bind source, finite schedule, stream window, dyadic observation, regular rational readback, Real seal, and structural rows.

Request admission is supplied by Q. Since Q is the first coordinate of the continuation-facing route, a consumer cannot inspect the bind source or the public completion face before the request row is displayed.

Bind-source admission is supplied by B and by the Kleisli boundary of chapter 1845. The route fixes the bind source before opening the shared schedule, window, dyadic observation, readback, and seal.

Schedule and window exactness are projections to S, W. The packet contains no second schedule or hidden window family, so every continuation replay uses the displayed finite window surface.

Readback, seal coherence, and non-escape are structural. Componentwise transport H, continuation replay C, provenance P, and the local name N preserve and source only the displayed continuation packet, so no excluded completion coordinate is available. □

Theorem 1869.3 (Cauchy completion continuation bind route)**.** *Every accepted* $\mathsf{CauchyCompletionContinuation}^{\uparrow}$ *carrier* $T = (Q, B, S, W, D, R, E, H, C, P, N)$ *routes a continuation request through the bind source before any public L10 completion row is read:*

$$Q \longmapsto B \longmapsto S \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

Thus the StreamName window, DyadicRatCore observation, RegSeqRat readback, and Real seal are continuation results of the displayed bind source rather than detached representatives.

Proof. Start with an accepted packet $T = (Q, B, S, W, D, R, E, H, C, P, N)$. Projection gives the continuation request row Q before any downstream coordinate is available.

The bind source row B is the next displayed coordinate. It is inherited from the Kleisli and monad surfaces, so the continuation reads the source family before opening a schedule or stream window.

Only after Q, B have been read does the carrier expose S, W, D, R, E. The route-order inspection of the tuple therefore gives the displayed chain from request through bind source, schedule, window, dyadic observation, regular readback, and Real seal.

The structural rows H, C, P, N transport, replay, source, and name this same chain. They do not insert a detached stream representative or a second seal-facing route. □

Theorem 1869.4 (Cauchy completion continuation Real seal nonescape)**.** *Let* $T = (Q, B, S, W, D, R, E, H, C, P, N$ *be an accepted* $\mathsf{CauchyCompletionContinuation}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of* T *receives the terminal seal* E *only after the displayed continuation request, bind source, schedule, finite stream window, dyadic observation, and regular rational readback have been read. The consumer receives no quotient stream, selected limit, countable-choice selector, host equality comparison, hidden representative, ambient completeness datum, completeness theorem for* $\mathsf{Real}^{\uparrow}$*, Lean verification credential, or bridge credential.*

Proof. Project the accepted carrier. The only seal coordinate is E, and it appears downstream of Q, B, S, W, D, R.

The replay row C confirms the same order for a continuation request. A Real-facing consumer therefore reaches the seal only through the displayed request, bind source, schedule, window, dyadic observation, and readback route.

Transport and provenance do not enlarge the route. The hsame row H is componentwise, P cites only the listed completion and L10 source chapters, and N names the local certificate.

Since none of the excluded objects occurs as a carrier coordinate, classifier row, or replay target, no $\mathsf{Real}^{\uparrow}$-facing consumer can escape to quotient streams, selected limits, countable choice, host equality, hidden representatives, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionContinuation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionContinuation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1869.1, Definition 1869.2, Definition 1869.3, and Definition 1869.4 seed the finite continuation-facing route from a request through a bind source and into the L10 Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionContinuation}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyCompletionContinuationUp.CauchyCompletionContinuationTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality, hidden representatives, ambient completeness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request admission, bind-source admission, schedule-window exactness, readback-seal coherence, bind-route, and Real-seal non-escape proofs for BEDC.Derived.CauchyCompletionContin

Constructive story: A CauchyCompletionContinuation$^{\uparrow}$ packet is generated from a continuation request row, a bind source row, a finite schedule, StreamName windows, DyadicRatCore observations, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1870

A Concrete Naming Certificate for $\mathsf{CauchyTightness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyTightness}^{\uparrow}$ packet names the finite tight-tail route by which a Cauchy read becomes tight enough for the Real and Completion phase. It sits after the tail-basis route of chapter 1868, the convergence-criterion route of chapter 1509, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the Cauchy limit seal of chapter 386, and the terminal Real seal of chapter 38.

The packet is a finite tightness certificate rather than a completeness principle. It records the tail-basis, convergence-criterion, threshold, dyadic, readback, limit-seal, Real-seal, transport, replay, provenance, and local NameCert rows that a downstream consumer must read before treating a Cauchy tail as tight.

Definition 1870.1 (Cauchy tightness carrier)**.** *A* $\mathsf{CauchyTightness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, G, M, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CauchyTailBasis}^{\uparrow}$ *row, displaying the finite tail requests, refinements, and common-tail basis meets through which tightness may be inspected.*
2. *G is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row, displaying the criterion that converts the admitted tail-basis requests into convergence-facing tolerance reads.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, or the equivalent finite tail-threshold data, assigning each dyadic tolerance request a displayed threshold over B and G.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same threshold and tail-basis reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only from the displayed B, G, M, D surface.*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row consuming the same tail, criterion, threshold, dyadic, and readback route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, G, M, D, R, L have been displayed.*
8. *H records componentwise* hsame *transport for the tail-basis, convergence-criterion, threshold, dyadic-ledger, readback, limit-seal, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each tightness request through $B, G, M, D, R, L, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyTightness}^{\uparrow}$, $\mathsf{CauchyTailBasis}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTightness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, G, M, D, R, L, E, H, C, P, N$. The carrier contains no quotient stream equality, selected limit, countable-choice selector, host real equality row, trichotomy row, open-cover compactness row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, Lean verification credential, or standard bridge.*

Lean4 ✓ `BEDC.Derived.CauchyTightnessUp.CauchyTightnessTasteGate single carrier alignment`

Theorem 1870.2 (Cauchy tightness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTightness}^{\uparrow}$ *carrier $T = (B, G, M, D, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTightness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed tail-basis row, convergence-criterion row, threshold row, dyadic tolerance ledger, regular rational readback, Cauchy limit seal, Real seal, transport, replay, provenance, and local naming row;*
2. *criterion-to-tail compatibility, requiring every convergence-criterion request in G to be read only through the admitted tail-basis row B;*
3. *tail-basis handoff, requiring each tightness read to factor through B and G before any threshold, dyadic ledger, regular readback, limit seal, or Real seal is exposed;*
4. *readback exactness, requiring R to be obtained from the same B, G, M, D surface and from no second stream representative;*
5. *limit-seal compatibility, requiring L to consume exactly the displayed tail, criterion, threshold, dyadic, and readback rows;*
6. *Real-seal nonescape, requiring E to remain downstream of B, G, M, D, R, L;*
7. *ledger nonescape, requiring H, C, P, N to transport, replay, source, and name no quotient stream, selected limit, countable-choice selector, host real equality row, trichotomy row, open-cover compactness row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1870.1. The accepted packet displays exactly the tail-basis, convergence-criterion, threshold, dyadic-ledger, regular-readback, Cauchy-limit-seal, Real-seal, and structural rows.

Criterion-to-tail compatibility follows from the placement of G after B. A convergence-criterion request can be replayed only against the admitted tail-basis row already displayed by the carrier.

The handoff and readback clauses are then route projections. The tightness request reads B and G first, then the threshold row M and dyadic ledger D, and only then the regular rational readback R.

Limit-seal compatibility and Real-seal nonescape are downstream projections. The rows L and E have no independent source; each consumes the same B, G, M, D, R surface.

Finally, H, C, P, N preserve, replay, source, and name this displayed packet. Since the accepted carrier has no coordinate outside $B, G, M, D, R, L, E, H, C, P, N$, the listed obligation rows exhaust the local NameCert surface. □

Theorem 1870.3 (Cauchy tightness tail-basis handoff)**.** *Every accepted* $\mathsf{CauchyTightness}^{\uparrow}$ *carrier* $T = (B, G, M, D, R, L, E, H, C, P, N)$ *factors each tightness read through the finite route*

$$B \longmapsto G \longmapsto M \longmapsto D \longmapsto R \longmapsto L \longmapsto E$$

before any Real-facing consumer can use the terminal seal. Thus a tightness read first passes through $\mathsf{CauchyTailBasis}^{\uparrow}$ *and* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$*, then through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, without detaching a quotient stream, selected limit, countable-choice selector, host real equality comparison, trichotomy decision, open-cover compactness object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $T = (B, G, M, D, R, L, E, H, C, P, N)$. Projection gives the tail-basis row B as the first tightness surface.

The convergence row G is read next. It supplies the criterion-facing interpretation of the same tail requests, so no threshold or dyadic ledger can be inspected before the tail-basis and criterion rows are fixed.

The route then exposes M, D, R, L, E in order. The threshold row M and dyadic ledger D support the regular readback R, the Cauchy limit seal L consumes that same readback route, and the Real seal E is downstream of L.

The structural rows H, C, P, N do not change the order. They transport, replay, source, and name the displayed chain, so every tightness read factors through the stated route and no excluded coordinate is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTightness}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTightness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1870.1, Definition 1870.2, and Definition 1870.3 seed the finite tight-tail route through $\mathsf{CauchyTailBasis}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTightness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-tail compatibility, tail-basis handoff, readback exactness, CauchyLimitSeal compatibility, Real-seal nonescape, and ledger nonescape for BEDC.Derived.CauchyTightnessUp.

Constructive story: A $\mathsf{CauchyTightness}^{\uparrow}$ packet is generated from a CauchyTailBasis row, a CauchyConvergence-Criterion row, a CauchyModulus or finite threshold row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a CauchyLimitSeal compatibility row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1871

A Concrete Naming Certificate for $\mathsf{DirichletApproximation}^{\uparrow}$

Following Hardy–Wright's presentation of Dirichlet rational approximation, the $\mathsf{DirichletApproximation}^{\uparrow}$ packet names the finite denominator-bounded route that sends a real-facing rational request to a displayed approximant before any host real equality or best-approximant selector is visible. It sits over the rational-tree source of chapter 1704, the interval boundary of chapter 354, the dyadic tolerance ledger of chapter 291, the finite rational-readback route of chapter 1668, and the terminal seal of chapter 38. The packet also consumes $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows so the approximation certificate remains a finite NameCert surface rather than an ambient theorem about completed reals.

Definition 1871.1 (Dirichlet approximation carrier)**.** *A* $\mathsf{DirichletApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, B, M, D, S, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational source row. It records the* $\mathsf{Rat}^{\uparrow}$ *input rationals and interval endpoints that the approximation request may inspect.*
2. *B is the denominator-bound row. It stores the positive finite bound, the visible denominator budget, and the comparison rows used to certify that the exported denominator stays within the bound.*
3. *M is the Stern–Brocot or Farey mediant route. It records the finite branch sequence, mediant bracketing data, and rational candidate selected by displayed tree search rather than by an external choice principle.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the requested dyadic error, the slack ledger, and the endpoint comparisons used to validate the candidate against the displayed rational interval.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite real-name windows that may be read by the approximation request.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It reads the windows in S as regular rational data before any real-facing handoff is exposed.*
7. *R is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff row. It packages the rational candidate, denominator bound, dyadic tolerance, StreamName window, and RegSeqRat readback as the finite approximation surface consumed downstream.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It can be read only after Q, B, M, D, S, G, R have been displayed.*
9. *H records componentwise* hsame *transport for the rational source, denominator bound, mediant route, dyadic tolerance, StreamName window, RegSeqRat readback, real-rational handoff, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Dirichlet approximation request through $Q, B, M, D, S, G, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{DirichletApproximation}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite denominator-bounded rational-approximation packet. It has no coordinate for host real equality, quotient representatives, choice-selected best approximants, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1871.2 (Dirichlet approximation NameCert obligations)**.** *For every accepted* $\mathsf{DirichletApproximation}^{\uparrow}$ *carrier $A = (Q, B, M, D, S, G, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirichletApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational source, denominator bound, Stern–Brocot or Farey mediant route, dyadic tolerance, StreamName window, RegSeqRat readback, RealRationalApproximation handoff, RealUp seal, transport, replay, provenance, and local naming rows;*
2. *rational-source admission, requiring every exported approximant and interval endpoint to be read from the displayed row Q;*
3. *denominator-bound admission, requiring the denominator comparison to pass through the finite row B before the candidate can be exported;*
4. *mediant-route admission, requiring the rational candidate to be produced by the displayed Stern–Brocot or Farey route M;*
5. *dyadic-tolerance admission, requiring the approximation error and slack comparisons to be read from D;*
6. *window-readback admission, requiring S and G to be fixed before the real-rational handoff R is visible;*
7. *Real-seal admission, requiring E to remain terminal and to expose no host real equality, quotient representative, best-approximant selector, trichotomy branch, or ambient completeness theorem;*
8. *transport, replay, provenance, and naming exhaustion, requiring H, C, P, N to preserve and source exactly the displayed packet.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DirichletApproximationUp.DirichletApproximationTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1871.1. The accepted packet displays exactly the rational source, denominator bound, mediant route, dyadic tolerance, StreamName window, RegSeqRat readback, real-rational handoff, terminal Real seal, and structural rows.

The rational-source and denominator-bound obligations are forced by row order. The candidate and interval endpoints first read Q, while the finite denominator comparison is stored in B before any approximation handoff can be exported.

The mediant and tolerance obligations use the next two rows. The row M supplies the Stern–Brocot or Farey path that names the rational candidate, and D supplies the dyadic error, slack, and comparison ledger used to validate it.

The window-readback obligation is finite. The row S exposes only displayed StreamName windows, and G reads those windows as regular rational data before the real-rational approximation row R is available.

The terminal seal and structural rows preserve the same boundary. The row E seals the route as a $\mathsf{Real}^{\uparrow}$-facing packet, while H, C, P, N transport, replay, source, and name the displayed data without adding host real equality, quotient representatives, choice-selected best approximants, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Theorem 1871.3 (Dirichlet approximation finite handoff)**.** *Every* $\mathsf{RealRationalApproximation}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{DirichletApproximation}^{\uparrow}$ *carrier* $A = (Q, B, M, D, S, G, R, E, H, C, P, N)$ *factors through the finite route*

$$Q, B, M, D, S, G \longmapsto R, E$$

before transport, replay, provenance, or local naming can be read. Consequently the exported approximation is witnessed by a rational source row, a denominator-bound row, a displayed mediant route, a dyadic tolerance ledger, StreamName windows, and RegSeqRat readback, not by host real equality or by choosing a best representative.

Proof. Begin with an exported real-rational approximation or Real-facing read. The carrier has only the handoff row R and the terminal seal row E for such reads, and both occur after Q, B, M, D, S, G.

The source and bound rows fix the rational data. The row Q supplies the visible rational source and interval endpoints, while B records the finite denominator budget and its comparison evidence.

The candidate route is then displayed. The row M names the Stern–Brocot or Farey mediant path, and D records the dyadic tolerance and slack comparisons that certify the approximation request.

The real-facing input is read through windows. The row S supplies the finite StreamName window, and G converts that window into RegSeqRat rational readback before R may package the handoff.

Only after these rows are present can R and then E be read. Transport, replay, provenance, and local naming preserve this route, so the exported approximation cannot depend on host real equality, quotient representatives, choice-selected best approximants, trichotomy, or ambient Real completeness. □

Theorem 1871.4 (Dirichlet approximation denominator window)**.** *Every downstream approximation consumer of an accepted* $\mathsf{DirichletApproximation}^{\uparrow}$ *carrier*

$$A = (Q, B, M, D, S, G, R, E, H, C, P, N)$$

that uses a denominator-bound read must expose the displayed rows in the order

$$Q \longmapsto B \longmapsto M \longmapsto D, S, G, R, E.$$

In particular, a $\mathsf{KroneckerApproximation}^{\uparrow}$ *consumer using the Dirichlet handoff row* D *and a* RationalBestApproximat *consumer using the denominator-budget surface must first read the rational source row* Q*, then the*

finite denominator-bound row B, and then the Stern–Brocot or Farey route M before consuming dyadic tolerance, StreamName, RegSeqRat, RealRationalApproximation, or terminal Real rows.

Proof. Project the carrier in Definition 1871.1. The rational source row Q is the only source for exported approximants and interval endpoints, while the row B is the only displayed denominator budget.

Apply Definition 1871.3. Its handoff route places the mediant row M before the dyadic tolerance row D, the StreamName window row S, the RegSeqRat readback row G, and the real-facing rows R, E.

Thus any consumer that needs a denominator-bound read must pass through Q, B, M before it can consume the remaining approximation rows. This covers the $\mathsf{KroneckerApproximation}^{\uparrow}$ handoff that cites the Dirichlet row D and the $\mathsf{RationalBestApproximation}^{\uparrow}$ handoff that reuses the same denominator-budget surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1871.1, Definition 1871.2, and Definition 1871.3 seed the finite denominator-bounded rational-approximation route through rational source, mediant, dyadic tolerance, StreamName, RegSeqRat, RealRationalApproximation, and terminal Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient representatives, choice-selected best approximants, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-source admission, denominator-bound admission, mediant-route admission, dyadic-tolerance admission, window-readback admission, finite handoff factorization, transport stability, and non-escape proofs for BEDC.Derived.DirichletApproximationUp.

Constructive story: A $\mathsf{DirichletApproximation}^{\uparrow}$ packet is generated from rational source data, a denominator-bound row, a Stern–Brocot or Farey mediant route, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealRationalApproximation handoff, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1872

Countable choice boundary NameCert construction

Following Bishop-style constructive analysis, the $\mathsf{CountableChoiceBoundary}^{\uparrow}$ packet separates finite witness choice from countable choice in the Real and Completion corridor. It sits after the finite stream-window surface of chapter 126, the regular rational readback discipline of chapter 301, the terminal Real seal of chapter 38, the no-choice diagonal boundary of chapter 1173, and the deterministic relation readback of chapter 1739.

The packet records the point at which finite windows may carry displayed witnesses while no global selector, host sequence choice, selected limit, quotient stream equality, or ambient completion principle is admitted. It is a human-origin boundary object because the distinction between finite witness extraction and countable choice is part of constructive analysis rather than a BEDC-native discovery.

Definition 1872.1 (Countable choice boundary carrier)**.** *A* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, W, R, E, F, L, U, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row naming the observation window and tolerance demand being served.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by Q.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, W, R have been displayed.*
5. *F is the finite witness row. It records witnesses for the displayed window and no data outside that window.*
6. *L is the local no-selected-limit row. It refuses a global selector, host sequence choice, countable choice, quotient stream equality, and ambient Real-completion evidence.*
7. *U is the deterministic unique-readback row, inherited from* $\mathsf{UniqueChoicePrinciple}^{\uparrow}$ *only for already displayed finite relation witnesses.*
8. *H records componentwise* hsame *transport for request, window, readback, seal, finite witnesses, refusal, unique readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from Q through $W, R, E, F, L, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CountableChoiceBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyChoiceBoundary}^{\uparrow}$, $\mathsf{UniqueChoicePrinciple}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableChoiceBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a global selector, a host sequence-choice function, a countable-choice principle, quotient stream equality, selected limits, ambient completion, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.CountableChoiceBoundaryUp.TasteGate.CountableChoiceBoundaryUp`
Lean4 ✓ `BEDC.Derived.CountableChoiceBoundaryUp.TasteGate.CountableChoiceBoundaryTasteGate`
`single carrier alignment`

Theorem 1872.2 (Countable choice boundary finite witness row)**.** *Let $B = (Q, W, R, E, F, L, U, H, C, P, N)$ be an accepted* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *carrier. Every witness exported by B factors through the displayed finite route*

$$Q \longmapsto W \longmapsto R \longmapsto F.$$

Thus finite $\mathsf{StreamName}^{\uparrow}$ *windows may be witnessed, and their* $\mathsf{RegSeqRat}^{\uparrow}$ *readback may be inspected, only inside the finite request already named by Q.*

Proof. Project the accepted carrier of Definition 1872.1. The only witness coordinate is F, and it appears after the finite request, stream window, and regular rational readback rows.

The BHist shape keeps the witness inside the displayed finite packet. Componentwise hsame transport in H moves the same rows together, while the Cont replay row C follows the ordered route from Q through W, R, F.

The provenance row P cites $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ as finite-window sources, not as a host family of all choices. Hence the packet permits witnessed finite windows while preserving the refusal of any global selector or host sequence-choice function. □

Theorem 1872.3 (Countable choice boundary no selected limit)**.** *No accepted* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *carrier exports a selected limit, a global stream selector, host sequence choice, countable choice, quotient stream equality, or ambient completion datum. The only Real-facing data exported by*

$$B = (Q, W, R, E, F, L, U, H, C, P, N)$$

are the displayed finite request, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, finite witness row, refusal row, deterministic finite readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The refusal row L explicitly excludes selected limits, global stream selectors, host sequence choice, countable choice, quotient stream equality, and ambient completion data, and the tuple has no additional coordinate from which such data could be reconstructed.

The seal row E is terminal for the displayed finite route. A consumer reaches it only after reading Q, W, R, so the $\mathsf{Real}^{\uparrow}$ face is attached to the visible finite packet rather than to a selected host limit.

The unique-readback row U does not add a choice principle. By the role inherited from $\mathsf{UniqueChoicePrinciple}^{\uparrow}$, it compares already displayed finite relation witnesses and therefore cannot choose a sequence of witnesses over all requests.

Transport, replay, and provenance preserve the same boundary. The rows H, C, P, N move, replay, source, and name only the finite packet, while the no-choice sibling $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ blocks the diagonal route from being read as a global selector. □

Theorem 1872.4 (Countable choice boundary Cauchy witness refusal)**.** *Any* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *consumer of chapter 3579 that uses finite* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows may cite* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *only as a refusal boundary. It may read the displayed finite witness rows supplied by Definition 1872.2, but Definition 1872.3 gives it no global selector, no host sequence choice, no selected limit, and no quotient stream equality.*

Proof. Project the accepted carrier of Definition 1872.1. The finite witness row F is downstream of the request, stream window, and regular rational readback rows, so a completion witness consumer can inspect only the displayed rows already named by the packet.

Apply Definition 1872.2. It permits finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback inside the named request, and it supplies no coordinate for a host family of choices.

Apply Definition 1872.3. The refusal row L excludes selected limits, global selectors, host sequence choice, countable choice, quotient stream equality, and ambient completion data. The $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer therefore receives a boundary certificate for finite witness rows, not a choice principle for completing all Cauchy presentations. □

Theorem 1872.5 (Countable choice boundary Real-completion discipline)**.** *Every Real and Completion consumer of an accepted* $\mathsf{CountableChoiceBoundary}^{\uparrow}$ *carrier must pass through the finite witness row of Definition 1872.2 and the no-selected-limit boundary of Definition 1872.3. Consequently* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *arguments may use finite-window witnesses but may not import countable choice, host sequence choice, selected limits, quotient stream equality, or ambient Real-completion evidence.*

Proof. Begin with a Real and Completion consumer of B. The carrier exposes the request row Q, the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{Real}^{\uparrow}$ seal E before any structural row is consumed.

Apply Definition 1872.2. The only witness surface is the finite row F, so the consumer may use witnesses for the displayed finite window and no larger family.

Apply Definition 1872.3. The refusal row L and the sibling $\mathsf{CauchyChoiceBoundary}^{\uparrow}$ exclude selected limits, global selectors, host sequence choice, countable choice, quotient stream equality, and ambient completion data.

Finally read U, H, C, P, N. The deterministic finite readback inherited from $\mathsf{UniqueChoicePrinciple}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming rows preserve the same finite route, so the Real-completion consumer remains inside the finite witness boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableChoiceBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableChoiceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1872.1, Definition 1872.2, Definition 1872.3, and Definition 1872.5 seed the finite witness boundary separating displayed window witnesses from countable choice in the Real and Completion corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableChoiceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected limits, host sequence choice, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite witness row, no-selected-limit boundary, and Real-completion discipline proofs for BEDC.Derived.CountableChoiceBoundaryUp.

Constructive story: A $\mathsf{CountableChoiceBoundary}^{\uparrow}$ packet is generated from one finite request row, one StreamName window, one RegSeqRat readback, one RealUp seal, one finite witness row, one refusal row, deterministic finite readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1873

Cauchy Diagonal Argument NameCert Construction

Following Bishop-style constructive diagonal arguments for regular Cauchy tails, the $\mathsf{CauchyDiagonalArgument}^{\uparrow}$ packet names the finite route by which a row of regular Cauchy rational names supplies one displayed diagonal selection under a cofinal tail budget. It sits downstream of the regular extraction surface of chapter 1773, uses the tail-basis route of chapter 1868, reuses the diagonal-limit seal of chapter 1800, and keeps the endpoint comparison boundary aligned with chapter 1874.

The packet is a selection certificate, not a completion theorem. It records rows, budgets, ledgers, and local naming for constructive diagonal passage through regular Cauchy tails while excluding quotient stream equality, host selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge credentials.

Definition 1873.1 (Cauchy diagonal argument carrier)**.** *A* $\mathsf{CauchyDiagonalArgument}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, S, T, B, L, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular-Cauchy row family, listing the finitely inspected* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows and their admissible tail windows.*
2. *S is the diagonal index-selection row, recording the selected index for each visible precision request without importing a countable-choice schedule.*
3. *T is the cofinal tail-budget row, displaying the tail threshold that simultaneously supports the selected row, the diagonal window, and every downstream readback.*
4. *B is the* $\mathsf{CauchyTailBasis}^{\uparrow}$ *basis row used to compare the selected tail with sibling tail requests.*
5. *L is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error ledger, storing dyadic radius, slack, and comparison entries for the selected row and cofinal tail.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row read from the selected tail.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row assembled from R, S, T, L, W.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after the row family, selected index, cofinal tail budget, basis comparison, dyadic ledger, window, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for the row family, index selection, tail budget, basis row, dyadic ledger, finite window, diagonal readback, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from each precision request through $R, S, T, B, L, W, Q, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyDiagonalArgument}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{CauchyTailBasis}^{\uparrow}$, $\mathsf{CauchyDiagonalLimit}^{\uparrow}$, $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalArgument}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, T, B, L, W, Q, E, H, C, P, N$. It contains no quotient stream equality, detached limit selector, countable-choice row, trichotomy row, ambient $\mathsf{Real}^{\uparrow}$ *completeness datum, Lean credential, or standard bridge.*

Lean4 (definition) `BEDC.Derived.CauchyDiagonalArgumentUp.CauchyDiagonalArgumentTasteGate single carrier alignment`

Theorem 1873.2 (Cauchy diagonal argument tail selection)**.** *Let $A = (R, S, T, B, L, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalArgument}^{\uparrow}$ *carrier. Every Real-facing precision request that enters the packet factors through one displayed diagonal index in S and one cofinal tail budget in T before it can read W, Q, or E. The selected route is*

$$R, S, T, B, L \longmapsto W \longmapsto Q \longmapsto E.$$

Thus the diagonal selection is a finite NameCert route over regular Cauchy tails, not a host-selected limit, quotient stream, countable-choice selector, or ambient completeness witness.

Proof. Begin with an accepted carrier and a Real-facing precision request. Carrier admission provides the row family R as the only source of regular Cauchy data, so the request cannot inspect a stream window or Real seal before a source row has been displayed.

The diagonal index is then read from S. This row stores the visible selected index for the request as packet data, and it has no coordinate for an external choice function or detached limit selector.

The cofinal tail budget T is read next. It is the row that binds the selected index to the tail threshold used by the basis row B and the dyadic ledger L, so every downstream window remains tied to the same finite request.

Finally W, Q, E are downstream of the selected budget. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the consumer receives the displayed diagonal passage and no quotient stream, host limit, countable-choice row, trichotomy principle, or ambient completeness datum. □

Theorem 1873.3 (Cauchy diagonal argument NameCert obligations)**.** *For every accepted* CauchyDiagonalArgumen *carrier $A = (R, S, T, B, L, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalArgument}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the regular-Cauchy row family, diagonal index-selection row, cofinal tail-budget row, CauchyTailBasis row, dyadic error ledger, finite StreamName window, RegSeqRat diagonal readback, terminal Real seal, transport, replay, provenance, and local naming rows;*

2. *index-selection exactness, requiring S to be the only row that supplies the displayed diagonal index for a visible precision request;*

3. *cofinal tail-budget exactness, requiring T and B to bind that selected index to a common tail before any readback is exposed;*
4. *dyadic error-ledger exactness, requiring L to store precisely the radius, slack, and comparison entries consumed by W and Q;*
5. *Real seal non-escape, requiring E, H, C, P, N to export only the displayed diagonal route and no quotient stream equality, detached limit selector, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Cauchy diagonal argument obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1873.1. The accepted packet displays exactly the row family, index-selection row, cofinal tail budget, basis comparison, dyadic ledger, finite window, diagonal readback, Real seal, and structural rows.

Index-selection exactness follows by inspecting S. Since no other coordinate supplies a selected diagonal index, every visible precision request uses the displayed index row.

Cofinal tail-budget exactness follows by inspecting T and B. The tail threshold is recorded before W, Q, or E is visible, and the basis row compares it with sibling tail requests through the same finite carrier.

The dyadic ledger clause is the projection to L. The ledger stores the radius, slack, and comparison entries used by the finite window W and the regular rational readback Q, so the error budget cannot be replaced by a detached representative.

Finally inspect E, H, C, P, N. These rows seal, transport, replay, source, and name the displayed route only; because the carrier has no quotient stream, selected host limit, countable-choice row, trichotomy principle, ambient completeness datum, Lean credential, or bridge credential, the listed five rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalArgument}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalArgument}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1873.1, Definition 1873.2, and Definition 1873.3 seed the constructive diagonal-selection route over regular Cauchy row families, displayed indices, cofinal tail budgets, CauchyTailBasis rows, DyadicRatCore ledgers, StreamName windows, RegSeqRat readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalArgument}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, index-selection exactness, cofinal tail-budget exactness, dyadic ledger exactness, Real seal nonescape, and route determinacy for BEDC.Derived.CauchyDiagonalArgumentUp.

Constructive story: A $\mathsf{CauchyDiagonalArgument}^{\uparrow}$ packet is generated from a regular-Cauchy row family, a displayed diagonal index-selection row, a cofinal tail-budget row, a CauchyTailBasis row, a DyadicRatCore error ledger, finite StreamName windows, a RegSeqRat diagonal readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1874

A Concrete Naming Certificate for $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ packet records the public L10 criterion by which two regular Cauchy rational names receive the same $\mathsf{Real}^{\uparrow}$ endpoint: display the regular-Cauchy difference, supply null-sequence evidence for that displayed difference, hand the evidence through the zero-distance boundary, and only then expose the terminal Real equality seal. It sits over the regular rational source rows of chapter 301, the displayed difference route of chapter 429, the null-difference certificate of chapter 1758, the zero-distance handoff of chapter 1727, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38.

The criterion is a finite public equality route. It is not a quotient of streams, a host equality proof, a selected limit, a trichotomy comparison, countable choice, ambient completeness, Lean verification, or bridge checking.

Definition 1874.1 (Cauchy difference criterion carrier)**.** *A* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations are compared.*
2. *D is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row displaying the aligned pointwise difference stream $X - Y$.*
3. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *evidence row certifying that the displayed difference stream shrinks to zero.*
4. *Q is the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *handoff row that consumes D and Z before any equality-facing seal is read.*
5. *W stores the finite* $\mathsf{StreamName}^{\uparrow}$ *windows used by the source rows, the difference route, the null-sequence evidence, and the zero-distance handoff.*
6. *T stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance requests, slack rows, and comparison budgets shared by D, Z, Q.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal, readable only after X, Y, D, Z, Q, W, T have been displayed.*

8. *H records componentwise* hsame *transport for source rows, difference rows, null-sequence rows, zero-distance rows, windows, tolerance ledgers, the equality seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the source rows through D, then through Z, then through Q, and finally into the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDifferenceCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, D, Z, Q, W, T, E, H, C, P, N$. It has no coordinate for quotient stream equality, host equality of reals, selected limits, trichotomy, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionTasteGate single carrier alignment`

Theorem 1874.2 (Cauchy difference criterion NameCert obligations)**.** *For every accepted* CauchyDifferenceCriterio *carrier $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyDifferenceCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *displayed difference row, the* $\mathsf{NullSequence}^{\uparrow}$ *evidence row, the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *handoff row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal, transport, continuation, provenance, and local naming rows;*

2. *source and difference stability, requiring accepted comparisons to preserve X, Y, D, W, T componentwise through the displayed* hsame *and* Cont *rows;*

3. *null-evidence exactness, requiring Z to consume precisely the displayed difference route D with the windows W and tolerance ledger T;*

4. *zero-distance handoff exactness, requiring Q to read the null-difference route $X, Y, D, Z, W, T, H, C, P, N$ before the equality seal E is exposed;*

5. *Real seal nonescape, requiring every equality-facing read to factor through K and forbidding quotient stream equality, host real equality, selected limits, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking from the local surface.*

No Cauchy difference criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1874.1. The accepted packet displays the two regular rational source rows, the difference row, the null-sequence evidence row, the zero-distance handoff row, finite windows, dyadic tolerances, the Real equality seal, and the structural provenance rows.

Source and difference stability are componentwise. The two $\mathsf{RegSeqRat}^{\uparrow}$ rows transport their scheduled rational observations, the finite $\mathsf{StreamName}^{\uparrow}$ windows transport the inspected reads, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger transports the requested bounds used by the displayed difference route.

Null-evidence exactness is the next route constraint. The row Z is downstream of D and can read only the displayed difference through the shared windows W and tolerance ledger T, so the evidence is a finite null-sequence witness for that route.

Zero-distance handoff exactness then follows by reading Q. The handoff row consumes the same null-difference route before E is available, and the continuation row C replays that order without adding another equality coordinate.

Finally inspect the carrier inventory. Since quotient streams, host real equality, selected limits, trichotomy, countable-choice schedules, ambient completeness, Lean verification, and bridge credentials are absent from K, none can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionCarrier namecert obligations`

Theorem 1874.3 (Cauchy difference criterion tail-diameter handoff)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. Every tail-diameter request for the difference of the two regular Cauchy source rows factors through the displayed route*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow Q$$

using the finite windows W, the dyadic tolerance ledger T, and the structural rows H, C, P, N. Thus the zero-distance handoff receives null-sequence evidence for the displayed regular-Cauchy difference itself, not for a detached representative or a quotient stream.

Proof. Start with a tail-diameter request. By carrier admission, the request can inspect the source rows only through X, Y and the finite window row W.

The displayed difference route is then forced. The row D is the only coordinate that aligns the two source rows and forms the pointwise difference stream, with every tolerance read recorded in T.

The null evidence is consumed next. The row Z reads the displayed difference stream together with the same windows and tolerance ledger, so each tail-diameter bound used downstream is evidence about D.

Finally the zero-distance row Q receives exactly that finite null-difference route. The transport, continuation, provenance, and local naming rows preserve the route under componentwise comparison and replay, but they do not introduce a detached representative, quotient stream, host equality proof, or ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionTailDiameter handoff certificate`

Theorem 1874.4 (Cauchy difference criterion Real seal boundary)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *equality-facing consumer of the two regular Cauchy source rows must first read the finite route*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow Q \longrightarrow E.$$

Consequently the two endpoint names are sealed equal only through regular-Cauchy difference, null-sequence evidence, zero-distance handoff, and the displayed Real seal; the packet does not export quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionReal seal boundary`

Proof. Begin with an equality-facing consumer. The NameCert obligations of Definition 1874.2 force the consumer to read the carrier rows of K before any endpoint seal is visible.

The first analytic route is the displayed difference. Rows X and Y supply the regular rational observations, while D records their aligned pointwise difference through W and T.

The equality evidence then passes through two finite witnesses. The null-sequence row Z certifies the displayed difference as zero-tail evidence, and Definition 1874.3 sends that same evidence into the zero-distance row Q.

Only after Q is read may the consumer reach E. The structural rows H, C, P, N transport, replay, source, and name the same route, so the seal remains downstream of the null-difference handoff.

The boundary statement follows from the carrier inventory. No quotient stream equality, host equality proof, selected limit, trichotomy comparison, countable-choice row, or ambient completeness theorem is present in K, so the $\mathsf{Real}^{\uparrow}$ equality seal is exactly the displayed finite route. □

Theorem 1874.5 (Cauchy difference criterion zero-distance correspondence)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H,$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. The packet exports a Real equality-facing seal exactly through the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row D, the* $\mathsf{NullSequence}^{\uparrow}$ *row Z, and the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *row Q:*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow Q \longrightarrow E.$$

Any consumer lacking one of D, Z, Q cannot read equality of the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows X, Y from the accepted carrier.*

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionZeroDistanceCorrespondence`

Proof. Project the carrier and isolate the equality-facing rows. The only displayed difference coordinate is D, the only null-evidence coordinate for that difference is Z, and the only zero-distance handoff coordinate before the Real seal is Q.

Apply Definition 1874.3. It forces every tail-diameter request for the two source rows through the displayed regular-Cauchy difference and null-sequence evidence before the zero-distance handoff can be read.

Apply Definition 1874.4. The Real seal E is downstream of the same D, Z, Q route, so the equality-facing read is exactly the displayed zero-distance correspondence.

If a consumer omits one of D, Z, Q, the carrier has no remaining coordinate that can connect X, Y to E. Quotient streams, host real equality, selected representatives, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the packet, so none can replace the displayed route. □

Theorem 1874.6 (Cauchy difference criterion regular-tail stability)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. Any retained tail window of X and Y that preserves the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row D and the* $\mathsf{NullSequence}^{\uparrow}$ *row Z also preserves the zero-distance handoff Q and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E under the transport and replay rows H and C. The retained packet exports no quotient stream representative, selected tail limit, host equality proof, choice schedule, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionCarrier regular tail stability`

Proof. Project the carrier. The retained window can inspect the source rows only through X, Y and the finite window row W, while the displayed difference and null rows remain D and Z by hypothesis.

Use the tail-diameter handoff of Definition 1874.3. It keeps every tail request on the same displayed regular-Cauchy difference and null-sequence evidence, so the retained window has no detached representative through which to alter the equality route.

Apply the zero-distance correspondence of Definition 1874.5. Since the retained window preserves D and Z, the only available equality-facing handoff remains Q, and the Real-seal boundary of Definition 1874.4 keeps E downstream of that same handoff.

Finally transport and replay the retained rows through H and C. The carrier inventory has no quotient stream, selected representative, choice schedule, or ambient completeness row, so the terminal $\mathsf{Real}^{\uparrow}$ seal is preserved exactly as the displayed D, Z, Q, E route. □

Theorem 1874.7 (Cauchy difference criterion bidirectional exactness)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. A* $\mathsf{Real}^{\uparrow}$ *equality-facing read of the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows X, Y is exported by K if and only if the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row D is consumed with the* $\mathsf{NullSequence}^{\uparrow}$ *evidence row Z, reflected through the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *row Q, and then sealed at the terminal* $\mathsf{Real}^{\uparrow}$ *row E:*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow Q \longrightarrow E.$$

The exactness statement exports no quotient stream equality, host real equality, selected limits, trichotomy, countable choice, or ambient completeness.

Proof. For the forward direction, start with an equality-facing read of X, Y. Apply Definition 1874.4 to force the read through the displayed route ending at E.

Then apply Definition 1874.5. It identifies the visible equality route as the same D, Z, Q chain: the displayed regular-Cauchy difference is consumed with null-sequence evidence and reflected through the zero-distance row before the Real seal is exposed.

For the reverse direction, assume the displayed D, Z, Q, E route is present. Apply Definition 1874.3 to read every tail-diameter request for the source rows through the displayed difference and null evidence.

The carrier obligations of Definition 1874.2 then repack that finite route as the local equality-facing seal of K. Since the carrier contains no quotient stream equality, host real equality, selected-limit row, trichotomy row, countable-choice schedule, or ambient-completeness principle, the reconstructed route is the same displayed route and no other equality channel is exported. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionBidirectionalExactness`

Theorem 1874.8 (Cauchy difference criterion null-route converse)**.** *Let $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier. If K exposes the* $\mathsf{Real}^{\uparrow}$ *equality-facing seal for the source rows X, Y, then the seal route includes the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row D, the* $\mathsf{NullSequence}^{\uparrow}$ *evidence row Z, and the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *handoff row Q. Conversely, any packet with the same visible source rows but no Z row for the displayed difference cannot export equality of X, Y through the accepted carrier.*

Proof. For the forward direction, start with a $\mathsf{Real}^{\uparrow}$ equality-facing read of X, Y. Apply Definition 1874.7. The read is exported exactly through the displayed route D, Z, Q, E.

The same route is forced by Definition 1874.5. It identifies D as the displayed regular-Cauchy difference row, Z as the null-sequence evidence row for that displayed difference, and Q as the zero-distance handoff row before the terminal seal.

For the converse, remove the Z evidence row while keeping the same visible sources. By Definition 1874.4, the Real seal is downstream of the finite route through D, Z, Q; without Z, the route from the displayed difference to the zero-distance handoff cannot be completed.

The carrier has no replacement equality channel. Quotient stream equality, host equality, selected limits, trichotomy, countable choice, and ambient completeness are all excluded from Definition 1874.1, so a packet missing Z cannot export equality of X, Y through the accepted carrier. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionNullRouteConverse`

Theorem 1874.9 (Cauchy difference criterion shift invariance)**.** *Let*

$$K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$$

and

$$K' = (X', Y', D', Z', Q', W', T', E', H', C', P', N')$$

be accepted $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carriers. Suppose the two packets share the same source, displayed-difference, null-evidence, zero-distance, and Real-seal rows up to a finite* $\mathsf{StreamName}^{\uparrow}$ *tail shift, and suppose this shift preserves the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the displayed difference and null route. Then the equality-facing route*

$$X, Y \longrightarrow D \longrightarrow Z \longrightarrow Q \longrightarrow E$$

is the same accepted route as the shifted route

$$X', Y' \longrightarrow D' \longrightarrow Z' \longrightarrow Q' \longrightarrow E'.$$

Finite prefix deletion changes only the window rows W, W' and their replay positions; it does not change the displayed null-difference route consumed by the Real seal.

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterion shift invariance`

Proof. Project the two accepted carriers. The shared data are the source rows, displayed difference rows, null-sequence evidence rows, zero-distance handoff rows, dyadic tolerance ledgers, and terminal Real-seal rows, with only the finite StreamName windows shifted.

Use the tail-diameter handoff of Definition 1874.3. Since the shift preserves the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, each tail-diameter request for the shifted packet is still read through the displayed difference and null-evidence rows rather than through a detached representative.

Apply the Real-seal boundary and bidirectional exactness, Definition 1874.4 and Definition 1874.7. They force both packets to expose equality only through their D, Z, Q, E chains.

Finally use the null-route converse, Definition 1874.8. Any equality-facing seal must include the same displayed null route, so deleting a finite prefix can affect only the windows W, W' and their continuation replay. The accepted equality-facing route remains invariant under the tail shift. □

Theorem 1874.10 (Cauchy difference criterion kernel scope)**.** *Every accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *packet reads the difference criterion only through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *pointwise naming rows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-tail rows of chapter 301, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary of chapter 38 before exporting a null-route or zero-distance conclusion.*

Proof. Project an accepted carrier $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$. The source observations are the $\mathsf{RegSeqRat}^{\uparrow}$ rows X, Y, and their inspected pointwise names are visible only through the finite $\mathsf{StreamName}^{\uparrow}$ windows W.

The displayed difference and null route use the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger T. Thus every bound consumed by D, Z, and Q is a tolerance-row read rather than a host metric or selected tail witness.

The terminal equality-facing boundary is the $\mathsf{Real}^{\uparrow}$ seal E. By the Real-seal boundary and zero-distance correspondence, Definition 1874.4 and Definition 1874.5, any exported conclusion passes through D, Z, Q, E.

Consequently the packet's kernel scope is exactly the Real, RegSeqRat, DyadicRatCore, and StreamName dependency route. The structural rows H, C, P, N transport, replay, source, and name that route, but they do not add a quotient stream, host equality proof, selected limit, choice schedule, ambient completeness endpoint. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionKernelScope`
↪ variant `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionCarrier kernel scope`

Theorem 1874.11 (Cauchy difference criterion obligation export)**.** *The carrier definition for* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$*, together with its NameCert obligations, tail-diameter handoff, Real-seal boundary, zero-distance correspondence, regular-tail stability, bidirectional exactness, and null-route converse, assembles the obligation-facing L10 certificate over* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyDifference}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*,* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

The exported certificate has no coordinate for quotient streams, host equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionObligationExport`

Proof. Start with the carrier definition Definition 1874.1 and the NameCert obligations of Definition 1874.2. These two rows admit exactly the source $\mathsf{RegSeqRat}^{\uparrow}$ names, displayed difference row, null-sequence evidence, zero-distance handoff, finite window, dyadic tolerance, terminal Real seal, transport, replay, provenance, and local naming data.

Next apply the tail-diameter handoff Definition 1874.3 and the Real-seal boundary Definition 1874.4. They place every equality-facing read behind the finite route through $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Finally use the zero-distance correspondence, regular-tail stability, bidirectional exactness, and null-route converse, namely Definition 1874.5, Definition 1874.6, Definition 1874.7, and Definition 1874.8. These results repack the same finite rows as an obligation-facing certificate and show that no quotient stream, host equality, selected limit, trichotomy, choice schedule, ambient Real-completeness theorem, Lean verification, or bridge check is used. □

Theorem 1874.12 (Cauchy difference criterion scope package)**.** *For every accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier* $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$*, the public null-route and zero-distance conclusions of* K *are scoped exactly by the displayed rows*

$$X, Y, D, Z, Q, W, T, E, H, C, P, N.$$

Here X, Y *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows,* D *is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row,* Z *is the* $\mathsf{NullSequence}^{\uparrow}$ *evidence,* Q *is the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *handoff,* W *records* $\mathsf{StreamName}^{\uparrow}$ *windows,* T *records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* H, C, P, N *are the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. No null-route or zero-distance conclusion for* K *is available outside this finite kernel scope.*

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionScopePackage`

Proof. Project the accepted carrier. The kernel-scope theorem Definition 1874.10 already fixes the analytic reads to the $\mathsf{RegSeqRat}^{\uparrow}$ sources, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and $\mathsf{Real}^{\uparrow}$ seal boundary before any null-route or zero-distance conclusion is exported.

Next apply the obligation export theorem Definition 1874.11. It repacks the same carrier admission, NameCert obligations, tail-diameter handoff, Real-seal boundary, zero-distance correspondence, regular-tail stability, bidirectional exactness, and null-route converse as one L10 certificate over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

It remains to rule out escape from the displayed rows. The carrier definition has no quotient stream, host equality, selected limit, trichotomy, countable-choice schedule, ambient completeness, Lean-verification, or bridge-checking coordinate. The structural rows H, C, P, N only transport, replay, source, and name the displayed packet, so every public conclusion remains scoped by $X, Y, D, Z, Q, W, T, E, H, C, P, N$. □

Theorem 1874.13 (Cauchy difference criterion scoped kernel binding)**.** *For every accepted* CauchyDifferenceCriteric *carrier* $K = (X, Y, D, Z, Q, W, T, E, H, C, P, N)$*, the public null-route and zero-distance conclusions of* K *bind exactly to the scoped kernel route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyDifference}^{\uparrow},\ \mathsf{NullSequence}^{\uparrow},\ \mathsf{RegularCauchyZeroDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}$$

The source rows are only X, Y *as* $\mathsf{RegSeqRat}^{\uparrow}$ *rows, the displayed difference is only* D*, the null evidence is only* Z*, the zero-distance handoff is only* Q*, the finite windows are only* W*, the tolerance reads are only* T*, the equality-facing seal is only* E*, and the structural binding rows are only* H, C, P, N*. Thus no public null-route or zero-distance conclusion escapes the scoped rows of* K*.*

Proof. Project the carrier and apply Definition 1874.10. It fixes the analytic source of every public conclusion to the $\mathsf{RegSeqRat}^{\uparrow}$ rows, displayed $\mathsf{RegularCauchyDifference}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal boundary.

Next use Definition 1874.11. The obligation package supplies the $\mathsf{NullSequence}^{\uparrow}$ evidence row, the $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ handoff row, and the structural BHist, hsame, Cont, Pkg, and NameCert rows as part of the same finite L10 certificate.

Finally apply Definition 1874.12. It identifies the public null-route and zero-distance conclusions with exactly $X, Y, D, Z, Q, W, T, E, H, C, P, N$. Since the carrier has no quotient stream, host equality, selected-limit, trichotomy, countable-choice, ambient-completeness, Lean-verification, or bridge-checking coordinate, the scoped kernel binding is exhaustive. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionScopedKernelBinding`

Theorem 1874.14 (Cauchy difference criterion public export)**.** *Every accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier exports the public* $\mathsf{Real}^{\uparrow}$ *equality route exactly through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyDifference}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*,* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, together with the structural* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

The export has no coordinate for quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tail-diameter handoff of Definition 1874.3. It forces every equality-facing read to pass through the two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, the displayed $\mathsf{RegularCauchyDifference}^{\uparrow}$ row, the $\mathsf{NullSequence}^{\uparrow}$ evidence, finite $\mathsf{StreamName}^{\uparrow}$ windows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger.

Next apply the Real-seal boundary Definition 1874.4. The public route reaches $\mathsf{Real}^{\uparrow}$ only through the terminal seal row already carried by the accepted packet; no detached quotient stream or selected limit is introduced at the boundary.

Use the zero-distance correspondence and bidirectional exactness of Definition 1874.5 and Definition 1874.7. These identify the displayed null route with the $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ handoff in both directions over the same finite rows.

Finally apply the scoped kernel binding of Definition 1874.13. It repacks the route with the structural BHist, hsame, Cont, Pkg, and NameCert rows, and its nonescape clauses exclude trichotomy, countable choice, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the public export. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionPublicExport`

Theorem 1874.15 (Cauchy difference criterion public exactness)**.** *Every public* $\mathsf{Real}^{\uparrow}$ *equality consumer of an accepted* $\mathsf{CauchyDifferenceCriterion}^{\uparrow}$ *carrier receives exactly the following two routes. First, the displayed* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *row with* $\mathsf{NullSequence}^{\uparrow}$ *evidence implies the* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ *handoff and the terminal zero-distance* $\mathsf{Real}^{\uparrow}$ *seal. Second, any zero-distance* $\mathsf{Real}^{\uparrow}$ *seal exported by the carrier reconstructs the same displayed null-difference route through* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *and* $\mathsf{NullSequence}^{\uparrow}$*.*

Both directions pass only through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{RegularCauchyDifference}^{\uparrow}$*,* $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The public exactness surface has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the forward route, start with the displayed $\mathsf{RegularCauchyDifference}^{\uparrow}$ row and $\mathsf{NullSequence}^{\uparrow}$ evidence. Apply the bidirectional exactness theorem Definition 1874.7; it routes the null difference through the $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ row and then through the terminal $\mathsf{Real}^{\uparrow}$ seal. The public export theorem Definition 1874.14 keeps this route on the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and structural rows.

For the reverse route, start with a zero-distance $\mathsf{Real}^{\uparrow}$ seal exported by the accepted carrier. The null-route converse Definition 1874.8 reconstructs the displayed $\mathsf{RegularCauchyDifference}^{\uparrow}$ and $\mathsf{NullSequence}^{\uparrow}$ rows behind that seal. The zero-distance correspondence Definition 1874.5 identifies the reconstructed route with the same $\mathsf{RegularCauchyZeroDistance}^{\uparrow}$ handoff.

The route is stable under finite window changes by Definition 1874.9. Thus the exactness claim does not depend on a selected tail representative or quotient stream equality; finite $\mathsf{StreamName}^{\uparrow}$ shifts preserve the displayed D, Z, Q, E chain.

Finally apply the scoped kernel binding Definition 1874.13. It binds both directions to the finite kernel rows $X, Y, D, Z, Q, W, T, E, H, C, P, N$, so the public equality consumer receives exactly the two displayed routes and no host equality, selected limit, trichotomy, countable-choice schedule, or ambient-completeness witness. □

Lean4 ✓ `BEDC.Derived.CauchyDifferenceCriterionUp.CauchyDifferenceCriterionPublicExactness`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDifferenceCriterion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDifferenceCriterion}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 1874.1, Definition 1874.2, Definition 1874.12, and Definition 1874.13 bind the public L10 criterion from two RegSeqRat source rows through RegularCauchyDifference, NullSequence evidence, RegularCauchyZeroDistance handoff, StreamName windows, DyadicRatCore tolerances, a terminal Real equality seal, and the structural BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDifferenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridged closure requires the standard bridge or bridge-facing theorem for the public exactness route.

Constructive story: A CauchyDifferenceCriterion$^{\uparrow}$ packet is generated from two RegSeqRat source rows, one RegularCauchyDifference route, one NullSequence evidence row, one RegularCauchyZeroDistance handoff row, finite StreamName windows, DyadicRatCore tolerances, a terminal RealUp equality-seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1875

A Concrete Naming Certificate for $\mathsf{GramSchmidt}^{\uparrow}$

The classical Gram-Schmidt orthogonalization procedure is a finite-dimensional linear algebra construction that turns an ordered list of vectors into an orthogonal list by subtracting previously displayed projection components. The BEDC $\mathsf{GramSchmidt}^{\uparrow}$ packet records only this finite constructive surface: it is grounded in the vector carrier of chapter 47, the inner-product form rows of chapter 86, the norm-square boundary of chapter 89, and the finite row ledger of chapter 49.

The point of the packet is not to choose a basis. It exposes the ordered replay of the orthogonalization calculation, and when a pivot is not known to be nonzero it records a refusal row rather than selecting a replacement vector or invoking host choice.

Definition 1875.1 (Gram-Schmidt carrier)**.** *A* $\mathsf{GramSchmidt}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, O, Z, A, R, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite source-vector row, listing the ordered* $\mathsf{VecSpace}^{\uparrow}$ *vectors to be inspected.*
2. *O is the orthogonalized output row, listing the displayed residual vectors produced from the earlier rows of S.*
3. *Z is the pivot boundary row. For each displayed residual it records either a nonzero norm-square witness or a refusal row saying that the packet does not promote that residual to a selected basis vector.*
4. *A is the scalar projection-coefficient ledger, with each coefficient read from the* $\mathsf{InnerProduct}^{\uparrow}$ *form row and the relevant norm-square pivot row.*
5. *R is the norm-squared row for all displayed residuals and pivot comparisons, sourced through* $\mathsf{Norm}^{\uparrow}$ *and the inner-product diagonal.*
6. *T is the triangular replay row, recording that the ith output row is obtained from the ith source vector by subtracting only projection components along earlier displayed output rows.*
7. *H records componentwise* hsame *transport for source vectors, output rows, pivot witnesses or refusals, projection coefficients, norm-squared rows, triangular replay, continuation, provenance, and local naming.*

8. *C records displayed* Cont *replay from the source row through projection coefficients, residual formation, norm-square inspection, pivot boundary, and local certificate.*
9. *P is the* Pkg *provenance over* $\mathsf{GramSchmidt}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a host basis, dimension theorem, spectral theorem, Hilbert completion, quotient equality, or selected replacement pivot.

Lean4 ✓ `BEDC.Derived.GramSchmidtUp.GramSchmidtTasteGate single carrier alignment`

Theorem 1875.2 (Gram-Schmidt NameCert obligation surface)**.** *For every accepted* $\mathsf{GramSchmidt}^{\uparrow}$ *carrier* $G = (S, O, Z, A, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}$ *surface has exactly five finite rows:*

1. *carrier exposure for the source-vector list, orthogonalized output rows, pivot boundary, projection coefficients, norm-squared rows, triangular replay, transport, continuation, provenance, and local naming;*
2. *inner-product projection coefficient exposure, requiring every scalar in A to be read from displayed* $\mathsf{InnerProduct}^{\uparrow}$ *form rows together with the relevant nonzero pivot witness in Z;*
3. *orthogonality route, requiring every distinct displayed output pair to be certified by the triangular replay row T and the projection-coefficient ledger A;*
4. *nonzero-pivot and refusal boundary, requiring each residual row to carry either the nonzero norm-square witness needed for projection division or the explicit refusal that the row is not promoted to a selected basis vector;*
5. *transport, provenance, and local naming, requiring componentwise* hsame *transport,* Cont *replay,* Pkg *source rows, and the local* $\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}$ *row to preserve exactly the displayed packet.*

No Gram-Schmidt obligation is stored outside these rows.

Proof. Proceed by finite induction on the displayed vector list S. The empty list exposes the empty source row, empty output row, empty coefficient ledger, empty triangular replay, and the local naming row.

For the step from a displayed prefix to the next source vector, use the induction data to read the earlier output rows and their pivot boundary rows. The projection coefficient for each earlier nonzero pivot is read from the displayed $\mathsf{InnerProduct}^{\uparrow}$ form value and the corresponding norm-square row, and the residual is recorded in O by triangular replay through T.

Inspect the new residual through its norm-squared row R. If the packet carries a nonzero pivot witness, the residual becomes the next displayed output row available to later projection coefficients. If such a witness is not displayed, the row is recorded in Z as a refusal, so the construction keeps the finite source vector without choosing a replacement or importing a host basis.

Orthogonality is the invariant preserved by the same finite replay. Earlier output rows are orthogonal by the induction hypothesis, and the new residual subtracts exactly the displayed projection components against those rows. Componentwise hsame transport in H, Cont replay in C, Pkg provenance in P, and local naming in N then move only the rows listed in the carrier. □

Theorem 1875.3 (Gram-Schmidt projection orthogonality induction)**.** *Let $G = (S, O, Z, A, R, T, H, C, P, N)$ be an accepted* $\mathsf{GramSchmidt}^{\uparrow}$ *carrier. Finite prefix induction on the source row S preserves the following invariant: every promoted output row in O is orthogonal, through the displayed* $\mathsf{InnerProduct}^{\uparrow}$ *form row, to every earlier promoted output row; every projection coefficient used in A is read only from that* $\mathsf{InnerProduct}^{\uparrow}$ *form row together with the relevant nonzero pivot witness in Z; and any residual whose pivot witness is absent is recorded by Z as a refusal row and is not promoted into O. The invariant is transported only through the rows exposed in Definition 1875.1 and the obligation surface of Definition 1875.2.*

Proof. Induct on the finite prefix of S. The empty prefix has no promoted output rows, no projection coefficients, and no pivot to inspect, so the invariant is the empty route through $O, Z, A, R, T, H, C, P, N$.

For the step, assume the invariant for the earlier prefix. The triangular replay row T forms the next residual by subtracting only projection components along earlier promoted output rows. Each coefficient in that subtraction is read from the displayed $\mathsf{InnerProduct}^{\uparrow}$ form value and the corresponding nonzero pivot witness in Z, exactly as required by the coefficient row of Definition 1875.2.

The induction hypothesis gives orthogonality among the earlier promoted rows. The new residual is compared against each earlier promoted row by the same replay: its projection component was chosen from the displayed inner-product value and the pivot witness for that earlier row, so the remaining inner-product read is the zero row certified by T and A.

It remains to inspect the new pivot. If Z displays a nonzero norm-square witness, the residual is promoted into O and becomes available to later prefixes with the orthogonality just established. If the witness is absent, Z records the refusal row, and the residual is not promoted; later projection coefficients therefore cannot read it as a pivot row. The transport, continuation, provenance, and naming rows H, C, P, N preserve this finite route without adding a host basis, replacement choice, or quotient equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1875.1 and Definition 1875.2 seed the finite constructive orthogonalization packet over displayed vector-space, inner-product, norm-square, and matrix rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GramSchmidt}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a full basis theorem, dimension theorem, spectral theorem, Hilbert completion, host choice, quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection coefficient exposure, orthogonality route, pivot refusal boundary, transport, provenance, and local naming for BEDC.Derived.GramSchmidtUp.

Constructive story: A $\mathsf{GramSchmidt}^{\uparrow}$ packet is generated from a finite source-vector list, displayed orthogonalized rows, pivot nonzero or refusal witnesses, scalar projection coefficients, norm-squared rows, triangular replay, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1876

A Concrete Naming Certificate for $\mathsf{SeparatedUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on separated uniform spaces, the $\mathsf{SeparatedUniformSpace}^{\uparrow}$ packet records the finite interface at which a uniform space admits a diagonal separation witness without collapsing into a quotient completion. It refines the entourage surface of chapter 339, imports the zero-distance classifier discipline of chapter 1411, hands completion-facing reads to chapter 1473, and keeps its uniqueness boundary compatible with chapter 402 and chapter 1612. The packet gives $\mathsf{CompleteUniformSpace}^{\uparrow}$ and separated-completion consumers a reusable NameCert boundary for separated uniform data, but it does not construct quotient completions or choose representatives.

Definition 1876.1 (SeparatedUniformSpace carrier)**.** *A* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, D, Z, F, H, R, C, P, N)$$

with the following displayed rows:

1. *U is the underlying* $\mathsf{UniformSpace}^{\uparrow}$ *row, including the finite entourage basis, diagonal-membership ledger, refinement row, and uniform-continuity interface read by downstream consumers.*
2. *D is the diagonal separation witness row: whenever two carried observations are indistinguishable by all displayed entourages of U, D records the route from diagonal uniform evidence to the local point classifier.*
3. *Z is the separated-metric classifier row, used only as a finite comparison surface for metric-generated or metric-compatible uniform readings.*
4. *F is the uniform-completion handoff row, routing completion-facing consumers through the* $\mathsf{UniformCompletion}^{\uparrow}$ *interface before any Hausdorff or separated-completion uniqueness boundary is inspected.*
5. *H is the Hausdorff and separated-completion uniqueness boundary, exposing the comparison rows inherited from* $\mathsf{HausdorffCompletion}^{\uparrow}$ *and* $\mathsf{SeparatedCompletion}^{\uparrow}$ *over the same uniform data, diagonal witness, classifier, and completion handoff.*
6. *R records componentwise* hsame *transport for U, D, Z, F, H and for the local continuation, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the route*

$$U, D \longmapsto Z \longmapsto F \longmapsto H.$$

8. *P is the* Pkg *provenance over* $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedUniformSpace}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares observations only through $U, D, Z, F, H, R, C, P, N$. It has no coordinate for quotient uniform completions, host equality of uniform points, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Lean4 (definition) `BEDC.Derived.SeparatedUniformSpaceUp.SeparatedUniformSpaceTasteGate single carrier alignment`

Theorem 1876.2 (SeparatedUniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier*

$$S = (U, D, Z, F, H, R, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedUniformSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the uniform-space row, diagonal separation witness, separated-metric classifier, uniform-completion handoff, Hausdorff and separated-completion uniqueness boundary, transport, replay, provenance, and local naming rows;*
2. *uniform-basis visibility, requiring every separated comparison to begin with the displayed entourage and diagonal-membership rows of U;*
3. *diagonal separation exactness, requiring all indistinguishability-to-classifier reads to pass through D rather than through host equality;*
4. *classifier compatibility, requiring metric-generated comparisons to use the separated-metric row Z only after the uniform entourage route has been displayed;*
5. *completion handoff compatibility, requiring completion-facing consumers to read F before using the Hausdorff or separated-completion uniqueness rows in H;*
6. *ledger non-escape, excluding quotient uniform completions, host equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No local NameCert obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 1876.1. The accepted tuple displays exactly the uniform-space row, diagonal witness, classifier row, completion handoff, uniqueness boundary, and structural rows.

Uniform-basis visibility follows from the order of the carrier. The row U supplies the entourage and diagonal-membership data before D can recognize separatedness, so a consumer cannot replace finite uniform evidence by host equality of points.

Diagonal separation exactness is the role of D. An indistinguishability read through all displayed entourages reaches the local point classifier only through that row. The separated-metric row Z is compatible with this route because it is downstream of U, D and supplies only a metric-generated comparison surface.

The completion handoff is the row F. Completion consumers must pass through F before inspecting the Hausdorff or separated-completion boundary H, so uniqueness is attached to the same uniform data and diagonal witness that produced the separated comparison.

Finally inspect R, C, P, N. Transport is componentwise, replay follows the displayed route, and provenance plus local naming mention only the rows listed in the carrier. Hence the packet exports no quotient uniform completion, host equality, selected representative, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification, or bridge checking. □

Theorem 1876.3 (SeparatedUniformSpace diagonal uniformity boundary)**.** *Let $S = (U, D, Z, F, H, R, C, P, N)$ be an accepted* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier. Every comparison of two carried observations by uniform indistinguishability factors through the finite route*

$$U \longmapsto D \longmapsto Z \longmapsto R, C, P, N.$$

Thus the diagonal separation boundary is a local uniform classifier route, not a quotient by the intersection of all entourages and not a host equality principle.

Proof. Read a comparison of two carried observations from the packet. The carrier first exposes U, so the comparison is expressed through the finite entourage basis and diagonal-membership rows of the underlying uniform space.

Apply the diagonal witness D. This row is the only exported route from uniform indistinguishability to the local point classifier, and it is stored as finite NameCert data rather than as an equality of host points.

The separated-metric classifier Z may replay metric-generated instances of the same comparison, but it remains downstream of U, D. Transport and continuation in R, C preserve this order, while provenance and naming in P, N add no quotient or external equality coordinate. □

Theorem 1876.4 (SeparatedUniformSpace completion non-escape)**.** *Every completion-facing consumer of an accepted* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier*

$$S = (U, D, Z, F, H, R, C, P, N)$$

must factor through the displayed handoff

$$U, D, Z \longmapsto F \longmapsto H \longmapsto R, C, P, N.$$

Consequently the packet can feed $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, *and* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *consumers without exporting quotient uniform completions, host equality of uniform points, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted packet. The rows U, D, Z fix the uniform-space surface, diagonal separation witness, and separated classifier before any completion-facing row is available.

The handoff row F is therefore the only route into uniform completion data. A consumer cannot bypass it by selecting a completion representative, forming a quotient class, or appealing to an ambient complete uniform space.

The uniqueness boundary H is attached to the same U, D, Z, F route. It reads Hausdorff and separated-completion comparison rows only after the handoff has been displayed, so completion uniqueness remains a local NameCert boundary.

The remaining rows R, C, P, N preserve and name this route. Since the carrier has no field for quotient uniform completions, host equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, no completion-facing consumer can project such data from the packet. □

Theorem 1876.5 (SeparatedUniformSpace proximity boundary)**.** *Every accepted* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier supplies the finite diagonal, symmetry, and composition rows required by the* $\mathsf{ProximitySpace}^{\uparrow}$ *nearness boundary, but only through displayed entourages and without adding quotient filters or selected completions. The boundary route is*

$$U, D, Z, F, H \longmapsto \mathsf{ProximitySpace}^{\uparrow} \longmapsto R, C, P, N.$$

Thus a proximity-facing consumer receives separated uniform evidence as finite entourage data rather than as a quotient uniform completion, selected representative, host equality, or choice principle.

Proof. Project the carrier of Definition 1876.1. The row U supplies the finite entourage basis and diagonal-membership ledger, while D supplies the separated diagonal witness used by any downstream nearness boundary.

Apply Definition 1876.2. The displayed uniform basis, diagonal exactness, classifier compatibility, and completion handoff rows are the only obligation rows available before structural transport and naming.

Use Definition 1876.3. It keeps every uniform indistinguishability comparison on the finite route through U, D, Z, R, C, P, N, so the proximity-facing read receives diagonal and symmetry information from displayed entourages rather than from host equality.

The downstream nearness consumer is chapter 6310. Its finite nearness interface may read the separated uniform diagonal, symmetry, and composition evidence, but the separated uniform carrier has no coordinate for quotient filters, selected completions, selected representatives, countable choice, or ambient completeness, so none of those objects is exported across the boundary. □

Theorem 1876.6 (SeparatedUniformSpace diagonal-separation obligation scope)**.** *Every accepted* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *carrier*

$$S = (U, D, Z, F, H, R, C, P, N)$$

scopes diagonal separation through the following finite rows:

1. *the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage rows in* U*, including the finite entourage basis and diagonal-membership ledger;*
2. *the local diagonal witness* D *and the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row* Z*;*
3. *the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row* F*;*
4. *the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *and* $\mathsf{SeparatedCompletion}^{\uparrow}$ *uniqueness rows in* H*;*
5. *componentwise* hsame *transport* R*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

Thus every completion-facing separatedness read remains a finite uniform-entourage route and does not become quotient-completion equality, host equality of uniform points, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1876.1. The accepted packet exposes only the uniform row, diagonal witness, separated classifier, completion handoff, uniqueness boundary, transport, replay, provenance, and local naming rows.

Apply Definition 1876.2. Carrier admission, uniform-basis visibility, diagonal separation exactness, classifier compatibility, and completion handoff compatibility are the finite obligations available to the local certificate.

Use Definition 1876.3. Uniform indistinguishability reaches the local classifier through U, D, Z rather than through host equality or a quotient of all entourages.

Finally apply Definition 1876.4. The handoff to completion-facing consumers passes through F and then through the Hausdorff and separated-completion boundary H, while R, C, P, N preserve and name the same route. Hence no quotient representative, selected completion point, countable-choice schedule, or ambient completeness coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedUniformSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedUniformSpace}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1876.1, Definition 1876.2, Definition 1876.3, and Definition 1876.4 seed the finite separated-uniform interface over uniform-space, diagonal witness, separated classifier, completion handoff, uniqueness boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedUniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient uniform completions, host equality of uniform points, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-basis visibility, diagonal separation exactness, classifier compatibility, completion handoff compatibility, and non-escape for BEDC.Derived.SeparatedUniformSpaceUp.

Constructive story: A $\mathsf{SeparatedUniformSpace}^{\uparrow}$ packet is generated from a UniformSpace row, a diagonal separation witness, a SeparatedMetric classifier row, a UniformCompletion handoff, HausdorffCompletion and SeparatedCompletion uniqueness rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1877

A Concrete Naming Certificate for DyadicCoverRefinement$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the finite dyadic cover discipline, the DyadicCoverRefinement$^{\uparrow}$ packet records the common refinement data needed to compare two dyadic cover presentations without invoking an open-cover compactness principle. It sits between the dyadic cover surface of chapter 446, the one-step refinement route of chapter 1831, the ordered partition surface of chapter 1863, and the finite interval-cover handoff of chapter 1596. The packet gives Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and interval finite-cover consumers a reusable finite common-refinement certificate while staying inside displayed cover rows, restriction ledgers, transport, replay, provenance, and local naming.

Definition 1877.1 (Dyadic cover refinement carrier)**.** *A* DyadicCoverRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D_L, D_R, F, L, R_0, H, C, P, N)$$

with the following displayed rows:

1. *D_L and D_R are the two accepted* DyadicCover$^{\uparrow}$ *packets being compared. They expose only finite dyadic centers, radii, interval cells, mesh rows, coverage witnesses, and their local naming data.*
2. *F is the displayed common dyadic refinement cover. Its cells are finite dyadic intervals whose endpoints are read through the same dyadic arithmetic and partition discipline used by* DyadicIntervalRefinement$^{\uparrow}$ *and* DyadicPartition$^{\uparrow}$*.*
3. *L is the left restriction ledger. For each cell of F, it records the parent cell of D_L that contains it, together with endpoint containment and coverage preservation rows.*
4. *R_0 is the right restriction ledger. For each cell of F, it records the parent cell of D_R that contains it, together with endpoint containment and coverage preservation rows.*
5. *H records componentwise* hsame *transport for the two source covers, the common refinement cover, both restriction ledgers, continuation, provenance, and local naming.*
6. *C records displayed* Cont *replay of the finite route*

$$D_L, D_R \longmapsto F \longmapsto L, R_0.$$

7. *P is the* Pkg *provenance over* DyadicCoverRefinement$^{\uparrow}$*,* DyadicCover$^{\uparrow}$*,* DyadicIntervalRefinement$^{\uparrow}$*,* DyadicPartition$^{\uparrow}$*,* IntervalFiniteCoverCertificate$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCoverRefinement}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares the two cover presentations only through $D_L, D_R, F, L, R_0, H, C, P, N$. It has no coordinate for general open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicCoverRefinementUp.DyadicCoverRefinementTasteGate single carrier alignment`

Theorem 1877.2 (Dyadic cover refinement NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *carrier*

$$R = (D_L, D_R, F, L, R_0, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCoverRefinement}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the two source* $\mathsf{DyadicCover}^{\uparrow}$ *packets, the displayed common refinement cover, both restriction ledgers, transport, replay, provenance, and local naming;*
2. *left restriction exactness, requiring every cell of F to read its left parent cell through L and preserving the displayed dyadic endpoint containment and finite coverage rows of D_L;*
3. *right restriction exactness, requiring every cell of F to read its right parent cell through R_0 and preserving the displayed dyadic endpoint containment and finite coverage rows of D_R;*
4. *common-refinement coverage, requiring every coverage read of either source cover to factor through the finite cell family F before any downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or interval finite-cover consumer can inspect the comparison;*
5. *classifier transport through componentwise* hsame, *displayed* Cont, Pkg *provenance, and local* NameCert *naming, with no escape to general open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these five rows.

Proof. Carrier admission is the projection of Definition 1877.1. The tuple displays the two source covers, the finite common refinement cover, the two restriction ledgers, and the structural rows.

Left restriction exactness is the role of L. Each refined cell reaches the left source cover only by naming its containing parent cell in D_L and replaying the displayed endpoint containment and finite coverage rows already carried by that source cover.

Right restriction exactness is symmetric through R_0. Each refined cell reaches the right source cover only by naming its containing parent cell in D_R and replaying the corresponding endpoint containment and finite coverage rows.

Common-refinement coverage follows because the only comparison cover in the carrier is F. Any downstream read by a regular-rational, real-facing, or finite interval-cover consumer must first project F and then use the appropriate restriction ledger.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier contains no field for open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the classifier cannot export such data. □

Theorem 1877.3 (Dyadic cover common refinement exactness)**.** *Let*

$$R = (D_L, D_R, F, L, R_0, H, C, P, N)$$

be an accepted $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *carrier. Every consumer comparing the two dyadic-cover presentations* D_L *and* D_R *factors through the displayed finite route*

$$D_L, D_R \longmapsto F \longmapsto L, R_0 \longmapsto H, C, P, N.$$

Consequently the comparison reads exactly the common refinement cover, the left and right restriction ledgers, componentwise transport, continuation replay, provenance, and local naming. It cannot read open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from this packet.*

Proof. Project the accepted carrier. The only rows mentioning the two dyadic cover presentations are D_L and D_R, and the only row relating them is the displayed common refinement cover F.

Read a comparison request. Before it can inspect either source cover through a downstream consumer, it must pass to the finite cells of F; there is no separate comparison coordinate in the carrier.

The left and right projections from F are exactly L and R_0. Thus the source covers are visible only as parent-cell and coverage-preservation reads along those two restriction ledgers.

The remaining rows H, C, P, N preserve this route by componentwise transport, continuation replay, provenance, and local naming. None of those rows introduces open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient completeness, Lean verification, or bridge data, so every accepted comparison factors through the displayed common refinement. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCoverRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCoverRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1877.1, Definition 1877.2, and Definition 1877.3 seed the finite common-refinement route between two dyadic cover packets, their displayed refinement cover, both restriction ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCoverRefinement}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DyadicCoverRefinementUp`

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, Heine–Borel, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left and right restriction exactness, common-refinement coverage, classifier transport, and common-refinement exactness for BEDC.Derived.DyadicCoverRefinementUp.

Constructive story: A $\mathsf{DyadicCoverRefinement}^{\uparrow}$ packet is generated from two DyadicCover packets, one displayed finite common dyadic refinement cover, left and right restriction ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1878

A Concrete Naming Certificate for $\mathsf{ExtendedRealLine}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop–Bridges on the extended real line and one-sided real boundaries, the $\mathsf{ExtendedRealLine}^{\uparrow}$ packet records the finite interface obtained by adjoining positive and negative infinity tags to already named real data. It is a boundary surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, and $\mathsf{CountableChoiceBoundary}^{\uparrow}$: finite rows may cite ordinary real names and one-sided infinity behavior, but the packet does not import a host extended-real primitive, a quotient cut, trichotomy, a selected supremum, a countable-choice selector, or an ambient completeness theorem.

The finite-real row is consumed from chapter 38; the one-sided boundary rows are consumed from chapter 1446 and chapter 1447; the positive-bound discipline is consumed from chapter 1587; and the non-choice boundary is consumed from chapter 1872. Thus $\mathsf{ExtendedRealLine}^{\uparrow}$ is a tagged consumer of those sibling rows, with finite, lower, upper, positive-bound, and non-choice evidence displayed as carrier coordinates, rather than a host extended-real primitive that could select boundary representatives or import completeness.

Definition 1878.1 (ExtendedRealLine carrier)**.** *An* $\mathsf{ExtendedRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, R, L, U, P, B, H, C, K, N)$$

with the following displayed rows:

1. *T is the finite tag row. Each accepted observation is tagged as finite real, positive infinity, or negative infinity.*
2. *R is the finite-real row. It is read only through the already named* $\mathsf{Real}^{\uparrow}$ *surface, and it is available exactly for observations whose tag in T is finite real.*
3. *L is the lower-boundary row. It records negative-infinity-facing lower real behavior through* $\mathsf{LowerReal}^{\uparrow}$ *and exports no chosen cut representative.*
4. *U is the upper-boundary row. It records positive-infinity-facing upper real behavior through* $\mathsf{UpperReal}^{\uparrow}$ *and exports no selected supremum.*
5. *P is the positive-bound discipline row. It reads positive finite bounds through* $\mathsf{PositiveReal}^{\uparrow}$ *before any positive-infinity comparison is accepted.*
6. *B is the non-choice boundary row, inherited from* $\mathsf{CountableChoiceBoundary}^{\uparrow}$*, which refuses countable selectors, global supremum choices, selected limit schedules, and ambient completeness*

witnesses.

7. *H records componentwise* hsame *transport for tag, finite-real, lower-boundary, upper-boundary, positive-bound, non-choice, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$T \longmapsto R, L, U, P \longmapsto B.$$

9. *K is the* Pkg *provenance over* $\mathsf{ExtendedRealLine}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{CountableChoiceBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ExtendedRealLine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, L, U, P, B, H, C, K, N$. It has no coordinate for a host extended-real type, quotient cuts, trichotomy, selected suprema, countable-choice selectors, ambient completeness, Lean verification, or bridge data.

Lean4 ✓ `BEDC.Derived.ExtendedRealLineUp.TasteGate.ExtendedRealLineTasteGate single carrier alignment`

Theorem 1878.2 (ExtendedRealLine NameCert obligations)**.** *For every accepted* $\mathsf{ExtendedRealLine}^{\uparrow}$ *carrier*

$$E = (T, R, L, U, P, B, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{ExtendedRealLine}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tag row, finite-real row, lower-boundary row, upper-boundary row, positive-bound row, non-choice boundary, transport, replay, provenance, and local naming;*

2. *tag classifier stability, requiring every consumer comparison to preserve the finite real, positive infinity, and negative infinity cases named by T;*

3. *finite-real seal discipline, requiring finite-tagged observations to read ordinary real data only through R and hence through* $\mathsf{Real}^{\uparrow}$*;*

4. *one-sided infinity boundary nonescape, requiring negative-infinity reads to pass through L and* $\mathsf{LowerReal}^{\uparrow}$*, and positive-infinity reads to pass through U,* $\mathsf{UpperReal}^{\uparrow}$*, and the positive-bound discipline P;*

5. *provenance and local naming, requiring H, C, K, N to transport, replay, source, and name only the displayed rows of the packet;*

6. *non-choice refusal, requiring the boundary row B to exclude host extended-real primitives, quotient cuts, trichotomy, selected suprema, countable-choice selectors, ambient completeness theorems, Lean verification, and bridge checking.*

No ExtendedRealLine obligation is stored outside these six rows.

Lean4 ✓ `BEDC.Derived.ExtendedRealLineUp.ExtendedRealLineTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1878.1. The accepted tuple displays exactly the tag row, the three tag-dependent value rows, the positive-bound discipline, the non-choice boundary, and the structural rows.

Tag classifier stability follows from the role of T. Every accepted observation is first classified as finite real, positive infinity, or negative infinity, and all later rows are read under that displayed tag rather than under host trichotomy for an ambient extended-real type.

The finite-real seal is the row R. A finite-tagged observation reaches ordinary real data only through R, whose source is $\mathsf{Real}^{\uparrow}$; hence no quotient cut or host extended-real primitive is projected from the finite case.

For the one-sided tags, negative-infinity behavior is read through L and $\mathsf{LowerReal}^{\uparrow}$, while positive-infinity behavior is read through U, $\mathsf{UpperReal}^{\uparrow}$, and the positive-bound row P. The non-choice row B then refuses selected suprema, countable-choice selectors, selected limit schedules, and ambient completeness witnesses.

Finally inspect H, C, K, N. Transport is componentwise, replay follows the displayed route, and provenance plus local naming mention only the rows listed in the carrier. Therefore the six obligation rows exhaust the local NameCert surface. □

Theorem 1878.3 (ExtendedRealLine finite boundary)**.** *Every consumer route out of an accepted* $\mathsf{ExtendedRealLine}^{\uparrow}$ *carrier*

$$E = (T, R, L, U, P, B, H, C, K, N)$$

factors through

$$\mathsf{Real}^{\uparrow},\ \mathsf{LowerReal}^{\uparrow},\ \mathsf{UpperReal}^{\uparrow},\ \mathsf{PositiveReal}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Consequently the packet exports no host extended-real primitive, quotient cut, trichotomy, selected supremum, countable-choice selector, or ambient completeness theorem.

Proof. Begin with a consumer of E. The tag row T is the only entry point, so the consumer must first decide which displayed case of the finite packet is being read.

In the finite case the consumer reaches R, and R routes through $\mathsf{Real}^{\uparrow}$. In the negative-infinity case the consumer reaches L, and L routes through $\mathsf{LowerReal}^{\uparrow}$. In the positive-infinity case the consumer reaches U and then the positive-bound discipline P, so the route passes through $\mathsf{UpperReal}^{\uparrow}$ and $\mathsf{PositiveReal}^{\uparrow}$.

The carrier itself is a BHist packet. Its equality-like motion is componentwise hsame, its rereading surface is Cont, its source ledger is Pkg, and its public name is the local NameCert row.

The non-choice row B is read on every boundary-facing route. Since B is inherited from $\mathsf{CountableChoiceBoundary}^{\uparrow}$, it blocks countable-choice selectors, selected suprema, selected limits, and ambient completeness witnesses; since the carrier inventory has no host extended-real coordinate, no quotient cut or trichotomy principle can be exported either. □

Closure status for $\mathsf{NameCert}_{\mathsf{ExtendedRealLine}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtendedRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1878.1, Definition 1878.2, and Definition 1878.3 seed the finite extended-real boundary over ordinary real rows, one-sided lower and upper rows, positive-bound discipline, non-choice refusal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtendedRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host extended-real primitives, quotient cuts, trichotomy, selected suprema, countable-choice selectors, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tag classifier stability, finite-real seal discipline, one-sided infinity boundary nonescape, provenance and local naming, non-choice refusal, and finite consumer factorization for BEDC.Derived.ExtendedRealLineUp.

Constructive story: An $\mathsf{ExtendedRealLine}^{\uparrow}$ packet is generated from a finite tag row, a $\mathsf{Real}^{\uparrow}$ finite row, $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ one-sided boundary rows, a $\mathsf{PositiveReal}^{\uparrow}$ positive-bound row, a $\mathsf{CountableChoiceBoundary}^{\uparrow}$ non-choice row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1879

A Concrete Naming Certificate for $\mathsf{KroneckerLemma}^{\uparrow}$

Following Kronecker and Bishop–Bridges constructive analysis, the $\mathsf{KroneckerLemma}^{\uparrow}$ packet records the finite Toeplitz–Cesaro route by which weighted prefix tails are shown to be summability-controlled before any completion-facing real endpoint is read. It consumes the finite matrix-window discipline of chapter 1682, the displayed averaging surface of chapter 1765, the convergence-window route of chapter 1827, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite summability certificate for Kronecker-style tail control; it does not import quotient sums, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1879.1 (Kronecker lemma carrier)**.** *A* $\mathsf{KroneckerLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, T, H, C, R, E, L, Q, N)$$

with the following displayed rows:

1. *S is the source weighted-tail row, exposing only the finite sequence windows, partial-sum windows, and denominator windows currently requested by the Kronecker estimate.*
2. *B is the bounded-prefix ledger, recording the finite prefix-sum bounds and monotone denominator comparisons used by the inspected tail.*
3. *T is the* $\mathsf{ToeplitzLemma}^{\uparrow}$ *row carrying the finite lower-triangular coefficient window that transforms the bounded prefix data into the displayed weighted tail.*
4. *H is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row for the Cesaro averaging of the transformed tail windows.*
5. *C is the* $\mathsf{CesaroConvergence}^{\uparrow}$ *row certifying that the displayed Hardy–Cesaro means enter the convergence-window surface.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached only after the Toeplitz, Hardy–Cesaro, and Cesaro-convergence rows have been displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational handoff R has been exposed.*
8. *L stores componentwise* hsame *transport and displayed* Cont *replay for the source tail row, bounded-prefix ledger, Toeplitz row, Hardy–Cesaro row, Cesaro-convergence row, regular handoff, real seal, provenance, and local naming.*

9. *Q is the* Pkg *provenance over* $\mathsf{KroneckerLemma}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{KroneckerLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, T, H, C, R, E, L, Q, N$. It has no coordinate for quotient sums, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.KroneckerLemmaUp.TasteGate.KroneckerLemmaTasteGate single carrier alignment`

Theorem 1879.2 (Kronecker lemma NameCert obligations)**.** *For every accepted* $\mathsf{KroneckerLemma}^{\uparrow}$ *carrier*

$$K = (S, B, T, H, C, R, E, L, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{KroneckerLemma}^{\uparrow}}$ *surface consists exactly of the following six finite obligation rows:*

1. *carrier admission for the displayed source weighted-tail row, bounded-prefix ledger, Toeplitz row, Hardy–Cesaro row, Cesaro-convergence row, regular handoff, real seal, transport and replay ledger, provenance, and local naming row;*
2. *bounded-prefix exactness, requiring every accepted tail estimate to read its partial-sum bound and denominator comparison through S and B;*
3. *Toeplitz-window classification, requiring the weighted-tail transform to pass through the finite coefficient row T before any Cesaro mean is available;*
4. *Cesaro route compatibility, requiring H and C to consume the same transformed windows exported by T and to expose no independent convergence witness;*
5. *regular handoff and real seal factorization, requiring every endpoint-facing read to pass through R and then E after the Toeplitz–Cesaro route has been displayed;*
6. *ledger non-escape, requiring L, Q, N to preserve, source, and name only the displayed rows and to exclude quotient sums, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 1879.1. Carrier admission is exactly the inventory $S, B, T, H, C, R, E, L, Q, N$ so the local certificate has no detached row for an infinite weighted series, a chosen limit, or a host equality principle.

Bounded-prefix exactness is read before the summability transform. The source row S exposes only the inspected sequence, partial-sum, and denominator windows, while B records the finite prefix bound and denominator comparison needed by the current tail estimate.

The route then passes through T, H, C. The Toeplitz row T supplies the finite coefficient window for the weighted-tail transform, the Hardy–Cesaro row H averages the transformed windows, and the Cesaro-convergence row C is the only convergence-window classifier available to the packet.

Finally, R and E seal the endpoint-facing read. The ledger L, provenance Q, and local naming row N transport, replay, source, and name this same route; none of those rows contributes quotient sums, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge data. □

Theorem 1879.3 (Kronecker lemma Toeplitz–Cesaro route)**.** *Every accepted* $\mathsf{KroneckerLemma}^{\uparrow}$ *read reaching a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$S, B \longmapsto T \longmapsto H \longmapsto C \longmapsto R \longmapsto E \longmapsto (L, Q, N).$$

Thus a completion-facing consumer receives weighted-tail control only after the bounded-prefix ledger, Toeplitz matrix window, Hardy–Cesaro averaging row, Cesaro-convergence row, regular rational handoff, real seal, transport and replay ledger, provenance, and local NameCert have all been displayed.

Proof. Start with a completion-facing read of an accepted carrier. By Definition 1879.1, the only source-side data available before the transform are S and B, so the read first consumes the displayed sequence windows, partial-sum windows, denominator windows, and bounded-prefix ledger.

The transformed weighted-tail row is then forced through T. This is the finite Toeplitz surface of chapter 1682; it supplies the coefficient-window route and does not expose an infinite matrix or a host summability method.

The Cesaro part is exhausted by H and C. The Hardy–Cesaro row H supplies the finite averaging route of chapter 1765, and the Cesaro-convergence row C supplies the convergence-window handoff of chapter 1827.

Only after those rows have been displayed can the packet reach the regular rational handoff R and then the real seal E. The remaining rows L, Q, N preserve the route by transport, replay, provenance, and local naming, so no consumer can bypass the Toeplitz–Cesaro path through quotient sums, selected limits, countable choice, host real equality, trichotomy, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{KroneckerLemma}^{\uparrow}$ completion-facing read factors through finite source weighted-tail windows, a bounded-prefix ledger, a ToeplitzLemma row, a HardyCesaroMean row, a CesaroConvergence row, RegSeqRat handoff, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: toeplitzlemma, hardycesaromean, cesaroconvergence, regseqrat, real: $\mathsf{KroneckerLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source weighted-tail windows, bounded-prefix data, Toeplitz finite matrix windows, Hardy–Cesaro averaging, Cesaro convergence, regular handoff, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KroneckerLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{KroneckerLemma}^{\uparrow}$ packet is generated from source weighted-tail windows, bounded-prefix data, a Toeplitz finite matrix row, HardyCesaroMean averaging, CesaroConvergence classification, RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KroneckerLemma}^{\uparrow}}$, seed)

Scope closed: Definition 1879.1, Definition 1879.2, and Definition 1879.3 seed the finite Toeplitz–Cesaro summability route for Kronecker-style weighted-tail control.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KroneckerLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient sums, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-prefix exactness, Toeplitz-window classification, Cesaro route compatibility, regular handoff, real seal factorization, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.KroneckerLemmaUp.

Chapter 1880

A Concrete Naming Certificate for $\mathsf{DarbouxSum}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxSum}^{\uparrow}$ packet names the finite upper/lower sum surface over a dyadic partition and step-function brackets. It consumes the dyadic partition ledger of chapter 1863, the bracket rows of chapter 428, the tagged finite-sum consumer of chapter 1615, the Darboux integration carrier of chapter 1460, and the terminal real-name seal of chapter 38. The packet supplies the finite SU/SL boundary that later Darboux and Riemann integration chapters can cite without importing open-cover compactness, selected mesh limits, measure-theoretic integration, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1880.1 (Darboux sum carrier)**.** *A* $\mathsf{DarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, L, U, SL, SU, G, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{DyadicPartition}^{\uparrow}$ *packet naming the finite ordered dyadic cell list and its common-refinement ledger.*
2. *L and U are lower and upper* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packets over the displayed partition cells, with every bracket value read through the same partition row P.*
3. *SL is the finite lower Darboux-sum row computed from the displayed cell widths of P and the carried lower values of L.*
4. *SU is the finite upper Darboux-sum row computed from the displayed cell widths of P and the carried upper values of U.*
5. *G is the upper-minus-lower gap row, exposing only the finite comparison between SU and SL after the partition and step-function brackets have been displayed.*
6. *H records componentwise* hsame *transport for the partition row, lower and upper step-function rows, lower and upper finite sum rows, gap row, continuation replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from partition cells to bracket reads, from bracket reads to finite sums, and from the two finite sums to the gap comparison.*
8. *Q is the* Pkg *provenance over* $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{DarbouxSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, L, U, SL, SU, G, H, C, Q, N$. It has no coordinate for open-cover compactness, selected mesh limits, measure-theoretic integration, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DarbouxSumUp.TasteGate.DarbouxSumTasteGate single carrier alignment`

Theorem 1880.2 (Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxSum}^{\uparrow}$ *carrier*

$$S = (P, L, U, SL, SU, G, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed dyadic partition packet, lower and upper step-function brackets, finite lower and upper sum rows, gap row, transport and replay ledger, provenance, and local naming row;*
2. *partition compatibility, requiring every bracket value and every cell width used by SL and SU to be read through the displayed* $\mathsf{DyadicPartition}^{\uparrow}$ *row P;*
3. *bracket stability, requiring* hsame *transport of L and U to carry the same lower/upper role over any displayed common refinement of P;*
4. *finite sum stability, requiring SL and SU to replay as finite cellwise sums after transport of the partition and bracket rows;*
5. *ledger non-escape, requiring H, C, Q, N to preserve, replay, source, and name only the displayed rows and to exclude open-cover compactness, selected mesh limits, measure-theoretic integration, quotient real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Carrier admission is the projection of Definition 1880.1. The accepted packet displays exactly the partition, lower bracket, upper bracket, lower finite sum, upper finite sum, gap, transport, replay, provenance, and local naming rows.

Partition compatibility is forced before any sum row can be read. The row P supplies the finite ordered dyadic cells and their widths, while L and U supply lower and upper values over those same displayed cells.

Bracket stability and finite sum stability are then replayed through H and C. Transport moves the partition and both brackets componentwise, and continuation replay recomputes SL and SU as finite cellwise sums over the transported rows.

Finally, Q and N source and name this same finite route. Since no carrier coordinate stores open-cover compactness, selected mesh limits, measure data, quotient real equality, or ambient completeness, the ledger non-escape row exhausts the remaining obligation surface. □

Theorem 1880.3 (Darboux upper-lower gap)**.** *For every accepted* $\mathsf{DarbouxSum}^{\uparrow}$ *carrier $S = (P, L, U, SL, SU, G, H, C, Q, N)$, the upper-minus-lower gap row G is read only through the displayed finite partition P, the displayed lower and upper step-function brackets L and U, and the resulting finite sum rows SL and SU. Any consumer of G therefore receives a finite partition-and-bracket comparison surface, not open-cover compactness, a selected mesh limit, measure-theoretic integration, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only rows upstream of G are P, L, U, SL, SU together with the transport and replay structure that preserves them.

The lower and upper finite sums are computed before the gap is visible. The row SL uses the cell widths of P and the carried lower values of L, while SU uses the same cell widths and the carried upper values of U.

Reading G then compares these two displayed finite sums. Componentwise transport through H and replay through C may move the packet to a displayed common refinement, but the read still factors through the transported partition, transported brackets, and recomputed finite sum rows.

The excluded analytic objects are absent from the carrier. Thus no consumer can obtain them by inspecting the upper-minus-lower gap. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxSum}^{\uparrow}$ packet is generated from a DyadicPartition row, lower and upper DyadicStepFunction brackets, lower and upper finite Darboux sums, an upper-minus-lower gap row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxSum}^{\uparrow}}$, seed)

Scope closed: Definition 1880.1, Definition 1880.2, and Definition 1880.3 seed the finite Darboux upper/lower sum interface over displayed dyadic partitions and step-function brackets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DarbouxSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected mesh limits, measure-theoretic integration, quotient real equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition compatibility, bracket stability, finite sum stability, ledger non-escape, and upper-lower gap factorization for BEDC.Derived.DarbouxSumUp.

Chapter 1881

A Concrete Naming Certificate for $\mathsf{CauchyRealApartness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRealApartness}^{\uparrow}$ packet records the positive finite boundary by which two displayed real names are certified apart before any order trichotomy, quotient stream equality, selected limit, or ambient completeness principle can be read. It sits over the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the finite window discipline of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names a constructive positive-apartness surface for later order and metric consumers; it does not decide equality or total order on $\mathsf{Real}^{\uparrow}$.

Definition 1881.1 (Cauchy real apartness carrier)**.** *A* $\mathsf{CauchyRealApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E_0, E_1, R_0, R_1, W_0, W_1, D, T, H, C, P, N)$$

with the following displayed rows:

1. *E_0 and E_1 are the two* $\mathsf{Real}^{\uparrow}$ *seal rows. They are terminal classifier rows only; they expose no host real equality, quotient representative, trichotomy branch, or completeness theorem.*
2. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness rows. Each reads the corresponding real name through displayed regular rational data before the apartness tolerance is interpreted.*
3. *W_0 and W_1 are finite* $\mathsf{StreamName}^{\uparrow}$ *window rows for the same precision request. They record the inspected windows and no infinite stream equality.*
4. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores a displayed positive dyadic gap, the comparison ledger, and the slack budget used to separate the two inspected windows.*
5. *T is the apartness transport row. It records componentwise* hsame *transport for the two real seals, the regular readbacks, the finite windows, the positive tolerance ledger, replay, provenance, and local naming.*
6. *H is the displayed* Cont *continuation row. It replays a consumer request through $E_0, E_1, R_0, R_1, W_0, W_1, D, T, P,$ without adding a selected limit or an equality decision.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealApartness}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite positive-apartness packet. It has no coordinate for trichotomy, quotient stream equality, selected limits, countable choice, metric-space completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRealApartnessUp.CauchyRealApartnessTasteGate single carrier alignment`

Theorem 1881.2 (Cauchy real apartness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealApartness}^{\uparrow}$ *carrier*

$$A = (E_0, E_1, R_0, R_1, W_0, W_1, D, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealApartness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two Real seals, two regular rational readbacks, two finite StreamName windows, positive dyadic tolerance ledger, transport row, continuation row, provenance row, and local naming row;*
2. *positive tolerance stability, requiring the positive dyadic gap in* D *to remain positive under the displayed comparison ledger and the componentwise transports in* T*;*
3. *RegSeqRat handoff, requiring every apartness read to pass from the two finite windows* W_0, W_1 *through* R_0, R_1 *before the Real seal rows* E_0, E_1 *are consumed;*
4. *Real seal nonescape, requiring* E_0 *and* E_1 *to be terminal seals rather than carriers of equality, trichotomy, selected representatives, selected limits, or completeness;*
5. *ledger rows, requiring* T, H, P, N *to transport, replay, source, and name only the displayed rows of the packet.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1881.1. The accepted packet displays exactly the two terminal real seals, two regular rational readbacks, two finite window rows, one positive dyadic tolerance ledger, and the structural rows.

Positive tolerance stability is finite. The row D records the displayed positive gap and its slack comparisons, while T transports only the named rows componentwise, so no transport step can replace the gap with an undecided order comparison.

The handoff obligation follows the row order. The finite windows W_0, W_1 are read first, the regular rational witnesses R_0, R_1 interpret those windows, and only then may the real seals E_0, E_1 be consumed by an apartness reader.

Finally, the seal and ledger obligations are exhausted by E_0, E_1, T, H, P, N. Those rows preserve and name the displayed route; they do not add trichotomy, quotient stream equality, selected limits, countable choice, metric completeness, ambient Real completeness, Lean verification, or bridge data. □

Lean4 ✓ `BEDC.Derived.CauchyRealApartnessUp.CauchyRealApartnessCarrier namecert obligations`

Theorem 1881.3 (Cauchy real apartness RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyRealApartness}^{\uparrow}$ *read reaching a* $\mathsf{Real}^{\uparrow}$ *apartness consumer factors through the finite route*

$$W_0, W_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto E_0, E_1 \longmapsto (T, H, P, N).$$

Thus the apartness witness reaches $\mathsf{Real}^{\uparrow}$ *only through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Proof. Start with a Real-facing apartness read of an accepted carrier. By Definition 1881.1, the only window data available to that read are W_0 and W_1.

The regular rational stage is forced next. The rows R_0 and R_1 read the two finite StreamName windows as regular rational witnesses, so the consumer receives no quotient stream equality or infinite stream comparison.

The positive gap is then read from D. This dyadic ledger supplies the finite positive tolerance and the displayed slack comparisons used by the apartness request.

Only after these rows are displayed can the read consume E_0 and E_1 as terminal Real seals. The structural rows T, H, P, N preserve the same route by transport, continuation, provenance, and local naming. □

Theorem 1881.4 (Cauchy real apartness nonescape)**.** *An accepted* $\mathsf{CauchyRealApartness}^{\uparrow}$ *carrier exports no row for order trichotomy, quotient stream equality, selected limits, countable choice, metric-space completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer asking for one of those principles must leave the local packet and cannot obtain it from the carrier rows*

$$E_0, E_1, R_0, R_1, W_0, W_1, D, T, H, C, P, N.$$

Proof. Inspect the carrier rows. The StreamName rows are finite windows, the RegSeqRat rows are displayed readbacks, and the DyadicRatCore row is a positive tolerance ledger; none of these rows carries quotient stream equality or a selected limit.

The two Real rows are seals only. They are consumed after the regular rational and dyadic rows have been displayed, and their role is terminal naming rather than equality decision, order trichotomy, metric completeness, or ambient completeness.

The remaining rows are structural. Transport, continuation, provenance, and local naming preserve the displayed route, so countable choice, selected representatives, quotient equality, trichotomy, metric-space completeness, and $\mathsf{Real}^{\uparrow}$ completeness remain outside the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealApartness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1881.1, Definition 1881.2, Definition 1881.3, and Definition 1881.4 seed the finite positive-apartness route over two Real seals, regular rational readbacks, StreamName windows, and a positive DyadicRatCore tolerance ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, full order trichotomy, metric-space completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, positive tolerance stability, RegSeqRat handoff, Real seal nonescape, ledger exhaustion, and TasteGate field faithfulness for BEDC.Derived.CauchyRealApartnessUp.

Constructive story: A $\mathsf{CauchyRealApartness}^{\uparrow}$ packet is generated from two RealUp seals, two RegSeqRat readbacks, two StreamName windows, a positive DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont continuation, Pkg provenance, and a local NameCert row.

Chapter 1882

Rational Interval Cover NameCert Construction

Following Bishop–Bridges constructive analysis and the finite interval methods of Moore–Kearfott–Cloud, the $\mathsf{RationalIntervalCover}^{\uparrow}$ packet records finite rational interval covers as displayed endpoint, coverage, and refinement data. It sits between the rational interval surface of chapter 354, the refinement rows of chapter 1432, the located interval boundary of chapter 394, the dyadic tolerance core of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite cover ledger for Real and Completion consumers; it is not an open-cover compactness principle, a countable-choice construction, quotient equality of completed reals, selected-limit data, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1882.1 (RationalIntervalCover carrier)**.** *A* $\mathsf{RationalIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (I, F, R, W, H, K, P, N)$$

with the following displayed rows:

1. *I is the finite family of* $\mathsf{RationalInterval}^{\uparrow}$ *rows whose endpoint and containment surfaces are read through chapter 354.*
2. *F is the finite coverage ledger, recording exactly which rational samples, located interval requests, and regular rational windows are covered by rows of I.*
3. *R is the finite refinement ledger, displaying the* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *rows by which accepted subcovers and retained windows refine through the intervals in I.*
4. *W is the StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *window boundary, naming the finite stream indices, rational tolerances, and regular-readback requests for which the cover is available.*
5. *H records componentwise* hsame *transport for interval-family rows, coverage rows, refinement rows, window-boundary rows, continuation rows, provenance, and local naming.*
6. *K records displayed* Cont *replay, so a consumer can reread I, F, R, W, H, P, N without changing the cover ledger.*
7. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalCover}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RationalIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalCover}^{\uparrow}}$ *row.*

The classifier compares accepted covers only through the displayed rows I, F, R, W, H, K, P, N. It has no coordinate for an arbitrary open cover, countable cover selection, quotient real equality, selected limit, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RationalIntervalCoverUp.RationalIntervalCoverTasteGate single carrier alignment`

Theorem 1882.2 (RationalIntervalCover NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalCover}^{\uparrow}$ *carrier*

$$C = (I, F, R, W, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite rational interval family, coverage ledger, refinement ledger, StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *boundary, transports, replay, provenance, and local naming;*
2. *interval-family exactness, requiring every cover read to expose its* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint and containment rows before any completion-facing consumer may use the cover;*
3. *coverage-ledger exactness, requiring every covered rational sample, located interval request, or regular rational window to appear as a displayed row of F;*
4. *refinement-ledger exactness, requiring every retained subcover or narrowed window to factor through displayed* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *rows in R;*
5. *window-boundary ordering, requiring the StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows in W to be read before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *ledger non-escape, requiring H, K, P, N to export no open-cover compactness, countable choice, quotient real equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No RationalIntervalCover obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1882.1. The accepted packet displays exactly one finite interval-family row, one finite coverage ledger, one finite refinement ledger, one finite window boundary, and the structural rows that transport, replay, package, and name them.

Interval-family exactness follows because I is the only coordinate containing $\mathsf{RationalInterval}^{\uparrow}$ packets. A consumer can therefore inspect interval endpoints and containment only through the displayed rational interval rows.

Coverage-ledger exactness is the row F. Every accepted cover request must be one of its displayed rational samples, located interval requests, or regular rational windows, and there is no second coordinate from which an unrecorded cover read could be obtained.

Refinement-ledger exactness is the row R. Its entries are the displayed $\mathsf{RationalIntervalRefinement}^{\uparrow}$ witnesses, so any retained subcover or narrowed window is read through those finite refinement rows.

The remaining rows W, H, K, P, N bind the cover to finite stream windows, regular rational readback, transport, replay, provenance, and local naming. Since the excluded compactness, choice, quotient, limit, and completeness objects are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1882.3 (RationalIntervalCover finite refinement)**.** *Every accepted* $\mathsf{RationalIntervalCover}^{\uparrow}$ *carrier $C = (I, F, R, W, H, K, P, N)$ refines through the displayed* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *rows in R. More precisely, each accepted subcover, retained window, or consumer-facing narrowed interval is obtained by reading a finite row of R against the interval family I, coverage ledger*

F, StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ *boundary W, componentwise* hsame *transport H, displayed* Cont *replay K,* Pkg *provenance P, and local* $\mathsf{NameCert}$ *row N. The refinement route does not use open-cover compactness, countable choice, quotient real equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 1882.1. The interval family I supplies the finite rational interval rows, and F records which rational samples, located interval requests, and regular rational windows are covered by those rows.

The only refinement coordinate is R. Reading a subcover or narrowed window must therefore choose a displayed row of R, and that row is a $\mathsf{RationalIntervalRefinement}^{\uparrow}$ witness by the carrier definition.

The window boundary W fixes the StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ requests for which the refinement is being read. The structural rows H and K transport and replay the same finite data, while P and N bind provenance and local naming.

No carrier coordinate contains an arbitrary open cover, a countable selector, a quotient-real equality proof, a selected limit, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. Thus every accepted cover refinement is finite and factors through the displayed rational interval refinement ledger. □

Theorem 1882.4 (RationalIntervalCover finite consumer factorization)**.** *For every accepted* $\mathsf{RationalIntervalCover}^{\uparrow}$ *carrier*

$$C = (I, F, R, W, H, K, P, N),$$

every Real or Completion consumer of the cover factors through the displayed route

$$I \to F \to R \to W \to H, K \to P, N.$$

Here I supplies the $\mathsf{RationalInterval}^{\uparrow}$ *rows, F supplies the finite coverage ledger, R supplies the* $\mathsf{RationalIntervalRefinement}^{\uparrow}$ *rows, and W supplies StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *boundary data before any* $\mathsf{Real}^{\uparrow}$ *seal is reachable. The factorization route has no coordinate for open-cover compactness, countable choice, quotient real equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 1882.1. The first admissible reads are the rational interval rows in I and the finite coverage ledger F, because the NameCert obligations require interval-family exactness and coverage-ledger exactness before a Real or Completion consumer can use the cover.

Next use Definition 1882.3. Any retained window or consumer-facing narrowed interval is forced through a displayed row of R, so the consumer cannot bypass the finite $\mathsf{RationalIntervalRefinement}^{\uparrow}$ ledger when it asks for a narrower cover view.

The window-boundary obligation then orders the StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ data in W before the terminal real-facing seal. The structural rows H and K transport and replay the same displayed data, and P, N bind the route to provenance and local naming.

No coordinate of C stores open-cover compactness, countable choice, quotient real equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every accepted Real or Completion consumer reaches the cover only through the finite route $I \to F \to R \to W \to H, K \to P, N$. □

Falsifiable prediction: Every $\mathsf{RationalIntervalCover}^{\uparrow}$ read factors through a finite family of $\mathsf{RationalInterval}^{\uparrow}$ rows, one finite coverage ledger, one finite $\mathsf{RationalIntervalRefinement}^{\uparrow}$ ledger, one StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ window boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: rational-interval, rational-interval-refinement, located-interval, dyadicratcore, regseqrat, real: $\mathsf{RationalIntervalCover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite interval family, finite coverage ledger, finite refinement ledger, stream-window boundary, regular rational readback boundary, and terminal Real-facing provenance as one cover packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalCover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1882.1, Definition 1882.2, Definition 1882.3, and Definition 1882.4 seed the finite rational interval cover route from rational interval rows through finite coverage, displayed refinement, StreamName and regular rational windows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-family exactness, coverage-ledger exactness, refinement-ledger exactness, window-boundary ordering, ledger non-escape, and finite consumer factorization for BEDC.Derived.RationalIntervalCoverUp.

Constructive story: A $\mathsf{RationalIntervalCover}^{\uparrow}$ packet is generated from a finite family of $\mathsf{RationalInterval}^{\uparrow}$ rows, a finite coverage ledger, a finite $\mathsf{RationalIntervalRefinement}^{\uparrow}$ ledger, StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ window-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1883

A Concrete Naming Certificate for $\mathsf{RoundedIdealCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, Vickers-style formal topology, and the Edalat–Heckmann formal-ball domain presentation of metric completion, the $\mathsf{RoundedIdealCompletion}^{\uparrow}$ packet names completion data as rounded directed families of formal balls rather than quotient classes of Cauchy names. It sits after the formal-ball interface of chapter 1510, the metric source surface of chapter 57, the completion boundary of chapter 95, the stream-window interface of chapter 126, and the dyadic radius ledger of chapter 291.

The carrier is intended for completion consumers that can inspect finite lower information about an ideal. It records only finite BHist rows for roundedness, directedness, cofinal refinements, classifier transport, replay, provenance, and local naming, so the packet can be replayed without selecting a limit point or assuming that the ambient metric space is already complete.

Definition 1883.1 (Rounded ideal completion carrier)**.** *A* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, F, D, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including accepted point histories, metric-distance witnesses, and the point classifier used by every formal-ball center.*
2. *F is the finite* $\mathsf{FormalBall}^{\uparrow}$ *family row. Its entries are formal balls (x, ρ) read through the same metric row M, with radius requests exposed through finite windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. It records positive-radius witnesses, strict comparison budgets, radius addition, and refinement inequalities for the rows of F.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to read finite center observations and finite ideal membership requests.*
5. *R is the roundedness row: every displayed ball in F is accompanied by a smaller displayed ball whose dyadic radius comparison is recorded in D and whose center window is read in S.*
6. *E is the directedness row: every displayed pair of balls in F has a common displayed refinement in F, with both comparison witnesses recorded in D over the same metric row M.*
7. *T records componentwise* hsame *transport for the metric source, formal-ball family, dyadic ledger, stream window, roundedness row, directedness row, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from finite ideal membership requests through $M, F, D, S, R, E, T, P, N$.*
9. *P is the* Pkg *provenance over* RoundedIdealCompletion$^{\uparrow}$, FormalBall$^{\uparrow}$, Met$^{\uparrow}$, Completion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RoundedIdealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $M, F, D, S, R, E, T, C, P, N$. The carrier has no coordinate for quotient completion, selected limits, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RoundedIdealCompletionUp.TasteGate.RoundedIdealCompletionTasteGate single carrier alignment`

Theorem 1883.2 (RoundedIdealCompletion NameCert obligations)**.** *For every accepted* RoundedIdealCompletion$^{\uparrow}$ *carrier*

$$I = (M, F, D, S, R, E, T, C, P, N),$$

the local NameCert$_{\text{RoundedIdealCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, formal-ball family, dyadic ledger, stream window, roundedness row, directedness row, transport row, replay row, provenance row, and local naming row;*
2. *roundedness, requiring each displayed ideal member to admit a strictly smaller displayed formal ball in the same family, with the comparison recorded in D and the center read through S;*
3. *directedness, requiring every displayed pair of ideal members to admit a common displayed refinement in the same family, with both comparison witnesses recorded in D over M;*
4. *cofinal refinement stability, requiring any finite radius tightening accepted by D and replayed through C to remain inside the same ideal-family row F;*
5. *classifier transport, requiring* hsame *comparison of accepted packets to be componentwise through T and to preserve the displayed roundedness and directedness rows;*
6. *ledger non-escape, requiring every completion-facing read to pass through $M, F, D, S, R, E, T, C, P, N$ and to export no quotient completion, selected limit, countable-choice selector, open-cover compactness proof, or ambient completeness datum.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1883.1. The accepted packet displays exactly the metric source, formal-ball family, dyadic ledger, stream window, roundedness row, directedness row, transport, replay, provenance, and local naming row.

Roundedness is read from R. For each displayed ideal member, R names a smaller displayed formal ball, while D supplies the strict radius comparison and S supplies the finite center window.

Directedness is read from E. Given two displayed ideal members, E supplies a common formal-ball refinement in F, and D records the two comparison witnesses over the same metric row M.

Cofinal refinement stability and classifier transport are structural. The replay row C keeps finite tightening requests inside the same displayed route, and the transport row T compares packets componentwise without adding a second classifier.

Finally, P and N only source and name the displayed packet. Since quotient completion, selected limits, countable choice, open-cover compactness, and ambient completeness have no carrier coordinate, no replay edge, and no provenance row, the local NameCert surface is exactly the listed obligation surface. □

Theorem 1883.3 (RoundedIdealCompletion cofinal replay)**.** *Let $I = (M, F, D, S, R, E, T, C, P, N)$ be an accepted* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *carrier. Every finite completion-facing request for a tighter formal-ball neighbourhood is answered by the route*

$$M, F \longmapsto D, S \longmapsto R, E \longmapsto T, C, P, N.$$

The answer is a displayed member of the same rounded directed family F, not a selected limit, not a quotient representative, and not an ambient-completeness witness.

Proof. Begin with the metric source and formal-ball family. Projection gives M and F, so the request can inspect only the displayed centers, radii, and formal-ball rows of the accepted packet.

The requested tightening is checked against D and S. The dyadic ledger records the radius comparison budget, and the stream window records the finite center observations used by that comparison.

Roundedness and directedness then supply the internal answer. The row R handles a one-ball tightening request, while E handles the common-refinement demand generated by a pair of displayed members.

The structural rows preserve the route. The transport row T, replay row C, provenance row P, and naming row N add no field for quotient classes, selected limits, or completeness principles, so the answer remains a displayed member of the same rounded directed family. □

Theorem 1883.4 (RoundedIdealCompletion classifier transport)**.** *If two accepted* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *carriers are related by componentwise* hsame *transport through their transport rows, then their classifier-visible rounded ideal data agree on the metric source, formal-ball membership rows, dyadic comparisons, stream windows, roundedness witnesses, directedness witnesses, replay route, provenance, and local name.*

Consequently classifier transport preserves cofinal refinement stability and ledger non-escape for the rounded ideal completion surface.

Proof. Project the two accepted carriers. The only classifier-visible coordinates are the metric source, formal-ball family, dyadic ledger, stream window, roundedness row, directedness row, replay row, provenance row, and local name.

Apply componentwise hsame transport through the transport rows. The metric source remains the same source surface, formal-ball membership is compared row by row, dyadic comparison witnesses are preserved, and stream windows are transported with their finite observations.

The roundedness and directedness rows are transported with the family they certify. Hence a smaller displayed ball, or a common displayed refinement, remains a witness for the transported packet.

Replay, provenance, and naming preserve the same route. Since no excluded coordinate is available to either classifier, cofinal refinement stability and ledger non-escape are invariant under the displayed transport. □

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedIdealCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedIdealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1883.1, Definition 1883.2, Definition 1883.3, and Definition 1883.4 seed the rounded directed formal-ball ideal surface for choice-free completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedIdealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, roundedness, directedness, cofinal refinement stability, classifier transport, replay stability, provenance, local naming, and ledger non-escape for the same packet.

Constructive story: A RoundedIdealCompletion$^{\uparrow}$ packet is generated from a metric source, a finite FormalBall family, a DyadicRatCore radius ledger, StreamName windows, roundedness and directedness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1884

A Concrete Naming Certificate for $\mathsf{CompleteBaireSpace}^{\uparrow}$

Following Brouwer's spread semantics and the Baire-space practice of Troelstra–van Dalen and Bishop–Bridges, the $\mathsf{CompleteBaireSpace}^{\uparrow}$ packet records the complete-prefix-stream object by finite prefix data rather than by a host sequence space. It refines the $\mathsf{BaireSpace}^{\uparrow}$ carrier of chapter 334, reads the prefix metric rows of $\mathsf{BaireMetric}^{\uparrow}$ from chapter 1602, uses the finite-window interface of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, keeps rational readback on the $\mathsf{RegSeqRat}^{\uparrow}$ route of chapter 301, and exposes its completion boundary through $\mathsf{UniformCompletion}^{\uparrow}$ in chapter 1473 or $\mathsf{HausdorffCompletion}^{\uparrow}$ in chapter 402. The object is a concrete L10 completion surface for prefix streams; it is not countable choice, quotient equality of streams, a selected branch principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1884.1 (CompleteBaireSpace carrier)**.** *A* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (B, M, S, R, L, T, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row, naming the inspected natural-number prefixes and their extension order.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *prefix-metric row, recording the finite common-prefix comparisons and dyadic radii used by the carrier.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that schedules which prefix coordinates and metric comparisons are available to a consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows, dyadic radii, and prefix-metric comparisons.*
5. *L is the completion witness row, displayed as a* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{HausdorffCompletion}^{\uparrow}$ *handoff for prefix-Cauchy streams and not as an ambient complete-space theorem.*
6. *T is the prefix-limit trace ledger, recording the finite route from compatible prefix windows through the metric rows and regular readback into the completion witness.*
7. *H records componentwise* hsame *transport for prefix data, metric rows, stream windows, readback rows, completion witnesses, trace rows, continuation rows, provenance, and local naming.*

8. *C records displayed* Cont *replay, so a consumer rereads $B, M, S, R, L, T, H, P, N$ without changing the prefix-stream packet.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteBaireSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteBaireSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $B, M, S, R, L, T, H, C, P, N$. It has no coordinate for countable choice, quotient stream equality, a selected infinite branch, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompleteBaireSpaceUp.CompleteBaireSpaceTasteGate single carrier alignment`

Theorem 1884.2 (CompleteBaireSpace NameCert obligations)**.** *For every accepted* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *carrier*

$$X = (B, M, S, R, L, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteBaireSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Baire prefix row, prefix metric row, stream-window row, regular rational readback, completion witness, trace ledger, transport, replay, provenance, and local naming;*
2. *prefix-metric exactness, requiring every accepted comparison to read the same displayed* $\mathsf{BaireSpace}^{\uparrow}$ *prefixes through the* $\mathsf{BaireMetric}^{\uparrow}$ *row M;*
3. *window-readback exactness, requiring every consumer-visible prefix request to pass through the* $\mathsf{StreamName}^{\uparrow}$ *row S and then the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. *completion-witness stability, requiring the prefix-Cauchy handoff to use only the displayed* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{HausdorffCompletion}^{\uparrow}$ *row L and trace ledger T;*
5. *nonescape, excluding countable choice, quotient stream equality, selected branches, ambient Real completeness, Lean verification, and bridge checking.*

No CompleteBaireSpace obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1884.1. The accepted tuple displays exactly the prefix row, prefix metric row, finite stream-window row, regular rational readback, completion witness, trace ledger, transport, replay, provenance, and local name.

Prefix-metric exactness is forced by B and M. The only prefix data are stored in B, and the only comparison row is M, so every accepted metric comparison is a displayed common-prefix comparison of the same finite Baire-space packet.

Window-readback exactness is forced by S and R. A consumer first reads the scheduled finite prefix windows in S and then reads the corresponding regular rational tolerance and radius comparisons in R.

Completion-witness stability is the pair L, T. The completion row is displayed as a uniform or Hausdorff completion handoff for prefix-Cauchy streams, while the trace ledger records how finite compatible windows reach that handoff.

The structural rows H, C, P, N transport, replay, source, and name only the displayed packet. Since the excluded choice, quotient, selected-branch, and ambient completeness data are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1884.3 (CompleteBaireSpace prefix-Cauchy limit extraction)**.** *Let $X = (B, M, S, R, L, T, H, C, P, N)$ be an accepted* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *carrier. Every prefix-Cauchy consumer read of X extracts its completion-facing limit witness through the displayed route*

$$\mathsf{BaireSpace}^{\uparrow}(B) \longrightarrow \mathsf{BaireMetric}^{\uparrow}(M) \longrightarrow \mathsf{StreamName}^{\uparrow}(S) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow L.$$

The extraction uses finite compatible prefixes, prefix-metric comparisons, scheduled windows, and regular rational readback before reaching the completion witness. It does not use a selected infinite branch, quotient equality of streams, countable choice, or ambient Real completeness.

Proof. Begin with a prefix-Cauchy consumer read. By the carrier definition, the only source of stream data is the finite prefix row B, and the only metric evidence available for Cauchy comparison is the prefix-metric row M.

The scheduled request is then read through S. This step restricts the consumer to the displayed finite StreamName window rather than to an arbitrary host stream of choices.

The rational tolerance and radius readback is R. Thus the prefix-Cauchy claim is inspected only through the same finite windows and the same $\mathsf{RegSeqRat}^{\uparrow}$ regular readback row.

Finally the completion witness is L, with T recording the route from compatible prefix windows to that witness. Componentwise transport and replay in H and C preserve the same rows, while P and N bind provenance and local naming.

No row in the packet contains a selected branch, quotient stream equality, countable choice, or ambient Real completeness theorem. Therefore every accepted limit extraction factors through the displayed prefix-Cauchy route. □

Theorem 1884.4 (CompleteBaireSpace prefix-window stability)**.** *Let $X = (B, M, S, R, L, T, H, C, P, N)$ and $X' = (B', M', S', R', L', T', H', C', P', N')$ be accepted* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *carriers. Suppose the displayed* $\mathsf{BaireSpace}^{\uparrow}$ *prefix rows,* $\mathsf{BaireMetric}^{\uparrow}$ *prefix-metric rows,* $\mathsf{StreamName}^{\uparrow}$ *window rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows agree componentwise under* hsame *and replay through displayed* Cont *routes. Then the completion witness rows L, L' and trace ledgers T, T' transport to the same finite prefix-Cauchy route.*

Proof. Project both carriers through Definition 1884.1. The completion-facing data can only be reached after the rows B, M, S, R and B', M', S', R' have been displayed.

Use Definition 1884.3. It forces every prefix-Cauchy consumer read through the route B, M, S, R, L for X and through the corresponding route B', M', S', R', L' for X'.

The componentwise hsame hypotheses identify the displayed prefix rows, prefix-metric rows, StreamName windows, and regular readback rows. The displayed Cont replays keep those reads on the same finite windows, so the two prefix-Cauchy routes have the same visible source.

The completion witness and trace ledger are downstream of that source. Repack the transported rows with L, T, H, C, P, N and L', T', H', C', P', N'; the resulting completion witness rows and trace ledgers therefore describe the same finite prefix-Cauchy route. □

Theorem 1884.5 (CompleteBaireSpace Real-completion boundary)**.** *For an accepted* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *carrier $X = (B, M, S, R, L, T, H, C, P, N)$, every Real or Completion consumer route is exhausted by the displayed prefix packet*

$$B, M, S, R, L, T, H, C, P, N.$$

Such a consumer may inspect finite Baire prefixes, prefix-metric comparisons, StreamName windows, RegSeqRat readback, the displayed uniform or Hausdorff completion witness, componentwise transport, continuation replay, provenance, and local naming. It may not replace that route by

countable choice, quotient streams, selected branches, hidden limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The Real and Completion-facing rows are S, R, L, T together with the source prefix and metric rows B, M.

The finite source route is fixed before completion is read. The carrier requires the consumer to see the Baire prefix row, then the prefix metric row, then the scheduled finite StreamName windows and RegSeqRat readback.

The completion boundary is the displayed row L. It is either a uniform-completion or Hausdorff-completion handoff for the same prefix-Cauchy stream data, and T records that finite route inside the packet.

The remaining rows do not add analytic strength. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same prefix packet.

Because no carrier coordinate stores countable choice, quotient stream equality, a selected branch, a hidden limit selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every accepted Real or Completion consumer remains inside the displayed boundary. □

Falsifiable prediction: Every $\mathsf{CompleteBaireSpace}^{\uparrow}$ read factors through one finite $\mathsf{BaireSpace}^{\uparrow}$ prefix row, one $\mathsf{BaireMetric}^{\uparrow}$ prefix-metric row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one displayed completion witness, one trace ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bairespace, bairemetric, streamname, regseqrat, uniformcompletion, hausdorffcompletion: $\mathsf{CompleteBaireSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries prefix-stream data, prefix-metric comparisons, finite window scheduling, regular rational readback, a completion witness, and a Real-completion refusal boundary as one local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteBaireSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteBaireSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1884.1, Definition 1884.2, Definition 1884.3, and Definition 1884.5 seed the complete prefix-stream route from Baire prefixes through prefix metrics, finite StreamName windows, RegSeqRat readback, displayed completion handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteBaireSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient stream equality, selected branches, hidden limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-metric exactness, window-readback exactness, completion-witness stability, prefix-Cauchy limit extraction, and Real-completion boundary proofs for BEDC.Derived.CompleteBaireSpace

Constructive story: A $\mathsf{CompleteBaireSpace}^{\uparrow}$ packet is generated from finite Baire prefix data, a BaireMetric prefix-metric row, StreamName finite windows, RegSeqRat readback, a UniformCompletionUp or HausdorffCompletionUp handoff, a trace ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1885

A Concrete Naming Certificate for $\mathsf{LocatedIntervalCover}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalCover}^{\uparrow}$ packet records a finite cover of a displayed located real interval by rational cover rows and dyadic refinement rows. It sits after the source interval surface of chapter 1709, the finite rational cover surface of chapter 1882, the dyadic mesh route of chapter 1428, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers one located-interval cover boundary; it is not an open-cover compactness theorem, a countable-choice construction, quotient equality of real names, selected-limit data, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1885.1 (LocatedIntervalCover carrier)**.** *A* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (I, Q, D, R, W, S, H, K, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, supplying the endpoint, radius, locator, locatedness, and metric-bracket data for the interval being covered.*
2. *Q is the finite* $\mathsf{RationalIntervalCover}^{\uparrow}$ *family whose rational interval rows, coverage ledger, and regular-window boundary are accepted for the source interval.*
3. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *mesh and refinement row, recording the dyadic scale, radius ledger, finite membership rows, and visible real-name seal used to test the cover.*
4. *R is the finite refinement ledger comparing the rational cover family Q with the dyadic mesh row D over the located source I.*
5. *W is the StreamName and* $\mathsf{RegSeqRat}^{\uparrow}$ *window boundary, naming the finite stream windows and regular rational readbacks through which the cover is inspected.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the located interval, rational cover, dyadic mesh, refinement, and window rows have been replayed.*
7. *H records componentwise* hsame *transport for the source interval, rational cover family, dyadic cover row, refinement ledger, window boundary, terminal seal, replay, provenance, and local naming.*

8. *K records displayed* Cont *replay, so a consumer can reread $I, Q, D, R, W, S, H, P, N$ without changing the cover route.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RationalIntervalCover}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalCover}^{\uparrow}}$ *row.*

The classifier compares accepted located interval covers only through the displayed rows $I, Q, D, R, W, S, H, K, P, N$. It has no coordinate for arbitrary open covers, countable cover selection, quotient real equality, selected limits, or ambient Real completeness.

Theorem 1885.2 (LocatedIntervalCover NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *carrier*

$$C = (I, Q, D, R, W, S, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located source interval, rational cover family, dyadic cover mesh, refinement ledger, window boundary, terminal seal, transports, replay, provenance, and local naming;*
2. *source-interval exactness, requiring every endpoint, radius, locator, locatedness, and metric-bracket request to pass through the displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row I;*
3. *cover-family exactness, requiring every rational interval, finite coverage read, and retained regular-window request to pass through the displayed* $\mathsf{RationalIntervalCover}^{\uparrow}$ *family Q;*
4. *refinement-route exactness, requiring every dyadic mesh read, finite membership row, and rational-to-dyadic narrowing to pass through D and the ledger R;*
5. *window and readback ordering, requiring W to expose StreamName windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks before the terminal* $\mathsf{Real}^{\uparrow}$ *seal S is available;*
6. *ledger non-escape, requiring H, K, P, N to export no open-cover compactness, countable choice, quotient real equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No located interval cover obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1885.1. The accepted packet displays exactly one located interval source, one rational cover family, one dyadic cover row, one refinement ledger, one window boundary, one terminal seal, and the structural rows that transport, replay, package, and name them.

Source-interval exactness follows because I is the only coordinate containing $\mathsf{LocatedRealInterval}^{\uparrow}$ data. Endpoint, radius, locator, locatedness, and metric-bracket reads therefore enter the cover packet through the displayed source interval and cannot be replaced by a host closed interval.

Cover-family exactness is supplied by Q. Its rational interval rows, coverage ledger, and regular-window boundary are the only rational-cover coordinates available to the packet, so every rational cover read is a finite projection from Q.

Refinement-route exactness is supplied jointly by D and R. The dyadic mesh, radius, and membership rows are read from D, while the comparison between those rows and the rational family Q is read from the finite ledger R.

The remaining rows W, S, H, K, P, N impose the readback order. StreamName windows and regular rational readbacks are exposed before the terminal real-name seal, and the transport, replay, provenance, and naming rows contain no excluded compactness, choice, quotient, limit, or completeness coordinate. □

Lean4 ✓ `BEDC.Derived.LocatedIntervalCoverUp.TasteGate.LocatedIntervalCoverTasteGate single carrier alignment`

Theorem 1885.3 (LocatedIntervalCover refinement route)**.** *Every located interval cover read of an accepted* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *carrier factors through the displayed route*

$$I \longmapsto Q, D, R \longmapsto W, S, H, K, P, N.$$

Thus the consumer first receives the located interval source row, then the finite rational cover family, dyadic mesh, and refinement ledger, and only then the StreamName, $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, transport, replay, provenance, and local naming rows. The route exports no open-cover compactness, countable choice, quotient real equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $C = (I, Q, D, R, W, S, H, K, P, N)$. The only source interval coordinate is I, so every cover request begins with the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ endpoint, radius, locator, locatedness, and metric-bracket data.

The cover rows are then read through Q, D, R. The rational family Q supplies the finite rational cover ledger, the dyadic row D supplies the mesh and membership tests, and the refinement ledger R records how the two surfaces agree over the located source interval.

The window and seal rows are downstream of that refinement surface. The StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ boundary W is read before the terminal $\mathsf{Real}^{\uparrow}$ seal S, so a consumer cannot bypass finite readback by appealing directly to a completed real object.

Finally, H, K, P, N transport, replay, package, and name the same displayed route. Since the accepted packet contains no arbitrary open cover, countable selector, quotient real equality proof, selected limit, or Real completeness row, those objects cannot be exported by the factorization. □

Falsifiable prediction: Every $\mathsf{LocatedIntervalCover}^{\uparrow}$ read factors through one $\mathsf{LocatedRealInterval}^{\uparrow}$ source row, one finite $\mathsf{RationalIntervalCover}^{\uparrow}$ family, one $\mathsf{DyadicIntervalCover}^{\uparrow}$ mesh row, one finite refinement ledger, one StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ window boundary, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real-interval, rational-interval-cover, dyadic-interval-cover, stream-name, regseqrat, real: $\mathsf{LocatedIntervalCover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located source interval, rational cover family, dyadic mesh, refinement ledger, finite readback boundary, terminal real seal, and local NameCert route as one finite cover packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalCover}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1885.1, Definition 1885.2, and Definition 1885.3 seed the finite located interval cover route from a located real interval source through rational cover rows, dyadic refinement rows, finite windows, terminal real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, cover-family exactness, refinement-route exactness, window and readback ordering, ledger non-escape, and finite consumer factorization for BEDC.Derived.LocatedIntervalCoverUp.

Constructive story: A $\mathsf{LocatedIntervalCover}^{\uparrow}$ packet is generated from a $\mathsf{LocatedRealInterval}^{\uparrow}$ source row, a finite $\mathsf{RationalIntervalCover}^{\uparrow}$ family, a $\mathsf{DyadicIntervalCover}^{\uparrow}$ mesh and refinement row, a finite refinement ledger, StreamName and $\mathsf{RegSeqRat}^{\uparrow}$ windows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1886

A Concrete Naming Certificate for $\mathsf{CantorFanBarrier}^{\uparrow}$

Following Brouwer's finite fan arguments, Troelstra–van Dalen's treatment of decidable bars, and Bishop–Bridges constructive analysis, the $\mathsf{CantorFanBarrier}^{\uparrow}$ packet records the finite barrier layer between the fan certificate of chapter 400, the Cantor-space prefix surface of chapter 1475, and the Real and Completion modulus route of chapter 1707. It is a displayed finite packet over branch-prefix histories, StreamName windows, RegSeqRat readback, and a Real seal; it is not an open-cover compactness principle, a selected infinite branch, a quotient stream equality principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1886.1 (Cantor fan barrier carrier)**.** *A* $\mathsf{CantorFanBarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, A, B, D, W, S, Q, R, H, C, P, N)$$

with the following visible rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *row, carrying the finite binary fan surface, prefix closure, decidable finite-bar read, and fan-depth bound.*
2. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *address row, naming the finite Boolean prefix cylinder on which the barrier is inspected without selecting an infinite branch.*
3. *B is the decidable barrier row, assigning a visible* $\mathsf{Bool}^{\uparrow}$ *decision to each displayed prefix in the retained Cantor cylinder.*
4. *D is the unary depth ledger, a finite* $\mathsf{Nat}^{\uparrow}$ *row recording the maximal fan depth and the Real-facing read depth associated with each retained barrier decision.*
5. *W is the finite witness window, listing exactly the prefixes, barrier decisions, and depth entries read from F, A, B, D.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that converts the finite prefix-depth ledger into the source windows inspected by downstream Real and Completion consumers.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed StreamName windows.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after the rows F, A, B, D, W, S, Q have been displayed.*

9. *H stores componentwise* hsame *transport for the fan row, Cantor address, decidable barrier, depth ledger, witness window, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a consumer request through $F, A, B, D, W, S, Q, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CantorFanBarrier}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorFanBarrier}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows $F, A, B, D, W, S, Q, R, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected infinite branches, quotient stream equality, selected Real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorFanBarrierUp.CantorFanBarrierTasteGate single carrier alignment`

Theorem 1886.2 (Cantor fan barrier NameCert obligations)**.** *For every accepted* $\mathsf{CantorFanBarrier}^{\uparrow}$ *carrier*

$$K = (F, A, B, D, W, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorFanBarrier}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed fan, Cantor-address, decidable-barrier, depth-ledger, witness-window, StreamName-window, RegSeqRat-readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *fan coverage, requiring every branch-prefix request to be bounded by the finite fan row F over the Cantor address surface A;*
3. *barrier decidability, requiring every retained prefix in W to read its barrier decision from the finite* $\mathsf{Bool}^{\uparrow}$ *row B;*
4. *depth exactness, requiring the unary ledger D to account for every fan-depth and Real-facing window-depth read used by S, Q, R;*
5. *Real-window handoff, requiring every downstream Real or Completion request to factor through S, then Q, then R;*
6. *transport, replay, provenance, and local naming stability, requiring H, C, P, N to preserve and name exactly the displayed packet.*

No Cantor fan barrier obligation is stored outside these six rows.

Proof. Proceed by structural induction on the displayed unary depth ledger D.

In the zero-depth case, the retained witness window is the root cylinder named by A. The fan row F supplies the root-prefix surface, the barrier decision is the corresponding entry of B, and the StreamName, RegSeqRat, and Real rows are read only through S, Q, R after that finite prefix has been displayed.

For the successor step, assume the obligation surface has been constructed through depth d. A depth $d + 1$ entry adds only the two immediate Boolean extensions of prefixes already present in the Cantor address row. The fan row F bounds these branch-prefix reads, B supplies the two displayed barrier decisions, and W records the finite witness entries attached to the new depth.

The StreamName window row S is then extended only by the windows named by the new ledger entries. RegSeqRat readback through Q and the Real seal through R are downstream of

the same finite schedule, so no quotient stream equality, selected branch, or ambient completeness object is introduced.

Finally, H, C, P, N transport, replay, record, and name the same finite rows. Coordinate exhaustion gives exactly the six obligation rows and excludes open-cover compactness, countable choice, selected infinite branches, quotient stream equality, selected Real limits, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{CantorFanBarrier}^{\uparrow}$ Real or Completion-facing read factors through a finite fan row, Cantor-prefix address row, decidable barrier row, unary depth ledger, finite witness window, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Independence witness: fantheorem, cantorspace, bishop-fan-modulus: $\mathsf{CantorFanBarrier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the fan row, Cantor address row, decidable barrier row, unary fan-depth ledger, finite witness window, StreamName window, RegSeqRat readback, Real seal, and structural support while refusing open-cover compactness, countable choice, selected branches, quotient stream equality, selected Real limits, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorFanBarrier}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorFanBarrier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1886.1 and Definition 1886.2 seed the finite barrier route through $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorFanBarrier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, quotient stream equality, countable choice, selected Real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan coverage, barrier decidability, depth exactness, Real-window handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CantorFanBarrierUp.

Constructive story: A $\mathsf{CantorFanBarrier}^{\uparrow}$ packet is generated from a finite $\mathsf{FanTheorem}^{\uparrow}$ row, $\mathsf{CantorSpace}^{\uparrow}$ address row, decidable $\mathsf{Bool}^{\uparrow}$ barrier row, unary $\mathsf{Nat}^{\uparrow}$ depth ledger, finite witness window, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1887

A Concrete Naming Certificate for $\mathsf{WeaklyCauchySequence}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of Cauchy data, the $\mathsf{WeaklyCauchySequence}^{\uparrow}$ packet records finite weak-Cauchy evidence without turning it into a regular modulus or a located real limit. It sits between the finite StreamName windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal Real boundary of chapter 38, and the Specker-style obstruction surface of chapter 1406. The packet names the finite tail-comparison evidence visible to a BEDC consumer; it does not supply a regular modulus, selected limit, located supremum, quotient stream equality, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1887.1 (Weakly Cauchy sequence carrier)**.** *A* $\mathsf{WeaklyCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the inspected sequence indices and the stream observations available at those indices.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same inspected windows, recording the rational approximants read from the displayed stream data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii used in the inspected pairwise tail comparisons.*
4. *T is the weak-Cauchy tail-comparison ledger: for each displayed tolerance request it stores only the finite pairs of inspected tail indices and the dyadic comparison rows witnessed for those pairs.*
5. *E is an optional* $\mathsf{Real}^{\uparrow}$ *seal-boundary row, present only as a downstream consumer boundary after S, R, D, T have been displayed and never as a completed limit coordinate.*
6. *H records componentwise* hsame *transport for stream windows, rational readbacks, dyadic tolerances, tail-comparison rows, optional Real boundary rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite tolerance query through S, R, D, T and then through E, H, P, N when the optional Real boundary is read.*

8. *P is the* Pkg *provenance over* $\mathsf{WeaklyCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, T, E, H, C, P, N$. It has no coordinate for a regular modulus, selected limit, located supremum, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.WeaklyCauchySequenceUp.WeaklyCauchySequenceTasteGate single carrier alignment`

Theorem 1887.2 (Weakly Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyCauchySequence}^{\uparrow}$ *carrier*

$$W = (S, R, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeaklyCauchySequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, weak tail-comparison ledger, optional Real boundary, transport, replay, provenance, and local naming;*
2. *window exactness, requiring every inspected sequence coordinate to be read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S before any rational readback is used;*
3. *readback exactness, requiring the rational observations used by the tail comparisons to be the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. *tolerance exactness, requiring every pairwise comparison in T to cite a dyadic tolerance row in D;*
5. *weak-tail exposure, requiring T to store only finite inspected tail comparisons and not a global regular modulus;*
6. *boundary nonescape, excluding selected limits, located suprema, quotient stream equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No weak-Cauchy obligation is stored outside these rows.

Proof. Project the carrier of Definition 1887.1. The projection exposes exactly one finite StreamName window row, one RegSeqRat readback row, one dyadic tolerance ledger, one weak tail-comparison ledger, the optional Real boundary row, and the structural transport, replay, provenance, and naming rows.

Window and readback exactness are forced by S and R. Every inspected sequence coordinate is first scheduled by S, and every rational observation used downstream is then read through the corresponding regular rational row R.

Tolerance exactness is forced by D. Each pairwise tail comparison in T names a dyadic radius and comparison row already present in the tolerance ledger, so no hidden metric or host real comparison can be inserted.

The weak-tail obligation is exactly the ledger T. It records finite inspected tail comparisons, but it does not choose a threshold function for all tolerances and it does not package a regular Cauchy modulus.

The rows E, H, C, P, N only preserve or expose the same packet. Since the excluded limit, supremum, quotient, choice, trichotomy, completeness, verification, and bridge coordinates are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1887.3 (Weakly Cauchy finite-window boundary)**.** *Let $W = (S, R, D, T, E, H, C, P, N)$ be an accepted* $\mathsf{WeaklyCauchySequence}^{\uparrow}$ *carrier. Every consumer-visible weak-Cauchy read factors through the finite route*

$$\mathsf{StreamName}^{\uparrow}(S) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D) \longrightarrow T \longrightarrow H, C, P, N.$$

The route may compare finitely inspected tail windows at displayed dyadic tolerances. It cannot be strengthened into a regular modulus, quotient stream equality, selected limit, countable-choice tail selector, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with a consumer-visible weak-Cauchy read. The carrier definition gives S as the only row that schedules sequence observations, so the consumer first receives finite StreamName windows rather than an unrestricted host stream.

The rational observations are then read through R. This keeps every inspected value on the RegSeqRat route named by the packet.

The dyadic scale is read through D, and the actual weak-tail comparison is one of the finite entries of T. Thus the consumer obtains only a finite comparison between displayed tail indices at displayed tolerances.

The structural rows H, C, P, N transport, replay, source, and name that same route. No row produces a global threshold function, quotient equality, selected limit, choice selector, trichotomy decision, or ambient completeness theorem, so every accepted read remains a finite-window boundary read. □

Theorem 1887.4 (Weakly Cauchy regular-modulus refusal)**.** *Let*

$$W = (S, R, D, T, E, H, C, P, N)$$

be an accepted $\mathsf{WeaklyCauchySequence}^{\uparrow}$ *carrier. Any route that exports a* $\mathsf{RegSeqRat}^{\uparrow}$ *regular modulus or a* $\mathsf{Real}^{\uparrow}$ *limit from W must factor through an additional modulus row not contained in $S, R, D, T, E, H, C, P, N$. The displayed weak-tail ledger T alone supplies only inspected pairwise tail comparisons at displayed dyadic tolerances.*

Proof. Project the carrier of Definition 1887.1. Its displayed inventory contains StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, the finite weak-tail ledger, the optional Real boundary row, transport, replay, provenance, and local naming; it contains no coordinate that chooses a threshold function for every tolerance.

Apply Definition 1887.2. The weak-tail exposure obligation identifies T as finite inspected tail evidence and explicitly prevents that ledger from being read as a global regular modulus.

Apply Definition 1887.3. Every consumer-visible read passes through

$$\mathsf{StreamName}^{\uparrow}(S) \longrightarrow \mathsf{RegSeqRat}^{\uparrow}(R) \longrightarrow \mathsf{DyadicRatCore}^{\uparrow}(D) \longrightarrow T,$$

before the structural rows replay or name the route, so a regular-modulus export needs data beyond the displayed finite route.

Finally apply Definition 1887.5. The Specker-facing boundary excludes selected limits, located suprema, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so no $\mathsf{Real}^{\uparrow}$ limit can be obtained from W unless an additional modulus row is supplied outside the accepted carrier. □

Theorem 1887.5 (Weakly Cauchy sequence Specker boundary)**.** *For an accepted* $\mathsf{WeaklyCauchySequence}^{\uparrow}$ *carrier $W = (S, R, D, T, E, H, C, P, N)$, every Specker-style or Real-facing consumer route is exhausted by the displayed rows*

$$S, R, D, T, E, H, C, P, N.$$

In particular, the packet may expose finite weak-Cauchy tail comparisons before a $\mathsf{Real}^{\uparrow}$ *boundary row is read, but it does not convert those comparisons into a located supremum, a regular Cauchy modulus, a selected limit, quotient stream equality, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The Specker-facing information is the finite schedule S, the rational readback row R, the dyadic ledger D, and the weak tail-comparison ledger T.

A Real-facing read can only use E after those finite rows have been displayed. The row E is therefore a boundary row for consumers of the evidence, not an independent completed real or located supremum coordinate.

The structural rows H, C, P, N do not add analytic strength. They preserve componentwise equality, replay the displayed continuation route, record provenance, and name the local certificate.

The Specker obstruction remains visible because the packet stops at finite weak-Cauchy evidence. Since no carrier coordinate stores a regular modulus, selected limit, located supremum, quotient stream equality, countable choice, trichotomy, or ambient completeness theorem, every accepted Specker-style or Real-facing route is exhausted by the displayed boundary. □

Falsifiable prediction: Every $\mathsf{WeaklyCauchySequence}^{\uparrow}$ read factors through finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a finite weak tail-comparison ledger, optional RealUp boundary exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, speckersequence: $\mathsf{WeaklyCauchySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inspected sequence windows, regular rational readback, dyadic tolerance comparisons, finite weak-tail evidence, and a Specker-facing Real boundary refusal as one local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{WeaklyCauchySequence}^{\uparrow}$ packet is generated from inspected StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, finite weak-Cauchy tail comparisons, an optional RealUp boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1887.1, Definition 1887.2, Definition 1887.3, Definition 1887.4, and Definition 1887.5 seed the finite weak-Cauchy boundary over inspected StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, finite tail comparisons, RealUp boundary exposure, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeaklyCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close regular moduli, selected limits, located suprema, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, readback exactness, tolerance exactness, weak-tail exposure, finite-window boundary, regular-modulus refusal, and Specker boundary proofs for BEDC.Derived.WeaklyCauchySequenceUp.

Chapter 1888

A Concrete Naming Certificate for $\mathsf{PicardLindelof}^{\uparrow}$

Following Picard and Lindelöf, the $\mathsf{PicardLindelof}^{\uparrow}$ packet records the local constructive existence-and-uniqueness route for ordinary differential equations with a Lipschitz vector field. It consumes the local-flow source of chapter 193, the Banach state classifier of chapter 90, the map-level Lipschitz ledger of chapter 410, the complete-metric convergence boundary of chapter 128, the finite iteration route of chapter 395, and the fixed-point endpoint discipline of chapter 396. The packet names a local time-state window and a displayed Picard iteration certificate; it is not a global ODE solver, a maximal-interval continuation theorem, a host function-space equality principle, or a choice-selected solution object.

Definition 1888.1 (Picard–Lindelof carrier)**.** *A* $\mathsf{PicardLindelof}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

with the following visible coordinates:

1. *I is the initial time-state cell, recording a displayed time row t_0, a state row x_0, and the shared* $\mathsf{Ode}^{\uparrow}$ *local-flow source in which both rows are interpreted;*
2. *F is the Lipschitz vector-field ledger, a* $\mathsf{LipschitzMap}^{\uparrow}$ *row for the vector field on the named local time-state window, including the displayed bound and graph rows used by ODE evaluations;*
3. *L is the Picard iteration route, a* $\mathsf{PicardContraction}^{\uparrow}$ *ledger that records the finite sequence of successive integral or step approximants starting at I and reading every vector-field value through F;*
4. *R is the complete-metric convergence row, supplied by* $\mathsf{CompleteMetric}^{\uparrow}$ *for the carried space of local approximants and containing the displayed Cauchy modulus, limit row, and late-observation comparisons for the Picard route;*
5. *Q is the local uniqueness classifier, recording that two displayed local-flow rows with the same initial cell and the same Lipschitz ledger are classified by the endpoint classifier inherited from* $\mathsf{Ode}^{\uparrow}$ *and the Banach state surface;*
6. *U is the nonescape window row, naming the local time interval and state ball in which the Picard iterates, the convergence limit, and the uniqueness comparison are all read;*

7. *H stores componentwise* hsame *transports for the initial cell, vector-field ledger, Picard route, convergence row, uniqueness classifier, nonescape window, replay rows, provenance row, and local naming row;*
8. *C stores the displayed* Cont *replay from the initial time-state cell through the vector-field ledger, Picard iteration route, complete-metric convergence row, uniqueness classifier, and local-flow endpoint;*
9. *G is the* Pkg *provenance row over* $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardLindelof}^{\uparrow}}$ *row.*

The carrier exposes only a local NameCert route from displayed ODE and Lipschitz data to a classified local-flow endpoint. It has no coordinate for global existence, maximal continuation, host equality of solution functions, selected representatives of solutions, or bridge-checked standardization.

Lean4 ✓ `BEDC.Derived.PicardLindelofUp.PicardLindelofTasteGate single carrier alignment`

Theorem 1888.2 (Picard–Lindelof NameCert obligations)**.** *For an accepted* $\mathsf{PicardLindelof}^{\uparrow}$ *carrier*

$$P = (I, F, L, R, Q, U, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PicardLindelof}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, initial time-state compatibility for I, Lipschitz vector-field stability for F, Picard iteration compatibility for L, complete-metric convergence visibility for R, local uniqueness classification for Q, nonescape-window stability for U, componentwise transport through H, displayed replay through C, package provenance through G, and local naming through N.*

Proof. Carrier admission is projection from Definition 1888.1. Each accepted packet displays the initial cell, vector-field ledger, Picard route, convergence row, uniqueness classifier, nonescape window, transport, replay, provenance, and local name.

The source-side obligations are local to I, F, U. The row I fixes the $\mathsf{Ode}^{\uparrow}$ time-state source, the row F supplies the $\mathsf{LipschitzMap}^{\uparrow}$ vector-field graph and bound rows on the named window, and U records the finite nonescape boundary in which those reads are valid.

The iterative and limiting obligations are then forced in order. The row L reads only the displayed Picard route supplied by $\mathsf{PicardContraction}^{\uparrow}$, and the row R reads the resulting regular Cauchy ledger through $\mathsf{CompleteMetric}^{\uparrow}$.

Finally Q classifies any competing local-flow row with the same initial cell and vector-field ledger. The rows H, C, G, N transport, replay, source, and name this same route, so the certificate cannot export a global continuation, a host function-space equality, or a selected solution. □

Theorem 1888.3 (Picard–Lindelof Picard iteration route)**.** *Every accepted* $\mathsf{PicardLindelof}^{\uparrow}$ *carrier factors each local-flow endpoint read through the finite route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{Banach}^{\uparrow} \longmapsto \mathsf{LipschitzMap}^{\uparrow} \longmapsto \mathsf{PicardContraction}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{BanachFixedPoint}^{\uparrow}$$

as instantiated by I, F, L, R, Q, U. The route first opens the ODE time-state source and Banach state classifier, then reads the Lipschitz vector-field ledger, then records the Picard iteration route, then reads the complete-metric convergence row, and only then exposes the local uniqueness classifier for the endpoint. No endpoint read bypasses these rows.

Proof. Read the carrier coordinates. The initial cell I is the only entrance to the $\mathsf{Ode}^{\uparrow}$ local-flow source, and the vector-field row F is the only Lipschitz ledger available on the named window.

The Picard approximants are supplied by L. Since L records the $\mathsf{PicardContraction}^{\uparrow}$ route using evaluations from F, every displayed iterate remains attached to the same ODE source and nonescape window.

The limiting step is supplied by R. It reads the Picard ledger as a regular Cauchy object in the carried complete metric and exposes the limit only through the displayed $\mathsf{CompleteMetric}^{\uparrow}$ convergence row.

The uniqueness boundary is supplied by Q and may be compared with the fixed-point endpoint discipline of $\mathsf{BanachFixedPoint}^{\uparrow}$ only after the previous rows have been displayed. Because the carrier lacks global continuation, host function-space equality, and choice-selected solution coordinates, the theorem exports only the stated local route. □

Theorem 1888.4 (Picard–Lindelof local existence handoff)**.** *Let*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

be an accepted $\mathsf{PicardLindelof}^{\uparrow}$ *carrier. A local-flow endpoint is exported only after the reader has opened the initial cell I, the Lipschitz vector-field ledger F, the Picard iteration route L, the complete-metric convergence row R, the nonescape window U, the componentwise transport row H, the displayed replay row C, the provenance row G, and the local naming row N. The endpoint is then classified by Q as a local endpoint row, without adding global continuation, maximal-interval data, host function-space equality, or a selected solution representative.*

Proof. Project the accepted carrier through Definition 1888.1. This exposes the initial cell, vector-field ledger, Picard route, convergence row, classifier, nonescape window, transport, replay, provenance, and local name.

Use Definition 1888.3 for L. The local endpoint read must pass through the displayed Picard iteration route, so the endpoint cannot be detached from the initial time-state cell or the Lipschitz vector-field ledger.

The convergence and locality checks are then read from R and U. The row R supplies the complete-metric convergence visibility for the Picard approximants, while U records the finite nonescape boundary in which the endpoint is read.

Finally expose Q only as the classifier row. The structural rows H, C, G, N transport, replay, source, and name the same local route, and no row supplies global continuation or a selected solution object. □

Theorem 1888.5 (Picard–Lindelof Gronwall–Osgood nonescape stability)**.** *Let*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

be an accepted $\mathsf{PicardLindelof}^{\uparrow}$ *carrier. If the Lipschitz ledger F is accompanied by a displayed* $\mathsf{GronwallInequality}^{\uparrow}$ *comparison row and the uniqueness classifier Q is accompanied by a displayed* $\mathsf{OsgoodCriterion}^{\uparrow}$ *modulus row, then every endpoint read classified by Q remains inside the same nonescape window U. The exported endpoint still factors through $I, F, L, R, Q, U, H, C, G, N$ and through the comparison rows; it does not introduce a host solver, a selected solution curve, an ambient completeness assertion, or a global continuation coordinate.*

Proof. Use Definition 1888.3. The endpoint read first passes through the initial cell, Lipschitz vector-field ledger, Picard iteration route, complete-metric convergence row, and nonescape window U before the classifier Q is exposed.

Apply Definition 2034.3 to the displayed Lipschitz ledger. The Gronwall row bounds the finite comparison envelope inside the same Picard-facing route, so it can tighten the read within U but cannot replace U by a new ambient interval or a host solution object.

Apply Definition 6159.3 to the uniqueness modulus. The Osgood row constrains competing endpoint reads through the same displayed ODE and Picard route, so the uniqueness comparison remains attached to the nonescape window already carried by P.

Finally replay the structural rows H, C, G, N. They transport, replay, source, and name the same finite certificate, and no row supplies a selected solution, host solver, ambient completeness principle, or continuation beyond U. □

Theorem 1888.6 (Picard–Lindelof Gronwall uniqueness handoff)**.** *Let*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

be an accepted $\mathsf{PicardLindelof}^{\uparrow}$ *carrier. If two displayed endpoint reads share the same initial cell I and the same Lipschitz ledger F, and the displayed* $\mathsf{GronwallInequality}^{\uparrow}$ *comparison row bounds their finite difference envelope inside the nonescape window U, then the local uniqueness classifier Q classifies the two endpoint reads as the same local endpoint of the packet. The handoff uses only $I, F, L, R, Q, U, H, C, G, N$ and the Gronwall comparison row; it does not add global continuation, a selected solution curve, host function-space equality, or an ambient completeness claim.*

Proof. Project the carrier rows of Definition 1888.1. The row Q is the only local uniqueness classifier, and it is keyed by the same initial cell I and Lipschitz vector-field ledger F named in the theorem statement.

Apply Definition 1888.5. The Gronwall comparison row keeps the finite difference envelope for the two displayed endpoint reads inside U, so the comparison remains within the same local time-state window carried by P.

Now read the uniqueness row Q. Because both endpoint reads share I and F, and the comparison envelope is bounded inside U, Q classifies them through the displayed ODE endpoint classifier rather than through a host function-space equality.

Finally transport and replay the classification through H and C, with provenance and naming supplied by G and N. These support rows do not add a selected solution curve, global continuation, or ambient completeness coordinate, so the uniqueness handoff stays local to the packet. □

Theorem 1888.7 (Picard–Lindelof local uniqueness induction)**.** *Let*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

be an accepted $\mathsf{PicardLindelof}^{\uparrow}$ *carrier, and let $W_0, \ldots, W_m$ be a finite list of Picard windows displayed by the route L inside the nonescape row U. If the initial window W_0 reads the same initial cell I and Lipschitz ledger F, and each successor window is obtained by the displayed* $\mathsf{PicardContraction}^{\uparrow}$ *step recorded in L, then every endpoint window W_i is classified by the same local classifier Q after the route has passed through L, R, H, C, G, N. The induction preserves the shared I, F, U rows at every stage and exports no host function equality, global solver, maximal continuation, or selected solution curve.*

Proof. The base window is W_0. By the carrier definition, W_0 is read from the initial cell I, the Lipschitz vector-field ledger F, and the nonescape window U before any endpoint classifier is visible.

For the successor step, use Definition 1888.3. The PicardContraction row L is the only available step rule, and every step in L reads vector-field values through F while remaining inside the same local window U.

The finite induction therefore preserves the three shared rows I, F, U along the whole displayed list. At each stage, the convergence-facing data is still the same complete-metric row R, and the support rows H, C, G, N transport, replay, source, and name the same packet.

Apply Definition 1888.6 at the endpoint stage. Since the endpoint read still shares I and F and remains inside U, the classifier Q supplies the local uniqueness read without introducing a host equality of functions, a global solver coordinate, or any continuation beyond the displayed finite windows. □

Theorem 1888.8 (Picard–Lindelof Cauchy completion route)**.** *Every completion-facing endpoint read of an accepted* $\mathsf{PicardLindelof}^{\uparrow}$ *carrier*

$$P = (I, F, L, R, Q, U, H, C, G, N)$$

must pass from the Picard route L through the complete-metric row R before the local uniqueness classifier Q is exposed. The displayed Cauchy modulus is stored only in the finite rows of L and R, is replayed only through H, C, G, N, and is not a claim of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit representative, a quotient equality, or a global ODE solution.*

Proof. Start with the Picard route. By Definition 1888.3, an endpoint read first opens I, F, L, U and reads each approximant through the $\mathsf{PicardContraction}^{\uparrow}$ sibling route of chapter 395.

The completion-facing read then enters R. The complete-metric row is the only carrier coordinate that can expose Cauchy convergence data, and it is supplied through the $\mathsf{CompleteMetric}^{\uparrow}$ sibling route of chapter 128.

Only after this route may Q classify the endpoint. The support rows H, C, G, N keep the Cauchy modulus attached to the displayed finite packet, so the public endpoint remains a local NameCert read rather than a selected limit, an ambient completeness theorem, or a global solution object. □

Falsifiable prediction: Every $\mathsf{PicardLindelof}^{\uparrow}$ local endpoint read factors through one initial time-state cell, one Lipschitz vector-field ledger, one PicardContraction route, one CompleteMetric convergence row, one local uniqueness classifier, one nonescape window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ode, lipschitzmap, completemetric, picardcontraction, banachfixedpoint: $\mathsf{PicardLindelof}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the initial time-state cell, vector-field Lipschitz ledger, Picard iteration route, complete-metric convergence row, local uniqueness classifier, nonescape window, and the transport/replay/provenance rows tying them together.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardLindelof}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardLindelof}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1888.1, Definition 1888.2, and Definition 1888.3 seed the local constructive ODE existence-and-uniqueness route over an initial time-state cell, a Lipschitz vector-field ledger, a Picard iteration route, a complete-metric convergence row, a local uniqueness classifier, a nonescape window, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardLindelof}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, maximal interval continuation, host function-space equality, choice-selected solutions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, initial time-state compatibility, Lipschitz vector-field stability, Picard iteration compatibility, complete-metric convergence visibility, local uniqueness classification, nonescape-window stability, transport, replay, provenance, and local naming for BEDC.Derived.PicardLindelofUp.

Constructive story: A PicardLindelof$^{\uparrow}$ packet is generated from one initial time-state cell, one Lipschitz vector-field ledger, one PicardContraction route, one CompleteMetric convergence row, one local uniqueness classifier, one nonescape window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1889

A Concrete Naming Certificate for $\mathsf{UniformCauchySequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformCauchySequence}^{\uparrow}$ packet records the finite NameCert surface for one regular rational sequence whose Cauchy comparisons are governed by a shared modulus schedule. It sits on the L10 route from the stream-window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the explicit threshold surface of chapter 332, and the terminal seal boundary of chapter 38. The packet names the sequence-level carrier consumed by completion routes; it does not select a limit, quotient streams, host real equality, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 1889.1 (UniformCauchySequence carrier)**.** *A* $\mathsf{UniformCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, Q, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window surface for the sequence observations;*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assigning rational names to the displayed stream windows;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every displayed Cauchy comparison;*
4. *M is the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *shared threshold row that binds all late-window comparisons;*
5. *R is the finite uniform Cauchy comparison row, reading pairs of sufficiently late windows through S, Q, D, M;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the shared modulus row and comparison row have been displayed;*
7. *H records* hsame *transport for the stream windows, regular readbacks, dyadic ledger, shared modulus, comparison row, real seal, replay row, provenance row, and local naming row;*
8. *C records displayed* Cont *replay from stream inspection through regular readback, dyadic tolerance, shared threshold, comparison, and real-seal delivery;*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySequence}^{\uparrow}}$ *row.*

The carrier stores no quotient stream, no selected host limit, no host equality of real names, no countable-choice selector, and no ambient completeness field.

Theorem 1889.2 (UniformCauchySequence NameCert obligations)**.** *For an accepted* $\mathsf{UniformCauchySequence}^{\uparrow}$ *carrier* $U = (S, Q, D, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySequence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed stream windows, regular rational readbacks, dyadic tolerance ledger, shared modulus row, comparison row, real seal, transports, replay route, provenance, and local naming row;*
2. *shared-modulus exactness, requiring every late-window Cauchy comparison in* R *to use the same* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *threshold row* M *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*;*
3. *readback compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *to be the only rational readback consumed by the comparison row* R*;*
4. *route ordering, requiring* R *to be displayed only after* S, Q, D, M *and requiring* E *to be displayed only after* R*;*
5. *transport stability, requiring* H *and* C *to preserve the route* S, Q, D, M, R, E, P, N *without adding a hidden threshold, independent rational stream, quotient stream, or selected limit;*
6. *ledger exhaustion, stating that no quotient stream, selected host limit, host real equality, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle is exported.*

Proof. Carrier admission is projection from Definition 1889.1. The tuple displays the stream window row, regular readback row, dyadic tolerance ledger, shared modulus row, comparison row, real seal, transport, replay, provenance, and local name.

Shared-modulus exactness follows from the single coordinates M and D. Since every comparison row in R is read through those coordinates, the packet has no second threshold schedule and no hidden tolerance ledger.

Readback compatibility and route ordering are forced by the displayed coordinate order. The comparison row R consumes S, Q, D, M, while the real seal E is downstream of R.

Transport stability and ledger exhaustion are the role of H, C, P, N. These rows replay and name the same finite route, so they cannot introduce quotient streams, selected limits, host real equality, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.UniformCauchySequenceUp.TasteGate.UniformCauchySequenceTasteGate single carrier alignment`

Theorem 1889.3 (UniformCauchySequence RegSeqRat handoff)**.** *Let* $U = (S, Q, D, M, R, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformCauchySequence}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by a Cauchy comparison in* R *factors through the visible handoff*

$$S, Q, D, M, R, H, C, P, N.$$

In particular, a comparison cannot replace Q *by an independent rational stream, cannot replace* M *by a local threshold choice, and cannot replace* D *by a hidden host tolerance.*

Proof. Project the carrier coordinates. The only regular rational readback coordinate is Q, and the only shared threshold coordinate is M.

Apply the readback compatibility and shared-modulus clauses of Definition 1889.2. They force every comparison in R to read Q together with S, D, M before the comparison is accepted.

Finally use the transport rows H and C. They preserve the same displayed route with provenance P and local name N, so the handoff has no access to an independent rational stream, local threshold choice, or hidden host tolerance. □

Theorem 1889.4 (UniformCauchySequence Real seal boundary)**.** *For an accepted* $\mathsf{UniformCauchySequence}^{\uparrow}$ *carrier* $U = (S, Q, D, M, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal read reaches* E *only after passing through the finite stream row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *shared threshold* M*, and uniform Cauchy comparison row* R*. The seal boundary exports no quotient streams, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ seal read of U. By the carrier definition, the only terminal seal coordinate is E.

The NameCert obligations place E downstream of the comparison row R, and R downstream of S, Q, D, M. Hence the seal read must traverse the stream windows, regular readback, dyadic tolerance ledger, shared modulus, and comparison row first.

The rows H, C, P, N preserve and name the same route. Since the carrier lists no quotient-stream coordinate, selected-limit coordinate, countable-choice selector, host real equality, or ambient completeness row, the real seal boundary is exhausted by the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchySequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1889.1, Definition 1889.2, Definition 1889.3, and Definition 1889.4 seed the L10 shared-modulus sequence packet over StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, ModulusOfConvergence thresholds, and the Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, readback compatibility, route ordering, transport stability, ledger exhaustion, and Real seal nonescape for BEDC.Derived.UniformCauchySequenceUp.

Constructive story: A $\mathsf{UniformCauchySequence}^{\uparrow}$ packet is generated from $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{ModulusOfConvergence}^{\uparrow}$ shared threshold row, finite comparison rows, a $\mathsf{Real}^{\uparrow}$ seal boundary, hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1890

A Concrete Naming Certificate for $\mathsf{RealOrderTopology}^{\uparrow}$

Following Kelley, *General Topology*, and Bishop–Bridges constructive analysis, the $\mathsf{RealOrderTopology}^{\uparrow}$ packet records the finite order-topology route on BEDC real names. It binds the real-name carrier of chapter 38, the comparison rows of chapter 766, the generated-open surface of chapter 88, the ball-basis packet of chapter 1450, and the observable-open truth interface of chapter 1860. The packet gives later L10 consumers a BEDC-native topology interface for $\mathsf{Real}^{\uparrow}$ without treating host subsets, completeness, trichotomy, Dedekind cuts, or open-cover compactness as primitive structure.

Definition 1890.1 (Real order topology carrier)**.** *A* $\mathsf{RealOrderTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, A, B, S, T, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *point-name surface, exposing only finite real-name windows and their admitted seal rows.*
2. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison row used to read rational and real-name order probes against the displayed point windows.*
3. *B is the interval-or-ball basis row, recording finite rays, bounded intervals, and* $\mathsf{MetricBall}^{\uparrow}$ *cells that are generated from the comparison service.*
4. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row, recording which displayed point-window probes certify membership in a basic open.*
5. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open row, closing only under the finite generated-open operations displayed by B and S.*
6. *H records componentwise* hsame *transport for the real point rows, comparison rows, basis rows, observable truth rows, generated-open rows, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a real-name observation through comparison, basis generation, observable-open truth, generated-open formation, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RealOrderTopology}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, order trichotomy, Dedekind-cut construction, host subset equality, open-cover compactness, Lean verification, or bridge checking.

Definition 1890.2 (Real order topology classifier)**.** *Two accepted* $\mathsf{RealOrderTopology}^{\uparrow}$ *carriers*

$$O = (R, A, B, S, T, H, C, P, N) \quad \text{and} \quad O' = (R', A', B', S', T', H', C', P', N')$$

are classified by $\mathsf{msame}_{\mathsf{RealOrderTopology}^{\uparrow}}$ *exactly when their displayed real point rows, order-comparison rows, interval-or-ball basis rows, Sierpinski observable-open truth rows, generated-open rows, transports, replay routes, provenance rows, and local naming rows are matched componentwise by the visible* hsame *and* Cont *evidence. The classifier cannot inspect hidden host subsets, completeness witnesses, trichotomy decisions, Dedekind cuts, open-cover compactness data, Lean proof terms, or bridge certificates.*

Theorem 1890.3 (Real order topology NameCert obligations)**.** *For an accepted* $\mathsf{RealOrderTopology}^{\uparrow}$ *carrier* $O = (R, A, B, S, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealOrderTopology}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the real point-name rows, order-comparison rows, interval-or-ball basis rows, observable-open truth rows, generated-open topology rows, transport, replay, provenance, and local naming;*
2. *comparison-source exactness, requiring every interval endpoint, ray boundary, and ball-center comparison used by B to pass through the displayed* $\mathsf{RealAlgOrder}^{\uparrow}$ *row A and the displayed* $\mathsf{Real}^{\uparrow}$ *row R;*
3. *basis-generation exactness, requiring every basic open to be generated from the displayed interval-or-ball row B and not from a host subset predicate;*
4. *observable-open compatibility, requiring every membership read exported to* $\mathsf{SierpinskiSpace}^{\uparrow}$ *to be replayed through the same rows R, A, B before the truth row S is accepted;*
5. *topology-row stability, requiring the* $\mathsf{Topology}^{\uparrow}$ *row T to close only the generated-open operations displayed by B and S;*
6. *transport and replay stability, requiring H and C to preserve the route R, A, B, S, T, P, N without adding hidden completeness, trichotomy, Dedekind-cut, compactness, Lean, or bridge data;*
7. *ledger exhaustion, stating that no Real completeness, trichotomy, Dedekind-cut construction, open-cover compactness, Lean verification, or bridge checking is exported.*

No order-topology obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RealOrderTopologyUp.TasteGate.RealOrderTopologyTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1890.1. The accepted packet displays exactly one real point-name surface, one comparison row, one basis row, one observable-open truth row, one generated-open row, and the structural rows.

Comparison-source exactness follows from the pair R, A. Every endpoint, ray, and ball-center comparison consumed by the basis row has no second comparison coordinate and no hidden host-order field.

Basis generation and observable-open compatibility are forced by B and S. The basis row lists only finite interval-or-ball cells, and the observable truth row reads those cells through the same real-name and comparison rows.

Topology-row stability is the role of T. It records the generated-open operations admitted from B and S, so it cannot replace the generated surface with an ambient subset predicate.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the excluded completeness, trichotomy, Dedekind-cut, compactness, Lean, and bridge data are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1890.4 (Real order topology Cauchy-window basis)**.** *Every basic-open read of an accepted* $\mathsf{RealOrderTopology}^{\uparrow}$ *carrier factors through finite* $\mathsf{Real}^{\uparrow}$ *windows,* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparisons, and an interval-or-ball basis row before it reaches either the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row or the* $\mathsf{Topology}^{\uparrow}$ *generated-open row. In particular, a Cauchy-window consumer of real names can use the displayed route*

$$R, A, B \longmapsto S, T, H, C, P, N$$

without importing a host subset predicate, a selected limit, real-order trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $O = (R, A, B, S, T, H, C, P, N)$. The only point-name coordinate is R, so any real-name window opened by a Cauchy-style consumer must begin with that finite surface.

The only order service is A. Thus endpoint, ray, and ball-center tests used to recognize a basic open are read through the displayed comparison row.

The basis row B receives those finite comparison reads and records the interval-or-ball cells. The observable truth row S and the generated-open row T are downstream of that basis row, so neither can introduce an independent host subset.

The structural rows H, C, P, N preserve the same route. Because the packet has no selected-limit coordinate, trichotomy field, or completeness witness, the Cauchy-window consumer remains inside the stated finite basis route. □

Theorem 1890.5 (Real order topology Real seal nonescape)**.** *For an accepted* $\mathsf{RealOrderTopology}^{\uparrow}$ *carrier* $O = (R, A, B, S, T, H, C, P, N)$*, every read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal or a generated-open seal has already passed through the finite route*

$$R, A, B, S, T, H, C, P, N.$$

The read exports no Real completeness principle, no trichotomy decision, no Dedekind-cut construction, no open-cover compactness, no Lean verification, and no bridge checking.

Proof. A $\mathsf{Real}^{\uparrow}$ seal read can only inspect the point-name coordinate R and the downstream route displayed by the carrier. The packet contains no second real-name surface and no host equality coordinate.

A generated-open seal read must pass through A, B, S, T. The comparison row supplies finite order probes, the basis row supplies intervals or balls, the observable truth row supplies Sierpinski membership reads, and the topology row records generated-open formation.

Transport and replay are then supplied by H and C, with provenance and local naming supplied by P and N. These rows preserve the displayed route and cannot add the excluded completeness, trichotomy, Dedekind-cut, compactness, Lean, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealOrderTopology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealOrderTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1890.1, Definition 1890.2, Definition 1890.3, Definition 1890.4, and Definition 1890.5 seed the finite order-topology route over Real point rows, RealAlgOrder comparisons, interval-or-ball bases, Sierpinski observable-open truth, and Topology generated-open rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealOrderTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, Dedekind-cut construction, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, comparison-source exactness, basis generation, observable-open compatibility, topology-row stability, Cauchy-window basis routing, and Real seal nonescape for BEDC.Derived.RealOrderTopologyUp.

Constructive story: A $\mathsf{RealOrderTopology}^{\uparrow}$ packet is generated from Real point rows, RealAlgOrder comparison rows, interval or MetricBall basis rows, Sierpinski observable-open truth rows, Topology generated-open rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1891

A Concrete Naming Certificate for $\mathsf{CompactIntervalLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactIntervalLocator}^{\uparrow}$ packet records finite locator data for a compact real interval through existing $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$, $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. The chapter binds the interval surface of chapter 1709, the comparison service of chapter 1669, and the locatedness budget of chapter 1627 as one finite consumer packet. It gives Real and Completion routes a compact-interval locator surface; it is not open-cover compactness, endpoint selection, real-order trichotomy, quotient-stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1891.1 (Compact interval locator carrier)**.** *A* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, \Lambda, M, D, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source interval row, exposing the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, radius budget, endpoint compatibility, locator reads, and locatedness services already admitted by the interval packet.*
2. *Λ is the* $\mathsf{RealLocator}^{\uparrow}$ *comparison row used for rational probes against the displayed interval boundary.*
3. *M is the* $\mathsf{LocatednessModulus}^{\uparrow}$ *row that turns each positive tolerance request into finite endpoint windows, rational cells, dyadic refinements, and real-name readbacks.*
4. *D is the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *ledger for endpoint enclosure, interval refinement, containment comparison, and outward dyadic slack.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the Real and Completion consumer request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint and tolerance windows.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested positive dyadic scale and the rational comparison cells inspected by the locator.*
8. *H records componentwise* hsame *transport for the located interval, locator, modulus, dyadic interval ledger, stream windows, regular readback, dyadic tolerance ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-interval location request through* $I, \Lambda, M, D, S, R, Q, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalLocator}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{LocatednessModulus}^{\uparrow}$ $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalLocator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $I, \Lambda, M, D, S, R, Q, H, C, P, N$. It has no coordinate for open-cover compactness, selected endpoints, host closed-interval equality, trichotomy, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompactIntervalLocatorUp.CompactIntervalLocatorTasteGate single carrier alignment`

Theorem 1891.2 (Compact interval locator NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *carrier*

$$K = (I, \Lambda, M, D, S, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval row, real locator row, locatedness modulus row, dyadic interval arithmetic ledger, StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming;*
2. *interval-source exactness, requiring every endpoint, radius, containment, and boundary request to pass through the displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row I;*
3. *locator-modulus compatibility, requiring Λ and M to answer rational probes and tolerance requests through the same finite endpoint windows, dyadic cells, and real-name readback route;*
4. *dyadic ledger stability, requiring D and Q to store only displayed dyadic interval arithmetic, positive scales, and rational comparison cells;*
5. *consumer-window exhaustion, requiring S and R to expose finite windows and regular rational readback before any* $\mathsf{Real}^{\uparrow}$ *or Completion-facing seal is read;*
6. *nonescape, excluding open-cover compactness, selected endpoints, host interval equality, real-order trichotomy, quotient stream equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No compact interval locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1891.1. The accepted packet displays exactly one located interval source, one locator row, one locatedness modulus, one dyadic interval ledger, one finite stream-window row, one regular readback row, one dyadic tolerance ledger, and the structural rows.

Interval-source exactness is forced by I. Endpoint, radius, containment, separation, and refinement requests have no second interval coordinate, so every such read begins with the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ carrier.

Locator-modulus compatibility is the pair Λ, M. The locator row answers rational boundary probes, while the modulus row converts tolerance requests into the same finite endpoint windows, rational cells, dyadic refinements, and real-name readbacks.

Dyadic and consumer-window stability are supplied by D, Q, S, R. The dyadic rows name only displayed interval arithmetic, positive scales, and rational comparison cells; the window rows expose only finite StreamName windows and RegSeqRat readback.

The remaining rows H, C, P, N transport, replay, source, and name the same packet. Since the excluded compactness, endpoint-selection, order, quotient, choice, and ambient completeness data are absent from the carrier inventory, the listed rows exhaust the local NameCert surface. □

Theorem 1891.3 (Compact interval locator Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing read of an accepted* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *carrier factors through the finite route*

$$I \longmapsto \Lambda, M, D, Q \longmapsto S, R \longmapsto H, C, P, N.$$

The read may inspect the located real interval, locator decisions, locatedness budgets, dyadic interval arithmetic, dyadic tolerance cells, finite stream windows, regular rational readback, transport, replay, provenance, and local naming. It may not replace that route by open-cover compactness, selected endpoints, host interval equality, trichotomy, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (I, \Lambda, M, D, S, R, Q, H, C, P, N)$. The only source interval coordinate is I, so every Real or Completion-facing request first reads the located interval row with its endpoint, radius, locator, and locatedness boundary.

The computational route is then forced through Λ, M, D, Q. These rows provide the finite locator decisions, tolerance-to-window budget, dyadic interval arithmetic, and dyadic rational comparison cells for the same displayed interval.

The consumer-facing observation is finite. The request opens the StreamName window S and reads it through the RegSeqRat row R before any Real-facing seal can be consumed.

Finally H, C, P, N preserve, replay, source, and name the displayed route. None of these rows contains open-cover compactness, selected endpoint functionals, host interval equality, trichotomy, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so every accepted read remains inside the stated route. □

Falsifiable prediction: Every $\mathsf{CompactIntervalLocator}^{\uparrow}$ read factors through one $\mathsf{LocatedRealInterval}^{\uparrow}$ source row, one $\mathsf{RealLocator}^{\uparrow}$ comparison row, one $\mathsf{LocatednessModulus}^{\uparrow}$ tolerance row, one $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ ledger, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real-interval, real-locator, locatedness-modulus: $\mathsf{CompactIntervalLocator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source interval data, boundary locator decisions, tolerance-to-window budgets, dyadic interval arithmetic, finite observation windows, regular rational readback, and a Real or Completion-facing nonescape boundary as one local NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1891.1, Definition 1891.2, and Definition 1891.3 seed the finite compact-interval locator route over located real intervals, real locators, locatedness moduli, dyadic interval arithmetic, stream windows, regular rational readback, and dyadic tolerance ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected endpoints, host interval equality, real-order trichotomy, quotient stream equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exactness, locator-modulus compatibility, dyadic ledger stability, consumer-window exhaustion, and Real-route nonescape proofs for BEDC.Derived.CompactIntervalLocatorUp.

Constructive story: A CompactIntervalLocator$^{\uparrow}$ packet is generated from a LocatedRealIntervalUp row, a RealLocatorUp comparison row, a LocatednessModulusUp tolerance row, DyadicIntervalArithmeticUp rows, StreamNameUp windows, RegSeqRatUp readbacks, a DyadicRatCoreUp tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1892

A Concrete Naming Certificate for $\mathsf{UniformCompletionFunctor}^{\uparrow}$

Following Kelley on uniform spaces and Bishop–Bridges on constructive completion, the $\mathsf{UniformCompletionFunctor}^{\uparrow}$ packet names the functorial handoff from $\mathsf{UniformCompletion}^{\uparrow}$ data to completion consumers. It sits between the uniform-completion source of chapter 1473, the functorial Cauchy handoff of chapter 423, the universal-extension surface of chapter 397, the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet records a finite functorial route for completion-facing consumers; it is not a quotient completion, selected limit, host complete uniform space, open-cover compactness theorem, or choice principle.

Definition 1892.1 (UniformCompletionFunctor carrier)**.** *A* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

with the following visible rows:

1. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row, supplying the displayed Cauchy-filter, Cauchy-net, universal-extension, classifier, transport, replay, provenance, and local naming data read by the functor.*
2. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row, recording the source-to-completion unit route, completion-facing bind route, and functorial classifier surface accepted for the same source.*
3. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *universal-extension row, naming the finite extension and uniqueness ledger consumed after U and F have been projected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row used when the completion consumer asks for a regular source name or a regular output name.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite observation windows through which both source and completed rows are inspected.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision, comparison, and refinement budgets used by R and W.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the uniform-completion source, functorial handoff, universal-extension ledger, regular readback, stream windows, and dyadic tolerance rows have been replayed.*

8. *H records componentwise* hsame *transport for $U, F, E, R, W, D, S, C, P, N$.*
9. *C records displayed* Cont *replay, so finite completion-consumer requests reread $U, F, E, R, W, D, S, H, P, N$ without changing the route.*
10. *P is the* Pkg *provenance over* UniformCompletionFunctor$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted functor packets only through the displayed rows $U, F, E, R, W, D, S, H, C, P, N$. There is no coordinate for quotient completion equality, selected limits, host complete uniform spaces, open-cover compactness, or choice.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier definition surface`

Theorem 1892.2 (UniformCompletionFunctor NameCert obligations)**.** *For every accepted* UniformCompletionFunc *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N),$$

the local NameCert$_{\text{UniformCompletionFunctor}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed uniform-completion source, CauchyCompletionFunctor handoff, universal-extension row, regular readback row, stream-window row, dyadic tolerance row, Real seal row, transport, replay, provenance, and local naming;*
2. *source compatibility, requiring every completion-facing read to project the* UniformCompletion$^{\uparrow}$ *source row U before any functorial handoff is available;*
3. *handoff exactness, requiring F to expose the unit, bind, and classifier rows consumed by completion functor readers over the same source row;*
4. *universal-extension ordering, requiring E to be read after U and F and before any terminal* Real$^{\uparrow}$ *seal is exposed;*
5. *readback and tolerance exactness, requiring R, W, D to carry all regular rational, finite-window, and dyadic precision requests used by the handoff;*
6. *ledger non-escape, requiring H, C, P, N to export no quotient completion, selected limit, host complete uniform space, open-cover compactness proof, or choice principle.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1892.1. The accepted packet displays exactly the source, handoff, extension, readback, window, tolerance, seal, transport, replay, provenance, and local naming rows.

Source compatibility and handoff exactness are read from U and F. A completion consumer first reads the UniformCompletion$^{\uparrow}$ source row and then the CauchyCompletionFunctor$^{\uparrow}$ handoff row, so the packet has no separate entry point for a quotient completion or selected limit.

Universal-extension ordering is read from E, while readback and tolerance exactness are read from R, W, D. The seal row S is downstream of those rows, and H, C, P, N only transport, replay, source, and name the same displayed route. Since the excluded principles have no carrier coordinate, no replay edge, and no provenance row, the local NameCert surface is exactly the listed obligation surface. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorNamecertObligations`

Theorem 1892.3 (UniformCompletionFunctor source obligation)**.** *Every accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

must expose the UniformCompletion$^{\uparrow}$ *source row U before any completion-facing consumer may read the functorial handoff, universal-extension ledger, regular readback, stream window, dyadic tolerance, Real seal, transport, replay, provenance, or local naming rows. A consumer that bypasses U is not reading the displayed completion-functor packet.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier source factorization`

Proof. Project the displayed carrier of Definition 1892.1. The source row U is the only coordinate that supplies uniform-completion data to the packet.

Apply the NameCert obligations of Definition 1892.2. Their source-compatibility row requires each completion-facing read to project U before the handoff row F or any downstream row is public.

The sibling chapters named in the introduction supply the source and consumer surfaces: UniformCompletion$^{\uparrow}$ supplies U, while CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ are downstream rows. Since no separate source coordinate exists, the source obligation is exactly the projection through U. □

Theorem 1892.4 (UniformCompletionFunctor Cauchy handoff)**.** *Every* CauchyCompletionFunctor$^{\uparrow}$ *handoff consumed from an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier factors through the displayed rows*

$$U \longmapsto F \longmapsto E.$$

The row F may expose the unit, bind, and functorial classifier surface only for the same UniformCompletion$^{\uparrow}$ *source row U, and the universal-extension row E is read only after that handoff has been displayed.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier cauchy handoff`

Proof. Read the carrier rows U, F, E. The handoff row F is not an independent completion object; it is displayed as the CauchyCompletionFunctor$^{\uparrow}$ row attached to the source U.

Use Definition 1892.2. Handoff exactness confines the unit, bind, and classifier rows to the same source, while universal-extension ordering places E downstream of both U and F.

The introduction's CauchyCompletionFunctor and CompletionUniversalProperty siblings provide exactly those consumer surfaces. Nothing in the carrier adds quotient completion equality, a selected limit, host complete-uniform-space structure, or choice, so the Cauchy handoff is the displayed route $U \longmapsto F \longmapsto E$. □

Theorem 1892.5 (UniformCompletionFunctor bridge source lock)**.** *Every bridge read of the source coordinate in an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ factors through the ordered finite route*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D$$

before the terminal Real$^{\uparrow}$ *seal row S is exposed. The read cannot bypass the* UniformCompletion$^{\uparrow}$ *source row U, cannot treat F, E, R, W, D as independent completion sources, and cannot export a bridge surface that omits the displayed source before reaching S, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier bridge source lock`

Proof. Begin with the carrier of Definition 1892.1. The only source coordinate is U, and every downstream completion-facing row is displayed after it.

Use Definition 1892.3. It fixes U as the first public row of any completion-facing read, so a bridge cannot begin from F, E, R, W, D or from the terminal Real seal.

Next apply Definition 1892.4. The Cauchy-completion handoff F is attached to the same source row, and the universal-extension row E is downstream from that handoff.

Finally use Definition 1892.9. The readback, window, and dyadic rows R, W, D appear before the Real seal S, while H, C, P, N only transport, replay, source, and name the same route. Hence every bridge read is source-locked through U, F, E, R, W, D before any Real-facing seal is consumed. □

Theorem 1892.6 (UniformCompletionFunctor bridge target interface)**.** *Every target-side bridge read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ *factors through the displayed rows*

$$U, F, E, R, W, D, S, H, C, P, N.$$

In particular, the target interface preserves the $\mathsf{UniformCompletion}^{\uparrow}$ *source row* U*, the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row* F*, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* S *before any transport, replay, provenance, or local naming row is exported.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorBridgeTargetInterface`

Proof. Project the carrier of Definition 1892.1. The target-side bridge can read only the displayed coordinates $U, F, E, R, W, D, S, H, C, P, N$.

Use the source obligation, Definition 1892.3. It fixes U as the UniformCompletion source for every completion-facing read, so the bridge target cannot begin from a target seal or from an isolated handoff row.

Apply the Cauchy handoff theorem, Definition 1892.4, and the bridge source lock, Definition 1892.5. These place the CauchyCompletionFunctor row and the universal-property row downstream of the same source, then route the bridge through the regular readback, finite windows, and dyadic tolerance rows.

Finally use the Real-seal nonescape theorem, Definition 1892.9. It keeps S terminal for the displayed readback route and confines H, C, P, N to componentwise transport, replay, provenance, and local naming. Therefore every target-side bridge read exposes exactly the listed interface and no quotient completion, selected limit, host complete uniform space, open-cover compactness theorem, or choice principle. □

Theorem 1892.7 (UniformCompletionFunctor target-source order)**.** *Every target-side bridge read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet reaches the terminal target seal only after the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *window row, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row have been displayed. The target-side route therefore reads*

$$U \longmapsto F \longmapsto E \longmapsto R \longmapsto W \longmapsto D \longmapsto S$$

inside the carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ *before any terminal target seal, transport, replay, provenance, or local naming row is consumed.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorTargetSourceOrder`

Proof. Project the carrier of Definition 1892.1. The target-side bridge has no row other than $U, F, E, R, W, D, S, H, C, P, N$, so the terminal seal can only be reached through the displayed source-to-target route.

Apply Definition 1892.3. It places the $\mathsf{UniformCompletion}^{\uparrow}$ source row first for every completion-facing read and prevents the bridge from starting at the terminal seal.

Use Definition 1892.4 together with Definition 1892.5. These force the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ row to remain downstream of the source row before the bridge can read regular sequences, finite StreamName windows, or dyadic tolerances.

Finally apply Definition 1892.47. The existing target interface exposes the readback, window, tolerance, and terminal seal rows in the displayed carrier order and confines H, C, P, N to transport, replay, provenance, and local naming. Hence a target-side bridge read is source-before-target rather than a direct read of the terminal seal. □

Theorem 1892.8 (UniformCompletionFunctor Cauchy-completion handoff ordering)**.** *The* CauchyCompletionFunc *row F of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier is consumable only after the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row U has been displayed and only before the universal-extension row E is read. Any downstream* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer receives F only through the ordered route*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D \longmapsto S.$$

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCauchyCompletionHandoffOrdering`

Proof. Project the rows of Definition 1892.1. The Cauchy-completion handoff coordinate F is placed between the source coordinate U and the universal-extension coordinate E.

Apply Definition 1892.3 and Definition 1892.4. Together they require U before F and require E only after the handoff has been displayed.

For downstream consumers, use Definition 1892.10. It sends regular rational readback, stream windows, dyadic tolerance, and the Real seal through R, W, D, S after the same U, F, E prefix.

Thus F is neither a free-standing completion object nor a late Real-facing seal. It is the finite handoff row consumed after U and before E, with $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ strictly downstream. □

Theorem 1892.9 (UniformCompletionFunctor Real seal nonescape from root rows)**.** *For every accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row S can be consumed only after the rows R, W, D have been displayed downstream of U, F, E. The rows H, C, P, N may transport, replay, source, and name that same route, but they cannot turn S into a quotient completion, selected limit, host complete uniform space, open-cover compactness proof, or choice principle.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRealSealNonescape`

Proof. Project the carrier and isolate the terminal block R, W, D, S, H, C, P, N. The seal row S appears only after the readback, finite-window, and dyadic rows that mediate real-facing completion reads.

Apply Definition 1892.2. Readback and tolerance exactness assigns all regular rational, stream-window, and dyadic precision requests to R, W, D, while ledger non-escape confines the structural rows to the same packet.

The Real sibling named in the introduction is therefore reached only as a terminal seal for the displayed completion-consumer route. Since the remaining rows H, C, P, N preserve and name no additional coordinate, the excluded completion, compactness, and choice objects cannot escape from S. □

Theorem 1892.10 (UniformCompletionFunctor RegSeqRat and Real source handoff)**.** *Every completion-facing functor read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P,$ *that reaches the* $\mathsf{Real}^{\uparrow}$ *seal must first display the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row* U*, the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row* F*, the universal-extension row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S*. The structural rows* H, C, P, N *transport, replay, source, and name that same route, and no consumer may replace it by quotient completion equality, a selected limit, a host complete uniform space, open-cover compactness, or choice-selected representatives.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRegSeqRatRealSourceHandoff`

Proof. Start with the source obligation, Definition 1892.3. It fixes U as the first visible row for every completion-facing read.

Apply the Cauchy handoff theorem, Definition 1892.4. The functorial handoff F and universal-extension ledger E are downstream of the same source row, not independent completion objects.

Use the Real seal nonescape theorem above. It places the terminal seal S after the readback, finite-window, and dyadic rows R, W, D, and it confines H, C, P, N to transport and replay of that route.

Finally apply the universal handoff theorem, Definition 1892.23. The only completion-facing route is $U, F, E, R, W, D, S, H, C, P, N$, so the RegSeqRat and Real source handoff is finite, ordered, and free of quotient, selector, host-completeness, compactness, and choice coordinates. □

Theorem 1892.11 (UniformCompletionFunctor cofinal-window factorization)**.** *Every completion-facing cofinal-window read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ *factors through the ordered route*

$$U \longmapsto F \longmapsto E \longmapsto W, R, D \longmapsto S.$$

The structural rows H, C, P, N *preserve the same route under componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. No accepted consumer may replace the cofinal-window segment by quotient completion equality, a selected limit, a host complete uniform space, an open-cover compactness theorem, or choice-selected representatives.*

Proof. Start with the source obligation, Definition 1892.3. It fixes U as the first row of every completion-facing read, so a cofinal-window consumer cannot enter through W, R, D, or S directly.

Apply the Cauchy handoff theorem, Definition 1892.4. The handoff row F and universal-extension row E sit downstream of the same source row and upstream of the readback, window, tolerance, and seal rows.

Use the RegSeqRat and Real source handoff, Definition 1892.10. It places the cofinal-window service on the displayed W, R, D segment before the terminal seal S can be consumed.

Finally apply the Real seal nonescape theorem, Definition 1892.9. It confines H, C, P, N to transport, replay, provenance, and naming of the displayed route, so the cofinal-window factorization exports exactly $U, F, E, W, R, D, S, H, C, P, N$ and no hidden completion coordinate. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCofinalWindowFactorization`

Theorem 1892.12 (UniformCompletionFunctor Cauchy-filter nonescape)**.** *Let $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N$ be an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet. Any Cauchy-filter or completion-filter consumer that reads $\mathcal{F}$ receives only the displayed source, Cauchy handoff, extension, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance, Real seal, transport, replay, provenance, and local naming rows. It receives no quotient completion equality, selected limits, host complete uniform spaces, open-cover compactness, or choice-selected representatives.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCauchyFilterNonescape`

Proof. Project the carrier and apply Definition 1892.2. Carrier admission and ledger non-escape restrict every completion-filter read to the displayed rows $U, F, E, R, W, D, S, H, C, P, N$.

Use Definition 1892.10. It orders those rows before the Real seal and prevents a filter consumer from bypassing the RegSeqRat, StreamName, and DyadicRatCore source route.

Since no excluded completion, compactness, representative-selection, or host-completeness row occurs in the carrier, the Cauchy-filter read cannot escape the displayed finite packet. □

Theorem 1892.13 (UniformCompletionFunctor Real-RegSeqRat L10 nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RegSeqRat}^{\uparrow}$*-facing read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P$ must expose the rows $U, F, E, R, W, D, S, H, C, P, N$, with* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *visible in that order along the readback-and-seal segment. The packet exports no quotient completion equality, selected limit, host complete uniform space, open-cover compactness, choice-selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the RegSeqRat and Real source handoff, Definition 1892.10. It forces any terminal Real-facing read to display U, F, E, R, W, D, S before the seal can be consumed, and it keeps H, C, P, N on the same transport, replay, provenance, and naming route.

Apply the Cauchy-filter nonescape result, Definition 1892.12. Completion-filter consumers receive only the displayed carrier rows, so the Real and RegSeqRat faces cannot be replaced by quotient completion equality, selected limits, host complete uniform spaces, open-cover compactness, or choice-selected representatives.

The scoped L10 route of Definition 1892.42 identifies the visible readback segment as the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ corridor. The public export theorem, Definition 1892.43, then repacks exactly that ordered corridor into the public NameCert surface, with no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate added. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRealRegSeqRatL10Nonescape`

Theorem 1892.14 (UniformCompletionFunctor source factorization)**.** *Every completion-facing read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

factors through the $\mathsf{UniformCompletion}^{\uparrow}$ *source row U before the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row F is public. Thus the functorial unit, bind, and classifier reads cannot be sourced from quotient completion equality, a selected limit, a host complete uniform space, or a choice-selected representative.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorSourceFactorization`

Proof. Project the carrier of Definition 1892.1. The row U is the only source coordinate in the packet, and F is displayed as the handoff row attached to that source.

Apply Definition 1892.2. Its source-compatibility clause requires every completion-facing read to project U before any functorial handoff is available, while handoff exactness confines F to the same displayed source.

Since the carrier and obligation surface contain no quotient completion row, selected limit row, host complete-uniform-space row, or choice row, no consumer can bypass the source factorization through U. □

Theorem 1892.15 (UniformCompletionFunctor extension ordering)**.** *For every accepted* UniformCompletionFunctor *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$*, the universal-extension row* E *is consumed after the source and functorial handoff rows* U, F *and before the readback, window, dyadic, seal, transport, replay, provenance, and naming rows* R, W, D, S, H, C, P, N*. The ordered public route is therefore*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D \longmapsto S, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorExtensionOrdering`

Proof. Read the displayed carrier. The row E is the only $\mathsf{CompletionUniversalProperty}^{\uparrow}$ coordinate, and it sits between the source-handoff block U, F and the downstream readback block R, W, D, S.

Use the NameCert obligations of Definition 1892.2. Source compatibility and handoff exactness place U, F before E, and universal-extension ordering places E before any terminal $\mathsf{Real}^{\uparrow}$ seal can be exposed.

The remaining rows preserve the same order. The readback and tolerance clause accounts for R, W, D, while H, C, P, N only transport, replay, source, and name the displayed packet. Hence every public route has the stated ordering. □

Theorem 1892.16 (UniformCompletionFunctor readback seal exactness)**.** *In an accepted* UniformCompletionFunctor *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S *is exact only for readbacks that first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *row* W*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*. No completion-facing consumer may read* S *as a host real value, a quotient completion seal, or a selected limit independent of* R, W, D*.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorReadbackSealExactness`

Proof. Start from the carrier projection. The seal row S appears downstream of R, W, D, and there is no second Real-facing coordinate in the packet.

Apply Definition 1892.2. Its readback and tolerance exactness clause assigns all regular rational, finite-window, and dyadic precision requests to R, W, D, while universal-extension ordering keeps the seal downstream from U, F, E.

Thus a consumer reaches S only after the displayed readback surface has been consumed. The structural rows H, C, P, N preserve and name that same route, so the seal cannot be replaced by a host real value, quotient completion seal, selected limit, open-cover compactness proof, or choice principle. □

Theorem 1892.17 (UniformCompletionFunctor root obligation lattice)**.** *For every accepted* UniformCompletionFunctor *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N),$$

the root obligations form the ordered finite lattice

$$U \longmapsto F \longmapsto E \longmapsto (R, W, D) \longmapsto S, H, C, P, N.$$

The source row U is the unique root; carrier admission is the display of $\mathcal{F}$ as a single BHist *packet; source compatibility requires every consumer to read U before the handoff; the Cauchy handoff F is available only downstream of that source; universal-extension ordering puts E downstream of F and upstream of all readback rows; readback and tolerance exactness put R, W, D downstream of E and upstream of S; Real-seal nonescape keeps S terminal for the same route; and transport, replay, provenance, and local naming are exactly the terminal rows H, C, P, N. The lattice exports no quotient completion, selected limit, host complete uniform space, open-cover compactness theorem, or choice principle.*

Proof. Project the carrier of Definition 1892.1. Its row order displays one admitted BHist packet, one source coordinate U, one handoff coordinate F, one universal-extension coordinate E, the downstream readback-window-tolerance block R, W, D, and then S, H, C, P, N.

Apply the NameCert obligations of Definition 1892.2. Carrier admission accounts for the whole displayed packet. Source compatibility places U before any completion-facing handoff, Cauchy handoff exactness places F before any universal-extension read, universal-extension ordering places E before the terminal seal, and readback and tolerance exactness place R, W, D between E and the terminal rows.

Use the universal handoff theorem, Definition 1892.23, to identify the same order as the only consumer-visible route. The Real seal is read only as S after R, W, D, while H, C, P, N give transport, replay, provenance, and local naming for that route. Since the carrier has no coordinate outside this lattice, none of the excluded completion, compactness, or choice principles can be projected from the packet. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRootObligationLattice`

Theorem 1892.18 (UniformCompletionFunctor source-extension readback)**.** *Every* RegSeqRat$^{\uparrow}$ *or* StreamName$^{\uparrow}$ *readback consumed from an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier is downstream of both the* UniformCompletion$^{\uparrow}$ *source row U and the universal-extension row E. Concretely, each read of R or W factors through*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D$$

before the Real$^{\uparrow}$ *seal, transport, replay, provenance, or local naming rows are available.*

Proof. Read the carrier definition. The rows R and W occur in the same displayed packet as $U, F, E, D, S, H, C, P, N$, and no separate readback source is listed.

The NameCert obligations force the source and universal-extension order. Source compatibility makes U the first row for every completion-facing read, handoff exactness routes through F, and universal-extension ordering places E before any terminal seal can be exposed.

Now apply Definition 1892.23. Its route sends every completion-facing consumer through U, F, E before R, W, D. Hence a RegSeqRat or StreamName readback is a displayed downstream read of the UniformCompletion source and universal-extension ledger, not a quotient completion read, selected limit, host complete uniform space read, open-cover compactness proof, or choice-selected object. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorSourceExtensionReadback`

Theorem 1892.19 (UniformCompletionFunctor universal extension uniqueness ledger)**.** *Every uniqueness read of the universal-extension row in an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

factors through

$$U \longmapsto F \longmapsto E$$

before any regular-rational, stream-window, dyadic-tolerance, or terminal Real row is visible. The uniqueness ledger is therefore a displayed extension route over the $\mathsf{UniformCompletion}^{\uparrow}$ *source and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff, not a host uniqueness theorem about completed uniform spaces.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorUniversalExtensionUniquenessLe`

Proof. Project the carrier of Definition 1892.1. The extension row E is downstream of the source row U and the Cauchy handoff row F, while R, W, D, S, H, C, P, N are later readback and structural rows.

Apply the NameCert obligations and source obligation. Source compatibility forces every completion-facing read to start at U, and handoff exactness routes it through F before an extension ledger can be inspected.

Use the Cauchy handoff, universal handoff, and source-extension readback theorems. They keep E upstream of R, W, D and terminal Real sealing, so uniqueness is recorded only as the displayed U, F, E route and cannot import quotient completion equality, selected limits, host complete uniform spaces, compactness, or choice. □

Theorem 1892.20 (UniformCompletionFunctor readback tolerance terminal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier must expose the rows* R, W, D *as the regular-rational readback, stream-window readback, and dyadic tolerance ledger. Thus any route to* S *factors through*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D \longmapsto S$$

and no consumer may treat the seal as a host real value, a selected completion limit, or a quotient-completion endpoint.

Proof. Start from the readback seal exactness theorem. It already places the terminal seal S downstream of the regular rational row R, the stream window row W, and the dyadic tolerance row D.

Now apply the source obligation and universal handoff. The route to R, W, D is not free-standing; it first passes through U, F, E, so the readback and tolerance rows are terminal-route data for the same completion source.

Finally use the source-extension readback theorem. Since every R or W read is downstream of U, F, E and paired with D before S is visible, the Real seal remains a finite BEDC readback route rather than a host real, selected limit, quotient endpoint, compactness witness, or choice-selected completion. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorReadbackToleranceTerminalRoute`

Theorem 1892.21 (UniformCompletionFunctor functorial replay stability)**.** *Let* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P,$ *be an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier. Under functorial transport of the displayed source row* U*, the componentwise* hsame *row* H *and displayed* Cont *replay row* C *preserve the same route*

$$U, F, E, R, W, D, S, H, C, P, N.$$

The transported packet therefore exposes the same source, Cauchy handoff, universal-extension, readback, tolerance, Real seal, transport, replay, provenance, and local naming rows, with no additional completion object.

Proof. Project the accepted carrier and the transported source. The only source coordinate is U, and the handoff, extension, readback, tolerance, seal, and structural rows are attached to that same carrier.

Apply the NameCert obligations, source obligation, Cauchy handoff, and universal handoff. These order the consumer route as $U, F, E, R, W, D, S, H, C, P, N$ before any completion-facing data is exported.

Use H and C for the transported packet. Componentwise hsame moves each displayed row, while Cont replay presents the same row order to consumers. The provenance and local naming rows then cite the transported route without adding quotient completion equality, selected limits, host complete uniform spaces, compactness, or choice. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorFunctorialReplayStability`

Theorem 1892.22 (UniformCompletionFunctor completion consumer frontier)**.** *The completion-consumer frontier of an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier is exactly the displayed row family*

$$U, F, E, R, W, D, S, H, C, P, N.$$

A completion consumer may receive the source, handoff, universal-extension, readback, finite-window, dyadic, Real seal, transport, replay, provenance, and naming rows, and no further completion datum. In particular, quotient completions, selected limits, host complete uniform spaces, open-cover compactness, and choice are outside the frontier.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCompletionConsumerFrontier`

Proof. Carrier projection gives the candidate frontier. The definition of Definition 1892.1 lists exactly $U, F, E, R, W, D, S, H, C, P, N$ as visible rows, so any consumer-visible datum must be one of these rows or a replay through them.

The NameCert obligations identify how those rows are admitted. Carrier admission, source compatibility, handoff exactness, universal-extension ordering, readback and tolerance exactness, and ledger non-escape account for each row in the displayed family.

Finally use Definition 1892.23. It orders the same family as the completion-facing route and confirms that terminal structural rows only transport, replay, source, and name the displayed data. Because the carrier and its obligations contain no other coordinate, the excluded completion, compactness, and choice objects are not part of the consumer frontier. □

Theorem 1892.23 (UniformCompletionFunctor universal handoff)**.** *Every completion-facing consumer of an accepted* UniformCompletionFunctor$^{\uparrow}$ *carrier factors through the displayed route*

$$U \longmapsto F \longmapsto E \longmapsto R, W, D \longmapsto S, H, C, P, N.$$

Thus the consumer receives the uniform-completion source row, then the CauchyCompletionFunctor handoff, then the universal-extension ledger, and only then the regular rational, StreamName, DyadicRatCore, Real seal, transport, replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorUniversalHandoff`

Proof. Start with an accepted carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$. The only uniform-completion source coordinate is U, so every completion-facing request begins by projecting the displayed UniformCompletion$^{\uparrow}$ data.

The functorial handoff is then read through F. Its unit, bind, and classifier rows are attached to the same source row, and the universal-extension row E records the extension and uniqueness ledger available to the consumer after that handoff.

The readback rows are downstream of the extension ledger. Regular rational readback uses R, finite observation uses W, dyadic precision uses D, and the terminal real-name seal is exposed only through S after those requests have been replayed.

Finally H, C, P, N transport, replay, package, and name the same finite route. The carrier definition contains no coordinate for quotient completion equality, selected limits, host complete uniform spaces, open-cover compactness, or choice, so no completion-facing consumer can read such data from the handoff. □

Theorem 1892.24 (UniformCompletionFunctor Real seal route)**.** *Every accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *real-facing consumer must replay the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *finite window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal in that order before exporting a completion result. The export route is therefore*

$$D \longmapsto W \longmapsto R \longmapsto S \longmapsto H, C, P, N$$

inside the accepted carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$*, downstream of the source, handoff, and universal-extension rows* U, F, E*.*

No real-facing consumer may replace this route by a host complete-space theorem, open-cover compactness argument, quotient completion, selected limit, or choice principle.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRealSealRoute`

Proof. Start with Definition 1892.23. It forces every completion-facing consumer through U, F, E before any readback, window, tolerance, or Real-seal row is available.

Use the carrier and obligation surface of Definition 1892.1 and Definition 1892.2. The dyadic tolerance row D supplies the precision budget for the subsequent finite-window read W; the window row then bounds the regular rational readback R; only after those rows have been displayed can the terminal Real seal S be read.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for host complete spaces, open-cover compactness, quotient completions, selected limits, or choice, none of those principles can be exported as a completion result. □

Theorem 1892.25 (UniformCompletionFunctor source-unit compatibility)**.** *Every accepted* $\mathsf{UniformCompletionFun}$ *packet reads the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row* U *before the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row* F*. Hence the functor unit is sourced from the displayed uniform-completion packet and cannot be sourced from a quotient completion, a selected limit, or a host complete uniform space.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorSourceUnitCompatibility`

Proof. Project the carrier of Definition 1892.1. The source coordinate U is the only row that supplies uniform-completion data to the packet, while F is explicitly the handoff row attached to that same source.

Apply Definition 1892.2. Its source-compatibility and handoff-exactness clauses require completion-facing reads to project U before any handoff through F is available.

Apply Definition 1892.23. The displayed route begins with U and then reads F, so the unit route is a ledger entry over the accepted source row rather than a datum recovered from quotient completion equality, selected limits, or a host complete uniform space. □

Theorem 1892.26 (UniformCompletionFunctor universal-extension uniqueness route)**.** *For every accepted* UniformCompletionFunctor$^{\uparrow}$ *packet, the universal-extension row E is consumed after U, F and before R, W, D, S. The uniqueness information therefore appears as a displayed ledger route from the source and functorial handoff into the readback, window, tolerance, and seal rows, not as a host complete-uniform-space theorem.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorUniversalExtensionUniquenessRou`

Proof. Project the accepted carrier

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N).$$

The row E is the only universal-extension coordinate, and the carrier places it between the source-handoff rows U, F and the downstream readback rows R, W, D, S.

Apply Definition 1892.2. The universal-extension ordering clause requires E to be read after the source row and CauchyCompletionFunctor handoff, before any terminal Real seal is exposed.

Apply Definition 1892.23. The consumer-visible route is $U \longmapsto F \longmapsto E \longmapsto R, W, D \longmapsto S, H, C, P, N$, so uniqueness is consumed as part of that finite ledger. Since the packet has no coordinate for a host complete-uniform-space theorem, quotient completion equality, selected limit, or choice principle, no stronger uniqueness source is exported. □

Theorem 1892.27 (UniformCompletionFunctor RegSeqRat readback totality)**.** *Every accepted completion-facing read that reaches the* RegSeqRat$^{\uparrow}$ *row R of a* UniformCompletionFunctor$^{\uparrow}$ *packet must first project the source row U, the functorial handoff row F, and the universal-extension row E, and must then expose the same* StreamName$^{\uparrow}$ *finite window W used by the readback. Thus R is a total displayed readback only for the route*

$$U, F, E, W, R,$$

not for quotient completion equality, selected representatives, host complete uniform spaces, open-cover compactness, or choice.

Proof. Project the carrier of Definition 1892.1. The row R occurs only after U, F, E have been displayed, and its window source is the same row W appearing in the carrier.

Apply Definition 1892.2. Its readback and tolerance clauses make R an obligation over the finite source, handoff, universal-extension, and window rows.

Use Definition 1892.23 and Definition 1892.26. Together they route every completion-facing consumer through U, F, E before any downstream readback row is inspected. Since no excluded completion or choice coordinate occurs in the carrier, the displayed readback route is total exactly over those rows. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.RegSeqRatReadbackTotality.UniformCompletionFunctorRegSe`

Theorem 1892.28 (UniformCompletionFunctor dyadic window stability)**.** *For every accepted* UniformCompletionFunctor$^{\uparrow}$ *packet, the* DyadicRatCore$^{\uparrow}$ *row D supplies the precision budget used by both the readback row R and the finite-window row W. Under componentwise* hsame *transport H and displayed* Cont *replay C, the same dyadic budget is preserved along the completion-facing route*

$$U, F, E, R, W, D, S.$$

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier dyadic window stability`

Proof. Read the carrier definition. The rows R, W, D occur together in the downstream part of $U, F, E, R, W, D, S, H, C, P, N$, so there is no separate tolerance source outside the displayed packet.

The NameCert obligations identify D as the tolerance row governing readback and window exactness. They also confine classifier-visible comparison to componentwise transport, which is the row H.

Finally apply the universal handoff and universal-extension uniqueness route. They force the consumer to replay R, W, D after U, F, E and before S. The displayed replay row C therefore preserves the same dyadic budget; no host complete-space or open-cover argument can replace it. □

Theorem 1892.29 (UniformCompletionFunctor Real seal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal row S of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet is reachable only after the source, handoff, universal-extension, readback, window, and dyadic rows*

$$U, F, E, R, W, D$$

have been displayed. The seal exports no quotient completion, selected limit, host complete uniform space, open-cover compactness, or choice principle.

Proof. Begin with the carrier and obligation surface. The seal row S is downstream from U, F, E, R, W, D, and the local NameCert obligations attach seal nonescape to that displayed order.

Use Definition 1892.23. It gives the completion-facing route from the source row through the functorial and universal-extension rows before readback, windows, tolerance, and the terminal seal are consumed.

Apply Definition 1892.26. Since uniqueness is already recorded as a finite ledger route before S, the seal cannot introduce a host completed object or selected limit. The excluded completion, compactness, and choice principles are not coordinates of the packet. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRealSealNonescapeFromDisplayedP`

Theorem 1892.30 (UniformCompletionFunctor ledger exactness)**.** *The carrier, obligation surface, and universal handoff of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet exhaust its completion-consumer ledger. Every exported route is a projection from*

$$U, F, E, R, W, D, S, H, C, P, N,$$

and every classifier-visible comparison is componentwise through the same rows. No exported route can bypass the displayed ledger to obtain quotient completion, selected limits, host complete uniform spaces, open-cover compactness, or choice.

Proof. Project the accepted carrier of Definition 1892.1. The displayed coordinates are exactly $U, F, E, R, W, D, S, H, C, P, N$, and the local classifier is stated only over those coordinates.

Apply Definition 1892.2 to account for carrier admission, source compatibility, handoff exactness, universal-extension ordering, readback and tolerance exactness, and ledger non-escape. Apply Definition 1892.23 to identify the order in which a completion-facing consumer can inspect those rows.

It remains to rule out escape from the ledger. Componentwise hsame transport is confined to H, displayed replay is confined to C, provenance is confined to P, and local naming is confined to N. Because no further coordinate is present, the packet exports exactly the displayed ledger and none of the excluded completion or choice principles. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorLedgerExactness`

Theorem 1892.31 (UniformCompletionFunctor carrier admission)**.** *Every accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet admits exactly one displayed carrier row*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

whose coordinates are read as $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame, *displayed* Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. Admission is refused for any completion-facing route that omits the* BHist *carrier row, bypasses* U*, or treats* Pkg *and* $\mathsf{NameCert}$ *as external annotations rather than displayed coordinates of the same packet.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier admission`

Proof. Project the carrier definition. Its rows are visible as one finite BHist packet, so admission is not a separate source-equivalence transport; it is the constructive projection of the displayed tuple.

The source row U is the unique $\mathsf{UniformCompletion}^{\uparrow}$ coordinate, and the handoff row F is the unique $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ coordinate attached to that source. The universal row E, readback row R, window row W, dyadic row D, and terminal seal S are then admitted only as downstream coordinates of the same packet.

Finally H, C, P, N expose hsame, Cont, Pkg, and $\mathsf{NameCert}$ inside the carrier. Since no row lies outside the tuple, a route without the displayed BHist packet, source coordinate, provenance coordinate, or naming coordinate is not an accepted $\mathsf{UniformCompletionFunctor}^{\uparrow}$ carrier. □

Theorem 1892.32 (UniformCompletionFunctor functoriality)**.** *Let a completion-facing request read an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N).$$

The functorial unit, bind, and classifier routes exposed by the packet are exactly the finite route compositions through $U, F, E, R, W, D, S, H, C, P, N$*. In particular, functoriality is witnessed by the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row* F*, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row* E*, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observation rows* W, R*, the* $\mathsf{Real}^{\uparrow}$ *seal* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*; it is not transported from a source-equivalence theorem outside the packet.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorFunctoriality`

Proof. Read the packet from left to right. The source coordinate U determines the uniform-completion input, and F supplies the unit, bind, and classifier handoff rows for that same input.

Next, E records the extension and uniqueness ledger consumed by the handoff. The finite observation rows W, R, D provide stream windows, regular rational readback, and dyadic tolerance before the terminal $\mathsf{Real}^{\uparrow}$ seal S can be read.

The functorial comparison then composes through H, C, P, N. Componentwise hsame transport, displayed Cont replay, provenance, and local naming are all coordinates of the packet, so the composition is finite route composition over BHist rows rather than transport from an external source equivalence. □

Theorem 1892.33 (UniformCompletionFunctor completion handoff)**.** *Every completion consumer receiving an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet must factor its read through*

$$\mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRa}$$

Thus the packet hands completion data to downstream consumers by displayed rows only: the source row U*, functorial handoff* F*, universal-property row* E*, window and readback rows* W, R, D*, seal row* S*, transport* H*, replay* C*, provenance* P*, and naming* N*.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCompletionHandoff`

Proof. Start with the carrier row $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$. A completion consumer has no coordinate other than these displayed rows, so it first reads the $\mathsf{UniformCompletion}^{\uparrow}$ source U and then the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff F.

The universal handoff is mediated by E. Before a real-facing result is exposed, the consumer must read the finite $\mathsf{StreamName}^{\uparrow}$ window row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, after which S supplies the terminal $\mathsf{Real}^{\uparrow}$ seal.

The remaining structural rows keep the route inside the packet. The BHist carrier is the ambient finite packet, H gives componentwise hsame transport, C gives displayed Cont replay, P gives Pkg provenance, and N gives local $\mathsf{NameCert}$ naming. This is a constructive completion handoff by finite projections and route composition, not a source-equivalence transport. □

Theorem 1892.34 (UniformCompletionFunctor root obligation surface)**.** *Before any completion consumer can read the functorial extension exported by an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N),$$

the packet must expose exactly the root obligation surface

$$U, F, E, R, W, D, S, H, C, P, N.$$

Here U *is the source uniformity row,* F *is the Cauchy-completion unit and handoff row,* E *is the universal-extension row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* W *is the* $\mathsf{StreamName}^{\uparrow}$ *scheduling row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-comparison row,* S *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* H *is componentwise transport,* C *is replay,* P *is provenance, and* N *is local naming. No completion consumer can read the functorial extension before this surface is displayed, and the ledger-exactness row excludes quotient completions, selected limits, host complete uniform spaces, open-cover compactness, and choice.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRootObligationSurface`

Proof. Project the carrier of Definition 1892.1. It displays the source row U, handoff row F, universal-extension row E, readback row R, scheduling row W, tolerance row D, seal row S, and the structural rows H, C, P, N as one finite BHist packet.

Apply Definition 1892.2. Its carrier admission, source compatibility, handoff exactness, universal-extension ordering, readback and tolerance exactness, and ledger non-escape clauses assign each required obligation to one of the displayed rows.

Use Definition 1892.23 and Definition 1892.33. Together they force every completion-facing read to pass through U, F, E, R, W, D, S before transport, replay, provenance, and local naming are consumed.

Finally apply Definition 1892.30. Since every exported route is a projection from the same row family and no excluded completion or choice datum has a carrier coordinate, the displayed rows are exactly the root obligation surface needed before the functorial extension can be read. □

Theorem 1892.35 (UniformCompletionFunctor route obligations)**.** *Every accepted* UniformCompletionFunctor$^{\uparrow}$ *route must read the* CauchyCompletionFunctor$^{\uparrow}$ *handoff row, the* CompletionUniversalProperty$^{\uparrow}$ *universal-property row, the* DyadicRatCore$^{\uparrow}$ *tolerance row, the* Real$^{\uparrow}$ *seal row, the* RegSeqRat$^{\uparrow}$ *readback row, the* StreamName$^{\uparrow}$ *window row, and the* UniformCompletion$^{\uparrow}$ *source row before exporting a functorial completion handoff. Equivalently, every completion-facing handoff is a projection from*

$$U, F, E, R, W, D, S, H, C, P, N$$

inside the displayed BHist *carrier, with no host quotient completion or external selector exported by the packet.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier route obligations`

Proof. Project the accepted carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$. The carrier row already displays the UniformCompletion source, CauchyCompletionFunctor handoff, universal-property ledger, regular readback, stream window, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows as one finite packet.

Use the NameCert obligations and the root obligation surface. The source row U is read first, the handoff row F is attached to that source, the universal-property row E is downstream of both, and the readback block R, W, D must be visible before the terminal seal S.

Finally apply the universal handoff and ledger exactness theorems. They identify $U, F, E, R, W, D, S, H, C, P, N$ as the only consumer-visible route. Since the carrier has no coordinate for host quotient completion or selected representatives, the functorial completion handoff exports only the displayed BEDC rows. □

Theorem 1892.36 (UniformCompletionFunctor unit root obligation)**.** *The unit map of an accepted* UniformCompletionFunctor$^{\uparrow}$ *completion route must expose the source carrier row U, the* CauchyCompletionFunctor$^{\uparrow}$ *handoff row F, the Cauchy filter or* RegSeqRat$^{\uparrow}$ *window row R, the terminal* Real$^{\uparrow}$ *seal S, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before any completed point is exported.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorCarrier unit root obligation`

Proof. Start with the source obligation of Definition 1892.3. It fixes the source carrier row U as the first visible row of the completion route.

Use the Cauchy handoff of Definition 1892.4. The handoff row F and readback row R expose the Cauchy filter or regular rational window before the completed point is available.

The Real seal nonescape theorem Definition 1892.9 keeps the terminal row S from becoming ambient completeness or a hidden selected limit. The root obligation surface of Definition 1892.34 then adds only H, C, P, N for transport, replay, provenance, and local naming. □

Theorem 1892.37 (UniformCompletionFunctor extension root obligation)**.** *Every uniformly continuous extension read exported by an accepted* UniformCompletionFunctor$^{\uparrow}$ *packet factors through the displayed source-extension ledger*

$$U, F, E, R, W, D, S, H, C, P, N.$$

The read exports no selected limit, quotient equality, host function extensionality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorExtensionRootObligation`

Proof. Project the universal-extension row E from the carrier of Definition 1892.1. It is downstream of the source row U and the Cauchy handoff row F.

Apply Definition 1892.23. The extension read must pass through the source, handoff, readback, scheduling, tolerance, and Real seal rows before it reaches the structural rows.

Use Definition 1892.34. It identifies the same finite row family as the root surface, so the extension is a displayed source-extension ledger rather than a selected limit, quotient equality, host function-extensionality argument, or ambient completeness theorem. □

Theorem 1892.38 (UniformCompletionFunctor functorial root obligation)**.** *Composition of two accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packets preserves the displayed source, completion, extension, transport, replay, and provenance rows. The composite route may read only the two source rows, the two Cauchy-completion handoff rows, the two universal-extension rows, the displayed readback and tolerance rows, componentwise transport, continuation replay, provenance, and local naming rows already present in the packets.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorFunctorialRootObligation`

Proof. Start with the functoriality theorem Definition 1892.32. It states that unit, bind, and classifier routes are finite route compositions through the displayed carrier rows.

For each packet, use the source obligation and Cauchy handoff, Definition 1892.3 and Definition 1892.4. These keep the source and completion rows visible before any composite extension is read.

The extension root obligation above places uniformly continuous extension reads on the same $U, F, E, R, W, D, S, H, C, P, N$ route. Componentwise transport and displayed replay compose only those rows, while provenance records the two packet sources. Thus composition preserves the displayed source, completion, extension, transport, replay, and provenance data without importing an external completion principle. □

Theorem 1892.39 (UniformCompletionFunctor L10 source admission)**.** *An accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet admits an L10 completion source only through the displayed finite route*

$$\mathsf{UniformCompletion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, a completion-facing source read must project the $\mathsf{UniformCompletion}^{\uparrow}$ *row* U*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S*, the ambient* BHist *packet, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. It is not admitted by source-equivalence transport outside the packet.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorL10SourceAdmission`

Proof. Project the accepted carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$. The source row U is public only as a coordinate of the same finite BHist packet that carries R, W, D, S, H, C, P, N.

Use Definition 1892.14 for the first projection through U. Then use Definition 1892.16 to force the L10 readback through D, W, R before the Real seal S is consumed.

The structural rows finish the admission route. Componentwise hsame transport and displayed Cont replay preserve the projected rows, while Pkg provenance and local $\mathsf{NameCert}$ naming record the same finite source. No external source-equivalence object is a coordinate of the packet, so it cannot admit the L10 source. □

Theorem 1892.40 (UniformCompletionFunctor finite-window functoriality)**.** *Functorial composition of accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packets preserves the finite-window completion route through*

$$U, F, E, D, W, R, S, H, C, P, N.$$

For each input packet, the composite may read only the $\mathsf{UniformCompletion}^{\uparrow}$ *source row* U*, the Cauchy handoff row* F*, the universal-extension row* E*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*.*

Proof. Begin with Definition 1892.32. It identifies functorial unit, bind, and classifier reads with finite route composition through the displayed carrier rows.

For each composed packet, apply Definition 1892.33. The route first projects U, F, E and then reads the finite tolerance, window, and readback block D, W, R before the terminal Real seal.

It remains to check that composition does not add a hidden source. The rows H and C perform componentwise transport and replay over the same finite windows, while P and N preserve provenance and local naming. Thus functoriality is a finite-window induction over the displayed route, not transport from an external source-equivalence theorem. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorWindowFunctoriality`

Theorem 1892.41 (UniformCompletionFunctor Real-seal consumer route)**.** *Every Real-facing consumer of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet must reach the* $\mathsf{Real}^{\uparrow}$ *seal by the concrete L10 route*

$$\mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCe}$$

The consumer receives a sealed completion read only after the dyadic tolerance, finite stream window, regular rational readback, componentwise transport, replay, provenance, and local naming rows have been projected.

Proof. Start with Definition 1892.24. It places the Real seal downstream from the source, handoff, universal-extension, readback, window, and dyadic rows.

Apply Definition 1892.39. The source is admitted only through the same BHist packet and the displayed D, W, R, S, H, C, P, N rows.

Finally use Definition 1892.30. The ledger has no coordinate for selected limits, quotient completion equality, host complete uniform spaces, open-cover compactness, or choice. Therefore every Real-facing consumer route is the finite L10 route through the displayed rows. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRealSealConsumerRoute`

Theorem 1892.42 (UniformCompletionFunctor scoped L10 obligation route)**.** *The scoped obligation surface of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet is exactly the finite route*

$$\mathsf{UniformCompletion}^{\uparrow},\ \mathsf{CauchyCompletionFunctor}^{\uparrow},\ \mathsf{CompletionUniversalProperty}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},$$

Concretely, every source read must project the row U*, every target completion read must pass through the Cauchy-completion handoff* F *and universal-property row* E*, every regular and real-facing read must display* R, S, W*, and every structural replay must remain inside* H, C, P, N*. No quotient completion class, ambient completeness assumption, host complete uniform space, open-cover compactness principle, or choice-selected representative belongs to the scope.*

Proof. Start with the carrier and local NameCert obligations, Definition 1892.1 and Definition 1892.2. They list exactly the source row U, handoff row F, universal-extension row E, readback rows R, W, D, S, and structural rows H, C, P, N.

Use Definition 1892.23 and Definition 1892.34. These theorems order the same rows from the UniformCompletion source through the CauchyCompletionFunctor and CompletionUniversalProperty handoff before any RegSeqRat, StreamName, DyadicRatCore, or Real seal row can be consumed.

Finally apply Definition 1892.41 and Definition 1892.30. The Real-facing route remains inside the displayed L10 packet, and the ledger excludes quotient completion classes, ambient completeness, host complete uniform spaces, open-cover compactness, and choice. Hence the scoped obligation surface is the finite BEDC route named in the statement. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorScopedL10ObligationRoute`

Theorem 1892.43 (UniformCompletionFunctor public NameCert export)**.** *An accepted* UniformCompletionFunctor *packet exports its public* NameCert *only after reading the* CauchyCompletionFunctor$^{\uparrow}$ *route, the* UniformCompletion$^{\uparrow}$ *source,* StreamName$^{\uparrow}$ *finite windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance rows, the* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The export refuses quotient completion classes, selected limits, countable choice, and ambient Real completeness.*

Proof. Begin with the carrier and the NameCert obligations. They admit exactly the source row U, handoff row F, universal-extension row E, readback rows R, W, D, S, and structural rows H, C, P, N.

Apply the completion handoff and L10 source-admission routes. The public consumer first reads the UniformCompletion$^{\uparrow}$ source and CauchyCompletionFunctor$^{\uparrow}$ handoff, then exposes StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance, and the terminal Real$^{\uparrow}$ seal.

Finally use ledger exactness. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same finite route, while quotient completion, selected limits, countable choice, and ambient Real completeness have no coordinate in the packet. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorPublicNamecertExport`

Theorem 1892.44 (UniformCompletionFunctor L10 bridge schema)**.** *An accepted* UniformCompletionFunctor$^{\uparrow}$ *packet supplies a bridge from its public* NameCert *export to the concrete L10 completion schema exactly through* UniformCompletion$^{\uparrow}$*,* CauchyCompletionFunctor$^{\uparrow}$*,* CompletionUniversalProperty$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The bridge schema is the finite route that first displays the public export and then reuses the same source, functorial handoff, universal-extension, finite-window, regular-readback, dyadic-tolerance, Real-seal, transport, replay, provenance, and local naming rows for L10 completion consumers.*

Proof. Start with the scoped L10 obligation route, Definition 1892.42. It identifies the finite route through the source, completion handoff, universal-property row, regular readback, Real seal, StreamName windows, and structural rows.

Apply the Real-seal consumer route and ledger exactness, Definition 1892.41 and Definition 1892.30. These results keep every completion-facing read inside the displayed dyadic, window, readback, Real, transport, replay, provenance, and local naming rows.

Finally use the public NameCert export, Definition 1892.43. The export supplies the public endpoint, while the previous routes provide the concrete L10 completion schema consumed downstream. No source-equivalence parameters or abstract carrier transport rows are introduced. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorBridgeSchema`

Theorem 1892.45 (UniformCompletionFunctor RegSeqRat-to-Real bridge)**.** *An accepted* UniformCompletionFunct *packet whose functorial handoff consumes* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readbacks reaches its* Real$^{\uparrow}$ *endpoint only through the same displayed uniform-completion functor rows, dyadic tolerance ledger, RegSeqRat handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The bridge exports no quotient completion equality, selected representative, ambient Real completeness theorem, or detached standard bridge.*

Proof. Start with the public export and functorial handoff routes, Definition 1892.43 and Definition 1892.44. They keep every completion-facing read inside the source row U, handoff row F, universal-extension row E, readback rows R, W, D, S, and structural rows H, C, P, N.

Use the L10 readback path already carried by Definition 1892.1. A finite StreamName$^{\uparrow}$ window is read through the RegSeqRat$^{\uparrow}$ handoff and the DyadicRatCore$^{\uparrow}$ tolerance ledger before the terminal Real$^{\uparrow}$ seal is exposed.

Finally apply the Real-seal consumer route of Definition 1892.41. Since the carrier has no coordinate for quotient completion equality, selected representatives, ambient Real completeness, or a detached standard bridge, the RegSeqRat-to-Real endpoint remains the displayed finite route through D, W, R, S, H, C, P, N. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorRegSeqRealBridge`

Theorem 1892.46 (UniformCompletionFunctor L10 status product noncollapse)**.** *An accepted* UniformCompletionFunctor$^{\uparrow}$ *packet exposes four separate local status faces for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$, *together with the* UniformCompletion$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, *and* CompletionUniversalProperty$^{\uparrow}$ *rows carried by the same completion route. No aggregate completion status route may replace the four face-local closurestatus, formalstatus, and bridge rows attached to those L10 faces.*

Proof. Start with the public export and bridge schema, Definition 1892.43 and Definition 1892.44. They expose the finite completion route through the source row, functorial handoff, universal-extension row, dyadic tolerance row, StreamName window row, RegSeqRat readback row, Real seal, and structural rows.

Use the RegSeqRat-to-Real bridge of Definition 1892.45. It keeps the readback path face-local: dyadic tolerance, stream windows, regular rational readback, and the terminal Real seal remain separately visible before the completion-facing endpoint is read.

The carrier rows in Definition 1892.1 supply no fused status coordinate. Therefore a downstream completion consumer may project the UniformCompletion, CauchyCompletion, and universal-property rows, but it cannot replace the four L10 face records by a single aggregate closurestatus, formalstatus, or bridge row. □

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorL10StatusProductNoncollapse`

Falsifiable prediction: Every UniformCompletionFunctor$^{\uparrow}$ completion-facing read factors through one UniformCompletion$^{\uparrow}$ source row, one CauchyCompletionFunctor$^{\uparrow}$ handoff row, one CompletionUniversalProperty$^{\uparrow}$ universal-extension row, one RegSeqRat$^{\uparrow}$ readback row, one StreamName$^{\uparrow}$ finite-window row, one DyadicRatCore$^{\uparrow}$ tolerance row, one terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniformcompletion, cauchycompletionfunctor, completionuniversalproperty, regseqrat, streamname, dyadicratcore, real: UniformCompletionFunctor$^{\uparrow}$ is not bijective to any

listed sibling because it jointly carries the source, functorial handoff, universal-extension, regular readback, finite-window, dyadic tolerance, Real seal, transport, replay, provenance, and local NameCert rows as one finite completion-consumer packet.

Theorem 1892.47 (UniformCompletionFunctor bridge target interface)**.** *Every target-side bridge read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier* $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ *factors through the displayed rows*

$$U, F, E, R, W, D, S, H, C, P, N.$$

In particular, the target interface preserves the $\mathsf{UniformCompletion}^{\uparrow}$ *source row* U*, the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row* F*, the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* S *before any transport, replay, provenance, or local naming row is exported.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorBridgeTargetInterface`

Proof. Project the carrier of Definition 1892.1. The target-side bridge can read only the displayed coordinates $U, F, E, R, W, D, S, H, C, P, N$.

Use the source obligation, Definition 1892.3. It fixes U as the UniformCompletion source for every completion-facing read, so the bridge target cannot begin from a target seal or from an isolated handoff row.

Apply the Cauchy handoff theorem, Definition 1892.4, and the bridge source lock, Definition 1892.5. These place the CauchyCompletionFunctor row and the universal-property row downstream of the same source, then route the bridge through the regular readback, finite windows, and dyadic tolerance rows.

Finally use the Real-seal nonescape theorem, Definition 1892.9. It keeps S terminal for the displayed readback route and confines H, C, P, N to componentwise transport, replay, provenance, and local naming. Therefore every target-side bridge read exposes exactly the listed interface and no quotient completion, selected limit, host complete uniform space, open-cover compactness theorem, or choice principle. □

Theorem 1892.48 (UniformCompletionFunctor target-source order)**.** *Every target-side bridge read of an accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *packet reaches the terminal target seal only after the* $\mathsf{UniformCompletion}^{\uparrow}$ *source row,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *window row, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row have been displayed. The target-side route therefore reads*

$$U \longmapsto F \longmapsto E \longmapsto R \longmapsto W \longmapsto D \longmapsto S$$

inside the carrier $\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$ *before any terminal target seal, transport, replay, provenance, or local naming row is consumed.*

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorTargetSourceOrder`

Proof. Project the carrier of Definition 1892.1. The target-side bridge has no row other than $U, F, E, R, W, D, S, H, C, P, N$, so the terminal seal can only be reached through the displayed source-to-target route.

Apply Definition 1892.3. It places the $\mathsf{UniformCompletion}^{\uparrow}$ source row first for every completion-facing read and prevents the bridge from starting at the terminal seal.

Use Definition 1892.4 together with Definition 1892.5. These force the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ row to remain downstream of the source row before the bridge can read regular sequences, finite StreamName windows, or dyadic tolerances.

Finally apply Definition 1892.47. The existing target interface exposes the readback, window, tolerance, and terminal seal rows in the displayed carrier order and confines H, C, P, N to transport, replay, provenance, and local naming. Hence a target-side bridge read is source-before-target rather than a direct read of the terminal seal. □

Theorem 1892.49 (UniformCompletionFunctor bridge route)**.** *Every accepted* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *carrier*

$$\mathcal{F} = (U, F, E, R, W, D, S, H, C, P, N)$$

packages a bridge route from its public completion-facing NameCert *export to the L10 completion schema through exactly the displayed rows*

$\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, Dyadi

The bridge first exposes the uniform-completion source U, then the Cauchy-completion handoff F, the universal-property row E, the regular readback R, the finite window row W, the dyadic tolerance ledger D, and the terminal Real seal S before any transport, replay, provenance, or local naming row is exported.

Lean4 ✓ `BEDC.Derived.UniformCompletionFunctorUp.UniformCompletionFunctorBridgeRoute`

Proof. Start with the public export of Definition 1892.43. It already binds every completion-facing consumer to the displayed source, functorial handoff, universal-extension, readback, finite-window, dyadic-tolerance, Real-seal, transport, replay, provenance, and local naming rows.

Apply the bridge schema of Definition 1892.44. That schema reads the same public route as a standard bridge surface: $\mathsf{UniformCompletion}^{\uparrow}$ supplies the source, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ supplies the handoff, and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ supplies the extension ledger before L10 readback rows are consumed.

Use the target interface of Definition 1892.47. It keeps the target-side bridge ordered through R, W, D, S and confines H, C, P, N to componentwise transport, displayed replay, provenance, and local naming.

Thus the bridge route is exactly the finite package named in the statement. The carrier contains no quotient completion equality, selected limit, host complete uniform space, open-cover compactness theorem, or choice-selected representative, so none can be exported by the bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionFunctor}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionFunctor}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 1892.1, Definition 1892.2, Definition 1892.23, Definition 1892.34, Definition 1892.42, Definition 1892.43, Definition 1892.47, and Definition 1892.49 close the finite bridge route through $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host complete uniform spaces, open-cover compactness, choice principles, Lean verification, or bridge checking.

Upgrade path: The mature L10 functorial handoff is complete on the paper axis; Lean verification and bridge checking remain unclaimed.

Constructive story: A $\mathsf{UniformCompletionFunctor}^{\uparrow}$ packet is generated from a $\mathsf{UniformCompletion}^{\uparrow}$ source row, a $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff row, a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ extension ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1893

A Concrete Naming Certificate for $\mathsf{NatRealEmbedding}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{NatRealEmbedding}^{\uparrow}$ packet records the finite route by which a displayed $\mathsf{Nat}^{\uparrow}$ unary source row becomes a $\mathsf{Real}^{\uparrow}$ name through an exact rational constant. It sits over the natural-number certificate of chapter 28, the rational source surface of chapter 37, the rational-to-real constant route of chapter 1849, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic zero-tail tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet routes a displayed $\mathsf{Nat}^{\uparrow}$ unary source row into a rational constant row, constant $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ zero-tail tolerance, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Definition 1893.1 (Nat-real embedding carrier)**.** *A* $\mathsf{NatRealEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (n, q, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *n is the* $\mathsf{Nat}^{\uparrow}$ *unary source row naming the natural-number input.*
2. *q is the* $\mathsf{Rat}^{\uparrow}$ *rational constant row obtained by reading n as $n/1$.*
3. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule whose admitted finite windows all read the same rational row q.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the constant schedule S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero-tail tolerance ledger attached to the finite reads of R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after n, q, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the natural source, rational constant row, constant stream schedule, regular readback, dyadic tolerance ledger, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a nat-real embedding request through $n, q, S, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{NatRealEmbedding}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NatRealEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $n, q, S, R, D, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host $\mathsf{Nat}^{\uparrow}$ *equality, host real equality, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1893.2 (Nat-real embedding NameCert obligations)**.** *For every accepted* $\mathsf{NatRealEmbedding}^{\uparrow}$ *carrier $K = (n, q, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NatRealEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the natural source, rational constant row, constant stream windows, regular rational readback, dyadic zero-tail tolerance ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *source admission, requiring every exported constant to pass through the displayed* $\mathsf{Nat}^{\uparrow}$ *row n before the rational row q is read;*
3. *constant-window exactness, requiring every admitted* $\mathsf{StreamName}^{\uparrow}$ *window in S to replay the same rational constant row q and no independent tail schedule;*
4. *regular-readback admission, requiring R to expose the constant rational name only through the displayed windows in S;*
5. *dyadic zero-tail tolerance, requiring D to answer finite regularity requests by the zero-tail ledger attached to q, S, R;*
6. *Real seal nonescape, requiring E to be downstream of n, q, S, R, D and to expose no quotient stream, selected limit, host* $\mathsf{Nat}^{\uparrow}$ *equality, host real equality, trichotomy, countable-choice selector, ambient completeness theorem, Lean verification credential, or bridge credential;*
7. *ledger exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the same finite nat-real embedding packet.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.NatRealEmbeddingUp.NatRealEmbeddingTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1893.1. The accepted packet displays exactly the natural source, rational constant row, constant windows, regular readback, dyadic zero-tail ledger, terminal real seal, and structural rows.

Source admission and constant-window exactness are forced by row order. A real-facing constant first reads n, then the rational row q, and every window in S is attached to that same rational row rather than to a separate stream seed.

The regular-readback and dyadic-tolerance rows are downstream. The row R reads the constant windows as a regular rational name, while D supplies the zero-tail tolerance ledger for finite regularity tests.

The real seal and ledger rows cannot add external principles. The row E is reached only after n, q, S, R, D, and the rows H, C, P, N preserve, replay, source, and name the same displayed packet. Since none of the excluded objects appears as a coordinate, the listed rows exhaust the local surface. □

Theorem 1893.3 (Nat-real embedding Real seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{NatRealEmbedding}^{\uparrow}$ *carrier $K = (n, q, S, R, D, E, H, C, P, N)$ factors through the finite route*

$$n, q, S, R, D \longmapsto E$$

before transport, replay, provenance, or local naming can be read. Consequently the embedding of a natural-number constant into the real-name layer is witnessed by the displayed $\mathsf{Nat}^{\uparrow}$ *source, rational constant row, constant* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *zero-tail tolerance ledger, not by quotienting streams or selecting a limit.*

Proof. Start with an exported real seal. The carrier has only one real-facing coordinate, namely E, and that row is listed after the natural source, rational constant, constant-window, regular-readback, and dyadic-ledger rows.

The source row n fixes the natural-number input. The rational row q records the exact constant $n/1$, and the stream row S opens only finite windows over that same rational row.

The regular readback row R reads those windows as a constant regular rational name. The dyadic ledger D records the zero-tail tolerance facts needed by the displayed regularity requests.

Only after these five rows are fixed may E be read as a $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming preserve that same order, so the exported seal factors through n, q, S, R, D before any structural row is consulted. □

Closure status for $\mathsf{NameCert}_{\mathsf{NatRealEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NatRealEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1893.1, Definition 1893.2, and Definition 1893.3 seed the finite natural-constant embedding route from $\mathsf{Nat}^{\uparrow}$ through a rational constant row, constant $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ zero-tail tolerance, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NatRealEmbedding}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.NatRealEmbeddingUp.NatRealEmbeddingTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host $\mathsf{Nat}^{\uparrow}$ equality, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source admission, constant-window exactness, regular-readback admission, dyadic zero-tail tolerance, Real seal nonescape, ledger exhaustion, and factorization proofs for BEDC.Derived.NatRealEmbeddingUp.

Constructive story: A $\mathsf{NatRealEmbedding}^{\uparrow}$ packet is generated from a NatUp unary source row, RatUp rational constant row, constant StreamName windows, RegSeqRat constant readback, DyadicRatCore zero-tail tolerance ledger, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 1894

A Concrete Naming Certificate for $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ packet records the reverse-triangle estimate for displayed regular rational Cauchy real names as a finite route through subtraction, absolute-value readbacks, and a dyadic tolerance ledger. It sits over the regular source rows of chapter 301, the finite window discipline of chapter 126, the dyadic absolute-value surface of chapter 293, the regular absolute-value lift of chapter 1499, the subtraction route of chapter 1696, the real absolute-value boundary of chapter 1608, and the terminal real-name seal of chapter 38. The packet exposes the finite inequality route behind $||x| - |y|| \leq |x - y|$; it is not host real order, quotient equality of Cauchy streams, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1894.1 (Regular Cauchy reverse-triangle carrier)**.** *A* $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, S, A_X, A_Y, D, A_D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two source* $\mathsf{RegSeqRat}^{\uparrow}$ *rows for displayed regular rational Cauchy real names.*
2. *S is the* $\mathsf{RegularCauchySubtraction}^{\uparrow}$ *route for $X - Y$, exposing only its source rows, sign-flip-plus-sum route, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance rows, and subtraction readback.*
3. *A_X and A_Y are the two* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *routes for the source rows X and Y.*
4. *D is the* $\mathsf{RegularCauchySubtraction}^{\uparrow}$ *route for the displayed difference $|X| - |Y|$, using the readbacks exported by A_X and A_Y.*
5. *A_D is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *route for the regular readback of $|X| - |Y|$.*
6. *L is the dyadic comparison ledger whose finite rows read the endpoint-and-tolerance data from A_X, A_Y, D, A_D, S and record the inequality window $||x| - |y|| \leq |x - y|$ at each requested dyadic tolerance.*
7. *R is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the bounded absolute difference, assembled only after the subtraction and absolute-value rows have been displayed.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R and the finite comparison ledger L have been exposed.*

9. *H records componentwise* hsame *transport for the two source rows, subtraction routes, absolute-value routes, comparison ledger, readback, seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from any reverse-triangle-facing precision request through $X, Y, S, A_X, A_Y, D, A_D$*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{RegularCauchyAbs}^{\uparrow}$, $\mathsf{RegularCauchySubtraction}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReverseTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, A_X, A_Y, D, A_D, L, R, E, H, C, P, N$. The carrier has no coordinate for host real order, quotient Cauchy equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 (definition) `BEDC.Derived.RegularCauchyReverseTriangleUp.RegularCauchyReverseTriangleUp`

Theorem 1894.2 (Regular Cauchy reverse-triangle NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ *carrier*

$$T = (X, Y, S, A_X, A_Y, D, A_D, L, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReverseTriangle}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the $X-Y$ subtraction route, the absolute-value routes for both sources, the subtraction route for $|X|-|Y|$, the absolute-value route for that difference, the dyadic comparison ledger, the resulting readback, and the structural rows;*

2. *subtraction handoff, requiring S and D to expose their outputs only through the finite* $\mathsf{RegularCauchySubtraction}^{\uparrow}$ *route;*

3. *absolute-value handoff, requiring A_X, A_Y, A_D to expose their outputs only through the finite* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *route and its* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint-and-tolerance ledger;*

4. *comparison exactness, requiring L to compare the displayed dyadic windows for $||x|-|y||$ and $|x-y|$ before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives a bounded read;*

5. *Real seal nonescape, requiring E to be downstream of R and L and to expose no host order, quotient equality, selected limit, trichotomy, or completeness row;*

6. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite reverse-triangle packet.*

No reverse-triangle obligation is stored outside these finite rows.

Theorem 1894.3 (Regular Cauchy reverse-triangle RegSeqRat handoff)**.** *For every accepted* $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ *carrier $T = (X, Y, S, A_X, A_Y, D, A_D, L, R, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R is assembled only after the packet displays the source rows X, Y, the subtraction readback for $X-Y$, the absolute-value readbacks for X, Y, the subtraction readback for $|X|-|Y|$, the absolute-value readback for that difference, and the dyadic comparison ledger L. Thus a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of the reverse-triangle estimate reads the finite route*

$$X, Y \longmapsto S, A_X, A_Y \longmapsto D, A_D \longmapsto L \longmapsto R$$

through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{RegularCauchySubtraction}^{\uparrow}$, *and* $\mathsf{RegularCauchyAbs}^{\uparrow}$, *with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Theorem 1894.4 (Regular Cauchy reverse-triangle Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealAbs}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ *carrier reaches the seal row* E *only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *and the comparison ledger* L *have been displayed. The real-facing boundary exposes the finite source rows, subtraction routes, absolute-value routes, dyadic tolerance comparison ledger for* $||x| - |y|| \leq |x - y|$*, regular readback, componentwise transport, continuation replay, provenance, and local naming. It exposes no host real order, quotient Cauchy equality, selected limit, trichotomy witness, ambient completeness principle, Lean verification, or public bridge row.*

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReverseTriangle}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReverseTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1894.1, Definition 1894.2, Definition 1894.3, and Definition 1894.4 seed the finite reverse-triangle route through two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, subtraction and absolute-value routes, dyadic comparison ledger, resulting readback, and $\mathsf{Real}^{\uparrow}$ seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReverseTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real order, quotient Cauchy equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subtraction handoff, absolute-value handoff, comparison exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyReverseTriangleUp.

Constructive story: A $\mathsf{RegularCauchyReverseTriangle}^{\uparrow}$ packet is generated from two RegSeqRat source rows, finite StreamName windows, subtraction routes for $X - Y$ and $|X| - |Y|$, RegularCauchyAbs routes for X, Y, and $|X| - |Y|$, DyadicAbsoluteValue tolerance rows, a comparison ledger for $||x| - |y|| \leq |x - y|$, a RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Chapter 1895

A Concrete Naming Certificate for $\mathsf{CantorFunction}^{\uparrow}$

Following Cantor and constructive analysis, the $\mathsf{CantorFunction}^{\uparrow}$ packet records the Cantor ternary-prefix to binary-prefix function as finite BEDC data. It depends on the retained-prefix and deleted-gap carrier of chapter 1530, the graph and pointwise-modulus surface of chapter 130, the compact source discipline of chapter 129, the dyadic readback ledger of chapter 418, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet exposes only finite ternary-prefix inspection, binary-prefix output, graph-row routing, transport, continuation, provenance, and local naming; it does not import measure theory, singularity analysis, host function extensionality, quotient real equality, or selected infinite branches.

Definition 1895.1 (Cantor function carrier)**.** *A* $\mathsf{CantorFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, B, G, E, H, C, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{CantorSet}^{\uparrow}$ *source-prefix row, naming a retained finite ternary prefix and the displayed deleted-gap checks inherited from the Cantor set certificate.*
2. *T is the ternary retained-prefix ledger, recording the left/right retained-third decisions that have been inspected and the finite depth at which the packet is read.*
3. *B is the binary output-prefix ledger, recording the corresponding finite binary prefix and the dyadic precision window consumed by* $\mathsf{BinaryExpansion}^{\uparrow}$ *readers.*
4. *G is the graph row, pairing the displayed source prefix with the displayed output prefix before any real-facing seal is opened.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the binary output ledger through the regular readback route exposed by* $\mathsf{BinaryExpansion}^{\uparrow}$*.*
6. *H stores componentwise* hsame *transport for the Cantor source prefix, ternary retained-prefix ledger, binary output-prefix ledger, graph row, real-seal row, continuation replay, provenance, and local naming row.*
7. *C stores displayed* Cont *replay from retained-prefix requests to binary-prefix output, from binary-prefix output to the graph row, and from the graph row through the continuous-map and real-seal handoff rows.*

8. *P is the* Pkg *provenance over* $\mathsf{CantorFunction}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorFunction}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the source prefix is a retained $\mathsf{CantorSet}^{\uparrow}$ *prefix, the ternary ledger and binary ledger have the same inspected retained-branch content, the graph row pairs those displayed ledgers, and every* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer reads through G, E, H, C, P, N. It has no coordinate for measure, singularity, host equality of functions, quotient real equality, a selected infinite branch, or an ambient pointwise function outside the packet.*

Lean4 ✓ `BEDC.Derived.CantorFunctionUp.CantorFunctionTasteGate single carrier alignment`

Theorem 1895.2 (Cantor function NameCert obligations)**.** *For an accepted* $\mathsf{CantorFunction}^{\uparrow}$ *carrier $F = (S, T, B, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, prefix classifier stability, monotone prefix exactness, graph-row transport, continuous-map handoff, real-seal nonescape, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission projects the finite tuple of Definition 1895.1. The source row S and retained-prefix ledger T are accepted only through the $\mathsf{CantorSet}^{\uparrow}$ prefix classifier, so every source-side observation is already finite.

Prefix classifier stability is read componentwise. A shared retained ternary prefix transports through H, and the continuation row C sends it to the unique displayed binary prefix row B before any graph or real-seal row is inspected.

Monotone prefix exactness and graph-row transport are then forced by B and G. Extending a retained prefix appends only the corresponding binary output bits, and the graph row pairs the two displayed prefixes without adding a host function graph.

Finally, the continuous-map and real-facing obligations are downstream of the graph row. The rows E, H, C, P, N expose the $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff route while preventing measure data, singularity data, host function extensionality, quotient real equality, selected infinite branches, or bridge checking from entering the certificate. □

Theorem 1895.3 (Cantor function prefix monotonicity)**.** *Let $F = (S, T, B, G, E, H, C, P, N)$ be an accepted* $\mathsf{CantorFunction}^{\uparrow}$ *carrier. If a retained ternary prefix row t' refines a retained ternary prefix row t inside T, then the binary output prefix row attached to t' refines the binary output prefix row attached to t inside B. This refinement is internal to the prefix ledgers and graph row: it cannot introduce host function equality, measure data, singularity data, quotient real equality, or a selected infinite branch.*

Proof. The retained-prefix relation is a finite comparison inside T. Since the carrier admits source prefixes only through S and the $\mathsf{CantorSet}^{\uparrow}$ classifier, both t and t' remain displayed retained-branch rows.

The output prefix attached to a retained row is read through the continuation route from T to B. Refining the source prefix therefore exposes a longer finite binary ledger entry rather than a different output object.

The graph row G only pairs the displayed source and output prefixes. It contains no coordinate for an ambient function, measure computation, singularity assertion, quotient real comparison, or branch selection, so the monotonic statement is exhausted by the two finite ledgers and their transport rows. □

Theorem 1895.4 (Cantor function continuous-map handoff)**.** *Every* $\mathsf{ContinuousMap}^{\uparrow}$ *read of an accepted* $\mathsf{CantorFunction}^{\uparrow}$ *packet factors through the finite graph and prefix rows before the* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, the read route is*

$$S \longmapsto T \longmapsto B \longmapsto G \longmapsto E$$

together with the componentwise transport, continuation replay, provenance, and local naming rows H, C, P, N*.*

Proof. A $\mathsf{ContinuousMap}^{\uparrow}$ consumer enters the packet through the displayed replay row C. The first available source data are the $\mathsf{CantorSet}^{\uparrow}$ source-prefix row S and the retained-prefix ledger T.

The continuation route then reads the binary output-prefix ledger B. By Definition 1895.3, stronger source requests remain on the same monotone prefix route.

Only after those two prefix ledgers have been displayed can the consumer inspect the graph row G. The row G is the handoff surface to $\mathsf{ContinuousMap}^{\uparrow}$, and the real-facing row E is downstream of it through $\mathsf{BinaryExpansion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$.

The transport, provenance, and local naming rows H, P, N record this same finite route. Since the carrier lacks host function extensionality, quotient real equality, measure and singularity coordinates, and selected infinite branches, every continuous-map read factors through the displayed graph and prefix rows before the real seal. □

Theorem 1895.5 (Cantor function uniform-modulus route)**.** *Every tolerance request for the* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{CantorFunction}^{\uparrow}$ *packet is routed through a finite retained ternary-prefix depth, the corresponding binary-prefix ledger, the graph row, and the existing* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, the modulus-facing route is*

$$S, T \longmapsto B \longmapsto G \longmapsto E, H, C, P, N,$$

with the retained-prefix depth controlling the binary output window before any real-facing consumer reads the seal.

Proof. Begin with a tolerance request at the continuous-map or real-facing boundary. The carrier of Definition 1895.1 gives no direct host-function row, so the request must first expose the source prefix row S and the retained-prefix ledger T.

Apply Definition 1895.3. Refining the retained ternary prefix refines the binary output prefix inside B, so a finite retained-prefix depth supplies the binary window needed by the tolerance request.

Now use the continuous-map handoff of Definition 1895.4. The graph row G is inspected only after the source and output ledgers have been displayed, and the real-facing row E remains downstream through the $\mathsf{BinaryExpansion}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ route.

The structural rows H, C, P, N transport and name the same finite route. Since measure data, singularity data, host function extensionality, quotient real equality, and selected infinite branches are absent from the carrier, the uniform-modulus read has no alternate escape from the retained-prefix ledger to the real seal. □

Falsifiable prediction: Every $\mathsf{CantorFunction}^{\uparrow}$ continuous-map or real-facing read factors through one CantorSet source prefix, one ternary retained-prefix ledger, one binary output-prefix ledger, one graph row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cantorset, continuousmap, compactmetric, binaryexpansion, real: $\mathsf{CantorFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cantor source-prefix row, ternary retained-prefix ledger, binary output-prefix ledger, graph row, RealUp seal, transport, replay, provenance, and local naming for the ternary-to-binary prefix function.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1895.1, Definition 1895.2, Definition 1895.3, and Definition 1895.4 seed the finite Cantor ternary-prefix to binary-prefix function carrier through source-prefix, output-prefix, graph, continuous-map handoff, and $\mathsf{Real}^{\uparrow}$ seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure theory, singularity analysis, host function extensionality, quotient real equality, selected infinite branches, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix classifier stability, monotone prefix exactness, graph-row transport, continuous-map handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorFunctionUp.

Constructive story: A $\mathsf{CantorFunction}^{\uparrow}$ packet is generated from one CantorSet source prefix, one ternary retained-prefix ledger, one binary output-prefix ledger, one graph row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1896

A Concrete Naming Certificate for $\mathsf{CauchyBornology}^{\uparrow}$

Following Bourbaki and Kelley on bornologies and Bishop–Bridges on bounded constructive metric data, the $\mathsf{CauchyBornology}^{\uparrow}$ packet records the finite bounded-family surface needed by the Real and Completion route before any consumer asks for ambient completeness or a selected bound. It sits between the dyadic arithmetic of chapter 291, finite observations from chapter 126, regular rational readback from chapter 301, ball witnesses from chapter 1450 and chapter 355, effective Cauchy source data from chapter 1705, rate budgets from chapter 929, complete-metric consumers from chapter 128, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a bounded-family NameCert interface, not a host supremum theorem, quotient-stream equality, countable-choice schedule, selected bound, or ambient Real completeness principle.

Definition 1896.1 (CauchyBornology carrier)**.** *A* $\mathsf{CauchyBornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B_{\mathrm{cau}} = (F, R, M, U, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded-family source row. It lists the finite family requests whose members may be inspected by Real-facing or Completion-facing consumers.*
2. *R is the dyadic radius-budget row read through* $\mathsf{DyadicRatCore}^{\uparrow}$*. It stores the displayed centers, radii, slack conversions, and comparison budgets assigned to the accepted family requests.*
3. *M is the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{RationalBall}^{\uparrow}$ *witness ledger, recording that each accepted source request is routed through the corresponding displayed ball witness and radius comparison.*
4. *U is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite observation windows that may be opened for the accepted family before regular rational readback is exposed.*
5. *W is the* $\mathsf{CauchyRateBudget}^{\uparrow}$ *row. It records the monotone tolerance-depth schedule used to decide which windows of U are late enough for the displayed radius budget R.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the bounded-family source, radius budget, ball witness ledger, stream windows, and rate schedule have been displayed.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached downstream of F, R, M, U, W, Q.*
8. *H records componentwise* hsame *transport for the source, radius budget, ball witnesses, stream windows, rate schedule, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any bounded-family request through $F, R, M, U, W, Q, A, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyBornology}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyRateBudget}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBornology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, R, M, U, W, Q, A, H, C, P, N$. The carrier has no coordinate for a host supremum, a selected bound, quotient-stream equality, countable choice, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyBornologyUp.CauchyBornologyTasteGate single carrier alignment`

Theorem 1896.2 (CauchyBornology NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBornology}^{\uparrow}$ *carrier*

$$B_{\mathrm{cau}} = (F, R, M, U, W, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyBornology}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the bounded-family source, dyadic radius budgets,* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{RationalBall}^{\uparrow}$ *witnesses,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{CauchyRateBudget}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *classifier stability, requiring accepted family reads in F to be preserved componentwise by the displayed transport row H;*
3. *bounded-family ledger exactness, requiring every accepted family request to pass through the dyadic radius budget R and the ball witness ledger M before any regular readback is exposed;*
4. *radius-budget monotonicity, requiring each deeper tolerance request in W to use a radius budget in R at least as late and at least as tight as the displayed predecessor row;*
5. *completion-facing handoff, requiring any* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{CauchyRateBudget}^{\uparrow}$ *consumer to read Q and A only after F, R, M, U, W have been displayed;*
6. *non-escape, requiring P and N to export no host supremum, selected bound, quotient-stream equality, countable choice, open-cover compactness, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No CauchyBornology NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 1896.1. The projected rows are exactly the bounded-family source F, dyadic radius budget R, ball witness ledger M, stream windows U, rate schedule W, regular readback Q, Real seal A, and structural rows H, C, P, N.

Classifier stability is componentwise. The row H transports accepted family requests, radius budgets, ball witnesses, windows, schedules, readbacks, seals, replay, provenance, and local naming without adding a host-family predicate.

Ledger exactness is forced by the order of the displayed route. A family request first opens F, then reads its dyadic budget from R, then verifies the corresponding $\mathsf{MetricBall}^{\uparrow}$ or $\mathsf{RationalBall}^{\uparrow}$ witness in M before any $\mathsf{RegSeqRat}^{\uparrow}$ row can be consumed.

Radius-budget monotonicity is the local content of W read against R. Each deeper tolerance-depth request must replay the predecessor budget and then read the displayed later budget, so the schedule cannot jump to an unrecorded radius or to an ambient bound.

Finally, the completion-facing handoff is exhausted by Q and A. Since P and N package only the listed rows, the excluded host supremum, selected bound, quotient-stream equality, countable choice, open-cover compactness, trichotomy, and ambient Real completeness data have no coordinate from which to escape. □

Theorem 1896.3 (CauchyBornology bounded-tail handoff)**.** *Every completion-facing bounded-tail read of an accepted* $\mathsf{CauchyBornology}^{\uparrow}$ *carrier factors through the displayed route*

$$F, R, M, U, W, Q, A \longmapsto H, C, P, N.$$

*Here F, R, M provide the Cauchy filter or basis request, the **??** bounded-family row, and the ball-witness ledger; U provides finite* $\mathsf{StreamName}^{\uparrow}$ *windows; R, W provide the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and late-window schedule; Q provides* $\mathsf{RegSeqRat}^{\uparrow}$ *readback; A provides the* $\mathsf{Real}^{\uparrow}$ *seal; and H, C, P, N provide componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. The handoff does not select limits, form quotient filters, use countable choice, or appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The completion-facing rows are exactly the bounded-family source F, dyadic radius budget R, ball witness ledger M, stream windows U, rate schedule W, regular readback Q, Real seal A, and structural rows H, C, P, N.

Read the bornology and metric rows next. The source and radius data are bounded only through F, R, M, so the tail request remains a displayed bounded-family read rather than a quotient filter or an ambient family predicate.

Use the sequence-facing rows U, W, Q to expose finite StreamName windows, late-window tolerance control, and RegSeqRat readback. These rows are the sibling Cauchy and metric handoff surface before the terminal Real seal A is available.

Finally replay through H, C, P, N. Componentwise transport, Cont replay, provenance, and local naming preserve the same finite route, and no row supplies selected limits, countable choice, quotient filters, or ambient Real completeness. □

Theorem 1896.4 (CauchyBornology completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*, or* $\mathsf{CauchyRateBudget}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyBornology}^{\uparrow}$ *carrier reads bounded-family data only through the displayed route*

$$F \longmapsto R \longmapsto M \longmapsto U \longmapsto W \longmapsto Q \longmapsto A \longmapsto H, C, P, N.$$

Consequently the consumer obtains finite bounded-family source rows, dyadic radius budgets, ball witnesses, stream windows, rate schedules, regular readback, the terminal Real seal, transport, replay, provenance, and local naming, and obtains no host supremum, selected bound, quotient-stream equality, countable choice, open-cover compactness, trichotomy, or ambient Real completeness.

Proof. Start from an accepted consumer request. By Definition 1896.1, the only bounded-family source coordinate is F, and every analytic row downstream of it is displayed inside the same finite packet.

The radius and ball data are next. The dyadic budget R supplies the radius, slack, and comparison rows, while M records the corresponding $\mathsf{MetricBall}^{\uparrow}$ and $\mathsf{RationalBall}^{\uparrow}$ witnesses. Thus no consumer can read boundedness before the displayed radius and ball ledgers are opened.

The sequence-facing data are downstream of those ledgers. The $\mathsf{StreamName}^{\uparrow}$ row U gives the finite windows, the $\mathsf{CauchyRateBudget}^{\uparrow}$ row W gives the late-window schedule, and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q is available only after F, R, M, U, W have been displayed.

The terminal handoff then reaches the $\mathsf{Real}^{\uparrow}$ seal A and the structural rows H, C, P, N. These rows preserve the same route for $\mathsf{EffectiveCauchySequence}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers, so the packet exports only the listed bounded-family data and none of the excluded host principles. □

Theorem 1896.5 (CauchyBornology bounded-family non-escape)**.** *Let $B_{\mathrm{cau}} = (F, R, M, U, W, Q, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyBornology}^{\uparrow}$ *carrier. Any bounded-family fact exported from B_{cau} is a row-wise consequence of the displayed source, dyadic radius budget, ball witness ledger, stream window, rate schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows. It is not a host supremum, selected bound, quotient-stream equality, countable choice, open-cover compactness, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The analytic coordinates are exactly F, R, M, U, W, Q, A, and the structural coordinates are exactly H, C, P, N.

A bounded-family read has no other entry point. If the read concerns a member of the family, it begins at F; if it concerns a radius, it passes through R; if it concerns containment, it passes through M; and if it concerns Cauchy or completion data, it proceeds through U, W, Q, A.

The structural rows preserve only those coordinates. Componentwise hsame transport and displayed Cont replay cannot manufacture a host supremum or a selected bound, while provenance and local naming record only the finite packet.

Therefore every exported bounded-family fact is confined to the displayed rows, and each excluded host principle has no coordinate, replay edge, provenance entry, or local NameCert row. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBornology}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBornology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1896.1, Definition 1896.2, Definition 1896.4, and Definition 1896.5 seed the finite bounded-family NameCert surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyRateBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host supremum theorem, selected bounds, quotient-stream equality, countable choice, open-cover compactness, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier stability, bounded-family ledger exactness, radius-budget monotonicity, completion-facing handoff, provenance, local naming, and non-escape for BEDC.Derived.CauchyBornologyUp

Constructive story: A $\mathsf{CauchyBornology}^{\uparrow}$ packet is generated from bounded-family source rows, displayed dyadic radius budgets, $\mathsf{MetricBall}^{\uparrow}$ and $\mathsf{RationalBall}^{\uparrow}$ witnesses, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{CauchyRateBudget}^{\uparrow}$ schedules, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1897

A Concrete Naming Certificate for $\mathsf{ContinuousIntervalImage}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ContinuousIntervalImage}^{\uparrow}$ packet records the finite route by which the image of a connected closed bounded real interval under a displayed continuous Real-facing map remains available as a connected Real-facing image surface. It sits over the closed bounded interval boundary of chapter 1591, the connected interval route of chapter 1462, the intermediate-value bisection discipline of chapter 1402, the graph and modulus rows of chapter 130, and the terminal real-name seal of chapter 38. The packet is a finite image certificate for Real and Completion consumers; it is not a host image set, an endpoint-selection principle, quotient equality of real names, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1897.1 (Continuous interval image carrier)**.** *A* $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, J, F, V, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing the displayed lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, rational bounds, endpoint-window requests, and interval membership ledger.*
2. *J is the* $\mathsf{ConnectedInterval}^{\uparrow}$ *row for the same source interval, recording the finite connectedness handoff consumed by image-connectedness reads.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus row, carrying the displayed Real-facing map, finite graph observations, modulus budget, and compatibility with the source interval windows.*
4. *V is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row, naming the bisection and sign-change surface used whenever an image request must pass between two displayed values.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for source endpoint names, graph observations, bisection probes, and image output reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational approximants used by the map, intermediate-value route, and output seal.*
7. *E is the output* $\mathsf{Real}^{\uparrow}$ *seal row; it is reached only after the source interval, connectedness row, continuous-map row, intermediate-value route, finite windows, and regular readbacks have been replayed.*

8. *H records componentwise* hsame *transport for the closed bounded interval, connected interval row, continuous map, intermediate-value route, window row, readback row, output seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from an image request through $B, J, F, V, W, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* ContinuousIntervalImage$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, ConnectedInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, IntermediateValue$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ContinuousIntervalImage}^{\uparrow}}$ *row.*

The classifier compares accepted image packets only through the displayed rows $B, J, F, V, W, R, E, H, C, P, N$. It has no coordinate for a host image subset, selected extrema, endpoint-choice functionals, quotient equality of real names, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContinuousIntervalImageUp.ContinuousIntervalImageTasteGate single carrier alignment`

Theorem 1897.2 (Continuous interval image NameCert obligations)**.** *For every accepted* ContinuousIntervalImage$^{\uparrow}$ *carrier*

$$I = (B, J, F, V, W, R, E, H, C, P, N),$$

the local NameCert$_{\text{ContinuousIntervalImage}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded interval row, connected interval row, continuous-map graph and modulus row, intermediate-value sign route, finite window row, regular readback row, output Real seal, transport, replay, provenance, and local naming;*
2. *source-interval exactness, requiring every endpoint, rational bound, membership, and interval-window request to pass through the displayed* ClosedBoundedInterval$^{\uparrow}$ *row B;*
3. *connected-source exactness, requiring connectedness reads to pass through the displayed* ConnectedInterval$^{\uparrow}$ *row J and the same source interval boundary B;*
4. *map-modulus exactness, requiring graph observations and continuity budgets to pass through the displayed* ContinuousMap$^{\uparrow}$ *row F and its finite window requests;*
5. *intermediate-value routing, requiring every image handoff between displayed source values to pass through the* IntermediateValue$^{\uparrow}$ *sign-route row V rather than through host order completeness;*
6. *output-seal ordering, requiring W and R to expose StreamName windows and RegSeqRat readbacks before the terminal* Real$^{\uparrow}$ *seal E is available;*
7. *nonescape, excluding host image subsets, selected endpoints or extrema, quotient real equality, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No continuous interval image obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1897.1. The accepted packet displays exactly one closed bounded interval source, one connected interval row, one continuous-map graph and modulus row, one intermediate-value sign route, one finite window row, one regular readback row, one output real seal, and the structural rows.

Source-interval exactness follows from B. Endpoint, rational-bound, membership, and source-window reads have no second interval coordinate, so each such read enters the image packet through the displayed $\mathsf{ClosedBoundedInterval}^{\uparrow}$ carrier.

Connected-source exactness is supplied by J over the same source row B. Since the packet contains no independent connectedness object, connected image reads must consume the connected interval route already attached to the closed bounded interval boundary.

Map-modulus exactness is supplied by F. Graph observations, continuity budgets, and compatibility checks all pass through the displayed $\mathsf{ContinuousMap}^{\uparrow}$ row before any image-output seal can be inspected.

Intermediate-value routing is supplied by V. When a consumer asks for a value between two displayed image values, the sign-route row provides the bisection surface and prevents the request from being replaced by host order completeness or a selected preimage.

The remaining rows W, R, E, H, C, P, N impose readback order and local naming. StreamName windows and RegSeqRat readbacks precede the output Real seal, while transport, replay, provenance, and naming contain none of the excluded host image, choice, quotient, trichotomy, completeness, verification, or bridge data. □

Theorem 1897.3 (Continuous interval image Real seal route)**.** *Every Real-facing output read of an accepted* $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *carrier factors through the finite route*

$$B, J, F, V \longmapsto W, R \longmapsto E, H, C, P, N.$$

Thus the consumer first receives the closed bounded source interval, its connected interval handoff, the continuous-map graph and modulus data, and the intermediate-value sign route; only then may it read finite windows, regular rational approximants, the output $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. The route exports no host image set, selected endpoint or extremum, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $I = (B, J, F, V, W, R, E, H, C, P, N)$. The source side is forced through B and J: the closed bounded interval row supplies the endpoint and membership boundary, and the connected interval row supplies the connectedness handoff over that same boundary.

The image computation is then forced through F and V. The continuous-map row provides graph observations and modulus budgets, while the intermediate-value row supplies the sign-route surface needed to pass between displayed values.

The Real-facing output is downstream of finite observation. The window row W and readback row R expose the finite stream and regular rational data before E seals the output as a $\mathsf{Real}^{\uparrow}$ name.

Finally H, C, P, N transport, replay, source, and name the same route. Since no row contains a host image subset, endpoint selector, extremum selector, quotient-real equality, trichotomy, countable choice, or ambient completeness witness, every output read remains inside the displayed factorization. □

Theorem 1897.4 (Continuous interval image connected handoff)**.** *Every connectedness read exported by an accepted* $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *carrier factors through*

$$J \longmapsto F, V \longmapsto W, R, E, H, C, P, N.$$

The read uses the source $\mathsf{ConnectedInterval}^{\uparrow}$ *row, the continuous-map graph and modulus row, the intermediate-value sign route, finite windows, regular readbacks, the output* $\mathsf{Real}^{\uparrow}$ *seal, and the structural NameCert rows. It does not introduce a separate host connected image object or a choice-selected preimage.*

Proof. The source connectedness coordinate is J, and it is tied to the closed bounded interval row B in the carrier. A connectedness read therefore begins with the displayed $\mathsf{ConnectedInterval}^{\uparrow}$ handoff rather than a host connected-subset assertion.

Continuity transport is then read through F. The graph and modulus rows determine how source-window data are carried to image-window data, while V supplies the intermediate-value sign route needed to keep the image between displayed values connected at the finite certificate level.

The output side is read only after W, R, E have been replayed. These rows expose finite StreamName windows, RegSeqRat approximants, and the output Real seal before the structural rows H, C, P, N transport, replay, source, and name the packet.

Because the packet has no coordinate for a host image set, selected preimage, endpoint selector, quotient real equality, trichotomy, countable choice, or ambient completeness, the displayed handoff is the whole connectedness surface exported by the certificate. □

Falsifiable prediction: Every $\mathsf{ContinuousIntervalImage}^{\uparrow}$ read factors through one $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source row, one $\mathsf{ConnectedInterval}^{\uparrow}$ source-connectedness row, one $\mathsf{ContinuousMap}^{\uparrow}$ graph and modulus row, one $\mathsf{IntermediateValue}^{\uparrow}$ sign-route row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one output $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closedboundedinterval, connectedinterval, continuousmap, intermediatevalue, real: $\mathsf{ContinuousIntervalImage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source interval boundary, connectedness handoff, continuous-map graph and modulus data, intermediate-value bisection route, finite readback boundary, output real seal, and local NameCert route as one finite image packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousIntervalImage}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousIntervalImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1897.1, Definition 1897.2, Definition 1897.3, and Definition 1897.4 seed the finite continuous interval image route from a closed bounded connected source interval through continuous-map modulus data, intermediate-value routing, finite windows, regular readback, output real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousIntervalImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host image-set construction, selected preimages, endpoint or extremum selection, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, connected-source exactness, map-modulus exactness, intermediate-value routing, output-seal ordering, Real-route nonescape, and connected-handoff proofs for BEDC.Derived.ContinuousIntervalImageUp.

Constructive story: A $\mathsf{ContinuousIntervalImage}^{\uparrow}$ packet is generated from a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source row, a $\mathsf{ConnectedInterval}^{\uparrow}$ source-connectedness row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph and modulus row, an $\mathsf{IntermediateValue}^{\uparrow}$ sign-route row, StreamName windows, RegSeqRat readbacks, an output $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1898

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalCompactness}^{\uparrow}$

Following Bolzano, Weierstrass, and Bishop–Bridges constructive analysis, the $\mathsf{CauchyDiagonalCompactness}^{\uparrow}$ packet records the finite diagonal subsequence boundary used by the L10 Real and Completion route before any consumer may read a compactness-shaped seal. It sits downstream of the bounded-sequence source of chapter 1415, the finite subsequence compactness discipline of chapter 1254, the regular extraction boundary of chapter 381, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite diagonal window extraction route; it does not assert open-cover compactness, choice-selected infinite subsequences, quotient stream equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1898.1 (CauchyDiagonalCompactness carrier)**.** *A* $\mathsf{CauchyDiagonalCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row, carrying the bounded sequence windows available to the diagonal construction;*
2. *F is the finite net and refinement schedule, recording only the inspected cells and refinement levels used by the diagonal route;*
3. *W is the diagonal subsequence selector-window row, displaying for each visible precision request the finite window of selected indices consumed by the packet;*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from B, F, W and read only through the finite windows of chapter 126;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic radius, slack, and comparison entries for the displayed net refinements and selected windows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after B, F, W, Q, D have been displayed;*
7. *H records componentwise* hsame *transport for the bounded source, finite refinement schedule, selector windows, regular readback, dyadic ledger, Real seal, replay row, provenance row, and local naming row;*

8. *C records displayed* Cont *replay from each finite request through $B, F, W, Q, D, E, H, P, N$;*
9. *P is the* Pkg *provenance over* CauchyDiagonalCompactness$^{\uparrow}$, BolzanoWeierstrass$^{\uparrow}$, BoundedRealSequence$^{\uparrow}$, RegularCauchySubsequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* NameCert$_{\text{CauchyDiagonalCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, W, Q, D, E, H, C, P, N$. It contains no open-cover compactness field, no selected infinite subsequence, no quotient stream equality, no host real equality, no countable-choice row, and no ambient Real$^{\uparrow}$ *completeness datum.*

Lean4 ✓ BEDC.Derived.CauchyDiagonalCompactnessUp.TasteGate.CauchyDiagonalCompactnessTasteGate single carrier alignment

Theorem 1898.2 (CauchyDiagonalCompactness NameCert obligations)**.** *For an accepted* CauchyDiagonalCompact *carrier $K = (B, F, W, Q, D, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyDiagonalCompactness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the bounded source, finite net and refinement schedule, selector-window row, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *bounded-source exactness, requiring every diagonal window to read the same* BoundedRealSequence$^{\uparrow}$ *row B that feeds the* BolzanoWeierstrass$^{\uparrow}$ *and* RegularCauchySubsequence$^{\uparrow}$ *routes;*
3. *finite-refinement exactness, requiring F to list the inspected net cells and refinement levels before any selector window is exposed;*
4. *readback compatibility, requiring Q to be the only* RegSeqRat$^{\uparrow}$ *readback obtained from the selected windows and requiring D to be the only dyadic tolerance ledger consumed by that readback;*
5. *Real seal ordering, requiring E to be displayed only after B, F, W, Q, D;*
6. *route stability, requiring H, C, P, N to preserve the displayed packet without adding open-cover compactness, choice-selected subsequences, quotient stream equality, host real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 1898.1. The tuple displays the bounded source, finite refinement schedule, selector windows, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local name.

Bounded-source exactness follows from the single coordinate B. Since the carrier has no second sequence source, every diagonal window and downstream readback must inspect the same bounded row used by the Bolzano–Weierstrass and regular-subsequence siblings.

Finite-refinement exactness is the projection to F. The schedule stores inspected cells and refinement levels before W, Q, D, or E is available, so the selector window cannot be supplied by an external compactness oracle.

Readback compatibility and Real seal ordering follow from W, Q, D, E. The readback row Q is downstream of the selected windows, the dyadic ledger D is the only tolerance row consumed by that readback, and the terminal seal E appears only after those rows are displayed.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite route, and the carrier has no coordinate for open-cover compactness, a choice-selected infinite subsequence, quotient stream equality, host real equality, countable choice, or ambient completeness. □

Theorem 1898.3 (CauchyDiagonalCompactness window extraction)**.** *Let $K = (B, F, W, Q, D, E, H, C, P, N)$ be an accepted* CauchyDiagonalCompactness$^{\uparrow}$ *carrier. Every finite precision request that reaches the diagonal route factors through the displayed extraction spine*

$$B, F, W \longmapsto Q, D \longmapsto E.$$

Thus the packet exposes a finite selector-window extraction over bounded real sequence data, not a choice-selected infinite subsequence, quotient stream, open-cover compactness proof, or ambient completion principle.

Proof. Begin with a finite precision request. Carrier admission provides B as the only bounded sequence source and F as the only refinement schedule.

The selector-window row W is then read from B and F. It stores visible windows of selected indices, rather than an infinite selected subsequence or a host choice function.

Next the packet reads Q and D. The regular rational readback is assembled from the selected windows, and the dyadic ledger records the tolerance data consumed by that readback.

The Real seal E is downstream of this spine. The structural rows H, C, P, N preserve the same finite extraction route, so no open-cover compactness field, quotient stream, selected infinite subsequence, or ambient completion datum can enter the consumer surface. □

Theorem 1898.4 (CauchyDiagonalCompactness Real seal non-escape)**.** *For an accepted* CauchyDiagonalCompactn *carrier $K = (B, F, W, Q, D, E, H, C, P, N)$, every* Real$^{\uparrow}$ *seal read reaches E only through the bounded source B, finite net and refinement schedule F, selector-window row W,* RegSeqRat$^{\uparrow}$ *readback Q, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger D. The seal exports no open-cover compactness, choice-selected subsequence, quotient stream equality, host real equality, countable-choice row, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with a Real$^{\uparrow}$ seal read. The carrier definition supplies E as the only terminal Real seal coordinate.

The NameCert obligations place E after B, F, W, Q, D. A consumer must therefore traverse the bounded source, finite refinement schedule, selector window, regular readback, and dyadic tolerance ledger before any Real seal can be read.

The transport and replay rows H and C preserve that same order, while P and N source and name the packet. Since the carrier contains no open-cover compactness field, selected infinite subsequence, quotient stream equality, host real equality, countable-choice row, or ambient completeness datum, the Real seal is exhausted by the displayed finite route. □

Closure status for NameCert$_{\text{CauchyDiagonalCompactness}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchyDiagonalCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 1898.1, Definition 1898.2, Definition 1898.3, and Definition 1898.4 seed the finite diagonal compactness packet over BoundedRealSequence sources, BolzanoWeierstrass refinement discipline, RegularCauchySubsequence windows, StreamName inspection, RegSeqRat readback, DyadicRatCore tolerances, and Real seals.

Formal verification: FormalStatus(NameCert$_{\text{CauchyDiagonalCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected infinite subsequences, quotient stream equality, host real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exactness, finite-refinement exactness, selector-window extraction, readback compatibility, Real seal nonescape, and route stability for BEDC.Derived.CauchyDiagonalCompac

Constructive story: A CauchyDiagonalCompactness$^{\uparrow}$ packet is generated from a BoundedRealSequence$^{\uparrow}$ source, a finite net and refinement schedule, displayed diagonal selector windows, a RegSeqRat$^{\uparrow}$ readback row, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1899

A Concrete Naming Certificate for $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$

Following Picard, Lindelof, and Bishop–Bridges, the $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ packet records the local-flow part of constructive ODE existence and uniqueness as finite NameCert data. It refines the ODE source of chapter 193, the Lipschitz vector-field ledger of chapter 410, the contraction-map surface of chapter 1418, the finite Picard route of chapter 395, the complete-metric limit handoff of chapter 128, the stream naming discipline of chapter 126, and the regular rational sequence ledger of chapter 301. The packet names a local flow row over a displayed time-state window; it does not select a global solution curve, quotient completed paths, assert ambient real completeness, or compare host functions extensionally.

Definition 1899.1 (Picard–Lindelof local-flow carrier)**.** *A* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (O, F, K, L, M, S, R, H, C, G, N)$$

with the following visible rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *local-flow source row, including the initial time-state cell, the local time window, and the displayed endpoint row to be named;*
2. *F is the* $\mathsf{LipschitzMap}^{\uparrow}$ *vector-field ledger on the same local window, with graph reads and a displayed Lipschitz bound;*
3. *K is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row obtained by reading the Picard operator as a contraction on the local approximant surface;*
4. *L is the* $\mathsf{PicardContraction}^{\uparrow}$ *finite-iteration row, recording the successive Picard approximants and the one-step contraction comparisons that keep them inside the window;*
5. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row for the carried approximant space, containing the Cauchy modulus, limit row, and late-observation comparisons for the Picard stream;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the visible approximant stream without replacing it by a host sequence object;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational control row that records the local radius, time-step, and modulus budgets used by the contraction and completion rows;*

8. *H stores componentwise* hsame *transport for the ODE source, Lipschitz ledger, contraction row, Picard route, completion handoff, stream name, rational control row, replay rows, provenance row, and local naming row;*
9. *C stores the displayed* Cont *replay from the initial ODE row through the Lipschitz graph reads, Picard operator, finite iteration route, complete-metric handoff, and endpoint classifier;*
10. *G is the* Pkg *provenance row over* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalFlow}^{\uparrow}}$ *row.*

The carrier exposes only these finite rows. It has no coordinate for host function equality, quotient completion equality, choice-selected solution curves, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or maximal continuation.*

Lean4 ✓ `BEDC.Derived.PicardLindelofLocalFlowUp.PicardLindelofLocalFlowTasteGate single carrier alignment`

Theorem 1899.2 (Picard–Lindelof local-flow NameCert obligations)**.** *For an accepted* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ *carrier*

$$P = (O, F, K, L, M, S, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalFlow}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, ODE source compatibility for O, Lipschitz graph stability for F, contraction packaging for K, finite Picard iteration for L, complete-metric handoff for M, stream-name visibility for S, rational budget visibility for R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through G, and local naming through N.*

Proof. Carrier admission is projection from Definition 1899.1. Each accepted packet displays the ODE source row, vector-field ledger, contraction row, Picard route, completion row, stream name, rational budget, transport, replay, provenance, and local name.

The source rows O, F, R fix the local window. The ODE row supplies the initial time-state cell and endpoint row, the Lipschitz row supplies the graph reads and bound, and the rational control row supplies the time-step and radius budgets that keep the later rows inside the same window.

The iteration rows K, L, S are finite. The contraction row packages the Picard operator through chapter 1418, the Picard route records only finitely many displayed approximants through chapter 395, and the stream-name row names that visible approximant stream without selecting an external sequence.

The completion row M is a handoff to chapter 128. It contributes the Cauchy modulus, limit row, and late-observation comparisons needed by the local endpoint classifier, but it does not identify quotient representatives or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally H, C, G, N transport, replay, source, and name the same finite surface. Since all exported rows are fields of P, the certificate cannot assert host function equality, choose a solution curve, or extend the local window beyond the displayed nonescape data. □

Theorem 1899.3 (Picard–Lindelof local-flow Picard handoff)**.** *Every accepted* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ *carrier factors each local-flow endpoint read through the finite route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{LipschitzMap}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardContraction}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}.$$

The route first opens the ODE source and Lipschitz vector-field ledger, then packages the Picard operator as a contraction, then records finite Picard iteration, then hands the resulting Cauchy stream to the complete-metric row. The local uniqueness boundary compares only local-flow rows that share the same O, F, R window data and the same displayed handoff rows K, L, M, S, H, C, G, N.

Proof. Start with O and F. The ODE source fixes the local-flow problem and the Lipschitz ledger supplies the only vector-field graph reads available to the packet.

Build the finite Picard stage next. The row K reads the Picard operator as a $\mathsf{ContractionMapping}^{\uparrow}$ packet, while L records the displayed finite approximants and one-step comparisons. No step is a host function application; each step is a BHist row replayed by C and transported componentwise by H.

Use S and R to keep the iteration visible. The stream-name row names the finite prefix and its continuing ledger, and the rational control row supplies the budgets used to keep the prefix inside the nonescape window.

The complete-metric handoff is the row M. It reads the Picard stream as a Cauchy object, records the limit row and late comparisons, and returns only the endpoint row licensed by the same local window.

For uniqueness, compare two accepted packets over the same O, F, R rows. Their endpoint comparison is the local classifier induced by the shared contraction, Picard, completion, stream, transport, replay, provenance, and naming rows. The comparison therefore stays inside the local NameCert surface and does not become host function equality, quotient completion equality, a selected solution curve, or a global continuation theorem. □

Falsifiable prediction: Every $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ endpoint read factors through one Ode row, one LipschitzMap vector-field ledger, one ContractionMapping Picard-operator row, one PicardContraction finite-iteration row, one CompleteMetric handoff row, one StreamName row, one RegSeqRat budget row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ode, lipschitzmap, contractionmapping, picardcontraction, complete-metric, streamname, regseqrat: $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ODE source data, Lipschitz graph data, contraction packaging, finite Picard iteration, complete-metric handoff, stream naming, rational local-window budgets, and the transport/replay/provenance rows binding them into one local-flow certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalFlow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardLindelofLocalFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1899.1, Definition 1899.2, and Definition 1899.3 seed the constructive local-flow packet over ODE source rows, Lipschitz vector-field ledgers, contraction packaging, finite Picard iteration, complete-metric handoff, stream naming, rational local-window budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardLindelofLocalFlow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, maximal interval continuation, host function equality, quotient completion equality, choice-selected solution curves, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ODE source compatibility, Lipschitz graph stability, contraction packaging, finite Picard iteration, complete-metric handoff, stream-name visibility, rational budget visibility, transport, replay, provenance, and local naming for BEDC.Derived.PicardLindelofLocalFlowUp.

Constructive story: A $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ packet is generated from one ODE local-flow source row, one LipschitzMap vector-field ledger, one ContractionMapping Picard-operator row, one PicardContraction finite-iteration row, one CompleteMetric handoff row, one StreamName row, one RegSeqRat local-window budget row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1900

A Concrete Naming Certificate for $\mathsf{RolleTheorem}^{\uparrow}$

Following Rolle and the Bishop–Bridges reading of constructive analysis, the $\mathsf{RolleTheorem}^{\uparrow}$ packet records the finite NameCert boundary behind the familiar stationary-point theorem without turning it into an ambient completeness principle. Its source rows are the terminal real-name surface of chapter 38, the pointwise continuity and modulus rows of chapter 130, the derivative quotient route of chapter 122, the sign-change and bisection ledger of chapter 1402, the regular rational readback of chapter 301, and the dyadic tolerance arithmetic of chapter 291. The chapter names the theorem-facing finite interface only: endpoint equality, interval localization, continuity, derivative inspection, rational readback, dyadic budgets, and a displayed zero-derivative handoff remain visible rows.

Definition 1900.1 (Rolle theorem carrier)**.** *A* $\mathsf{RolleTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, E, F, Q, D, I, Z, H, C, P, N)$$

with the following displayed rows:

1. *A, B are the two endpoint real-name rows, read through the* $\mathsf{Real}^{\uparrow}$ *surface and retained with their regular rational readbacks.*
2. *E is the endpoint-equality row, recording classifier equality of the two endpoint value seals rather than host equality of completed reals.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the displayed real-valued map over the located interval cells.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite endpoint and interior value inspections consumed by the theorem route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the inspected rational windows and derivative quotients.*
6. *I is the* $\mathsf{IntermediateValue}^{\uparrow}$ *handoff row that supplies the finite bisection-facing sign and location ledger used to isolate the stationary witness.*
7. *Z is the* $\mathsf{Derivative}^{\uparrow}$ *zero-handoff row, exposing a displayed derivative quotient whose terminal seal is the* $\mathsf{Real}^{\uparrow}$ *zero seal.*

8. *H records componentwise* hsame *transport for endpoint names, endpoint equality, map data, rational readbacks, dyadic tolerances, intermediate-value handoff, derivative quotient, zero seal, replay rows, provenance rows, and local naming rows.*
9. *C records displayed* Cont *replay from endpoint reads to map windows, from rational readbacks to dyadic tolerance checks, from the intermediate-value ledger to the derivative-zero handoff, and from the derivative quotient to the terminal real-name zero seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RolleTheorem}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RolleTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted theorem packets only through these rows. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy of real numbers, a selected root, host differentiability, quotient real equality, open-cover compactness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RolleTheoremUp.RolleTheoremTasteGate single carrier alignment`

Theorem 1900.2 (Rolle theorem NameCert obligations)**.** *For every accepted* $\mathsf{RolleTheorem}^{\uparrow}$ *carrier*

$$T = (A, B, E, F, Q, D, I, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RolleTheorem}^{\uparrow}}$ *surface consists exactly of the finite obligation rows for endpoint real-name admission, endpoint-equality routing, continuous-map admission, regular rational readback, dyadic tolerance budgeting, intermediate-value handoff, derivative quotient inspection, terminal zero sealing, componentwise transport, displayed replay, provenance, and local naming. No obligation row may bypass the displayed* $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *sources.*

Proof. Carrier admission is projection from Definition 1900.1. The rows A, B, E supply the endpoint names and the equality route, while F supplies the sole continuous-map surface.

The arithmetic and localization rows are then fixed. The readback row Q and the tolerance ledger D are the only regular-rational and dyadic routes available to the packet, and I is the only intermediate-value handoff row.

Finally inspect the derivative-facing row Z. It is attached to the displayed derivative quotient and terminal zero seal, and it is transported and replayed only through H and C.

The provenance and local naming rows P, N add source and name data without adding mathematical coordinates. Since the carrier has no further field, the listed rows exhaust the NameCert surface. □

Theorem 1900.3 (Rolle theorem derivative-zero handoff)**.** *Every derivative-zero statement exported by an accepted* $\mathsf{RolleTheorem}^{\uparrow}$ *carrier factors through the ordered route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{IntermediateValue}^{\uparrow},\ \mathsf{Derivative}^{\uparrow}.$$

The exported conclusion is a displayed zero-handoff for the derivative quotient row Z, not a host differentiability claim and not a mean-value theorem over an ambient real function.

Proof. Start with an accepted carrier $T = (A, B, E, F, Q, D, I, Z, H, C, P, N)$. The endpoint real-name rows and the regular rational readback row are inspected before any derivative-zero read is available.

The dyadic ledger D bounds the finite readback windows, and the continuous-map row F fixes the displayed function data consumed by the theorem route. The intermediate-value handoff I supplies the finite bisection-facing location route used by the stationary witness.

Only after these rows are present can the derivative row Z be read. Transport through H and replay through C preserve the same order, while P and N add provenance and naming only.

Thus the derivative-zero export is the displayed handoff through Z over the named rows. The packet contains no host differentiability predicate, no ambient mean-value theorem, and no detached equality of completed functions. □

Theorem 1900.4 (Rolle theorem Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumed or exported by an accepted* $\mathsf{RolleTheorem}^{\uparrow}$ *carrier is routed through one of the displayed endpoint, readback, intermediate-value, or derivative-zero rows of Definition 1900.1. Consequently the carrier cannot export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, selected roots, host differentiability, quotient real equality, or standard-bridge data.*

Proof. Inspect the real-facing coordinates of the carrier. The endpoint rows A, B expose the initial real names, Q exposes regular rational readback, I exposes the intermediate-value handoff, and Z exposes the derivative-zero seal.

Each of these rows is paired with componentwise transport and displayed replay. The dyadic ledger D fixes the finite tolerance budget, and the continuous-map row F fixes the function surface that the route may inspect.

There is no additional coordinate for a completed-real selector, an order trichotomy witness, a selected root, a host differentiability predicate, or a bridge object. Provenance and local naming record the same row inventory.

Therefore every real seal remains inside the displayed packet boundary, and no Real or bridge claim can escape the carrier. □

Falsifiable prediction: Every $\mathsf{RolleTheorem}^{\uparrow}$ read factors through endpoint $\mathsf{Real}^{\uparrow}$ names, endpoint equality, $\mathsf{ContinuousMap}^{\uparrow}$ map data, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, $\mathsf{IntermediateValue}^{\uparrow}$ handoff, $\mathsf{Derivative}^{\uparrow}$ zero routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, continuousmap, derivative, intermediatevalue, regseqrat, dyadicratcore: $\mathsf{RolleTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds endpoint equality, theorem-facing continuity, rational readback, dyadic tolerance, intermediate-value handoff, derivative-zero routing, transport, replay, provenance, and local naming as one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RolleTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{RolleTheorem}^{\uparrow}$ packet is generated from endpoint real-name rows, endpoint equality, continuous-map data, regular rational readback, dyadic tolerance arithmetic, intermediate-value handoff, derivative-zero routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RolleTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1900.1, Definition 1900.2, Definition 1900.3, and Definition 1900.4 seed the finite theorem-facing Rolle boundary over displayed Real, continuous-map, derivative, intermediate-value, regular-rational, and dyadic rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RolleTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient RealUp completeness, trichotomy, selected roots, host differentiability, quotient real equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligation coverage, derivative-zero handoff, Real seal non-escape, and TasteGate field faithfulness for BEDC.Derived.RolleTheoremUp.

Chapter 1901

A Concrete Naming Certificate for $\mathsf{SecondCountable}^{\uparrow}$

Following Bourbaki and Kelley on general topology, the $\mathsf{SecondCountable}^{\uparrow}$ packet records a countable-base interface as finite BEDC rows. It sits over the metric source surface of chapter 57, the generated-open classifier of chapter 88, the finite observation windows of chapter 126, the dyadic precision ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed base enumeration, local basis reads, refinement rows, and transport ledgers; it does not identify host sets, assert open-cover compactness, select a basis by choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1901.1 (Second-countable carrier)**.** *A* $\mathsf{SecondCountable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, T, M, B, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the point or source carrier row whose visible histories supply the only points read by the packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-set classifier row, recording generated-open membership only through displayed BEDC histories.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *metric-ball witness row when the packet is metric-presented; if no metric row is present, M is the explicit empty witness row.*
4. *B is the displayed countable-base enumeration row, recording finite index windows and the basic-open cells exposed at those windows.*
5. *D is the dyadic radius and precision ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$ *used by metric-ball and boundary reads.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for finite inspection windows of the base enumeration and consumer probes.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary and seal row used by radius, completion-facing, or real-valued classifier reads.*
8. *H stores componentwise* hsame *transport for the source, topology classifier, metric witness, base enumeration, dyadic ledger, stream windows, real seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a point or source read through the base enumeration, local basis test, refinement row, classifier transport, precision ledger, stream window, real seal, provenance, and local name.*

10. *P is the* Pkg *provenance row over* $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SecondCountable}^{\uparrow}}$ *row.*

The carrier exposes no host set of opens, no selected countable basis, no countable-choice witness, no open-cover compactness proof, no Lindelof theorem, and no $\mathsf{Real}^{\uparrow}$ *completeness field.*

Lean4 ✓ `BEDC.Derived.SecondCountableUp.SecondCountableTasteGate single carrier alignment`

Theorem 1901.2 (Second-countable NameCert obligations)**.** *For an accepted* $\mathsf{SecondCountable}^{\uparrow}$ *carrier*

$$Q = (X, T, M, B, D, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SecondCountable}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the point or source row, topology classifier, optional metric-ball witness, displayed base enumeration, dyadic ledger, stream windows, real seal, transport, replay, provenance, and local name;*

2. *local-basis coverage, requiring every accepted point-window and open-classifier read to meet a displayed basic cell from B through the same source row X and topology row T;*

3. *basis refinement closure, requiring any two displayed basic cells accepted at a common point-window to refine through another displayed basic cell indexed by the same enumeration row B;*

4. *topology and metric classifier transport, requiring every open-set or metric-ball read to pass through T and, when present, M before it is exported to a consumer;*

5. *ledger nonescape, requiring every dyadic radius, stream-window, real-boundary, transport, replay, provenance, and local-naming read to remain inside the rows D, S, R, H, C, P, N.*

No second-countability obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SecondCountableUp.SecondCountableCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1901.1. The accepted packet displays exactly one source row, topology classifier, metric witness slot, base enumeration, dyadic ledger, stream-window row, real-seal row, and structural rows.

Local-basis coverage is read from X, T, B. A consumer opens a point or source window in X, reads the generated-open classifier in T, and receives only a displayed basic cell from B that is witnessed at the same finite window.

Basis refinement closure is also internal to B. When two displayed basic cells are accepted at a common point-window, the refinement row is another finite base-enumeration entry, not a host intersection or a choice-selected neighborhood.

Topology and metric classifier transport are forced by T and M. The topology row supplies the open-set classifier, while the metric witness row supplies the ball route when the presentation has metric data; a non-metric packet exposes only the explicit empty metric slot.

Finally D, S, R, H, C, P, N keep the read inside the finite ledger. Dyadic precision, stream windows, real boundary reads, componentwise transport, replay, provenance, and local naming are

all packet fields, so no open-cover compactness theorem, Lindelof theorem, host basis, countable-choice witness, or $\mathsf{Real}^{\uparrow}$ completeness field can be exported. □

Theorem 1901.3 (Second-countable metric-base handoff)**.** *Every metric or topology consumer read of an accepted* $\mathsf{SecondCountable}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before any open-set or completion-facing read is accepted. A metric-presented packet first reads the $\mathsf{Met}^{\uparrow}$ *ball witness and dyadic radius ledger, then the* $\mathsf{StreamName}^{\uparrow}$ *inspection window, then the* $\mathsf{Topology}^{\uparrow}$ *generated-open classifier, and finally any* $\mathsf{Real}^{\uparrow}$ *boundary seal needed by a completion-facing consumer. A non-metric packet uses the same topology, stream, dyadic, and real-boundary rows while keeping the metric witness row empty.*

Lean4 ✓ `BEDC.Derived.SecondCountableUp.SecondCountableCarrier metric base handoff`

Proof. Start with the packet $Q = (X, T, M, B, D, S, R, H, C, P, N)$. A topology consumer has no open-set coordinate except T, so every open read begins with the displayed source row X, the topology classifier T, and the base enumeration row B.

If the consumer asks for a metric-ball presentation, the row M is the only metric witness. Its radius and precision reads must pass through the dyadic ledger D before the basic-open cell in B is accepted.

The stream row S names the finite inspection windows for the enumeration and consumer probes. It prevents the read from becoming a host sequence of basic opens or a selected countable basis.

The real boundary row R is the only endpoint-facing seal. Completion-facing consumers therefore see the route through M, D, S, T, B, R before any real-valued or completion-facing read is exported.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the packet has no host-set equality coordinate, open-cover compactness witness, countable-choice field, or $\mathsf{Real}^{\uparrow}$ completeness field, the handoff stays inside the stated BEDC rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{SecondCountable}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SecondCountable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1901.1, Definition 1901.2, and Definition 1901.3 seed the finite countable-base interface over displayed source rows, topology classifiers, optional metric-ball witnesses, dyadic precision ledgers, stream windows, real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SecondCountable}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SecondCountableUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Lindelof theorems, selected bases as host sets, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-basis coverage, basis refinement, metric-base handoff, transport, replay, provenance, and local naming for BEDC.Derived.SecondCountableUp.

Constructive story: A $\mathsf{SecondCountable}^{\uparrow}$ packet is generated from one point or source carrier row, one Topology classifier row, an optional Metric ball witness row, one displayed countable-base enumeration row, one DyadicRatCore precision ledger, one StreamName finite-inspection row, one Real boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1902

A Concrete Naming Certificate for $\mathsf{Rolle}^{\uparrow}$

Following Rolle, Weierstrass, and Bishop–Bridges constructive analysis, the $\mathsf{Rolle}^{\uparrow}$ packet records the finite BEDC reading of Rolle's theorem for a continuous real-valued map on a compact interval whose endpoint values agree and whose interior derivative quotient admits a zero seal. It consumes the terminal real-name surface of chapter 38, the graph and pointwise-modulus rows of chapter 130, the compact interval locator of chapter 1891, the finite attainment route of chapter 1493, and the derivative metric quotient rows of chapter 122. The packet names a stationary-point certificate over displayed interval, endpoint, continuity, attainment, derivative, and zero-seal rows; it is not a host mean-value theorem, a classical differentiability principle, quotient real equality, open-cover compactness, or a choice-selected interior point.

Definition 1902.1 (Rolle carrier)**.** *A* $\mathsf{Rolle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, E, F, V, D, W, Z, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, exposing the located endpoint names, interval locator, finite windows, and compact-interval boundary data used by the theorem.*
2. *E is the endpoint-equality row, recording that the two endpoint value seals for the displayed map are classifier-equal through the same* $\mathsf{Real}^{\uparrow}$ *route.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the real-valued map on the compact interval source.*
4. *V is the* $\mathsf{ExtremeValue}^{\uparrow}$ *witness row, carrying the finite attainment route for the displayed continuous map on the compact interval data.*
5. *D is the* $\mathsf{Derivative}^{\uparrow}$ *metric quotient row, recording the inspected interior derivative quotient route rather than a host differentiability predicate.*
6. *W is the interior witness row, binding the stationary-point read to a displayed interior interval cell and excluding endpoint-only witnesses.*
7. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *zero-seal row for the derivative quotient named by D at the interior witness named by W.*
8. *H records componentwise* hsame *transport for the compact interval source, endpoint equality, continuous-map row, extreme-value witness, derivative quotient, interior witness, zero seal, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from interval inspection to map windows, from endpoint equality to the extreme-value comparison route, from the attainment row to the interior witness, and from the derivative quotient route to the zero seal.*

10. *P is the* Pkg *provenance over* $\mathsf{Rolle}^{\uparrow}$, $\mathsf{CompactIntervalLocator}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ExtremeValue}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Rolle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host real equality, open-cover compactness, arbitrary point choice, ambient differentiability, a full mean-value theorem, quotient real equality, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RolleUp.RolleUpTasteGate single carrier alignment`

Theorem 1902.2 (Rolle NameCert obligations)**.** *For every accepted* $\mathsf{Rolle}^{\uparrow}$ *carrier*

$$R = (I, E, F, V, D, W, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Rolle}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact interval source admission through I;*
2. *endpoint equality for the two displayed endpoint value seals through E;*
3. *continuous real-valued map admission through F;*
4. *extreme-value witness routing through V;*
5. *derivative metric quotient and interior zero-seal routing through D, W, Z.*

The structural rows H, C, P, N transport, replay, source, and name only these five rows. No Rolle obligation is stored outside the displayed carrier.

Proof. Carrier admission is projection from Definition 1902.1. The source row I supplies the compact interval data, and the endpoint row E supplies the equality of the two displayed endpoint value seals.

The map and attainment rows are then fixed. The row F is the only continuous-map surface, and V is the only extreme-value witness route attached to that map and interval source.

Finally inspect the derivative-facing rows. The quotient row D, the interior witness row W, and the zero-seal row Z form the only derivative-zero route in the packet.

The remaining rows H, C, P, N preserve, replay, source, and name the same data. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RolleUp.TasteGate.RolleTasteGate single carrier alignment`

Theorem 1902.3 (Rolle extreme-value stationary point)**.** *Every stationary witness exported by an accepted* $\mathsf{Rolle}^{\uparrow}$ *carrier factors through the extreme-value row V together with the endpoint-equality row E. The witness is therefore read as an interior stationary certificate produced from displayed attainment and equal-endpoint rows, not as a choice-selected point or an ambient compactness conclusion.*

Proof. Start with an accepted carrier $R = (I, E, F, V, D, W, Z, H, C, P, N)$. The compact interval source and continuous-map row expose the domain and graph data before any stationary read is available.

Read the endpoint equality E and the extreme-value row V. These are the only rows connecting the two endpoint value seals to the attained-value comparison route.

The interior witness W is downstream of that comparison. Transport and replay through H and C preserve the same row inventory, while P and N add only provenance and local naming.

Thus any exported stationary witness passes through the endpoint-equality and extreme-value rows before the derivative-zero seal can be displayed. The packet contains no separate selected point or host compactness conclusion. □

Theorem 1902.4 (Rolle derivative-zero seal boundary)**.** *The derivative-zero claim exported by an accepted* $\mathsf{Rolle}^{\uparrow}$ *carrier is only the displayed* $\mathsf{Real}^{\uparrow}$ *zero seal* Z *over the derivative quotient route* D *at the interior witness* W*. It does not assert host differentiability, a general mean-value theorem, quotient real equality, or an ambient equality of completed real functions.*

Proof. Project the derivative-facing part of Definition 1902.1. The packet contains a derivative metric quotient row D, an interior witness row W, and a terminal zero-seal row Z.

The derivative chapter supplies the quotient route, while the real chapter supplies the meaning of the displayed zero seal. Neither row stores a host differentiability predicate or an equality of completed functions.

The endpoint and extreme-value rows explain why this zero seal is exported for the Rolle packet, but they do not enlarge it into a mean-value theorem. The structural rows only transport and replay the displayed route.

Consequently the exported derivative-zero statement remains a $\mathsf{Real}^{\uparrow}$ seal over D and W, with no hidden host differentiability or ambient real-equality content. □

Falsifiable prediction: Every $\mathsf{Rolle}^{\uparrow}$ stationary read factors through a compact interval source row, endpoint equality row, $\mathsf{ContinuousMap}^{\uparrow}$ row, $\mathsf{ExtremeValue}^{\uparrow}$ witness row, $\mathsf{Derivative}^{\uparrow}$ metric quotient row, interior witness row, terminal $\mathsf{Real}^{\uparrow}$ zero seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, continuousmap, compact-interval-locator, extremevalue, derivative: $\mathsf{Rolle}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries equal endpoint value seals, a compact interval source, a continuous real-valued map, an extreme-value witness, an interior derivative quotient, a zero seal, transport, replay, provenance, and local naming as one stationary-point packet.

Closure status for $\mathsf{NameCert}_{\mathsf{Rolle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Rolle}^{\uparrow}$ packet is generated from a compact interval source row, an endpoint-equality row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph row, an $\mathsf{ExtremeValue}^{\uparrow}$ witness row, a $\mathsf{Derivative}^{\uparrow}$ metric quotient row, an interior witness row, a terminal $\mathsf{Real}^{\uparrow}$ zero seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Rolle}^{\uparrow}}$, seed)

Scope closed: Definition 1902.1, Definition 1902.2, Definition 1902.3, and Definition 1902.4 seed the finite Rolle stationary-point reading over compact interval, endpoint equality, continuous-map, extreme-value, derivative quotient, interior witness, and Real zero-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Rolle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host differentiability, a general mean-value theorem, quotient real equality, open-cover compactness, selected interior points, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint equality, extreme-value factorization, derivative quotient zero-seal boundary, and TasteGate field faithfulness for BEDC.Derived.RolleUp.

Chapter 1903

A Concrete Naming Certificate for $\mathsf{CompactOpenMetric}^{\uparrow}$

Following Kelley and Bourbaki on the compact-open topology, and Bishop-style finite nets for compact metric spaces, the $\mathsf{CompactOpenMetric}^{\uparrow}$ packet names the constructive compact-open interface seen by continuous-map families. It records a sup-distance boundary over finite source nets rather than an ambient topology on all functions. The packet sits between the compact source rows of chapter 129, the continuous-map graph and modulus rows of chapter 130, the complete target rows of chapter 128, the finite handoff of chapter 388, the modulus ledger of chapter 369, and the Real-distance surface of chapter 1624.

The chapter isolates the function-family metric boundary consumed by compact-continuity arguments such as chapter 412. It does not construct an open-cover compactness theorem, does not install a host function-space topology, and does not choose subsequences or limits outside the displayed finite rows.

Definition 1903.1 (Compact-open metric carrier)**.** *A* $\mathsf{CompactOpenMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, G, M, D, S, H, C, P, N)$$

with the following displayed coordinates:

1. *X is the source* $\mathsf{CompactMetric}^{\uparrow}$ *row, including the compact metric distance surface and finite* $\mathsf{ProbeBundle}$ *net rows used for source inspection.*
2. *Y is the target* $\mathsf{CompleteMetric}^{\uparrow}$ *row whose distance witnesses and completion-facing reads receive every sampled value.*
3. *F is a finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows from X to Y.*
4. *G records the displayed graph rows of the maps in F on the finite source-net windows.*
5. *M records* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$ *handoff rows turning pointwise continuity data into uniform finite source windows.*
6. *D records finite target-distance samples for pairs of rows in F over the source-net windows, read through the target metric and the* $\mathsf{RealUniformStructure}^{\uparrow}$ *distance boundary.*
7. *S records the sup-distance comparison ledger obtained by taking the finite maximum of the sampled target-distance rows at each requested window.*

8. *H records componentwise* hsame *transport for source compact-metric rows, target complete-metric rows, continuous-map rows, graph rows, modulus rows, distance samples, sup-distance ledgers, replay rows, provenance, and local naming.*

9. *C records displayed* Cont *replay from a compact-open distance request through source-net selection, graph evaluation, target-distance sampling, finite maximum comparison, and modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactOpenMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, F, G, M, D, S, H, C, P, N$. No carrier coordinate names open-cover compactness, an ambient function-space topology, choice-selected subsequences, or Real completeness.

Lean4 ✓ `BEDC.Derived.CompactOpenMetricUp.CompactOpenMetricTasteGate single carrier alignment`

Theorem 1903.2 (Compact-open metric NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenMetric}^{\uparrow}$ *carrier $K = (X, Y, F, G, M, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenMetric}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the source compact-metric row, target complete-metric row, continuous-map family ledger, displayed graph rows, finite uniform-modulus rows, target-distance samples, finite sup-distance comparison rows, transport, replay, provenance, and local naming;*

2. *finite source inspection, requiring every compact-open distance request to choose its source windows from the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coordinate X;*

3. *graph and distance sampling, requiring every sampled value to be read through G and every sampled distance to be read through Y and the* $\mathsf{RealUniformStructure}^{\uparrow}$ *distance boundary;*

4. *uniform handoff, requiring M to record the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$ *route that makes finite source-net samples stable for the requested tolerance;*

5. *classifier transport and non-escape, requiring H, C, P, N to preserve the same rows and to export no open-cover compactness, ambient function-space topology, choice-selected subsequence, or Real completeness.*

No NameCert obligation is stored outside those displayed rows.

Proof. Carrier admission is projection from Definition 1903.1. The packet exposes exactly the source compact-metric row, target complete-metric row, family ledger, graph rows, modulus rows, sampled-distance rows, finite sup-distance ledger, and structural rows.

Finite source inspection is forced by the coordinate X. A compact-open distance request first reads the compact-metric finite-net rows and cannot inspect source points through an unrecorded open cover.

Graph and distance sampling are then read from G, Y, and D. The family ledger supplies the named continuous-map graph rows, the target row supplies complete-metric distance witnesses, and the Real uniform-structure boundary names the distance surface used by the finite comparison.

The modulus handoff is the coordinate M. It records the compact-uniform-continuity and uniform-modulus rows that make the finite samples stable under the requested tolerance, so the sup-distance ledger is a finite certificate rather than a topology on all functions.

Finally, H, C, P, N transport and replay the same route. Because the carrier has no row for open-cover compactness, an ambient function-space topology, choice-selected subsequences, or Real completeness, the local certificate cannot export any of them. □

Theorem 1903.3 (Compact-open metric uniform-distance handoff)**.** *Every compact-continuity consumer reading an accepted* $\mathsf{CompactOpenMetric}^{\uparrow}$ *carrier obtains a finite uniform-distance handoff through the displayed rows*

$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{CompactUniformContinuity}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}, \mathsf{RealUniformStru}$

The handoff supplies source finite nets, continuous-map graph samples, target distance rows, finite sup-distance comparisons, and modulus stability. It supplies no open-cover compactness, ambient function-space topology, choice-selected subsequence, or Real completeness.

Proof. Begin with a compact-continuity consumer and project the accepted packet K. The source coordinate X supplies the finite ProbeBundle windows through the compact-metric row, and the family coordinate F names the continuous-map rows that may be sampled.

Next read the graph coordinate G and target coordinate Y. Each sampled value is a displayed graph read, and each target-distance comparison is a displayed complete-metric read whose Real-distance surface is recorded in D.

The finite comparison ledger S gathers those target-distance samples into the sup-distance boundary for the requested window. The modulus coordinate M records the compact-uniform-continuity and uniform-modulus route that makes this finite comparison stable for downstream tolerance requests.

The structural rows H, C, P, N preserve the same finite route under classifier transport and replay. Since no further coordinate is present, the handoff cannot add open-cover compactness, host function-space topology, choice-selected subsequences, or Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenMetric}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a CompleteMetricUp target row, ContinuousMapUp family and graph rows, CompactUniformContinuityUp and UniformModulusUp handoff rows, RealUniformStructureUp target-distance rows, finite sup-distance comparison rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1903.1, Definition 1903.2, and Definition 1903.3 seed the constructive compact-open sup-metric boundary for continuous-map families.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenMetric}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactOpenMetricUp.CompactOpenMetricTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient function-space topology, choice-selected subsequences, Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires carrier admission, finite source inspection, graph and distance sampling, uniform handoff, classifier transport, and non-escape proofs for the same packet.

Chapter 1904

A Concrete Naming Certificate for $\mathsf{RegularCauchyCancellation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyCancellation}^{\uparrow}$ packet records the finite cancellation boundary for displayed regular rational Cauchy names. It ties the pointwise sum route of chapter 417, the difference route of chapter 429, the zero route of chapter 624, the finite precision ledger of chapter 385, the regular source rows of chapter 301, and the terminal real-name seal of chapter 38 into one inspectable packet. The packet supplies a bounded route for cancelling a shared displayed summand before any $\mathsf{Real}^{\uparrow}$ consumer reads the result; it is not a completed-real additive group law, quotient stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1904.1 (Regular Cauchy cancellation carrier)**.** *A* $\mathsf{RegularCauchyCancellation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, S_X, S_Y, D, B, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. X, Y, Z *are the three* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, with* Z *the shared summand offered to both sum routes.*
2. S_X *is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *route for the displayed sum* $X + Z$*.*
3. S_Y *is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *route for the displayed sum* $Y + Z$*.*
4. D *is the* $\mathsf{RegularCauchyDifference}^{\uparrow}$ *route comparing the two displayed sums and exposing the difference readback only through finite source windows, dyadic endpoint rows, and regular readback data.*
5. B *is the* $\mathsf{FiniteErrorBudget}^{\uparrow}$ *ledger that accounts for the two sum budgets, the difference budget, and the cancellation tolerance requested by a downstream consumer.*
6. Q *is the* $\mathsf{RegularCauchyZero}^{\uparrow}$ *boundary row used to identify the shared* $Z - Z$ *contribution as the displayed zero route, with any* $\mathsf{RegularCauchyNegation}^{\uparrow}$ *read confined to the sibling difference packet.*
7. R *is the resulting* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite cancellation route from* $(X+Z)-(Y+Z)$ *to* $X - Y$*.*
8. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S_X, S_Y, D, B, Q, R *have been displayed.*

9. *H records componentwise* hsame *transport for the source rows, two sum routes, difference route, budget ledger, zero boundary, readback, seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a cancellation-facing precision request through $X, Y, Z, S_X, S_Y, D, B, Q, R,$*
11. *P is the* Pkg *provenance over* RegularCauchyCancellation$^\uparrow$, RegularCauchySum$^\uparrow$, RegularCauchyDifference$^\uparrow$, RegularCauchyZero$^\uparrow$, RegularCauchyNegation$^\uparrow$, FiniteErrorBudget$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{RegularCauchyCancellation}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, Y, Z, S_X, S_Y, D, B, Q, R, E, H, C, P, N$. The carrier has no coordinate for completed-real additive group laws, quotient stream equality, selected limits, trichotomy, countable choice, ambient Real$^\uparrow$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCancellationUp.TasteGate.RegularCauchyCancellationTasteGate`
`single carrier alignment`

Theorem 1904.2 (Regular Cauchy cancellation NameCert obligation surface)**.** *For every accepted* RegularCauchyCancellation$^\uparrow$ *carrier*

$$K = (X, Y, Z, S_X, S_Y, D, B, Q, R, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyCancellation}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the three* RegSeqRat$^\uparrow$ *source rows, two* RegularCauchySum$^\uparrow$ *routes, the* RegularCauchyDifference$^\uparrow$ *comparison route, the* FiniteErrorBudget$^\uparrow$ *ledger, the zero boundary row, the readback row, and the structural rows;*
2. *sum exactness, requiring S_X and S_Y to expose the two displayed sums only through aligned* StreamName$^\uparrow$ *windows,* DyadicRatCore$^\uparrow$ *endpoint rows, and their own finite error ledgers;*
3. *difference exactness, requiring D to compare the displayed sums through the sibling* RegularCauchyDifference$^\uparrow$ *route before any cancellation readback is available;*
4. *zero-boundary exactness, requiring Q to display the shared $Z-Z$ contribution through* RegularCauchyZero$^\uparrow$ *rather than through host real algebra;*
5. *budget exactness, requiring B to account for the two sum budgets, the difference budget, and the cancellation tolerance before R is consumed;*
6. *Real seal nonescape, requiring E to be downstream of R and to expose no completed-real additive group law, quotient stream equality, selected limit, trichotomy, countable choice, ambient completeness, Lean verification, or bridge row;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the same finite cancellation packet.*

No cancellation obligation is stored outside these rows.

Theorem 1904.3 (Regular Cauchy left cancellation finite route)**.** *Let $K = (X, Y, Z, S_X, S_Y, D, B, Q, R, E, H, C, P,$ be an accepted* RegularCauchyCancellation$^\uparrow$ *carrier. Every consumer that reads the cancellation of the shared left-hand summand factors through the finite route*

$$X, Y, Z \longmapsto S_X, S_Y \longmapsto D, B, Q \longmapsto R, E, H, C, P, N.$$

The consumer first reads the three source RegSeqRat$^{\uparrow}$ *rows and the two displayed sum routes, then the difference route between those sums, the finite budget ledger, and the zero boundary for the shared $Z - Z$ contribution, and only then obtains the regular readback and* Real$^{\uparrow}$ *seal. The route exports no completed-real additive group law, quotient stream equality, selected limit, trichotomy witness, countable-choice selector, or ambient completeness principle.*

Theorem 1904.4 (Regular Cauchy cancellation Real seal nonescape)**.** *Every* Real$^{\uparrow}$ *consumer of an accepted* RegularCauchyCancellation$^{\uparrow}$ *carrier reaches the seal row E only after the* RegSeqRat$^{\uparrow}$ *readback R has been assembled from the two* RegularCauchySum$^{\uparrow}$ *routes, the* RegularCauchyDifference$^{\uparrow}$ *route, the* RegularCauchyZero$^{\uparrow}$ *boundary row, and the* FiniteErrorBudget$^{\uparrow}$ *ledger. The real-facing boundary exposes the finite source rows, sum routes, difference route, zero route, budget ledger, readback, componentwise transport, continuation replay, provenance, and local naming. It exposes no completed-real additive group law, quotient stream equality, selected limit, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or public bridge row.*

Closure status for NameCert$_{\text{RegularCauchyCancellation}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCancellation}^{\uparrow}}$, seed)

Scope closed: Definition 1904.1, Definition 1904.2, Definition 1904.3, and Definition 1904.4 seed the finite cancellation boundary through three RegSeqRat$^{\uparrow}$ source rows, two sum routes, one difference route, one zero boundary row, one finite budget ledger, a regular readback, and a Real$^{\uparrow}$ seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCancellation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed-real additive group laws, quotient stream equality, selected limits, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sum exactness, difference exactness, zero-boundary exactness, budget exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCancellationUp.

Constructive story: A RegularCauchyCancellation$^{\uparrow}$ packet is generated from three RegSeqRat$^{\uparrow}$ source rows, two RegularCauchySum$^{\uparrow}$ routes, one RegularCauchyDifference$^{\uparrow}$ route, one RegularCauchyZero$^{\uparrow}$ boundary row, one FiniteErrorBudget$^{\uparrow}$ ledger, a RegSeqRat$^{\uparrow}$ readback, a Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1905

A Concrete Naming Certificate for $\mathsf{RealApartnessOrderBridge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ packet records the constructive route from positive real apartness evidence to order-facing comparison data. It consumes chapter 304, **??**, chapter 38, chapter 301, chapter 291, and chapter 126. The packet names the finite handoff by which a displayed positive radius can be read by order consumers without installing trichotomy, quotient real equality, a selected limit, or completeness for $\mathsf{Real}^{\uparrow}$.

Definition 1905.1 (RealApartnessOrderBridge carrier)**.** *A* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, A, L, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two endpoint* $\mathsf{Real}^{\uparrow}$ *rows, exposing only their terminal real-name seals and the finite comparison handles needed by downstream order consumers.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-radius witness row, binding the endpoint pair to an inspected radius seal and recording that the separation evidence is positive apartness data rather than an equality decision.*
3. *L records the* $\mathsf{LocatedReal}^{\uparrow}$ *refinement rows for the two endpoints, including the located dyadic refinements used to align the endpoint windows with the positive-radius witness.*
4. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for the rational regular sequences underlying the displayed endpoint names and refinement windows.*
5. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and comparison rows, including the positive radius token, dyadic lower-bound comparison, and finite order-facing comparison cell exported to consumers.*
6. *S records the* $\mathsf{StreamName}^{\uparrow}$ *finite window rows used to inspect the endpoint streams, radius stream, and comparison windows at the displayed precision.*
7. *H records componentwise* hsame *transport for the endpoint rows, apartness witness, located refinements, regular-sequence sources, dyadic-radius and comparison rows, stream windows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from endpoint names to stream windows, from stream windows to located dyadic refinements, from the positive-radius witness to the dyadic comparison cell, and from the comparison cell to the order-facing handoff row.*

9. *P is the* Pkg *provenance over* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessOrderBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for equality-or-apartness decisions, total order, quotient real equality, selected real limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealApartnessOrderBridgeUp.RealApartnessOrderBridgeTasteGate single carrier alignment`

Theorem 1905.2 (RealApartnessOrderBridge NameCert obligations)**.** *For every accepted* RealApartnessOrderBrid *carrier*

$$G = (X, Y, A, L, R, D, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealApartnessOrderBridge}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *endpoint admission for the two displayed* $\mathsf{Real}^{\uparrow}$ *rows X, Y;*
2. *positive apartness witness admission through the* $\mathsf{ApartnessReal}^{\uparrow}$ *row A;*
3. *located dyadic refinement admission through the* $\mathsf{LocatedReal}^{\uparrow}$ *row L;*
4. *rational and dyadic comparison source admission through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows R, D;*
5. *finite stream-window admission through the* $\mathsf{StreamName}^{\uparrow}$ *row S.*

The structural rows H, C, P, N transport, replay, source, and name only these five obligation rows. No order-facing handoff obligation is stored outside the displayed carrier.

Proof. Carrier admission is projection from Definition 1905.1. The endpoint rows X, Y supply the two displayed real-name surfaces, and the apartness row A supplies the positive-radius witness attached to that endpoint pair.

The refinement and source rows are then fixed. The row L is the only located-refinement surface, while R and D are the only rational and dyadic rows that expose the radius and comparison data.

Finally inspect the finite-window row S. It is the only stream-name surface from which the endpoint, radius, and comparison windows are read.

The remaining rows H, C, P, N preserve, replay, source, and name the same data. Since the carrier has no additional coordinate, the listed five rows exhaust the local NameCert surface. □

Theorem 1905.3 (RealApartnessOrderBridge positive order handoff)**.** *Every order-facing consumer read exported by an accepted* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *carrier factors through the displayed positive-separation handoff formed by the* $\mathsf{ApartnessReal}^{\uparrow}$ *witness row A, the* $\mathsf{LocatedReal}^{\uparrow}$ *refinement row L, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row D, and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S. The consumer may read only this finite positive-separation handoff, not an ambient comparison principle.*

Proof. Start with an accepted carrier $G = (X, Y, A, L, R, D, S, H, C, P, N)$. The endpoint rows expose the two real names before any order-facing read is available.

Read the positive-radius witness A together with the located refinements L. These rows bind the apartness evidence to displayed dyadic windows rather than to a host comparison of completed reals.

The dyadic comparison row D and stream-window row S then provide the finite comparison cell consumed by an order-facing reader. Transport and replay through H and C preserve that same row inventory, while P and N add only provenance and local naming.

Thus every exported consumer read factors through the displayed positive-separation handoff. The packet contains no separate ambient comparison principle or selected real object. □

Theorem 1905.4 (RealApartnessOrderBridge no trichotomy)**.** *An accepted* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *carrier exports no equality-or-apartness decision, total order on* $\mathsf{Real}^{\uparrow}$*, quotient real, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness principle. Its public content is only the finite positive-separation handoff described by the displayed endpoint, apartness, located-refinement, rational-source, dyadic-comparison, stream-window, transport, replay, provenance, and naming rows.*

Proof. Project the rows of Definition 1905.1. The endpoint rows, positive-radius witness, located refinements, rational sources, dyadic comparison rows, and stream windows are the only mathematical data coordinates in the carrier.

None of these rows stores a trichotomy branch, quotient constructor, equality decision, selected limit, or completeness witness. The apartness row records positive evidence, and the comparison row records only the finite handoff cell derived from that evidence.

The structural rows H, C, P, N cannot create new mathematical coordinates; they only transport, replay, source, and name the displayed rows. Therefore the packet exports the stated positive-separation handoff and none of the listed excluded principles. □

Falsifiable prediction: Every $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ order-facing read factors through two endpoint $\mathsf{Real}^{\uparrow}$ rows, an $\mathsf{ApartnessReal}^{\uparrow}$ positive-radius witness, $\mathsf{LocatedReal}^{\uparrow}$ refinement rows, $\mathsf{RegSeqRat}^{\uparrow}$ source rows, $\mathsf{DyadicRatCore}^{\uparrow}$ radius and comparison rows, $\mathsf{StreamName}^{\uparrow}$ finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: apartnessreal, locatedreal, real, regseqrat, dyadicratcore, streamname: $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint names, a positive apartness witness, located refinements, rational sequence sources, dyadic radius and comparison data, finite stream windows, transport, replay, provenance, and local naming as one order-facing handoff packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessOrderBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ rows, an $\mathsf{ApartnessReal}^{\uparrow}$ positive-radius witness, $\mathsf{LocatedReal}^{\uparrow}$ refinement rows, $\mathsf{RegSeqRat}^{\uparrow}$ source rows, $\mathsf{DyadicRatCore}^{\uparrow}$ radius and comparison rows, $\mathsf{StreamName}^{\uparrow}$ finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApartnessOrderBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1905.1, Definition 1905.2, Definition 1905.3, and Definition 1905.4 seed the finite positive-apartness route from endpoint real names to order-facing comparison data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApartnessOrderBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close equality-or-apartness decisions, total order on RealUp, quotient real construction, selected limits, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius handoff, no-trichotomy boundary, and TasteGate field faithfulness for BEDC.Derived.RealApartnessOrderBridgeUp.

Chapter 1906

A Concrete Naming Certificate for $\mathsf{RegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegulatedFunction}^{\uparrow}$ packet names a regulated function by finite BHist rows of dyadic step-function approximation schedules. It sits over the dyadic mesh surface of chapter 358, the dyadic step-function carrier of chapter 428, the finite Darboux sum boundary of chapter 1880, the tagged-mesh Riemann route of chapter 1526, and the terminal real-name seal of chapter 38. The packet gives integration-facing consumers a displayed step-approximation handoff without importing host function quotients, measure-theoretic integration, open-cover compactness, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1906.1 (Regulated function carrier)**.** *A* $\mathsf{RegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, M, A, T, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain row, read through the terminal* $\mathsf{Real}^{\uparrow}$ *seal and its interval-facing data from chapter 38.*
2. *M is the finite dyadic mesh ledger, read through chapter 358, listing the displayed partitions used by the approximation schedule.*
3. *A is the dyadic step-function approximation schedule. Each row of A is a* $\mathsf{DyadicStepFunction}^{\uparrow}$ *packet over one mesh row of M, with its cells and values read through chapter 428.*
4. *T is the tolerance and tail ledger recording, for every displayed dyadic tolerance request, which later step rows are close enough to represent the same regulated read.*
5. *S is the finite sum handoff row connecting the step schedule to the* $\mathsf{DarbouxSum}^{\uparrow}$ *boundary of chapter 1880 and the* $\mathsf{RiemannIntegral}^{\uparrow}$ *tagged-mesh route of chapter 1526.*
6. *H records componentwise* hsame *transport for domain, mesh, step schedule, tolerance tail, finite sum handoff, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a domain request to a mesh row, from that mesh to a step-function approximant, from the approximant to the tail ledger, and from the tail ledger to the finite sum handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegulatedFunction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, M, A, T, S, H, C, P, N$. It has no coordinate for host function quotients, measure-theoretic integration, open-cover compactness, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedFunctionUp.RegulatedFunctionTasteGate single carrier alignment`

Theorem 1906.2 (Regulated function NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *carrier $F = (D, M, A, T, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, real-domain admission, dyadic mesh admission, dyadic step-function schedule admission, tolerance-tail routing, Darboux/Riemann finite-sum handoff, componentwise transport, continuation replay, provenance, local naming, and non-escape from host function quotients, measure-theoretic integration, open-cover compactness, selected limits, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1906.1. The accepted packet displays only the domain row D, mesh ledger M, step schedule A, tail ledger T, finite sum handoff S, transports H, continuations C, provenance P, and local name N.

The analytic rows are all finite. The real-facing domain row is read through chapter 38; each mesh row is read through chapter 358; and each approximant in A is read through the dyadic step-function carrier of chapter 428.

The tail and handoff rows impose the regulated-function route. The ledger T records which displayed approximants answer each tolerance request, while S exposes only the finite Darboux-sum and Riemann-mesh handoff named by chapter 1880 and chapter 1526. The rows H, C, P, N preserve and name this same route, so no excluded host object is available. □

Theorem 1906.3 (Regulated function step-approximation handoff)**.** *Every accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *packet exposes any integration-facing read through the finite route*

$$D, M, A, T \longmapsto S.$$

The row S can be consumed by Darboux and Riemann chapters only after the real-domain row, dyadic mesh ledger, dyadic step-function schedule, and tolerance-tail ledger have all been displayed.

Proof. Start with an integration-facing consumer request. The continuation row C first replays the real-domain row D and then selects a finite dyadic mesh row from M, so the request is already confined to displayed real and mesh data.

The chosen mesh row determines the visible step approximant in A. That approximant is a dyadic step-function packet, hence its cells, endpoint transports, and carried values are finite rows before any sum handoff is read.

It remains to reach S. The tail ledger T records that the selected approximant belongs to the displayed regulated schedule at the requested tolerance. Only then does C expose the finite sum handoff row S, whose consumers are the Darboux sum and Riemann integral surfaces. No carrier coordinate offers a host quotient of functions, a measure-theoretic integral, an open-cover compactness proof, a selected limit, or an ambient completeness witness for $\mathsf{Real}^{\uparrow}$. □

Theorem 1906.4 (Regulated function bounded step envelope)**.** *Every accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *packet exposes any boundedness-facing read only through*

$$D, M, A, T$$

before the finite-sum handoff row S can be consumed. The finite dyadic step approximants in A, indexed by the mesh ledger M and tolerance-tail ledger T, supply the only visible envelope rows available to downstream boundedness consumers.

Proof. Project the accepted carrier $F = (D, M, A, T, S, H, C, P, N)$ from Definition 1906.1. The rows preceding S are precisely the real-domain row, dyadic mesh ledger, dyadic step-function schedule, and tolerance-tail ledger.

Apply the step-approximation handoff of Definition 1906.3. Any consumer that reaches S has already read D, M, A, T, and the approximants in A are finite $\mathsf{DyadicStepFunction}^{\uparrow}$ packets over the displayed mesh rows.

Thus a boundedness-facing consumer can request envelope data only from the visible step schedule and its tolerance ledger. The carrier provides no host function quotient, selected supremum, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate from which another bound could be read. □

Theorem 1906.5 (Regulated function variation-control route)**.** *An accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *packet whose finite step schedule exposes compatible endpoint variation rows can be consumed by* $\mathsf{BoundedVariation}^{\uparrow}$ *and* $\mathsf{VariationMeasure}^{\uparrow}$ *only through the displayed route*

$$D, M, A, T, S, H, C, P, N$$

of Definition 1906.1, together with the sibling finite partition-variation ledger read by chapter 398 and chapter 3593. The route exports regulated step envelopes and finite partition variation data, not host total variation, measure completion, selected suprema, function quotients, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 1906.1. Its only analytic rows are the real-domain row D, finite dyadic mesh ledger M, dyadic step schedule A, tolerance-tail ledger T, and finite-sum handoff row S; the rows H, C, P, N only transport, replay, source, and name that same finite route.

Apply the step-approximation handoff of Definition 1906.3. Any consumer of the schedule first reads D, M, A, T before S is available, so compatible endpoint variation data can be attached only to the displayed finite step rows.

Now use the bounded step envelope of Definition 1906.4. It confines boundedness-facing reads to the visible step schedule and tail ledger, which is exactly the finite partition-variation surface consumed by $\mathsf{BoundedVariation}^{\uparrow}$ and then by $\mathsf{VariationMeasure}^{\uparrow}$.

Thus the cross-chapter handoff is the displayed regulated-function route plus the sibling finite partition ledger. No carrier coordinate supplies host total variation, measure completion, selected suprema, function quotients, or ambient real completeness. □

Theorem 1906.6 (Regulated function obligation closure package)**.** *The carrier of Definition 1906.1, the NameCert surface of Definition 1906.2, and the step-approximation handoff of Definition 1906.3 jointly supply the finite obligation surface for* $\mathsf{RegulatedFunction}^{\uparrow}$*. This surface consists exactly of carrier admission,* $\mathsf{Real}^{\uparrow}$ *domain admission,* $\mathsf{DyadicMesh}^{\uparrow}$ *admission,* $\mathsf{DyadicStepFunction}^{\uparrow}$ *schedule admission, tolerance-tail routing, Darboux/Riemann finite-sum handoff, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}$ *admission, and non-escape from host function quotients, measure-theoretic integration, open-cover compactness, selected limits, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $F = (D, M, A, T, S, H, C, P, N)$. The carrier definition displays exactly the domain row, mesh ledger, step schedule, tail ledger, handoff row, transport row, continuation row, provenance row, and local naming row.

The NameCert-obligation theorem identifies these displayed rows as the full local obligation surface. In particular, D carries real-domain admission, M carries dyadic mesh admission, A carries dyadic step-function schedule admission, T carries tolerance-tail routing, and S carries the Darboux/Riemann finite-sum handoff.

The handoff theorem then supplies the integration-facing direction. It forces every consumer of S to pass through D, M, A, T first. Rows H, C, P, N preserve, replay, source, and name the same route, so the excluded host structures never become available as obligations. □

Theorem 1906.7 (Regulated function scoped kernel route)**.** *Every accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer read factors through the displayed route*

$$D, M, A, T, S, H, C, P, N$$

over $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicMesh}^{\uparrow}$*,* $\mathsf{DyadicStepFunction}^{\uparrow}$*,* $\mathsf{DarbouxSum}^{\uparrow}$*,* $\mathsf{RiemannIntegral}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No accepted read factors through host function quotients, measure-theoretic integration, open-cover compactness, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier projection. The only analytic rows in an accepted packet are the real-domain row D, the dyadic mesh ledger M, the dyadic step-function schedule A, the tolerance-tail ledger T, and the finite-sum handoff row S.

The obligation closure package binds those analytic rows to the structural rows H, C, P, N. Thus componentwise transport, continuation replay, provenance, and local naming do not create a second carrier surface; they route the same finite data through the BEDC kernel objects listed in the statement.

Finally apply the step-approximation handoff. Any integration-facing consumer reaches S only after reading D, M, A, T, and any structural consumer reaches those rows only through H, C, P, N. This exhausts the scoped kernel route and leaves no coordinate for the excluded host objects. □

Theorem 1906.8 (Regulated function left–right limit consumer)**.** *Every accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer that reads paired one-sided limits must keep the source route*

$$D, M, A, T, S, H, C, P, N$$

visible before any $\mathsf{OneSidedLimit}^{\uparrow}$ *rows or terminal* $\mathsf{Real}^{\uparrow}$ *seals are consumed. Consequently a* $\mathsf{CadlagRealFunction}^{\uparrow}$ *or side-window consumer may use the regulated source as the displayed step-approximation schedule of chapter 5841 and chapter 5968, but it cannot replace that source with host function quotients, selected limits, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1906.1. The only source rows are the real-domain row D, dyadic mesh ledger M, dyadic step schedule A, tolerance-tail ledger T, finite-sum handoff S, and structural rows H, C, P, N.

Apply Definition 1906.7. It keeps every accepted consumer read inside those rows and refuses host function quotients, measure-theoretic integration, open-cover compactness, selected limits, and ambient Real completeness.

The one-sided-limit consumer can then read left and right side-window rows only downstream from that regulated source route. The cadlag chapter consumes the same paired side-window data after the regulated source has been displayed, so the paired limit handoff remains attached to $D, M, A, T, S, H, C, P, N$ rather than to an external function object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedFunction}^{\uparrow}$ packet is generated from a real-domain row, a finite dyadic mesh ledger, a dyadic step-function approximation schedule, a tolerance-tail ledger, a Darboux/Riemann finite-sum handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedFunction}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1906.1, Definition 1906.2, Definition 1906.3, Definition 1906.6, and Definition 1906.7 close the finite regulated-function surface over real-domain, dyadic mesh, dyadic step-function, tail-ledger, finite-sum handoff, transport, continuation, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function quotients, measure-theoretic integration, open-cover compactness, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires checked carrier admission, mesh and step-schedule admission, tolerance-tail routing, Darboux/Riemann handoff, non-escape, and field faithfulness for BEDC.Derived.RegulatedFunctionUp.

Chapter 1907

A Concrete Naming Certificate for $\mathsf{ConstructiveMeanValue}^{\uparrow}$

Following Cauchy, Lagrange, Rolle, and the Bishop–Bridges treatment of constructive analysis, the $\mathsf{ConstructiveMeanValue}^{\uparrow}$ packet records the finite BEDC boundary of the mean-value witness that is obtained by reducing a displayed secant comparison to the already named Rolle packet. It consumes the terminal real-name surface of chapter 38, the pointwise graph and modulus rows of chapter 130, the compact interval locator of chapter 1891, the finite attainment route of chapter 1493, the derivative metric quotient rows of chapter 122, and the stationary packet of chapter 1902. The chapter names only the finite witness surface: interval data, secant-slope rows, adjusted Rolle data, derivative quotient inspection, an interior witness, a terminal $\mathsf{Real}^{\uparrow}$ equality seal, transport, replay, provenance, and local naming.

Definition 1907.1 (Constructive mean-value carrier)**.** *A* $\mathsf{ConstructiveMeanValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, R, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the compact interval source row, exposing located endpoint names, interval cells, and the finite boundary data inspected by the witness route.*
2. *S is the secant-slope row, recording the displayed endpoint-value difference, interval-width seal, and quotient route as* $\mathsf{Real}^{\uparrow}$ *data rather than as a host real quotient.*
3. *R is the adjusted* $\mathsf{Rolle}^{\uparrow}$ *packet obtained by subtracting the displayed secant line from the named* $\mathsf{ContinuousMap}^{\uparrow}$ *row over the same compact interval source.*
4. *D is the* $\mathsf{Derivative}^{\uparrow}$ *metric quotient row for the adjusted map together with the derivative quotient route of the original displayed map.*
5. *W is the interior witness row exported by the Rolle packet R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal saying that the derivative quotient read at W is classifier-equal to the secant-slope row S.*
7. *H records componentwise* hsame *transport for the interval source, secant slope, adjusted Rolle packet, derivative quotient, interior witness, equality seal, replay rows, provenance rows, and local naming rows.*

8. *C records displayed* Cont *replay from interval inspection to endpoint reads, from the secant row to the adjusted Rolle packet, from the Rolle stationary row to derivative inspection, and from the derivative quotient to the terminal equality seal.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveMeanValue}^{\uparrow}$, $\mathsf{Rolle}^{\uparrow}$, $\mathsf{CompactIntervalLocator}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ExtremeValue}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveMeanValue}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host differentiability, a choice-selected point, quotient real equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.ConstructiveMeanValueUp.ConstructiveMeanValueTasteGate single carrier alignment`

Theorem 1907.2 (Constructive mean-value NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveMeanValue}^{\uparrow}$ *carrier*

$$M = (I, S, R, D, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveMeanValue}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact interval admission through I;*
2. *secant-slope construction through S over displayed* $\mathsf{Real}^{\uparrow}$ *endpoint and interval-width seals;*
3. *adjusted Rolle admission through R over the same interval and continuous-map source;*
4. *derivative quotient routing through D and the interior witness row W;*
5. *terminal equality sealing through E;*
6. *structural stability through H, C, P, N.*

No mean-value obligation is stored outside the displayed carrier.

Proof. Carrier admission is projection from Definition 1907.1. The row I supplies the compact interval source, and S supplies the only secant-slope data available to the packet.

The adjusted theorem route is then fixed. The row R is the sole $\mathsf{Rolle}^{\uparrow}$ packet in the carrier, and it is built over the same interval and continuous-map source rather than over a detached host function.

Inspect the derivative-facing rows. The quotient row D, the interior witness row W, and the terminal equality row E are the only path from the adjusted Rolle stationary read back to the original secant-slope comparison.

Finally, the structural rows H, C, P, N transport, replay, source, and name the same row inventory. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 1907.3 (Constructive mean-value Rolle factorization)**.** *Every mean-value witness exported by an accepted* $\mathsf{ConstructiveMeanValue}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{Rolle}^{\uparrow}$ *stationary packet R. The exported witness is the interior row W together with the terminal* $\mathsf{Real}^{\uparrow}$ *equality seal E between the derivative quotient route D and the secant-slope row S; it does not assert host differentiability, a selected point, quotient equality of completed reals, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $M = (I, S, R, D, W, E, H, C, P, N)$. The interval and secant data are visible before any witness is read, and the carrier exposes no other endpoint or quotient source.

Read the adjusted Rolle packet R. By chapter 1902, its stationary output is already constrained to the compact interval source, endpoint equality, continuous-map row, extreme-value witness, derivative quotient route, interior witness, and zero-seal boundary displayed inside the Rolle packet.

The mean-value carrier consumes exactly that stationary output. Its row W is the interior witness inherited from R, while D and E translate the adjusted zero-seal into the displayed equality between the original derivative quotient and the secant slope.

Transport and replay through H and C preserve this factorization, and P, N add only source and naming data. Thus every exported mean-value witness passes through R before it reaches D, W, E, with no hidden host differentiability, selected point, quotient equality, open-cover compactness, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{ConstructiveMeanValue}^{\uparrow}$ witness read factors through a compact interval source row, secant-slope row, adjusted $\mathsf{Rolle}^{\uparrow}$ packet, $\mathsf{Derivative}^{\uparrow}$ quotient route, interior witness row, terminal $\mathsf{Real}^{\uparrow}$ equality seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, continuousmap, compact-interval-locator, extremevalue, derivative, rolle: $\mathsf{ConstructiveMeanValue}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries secant-slope data, an adjusted $\mathsf{Rolle}^{\uparrow}$ stationary packet, derivative quotient comparison, an interior witness, a terminal equality seal, transport, replay, provenance, and local naming as one mean-value witness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveMeanValue}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveMeanValue}^{\uparrow}$ packet is generated from a compact interval source row, a secant-slope row, an adjusted $\mathsf{Rolle}^{\uparrow}$ packet, a $\mathsf{Derivative}^{\uparrow}$ quotient route, an interior witness row, a terminal $\mathsf{Real}^{\uparrow}$ equality seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveMeanValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1907.1, Definition 1907.2, and Definition 1907.3 seed the finite mean-value witness surface over compact interval, secant-slope, adjusted Rolle, derivative quotient, interior witness, and Real equality-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveMeanValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, selected points, quotient equality of completed reals, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, secant-slope construction, adjusted Rolle factorization, derivative quotient equality sealing, and TasteGate field faithfulness for BEDC.Derived.ConstructiveMeanValueUp.

$\mathsf{Cauchy}^{\uparrow}$ is an L10 route whose regular-sequence, finite-window, dyadic-tolerance, and terminal seal reads are supplied through chapter 301, chapter 126, chapter 291, and chapter 38.

Chapter 1908

A Concrete Naming Certificate for $\mathsf{CauchyRealLocator}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRealLocator}^{\uparrow}$ packet records the finite locator surface by which a displayed Cauchy real answers Real-facing precision and side-location requests. It refines the modulus route of chapter 1423 and chapter 292, reads finite observations through chapter 126, uses the regular rational handoff of chapter 301, records dyadic tolerances through chapter 291, and reaches the terminal seal of chapter 38. The packet is a traditional constructive-analysis locator object reconstructed as finite NameCert data; it does not select a host limit, identify quotient streams, invoke countable choice, decide trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1908.1 (Cauchy real locator carrier)**.** *A* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row. It records the displayed RealCauchyModulus and CauchyModulus threshold data used for every locator request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the locator at the requested precision and side test.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the positive dyadic scale, comparison cells, and slack budget for the displayed windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows into regular rational data before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *locator seal. It records only the Real-facing locator boundary delivered after M, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic ledger, regular readback, terminal locator seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a locator request through M, S, D, Q, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$*, RealCauchyModulus, CauchyModulus,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $M, S, D, Q, R, H, C, P, N$. It has no coordinate for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 1908.2 (Cauchy real locator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier*

$$L = (M, S, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the modulus schedule, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal Real locator seal, transport, replay, provenance, and local naming;*
2. *modulus exactness, requiring every locator request to use the displayed row M rather than an independent threshold, host convergence theorem, or selected limit;*
3. *window and tolerance compatibility, requiring the inspected windows in S to be compared only through the dyadic ledger D;*
4. *RegSeqRat readback, requiring the rational data consumed by the locator seal to pass through the displayed row Q;*
5. *route ordering, requiring R to be read only after M, S, D, Q have been displayed;*
6. *nonescape, excluding host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, and bridge checking.*

No local locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3410.1. The accepted packet displays exactly the modulus schedule, finite windows, dyadic ledger, regular readback, terminal locator seal, and structural rows.

Modulus exactness is forced by the single coordinate M. Every locator request reads its threshold from this row, so the packet has no second convergence witness, hidden host completeness theorem, or selected limit coordinate.

Window and tolerance compatibility are supplied by S and D. The stream row exposes only finite windows, while the dyadic row records the positive scale, comparison cells, and slack budget used to interpret those windows.

The readback and ordering clauses follow from Q and R. The regular rational row is the only rational readback coordinate, and the terminal Real locator seal occurs downstream of M, S, D, Q.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no row for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 1908.3 (Cauchy real locator modulus route)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier factors each locator request through the displayed finite route*

$$M \longmapsto S, D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Thus the request reaches the $\mathsf{Real}^{\uparrow}$ *locator seal only through the visible modulus schedule, finite stream windows, dyadic tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Proof. Begin with an accepted carrier $L = (M, S, D, Q, R, H, C, P, N)$ and a displayed locator request. The only threshold coordinate available to the request is M, so the modulus schedule is read first.

The finite observation step is next. The request opens the window row S and interprets its tolerance through D, whose dyadic ledger records the positive scale and comparison cells.

The rational handoff is then forced through Q. No independent stream or host rational approximation is present in the carrier, so the locator request reaches the Real-facing boundary only after the regular rational readback has been displayed.

Finally the terminal row R is consumed and the structural rows H, C, P, N preserve the same route. This proves the stated factorization and excludes a bypass through selected limits, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Theorem 1908.4 (Cauchy real locator Real seal nonescape)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *locator-seal read reaches* R *only after the displayed rows* M, S, D, Q *have been consumed. The seal exports no host Real equality, quotient stream representative, selected limit, countable-choice selector, trichotomy branch, open-cover compactness witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Inspect the terminal coordinate of the carrier. The only Real-facing locator seal is R, and it is listed after the modulus schedule, stream-window, dyadic-ledger, and regular-readback rows.

Apply the obligation surface of Definition 3410.2. It requires route ordering through M, S, D, Q before R can be read by a Real-facing consumer.

The structural rows preserve this boundary. The hsame transports and Cont replay keep the same displayed packet, while Pkg provenance and local naming record only that route.

Because no excluded coordinate appears in L, the terminal seal cannot export host Real equality, quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. Any consumer asking for one of those principles must leave the local carrier. □

Theorem 1908.5 (Cauchy real locator four-object exit)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *Real-facing locator read factors through exactly the four L10 exit objects named by Definition 3412.8: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *locator seal. In carrier coordinates, any accepted read of* $L = (M, S, D, Q, R, H, C, P, N)$ *uses* S, D, Q, R *as the four object faces and uses* M, H, C, P, N *only as the modulus schedule, transport, replay, provenance, and local naming support. It exports no quotient stream equality, host Real equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier tuple. The row S is the finite $\mathsf{StreamName}^{\uparrow}$ window, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and R is the terminal $\mathsf{Real}^{\uparrow}$ locator seal.

The modulus route theorem Definition 3410.3 fixes the order of consumption. A locator request reads the schedule M, opens the finite windows S, interprets them through the dyadic ledger D, passes through the regular readback Q, and only then reaches R.

The Real-seal nonescape theorem Definition 3410.4 supplies the boundary condition for the terminal read. It rules out every Real-facing bypass that would identify quotient streams, appeal to host Real equality, choose a limit, branch by trichotomy, use open covers, or import ambient completeness.

Finally compare this carrier projection with Definition 3412.8. The four object faces are exactly S, D, Q, R, while M, H, C, P, N supply only schedule, transport, replay, provenance, and naming support. Hence an accepted locator read has the claimed four-object L10 exit route and no additional exported coordinate. □

Theorem 1908.6 (Cauchy real locator induction route)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every locator-side read available to induction is obtained by the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

The induction step reads finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal rows without deciding trichotomy and without choosing a host limit.*

Proof. Project the carrier of Definition 3410.1. The modulus schedule M is the only threshold source, so a locator-side induction step begins by reading the same schedule used in the modulus route.

The finite observation is then S interpreted through D. The stream row exposes only bounded windows, while the dyadic row supplies the tolerance scale and comparison cells needed to use those windows.

The rational and Real-facing parts follow through Q and R. The modulus route theorem Definition 3410.3 fixes this order, and the four-object exit theorem Definition 3410.5 identifies S, D, Q, R as the four L10 exit faces.

Finally H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the carrier has no coordinate for trichotomy or selected host limits, the induction route cannot decide a side by trichotomy or choose a limit outside the displayed locator data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3410.1, Definition 3410.2, Definition 3410.3, Definition 3410.4, Definition 3410.5, and Definition 3410.6 seed the finite locator route over Cauchy modulus schedules, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and the Real locator seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, window and tolerance compatibility, RegSeqRat readback, route ordering, nonescape, and local NameCert registration for BEDC.Derived.CauchyRealLocatorUp.

Constructive story: A $\mathsf{CauchyRealLocator}^{\uparrow}$ packet is generated from a RealCauchyModulusUp and CauchyModulusUp schedule row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp locator seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1909

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 1909.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1909.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 1909.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 1909.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 1909.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 1909.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 1909.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1910

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1910.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through* $X, Y, W, D, A, M, S, H, P, N$.
9. *P is the* Pkg *provenance over* CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 1910.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* CauchySequenceAlgebra$^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *operation budgets,* RegularCauchyAddition$^{\uparrow}$ *route,* RegularCauchyProduct$^{\uparrow}$ *route,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* RegSeqRat$^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 1910.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 1910.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 1910.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 1910.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Cauchy$^{\uparrow}$ gateway reads its completion-facing source through chapter 126, chapter 301, chapter 291, and chapter 38 before any terminal seal is public.

Theorem 1910.7 (Cauchy L10 sibling route)**.** *The Cauchy packet family exported by this hub is consumed through the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

Concretely, the finite-window surface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the completion boundary of chapter 1506 are sibling inputs to the Cauchy carriers rather than hidden ambient structure.

Proof. Read any Cauchy carrier exported by the hub through its finite observation rows. The stream coordinate supplies the inspected windows, the regular-rational coordinate supplies the numerical readback, and the dyadic coordinate supplies the tolerance budget.

The Real-facing seal is reached only after those rows have been displayed. When a completion consumer is present, it receives the same displayed route through the metric-completion boundary.

Thus the hub's Cauchy packet family is attached to the Real and MetricCompletion siblings by visible L10 rows, with no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient completeness coordinate added by the route. □

The Cauchy$^{\uparrow}$ route consumes the L10 sibling chapters before the real locator and sequence algebra inputs are read. In particular, the regular-rational source of chapter 301, the finite stream windows of chapter 126, and the terminal seal of chapter 38 are already part of the displayed dependency spine for every downstream Cauchy locator or algebra consumer. The completion-side consumer in chapter 1937 reads the same RegSeqRat, StreamName, and RealUp spine before exposing its functorial completion route.

Theorem 1910.8 (Cauchy L10 four-object seal route)**.** *Every accepted* Cauchy$^{\uparrow}$ *Real-facing seal route used by the current L10 phase factors through the four displayed sibling faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Concretely, the finite window family of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38 are the only Real-facing sibling rows exported by the route. The supporting rows are the displayed BHist *carrier data, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *registration.*

Proof. Read the route through the existing CauchyRealLocator carrier and its Real-seal nonescape theorem. The carrier exposes a finite StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, and a terminal RealUp locator seal before any Real-facing consumer can read the packet.

Project those four named faces. The StreamName row supplies only inspected finite windows, the DyadicRatCore row supplies the tolerance and comparison budget, the RegSeqRat row supplies the rational readback, and the RealUp row supplies the terminal seal.

The structural rows preserve this factorization. The BHist data name the carried packet, hsame transports the displayed coordinates, Cont replays the same route, Pkg records provenance, and NameCert records the local accepted surface.

Since the carrier has no coordinate for quotient streams, host equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the Cauchy Real-facing seal route. □

Chapter 1911

A Concrete Naming Certificate for $\mathsf{Cauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{Cauchy}^{\uparrow}$ packet records the ordinary constructive Cauchy-real carrier as finite BEDC naming data. It sits above the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational sequence surface of chapter 301, and the terminal seal of chapter 38. The packet is not a quotient construction of reals; it is a finite route by which a modulus, rational readback, transport, replay, provenance, and local naming row become visible before any $\mathsf{Real}^{\uparrow}$ consumer can read the Cauchy object.

Definition 1911.1 (Cauchy carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the finite windows of rational approximants inspected by the carrier.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the positive dyadic scale, comparison cells, and slack conversions used to interpret each requested window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational readback consumed downstream.*
4. *M is the modulus row, recording the displayed threshold assignment used by Cauchy requests and refusing any hidden convergence theorem.*
5. *Q is the modulus/readback compatibility row, requiring the stream window, dyadic tolerance, regular rational sequence, and modulus rows to agree at every displayed precision request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after S, D, R, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-sequence, modulus, compatibility, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy request through $S, D, R, M, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $S, D, R, M, Q, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyUp.CauchyCarrier`

Theorem 1911.2 (Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *carrier*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 3413.1.*

Proof. Carrier admission is projection from Definition 3413.1. The accepted packet displays precisely the stream, dyadic, regular-sequence, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

The stability and exactness obligations are rowwise. The stream row supplies only finite windows, the dyadic row supplies only displayed tolerance cells, the regular-sequence row supplies only rational readback, and the modulus row supplies only the visible threshold assignment.

The compatibility row Q binds these four data rows before any $\mathsf{Real}^{\uparrow}$ seal can be consumed. The structural rows H, C, P, N then transport, replay, source, and name the same route without adding quotient equality, selected limits, host real equality, choice, trichotomy, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCarrier namecert obligations`

Theorem 1911.3 (Cauchy modulus readback route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer route factors through the displayed path*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a precision request reads finite $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the explicit modulus row, the compatibility row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal-facing boundary.*

Proof. Begin with an accepted carrier $X = (S, D, R, M, Q, E, H, C, P, N)$ and a displayed precision request. The only finite observations available to the request are the windows in S, and their tolerance scale is read through D.

The request next reads the explicit modulus row M. The compatibility row Q is the only coordinate that relates the chosen window, tolerance, threshold, and regular-rational readback, so the route cannot bypass it.

After Q is displayed, the consumer may read the $\mathsf{RegSeqRat}^{\uparrow}$ row R and then the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row E. The rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming.

Since the carrier has no quotient, selected-limit, host equality, countable-choice, trichotomy, open-cover, or ambient-completeness coordinate, the displayed route is the whole public Cauchy readback surface. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusReadbackRoute`

Theorem 1911.4 (Cauchy StreamName RegSeqRat window)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet exposes its* $\mathsf{RegSeqRat}^{\uparrow}$ *row only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and modulus/readback compatibility row. Thus a regular rational read of* $X = (S, D, R, M, Q, E, H, C, P, N)$ *must factor through* S, D, M, Q *before it can inspect* R*.*

Proof. Project the carrier of Definition 3413.1. The row S is the only finite observation surface, D is the only displayed tolerance surface, and M is the only modulus row available to the packet.

Apply Definition 3413.2 to identify Q as the compatibility row binding stream windows, dyadic tolerances, regular rational readback, and modulus data. Then apply Definition 3413.3: the displayed route reads S, D, M and Q before R.

The carrier has no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient completeness coordinate. Hence the stated window and compatibility rows exhaust the public access path to $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 1911.5 (Cauchy finite-tail criterion)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

be an accepted $\mathsf{Cauchy}^{\uparrow}$ *carrier. Suppose every displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window admitted by* S *and every dyadic tolerance cell admitted by* D *satisfy the tail bound named by the modulus row* M*, and suppose the compatibility row* Q *identifies those windows with the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of* X *is routed through the same finite tail ledger*

$$S, D, M \longmapsto Q \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The criterion exports no selected limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows available for a tail read are the finite StreamName window row S, the DyadicRatCore tolerance row D, the modulus row M, the compatibility row Q, the RegSeqRat readback row R, and the terminal seal row E.

Induct over the displayed window list retained by S. The empty window has no readback outside the recorded tolerance ledger. In the successor step, the new inspected position is checked against the dyadic cell in D and the threshold supplied by M, so the same finite tail bound remains attached to the extended window.

Apply Definition 3413.4. It forces any access to R through the displayed S, D, M, Q route. Apply Definition 3413.3 to identify Q as the point where the window, tolerance, modulus, and readback rows are jointly inspected.

Finally use Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal E is downstream of the regular rational readback, so no Real-facing consumer can bypass the finite tail ledger. Since the carrier has no selected-limit, quotient-stream, countable-choice, trichotomy, or ambient-completeness coordinate, the displayed finite induction exhausts the criterion. □

Definition 1911.6 (Cauchy finite-window modulus carrier)**.** *The finite-window modulus carrier associated to an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet is the displayed handoff surface*

$$W = (S, D, M, Q, R, E, H, C, P, N)$$

read from the carrier of Definition 3413.1. Its rows have the following roles:

1. *S records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by a tail request.*

2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells and slack conversions attached to those windows.*
3. *M records the explicit modulus threshold row used to accept later-window requests.*
4. *Q records the compatibility row tying each displayed window, dyadic tolerance, modulus threshold, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
5. *R records the regular-rational readback exposed only after S, D, M, Q have been displayed.*
6. *E records the downstream* $\mathsf{Real}^{\uparrow}$ *seal-facing row.*
7. *H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming for the same finite route.*

The carrier admits only these displayed coordinates. It has no coordinate for quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1911.7 (Cauchy finite-window tail induction)**.** *Let $W = (S, D, M, Q, R, E, H, C, P, N)$ be the finite-window modulus carrier of Definition 3413.6. Every accepted tail request obtained by extending the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row preserves the dyadic tolerance row D, the modulus threshold row M, the compatibility row Q, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. Thus the induction route is*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E$$

with H, C, P, N transporting, replaying, sourcing, and naming the same displayed handoff.

Proof. Project the underlying accepted Cauchy carrier through Definition 3413.1 and read it in the window/modulus order specified by Definition 3413.6. The only tail-request rows available are $S, D, M, Q, R, E, H, C, P, N$.

Induct over the displayed $\mathsf{StreamName}^{\uparrow}$ windows retained by S. The empty window has no tolerance request outside D and no readback outside the route named by Definition 3413.4.

For the successor step, the new inspected window is accepted only through the dyadic cell in D and the threshold supplied by M. Apply Definition 3413.5 to keep the same finite tail ledger attached to the extended request.

The compatibility row Q is then the unique displayed row where the window, dyadic tolerance, modulus threshold, and regular readback are jointly inspected. Apply Definition 3413.3 to place R after Q and E after R.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Since none of the rows supplies quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the induction cannot expose any such principle. □

Theorem 1911.8 (Cauchy RegSeqRat handoff obligation)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the finite-window modulus carrier of Definition 3413.6 and the tail-induction route of Definition 3413.7. In particular, the consumer reads S, D, M, Q before R, and any Real-facing use reads E only after that* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Begin with an accepted Cauchy packet and project its rows by Definition 3413.1. The carrier supplies finite stream windows, dyadic tolerances, a modulus row, a compatibility row, regular-rational readback, a terminal Real-facing seal, and structural transport, replay, provenance, and naming rows.

Apply Definition 3413.4. It forces access to the $\mathsf{RegSeqRat}^{\uparrow}$ row through the displayed S, D, M, Q route, so no consumer can inspect R independently of the finite window and tolerance data.

Next apply Definition 3413.5 and Definition 3413.7. These identify the finite tail ledger and show that accepted extensions preserve the dyadic tolerance, modulus threshold, compatibility, and regular readback rows before the Real seal is exposed.

Finally use Definition 3413.3. The structural rows H, C, P, N transport and name the same handoff, while the carrier contains no quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ consumer factors through the stated finite-window induction surface. □

Theorem 1911.9 (Cauchy RegSeqRat Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal real seal. In the notation of Definition 3413.1, the seal-facing route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

It does not expose quotient streams, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $X = (S, D, R, M, Q, E, H, C, P, N)$ from Definition 3413.1. The seal row E is declared terminal and seal-facing; it is not an independent source of Cauchy data.

Use Definition 3413.3 to order the displayed read. The route reaches R after S, D, M, Q and reaches E only after R has been displayed.

Finally apply Definition 3413.2. The rows H, C, P, N transport, replay, source, and name the same route, so they cannot insert quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient completeness between $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 1911.10 (Cauchy modulus transport stability)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

and

$$X' = (S', D', R', M', Q', E', H', C', P', N')$$

be accepted $\mathsf{Cauchy}^{\uparrow}$ *packets. Suppose the stream row, dyadic tolerance row, modulus row, modulus/readback compatibility row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, replay row, provenance row, and local naming row are transported componentwise by the displayed* hsame *data in* H *and* H'*, and suppose both packets are replayed through the same displayed* Cont *route. Then the public Cauchy readback route*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

is the same BEDC route as

$$S', D', M' \longmapsto Q' \longmapsto R' \longmapsto E' \longmapsto H', C', P', N'.$$

In particular, componentwise transport of the modulus/readback surface cannot introduce quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, or ambient completeness.

Proof. Project both accepted packets through Definition 3413.1. The only visible coordinates are the stream, dyadic tolerance, regular-rational readback, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

Apply Definition 3413.3 to X and to X'. The two projected routes must read their displayed stream, dyadic, and modulus rows before their compatibility rows, and the shared Cont replay identifies these reads through the componentwise hsame data.

Next use Definition 3413.4. Access to R and R' is forced through the transported S, D, M, Q and S', D', M', Q' rows, so no other regular-rational readback path is available.

Finally apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal remains downstream of the transported $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the structural rows H, C, P, N and H', C', P', N' only transport, replay, source, and name that path. Since neither carrier contains quotient streams, selected limits, host equality, choice, trichotomy, or ambient completeness, the two displays name the same BEDC route. □

Theorem 1911.11 (Cauchy uniform-modulus Real seal)**.** *For an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

any uniform-modulus consumer of the explicit modulus row M *that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must read* S, D, M, Q *before* R*, and* R *before* E*. The route contains no quotient stream equality, selected limit, countable choice, trichotomy, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 3413.1. The explicit modulus row M is packaged with the finite stream window S, the dyadic tolerance row D, the compatibility row Q, the regular rational readback R, and the terminal seal-facing row E.

Apply Definition 3413.3. It forces a precision request to read S, D, M and then Q before any regular-rational row can be inspected.

Next apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of R, so a uniform-modulus consumer cannot reach the seal before the regular rational readback has been displayed.

Finally use Definition 3413.10. Componentwise transport and the displayed Cont replay preserve the same route for every accepted consumer, and the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1911.12 (Cauchy root filter-completion anchor)**.** *Every Cauchy root consumer that first reaches the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *surface of chapter 1569 must already have read the Cauchy core route, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Project the root hub through its Cauchy core child. The core route exposes the stream-window, tolerance, readback, transport, replay, provenance, and naming rows before any completion-facing consumer can inspect the root packet.

Now read the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ chapter. Its completion handoff consumes the same finite windows and regular rational readback before the terminal Real seal is visible, so the root consumer cannot enter through a detached filter object.

The structural rows preserve the path. Componentwise hsame transport and displayed Cont replay keep the consumer inside the Cauchy root packet, while Pkg provenance and the local $\mathsf{NameCert}$ row record that no quotient filter, selected limit, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported. □

Theorem 1911.13 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 1911.14 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream

consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 1911.15 (Cauchy finite-window modulus pullback)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *modulus read pulls back to the same finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any consumer can inspect a Cauchy value. In the notation of Definition 3413.1, the pullback route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No quotient stream equality, selected limit, host real equality, countable-choice coordinate, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is available along this route.*

Proof. Project the accepted carrier through Definition 3413.1. The finite stream window S, dyadic tolerance row D, modulus row M, compatibility row Q, regular-rational readback R, and terminal Real seal E are the only rows available before the structural rows are read.

Apply Definition 3413.2. It identifies the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ surface with stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility, Real seal ordering, transport, replay, provenance, and local naming.

Now use Definition 3413.3. A precision request first reads S, D, M, Q, then reaches R, and only then reaches E. The route is sharpened by Definition 3413.4, which keeps the $\mathsf{RegSeqRat}^{\uparrow}$ row behind the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, modulus, and compatibility rows.

Finally apply Definition 3413.9. The terminal Real seal is reached only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback has been displayed. Thus every accepted modulus read pulls back to the same four L10 sibling rows before a consumer sees a Cauchy value, and none of the excluded quotient, selected-limit, choice, order, compactness, or completeness coordinates can be inserted. □

Theorem 1911.16 (Regular-Cauchy modulus pullback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of the* $\mathsf{Cauchy}^{\uparrow}$ *finite-window modulus route obtains a dyadic tolerance schedule that is cofinal with the route displayed in Definition 3413.15. In the notation of Definition 3413.1, the consumer may read*

$$S, D, M, Q \longmapsto R \longmapsto E$$

only as a regular-rational handoff: the $\mathsf{Real}^{\uparrow}$ *seal is reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, not before it. The route exports no quotient stream equality, selected limit, host real equality, choice principle, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3413.15. It fixes the finite route from the StreamName window S, dyadic tolerance row D, modulus row M, and compatibility row Q into the regular-rational readback R, and only then into the terminal Real seal E.

Project the route at the $\mathsf{RegSeqRat}^{\uparrow}$ boundary. A RegSeqRat-facing consumer reads the dyadic tolerance row D through the same finite windows S and compatibility row Q, so its tolerance requests are bounded by a cofinal schedule already displayed by the Cauchy modulus route.

Transport the schedule through the regular-rational classifier row. The readback row R is the first row that presents the schedule as regular-rational data, and the Real seal E is reachable only after that readback has been displayed.

Thus the pullback is cofinal at the RegSeqRat surface and terminal at the Real seal. Since the carrier has no coordinate for quotient stream equality, selected limits, host equality, choice,

trichotomy, or Real completeness, none of those principles can be inserted into the consumer route. □

Theorem 1911.17 (Cauchy finite-window pullback uniqueness)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing Cauchy modulus consumer accepted by the* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the same finite route*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

displayed in Definition 3413.15. The consumer cannot replace this route by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows visible to a modulus consumer are the finite StreamName window, dyadic tolerance ledger, modulus row, compatibility row, RegSeqRat readback, Real seal, and the structural rows.

Apply Definition 3413.2. It identifies the local NameCert surface with precisely those rows, so there is no second carrier coordinate from which an alternative Real-facing route could be read.

Use Definition 3413.3. A precision request reaches R only after S, D, M, Q have been displayed, and it reaches the terminal seal E only after the regular-rational readback has been displayed.

Finally apply Definition 3413.15 and Definition 3413.16. The first theorem fixes the finite pullback route to the L10 sibling rows, while the second shows that every RegSeqRat-facing consumer receives the same cofinal handoff. Hence any Real-facing Cauchy modulus consumer factors through the displayed route and gains no quotient, selected-limit, choice, host-equality, or completeness coordinate. □

Every completion-facing consumer of the $\mathsf{Cauchy}^{\uparrow}$ hub reads the finite Cauchy packet through chapter 301 and chapter 38 before any Real seal is exposed to completion rows. The route first displays the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the dyadic tolerance ledger, then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal, and only then lets Cauchy completion consumers inspect the same BHist, hsame, Cont, Pkg, and local NameCert rows. Thus the hub records a structural dependency route, not quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 1911.18 (Cauchy RegSeqRat Real forward dependency)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *source-window read consumed by a Real-facing downstream chapter must pass through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is exposed. The route reads only displayed finite windows, regular rational tail data, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier route in Definition 3413.1. Its source-window reads are displayed before the regular rational readback and terminal real seal rows are available.

Use the regular readback and real-seal routes already named by the Cauchy child files. They order downstream consumption through $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$, with the structural rows transporting and naming the same finite windows.

Since the hub exposes only child-file surfaces and no independent carrier coordinate, a Real-facing consumer can cite $\mathsf{Cauchy}^{\uparrow}$ only through that forward dependency route. The read therefore reaches chapter 301 and chapter 38 before any Real-facing seal is exposed. □

Theorem 1911.19 (Cauchy located-uniform filter forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *route whose explicit modulus row is consumed by a located or uniform Cauchy filter completion must pass through the displayed carrier route*

$$S, D, M, Q, R, E, H, C, P, N$$

of Definition 3413.1, then through chapter 1661 or chapter 1694, before any $\mathsf{Real}^{\uparrow}$ *seal-facing completion row is exported. The forward route has no quotient stream equality, selected limit, countable choice, metric-completeness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host equality.*

Proof. Start with the explicit modulus readback of the Cauchy carrier. By Definition 3413.3, a precision request reads the finite StreamName row S, the DyadicRatCore row D, the explicit modulus row M, and the compatibility row Q before it can inspect the regular rational row R.

Apply Definition 3413.9. The seal-facing row E is downstream of R, so a Real-facing completion consumer cannot bypass the regular rational readback.

Now use Definition 3413.11. Any uniform-modulus consumer of M that reaches a $\mathsf{Real}^{\uparrow}$ seal preserves the same ordering S, D, M, Q before R and R before E.

The remaining completion-neighbour branch is supplied only by the sibling chapters chapter 1661 and chapter 1694. These chapters consume the already displayed Cauchy route as located or uniform filter evidence; neither adds quotient stream equality, selected limits, choice, metric-completeness data, ambient completeness, or host equality to the core Cauchy carrier. □

Theorem 1911.20 (Cauchy route to CauchyCriterion)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *hub's finite route is consumed by chapter 392 through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exports no detached completion carrier, quotient sequence equality, selected modulus outside the displayed rows, countable-choice schedule, or host completeness principle.*

Proof. Read the hub through its existing finite siblings. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows provide the only Cauchy data visible to downstream criteria.

The structural rows hsame, Cont, Pkg, and NameCert transport, replay, source, and name that same finite route. Therefore the Cauchy criterion consumer receives exactly the displayed BEDC rows and no external completion or choice object. □

Theorem 1911.21 (Cauchy hub Bishop completion forward route)**.** *The regular-source material routed by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by chapter 1639 through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The forward route exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness principle.*

Proof. Read the Cauchy hub through its regular-source children. Those children expose dyadic tolerance arithmetic, finite stream windows, regular rational readback, and the terminal real seal as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The Bishop regular Cauchy completion chapter consumes exactly that L10-facing route. Its completion read is mediated by the same finite BHist carrier discipline, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Since the hub route contains no detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness coordinate, the forward edge to the Bishop completion packet is precisely the displayed finite dependency route. □

Chapter 1912

A Concrete Naming Certificate for $\mathsf{MeanValueTheorem}^{\uparrow}$

Following Lagrange, Cauchy, and the Bishop–Bridges reading of constructive analysis, the $\mathsf{MeanValueTheorem}^{\uparrow}$ packet records the theorem-facing finite BEDC boundary between endpoint secant data and a displayed derivative quotient row. It consumes the terminal real-name surface of chapter 38, the graph and modulus rows of chapter 130, the derivative quotient interface of chapter 122, the Rolle handoff of chapter 1900, the bisection-facing route of chapter 1402, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The chapter names a finite route from displayed endpoint values and secant slope to a theorem-facing derivative row; it is not a host differentiability theorem, an equality of quotient reals, or a selected interior point in an ambient completed line.

Definition 1912.1 (Mean value theorem carrier)**.** *A* $\mathsf{MeanValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, S, F, D, R, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *A, B are the endpoint* $\mathsf{Real}^{\uparrow}$ *name rows for the displayed interval boundary and endpoint value reads.*
2. *S is the secant-slope seal row, binding the endpoint difference and interval-width readback to a displayed* $\mathsf{Real}^{\uparrow}$ *slope seal rather than host division in a completed field.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the real-valued map on the displayed interval source.*
4. *D is the* $\mathsf{Derivative}^{\uparrow}$ *quotient row, exposing only the inspected metric quotient route used by the theorem-facing conclusion.*
5. *R is the* $\mathsf{RolleTheorem}^{\uparrow}$ *handoff row, carrying the finite Rolle-facing packet applied to the usual auxiliary difference against the secant slope.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint values, interval width, secant windows, and derivative-quotient inspections.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger accounting for the dyadic windows used by S, Q, D, and the Rolle handoff row.*

8. *H records componentwise* hsame *transport for endpoint names, secant-slope seals, continuous-map rows, derivative quotients, Rolle handoff rows, rational readbacks, dyadic ledgers, replay rows, provenance rows, and local naming rows.*
9. *C records displayed* Cont *replay from endpoint reads to the secant seal, from map windows to the auxiliary Rolle packet, from rational readback to dyadic tolerance checks, and from the Rolle handoff to the theorem-facing derivative row.*
10. *P is the* Pkg *provenance over* $\mathsf{MeanValueTheorem}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RolleTheorem}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MeanValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted theorem packets only through these rows. It has no coordinate for host differentiability, quotient real equality, a selected interior point, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.MeanValueTheoremUp.MeanValueTheoremTasteGate single carrier alignment`

Theorem 1912.2 (Mean value theorem NameCert obligations)**.** *For every accepted* $\mathsf{MeanValueTheorem}^{\uparrow}$ *carrier*

$$M = (A, B, S, F, D, R, Q, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MeanValueTheorem}^{\uparrow}}$ *surface consists exactly of the finite obligation rows for endpoint real-name admission, secant-slope sealing, continuous-map admission, derivative quotient inspection, Rolle handoff, regular rational readback, dyadic tolerance budgeting, componentwise transport, displayed replay, provenance, and local naming. No obligation row may bypass the displayed* $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RolleTheorem}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *sources.*

Proof. Carrier admission is projection from Definition 1912.1. The endpoint rows A, B supply the displayed real names, and the row S is the only secant-slope seal available to the packet.

The map and derivative surfaces are fixed next. The row F supplies the continuous-map data, while D supplies the displayed derivative quotient route.

The theorem-facing handoff then passes through R. That row is the only Rolle packet in the carrier, and the readback and tolerance rows Q, T are the only finite arithmetic ledgers attached to it.

Finally, H, C, P, N preserve, replay, source, and name the same rows. Since the carrier has no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1912.3 (Mean value theorem Rolle handoff)**.** *Every theorem-facing derivative row exported by an accepted* $\mathsf{MeanValueTheorem}^{\uparrow}$ *carrier factors through the displayed Rolle handoff row R. The route reads the auxiliary difference against the secant-slope seal through A, B, S, F, Q, T, then exports the derivative quotient row D only after the Rolle packet has supplied the stationary-zero boundary.*

Proof. Start with an accepted carrier $M = (A, B, S, F, D, R, Q, T, H, C, P, N)$. The endpoint rows and secant-slope seal determine the auxiliary difference data before any theorem-facing derivative conclusion is available.

The continuous-map row F, regular readback Q, and dyadic ledger T supply the finite observations consumed by the auxiliary packet. These rows are the only route from endpoint data to the displayed Rolle input.

Read the Rolle handoff row R. It supplies the stationary-zero boundary for the auxiliary difference, and the derivative quotient row D is exported only through that boundary.

Transport and replay through H and C preserve this order, while P and N add only provenance and local naming. Thus the derivative row is routed through R rather than through a host mean-value theorem or a selected ambient point. □

Theorem 1912.4 (Mean value theorem Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal consumed or exported by an accepted* $\mathsf{MeanValueTheorem}^{\uparrow}$ *carrier is routed through one of the displayed endpoint, secant-slope, regular-readback, dyadic-ledger, Rolle-handoff, or derivative-quotient rows of Definition 1912.1. Consequently the carrier cannot export host differentiability, quotient real equality, selected interior points, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge data.*

Proof. Inspect the real-facing coordinates of the carrier. The endpoint rows A, B expose the initial real names, S exposes the secant-slope seal, and Q, T expose the finite readback and tolerance budget.

The Rolle handoff row R is the only theorem route from those seals to a stationary-zero boundary. The derivative quotient row D is therefore read as a displayed Real seal attached to that route.

The structural rows do not add new real data. They transport, replay, source, and name the same carrier inventory.

There is no coordinate for host differentiability, quotient real equality, a selected interior point, trichotomy, ambient Real completeness, or a bridge object. Therefore every Real seal remains inside the displayed packet boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{MeanValueTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MeanValueTheorem}^{\uparrow}$ packet is generated from endpoint real-name rows, a secant-slope seal, a continuous-map row, a derivative quotient row, a Rolle handoff row, regular rational readback, a dyadic tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeanValueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1912.1, Definition 1912.2, Definition 1912.3, and Definition 1912.4 seed the finite theorem-facing mean-value boundary over displayed endpoint, secant-slope, continuous-map, derivative, Rolle, regular-rational, and dyadic rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeanValueTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, quotient real equality, selected interior points, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligation coverage, Rolle handoff routing, Real seal boundary, and TasteGate field faithfulness for BEDC.Derived.MeanValueTheoremUp.

Chapter 1913

A Concrete Naming Certificate for $\mathsf{BolzanoCauchy}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BolzanoCauchy}^{\uparrow}$ packet records the constructive bidirectional Cauchy convergence criterion as finite BEDC NameCert data. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal surface of chapter 38, the explicit criterion route of chapter 1509, and the limit-seal boundary of chapter 386. The packet names the two constructive directions as displayed rows; it is not ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, a selected host limit, trichotomy, open-cover compactness, countable choice, or bridge checking.

Definition 1913.1 (Bolzano-Cauchy carrier)**.** *A* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, D, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposing exactly the finite observations of the candidate sequence.*
2. *M is the Cauchy modulus data assigning each dyadic tolerance request a displayed threshold for the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic scales, rational comparisons, and exhaustion budget used by both directions.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from S, M, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row obtained after the criterion route has displayed the regular rational handoff.*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row that binds the same S, M, D, R, E route to a limit-seal consumer.*
7. *H records componentwise* hsame *transport for the stream schedule, modulus data, dyadic ledger, regular readback, Real seal, limit-seal compatibility, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through S, M, D, R, E, L and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{BolzanoCauchy}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BolzanoCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, trichotomy, open-cover compactness, countable choice, or a detached limit seal.*

Lean4 ✓ `BEDC.Derived.BolzanoCauchyUp.BolzanoCauchyTasteGate single carrier alignment`

Theorem 1913.2 (Bolzano-Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier*

$$B = (S, M, D, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BolzanoCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed stream schedule, Cauchy modulus data, dyadic tolerance ledger, regular readback, Real seal, limit-seal compatibility, transport, continuation, provenance, and local name;*
2. *Cauchy-to-limit routing, requiring any Cauchy-read consumer to pass through S, M, D, R before E or L is available;*
3. *limit-to-Cauchy routing, requiring any limit-seal read to replay the same S, M, D, R tolerance route before it can export a Cauchy conclusion;*
4. *ledger exactness, requiring every tolerance request in either direction to be accounted for by D and sourced by P;*
5. *structural stability, requiring H and C to preserve and replay only the displayed rows.*

No Bolzano-Cauchy obligation is stored outside these rows.

Proof. Project the carrier of Definition 1913.1. The visible coordinates are exactly the stream schedule, modulus data, dyadic ledger, regular readback, Real seal, limit-seal compatibility, transport, continuation, provenance, and local naming rows.

The two route obligations are forced by the displayed order. A Cauchy-read consumer has no source except the windows S, thresholds M, ledger D, and readback R before the Real and limit-seal rows are exposed. Conversely, a limit-seal consumer reading L must replay the same finite tolerance route before it can return a Cauchy conclusion.

The ledger and structural rows then exhaust the remaining obligations. The dyadic row D accounts for every tolerance request, the provenance row P records its source, and H, C can only transport and replay the displayed packet. Since the carrier has no additional coordinate, the five listed rows are the complete local NameCert surface. □

Theorem 1913.3 (Bolzano-Cauchy Cauchy-to-limit route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier that begins from the Cauchy side factors through the finite route*

$$S \longmapsto M \longmapsto D \longmapsto R \longmapsto E \longmapsto L$$

and then through the stored hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The route is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff read through a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility boundary; it exports no ambient completeness theorem, quotient stream, selected host limit, trichotomy row, open-cover compactness row, countable choice principle, or detached seal.*

Proof. Start with an accepted Cauchy-side read. By carrier admission, the only sequence data are the finite windows S, and the only tolerance data are the threshold row M and dyadic ledger D.

Apply the existing convergence-criterion route from chapter 1509. Its readback stage is the displayed $\mathsf{RegSeqRat}^{\uparrow}$ row R, and its Real-facing endpoint is the seal row E.

The Bolzano-Cauchy packet adds no new endpoint between E and the limit-seal consumer. The compatibility row L reads the same finite handoff, while H, C, P, N transport, replay, source, and name that packet.

Thus the Cauchy-side read reaches the limit-seal boundary only through the displayed route. The carrier inventory excludes ambient completeness, quotient streams, selected host limits, trichotomy, open-cover compactness, countable choice, and detached seals. □

Theorem 1913.4 (Bolzano-Cauchy limit-to-Cauchy route)**.** *Every limit-facing read of an accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier that begins from the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row factors back through the same displayed route*

$$L \longmapsto E \longmapsto R \longmapsto D \longmapsto M \longmapsto S$$

as a Cauchy conclusion over the finite stream windows. The reverse direction uses only the stored compatibility, Real seal, regular readback, dyadic ledger, modulus data, stream schedule, transport, continuation, provenance, and local NameCert rows; it does not use ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, trichotomy, open-cover compactness, countable choice, or bridge checking.*

Proof. Begin with a limit-facing read of the row L. By Definition 1913.1, that row is compatible only with the displayed S, M, D, R, E route and has no independent source of sequence or limit data.

Read the Real and regular-rational stages next. The row E is downstream of R, and R is downstream of the same dyadic ledger and modulus data used by the Cauchy side. Hence the reverse read cannot bypass D or M.

Finally return to the finite stream windows. The schedule S is the only observation source, and H, C, P, N merely preserve, replay, source, and name the same packet. Therefore the limit-facing read yields a Cauchy conclusion only over the displayed finite windows, with no appeal to ambient completeness, quotient streams, selected host limits, trichotomy, open-cover compactness, countable choice, or bridge checking. □

Theorem 1913.5 (Bolzano-Cauchy subsequence window extraction)**.** *An accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *packet with a bounded finite* $\mathsf{Real}^{\uparrow}$ *sequence window exposes a finite subsequence-window route before any Cauchy read or limit read is exported. The extracted route is carried by the displayed* $\mathsf{StreamName}^{\uparrow}$ *window schedule* S*, dyadic tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, together with the same* M, L, H, C, P, N *support rows of the packet.*

This extraction exports no selected host limit, countable-choice subsequence selector, ambient completeness principle, open-cover compactness row, quotient stream equality, or detached limit seal.

Proof. Project the carrier of Definition 1913.1. The bounded finite sequence window can only be read through the displayed StreamName schedule S, the modulus data M, the dyadic ledger D, and the regular readback row R.

Apply the NameCert obligations of Definition 1913.2. They force any Cauchy-side or limit-side consumer to use the displayed window, modulus, ledger, readback, seal, and compatibility rows before the local certificate can be consumed.

Use Definition 1913.3 and Definition 1913.4. Both directions pass through the same S, M, D, R, E, L surface, so a finite subsequence-window read is exposed at the StreamName and RegSeqRat stages before either route exports a Cauchy or limit conclusion.

The structural rows H, C, P, N only transport, replay, source, and name that finite packet. Because the carrier has no coordinate for selected host limits, countable-choice selectors, ambient completeness, open-cover compactness, quotient stream equality, or detached limit seals, the extraction remains a finite subsequence-window route. □

Theorem 1913.6 (Bolzano-Cauchy ledger exactness)**.** *For any accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier* $B = (S, M, D, R, E, L, H, C, P, N)$*, the Cauchy-to-limit route and the limit-to-Cauchy route expose the same finite ledger rows* $S, M, D, R, E, L, H, C, P, N$*. In particular, no consumer can insert a selected host limit, quotient stream equality, or ambient completeness coordinate between the* $\mathsf{Real}^{\uparrow}$ *seal row* E *and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row* L*.*

Proof. Apply Definition 1913.2. The obligation surface lists one stream schedule, one modulus row, one dyadic tolerance ledger, one regular readback, one Real seal, one limit-seal compatibility row, and the structural rows H, C, P, N.

Use Definition 1913.3 for the forward direction. It routes a Cauchy-side read through S, M, D, R, E, L and then through the structural rows, so the ledger visible before L is exactly the displayed finite packet.

Use Definition 1913.4 for the reverse direction. It reads the same compatibility row L back through E, R, D, M, S, with H, C, P, N preserving the packet. Because both routes consume the same rows and the carrier has no coordinate between E and L, selected host limits, quotient stream equality, and ambient completeness cannot be inserted there. □

Theorem 1913.7 (Bolzano-Cauchy RegSeqRat window factorization)**.** *For any accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier* $B = (S, M, D, R, E, L, H, C, P, N)$*, both the Cauchy-to-limit read and the limit-to-Cauchy read factor through the same finite* S, M, D, R *window before either the* $\mathsf{Real}^{\uparrow}$ *seal row* E *or the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row* L *is exposed. In particular, no quotient stream, selected host limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can be inserted between the* $\mathsf{RegSeqRat}^{\uparrow}$ *window* R *and the seal rows* E, L*.*

Proof. Use Definition 1913.3 for the Cauchy-to-limit direction. Its read reaches the $\mathsf{Real}^{\uparrow}$ and limit-seal rows only after the finite stream schedule S, modulus row M, dyadic ledger D, and regular readback R have been displayed.

Use Definition 1913.4 for the limit-to-Cauchy direction. Reading back from L passes through E, R, D, M, S, so the reverse route exposes the same S, M, D, R window before returning a Cauchy conclusion.

Finally apply Definition 1913.6. The two routes consume the same finite ledger rows, and the carrier has no coordinate between R and the seal rows. Hence quotient streams, selected host limits, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be inserted into the factorization. □

Theorem 1913.8 (Bolzano-Cauchy uniform subsequence modulus)**.** *For any accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier* $B = (S, M, D, R, E, L, H, C, P, N)$*, the displayed* S, M, D, R, E, L *window factorization can be consumed by a localized Cauchy subsequence or uniform Cauchy rate packet only through the* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows. In particular, the localized consumer of chapter 6046 may read the factorization as a finite subsequence-window route, but it receives no selected host limit, countable-choice subsequence selector, ambient completeness principle, quotient stream equality, or detached limit seal.*

Proof. Start with the window factorization of Definition 1913.7. It displays the finite route through S, M, D, R before the seal rows E, L are public.

Apply Definition 1913.5. The same route exposes the StreamName, RegSeqRat, DyadicRatCore, and Real-facing rows needed by a subsequence consumer before any Cauchy or limit conclusion is exported.

The localized consumer in chapter 6046 reads exactly those rows as its downstream input. Since the Bolzano-Cauchy carrier has no selected host limit, countable-choice selector, ambient completeness coordinate, quotient stream equality, or detached seal, none of those objects can be exported with the handoff. □

Theorem 1913.9 (Bolzano-Cauchy limit-readback nonescape)**.** *A* $\mathsf{CauchyLimitSeal}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BolzanoCauchy}^{\uparrow}$ *carrier*

$$B = (S, M, D, R, E, L, H, C, P, N)$$

can read back a Cauchy conclusion only through $L, E, R, D, M, S, H, C, P, N$ *and the same* S, M, D, R *window used by the Cauchy-to-limit direction. The readback exports no selected host limit, quotient stream equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, open-cover compactness, countable choice, or detached seal data.*

Proof. Apply the limit-to-Cauchy route of Definition 1913.4. A limit-facing read starts at L and returns through E, R, D, M, S, with H, C, P, N preserving, replaying, sourcing, and naming the same finite packet.

Use ledger exactness from Definition 1913.6. The reverse read consumes the same rows as the forward Cauchy-to-limit route, so no additional limit coordinate can be inserted between the $\mathsf{Real}^{\uparrow}$ seal and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ compatibility row.

Apply RegSeqRat window factorization from Definition 1913.7. Both directions pass through the same S, M, D, R window before either seal row can be exported.

Finally use the uniform subsequence modulus of Definition 1913.8 for localized consumers. The localized route still reads only the StreamName schedule, DyadicRatCore ledger, RegSeqRat readback, Real seal, limit compatibility, and structural rows, so the excluded host, quotient, order, compactness, choice, and detached-seal data remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BolzanoCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{BolzanoCauchy}^{\uparrow}$ packet is generated from a finite StreamName window schedule, Cauchy modulus data, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, CauchyLimitSeal compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BolzanoCauchy}^{\uparrow}}$, obligation)

Scope closed: Definition 1913.1, Definition 1913.2, Definition 1913.3, Definition 1913.4, Definition 1913.6, and Definition 1913.7 display carrier admission, Cauchy-to-limit routing, limit-to-Cauchy routing, ledger exactness, structural stability, and RegSeqRat window factorization for the finite bidirectional Cauchy convergence criterion.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BolzanoCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient RealUp completeness, quotient stream equality, selected host limits, trichotomy, open-cover compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-to-limit routing, limit-to-Cauchy routing, ledger exactness, and structural stability for BEDC.Derived.BolzanoCauchyUp.

Chapter 1914

A Concrete Naming Certificate for $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$ packet records bounded monotone real convergence as finite BEDC NameCert data. It sits over the bounded sequence surface of chapter 1415, the located upper-bound surface of chapter 1427, the monotone tail discipline of chapter 438, the bidirectional convergence criterion of chapter 1913, the regular rational handoff of chapter 301, the stream-window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter packages the constructive least-upper-bound route available for a displayed bounded monotone sequence; it does not import an ambient completeness theorem for arbitrary real data.

Definition 1914.1 (Bishop monotone real limit carrier)**.** *A* $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, B, M, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row, carrying the inspected real sequence windows, regular rational readbacks, real-name seals, and boundedness surface.*
2. *B is the bound ledger, recording the finite upper-bound and lower-window data read from S by a consumer requesting a monotone convergence witness.*
3. *M is the monotone comparison data, displaying the finite comparison rows that certify the inspected windows are nondecreasing along the requested index route.*
4. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row, reached from the same bound ledger and comparison data and read only as a located upper-bound and lower-approximation surface.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and Cauchy-modulus ledger, recording each requested dyadic precision, the monotone tail window selected for that precision, and the comparison budget used to trap later terms.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows, monotone comparisons, located-supremum approximants, and dyadic tolerance ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal descent row reached only after S, B, M, L, D, R have been displayed.*

8. *H records componentwise* hsame *transport for the sequence source, bound ledger, monotone comparisons, located supremum, dyadic ledger, regular readback, real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through the bounded source, bound ledger, monotone row, located supremum, dyadic ledger, regular readback, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{BolzanoCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, real trichotomy, selected subsequences, countable choice, quotient stream equality, open-cover compactness, or a host-level supremum operator.*

Lean4 ✓ `BEDC.Derived.BishopMonotoneRealLimitUp.BishopMonotoneRealLimitTasteGate single carrier alignment`

Theorem 1914.2 (Bishop monotone real limit NameCert obligations)**.** *For every accepted* BishopMonotoneRealLim *carrier*

$$T = (S, B, M, L, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed bounded sequence source, bound ledger, monotone comparison data, located supremum, dyadic tolerance ledger, regular readback, real seal, transport, continuation, provenance, and local naming row;*

2. *monotone bounded-source routing, requiring every convergence read to begin with S, B, M before any located supremum or real-seal row is exposed;*

3. *located-supremum routing, requiring lower approximants and upper-bound reads to pass through L with the same bound ledger and comparison data;*

4. *Cauchy-route routing, requiring each dyadic precision request to pass through D and then through R, E before the limit-facing read is available;*

5. *structural stability, requiring H and C to transport and replay only the displayed packet, with provenance supplied by P and local naming supplied by N.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1914.1. The displayed coordinates are exactly the bounded sequence source, bound ledger, monotone comparisons, located supremum, dyadic tolerance ledger, regular readback, real seal, transport, continuation, provenance, and local name.

The first analytic route is forced by the order of the packet. A convergence consumer must read the bounded sequence surface and the monotone comparison data before it can inspect the located supremum row.

The second route is the Cauchy handoff. A precision request enters the dyadic ledger, obtains a finite monotone tail window, passes through the regular rational readback, and reaches the real seal only after those rows have been displayed.

The structural rows exhaust the remaining obligations. The transports and continuations can preserve and replay only the displayed packet, while the provenance and local name add no analytic coordinate. Hence there is no further obligation outside the listed rows. □

Theorem 1914.3 (Bishop monotone real limit Cauchy route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto B \longmapsto M \longmapsto L \longmapsto D \longmapsto R \longmapsto E$$

and then through the stored hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. The route is the bounded monotone sequence read through located supremum data and the Bolzano-Cauchy Cauchy criterion; it exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, trichotomy row, selected subsequence, countable-choice operator, quotient stream equality, or open-cover compactness row.*

Proof. Begin with a real-facing read. By carrier admission, the only sequence source is S, and the only bound and monotonicity data are the displayed rows B and M.

Read the located supremum row next. The row L supplies the upper-bound and lower-approximation surface already available in chapter 1427, but it is attached here to the same bounded monotone source.

The dyadic ledger D then converts a requested precision into a finite monotone tail window. The regular readback R and real seal E consume that same window, matching the Cauchy-route discipline of chapter 1913.

Finally H, C, P, N transport, replay, source, and name the route. Since the carrier contains none of the excluded analytic objects, the real-facing read cannot leave the displayed path. □

Theorem 1914.4 (Bishop monotone real limit nonescape)**.** *Every public read of an accepted* $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$ *carrier is exhausted by the rows* $S, B, M, L, D, R, E, H, C, P, N$*. In particular, the packet cannot export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, real trichotomy, a selected subsequence, countable choice, quotient stream equality, open-cover compactness, or a host-level supremum operation.*

Proof. Inspect the carrier coordinates. The analytic rows are the bounded sequence source, the bound ledger, the monotone comparison data, the located supremum row, the dyadic Cauchy ledger, the regular readback, and the real seal.

The only remaining coordinates are structural. Transport H preserves componentwise rows, continuation C replays displayed reads, provenance P records the sibling sources, and N names the local certificate.

No coordinate remains for completeness, trichotomy, selected subsequences, countable choice, quotient equality, open-cover compactness, or a host-level supremum operator. Therefore every public read is confined to the finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$ packet is generated from bounded real sequence data, a bound ledger, monotone comparison data, a LocatedSupremum row, a DyadicRatCore tolerance and Cauchy-modulus ledger, RegSeqRat readback, RealUp seal descent, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopMonotoneRealLimit}^{\uparrow}}$, seed)

Scope closed: Definition 1914.1, Definition 1914.2, Definition 1914.3, and Definition 1914.4 seed the bounded monotone real convergence packet over bounded sequence data, located supremum data, dyadic tolerance rows, regular readback, and RealUp seal descent.

Formal verification: FormalStatus(NameCert$_{\text{BishopMonotoneRealLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient RealUp completeness, trichotomy, selected subsequences, countable choice, quotient stream equality, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone bounded-source routing, located-supremum routing, Cauchy-route routing, structural stability, and nonescape for BEDC.Derived.BishopMonotoneRealLimitUp.

Chapter 1915

A Concrete Naming Certificate for $\mathsf{CauchyFilterBase}^{\uparrow}$

Following Kelley and Bourbaki on filters, filter bases, and uniform structures, the $\mathsf{CauchyFilterBase}^{\uparrow}$ packet names the finite base route through which completion-facing Cauchy data is requested before a full filter object is exposed. It sits between the generic $\mathsf{FilterBase}^{\uparrow}$ interface of chapter 1601, the entourage base surface of chapter 1819, the uniform entourage structure of chapter 339, the Cauchy filter surface of chapter 333, and the entourage-driven Cauchy handoff of chapter 1791. The packet records finite base indices, entourage requests, directed refinements, Cauchy readiness, transport, replay, provenance, and local naming only; it is not a quotient filter, not a selected limit, not countable choice, and not an ambient completion principle.

Definition 1915.1 (Cauchy filter-base carrier)**.** *A* $\mathsf{CauchyFilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, U, D, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite base-index row. Each index names a displayed base packet that may be requested by a* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CauchyEntourageFilter}^{\uparrow}$*, or completion-facing consumer.*
2. *U is the entourage-request row. It records the* $\mathsf{EntourageBase}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *request data attached to each displayed base index, including the requested scale and the local admissibility read.*
3. *D is the directed-refinement row. For every two displayed base requests it names a third displayed base index, a common entourage request at a refining scale, and the two containment reads back to the original requests.*
4. *M is the Cauchy readiness row. It records the finite modulus/readiness evidence showing that every requested precision reaches an indexed base packet whose entourage request is small enough for Cauchy-filter consumption.*
5. *H records componentwise* hsame *transport for base indices, entourage requests, directed refinements, readiness rows, replay rows, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a base request through its entourage request, through the directed common refinement, through the readiness row, and then to downstream Cauchy-filter consumers.*

7. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyEntourageFilter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed route I, U, D, M, H, C, P, N. No coordinate stores a host filter, quotient filter, selected limit, countable-choice selector, global completion object, Lean verification, or bridge certificate.

Theorem 1915.2 (Cauchy filter-base NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterBase}^{\uparrow}$ *carrier*

$$K = (I, U, D, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterBase}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the displayed base-index row, entourage-request row, directed-refinement row, Cauchy readiness row, transport, replay, provenance, and local naming row;*

2. *base-to-entourage routing, requiring every indexed base request to read its entourage data from U before any Cauchy-filter or completion consumer can inspect it;*

3. *directed common refinement, requiring every pair of displayed base requests to read a third base index, a common refining entourage request, and two containment reads from D;*

4. *Cauchy readiness, requiring every precision request to pass through M and land in a displayed base index whose entourage request is small enough for the downstream Cauchy route;*

5. *transport and replay stability, requiring H and C to preserve the order I, U, D, M under componentwise* hsame *transport and displayed* Cont *replay;*

6. *provenance exhaustion, requiring P and N to name only this finite base route and to export no quotient filter, selected limit, countable-choice selector, host equality, ambient completion principle, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1915.1. The projection exposes exactly the finite base indices, entourage requests, common-refinement data, Cauchy readiness evidence, and structural rows H, C, P, N.

The base-to-entourage route is fixed by the row order. A consumer first names an index in I, then reads the requested entourage surface from U before any Cauchy-filter or completion-facing read is available.

Directed common refinement and Cauchy readiness are separate finite projections. The row D supplies the third base index and the two containment reads for each pair of requests, while M supplies the precision-facing readiness row used by Cauchy consumers.

Transport, replay, and provenance cannot enlarge the carrier. The row H compares the displayed coordinates componentwise, C replays the same dependency order, and P, N name only this packet; hence quotient filters, selected limits, choice schedules, host equality, ambient completion, Lean verification, and bridge checking are outside the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyFilterBaseUp.TasteGate.CauchyFilterBaseTasteGate single carrier alignment`

Theorem 1915.3 (Cauchy filter-base entourage refinement route)**.** *Let $K = (I, U, D, M, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterBase}^{\uparrow}$ *carrier. For any two displayed base requests $i, j \in I$ with entourage*

requests U_i and U_j, the directed-refinement row D names a displayed index $k \in I$ and an entourage request U_k such that the route

$$(i, U_i), (j, U_j) \longmapsto (k, U_k) \longmapsto H, C$$

contains the two refinement reads from U_k back to U_i and U_j. Consequently two completion-facing base requests reach a common entourage refinement through finite rows of the same carrier.

Proof. Project the two requested base rows from I and their entourage rows from U. These projections stay inside the finite carrier of Definition 1915.1.

Read the row of D attached to the pair. By definition of the directed-refinement row, it names the third index k, the common entourage request U_k, and the two containment reads back to U_i and U_j.

The structural rows preserve the displayed route. Componentwise hsame transport in H compares the requested rows with their transported copies, and Cont replay in C repeats the route from the pair of requests through the selected common refinement.

Thus the common-refinement conclusion is not an ambient directedness principle. It is the finite row D read through the displayed carrier coordinates and replayed by H, C. □

Theorem 1915.4 (Cauchy filter-base completion handoff)**.** *Every* CauchyFilter$^{\uparrow}$, CauchyEntourageFilter$^{\uparrow}$, *or completion-facing consumer of an accepted* CauchyFilterBase$^{\uparrow}$ *carrier $K = (I, U, D, M, H, C, P, N)$ reads only the finite route*

$$I, U, D, M \longmapsto H, C, P, N.$$

In particular, the consumer receives base indices, entourage requests, directed common refinements, Cauchy readiness rows, componentwise transport, replay, provenance, and local naming. It does not receive a host filter, quotient filter, selected limit, countable-choice selector, global completion object, Lean verification row, or bridge certificate.

Proof. Start with a downstream read named by P. The consumer is exposed to CauchyFilterBase$^{\uparrow}$ only through the rows listed in Definition 1915.1, so it first reads a finite base index from I and its entourage request from U.

When the consumer compares two requests, Definition 1915.3 forces the read through D. The selected common base index and its entourage request are therefore finite rows of the same packet.

When the consumer asks for Cauchy readiness, it reads M. This supplies the precision-facing base index and the small-entourage evidence before any CauchyFilter$^{\uparrow}$ or CauchyEntourageFilter$^{\uparrow}$ handoff can occur.

The remaining rows are structural. The transport row H, replay row C, provenance row P, and naming row N preserve and name the route I, U, D, M; none of them adds a host filter, quotient filter, selected limit, choice schedule, completion object, verification row, or bridge certificate. □

Closure status for NameCert$_{\text{CauchyFilterBase}^{\uparrow}}$**.**

Constructive story: A CauchyFilterBase$^{\uparrow}$ packet is generated from finite base indices, entourage requests, directed common refinements, Cauchy readiness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyFilterBase}^{\uparrow}}$, seed)

Scope closed: Definition 1915.1, Definition 1915.2, Definition 1915.3, and Definition 1915.4 seed the finite interface between FilterBase, EntourageBase, UniformSpace, CauchyFilter, and CauchyEntourageFilter routes.

Formal verification: FormalStatus(NameCert$_{\text{CauchyFilterBase}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, base-to-entourage routing, directed common refinement, Cauchy readiness, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterBaseUp.

Chapter 1916

A Concrete Naming Certificate for $\mathsf{DyadicIntervalBasis}^{\uparrow}$

Following Bishop and Bridges constructive analysis together with Kelley's basis presentation of topology, the $\mathsf{DyadicIntervalBasis}^{\uparrow}$ packet records the finite dyadic interval basis used by the L10 Real and Completion route. It sits over the endpoint-cell packet of chapter 306, the dyadic endpoint ledger of chapter 291, the rational containment surface of chapter 354, the order-topology basis reads of chapter 1890, the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite basis cells and their refinement route; it is not Dedekind completeness, open-cover compactness, quotient interval equality, host real equality, countable choice, or a selected completion point.

Definition 1916.1 (DyadicIntervalBasis carrier)**.** *A* $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, E, O, R, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint-cell row, recording the lower and upper dyadic cells exposed to a basis consumer.*
2. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger, carrying the signed dyadic endpoint rows, comparison budget, and radius tolerances read by I.*
3. *O is the* $\mathsf{RationalInterval}^{\uparrow}$ *containment row, recording the rational interval cell that contains the displayed dyadic cell and bounds its endpoint order.*
4. *R is the readback row consisting of the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational observations read through those windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached by those observations.*
5. *B is the* $\mathsf{RealOrderTopology}^{\uparrow}$ *basis-read row, exposing only the order-open basis cell generated by I, E, O, R.*
6. *H records componentwise* hsame *transport for the dyadic cell, endpoint ledger, rational containment row, readback row, basis-read row, replay, provenance, and local name.*
7. *C records displayed* Cont *replay from a basis-cell request through I, E, O, R, B and then through the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalBasis}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for Dedekind completeness, open-cover compactness, quotient interval equality, host real equality, countable choice, or a selected completion point.

Lean4 ✓ `BEDC.Derived.DyadicIntervalBasisUp.DyadicIntervalBasisTasteGate single carrier alignment`

Theorem 1916.2 (DyadicIntervalBasis NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *carrier*

$$D = (I, E, O, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic interval cell, dyadic endpoint ledger, rational containment row, finite readback row, order-topology basis row, transport, replay, provenance, and local name;*
2. *endpoint exactness, requiring each lower and upper endpoint read of I to be sourced by the dyadic ledger E and bounded by the rational containment row O;*
3. *basis generation, requiring every basis read in B to arise from the displayed dyadic cell together with the stream-window, regular-rational, and real-seal data in R;*
4. *refinement routing, requiring any accepted subcell or containment refinement to pass through the same I, E, O, R, B rows before it reaches a consumer;*
5. *structural stability, requiring H and C to preserve and replay only the displayed packet rows.*

No DyadicIntervalBasis obligation is stored outside these rows.

Proof. Project the carrier of Definition 1916.1. The visible coordinates are exactly the dyadic interval cell, endpoint ledger, rational containment row, finite readback row, order-topology basis row, transport, replay, provenance, and local naming rows.

Endpoint exactness is forced by the displayed order. The dyadic cell I has no endpoint source except the dyadic ledger E, and the rational containment row O is the only interval bound attached to those endpoints.

Basis generation then reads through R and B. The stream windows, regular-rational observations, and real seal in R are the only real-facing data available before the order-topology basis row B is exposed.

Finally, refinement routing and structural stability are exhausted by H, C, P, N. These rows can transport, replay, source, and name the displayed packet, but they add no coordinate for Dedekind completeness, open-cover compactness, quotient interval equality, host real equality, countable choice, or a selected completion point. □

Theorem 1916.3 (DyadicIntervalBasis refinement stability)**.** *Let $D = (I, E, O, R, B, H, C, P, N)$ and $D' = (I', E', O', R', B', H', C', P', N')$ be accepted* $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *carriers. If D' is obtained from D by a displayed dyadic subcell refinement whose endpoints are transported by the stored* hsame *rows and replayed through the stored* Cont *routes, then D' is accepted by the same local classifier. The transported packet preserves the* $\mathsf{DyadicInterval}^{\uparrow}$ *cell,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger,* $\mathsf{RationalInterval}^{\uparrow}$ *containment row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal rows,* $\mathsf{RealOrderTopology}^{\uparrow}$ *basis read,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$ *row.*

Proof. Start with the accepted packet D and a displayed dyadic subcell refinement. The refinement changes the interval cell only through the $\mathsf{DyadicInterval}^{\uparrow}$ coordinate I, and its endpoints remain sourced by the transported $\mathsf{DyadicRatCore}^{\uparrow}$ ledger E.

The rational containment row is transported next. Because O is the only displayed $\mathsf{RationalInterval}^{\uparrow}$ bound, the refined cell remains attached to the same containment route rather than to an unrecorded host interval.

The readback row R and basis row B then replay along the stored Cont routes. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ observations, $\mathsf{Real}^{\uparrow}$ seal rows, and $\mathsf{RealOrderTopology}^{\uparrow}$ basis read are transported componentwise, so the consumer sees the refined cell through the same finite order.

The structural rows finish the classification. The provenance and local name are transported with the packet, and the carrier has no extra coordinate from which a quotient interval equality, host real equality, open-cover compactness theorem, Dedekind completeness principle, countable-choice witness, or selected completion point could be read. □

Theorem 1916.4 (DyadicIntervalBasis refinement cover)**.** *Let $D = (I, E, O, R, B, H, C, P, N)$ be an accepted $\mathsf{DyadicIntervalBasis}^{\uparrow}$ carrier. Any finite family of displayed dyadic subcell refinements whose endpoint rows are transported through H and replayed through C assembles into finite basis-cover data over the same $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealOrderTopology}^{\uparrow}$ rows. Each refined cell contributes only a transported endpoint cell, endpoint ledger, rational containment row, finite readback row, Real seal, order-topology basis read, provenance, and local name; the assembled cover exports no Dedekind completeness, open-cover compactness, quotient interval equality, host real equality, countable choice, or selected completion point.*

Proof. Induct on the finite refinement family. The empty family assembles to the empty finite basis-cover packet over the displayed carrier rows, with no new mathematical coordinate.

For the successor step, assume the preceding refinements have already been assembled as finite basis-cover data over $I, E, O, R, B, H, C, P, N$. Apply Definition 1916.3 to the next displayed subcell refinement. Its endpoints are transported by H, its route is replayed by C, and its accepted packet preserves the dyadic interval cell, endpoint ledger, rational containment row, finite readback row, Real seal rows, and order-topology basis read.

Append that accepted subcell packet to the finite cover list. The provenance and local naming rows P, N source the appended cell in the same carrier surface, so the assembled cover remains finite and basis-facing. Since every induction step uses only the transported carrier rows, none of the excluded host principles can enter the assembled data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalBasis}^{\uparrow}$ packet is generated from a DyadicInterval endpoint cell, a DyadicRatCore endpoint ledger, a RationalInterval containment row, StreamName finite windows, RegSeqRat readbacks, RealUp seal rows, a RealOrderTopology basis read, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$, seed)

Scope closed: Definition 1916.1, Definition 1916.2, Definition 1916.3, and Definition 1916.4 seed the finite dyadic basis-cell route over dyadic endpoint cells, endpoint ledgers, rational containment, finite readbacks, Real seals, and order-topology basis reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, open-cover compactness, quotient interval equality, host real equality, countable choice, a selected completion point, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, basis generation, refinement routing, and structural stability for BEDC.Derived.DyadicIntervalBasisUp.

Chapter 1917

A Concrete Naming Certificate for $\mathsf{DoubleLimitInterchange}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DoubleLimitInterchange}^{\uparrow}$ packet records a finite certificate for exchanging two Cauchy limit routes through displayed windows rather than through an ambient completeness principle. Its dependency surface is the $\mathsf{StreamName}^{\uparrow}$ schedule interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the shared dyadic tolerance rows of chapter 291, the diagonal-route discipline of chapter 1800, the uniform-window route of chapter 1749, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records a boundary for exchanging two displayed completion-facing reads; it does not claim selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1917.1 (Double-limit interchange carrier)**.** *A* $\mathsf{DoubleLimitInterchange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, J, D, \Delta, R_0, R_1, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the outer* $\mathsf{StreamName}^{\uparrow}$ *schedule, listing the finite outer windows inspected by a completion-facing consumer.*
2. *J is the inner* $\mathsf{StreamName}^{\uparrow}$ *schedule, listing the finite inner windows read under each displayed outer window.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic request, the two Cauchy-window budgets, and the comparison margin used by both routes.*
4. *Δ is the diagonal selector row supplied through the* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ *route, choosing a displayed common window only from O, J, D.*
5. *R_0 is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, obtained by reading the outer schedule before the inner schedule and then applying the shared dyadic ledger.*
6. *R_1 is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, obtained by reading the inner schedule before the outer schedule and then applying the same shared dyadic ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the common displayed Cauchy-window read.*

8. *H records componentwise* hsame *transport for the outer schedule, inner schedule, dyadic ledger, diagonal selector, first readback, second readback, real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from any requested precision through $O, J, D, \Delta, R_0, R_1, E$.*
10. *P is the* Pkg *provenance over* $\mathsf{DoubleLimitInterchange}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyDiagonalLimit}^{\uparrow}$, $\mathsf{UniformLimitInterchange}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DoubleLimitInterchange}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DoubleLimitInterchangeUp.DoubleLimitInterchangeTasteGate single carrier alignment`

Theorem 1917.2 (Double-limit interchange NameCert obligations)**.** *For every accepted* $\mathsf{DoubleLimitInterchange}^{\uparrow}$ *carrier*

$$I = (O, J, D, \Delta, R_0, R_1, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DoubleLimitInterchange}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed outer schedule, inner schedule, shared dyadic tolerance ledger, diagonal selector, first readback, second readback, real seal, transport, continuation, provenance, and local naming row;*
2. *shared-window routing, requiring every route comparison to expose O, J, D before either* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row is available;*
3. *diagonal routing, requiring the selector Δ to be read through the* $\mathsf{CauchyDiagonalLimit}^{\uparrow}$ *packet before any common real seal can be consumed;*
4. *uniform interchange routing, requiring both readback directions R_0 and R_1 to use the same dyadic tolerance ledger and the same displayed windows, as in the* $\mathsf{UniformLimitInterchange}^{\uparrow}$ *route;*
5. *structural stability, requiring H and C to transport and replay only the displayed rows, with provenance supplied by P and local naming supplied by N.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1917.1. The displayed coordinates are exactly the two schedules, the shared dyadic tolerance ledger, the diagonal selector, the two regular readbacks, the real seal, the structural transport and replay rows, provenance, and the local name.

The route comparison first exposes the schedules and the shared tolerance. Since both readback rows are downstream of O, J, D, a consumer cannot replace one side by a separately chosen window or tolerance.

The diagonal selector supplies the common Cauchy window. It is read through the same finite diagonal route used by $\mathsf{CauchyDiagonalLimit}^{\uparrow}$, so it cannot introduce a selected limit or a quotient-stream representative.

The interchange step then reads R_0 and R_1 through the uniform-window discipline of $\mathsf{UniformLimitInterchange}^{\uparrow}$. The two orders of reading are admitted only as two routes through the same displayed rows.

Finally H, C, P, N transport, replay, source, and name that finite surface. These structural rows add no analytic coordinate, so the listed obligations exhaust the local certificate. □

Theorem 1917.3 (Double-limit interchange diagonal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{DoubleLimitInterchange}^{\uparrow}$ *carrier factors through the finite route*

$$O, J, D \longmapsto \Delta \longmapsto (R_0, R_1) \longmapsto E$$

and then through the stored hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. The two orders of Cauchy-window readback therefore meet at a displayed diagonal selector and a shared dyadic tolerance ledger; the route exports no selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a real-facing precision request. Carrier admission gives the request only the outer schedule O, the inner schedule J, and the shared dyadic tolerance ledger D.

The diagonal row Δ is the next available common-window object. It is computed from the displayed schedules and tolerance ledger, matching the diagonal route supplied by chapter 1800.

The two readbacks R_0 and R_1 are then read through the same finite window data. The first uses the outer-then-inner order and the second uses the inner-then-outer order, but both consume the same O, J, D, Δ surface.

The terminal seal E is downstream of those readbacks. After E, the rows H, C, P, N only transport, replay, source, and name the route. Since no excluded analytic object occurs in the carrier inventory, the real-facing read cannot leave the displayed path. □

Theorem 1917.4 (Double-limit interchange nonescape)**.** *Every public read of an accepted* $\mathsf{DoubleLimitInterchange}^{\uparrow}$ *carrier is exhausted by the rows* $O, J, D, \Delta, R_0, R_1, E, H, C, P, N$*. In particular, the packet cannot export selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier coordinates. The analytic rows are the two stream schedules, the shared dyadic tolerance ledger, the diagonal selector, the two regular readbacks, and the terminal real seal.

The remaining rows are structural. Transport H preserves componentwise classifier data, continuation C replays displayed precision routes, provenance P records the sibling sources, and N supplies the local certificate row.

There is no coordinate left for selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient completeness. Therefore every public read is confined to the finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{DoubleLimitInterchange}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DoubleLimitInterchange}^{\uparrow}$ packet is generated from an outer StreamName schedule, an inner StreamName schedule, a shared DyadicRatCore tolerance ledger, a CauchyDiagonalLimit selector, two RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DoubleLimitInterchange}^{\uparrow}}$, seed)

Scope closed: Definition 1917.1, Definition 1917.2, Definition 1917.3, and Definition 1917.4 seed the double-limit interchange packet over stream schedules, regular readbacks, dyadic tolerance rows, diagonal selection, uniform interchange routing, and RealUp seal descent.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DoubleLimitInterchange}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, host function equality, quotient streams, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window routing, diagonal routing, uniform interchange routing, structural stability, and nonescape for BEDC.Derived.DoubleLimitInterchangeUp.

Chapter 1918

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ packet records the finite request-to-tail-window realizer by which a regular Cauchy modulus becomes a displayed regular rational read before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the source decoding of chapter 1771, the Real-facing threshold route of chapter 1423, the generic Cauchy-modulus schedule of chapter 292, the finite window interface of chapter 126, the regular rational readback boundary of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies the L10 Real and Completion route with one finite witness object for turning a precision request into the tail window it reads; it is not Real completeness, quotient stream equality, host real equality, countable choice, a selected limit, trichotomy, or a standard bridge.

Definition 1918.1 (Regular Cauchy modulus realizer carrier)**.** *A* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, M, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the precision request row, recording the finite Real or Completion demand that must be answered by a tail-window read.*
2. *M is the Cauchy-modulus threshold row imported from the Real-facing modulus route and the generic modulus schedule.*
3. *W is the selected finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window row determined by A and M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested radius, slack comparisons, and dyadic arithmetic consumed by the selected window.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the window W under the tolerance ledger D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the request, threshold, window, tolerance, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the request, modulus threshold, selected window, tolerance ledger, readback, seal, replay, provenance, and local name.*

8. *C records displayed* Cont *replay from the precision request through A, M, W, D, Q, E and then through H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$, $\mathsf{CauchyRealizer}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, M, W, D, Q, E, H, C, P, N$. It has no coordinate for Real completeness, quotient stream equality, host real equality, countable choice, a selected limit, trichotomy, open-cover compactness, Lean verification, or a standard bridge.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusRealizerUp.RegularCauchyModulusRealizerTasteGate`
`single carrier alignment`

Theorem 1918.2 (Regular Cauchy modulus realizer NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carrier*

$$R = (A, M, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the precision request A, Cauchy-modulus threshold M, selected tail window W, dyadic tolerance ledger D, regular rational readback Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N;*
2. *request admission, requiring the Real or Completion precision request to enter through A before any threshold or window is read;*
3. *threshold realization, requiring the threshold row M to be the only Cauchy-modulus source used to select the tail window;*
4. *tail-window selection, requiring W to be the finite* $\mathsf{StreamName}^{\uparrow}$ *row selected by the pair A, M;*
5. *tolerance accounting, requiring D to record exactly the dyadic radius and slack comparisons consumed by W, Q, E;*
6. *readback and seal routing, requiring Q and E to be downstream of the displayed A, M, W, D route;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *non-escape for Real completeness, quotient stream equality, host real equality, countable choice, selected limits, trichotomy, open-cover compactness, Lean verification, and standard-bridge data.*

No regular Cauchy modulus realizer obligation is stored outside these rows.

Proof. Project the carrier of Definition 1918.1. The precision request A, threshold row M, tail window W, tolerance ledger D, regular readback Q, and terminal seal E are the only analytic coordinates, while H, C, P, N are the only structural coordinates.

The request-to-window order is forced by the displayed tuple. A consumer first reads A, then the threshold row M, and only then the finite $\mathsf{StreamName}^{\uparrow}$ window W selected for the requested tolerance.

The tolerance and seal rows are downstream. The ledger D supplies the dyadic radius and slack comparisons used by the regular readback Q, and the seal E is reached after those rows have been displayed.

The structural rows transport, replay, source, and name the same route. Since none of the excluded objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1918.3 (Regular Cauchy modulus realizer tail realization)**.** *Every Real or Completion precision request accepted by a* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carrier factors through the finite route*

$$A \longmapsto M \longmapsto W \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed request row, a Cauchy-modulus threshold, one selected finite $\mathsf{StreamName}^{\uparrow}$ *tail window, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming. The factorization supplies no ambient completeness principle, quotient stream equality, host equality, countable-choice selector, selected limit, trichotomy row, open-cover compactness theorem, Lean evidence, or standard bridge.*

Proof. Start with an accepted carrier and inspect its precision request row. The row A is the only entrance for the requested tolerance, so a consumer cannot first read a detached stream window or terminal Real seal.

The threshold row is read next. The row M is the only Cauchy-modulus coordinate in the carrier, hence it is the only source from which the tail boundary for the request can be obtained.

Once A and M are fixed, the selected window W is the finite stream row consumed by the readback. The dyadic ledger D then records the tolerance arithmetic for that window, and Q is the regular rational observation obtained from the same displayed data.

Finally the seal E and structural rows H, C, P, N close the finite route. They preserve and name the displayed path, but they introduce none of the excluded completeness, quotient, choice, order, verification, or bridge data. □

Theorem 1918.4 (Regular Cauchy modulus realizer stream handoff)**.** *Every accepted* RegularCauchyModulusRealiz *read that reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite window, selected tail schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, and componentwise* hsame *and* Cont *replay rows before the Real seal row can be consumed.*

Proof. Project the carrier of Definition 1918.1. The stream-facing window is the row W, the dyadic budget is D, and the structural replay is carried by H and C.

Use the tail realization of Definition 1918.3. It fixes the selected tail window by the displayed route from the precision request row A through the Cauchy-modulus threshold row M before any regular readback or terminal seal is visible.

Now apply request transport determinacy, Definition 1918.5. Under componentwise hsame transport and displayed Cont replay, the same request-threshold route keeps the same selected window, tolerance ledger, regular readback, and seal-facing path.

Thus any downstream $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives the finite window, selected tail schedule, dyadic tolerance budget, and replay rows before reading the Real seal. The carrier contains no selected-limit, quotient-stream, host-real-equality, choice, or ambient-completeness coordinate that could bypass this stream handoff. □

Theorem 1918.5 (Regular Cauchy modulus realizer request transport determinacy)**.** *Let*

$$R = (A, M, W, D, Q, E, H, C, P, N)$$

and

$$R' = (A', M', W', D', Q', E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carriers. Suppose the precision request rows* A, A' *and Cauchy-modulus threshold rows* M, M' *are carried to one another by the displayed componentwise* hsame *rows in* H, H'*, and suppose the displayed* Cont *replays in* C, C' *read the same request-to-threshold route. Then the selected* $\mathsf{StreamName}^{\uparrow}$ *tail-window rows* W, W'*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D, D'*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* Q, Q'*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows* E, E' *are the same classifier-facing finite route. The determinacy result exports no selected limit, quotient stream equality, host real equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Proof. Project the two carrier presentations of Definition 1918.1. In each accepted packet the only entrance for a precision demand is the request row, and the only threshold source downstream of it is the Cauchy-modulus row.

Use the ordering supplied by Definition 1918.3. Both displayed replays read the route from A to M and from A' to M' before any window, ledger, readback, or seal row is exposed. The hypotheses identify the request and threshold rows through the componentwise hsame data in H, H'.

After that identification, the accepted carrier has a single downstream window coordinate for the fixed request-threshold pair: W in R and W' in R'. The dyadic tolerance ledger is then the single row downstream of the selected window, so the classifier-facing ledger route is D, D'.

The readback and seal rows are forced in the same way. The only $\mathsf{RegSeqRat}^{\uparrow}$ readback downstream of the ledger is Q, Q', and the only terminal $\mathsf{Real}^{\uparrow}$ seal downstream of that readback is E, E'.

Finally apply the non-escape boundary of Definition 1918.7. Since neither carrier contains any coordinate for selected limits, quotient stream equality, host real equality, countable choice, trichotomy, ambient completeness, Lean evidence, or a standard bridge, the transported request can select only the finite route through W, D, Q, E and its counterpart W', D', Q', E'. □

Theorem 1918.6 (Regular Cauchy modulus realizer selector budget exhaustion)**.** *Let*

$$R = (A, M, W, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carrier. Any* $\mathsf{RealObservationBudget}^{\uparrow}$ *consumer or* $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ *consumer serving the same precision request factors through*

$$A \longmapsto M \longmapsto W \longmapsto D \longmapsto Q \longmapsto E$$

and then through the structural rows H, C, P, N *of* R*. The selector receives no independent stream representative, hidden schedule, selected limit, quotient equality, countable-choice row, trichotomy row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum. Thus the budget consumers of chapter 611 and chapter 782 can consume the same finite request-to-tail-window route without importing data outside the accepted packet.*

Proof. Start from the tail-realization factorization in Definition 1918.3. It already places every accepted Real or Completion precision request on the displayed route through $A, M, W, D, Q, E, H, C, P, N$,

so a budget consumer reading the request has a finite carrier-facing spine before it reads any terminal seal.

Apply the request-transport determinacy of Definition 1918.5. If a $\mathsf{RealObservationBudget}^{\uparrow}$ or $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$ consumer serves the same request through the same displayed threshold replay, the selected window, tolerance ledger, readback, and terminal seal are the same classifier-facing downstream route.

It remains to exclude extra budget data. The carrier definition of Definition 1918.1 has no coordinate for an independent stream representative, hidden schedule, selected limit, quotient equality, countable-choice row, trichotomy row, or ambient completeness datum. The sibling budget consumers therefore receive only the finite route exposed by R and its structural rows. □

Theorem 1918.7 (Regular Cauchy modulus realizer Real seal nonescape)**.** *Let $R = (A, M, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *carrier. Any downstream use of the terminal* $\mathsf{Real}^{\uparrow}$ *seal E must pass through the precision request row, Cauchy-modulus threshold, selected finite tail window, dyadic tolerance ledger, regular rational readback, and structural rows of the same carrier. Thus the seal cannot be used as a hidden endpoint for Real completeness, quotient stream equality, host real equality, countable choice, a selected limit, trichotomy, open-cover compactness, Lean verification, or a standard bridge.*

Proof. Use the tail-realization route of Definition 1918.3. It places E after A, M, W, D, Q, so every accepted seal read is attached to the displayed request, threshold, window, tolerance, and readback rows.

Now inspect the classifier surface of Definition 1918.1. The carrier has no second Real-seal coordinate and no coordinate for completeness, quotient equality, host equality, choice, selected limits, trichotomy, open-cover compactness, Lean evidence, or bridge data.

Therefore a downstream read that bypasses one of $A, M, W, D, Q, H, C, P, N$ is not a read of the accepted carrier. The only available terminal route is the finite route named by R. □

Falsifiable prediction: Every $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ precision read factors through a request row, Cauchy-modulus threshold, finite StreamName tail window, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-realizer, realcauchymodulus, cauchymodulus, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries request admission, threshold realization, finite tail-window selection, dyadic tolerance accounting, regular readback, terminal seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ packet is generated from a precision request row, a Cauchy-modulus threshold row, a selected finite StreamName tail window, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealizer}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 1918.2, Definition 1918.3, Definition 1918.5, Definition 1918.6, and Definition 1918.7 give the finite request-to-tail-window obligation witness set for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusRealizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, countable choice, selected limits, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, request admission, threshold realization, tail-window selection, tolerance accounting, readback and seal routing, tail realization, and Real-seal non-escape for BEDC.Derived.RegularCauchyM

Chapter 1919

A Concrete Naming Certificate for $\mathsf{StoneWeierstrass}^{\uparrow}$

Following Stone's Stone–Weierstrass theorem, the $\mathsf{StoneWeierstrass}^{\uparrow}$ packet records the finite separating-algebra route by which compact real approximation data may be read through displayed algebra, compactness, continuity, and real-name surfaces. It sits over the compact carrier of chapter 60, the continuous-map surface of chapter 130, the real-name boundary of chapter 38, the ring operations of chapter 43, and the polynomial approximation consumer of chapter 1634. The packet names a finite BEDC certificate for separation, constants, algebraic closure, approximation handoff, and transport only; it does not claim nonconstructive compactness, choice-selected dense subsets, host function equality, or ambient completeness.

Definition 1919.1 (Stone–Weierstrass carrier)**.** *A* $\mathsf{StoneWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, A, U, V, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through the* $\mathsf{Compact}^{\uparrow}$ *interface and its finite cover or finite-net witnesses.*
2. *A is the algebra row, exposing only finitely named* $\mathsf{ContinuousMap}^{\uparrow}$ *generators over K together with displayed addition, multiplication, scalar, and constant-one operations routed through* $\mathsf{Ring}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*
3. *U is the unit and constants ledger, recording the visible constant real functions admitted into the algebra row.*
4. *V is the separation ledger, assigning each displayed apartness request between named compact points a generator in A and a finite real-window comparison that separates the two inspected points.*
5. *R is the real-valued evaluation ledger, reading every generator, algebraic combination, and constant through the same* $\mathsf{Real}^{\uparrow}$ *windows used by the compact and continuous-map surfaces.*
6. *M is the compact-continuity modulus row, recording the finite continuity and compactness data needed before approximation is requested.*
7. *E is the approximation handoff row, sending an inspected continuous target, a rational error row, and the displayed algebra data to a* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *consumer.*

8. *H stores componentwise* hsame *transport for the compact source, algebra, constants, separation, evaluation, modulus, approximation, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a compact-point request through algebra evaluation, separation comparison, compact-continuity modulus, and approximation handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{StoneWeierstrass}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{WeierstrassApproximation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for nonconstructive compactness, choice-selected dense subsets, host function equality, an ambient function-space quotient, a host supremum norm, or ambient completeness.

Lean4 (definition) `BEDC.Derived.StoneWeierstrassUp.StoneWeierstrassUp`

Theorem 1919.2 (Stone–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{StoneWeierstrass}^{\uparrow}$ *carrier*

$$S = (K, A, U, V, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the compact source, algebra, constants, separation, real-valued evaluation, compact-continuity modulus, approximation handoff, transport, replay, provenance, and local naming rows;*
2. *compact-source compatibility, requiring every point and finite witness consumed by the algebra and separation rows to pass through the same* $\mathsf{Compact}^{\uparrow}$ *row K;*
3. *algebra closure visibility, requiring A and U to display the continuous generators, constants, and finite algebraic operations used by the packet rather than a host algebra of all functions;*
4. *separation exactness, requiring each separation request in V to name the generator, the two compact-point windows, and the real-window apartness comparison used to distinguish them;*
5. *evaluation and modulus routing, requiring R and M to mediate every real-valued read before any approximation handoff is exposed;*
6. *approximation handoff exactness, requiring E to send the inspected continuous target and rational error through the displayed algebra data before it reaches* $\mathsf{WeierstrassApproximation}^{\uparrow}$*;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *ledger refusal for nonconstructive compactness, choice-selected dense subsets, host function equality, ambient function-space quotients, host supremum norms, ambient completeness, Lean verification, and bridge checking.*

No Stone–Weierstrass obligation is stored outside these rows.

Proof. Project the carrier of Definition 1919.1. The packet exposes exactly the compact source, algebra, constants, separation, evaluation, modulus, approximation, transport, replay, provenance, and local naming rows.

The compact-source and algebra clauses are then fixed by row membership. The row K is the only compact input surface, while A and U are the only places where generators, constants, and algebraic operations are named.

For a separation request, the row V supplies a displayed generator and the corresponding real-window comparison. The evaluation row R routes that comparison through $\mathsf{Real}^{\uparrow}$, and M supplies the compact-continuity information needed before a downstream approximation read is allowed.

The approximation handoff is the row E. It sends the inspected continuous target and rational error to the $\mathsf{WeierstrassApproximation}^{\uparrow}$ consumer only after the algebra, separation, evaluation, and modulus rows have been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded objects occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 1919.3 (Stone–Weierstrass separating-algebra route)**.** *Every approximation-facing consumer of an accepted* $\mathsf{StoneWeierstrass}^{\uparrow}$ *carrier factors through the finite route*

$$K \longmapsto A, U, V \longmapsto R, M \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer reads compactness, continuous-map generators, real constants, separating witnesses, real-window evaluations, compact-continuity moduli, approximation handoff data, transport, replay, provenance, and local naming. The route does not provide nonconstructive compactness, choice-selected dense subsets, host function equality, ambient function-space quotients, host supremum norms, ambient completeness, Lean verification, or a standard bridge.

Proof. Begin with an approximation-facing read of the accepted carrier. The source point and compact witness data first enter through K, so the consumer has no direct route to an ambient compactness principle.

The algebraic data are then read through A and U. Separation requests pass through V, which names a generator and a finite real-window comparison for the displayed compact-point windows.

Next the route reaches R and M. These rows ensure that each function value and compact-continuity read is mediated by the displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{Compact}^{\uparrow}$ surfaces.

Only after those rows are present can E hand the inspected target and rational error to the $\mathsf{WeierstrassApproximation}^{\uparrow}$ consumer. The structural rows H, C, P, N preserve and name the same route, so the excluded compactness, choice, equality, quotient, norm, completeness, verification, and bridge data cannot be recovered from the packet. □

Definition 1919.4 (Constructive finite Stone–Weierstrass carrier)**.** *Following Bishop and Bridges, the constructive finite route inside a* $\mathsf{StoneWeierstrass}^{\uparrow}$ *packet is the subpacket*

$$S_{\mathrm{fin}} = (K, A_{\mathrm{fin}}, U_{\mathrm{fin}}, V_{\mathrm{fin}}, R_{\mathrm{fin}}, M_{\mathrm{fin}}, E_{\mathrm{fin}}, H, C, P, N)$$

whose algebra, constants, separation, real-window, modulus, and approximation rows are all read as finite displayed ledgers. The compact source K is still the source of chapter 60; the continuous generators in A_{fin} are still read through chapter 130; and the approximation handoff E_{fin} remains a consumer of chapter 1634.

The classifier for this subpacket compares only the displayed finite rows. It has no coordinate for a host dense subalgebra, an externally completed function space, a selected family of all separating functions, pointwise function equality outside the named windows, or a host supremum norm.

Theorem 1919.5 (Constructive Stone–Weierstrass NameCert obligations)**.** *For every accepted constructive finite subpacket S_{fin} of Definition 1919.4, the local* $\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}$ *obligations are exactly the finite compact-source row, the finite continuous-generator ledger, the finite constants ledger, the displayed separating-witness ledger, the real-window evaluation ledger, the compact-continuity modulus ledger, the WeierstrassApproximation handoff row, and the structural H, C, P, N rows.*

Proof. Project S_{fin} to the carrier of Definition 1919.1. Each row of the subpacket is one of the rows already named by the ambient $\mathsf{StoneWeierstrass}^{\uparrow}$ certificate, with the algebra, constants, separation, evaluation, modulus, and handoff rows restricted to finite displayed ledgers.

Apply Definition 1919.2. Its compact-source, algebra-closure, separation-exactness, evaluation, modulus, and handoff clauses are row-local, so restricting those rows to finite ledgers does not create any additional obligation.

The structural rows H, C, P, N are unchanged. They transport, replay, source, and name the same finite route, and the refusal clause continues to exclude host dense algebras, external function-space equality, selected infinite families, host norms, and ambient completeness. □

Theorem 1919.6 (Constructive separating-algebra handoff)**.** *Every approximation consumer of an accepted constructive finite subpacket factors through*

$$K \longmapsto A_{\mathrm{fin}}, U_{\mathrm{fin}}, V_{\mathrm{fin}} \longmapsto R_{\mathrm{fin}}, M_{\mathrm{fin}} \longmapsto E_{\mathrm{fin}} \longmapsto H, C, P, N.$$

Thus the consumer receives only compact source data, finitely displayed continuous generators, constants, separating witnesses, real-window evaluations, compact-continuity moduli, WeierstrassApproximation handoff data, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Start with Definition 1919.3. It already forces every approximation-facing read through $K, A, U, V, R, M, E, H, C, P, N$ before any handoff can be consumed.

Now use the finite subpacket of Definition 1919.4. The rows A, U, V, R, M, E are read as $A_{\mathrm{fin}}, U_{\mathrm{fin}}, V_{\mathrm{fin}}, R_{\mathrm{fin}}, M_{\mathrm{fin}}, E_{\mathrm{fin}}$, so the same route passes through finite displayed ledgers only.

Finally apply Definition 1919.5. The obligation surface names no host dense subalgebra, external function-space equality, selected infinite separating family, host supremum norm, or ambient completeness row; therefore the handoff is exactly the constructive finite separating-algebra route. □

Falsifiable prediction: Every accepted $\mathsf{StoneWeierstrass}^{\uparrow}$ approximation-facing read factors through compact source rows, finite continuous-map algebra generators, constants, separating witnesses, real-window evaluations, compact-continuity moduli, WeierstrassApproximation handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact, continuousmap, real, ring, weierstrass-approximation: $\mathsf{StoneWeierstrass}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact source data, a finite separating algebra of continuous real-valued generators, constant and algebraic closure rows, real-window evaluation, compact-continuity routing, approximation handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StoneWeierstrass}^{\uparrow}$ packet is generated from a compact source row, finite continuous-map algebra generators, constants, separating witnesses, real-window evaluations, compact-continuity moduli, WeierstrassApproximation handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1919.1, Definition 1919.4, Definition 1919.2, Definition 1919.5, Definition 1919.3, and Definition 1919.6 seed the finite separating-algebra route over compact, continuous-map, real, ring, and WeierstrassApproximation surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close nonconstructive compactness, choice-selected dense subsets, host function equality, ambient function-space quotients, host supremum norms, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source compatibility, algebra closure visibility, separation exactness, evaluation and modulus routing, approximation handoff exactness, and separating-algebra route proofs for BEDC.Derived.StoneWeierstrassUp.

Chapter 1920

A Concrete Naming Certificate for $\mathsf{CauchyApproximationSystem}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{CauchyApproximationSystem}^{\uparrow}$ packet records the finite approximation system that turns displayed dyadic observation windows into regular Cauchy-name evidence. It sits between the dyadic endpoint ledger of chapter 291, the finite stream-window source of chapter 126, the regular rational handoff of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the nested dyadic Cauchy route of chapter 1811. The packet packages finite dyadic observation windows into regular Cauchy-name evidence before any consumer may read a completed real-name boundary.

Definition 1920.1 (Cauchy approximation-system carrier)**.** *A* $\mathsf{CauchyApproximationSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{sys}} = (S, W, A, R, M, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, carrying the finite stream schedule whose observations may be opened by the packet.*
2. *W is the unary finite-window row, assigning each displayed precision request to an inspected initial window of S.*
3. *A is the dyadic approximant row, recording for each inspected window the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximant read from that window.*
4. *R is the refinement row, recording displayed inclusions from a smaller unary window to a larger unary window together with the dyadic comparison budget used to keep the approximants coherent.*
5. *M is the modulus budget, assigning every displayed Cauchy precision to a unary window after which all later accepted windows refine through R.*
6. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationSystem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedDyadicCauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationSystem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, W, A, R, M, P, N. It contains no coordinate for quotient real equality, choice-selected limits, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationSystemUp.CauchyApproximationSystemUpTasteGate single carrier alignment`

Theorem 1920.2 (Cauchy approximation-system window refinement)**.** *Let*

$$A_{\mathrm{sys}} = (S, W, A, R, M, P, N)$$

be an accepted $\mathsf{CauchyApproximationSystem}^{\uparrow}$ *carrier. If a unary window* w' *extends a displayed unary window* w *and the extension is recorded by the refinement row* R*, then the larger-window packet is accepted by the same local classifier. Its stream source remains* S*, its dyadic approximants are the transported rows of* A*, its Cauchy budget is still governed by* M*, and its provenance and local name remain* P *and* N*.*

Proof. Project the carrier of Definition 1920.1. The only window data are the source row S, the unary finite-window row W, the dyadic approximant row A, the refinement row R, and the modulus budget M.

Use the displayed extension $w \subseteq w'$ through R. The refinement row is part of the carrier, so the larger window is not chosen by an external selector; it is one of the finite windows whose comparison against the old window is already recorded.

The dyadic approximants transport along the same refinement row. Since every approximant in A is sourced by the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger and every later Cauchy request is bounded by M, enlarging the window preserves the displayed approximation budget.

Finally P and N source and name the same finite packet. No quotient real equality, choice-selected limit, open-cover compactness, trichotomy, or ambient completeness coordinate is introduced by the larger window, so the local classifier accepts the transported packet. □

Theorem 1920.3 (Cauchy approximation-system regular-name route)**.** *Every coherent family of accepted* $\mathsf{CauchyApproximationSystem}^{\uparrow}$ *window refinements supplies the regular-name route consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, a consumer that reads finite windows through* S, W*, dyadic approximants through* A*, refinement stability through* R*, and Cauchy budgets through* M *factors through the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with the nested-window compatibility supplied by chapter 1811. The route exports no quotient real equality, selected limit, trichotomy, open-cover compactness, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a coherent family of accepted window refinements. Each member exposes the same stream source S, the inspected unary windows W, dyadic approximants A, refinement rows R, and modulus budget M.

Apply Definition 1920.2 along the refinement family. Larger windows preserve the same source, approximant, budget, provenance, and local naming rows, so the family gives one stable finite observation route rather than unrelated window reads.

The stable route is exactly the input shape required by the $\mathsf{RegSeqRat}^{\uparrow}$ handoff. The StreamName surface supplies finite observations, the DyadicRatCore surface supplies dyadic approximants and comparison budgets, and the refinement rows supply the Cauchy coherence needed before regular rational readback.

Finally the $\mathsf{Real}^{\uparrow}$ seal is consumed only after that regular handoff. The nested-window compatibility is the sibling route of chapter 1811, so the completed read remains a finite BEDC route and cannot export quotient real equality, selected limits, trichotomy, open-cover compactness, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationSystem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyApproximationSystem}^{\uparrow}$ packet is generated from a StreamName source row, unary finite-window rows, DyadicRatCore approximant rows, displayed refinement rows, a modulus budget, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSystem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1920.1, Definition 1920.2, and Definition 1920.3 seed the finite approximation-system route from stream windows and dyadic approximants to RegSeqRat readback and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationSystem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, choice-selected limits, open-cover compactness, trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window refinement stability, coherent regular-name routing, and nonescape for BEDC.Derived.CauchyApproximationSystemUp.

Chapter 1921

A Concrete Naming Certificate for $\mathsf{CauchyCompletionRetraction}^{\uparrow}$

Following Bishop and Bridges constructive completion and the Bourbaki–Kelley language of reflective completion, the $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ packet records the finite BEDC route by which the unit and counit rows form a displayed split for the L10 Cauchy-completion boundary. It binds the unit admission of chapter 1527, the returned read of chapter 1539, the universal route of chapter 1456, the idempotent completion surface of **??**, and the triangle route of chapter 1795. The chapter is a finite NameCert split packet only; it does not introduce quotient completions, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge-axis evidence.

Definition 1921.1 (CauchyCompletionRetraction carrier)**.** *A* $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, U, V, A, I, G, S, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed source row whose completion-facing admission is being split.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route admitting X into the completion-facing surface.*
3. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *route returning already displayed completion reads to the terminal real-seal boundary.*
4. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *universal route that compares unit-admitted consumers with completion-facing consumers over the same finite source.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *row recording that a repeated completion read contributes no new public completion row.*
6. *G is the* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *row comparing the unit-then-counit route with the direct displayed real route.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read before the split can expose any rational observation.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the stream windows and by the split comparison.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row consumed by the terminal seal.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which the retracted read is observed.*

11. *H records componentwise* hsame *transport for $X, U, V, A, I, G, S, D, R, E$ and the structural rows.*
12. *C records displayed* Cont *replay for consumers that inspect the unit side, the counit side, or the split comparison.*
13. *P is the* Pkg *provenance over* CauchyCompletionRetraction$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionCounit$^{\uparrow}$, CauchyCompletionAdjunction$^{\uparrow}$, CauchyCompletionIdempotent$^{\uparrow}$, CauchyCompletionTriangleIdentity$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* NameCert$_{\text{CauchyCompletionRetraction}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected completion representatives, countable-choice schedules, host equality of completed objects, ambient completeness, or public bridge credentials.

Lean4 ✓ `BEDC.Derived.CauchyCompletionRetractionUp.CauchyCompletionRetractionTasteGate single carrier alignment`

Theorem 1921.2 (CauchyCompletionRetraction NameCert obligations)**.** *For an accepted* CauchyCompletionRetrac *carrier $T = (X, U, V, A, I, G, S, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionRetraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source row, unit row, counit row, adjunction row, idempotent row, triangle row, finite-window row, dyadic ledger, regular readback, terminal seal, transport, replay, provenance, and local naming rows;*
2. *unit-side exposure, requiring every completion-facing read exported from X to pass through U before any completion consumer can inspect it;*
3. *counit-side return, requiring every already displayed completion-facing read to pass through V before reaching S, D, R, E;*
4. *split exactness, requiring the composite through U, V, A, I, G to expose exactly the same displayed route to E as the direct finite route through S, D, R;*
5. *classifier stability, requiring transported packets with the same displayed rows to be identified by the carried* hsame *and* Cont *data;*
6. *ledger non-escape, stating that the packet exports no quotient-completion equality, selected representative, countable-choice schedule, host equality, ambient-completeness theorem,* Real$^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Carrier admission is projection from Definition 1921.1. The tuple lists exactly the source, unit, counit, adjunction, idempotent, triangle, finite-window, dyadic, readback, seal, transport, replay, provenance, and local naming rows.

The unit-side exposure is read from U. A completion-facing consumer starting at X receives no direct completion object; the carrier first displays the route by which X enters the completion-facing surface.

The counit-side return is read from V. Once a completion-facing read is already displayed, the packet forces the return route through the finite observation rows S, D, R before the terminal seal E can be consumed.

Split exactness is the shared comparison supplied by A, I, G. The adjunction row fixes the universal consumer route, the idempotent row states that a repeated completion read contributes no new public row, and the triangle row compares the unit-then-counit route with the direct route to the same seal.

Classifier stability is componentwise transport along H followed by replay through C. Since the carrier fields are exhausted by $X, U, V, A, I, G, S, D, R, E, H, C, P, N$, the excluded quotient, representative-choice, countable-choice, host-equality, ambient-completeness, real-completeness, and bridge data have no row from which a consumer could read them. □

Theorem 1921.3 (CauchyCompletionRetraction unit-counit split exactness)**.** *Let $T = (X, U, V, A, I, G, S, D, R, E$ be an accepted* $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ *carrier. Every consumer of the completion-facing split factors through the displayed finite route*

$$X \longmapsto U \longmapsto A, I, G \longmapsto V \longmapsto S, D, R \longmapsto E \longmapsto H, C, P, N.$$

After this route is projected, the visible seal-facing read is the same direct route

$$S, D, R \longmapsto E$$

that the source row already displays. The split is therefore a unit-counit retraction over finite BEDC rows, not equality of quotient completions, not a selected representative, not host equality, and not countable choice or ambient completeness.

Proof. Start with a consumer of the split packet and project the source row X. The only completion-facing admission exposed by the carrier is U, so the consumer first passes through the unit row.

The middle comparison is fixed by A, I, G. The adjunction row A records that the completion-facing consumer is a consumer of the same reflected route, the idempotent row I prevents a second completion read from adding a new public row, and the triangle row G identifies the unit-counit composite with the displayed route to the seal.

The return part is then forced by V. The counit row routes the already displayed completion-facing read through the finite stream window S, the dyadic tolerance ledger D, and the regular rational readback R before the terminal seal E is inspected.

Transport by H and replay by C preserve this route for equivalent packets, while P and N record provenance and local naming. No other carrier coordinate exists, so the split exports exactly the displayed finite route and none of the excluded quotient, representative, host-equality, choice, or completeness data. □

Theorem 1921.4 (CauchyCompletionRetraction nonescape)**.** *For an accepted* $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ *carrier $T = (X, U, V, A, I, G, S, D, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing, or completion-facing read exported by T is generated by the rows named in Definition 1921.1. In particular, the packet exports no quotient-completion equality, selected completion representative, countable-choice schedule, host equality of completed objects, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Project the accepted carrier. A $\mathsf{Real}^{\uparrow}$-facing or $\mathsf{RegSeqRat}^{\uparrow}$-facing read must pass through S, D, R, E, because these are the only finite observation and terminal seal rows in the tuple.

A completion-facing read must pass through U, V, A, I, G. These are the only completion-route rows in the tuple, and each is stored as displayed NameCert data rather than as a quotient point or selected representative.

The structural rows H, C, P, N transport, replay, source, and name the same route. They add no carrier coordinate beyond the listed rows.

Thus every exported read is generated by Definition 1921.1. The forbidden quotient, representative, choice, host-equality, ambient-completeness, real-completeness, and bridge data are absent from the carrier and cannot be produced by projection, transport, replay, provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRetraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ packet is generated from a displayed source row, CauchyCompletionUnit route, CauchyCompletionCounit route, CauchyCompletionAdjunction route, CauchyCompletionIdempotent row, CauchyCompletionTriangleIdentity row, StreamName finite windows, DyadicRatCore tolerances, RegSeqRat readback, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRetraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1921.1, Definition 1921.2, Definition 1921.3, and Definition 1921.4 seed the finite unit-counit split route for the L10 Cauchy-completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRetraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected completion representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, counit return, split exactness, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionRetractionUp.

Chapter 1922

A Concrete Naming Certificate for $\mathsf{AbelLimitTheorem}^{\uparrow}$

Following Abel and Hardy's boundary-limit theorem in Bishop–Bridges constructive analysis, the $\mathsf{AbelLimitTheorem}^{\uparrow}$ packet records the finite NameCert surface by which an Abel-summed family approaches an endpoint real when the source sequence or series window has a displayed limit. Its source transform reads the $\mathsf{AbelSummation}^{\uparrow}$ route of chapter 1763 and the Abel-test boundary of chapter 1496; its endpoint row uses the real sequence limit surface, the uniform approach row of chapter 463, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a constructive boundary handoff for displayed finite rows; it is not a host power-series semantics, a selected limit, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1922.1 (AbelLimitTheorem carrier)**.** *An* $\mathsf{AbelLimitTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, S, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, displaying the finite Abel transform or summation-by-parts window read by the boundary argument.*
2. *S is the source sequence or series-window row, displaying the inspected source values together with the finite tails addressed by A.*
3. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row for the endpoint limit of the source side.*
4. *U is the* $\mathsf{UniformLimitContinuity}^{\uparrow}$ *row, or an equivalent uniform-window row, governing approach to the endpoint along the admitted Abel parameter windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff obtained from A, S, L, U.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal attached to that regular rational handoff.*
7. *H records componentwise* hsame *transport for Abel-summation rows, source windows, endpoint-limit rows, uniform approach rows, regular handoff, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an endpoint precision request through U to finite parameter windows, through A, S, L to the regular handoff R, and then to the seal row E.*

9. *P is the* Pkg *provenance over* $\mathsf{AbelLimitTheorem}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{UniformLimitContinuity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for host power-series semantics, selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AbelLimitTheoremUp.AbelLimitTheoremTasteGate single carrier alignment`

Theorem 1922.2 (AbelLimitTheorem NameCert obligations)**.** *For every accepted* $\mathsf{AbelLimitTheorem}^{\uparrow}$ *carrier* $Q = (A, S, L, U, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *Abel-window exposure for A, preserving the finite summation row consumed by the endpoint argument;*
3. *source-window exposure for S, preserving the inspected sequence or series values and the finite tails read by A;*
4. *endpoint-limit exposure for L, preserving the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row used to name the source-side limit;*
5. *uniform endpoint approach for U, forcing every precision request through a displayed finite Abel-parameter window;*
6. *regular handoff exactness for R, E, stating that the terminal real seal is reached only through the displayed Abel, source, limit, and uniform rows;*
7. *transport stability, continuation replay, provenance, and local naming for exactly the route recorded by H, C, P, N.*

These obligations are finite-row obligations over the Abel summation, endpoint limit, uniform-window, regular rational, and real seal surfaces. They do not assert an ambient analytic power-series object or a completed host-limit principle.

Proof. Project the accepted packet $Q = (A, S, L, U, R, E, H, C, P, N)$. The projection exposes exactly the Abel-summation row, source row, endpoint-limit row, uniform-window row, regular handoff, real seal, transport rows, continuation replay, provenance, and local naming row.

Read a displayed endpoint precision request through U. The uniform row supplies a finite Abel-parameter window, so the downstream read cannot select a limiting parameter outside the packet.

Now combine the analytic rows. The Abel row A displays the finite transform over the source window S, while L names the endpoint limit of the same source side. The regular handoff R may therefore read only the finite Abel transform, source window, endpoint-limit row, and uniform approach row.

Finally apply the structural rows. Componentwise transport in H preserves those rows together, while C, P, N replay, source, and name the same route. Since the carrier has no further coordinate, the seven listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}$ surface. □

Theorem 1922.3 (AbelLimitTheorem real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{AbelLimitTheorem}^{\uparrow}$ *carrier factors through*

$$A, S, L, U \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the terminal real seal may read the boundary-limit claim only after reading the finite Abel row, the source sequence or series window, the endpoint real-sequence limit row, and the uniform approach row. No accepted consumer can import host power-series semantics, selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Begin with an accepted carrier and a $\mathsf{Real}^{\uparrow}$ read. By the carrier definition, the only seal-facing rows are R, E, H, C, P, N, and R is downstream of the displayed analytic rows A, S, L, U.

Trace R backward through the packet. The uniform row U translates a seal precision request to a finite endpoint window, the Abel row A supplies the finite transform on that window, the source row S supplies the inspected values and tails, and the limit row L supplies the named endpoint target.

The terminal seal E therefore receives only the regular rational handoff built from those four displayed rows. The structural rows preserve and replay that route componentwise; none of them contains a coordinate for a host power series, a selected limiting process, quotient equality, countable choice, trichotomy, or ambient completeness.

Hence every real-seal consumer factors through the displayed chain and cannot bypass the finite $\mathsf{AbelLimitTheorem}^{\uparrow}$ certificate. □

Falsifiable prediction: An accepted $\mathsf{AbelLimitTheorem}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal without reading the $\mathsf{AbelSummation}^{\uparrow}$ row, source sequence or series-window row, $\mathsf{RealSequenceLimit}^{\uparrow}$ endpoint row, and $\mathsf{UniformLimitContinuity}^{\uparrow}$ endpoint-window row refutes the carrier surface.

Independence witness: abel-summation, abel-test, real-sequence-limit, uniform-limit-continuity, regseqrat, real: no listed sibling supplies the whole ten-row packet $Q = (A, S, L, U, R, E, H, C, P, N)$, because each sibling contributes only a finite Abel transform, convergence-test boundary, endpoint limit, uniform approach, regular readback, or seal face while this chapter binds them into one terminal boundary-limit NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelLimitTheorem}^{\uparrow}$ packet is generated from an $\mathsf{AbelSummation}^{\uparrow}$ row, a displayed source sequence or series-window row, a $\mathsf{RealSequenceLimit}^{\uparrow}$ endpoint-limit row, a $\mathsf{UniformLimitContinuity}^{\uparrow}$ endpoint-window row, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1922.1, Definition 1922.2, and Definition 1922.3 seed the finite Abel boundary-limit route over $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{UniformLimitContinuity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelLimitTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host power-series semantics, selected limits, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface and real-seal handoff theorem.

Chapter 1923

A Concrete Naming Certificate for $\mathsf{SchauderFixedPoint}^{\uparrow}$

Following Schauder's fixed point theorem in the compact-convex tradition, the $\mathsf{SchauderFixedPoint}^{\uparrow}$ packet records only the finite BEDC route that a compact convex self-map certificate must expose. It carries compactness, convexity, continuous-map, approximation-net, residual, transport, replay, provenance, and local NameCert rows, and it does not assert a host fixed-point choice, a quotient of approximating nets, ambient Real completeness, or an uncarried topological vector-space theorem.

The route is anchored by the carried completeness and compactness interfaces of chapter 128 and chapter 127, by the interval fixed-point and contraction comparison surfaces of chapter 1437 and chapter 396, and by the convex and continuous-map interfaces of chapter 208 and chapter 130. The packet is therefore a finite naming certificate for the compact-convex handoff rather than a semantic proof object for Schauder's theorem in an ambient analysis library.

Definition 1923.1 (SchauderFixedPoint carrier)**.** *A* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, Q, V, A, F, E, R, L, H, C, P, N)$$

with the following visible rows:

1. *K is the compact-metric support row, read through the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *surfaces, including the finite-net coverage rows used by later approximation reads;*
2. *Q is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the carried subset, exposing only affine-combination closure, scalar classifier transport, and membership rows;*
3. *V is the ambient vector and metric compatibility row that lets compact-net centers, convex combinations, and residual estimates be compared inside the same carried metric context;*
4. *A is the approximation-net ledger, recording finite centers, rational tolerances, barycentric or finite-affine cells, and the finite projection route from compact coverage into the convex carrier;*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row, including graph, pointwise modulus, source membership, and target membership data for the compact convex carrier;*
6. *E is the approximate-endpoint ledger, exposing for each displayed tolerance a finite cell where the approximation route and the continuous-map route meet;*
7. *R is the residual row that compares the displayed point, its image under F, and the finite-affine approximation cell through the metric and convexity ledgers;*

8. *L is the limiting handoff row, using only the carried complete-metric and total-boundedness data to retain a candidate name boundary for downstream readers;*
9. *H stores componentwise* hsame *transports for K, Q, V, A, F, E, R, L and the local certificate row;*
10. *C stores the displayed* Cont *replay routes for compact-net reads, convex-affine reads, continuous-map reads, residual comparison, limiting handoff, and consumer export;*
11. *P is the* Pkg *provenance over* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SchauderFixedPoint}^{\uparrow}}$ *registration row.*

The carrier predicate says that these rows form one finite compact-convex fixed-point certificate for the displayed self-map. It has no coordinate for a chosen fixed point, an open-cover compactness principle, a quotient limit, a host topological vector space, or ambient completeness beyond the carried rows.

Lean4 ✓ `BEDC.Derived.SchauderFixedPointUp.SchauderFixedPointTasteGate single carrier alignment`

Theorem 1923.2 (SchauderFixedPoint NameCert obligations)**.** *For an accepted packet $S = (K, Q, V, A, F, E, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SchauderFixedPoint}^{\uparrow}}$ *obligations are exhausted by the carrier rows of Definition 1923.1. They consist of compact-net coverage, carried completeness handoff, convex membership and affine-combination stability, continuous-map graph and modulus replay, approximation-net coherence, residual comparison, limiting handoff, componentwise classifier transport, displayed continuation replay, package provenance, and local naming. No obligation may call on host fixed-point choice, open-cover compactness, ambient Real completeness, quotient equality of nets, uncarried topological vector-space structure, or a global Schauder theorem outside the packet.*

Proof. Project the accepted packet. Carrier admission gives exactly the rows $K, Q, V, A, F, E, R, L, H, C, P, N$ listed in Definition 1923.1.

The compact and convex obligations are separated but share the same carried metric context. The compact row K supplies finite coverage and carried completeness, while Q supplies membership and affine-combination closure inside the displayed convex carrier. The compatibility row V is the only route through which those two surfaces are read together.

The map and approximation obligations then consume F and A. The continuous-map row supplies graph and modulus replay, and the approximation-net ledger supplies the finite centers and affine cells on which the residual row R is computed. The limiting handoff L is available only after those rows have been replayed.

Finally H, C, P, and N transport, replay, record provenance, and register the same finite route. Since no coordinate outside this route is present, the excluded host, quotient, compactness, Real-completeness, and global fixed-point principles cannot enter the local NameCert surface. □

Theorem 1923.3 (SchauderFixedPoint compact-convex handoff)**.** *Every downstream read of an accepted* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *packet factors through the displayed compact-convex route*

$$K, Q, V, A, F, E, R, L, H, C, P, N.$$

In particular, a consumer can inspect a candidate boundary only after the $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net coverage, the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff, the* $\mathsf{ConvexSet}^{\uparrow}$ *affine-membership rows, and the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus rows have all been replayed through the approximation and residual ledgers.*

Proof. Start with a consumer of the accepted packet. The only consumer-facing route is the continuation ledger C, and its compact reads are attached to K, so the finite-net and carried-completeness surfaces are read before any candidate boundary appears.

The convex data are read next through Q and V. These rows align finite centers, affine cells, and carrier membership, so the approximation ledger cannot export a point outside the displayed compact-convex context.

The continuous self-map row F and the approximation ledger A then feed the approximate-endpoint and residual rows E and R. The limiting handoff L is therefore a boundary of the displayed residual route, not a selected host fixed point.

The transport, replay, provenance, and local naming rows keep this order intact. Hence every downstream read factors through the compact-convex handoff surface named in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderFixedPoint}^{\uparrow}$ packet is generated from compact-metric finite-net coverage, carried completeness, convex membership and affine-combination rows, ambient vector-metric compatibility, continuous-map graph and modulus data, approximation-net cells, residual comparison, limiting handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1923.1, Definition 1923.2, and Definition 1923.3 seed the finite compact-convex fixed-point naming route over the displayed compactness, completeness, convexity, continuous-map, approximation, residual, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host fixed-point choice, open-cover compactness, ambient Real completeness, quotient equality of approximating nets, a topological vector-space theorem outside the carried rows, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, convex-affine membership stability, continuous-map replay, approximation-net coherence, residual comparison, limiting handoff, and local NameCert coverage for the displayed packet.

Chapter 1924

A Concrete Naming Certificate for $\mathsf{NestedIntervalReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{NestedIntervalReal}^{\uparrow}$ packet records the classical nested located interval route to a real name as finite BEDC data. It sits after the located interval carrier of chapter 1709, the bisection route of chapter 1711, the dyadic refinement surface of chapter 291, the nested dyadic Cauchy route of chapter 1811, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names only the displayed interval-nest route; it is not open-cover compactness, countable choice, quotient real equality, selected-limit data, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1924.1 (NestedIntervalReal carrier)**.** *A* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, B, D, E, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite family of nested* $\mathsf{LocatedRealInterval}^{\uparrow}$ *rows whose endpoints, radii, locators, and metric brackets are visible to the packet.*
2. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement ledger, recording how each displayed interval row narrows to the next row.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter-decay ledger, assigning to each displayed stage a dyadic bound whose carrier proves that later intervals have diameter at most the requested tolerance.*
4. *E records the endpoint* $\mathsf{Real}^{\uparrow}$ *seals for all displayed interval endpoints and states that endpoint reads are available only through the interval rows in I.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the midpoint, left-endpoint, right-endpoint, and diameter observations used by the nest.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W, D, E, so every rational approximation is sourced by a displayed window and a dyadic diameter bound.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the nested interval route after I, B, D, E, W, R have been replayed.*
8. *H records componentwise* hsame *transport for the interval rows, bisection ledger, diameter ledger, endpoint seals, stream windows, regular readback, terminal seal, replay, provenance, and local naming data.*

9. *C records displayed* Cont *replay for consumers that inspect the interval nest, the dyadic diameter decay, the regular readback, or the terminal seal.*
10. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, D, E, W, R, S, H, C, P, N$. It has no coordinate for arbitrary open covers, a countable selector, a quotient construction of reals, selected limiting points, host real equality, or a proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.NestedIntervalRealUp.TasteGate.NestedIntervalRealTasteGate single carrier alignment`

Theorem 1924.2 (NestedIntervalReal NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier*

$$J = (I, B, D, E, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval rows, bisection ledger, dyadic diameter-decay ledger, endpoint seals, stream windows, regular readback, terminal seal, transports, replay, provenance, and local naming;*
2. *nesting exactness, requiring every later interval read to factor through the previous interval row and the displayed* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement in B;*
3. *diameter decay, requiring every requested tolerance to be witnessed by a dyadic bound in D and by endpoint seals in E before the regular readback row is exposed;*
4. *readback ordering, requiring* $\mathsf{StreamName}^{\uparrow}$ *windows in W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R to be replayed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal S is available;*
5. *classifier stability, requiring transported packets to compare only through the stored* hsame *and* Cont *rows;*
6. *ledger non-escape, requiring P, N to source and name no open-cover compactness, countable choice, selected limit, host real equality, quotient real construction, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No nested-interval real obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1924.1. The accepted packet displays exactly one interval-nest family, one bisection ledger, one dyadic diameter ledger, one endpoint-seal row, one finite window family, one regular readback row, one terminal real seal, and the structural rows that transport, replay, package, and name them.

Nesting exactness is read from I and B. The only way to pass from one interval stage to the next is the displayed bisection row, so a consumer cannot substitute an external interval selection rule for the stored refinement ledger.

Diameter decay is read from D together with the endpoint seals E. For a requested tolerance, the packet first displays the dyadic bound and then opens only the endpoint reads attached to the corresponding interval rows.

The readback route is downstream of those rows. Stream windows in W expose finite observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row R packages them as regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal S is reached only after that readback is replayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no open-cover, choice, selected-limit, host-equality, quotient-construction, or real-completeness coordinate, none of those objects can enter the local NameCert surface. □

Theorem 1924.3 (NestedIntervalReal Cauchy selection route)**.** *For every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier* $J = (I, B, D, E, W, R, S, H, C, P, N)$ *and every displayed precision request, the exported rational approximation is valid only through the staged finite obligation*

$$D_{\mathrm{diam}} \longrightarrow B_{\mathrm{path}} \longrightarrow E_{\mathrm{end}} \longrightarrow W_{\mathrm{win}} \longrightarrow R_{\mathrm{reg}} \longrightarrow S.$$

That is, the packet must first exhibit a stage row whose dyadic diameter entry in D is below the requested tolerance, then exhibit through B the bisection path from the earlier interval rows in I to that stage, then read the endpoint seals through E, then open only the corresponding finite $\mathsf{StreamName}^{\uparrow}$ *windows in W, then package the resulting rational approximation by the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and only then expose the terminal* $\mathsf{Real}^{\uparrow}$*-facing seal S. The approximation is therefore a displayed regular-readback product of the carrier rows, not a point chosen from an ambient intersection.*

Proof. Project the accepted carrier and fix a displayed precision request. The diameter ledger D supplies the tolerance witness: a finite stage row whose dyadic diameter entry is below the requested bound.

The stage is reachable only through the stored interval data. The rows in I display the located interval family, and the bisection ledger B records the finite path from earlier intervals to the chosen stage, so the construction has no separate selection rule for later intervals.

Endpoint access is then ordered through E. The candidate rational approximation is read from endpoint seals attached to the same displayed stage, not from a host real point or an unrecorded intersection element.

Regular readback begins only after those seals have been fixed. The finite observations pass through the matching $\mathsf{StreamName}^{\uparrow}$ windows in W, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R packages the resulting rational read as the Cauchy-facing approximation.

Finally, S seals exactly that read for the $\mathsf{Real}^{\uparrow}$-facing consumer. The transport, replay, provenance, and local naming rows H, C, P, N preserve the staged route under comparison and replay, so no ambient intersection point, choice-selected subsequence, quotient representative, or independent completeness theorem is used to obtain the approximation. □

Theorem 1924.4 (NestedIntervalReal tail-modulus readback)**.** *For every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier*

$$J = (I, B, D, E, W, R, S, H, C, P, N)$$

and every displayed precision request whose dyadic diameter witness is read from D, the request determines a finite tail-modulus readback route

$$D \longrightarrow W \longrightarrow R \longrightarrow S.$$

The diameter row supplies the requested tolerance bound, the matching $\mathsf{StreamName}^{\uparrow}$ *windows in W expose only the finite observations at that bound, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R packages those observations as the regular rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal S consumes exactly that readback. Thus the modulus exported to the regular rational surface is a displayed BEDC obligation of J, not an independent completeness or intersection principle.*

Proof. Project the accepted carrier J. By Definition 1924.3, a displayed precision request is already ordered by the staged finite route from the diameter ledger to the terminal seal. In particular, the dyadic entry in D is the row that fixes the requested tail bound before any regular readback is exposed.

Once that bound is fixed, the only observations available at the bound are the corresponding finite $\mathsf{StreamName}^{\uparrow}$ windows in W. The carrier definition in Definition 1924.1 gives no coordinate for an independent selector, ambient intersection point, or external modulus source, so the request can only continue through the rows already displayed by J.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R then repackages exactly those finite window observations as the regular rational readback. The terminal $\mathsf{Real}^{\uparrow}$ seal S is downstream of R in the carrier and in the staged route, so it consumes that readback rather than a separately chosen real point. Therefore the tail-modulus readback is the finite BEDC obligation

$$D \longrightarrow W \longrightarrow R \longrightarrow S$$

inside the packet. □

Theorem 1924.5 (NestedIntervalReal diameter modulus)**.** *For every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier*

$$J = (I, B, D, E, W, R, S, H, C, P, N),$$

each displayed precision request whose witness lies in the dyadic diameter ledger D yields a finite diameter-modulus row by the route

$$D \longrightarrow B \longrightarrow E \longrightarrow W \longrightarrow R \longrightarrow S.$$

The request first checks the nested bisection path in B, then the endpoint seals in E, then the matching finite windows in W, then the regular readback row R, and finally the terminal $\mathsf{Real}^{\uparrow}$ *seal S. The row is a regular Cauchy modulus export for RegSeqRat and Real consumers, not a selected intersection point, open-cover compactness principle, quotient real equality, countable-choice selector, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the precision request and its witness in D. The Cauchy selection route of Definition 1924.3 already orders such a request through the displayed diameter witness, the bisection path, the endpoint seals, the finite windows, regular readback, and terminal seal.

The tail-modulus readback of Definition 1924.4 then identifies the downstream handoff from the diameter ledger to W, R, S. Combining the two routes inserts the bisection ledger B and endpoint seals E before the regular readback is exposed.

Thus the extracted modulus row is finite and carrier-local. The structural rows H, C, P, N transport, replay, source, and name the same route, while Definition 1924.9 excludes selected intersection points, open-cover compactness, quotient real equality, countable choice, host equality, and Real completeness. □

Theorem 1924.6 (NestedIntervalReal Cauchy seal)**.** *For every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier*

$$J = (I, B, D, E, W, R, S, H, C, P, N),$$

the exported $\mathsf{Real}^{\uparrow}$*-facing seal is available only by the finite route*

$$I, B, D, E \longrightarrow W \longrightarrow R \longrightarrow S.$$

The nested interval rows and dyadic diameter ledger supply the Cauchy modulus, the $\mathsf{StreamName}^{\uparrow}$ *windows expose finite observations at the requested bound, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row packages the regular rational readback, and* $\mathsf{Real}^{\uparrow}$ *consumes exactly that seal. Thus the route composes the nested interval packet into a regular Cauchy real-name handoff, not into ambient completeness.*

Proof. Start from the carrier of Definition 1924.1. The rows I, B, D, E display the located interval family, the bisection refinement ledger, the dyadic diameter-decay ledger, and the endpoint seals.

The Cauchy-stage choice is already constrained by Definition 1924.3. A precision request first reads a dyadic diameter witness in D, then follows the refinement path in B, and then reads only the endpoint seals in E attached to the corresponding interval rows in I.

The tail-modulus route of Definition 1924.4 supplies the downstream handoff. Once the bound is fixed, the $\mathsf{StreamName}^{\uparrow}$ windows in W expose the finite observations at that bound, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R packages them as regular rational readback.

The terminal seal S consumes exactly that readback. By Definition 1924.9, the carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, quotient real equality, or a selected intersection point, so the theorem exports only the displayed Cauchy seal. □

Theorem 1924.7 (NestedIntervalReal located-limit uniqueness)**.** *Let* $J = (I, B, D, E, W, R, S, H, C, P, N)$ *and* $J' = (I', B', D', E', W', R', S', H', C', P', N')$ *be accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carriers. If* I *and* I' *are componentwise related by* hsame*, the bisection ledgers* B *and* B' *display the same refinement path, and the dyadic diameter ledgers* D *and* D' *contain cofinal witnesses for the same precision requests, then the terminal* $\mathsf{Real}^{\uparrow}$ *seals* S *and* S' *are compared by the Real classifier through the same route*

$$W \longrightarrow R \longrightarrow S.$$

The claim is route uniqueness for the located-limit seal inside the displayed nested-interval data. It is not open-cover compactness, countable choice, quotient real equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 1924.4 for both accepted carriers. The shared hsame interval rows, common bisection ledger, and cofinal dyadic diameter witnesses force the same displayed tail-modulus request to be read from D and D'.

Apply Definition 1924.5. The common diameter request routes through the same bisection path and endpoint-seal ordering before any regular rational readback is exposed.

Now apply Definition 1924.6. Both packets must pass through the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is available, so the classifier sees the same $W \longrightarrow R \longrightarrow S$ pattern.

Componentwise hsame transport in the structural rows carries the comparison from interval rows to windows, readback, and terminal seals. Since Definition 1924.9 excludes open-cover compactness, countable choice, quotient real equality, host equality, and ambient completeness, the conclusion is only located route uniqueness for the displayed Real-facing seal. □

Theorem 1924.8 (NestedIntervalReal scoped kernel route)**.** *Every accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *Cauchy seal factors through the displayed nested interval row, dyadic diameter modulus, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}$ *row. In carrier notation* $J = (I, B, D, E, W, R, S, H, C, P, N)$*, every Real-facing or RegSeqRat-facing read of the seal is routed by*

$$I, B, D, E \longrightarrow W \longrightarrow R \longrightarrow S \longrightarrow H, C, P, N.$$

The route exports no quotient stream equality, selected limit, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, or open-cover compactness principle.*

Proof. Project the accepted carrier and read the Cauchy seal through Definition 1924.6. That theorem already orders the terminal $\mathsf{Real}^{\uparrow}$-facing seal through the nested interval rows, endpoint seals, finite windows, regular rational readback, and terminal seal.

The finite stage used by the seal is supplied by the Cauchy selection route of Definition 1924.3. The selected precision request is therefore not an ambient selected limit; it is a displayed row of the interval nest and bisection ledger.

The requested bound is controlled by Definition 1924.4 and Definition 1924.5. These statements force the dyadic diameter modulus through D, then through the matching finite windows W and the readback row R before S is available.

Finally apply ledger nonescape, Definition 1924.9. The structural rows H, C, P, N transport, replay, source, and name the same route, while the excluded quotient, selector, trichotomy, choice, completeness, host-equality, and compactness objects are absent from the packet. □

Theorem 1924.9 (NestedIntervalReal ledger nonescape)**.** *For an accepted* $\mathsf{NestedIntervalReal}^{\uparrow}$ *carrier* $J = (I, B, D, E, W, R, S, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing, or interval-facing read exported by* J *is generated by the rows named in Definition 1924.1. In particular, the packet exports no open-cover compactness principle, countable-choice selector, selected limit, host real equality, quotient construction of reals, or proof of* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. An interval-facing read must pass through I, B, D, E, because these are the only rows that contain located interval data, bisection refinement, dyadic diameter decay, and endpoint seals.

A $\mathsf{RegSeqRat}^{\uparrow}$-facing read must pass through W and R. These are the only finite window and regular readback rows available to the packet.

A $\mathsf{Real}^{\uparrow}$-facing read must then pass through S, and any comparison of two accepted packets must use H and C before P and N can record provenance and local naming.

Thus every exported read is generated by the displayed ledger. The excluded compactness, choice, selected-limit, host-equality, quotient-construction, and completeness objects are absent from the carrier and cannot be produced by projection, transport, replay, provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NestedIntervalReal}^{\uparrow}$ packet is generated from nested located interval rows, a located-bisection refinement ledger, dyadic diameter-decay rows, endpoint $\mathsf{Real}^{\uparrow}$ seals, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1924.1, Definition 1924.2, Definition 1924.3, Definition 1924.4, and Definition 1924.9 seed the finite nested located interval route from diameter-decaying intervals to regular rational readback and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected limits, host real equality, quotient construction, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nesting exactness, diameter decay, endpoint-seal ordering, Cauchy selection routing, classifier stability, and ledger non-escape for BEDC.Derived.NestedIntervalRealUp.

Chapter 1925

A Concrete Naming Certificate for $\mathsf{HausdorffSpace}^{\uparrow}$

Following Hausdorff and Bourbaki/Kelley general topology, the $\mathsf{HausdorffSpace}^{\uparrow}$ packet records separated points by disjoint open-neighborhood rows over $\mathsf{Topology}^{\uparrow}$. When a $\mathsf{Met}^{\uparrow}$ row supplies a positive-distance separation witness, the packet stores the resulting open-neighborhood handoff as finite NameCert data rather than as host point equality. The chapter sits between the topology surface of chapter 88, the metric surface of chapter 57, and the separated completion consumer of chapter 402.

Definition 1925.1 (Hausdorff space carrier)**.** *A* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, including the displayed open predicates, finite neighborhood ledger, and classifier route read by topological consumers.*
2. *x, y are the two point-name rows compared by the separatedness witness. They are point observations inside the same topology source row, not host-level representatives.*
3. *U, V are the open-neighborhood rows, with U containing the displayed name x and V containing the displayed name y.*
4. *D is the disjointness ledger certifying that the two open rows have no shared displayed point observation under the local topology classifier.*
5. *M is the optional* $\mathsf{Met}^{\uparrow}$ *distance-separation handoff row. When present, it records the metric route that generates U, V, D from a positive displayed distance.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row used by metric-generated separation reads, exposing only the displayed distance and positivity evidence available through the real-name interface.*
7. *H records componentwise* hsame *transport for $T, x, y, U, V, D, M, E, C, P, N$.*
8. *C records displayed* Cont *replay from a point-pair comparison through the neighborhood and disjointness route*

$$T, x, y \longmapsto U, V, D \longmapsto M, E.$$

9. *P is the* Pkg *provenance over* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares separated point pairs only through these rows. It contains no coordinate for quotient point equality, open-cover compactness, selected representatives, ambient completeness, or standard bridge data.

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceTasteGate single carrier alignment`

Theorem 1925.2 (HausdorffSpace NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, point pair, open-neighborhood rows, disjointness ledger, optional metric handoff, real seal, transport, replay, provenance, and local naming rows;*
2. *disjoint open-neighborhood separation, requiring U and V to be displayed as open rows in T, to contain x and y respectively, and to be separated by D;*
3. *classifier transport, requiring every transported separatedness read to preserve T, x, y, U, V, D through the componentwise* hsame *row H;*
4. *metric handoff, requiring any* $\mathsf{Met}^{\uparrow}$ *distance-apartness read to pass through M and the* $\mathsf{Real}^{\uparrow}$ *seal row E before it yields U, V, D;*
5. *nonescape, requiring downstream topology, metric, real, and completion consumers to read only $T, x, y, U, V, D, M, E, H, C, P, N$.*

No HausdorffSpace obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceNameCert obligations`

Proof. Carrier admission is projection from Definition 1925.1. The accepted tuple displays exactly the topology source, point pair, open-neighborhood rows, disjointness ledger, optional metric handoff, real seal, and structural rows.

Disjoint open-neighborhood separation is the role of U, V, D. The rows U and V are declared open inside the topology source T, the point-name rows x, y are read through those neighborhoods, and D records their disjointness under the local classifier.

Classifier transport is componentwise. The row H transports the same topology source, point pair, neighborhood rows, and disjointness ledger, while C replays the displayed route from the point-pair comparison to the separated-neighborhood witness.

Metric handoff uses only M and E. A metric-generated read first exposes the positive displayed distance through the $\mathsf{Real}^{\uparrow}$ seal, then records the open-neighborhood separation route in M. It never replaces the separated-neighborhood data by host equality or host apartness of points.

Finally inspect P, N. Provenance and local naming mention only the displayed rows, so topology, metric, real, and completion consumers have no coordinate from which to read quotient point equality, open-cover compactness, selected representatives, ambient completeness, or standard bridge closure. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier namecert obligations`

Theorem 1925.3 (HausdorffSpace metric separation handoff)**.** *Let*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier. If the optional* $\mathsf{Met}^{\uparrow}$ *row* M *is present, every displayed positive-distance apartness witness in* M *factors through the* $\mathsf{Real}^{\uparrow}$ *seal row* E *and yields the same open-neighborhood rows* U, V *with disjointness ledger* D*. Thus metric distance-apartness contributes a Hausdorff separation row without importing host equality, quotient point identity, selected representatives, or ambient completeness.*

Proof. Read the metric-generated branch of the carrier. The row M is attached to the same point-name rows x, y and the same topology source T as the displayed neighborhoods.

The distance value is visible only through the $\mathsf{Real}^{\uparrow}$ seal E. Positivity therefore supplies a finite separation budget for the neighborhood construction, not an external equality or apartness relation on host representatives.

The handoff then lands in U, V, D. These rows are already topology rows in the carrier: U and V are open neighborhoods of x and y, and D is their disjointness ledger. Componentwise transport and replay through H, C preserve this factorization for downstream consumers. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier metric separation handoff`

Theorem 1925.4 (HausdorffSpace classifier separation route)**.** *Let*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier. Every separated-point classifier read generated from the optional metric row* M *factors through the* $\mathsf{Met}^{\uparrow}$ *distance witness, the* $\mathsf{Real}^{\uparrow}$ *positive-distance seal* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ *row* N *before it yields the open-neighborhood rows* U, V *and disjointness ledger* D*. Thus metric classifier separation is a finite NameCert route over the displayed carrier rows, not a host apartness predicate on points.*

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier classifier separation route`

Proof. Start with a metric-generated classifier read. The only metric coordinate in the carrier is M, and it is attached to the same topology source T and point-name rows x, y as the displayed neighborhoods.

The distance value and positivity evidence are exposed only through the $\mathsf{Real}^{\uparrow}$ seal E. The metric route therefore supplies a finite positive-distance budget, after which the packet returns to the topological output rows U, V, D.

Transport and replay are restricted to H and C. Provenance and local naming are restricted to P and N, so the resulting classifier read has no access to host point apartness, selected representatives, quotient equality, or ambient completeness. □

Theorem 1925.5 (HausdorffSpace diagonal nonescape)**.** *For an accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N),$$

the diagonal equality boundary of the packet is exhausted by the displayed separation rows T, x, y, U, V, D *together with optional metric and real handoff rows* M, E*. A downstream topology, metric, or completion consumer cannot replace that boundary by host point equality, quotient topology equality, real trichotomy, open-cover compactness, selected representatives, or ambient completeness.*

Proof. Inspect the carrier. The point names x, y occur only as displayed rows inside the topology source T, and the separation output is the pair of open-neighborhood rows U, V together with the disjointness ledger D.

When the optional metric route is present, it passes through M and the real seal E before returning to the same topological rows. This adds a displayed positive-distance source for separation, but it does not add host equality, quotient equality, trichotomy, compactness, representatives, or completeness coordinates.

The structural rows H, C, P, N preserve and name only the displayed packet. Hence every consumer remains inside the finite diagonal boundary described by the carrier. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier diagonal nonescape`

Theorem 1925.6 (HausdorffSpace choice-free point separation)**.** *Let*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier. If the optional metric row M displays a positive-distance witness through the* $\mathsf{Real}^{\uparrow}$ *seal E, then the carrier produces the open-neighborhood rows U, V and the disjointness ledger D as the separatedness output. This construction does not select representatives, quotient points, host point equalities, or ambient completeness data.*

Proof. Start from the carrier of Definition 1925.1. The rows x, y are displayed point names inside the topology source T, while U, V, D are already the open-neighborhood and disjointness rows of the same packet.

Use the NameCert obligations of Definition 1925.2. The metric handoff obligation requires every metric-generated read to pass through M and the $\mathsf{Real}^{\uparrow}$ seal E before it yields U, V, D.

Apply the metric separation handoff of Definition 1925.3. A positive displayed distance supplies only the finite separation budget for the recorded open neighborhoods. Since the carrier has no coordinate for representatives, quotient point identity, host equality, or ambient completeness, none of those objects is selected or consumed. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier point separation choicefree`

Theorem 1925.7 (HausdorffSpace open-neighborhood classifier transport)**.** *Let*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

be an accepted $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier. Transporting any accepted separated-neighborhood read along the componentwise row H preserves the topology source T, point rows x, y, open-neighborhood rows U, V, disjointness ledger D, optional metric row M, and optional* $\mathsf{Real}^{\uparrow}$ *seal row E. Consequently the transported classifier still reads separation through the same displayed open-neighborhood and disjointness rows, and it cannot replace the read by quotient point equality, selected representatives, host point apartness, or ambient completeness.*

Proof. Project the carrier of Definition 1925.1. The only classifier-visible topology data are T, x, y, U, V, D, with optional metric and real handoff rows M, E attached to the same packet.

Use the NameCert obligation rows of Definition 1925.2. They require classifier transport to preserve T, x, y, U, V, D through the componentwise hsame row H, and require metric-generated reads to return through M, E before yielding the same neighborhood rows.

The replay row C then repeats the displayed route from the point-pair comparison to U, V, D. Since the carrier has no coordinate for quotient point equality, selected representatives, host point apartness, or ambient completeness, transport cannot create one for the classifier to read. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier open neighborhood classifier transport`

Theorem 1925.8 (HausdorffSpace Real seal nonescape)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N)$$

reads only the displayed $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. In particular, such a consumer cannot extract quotient point equality, open-cover compactness, selected representatives, ambient completeness, or standard bridge closure from the packet.*

Proof. Project the carrier. Topology-facing reads use T, U, V, D; metric-generated reads additionally pass through M; Real-facing reads pass through the seal row E before returning to the same displayed separation rows.

The structural rows do not add hidden data. Transport is restricted to H, continuation replay to C, provenance to P, and local naming to N.

Since no other coordinate appears in the carrier, completion-facing and Real-facing consumers can inspect only the displayed topology, metric, Real, history, transport, continuation, package, and naming rows. The excluded quotient, compactness, representative-selection, ambient-completeness, and bridge data have no projection from the packet. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier real seal nonescape`

Theorem 1925.9 (HausdorffSpace public separation export)**.** *For an accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N),$$

the carrier row of Definition 1925.1, the NameCert obligations of Definition 1925.2, the metric separation handoff of Definition 1925.3, the classifier transport of Definition 1925.7, the point-separation choice-free route of Definition 1925.6, and the Real-seal nonescape row of Definition 1925.8 assemble a public $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ *separation export over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows. This export has no coordinate for quotient point equality, selected representatives, open-cover compactness, ambient completeness, Lean verification, or bridge closure.*

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceCarrier public separation export`

Proof. Start from the carrier of Definition 1925.1. It displays the topology source, point-name rows, disjoint open-neighborhood rows, optional metric handoff, Real seal, and structural rows used by every public separation read.

Apply the NameCert obligations of Definition 1925.2. They identify the finite obligation surface for carrier admission, disjoint open-neighborhood separation, classifier transport, metric handoff, and nonescape.

Next use the metric handoff and classifier-transport routes, Definition 1925.3 and Definition 1925.7. These keep any metric-generated separation behind the displayed distance seal and preserve the same open-neighborhood rows under componentwise transport.

Finally apply Definition 1925.6 and Definition 1925.8. The public export is therefore exactly the finite separation packet over topology, metric, Real, history, transport, continuation, provenance, and naming rows, while quotient point equality, representatives, compactness, completeness, Lean verification, and bridge closure remain unavailable. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpacePublicSeparationExport`

Theorem 1925.10 (HausdorffSpace mature standard-bridge readiness)**.** *For an accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N),$$

the public separation export of Definition 1925.9, the metric handoff of Definition 1925.3, the classifier transport of Definition 1925.7, the choice-free point separation of Definition 1925.6, and the Real-seal nonescape of Definition 1925.8 are the complete paper-side mature interface for the packet. Lean verification and the standard bridge remain separate verification-axis obligations, but the theory-side carrier exports no additional coordinate beyond $T, x, y, U, V, D, M, E, H, C, P, N$*.*

Proof. Read the public separation export. It assembles the carrier admission, NameCert obligations, metric handoff, classifier transport, choice-free point separation, and Real-seal boundary into one public $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ surface.

Inspect the listed routes. Metric-generated separation must pass through M, E before it returns to U, V, D, classifier transport preserves the displayed rows through H, C, and point separation selects no representatives or quotient points.

Finally inspect provenance and local naming. The rows P, N name only the displayed packet, so the paper-side mature interface is exhausted by $T, x, y, U, V, D, M, E, H, C, P, N$. The Lean target and the standard bridge are therefore verification-axis tasks rather than additional theory-side coordinates. □

Lean4 ✓ `BEDC.Derived.HausdorffSpaceUp.HausdorffSpaceMatureStandardBridgeReadiness`

Theorem 1925.11 (HausdorffSpace bridge-facing certificate)**.** *For an accepted* $\mathsf{HausdorffSpace}^{\uparrow}$ *carrier*

$$S = (T, x, y, U, V, D, M, E, H, C, P, N),$$

the public separation export of Definition 1925.9 is the bridge-facing finite route through the $\mathsf{Topology}^{\uparrow}$ *source row* T*, optional* $\mathsf{Met}^{\uparrow}$ *separation handoff* M*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$ *row* N*. The route exposes disjoint open-neighborhood rows* U, V *with disjointness ledger* D*, and it refuses quotient point equality, open-cover compactness, selected representatives, ambient completeness, and machine bridge checking.*

Proof. Start from the public separation export. It already packages the carrier, NameCert obligations, metric handoff, classifier transport, point-separation route, and Real-seal nonescape into a finite separated-topology surface.

Read that surface bridge-facingly. The topology source is T, the optional metric contribution is M, and any metric distance value is visible only through the Real seal E before the route returns to U, V, D.

The structural rows provide the remaining bridge-facing data. Componentwise transport is H, replay is C, package provenance is P, and local naming is N.

Since the carrier has no coordinate for quotient point equality, open-cover compactness, selected representatives, ambient completeness, or machine bridge checking, the bridge-facing certificate cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$**.**
Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1925.1, Definition 1925.2, Definition 1925.3, Definition 1925.7, Definition 1925.6, Definition 1925.8, and Definition 1925.9 bind the public separated-topology export over carrier admission, NameCert obligations, metric separation handoff, classifier transport, choice-free point separation, and Real seal nonescape.

Formal verification: FormalStatus(NameCert$_{\mathsf{HausdorffSpace}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not claim quotient point equality, open-cover compactness, selected representatives, ambient completeness, Lean verification, or standard bridge closure.

Upgrade path: Mature closure requires examples or downstream consumers of the bridge-facing separated-topology export, while verification still requires a checked BEDC.Derived.HausdorffSpaceUp target.

Constructive story: A HausdorffSpace$^{\uparrow}$ packet is generated from a Topology source row, two point-name rows, two disjoint open-neighborhood rows, a disjointness ledger, an optional MetricSpace distance-separation handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1926

A Concrete Naming Certificate for $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$ packet records the rational nested-interval construction of a regular Cauchy name through finite endpoint windows, not a host interval intersection or ambient real-completeness theorem. It sits between the nested-interval surface of chapter 320, the rational interval endpoint carrier of chapter 354, the dyadic tolerance ledger of chapter 291, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter isolates the interval-to-regular-name handoff: finite rational windows and width controls are displayed before any consumer may read the $\mathsf{RegSeqRat}^{\uparrow}$ row or the $\mathsf{Real}^{\uparrow}$ seal.

Definition 1926.1 (Rational nested-interval limit carrier)**.** *A* $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, W, E, R, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the rational nested-interval row. It records the inspected finite chain of rational intervals, the endpoint order witnesses, and the displayed containment step from each interval to its predecessor.*
2. *W is the width and tolerance ledger. It records the finite dyadic and rational bounds that turn each inspected interval width into the requested regular-Cauchy tolerance.*
3. *E is the endpoint-enclosure row. It records the lower and upper rational endpoint windows read through the* $\mathsf{RationalInterval}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling surfaces.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed endpoint windows and the width ledger.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, visible only after I, W, E, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval chain, width ledger, endpoint enclosures, regular rational handoff, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite nested-interval request through I, W, E, R, S.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RationalNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, W, E, R, S, H, C, P, N$. The carrier has no coordinate for host interval intersections, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1926.2 (Rational nested-interval limit NameCert obligations)**.** *For every accepted* $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$ *carrier*

$$L = (I, W, E, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed packet with rational nested-interval data, width and tolerance data, endpoint enclosures, regular rational handoff, terminal real seal, transport, replay, provenance, and local naming;*
2. *nesting exactness: every interval read is one of the displayed rational rows of I, and every successor interval is read together with its displayed containment witness;*
3. *width-to-tolerance control: every requested tolerance consumed by the handoff row is discharged by a visible width entry in W and the corresponding endpoint enclosure in E;*
4. *RegSeqRat handoff exactness: the row R reads only the rational interval windows and width entries displayed by I, W, E;*
5. *Real-seal nonescape: the terminal seal S is reached only after I, W, E, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RationalNestedIntervalLimitUp.RationalNestedIntervalLimitCarrier semantic name certificate`

Proof. Carrier admission is projection from Definition 1926.1. The accepted packet displays exactly the rational interval chain, width ledger, endpoint enclosures, regular rational handoff, Real seal, and structural rows.

Nesting exactness is forced by I. A consumer can inspect only the finite chain already present in the packet, and each successor interval is carried with its containment witness into the preceding interval.

Width-to-tolerance control is forced by W and E. The tolerance request cannot reach the regular-Cauchy handoff until the width row supplies the bound and the endpoint-enclosure row supplies the rational windows used to witness it.

The RegSeqRat handoff exactness clause follows because R is downstream of I, W, E. It repacks those displayed interval windows as a regular rational readback and has no coordinate for a second chain, a quotient stream, or a selected host limit.

Finally, S, H, C, P, N close the route. The Real seal is terminal, while transport, replay, provenance, and the local NameCert row preserve the same finite data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 1926.3 (Rational nested-interval RegSeqRat handoff)**.** *Let $L = (I, W, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{RationalNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by L*

factors through the displayed rational interval windows I, E and the width ledger W before reaching the terminal $\mathsf{Real}^{\uparrow}$ *seal S. Equivalently, the only exported completion-facing route has the form*

$$I, E, W \longmapsto R \longmapsto S.$$

The route provides a regular rational name from finite rational nested-interval data; it does not provide host interval intersection, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer read of the regular rational handoff. The carrier gives no direct edge to R except through the displayed rational nested-interval row I, the endpoint-enclosure row E, and the width ledger W.

The endpoint windows in E are tied to the same inspected interval chain recorded in I. The width ledger W then supplies the tolerance comparison required for the regular-Cauchy readback, so the consumer cannot replace the displayed chain by a host intersection or by a selected endpoint.

After I, E, W have been displayed, the read may reach R. Only then may the terminal seal S be consumed, and the structural rows H, C, P, N transport, replay, source, and name the same route.

Because the carrier contains no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient Real completeness, every exported $\mathsf{RegSeqRat}^{\uparrow}$ read remains the finite route shown above. □

Theorem 1926.4 (Rational nested-interval limit window regularity)**.** *For every accepted* RationalNestedIntervalLim *carrier $L = (I, W, E, R, S, H, C, P, N)$, the terminal limit read is exposed only after the finite rational interval windows I, E and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R agree under the same* $\mathsf{StreamName}^{\uparrow}$ *schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RationalNestedInte}}$ *row. In particular, every Real-facing consumer must read*

$$I, E, W \longmapsto R \longmapsto S \longmapsto H, C, P, N$$

before treating the packet as a limit name.

Proof. Project the carrier of Definition 1926.1. The only endpoint windows available to the packet are the rational interval rows in I and the endpoint-enclosure row E, and the only tolerance schedule is the width ledger W.

Apply the RegSeqRat handoff theorem, Definition 1926.3. It forces every regular rational read through I, E, W before R is exposed, so the terminal limit cannot be read before the interval windows and regular readback agree on the same displayed schedule.

The structural rows H, C, P, N then transport, replay, source, and name that schedule. Since the carrier has no separate host intersection, quotient stream equality, selected limit, trichotomy, countable choice, or ambient completeness coordinate, the terminal seal S is readable only through the displayed finite-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalNestedIntervalLimit}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalNestedIntervalLimit}^{\uparrow}}$, seed)

Scope closed: Definition 1926.1, Definition 1926.2, and Definition 1926.3 seed the finite rational nested-interval route from endpoint windows and width ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalNestedIntervalLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval intersections, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nesting exactness, width-to-tolerance control, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalNestedIntervalL

Constructive story: A RationalNestedIntervalLimit$^{\uparrow}$ packet is generated from a rational nested-interval row, a width and tolerance ledger, endpoint enclosure windows, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1927

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusStability}^{\uparrow}$

Following Bishop and Bridges constructive Cauchy sequence discipline, the $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ packet records the finite transport surface by which a displayed regular Cauchy modulus remains usable after tail-equivalent regular names are presented to Real and Completion consumers. It sits over the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback boundary of chapter 301, the terminal real-name seal of chapter 38, and the request-to-window route of chapter 1918. The packet is a finite stability certificate for reusing a Cauchy modulus across a displayed tail-equivalence route; it is not quotient stream equality, selected limit data, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Definition 1927.1 (Regular Cauchy modulus stability carrier)**.** *A* $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, M, T, W_A, W_B, D_A, D_B, Q_A, Q_B, E_A, E_B, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose finite regular Cauchy observations are related by a displayed tail-equivalence route.*
2. *M is the reusable Cauchy-modulus row, read through the* $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$ *request-to-window surface.*
3. *T is the finite tail-equivalence ledger, recording the cutoff, paired tail windows, and rowwise agreement evidence that allows the same modulus request to be read from either source after the cutoff.*
4. *W_A and W_B are the* $\mathsf{StreamName}^{\uparrow}$ *tail-window rows selected from A and B by the request, the reusable modulus row M, and the ledger T.*
5. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers attached to the two selected tail windows.*
6. *Q_A and Q_B are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the selected windows and dyadic ledgers.*
7. *E_A and E_B are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the source, modulus, tail-equivalence, window, tolerance, and readback rows have been displayed.*

8. *H records componentwise* hsame *transport for the source rows, reusable modulus, tail-equivalence ledger, selected windows, tolerance ledgers, readbacks, real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from every Real or Completion request through $M, T, W_A, W_B, D_A, D_B, Q_A, Q_B,$ and then through H, P, N.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusStability}^{\uparrow}$, $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusStability}^{\uparrow}}$ *row.*

The classifier compares accepted stability packets only through the displayed rows $A, B, M, T, W_A, W_B, D_A, D_B, Q_A,$ It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusStabilityUp.RegularCauchyModulusStabilityTasteGate`
`single carrier alignment`

Theorem 1927.2 (Regular Cauchy modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ *carrier*

$$S = (A, B, M, T, W_A, W_B, D_A, D_B, Q_A, Q_B, E_A, E_B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusStability}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the source rows A, B, reusable modulus M, tail-equivalence ledger T, selected windows W_A, W_B, tolerance ledgers D_A, D_B, readbacks Q_A, Q_B, real seals E_A, E_B, and structural rows H, C, P, N;*

2. *modulus admission, requiring every transported precision request to read the displayed modulus row M before either tail window is selected;*

3. *tail-equivalence exactness, requiring T to store the cutoff and paired tail-window agreement used for the two sources;*

4. *window selection, requiring W_A and W_B to be selected from the same request and the same modulus row, with T supplying only the tail-alignment data;*

5. *dyadic accounting, requiring D_A and D_B to store exactly the tolerance radii and slack comparisons consumed by their corresponding selected windows;*

6. *readback and seal routing, requiring Q_A, E_A and Q_B, E_B to remain downstream of their matching source, window, and dyadic ledger;*

7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*

8. *non-escape for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard-bridge data.*

No regular Cauchy modulus stability obligation is stored outside these rows.

Proof. Project the carrier of Definition 1927.1. The only analytic coordinates are the two sources A, B, the reusable modulus M, the tail-equivalence ledger T, the two selected windows, the two dyadic ledgers, the two readbacks, and the two terminal seals.

The modulus route is forced before any branch read. A Real or Completion request first reads M, and the selected tail windows W_A, W_B are then formed from the same request together with the tail-equivalence ledger T.

The ledger T contributes only the finite cutoff and paired tail-window agreement evidence. It does not identify the streams as quotient objects and does not select a limiting real.

The dyadic, readback, and seal rows remain branchwise. The rows D_A, Q_A, E_A are downstream of A, W_A, and the rows D_B, Q_B, E_B are downstream of B, W_B.

Finally H, C, P, N transport, replay, source, and name the same displayed finite route. Since none of the excluded objects appears as a coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1927.3 (Regular Cauchy modulus stability tail-equivalence transport)**.** *Let*

$$S = (A, B, M, T, W_A, W_B, D_A, D_B, Q_A, Q_B, E_A, E_B, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ *carrier. Every Real or Completion request that uses the reusable modulus M on the A source has a transported finite route on the B source through the same modulus row, the displayed tail-equivalence ledger, the selected B-window, the B-side dyadic ledger, the B-side readback, and the B-side real seal. Conversely, every B-side routed request transports back through the displayed A-side rows. The transport supplies reuse of the modulus across tail-equivalent regular names without quotient equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Proof. Start with an A-side request. By Definition 1927.2, the request reads M before any selected window is consumed.

Use the tail-equivalence ledger T. It records the finite cutoff and paired tail-window agreement rows, so the same request and modulus row select the B-side window W_B whenever the A-side route has selected W_A beyond the cutoff.

The transported branch then reads D_B, Q_B, and E_B in order. These rows are the only B-side tolerance, readback, and seal coordinates in the carrier, so the transported route has no additional branch or hidden endpoint.

The reverse direction is symmetric. Begin with the B-side route, read the same modulus row M, use the same ledger T to recover the paired A-side window W_A, and then pass through D_A, Q_A, E_A.

The structural rows H, C, P, N record both directions as componentwise transport and displayed replay over the same finite packet. Thus the carrier transports the reusable modulus across the displayed tail-equivalence surface and introduces none of the excluded quotient, selector, choice, equality, order, completeness, verification, or bridge objects. □

Falsifiable prediction: Every $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ Real or Completion read that reuses a Cauchy modulus across tail-equivalent regular names factors through two RegSeqRat source rows, one reusable RegularCauchyModulusRealizer route, a finite tail-equivalence ledger, paired StreamName tail windows, paired DyadicRatCore tolerance ledgers, paired RegSeqRat readbacks, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-modulus-realizer, regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two source rows, one reusable modulus row, a tail-equivalence ledger, paired selected windows,

paired tolerance ledgers, paired readbacks, paired terminal seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusStability}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one reusable RegularCauchyModulusRealizer route, a finite tail-equivalence ledger, paired StreamName tail windows, paired DyadicRatCore tolerance ledgers, paired RegSeqRat readbacks, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1927.1, Definition 1927.2, and Definition 1927.3 seed the finite modulus-transport route for tail-equivalent regular names consumed by L10 Real and Completion interfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, modulus admission, tail-equivalence exactness, window selection, dyadic accounting, readback and seal routing, and tail-equivalence transport for BEDC.Derived.RegularCauchyModulusStabil

Chapter 1928

A Concrete Naming Certificate for $\mathsf{CauchyCompletionSplit}^{\uparrow}$

Following Bishop and Bridges constructive completion and the Bourbaki–Kelley language of completion as a universal reflective boundary, the $\mathsf{CauchyCompletionSplit}^{\uparrow}$ packet records the finite unit-counit split at the open Real and Completion phase. It binds the unit row of chapter 1527, the counit row of chapter 1539, the adjunction row of chapter 1456, the idempotent completion row of **??**, the triangle row of chapter 1795, the finite stream surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational route of chapter 301, the terminal real-name seal of chapter 38, and the retraction packet of chapter 1921. The chapter is a finite NameCert boundary only; it does not assert quotient completions, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge-axis evidence.

Definition 1928.1 (CauchyCompletionSplit carrier)**.** *A* $\mathsf{CauchyCompletionSplit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, U, V, A, I, G, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed source row whose completion-facing split is being named.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route admitting X into the completion-facing surface.*
3. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *route returning already displayed completion reads to the real-facing boundary.*
4. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row comparing unit-admitted consumers with completion-facing consumers over the same finite source.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *row recording that a repeated completion read contributes no further public completion row.*
6. *G is the* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *row comparing the unit-counit composite with the direct displayed read.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before any rational observation can be exported.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the window and split comparisons.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row consumed by the terminal seal.*
10. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the split has passed through W, D, R.*

11. *H records componentwise* hsame *transport for $X, U, V, A, I, G, W, D, R, E$ and the structural rows.*

12. *C records displayed* Cont *replay for consumers that inspect the unit side, the counit side, the triangle comparison, or the terminal real-facing route.*

13. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionSplit}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$, $\mathsf{CauchyCompletionRetraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSplit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected completion representatives, countable-choice schedules, host equality of completed objects, ambient completeness, or public bridge credentials.

Theorem 1928.2 (CauchyCompletionSplit NameCert obligations)**.** *For an accepted* $\mathsf{CauchyCompletionSplit}^{\uparrow}$ *carrier $S = (X, U, V, A, I, G, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSplit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source row, unit row, counit row, adjunction row, idempotent row, triangle row, finite-window row, dyadic ledger, regular readback, terminal seal, transport, replay, provenance, and local naming rows;*

2. *unit exposure, requiring every completion-facing read exported from X to pass through U before a completion consumer can inspect it;*

3. *counit return, requiring every displayed completion-facing read to pass through V before it reaches W, D, R, E;*

4. *finite split exactness, requiring the composite through U, A, I, G, V to expose the same terminal route through W, D, R, E as the direct source read;*

5. *classifier stability, requiring transported packets with the same displayed rows to be identified by the carried* hsame *and* Cont *data;*

6. *non-escape, requiring the packet to export no quotient-completion equality, selected representative, countable-choice schedule, host equality, ambient-completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionSplitUp.CauchyCompletionSplitTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 1928.1. The tuple lists exactly the source, unit, counit, adjunction, idempotent, triangle, finite-window, dyadic, readback, seal, transport, replay, provenance, and local naming rows.

Unit exposure is read from U. A completion-facing consumer starting at X receives no direct completion object; the carrier first displays the route by which X enters the completion-facing surface.

Counit return is read from V. Once a completion-facing read is already displayed, the packet forces the return route through the finite observation rows W, D, R before the terminal seal E can be consumed.

Finite split exactness is the shared comparison supplied by A, I, G. The adjunction row fixes the universal consumer route, the idempotent row states that a repeated completion read contributes

no further public row, and the triangle row compares the unit-counit route with the direct route to the same seal.

Classifier stability is componentwise transport along H followed by replay through C. Since the carrier fields are exhausted by $X, U, V, A, I, G, W, D, R, E, H, C, P, N$, the excluded quotient, representative-choice, countable-choice, host-equality, ambient-completeness, real-completeness, and bridge data have no row from which a consumer could read them. □

Theorem 1928.3 (CauchyCompletionSplit unit-counit route)**.** *Let $S = (X, U, V, A, I, G, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionSplit}^{\uparrow}$ *carrier. Every consumer of the finite unit-counit split factors through the displayed route*

$$X \longmapsto U \longmapsto A, I, G \longmapsto V \longmapsto W, D, R \longmapsto E \longmapsto H, C, P, N.$$

After this route is projected, the visible seal-facing read is the same finite route

$$W, D, R \longmapsto E$$

that the source row already displays. The split is therefore a finite unit-counit boundary over BEDC rows, not equality of quotient completions, not a selected representative, not host equality, and not countable choice or ambient completeness.

Proof. Start with a consumer of the split packet and project the source row X. The only completion-facing admission exposed by the carrier is U, so the consumer first passes through the unit row.

The middle comparison is fixed by A, I, G. The adjunction row A records that the completion-facing consumer is a consumer of the same reflected route, the idempotent row I prevents a repeated completion read from adding a public row, and the triangle row G compares the unit-counit composite with the displayed route to the seal.

The return part is then forced by V. The counit row routes the already displayed completion-facing read through the finite stream window W, the dyadic tolerance ledger D, and the regular rational readback R before the terminal seal E is inspected.

Transport by H and replay by C preserve this route for equivalent packets, while P and N record provenance and local naming. No other carrier coordinate exists, so the split exports exactly the displayed finite route and none of the excluded quotient, representative, host-equality, choice, or completeness data. □

Theorem 1928.4 (CauchyCompletionSplit nonescape)**.** *For an accepted* $\mathsf{CauchyCompletionSplit}^{\uparrow}$ *carrier $S = (X, U, V, A, I, G, W, D, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing,* $\mathsf{StreamName}^{\uparrow}$*-facing, or completion-facing read exported by S is generated by the rows named in Definition 1928.1. In particular, the packet exports no quotient-completion equality, selected completion representative, countable-choice schedule, host equality of completed objects, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Project the accepted carrier. A $\mathsf{StreamName}^{\uparrow}$-facing, $\mathsf{RegSeqRat}^{\uparrow}$-facing, or $\mathsf{Real}^{\uparrow}$-facing read must pass through W, D, R, E, because these are the only finite observation, tolerance, readback, and terminal seal rows in the tuple.

A completion-facing read must pass through U, V, A, I, G. These are the only completion-route rows in the tuple, and each is stored as displayed NameCert data rather than as a quotient point or selected representative.

The structural rows H, C, P, N transport, replay, source, and name the same route. They add no carrier coordinate beyond the listed rows.

Thus every exported read is generated by Definition 1928.1. The forbidden quotient, representative, choice, host-equality, ambient-completeness, real-completeness, and bridge data are absent from the carrier and cannot be produced by projection, transport, replay, provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionSplit}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionSplit}^{\uparrow}$ packet is generated from a displayed source row, CauchyCompletionUnit route, CauchyCompletionCounit route, CauchyCompletionAdjunction route, CauchyCompletionIdempotent row, CauchyCompletionTriangleIdentity row, StreamName finite windows, DyadicRatCore tolerances, RegSeqRat readback, terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSplit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1928.1, Definition 1928.2, Definition 1928.3, and Definition 1928.4 seed the finite unit-counit split boundary for the open Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSplit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected completion representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, counit return, split exactness, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionSplitUp.

Chapter 1929

A Concrete Naming Certificate for $\mathsf{MonotoneInverse}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MonotoneInverse}^{\uparrow}$ packet records inverse readback for strictly monotone real maps by finite interval and bisection data. It is a NameCert surface for an inverse-reading route, not a host inverse function, real trichotomy, quotient real equality, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. The packet consumes the sign and interval refinement discipline of chapter 1422, the monotone limit route of chapter 1914, the local slope and image controls of chapter 1907 and chapter 1912, and the real-name readback lattice of chapter 38, chapter 126, chapter 301, and chapter 291.

Definition 1929.1 (Monotone inverse carrier)**.** *A* $\mathsf{MonotoneInverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, T, M, B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source interval row. It records the input endpoint windows, endpoint order evidence, located subinterval identity, and the displayed domain restriction on which the map is inspected.*
2. *T is the target interval row. It records the target real-name window, endpoint-bracket data, and the finite assertion that the target request lies between the displayed source endpoint images.*
3. *M is the strict monotonicity evidence row. It stores the finite comparison route for source windows and the orientation flag, increasing or decreasing, used by every inverse read.*
4. *B is the finite bisection schedule. It records branch prefixes, midpoint image comparisons, retained source intervals, and the handoff to the intermediate-value bisection surface.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for source endpoints, target requests, midpoint images, and retained source windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational approximants extracted from W before any real-facing seal is inspected.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records requested precisions, dyadic interval widths, branch-depth budgets, image-comparison tolerances, and retained-window radii.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *inverse seal. It is reached only after the source interval, target interval, monotonicity evidence, bisection schedule, stream windows, regular readback, and dyadic tolerance rows have been displayed.*

9. *H records componentwise* hsame *transport for the interval rows, monotonicity row, bisection schedule, stream windows, regular readbacks, dyadic tolerances, inverse seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a target request through interval inspection, monotonicity comparison, bisection refinement, stream-window exposure, regular readback, dyadic tolerance accounting, and inverse-seal consumption.*

11. *P is the* Pkg *provenance over* $\mathsf{MonotoneInverse}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$, $\mathsf{ConstructiveMeanValue}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneInverse}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, T, M, B, W, R, D, E, H, C, P, N$. It has no coordinate for a host inverse function, real trichotomy, quotient real equality, selected representative, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an unrecorded comparison of real values.*

Lean4 ✓ `BEDC.Derived.MonotoneInverseUp.MonotoneInverseTasteGate single carrier alignment`

Theorem 1929.2 (Monotone inverse NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneInverse}^{\uparrow}$ *carrier*

$$I = (A, T, M, B, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MonotoneInverse}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source interval, target interval, strict monotonicity evidence, finite bisection schedule, stream windows, regular readbacks, dyadic tolerance rows, inverse seal, transport, replay, provenance, and local naming;*

2. *interval discipline, requiring A and T to expose the source bracket and target-in-image bracket before any branch comparison or real-facing inverse row is available;*

3. *monotonicity discipline, requiring M to orient every midpoint comparison and retained-interval decision before the bisection schedule can be read;*

4. *bisection handoff discipline, requiring B, W, R, D to display branch prefixes, finite windows, regular rational readbacks, and dyadic tolerance accounting before the inverse seal is inspected;*

5. *structural nonescape, requiring E, H, C, P, N to export only the displayed route and no host inverse function, real trichotomy, quotient real equality, selected representative, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or unrecorded real comparison.*

No MonotoneInverse obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1929.1. The accepted packet displays exactly the source interval, target interval, strict monotonicity evidence, bisection schedule, stream windows, regular readbacks, dyadic tolerances, inverse seal, and structural rows.

The interval discipline is the entry point. The source row A fixes the domain bracket, while the target row T fixes the requested value as a displayed bracket between endpoint image rows. Without those two rows there is no bisection request to orient.

The monotonicity discipline supplies the orientation. The row M is the only stored evidence that source-order comparisons determine image-order comparisons, so each branch decision in B must read M before retaining a half-interval.

The bisection handoff is then forced by the packet order. The schedule B produces finite retained source windows, W exposes those windows as StreamName data, R reads them as regular

rational approximants, and D records the dyadic tolerance budget used by the requested inverse precision.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the excluded analytic objects are absent from the carrier, no local obligation can export them. □

Theorem 1929.3 (Monotone inverse bisection handoff)**.** *Every inverse read of an accepted* $\mathsf{MonotoneInverse}^{\uparrow}$ *carrier factors through the displayed route*

$$A, T, M \longmapsto B \longmapsto W, R, D \longmapsto E, H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *inverse seal is public. The route first reads the source and target interval rows, then the strict monotonicity orientation, then the finite bisection schedule, then the StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows, and only then the terminal real-name seal. It exports a finite inverse-read packet rather than a host inverse function or an ambient completeness argument.*

Proof. Start with an accepted packet $I = (A, T, M, B, W, R, D, E, H, C, P, N)$. The only source-domain and target-request coordinates are A and T, so every inverse read begins by opening those displayed interval rows.

Next use strict monotonicity. The row M supplies the orientation that turns each midpoint image comparison into a retained source half-interval. The bisection schedule B is therefore downstream of M, and it cannot be read as an unordered search or as real trichotomy.

The readback stage is finite. The branch prefix in B exposes retained windows through W, the regular rational approximants are read through R, and the requested precision is accounted for through D. These are exactly the StreamName, RegSeqRat, and DyadicRatCore rows available before sealing.

The $\mathsf{Real}^{\uparrow}$ inverse seal E is terminal. The transport, replay, provenance, and local naming rows H, C, P, N preserve and present the same route. Since no carrier coordinate stores a host inverse, quotient equality, selected representative, or ambient completeness witness, every inverse read factors through the displayed bisection and monotonicity route before the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneInverse}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneInverse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1929.1, Definition 1929.2, and Definition 1929.3 seed the finite constructive inverse-read route for strictly monotone real maps through displayed intervals, bisection data, stream windows, regular readbacks, dyadic tolerances, and a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneInverse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, real trichotomy, quotient real equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window exactness, bisection handoff, inverse-seal nonescape, and ledger provenance for BEDC.Derived.MonotoneInverseUp.

Constructive story: A $\mathsf{MonotoneInverse}^{\uparrow}$ packet is generated from source and target interval rows, strict monotonicity evidence, a finite bisection schedule, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, a $\mathsf{Real}^{\uparrow}$ inverse seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1930

A Concrete Naming Certificate for $\mathsf{RegularCauchyExtensionality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ packet records the finite tail-coincidence route by which two regular Cauchy rational names present the same sealed real name. It sits between the regular rational rows of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the separated uniqueness route of chapter 427. The certificate is a tail-invariance surface for Real and Completion consumers; it is not quotient stream equality, host real equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1930.1 (Regular Cauchy extensionality carrier)**.** *A* $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R_0, R_1, S_0, S_1, D, T, A_0, A_1, H, C, P, N)$$

with the following displayed rows:

1. *R_0 and R_1 are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name rows to be compared.*
2. *S_0 and S_1 are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which downstream requests read R_0 and R_1.*
3. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-tolerance ledger recording the visible radii, slack comparisons, and finite dyadic requests used by both names.*
4. *T is the eventual tail-coincidence row, displaying a threshold after which every requested window of S_0 and S_1 reads the same dyadic rational observation through D.*
5. *A_0 and A_1 are the two* $\mathsf{Real}^{\uparrow}$ *seal rows reached from the two regular names after the finite windows and shared ledger have been exposed.*
6. *H records componentwise* hsame *transport for the two regular rows, two stream-window rows, shared dyadic ledger, tail-coincidence row, two Real seals, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each Real or Completion request through the selected tail threshold, the corresponding finite stream windows, the shared dyadic ledger, the two regular readbacks, the two Real seals, and the structural rows.*

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularLimitUniqueness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionality}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyExtensionalityUp.TasteGate.RegularCauchyExtensionalityTasteGate`
`single carrier alignment`

Theorem 1930.2 (Regular Cauchy extensionality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ *carrier*

$$E = (R_0, R_1, S_0, S_1, D, T, A_0, A_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* R_0, R_1*, two* $\mathsf{StreamName}^{\uparrow}$ *window rows* S_0, S_1*, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, tail-coincidence row* T*, two* $\mathsf{Real}^{\uparrow}$ *seals* A_0, A_1*, and structural rows* H, C, P, N*;*

2. *tail-threshold exactness, requiring* T *to display a finite threshold after which the two stream-window routes make the same dyadic observations through* D*;*

3. *window stability, requiring every downstream request beyond the displayed threshold to be read through matching finite windows of* S_0 *and* S_1 *before either Real seal is inspected;*

4. *regular-readback routing, requiring the compared observations to remain attached to* R_0 *and* R_1 *as* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks;*

5. *Real-seal routing, requiring* A_0 *and* A_1 *to be downstream of the displayed regular rows, stream windows, dyadic ledger, and tail-coincidence row;*

6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*;*

7. *non-escape for quotient stream equality, host real equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No extensionality obligation is stored outside these rows.

Proof. Project the carrier of Definition 1930.1. The analytic rows are exactly the two regular rational names, their two finite stream-window surfaces, the shared dyadic ledger, the tail-coincidence threshold, and the two terminal Real seals.

Read the threshold row next. The row T is the only place where the two names are compared, and it compares them only after a displayed finite threshold. Beyond that threshold, the requested windows of S_0 and S_1 read the same dyadic observations through D before either route reaches a Real seal.

The structural rows preserve the same route. The hsame rows transport the visible coordinates, the Cont rows replay the requests through the displayed windows and ledger, the Pkg row records provenance, and the local naming row names precisely this packet.

Since the omitted objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 1930.3 (Regular Cauchy extensionality tail-coincidence seal)**.** *Let*

$$E = (R_0, R_1, S_0, S_1, D, T, A_0, A_1, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ *carrier. Suppose* T *displays a tail threshold* k *such that every downstream precision request served beyond* k *reads the same finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S_0 *and* S_1 *and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows in* D*. Then every* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *tail route before reaching* A_0 *or* A_1*.*

Consequently the comparison exported by the packet is tail-coincidence invariance of the two sealed regular names, with visible BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance. It excludes quotient stream equality, host real equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the displayed tail threshold k in the row T. A consumer request below the threshold is not used to identify the two seals; the comparison route begins only once the request has passed into the tail certified by T.

After that point, both names read the same finite windows. The row S_0 serves the request for R_0, the row S_1 serves the request for R_1, and the threshold row states that these two window reads call on the same entries of the shared ledger D.

The dyadic ledger forces the two readbacks to use one tolerance surface. Each compared observation remains attached to its source regular row, so the consumer route is still a $\mathsf{RegSeqRat}^{\uparrow}$ route, but the tail portion of the route is common.

Only then are the two Real seals inspected. The rows A_0 and A_1 are downstream of the displayed regular rows, stream windows, shared dyadic ledger, and tail-coincidence row, so every Real-facing consumer factors through the same finite tail surface before reading a seal.

The exclusion clauses follow from the carrier inventory. No coordinate contains quotient stream equality, host real equality, a selected limit, countable choice, or ambient Real completeness, and the structural rows can replay only the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ Real or Completion consumer that identifies two $\mathsf{Real}^{\uparrow}$ seals without reading a displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail route, matching $\mathsf{StreamName}^{\uparrow}$ windows, and the shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-limit-uniqueness: no listed sibling supplies the whole twelve-row packet $E = (R_0, R_1, S_0, S_1, D, T, A_0, A_1, H, C, P, N)$, because each sibling contributes only a regular rational source, finite window surface, dyadic ledger, Real seal, or separated uniqueness route while this chapter binds tail-coincidence invariance with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyExtensionality}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ rows, two $\mathsf{StreamName}^{\uparrow}$ finite-window rows, a shared $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, an eventual tail-coincidence row, two $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1930.1, Definition 1930.2, and Definition 1930.3 seed the tail-coincidence invariance route from regular Cauchy names to Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyExtensionality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, tail-threshold exactness, window stability, regular-readback routing, Real-seal routing, and tail-coincidence seal factorization for BEDC.Derived.RegularCauchyExtensionalityUp.

Chapter 1931

A Concrete Naming Certificate for $\mathsf{CauchyCompletionTrace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletionTrace}^{\uparrow}$ packet records the finite source-to-completion trace behind completion-facing reads. It binds the source schedule of chapter 126, the regular rational readback of chapter 301, the dyadic budget ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the monad route of chapter 378, the functor route of chapter 423, the universal-property route of chapter 397, and the displayed retraction route of chapter 1921. The chapter packages these rows as a finite L10 Real-and-Completion trace only; it does not assert quotient completions, selected limits, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1931.1 (CauchyCompletionTrace carrier)**.** *A* $\mathsf{CauchyCompletionTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, E, U, F, V, H, C, P, N)$$

with the following visible rows:

1. *S is the source schedule row, recording the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite windows that a completion-facing read is allowed to inspect.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational name extracted from the source schedule before the terminal real seal is opened.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, recording the dyadic tolerance ledger used by S and R.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the source schedule, dyadic budget, and regular readback have been displayed.*
5. *U is the completion unit or monad row, recording the finite* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *admission route consumed by completion-facing reads.*
6. *F is the completion functor row, recording the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *transport of the same source and readback data.*
7. *V is the universal-property or retraction-facing trace row, recording the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *route or the* $\mathsf{CauchyCompletionRetraction}^{\uparrow}$ *route through which a completion consumer returns to the displayed real boundary.*
8. *H records componentwise* hsame *transport for S, R, D, E, U, F, V and the structural rows.*

9. *C records displayed* Cont *replay for consumers that inspect the source schedule, completion admission, functor transport, universal-property route, or retraction-facing route.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionTrace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, CauchyCompletionR BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-axis evidence.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionTraceUp.CauchyCompletionTraceTasteGate single carrier alignment`

Theorem 1931.2 (CauchyCompletionTrace NameCert obligations)**.** *For an accepted* $\mathsf{CauchyCompletionTrace}^{\uparrow}$ *carrier* $T = (S, R, D, E, U, F, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionTrace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source schedule, regular readback, dyadic budget, real seal, completion unit or monad row, completion functor row, universal-property or retraction-facing trace row, transport, replay, provenance, and local naming rows;*

2. *trace ordering, requiring every completion-facing consumer to read S, D, R, E before treating the completion rows U, F, V as visible completion data;*

3. *source-to-completion compatibility, requiring U to admit only the source and readback data displayed by S, D, R, E;*

4. *functor compatibility, requiring F to transport the same source schedule, dyadic budget, regular readback, and real seal rows;*

5. *section routing, requiring V to return a universal-property or retraction-facing consumer to the displayed S, D, R, E boundary;*

6. *classifier stability, requiring transported packets with the same displayed rows to be identified by the carried* hsame *and* Cont *data;*

7. *boundary non-escape, stating that the packet exports no quotient completion, selected limit, host equality of completed objects, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Carrier admission is projection from Definition 1931.1. The tuple lists exactly the source schedule, readback, dyadic budget, seal, unit or monad, functor, universal-property or retraction trace, transport, replay, provenance, and local naming rows.

Trace ordering is finite-route induction on the displayed consumer route. The base read is the source schedule S, and each later real-facing observation must pass through the dyadic budget D, the regular readback R, and the terminal seal E before any completion row is inspected.

The completion-facing rows are then forced in order. The unit or monad row U admits the displayed source route, the functor row F transports that same route, and the row V supplies the universal-property or retraction-facing return to the real boundary.

Classifier stability is componentwise transport along H followed by replay through C. Since the carrier fields are exhausted by $S, R, D, E, U, F, V, H, C, P, N$, the excluded quotient, selected-limit,

host-equality, countable-choice, ambient-completeness, and bridge data have no row from which a consumer could read them. □

Theorem 1931.3 (CauchyCompletionTrace section route)**.** *Let $T = (S, R, D, E, U, F, V, H, C, P, N)$ be an accepted* $\mathsf{CauchyCompletionTrace}^{\uparrow}$ *carrier. Every completion-facing consumer exported by T factors through the displayed finite route*

$$S, D, R, E \longmapsto U \longmapsto F \longmapsto V \longmapsto H, C, P, N.$$

The section seen by the consumer is therefore the route from source schedule and regular readback to completion admission, functor transport, and universal-property or retraction-facing return. It is not equality of quotient completions, not a selected limit, not host equality, not countable choice, and not ambient real completeness.

Proof. Project the accepted carrier and start with a completion-facing consumer. Its source-facing observations are confined to S, D, R, E, because these are the only stream, dyadic, readback, and real-seal coordinates of the carrier.

The completion admission is then read from U. This row receives the already displayed source schedule and regular readback route; it does not create a separate completion object or choose a representative.

Functorial transport is read from F, and the return or section-facing comparison is read from V. The two rows share the same preceding S, D, R, E, U data, so finite-route induction gives one section route rather than unrelated completion reads.

Finally H and C transport and replay the same route, while P and N source and name it. Since no other coordinate exists, every completion-facing consumer factors through the displayed section route and none of the excluded data can be exported. □

Theorem 1931.4 (CauchyCompletionTrace real boundary)**.** *For an accepted* $\mathsf{CauchyCompletionTrace}^{\uparrow}$ *carrier $T = (S, R, D, E, U, F, V, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing read exported by the trace is generated by the source schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal seal E. The completion rows U, F, V may cite that boundary and return to it, but they cannot export quotient completions, selected limits, host equality of completed objects, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credentials.*

Proof. Project the accepted carrier. A real-facing read must inspect the source schedule row S, the dyadic budget row D, the regular readback row R, and the terminal seal row E, since these are the only rows that carry real-boundary observations.

The completion rows do not add a second real boundary. The unit or monad row U admits the displayed route, the functor row F transports it, and the universal-property or retraction-facing row V returns completion consumers to the same S, D, R, E boundary.

Transport, replay, provenance, and local naming preserve this boundary as H, C, P, N. They add no quotient-completion coordinate, no selected-limit coordinate, no host-equality coordinate, no countable-choice coordinate, no ambient-completeness theorem, and no bridge credential.

Thus every real-facing read is generated by the displayed finite source-to-seal route, and every completion-facing read remains bounded by that route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionTrace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionTrace}^{\uparrow}$ packet is generated from a StreamName source schedule, RegSeqRat readback, DyadicRatCore budget, Real seal, CauchyCompletionMonad admission route, CauchyCompletionFunctor transport row, CompletionUniversalProperty or CauchyCompletionRetraction return row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyCompletionTrace}\uparrow}$, seed)

Scope closed: Definition 1931.1, Definition 1931.2, Definition 1931.3, and Definition 1931.4 seed the finite source-to-completion trace behind completion-facing reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyCompletionTrace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality of completed objects, countable choice, ambient Real$^{\uparrow}$ completeness, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, trace ordering, section routing, real-boundary nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionTraceUp.

Chapter 1932

A Concrete Naming Certificate for $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$

Following Hardy–Littlewood and Hardy summability theory, the $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ packet records the constructive finite route by which an Abel-limit boundary and a one-sided Tauberian tail condition hand a real series back to ordinary convergence. It sits after the Abel-test surface of chapter 1496, the finite summation-by-parts route of chapter 1763, the Cesaro averaging route of chapter 1765, the real-series source of chapter 1581, the power-series boundary of chapter 1583, the summable-tail modulus of chapter 1799, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite NameCert surface for Abel-limit handoff; it is not an ambient Tauberian theorem, a quotient series equality, a selected host limit, trichotomy, countable choice, open-cover compactness, or host completeness.

Definition 1932.1 (Hardy–Littlewood Tauberian carrier)**.** *A* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, P, A, L, U, M, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing only finite inspected coefficient windows and finite partial-sum rows.*
2. *P is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *Abel row, displaying the finite radial polynomial windows used before the boundary parameter approaches* 1*.*
3. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *route row, displaying the summation-by-parts identity that relates source partial sums to Abel-window estimates.*
4. *L is the Abel-limit handoff row, displaying a regular boundary schedule and its finite radial estimates.*
5. *U is the Tauberian one-sided tail row, recording the finite boundedness or slow-decrease ledger needed to prevent mass from escaping between the Abel windows and ordinary partial sums.*
6. *M is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *comparison row, assigning every requested dyadic tolerance to a finite source window, radial window, and Tauberian tail ledger.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the ordinary partial-sum route obtained after S, P, A, L, U, M have been displayed.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular readback row.*

9. *H records componentwise* hsame *transport for source windows, Abel polynomial windows, summation-by-parts rows, Abel-limit estimates, Tauberian tail ledgers, modulus comparisons, regular readback, real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a precision request through the Abel-limit handoff, through the Tauberian tail ledger, then through the ordinary partial-sum readback and real seal.*
11. *Q is the* Pkg *provenance over* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyLittlewoodTauberian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve displayed rows. It has no coordinate for a host Abel theorem, quotient equality of series names, selected boundary limit, countable choice, trichotomy, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.HardyLittlewoodTauberianUp.HardyLittlewoodTauberianTasteGate single carrier alignment`

Theorem 1932.2 (Hardy–Littlewood Tauberian NameCert obligations)**.** *For every accepted* HardyLittlewoodTaube *carrier $T = (S, P, A, L, U, M, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HardyLittlewoodTauberian}^{\uparrow}}$ *surface consists exactly of carrier admission for that twelve-row packet, source-series exposure through S, Abel-window exposure through P, finite summation-by-parts exposure through A, Abel-limit boundary exposure through L, Tauberian one-sided tail exposure through U, modulus comparison through M, ordinary partial-sum readback through R, real sealing through E, and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted packet $T = (S, P, A, L, U, M, R, E, H, C, Q, N)$. Carrier admission exposes exactly the twelve rows listed in Definition 1932.1; no hidden limit selector or host summability coordinate remains available.

Read the analytic route in order. The source row S supplies finite coefficient and partial-sum windows, the Abel row P supplies the finite radial windows, and the summation row A binds those two surfaces by the displayed summation-by-parts identity.

The Abel-limit row L then gives only regular boundary estimates. The Tauberian row U and modulus row M are the sole rows that compare those estimates with ordinary tails, so every accepted ordinary readback must pass through them before reaching R.

Finally, componentwise hsame transport in H preserves the same route, and Cont replay in C can only replay rows already present in the carrier. The provenance and local naming rows Q, N source and name that packet, so the listed clauses exhaust the local certificate surface. □

Theorem 1932.3 (Hardy–Littlewood Abelian boundary)**.** *Every Abel-limit read exported by an accepted* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ *carrier factors through*

$$S, P, A, L, H, C, Q, N.$$

In particular, an Abel-facing consumer may inspect real-series windows, power-series radial windows, and the AbelSummation route, but it cannot read a completed ordinary sum or host boundary value from the Abel side alone.

Proof. Begin with an Abel-facing read. The only rows before the Tauberian comparison stage are the source row S, the radial power-series row P, the finite summation-by-parts row A, and the Abel-limit handoff row L.

The structural rows H, C, Q, N transport, replay, source, and name that same Abel boundary. They do not contain an ordinary convergence proof or a selected completed real value.

Since U, M, R, E are downstream of the Tauberian comparison, an Abel-facing read that has not consumed those rows remains an Abelian boundary read. It can supply the boundary input for the Tauberian step, but it cannot by itself export ordinary series convergence. □

Theorem 1932.4 (Hardy–Littlewood real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ *carrier factors through the full finite route*

$$S, P, A, L, U, M, R, E, H, C, Q, N.$$

Thus the ordinary real-series endpoint is public only after the Abel boundary and the Tauberian tail ledger have both been displayed through the same componentwise hsame *and* Cont *route.*

Proof. Start with a Real-facing consumer. By Definition 1932.1, the only real seal row is E, and E is reachable only from the regular readback row R.

Trace R backward. Its input is the modulus comparison row M, which compares the Abel-limit estimates in L with ordinary partial-sum tails using the one-sided Tauberian ledger U. The Abel-limit row itself is supplied by the source, radial, and summation rows S, P, A.

The transport and replay rows preserve this route componentwise. Therefore a Real-facing read cannot bypass the Abel boundary, cannot omit the Tauberian tail ledger, and cannot replace the regular readback by host completeness, quotient series equality, selected limits, countable choice, trichotomy, or open-cover compactness. □

Falsifiable prediction: An accepted $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the Abel boundary row, the Tauberian one-sided tail row, the SummableTailModulus comparison row, and the RegSeqRat readback row refutes the carrier surface.

Independence witness: abel-test, abel-summation, hardy-cesaro-mean, realseries, realpowerseries, summable-tail-modulus, regseqrat, real: no listed sibling supplies the full twelve-row packet $T = (S, P, A, L, U, M, R, E, H, C, Q, N)$, because each sibling contributes only a source, transform, averaging comparison, tail modulus, readback, or seal face while this chapter binds the Abel-limit boundary to the ordinary real-series endpoint in one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HardyLittlewoodTauberian}^{\uparrow}}$.

Constructive story: A $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$ packet is generated from real-series source windows, real-power-series radial windows, AbelSummation rows, Abel-limit handoff rows, Tauberian one-sided tail ledgers, SummableTailModulus comparison rows, RegSeqRat readback, a RealUp seal, componentwise hsame transports, Cont replay, Pkg provenance, and local NameCert.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodTauberian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1932.1, Definition 1932.2, Definition 1932.3, and Definition 1932.4 seed the finite Abel-limit handoff from constructive real series to a RealUp endpoint.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodTauberian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Tauberian theory, quotient series equality, selected limits, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the twelve-row carrier and prove the finite NameCert obligation surface, Abelian boundary theorem, Real seal handoff theorem, and TasteGate field faithfulness for BEDC.Derived.HardyLittlewoodTauberianUp.

Chapter 1933

Located Real NameCert Construction

Following Bishop and Bridges, a located real is represented by certified regular real-name data plus finite dyadic locatedness windows, not by a quotient or classical trichotomy. The $\mathsf{LocatedReal}^{\uparrow}$ chapter records this central L10 object as the public-facing route beneath the interval consumer of chapter 1709, the comparison consumer of chapter 1730, and the tail-envelope consumer of chapter 1769. It is the hyphenated reference point for the same finite route already exposed by chapter 309: $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Definition 1933.1 (Located real carrier)**.** *A* $\mathsf{LocatedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation schedule that names which regular rational windows are read at a requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational stream, regularity budget, and readback data visible at the scheduled windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger; it stores the dyadic lower endpoint, upper endpoint, radius, and slack data used by the locatedness window.*
4. *E is the located enclosure row, asserting that the scheduled R-windows lie inside the dyadic interval supplied by D at the requested radius.*
5. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row obtained by forgetting only the locatedness enclosure while retaining the same regular real-name source surface.*
6. *H records componentwise* hsame *transport for S, R, D, E, Q and for the structural replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a locatedness request through S, R, D, E, Q, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located real-name surface. It contains no Dedekind cut, quotient real, host real equality, trichotomy decision, countable-choice selector, compactness witness, metric-completion object, or bridge credential.

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealTasteGate single carrier alignment`

Definition 1933.2 (Located real classifier)**.** *Two accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carriers*

$$L = (S, R, D, E, Q, H, C, P, N)$$

and

$$L' = (S', R', D', E', Q', H', C', P', N')$$

are classified as the same located real exactly when their $\mathsf{StreamName}^{\uparrow}$ *schedules admit a common finite observation window, their* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks agree on that window through the displayed regular rational classifier, their* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledgers refine to a common dyadic enclosure, and the enclosure and* $\mathsf{Real}^{\uparrow}$ *seal rows transport through the same* H, C, P, N *structural route.*

The classifier is therefore common-window regular readback plus dyadic enclosure refinement. It is not quotient equality of real names, host equality of completed reals, trichotomy of endpoints, or a selected representative for the underlying $\mathsf{Real}^{\uparrow}$ *seal.*

Theorem 1933.3 (Located real NameCert obligations)**.** *For every accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carrier* $L = (S, R, D, E, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream schedule, regular rational readback, dyadic endpoint and radius ledger, located enclosure, real seal, transport, replay, provenance, and local naming rows;*
2. *classifier exactness, requiring sameness to be common finite-window agreement of* S, R *together with common dyadic enclosure refinement of* D, E*;*
3. *regular seal compatibility, requiring the* $\mathsf{Real}^{\uparrow}$ *row* Q *to read the same* $\mathsf{RegSeqRat}^{\uparrow}$ *source surface and no extra locatedness-independent representative;*
4. *transport stability, requiring* H *and* C *to preserve the schedule, readback, dyadic ledger, enclosure row, real seal row, provenance, and local name componentwise;*
5. *public-route exhaustion, requiring every downstream* $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, *or* $\mathsf{LocatedRealTailEnvelope}^{\uparrow}$ *consumer to factor through* $S, R, D, E, Q, H, C, P, N$*;*
6. *non-escape, excluding Dedekind cuts, quotient reals, host real equality, trichotomy, countable choice, compactness, metric completion, Lean verification, and bridge checking as local data.*

No located-real obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1933.1. The accepted packet displays exactly the stream schedule, regular rational source, dyadic endpoint and radius ledger, enclosure row, real seal, and structural rows.

Classifier exactness is projection from Definition 1933.2. A comparison of two packets first chooses a common finite window for their $\mathsf{StreamName}^{\uparrow}$ schedules, then compares the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks on that window, and only then refines the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and radius ledgers.

Regular seal compatibility is the row Q. It forgets the located enclosure but keeps the same regular rational source and scheduled observations, so it cannot introduce a host representative or quotient equality.

Transport and public-route exhaustion are structural. The rows H and C move and replay the displayed packet, while P and N source and name it. Since chapter 1709 and the comparison and tail-envelope consumers cite $\mathsf{LocatedReal}^{\uparrow}$ only through these public rows, no excluded coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.LocatedRealUp.LocatedRealNameCertObligations`

Theorem 1933.4 (Located real regular seal)**.** *Every accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carrier* $L = (S, R, D, E, Q, H, C, P, N)$ *presents its terminal* $\mathsf{Real}^{\uparrow}$ *seal by retaining the scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *source* R *and the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, while forgetting only the located enclosure row* E *and its dyadic endpoint-radius ledger* D*.*

Conversely, a $\mathsf{Real}^{\uparrow}$ *seal-facing regular source may be equipped with a* $\mathsf{LocatedReal}^{\uparrow}$ *carrier only by adding finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius rows together with an enclosure row over the already displayed* S, R *windows.*

Proof. Read the carrier projection. The $\mathsf{Real}^{\uparrow}$ seal coordinate Q is downstream of the same S and R rows used by the locatedness enclosure, so it sees the scheduled regular rational source before any locatedness row is consulted.

Forgetting D and E leaves S, R, Q, H, C, P, N and therefore preserves the regular real-name surface. The forgotten rows are precisely the dyadic endpoint, radius, and enclosure data.

For the converse, start with a displayed regular seal source S, R, Q and attach a finite $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D plus an enclosure row E for the scheduled observations. Repacking with H, C, P, N yields the carrier of Definition 1933.1; no quotient, trichotomy, or selected completed real is introduced. □

Theorem 1933.5 (Located real transport stability)**.** *Let* $L = (S, R, D, E, Q, H, C, P, N)$ *and* $L' = (S', R', D', E', Q', H', C', P', N')$ *be accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carriers. If* H *records componentwise* hsame *transport from each analytic row of* L *to the corresponding analytic row of* L'*, and the displayed continuations replay the transported rows, then* L' *is accepted by the same located-real classifier route.*

The transported packet reads only the $\mathsf{StreamName}^{\uparrow}$ *schedules,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, enclosure rows,* $\mathsf{Real}^{\uparrow}$ *seals, and the visible* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *provenance.*

Proof. Unpack L through Definition 1933.1. The analytic coordinates are S, R, D, E, Q, and the remaining coordinates only transport, replay, source, and name those rows.

Transport S and R to the scheduled source rows of L'. Transport D and E along the same componentwise route, so the dyadic endpoint and radius ledger still encloses the transported regular rational windows.

The seal row Q transports to Q' by the regular seal statement of Definition 1933.4. Replaying through C', P', N' repacks the accepted carrier, and Definition 1933.2 identifies the classifier route. No unlisted host equality, quotient, or trichotomy row is available during the transport. □

Theorem 1933.6 (Located real public route)**.** *Every downstream consumer of an accepted* $\mathsf{LocatedReal}^{\uparrow}$ *carrier, including the interval consumer of chapter 1709, the comparison consumer of chapter 1730, and the tail-envelope consumer of chapter 1769, receives exactly the public route*

$$S, R, D, E \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer can inspect scheduled regular rational windows, dyadic endpoint and radius ledgers, located enclosure data, the regular $\mathsf{Real}^{\uparrow}$ *seal, and the visible provenance rows. It cannot inspect a quotient real, host real equality, trichotomy decision, countable-choice selector, compactness witness, metric-completion object, or bridge credential.*

Proof. Begin with the accepted carrier of Definition 1933.1. Its only analytic rows are the stream schedule S, the regular rational readback R, the dyadic ledger D, the enclosure row E, and the real seal Q.

Apply Definition 1933.4 to bind the seal-facing read to the same S, R source. Apply Definition 1933.5 to show that componentwise movement of the packet remains inside the same displayed carrier route.

The cited interval, comparison, and tail-envelope consumers receive $\mathsf{LocatedReal}^{\uparrow}$ endpoint or source data only through that carrier route. Since the carrier and classifier contain none of the excluded objects, every downstream read is exhausted by $S, R, D, E, Q, H, C, P, N$. $\square$

Chapter 1934

A Concrete Naming Certificate for $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$

Following Bishop and Bridges constructive completion and Bourbaki/Kelley completion terminology, the packet records the finite BEDC certificate for the canonical entry of a displayed source into its Cauchy completion. It sits after the unit route of chapter 1527, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names the entry map as finite NameCert data; it does not assert quotient completion equality, select limits, identify host objects, invoke countable choice, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1934.1 (CauchyCompletionEmbedding carrier)**.** *A* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (X, U, C, S, R, D, E, H, Q, P, N)$$

with the following visible rows:

1. *X is the displayed source row whose point, metric, or regular-Cauchy data are admitted into the completion-facing entry.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route that exposes the source row before any completion seal is read.*
3. *C is the completion seal row recording that the entry is consumed by the Cauchy-completion surface as displayed finite data.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source is observed before tolerance or completion-facing reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by finite consumers of the entry.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds each readback and entry request.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the route through S, R, D has been displayed.*
8. *H records componentwise* hsame *transport for the source row, unit route, completion seal, finite-window row, regular readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay for consumers that inspect the source row, the unit route, the finite readback, or the completion seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or public bridge credentials.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionEmbeddingUp.CauchyCompletionEmbeddingTasteGate single carrier alignment`

Theorem 1934.2 (CauchyCompletionEmbedding NameCert obligations)**.** *For an accepted* CauchyCompletionEmbe *carrier $J = (X, U, C, S, R, D, E, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source row, unit route, completion seal, finite-window row, regular readback, dyadic ledger, Real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *entry exposure, requiring every completion-facing consumer to read X through U before the completion seal C is available;*
3. *finite-window locality, requiring the source observation to pass through S before R, D, E or C can be inspected;*
4. *tolerance and readback exactness, requiring each finite precision request to be answered by D and then R before the terminal seal or completion seal is consumed;*
5. *classifier stability, requiring transported packets with the same displayed X, U, C, S, R, D, E route to be identified by the carried* hsame *and* Cont *data;*
6. *ledger non-escape, stating that the packet exports no quotient completion equality, selected limit, host equality, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Proof. Carrier admission is projection from Definition 1934.1. The tuple lists exactly the source, unit, completion seal, finite-window, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows.

Entry exposure is constructive because the unit route U is the only row connecting X to the completion seal C. A consumer starting from the source has no detached completion point to inspect; it must first read the displayed unit route.

Finite-window locality then projects S. The regular readback R, the dyadic ledger D, the Real seal E, and the completion seal C all occur after the finite source observation, so no later row can be consumed before the window row is present.

Tolerance and readback exactness use D and R. Each finite request is bounded by the dyadic ledger and read back as regular rational data before either seal is exposed.

Classifier stability is componentwise transport along H followed by replay through Q. Since the carrier fields are exhausted by $X, U, C, S, R, D, E, H, Q, P, N$, the excluded quotient, selected-limit, host-equality, choice, ambient-completeness, real-completeness, and bridge data have no row from which a consumer could read them. □

Theorem 1934.3 (CauchyCompletionEmbedding density handoff)**.** *Let $J = (X, U, C, S, R, D, E, H, Q, P, N)$ be an accepted* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ *carrier. Every density, functoriality, reflector, or*

universal-property consumer of the source-to-completion entry factors through the finite route

$$X \longmapsto U \longmapsto S \longmapsto D, R \longmapsto E \longmapsto C \longmapsto H, Q, P, N.$$

The handoff uses only finite windows and regular readback. It is not equality of quotient completions, not a selected limit, not host equality, and not an appeal to countable choice or ambient completeness.

Proof. Start with a consumer of the entry packet. Projection gives the displayed source row X, and the only completion-facing admission row available from X is the unit route U.

The consumer must next read the finite window row S. This keeps the handoff local to a bounded observation of the source before any regular readback, dyadic tolerance ledger, Real seal, or completion seal is exposed.

The tolerance and readback rows D and R answer the finite request. The terminal real-facing boundary E is reached only after that regular route has been displayed, and the completion seal C consumes the same finite route as the entry map.

Transport by H and replay by Q preserve this route for equivalent packets, while P and N record provenance and local naming. No carrier coordinate exists for quotient equality, selected limits, host equality, countable choice, or ambient completeness, so downstream density and universal-property consumers receive only the finite handoff described above. □

Theorem 1934.4 (CauchyCompletionEmbedding Real seal non-escape)**.** *For an accepted* CauchyCompletionEmbed *carrier $J = (X, U, C, S, R, D, E, H, Q, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read exported by J is generated by the rows named in Definition 1934.1. In particular, the Real seal cannot be used as a detached completed point, quotient-completion equality witness, selected limit, host equality witness, countable-choice selection, or ambient completeness witness.*

Proof. Project the accepted carrier. A $\mathsf{Real}^{\uparrow}$-facing read must pass through S, D, R, E, because those are the only finite observation, tolerance, readback, and terminal seal rows in the tuple.

A completion-facing read must also pass through U and C. The unit route records the source entry before the completion seal is consumed, and the completion seal is stored as displayed NameCert data rather than as a quotient point, selected limit, or host equality witness.

The structural rows H, Q, P, N transport, replay, source, and name the same route. They add no carrier coordinate beyond the listed rows.

Thus every exported read is generated by Definition 1934.1. The forbidden quotient, selected-limit, host-equality, choice, ambient-completeness, real-completeness, and bridge data are absent from the carrier and cannot be produced by projection, transport, replay, provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ packet is generated from a displayed source row, a CauchyCompletionUnit route, a completion seal, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 1934.1, Definition 1934.2, Definition 1934.3, and Definition 1934.4 seed the finite source-to-completion entry surface for L10 completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, entry exposure, finite-window locality, tolerance exactness, readback exactness, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionEmbeddingUp.

Chapter 1935

A Concrete Naming Certificate for $\mathsf{LocatedRegularization}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRegularization}^{\uparrow}$ packet records the finite boundary where a Cauchy stream is regularized while retaining positive locatedness and apartness evidence. It sits after the regularization route of chapter 376, the positive apartness boundary of chapter 1881, the Real-facing Cauchy readiness surface of chapter 1590, the finite windows of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later Real and completion chapters one finite route from source Cauchy data to located regular names; it is not a quotient stream, not a selected limit, not a trichotomy decision, not countable choice, not ambient $\mathsf{Real}^{\uparrow}$ completeness, and not bridge checking.

Definition 1935.1 (LocatedRegularization carrier)**.** *A* $\mathsf{LocatedRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, M, A, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy stream row before regularization or locatedness evidence is read.*
2. *M is the modulus-threshold row consumed from the* $\mathsf{CauchyRegularization}^{\uparrow}$ *route.*
3. *A is the positive located/apartness row consumed from the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *boundary.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window schedule through which the source stream and apartness row are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff obtained from the scheduled windows and the modulus threshold.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, slack rows, and comparison tolerances used by the located window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the regularized name after W, R, D and the positive row A have been displayed.*
8. *H records componentwise* hsame *transports for S, M, A, W, R, D, E and for the structural replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each located regularization request through $S, M, A, W, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRegularization}^{\uparrow}$, $\mathsf{CauchyRegularization}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularization}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located regularization surface. It contains no quotient stream, selected limit, trichotomy decision, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness witness, or bridge credential.*

Lean4 ✓ `BEDC.Derived.LocatedRegularizationUp.LocatedRegularizationTasteGate single carrier alignment`

Theorem 1935.2 (LocatedRegularization NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRegularization}^{\uparrow}$ *carrier* $L = (S, M, A, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy stream, modulus threshold, positive located/apartness row, finite windows, regular rational handoff, dyadic tolerance ledger, Real seal, transports, replay, provenance, and local naming rows;*

2. *located-window compatibility, requiring each positive locatedness read to inspect S only through the* $\mathsf{StreamName}^{\uparrow}$ *schedule W and the dyadic ledger D before R or E can be consumed;*

3. *apartness preservation, requiring the positive row A to remain attached to the same scheduled regularization route S, M, W, R, D under componentwise* hsame *transport and displayed* Cont *replay;*

4. *Real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$*-facing read of E to pass through the displayed regularization, window, readback, tolerance, and positive-apartness rows rather than through a detached completed real;*

5. *provenance exhaustion, requiring every downstream Real, completion, Cauchy-criterion, or apartness consumer to read only $S, M, A, W, R, D, E, H, C, P, N$ and no quotient stream, selected limit, trichotomy, countable-choice source, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or bridge credential.*

No located-regularization obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1935.1. The accepted packet lists the source stream, modulus threshold, positive located/apartness row, finite window schedule, regular rational handoff, dyadic tolerance ledger, Real seal, and structural rows.

For located-window compatibility, read the row order in the carrier. The positive row A is not exported directly to a Real-facing consumer; it is inspected through the same finite window schedule W and the same dyadic ledger D that feed the regular rational handoff R.

Apartness preservation follows by row-order induction along the transported packet. The base step preserves S, M, A by the corresponding coordinates of H, and the window, handoff, tolerance, and seal steps replay those transported rows through C without changing the positive apartness source.

Real-seal nonescape is the terminal-row clause. The seal E appears only after W, R, D and A have been displayed, so a consumer cannot obtain a detached completed real, selected limit, quotient representative, or ambient completeness witness by projecting E.

Finally, provenance exhaustion is projection from P and N together with the absence of further carrier coordinates. Since L has only the rows $S, M, A, W, R, D, E, H, C, P, N$, downstream Cauchy-criterion, apartness, Real, and completion consumers receive exactly that finite route and no excluded bridge or choice data. □

Theorem 1935.3 (LocatedRegularization real-seal factorization)**.** *For every accepted* $\mathsf{LocatedRegularization}^{\uparrow}$ *carrier* $L = (S, M, A, W, R, D, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read of the terminal seal* E *factors through the displayed finite row sequence*

$$S \longrightarrow M \longrightarrow A \longrightarrow W \longrightarrow R \longrightarrow D \longrightarrow E$$

together with the componentwise transport, replay, provenance, and local naming rows H, C, P, N*. Thus the seal can be consumed only after the source Cauchy stream* S*, the* $\mathsf{CauchyRegularization}^{\uparrow}$ *modulus threshold* M*, the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive row* A*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D *have all been displayed on the same carrier route. The factorization provides no quotient stream, selected limit, trichotomy decision, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness, or bridge credential.*

Proof. Project the accepted carrier to the row list of Definition 1935.1. The seal coordinate E is not a standalone Real object in that list; it is the terminal row following the source stream, regularization threshold, positive apartness row, finite windows, regular rational handoff, and dyadic tolerance ledger.

Next inspect the local obligation surface of Definition 1935.2. Located-window compatibility forces the positive row A to be read through W and D, apartness preservation keeps A attached to the route S, M, W, R, D under H and C, and the Real-seal nonescape row requires every $\mathsf{Real}^{\uparrow}$ consumer of E to use that same displayed route.

It remains only to exclude alternate exits. The carrier has no coordinates for quotient streams, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge credentials, and provenance exhaustion says that downstream Real and completion consumers may read only $S, M, A, W, R, D, E, H, C, P, N$. Therefore every accepted seal read factors through the stated finite rows before reaching E. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRegularization}^{\uparrow}$ packet is generated from a source Cauchy stream, a CauchyRegularization modulus threshold, a CauchyRealApartness positive row, StreamName finite windows, RegSeqRat handoff, DyadicRatCore tolerance rows, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 1935.1, Definition 1935.2, and Definition 1935.3 seed the finite located regularization boundary between Cauchy regularization, positive apartness, Real Cauchy readiness, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window compatibility, apartness preservation, provenance exhaustion, and a Lean carrier/obligation formalization for BEDC.Derived.LocatedRegularizationUp.

Chapter 1936

A Concrete Naming Certificate for $\mathsf{GrzegorczykReal}^{\uparrow}$

Following Grzegorczyk's computable-real presentation and Weihrauch-style represented-space terminology, the $\mathsf{GrzegorczykReal}^{\uparrow}$ packet records the effective real-name boundary where a displayed algorithmic approximant is read through finite windows before it reaches the terminal real seal. It sits over the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the computable-real service boundary of chapter 336, the terminal real-name seal of chapter 38, and the noncomputable contrast supplied by chapter 1406. The packet names an effective route from approximant data to a $\mathsf{Real}^{\uparrow}$ consumer; it is not quotient stream equality, a completed real selected by choice, a host-real comparison, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 1936.1 (Grzegorczyk real carrier)**.** *A* $\mathsf{GrzegorczykReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the effective approximant row, recording the finite program, request precision, and rational output cell presented in Grzegorczyk's computable-real format.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which the approximant is inspected at the requested precisions.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording dyadic radii, slack comparisons, and finite error budgets attached to the approximant windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reachable only after the effective approximant, finite windows, dyadic ledger, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for A, S, D, R, E and for the structural replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each effective real-name request through the route*

$$A \longmapsto S \longmapsto D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{GrzegorczykReal}^{\uparrow}$, $\mathsf{ComputableReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GrzegorczykReal}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite effective real-name surface. It contains no quotient stream equality, selected limit, host real comparison, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness witness, or bridge credential.*

Lean4 ✓ `BEDC.Derived.GrzegorczykRealUp.TasteGate.GrzegorczykRealTasteGate single carrier alignment`

Theorem 1936.2 (Grzegorczyk real NameCert obligations)**.** *For every accepted* $\mathsf{GrzegorczykReal}^{\uparrow}$ *carrier $G = (A, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GrzegorczykReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the effective approximant, finite StreamName windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, transports, replay, provenance, and local naming rows;*
2. *modulus-window exactness, requiring each precision request made by the effective approximant row A to be witnessed by the finite window row S before R or E can be consumed;*
3. *dyadic tolerance stability, requiring the ledger D to preserve the displayed error budget for every transported approximant window and every regular readback cell;*
4. *RegSeqRat handoff, requiring R to be the only route by which the effective approximant windows become a regular rational real name;*
5. *RealUp seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer of E to pass through A, S, D, R and the structural rows H, C, P, N rather than through quotient equality, a selected limit, countable choice, a host real, or ambient completeness.*

No Grzegorczyk-real obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1936.1. The accepted packet lists exactly the effective approximant, finite windows, dyadic ledger, regular readback, terminal seal, and structural rows.

For modulus-window exactness, read the carrier route from A to S. A precision request in A is not exported directly; it is witnessed by the finite $\mathsf{StreamName}^{\uparrow}$ window row S before any regular readback or Real-facing seal can be inspected.

Dyadic tolerance stability is the next coordinate of the route. The ledger D stores the radii and slack comparisons attached to each displayed window, and the transport row H preserves those cells componentwise while C replays the same finite path.

The RegSeqRat handoff is then forced by row order. The row R appears after the window and tolerance data and before the seal, so a consumer cannot treat the approximant stream as a real name without passing through the regular rational readback.

Finally, RealUp seal non-escape follows from terminality of E and provenance exhaustion in P, N. Since the carrier has no further coordinates, every $\mathsf{Real}^{\uparrow}$ consumer factors through $A, S, D, R, E, H, C, P, N$ and receives no quotient stream equality, selected limit, countable-choice source, host real comparison, ambient completeness principle, or bridge credential. □

Theorem 1936.3 (Grzegorczyk real stream-real seal factorization)**.** *For every accepted* $\mathsf{GrzegorczykReal}^{\uparrow}$ *carrier $G = (A, S, D, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$ *consumer of the seal E factors through the displayed effective approximant, finite stream window, dyadic tolerance, regular readback, and seal*

rows:

$$A \longmapsto S \longmapsto D \longmapsto R \longmapsto E.$$

In particular the consumer cannot bypass the $\mathsf{ComputableReal}^{\uparrow}$ *approximant row, cannot replace* S *by a quotient stream, cannot replace* R *by a selected limit, and cannot read* E *as an ambient completeness witness.*

Proof. Start with a seal-facing request. Carrier projection exposes E only as the terminal coordinate of the displayed Grzegorczyk-real route, not as an independently completed real object.

Trace the route backward from E. The seal is downstream of R, the regular readback is downstream of the dyadic tolerance ledger D, and the ledger is attached to finite windows S requested by the effective approximant row A.

The structural rows preserve this route. Componentwise hsame transport in H keeps the same approximant, window, tolerance, readback, and seal coordinates, while the Cont replay row C records that each consumer request follows the same finite path.

The provenance and local naming rows P, N name no additional real-completion principle. Thus every seal consumer factors through A, S, D, R, E together with H, C, P, N, and no excluded quotient, selection, host-real, countable-choice, ambient-completeness, or bridge route is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{GrzegorczykReal}\uparrow}$.

Constructive story: A $\mathsf{GrzegorczykReal}^{\uparrow}$ packet is generated from an effective approximant row, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GrzegorczykReal}\uparrow}$, seed)

Scope closed: Definition 1936.1, Definition 1936.2, and Definition 1936.3 seed the effective real-name boundary between computable approximants, finite stream windows, regular rational readback, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GrzegorczykReal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real comparison, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-window exactness, dyadic tolerance stability, RegSeqRat handoff, RealUp seal non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.GrzegorczykRealUp.

Chapter 1937

A Concrete Naming Certificate for $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane's functorial language, the $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ packet records the finite BEDC surface by which a uniformly continuous map between displayed sources induces a map between their Cauchy-completion seals. It reads the unit route of chapter 1527, the adjunction ledger of chapter 1456, the extension handoff of chapter 769, and the separated uniqueness boundary of **??**. The chapter names only the finite NameCert boundary for this functorial action; it does not identify quotient completions, choose representatives, or assert host equality of completed spaces.

Definition 1937.1 (CauchyCompletionFunctoriality carrier)**.** *A* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, f, E, U, I, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source completion row, including the displayed source object, its Cauchy-completion seal, and the unit route consumed before any completed read is available.*
2. *T is the target completion row, including the displayed target object, its Cauchy-completion seal, and the matching unit route.*
3. *f is the uniformly continuous source-map row, carrying the finite modulus and graph evidence used before completion-facing extension.*
4. *E is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row that extends f from the dense source route into the target completion seal.*
5. *U is the* $\mathsf{CompletionExtensionUniqueness}^{\uparrow}$ *row comparing any two extension candidates after agreement on the dense source and target completion rows.*
6. *I is the identity ledger, recording that the source identity map extends through the same unit and adjunction rows to the identity completed read.*
7. *L is the composition ledger, recording that two composable source-map rows extend in the same order as their source-side composition.*
8. *H records componentwise* hsame *transport for the source completion row, target completion row, source-map row, extension row, uniqueness row, identity ledger, composition ledger, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the route*

$$S, T, f \longmapsto E \longmapsto U \longmapsto I, L.$$

10. *P is the* Pkg *provenance over* CauchyCompletionFunctoriality$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionAdjunc UniformContinuousExtension$^{\uparrow}$, CompletionExtensionUniqueness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCompletionFunctoriality}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed spaces, open-cover compactness, or completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionFunctorialityUp.CauchyCompletionFunctorialityTasteGate`
`single carrier alignment`

Theorem 1937.2 (CauchyCompletionFunctoriality NameCert obligations)**.** *For every accepted* CauchyCompletionFunctoriality$^{\uparrow}$ *carrier*

$$F = (S, T, f, E, U, I, L, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionFunctoriality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source completion row, target completion row, uniformly continuous source-map row, extension row, uniqueness row, identity ledger, composition ledger, componentwise transport, replay, provenance, and local naming row;*
2. *unit compatibility, requiring both completion rows to expose their dense source routes before an extended completed-map read is available;*
3. *extension exactness, requiring the row E to be the only completion-facing route generated from the uniformly continuous source-map row f;*
4. *uniqueness determinacy, requiring any two extension candidates over the same dense route to be compared only through the row U;*
5. *identity replay, requiring the identity source-map row to pass through the same unit and adjunction rows before the identity completed read is named by I;*
6. *composition replay, requiring composable source-map rows to pass through the same extension and uniqueness surfaces before the composed completed read is named by L;*
7. *ledger non-escape, requiring H, C, P, N to expose no quotient completion equality, selected representatives, countable-choice schedule, host equality of completed spaces, open-cover compactness, or completeness of* Real$^{\uparrow}$.

Proof. Carrier admission is projection from Definition 1937.1. The accepted packet displays exactly the source completion row, target completion row, uniformly continuous source-map row, extension row, uniqueness row, identity ledger, composition ledger, and structural rows.

Unit compatibility is read from S and T. Each completed read begins with the corresponding source-side unit route, so no completed-map row can be inspected before the dense source and target routes have been displayed.

Extension exactness is the projection to f and E. The source map is uniformly continuous by the carried finite modulus and graph evidence, and the only completion-facing image row generated from it is the UniformContinuousExtension$^{\uparrow}$ surface named by E.

Uniqueness determinacy is the projection to U. When two candidates agree on the displayed dense source route and target completion route, the packet compares them through the separated completion-extension uniqueness surface rather than through host function equality.

Finally, identity and composition are finite replay obligations. The rows I and L reuse the same unit and adjunction coordinates already present in S, T, E, U, while H, C, P, N preserve only those displayed coordinates under componentwise transport, replay, provenance, and local naming. Since no excluded principle is a carrier coordinate, the seven obligation rows exhaust the local NameCert surface. □

Theorem 1937.3 (CauchyCompletionFunctoriality map route)**.** *Let*

$$F = (S, T, f, E, U, I, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *carrier. Every completed-map read induced by the uniformly continuous source-map row f factors through the finite route*

$$S, T, f \longmapsto E \longmapsto U \longmapsto H, C, P, N.$$

The route uses the existing uniform-continuous extension surface for existence and the completion-extension uniqueness surface for determinacy. It does not use quotient completion equality, selected representatives, countable choice, host equality of completed spaces, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with a completed-map consumer. Projection gives the source and target completion rows S, T and the uniformly continuous source-map row f.

The map route then enters the extension surface. The row E is the only coordinate that turns the source-map evidence into a completion-facing target read, so the consumer must pass through the $\mathsf{UniformContinuousExtension}^{\uparrow}$ handoff before it can read the completed map.

Determinacy is supplied by U. If a second candidate is presented over the same dense source route and target completion row, the comparison is routed through $\mathsf{CompletionExtensionUniqueness}^{\uparrow}$ and not through equality of host functions or completed spaces.

Transport and naming close the finite route. The componentwise hsame ledger H, the displayed Cont replay C, the provenance row P, and the local naming row N preserve the same map route and add no coordinate for any excluded completion or choice principle. □

Theorem 1937.4 (CauchyCompletionFunctoriality composition)**.** *For accepted* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *carriers over composable uniformly continuous source-map rows, the completed read of the composite source map is named by replaying the same unit, adjunction, extension, and uniqueness rows used for the two component maps. The identity source map is named by the same replay with the identity ledger I, and the composite source map is named by the composition ledger L.*

Proof. For identity, project the carrier and read the ledger I. The source identity row first passes through the unit route in the source completion row, then through the adjunction-facing completion route, and then through the extension row E. The uniqueness row U identifies the resulting completed read with the identity completed read carried by I.

For composition, apply the same projection to the composable source-map rows. Each component map is extended through its own displayed E row, and the composite source-map row is then extended through the same uniform-continuous extension surface. The ledger L records that these two completion-facing routes are compared only by the shared completion-extension uniqueness row.

The componentwise transport row H and replay row C preserve the identity and composition ledgers under accepted carrier comparison. Provenance P and local naming N therefore expose a functorial completed-map surface without exporting quotient completion equality, selected representatives, countable choice, host equality of completed spaces, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctoriality}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ packet is generated from source and target completion rows, a uniformly continuous source-map row, a UniformContinuousExtension handoff, a CompletionExtensionUniqueness ledger, identity and composition ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctoriality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1937.1, Definition 1937.2, Definition 1937.3, and Definition 1937.4 seed the finite functorial action surface for Cauchy completion on uniformly continuous maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionFunctoriality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed spaces, open-cover compactness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit compatibility, extension exactness, uniqueness determinacy, identity replay, composition replay, and ledger non-escape for BEDC.Derived.CauchyCompletionFunctorialityUp.

Chapter 1938

A Concrete Naming Certificate for $\mathsf{IntervalLength}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{IntervalLength}^{\uparrow}$ packet names the finite diameter data of a closed interval before any appeal to compactness, selected endpoints, or completed real equality. It sits after the dyadic arithmetic surface of chapter 291, the endpoint window interfaces of chapter 306 and chapter 354, and the real-window enclosure route of chapter 1625. Downstream uses in chapter 38 read only this finite diameter certificate when a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ argument needs the length of an interval window.

Definition 1938.1 (IntervalLength carrier)**.** *An* $\mathsf{IntervalLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, O, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left endpoint row, accepted either as a* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint projection or as a* $\mathsf{RationalInterval}^{\uparrow}$ *endpoint row.*
2. *R is the right endpoint row, accepted through the same endpoint surface as L.*
3. *O is the endpoint-order ledger proving that the displayed left row precedes the displayed right row in the corresponding finite rational or dyadic comparison surface.*
4. *D is the dyadic diameter row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *row naming the finite value read as $R - L$ on the common endpoint window.*
5. *E is the real-interval enclosure handoff row, linking the same endpoint and diameter data to the finite enclosure surface of chapter 1625.*
6. *H records componentwise* hsame *transport for the endpoint rows, order ledger, diameter row, enclosure handoff, continuation row, provenance row, and local naming row.*
7. *C records the* Cont *replay by which a later finite consumer reads the same endpoint, order, diameter, and enclosure rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalLength}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these nine rows form one finite interval-length packet. It has no coordinate for open-cover compactness, selected endpoints, quotient real equality, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.IntervalLengthUp.IntervalLengthUpTasteGate single carrier alignment`

Theorem 1938.2 (IntervalLength NameCert obligations)**.** *For every accepted* $\mathsf{IntervalLength}^{\uparrow}$ *carrier* $I = (L, R, O, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalLength}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two endpoint rows, endpoint-order ledger, dyadic diameter row, enclosure handoff, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *endpoint-order exposure, requiring every length read to display the same* O *row that certifies the endpoint orientation;*
3. *diameter exactness, requiring* D *to be the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *subtraction row* $R - L$ *on the common endpoint window;*
4. *enclosure handoff, requiring* E *to route the same endpoints and diameter into the finite real-interval enclosure surface;*
5. *classifier transport, requiring* H *to transport endpoint, order, diameter, enclosure, provenance, and naming rows componentwise;*
6. *replay stability, requiring* C *to replay the same finite rows for later* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers;*
7. *provenance exhaustion, requiring* P *to name every consumed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*, and* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *source row;*
8. *ledger non-escape, excluding any read through open-cover compactness, selected endpoint choice, quotient real equality, trichotomy, or ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 1938.1. The projection exposes exactly two endpoint rows, one endpoint-order ledger, one dyadic diameter row, one enclosure handoff, and the structural rows H, C, P, N.

Endpoint-order exposure is the row O. Since the carrier stores D next to O, a consumer cannot read a length without also reading the finite orientation ledger for the same endpoint window.

Diameter exactness is the row D. It is typed as a $\mathsf{DyadicRatCore}^{\uparrow}$ subtraction row on the displayed endpoint surface, so the value read by a consumer is the finite dyadic difference $R - L$ and not a host metric length.

The handoff row E is then read through the real-interval enclosure surface. Transport by H and replay by C preserve the same endpoint, order, diameter, and enclosure rows componentwise.

Finally, P and N exhaust provenance and naming. Because the carrier has no coordinate for open-cover compactness, selected endpoints, quotient real equality, trichotomy, or ambient completeness, the listed non-escape conditions are immediate from the packet inventory. □

Theorem 1938.3 (IntervalLength dyadic diameter handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntervalLength}^{\uparrow}$ *packet reads the interval diameter only through the displayed route*

$$L, R, O \longmapsto D \longmapsto E \longmapsto H, C, P, N.$$

The route factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. It does not expose an ambient metric length, a selected endpoint pair, quotient equality of real names, trichotomy, open-cover compactness, or a completeness witness.*

Proof. Start with an accepted carrier $I = (L, R, O, D, E, H, C, P, N)$. The only endpoint coordinates are L and R, and the only orientation coordinate attached to them is O.

The diameter coordinate is D. By Definition 1938.1, D is a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row computed from the same endpoint window, so every diameter read first factors through the finite dyadic arithmetic surface.

The enclosure handoff is E. It routes the same endpoint and diameter rows into the finite real-interval enclosure packet, which is the only surface available to a $\mathsf{Real}^{\uparrow}$ consumer at this boundary.

Finally H, C, P, N package transport, replay, provenance, and local naming. Thus a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives exactly the displayed BEDC rows and no ambient metric length, selected endpoint pair, quotient real equality, trichotomy, compactness theorem, or completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalLength}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalLength}^{\uparrow}$ packet is generated from displayed left and right endpoint rows, an endpoint-order ledger, a dyadic diameter row, a real-interval enclosure handoff, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntervalLength}^{\uparrow}}$, seed)

Scope closed: Definition 1938.1, Definition 1938.2, and Definition 1938.3 seed the finite interval-diameter surface used by DyadicRatCore, DyadicInterval, RationalInterval, RealIntervalEnclosure, RegSeqRat, and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntervalLength}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected endpoints, quotient real equality, trichotomy, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order exposure, diameter exactness, enclosure handoff, classifier transport, replay stability, provenance exhaustion, and ledger non-escape proofs for BEDC.Derived.IntervalLengthUp.

Chapter 1939

A Concrete Naming Certificate for $\mathsf{FejerKernel}^{\uparrow}$

Following Fejer's positive summability kernels and the classical accounts of Zygmund and Katznelson, the $\mathsf{FejerKernel}^{\uparrow}$ packet records the finite Fourier kernel that averages symmetric coefficient windows before any constructive approximation theorem consumes a uniform-limit handoff. It sits over the finite Fourier observation interface of chapter 124, the circle source of chapter 58, the real-analytic trigonometric rows of chapter 74, the normalization route of chapter 123, and the approximation-facing boundary of chapter 463. The chapter names a positive finite kernel object; it does not import Haar measure, quotient function spaces, host Fourier convergence, or a completed approximation theorem.

Definition 1939.1 (Fejer kernel carrier)**.** *A* $\mathsf{FejerKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, A, I, U, P, T, C, Q, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row, recording the symmetric window, triangular weights, endpoint truncation bound, and finite coefficient probes used by the kernel.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-sample row, exposing only displayed circle samples and their componentwise classifier data.*
3. *A is the* $\mathsf{RealAnalytic}^{\uparrow}$ *sine/cosine polynomial row, presenting the trigonometric polynomial determined by F and S as finite real-analytic algebra data.*
4. *I is the* $\mathsf{Integral}^{\uparrow}$ *normalization witness, recording the displayed integral row that makes the kernel total mass equal to the certified unit.*
5. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row available to later approximation packets after the finite kernel and its normalization have been exposed.*
6. *P is the positivity ledger, listing the square-modulus presentation, nonnegative coefficient witnesses, finite averaging denominator, and nonnegative-kernel stability entries used by the carrier.*
7. *T records componentwise* hsame *transport for coefficient windows, phase samples, trigonometric polynomial rows, integral witnesses, uniform-limit handoff rows, positivity ledgers, continuation replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a phase and coefficient-window request through F, S, A, P, then through I to the normalized kernel and through U to the approximation-facing handoff.*
9. *Q is the* Pkg *provenance over* $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{RealAnalytic}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FejerKernel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, S, A, I, U, P, T, C, Q, N$. It has no coordinate for Haar measure, an ambient quotient of functions, host-level almost-everywhere equality, or an external Fourier convergence principle.

Lean4 ✓ `BEDC.Derived.FejerKernelUp.FejerKernelTasteGate single carrier alignment`

Theorem 1939.2 (Fejer kernel NameCert obligations)**.** *For every accepted* $\mathsf{FejerKernel}^{\uparrow}$ *carrier $K = (F, S, A, I, U, P, T, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FejerKernel}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier admission for the displayed Fourier coefficient-window, circle phase-sample, real-analytic polynomial, integral-normalization, uniform-limit handoff, positivity-ledger, transport, continuation, provenance, and naming rows;*
2. *classifier transport, requiring* hsame *comparison to move componentwise through $F, S, A, I, U, P, T, C, Q, N$ without replacing the displayed finite kernel by host function equality;*
3. *finite positivity and nonnegative-kernel stability, requiring the square-modulus presentation and triangular-weight ledger in P to be preserved by the displayed finite algebra rows;*
4. *integral-normalization handoff, requiring the unit-mass witness in I to be read before any* $\mathsf{UniformLimit}^{\uparrow}$ *approximation consumer receives the handoff row U;*
5. *ledger non-escape, requiring every coefficient window, phase sample, sine/cosine polynomial row, normalization witness, positivity entry, transport row, continuation replay, and provenance row to remain inside the same finite packet.*

These obligations certify a positive finite summability kernel over the named source rows, not a host convergence theorem.

Proof. Project the carrier of Definition 1939.1. Carrier admission is exactly the finite inventory $F, S, A, I, U, P, T, C, Q, N$, so the certificate has no hidden coordinate for a completed function space, a Haar-measure object, or an external convergence theorem.

Classifier transport is componentwise. The $\mathsf{Fourier}^{\uparrow}$ row transports coefficient windows, the $\mathsf{S}^{1\uparrow}$ row transports phase samples, the $\mathsf{RealAnalytic}^{\uparrow}$ row transports the sine/cosine polynomial data, the $\mathsf{Integral}^{\uparrow}$ row transports the normalization witness, and the $\mathsf{UniformLimit}^{\uparrow}$ row transports only the handoff exposed by the earlier rows.

The positivity clause is read from the displayed ledger P. The square-modulus presentation and triangular weights are finite trigonometric-polynomial algebra data, so nonnegativity is stable under the same componentwise transport that preserves F, S, A.

The integral-normalization handoff is then forced through I. A later approximation packet may read U only after the normalization row has displayed the unit-mass witness for the same finite kernel.

Finally, T, C, Q, N preserve, replay, source, and name exactly the displayed route. Since every analytic datum used by the kernel has already appeared in the carrier rows, the ledger cannot be used to recover an unrecorded measure, quotient, or convergence principle. □

Falsifiable prediction: Every $\mathsf{FejerKernel}^{\uparrow}$ approximation-facing read factors through Fourier coefficient windows, SOne phase samples, RealAnalytic sine/cosine polynomial rows, an Integral normalization witness, a UniformLimit handoff, a positivity ledger, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fourier, s1, real-analytic, integral, uniformlimit: $\mathsf{FejerKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries triangular Fourier coefficient windows, circle phase samples, real-analytic trigonometric polynomial rows, integral normalization, uniform-limit handoff, positivity ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FejerKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{FejerKernel}^{\uparrow}$ packet is generated from finite Fourier coefficient windows with triangular weights, displayed $\mathsf{S}^{1\,\uparrow}$ phase samples, $\mathsf{RealAnalytic}^{\uparrow}$ sine/cosine polynomial rows, an $\mathsf{Integral}^{\uparrow}$ normalization witness, a $\mathsf{UniformLimit}^{\uparrow}$ handoff row, a positivity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FejerKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1939.1 and Definition 1939.2 seed the finite positive summability-kernel route from Fourier coefficient windows to constructive approximation-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FejerKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Haar measure, quotient function spaces, host Fourier convergence, pointwise convergence, uniform convergence of Fourier means, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite positivity stability, integral-normalization handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.FejerKernelUp.

Chapter 1940

A Concrete Naming Certificate for $\mathsf{Parseval}^{\uparrow}$

Following Parseval's identity in Fourier analysis, the $\mathsf{Parseval}^{\uparrow}$ packet records only the finite energy-identity surface that can be inspected before any Plancherel, inversion, or convergence theorem is claimed. It sits over the finite Fourier observation layer of chapter 124, the circle source of chapter 58, the pairing row of chapter 86, the integral energy row of chapter 123, and the real seal of chapter 38. The certificate names the displayed equality between finite coefficient energy ledgers and finite integral-energy ledgers; it does not identify host functions or complete a quotient coefficient space.

Definition 1940.1 (Parseval carrier)**.** *A* $\mathsf{Parseval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, S, I, E, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row, listing the finite modes, coefficient probes, truncation window, and coefficient-energy sums under inspection.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, exposing only the circle phase samples and classifier data used by the Fourier observations.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row, recording the finite pairing witnesses that compare coefficient rows and source rows.*
4. *E is the* $\mathsf{Integral}^{\uparrow}$ *energy row, storing the displayed integral-energy observation paired with the same finite source window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficient-energy sums and the integral-energy observation.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for finite summation, pairing, integral readback, and energy-comparison tolerances.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Fourier window, circle source, inner-product pairing, integral-energy row, regular readback, and dyadic tolerance ledger have all been displayed.*
8. *H stores componentwise* hsame *transport for the coefficient windows, circle source, pairing row, integral-energy row, readback row, tolerance ledger, real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from coefficient-window requests through F, S, I, E, R, D and then to the* $\mathsf{Real}^{\uparrow}$ *seal L.*
10. *P is the* Pkg *provenance over* $\mathsf{Parseval}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, S, I, E, R, D, L, H, C, P, N$. It has no coordinate for a host Fourier transform, completed Hilbert space, quotient coefficient class, selected limit, pointwise equality of functions, or ambient Plancherel theorem.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalTasteGate single carrier alignment`

Theorem 1940.2 (Parseval NameCert obligations)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier $K = (F, S, I, E, R, D, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Fourier coefficient-window,* $\mathsf{S}^{1\uparrow}$ *source,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing,* $\mathsf{Integral}^{\uparrow}$ *energy,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *coefficient-window exactness, requiring every coefficient-energy read to be attached to the modes and probes displayed in F and to the source row S;*
3. *finite partial-energy ledger accounting, requiring the coefficient-energy sum and the pairing witnesses in I to be read before the integral row is compared;*
4. *integral-energy handoff, requiring E to consume the same finite source and pairing rows before any real-facing energy comparison reaches L;*
5. *dyadic tolerance accounting, requiring D to record the finite summation, pairing, integral readback, and comparison tolerances used by the energy identity surface;*
6. *Real seal non-escape, requiring L to seal only the displayed energy comparison and not a host function equality or completed norm equality;*
7. *componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

These obligations certify only the finite Parseval energy-identity surface over the named source rows. They do not close Plancherel, inversion, convergence, Hilbert-space completeness, host function equality, quotient coefficient spaces, or Lean verification.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalNameCertObligations`

Proof. Project the carrier of Definition 1940.1. Carrier admission is exactly the finite inventory $F, S, I, E, R, D, L, H, C, P, N$, so no hidden analytic coordinate is available.

Coefficient-window exactness is read from F together with the $\mathsf{S}^{1\uparrow}$ source row S. The pairing row I and integral row E then compare only the displayed finite energy observations, while R and D expose the regular readback and dyadic tolerance ledger used for that comparison.

The $\mathsf{Real}^{\uparrow}$ seal L is downstream of those rows. Thus the endpoint is a seal for the finite energy ledger, not a proof of host function equality, Hilbert-space completeness, or a quotient coefficient-space theorem.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since every readable coordinate has already been listed, the certificate cannot export Plancherel, inversion, convergence closure, or Lean verification. □

Theorem 1940.3 (Parseval root energy carrier admission)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, any root energy consumer must admit the packet through all concrete rows* $F, S, I, E, R, D, L, H, C, P, N$ *of Definition 1940.1 before it may read the finite energy-identity surface certified by Definition 1940.2. Carrier admission is therefore a displayed NameCert route through Fourier coefficient windows, SOne source rows, InnerProduct pairing witnesses, Integral energy rows, RegSeqRat readback, DyadicRatCore tolerances, the Real seal, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 1940.1. The rows $F, S, I, E, R, D, L, H, C, P, N$ are the entire accepted inventory, so a root consumer has no separate analytic coordinate to inspect.

Apply Definition 1940.2. Its carrier-admission clause lists the same Fourier, source, pairing, integral, readback, tolerance, seal, transport, replay, provenance, and naming rows as the local NameCert surface.

Thus root energy admission is exactly the displayed packet route. The admission step does not introduce Plancherel, inversion, Hilbert completeness, quotient coefficient spaces, or host function equality. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootEnergyCarrierAdmission`

Theorem 1940.4 (Parseval root Fourier-energy budget)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every root Fourier-energy comparison must display the* $\mathsf{Fourier}^{\uparrow}$ *coefficient windows in* F*, the* $\mathsf{S}^{1\uparrow}$ *source rows in* S*, the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row* I*, the* $\mathsf{Integral}^{\uparrow}$ *energy row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the* $\mathsf{Real}^{\uparrow}$ *seal* L *before the local* $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$ *row* N *can be consumed. The comparison is therefore a finite BEDC route through* $F, S, I, E, R, D, L, H, C, P, N$*, and it refuses Plancherel, Fourier inversion, completed Hilbert norm equality, quotient coefficient classes, and host function equality.*

Proof. Project the carrier of Definition 1940.1. The coefficient-energy side is visible only in the Fourier window row F, and the source and pairing rows are exactly S and I.

Apply the NameCert obligations of Definition 1940.2. The coefficient-window exactness, finite partial-energy ledger accounting, integral-energy handoff, dyadic tolerance accounting, and Real seal non-escape clauses require F, S, I, E, R, D, L before the structural rows can name the comparison.

Use root carrier admission from Definition 1940.3. It forces any root energy consumer through the complete displayed packet $F, S, I, E, R, D, L, H, C, P, N$ before the finite energy-identity surface is read.

Since none of those carrier rows stores Plancherel, Fourier inversion, a completed Hilbert norm equality, quotient coefficient classes, or host function equality, the root Fourier-energy budget cannot export any of those analytic claims. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootFourierEnergyBudget`

Theorem 1940.5 (Parseval integral-energy handoff obligation)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, an energy-identity consumer must expose Fourier coefficient windows,* $\mathsf{S}^{1\uparrow}$ *source rows,* $\mathsf{InnerProduct}^{\uparrow}$ *pairings, the* $\mathsf{Integral}^{\uparrow}$ *energy row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any energy identity is consumed. The visible handoff is the finite route*

$$F, S, I \longmapsto E, R, D \longmapsto L, H, C, P, N.$$

It exports no Plancherel theorem, Fourier inversion theorem, Hilbert completion, quotient coefficient space, host function equality coordinate, infinite coefficient summation, or completed ambient norm.

Proof. Project the carrier of Definition 1940.1. The rows $F, S, I, E, R, D, L, H, C, P, N$ are the complete accepted inventory, so the energy consumer first sees only finite coefficient windows, circle-source rows, pairings, integral-energy data, readback, tolerances, the Real seal, and structural certificate rows.

Invoke Definition 1940.2. Its carrier-admission, finite partial-energy ledger, integral-energy handoff, dyadic tolerance accounting, and Real seal non-escape clauses require F, S, I, E, R, D, L before the energy comparison reaches the structural rows.

It remains to inspect what the packet refuses to export. The same obligations identify no coordinate for Plancherel, inversion, Hilbert completion, quotient coefficient spaces, host function equality, infinite coefficient summation, or completed ambient norms, so no downstream energy identity can consume any of those objects through $\mathsf{Parseval}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalIntegralEnergyHandoffObligation`

Theorem 1940.6 (Parseval root finite coefficient ledger)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the root coefficient ledger visible to downstream consumers is finite and is read only through the rows* $F, S, I, E, R, D, L, H, C, P, N$ *of Definition 1940.1. The coefficient-window row* F*, source row* S*, pairing row* I*, integral row* E*, readback row* R*, and tolerance row* D *must all be exposed before* L, H, C, P, N *can seal, transport, replay, source, and name the finite comparison required by Definition 1940.2.*

Proof. Start from the coefficient-window exactness and finite partial-energy ledger clauses of Definition 1940.2. They attach coefficient reads to F and S, and require the pairing witnesses in I before the integral row is compared.

The carrier definition then supplies the remaining visible ledger rows. The row E stores the integral-energy observation, R stores regular readback, and D stores the dyadic numerical budget for the same finite comparison.

Only after these rows are displayed can L, H, C, P, N seal and preserve the route. Since the route is finite at each row, it cannot be replaced by Plancherel, Fourier inversion, Hilbert completion, quotient coefficient spaces, or host function equality. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootFiniteCoefficientLedger`

Theorem 1940.7 (Parseval root orthogonality ledger)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every root orthogonality read must expose the* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row* F*, the* $\mathsf{S}^{1\uparrow}$ *source row* S*, and the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row* I *before it can be compared with the* $\mathsf{Integral}^{\uparrow}$ *energy row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L*. The orthogonality ledger is therefore a finite NameCert route through* $F, S, I, E, R, D, L, H, C, P, N$*, not a complete orthonormal basis, Plancherel theorem, Fourier inversion theorem, quotient coefficient space, or host equality of circle functions.*

Proof. Project the carrier of Definition 1940.1. The only rows that can support an orthogonality read are the coefficient windows F, circle source S, and pairing witnesses I.

Apply Definition 1940.6. It forces the root coefficient ledger to display F, S, I, E, R, D before the seal and structural rows can preserve the comparison.

Finally use the NameCert obligations of Definition 1940.2. The coefficient-window exactness and finite partial-energy ledger clauses keep the orthogonality read inside the displayed Fourier, SOne, and InnerProduct rows, so no host basis theorem or quotient coefficient space is exported. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootOrthogonalityLedger`

Theorem 1940.8 (Parseval root coefficient integral budget)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the root coefficient-to-integral budget is exhausted by the finite route from coefficient windows and source samples through pairing witnesses to the integral-energy row, regular readback, dyadic tolerance ledger, and Real seal. In symbols, the readable route is*

$$F, S, I \longmapsto E, R, D \longmapsto L, H, C, P, N.$$

No separate coefficient budget for infinite summation, completed norms, quotient coefficient equality, Plancherel, inversion, or host function equality is available outside this displayed route.

Proof. Start with the root finite coefficient ledger of Definition 1940.6. It already makes F, S, I, E, R, D visible before the seal, transport, replay, provenance, and naming rows are consumed.

Use the NameCert obligations of Definition 1940.2. The finite partial-energy ledger accounting row requires the pairing witnesses in I before the integral row E is compared, and the dyadic tolerance accounting row assigns the numerical budget to D.

The Real seal row L and the structural rows H, C, P, N then preserve only this displayed budget. Since no carrier coordinate names an infinite coefficient series, completed norm, quotient equality, Plancherel theorem, inversion theorem, or host equality, those objects cannot enter the root budget. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootCoefficientIntegralBudget`

Theorem 1940.9 (Parseval root Real energy seal boundary)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the root* $\mathsf{Real}^{\uparrow}$ *energy seal boundary is reached only after the Fourier coefficient windows, SOne source row, InnerProduct pairing row, Integral energy row, RegSeqRat readback row, and DyadicRatCore tolerance ledger have all been displayed. The seal* L *records the finite energy comparison preserved by* H, C, P, N *and refuses Plancherel, Fourier inversion, convergence closure, Hilbert-space completeness, quotient coefficient spaces, infinite coefficient summation, and host function equality.*

Proof. Read the carrier and the Real seal non-escape clause. By Definition 1940.1 and Definition 1940.2, L is downstream of F, S, I, E, R, D and seals only the displayed finite energy comparison.

Apply Definition 1940.7 and Definition 1940.8. The first fixes the Fourier, SOne, and InnerProduct rows before any orthogonality-facing read, while the second exhausts the coefficient-to-integral budget through E, R, D.

Finally project the structural rows H, C, P, N. They transport, replay, source, and name the same sealed comparison without adding analytic coordinates, so the root Real boundary cannot export the refused host theorems, quotient objects, infinite summations, or completed norms. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootRealEnergySealBoundary`

Theorem 1940.10 (Parseval root Real seal nonescape)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the root* $\mathsf{Real}^{\uparrow}$ *seal* L *is nonescaping: it seals only the displayed finite energy comparison built from* F, S, I, E, R, D *and preserved by* H, C, P, N*. By Definition 1940.1 and Definition 1940.2,* L *exports no Plancherel theorem, inversion theorem, Hilbert-completeness claim, quotient coefficient space, or host function equality.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootRealSealNonescape`

Proof. Read the Real seal row in Definition 1940.1. The seal L is downstream from the coefficient windows, source rows, pairing witnesses, integral observation, regular readback, and dyadic tolerance ledger.

Apply the Real seal non-escape clause of Definition 1940.2. It states that L seals only the displayed energy comparison and not host equality or completed norm equality.

The structural rows H, C, P, N preserve and name that same finite comparison. Therefore the root seal cannot export Plancherel, inversion, Hilbert completeness, quotient coefficient spaces, or host function equality. □

Theorem 1940.11 (Parseval coefficient integral handoff)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every coefficient-facing energy comparison must read the finite route*

$$F \longmapsto S \longmapsto I \longmapsto E$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal* L *is available. The route records Fourier coefficient windows, the SOne source row, the InnerProduct pairing row, and the Integral energy row as one finite handoff.*

Proof. Project the carrier of Definition 1940.1. The rows F, S, I, E are displayed before the readback row, tolerance ledger, and terminal seal.

Apply the NameCert obligations of Definition 1940.2. Coefficient-window exactness reads F with the source row S, finite partial-energy ledger accounting reads the pairing witnesses in I, and integral-energy handoff reads E from the same finite source window.

Thus a coefficient-facing energy comparison reaches L only after the route $F \longmapsto S \longmapsto I \longmapsto E$ has been displayed. The theorem records the finite Parseval handoff and does not add Plancherel, inversion, convergence, or Hilbert completeness. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalCoefficientIntegralHandoff`

Theorem 1940.12 (Parseval dyadic tolerance ledger)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *accounts for exactly the finite summation, pairing, integral readback, and energy-comparison tolerances used by the displayed Parseval surface. No tolerance for an infinite series, completed norm, quotient coefficient class, or host function equality is available outside* D*.*

Proof. Project the carrier. The only tolerance coordinate is D, and it is listed after the finite coefficient windows, source row, pairing row, integral row, and regular readback.

Use the dyadic tolerance accounting row in Definition 1940.2. It assigns D to finite summation, pairing, integral readback, and comparison tolerances, so every numerical budget for the energy comparison is read through the same DyadicRatCore ledger.

Since the carrier has no coordinate for infinite-series convergence, completed Hilbert norms, quotient coefficient spaces, or host function equality, no additional tolerance source can be exported by the packet. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalDyadicToleranceLedger`

Definition 1940.13 (Parseval inner-product carrier route)**.** *For an accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the inner-product carrier route is the displayed subpacket*

$$(F, S, I, E, R, D, L, H, C, P, N)$$

read as Fourier coefficient windows, $\mathsf{S}^{1\uparrow}$ *source rows,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing witnesses,* $\mathsf{Integral}^{\uparrow}$ *energy observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$ *naming. It contains no row for completed Hilbert norms, quotient coefficient classes, selected representatives, Plancherel, inversion, or host function equality.*

Lean4 (definition) `BEDC.Derived.ParsevalUp.ParsevalInnerProductCarrierRoute`

Theorem 1940.14 (Parseval inner-product NameCert obligations)**.** *Every consumer of the inner-product carrier route in Definition 1940.13 must display all rows* $F, S, I, E, R, D, L, H, C, P, N$ *before it reads an energy comparison. The rows* F, S, I *bind Fourier coefficient windows, circle-source samples, and finite pairing witnesses; the rows* E, R, D, L *bind integral-energy observation, regular rational readback, dyadic tolerance accounting, and the terminal Real seal; the rows* H, C, P, N *preserve, replay, source, and name the same route. The obligation does not claim Plancherel, inversion, completed Hilbert norms, quotient coefficient classes, or host function equality.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalInnerProductNameCertObligations`

Proof. Start with the carrier definition Definition 1940.1. Its accepted inventory is exactly $F, S, I, E, R, D, L, H, C, P$ and Definition 1940.13 reads no additional coordinate.

Apply Definition 1940.2. Carrier admission, coefficient-window exactness, finite partial-energy ledger accounting, integral-energy handoff, dyadic tolerance accounting, Real seal non-escape, and structural transport already require each listed row.

The refused analytic endpoints have no row in the accepted packet. Therefore a consumer of the inner-product route receives only the displayed NameCert obligations and cannot import Plancherel, inversion, a completed Hilbert norm, a quotient coefficient class, or host equality of functions. □

Theorem 1940.15 (Parseval L2 energy identity route)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the only L2-facing energy identity route is the finite path from* F, S, I *through* E, R, D *to* L*, then through* H, C, P, N*. The route compares coefficient energy and integral energy only through displayed* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. It does not assert completed Hilbert-space equality, Plancherel, Fourier inversion, quotient coefficient equality, or host function equality.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalL2EnergyIdentityRoute`

Proof. Read the route from Definition 1940.13. The coefficient side is F, S, I, the integral side is E, and the readback and tolerance rows are R, D before the Real seal L.

Use Definition 1940.14. It forces every energy comparison to display the full row family before a consumer may read the comparison.

Finally use Definition 1940.12. The tolerance ledger is finite and attached to summation, pairing, integral readback, and energy comparison only. Hence the L2-facing phrase names this finite route and not a completed Hilbert-space identity or host theorem. □

Theorem 1940.16 (Parseval Fourier Real readback boundary)**.** *Every* $\mathsf{Fourier}^{\uparrow}$ *consumer that reads a* $\mathsf{Real}^{\uparrow}$ *endpoint through* $\mathsf{Parseval}^{\uparrow}$ *must pass through the displayed carrier rows* $F, S, I, E, R, D, L, H, C, P, N$*. The readback boundary is the route*

$$F, S, I, E, R, D \longmapsto L \longmapsto H, C, P, N.$$

It is a finite Fourier-to-Real energy seal through $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, not a Plancherel theorem, inversion theorem, quotient coefficient class, completed Hilbert norm, or host function equality.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFourierRealReadbackBoundary`

Proof. Project the accepted carrier. A Fourier-facing consumer first sees coefficient windows in F, source samples in S, and finite pairing witnesses in I.

Apply Definition 1940.15. The route reaches the integral row, regular readback, tolerance ledger, and Real seal only through the displayed finite path.

The structural rows H, C, P, N preserve that same path after the seal is read. Because the carrier lacks coordinates for Plancherel, inversion, quotient coefficient classes, completed norms, or host equality, none of those endpoints can cross the Fourier-to-Real boundary. □

Theorem 1940.17 (Parseval SOne orthogonality handoff)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, any orthogonality-facing read from the* $\mathsf{S}^{1\uparrow}$ *row* S *is handed to* $\mathsf{InnerProduct}^{\uparrow}$ *only through the finite pairing row* I *and then to* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *through* E, R, D, L*. The handoff records displayed circle-source rows and pairing witnesses; it does not claim a complete orthonormal basis, Fourier inversion, Plancherel, quotient coefficient equality, or equality of host functions on the circle.*

Proof. Start with Definition 1940.1. The circle source is visible only as S, and the pairing witnesses are visible only as I after the Fourier coefficient row F has fixed the finite observation window.

Use Definition 1940.14. It makes $F, S, I, E, R, D, L, H, C, P, N$ mandatory before any energy comparison is consumed, so an orthogonality-facing read cannot leave the displayed carrier.

Combine this with Definition 1940.16. The downstream Real endpoint is reached only through E, R, D, L and structural preservation. Thus the handoff is finite and local to the packet, with no complete basis, inversion theorem, Plancherel theorem, quotient class, or host circle-function equality exported. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalSoneOrthogonalityHandoff`

Theorem 1940.18 (Parseval Real seal nonescape)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal* L *seals only the displayed finite energy comparison built from* F, S, I, E, R, D*. It excludes Plancherel, inversion, convergence, Hilbert completeness, host function equality, and quotient coefficient spaces.*

Proof. Project the carrier. The seal L is downstream of the Fourier window, SOne source, InnerProduct pairing, Integral energy row, RegSeqRat readback, and DyadicRatCore tolerance ledger.

Apply Definition 1940.2. Its Real seal non-escape row states that L seals only the displayed energy comparison and not a host function equality or completed norm equality.

The exclusions follow from the carrier inventory and the same obligation rows. Plancherel, inversion, convergence, Hilbert completeness, host function equality, and quotient coefficient spaces have no readable coordinate in K, so they cannot escape through L. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRealSealNonescape`

Theorem 1940.19 (Parseval Fourier energy window)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every accepted Parseval energy read first exposes the finite* $\mathsf{Fourier}^{\uparrow}$ *coefficient rows in* F*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows in* D*, and the* $\mathsf{Integral}^{\uparrow}$ *energy rows in* E *before any equality ledger is consumed. The read is a finite* BHist *route through coefficient windows, source rows, tolerance accounting, integral observation, transport, replay, provenance, and local naming; it is not a host Fourier-transform equality or an infinite coefficient summation claim.*

Proof. Project the carrier of Definition 1940.1. The coefficient-window row F, integral-energy row E, and tolerance ledger D are displayed before the terminal real seal and before the local name can export an energy read.

Apply Definition 1940.2. Coefficient-window exactness exposes F, integral-energy handoff exposes E, and dyadic tolerance accounting exposes D as the finite numerical budget for the comparison.

The equality ledger is therefore consumed only after those rows are visible. Since the carrier has no coordinate for a host Fourier transform or an infinite coefficient summation, the exported read remains the finite BHist route stated above. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFourierEnergyWindow`

Theorem 1940.20 (Parseval root Fourier and inner-product route)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every root-level consumer that reads a Fourier energy comparison must first expose the* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row* F*, the* $\mathsf{S}^{1\uparrow}$ *source row* S*, and the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row* I *before the comparison can reach the integral, readback, tolerance, and Real-seal rows. The route is finite:*

$$F, S, I \longmapsto E, R, D, L, H, C, P, N.$$

It records only displayed coefficient windows, circle-source samples, finite pairing witnesses, and the downstream NameCert route; it refuses Plancherel, inversion, Hilbert completeness, quotient coefficient spaces, and host function equality.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootFourierInnerProductRoute`

Proof. Project the carrier of Definition 1940.1. The rows F, S, I are the only Fourier, circle-source, and pairing coordinates visible to a root consumer.

Apply Definition 1940.2. Its coefficient-window exactness and finite partial-energy ledger clauses require coefficient reads to remain attached to F, S and require the pairing witnesses in I before the integral row is compared.

The downstream rows E, R, D, L, H, C, P, N preserve that same finite route. Since the carrier has no coordinate for a host transform, quotient coefficient class, Hilbert completion, Plancherel theorem, inversion theorem, or host function equality, none of those objects can be exported by this root route. □

Theorem 1940.21 (Parseval root integral and regular-readback handoff)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every root-level integral-energy read must expose the* $\mathsf{Integral}^{\uparrow}$ *row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before it reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L*. Thus the root handoff is the finite route*

$$E, R, D \longmapsto L \longmapsto H, C, P, N$$

after the Fourier, SOne, and InnerProduct rows have already been displayed. The handoff does not assert convergence of coefficient series, completed norm equality, quotient coefficient equality, Plancherel, inversion, or host equality of functions.

Proof. Start from Definition 1940.1. The integral-energy observation is stored in E, its regular rational readback is stored in R, and the numerical budget is stored in D.

Use Definition 1940.2. Integral-energy handoff requires E to consume the same finite source and pairing rows before the Real-facing comparison reaches L, and dyadic tolerance accounting confines the comparison budget to D.

The seal row L and the structural rows H, C, P, N therefore receive only the displayed integral/readback/tolerance packet. They cannot add convergence, completed Hilbert norms, quotient coefficient equality, Plancherel, inversion, or host function equality because none of those rows appears in the accepted carrier. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootIntegralRegSeqRatHandoff`

Theorem 1940.22 (Parseval root SOne and Real-seal boundary)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the root route from the* $\mathsf{S}^{1\uparrow}$ *source row* S *to the* $\mathsf{Real}^{\uparrow}$ *seal row* L *is mediated by the displayed Fourier, InnerProduct, Integral, RegSeqRat, and DyadicRatCore rows. In particular, the only exposed boundary is*

$$S, F, I, E, R, D \longmapsto L.$$

The Real seal records a finite energy-comparison endpoint and not a host circle-function equality, Plancherel theorem, Fourier inversion theorem, Hilbert-completeness claim, quotient coefficient space, or ambient completed norm.

Proof. Project the rows of Definition 1940.1. The $\mathsf{S}^{1\uparrow}$ source is visible only as S, and the $\mathsf{Real}^{\uparrow}$ endpoint is visible only as L after the coefficient, pairing, integral, readback, and tolerance rows have been displayed.

Apply Definition 1940.2. The Real seal non-escape clause states that L seals only the displayed energy comparison and not a host equality or completed norm equality.

Combine this with the Fourier energy window of Definition 1940.19. The source-to-seal boundary is therefore the finite route through S, F, I, E, R, D, L, preserved by H, C, P, N, with no coordinate for the refused analytic completions or host equalities. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootSoneRealSealBoundary`

Theorem 1940.23 (Parseval inner-product norm scope)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, every norm-squared comparison is scoped through the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row* I *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* L*. The packet records finite pairing witnesses, regular readback, dyadic tolerance accounting, componentwise transport, continuation replay, provenance, and local naming; it does not import Hilbert-space completion, quotient coefficient equality, or a completed norm theorem.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalInnerProductNormScope`

Proof. Read the norm-facing comparison from the carrier. The only pairing coordinate is I, and the only real-facing endpoint is L after the readback and tolerance rows R, D.

Use Definition 1940.2. The finite partial-energy ledger accounting row requires the pairing witnesses in I before the integral row is compared, and the Real seal non-escape row limits L to the displayed energy comparison.

Thus the norm-squared comparison has no path to Hilbert-space completion or quotient coefficient equality. The structural rows H, C, P, N preserve and name the finite route without adding a completed norm theorem. □

Theorem 1940.24 (Parseval S1 basis ledger)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P$ *the circle-source basis rows are read through the* $\mathsf{S}^{1\uparrow}$ *row* S *and the finite* $\mathsf{Fourier}^{\uparrow}$ *observation windows in* F*. A consumer may inspect the displayed modes, phase samples, coefficient probes, pairing witnesses, integral-energy rows, tolerance ledger, transport, replay, provenance, and local naming, but it cannot replace them by host functions on an ambient circle.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalSoneBasisLedger`

Proof. Project the source-facing rows. The carrier exposes the circle source as S and the Fourier observation windows as F; no ambient circle function space is stored in the packet.

Apply Definition 1940.11. The energy route reads F and S before the pairing and integral rows I, E, so the displayed modes and phase samples remain attached to the finite observation window.

Finally use Definition 1940.18. The terminal seal can export only the displayed finite energy comparison, and the structural rows H, C, P, N name the same data. Host functions on an ambient circle therefore have no carrier coordinate through which to enter the ledger. □

Theorem 1940.25 (Parseval root obligation surface)**.** *Every root-level* $\mathsf{Parseval}^{\uparrow}$ *energy consumer must read the finite route*

$$F \longmapsto S \longmapsto I \longmapsto E \longmapsto D \longmapsto R \longmapsto L \longmapsto H, C, P, N$$

through $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before any energy equality is consumed. The route displays coefficient windows, circle-source rows, pairing witnesses, integral-energy observations, dyadic tolerance budgets, regular rational readback, the terminal real seal, transport, replay, provenance, and local naming. It does not import Plancherel, inversion, Hilbert completeness, quotient coefficient spaces, host function equality, or an ambient completed norm.*

Proof. Project the carrier of Definition 1940.1. The rows F, S, I, E, D, R, L are the only analytic coordinates available before the structural rows are read.

Apply Definition 1940.11 to expose the Fourier coefficient windows, circle source, inner-product pairing, and integral-energy row. Then apply Definition 1940.12 to force the numerical budget through D and the regular readback through R.

Finally use Definition 1940.18. The seal L is only the displayed finite energy comparison, and H, C, P, N preserve and name that same route. The refused analytic objects therefore have no carrier coordinate from which to enter the root surface. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootObligationSurface`

Theorem 1940.26 (Parseval root coefficient stability)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, coefficient-energy stability is witnessed only after the coefficient rows in* $\mathsf{Fourier}^{\uparrow}$*, the source rows in* $\mathsf{S}^{1\uparrow}$*, the pairing rows in* $\mathsf{InnerProduct}^{\uparrow}$*, the integral rows in* $\mathsf{Integral}^{\uparrow}$*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal have all been displayed. A stable coefficient row may be transported through* H, C, P, N*, but it cannot be replaced by a quotient coefficient space, host Fourier transform, Plancherel theorem, inversion theorem, or Hilbert-completion argument.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootCoefficientStability`

Proof. Start from the coefficient-facing route in Definition 1940.19. It already requires the finite Fourier coefficient rows, the integral-energy rows, and the dyadic tolerance ledger before an equality ledger is consumed.

Attach the circle source and pairing rows using Definition 1940.24 and Definition 1940.23. These rows keep coefficient stability tied to the displayed source and pairing witnesses rather than to a host function space.

The regular readback and terminal seal are then supplied by the carrier rows R and L. Transport, replay, provenance, and local naming can preserve the stable coefficient route, but they cannot add quotient coefficient spaces, Plancherel, inversion, or Hilbert completeness. □

Theorem 1940.27 (Parseval finite Fourier carrier admission)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, any Parseval carrier read must expose the finite* $\mathsf{Fourier}^{\uparrow}$ *coefficient rows in* F*, the* $\mathsf{S}^{1\uparrow}$ *basis and source rows in* S*, the finite* $\mathsf{Integral}^{\uparrow}$ *energy rows in* E*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* L *before the energy equality is consumed. The exposed route is the same carrier route certified by Definition 1940.1, Definition 1940.2, and Definition 1940.26; no downstream consumer may read a coefficient equality while omitting any of* $F, S, I, E, R, D, L, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFiniteFourierCarrierAdmission`

Proof. Project the carrier in Definition 1940.1. The Fourier, circle-source, integral-energy, readback, tolerance, Real-seal, and structural coordinates are exactly $F, S, I, E, R, D, L, H, C, P, N$.

Apply Definition 1940.2. Its carrier-admission, coefficient-window exactness, finite partial-energy ledger, integral-energy handoff, dyadic tolerance, and Real-seal clauses require these rows before the finite energy comparison can be read.

Finally use Definition 1940.26. Coefficient stability remains tied to the displayed Fourier, SOne, InnerProduct, Integral, DyadicRatCore, RegSeqRat, and Real rows, so a carrier read that omits any listed row is outside the packet. □

Theorem 1940.28 (Parseval inner-product energy ledger)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the coefficient-side inner-product energy ledger and the integral-side energy ledger share the same* $\mathsf{BHist/hsame/Cont/Pkg}$ *route. The coefficient side is read through* F, S, I*, the integral side is read through* E, R, D, L*, and both sides are transported, replayed, sourced, and named only through* H, C, P, N*.*

Proof. Read the finite route from Definition 1940.25. It forces every root energy consumer through the same displayed carrier rows before any equality is consumed.

For the coefficient side, apply Definition 1940.20. The Fourier coefficient windows, SOne source rows, and InnerProduct pairing witnesses are the only coefficient-side energy rows available.

For the integral side, apply Definition 1940.21 and Definition 1940.18. The integral-energy row, regular readback, dyadic tolerance ledger, and Real seal are downstream from the same source and pairing data.

The structural rows are shared because the carrier has only one H, C, P, N family. Hence both ledgers use the same $\mathsf{BHist/hsame/Cont/Pkg}$ route and cannot split into independent coefficient and integral certificates. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalInnerProductEnergyLedger`

Theorem 1940.29 (Parseval Real-seal refusal boundary)**.** *Every accepted* $\mathsf{Parseval}^{\uparrow}$ *packet exports no Plancherel theorem, quotient function equality, selected convergence datum, host* L^2 *space, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its Real-facing boundary is only the displayed finite seal* L *over* $F, S, I, E, R, D, H, C, P, N$*, as constrained by Definition 1940.18 and Definition 1940.30.*

Proof. Start with the classifier in Definition 1940.1. It compares accepted carriers only through the displayed rows and has no coordinate for Plancherel, quotient functions, selected convergence, host L^2 spaces, or ambient completeness.

Apply Definition 1940.18. The Real seal L seals only the finite energy comparison built from the Fourier, SOne, InnerProduct, Integral, RegSeqRat, and DyadicRatCore rows.

Finally use Definition 1940.30. The root Real read stays downstream from the same displayed carrier route, so none of the refused analytic objects can cross the Real-seal boundary. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRealSealRefusalBoundary`

Theorem 1940.30 (Parseval root Real energy nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *energy read exported by* $\mathsf{Parseval}^{\uparrow}$ *is downstream from* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows in the displayed carrier. The exported seal is a finite energy-comparison endpoint only: it refuses Plancherel, Fourier inversion, Hilbert completeness, quotient coefficient spaces, host function equality, infinite coefficient summation, and completed ambient norms.*

Proof. Read the root obligation surface of Definition 1940.25. It forces every energy consumer through the finite coefficient, source, pairing, integral, tolerance, and readback rows before L is available.

Now apply Definition 1940.26. Coefficient stability stays attached to the same displayed rows and cannot be transported into a quotient coefficient space or host Fourier transform.

The final endpoint is controlled by Definition 1940.18. Since L seals only the finite energy comparison, no exported Real read can supply Plancherel, inversion, Hilbert completeness, host function equality, infinite summation, or completed ambient norms. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootRealEnergyNonescape`

Theorem 1940.31 (Parseval root Fourier-integral energy exhaustion)**.** *Every root* $\mathsf{Parseval}^{\uparrow}$ *consumer that reads an energy identity must first expose Fourier coefficient rows,* $\mathsf{S}^{1\uparrow}$ *basis rows,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing rows, the* $\mathsf{Integral}^{\uparrow}$ *energy row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. In carrier notation for* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the identity is readable only after* $F, S, I, E, D, R, L, H, C, P, N$ *have all been displayed. No downstream Real or Integral consumer can replace this finite route by Plancherel, inversion, Hilbert completeness, quotient coefficient spaces, host function equality, infinite coefficient summation, or an ambient completed norm.*

Proof. Begin with the root carrier admission theorem Definition 1940.3. It requires the full displayed packet $F, S, I, E, R, D, L, H, C, P, N$ before a root energy consumer may read the finite energy-identity surface.

The finite coefficient side is then fixed by Definition 1940.6. The Fourier row F, the circle-source basis row S, the pairing row I, the integral-energy row E, and the tolerance row D must be visible before the structural rows can seal and preserve the comparison.

Use Definition 1940.11 and Definition 1940.12. These theorems force the coefficient-to-integral handoff through F, S, I, E and place all finite summation, pairing, integral readback, and comparison tolerances inside D.

Finally apply Definition 1940.30. The terminal Real read remains downstream from the same Fourier, pairing, integral, source, dyadic, and regular-readback rows, so the exhausted route cannot export any refused host theorem or completed ambient norm. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootFourierIntegralEnergyExhaustion`

Theorem 1940.32 (Parseval Fourier coefficient energy row)**.** *For every accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the Fourier coefficient-energy row exported to a downstream consumer is the displayed finite route through* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *carried by* $F, S, I, E, R, D, L, H, C, P, N$*. The row records coefficient windows, circle-source reads, pairing witnesses, integral-energy observations, regular rational readback, dyadic tolerances, and the terminal Real seal; it does not export an infinite Fourier coefficient sequence, quotient coefficient equality, Plancherel, inversion, or completed norm equality.*

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFourierCoefficientEnergyRow`

Proof. Project the carrier of Definition 1940.1. The coefficient-energy read starts at F and S, then reaches pairing and integral observations through I and E.

Use Definition 1940.31. It requires the coefficient, source, pairing, integral, tolerance, readback, Real-seal, transport, replay, provenance, and local naming rows before a root energy identity can be read.

Finally apply Definition 1940.18. The Real endpoint seals only the displayed finite comparison, so no infinite coefficient object, quotient equality, Plancherel theorem, inversion theorem, or completed norm equality can be read from the row. □

Theorem 1940.33 (Parseval finite partial-sum induction)**.** *Every finite partial-sum inspection admitted by an accepted* $\mathsf{Parseval}^{\uparrow}$ *carrier is inductive over the displayed Fourier window row* F*. The base inspection reads the empty coefficient window through the same* $\mathsf{S}^{1\uparrow}$ *source,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing,* $\mathsf{Integral}^{\uparrow}$ *energy,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, and* $\mathsf{Real}^{\uparrow}$ *seal rows, and the successor inspection extends only the finite ledger already transported and named by* H, C, P, N*. No step introduces countable summation, completed Hilbert structure, quotient coefficient equality, or host function equality.*

Proof. For the base window, read Definition 1940.6. The visible coefficient ledger is finite and already attached to $F, S, I, E, R, D, L, H, C, P, N$.

For the successor window, use the same ledger clause together with Definition 1940.32. Adding one displayed coefficient extends the finite Fourier row before the pairing, integral, readback, tolerance, and Real-seal rows are replayed.

The induction is confined by Definition 1940.31. Since each step remains inside the displayed carrier route, the induction cannot produce countable summation, Hilbert completeness, quotient coefficient equality, or host function equality. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFinitePartialSumInduction`

Theorem 1940.34 (Parseval finite energy induction readback)**.** *Every accepted* $\mathsf{Parseval}^{\uparrow}$ *packet whose finite partial-sum induction row is consumed must expose the Fourier coefficient window,* $\mathsf{S}^{1\uparrow}$ *source row,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row,* $\mathsf{Integral}^{\uparrow}$ *energy row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$ *row before the energy identity can be read. In carrier notation for* $K = (F, S, I, E, R, D, L, H, C, P, N)$*, the readback route is exactly*

$$F, S, I, E, R, D, L, H, C, P, N.$$

No consumer may replace this finite induction route by Plancherel, inversion, completed Hilbert norms, quotient coefficient equality, host function equality, infinite coefficient summation, or an ambient completed norm.

Proof. Start with the root energy decomposition of Definition 1940.35. It identifies the readable energy surface with the displayed carrier rows $F, S, I, E, R, D, L, H, C, P, N$ and refuses every analytic endpoint not present in the carrier.

Next use the Fourier coefficient energy row of Definition 1940.32. It exposes the finite coefficient window, circle-source row, pairing row, integral-energy observations, regular readback, dyadic tolerances, and Real seal before any downstream energy comparison is exported.

Now apply finite partial-sum induction, Definition 1940.33. The base and successor inspections stay inside the same finite Fourier window row and are transported only through the displayed structural rows.

Finally use the Real-seal refusal boundary of Definition 1940.29. Since the terminal seal exports only the finite comparison over the displayed rows, the consumed induction row cannot be read as Plancherel, inversion, Hilbert completion, quotient coefficient equality, host function equality, infinite summation, or completed ambient norm data. □

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalFiniteEnergyInductionReadback`

Theorem 1940.35 (Parseval root energy decomposition)**.** *Every accepted* $\mathsf{Parseval}^{\uparrow}$ *root energy read decomposes into the Fourier coefficient window* F*, the* $\mathsf{S}^{1\uparrow}$ *source row* S*, the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row* I*, the* $\mathsf{Integral}^{\uparrow}$ *energy row* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Real}^{\uparrow}$ *seal* L*, and the support rows* H, C, P, N *of Definition 1940.1. The decomposition exports only this displayed route*

$$F, S, I, E, R, D, L, H, C, P, N.$$

It exports no Plancherel theorem, Fourier inversion theorem, Hilbert completion, quotient coefficient equality, or infinite summation.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRootEnergyDecomposition`

Proof. Start with the finite partial-sum induction of Definition 1940.33. It keeps coefficient-energy inspection inside the finite Fourier window row F and transports each base or successor inspection through the same structural rows.

Next use the finite coefficient ledger and coefficient-integral handoff, Definition 1940.6 and Definition 1940.11. These attach F to the source row S, require the pairing row I, and route the comparison to the integral-energy row E before any energy read reaches the terminal seal.

Apply the dyadic tolerance ledger and Real seal nonescape, Definition 1940.12 and Definition 1940.30. They force regular readback and numerical tolerance through R, D and keep L as a finite energy seal rather than a completed analytic endpoint.

Finally read root Fourier-integral exhaustion from Definition 1940.31. The support rows H, C, P, N transport, replay, source, and name the same displayed decomposition, and no carrier coordinate remains for Plancherel, inversion, Hilbert completion, quotient coefficient equality, or infinite summation. □

Theorem 1940.36 (Parseval Real L^2 boundary refusal)**.** *Every Real-facing* $\mathsf{Parseval}^{\uparrow}$ *read that is later used as an* L^2 *energy boundary remains confined to the displayed carrier route through* $\mathsf{Fourier}^{\uparrow}$*,* $\mathsf{S}^{1\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*. The packet exports*

only the finite Real seal L over $F, S, I, E, R, D, H, C, P, N$ and refuses ambient L^2 completeness, host function equality, quotient coefficient spaces, infinite coefficient summation, Plancherel, and Fourier inversion.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalRealL2BoundaryRefusal`

Proof. Begin with Definition 1940.30. It identifies the terminal Real read as downstream from the displayed Fourier, source, pairing, integral, tolerance, readback, and structural rows.

Use Definition 1940.32 to expose the coefficient-energy side. The row is finite and NameCert-local, so it has no coordinate for an ambient L^2 carrier or host equality of functions.

Finally apply Definition 1940.31. Any Real or Integral consumer must traverse the same finite route, which rules out Plancherel, inversion, quotient coefficient spaces, infinite coefficient summation, and completed ambient norms. □

Theorem 1940.37 (Parseval kernel-scope package)**.** *The carrier of Definition 1940.1 and the obligations of Definition 1940.2 assemble a kernel-scoped* $\mathsf{Parseval}^{\uparrow}$ *package over coefficient windows, finite partial-energy ledgers, integral-energy handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, local* NameCert*, and TasteGate field-faithfulness. Every accepted package read is confined to the displayed route*

$$F, S, I, E, R, D, L, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalKernelScopePackage`

Proof. Start with Definition 1940.2. It already lists carrier admission, coefficient-window exactness, finite partial-energy ledger accounting, integral-energy handoff, dyadic tolerance accounting, Real seal non-escape, componentwise transport, replay, provenance, and local naming as the complete local obligation surface.

Use the finite coefficient ledger and coefficient-integral handoff of Definition 1940.6 and Definition 1940.11. These force the coefficient windows F, source row S, pairing row I, integral-energy row E, regular readback R, and tolerance ledger D to be displayed before the terminal seal.

Apply the dyadic tolerance ledger and Real-energy nonescape theorems, Definition 1940.12 and Definition 1940.30. They keep the numerical budget inside D and the endpoint inside the finite $\mathsf{Real}^{\uparrow}$ seal L, with no host equality, quotient coefficient space, infinite summation, Plancherel, inversion, or ambient Hilbert-completion coordinate.

Finally read the exhaustion, induction, and Real L^2 boundary routes of Definition 1940.31, Definition 1940.33, and Definition 1940.36. The support rows H, C, P, N preserve only the same finite carrier route, and the checked TasteGate alignment marker for Definition 1940.1 keeps field-faithfulness tied to those displayed rows. □

Theorem 1940.38 (Parseval scoped energy route)**.** *Every* $\mathsf{Parseval}^{\uparrow}$ *energy consumer must pass through* $\mathsf{Fourier}^{\uparrow}$ *coefficient windows,* $\mathsf{S}^{1\uparrow}$ *source rows,* $\mathsf{InnerProduct}^{\uparrow}$ *pairing,* $\mathsf{Integral}^{\uparrow}$ *energy,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before reading the energy identity surface. In carrier notation for $K = (F, S, I, E, R, D, L, H, C, P, N)$, the scoped route is exhausted by*

$$F, S, I, E, R, D, L, H, C, P, N.$$

It does not export Plancherel, Fourier inversion, Hilbert completion, quotient coefficient equality, selected limits, host function equality, infinite summation, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ParsevalUp.ParsevalScopedEnergyRoute`

Proof. Begin with root carrier admission and the NameCert obligations of Definition 1940.1 and Definition 1940.2. They expose exactly the rows $F, S, I, E, R, D, L, H, C, P, N$ before a local Parseval name can be consumed.

Next compose the Fourier-energy budget, integral-energy handoff, finite coefficient ledger, and coefficient-integral budget theorems: Definition 1940.4, Definition 1940.5, Definition 1940.6, and Definition 1940.8. These force every energy read to display coefficient windows, source samples, pairings, the integral row, readback, and dyadic tolerances before the Real seal is reached.

Finally use the kernel-scope package theorem Definition 1940.37. Its transport, replay, provenance, naming, and TasteGate alignment rows preserve only the same finite route, so the scoped package cannot be read as Plancherel, inversion, Hilbert completion, quotient coefficient equality, selected limits, host equality, infinite summation, or ambient Real completeness. □

Falsifiable prediction: Every $\mathsf{Parseval}^{\uparrow}$ energy-facing read factors through Fourier coefficient windows, an SOne source row, an InnerProduct pairing row, an Integral energy row, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fourier, s1, innerproduct, integral, regseqrat, dyadicratcore, real: $\mathsf{Parseval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries coefficient-energy windows, source rows, pairing witnesses, integral-energy observations, regular readbacks, dyadic tolerance accounting, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Parseval}^{\uparrow}$ packet is generated from finite Fourier coefficient windows, a displayed $\mathsf{S}^{1\uparrow}$ source row, an $\mathsf{InnerProduct}^{\uparrow}$ pairing row, an $\mathsf{Integral}^{\uparrow}$ energy row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 1940.38 closes the finite energy route through coefficient windows, source rows, pairing, integral energy, readback, tolerances, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Parseval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Plancherel, inversion, convergence closure, Hilbert-space completeness, host function equality, quotient coefficient spaces, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public energy-interface theorem exporting the same finite route without adding Plancherel, inversion, Hilbert completeness, quotient coefficient equality, selected limits, or ambient Real completeness.

Chapter 1941

A Concrete Naming Certificate for McShaneExtension$^{\uparrow}$

Following McShane and Whitney Lipschitz extension and Bishop–Bridges constructive analysis, the McShaneExtension$^{\uparrow}$ packet records the finite certificate by which a displayed real-valued Lipschitz map on a displayed subcarrier is read as an extension candidate on the ambient metric carrier. It consumes the map-level bound surface of chapter 410, the dyadic tolerance route of chapter 1864, the uniformly continuous extension handoff of chapter 769, and the separated completion uniqueness boundary of **??**. The packet names a finite Lipschitz-extension certificate; it is not a host formula for pointwise infima or suprema, not a quotient function-space construction, and not an appeal to selected bounds, function extensionality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 1941.1 (McShaneExtension carrier)**.** *A* McShaneExtension$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, A, L, D, E, S, R, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient source* Met$^{\uparrow}$ *row, exposing only the metric observations used by the extension request.*
2. *A is the displayed subcarrier or domain row, recording the admitted source points, the inclusion route into X, and the finite membership evidence consumed by later rows.*
3. *L is the source* LipschitzMap$^{\uparrow}$ *row for the real-valued map on A, including the displayed Lipschitz constant and the finite graph windows read on the subcarrier.*
4. *D is the* DyadicLipschitzModulus$^{\uparrow}$ *row that turns the displayed Lipschitz bound into dyadic tolerance data for source and ambient reads.*
5. *E is the candidate extension row, recording the finite McShane–Whitney envelope comparisons available to an ambient input without selecting a global infimum or supremum.*
6. *S is the source-agreement row, requiring the candidate extension read to coincide with the original source graph when the ambient input is admitted by A.*
7. *R records the* Real$^{\uparrow}$ *seal rows for source values, envelope comparisons, and ambient extension reads.*

8. *K records the* $\mathsf{Met}^{\uparrow}$ *seal rows for the source metric, subcarrier inclusion, and ambient distance comparisons.*
9. *H records componentwise* hsame *transport for the ambient metric, subcarrier, source Lipschitz row, dyadic modulus, candidate extension, source agreement, Real and Metric seals, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an ambient precision request through $X, A, L, D, E, S, R, K, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{McShaneExtension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{DyadicLipschitzModulus}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{CompletionExtensionUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{McShaneExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve displayed rows. It has no coordinate for host function equality, quotient function spaces, selected infima or suprema, countable-choice selectors, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a public bridge credential.*

Lean4 ✓ `BEDC.Derived.McShaneExtensionUp.McShaneExtensionTasteGate single carrier alignment`

Theorem 1941.2 (McShaneExtension NameCert obligations)**.** *For every accepted* $\mathsf{McShaneExtension}^{\uparrow}$ *carrier*

$$M = (X, A, L, D, E, S, R, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{McShaneExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed ambient metric, subcarrier, source Lipschitz, dyadic modulus, candidate extension, source agreement, Real and Metric seal, componentwise transport, replay, provenance, and local naming rows;*
2. *domain compatibility, requiring every source-value read to pass through A before it is compared with an ambient extension read;*
3. *Lipschitz-source admission, requiring L to be the only row that exposes the source graph and displayed Lipschitz constant;*
4. *dyadic modulus transport, requiring D to mediate every tolerance transfer from source distances to extension-value estimates;*
5. *candidate-extension exactness, requiring E to store only finite envelope comparison data and no selected infimum, selected supremum, quotient function, or host function object;*
6. *source-agreement exactness, requiring S to identify candidate and source reads only through the same displayed A, L, D, R, K route;*
7. *ledger non-escape, requiring H, C, P, N to preserve and replay the listed rows without adding host function equality, quotient function spaces, selected bounds, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1941.1. The accepted packet displays exactly the ambient metric, subcarrier, source Lipschitz row, dyadic modulus, candidate extension row, source-agreement row, Real and Metric seals, and the structural rows.

The domain and source clauses are read next. The row A is the only subcarrier admission row, and L is the only row exposing the real-valued Lipschitz graph on that subcarrier. Thus a source-value comparison cannot be formed before the packet has read both rows.

The tolerance route is then forced by D. By chapter 410 and chapter 1864, the displayed Lipschitz constant reaches ambient precision requests only through the finite dyadic modulus route, not through a host estimate.

The candidate and agreement clauses project E and S. The candidate row stores finite McShane–Whitney envelope comparisons, while the source-agreement row checks the same data on points admitted by A. These rows do not select a global infimum or supremum and do not turn the candidate into a quotient function-space element.

Finally inspect H, C, P, N. Transport is componentwise, replay follows the displayed route, and provenance plus local naming mention only the packet and its sibling dependencies, including chapter 769 and **??**. Hence the local NameCert obligations are exactly the listed finite rows. □

Theorem 1941.3 (McShaneExtension Lipschitz bound)**.** *Let $M = (X, A, L, D, E, S, R, K, H, C, P, N)$ be an accepted* $\mathsf{McShaneExtension}^{\uparrow}$ *carrier. Every ambient extension read exported by M factors through the finite route*

$$X, A \longmapsto L \longmapsto D \longmapsto E \longmapsto R, K \longmapsto H, C, P, N.$$

Along that route the displayed source Lipschitz constant remains the only bound used for the ambient candidate. The packet therefore exports a Lipschitz-bound certificate for the candidate extension without using host function equality, quotient function spaces, selected infima or suprema, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an ambient extension read. The ambient metric observations come from X, while subcarrier information and source comparison data come from A. No extension-value estimate is available before these two rows have been displayed.

Read the source Lipschitz and modulus rows. The row L supplies the displayed source graph and Lipschitz constant, and D converts that bound into dyadic tolerance data. This is the same finite tolerance discipline isolated by chapter 1864.

The candidate extension row E then records only finite envelope comparisons. Its reads are sealed through R and K, so Real-facing and Metric-facing consumers inspect the same finite route before reaching structural transport.

Transport, replay, provenance, and local naming preserve the route through H, C, P, N. The completion-facing siblings chapter 769 and **??** may consume this bound certificate downstream, but the carrier itself exports no host function equality, no quotient function space, no selected infimum or supremum, no choice principle, and no ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 1941.4 (McShaneExtension source-agreement envelope)**.** *Let $M = (X, A, L, D, E, S, R, K, H, C, P, N)$ be an accepted* $\mathsf{McShaneExtension}^{\uparrow}$ *carrier, and let an ambient input be admitted by the displayed subcarrier row A. The candidate envelope row E can be read as agreeing with the source Lipschitz row L only through the source-agreement row S and the finite route*

$$A, L, D, E, S \longmapsto R, K \longmapsto H, C, P, N.$$

Thus source agreement for the ambient read is a restriction of the finite McShane–Whitney envelope certificate back to the source row, mediated by S, D, R, K, H, C, P, N, and not by host function equality, selected envelope extrema, quotient function spaces, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and the admission of the ambient input through A. The source graph and displayed Lipschitz constant are exposed only by L, while the candidate envelope comparisons are exposed only by E.

Use the dyadic modulus row D to put the source read and the candidate envelope read at the same displayed tolerance. The source-agreement row S is then the only row that identifies the restricted envelope read with the original source read on an input admitted by A.

Finally inspect R, K, H, C, P, N. The Real and Metric seal rows keep the comparison on the same finite readback route, and the transport, replay, provenance, and local naming rows preserve that route componentwise. Since no listed row stores host equality, selected extrema, quotient function-space data, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every source-agreement read is the stated restriction of the finite envelope certificate. □

Theorem 1941.5 (McShaneExtension finite envelope route)**.** *Let*

$$M = (X, A, L, D, E, S, R, K, H, C, P, N)$$

be an accepted $\mathsf{McShaneExtension}^{\uparrow}$ *carrier. Every extension consumer of M reaches the candidate extension row only through the displayed rows $X, A, L, D, E, S, R, K, H, C, P, N$: the row L supplies the source graph and displayed Lipschitz constant, the row D supplies the dyadic tolerance route, the row E supplies the finite McShane–Whitney envelope comparisons, and the row S supplies the source-agreement restriction. Thus the human McShane–Whitney envelope derivation exported by M is a finite NameCert route, not a host infimum, host supremum, quotient function, selected representative, choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Proof. Begin with the NameCert obligations of Definition 1941.2. They expose the ambient metric row, displayed subcarrier, source Lipschitz row, dyadic modulus, candidate extension, source agreement, Real and Metric seals, transport, replay, provenance, and local naming as the complete obligation surface for an accepted packet.

Apply the Lipschitz-bound theorem Definition 1941.3. It forces an ambient extension read through $X, A, L, D, E, R, K, H, C, P, N$, with L as the only source of the graph and bound and D as the only dyadic tolerance route. The LipschitzMap and DyadicLipschitzModulus sibling chapters cited in the introduction provide exactly those source-bound and tolerance rows.

Now apply the source-agreement envelope theorem Definition 1941.4. When the ambient input lies in the displayed subcarrier, S is the only row that restricts the candidate envelope read back to the original source graph. The UniformContinuousExtension and CompletionExtensionUniqueness sibling chapters may consume the resulting handoff downstream, but they do not add an infimum, supremum, quotient function, selected representative, choice principle, or ambient completeness coordinate to this packet.

Therefore every extension consumer reaches the candidate row by the stated finite route through $X, A, L, D, E, S, R, K, H, C, P, N$, and the McShane–Whitney envelope derivation remains a displayed certificate over those rows. □

Falsifiable prediction: An accepted $\mathsf{McShaneExtension}^{\uparrow}$ consumer that reaches an ambient extension read without first reading the source $\mathsf{LipschitzMap}^{\uparrow}$ row, the $\mathsf{DyadicLipschitzModulus}^{\uparrow}$ row, the candidate extension row, and the source-agreement row refutes the carrier surface.

Independence witness: lipschitzmap, dyadic-lipschitz-modulus, uniform-continuous-extension, completion-extension-uniqueness: no listed sibling supplies the full twelve-row packet $M = (X, A, L, D, E, S, R, K,$ because each sibling contributes only a source bound, a dyadic tolerance route, a completion handoff, or a uniqueness boundary while this chapter binds the finite McShane–Whitney extension certificate into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{McShaneExtension}^{\uparrow}}$.

Constructive story: A McShaneExtension$^{\uparrow}$ packet is generated from an ambient MetricUp row, a displayed subcarrier row, a source LipschitzMap row, a DyadicLipschitzModulus row, a candidate extension row, a source-agreement row, Real and Metric seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{McShaneExtension}\uparrow}$, seed)

Scope closed: Definition 1941.1, Definition 1941.2, Definition 1941.3, and Definition 1941.4 seed the finite McShane–Whitney Lipschitz-extension certificate over displayed metric and subcarrier data.

Formal verification: FormalStatus(NameCert$_{\mathsf{McShaneExtension}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected infima or suprema, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, domain compatibility, source Lipschitz admission, dyadic modulus transport, candidate-extension exactness, source-agreement exactness, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.McShaneExtensionUp.

Chapter 1942

A Concrete Naming Certificate for $\mathsf{RealDyadicEmbedding}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RealDyadicEmbedding}^{\uparrow}$ packet records the finite certificate by which a displayed dyadic rational endpoint is admitted as a constant regular real name. It sits after the dyadic endpoint ledger of chapter 291, the constant finite-window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet names the constant-dyadic entry route; it does not assert quotient equality of real names, trichotomy, selected limits, density, algebraic closure, or ambient completeness.

Definition 1942.1 (RealDyadicEmbedding carrier)**.** *A* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, including the signed dyadic numerator, scale, endpoint comparison ledger, and finite tolerance rows used by later precision requests.*
2. *S is the constant* $\mathsf{StreamName}^{\uparrow}$ *schedule that repeats the displayed endpoint row of D at every finite observation window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *constant readback row, routing each requested regular rational observation through S and then back to the same dyadic endpoint data in D.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the route through D, S, R has been exposed.*
5. *H records componentwise* hsame *transport for the dyadic endpoint, constant stream schedule, regular readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay for every Real-facing or algebra-facing consumer that asks for a finite precision window of the constant name.*
7. *P is the* Pkg *provenance over* $\mathsf{RealDyadicEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDyadicEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of real names, order trichotomy, selected limits, density of dyadics, algebraic closure, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealDyadicEmbeddingUp.RealDyadicEmbeddingUpTasteGate single carrier alignment`

Theorem 1942.2 (RealDyadicEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *carrier* $K = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealDyadicEmbedding}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed dyadic endpoint, constant stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *endpoint constancy, requiring every finite observation window of* S *to read the same dyadic endpoint row* D*;*
3. *regular-readback exactness, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* R *to route through the constant schedule* S *and return to the endpoint ledger* D *before the Real seal is inspected;*
4. *Real-seal locality, requiring the terminal row* E *to be downstream of* D, S, R *and never detached as a host real object;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape for quotient equality, trichotomy, selected limits, density, algebraic closure, countable choice, and ambient completeness.*

No constant-dyadic admission obligation is stored outside these rows.

Proof. Project the accepted packet $K = (D, S, R, E, H, C, P, N)$. The carrier row exposes exactly one dyadic endpoint, one constant stream schedule, one regular readback, one Real seal, and the four structural rows.

Endpoint constancy is read from S. Since every finite observation window of the schedule repeats the endpoint data in D, a consumer cannot obtain a second dyadic source or a varying tail from the packet.

Regular-readback exactness is the route through R. Each requested regular rational observation first reads the constant schedule S, then returns to the dyadic ledger D, and only after that route may the terminal seal E be consumed.

The structural rows preserve the same route. Componentwise transport by H and replay by C can only move along the displayed coordinates, while P and N source and name the packet.

The omitted objects have no carrier coordinate. Hence quotient equality, trichotomy, selected limits, density, algebraic closure, countable choice, and ambient completeness cannot be produced by the local certificate. □

Theorem 1942.3 (RealDyadicEmbedding RegSeq handoff)**.** *Let* $K = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the embedded dyadic name factors through*

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before it can inspect the terminal seal. In particular, the consumer reads the DyadicRatCore endpoint, the constant StreamName schedule, and the RegSeqRat readback before the Real seal, and the route exports no quotient equality, trichotomy, selected limit, density theorem, algebraic closure theorem, countable-choice schedule, or ambient completeness witness.

Proof. Start with a Real-facing consumer of the packet. By Definition 1942.1, the only terminal Real row is E, and that row is downstream of the displayed dyadic endpoint, stream schedule, and regular readback.

The first source read is D. The constant schedule S exposes the same endpoint at every finite window, so the consumer's precision request remains tied to a displayed DyadicRatCore row.

The readback row R then converts the scheduled observation into regular rational data. This is the only row that hands the constant finite windows to the Real seal, so the seal cannot be inspected before D, S, R have all been read.

Finally, H, C, P, N transport, replay, source, and name that same finite path. Since no other analytic coordinates occur in the carrier, every RealUp consumer factors through the DyadicRatCore, StreamName, and RegSeqRat route before the terminal seal. □

Lean4 ✓ `BEDC.Derived.RealDyadicEmbeddingUp.RealDyadicEmbeddingCarrier regseq handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{RealDyadicEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDyadicEmbedding}^{\uparrow}$ packet is generated from a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint, a constant $\mathsf{StreamName}^{\uparrow}$ schedule, a constant $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDyadicEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1942.1, Definition 1942.2, and Definition 1942.3 seed the finite constant-dyadic admission route from a DyadicRatCore endpoint to RealUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDyadicEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, trichotomy, selected limits, density of dyadics, algebraic closure, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint constancy, regular-readback exactness, Real-seal locality, componentwise transport, replay, provenance, local naming, and non-escape for BEDC.Derived.RealDyadicEmbeddingUp.

Chapter 1943

A Concrete Naming Certificate for $\mathsf{RegularCauchyDivision}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDivision}^{\uparrow}$ packet names guarded division of regular Cauchy data as a finite reciprocal-then-product route. It sits over the regular rational source surface of chapter 301, the finite stream-window discipline of chapter 126, the dyadic endpoint arithmetic of chapter 291, the apartness-guarded reciprocal route of chapter 1471, the product surface of chapter 461, the real inverse boundary of chapter 635, and the terminal real seal of chapter 38. The chapter records only the finite NameCert boundary for X/Y when the denominator has displayed apartness; it is not host division and not a total reciprocal principle.

Definition 1943.1 (Regular Cauchy division carrier)**.** *A* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, A, I, P, W, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *X and Y are the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source rows for the numerator and denominator.*
2. *A is the denominator apartness guard exposing the finite nonzero-domain evidence for Y.*
3. *I is the* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *route that reads Y only through A and returns the reciprocal readback surface.*
4. *P is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route that multiplies X by the reciprocal route I.*
5. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for X, Y, the reciprocal readback, and the product readback.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the guarded quotient candidate produced by the product route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal receiving the guarded quotient packet.*
8. *H records componentwise* hsame *transport for $X, Y, A, I, P, W, R, E, C, Q, N$.*
9. *C records displayed* Cont *replay for source inspection, guard exposure, reciprocal routing, product routing, finite-window handoff, readback, and seal handoff.*

10. *Q records* Pkg *provenance over* $\mathsf{RegularCauchyDivision}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RealInverse}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDivision}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $X, Y, A, I, P, W, R, E, H, C, Q, N$.

Lean4 (definition) `BEDC.Derived.RegularCauchyDivisionUp.RegularCauchyDivisionUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyDivisionUp.RegularCauchyDivisionUpTasteGate single carrier alignment`

Theorem 1943.2 (Regular Cauchy division NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyDivision}^{\uparrow}}$ *surface for Definition 1943.1 consists exactly of numerator and denominator source exposure, denominator apartness exposure, reciprocal-route exposure, product-route exposure, finite StreamName windows, regular rational quotient readback, terminal Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and the local naming row.*

Proof. Project the carrier $D = (X, Y, A, I, P, W, R, E, H, C, Q, N)$. Its coordinates list the two regular Cauchy sources, the denominator guard, reciprocal route, product route, finite windows, quotient readback, seal, transport, replay, provenance, and local name.

The analytic rows are ordered by dependency. The reciprocal route I reads Y only after the apartness guard A, while the product route P reads X together with the reciprocal readback exposed by I.

The structural rows H, C, Q, N transport, replay, source, and name that same finite route. Since every listed obligation is a coordinate or a projection from a coordinate, the seed NameCert surface is exhausted by the displayed packet. □

Theorem 1943.3 (Regular Cauchy division reciprocal-product handoff)**.** *For every accepted* RegularCauchyDivision *carrier $D = (X, Y, A, I, P, W, R, E, H, C, Q, N)$, every quotient readback factors through*

$$(Y, A) \longmapsto I, \qquad (X, I) \longmapsto P, \qquad P \longmapsto R.$$

No product read can consume the denominator as a divisor until the reciprocal route has exposed the apartness-guarded reciprocal surface.

Proof. Read the denominator side first. The row A is the only coordinate licensing reciprocal use of Y, and the reciprocal route I is downstream of that guard.

Next read the product row P. Its left input is the numerator source X, and its right input is the reciprocal surface supplied by I; the carrier contains no separate division operation row.

Finally project the readback row R. Because R is downstream of P, every quotient read has already passed through the denominator guard, reciprocal route, and product route displayed in the statement. □

Theorem 1943.4 (Regular Cauchy division Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *packet reaches the seal row E only after the two source rows, denominator apartness guard, reciprocal route, product route, finite windows, quotient readback, transports, continuations, provenance, and local naming row have been displayed. The seal exports no total division, trichotomy, quotient stream equality, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start from the NameCert obligations. They expose $X, Y, A, I, P, W, R, E, H, C, Q, N$ as the complete packet surface.

Use the reciprocal-product handoff. It places I after the denominator guard and places P after the numerator source and reciprocal route, so the readback R cannot be reached by a direct host division route.

The Real seal row E is terminal. Since the carrier has no additional coordinate for total division, trichotomy, quotient stream equality, selected representatives, or ambient completeness, the seal can export only the guarded finite quotient route already displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDivision}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDivision}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, a denominator apartness guard, a $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ route, a $\mathsf{RegularCauchyProduct}^{\uparrow}$ route, finite $\mathsf{StreamName}^{\uparrow}$ windows, a quotient $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1943.1, Definition 1943.2, Definition 1943.3, and Definition 1943.4 seed the guarded division boundary over regular Cauchy sources, denominator apartness, reciprocal routing, product routing, finite windows, readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDivision}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyDivisionUp.RegularCauchyDivisionUpTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim total division, real trichotomy, quotient stream equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reciprocal-product handoff, Real seal non-escape, and the corresponding finite guarded-division NameCert rows.

Chapter 1944

A Concrete Naming Certificate for $\mathsf{StreamMerge}^{\uparrow}$

Following Bishop and Bridges on constructive sequences, the $\mathsf{StreamMerge}^{\uparrow}$ packet records the finite operation of reading two certified stream schedules through one visible merge selector. It sits over the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance surface of chapter 291, and the terminal real-name seal of chapter 38. It is the finite schedule-merge surface consumed by paired-window constructions such as chapter 414; it is not a quotient of streams, a countable choice selector, a host pairing primitive, or an ambient completion principle.

Definition 1944.1 (Stream merge carrier)**.** *A* $\mathsf{StreamMerge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S_0, S_1, \sigma, W_0, W_1, W, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two certified* $\mathsf{StreamName}^{\uparrow}$ *schedules whose finite observation windows may be inspected.*
2. *σ is the visible merge selector. Each displayed selector row records whether the merged position reads from S_0 or from S_1.*
3. *W_0 and W_1 are the source windows opened from S_0 and S_1.*
4. *W is the merged window ledger obtained by reading each displayed position through σ and then through the selected source window.*
5. *H records componentwise* hsame *transport for the two schedules, selector row, two source windows, merged window, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a merged-window request through σ, the selected S_i, the selected W_i, the merged row W, and the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{StreamMerge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamMerge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, \sigma, W_0, W_1, W, H, C, P, N$. It has no coordinate for quotient stream equality, a hidden selector, a host zip operation, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.StreamMergeUp.StreamMergeTasteGate single carrier alignment`

Theorem 1944.2 (Stream merge NameCert obligations)**.** *For every accepted* $\mathsf{StreamMerge}^{\uparrow}$ *carrier*

$$M = (S_0, S_1, \sigma, W_0, W_1, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamMerge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{StreamName}^{\uparrow}$ *schedules, visible selector, source windows, merged window, componentwise transport, displayed replay, provenance, and local naming rows;*
2. *selector-stable classifier transport, requiring every transported merged position to keep the same selected side under* H *and* C*;*
3. *source projection recovery, requiring the projection of* W *along either visible selector side to recover the displayed source window* W_0 *or* W_1*;*
4. *merged-window coverage, requiring every displayed row of* W *to be read from exactly one of the two source windows through* σ*;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer non-escape, requiring every regular-rational or real-facing consumer of the merged stream to pass through the selected* $\mathsf{StreamName}^{\uparrow}$ *window and the dyadic observation route before reaching a downstream seal.*

No stream-merge obligation is stored outside these rows.

Proof. Project the carrier of Definition 1944.1. The only schedule data available to the classifier are the two source schedules S_0, S_1, the visible selector σ, the source windows W_0, W_1, and the merged ledger W.

Selector-stable classifier transport is read from H and C. A transported selector row still names one of the two displayed source sides, so a merged position cannot change sides without changing a packet coordinate.

Source projection recovery follows by restricting the merged ledger to one selector side. The positions selected as 0 read through S_0 and W_0, while the positions selected as 1 read through S_1 and W_1.

Merged-window coverage is the converse read of the same finite data. Every row of W is produced by first inspecting σ and then using exactly the selected source window; the carrier has no third schedule or unscheduled row.

Finally, the provenance row P names the downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumer route, while N names the local certificate. Since quotient stream equality, hidden selectors, host pairing, selected limits, countable choice, and ambient completion are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 1944.3 (Stream merge projection stability)**.** *Let*

$$M = (S_0, S_1, \sigma, W_0, W_1, W, H, C, P, N)$$

be an accepted $\mathsf{StreamMerge}^{\uparrow}$ *carrier. Projecting* M *back to either source schedule preserves the displayed* $\mathsf{StreamName}^{\uparrow}$ *window on that side and preserves the dyadic,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumer route attached to that window. Thus a consumer of the projected source receives the same source schedule, source window, transport, replay, provenance, and local naming rows that were present before the merge.*

Proof. Project the accepted carrier and choose one side of the visible selector. The selected side determines whether the read uses S_0, W_0 or S_1, W_1.

Use source projection recovery from Definition 1944.2. The restriction of W to the chosen selector side is exactly the source window displayed in the carrier, so projection does not synthesize a different schedule window.

The dyadic and regular-rational route is unchanged because P records the same $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ dependency surface for the selected source window. Any real-facing consumer then reaches $\mathsf{Real}^{\uparrow}$ only after that selected StreamName window and its dyadic readback route have been displayed.

Transport and replay remain componentwise through H and C, and the local row N names the same finite packet. Therefore source projection preserves the displayed window and the downstream consumer route rather than exporting quotient stream equality, a hidden selector, a host zip operation, selected limits, countable choice, or ambient completion. □

Falsifiable prediction: Every paired-window Real or Completion consumer using two certified stream schedules factors through a $\mathsf{StreamMerge}^{\uparrow}$ packet that displays both source windows, one visible selector, one merged window ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, regularcauchyinterleaving: $\mathsf{StreamMerge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two certified schedules, a visible merge selector, two source windows, one merged window ledger, classifier transport, projection recovery, downstream regular-rational and real consumer routing, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamMerge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StreamMerge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1944.1, Definition 1944.2, and Definition 1944.3 seed the finite two-schedule merge route through StreamName windows, a visible selector, merged-window coverage, source projection recovery, and regular-rational and real-facing non-escape.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StreamMerge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, hidden selectors, host pairing, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, selector-stable classifier transport, source projection recovery, merged-window coverage, consumer non-escape, and TasteGate field faithfulness for BEDC.Derived.StreamMergeUp.

Constructive story: A $\mathsf{StreamMerge}^{\uparrow}$ packet is generated from two StreamName schedules, one visible merge selector, two source windows, one merged window ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1945

A Concrete Naming Certificate for $\mathsf{DyadicIntervalDiameter}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ packet names the finite dyadic width certificate used when an interval enclosure is handed to regular-rational and real-name consumers. It sits after the endpoint packet of chapter 306, the dyadic rational carrier of chapter 291, the rounding and tolerance surface of chapter 433, the nested dyadic enclosure route of chapter 328, the finite stream-window layer of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter isolates the width row that says how a displayed dyadic interval supplies an error budget before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads it.

Definition 1945.1 (Dyadic interval diameter carrier)**.** *A* $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, L, U, D, R, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{DyadicInterval}^{\uparrow}$ *row whose interval-order ledger and endpoint slots are being measured.*
2. *L and U are the lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows read from the same interval packet I.*
3. *D is the displayed nonnegative dyadic diameter row, recording the width comparison $0 \leq D$ and the endpoint bound $U - L \leq D$ in the visible dyadic order ledger.*
4. *R is the* $\mathsf{DyadicRounding}^{\uparrow}$ *tolerance row that translates the displayed diameter into the requested dyadic error budget.*
5. *W is the readback window opened for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers that need the interval midpoint, endpoint, or width information.*
6. *H records componentwise* hsame *transport for the interval row, endpoint rows, diameter row, tolerance row, readback window, replay, provenance, and local naming data.*
7. *C records displayed* Cont *replay for consumers that inspect the interval, the endpoints, the diameter bound, the rounding tolerance, or the readback window.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalDiameter}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicRounding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalDiameter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, U, D, R, W, H, C, P, N$. It has no coordinate for a host interval diameter, quotient real equality, a selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntervalDiameterUp.DyadicIntervalDiameterTasteGate single carrier alignment`

Theorem 1945.2 (Dyadic interval diameter NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ *carrier*

$$Q = (I, L, U, D, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalDiameter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed interval row, lower endpoint, upper endpoint, diameter row, tolerance row, readback window, transport, replay, provenance, and local naming;*
2. *endpoint source exactness, requiring L and U to be the endpoint rows exposed by the same* $\mathsf{DyadicInterval}^{\uparrow}$ *packet I;*
3. *nonnegative diameter exactness, requiring D to carry both the displayed nonnegativity witness and the visible dyadic bound from L, U to the width row;*
4. *tolerance handoff exactness, requiring every consumer error budget to pass through R before the readback window W is exposed;*
5. *classifier stability, requiring transported packets to compare only through the stored* hsame *and* Cont *rows;*
6. *ledger non-escape, requiring P, N to source and name no host interval diameter, quotient real equality, selected limit, trichotomy, countable choice, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1945.1. The accepted packet displays exactly one interval row, two endpoint rows, one nonnegative diameter row, one tolerance row, one readback window, and the structural rows that transport, replay, source, and name them.

Endpoint source exactness is read from I, L, U. The lower and upper rows are not independent dyadic rationals chosen by a host interval operation; they are the endpoint slots exposed by the displayed $\mathsf{DyadicInterval}^{\uparrow}$ carrier.

The nonnegative diameter clause is read from D. Its visible order ledger records $0 \leq D$ and the bound $U - L \leq D$, so the local NameCert surface can answer width queries by projection through the same dyadic rows.

The tolerance handoff clause is downstream of R and W. A requested error budget first reaches the rounding row R, and only then opens the finite readback window W for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumers.

Finally, H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Since the carrier contains no excluded coordinate, none of the excluded objects can enter the local NameCert surface. □

Theorem 1945.3 (Dyadic interval diameter width handoff)**.** *Let $Q = (I, L, U, D, R, W, H, C, P, N)$ be an accepted* $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ *carrier. Every* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read of interval width factors through the displayed route*

$$I, L, U, D \longmapsto R \longmapsto W.$$

The handoff supplies a finite dyadic error budget from the endpoint rows of I; it does not supply a host interval diameter, quotient stream equality, selected limiting point, trichotomy decision, countable-choice selector, or ambient completeness witness.

Proof. Begin with a consumer read of the width information. The only interval-facing coordinate is I, and the only endpoint-facing coordinates are L and U, so the read first projects the displayed dyadic interval and its two endpoint rows.

The diameter query then reaches D. This row supplies the nonnegative width certificate and the displayed bound from $U - L$ to D, so the consumer receives a dyadic error budget rather than an external interval-measure operation.

The tolerance row R repacks that dyadic budget for the requested readback precision. Once R has been displayed, W opens the finite stream and regular-rational window consumed downstream.

Because H, C, P, N only transport, replay, source, and name this route, every exported width read factors through the displayed chain and none of the excluded host objects is produced. □

Theorem 1945.4 (Dyadic interval diameter real error boundary)**.** *For an accepted* $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ *carrier* $Q = (I, L, U, D, R, W, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$*-facing error claim exported by* Q *is bounded by the displayed dyadic diameter row before it reaches the terminal real-name consumer. Concretely, the only public route to a real error boundary is*

$$I, L, U, D, R, W \longmapsto \mathsf{Real}^{\uparrow}.$$

The route certifies the finite enclosure width available to the real-name layer, but it does not identify quotient real representatives, choose a limit point, decide trichotomy, invoke countable choice, or close ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. A real-facing error claim cannot bypass the interval row I, because the endpoint rows L, U and the diameter row D are named as coordinates of that same finite packet.

The displayed diameter row is then the only source of a width bound. It provides the dyadic comparison needed for an error boundary, while the tolerance row R records the finite rounding step required before any readback window opens.

The window W is the only public bridge to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. Therefore a terminal real-name read receives the interval width through I, L, U, D, R, W and not through a quotient equality or a selected limit.

The transport, replay, provenance, and local naming rows preserve this route. Since the excluded objects are absent from the carrier, the real-facing boundary remains a finite displayed dyadic error certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalDiameter}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalDiameter}^{\uparrow}$ packet is generated from a DyadicInterval row, lower and upper DyadicRatCore endpoint rows, a displayed nonnegative diameter row, a DyadicRounding tolerance row, a finite StreamName/RegSeqRat/Real readback window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicIntervalDiameter}^{\uparrow}}$, seed)

Scope closed: Definition 1945.1, Definition 1945.2, Definition 1945.3, and Definition 1945.4 seed the finite diameter route from dyadic interval endpoints to regular rational and real-name error budgets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicIntervalDiameter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval diameter, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint source exactness, nonnegative diameter exactness, tolerance handoff exactness, real error boundary routing, classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalDiameterUp.

Chapter 1946

A Concrete Naming Certificate for $\mathsf{CauchyCompression}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompression}^{\uparrow}$ packet records the finite constructive operation that turns modulus-controlled Cauchy data into a fast Cauchy certificate. It sits over the visible modulus schedule of chapter 292, the compressed fast-Cauchy interface of chapter 323, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the L10 Real and Completion route with a standard finite bridge from an explicit modulus to a fast Cauchy certificate; it is not quotient stream equality, a selected limit, countable choice, host equality, ambient completeness, Lean verification, or a standard bridge.

Definition 1946.1 (Cauchy compression carrier)**.** *A* $\mathsf{CauchyCompression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, S, D, Q, F, E, H, C_0, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, recording the visible request-to-threshold map used to choose the finite tail windows.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite stream observations selected by the modulus schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested radii, slack comparisons, and dyadic arithmetic consumed by the selected windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same selected windows and tolerance ledger.*
5. *F is the* $\mathsf{FastCauchy}^{\uparrow}$ *compressed output row, recording the fast Cauchy certificate produced from M, S, D, Q.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the modulus, window, tolerance, readback, and fast-Cauchy rows have been displayed.*
7. *H records componentwise* hsame *transport for the modulus schedule, selected windows, tolerance ledger, regular readback, fast output, real seal, replay, provenance, and local naming rows.*
8. *C_0 records displayed* Cont *replay from each precision request through M, S, D, Q, F, E and then through H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompression}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompression}^{\uparrow}}$ *row.*

The classifier compares accepted compression packets only through the displayed rows $M, S, D, Q, F, E, H, C_0, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.CauchyCompressionUp.CauchyCompressionTasteGate single carrier alignment`

Theorem 1946.2 (Cauchy compression NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompression}^{\uparrow}$ *carrier*

$$C = (M, S, D, Q, F, E, H, C_0, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompression}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule M,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q,* $\mathsf{FastCauchy}^{\uparrow}$ *compressed output F,* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C_0, P, N;*
2. *modulus admission, requiring each precision request to enter through the displayed schedule M before any stream window is inspected;*
3. *finite window selection, requiring S to contain exactly the windows selected by M for the displayed precision requests;*
4. *dyadic accounting, requiring D to store exactly the tolerance radii and slack comparisons consumed by the selected windows;*
5. *readback and fast-output routing, requiring Q and F to remain downstream of M, S, D and to use no unlisted stream coordinate;*
6. *real-seal routing, requiring E to be reached only after the compressed fast-Cauchy output has been displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C_0, P, N;*
8. *non-escape for quotient streams, selected limits, countable choice, host equality, ambient completeness, Lean verification, and standard-bridge data.*

No Cauchy compression obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyCompressionUp.CauchyCompressionCarrier namecert obligations`

Proof. Project the carrier of Definition 1946.1. The analytic rows are the visible modulus schedule, selected stream windows, dyadic tolerance ledger, regular readback, fast-Cauchy output, and terminal real seal; the remaining rows transport, replay, source, and name that same route.

The modulus row is read before any window is opened. For each displayed precision request, M supplies a finite threshold, and the window row S records only the observations selected from that threshold. The construction therefore uses visible finite windows rather than a quotient stream or a selected limiting object.

The selected windows then pass through the dyadic ledger and readback rows. The ledger D contains the requested radii and slack comparisons, while Q is the regular rational readback obtained from those same displayed windows.

The compressed row F is downstream of M, S, D, Q. It is the only $\mathsf{FastCauchy}^{\uparrow}$ coordinate in the carrier, so every accepted fast certificate is produced from the displayed modulus-controlled route.

Finally E, H, C_0, P, N seal, transport, replay, source, and name the same finite route. Since none of the excluded objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1946.3 (Cauchy compression regular-sequence handoff)**.** *Let*

$$C = (M, S, D, Q, F, E, H, C_0, P, N)$$

be an accepted $\mathsf{CauchyCompression}^{\uparrow}$ *carrier. Every precision request answered by the fast Cauchy row F has the same finite route through the regular rational row Q and the real seal E:*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto F \longmapsto E \longmapsto H, C_0, P, N.$$

Consequently a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the displayed modulus schedule, selected windows, tolerance ledger, regular readback, fast-Cauchy certificate, seal, transport, replay, provenance, and local naming. The handoff does not export quotient streams, selected limits, countable choice, host equality, ambient completeness, Lean evidence, or standard-bridge data.*

Proof. Begin with the displayed precision request and read the modulus schedule M. The schedule selects a finite threshold, and the window row S records the stream observations opened at that threshold.

Pass the selected windows through D and Q. The dyadic ledger supplies the tolerance arithmetic, and the regular rational readback records the observation that the downstream consumer can inspect.

Now read the compressed row F. Because F is downstream of M, S, D, Q, the fast Cauchy certificate is obtained from the same finite route that produced the regular rational readback.

The terminal seal E is reached after F, and the structural rows H, C_0, P, N preserve the displayed route componentwise. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ handoff and the $\mathsf{Real}^{\uparrow}$ handoff consume the same finite data and introduce none of the excluded quotient, selector, choice, equality, completeness, verification, or bridge objects. □

Falsifiable prediction: Every $\mathsf{CauchyCompression}^{\uparrow}$ precision read factors through a CauchyModulus schedule, selected StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a FastCauchy compressed output, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchymodulus, fastcauchy, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyCompression}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries visible modulus scheduling, finite window selection, dyadic tolerance accounting, regular rational readback, fast Cauchy compression, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompression}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompression}^{\uparrow}$ packet is generated from a CauchyModulus schedule, selected StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a FastCauchy compressed output, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompression}^{\uparrow}}$, seed)

Scope closed: Definition 1946.1, Definition 1946.2, and Definition 1946.3 seed the finite route from visible modulus-controlled Cauchy data to a fast Cauchy certificate consumed by regular rational and Real interfaces.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompression}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, modulus admission, finite window selection, dyadic accounting, readback and fast-output routing, real-seal routing, and regular-sequence handoff for BEDC.Derived.CauchyCompressionUp.

Chapter 1947

A Concrete Naming Certificate for $\mathsf{RegularCauchyMap}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyMap}^{\uparrow}$ packet records the finite BEDC route by which a displayed map sends regular Cauchy rational names to regular Cauchy rational names before a Real-facing consumer reads the result. It sits between the finite source-window discipline of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, the continuity surface of chapter 130, and the terminal seal of chapter 38. The chapter names only the finite map surface between regular Cauchy names; it does not identify quotient streams, select limits, compare host functions, use countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1947.1 (Regular Cauchy map carrier)**.** *A* $\mathsf{RegularCauchyMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, F, T, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the source windows requested by the packet.*
2. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attached to those windows before any map output is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the visible dyadic radii, slack comparisons, and finite requests used by the source and target readbacks.*
4. *F is the displayed map-and-modulus row, recording for each inspected source precision a finite target precision request and the modulus comparison that justifies it.*
5. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *finite-window row produced from S, R, D, F.*
6. *Q is the target* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attached to T and to the same dyadic tolerance ledger.*
7. *A is the target* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after T and Q have been displayed.*
8. *H records componentwise* hsame *transport for the source windows, source regular handoff, dyadic ledger, map-and-modulus row, target windows, target regular handoff, target Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S \longmapsto R \longmapsto D, F \longmapsto T \longmapsto Q \longmapsto A.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyMapUp.RegularCauchyMapTasteGate single carrier alignment`

Theorem 1947.2 (Regular Cauchy map NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMap}^{\uparrow}$ *carrier*

$$M = (S, R, D, F, T, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source StreamName windows, source RegSeqRat handoff, DyadicRatCore tolerance ledger, displayed map-and-modulus row, target StreamName windows, target RegSeqRat handoff, target Real seal, and structural rows H, C, P, N;*

2. *source-window exactness, requiring every map request to project the finite source windows S before the source handoff R or any target row is inspected;*

3. *source regularity routing, requiring R to attach the inspected source windows to a regular rational name before F can read a source approximation;*

4. *dyadic tolerance transport, requiring D to carry the radii and slack comparisons used by the map-and-modulus row F;*

5. *target-window production, requiring T to be generated from the displayed source windows, source handoff, tolerance ledger, and map-and-modulus row;*

6. *target regularity and seal routing, requiring Q to mediate between T and A before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed;*

7. *non-escape for quotient stream equality, selected limits, host function equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking.*

No local regular-Cauchy map obligation is stored outside these rows.

Proof. Project the carrier of Definition 1947.1. The analytic coordinates are exactly the source finite windows, the source regular handoff, the dyadic tolerance ledger, the displayed map-and-modulus row, the target finite windows, the target regular handoff, and the target Real seal.

The source route is read first. A map request opens S, then passes through the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, so the packet never applies the map to an uninspected stream representative or to a selected limit.

The dyadic route is then transported through D and F. The ledger D supplies the visible radii and slack comparisons, while the row F turns each inspected source precision into a finite target precision request through its displayed modulus row.

Finally the target route is exposed. The windows T are produced only from the preceding rows, the target regular handoff Q attaches those windows to a regular rational name, and the Real seal A is downstream of Q. The structural rows H, C, P, N replay and name this same finite route, so the listed obligations exhaust the local NameCert surface. □

Theorem 1947.3 (Regular Cauchy map source-window preservation)**.** *Let*

$$M = (S, R, D, F, T, Q, A, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyMap}^{\uparrow}$ *carrier. Every target window produced by the displayed map-and-modulus row* F *is backed by a finite source window from* S*, a source* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*, and dyadic tolerance rows from* D*. Consequently a target-window consumer cannot bypass the source StreamName and RegSeqRat route.*

Proof. Begin with a target-window consumer. Projection gives the target row T, but T is a downstream coordinate of the route displayed by the carrier, not an independent stream representative.

Trace the target window backward through F. The map-and-modulus row stores the finite source precision and the target precision it requests, so the consumer must recover a source window in S before the target cells can be named.

The source window is attached to regular rational readback by R. The dyadic tolerance rows in D then supply the radii and slack comparisons used by the map row when it forms the target request.

Thus every target-window read preserves a source-window witness, a source regular handoff, and a dyadic tolerance route. None of these projections introduces quotient stream equality, a selected limit, host function equality, or countable choice. □

Theorem 1947.4 (Regular Cauchy map Real seal handoff)**.** *For every accepted* $\mathsf{RegularCauchyMap}^{\uparrow}$ *carrier* $M = (S, R, D, F, T, Q, A, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *seal consumer factors through*

$$S \longmapsto R \longmapsto D, F \longmapsto T \longmapsto Q \longmapsto A$$

together with the structural rows H, C, P, N*. In particular the target* $\mathsf{Real}^{\uparrow}$ *seal is reached only after the target* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff has been displayed.*

Proof. Start with a seal-facing request. Carrier projection exposes A as a target Real seal row, but the displayed route places A after the target regular handoff Q.

Read the target handoff. The row Q is attached to the target windows T, and those windows are produced by the source route S, R together with the dyadic ledger D and map-and-modulus row F.

Only then may the consumer inspect the Real seal. The replay row C records this ordering, the transport row H preserves it under componentwise comparison, and the provenance and local naming rows P, N name no additional principle.

Therefore the target Real seal is a downstream endpoint of the finite regular-Cauchy map route, not an ambient completeness or host function equality assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMap}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMap}^{\uparrow}$ packet is generated from source StreamName windows, a source RegSeqRat handoff, DyadicRatCore tolerance rows, a displayed map-and-modulus row, target StreamName windows, a target RegSeqRat handoff, a target RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyMap}^{\uparrow}}$, seed)

Scope closed: Definition 1947.1, Definition 1947.2, Definition 1947.3, and Definition 1947.4 seed the finite L10 map route between regular Cauchy names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, source regularity routing, dyadic tolerance transport, target-window production, target regularity and seal routing, and ledger non-escape for BEDC.Derived.RegularCauchyMapUp.

Chapter 1948

A Concrete Naming Certificate for $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$ packet records that the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ order of dyadic endpoints is preserved by the $\mathsf{RealDyadicEmbedding}^{\uparrow}$ route before $\mathsf{Real}^{\uparrow}$, density, or located-Archimedean consumers read the terminal seal. It sits after the dyadic order ledger of chapter 291, the constant dyadic real-name route of chapter 1942, the density witness surface of chapter 1484, the located Archimedean boundary of chapter 1851, the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal real-name seal of chapter 38. The chapter isolates the finite order-preserving handoff from dyadic endpoint data to Real-facing consumers.

Definition 1948.1 (RealDyadicOrderEmbedding carrier)**.** *A* $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D_0, D_1, O, E_0, E_1, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D_0 and D_1 are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, including their signed dyadic numerators, scales, endpoint tolerance ledgers, and visible comparison data.*
2. *O is the dyadic order row recording the comparison of D_0 and D_1 in the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *order ledger.*
3. *E_0 and E_1 are the two* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *rows sending the endpoint rows D_0, D_1 into constant regular real names.*
4. *S is the paired* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the two constant embedding routes before any Real-facing order consumer receives a seal.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S and returning each read to its source endpoint and order row.*
6. *A is the* $\mathsf{LocatedArchimedean}^{\uparrow}$ *and* $\mathsf{DyadicDensityReal}^{\uparrow}$ *handoff row used by later density or located-Archimedean consumers that ask for an embedded dyadic interval.*
7. *H records componentwise* hsame *transport for the two endpoint rows, order row, embedding rows, finite windows, readbacks, handoff row, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay for every consumer that inspects endpoint order, constant embedding data, density handoff, located-Archimedean search, or terminal* $\mathsf{Real}^{\uparrow}$ *seals.*
9. *P is the* Pkg *provenance over* $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealDyadicEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicDensityReal}^{\uparrow}$, $\mathsf{LocatedArchimedean}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDyadicOrderEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D_0, D_1, O, E_0, E_1, S, R, A, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected limits, countable choice, a host density theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealDyadicOrderEmbeddingUp.TasteGate.RealDyadicOrderEmbeddingTasteGate`
`single carrier alignment`

Theorem 1948.2 (RealDyadicOrderEmbedding NameCert obligations)**.** *For every accepted* RealDyadicOrderEmbe *carrier*

$$K = (D_0, D_1, O, E_0, E_1, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealDyadicOrderEmbedding}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed dyadic endpoint rows, order row, embedding rows, stream windows, regular readbacks, density and located-Archimedean handoff, transport, replay, provenance, and local naming;*
2. *dyadic-order exactness, requiring O to read the comparison of the endpoint rows D_0, D_1 from the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger rather than from a detached host order;*
3. *embedding-order preservation, requiring E_0, E_1 to expose the constant real-name images of D_0, D_1 only together with the order row O;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring every finite Real-facing order read to pass through S and R before a terminal* $\mathsf{Real}^{\uparrow}$ *seal is inspected;*
5. *ledger non-escape, requiring H, C, P, N to preserve, replay, source, and name only the displayed rows and to export no trichotomy, quotient real equality, selected limit, countable-choice schedule, host density theorem, ambient Real completeness, Lean verification, or bridge checking.*

No order-embedding obligation is stored outside these rows.

Proof. Project the accepted packet $K = (D_0, D_1, O, E_0, E_1, S, R, A, H, C, P, N)$. The carrier admission row exposes exactly two dyadic endpoint rows, one dyadic comparison row, two constant embedding rows, one paired window family, one paired readback row, one density and located-Archimedean handoff row, and the four structural rows.

Dyadic-order exactness is read from O. Since O compares the endpoint rows D_0, D_1 inside the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, the packet cannot replace that comparison by a host real order or by a trichotomy decision.

Embedding-order preservation is the route through E_0, E_1. Each embedding row is the constant $\mathsf{RealDyadicEmbedding}^{\uparrow}$ image of its matching endpoint row, and both rows are carried with O, so a Real-facing order read receives the dyadic comparison before it receives either terminal seal.

The window and readback clauses are then projection through S and R. Every finite consumer request opens the paired $\mathsf{StreamName}^{\uparrow}$ windows and paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks before the same data can be handed to $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name only this displayed route. Since the carrier contains no excluded coordinate, the five rows exhaust the local certificate surface. □

Theorem 1948.3 (RealDyadicOrderEmbedding density handoff)**.** *Every density or located-Archimedean consumer of the embedded dyadic interval of an accepted* $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$ *carrier factors through*

$$D_0, D_1, O \longmapsto E_0, E_1 \longmapsto S, R \longmapsto A \longmapsto \mathsf{Real}^{\uparrow}.$$

In particular, the consumer reads the dyadic order row and the two $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *rows before reaching* $\mathsf{DyadicDensityReal}^{\uparrow}$, $\mathsf{LocatedArchimedean}^{\uparrow}$, *or the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with a density or located-Archimedean consumer of the embedded interval. By Definition 1948.1, the only interval endpoints available to that consumer are the dyadic rows D_0, D_1 and their comparison row O.

The consumer then crosses the constant embedding rows. The rows E_0, E_1 are the only $\mathsf{RealDyadicEmbedding}^{\uparrow}$ coordinates, so the dyadic endpoints enter the Real-facing surface only through the two displayed embedding routes.

The finite observation layer is S, R. These rows expose the paired $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks attached to the two embeddings, and they precede every handoff row in the carrier.

The density and located-Archimedean handoff row A is downstream of those reads. Thus any later $\mathsf{DyadicDensityReal}^{\uparrow}$ or $\mathsf{LocatedArchimedean}^{\uparrow}$ request receives the dyadic order data and the two constant embedding routes before it can inspect a terminal $\mathsf{Real}^{\uparrow}$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDyadicOrderEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDyadicOrderEmbedding}^{\uparrow}$ packet is generated from two $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, one displayed dyadic order row, two $\mathsf{RealDyadicEmbedding}^{\uparrow}$ rows, paired $\mathsf{StreamName}^{\uparrow}$ windows, paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicDensityReal}^{\uparrow}$ and $\mathsf{LocatedArchimedean}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDyadicOrderEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1948.1, Definition 1948.2, and Definition 1948.3 seed the finite order-preserving dyadic-to-real handoff for density and located-Archimedean consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDyadicOrderEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-order exactness, embedding-order preservation, StreamName and RegSeqRat handoff, density handoff, ledger non-escape, and bridge coverage for BEDC.Derived.RealDyadicOrderEmb

Chapter 1949

A Concrete Naming Certificate for $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley-style metric completion, the $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$ packet records the finite certificate by which a closed metric subspace inherits the completion-limit route carried by its ambient metric completion. It sits after the subspace admission surface of chapter 1519, the Bishop regular Cauchy completion route of chapter 1639, the completion carrier of chapter 1506, the complete-metric limit row of chapter 128, the separated metric boundary of chapter 1411, the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal real-name seal of chapter 38. The chapter isolates the displayed route from an admitted subspace Cauchy packet to a retained subspace limit, without quotienting the subset or selecting limits by choice.

Definition 1949.1 (ClosedSubspaceCompletion carrier)**.** *A* $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, A, K, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the parent* $\mathsf{MetricCompletion}^{\uparrow}$ *row, including the ambient separated metric carrier, the regular Cauchy source data, and the completion handoff route used by downstream consumers.*
2. *A is the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row naming the displayed subspace predicate, the inherited metric restriction, and the finite membership ledger for the source Cauchy packet.*
3. *K is the closedness-under-limit row: whenever a Cauchy source is admitted by A and its ambient completion route in X reaches a displayed limit, the same route carries the subspace-membership witness to that limit.*
4. *L is the inherited* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row obtained by replaying the ambient completion route through K and then reading the result as a subspace limit rather than as a detached ambient point.*
5. *H records componentwise* hsame *transport for the parent completion row, subspace admission row, closedness row, inherited limit row, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay for every consumer that inspects subspace admission, ambient completion data, closedness-under-limit, or the inherited complete-metric row.*

7. *P is the* Pkg *provenance over* $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSubspaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, A, K, L, H, C, P, N. It has no coordinate for quotient subset equality, selected limits, countable choice, open-cover compactness, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ClosedSubspaceCompletionUp.ClosedSubspaceCompletionTasteGate single carrier alignment`

Theorem 1949.2 (ClosedSubspaceCompletion NameCert obligations)**.** *For every accepted* ClosedSubspaceComplet *carrier*

$$Q = (X, A, K, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedSubspaceCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the parent metric-completion row, metric-subspace admission row, closedness-under-limit row, inherited complete-metric limit row, transport, replay, provenance, and local naming;*

2. *source compatibility, requiring the Cauchy source read by X to be admitted through the same subspace row A before any limit row is inspected;*

3. *closedness retention, requiring K to carry the subspace-membership witness along the ambient completion route rather than through quotient subset equality or a selected representative;*

4. *inherited-limit exactness, requiring L to be obtained by replaying the ambient* $\mathsf{CompleteMetric}^{\uparrow}$ *row through K and not by invoking ambient completeness for the subspace;*

5. *classifier stability, requiring transported packets to compare only through the stored* hsame *and* Cont *rows;*

6. *ledger non-escape, requiring P, N to source and name no selected limit, countable-choice selector, open-cover compactness proof, ambient completeness principle,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1949.1. The accepted packet displays exactly one parent completion row, one subspace admission row, one closedness-under-limit row, one inherited limit row, and the structural rows that transport, replay, source, and name them.

Source compatibility is read from X and A. The Cauchy source used by the parent completion route is the same source whose finite membership ledger is exposed by the subspace admission row, so a consumer cannot replace it by a detached subspace quotient.

Closedness retention is the role of K. It transports the displayed membership witness along the ambient completion route and returns a limit that is still read through the subspace row.

Inherited-limit exactness is then projection through L. The complete-metric limit row is obtained after K has replayed the ambient route, so the subspace receives a retained limit rather than a new ambient-completeness principle.

Finally, H, C, P, N preserve, replay, source, and name only this displayed route. Since the carrier contains no excluded coordinate, the listed rows exhaust the local certificate surface. □

Theorem 1949.3 (ClosedSubspaceCompletion limit retention)**.** *Let $Q = (X, A, K, L, H, C, P, N)$ be an accepted* $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$ *carrier. Every subspace Cauchy source admitted by A and completed through the parent row X reaches its inherited limit only through the retained route*

$$X, A \longmapsto K \longmapsto L.$$

Thus the exported limit remains a subspace limit named by the displayed closedness row, and no selected ambient limit, quotient subset equality, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness is exported.*

Proof. Start with a Cauchy source admitted by A. By the carrier definition, the only completion data available for that source is the parent row X, so the source first follows the displayed metric-completion route.

The closedness row K is then forced. It is the only coordinate that can turn the ambient limit reached through X into a subspace-membership witness, and it does so by replaying the same finite source route.

The inherited limit row L is downstream of K. Consequently the complete-metric consumer receives a limit only after the closedness-under-limit witness has retained subspace membership.

The structural rows preserve this order under transport and replay. Since no excluded limit-selection or ambient-completeness coordinate is present, every exported subspace limit factors through X, A, K, L. □

Theorem 1949.4 (ClosedSubspaceCompletion nonescape)**.** *For an accepted* $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$ *carrier $Q = (X, A, K, L, H, C, P, N)$, every* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the inherited subspace completion data reads only the displayed rows*

$$X, A, K, L, H, C, P, N.$$

In particular, the packet cannot export quotient subset equality, selected limits, countable choice, open-cover compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness beyond the named finite route.*

Proof. Project the accepted carrier. The only consumer-facing rows are the parent completion row X, subspace admission row A, closedness-under-limit row K, inherited limit row L, and the four structural rows.

A $\mathsf{CompleteMetric}^{\uparrow}$ consumer must read L, and L is available only after the closedness row K has replayed the ambient route from X with the subspace admission ledger A attached.

The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers are no freer. Their finite windows, regular readbacks, and terminal real-name seals are inherited through the parent completion row and are replayed through the same closedness and subspace-admission route.

The rows H, C, P, N do not add data; they only transport, replay, source, and name the accepted packet. Since none of the excluded objects is a coordinate of Q, no consumer can recover such an object from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSubspaceCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$ packet is generated from a parent $\mathsf{MetricCompletion}^{\uparrow}$ row, a $\mathsf{MetricSubspace}^{\uparrow}$ admission row, a closedness-under-limit row, an inherited $\mathsf{CompleteMetric}^{\uparrow}$ limit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedSubspaceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1949.1, Definition 1949.2, Definition 1949.3, and Definition 1949.4 seed the finite closed-subspace completion route from admitted Cauchy source to retained subspace limit.

Formal verification: FormalStatus(NameCert$_{\text{ClosedSubspaceCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient subset equality, selected limits, countable choice, open-cover compactness, ambient completeness, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, closedness retention, inherited-limit exactness, consumer non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ClosedSubspaceCompletion

Chapter 1950

A Concrete Naming Certificate for $\mathsf{NormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges constructive analysis, the $\mathsf{NormedSpace}^{\uparrow}$ packet records the finite NameCert surface for a non-singleton normed vector space. It consumes the displayed vector-space source of chapter 47, the real-valued norm interface of chapter 89, the metric readback discipline of chapter 57, and the completion-facing boundary of chapter 95. The packet is an interface for later Banach, Hilbert, bounded-operator, and functional-analysis certificates; it is not the singleton $\mathsf{Norm}^{\uparrow}$ instance, not a host normed-space structure, not quotient completion, and not a choice principle.

Definition 1950.1 (NormedSpace carrier)**.** *A* $\mathsf{NormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, N, M, Q, H, T, P, C)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row, exposing the vector carrier, vector classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by all norm values, scalar absolute values, metric distances, and completion tolerances.*
3. *N is the real-valued norm row, carrying a map from admitted vector histories to R-sealed real histories together with nonnegativity, zero exactness, triangle inequality, and scalar compatibility evidence.*
4. *M is the induced* $\mathsf{Met}^{\uparrow}$ *row, whose distance read between x and y is the norm read of the displayed difference $x - y$ through V, N, R.*
5. *Q is the completion-facing row, recording Cauchy windows, tolerance requests, and handoff data for completion consumers without selecting a completed point.*
6. *H records componentwise* hsame *transport for V, R, N, M, Q and for the replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from every norm, metric, or completion request through the same rows V, R, N, M, Q, H, P, C.*
8. *P is the* Pkg *provenance over* $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *completion-facing Cauchy data,* BHist, hsame, Cont, *and* NameCert.

9. C is the local $\mathsf{NameCert}_{\mathsf{NormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for host normed-space equality, quotient completion, selected limits, basis choice, classical duality, or an ambient Banach or Hilbert structure.

Lean4 (definition) `BEDC.Derived.NormedSpaceUp.NormedSpaceUp`

Theorem 1950.2 (NormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, N, M, Q, H, T, P, C),$$

the local $\mathsf{NameCert}_{\mathsf{NormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, norm, induced metric, completion-facing, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring equality of packet reads to be checked componentwise through $V, R, N, M, Q, H, T, P, C$ and never through host extensional equality;*
3. *zero exactness, requiring the norm-zero classifier in N to coincide with the displayed vector-zero classifier in V;*
4. *triangle inequality, requiring every norm read of a displayed sum to factor through the R-sealed addition and order rows already attached to N;*
5. *scalar compatibility, requiring each scalar multiplication read to pass through the scalar row of V, the real absolute-value row of R, and the norm row N;*
6. *metric handoff, requiring M to read distances only as norm reads of displayed vector differences through V, N, R;*
7. *completion-facing nonescape, requiring Q, H, T, P, C to preserve the same displayed route for Cauchy and completion consumers without adding quotient completions, selected limits, basis choice, choice, or host Banach/Hilbert structure.*

No $\mathsf{NormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1950.1. The accepted packet lists the vector-space, Real, norm, induced metric, completion-facing, transport, replay, provenance, and local naming rows.

Classifier transport is then componentwise. A comparison of two accepted packets can inspect only the rows displayed in the carrier, so vector data are compared through V, real and norm data through R, N, metric data through M, completion-facing data through Q, and structural replay through H, T, P, C.

The zero, triangle, and scalar rows are the norm obligations. Zero exactness identifies a zero norm read precisely with the displayed vector-zero classifier of V. The triangle row reads the norm of a sum through the real addition and order data in R, and scalar compatibility reads scalar multiplication through the scalar row of V together with the absolute-value row of R.

The metric row is forced by construction: M has no independent distance source. Its distance read from x to y is the norm read of the displayed difference $x - y$, so every metric consumer must pass through V, N, R before it reaches M.

Finally, the completion-facing row Q and the structural rows H, T, P, C replay the same displayed route for Cauchy windows and completion requests. Since the carrier has no quotient-completion, selected-limit, basis-choice, or host Banach/Hilbert coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 1950.3 (NormedSpace zero exactness)**.** *Let $S = (V, R, N, M, Q, H, T, P, C)$ be an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet. The zero classifier exposed by the norm row N is exact precisely along the displayed vector-zero row of V: a norm read may certify zero only by reading the vector zero classifier in V, the real zero and order rows in R, and the local transport and replay rows H, T, C. It cannot certify zero by host vector equality, quotient equality, basis coordinates, or an ambient Banach-space identity.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier zero exactness`

Proof. Project the carrier of Definition 1950.1. The only vector zero datum is in V, and the only real zero and order data available to a norm read are in R.

Apply the zero-exactness clause of Definition 1950.2. It identifies a zero norm read with the displayed vector-zero classifier rather than with a host equality relation.

The structural rows H, T, C preserve this route under transport and replay, while P and the local naming row keep it inside the same packet. Since no quotient equality, basis coordinate, or ambient Banach identity is a carrier row, the zero read is exact along V, R, N. □

Theorem 1950.4 (NormedSpace triangle and scalar compatibility)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet $S = (V, R, N, M, Q, H, T, P, C)$, the triangle and scalar-compatibility obligations are routed through the same displayed rows. A norm read of a sum must first expose vector addition in V, then the norm row N, and then the real addition and order rows in R. A norm read of a scalar multiple must first expose scalar action in V, then the scalar absolute-value read in R, and then the norm row N. Neither read may use host linearity, an external ordered field, basis expansion, quotient vectors, or a hidden continuity principle.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion request budget`

Proof. Start with the triangle request. The sum is not a host vector expression; it is the displayed addition operation in V. Once the sum row is read, the norm row N supplies the norm observation and the Real row R supplies the addition and order comparison.

Now consider scalar compatibility. The scalar action is again a displayed operation in V, while the absolute-value and real multiplication reads come from R. The norm row N relates those two displayed sources.

Finally transport and replay the two routes through H, T, C with provenance in P. Because the carrier contains no host linearity, external ordered field, basis expansion, quotient-vector coordinate, or hidden continuity row, the triangle and scalar obligations stay inside V, R, N. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier triangle scalar compatibility`

Theorem 1950.5 (NormedSpace metric nonescape)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet, the metric row M can read a distance between displayed vectors only as the norm read of their displayed difference through V, N, R. The metric consumer may then pass through Q, H, T, P, C for completion-facing handoff, transport, replay, provenance, and local naming, but it cannot export an independent host metric, quotient completion, selected limit, or Banach/Hilbert structure.*

Proof. Read M from Definition 1950.1. Its distance data are induced data, and the carrier names no distance source except the norm row N applied to a displayed vector difference from V.

Apply the metric handoff clause of Definition 1950.2. It forces each distance request through V, N, R before M can be read by a downstream consumer.

The completion-facing row Q and the structural rows H, T, P, C preserve the same displayed route. Since no independent host metric, quotient completion, selected limit, or Banach/Hilbert coordinate is present, metric reads cannot escape the induced norm route. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier metric nonescape`

Theorem 1950.6 (NormedSpace completion handoff)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet reads the packet through the displayed route*

$$V, R, N, M, Q \longmapsto H, T, P, C.$$

In particular, the consumer obtains its vector operations from V, its real-valued norm and scalar absolute-value reads from R, N, its metric comparisons from M, and its Cauchy or completion-facing requests from Q. It does not obtain a host normed-space object, quotient completion, selected limit, basis choice, choice principle, or ambient Banach/Hilbert structure from the packet.

Proof. Start with a downstream consumer request. A vector operation can be answered only by projecting the displayed vector-space row V, while a norm or scalar-size request must read R and N before any real-valued comparison is available.

Metric requests are not separate host data. By Definition 1950.2, the metric row M reads distance as the norm of a displayed vector difference, so the route necessarily passes through V, N, R before a metric comparison is exported.

Completion-facing requests then pass through Q. This row stores only Cauchy windows, tolerance requests, and handoff data, so a Banach or Hilbert chapter may consume readiness for completion without receiving a selected completed point or a quotient object.

The structural rows H, T, P, C preserve the same route under transport, replay, provenance, and local naming. Thus every listed consumer sees exactly the displayed VecSpace, Real, Metric, and Completion rows, and no host normed-space structure or choice-based completion datum is introduced. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion handoff`

Theorem 1950.7 (NormedSpace root-unblock source triad)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet*

$$S = (V, R, N, M, Q, H, T, P, C),$$

the root-unblock handoff to Banach, Hilbert, bounded-operator, and functional-analysis consumers is exhausted by the following three displayed source obligations:

1. *the* $\mathsf{VecSpace}^{\uparrow}$ *carrier is transported through the* $\mathsf{Norm}^{\uparrow}$ *row N by the same* BHist, hsame, Cont, Pkg, *and* NameCert *rows that name S;*
2. *the* $\mathsf{Met}^{\uparrow}$ *identity-of-indiscernibles read is routed through the zero norm row, the* $\mathsf{Real}^{\uparrow}$ *zero and order cells in R, and the displayed vector-zero classifier in V;*
3. *the* $\mathsf{Completion}^{\uparrow}$ *handoff is a nonescape handoff through Q, H, T, P, C, carrying only Cauchy windows and tolerance requests derived from V, R, N, M.*

No source obligation quantifies over abstract source-equivalence pairs, and no consumer receives host vector equality, quotient completion, selected limits, basis choice, classical duality, or an ambient Banach or Hilbert structure.

Proof. Project the carrier S. The vector-space source is the row V, and the norm row N is the only row that can expose a real-valued norm read over vectors from V.

For the carrier-transport obligation, read V through N and then through H, T, P, C. This keeps the $\mathsf{VecSpace}^{\uparrow}$ source inside the same BHist packet and makes the local $\mathsf{NameCert}$ row the only naming boundary.

For identity of indiscernibles, use Definition 1950.3. The zero norm row reads the $\mathsf{Real}^{\uparrow}$ zero and order cells in R and then returns to the displayed vector-zero classifier in V, so the induced $\mathsf{Met}^{\uparrow}$ row M has no independent equality source.

For completion handoff, use Definition 1950.6. The row Q stores Cauchy windows and tolerance requests only after the vector, Real, norm, and metric rows have been displayed, while H, T, P, C preserve the same route for downstream consumers.

Thus the three obligations are finite source rows of the packet itself. Since no abstract source-equivalence pair, host vector equality, quotient completion, selected limit, basis choice, classical duality, or ambient Banach or Hilbert coordinate occurs in S, the root-unblock surface is exactly the displayed triad. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier root unblock source triad`

Theorem 1950.8 (NormedSpace completion-facing obligation triad)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet*

$$S = (V, R, N, M, Q, H, T, P, C),$$

the completion-facing surface is exhausted by the displayed $\mathsf{VecSpace}^{\uparrow}$ *row* V*, the* $\mathsf{Real}^{\uparrow}$ *row* R*, the norm row* N*, the induced* $\mathsf{Met}^{\uparrow}$ *row* M*, and the Cauchy-window handoff row* Q *transported and replayed through* H, T, P, C*. No completion-facing obligation is stored outside the VecSpace, Real, Norm, Metric, and Cauchy-window rows.*

Proof. Project the carrier S. The rows V, R, N, M, Q are the only coordinates that a completion consumer can inspect before structural transport and replay.

Apply Definition 1950.3 to pin zero reads to V, R, N. Apply Definition 1950.5 to pin distance reads to the induced metric row M through the same vector, norm, and Real rows.

Use Definition 1950.6 and Definition 1950.7. The handoff theorem identifies Q as the only Cauchy-window and tolerance-request boundary, while the triad theorem shows that downstream consumers receive the displayed route rather than an abstract source-equivalence surface.

Therefore the completion-facing obligations are exactly the VecSpace, Real, Norm, Metric, and Cauchy-window rows, preserved by H, T, P, C. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion facing obligation triad`

Theorem 1950.9 (NormedSpace completion-facing nonescape)**.** *Let* $S = (V, R, N, M, Q, H, T, P, C)$ *be an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet. No quotient completion, selected limit, basis choice, host vector equality, ambient Banach structure, or ambient Hilbert structure can be read from* Q, H, T, P, C*. These rows transport, replay, source, and locally name only the displayed completion-facing route already determined by* V, R, N, M*.*

Proof. Read the completion-facing row Q. By Definition 1950.6, this row stores Cauchy windows and tolerance requests only after the vector, Real, norm, and metric rows have been displayed.

Apply Definition 1950.5. The metric row feeding Q is induced from V, N, R, so Q cannot import an independent host metric or quotient equality.

Finally use Definition 1950.7. The rows H, T, P, C preserve the same finite route for downstream consumers. Since quotient completions, selected limits, basis choices, host vector equality, and ambient Banach or Hilbert structures are absent from the carrier, none can be read from the completion-facing surface. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion facing nonescape`

Theorem 1950.10 (NormedSpace consumer exhaustion)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet must factor through the displayed route*

$$V, R, N, M, Q, H, T, P, C.$$

The consumer obtains vector operations from V, real scalar and norm comparisons from R, N, metric comparisons from M, completion-facing windows from Q, and structural transport, replay, provenance, and local naming from H, T, P, C; no additional host analysis structure is exported by the packet.

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier consumer exhaustion`

Proof. Start with a downstream analysis request. By Definition 1950.8, the only completion-facing surface available to the request is the displayed VecSpace, Real, Norm, Metric, and Cauchy-window route.

Use Definition 1950.9. The structural rows do not add quotient completion, selected limits, basis choice, host vector equality, or ambient Banach/Hilbert structure.

Apply Definition 1950.6. Banach, Hilbert, bounded-operator, and functional-analysis consumers all receive their data from the same displayed route $V, R, N, M, Q, H, T, P, C$.

Thus every such consumer factors through the named finite packet, and the consumer surface is exhausted by that route. □

Theorem 1950.11 (NormedSpace Banach-completion nonescape)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet*

$$S = (V, R, N, M, Q, H, T, P, C)$$

must read the displayed route $V, R, N, M, Q, H, T, P, C$ before any completion-facing handoff is available. The packet exposes vector operations only through $\mathsf{VecSpace}^{\uparrow}$ *row V, real-valued norm and scalar-size data only through R, N, metric comparisons only through the induced metric row M, and completion requests only through Q, H, T, P, C.*

It cannot supply quotient completion, selected limits, basis choice, host vector equality, classical duality, or ambient Banach/Hilbert structure.

Proof. Apply Definition 1950.3 for the vector-zero boundary. A zero norm read returns to the displayed vector-zero classifier in V after passing through the $\mathsf{Real}^{\uparrow}$ zero and order cells in R, so no host vector equality or quotient equality can enter through the norm row.

Use Definition 1950.5 for metric consumers. The metric row M reads distance as a norm of a displayed vector difference through V, N, R, and therefore has no independent host metric or ambient Banach coordinate.

Next apply Definition 1950.6. Completion-facing reads pass through Q only after the vector, Real, norm, and metric rows have been displayed, and Q stores Cauchy windows and tolerance requests rather than a quotient completion or selected completed point.

Finally use Definition 1950.7. The structural rows H, T, P, C transport, replay, source, and name the same route for downstream Banach, Hilbert, bounded-operator, and functional-analysis consumers, so the displayed packet admits no basis choice, classical duality, host equality, selected limit, or ambient Banach/Hilbert datum. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier banach completion nonescape`

Theorem 1950.12 (NormedSpace vector transport obligation)**.** *Let*

$$S = (V, R, N, M, Q, H, T, P, C)$$

be an accepted $\mathsf{NormedSpace}^{\uparrow}$ *packet. Every vector-space read exported to Banach, Hilbert, bounded-operator, or functional-analysis consumers must transport the displayed row V through the same* BHist, hsame, Cont, Pkg, *and local* NameCert *rows H, T, P, C that name S, while keeping the norm row N, the real row R, the induced metric row M, and the completion-facing row Q in scope. The transport cannot be replaced by host vector equality, quotient completion, selected limits, basis choice, or host normed-space equality.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier vector transport obligation`

Proof. Project the carrier of Definition 1950.1. The row V is the only vector-space source, and H, T, P, C are the only componentwise transport, replay, provenance, and local naming rows available to a consumer.

Apply Definition 1950.2. Its classifier-transport row compares packet reads through $V, R, N, M, Q, H, T, P, C$ and never through host extensional equality.

Finally use Definition 1950.7. The vector-space source is transported through the norm row and the same structural rows before any downstream consumer can read it. Since the carrier has no quotient-completion, selected-limit, basis-choice, or host normed-space equality coordinate, the displayed transport route is forced. □

Definition 1950.13 (NormedSpace completion-window carrier)**.** *For an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet $S = (V, R, N, M, Q, H, T, P, C)$, the completion-window carrier is the finite subroute*

$$W_S = (V, R, N, M, Q, H, T, P, C)$$

read with Q as the Cauchy-window row and with V, R, N, M already displayed as the vector-space source, Real seal, norm row, and induced metric row. A Banach, Hilbert, bounded-operator, or functional-analysis completion request may inspect W_S only through BHist, hsame, Cont, Pkg, *and local* NameCert *transport; it is not a quotient completion, selected limit, ambient Banach carrier, Hilbert inner-product surface, or host normed-space equality.*

Lean4 (definition) `BEDC.Derived.NormedSpaceUp.NormedSpaceCompletionWindowCarrier`

Theorem 1950.14 (NormedSpace Cauchy-window handoff)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis completion request over an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet must first read the completion-window carrier of Definition 1950.13. The request receives the displayed* $\mathsf{VecSpace}^{\uparrow}$ *row V, terminal* $\mathsf{Real}^{\uparrow}$ *row R,* $\mathsf{Norm}^{\uparrow}$ *row N, induced* $\mathsf{Met}^{\uparrow}$ *row M, and Cauchy-window row Q through H, T, P, C before any completion-facing conclusion is available.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier cauchy window handoff`

Proof. Start with the source triad of Definition 1950.7. It requires vector-source reads, metric identity reads, and completion handoff reads to pass through the displayed rows $V, R, N, M, Q, H, T, P, C$.

Use the zero exactness and metric nonescape theorems, Definition 1950.3 and Definition 1950.5. They force the induced metric row to read through the vector zero classifier, Real zero and order cells, and real-valued norm row rather than through host equality.

Finally apply the completion handoff and vector transport obligation theorems, Definition 1950.6 and Definition 1950.12. The completion request can reach Q only after those rows have been transported and replayed through H, T, P, C, which is exactly the carrier W_S. □

Theorem 1950.15 (NormedSpace completion-window nonescape)**.** *For every accepted* NormedSpace$^{\uparrow}$ *packet, the completion-window carrier W_S is the whole completion-facing surface exported to Banach, Hilbert, bounded-operator, and functional-analysis consumers. No consumer can extract a quotient completion, selected completed point, host vector equality, basis choice, classical dual object, ambient Banach object, or Hilbert-space structure from W_S.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion window nonescape`

Proof. Read the handoff theorem Definition 1950.14. It places every completion request on W_S after the VecSpace, Real, Norm, induced Metric, and Cauchy-window rows have been displayed.

The existing nonescape boundaries then close the excluded routes. Metric nonescape blocks an independent metric source, completion handoff blocks a selected completed point or quotient object, and the root-unblock source triad blocks abstract source-equivalence replacement.

The vector transport obligation preserves the same finite route under BHist, hsame, Cont, Pkg, and local NameCert rows. Since none of the excluded host objects is a coordinate of the carrier, every listed downstream consumer remains inside W_S. □

Theorem 1950.16 (NormedSpace real-norm metric factorization)**.** *For every accepted* NormedSpace$^{\uparrow}$ *packet*

$$S = (V, R, N, M, Q, H, T, P, C),$$

each metric comparison exported by M factors through the real-valued norm route V, R, N before it reaches the structural rows H, T, P, C and the completion-facing row Q. Thus the identity, triangle, and scalar-size reads seen by a downstream consumer are BHist*-anchored* hsame *and* Cont *reads through the displayed rows, not host metric equality, quotient completion, selected limits, basis choice, or host normed-space equality.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier real norm metric factorization`

Proof. Begin with Definition 1950.5. The induced metric row M can read a distance only as the norm of a displayed vector difference through V, N, R.

The zero, triangle, and scalar rows then come from the established norm obligations. By Definition 1950.3, zero distance returns to the displayed vector-zero classifier in V after reading the real zero and order data in R. By Definition 1950.4, triangle and scalar-size reads pass through V, R, N.

Transport the resulting metric read through H, T, P, C, with completion-facing requests held in Q. The carrier contains no host metric equality, quotient completion, selected limit, basis choice, or host normed-space equality row, so the real-norm factorization is the only exported metric route. □

Theorem 1950.17 (NormedSpace completion request budget)**.** *Let*

$$S = (V, R, N, M, Q, H, T, P, C)$$

be an accepted NormedSpace$^{\uparrow}$ *packet. Every completion request exported by Q is budgeted by the already displayed vector, real, norm, and metric rows V, R, N, M and is transported through the same* BHist, hsame, Cont, Pkg, *and local* NameCert *rows H, T, P, C. The request may carry Cauchy windows and tolerance data only; it may not add quotient completion, selected limits, basis choice, host vector equality, host normed-space equality, or an ambient Banach/Hilbert object.*

Proof. Read the completion-facing row in Definition 1950.1. The row Q records Cauchy windows, tolerance requests, and handoff data, and it is attached to the rows V, R, N, M, H, T, P, C of the same packet.

Apply Definition 1950.6. Banach, Hilbert, bounded-operator, and functional-analysis consumers must read the displayed route V, R, N, M, Q before the structural rows H, T, P, C can export the handoff.

Use Definition 1950.11 to close the boundary. Because the packet has no quotient-completion, selected-limit, basis-choice, host vector equality, host normed-space equality, or ambient Banach/Hilbert coordinate, Q supplies only the finite request budget for later completion work. □

Theorem 1950.18 (NormedSpace completion budget envelope)**.** *Let $S = (V, R, N, M, Q, H, T, P, C)$ be an accepted* NormedSpace$^{\uparrow}$ *packet. Every completion request read by a downstream consumer is a* BHist*-anchored envelope over the displayed rows $V, R, N, M, Q, H, T, P, C$: the vector source, real-valued norm, induced metric, and completion-facing budget must be read before the request reaches any Banach, Hilbert, bounded-operator, or functional-analysis consumer. The envelope is exactly the route isolated by Definition 1950.16, Definition 1950.17, and Definition 1950.11; it cannot add quotient completion, selected limits, basis choice, host normed-space equality, or an ambient complete object.*

Proof. Start with Definition 1950.17. It places every completion-facing request in the row Q and keeps that request attached to V, R, N, M, H, T, P, C.

Use Definition 1950.16 to expose the source of the budget. The metric read is already downstream of the real-valued norm route through V, R, N, M, so Q cannot read a completion request independently of those rows.

Finally apply Definition 1950.11. Banach and Hilbert consumers receive only the displayed completion request, transported by the same history, replay, provenance, and local naming rows. Since the excluded host objects are absent from the carrier, the displayed rows form the whole envelope. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedspaceCompletionBudgetEnvelope`

Theorem 1950.19 (NormedSpace metric completion source)**.** *For every accepted* NormedSpace$^{\uparrow}$ *packet $S = (V, R, N, M, Q, H, T, P, C)$, each induced metric read that supplies a completion source factors through the real-norm route V, R, N, M and then through the completion-facing row Q. The metric source is therefore a* BHist*-anchored read through $V, R, N, M, Q, H, T, P, C$, with the same structural transport required by Definition 1950.16 and the same completion budget required by Definition 1950.17. It is not host metric equality, quotient vector equality, a selected Cauchy limit, or an ambient Banach/Hilbert metric.*

Proof. Read the induced metric row through Definition 1950.16. A distance exported by M is the norm of a displayed vector difference, so the source rows are V, R, N, M.

Attach the completion-facing row by Definition 1950.17. The request in Q can carry Cauchy windows and tolerance data only after the metric read has passed through those source rows.

The nonescape boundary of Definition 1950.11 then rules out an alternate metric source for Banach or Hilbert consumers. Transport and replay through H, T, P, C preserve the same finite route, so every metric completion source is read through the displayed packet. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedspaceMetricCompletionSource`

Theorem 1950.20 (NormedSpace functional-analysis handoff)**.** *Every functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$ *must receive the handoff through the displayed rows* $V, R, N, M, Q, H, T, P, C$*. The handoff first reads the vector carrier and real-valued norm, then the induced metric, then the completion-facing budget, and finally the* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The route is the common downstream surface of Definition 1950.16, Definition 1950.17, and Definition 1950.11; it exports no host duality theorem, basis choice, quotient completion, selected limit, or ambient complete normed space.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier functional analysis handoff`

Proof. Project the carrier of Definition 1950.1. The only available rows for a general functional-analysis consumer are $V, R, N, M, Q, H, T, P, C$.

The metric and completion parts of the handoff are already fixed. Apply Definition 1950.16 for the path from vector data to induced metric reads, then apply Definition 1950.17 for the completion-facing request budget.

Use Definition 1950.11 for the downstream boundary. It prevents Banach and Hilbert consumers from receiving a completed host object or selected limit. Since bounded-operator and functional-analysis consumers depend on the same displayed normed-space packet, their handoff is forced through the same rows. □

Theorem 1950.21 (NormedSpace carrier transport completion service)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$*, carrier transport through the componentwise* hsame *rows preserves the common* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$ *service rows. A completion-facing consumer transported along* H *therefore receives the same vector carrier, real-valued norm, induced metric, completion request budget, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *route, not a host equivalence of normed spaces or a selected completed object.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrierTransportCompletionService`

Proof. Project the accepted carrier. The rows V, R, N, M, Q are the displayed analytic service rows, while H, T, P, C transport, replay, source, and name them.

Use the componentwise clauses in H. They move the vector carrier, real row, norm row, metric row, and completion budget to a common finite comparison surface without changing which rows are exposed.

The completion service is then read through Q together with T, P, C. Since the carrier has no coordinate for a host equivalence or selected completion, transport preserves the displayed service route and nothing else. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier transport completion service`

Theorem 1950.22 (NormedSpace Real norm metric route)**.** *Every metric read induced by the norm of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$ *factors through the* $\mathsf{Real}^{\uparrow}$ *norm row* R, N*, the triangle ledger, the scalar compatibility row, the induced metric row* M*, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The metric route is therefore the finite norm route of the carrier, not host metric equality, quotient vector equality, basis coordinates, or an external ordered-field theorem.*

Proof. Begin with the normed-space carrier. The norm is not a detached metric; it is stored as the real-valued norm row N over the real row R and vector source V.

Apply the triangle and scalar compatibility obligations already displayed in the packet. They are the finite ledgers that allow a norm read to become a metric read through M.

The structural rows T, P, C replay, source, and name this route, while H keeps it stable under componentwise transport. Hence every induced metric read factors through the displayed Real norm route. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedspaceRealNormMetricRoute`

Theorem 1950.23 (NormedSpace functional-analysis root package)**.** *The displayed norm, induced metric, vector-space carrier, completion request budget, replay, provenance, and naming rows of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$ *form the root package consumed by Banach, Hilbert, bounded-operator, and functional-analysis chapters. Downstream chapters receive the finite rows* $V, R, N, M, Q, H, T, P, C$ *and their local* NameCert *surface; they do not receive a completed host normed space, a basis choice, a quotient completion, selected limits, or a classical duality theorem.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedspaceFunctionalAnalysisRootPackage`

Proof. Read the functional-analysis handoff of Definition 1950.20. It already places every downstream consumer behind the displayed rows $V, R, N, M, Q, H, T, P, C$.

The carrier transport service and Real norm metric route identify the two root interfaces needed by those consumers: completion-facing transport through Q, H, T, P, C and metric-facing norm reads through V, R, N, M.

Package the two interfaces with the local naming row. Banach, Hilbert, bounded-operator, and functional-analysis chapters may depend on that finite root package, but the excluded host objects are not exported by the carrier. □

Theorem 1950.24 (NormedSpace L10 completion obligation)**.** *Every L10-facing completion request exported by an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$ *must read the norm-to-metric route before it reads the completion-facing row* Q*. The request is therefore a finite dependency chain through* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*, with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It does not introduce abstract carrier-equivalence transport, selected completion points, quotient limits, or an ambient Banach carrier.*

Proof. Project the accepted packet. The vector row V, real row R, norm row N, and metric row M are already displayed before the completion request Q is available to a downstream consumer.

Use the Real norm metric route to bind M to the visible norm data. The L10 request then reads Q only as a completion-facing budget over that displayed metric route, with H, T, P, C providing transport, replay, provenance, and local naming.

Since no row of S contains carrier-equivalence transport, selected completion points, quotient limits, or a Banach carrier, the completion obligation remains the finite NameCert route through the displayed rows. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier l10 completion obligation`

Theorem 1950.25 (NormedSpace completion triad)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet* $S = (V, R, N, M, Q, H, T, P, C)$*, the displayed* $\mathsf{VecSpace}^{\uparrow}$ *row* V*,* $\mathsf{Norm}^{\uparrow}$ *row* N*, and induced* $\mathsf{Met}^{\uparrow}$ *row* M *jointly determine the completion-facing NormedSpace packet. The completion row* Q *may be consumed only after the vector operations, zero classifier, Real-valued norm, triangle ledger, scalar-compatibility ledger, and norm-induced metric route have been read through* V, R, N, M, H, T, P, C*.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion triad`

Proof. Project the carrier of Definition 1950.1. The vector row V, Real row R, norm row N, induced metric row M, and completion row Q are displayed coordinates of the same finite packet.

Use the zero exactness and compatibility routes already established above. The zero classifier is read through V, R, N, while the triangle and scalar-compatibility ledgers attach addition, scalar action, Real order, and norm reads to the same carrier.

Finally apply Definition 1950.24. The completion row Q is available only after the norm-to-metric route has been exposed, and H, T, P, C transport, replay, source, and name the resulting NormedSpace packet. □

Theorem 1950.26 (NormedSpace metric completion unblock)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet must read the norm-induced metric row* M *before it reads the completion-facing row* Q*. The consumer therefore receives a metric completion source through* $V, R, N, M, Q, H, T, P, C$*, not through an independent host metric, quotient vector equality, selected completion point, or ambient Banach carrier.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedspaceMetricCompletionUnblock`

Proof. Read the induced metric route of Definition 1950.22. It makes M a metric row generated from the Real-valued norm over the displayed vector source, rather than an independent metric coordinate.

The completion-facing handoff is then forced by Definition 1950.24. A completion request in Q is downstream of the metric row and remains attached to the same transport, replay, provenance, and local naming rows.

The nonescape boundary follows from Definition 1950.30. Since no row supplies a host metric, quotient equality, selected limit, or ambient Banach carrier, a MetricCompletion consumer can only use the displayed metric-completion route. □

Theorem 1950.27 (NormedSpace Banach and Hilbert consumer route)**.** *Every Banach or Hilbert consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet receives only the displayed VecSpace, Real, Norm, Metric, Completion, transport, replay, provenance, and local naming rows*

$$V, R, N, M, Q, H, T, P, C.$$

In particular, the consumer may request completion data through Q *and metric data through* M*, but it receives no completed host normed space, Hilbert inner product, quotient completion, selected limit, basis choice, or classical dual object.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier banach hilbert consumer route`

Proof. Start with Definition 1950.25. It identifies the VecSpace, Norm, and Metric rows that determine the completion-facing NormedSpace packet.

Use Definition 1950.26 for the completion entrance. Banach and Hilbert consumers must pass through the induced metric row M before reaching Q, and the same H, T, P, C rows preserve the route.

Finally apply Definition 1950.30. The carrier contains no host Banach object, Hilbert inner product, quotient completion, selected limit, basis choice, or dual object, so the displayed rows are the entire consumer surface. □

Theorem 1950.28 (NormedSpace Real metric transport)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet, a metric read transported along the Real-valued norm row must pass through* R, N, M, H, T, P, C*. The transported metric surface is the same displayed norm-induced metric route: it preserves the zero-row, triangle, scalar-compatibility, replay, provenance, and naming ledgers, and it does not pass through host metric equality, quotient vector identity, basis coordinates, or abstract carrier equivalence.*

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier real metric transport`

Proof. Start with the metric row M of the carrier. By construction it is induced from the Real-valued norm row N over R, rather than supplied as an independent host metric.

Transport the rows componentwise through H. The zero, triangle, and scalar compatibility ledgers remain attached to the same Real norm data, while T, P, C replay, source, and name the transported metric read.

The carrier contains no coordinate for host metric equality, quotient vector identity, basis coordinates, or carrier-equivalence transport. Thus the transported metric surface is exactly the displayed Real norm route. □

Theorem 1950.29 (NormedSpace completion-metric root unblock)**.** *For every accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet*

$$S = (V, R, N, M, Q, H, T, P, C),$$

each $\mathsf{MetricCompletion}^{\uparrow}$*, Banach, Hilbert, or bounded-operator consumer of* S *must read first through the displayed VecSpace row* V*, Real row* R*, norm row* N*, induced metric row* M*, and completion window row* Q*, and only then through* H, T, P, C*. The packet exports no quotient completion, selected limit, basis choice, host metric, or ambient Banach or Hilbert carrier.*

Proof. Project the carrier S. The vector source, Real-valued norm, induced metric, and completion-facing window are exactly the rows V, R, N, M, Q, while H, T, P, C supply transport, replay, provenance, and local naming.

Use zero exactness and metric nonescape first. The zero row is exact only along V, R, N, and the metric row M reads distances only as norm reads of displayed differences through the same source rows.

Next apply the completion handoff and the L10 completion obligation. They place every completion-facing request behind the route V, R, N, M, Q and prevent the request from selecting a completed point or quotient limit.

Finally use the Real metric transport theorem above. Transport preserves the Real norm route through R, N, M, H, T, P, C, so downstream MetricCompletion, Banach, Hilbert, and bounded-operator consumers receive only the displayed completion-metric root surface and none of the excluded host data. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier completion metric root unblock`

Theorem 1950.30 (NormedSpace Banach consumer nonescape)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet can read only the rows*

$$V, R, N, M, Q, H, T, P, C.$$

Such a consumer may request completion-facing data through Q *and metric data through* M*, but it cannot extract a completed host normed space, selected limit, quotient completion, basis choice, dual object, or Hilbert-space structure from the* $\mathsf{NormedSpace}^{\uparrow}$ *certificate alone.*

Proof. Use Definition 1950.23. It identifies the root package consumed by Banach, Hilbert, bounded-operator, and functional-analysis chapters as the displayed rows $V, R, N, M, Q, H, T, P, C$.

Separate the two consumer routes. Completion-facing reads pass through Q together with transport, replay, provenance, and naming; metric-facing reads pass through M together with the Real norm route.

No additional row remains for a completed host normed space, selected limit, quotient completion, basis choice, dual object, or Hilbert structure. Hence each listed consumer is bounded by the finite $\mathsf{NormedSpace}^{\uparrow}$ carrier. □

Lean4 ✓ `BEDC.Derived.NormedSpaceUp.NormedSpaceCarrier banach consumer nonescape`

VecSpace carrier transport obligation. Every $\mathsf{NormedSpace}^{\uparrow}$ consumer that needs the vector-space source must read the $\mathsf{VecSpace}^{\uparrow}$ carrier row V through the $\mathsf{Norm}^{\uparrow}$ row N and the structural rows H, T, P, C. The transported carrier is therefore a finite BHist route with hsame stability, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming; it is not a host vector-space equality or an abstract source-equivalence comparison.

Metric zero-row identity obligation. Every $\mathsf{Met}^{\uparrow}$ identity-of-indiscernibles read exported by M must pass through the zero norm row in N, the $\mathsf{Real}^{\uparrow}$ zero and order cells in R, and the displayed vector-zero classifier in V. The packet supplies metric identity only through this zero-norm route, not through quotient vector equality, host metric equality, basis coordinates, or an external ordered-field argument.

Completion handoff nonescape obligation. Every $\mathsf{Completion}^{\uparrow}$ request exported by Q is a handoff over the already displayed V, R, N, M rows. The completion-facing consumer receives Cauchy windows and tolerance requests transported by H, replayed by T, sourced by P, and named by C, but receives no quotient completion, selected completed point, choice principle, ambient Banach object, or Hilbert-space structure.

Theorem 1950.31 (NormedSpace public NameCert export)**.** *Every Banach, Hilbert, bounded-operator, or functional-analysis consumer of an accepted* $\mathsf{NormedSpace}^{\uparrow}$ *packet receives the public* $\mathsf{NormedSpace}^{\uparrow}$ *interface only through the displayed route*

$$V, R, N, M, Q, H, T, P, C.$$

The public export consists of the vector source, Real-valued norm, induced metric, completion-facing handoff, componentwise transport, displayed replay, provenance, and local $\mathsf{NameCert}$ *rows. It exports no completed host normed space, quotient completion, selected limit, basis choice, dual object, or Hilbert-space structure.*

Proof. Start from the Banach consumer nonescape theorem. Definition 1950.30 already fixes the only route available to Banach, Hilbert, bounded-operator, and functional-analysis consumers as $V, R, N, M, Q, H, T, P, C$.

Next project the three local obligations above. VecSpace carrier transport keeps V behind N, H, T, P, C, metric zero-row identity keeps M behind the Real norm route, and completion handoff nonescape keeps every completion-facing request behind Q and the same structural rows.

Therefore the exported public interface is exactly the displayed finite NameCert route. Since no additional carrier coordinate names a host completion, selected limit, basis, dual object, or Hilbert structure, every listed consumer receives $\mathsf{NormedSpace}^{\uparrow}$ only through the public rows stated above. □

Falsifiable prediction: An accepted $\mathsf{NormedSpace}^{\uparrow}$ consumer that reaches a Banach, Hilbert, bounded-operator, or functional-analysis read without first reading the displayed rows $V, R, N, M, Q, H, T, P, C$ refutes the carrier surface.

Independence witness: norm, normed-real-line, metricspace, cauchycompletion: no listed sibling supplies the full non-singleton packet $S = (V, R, N, M, Q, H, T, P, C)$, because each sibling contributes only a singleton norm instance, a one-dimensional real-line normed surface, a metric readback, or a completion handoff while this chapter binds the general normed-vector-space NameCert interface into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{NormedSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{NormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real-valued norm row, an induced Metric row, a completion-facing Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NormedSpace}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1950.1, Definition 1950.2, Definition 1950.6, and Definition 1950.31 bind the displayed VecSpace, Real norm, induced Metric, completion-facing Cauchy handoff, hsame, Cont, Pkg, and NameCert route for Banach, Hilbert, bounded-operator, and functional-analysis consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host normed-space equality, quotient completion, selected limits, basis choice, classical duality, Lean verification, or bridge checking.

Upgrade path: Lean verification and bridge checking of the mature VecSpace, Real norm, induced Metric, completion-facing Cauchy handoff, hsame, Cont, Pkg, and NameCert route remain the next obligations.

Chapter 1951

A Concrete Naming Certificate for $\mathsf{CauchyModulusPullback}^{\uparrow}$

Following Bishop and Bridges on explicit moduli and cofinal subsequences, the $\mathsf{CauchyModulusPullback}^{\uparrow}$ packet records the finite certificate by which a displayed Cauchy modulus is read along a cofinal reindexing route before any regular rational or real-name consumer may use its threshold. It sits between the source and target schedule discipline of chapter 1568, the explicit threshold surface of chapter 292, the finite observation rows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal boundary of chapter 38. The packet names only the finite pullback route; it is not a quotient of streams, not a selected subsequence limit, not countable choice, not host real equality, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1951.1 (Cauchy modulus pullback carrier)**.** *A* $\mathsf{CauchyModulusPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, I, K, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *schedule whose original explicit Cauchy modulus is already displayed.*
2. *I is the reindexed* $\mathsf{StreamName}^{\uparrow}$ *schedule obtained by reading S along the cofinal route.*
3. *K is the* $\mathsf{CauchyReindexing}^{\uparrow}$ *route row, containing the monotone and cofinal unary index data that connect I back to S.*
4. *M is the original* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row on S, read only after K has turned a target tolerance request into a source-window request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the target request, the source threshold selected by M, and the slack introduced by the route K.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row showing that the reindexed schedule remains a regular rational name after K, M, D have been consumed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the reindexing, modulus, dyadic tolerance, and regular readback rows have been displayed.*

8. *H records componentwise* hsame *transport for the source schedule, target schedule, reindexing route, modulus threshold, dyadic tolerance, regular readback, Real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route from a target precision request through $I, K, S, M, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusPullback}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusPullback}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, countable-choice selectors, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusPullbackUp.TasteGate.CauchyModulusPullbackTasteGate single carrier alignment`

Theorem 1951.2 (Cauchy modulus pullback NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusPullback}^{\uparrow}$ *carrier*

$$B = (S, I, K, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusPullback}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source schedule S, reindexed schedule I, reindexing route K, original modulus row M, dyadic tolerance row D, regular readback row R, Real seal row E, and structural rows H, C, P, N;*
2. *route-before-threshold discipline, requiring each target precision request to replay through I and K before it can consult the original modulus row M on S;*
3. *cofinal threshold pullback, requiring the threshold selected by M to be converted into a target-window bound only by the monotone and cofinal rows inside K;*
4. *dyadic tolerance compatibility, requiring D to record the source tolerance, target tolerance, and route slack used by the pulled-back threshold;*
5. *regular rational transport, requiring R to consume K, M, D before it exports any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
6. *terminal seal routing, requiring E to remain downstream of I, K, S, M, D, R and the structural replay rows;*
7. *ledger refusal for quotient streams, selected limits, countable choice, host real equality, trichotomy, and ambient Real completeness.*

No local pullback obligation is stored outside these rows.

Proof. Project the carrier of Definition 1951.1. The analytic coordinates are exactly the source schedule, target schedule, reindexing route, original modulus threshold, dyadic tolerance ledger, regular readback, and terminal seal.

Read the request in dependency order. A target precision request first exposes I, then the route K back to S, and only then the original modulus row M can be consulted.

The cofinal and monotone data inside K convert the source threshold into a target-window bound. The dyadic row D stores the tolerance comparison used during that conversion, and the row R consumes the converted window before any regular rational readback is exported.

Finally the seal row E is downstream of R. The transport, continuation, provenance, and naming rows replay and package the same finite route, so no excluded analytic coordinate remains available. □

Lean4 ✓ `BEDC.Derived.CauchyModulusPullbackUp.CauchyModulusPullbackCarrier namecert obligations`

Theorem 1951.3 (Cauchy modulus pullback reindex threshold)**.** *Let $B = (S, I, K, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusPullback}^{\uparrow}$ *carrier. Every pulled-back threshold consumed by the reindexed schedule I is read only after the* $\mathsf{CauchyReindexing}^{\uparrow}$ *route K has displayed the monotone and cofinal connection from I to S. Thus the original modulus row M cannot be used as a target-schedule threshold before the reindexing route has been replayed through C.*

Proof. Start with a target-schedule threshold request. The carrier projection gives the target schedule row I, but the threshold row M is a modulus row for the source schedule S.

Replay the continuation row C. The request must pass from I through K before it reaches S; the monotone row inside K controls the direction of target-window reads, while the cofinal row supplies the source-window reach needed for the requested tolerance.

Only after that route is displayed can M choose a source threshold. The dyadic row D then records the comparison that turns the source threshold into a target-window bound, and R, E, H, P, N consume only the resulting pulled-back row. □

Theorem 1951.4 (Cauchy modulus pullback Real seal nonescape)**.** *For every accepted* $\mathsf{CauchyModulusPullback}^{\uparrow}$ *carrier $B = (S, I, K, M, D, R, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$*-facing read factors through the finite route*

$$I \longmapsto K \longmapsto S \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

together with the structural rows H, C, P, N. The exported route contains no quotient stream, selected limit, countable-choice row, host real equality, trichotomy row, or ambient Real completeness principle.

Proof. Begin with a Real-facing consumer. Carrier admission exposes E only as a terminal row downstream of the regular readback row R.

Trace R backward through the packet. The regular readback consumes the dyadic tolerance row D, which in turn records the comparison between the reindexed request and the source threshold obtained from M.

The modulus row M is attached to the source schedule S, so the target schedule I reaches it only through the reindexing route K. This gives the displayed route from I to K, then to S, M, D, R, E.

The structural rows H, C, P, N preserve and name that route without adding an analytic source. Since the carrier of Definition 1951.1 has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient Real completeness, none of those objects can be exported by the Real-facing read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusPullback}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusPullback}^{\uparrow}$ packet is generated from a source StreamName schedule, a reindexed StreamName schedule, a CauchyReindexing route, an original CauchyModulus threshold row, DyadicRatCore tolerance rows, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1951.1, Definition 1951.2, Definition 1951.3, and Definition 1951.4 seed the finite certificate for moving explicit Cauchy modulus schedules across cofinal reindexing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold pullback, transport, replay, and ledger non-escape for the finite Cauchy modulus pullback surface.

Chapter 1952

A Concrete Naming Certificate for $\mathsf{CompactOperator}^{\uparrow}$

Following Riesz and Schauder, the $\mathsf{CompactOperator}^{\uparrow}$ packet records the finite BEDC surface by which a bounded linear operator between Banach carriers is witnessed through finite image-net schedules and tail moduli. It reads the bounded-operator row of chapter 1644, the Banach endpoint rows of chapter 90, the compact-witness rows already consumed by chapter 94, the operator-ideal dependency of chapter 213, and the compact-prefix consumer route of chapter 215. The chapter names only the reusable compact-operator certificate; it does not assert spectral theory, trace-class theory, choice-selected compactness, or host equality of operators.

Definition 1952.1 (Compact operator carrier)**.** *A* $\mathsf{CompactOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, B, N, M, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{Banach}^{\uparrow}$ *source row, including the carried source endpoints, source classifier, norm-induced metric rows, and visible completion ledger.*
2. *T is the* $\mathsf{Banach}^{\uparrow}$ *target row, including the carried target endpoints, target classifier, norm-induced metric rows, and visible completion ledger.*
3. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row recording the displayed operator action, linearity ledger, norm-bound endpoint, and bounded-operator provenance over S and T.*
4. *N is the finite image-net schedule: for each displayed precision request it records a finite family of target endpoints and the source-unit-bound rows whose images are covered by those endpoints.*
5. *M is the tail/modulus row binding the finite schedules in N to the target Banach metric and recording the precision transitions used by downstream consumers.*
6. *H records componentwise* hsame *transport for S, T, B, N, M and the structural rows.*
7. *C records displayed* Cont *replay of the route*

$$S, T, B \longmapsto N \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{CompactOperator}^\uparrow}$ *row.*

The classifier of the carrier compares accepted packets only through the displayed Banach endpoints, bounded-operator row, finite image-net schedules, tail/modulus rows, componentwise transport, continuation replay, provenance, and local naming row.

Lean4 ✓ `BEDC.Derived.CompactOperatorUp.CompactOperatorTasteGate single carrier alignment`

Theorem 1952.2 (Compact operator NameCert obligations)**.** *For every accepted* $\mathsf{CompactOperator}^\uparrow$ *carrier*

$$K = (S, T, B, N, M, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOperator}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Banach row, target Banach row, bounded-linear-operator row, finite image-net schedule, tail/modulus row, transport row, replay row, provenance row, and local naming row;*
2. *finite-image-net stability, requiring every transported precision request to replay through the same finite schedule row N and the same target metric coverage rows;*
3. *classifier transport, requiring componentwise* hsame *transport of the source endpoint, target endpoint, bounded-operator endpoint, image-net endpoints, and tail/modulus ledger;*
4. *ledger non-escape, requiring P and L to source no spectral row, trace-class row, choice-selected compactness row, host operator equality, or external compactness principle;*
5. *bounded-operator compatibility, requiring every compactness read to enter through the bounded-linear-operator row B before any finite image-net or tail/modulus row can be consumed.*

No compact-operator obligation is stored outside these rows.

Proof. Project the carrier of Definition 1952.1. The visible analytic coordinates are exactly the two Banach endpoint rows, the bounded-linear-operator row, the finite image-net schedule, and the tail/modulus ledger.

Finite-image-net stability is componentwise. A transported precision request keeps the same source-unit-bound row and target metric row after hsame transport, so replay through C reaches the same schedule coordinate N and the same tail/modulus coordinate M.

The classifier clauses are inherited row by row from the Banach and bounded-linear-operator surfaces. The source and target classifiers move endpoints, the bounded-operator classifier moves the operator and norm-bound ledger, and the compact schedule moves only through the displayed image-net endpoints and moduli.

Finally inspect P and L. Their source list is finite and contains no spectral, trace-class, choice, host-equality, or external compactness coordinate. Since every compactness read is downstream of B, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier namecert obligations`

Theorem 1952.3 (Compact operator functional-analysis route)**.** *Every compact-operator witness exposed by* $\mathsf{FunctionalAnalysis}^\uparrow$ *factors through an accepted* $\mathsf{CompactOperator}^\uparrow$ *carrier before it can be consumed by* $\mathsf{Nuclear}^\uparrow$ *or* $\mathsf{OperatorIdeal}^\uparrow$*. Equivalently, a downstream compact read must pass through the finite route*

$$\mathsf{Banach}^\uparrow, \mathsf{BoundedLinearOperator}^\uparrow \longmapsto \mathsf{CompactOperator}^\uparrow \longmapsto \mathsf{OperatorIdeal}^\uparrow, \mathsf{Nuclear}^\uparrow$$

where the middle packet supplies the image-net schedules, tail/modulus rows, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route is a certificate-level factorization, not host equality between operators.*

Proof. Begin with a compact witness from the $\mathsf{FunctionalAnalysis}^{\uparrow}$ surface. Its bounded-operator part is already a displayed $\mathsf{BoundedLinearOperator}^{\uparrow}$ row over Banach source and target endpoints, and its compactness evidence is a finite probe ledger for images of carried source-unit-bound rows.

Package those rows as in Definition 1952.1. The Banach endpoints supply S and T, the bounded-operator row supplies B, and the compact probe ledger supplies the finite schedule N together with its tail/modulus row M.

The downstream handoff is then forced by Definition 1952.2. $\mathsf{OperatorIdeal}^{\uparrow}$ may read the compact row only through the bounded-operator-compatible carrier, and $\mathsf{Nuclear}^{\uparrow}$ may read compact prefixes only after the same carrier has supplied its finite image-net and tail/modulus rows. No host operator equality is introduced in either handoff. □

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier functional analysis route`

Definition 1952.4 (Compact operator approximation scheme)**.** *A compact operator approximation scheme is a finite* BHist *packet*

$$A = (S, T, B, N, M, H, C, P, L)$$

whose rows display a source $\mathsf{Banach}^{\uparrow}$ *row S, a target* $\mathsf{Banach}^{\uparrow}$ *row T, a* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row B, a finite image-net schedule N, a tail/modulus ledger M, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and a local* $\mathsf{NameCert}_{\mathsf{CompactOperator}^{\uparrow}}$ *row L. The scheme is accepted only when N and M are read after B, so compactness is exposed by finite image-net and tail data rather than by host equality of operators or a choice-selected compactness witness.*

Lean4 (definition) `BEDC.Derived.CompactOperatorUp.CompactOperatorApproximationScheme`

Theorem 1952.5 (Compact operator bounded-image precompactness)**.** *Every accepted compact operator approximation scheme exposes bounded-image precompactness only through its displayed finite image-net schedule and tail/modulus rows. A consumer may read precompactness from $A = (S, T, B, N, M, H, C, P, L)$ only after the source row S, target row T, bounded-linear-operator row B, schedule row N, and ledger row M have been displayed.*

Proof. Use Definition 1952.4 to project the accepted scheme. The analytic rows available to a consumer are the Banach endpoints S, T, the bounded-linear-operator row B, the finite image-net schedule N, and the tail/modulus ledger M.

Apply Definition 1952.2. Its carrier admission and finite-image-net stability rows force every compactness read through N and M, and its bounded-operator compatibility row requires B before those compact rows can be consumed.

The remaining rows H, C, P, L only transport, replay, source, and name the same finite packet. Since none of them stores host operator equality or a choice-selected compactness coordinate, bounded-image precompactness is visible exactly through the displayed finite image-net and tail/modulus rows. □

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier bounded image precompactness`

Theorem 1952.6 (Compact operator ideal consumer boundary)**.** OperatorIdeal$^{\uparrow}$ *and* Nuclear$^{\uparrow}$ *consumers may read compactness from a* CompactOperator$^{\uparrow}$ *packet only after the* BoundedLinearOperator$^{\uparrow}$ *row and finite image-net schedule have been displayed. The consumer route passes through the Banach endpoints of chapter 90, the bounded operator surface of chapter 1644, and then the ideal and nuclear readers of chapter 213 and chapter 215; it does not pass through host operator equality or choice-selected compactness.*

Proof. Start with the compact operator carrier of Definition 1952.1. It displays Banach endpoint rows S, T and a bounded-linear-operator row B before the compact image-net schedule N and tail/modulus row M.

Use Definition 1952.3. The functional-analysis handoff already factors downstream compact reads through CompactOperator$^{\uparrow}$ before OperatorIdeal$^{\uparrow}$ or Nuclear$^{\uparrow}$ can consume them.

Finally apply Definition 1952.5. It pins precompactness to B, N, M and the finite transport and replay rows, so the ideal and nuclear consumers inherit compactness through the displayed schedule rather than through host operator equality, an external selector, or an unledgered compactness principle. □

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier ideal consumer boundary`

Theorem 1952.7 (Compact operator tail-modulus consumer exactness)**.** *For every accepted* CompactOperator$^{\uparrow}$ *carrier*

$$K = (S, T, B, N, M, H, C, P, L),$$

the tail/modulus row M is the exact consumer boundary for compactness reads after the finite image-net schedule N has been displayed. Any OperatorIdeal$^{\uparrow}$ *or* Nuclear$^{\uparrow}$ *consumer inspecting compactness through K must read the Banach endpoints S, T, the bounded-linear-operator row B, the finite schedule N, and then the displayed modulus ledger M before using the transport, replay, provenance, or local naming rows H, C, P, L.*

Proof. Project the carrier of Definition 1952.1. The only compactness rows available after the bounded-linear-operator row are the finite image-net schedule N and the tail/modulus ledger M.

Apply Definition 1952.2. Its finite-image-net stability clause fixes the schedule coordinate N, while its bounded-operator compatibility clause forces every compact read to pass through B before reaching N or M.

Use Definition 1952.6. The ideal and nuclear consumers inherit compactness only through the displayed compact-operator carrier. Since H, C, P, L merely transport, replay, source, and name the same rows, M is the exact final analytic ledger inspected by those consumers. □

Lean4 ✓ `BEDC.Derived.CompactOperatorUp.CompactOperatorTailModulusConsumerExactness`
Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier tail modulus consumer exactness`

Theorem 1952.8 (Compact operator finite-rank approximation boundary)**.** *The* CompactOperator$^{\uparrow}$ *carrier exposes finite-rank approximation data only as a boundary read from its displayed finite image-net schedules and tail/modulus rows. A consumer may treat*

$$K = (S, T, B, N, M, H, C, P, L)$$

as approximation-ready only when the source and target Banach rows S, T, the bounded-linear-operator row B, the finite image-net row N, and the modulus row M have all been displayed; the certificate does not introduce a host finite-rank projection, a selected approximating sequence, or operator equality outside K.

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier finite rank approximation boundary`

Proof. Read the approximation scheme of Definition 1952.4. It uses the same rows $S, T, B, N, M, H, C, P, L$ and accepts compactness only after N and M have been exposed downstream of B.

Apply Definition 1952.5. Bounded-image precompactness is visible exactly through the finite schedule and tail/modulus rows, so any finite-rank approximation boundary can only read the displayed image-net endpoints and precision ledger.

Finally use Definition 1952.6. The operator-ideal and nuclear handoffs inspect that same carrier surface. Because no row stores a host projection, selected approximating sequence, or external operator equality, the finite-rank approximation content remains a boundary of N and M rather than a new object. □

Theorem 1952.9 (Compact operator spectral refusal boundary)**.** *No* $\mathsf{CompactOperator}^{\uparrow}$ *consumer may read spectral data from the compactness certificate. For every accepted carrier*

$$K = (S, T, B, N, M, H, C, P, L),$$

the compactness surface available to $\mathsf{OperatorIdeal}^{\uparrow}$ *and* $\mathsf{Nuclear}^{\uparrow}$ *is exhausted by the displayed Banach rows, bounded-linear-operator row, finite image-net schedules, tail/modulus ledgers, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row; spectral values, eigenspaces, resolvents, and trace data are outside the carrier.*

Lean4 ✓ `BEDC.Derived.CompactOperatorCarrier spectral refusal boundary`

Proof. Inspect the non-escape clause in Definition 1952.2. The provenance and local naming rows source no spectral row, trace-class row, choice-selected compactness row, host operator equality, or external compactness principle.

Use Definition 1952.7. Downstream consumers inspect compactness only after S, T, B, N, M have been displayed, with H, C, P, L preserving that same finite packet.

The consumer boundary of Definition 1952.6 then blocks an $\mathsf{OperatorIdeal}^{\uparrow}$ or $\mathsf{Nuclear}^{\uparrow}$ reader from replacing the finite image-net and modulus route by spectral theory. Spectral values, eigenspaces, resolvents, and trace data therefore remain explicit non-claims of the compact-operator certificate. □

Theorem 1952.10 (Compact operator obligation scope)**.** *The* $\mathsf{CompactOperator}^{\uparrow}$ *obligation surface is scoped exactly to the functional-analysis route of Definition 1952.3, the approximation scheme of Definition 1952.4, the bounded-image precompactness boundary of Definition 1952.5, the ideal consumer boundary of Definition 1952.6, the tail-modulus exactness theorem of Definition 1952.7, the finite-rank approximation boundary of Definition 1952.8, and the spectral refusal boundary of Definition 1952.9. The only primitive rows in scope are the source and target* $\mathsf{Banach}^{\uparrow}$ *rows, the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row, the finite image-net schedule, the tail/modulus row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row of Definition 1952.1.*

No scoped obligation imports spectral data, trace-class data, host operator equality, choice-selected compactness, or an external compactness theorem.

Lean4 ✓ `BEDC.Derived.CompactOperatorUp.CompactOperatorObligationScope`

Proof. Project the carrier $K = (S, T, B, N, M, H, C, P, L)$ of Definition 1952.1. The carrier exposes only Banach endpoint rows, a bounded-linear-operator row, finite image-net schedules, a tail/modulus ledger, transport, replay, provenance, and local naming.

Apply Definition 1952.2. Carrier admission, finite-image-net stability, classifier transport, ledger non-escape, and bounded-operator compatibility are all stated over those same rows.

Use the functional-analysis route and approximation scheme, Definition 1952.3 and Definition 1952.4. They place every downstream compact read behind the Banach endpoints, bounded operator, finite schedules, and modulus ledger before any ideal or nuclear consumer is allowed to inspect compactness.

Finally combine Definition 1952.5, Definition 1952.6, Definition 1952.7, Definition 1952.8, and Definition 1952.9. These theorems exhaust the bounded-image, ideal, tail-modulus, finite-rank, and spectral-refusal boundaries without adding any row outside the displayed packet. □

Theorem 1952.11 (Compact operator public export status)**.** *The* $\mathsf{CompactOperator}^{\uparrow}$ *packet has a public export surface over the finite rows*

$$S, T, B, N, M, H, C, P, L.$$

Every public consumer of the compact-operator certificate must read the Banach endpoint rows, the bounded-linear-operator row, the finite image-net schedule, the tail/modulus ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before treating compactness as available to* $\mathsf{OperatorIdeal}^{\uparrow}$ *or* $\mathsf{Nuclear}^{\uparrow}$*.*

Proof. Use the carrier of Definition 1952.1. It displays exactly the rows $S, T, B, N, M, H, C, P, L$, with the compactness surface entering through the finite image-net schedule and tail/modulus ledger after the bounded-linear-operator row.

Apply the ideal consumer boundary and tail-modulus exactness theorems, Definition 1952.6 and Definition 1952.7. They force the public $\mathsf{OperatorIdeal}^{\uparrow}$ and $\mathsf{Nuclear}^{\uparrow}$ reads to pass through the same finite compact-operator packet rather than through host operator equality or an external compactness principle.

Finally use Definition 1952.10. Its scope statement binds the same rows to the Banach, bounded-linear-operator, functional-analysis, operator-ideal, and nuclear routes, so the displayed packet is public-facing while the verification axis remains the encoded definition cited in the closure status block. □

Lean4 ✓ `BEDC.Derived.CompactOperatorUp.CompactOperatorPublicExportStatus`

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOperator}^{\uparrow}$ packet is generated from source and target Banach rows, a bounded-linear-operator row, displayed finite image-net schedules, tail/modulus rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactOperator}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1952.10 and Definition 1952.11 close the mature finite image-net treatment over $S, T, B, N, M, H, C, P, L$ for the Banach, bounded-linear-operator, functional-analysis, operator-ideal, and nuclear sibling routes used by the compact-operator certificate.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactOperator}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.CompactOperatorUp`

Bridge status: none

Not claimed: This chapter does not close spectral theory, trace-class theory, choice-selected compactness, host operator equality, Lean-checked obligations, or bridge checking.

Upgrade path: The paper theory axis is mature for the finite image-net and tail-modulus export surface; bridgeChecked remains a separate formal-axis target for BEDC.Derived.CompactOperatorUp.

Chapter 1953

A Concrete Naming Certificate for $\mathsf{RapidCauchyName}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RapidCauchyName}^{\uparrow}$ packet records the finite rate-controlled Cauchy real-name surface used by L10 Real and Completion consumers before any terminal $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the finite window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, the terminal real-name seal of chapter 38, and the Real-facing modulus route of chapter 1423. The packet supplies an explicit rapid Cauchy naming certificate without quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Definition 1953.1 (Rapid Cauchy name carrier)**.** *A* $\mathsf{RapidCauchyName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (R_0, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *is the dyadic rate schedule row, recording the finite precision requests and the displayed rule that a request at level* k *is answered inside the dyadic tolerance budget assigned to* k*.*
2. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by the rate schedule for the visible requests.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic radii, slack comparisons, and arithmetic rows consumed by the selected windows.*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows* S *under the tolerance ledger* D*.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rate schedule, finite windows, tolerance ledger, and regular rational readback have been displayed.*
6. H *records componentwise* hsame *transport for the rate schedule, windows, tolerance ledger, readback, seal, replay, provenance, and local name.*
7. C *records displayed* Cont *replay from each finite precision request through* R_0, S, D, Q, E *and then through* H, P, N*.*
8. P *is the* Pkg *provenance over* $\mathsf{RapidCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RapidCauchyName}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $R_0, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.RapidCauchyNameUp.RapidCauchyNameTasteGate single carrier alignment`

Theorem 1953.2 (Rapid Cauchy name NameCert obligations)**.** *For every accepted* $\mathsf{RapidCauchyName}^{\uparrow}$ *carrier*

$$R = (R_0, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RapidCauchyName}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the dyadic rate schedule R_0, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N;*
2. *rate admission, requiring every precision request to enter through R_0 before any stream window, tolerance comparison, readback, or seal row is consumed;*
3. *finite-window selection, requiring S to contain only the windows selected by the displayed rate schedule;*
4. *dyadic accounting, requiring D to store exactly the tolerance radii and slack comparisons consumed by the selected windows and readback rows;*
5. *readback and seal routing, requiring Q and E to remain downstream of the displayed R_0, S, D route;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
7. *non-escape for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard-bridge data.*

No rapid Cauchy name obligation is stored outside these rows.

Proof. Project the carrier of Definition 1953.1. The analytic coordinates are exactly the dyadic rate schedule, finite windows, dyadic tolerance ledger, regular rational readback, and terminal seal.

The request order is forced by the tuple. A precision request first reads R_0, and the selected windows S are visible only as windows controlled by that schedule.

The tolerance and readback rows are downstream. The ledger D supplies the dyadic radii and slack comparisons consumed by S and Q, and the seal E is reached only after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1953.3 (Rapid Cauchy name dyadic rate route)**.** *Every precision request accepted by a* $\mathsf{RapidCauchyName}^{\uparrow}$ *carrier factors through the finite route*

$$R_0 \longmapsto S \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed dyadic rate schedule, selected finite $\mathsf{StreamName}^{\uparrow}$ *windows, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming. The route supplies no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy row, ambient completeness theorem, Lean evidence, or standard bridge.*

Proof. Begin with an accepted carrier and inspect the precision request. The only entrance for that request is the rate schedule row R_0, so the consumer cannot first read a detached stream window or a terminal Real seal.

Induct over the finite list of visible requests in the schedule. The empty list has no window or tolerance row to consume. In the cons step, the head request selects its finite window in S and reads its dyadic radius and slack comparisons in D; the induction hypothesis handles the remaining requests.

The regular readback Q is then obtained from the selected windows and tolerance ledger. The seal E and structural rows H, C, P, N close the same displayed route, and no carrier coordinate remains from which the excluded quotient, selector, choice, equality, order, completeness, verification, or bridge data could be read. □

Theorem 1953.4 (Rapid Cauchy name Real seal boundary)**.** *Let*

$$R = (R_0, S, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{RapidCauchyName}^{\uparrow}$ *carrier. Any downstream use of the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *must pass through the same dyadic rate schedule, selected finite windows, tolerance ledger, regular rational readback, and structural rows. Thus the seal cannot be used as a hidden endpoint for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Proof. Use the route of Definition 1953.3. It places E after R_0, S, D, Q, so every accepted seal read is attached to the displayed rate schedule, finite windows, tolerance accounting, and regular readback.

Now inspect the classifier surface of Definition 1953.1. The carrier has no second Real-seal coordinate and no coordinate for quotient equality, selected limits, countable choice, host equality, trichotomy, ambient completeness, Lean evidence, or bridge data.

Therefore a downstream read that bypasses one of R_0, S, D, Q, H, C, P, N is not a read of the accepted carrier. The only available terminal route is the finite route named by R. □

Falsifiable prediction: Every $\mathsf{RapidCauchyName}^{\uparrow}$ precision read factors through a dyadic rate schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, realcauchymodulus, real: $\mathsf{RapidCauchyName}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rate admission, finite-window selection, dyadic tolerance accounting, regular readback, terminal seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RapidCauchyName}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RapidCauchyName}^{\uparrow}$ packet is generated from a dyadic rate schedule, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RapidCauchyName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1953.1, Definition 1953.2, Definition 1953.3, and Definition 1953.4 seed the finite rapid Cauchy real-name route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RapidCauchyName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, rate admission, window selection, dyadic accounting, readback and seal routing, dyadic rate routing, and Real-seal non-escape for BEDC.Derived.RapidCauchyNameUp.

Chapter 1954

A Concrete Naming Certificate for $\mathsf{CauchyCompletionComparison}^{\uparrow}$

Following Bishop and Bridges, Kelley, and Bourbaki, the $\mathsf{CauchyCompletionComparison}^{\uparrow}$ packet records the finite BEDC surface by which a unit-and-universal-property completion presentation is compared with a functorial metric-completion presentation. It reads the dense unit route of chapter 1527, the universal-property route of chapter 397, the functorial action route of chapter 1937, the metric-completion boundary of chapter 1559, the finite stream window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names a comparison certificate for displayed completion presentations only; it does not identify quotient completions, choose representatives, assert host equality, or import completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1954.1 (CauchyCompletionComparison carrier)**.** *A* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, F, M, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-admission row for the unit-and-universal-property presentation.*
2. *V is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row recording the finite extension obligation and its displayed uniqueness comparison.*
3. *F is the* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *row recording the corresponding completed-map route on the functorial side.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row recording the metric-completion presentation whose finite Cauchy data are being compared with U, V, F.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row shared by the two presentations.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected before either presentation reaches a completion-facing seal.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and diameter ledger attached to the same stream windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the shared S, D, R route has been displayed.*

9. *H records componentwise* hsame *transport for U, V, F, M, R, S, D, E and the structural rows.*
10. *C records displayed* Cont *replay of the route*

$$U, V, F, M \longmapsto S, D, R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, CompletionUniversalPrope $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionComparisonUp.CauchyCompletionComparisonTasteGate single carrier alignment`

Theorem 1954.2 (CauchyCompletionComparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *carrier*

$$K = (U, V, F, M, R, S, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unit row, universal-property row, functorial row, metric-completion row, regular readback, stream windows, dyadic ledger, terminal seal, transport, replay, provenance, and local naming row;*
2. *shared-window coverage, requiring both presentations to inspect only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in S before reading D, R, E;*
3. *universal-to-functorial comparison, requiring any unit/universal-property completion consumer to pass through U, V before it is compared with the functorial row F;*
4. *metric presentation comparison, requiring any metric-completion consumer to pass through M and then the same S, D, R, E route;*
5. *classifier stability, requiring transported packets with componentwise matching rows to replay the same comparison through H and C;*
6. *non-escape, requiring P, N to source and name no quotient completion equality, selected representative, countable-choice schedule, host equality, open-cover compactness, or completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 1954.1. The accepted tuple displays exactly the four presentation rows U, V, F, M, the shared observation route S, D, R, E, and the structural rows H, C, P, N.

Shared-window coverage is proved by induction on the finite $\mathsf{StreamName}^{\uparrow}$ windows in S. The empty window exposes no rational readback. In the cons step, the next displayed window contributes only its attached $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the induction hypothesis replays the remaining windows.

The universal-to-functorial comparison is componentwise. The unit route U supplies the dense source admission, V supplies the extension and uniqueness surface, and F supplies the functorial completed-map route over the same displayed source data.

The metric presentation comparison uses the same induction on S and the same replay through D, R, E. Since M has no separate terminal seal coordinate in the carrier, a metric-completion read reaches the public comparison only through the shared finite route.

Finally transport and naming are read from H, C, P, N. These rows preserve the displayed coordinates and add no quotient, representative-choice, countable-choice, host-equality, compactness, or real-completeness coordinate, so the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionComparisonUp.CauchyCompletionComparisonCarrier namecert obligations`

Theorem 1954.3 (CauchyCompletionComparison route)**.** *Let*

$$K = (U, V, F, M, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *carrier. Every consumer comparing the unit/universal-property presentation with the functorial metric-completion presentation factors through the finite route*

$$U, V, F, M \longmapsto S, D, R \longmapsto E \longmapsto H, C, P, N.$$

The route is a comparison of displayed NameCert rows. It is not quotient completion equality, not selected representatives, not countable choice, not host equality of completed objects, not open-cover compactness, and not completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with a comparison consumer and project the carrier. The only unit/universal-property coordinates are U and V, and the only functorial coordinate is F.

The metric side is then forced through M. Any metric-completion read that reaches the comparison must expose the finite presentation row before the shared observation route is available.

We now replay the shared observation route by induction on S. At the base window there is no rational observation to compare. At each finite step, the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance in D and the $\mathsf{RegSeqRat}^{\uparrow}$ readback in R are transported componentwise, and the remaining windows are handled by the induction hypothesis.

The terminal seal is E. After E is reached, H and C preserve the route under displayed hsame transport and Cont replay, while P and N record provenance and local naming. No other carrier coordinate exists, so the route cannot export any excluded completion or choice principle. □

Theorem 1954.4 (CauchyCompletionComparison sibling handoff)**.** *For an accepted* $\mathsf{CauchyCompletionComparison}$ *carrier* $K = (U, V, F, M, R, S, D, E, H, C, P, N)$*, every sibling consumer from* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *receives exactly the rows named in Definition 1954.1. The handoff therefore preserves the shared finite comparison route and exports no quotient completion equality, selected representative, countable-choice schedule, host equality, open-cover compactness, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier and split the sibling consumers by their entry row.

Consumers entering through $\mathsf{CauchyCompletionUnit}^{\uparrow}$ or $\mathsf{CompletionUniversalProperty}^{\uparrow}$ read U, V first. They may continue to the functorial route F and the metric route M, but only through the displayed comparison coordinates.

Consumers entering through $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ read F, M first. They still must pass through the same stream, dyadic, rational, and real-seal route S, D, R, E before receiving a completion-facing comparison.

Consumers entering through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ are already inside the shared observation route. Induction on the finite window family S shows that such consumers can inspect only the displayed window, tolerance, readback, and seal rows, preserved componentwise by H and replayed by C.

The package row P and local naming row N source exactly these handoffs. Since the carrier has no coordinate for quotient equality, representative selection, countable choice, host equality, open-cover compactness, or real completeness, no sibling handoff can export one. □

Theorem 1954.5 (CauchyCompletionComparison seal agreement)**.** *For an accepted* $\mathsf{CauchyCompletionComparison}$ *carrier* $K = (U, V, F, M, R, S, D, E, H, C, P, N)$*, every comparison consumer that enters through the unit/universal-property route* U, V *and every comparison consumer that enters through the functorial metric-completion route* F, M *reaches the same displayed seal route*

$$S, D, R \longmapsto E.$$

Equivalently, the public comparison surface has one $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route after the presentation rows have been displayed. No second terminal seal, detached real route, quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness argument, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem is available to either consumer.*

Proof. Project the accepted carrier using Definition 1954.1. The presentation rows are exactly U, V, F, M, and the only analytic readback rows are S, D, R, E.

Apply the obligation surface of Definition 1954.2. The unit/universal-property consumer must expose U, V before reaching the shared-window row, while the functorial metric-completion consumer must expose F, M before reaching the same shared-window row.

Now apply the route theorem and sibling handoff, Definition 1954.3 and Definition 1954.4. Both entry routes are replayed through S, D, R and terminate at E, with H, C, P, N preserving and naming only that displayed route. Since the carrier stores no second seal or excluded completion principle, the seal route is common to both consumers. □

Theorem 1954.6 (CauchyCompletionComparison nonquotient boundary)**.** *Let* $K = (U, V, F, M, R, S, D, E, H, C,$ *be an accepted* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *carrier. Replacing the route*

$$U, V, F, M \longmapsto S, D, R \longmapsto E$$

by quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is outside the carrier and outside every* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}$ *handoff generated by* K*.*

Proof. The carrier definition lists the full packet, so every admitted row is one of $U, V, F, M, R, S, D, E, H, C, P, N$. None of these rows is a quotient equality, a representative selector, a host-equality witness, a choice schedule, a compactness proof, or an ambient completeness theorem.

The NameCert obligations of Definition 1954.2 require comparison consumers to expose the presentation rows and then the shared S, D, R, E route. The route theorem fixes this order, and the sibling handoff theorem shows that every sibling consumer receives only the rows already named by the carrier.

Thus any attempted replacement by one of the excluded principles would need a coordinate not present in the packet or in its local $\mathsf{NameCert}$ surface. Such a replacement is therefore not a comparison handoff of K. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionComparison}^{\uparrow}$ packet is generated from CauchyCompletionUnit, CompletionUniversalProperty, CauchyCompletionFunctoriality, and MetricCompletion presentation rows, shared finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1954.1, Definition 1954.2, Definition 1954.3, Definition 1954.4, Definition 1954.5, and Definition 1954.6 seed the finite comparison surface between unit/universal-property and functorial metric-completion presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, open-cover compactness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window coverage, presentation comparison, classifier stability, sibling handoff, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionComparisonUp.

Chapter 1955

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$ packet records the finite BEDC route by which a single regular Cauchy rational source presents two displayed modulus ledgers while producing one $\mathsf{Real}^{\uparrow}$ seal. The chapter sits after the finite window discipline of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, the modulus realizer surface of chapter 1918, the stability route of chapter 1927, and the terminal seal of chapter 38. Its purpose is to make the $\mathsf{Real}^{\uparrow}$ seal depend on the shared regular Cauchy source and the displayed cofinal comparison, not on an accidental choice of modulus ledger.

Definition 1955.1 (Regular Cauchy modulus-independence carrier)**.** *A* $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (R, S_0, S_1, D_0, D_1, J, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the single* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, exposing the regular rational approximants that both modulus ledgers read.*
2. *S_0 and S_1 are the two* $\mathsf{StreamName}^{\uparrow}$ *schedule rows used to request finite windows from the same source row R.*
3. *D_0 and D_1 are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus-ledger rows, including their displayed dyadic radii, slack comparisons, and finite precision requests.*
4. *J is the cofinal comparison row, recording the finite zig-zag by which requests through D_0 and requests through D_1 are compared over the shared source row R.*
5. *A is the unique* $\mathsf{Real}^{\uparrow}$ *seal row reached after the two schedules, two ledgers, and the cofinal comparison have been displayed.*
6. *H records componentwise* hsame *transport for the source row, both schedule rows, both dyadic modulus ledgers, the cofinal comparison row, the Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$R \longmapsto S_0, S_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto A.$$

8. *P is the* Pkg *provenance over* RegularCauchyModulusIndependence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyModulusRealizer$^{\uparrow}$, RegularCauchyModulusStability$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{RegularCauchyModulusIndependence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S_0, S_1, D_0, D_1, J, A, H, C, P, N$. It has no coordinate for a preferred modulus ledger, quotient stream equality, selected limits, countable choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusIndependenceUp.RegularCauchyModulusIndependenceTasteGate`
`single carrier alignment`

Theorem 1955.2 (Regular Cauchy modulus-independence NameCert obligations)**.** *For every accepted* RegularCauchyModulusIndependence$^{\uparrow}$ *carrier*

$$I = (R, S_0, S_1, D_0, D_1, J, A, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyModulusIndependence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the shared* RegSeqRat$^{\uparrow}$ *source row, the two* StreamName$^{\uparrow}$ *schedules, the two* DyadicRatCore$^{\uparrow}$ *modulus ledgers, the cofinal comparison row, the unique* Real$^{\uparrow}$ *seal, and the structural rows H, C, P, N;*

2. *shared-source exposure, requiring both schedules and both ledgers to project the same source row R before any seal-facing row is inspected;*

3. *two-modulus cofinal comparison, requiring J to display finite comparison data from the D_0 route to the D_1 route and back over the source row R;*

4. RegSeqRat$^{\uparrow}$ *handoff, requiring every ledger request to be read as a regular rational approximation from R rather than from a detached stream representative;*

5. Real$^{\uparrow}$ *seal factorization, requiring every seal consumer to pass through the shared source row and the cofinal comparison before reaching A;*

6. *transport, replay, and provenance, requiring H, C, P, N to preserve, replay, source, and name only the displayed rows of the packet;*

7. *ledger non-escape, requiring the certificate to export no preferred modulus ledger, quotient stream equality, selected limit, countable-choice schedule, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No modulus-independence obligation is stored outside these rows.

Proof. Project the accepted packet $I = (R, S_0, S_1, D_0, D_1, J, A, H, C, P, N)$. Carrier admission exposes one regular rational source row, two finite schedules, two dyadic modulus ledgers, one cofinal comparison row, one terminal Real seal, and the four structural rows.

The shared-source clause is read first. Both schedules S_0, S_1 request windows from R, and both dyadic ledgers D_0, D_1 are attached to those requests, so neither ledger can present an independent stream representative.

The cofinal clause is the row J. It records finite comparison data in both directions between the requests displayed by D_0 and D_1, always over the same source row R.

The handoff to Real$^{\uparrow}$ is downstream. A seal-facing consumer must follow the route through R, the two schedules, the two ledgers, and J before it can inspect A.

Finally, H, C, P, N transport, replay, source, and name this route without adding an excluded coordinate. Since the carrier has no row for a preferred ledger or for an ambient completeness principle, the listed rows exhaust the local NameCert surface. □

Theorem 1955.3 (Regular Cauchy modulus-independence Real seal factorization)**.** *For every accepted* $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$ *carrier*

$$I = (R, S_0, S_1, D_0, D_1, J, A, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer factors through*

$$R \longmapsto S_0, S_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto A$$

together with the structural rows H, C, P, N*. In particular, replacing the selected finite request route from* D_0 *by the displayed cofinal route through* D_1*, or conversely, leaves the exposed* $\mathsf{Real}^{\uparrow}$ *seal row* A *named by the same local certificate.*

Proof. Start with a consumer of the terminal Real seal. The carrier projection exposes A, but the displayed route places A after the two schedule rows, the two dyadic modulus ledgers, and the cofinal comparison row.

Trace the consumer backward from A. The only source coordinate available to either modulus ledger is R, and the only schedules available to the ledgers are S_0 and S_1.

The comparison step is J. A request that arrives through D_0 is compared to a request through D_1, and the converse comparison is also displayed, so the consumer receives a finite cofinal witness before the Real seal is read.

The structural rows preserve this factorization. The hsame row transports corresponding displayed components, the Cont row replays the same ordering, and the Pkg and NameCert rows name no extra modulus choice.

Thus the Real seal is a downstream endpoint of the shared regular-Cauchy source and its cofinal ledger comparison. It is not a certificate of a preferred modulus, quotient stream equality, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusIndependence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$ packet is generated from one $\mathsf{RegSeqRat}^{\uparrow}$ source row, two $\mathsf{StreamName}^{\uparrow}$ schedules, two $\mathsf{DyadicRatCore}^{\uparrow}$ modulus ledgers, one cofinal comparison row, one $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusIndependence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1955.1, Definition 1955.2, and Definition 1955.3 seed the finite L10 route separating Real seals from accidental modulus-ledger choices.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusIndependence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyModulusIndependenceUp.RegularCauchyModulusIndependenceTasteGate\_transport`

Bridge status: none

Not claimed: This chapter does not close preferred modulus selection, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-source exposure, two-modulus cofinal comparison, RegSeqRat handoff, Real-seal factorization, transport and replay coverage, provenance coverage, and ledger non-escape for BEDC.Derived.RegularCauchyModulusIndependenceUp.

Chapter 1956

A Concrete Naming Certificate for $\mathsf{TietzeExtension}^{\uparrow}$

Following Tietze, Urysohn, Kelley, and Bourbaki, the $\mathsf{TietzeExtension}^{\uparrow}$ packet records the finite BEDC route from displayed normal-space separation data to displayed continuous $\mathsf{Real}^{\uparrow}$-valued extension data. It sits above the normal-space carrier of chapter 1866 and the separation-function carrier of **??**, and it reads its function, codomain, and topological boundaries through chapter 130, chapter 38, chapter 88, chapter 1858, and chapter 1857. The packet names only a local finite extension certificate for a displayed closed-domain row and a bounded displayed function on that row; it does not select complements, invoke excluded middle, identify host functions extensionally, or appeal to ambient compactness.

Definition 1956.1 (Tietze extension carrier)**.** *A* $\mathsf{TietzeExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (N, D, B, U, M, G, R, H, K, P, L)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *source row, exposing the ambient* $\mathsf{Topology}^{\uparrow}$ *row and the finite closed-pair separation discipline used by the extension read.*
2. *D is the closed-domain row, displayed over the topology source as a finite closed-set witness rather than as a selected complement of an open row.*
3. *B is the bounded* $\mathsf{Real}^{\uparrow}$ *target-interval row for the input map on D, recording the two endpoint bounds and the displayed nonescape ledger for the interval.*
4. *U is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *ladder row, recording the finite sequence of separation-function reads consumed to extend the bounded data stage by stage.*
5. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the input map on the closed-domain surface, exposing only graph cells and continuity replay visible on D.*
6. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *extension-graph row, tying the ambient topology source to the extended* $\mathsf{Real}^{\uparrow}$*-valued graph and recording agreement with M on the displayed closed-domain row.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, certifying that every displayed extension value remains inside the bounded target interval named by B.*
8. *H records componentwise* hsame *transport for the normal-space source, closed-domain row, bounded interval row, Urysohn ladder, input map, extension graph, Real seal, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay from a Tietze-extension request through $N, D, B, U, M, G, R, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{TietzeExtension}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ZeroSet}^{\uparrow}$, $\mathsf{CozeroSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, D, B, U, M, G, R, H, K, P, L$. It has no coordinate for selected complements, excluded middle, host function extensionality, arbitrary open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.TietzeExtensionUp.TietzeExtensionTasteGate single carrier alignment`

Theorem 1956.2 (Tietze extension NameCert obligations)**.** *For every accepted* $\mathsf{TietzeExtension}^{\uparrow}$ *carrier*

$$E = (N, D, B, U, M, G, R, H, K, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the normal-space source, closed-domain row, bounded interval row, Urysohn ladder, input map, extension graph, Real seal, transport, replay, provenance, and local naming;*
2. *closed-domain exposure, requiring the extension request to display D over the topology source named by N before any ambient extension graph can be consumed;*
3. *Urysohn-step route, requiring every stage of the bounded extension read to pass through the* $\mathsf{UrysohnLemma}^{\uparrow}$ *ladder row U rather than through selected complements or excluded middle;*
4. *continuous endpoint exactness, requiring M and G to agree on the displayed closed-domain row and requiring the agreement read to use the visible* $\mathsf{ContinuousMap}^{\uparrow}$ *graph cells;*
5. *Real boundary nonescape, requiring R to preserve the bounded interval named by B for all displayed extension values;*
6. hsame, Cont, *and* Pkg *transport, requiring componentwise transport, displayed replay, and provenance to preserve exactly the listed rows and no hidden host-function data.*

No local Tietze-extension obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 1956.1. The accepted tuple displays exactly the normal-space source, closed-domain row, bounded interval, Urysohn ladder, input map, extension graph, Real seal, and structural transport, replay, provenance, and naming rows.

Closed-domain exposure follows from the order of the carrier. An extension request first reads N and then D over the same topological source; the extension graph row G is unavailable until the closed-domain row has been displayed.

The Urysohn-step route is the projection to U. Each bounded correction step is named as a $\mathsf{UrysohnLemma}^{\uparrow}$ ladder read, so the packet has no row from which a selected complement, excluded middle argument, or unrecorded separation theorem can be consumed.

Continuous endpoint exactness is read through M and G. The row M displays the original $\mathsf{Real}^{\uparrow}$-valued continuous graph on D, while G displays the ambient graph and its agreement with M along that closed-domain row.

Real boundary nonescape is the projection to B and R. The interval row records the endpoint bounds, and the Real seal row records that the displayed extension graph stays inside those bounds.

Finally, transport and provenance are exhausted by H, K, P, L. These rows move and replay exactly the displayed carrier coordinates, so no local obligation exports host function extensionality, arbitrary open-cover compactness, ambient Real completeness, Lean verification, or bridge checking. □

Theorem 1956.3 (Tietze extension Urysohn route)**.** *Every extension read exported by an accepted* $\mathsf{TietzeExtension}^{\uparrow}$ *packet factors through*

$$\mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{UrysohnLemma}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the local $\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}$ *row can be inspected. No extension read factors through a selected complement, excluded middle, host function extensionality, arbitrary open-cover compactness, or ambient Real completeness.*

Proof. Begin with an accepted carrier $E = (N, D, B, U, M, G, R, H, K, P, L)$. Carrier admission exposes the $\mathsf{NormalSpace}^{\uparrow}$ source N and the closed-domain row D before any ladder, graph, or Real-boundary row is available.

The next gate is U. This row is precisely the $\mathsf{UrysohnLemma}^{\uparrow}$ ladder consumed by the extension read, so every separation-function stage is read from the displayed normal-space route and not from a selected complement or excluded-middle branch.

The function gate is M and G. The input graph and the ambient extension graph are both $\mathsf{ContinuousMap}^{\uparrow}$ rows, and their agreement is visible only along the displayed closed-domain row.

The value gate is B and R. These rows tie the extension graph to the bounded $\mathsf{Real}^{\uparrow}$ interval and record nonescape inside that interval.

The replay and local naming rows H, K, P, L preserve this order. Since the classifier has no further coordinate, every exported extension read factors through $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with none of the excluded classical or host-level principles available. □

Closure status for $\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1956.1, Definition 1956.2, and Definition 1956.3 seed the finite L10 route from normal-space separation data through Urysohn ladder rows into bounded continuous $\mathsf{Real}^{\uparrow}$-valued extension data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TietzeExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected complements, excluded middle, host function extensionality, arbitrary open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-domain exposure, Urysohn-step routing, continuous endpoint exactness, Real boundary nonescape, transport stability, and non-escape proofs for BEDC.Derived.TietzeExtensionUp.

Constructive story: A $\mathsf{TietzeExtension}^{\uparrow}$ packet is generated from a $\mathsf{NormalSpace}^{\uparrow}$ source row, a closed-domain row, a bounded $\mathsf{Real}^{\uparrow}$ interval row, a $\mathsf{UrysohnLemma}^{\uparrow}$ ladder, $\mathsf{ContinuousMap}^{\uparrow}$ input and extension graph rows, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1957

A Concrete Naming Certificate for $\mathsf{TernaryExpansion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Cantor's middle-third construction, the $\mathsf{TernaryExpansion}^{\uparrow}$ packet records finite base-three digit-prefix rows as displayed BEDC readback data. It does not store a host infinite ternary stream, quotient real, endpoint convention selector, or chosen representative. Its role is to route inspected base-three prefixes through the finite stream surface of chapter 126, the base-three rational place-value ledger parallel to chapter 1417 and chapter 418, the dyadic comparison discipline of chapter 291, the regular rational handoff of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the retained-third consumer surface of chapter 1530.

Definition 1957.1 (TernaryExpansion carrier)**.** *A* $\mathsf{TernaryExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, W, V, Q, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite ternary digit ledger, whose rows record inspected base-three digits together with any local sign or integral-prefix convention carried by the packet.*
2. *W is the place-window ledger, listing the finite prefix windows and request precisions exposed to* $\mathsf{StreamName}^{\uparrow}$ *consumers.*
3. *V is the base-three rational place-value row that evaluates each inspected prefix as a rational sum with denominator a displayed power of three.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-comparison row that compares the base-three rational place-value row with the dyadic precision language consumed by downstream real readers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from the same digit, window, place-value, and dyadic-comparison rows.*
6. *A is the optional* $\mathsf{CantorSet}^{\uparrow}$ *retained-prefix row, present exactly when the inspected digits in D are the retained-third digits used by the Cantor set surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, W, V, Q, R and, when present, A have been displayed.*
8. *H is the componentwise* hsame *transport row for ternary digits, prefix windows, base-three place values, dyadic comparisons, regular handoff, retained-prefix handoff, and real-seal coordinates.*

9. *C is the displayed* Cont *replay row that reads finite ternary windows, place-value rows, dyadic comparisons, handoff rows, retained-prefix rows, and real seals without adding a host stream.*
10. *P is the* Pkg *provenance over* $\mathsf{TernaryExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TernaryExpansion}^{\uparrow}}$ *row.*

The carrier predicate accepts exactly such finite base-three readback packets. It has no coordinate for a completed ternary stream, quotient equality of expansions, endpoint uniqueness convention, selected representative, trichotomy principle, measure-theoretic normality row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.TernaryExpansionUp.TernaryExpansionTasteGate single carrier alignment`

Theorem 1957.2 (TernaryExpansion RegSeqRat readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TernaryExpansion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{TernaryExpansion}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads a finite ternary window, then the base-three place-value row for that window, then the dyadic comparison row, then the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and only then the displayed* $\mathsf{Real}^{\uparrow}$ *seal. The read uses no host infinite stream, quotient equality, selected representative, trichotomy decision, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $T = (D, W, V, Q, R, A, E, H, C, P, N)$. The $\mathsf{StreamName}^{\uparrow}$ entrance is the finite window row W together with the digit rows of D selected by that window, so a real-facing consumer cannot read a ternary prefix before a finite observation window has been displayed.

The place-value row V evaluates that same finite prefix as a rational sum with base-three denominators. The comparison row Q is the only route from this base-three ledger into the dyadic tolerance discipline used by $\mathsf{RegSeqRat}^{\uparrow}$ and later real-name consumers.

The handoff row R is therefore downstream of D, W, V, Q, and the seal row E is downstream of R. The rows H, C, P, N transport, replay, package, and locally name the same finite route without adding any external stream or real representative.

Thus every real-facing read factors through the displayed finite window, place-value, dyadic-comparison, regular-handoff, and real-seal rows. The excluded host stream, quotient equality, representative choice, trichotomy, and completeness principles are not coordinates of the carrier. □

Theorem 1957.3 (TernaryExpansion CantorSet handoff)**.** *When all inspected digits in a* $\mathsf{TernaryExpansion}^{\uparrow}$ *prefix are retained-third digits, the finite ternary prefix factors through the* $\mathsf{CantorSet}^{\uparrow}$ *retained-prefix surface before any* $\mathsf{CantorSet}^{\uparrow}$ *or Cantor-function consumer reads it. The handoff uses the same digit ledger D, window ledger W, retained-prefix row A, transport row H, replay row C, provenance row P, and local naming row N, and it exports no host subset, selected infinite branch, quotient real equality, endpoint uniqueness convention, or* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Assume the inspected prefix window of W selects only retained-third digit rows from D. By the carrier definition, precisely in this case the optional row A is present as the $\mathsf{CantorSet}^{\uparrow}$ retained-prefix handoff for that same finite window.

The continuation row C reads the digit window before it reads A, and the transport row H keeps the digit rows, the window, and the retained-prefix row componentwise attached. Hence a $\mathsf{CantorSet}^{\uparrow}$ consumer receives the same finite prefix, not a separate branch object.

The provenance row P and local certificate row N package the handoff over $\mathsf{TernaryExpansion}^{\uparrow}$ and $\mathsf{CantorSet}^{\uparrow}$. Any later Cantor-set or Cantor-function consumer therefore enters through the retained-prefix row already displayed by the ternary packet.

No host subset, selected infinite branch, quotient equality, endpoint convention, or completeness row is listed in the carrier, so none can be exported by this handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{TernaryExpansion}^{\uparrow}}$.

Constructive story: A $\mathsf{TernaryExpansion}^{\uparrow}$ packet is generated from finite ternary digit rows, place-window rows, base-three rational place-value rows, dyadic-tolerance comparison rows, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, an optional $\mathsf{CantorSet}^{\uparrow}$ retained-prefix handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TernaryExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1957.1, Definition 1957.2, and Definition 1957.3 seed the finite base-three prefix readback surface into $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CantorSet}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TernaryExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host ternary streams, quotient real equality, endpoint uniqueness, selected representatives, trichotomy, RealUp completeness, measure-theoretic normality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, componentwise classifier transport, dyadic comparison, RegSeqRat readback, RealUp seal, retained-prefix handoff, and non-escape targets for BEDC.Derived.TernaryExpansionUp.

Chapter 1958

A Concrete Naming Certificate for $\mathsf{CauchyFilterMeet}^{\uparrow}$

Following Bourbaki and Kelley filter terminology, the $\mathsf{CauchyFilterMeet}^{\uparrow}$ packet records the finite common-refinement operation by which two displayed Cauchy-filter packets over the same uniform surface are read as one refinement packet before any limit or completion-facing consumer inspects the route. It sits after the $\mathsf{UniformSpace}^{\uparrow}$ entourage surface of chapter 339, the source $\mathsf{CauchyFilter}^{\uparrow}$ packets of chapter 333, the basis generator of chapter 1501, and the limit boundary of chapter 1558; it reads finite windows through chapter 126, regular rational rows through chapter 301, dyadic tolerances through chapter 291, and terminal seals through chapter 38. The packet names a binary meet of displayed filter data, not quotient filters, selected limits, countable choice, host equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 1958.1 (Cauchy filter meet carrier)**.** *A* $\mathsf{CauchyFilterMeet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (U, F_0, F_1, B_0, B_1, J, S, R, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the shared* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row. It supplies the finite entourage requests, diagonal reads, and refinement reads through which both source filters are interpreted.*
2. *F_0 and F_1 are the two source* $\mathsf{CauchyFilter}^{\uparrow}$ *rows whose displayed requests are to be met.*
3. *B_0 and B_1 are the corresponding* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *rows. Each basis row exposes the finite basis requests, window data, regular readback, and dyadic tolerance ledger used by its source filter.*
4. *J is the binary common-refinement row. For every displayed pair of basis requests from B_0 and B_1, it names a displayed basis request refining both sources over the same entourage surface U.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected after the common-refinement row has been read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to verify that the common-refinement request is small enough for both Cauchy-filter sources.*
8. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff row available only after $U, F_0, F_1, B_0, B_1, J, S, R, D$ have been displayed.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by a Real-facing consumer through the same handoff route.*

10. *H records componentwise* hsame *transport for the uniform row, both source-filter rows, both basis rows, the common-refinement row, the window, readback, tolerance, handoff, seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a pair of source-filter requests through $U, F_0, F_1, B_0, B_1, J, S, R, D, L, E$ in that order.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterMeet}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterMeet}^{\uparrow}}$ *row.*

The classifier compares accepted meet packets only through the displayed route

$$U, F_0, F_1, B_0, B_1, J, S, R, D, L, E \longmapsto H, C, P, N.$$

No carrier coordinate stores a quotient filter, selected limit, countable-choice schedule, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.CauchyFilterMeetUp.CauchyFilterMeetTasteGate single carrier alignment`

Theorem 1958.2 (Cauchy filter meet NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterMeet}^{\uparrow}$ *carrier*

$$M = (U, F_0, F_1, B_0, B_1, J, S, R, D, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterMeet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the shared uniform row, two source Cauchy-filter rows, two Cauchy-filter-basis rows, common-refinement row, window row, regular readback row, dyadic tolerance row, limit handoff, terminal seal, transport, replay, provenance, and local naming row;*
2. *same-uniform-source discipline, requiring both source filters and both basis rows to be read over the shared entourage surface U before a meet request is accepted;*
3. *binary common-refinement coverage, requiring every pair of displayed basis requests from B_0 and B_1 to pass through J and name a displayed request refining both;*
4. *finite observation exactness, requiring the refinement request to read only S, R, D before it reaches the limit handoff L and terminal seal E;*
5. *transport and replay stability, requiring H and C to preserve the route order under componentwise* hsame *transport and displayed* Cont *replay;*
6. *provenance exhaustion, requiring P, N to name only this finite common-refinement route and to export no quotient filter, selected limit, countable choice, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyFilterMeetUp.CauchyFilterMeetCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1958.1. The projection exposes exactly the shared uniform row, both source filters, both basis generators, the common-refinement row, the finite observation rows, the limit and seal rows, and the structural rows H, C, P, N.

The same-uniform-source discipline is fixed by the row order. A meet request first reads U, then reads F_0, F_1 and B_0, B_1 as source data over that same uniform surface before any common-refinement row is available.

Binary common-refinement coverage is the role of J. For a displayed pair of basis requests, J names the refining request and the two refinement reads back to the source requests. The row is finite data in the carrier, not an appeal to an ambient filter lattice.

Finite observation exactness follows by replaying C after J is read. The refining request passes through the displayed $\mathsf{StreamName}^{\uparrow}$ windows S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D before it may reach L or E.

Finally H, C, P, N cannot enlarge the packet. They transport, replay, source, and name the displayed rows only, so the excluded quotient, choice, host-equality, real-completeness, Lean-verification, and bridge-checking data are outside the local NameCert surface. □

Theorem 1958.3 (Cauchy filter meet common-refinement factorization)**.** *Let*

$$M = (U, F_0, F_1, B_0, B_1, J, S, R, D, L, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyFilterMeet}^{\uparrow}$ *carrier. Every completion-facing or Real-facing consumer of the binary meet factors through the finite route*

$$U, F_0, F_1, B_0, B_1 \longmapsto J \longmapsto S, R, D \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed common refinement of the two Cauchy-filter packets before it receives a $\mathsf{CauchyFilterLimit}^{\uparrow}$ *handoff or a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The factorization exports no quotient filters, selected limits, countable-choice schedule, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge checking.*

Proof. Begin with a completion-facing or Real-facing consumer exposed by P. Carrier projection shows that the consumer can enter the meet packet only through the displayed rows of Definition 1958.1.

The source part of the route is forced through U, F_0, F_1, B_0, B_1. These rows display the shared entourage surface, the two Cauchy-filter packets, and the two finite basis generators whose requests are being compared.

We now replay the common finite-refinement row. For each displayed pair of basis requests, J supplies the refining request and the two reads back to B_0 and B_1. The replay uses only the stored row J and therefore stays inside the same finite packet.

The observation part of the route is S, R, D. The displayed window, regular readback, and dyadic tolerance rows verify that the refining request remains small enough for both Cauchy sources before the limit boundary is inspected.

Only after this route has been replayed can the packet expose L, and only after L can it expose E. Componentwise transport through H and continuation replay through C preserve the same route, while P, N record provenance and local naming without adding any excluded coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterMeet}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterMeet}^{\uparrow}$ packet is generated from a shared UniformSpace row, two CauchyFilter source rows, two CauchyFilterBasis rows, a binary common-refinement row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a CauchyFilterLimit handoff, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyFilterMeet}^{\uparrow}}$, seed)

Scope closed: Definition 1958.1, Definition 1958.2, and Definition 1958.3 seed the finite common-refinement operation for two displayed Cauchy-filter packets over a shared uniform surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyFilterMeet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, same-uniform-source discipline, binary common-refinement coverage, finite observation exactness, limit handoff factorization, transport stability, replay stability, and ledger refusal for BEDC.Derived.CauchyFilterMeetUp.

Chapter 1959

A Concrete Naming Certificate for $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$

Following Brouwer, Kleene, and Bishop–Bridges, the $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$ packet records the finite branch-schedule boundary where a finitely branching prefix tree is read as a Cauchy inspection route before any Real or Completion consumer may ask for a terminal seal. It sits after the prefix-tree surface of chapter 1622, the finite fan-depth ledger of chapter 400, the spread-law discipline of chapter 1532, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite boundary for branch-indexed Cauchy reads; it does not choose an infinite path, identify quotient streams, invoke countable choice, assert open-cover compactness, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1959.1 (Finite branching Cauchy tree carrier)**.** *A* $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, B, F, S, R, D, E, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the prefix-tree row. It records the finite* $\mathsf{KleeneTree}^{\uparrow}$ *branch-prefix cells, root cell, successor cells, and prefix-closure checks visible to the packet.*
2. *B is the branch-window row. For each displayed prefix in K, it records the finite successor window, branch-depth bound, and inspection address consumed from the* $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{SpreadLaw}^{\uparrow}$ *surfaces.*
3. *F is the finite branch-schedule row. It states that every regular branch read used by the packet enters through a displayed prefix in K and a displayed window in B.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row opened by the scheduled branch windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the stream windows selected by F.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that assigns dyadic radii and Cauchy slack checks to the branch windows before R can be inspected.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, B, F, S, R, D have been displayed.*
8. *O is the obstruction row. It records the local refusal of path choice, quotient stream equality, countable choice, open-cover compactness, and ambient real-completeness coordinates.*

9. *H records componentwise* hsame *transport for the prefix tree, branch windows, schedule, stream windows, regular readback, dyadic tolerances, terminal seal, obstruction row, continuation, provenance, and local naming row.*

10. *C records displayed* Cont *replay from any regular-branch request through*

$$K \longmapsto B \longmapsto F \longmapsto S \longmapsto R \longmapsto D \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteBranchingCauchyTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed route

$$K, B, F, S, R, D, E \longmapsto O, H, C, P, N.$$

No carrier coordinate stores an infinite path selector, quotient stream equality, countable-choice schedule, open-cover compactness proof, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge credential.*

Lean4 ✓ `BEDC.Derived.FiniteBranchingCauchyTreeUp.FiniteBranchingCauchyTreeTasteGate single carrier alignment`

Theorem 1959.2 (Finite branching Cauchy tree NameCert obligations)**.** *For every accepted* $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$ *carrier*

$$T = (K, B, F, S, R, D, E, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteBranchingCauchyTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix tree, branch-window row, finite schedule row, stream windows, regular rational readback, dyadic tolerance ledger, terminal Real seal, obstruction row, transports, replay, provenance, and local naming;*

2. *prefix-tree closure, requiring every branch-prefix read to be admitted by K before any successor window or Cauchy row is visible;*

3. *finite branching coverage, requiring each scheduled branch window in B to expose only finitely many displayed successors and a displayed depth bound;*

4. *spread-law compatibility, requiring the schedule row F to respect the prefix discipline supplied by the* $\mathsf{SpreadLaw}^{\uparrow}$ *sibling surface;*

5. *Cauchy-window exactness, requiring every regular branch read to pass through S, R, D before the Real seal E can be consumed;*

6. *obstruction stability, requiring O, H, C, P, N to preserve the refusal of path choice, quotient stream equality, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No finite-branching Cauchy-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1959.1. The accepted packet lists exactly the prefix tree, branch windows, finite schedule, Cauchy readback rows, terminal seal, obstruction row, and structural rows.

Prefix-tree closure is read from K. A branch-prefix request has no independent entry point; it must first be displayed as a finite prefix-tree cell with root, successor, and prefix-closure checks.

Finite branching coverage is the role of B. For each displayed prefix, the packet stores a finite successor window and a depth ledger, so a consumer cannot replace the window by a hidden infinite branch.

Spread-law compatibility is then enforced by F. The finite schedule reads only prefixes accepted by K and windows accepted by B, and it replays the sibling $\mathsf{SpreadLaw}^{\uparrow}$ discipline before any analytic row is exposed.

Cauchy-window exactness follows from the route through S, R, D. The stream windows are opened first, the regular rational row packages their readback, and the dyadic tolerance ledger checks the Cauchy slack before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Finally O, H, C, P, N do not add new coordinates. They record refusal, transport, replay, provenance, and local naming for the displayed rows only, so the excluded path-choice, quotient, choice, compactness, real-completeness, Lean-verification, and bridge-checking objects remain outside the NameCert surface. □

Theorem 1959.3 (Finite branching Cauchy tree regular-branch boundary)**.** *Let*

$$T = (K, B, F, S, R, D, E, O, H, C, P, N)$$

be an accepted $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$ *carrier. Every regular-branch read exported by* T *factors through the displayed finite tree and Cauchy rows*

$$K \longmapsto B \longmapsto F \longmapsto S \longmapsto R \longmapsto D \longmapsto E$$

before it is visible to a $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or Completion-facing consumer. Consequently the read is a finite prefix-tree and Cauchy-window boundary, not a selected path, quotient stream equality, countable-choice schedule, open-cover compactness proof, or ambient real-completeness witness.*

Proof. Start with a regular-branch request exported by P. Carrier projection shows that the only available entry point for such a request is the prefix-tree row K.

The request then enters the finite branching boundary. The row B supplies the successor window and depth ledger for the displayed prefix, and the schedule row F records that this is the window used by the branch read.

We now pass to the Cauchy rows. The scheduled window opens S, the regular rational readback is then formed in R, and the dyadic tolerance row D records the Cauchy slack attached to that read.

Only after this finite route has been replayed is the terminal seal E visible. Componentwise transport through H and continuation replay through C preserve the same route, while O keeps the excluded coordinates absent.

Thus every exported regular-branch read factors through K, B, F, S, R, D, E together with the structural rows O, H, C, P, N. No selected path, quotient stream equality, countable-choice schedule, open-cover compactness theorem, ambient real-completeness theorem, Lean verification, or bridge credential is produced. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteBranchingCauchyTree}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteBranchingCauchyTree}^{\uparrow}$ packet is generated from a finite KleeneTree prefix row, branch-window and spread-law schedule rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a terminal RealUp seal, an obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteBranchingCauchyTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1959.1, Definition 1959.2, and Definition 1959.3 seed the finite branch-schedule boundary between prefix trees, branch windows, regular rational readback, dyadic tolerances, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteBranchingCauchyTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path choice, quotient stream equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-tree closure, finite branching coverage, spread-law compatibility, Cauchy-window exactness, regular-branch factorization, obstruction stability, and a Lean carrier/obligation formalization for BEDC.Derived.FiniteBranchingCauchyTreeUp.

Chapter 1960

A Concrete Naming Certificate for $\mathsf{RiemannIntegrable}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive analysis, the $\mathsf{RiemannIntegrable}^{\uparrow}$ packet names finite Darboux-gap evidence that an integrand admits integration data without already choosing an integral value. It sits over the terminal real-name seal of chapter 38, the dyadic partition ledger of chapter 1863, the Darboux finite-sum boundary of chapter 1880, the tagged-mesh integration route of chapter 1526, and the regulated approximation handoff of chapter 1906. The packet gives downstream integration chapters a displayed finite gap interface without importing a selected integral value, host function quotients, measure-theoretic integration, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1960.1 (Riemann integrable carrier)**.** *A* $\mathsf{RiemannIntegrable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (D, M, L, U, G, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the real-domain row, read through the terminal* $\mathsf{Real}^{\uparrow}$ *seal and the interval-facing data carried by chapter 38.*
2. *M is the* $\mathsf{DyadicPartition}^{\uparrow}$ *ledger naming the finite ordered dyadic partition on which the Darboux comparison is witnessed.*
3. *L and U are the lower and upper Darboux-sum rows, read through the* $\mathsf{DarbouxSum}^{\uparrow}$ *boundary of chapter 1880 over the partition row M.*
4. *G is the finite Darboux-gap ledger, displaying a dyadic tolerance request together with the comparison that the upper row U and lower row L meet that request.*
5. *T is the tagged-mesh and regulated-function handoff row, exposing only the* $\mathsf{RiemannIntegral}^{\uparrow}$ *and* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumers that may later use the displayed gap evidence.*
6. *H records componentwise* hsame *transport for the real-domain row, partition row, lower and upper Darboux rows, gap ledger, handoff row, continuation replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay from the real-domain row to the partition row, from the partition row to the lower and upper Darboux rows, from those rows to the gap ledger, and from the gap ledger to the downstream handoff row.*

8. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrable}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, L, U, G, T, H, C, P, N$. It has no coordinate for a selected integral value, quotient equality of host functions, measure-theoretic integration, open-cover compactness, a choice-selected partition family, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegrableUp.RiemannIntegrableTasteGate single carrier alignment`

Theorem 1960.2 (Riemann integrable NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrable}^{\uparrow}$ *carrier*

$$J = (D, M, L, U, G, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannIntegrable}^{\uparrow}}$ *surface consists exactly of carrier admission, real-domain admission, dyadic partition admission, lower and upper Darboux-sum compatibility, finite gap routing, tagged-mesh and regulated-function handoff, componentwise transport, continuation replay, provenance, local naming, and non-escape from selected integral values, host function quotients, measure-theoretic integration, open-cover compactness, choice-selected partition families, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the projection of Definition 1960.1. The accepted packet displays only the real-domain row D, partition ledger M, lower and upper Darboux rows L, U, gap ledger G, downstream handoff T, transports H, continuations C, provenance P, and local name N.

The analytic rows are finite. The real-domain row is read through $\mathsf{Real}^{\uparrow}$, the partition row is read through $\mathsf{DyadicPartition}^{\uparrow}$, and the two Darboux rows are read through the $\mathsf{DarbouxSum}^{\uparrow}$ surface over the same displayed partition.

The gap obligation is then the row G. It records that the displayed upper and lower rows answer the carried dyadic tolerance request before any tagged-mesh or regulated-function consumer may inspect T.

Transport and replay preserve the same route componentwise. Since the carrier has no coordinate for an integral value, a host quotient, a measure object, open-cover compactness, a choice-selected partition family, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 1960.3 (Riemann integrable Darboux-gap route)**.** *Every accepted* $\mathsf{RiemannIntegrable}^{\uparrow}$ *packet exposes integration-facing evidence only through the finite route*

$$D \longmapsto M \longmapsto (L, U) \longmapsto G \longmapsto T.$$

The handoff row T may be consumed by $\mathsf{RiemannIntegral}^{\uparrow}$ *or* $\mathsf{RegulatedFunction}^{\uparrow}$ *chapters only after the real-domain row, dyadic partition ledger, lower and upper Darboux rows, and finite Darboux-gap ledger have all been displayed.*

Proof. Start with an integration-facing consumer request. The continuation row C first replays the real-domain row D and then the finite partition row M, so the request is already confined to displayed real and dyadic partition data.

The rows L and U are read next through the $\mathsf{DarbouxSum}^{\uparrow}$ boundary. They use the same partition row M, hence the upper-minus-lower comparison is a finite Darboux comparison over displayed cells.

Only after these rows are present can the gap ledger G be read. The row G supplies the finite tolerance evidence that downstream consumers need, and the handoff row T records the permitted $\mathsf{RiemannIntegral}^{\uparrow}$ and $\mathsf{RegulatedFunction}^{\uparrow}$ uses of that evidence.

Thus the packet exports integrability evidence without exporting an already chosen integral value. The excluded host objects are absent from the carrier, so no consumer can recover them from the Darboux-gap route. □

Lean4 ✓ `BEDC.Derived.RiemannIntegrableUp.RiemannIntegrableCarrier darboux gap route`

Theorem 1960.4 (Riemann integrable gap-ledger monotonicity)**.** *Let*

$$J = (D, M, L, U, G, T, H, C, P, N)$$

be an accepted $\mathsf{RiemannIntegrable}^{\uparrow}$ *carrier. If the Darboux-gap row G answers a displayed dyadic tolerance ε, and if another displayed dyadic tolerance ε' is transported above ε through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison route, then the same rows $D, M, L, U, T, H, C, P, N$ answer the weaker tolerance request ε'.*

Proof. Project the carrier. The lower and upper Darboux rows L and U are read over the same finite partition ledger M, while the row G records the displayed comparison answering ε.

Transport the weaker tolerance request along the componentwise row H. Since ε' is displayed above ε in the same dyadic comparison route, the comparison already recorded in G remains sufficient after transport.

Replay the continuation row C with the same real-domain row D, partition row M, Darboux rows L, U, handoff row T, provenance row P, and local name row N. The handoff row T is downstream of the gap ledger, so no selected integral value, host function quotient, or separate integration object is introduced while weakening the tolerance. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrable}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegrable}^{\uparrow}$ packet is generated from a real-domain row, a finite dyadic partition ledger, lower and upper Darboux-sum rows, a finite Darboux-gap ledger, a tagged-mesh and regulated-function handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1960.1, Definition 1960.2, and Definition 1960.3 seed the finite Darboux-gap integrability interface over $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, and BHist.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrable}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected integral values, host function quotients, measure-theoretic integration, open-cover compactness, choice-selected partition families, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, real-domain admission, dyadic partition admission, Darboux-sum compatibility, finite gap routing, downstream handoff, non-escape, and field faithfulness for BEDC.Derived.RiemannIntegrab

Chapter 1961

A Concrete Naming Certificate for $\mathsf{PeriodicContinuedFraction}^{\uparrow}$

Following Lagrange and Hardy–Wright on periodic continued fractions, the $\mathsf{PeriodicContinuedFraction}^{\uparrow}$ packet records finite partial-quotient prefixes, a nonempty repeating block, convergent recurrence rows, rational-convergent windows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and a terminal $\mathsf{Real}^{\uparrow}$ seal. It refines the partial-quotient source of chapter 342, the recurrence ledger of chapter 1777, the denominator-bounded approximation route of chapter 1871, the rational-tree boundary of chapter 1704, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet is a finite periodic-block route into rational approximation; it is not a host quadratic irrational, a selected infinite expansion, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1961.1 (PeriodicContinuedFraction carrier)**.** *A* $\mathsf{PeriodicContinuedFraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, B, L, Q, D, S, G, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the partial-quotient prefix row, recording the finite initial block and the inspected continued-fraction source inherited from* $\mathsf{ContinuedFraction}^{\uparrow}$*.*
2. *B is the nonempty period-block row, recording the finite repeating block together with the local phase coordinate used to unfold displayed windows.*
3. *L is the recurrence and convergent ledger, recording numerator and denominator recurrence rows for the prefix followed by repetitions of B.*
4. *Q is the rational normalization row, recording the normalized rational convergents exported by the recurrence ledger.*
5. *D is the denominator-bound comparison row, recording the finite denominator budgets and comparison witnesses used by the Dirichlet-style approximation surface.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording exactly which unfolded prefix-and-period windows may be read by downstream consumers.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reached from I, B, L, Q, D, S before any real-facing seal is visible.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the prefix, period, recurrence, rational normalization, denominator comparison, StreamName window, and RegSeqRat rows have been displayed.*

9. *H is the componentwise* hsame *transport row for the prefix, period block, recurrence ledger, rational normalization, denominator comparisons, StreamName windows, RegSeqRat handoff, Real seal, replay, provenance, and local naming rows.*

10. *C is the displayed* Cont *replay row from a finite periodic-window request through the recurrence ledger, rational normalization, denominator comparison, regular handoff, and terminal seal.*

11. *P is the* Pkg *provenance over* $\mathsf{PeriodicContinuedFraction}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{ContinuedFractionConvergents}^{\uparrow}$, $\mathsf{DirichletApproximation}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}$ *row.*

The carrier predicate accepts exactly these finite prefix-and-period packets. It has no coordinate for a host quadratic irrational, a chosen infinite continued fraction, quotient real equality, trichotomy, global irrationality measure, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.PeriodicContinuedFractionUp.PeriodicContinuedFractionTasteGate single carrier alignment`

Theorem 1961.2 (PeriodicContinuedFraction NameCert obligations)**.** *For every accepted* PeriodicContinuedFractio *carrier $F = (I, B, L, Q, D, S, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed prefix, nonempty period block, recurrence ledger, rational normalization, denominator comparison, StreamName window, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming rows;*

2. *periodic-block stability, requiring every unfolded window in S to read the same nonempty row B with its displayed phase coordinate;*

3. *recurrence exactness, requiring every convergent in L and Q to be obtained from the displayed prefix and period rows by the finite numerator and denominator recurrence;*

4. *convergent-window compatibility, requiring the rational normalization and denominator-bound comparisons in Q, D to address only the windows exposed by S;*

5. *Real-seal nonescape, requiring E to be downstream of G and to expose no host quadratic irrational, selected infinite expansion, quotient real equality, trichotomy branch, or ambient completeness theorem;*

6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the finite route recorded by I, B, L, Q, D, S, G, E.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1961.1. The accepted packet displays exactly the prefix, period block, recurrence ledger, rational normalization, denominator comparison, StreamName windows, regular handoff, terminal Real seal, and structural rows.

The periodic-block and recurrence obligations are finite. The row B is nonempty and supplies the only repeated block, while L unfolds that row after the prefix I by the displayed numerator and denominator recurrence.

The rational approximation obligations then read only displayed windows. The row Q normalizes the convergents from L, the row D compares their denominators against the finite budgets, and the row S records the finite StreamName windows to which those comparisons apply.

The handoff and seal are terminal. The row G is downstream of I, B, L, Q, D, S, and the row E is downstream of G, so a Real-facing consumer cannot bypass the regular rational route.

Finally, H, C, P, N preserve the same packet componentwise. Since the carrier contains no further coordinate, the six listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}$ surface. □

Theorem 1961.3 (PeriodicContinuedFraction quadratic boundary)**.** *An accepted* $\mathsf{PeriodicContinuedFraction}^{\uparrow}$ *carrier exports only the finite periodic-block-to-convergent boundary*

$$I, B \longmapsto L, Q, D, S \longmapsto G \longmapsto E.$$

Consequently the packet may be consumed as a finite rational-approximation source for $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, but it does not export a host quadratic irrational, selected infinite expansion, quotient real equality, trichotomy, irrationality-measure bound, or* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start from an accepted carrier $F = (I, B, L, Q, D, S, G, E, H, C, P, N)$. The only periodic data are the finite prefix row I and the nonempty block row B.

The rows L, Q, D, S turn this finite source into convergent and comparison data. They record recurrence, normalization, denominator bounds, and finite windows, but they do not add an ambient completed expansion.

The rows G and E provide the only regular rational and Real-facing exits. Their order forces every such exit through the finite convergent boundary before the terminal seal is visible.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no row stores a host quadratic irrational, quotient equality, trichotomy branch, irrationality-measure theorem, or completeness witness, none is exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PeriodicContinuedFraction}^{\uparrow}$ packet is generated from a finite partial-quotient prefix, a nonempty period block, recurrence and convergent rows, rational normalization, denominator-bound comparison, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1961.1, Definition 1961.2, and Definition 1961.3 seed the finite periodic-block route from continued fractions to rational convergent windows, RegSeqRat readback, and a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PeriodicContinuedFraction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PeriodicContinuedFractionUp.PeriodicContinuedFractionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim a host quadratic irrational, selected infinite expansion, quotient real equality, trichotomy, irrationality-measure theorem, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, periodic-block stability, recurrence exactness, convergent-window compatibility, Real-seal nonescape, provenance exhaustion, and transport stability for BEDC.Derived.PeriodicContinuedFra

Chapter 1962

A Concrete Naming Certificate for $\mathsf{FanSpace}^{\uparrow}$

Following Brouwer, Troelstra, and Bishop-style constructive analysis, the $\mathsf{FanSpace}^{\uparrow}$ packet records a finite-branching Boolean prefix tree by the finite information exposed to a proof. It sits between the finite Boolean prefix surface of chapter 1475, the unfolding-prefix discipline of chapter 1482, and the L10 finite-request gateway of chapter 126. The packet names only finite tree nodes, Boolean observations, prefix-spine rows, transport, replay, provenance, and a local naming certificate; it does not import a host function space, a completed stream, an open-cover compactness theorem, or a countable-choice branch.

Definition 1962.1 (FanSpace carrier)**.** *A* $\mathsf{FanSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, B, L, C, Q, H, R, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite node requests made by the current consumer;*
2. *T is the finite binary-branch prefix-tree row, closed under displayed predecessors and carrying at most the two Boolean successors named at each node;*
3. *B is the* $\mathsf{Bool}^{\uparrow}$ *observation ledger attached to the requested branches of T;*
4. *L is the* $\mathsf{List}^{\uparrow}$ *spine row ordering the exposed prefix nodes and their branch labels as finite data;*
5. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *boundary row used when a maximal displayed prefix is read as a Boolean window;*
6. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *handoff row used when the same finite prefix is consumed as an unfolding choice-sequence prefix;*
7. *H is the componentwise* hsame *transport row for S, T, B, L, C, Q;*
8. *R is the displayed* Cont *replay row for already accepted prefix-tree restrictions and boundary reads;*
9. *P is the* Pkg *provenance over* $\mathsf{FanSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$ *row.*

The carrier admits only the finite branching data explicitly displayed in T. It exports no ambient space of maps $\mathsf{Nat}^{\uparrow} \to \mathsf{Bool}^{\uparrow}$*, no completed Boolean stream, no hidden infinite branch, no open-cover compactness witness, and no countable-choice selector.*

Lean4 ✓ `BEDC.Derived.FanSpaceUp.TasteGate.FanSpaceTasteGate single carrier alignment`

Theorem 1962.2 (FanSpace NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$ *surface of Definition 1962.1 consists of the following finite obligations:*

1. *carrier admission for the displayed packet* $F = (S, T, B, L, C, Q, H, R, P, N)$*;*
2. *finite branching stability, requiring every accepted node of T to carry only the displayed zero-successor and one-successor rows admitted by* $\mathsf{Bool}^{\uparrow}$*;*
3. *prefix restriction, requiring every displayed subtree below a requested node to repack through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{List}^{\uparrow}$ *spine,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming rows;*
4. $\mathsf{CantorSpace}^{\uparrow}$ *boundary compatibility for every displayed Boolean prefix window;*
5. $\mathsf{ChoiceSequence}^{\uparrow}$ *handoff compatibility for every displayed unfolding prefix consumed as choice-sequence data;*
6. *transport and replay stability for componentwise* hsame *moves and already accepted* Cont *reads;*
7. *non-escape from host function spaces, completed streams, open-cover compactness principles, and countable-choice branches.*

Proof. Carrier admission is projection of the ten displayed rows in Definition 1962.1. The schedule, tree, Boolean ledger, spine, boundary, handoff, transport, replay, provenance, and local naming rows are finite BHist data.

Finite branching stability is read from the tree row T. Each successor row is labelled by a displayed $\mathsf{Bool}^{\uparrow}$ branch observation, so a node has only the zero-branch and one-branch rows already present in the packet.

For prefix restriction, choose a displayed subtree below a requested node of T. Its nodes and branch labels are subrows of T, B, L, and the same S, H, R, P, N rows transport, replay, name, and package the restricted packet.

The $\mathsf{CantorSpace}^{\uparrow}$ boundary row C handles reads that treat a displayed maximal prefix as a Boolean window, while the $\mathsf{ChoiceSequence}^{\uparrow}$ handoff row Q handles reads that consume the same prefix as unfolding sequence data.

Finally inspect the carrier coordinates. The packet contains no row for a host function space, completed stream, hidden branch, open-cover compactness witness, or countable-choice selector, so no local $\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$ consumer can export one. □

Theorem 1962.3 (FanSpace StreamName-Cantor scope)**.** *Every public read of an accepted* $\mathsf{FanSpace}^{\uparrow}$ *carrier factors through a finite* $\mathsf{StreamName}^{\uparrow}$ *request and a* $\mathsf{CantorSpace}^{\uparrow}$ *Boolean prefix window before any downstream L10 consumer may inspect the packet. The visible route is*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow},\ \mathsf{ChoiceSequence}^{\uparrow},\ \mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}.$$

The route binds finite node requests, Boolean branch observations, prefix-spine ordering, Cantor boundary reads, choice-sequence handoffs, transport, replay, provenance, and local naming. It does not claim a completed fan, a chosen branch, host-function extensionality, open-cover compactness, or countable choice.

Proof. Begin with a public read of $F = (S, T, B, L, C, Q, H, R, P, N)$. The consumer first sees the finite request schedule S and the finite prefix-tree row T; without these rows there is no named node at which a branch value can be read.

The branch observations are then read through the $\mathsf{Bool}^{\uparrow}$ ledger B and the ordered $\mathsf{List}^{\uparrow}$ spine L. Any maximal displayed prefix requested by the consumer reaches the $\mathsf{CantorSpace}^{\uparrow}$ boundary row C before it can be passed to a downstream Boolean-window consumer.

If the same prefix is consumed as unfolding sequence data, the read passes through the $\mathsf{ChoiceSequence}^{\uparrow}$ handoff row Q after the StreamName and Cantor-visible rows have been fixed. The componentwise hsame row H, Cont replay row R, Pkg provenance row P, and local naming row N then route only the displayed finite data.

Thus every public read factors through the announced finite route before any L10 consumer sees the packet. The excluded infinite objects are not carrier coordinates, so the route cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{FanSpace}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ request schedule, a displayed binary prefix-tree row, $\mathsf{Bool}^{\uparrow}$ branch observations, a $\mathsf{List}^{\uparrow}$ spine, $\mathsf{CantorSpace}^{\uparrow}$ boundary reads, $\mathsf{ChoiceSequence}^{\uparrow}$ handoffs, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$, seed)

Scope closed: Definition 1962.1, Definition 1962.2, and Definition 1962.3 seed the finite-branching Boolean prefix-tree surface over $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{Bool}^{\uparrow}$ branch rows, $\mathsf{List}^{\uparrow}$ spines, $\mathsf{CantorSpace}^{\uparrow}$ boundary reads, $\mathsf{ChoiceSequence}^{\uparrow}$ handoffs, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close host function spaces, completed Boolean streams, open-cover compactness, fan-theorem compactness, hidden infinite branches, or countable choice.

Upgrade path: The next step is obligationClosure: carrier admission, finite branching stability, prefix restriction, CantorSpace boundary compatibility, ChoiceSequence handoff compatibility, transport, replay, provenance, and non-escape should be formalized under BEDC.Derived.FanSpaceUp.

Chapter 1963

A Concrete Naming Certificate for $\mathsf{Paracompact}^{\uparrow}$

Following Kelley and Bourbaki on paracompact spaces and locally finite refinements, the $\mathsf{Paracompact}^{\uparrow}$ packet records the finite BEDC surface by which a displayed open-cover request receives a locally finite open refinement ledger. It sits above the open-row discipline of chapter 88, beside the separation boundary of chapter 1866, and below later partition-of-unity and manifold-gluing consumers. It also records the compatibility route through **??** and, when the source is metric or compact-metric, the finite-radius handoff through chapter 57 and the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ surface. The packet names only a displayed refinement carrier; it does not select subcovers from arbitrary open families or close partition-of-unity data.

Definition 1963.1 (Paracompact carrier)**.** *A* $\mathsf{Paracompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, A, R, L, N, M, H, C, P, Z)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the ambient open-set operations and the common observation surface on which the cover and refinement rows are read.*
2. *A is the open-cover request row, a finite request ledger whose cells name the displayed open rows to be refined rather than an arbitrary host-indexed cover.*
3. *R is the locally finite open-refinement ledger, recording for each displayed source cell a finite neighborhood row meeting only finitely many refinement cells and recording the refinement map back into A.*
4. *L is the coverage ledger, recording that the refinement cells cover the same displayed source region that A covers, with the containment read carried by the topology row T.*
5. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *compatibility row, requiring the refinement ledger to preserve the closed-pair separation route of chapter 1866 whenever the cover request is used with normal-space separation data.*
6. *M is the metric-or-finite-radius handoff row, read either from a* $\mathsf{Met}^{\uparrow}$ *source or from a* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive finite-radius ledger when such data are available; the handoff exports only the displayed finite-radius route.*

7. *H records componentwise* hsame *transport for the topology source, cover request, locally finite refinement ledger, coverage ledger, normal-space compatibility row, metric-or-finite-radius handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a paracompact-refinement request through $T, A, R, L, N, M, H, P, Z$.*
9. *P is the* Pkg *provenance over* $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Z is the local* $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, R, L, N, M, H, C, P, Z$. It has no coordinate for selected subcovers, arbitrary open-cover compactness, partition-of-unity weights, host complement selection, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactTasteGate single carrier alignment`

Theorem 1963.2 (Paracompact NameCert obligations)**.** *For every accepted* $\mathsf{Paracompact}^{\uparrow}$ *carrier*

$$Q = (T, A, R, L, N, M, H, C, P, Z),$$

the local $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, open-cover request, locally finite refinement ledger, coverage ledger, normal-space compatibility row, metric-or-finite-radius handoff, transport, replay, provenance, and local naming;*
2. *open-cover exposure, requiring every refinement request to display A over the topology source T before any refinement cell can be consumed;*
3. *locally finite refinement exactness, requiring R to display both the refinement cells and the finite neighborhood ledger witnessing local finiteness;*
4. *coverage and containment exactness, requiring L to record that each displayed source cell covered by A is covered by the refinement ledger and that every refinement cell is contained in its assigned cover cell;*
5. *normal-space compatibility, requiring every closed-pair separation consumer to pass through the* $\mathsf{NormalSpace}^{\uparrow}$ *row N and the Urysohn-facing route of* **??** *before it reads the refinement as separation-compatible;*
6. *metric-or-finite-radius handoff, requiring a metric consumer to read M through the displayed* $\mathsf{Met}^{\uparrow}$ *source or through the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *finite-radius ledger, with no selected subcover or host compactness coordinate;*
7. hsame, Cont, *and* Pkg *transport, requiring componentwise transport, displayed replay, and provenance to preserve exactly the listed rows and no hidden host-refinement data.*

No local paracompact-refinement obligation is stored outside these seven rows.

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 1963.1. The accepted tuple displays exactly the topology source, cover request, locally finite refinement ledger, coverage ledger, normal-space compatibility row, metric-or-finite-radius handoff, and structural transport, replay, provenance, and naming rows.

Open-cover exposure follows from the order of the carrier. A refinement request first reads T and then A over that same topological source; no refinement cell is available before the cover request has been displayed.

Locally finite refinement exactness is the projection to R. The row records the refinement cells, the assignment back into A, and the finite neighborhood ledger witnessing that each displayed source cell meets only finitely many refinement cells.

Coverage and containment exactness is the projection to L. It records both directions needed by the local request: coverage of the displayed source region and containment of every refinement cell in its assigned cover cell.

Normal-space compatibility is read through N and the cited Urysohn-facing route. The packet can be consumed together with normal-space separation data only after the closed-pair route of chapter 1866 and the separation-function route of **??** have been displayed.

Metric-or-finite-radius handoff is read through M. When metric or compact-metric data are present, M exposes the $\mathsf{Met}^{\uparrow}$ or $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ finite-radius route; otherwise it exports no compactness or subcover-selection principle.

Finally, transport and provenance are exhausted by H, C, P, Z. These rows move and replay exactly the displayed carrier coordinates, so no local obligation exports selected subcovers, arbitrary open-cover compactness, partition-of-unity weights, host complement selection, Lean verification, or bridge checking. □

Theorem 1963.3 (Paracompact refinement consumer route)**.** *Every refinement read exported by an accepted* $\mathsf{Paracompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{Paracompact}^{\uparrow} \longrightarrow \mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{UrysohnLemma}^{\uparrow}$$

when consumed by a separation-compatible topology route, and through the displayed $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *handoff when a metric finite-radius route is available. No exported refinement read factors through a selected subcover, arbitrary open-cover compactness, partition-of-unity weights, host complement selection, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactCarrier refinement consumer route`

Proof. Begin with an accepted carrier $Q = (T, A, R, L, N, M, H, C, P, Z)$. Carrier admission exposes the $\mathsf{Topology}^{\uparrow}$ source and the displayed open-cover request before any refinement ledger can be inspected.

The refinement gate is R and L. These rows provide the locally finite refinement cells, their assignment to the cover request, the finite neighborhood ledger, and the coverage and containment reads.

The separation-compatible gate is N. A consumer that also needs normal-space separation must read the displayed $\mathsf{NormalSpace}^{\uparrow}$ compatibility route and the sibling $\mathsf{UrysohnLemma}^{\uparrow}$ route before treating the refinement as compatible with separation-function data.

The metric gate is M. A metric or compact-metric consumer may read the displayed $\mathsf{Met}^{\uparrow}$ or $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ handoff, but the row carries only finite-radius data already present in the packet.

The replay and local naming rows H, C, P, Z preserve this order. Since the classifier has no further coordinate, no exported refinement read can obtain selected subcovers, arbitrary open-cover compactness, partition-of-unity weights, host complement selection, Lean verification, or bridge checking. □

Definition 1963.4 (Paracompact locally finite refinement carrier)**.** *A locally finite refinement carrier for* $\mathsf{Paracompact}^{\uparrow}$ *is a finite* BHist *packet*

$$Q_{\mathrm{lf}} = (T, G, A, R, F, L, N, U, M, H, C, P, Z)$$

whose rows refine Definition 1963.1 as follows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row and G is the observation window on which cover and neighbourhood reads are made.*
2. *A is the displayed open-cover request over T.*
3. *R is the refinement-family row, listing the refinement cells and their assignment back into cells of A.*
4. *F is the local-finiteness ledger: for each displayed point or source cell in G, it records a neighbourhood row and the finite list of refinement cells that meet that neighbourhood.*
5. *L is the coverage and containment ledger, recording that the refinement covers the same displayed source region and that every refinement cell is contained in its assigned cover cell.*
6. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *separation-compatibility row and U is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *handoff row used when a closed-pair separation consumer reads the refinement.*
7. *M is the metric finite-radius handoff, read through* $\mathsf{Met}^{\uparrow}$ *and, when available, through the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger.*
8. *H, C, P, Z are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$ *rows.*

The carrier has no coordinate for a selected subcover, an arbitrary host-indexed refinement family, partition-of-unity weights, a host complement selector, open-cover compactness, or an ambient refinement theorem.

Theorem 1963.5 (Paracompact refinement NameCert obligations)**.** *For a locally finite refinement carrier*

$$Q_{\mathrm{lf}} = (T, G, A, R, F, L, N, U, M, H, C, P, Z),$$

the local $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$ *surface is exhausted by carrier admission, cover-refinement rows, the locally finite ledger, normal-space separation compatibility, and transport/replay/provenance/local naming rows. Concretely, every refinement consumer must read:*

1. *T, G, A before a cover request is available;*
2. *R, L before a refinement cell can be used as covering and contained in its assigned cover cell;*
3. *F before the refinement can be read as locally finite on the displayed observation window;*
4. *N, U before the refinement can be consumed by a normal-space or Urysohn-facing separation route;*
5. *M before a metric finite-radius route is exported; and*
6. *H, C, P, Z before the transported packet is replayed or named.*

No NameCert obligation for locally finite paracompact refinement is stored outside these displayed rows.

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactCarrier refinement namecert obligations`

Proof. Carrier admission is projection from Definition 1963.4. The tuple displays the topology source, observation window, cover request, refinement family, local-finiteness ledger, coverage and containment ledger, normal-space and Urysohn rows, metric handoff, transport, replay, provenance, and local name.

The cover-refinement obligations are read from A, R, L. The cover request A is tied to the same topological source T, the refinement row R assigns each refinement cell back to the cover request, and the ledger L records coverage and containment.

Local finiteness is read only from F. For each displayed point or source cell in G, the row gives a neighbourhood together with the finite list of refinement cells meeting it, so a consumer cannot infer local finiteness from the existence of the refinement family alone.

Separation compatibility is read through N, U. These rows force a normal-space consumer to pass through the displayed $\mathsf{NormalSpace}^{\uparrow}$ and $\mathsf{UrysohnLemma}^{\uparrow}$ surfaces before treating the refinement as separation-compatible.

Finally, M, H, C, P, Z exhaust metric export, transport, replay, provenance, and local naming. Since the carrier has no further coordinate, selected subcovers, host-indexed refinements, partition-of-unity weights, host complement selection, open-cover compactness, and ambient refinement theorems are not part of the local certificate. □

Theorem 1963.6 (Paracompact metric-normal route)**.** *Every metric* $\mathsf{Paracompact}^{\uparrow}$ *refinement read exported from a locally finite refinement carrier first factors through the* $\mathsf{Met}^{\uparrow}$ *finite-radius source and the* $\mathsf{NormalSpace}^{\uparrow}$ *separation row before it reaches the refinement export. In symbols, the visible route is*

$$\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{FiniteLebesgueNumber}^{\uparrow} \longrightarrow \mathsf{Topology}^{\uparrow} \longrightarrow \mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{Paracompact}^{\uparrow}$$

whenever a finite-radius ledger is present, and the normal-space row remains mandatory for every separation-compatible export.

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactCarrier metric normal route`

Proof. Start with $Q_{\mathrm{lf}} = (T, G, A, R, F, L, N, U, M, H, C, P, Z)$. The metric information available to the packet is exactly the handoff row M, so a metric consumer reads the displayed $\mathsf{Met}^{\uparrow}$ source and any available $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ finite-radius ledger before it can use a refinement export.

The refinement export itself is still topological. The rows T, A, R, F, L expose the open-cover request, the refinement family, the local-finiteness ledger, and the coverage and containment ledger over the displayed topology.

For separation-compatible use, the route must pass through N and then U. Thus the normal-space separation row and the Urysohn-facing handoff are read before the refinement is exported to any consumer that treats it as compatible with closed-pair separation data.

The replay and naming rows H, C, P, Z preserve this order componentwise. With no coordinate for selected subcovers, partition-of-unity weights, host complement selection, open-cover compactness, or ambient normality, every metric paracompact read passes through the displayed metric and normal-space rows before refinement export. □

Theorem 1963.7 (Paracompact obligation closure package)**.** *Every accepted* $\mathsf{Paracompact}^{\uparrow}$ *packet has its obligation-closure surface given exactly by Definition 1963.1, Definition 1963.2, Definition 1963.3, Definition 1963.4, Definition 1963.5, and Definition 1963.6. The package exposes displayed cover requests, locally finite refinement ledgers, coverage and containment rows, normal-space and Urysohn-facing separation routes, metric finite-radius handoffs, componentwise transport, replay, provenance, and local naming. It exports no selected subcovers, arbitrary open-cover compactness, partition-of-unity closure, or host complement selection.*

Proof. Project the displayed carrier of Definition 1963.1. Its rows expose the topology source, open-cover request, locally finite refinement ledger, coverage ledger, normal-space compatibility row, metric-or-finite-radius handoff, transport, replay, provenance, and local naming.

Apply Definition 1963.2 and Definition 1963.3. These obligations force every refinement consumer to read the cover request, refinement exactness, coverage and containment rows, separation-compatible handoff, and metric finite-radius route in the displayed order.

For the locally finite surface, project Definition 1963.4 and apply Definition 1963.5. The local-finiteness ledger is a named row, and it is transported only with the same topology, cover, refinement, separation, metric, replay, provenance, and naming rows.

Finally apply Definition 1963.6. The metric-normal consumer route remains downstream of the displayed metric finite-radius source and the normal-space separation row, so the package has no coordinate from which selected subcovers, arbitrary open-cover compactness, partition-of-unity closure, or host complement selection could be read. □

Lean4 ✓ `BEDC.Derived.ParacompactUp.ParacompactCarrier obligation closure package`

Theorem 1963.8 (Paracompact scoped refinement export)**.** *Every accepted* $\mathsf{Paracompact}^{\uparrow}$ *refinement consumer factors through the displayed topology source, cover request, locally finite refinement ledger, coverage and containment rows,* $\mathsf{NormalSpace}^{\uparrow}$ *and* $\mathsf{UrysohnLemma}^{\uparrow}$ *separation route,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$ *rows named by Definition 1963.7. The exported scope is exactly this public refinement route, and it has no coordinate for selected subcovers, arbitrary open-cover compactness, partition-of-unity weights, host complement selection, Lean verification, or bridge checking.*

Proof. Start with the accepted carrier and the obligation surface of Definition 1963.2. These rows expose the topology source, displayed cover request, locally finite refinement ledger, coverage ledger, separation-compatibility row, metric handoff, transport, replay, provenance, and local naming before any consumer can inspect a refinement.

Apply Definition 1963.3. It orders every refinement read through the displayed topology, the refinement ledger, and the separation or finite-radius handoff appropriate to the consumer.

For the locally finite portion, use Definition 1963.4 and Definition 1963.5. The rows $T, G, A, R, F, L, N, U, M, H$, make the local-finiteness ledger, coverage and containment ledger, normal-space route, Urysohn handoff, metric handoff, and structural rows explicit.

Finally compose Definition 1963.6 with Definition 1963.7. The metric-normal route fixes the downstream consumer order, and the closure package exhausts the visible rows, so the scoped export cannot read any selected subcover, arbitrary open-cover compactness, partition-of-unity weight, host complement selector, Lean verification row, or bridge-checking row. □

Closure status for $\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 1963.1, Definition 1963.2, Definition 1963.3, Definition 1963.4, Definition 1963.5, Definition 1963.6, Definition 1963.7, and Definition 1963.8 close the finite public refinement route from displayed open-cover requests through locally finite refinement ledgers into normal-space and metric consumer handoffs.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Paracompact}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ParacompactUp.ParacompactCarrier\_refinement\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close selected subcovers, arbitrary open-cover compactness, partition-of-unity closure, host complement selection, Lean verification, or bridge checking.

Upgrade path: No paper-side theory upgrade remains for the displayed refinement route; Lean verification and bridge status remain recorded on their own axis.

Constructive story: A Paracompact$^{\uparrow}$ packet is generated from a Topology$^{\uparrow}$ source row, an open-cover request row, a locally finite open-refinement ledger, a coverage ledger, a NormalSpace$^{\uparrow}$ compatibility row, a Met$^{\uparrow}$ or FiniteLebesgueNumber$^{\uparrow}$ handoff row when available, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1964

A Concrete Naming Certificate for $\mathsf{RegularSequenceLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularSequenceLimit}^{\uparrow}$ packet records the finite relation by which a regular rational name is admitted as the displayed limit route for a $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the limit-seal boundary of chapter 386, the terminal real-name seal of chapter 38, and the explicit Cauchy convergence route of chapter 1509. The packet names the RegSeqRat-to-Real limit relation as certificate data; it is not ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, a selected host limit, trichotomy, open-cover compactness, countable choice, Lean verification, or bridge checking.

Definition 1964.1 (RegularSequenceLimit carrier)**.** *A* $\mathsf{RegularSequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposing exactly the finite observations of the rational sequence name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the windows in S.*
3. *Q is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row that records the displayed limit-seal compatibility of the same S, R route.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row obtained only after the regular rational route has passed through Q.*
5. *H records componentwise* hsame *transport for the stream schedule, regular readback, limit-seal row, Real seal, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a finite precision request through S, R, Q, E and then through the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceLimit}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, R, Q, E, H, C, P, N. It contains no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, trichotomy,*

open-cover compactness, countable choice, Lean verification, bridge checking, or any detached convergence relation.

Lean4 ✓ `BEDC.Derived.RegularSequenceLimitUp.RegularSequenceLimitTasteGate single carrier alignment`

Theorem 1964.2 (RegularSequenceLimit NameCert obligations)**.** *For every accepted* $\mathsf{RegularSequenceLimit}^{\uparrow}$ *carrier*

$$L = (S, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularSequenceLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed stream schedule, regular rational readback, limit-seal row, Real seal, transport, continuation, provenance, and local name;*
2. *regular-name routing, requiring every limit read to pass through S and R before Q or E is available;*
3. *seal compatibility, requiring Q to consume the same finite window and regular-readback route that supplies the* $\mathsf{Real}^{\uparrow}$ *seal E;*
4. *continuation exactness, requiring each finite precision request to replay through C without adding a second window family, hidden modulus, quotient stream, selected host limit, or ambient completeness coordinate;*
5. *structural stability, requiring H, P, N to transport, source, and name only the displayed rows.*

No RegularSequenceLimit obligation is stored outside these rows.

Proof. Project the accepted packet $L = (S, R, Q, E, H, C, P, N)$. The carrier admission row exposes exactly one finite stream schedule, one regular rational readback row, one limit-seal compatibility row, one Real seal, and the transport, replay, provenance, and naming rows.

The routing and seal-compatibility clauses follow from the displayed order. The row R is the only regular rational readback attached to S, while Q is the only limit-seal row attached to that readback. Hence a limit consumer cannot inspect Q or E without first reading the same S, R route.

Continuation exactness is the row C. Each finite precision request replays the visible route and has no coordinate from which to introduce a second window family, a hidden modulus, quotient stream equality, a selected host limit, or ambient completeness.

Finally, H, P, N transport, source, and name only the displayed packet. Since the carrier has no further coordinate, the five listed rows exhaust the local NameCert surface. □

Theorem 1964.3 (RegularSequenceLimit window-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularSequenceLimit}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto R \longmapsto Q \longmapsto E$$

and then through the stored hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The route is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *regular-name handoff read through a* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary; it exports no ambient completeness theorem, quotient stream, selected host limit, trichotomy row, open-cover compactness row, countable choice principle, Lean verification, or bridge check.*

Proof. Begin with a Real-facing read. By carrier admission, the only observation source is the finite $\mathsf{StreamName}^{\uparrow}$ schedule S, and the only regular rational name exposed from that source is R.

The limit-seal row is downstream. The row Q consumes the same S, R route, so the read reaches the limit boundary only after the regular rational handoff has been displayed.

The terminal Real row is E. The structural rows H, C, P, N then transport, replay, source, and name the same packet. No other carrier coordinate can provide a host completeness theorem, quotient stream, selected host limit, trichotomy row, open-cover compactness row, countable choice principle, Lean verification, or bridge check. □

Theorem 1964.4 (RegularSequenceLimit classifier determinacy)**.** *Let*

$$L = (S, R, Q, E, H, C, P, N)$$

and

$$L' = (S', R', Q', E', H', C', P', N')$$

be accepted $\mathsf{RegularSequenceLimit}^{\uparrow}$ *carriers. If their displayed stream schedules, regular readbacks, limit-seal rows, Real seals, transport rows, continuation rows, provenance rows, and local name rows are componentwise* hsame*, then the local classifier identifies* L *and* L'*. Conversely, any local classifier identification of accepted carriers is witnessed by componentwise* hsame *comparison of exactly those eight displayed rows.*

Proof. For the forward direction, read the classifier as the local $\mathsf{NameCert}_{\mathsf{RegularSequenceLimit}^{\uparrow}}$ comparison. Its carrier inventory is S, R, Q, E, H, C, P, N, so componentwise hsame on those rows supplies every comparison required by the local classifier.

For the reverse direction, project the accepted carrier comparison. The definition of the local classifier has no hidden coordinate beyond the displayed rows, and therefore any identification it accepts must be witnessed on S, R, Q, E, H, C, P, N component by component.

The two projections use the same finite row list. Thus classifier identity is determined exactly by componentwise hsame comparison of the displayed stream schedule, regular readback, limit-seal row, Real seal, transport, continuation, provenance, and local name. □

Theorem 1964.5 (RegularSequenceLimit Real nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{RegularSequenceLimit}^{\uparrow}$ *carrier receives the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility row have been replayed through* H, C, P, N*. Consequently the packet cannot export ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a selected host limit, trichotomy, open-cover compactness, countable choice, Lean verification, or bridge checking.*

Proof. Take a Real-facing consumer of the accepted carrier. The carrier definition places E after the displayed S, R, Q route, so the consumer has no access to a Real seal before the finite windows, regular readback, and limit-seal boundary have been read.

The structural rows do not add mathematical content outside that route. Transport H preserves the same displayed coordinates, continuation C replays them for finite precision requests, provenance P records their source, and the local name row N names the packet.

Thus every Real-facing read remains inside the displayed route. Since the carrier contains no excluded coordinate, the listed nonescape claims follow. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularSequenceLimit}^{\uparrow}$ packet is generated from a displayed $\mathsf{StreamName}^{\uparrow}$ window schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback row, a $\mathsf{CauchyLimitSeal}^{\uparrow}$ compatibility row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1964.1, Definition 1964.2, Definition 1964.3, Definition 1964.4, and Definition 1964.5 seed the finite RegSeqRat-to-Real limit relation over StreamName windows, CauchyLimitSeal compatibility, and the Real seal route.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularSequenceLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient Real$^{\uparrow}$ completeness, quotient stream equality, selected host limits, trichotomy, open-cover compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-name routing, seal compatibility, continuation exactness, classifier determinacy, Real nonescape, and bridge coverage for BEDC.Derived.RegularSequenceLimitUp.

Chapter 1965

A Concrete Naming Certificate for $\mathsf{RealConvergenceRate}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Weihrauch-style computable analysis, the $\mathsf{RealConvergenceRate}^{\uparrow}$ packet records an explicit rate-of-convergence object for Real and Completion consumers. It sits after the dyadic rate discipline of chapter 1953, the modulus boundary of chapter 1423, the finite stream observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes a finite rate schedule before any $\mathsf{Real}^{\uparrow}$ seal is consumed, so downstream Real and Completion phases can request precision through displayed data rather than through quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Definition 1965.1 (Real convergence rate carrier)**.** *A* $\mathsf{RealConvergenceRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the rate schedule row, listing the finite precision requests accepted by the packet and the displayed bound that each request must consult before any real-name seal can be read.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row selected by the schedule R.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, slack comparisons, and budget cells attached to the scheduled windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and the tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after R, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the rate schedule, finite windows, tolerance ledger, readback row, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of each accepted precision request through*

$$R \longmapsto S \longmapsto D \longmapsto Q \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RealConvergenceRate}^{\uparrow}$*,* $\mathsf{RapidCauchyName}^{\uparrow}$*,* $\mathsf{RealCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RealConvergenceRate}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, S, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.RealConvergenceRateUp.RealConvergenceRateTasteGate single carrier alignment`

Theorem 1965.2 (Real convergence rate tail control)**.** *For every accepted* $\mathsf{RealConvergenceRate}^{\uparrow}$ *carrier*

$$K = (R, S, D, Q, E, H, C, P, N),$$

every accepted precision request factors through the rate schedule row R before any $\mathsf{Real}^{\uparrow}$ *seal row E is consumed. Consequently the tail read seen by a Real or Completion consumer is routed through the finite path*

$$R \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

together with componentwise transport, displayed replay, provenance, and local naming.

Proof. Project the accepted packet K. The carrier exposes the rate schedule R as the first analytic row and places the terminal seal E only after the scheduled windows, dyadic ledger, and regular readback.

Induct over the finite list of precision requests displayed by R. The empty schedule has no accepted request and therefore no seal-facing tail read.

For the cons step, the head request reads its bound from R, selects its finite window in S, and consumes the corresponding dyadic radius and slack comparison in D. The regular readback Q is then the only row from which a seal-facing consumer can proceed to E.

The induction hypothesis handles the remaining scheduled requests. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so there is no carrier coordinate from which a consumer can bypass R and read the Real seal directly. □

Theorem 1965.3 (Real convergence rate NameCert obligations)**.** *For every accepted* $\mathsf{RealConvergenceRate}^{\uparrow}$ *carrier*

$$K = (R, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealConvergenceRate}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the rate schedule R, finite* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N;*
2. *classifier stability under componentwise* hsame *transport of the displayed rate schedule, window, ledger, readback, seal, replay, provenance, and local naming rows;*
3. *rate-to-tail control, requiring every accepted precision request to consult R before any tail read reaches S, D, Q or the Real seal E;*
4. *dyadic tolerance accounting, requiring D to store exactly the radii, slack comparisons, and budget cells consumed by the scheduled windows and readback row;*
5. *ledger provenance, requiring P to name the displayed Real and Completion dependencies through* $\mathsf{RapidCauchyName}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
6. *non-escape from quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard-bridge data.*

No real-convergence-rate obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 1965.1. The accepted packet lists exactly the rate schedule, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal seal, and four structural rows.

Classifier stability is componentwise. The hsame row H transports each displayed coordinate of the packet, while the classifier has no extra coordinate that could change independently of those rows.

Rate-to-tail control is Definition 1965.2. It forces every accepted precision request through R before any scheduled window, tolerance cell, readback row, or terminal seal can be consumed.

The tolerance and provenance clauses are then row inspections. The ledger D contains the dyadic accounting consumed by S and Q, and P names only the listed Real and Completion dependencies.

Finally, the carrier has no coordinate for quotient equality, a selected limit, countable choice, host equality, trichotomy, ambient completeness, Lean evidence, or bridge data. The enumerated rows therefore exhaust the local NameCert obligation surface. □

Falsifiable prediction: Every $\mathsf{RealConvergenceRate}^{\uparrow}$ precision request factors through a displayed rate schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: rapidcauchyname, realcauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{RealConvergenceRate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rate admission, finite tail routing, dyadic tolerance accounting, regular readback, terminal Real seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealConvergenceRate}^{\uparrow}}$.

Constructive story: A $\mathsf{RealConvergenceRate}^{\uparrow}$ packet is generated from a rate schedule, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealConvergenceRate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1965.1, Definition 1965.2, and Definition 1965.3 seed the explicit rate-of-convergence route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealConvergenceRate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealConvergenceRateUp.RealConvergenceRateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier stability, rate-to-tail control, dyadic tolerance accounting, ledger provenance, and non-escape for BEDC.Derived.RealConvergenceRateUp.

Chapter 1966

A Concrete Naming Certificate for $\mathsf{BishopRealNormalizer}^{\uparrow}$

Following Bishop and Bridges' constructive real-number presentation, the $\mathsf{BishopRealNormalizer}^{\uparrow}$ packet records the finite boundary where an arbitrary regular Cauchy rational name is normalized before a terminal $\mathsf{Real}^{\uparrow}$ seal may be read. It sits over the dyadic tolerance ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the completion boundary of chapter 1639, the effective comparison packet of chapter 1936, and the modulus-neutral seal discipline of chapter 1955. The packet names a canonical normalization route for L10 Real and Completion consumers; it is not quotient stream equality, host real equality, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1966.1 (Bishop real normalizer carrier)**.** *A* $\mathsf{BishopRealNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, M, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy name row, exposing the rational approximants and finite precision requests before normalization.*
2. *M is the monotone normalized tail-modulus row, recording the displayed tail schedule whose requested indices are nondecreasing and whose tolerance budget refines the source requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack comparisons, and finite error budgets consumed by the source and normalized rows.*
4. *Q is the normalized* $\mathsf{RegSeqRat}^{\uparrow}$ *output row obtained from S through M and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source name, monotone tail modulus, dyadic tolerance ledger, and normalized regular output have been displayed.*
6. *H records componentwise* hsame *transport for S, M, D, Q, E and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the normalization route*

$$S \longmapsto M \longmapsto D \longmapsto Q \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopRealNormalizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{GrzegorczykReal}^{\uparrow}$, $\mathsf{RegularCauchyModulusIndependence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealNormalizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.BishopRealNormalizerUp.BishopRealNormalizerTasteGate single carrier alignment`

Theorem 1966.2 (Bishop real normalizer NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealNormalizer}^{\uparrow}$ *carrier*

$$Z = (S, M, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealNormalizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy name, monotone normalized tail modulus, dyadic tolerance ledger, normalized* $\mathsf{RegSeqRat}^{\uparrow}$ *output, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows H, C, P, N;*

2. *source exposure, requiring every normalization request to begin with the displayed source row S rather than with a detached stream representative;*

3. *monotone tail-modulus normalization, requiring M to present a nondecreasing finite tail schedule whose tolerance budget refines the requests exported by S;*

4. *dyadic accounting, requiring D to store exactly the dyadic radii and slack comparisons used to justify the normalized output row Q;*

5. *regular-output handoff, requiring Q to be the only route by which the normalized finite windows become a* $\mathsf{RegSeqRat}^{\uparrow}$ *real name;*

6. *Real-seal factorization, requiring every downstream consumer of E to pass through S, M, D, Q and the structural rows before reading the terminal seal;*

7. *transport, replay, and provenance non-escape, requiring H, C, P, N to preserve and name only the displayed normalization rows, with no quotient stream equality, host real equality, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Bishop real normalizer obligation is stored outside these rows.

Proof. Project the accepted packet $Z = (S, M, D, Q, E, H, C, P, N)$. Carrier admission exposes exactly the source regular name, monotone tail-modulus row, dyadic tolerance ledger, normalized regular output, terminal Real seal, and the four structural rows.

The source exposure clause is first in the route. A normalization request cannot start from an unnamed stream representative, because the only source coordinate in the carrier is the displayed row S.

The monotone tail-modulus clause is the row M. It records the finite nondecreasing schedule used to choose tails and ties each chosen tail to the source requests that it refines.

Dyadic accounting is then forced by D. The ledger stores the radii and slack comparisons consumed when the selected tails are read as the normalized row Q, so the regular output has no tolerance data outside the packet.

Finally, the Real seal and structural rows exhaust the boundary. The seal E is downstream of S, M, D, Q, while H, C, P, N transport, replay, source, and name only those displayed rows. Hence no excluded quotient, host equality, selected limit, countable-choice object, ambient-completeness principle, Lean evidence, or bridge credential is exported. □

Theorem 1966.3 (Bishop real normalizer Real-seal factorization)**.** *For every accepted* $\mathsf{BishopRealNormalizer}^{\uparrow}$ *carrier*

$$Z = (S, M, D, Q, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* E *factors through the displayed normalization route*

$$S \longmapsto M \longmapsto D \longmapsto Q \longmapsto E$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. In particular, the consumer cannot replace* S *by quotient stream equality, cannot replace* Q *by a selected limit, cannot compare host real values, cannot invoke countable choice, and cannot read* E *as an ambient completeness witness.*

Proof. Begin with a seal-facing consumer. Carrier projection exposes E only as the terminal coordinate of the normalized regular-Cauchy route, not as an independently completed host real.

Trace the route backward from E. The seal is downstream of Q, the normalized $\mathsf{RegSeqRat}^{\uparrow}$ output is downstream of the dyadic tolerance ledger D, and the ledger is controlled by the monotone tail schedule M applied to the source row S.

The structural coordinates preserve the same route. Componentwise hsame in H transports the source, modulus, tolerance, output, and seal rows; displayed Cont in C replays their order; and P, N name the packet without adding a second real-name surface.

Thus every Real-facing use factors through $S, M, D, Q, E, H, C, P, N$. The carrier contains no coordinate from which quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking could be read. □

Falsifiable prediction: Every $\mathsf{BishopRealNormalizer}^{\uparrow}$ Real-facing read factors through a source regular Cauchy row, a monotone normalized tail modulus, a DyadicRatCore tolerance ledger, a normalized RegSeqRat output, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-cauchy-completion, grzegorczyk-real, regular-cauchy-modulus-independence: $\mathsf{BishopRealNormalizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source exposure, monotone tail-modulus normalization, dyadic tolerance accounting, normalized regular output, terminal Real seal factorization, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealNormalizer}^{\uparrow}$ packet is generated from a source regular Cauchy name, a monotone normalized tail modulus, DyadicRatCore tolerance rows, a normalized RegSeqRat output, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealNormalizer}^{\uparrow}}$, seed)

Scope closed: Definition 1966.1, Definition 1966.2, and Definition 1966.3 seed the canonical normalization boundary for regular-Cauchy real names before a Real or Completion consumer reads the terminal seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealNormalizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, monotone tail-modulus normalization, dyadic accounting, regular-output handoff, Real-seal factorization, transport/replay/provenance coverage, and non-escape for BEDC.Derived.BishopRealNormalizerUp.

Chapter 1967

A Concrete Naming Certificate for $\mathsf{EulerPolygon}^{\uparrow}$

Following Euler's polygonal method and Bishop–Bridges constructive analysis, the $\mathsf{EulerPolygon}^{\uparrow}$ packet records the finite approximation route by which local slope data, a dyadic mesh, and displayed polygon vertices are read before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees a terminal seal. It sits on the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the local Lipschitz control packet of chapter 410. The certificate names a finite polygonal approximation object; it is not an ODE solution, not a selected limiting curve, not quotient stream equality, not host real equality, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1967.1 (Euler polygon carrier)**.** *An* $\mathsf{EulerPolygon}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, A, L, V, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the mesh row, listing a finite dyadic time grid, the step widths between adjacent grid points, and the local tolerance budget attached to each step.*
2. *A is the initial point row, recording the displayed initial time and value from which the polygonal path begins.*
3. *L is the local slope and Lipschitz row, carrying the finite slope samples and the local Lipschitz bounds licensed through* $\mathsf{LipschitzMap}^{\uparrow}$ *before any vertex is advanced.*
4. *V is the polygon-vertex* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite sequence of vertices obtained by reading A, M, and L in order.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *step ledger, storing the dyadic products, radius comparisons, slack cells, and accumulated step errors used by the vertex row.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the vertex windows and the dyadic ledger.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after M, A, L, V, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the mesh, initial point, local slope/Lipschitz row, vertex windows, step ledger, readback row, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the finite polygon route*

$$(M, A, L) \longmapsto V \longmapsto D \longmapsto Q \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{EulerPolygon}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerPolygon}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, L, V, D, Q, R, H, C, P, N$. It has no coordinate for an ODE solution, selected limiting curve, global existence theorem, quotient stream equality, host real equality, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.EulerPolygonUp.EulerPolygonTasteGate single carrier alignment`

Theorem 1967.2 (Euler polygon NameCert obligations)**.** *For every accepted* $\mathsf{EulerPolygon}^{\uparrow}$ *carrier*

$$E = (M, A, L, V, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerPolygon}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic mesh, initial point, local slope/Lipschitz row, polygon-vertex* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *step ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and structural rows H, C, P, N;*

2. *mesh exposure, requiring every vertex advance to consult the displayed mesh row M and its dyadic step width before any readback row is consumed;*

3. *initial-point anchoring, requiring the first vertex window in V to be sourced from the displayed initial row A;*

4. *local slope control, requiring each finite step to read its slope sample and Lipschitz bound from L rather than from an external differential equation object;*

5. *dyadic step accounting, requiring D to store exactly the dyadic products, radius comparisons, slack cells, and accumulated error cells consumed by the displayed vertex windows;*

6. *regular rational handoff, requiring Q to be the only route by which the polygon-vertex windows become a* $\mathsf{RegSeqRat}^{\uparrow}$ *name;*

7. *Real-seal nonescape, requiring every consumer of R to pass through M, A, L, V, D, Q and the structural rows, with no ODE solution, selected limit, quotient stream equality, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Euler polygon obligation is stored outside these rows.

Proof. Project the accepted packet $E = (M, A, L, V, D, Q, R, H, C, P, N)$. Carrier admission exposes exactly the mesh, initial point, local slope/Lipschitz row, vertex windows, dyadic ledger, regular readback, terminal Real seal, and the four structural rows.

The mesh and initial-point clauses are forced at the start of the route. A vertex advance has no source coordinate except M for its step width and A for the initial vertex.

Local slope control is the row L. Each displayed step reads its slope sample and Lipschitz bound there, so the packet does not import a separate differential-equation object.

Dyadic accounting is then forced by D. The ledger stores the products, radius comparisons, slack cells, and accumulated error cells consumed by V before any regular rational readback can occur.

Finally, Q, R, H, C, P, N exhaust the handoff boundary. The readback row Q is the only route from vertex windows to the Real seal R, while transport, replay, provenance, and naming preserve only the displayed rows. Thus no excluded solution, selected limit, quotient equality, host equality, choice schedule, ambient completeness principle, Lean evidence, or bridge credential is exported. □

Theorem 1967.3 (Euler polygon step-ledger induction)**.** *For every accepted* $\mathsf{EulerPolygon}^{\uparrow}$ *carrier*

$$E = (M, A, L, V, D, Q, R, H, C, P, N),$$

induction over the finite dyadic mesh row M shows that every displayed prefix of the polygon route has a matching prefix of polygon-vertex windows in V and accumulated dyadic ledger cells in D. The base prefix is the initial row A. At each successor mesh step, the route reads exactly one step width from M, exactly one slope and Lipschitz-control row from L, appends one vertex window to V, and appends the corresponding dyadic product, slack, and error cell to D. Thus the completed finite ledger is available before the $\mathsf{RegSeqRat}^{\uparrow}$ *row Q or the terminal* $\mathsf{Real}^{\uparrow}$ *seal R can be read.*

Proof. Induct on the displayed finite mesh row M. The empty-prefix case exposes no step width, so the only route prefix is the initial point row A. The matching vertex prefix is the first displayed vertex window in V, and the matching ledger prefix in D is empty except for the accumulated zero-step accounting attached to the initial row.

For the successor case, assume the prefix already has matching displayed vertex windows and accumulated dyadic ledger cells. The next mesh entry of M supplies the single step width allowed for the next advance. The local row L supplies the corresponding slope sample and Lipschitz bound; no external differential-equation object is available in the carrier.

Using exactly those displayed rows, the polygon route appends one vertex window to V. The dyadic ledger D appends the product cell for the step, the slack comparison for the local radius budget, and the accumulated error cell for the enlarged prefix. Hence the extended route prefix again has matching prefixes in V and D.

Finite induction over all displayed mesh entries therefore constructs the full vertex-and-ledger route

$$(M, A, L) \longmapsto V \longmapsto D.$$

The existing $\mathsf{RegSeqRat}^{\uparrow}$ handoff and $\mathsf{Real}^{\uparrow}$ nonescape results consume this completed finite induction ledger: Q can read only the already displayed V, D prefixes, and R is downstream of Q. □

Theorem 1967.4 (Euler polygon RegSeqRat handoff)**.** *For every accepted* $\mathsf{EulerPolygon}^{\uparrow}$ *carrier*

$$E = (M, A, L, V, D, Q, R, H, C, P, N),$$

the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q is determined only by the polygon-vertex* $\mathsf{StreamName}^{\uparrow}$ *windows V together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *step ledger D and the displayed structural rows H, C, P, N. In particular, Q cannot be replaced by a selected solution curve, quotient equality of vertex streams, host real equality, or an ambient completeness witness.*

Proof. Read the packet through the vertex boundary. The only coordinates feeding a regular rational name are the finite vertex windows V and the dyadic step ledger D.

The mesh, initial point, and local slope rows determine which windows and ledger cells are displayed, but they do not themselves export a $\mathsf{RegSeqRat}^{\uparrow}$ object. The handoff to a regular rational name occurs at Q.

The structural rows preserve that same handoff. Componentwise hsame transports V, D, Q, displayed Cont replays their order, and P, N name the route without adding a second readback surface.

Therefore every accepted regular-rational read factors through V, D, Q, H, C, P, N. The carrier contains no coordinate from which a selected solution curve, quotient stream equality, host real equality, or ambient completeness witness could replace Q. □

Theorem 1967.5 (Euler polygon Real-seal nonescape)**.** *For every accepted* $\mathsf{EulerPolygon}^{\uparrow}$ *carrier*

$$E = (M, A, L, V, D, Q, R, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* R *factors through the displayed finite route*

$$(M, A, L) \longmapsto V \longmapsto D \longmapsto Q \longmapsto R$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The seal does not assert existence of an ODE solution, convergence of refined meshes, uniqueness of a limiting curve, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a seal-facing consumer. The carrier exposes R only as the terminal coordinate after mesh data, initial data, local slope control, vertex windows, dyadic step accounting, and regular rational readback have been displayed.

Trace the route backward from R. The seal is downstream of Q, the readback row is downstream of D and V, and the vertex windows are downstream of the mesh, initial point, and local slope/Lipschitz rows.

The structural rows preserve this order. The transport row H is componentwise, the replay row C displays the same finite route, and P, N provide provenance and local naming without adding an independent Real surface.

Thus every Real-facing use of the seal factors through $M, A, L, V, D, Q, R, H, C, P, N$. Since the packet has no coordinate for global solution existence, refined-mesh convergence, limiting-curve uniqueness, host real equality, countable choice, or ambient Real completeness, none of those claims escapes through the terminal seal. □

Falsifiable prediction: Every $\mathsf{EulerPolygon}^{\uparrow}$ Real-facing read factors through a dyadic mesh row, an initial point row, a local slope/Lipschitz row, polygon-vertex StreamName windows, a DyadicRatCore step ledger, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: streamname, regseqrat, dyadicratcore, real, lipschitzmap: $\mathsf{EulerPolygon}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a dyadic mesh, initial point, local slope/Lipschitz data, polygon-vertex windows, step-error accounting, regular rational handoff, terminal Real seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerPolygon}^{\uparrow}}$**.**

Constructive story: An EulerPolygon$^{\uparrow}$ packet is generated from a dyadic mesh, an initial point row, local slope and Lipschitz rows, polygon-vertex StreamName windows, a DyadicRatCore step ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{EulerPolygon}\uparrow}$, seed)

Scope closed: Definition 1967.1, Definition 1967.2, Definition 1967.4, and Definition 1967.5 seed the finite Euler polygon route used by Real and Completion consumers before any terminal seal is read.

Formal verification: FormalStatus(NameCert$_{\mathsf{EulerPolygon}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ODE solution existence, refined-mesh convergence, selected limiting curves, quotient stream equality, host real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh exposure, initial-point anchoring, local slope control, dyadic step accounting, RegSeqRat handoff, Real-seal nonescape, transport/replay/provenance coverage, and non-escape for BEDC.Derived.EulerPolygonUp.

Chapter 1968

A Concrete Naming Certificate for $\mathsf{CauchyModulusNormalForm}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ packet isolates the effective-modulus normalization step used by the L10 Real and Completion route. It sits between the $\mathsf{CauchyModulus}^{\uparrow}$ schedule consumed by chapter 415, the dyadic tail-bound ledger of chapter 379, the finite window service of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a monotone normal form for a displayed modulus before a CauchyRate, RegSeqRat, or Real consumer reads it; it is not Real completeness, quotient stream equality, a selected limit, trichotomy, countable choice, Lean verification, or bridge checking.

Definition 1968.1 (Cauchy modulus normal-form carrier)**.** *A* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (p, \mu, \nu, \tau, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *p is the precision-request row whose finite tolerance demand starts the normalization route;*
2. *μ is the raw* $\mathsf{CauchyModulus}^{\uparrow}$ *row attached to the source Cauchy name;*
3. *ν is the normalized monotone modulus row, displaying a nondecreasing finite window schedule refining μ on every request named by p;*
4. *τ is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *row whose dyadic tail comparisons justify the normalized windows;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected after ν and τ have been fixed;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the normalized window route;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after p, μ, ν, τ, W, Q have been displayed;*
8. *H records componentwise* hsame *transport for the precision, raw modulus, normal-form modulus, tail-bound, window, readback, seal, route, provenance, and local name rows;*
9. *C records the displayed* Cont *replay*

$$p \longmapsto \mu \longmapsto \nu \longmapsto \tau \longmapsto W \longmapsto Q \longmapsto E;$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicTailBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $p, \mu, \nu, \tau, W, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.CauchyModulusNormalFormUp.CauchyModulusNormalFormTasteGate single carrier alignment`

Theorem 1968.2 (Cauchy modulus normal-form NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ *carrier*

$$Z = (p, \mu, \nu, \tau, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusNormalForm}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier exposure for the precision request, raw modulus, normalized monotone modulus, dyadic tail bound, finite window, regular readback, Real seal, and structural rows H, C, P, N;*
2. *normalization exactness, requiring ν to be the displayed nondecreasing normal form refining the raw* $\mathsf{CauchyModulus}^{\uparrow}$ *row μ on the precision requests named by p;*
3. *monotone-window refinement, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be opened through the normalized row ν and justified by the same* $\mathsf{DyadicTailBound}^{\uparrow}$ *row τ;*
4. *RegSeqRat/Real handoff compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and terminal* $\mathsf{Real}^{\uparrow}$ *seal E to factor through p, μ, ν, τ, W in the displayed order;*
5. *ledger non-escape, requiring H, C, P, N to preserve only the displayed route and to export no quotient stream equality, selected limit, host real equality, trichotomy, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Cauchy modulus normal-form obligation is stored outside these rows.

Proof. Project the accepted packet $Z = (p, \mu, \nu, \tau, W, Q, E, H, C, P, N)$. Carrier exposure is immediate because these are exactly the displayed coordinates available to the local classifier.

Normalization exactness is read from ν. It is downstream of the raw row μ and the precision request p, so every normalized demand remains tied to the source Cauchy modulus rather than to an external schedule.

Monotone-window refinement then uses τ and W. The dyadic tail-bound row justifies the finite windows opened by the nondecreasing schedule, and no separate window oracle appears between ν and W.

The readback and seal rows finish the route. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q is obtained only after the normalized windows have been displayed, and the $\mathsf{Real}^{\uparrow}$ boundary E is downstream of that readback.

Finally, H, C, P, N transport, replay, name, and source only the same finite route. Since the carrier has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient completeness, Lean evidence, or bridge checking, the five obligation rows exhaust the certificate surface. □

Theorem 1968.3 (Cauchy modulus normal-form handoff)**.** *For every accepted* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ *carrier*

$$Z = (p, \mu, \nu, \tau, W, Q, E, H, C, P, N),$$

every $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer that uses the normalized modulus factors through the displayed route*

$$p \longmapsto \mu \longmapsto \nu \longmapsto \tau \longmapsto W \longmapsto Q \longmapsto E$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The consumer cannot replace ν by an unrecorded modulus, cannot read W without τ, cannot read Q without the normalized windows, and cannot read E as Real completeness or as a selected limit.*

Proof. Start with a consumer of the normalized modulus. The carrier exposes ν only as the normal-form row downstream of the raw modulus μ and the precision request p.

Read forward along the route. The tail-bound row τ justifies the normalized finite windows, the $\mathsf{StreamName}^{\uparrow}$ row W supplies the inspected window surface, the $\mathsf{RegSeqRat}^{\uparrow}$ row Q is the regular rational readback of that surface, and the $\mathsf{Real}^{\uparrow}$ row E is the terminal seal boundary.

The structural rows preserve the same path. Componentwise transport in H keeps the raw and normalized modulus rows tied to the same tail-bound and window rows, while C replays the route order and P, N name the local certificate.

Thus every CauchyRate, RegSeqRat, or Real-facing use factors through $p, \mu, \nu, \tau, W, Q, E, H, C, P, N$. The packet contains no hidden modulus, detached tail ledger, quotient stream equality, selected limit, countable-choice object, trichotomy witness, ambient Real-completeness principle, Lean evidence, or bridge credential. □

Falsifiable prediction: Every $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ consumer read factors through a precision request, raw CauchyModulus row, normalized monotone modulus row, DyadicTailBound row, StreamName finite-window row, RegSeqRat readback row, RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: cauchyrate, cauchymodulus, dyadictailbound, streamname, regseqrat, real: $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries raw-modulus exposure, monotone normal-form exactness, dyadic tail-bound justification, finite-window refinement, RegSeqRat readback, Real seal factorization, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ packet is generated from a precision request, raw $\mathsf{CauchyModulus}^{\uparrow}$ row, normalized monotone modulus row, $\mathsf{DyadicTailBound}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 1968.1, Definition 1968.2, and Definition 1968.3 seed the finite monotone effective-modulus normal-form boundary for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient stream equality, selected limits, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, normalization exactness, monotone-window refinement, RegSeqRat/Real handoff compatibility, ledger non-escape, and consumer factorization for BEDC.Derived.CauchyModulusNormalForm

Chapter 1969

A Concrete Naming Certificate for $\mathsf{RealIntervalArithmetic}^{\uparrow}$

Following Moore and Bishop-Bridges, the $\mathsf{RealIntervalArithmetic}^{\uparrow}$ packet records constructive interval arithmetic after dyadic enclosures have been admitted to a real-name surface. It sits after the dyadic endpoint operation route of chapter 1645, the located endpoint boundary of chapter 1709, the tail-diameter budget of chapter 1746, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion arguments one finite interval-enclosure consumer over $\mathsf{Real}^{\uparrow}$, while keeping host interval equality, real trichotomy, countable choice, open-cover compactness, and ambient Real completeness outside the carrier.

Definition 1969.1 (Real interval arithmetic carrier)**.** *A* $\mathsf{RealIntervalArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, D, L, T, +_R, \cdot_R, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are input interval rows accepted by* $\mathsf{LocatedRealInterval}^{\uparrow}$*, exposing lower and upper located-real endpoint names together with their finite enclosure ledgers.*
2. *D is the* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ *operation row that computes outward dyadic endpoint enclosures for the requested interval operation before any* $\mathsf{Real}^{\uparrow}$ *seal is read.*
3. *L is the located-real lift row, relating the dyadic enclosure endpoints in D to the located endpoint rows carried by I and J.*
4. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *budget row, recording the requested tail diameter and the tolerance ledger used to justify the exported real interval width.*
5. *$+_R$ is the real-interval addition row, whose output interval enclosure is obtained from the displayed dyadic endpoint sums, located endpoint lifts, and tail-diameter budget.*
6. *$\cdot_R$ is the real-interval multiplication row, whose output interval enclosure is obtained from the displayed dyadic endpoint products, outward comparison rows, located endpoint lifts, and tail-diameter budget.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *enclosure-seal row, requiring every exported interval endpoint and every width claim to pass through the same real-name seal surface.*

8. *H records componentwise* hsame *transport for the input intervals, dyadic operation row, located lift, tail-diameter budget, real operation rows, enclosure seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a requested interval operation through $I, J, D, L, T, +_R, \cdot_R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealIntervalArithmetic}^{\uparrow}$, $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalArithmetic}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive real interval-operation packet. It has no coordinate for host interval equality, trichotomy of real endpoints, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, or a selected host representative.*

Lean4 ✓ `BEDC.Derived.RealIntervalArithmeticUp.RealIntervalArithmeticTasteGate single carrier alignment`

Theorem 1969.2 (Real interval arithmetic NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalArithmetic}^{\uparrow}$ *carrier*

$$A = (I, J, D, L, T, +_R, \cdot_R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalArithmetic}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two located real interval inputs, dyadic interval arithmetic row, located-real lift row, Cauchy tail-diameter budget, real addition and multiplication rows, Real enclosure seal, and structural rows H, C, P, N;*
2. *dyadic-to-real lift discipline, requiring every real interval endpoint exported by $+_R$ or $\cdot_R$ to originate in the displayed dyadic enclosure row D and then pass through the located lift row L;*
3. *tail-diameter discipline, requiring T to be the only width and tolerance budget consumed when a downstream Real or Completion argument reads the exported enclosure;*
4. *operation enclosure soundness, requiring $+_R$ and $\cdot_R$ to preserve containment for all scheduled endpoint windows supplied by I and J;*
5. *seal non-escape, requiring E to be the only* $\mathsf{Real}^{\uparrow}$ *handoff for exported endpoints and interval-width claims;*
6. *transport and replay stability, requiring H and C to preserve the dyadic operation row, located lift, tail budget, operation rows, enclosure seal, provenance, and local name componentwise;*
7. *exclusion, ruling out host interval equality, real trichotomy, countable-choice selectors, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, and selected host representatives as local data.*

No RealIntervalArithmetic obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 1969.1. The projection exposes exactly the two located real interval inputs, the dyadic operation row, located lift, tail-diameter budget, operation rows, Real seal, and structural rows.

The dyadic-to-real lift discipline is the route $D \longmapsto L \longmapsto +_R, \cdot_R$. The dyadic packet supplies outward endpoint enclosures, and the located lift reads those endpoints as located real interval data before any real-name seal is consumed.

The width and tolerance information is stored in T. Thus a Real or Completion consumer can inspect a diameter claim only through the displayed $\mathsf{CauchyTailDiameter}^{\uparrow}$ budget and cannot substitute an ambient completeness argument.

Operation enclosure soundness is read from $+_R$ and $\cdot_R$. The addition row uses displayed dyadic endpoint sums, while the multiplication row uses displayed dyadic endpoint products and outward comparison rows; both rows then pass through L, T, E before reaching a public consumer.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no host interval equality, trichotomy, countable-choice selector, open-cover compactness witness, ambient Real completeness principle, quotient endpoint equality, or selected host representative, none of those objects can be exported. □

Theorem 1969.3 (Real interval arithmetic enclosure soundness)**.** *Every exported operation of an accepted* $\mathsf{RealIntervalArithmetic}^{\uparrow}$ *carrier factors through the displayed route*

$$I, J \longmapsto D \longmapsto L \longmapsto T \longmapsto +_R, \cdot_R \longmapsto E \longmapsto H, C, P, N.$$

Consequently every downstream $\mathsf{Real}^{\uparrow}$ *or Completion consumer receives only a constructive interval enclosure whose endpoints, operation rows, and diameter budget are displayed by* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, *and* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read. The consumer cannot replace this route by host interval equality, trichotomy, countable choice, open-cover compactness, ambient Real completeness, quotient endpoint equality, or a selected host representative.*

Proof. Start with an accepted carrier $A = (I, J, D, L, T, +_R, \cdot_R, E, H, C, P, N)$. The only input interval coordinates are I and J, so every operation begins with displayed located real interval endpoint data.

The dyadic operation row D computes outward endpoint enclosures. The lift row L relates those dyadic endpoints to located real interval endpoints, and the tail-diameter row T records the tolerance budget read by the exported width claim.

For addition, project $+_R$ and read the output enclosure from the displayed endpoint sums, lift, and budget. For multiplication, project $\cdot_R$ and read the output enclosure from the displayed endpoint products, outward comparison rows, lift, and budget.

The seal row E is the only route into $\mathsf{Real}^{\uparrow}$. Componentwise transport and replay through H and C, together with provenance and local naming in P, N, preserve the same route for downstream consumers. Therefore the exported enclosure is exhausted by the displayed finite rows and no excluded host, choice, compactness, completeness, quotient, or representative coordinate is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalArithmetic}^{\uparrow}$ packet is generated from two LocatedRealInterval input rows, one DyadicIntervalArithmetic operation row, a located-real lift, a CauchyTailDiameter budget, displayed addition and multiplication enclosure rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealIntervalArithmetic}^{\uparrow}}$, seed)

Scope closed: Definition 1969.1, Definition 1969.2, and Definition 1969.3 seed the finite constructive interval-enclosure arithmetic interface over RealUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealIntervalArithmetic}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient endpoint equality, selected host representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-to-real lift discipline, tail-diameter discipline, operation enclosure soundness, Real-seal non-escape, transport/replay/provenance coverage, and exclusion for BEDC.Derived.RealIntervalArithmeticUp.

Chapter 1970

A Concrete Naming Certificate for $\mathsf{BishopIntervalCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopIntervalCompletion}^{\uparrow}$ packet records constructive completion of an interval object by finite rational enclosure data and regular real-name readback. It sits after the dyadic rational core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located interval boundary of chapter 394, the rational interval endpoint ledger of chapter 354, and the real interval enclosure route of chapter 1625. The packet gives the L10 Real and Completion phase a finite interval-completion certificate while refusing quotient completion, selected limits, countable choice, trichotomy, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1970.1 (Bishop interval completion carrier)**.** *A* $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, Q, D, E, L, U, A, R, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{RationalInterval}^{\uparrow}$ *or* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row whose lower and upper rational endpoints are admitted before any real-name seal is read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for the interval-completion request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the dyadic radius, endpoint scale, and comparison rows used by the completion window.*
5. *E is the* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *row that displays a lower and upper enclosure around the candidate completed interval endpoint data.*
6. *L and U are the displayed lower and upper completion-bound rows, read from the source interval and the enclosure row rather than from host endpoint values.*
7. *A is the interval admission row saying that the readback Q with tolerance ledger D lies inside the displayed enclosure E and respects the lower and upper completion bounds.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, S, Q, D, E, L, U, A have been displayed.*
9. *T records the interval-completion route ledger from rational endpoint data to located interval data, then to real interval enclosure data, and finally to the terminal real-name seal.*

10. *H records componentwise* hsame *transport for all displayed rows of the packet.*

11. *C records displayed* Cont *replay from a requested completion read through $I, S, Q, D, E, L, U, A, R, T, H, P, N$.*

12. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalCompletion}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive interval-completion packet. It has no coordinate for quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, or a host interval representative.*

Lean4 ✓ `BEDC.Derived.BishopIntervalCompletionUp.BishopIntervalCompletionTasteGate single carrier alignment`

Definition 1970.2 (Bishop interval completion classifier)**.** *The classifier for accepted* $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *carriers compares two packets only by componentwise agreement of the displayed source interval row, stream window, regular rational readback, dyadic tolerance ledger, real interval enclosure row, lower and upper completion bounds, interval admission row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, route ledger, transport, replay, provenance, and local naming rows. Equivalence of carriers is generated by the visible* hsame *rows and by replay of the same* Cont *route; it is not generated by quotient completion, chosen limits, countable-choice selectors, real trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, or selected host representatives.*

Theorem 1970.3 (Bishop interval completion NameCert obligations)**.** *For every accepted* BishopIntervalCompletion *carrier*

$$K = (I, S, Q, D, E, L, U, A, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source interval, stream window, regular rational readback, dyadic tolerance ledger, real interval enclosure, completion bounds, admission row, terminal real seal, route ledger, and structural rows H, C, P, N;*

2. *endpoint source discipline, requiring I to provide the only rational or located interval endpoint data consumed by the packet;*

3. *window-readback discipline, requiring S, Q, D to be read as one finite StreamName–RegSeqRat–DyadicRatCore route;*

4. *enclosure admission, requiring E, L, U, A to bind the inspected readback and dyadic tolerance inside displayed interval bounds;*

5. *real-seal handoff, requiring R to be reachable only after the source interval, finite window, regular readback, dyadic ledger, enclosure, bounds, and admission rows have been displayed;*

6. *route stability, requiring T, H, C to preserve the same interval-completion route under componentwise transport and later finite-window replay;*

7. *exclusion, ruling out quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, and selected host representatives as local data.*

No BishopIntervalCompletion obligation is stored outside these rows.

Proof. Project the carrier of Definition 1970.1. The projection exposes exactly the source interval row, finite stream window, regular rational readback, dyadic tolerance ledger, real interval enclosure, completion bounds, admission row, terminal real seal, route ledger, and structural rows.

The endpoint source discipline is the coordinate I. It is read through the rational and located interval sibling chapters before any completion-facing consumer can inspect the terminal real seal.

The finite observation face is S, Q, D. The stream window supplies the inspected data, the regular rational row reads that window, and the dyadic ledger supplies the tolerance scale used by the interval admission row.

The enclosure face is E, L, U, A. These rows place the inspected readback between displayed lower and upper completion bounds, so the packet exports an interval-completion enclosure rather than a chosen endpoint pair.

Finally R, T, H, C, P, N package and replay the same route. Since no carrier coordinate contains quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient endpoint equality, or selected host representatives, the listed obligations exhaust the local NameCert surface. □

Theorem 1970.4 (Bishop interval completion route separation)**.** *Every accepted* $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *carrier keeps the endpoint-source route and the real-seal route separated until the displayed enclosure admission row is present. In particular, a consumer may read*

$$I \longmapsto S, Q, D \longmapsto E, L, U, A$$

as finite interval-completion evidence, but it may reach the terminal $\mathsf{Real}^{\uparrow}$ *seal* R *only through*

$$I, S, Q, D, E, L, U, A \longmapsto R \longmapsto T, H, C, P, N.$$

Thus no Real or Completion consumer can bypass the displayed interval enclosure by using quotient completion, a selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient endpoint equality, or a selected host representative.*

Proof. Start with an accepted carrier $K = (I, S, Q, D, E, L, U, A, R, T, H, C, P, N)$. The endpoint-source route begins at I and reaches the finite observation rows S, Q, D before any terminal real-name coordinate occurs.

The admission row A is downstream of E, L, U. It binds the regular rational readback and dyadic tolerance to displayed interval bounds, so it is the first point where source endpoint information and finite completion evidence are jointly visible.

The real-seal row R occurs only after that admission face in the carrier. Therefore a consumer seeking a terminal $\mathsf{Real}^{\uparrow}$ read must pass through the displayed source interval, finite window, regular readback, dyadic ledger, enclosure, bounds, and admission rows.

Transport and replay through T, H, C, P, N preserve this order. The route ledger contains no coordinate for quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient completeness, quotient endpoint equality, or selected host representatives, so none of those routes can bypass the displayed admission row. □

Theorem 1970.5 (Bishop interval completion public route)**.** *The public consumer surface of an accepted* $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *carrier is exhausted by the finite route*

$$\mathsf{RationalInterval}^{\uparrow}, \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RealIntervalEnclosure}^{\uparrow} \longmapsto \mathsf{R}$$

as instantiated by $I, S, Q, D, E, L, U, A, R, T, H, C, P, N$. Any downstream L10 Real and Completion argument receives only this displayed constructive interval-completion certificate and not a quotient completion, selected limit, countable-choice principle, trichotomy witness, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient endpoint equality, or host interval representative.*

Proof. Read the public rows in route order. The source interval row I supplies rational or located endpoint data, and the finite observation rows S, Q, D turn that source into a regular rational readback with a dyadic tolerance ledger.

Next the enclosure rows E, L, U, A expose the interval-completion evidence through the same finite data. A public consumer therefore receives lower and upper completion bounds only as displayed rows of the packet.

The terminal seal R is then read as the $\mathsf{Real}^{\uparrow}$ handoff, with T, H, C, P, N preserving the route under transport, replay, provenance, and local naming. The consumer surface is precisely the listed sibling route, so the excluded quotient, choice, trichotomy, host-equality, completeness, quotient-endpoint, and host-representative readings are outside the packet. □

Theorem 1970.6 (Bishop interval completion nested-enclosure induction)**.** *Let*

$$K = (I, S, Q, D, E, L, U, A, R, T, H, C, P, N)$$

be an accepted $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *carrier. For every finite displayed chain of enclosure refinements starting from K, if each successor step refines the visible enclosure row through the same interval-completion route, then every stage preserves the source interval row I, StreamName window S, RegSeqRat readback Q, DyadicRatCore tolerance ledger D, enclosure row E, lower and upper bounds L, U, admission row A, terminal real seal R, route ledger T, and structural rows H, C, P, N as displayed carrier data.*

Proof. Proceed by induction on the finite displayed refinement chain.

For the empty chain, the claim is the carrier projection of Definition 1970.1. The rows $I, S, Q, D, E, L, U, A, R, T, H, C, P, N$ are precisely the visible coordinates of the accepted packet.

At a successor step, the induction hypothesis supplies the same source interval, finite window, rational readback, dyadic tolerance, enclosure, bounds, admission, seal, ledger, and structural rows at the previous stage. The route separation of Definition 1970.4 forces the next refinement to pass through I, S, Q, D, E, L, U, A before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed.

The public route theorem then repacks the successor stage through the same displayed route to $\mathsf{Real}^{\uparrow}$. Since no refinement coordinate contains quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient completeness, quotient endpoint equality, or a host representative, the successor preserves exactly the displayed carrier rows. □

Theorem 1970.7 (Bishop interval completion dyadic window covering)**.** *For any finite family of* $\mathsf{BishopIntervalCompletion}^{\uparrow}$ *completion requests over the same interval source row I, the family is covered by a finite family of StreamName windows and DyadicRatCore tolerance rows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed. More precisely, after reindexing the requests in route order, each request is witnessed by displayed rows*

$$S_j, Q_j, D_j, E_j, L_j, U_j, A_j$$

over the same source I, and the terminal real-name handoff for the family occurs only after these finite window and tolerance rows have been displayed.

Proof. Induct on the finite request family.

For the empty family, no window or tolerance row is required, and there is no terminal real-name handoff to consume.

For a successor family, apply the induction hypothesis to the previous requests. This gives a finite cover by displayed StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, enclosures, bounds, and admission rows over the same interval source.

For the new request, use Definition 1970.6. It preserves the source row I while exposing the successor request through its displayed S, Q, D, E, L, U, A rows. Appending these rows to the previous finite cover gives the required family of windows and dyadic tolerance rows.

Finally, Definition 1970.4 and Definition 1970.5 place every terminal $\mathsf{Real}^{\uparrow}$ seal strictly after the displayed finite-window and tolerance evidence. Thus the family is covered before any real-name seal is consumed. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalCompletion}^{\uparrow}$ packet is generated from a rational or located interval source row, a finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealIntervalEnclosure row, displayed completion bounds, interval admission, terminal RealUp seal, route ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1970.1, Definition 1970.2, Definition 1970.3, Definition 1970.4, Definition 1970.5, Definition 1970.6, and Definition 1970.7 seed the finite constructive interval-completion certificate over RationalInterval, LocatedInterval, StreamName, RegSeqRat, DyadicRatCore, RealIntervalEnclosure, and RealUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BishopIntervalCompletionUp.BishopIntervalCompletionTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient endpoint equality, selected host representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, endpoint source discipline, StreamName–RegSeqRat–DyadicRatCore readback, enclosure admission, RealUp seal non-escape, route separation, and public-route exhaustion for BEDC.Derived.BishopIntervalCompletionUp.

Chapter 1971

A Concrete Naming Certificate for $\mathsf{RegularCauchyScalar}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyScalar}^{\uparrow}$ packet names dyadic scalar multiplication of a regular rational Cauchy name by finite endpoint arithmetic before any completed $\mathsf{Real}^{\uparrow}$ or algebraic consumer can read the scaled name. It sits over the regular rational source row of chapter 301, the dyadic scalar and endpoint arithmetic of chapter 291, the finite-window stream discipline of chapter 126, the terminal real-name seal of chapter 38, and the sibling operation surfaces of chapter 417 and chapter 459. The packet gives later real-algebra and completion consumers a finite scalar route; it does not import quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness.

Definition 1971.1 (RegularCauchyScalar carrier)**.** *A* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, A, W, D, F, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row for the regular rational Cauchy name.*
2. *A is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scalar row, recording the displayed dyadic coefficient and its exact arithmetic provenance.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to expose the source observations requested by the scalar-facing consumer.*
4. *D is the source dyadic observation row read from X through the window schedule W.*
5. *F is the scaled dyadic observation ledger, obtained by multiplying the displayed source observations in D by the scalar row A inside the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic surface.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff, exposed only after X, A, W, D, F and their structural rows have been displayed.*
7. *H records componentwise* hsame *transport for the source row, scalar row, window schedule, source observations, scaled ledger, Real seal, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from every scalar-facing precision request through $X, A, W, D, F, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyScalar}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyScalar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, W, D, F, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyScalarUp.RegularCauchyScalarTasteGate single carrier alignment`

Theorem 1971.2 (RegularCauchyScalar NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *carrier*

$$S = (X, A, W, D, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyScalar}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *scalar row,* $\mathsf{StreamName}^{\uparrow}$ *window schedule, source dyadic observation row, scaled dyadic ledger,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and structural rows;*
2. *scalar-row exactness, requiring A to be read as displayed dyadic arithmetic data before any scaled observation is available;*
3. *finite-window handoff, requiring W and D to expose only finitely many source observations at each scalar-facing precision request;*
4. *scaled-ledger stability, requiring F to be computed from A and D inside the* $\mathsf{DyadicRatCore}^{\uparrow}$ *route and to transport componentwise through H;*
5. *classifier transport, requiring the* hsame *rows in H to preserve the same source, scalar, window, observation, scaled ledger, Real seal, replay, provenance, and local naming data;*
6. *Real seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *or algebraic consumer to pass through X, A, W, D, F before reading E;*
7. *provenance and local naming, requiring P and N to name exactly the finite scalar packet and no external completed-real operation.*

No scalar-multiplication obligation is stored outside these finite rows, and the surface refuses quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local data.*

Proof. Project the accepted carrier. The analytic rows are the source row X, scalar row A, window schedule W, source observations D, scaled ledger F, and Real seal handoff E.

Read the scalar and source windows before the scaled name. The scalar row A supplies exact dyadic arithmetic data, while the finite schedule W exposes only the source observations recorded in D for the current precision request.

Form the scaled ledger inside the displayed dyadic route. The row F consumes A and D and records the endpoint products as finite $\mathsf{DyadicRatCore}^{\uparrow}$ data, so classifier transport through H preserves the same source, scalar, observation, and scaled-ledger coordinates.

Finally replay and package the route through C, P, N. The seal row E is downstream of X, A, W, D, F, so any Real-facing or algebra-facing consumer reads the finite scalar packet before the scaled regular name. Since no carrier coordinate stores quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient completeness, Lean verification, or bridge data, none can become a local NameCert obligation. □

Theorem 1971.3 (RegularCauchyScalar Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or algebraic consumer of an accepted* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *carrier reads the route*

$$(X, A) \longmapsto W \longmapsto D \longmapsto F \longmapsto E \longmapsto (H, C, P, N)$$

before seeing the scaled regular name. Thus the consumer receives the source regular row, exact dyadic scalar row, finite window schedule, source dyadic observations, scaled dyadic ledger, Real seal handoff, displayed transports, replay, provenance, and local naming rows. It receives no quotient stream equality, selected limit, countable-choice selector, trichotomy witness, host real equality, ambient completeness coordinate, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyScalarUp.RegularCauchyScalarRealHandoff`

Proof. Begin with the accepted packet S. The only source of regular-name data is X, and the only scalar source is the displayed dyadic row A.

The finite schedule W supplies the observations D before scalar multiplication is visible. Multiplication by the scalar is then recorded in F as dyadic endpoint arithmetic rather than as a completed-real operation.

The Real-facing boundary is downstream of that ledger. The seal handoff E is exposed only after X, A, W, D, F, and the rows H, C, P, N preserve the same dependency order for algebraic consumers.

Therefore the scaled regular name is read through the finite BEDC scalar route. The listed refusal data are absent from the carrier, so no consumer can obtain them from this handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyScalar}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyScalar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1971.1, Definition 1971.2, and Definition 1971.3 seed the finite dyadic scalar-multiplication surface over a $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{DyadicRatCore}^{\uparrow}$ scalar row, $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, source dyadic observations, scaled dyadic ledger, $\mathsf{Real}^{\uparrow}$ seal handoff, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyScalar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-row exactness, finite-window handoff, scaled-ledger stability, classifier transport, Real seal non-escape, provenance, and local naming for BEDC.Derived.RegularCauchyScalarUp.

Constructive story: A $\mathsf{RegularCauchyScalar}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, a $\mathsf{DyadicRatCore}^{\uparrow}$ scalar row, a $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, a source dyadic observation row, a scaled dyadic observation ledger, a $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1972

A Concrete Naming Certificate for $\mathsf{RoundedDyadicLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Moore–Kearfott–Cloud interval arithmetic, the $\mathsf{RoundedDyadicLocator}^{\uparrow}$ packet exposes outward-rounded dyadic locator rows for $\mathsf{Real}^{\uparrow}$ names. It sits after the located comparison route of chapter 1669, the dyadic finite-observation surface of chapter 335, the interval enclosure ledger of chapter 302, the dyadic request layer of chapter 291, the finite stream window of chapter 126, the regular rational readback row of chapter 301, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers one finite route from a real-name query to an outward-rounded dyadic enclosure, while keeping host trichotomy, quotient real equality, selected representatives, countable choice, and ambient Real completeness outside the carrier.

Definition 1972.1 (Rounded dyadic locator carrier)**.** *A* $\mathsf{RoundedDyadicLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, Q, S, R, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the terminal* $\mathsf{Real}^{\uparrow}$ *seal whose finite observation is being located.*
2. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *request row, naming the queried dyadic center, scale, and finite precision budget.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation window opened from X for the request Q.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream window S.*
5. *D is the dyadic tolerance ledger, recording the radius, endpoint scale, and outward-rounding budget consumed by the locator.*
6. *L is the outward-rounded lower locator row, exposing a dyadic lower endpoint certified by the route Q, S, R, D.*
7. *U is the outward-rounded upper locator row, exposing a dyadic upper endpoint certified by the same route Q, S, R, D.*
8. *H records componentwise* hsame *transport for the Real seal, dyadic request, stream window, regular readback, tolerance ledger, lower and upper locator rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a downstream Real, RealLocator, DyadicApproximation, or IntervalArithmetic request through $Q, S, R, D, L, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RoundedDyadicLocator}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, $\mathsf{IntervalArithmetic}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RoundedDyadicLocator}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite outward-rounded dyadic locator packet. It has no coordinate for host trichotomy, quotient real equality, selected representatives, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1972.2 (Rounded dyadic locator NameCert obligations)**.** *For every accepted* $\mathsf{RoundedDyadicLocator}^{\uparrow}$ *carrier*

$$W = (X, Q, S, R, D, L, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RoundedDyadicLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Real seal, dyadic request, stream window, regular readback, dyadic tolerance ledger, outward-rounded lower and upper locator rows, transport, replay, provenance, and local naming data;*

2. *rounded-bound exactness, requiring L and U to be read as outward-rounded dyadic endpoints justified by the displayed tolerance ledger D and not by a host real-order comparison;*

3. *window-readback compatibility, requiring the stream window S and regular rational readback R to be the only observation route from X to the dyadic endpoints L and U;*

4. *classifier transport, requiring H and C to preserve the same request, window, readback, tolerance ledger, endpoint rows, provenance, and local name componentwise;*

5. *Real-seal non-escape, requiring every downstream locator, approximation, or interval consumer to inspect X only through $Q, S, R, D, L, U, H, C, P, N$.*

No rounded dyadic locator obligation is stored outside these rows, and no obligation may be replaced by host trichotomy, quotient equality of real names, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RoundedDyadicLocatorUp.RoundedDyadicLocatorTasteGate single carrier alignment`

Proof. Project the accepted packet $W = (X, Q, S, R, D, L, U, H, C, P, N)$. The carrier exposes exactly one Real seal, one dyadic request, one stream observation window, one regular rational readback, one dyadic tolerance ledger, two outward-rounded endpoint rows, and the structural rows.

Rounded-bound exactness is the route through D, L, U. The lower endpoint L and upper endpoint U are not host-order facts about a completed real; they are displayed dyadic rows whose rounding direction and tolerance budget are recorded by D.

Window-readback compatibility is forced by the position of S and R in the carrier. A finite observation of X must first open the stream window S and then pass through the regular rational readback R before the dyadic tolerance ledger and endpoint rows are available.

Transport and replay stability are the roles of H and C. They preserve the same request, window, readback, tolerance, endpoint, provenance, and naming coordinates under componentwise hsame transport and displayed continuation replay.

Finally, P and N source and name the same finite packet. Since the carrier inventory contains no host trichotomy coordinate, quotient equality field, representative selector, countable-choice schedule, or ambient completeness witness, none of those objects can appear as a local NameCert obligation. □

Theorem 1972.3 (Rounded dyadic locator window handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLocator}^{\uparrow}$, $\mathsf{DyadicApproximation}^{\uparrow}$, *or* $\mathsf{IntervalArithmetic}^{\uparrow}$ *consumer of an accepted* $\mathsf{RoundedDyadicLocator}^{\uparrow}$ *carrier reads through the displayed handoff route*

$$Q \longmapsto S \longmapsto R \longmapsto D \longmapsto L, U \longmapsto H, C, P, N.$$

The consumer receives only the outward-rounded dyadic lower and upper locator rows certified by that finite route. It cannot replace the route by host trichotomy, quotient equality of real names, a selected representative, countable choice, or ambient Real completeness.

Proof. Let a downstream Real, locator, approximation, or interval consumer inspect an accepted packet W. The request row Q fixes the dyadic center, scale, and finite precision budget before any observation of the real name is read.

The only observation path from the Real seal is $Q \longmapsto S \longmapsto R$. The stream window S exposes the finite part of the name, and the regular rational readback R supplies the rational data from which the tolerance ledger may be inspected.

The ledger D then records the outward-rounding budget, and the endpoint rows L and U expose the lower and upper dyadic locator bounds. Thus the consumer reaches locator or interval data only after the finite window, readback, and tolerance rows have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name that same handoff route. Because no other analytic coordinate occurs in the carrier, the downstream consumer has no access to host trichotomy, quotient real equality, selected representatives, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedDyadicLocator}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedDyadicLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1972.1, Definition 1972.2, and Definition 1972.3 seed the finite outward-rounded dyadic locator route from $\mathsf{Real}^{\uparrow}$ names to locator, approximation, and interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedDyadicLocator}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RoundedDyadicLocatorUp.RoundedDyadicLocatorTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public bridge.

Upgrade path: Obligation closure requires checked carrier admission, rounded-bound exactness, window-readback compatibility, classifier transport, Real-seal non-escape, and window handoff for BEDC.Derived.RoundedDyadicLocatorUp.

Constructive story: A $\mathsf{RoundedDyadicLocator}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{DyadicRatCore}^{\uparrow}$ request, a $\mathsf{StreamName}^{\uparrow}$ observation window, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a dyadic tolerance ledger, outward-rounded lower and upper locator rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1973

A Concrete Naming Certificate for $\mathsf{ArithmeticGeometricMean}^{\uparrow}$

Following Gauss' arithmetic-geometric mean and Bishop–Bridges constructive real analysis, the $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ packet records the finite mean-iteration route available for two displayed positive real names. It sits after the terminal real-name seal of chapter 38, the absolute-value boundary of chapter 1608, the constructive square-root service of chapter 1756, the dyadic arithmetic carrier of chapter 291, the dyadic tolerance scale of chapter 1768, the finite window surface of chapter 126, the regular rational readback route of chapter 301, and the located real boundary of chapter 3359. The packet gives L10 Real and Completion consumers one visible certificate for arithmetic and geometric mean iteration while withholding host real equality, quotient Cauchy equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and standard-bridge data.

Definition 1973.1 (Arithmetic-geometric mean carrier)**.** *An* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, A, G, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are positive* $\mathsf{Real}^{\uparrow}$ *endpoint rows, each exposing a terminal real-name seal, positivity witness, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback data.*
2. *A is the arithmetic midpoint row, recording the displayed half-sum operation from the current endpoint rows before any downstream seal is consumed.*
3. *G is the geometric midpoint row, routing the displayed product of the two positive endpoint rows through* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *and the accompanying square-root readback service.*
4. *D is the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *ledger, storing the precision scale, dyadic radii, comparison slack, and budget cells used by the midpoint and square-root rows.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row selected for the current arithmetic and geometric midpoint requests.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and dyadic tolerance ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported mean-iteration state, reachable only after X, Y, A, G, D, S, Q have been displayed.*

8. *H records componentwise* hsame *transport for the endpoint rows, arithmetic midpoint, geometric midpoint, tolerance ledger, finite windows, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of each accepted precision request through*

$$X, Y \longmapsto A, G \longmapsto D \longmapsto S \longmapsto Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ArithmeticGeometricMean}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, A, G, D, S, Q, E, H, C, P, N$. It has no coordinate for host real equality, quotient Cauchy equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.ArithmeticGeometricMeanUp.ArithmeticGeometricMeanTasteGate single carrier alignment`

Theorem 1973.2 (Arithmetic-geometric mean NameCert obligations)**.** *For every accepted* ArithmeticGeometricMea *carrier*

$$M = (X, Y, A, G, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArithmeticGeometricMean}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two positive* $\mathsf{Real}^{\uparrow}$ *endpoint rows X, Y, the arithmetic midpoint row A, the geometric midpoint and square-root row G, the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and the structural rows H, C, P, N;*

2. *positivity routing, requiring the geometric midpoint row to read the positive endpoint data in X, Y before invoking the square-root service;*

3. *midpoint discipline, requiring A and G to be the only arithmetic and geometric mean rows consumed by the exported iteration state;*

4. *tolerance accounting, requiring D to store exactly the dyadic scale, radii, comparison slack, and budget cells consumed by A, G, S, Q, E;*

5. *window and readback handoff, requiring every precision request to pass through S and Q before the Real seal E is read;*

6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*

7. *non-escape for host real equality, quotient Cauchy equality, selected limits, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and standard-bridge data.*

No arithmetic-geometric-mean obligation is stored outside these rows.

Proof. Project the carrier of Definition 1973.1. The two endpoint rows, arithmetic midpoint, geometric midpoint, tolerance ledger, finite windows, readback, and Real seal are the only analytic coordinates, while H, C, P, N are the only structural coordinates.

The positivity route is forced by the displayed tuple. The geometric midpoint row G reads the positive endpoint data from X and Y, then uses the square-root service named in the provenance row; there is no detached square-root coordinate available to the classifier.

The tolerance and readback rows are downstream of the midpoint rows. The ledger D stores the dyadic scale and slack data consumed by A and G, the window row S exposes the finite observations, and Q is the regular rational readback that precedes the terminal seal.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded host, quotient, choice, completeness, verification, or bridge objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 1973.3 (Arithmetic-geometric mean Real-seal handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ *carrier reaches the exported mean-iteration seal only through the displayed route*

$$X, Y \longmapsto A, G \longmapsto D \longmapsto S \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives two positive Real endpoint rows, an arithmetic midpoint row, a geometric midpoint routed through $\mathsf{RegularCauchySquareRoot}^{\uparrow}$*, a dyadic tolerance ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming. The handoff supplies no host real equality, quotient Cauchy equality, selected limit, trichotomy witness, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean evidence, or standard bridge.*

Proof. Start with an accepted carrier $M = (X, Y, A, G, D, S, Q, E, H, C, P, N)$. The endpoint rows X and Y are the only sources for the two positive real names, so every accepted midpoint request begins with displayed Real seals and their finite observation data.

The arithmetic midpoint row A records the half-sum route. The geometric midpoint row G records the product-and-square-root route, and the square-root service is read only through the displayed positive endpoint data and the provenance named by the same carrier.

The dyadic tolerance row D then fixes the precision scale consumed by both midpoint rows. The finite windows S and regular rational readback Q are the only observation rows from which a downstream consumer can proceed to the terminal seal E.

Componentwise transport and replay through H and C, together with provenance and local naming in P, N, preserve the same route. The carrier contains no coordinate for the excluded host, quotient, choice, completeness, verification, or bridge data, so the displayed handoff is exhaustive. □

Theorem 1973.4 (ArithmeticGeometricMean window contraction)**.** *For an accepted* $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ *carrier*

$$M = (X, Y, A, G, D, S, Q, E, H, C, P, N),$$

each finite precision window exported by S is contracted through the displayed arithmetic row A, geometric row G, dyadic arithmetic and tolerance data from chapter 291, $\mathsf{StreamName}^{\uparrow}$ *windows from chapter 126, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback from chapter 301 before the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 or the located boundary of chapter 3359 is exposed. The contraction is a finite packet property; it does not select a quotient real equality, choose square roots beyond the displayed G row, decide trichotomy, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier in Definition 1973.1. The endpoint rows X, Y supply the two positive Real names, and the arithmetic and geometric midpoint rows A, G are the only rows that can narrow the displayed interval for the next finite request.

Apply the obligation theorem Definition 1973.2. Its tolerance accounting and window handoff clauses force the dyadic arithmetic and scale data through D, then the finite observations through S, and finally the regular rational readback through Q before any terminal real-facing row is consumed.

Use the Real-seal handoff of Definition 1973.3. The seal row E and the structural rows H, C, P, N preserve the same contracted finite route, so no selected quotient equality, external square-root selector, trichotomy witness, or ambient completeness principle can be inserted after the window has been read. □

Falsifiable prediction: Every $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ precision read factors through two positive RealUp endpoint rows, arithmetic midpoint data, geometric midpoint and square-root data, a DyadicToleranceScale ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, real-abs, regular-cauchy-square-root, dyadic-tolerance-scale, streamname, regseqrat: $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two positive endpoint rows, arithmetic midpoint routing, geometric midpoint square-root routing, dyadic tolerance accounting, finite window selection, regular readback, terminal Real-seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmeticGeometricMean}^{\uparrow}}$.

Constructive story: An $\mathsf{ArithmeticGeometricMean}^{\uparrow}$ packet is generated from two positive RealUp endpoint rows, one arithmetic midpoint row, one geometric midpoint routed through RegularCauchySquareRoot, a DyadicToleranceScale ledger, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArithmeticGeometricMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1973.1, Definition 1973.2, Definition 1973.3, and Definition 1973.4 seed the finite constructive arithmetic-geometric mean route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArithmeticGeometricMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient Cauchy equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, positivity routing, midpoint discipline, tolerance accounting, window/readback handoff, Real-seal non-escape, transport/replay/provenance coverage, and exclusion for BEDC.Derived.ArithmeticGeometricMeanUp.

Chapter 1974

A Concrete Naming Certificate for $\mathsf{KroneckerApproximation}^{\uparrow}$

Following Hardy–Wright and Cassels on Kronecker approximation, the $\mathsf{KroneckerApproximation}^{\uparrow}$ packet names only the finite orbit-window route by which a displayed multiplier sends a source real name into a target tolerance window. It records finite orbit-window data, rational multiplier ledgers, dyadic tolerance rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RealRationalApproximation}^{\uparrow}$ handoff, and a terminal $\mathsf{Real}^{\uparrow}$ seal. The packet consumes the denominator-bounded route of chapter 1871, the rational-tree witnesses of chapter 1704 and chapter 1759, the dyadic tolerance ledger of chapter 291, and the real-facing approximation surface of chapter 1668; it does not import quotient reals, selected infinite orbits, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 1974.1 (Kronecker approximation carrier)**.** *A* $\mathsf{KroneckerApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, M, D, T, U, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-name row. It exposes only the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows of the source point requested by the approximation read.*
2. *W is the target orbit-window row. It stores the finite list of requested orbit indices, target rational windows, and comparison slots visible to the certificate.*
3. *M is the integer and rational multiplier ledger. It records the inspected integer multiples, rational multiplier presentations, denominator bounds, and visible arithmetic comparisons.*
4. *D is the* $\mathsf{DirichletApproximation}^{\uparrow}$ *handoff row. It supplies the denominator-bounded rational-approximation request from which the multiplier ledger may be read.*
5. *T is the Stern–Brocot or Farey witness row. It records the finite branch, mediant, or neighboring-fraction data certifying the rational candidate used by D.*
6. *U is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the dyadic tolerance, slack ledger, and endpoint comparisons used to accept the finite orbit window.*
7. *G is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target reads inspected by the orbit request.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It converts the displayed StreamName windows into regular rational data before any real-facing approximation handoff is exposed.*

9. *E is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *handoff row. It packages the multiplier ledger, target window, dyadic tolerance, StreamName windows, and RegSeqRat readback as a finite approximation surface.*

10. *H is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It can be read only after $S, W, M, D, T, U, G, R, E$ have been displayed.*

11. *C stores componentwise* hsame *transport and displayed* Cont *replay for the source name, target window, multiplier ledger, Dirichlet handoff, Stern–Brocot or Farey witness, dyadic tolerance, StreamName window, RegSeqRat readback, RealRationalApproximation handoff, Real seal, provenance, and local naming rows.*

12. *P is the* Pkg *provenance over* $\mathsf{KroneckerApproximation}^{\uparrow}$, $\mathsf{DirichletApproximation}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{KroneckerApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, M, D, T, U, G, R, E, H, C, P, N$. It has no coordinate for host real equality, quotient real representatives, trichotomy, countable choice, a selected infinite orbit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.KroneckerApproximationUp.KroneckerApproximationTasteGate single carrier alignment`

Theorem 1974.2 (Kronecker approximation NameCert obligations)**.** *For every accepted* $\mathsf{KroneckerApproximation}^{\uparrow}$ *carrier*

$$K = (S, W, M, D, T, U, G, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KroneckerApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source real name, target orbit window, multiplier ledger, Dirichlet handoff, Stern–Brocot or Farey witness, dyadic tolerance, StreamName window, RegSeqRat readback, RealRationalApproximation handoff, Real seal, transport and replay ledger, provenance, and local naming row;*

2. *orbit-window stability, requiring each accepted target comparison to be read from the finite window W and preserved by the multiplier ledger M;*

3. *rational multiplier source, requiring every exported integer or rational multiple to pass through M and the denominator-bounded handoff D before it can be used by the orbit window;*

4. *diophantine approximation handoff, requiring the rational candidate to be witnessed through D and the Stern–Brocot or Farey row T rather than through an external selector;*

5. *real-seal factorization, requiring every real-facing approximation read to pass through U, G, R, E and then the terminal seal H;*

6. *ledger non-escape, requiring C, P, N to preserve, source, and name only the displayed rows and to exclude host real equality, quotient real representatives, trichotomy, countable choice, selected infinite orbits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Theorem 1974.3 (Kronecker approximation finite orbit window)**.** *Every accepted* $\mathsf{KroneckerApproximation}^{\uparrow}$ *approximation read reaching a* $\mathsf{RealRationalApproximation}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite route*

$$S, W, M \longmapsto D, T, U, G, R, E, H \longmapsto (C, P, N).$$

Equivalently, the read passes through DirichletApproximation$^{\uparrow}$, *either* SternBrocot$^{\uparrow}$ *or* FareySequence$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealRationalApproximation$^{\uparrow}$, *and* Real$^{\uparrow}$ *before transport, replay, provenance, or local naming can be inspected. No accepted read exposes host real equality, quotient real representatives, trichotomy, countable choice, a selected infinite orbit, or ambient completeness.*

Closure status for NameCert$_{\text{KroneckerApproximation}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{KroneckerApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 1974.1, Definition 1974.2, and Definition 1974.3 seed the finite Kronecker approximation route through orbit windows, rational multipliers, dyadic tolerance, StreamName, RegSeqRat, RealRationalApproximation, and terminal Real rows.

Formal verification: FormalStatus(NameCert$_{\text{KroneckerApproximation}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.KroneckerApproximationUp.KroneckerApproximationTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient real representatives, trichotomy, countable choice, selected infinite orbits, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, orbit-window stability, rational multiplier sourcing, diophantine approximation handoff, real-seal factorization, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.KroneckerApproximationUp.

Constructive story: A KroneckerApproximation$^{\uparrow}$ packet is generated from source real-name windows, a target orbit window, integer and rational multiplier ledgers, a DirichletApproximation handoff, a SternBrocot or FareySequence witness row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealRationalApproximation handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1975

A Concrete Naming Certificate for $\mathsf{LocatedCutApproximation}^{\uparrow}$

Following Bishop and Bridges on located real-number data, the $\mathsf{LocatedCutApproximation}^{\uparrow}$ packet records the finite approximation layer between the located lower/upper dyadic windows of chapter 404 and the regular-Cauchy route consumed by chapter 301 and chapter 38. It keeps the refusal boundary of chapter 1745 intact: the packet observes located cut windows and hands them to the Cauchy-facing route, while chapter 1424 remains only a comparison sibling. Its dependency surface is the finite dyadic ledger of chapter 291, the stream-window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The chapter names only finite observation and handoff rows; it is not a Dedekind construction of $\mathsf{Real}^{\uparrow}$, a quotient-cut object, a selected supremum, trichotomy, choice, Quot.sound, propext, or ambient completeness.

Definition 1975.1 (Located cut approximation carrier)**.** *A* $\mathsf{LocatedCutApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, W, A, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located dyadic row read from* $\mathsf{LocatedCut}^{\uparrow}$*, recording finite rational witnesses below the cut.*
2. *U is the upper located dyadic row read from* $\mathsf{LocatedCut}^{\uparrow}$*, recording finite rational witnesses above the cut.*
3. *W is the located window row binding the lower and upper dyadic observations to one inspected precision request.*
4. *A is the approximation row, storing the finite midpoint, endpoint, or bracket readback exported by L, U, W.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the approximation row is read as finite stream data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained after the displayed stream windows and approximation row have been fixed.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack comparisons, and endpoint budgets used by W, A, S, R.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, W, A, S, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the lower row, upper row, located window, approximation row, stream windows, regular handoff, dyadic ledger, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$L, U \longmapsto W \longmapsto A \longmapsto S \longmapsto R \longmapsto D \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCutApproximation}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, W, A, S, R, D, E, H, C, P, N$. It has no coordinate for quotient cuts, selected suprema, trichotomy, choice, Quot.sound*, propext, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedCutApproximationUp.LocatedCutApproximationTasteGate single carrier alignment`

Theorem 1975.2 (Located cut approximation NameCert obligations)**.** *For every accepted* LocatedCutApproximatio *carrier*

$$Q = (L, U, W, A, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCutApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed lower row, upper row, located window, approximation row, stream windows, regular-Cauchy handoff, dyadic tolerance ledger, Real seal, and structural rows H, C, P, N;*

2. *located-window exposure, requiring every approximation request to display both L and U through the common window row W before any stream or regular-Cauchy read can be consumed;*

3. *approximation exactness, requiring A to store only finite dyadic readback data obtained from L, U, W and the tolerance ledger D;*

4. *stream-window readiness, requiring S to expose the finite* $\mathsf{StreamName}^{\uparrow}$ *windows used by the approximation row before R is read;*

5. *regular-Cauchy handoff, requiring R to be the only route by which the located-window approximation becomes a* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*

6. *Real-seal factorization, requiring every downstream read of E to pass through L, U, W, A, S, R, D and the structural rows;*

7. *transport, replay, and provenance non-escape, requiring H, C, P, N to preserve and name exactly the displayed rows, with no Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, quotient cut, selected supremum, trichotomy, choice,* Quot.sound*, propext, ambient completeness, Lean verification, or bridge credential.*

No located-cut approximation obligation is stored outside these rows.

Proof. Project the accepted packet $Q = (L, U, W, A, S, R, D, E, H, C, P, N)$. Carrier admission exposes exactly the lower and upper located dyadic rows, the common located window, the finite

approximation row, the stream windows, the regular-Cauchy handoff, the dyadic tolerance ledger, the Real seal, and the four structural rows.

The located-window obligation is the projection to W together with its source rows L and U. A precision request cannot inspect a lower row or an upper row separately and then enter the Cauchy route; the common window row records the finite bracket in which both sides are read.

Approximation exactness is then the projection to A and D. The approximation row stores only the displayed dyadic readback supported by the tolerance ledger, so it cannot introduce quotient-cut equality, a selected supremum, trichotomy, or choice.

The stream and regular handoff obligations are the projections to S and R. The stream row exposes the finite windows used by the approximation, and the regular row is downstream of those windows rather than a detached regular representative.

Finally, the Real seal and structural rows exhaust the boundary. The seal E is downstream of L, U, W, A, S, R, D, while H, C, P, N transport, replay, name, and source only those displayed rows. Hence no Dedekind construction of $\mathsf{Real}^{\uparrow}$, quotient cut, selected supremum, trichotomy, choice, $\mathsf{Quot.sound}$, propext, ambient completeness, Lean verification, or bridge credential is exported. □

Theorem 1975.3 (Located cut approximation regular-Cauchy handoff)**.** *For every accepted* LocatedCutApproximat *carrier*

$$Q = (L, U, W, A, S, R, D, E, H, C, P, N),$$

every $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* Q *factors through the displayed route*

$$\mathsf{LocatedCut}^{\uparrow} \longrightarrow \mathsf{LocatedCutApproximation}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}$$

by reading the finite rows $L, U, W, A, S, R, D, H, C, P, N$*. In particular, the handoff does not use a Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, quotient cut equality, a selected supremum, trichotomy, choice,* $\mathsf{Quot.sound}$*, propext, or ambient completion data.*

Proof. Begin with a regular-Cauchy consumer. Carrier projection exposes the handoff row R only after the source located rows L, U, the common located window W, the approximation row A, the stream windows S, and the dyadic tolerance ledger D have been displayed.

The route inspection is finite. The lower and upper located rows determine the observed bracket, W binds them to a single precision request, A records the approximation readback, and S opens exactly the stream windows used by that readback.

The handoff row R is therefore downstream of the displayed stream route. Componentwise transport and replay through H and C preserve the same order, while P and N only name the visible packet.

Since no carrier coordinate stores quotient cuts, selected suprema, trichotomy data, choice data, $\mathsf{Quot.sound}$, propext, or ambient completeness, no $\mathsf{RegSeqRat}^{\uparrow}$ read exported by the packet can factor through those excluded sources. □

Theorem 1975.4 (Located cut approximation Real-seal nonescape)**.** *For every accepted* $\mathsf{LocatedCutApproximation}^{\uparrow}$ *carrier*

$$Q = (L, U, W, A, S, R, D, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* E *receives only the displayed located-window approximation route*

$$L, U, W, A, S, R, D, E, H, C, P, N.$$

The seal is not a Dedekind construction of $\mathsf{Real}^{\uparrow}$*, not a quotient-cut completion, not a selected supremum, not a trichotomy witness, not a choice object, not a* $\mathsf{Quot.sound}$ *or propext bridge, and not an ambient completeness theorem.*

Proof. Start from the terminal seal E. In the carrier, E is not an independent real-completion coordinate; it is the endpoint of the displayed route through the located lower and upper rows, the common located window, the finite approximation row, the stream windows, the regular handoff, and the dyadic tolerance ledger.

Trace the route backward. The Real-facing read reaches E only after R has been read, R only after S and A have been displayed, and A only after the lower and upper located rows have been bound by W and checked against D.

The structural coordinates do not add a second seal source. The transport row H is componentwise, the replay row C displays the same route order, and the provenance and local certificate rows P, N name this packet rather than importing a host completion principle.

Thus every Real-facing consumer is confined to $L, U, W, A, S, R, D, E, H, C, P, N$. The excluded Dedekind, quotient-cut, selected-supremum, trichotomy, choice, Quot.sound, propext, and ambient-completeness readings are outside the carrier inventory. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCutApproximation}^{\uparrow}$ packet is generated from lower and upper located dyadic rows, a common located window, a finite approximation row, StreamName windows, a RegSeqRat handoff, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1975.1, Definition 1975.2, Definition 1975.3, and Definition 1975.4 seed the finite located-window approximation packet between located-cut observations and regular-Cauchy Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Dedekind construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, selected suprema, trichotomy, choice, Quot.sound, propext, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window exposure, approximation exactness, stream-window readiness, regular-Cauchy handoff, Real-seal factorization, transport/replay/provenance coverage, and non-escape for BEDC.Derived.LocatedCutApproximationUp.

Chapter 1976

A Concrete Naming Certificate for $\mathsf{SignedBinaryExpansion}^{\uparrow}$

Following Avizienis signed-digit arithmetic and Bishop–Bridges constructive analysis, the $\mathsf{SignedBinaryExpansion}^{\uparrow}$ packet records finite signed-binary real readback before any real-name consumer may treat a signed prefix as a terminal seal. It sits after the raw binary expansion prefix surface of chapter 418 and the endpoint uniqueness boundary of chapter 420, while using the signed-digit stream handoff of chapter 316, the dyadic ledger of chapter 291, the finite schedule of chapter 126, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet is a finite readback certificate for displayed sign and digit windows; it is not quotient stream equality, not a selected representative of an infinite expansion, and not an ambient completeness principle.

Definition 1976.1 (SignedBinaryExpansion carrier)**.** *A* $\mathsf{SignedBinaryExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, B, U, D, W, E, Y, R, L, H, C, P, N)$$

with the following visible rows:

1. *Q is the sign row, exposing the displayed positive, negative, or zero-facing prefix orientation used by the readback.*
2. *B is the accepted binary expansion finite prefix row whose unsigned digit windows are read before signs are attached.*
3. *U is the accepted* $\mathsf{BinaryEndpointUniqueness}^{\uparrow}$ *row recording the endpoint-normalization data and excluding a second binary endpoint convention.*
4. *D is the signed digit-prefix row, displayed as the local signed-binary digit window before it is handed to the signed stream interface.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule selecting exactly the observed digit prefixes and their requested precisions.*
6. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure row recording the dyadic center and radius attached to the signed prefix window.*
7. *Y is the signed-digit stream handoff row that repacks the sign, digit-prefix, carry, and endpoint data as a signed-digit stream window.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed window schedule and dyadic enclosure rows.*

9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, B, U, D, W, E, Y, R have been displayed.*
10. *H records componentwise* hsame *transport for the sign, binary prefix, endpoint-normalization, signed digit, window, enclosure, stream handoff, regular handoff, and seal rows.*
11. *C records displayed* Cont *replay from a consumer request through the sign and digit windows, then the endpoint-normalization row, dyadic enclosure rows, signed-digit stream handoff, regular rational handoff, and terminal seal.*
12. *P is the* Pkg *provenance over* $\mathsf{SignedBinaryExpansion}^{\uparrow}$*, the binary expansion prefix surface,* $\mathsf{BinaryEndpointUniqueness}^{\uparrow}$*, the signed-digit stream handoff,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedBinaryExpansion}^{\uparrow}}$ *row.*

The carrier predicate accepts only these finite rows. It has no coordinate for quotient stream equality, an infinite selected representative, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SignedBinaryExpansionUp`

Theorem 1976.2 (SignedBinaryExpansion NameCert obligations)**.** *For every accepted* $\mathsf{SignedBinaryExpansion}^{\uparrow}$ *carrier $S = (Q, B, U, D, W, E, Y, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SignedBinaryExpansion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the sign row, binary prefix row, endpoint-normalization row, signed digit-prefix row, finite window schedule, dyadic enclosure row, signed-digit stream handoff, regular rational handoff, Real seal, componentwise transport, continuation replay, provenance, and local naming;*
2. *endpoint-normalization exactness, requiring U to be consumed before any signed digit prefix is interpreted as a unique endpoint readback;*
3. *signed-window stability, requiring D and W to be preserved componentwise under the displayed* hsame *transport H and replayed by the displayed* Cont *row C;*
4. *dyadic enclosure visibility, requiring every regular rational or real-name consumer to inspect the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row E before reading Y, R, or L;*
5. *handoff order, requiring the signed-digit stream row Y to precede the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the terminal* $\mathsf{Real}^{\uparrow}$ *seal L;*
6. *provenance locality, requiring the package row P and local NameCert row N to expose no quotient stream equality, selected representative, trichotomy, countable-choice selector, or ambient completeness coordinate.*

No signed-binary expansion obligation is stored outside these rows.

Proof. Project the carrier of Definition 1976.1. The tuple exposes exactly the sign, binary prefix, endpoint-normalization, signed digit, window schedule, dyadic enclosure, signed-stream handoff, regular handoff, real seal, and structural rows.

The endpoint-normalization clause is separate from the raw prefix read. The row B supplies only finite binary windows, while U supplies the endpoint-normalization data inherited from chapter 420.

The signed-window clause then reads D under the finite schedule W. Transport through H and replay through C preserve the same displayed rows before any downstream consumer is allowed to inspect the enclosure row E.

The handoff rows are ordered. The signed digit stream row Y reads the sign and digit-prefix data first, the regular rational row R reads the scheduled dyadic enclosures next, and the real seal L is terminal.

Finally, P and N record only the package and local naming route. Since quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, and ambient completeness are absent from the tuple, the listed rows exhaust the seed obligation surface. □

Theorem 1976.3 (SignedBinaryExpansion real readback boundary)**.** *Let $S = (Q, B, U, D, W, E, Y, R, L, H, C, P, N$ be an accepted* $\mathsf{SignedBinaryExpansion}^{\uparrow}$ *carrier. Every finite signed-binary readback consumed as a regular rational name or as a terminal real-name seal factors through the displayed route*

$$Q, B, U, D, W \longmapsto E \longmapsto Y \longmapsto R \longmapsto L$$

with structural support H, C, P, N. Thus the consumer receives a sign row, finite binary prefix row, endpoint-normalization row, signed digit-prefix window, dyadic enclosure row, signed-digit stream handoff, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal, all as finite BEDC data.*

The boundary supplies no quotient equality of streams, no selected representative of an infinite expansion, no trichotomy row, no countable-choice row, no host real equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.SignedBinaryExpansionUp.SignedBinaryExpansionCarrier real readback boundary`

Proof. Start with a downstream consumer of R or L. By the carrier definition, neither row is an independent real object; both are downstream of the displayed sign and digit-window route.

The consumer first reads the sign row Q, the raw binary prefix row B, and the endpoint-normalization row U. These rows fix the finite signed-prefix presentation and the endpoint convention before any dyadic enclosure is visible.

Next the schedule W selects the observed digit prefix D and exposes the dyadic enclosure E. The enclosure row is the only row that a regular rational or real-name consumer may inspect as a numeric readback of the prefix.

The signed digit stream handoff Y repacks the same finite data for signed-stream consumers, and the regular handoff R reads that package before the terminal seal L is available.

The structural rows H, C, P, N transport, replay, package, and name the same route. Because no excluded coordinate appears in the packet, every real readback remains inside the displayed finite boundary. □

Lean4 ✓ `BEDC.Derived.SignedBinaryExpansionUp.TasteGate.SignedBinaryExpansion real readback boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{SignedBinaryExpansion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedBinaryExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1976.1, Definition 1976.2, and Definition 1976.3 seed the finite signed-binary readback route from binary endpoint normalization through signed-digit stream windows, dyadic enclosures, regular rational handoff, and terminal real seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedBinaryExpansion}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-normalization exactness, signed-window stability, dyadic enclosure visibility, handoff order, and real readback boundary proofs for the same packet.

Constructive story: A $\mathsf{SignedBinaryExpansion}^{\uparrow}$ packet is generated from a sign row, finite BinaryExpansion prefix row, BinaryEndpointUniqueness row, signed digit-prefix window, StreamName schedule, DyadicRatCore enclosure row, SignedDigitStream handoff, RegSeqRat handoff, terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1977

A Concrete Naming Certificate for $\mathsf{LesserLimitedOmniscience}^{\uparrow}$

Following Brouwer, Bishop–Bridges, and Troelstra–van Dalen constructive analysis, the $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ packet records the finite boundary at which a two-alternative inspection may be named without accepting lesser limited omniscience as a global principle. It is read beside the Specker obstruction of chapter 1406, the Markov boundary of chapter 1666, the fan-theorem boundary of chapter 400, the stream schedule of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a Real/Completion boundary object: it displays the finite rows a consumer may inspect before any $\mathsf{Real}^{\uparrow}$ seal read and refuses every route from those rows to LPO, trichotomy, quotient real equality, selected limits, countable choice, open-cover compactness, or ambient real completeness.

Definition 1977.1 (Lesser limited omniscience carrier)**.** *A* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A_0, A_1, S, R, D, E, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A_0 is the first alternative row, naming the finite schedule branch on which the left disjunct would be certified if the inspected evidence reaches it.*
2. *A_1 is the second alternative row, naming the finite schedule branch on which the right disjunct would be certified if the inspected evidence reaches it.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row listing the finite stream requests admitted by the boundary.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational values actually inspected from S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the dyadic radii, endpoint comparisons, and finite margin rows used to keep A_0 and A_1 apart.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the rows A_0, A_1, S, R, D have been displayed.*
7. *W is the refusal/witness row: it either records the displayed finite alternative witnessed by the inspected packet or records the finite refusal that no global LLPO principle is exported.*
8. *H stores componentwise* hsame *transport for the two alternative rows, stream schedule, regular-rational readback, dyadic ledger, real seal, witness/refusal row, continuation row, provenance row, and local name.*

9. *C stores* Cont *replay from finite stream requests through R and D, then through the alternative rows A_0, A_1, and finally into the sealed real-facing row W.*
10. *P records* Pkg *provenance over* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}$ *row.*

The carrier has no coordinate for imported LLPO, LPO, trichotomy, countable choice, quotient stream equality, quotient real equality, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LesserLimitedOmniscienceUp.LesserLimitedOmniscienceTasteGate single carrier alignment`

Theorem 1977.2 (Lesser limited omniscience NameCert obligations)**.** *For every accepted* LesserLimitedOmniscience *carrier $L = (A_0, A_1, S, R, D, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier exposure for the two alternative rows, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, refusal/witness row, transport, continuation, package, and local naming rows;*
2. *classifier componentwise transport, requiring every comparison between packets to use the* hsame *rows stored in H coordinate by coordinate;*
3. *alternative-row disjointness, requiring any accepted left/right alternative to cite the displayed dyadic separation and finite margin data in D;*
4. *real-seal handoff, requiring every Real/Completion consumer to reach E only after the schedule, readback, ledger, alternatives, and refusal/witness row have been displayed;*
5. *ledger nonescape from imported LLPO, LPO, trichotomy, countable choice, quotient stream equality, quotient real equality, selected limits, open-cover compactness, and ambient real completeness.*

No NameCert obligation is stored outside these five rows.

Proof. Carrier exposure is projection from Definition 1977.1. The visible data are the two alternative rows A_0, A_1, the schedule S, readback R, tolerance ledger D, terminal seal E, refusal/witness row W, and structural rows H, C, P, N.

Classifier transport is componentwise because the packet offers no aggregate classifier coordinate. A comparison of accepted packets must therefore use the displayed hsame rows in H for each alternative, readback, ledger, seal, witness, replay, provenance, and local name.

Alternative-row disjointness is forced by the tolerance ledger D. A branch may expose an alternative only by naming a dyadic radius, endpoint comparison, and finite margin row already present in the ledger; otherwise W records only the refusal of a global disjunctive principle.

The real-seal handoff is downstream of E, and the replay path is supplied by C. Since the carrier contains no coordinate for imported LLPO, LPO, trichotomy, choice, quotient equality, selected limits, open-cover compactness, or ambient completeness, those principles cannot be exported through the local $\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}$ row. □

Theorem 1977.3 (Lesser limited omniscience nonescape)**.** *Every Real/Completion consumer of an accepted* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ *packet receives only the two displayed alternative rows, finite*

schedule, regular-rational readback, dyadic ledger, terminal real seal, refusal/witness row, componentwise transport, continuation replay, provenance, and local NameCert row. It receives no imported LLPO, no LPO, no trichotomy, no countable choice, no quotient stream equality, no quotient real equality, no selected limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Let a Real/Completion consumer accept $L = (A_0, A_1, S, R, D, E, W, H, C, P, N)$. By Definition 1977.1, the only real-facing coordinate is the seal E, and that seal is reached after the two alternatives, stream schedule, regular-rational readback, dyadic ledger, and refusal/witness row have been displayed.

The consumer's branch information is therefore finite. It can replay the schedule through R, cite only dyadic tolerance rows from D, and transport the two alternatives through H and C.

No omitted semi-classical or completion principle is a coordinate of the packet. Thus factorization through the displayed rows cannot export imported LLPO, LPO, trichotomy, countable choice, quotient stream equality, quotient real equality, selected limits, or ambient real completeness. □

Theorem 1977.4 (Specker, Markov, and fan boundary factorization)**.** *Every* $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, *or* $\mathsf{FanTheorem}^{\uparrow}$ *consumer using an accepted* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ *packet factors through the displayed finite rows*

$$A_0, A_1, S, R, D, E, W, H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *seal read.*

Proof. The carrier names the two alternatives and the finite stream schedule before the real seal. Thus a Specker consumer can inspect only the displayed obstruction-facing schedule and readback rows S, R before it reaches E.

A Markov-boundary consumer reaches its negative-evidence boundary through the same finite rows. It may cite W as a refusal/witness row, but W is downstream of A_0, A_1, S, R, D and therefore cannot introduce an unlisted principle.

A fan-theorem consumer likewise receives only the finite branch rows and the dyadic ledger before the seal. Componentwise transport in H and continuation replay in C preserve this route, while P and N repackage the same finite data as the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}$.

Constructive story: A $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ packet is generated from two disjoint alternative rows, a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, refusal/witness row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1977.1, Definition 1977.2, Definition 1977.3, and Definition 1977.4 seed the finite LLPO boundary for Real/Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LesserLimitedOmniscience}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LesserLimitedOmniscienceUp.LesserLimitedOmniscienceTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not claim LLPO as a global principle, LPO, trichotomy, countable choice, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier componentwise transport, alternative-row disjointness, real-seal handoff, Specker/Markov/Fan factorization, and ledger nonescape for the same finite packet.

Chapter 1978

A Concrete Naming Certificate for $\mathsf{LLPOBoundary}^{\uparrow}$

Following Brouwer and Bishop–Bridges constructive analysis, the $\mathsf{LLPOBoundary}^{\uparrow}$ packet records the finite Real/Completion boundary at which a binary zerohood inspection may be named without importing limited principles of omniscience. It is read beside the Ishihara inspection packet of chapter 1665, the Specker obstruction of chapter 1406, the Markov boundary of chapter 1666, the stream schedule of chapter 126, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a positive refusal surface for later real-comparison and completion arguments: consumers may cite the displayed finite rows, but they do not receive an imported LLPO principle, an LPO test, a WLPO test, trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

YoungIntegral-style controlled Riemann-sum consumers may read this boundary only as the finite zerohood and refusal surface used by chapter 3910; the read supplies no binary decision rule for choosing a hidden mesh branch.

Definition 1978.1 (LLPO boundary carrier)**.** *An* $\mathsf{LLPOBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Z_0, Z_1, S, R, D, E, W, H, C, P, N)$$

with the following displayed coordinates:

1. Z_0 *is the left zerohood row, naming the finite stream positions whose vanishing would certify the left inspection alternative.*
2. Z_1 *is the right zerohood row, naming the finite stream positions whose vanishing would certify the right inspection alternative.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row listing the finite stream requests admitted by the boundary.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational values actually inspected from* S*.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the dyadic radii, endpoint comparisons, and finite margin rows used to keep the two zerohood rows apart.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after* Z_0, Z_1, S, R, D *have been displayed.*
7. W *is the refusal row recording that the accepted packet exports only the displayed finite comparison data and no global LLPO decision rule.*

8. *H stores componentwise* hsame *transport for the two zerohood rows, stream schedule, regular-rational readback, dyadic ledger, real seal, refusal row, continuation row, provenance row, and local name.*

9. *C stores* Cont *replay from finite stream requests through R and D, then through the zerohood rows Z_0, Z_1, and finally into the sealed real-facing refusal row W.*

10. *P records* Pkg *provenance over* $\mathsf{LLPOBoundary}^{\uparrow}$, $\mathsf{IshiharaTrick}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for imported LLPO, LPO, WLPO, trichotomy, quotient real equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LLPOBoundaryUp.LLPOBoundaryTasteGate single carrier alignment`

Theorem 1978.2 (LLPO boundary NameCert obligations)**.** *For every accepted* $\mathsf{LLPOBoundary}^{\uparrow}$ *carrier $B = (Z_0, Z_1, S, R, D, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier exposure for the two zerohood rows, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, refusal row, transport, continuation, package, and local naming rows;*

2. *classifier componentwise transport, requiring every comparison between packets to use the* hsame *rows stored in H coordinate by coordinate;*

3. *zerohood separation, requiring any accepted left/right inspection to cite the displayed dyadic separation and finite margin data in D;*

4. *real-seal handoff, requiring every Real/Completion consumer to reach E only after the schedule, readback, ledger, zerohood rows, and refusal row have been displayed;*

5. *ledger nonescape from imported LLPO, LPO, WLPO, trichotomy, quotient real equality, countable choice, selected limits, and ambient real completeness.*

No NameCert obligation is stored outside these five rows.

Proof. Carrier exposure is projection from Definition 1978.1. The visible data are the two zerohood rows Z_0, Z_1, the schedule S, readback R, tolerance ledger D, terminal seal E, refusal row W, and structural rows H, C, P, N.

Classifier transport is componentwise because the packet offers no aggregate classifier coordinate. A comparison of accepted packets must use the displayed hsame rows in H for each zerohood row, readback, ledger, seal, refusal, replay, provenance, and local name.

Zerohood separation is forced by the tolerance ledger D. A consumer may inspect a left/right zerohood alternative only by naming a dyadic radius, endpoint comparison, and finite margin row already present in the ledger; otherwise W records only refusal of a global binary decision rule.

The real-seal handoff is downstream of E, and the replay path is supplied by C. Since the carrier contains no coordinate for imported LLPO, LPO, WLPO, trichotomy, quotient real equality, choice, selected limits, or ambient completeness, those principles cannot be exported through the local $\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$ row. □

Theorem 1978.3 (LLPO boundary real-comparison refusal)**.** *Every Real/Completion consumer of an accepted* $\mathsf{LLPOBoundary}^{\uparrow}$ *packet receives only the two displayed zerohood rows, finite schedule,*

regular-rational readback, dyadic ledger, terminal real seal, refusal row, componentwise transport, continuation replay, provenance, and local NameCert row. It receives no imported LLPO, no LPO, no WLPO, no trichotomy, no quotient real equality, no countable choice, no selected limit, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Let a Real/Completion consumer accept $B = (Z_0, Z_1, S, R, D, E, W, H, C, P, N)$. By Definition 1978.1, the only real-facing coordinate is the seal E, and that seal is reached after the two zerohood rows, stream schedule, regular-rational readback, dyadic ledger, and refusal row have been displayed.

The consumer's comparison information is therefore finite. It can replay the schedule through R, cite only dyadic tolerance rows from D, and transport the two zerohood rows through H and C.

No omitted semi-classical or completion principle is a coordinate of the packet. Thus factorization through the displayed rows cannot export imported LLPO, LPO, WLPO, trichotomy, quotient real equality, countable choice, selected limits, or ambient real completeness. □

Theorem 1978.4 (LLPO boundary Young-integral nonescape)**.** *Every* $\mathsf{YoungIntegral}^{\uparrow}$ *or controlled Riemann-sum consumer of an accepted* $\mathsf{LLPOBoundary}^{\uparrow}$ *packet reads only the displayed zerohood rows* Z_0, Z_1*, stream schedule* S*, regular-rational readback* R*, dyadic tolerance ledger* D*, Real seal* E*, refusal row* W*, transport row* H*, replay row* C*, provenance row* P*, and local name* N*. It may use those rows as a finite boundary for mesh and tolerance comparisons, but it receives no imported LLPO, LPO, WLPO, trichotomy, quotient real equality, countable choice, selected limit, ambient Real completeness, or hidden branch selector.*

Proof. Start with the real-comparison refusal theorem Definition 1978.3. It confines every Real/Completion consumer to $Z_0, Z_1, S, R, D, E, W, H, C, P, N$.

A controlled Riemann-sum consumer uses only finite comparison data. The stream schedule, regular readback, and dyadic ledger supply the same kind of finite mesh and tolerance rows that the $\mathsf{YoungIntegral}^{\uparrow}$ packet reads through its controlled-window and Riemann-sum route.

The refusal row W remains part of the handoff. Since W records absence of imported semi-classical decision principles, the Young-integral consumer cannot turn zerohood data into a hidden branch choice or selected refinement.

Transport, replay, provenance, and local naming preserve the same finite route. Therefore the controlled-sum consumer receives a zerohood boundary and refusal guard, not a decision principle or completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$.

Constructive story: An $\mathsf{LLPOBoundary}^{\uparrow}$ packet is generated from two zerohood rows, a StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, refusal row, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 1978.1, Definition 1978.2, and Definition 1978.3 seed the finite Real/Completion boundary for binary zerohood inspection.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LLPOBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim imported LLPO, LPO, WLPO, trichotomy, quotient real equality, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier componentwise transport, zerohood separation, real-seal handoff, real-comparison refusal, and ledger nonescape for the same finite packet.

Chapter 1979

A Concrete Naming Certificate for $\mathsf{CompactCauchyCover}^{\uparrow}$

Following Bishop and Bridges on totally bounded metric data and constructive Cauchy approximation, the $\mathsf{CompactCauchyCover}^{\uparrow}$ packet records the finite cover-window layer that can feed a regular-Cauchy real route without importing open-cover compactness. It reads the compact metric carrier of chapter 129, the finite stream-window interface of chapter 126, the rational regular sequence handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion arguments a finite route from displayed cover windows to Cauchy data while leaving selected subcovers, quotient stream equality, choice, and detached $\mathsf{Real}^{\uparrow}$ completeness outside the certificate.

Definition 1979.1 (Compact Cauchy cover carrier)**.** *A* $\mathsf{CompactCauchyCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the carried metric endpoints, metric classifier, finite total-boundedness windows, and visible compact-metric provenance.*
2. *W is the finite cover-window row. For each inspected precision request it lists finitely many displayed metric windows, their centers, and the local membership probes used by the Cauchy-facing route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, storing the dyadic cover radii, shrinkage comparisons, overlap tolerances, and the monotone budget used when windows are refined.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, exposing the finite order in which accepted cover windows are read as stream data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, obtained only after W, D, S have fixed the window, radius, and stream schedule data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the regular-Cauchy readback and never as an independent real-completion coordinate.*
7. *H records componentwise* hsame *transport for the compact metric row, cover windows, dyadic radius ledger, stream schedule, regular readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the finite route*

$$M \longmapsto W \longmapsto D \longmapsto S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactCauchyCover}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, W, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected finite subcovers, quotient stream equality, countable choice, detached $\mathsf{Real}^{\uparrow}$ *completeness, host metric completion, or selected representatives.*

Lean4 ✓ `BEDC.Derived.CompactCauchyCoverUp.TasteGate.CompactCauchyCoverTasteGate single carrier alignment`

Theorem 1979.2 (Compact Cauchy cover NameCert obligations)**.** *For every accepted* $\mathsf{CompactCauchyCover}^{\uparrow}$ *carrier*

$$K = (M, W, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactCauchyCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source row, finite cover-window row, dyadic radius ledger, stream schedule, regular-Cauchy readback, Real seal, and structural rows H, C, P, N;*
2. *cover-window finiteness, requiring each inspected precision request to expose only the finitely listed windows and membership probes stored in W;*
3. *dyadic radius discipline, requiring every refinement, overlap read, and tolerance claim to use the ledger D rather than an ambient compactness principle;*
4. *stream schedule readiness, requiring S to display the finite order in which cover windows become stream data before R is read;*
5. *regular-Cauchy handoff, requiring R to be the only route by which the cover-window data becomes a* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
6. *Real-seal factorization, requiring every downstream read of E to pass through M, W, D, S, R and the structural rows;*
7. *transport, replay, and provenance non-escape, requiring H, C, P, N to preserve and name exactly the displayed rows, with no open-cover compactness, selected subcover, quotient stream equality, choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, host metric completion, or selected representative.*

No compact-Cauchy-cover obligation is stored outside these rows.

Proof. Project the accepted packet $K = (M, W, D, S, R, E, H, C, P, N)$. Carrier admission exposes exactly the compact metric source, the finite cover windows, the dyadic radius ledger, the stream schedule, the regular-Cauchy readback, the Real seal, and the four structural rows.

The cover-window obligation is finite because W is a displayed row of the packet. A precision request may inspect only the centers, windows, and membership probes listed in that row, so it cannot invoke an open-cover compactness theorem or choose a hidden subcover.

The dyadic discipline is the projection to D. Each shrinkage comparison, radius bound, and overlap tolerance used by a refinement is read from the dyadic ledger before any stream or regular-Cauchy coordinate is consumed.

The stream and regular obligations are the projections to S and R. The stream schedule fixes the finite read order of the windows, and the regular readback is downstream of that schedule rather than a detached representative.

Finally, the Real seal and structural rows exhaust the boundary. The seal E is reached only after M, W, D, S, R, while H, C, P, N transport, replay, source, and name this same finite route. Hence the excluded compactness, selection, quotient, choice, completion, and representative coordinates cannot be exported. □

Theorem 1979.3 (Compact Cauchy cover regular handoff)**.** *For every accepted* $\mathsf{CompactCauchyCover}^{\uparrow}$ *carrier*

$$K = (M, W, D, S, R, E, H, C, P, N),$$

every $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by* K *factors through the displayed finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{CompactCauchyCover}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff supplies only cover windows, dyadic radii, stream schedule data, regular readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not supply open-cover compactness, selected subcovers, quotient stream equality, choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, host metric completion, or selected representatives.*

Proof. Begin with a downstream consumer of R or E. The carrier projection shows that R is available only after the compact metric row M, finite cover windows W, dyadic ledger D, and stream schedule S have been displayed.

The finite route is therefore forced. The compact metric row supplies the metric and total-boundedness surface, W records the windows inspected at the requested precision, D records the radii and tolerance comparisons, and S orders those windows as stream data.

The regular readback row R packages that stream data as the $\mathsf{RegSeqRat}^{\uparrow}$ handoff. The terminal seal E is downstream of R, so a Real-facing consumer receives a sealed Cauchy readback rather than an independent completeness theorem.

Componentwise transport and replay through H and C preserve the same route, while P and N name only the displayed packet. Since the carrier inventory has no open-cover, selected-subcover, quotient-stream, choice, detached-completeness, host-completion, or representative coordinate, none can be consumed downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchyCover}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCauchyCover}^{\uparrow}$ packet is generated from a CompactMetric source row, finite cover windows, a DyadicRatCore radius ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCauchyCover}^{\uparrow}}$, seed)

Scope closed: Definition 1979.1, Definition 1979.2, and Definition 1979.3 seed the finite cover-window route from compact metric data to regular-Cauchy Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCauchyCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient stream equality, choice, detached $\mathsf{Real}^{\uparrow}$ completeness, host metric completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-window finiteness, dyadic radius discipline, stream schedule readiness, regular-Cauchy handoff, Real-seal factorization, transport/replay/provenance coverage, and non-escape for BEDC.Derived.CompactCauchyCoverUp.

The $\mathsf{Cauchy}^{\uparrow}$ hub is read through chapter 126, chapter 301, chapter 291, and chapter 38 before Real-facing completion consumers inspect the Cauchy route.

Chapter 1980

A Concrete Naming Certificate for $\mathsf{CauchySequence}^{\uparrow}$

Bishop–Bridges constructive analysis treats Cauchy sequences as tail-stable observable data before any completed real, quotient representative, or chosen limit is available. The $\mathsf{CauchySequence}^{\uparrow}$ packet records that ordinary sequence layer as a finite NameCert surface between the explicit-modulus route of chapter 1705 and the weak finite-tail route of chapter 1887. It reads finite observations through the $\mathsf{StreamName}^{\uparrow}$ interface of chapter 126, optional threshold data through the $\mathsf{CauchyModulus}^{\uparrow}$ surface of chapter 292, dyadic tolerances through chapter 291, regular rational readback through chapter 301, and its terminal boundary through chapter 38. The packet is the ordinary Cauchy sequence carrier used by Real and Completion consumers; it is not quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Definition 1980.1 (Cauchy sequence carrier)**.** *A* $\mathsf{CauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, T, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence row, naming the observable sequence surface whose values are inspected only through displayed finite requests.*
2. *T is the tail-comparison ledger, recording the finite inspected pairs of tail indices and the comparison rows witnessing Cauchy behavior at the displayed tolerances.*
3. *M is an optional* $\mathsf{CauchyModulus}^{\uparrow}$ *row. When present, it gives an explicit threshold schedule; when absent, the packet remains on the ordinary tail-comparison surface rather than the effective-modulus surface.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the inspected source windows used by T and by any downstream readback.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack rows, comparison budgets, and tolerance conversions used by the tail comparisons.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, available only after the source, tail-comparison, optional-modulus, window, and dyadic rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, read only downstream of S, T, M, W, D, R and never as a selected limit or ambient completeness witness.*

8. *H records componentwise* hsame *transport for the source, tail ledger, optional modulus, finite windows, dyadic ledger, regular handoff, Real boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each Real or Completion request through $S, T, M, W, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{WeaklyCauchySequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, M, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1980.2 (Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequence}^{\uparrow}$ *carrier*

$$X = (S, T, M, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source sequence row, tail-comparison ledger, optional modulus, StreamName windows, DyadicRatCore tolerances, RegSeqRat handoff, Real boundary, transport, replay, provenance, and local naming;*
2. *window exactness, requiring every inspected sequence value used by T or R to be read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row W;*
3. *tail-comparison exactness, requiring T to store the finite pairwise tail comparisons that make the ordinary Cauchy evidence visible;*
4. *optional-modulus compatibility, requiring any displayed M row to refine the same tail comparisons without replacing the ordinary packet by the explicit-modulus packet;*
5. *dyadic tolerance accounting, requiring every comparison in T and every readback exported by R to cite the dyadic radii, slack rows, and conversions stored in D;*
6. *regular handoff and Real boundary discipline, requiring R and then E to remain downstream of S, T, M, W, D;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N;*
8. *non-escape for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No ordinary Cauchy-sequence obligation is stored outside these rows.

Proof. Project the carrier of Definition 1980.1. The projection exposes exactly one source sequence row S, one finite tail-comparison ledger T, one optional modulus row M, one finite window row W, one dyadic tolerance ledger D, one regular handoff row R, one terminal Real boundary row E, and the structural rows H, C, P, N.

Window and tail-comparison exactness are forced by the order of the rows. Every inspected sequence value is first read through the finite $\mathsf{StreamName}^{\uparrow}$ window row W, and every ordinary Cauchy assertion visible to the packet is then one of the displayed finite comparisons stored in T.

The optional-modulus clause is conservative. If M is present, it cites the same source, window, and dyadic rows as T; if it is absent, the packet still retains the ordinary tail-comparison ledger. Thus the carrier sits between the explicit-modulus chapter and the weak finite-tail chapter without collapsing into either one.

Dyadic accounting is supplied by D. Since the packet has no other tolerance coordinate, all comparison budgets, slack rows, and conversions used by T or by the regular readback R must pass through that ledger.

Finally, the rows R, E, H, C, P, N export, seal, transport, replay, source, and name the same displayed data. None of those rows adds quotient equality, a selected endpoint, a choice principle, trichotomy, host real equality, completeness of $\mathsf{Real}^{\uparrow}$, Lean evidence, or bridge data, so the listed obligations exhaust the local NameCert surface. □

Theorem 1980.3 (Cauchy sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchySequence}^{\uparrow}$ *carrier* $X = (S, T, M, W, D, R, E, H, C, P, N)$ *factors through the finite route*

$$S \longmapsto W \longmapsto T, M \longmapsto D \longmapsto R \longmapsto H, C, P, N.$$

Consequently a regular rational consumer receives the displayed source, finite StreamName windows, ordinary tail comparisons, any compatible modulus row, dyadic tolerance accounting, transport, replay, provenance, and local naming before it can inspect the handoff.

Proof. Begin with a regular rational consumer of X. By Definition 1980.1, the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate is the handoff row R, and R is downstream of the source, window, tail-comparison, optional-modulus, and dyadic rows.

The source observations are first exposed through the $\mathsf{StreamName}^{\uparrow}$ row W. The ordinary Cauchy information is then the tail-comparison ledger T, while an available modulus row M is read only as a compatible refinement of those displayed comparisons.

The tolerance work is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D. Only after the relevant dyadic radii, slack rows, comparison budgets, and conversions have been displayed can the regular rational handoff R be exported.

The structural rows H, C, P, N preserve and name the same route. Hence every regular rational read factors through the displayed ordinary Cauchy sequence packet and has no bypass through quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient completeness, Lean verification, or bridge checking. □

Theorem 1980.4 (Cauchy sequence L10 sibling lattice)**.** *Any ordinary Cauchy sequence consumer using L10 completion data from an accepted* $\mathsf{CauchySequence}^{\uparrow}$ *carrier*

$$X = (S, T, M, W, D, R, E, H, C, P, N)$$

must expose the sibling route through chapter 126, chapter 292, chapter 291, chapter 301, and chapter 38 before it can read $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing data. The route is the finite lattice* $S, W, T, M, D, R, E, H, C, P, N$*, not a kernel-only completion claim, selected limit, quotient stream equality, or ambient completeness principle.*

Proof. Project the carrier rows of Definition 1980.1. The source row S is inspected only through the finite StreamName windows W, while ordinary tail comparisons are stored in T and any explicit threshold schedule is stored in the optional CauchyModulus row M.

The tolerance data are then forced through the DyadicRatCore ledger D. Thus every comparison that later reaches a regular rational or Real-facing read has already exposed the stream, optional modulus, and dyadic sibling routes.

Apply the RegSeqRat handoff theorem, Definition 1980.3. It places R downstream of S, W, T, M, D and keeps the terminal Real boundary E downstream of the same route.

Finally H, C, P, N transport, replay, source, and name this lattice. Since no row stores a selected limit, quotient stream equality, host real equality, countable choice, trichotomy, or ambient completeness, an L10 completion consumer can read only the displayed sibling route. □

Theorem 1980.5 (Cauchy sequence root stream comparison exhaustion)**.** *Any ordinary Cauchy comparison exported by an accepted* $\mathsf{CauchySequence}^{\uparrow}$ *carrier*

$$X = (S, T, M, W, D, R, E, H, C, P, N)$$

is exhausted by the source sequence row S, finite $\mathsf{StreamName}^{\uparrow}$ *windows W, tail-comparison ledger T, optional modulus row M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal boundary E, and structural rows H, C, P, N. No comparison exported by X can bypass this route through quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the carrier of Definition 1980.1. The root source is S, and every observed sequence value used by an ordinary comparison must first be read through the finite StreamName windows W.

Use the NameCert obligations of Definition 1980.2. They force tail comparisons through T, optional threshold schedules through M when present, and every tolerance budget through the DyadicRatCore ledger D.

Apply the RegSeqRat handoff of Definition 1980.3. Any regular rational or Real-facing comparison must pass through R and then the terminal seal boundary E, while H, C, P, N transport, replay, source, and name the same route.

Since the carrier has no quotient stream equality, selected limit, countable-choice schedule, trichotomy decision, host real equality, ambient completeness, Lean evidence, or bridge-checking row, the displayed rows exhaust every ordinary root-stream comparison exported by X. □

Falsifiable prediction: Every $\mathsf{CauchySequence}^{\uparrow}$ regular or Real-facing read factors through a source sequence row, finite StreamName windows, tail-comparison ledger, optional CauchyModulus row, DyadicRatCore tolerance ledger, RegSeqRat handoff, RealUp boundary seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, cauchymodulus, dyadicratcore, regseqrat, real, effective-cauchy-sequence, weakly-cauchy-sequence: $\mathsf{CauchySequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ordinary tail comparisons, optional explicit-modulus compatibility, finite stream windows, dyadic tolerance accounting, regular rational handoff, terminal Real boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequence}^{\uparrow}$ packet is generated from a source sequence row, finite tail-comparison ledger, optional CauchyModulus row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, terminal RealUp boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySequence}^{\uparrow}}$, seed)

Scope closed: Definition 1980.1, Definition 1980.2, and Definition 1980.3 seed the ordinary Cauchy sequence route between explicit-modulus and weak finite-tail sequence surfaces.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, tail-comparison exactness, optional-modulus compatibility, dyadic tolerance accounting, regular handoff, Real boundary discipline, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceUp.

Chapter 1981

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$

Following Mac Lane and Eilenberg–Moore on algebras for monads, and Bishop–Bridges on constructive Cauchy completion, the $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ packet records the finite BEDC surface by which the L10 completion monad exposes an algebra action without turning completed points into quotient representatives. It consumes the completion-monad route of chapter 378, the complete-metric boundary of chapter 128, the universal-property route of chapter 397, the real completion seal of chapter 360, and the terminal real-name seal of chapter 38. The chapter names an Eilenberg–Moore algebra surface for displayed completion routes only; it does not assert quotient completion equality, selected representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1981.1 (CauchyCompletionAlgebra carrier)**.** *A* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, Q, U, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *route carrying the unit, bind, and multiplication-facing data of the L10 completion monad.*
2. *Q is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary on which the algebra action is displayed.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *route that constrains algebra-facing maps through the displayed completion surface.*
4. *L is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *seal reached after the monad route, complete-metric boundary, and universal-property route have been displayed.*
5. *H records componentwise* hsame *transport for the monad route, complete-metric boundary, universal-property route, real completion seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the algebra route*

$$M, Q \longmapsto U \longmapsto L.$$

7. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through M, Q, U, L, H, C, P, N. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 1981.2 (CauchyCompletionAlgebra NameCert obligations)**.** *For every accepted* CauchyCompletionAlgebr *carrier*

$$A = (M, Q, U, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the monad route, complete-metric boundary, universal-property route, real completion seal, transport, replay, provenance, and local naming row;*
2. *algebra-action exposure, requiring every algebra-facing completion read to enter through M and Q before the universal-property route or real completion seal can be consumed;*
3. *universal-property discipline, requiring U to mediate every public map out of the displayed completion surface;*
4. *real-completion sealing, requiring L to remain downstream of M, Q, U and never to stand as a selected quotient representative;*
5. *classifier stability, requiring transported packets with componentwise matching rows to replay the same algebra route through H and C;*
6. *provenance and non-escape, requiring P, N to source and name exactly the displayed rows, with no quotient completion equality, selected representative, host equality of completed objects, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge credential.*

No $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the accepted packet $A = (M, Q, U, L, H, C, P, N)$. Carrier admission exposes exactly the monad route, complete-metric boundary, universal-property route, real completion seal, and structural rows.

The algebra-action obligation is the projection to M and Q. A completion-facing consumer does not receive an ambient completed carrier; it first reads the displayed monad route and then the complete-metric boundary on which the action is finite BEDC data.

The universal-property obligation is the projection to U. It constrains every public map out of the displayed completion surface to pass through the same finite route rather than through a host completion object.

The real-completion seal is the projection to L. It is downstream of M, Q, U, so it cannot be consumed as a selected quotient representative or as host equality of completed objects.

Finally, H, C, P, N exhaust transport, replay, provenance, and naming. These rows preserve the displayed coordinates and add no quotient completion equality, selected representative, host equality, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge credential, so the listed rows are exactly the local NameCert surface. □

Theorem 1981.3 (CauchyCompletionAlgebra complete-metric handoff)**.** *For every accepted* CauchyCompletionAlg *carrier*

$$A = (M, Q, U, L, H, C, P, N),$$

every $\mathsf{CompleteMetric}^{\uparrow}$ *consumer read exported by* A *factors through the finite route*

$$\mathsf{CauchyCompletionMonad}^{\uparrow} \longrightarrow \mathsf{CompleteMetric}^{\uparrow} \longrightarrow \mathsf{CompletionUniversalProperty}^{\uparrow} \longrightarrow \mathsf{RealCauchyCompletion}^{\uparrow}.$$

The handoff preserves the displayed algebra route and exports no quotient completion equality, selected representative, host equality of completed objects, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge credential.

Proof. Begin with a complete-metric consumer and project the carrier. The only complete-metric row is Q, and it is downstream of the monad route M.

The public map route is forced through U. The universal-property row keeps the complete-metric boundary attached to the displayed completion algebra rather than to an ambient completed carrier.

The consumer then reaches the completion seal L. This seal exposes only the displayed real Cauchy completion route fixed by M, Q, U.

Transport by H and replay by C preserve this route componentwise, and P, N only source and name the same packet. Since the carrier has no further coordinate, the handoff cannot produce quotient equality, representatives, host equality, choice, ambient completeness, Lean verification, or bridge data. □

Theorem 1981.4 (CauchyCompletionAlgebra public handoff)**.** *For every accepted* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *carrier*

$$A = (M, Q, U, L, H, C, P, N),$$

any public completion-algebra consumer receives the finite certificate route

$$\mathsf{CauchyCompletionMonad}^{\uparrow} \longrightarrow \mathsf{CompleteMetric}^{\uparrow} \longrightarrow \mathsf{CompletionUniversalProperty}^{\uparrow} \longrightarrow \mathsf{RealCauchyCompletion}^{\uparrow}$$

together with the displayed hsame*,* Cont*,* Pkg*, and* NameCert *rows. The handoff exports no quotient completion equality, selected limit, host equality of completed objects, ambient completeness theorem, Lean verification, or bridge credential.*

Proof. Begin with a public completion-algebra consumer and project the carrier. The visible algebra action is sourced by M and is displayed over the complete-metric boundary Q.

The public mapping discipline is then forced through U. A consumer can use the universal-property route only as part of this same packet, so no separate host completion object or selected quotient representative is introduced.

The terminal completion read is the seal L. Since L is downstream of M, Q, U, it records the displayed real completion route rather than ambient completeness or equality in a host model.

The structural rows H, C, P, N preserve and name this route. They add transport, replay, provenance, and local certification, but no extra coordinate from which quotient equality, selected limits, host equality, Lean verification, or bridge data could be exported. □

Theorem 1981.5 (CauchyCompletionAlgebra Real-seal nonescape)**.** *For every accepted* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *carrier*

$$A = (M, Q, U, L, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* L *receives only the displayed algebra route*

$$M, Q, U, L, H, C, P, N.$$

The seal is not quotient completion equality, not a selected representative, not host equality of completed objects, not countable choice, not ambient $\mathsf{Real}^{\uparrow}$ *completeness, not Lean verification, and not a bridge credential.*

Proof. Start from the terminal seal L. In the carrier, L is downstream of the displayed monad route, complete-metric boundary, and universal-property route rather than an independent completed-object coordinate.

Trace the route backward. A Real-facing read reaches L only after U has been displayed, and U is available to the algebra action only through M, Q.

The structural rows do not add a second seal source. The transport row H is componentwise, the replay row C displays the same route order, and the provenance and local certificate rows P, N name this packet without importing an ambient completion principle.

Thus every Real-facing consumer is confined to M, Q, U, L, H, C, P, N. The excluded quotient, representative, host-equality, countable-choice, ambient-completeness, Lean-verification, and bridge readings are outside the carrier inventory. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ packet is generated from the CauchyCompletionMonad route, CompleteMetric boundary, CompletionUniversalProperty route, RealCauchyCompletion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 1981.1, Definition 1981.2, Definition 1981.3, Definition 1981.4, and Definition 1981.5 seed the finite Eilenberg–Moore algebra surface for the L10 completion monad over displayed BEDC rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, algebra-action exposure, complete-metric boundary discipline, universal-property discipline, RealCauchyCompletion sealing, classifier stability, public handoff, Real-seal non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionAlgebraUp.

Cauchy sequence L10 dependency route. The $\mathsf{CauchySequence}^{\uparrow}$ carrier consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ interface of chapter 126, the threshold row of chapter 292, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. This route records the horizontal L10 dependency lattice for ordinary Cauchy sequence consumers without adding a quotient stream, selected limit, or ambient completeness coordinate. The $\mathsf{Cauchy}^{\uparrow}$ hub exposes its sequence-facing child through the same L10 naming lattice used by Real and Completion consumers. A Cauchy sequence read is not a detached stream fact: it is a finite route through the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerances of chapter 291, and the terminal Real seal of chapter 38.

Theorem 1981.6 (Cauchy sequence RegSeqRat window route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *sequence read used by a* $\mathsf{Real}^{\uparrow}$ *or Completion consumer factors through a displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{Real}^{\uparrow}$ *seal before it can be consumed as a Real-facing or completion-facing boundary. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem.*

Proof. Read the sequence child as the visible Cauchy-facing row of the hub. A consumer first inspects the finite StreamName window, then reads the same data as a regular rational row, and then checks the requested tolerance through the dyadic ledger.

Only after those displayed rows are present can the terminal Real seal be consumed. Since the route contains no quotient coordinate, selected endpoint, choice schedule, host equality row, or ambient completeness row, the consumer receives exactly the finite L10 window route. □

Definition 1981.7 (Cauchy root stream carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *root stream carrier is a finite* BHist *packet*

$$Q = (S, T, M, W, H, C, P, N)$$

with the following rows:

1. *S is the certified finite observation stream. It records only the visible request windows that a Cauchy consumer may inspect.*
2. *T is the tail-threshold row. It assigns each displayed precision request a later visible window of S.*
3. *M is the modulus row. It records the finite evidence that two observations past the displayed threshold meet the requested tolerance.*
4. *W is the witness ledger. It stores the accepted comparisons between observed entries, not a quotient class, selected limit, or ambient completeness theorem.*
5. *H is componentwise* hsame *transport for the stream, threshold, modulus, witness, replay, provenance, and naming rows.*
6. *C is displayed* Cont *replay from an admitted Cauchy request through S, T, M, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The classifier compares accepted root stream packets only through the displayed tuple S, T, M, W, H, C, P, N.

Theorem 1981.8 (Cauchy root tail stability)**.** *Let $Q = (S, T, M, W, H, C, P, N)$ be an accepted* $\mathsf{Cauchy}^{\uparrow}$ *root stream carrier. Every* hsame *transport of Q and every displayed* Cont *replay of a Cauchy request preserves the tail-threshold row T, the modulus row M, and the witness ledger W as componentwise rows over the same finite observation stream S. Consequently a tail read transported or replayed from Q remains a Cauchy root stream read, and it does not acquire a quotient stream equality, a selected limit, a choice schedule, or a host completeness assertion.*

Proof. Project the carrier of Definition 1981.7. The only rows available to a Cauchy request are the displayed stream window, tail threshold, modulus, witness ledger, transport, replay, provenance, and naming rows.

The hsame row H is componentwise by definition of the carrier surface. It transports S, T, M, W, C, P, N as rows of the same packet, so it cannot introduce a second stream, an external representative, or an ambient equality oracle.

The Cont row C replays an admitted request through the displayed packet. Since the replay reads T, M, W after S and before P, N, the transported request still witnesses the same finite tail-stability surface. □

Theorem 1981.9 (Cauchy root NameCert obligation surface)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *root stream carrier $Q = (S, T, M, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface consists exactly of the following rows:*

1. *carrier admission for the finite observation stream S;*
2. *classifier admission for the tail-threshold row T and modulus row M;*
3. *stability under componentwise* hsame *transport H and displayed* Cont *replay C;*

4. comparison fidelity for the witness ledger W against the displayed tolerance requests;

5. provenance and local naming through P and N.

No obligation row certifies quotient streams, selected limits, countable choice, host real equality, or an ambient completion theorem.

Proof. Read the eight rows of Definition 1981.7. Carrier admission is the row S, while classifier admission is carried by the threshold and modulus rows T, M.

Stability is supplied by the structural rows H and C. These rows keep the request inside the displayed packet and force every transported or replayed tail read to pass through the same finite stream surface.

The witness ledger W supplies the comparison data for the displayed tolerance requests, and P, N source and name the packet. Since no remaining coordinate exists, the listed rows exhaust the local NameCert surface. □

Theorem 1981.10 (Cauchy root completion consumer route)**.** *Every completion-facing consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *root stream carrier first reads the finite stream row of chapter 126, then the regular rational readback route of chapter 301, then the terminal Real seal of chapter 38. Metric and uniform completion consumers may use the same displayed route only after those rows have exposed the tail threshold, modulus, and witness ledger of Q.*

Proof. Start with a completion-facing read from Q. The carrier projection exposes the finite stream row before any threshold, modulus, witness, or structural row can be consumed.

The stream row is then read through the regular rational route. This is the point at which completion-facing consumers can inspect the finite rational data attached to the Cauchy window, while the Real seal remains downstream.

Finally the terminal Real seal is consumed after the tail-threshold, modulus, and witness rows are visible. The route therefore reaches completion consumers through StreamName, RegSeqRat, and RealUp data, not through quotient completions or selected representatives. □

Theorem 1981.11 (Cauchy sequence RegSeqRat stream anchor)**.** *Every Cauchy sequence root read that feeds an L10* $\mathsf{Real}^{\uparrow}$ *or completion consumer factors through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Project the sequence root carrier through the hub child route. A consumer first receives finite window requests, not an infinite sequence object or a selected representative.

The regular rational readback is then interpreted through the displayed dyadic tolerance ledger. This places every tail request inside the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before any Real-facing or completion-facing seal can be consumed.

The structural rows preserve the same route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row source and name the finite readback path, so no quotient stream equality, selected limit, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported. □

Definition 1981.12 (Sequence Cauchy root observation carrier)**.** *For the root* $\mathsf{Cauchy}^{\uparrow}$ *sequence route, a finite observation carrier is a* BHist *packet*

$$O = (S, R, D, H, C, P, N)$$

where S is the $\mathsf{StreamName}^{\uparrow}$ *finite observation row of chapter 126, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291, and H, C, P, N are respectively componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The carrier has no quotient sequence coordinate, host-sequence equality coordinate, selected limit, or ambient completion proof.*

Theorem 1981.13 (Sequence Cauchy root stream obligation)**.** *Every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *sequence route factors through the finite observation carrier of Definition 1981.12. Its local obligation surface consists of StreamName window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance accounting, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Read the root stream carrier of Definition 1981.7. Its stream row supplies the finite observation windows and its witness ledger supplies only displayed comparisons.

Route those windows through chapter 126, then through the regular rational readback of chapter 301, and finally through the dyadic tolerance ledger of chapter 291. These three rows are exactly S, R, D in the observation carrier.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient or host-sequence equality coordinate, every accepted Cauchy read remains inside the displayed NameCert surface. □

Theorem 1981.14 (Sequence Cauchy hsame-stable tail classifier)**.** *Let $O = (S, R, D, H, C, P, N)$ be an accepted sequence Cauchy root observation carrier. Any tail classifier read transported by H preserves the same StreamName window row S, the same RegSeqRat readback row R, and the same DyadicRatCore tolerance ledger D. Consequently the transported tail classifier remains a root* $\mathsf{Cauchy}^{\uparrow}$ *read and cannot introduce quotient sequence equality, host-sequence equality, selected limits, or ambient completion.*

Proof. Project the carrier O. The tail classifier has access only to S, R, D, H, C, P, N, and H is declared componentwise over exactly those rows.

Transporting a tail read by H therefore replaces no observation row by an external sequence and adds no equality oracle. The StreamName, RegSeqRat, and DyadicRatCore rows stay visible before the classifier verdict is read.

The transported classifier is then repacked with the same provenance and local naming rows. This is the root Cauchy tail surface, not a quotient or host-sequence equality claim. □

Theorem 1981.15 (Sequence Cauchy Cont-stable replay ledger)**.** *For every accepted sequence Cauchy root observation carrier $O = (S, R, D, H, C, P, N)$, each displayed* Cont *replay of a Cauchy request must read S, then R, then D, and then the structural rows H, P, N. The replay ledger is therefore stable under finite-window refinement and remains a finite NameCert ledger for StreamName, RegSeqRat, and DyadicRatCore data.*

Proof. Start with a displayed replay row C. By the carrier surface, C is a replay through the finite observation carrier and not through an ambient sequence object.

The replay order exposes the StreamName window before the regular rational readback and the dyadic tolerance ledger. A finite-window refinement may restrict S, but the refined route still factors through the corresponding R and D rows.

Finally the replay is named by P, N and transported by H. The ledger remains finite because every replayed request is read through displayed windows and tolerance rows rather than through quotient or host-sequence equality. □

Theorem 1981.16 (Cauchy Real completion lattice repair)**.** *Every completion-facing* Cauchy$^{\uparrow}$ *read that reaches the Real and Completion lattice factors through finite* StreamName$^{\uparrow}$ *windows from chapter 126,* RegSeqRat$^{\uparrow}$ *regular rational readback from chapter 301,* DyadicRatCore$^{\uparrow}$ *tolerance rows from chapter 291, the terminal* Real$^{\uparrow}$ *seal of chapter 38, and the* CauchyFilterCompletion$^{\uparrow}$ *consumer rows of chapter 1569 before any completion-facing conclusion is exposed. The route carries the same finite* BHist*, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *rows as the root sequence packet, and it exports no quotient streams, selected limits, countable choice, host equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the existing Cauchy$^{\uparrow}$ root sequence surface. Its public rows expose finite sequence observations and structural BHist, hsame, Cont, Pkg, and NameCert data before any downstream consumer can read a completion-facing claim.

Follow the named sibling route. The observation is first read as a StreamName$^{\uparrow}$ finite window, then as a RegSeqRat$^{\uparrow}$ regular rational readback, then through DyadicRatCore$^{\uparrow}$ tolerance rows, and finally through the terminal Real$^{\uparrow}$ seal.

Only after those rows are visible may the consumer enter the CauchyFilterCompletion$^{\uparrow}$ surface. Since no row in the route stores quotient stream equality, selected limits, countable choice, host equality, or ambient Real completeness, the completion-facing conclusion is exactly the finite lattice read described above. □

The Cauchy$^{\uparrow}$ rows are consumed downstream by chapter 126, chapter 301, and chapter 38. The route opens only finite stream windows, regular Cauchy tail budgets, and terminal Real$^{\uparrow}$ seal rows, so a Cauchy packet can feed the L10 real-completion argument without adding content to the hub file.

Theorem 1981.17 (Cauchy hub Bishop completion forward route)**.** *The regular-source material routed by the* Cauchy$^{\uparrow}$ *hub is consumed by chapter 1639 through the displayed* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The forward route exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness principle.*

Proof. Read the Cauchy hub through its regular-source children. Those children expose dyadic tolerance arithmetic, finite stream windows, regular rational readback, and the terminal real seal as DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ rows.

The Bishop regular Cauchy completion chapter consumes exactly that L10-facing route. Its completion read is mediated by the same finite BHist carrier discipline, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Since the hub route contains no detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness coordinate, the forward edge to the Bishop completion packet is precisely the displayed finite dependency route. □

The Cauchy$^{\uparrow}$ dossier also routes finite Cauchy source data into the L10 completion interface: its displayed request windows and modulus rows feed the StreamName$^{\uparrow}$ surface of chapter 126, its rational readback is consumed by chapter 301, and its terminal seal-facing row reaches chapter 38 through the finite section packet of chapter 6518. Thus Cauchy$^{\uparrow}$ supplies the finite Cauchy source row consumed by the L10 completion route without asserting quotient stream equality, selected limits, countable choice, host real equality, or ambient Real$^{\uparrow}$ completeness. The Cauchy$^{\uparrow}$ hub supplies a forward L10 route to chapter 1569: finite sequence evidence is first read through StreamName$^{\uparrow}$ windows, then through RegSeqRat$^{\uparrow}$ regular-rational readback, then through DyadicRatCore$^{\uparrow}$ tolerance rows, and finally through the terminal Real$^{\uparrow}$ seal before the CauchyFilterCompletion$^{\uparrow}$ packet

consumes completion data. This route records dependency flow only; it exports no quotient stream equality, selected filter limit, countable-choice schedule, host real equality, trichotomy, or ambient completeness coordinate.

Theorem 1981.18 (Cauchy filter-completion dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *read used by completion-facing consumers is routed through the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *packet of chapter 1569 only after the* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows are displayed. The route begins with finite Cauchy sequence evidence, opens the Cauchy-filter basis, records the inspected stream windows, reads the regular rational data, accounts for dyadic tolerances, and then seals the terminal Real row before completion data is consumed. It exports no quotient stream equality, quotient filter, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient completeness theorem.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its sequence and regular-source children. Those children display the finite stream windows and regular rational readback before any completion-facing row is available.

The $\mathsf{CauchyFilter}^{\uparrow}$ basis is then exposed as a filter route over the same finite evidence. The $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows and the terminal $\mathsf{Real}^{\uparrow}$ seal record the numerical part of the route before the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ packet can consume it.

The structural rows preserve the same finite handoff through BHist, hsame, Cont, Pkg, and NameCert data. Since the hub provides no quotient stream equality, quotient filter, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient completeness coordinate, the completion dependency is exactly the displayed route. □

Theorem 1981.19 (Cauchy hub Real-completion lattice route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *route consumed by a Real-completion chapter factors through the finite stream-window surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38 before completion consumers read it. The hub exports only the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows attached to that lattice; it exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the hub input order for $\mathsf{Cauchy}^{\uparrow}$. The root sequence and stream obligation files expose finite observation windows and the local NameCert route before any completion-facing sibling can consume the packet.

Follow the Real-completion lattice through the named sibling chapters. A consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ window, then the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational row, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and finally the $\mathsf{Real}^{\uparrow}$ seal that is visible to completion packets.

The carrier rows available at the hub are exactly the displayed BHist route data, componentwise hsame transport, Cont replay, Pkg provenance, and local naming rows. Since none of those rows stores quotient streams, selected limits, choice schedules, host real equality, or ambient Real completeness, every completion-facing read factors through the finite lattice stated above. □

Theorem 1981.20 (Cauchy RegSeqRat completion forward route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *read consumed by a completion-facing chapter must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the local* $\mathsf{Cauchy}^{\uparrow}$ *route before completion data is exposed. The route carries only the displayed* BHist*, componentwise* hsame*,* Cont*,* Pkg*, and local* NameCert *rows, and it exports no quotient stream, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the Cauchy root sequence and stream routes. They expose finite observations as named windows before any completion-facing consumer can inspect the packet.

Apply the existing Cauchy sibling routes through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The read is first a finite stream window, then a regular rational tail budget, and then a real-name seal attached to the same local Cauchy packet.

Finally compose with the Real-completion lattice route. Since the displayed rows contain only BHist, hsame, Cont, Pkg, and local NameCert data, the completion-facing consumer receives the named forward route without a quotient stream, selected limit, choice schedule, host equality, or ambient Real-completeness coordinate. □

The $\mathsf{Cauchy}^{\uparrow}$ carrier is consumed by the Real and regular-sequence completion lattice through the substantive routes of chapter 38, chapter 301, chapter 126, and chapter 1569. A completion-facing read therefore reaches Real seals, regular rational readback, finite stream windows, and Cauchy-filter completion consumers as displayed BHist, hsame, Cont, Pkg, and NameCert rows, not as quotient stream equality, a selected limit, countable choice, host real equality, or ambient completeness.

Theorem 1981.21 (Cauchy forward Real-completion route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer that reaches a Real-completion reading must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *route of chapter 126, the regular-rational completion rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *completion and seal rows of chapter 38. The route may use the Cauchy kernel predicate only as a displayed finite observation condition; it cannot treat* $\mathsf{Cauchy}^{\uparrow}$ *as an isolated kernel predicate that already supplies a selected limit, quotient stream equality, host real equality, or ambient completeness.*

Proof. Start with a completion-facing consumer of the $\mathsf{Cauchy}^{\uparrow}$ hub. The hub exposes finite sequence and stream observation rows before any completion packet can read the predicate.

Follow the forward route through the named siblings. The consumer first reads $\mathsf{StreamName}^{\uparrow}$ windows, then the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational completion handoff, and finally the $\mathsf{Real}^{\uparrow}$ completion and seal rows.

The structural rows preserve only that displayed route. Since no coordinate in the hub supplies selected limits, quotient stream equality, host real equality, or ambient completeness, the consumer cannot bypass $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ by reading Cauchy as an isolated kernel predicate. □

1981.1 StreamName Readback Route

The $\mathsf{Cauchy}^{\uparrow}$ completion hub has a substantive L10 route through the finite-window sibling lattice rather than only through the kernel sequence carrier. A completion consumer that reads a Cauchy packet must first expose $\mathsf{StreamName}^{\uparrow}$ windows as in chapter 126, replay the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback, check the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and then seal the resulting endpoint through $\mathsf{Real}^{\uparrow}$.

Theorem 1981.22 (Cauchy stream-name sibling route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion consumer that claims a Real-facing endpoint factors its read through* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The route exposes the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *coordinates used by the Cauchy hub, and it has no direct access to selected limits, quotient completion classes, ambient Real completeness, or countable choice.*

Proof. Read the Cauchy hub through its existing sequence and completion children. The only windowed observation route begins with the $\mathsf{StreamName}^{\uparrow}$ finite-window sibling and then replays the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback against the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows.

The endpoint seal is therefore downstream of the finite readback route. Once $\mathsf{Real}^{\uparrow}$ receives the sealed row, the structural coordinates $\mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ preserve that same finite route and provide no coordinate for selected limits, quotient classes, ambient completeness, or countable choice. □

The $\mathsf{Cauchy}^{\uparrow}$ hub supplies the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows consumed by chapter 1536 through its epigraph Real-export route. The forward edge reads only synchronized stream windows, regular rational tail budgets, terminal $\mathsf{Real}^{\uparrow}$ seal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows; it does not add a quotient stream equality, selected limit, countable choice row, host lower-semicontinuity predicate, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Chapter 1982

A Concrete Naming Certificate for $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completions, and Kelley and Bourbaki on filter bases, the $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ packet names the finite bridge by which located Cauchy-filter requests are regularized into modulus rows before a completion-facing consumer reads a real seal. It sits between the located-filter source of chapter 1661, the finite base route of chapter 1915, the common-refinement operation of chapter 1958, and the ideal-completion boundary of chapter 1883. The downstream L10 route reads finite windows through chapter 126, regular rational rows through chapter 301, dyadic tolerances through chapter 291, and terminal seals through chapter 38. The packet is a finite transfer surface from filter-base evidence to regular Cauchy modulus data; it is not a quotient filter, selected limit, countable-choice schedule, host equality comparison, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification row, or bridge certificate.

Definition 1982.1 (Cauchy filter modulus transfer carrier)**.** *A* $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, B, L, J, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the Cauchy filter source row. It records the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *request surface whose precision requests are to be regularized.*
2. *B is the* $\mathsf{CauchyFilterBase}^{\uparrow}$ *basis row. It exposes finite base indices, entourage requests, directed-refinement reads, and Cauchy readiness rows used by the source filter.*
3. *L is the located-tail row. It records the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *evidence that every requested tail can be inspected by a located finite base request before any regular modulus row is exported.*
4. *J is the common-refinement row. It reads the* $\mathsf{CauchyFilterMeet}^{\uparrow}$ *operation and names, for each requested tolerance and each located tail request, a displayed refinement accepted by both the filter source and the basis route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row reached only after F, B, L, J have been displayed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the accepted window route as a regular rational sequence row.*

7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested dyadic scale, the smaller scale selected by the located-tail row, and the budget comparison used to certify regular Cauchy control.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary. It is visible only after the filter source, basis route, located-tail row, common-refinement row, window row, regular readback row, and dyadic ledger have been replayed.*

9. *H records componentwise* hsame *transport for the source filter, basis row, located-tail row, common-refinement row, window, regular readback, dyadic tolerance, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a filter request through F, B, L, J, S, R, D, E in that order.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBase}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterMeet}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterModulusTransfer}^{\uparrow}}$ *row.*

The classifier compares accepted transfer packets only through the displayed route

$$F, B, L, J \longmapsto S, R, D, E \longmapsto H, C, P, N.$$

No carrier coordinate stores a quotient filter, selected limit, countable-choice schedule, host equality of filter values, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.CauchyFilterModulusTransferUp.CauchyFilterModulusTransferTasteGate single carrier alignment`

Theorem 1982.2 (Cauchy filter modulus transfer NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ *carrier*

$$T = (F, B, L, J, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterModulusTransfer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Cauchy filter source, basis row, located-tail row, common-refinement row, window row, regular rational handoff, dyadic tolerance ledger, terminal seal, transport, replay, provenance, and local naming row;*

2. *source-to-basis routing, requiring each displayed filter request in F to read its finite base witness from B before the located-tail row can be inspected;*

3. *located-tail regularization, requiring each requested precision to pass through L and name a finite tail request that is small enough for regular Cauchy control;*

4. *common-refinement coverage, requiring J to refine the basis request and located-tail request over the same filter source before any window, readback, tolerance, or seal row is visible;*

5. *finite modulus handoff, requiring the route from J to pass through S, R, D before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary E;*

6. *transport and replay stability, requiring H and C to preserve the row order F, B, L, J, S, R, D, E under componentwise* hsame *transport and displayed* Cont *replay;*

7. *provenance exhaustion, requiring P, N to name only this finite transfer route and to export no quotient filter, selected limit, countable-choice schedule, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1982.1. The projection exposes exactly the filter source, basis route, located-tail row, common-refinement row, finite window, regular readback, dyadic ledger, terminal seal, and structural rows H, C, P, N.

The source-to-basis route is fixed by row order. A transfer request first reads F, then reads the finite base index, entourage request, and readiness evidence supplied by B before any located-tail regularization is available.

Located-tail regularization is the role of L. For a requested precision, L names the displayed tail request and records the finite locatedness evidence showing that this request is small enough for the regular Cauchy handoff.

The common-refinement row J then aligns the basis request with the located-tail request. Since J is a carrier coordinate, the refinement is finite data in the packet rather than an ambient filter lattice principle.

The modulus handoff is replayed by C through S, R, D before E can be inspected. Componentwise transport in H preserves the same rows, while P, N source and name only this packet; therefore the excluded quotient, choice, host-equality, real-completeness, Lean-verification, and bridge-checking data are outside the local NameCert surface. □

Theorem 1982.3 (Cauchy filter modulus transfer basis regularization)**.** *Let*

$$T = (F, B, L, J, S, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ *carrier. Every displayed filter precision request factors through a finite regularization route*

$$F \longmapsto B \longmapsto L \longmapsto J \longmapsto S, R, D.$$

Consequently the packet exports a regular Cauchy modulus row only after a source filter request has been witnessed by a basis row, located as a finite tail request, refined against that basis request, and checked by the displayed window, regular readback, and dyadic tolerance ledger.

Proof. Start with a displayed precision request in F. The carrier exposes no direct regular-modulus coordinate, so the request must first be read through the basis row B.

The located-tail row L supplies the finite tail request attached to the same precision. This step is constructive: the row contains the inspected tail witness and not a selected limit or a countable-choice schedule.

Next read J. It names the common refinement of the basis request and the located-tail request, so the regularization route remains attached to one displayed filter source.

Finally replay the finite observation rows S, R, D. The window row supplies the finite observations, the regular rational row repacks them as $\mathsf{RegSeqRat}^{\uparrow}$ data, and the dyadic ledger records the tolerance comparison needed for regular Cauchy control. Thus the regular modulus row is exported only through the displayed route. □

Theorem 1982.4 (Cauchy filter modulus transfer real seal boundary)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ *carrier*

$$T = (F, B, L, J, S, R, D, E, H, C, P, N)$$

reaches the terminal seal E only through the finite route

$$F, B, L, J, S, R, D \longmapsto E \longmapsto H, C, P, N.$$

In particular, the consumer receives a filter source row, basis row, located-tail row, common-refinement row, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise transport, continuation replay, provenance, and local naming. It receives no selected limit, quotient filter, countable-choice schedule, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge certificate.*

Proof. Begin with a completion-facing or $\mathsf{Real}^{\uparrow}$-facing read named by P. Carrier projection restricts the consumer to the rows displayed in Definition 1982.1.

Apply Definition 1982.3. The precision request must pass through F, B, L, J, S, R, D before a regular modulus row is available.

The seal row E is placed after that regularization route. A consumer therefore sees the terminal $\mathsf{Real}^{\uparrow}$ boundary only after the source filter, basis evidence, located tail, common refinement, finite windows, regular rational handoff, and dyadic tolerance ledger have all been replayed.

The structural rows do not add analytic content. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the displayed route, so selected limits, quotient filters, choice schedules, host equality, ambient real completeness, Lean verification, and bridge checking cannot enter through the seal boundary. □

Theorem 1982.5 (Cauchy filter modulus transfer RegularModulus bridge)**.** *Every accepted* CauchyFilterModulusTr *carrier*

$$T = (F, B, L, J, S, R, D, E, H, C, P, N)$$

whose route reaches S, R, D exports exactly the $\mathsf{RegularModulus}^{\uparrow}$ *row read by* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and completion-facing consumers. The exported row is the displayed finite modulus surface produced by F, B, L, J, S, R, D and preserved by E, H, C, P, N; it is not a quotient filter, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Apply Definition 1982.3. It routes each filter precision request through the source row, finite basis row, located-tail row, common-refinement row, StreamName window, RegSeqRat readback, and dyadic tolerance ledger.

Apply Definition 1982.4. The terminal seal can be inspected only after that regularization route has reached S, R, D, so the completion-facing consumer cannot bypass the displayed finite modulus data.

Finally use the $\mathsf{RegularModulus}^{\uparrow}$ handoff theorem, Definition 3768.3. The same window, regular-rational readback, and dyadic ledger are the rows read by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and completion consumers. Since neither carrier has a coordinate for a quotient filter, selected limit, countable-choice schedule, host equality, or ambient completeness, the bridge exports exactly the regular-modulus row and nothing more. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterModulusTransfer}^{\uparrow}$ packet is generated from a CauchyFilter source row, a CauchyFilterBase basis row, a LocatedCauchyFilter tail row, a CauchyFilterMeet common-refinement row, StreamName windows, RegSeqRat handoff, DyadicRatCore tolerances, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterModulusTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1982.1, Definition 1982.2, Definition 1982.3, and Definition 1982.4 seed the finite bridge from located Cauchy-filter evidence and filter-base requests to regular Cauchy modulus rows before the terminal Real seal.

Formal verification: FormalStatus(NameCert$_{\text{CauchyFilterModulusTransfer}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient filters, countable choice, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-to-basis routing, located-tail regularization, common-refinement coverage, finite modulus handoff, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterModulusTransferUp.

Chapter 1983

A Concrete Naming Certificate for $\mathsf{IrrationalWitness}^{\uparrow}$

Following Euclid's parity proof for the irrationality of $\sqrt{2}$ and Bishop–Bridges constructive analysis, the $\mathsf{IrrationalWitness}^{\uparrow}$ packet records a finite contradiction route from rational-candidate data to a Real-facing refusal seal. It reads rational numerator and denominator rows through chapter 37 and chapter 31, checks dyadic tolerances through chapter 291, routes square-root existence through chapter 1756 and chapter 1837, and exposes only a terminal Real seal from chapter 38. The packet is evidence for a finite irrationality obstruction inside the BEDC carrier discipline; it is not quotient real equality, a completeness theorem for $\mathsf{Real}^{\uparrow}$, a trichotomy principle, a selected real representative, or a host theorem that $\sqrt{2}$ is irrational.

Definition 1983.1 (IrrationalWitness carrier)**.** *An* $\mathsf{IrrationalWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, D, S, O, T, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational-candidate row, recording the inspected numerator and denominator data inherited from* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$*.*
2. *D is the denominator-positivity row, recording that the displayed denominator data are admitted before any square or root comparison may be read.*
3. *S is the square and root handoff row, routing the candidate equation through* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *and* $\mathsf{RealRootExistence}^{\uparrow}$ *before a Real-facing seal is exposed.*
4. *O is the parity or square-obstruction ledger, storing the finite divisibility, parity, or square-comparison rows used to contradict the displayed rational candidate.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording dyadic radii and comparison budgets for the rational and square-root reads.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, D, S, O, T have been displayed.*
7. *H is the componentwise* hsame *transport row for the rational candidate, denominator-positivity, square/root handoff, obstruction ledger, dyadic tolerance, Real seal, replay, provenance, and naming rows.*
8. *C is the displayed* Cont *replay row from a Real-facing irrationality request back through the square/root handoff, dyadic tolerance ledger, obstruction ledger, and rational-candidate rows.*

9. *P is the* Pkg *provenance over* $\mathsf{IrrationalWitness}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{RealRootExistence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IrrationalWitness}^{\uparrow}}$ *row.*

The carrier predicate accepts exactly these finite irrationality-witness packets. It has no coordinate for quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, selected representatives, or host irrationality theorems.*

Lean4 ✓ `BEDC.Derived.IrrationalWitnessUp.TasteGate.IrrationalWitnessTasteGate single carrier alignment`

Theorem 1983.2 (IrrationalWitness NameCert obligations)**.** *For every accepted* $\mathsf{IrrationalWitness}^{\uparrow}$ *carrier*

$$W = (Q, D, S, O, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IrrationalWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed rational-candidate row, denominator-positivity row, square/root handoff row, obstruction ledger, dyadic tolerance row, Real seal, and structural rows H, C, P, N;*
2. *rational-candidate exposure, requiring every contradiction read to display the numerator and denominator data in Q and the positivity row D before square data are consumed;*
3. *square/root route, requiring the row S to factor through* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *and* $\mathsf{RealRootExistence}^{\uparrow}$ *before any* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
4. *contradiction ledger, requiring the obstruction row O to store only the finite parity, divisibility, or square-comparison rows attached to Q, D, S, T;*
5. *Real-seal nonescape, requiring E to be downstream of Q, D, S, O, T and to expose no quotient real equality, full* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy branch, selected representative, or host irrationality theorem;*
6. *transport, replay, provenance, and naming, requiring H, C, P, N to preserve, replay, source, and name exactly the displayed route.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1983.1. The accepted packet displays exactly the rational-candidate row, denominator-positivity row, square/root handoff, obstruction ledger, dyadic tolerance row, terminal Real seal, and structural rows.

Rational-candidate exposure is finite. The row Q records the numerator and denominator data, while D records the positivity condition needed before the candidate can enter the square comparison route.

The square/root route is then forced by S and T. The handoff row passes through the regular-Cauchy square-root and root-existence packets, and the tolerance row records the dyadic budgets used by those reads.

The contradiction ledger is local to O. Its entries are parity, divisibility, or square-comparison rows attached to the displayed candidate and tolerance data, so it cannot import quotient equality, trichotomy, selected representatives, or ambient completeness.

Finally, E, H, C, P, N exhaust the Real-facing boundary. The seal E is downstream of the displayed route, and the structural rows transport, replay, source, and name only that route. Since the carrier has no further coordinate, the six listed clauses exhaust the local $\mathsf{NameCert}_{\mathsf{IrrationalWitness}^{\uparrow}}$ surface. □

Theorem 1983.3 (IrrationalWitness square-root handoff)**.** *For every accepted* $\mathsf{IrrationalWitness}^{\uparrow}$ *carrier*

$$W = (Q, D, S, O, T, E, H, C, P, N),$$

every Real-facing irrationality read exported by W *factors through the displayed route*

$$\mathsf{Rat}^{\uparrow}, \mathsf{Int}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegularCauchySquareRoot}^{\uparrow}, \mathsf{RealRootExistence}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

by reading the finite rows $Q, D, T, S, O, H, C, P, N$ *before the seal* E*. In particular, the read cannot bypass the rational, integer, dyadic-tolerance, square-root, and root-existence route by invoking quotient real equality, full* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, a selected representative, or a host irrationality theorem.*

Proof. Begin with a Real-facing irrationality consumer. Carrier projection exposes the terminal seal E only after the rational-candidate, denominator-positivity, dyadic-tolerance, square/root, and obstruction rows have been displayed.

The rational and integer data are read first through Q and D. The dyadic tolerance row T supplies the comparison budgets needed before the square-root route is inspected.

The row S then routes the read through the regular-Cauchy square-root packet and the real-root-existence packet. The obstruction ledger O records the finite contradiction rows attached to that route, not an external real-number theorem.

The structural rows do not add a second exit. The transport row H, replay row C, provenance row P, and local certificate row N preserve the same ordered route, so every Real-facing read factors through the displayed Rat/Int/DyadicRatCore and square-root/root-existence path before reaching E. □

Theorem 1983.4 (IrrationalWitness sqrt-two specialization)**.** *When the rational-candidate row* Q *of an accepted* $\mathsf{IrrationalWitness}^{\uparrow}$ *packet is the displayed sqrt-two candidate, the Real-facing refusal route must read the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$ *parity row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* T*, the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *and* $\mathsf{RealRootExistence}^{\uparrow}$ *handoff row* S*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *refusal seal* E*. The contradiction ledger* O *records the finite parity obstruction attached to that displayed candidate; it does not assert a host theorem that* $\sqrt{2}$ *is irrational, quotient real equality, full* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, or a selected real representative.*

Proof. Start from the displayed sqrt-two rational candidate in Q. The rational and integer rows are the only carrier coordinates that can expose numerator, denominator, divisibility, and parity data, so the obstruction begins inside the $\mathsf{Rat}^{\uparrow}$ and $\mathsf{Int}^{\uparrow}$ surface.

The dyadic tolerance row T is then consumed before the square/root comparison. It records the finite budgets used to compare the rational candidate with the square-root route, and it supplies no external real equality.

Apply the square-root handoff theorem Definition 1983.3. The row S routes the comparison through $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ and $\mathsf{RealRootExistence}^{\uparrow}$, while O stores the finite parity contradiction attached to Q, D, S, T.

Finally the seal E is read only after those rows and the structural transport, replay, provenance, and naming rows. Since the carrier has no coordinate for a host irrationality theorem, quotient equality, completeness, trichotomy, or selected representatives, the sqrt-two case remains a finite packet obstruction. □

Closure status for $\mathsf{NameCert}_{\mathsf{IrrationalWitness}^{\uparrow}}$**.**

Constructive story: An IrrationalWitness$^{\uparrow}$ packet is generated from rational-candidate rows, denominator positivity, square/root handoff rows, a parity or square-obstruction ledger, dyadic tolerance rows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{IrrationalWitness}^{\uparrow}}$, seed)

Scope closed: Definition 1983.1, Definition 1983.2, and Definition 1983.3 seed the finite route from rational-candidate evidence through dyadic and square-root data to a terminal RealUp irrationality refusal seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{IrrationalWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, full Real$^{\uparrow}$ completeness, trichotomy, selected representatives, host irrationality theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-candidate exposure, square/root routing, contradiction-ledger exactness, Real-seal nonescape, transport/replay/provenance coverage, and naming for BEDC.Derived.IrrationalWitnessUp.

Chapter 1984

A Concrete Naming Certificate for $\mathsf{LocatedRealSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRealSequence}^{\uparrow}$ packet records a finite sequence of located real names by displaying the sequence windows, located-real envelopes, modulus surface, regular rational handoff, and real-name seal before any consumer can read the sequence as a completed object. It sits after the finite StreamName carrier of chapter 126, the located-real boundary of **??**, the sequence-modulus surface of chapter 1801, the located Cauchy sequence packet of chapter 1595, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter isolates the finite located-real sequence carrier used by the real and completion route; it does not assume quotient streams, selected limits, countable choice, trichotomy, ambient real completeness, Lean verification, or checked bridge data.

Definition 1984.1 (Located real sequence source)**.** *A* $\mathsf{LocatedRealSequence}^{\uparrow}$ *source is a finite* BHist *packet*

$$X = (S, L, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *index-window row. It names the finite sequence indices and stream windows that a consumer may inspect.*
2. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *window row. It records the located-real enclosure exposed at each inspected index of S.*
3. *M is the* $\mathsf{LocatedSequenceModulus}^{\uparrow}$ *row. It binds finite tolerance requests to the inspected index windows and records the reusable sequence-modulus surface.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It carries the dyadic error budgets used when located-real windows are compared componentwise.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It records the regular rational readback obtained from the displayed sequence windows, located-real rows, modulus row, and dyadic tolerance row.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is read only after the regular rational handoff has been displayed, and it adds no selected limit or quotient-stream coordinate.*
7. *H records componentwise* hsame *transport for the index windows, located-real windows, modulus row, dyadic tolerance row, regular rational handoff, real seal, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from each sequence precision request through* $S, L, M, D, R, E, H, P, N$.
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedSequenceModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealSequence}^{\uparrow}}$ *row.*

The source exposes only these finite sequence, locatedness, tolerance, readback, seal, transport, continuation, provenance, and naming rows.

Lean4 ✓ `BEDC.Derived.LocatedRealSequenceUp.TasteGate.LocatedRealSequenceTasteGate single carrier alignment`

Theorem 1984.2 (Located real sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealSequence}^{\uparrow}$ *source* $X = (S, L, M, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealSequence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $S, L, M, D, R, E, H, C, P, N$ *packet;*
2. *located-window exposure, requiring each consumer-visible sequence coordinate to pass through the* $\mathsf{StreamName}^{\uparrow}$ *index-window row* S *and the corresponding* $\mathsf{LocatedReal}^{\uparrow}$ *row* L*;*
3. *componentwise classifier transport, requiring sameness of accepted sources to be transported only through the displayed* hsame *and* Cont *rows over the component windows, tolerance data, handoff, seal, provenance, and local name;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the readback row* R *to be formed only after* S, L, M, D *have been displayed;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring* E, P, N *to export no quotient stream, selected limit, countable-choice selector, trichotomy branch, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification certificate, or checked bridge.*

No located-real-sequence obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 1984.1. The accepted packet displays exactly the index windows, located-real windows, sequence modulus, dyadic tolerance row, regular rational handoff, real seal, transport, continuation, provenance, and local naming data.

Located-window exposure is forced before any component readback can occur. A consumer starts with the inspected index window S and receives the corresponding located-real row L at that same finite coordinate.

Componentwise classifier transport is structural. The rows H and C move and replay only the displayed rows S, L, M, D, R, E, while P and N source and name that same finite route.

The handoff and seal rows are downstream. The row R can be read only after S, L, M, D have been displayed, and the row E can be read only after R. Since no listed row stores quotient-stream equality, a selected limit, countable choice, trichotomy, ambient completeness, Lean verification, or bridge data, the five obligation rows exhaust the local NameCert surface. □

Theorem 1984.3 (Located real sequence kernel scope)**.** *Every consumer route out of an accepted* $\mathsf{LocatedRealSequence}^{\uparrow}$ *source factors through the finite chain*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{LocatedSequenceModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows. The route explicitly refuses quotient streams, selected limits, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the source of Definition 1984.1. The only entry point for a consumer is the finite index-window row S, and each inspected coordinate then exposes its located-real window through L.

The sequence-level control is the modulus row M. It binds the finite tolerance request before the dyadic tolerance row D can be used to compare component windows.

The regular rational and real-facing exits are then fixed. The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and the row E is the only $\mathsf{Real}^{\uparrow}$ seal, so every analytic read factors through the displayed chain from stream windows to the real seal.

The remaining rows BHist, hsame, Cont, Pkg, and NameCert transport, replay, source, and name the same finite route. They add no coordinate for quotient streams, selected limits, countable choice, trichotomy, ambient real completeness, Lean verification, or bridge checking. □

Theorem 1984.4 (Located real sequence null-tail handoff)**.** *Let*

$$X = (S, L, M, D, R, E, H, C, P, N)$$

and

$$X' = (S', L', M', D', R', E', H', C', P', N')$$

be accepted $\mathsf{LocatedRealSequence}^{\uparrow}$ *reads. Any zero-tail comparison between them that is consumed by a* $\mathsf{Real}^{\uparrow}$ *equality-facing route must first factor through the displayed sequence route*

$$S, L, M, D, R, E, H, C, P, N$$

and its primed counterpart, and then through a displayed $\mathsf{NullSequence}^{\uparrow}$ *packet. The handoff exposes finite StreamName windows, located-real component rows, sequence moduli, dyadic tolerances, regular readbacks, Real seals, transport, replay, provenance, and local names before any equality-facing consumer can read the zero-tail evidence.*

Proof. Project both accepted sources by Definition 1984.1. The only component rows available before the Real seal are the finite stream windows, located-real windows, sequence moduli, dyadic tolerances, regular readbacks, and structural rows.

Apply Definition 1984.3 to each source. Every consumer route out of either source must pass through the chain from $\mathsf{StreamName}^{\uparrow}$ to $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedSequenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, together with the displayed structural rows.

A zero-tail comparison is therefore an additional displayed consumer row rather than a replacement for either source route. It may be read as a $\mathsf{NullSequence}^{\uparrow}$ packet only after the two $S, L, M, D, R, E, H, C, P, N$ routes have been exposed.

Since neither carrier has a coordinate for quotient streams, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or checked bridge data, the equality-facing consumer receives only the displayed null-tail handoff and no hidden Real equality principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealSequence}^{\uparrow}$ packet is generated from finite StreamName index windows, LocatedReal component windows, a LocatedSequenceModulus row, DyadicRatCore tolerances, a RegSeqRat handoff, a Real seal, componentwise hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1984.1, Definition 1984.2, and Definition 1984.3 seed the finite located-real sequence carrier between StreamName, LocatedReal, LocatedSequenceModulus, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean closure requires checked carrier admission for Definition 1984.1, the obligation surface of Definition 1984.2, and the kernel scope theorem of Definition 1984.3 for BEDC.Derived.LocatedRealSequenceUp.

Chapter 1985

A Concrete Naming Certificate for $\mathsf{DenseImageCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completion and dense embeddings, the $\mathsf{DenseImageCompletion}^{\uparrow}$ packet records the finite L10 interface by which displayed source points approximate completion-facing consumers through the already named entry map of chapter 1934. It sits between the metric-completion surface of chapter 1506, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter names only finite approximation rows: it does not construct quotient completions, select limits, identify host objects, invoke countable choice, use ambient $\mathsf{Real}^{\uparrow}$ completeness, or claim bridge checking.

Definition 1985.1 (Dense image completion carrier)**.** *A* $\mathsf{DenseImageCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, E, A, M, R, S, D, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed source metric point row whose finite observations are available before any completion-facing read.*
2. *E is the* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ *entry row that exposes the source point as displayed completion-entry data.*
3. *A is the dense approximation witness row, recording the inspected completion request, the displayed source point chosen for that request, and the finite tolerance comparison between them.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row consumed by completion-facing clients after X, E, A have been displayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to answer numerical tolerance requests from the dense approximation witness.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source and approximation reads are exposed to consumers.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic radius, slack comparison, and budget consumed by A, R, S, M.*
8. *H records componentwise* hsame *transport for the source point, embedding row, approximation witness, metric-completion row, regular readback, stream windows, dyadic ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$X \longmapsto E \longmapsto A \longmapsto S \longmapsto R, D \longmapsto M.$$

10. *P is the* Pkg *provenance over* $\mathsf{DenseImageCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseImageCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.DenseImageCompletionUp.DenseImageCompletionTasteGate single carrier alignment`

Theorem 1985.2 (Dense image completion NameCert obligations)**.** *For an accepted* $\mathsf{DenseImageCompletion}^{\uparrow}$ *carrier $G = (X, E, A, M, R, S, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseImageCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source point, completion embedding row, dense approximation witness, metric-completion row, regular readback, stream windows, dyadic tolerance ledger, transport, replay, provenance, and local naming rows;*
2. *entry-before-approximation exposure, requiring every completion-facing request to read X through E before the dense witness A can be consumed;*
3. *dense approximation locality, requiring A to bind one displayed completion request, one displayed source row, and one dyadic tolerance budget from D;*
4. *readback readiness, requiring each numerical approximation claim to pass through S and R before it is offered to the metric-completion row M;*
5. *classifier stability, requiring transported packets with the same displayed X, E, A, M, R, S, D route to be identified through the carried* hsame *and* Cont *data;*
6. *ledger non-escape, stating that no quotient completion equality, selected limit, host equality, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential is exported by the packet.*

Proof. Carrier admission is projection from Definition 1985.1. The tuple lists exactly the source point, completion embedding, dense witness, metric-completion, readback, stream, dyadic ledger, transport, replay, provenance, and local naming rows.

Entry-before-approximation exposure follows because E is the only row connecting the displayed source point X to the completion-facing surface. A consumer cannot inspect a dense witness before the source point has been routed through this entry row.

Dense approximation locality is the projection to A together with the tolerance ledger D. The witness records a finite request, a displayed source point, and a dyadic budget; it does not create a host completion object or choose a limit.

Readback readiness uses S and R. The stream row exposes the finite windows inspected by the approximation witness, and the regular readback row is the only numerical route offered to the metric-completion consumer M.

Classifier stability is componentwise transport along H followed by replay through C. Since P and N only source and name the same finite packet, the excluded quotient, selected-limit, host-equality, choice, ambient-completeness, and bridge data have no carrier row from which they could be projected. □

Theorem 1985.3 (Dense image completion approximation route)**.** *Let $G = (X, E, A, M, R, S, D, H, C, P, N)$ be an accepted* $\mathsf{DenseImageCompletion}^{\uparrow}$ *carrier. Every completion-facing approximation exported by G factors through the finite route*

$$X \longmapsto E \longmapsto A \longmapsto S \longmapsto R, D \longmapsto M \longmapsto H, C, P, N.$$

Thus the dense-image statement is a displayed BEDC handoff from source rows to metric-completion consumers, not quotient completion equality, not a selected limit, not host equality, and not an appeal to countable choice or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing approximation request. Projection gives the source row X and the embedding row E, so the request first enters the completion interface through the same finite entry surface named by chapter 1934.

The dense witness A is then read with its dyadic budget D. This records a displayed source point near the inspected completion request at the stated finite tolerance.

The consumer next reads S and R. The stream row provides the finite windows behind the approximation, and the regular rational readback supplies the numerical data passed downstream.

Only after those rows are fixed does the route reach M, the metric-completion row. Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed route and add no quotient, selected-limit, host-equality, choice, ambient-completeness, or bridge credential. □

Theorem 1985.4 (Dense image completion ledger exhaustion)**.** *For an accepted* $\mathsf{DenseImageCompletion}^{\uparrow}$ *carrier $G = (X, E, A, M, R, S, D, H, C, P, N)$, every tolerance-facing, readback-facing, or metric-completion-facing read exported by G is generated by the rows named in Definition 1985.1. In particular, the packet exports no quotient completion equality, no selected limit, no host equality, no countable-choice schedule, no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no bridge-checking credential.*

Proof. Project the accepted carrier. A tolerance-facing read must pass through D, and the only rows that can consume that dyadic budget are the dense witness A, the stream windows S, the regular readback R, and the metric-completion row M.

A readback-facing consumer is similarly confined to S and R. The stream row fixes the finite observation window, while the regular readback row records the numerical approximation produced from that window and the dyadic ledger.

A metric-completion-facing consumer must pass through X, E, A, S, R, D before M is available. The completion row is therefore downstream of the displayed source entry and the finite dense-witness ledger.

The structural rows H, C, P, N transport, replay, source, and name only the same route. Since the tuple has no other coordinates, route exhaustion rules out quotient completion equality, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{DenseImageCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseImageCompletion}^{\uparrow}$ packet is generated from a displayed source metric point, a CauchyCompletionEmbedding entry row, a dense approximation witness, a MetricCompletion row, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseImageCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1985.1, Definition 1985.2, Definition 1985.3, and Definition 1985.4 seed the finite dense-image interface between completion embeddings and L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseImageCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entry-before-approximation exposure, dense approximation locality, readback readiness, classifier stability, ledger exhaustion, and a Lean carrier/obligation formalization for BEDC.Derived.DenseImageCompletionUp.

Chapter 1986

A Concrete Naming Certificate for $\mathsf{CauchyModulusSelection}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusSelection}^{\uparrow}$ packet records the finite threshold-selection surface that a regular Cauchy request exposes before any completed real object or quotient stream is consulted. It sits on the L10 route through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational source row of chapter 301, the Real boundary of chapter 38, and the extraction discipline of chapter 641. The packet records threshold selection for displayed requests only; it is not host choice, quotient stream equality, a selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1986.1 (Cauchy modulus-selection carrier)**.** *A* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (R, D, W, M, T, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational sequence surface whose requests are being served.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request row, including the displayed dyadic radius, slack, comparison, and scale data used by the selection.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window family available for the request.*
4. *M is the selected unary threshold row, read as a modulus value for the displayed tolerance and source row.*
5. *T is the request-threshold exactness ledger, recording that the threshold in M answers the tolerance request in D using only the inspected windows in W over the source row R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *request boundary, exposed only after R, D, W, M, T have been displayed and never as a completed limit or ambient completeness witness.*
7. *H records componentwise* hsame *transport for the regular source, dyadic request, finite windows, selected threshold, exactness ledger, Real boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every Real-facing request through $D, W, R, M, T, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSelection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, W, M, T, E, H, C, P, N$. It has no coordinate for host-level choice, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge data.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSelectionUp.CauchyModulusSelectionTasteGate single carrier alignment`

Theorem 1986.2 (Cauchy modulus-selection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *carrier*

$$X = (R, D, W, M, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, selected unary threshold, exactness ledger, Real request boundary, transport, replay, provenance, and local naming;*
2. *selected-threshold exactness, requiring T to bind the selected row M to the displayed tolerance request in D over the source row R;*
3. *dyadic tolerance compatibility, requiring every comparison and slack row used by M and T to be read through D;*
4. *finite-window handoff, requiring every source observation used by the selection to occur inside the displayed window family W;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring the source data consumed by the selector to be read through R before M is exported;*
6. $\mathsf{Real}^{\uparrow}$ *non-escape, requiring the boundary row E to remain downstream of R, D, W, M, T and to add no selected limit, quotient equality, trichotomy, or ambient completeness principle;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local modulus-selection obligation is stored outside these rows.

Proof. Project the carrier of Definition 1986.1. The analytic coordinates are exactly the regular rational source R, the dyadic tolerance request D, the finite windows W, the selected threshold M, the exactness ledger T, and the Real request boundary E.

Threshold exactness is the role of T. It records that M answers the tolerance request in D by reading the source row R only through the finite windows W.

Dyadic compatibility is forced by the absence of any second tolerance coordinate. Every radius, slack row, and comparison used by the selection is therefore read through D.

The handoff to Real-facing consumers is terminal. The row E is downstream of R, D, W, M, T, while H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host choice, quotient equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 1986.3 (Cauchy modulus-selection Real request handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing finite request accepted by a* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *carrier*

$$X = (R, D, W, M, T, E, H, C, P, N)$$

factors through the finite route

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as recorded by the displayed rows D, W, R, E, with the selected threshold row M and exactness ledger T visible before the Real seal is reached.

Proof. Begin with a Real-facing request. By the carrier definition, the request boundary is the row E, and E is available only after the dyadic request, finite windows, regular source, selected threshold, and exactness ledger have been displayed.

Read the route backward from E. The tolerance side is the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, the finite observation surface is the $\mathsf{StreamName}^{\uparrow}$ row W, and the source readback is the $\mathsf{RegSeqRat}^{\uparrow}$ row R.

The selected threshold M is not external to this route. The ledger T binds it to the same D, W, R data before any consumer can inspect the Real boundary.

Finally, H, C, P, N preserve and name the displayed route. Hence the accepted request factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order, with the selected threshold visible before the Real seal and with no bypass through host choice, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Theorem 1986.4 (Cauchy modulus-selection threshold uniqueness)**.** *For every accepted* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *carrier*

$$X = (R, D, W, M, T, E, H, C, P, N),$$

every exported threshold for the displayed request is read through the single selected row M and certified by the exactness ledger T. No consumer of the packet can introduce a second modulus row outside the displayed coordinates $R, D, W, M, T, E, H, C, P, N$.

Proof. Project the carrier of Definition 1986.1. Its only coordinate whose role is a selected unary threshold is M, and its only coordinate whose role is request-threshold exactness is T.

Apply Definition 1986.2. The obligation surface binds selected-threshold exactness to T and requires T to bind M to the displayed dyadic request in D over the source row R.

The classifier has no further threshold coordinate. Therefore any exported threshold visible to a consumer must be the row already displayed as M, certified by T, and transported or replayed only through H, C, P, N. □

Theorem 1986.5 (Cauchy modulus-selection dyadic monotonicity)**.** *For every accepted* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *carrier*

$$X = (R, D, W, M, T, E, H, C, P, N),$$

any refinement of a dyadic tolerance request must be routed through the same $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the selected threshold row M and exactness ledger T are visible. In particular, refined requests cannot bypass the dyadic tolerance ledger or the finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*

Proof. Read the analytic route from the carrier. The tolerance request is D, the finite observation surface is W, the regular rational source row is R, and the selected threshold and exactness rows are M, T.

Apply Definition 1986.3. Every Real-facing request reaches the boundary E only after passing through the displayed D, W, R route with M and T visible before the Real seal.

A refined dyadic tolerance request is still a dyadic request. Since the carrier has no second tolerance ledger and no external window coordinate, refinement can only change the data read inside D and then pass through W, R, M, T in the same order. □

Theorem 1986.6 (Cauchy modulus-selection transport closure)**.** *If two accepted* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *packets are classifier-related through their displayed rows, then their selected threshold rows transport componentwise through* H *and replay the same finite route through* C*. The transported route preserves the order* D, W, R, M, T, E *and introduces no external modulus row, tolerance ledger, Real limit, or quotient stream equality.*

Proof. Use the carrier coordinates for the two accepted packets. Classifier relatedness is measured only through the displayed rows $R, D, W, M, T, E, H, C, P, N$, so componentwise comparison has a threshold component, an exactness component, a Real-boundary component, and the transport and replay components.

The row H records the componentwise hsame transport for the regular source, dyadic request, finite windows, selected threshold, exactness ledger, Real boundary, replay, provenance, and local naming rows. Transport of the selected threshold is therefore the M component of the same displayed relation, not a new selector.

The row C records displayed Cont replay from every Real-facing request through $D, W, R, M, T, E, H, P, N$. Replaying after transport follows that route in the same order, so the transported packet preserves the finite handoff and remains inside the original classifier surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSelection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusSelection}^{\uparrow}$ packet is generated from a RegSeqRat source row, a DyadicRatCore tolerance request, finite StreamName observation windows, a selected unary threshold row, a request-threshold exactness ledger, a RealUp request boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1986.1, Definition 1986.2, Definition 1986.3, Definition 1986.4, Definition 1986.5, and Definition 1986.6 seed the finite threshold-selection certificate for regular Cauchy requests on the L10 route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selected-threshold exactness, dyadic tolerance compatibility, finite-window handoff, RegSeqRat readback, RealUp non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSelectionUp.

Chapter 1987

A Concrete Naming Certificate for $\mathsf{FiniteEpsilonWitness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{FiniteEpsilonWitness}^{\uparrow}$ packet isolates the finite row that answers one inspected epsilon request before any completed real, quotient stream, or ambient choice principle is consulted. It sits on the L10 Real and Completion route through the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite $\mathsf{StreamName}^{\uparrow}$ observation surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, the threshold-selection interface of chapter 1986, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records one explicit epsilon-witness row; it is not a global modulus, a selected limit, quotient equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1987.1 (Finite epsilon-witness carrier)**.** *A* $\mathsf{FiniteEpsilonWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, D, W, M, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the inspected epsilon request row, including the displayed request token and the local comparison budget that the packet is required to answer.*
2. *D is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row refining E into a dyadic radius, scale, slack, and positivity ledger.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose inspected coordinates are sufficient for the displayed request.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row, recording the finite index bound used for the single epsilon request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the source stream inside the window W and threshold M.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, exposed only after E, D, W, M, R have been displayed and never as an independent completeness or quotient-equality witness.*
7. *H records componentwise* hsame *transport for the epsilon request, dyadic tolerance, finite windows, threshold, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the inspected epsilon request through D, W, M, R, S, H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteEpsilonWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteEpsilonWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, W, M, R, S, H, C, P, N$. It has no coordinate for a family of epsilon requests, a global modulus, a selected limit, quotient stream equality, trichotomy, ambient completeness, Lean verification, or standard bridge data.

Lean4 ✓ `BEDC.Derived.FiniteEpsilonWitnessUp.FiniteEpsilonWitnessTasteGate single carrier alignment`

Theorem 1987.2 (Finite epsilon-witness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteEpsilonWitness}^{\uparrow}$ *carrier*

$$X = (E, D, W, M, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteEpsilonWitness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected epsilon request, positive dyadic tolerance row, finite StreamName window, CauchyModulus threshold, RegSeqRat readback, Real seal boundary, transport, replay, provenance, and local naming;*

2. *request positivity, requiring D to record a positive dyadic tolerance refining the inspected request E;*

3. *finite-window sufficiency, requiring every coordinate used to answer E to occur inside the displayed window row W;*

4. *threshold exactness, requiring M to bind the request E and tolerance row D to a single finite bound before readback;*

5. *rational readback, requiring R to be obtained from the* $\mathsf{RegSeqRat}^{\uparrow}$ *source through W and M;*

6. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring S to remain downstream of E, D, W, M, R and to add no selected limit, quotient equality, trichotomy, or ambient completeness principle;*

7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local finite epsilon-witness obligation is stored outside these rows.

Proof. Project the carrier of Definition 1987.1. The analytic rows are exactly the inspected request E, the positive dyadic tolerance D, the finite windows W, the threshold row M, the rational readback R, and the Real seal boundary S.

Request positivity is the role of D. Since the carrier has no second tolerance coordinate, the radius, scale, slack, and positivity ledger answering E must be read from D.

The finite computation is forced through W, M, R. The window row W lists the inspected StreamName coordinates, the threshold row M binds those coordinates to the single request E and tolerance D, and R records the RegSeqRat readback available after that threshold has been displayed.

The seal row is terminal. The row S is downstream of E, D, W, M, R, while H, C, P, N transport, replay, source, and name the same finite route. Because the classifier has no coordinate for a family of requests, a global modulus, selected limits, quotient stream equality, trichotomy, ambient completeness, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 1987.3 (Finite epsilon-witness Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing answer accepted by a* $\mathsf{FiniteEpsilonWitness}^{\uparrow}$ *carrier*

$$X = (E, D, W, M, R, S, H, C, P, N)$$

factors through the finite route

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as recorded by the displayed rows D, W, M, R, S*, with the inspected epsilon request* E *visible at the start of the route.*

Proof. Begin with a Real-facing answer admitted by the carrier. By definition, its seal coordinate is S, and S is exposed only after the request E, tolerance row D, finite windows W, threshold row M, and rational readback R have been displayed.

Read the route from the request side. The inspected epsilon row E is refined by the positive $\mathsf{DyadicRatCore}^{\uparrow}$ row D. The finite observation surface is the $\mathsf{StreamName}^{\uparrow}$ row W, and the bound used for that observation is the $\mathsf{CauchyModulus}^{\uparrow}$ row M.

The readback into rational sequence data is R. Only after this readback is visible may the terminal $\mathsf{Real}^{\uparrow}$ seal boundary S be inspected.

Finally, H, C, P, N preserve and name the displayed route. Hence every accepted Real-facing answer factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order, with no bypass through a global modulus, selected limit, quotient stream equality, trichotomy, ambient completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteEpsilonWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteEpsilonWitness}^{\uparrow}$ packet is generated from one inspected epsilon request, a positive DyadicRatCore tolerance row, a StreamName finite-window row, a CauchyModulus threshold row, a RegSeqRat readback row, a RealUp seal boundary, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteEpsilonWitness}^{\uparrow}}$, seed)

Scope closed: Definition 1987.1, Definition 1987.2, and Definition 1987.3 seed the finite single-request epsilon-witness certificate on the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteEpsilonWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteEpsilonWitnessUp.FiniteEpsilonWitnessTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close global moduli, families of epsilon requests, selected limits, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request positivity, finite-window sufficiency, threshold exactness, RegSeqRat readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteEpsilonWitnessUp.

Chapter 1988

A Concrete Naming Certificate for $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$

Following Bishop–Bridges constructive completion, the $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$ packet records the finite composite admission route from dyadic tolerance data to a $\mathsf{Real}^{\uparrow}$-facing completion read. It ties together the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the threshold handoff of chapter 1986, the nested interval route of chapter 1924, and the universal boundary of chapter 1838. The certificate is a finite route record only; it is not quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1988.1 (Real completion finite-route carrier)**.** *A* $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, W, R, A, M, J, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, displaying the requested dyadic scale, slack, comparison, and radius data used by the completion read.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the displayed tolerance request and for every rational observation consumed downstream.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from D and W.*
4. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dyadic tolerance, finite windows, and regular readback rows have been displayed.*
5. *M is the* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *row binding the displayed tolerance request to a finite threshold and exactness ledger before A is exported.*
6. *J is the* $\mathsf{NestedIntervalReal}^{\uparrow}$ *route row, recording the nested-interval source, dyadic diameter ledger, endpoint seals, finite windows, regular readback, and terminal Real seal used by the same completion read.*
7. *U is the* $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$ *face, recording the extension and uniqueness boundary available to consumers of the displayed completion route.*

8. *H records componentwise* hsame *transport for the dyadic tolerance row, finite windows, regular readback, terminal Real seal, modulus-selection row, nested-interval route, universal-boundary face, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay from every Real-facing completion request through $D, W, R, A, M, J, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusSelection}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{BishopCompletionUniversalBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionFiniteRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, A, M, J, U, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge data.*

Lean4 ✓ `BEDC.Derived.RealCompletionFiniteRouteUp.RealCompletionFiniteRouteTasteGate single carrier alignment`

Theorem 1988.2 (Real completion finite-route NameCert obligations)**.** *For every accepted* RealCompletionFiniteRo *carrier*

$$F = (D, W, R, A, M, J, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionFiniteRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance row, finite stream windows, regular rational readback, terminal Real seal, modulus-selection row, nested-interval route, universal-boundary face, transport, replay, provenance, and local naming row;*

2. *tolerance exactness, requiring every completion request to expose its* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, slack, scale, and comparison data in D before any regular readback or Real seal is inspected;*

3. *finite-window exactness, requiring every rational observation used by the readback, modulus selector, nested route, or universal-boundary consumer to occur inside the displayed* $\mathsf{StreamName}^{\uparrow}$ *family W;*

4. *regular-readback ordering, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be generated from D, W before the terminal* $\mathsf{Real}^{\uparrow}$ *seal A is exported;*

5. *modulus-selection compatibility, requiring M to bind the same tolerance row D and window family W to a finite threshold ledger for the Real-facing request;*

6. *nested-route compatibility, requiring J to reuse the same dyadic, stream, regular-readback, and Real-seal surface instead of introducing an ambient intersection point or selected limit;*

7. *universal-boundary compatibility, requiring U to expose extension and uniqueness reads only after D, W, R, A, M, J have fixed the displayed completion route;*

8. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient stream, selected limit, host real equality, countable-choice schedule, trichotomy branch, open-cover compactness row, ambient Real completeness theorem, Lean verification marker, or bridge credential.*

No Real completion finite-route obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1988.1. The accepted packet displays one dyadic tolerance row, one finite window family, one regular rational readback, one terminal Real

seal, one modulus-selection row, one nested-interval route, one universal-boundary face, and the structural rows that transport, replay, source, and name them.

Tolerance exactness is read from D. Since the carrier has no second tolerance coordinate, every scale, slack, radius, and comparison used by the completion read must be the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ data.

Finite-window exactness is read from W. The regular readback R, the selected threshold row M, the nested-interval route J, and the universal-boundary face U all consume the same finite observations, so no hidden stream or quotient representative can supply additional data.

The ordering of the analytic route is fixed by R, A, M, J, U. The readback R is downstream of D, W, the Real seal A is downstream of R, the modulus row M binds the tolerance request before that seal is exported, the nested route J records the same read as interval data, and the boundary face U is available only after the completion route has been displayed.

Finally, H, C, P, N preserve, replay, source, and name the same finite route. The excluded quotient, selected-limit, host-equality, choice, trichotomy, open-cover, ambient-completeness, Lean-verification, and bridge objects are absent from the carrier, so the listed rows exhaust the local NameCert surface. □

Theorem 1988.3 (Real completion finite-route determinacy)**.** *Let*

$$F = (D, W, R, A, M, J, U, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing completion read exported by* F *factors through exactly the displayed finite route*

$$D \longrightarrow W \longrightarrow R \longrightarrow M \longrightarrow J \longrightarrow U \longrightarrow A$$

with componentwise transport, replay, provenance, and local naming supplied only by H, C, P, N*. Consequently the exported read cannot use quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Fix a Real-facing completion read exported by F. The only Real-seal coordinate in the carrier is A, so the read must reach the Real boundary through the displayed terminal seal.

Read the dependencies of A inside the carrier. The seal is downstream of the regular readback R, and R is generated from the dyadic tolerance row D and the finite stream windows W.

The same request is constrained by the local route certificates. The row M records the finite modulus-selection handoff for the displayed tolerance and windows, J records the nested-interval realization of the same readback and seal surface, and U records the Bishop completion universal boundary visible to extension and uniqueness consumers.

The structural rows do not add analytic data. The hsame row H transports the displayed coordinates componentwise, the Cont row C replays the displayed route, and P, N source and name it.

There are no remaining coordinates from which a different route could be read. In particular the carrier contains no quotient stream, selected-limit row, host equality, countable-choice schedule, trichotomy branch, open-cover compactness object, or ambient completeness theorem. Thus every exported Real-facing completion read factors through exactly the displayed finite route and through no excluded object. □

Theorem 1988.4 (Real completion finite-route modulus refinement)**.** *Let* $F = (D, W, R, A, M, J, U, H, C, P, N)$ *be an accepted* $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$ *carrier. If a later consumer strengthens the displayed*

$\mathsf{CauchyModulusSelection}^{\uparrow}$ *row M by a finer finite threshold ledger over the same dyadic request, the exported completion read still factors through the same route*

$$D \longrightarrow W \longrightarrow R \longrightarrow M \longrightarrow J \longrightarrow U \longrightarrow A.$$

The refinement only narrows the finite window requests before the terminal $\mathsf{Real}^{\uparrow}$ *seal A is read; it does not introduce a selected limit, quotient stream, trichotomy branch, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 1988.1. The only modulus-selection coordinate is M, and it is already tied to the displayed dyadic tolerance row D and finite StreamName windows W.

Apply Definition 1988.2. Modulus-selection compatibility requires M to bind the same D, W data used by the regular readback, nested-route row, universal-boundary face, and terminal Real seal.

Apply Definition 1988.3. Every exported read is forced through

$$D \longrightarrow W \longrightarrow R \longrightarrow M \longrightarrow J \longrightarrow U \longrightarrow A.$$

A finer threshold ledger therefore changes only which finite windows inside W are requested before R and A are consumed.

No excluded coordinate is created by that narrowing. The carrier still has no selected limit, quotient stream, trichotomy branch, open-cover compactness row, or ambient Real completeness theorem, and the structural rows H, C, P, N only transport, replay, source, and name the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionFiniteRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionFiniteRoute}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, a CauchyModulusSelection row, a NestedIntervalReal route, a BishopCompletionUniversalBoundary face, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionFiniteRoute}^{\uparrow}}$, seed)

Scope closed: Definition 1988.1, Definition 1988.2, and Definition 1988.3 seed the finite composite route from dyadic tolerance data to Real-facing completion reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionFiniteRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, finite-window exactness, regular-readback ordering, modulus-selection compatibility, nested-route compatibility, universal-boundary compatibility, route determinacy, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionFiniteRouteUp.

Chapter 1989

A Concrete Naming Certificate for $\mathsf{RealCauchyFilter}^{\uparrow}$

Following Bourbaki and Kelley on Cauchy filters, and Bishop–Bridges on constructive real completion, the $\mathsf{RealCauchyFilter}^{\uparrow}$ packet records the finite route by which a displayed Cauchy-filter base is read as regular rational sequence data before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits after the filter-to-net correspondence of chapter 1803, the finite common-refinement operation of chapter 1958, the $\mathsf{RegSeqRat}^{\uparrow}$ handoff of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet names a finite Real-facing completion interface; it is not a quotient-filter construction, ultrafilter principle, selected limit, choice principle, ambient completeness theorem, Lean verification row, or checked bridge.

Definition 1989.1 (Real Cauchy filter carrier)**.** *A* $\mathsf{RealCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, M, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the filter-base row. It records the finite Cauchy-filter base requests, refinement witnesses, and membership reads available before any real-facing consumer inspects the packet.*
2. *M is the finite meet row. It records the* $\mathsf{CauchyFilterMeet}^{\uparrow}$ *common-refinement reads that combine two displayed base requests over the same Cauchy-filter surface.*
3. *D is the dyadic entourage and tolerance ledger. It records the requested dyadic scale, the refinement scale used by the meet row, and the comparison showing that the displayed base request is Cauchy at that scale.*
4. *S is the regular-sequence extraction row. It reads finite windows through* $\mathsf{StreamName}^{\uparrow}$ *and repacks the accepted Cauchy-filter observations as* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, M, D, S have been displayed.*
6. *E records the* hsame *transport routes for the filter base, meet, dyadic ledger, regular-sequence extraction, Real seal, continuation, provenance, and local naming rows.*
7. *H records displayed* Cont *replay from a Real-facing precision request through F, M, D, S, Q in that order.*

8. *C is the* Pkg *provenance over* RealCauchyFilter$^{\uparrow}$, CauchyFilterNetCorrespondence$^{\uparrow}$, CauchyFilterMeet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *P records the local package provenance tying the displayed route to the surrounding Real-and-Completion corridor.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed route

$$F, M, D, S, Q \longmapsto E, H, C, P, N.$$

There is no carrier coordinate for quotient filters, ultrafilters, selected limits, Classical.choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealCauchyFilterUp.RealCauchyFilterTasteGate single carrier alignment`

Theorem 1989.2 (Real Cauchy filter NameCert obligations)**.** *For every accepted* RealCauchyFilter$^{\uparrow}$ *carrier*

$$R = (F, M, D, S, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the filter-base row, finite meet row, dyadic tolerance ledger, regular-sequence extraction row, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *filter-base stability, requiring every transported base request to remain within the displayed* hsame *route before any meet or extraction row is read;*
3. *finite-meet closure, requiring pairs of displayed base requests to pass through M and return a common refinement over the same filter-base row;*
4. *Cauchy entailment, requiring each Real-facing tolerance request to be witnessed by the dyadic ledger D after the finite meet row has been read;*
5. *regular-sequence handoff, requiring the accepted filter observations to pass through* StreamName$^{\uparrow}$ *windows and the* RegSeqRat$^{\uparrow}$ *extraction row S before the Real seal is visible;*
6. Real$^{\uparrow}$ *seal descent, requiring the terminal row Q to be consumed only after the route F, M, D, S has been replayed;*
7. *nonescape, requiring the provenance and local naming rows to export no quotient filters, ultrafilters, selected limits, Classical.choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1989.1. The projection exposes exactly the filter base, finite meet, dyadic tolerance ledger, regular-sequence extraction, Real seal, transport, replay, provenance, and local naming rows.

Filter-base stability is supplied by the hsame row E. A transported precision request must still name the same displayed base surface before it can be routed to M, D, or S.

Finite-meet closure is the role of M. Given two displayed base requests, the meet row names a common finite refinement over the same filter base; it does not appeal to an ambient lattice of filters or to quotient-filter equality.

Cauchy entailment is read through D. After the meet row has fixed a common refinement, the dyadic ledger records the requested tolerance and the smaller displayed tolerance that makes the base request Cauchy at the requested scale.

The regular-sequence handoff then passes through S. This row reads the same observations through $\mathsf{StreamName}^{\uparrow}$ windows and repacks them as $\mathsf{RegSeqRat}^{\uparrow}$ data before the terminal seal can be consumed.

Finally the Real seal row Q is downstream of the whole route, while H, C, P, N only replay, source, and name that route. The excluded quotient, ultrafilter, selected-limit, choice, ambient-completeness, verification, and bridge-checking data therefore remain outside the local NameCert surface. □

Theorem 1989.3 (Real Cauchy filter regular-sequence handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RealCauchyFilter}^{\uparrow}$ *carrier*

$$R = (F, M, D, S, Q, E, H, C, P, N)$$

factors through the finite route

$$F \longmapsto M \longmapsto D \longmapsto S \longmapsto Q \longmapsto E, H, C, P, N.$$

In particular, the read first consumes the filter-base row, then the finite-meet row, then the dyadic tolerance ledger, then the $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *extraction row, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It receives no quotient filter, ultrafilter, selected limit, Classical.choice, ambient completeness principle, Lean verification row, or bridge certificate.*

Proof. Begin with a Real-facing consumer named by the provenance rows. By Definition 1989.1, the only terminal Real coordinate is Q, and the displayed continuation to Q is routed through F, M, D, S.

The filter-base row F supplies the source requests. The finite meet row M then aligns pairs of such requests into common refinements over the same base, so the route has no need for quotient filters or ultrafilters.

The dyadic ledger D records the tolerance comparison attached to the chosen refinement. This is the Cauchy evidence consumed by the later regular-sequence extraction row, not a selected host limit.

Next S reads the displayed observations through $\mathsf{StreamName}^{\uparrow}$ windows and repacks the route as $\mathsf{RegSeqRat}^{\uparrow}$ data. Only after that handoff can the terminal $\mathsf{Real}^{\uparrow}$ seal row Q be inspected.

The structural rows E, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory has no coordinate for quotient filters, ultrafilters, selected limits, Classical.choice, ambient completeness, Lean verification, or bridge checking, every Real-facing read is exhausted by the factorization above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyFilter}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 1989.1, Definition 1989.2, and Definition 1989.3 seed the finite Cauchy-filter route into regular rational sequence data and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, ultrafilters, selected limits, Classical.choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-base stability, finite-meet closure, Cauchy entailment, regular-sequence handoff, RealUp seal descent, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyFilter

Constructive story: A $\mathsf{RealCauchyFilter}^{\uparrow}$ packet is generated from a filter-base row, finite meet row, dyadic tolerance ledger, regular-sequence extraction row, Real seal row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1990

A Concrete Naming Certificate for $\mathsf{RealCauchySum}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealCauchySum}^{\uparrow}$ packet records the sum of a constructive real series as a displayed Cauchy endpoint for finite partial sums, not as a host infinite-sum object. It sits after the real-series carrier of chapter 1581, the uniform tail route of chapter 1736, the absolute-domination certificate of chapter 1700, the selected stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the endpoint seal of chapter 38. The packet gives later $\mathsf{Real}^{\uparrow}$ arithmetic and convergence consumers a finite sum-seal interface whose obligations are local to BEDC rows.

Definition 1990.1 (Real Cauchy sum carrier)**.** *A* $\mathsf{RealCauchySum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, W, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{RealSeries}^{\uparrow}$ *term row, read only through the finite real-valued observations already exposed by the series packet.*
2. *S is the finite partial-sum window row, listing selected initial segments, adjacent summand contributions, and the displayed* $\mathsf{Real}^{\uparrow}$ *seals for those partial sums.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family used for the summands, partial sums, tolerance comparisons, and endpoint-facing reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every selected summand, partial-sum window, and endpoint approximation opened by the packet.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording addition budgets, comparison radii, tail requests, and readback tolerances.*
6. *M is the Cauchy-tail modulus row: each requested tolerance is paired with a displayed threshold after which all selected partial-sum windows are compared inside the dyadic ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *sum seal, visible only after T, S, W, R, D, M have displayed the finite Cauchy route.*
8. *H stores componentwise* hsame *transport for the series term row, partial-sum windows, StreamName windows, RegSeqRat readbacks, dyadic ledger, tail modulus, Real seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from term reads to partial sums, from partial sums to regular readback, from readback to the dyadic tolerance ledger, from the ledger to the Cauchy-tail modulus, and from that modulus to the Real sum seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealCauchySum}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchySum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, W, R, D, M, E, H, C, P, N$. It has no coordinate for a host infinite sum, quotient stream equality, selected limit, countable-choice selector, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or standard bridge.*

Theorem 1990.2 (Real Cauchy sum NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchySum}^{\uparrow}$ *carrier $K = (T, S, W, R, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCauchySum}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed RealSeries term row, partial-sum windows, StreamName windows, RegSeqRat readbacks, DyadicRatCore ledger, Cauchy-tail modulus, Real sum seal, transport, replay, provenance, and local naming rows;*

2. *partial-sum window exactness, requiring each finite sum observation to pass through T, S, W, R, D before any tail or endpoint row is read;*

3. *Cauchy-tail modulus exactness, requiring M to supply the threshold used for every accepted comparison of later partial-sum windows at the requested tolerance;*

4. *RegSeqRat and dyadic readback coherence, requiring R and D to expose only the selected windows and tolerances named by the packet;*

5. *Real seal nonescape, requiring E to consume the displayed Cauchy route and to export no host infinite sum, quotient stream equality, selected limit, countable choice, trichotomy, ambient completeness, Lean verification, or bridge datum;*

6. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Real Cauchy sum obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (T, S, W, R, D, M, E, H, C, P, N)$. The term row T is already a $\mathsf{RealSeries}^{\uparrow}$ surface, so the classifier sees only the finite term observations and partial-sum data exposed by the packet.

The partial-sum row S supplies the finite windows. These windows are opened through W, read as regular rational data through R, and measured against the dyadic budgets recorded in D.

The modulus row M is the only tail-control coordinate. Once a tolerance is requested, every accepted later-window comparison must use the displayed threshold in M and the ledger route through D.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since no excluded object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealCauchySumUp.RealCauchySumCarrier namecert obligations`

Theorem 1990.3 (Real Cauchy sum RegSeqRat handoff)**.** *Every endpoint-facing read from an accepted* $\mathsf{RealCauchySum}^{\uparrow}$ *carrier factors through*

$$T \longmapsto S \longmapsto W \longmapsto R \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

In particular, the RegSeqRat readback row is the only route from selected partial-sum windows to the dyadic tolerance ledger, and the Real sum seal is read only after that regular readback and the displayed Cauchy-tail modulus are already in place.

Proof. Begin with an endpoint-facing request. The carrier definition forces the request to read the $\mathsf{RealSeries}^{\uparrow}$ term row T and the finite partial-sum windows S before any endpoint seal is visible.

The selected observations then pass through W and R. Thus the packet exposes its summands, partial sums, and endpoint approximations as $\mathsf{StreamName}^{\uparrow}$ windows with $\mathsf{RegSeqRat}^{\uparrow}$ readback, not as a host sequence quotient or a chosen limit.

The dyadic ledger D and tail modulus M provide the Cauchy comparison surface. Each requested tolerance is accounted for in D and paired with the displayed threshold stored in M.

The seal row E is downstream of this route, and H, C, P, N preserve, replay, source, and name the same data. Therefore the handoff reaches the Real sum seal without exporting any hidden infinite-sum principle. □

Theorem 1990.4 (Real Cauchy sum cofinal-tail closure)**.** *For every accepted* $\mathsf{RealCauchySum}^{\uparrow}$ *carrier* $K = (T, S, W, R, D, M, E, H, C, P, N)$*, every endpoint-facing comparison of later partial-sum windows at a requested dyadic tolerance factors through the displayed* $\mathsf{RealSeries}^{\uparrow}$ *term row, the finite partial-sum windows, the selected* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the Cauchy-tail modulus row* M*, the terminal* $\mathsf{Real}^{\uparrow}$ *sum seal* E*, and the stored* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. Consequently, the route from cofinal tail control to the endpoint seal is finite inside the packet and has no coordinate for a quotient stream, host infinite sum, selected limit, ambient completeness row, or detached Real seal.*

Proof. Project the accepted carrier supplied by Definition 1990.1. The rows $T, S, W, R, D, M, E, H, C, P, N$ are the whole classifier surface, so an endpoint-facing tolerance request can only address those displayed coordinates.

Apply Definition 1990.2 to expose carrier admission, partial-sum window exactness, and Cauchy-tail modulus exactness. The requested dyadic tolerance is first recorded in D, then paired with its displayed cofinal threshold in M.

Apply Definition 1990.3 to force each later-window read through the same finite partial-sum windows and the same $\mathsf{RegSeqRat}^{\uparrow}$ row R. Thus the comparison route cannot bypass W, R, D while still reaching the endpoint-facing seal.

The final seal E is downstream of M, and H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no slot for a quotient stream, host infinite sum, selected limit, ambient completeness row, or detached Real seal, the cofinal-tail comparison closes as a finite BEDC route. □

Falsifiable prediction: Every $\mathsf{RealCauchySum}^{\uparrow}$ endpoint-facing read factors through a RealSeries term row, finite partial-sum windows, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Cauchy-tail modulus, a RealUp sum seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, uniform-cauchy-series, absolute-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{RealCauchySum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite series surface, partial-sum windows, regular readback, dyadic tolerance accounting, a Cauchy-tail modulus, endpoint Real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchySum}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\mathsf{RealCauchySum}^{\uparrow}}$, seed)

Scope closed: Definition 1990.1, Definition 1990.2, Definition 1990.3, and Definition 1990.4 seed the finite constructive series-sum route from RealSeries rows through RegSeqRat readback, DyadicRatCore tolerance accounting, Cauchy-tail modulus control, and RealUp endpoint sealing.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCauchySum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum window exactness, RegSeqRat and DyadicRatCore handoff, Cauchy-tail modulus exactness, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.RealCauchySumUp.

Constructive story: A RealCauchySum$^{\uparrow}$ packet is generated from a RealSeries term row, finite partial-sum windows, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Cauchy-tail modulus, a RealUp sum seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1991

A Concrete Naming Certificate for $\mathsf{TauberianRemainder}^{\uparrow}$

Following Hardy's treatment of Tauberian theorems and Bishop–Bridges constructive analysis, the $\mathsf{TauberianRemainder}^{\uparrow}$ packet records the finite remainder boundary between Abel summation and Cesaro or ordinary real readback. It sits after the summation-by-parts route of chapter 1763, the averaging surface of chapter 1765, the regular rational readback of chapter 301, the dyadic budget carrier of chapter 291, the terminal real seal of chapter 38, and the Tauberian handoff of chapter 1932. The packet names only a finite remainder route; it is not a selected limit, quotient series equality, host real completeness theorem, countable choice, trichotomy, or open-cover compactness principle.

Definition 1991.1 (TauberianRemainder carrier)**.** *A* $\mathsf{TauberianRemainder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, H, T, D, R, E, K, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, displaying the finite Abel transform or summation-by-parts window that supplies the boundary estimate.*
2. *H is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *row, displaying the Cesaro averaging window against which the Abel boundary is compared.*
3. *T is the Tauberian side-condition row, recording the finite slow-decrease, bounded-remainder, or one-sided tail ledger admitted for the inspected window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder-budget row, assigning the inspected Abel and Cesaro windows a dyadic error allowance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after A, H, T, D have been displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular readback row.*
7. *K records componentwise* hsame *transport for Abel rows, Cesaro rows, Tauberian side conditions, dyadic budgets, regular readback, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a precision request through the Abel row, the Cesaro row, the side-condition ledger, the dyadic remainder budget, the regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{TauberianRemainder}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HardyLittlewoodTauberian}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a selected limiting value, quotient equality of series names, host real completeness, countable choice, trichotomy, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.TauberianRemainderUp.TauberianRemainderTasteGate single carrier alignment`

Theorem 1991.2 (TauberianRemainder NameCert obligations)**.** *For every accepted* $\mathsf{TauberianRemainder}^{\uparrow}$ *carrier $Q = (A, H, T, D, R, E, K, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}$ *surface consists exactly of carrier admission for that ten-row packet, Abel-window exposure through A, Cesaro-window exposure through H, Tauberian side-condition exposure through T, dyadic remainder-budget exposure through D, regular rational readback through R, real sealing through E, and transport, replay, provenance, and local naming through K, C, P, N.*

Proof. Project the accepted packet $Q = (A, H, T, D, R, E, K, C, P, N)$. Carrier admission exposes exactly the ten rows listed in Definition 1991.1; there is no hidden coordinate for an ambient limit, host equality, or unrestricted completeness principle.

Read the analytic rows in order. The Abel row A supplies the finite boundary estimate, the Cesaro row H supplies the averaging comparison window, and the side-condition row T records the only Tauberian hypothesis admitted by the packet.

The dyadic row D then fixes the visible remainder budget for those inspected rows. The regular readback R and seal E are downstream of A, H, T, D, so no accepted real read can bypass the finite remainder budget.

Finally, the transport row K and replay row C preserve only displayed packet coordinates, while P, N source and name the same carrier. These clauses exhaust the local $\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}$ surface. □

Theorem 1991.3 (TauberianRemainder Abel–Cesaro route)**.** *Every Abel–Cesaro comparison exported by an accepted* $\mathsf{TauberianRemainder}^{\uparrow}$ *carrier factors through*

$$A, H, T, D, K, C, P, N.$$

The route may compare the AbelSummation window with the HardyCesaroMean window only after reading the Tauberian side-condition row and the DyadicRatCore remainder budget attached to the same finite packet.

Proof. Begin with an Abel–Cesaro comparison request. The carrier has one Abel row A and one Cesaro row H, so any comparison between those faces starts on the displayed finite windows.

The side-condition row T is the only row that licenses a Tauberian comparison between the two windows. The dyadic row D is the only row that records the remainder budget for the inspected comparison.

The structural rows K, C, P, N transport, replay, source, and name that same finite route. Since the comparison does not yet consume R or E, it remains a boundary comparison rather than a selected real limit or a completed host equality. □

Theorem 1991.4 (TauberianRemainder real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{TauberianRemainder}^{\uparrow}$ *carrier factors through the full finite route*

$$A, H, T, D, R, E, K, C, P, N.$$

Consequently the real-facing endpoint cannot be obtained from a selected limit, quotient series equality, host real completeness, countable choice, trichotomy, or open-cover compactness; it is visible only through the displayed Abel, Cesaro, Tauberian side-condition, dyadic remainder, regular readback, and seal rows.

Lean4 ✓ `BEDC.Derived.TauberianRemainderUp.TauberianRemainderCarrier real seal nonescape`

Proof. Start with a $\mathsf{Real}^{\uparrow}$ consumer. By Definition 1991.1, the only real seal coordinate is E, and E is downstream of the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Trace R backward through the packet. It reads the Abel row A, Cesaro row H, side-condition row T, and dyadic remainder-budget row D before any regular rational readback is available.

The transport and continuation rows preserve exactly that dependency chain. Thus the consumer cannot replace the displayed route by a selected limit, quotient series equality, host real completeness, countable choice, trichotomy, or open-cover compactness. □

Falsifiable prediction: An accepted $\mathsf{TauberianRemainder}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the $\mathsf{AbelSummation}^{\uparrow}$ row, $\mathsf{HardyCesaroMean}^{\uparrow}$ row, Tauberian side-condition row, $\mathsf{DyadicRatCore}^{\uparrow}$ remainder budget, and $\mathsf{RegSeqRat}^{\uparrow}$ readback row refutes the carrier surface.

Independence witness: abel-summation, hardy-cesaro-mean, regseqrat, dyadicratcore, real, hardy-littlewood-tauberian: no listed sibling supplies the whole ten-row packet $Q = (A, H, T, D, R, E, K, C, P, N)$, because each sibling contributes only a summation route, averaging route, readback surface, budget ledger, seal face, or Tauberian handoff while this chapter binds the finite remainder boundary into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}$.

Constructive story: A $\mathsf{TauberianRemainder}^{\uparrow}$ packet is generated from an $\mathsf{AbelSummation}^{\uparrow}$ row, a $\mathsf{HardyCesaroMean}^{\uparrow}$ row, a Tauberian side-condition row, a $\mathsf{DyadicRatCore}^{\uparrow}$ remainder-budget row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1991.1, Definition 1991.2, Definition 1991.3, and Definition 1991.4 seed the finite Tauberian remainder route over AbelSummation, HardyCesaroMean, DyadicRatCore, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TauberianRemainder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a selected limit, quotient series equality, host real completeness, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface, Abel–Cesaro route theorem, real-seal nonescape theorem, and TasteGate field faithfulness for BEDC.Derived.TauberianRemainderUp.

Chapter 1992

A Concrete Naming Certificate for $\mathsf{CauchyRealOrderedField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealOrderedField}^{\uparrow}$ packet joins the $\mathsf{CauchyRealField}^{\uparrow}$ operation surface with the constructive comparison rows of $\mathsf{CauchyRealOrder}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, and $\mathsf{ArchimedeanOrderedField}^{\uparrow}$. It sits after the real-name seal of chapter 38, the regular rational source rows of chapter 301, the stream-window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the field-facing Cauchy real packet of chapter 1717, the constructive order packet of chapter 1716, the algebra/order comparison packet of chapter 766, and the Archimedean ordered-field comparison packet of chapter 1419. The packet gives Real and Completion consumers a finite ordered-field interface; it is not host real arithmetic, quotient equality, trichotomy, a selected-limit operation, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1992.1 (Cauchy real ordered field carrier)**.** *A* $\mathsf{CauchyRealOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, O, A, B, Z, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyRealField}^{\uparrow}$ *field row. It records the regular rational source family, stream windows, dyadic arithmetic ledger, addition, product, reciprocal, constructive order input, and field-facing real seal already named by the Cauchy real field packet.*
2. *O is the* $\mathsf{CauchyRealOrder}^{\uparrow}$ *constructive order and apartness row. It supplies the positive-gap route, finite refusal of positive apartness, and oriented comparison verdicts read by the ordered-field packet.*
3. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison row. It records the displayed addition, negation, multiplication, absolute-value, nonnegative-order, and strict-apartness reads needed to compare algebra and order data on the same real-name surface.*
4. *B is the* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *bound row. It records the rational-source, rational-embedding, and Archimedean comparison rows that locate the ordered-field interface inside finite rational bounds.*
5. *Z is the shared* $\mathsf{Real}^{\uparrow}$ *seal row. It is downstream of the field, order, algebra/order, and Archimedean rows rather than an independent completed-real carrier.*
6. *H records componentwise* hsame *transport for the field row, order row, algebra/order row, Archimedean row, real seal row, replay row, provenance row, and local naming row.*

7. *C records displayed* Cont *replay from each ordered-field request through F, O, A, B, Z, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealOrderedField}^{\uparrow}$, $\mathsf{CauchyRealField}^{\uparrow}$, $\mathsf{CauchyRealOrder}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealOrderedField}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive ordered-field packet. It has no coordinate for host real arithmetic, quotient equality, trichotomy, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRealOrderedFieldUp.CauchyRealOrderedFieldTasteGate single carrier alignment`

Theorem 1992.2 (Cauchy real ordered field NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealOrderedField}^{\uparrow}$ *carrier*

$$K = (F, O, A, B, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealOrderedField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Cauchy real field row, constructive order row, RealAlgOrder comparison row, Archimedean ordered-field bound row, shared real seal, transport, replay, provenance, and local naming row;*
2. *field projection, requiring every addition, product, reciprocal, zero, one, and negation read to pass through the displayed* $\mathsf{CauchyRealField}^{\uparrow}$ *row F;*
3. *order projection, requiring every positive-gap, apartness-refusal, and oriented comparison read to pass through the displayed* $\mathsf{CauchyRealOrder}^{\uparrow}$ *row O;*
4. *algebra/order compatibility, requiring the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row A to read the same real-name seal and dyadic comparison ledgers used by F and O;*
5. *Archimedean boundedness, requiring the* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *row B to supply only displayed rational-source, rational-embedding, and bound rows over the same field and order surface;*
6. *shared seal discipline, requiring Z to be downstream of F, O, A, B and never a separate completed-real object;*
7. *transport and replay, requiring H and C to preserve the field, order, comparison, bound, seal, provenance, and naming rows together;*
8. *ledger exhaustion, excluding host real arithmetic, quotient equality, trichotomy, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Cauchy real ordered-field obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1992.1. The projection exposes exactly the field row, order row, algebra/order comparison row, Archimedean bound row, shared seal row, and structural rows.

The field and order projections are inherited from the sibling Cauchy real surfaces. The row F supplies the finite operation route of chapter 1717, while the row O supplies the positive-gap, negative-apartness, and oriented-order routes of chapter 1716.

The comparison and bound projections are separate visible rows. The row A keeps algebraic and order reads inside the displayed $\mathsf{RealAlgOrder}^{\uparrow}$ endpoint, dyadic, finite-window, and apartness

ledgers, and the row B keeps rational-bound reads inside the displayed $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ rational-source and Archimedean comparison rows.

Finally, Z, H, C, P, N seal, transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for any excluded principle, none can become a local NameCert obligation. □

Theorem 1992.3 (Cauchy real ordered field order-field route)**.** *Every accepted* $\mathsf{CauchyRealOrderedField}^{\uparrow}$ *carrier routes ordered-field reads through the finite chain*

$$F, O \longmapsto A \longmapsto B \longmapsto Z \longmapsto H, C, P, N.$$

Thus the operation surface and the comparison surface are joined only by displayed Cauchy real field, Cauchy real order, RealAlgOrder, and Archimedean ordered-field rows.

Proof. Start from the accepted packet $K = (F, O, A, B, Z, H, C, P, N)$. The only operation coordinate is F, and the only constructive Cauchy comparison coordinate is O.

Read the algebra/order compatibility row next. The row A consumes the same displayed real-name, dyadic, finite-window, and apartness data exposed by the field and order rows, so it cannot introduce a second operation or comparison surface.

Read the Archimedean row after that. The row B contributes rational-source, rational-embedding, and bound evidence over the same real-name surface; it does not replace the field and order rows with an ambient ordered field.

The real seal and structural rows are downstream. The row Z is reached only after F, O, A, B have been displayed, and H, C, P, N preserve this dependency order for every ordered-field consumer. □

Theorem 1992.4 (Cauchy real ordered field non-escape)**.** *For every accepted* $\mathsf{CauchyRealOrderedField}^{\uparrow}$ *carrier* $K = (F, O, A, B, Z, H, C, P, N)$*, every Real or Completion consumer receives only the finite rows displayed by* K*. In particular, the ordered-field packet exports no host real arithmetic, quotient equality, trichotomy, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum, Lean verification, or bridge-checked standard structure.*

Proof. Inspect the carrier inventory. The only public rows are $F, O, A, B, Z, H, C, P, N$, and each of the first four rows is itself a named finite BEDC packet with visible source, ledger, comparison, and naming data.

A Real or Completion consumer may project field operations from F, constructive comparisons from O, algebra/order compatibility from A, or rational bounds from B. Each projection remains tied to the shared seal Z and the same transport and replay rows.

The structural rows do not add analytic content. They transport, replay, source, and name the displayed packet. Therefore no excluded principle has a coordinate, a replay path, or a provenance row through which it can escape to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealOrderedField}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealOrderedField}^{\uparrow}$ packet is generated from a $\mathsf{CauchyRealField}^{\uparrow}$ field row, a $\mathsf{CauchyRealOrder}^{\uparrow}$ constructive order row, a $\mathsf{RealAlgOrder}^{\uparrow}$ comparison row, an $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ rational-bound row, a shared $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealOrderedField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1992.1, Definition 1992.2, Definition 1992.3, and Definition 1992.4 seed the finite ordered-field route over the existing Cauchy real field, Cauchy real order, RealAlgOrder, Archimedean ordered-field, and shared real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealOrderedField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real arithmetic, quotient equality, trichotomy, selected limits, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked finite carrier theorem for BEDC.Derived.CauchyRealOrderedFieldUp binding the displayed field, order, algebra/order, Archimedean, real-seal, transport, replay, provenance, and local naming rows.

Chapter 1993

A Concrete Naming Certificate for $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$

Following Riemann, Darboux, and Bishop–Bridges constructive analysis, the $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$ packet names the finite Cauchy-refinement criterion for Riemann/Darboux integrability. It sits after the tagged-sum surface of chapter 1615, the upper-lower gap route of chapter 1460, the dyadic finite partition carrier of chapter 1863, and the regulated step-primitive handoff of chapter 1557. The packet turns refinement Cauchy control into a $\mathsf{Real}^{\uparrow}$-facing integral handoff without importing host integrability, quotient function equality, selected tags, measure completion, or ambient Real completeness.

Definition 1993.1 (Riemann integrability criterion carrier)**.** *A* $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, S, G, R, B, Q, E, H, C, V, N)$$

with the following displayed rows:

1. *P is the finite partition row, read through* $\mathsf{DyadicPartition}^{\uparrow}$*, naming the ordered cells, endpoints, boundary contact, and common-refinement interface.*
2. *S is the Riemann-sum row, read through* $\mathsf{RiemannSum}^{\uparrow}$*, exposing tagged sums only through displayed cells, tags, widths, values, and finite arithmetic.*
3. *G is the Darboux upper-lower gap row, read through* $\mathsf{DarbouxIntegral}^{\uparrow}$*, recording the visible upper sum, lower sum, and dyadic gap budget on a displayed common refinement.*
4. *R is the common-refinement row. It names the finite refinement that simultaneously transports the partition cells, tagged-sum cells, and Darboux upper/lower approximants before any Cauchy comparison is read.*
5. *B is the dyadic tolerance budget. It records the requested precision, the finite refinement depth, and the budget split used to compare all carried Riemann and Darboux rows inside the same dyadic interval.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row produced from the displayed refinement Cauchy data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal attached to the regular rational handoff, exposing the candidate integral only through the Cauchy-refinement budget.*

8. *H records componentwise* hsame *transport for partitions, Riemann sums, Darboux gaps, refinement rows, tolerance budgets,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, continuation rows, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from finite partitions to tagged sums, from tagged sums and upper/lower approximants to common refinement, from refinement gap control to the regular rational handoff, and from that handoff to the real seal.*
10. *V is the* Pkg *provenance over* $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, S, G, R, B, Q, E, H, C, V, N$. It has no coordinate for a host integrability predicate, quotient equality of integrands, a selected tag family, measure completion, or ambient Real-completeness data.

Lean4 ✓ `BEDC.Derived.RiemannIntegrabilityCriterionUp.RiemannIntegrabilityCriterionTasteGate`
`single carrier alignment`

Theorem 1993.2 (Riemann integrability criterion NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$ *carrier $K = (P, S, G, R, B, Q, E, H, C, V, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityCriteric}}$ *surface consists of carrier admission, finite partition exposure, Riemann-sum readback, Darboux upper/lower gap exposure, common-refinement stability, dyadic tolerance-budget exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged by the finite route*

$$P, S, G \longmapsto R \longmapsto B \longmapsto Q \longmapsto E$$

together with H, C, V, N.

Proof. Carrier admission is the projection of the tuple in Definition 1993.1. The partition row P, the Riemann-sum row S, and the Darboux gap row G are displayed before any limiting language is available to a consumer.

Common-refinement stability is obtained by finite refinement induction on R. The base case is the carried partition itself. In the refinement step, each new dyadic subcell records its parent cell, transports the tag and upper/lower approximation reads through H, and replays the finite sum and gap routes through C.

The dyadic tolerance-budget row B then compares the transported Riemann sums and Darboux upper/lower gaps on the same refined cells. Since the budget is finite, the comparison is a row of BHist arithmetic rather than a host quantification over all refinements.

The handoff rows Q and E are read only after the budget route has been displayed. Provenance V and the local name N source the packet, and the absence of host-integrability, quotient-function, selected-tag, measure-completion, and ambient-completeness coordinates gives ledger non-escape. □

Theorem 1993.3 (Cauchy refinement handoff)**.** *An accepted* $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$ *packet hands a Real-facing integration consumer only the displayed finite partition row, Riemann-sum row, Darboux gap row, common-refinement row, dyadic tolerance budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transports, displayed* Cont *replays,* Pkg *provenance, and local* NameCert *row. The handoff supplies the constructive Cauchy refinement criterion for the integral value, but it does not export host integrability, quotient equality of functions, selected tags, measure completion, or ambient Real completeness.*

Proof. Start from the obligation surface of Definition 1993.2. A consumer first receives the finite partition, Riemann-sum, and Darboux gap rows, with the sibling surfaces of chapter 1615, chapter 1460, and chapter 1863 fixing the finite reads.

The common-refinement row R is then consumed before any regular rational name is exposed. Finite refinement induction transports each relevant tagged-sum and upper/lower gap read to the shared dyadic cells, so the tolerance budget B compares rows that live in one displayed refinement ledger.

The row Q packages these budgeted comparisons as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and E seals the result as the Real-facing candidate integral. A $\mathsf{RegulatedIntegral}^{\uparrow}$ consumer may use the same finite step-primitive route of chapter 1557, but it receives only the rows listed in the carrier.

Thus the Cauchy refinement criterion reaches the Real boundary through finite BEDC data. The excluded analytic objects have no carrier coordinate and no continuation route, so they cannot be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}$ packet is generated from finite partition rows, Riemann-sum rows, Darboux upper/lower gap rows, a displayed common-refinement row, a dyadic tolerance budget, a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1993.1, Definition 1993.2, and Definition 1993.3 seed the constructive Cauchy-refinement criterion connecting finite Riemann/Darboux control to a Real-facing integral handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host integrability, quotient function equality, selected tag families, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, Riemann-sum readback, Darboux gap exposure, common-refinement stability, dyadic tolerance-budget exactness, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and non-escape for BEDC.Derived.RiemannIntegrabilityCriterionUp.

Chapter 1994

A Concrete Naming Certificate for $\mathsf{CauchySpeedup}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchySpeedup}^{\uparrow}$ packet records the finite route by which an arbitrary displayed Cauchy source is accelerated along a cofinal schedule before it is read as regular rational data. It sits after the regularization interface of chapter 376, the finite acceleration surfaces of chapter 1518 and chapter 381, the window interface of chapter 126, the rational readback of chapter 301, and the terminal real seal of chapter 38. The packet is a finite L10 speedup certificate; it is not quotient stream equality, a selected limit, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 1994.1 (Cauchy speedup carrier)**.** *A* $\mathsf{CauchySpeedup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, J, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source Cauchy row, recording the finite requests and source observations available before any speedup schedule is inspected.*
2. *J is the cofinal speedup schedule row, carrying the monotone tail-access route and the displayed evidence that every downstream tolerance request is eventually reached.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested dyadic scales, slack comparisons, and exact tolerance accounting used by the accelerated route.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the schedule J over the source row A.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same scheduled windows and dyadic ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, J, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source Cauchy row, cofinal schedule, dyadic ledger, finite windows, regular rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each Real-facing request through A, J, D, W, R, E and then through H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySpeedup}^{\uparrow}$, $\mathsf{CauchyRegularization}^{\uparrow}$, $\mathsf{FastCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySpeedup}^{\uparrow}}$ *row.*

The classifier compares accepted speedup packets only through $A, J, D, W, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean evidence, or standard-bridge data.*

Lean4 ✓ `BEDC.Derived.CauchySpeedupUp.CauchySpeedupTasteGate single carrier alignment`

Theorem 1994.2 (Cauchy speedup NameCert obligations)**.** *For every accepted* $\mathsf{CauchySpeedup}^{\uparrow}$ *carrier*

$$S = (A, J, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySpeedup}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the source Cauchy row A, cofinal speedup schedule J, dyadic tolerance ledger D, scheduled stream windows W, regular rational readback R, Real seal E, and structural rows H, C, P, N;*
2. *classifier transport, requiring every accepted replacement to preserve the same componentwise source, schedule, tolerance, window, readback, seal, replay, provenance, and naming rows through H and C;*
3. *cofinal schedule stability, requiring J to remain monotone and cofinal for the displayed tail requests over A under classifier transport;*
4. *dyadic tolerance exactness, requiring every accelerated tolerance claim to be read through the ledger D and the scheduled windows W;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be downstream of A, J, D, W and to use no unlisted stream coordinate;*
6. $\mathsf{Real}^{\uparrow}$ *non-escape, requiring E to be consumed only after the scheduled regular rational route has been displayed and to add no selected limit, quotient equality, trichotomy, choice principle, or ambient completeness claim;*
7. *ledger exactness, requiring H, C, P, N to transport, replay, source, and name exactly the displayed route A, J, D, W, R, E.*

No local speedup obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchySpeedupUp.CauchySpeedupNameCertObligations`

Proof. Project the carrier of Definition 1994.1. The analytic rows are the source Cauchy row A, the cofinal schedule J, the dyadic ledger D, the finite windows W, the regular rational readback R, and the terminal Real seal E.

Classifier transport is componentwise. The row H transports the same source, schedule, tolerance, window, readback, seal, replay, provenance, and naming coordinates, while C replays Real-facing requests through the same ordered route.

Cofinal schedule stability is the role of J. Since the carrier has no second schedule coordinate, every accelerated tail request must be read through the displayed schedule, and transport can only preserve that schedule through the corresponding hsame component.

Dyadic exactness is forced by D and W. The ledger stores the requested dyadic scales and slack comparisons, and the windows opened by J are exactly the windows available for the tolerance read.

The handoff to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ is downstream. The readback row R is built from A, J, D, W, the seal E is reached only after R, and the structural rows H, C, P, N only preserve and name that route. Because no excluded coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchySpeedupUp.TasteGate.CauchySpeedupTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySpeedup}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySpeedup}^{\uparrow}$ packet is generated from a source Cauchy row, a cofinal speedup schedule, a DyadicRatCore tolerance ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySpeedup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1994.1 and Definition 1994.2 seed the finite route from arbitrary displayed Cauchy data through cofinal acceleration into regular rational readback and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySpeedup}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchySpeedupUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, classifier transport, cofinal schedule stability, dyadic tolerance exactness, RegSeqRat handoff, RealUp non-escape, ledger exactness, provenance, and local naming for BEDC.Derived.CauchySpeedupUp.

Chapter 1995

A Concrete Naming Certificate for $\mathsf{HilbertCube}^{\uparrow}$

Following Hilbert's cube as treated in Kelley and Bourbaki, the $\mathsf{HilbertCube}^{\uparrow}$ packet records the countable product boundary as a finite BEDC naming certificate rather than as a completed host product. It consumes the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, the binary product-metric ledger of chapter 424, and the finite compact-metric product route of chapter 1822. The packet is the standard constructive test object: a countable coordinate schedule over real interval seals whose every consumer request is reduced to finitely many displayed coordinate windows.

Definition 1995.1 (Hilbert cube carrier)**.** *A* $\mathsf{HilbertCube}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, W, S, R, E, M, K, T, C, Q, N)$$

with the following displayed rows:

1. *I is the coordinate-window row. It records a finite set of inspected natural-number coordinates, the requested rational tolerances for those coordinates, and the interval bounds assigned to the inspected coordinates.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule. It names exactly the stream entries inspected at the coordinates listed by I and contains no row for an unrequested tail coordinate.*
3. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It reads the rational approximants and tolerance witnesses exposed by W for the displayed coordinate windows.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *interval seal row. For every coordinate named in I, it seals the associated regular rational readback into the displayed closed interval boundary for that coordinate.*
5. *E is the finite projection row. It packages the displayed coordinate windows as a finite product route, using the* $\mathsf{ProductMetric}^{\uparrow}$ *ledger for binary projection steps and the* $\mathsf{CompactMetricProduct}^{\uparrow}$ *route when a finite compact-metric product consumer asks for paired interval data.*
6. *M records componentwise* hsame *transport for the coordinate-window row, the stream-window schedule, the rational readback, the real interval seals, the finite projection row, replay, provenance, and local naming.*
7. *K records displayed* Cont *replay from a consumer request through*

$$I \longmapsto W \longmapsto S \longmapsto R \longmapsto E \longmapsto M, Q, N.$$

8. *T is the tail-boundary row. It records that coordinates not named by I are invisible to the current consumer read and contribute only the displayed interval-bound promise, not a selected coordinate value.*

9. *Q is the* Pkg *provenance over* $\mathsf{HilbertCube}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, W, S, R, E, M, K, T, Q, N$. It has no coordinate for quotient streams, choice-selected coordinates, open-cover compactness, host function equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.HilbertCubeUp.HilbertCubeTasteGate single carrier alignment`

Theorem 1995.2 (Hilbert cube NameCert obligations)**.** *For every accepted* $\mathsf{HilbertCube}^{\uparrow}$ *carrier $H = (I, W, S, R, E, M, K, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the coordinate-window row, finite stream schedule, regular rational readback, real interval seals, finite projection row, tail boundary, transport, replay, provenance, and local naming;*

2. *coordinate-window exactness, requiring every displayed coordinate request to be read first from I and then through the matching* $\mathsf{StreamName}^{\uparrow}$ *window in W;*

3. *real interval sealing, requiring every coordinate value exposed to a consumer to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row S and the* $\mathsf{Real}^{\uparrow}$ *interval seal row R;*

4. *finite projection compatibility, requiring every product-metric or compact-metric product read to factor through the finite projection row E and the displayed structural rows M, K, Q, N;*

5. *boundary nonescape, requiring the packet to export no quotient stream, choice-selected coordinate, open-cover compactness argument, host function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No Hilbert cube obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 1995.1. The accepted packet displays the coordinate-window row I, the finite $\mathsf{StreamName}^{\uparrow}$ schedule W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback S, the $\mathsf{Real}^{\uparrow}$ interval seals R, the finite projection row E, the tail boundary T, and the structural rows M, K, Q, N.

Coordinate-window exactness follows from the order of the carrier. A consumer names only the coordinates listed in I, and the only stream observations available for those coordinates are the finite windows in W.

The real interval clause is downstream of this finite window. The row S reads rational approximants from the displayed windows, and R seals those readbacks as coordinate interval data before any product-metric or compact-metric product row is visible.

Projection compatibility is then a finite-product statement. The row E repacks only the displayed coordinate seals, using the $\mathsf{ProductMetric}^{\uparrow}$ and $\mathsf{CompactMetricProduct}^{\uparrow}$ routes already named in the provenance row.

Since the carrier has no coordinate for quotient streams, selected coordinates, open-cover compactness, host function equality, or ambient completeness, the local NameCert surface cannot store such an obligation. □

Theorem 1995.3 (Hilbert cube finite coordinate projection)**.** *Let $H = (I, W, S, R, E, M, K, T, Q, N)$ be an accepted* $\mathsf{HilbertCube}^{\uparrow}$ *carrier. Every consumer read of H factors through finitely many displayed coordinate windows:*

$$I \longmapsto W \longmapsto S \longmapsto R \longmapsto E \longmapsto M, K, Q, N.$$

In particular, a $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{CompactMetricProduct}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer can inspect only the coordinates named by I and cannot force a read of the unlisted countable tail.*

Proof. Begin with an arbitrary consumer read. The carrier exposes no coordinate data before I, so the read first names a finite coordinate-window request.

The request then moves through W and S. These rows contain only the finite stream windows and rational readbacks attached to the coordinates already listed by I.

The $\mathsf{Real}^{\uparrow}$ interval seals in R are built from those readbacks. A real-facing consumer therefore sees a finite family of sealed coordinate intervals rather than a completed countable sequence.

For product-facing consumers, the only subsequent route is E. That row repacks the finite sealed family through the product-metric and compact-metric product surfaces, while M, K, Q, N transport, replay, package, and name the same finite route.

The tail-boundary row T records noninspection of every unlisted coordinate. Hence the accepted consumer read factors through the displayed finite coordinate windows and cannot force an additional tail read. □

Theorem 1995.4 (Hilbert cube real-completion boundary)**.** *Every accepted* $\mathsf{HilbertCube}^{\uparrow}$ *carrier excludes quotient streams, choice-selected coordinates, open-cover compactness, host function equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from its public boundary. Its real-facing completion surface is only the finite route through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *interval seals, the finite* $\mathsf{ProductMetric}^{\uparrow}$ *or* $\mathsf{CompactMetricProduct}^{\uparrow}$ *projection row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}$ *row.*

Proof. Project the accepted carrier. The only real-facing rows are the finite stream-window schedule W, the regular rational readback S, and the interval seals R attached to the coordinate-window row I.

The finite product surface does not add a completed countable product. It reads only the row E, whose inputs are the displayed coordinate seals and whose provenance points to the existing $\mathsf{ProductMetric}^{\uparrow}$ and $\mathsf{CompactMetricProduct}^{\uparrow}$ finite routes.

Transport and replay preserve this boundary componentwise. The rows M and K move the listed coordinates, readbacks, interval seals, finite projection row, and tail boundary without introducing a second equality or completion coordinate.

The provenance and local naming rows package exactly this finite route. Because quotient streams, selected coordinates, open-cover compactness, host function equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from $I, W, S, R, E, M, K, T, Q, N$, no public consumer can read them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1995.1, Definition 1995.2, Definition 1995.3, and Definition 1995.4 seed the finite coordinate-window boundary for the Hilbert cube over StreamName, RegSeqRat, Real interval seals, product metrics, compact-metric finite products, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertCube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, choice-selected coordinates, open-cover compactness, host function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coordinate-window exactness, real interval sealing, finite projection compatibility, boundary nonescape, consumer factorization, transport, replay, provenance, and local naming for BEDC.Derived.HilbertCubeUp.

Constructive story: A HilbertCube$^{\uparrow}$ packet is generated from a finite coordinate-window row, StreamName finite-window schedule, RegSeqRat readback, RealUp interval seals, ProductMetric or CompactMetricProduct finite projection rows, a tail boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 1996

A Concrete Naming Certificate for $\mathsf{EquicontinuousCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Bourbaki–Kelley language of uniform completion, the $\mathsf{EquicontinuousCompletion}^{\uparrow}$ packet records the finite route by which an equicontinuous family remains readable after completion handoff. It sits after the finite shared-modulus surface of chapter 413, the Cauchy filter and directed-net completion boundary of chapter 1473, the complete metric row of chapter 128, and the Cauchy-sequence carrier of chapter 406. Its L10 readback is routed through the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite NameCert surface for equicontinuous completion data; it is not Arzela–Ascoli compactness, selected limits, quotient completion, host function extensionality, countable choice, or ambient completeness.

Definition 1996.1 (EquicontinuousCompletion carrier)**.** *An* $\mathsf{EquicontinuousCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, U, M, S, W, R, A, T, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row displaying the shared source and target metrics, finite family ledger, finite probe centers, rational tolerance histories, and common modulus rows.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row through which completion-facing consumers read Cauchy-filter, directed-net, universal-extension, and completion-classifier data.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row supplying the finite completeness boundary used by the completion handoff, without adding a selected limit.*
4. *S is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *row recording the sequence-side Cauchy names consumed by the same completion request.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for every displayed source, target, tolerance, modulus, and completion read.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and used for all rational precision comparisons exposed by the packet.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after E, U, M, S, W, R have been displayed.*
8. *T is the route ledger from shared equicontinuity data through completion handoff and then through finite StreamName–RegSeqRat–Real readback.*

9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from each completion-facing family request through $E, U, M, S, W, R, A, T, H, P, N$*
11. *P is the* Pkg *provenance over* $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $E, U, M, S, W, R, A, T, H, C, P, N$. It has no coordinate for open-cover compactness, function-space compactness, selected completed functions, quotient equality of completion representatives, host function extensionality, countable choice, or ambient completeness.

Theorem 1996.2 (EquicontinuousCompletion NameCert obligation surface)**.** *For every accepted* $\mathsf{EquicontinuousCompletion}^{\uparrow}$ *carrier*

$$Q = (E, U, M, S, W, R, A, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuousCompletion}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier exposure, requiring the displayed packet $E, U, M, S, W, R, A, T, H, C, P, N$ to be the whole public carrier;*
2. *common-modulus classifier stability, requiring carrier comparison to preserve the shared* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *source, target, finite family, probe-center, rational-tolerance, and common-modulus rows componentwise;*
3. *finite-window exactness, requiring every source, target, tolerance, modulus, Cauchy-sequence, completion, and real-seal read to occur through the stored* $\mathsf{StreamName}^{\uparrow}$ *window family W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R;*
4. *completion handoff, requiring completion-facing consumers to pass from the equicontinuity row E through the displayed* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *rows U, M, S before the terminal* $\mathsf{Real}^{\uparrow}$ *seal A is exported;*
5. *ledger non-escape, requiring T, H, C, P, N to transport, replay, source, and name only the displayed rows and never an Arzela–Ascoli compactness principle, selected limit, quotient completion, host function equality, countable-choice schedule, or ambient completeness theorem.*

No EquicontinuousCompletion obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.EquicontinuousCompletionUp.EquicontinuousCompletionTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 1996.1. The accepted packet displays the equicontinuity-family row, uniform-completion row, complete-metric row, Cauchy-sequence row, finite stream windows, regular rational readback, terminal Real seal, route ledger, transport, replay, provenance, and local name row.

Common-modulus classifier stability is read from E and H. The source and target metrics, finite family ledger, finite probe centers, rational tolerance histories, and common modulus rows are compared componentwise, so a completion-facing comparison cannot replace the shared modulus surface by a chosen external modulus.

Finite-window exactness is read from W and R. Every inspected family value, tolerance request, radius response, Cauchy-sequence observation, completion witness, and Real seal comparison is

routed through the displayed window family and regular rational readback before any terminal seal is available.

The completion handoff is the ordered route through U, M, S, A. A consumer first sees the equicontinuity-family row E, then the uniform-completion, complete-metric, and Cauchy-sequence rows that mediate completion data, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally T, H, C, P, N preserve this same route under transport, replay, provenance, and naming. Because the carrier has no coordinates for Arzela–Ascoli compactness, selected limits, quotient completion, host function equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuousCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EquicontinuousCompletion}^{\uparrow}$ packet is generated from an EquicontinuityFamily row, UniformCompletion row, CompleteMetric row, CauchySequenceSpace row, finite StreamName windows, RegSeqRat readback, terminal RealUp seal, route ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1996.1 and Definition 1996.2 seed the finite common-modulus completion route over EquicontinuityFamily, UniformCompletion, CompleteMetric, CauchySequenceSpace, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuousCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Arzela–Ascoli compactness, function-space compactness, selected completed functions, quotient completion equality, host function extensionality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, common-modulus classifier stability, finite-window exactness, completion handoff, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.EquicontinuousCo

Chapter 1997

A Concrete Naming Certificate for MonotoneSubsequenceTheorem$^{\uparrow}$

Following Erdős–Szekeres and Bishop–Bridges constructive analysis, the MonotoneSubsequenceTheorem$^{\uparrow}$ packet records the finite selector-theorem surface used by downstream compactness routes. It stores source sequence windows, a peak-valley or monotone-branch selector ledger, cofinality rows, selected monotone windows, dyadic tolerance rows, a RegSeqRat$^{\uparrow}$ handoff, a Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert data. It is a displayed certificate surface for finite BEDC consumers, not a countable-choice principle or an open-cover compactness theorem.

Definition 1997.1 (Monotone subsequence theorem carrier)**.** *A* MonotoneSubsequenceTheorem$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, B, Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence-window ledger. Each row records the inspected finite source window and the comparison cells visible to the theorem surface.*
2. *B is the peak-valley or monotone-branch selector ledger. A row either records a displayed peak-valley split or records the finite monotone branch selected from the inspected window.*
3. *Q is the cofinality evidence for the inspected selector thresholds, limited to the thresholds that occur in the packet.*
4. *W is the selected monotone-window family read from S, B, Q through the finite* MonotoneSubsequence$^{\uparrow}$ *route of chapter 1405.*
5. *D is the dyadic tolerance ledger attached to the selected windows before any regular-rational or real-facing handoff is available.*
6. *R is the* RegSeqRat$^{\uparrow}$ *handoff row for the selected monotone windows.*
7. *E is the* Real$^{\uparrow}$ *seal row exposed only after the selected-window handoff has passed through D and R.*
8. *H stores componentwise* hsame *transport for source windows, branch rows, cofinality rows, selected monotone windows, dyadic tolerance rows, the regular handoff, the Real seal, continuation rows, provenance rows, and local naming rows.*

9. *C stores displayed* Cont *replay from S, B, Q to W, from W to D, from D to R, and from R to E.*
10. *P is the* Pkg *provenance over* MonotoneSubsequenceTheorem$^{\uparrow}$, MonotoneSubsequence$^{\uparrow}$, BolzanoWeierstrassSelecto BolzanoWeierstrass$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MonotoneSubsequenceTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for countable choice, arbitrary infinite subsequence selection, quotient stream equality, host real equality, open-cover compactness, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneSubsequenceTheoremUp.TasteGate.MonotoneSubsequenceTheoremTasteGate single carrier alignment`

Theorem 1997.2 (Monotone subsequence theorem NameCert obligations)**.** *For every accepted* MonotoneSubsequenceTheorem$^{\uparrow}$ *carrier*

$$T = (S, B, Q, W, D, R, E, H, C, P, N),$$

the local NameCert$_{\text{MonotoneSubsequenceTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source windows, branch ledger, cofinality rows, selected monotone windows, dyadic rows, regular handoff, seal row, transport, replay, provenance, and local name;*
2. *branch stability for the peak-valley or monotone-branch selector ledger under the recorded* hsame *transports;*
3. *cofinal coverage for every inspected selector threshold listed in Q;*
4. *selected-window handoff through* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *before the* Real$^{\uparrow}$ *seal row is consumed;*
5. *non-escape from countable choice, arbitrary infinite subsequence selection, quotient stream equality, host real equality, open-cover compactness, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 1997.1. The rows $S, B, Q, W, D, R, E, H, C, P, N$ are the only available coordinates, so carrier admission is exactly admission of those displayed rows.

The selector-theorem content is finite. Branch stability is read from the peak-valley or monotone-branch ledger B, and cofinal coverage is read from the inspected-threshold ledger Q. The actual selected windows then route through the finite MonotoneSubsequence$^{\uparrow}$ packet of chapter 1405.

The analytic handoff uses the existing downstream interfaces. Selected windows are consumed through the StreamName$^{\uparrow}$ surface of chapter 126, dyadic tolerance rows feed the RegSeqRat$^{\uparrow}$ handoff of chapter 301, and only then does the terminal Real$^{\uparrow}$ boundary of chapter 38 become visible.

Finally inspect the excluded data. The carrier has no row for countable choice, arbitrary infinite subsequence selection, quotient stream equality, host real equality, open-cover compactness, or ambient completeness, and the structural rows only transport, replay, source, and name the displayed packet. Hence the five listed obligations exhaust the local NameCert surface. □

Theorem 1997.3 (Monotone subsequence theorem Bolzano–Weierstrass handoff)**.** *Every accepted* MonotoneSubsequenceTheorem$^{\uparrow}$ *packet exports to Bolzano–Weierstrass consumers only the finite route*

$$S, B, Q \longmapsto W \longmapsto D \longmapsto R \longmapsto E.$$

The handoff factors through chapter 1405, chapter 1408, and chapter 1254; its window, regular-rational, and Real endpoints are read through chapter 126, chapter 301, and chapter 38.

Proof. Start with the selector route. The source windows S, branch ledger B, and inspected cofinality rows Q determine the selected monotone windows W only through the finite selector packet recorded in chapter 1405.

Next pass to the Bolzano–Weierstrass selector boundary. The selector consumer of chapter 1408 receives the selected monotone windows together with their dyadic tolerance and regular handoff rows, not an ambient subsequence object.

The Bolzano–Weierstrass chapter of chapter 1254 can therefore consume only the displayed route. The windows are read through chapter 126, the selected regular-rational handoff is read through chapter 301, and the terminal seal is read through chapter 38.

Since the route has no row for countable choice, arbitrary infinite subsequence selection, quotient stream equality, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Bolzano–Weierstrass-facing use remains inside the finite selector-certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneSubsequenceTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequenceTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1997.1, Definition 1997.2, and Definition 1997.3 seed the finite selector-theorem carrier, its local NameCert obligations, and its Bolzano–Weierstrass handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneSubsequenceTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, arbitrary infinite subsequence selection, quotient stream equality, host real equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch stability, inspected-threshold cofinal coverage, selected-window RegSeqRat handoff, and Bolzano–Weierstrass handoff non-escape for BEDC.Derived.MonotoneSubsequenceTheoremU

Constructive story: A $\mathsf{MonotoneSubsequenceTheorem}^{\uparrow}$ packet is generated from source sequence windows, a peak-valley or monotone-branch selector ledger, cofinality rows, selected monotone windows, dyadic tolerance rows, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1998

A Concrete Naming Certificate for $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is replaced by a cofinal subsequence without changing the real-name seal visible to downstream consumers. It sits over the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance discipline of chapter 291, and the terminal real-name boundary of chapter 38. It also refines the subsequence route of chapter 381 through the explicit cofinal selector ledger of chapter 405.

Definition 1998.1 (RegularCauchyCofinalSubsequence carrier)**.** *A* $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, W, \phi, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular Cauchy rational observations are being reindexed.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the inspected source windows and selected cofinal windows.*
3. *ϕ is a unary cofinal index map row; it records the selected source index for each displayed request, monotonicity on the inspected requests, and cofinality witnesses for the inspected thresholds.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic radii, endpoint observations, comparison cells, and slack rows before and after the selected index map is applied.*
5. *R is the pulled-back* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and their transported dyadic tolerance rows.*
6. *E is the inherited* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after S, W, ϕ, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source row, observation schedule, cofinal map, tolerance ledger, regularity readback, real seal, replay, package, and local name rows.*
8. *C records displayed* Cont *replay from a subsequence request through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CofinalSubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, \phi, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a host subsequence function, selected ambient limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 1998.2 (RegularCauchyCofinalSubsequence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ *carrier*

$$K = (S, W, \phi, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, cofinal index map, dyadic tolerance ledger, pulled-back regularity row, inherited* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local name;*
2. *cofinal schedule stability, requiring ϕ to be unary, monotone on displayed requests, and cofinal on every inspected threshold before any selected window is consumed;*
3. *regularity pullback along the cofinal map, requiring the selected* $\mathsf{StreamName}^{\uparrow}$ *windows and their* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows to feed the pulled-back* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. $\mathsf{Real}^{\uparrow}$ *seal compatibility, requiring E to be read only after the source, selected schedule, cofinal map, tolerance ledger, and regularity pullback have been displayed;*
5. *ledger non-escape, requiring H, C, P, N to exhaust the certificate route and to export no quotient stream, host subsequence function, selected ambient limit, countable choice, ambient completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCofinalSubsequenceUp.RegularCauchyCofinalSubsequenceTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 1998.1. Its displayed rows are exactly the regular rational source, observation schedule, cofinal index map, tolerance ledger, pulled-back regularity row, seal row, and structural provenance.

Cofinal schedule stability is read from the ϕ coordinate. Since ϕ is the only row that names selected indices, every selected window in W must be reached through its displayed monotone and cofinal checks.

Regularity pullback is then forced by the order of the rows. The dyadic ledger D attaches the selected tolerances to those windows, and the readback row R consumes exactly those selected windows and tolerance rows before any real-facing read can occur.

Seal compatibility follows because E is downstream of S, W, ϕ, D, R. The componentwise hsame rows and displayed Cont replay preserve that order, while the package and local name rows cite only the listed dependencies.

Thus the five obligations exhaust the finite carrier route. With no coordinate for quotient streams, host subsequence functions, selected ambient limits, countable choice, or ambient completeness, no excluded datum can be exported. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ real-facing read factors through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ schedule, cofinal index map, dyadic

tolerance ledger, pulled-back regularity row, and inherited $\mathsf{Real}^{\uparrow}$ seal before any consumer can inspect the seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 1998.1 and Definition 1998.2 seed the finite cofinal-subsequence route for regular Cauchy rational names over displayed StreamName windows, dyadic tolerance ledgers, RegSeqRat pullback, and the inherited RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host subsequence functions, selected ambient limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal schedule stability, regularity pullback, RealUp seal compatibility, and ledger non-escape for BEDC.Derived.RegularCauchyCofinalSubsequenceUp.

Constructive story: A $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName observation schedule, unary cofinal index map, DyadicRatCore tolerance ledger, pulled-back regularity row, inherited RealUp seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 1999

A Concrete Naming Certificate for $\mathsf{GaugeIntegral}^{\uparrow}$

Following Henstock, Kurzweil, and Bishop–Bridges constructive analysis, the $\mathsf{GaugeIntegral}^{\uparrow}$ packet names gauge-partition integration by finite tagged subdivisions, visible sum rows, Darboux gap comparison, regulated-integral compatibility, and Real/Completion handoff data. It consumes the gauge-subdivision surface of chapter 1861, the tagged-mesh sum boundary of chapter 1526, the upper-lower gap route of chapter 1460, the finite step primitive route of chapter 1557, and the Real/Completion seal rows used throughout the L10 substrate. The packet is a finite naming surface for one displayed interval and integrand window; it is not general open-cover compactness, a selected family of partitions, a quotient of functions, measure completion, or ambient real completeness.

Definition 1999.1 (Gauge integral carrier)**.** *A* $\mathsf{GaugeIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, \Gamma, T, S, D, Q, R, H, C, P, N)$$

with the following displayed rows.

1. *I is the interval row, naming the endpoints, endpoint order, and the cell boundary data exposed to every downstream partition and sum read.*
2. *Γ is the positive gauge row. Each displayed tag request carries a local positive radius and the finite observation window by which that radius is read.*
3. *T is the Cousin tagged-subdivision row, routed through chapter 1861 from I and Γ.*
4. *S is the Riemann tagged-sum row, routed through chapter 1526 from the same tags, cells, integrand reads, and mesh data displayed by T.*
5. *D is the Darboux gap row, routed through chapter 1460 by comparing upper and lower step bounds against the same interval and integrand windows.*
6. *Q is the regulated-integral compatibility row, routed through chapter 1557 whenever the integrand window is carried by a displayed regulated-function approximation ledger.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Completion}^{\uparrow}$ *handoff row reached only after the gauge, Cousin, Riemann, Darboux, and regulated-compatibility rows have been recorded.*
8. *H records componentwise* hsame *transport for the interval, gauge, tagged subdivision, tagged sum, Darboux gap, regulated compatibility, Real/Completion handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a gauge-integral request through*

$$I, \Gamma \longmapsto T \longmapsto S \longmapsto D \longmapsto Q \longmapsto R \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaugeIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, \Gamma, T, S, D, Q, R, H, C, P, N$. It has no coordinate for open-cover compactness, choice-selected partitions, quotient functions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.GaugeIntegralUp.GaugeIntegralTasteGate single carrier alignment`

Theorem 1999.2 (Gauge integral NameCert obligations)**.** *For every accepted* $\mathsf{GaugeIntegral}^{\uparrow}$ *carrier*

$$G = (I, \Gamma, T, S, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GaugeIntegral}^{\uparrow}}$ *surface consists exactly of the following obligation rows.*

1. *carrier admission for the displayed interval, positive gauge, Cousin subdivision, Riemann tagged sum, Darboux gap, regulated compatibility, Real/Completion handoff, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *gauge locality, requiring every tag row in T to read its radius from Γ and to certify that the tagged cell lies inside the corresponding local gauge window;*
3. *tagged-subdivision exactness through* $\mathsf{CousinLemma}^{\uparrow}$*, requiring T to be the finite output of chapter 1861 over the same I and Γ rows;*
4. *Darboux/Riemann compatibility, requiring the tagged-sum row S and the gap row D to inspect the same cells, endpoint comparisons, integrand reads, and step-bound data;*
5. *regulated-integral compatibility, requiring Q to consume only the finite step approximation and error ledger exposed by chapter 1557;*
6. *Real/Completion non-escape, requiring R to expose* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Completion}^{\uparrow}$ *data only after T, S, D, Q are present;*
7. *structural faithfulness, requiring H, C, P, N to transport, replay, source, and name no open-cover compactness theorem, choice-selected partition family, quotient of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 1999.1. The accepted packet displays the interval, gauge, subdivision, sum, gap, compatibility, handoff, transport, replay, provenance, and naming rows.

Gauge locality and subdivision exactness are read from Γ and T. The $\mathsf{CousinLemma}^{\uparrow}$ route supplies the finite tagged partition over the displayed interval and positive gauge, so no selected partition family or general compactness theorem is available to the classifier.

The Riemann and Darboux rows are then compared as finite ledger coordinates. The tagged-sum row S and gap row D share the interval cells, endpoint comparisons, integrand reads, and step-bound data, so their compatibility is a row equality inside the packet rather than a host theorem about all refinements.

The regulated row Q can be consumed only after the regulated-integral approximation ledger is displayed. The Real/Completion row R is downstream of T, S, D, Q, and the structural rows H, C, P, N replay and name exactly that route. Since the excluded principles have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 1999.3 (Gauge integral Cousin–Riemann handoff)**.** *Let* $G = (I, \Gamma, T, S, D, Q, R, H, C, P, N)$ *be an accepted* $\mathsf{GaugeIntegral}^{\uparrow}$ *carrier. Every consumer read that reaches the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *handoff row* R *first routes the gauge partition through chapter 1861, then the tagged-sum row through chapter 1526, then the gap comparison through chapter 1460, and then the step-compatible route through chapter 1557. The handoff order is*

$$I, \Gamma \longmapsto T \longmapsto S \longmapsto D \longmapsto Q \longmapsto R.$$

No accepted packet exposes a $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *read before this finite route is present.*

Lean4 ✓ `BEDC.Derived.GaugeIntegralUp.GaugeIntegralCousinRiemannHandoff`

Proof. Start with a consumer read of R. By Definition 1999.1, the row R is not an independent coordinate: it is specified as the handoff row reached after the gauge, Cousin, Riemann, Darboux, and regulated-compatibility rows have been recorded.

The first non-structural rows are I and Γ. They display the interval and positive gauge. The row T is then the Cousin tagged-subdivision output over those same rows, so every later sum sees a finite tagged partition already certified by chapter 1861.

The row S reads this tagged subdivision through the Riemann tagged-sum surface. The row D compares the same cells and integrand windows through the Darboux gap boundary, and Q reads the regulated step approximation route when such a regulated ledger is present.

Only after these rows are present can the continuation row replay a handoff to R. Componentwise hsame transport and Pkg provenance preserve the same order, and the local NameCert row names no independent Real or Completion source. Hence every accepted Real/Completion read factors through the displayed Cousin–Riemann–Darboux–regulated route. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaugeIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{GaugeIntegral}^{\uparrow}$ packet is generated from a displayed interval row, positive gauge row, Cousin tagged-subdivision row, Riemann tagged-sum row, Darboux gap row, regulated-integral compatibility row, RegSeqRat/Real/Completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GaugeIntegral}^{\uparrow}}$, seed)

Scope closed: Definition 1999.1, Definition 1999.2, and Definition 1999.3 seed the finite gauge-partition integration route over Cousin, Riemann, Darboux, regulated-integral, and Real/Completion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GaugeIntegral}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.GaugeIntegralUp.GaugeIntegralTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected partitions, quotient functions, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge locality, Cousin tagged-subdivision exactness, Darboux/Riemann compatibility, regulated-integral compatibility, Real/Completion non-escape, and field faithfulness for BEDC.Derived.GaugeIntegralUp.

Chapter 2000

A Concrete Naming Certificate for $\mathsf{ProductTopology}^{\uparrow}$

Following Kelley and Munkres on product topologies, the $\mathsf{ProductTopology}^{\uparrow}$ packet records the finite BEDC route by which two displayed topological carriers are paired and read through projection-continuity rows. It sits after the indexed-open surface of chapter 88, the binary product carrier of chapter 34, and the graph-and-modulus interface of chapter 130. The packet names only the finite binary product-topology certificate needed by later product, compact-metric, and continuous-map consumers; it is not a box topology, quotient topology, host powerset construction, selected open family, choice principle, or ambient set-theoretic product.

Definition 2000.1 (ProductTopology carrier)**.** *A* $\mathsf{ProductTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, O_X, O_Y, \Pi, \pi_X, \pi_Y, B, T, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two component carrier histories whose classifiers are transported only through displayed* hsame *rows.*
2. *O_X and O_Y are the two* $\mathsf{Topology}^{\uparrow}$ *indexed-open rows, exposing component open indices, component membership reads, and finite intersection ledgers.*
3. *Π is the* $\mathsf{Prod}^{\uparrow}$ *pairing row whose first and second projection readbacks expose paired histories only through the two component rows.*
4. *π_X and π_Y are the two* $\mathsf{ContinuousMap}^{\uparrow}$ *projection rows from the product packet to the component topology rows.*
5. *B is the rectangular subbasis row, recording finite pairs of component-open indices and their paired membership route through Π, O_X, O_Y.*
6. *T is the generated* $\mathsf{Topology}^{\uparrow}$ *row for the product carrier, admitting exactly the finite generated-open operations displayed by B and no host powerset of pairs.*
7. *H records componentwise* hsame *transport for component carriers, component topology rows, the* $\mathsf{Prod}^{\uparrow}$ *readback row, both projection-continuity rows, the rectangular subbasis row, the generated topology row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a product observation through pairing, component readback, rectangular subbasis membership, generated-open formation, projection continuity, transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{ProductTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ProductTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for box-topology opens, quotient topology, an external powerset of pairs, arbitrary host-indexed open families, choice-selected bases, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ProductTopologyUp.ProductTopologyTasteGate single carrier alignment`

Theorem 2000.2 (ProductTopology NameCert obligations)**.** *For every accepted* $\mathsf{ProductTopology}^{\uparrow}$ *carrier*

$$Q = (X, Y, O_X, O_Y, \Pi, \pi_X, \pi_Y, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProductTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for that thirteen-row packet, component-topology exposure through O_X, O_Y, product readback through Π, projection-continuity exposure through π_X, π_Y, rectangular subbasis generation through B, generated topology admission through T, and transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the accepted packet. The carrier displays exactly two component histories, two component topology rows, one product readback row, two projection-continuity rows, one rectangular subbasis row, one generated topology row, and the structural rows.

Read the topology rows before the product row. The component open indices and membership reads are supplied by O_X and O_Y, while Π supplies only the first and second component readbacks for paired histories.

The subbasis row B then binds a component-open pair to the corresponding product observation, and T records the generated-open operations admitted from that rectangular row. The projection rows π_X, π_Y are downstream records that the same paired readbacks are continuous into the component topology rows.

Finally, H, C, P, N transport, replay, source, and name that same finite route. Since box opens, quotient opens, host powersets, arbitrary host-indexed open families, selected bases, Lean proof terms, and bridge certificates are absent from the carrier inventory, the listed rows exhaust the local certificate surface. □

Theorem 2000.3 (ProductTopology projection continuity)**.** *Every projection-continuity read exported by an accepted* $\mathsf{ProductTopology}^{\uparrow}$ *carrier factors through*

$$\Pi, O_X, O_Y, B, T, H, C, P, N.$$

The first projection reads the first component through Π and O_X, and the second projection reads the second component through Π and O_Y; neither route can bypass the rectangular subbasis row or the generated $\mathsf{Topology}^{\uparrow}$ *row.*

Proof. Start with a projection-continuity request. The only product readback coordinate is Π, so the request must choose the first or second displayed projection route of that row.

On the first route, the component open evidence is read from O_X and the continuity row is π_X. On the second route, the component open evidence is read from O_Y and the continuity row is π_Y.

In both cases the rectangular subbasis row B records how the component-open observation is represented as a product-open observation, and T admits that row as part of the generated product topology. The structural rows H, C, P, N preserve the same route, so a projection consumer receives no host pair equality, box topology, quotient topology, or external powerset data. □

Theorem 2000.4 (ProductTopology universal route)**.** *Let a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *packet into an accepted* $\mathsf{ProductTopology}^{\uparrow}$ *carrier have component reads that factor through the two projection rows. Then its product-facing route is the finite ledger*

$$O_X, O_Y, \Pi, \pi_X, \pi_Y, B, T, H, C, P, N.$$

Thus the binary product-topology universal property is recorded only as a BEDC route through paired histories, component topology rows, projection-continuity rows, rectangular subbasis generation, and generated-open replay; it does not assert a host function-space theorem, a box-topology comparison, quotient topology, choice of arbitrary opens, or ambient set product.

Proof. Begin with the displayed map packet and its two component reads. Each component read is a $\mathsf{ContinuousMap}^{\uparrow}$ route into one of the topology rows O_X or O_Y, and the only product carrier available for pairing those reads is Π.

The projection rows π_X, π_Y identify the component reads as the two product projections. The rectangular subbasis row B is therefore the only row that can turn paired component-open observations into product-open observations.

The generated topology row T admits exactly the finite generated-open operations exposed by B. Transport and replay through H and C, together with provenance and local naming through P, N, repack the displayed component routes into the product-facing ledger.

No further universal-property coordinate is present. The carrier contains no host function-space theorem, box-topology comparison, quotient topology, choice-selected open family, or ambient set product, so the stated finite ledger is the entire exported route. □

Falsifiable prediction: An accepted $\mathsf{ProductTopology}^{\uparrow}$ consumer that obtains a projection-continuity read without first reading the $\mathsf{Prod}^{\uparrow}$ projection row, the corresponding $\mathsf{Topology}^{\uparrow}$ component row, the rectangular subbasis row, and the generated $\mathsf{Topology}^{\uparrow}$ row refutes the carrier surface.

Independence witness: topology, prod, continuousmap: no listed sibling supplies the whole thirteen-row packet $Q = (X, Y, O_X, O_Y, \Pi, \pi_X, \pi_Y, B, T, H, C, P, N)$, because Topology supplies indexed-open rows, Prod supplies paired readbacks, and ContinuousMap supplies projection-continuity rows, while this chapter binds those rows into one product-topology NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ProductTopology}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProductTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2000.1, Definition 2000.2, Definition 2000.3, and Definition 2000.4 seed the binary product-topology route over Topology, Prod, and ContinuousMap rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProductTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close box topology, quotient topology, host powersets, arbitrary host-indexed open families, selected bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, component topology exposure, product readback exactness, projection-continuity routing, rectangular subbasis generation, generated topology admission, and universal-route factorization for BEDC.Derived.ProductTopologyUp.

Constructive story: A $\mathsf{ProductTopology}^{\uparrow}$ packet is generated from two component carrier histories, two $\mathsf{Topology}^{\uparrow}$ indexed-open rows, a $\mathsf{Prod}^{\uparrow}$ pairing and projection row, two $\mathsf{ContinuousMap}^{\uparrow}$ projection-continuity rows, a rectangular subbasis row, a generated $\mathsf{Topology}^{\uparrow}$ row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2001

A Concrete Naming Certificate for $\mathsf{DyadicMetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicMetric}^{\uparrow}$ packet names the finite dyadic metric ledger read by regular rational streams, real-name seals, and metric-space consumers before any ambient completion can be invoked. It sits over the dyadic endpoint and distance rows of chapter 291, reads the nonnegative endpoint surface of chapter 293, hands scheduled distance ledgers to chapter 301, and presents the resulting finite witness surface to chapter 57. The chapter isolates the metric ledger itself: source pairs, dyadic distance, zero boundary, symmetry, triangle bounds, consumer handoff, transport, replay, provenance, and local naming.

Definition 2001.1 (Dyadic metric carrier)**.** *A* $\mathsf{DyadicMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (s, d, a, z, u, t, r, h, c, p, n)$$

with the following displayed rows:

1. *s is the source-pair row. It records two accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints together with their visible mantissa, denominator,* $\mathsf{Int}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *provenance.*
2. *d is the dyadic distance ledger for the ordered pair in s. It records the common-denominator subtraction window, radius row, and finite comparison data used by distance consumers.*
3. *a is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint row attached to d, exposing the nonnegative dyadic distance endpoint through the absolute-value packet rather than through a host absolute-value function.*
4. *z is the zero-distance boundary row. It states that a displayed zero distance is admitted only by the finite dyadic equality boundary already visible in the source-pair and distance rows.*
5. *u is the symmetry row. It swaps the two source endpoints and reuses the same absolute-value endpoint and tolerance ledger with the ordered-pair direction reversed.*
6. *t is the triangle-bound row. It records the finite three-source ledger, the two intermediate distance rows, the outer distance row, and the dyadic comparison witnessing the displayed triangle bound.*
7. *r is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer handoff row. It exposes the distance ledger as the finite tolerance surface read by regular rational streams, real-name seals, and metric witnesses.*
8. *h records componentwise* hsame *transport for the source pair, dyadic distance ledger, absolute-value endpoint, zero boundary, symmetry row, triangle-bound row, consumer handoff, replay, provenance, and local naming rows.*

9. *c records displayed* Cont *replay from the source pair through dyadic distance, absolute value, zero boundary, symmetry, triangle bound, consumer handoff, transport, provenance, and local naming.*

10. *p is the* Pkg *provenance over* $\mathsf{DyadicMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *n is the local* $\mathsf{NameCert}_{\mathsf{DyadicMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, d, a, z, u, t, r, h, c, p, n$. It has no coordinate for host metric spaces, quotient rationals, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicMetricUp.DyadicMetricTasteGate single carrier alignment`

Theorem 2001.2 (Dyadic metric NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMetric}^{\uparrow}$ *carrier $M = (s, d, a, z, u, t, r, h, c, p, n)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicMetric}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic distance-ledger exposure, absolute-value endpoint exposure, zero-distance boundary, symmetry routing, triangle-bound routing, RegSeqRat and Real consumer handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. No dyadic metric obligation is stored outside those finite rows.*

Proof. Carrier admission is projection from Definition 2001.1. The visible packet lists the source pair s, dyadic distance ledger d, absolute-value endpoint a, zero boundary z, symmetry row u, triangle-bound row t, consumer handoff r, and structural rows h, c, p, n.

The metric-law rows are finite ledger rows, not external laws. The zero boundary reads only the displayed dyadic equality boundary; the symmetry row swaps the source pair and reuses the same absolute-value endpoint; the triangle row reads three displayed sources, the corresponding distance ledgers, and one dyadic comparison row.

The handoff clause is downstream of those ledgers. A $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{Met}^{\uparrow}$ consumer can read the distance only through r after the source pair, distance ledger, absolute-value endpoint, and metric-law rows have already been displayed.

Transport and replay are componentwise through h and c. The provenance and local naming rows package exactly the listed coordinates, so the excluded host metric, quotient-rational, trichotomy, selected-limit, completeness, verification, and bridge data cannot appear as hidden obligations. □

Theorem 2001.3 (Dyadic metric RegSeqRat consumer handoff)**.** *Let $M = (s, d, a, z, u, t, r, h, c, p, n)$ be an accepted* $\mathsf{DyadicMetric}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Met}^{\uparrow}$ *consumer of M must read the finite route*

$$s \longmapsto d \longmapsto a \longmapsto z, u, t \longmapsto r \longmapsto h, c, p, n$$

before the dyadic distance ledger is available as a regular-rational tolerance, a real-name distance surface, or a metric witness. Consequently the consumer route cannot bypass the dyadic metric packet through a host metric space, quotient rational equality, trichotomy, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only consumer-facing coordinate is r, and r is placed after the source-pair row, dyadic distance ledger, absolute-value endpoint, zero boundary, symmetry row, and triangle-bound row.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, the distance ledger is read as a dyadic tolerance row. Since the tolerance comes from d and a, the regular rational route must pass through the displayed source pair and nonnegative endpoint before it can inspect a scheduled finite window.

For a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Met}^{\uparrow}$ consumer, the same route is terminal. The real-name seal and the metric witness receive only the distance surface exposed by r, while h, c, p, n transport, replay, package, and name that surface without adding a second distance function.

The excluded objects are absent from the carrier. Thus every accepted consumer route factors through the displayed dyadic metric ledger and cannot obtain an ambient host metric, quotient-rational comparison, trichotomy witness, selected limit, or completeness theorem from this packet. □

Theorem 2001.4 (Dyadic metric classifier transport)**.** *If two accepted* $\mathsf{DyadicMetric}^{\uparrow}$ *packets agree on the displayed source-pair, distance-ledger, absolute-value endpoint, zero-boundary, symmetry, triangle-bound,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows of Definition 2001.1, then every zero-distance, symmetry, triangle, and* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read is unchanged by passing from one packet to the other.*

Proof. Project both accepted carriers as

$$M = (s, d, a, z, u, t, r, h, c, p, n) \quad \text{and} \quad M' = (s', d', a', z', u', t', r', h', c', p', n').$$

The displayed component agreement identifies the source pair, dyadic distance ledger, absolute-value endpoint, zero boundary, symmetry row, triangle row, and consumer handoff row on the two sides.

Use the componentwise hsame row h for transport. Zero-distance reads use only s, d, a, z; symmetry reads use only s, d, a, u; triangle reads use only the three-source ledger inside t; and $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads use only r after the preceding rows have been displayed.

Finally read c, p, n. These rows replay, source, and name the same transported finite route. They carry no extra host metric, quotient rational, trichotomy, selected limit, or completeness coordinate, so classifier transport cannot change any listed consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMetric}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2001.1, Definition 2001.2, and Definition 2001.3 seed the finite dyadic metric route from source pairs through distance ledger, absolute-value endpoint, zero boundary, symmetry, triangle bound, RegSeqRat and Real handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host metric spaces, quotient rationals, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic distance-ledger exposure, absolute-value endpoint exposure, zero-distance boundary, symmetry routing, triangle-bound routing, consumer handoff, provenance, transport, replay, and local naming for BEDC.Derived.DyadicMetricUp.

Constructive story: A $\mathsf{DyadicMetric}^{\uparrow}$ packet is generated from a displayed pair of DyadicRatCore endpoints, a dyadic distance ledger, a DyadicAbsoluteValue endpoint, zero-distance, symmetry, and triangle rows, a RegSeqRat and Real consumer handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2002

A Concrete Naming Certificate for $\mathsf{CauchyCompletionEliminator}^{\uparrow}$

Following Bishop–Bridges constructive completion and the Mac Lane–Eilenberg–Moore vocabulary of recursion from an algebraic completion object, the $\mathsf{CauchyCompletionEliminator}^{\uparrow}$ packet records the BEDC eliminator interface for reading out of a Cauchy completion. It sits after the completion monad of chapter 378, the Kleisli consumer route of chapter 1845, the functorial action of chapter 1937, and the algebra boundary of chapter 1981. Its analytic input surface is the finite stream window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38.

The packet is an eliminator boundary, not a representative selector. It exposes exactly the finite rows needed to extend a dense-source action through regular Cauchy tails and to compare the resulting completed reads by tail agreement.

Definition 2002.1 (Cauchy completion eliminator carrier)**.** *A* $\mathsf{CauchyCompletionEliminator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, T, M, A, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source data row, displaying the source NameCert, its finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance observations before any completed read is inspected.*

2. *T is the target NameCert data row, including the codomain carrier surface, its accepted equality classifier, and the local naming row for the target of the eliminator.*

3. *M is the Cauchy-continuity modulus row, recording how a finite dyadic tolerance request on the target read is answered by a finite tail request on the dense source data.*

4. *A is the extension action row, sending an accepted regular Cauchy tail over D to a target packet over T by the modulus-controlled route M.*

5. *U is the uniqueness-by-tail-agreement ledger: two extension actions over the same dense-source action and the same modulus are compared by displayed* hsame *equivalence of their regular tails.*

6. *H records componentwise* hsame *transport for the dense source data, target NameCert data, Cauchy-continuity modulus, extension action, uniqueness ledger, replay row, provenance row, and local naming row.*

7. *C records displayed* Cont *replay of the route*

$$D, T, M \longmapsto A \longmapsto U \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionEliminator}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionEliminator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, M, A, U, H, C, P, N$. The carrier has no coordinate for quotient representatives, host equality of completed objects, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionEliminatorUp.TasteGate.CauchyCompletionEliminatorTasteGate`
`single carrier alignment`

Theorem 2002.2 (Cauchy completion eliminator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionEliminator}^{\uparrow}$ *carrier*

$$E = (D, T, M, A, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionEliminator}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the dense source data, target NameCert data, Cauchy-continuity modulus, extension action, uniqueness ledger, transport, replay, provenance, and local naming row;*
2. *dense-source action stability, requiring the action on displayed dense rows to be invariant under the carried* hsame *transport of* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations;*
3. *modulus-controlled extension, requiring every completed-route read to pass through the modulus row M before the extension action row A is inspected;*
4. *uniqueness through Cauchy tail agreement, requiring two completed reads over the same dense action and modulus to be compared only through the ledger U and displayed* hsame *equivalence of regular tails;*
5. *refusal of quotient representatives, host equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking as local obligation data.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2002.1. The accepted packet displays exactly the dense source data, target NameCert data, modulus, action, uniqueness ledger, and structural rows.

Dense-source stability is read from D and H. The dense action can inspect only finite stream windows, regular rational readback, and dyadic tolerance observations, and componentwise transport preserves those displayed rows before any target packet is named.

The extension clause is supplied by M and A. A completed-route consumer first presents a target tolerance request to the modulus row, receives a finite tail request on the dense source, and then applies the extension action to that displayed tail.

Uniqueness is supplied by U. When two accepted actions have the same dense-source action and the same modulus, the only comparison row available is tail agreement of the regular Cauchy presentations through displayed hsame routes.

The refusal clause is structural. Since no excluded object appears as a coordinate of E, the replay, provenance, and local naming rows cannot export quotient representatives, host equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking. □

Theorem 2002.3 (Cauchy completion eliminator uniqueness)**.** *Let*

$$E = (D, T, M, A, U, H, C, P, N) \quad \textit{and} \quad E' = (D', T', M', A', U', H', C', P', N')$$

be accepted $\mathsf{CauchyCompletionEliminator}^{\uparrow}$ *packets. If E and E' display the same dense-source action, the same Cauchy-continuity modulus, and componentwise* hsame *transport of the target NameCert rows, then the two eliminator packets are classifier-equal on every completed route whose regular tails are* hsame*-equivalent. The comparison is exhausted by the uniqueness ledgers U, U' and the displayed tail-agreement rows; it does not pass through quotient representatives, host equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project both accepted carriers. The hypotheses identify the dense-source action and the modulus rows, so any completed-route read issued to E and E' produces the same finite tail request on the dense source.

Now compare the regular tails. The assumed hsame equivalence of those tails is exactly the input consumed by the uniqueness ledgers U and U'. These ledgers compare the two target reads through the carried target NameCert transport, not through host equality of completed objects.

The structural rows finish the classifier comparison. Componentwise transport H, H', replay C, C', provenance P, P', and local names N, N' preserve the same dense action, modulus, and tail-agreement route. Since the carriers contain no excluded coordinate, the classifier equality cannot appeal to quotient representatives, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEliminator}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEliminator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2002.1, Definition 2002.2, and Definition 2002.3 seed the finite eliminator surface for Cauchy-completion reads out of regular tails.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEliminator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source action stability, modulus-controlled extension, uniqueness through Cauchy tail agreement, and non-escape proofs for BEDC.Derived.CauchyCompletionEliminatorUp.

Constructive story: A $\mathsf{CauchyCompletionEliminator}^{\uparrow}$ packet is generated from dense source data, target NameCert data, a Cauchy-continuity modulus, an extension action, a uniqueness-by-tail-agreement ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2003

A Concrete Naming Certificate for $\mathsf{SpernerLemma}^{\uparrow}$

Following Sperner, the $\mathsf{SpernerLemma}^{\uparrow}$ packet names the finite labeled-simplex ledger used by constructive dyadic compactness and fixed-point routes before any host geometric realization is consulted. It sits over the finite face inventory of chapter 238, the dyadic subdivision surface of chapter 1863, and the cover-routing rows of chapter 446. The chapter isolates the combinatorial certificate itself: simplex inventory, dyadic subdivision, boundary labels, face-label transport, fully-labeled simplex witness, structural transport, replay, provenance, and local naming.

Definition 2003.1 (Sperner lemma carrier)**.** *A* $\mathsf{SpernerLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (k, d, b, f, w, h, c, p, n)$$

with the following displayed rows:

1. *k is the finite simplicial-complex row. It records the accepted* $\mathsf{SimplicialComplex}^{\uparrow}$ *simplex inventory, face-incidence relation, dimension grading, and carried-face closure used by the labeled subdivision.*
2. *d is the dyadic subdivision row. It records the accepted* $\mathsf{DyadicPartition}^{\uparrow}$ *subdivision or partition ledger, including the dyadic cell indices and their incidence with the finite simplex row.*
3. *b is the boundary-label ledger. It records the boundary faces, their admissible colors, and the displayed exclusion of forbidden boundary colors.*
4. *f is the face-label transport row. It transports labels along the finite face relation of k and the dyadic subdivision incidence of d without using an ambient geometric simplex.*
5. *w is the fully-labeled simplex witness row. It records one carried subsimplex, its vertices, and the finite color list witnessing that each required color appears exactly through the displayed label ledger.*
6. *h records componentwise* hsame *transport for the simplicial-complex row, dyadic subdivision row, boundary-label ledger, face-label transport, fully-labeled witness, replay, provenance, and local naming rows.*
7. *c records displayed* Cont *replay from the finite simplex inventory through subdivision, boundary labels, face-label transport, fully-labeled witness, structural transport, provenance, and local naming.*

8. *p is the* Pkg *provenance over* $\mathsf{SpernerLemma}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *n is the local* $\mathsf{NameCert}_{\mathsf{SpernerLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $k, d, b, f, w, h, c, p, n$. It has no coordinate for host geometric realization, open-cover compactness, real trichotomy, selected fixed points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SpernerLemmaUp.SpernerLemmaTasteGate single carrier alignment`

Theorem 2003.2 (Sperner lemma NameCert obligations)**.** *For every accepted* $\mathsf{SpernerLemma}^{\uparrow}$ *carrier $S = (k, d, b, f, w, h, c, p, n)$, the local* $\mathsf{NameCert}_{\mathsf{SpernerLemma}^{\uparrow}}$ *surface consists exactly of the following five certificate rows:*

1. *the finite carrier row k, d combining the* $\mathsf{SimplicialComplex}^{\uparrow}$ *simplex inventory with the* $\mathsf{DyadicPartition}^{\uparrow}$ *subdivision ledger;*
2. *the boundary-coloring row b recording admissible boundary colors and forbidden-color exclusions;*
3. *the face-label transport row f moving labels through carried faces and dyadic incidence;*
4. *the fully-labeled simplex handoff row w exposing the finite witness to dyadic compactness and fixed-point consumers;*
5. *the structural row h, c, p, n for componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No Sperner obligation is stored outside these finite rows.

Proof. Carrier admission is projection from Definition 2003.1. The visible packet lists the finite simplicial-complex row k, the dyadic subdivision row d, the boundary-label ledger b, the face-label transport row f, the fully-labeled witness row w, and the structural rows h, c, p, n.

The first certificate row is the pair k, d. It fixes the finite simplex inventory and the dyadic subdivision surface before any labels are read.

The second and third rows are b and f. The boundary-coloring data is accepted only as a finite ledger, and the label transport is accepted only along the displayed face and subdivision incidence rows.

The fourth row is w. A compactness or fixed-point consumer receives a fully-labeled simplex only through this witness row after the carrier, boundary, and transport rows have been displayed.

The fifth row is structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local naming row package exactly the listed coordinates, so the excluded host-geometric, open-cover, trichotomy, selected-point, completeness, verification, and bridge data cannot appear as hidden obligations. □

Theorem 2003.3 (Sperner boundary-coloring route)**.** *Let $S = (k, d, b, f, w, h, c, p, n)$ be an accepted* $\mathsf{SpernerLemma}^{\uparrow}$ *carrier. Every read of the boundary-coloring ledger factors through the displayed* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite face row k, the* $\mathsf{DyadicPartition}^{\uparrow}$ *subdivision row d, and the* $\mathsf{DyadicCover}^{\uparrow}$ *cover-routing provenance recorded in p. In particular, boundary admissibility cannot be read directly from a host simplex, an open cover of a compact space, real trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The boundary-coloring coordinate is b, and the definition places b after the finite simplex row k and dyadic subdivision row d.

The simplicial row supplies the finite boundary faces and their face-incidence relation. The dyadic subdivision row supplies the finite cells whose boundary labels are being read. The package row p records the $\mathsf{DyadicCover}^{\uparrow}$ provenance that makes the label ledger visible to dyadic cover consumers.

Thus any boundary-coloring read reaches b only after the finite face inventory, the dyadic subdivision incidence, and the cover-routing provenance have been displayed. Since the classifier has no coordinate for host simplices, open-cover compactness, trichotomy, or ambient completeness, none of those objects can supply an alternate boundary-coloring route. □

Theorem 2003.4 (Sperner fully-labeled simplex handoff)**.** *Let $S = (k, d, b, f, w, h, c, p, n)$ be an accepted* $\mathsf{SpernerLemma}^{\uparrow}$ *carrier. Every fully-labeled simplex handoff exported to* $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ *or to a dyadic fixed-point consumer passes through the finite route*

$$k \longmapsto d \longmapsto b \longmapsto f \longmapsto w \longmapsto h, c, p, n.$$

Consequently the handoff cannot bypass the displayed finite ledger through host geometric realization, open-cover compactness, real trichotomy, selected fixed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only coordinate exposing a fully-labeled simplex is w, and w is downstream of the finite simplex inventory k, the dyadic subdivision row d, the boundary-label ledger b, and the face-label transport row f.

A $\mathsf{CompactIntervalFixedPoint}^{\uparrow}$ consumer therefore receives a labeled subsimplex only after the finite subdivision and boundary ledger have already been displayed. A dyadic fixed-point consumer reads the same witness through the package and replay rows, which transport the finite certificate without adding a selected point.

The excluded objects are absent from the classifier. Hence the exported handoff is exactly the displayed finite route and cannot obtain a host geometric realization, open-cover compactness theorem, real-trichotomy branch, selected fixed point, or ambient completeness witness from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpernerLemma}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpernerLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2003.1, Definition 2003.2, Definition 2003.3, and Definition 2003.4 seed the finite labeled-simplex route from simplicial-complex inventory through dyadic subdivision, boundary labels, face-label transport, fully-labeled witness, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpernerLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host geometric realization, open-cover compactness, classical Brouwer fixed points, real trichotomy, selected fixed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-coloring route, face-label transport, fully-labeled simplex handoff, provenance, transport, replay, and local naming for BEDC.Derived.SpernerLemmaUp.

Constructive story: A $\mathsf{SpernerLemma}^{\uparrow}$ packet is generated from a displayed finite simplicial-complex row, a dyadic subdivision or partition row, a boundary-label ledger, face-label transport rows, a fully-labeled simplex witness row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2004

A Concrete Naming Certificate for $\mathsf{SeparatedQuotientMetric}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Bourbaki–Kelley separated-reflection language, the $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ packet records the standard passage from a pseudometric surface to a separated metric surface as finite BEDC data. It starts with the pre-separated distance rows of chapter 1614, exposes the separated classifier route of chapter 1411, and hands only the routed classifier to the metric consumer of chapter 57. Its numerical readback is confined to the regular rational windows of chapter 301 and the terminal real-name seal of chapter 38. The packet therefore names the constructive zero-distance separation route without adding quotient primitives, host equality, trichotomy, countable choice, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2004.1 (SeparatedQuotientMetric carrier)**.** *A* $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, P, Z, S, R, E, H, C, N)$$

with the following displayed rows:

1. *X is the source point-history admission row. It records the carried point names and the classifier inherited from the source packet, but it does not identify point rows by host equality.*
2. *P is the source* $\mathsf{Pseudometric}^{\uparrow}$ *packet. It supplies the displayed distance witness, dyadic tolerance ledger, zero-distance boundary, stream windows, regular rational readback, and real-distance seal available before separation.*
3. *Z is the displayed zero-distance evidence row. It records finite requests showing that two admitted point rows have pseudometric distance zero through the rows of P.*
4. *S is the separated classifier output. It consumes Z and records the finite classifier evidence that the two source point histories have the same separated point reading.*
5. *R is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window handoff. It names the stream windows and regular rational approximants that a metric or real consumer may inspect before the separated classifier is read.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the pseudometric distance readback after the windows in R have been displayed.*
7. *H stores componentwise* hsame *transport for source point admission, pseudometric rows, zero-distance evidence, separated classifier output, finite windows, regular readback, real seal, continuation, provenance, and local naming rows.*

8. *C stores the displayed* Cont *replay from point admission through pseudometric distance inspection, zero-distance evidence, separated classifier output, finite-window readback, and real sealing.*
9. *N is both the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows and the local* $\mathsf{NameCert}_{\mathsf{SeparatedQuotientMetric}^{\uparrow}}$ *row.*

The carrier is a finite classifier certificate. It has no quotient-space coordinate, no host equality coordinate, no selected representative, no trichotomy row, no countable-choice selector, no selected limit, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SeparatedQuotientMetricUp.SeparatedQuotientMetricTasteGate single carrier alignment`

Theorem 2004.2 (SeparatedQuotientMetric NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ *carrier* $Q = (X, P, Z, S, R, E, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedQuotientMetric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source point row, source* $\mathsf{Pseudometric}^{\uparrow}$ *packet, zero-distance evidence, separated classifier output, finite-window and regular-rational handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *zero-distance classifier exposure, requiring every displayed zero-distance witness in Z to be consumed by the separated classifier row S before any point-identity surface is exported;*
3. *separated readback, requiring the metric-facing consumer to read only the classifier output in S and not an equivalence class or host equality proof;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, requiring finite windows and regular rational approximants in R to precede the real seal E for every distance read;*
5. *transport and replay stability, requiring H and C to move the source, zero-distance, classifier, readback, real-seal, provenance, and naming rows componentwise and in the displayed order;*
6. *provenance visibility, requiring N to name exactly the listed BEDC dependencies and the local NameCert row;*
7. *quotient non-escape, excluding quotient metric spaces, host equality, trichotomy, countable choice, selected representatives, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No separated-quotient metric NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2004.1. The packet displays precisely the source point row, source pseudometric packet, zero-distance evidence, separated classifier output, finite-window and regular-rational handoff, real seal, transport, replay, provenance, and local naming rows.

Zero-distance classifier exposure follows from the order P, Z, S. The pseudometric packet supplies the distance surface, the row Z records the zero-distance evidence, and the row S is the first coordinate that can export separated point-classifier evidence.

Separated readback is therefore classifier readback. A metric-facing consumer receives the output of S, not a quotient class, selected representative, or host equality proof.

The numerical handoff is read through R and E. Finite stream windows and regular rational approximants appear before the real seal, so a consumer cannot inspect a detached real distance without the displayed finite readback.

Transport and replay stability are the roles of H and C. They move the listed coordinates componentwise and replay them in the carrier order, preserving the route from pseudometric evidence to separated classifier output.

The provenance and non-escape clauses are inventory checks on N and on the fields of Q. Since no quotient metric space, host equality, trichotomy, choice selector, selected representative, selected limit, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean marker, or bridge row appears in the carrier, none can be part of the local NameCert surface. □

Theorem 2004.3 (SeparatedQuotientMetric zero-distance route)**.** *Let $Q = (X, P, Z, S, R, E, H, C, N)$ be an accepted* $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ *packet. Every consumer that reads point identity from pseudometric zero-distance evidence must factor through the displayed route*

$$\mathsf{Pseudometric}^{\uparrow} \to Z \to S \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{Met}^{\uparrow}.$$

In particular, every $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer sees the point-identity surface only after the zero-distance evidence has passed through the separated classifier S. The packet exports no alternative route through quotient metric spaces, host equality, trichotomy, countable choice, selected limits, selected representatives, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Start with an accepted packet Q. The source pseudometric row P exposes distance witnesses and zero-distance boundary data, while Z records the displayed zero-distance evidence for the pair of source point histories.

Before a point-identity surface can be read, the replay row C sends the evidence in Z through the separated classifier output S. This is the only coordinate in Q that turns zero-distance evidence into separated classifier evidence.

Metric consumers read the resulting separated classifier before treating the point surface as metric-separated. Regular rational and real consumers read finite windows through R and the real seal E only along the same replay route, so they cannot bypass the classifier by inspecting the pseudometric row directly.

The exclusions follow from the carrier inventory. Quotient metric spaces, host equality, trichotomy, countable-choice selectors, selected limits, selected representatives, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not fields of Q, and the provenance row names no such route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedQuotientMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ packet is generated from source point-history admission, a source $\mathsf{Pseudometric}^{\uparrow}$ packet, displayed zero-distance evidence, separated classifier output, finite StreamName and RegSeqRat readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedQuotientMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2004.1, Definition 2004.2, and Definition 2004.3 seed the finite zero-distance separation certificate between pseudometric rows and metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedQuotientMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient metric spaces, host equality, trichotomy, countable choice, selected limits, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance classifier exposure, separated readback, regular rational and real handoff, transport and replay stability, provenance visibility, and quotient non-escape for BEDC.Derived.SeparatedQuotientMetricUp.

Chapter 2005

A Concrete Naming Certificate for $\mathsf{CompleteMetricProduct}^{\uparrow}$

Following Bishop–Bridges constructive completion and the Bourbaki–Kelley binary product metric construction, the $\mathsf{CompleteMetricProduct}^{\uparrow}$ packet records the standard product-stability route for complete metric spaces as finite BEDC NameCert data. It reads the displayed product metric of chapter 424, the two component complete-metric limit witnesses of chapter 128, the finite stream-name windows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the binary product carrier of chapter 34. The packet names the constructive route from a product Cauchy source to component limits and then back to a displayed product limit, without quotient products, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2005.1 (Complete metric product carrier)**.** *A* $\mathsf{CompleteMetricProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, X, Y, S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y, \Pi, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{ProductMetric}^{\uparrow}$ *row on the binary product carrier. It stores the two projection routes, the two component distance reads, and the product-distance ledger.*
2. *X and Y are the first and second* $\mathsf{CompleteMetric}^{\uparrow}$ *source rows. Each source is read only as a displayed component limit-witness surface.*
3. *S_X and S_Y are the component* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened by projecting a product Cauchy source through the two projection routes of M.*
4. *R_X and R_Y are the component* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows and from the product Cauchy tolerance ledger.*
5. *A_X and A_Y are the terminal component* $\mathsf{Real}^{\uparrow}$ *seal rows reached after the corresponding* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have been displayed.*
6. *L_X and L_Y are the two component limit rows supplied by the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *witnesses X and Y.*
7. *Π is the* $\mathsf{Prod}^{\uparrow}$ *pairing row that repacks the two component limit rows as the displayed product endpoint.*

8. *L is the product-limit route ledger from the product Cauchy source through M, then through $S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y$, and finally through Π.*
9. *H records componentwise* hsame *transport for the product metric, the two complete-metric sources, the component stream windows, regular readbacks, real seals, component limits, product endpoint, route ledger, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a completeness request through*

$$M, X, Y \longmapsto S_X, S_Y \longmapsto R_X, R_Y \longmapsto A_X, A_Y \longmapsto L_X, L_Y, \Pi, L, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricProduct}^{\uparrow}$, $\mathsf{ProductMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricProduct}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, X, Y, S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y, \Pi, L, H, C, P, N$. The carrier has no coordinate for quotient products, selected limits, countable-choice selectors, host pair equality, quotient equality of Cauchy sources, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteMetricProductUp.CompleteMetricProductTasteGate single carrier alignment`

Theorem 2005.2 (Complete metric product NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricProduct}^{\uparrow}$ *carrier*

$$Q = (M, X, Y, S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y, \Pi, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricProduct}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier exposure, requiring the displayed packet $M, X, Y, S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y, \Pi, L, H, C, P, N$ to be the whole public carrier;*
2. *product-metric exactness, requiring every product distance and projection read to pass through the displayed* $\mathsf{ProductMetric}^{\uparrow}$ *row M;*
3. *component Cauchy readback, requiring every projected Cauchy source to open the matching* $\mathsf{StreamName}^{\uparrow}$ *window and then the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback before a component limit witness is visible;*
4. *component limit admission, requiring L_X and L_Y to be obtained from the two displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *rows X and Y after the terminal* $\mathsf{Real}^{\uparrow}$ *seals A_X and A_Y have been displayed;*
5. *product endpoint pairing, requiring the product limit to be the* $\mathsf{Prod}^{\uparrow}$ *pairing row Π applied to the two component limit rows and recorded by the route ledger L;*
6. *non-escape provenance, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and never a quotient product, selected limit, countable-choice schedule, host equality, quotient equality, or ambient completeness theorem.*

No CompleteMetricProduct obligation is stored outside these rows.

Proof. Carrier exposure is tuple inspection of Definition 2005.1. The product-facing metric route is fixed by M, so every product distance and projection read starts from the displayed $\mathsf{ProductMetric}^{\uparrow}$ packet.

The component Cauchy readback obligations follow by projecting the product Cauchy source along the two projection routes stored in M. Each projected source opens its own finite $\mathsf{StreamName}^{\uparrow}$ window and then its $\mathsf{RegSeqRat}^{\uparrow}$ readback before any terminal real seal is available.

The component limit obligations are the two applications of the displayed $\mathsf{CompleteMetric}^{\uparrow}$ rows X and Y. They consume the corresponding stream-name, regular readback, and real-seal rows and return the component limit rows L_X and L_Y.

The product endpoint obligation is then finite pairing. The row Π pairs L_X and L_Y, while L records the route from the original product Cauchy source through the two component reads and back to that product endpoint.

Finally, H, C, P, N only transport, replay, source, and name the rows already displayed. Because none of the carrier coordinates stores quotient products, selected limits, countable choice, host equality, quotient equality, or ambient completeness, the local NameCert cannot export such data. □

Theorem 2005.3 (Complete metric product limit pairing)**.** *Let*

$$Q = (M, X, Y, S_X, S_Y, R_X, R_Y, A_X, A_Y, L_X, L_Y, \Pi, L, H, C, P, N)$$

be an accepted $\mathsf{CompleteMetricProduct}^{\uparrow}$ *carrier. Every componentwise Cauchy source completed in the two factors determines the product-limit route through the displayed* $\mathsf{ProductMetric}^{\uparrow}$ *row* M*: project the product source through* M*, read the two component* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *routes, consume the two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit witnesses* X *and* Y*, and pair the resulting limit rows through* Π*.*

The resulting product-limit certificate is exactly the finite route ledger L*. It introduces no quotient product, selected limit, countable-choice selector, host equality, quotient equality of Cauchy sources, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a product Cauchy source visible to the packet. The only product metric coordinate is M, so both component observations are obtained by the projection routes and distance ledger stored in that $\mathsf{ProductMetric}^{\uparrow}$ row.

The first projection opens S_X, then R_X, and reaches the terminal $\mathsf{Real}^{\uparrow}$ seal A_X. The second projection opens S_Y, then R_Y, and reaches A_Y. Thus both component Cauchy sources are read through finite StreamName and RegSeqRat data before a completion witness is consulted.

Apply the two displayed complete-metric witnesses. The first witness X returns L_X for the first component route, and the second witness Y returns L_Y for the second component route. These are component limit rows, not selected host limits.

Pair the two component limits through Π. The ledger L records exactly this route from the original product Cauchy source to the paired endpoint, so the product-limit certificate is componentwise before it is product-facing.

The exclusions follow from the carrier boundary. Since Q contains no quotient product, no chosen limit selector, no countable-choice schedule, no host equality row, no quotient equality row, and no ambient completeness coordinate, none can be recovered from the route ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricProduct}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2005.1, Definition 2005.2, and Definition 2005.3 seed the product-stability certificate for complete metric spaces through ProductMetric, component CompleteMetric limit witnesses, StreamName windows, RegSeqRat readback, Real seals, product pairing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient products, selected limits, countable choice, host equality, quotient equality of Cauchy sources, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-metric exactness, component Cauchy readback, component limit admission, product endpoint pairing, provenance non-escape, and local naming for BEDC.Derived.CompleteMetricP

Constructive story: A CompleteMetricProduct$^{\uparrow}$ packet is generated from one ProductMetricUp row, two CompleteMetricUp rows, component StreamName windows, RegSeqRat readbacks, RealUp seals, two component limit rows, one ProdUp pairing row, a product-limit route ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2006

A Concrete Naming Certificate for $\mathsf{UniformIntegralLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformIntegralLimit}^{\uparrow}$ packet names the finite handoff by which a uniformly controlled sequence of integration packets reaches a single $\mathsf{Real}^{\uparrow}$ seal. It sits after the tagged-mesh route of chapter 1526, the upper-lower gap route of chapter 1460, and the gauge-partition route of chapter 1999. It consumes regulated approximation rows from chapter 1906, regular rational handoff rows from chapter 301, and the terminal real-name boundary of chapter 38. The packet records a uniform passage of integral rows to one real seal; it does not introduce measure completion, quotient function spaces, selected limiting partitions, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2006.1 (Uniform integral limit carrier)**.** *A* $\mathsf{UniformIntegralLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, F, T, J, A, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row, naming the endpoint histories, order evidence, and cell boundary ledger shared by every integration packet in the window.*
2. *F is the integrand-window sequence. Each row records a displayed function-read window, its index, and the finite observation data used by the corresponding integral packet.*
3. *T is the uniform tail ledger. It records the common tolerance schedule and the visible index bound after which all carried integrand windows and integral rows remain in the same requested regular rational window.*
4. *J is the Riemann, Darboux, or Gauge integral row attached to each accepted window. Its entries are routed through* $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, *or* $\mathsf{GaugeIntegral}^{\uparrow}$ *over the same interval row I.*
5. *A is the regulated approximation row, recording the* $\mathsf{RegulatedFunction}^{\uparrow}$ *step-approximation ledger used to compare the integrand windows before the integral rows are sealed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row produced from the tail ledger and the displayed integral rows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the interval, integrand-window, uniform-tail, integral, regulated-approximation, and regular rational rows have been displayed.*

8. *H records componentwise* hsame *transport for interval rows, integrand-window sequences, uniform tail ledgers, Riemann/Darboux/Gauge integral rows, regulated approximation rows, regular rational handoff, real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from*

$$I, F \longmapsto T \longmapsto J \longmapsto A \longmapsto Q \longmapsto E$$

and then through the transport, provenance, and local naming rows.

10. *P is the* Pkg *provenance over* UniformIntegralLimit$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, GaugeIntegral$^{\uparrow}$, RegulatedFunction$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{UniformIntegralLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, T, J, A, Q, E, H, C, P, N$. It has no coordinate for a measure completion, a quotient of functions, selected limiting partitions, open-cover compactness, or an ambient completeness theorem for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformIntegralLimitUp.UniformIntegralLimitTasteGate single carrier alignment`

Theorem 2006.2 (Uniform integral limit NameCert obligations)**.** *For every accepted* UniformIntegralLimit$^{\uparrow}$ *carrier*

$$U = (I, F, T, J, A, Q, E, H, C, P, N),$$

the local NameCert$_{\text{UniformIntegralLimit}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the interval row, integrand-window sequence, uniform tail ledger, integration row family, regulated approximation row, regular rational handoff, real seal, componentwise transport, continuation replay, provenance, and local naming rows;*

2. *interval coherence, requiring every Riemann, Darboux, or Gauge entry in J to read the same displayed interval row I;*

3. *uniform tail exposure, requiring T to be present before the regular rational handoff Q and the real seal E are readable;*

4. *integration-route visibility, requiring each entry of J to route through* RiemannIntegral$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, *or* GaugeIntegral$^{\uparrow}$ *rather than through a host integral;*

5. *regulated approximation compatibility, requiring A to expose the finite regulated-function approximation rows used to compare the integrand windows;*

6. *real-seal non-escape, requiring E to be downstream of T, J, A, Q and to export no independent completeness principle;*

7. *structural faithfulness, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no measure completion, quotient function space, selected limiting partition, open-cover compactness theorem, ambient* Real$^{\uparrow}$ *completeness witness, Lean verification, or bridge row.*

No uniform integral limit NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2006.1. The packet displays the interval row, the integrand-window sequence, the uniform tail ledger, the integration row family, regulated approximation data, regular rational handoff, real seal, transport, replay, provenance, and local name.

Interval coherence and integration-route visibility are read from I and J. Each integration row in J is attached to the same interval row and is routed through one of the already named Riemann, Darboux, or Gauge packets, so the carrier never reads a detached host integral.

Uniform tail exposure is the role of T. The regular rational row Q is available only after the common tail ledger has been displayed, and the real seal E is available only after Q and the carried integration rows have been read.

The regulated approximation row A supplies the comparison surface for the integrand-window sequence. Transport through H and replay through C preserve the same order, while P and N record the provenance and local naming. Since the excluded analytic objects are not fields of U, the listed rows exhaust the local certificate surface. □

Theorem 2006.3 (Uniform integral limit real seal handoff)**.** *Let* $U = (I, F, T, J, A, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{UniformIntegralLimit}^{\uparrow}$ *packet. Every consumer read of the* $\mathsf{Real}^{\uparrow}$ *seal* E *factors through the finite route*

$$I, F \longmapsto T \longmapsto J \longmapsto A \longmapsto Q \longmapsto E.$$

Thus the real-facing output records the common interval, the displayed integrand-window sequence, the uniform tail ledger, the Riemann/Darboux/Gauge integration rows, the regulated approximation row, and the regular rational handoff, and it does not expose measure completion, quotient function spaces, selected limiting partitions, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Start with a consumer read of E. By Definition 2006.1, the seal is downstream of the interval and integrand-window rows, the uniform tail ledger, the integration rows, the regulated approximation row, and the regular rational handoff.

The row T supplies the common tail control before Q can be read. The row J supplies the finite integration surface through the Riemann, Darboux, or Gauge packets, and A records the regulated approximation ledger needed to compare the carried integrand windows.

Continuation replay through C rebuilds this route, and componentwise transport through H preserves it under hsame. The provenance row P and local name N add no additional analytic carrier.

Therefore every real-facing consumer of E sees only the displayed finite route. The excluded objects have no coordinate in the carrier and cannot be exported by the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformIntegralLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformIntegralLimit}^{\uparrow}$ packet is generated from a displayed interval row, an integrand-window sequence, a uniform tail ledger, Riemann/Darboux/Gauge integral rows, a regulated approximation row, a RegSeqRat handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformIntegralLimit}^{\uparrow}}$, seed)

Scope closed: Definition 2006.1, Definition 2006.2, and Definition 2006.3 seed the finite uniform passage from integration packets to a Real seal over Riemann, Darboux, Gauge, RegulatedFunction, RegSeqRat, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformIntegralLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function spaces, selected limiting partitions, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval coherence, uniform tail exposure, integration-route visibility, regulated approximation compatibility, real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformIntegralLimitUp.

Chapter 2007

A Concrete Naming Certificate for $\mathsf{EngelExpansion}^{\uparrow}$

Following Engel's expansion theorem and the Hardy–Wright treatment of rational expansion by increasing denominators, the $\mathsf{EngelExpansion}^{\uparrow}$ packet records finite Engel readback before a rational expansion surface is allowed to enter the regular rational and Real seal pipeline. It sits beside the rational carrier of chapter 37, the dyadic bound ledger of chapter 291, the finite window discipline of chapter 126, the regular rational handoff of chapter 301, the terminal real seal of chapter 38, and the continued-fraction comparison route of chapter 342. The packet is a finite denominator-schedule certificate; it does not assert quotient reals, selected infinite sums, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2007.1 (Engel expansion carrier)**.** *An* $\mathsf{EngelExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, B, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the denominator schedule row, listing the accepted finite Engel denominators and their monotone successor checks.*
2. *S is the* $\mathsf{Rat}^{\uparrow}$ *partial-sum row, displaying the finite rational sums obtained from the visible denominator schedule.*
3. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder-bound row, recording the dyadic upper bounds that certify the unread Engel tail for each exposed window.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, selecting exactly the denominator prefixes, partial sums, and remainder bounds inspected by the readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D, S, B, W and no other numeric coordinate.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback row has been displayed.*
7. *H records componentwise* hsame *transport for the denominator schedule, partial sums, remainder bounds, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through*

$$D, S, W \longmapsto B \longmapsto R \longmapsto L.$$

9. *P is the* Pkg *provenance over* $\mathsf{EngelExpansion}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EngelExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient real equality, a selected value of an infinite sum, trichotomy, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a continued-fraction uniqueness theorem.*

Theorem 2007.2 (EngelExpansion NameCert obligations)**.** *For every accepted* $\mathsf{EngelExpansion}^{\uparrow}$ *carrier $E = (D, S, B, W, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EngelExpansion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the denominator schedule, partial-sum rows, remainder-bound rows, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *denominator-schedule visibility, requiring each displayed denominator and successor check to be recorded in D before any partial sum is read;*
3. *rational partial-sum exactness, requiring S to expose only* $\mathsf{Rat}^{\uparrow}$ *sums generated from the denominator windows selected by W;*
4. *dyadic remainder control, requiring every tail claim consumed by R or L to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row B;*
5. *readback order, requiring the finite route $D, S, W \longmapsto B \longmapsto R \longmapsto L$ before any Real-facing consumer receives the packet;*
6. *transport, continuation replay, provenance, and local naming for exactly the rows recorded in H, C, P, N.*

No EngelExpansion NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.EngelExpansionUp.EngelExpansionTasteGate single carrier alignment`

Proof. Project the accepted carrier $E = (D, S, B, W, R, L, H, C, P, N)$. The projection exposes exactly the denominator schedule, partial sums, remainder bounds, stream windows, readback rows, and structural rows.

The finite Engel data are read before the analytic handoff. The row D supplies the visible denominator schedule, and W selects the finite windows in which that schedule may be inspected. The partial-sum row S can therefore present only rational sums generated from those displayed windows.

The remainder step is separate. The row B supplies dyadic tail bounds for the unread Engel part, so a regular rational or Real-facing consumer cannot treat the expansion as a selected infinite sum.

Finally R and L are downstream of B. Componentwise transport in H and replay in C preserve the same displayed route, while P and N only source and name it. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2007.3 (EngelExpansion real readback boundary)**.** *Let $E = (D, S, B, W, R, L, H, C, P, N)$ be an accepted* $\mathsf{EngelExpansion}^{\uparrow}$ *carrier. Every finite Engel expansion readback consumed as a regular rational name or as a terminal real-name seal factors through*

$$D, S, W \longmapsto B \longmapsto R \longmapsto L$$

with structural support H, C, P, N. Thus the consumer receives a denominator schedule, $\mathsf{Rat}^{\uparrow}$ *partial sums,* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder bounds,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal, all as finite BEDC data.*

The boundary supplies no quotient real equality, no selected infinite sum, no trichotomy row, no countable-choice selector, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no claim that the continued-fraction and Engel presentations are unique as host real objects.*

Proof. Start with a downstream consumer of R or L. By the carrier definition, neither row is an independent real object; both are downstream of the finite Engel schedule and dyadic tail ledger.

The consumer first reads D, S, and W. These rows expose the accepted denominators, the rational partial sums, and the finite windows in which the expansion is being inspected.

Next the consumer reads B. This row is the only remainder-control coordinate in the packet, and it records dyadic bounds for the unread Engel tail before any regular rational readback is available.

The row R then repacks the displayed partial sums and bounds as a $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal row L seals only that regular rational route as a $\mathsf{Real}^{\uparrow}$ presentation. The structural rows transport, replay, package, and name the same route, so the excluded principles cannot be recovered from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EngelExpansion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EngelExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2007.1, Definition 2007.2, and Definition 2007.3 seed the finite Engel denominator-schedule route through rational partial sums, dyadic remainder bounds, regular rational readback, and terminal Real seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EngelExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, selected infinite sums, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or uniqueness against continued-fraction presentations as host real objects.

Upgrade path: Obligation closure requires checked carrier admission, denominator-schedule visibility, rational partial-sum exactness, dyadic remainder control, readback-order, and real readback boundary proofs for BEDC.Derived.EngelExpansionUp.

Constructive story: An $\mathsf{EngelExpansion}^{\uparrow}$ packet is generated from finite denominator schedule rows, RatUp partial-sum rows, DyadicRatCore remainder-bound rows, StreamName windows, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2008

A Concrete Naming Certificate for $\mathsf{GeometricSeries}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{GeometricSeries}^{\uparrow}$ packet records the finite geometric majorant route used by Real and Completion-facing certificates before a tail estimate is handed to a regular rational or Real seal. It sits beside the dyadic finite-tail ledger of **??**, the ratio and root-test routes of chapter 1461 and chapter 1481, the radius transport of chapter 66, and the complex partial-sum surface of chapter 69. The packet names a displayed majorant discipline; it does not assert selected infinite sums, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2008.1 (Geometric series carrier)**.** *A* $\mathsf{GeometricSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (a, q, n, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *a is the initial-term row, recording the finite magnitude budget that enters the majorant.*
2. *q is the dyadic ratio row, displaying a ratio strictly below the unit tolerance scale together with the dyadic slack used by later tail requests.*
3. *n is the unary exponent-window row, recording the finite exponent range and successor action on which the powers of q are inspected.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family that exposes the source tail and the finite power windows used by the packet.*
5. *T is the* $\mathsf{DyadicGeometricTail}^{\uparrow}$ *ledger that turns a, q, and n into a finite geometric tail budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only from the displayed tail ledger and the inspected stream windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after the regular rational readback has been displayed.*
8. *H records componentwise* hsame *transport for the initial term, ratio, exponent window, stream windows, dyadic tail ledger, regular readback, Real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a precision request through*

$$a, q, n \longmapsto W \longmapsto T \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{GeometricSeries}^{\uparrow}$, $\mathsf{DyadicGeometricTail}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GeometricSeries}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a selected infinite sum, quotient stream equality, countable-choice selector, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or bridge credential.*

Lean4 ✓ `BEDC.Derived.GeometricSeriesUp.TasteGate.GeometricSeriesTasteGate single carrier alignment`

Theorem 2008.2 (Geometric series NameCert obligations)**.** *For every accepted* $\mathsf{GeometricSeries}^{\uparrow}$ *carrier* $G = (a, q, n, W, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GeometricSeries}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed eleven-row packet;*
2. *initial-term exposure for a and dyadic ratio exposure for q, including the explicit strict-below-unit slack;*
3. *unary exponent-window coverage for n and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W;*
4. *tail-ledger exactness, requiring T to be read from a, q, n, W through the* $\mathsf{DyadicGeometricTail}^{\uparrow}$ *surface;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring R and E to factor through the displayed tail ledger;*
6. *transport, continuation replay, provenance, and local naming for exactly the rows recorded in H, C, P, N.*

No geometric-series NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier $G = (a, q, n, W, T, R, E, H, C, P, N)$. The projection exposes the initial term, ratio, exponent window, stream windows, tail ledger, regular readback, Real seal, and structural rows.

The numeric part of the certificate is finite. The rows a and q display the starting magnitude and the strict subunit dyadic ratio, while n and W specify the only exponent and source windows admitted for inspection.

The row T is then the sole tail-producing coordinate. It sends the displayed data through the $\mathsf{DyadicGeometricTail}^{\uparrow}$ ledger, so the geometric majorant is a finite BEDC row rather than an appeal to a host infinite sum.

Finally R and E are downstream of T. Componentwise transport in H preserves the same displayed route, and C, P, N only replay, source, and name that route. Therefore the listed rows exhaust the local NameCert obligations. □

Theorem 2008.3 (Geometric series tail modulus)**.** *Let $G = (a, q, n, W, T, R, E, H, C, P, N)$ be an accepted* $\mathsf{GeometricSeries}^{\uparrow}$ *carrier. Every precision request consumed by G receives a finite tail threshold by the route*

$$a, q, n, W \longmapsto T \longmapsto R.$$

The resulting modulus is the one displayed by the $\mathsf{DyadicGeometricTail}^{\uparrow}$ *ledger and then read back through* $\mathsf{RegSeqRat}^{\uparrow}$*; it is not a selected limit of the infinite geometric series.*

Proof. Fix a displayed precision request. The rows a and q provide the starting budget and subunit ratio, while n selects the finite exponent window in which the request is evaluated.

Open the stream windows in W. These windows are the only source observations visible to the packet, so the request cannot inspect an unrecorded tail or a quotient stream representative.

Apply the tail ledger T. The ledger computes the dyadic geometric budget and returns the finite threshold required by the request. The row R reads that threshold as a $\mathsf{RegSeqRat}^{\uparrow}$ modulus, and the structural rows preserve the same route under transport and replay. □

Theorem 2008.4 (Geometric series majorant handoff)**.** *Every accepted* $\mathsf{GeometricSeries}^{\uparrow}$ *majorant exported to a ratio-test, root-test, convergence-radius, complex-series, or Real-facing consumer factors through the finite handoff*

$$a, q, n \longmapsto W, T \longmapsto R, E, H, C, P, N.$$

Consequently the consumer receives only the displayed initial-term row, subunit dyadic ratio row, unary exponent window, stream windows, $\mathsf{DyadicGeometricTail}^{\uparrow}$ *ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal row, transport package, replay route, provenance, and local naming row.*

Proof. Start with an exported majorant. Carrier projection shows that the only available numerical rows are a, q, and n, and the only admitted observations of the source tail lie in W.

The majorant must then pass through T. Since T is the displayed $\mathsf{DyadicGeometricTail}^{\uparrow}$ ledger, every downstream comparison receives a finite dyadic tail budget before it can read the regular rational handoff.

The handoff to R and E does not add analytic power. It merely presents the displayed tail budget as a $\mathsf{RegSeqRat}^{\uparrow}$ readback and a Real-facing seal row.

The remaining structural rows preserve the same route. Thus a consumer of the ratio-test, root-test, convergence-radius, complex-series, or Real-facing surface cannot read a selected infinite sum, quotient stream equality, countable-choice selector, host real equality, or ambient completeness theorem from the $\mathsf{GeometricSeries}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeometricSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{GeometricSeries}^{\uparrow}$ packet is generated from an initial-term row, a dyadic subunit-ratio row, a unary exponent-window row, finite StreamName windows, a DyadicGeometricTail ledger, a RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GeometricSeries}^{\uparrow}}$, seed)

Scope closed: Definition 2008.1, Definition 2008.2, Definition 2008.3, and Definition 2008.4 seed the finite geometric majorant route over dyadic tail ledgers, regular rational readback, Real seals, stream windows, and BEDC transport rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GeometricSeries}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected infinite sums, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite NameCert obligation surface, tail-modulus theorem, and majorant-handoff theorem for BEDC.Derived.GeometricSeriesUp.

The $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ packet routes finite product coordinates through chapter 301, chapter 126, and the completion-facing boundary of chapter 1569.

Chapter 2009

A Concrete Naming Certificate for $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real arithmetic, the $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ packet records the finite product of displayed regular Cauchy rational names as one synchronized NameCert route. It sits after the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, the terminal real-name seal of chapter 38, the binary regular product route of chapter 461, and the completion-facing product packet of chapter 464. The carrier packages repeated binary product evidence without importing a host finite product, quotient stream equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The finite-product packet consumes chapter 301, chapter 126, chapter 291, and chapter 38 as real L10 dependencies: their rows supply the source sequences, synchronized windows, dyadic product budgets, and terminal seal carried by the packet, rather than ambient host finite-product structure.

Definition 2009.1 (Regular Cauchy finite product carrier)**.** *A* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, W, D, P, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *I is the finite unary index row. It lists exactly the component slots opened by the product request and the fold order used by the displayed route.*
2. *S is the component* $\mathsf{RegSeqRat}^{\uparrow}$ *source family, one row for each slot of I.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for the component reads, intermediate product reads, and final product-facing read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, storing the component dyadic observations, intermediate product observations, requested radii, and slack cells.*
5. *P is the finite product-budget ledger. It records the binary-product schedule, radius allocation, and accumulated dyadic error account for the fold named by I.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the final folded product, obtained only from S, W, D, P.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback of the folded product has been displayed.*

8. *H records componentwise* hsame *transport for the index row, sources, synchronized windows, dyadic observations, product-budget ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from any finite product request through the same rows.*
10. *Q is the* Pkg *provenance over* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, P, R, E, H, C, Q, N$. It has no coordinate for a host finite product, quotient stream equality, selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Theorem 2009.2 (Regular Cauchy finite product NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier*

$$K = (I, S, W, D, P, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteProduct}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the finite unary index row, component* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations, product-budget ledger, final readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local name;*
2. *finite tuple-window coverage, requiring every component slot in I to open the matching source row and all synchronized windows needed by the fold;*
3. *dyadic product-budget exactness, requiring every intermediate product observation to consume its assigned radius and slack cells from D and P;*
4. *RegSeqRat readback stability, requiring the final readback row R to depend only on the displayed sources, windows, dyadic observations, and product-budget ledger;*
5. *Real seal non-escape, requiring E, H, C, Q, N to expose no host finite product, quotient stream equality, selected limit, trichotomy, ambient completeness theorem, Lean verification, or bridge credential.*

No Regular Cauchy finite product obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 2009.1. The displayed tuple $I, S, W, D, P, R, E, H, C, Q, N$ is the entire public surface, so carrier admission is exactly the admission of those rows.

Tuple-window coverage is forced by the index row. Each slot named by I has one component $\mathsf{RegSeqRat}^{\uparrow}$ source in S, and the synchronized window row W is the only place where component, intermediate, and final product reads are opened.

Dyadic exactness is the joint role of D and P. The observations in D supply component and intermediate dyadic cells, while P assigns the binary-product schedule, radii, slack, and accumulated error account for the fold.

The final regular readback row R is downstream of S, W, D, P. Therefore readback stability follows from the carrier order: no product-facing $\mathsf{RegSeqRat}^{\uparrow}$ data can be inspected until the synchronized windows and dyadic product budget have already been displayed.

Finally, the seal, transport, replay, provenance, and local name rows preserve only this displayed route. Since none of the excluded objects is a coordinate of the carrier, the listed five rows exhaust the local NameCert obligations. □

Theorem 2009.3 (Regular Cauchy finite product tail budget)**.** *Let*

$$K = (I, S, W, D, P, R, E, H, C, Q, N)$$

be an accepted $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier. Every finite product read from* K *factors through*

$$I, S \longmapsto W \longmapsto D, P \longmapsto R \longmapsto E, H, C, Q, N.$$

In particular, no read reaches the final $\mathsf{RegSeqRat}^{\uparrow}$ *product row before the synchronized windows and dyadic product-budget ledger have been displayed.*

Proof. Begin with a finite product request. The request first names its finite unary index row I and the component source family S, because those are the only carrier rows that identify which regular Cauchy names participate in the fold.

The requested component and intermediate observations then pass through W. This synchronized window row is the only row that aligns the component reads with the intermediate product reads used by the fold.

The dyadic part of the request is read next. The observation row D supplies the dyadic product cells and requested radii, while the budget row P supplies the binary-product schedule and accumulated slack account.

Only after those rows are displayed can the final regular rational readback R be consumed. The downstream rows E, H, C, Q, N seal, transport, replay, source, and name the same route, so the finite product read cannot bypass the synchronized windows or the dyadic ledger. □

Theorem 2009.4 (Regular Cauchy finite product budget associativity)**.** *Let*

$$K = (I, S, W, D, P, R, E, H, C, Q, N)$$

be an accepted $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier. Regrouping adjacent binary product folds in the displayed fold order preserves the same synchronized windows* W*, dyadic observations* D*, product-budget ledger* P*, final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E*. The regrouped read is a projection of the displayed fold rows of* K*, not an appeal to host associativity of finite products or quotient stream equality.*

Proof. Apply Definition 2009.3. Every finite product read from K factors through the index and source rows, then through the synchronized windows, then through the dyadic observations and product-budget ledger, before reaching the final readback and seal.

The carrier rows of Definition 2009.1 identify where regrouping can be displayed. The finite unary index row I names the fold order, while W, D, P record the windows, observations, product schedule, radii, and slack cells used by that order.

Regrouping adjacent binary product folds therefore changes only the displayed parenthesization inside the schedule already stored in I and P. It does not create a second window family, a second dyadic observation row, a second product-budget ledger, or a second final readback.

Since R and E are downstream of the fixed rows W, D, P, the regrouped read preserves the same final $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ seal. The argument is projection from the accepted carrier, so no host finite-product operation or quotient stream equality is introduced. □

Theorem 2009.5 (Regular Cauchy finite product projection stability)**.** *For every accepted* RegularCauchyFiniteProd *carrier* $K = (I, S, W, D, P, R, E, H, C, Q, N)$*, projecting any component slot from* K *factors through the same finite index row* I*, source family* S*, synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family* W*, dyadic*

product-budget ledger D, P, final $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and* $\mathsf{Real}^{\uparrow}$ *seal E. Component projection can expose a component readback only through this displayed route; it cannot bypass the budget ledger or invoke host finite products, quotient stream equality, selected component representatives, trichotomy, or ambient completeness.*

Proof. Project the carrier through Definition 2009.1. The component slot is named by I, its source is a row of S, and every read serving the projection must pass through the synchronized window family W.

Apply the tail-budget theorem Definition 2009.3. It places every product-facing read through $I, S, W, D, P, R, E, H, C, Q, N$ before a component-facing consumer can inspect the result.

Use budget associativity from Definition 2009.4. Regrouping the displayed fold may expose the requested component slot, but it preserves the same dyadic observations, product-budget ledger, final regular readback, and Real seal.

The structural rows H, C, Q, N transport, replay, source, and name only that displayed route. Since the carrier has no host finite-product coordinate, quotient stream equality, selected component representative, trichotomy, or ambient completeness theorem, projection stability is exhausted by the finite-product packet itself. □

Theorem 2009.6 (Regular Cauchy finite product Real seal non-escape)**.** *For every accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier K, the* $\mathsf{Real}^{\uparrow}$ *seal row exposed by K is a downstream seal for the displayed finite route*

$$I, S, W, D, P, R.$$

It exposes no host finite product, no quotient stream equality, no selected limit, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the carrier boundary. The seal row E appears only after the index, source, synchronized window, dyadic observation, product-budget, and regular readback rows.

The transport and replay rows H and C preserve this order. They can move or replay the displayed route, but they do not add an independent product coordinate or a host equality coordinate.

The provenance and local naming rows Q and N cite the same dependencies. Since the carrier has no coordinate for a host finite product, quotient stream equality, selected limit, trichotomy, or ambient completeness theorem, the $\mathsf{Real}^{\uparrow}$ seal cannot export any of them. □

Theorem 2009.7 (Regular Cauchy finite product projection roundtrip)**.** *For every accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier $K = (I, S, W, D, P, R, E, H, C, Q, N)$, any left or right projection read consumed by* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *factors back through the same finite index row I, source family S, synchronized window family W, dyadic observations D, product-budget ledger P, and final readback R. The projection packet may expose component readbacks, but it cannot introduce a host product, selected component representative, or quotient stream equality.*

Proof. Begin with an accepted finite-product carrier $K = (I, S, W, D, P, R, E, H, C, Q, N)$ and a projection read consumed by chapter 2076. The projection can name a left or right component only after it names the product packet from which that component is read.

Apply the tail-budget theorem Definition 2009.3. The product-facing read factors through I, S, W, D, P, R before any $\mathsf{Real}^{\uparrow}$ seal or component-facing export is available.

Apply budget associativity from Definition 2009.4. Regrouping the binary fold may expose the component position needed by the projection packet, but it preserves the same synchronized windows, dyadic observations, product-budget ledger, final readback, and seal route.

Finally use the Real seal non-escape theorem Definition 2009.6. The projection packet may expose component readbacks downstream of the displayed finite product route, but no host product, selected component representative, or quotient stream equality is present in the carrier. □

Theorem 2009.8 (Regular Cauchy finite product diagonal seal)**.** *For every accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *carrier* $K = (I, S, W, D, P, R, E, H, C, Q, N)$*, any diagonal read or component projection read from* K *preserves the same synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations* D*, product-budget ledger* P*, final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. Thus a finite-product consumer may reuse the projected route as a sealed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *input, but it cannot introduce a host product, selected component representative, quotient stream equality, trichotomy, or ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 2009.2. They order every accepted carrier through the finite unary index row, component sources, synchronized windows, dyadic observations, product-budget ledger, final readback, seal, transport, replay, provenance, and local name.

Induct over the displayed finite fold route. The base read is one component source with its synchronized window and dyadic observation rows, so the readback and seal are exactly the rows displayed in R and E.

For the fold step, use budget associativity from Definition 2009.4. Regrouping a binary product exposes the component or diagonal position without changing W, D, P, R, E.

Finally apply the Real seal non-escape theorem Definition 2009.6 and the projection roundtrip Definition 2009.7. The projected read still passes through the same finite product route before it reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ sealing, so no excluded host equality or completion object is exported. □

Theorem 2009.9 (Regular Cauchy finite product classifier bridge)**.** *Two accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packets*

$$K = (I, S, W, D, P, R, E, H, C, Q, N) \quad \textit{and} \quad K' = (I', S', W', D', P', R', E', H', C', Q', N')$$

are classifier-equal exactly when the finite unary index rows I, I' *match, the component* $\mathsf{RegSeqRat}^{\uparrow}$ *classifiers on* S, S' *agree slotwise, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window classifiers on* W, W' *agree for component, intermediate, and final reads, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and readback ledgers* D, P *and* D', P' *are equal as displayed budget data, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal route through* R, E, H, C, Q, N *is the same as the route through* R', E', H', C', Q', N'*. This correspondence is the finite-product classifier surface consumed by product regular-Cauchy limit certificates and Real algebra finite-product certificates.*

Proof. For the forward direction, project both accepted carriers using Definition 2009.1. The classifier for an accepted packet has no coordinate outside $I, S, W, D, P, R, E, H, C, Q, N$, so equality of packet classifiers can be read only from the listed rows.

The first projection reads the finite unary index row. Once I and I' match, the component slots are aligned, and the component classifier comparison is forced to be slotwise comparison of the $\mathsf{RegSeqRat}^{\uparrow}$ source family S against S'.

Next read the synchronized window and dyadic budget rows. The window classifiers on W, W' cover the component, intermediate, and final product reads, while the ledgers D, P and D', P' record the tolerance cells, readback cells, radius allocation, fold schedule, and slack account. These are the only budget rows available to the classifier.

Finally, use the Real seal non-escape theorem Definition 2009.6. The route through R, E, H, C, Q, N cannot export a host finite product or quotient stream equality, so downstream classifier agreement is exactly agreement of that displayed seal route.

For the reverse direction, rebuild the packet classifier from the matching rows. The common finite index row aligns the sources; the component $\mathsf{RegSeqRat}^{\uparrow}$ classifiers and synchronized $\mathsf{StreamName}^{\uparrow}$ window classifiers rebuild the inspected fold; the equal dyadic tolerance and budget ledgers rebuild the product-accounting part; and the common route through R, E, H, C, Q, N supplies the same terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local name.

The diagonal seal theorem Definition 2009.8 then shows that every product-facing consumer sees the same projected route. Since the carrier exposes no additional classifier coordinate, the rebuilt rows are sufficient for classifier equality, and the same bridge is available to downstream regular-Cauchy limit and Real algebra finite-product consumers. □

Theorem 2009.10 (Regular Cauchy finite product coordinate envelope)**.** *For every pair of classifier-equal accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packets, any coordinate-facing consumer must read the aligned finite index row, the component* $\mathsf{RegSeqRat}^{\uparrow}$ *classifiers, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and product-budget ledgers, the final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before exposing a coordinate. The coordinate envelope is the displayed finite-product packet itself; it excludes host finite products, selected representatives, quotient stream equality, trichotomy, and ambient completeness.*

Proof. Start with the carrier definition Definition 2009.1. A coordinate slot is named only by the finite unary index row I, and its component source is read only through the displayed family S.

Apply the tail-budget theorem Definition 2009.3. It forces every product-facing read through the synchronized windows W, dyadic observations D, product-budget ledger P, final readback R, and downstream rows before a coordinate-facing consumer can inspect the result.

Use the diagonal seal Definition 2009.8 and the classifier bridge Definition 2009.9. Classifier equality aligns the finite index row, component classifiers, synchronized windows, dyadic ledgers, readback, and seal route across the two packets.

Finally apply Real seal non-escape from Definition 2009.6. Since neither packet has a coordinate for a host finite product, selected representative, quotient stream equality, trichotomy, or ambient completeness, the displayed rows form the whole coordinate envelope. □

Falsifiable prediction: Every $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ product-facing read factors through a finite unary index row, component RegSeqRat sources, synchronized StreamName windows, DyadicRatCore observations, a finite product-budget ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyproduct, cauchyproduct: $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite unary product index, synchronized multi-source windows, dyadic product-budget accounting, final RegSeqRat readback, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteProduct}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2009.1, Definition 2009.2, Definition 2009.3, Definition 2009.4, and Definition 2009.6 seed the finite product route from displayed regular Cauchy sources through synchronized windows, dyadic product budgets, final RegSeqRat readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteProduct}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyFiniteProductUp`

Bridge status: none

Not claimed: This chapter does not close host finite products, quotient stream equality, selected limits, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tuple-window coverage, budget exactness, readback stability, seal non-escape, and local naming for BEDC.Derived.RegularCauchyFiniteProductUp.

Constructive story: A RegularCauchyFiniteProduct$^{\uparrow}$ packet is generated from a finite unary index row, component RegSeqRat source rows, synchronized StreamName windows, DyadicRatCore observations, a finite product-budget ledger, final RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The downstream projection packet of chapter 2076 consumes the RegularCauchyFiniteProduct$^{\uparrow}$ carrier's finite index row, component source rows, synchronized window family, dyadic product-budget ledger, and final readback row as the shared route from which left and right projection readbacks are exposed. Thus projection consumers reuse the finite product's displayed source, window, budget, and readback rows rather than importing a host product or a selected component representative. The finite product packet depends on the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, and the terminal real-name seal of chapter 38. Its product-facing route consumes synchronized regular Cauchy source windows and a displayed dyadic product-budget ledger; it does not import a host finite product, a quotient stream equality, or a selected tuple representative. The genuine input route for RegularCauchyFiniteProduct$^{\uparrow}$ reads the finite product ledger of chapter 34, then the synchronized finite observation windows of chapter 126, and then the regular rational readback surface of chapter 301 before any Real-facing projection or terminal seal is consumed. This order keeps **??**, StreamName$^{\uparrow}$, and RegSeqRat$^{\uparrow}$ visible as horizontal dependencies rather than hiding the finite product inside a kernel-only carrier or a downstream Real$^{\uparrow}$ projection. The forward lattice for RegularCauchyFiniteProduct$^{\uparrow}$ is consumed by chapter 2076 and chapter 464 through the shared finite index, synchronized StreamName$^{\uparrow}$ windows, dyadic product-budget ledger, and RegSeqRat$^{\uparrow}$ readback. Downstream product and projection packets therefore reuse the displayed finite route rather than adding a host product, a selected component representative, quotient stream equality, trichotomy, or ambient completeness. The product-budget projection lattice for RegularCauchyFiniteProduct$^{\uparrow}$ is consumed by chapter 2076, chapter 461, and chapter 464. The route exports the finite index rows, synchronized StreamName$^{\uparrow}$ windows, DyadicRatCore$^{\uparrow}$ product-budget ledger, projection rows, and RegSeqRat$^{\uparrow}$ readbacks as one displayed dependency surface, so downstream product consumers reuse the finite ledger rather than importing a host product coordinate, a selected component representative, quotient stream equality, trichotomy, or ambient completeness.

Theorem 2009.11 (Regular Cauchy finite product dependency-lattice route)**.** *Every accepted* RegularCauchyFiniteProduct$^{\uparrow}$ *packet reads its horizontal dependency lattice through* ***??***, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, *and* Real$^{\uparrow}$ *before a finite-product read is exported. The route consumes synchronized stream windows and dyadic budget rows for the displayed fold; it does not import host finite products, selected component representatives, quotient stream equality, trichotomy, or ambient completeness.*

Proof. Start with the product-facing request. The finite product source is opened through **??** only as a displayed dependency row, and the component regular Cauchy data are read through RegSeqRat$^{\uparrow}$ sources.

The synchronized observation surface comes next. StreamName$^{\uparrow}$ supplies the shared finite windows for the components and intermediate reads, while DyadicRatCore$^{\uparrow}$ supplies the tolerance and product-budget rows used by the fold.

The binary product route through RegularCauchyProduct$^{\uparrow}$ is then replayed inside the finite-product packet, and the terminal Real$^{\uparrow}$ seal is read only after the final regular readback is displayed.

Because every step uses these finite rows, the dependency lattice exports no host finite product or selected coordinate object. □

Theorem 2009.12 (RegularCauchyFiniteProduct coordinate stability)**.** *If two accepted* RegularCauchyFiniteProduc *packets are classifier-equivalent, then each projected coordinate remains a* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible regular Cauchy readback under the same* $\mathsf{StreamName}^{\uparrow}$ *windows and product-projection rows. In particular, a coordinate projection cannot separate the two packets unless the separation is already visible in the finite product source row, synchronized window family, dyadic product-budget ledger, regular-rational readback, or displayed projection route.*

Proof. Read the product projection route first. The dependency route through chapter 2076 exposes the finite product source, synchronized windows, dyadic product budget, and projection readback as the only public route to a coordinate.

Next read the regular-rational and stream-name route. The dependency line through chapter 301 and chapter 126 fixes the windows and readback rows used by each coordinate.

Classifier equivalence of the two finite-product packets therefore transports through the same source, window, budget, readback, and projection rows. Each coordinate remains a RegSeqRat-compatible regular Cauchy readback, and any failure of stability must already be visible in one of the displayed rows rather than in a hidden host product or selected component representative. □

Theorem 2009.13 (Regular Cauchy finite product window restriction)**.** *Restricting a coordinate read from an accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packet to any displayed synchronized* $\mathsf{StreamName}^{\uparrow}$ *subwindow preserves the* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible coordinate readback and the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget ledger, provided the finite index row still names the coordinate slot. The restriction exposes no host product coordinate, selected component representative, quotient stream equality, trichotomy, or ambient completeness datum.*

Proof. Read the product projection route through chapter 2076. The coordinate slot remains available exactly when the finite index row still names it.

Next restrict the synchronized window row. The $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ routes fix the displayed subwindow used by the coordinate read, so the restricted read uses the same component source and the same regular-rational readback route.

Apply coordinate stability from Definition 2009.12. Since the product-budget ledger and projection route are unchanged, the coordinate remains a RegSeqRat-compatible readback through the same DyadicRatCore product-budget data.

The restricted subwindow is therefore a local view of the displayed finite-product packet. It cannot introduce a host finite product, selected representative, quotient stream equality, trichotomy, or ambient completeness datum. □

Theorem 2009.14 (Regular Cauchy finite product coordinate pairing)**.** *Let two coordinate reads be displayed from the same accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packet. If the two reads share the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget ledger, and the finite index rows naming their coordinate slots, then they form a jointly* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible pair. The pair is read only through the product-projection dependency route of chapter 2076; it exports no host product coordinates, selected component representatives, quotient stream equality, trichotomy, or ambient completeness datum.*

Proof. Start with the two displayed coordinate reads from the same finite-product packet. Apply coordinate stability, Definition 2009.12, to each read along the common product source, synchronized windows, product-budget ledger, regular-rational readback, and projection route.

Next apply the window restriction theorem, Definition 2009.13. Because both reads share the same $\mathsf{StreamName}^{\uparrow}$ window family and $\mathsf{DyadicRatCore}^{\uparrow}$ product-budget ledger, restricting to any displayed common subwindow preserves the regular-rational readback for each coordinate.

Finally use the product-projection dependency route in chapter 2076. The finite index rows keep the two coordinate slots named inside the same packet, so the two preserved readbacks assemble as a jointly $\mathsf{RegSeqRat}^{\uparrow}$-compatible pair.

The assembled pair contains only the displayed product packet, synchronized windows, dyadic budget, finite index rows, projection route, and regular-rational handoffs. No host product coordinate, selected representative, quotient stream equality, trichotomy, or ambient completeness row is introduced. □

Theorem 2009.15 (Regular Cauchy finite product coordinate gluing)**.** *Let a finite family of coordinate reads share the same* $\mathsf{StreamName}^{\uparrow}$ *window family, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget ledger, and finite index rows naming their coordinate slots. If each read is accepted through the product-projection dependency route of chapter 2076 and remains* $\mathsf{RegSeqRat}^{\uparrow}$*-compatible under the coordinate stability boundary, then the family glues into one accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packet whose projections recover the original coordinate reads. The glued packet exports only the displayed windows, dyadic product-budget ledger, finite index rows, projection routes, and regular-rational handoffs; it introduces no host product, quotient equality, trichotomy, selected component representative, or ambient completeness datum.*

Proof. Induct over the finite index rows. The empty family is the displayed empty finite-product packet, whose window family, product-budget ledger, and projection route contain no hidden host product data.

For the step, assume the already indexed coordinate reads have been glued into an accepted finite-product packet. Add the next coordinate read using the same $\mathsf{StreamName}^{\uparrow}$ windows and the same $\mathsf{DyadicRatCore}^{\uparrow}$ product-budget ledger.

Apply coordinate stability from Definition 2009.12 to keep the new read on the displayed regular-rational route. Then apply coordinate pairing from Definition 2009.14 to attach the new coordinate to the already glued packet through the shared projection dependency route.

The induction preserves the finite index ledger at each step, so every projection of the resulting packet recovers the coordinate read placed at that index. Since each step uses only the displayed windows, dyadic budget, finite index rows, projection route, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the glued packet gains no host product, quotient stream equality, trichotomy, selected component representative, or ambient completeness row. □

Theorem 2009.16 (Regular-Cauchy finite product downstream projection route)**.** *Every downstream* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *packet reads the finite product packet's index row, component source rows, synchronized window family, dyadic product-budget ledger, final readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal through the projection certificate of chapter 2076. The projection route exports no host product object, selected component representative, quotient stream equality, countable choice schedule, or ambient completeness principle.*

Proof. Project the finite product carrier through its displayed index, source, synchronized window, dyadic budget, readback, and Real seal rows. These are the rows already consumed by the product projection dependency route.

Now read the downstream projection packet of chapter 2076. Its left and right coordinate reads are admitted only after the finite product route has supplied the shared product index, component sources, windows, budget, and final readback.

The terminal $\mathsf{Real}^{\uparrow}$ seal remains downstream of that route and upstream of the local projection naming rows. Since no coordinate in the finite product packet carries a host product, selected component representative, quotient stream equality, choice schedule, or ambient completeness object, the downstream projection consumer cannot export any of them. □

Chapter 2010

Dedekind Cauchy Sandwich NameCert Construction

Following Bishop and Bridges on constructive real representations, the $\mathsf{DedekindCauchySandwich}^{\uparrow}$ packet records the finite sandwich used when located-cut evidence is compared with a regular-Cauchy name before the terminal real seal is consumed. It sits between the lower and upper cut rows of chapter 404, the comparison route of chapter 1424, and the interval enclosure surface of chapter 1709. Its downstream handoff is the regular-rational route of chapter 301, the finite stream windows of chapter 126, and the terminal seal of chapter 38. The chapter names only finite carrier, regularization, transport, replay, provenance, and local NameCert rows; it does not claim a quotient cut, selected supremum, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2010.1 (Dedekind Cauchy sandwich carrier)**.** *A* $\mathsf{DedekindCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located-cut row, recording finite dyadic witnesses below the inspected located cut.*
2. *U is the upper located-cut row, recording finite dyadic witnesses above the same inspected located cut.*
3. *I is the dyadic interval enclosure row, binding each visible unary precision to an interval whose lower endpoint is certified by L and whose upper endpoint is certified by U.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened only after the corresponding lower, upper, and interval rows have been displayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row assembled from the displayed stream windows and dyadic enclosure budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through L, U, I, W, R.*
7. *H records componentwise* hsame *transport for the lower row, upper row, interval enclosure, stream windows, regular handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$L, U \longmapsto I \longmapsto W \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{DedekindCauchySandwich}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCauchySandwich}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, I, W, R, E, H, C, P, N$. It has no coordinate for quotient cuts, selected suprema, trichotomy, countable choice, host completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.DedekindCauchySandwichUp.DedekindCauchySandwichTasteGate single carrier alignment`

Theorem 2010.2 (Dedekind Cauchy sandwich regularization)**.** *For every accepted* $\mathsf{DedekindCauchySandwich}^{\uparrow}$ *carrier*

$$S = (L, U, I, W, R, E, H, C, P, N),$$

each visible unary precision exported by S has a displayed $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R squeezed between the lower and upper located-cut rows L and U through the dyadic interval enclosure row I and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W. The regular-Cauchy read is obtained without quotient-cut equality, selected suprema, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 2010.1. The only lower and upper cut data available to a precision request are the displayed rows L and U.

The interval row I binds those two rows at the same visible unary precision. Its lower endpoint is certified by L, its upper endpoint is certified by U, and the carrier has no separate coordinate from which a detached bound could be read.

The stream and regular-Cauchy rows are downstream of that enclosure. The finite windows W are opened after I, and the regular handoff R is assembled from those windows and the same dyadic enclosure budgets.

Finally, H, C, P, N transport, replay, source, and name the displayed route only. Since the carrier contains no quotient cut, selected supremum, trichotomy witness, countable-choice row, or ambient completeness coordinate, every exported regular-Cauchy read is squeezed between the displayed lower and upper rows through I and W. □

Theorem 2010.3 (Dedekind Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCauchySandwic}$ *carrier*

$$S = (L, U, I, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DedekindCauchySandwich}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed lower located-cut row, upper located-cut row, dyadic interval enclosure, stream windows, regular-Cauchy handoff, Real seal, transport, replay, provenance, and local naming rows;*

2. *lower-upper disjointness, requiring the lower and upper rows to appear as separated faces of the same finite located-cut observation before any interval or stream row is read;*

3. *locatedness, requiring each visible unary precision to display a dyadic enclosure row I whose endpoints are sourced from L and U;*

4. *regularization, requiring R to be assembled from the finite windows W opened over the displayed enclosure rather than from a detached Cauchy representative;*

5. *classifier transport, requiring componentwise* hsame *transport to preserve the lower row, upper row, enclosure row, stream windows, handoff row, Real seal, replay, provenance, and local NameCert row together;*

6. *ledger exclusion, requiring the packet to expose no quotient cut, selected supremum, trichotomy witness, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, Lean verification, or bridge credential.*

No $\mathsf{DedekindCauchySandwich}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is the projection of the packet $S = (L, U, I, W, R, E, H, C, P, N)$ to its ten displayed coordinates. These coordinates are the whole accepted carrier inventory.

Lower-upper disjointness and locatedness are read through L, U, I. The two located-cut faces are displayed before the interval enclosure is available, and each visible unary precision receives its enclosure from that same pair of rows.

Regularization is the projection to W and R. The stream windows are opened over the displayed enclosure, and the regular-Cauchy handoff is downstream of those windows rather than a free-standing representative.

Classifier transport is the projection to H together with the replay and provenance rows C and P. These rows preserve the displayed coordinates componentwise and do not introduce a second classifier surface.

The ledger exclusion row is therefore exhaustive. Since no excluded coordinate appears in the carrier, the local NameCert can only certify the displayed sandwich route and cannot export quotient-cut equality, selected suprema, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCauchySandwich}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCauchySandwich}^{\uparrow}$ packet is generated from lower and upper located-cut rows, dyadic interval enclosure rows, finite StreamName windows, a RegSeqRat handoff, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DedekindCauchySandwich}^{\uparrow}}$, seed)

Scope closed: Definition 2010.1, Definition 2010.2, and Definition 2010.3 seed the finite located-cut to regular-Cauchy sandwich packet used by Real representation comparison.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DedekindCauchySandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient cuts, selected suprema, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regularization, classifier transport, ledger exclusion, and local NameCert proofs for BEDC.Derived.DedekindCauchySandwichUp.

Chapter 2011

A Concrete Naming Certificate for $\mathsf{CutRegularization}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{CutRegularization}^{\uparrow}$ packet records the finite route from rounded located cut observations to a regular-Cauchy $\mathsf{Real}^{\uparrow}$ name. It sits after the rounded lower and upper presentation of chapter 436, the located dyadic window surface of chapter 404, and the approximation interface of chapter 1975. Its dependency surface is the dyadic tolerance ledger of chapter 291, the stream-window interface of chapter 126, the regular rational handoff of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names only finite carrier rows and their non-escape boundary; it is not a quotient of cuts, a selected supremum, a trichotomy principle, a host-completion theorem, or an appeal to choice.

Definition 2011.1 (Cut regularization carrier)**.** *A* $\mathsf{CutRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, L, A, \Delta, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the rounded* $\mathsf{DedekindCut}^{\uparrow}$ *source row, exposing the lower and upper rational observations as finite BEDC data.*
2. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *located-window row, binding each inspected precision request to lower and upper dyadic witnesses.*
3. *A is the* $\mathsf{LocatedCutApproximation}^{\uparrow}$ *approximation row, storing the visible midpoint, endpoint, or bracket readback obtained from D and L.*
4. *Δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint slack, and width comparisons for the regularization route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the finite brackets are read in precision order.*
6. *S is the stabilization row requiring the displayed windows to be nested or refined enough to feed the regular-Cauchy modulus demanded downstream.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from A, Δ, W, S.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only through the regular-Cauchy handoff R.*
9. *H records componentwise* hsame *transport for all source, window, tolerance, handoff, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$D, L \longmapsto A \longmapsto \Delta \longmapsto W \longmapsto S \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CutRegularization}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedCutApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CutRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, A, \Delta, W, S, R, E, H, C, P, N$. It has no coordinate for quotient-cut equality, selected suprema, trichotomy, host real equality, ambient completeness, Quot.sound*, propext, classical choice, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CutRegularizationUp.TasteGate.CutRegularizationTasteGate single carrier alignment`

Theorem 2011.2 (Cut regularization NameCert obligations)**.** *For every accepted* $\mathsf{CutRegularization}^{\uparrow}$ *carrier*

$$Q = (D, L, A, \Delta, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CutRegularization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed rounded cut source, located window, approximation row, dyadic tolerance ledger, stream windows, stabilization row, regular-Cauchy handoff, Real seal, and structural rows H, C, P, N;*
2. *cut-source exposure, requiring every regularization request to read the rounded* $\mathsf{DedekindCut}^{\uparrow}$ *source only through the displayed lower and upper rational observations;*
3. *located-window exposure, requiring the* $\mathsf{LocatedCut}^{\uparrow}$ *row L to bind each inspected precision to finite lower and upper dyadic witnesses before any stream or regular-Cauchy row is consumed;*
4. *approximation exactness, requiring the* $\mathsf{LocatedCutApproximation}^{\uparrow}$ *row A to store only finite bracket data obtained from D, L and the tolerance ledger Δ;*
5. *tolerance accounting, requiring Δ to record the dyadic radius and endpoint slack used by each displayed stream window and stabilization read;*
6. *stream stabilization, requiring W and S to expose the finite* $\mathsf{StreamName}^{\uparrow}$ *windows whose refinement feeds the regular-Cauchy modulus;*
7. *regular-Cauchy handoff, requiring R to be the only route by which the rounded located cut data becomes a* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
8. *Real-seal factorization, requiring every downstream read of E to pass through D, L, A, Δ, W, S, R and the structural rows;*
9. *transport, replay, and provenance non-escape, requiring H, C, P, N to preserve and name exactly the displayed rows, with no quotient-cut equality, selected supremum, trichotomy, host real equality, ambient completeness,* Quot.sound*, propext, classical choice, Lean verification, or bridge credential.*

No cut-regularization obligation is stored outside these rows.

Proof. Project the accepted packet $Q = (D, L, A, \Delta, W, S, R, E, H, C, P, N)$. Carrier admission exposes exactly the rounded cut source, located window, approximation row, dyadic tolerance

ledger, stream windows, stabilization row, regular-Cauchy handoff, Real seal, and four structural rows.

The cut-source and located-window obligations are the projections to D and L. The source row supplies the rounded lower and upper rational observations, and the located row binds each inspected precision to finite dyadic witnesses before any Cauchy-facing route is read.

Approximation exactness and tolerance accounting are the projections to A and Δ. The approximation row stores only bracket data read from the displayed source and located windows, while the tolerance ledger records the dyadic radius, endpoint slack, and width comparisons used by that read.

The stream and stabilization obligations are the projections to W and S. The stream row exposes the finite windows in precision order, and the stabilization row records the refinement needed to feed the regular-Cauchy modulus without selecting a quotient cut or a supremum.

Finally, the regular handoff, Real seal, and structural rows exhaust the boundary. The row R is downstream of D, L, A, Δ, W, S, the seal E is downstream of R, and H, C, P, N transport, replay, name, and source only the displayed rows. Hence no quotient-cut equality, selected supremum, trichotomy, host real equality, ambient completeness, Quot.sound, propext, classical choice, Lean verification, or bridge credential is exported. □

Theorem 2011.3 (Cut regularization window to regular-Cauchy)**.** *For every accepted* $\mathsf{CutRegularization}^{\uparrow}$ *carrier*

$$Q = (D, L, A, \Delta, W, S, R, E, H, C, P, N),$$

every $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* Q *factors through the displayed finite route*

$$\mathsf{DedekindCut}^{\uparrow} \longrightarrow \mathsf{LocatedCut}^{\uparrow} \longrightarrow \mathsf{LocatedCutApproximation}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow}$$

by reading the rows $D, L, A, \Delta, W, S, R, H, C, P, N$*. In particular, the handoff uses the dyadic tolerance ledger and stream stabilization rows, not quotient-cut equality, host real equality, a selected supremum, trichotomy, ambient completeness,* Quot.sound*, propext, or classical choice.*

Proof. Begin with a regular-Cauchy consumer. Carrier projection exposes the handoff row R only after the rounded cut source D, the located window L, the approximation row A, the dyadic tolerance ledger Δ, the stream windows W, and the stabilization row S have been displayed.

The route inspection is finite. The rounded source supplies lower and upper rational observations, the located row binds those observations to a precision window, the approximation row records the finite bracket, and the dyadic ledger stores the radius and slack comparisons used by that bracket.

The stream row then presents the bracket sequence as visible windows, and the stabilization row records the refinement relation needed for the regular-Cauchy modulus. Thus the handoff row R is downstream of the displayed stream route rather than a detached representative.

Componentwise transport and replay through H and C preserve the same order, while P and N only name the visible packet. Since no carrier coordinate stores quotient-cut equality, host real equality, selected suprema, trichotomy, ambient completeness, Quot.sound, propext, or classical choice, no $\mathsf{RegSeqRat}^{\uparrow}$ read exported by the packet can factor through those excluded sources. □

Theorem 2011.4 (Cut regularization Real seal boundary)**.** *For every accepted* $\mathsf{CutRegularization}^{\uparrow}$ *carrier*

$$Q = (D, L, A, \Delta, W, S, R, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer of the terminal seal* E *receives only the displayed finite route*

$$D, L, A, \Delta, W, S, R, E, H, C, P, N.$$

The seal is not a Dedekind construction of $\mathsf{Real}^{\uparrow}$*, not a quotient-cut completion, not a selected supremum, not a trichotomy witness, not a host-completion theorem, not a choice object, and not a* Quot.sound *or propext bridge.*

Proof. Start from the terminal seal E. In the carrier, E is not an independent completion coordinate; it is the endpoint of the displayed route through the rounded cut source, located window, finite approximation, dyadic tolerance ledger, stream windows, stabilization row, and regular handoff.

Trace the route backward. The Real-facing read reaches E only after R has been read, R only after W and S have exposed the finite stabilized windows, and those windows only after A has been checked against D, L and Δ.

The structural coordinates do not add a second seal source. The transport row H is componentwise, the replay row C displays the same route order, and the provenance and local certificate rows P, N name this packet rather than importing a host completion principle.

Thus every Real-facing consumer is confined to $D, L, A, \Delta, W, S, R, E, H, C, P, N$. The excluded Dedekind construction, quotient-cut completion, selected-supremum, trichotomy, host-completion, choice, Quot.sound, and propext readings are outside the carrier inventory. □

Closure status for $\mathsf{NameCert}_{\mathsf{CutRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{CutRegularization}^{\uparrow}$ packet is generated from a rounded DedekindCut source, a LocatedCut window row, a LocatedCutApproximation bracket row, a DyadicRatCore tolerance ledger, StreamName windows, a stabilization row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CutRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 2011.1, Definition 2011.2, Definition 2011.3, and Definition 2011.4 seed the finite route from rounded located cut observations to regular-Cauchy RealUp names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CutRegularization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a quotient construction of $\mathsf{Real}^{\uparrow}$, selected suprema, trichotomy, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-source exposure, located-window exposure, approximation exactness, tolerance accounting, stream stabilization, regular-Cauchy handoff, Real-seal factorization, transport/replay/provenance coverage, and non-escape for BEDC.Derived.CutRegularizationUp.

Chapter 2012

A Concrete Naming Certificate for $\mathsf{FormalBallCompletion}^{\uparrow}$

Following the Edalat–Heckmann formal-ball presentation of metric completion and Vickers formal topology, the $\mathsf{FormalBallCompletion}^{\uparrow}$ packet records constructive completion as a rounded directed family of formal balls with a completion-facing classifier boundary. It sits after the formal-ball surface of chapter 1510, the rounded ideal completion surface of chapter 1883, the metric carrier of chapter 57, the completion boundary of chapter 95, the finite stream-window interface of chapter 126, the dyadic radius ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter gives Real-and-Completion consumers a finite NameCert surface for formal-ball completion without claiming quotient completion, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Definition 2012.1 (Formal-ball completion carrier)**.** *A* $\mathsf{FormalBallCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, B, I, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including point histories, distance witnesses, and the point classifier used by every formal-ball center.*
2. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *family row, exposing the finite formal balls whose centers are read through M and whose radii are accounted for by the dyadic ledger.*
3. *I is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *handoff row, recording roundedness, directedness, and cofinal refinement for the displayed formal-ball family.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording positive-radius witnesses, strict comparisons, radius sums, and refinement budgets used by B and I.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, binding every inspected center observation and ideal-membership query to finite stream data.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance-seal row reached only through the displayed formal-ball and rounded-ideal route.*
7. *T records componentwise* hsame *transport for the metric source, formal-ball family, rounded ideal handoff, dyadic ledger, stream windows, real seal, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay of the route*

$$M \longmapsto B \longmapsto I \longmapsto D, S \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, I, D, S, R, T, C, P, N$. It has no coordinate for quotient completion equality, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.FormalBallCompletionUp.FormalBallCompletionTasteGate single carrier alignment`

Theorem 2012.2 (Formal-ball completion NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallCompletion}^{\uparrow}$ *carrier*

$$K = (M, B, I, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FormalBallCompletion}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed metric source, formal-ball family, rounded-ideal handoff, dyadic ledger, stream windows, real seal, and structural rows T, C, P, N;*
2. *formal-ball family membership, requiring every completion-facing neighbourhood query to be read as a displayed* $\mathsf{FormalBall}^{\uparrow}$ *member over M with radius data accounted for in D;*
3. *rounded-directed ideal handoff, requiring the row I to be the only route from formal-ball membership to completion-facing roundedness, directedness, and cofinal refinement;*
4. *classifier transport, requiring accepted packets to be compared componentwise through T and replayed through C without adding a second classifier;*
5. *ledger non-escape, requiring every downstream read to pass through $M, B, I, D, S, R, T, C, P, N$ and to export no quotient completion, selected limit, countable-choice selector, open-cover compactness proof, ambient completeness datum, Lean verification credential, or bridge credential.*

No formal-ball completion obligation is stored outside these rows.

Proof. Project the accepted packet $K = (M, B, I, D, S, R, T, C, P, N)$. Carrier admission exposes exactly the metric source, formal-ball family, rounded-ideal handoff, dyadic ledger, stream windows, real seal, transport, replay, provenance, and local naming row.

Formal-ball family membership is the projection to B together with its source rows M, D, S. The metric row supplies the center classifier, the dyadic row supplies radius comparisons, and the stream row supplies the finite observation windows used by each displayed ball.

The rounded-directed handoff is the projection to I. It is the only row that packages roundedness, directedness, and cofinal refinement for the displayed family, so a completion-facing consumer must pass through it before reading the seal row R.

Transport and replay are confined to T and C, while P and N only record provenance and local naming. Since the carrier has no coordinate for quotient completion equality, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the NameCert surface. □

Theorem 2012.3 (Formal-ball completion rounded ideal handoff)**.** *For every accepted* $\mathsf{FormalBallCompletion}^{\uparrow}$ *carrier*

$$K = (M, B, I, D, S, R, T, C, P, N),$$

each completion-facing neighbourhood request factors through the displayed handoff

$$\mathsf{FormalBall}^{\uparrow} \longrightarrow \mathsf{RoundedIdealCompletion}^{\uparrow} \longrightarrow \mathsf{Completion}^{\uparrow}$$

by reading B, I, D, S, T, C, P, N *over the same metric source* M*. The handoff exports a rounded directed formal-ball family and its finite replay boundary, not a quotient class of Cauchy names, not a selected representative, not a selected limit, and not an ambient-completeness witness.*

Proof. Begin with the formal-ball family row. The accepted packet exposes B only over the metric source M, with radius comparisons recorded in D and center observations read through S.

Pass next to the rounded ideal row I. This row records the roundedness, directedness, and cofinal-refinement data needed to regard the displayed formal balls as completion data, and it is reached before any completion-facing seal can be read.

The structural rows preserve the same route. Componentwise transport through T keeps M, B, I, D, S, R aligned, replay through C preserves the handoff order, and P, N source and name the same packet.

No row stores quotient completion equality, a selected Cauchy representative, a selected limit, countable choice, open-cover compactness, or ambient completeness. Hence the handoff is exactly the displayed rounded directed formal-ball boundary. □

Theorem 2012.4 (Formal-ball completion classifier boundary)**.** *If two accepted* $\mathsf{FormalBallCompletion}^{\uparrow}$ *carriers are related by componentwise* hsame *transport through their transport rows, then their classifier-visible data agree on the metric source, formal-ball family membership, rounded-directed ideal handoff, dyadic radius ledger, stream windows, real seal, replay route, provenance, and local name.*

Consequently classifier transport preserves the formal-ball family membership row, the rounded-directed handoff row, and the ledger non-escape boundary for the completion surface.

Proof. Project the two accepted carriers. The classifier-visible coordinates are exactly the metric source, formal-ball family, rounded ideal handoff, dyadic ledger, stream windows, real seal, replay route, provenance, and local name.

Apply componentwise hsame transport through the two transport rows. Formal-ball membership is compared over the same metric source, radius comparisons remain in the dyadic ledger, and finite center windows remain in the stream row.

The rounded-directed handoff is transported with the family it certifies. Thus roundedness, directedness, and cofinal refinement remain attached to the same displayed formal-ball data.

Replay, provenance, and naming add no independent classifier. Since the excluded completion principles have no carrier coordinate, classifier transport preserves both the handoff and the non-escape boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalBallCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2012.1, Definition 2012.2, Definition 2012.3, and Definition 2012.4 seed the finite formal-ball completion surface for rounded directed constructive completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalBallCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formal-ball family membership, rounded-directed ideal handoff, classifier transport, ledger non-escape, replay stability, provenance, and local naming for BEDC.Derived.FormalBallCompletionUp

Constructive story: A FormalBallCompletion$^{\uparrow}$ packet is generated from a MetricSpace source, a FormalBall family, a RoundedIdealCompletion handoff, a DyadicRatCore radius ledger, StreamName windows, a RealUp tolerance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2013

A Concrete Naming Certificate for $\mathsf{FeketeSubadditive}^{\uparrow}$

Following Fekete and Bishop–Bridges constructive analysis, the $\mathsf{FeketeSubadditive}^{\uparrow}$ packet records the finite subadditive normalized-average route used by Real and series-facing certificates before a lower-bound readback is handed to regular rational and Real seals. It sits beside the series certificate of chapter 121, the real-sequence limit route of chapter 1413, the dyadic arithmetic core of chapter 291, the regular rational sequence boundary of chapter 301, and the terminal Real surface of chapter 38. The packet names a displayed finite ledger for subadditive averages; it does not assert an ambient least infimum operator or a host-level completeness theorem.

Definition 2013.1 (Fekete subadditive carrier)**.** *A* $\mathsf{FeketeSubadditive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, A, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the real sequence source row. It records the inspected source windows and the finite indexing data from which normalized averages are read.*
2. *A is the subadditivity witness ledger. Each row records a displayed decomposition $m+n$ together with the finite comparison cell witnessing that the source value at $m+n$ is bounded by the visible sum of the source values at m and n.*
3. *D is the dyadic normalized-average comparison ledger. It stores the rational denominator windows, dyadic comparison tolerances, and finite average rows used to compare inspected values of S.*
4. *L is the located lower-bound readback row. It records the finite lower-bound and located-infimum data actually displayed by D, without exposing a host-level infimum operator.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained only after the normalized-average ledger and lower-bound readback have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff has been displayed.*
7. *H records componentwise* hsame *transport for the source, subadditivity, dyadic average, lower-bound, regular handoff, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through*

$$S, A \longmapsto D \longmapsto L \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FeketeSubadditive}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FeketeSubadditive}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for ambient completeness, trichotomy, countable choice, quotient stream equality, host real equality, host-level infimum operators, or a selected external limit.

Theorem 2013.2 (Fekete subadditive NameCert obligations)**.** *For every accepted* $\mathsf{FeketeSubadditive}^{\uparrow}$ *carrier $F = (S, A, D, L, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FeketeSubadditive}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. *source-window exposure for S, including every index and normalized-average denominator inspected by the certificate;*
3. *subadditivity ledger exactness for each decomposition row recorded in A;*
4. *dyadic normalized-average comparison stability for D under the recorded* hsame *transports;*
5. *located lower-bound readback, requiring L to be read only from the displayed average comparisons in D;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring R and E to factor through S, A, D, L;*
7. *transport, continuation replay, provenance, and local naming for exactly the rows recorded in H, C, P, N.*

No Fekete subadditive NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.FeketeSubadditiveUp.FeketeSubadditiveTasteGate single carrier alignment`

Proof. Project the accepted carrier $F = (S, A, D, L, R, E, H, C, P, N)$. The projection exposes the source sequence windows, subadditivity witness ledger, dyadic average comparisons, lower-bound readback, regular rational handoff, Real seal, and structural rows.

The order-theoretic content is finite. The rows S and A list only the inspected source windows and decomposition witnesses, while D stores the normalized-average comparisons that can be replayed for the requested denominator windows.

The lower-bound readback is then forced through L. Since L is read from D, the certificate does not appeal to a host-level infimum operator, trichotomy, or an ambient completeness theorem.

Finally R and E are downstream of the displayed ledger. Componentwise transport in H preserves the same route, and C, P, N only replay, source, and name that route. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 2013.3 (Fekete subadditive Cauchy route)**.** *Let $F = (S, A, D, L, R, E, H, C, P, N)$ be an accepted* $\mathsf{FeketeSubadditive}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by F factors through the finite route*

$$S, A \longmapsto D \longmapsto L \longmapsto R \longmapsto E.$$

Consequently a downstream Cauchy, series, or real-sequence-limit consumer receives only the displayed source windows, subadditivity witnesses, dyadic normalized-average comparisons, located lower-bound readback, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport package, replay route, provenance, and local naming row.*

Proof. Start with a Real-facing read. Carrier projection shows that the only available numerical rows are the source windows S, the subadditivity witnesses A, the dyadic comparison ledger D, and the lower-bound readback L.

The Cauchy information is produced by the normalized-average route. The ledger D supplies finite dyadic comparisons between inspected averages, and L records the lower-bound readback that those comparisons actually expose.

The handoff to R and E does not add analytic data. It presents the displayed lower-bound route as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff and then as a Real-facing seal.

The structural rows preserve the same route under transport and replay. Therefore a downstream Cauchy, series, or real-sequence-limit consumer cannot read ambient completeness, trichotomy, countable choice, quotient stream equality, host real equality, host-level infimum operators, or an external selected limit from the $\mathsf{FeketeSubadditive}^{\uparrow}$ packet. □

Theorem 2013.4 (Fekete subadditive finite-prefix bound)**.** *Let $F = (S, A, D, L, R, E, H, C, P, N)$ be an accepted* $\mathsf{FeketeSubadditive}^{\uparrow}$ *carrier, and let a downstream* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *series, or real-sequence-limit consumer request a finite read of the lower-bound row L. Then the read must expose a finite prefix containing the source windows S, the subadditivity ledger A, and the dyadic normalized-average ledger D before any lower-bound row is available.*

In particular, the prefix bound is determined by the displayed route

$$S, A \longmapsto D \longmapsto L$$

and by the downstream continuation

$$L \longmapsto R \longmapsto E.$$

No host infimum, trichotomy, quotient stream equality, selected external limit, ambient Real completeness, countable choice, or host real equality can enter the prefix bound.

Proof. Project the accepted carrier. The carrier definition displays the rows $S, A, D, L, R, E, H, C, P, N$, and the local NameCert obligations force the source windows S, the subadditivity ledger A, and the normalized-average comparison ledger D to be present before L is read.

The finite prefix is therefore obtained by truncating the displayed carrier at the last row used by the requested lower-bound read. Since every comparison in D cites only inspected windows of S and decomposition rows of A, the truncation contains exactly the finite data needed to justify the lower-bound row.

Apply the Cauchy route theorem to the downstream part of the read. The handoff rows R and E consume the already displayed prefix S, A, D, L and do not add analytic data of their own.

Thus the requested consumer receives a finite prefix bound through the displayed ledger alone. The excluded host objects have no coordinate in the carrier, no NameCert obligation row, and no continuation route into L, R, or E. □

Closure status for $\mathsf{NameCert}_{\mathsf{FeketeSubadditive}^{\uparrow}}$.

Constructive story: A $\mathsf{FeketeSubadditive}^{\uparrow}$ packet is generated from real sequence source windows, a subadditivity witness ledger, dyadic normalized-average comparison rows, a located lower-bound readback, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FeketeSubadditive}^{\uparrow}}$, seed)

Scope closed: Definition 2013.1, Definition 2013.2, and Definition 2013.3 seed the finite subadditive normalized-average carrier, its local NameCert obligations, and its Cauchy-facing readback boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FeketeSubadditive}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ambient completeness, trichotomy, countable choice, quotient stream equality, host real equality, host-level infimum operators, selected external limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subadditivity ledger exactness, dyadic normalized-average comparison stability, located lower-bound readback, and Cauchy-route non-escape for BEDC.Derived.FeketeSubadditiveUp.

Chapter 2014

A Concrete Naming Certificate for $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ packet records the finite Cauchy criterion route by which displayed partial-sum windows of a real series are sealed as regular Cauchy data before a $\mathsf{Real}^{\uparrow}$ endpoint is read. It refines the real-series surface of chapter 1581, uses the regular rational handoff of chapter 301, reads term and partial-sum windows through chapter 126, accounts for tolerances through chapter 291, and lands at the Bishop regular-Cauchy limit seal of chapter 1821. The packet names a criterion over finite displayed rows; it does not import host infinite sums, selected limits, quotient equality of real streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2014.1 (Real series Cauchy criterion carrier)**.** *A* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, D, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the real-valued term stream, accessed only through selected* $\mathsf{StreamName}^{\uparrow}$ *windows and the displayed term rows already admitted by the* $\mathsf{RealSeries}^{\uparrow}$ *surface.*
2. *S is the partial-sum window row, listing the finite initial segments, later comparison windows, and the displayed subtraction or difference reads used to compare two partial sums.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested dyadic error bounds, finite addition errors, and comparison radii consumed by the criterion.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same partial-sum windows and dyadic tolerance ledger.*
5. *M is the shared tail-threshold row: for each displayed tolerance request, one threshold governs every pair of partial-sum windows whose indices lie beyond it.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after T, S, D, R, M have been displayed and routed through the Bishop regular-Cauchy limit seal.*
7. *H stores componentwise* hsame *transport for the term stream, partial-sum window, dyadic ledger, regular readback, shared threshold, endpoint seal, continuation, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from term windows to partial sums, from partial sums to dyadic comparisons, from dyadic comparisons to the shared threshold, and from that threshold through regular readback to the endpoint seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopRegularCauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeriesCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, D, R, M, E, H, C, P, N$. It has no coordinate for a host infinite sum, selected sequence limit, quotient equality of real streams, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RealSeriesCauchyCriterionUp.TasteGate.RealSeriesCauchyCriterionTasteGate`
`single carrier alignment`

Theorem 2014.2 (Real series Cauchy criterion obligation surface)**.** *For every accepted* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *carrier $K = (T, S, D, R, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealSeriesCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the ten displayed rows;*
2. *partial-sum window coverage, requiring every term read and every two-window comparison to factor through T and S;*
3. *Cauchy-tail threshold stability, requiring the shared row M to serve all later partial-sum comparisons for the displayed tolerance request;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff, requiring R and E to be downstream of the same finite threshold route and the Bishop regular-Cauchy limit seal;*
5. *ledger nonescape, requiring every tolerance, transport, continuation, provenance, and local naming read to remain inside D, H, C, P, N.*

No obligation row asserts host summability, selected infinite sums, quotient equality of real streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet $K = (T, S, D, R, M, E, H, C, P, N)$. Carrier admission exposes exactly the term-stream row, partial-sum window row, dyadic ledger, regular readback, shared threshold, endpoint seal, and structural rows.

For a displayed tolerance request, inspect the finite partial-sum windows in S. The term data used to build those windows are read through T, and the only admissible comparison information is the dyadic accounting stored in D.

The shared threshold row M is then replayed. Its stability claim is finite: the same displayed threshold is reused for all later partial-sum comparisons requested by the packet, and no unrecorded global convergence object is consulted.

Finally the handoff to R and E follows the same route through the Bishop regular-Cauchy limit seal. The rows H, C, P, N transport, replay, source, and name that finite route, so the listed obligations exhaust the NameCert surface and leave no coordinate for host summability or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeriesCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ packet is generated from a real-valued term stream, finite partial-sum windows, a dyadic tolerance ledger, RegSeqRat readback, one shared tail-threshold row, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealSeriesCauchyCriterion}\uparrow}$, seed)

Scope closed: Definition 2014.1 and Definition 2014.2 seed the finite Cauchy criterion route from displayed real-series partial sums to regular-Cauchy Real seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealSeriesCauchyCriterion}\uparrow}$, unformalized)

Lean target: `BEDC.Derived.RealSeriesCauchyCriterionUp`

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected sequence limits, quotient equality of real streams, countable choice, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: A checked carrier theorem can encode the same term-stream, partial-sum window, dyadic ledger, shared threshold, RegSeqRat handoff, Real seal, transport, continuation, provenance, and local NameCert rows.

Chapter 2015

A Concrete Naming Certificate for $\mathsf{CauchySeriesCriterion}^{\uparrow}$

From Cauchy through Bishop–Bridges constructive analysis, the Cauchy criterion for series is the finite tail-comparison surface that turns partial sums into regular completion data. The $\mathsf{CauchySeriesCriterion}^{\uparrow}$ packet sits between the uniform criterion of chapter 1736, the majorant route of chapter 1774, and the absolute-convergence certificate of chapter 1700. It reads the underlying $\mathsf{Series}^{\uparrow}$ rows of chapter 121, hands finite partial-sum comparisons to the $\mathsf{RegSeqRat}^{\uparrow}$ surface of chapter 301, and exposes the completion-facing endpoint only through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records a criterion route; it does not import a host infinite-sum operator, countable choice, quotient equality of streams, or ambient completeness for real numbers.

Definition 2015.1 (Cauchy series criterion carrier)**.** *A* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, S, W, Q, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Series}^{\uparrow}$ *term row, read only through the finite term observations admitted by the series packet.*
2. *S is the partial-sum thread, listing selected finite sums, later comparison windows, and the displayed subtraction rows used to compare two partial sums.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family used to open the selected terms, partial sums, and endpoint-facing approximations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and to the rational comparison data extracted from them.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested dyadic bounds, addition budgets, and comparison radii.*
6. *M is the shared Cauchy-tail threshold row: each displayed tolerance request receives one finite threshold that governs all later partial-sum comparisons in the packet.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, S, W, Q, D, M have been displayed.*
8. *H stores componentwise* hsame *transport for the term row, partial-sum thread, window family, regular readback, tolerance ledger, threshold row, real seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from term rows to partial sums, from partial sums to regular readback, from regular readback to dyadic tolerances, from tolerances to the shared threshold, and from that threshold to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchySeriesCriterion}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySeriesCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, W, Q, D, M, E, H, C, P, N$. It has no coordinate for a host infinite sum, selected limit, quotient stream equality, countable-choice selector, trichotomy, ambient real completeness, Lean verification, or standard bridge.

Lean4 ✓ `BEDC.Derived.CauchySeriesCriterionUp.CauchySeriesCriterionTasteGate single carrier alignment`

Theorem 2015.2 (Cauchy series criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *carrier $K = (A, S, W, Q, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySeriesCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed series term row, partial-sum thread, selected windows, regular readback, dyadic ledger, shared tail threshold, real seal, transport, replay, provenance, and local naming rows;*

2. *partial-sum thread exactness, requiring every finite sum observation and every later-window comparison to pass through A, S, W, Q, D before the threshold row is read;*

3. *Cauchy-tail stability, requiring M to be the displayed threshold used for all accepted comparisons of partial sums whose indices lie beyond the requested tolerance boundary;*

4. *regular-readback handoff, requiring Q to carry the partial-sum thread into the same dyadic tolerance ledger D used by the tail comparison;*

5. *real-seal boundary, requiring E to be downstream of the displayed Cauchy route and to export no host infinite sum, selected limit, quotient stream equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge datum;*

6. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Cauchy series criterion obligation is stored outside these rows.

Proof. Project the accepted packet $K = (A, S, W, Q, D, M, E, H, C, P, N)$. The carrier definition exposes exactly the finite series row, partial-sum thread, selected windows, regular readback, dyadic ledger, tail threshold, real seal, and structural rows.

For a displayed tolerance request, the partial-sum thread S supplies the only later-window comparisons available to the classifier. Those windows are opened through W, read back through Q, and measured only against the dyadic ledger D.

The shared threshold row M is then replayed. Its claim is finite: the same displayed threshold controls every accepted comparison of later partial sums for the requested tolerance.

Finally E, H, C, P, N seal, transport, replay, source, and name that same route. Since no excluded object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2015.3 (Cauchy series criterion partial-sum thread)**.** *Every completion-facing read from an accepted* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *carrier factors through*

$$A \longmapsto S \longmapsto W \longmapsto Q \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

Thus the route from finite series terms to the completion-facing endpoint is mediated by the displayed partial-sum thread and its $\mathsf{RegSeqRat}^{\uparrow}$ *readback, not by an external sum object or a selected limiting process.*

Proof. Begin with a completion-facing request. The carrier exposes the $\mathsf{Series}^{\uparrow}$ term row A before any partial-sum or endpoint row can be inspected.

The request then passes through the partial-sum thread S and the selected window family W. These rows give the finite initial segments and later comparison windows used by the criterion.

The readback row Q and dyadic ledger D provide the comparison surface. The threshold row M is the only tail-control coordinate, so every later-window comparison uses the displayed threshold attached to the requested tolerance.

The endpoint seal E is downstream of this route, and H, C, P, N preserve, replay, source, and name it. Therefore the completion-facing read cannot bypass the partial-sum thread or replace it by an external infinite-sum datum. □

Theorem 2015.4 (Cauchy series criterion real seal boundary)**.** *For every accepted* $\mathsf{CauchySeriesCriterion}^{\uparrow}$ *carrier* $K = (A, S, W, Q, D, M, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal* E *is readable only after the displayed* $\mathsf{Series}^{\uparrow}$ *term row, finite partial-sum thread, selected* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and shared Cauchy-tail threshold have been displayed. Consequently, every real-seal consumer receives the same finite route and no host completeness theorem, quotient stream equality, selected sum, countable-choice selector, or detached real endpoint.*

Proof. Project the carrier given by Definition 2015.1. Its real endpoint coordinate is the row E, and the definition places E after A, S, W, Q, D, M in the displayed route.

Apply Definition 2015.2. The NameCert obligations require partial-sum exactness, regular-readback handoff, Cauchy-tail stability, and the real-seal boundary before any endpoint-facing consumer may read E.

Apply Definition 2015.3. The only route to E goes through the term row, partial-sum thread, windows, regular readback, dyadic ledger, and threshold row.

The structural rows H, C, P, N transport, replay, source, and name precisely that route. Since the carrier has no slot for host completeness, quotient stream equality, selected sums, choice selectors, or detached real endpoints, the real-seal boundary is exhausted by the displayed finite packet. □

Falsifiable prediction: Every $\mathsf{CauchySeriesCriterion}^{\uparrow}$ completion-facing read factors through a SeriesUp term row, finite partial-sum thread, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, one shared Cauchy-tail threshold, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: series, uniform-cauchy-series, cauchy-majorant-series, absolute-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchySeriesCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite series terms, partial-sum comparisons, selected windows, regular readback, dyadic tolerance accounting, one shared tail threshold, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySeriesCriterion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySeriesCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 2015.1, Definition 2015.2, Definition 2015.3, and Definition 2015.4 seed the finite constructive series criterion route from Series rows through partial sums, RegSeqRat readback, DyadicRatCore tolerance accounting, Cauchy-tail threshold control, and RealUp endpoint sealing.

Formal verification: FormalStatus(NameCert$_{\text{CauchySeriesCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum thread exactness, RegSeqRat and DyadicRatCore handoff, Cauchy-tail threshold stability, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchySeriesCriterionUp.

Constructive story: A CauchySeriesCriterion$^{\uparrow}$ packet is generated from a Series term row, finite partial-sum thread, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, one shared Cauchy-tail threshold, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2016

A Concrete Naming Certificate for $\mathsf{RoundedCauchyFilter}^{\uparrow}$

Following Kelley, Bourbaki, and Bishop–Bridges, the $\mathsf{RoundedCauchyFilter}^{\uparrow}$ packet records the rounded-base presentation of a Cauchy filter as finite NameCert data. A rounded filter base is the constructive surface on which every accepted neighborhood has a smaller displayed neighborhood inside it, so the completion consumer can inspect refinement before reading a limiting real-name seal. The packet sits between the filter-base row of chapter 1601, the Cauchy-filter surface of chapter 333, the filter-refinement route of chapter 1626, and the completion handoff of chapter 1569. It exposes a non-quotient refinement surface for later $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads; it does not import quotient equality of filters, an ultrafilter extension, choice-selected cluster points, trichotomy, host completeness, Lean verification, or a standard bridge.

Definition 2016.1 (Rounded Cauchy filter carrier)**.** *A* $\mathsf{RoundedCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, R, U, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row whose basic neighborhoods are available only through finite displayed base tokens.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row carried by the same base tokens and by the displayed Cauchy comparison requests.*
3. *R is the rounded-base refinement row: every accepted base token exports a strictly smaller displayed token below it, together with the finite inclusion route witnessing the refinement.*
4. *U is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row that may be read only after the base, Cauchy, and rounded-refinement rows have been displayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect the finite base tokens, refinement witnesses, and completion-facing observations.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and to the rational observations exported by the rounded filter route.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radii, rounded inclusions, and Cauchy comparison budgets.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, F, R, U, W, Q, D have been displayed.*

9. *H stores componentwise* hsame *transport for the base row, Cauchy row, rounded refinement row, completion handoff, stream windows, regular readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from filter-base tokens to rounded refinements, from rounded refinements to Cauchy comparisons, from Cauchy comparisons to regular readback and dyadic tolerances, and from that route through the completion handoff to the real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{RoundedCauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RoundedCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, R, U, W, Q, D, E, H, C, P, N$. It has no coordinate for quotient equality of filters, an ultrafilter extension, selected cluster points, trichotomy, host real completeness, Lean verification, or bridge data.

Theorem 2016.2 (Rounded Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{RoundedCauchyFilter}^{\uparrow}$ *carrier $K = (B, F, R, U, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RoundedCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed filter base, Cauchy filter, rounded refinement, completion handoff, stream-window, regular-readback, dyadic-ledger, real-seal, transport, replay, provenance, and local naming rows;*

2. *base alignment, requiring the* $\mathsf{FilterBase}^{\uparrow}$ *row B and the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F to read the same displayed base tokens before any completion-facing observation is accepted;*

3. *roundedness, requiring each accepted base token to pass through the row R and receive a displayed smaller token with its finite inclusion route;*

4. *Cauchy comparison stability, requiring every tolerance request to use the same rounded base tokens and dyadic ledger D before the completion handoff U is read;*

5. *regular readback and real-seal boundary, requiring W, Q, D to mediate the route from rounded filter data to the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

6. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N for exactly the rows displayed above.*

No rounded Cauchy filter obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RoundedCauchyFilterUp.RoundedCauchyFilterTasteGate single carrier alignment`

Proof. Project the accepted packet $K = (B, F, R, U, W, Q, D, E, H, C, P, N)$. Carrier admission exposes exactly the filter-base row, Cauchy-filter row, rounded refinement row, completion handoff, stream-window row, regular readback, dyadic ledger, real seal, and structural rows.

Base alignment is finite. The same displayed base tokens feed B and F, so a consumer cannot read a Cauchy comparison apart from the filter-base surface that names its neighborhoods.

Roundedness is then read through R. For each accepted base token, R supplies the smaller displayed token and the inclusion route that places it below the original token. The Cauchy comparison rows use those rounded tokens and the dyadic ledger D before the handoff U can be consumed.

Finally W, Q, D, E, H, C, P, N provide the observation, readback, tolerance, endpoint, transport, replay, source, and local name of the same route. Since the carrier has no slot for quotient

equality, ultrafilter extension, selected cluster points, trichotomy, host completeness, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 2016.3 (Rounded Cauchy filter regular readback route)**.** *Every completion-facing read from an accepted* $\mathsf{RoundedCauchyFilter}^{\uparrow}$ *carrier factors through*

$$B \longmapsto F \longmapsto R \longmapsto W \longmapsto Q \longmapsto D \longmapsto U \longmapsto E \longmapsto H, C, P, N.$$

Thus the route from filter-base data to a $\mathsf{Real}^{\uparrow}$ *completion seal is mediated by the displayed rounded-refinement row and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, not by quotient equality of filters, a selected representative sequence, an ultrafilter extension, or ambient completeness.*

Proof. Begin with a completion-facing request. The carrier definition exposes the filter-base row B before the Cauchy row F and the rounded-refinement row R can be used.

The request then opens finite windows through W. Those windows inspect the base token, its rounded refinement, and the Cauchy comparison rows attached to that refinement.

Regular readback is forced through Q, and the tolerance accounting is forced through D. Only after this finite observation route has been displayed may the completion handoff U and the real seal E be consumed.

The structural rows H, C, P, N transport, replay, source, and name the same route. No excluded quotient, representative sequence, ultrafilter, or completeness coordinate appears in the carrier, so the displayed factorization is the only completion-facing read. □

Theorem 2016.4 (Rounded Cauchy filter regular-sequence handoff)**.** *Any accepted* $\mathsf{RoundedCauchyFilter}^{\uparrow}$ *packet whose completion-facing route reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *must expose the rounded base row* R*,* $\mathsf{StreamName}^{\uparrow}$ *window row* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff* U*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *in that order. The handoff is a finite readback route over the displayed packet rows; it exports no quotient filter equality, ultrafilter extension, selected representative sequence, selected cluster point, or ambient completeness.*

Proof. Project the carrier in Definition 2016.1. The only completion-facing route that reaches regular sequence data has to pass through $B, F, R, W, Q, D, U, E, H, C, P, N$.

Apply Definition 2016.2. The obligation surface forces roundedness through R, finite observation through W, regular readback through Q, dyadic tolerance accounting through D, completion handoff through U, and real sealing through E.

Now use Definition 2016.3. It orders the same rows before the structural transport, replay, provenance, and local naming coordinates H, C, P, N.

Since the carrier contains no quotient-filter equality, ultrafilter, selected representative sequence, selected cluster point, or ambient completeness row, the regular-sequence handoff exports only the finite displayed route. □

Falsifiable prediction: Every $\mathsf{RoundedCauchyFilter}^{\uparrow}$ completion-facing read factors through a FilterBaseUp row, a CauchyFilterUp row, a rounded-base refinement row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyFilterCompletionUp handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: filterbase, cauchyfilter, filter-refinement, cauchyfiltercompletion, streamname, regseqrat, dyadicratcore, real: $\mathsf{RoundedCauchyFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a filter base, Cauchy filter comparisons, a rounded-base refinement row,

completion handoff, finite windows, regular readback, dyadic tolerance accounting, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RoundedCauchyFilter}^{\uparrow}$ packet is generated from a FilterBaseUp row, a CauchyFilterUp row, a rounded-base refinement row, a CauchyFilterCompletionUp handoff, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2016.1, Definition 2016.2, and Definition 2016.3 seed the finite rounded Cauchy filter route from displayed filter-base tokens through rounded refinements, regular readback, dyadic tolerance accounting, completion handoff, and RealUp endpoint sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedCauchyFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of filters, ultrafilter extensions, selected representative sequences, selected cluster points, trichotomy, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, base alignment, roundedness, Cauchy comparison stability, RegSeqRat and DyadicRatCore handoff, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.RoundedCauchyFilterUp.

Chapter 2017

A Concrete Naming Certificate for $\mathsf{DyadicShrinkSchedule}^{\uparrow}$

Following Bishop and Bridges on epsilon-splitting in constructive analysis, the $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ packet records the finite schedule by which a requested dyadic tolerance is replaced by a displayed halved or refined tolerance before any regular-rational or real-name consumer reads it. It sits over the dyadic arithmetic ledger of chapter 291, the scale discipline of chapter 1768, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter packages precision halving as displayed finite data, not as an appeal to completed reals or to choice of limiting representatives.

Definition 2017.1 (Dyadic shrink schedule carrier)**.** *A* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (k, k', D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *k is the requested dyadic scale row. It names the finite tolerance request before any halving, window, readback, or seal data are consumed.*
2. *k' is the displayed halved or refined scale row. It records the finite witness that the requested scale has been replaced by the precision row used downstream.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to k and k', including the dyadic radius, comparison, containment, and halving rows consumed by the schedule.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window opened at the refined scale and budgeted by the dyadic ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed request, refined scale, dyadic ledger, and finite window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary exposed only after k, k', D, W, R have been read.*
7. *H records componentwise* hsame *transport for the requested scale, refined scale, dyadic ledger, finite window, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the requested scale through the refined scale, dyadic ledger, window, readback, Real seal, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicShrinkSchedule}^{\uparrow}}$ *row.*

The classifier compares accepted schedules only through $k, k', D, W, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or any hidden completeness witness.*

Lean4 ✓ `BEDC.Derived.DyadicShrinkScheduleUp.DyadicShrinkScheduleTasteGate single carrier alignment`

Theorem 2017.2 (Dyadic shrink schedule NameCert obligations)**.** *For every accepted* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ *carrier $S = (k, k', D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicShrinkSchedule}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the requested scale, refined scale, DyadicRatCore ledger, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *halving display, requiring k' to be read as a displayed finite refinement of k before the ledger, window, readback, or seal rows are available;*
3. *dyadic ledger compatibility, requiring D to carry the radius, comparison, containment, and halving checks relating the two scale rows;*
4. *classifier transport, requiring accepted comparisons to preserve $k, k', D, W, R, E, H, C, P, N$ componentwise and to expose no hidden precision coordinate;*
5. *RegSeqRat handoff, requiring R to consume the displayed request, refined scale, dyadic ledger, and StreamName window rather than a detached stream quotient or selected limit;*
6. *Real-seal non-escape, requiring E to remain terminal with respect to k, k', D, W, R and to export no host equality, countable choice, trichotomy, ambient completeness theorem, or selected limiting object;*
7. *provenance and local naming, requiring P and N to package exactly the displayed schedule rows and no additional completion datum.*

No dyadic shrink-schedule obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2017.1. The accepted packet consists exactly of the requested scale k, the refined scale k', the dyadic ledger D, the finite window W, the regular readback R, the Real seal E, and the structural rows H, C, P, N.

The halving and ledger clauses are also projections. The only row that displays the refined scale is k', and the only row that records dyadic radius, comparison, containment, and halving checks is D.

Classifier transport is componentwise through H and replayed through C. These rows preserve the route from k to k', from k' to D, from D to W, from W to R, and from R to E without adding a second precision source.

The handoff and seal clauses follow from the placement of R and E. The readback row is downstream of the displayed halving schedule, while the Real seal is terminal after the same finite route.

The excluded objects are absent from the carrier and are not produced by transport, replay, provenance, or local naming. Therefore no local NameCert obligation can be stored outside the displayed rows. □

Theorem 2017.3 (Dyadic shrink schedule halving closure)**.** *Let $S = (k, k', D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ *carrier. Every finite consumer of the schedule reads the*

precision-halving route

$$k \longmapsto k' \longmapsto D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

as a closed displayed route. In particular, halving a dyadic tolerance inside the packet produces another named finite precision row before any StreamName, RegSeqRat, or RealUp consumer can inspect the schedule.

Proof. Project the accepted carrier. The requested precision row is k, and the only displayed refined precision row is k'.

The dyadic ledger D is positioned after both scale rows and before the finite window. Thus every downstream read of the window, readback, or seal has already passed through the displayed refinement.

The transport and replay rows preserve this ordering componentwise. Since no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate appears in the packet, the refined precision row is closed as finite schedule data. □

Theorem 2017.4 (Dyadic shrink schedule RegSeqRat handoff)**.** *For every accepted* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ *carrier* $S = (k, k', D, W, R, E, H, C, P, N)$*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer reaches the readback row* R *only through the finite route from the requested scale to the refined scale, the DyadicRatCore ledger, and the StreamName window. No RegSeqRat handoff from* S *can bypass the displayed halving schedule by quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The only readback coordinate in the carrier is R. By Definition 2017.1, it is obtained from k, k', D, W rather than from an independent stream or limit object.

The rows H and C transport and replay the same displayed route, while P and N only package it as provenance and local naming data. None of these rows introduces a second source for regular rational readback.

Consequently a RegSeqRat consumer must pass through the requested scale, refined scale, dyadic ledger, and finite window before reaching R, with all excluded completion and choice principles remaining outside the schedule. □

Theorem 2017.5 (Dyadic shrink schedule Real seal boundary)**.** *For every accepted* $\mathsf{DyadicShrinkSchedule}^{\uparrow}$ *carrier* $S = (k, k', D, W, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is terminal over the displayed finite schedule. A Real consumer may read the seal only after the request, refinement, dyadic ledger, StreamName window, and RegSeqRat readback have been displayed, and the seal exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and isolate the seal coordinate E. It appears after k, k', D, W, R in the displayed schedule and has no independent source row.

The transport, replay, provenance, and local naming rows preserve the same order. They can certify the finite schedule but cannot turn the terminal seal into a host real equality, a selected limit, or a completeness witness.

Thus the $\mathsf{Real}^{\uparrow}$ boundary exposed by S is exactly the displayed seal after regular readback, with quotient, choice, trichotomy, host equality, and ambient completeness data excluded. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicShrinkSchedule}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicShrinkSchedule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2017.1, Definition 2017.2, Definition 2017.3, Definition 2017.4, and Definition 2017.5 seed the finite dyadic precision-halving schedule from a requested scale through refined scale, dyadic ledger, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicShrinkSchedule}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DyadicShrinkScheduleUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, halving display, dyadic ledger compatibility, classifier transport, RegSeqRat handoff, Real-seal non-escape, provenance, and local naming for BEDC.Derived.DyadicShrinkScheduleUp.

Constructive story: A DyadicShrinkSchedule$^{\uparrow}$ packet is generated from a requested dyadic scale row, a displayed refined scale row, a DyadicRatCore tolerance ledger, a StreamName finite window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2018

A Concrete Naming Certificate for $\mathsf{DyadicBinarySearch}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicBinarySearch}^{\uparrow}$ packet records finite bisection search over located dyadic intervals before any Real-and-Completion consumer receives a bracket. It sits on the dyadic arithmetic ledger of chapter 291, the finite bisection route of chapter 1711, the located interval surface of chapter 394, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal boundary of chapter 38. The packet names a bounded controller for interval refinement; it does not decide global trichotomy, select an infinite limit, identify quotient real names, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2018.1 (Dyadic binary search carrier)**.** *A* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, T, K, M, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial* $\mathsf{LocatedInterval}^{\uparrow}$ *row, containing the left endpoint, right endpoint, dyadic radius, and locatedness data from which the search begins.*
2. *T is the finite test row. It records the decidable inspected predicate or sign comparison available at each displayed midpoint, but only on the finite rows opened by the packet.*
3. *K is the unary depth row, bounding the number of bisection decisions made by the controller.*
4. *M is the midpoint row, computed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic from the current retained interval at every depth below K.*
5. *L is the retained-half ledger, recording for each stage whether the left or right dyadic half is kept and which located interval row results.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the inspected endpoint, midpoint, and retained-half data are exposed to downstream consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite stream windows and dyadic midpoint ledger.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *bracket seal handed off only after the retained interval, stream windows, and regular rational readback have been displayed.*
9. *H records componentwise* hsame *transport for the interval, test, depth, midpoint, retained-half, stream-window, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of every consumer request through*

$$I, T, K \longmapsto M, L \longmapsto W \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{DyadicBinarySearch}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBinarySearch}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for real trichotomy, quotient real equality, countable choice, a selected infinite refinement, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicBinarySearchUp.DyadicBinarySearchTasteGate single carrier alignment`

Theorem 2018.2 (Dyadic binary search NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *carrier $B = (I, T, K, M, L, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicBinarySearch}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed twelve-row packet;*
2. *initial located-interval exposure for I and finite test exposure for T;*
3. *unary depth coverage for K and midpoint arithmetic exactness for M through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *retained-half ledger exactness for L through the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *route;*
5. *stream-window coverage for W, regular rational readback for R, and* $\mathsf{Real}^{\uparrow}$ *bracket non-escape for E;*
6. *componentwise transport, continuation replay, provenance, and local naming for exactly the rows recorded in H, C, P, N.*

No dyadic binary-search NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier $B = (I, T, K, M, L, W, R, E, H, C, P, N)$. The projection exposes the initial located interval, finite test row, unary depth, midpoint ledger, retained-half ledger, stream windows, regular readback, Real bracket seal, and the structural rows.

The numerical part is finite. The row K bounds the number of midpoint computations, while M records each midpoint through $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic and L records the retained half through the $\mathsf{LocatedIntervalBisection}^{\uparrow}$ route.

The observation part is also finite. The test row T can inspect only the displayed midpoint and endpoint data, and the window row W is the only stream surface available to the readback row R.

Finally E is downstream of R. The rows H, C, P, N preserve, replay, source, and name the same route, so the listed rows exhaust the local NameCert obligations. □

Theorem 2018.3 (Dyadic binary search bisection step)**.** *Let $B = (I, T, K, M, L, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *carrier. For every displayed stage strictly below the unary depth K, the next retained interval is obtained by the finite route*

$$I, T, M \longmapsto L.$$

The retained interval is one of the two dyadic halves certified by the $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *row, and its midpoint computation is the one displayed by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*

Proof. Proceed by finite induction on the displayed prefix of K. In the base stage, the current interval is the initial row I, and the midpoint row M computes its dyadic midpoint from the endpoints already displayed in I.

For the step stage, assume the retained interval has been obtained from the previous prefix through L. The next midpoint is again read from M, and the finite test row T supplies the only branch datum permitted at that midpoint.

Apply the $\mathsf{LocatedIntervalBisection}^{\uparrow}$ retained-half row recorded in L. It stores exactly one of the two dyadic halves, with the corresponding located interval data, and componentwise transport in H preserves that row under the displayed classifier.

The induction therefore produces no extra analytic coordinate. Each stage is obtained by repeating the same finite midpoint, test, and retained-half route below the bound K. □

Theorem 2018.4 (Dyadic binary search Real bracket handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{DyadicBinarySearch}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{LocatedIntervalBisection}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer must read $I, T, K, M, L, W, R, H, C, P, N$ *before the bracket seal* E *is exposed.*

Proof. Begin with a consumer request for the bracket seal. Carrier projection shows that the only terminal Real-facing coordinate is E, and the continuation row C routes access to E through the displayed packet.

The route first reads the initial located interval I, the finite test row T, and the unary depth K. The midpoint ledger M and retained-half ledger L then provide the bounded bisection data through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{LocatedIntervalBisection}^{\uparrow}$.

Next the request passes through the stream windows W and the regular rational readback R. These rows are the only observation and readback surfaces connected to the terminal bracket.

The structural rows H, P, N bind the same route under transport, provenance, and local naming. Since the carrier contains no coordinate for trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the handoff exports none of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBinarySearch}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBinarySearch}^{\uparrow}$ packet is generated from an initial LocatedInterval row, a finite test row, a unary depth row, a DyadicRatCore midpoint ledger, a LocatedIntervalBisection retained-half ledger, StreamName windows, a RegSeqRat readback, a Real bracket seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBinarySearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2018.1, Definition 2018.2, Definition 2018.3, and Definition 2018.4 seed the finite dyadic bisection-search controller over located intervals, dyadic midpoint arithmetic, retained-half ledgers, stream windows, regular rational readback, and Real bracket seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBinarySearch}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DyadicBinarySearchUp.DyadicBinarySearchTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification of the NameCert obligation, bisection-step, and Real-bracket handoff theorems.

Upgrade path: Obligation closure requires checked finite NameCert obligation, bisection-step, and Real-bracket handoff theorems for BEDC.Derived.DyadicBinarySearchUp.

Chapter 2019

A Concrete Naming Certificate for $\mathsf{DyadicMetricTriangle}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicMetricTriangle}^{\uparrow}$ packet names the finite three-point dyadic triangle ledger that must be visible before regular rational names, real-name seals, or metric-space consumers may use a triangle estimate. It reads the three displayed dyadic endpoints of chapter 291, the nonnegative endpoint discipline of chapter 293, and the distance-ledger packet of chapter 2001; it hands the resulting bound to chapter 301 and then to chapter 38. The chapter isolates the triangle inequality as a finite comparison route, not as a law imported from a host metric space.

Definition 2019.1 (Dyadic metric triangle carrier)**.** *A* $\mathsf{DyadicMetricTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (x, y, z, dxy, dyz, dxz, a, b, t, h, c, p, n)$$

with the following displayed rows:

1. x, y, z *are the three accepted* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows, including their visible mantissa, denominator,* $\mathsf{Int}^{\uparrow}$*, and* $\mathsf{Rat}^{\uparrow}$ *provenance.*

2. dxy, dyz, dxz *are the three dyadic distance ledgers for the pairs* (x, y)*,* (y, z)*, and* (x, z)*. Each ledger records its common-denominator subtraction window, dyadic radius row, and finite comparison data through the* $\mathsf{DyadicMetric}^{\uparrow}$ *packet.*

3. a *records the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint rows attached to the three distance ledgers, so every nonnegative distance endpoint is read through the absolute-value packet rather than through a host absolute-value function.*

4. b *is the displayed comparison row*

$$dxz \leq dxy + dyz.$$

   It records the common-denominator addition window, the dyadic upper-bound ledger, and the finite order comparison witnessing the triangle bound.

5. t *is the downstream handoff row. It exposes the finite route from the three endpoints through the three distance ledgers, the absolute-value endpoints, and the comparison row to* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *consumers.*

6. h *records componentwise* hsame *transport for the endpoint rows, distance ledgers, absolute-value rows, comparison row, consumer handoff, replay, provenance, and local naming.*

7. *c records displayed* Cont *replay from the endpoint rows through distance, absolute value, comparison, consumer handoff, transport, provenance, and local naming.*
8. *p is the* Pkg *provenance over* $\mathsf{DyadicMetricTriangle}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMetric}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *n is the local* $\mathsf{NameCert}_{\mathsf{DyadicMetricTriangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $x, y, z, dxy, dyz, dxz, a, b, t, h, c, p, n$. It has no coordinate for host metric spaces, quotient rationals, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicMetricTriangleUp.DyadicMetricTriangleTasteGate single carrier alignment`

Theorem 2019.2 (Dyadic metric triangle NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMetricTriangle}^{\uparrow}$ *carrier $T = (x, y, z, dxy, dyz, dxz, a, b, t, h, c, p, n)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicMetricTriangle}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the displayed endpoints x, y, z and the three distance ledgers dxy, dyz, dxz;*
2. *absolute-value exposure through a for each nonnegative dyadic distance endpoint;*
3. *triangle comparison through the row b witnessing $dxz \leq dxy + dyz$;*
4. *consumer handoff through t to* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Met}^{\uparrow}$*;*
5. *componentwise structural replay through h, c, p, n.*

No triangle-bound obligation is stored outside those finite rows.

Lean4 ✓ `BEDC.Derived.DyadicMetricTriangleUp.DyadicMetricTriangleCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 2019.1. The visible packet lists three $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows and three $\mathsf{DyadicMetric}^{\uparrow}$ distance ledgers before any comparison row is available.

The nonnegative endpoint data is not a host operation. It is routed through the $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ rows in a, one for each displayed distance ledger.

The triangle row is the only comparison row. It displays the dyadic addition ledger and the finite order comparison for $dxz \leq dxy + dyz$, so the packet has no hidden metric-law coordinate.

The consumer and structural rows are terminal. Rows t, h, c, p, n hand off, transport, replay, package, and name the displayed triangle ledger without adding quotient rationals, trichotomy, selected limits, ambient completeness, Lean verification, or bridge checking. □

Theorem 2019.3 (Dyadic metric triangle RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicMetricTriangle}^{\uparrow}$ *triangle bound must pass through the finite route*

$$x, y, z \longmapsto dxy, dyz, dxz \longmapsto a \longmapsto b \longmapsto t \longmapsto h, c, p, n$$

before the triangle estimate is available as a regular-rational tolerance bound, a real-name distance surface, or a metric witness. Consequently no such consumer can obtain the triangle bound through a host metric space, quotient rational equality, trichotomy, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier. The only consumer-facing coordinate is t, and t is downstream of the endpoint rows, the three dyadic distance ledgers, the absolute-value endpoint rows, and the comparison row.

For a $\mathsf{RegSeqRat}^{\uparrow}$ consumer, the triangle estimate is read as a finite dyadic tolerance comparison. The comparison cannot be inspected until the three distance ledgers and the absolute-value endpoint route have displayed the row $dxz \leq dxy + dyz$.

For a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Met}^{\uparrow}$ consumer, the same route is terminal. The real-name seal and the metric witness receive only the surface exposed by t, while h, c, p, n transport, replay, package, and name that surface without introducing a second metric law.

The excluded objects are absent from the carrier. Thus every accepted downstream read factors through the displayed triangle ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMetricTriangle}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMetricTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2019.2 and Definition 2019.3 seed the finite dyadic triangle route from three $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints through three distance ledgers, absolute-value endpoints, the displayed comparison row, consumer handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMetricTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host metric spaces, quotient rationals, trichotomy, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, three-distance ledger exposure, absolute-value endpoint exposure, triangle comparison, consumer handoff, provenance, transport, replay, and local naming for BEDC.Derived.DyadicMetricTriangleUp.

Constructive story: A $\mathsf{DyadicMetricTriangle}^{\uparrow}$ packet is generated from three $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, three $\mathsf{DyadicMetric}^{\uparrow}$ distance ledgers, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ endpoint rows, one dyadic triangle comparison row, a $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ consumer handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2020

Dyadic Fast Cauchy NameCert Construction

Following Bishop and Bridges, the $\mathsf{DyadicFastCauchy}^{\uparrow}$ packet records the finite fast-Cauchy interface by which dyadic tolerance schedules become regular rational data before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the dyadic arithmetic ledger of chapter 291, the finite stream window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name boundary of chapter 38, and the displayed modulus discipline of chapter 1781. The chapter names the L10 Real and Completion handoff as finite NameCert data, not as a host completion theorem.

Definition 2020.1 (Dyadic fast Cauchy carrier)**.** *A* $\mathsf{DyadicFastCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack comparisons, addition rows, and containment checks consumed by the fast Cauchy route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected at the visible dyadic requests.*
3. *M is the* $\mathsf{FastCauchyModulus}^{\uparrow}$ *row, recording the displayed rule that a request at level k supplies windows whose mutual distance is bounded by the dyadic tolerance named by D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D, S, M.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the dyadic ledger, stream windows, modulus row, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, modulus row, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from dyadic requests through D, S, M, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicFastCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicFastCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, M, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicFastCauchyUp.TasteGate.DyadicFastCauchyTasteGate single carrier alignment`

Theorem 2020.2 (Dyadic fast Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{DyadicFastCauchy}^{\uparrow}$ *carrier* $F = (D, S, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicFastCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic ledger, finite stream windows, displayed fast-Cauchy modulus, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows;*
2. *dyadic tolerance exposure, requiring every requested bound to be read from* D *before a stream window or readback row is consumed;*
3. *finite-window coverage, requiring* S *to contain precisely the displayed windows selected for the visible dyadic requests;*
4. *modulus stability, requiring* M *to bind each visible request to the pairwise window comparison budget named in* D*;*
5. *RegSeqRat handoff, requiring* R *to be downstream of the same* D, S, M *route;*
6. *Real-seal boundary, requiring* E *to remain terminal after* R *and to export no quotient stream, selected host limit, countable-choice selector, trichotomy row, host real equality, ambient completeness theorem, Lean evidence, or checked bridge;*
7. *provenance and local naming, requiring* P *and* N *to package exactly the displayed fast-Cauchy route.*

No DyadicFastCauchy NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 2020.1. The analytic coordinates are exactly the dyadic ledger D, finite windows S, displayed modulus row M, regular readback R, and terminal Real seal E.

The tolerance and window clauses are finite projections. A requested bound enters through D, the visible windows are stored in S, and the modulus row M is the only row that relates the request to the pairwise window comparison budget.

The handoff clauses follow the displayed order of the packet. The readback row R is downstream of D, S, M, while E appears only after R and therefore cannot be read as an independent host limit or ambient completeness witness.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded objects appears as a carrier coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2020.3 (Dyadic fast Cauchy RegSeqRat handoff)**.** *For every accepted* $\mathsf{DyadicFastCauchy}^{\uparrow}$ *carrier* $F = (D, S, M, R, E, H, C, P, N)$*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer reaches the readback row* R *only through the finite route*

$$D \longmapsto S \longmapsto M \longmapsto R.$$

Thus the handoff from dyadic fast-Cauchy data to regular rational data cannot bypass the displayed tolerance ledger, the selected stream windows, or the fast-Cauchy modulus row by quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the accepted packet. The only regular rational coordinate is R, and it is placed after the dyadic ledger D, finite windows S, and displayed modulus row M.

The transport and replay rows preserve this ordering componentwise. Provenance and local naming then package the same route without adding a second source for readback data.

Consequently any RegSeqRat consumer of the packet must pass through D, S, M before reading R. The excluded quotient, selector, choice, order, equality, and completeness objects are absent from the carrier. □

Theorem 2020.4 (Dyadic fast Cauchy Real seal boundary)**.** *For every accepted* $\mathsf{DyadicFastCauchy}^{\uparrow}$ *carrier* $F = (D, S, M, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is terminal over the displayed dyadic fast-Cauchy route. A Real consumer may read the seal only after the DyadicRatCore ledger, StreamName windows, FastCauchyModulus row, and RegSeqRat readback have been displayed, and the seal exports no quotient stream equality, selected host limit, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the terminal coordinate E from the accepted carrier. It is downstream of D, S, M, R and has no independent host-limit coordinate.

The componentwise transport row H and replay row C preserve the route from the dyadic ledger through stream windows, modulus, readback, and seal. The provenance and local naming rows only record that route as finite certificate data.

Therefore a downstream Real consumer cannot use E as a hidden quotient, selected limit, choice principle, trichotomy claim, host equality, ambient completeness result, Lean theorem, or checked bridge. The seal is exactly the terminal boundary of the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicFastCauchy}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicFastCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2020.1, Definition 2020.2, Definition 2020.3, and Definition 2020.4 seed the finite dyadic fast-Cauchy route from displayed tolerance ledgers and stream windows to RegSeqRat readback and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicFastCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window coverage, modulus stability, RegSeqRat handoff, Real-seal boundary, provenance, and local naming for BEDC.Derived.DyadicFastCauchyUp.

Constructive story: A $\mathsf{DyadicFastCauchy}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, finite StreamName windows, a FastCauchyModulus row, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2021

A Concrete Naming Certificate for $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ packet records the finite dyadic-net total-boundedness certificate for closed $\mathsf{Real}^{\uparrow}$ intervals before compactness-facing consumers read the interval as a completed object. It sits after the closed interval mesh of chapter 1250, the dyadic refinement basis of chapter 1916, the total-bounded interface of chapter 127, the finite epsilon-net ledger of chapter 1767, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet isolates the total-boundedness row used by closed-interval compactness; it is not open-cover compactness, not a selected subcover, not quotient real equality, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2021.1 (Closed interval total-bounded carrier)**.** *A* $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, F, U, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the lower and upper closed* $\mathsf{Real}^{\uparrow}$ *endpoint rows, exposed only through their finite stream windows and regular rational readbacks.*
2. *M is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *mesh row carrying the unary dyadic scale, displayed cells, center rows, and interval coverage ledger for the closed interval determined by A, B.*
3. *D is the* $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *refinement row, recording how each displayed mesh cell is refined by dyadic basis cells at the requested positive tolerance.*
4. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *ledger obtained from the displayed mesh centers, coverage witnesses, and dyadic tolerance comparisons.*
5. *U is the* $\mathsf{TotallyBounded}^{\uparrow}$ *handoff row, read only after M, D, F have displayed the finite center bundle and coverage route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family for endpoint, center, dyadic-cell, and coverage observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same endpoint, center, and coverage windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S and R have exposed the relevant finite reads.*

9. *H records componentwise* hsame *transport for endpoints, interval nets, dyadic refinements, finite epsilon nets, the total-bounded handoff, stream windows, readbacks, the Real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay of every consumer request through*

$$A, B \longmapsto M \longmapsto D \longmapsto F \longmapsto U \longmapsto S \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicIntervalBasis}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalTotalBounded}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, selected finite subcovers, quotient real equality, trichotomy, countable choice, a hidden center selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalTotalBoundedUp.ClosedIntervalTotalBoundedTasteGate single carrier alignment`

Theorem 2021.2 (Closed interval total-bounded NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ *carrier* $T = (A, B, M, D, F, U, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalTotalBo}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two endpoint rows, the closed-bounded interval net, dyadic basis refinement, finite epsilon-net ledger, total-bounded handoff, stream windows, regular readback, Real seal, and structural rows H, C, P, N;*

2. *endpoint visibility, requiring A and B to be read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows before any interval-total-bounded route is consumed;*

3. *mesh coverage, requiring M to expose the dyadic scale, cell ledger, center rows, and closed-interval coverage witnesses supplied by the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *packet;*

4. *dyadic refinement exactness, requiring D to refine only the displayed mesh cells by the* $\mathsf{DyadicIntervalBasis}^{\uparrow}$ *route at the requested tolerance;*

5. *finite epsilon-net formation, requiring F to list exactly the displayed centers and coverage witnesses used by the tolerance request;*

6. *total-bounded handoff, requiring U to be downstream of M, D, F and to export no additional center selector or compactness theorem;*

7. *transport, replay, provenance, and local naming for exactly the displayed rows recorded in H, C, P, N.*

No closed-interval total-bounded NameCert obligation is stored outside these rows.

Proof. Project the accepted packet $T = (A, B, M, D, F, U, S, R, E, H, C, P, N)$. Carrier admission exposes exactly the endpoint rows, the interval mesh, the dyadic refinement row, the finite epsilon-net ledger, the total-bounded handoff, the stream windows, the regular readback, the Real seal, and the four structural rows.

The endpoint and mesh obligations are finite. The endpoints are visible only through S and R, while M supplies the displayed dyadic scale, center rows, and coverage witnesses from the closed-bounded interval-net packet.

The tolerance obligation is the route M, D, F, U. The dyadic basis row D refines the displayed mesh cells, the finite epsilon-net ledger F records the centers and coverage witnesses for the requested tolerance, and the total-bounded handoff U is read only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no open-cover, selected-subcover, quotient-equality, trichotomy, choice, hidden-selector, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2021.3 (Closed interval total-bounded dyadic-net handoff)**.** *For every accepted* $\mathsf{ClosedIntervalTotalBound}$ *carrier* $T = (A, B, M, D, F, U, S, R, E, H, C, P, N)$ *and every displayed positive tolerance request, the exported* $\mathsf{TotallyBounded}^{\uparrow}$ *row factors through the finite route*

$$\mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longrightarrow \mathsf{DyadicIntervalBasis}^{\uparrow} \longrightarrow \mathsf{FiniteEpsilonNet}^{\uparrow} \longrightarrow \mathsf{TotallyBounded}^{\uparrow}.$$

Thus the total-boundedness witness is the finite dyadic mesh and its displayed center bundle, not a compactness theorem or a selected subcover of an open family.

Proof. Begin with a tolerance request for the total-bounded handoff U. The carrier places U downstream of the closed-bounded interval-net row M, so the request first reads the displayed mesh cells, centers, and coverage witnesses.

The dyadic basis row D refines those mesh cells at the requested tolerance. This refinement is tied to the cells in M and does not introduce a separate family of centers.

The finite epsilon-net ledger F then packages the displayed centers and coverage witnesses as the finite net consumed by U. The rows H, C, P, N preserve and replay the same sequence of reads.

Therefore every total-bounded export factors through M, D, F, U. Because the carrier has no open-cover coordinate and no selected-subcover coordinate, the handoff is finite dyadic-net data rather than a compactness selection principle. □

Theorem 2021.4 (Closed interval total-bounded Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*, compact-net, or Heine–Borel consumer of an accepted* $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ *carrier reaches endpoint and coverage data only through*

$$A, B, M, D, F, U, S, R, E, H, C, P, N.$$

The terminal $\mathsf{Real}^{\uparrow}$ *seal is downstream of the displayed stream windows and regular rational readbacks, and the consumer receives no quotient real equality, host endpoint choice, selected subcover, countable choice, or ambient completeness row.*

Proof. Start with a consumer request for endpoint, coverage, or total-boundedness data. The only endpoint rows in the packet are A and B, and the only Real-facing terminal row is E.

The route to E passes through the finite windows S and regular readback R. Before those rows can serve the interval-total-bounded request, the mesh row M, dyadic refinement row D, finite epsilon-net row F, and total-bounded handoff U have already fixed the displayed center and coverage data.

The structural rows H, C, P, N then transport, replay, source, and name the same route for compact-net and Heine–Borel consumers. Since no quotient equality, host endpoint selector, selected subcover, countable-choice schedule, or ambient-completeness row appears in the carrier, none can escape through the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalTotalBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ packet is generated from closed RealUp endpoint rows, a ClosedBoundedIntervalNetUp dyadic mesh, a DyadicIntervalBasisUp refinement row, a FiniteEpsilonNetUp center ledger, a TotallyBoundedUp handoff, StreamNameUp windows, RegSeqRatUp readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalTotalBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2021.1, Definition 2021.2, Definition 2021.3, and Definition 2021.4 seed the finite dyadic-net total-boundedness certificate for closed RealUp intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalTotalBounded}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ClosedIntervalTotalBoundedUp.ClosedIntervalTotalBoundedTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification of dyadic-net coverage, finite epsilon-net formation, total-bounded handoff, Real-seal nonescape, and structural transport.

Upgrade path: Obligation closure requires checked dyadic-net coverage, finite epsilon-net formation, total-bounded handoff, Real-seal nonescape, and structural transport for BEDC.Derived.ClosedIntervalTotalBoundedUp.

Chapter 2022

Extension by Continuity NameCert Construction

Bishop and Bridges/Kelley completion language treats extension by continuity as a dense-source construction: a Cauchy-controlled map is first read on explicit dense Cauchy data, then carried through the completion universal property, and only then exposed as a continuous real-facing extension. The $\mathsf{ExtensionByContinuity}^{\uparrow}$ packet records that route as finite NameCert data between the extension surface of chapter 397, the operator boundary of **??**, the Cauchy-continuous map surface of chapter 1401, the finite source windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It gives later Real and Completion consumers a visible public object for the dense-to-continuous extension route without asserting quotient completions, selected limits, countable choice, host function equality, or ambient real completeness.

Definition 2022.1 (Extension by continuity carrier)**.** *An* $\mathsf{ExtensionByContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, F, U, L, W, R, T, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense-source row, naming the regular Cauchy data on which the source map is initially inspected.*
2. *F is the Cauchy-continuous source-map row, read through the surface of chapter 1401.*
3. *U is the completion-universal extension row, aligned with chapter 397 and the operator handoff of* **??**.
4. *L is the uniqueness ledger comparing two extension rows when their dense-source, Cauchy-continuous map, and completion-universal extension coordinates agree.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for dense-source observations and extension replay.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational source data carried by D and W.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling source closeness, map continuity requests, and completion-facing precision.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which the completed extension is read only after D, F, U, W, R, T have been replayed.*

9. *H records componentwise* hsame *transport for D, F, U, L, W, R, T, A and for the structural rows of the packet.*

10. *C records displayed* Cont *replay of the route*

$$D, W, R, T \longmapsto F \longmapsto U \longmapsto A.$$

11. *P is the* Pkg *provenance over* $\mathsf{ExtensionByContinuity}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyExtensionOperator}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ExtensionByContinuity}^{\uparrow}}$ *row.*

The carrier contains no quotient completion, no selected limit, no countable-choice schedule, no host function equality row, no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, no Lean verification coordinate, and no bridge-checking coordinate.*

Lean4 ✓ `BEDC.Derived.ExtensionByContinuityUp.TasteGate.ExtensionByContinuityTasteGate single carrier alignment`

Definition 2022.2 (Extension by continuity classifier)**.** *Two accepted* $\mathsf{ExtensionByContinuity}^{\uparrow}$ *carriers*

$$E = (D, F, U, L, W, R, T, A, H, C, P, N)$$

and

$$E' = (D', F', U', L', W', R', T', A', H', C', P', N')$$

classify together exactly when the classifier compares the displayed coordinates $D, F, U, L, W, R, T, A, H, C, P, N$ componentwise with $D', F', U', L', W', R', T', A', H', C', P', N'$, and when the displayed Cont *replay route has the same dense-source, map, extension, and real-seal order on both packets. The classifier has no field for quotient representatives, selected limits, countable choice, host function equality, ambient real completeness, Lean verification, or bridge checking.*

Theorem 2022.3 (Extension by continuity NameCert obligations)**.** *For the carrier of Definition 2022.1, the local* $\mathsf{NameCert}_{\mathsf{ExtensionByContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier display, admitting precisely the packet $E = (D, F, U, L, W, R, T, A, H, C, P, N)$;*

2. *classifier stability, requiring the classifier of Definition 2022.2 to compare only the displayed coordinates and their replay order;*

3. *extension-existence handoff, requiring the Cauchy-continuous source-map row F and the completion-universal row U to be read after the dense-source window, readback, and tolerance rows D, W, R, T;*

4. *uniqueness, requiring the ledger L to compare candidate extension rows only over the same dense-source, Cauchy-continuous map, and completion-universal coordinates;*

5. $\mathsf{Real}^{\uparrow}$*-seal nonescape, requiring every real-facing read to pass through A only after D, W, R, T, F, U have been replayed;*

6. *provenance and name rows, requiring H, C, P, N to expose only componentwise transport, displayed continuation replay, package provenance, and the local NameCert row.*

No obligation for the local certificate is stored outside those rows.

Proof. Carrier display is projection from Definition 2022.1. The accepted packet lists the dense-source row, Cauchy-continuous map row, completion-universal extension row, uniqueness ledger, finite window, regular readback, dyadic tolerance, real seal, and structural rows.

Classifier stability follows from Definition 2022.2. The comparison is componentwise on the displayed coordinates, and the only continuation data compare the same replay order on the two packets.

For extension existence, first read D, W, R, T. These rows expose the dense Cauchy source through finite windows, regular rational readback, and dyadic tolerance data. The source-map row F is available only after those rows are present, and the completion-universal row U consumes that map row as its extension handoff.

The uniqueness obligation is the projection to L. Since the ledger compares extension rows over the same D, F, U, W, R, T coordinates, it never appeals to equality of host functions or to an external completed object.

Finally inspect A, H, C, P, N. The seal row A is downstream of the displayed continuation route, while H, C, P, N record only transport, replay, provenance, and local naming. The excluded principles are absent from the carrier inventory, so the six listed obligations exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.ContinuityCarrierNameCertUp.ContinuityCarrierNameCert semantic name certificate`

Theorem 2022.4 (Extension by continuity public export)**.** *Every accepted* $\mathsf{ExtensionByContinuity}^{\uparrow}$ *packet exports a public completion-extension surface whose consumers factor through the single route*

$$D, W, R, T \longmapsto F \longmapsto U \longmapsto A$$

together with the uniqueness ledger L, componentwise hsame *transport H, displayed* Cont *replay C, package provenance P, and local naming row N. The export is visible to* $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyExtensionOperator}^{\uparrow}$, $\mathsf{CauchyContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *consumers only through those coordinates, and it exports no quotient completion, selected limit, countable-choice schedule, host function equality, ambient real completeness, Lean verification, or bridge-checking row.*

Proof. Begin with the accepted carrier. By Definition 2022.1, every public consumer sees the ordered coordinates $D, F, U, L, W, R, T, A, H, C, P, N$ and no other carrier field.

Use the obligation theorem next. The extension-existence handoff forces the dense-source rows D, W, R, T to be read before F, and forces the completion-universal extension row U to consume that Cauchy-continuous source-map row.

The real-facing boundary is then the seal coordinate A. The displayed continuation route records A as downstream of D, W, R, T, F, U, so a $\mathsf{Real}^{\uparrow}$ consumer cannot bypass the dense-source and completion-universal handoff.

The uniqueness and structural surfaces are exactly L, H, C, P, N. They compare candidate extensions, transport displayed rows, replay the route, record provenance, and name the local certificate. Since the carrier and obligations contain no coordinate for the excluded principles, the public export cannot expose them. □

Closure status for $\mathsf{NameCert}_{\mathsf{ExtensionByContinuity}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ExtensionByContinuity}^{\uparrow}$ packet is generated from a dense-source row, a Cauchy-continuous source-map row, a completion-universal extension row, a uniqueness ledger, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ExtensionByContinuity}\uparrow}$, seed)

Scope closed: Definition 2022.1, Definition 2022.2, Definition 2022.3, and Definition 2022.4 seed the finite dense-source-to-continuous-extension route over the L10 Real and Completion spine.

Formal verification: FormalStatus(NameCert$_{\text{ExtensionByContinuity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier display, classifier stability, extension-existence handoff, uniqueness, Real seal nonescape, provenance rows, and local NameCert rows for BEDC.Derived.ExtensionByContinuityUp.

Chapter 2023

A Concrete Naming Certificate for $\mathsf{SeriesLimitComparison}^{\uparrow}$

From Bishop and Bridges constructive analysis, the series limit-comparison test is a finite comparison route between two positive series, not an appeal to a host real limit of quotients. The $\mathsf{SeriesLimitComparison}^{\uparrow}$ packet sits after the finite tail criterion of chapter 2014 and the regular-Cauchy comparison surface of chapter 411. It reads source and comparator series only through $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, and the $\mathsf{DyadicRatCore}^{\uparrow}$ budget of chapter 291, then seals the accepted endpoint through the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38. The packet records a constructive comparison certificate for eventual positive ratio control; it does not import host limits, quotient series equality, countable choice, trichotomy, or ambient real completeness.

Definition 2023.1 (Series limit comparison carrier)**.** *A* $\mathsf{SeriesLimitComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, S, T, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source-series row, recording finite term and partial-sum observations for the series whose Cauchy behavior is being certified.*
2. *B is the comparator-series row, recording the matched finite term and partial-sum observations used as the comparison source.*
3. *S is the source partial-sum thread exposed to the* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *route.*
4. *T is the comparator partial-sum thread exposed to the same criterion surface.*
5. *R is the eventual positive comparison and ratio ledger: it records the displayed threshold after which comparator terms are positive and the source terms are bounded by a carried dyadic multiple of the comparator terms.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, including the ratio slack, product budget, tail tolerance, and comparison radii consumed by R, S, T.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opening exactly the inspected source terms, comparator terms, partial sums, and endpoint-facing approximations.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source and comparator windows and to the dyadic requests in D.*

9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal available only after the source row, comparator row, ratio ledger, dyadic budget, stream windows, regular readback, and Cauchy handoff rows have been displayed.*

10. *H stores componentwise* hsame *transport for the source series, comparator series, two partial-sum threads, ratio ledger, dyadic budget, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*

11. *C stores displayed* Cont *replay from source and comparator series rows to their partial-sum threads, from those threads to the ratio ledger, from the ratio ledger to the dyadic budget and regular readback, and from that readback to the real seal.*

12. *P is the* Pkg *provenance over* $\mathsf{SeriesLimitComparison}^{\uparrow}$, $\mathsf{LimitComparisonTest}^{\uparrow}$, $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeriesLimitComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, S, T, R, D, W, Q, E, H, C, P, N$. It has no coordinate for a host quotient limit, a host infinite sum, quotient equality of series, countable-choice selector, trichotomy row, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or standard bridge.*

Lean4 ✓ `BEDC.Derived.SeriesLimitComparisonUp.SeriesLimitComparisonUpTasteGate single carrier alignment`

Theorem 2023.2 (Series limit comparison NameCert obligations)**.** *For every accepted* $\mathsf{SeriesLimitComparison}^{\uparrow}$ *carrier $K = (A, B, S, T, R, D, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeriesLimitComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source series, comparator series, partial-sum threads, eventual positivity and ratio ledger, dyadic budget, stream windows, regular readback, real seal, transport, replay, provenance, and local naming rows;*

2. *source-thread exactness, requiring every source partial sum and tail observation consumed by the certificate to pass through A, S, W, Q, D;*

3. *comparator-thread exactness, requiring every comparator partial sum and tail observation consumed by the certificate to pass through B, T, W, Q, D;*

4. *eventual positive comparison, requiring the ledger R to display the finite threshold and the positive comparator rows used in every later comparison;*

5. *ratio-budget exactness, requiring R and D to display the carried dyadic multiple, slack, and tail budget before the source Cauchy handoff is read;*

6. *real-seal nonescape, requiring E to be downstream of the displayed* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *and* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *routes and to export no host limit, quotient series equality, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge datum;*

7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No series limit-comparison obligation is stored outside these rows.

Proof. Project the accepted packet $K = (A, B, S, T, R, D, W, Q, E, H, C, P, N)$. The carrier definition exposes exactly the source and comparator rows, their partial-sum threads, the ratio ledger, dyadic budget, finite windows, regular readback, real seal, and structural rows.

The source and comparator threads are read separately. The source side must pass through A, S, W, Q, D, while the comparator side must pass through B, T, W, Q, D.

The eventual positivity and ratio ledger R is then displayed. Its finite threshold and dyadic multiple are measured only against the tolerance and slack rows carried by D.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since no excluded analytic or choice object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2023.3 (Series limit comparison Cauchy handoff)**.** *Let $K = (A, B, S, T, R, D, W, Q, E, H, C, P, N)$ be an accepted* $\mathsf{SeriesLimitComparison}^{\uparrow}$ *carrier. Every completion-facing read of the source series factors through*

$$B, T, W, Q, D \longmapsto R \longmapsto A, S, W, Q, D \longmapsto E, H, C, P, N.$$

Thus comparator Cauchy control is handed to the source series only through the displayed eventual positive ratio ledger and dyadic tolerance budget, not through a host quotient limit, quotient equality of series, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing request for the source series. The comparator rows B, T, W, Q, D are the only comparator data available to the carrier, so the request first reads the comparator partial-sum and tolerance surface.

The ratio ledger R supplies the finite threshold, eventual positivity rows, and dyadic multiple used to compare later source terms with later comparator terms. This is a displayed ledger row, not a host limit assertion.

The request then returns to the source rows A, S, W, Q, D. The same stream windows, regular readback, and dyadic budget record the finite tail comparison needed by the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ route.

The seal E and structural rows H, C, P, N preserve this handoff. Since the carrier has no coordinate for quotient series equality, choice, trichotomy, or ambient completeness, every source completion read is exhausted by the displayed finite route. □

Theorem 2023.4 (Series limit comparison ratio nonescape)**.** *For every accepted* $\mathsf{SeriesLimitComparison}^{\uparrow}$ *carrier, the ratio ledger R cannot be read as a standalone quotient-limit object. Any consumer of R must also receive the source-series row, comparator-series row, stream-window family, regular readback, dyadic tolerance budget, real seal, transport table, continuation table, package provenance, and local naming row displayed in Definition 2023.1.*

Proof. Project the carrier of Definition 2023.1. The row R is listed between the two partial-sum threads and the dyadic budget, and the classifier compares accepted packets componentwise through the full tuple.

Apply Definition 2023.2. The eventual positive comparison and ratio-budget exactness rows require R to be displayed together with the dyadic budget D, stream windows W, and regular readback Q before a real-seal consumer can use it.

Apply Definition 2023.3. The only completion-facing use of the ratio ledger is the finite handoff from comparator Cauchy control to the source Cauchy route.

Therefore R is a local ledger row inside the finite packet. It cannot be detached as a host quotient limit or used to supply quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{SeriesLimitComparison}^{\uparrow}$ completion-facing read factors through source and comparator series rows, two partial-sum threads, an eventual positive ratio ledger,

a DyadicRatCore tolerance budget, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real-series-cauchy-criterion, regular-cauchy-comparison, streamname, regseqrat, dyadicratcore, real: $\mathsf{SeriesLimitComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two series rows, two partial-sum threads, eventual positivity, ratio-budget control, Cauchy handoff, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SeriesLimitComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeriesLimitComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2023.1, Definition 2023.2, Definition 2023.3, and Definition 2023.4 seed the finite constructive series limit-comparison route from comparator Cauchy control through eventual positive ratio budgeting to source-series RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeriesLimitComparison}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SeriesLimitComparisonUp`

Bridge status: none

Not claimed: This chapter does not close host limits, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source and comparator thread exactness, eventual positive ratio-ledger stability, DyadicRatCore and RegSeqRat budget handoff, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.SeriesLimitComparisonUp.

Constructive story: A $\mathsf{SeriesLimitComparison}^{\uparrow}$ packet is generated from source and comparator series rows, two partial-sum threads, an eventual positive comparison and ratio ledger, a DyadicRatCore tolerance budget, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2024

A Concrete Naming Certificate for $\mathsf{CantorBendixson}^{\uparrow}$

Following Cantor and Bendixson's derived-set theorem in the constructive closed-set reading, the $\mathsf{CantorBendixson}^{\uparrow}$ packet records a finite derivative-stage route for a displayed closed subset of the real line before any completion-facing consumer reads the retained kernel or the isolated-point ledger. It consumes the finite real-boundary chapters chapter 1752, chapter 1421, chapter 1713, chapter 126, chapter 301, chapter 291, and chapter 38. The packet names only the finite derivative data exposed by the carrier; it does not import the transfinite Cantor-Bendixson decomposition, choice of isolated points, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2024.1 (Cantor-Bendixson finite derivative carrier)**.** *A* $\mathsf{CantorBendixson}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, D, I, K, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the closed-set row. It records the displayed finite closed-set observations, interval enclosures, and located boundary reads supplied to the derivative packet.*
2. *D is the derivative-stage row. It names the finite stage budget and the finite successor-stage ledger used to pass from the current closed row to the next derived row.*
3. *I is the isolated-point ledger. It records only isolated points whose witnessing separating intervals, dyadic radii, and stage membership checks are displayed inside the packet.*
4. *K is the retained perfect-kernel row at the displayed finite stage. It is read as the residual closed row after the isolated ledger and derivative-stage rows have been consumed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the closed-set observations, isolating intervals, and retained-kernel reads are exposed to finite consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and derivative-stage ledger.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for the radii, comparisons, interval separations, and containment checks used by F, D, I, K, W, R.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary for the displayed closed-set and retained-kernel reads.*
9. *H records componentwise* hsame *transport for the closed-set, derivative-stage, isolated-ledger, kernel, window, readback, dyadic, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of every consumer request through*

$$F, D \longmapsto I, K \longmapsto W, Q \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CantorBendixson}^{\uparrow}$, $\mathsf{CantorCompleteness}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{LocatedRealNestedInt}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBendixson}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, D, I, K, W, R, Q, E, H, C, P, N$. It has no coordinate for a transfinite rank, a selected isolated-point enumeration, quotient real equality, countable choice, host closed-set equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2024.2 (Cantor-Bendixson NameCert obligations)**.** *For every accepted* $\mathsf{CantorBendixson}^{\uparrow}$ *carrier $B = (F, D, I, K, W, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorBendixson}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed closed-set, derivative-stage, isolated-ledger, retained-kernel, StreamName, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and local naming rows;*

2. *derivative-stage classifier stability, requiring accepted comparisons to preserve the finite stage budget and every displayed successor-stage derived-row check;*

3. *isolated-point ledger exactness, requiring each listed isolated point to carry its displayed separating interval, dyadic radius, and membership row, with no hidden selector for further isolated points;*

4. *retained-kernel handoff, requiring K to be downstream of the displayed derivative-stage and isolated-ledger rows before any StreamName, RegSeqRat, DyadicRatCore, or Real consumer reads it;*

5. *ledger nonescape, requiring W, R, Q, E, H, C, P, N to expose only the rows already displayed in F, D, I, K and to export no transfinite decomposition, selected enumeration, quotient real equality, countable choice, host closed-set equality, or ambient completeness.*

No Cantor-Bendixson NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CantorBendixsonUp.CantorBendixsonTasteGate single carrier alignment`

Proof. Project the accepted carrier $B = (F, D, I, K, W, R, Q, E, H, C, P, N)$. Carrier admission exposes exactly the closed-set row, finite derivative-stage row, isolated-point ledger, retained-kernel row, observation windows, regular readback, dyadic ledger, Real seal, and the structural rows.

The derivative-stage clause is finite. The row D records the stage budget and successor-stage checks, while H transports those checks componentwise and C replays them before the kernel row is inspected.

The isolated-point clause is also finite. Each entry in I carries its separating interval, dyadic radius, and membership check through Q, so the ledger lists displayed witnesses rather than a choice-selected enumeration.

The retained-kernel handoff follows from the placement of K after F, D, I and before W, R, Q, E. A consumer reaches the kernel only by the route recorded in the carrier.

Finally, provenance and local naming package the same rows. Since no carrier coordinate contains a transfinite rank, quotient real equality, countable choice, host closed-set equality, or ambient completeness theorem, the five listed rows exhaust the local NameCert surface. □

Theorem 2024.3 (Cantor-Bendixson finite derivative handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CantorBendixson}^{\uparrow}$ *carrier factors through the finite route displayed by chapter 1752, chapter 1421, chapter 1713, chapter 126, chapter 301, chapter 291, and chapter 38. Concretely, a consumer of* $B = (F, D, I, K, W, R, Q, E, H, C, P, N)$ *must read the closed-set row, finite derivative-stage ledger, isolated-point ledger, retained-kernel row, StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, transport, replay, provenance, and local naming before the Real seal* E *is exposed.*

Proof. Begin with a completion-facing request for the Real seal. Carrier projection shows that the only terminal Real-facing coordinate is E, and the continuation row C routes every access to E through the displayed packet.

The closed-set and derivative rows are read first. Their finite nested and interval observations are the part of the route governed by the Cantor completeness, Cantor intersection, and located real nested-interval chapters.

Next the request passes through the isolated-point ledger I and retained-kernel row K. The listed isolating intervals and residual closed row are exposed only by the finite stage route, not by a transfinite decomposition or a selector.

The observation and arithmetic rows then pass through W, R, Q. These are exactly the StreamName, RegSeqRat, and DyadicRatCore surfaces consumed before the terminal Real boundary.

The structural rows H, P, N bind the same path under transport, provenance, and local naming. Hence every Real-facing read factors through the seven cited finite boundary chapters and exports none of the excluded classical or quotient data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBendixson}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorBendixson}^{\uparrow}$ packet is generated from a displayed closed-set row, a finite derivative-stage row, an isolated-point ledger, a retained perfect-kernel row, StreamName windows, RegSeqRat readback, DyadicRatCore comparison data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorBendixson}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2024.1, Definition 2024.2, and Definition 2024.3 seed the finite Cantor-Bendixson derivative route from closed-set observations through displayed isolated-point ledgers and retained-kernel rows to Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorBendixson}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full transfinite Cantor-Bendixson decomposition, selected isolated points, quotient real equality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-stage classifier stability, isolated-point ledger exactness, retained-kernel handoff, ledger nonescape, and Real-facing handoff for BEDC.Derived.CantorBendixsonUp.

Chapter 2025

A Concrete Naming Certificate for $\mathsf{CompleteArchimedeanField}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley ordered-field completion viewpoint, the $\mathsf{CompleteArchimedeanField}^{\uparrow}$ packet names the finite certificate surface where a displayed real-name carrier is read simultaneously as an ordered field, an Archimedean dyadic-density object, and a Cauchy-completion route. It consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular-field route of chapter 1717, the ordered Archimedean surface of chapter 1419, the dyadic arithmetic ledger of chapter 291, the finite window interface of chapter 126, and the regular rational readback of chapter 301. The chapter records the capstone route needed by Real and Completion consumers without importing ambient completeness of all $\mathsf{Real}^{\uparrow}$ names, quotient real equality, trichotomy, choice, or bridge verification.

Definition 2025.1 (Complete Archimedean field carrier)**.** *A* $\mathsf{CompleteArchimedeanField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, O, A, D, W, Q, K, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, read only after the regular rational windows and dyadic tolerance rows have been displayed.*
2. *O is the ordered-field row supplied by the* $\mathsf{CauchyRealField}^{\uparrow}$ *and* $\mathsf{ArchimedeanOrderedField}^{\uparrow}$ *sibling surfaces, including addition, multiplication, order, and compatibility ledgers visible to the packet.*
3. *A is the Archimedean dyadic-density row, recording the finite rational and dyadic approximation witnesses that make the ordered-field read Archimedean at the displayed precision.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for dyadic scales, comparisons, density witnesses, and Cauchy tolerances used by O and A.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which inspected real-name observations are opened.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D before the terminal real seal is consumed.*
7. *K is the Cauchy-completion route row, recording the displayed passage from regular rational readback through the Real seal to the completion-facing field packet.*
8. *H records componentwise* hsame *transport for $R, O, A, D, W, Q, K, C, P, N$.*

9. *C records displayed* Cont *replay from a completion-facing request through $D, W, Q, R, O, A, K, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompleteArchimedeanField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealField}^{\uparrow}$, $\mathsf{ArchimedeanOrderedField}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteArchimedeanField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, O, A, D, W, Q, K, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient real equality, trichotomy, choice, or bridge verification.*

Lean4 ✓ `BEDC.Derived.CompleteArchimedeanFieldUp.CompleteArchimedeanFieldTasteGate single carrier alignment`

Theorem 2025.2 (Complete Archimedean field NameCert obligations)**.** *For every accepted* CompleteArchimedeanF *carrier*

$$F = (R, O, A, D, W, Q, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteArchimedeanField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Real seal, ordered-field row, Archimedean dyadic-density row, dyadic ledger, finite windows, regular rational readback, completion route, transport, replay, provenance, and local naming rows;*
2. *ordered-field exposure, requiring O to be the only row through which field operations, order, and algebra-order compatibility are read;*
3. *Archimedean density exposure, requiring A to carry the finite rational and dyadic approximation witnesses and to obtain all dyadic comparisons through D;*
4. *completion-route exactness, requiring K to pass through W, Q, R before any completion-facing field consumer may read the capstone packet;*
5. *transport and replay stability, requiring all classifier-visible data to move componentwise through H and to be replayed through C without adding another route;*
6. *provenance and local naming, requiring P and N to source the same displayed rows and no outside completeness principle.*

No Complete Archimedean field obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2025.1. The accepted packet displays exactly the Real seal, ordered-field row, Archimedean dyadic-density row, dyadic ledger, finite windows, regular rational readback, completion route, transport, replay, provenance, and local naming row.

The ordered-field and density clauses are then separated by their coordinates. The row O is the only field-and-order surface, while A records the finite Archimedean witnesses and reads every dyadic comparison through D.

The completion route is downstream of the regular rational route. A completion-facing consumer reaches K only after the finite windows W, the regular rational readback Q, and the terminal Real seal R have been displayed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, trichotomy, choice, and bridge verification are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2025.3 (Complete Archimedean field Real route)**.** *Every accepted* $\mathsf{CompleteArchimedeanField}^{\uparrow}$ *read that reaches a Real or Completion consumer factors through the displayed route*

$$D, W, Q \longmapsto R \longmapsto O, A, K \longmapsto H, C, P, N.$$

Thus the consumer receives the dyadic comparison ledger, finite StreamName windows, RegSeqRat readback, Real seal, ordered-field row, Archimedean dyadic-density row, Cauchy-completion route, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. The consumer receives no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient real equality, trichotomy principle, choice selector, or bridge verification credential.*

Proof. Project the accepted carrier $F = (R, O, A, D, W, Q, K, H, C, P, N)$. The finite observation side starts in the dyadic ledger D and the StreamName windows W, and the first regular rational readback available to the route is Q.

The terminal Real seal R is reached only after those rows are displayed. The ordered-field row O and the Archimedean dyadic-density row A are then read as visible coordinates attached to that same seal, while the completion route K records the passage into the completion-facing capstone packet.

The structural rows preserve this order. Componentwise transport through H keeps all rows aligned, replay through C repeats the displayed route, and P, N source and name the same packet. Because the excluded principles have no carrier coordinate, every Real or Completion read factors through the stated finite route. □

Falsifiable prediction: Every $\mathsf{CompleteArchimedeanField}^{\uparrow}$ read by a Real or Completion consumer factors through one DyadicRatCore comparison ledger, one StreamName finite-window row, one RegSeqRat readback row, one Real seal, one ordered-field row, one Archimedean dyadic-density row, one Cauchy-completion route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, cauchy-real-field, archimedeanorderedfield, dyadicratcore, streamname, regseqrat: $\mathsf{CompleteArchimedeanField}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Real seal, ordered-field row, Archimedean dyadic-density row, Cauchy-completion route, and the finite readback ledgers that make the capstone route visible as one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteArchimedeanField}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteArchimedeanField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2025.1, Definition 2025.2, and Definition 2025.3 seed the finite capstone route from dyadic and regular rational readback through the Real seal to ordered-field, Archimedean, and completion-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteArchimedeanField}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CompleteArchimedeanFieldUp`

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, trichotomy, choice, or bridge verification.

Upgrade path: Obligation closure requires checked carrier admission, ordered-field exposure, Archimedean dyadic-density exposure, completion-route exactness, transport and replay stability, provenance, and local naming for BEDC.Derived.CompleteArchime

Constructive story: A $\mathsf{CompleteArchimedeanField}^{\uparrow}$ packet is generated from a Real seal, an ordered-field row, an Archimedean dyadic-density row, a DyadicRatCore ledger, StreamName windows, RegSeqRat readback, a Cauchy-completion route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2026

A Concrete Naming Certificate for $\mathsf{EffectiveCompletion}^{\uparrow}$

Following Weihrauch-style computable analysis and Bishop–Bridges constructive completion, the $\mathsf{EffectiveCompletion}^{\uparrow}$ packet records effective Cauchy completion data as a finite NameCert surface rather than as a quotient object. It sits after the effective Cauchy source of chapter 1705, the Cauchy realizer handoff of chapter 1771, the represented-name boundary of chapter 1680, the admissible-representation surface of chapter 1760, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter gives Real-facing completion consumers a computable-analysis route through displayed finite data without claiming quotient completion, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 2026.1 (Effective completion carrier)**.** *An* $\mathsf{EffectiveCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, Q, K, W, R, D, A, U, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row, exposing the effective index schedule, Cauchy evidence, and source-side observations before any completion consumer reads the packet.*
2. *Q is the* $\mathsf{CauchyRealizer}^{\uparrow}$ *handoff row, recording the effective procedure that answers tolerance requests by finite Cauchy data read from E.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row, binding the represented-space name to the same effective Cauchy source and realizer route.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the source, realizer, representation, and downstream completion request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the displayed windows and their tolerance ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording dyadic scales, comparison witnesses, slack, and radius data used by Q, W, R.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only through the displayed effective source, realizer, representation, windows, readback, and dyadic tolerance rows.*

8. *U is the completion-consumer row, recording every Real-facing or completion-facing read that the packet permits after E, Q, K, W, R, D, A have been displayed.*

9. *H records componentwise* hsame *transport for the effective source, realizer, representation, windows, readback, tolerance ledger, real seal, completion consumer, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$E, Q, K \longmapsto W, D \longmapsto R \longmapsto A \longmapsto U.$$

11. *P is the* Pkg *provenance over* $\mathsf{EffectiveCompletion}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{CauchyRealizer}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, Q, K, W, R, D, A, U, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, countable choice, host equality of completed objects, trichotomy of reals, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EffectiveCompletionUp.EffectiveCompletionTasteGate single carrier alignment`

Theorem 2026.2 (Effective completion NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCompletion}^{\uparrow}$ *carrier*

$$X = (E, Q, K, W, R, D, A, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the effective Cauchy source, Cauchy realizer, Cauchy representation, finite windows, regular rational readback, dyadic tolerance ledger, real seal, completion consumer, transport, replay, provenance, and local naming row;*

2. *effective-source visibility, requiring every downstream read to expose the source schedule and Cauchy evidence in E before any represented-space or real-facing output is inspected;*

3. *finite-window exactness, requiring every observation used by the realizer, representation, regular readback, real seal, or completion consumer to occur inside the displayed* $\mathsf{StreamName}^{\uparrow}$ *row W;*

4. *Cauchy-realizer handoff, requiring tolerance requests to pass through Q and its displayed dyadic ledger D before R or A can be read;*

5. *represented-space compatibility, requiring the row K to bind the same effective source, realizer, finite windows, dyadic tolerance data, and real seal as the admissible representation route;*

6. *completion-consumer non-escape, requiring U to consume only E, Q, K, W, R, D, A and to export no quotient completion, selected limit, countable-choice selector, host equality, trichotomy branch, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential;*

7. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve, replay, source, and name the displayed route without adding a second classifier.*

No effective completion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2026.1. The accepted packet displays exactly the effective source, realizer, representation, finite windows, regular readback, dyadic tolerance ledger, real seal, completion consumer, and structural rows.

The effective-source and finite-window clauses are read from E and W. Since every downstream row is listed downstream of those coordinates, no represented-space read, regular readback, real seal, or completion consumer can inspect hidden stream data.

The realizer and represented-space clauses are read from Q, K, D, R, A. A tolerance request reaches Q, opens only the displayed dyadic data D and windows W, forms the readback R, and then reaches the terminal real seal A through the represented route K.

The consumer and structural clauses are exhausted by U, H, C, P, N. The carrier has no coordinate for quotient completion, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, so these objects cannot be exported by replay or local naming. □

Theorem 2026.3 (Effective completion Cauchy-realizer handoff)**.** *For every accepted* $\mathsf{EffectiveCompletion}^{\uparrow}$ *carrier*

$$X = (E, Q, K, W, R, D, A, U, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *or completion consumer of* X *factors through the displayed route*

$$E \longrightarrow Q \longrightarrow K \longrightarrow W, D \longrightarrow R \longrightarrow A \longrightarrow U$$

with transport, replay, provenance, and local naming supplied only by H, C, P, N*. The consumer receives no quotient completion, selected limit, countable choice, host equality of completed objects, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.EffectiveCompletionUp.EffectiveCompletionCauchyRealizerHandoff`

Proof. Fix a Real-facing or completion-facing consumer of X. The only consumer coordinate is U, so the read must pass through the displayed completion-consumer row.

Read the dependencies of U. It is downstream of the real seal A, and A is downstream of the regular readback R generated from finite stream windows W and dyadic tolerance data D.

The effective source of those rows is fixed by E, Q, K. The source row E supplies the effective Cauchy data, the realizer Q answers tolerance requests from that source, and the representation row K binds the same data to the represented-space interface.

The structural rows add no analytic data. Componentwise transport through H preserves the displayed coordinates, replay through C preserves the route order, and P, N only source and name the same packet.

Since no excluded object appears as a carrier coordinate, the consumer cannot obtain quotient completion equality, a selected limit, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Thus every allowed consumer factors through exactly the displayed Cauchy-realizer handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2026.1, Definition 2026.2, and Definition 2026.3 seed the finite computable-analysis completion route from effective Cauchy data to Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.EffectiveCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective-source visibility, finite-window exactness, Cauchy-realizer handoff, represented-space compatibility, completion-consumer non-escape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCompletionUp.

Constructive story: An EffectiveCompletion$^{\uparrow}$ packet is generated from an EffectiveCauchySequence source, a CauchyRealizer handoff, a CauchyRepresentation row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a completion-consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2027

A Concrete Naming Certificate for $\mathsf{FanContinuityPrinciple}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on fan functionals and continuous choice-sequence functionals, the $\mathsf{FanContinuityPrinciple}^{\uparrow}$ packet records the constructive fan-continuity principle as finite NameCert data rather than as host function-space continuity. Its dependency surface joins the finite bar depth of chapter 400, the prefix-cylinder context of chapter 334, the finite observation row of chapter 1482, the window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives later Real and Completion consumers one finite route for extracting a common prefix modulus from fan-bar data, without asserting open-cover compactness, host equality of functionals, countable choice, quotient streams, selected real limits, trichotomy, or ambient real completeness.

Definition 2027.1 (Fan continuity principle carrier)**.** *A* $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, A, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite bar-depth row, carrying a binary spread, a decidable bar classifier, and the displayed depth ledger that bounds the fan read.*
2. *A is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing the finite prefix context in which a Baire-name observation is inspected.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *finite observation row, listing the already displayed choice-sequence answers whose common prefix controls the read.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for the fan prefix, Baire prefix, and choice-sequence observation rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the finite prefix data are consumed by a regular-rational route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, A, Q, S, R have been replayed.*
7. *H records componentwise* hsame *transport for the bar-depth, prefix-cylinder, finite-observation, stream-window, readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$B, A, Q, S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{FanContinuityPrinciple}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FanContinuityPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, A, Q, S, R, E, H, C, P, N$ and the displayed replay order. It has no coordinate for open-cover compactness, host function equality, countable choice, quotient streams, selected $\mathsf{Real}^{\uparrow}$ *limits, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FanContinuityPrincipleUp.FanContinuityPrincipleTasteGate single carrier alignment`

Theorem 2027.2 (Fan continuity principle modulus extraction)**.** *Every accepted* $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *carrier*

$$F = (B, A, Q, S, R, E, H, C, P, N)$$

exports a finite prefix modulus for the displayed real-facing read: the modulus is the depth ledger supplied by the $\mathsf{FanTheorem}^{\uparrow}$ *row B, restricted to the prefix-cylinder row A, witnessed by the finite* $\mathsf{ChoiceSequence}^{\uparrow}$ *observation row Q, scheduled through S, and read back through R before reaching the* $\mathsf{Real}^{\uparrow}$ *seal E.*

Proof. Project the accepted carrier. The only depth datum in the packet is the finite bar-depth row B, and the only prefix context is the $\mathsf{BaireSpace}^{\uparrow}$ row A.

Use the displayed choice-sequence observation row Q next. Its entries are finite answers already visible through the stream window schedule S, so any two inspected choice sequences agreeing on the displayed B-depth have the same Q-prefix inside the A-cylinder.

The readback step is then forced through R. Since the continuation route is displayed as $B, A, Q, S \longmapsto R \longmapsto E$, the real-facing seal E can be reached only after the finite prefix data have been read back.

Thus the depth ledger in B, viewed inside A and witnessed by Q, S, R, is the exported modulus. No host function-space continuity or selected infinite branch is used. □

Theorem 2027.3 (Fan continuity principle bar stability)**.** *Let*

$$F = (B, A, Q, S, R, E, H, C, P, N)$$

and

$$F' = (B', A', Q', S', R', E', H', C', P', N')$$

be accepted $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *carriers whose displayed rows are related by componentwise* hsame *transport and whose* Cont *routes replay the same order. Then the prefix modulus extracted in Definition 2027.2 transports from F to F' by carrying the fan bar-depth, Baire prefix-cylinder, choice-sequence observation, StreamName window, RegSeqRat readback, and RealUp seal rows componentwise.*

Proof. Apply Definition 2027.2 to F. The extracted modulus is exactly the depth row B together with the surrounding prefix-cylinder, observation, window, readback, and seal rows.

Now inspect the transport rows H and H'. They compare B, A, Q, S, R, E with B', A', Q', S', R', E' componentwise, so the displayed depth ledger and every row used to read it are carried to the second packet.

Finally use the Cont replay rows C and C'. They preserve the order $B, A, Q, S \longmapsto R \longmapsto E$, so the transported packet still reads the same finite bar before opening the real-facing seal.

Therefore the modulus is stable under the displayed carrier classifier. The argument has no place to introduce open-cover compactness, host function equality, countable choice, quotient streams, selected limits, trichotomy, ambient real completeness, Lean verification, or bridge checking. □

Theorem 2027.4 (Fan continuity principle NameCert obligations)**.** *For the carrier of Definition 2027.1, the local* $\mathsf{NameCert}_{\mathsf{FanContinuityPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed fan bar-depth, Baire prefix-cylinder, choice-sequence observation, stream-window, readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *modulus extraction, requiring the exported prefix modulus to be the finite depth ledger of* B *read through* A, Q, S, R *before* E*;*
3. *bar stability, requiring componentwise* hsame *transport to preserve the same depth ledger and the same replay order;*
4. *real-seal nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through the displayed route* $B, A, Q, S \longmapsto R \longmapsto E$*;*
5. *provenance exactness, requiring* P *and* N *to name only the listed sibling chapters and the local certificate row.*

No local obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2027.1. The packet displays precisely the rows $B, A, Q, S, R, E, H, C, P, N$.

For modulus extraction, use Definition 2027.2. It identifies the exported prefix modulus with the finite bar-depth row B after restriction to the prefix and observation rows A, Q, S, R.

For stability, use Definition 2027.3. The componentwise transport rows preserve the depth ledger and every sibling row required for its replay.

The nonescape and provenance obligations are the remaining projections. The continuation row displays the only route to the real seal, and P, N record only package provenance and local naming for the same finite carrier. Since the carrier inventory contains no further coordinates, the five listed obligations exhaust the NameCert surface. □

Theorem 2027.5 (Fan continuity principle sibling dependency)**.** *Every accepted* $\mathsf{FanContinuityPrinciple}^{\uparrow}$ *packet depends on the sibling chapters chapter 400, chapter 334, chapter 1482, chapter 126, chapter 301, and chapter 38 only through the rows* B, A, Q, S, R, E *displayed in Definition 2027.1. It exports no independent compactness principle, function-space equality, countable-choice selector, quotient-stream equality, selected* $\mathsf{Real}^{\uparrow}$ *limit, trichotomy row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness row, Lean verification row, or bridge-checking row.*

Proof. Read the carrier rows in order. The fan dependency is exactly B, the Baire dependency is exactly A, the choice-sequence dependency is exactly Q, the stream dependency is exactly S, the regular-rational dependency is exactly R, and the real dependency is exactly E.

The structural rows H, C, P, N do not add further sibling mathematics. They transport the displayed rows, replay the displayed route, record package provenance, and name the local certificate.

Thus every sibling reference factors through the listed finite rows. The excluded compactness, equality, choice, quotient, limit, order, completeness, verification, and bridge data cannot be exported because they are absent from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanContinuityPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{FanContinuityPrinciple}^{\uparrow}$ packet is generated from a FanTheorem finite bar-depth row, a BaireSpace prefix-cylinder row, a ChoiceSequence finite observation row, a StreamName window schedule, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanContinuityPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2027.1, Definition 2027.2, Definition 2027.3, Definition 2027.4, and Definition 2027.5 seed the finite fan-continuity route joining fan bars, Baire prefixes, choice-sequence observations, StreamName windows, RegSeqRat readback, and RealUp seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanContinuityPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host function equality, countable choice, quotient streams, selected Real limits, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus extraction, bar stability, real-seal nonescape, provenance exactness, sibling dependency, and local NameCert rows for BEDC.Derived.FanContinuityPrincipleUp.

Chapter 2028

A Concrete Naming Certificate for $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, this packet records the real-name route whose equality and seal reads pass through displayed Cauchy classifier data, not quotient streams, selected limits, countable choice, host real equality, or ambient completeness. Its dependency surface is the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the classifier row of chapter 1820, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter isolates the finite L10 handoff used by later Real and Completion consumers when the route must remain selection-free.

Definition 2028.1 (Selection-free Cauchy real carrier)**.** *A* $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, Q, E, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which every finite observation of the Cauchy name is opened.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bounds used by the regular readback and by Cauchy comparison requests.*
4. *Q is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row, recording equality of accepted names only through displayed Cauchy comparison data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after S, R, D, Q have been displayed.*
6. *B is the no-choice boundary, recording that the packet exports no quotient stream representative, selected limit, countable-choice schedule, host real equality, or ambient completeness witness.*
7. *H records componentwise* hsame *transport for the window, readback, dyadic ledger, classifier, seal, boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of every equality or seal read through*

$$S, R, D, Q, B \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SelectionFreeCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, E, B, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.SelectionFreeCauchyRealUp.TasteGate.SelectionFreeCauchyRealTasteGate single carrier alignment`

Theorem 2028.2 (Selection-free Cauchy real NameCert obligations)**.** *For every accepted* $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$ *carrier*

$$K = (S, R, D, Q, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SelectionFreeCauchyReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream windows, regular readback, dyadic tolerance ledger, Cauchy classifier, Real seal, no-choice boundary, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring every observation used by equality or seal reads to be one of the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows in S;*
3. *regular readback, requiring each rational observation consumed downstream to pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. *dyadic tolerance exactness, requiring every Cauchy comparison budget to be the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
5. *classifier-law stability, requiring the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *row Q to preserve reflexive, symmetric, and transitive Cauchy comparison only through S, R, D;*
6. *Real seal boundary, requiring E to be read only after S, R, D, Q and the no-choice boundary B have been displayed;*
7. *no-choice refusal, requiring B to exclude quotient streams, selected limits, countable choice, host real equality, and ambient completeness;*
8. *structural rows, requiring H, C, P, N to transport, replay, source, and name exactly the displayed route.*

No selection-free Cauchy real NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2028.1. The visible packet consists of the stream windows, regular readback, dyadic ledger, Cauchy classifier, Real seal, no-choice boundary, and structural rows.

Window exactness and regular readback are forced by the first two observation coordinates. A downstream read may inspect only S, and every rational value extracted from those windows must pass through R.

Dyadic tolerance exactness is the row D. It supplies the comparison budgets used by the Cauchy classifier and by any later seal-facing request.

Classifier-law stability is local to Q. The classifier may use the displayed S, R, D rows to compare accepted names, but it has no coordinate for quotient stream equality or host real equality.

Finally, the Real seal boundary and the no-choice refusal are recorded by E and B. The rows H, C, P, N preserve the same route by transport, replay, provenance, and local naming, so the listed rows exhaust the local NameCert surface. □

Theorem 2028.3 (Selection-free Cauchy real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealCompleteness}^{\uparrow}$ *consumer of an accepted* $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$ *carrier*

$$K = (S, R, D, Q, E, B, H, C, P, N)$$

reaches the seal row E only after the stream windows S, regular readback R, dyadic tolerance ledger D, Cauchy classifier row Q, no-choice boundary B, transport row H, replay row C, provenance row P, and local naming row N have been displayed. Consequently such a consumer receives a finite Cauchy real-name route and receives no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem from the packet.

Proof. Let a $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealCompleteness}^{\uparrow}$ consumer read an accepted carrier. Projection from Definition 2028.1 gives only the rows $S, R, D, Q, E, B, H, C, P, N$.

The real-name route starts with the displayed windows S. The readback row R supplies regular rational observations, and the dyadic ledger D supplies the tolerance budget attached to those observations.

The equality-facing step then passes through Q. Its Cauchy classifier laws are read before the Real seal and only through the displayed window, readback, and dyadic rows.

The no-choice row B is part of the same route before E is exposed. It blocks quotient representatives, selected limits, choice schedules, host real equality, and ambient completeness witnesses from becoming seal data.

The structural rows H, C, P, N transport, replay, source, and name the route without adding further mathematical coordinates. Thus every Real or Completion consumer reaches E only through the displayed selection-free boundary. □

Theorem 2028.4 (Selection-free Cauchy real modulus boundary)**.** *Let $K = (S, R, D, Q, E, B, H, C, P, N)$ be an accepted* $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$ *carrier. Any refinement of a selection-free Cauchy real-name read must expose the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row Q before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is available. The refinement exports no quotient representative, selected limit, host real equality, countable choice, or ambient completeness theorem.*

Proof. Project the carrier of Definition 2028.1. The only observation rows are S, R, D, the only Cauchy classifier row is Q, and the only terminal Real seal is E.

Apply Definition 2028.2. Window exactness, regular readback, dyadic tolerance exactness, and classifier-law stability all occur before the Real seal boundary clause.

Apply Definition 2028.3. It forces every Real or Completion consumer through $S, R, D, Q, B, H, C, P, N$ before E is public, so a refinement can only make the displayed modulus data more precise inside those rows.

The boundary row B and structural rows H, C, P, N add no quotient representative, selected limit, host real equality, countable-choice schedule, or ambient completeness theorem. Thus the refined read remains selection-free and cannot bypass the displayed modulus rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{SelectionFreeCauchyReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SelectionFreeCauchyReal}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a RealUp seal, a no-choice boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SelectionFreeCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2028.1, Definition 2028.2, and Definition 2028.3 seed the finite selection-free Cauchy real-name route from displayed Cauchy classifier data to the public Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SelectionFreeCauchyReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.SelectionFreeCauchyRealUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, regular readback, dyadic tolerance exactness, classifier-law stability, Real seal boundary, no-choice refusal, and local NameCert obligations for BEDC.Derived.SelectionFreeCauchyRealUp.

Chapter 2029

A Concrete Naming Certificate for $\mathsf{LocatedPositiveReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedPositiveReal}^{\uparrow}$ packet records a positive real together with the displayed located-apartness route by which its separation from zero is inspected. It refines the bare positivity surface of chapter 1587: a consumer sees not only that the $\mathsf{Real}^{\uparrow}$ name is apart from zero, but also the finite radius, window, regular readback, and located-apartness handoff that make the witness inspectable. The dependency surface is the sealed $\mathsf{Real}^{\uparrow}$ row of chapter 38, the positive-real row of chapter 1587, the separation row of chapter 304, the located handoff of chapter 1380, the located-real boundary of **??**, the dyadic radius ledger of chapter 291, the stream-window interface of chapter 126, and the regular-rational readback of chapter 301.

Definition 2029.1 (Located positive real carrier)**.** *A* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *carrier is a finite packet*

$$X = (R, P, A, L, D, W, Q, H, C, S, N)$$

with the following displayed rows:

1. *R is the sealed* $\mathsf{Real}^{\uparrow}$ *row for the real-name presentation being certified.*
2. *P is the* $\mathsf{PositiveReal}^{\uparrow}$ *row naming positivity by apartness from the zero real-name row.*
3. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *separation-to-zero row whose endpoints are the zero real-name row and R.*
4. *L is the* $\mathsf{LocatedApartness}^{\uparrow}$ *handoff row displaying how the separation witness is inspected at the requested finite window.*
5. *D is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row used by A and exposed again by L.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the zero row, R, and the radius comparison are observed.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to the rational observations used by the radius comparison.*
8. *H records componentwise* hsame *transport for the real seal, positive row, apartness row, located handoff, dyadic radius, stream window, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of every consumer route through*

$$P, A, L, D, W, Q \longmapsto R.$$

10. *S is the* Pkg *provenance over* $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedPositiveReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, P, A, L, D, W, Q, H, C, S, N$. It has no coordinate for trichotomy, a total order on constructive reals, quotient stream equality, a selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.LocatedPositiveRealUp.LocatedPositiveRealTasteGate single carrier alignment`

Theorem 2029.2 (Located positive real NameCert obligations)**.** *For every accepted* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *carrier*

$$X = (R, P, A, L, D, W, Q, H, C, S, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedPositiveReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real seal, positive-real row, apartness-to-zero row, located-apartness handoff, positive dyadic radius, stream window, regular readback, transport, replay, provenance, and local naming rows;*
2. *positivity alignment, requiring P to read its positive witness only through the same A, D, W, Q rows displayed by the carrier;*
3. *located-apartness exposure, requiring L to display the inspection route for A before any metric, order, or completion-facing consumer can use the separation from zero;*
4. *radius exactness, requiring D to be the unique positive dyadic radius row exported by both the apartness and located-apartness coordinates;*
5. *real-seal factorization, requiring every read of R as a positive located object to pass through P, A, L, D, W, Q first;*
6. *transport and replay stability, requiring H, C, S, N to preserve and name exactly the displayed rows, with no trichotomy, total order, quotient stream equality, selected limit, ambient completeness, Lean verification, or bridge credential.*

No located-positive-real NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 2029.1. Carrier admission exposes exactly the sealed real row, the positive-real row, the apartness-to-zero row, the located-apartness handoff, the positive dyadic radius, the stream window, the regular readback, and the four structural rows.

The positivity alignment is the comparison between P and A. The $\mathsf{PositiveReal}^{\uparrow}$ row names positivity only by apartness from zero, and the apartness row displayed here is the sole separation-to-zero source in the carrier.

Located exposure is then supplied by L. It makes the finite inspection route visible before a downstream consumer can use the separation, so the route passes through the located-apartness handoff rather than through an uninspected order predicate.

The radius and readback rows exhaust the witness data. The positive radius is D, and the finite observations supporting it are the window W and readback Q.

Finally, componentwise transport and replay through H and C preserve the same rows, while S and N only record provenance and local naming. Since the carrier contains no trichotomy, total order, quotient stream equality, selected limit, ambient completeness, Lean verification, or bridge credential, the listed rows are the whole obligation surface. □

Theorem 2029.3 (Located positive real apartness witness)**.** *Every accepted* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *carrier*

$$X = (R, P, A, L, D, W, Q, H, C, S, N)$$

exports a displayed located-apartness witness for the positivity of R*: the witness is the finite route from the* $\mathsf{PositiveReal}^{\uparrow}$ *row* P *through the* $\mathsf{ApartnessReal}^{\uparrow}$ *separation row* A*, the* $\mathsf{LocatedApartness}^{\uparrow}$ *handoff* L*, the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*. This route is available without a trichotomy decision, a total-order comparison, quotient stream equality, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier. By Definition 2029.1, the positive row P is displayed in the same finite packet as the sealed real row R and the separation-to-zero row A.

Read the separation witness. The apartness row A names the zero-to-R gap, while D records the positive dyadic radius used by that gap.

The witness is located because L displays the finite inspection route for the same gap. The stream window W and regular readback Q are the only observation rows available to that route.

Thus every exported positivity witness is the displayed route P, A, L, D, W, Q. The excluded order, quotient, selected-limit, and completeness data are absent from the carrier inventory. □

Theorem 2029.4 (Located positive real transport stability)**.** *Let*

$$X = (R, P, A, L, D, W, Q, H, C, S, N)$$

and

$$X' = (R', P', A', L', D', W', Q', H', C', S', N')$$

be accepted $\mathsf{LocatedPositiveReal}^{\uparrow}$ *carriers related by the displayed componentwise* hsame *and* Cont *routes. Then the located positivity witness of Definition 2029.3 transports from* X *to* X' *by carrying the same real-seal, positive row, apartness row, located handoff, radius, window, readback, provenance, and local naming surfaces componentwise. No transported read may replace that witness by trichotomy, a total order, quotient stream equality, a selected limit, or ambient completeness.*

Proof. Apply Definition 2029.3 to X. The witness surface consists only of P, A, L, D, W, Q together with the surrounding seal, transport, replay, provenance, and local naming rows.

Now use the displayed hsame rows in H and H'. They compare the real seal, positive row, apartness row, located handoff, radius, window, and readback componentwise, so the same witness surface is carried to P', A', L', D', W', Q'.

Replay through C and C' preserves the order of the route. The provenance and local naming rows S, N, S', N' merely record the same packet boundary.

Therefore the transported object still exports located positivity by the displayed apartness route. The excluded order, quotient, selected-limit, and completeness coordinates cannot appear under transport because they are not coordinates of either accepted carrier. □

Theorem 2029.5 (Located positive real nonescape)**.** *Every metric, order-facing, or completion-facing consumer of an accepted* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *carrier receives only the rows*

$$R, P, A, L, D, W, Q, H, C, S, N$$

before using the positivity of the displayed real name. In particular, the consumer cannot extract trichotomy for $\mathsf{Real}^{\uparrow}$*, a total real order, quotient stream equality, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge from the located-positive-real packet.*

Proof. Project the accepted packet. Its only real-facing coordinate is R, and its only positivity coordinate is the displayed route through P, A, L, D, W, Q.

Any consumer of positivity must therefore read P and then the apartness and located-apartness rows A, L. The dyadic radius D, stream window W, and readback Q are the finite witness surface attached to that route.

The structural rows H, C, S, N do not create further mathematical data. They transport, replay, source, and name the rows already displayed by the carrier.

The packet inventory is exhausted by the displayed rows. Hence none of the excluded trichotomy, total-order, quotient-equality, selected-limit, ambient-completeness, Lean-verification, or bridge data can escape from the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedPositiveReal}\uparrow}$.

Constructive story: A $\mathsf{LocatedPositiveReal}^{\uparrow}$ packet is generated from a sealed RealUp row, a PositiveRealUp positivity row, an ApartnessRealUp separation-to-zero row, a LocatedApartnessUp handoff, a positive DyadicRatCoreUp radius, a StreamNameUp window, a RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedPositiveReal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2029.1, Definition 2029.2, Definition 2029.3, Definition 2029.4, and Definition 2029.5 seed the finite located-apartness witness route for positive real packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedPositiveReal}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedPositiveRealUp.LocatedPositiveRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, total order, quotient stream equality, selected limits, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, positivity alignment, located-apartness exposure, radius exactness, real-seal factorization, transport/replay/provenance stability, and non-escape for BEDC.Derived.LocatedPositiveRealUp.

Chapter 2030

A Concrete Naming Certificate for $\mathsf{BrouwerFixedPoint}^{\uparrow}$

Following Brouwer and Sperner, the $\mathsf{BrouwerFixedPoint}^{\uparrow}$ packet names the finite dyadic-subdivision route by which a compact convex self-map is inspected before any fixed point is read. It is grounded in the finite labeled-simplex certificate of chapter 2003, the simplex inventory of chapter 238, the dyadic subdivision and cover rows of chapter 1863 and chapter 446, the self-map interface of chapter 130, and the sealed real-name surface of chapter 38. The chapter records a finite NameCert route for approximate fixed-simplex readback, not a host fixed-point choice.

Definition 2030.1 (Brouwer fixed-point carrier)**.** *A* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *carrier is a finite packet*

$$B = (K, S, D, L, W, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact convex dyadic-domain row. It records the accepted finite simplex or polytope inventory, face incidence, convex-combination closure, and dyadic mesh scale used by the self-map packet.*
2. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *row naming the carried finite simplicial subdivision of K.*
3. *D is the paired* $\mathsf{DyadicPartition}^{\uparrow}$ *and* $\mathsf{DyadicCover}^{\uparrow}$ *row. It records the dyadic cells, cell-face incidence, mesh refinement, and finite cover routing used by the labelled subdivision.*
4. *L is the Sperner-labelling row. It contains the boundary-admissible labels induced by the displacement of the self-map and the finite transport of those labels along the faces of S and cells of D.*
5. *W is the approximate fixed-simplex witness row. It records the fully labelled simplex supplied through* $\mathsf{SpernerLemma}^{\uparrow}$*, the associated dyadic cell, and the finite residual bounds read from the self-map rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the real-name readback used to keep the approximate fixed simplex inside the displayed compact convex domain and to prevent escape through an ambient completion principle.*
7. *H records componentwise* hsame *transport for the domain, simplicial subdivision, dyadic partition and cover, labelling, witness, seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for every consumer route through*

$$K, S, D, L, W \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{SpernerLemma}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, D, L, W, E, H, C, P, N$. It has no coordinate for a host fixed-point choice, open-cover compactness, real trichotomy, quotient stream equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BrouwerFixedPointUp.BrouwerFixedPointTasteGate single carrier alignment`

Theorem 2030.2 (Brouwer fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *carrier*

$$B = (K, S, D, L, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerFixedPoint}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed compact convex dyadic domain, simplicial subdivision, dyadic partition and cover, Sperner labelling, approximate fixed-simplex witness, RealUp seal, transport, replay, provenance, and local naming rows;*
2. *boundary labelling, requiring L to read all boundary colors from the displacement rows of the carried* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map and from the boundary faces displayed by S and D;*
3. *Sperner handoff, requiring the fully labelled simplex coordinate of W to enter only through the* $\mathsf{SpernerLemma}^{\uparrow}$ *packet and its finite labelled-simplex route;*
4. *approximate fixed-point readback, requiring the residual and cell-diameter data in W to be read through K, S, D, L before a consumer can inspect the approximate fixed simplex;*
5. *RealUp seal nonescape, requiring E to keep the readback inside the displayed compact convex dyadic domain without adding host fixed-point choice, open-cover compactness, real trichotomy, quotient stream equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Brouwer fixed-point NameCert obligation is stored outside these finite rows.

Proof. Carrier admission is projection from Definition 2030.1. The visible packet lists the compact convex dyadic domain row K, the simplicial subdivision row S, the dyadic partition and cover row D, the labelling row L, the witness row W, and the structural rows E, H, C, P, N.

Boundary labelling is the row L. Its admissible colors are read after the finite boundary faces of S and the dyadic cell incidence of D have been displayed, and the only map data available to the read is the carried $\mathsf{ContinuousMap}^{\uparrow}$ self-map surface named in the provenance row.

The Sperner handoff is the fully labelled simplex part of W. Since the carrier places that witness downstream of L and records $\mathsf{SpernerLemma}^{\uparrow}$ in P, a consumer reaches the labelled simplex only through the finite Sperner route.

Approximate fixed-point readback uses the same witness row W together with the mesh and residual information from K, S, D, L. These rows determine the displayed approximate fixed simplex before the seal row is inspected.

Finally, E seals the readback as a $\mathsf{Real}^{\uparrow}$-facing object inside the carried domain. The remaining rows transport, replay, source, and name the same finite packet, so no excluded host fixed-point, open-cover, trichotomy, quotient, selected-limit, completeness, verification, or bridge data can appear as an extra obligation. □

Theorem 2030.3 (Brouwer fixed-point Sperner handoff)**.** *Every accepted* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *fixed-point read factors through the displayed route*

$$\mathsf{SpernerLemma}^{\uparrow} \longmapsto \mathsf{SimplicialComplex}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicCover}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the read uses the carrier rows $K, S, D, L, W, E, H, C, P, N$ *to obtain a finite approximate fixed-simplex certificate. It exports no host fixed-point choice, open-cover compactness, real trichotomy, quotient stream equality, selected limit, or ambient Real completeness.*

Proof. Project the accepted carrier. The only row that exposes a fixed-point-facing witness is W, and W is specified as the approximate fixed-simplex witness supplied through the finite labelled-simplex route of $\mathsf{SpernerLemma}^{\uparrow}$.

Reading W first requires the finite geometry rows S and D. These provide the simplicial-complex inventory, dyadic partition, and dyadic cover routing on which the Sperner labels are meaningful.

The labels in L are tied to the carried $\mathsf{ContinuousMap}^{\uparrow}$ self-map surface. Their displacement readback is then sealed by E as a $\mathsf{Real}^{\uparrow}$-facing object inside the compact convex dyadic domain row K.

Thus every fixed-point read factors through the named finite route and then through the structural rows H, C, P, N. Since none of the carrier coordinates is a host fixed-point choice, an open-cover compactness theorem, real trichotomy, quotient stream equality, a selected limit, or ambient Real completeness, none of those objects is exported. □

Theorem 2030.4 (Brouwer fixed-point finite residual stability)**.** *Let*

$$B = (K, S, D, L, W, E, H, C, P, N)$$

and

$$B' = (K', S', D', L', W', E', H', C', P', N')$$

be accepted $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *packets. If the displayed transport rows identify* $K, S, D, L, W, E, H, C, P, N$ *componentwise with* $K', S', D', L', W', E', H', C', P', N'$ *by* hsame*, then the residual bounds read from* W *are transported only through the same dyadic subdivision row* D*, the same Sperner labelling row* L*, the carried* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. In particular, no host fixed-point choice, selected limit, trichotomy datum, quotient stream equality, or ambient Real completeness row can alter the residual comparison.*

Proof. Project the two accepted carriers. The residual bounds occur only in the approximate fixed-simplex witness row W, and the carrier places that row downstream of the compact domain, subdivision, dyadic partition, and Sperner labelling rows.

Use the Sperner handoff of Definition 2030.3. It forces the labelled simplex in W through the finite route from $\mathsf{SpernerLemma}^{\uparrow}$ to $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicCover}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and finally $\mathsf{Real}^{\uparrow}$.

The rows D and L are read before W, so the dyadic subdivision and Sperner labels are fixed before any residual bound is public. Componentwise transport through H carries those rows together with the witness row, and displayed replay through C repeats the same finite route for downstream consumers.

The terminal seal E only records the $\mathsf{Real}^{\uparrow}$-facing readback inside the displayed compact convex domain. Since the carrier contains no coordinate for a host fixed point, selected limit, trichotomy, quotient stream equality, or ambient Real completeness, none of those objects can change the finite residual row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerFixedPoint}^{\uparrow}$ packet is generated from a compact convex dyadic domain row, a SimplicialComplexUp subdivision row, DyadicPartitionUp and DyadicCoverUp rows, a Sperner-labelled boundary row, an approximate fixed-simplex witness row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2030.1, Definition 2030.2, Definition 2030.3, and Definition 2030.4 seed the finite Brouwer fixed-point route from dyadic compact-convex data through Sperner labelling, approximate fixed-simplex readback, residual stability, RealUp sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point choice, open-cover compactness, real trichotomy, quotient stream equality, selected limits, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, boundary labelling, Sperner handoff, approximate fixed-point readback, RealUp seal nonescape, transport/replay/provenance stability, and non-escape for BEDC.Derived.BrouwerFixedPoin

Chapter 2031

A Concrete Naming Certificate for $\mathsf{FiniteChoice}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on finite choice, the $\mathsf{FiniteChoice}^{\uparrow}$ packet records finite index selection over displayed $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, and $\mathsf{Bool}^{\uparrow}$ rows and over the L10 $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ windows. It sits beside the countable-choice refusal surface of chapter 1872, the finite stream-window interface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion arguments a visible finite selector boundary while refusing countable-choice selectors.

Definition 2031.1 (FiniteChoice carrier)**.** *A* $\mathsf{FiniteChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, O, B, L, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-list row. It records a displayed* $\mathsf{List}^{\uparrow}$ *of inspected indices together with the order in which the packet reads them.*
2. *O is the* $\mathsf{Option}^{\uparrow}$ *decidability row, carrying for each displayed index either a selected row or an explicit absence token.*
3. *B is the* $\mathsf{Bool}^{\uparrow}$ *decisional ledger attached to O, recording the boolean case used for each finite option inspection.*
4. *L is the selected-row ledger. It contains only the rows selected from the displayed finite list and records no selector outside I.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened for the selected finite indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows in W.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after I, O, B, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the index list, option and boolean rows, selected ledger, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the finite index-list row through $O, B, L, W, R, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteChoice}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Option}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for countable choice, selected infinite branches, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FiniteChoiceUp.FiniteChoiceTasteGate single carrier alignment`

Theorem 2031.2 (FiniteChoice NameCert obligations)**.** *For every accepted* $\mathsf{FiniteChoice}^{\uparrow}$ *carrier*

$$F = (I, O, B, L, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteChoice}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed index-list row, option decidability row, boolean decision ledger, selected-row ledger, StreamName window row, RegSeqRat readback row, Real seal row, transport, replay, provenance, and local naming;*
2. *finite index coverage, requiring every selected row to appear in the displayed* $\mathsf{List}^{\uparrow}$ *row I;*
3. *option and boolean case exactness, requiring O and B to account for each selected or absent finite case before the selected-row ledger L is read;*
4. *window selection, requiring each stream window in W to be opened only for an index accepted by the finite selected-row ledger;*
5. *rational readback, requiring R to consume the displayed windows in W rather than a detached stream quotient or selected limit;*
6. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring S to remain downstream of I, O, B, L, W, R and to add no countable selector, infinite branch, quotient stream equality, trichotomy, ambient completeness principle, Lean verification, or bridge data;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local finite-choice obligation is stored outside these rows.

Proof. Project the carrier of Definition 2031.1. The visible choice data are exactly the finite index list I, the option row O, the boolean decision ledger B, and the selected-row ledger L.

Finite index coverage follows by case analysis over the displayed $\mathsf{List}^{\uparrow}$ row. In the empty-list case, no selected row can be exported. In the cons case, the head index is inspected by O and B, and the tail is handled by the same finite projection.

Option and boolean exactness is local to O and B. The present case contributes its selected row to L, while the absent case contributes only the explicit absence token. The mixed cases are exhausted because the boolean ledger records one displayed decision for each option row.

The analytic handoff then follows the selected rows. Each accepted selected row opens a finite $\mathsf{StreamName}^{\uparrow}$ window in W, and every regular rational readback in R consumes one of those windows.

Finally, S is terminal and H, C, P, N only transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for the excluded objects, the listed rows exhaust the local $\mathsf{NameCert}$ surface. □

Theorem 2031.3 (FiniteChoice window selection boundary)**.** *Let $F = (I, O, B, L, W, R, S, H, C, P, N)$ be an accepted* $\mathsf{FiniteChoice}^{\uparrow}$ *carrier. Every* $\mathsf{StreamName}^{\uparrow}$ *window exported by F is selected by the finite route*

$$I \longmapsto O \longmapsto B \longmapsto L \longmapsto W.$$

Consequently a window consumer may inspect selected finite windows and their option/boolean decisions, but it receives no countable selector, selected infinite branch, detached quotient stream, or ambient real-completeness witness.

Proof. Begin with an exported stream window. By carrier projection, the only window coordinate is W.

The row W is not free-standing. It is downstream of the finite index-list row I, the option row O, the boolean ledger B, and the selected-row ledger L.

Now split by the displayed option cases. A present option contributes a selected finite row to L and therefore may open the corresponding window in W. An absent option contributes no selected row, so no window can be exported from that index.

The rows H, C, P, N preserve and name this finite route without adding a second selector. Thus every exported window is selected by the displayed finite route, and all countable or completion-level choice data remain outside the packet. □

Theorem 2031.4 (FiniteChoice Real-completion handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{FiniteChoice}^{\uparrow}$ *carrier*

$$F = (I, O, B, L, W, R, S, H, C, P, N)$$

must read the finite route

$$\mathsf{List}^{\uparrow} \longmapsto \mathsf{Option}^{\uparrow} \longmapsto \mathsf{Bool}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by I, O, B, L, W, R, S. The handoff permits finite selection for the displayed windows and readbacks only; it does not supply countable choice, selected infinite branches, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier and read the Real-facing coordinate. The only Real seal row is S, and it is exposed only after the finite index, option, boolean, selected-row, stream-window, and regular-readback rows have been displayed.

Apply the window boundary of Definition 2031.3. Every window consumed by the Real or Completion route has already passed through the finite $\mathsf{List}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, and selected-ledger rows.

The readback row R then supplies the regular rational data for those windows, and the terminal seal S records the $\mathsf{Real}^{\uparrow}$ handoff. This route is finite because its source is the displayed list I and its case analysis is exhausted by the option and boolean rows.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate is present, a Real or Completion consumer receives finite selection data only through I, O, B, L, W, R, S and receives none of the stronger choice or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteChoice}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2031.1, Definition 2031.2, Definition 2031.3, and Definition 2031.4 seed the finite selection boundary used by L10 stream, regular-rational, and real-name windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected infinite branches, quotient stream equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite index coverage, option and boolean exactness, window selection, RegSeqRat readback, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteChoiceUp.

Constructive story: A FiniteChoice$^{\uparrow}$ packet is generated from one finite ListUp index row, OptionUp decidability rows, BoolUp decision rows, a selected-row ledger, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2032

A Concrete Naming Certificate for $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$

Following Bishop–Bridges constructive analysis, uniform continuity carries Cauchy source names to Cauchy target names by one displayed modulus schedule, not by ambient function equality or quotient completion. The $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$ packet records that standard theorem as finite NameCert data over the Cauchy-name theorem surface of chapter 1642, the uniform modulus boundary of chapter 391, the finite source-window interface of chapter 126, the source regular-rational route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It gives later Real and Completion chapters a finite route for transporting Cauchy convergence through a uniform modulus before any quotient stream, selected limit, countable-choice schedule, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 2032.1 (Cauchy uniform-continuity principle carrier)**.** *A* $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, U, M, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular Cauchy source data inspected by the theorem route.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the source name, the modulus request, and the target tail handoff.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested target precision, the uniform source budget, and the comparison arithmetic used by the handoff.*
4. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *row supplying the uniform modulus used for all source windows in the packet.*
5. *M is the* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *row, or the displayed modulus-compatibility row, witnessing that the uniform modulus in U is read as the Cauchy-continuity modulus required by the source Cauchy route.*
6. *T is the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *target handoff obtained after S, W, D, U, M have been replayed.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the target regular Cauchy tail row T is displayed.*
8. *H records componentwise* hsame *transport for S, W, D, U, M, T, E and for the structural replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a target tolerance request through D, then the uniform row U, then the source row S through windows W, then the compatibility row M, then the target tail estimate T, and finally the seal row E.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$, $\mathsf{CauchyContinuityPrinciple}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, U, M, T, E, H, C, P, N$ and the displayed replay order. It has no coordinate for host function equality, quotient completion, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyUniformContinuityPrincipleUp.CauchyUniformContinuityPrincipleTasteGate`
`single carrier alignment`

Theorem 2032.2 (Cauchy uniform-continuity principle NameCert obligations)**.** *For every accepted* $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$ *carrier $Q = (S, W, D, U, M, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source regular Cauchy row, finite source windows, dyadic tolerance ledger, uniform-continuity row, modulus-compatibility row, target regularity handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *uniform modulus exactness, requiring every target tolerance to obtain its source-window budget only through D and the uniform row U;*
3. *source-window stability, requiring the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row S and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W to be transported componentwise by H and replayed in the same order by C;*
4. *target regularity handoff, requiring the target Cauchy read to pass through the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *row T before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed;*
5. *Real-seal non-escape, requiring every real-facing read to consume the displayed route D, U, S, W, M, T, E;*
6. *transport and replay preservation, requiring H and C to preserve the displayed coordinates and their row order;*
7. *provenance exhaustion, requiring P to name only the sibling chapters and kernel rows listed in Definition 2032.1;*
8. *local naming, requiring N to name exactly the local certificate row for* $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$.

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2032.1. The accepted packet exposes exactly the rows $S, W, D, U, M, T, E, H, C, P, N$.

Uniform modulus exactness is exhausted by D and U. A target tolerance first reads the dyadic ledger, then the uniform-continuity row, and only then obtains the source-window budget used on S through W.

Source-window stability follows by inspecting H and C. The transport row moves S and W componentwise, while the replay row keeps the same order in every accepted packet.

For the target handoff, read M and T. The row M records the Cauchy-modulus or compatibility surface needed to use the uniform modulus on the Cauchy route, and T records the regular Cauchy target tail estimate produced from the replayed source data.

The Real seal is downstream of that handoff. Since E appears only after D, U, S, W, M, T in the continuation route, every real-facing read must pass through the displayed finite chain.

Finally inspect P and N. The provenance row names only the listed sibling chapters and kernel rows, and the local naming row names only this certificate. Because the carrier inventory contains no further coordinate, the eight obligations exhaust the NameCert surface. □

Theorem 2032.3 (Cauchy uniform-continuity principle finite handoff)**.** *Let $Q = (S, W, D, U, M, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$ *carrier. Every target* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by the packet factors through the finite route*

$$D, U, S, W, M, T.$$

Consequently the target Cauchy name is obtained from the source regular-rational row, finite windows, dyadic tolerance ledger, uniform modulus, compatibility row, and target tail estimate before the $\mathsf{Real}^{\uparrow}$ *seal is available.*

Proof. Start with the tolerance request. The dyadic ledger D supplies the requested target precision and the comparison budget used to choose the source window.

Next read the uniform modulus. The row U is the only uniform-continuity coordinate in the carrier, so the source-window budget for every source observation comes from U through D.

The source name is then inspected through S and W. The source row supplies the regular Cauchy data, and the window row supplies the finite observations admitted by the packet.

Use the compatibility row M to enter the Cauchy-name theorem surface. It records that the uniform modulus has the Cauchy-continuity form required for the source-to-target handoff.

The target read is forced through T. This row is the only target regularity handoff in the carrier, and the seal row E is downstream of it. Hence every exported target $\mathsf{RegSeqRat}^{\uparrow}$ read factors through the stated finite route. □

Theorem 2032.4 (Cauchy uniform-continuity principle Real boundary)**.** *For every accepted* CauchyUniformContinu *carrier $Q = (S, W, D, U, M, T, E, H, C, P, N)$, every* $\mathsf{Real}^{\uparrow}$ *consumer of the seal row E first reads the displayed sequence of sibling surfaces*

$\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$, RegularCauchyTail

The consumer receives no host function equality, quotient completion, countable-choice selector, selected limit, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge-checking row.*

Proof. Read the seal coordinate in Definition 2032.1. The row E is listed only as the terminal $\mathsf{Real}^{\uparrow}$ seal reached after the target regular Cauchy tail row T is displayed.

Apply Definition 2032.3. The target regular Cauchy read has already passed through D, U, S, W, M, T before a real-facing consumer can inspect the seal.

Replay the structural rows. The continuation row C preserves the dependency order, the transport row H preserves the same finite coordinates, and the provenance and naming rows P, N record no additional source.

Thus every Real consumer receives exactly the displayed finite route through the sibling surfaces. The excluded principles cannot be exported because they are absent from the carrier inventory. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2032.1, Definition 2032.2, Definition 2032.3, and Definition 2032.4 seed the finite route carrying displayed Cauchy source names through a uniform modulus to displayed target Cauchy names before Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient completion, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform modulus exactness, source-window stability, target regularity handoff, Real-seal nonescape, transport and replay preservation, provenance exhaustion, and local naming for BEDC.Derived.CauchyUniformContinuityPrincipleUp.

Constructive story: A $\mathsf{CauchyUniformContinuityPrinciple}^{\uparrow}$ packet is generated from a source RegSeqRat row, finite StreamName windows, a DyadicRatCore tolerance ledger, a UniformContinuity row, a CauchyModulusOfContinuity or modulus-compatibility row, a RegularCauchyTailEstimate target handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2033

A Concrete Naming Certificate for $\mathsf{UniformPartition}^{\uparrow}$

Following Bishop–Bridges/Riemann-Darboux uniform partitions, the $\mathsf{UniformPartition}^{\uparrow}$ packet records a finite equal-mesh subdivision of a displayed real interval. It sits between the $\mathsf{Real}^{\uparrow}$ interval source, the dyadic endpoint and mesh ledgers of chapter 1863 and chapter 358, and the Darboux, Riemann, and regulated-function consumers of chapter 1880, chapter 1526, and chapter 1906. The packet gives those consumers a single finite NameCert surface for equal-width cells without importing open-cover compactness, selected tags, quotient function equality, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2033.1 (Uniform partition carrier)**.** *A* $\mathsf{UniformPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, O, \delta, k, E, B, M, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval source row, naming the left and right real endpoint seals and the direction in which the finite partition is read.*
2. *O is the dyadic origin row from which the equal mesh is generated.*
3. *δ is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh-width row.*
4. *k is a unary cell-count row, recording the finite number of equal-width cells without using an ambient natural-number search principle.*
5. *E is the equal-width cell ledger. Its jth row records the dyadic endpoints $O+j\delta$ and $O+(j+1)\delta$, the adjacency check between successive cells, and the local width equality to δ.*
6. *B is the endpoint coverage row, asserting that the first cell begins at the interval source and the last cell reaches the displayed terminal endpoint through the same dyadic origin, width, and count rows.*
7. *M is the* $\mathsf{DyadicPartition}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$ *handoff row, exposing the ordered finite partition and its mesh bound to downstream dyadic consumers.*
8. *T is the Darboux, Riemann, and* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer-boundary row. It records that those consumers may read the interval, equal cells, endpoint coverage, and dyadic handoff, but not tags, integrand quotients, measure completions, or open-cover compactness.*

9. *H records componentwise* hsame *transport for the interval source, origin, positive width, count, equal-cell ledger, endpoint coverage, dyadic handoff, consumer boundary, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$I, O, \delta, k \longmapsto E \longmapsto B \longmapsto M \longmapsto T.$$

11. *P is the* Pkg *provenance over* $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, O, \delta, k, E, B, M, T, H, C, P, N$. It has no coordinate for open-cover compactness, selected tags, quotient function equality, measure completion, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.UniformPartitionUp.UniformPartitionTasteGate single carrier alignment`

Theorem 2033.2 (Uniform partition NameCert obligations)**.** *For every accepted* $\mathsf{UniformPartition}^{\uparrow}$ *carrier*

$$U = (I, O, \delta, k, E, B, M, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformPartition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval source, dyadic origin, positive mesh width, unary cell count, equal-width cell ledger, endpoint coverage, dyadic handoff, consumer boundary, transport, replay, provenance, and local naming rows;*

2. *equal-mesh exactness, requiring every cell listed in E to have width δ and to be adjacent to its successor through displayed dyadic arithmetic;*

3. *endpoint coverage, requiring B to identify the first and last cell boundaries with the interval source row I through O, δ, k, E;*

4. *dyadic handoff, requiring M to expose only the ordered* $\mathsf{DyadicPartition}^{\uparrow}$ *row and the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh-bound row computed from the same ledger E;*

5. *consumer boundary, requiring T to route Darboux, Riemann, and regulated-function consumers through I, E, B, M before any integration-facing read is available;*

6. *structural stability, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows.*

No local certificate obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2033.1. The accepted packet lists the interval row, dyadic origin, positive width, unary count, cell ledger, endpoint coverage, dyadic handoff, consumer boundary, and the four structural rows.

Equal-mesh exactness is the projection to E. Each finite cell row contains its two dyadic endpoints, the displayed adjacency relation to neighboring cells, and the equality of its width with δ.

Endpoint coverage is the projection to B. The first and last boundaries are read through the same origin, width, count, and ledger, so coverage does not appeal to compactness or to an external subdivision theorem.

The dyadic and integration-facing exports are terminal. Row M exposes the ordered partition and mesh bound, while T records the permitted Darboux, Riemann, and regulated-function consumers. Rows H, C, P, N transport, replay, source, and name only this finite surface, so the excluded host principles are absent from the obligation set. □

Theorem 2033.3 (Uniform partition dyadic mesh handoff)**.** *Every accepted* $\mathsf{UniformPartition}^{\uparrow}$ *packet hands off to* $\mathsf{DyadicPartition}^{\uparrow}$ *and* $\mathsf{DyadicMesh}^{\uparrow}$ *consumers only through the finite route*

$$I, O, \delta, k \longmapsto E \longmapsto B \longmapsto M.$$

The row M therefore exposes an ordered dyadic partition, a positive mesh width, endpoint coverage, and the mesh bound as a single visible object; it does not export selected tags, open-cover compactness, quotient equality of functions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only dyadic consumer coordinate is M, and the displayed replay row C places M after the interval source, origin, width, count, equal-cell ledger, and endpoint coverage rows.

The ordered partition read is supplied by E and B. Each cell row in E gives its dyadic endpoints and equal width, while B fixes the first and last endpoints against the real interval source.

The mesh read is therefore the same finite ledger viewed through $\mathsf{DyadicMesh}^{\uparrow}$. Since no tag, compactness, quotient, measure, or completeness coordinate is present in the carrier, the handoff cannot expose those host objects. □

Theorem 2033.4 (Uniform partition integration consumer boundary)**.** *A* $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *or* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformPartition}^{\uparrow}$ *carrier receives only the interval row I, equal-cell ledger E, endpoint coverage row B, dyadic handoff M, consumer-boundary row T, and structural rows H, C, P, N. The consumer may use the finite equal-mesh partition as a mesh surface, but the packet does not supply tags, lower or upper sums, integrand equality, a measure-completion object, open-cover compactness, or ambient real completeness.*

Proof. Begin with the consumer-boundary row T. By Definition 2033.1, T is downstream of the displayed route through I, O, δ, k, E, B, M.

A Darboux or Riemann consumer may read the equal-width cells and endpoint coverage through E and B, then use M as the dyadic partition or mesh surface. The carrier does not contain lower sums, upper sums, selected tags, or an integrand row, so those data must be supplied by the consuming chapter.

A regulated-function consumer receives the same finite partition boundary. Rows H, C, P, N preserve, replay, source, and name the route, but they do not create quotient function equality, measure completion, compactness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformPartition}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ interval source row, dyadic origin row, positive dyadic mesh-width row, unary cell-count row, equal-width cell ledger, endpoint coverage row, $\mathsf{DyadicPartition}^{\uparrow}$ and $\mathsf{DyadicMesh}^{\uparrow}$ handoff row, Darboux, Riemann, and regulated-function consumer boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2033.1, Definition 2033.2, Definition 2033.3, and Definition 2033.4 seed the finite equal-mesh partition certificate over the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected tags, quotient function equality, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, equal-mesh exactness, endpoint coverage, dyadic handoff, integration consumer boundary, transport, replay, provenance, and local NameCert rows for BEDC.Derived.UniformPartitionUp.

Chapter 2034

A Concrete Naming Certificate for $\mathsf{GronwallInequality}^{\uparrow}$

Following Gronwall and the Bishop–Bridges constructive ODE route, the $\mathsf{GronwallInequality}^{\uparrow}$ packet records the finite majorant certificate by which a nonnegative error envelope is bounded before a Picard local-flow comparison is read. It consumes the local existence route of chapter 1888, the local-flow endpoint surface of chapter 1899, the Lipschitz ledger of chapter 410, the complete-metric convergence boundary of chapter 128, the sealed real-name row of chapter 38, the regular rational readback of chapter 301, and the dyadic arithmetic core of chapter 291. The packet names only a displayed comparison route over finite interval, Lipschitz, envelope, dyadic exponential majorant, Picard-flow comparison, transport, replay, provenance, and local naming rows.

Definition 2034.1 (Gronwall inequality carrier)**.** *A* $\mathsf{GronwallInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, L, E, D, M, F, R, H, C, G, N)$$

with the following visible rows:

1. *I is the interval row, recording the displayed time window, its endpoint order, and the local* $\mathsf{Real}^{\uparrow}$ *names through which the comparison is inspected.*
2. *L is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row naming the Lipschitz constant used by the ODE or local-flow comparison on the same interval.*
3. *E is the nonnegative envelope row, storing the displayed error or separation function, its initial bound, and the monotonicity reads used by downstream comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-majorant row, containing the dyadic coefficients, finite truncation budget, and upper rational enclosure for the Gronwall factor.*
5. *M is the majorant comparison row, asserting that every inspected envelope read in E is bounded by the dyadic exponential row D after reading the Lipschitz row L and interval row I.*
6. *F is the Picard-flow comparison row, supplied by* $\mathsf{PicardLindelof}^{\uparrow}$ *or* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$ *when the carrier is consumed by local ODE stability.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations of the interval, Lipschitz constant, envelope samples, and dyadic majorant.*

8. *H records componentwise* hsame *transport for the interval, Lipschitz ledger, envelope, dyadic majorant, comparison, Picard-flow handoff, rational readback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through*

$$I, L, E, D, M, F, R \longmapsto \mathsf{NameCert}_{\mathsf{GronwallInequality}^{\uparrow}}.$$

10. *G is the* Pkg *provenance row over* $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GronwallInequality}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient real-completeness principle, a host ODE solver, a choice-selected solution curve, a host equality of functions, or a global continuation theorem.

Theorem 2034.2 (Gronwall inequality NameCert obligations)**.** *For an accepted* $\mathsf{GronwallInequality}^{\uparrow}$ *carrier*

$$P = (I, L, E, D, M, F, R, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{GronwallInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, dyadic majorant admission for D, envelope monotonicity for E, comparison stability for M, Picard handoff for F, rational readback for R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through G, and local naming through N. No obligation is discharged by a host ODE solver, ambient real completeness, host function equality, or a choice-selected solution curve.*

Lean4 ✓ `BEDC.Derived.GronwallInequalityUp.GronwallInequalityTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 2034.1. The accepted packet displays the interval, Lipschitz ledger, nonnegative envelope, dyadic majorant, comparison, Picard-flow handoff, rational readback, transport, replay, provenance, and local name.

The source rows I, L, E, R determine the comparison problem. The interval row fixes the finite time window, the Lipschitz row supplies the displayed constant, the envelope row supplies the nonnegative quantity being bounded, and the regular-rational row supplies the observations through which the comparison is inspected.

The dyadic majorant obligation is local to D. It records the finite exponential enclosure used by the comparison, so the certificate never needs an unbounded real exponential object or an ambient completeness argument.

The comparison row M then binds the envelope to the majorant after reading I, L, E, D, R. Its stability is componentwise: transporting the interval, Lipschitz, envelope, dyadic, and readback rows transports the displayed inequality route.

Finally F is the only ODE-facing exit. It may hand the bound to $\mathsf{PicardLindelof}^{\uparrow}$ or $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$, but only through the displayed rows. The rows H, C, G, N transport, replay, source, and name this same finite route, so the certificate cannot export a host ODE solver, a choice-selected solution curve, host function equality, or global continuation data. □

Theorem 2034.3 (Gronwall inequality Picard handoff)**.** *Every accepted* $\mathsf{GronwallInequality}^{\uparrow}$ *carrier factors any Picard–Lindelof local-flow stability read through the displayed route*

$$\mathsf{LipschitzMap}^{\uparrow} \longmapsto \mathsf{GronwallInequality}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The read first opens the Lipschitz constant, then the nonnegative envelope, then the dyadic exponential majorant, then the regular-rational observation rows, and only then seals the comparison as a $\mathsf{Real}^{\uparrow}$ *bound usable by* $\mathsf{PicardLindelof}^{\uparrow}$ *or* $\mathsf{PicardLindelofLocalFlow}^{\uparrow}$.

Proof. Begin with the accepted carrier P. The only Lipschitz data available to a stability consumer is the row L, and the only envelope data is the row E on the interval I.

Read the majorant next. The row D supplies the finite dyadic exponential enclosure, while R supplies the regular rational observations needed to compare envelope samples with the majorant.

The comparison row M combines these displayed rows into the Gronwall bound. Since M reads I, L, E, D, R before producing the comparison, any resulting real bound is sealed through the same $\mathsf{Real}^{\uparrow}$ row rather than through ambient completeness.

The Picard-facing row F can now consume the bound. It receives only the displayed Lipschitz, envelope, dyadic majorant, regular-rational, and real-seal rows together with transport, replay, provenance, and local naming. Thus every Picard–Lindelof local-flow stability read factors through the stated route and cannot bypass it by invoking a host solver, host function equality, or a choice-selected solution curve. □

Theorem 2034.4 (Gronwall inequality finite integral bound)**.** *For every accepted* $\mathsf{GronwallInequality}^{\uparrow}$ *carrier*

$$P = (I, L, E, D, M, F, R, H, C, G, N),$$

the comparison row M bounds each inspected envelope row in E by the dyadic exponential-majorant row D only after reading the interval row I, the Lipschitz row L, and the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The Picard-facing row F may consume that bound for local-flow stability, but it receives no host ODE solver, global continuation, selected solution curve, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier by Definition 2034.1. The rows I, L, E, D, M, F, R are the only analytic coordinates available to a stability consumer, and the structural rows H, C, G, N only transport, replay, source, and locally name those coordinates.

Apply the obligation surface of Definition 2034.2 to the rows D, E, M, R. The dyadic majorant is admitted by D, the envelope reads are admitted by E, the rational observations are admitted by R, and the comparison stability row M binds them only after the interval and Lipschitz rows have also been opened.

Now route the resulting comparison through Definition 2034.3. The Picard-facing row F receives the displayed finite bound through the same route, so it cannot import a host ODE solver, global continuation, selected solution curve, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{GronwallInequality}^{\uparrow}$ stability read factors through one interval row, one LipschitzMap constant row, one nonnegative envelope row, one DyadicRatCore exponential-majorant row, one comparison row, one Picard-flow handoff, one RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: picardlindelof, picardlindeloflocalflow, lipschitzmap, completemetric, real, regseqrat, dyadicratcore: $\mathsf{GronwallInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the interval, Lipschitz constant, nonnegative envelope, dyadic exponential majorant, Picard-flow comparison, rational readback, and the transport/replay/provenance rows binding them into a finite stability certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{GronwallInequality}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GronwallInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2034.1, Definition 2034.2, and Definition 2034.3 seed the constructive ODE stability route over an interval row, Lipschitz ledger, nonnegative envelope, dyadic exponential majorant, Picard-flow comparison, rational readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{GronwallInequality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ambient Real completeness, a host ODE solver, host function equality, choice-selected solution curves, global continuation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic majorant admission, envelope monotonicity, comparison stability, Picard handoff, rational readback, transport, replay, provenance, and local naming for BEDC.Derived.GronwallInequalityUp.

Constructive story: A GronwallInequality$^{\uparrow}$ packet is generated from one interval row, one LipschitzMap constant row, one nonnegative envelope row, one DyadicRatCore exponential-majorant row, one comparison row, one Picard-flow handoff, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2035

A Concrete Naming Certificate for $\mathsf{DoublingMetric}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis and Kelley's finite-cover regularity language, the $\mathsf{DoublingMetric}^{\uparrow}$ packet records the standard bounded-overlap cover control used before total-bounded, compact-metric, and dimension-facing consumers read a metric space at smaller scales. It sits over the displayed metric source of chapter 57, hands finite center ledgers to chapter 1767 and chapter 127, and supplies scale-local cover data usable by chapter 129. The packet names only finite ball-cover rows, positive radius rows, scale-halving ledgers, transport, replay, provenance, and local naming; it does not import open-cover compactness, a choice-selected cover family, or an ambient semantic doubling theorem.

Definition 2035.1 (Doubling metric carrier)**.** *A* $\mathsf{DoublingMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, \rho, B, K, E, T, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance rows, ball membership reads, and metric-law ledger through which every cover request is inspected.*
2. *ρ is the positive radius ledger. Each row stores a positive rational or dyadic radius request together with the displayed half-radius request used by the next scale.*
3. *B is the ball request row. It names the inspected center, the radius row from ρ, and the parent metric ball whose coverage is being certified.*
4. *K is the center-bundle row, a closed finite* $\mathsf{ProbeBundle}$ *of candidate centers admitted by the same metric source M.*
5. *E is the ball-cover coverage ledger. For every displayed point row admitted by B, it records an admitted center in K and the half-radius distance read witnessing membership in the corresponding smaller ball.*
6. *T is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *handoff row, exposing only the finite center bundle, tolerance read, and coverage witnesses computed from M, ρ, B, K, E.*
7. *S is the scale-halving ledger. It records that every consumer step from radius r to radius $r/2$ reuses the same metric source, the corresponding positive-radius row, and a finite center bundle of the displayed bounded size.*

8. *H records componentwise* hsame *transport for the metric source, positive radius ledger, ball request, center bundle, coverage ledger, finite-net handoff, scale-halving ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, \rho, B \longmapsto K \longmapsto E \longmapsto T \longmapsto S.$$

10. *P is the* Pkg *provenance over* $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DoublingMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, \rho, B, K, E, T, S, H, C, P, N$. It has no coordinate for an open cover, a selected subcover, a global compactness principle, a choice-selected family of centers, ambient real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.DoublingMetricUp.DoublingMetricTasteGate single carrier alignment`

Theorem 2035.2 (Doubling metric NameCert obligations)**.** *For every accepted* $\mathsf{DoublingMetric}^{\uparrow}$ *carrier*

$$D = (M, \rho, B, K, E, T, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DoublingMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, positive radius ledger, ball request, center bundle, coverage ledger, finite-net handoff, scale-halving ledger, transport, replay, provenance, and local naming rows;*

2. *ball-cover coverage, requiring each displayed point in the parent ball B to be covered by one of the centers in K through the half-radius metric read recorded in E;*

3. *scale monotonicity, requiring the ledger S to relate every radius row to its half-radius row without changing the source* $\mathsf{Met}^{\uparrow}$ *carrier or escaping the displayed positive-radius ledger;*

4. *finite-net handoff, requiring T to expose the center bundle and coverage rows only as* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *data over the same metric source;*

5. *transport stability, requiring H and C to preserve and replay the metric, radius, bundle, coverage, handoff, and scale rows componentwise;*

6. *non-escape, requiring every* $\mathsf{CompactMetric}^{\uparrow}$ *or covering-dimension consumer to read only $M, \rho, B, K, E, T, S, H$, and no open-cover compactness or choice-selected cover family.*

No local certificate obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2035.1. The accepted packet displays the metric source, positive radius ledger, ball request, finite center bundle, coverage ledger, finite-net handoff, scale-halving ledger, and the four structural rows.

Ball-cover coverage is the projection to E. Each inspected point in the parent ball named by B is paired with a center from K, and the metric distance row from M witnesses that the point lies in the corresponding half-radius ball.

Scale monotonicity is local to ρ and S. The ledger records the passage from r to $r/2$ as a displayed positive-radius step over the same source metric row, so a consumer cannot replace the metric carrier, alter the ball source, or introduce a hidden real-number choice.

The finite-net handoff is the projection to T. It repackages the same center bundle and coverage rows as $\mathsf{FiniteEpsilonNet}^{\uparrow}$ and $\mathsf{TotallyBounded}^{\uparrow}$ data, which a $\mathsf{CompactMetric}^{\uparrow}$ consumer may later read only through the visible finite route.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no row for open covers, selected subcovers, global compactness, ambient real completeness, or choice-selected center families, the listed non-escape boundary is part of the local certificate surface. □

Theorem 2035.3 (Doubling metric finite-net handoff)**.** *Every* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{DoublingMetric}^{\uparrow}$ *packet*

$$D = (M, \rho, B, K, E, T, S, H, C, P, N)$$

must read the same route $M, \rho, B, K, E, T, S, H, C, P, N$*. The row* T *exposes the finite center bundle and half-radius coverage rows as finite-net and total-bounded data, while* S *preserves the scale-halving ledger over the same* $\mathsf{Met}^{\uparrow}$ *source. The handoff exports no open-cover compactness principle, selected subcover, or choice-selected family of centers.*

Proof. Project the accepted carrier. The metric source is M, the positive-radius ledger is ρ, the parent ball request is B, and the center bundle and coverage witnesses are K and E.

Apply Definition 2035.2. The ball-cover coverage row requires each displayed point in B to be covered by a center in K through the half-radius metric read in E, and the scale monotonicity row keeps the passage from r to $r/2$ inside ρ and S.

The finite-net handoff is exactly the projection to T together with the preserved scale ledger S. Thus $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$ consumers can read finite centers and coverage only through $M, \rho, B, K, E, T, S, H, C, P, N$, with no row available for open-cover compactness or selected center families. □

Falsifiable prediction: Every $\mathsf{DoublingMetric}^{\uparrow}$ cover read factors through one MetricUp source row, one positive radius ledger, one parent ball request, one finite ProbeBundle of centers, one half-radius coverage ledger, one FiniteEpsilonNet and TotallyBounded handoff, one scale-halving ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, finite-epsilon-net, totallybounded, compactmetric: $\mathsf{DoublingMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the parent-ball request, bounded center-bundle coverage, scale-halving ledger, finite-net handoff, and transport/replay/provenance rows binding them into a finite metric regularity certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DoublingMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoublingMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2035.1 and Definition 2035.2 seed the finite scale-halving cover certificate over a displayed $\mathsf{Met}^{\uparrow}$ source and its $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$ consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoublingMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, global compactness, choice-selected center families, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ball-cover coverage, scale monotonicity, finite-net handoff, transport stability, non-escape, provenance, and local NameCert rows for BEDC.Derived.DoublingMetricUp.

Constructive story: A $\mathsf{DoublingMetric}^{\uparrow}$ packet is generated from a MetricUp source row, positive radius rows, a parent ball request, a finite ProbeBundle of centers, a half-radius coverage ledger, a FiniteEpsilonNetUp and TotallyBoundedUp handoff, a scale-halving ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2036

A Concrete Naming Certificate for $\mathsf{CauchyCompletionIsometry}^{\uparrow}$

Bishop–Bridges and Kelley-style completion language treats the unit map from a metric source into its Cauchy completion as distance preserving before density or extension consumers use the completed object. The $\mathsf{CauchyCompletionIsometry}^{\uparrow}$ packet records that boundary as finite BEDC data: it reads the source metric route of chapter 1638, the unit route of chapter 1527, the distance-preserving interface of chapter 439, the dense-image handoff of chapter 1560, and the terminal real-name seal of chapter 38. It names the unit distance-preservation surface for completion consumers without identifying quotient completion classes, choosing representatives, asserting host function equality, invoking countable choice, using open-cover compactness, assuming ambient completeness, or proving completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2036.1 (CauchyCompletionIsometry carrier)**.** *A* $\mathsf{CauchyCompletionIsometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (X, U, D_X, D_{\widehat{X}}, R, Q, I, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row whose displayed endpoints and finite distance requests are being admitted into the completion route.*
2. *U is the accepted* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row that sends the displayed source endpoints into the completion-facing surface.*
3. *D_X is the source-distance witness row for the displayed source endpoints, read through the source metric and real-metric comparison data.*
4. *$D_{\widehat{X}}$ is the completion-distance witness row for the corresponding unit images.*
5. *R is the* $\mathsf{RealMetric}^{\uparrow}$ *comparison row witnessing that the displayed source-distance read and completion-distance read name the same* $\mathsf{Real}^{\uparrow}$ *distance observation.*
6. *Q is the accepted* $\mathsf{CompletionDensity}^{\uparrow}$ *row available after the unit image has been admitted, so later density consumers can cite the same source route.*
7. *I is the accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *row that packages the source metric row, completion-facing target row, unit graph, the two distance witnesses, and the real-distance comparison.*

8. *H records componentwise* hsame *transport for the source metric row, unit row, both distance witnesses, real-metric comparison, density row, isometric-embedding row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a source distance request through the unit images, source and completion distance witnesses, the real-metric comparison, the isometric-embedding row, and the density handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionIsometry}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{CompletionDensity}^{\uparrow}$, $\mathsf{IsometricEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionIsometry}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host function equality, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionIsometryUp.CauchyCompletionIsometryTasteGate single carrier alignment`

Theorem 2036.2 (CauchyCompletionIsometry NameCert obligations)**.** *For an accepted* CauchyCompletionIsometry *carrier $J = (X, U, D_X, D_{\widehat{X}}, R, Q, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionIsometry}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed source metric row, completion unit row, source-distance witness, completion-distance witness, real-metric comparison, completion-density row, isometric-embedding row, transports, replay, provenance, and local naming row;*

2. *unit endpoint routing, requiring every source endpoint used by D_X to enter the completion-facing target only through the row U;*

3. *distance-pair exposure, requiring the source distance in D_X and the completion distance in $D_{\widehat{X}}$ to be displayed before the comparison row R is read;*

4. *isometry packaging, requiring the accepted row I to bind the same $X, U, D_X, D_{\widehat{X}}, R$ route as the local distance-preservation certificate;*

5. *density handoff, requiring the row Q to be downstream of the same unit images and not an independent dense subset;*

6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed route.*

No NameCert obligation is stored outside those rows, and none exports quotient completion equality, selected representatives, host function equality, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 2036.1. The tuple displays the source metric row, completion unit, two distance witnesses, real-metric comparison, density handoff, isometric-embedding row, transport, replay, provenance, and local name.

Unit endpoint routing follows from the placement of U before either completion-facing distance witness is consumed. A source endpoint can reach the target side of the packet only as the unit image recorded by the accepted $\mathsf{CauchyCompletionUnit}^{\uparrow}$ row.

Distance-pair exposure and isometry packaging are then finite row checks. The comparison row R reads only the displayed source and completion distances, while I packages that same comparison as an $\mathsf{IsometricEmbedding}^{\uparrow}$ route.

The density handoff is downstream of the same unit images. Since Q is a row of the carrier and is transported only componentwise by H and replayed only by C, it cannot supply a detached dense subset or a second completion object.

Finally P and N source and name exactly the displayed packet. The excluded quotient, representative, host-equality, choice, compactness, ambient-completeness, and real-completeness principles have no carrier coordinate from which a consumer could read them. □

Theorem 2036.3 (CauchyCompletionIsometry unit distance preservation)**.** *Let $J = (X, U, D_X, D_{\widehat{X}}, R, Q, I, H, C,$ be an accepted* $\mathsf{CauchyCompletionIsometry}^{\uparrow}$ *packet. Every finite source-distance read in D_X is routed through the unit row U to the completion-distance read in $D_{\widehat{X}}$, and the equality of the two displayed distance observations is read only through the* $\mathsf{RealMetric}^{\uparrow}$ *comparison row R and the accepted* $\mathsf{IsometricEmbedding}^{\uparrow}$ *row I.*

Thus the unit map is distance preserving at the finite NameCert surface:

$$X \longmapsto U \longmapsto (D_X, D_{\widehat{X}}) \longmapsto R \longmapsto I.$$

The route is not quotient completion equality, not selected representative equality, not host function equality, and not an appeal to countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with a source-distance request. The source metric row X and source-distance witness D_X display the two source endpoints and their finite distance observation.

Apply the completion-unit row U. It supplies the only completion-facing images of those endpoints, so the target distance witness $D_{\widehat{X}}$ is attached to the unit images rather than to a chosen quotient representative or host function.

The row R compares the two displayed distance observations inside the real-metric surface. It reads the same source and completion distance witnesses and therefore gives the finite equality of distance observations needed by an isometry consumer.

Finally the accepted row I packages the metric rows, unit graph, distance witnesses, and real-distance comparison as the $\mathsf{IsometricEmbedding}^{\uparrow}$ route. The structural rows H, C, P, N keep that route componentwise, replayable, sourced, and locally named, with no coordinate for any excluded completion principle. □

Theorem 2036.4 (CauchyCompletionIsometry density handoff)**.** *Every completion-density consumer of an accepted* $\mathsf{CauchyCompletionIsometry}^{\uparrow}$ *packet receives the dense source image only after the distance-preserving unit route*

$$X, U, D_X, D_{\widehat{X}}, R, I$$

has been displayed. The $\mathsf{CompletionDensity}^{\uparrow}$ *row Q is therefore a handoff from the same unit images and real-distance comparison, not a separate dense subset, not a quotient-completion selector, and not an ambient completeness theorem.*

Proof. Begin with a density consumer. The carrier exposes the density row Q only as a row of the same packet that already contains $X, U, D_X, D_{\widehat{X}}, R$ and I.

Use the unit distance-preservation theorem. Before Q can be read as a completion-density handoff, the source endpoints have passed through U, their source and completion distances have been displayed, and the real-distance comparison has been packaged by I.

Transport and replay preserve this dependency. The row H moves the source, unit, distance, comparison, isometric, and density rows componentwise, while C replays the same order for downstream consumers.

Thus Q hands off density for the displayed unit image that has already carried its distance-preservation certificate. Since the packet contains no separate dense subset, quotient selector, representative choice, host equality, compactness, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, the density handoff exports only that finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionIsometry}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionIsometry}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2036.1, Definition 2036.2, Definition 2036.3, and Definition 2036.4 seed the finite distance-preserving unit-map certificate from source metric rows into Cauchy completion density consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionIsometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host function equality, countable choice, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, unit endpoint routing, distance-pair exposure, isometry packaging, density handoff, and provenance exhaustion for BEDC.Derived.CauchyCompletionIsometryUp.

Constructive story: A $\mathsf{CauchyCompletionIsometry}^{\uparrow}$ packet is generated from a source metric row, a CauchyCompletionUnit row, source and completion distance witnesses, a RealMetric comparison row, a CompletionDensity row, an IsometricEmbedding row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2037

A Concrete Naming Certificate for $\mathsf{CauchyCompletionLift}^{\uparrow}$

Following Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionLift}^{\uparrow}$ packet records the finite NameCert surface by which a uniformly continuous map out of a displayed finite-source Cauchy route is lifted across Cauchy completion consumers. It sits after the functorial action of chapter 1937, the uniform completion interface of chapter 1892, and the effective completion route of chapter 2026. The chapter names only the carrier, obligation, and universal-route data needed by completion consumers; it does not identify quotient completion classes, choose limit representatives, assert host function equality, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2037.1 (CauchyCompletionLift carrier)**.** *A* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, F, U, E, K, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite-source metric row, including the displayed source observations, Cauchy windows, and admissible tolerance requests before any completed consumer is read.*
2. *T is the target completion-consumer row, exposing the completion-facing object only after its uniform structure and accepted consumer interface have been displayed.*
3. *F is the uniformly continuous source-map row from the finite source route into the target-facing precompletion data.*
4. *U is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *row that records the uniform completion surface used by the lift.*
5. *E is the* $\mathsf{EffectiveCompletion}^{\uparrow}$ *row that supplies the finite Cauchy-realizer, stream-window, regular-rational, dyadic-tolerance, and real-seal route consumed by the lift.*
6. *K is the* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *row that packages the map action on completion-facing data before the universal consumer reads the lifted map.*
7. *R is the universal-route row, recording that every allowed completion consumer reads the lifted map only through the displayed S, T, F, U, E, K chain.*
8. *H records componentwise* hsame *transport for the source row, target row, source-map row, uniform completion row, effective completion row, functoriality row, universal route, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the route*

$$S, T, F \longmapsto U, E \longmapsto K \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}$ $\mathsf{EffectiveCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionLift}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, F, U, E, K, R, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, representative choice, host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionLiftUp.CauchyCompletionLiftTasteGate single carrier alignment`

Theorem 2037.2 (CauchyCompletionLift NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *carrier*

$$L = (S, T, F, U, E, K, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLift}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite-source row, target completion-consumer row, uniformly continuous source-map row, uniform completion row, effective completion row, functoriality row, universal-route row, transport, replay, provenance, and local naming row;*
2. *finite-source visibility, requiring every lifted-map read to expose the source observations, Cauchy windows, and tolerance requests in S before completion-facing data is inspected;*
3. *uniform continuity handoff, requiring the source-map row F to be consumed through the uniform completion row U before any completed-map read is named;*
4. *effective completion exactness, requiring the row E to supply the stream windows, regular rational readback, dyadic tolerance ledger, and real seal used by the lift;*
5. *functorial action, requiring the row K to bind the same S, T, F, U, E route as the completion-facing map action;*
6. *universal consumer routing, requiring R to expose every allowed completion consumer only after the displayed functorial route is available;*
7. *ledger non-escape, requiring H, C, P, N to preserve, replay, source, and name only the listed rows and to export no quotient completion equality, selected limit, representative choice, host equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy completion lift obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2037.1. The accepted packet displays exactly the finite source, target consumer, source-map, uniform completion, effective completion, functoriality, universal-route, and structural rows.

Finite-source visibility is read from S. Since the source observations, Cauchy windows, and tolerance requests occur before any completion-facing row, the lifted map cannot be inspected from an unrecorded source object.

The uniform-continuity handoff is read from F and U. The source map enters the lift only as uniformly continuous finite data, and the completion-facing interface is the uniform completion row rather than a host function on completed objects.

Effective completion exactness is read from E. The lift uses the displayed stream windows, regular rational readback, dyadic tolerance ledger, and real seal already admitted by the effective completion route.

Functorial action and universal consumer routing are read from K and R. The row K packages the same source, target, source-map, uniform completion, and effective completion route, while R makes each allowed consumer read through that package.

Finally H, C, P, N add no analytic coordinate. They transport, replay, source, and name the displayed route, so the excluded quotient, selected-limit, representative-choice, host-equality, countable-choice, and ambient-completeness principles have no row from which a consumer could read them. □

Theorem 2037.3 (CauchyCompletionLift universal route)**.** *Let*

$$L = (S, T, F, U, E, K, R, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionLift}^{\uparrow}$ *carrier. Every completion consumer of the lifted source-map data factors through the finite route*

$$S, T, F \longmapsto U, E \longmapsto K \longmapsto R \longmapsto H, C, P, N.$$

The route is a universal lift surface for uniformly continuous finite-source maps across Cauchy completion consumers. It is not quotient completion equality, not selected-limit data, not representative choice, not host function equality, not countable choice, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Fix a completion consumer of the lifted data. Projection gives the source row S, target row T, and uniformly continuous source-map row F.

The lift then enters the uniform and effective completion rows. The row U supplies the uniform completion interface, while E supplies the finite Cauchy-realizer route through stream windows, regular rational readback, dyadic tolerance data, and the real seal.

The functoriality row K packages the map action on completion-facing data. It reads the same S, T, F, U, E coordinates, so a consumer cannot replace the route by a selected representative or by host function equality.

The universal route row R is downstream of K. Thus every allowed consumer reads the lifted map only after the uniform, effective, and functorial rows have been displayed.

Transport, replay, provenance, and local naming close the route. Since H, C, P, N preserve only the listed rows, the packet exports no quotient completion equality, selected limit, representative choice, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2037.4 (CauchyCompletionLift uniform-modulus handoff)**.** *Let*

$$L = (S, T, F, U, E, K, R, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionLift}^{\uparrow}$ *carrier. Every completion consumer of lifted map data must expose the uniformly continuous source-map row* F *and its modulus-bearing finite data before the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$*,* $\mathsf{EffectiveCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows can be read. Equivalently, the consumer route factors through*

$$S, T, F \longmapsto U, E \longmapsto K, R \longmapsto H, C, P, N$$

with F *as the mandatory uniform-modulus handoff row.*

Proof. Project the carrier of Definition 2037.1. The source row S, target consumer row T, and uniformly continuous source-map row F are displayed before any completion-facing row.

Apply Definition 2037.2. Its uniform-continuity handoff obligation requires F to be consumed through the uniform completion row U, while its effective completion exactness obligation makes the stream windows, regular rational readback, dyadic tolerance ledger, and real seal downstream of E.

Now apply Definition 2037.3. The universal consumer route reads the lifted map through K and R only after S, T, F, U, E have been displayed, and the structural rows H, C, P, N merely preserve that route.

The excluded alternatives have no carrier coordinate. Quotient completion equality, selected limits, host function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot supply the required modulus handoff because every accepted consumer has already been routed through F. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionLift}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2037.1, Definition 2037.2, Definition 2037.3, and Definition 2037.4 seed the finite universal lift route for uniformly continuous finite-source maps across Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLift}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyCompletionLiftUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, representative choice, host function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-source visibility, uniform-continuity handoff, effective completion exactness, functorial action, universal consumer routing, and ledger non-escape for BEDC.Derived.CauchyCompletionL

Constructive story: A $\mathsf{CauchyCompletionLift}^{\uparrow}$ packet is generated from a finite-source metric row, a target completion-consumer row, a uniformly continuous source-map row, a UniformCompletionFunctor row, an EffectiveCompletion row, a CauchyCompletionFunctoriality row, a universal-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2038

A Concrete Naming Certificate for $\mathsf{ArchimedeanBracket}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArchimedeanBracket}^{\uparrow}$ packet records the finite integer-window route by which a displayed real name is bracketed before a Real or Completion consumer reads a located approximation. It consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the integer boundary rows of chapter 31, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, and the regular rational readback of chapter 301. The packet gives downstream order, fractional-part, located-comparison, and approximation chapters a named bracket surface without importing host floor functions, trichotomy, quotient real equality, selected integer parts, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 2038.1 (Archimedean bracket carrier)**.** *An* $\mathsf{ArchimedeanBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, L, U, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row whose displayed name is being bracketed at the requested precision.*
2. *L is the* $\mathsf{Int}^{\uparrow}$ *lower bracket row, recording the displayed integer bound admitted below the readback window.*
3. *U is the* $\mathsf{Int}^{\uparrow}$ *upper bracket row, recording the displayed integer bound admitted above the readback window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to state the finite dyadic radius and comparison scale for the bracket request.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from R at the tolerance recorded by D.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the displayed stream samples into rational observations before the integer rows are compared.*
7. *H records componentwise* hsame *transport for the real source, lower and upper integer rows, dyadic tolerance, stream windows, regular rational readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$R, D, S, Q, L, U \longmapsto \mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}.$$

9. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanBracket}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, L, U, D, S, Q, H, C, P, N$. It has no coordinate for a host floor operation, trichotomy on real values, quotient real equality, a selected integer part, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.ArchimedeanBracketUp.ArchimedeanBracketUpTasteGate single carrier alignment`

Theorem 2038.2 (Archimedean bracket NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanBracket}^{\uparrow}$ *carrier*

$$B = (R, L, U, D, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Real source row, lower and upper integer bracket rows, dyadic tolerance row, stream-window row, regular rational readback row, transport, replay, provenance, and local naming row;*
2. *source-before-window routing, requiring S to be opened only from the displayed* $\mathsf{Real}^{\uparrow}$ *row R at the tolerance recorded by D;*
3. *readback-before-bracket routing, requiring Q to expose the rational observations used by both integer comparisons before either L or U is exported as a bracket endpoint;*
4. *lower and upper endpoint exposure, requiring L and U to be the only integer rows visible to downstream bracket consumers;*
5. *transport and replay stability, requiring H and C to preserve the same row order without adding a second real source or another integer endpoint;*
6. *provenance and local naming, requiring P and N to source and name only the displayed packet.*

No Archimedean bracket obligation is stored outside these rows, and none exports host floor, trichotomy, quotient real equality, selected integer parts, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Carrier admission is projection from Definition 2038.1. The accepted packet displays exactly the Real source, lower and upper integer rows, dyadic tolerance, stream windows, regular rational readback, transport, replay, provenance, and local name.

The source and tolerance rows determine the finite window. The row S is opened only from R at the dyadic scale stored in D, so a bracket consumer cannot replace the displayed real-name route by a host real value.

The rational readback precedes the integer endpoints. The row Q reads the displayed stream samples before L and U are compared with that observation, making the two integer rows bracket endpoints for the same finite window.

The structural rows preserve the route. Componentwise transport through H and replay through C keep R, D, S, Q, L, U aligned, while P and N provide only provenance and local naming.

Since the excluded principles have no coordinate in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2038.3 (Archimedean bracket Real-window handoff)**.** *Every accepted* $\mathsf{ArchimedeanBracket}^{\uparrow}$ *read exported to a Real or Completion consumer factors through the displayed finite route*

$$R \longmapsto D \longmapsto S \longmapsto Q \longmapsto (L, U) \longmapsto H, C, P, N.$$

The consumer receives a single $\mathsf{Real}^{\uparrow}$ *source row, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, one* $\mathsf{StreamName}^{\uparrow}$ *window row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, two* $\mathsf{Int}^{\uparrow}$ *endpoint rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. The handoff is not a host floor theorem, not real trichotomy, not quotient real equality, not a selected integer part, not ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and not bridge verification.*

Proof. Project the accepted carrier $B = (R, L, U, D, S, Q, H, C, P, N)$. The Real-facing read begins with R, and the only tolerance visible to the packet is the dyadic row D.

Open the finite stream window S at that tolerance. Its regular rational readback is the row Q, which supplies the finite observations against which the integer rows are displayed.

The lower and upper bracket rows are then read together. Because L and U are the only integer coordinates in the carrier, every downstream bracket endpoint is one of those two rows and is attached to the same readback Q.

Finally, H, C, P, N transport, replay, source, and name the same route. No carrier coordinate can supply a host floor, trichotomy, quotient equality, selected integer part, ambient completeness theorem, or bridge credential, so every exported bracket handoff factors through the displayed finite route. □

Falsifiable prediction: Every $\mathsf{ArchimedeanBracket}^{\uparrow}$ bracket read factors through one RealUp source row, one DyadicRatCore tolerance row, one StreamName finite-window row, one RegSeqRat readback row, two IntUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, int, dyadicratcore, streamname, regseqrat: $\mathsf{ArchimedeanBracket}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Real source, a dyadic tolerance, a stream window, a regular rational readback, and lower and upper integer endpoint rows as one bracket certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ArchimedeanBracket}^{\uparrow}$ packet is generated from a Real source row, lower and upper Int rows, a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}$, seed)

Scope closed: Definition 2038.1, Definition 2038.2, and Definition 2038.3 seed the finite integer-window bracket certificate over Real, dyadic tolerance, stream-window, regular rational readback, and integer endpoint rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ArchimedeanBracket}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host floor functions, trichotomy, quotient real equality, selected integer parts, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-before-window routing, readback-before-bracket routing, lower and upper endpoint exposure, transport and replay stability, provenance, and local naming for BEDC.Derived.ArchimedeanBracketUp.

Chapter 2039

A Concrete Naming Certificate for $\mathsf{RatioCauchyTransfer}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RatioCauchyTransfer}^{\uparrow}$ packet records the finite route by which a quotient of regular rational Cauchy names is transported into the Real and Completion surface. It sits after the denominator guard of chapter 1471, the guarded quotient route of chapter 1943, the modulus arithmetic discipline of chapter 1485, the apartness-completion boundary of chapter 1567, and the terminal Real seal of chapter 38. The packet names quotient transfer as displayed BEDC rows; it is not host division, quotient equality of Cauchy streams, a selected representative, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2039.1 (Ratio Cauchy transfer carrier)**.** *A* $\mathsf{RatioCauchyTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (N, D, A, R, Q, H, C, P, L)$$

with the following displayed rows:

1. *N is the numerator* $\mathsf{RegSeqRat}^{\uparrow}$ *Cauchy row, including the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows used by the numerator readback.*
2. *D is the denominator* $\mathsf{RegSeqRat}^{\uparrow}$ *Cauchy row, displayed with the same finite-window and dyadic-tolerance discipline as N.*
3. *A is the apartness lower-bound ledger for D, recording the finite nonzero guard consumed by reciprocal and completion-facing reads.*
4. *R is the* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *route opened from D only after A has been displayed.*
5. *Q is the* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *quotient handoff, reading the numerator row N and the reciprocal route R before it exports any quotient readback.*
6. *H records componentwise* hsame *transport for numerator, denominator, apartness ledger, reciprocal route, quotient handoff, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$N, D, A \longmapsto R \longmapsto Q \longmapsto \mathsf{Real}^{\uparrow}.$$

8. *P is the* Pkg *provenance over* $\mathsf{RatioCauchyTransfer}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyDivision}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{RealApartnessCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{RatioCauchyTransfer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, D, A, R, Q, H, C, P, L$. It has no coordinate for host division, quotient equality of streams, selected representatives, trichotomy, countable choice, hidden denominator inversion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RatioCauchyTransferUp.RatioCauchyTransferTasteGate single carrier alignment`

Theorem 2039.2 (Ratio Cauchy transfer NameCert obligations)**.** *For every accepted* $\mathsf{RatioCauchyTransfer}^{\uparrow}$ *carrier*

$$X = (N, D, A, R, Q, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{RatioCauchyTransfer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the numerator Cauchy row, denominator Cauchy row, apartness lower-bound ledger, reciprocal route, quotient handoff, transport rows, continuation rows, package provenance, and local naming row;*
2. *numerator visibility, requiring every quotient or Real-facing consumer to expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *numerator row N before using the quotient row Q;*
3. *denominator guard visibility, requiring every reciprocal or quotient read to expose the denominator row D and apartness ledger A before the reciprocal route R is opened;*
4. *modulus arithmetic compatibility, requiring the finite windows and dyadic tolerance rows used by N, D, R, Q to be read through the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *discipline rather than through a host limit comparison;*
5. *quotient handoff exactness, requiring Q to consume only N and the reciprocal surface R before the terminal* $\mathsf{Real}^{\uparrow}$ *read is available;*
6. *apartness-completion boundary, requiring every Real or Completion consumer to retain the displayed A, R, Q route before it cites the* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *boundary;*
7. *transport, replay, provenance, and local naming, requiring H, C, P, L to preserve, replay, source, and name the displayed route without adding a second classifier.*

No Ratio Cauchy transfer obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2039.1. The accepted packet displays exactly the numerator, denominator, apartness ledger, reciprocal route, quotient handoff, transport, continuation, provenance, and local naming rows.

The numerator and denominator clauses are read from N, D, A. A quotient consumer can reach Q only after the numerator row is visible, and the denominator side cannot open R until the apartness lower-bound ledger A has been displayed.

The modulus arithmetic clause is read through the finite observation rows attached to N, D, R, Q. These rows are $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data governed by the $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ discipline, so no host limit comparison is part of the carrier.

The quotient and completion clauses are exhausted by R, Q together with the $\mathsf{RealApartnessCompletion}^{\uparrow}$ boundary named in the provenance row. The structural rows H, C, P, L transport, replay, source, and name that same finite route.

Since the carrier has no coordinate for host division, quotient equality of streams, selected representatives, trichotomy, countable choice, hidden denominator inversion, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the local NameCert surface. □

Theorem 2039.3 (Ratio Cauchy denominator non-escape)**.** *For every accepted* $\mathsf{RatioCauchyTransfer}^{\uparrow}$ *carrier*

$$X = (N, D, A, R, Q, H, C, P, L),$$

every denominator use in a reciprocal, quotient, Real, or Completion read factors through

$$D \longrightarrow A \longrightarrow R \longrightarrow Q$$

with transport, replay, provenance, and local naming supplied only by H, C, P, L*. The denominator row cannot escape as a host divisor, quotient-stream representative, hidden inverse, selected nonzero witness, trichotomy branch, or ambient completeness object.*

Proof. Start with any consumer that uses the denominator. The only denominator coordinate in the carrier is D, and the only nonzero evidence attached to it is the apartness lower-bound ledger A.

The reciprocal route R is downstream of D and A. Thus a reciprocal read cannot treat D as a host divisor or open an inverse without first displaying the finite apartness ledger.

The quotient handoff Q is downstream of R and the numerator row N. Once the route reaches Q, the denominator has already been consumed through the guarded reciprocal surface, not through host division or quotient equality.

Finally inspect H, C, P, L. These rows can only transport, replay, source, and name the displayed route. They do not add a selected nonzero witness, hidden inverse, trichotomy branch, quotient-stream representative, or ambient completeness object, so the denominator has no escape channel beyond D, A, R, Q. □

Theorem 2039.4 (Ratio Cauchy quotient route)**.** *For every accepted* $\mathsf{RatioCauchyTransfer}^{\uparrow}$ *carrier*

$$X = (N, D, A, R, Q, H, C, P, L),$$

every $\mathsf{Real}^{\uparrow}$ *or Completion-facing quotient read factors through the finite route*

$$N, D, A \longmapsto R \longmapsto Q \longmapsto \mathsf{Real}^{\uparrow}$$

and the route remains stable under the displayed hsame *transports and* Cont *replay rows. The read exports no host division, quotient equality of Cauchy streams, selected representative, countable-choice selector, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Fix a Real-facing or Completion-facing quotient read. The only quotient handoff in the carrier is Q, so the read must pass through that row before any terminal Real surface is visible.

Read the dependencies of Q. It consumes the numerator row N and the reciprocal route R; the reciprocal route consumes the denominator row D only after the apartness lower-bound ledger A is displayed.

Transport stability is componentwise. The rows in H preserve N, D, A, R, Q under the displayed hsame classifications, and the rows in C replay the same dependency order through Cont without inserting a second quotient constructor.

The provenance row P records the sibling sources for regular Cauchy reciprocal, guarded division, modulus arithmetic, apartness completion, and Real sealing, while L names only this local packet. Since no excluded object appears among $N, D, A, R, Q, H, C, P, L$, the quotient read exports exactly the displayed finite route and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RatioCauchyTransfer}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RatioCauchyTransfer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2039.1, Definition 2039.2, Definition 2039.3, and Definition 2039.4 seed the finite quotient-transfer route from regular Cauchy numerator and guarded denominator rows to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RatioCauchyTransfer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host division, quotient equality of Cauchy streams, selected representatives, trichotomy, countable choice, hidden denominator inversion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, denominator guard visibility, modulus arithmetic compatibility, quotient handoff exactness, apartness-completion boundary, denominator non-escape, quotient route stability, transport, replay, provenance, and local naming for BEDC.Derived.RatioCauchyTransferUp.

Constructive story: A $\mathsf{RatioCauchyTransfer}^{\uparrow}$ packet is generated from numerator and denominator RegSeqRat rows, an apartness lower-bound ledger, a RegularCauchyReciprocal route, a RegularCauchyDivision quotient handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2040

A Concrete Naming Certificate for $\mathsf{CauchyModulusChoice}^{\uparrow}$

Bishop–Bridges constructive analysis treats a Cauchy modulus as displayed finite rate data: a later real-name or completion-facing consumer may read the selected threshold only after the raw modulus, the normal-form schedule, the window row, the regular rational readback, and the dyadic tolerance ledger have all been exposed. The $\mathsf{CauchyModulusChoice}^{\uparrow}$ packet records this selection route without importing a countable-choice principle. It sits after the raw modulus surface of chapter 292, the normal-form route of chapter 1968, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite modulus-choice certificate for Real and Completion consumers; it is not quotient stream equality, trichotomy, ambient Real completeness, or a hidden selector over countably many requests.

Definition 2040.1 (Cauchy modulus choice carrier)**.** *A* $\mathsf{CauchyModulusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, N, W, Q, D, R, H, C, P, L)$$

with the following displayed rows:

1. *M is the raw* $\mathsf{CauchyModulus}^{\uparrow}$ *row attached to the source Cauchy name and the finite precision request being answered.*
2. *N is the* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ *row that records the monotone normal-form representative of M for the same request.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the normal-form threshold.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the dyadic error budget consumed by W and Q.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, N, W, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the raw modulus, normal form, window, readback, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the selection route*

$$M \longmapsto N \longmapsto W \longmapsto Q \longmapsto D \longmapsto R.$$

9. *P is the* Pkg *provenance over* CauchyModulusChoice$^\uparrow$, CauchyModulus$^\uparrow$, CauchyModulusNormalForm$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{CauchyModulusChoice}^\uparrow}$ *row.*

The classifier compares accepted packets only through $M, N, W, Q, D, R, H, C, P, L$.

Lean4 ✓ `BEDC.Derived.CauchyModulusChoiceUp.CauchyModulusChoiceTasteGate single carrier alignment`

Theorem 2040.2 (Cauchy modulus choice NameCert obligations)**.** *For every accepted* CauchyModulusChoice$^\uparrow$ *carrier*

$$K = (M, N, W, Q, D, R, H, C, P, L),$$

the local NameCert$_{\text{CauchyModulusChoice}^\uparrow}$ *surface consists exactly of five finite obligation rows:*

1. *carrier exposure for the raw modulus, normal-form modulus, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, replay, provenance, and local naming rows;*
2. *normal-form selection, requiring the chosen threshold to be read from the displayed* CauchyModulusNormalForm$^\uparrow$ *row N rather than from an external choice function;*
3. *window-readback compatibility, requiring the* StreamName$^\uparrow$ *window W and* RegSeqRat$^\uparrow$ *readback Q to be opened by the same normal-form threshold;*
4. *dyadic tolerance and Real seal compatibility, requiring the* DyadicRatCore$^\uparrow$ *row D to be the tolerance ledger consumed before the terminal* Real$^\uparrow$ *seal R is exposed;*
5. *structural non-escape, requiring H, C, P, L to transport, replay, source, and name only the displayed route and to export no countable choice, quotient stream equality, trichotomy, ambient Real completeness, selected limit, or host real equality.*

No Cauchy modulus choice obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2040.1. The only rows available to the classifier are $M, N, W, Q, D, R, H, C, P, L$.

Normal-form selection is local to N. Since N is a displayed normal-form row downstream of the raw modulus M, the threshold read is a finite row read rather than an appeal to a choice function over all precision requests.

Window-readback compatibility uses W and Q. The finite window is opened by the same normal-form threshold, and the regular rational readback is attached to that displayed window.

Dyadic tolerance and the Real seal are then forced by D and R. The dyadic budget is consumed before the seal can be read, so the seal is not an ambient completeness assertion or a selected limit.

Finally H, C, P, L preserve exactly the same route by componentwise transport, replay, provenance, and local naming. Because the carrier has no coordinate for countable choice, quotient stream equality, trichotomy, ambient completeness, selected limits, or host real equality, the five obligation rows exhaust the local surface. □

Theorem 2040.3 (Cauchy modulus choice normal-form factorization)**.** *Every accepted* CauchyModulusChoice$^\uparrow$ *carrier*

$$K = (M, N, W, Q, D, R, H, C, P, L)$$

factors its selected threshold through the CauchyModulusNormalForm$^\uparrow$ *row N before any* StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, *or* Real$^\uparrow$ *consumer reads the route. Equivalently, every Real and Completion use of the choice surface is routed through*

$$M \longmapsto N \longmapsto W \longmapsto Q \longmapsto D \longmapsto R$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Begin with an accepted carrier K. The raw modulus row M exposes the Cauchy modulus request, while N records the normal-form representative consumed by later rows.

The route cannot reach a finite window before N is read. Thus the $\mathsf{StreamName}^{\uparrow}$ row W is the window opened by the displayed normal-form threshold, not by an unrecorded selector.

Readback and tolerance follow the same route. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q reads the displayed window, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the tolerance ledger for that readback.

The terminal $\mathsf{Real}^{\uparrow}$ seal R is downstream of those four rows. The structural rows H, C, P, L keep the route componentwise, replayable, sourced, and locally named, so every consumer factors through the normal-form surface before using the selected threshold. □

Theorem 2040.4 (Cauchy modulus choice window determinacy)**.** *Let*

$$K = (M, N, W, Q, D, R, H, C, P, L) \quad \textit{and} \quad K' = (M, N, W', Q', D', R', H', C', P', L')$$

be accepted $\mathsf{CauchyModulusChoice}^{\uparrow}$ *carriers with the same raw modulus row M and normal-form row N. Then every* $\mathsf{Real}^{\uparrow}$ *or completion-facing read of the selected threshold in either carrier passes through the same finite window, readback, and tolerance route determined by N: first the* $\mathsf{StreamName}^{\uparrow}$ *window row, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, before the* $\mathsf{Real}^{\uparrow}$ *seal. The claim is only route determinacy inside the displayed normal-form surface; it is not host uniqueness of limits, quotient stream equality, or countable choice.*

Proof. Project both accepted carriers using Definition 2040.1. The only rows upstream of a selected-threshold read are the raw modulus row M, the normal-form row N, the finite window row, the regular readback row, and the dyadic tolerance row.

Apply the normal-form selection obligation from Definition 2040.2. Since the two carriers share M and N, a threshold read in either packet is forced to use the same displayed normal-form source before any window can be opened.

Use Definition 2040.3. It places the downstream route in the order W, Q, D before R, so every Real or completion-facing read is mediated by the window, readback, and tolerance rows attached to that shared normal-form source.

The structural rows H, C, P, L and H', C', P', L' only transport, replay, source, and name the displayed route. Because neither carrier contains host limit uniqueness, quotient stream equality, or a countable-choice coordinate, the determinacy asserted here is exactly the finite route through W, Q, D before R. □

Theorem 2040.5 (Cauchy modulus choice Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyModulusChoice}^{\uparrow}$ *carrier receives only the displayed rows*

$$M, N, W, Q, D, R, H, C, P, L.$$

The Real seal R therefore cannot escape into countable choice, quotient stream equality, trichotomy, ambient Real completeness, selected limits, host real equality, or a second modulus-selection oracle.

Proof. Project the accepted carrier. Its Real-facing surface is exactly the raw modulus row, normal-form row, finite window, regular readback, dyadic tolerance, seal, transport, replay, provenance, and local naming row.

The normal-form factorization of Definition 2040.3 places every selected threshold before the finite window and readback rows. Therefore a Real or Completion consumer cannot bypass the displayed normal-form row to obtain a hidden selector.

The dyadic tolerance row is also part of the same route before the Real seal. Since R is exposed only after D, it cannot be read as a trichotomy witness, an ambient completeness principle, a quotient stream equality, or a selected host limit.

The structural rows add no mathematical coordinate beyond transport, replay, provenance, and local naming. Hence the displayed packet exhausts the Real seal route and blocks all listed escape principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusChoice}\uparrow}$.

Constructive story: A $\mathsf{CauchyModulusChoice}^{\uparrow}$ packet is generated from a raw CauchyModulus row, a CauchyModulusNormalForm row, a StreamName finite-window row, a RegSeqRat readback row, a DyadicRatCore tolerance row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusChoice}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2040.1, Definition 2040.2, Definition 2040.3, and Definition 2040.5 seed the finite choice-free Cauchy-modulus selection route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusChoice}\uparrow}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchyModulusChoiceUp`

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient stream equality, trichotomy, ambient Real completeness, selected limits, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, normal-form selection, window-readback compatibility, dyadic tolerance and Real seal compatibility, structural non-escape, normal-form factorization, and Real seal nonescape for BEDC.Derived.CauchyModulusChoiceUp.

Chapter 2041

A Concrete Naming Certificate for $\mathsf{DiniDerivative}^{\uparrow}$

Following Dini's one-sided derivative theory and Bishop–Bridges constructive analysis, the $\mathsf{DiniDerivative}^{\uparrow}$ packet records only displayed finite one-sided secant-window rows over $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$. It consumes the terminal real-name surface of chapter 38, the graph and modulus rows of chapter 130, the derivative quotient boundary of chapter 122, the dyadic arithmetic core of chapter 291, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, and the metric comparison rows of chapter 57. The packet names only finite right-hand and left-hand slope windows, their rational bounds, endpoint seals, componentwise transport, replay, provenance, and local naming; it does not import host limits, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2041.1 (Dini derivative carrier)**.** *A* $\mathsf{DiniDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, F, Q, S, B, R, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the local* $\mathsf{Met}^{\uparrow}$ *source row naming the inspected interval cell, its endpoint order, and the distance comparison used to choose one-sided neighborhoods.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed function on the same metric cell, including the finite graph reads and continuity modulus rows visible to the packet.*
3. *Q is the* $\mathsf{Derivative}^{\uparrow}$ *quotient-inspection row, recording the secant quotient route without asserting a two-sided derivative or a selected limit.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *one-sided window row, listing the inspected right-hand and left-hand approach samples.*
5. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *slope-bound row, storing the dyadic lower and upper rational bounds for every displayed one-sided secant window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximations of endpoint values, secant numerators, step sizes, and dyadic slope bounds.*
7. *W is the boundary-seal row, storing the* $\mathsf{Real}^{\uparrow}$ *endpoint names and the finite evidence that the requested Dini read is confined to the displayed side and window.*

8. *H records componentwise* hsame *transport for the metric source, continuous-map row, derivative quotient route, stream windows, dyadic bounds, rational readbacks, endpoint seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay through*

$$X, F, Q, S, B, R, W \longmapsto \mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}.$$

10. *P is the* Pkg *provenance over* $\mathsf{DiniDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host limit, trichotomy of reals, quotient equality of completed reals, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.DiniDerivativeUp.DiniDerivativeTasteGate single carrier alignment`

Theorem 2041.2 (Dini derivative NameCert obligations)**.** *For every accepted* $\mathsf{DiniDerivative}^{\uparrow}$ *carrier*

$$D = (X, F, Q, S, B, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *metric cell admission through X;*
2. *continuous-map row admission through F over the same cell;*
3. *derivative quotient inspection through Q;*
4. *one-sided finite window admission through S;*
5. *dyadic slope-bound admission through B;*
6. *regular rational readback through R;*
7. *endpoint and side boundary sealing through W;*
8. *structural stability through H, C, P, N.*

No Dini derivative obligation is stored outside the displayed carrier.

Proof. Carrier admission is projection from Definition 2041.1. The metric source row X fixes the inspected cell and the endpoint side, while F supplies the only continuous-map data visible to the packet.

The derivative-facing part is then finite. The row Q records the quotient route being inspected, and S lists the right-hand and left-hand sample windows on which that route may be read.

The bound rows are also displayed. The dyadic row B stores the rational lower and upper slope bounds, and R supplies the regular rational readback for the endpoint values, step sizes, and bound comparisons.

Finally, W seals the endpoint and side information, while H, C, P, N transport, replay, source, and name the same row inventory. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2041.3 (Dini derivative finite slope window)**.** *Every one-sided slope read exported by an accepted* $\mathsf{DiniDerivative}^{\uparrow}$ *carrier factors through the displayed finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The read first opens the metric cell and continuous-map row, then the derivative quotient route, then a finite one-sided stream window, then the dyadic slope-bound row, and only then seals the observed boundary as $\mathsf{Real}^{\uparrow}$ *data.*

Proof. Start with an accepted carrier $D = (X, F, Q, S, B, R, W, H, C, P, N)$. The packet exposes no slope source before the metric row X and continuous-map row F have been read.

Inspect the derivative row Q next. Its quotient data is not a host derivative; it is only the displayed route whose one-sided samples are listed by S.

The finite window is bounded through B and read back through R. These two rows provide the dyadic lower and upper slope bounds and the rational observations required to compare the displayed secant quotients.

The boundary row W then seals the observation as $\mathsf{Real}^{\uparrow}$ data on the chosen side. Transport and replay through H and C preserve the same route, and P, N add only source and naming information. Thus every exported one-sided slope read factors through the stated finite route. □

Theorem 2041.4 (Dini derivative real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *boundary read exported by an accepted* $\mathsf{DiniDerivative}^{\uparrow}$ *carrier is downstream of a displayed one-sided secant-window comparison. In particular, if a consumer reaches the boundary row* W*, then the consumer has already read the same metric cell* X*, continuous-map row* F*, derivative quotient row* Q*, stream-window row* S*, dyadic slope-bound row* B*, regular rational readback* R*, componentwise transport* H*, replay* C*, package provenance* P*, and local name* N*.*

Proof. The carrier places the $\mathsf{Real}^{\uparrow}$ boundary information in the single row W. By the definition of the carrier, W is not independent data: it records only endpoint names and side confinement for the displayed one-sided window.

To reach W, a consumer must therefore pass through the rows that make the window meaningful. The metric row X supplies the side and neighborhood comparison, F supplies the continuous-map observations, Q supplies the quotient route, S supplies the sampled windows, and B, R supply the dyadic and regular-rational comparison data.

The structural rows do not add a second boundary route. They preserve the same packet under hsame, replay it through Cont, record its provenance, and assign the local $\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}$ row.

Consequently the boundary read cannot be detached from the displayed finite secant-window comparison, and it cannot be replaced by a host limit, real trichotomy, quotient real equality, or ambient completeness argument. □

Falsifiable prediction: Every $\mathsf{DiniDerivative}^{\uparrow}$ one-sided slope read factors through one metric cell row, one ContinuousMap row, one Derivative quotient row, one StreamName finite window row, one DyadicRatCore slope-bound row, one RegSeqRat readback row, one RealUp boundary-seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, continuousmap, derivative, dyadicratcore, regseqrat, streamname, metric: $\mathsf{DiniDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one-sided finite secant windows, derivative quotient inspection, dyadic slope bounds, rational readback, endpoint side sealing, transport, replay, provenance, and local naming as one boundary certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniDerivative}^{\uparrow}$ packet is generated from a metric cell row, a ContinuousMap row, a Derivative quotient route, StreamName one-sided finite windows, DyadicRatCore slope bounds, RegSeqRat readbacks, a RealUp boundary-seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2041.1, Definition 2041.2, Definition 2041.3, and Definition 2041.4 seed the finite one-sided derivative-window certificate over metric, continuous-map, quotient, stream, dyadic, rational, and Real boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limits, real trichotomy, quotient equality of completed reals, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, one-sided window admission, dyadic slope-bound admission, rational readback, endpoint-side sealing, transport, replay, provenance, and local naming for BEDC.Derived.DiniDerivativeUp.

Chapter 2042

A Concrete Naming Certificate for $\mathsf{HyersUlamStability}^{\uparrow}$

Following Hyers and Ulam, the $\mathsf{HyersUlamStability}^{\uparrow}$ packet records constructive stability of approximate solutions as finite correction data rather than as an ambient completeness theorem. It reads the displayed target-completeness row of chapter 128, the Lipschitz-control ledger of chapter 410, the contraction route of chapter 1418 and chapter 395, and the terminal real-name seal of chapter 38. The chapter names the surface on which an approximate equation row, an error-bound row, and a correction witness row are kept together before Picard, ODE, or completion-facing consumers read the corrected solution.

Definition 2042.1 (Hyers–Ulam stability carrier)**.** *A* $\mathsf{HyersUlamStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, L, M, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the approximate-solution row, containing the displayed domain cell, candidate solution graph, residual observation, and tolerance request.*
2. *W is the correction-witness row, storing the corrected solution name, the finite comparison from the approximate graph to the corrected graph, and the witnessed error bound.*
3. *L is the Lipschitz and control ledger. It records the* $\mathsf{LipschitzMap}^{\uparrow}$ *row, the displayed control constant, the contraction threshold when present, and the finite residual-to-correction estimate.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, carrying the regular Cauchy limit witness or complete target surface needed to accept the corrected solution without selecting a host limit.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed error, control, and correction-bound observations.*
6. *H records componentwise* hsame *transport for the approximate row, correction witness, control ledger, complete-metric handoff, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through*

$$A, W, L, M, R \longmapsto \mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}.$$

8. *P is the* Pkg *provenance over* $\mathsf{HyersUlamStability}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, L, M, R, H, C, P, N$. It has no coordinate for host function equality, a selected exact solution, quotient completion equality, an unbounded stability constant, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.HyersUlamStabilityUp.HyersUlamStabilityUpTasteGate single carrier alignment`

Theorem 2042.2 (Hyers–Ulam stability NameCert obligations)**.** *For an accepted* $\mathsf{HyersUlamStability}^{\uparrow}$ *carrier*

$$S = (A, W, L, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *approximate-row admission for the displayed candidate solution, residual observation, and tolerance request in A;*
2. *correction-witness admission for the corrected solution name, graph comparison, and error-bound witness in W;*
3. *control-ledger admission for the Lipschitz row, contraction threshold when present, and residual-to-correction estimate in L;*
4. *complete-metric handoff through M, requiring every corrected-solution acceptance to pass through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *row;*
5. *real-bound sealing through R, requiring every error, control, and correction-bound read to reach* $\mathsf{Real}^{\uparrow}$ *only after A, W, L, M have been displayed;*
6. *structural stability through H, C, P, N.*

No Hyers–Ulam stability obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2042.1. The approximate candidate, correction witness, control ledger, complete-metric handoff, real-bound seal, transport, replay, provenance, and local name are all rows of the same finite packet.

The approximate and corrected solution data are separated but not detached. The row A exposes the residual and tolerance request, while W supplies the corrected solution name and the finite comparison back to the approximate graph.

The control and completion rows then bind the correction. The row L contains the Lipschitz or contraction ledger that turns the residual into a correction bound, and M supplies the complete-metric handoff through which the corrected solution is accepted.

Finally R, H, C, P, N seal and preserve exactly that route. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export a selected exact solution, host function equality, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2042.3 (Hyers–Ulam stability correction route)**.** *Let $S = (A, W, L, M, R, H, C, P, N)$ be an accepted* $\mathsf{HyersUlamStability}^{\uparrow}$ *carrier. Every correction read exported by S factors through the finite route*

$$\mathsf{LipschitzMap}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardContraction}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed inside L, M, R, H, C, P, N. In particular, the corrected solution is read as a NameCert witness downstream of the approximate residual and the control ledger, not as a host-selected exact solution and not as a consequence of ambient real completeness.

Proof. Start with a correction request. The row A displays the approximate solution and the residual tolerance whose stability is being inspected.

Read the control ledger L. Its $\mathsf{LipschitzMap}^{\uparrow}$ and contraction-facing rows provide the finite control data that relate the residual to a correction bound, and its Picard-facing row records the iterative correction route when the packet is consumed by a fixed-point or ODE surface.

The corrected solution is accepted through W and M. The witness row W gives the corrected name and comparison, while the complete-metric handoff M supplies the convergence or target-completeness route needed by downstream consumers.

The real-valued bound is sealed only after those rows have been displayed. The row R records the $\mathsf{Real}^{\uparrow}$ observations for the residual, control constant, and correction estimate, and the structural rows H, C, P, N transport, replay, source, and name the same finite route. Hence the exported correction has no path through a selected exact solution, host function equality, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2042.4 (Hyers–Ulam completion handoff)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{HyersUlamStability}^{\uparrow}$ *packet receives the approximate row, correction witness, Lipschitz-control ledger, and real-bound seal as one displayed route. The complete-metric row M is therefore a handoff for the correction witness W, not an independent assertion that all approximate solutions converge, not a completeness theorem for* $\mathsf{Real}^{\uparrow}$*, and not a choice principle for exact solutions.*

Proof. Begin with a completion-facing consumer. The only complete-metric coordinate in the carrier is M, and it is listed after the approximate row A, the correction witness W, and the control ledger L.

The consumer therefore reads M together with the rows that make the correction meaningful. The approximate residual is visible in A, the corrected solution name and graph comparison are visible in W, and the stability estimate is visible in L before the real-bound seal R is consumed.

Componentwise transport and displayed replay preserve this dependency. The rows H and C move and replay the same packet, while P and N source and name it as $\mathsf{HyersUlamStability}^{\uparrow}$ data.

Thus completion is exposed only as the handoff for the displayed correction route. No row supplies an independent convergence theorem for arbitrary approximate solutions, completeness of $\mathsf{Real}^{\uparrow}$, host equality of functions, selected exact solutions, or countable choice. □

Falsifiable prediction: Every $\mathsf{HyersUlamStability}^{\uparrow}$ correction read factors through one approximate-solution row, one correction witness row, one Lipschitz or contraction control ledger, one CompleteMetric handoff row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: complete-metric, lipschitz-map, contraction-mapping, picard-contraction, real: $\mathsf{HyersUlamStability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries approximate residual data, a correction witness, a control ledger, a complete-metric handoff, a RealUp seal, transport, replay, provenance, and local naming as one stability certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyersUlamStability}^{\uparrow}$ packet is generated from an approximate-solution row, a correction witness row, a Lipschitz or contraction control ledger, a CompleteMetric handoff row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2042.1, Definition 2042.2, Definition 2042.3, and Definition 2042.4 seed the finite stability certificate for approximate solutions through correction witnesses, control ledgers, complete-metric handoff, and Real-bound sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyersUlamStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected exact solutions, quotient completion equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximate-row admission, correction-witness admission, control-ledger admission, complete-metric handoff, real-bound sealing, transport, replay, provenance, and local naming for BEDC.Derived.HyersUlamStabilityUp.

Chapter 2043

A Concrete Naming Certificate for $\mathsf{LocatedSpace}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedSpace}^{\uparrow}$ packet records the finite route by which a metric source answers locatedness requests without importing a global real-order decision or a completed ambient space. It sits between the apartness interface of chapter 1514, the compact located handoff of chapter 1488, and the completion boundary of chapter 1506: every consumer sees the finite $\mathsf{StreamName}^{\uparrow}$ window of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before it can treat a distance-to-source query as located.

Definition 2043.1 (LocatedSpace carrier)**.** *A* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, R, A, G, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric or source carrier whose finite points, distance requests, and source-side observation windows are being certified.*
2. *R is the finite locatedness request, consisting of the displayed point or name to be tested, the source subset or source family being queried, and the dyadic tolerance budget at which the request is opened.*
3. *A is the positive* $\mathsf{ApartnessSpace}^{\uparrow}$ *gap row attached to the request, displaying the positive separation witness used when the request is answered on the apart side.*
4. *G is the positive gap budget shared by the apartness row, the metric distance comparison, and the completion-facing refusal boundary.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window through which the source point, target observation, tolerance budget, and gap comparison are inspected.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to every rational comparison used by the locatedness request.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached by the displayed distance and gap reads after R, A, G, W, Q have been exposed.*
8. *H records componentwise* hsame *transport for the source carrier, locatedness request, positive apartness gap, stream window, regular readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of every locatedness-facing consumer route through*

$$X, R, A, G, W, Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSpace}^{\uparrow}$, $\mathsf{ApartnessSpace}^{\uparrow}$, $\mathsf{LocatedCompact}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, R, A, G, W, Q, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected limits, open-cover compactness, choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceTasteGate single carrier alignment`

Theorem 2043.2 (LocatedSpace NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier*

$$L = (X, R, A, G, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric or source carrier, locatedness request, positive apartness gap, stream window, regular readback, real seal, transport, replay, provenance, and local naming rows;*

2. *locatedness stability, requiring every transported or replayed request to read the same source carrier X, request row R, gap row A, gap budget G, window W, and readback Q before exposing the real seal E;*

3. *positive-gap exactness, requiring the apartness gap used by A to be the same positive budget G read by the finite metric comparison and the regular rational readback;*

4. *completion-boundary nonescape, requiring every completion-facing or real-facing consumer to pass through R, A, G, W, Q, E and to receive no selected limit, quotient representative, ambient completeness theorem, or host real equality from the packet;*

5. *ledger refusal, requiring H, C, P, N to transport, replay, source, and name only the displayed rows, with no trichotomy, quotient real equality, open-cover compactness, choice, Lean verification, or bridge credential.*

No located-space NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 2043.1. The packet displays the metric source, the finite locatedness request, the apartness gap, the gap budget, the stream window, the regular readback, the real seal, and the four structural rows.

Locatedness stability is componentwise. A transported or replayed request keeps the same source row and the same finite request row, so the downstream consumer cannot replace the displayed window and readback by an ambient predicate on the completed space.

Positive-gap exactness is the shared role of A and G. The apartness row names the positive separation, while G is the budget read both by the metric comparison and by the regular rational observation.

For the completion boundary, the real seal E appears only after R, A, G, W, Q have been displayed. A completion-facing consumer therefore receives a finite locatedness route rather than a quotient class, selected limit, or completeness theorem.

Finally, H, C, P, N only transport, replay, source, and name the carrier rows. Since the packet contains no trichotomy, quotient real equality, open-cover compactness, choice, Lean verification, or bridge credential, the listed rows exhaust the NameCert surface. □

Theorem 2043.3 (LocatedSpace Real completion boundary)**.** *Every completion-facing read of an accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier*

$$L = (X, R, A, G, W, Q, E, H, C, P, N)$$

factors through the displayed locatedness request R, the positive apartness gap A, the positive gap budget G, the $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the* $\mathsf{Real}^{\uparrow}$ *seal boundary E. The factorization exports only the finite route*

$$X, R, A, G, W, Q \longmapsto E$$

and exports no trichotomy, quotient real equality, selected limit, open-cover compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier real completion boundary`

Proof. Start with an accepted carrier and a completion-facing read. Projection from Definition 2043.1 gives the rows $X, R, A, G, W, Q, E, H, C, P, N$ and no further completion object.

The read first opens the locatedness request R against the source carrier X. If the apart side is used, the only positive separation data are the apartness row A and the positive budget G.

The observation then passes through the finite stream window W and the regular rational readback Q. These are the only rows that expose the rational data used by the distance and gap comparisons.

Only after those rows have been displayed can the route reach the real seal E. The structural rows H, C, P, N preserve this order under transport, replay, provenance, and local naming.

Thus every completion-facing read factors through the displayed finite route. The excluded trichotomy, quotient equality, selected-limit, compactness, choice, and ambient-completeness data cannot be exported because they are absent from the accepted packet. □

Theorem 2043.4 (LocatedSpace apartness request scope)**.** *For every accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier*

$$L = (X, R, A, G, W, Q, E, H, C, P, N),$$

an apartness-facing locatedness request is scoped by the same metric source X, finite request row R, positive apartness gap A, and positive budget G before any $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, or* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. Componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$ *row preserve that scope and cannot introduce a second apartness gap or a host order decision.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier apartness request scope`

Proof. Project the rows from Definition 2043.1. The apartness-facing part of the packet is exactly X, R, A, G, with A carrying the positive gap and G carrying the budget shared with the metric comparison.

Apply Definition 2043.2. Its locatedness-stability and positive-gap-exactness clauses require every transported or replayed request to keep the same source, request row, gap row, and gap budget before any downstream readback is opened.

Finally use Definition 2043.3. The route to W, Q, E comes after the scoped request rows, and H, C, P, N preserve only the displayed carrier. Thus no host order decision or additional apartness gap can enter the request scope. □

Theorem 2043.5 (LocatedSpace positive-gap exactness route)**.** *For every accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier*

$$L = (X, R, A, G, W, Q, E, H, C, P, N),$$

the apartness gap row A and positive budget row G are the unique finite gap surface available to metric, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *consumers. Every completion-facing read must pass through R, A, G, W, Q before E, and no host trichotomy, quotient equality, selected limit, or second apartness witness is exported.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier positive gap exactness route`

Proof. Start with the NameCert obligations of Definition 2043.2. Carrier admission exposes $X, R, A, G, W, Q, E, H, C, P, N$, while positive-gap exactness binds the apartness row A to the same budget row G used by the metric and regular rational reads.

Use the apartness request scope of Definition 2043.4. It places X, R, A, G before any stream window, regular readback, or real seal can be consumed, so a second apartness witness has no carrier row.

Finally apply the Real completion boundary of Definition 2043.3. That theorem orders every completion-facing read through R, A, G, W, Q before E and excludes trichotomy, quotient equality, selected limits, and ambient completion data. Therefore A, G are the exact finite gap surface available to downstream consumers. □

Theorem 2043.6 (LocatedSpace metric request ledger)**.** *Every metric request exported by an accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier is ledgered by the finite route*

$$X, R, A, G, W, Q, E, H, C, P, N.$$

The metric source row X, request row R, apartness gap A, gap budget G, $\mathsf{StreamName}^{\uparrow}$ *window W, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q must all be displayed before the* $\mathsf{Real}^{\uparrow}$ *seal E is consumed. The structural rows* hsame*,* Cont*,* Pkg*, and local* NameCert *record only that route.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier metric request ledger`

Proof. The carrier projection fixes the metric source, request, gap, window, readback, real seal, and structural rows. No metric-facing consumer has access to a coordinate outside $X, R, A, G, W, Q, E, H, C, P, N$.

Apply the NameCert obligation theorem. Carrier admission exposes the displayed metric source and request rows, while locatedness stability and positive-gap exactness bind the metric comparison to the same apartness budget read by A and G.

The completion-boundary theorem then orders the readback: W and Q precede the seal E. Since H, C, P, N only transport, replay, source, and name the same route, the exported metric request remains a finite ledger rather than an ambient metric-completion object. □

Theorem 2043.7 (LocatedSpace stream readback stability)**.** *For every accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier, if a locatedness request is transported by* hsame *or replayed by* Cont*, then its* $\mathsf{StreamName}^{\uparrow}$ *window* W *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *remain attached to the same metric source* X*, finite request* R*, apartness gap* A*, gap budget* G*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$ *row therefore name a stable finite readback route and not a selected limit, quotient real equality, or ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier stream readback stability`

Proof. Begin with the accepted carrier and project the rows $X, R, A, G, W, Q, E, H, C, P, N$. The stream window and regular rational readback are explicit fields of the carrier, so they cannot be replaced by an unrecorded completion read.

Apply Definition 2043.2. Its locatedness-stability clause keeps X, R, A, G, W, Q fixed under transport and replay, and its completion-boundary nonescape clause requires E to be reached only after those rows are displayed.

Then apply Definition 2043.3. It identifies the same route to the real seal and excludes selected limits, quotient representatives, and ambient completeness. The provenance and local naming rows therefore certify stability of the finite readback route. □

Theorem 2043.8 (LocatedSpace positive-gap exactness)**.** *For every accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier*

$$L = (X, R, A, G, W, Q, E, H, C, P, N),$$

a positive locatedness request reaches the $\mathsf{Real}^{\uparrow}$ *seal* E *only after the positive* $\mathsf{ApartnessSpace}^{\uparrow}$ *gap row* A*, the metric-distance budget* G*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the displayed* $\mathsf{Real}^{\uparrow}$ *boundary row have all been exposed. The route exports no trichotomy, quotient real equality, selected representative, or ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.PositiveGapExactness.LocatedSpaceCarrier positive gap exactness`

Proof. Project the accepted carrier through Definition 2043.1. The positive locatedness request is displayed by the rows R, A, G, while W, Q, E are the only downstream rows capable of reading the finite observation and terminal real boundary.

Apply Definition 2043.4. It keeps the apartness-facing request scoped by X, R, A, G before any stream window, regular readback, or real seal can be consumed.

Apply Definition 2043.6. The metric-distance read is ledgered by the same $X, R, A, G, W, Q, E, H, C, P, N$ route, so the positive budget used by the apartness row is the budget read by the metric comparison and not an external order decision.

Finally use Definition 2043.7. Transport and replay preserve the same W, Q, E readback route, and the carrier contains no trichotomy, quotient equality, selected representative, or ambient completion coordinate. Thus the positive gap reaches the Real seal only through the displayed finite rows. □

Theorem 2043.9 (LocatedSpace obligation closure route)**.** *The accepted* $\mathsf{LocatedSpace}^{\uparrow}$ *carrier has the full obligation surface needed for* $\mathsf{seed} \to \mathsf{obligation}$*. The surface consists of the carrier and NameCert rows of Definition 2043.2, the exact positive-gap route of Definition 2043.5, the metric request ledger of Definition 2043.6, and the stream readback stability of Definition 2043.7; together these rows expose the source, request, gap, budget, window, regular readback, Real seal, transport, replay, provenance, and local naming before any locatedness consumer reads the packet.*

Lean4 ✓ `BEDC.Derived.LocatedSpaceUp.LocatedSpaceCarrier obligation closure route`

Proof. First project the NameCert obligations. They give carrier admission, locatedness stability, positive-gap exactness, completion-boundary nonescape, and ledger refusal over the displayed rows $X, R, A, G, W, Q, E, H, C, P, N$.

Next apply the positive-gap exactness route. It identifies A, G as the only finite gap surface available to metric, regular-rational, and Real-facing consumers, so the obligation surface cannot import a second apartness witness or host order decision.

Then use the metric request ledger and stream readback stability. The former orders every metric request through $X, R, A, G, W, Q, E, H, C, P, N$, while the latter keeps W, Q, E attached to the same source, request, gap, and budget under transport and replay.

These rows exhaust the obligation-level route. They upgrade the paper closure axis from seed to obligation without changing the formal verification status, since no checked Lean theorem or standard bridge is asserted here. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$, publicClosed)

Scope closed: Definition 2043.1, Definition 2043.2, Definition 2043.5, Definition 2043.6, Definition 2043.7, and Definition 2043.9 close the public paper-side export surface over positive-gap exactness, metric request ledgering, stream readback stability, and obligation closure for finite locatedness requests through BHist, hsame, Cont, Pkg, NameCert, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{ApartnessSpace}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected limits, open-cover compactness, choice, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Formal closure requires checked carrier admission, locatedness stability, positive-gap exactness, completion-boundary nonescape, and ledger refusal for BEDC.Derived.LocatedSpaceUp.

Constructive story: A $\mathsf{LocatedSpace}^{\uparrow}$ packet is generated from a metric source carrier, a finite locatedness request, a positive ApartnessSpace gap, a StreamName window, a RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2044

A Concrete Naming Certificate for $\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}$

Bishop–Bridges/Kelley-style constructive completion language treats the canonical passage from a Cauchy presentation into the completed ordered real surface as order-reflecting only through displayed regular tails, tolerance budgets, and real seals. The $\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}$ packet records that boundary as finite BEDC data: it reads the completion carrier of chapter 360, the constructive order route of chapter 1716, the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It exposes an ordered-completion interface for downstream consumers without quotient stream equality, selected representatives, trichotomy, ambient completeness, Lean verification, or bridge checking.

Definition 2044.1 (CauchyCompletionOrderEmbedding carrier)**.** *A* $\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, L, W, D, R, E, T, C, P, N)$$

with the following displayed rows:

1. *K is the source completion row, displaying the accepted* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *data whose two finite Cauchy names are being compared.*
2. *L is the source order row, displaying the* $\mathsf{CauchyRealOrder}^{\uparrow}$ *comparison that reads those names only through regular Cauchy evidence.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the order comparison and by the completion-facing real seal.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic budgets used to compare the displayed windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows and tolerance requests.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after K, L, W, D, R have been displayed.*
7. *T records componentwise* hsame *transport for the completion row, order row, stream windows, dyadic ledger, regular readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$K, L, W, D, R \longmapsto E \longmapsto T, P, N.$$

9. *P is the* Pkg *provenance over* CauchyCompletionOrderEmbedding$^{\uparrow}$, RealCauchyCompletion$^{\uparrow}$, CauchyRealOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchyCompletionOrderEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, L, W, D, R, E, T, C, P, N$. The carrier has no coordinate for quotient streams, selected representatives, trichotomy, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOrderEmbeddingUp.CauchyCompletionOrderEmbeddingTasteGate`
`single carrier alignment`

Theorem 2044.2 (CauchyCompletionOrderEmbedding NameCert obligations)**.** *For every accepted* CauchyCompletionOrderEmbedding$^{\uparrow}$ *carrier*

$$O = (K, L, W, D, R, E, T, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionOrderEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion row, order row, finite stream windows, dyadic tolerance ledger, regular rational readback, real seal, transport, replay, provenance, and local naming row;*
2. *completion-source exactness, requiring every order-facing real read to begin with the displayed* RealCauchyCompletion$^{\uparrow}$ *row K;*
3. *order-route exactness, requiring every comparison exported by the packet to pass through the displayed* CauchyRealOrder$^{\uparrow}$ *row L;*
4. *window and budget exposure, requiring the inspected* StreamName$^{\uparrow}$ *windows W, the* DyadicRatCore$^{\uparrow}$ *ledger D, and the* RegSeqRat$^{\uparrow}$ *readback R to be displayed before the* Real$^{\uparrow}$ *seal E is read;*
5. *provenance exhaustion, requiring T, C, P, N to transport, replay, source, and name only the listed route.*

No NameCert obligation is stored outside those rows, and none exports quotient streams, selected representatives, trichotomy, ambient completeness, Lean verification, or bridge checking.

Proof. Carrier admission is projection from Definition 2044.1. The accepted packet displays exactly the completion row, order row, window family, dyadic ledger, readback row, real seal, and structural rows.

Completion-source exactness follows from the placement of K at the start of every replay route. A downstream order-facing read cannot bypass the accepted completion row because no second completion coordinate appears in the carrier.

Order-route exactness is supplied by L. The only comparison row available to the classifier is the displayed CauchyRealOrder$^{\uparrow}$ row, and it reads the finite windows, dyadic budgets, and regular rational observations named in W, D, R.

The real-seal boundary is then a finite exposure check. The seal E is downstream of K, L, W, D, R, while T, C, P, N preserve and name the same route without adding mathematical coordinates. Thus the excluded quotient, representative, trichotomy, completeness, verification, and bridge principles have no row from which a consumer could read them. □

Theorem 2044.3 (CauchyCompletionOrderEmbedding order reflection)**.** *Let $O = (K, L, W, D, R, E, T, C, P, N)$ be an accepted* CauchyCompletionOrderEmbedding$^{\uparrow}$ *packet. Any finite* Real$^{\uparrow}$*-sealed comparison exported by O reflects back to the displayed source order row L: if the completed real seal is read as*

an ordered comparison, then the route factors through

$$K \longmapsto L \longmapsto (W, D, R) \longmapsto E.$$

The reflection is a packet-local route through displayed Cauchy data, not quotient stream equality, not a selected representative comparison, not trichotomy, and not ambient completeness.

Proof. Start with a completed real-seal comparison exported by O. Projection from the carrier gives the only seal row E and the only order row L available to that comparison.

The replay row C records the order of inspection. Before E can be read as an ordered real observation, the completion row K has been displayed, the comparison has passed through L, and the windows, tolerance ledger, and readback rows W, D, R have been opened.

Transport does not add a second comparison route. The row T moves the same displayed rows componentwise, and P, N source and name the same packet. Hence every completed ordered comparison reflects to the source order row L through finite Cauchy data alone. □

Theorem 2044.4 (CauchyCompletionOrderEmbedding real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}$ *carrier receives the seal row E only after the completion row K, the order row L, the stream-window rows W, the dyadic tolerance ledger D, and the regular readback row R have been displayed. Consequently the consumer receives an ordered-completion route and receives no quotient stream, selected representative, trichotomy witness, ambient completeness theorem, Lean verification, or bridge credential from the packet.*

Proof. Let a $\mathsf{Real}^{\uparrow}$ consumer read the accepted packet. The carrier inventory gives a single seal row E and identifies its upstream route as K, L, W, D, R.

The finite observation rows are inspected first. The stream windows W provide the displayed name observations, the dyadic ledger D supplies the tolerance budgets, and the readback row R records the rational observations consumed by the order row L.

Only then is the seal exposed. Since E is downstream of the completion and order rows, a consumer sees the real seal as part of the ordered-completion route rather than as an independent host real.

The structural rows close the boundary. Componentwise transport, continuation replay, package provenance, and the local name preserve the same coordinates, so the packet cannot export any of the excluded quotient, representative, trichotomy, completeness, verification, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2044.1, Definition 2044.2, Definition 2044.3, and Definition 2044.4 seed the finite ordered-completion route from displayed Cauchy completion data to the real-order seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-source exactness, order-route exactness, window and budget exposure, provenance exhaustion, order reflection, and real-seal boundary for BEDC.Derived.CauchyCompletionOrderE

Constructive story: A $\mathsf{CauchyCompletionOrderEmbedding}^{\uparrow}$ packet is generated from a RealCauchyCompletion row, a CauchyRealOrder row, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2045

A Concrete Naming Certificate for $\mathsf{AlexandroffCompactification}^{\uparrow}$

Following Alexandroff, Bourbaki, and Kelley, the one-point compactification of a locally compact Hausdorff space is recorded here as a finite BEDC boundary packet rather than as an imported host-space construction. The $\mathsf{AlexandroffCompactification}^{\uparrow}$ carrier reads the topology surface of chapter 88, the locally compact neighborhood data of chapter 1726, the Hausdorff separation surface of chapter 1925, and the finite compactness interface of chapter 60. It names the adjoined point and the compact-complement neighborhood ledger that compactification consumers need, while leaving open-cover compactness, quotient topology, selected representatives, choice, ambient completeness, Lean verification, and bridge checking outside the packet.

Definition 2045.1 (AlexandroffCompactification carrier)**.** *An* $\mathsf{AlexandroffCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, T, L, S, \infty, O, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying source carrier whose points are being compactified.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row on X, exposing only the ordinary open rows that may be reused away from the adjoined point.*
3. *L is the* $\mathsf{LocallyCompact}^{\uparrow}$ *row, displaying the finite local compact-neighborhood witnesses needed before any infinity-neighborhood row may be opened.*
4. *S is the* $\mathsf{HausdorffSpace}^{\uparrow}$ *row, displaying the separated-neighborhood ledger used to keep ordinary points apart from compact-complement boundary reads.*
5. *∞ is the named adjoined point row, stored as a new BEDC name rather than as a host representative or quotient class.*
6. *O is the ordinary-open ledger: a finite family of rows saying that a displayed open row in the compactified packet whose carrier avoids ∞ is read exactly through T.*
7. *K is the compact-complement infinity-neighborhood ledger: each row consists of a displayed compact witness from the* $\mathsf{Compact}^{\uparrow}$ *surface, its complement request, and the assertion that the resulting neighborhood contains ∞.*

8. *H records componentwise* hsame *transport for the source carrier, topology row, local compactness row, Hausdorff row, infinity name, ordinary-open ledger, compact-complement ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of every compactification-facing route through*

$$(X, T, L, S) \longmapsto (O, K, \infty) \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* AlexandroffCompactification$^{\uparrow}$, Topology$^{\uparrow}$, LocallyCompact$^{\uparrow}$, HausdorffSpace$^{\uparrow}$, Compact$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{AlexandroffCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, L, S, \infty, O, K, H, C, P, N$. There is no coordinate for open-cover compactness, quotient topology, selected representatives, choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.AlexandroffCompactificationUp.AlexandroffCompactificationTasteGate single carrier alignment`

Theorem 2045.2 (AlexandroffCompactification NameCert obligations)**.** *For every accepted* AlexandroffCompactific *carrier*

$$A = (X, T, L, S, \infty, O, K, H, C, P, N),$$

the local NameCert$_{\text{AlexandroffCompactification}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source carrier, topology row, locally compact row, Hausdorff row, infinity name, ordinary-open ledger, compact-complement ledger, transport, replay, provenance, and local naming row;*

2. *source topology exactness, requiring every ordinary open read in O to factor through the displayed* Topology$^{\uparrow}$ *row T on X;*

3. *local compactness exposure, requiring every infinity-neighborhood row in K to open the displayed* LocallyCompact$^{\uparrow}$ *row L before it may read a compact witness;*

4. *Hausdorff separation compatibility, requiring ordinary-point separation and compact-complement boundary reads to use the displayed* HausdorffSpace$^{\uparrow}$ *row S and no host point equality;*

5. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed route.*

No NameCert obligation is stored outside these rows, and none exports open-cover compactness, quotient topology, selected representatives, choice, ambient completeness, Lean verification, or bridge checking.

Proof. Carrier admission is projection from Definition 2045.1. The accepted packet displays the source carrier, topology row, local compactness row, Hausdorff row, infinity name, the two open-neighborhood ledgers, and the structural rows.

Source topology exactness follows from the role of O. An ordinary open row that does not contain ∞ is replayed through the displayed topology row T, and the carrier contains no second topology coordinate.

The infinity-neighborhood obligation is the role of K. Each row in K first reads local compactness through L, then names a compact witness through the Compact$^{\uparrow}$ surface, and only then records the compact-complement neighborhood containing ∞.

Hausdorff compatibility is supplied by S. The separated-neighborhood data are transported componentwise by H and replayed by C, so no host point equality or selected representative can be inserted into the route.

Finally, P and N source and name the same finite route. Since the carrier has no row for the excluded principles, the displayed rows exhaust the local NameCert surface. □

Theorem 2045.3 (AlexandroffCompactification neighborhood classifier)**.** *Let $A = (X, T, L, S, \infty, O, K, H, C, P, N$ be an accepted* $\mathsf{AlexandroffCompactification}^{\uparrow}$ *carrier. Every displayed compactification-neighborhood read is classified by exactly one of the following packet-local routes:*

1. *an ordinary-open route through O and the source topology row T, with no read of the infinity point;*
2. *an infinity-neighborhood route through K, beginning with the adjoined point row ∞ and continuing through the local compactness row L and a compact-complement witness from* $\mathsf{Compact}^{\uparrow}$.

Both routes are stable under componentwise hsame *transport and displayed* Cont *replay, and neither route exports quotient topology, selected representatives, choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Take an accepted carrier and a displayed neighborhood read. Projection from the carrier gives only the ordinary-open ledger O and the compact-complement ledger K as neighborhood sources.

If the read avoids ∞, the classifier uses the first route. The row in O points back to T, so the neighborhood is an ordinary source-topology row and no compact-complement data are inspected.

If the read contains ∞, the classifier uses the second route. The row in K names the adjoined point, opens local compactness through L, and records the compact witness whose complement supplies the infinity neighborhood.

The transport and replay rows do not add a third route. They preserve the same displayed coordinates, while P and N source and name the packet. The excluded quotient, representative, choice, completeness, verification, and bridge data are absent from the carrier. □

Theorem 2045.4 (AlexandroffCompactification compact boundary ledger)**.** *Every compact-facing consumer of an accepted* $\mathsf{AlexandroffCompactification}^{\uparrow}$ *carrier receives only the finite boundary ledger*

$$(T, L, S, \infty, O, K, H, C, P, N).$$

The ordinary side of the ledger reads source opens through T, and the infinity side reads compact-complement neighborhoods through L and the $\mathsf{Compact}^{\uparrow}$ *witness rows. The handoff therefore exposes a concrete compactification boundary without claiming open-cover compactness, quotient topology, selected representatives, choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Let a compact-facing consumer read the accepted packet. The carrier inventory identifies the only boundary rows available to the consumer: the source topology, locally compact and Hausdorff support, the adjoined point, the two neighborhood ledgers, and the structural rows.

On the ordinary side, the consumer reads an O row and thereby returns to the source topology T. This provides ordinary opens over X without creating a new host-space topology object.

On the infinity side, the consumer reads a K row. That row passes through the locally compact witness L and a compact witness from the $\mathsf{Compact}^{\uparrow}$ surface before exposing the complement as a neighborhood of ∞.

The remaining rows only preserve the ledger. Componentwise transport, continuation replay, package provenance, and local naming record the same route, so the handoff contains no row for

the excluded compactness, quotient, representative, choice, completeness, verification, or bridge principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandroffCompactification}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandroffCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2045.1, Definition 2045.2, Definition 2045.3, and Definition 2045.4 seed the finite one-point compactification route from locally compact Hausdorff source data to a compact-complement boundary ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandroffCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient topology, selected representatives, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source topology exactness, local compactness exposure, Hausdorff separation compatibility, provenance exhaustion, neighborhood classification, and compact boundary ledger coverage for BEDC.Derived.AlexandroffCompactificationUp.

Constructive story: An $\mathsf{AlexandroffCompactification}^{\uparrow}$ packet is generated from a source carrier, a Topology row, a LocallyCompact row, a HausdorffSpace row, an adjoined infinity name, ordinary-open rows, compact-complement infinity-neighborhood rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2046

A Concrete Naming Certificate for $\mathsf{AlexandrovCompactification}^{\uparrow}$

From Alexandroff/Engelking-style one-point compactification, the $\mathsf{AlexandrovCompactification}^{\uparrow}$ packet records the finite route by which locally compact Hausdorff data are read as a compact one-point boundary without importing a host compact space. It consumes the local compactness interface of chapter 1726, the separation rows of chapter 1925, the compact certificate boundary of chapter 60, and the topology ledger of chapter 88. Downstream compactness consumers see only the displayed carrier, the added point, compact-complement neighbourhood rows, Hausdorff separation rows, topology transport rows, continuation routes, provenance, and a local NameCert.

Definition 2046.1 (AlexandrovCompactification carrier)**.** *An* $\mathsf{AlexandrovCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, \infty, K, S, T, C, P, N)$$

with the following displayed rows:

1. *X is the locally compact Hausdorff source carrier whose finite open, compact, and separation data are being compactified.*
2. *∞ is the distinguished boundary point added to X; it is a named point row, not a quotient class and not a selected limit.*
3. *K is the compact-complement neighbourhood ledger. Each infinity-neighbourhood request is displayed as a finite row naming an open neighbourhood of ∞ by a compact subset of X whose complement is exposed as the neighbourhood trace on X.*
4. *S is the Hausdorff separation ledger. It records point–point separation inside X, point–boundary separation between $x \in X$ and ∞, and the compatibility of those rows with the compact-complement ledger.*
5. *T records topology transport rows for source opens, infinity-neighbourhood rows, finite intersections, finite unions, and componentwise* hsame *movement of the carrier, boundary point, compact-complement ledger, separation ledger, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of every compactification-facing consumer route through*

$$X, \infty, K, S, T \longmapsto P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{AlexandrovCompactification}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexandrovCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \infty, K, S, T, C, P, N$. It has no coordinate for open-cover compactness, host topological compactness, choice, quotient equality, selected limits, Lean verification, or a standard bridge.

Lean4 ✓ `BEDC.Derived.AlexandrovCompactificationUp.AlexandrovCompactificationTasteGate single carrier alignment`

Theorem 2046.2 (AlexandrovCompactification NameCert obligations)**.** *For every accepted* AlexandrovCompactific *carrier*

$$A = (X, \infty, K, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AlexandrovCompactification}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the locally compact Hausdorff source X, the added boundary point ∞, the compact-complement ledger K, the separation ledger S, topology transport T, continuation replay C, provenance P, and local naming row N;*
2. *source exactness, requiring every compactification read to begin with the displayed locally compact Hausdorff carrier X and not with an ambient compact host;*
3. *boundary-neighbourhood exactness, requiring every neighbourhood of ∞ to be read through a displayed compact-complement row of K;*
4. *Hausdorff compatibility, requiring point–point and point–boundary separation reads to pass through S and to preserve the same compact-complement row used by the boundary neighbourhood;*
5. *ledger exhaustion, requiring T, C, P, N to transport, replay, source, and name only the listed rows, with no open-cover compactness, host topological compactness, choice, quotient equality, selected limit, Lean verification, or bridge credential.*

No AlexandrovCompactification NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2046.1. The accepted packet displays the source carrier, boundary point, compact-complement ledger, separation ledger, topology transport rows, continuation replay, provenance, and local name.

Source exactness is the placement of X at the start of every replay route. Since the packet has no compact host coordinate, a compactification-facing read cannot replace the locally compact Hausdorff source by an already compact ambient space.

Boundary-neighbourhood exactness is the role of K. A neighbourhood of ∞ is admitted only as a compact-complement row, so its trace on X is read from the displayed compact subset and its complement rather than from an open-cover compactness principle.

Hausdorff compatibility is supplied by S. Point–point separation remains inside the displayed source, while point–boundary separation shares the same compact-complement row used by the infinity-neighbourhood request.

Finally, T, C, P, N only transport, replay, source, and name the displayed rows. Because the excluded compactness, choice, quotient, selected-limit, verification, and bridge data are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2046.3 (AlexandrovCompactification infinity filter boundary)**.** *Let $A = (X, \infty, K, S, T, C, P, N)$ be an accepted* $\mathsf{AlexandrovCompactification}^{\uparrow}$ *carrier. Every read of the neighbourhood filter at ∞ factors through a displayed compact-complement row of K. The factorization uses the route*

$$X, \infty, K \longmapsto S, T \longmapsto C, P, N$$

and does not factor through open-cover compactness, choice, quotient equality, selected limits, or host topological compactness.

Proof. Start with an infinity-neighbourhood read of the accepted packet. Projection from Definition 2046.1 gives the boundary point ∞ and the compact-complement ledger K as the only neighbourhood data attached to that point.

Opening the read selects a displayed row of K. That row names the compact subset of X and the complementary trace on X that defines the neighbourhood of ∞ inside the packet.

The separation and topology rows then preserve this same compact-complement row. The Hausdorff ledger S supplies the point–boundary separation route, and T transports finite open, union, and intersection reads without adding a second compactness source.

The continuation and provenance rows close the route. Since C, P, N replay, source, and name only the displayed rows, the infinity filter cannot be read through open-cover compactness, choice, quotient equality, selected limits, or host topological compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlexandrovCompactification}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlexandrovCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2046.1, Definition 2046.2, and Definition 2046.3 seed the finite one-point compactification route from locally compact Hausdorff source data to the compact-complement boundary filter.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlexandrovCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host topological compactness, choice, quotient equality, selected limits, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source exactness, boundary-neighbourhood exactness, Hausdorff compatibility, ledger exhaustion, and infinity-filter boundary for BEDC.Derived.AlexandrovCompactificationUp.

Constructive story: An $\mathsf{AlexandrovCompactification}^{\uparrow}$ packet is generated from a locally compact Hausdorff carrier, a named boundary point, a compact-complement neighbourhood ledger, Hausdorff separation rows, topology transport rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2047

A Concrete Naming Certificate for $\mathsf{SeriesDivergenceTest}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the packet records the nth-term divergence test as a finite BEDC refusal boundary: a displayed nonzero tail of a $\mathsf{Series}^{\uparrow}$ source prevents a $\mathsf{RegSeqRat}^{\uparrow}$ handoff and hence prevents a $\mathsf{Real}^{\uparrow}$ seal. It reads the series source of chapter 121, the Cauchy-facing obstruction of chapter 392, the finite windows of chapter 126, the dyadic comparison rows of chapter 291, the regular-rational handoff boundary of chapter 301, and the real seal of chapter 38. The packet names a necessary-condition refusal; it does not import quotient series equality, selected sums, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2047.1 (Series divergence test carrier)**.** *A* $\mathsf{SeriesDivergenceTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Series}^{\uparrow}$ *source, read only through the finite term observations admitted by the series packet.*
2. *W is the finite tail-window ledger, recording the selected later windows in which the necessary tail-zero condition is being tested.*
3. *D is the displayed nonzero or apartness row, carrying a later term observation that remains apart from zero inside the accepted tail window.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparison, recording the dyadic zero-neighborhood, apartness witness, and finite comparison budget used by D.*
5. *R is the refused* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff boundary, exposed exactly where a regular Cauchy consumer would have to discharge the tail-zero condition.*
6. *E is the refused* $\mathsf{Real}^{\uparrow}$ *seal boundary, downstream of R and available only as a named obstruction to a real endpoint seal.*
7. *H stores componentwise* hsame *transport for the source, tail-window ledger, nonzero row, dyadic comparison, refused regular handoff, refused real seal, replay, provenance, and local naming rows.*

8. *C stores displayed* Cont *replay from the series source to the tail windows, from the tail windows to the nonzero row and dyadic comparison, from that comparison to the refused regular handoff, and from the handoff boundary to the refused real seal boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{SeriesDivergenceTest}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeriesDivergenceTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, D, T, R, E, H, C, P, N$. It has no coordinate for selected sums, quotient series equality, host real convergence, countable-choice selectors, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.SeriesDivergenceTestUp.SeriesDivergenceTestTasteGate single carrier alignment`

Theorem 2047.2 (Series divergence test NameCert obligations)**.** *For every accepted* $\mathsf{SeriesDivergenceTest}^{\uparrow}$ *carrier $S = (A, W, D, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeriesDivergenceTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed series source, tail-window ledger, nonzero tail row, dyadic comparison, refused regular handoff boundary, refused real seal boundary, transport, replay, provenance, and local naming rows;*

2. *tail-window exposure, requiring every later term inspected by the refusal to pass through A and W before any handoff boundary is read;*

3. *nonzero-tail stability, requiring D and T to display the same later apartness witness under the accepted dyadic zero-neighborhood comparison;*

4. $\mathsf{RegSeqRat}^{\uparrow}$ *refusal, requiring R to be reached exactly at the point where a regular Cauchy handoff would need tail-zero evidence;*

5. $\mathsf{Real}^{\uparrow}$ *nonescape, requiring E to remain downstream of the refused regular handoff and to export no selected sum, quotient series equality, host real convergence, countable choice, trichotomy, ambient completeness, Lean verification, or bridge datum;*

6. *componentwise transport, displayed replay, provenance, and local naming for the same refusal route through H, C, P, N.*

No series divergence test obligation is stored outside these rows.

Proof. Project the accepted packet $S = (A, W, D, T, R, E, H, C, P, N)$. The carrier definition exposes exactly the series source, tail-window ledger, displayed nonzero row, dyadic comparison, two refusal boundaries, and structural rows.

The tail-window exposure row forces every later term used by the test to be read through A and W. The nonzero-tail stability row then couples D to the dyadic comparison T, so the displayed apartness witness is not detached from the tolerance budget that made it visible.

At the regular handoff boundary, the route reaches R. By the obligation surface, a consumer can pass this point only by supplying the missing tail-zero condition; the displayed nonzero row blocks that discharge inside the same finite window.

Finally E, H, C, P, N preserve, replay, source, and name the same refusal route. Since no excluded analytic or choice object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2047.3 (Series divergence test nonzero-tail refusal)**.** *Let $S = (A, W, D, T, R, E, H, C, P, N)$ be an accepted* $\mathsf{SeriesDivergenceTest}^{\uparrow}$ *carrier. Any consumer attempting a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff or a* $\mathsf{Real}^{\uparrow}$ *seal from the source A must first discharge the tail-zero condition for the displayed later windows. The nonzero tail row D, measured by the dyadic comparison T, blocks that route without quotient series equality, selected sums, countable choice, trichotomy, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer request for a regular handoff or real seal. The carrier exposes no direct endpoint coordinate from the series source; the only available route is

$$A \longmapsto W \longmapsto D, T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The request therefore reaches the tail-window ledger W before it reaches either refusal boundary. To continue toward R, it must turn the inspected later windows into the tail-zero evidence required by a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The displayed row D supplies an apartness witness for one of those later terms, and T records the dyadic comparison that keeps the witness visible. Thus the required tail-zero discharge is unavailable inside the accepted packet.

Since the attempted $\mathsf{Real}^{\uparrow}$ seal E is downstream of the refused regular handoff, the same obstruction prevents endpoint sealing. The carrier has no slot for quotient series equality, selected sums, countable choice, trichotomy, host real equality, or ambient completeness, so none of those devices can bypass the finite refusal row. □

Falsifiable prediction: Every $\mathsf{SeriesDivergenceTest}^{\uparrow}$ endpoint-facing request factors through a SeriesUp source, finite tail-window ledger, displayed nonzero tail row, DyadicRatCore comparison, refused RegSeqRat handoff boundary, refused RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: series, cauchycriterion, streamname, dyadicratcore, regseqrat, real: $\mathsf{SeriesDivergenceTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source series row, finite tail windows, a displayed nonzero tail obstruction, dyadic apartness accounting, refused regular and real seal boundaries, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SeriesDivergenceTest}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeriesDivergenceTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2047.1, Definition 2047.2, and Definition 2047.3 seed the finite constructive nth-term divergence refusal route from displayed nonzero tail data to RegSeqRat and RealUp handoff denial.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeriesDivergenceTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, host real convergence, countable choice, trichotomy, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exposure, nonzero-tail stability, RegSeqRat refusal, RealUp nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SeriesDivergenceTestUp.

Constructive story: A $\mathsf{SeriesDivergenceTest}^{\uparrow}$ packet is generated from a Series source, finite tail-window ledger, displayed nonzero tail row, DyadicRatCore comparison, refused RegSeqRat handoff boundary, refused RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2048

A Concrete Naming Certificate for $\mathsf{TotalBoundedMetric}^{\uparrow}$

Bishop–Bridges constructive metric compactness treats total boundedness as a tolerance-indexed finite-net interface attached to a displayed metric source. The $\mathsf{TotalBoundedMetric}^{\uparrow}$ packet records exactly that L10 boundary: it reads the metric carrier of chapter 57, the real metric comparison surface of chapter 1638, the finite epsilon-net ledger of chapter 1767, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the compact-metric consumer of chapter 129. The packet gives compact and completion-facing consumers a finite total-bounded metric package while leaving open-cover compactness, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, and bridge checking outside the carrier.

Definition 2048.1 (TotallyBoundedMetric carrier)**.** *A* $\mathsf{TotallyBoundedMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, R, E, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier, distance classifier, symmetry, zero-distance, and triangle ledgers read by the packet.*
2. *R is the* $\mathsf{RealMetric}^{\uparrow}$ *comparison row, exposing the real-valued metric readback only through displayed rational or dyadic observations.*
3. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *family row. For each positive tolerance request it displays a closed* $\mathsf{ProbeBundle}$ *of centers, membership rows, center-carrier rows, and coverage-distance witnesses.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic refinement and positivity data used by the finite-net family.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the source points and center names that enter a coverage claim.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations attached to distances, tolerances, and finite-net comparisons.*
7. *H records componentwise* hsame *transport for the metric source, real metric row, epsilon-net family, dyadic tolerance ledger, stream windows, regular rational readback, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay through*

$$M, R, D, S, Q \longmapsto E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{TotalBoundedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TotalBoundedMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, R, E, D, S, Q, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 (definition) `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotallyBoundedMetricUp`
Lean4 ✓ `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotallyBoundedMetricTasteGate single carrier alignment`

Theorem 2048.2 (TotallyBoundedMetric NameCert obligations)**.** *For every accepted* $\mathsf{TotallyBoundedMetric}^{\uparrow}$ *carrier*

$$T = (M, R, E, D, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TotallyBoundedMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, real metric comparison row, finite epsilon-net family, dyadic tolerance ledger, stream windows, regular rational readback, transport, replay, provenance, and local naming row;*
2. *metric-source exactness, requiring every distance and coverage row to pass through the displayed* $\mathsf{Met}^{\uparrow}$ *row M and the real metric comparison row R;*
3. *tolerance visibility, requiring each requested radius to be carried by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D before the finite-net row is read;*
4. *finite-net exposure, requiring every center and every coverage witness to enter through the displayed* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *family E;*
5. *window and readback compatibility, requiring the inspected point windows S and rational observations Q to be preserved by the same* hsame *transport and* Cont *replay rows;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed route.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotalBoundedMetricCarrier namecert obligations`
↪ variant `BEDC.Derived.TotalBoundedMetricUp.TotalBoundedMetricCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 2048.1. The accepted packet displays the metric source M, real metric row R, finite epsilon-net family E, tolerance ledger D, stream windows S, rational readback Q, and structural rows.

Metric-source exactness follows from the placement of M and R at the start of the replay route. A coverage distance cannot be read without first opening the point and distance classifiers of M and the displayed real metric comparison row R.

Tolerance and finite-net exposure are rowwise. The dyadic ledger D supplies the positive tolerance data, and E supplies the closed center bundle, center membership rows, and coverage-distance witnesses for that tolerance.

The window and readback rows keep the route finite. The row S exposes only the inspected source and center windows, while Q records the rational observations attached to the distance and tolerance comparisons.

Finally, H, C, P, N preserve, replay, source, and name the same row inventory. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export open-cover compactness, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking. □

Theorem 2048.3 (TotallyBoundedMetric finite-net factorization)**.** *Every compact or completion-facing consumer of an accepted* $\mathsf{TotallyBoundedMetric}^{\uparrow}$ *carrier reads total boundedness through the finite route*

$$M, R, D \longmapsto E \longmapsto S, Q \longmapsto H, C, P, N.$$

Thus each total-bounded metric claim first opens the metric source, real metric comparison row, and dyadic tolerance ledger, then a displayed $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *family of center bundles and coverage rows, and only then the stream-window, rational-readback, transport, replay, provenance, and local naming rows.*

Proof. Let $T = (M, R, E, D, S, Q, H, C, P, N)$ be accepted. A compact or completion-facing consumer enters the packet through the metric source M, because no coverage distance is meaningful before the point and distance classifiers have been displayed.

The real metric row R and dyadic tolerance row D are inspected next. They supply the comparison and positive-radius data required by the finite-net family.

The finite-net factor is exactly E. For the requested tolerance, E exposes a closed $\mathsf{ProbeBundle}$ of centers, center membership rows, center-carrier rows, and coverage-distance witnesses.

The readback route then remains finite. The stream windows S and regular rational observations Q record the inspected point and distance data, while H, C, P, N transport, replay, source, and name the same package without adding a second compactness route.

Therefore every exported total-bounded metric claim factors through the displayed finite-net route. The consumer receives no open-cover compactness, countable choice, selected limit, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification, or bridge credential from the packet. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotallyBoundedMetricCarrier finite net factorization`

Theorem 2048.4 (Total-bounded metric finite-net obligation)**.** *Let* $T = (M, R, E, D, S, Q, H, C, P, N)$ *be an accepted* $\mathsf{TotalBoundedMetric}^{\uparrow}$ *carrier. For every positive dyadic tolerance admitted by* D*, the finite-net obligation exposed by the local* $\mathsf{NameCert}_{\mathsf{TotalBoundedMetric}^{\uparrow}}$ *surface is exactly the closed* $\mathsf{ProbeBundle}$ *branch of* E *together with its center-membership rows, center-carrier rows over* M*, coverage-distance witnesses read through* R*, finite observation windows* S*, regular rational readback* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, provenance* P*, and local name* N*.*

In particular, a $\mathsf{CompactMetric}^{\uparrow}$ *or compact-uniform-continuity consumer can read a finite net only through the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{ProbeBundle}, \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{C}$$

The obligation exports no selected finite subcover, host enumeration of centers, open-cover compactness theorem, countable-choice schedule, or detached ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier. The tolerance request first reaches the dyadic ledger D, so positivity, refinement, and radius comparison are visible before any center bundle is consumed.

The finite-net row is then forced to be E. Its closed ProbeBundle branch supplies the center rows, the InBundle membership constructors, and the coverage witnesses for the displayed tolerance.

The metric and real reads are not independent coordinates. The center-carrier rows are read over M, coverage distances pass through R, and the finite observations and rational comparisons are recorded by S and Q.

Finally H, C, P, N transport, replay, source, and name the same route. Since no selected subcover, host enumeration, open-cover compactness theorem, choice schedule, or detached completeness principle occurs as a carrier coordinate, the listed finite route exhausts the obligation. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotalBoundedMetricCarrier finite net obligation`

Theorem 2048.5 (Total-bounded metric Cauchy-cover obligation)**.** *Let $T = (M, R, E, D, S, Q, H, C, P, N)$ be an accepted* TotalBoundedMetric$^{\uparrow}$ *carrier. Every Cauchy-cover read used by a* CompleteMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, *or compact-uniform-continuity route is obtained by replaying the same finite-net data at the requested dyadic tolerance: metric source M, real-distance comparison R, tolerance ledger D, finite center bundle E, observation windows S, readback row Q, and structural rows H, C, P, N.*

Thus the Cauchy-cover interface is a finite BEDC dependency between the Real and Completion phase and the compact-metric phase. It does not import a completed compact space, an open-cover proof, a selected convergent subsequence, a hidden dense enumeration, quotient equality of Cauchy names, countable choice, or host real completeness.

Proof. Begin with a Cauchy-cover request. The only metric source available to it is M, and the only real-distance comparison row available to it is R.

The requested cover scale is supplied by D. At that scale, Definition 2048.4 identifies E as the unique finite-net row through which centers, membership constructors, and coverage witnesses can be read.

The observation data remains finite because S stores the inspected source and center windows, while Q stores the regular rational readbacks used by the distance and tolerance comparisons.

The structural rows H, C, P, N then replay the same data for completion, compact-metric, and compact-uniform-continuity consumers. None of the excluded compactness, subsequence, dense-enumeration, quotient, choice, or host-completeness objects has a carrier coordinate, so no Cauchy-cover read can depend on it. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedMetricUp.TasteGate.TotallyBoundedMetricCarrier cauchy cover obligation`

Theorem 2048.6 (TotallyBoundedMetric Cauchy-window extraction)**.** *Let $T = (M, R, E, D, S, Q, H, C, P, N)$ be an accepted* TotallyBoundedMetric$^{\uparrow}$ *carrier. Any Cauchy-window request answered by the packet must expose a* FiniteEpsilonNet$^{\uparrow}$ *row E, a* DyadicRatCore$^{\uparrow}$ *tolerance row D, finite* StreamName$^{\uparrow}$ *observation windows S, and* RegSeqRat$^{\uparrow}$ *readback Q before any compact or completion-facing consumer reads the result. The extracted route is finite and exports no open-cover compactness, selected limit, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2048.1. The metric source M and real metric comparison row R give the source of the Cauchy-window request, while D, E, S, Q are the only tolerance, finite-net, observation-window, and rational-readback rows.

By Definition 2048.2, tolerance visibility requires the dyadic row D before a radius-dependent finite net is read. Finite-net exposure then requires the center family and coverage witnesses to enter through E.

By Definition 2048.3, compact and completion-facing consumers receive total boundedness only through the finite route $M, R, D, E, S, Q, H, C, P, N$. The StreamName windows S and RegSeqRat readback Q keep the Cauchy-window answer attached to displayed observations.

The structural rows H, C, P, N preserve and name the same route. Since the carrier has no coordinate for open-cover compactness, selected limits, countable choice, or ambient Real completeness, none of those principles can be read from the extracted Cauchy window. □

Lean4 ✓ `BEDC.Derived.TotalBoundedMetricUp.TotalBoundedMetricCarrier cauchy window extraction`

Closure status for $\mathsf{NameCert}_{\mathsf{TotalBoundedMetric}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalBoundedMetric}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2048.1, Definition 2048.2, Definition 2048.3, Definition 2048.4, and Definition 2048.5 seed the finite total-bounded metric package over displayed metric, real metric, finite-net, dyadic tolerance, stream-window, regular rational, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalBoundedMetric}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exactness, tolerance visibility, finite-net exposure, window and readback compatibility, provenance exhaustion, Cauchy-cover replay, and finite-net factorization for BEDC.Derived.TotalBoundedMetricUp.

Constructive story: A $\mathsf{TotalBoundedMetric}^{\uparrow}$ packet is generated from a MetricSpaceUp source row, a RealMetricUp comparison row, a FiniteEpsilonNetUp family, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2049

A Concrete Naming Certificate for $\mathsf{OnePointCompactification}^{\uparrow}$

Following Alexandroff compactification and Kelley topology, the $\mathsf{OnePointCompactification}^{\uparrow}$ packet records the one-point compactification boundary of a displayed locally compact metric source as finite BEDC NameCert data. It reads the local compact-neighbourhood surface of chapter 1726, the radius-window closed-ball ledger of **??**, the center-radius membership row of chapter 1451, the compact witness surface of chapter 129, and the open-index handoff of chapter 88. The packet names the added infinity row and the compact-complement neighbourhood route without importing open-cover compactness or a host compactification space.

Definition 2049.1 (OnePointCompactification carrier)**.** *A* $\mathsf{OnePointCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, L, B, \infty, Q, T, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the locally compact source row. It supplies the point classifier, local compact-neighbourhood windows, and source transport through the* $\mathsf{LocallyCompact}^{\uparrow}$ *surface.*
2. *L is the proper or closed-ball neighbourhood ledger. It records the displayed* $\mathsf{ProperMetric}^{\uparrow}$ *radius-window rows and the* $\mathsf{MetricClosedBall}^{\uparrow}$ *membership rows used to present compact complements around infinity.*
3. *B is the closed-ball boundary row. It exposes only center-radius non-strict membership comparisons already admitted by* $\mathsf{MetricClosedBall}^{\uparrow}$*.*
4. *∞ is the added infinity point row, disjoint from the displayed source-point rows by the local classifier and available only through the compactification packet.*
5. *Q is the compact-complement row. A neighbourhood of ∞ is read as the complement of a displayed compact closed-ball or compact metric window, with the compact witness carried by the rows L and B.*
6. *T is the topology handoff row. It sends source-open rows and infinity-neighbourhood rows into the indexed open surface of* $\mathsf{Topology}^{\uparrow}$ *without adding a second open-set carrier.*
7. *K is the compact-metric handoff row. It records that compactness consumers of the displayed complement window read the* $\mathsf{CompactMetric}^{\uparrow}$ *witness already attached to L and B.*

8. *H records componentwise* hsame *transport for the source row, neighbourhood ledger, closed-ball row, infinity point row, compact-complement row, topology handoff, compact-metric handoff, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay through*

$$X, L, B, \infty \longmapsto Q \longmapsto T, K \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{OnePointCompactification}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OnePointCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, L, B, \infty, Q, T, K, H, C, P, N$. It has no coordinate for open-cover compactness, a host compactification, selected points, quotient spaces, choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.OnePointCompactificationUp.OnePointCompactificationTasteGate single carrier alignment`

Theorem 2049.2 (OnePointCompactification NameCert obligations)**.** *For every accepted* OnePointCompactificatio *carrier*

$$A = (X, L, B, \infty, Q, T, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OnePointCompactification}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the locally compact source, proper or closed-ball neighbourhood ledger, closed-ball boundary row, infinity point row, compact-complement row, topology handoff, compact-metric handoff, transport, replay, provenance, and local naming row;*

2. *source-local exactness, requiring every ordinary-point neighbourhood read to factor through X and the cited* $\mathsf{LocallyCompact}^{\uparrow}$ *neighbourhood surface;*

3. *compact-complement exactness, requiring every infinity-neighbourhood read to open L and B before it reads the compact-complement row Q;*

4. *topology handoff, requiring the ordinary and infinity neighbourhood rows to be exported only through the indexed open surface T of* $\mathsf{Topology}^{\uparrow}$*;*

5. *compact-metric handoff, requiring compactness consumers to read only the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *witness row K attached to the same compact complement window;*

6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed route.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2049.1. The accepted packet displays the locally compact source X, the proper or closed-ball ledger L, the closed-ball row B, the infinity point row, the compact-complement row Q, the topology and compact-metric handoffs T, K, and the structural rows.

Source-local exactness follows from the first entry of the route. An ordinary-point neighbourhood read starts in X, so it can use only the local compact-neighbourhood rows supplied by $\mathsf{LocallyCompact}^{\uparrow}$.

Infinity-neighbourhood exactness is the compact-complement clause. The request first opens the neighbourhood ledger L and the closed-ball row B, then reads Q as the complement of the displayed compact closed-ball or compact metric window.

The two handoffs are separate. The row T exports the ordinary and infinity neighbourhood rows to $\mathsf{Topology}^{\uparrow}$, while K exposes the compact witness already attached to the compact complement. Neither row creates an open-cover compactness theorem.

Finally, H, C, P, N preserve, replay, source, and name the same row inventory. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export a host compactification, selected point, quotient space, choice principle, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification, or bridge credential. □

Theorem 2049.3 (OnePointCompactification compact-neighbourhood handoff)**.** *Every compactness, topology, or completion-facing consumer of an accepted* $\mathsf{OnePointCompactification}^{\uparrow}$ *carrier factors its infinity-neighbourhood read through*

$$X, L, B, \infty \longmapsto Q \longmapsto T, K \longmapsto H, C, P, N.$$

Thus the consumer receives a compact-complement neighbourhood of ∞ only as a displayed proper or closed-ball window with its compact metric witness and topology handoff, and receives no open-cover compactness proof, host compactification space, selected source point, quotient-space construction, choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer request at the added point. The only admitted added point is the row ∞, so the request must read the source row X together with the neighbourhood ledger L and closed-ball row B before any compact complement can be named.

The compact-complement row is exactly Q. It records the complement of the displayed compact closed-ball or compact metric window; the compact witness remains the one already routed through L, B and then through the compact-metric handoff K.

The topology export is exact. The row T sends the ordinary source neighbourhood rows and the infinity-neighbourhood rows into the indexed $\mathsf{Topology}^{\uparrow}$ surface, while H and C transport and replay the same data componentwise.

The provenance row P names only the cited sibling certificates, and N names only the local $\mathsf{OnePointCompactification}^{\uparrow}$ packet. Therefore every downstream compactness, topology, or completion-facing read factors through the displayed route and cannot recover any excluded host or choice-based datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{OnePointCompactification}^{\uparrow}}$**.**

Constructive story: A $\mathsf{OnePointCompactification}^{\uparrow}$ packet is generated from a LocallyCompactUp source row, ProperMetricUp or MetricClosedBallUp neighbourhood ledger, an added infinity row, compact-complement rows, TopologyUp handoff, CompactMetricUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OnePointCompactification}^{\uparrow}}$, seed)

Scope closed: Definition 2049.1, Definition 2049.2, and Definition 2049.3 seed the one-point compactification package over locally compact source, proper or closed-ball neighbourhood, infinity, compact-complement, topology, compact-metric, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OnePointCompactification}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, a host compactification space, selected points, quotient spaces, choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-local exactness, compact-complement exactness, topology handoff, compact-metric handoff, componentwise transport, provenance exhaustion, and compact-neighbourhood handoff for BEDC.Derived.OnePointCompactificationUp.

Chapter 2050

A Concrete Naming Certificate for $\mathsf{ChainConnectedMetric}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Engelking topology tradition, the $\mathsf{ChainConnectedMetric}^{\uparrow}$ packet records connectedness through finite epsilon-chain witnesses rather than through host-level separation principles. It attaches the interval-facing corridor of chapter 1462 to the located metric surface of chapter 2043, the dyadic tolerance core of chapter 291, the stream-window naming layer of chapter 126, the regular rational readback of chapter 301, and the real completion boundary of chapter 38. The certificate gives Real and Completion consumers a finite chain route while making no claim of host connectedness, quotient equality, Heine–Borel compactness, trichotomy, or open-cover compactness.

Definition 2050.1 (Chain-connected metric carrier)**.** *A* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, a, b, \epsilon, C, D, W, Q, E, H, R, P, N)$$

with the following displayed rows:

1. *X is the metric source row, carrying the point classifier, distance classifier, and metric law ledgers read by the packet.*
2. *a and b are endpoint rows over X, with their names exposed only through the displayed source classifier.*
3. *ϵ is a positive dyadic radius request admitted by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
4. *$C = (c_0, \ldots, c_m)$ is a finite chain list, with c_0 transported from a and c_m transported from b.*
5. *D is the dyadic tolerance ledger recording positivity, refinement, and link-radius comparison data for ϵ and every adjacent pair in C.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect the endpoint and chain names.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations attached to distances, endpoint comparisons, and dyadic tolerances.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the Real-facing completion observation obtained after the finite dyadic chain rows have been read.*
9. *H records componentwise* hsame *transport for the metric source, endpoints, dyadic radius, chain list, tolerance ledger, stream windows, rational readbacks, Real seal, replay row, provenance row, and local naming row.*

10. *R records displayed* Cont *replay through*

$$X, a, b, \epsilon, C, D, W, Q \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* ChainConnectedMetric$^{\uparrow}$, ConnectedInterval$^{\uparrow}$, LocatedSpace$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{ChainConnectedMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, a, b, \epsilon, C, D, W, Q, E, H, R, P, N$. It has no coordinate for host connectedness, quotient equality, Heine–Borel compactness, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.ChainConnectedMetricUp.TasteGate.ChainConnectedMetricUp`
Lean4 ✓ `BEDC.Derived.ChainConnectedMetricUp.TasteGate.ChainConnectedMetricTasteGate single carrier alignment`

Theorem 2050.2 (Chain-connected metric NameCert obligations)**.** *For every accepted* ChainConnectedMetric$^{\uparrow}$ *carrier*

$$K = (X, a, b, \epsilon, C, D, W, Q, E, H, R, P, N),$$

the local NameCert$_{\text{ChainConnectedMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, endpoints, dyadic radius, finite chain list, dyadic tolerance ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming row;*
2. *link-radius exactness, requiring every adjacent pair in C to have its distance comparison read through X, D, and Q before the link is admitted;*
3. *endpoint transport, requiring c_0 and c_m to be the displayed* hsame *transports of a and b along the same packet route;*
4. *chain concatenation stability, requiring compatible endpoint transport and dyadic ledgers to compose by finite list concatenation without creating a second metric source;*
5. *Real seal nonescape, requiring E to be read only after $X, a, b, \epsilon, C, D, W, Q$ have been displayed;*
6. *ledger refusal, rejecting any consumer that asks the packet for host connectedness, quotient equality, Heine–Borel compactness, trichotomy, open-cover compactness, Lean verification, or bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2050.1. The accepted packet displays the source X, endpoints a, b, dyadic radius ϵ, chain list C, dyadic ledger D, stream windows W, rational readback Q, Real seal E, and structural rows.

Link-radius exactness is rowwise. Each adjacent pair of the finite list C is inspected through the metric source X, and its dyadic comparison is recorded by D and read back through Q before the link can contribute to the certificate.

Endpoint transport uses the same componentwise hsame row H that transports the rest of the packet. Thus the first and last chain entries are tied to a and b without introducing an ambient equality principle.

Concatenation stability is finite list bookkeeping. Two accepted chains with matching transported endpoint rows concatenate by joining their displayed chain lists and ledgers, while the source row X and the replay route remain unchanged.

The Real seal is downstream of the finite rows. Since E appears only after $X, a, b, \epsilon, C, D, W, Q$ in the replay route, a Real-facing consumer cannot bypass the dyadic chain witnesses.

Finally, the classifier has no coordinates for the excluded principles. Any request for host connectedness, quotient equality, Heine–Borel compactness, trichotomy, open-cover compactness, Lean verification, or bridge checking therefore has no row to read and is refused by the local ledger. □

Theorem 2050.3 (Chain-connected metric Real completion boundary)**.** *Every Real-facing chain-connectedness consumer of an accepted* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *carrier factors through the finite route*

$$X, a, b, \epsilon, C, D, W, Q \longmapsto E \longmapsto H, R, P, N$$

before reading the $\mathsf{Real}^{\uparrow}$ *seal. Consequently the packet exports only dyadic chain rows and their transported Real readback, not a host connectedness theorem, quotient equality principle, Heine–Borel compactness result, trichotomy decision, or open-cover compactness claim.*

Proof. Let $K = (X, a, b, \epsilon, C, D, W, Q, E, H, R, P, N)$ be accepted. A Real-facing consumer first reads the metric source and endpoints because the chain entries and endpoint transports are meaningful only relative to the displayed source classifier.

The positive radius and finite chain are then inspected through ϵ and C. Every adjacent link is certified by the dyadic ledger D, the stream windows W, and the rational readback Q.

Only after those finite rows have been displayed does the replay route expose the Real seal E. The transport, replay, provenance, and local naming rows then preserve the same dependency order.

Thus each Real-facing chain-connectedness claim factors through the displayed dyadic chain route before it can read $\mathsf{Real}^{\uparrow}$. The packet has no remaining coordinate from which the excluded host-level principles could be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChainConnectedMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChainConnectedMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2050.1, Definition 2050.2, and Definition 2050.3 seed the finite epsilon-chain connectedness package over displayed metric, endpoint, dyadic tolerance, chain-list, stream-window, regular rational, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChainConnectedMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host connectedness, quotient equality, Heine–Borel compactness, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, link-radius exactness, endpoint transport, chain concatenation stability, Real seal nonescape, ledger refusal, and Real completion boundary factorization for BEDC.Derived.ChainConnectedMet

Constructive story: A $\mathsf{ChainConnectedMetric}^{\uparrow}$ packet is generated from a metric source row, endpoint rows, a dyadic radius, a finite chain list, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2051

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessModulus}^{\uparrow}$

Following Bishop and Bridges constructive regular-Cauchy completion, the $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ packet records the finite tail-modulus route that turns a displayed regular rational Cauchy row into a terminal $\mathsf{Real}^{\uparrow}$ completion seal. It sits between the active completion objects: the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the threshold discipline of chapter 292, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the Bishop boundary of chapter 1639, and the terminal real seal of chapter 38. The packet exposes explicit Cauchy tail control for completion consumers without adding a selected limit, quotient route, global choice principle, Lean verification, or standard bridge.

Definition 2051.1 (CauchyCompletenessModulus carrier)**.** *A* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, M, D, S, B, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational sequence, its local tail comparison classifier, and the finite request surface read by completion consumers.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold ledger. For each requested tolerance it records a displayed tail threshold and the rowwise evidence that later indices are read only beyond that threshold.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, including dyadic positivity, refinement, and budget-splitting rows used to compare tail requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite prefix, threshold window, and readback indices inspected by the route.*
5. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, displaying the completion-interface obligations that accept regular Cauchy data with an explicit modulus.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row produced by the completion boundary after the route through R, M, D, S, B has been displayed.*
7. *H records componentwise* hsame *transport for the regular sequence, modulus ledger, dyadic budget, stream windows, Bishop boundary, terminal real seal, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay through*

$$R, M, D, S \longmapsto B \longmapsto L \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, D, S, B, L, H, C, P, N$. It has no coordinate for a chosen representative quotient, a selected limit operator, countable choice, ambient completeness beyond the displayed seal, Lean verification, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CauchyCompletenessModulusUp.CauchyCompletenessModulusTasteGate single carrier alignment`

Theorem 2051.2 (CauchyCompletenessModulus NameCert obligations)**.** *For every accepted* CauchyCompletenessM *carrier*

$$K = (R, M, D, S, B, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletenessModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational source, Cauchy threshold ledger, dyadic tolerance data, stream windows, Bishop completion boundary, terminal real seal, transport, replay, provenance, and local naming row;*
2. *threshold exposure, requiring each completion request to display its* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row before any* $\mathsf{Real}^{\uparrow}$ *seal is read;*
3. *dyadic budget routing, requiring each tolerance comparison to factor through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positivity, refinement, and budget-splitting rows;*
4. *stream and readback readiness, requiring the inspected sequence entries, threshold windows, and readback indices to pass through the* $\mathsf{StreamName}^{\uparrow}$ *row and the same regular rational source R;*
5. *Real seal handoff, requiring the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary B to be opened before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row L is named;*
6. *classifier transport, requiring H and C to preserve the same R, M, D, S, B, L inventory under* hsame *and displayed* Cont *replay;*
7. *non-escape, requiring P and N to source and name only the listed route, with no hidden selected limit, quotient representative, countable choice, Lean verification, or standard bridge.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2051.1. The accepted packet displays the regular rational source R, modulus ledger M, dyadic tolerance row D, stream windows S, Bishop boundary B, terminal seal L, and structural rows H, C, P, N.

Threshold exposure is forced by the position of M in the carrier. A completion request cannot reach the seal row until it has first opened the Cauchy threshold row attached to the requested tolerance.

Dyadic budget routing is rowwise. The dyadic data D records positivity, refinement, and budget splitting before the threshold is consumed by the Bishop completion boundary.

The stream and readback rows keep the route finite. The window row S exposes only the inspected prefix, threshold window, and readback indices, while the source row R supplies the regular rational observations attached to those windows.

The Real seal handoff then passes through B and only afterward names L. The transport and replay rows H, C preserve the same inventory, and the provenance and naming rows P, N have no additional coordinate from which an unlisted obligation could be read. □

Theorem 2051.3 (CauchyCompletenessModulus Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-seal read exported by an accepted* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyModulus}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, a consumer can open the terminal seal row L only after reading the displayed threshold ledger M, dyadic tolerance row D, stream window row S, regular rational source R, and Bishop completion boundary B.

Proof. Let $K = (R, M, D, S, B, L, H, C, P, N)$ be accepted. A seal consumer enters through the completion request, so the first visible tail datum is the threshold ledger M.

The tolerance carried by that request is then compared through D. This supplies the dyadic positivity, refinement, and split-budget evidence needed to interpret the threshold as a finite tail claim.

The finite indices and observations are exposed through S and R. The stream window row supplies the inspected indices, and the regular rational source supplies the sequence observations read at those indices.

Only after those rows are displayed can the route open the Bishop completion boundary B. The terminal real seal L is therefore downstream of M, D, S, R, B, and the structural rows H, C, P, N merely transport, replay, source, and name that same path.

Thus every exported $\mathsf{Real}^{\uparrow}$ completion-seal read factors through the stated modulus-to-seal route. No consumer receives a second path through a selected limit, quotient representative, countable choice principle, Lean verification, or standard bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2051.1, Definition 2051.2, and Definition 2051.3 seed the finite modulus-to-seal packet over regular rational source, Cauchy threshold ledger, dyadic tolerance data, stream windows, Bishop completion boundary, terminal real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyCompletenessModulusUp.CauchyCompletenessModulusTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient representatives, countable choice, ambient completeness beyond the displayed RealUp seal, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exposure, dyadic budget routing, stream and readback readiness, Real seal handoff, classifier transport, non-escape, and seal-route factorization for BEDC.Derived.CauchyCompletenessModulusUp.

Constructive story: A $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, a CauchyModulus threshold ledger, DyadicRatCore tolerance data, StreamName windows, a BishopRegularCauchyCompletion boundary, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2052

A Concrete Naming Certificate for $\mathsf{ArchimedeanModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ArchimedeanModulus}^{\uparrow}$ packet records the finite precision modulus by which an explicit Archimedean bound is turned into a bounded Real-facing observation route. It sits over the dyadic tolerance surface of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, the terminal real seal of chapter 38, the Real-facing modulus route of chapter 1423, and the explicit rational-bound surface of chapter 430. The packet is a bridge object for L10 Real and Completion work: it supplies a displayed bound-to-window schedule without reading host real equality, trichotomy, selected limits, quotient representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2052.1 (Archimedean modulus carrier)**.** *An* $\mathsf{ArchimedeanModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, D, S, Q, E, A, H, C, N)$$

with the following displayed rows:

1. *P is the unary precision request. It names the requested observation depth as finite successor data rather than as an external real-valued tolerance.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bound/tolerance row. It records the dyadic comparison budget attached to the requested precision and to the displayed Archimedean bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule selected by P and D.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row inspected on the windows named by S.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after P, D, S, Q have been consumed.*
6. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *rational-bound witness attached to the same displayed real name as E.*
7. *H records componentwise* hsame *transport for the precision request, dyadic row, finite windows, regular readback, Real seal, Archimedean witness, continuation, and local naming rows.*
8. *C records the displayed* Cont *replay from each unary precision request through P, D, S, Q, E, A, H, N.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, trichotomy, selected limits, quotient representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden unbounded stream read.*

Lean4 ✓ `BEDC.Derived.ArchimedeanModulusUp.ArchimedeanModulusTasteGate single carrier alignment`

Theorem 2052.2 (Archimedean modulus NameCert obligations)**.** *For an accepted* $\mathsf{ArchimedeanModulus}^{\uparrow}$ *carrier* $M = (P, D, S, Q, E, A, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, unary precision exposure, dyadic bound/tolerance exposure, finite-window exposure, regular rational readback, Real seal exposure, Archimedean rational-bound handoff, componentwise transport, displayed continuation replay, and local naming. Each accepted precision request is classified only through the displayed route*

$$P \longmapsto D \longmapsto S \longmapsto Q \longmapsto E, A \longmapsto H, C, N.$$

Proof. Project the carrier tuple in Definition 2052.1. Its coordinates are exactly the unary precision request, dyadic bound/tolerance row, finite-window schedule, regular readback, Real seal, Archimedean bound witness, transport, continuation, and local naming rows.

Induct over the displayed unary precision request P. The zero request consumes the bound/tolerance row D only at the base observation depth and then names the corresponding finite window in S. The successor request repeats the same finite step after exposing one more unary layer, so the read never becomes an unbounded stream request.

After the window is named, the readback row Q and the seal row E are downstream of D and S, while A supplies only the already displayed Archimedean rational-bound witness. The rows H, C, N transport, replay, and name the same route. Since none of host real equality, trichotomy, selected limits, quotient representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a hidden selector has a coordinate in M, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ArchimedeanModulusUp.ArchimedeanModulusNameCertObligations`

Theorem 2052.3 (Archimedean modulus window bound)**.** *Every accepted* $\mathsf{ArchimedeanModulus}^{\uparrow}$ *packet* $M = (P, D, S, Q, E, A, H, C, N)$ *binds each unary precision request to a finite* $\mathsf{StreamName}^{\uparrow}$ *window before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is available. The selected window is bounded by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *together with the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *witness* A*; it is not selected by real trichotomy, a host supremum, quotient representatives, countable choice, a chosen limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the unary request P. By Definition 2052.2, every accepted read must expose D before S, so the dyadic bound/tolerance row is available before the finite window is named.

Induct on P. The base request uses the first displayed tolerance row and the Archimedean witness A to name the corresponding finite window. In the successor step, the previous finite route is replayed and the next tolerance row is read through the same D, A surface before the next finite window is admitted.

Now inspect the downstream rows. The regular readback Q and Real seal E are read only after the window in S is displayed, while H, C, N preserve and name that route. Thus the window bound is internal to the displayed dyadic and Archimedean rows and cannot be supplied by host real equality, trichotomy, a host supremum, quotient representatives, countable choice, selected limits, or ambient Real completeness. □

Theorem 2052.4 (Archimedean modulus Real handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{ArchimedeanModulus}^{\uparrow}$ *packet factors through*

$$P \longmapsto D \longmapsto S \longmapsto Q \longmapsto E \longmapsto A \longmapsto H, C, N$$

before reading the bounded precision-modulus surface. In particular, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and bounded lower-cut consumers receive the displayed dyadic tolerance, finite-window, regular readback, Real seal, and Archimedean witness rows, and receive no host real equality, trichotomy, selected limit, quotient representative, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted packet. The Real-facing rows are Q and E, but they are unavailable until the unary precision request P, dyadic row D, and finite-window schedule S have been read.

Apply the window-bound theorem. It binds every window request to the displayed dyadic tolerance row and Archimedean rational-bound witness, so downstream consumers inherit an explicit finite modulus surface rather than a host completeness argument.

Finally replay the continuation row C and transport row H. These rows preserve the same route for $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and bounded lower-cut consumers, while N names the packet locally. The carrier inventory excludes host real equality, trichotomy, selected limits, quotient representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of those sources can enter the Real handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2052.1, Definition 2052.2, Definition 2052.3, and Definition 2052.4 seed the finite Archimedean precision-modulus surface from unary precision requests through dyadic tolerances, finite stream windows, regular rational readback, Real seals, Archimedean rational bounds, transport, continuation, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, trichotomy, selected limits, quotient representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, window-bound exactness, dyadic stability, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.ArchimedeanModulusUp.

Constructive story: An $\mathsf{ArchimedeanModulus}^{\uparrow}$ packet is generated from a unary precision request, DyadicRatCore bound/tolerance row, StreamName finite-window schedule, RegSeqRat readback row, RealUp seal row, ArchimedeanReal rational-bound witness, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Chapter 2053

A Concrete Naming Certificate for $\mathsf{FirstCountableSpace}^{\uparrow}$

Following Bourbaki and Kelley on countability axioms in topology, the $\mathsf{FirstCountableSpace}^{\uparrow}$ packet records the local countable-neighbourhood interface needed between generated-open topology data and metric-completion consumers. It sits over the generated-open classifier of chapter 88, the metric-ball basis route of chapter 1655, the countable-base surface of chapter 1901, and the complete-separable metric consumer route of chapter 1490. The packet names only displayed point windows, local neighbourhood-base rows, refinement rows, radius ledgers, transport, replay, provenance, and local naming; it does not select neighbourhood bases by choice, assert open-cover compactness, identify quotient points, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2053.1 (First-countable-space carrier)**.** *A* $\mathsf{FirstCountableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, T, M, L, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the point or source carrier row whose finite histories are the only points inspected by the packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open classifier row; every open-neighbourhood read passes through this row before it reaches a consumer.*
3. *M is the* $\mathsf{MetricTopology}^{\uparrow}$ *or metric-ball basis route when the local base is metric-presented; if no metric route is displayed, M is the explicit empty witness row.*
4. *L is the local countable-neighbourhood ledger, recording for each accepted point-window the finite index windows and neighbourhood cells exposed at that point.*
5. *D is the dyadic radius and precision ledger used by metric-presented neighbourhood cells and completion-facing boundary reads.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row for finite inspection of the local countable schedule at a displayed point-window.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary and seal row used only when a radius, metric-topology, or completion-facing read requires a real-valued boundary.*

8. *H stores componentwise* hsame *transport for the source, topology classifier, optional metric-topology route, local schedule, dyadic ledger, stream windows, real seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a point-window through the topology classifier, local schedule, optional metric route, precision ledger, stream window, real seal, transport, provenance, and local name.*

10. *P is the* Pkg *provenance row over* $\mathsf{FirstCountableSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FirstCountableSpace}^{\uparrow}}$ *row.*

The carrier exports a displayed local countable base at each accepted point-window. It has no coordinate for a host family of all neighbourhoods, a choice-selected basis, quotient point equality, arbitrary open covers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FirstCountableSpaceUp.FirstCountableSpaceTasteGate single carrier alignment`

Theorem 2053.2 (First-countable-space NameCert obligations)**.** *For an accepted* $\mathsf{FirstCountableSpace}^{\uparrow}$ *carrier*

$$F = (X, T, M, L, D, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FirstCountableSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the point or source row, topology classifier, optional metric-topology route, local countable-neighbourhood ledger, dyadic ledger, stream windows, real seal, transport, replay, provenance, and local name;*

2. *pointwise local-basis coverage, requiring every accepted point-window and generated-open neighbourhood read to meet a displayed neighbourhood cell from L through the same source row X and topology row T;*

3. *pointwise refinement closure, requiring any two displayed local neighbourhood cells accepted at the same point-window to refine through another displayed local cell indexed by the same ledger L;*

4. *metric-topology handoff, requiring every metric-presented local cell to pass through M, D, S, R before it is exported to a completion-facing consumer;*

5. *ledger nonescape, requiring every point, open-neighbourhood, radius, stream-window, real-boundary, transport, replay, provenance, and local-naming read to remain inside $X, T, M, L, D, S, R, H, C, P, N$.*

No first-countability obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2053.1. The accepted packet displays one source row, one topology classifier, one optional metric-topology route, one local countable-neighbourhood ledger, and the structural rows needed to replay and name the packet.

Pointwise local-basis coverage is read from X, T, L. A consumer opens a point-window in X, reads a generated-open neighbourhood through T, and receives only a displayed local cell from L at the same finite point-window.

Pointwise refinement is also internal to L. When two local cells are accepted at one point-window, the refinement witness is another displayed entry of the same local ledger, not a selected neighbourhood from a host family.

Metric-presented cells use the route through M, D, S, R. The optional metric-topology row supplies the ball-basis handoff, the dyadic ledger supplies the radius precision, the stream row

supplies the finite inspection window, and the real row supplies only the boundary seal requested by the consumer.

Finally H, C, P, N transport, replay, source, and name the displayed rows. Since the packet contains no host neighbourhood family, choice-selected local base, quotient point equality, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness field, the listed obligations exhaust the local certificate surface. □

Theorem 2053.3 (First-countable to second-countable handoff)**.** *Every accepted* $\mathsf{SecondCountable}^{\uparrow}$ *packet induces a* $\mathsf{FirstCountableSpace}^{\uparrow}$ *read by restricting the displayed countable-base enumeration to the accepted point-window. The induced first-countable read uses the same source row, topology classifier, optional metric witness, dyadic ledger, stream windows, real seal, componentwise transport, replay, provenance, and local name; only the global base enumeration is read pointwise as the local ledger* L*.*

Proof. Start with the accepted $\mathsf{SecondCountable}^{\uparrow}$ carrier. Its source row, topology classifier, optional metric witness, dyadic ledger, stream row, real seal, transport, replay, provenance, and local name already have the same BEDC roles required by Definition 2053.1.

At a displayed point-window, read the second-countable base enumeration through its stream schedule. The local ledger L is the finite list of displayed basic cells from that enumeration that are witnessed at the same point-window.

Coverage and refinement are inherited pointwise. A generated-open neighbourhood first receives a displayed basic cell from the second-countable coverage row, and two such cells refine through the second-countable refinement row before being re-read as local entries of L.

The remaining rows are unchanged. No selected global basis, host open family, countable-choice principle, quotient point equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness field is introduced by the pointwise restriction. □

Theorem 2053.4 (First-countable-space nonescape)**.** *Let* $F = (X, T, M, L, D, S, R, H, C, P, N)$ *be an accepted* $\mathsf{FirstCountableSpace}^{\uparrow}$ *carrier. Any topology, metric-topology, completion-facing, or Polish-space consumer read of the local countable-neighbourhood surface factors through the displayed route*

$$X, T, M, L, D, S, R, H, C, P, N$$

before it exports a neighbourhood, radius, stream window, or boundary seal. Consequently the packet cannot export open-cover compactness, a choice-selected neighbourhood basis, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a topology consumer. It has no open-neighbourhood coordinate except T, and no local countable-neighbourhood coordinate except L, so the read starts at the source row X, passes through the topology classifier T, and then through the displayed local ledger L.

A metric-topology or completion-facing consumer additionally reads the optional metric route M. Radius and precision requests must pass through D, finite schedule requests through S, and real-boundary requests through R before the consumer can observe a local cell.

The structural rows keep the same route fixed under transport and replay. Componentwise transport is recorded in H, replay in C, provenance in P, and local naming in N.

No other coordinate is available. Since the carrier omits arbitrary open covers, selected neighbourhood bases, quotient point representatives, and ambient real-completeness fields, every exported read remains inside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FirstCountableSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirstCountableSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2053.1, Definition 2053.2, Definition 2053.3, and Definition 2053.4 seed the local countable-neighbourhood interface over displayed source rows, topology classifiers, optional metric-topology routes, local schedules, dyadic ledgers, stream windows, real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FirstCountableSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FirstCountableSpaceUp.FirstCountableSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected neighbourhood bases by choice, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise local-basis coverage, pointwise refinement, second-countable handoff, metric-topology handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FirstCountableSpaceUp.

Constructive story: A $\mathsf{FirstCountableSpace}^{\uparrow}$ packet is generated from one point or source carrier row, one Topology classifier row, an optional MetricTopology route, one local countable-neighbourhood ledger, one DyadicRatCore precision ledger, one StreamName finite-inspection row, one Real boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2054

A Concrete Naming Certificate for $\mathsf{BishopCauchyNormalization}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, the $\mathsf{BishopCauchyNormalization}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is replaced by a cofinal normalized name before the terminal real seal is read. It sits after the modulus-normal-form surface of chapter 1968, the cofinal subsequence route of chapter 1998, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal boundary of chapter 38. The packet gives later $\mathsf{Real}^{\uparrow}$ and completion consumers a normal-form route without quotient real equality, selected limits, countable choice, or ambient completeness.

Definition 2054.1 (BishopCauchyNormalization carrier)**.** *A* $\mathsf{BishopCauchyNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, \phi, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, carrying the regular Cauchy rational observations before normalization.*
2. *M is the* $\mathsf{CauchyModulusNormalForm}^{\uparrow}$ *row that converts the source regularity budget into a displayed monotone precision schedule.*
3. *ϕ is the* $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ *row selecting cofinal source windows along the normalized schedule.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by the source and normalized reads.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, endpoint observations, slack rows, and comparison cells consumed by M and ϕ.*
6. *R is the normalized* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the cofinal schedule and dyadic ledger have been displayed.*
7. *E is the inherited* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after S, M, ϕ, W, D, R have been consumed.*
8. *H records componentwise* hsame *transport for the source, modulus normal form, cofinal route, stream windows, dyadic ledger, normalized readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S, M \longmapsto \phi, W, D \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* BishopCauchyNormalization$^{\uparrow}$, CauchyModulusNormalForm$^{\uparrow}$, RegularCauchyCofinalSu StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BishopCauchyNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, \phi, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected subsequence functions, chosen real limits, trichotomy, countable choice, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyNormalizationUp.BishopCauchyNormalizationTasteGate single carrier alignment`

Theorem 2054.2 (BishopCauchyNormalization NameCert obligations)**.** *For every accepted* BishopCauchyNormaliz *carrier*

$$K = (S, M, \phi, W, D, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopCauchyNormalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular rational row, modulus-normal-form row, cofinal subsequence row, finite-window row, dyadic tolerance ledger, normalized readback, real seal, transport, replay, provenance, and local name;*
2. *modulus normalization, requiring M to expose the monotone precision schedule before any cofinal window is selected;*
3. *cofinal schedule stability, requiring ϕ to choose only displayed source windows and to witness cofinality for every inspected normalized precision;*
4. *regularity preservation, requiring W and D to feed the normalized* RegSeqRat$^{\uparrow}$ *row R with the dyadic slack named by the same schedule;*
5. *Real seal compatibility, requiring E to be read only after the source, normal-form, cofinal, window, dyadic, and normalized readback rows have been displayed;*
6. *ledger non-escape, requiring H, C, P, N to preserve and name only the listed route and to export no quotient stream, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient completeness row.*

No Bishop Cauchy normalization obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2054.1. The accepted packet displays exactly the source regular rational row, modulus-normal-form row, cofinal route, stream windows, dyadic ledger, normalized readback, seal, and structural rows.

Modulus normalization is read from M. Since M is the only coordinate that names the normalized precision schedule, any accepted cofinal read must expose that schedule before ϕ can select source windows.

Cofinal schedule stability is then read from ϕ. The selected windows are visible only as displayed windows in W, and their tolerances are carried by D, so the route cannot appeal to a host subsequence function or an unrecorded selector.

Regularity preservation and seal compatibility follow from the row order. The normalized readback R consumes the selected windows and dyadic slack before the Real seal E is available.

Finally H, C, P, N add only transport, replay, provenance, and naming. Since the carrier has no row for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2054.3 (BishopCauchyNormalization cofinal subsequence route)**.** *Let*

$$K = (S, M, \phi, W, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyNormalization}^{\uparrow}$ *carrier. Every normalized regular Cauchy read exported by K factors through the finite route*

$$S, M \longmapsto \phi, W, D \longmapsto R \longmapsto H, C, P, N.$$

The route is cofinal in the displayed source schedule: each inspected normalized precision reads a source window selected by ϕ after M has exposed the corresponding normal-form request, and before R exports the normalized $\mathsf{RegSeqRat}^{\uparrow}$ *row.*

Proof. Fix an inspected normalized precision. The only row naming the normal-form request is M, so the request is first read through the modulus-normal-form coordinate.

The cofinal row ϕ then supplies the selected source window for that request. Because ϕ is paired with the displayed finite-window row W, the selected index is consumed as a finite observation window rather than as an external subsequence function.

The dyadic ledger D attaches the required tolerance and slack data to the selected window. The normalized readback row R is downstream of both W and D, so it can export only the regular rational observation generated by the displayed cofinal route.

Transport, replay, provenance, and local naming preserve this route. Thus every normalized read factors through $S, M, \phi, W, D, R, H, C, P, N$, and cofinality is the displayed $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ row rather than a choice principle. □

Theorem 2054.4 (BishopCauchyNormalization Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopCauchyNormalization}^{\uparrow}$ *carrier reaches the seal only through*

$$S, M \longmapsto \phi, W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the normalized name exports the same finite source, modulus-normal-form, cofinal-window, dyadic-ledger, normalized-readback, and seal rows, and exports no quotient real equality, chosen limit, selected representative, trichotomy, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier and inspect the Real-facing coordinate. The seal row E is downstream of the source row S, modulus row M, cofinal route ϕ, finite windows W, dyadic ledger D, and normalized readback row R.

By the cofinal subsequence route theorem, each normalized read already passes through the displayed normal-form request, selected source window, dyadic tolerance ledger, and regular rational readback. A Real-facing consumer therefore cannot reach E before that route has been consumed.

The remaining rows H, C, P, N only transport, replay, source, and name the same finite route. Since the carrier inventory contains no quotient real equality, chosen limit, selected representative, trichotomy, countable-choice schedule, or ambient completeness row, none of those principles can escape through the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyNormalization}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2054.1, Definition 2054.2, Definition 2054.3, and Definition 2054.4 seed the finite Bishop Cauchy normalization route from source regular rational names through modulus normal forms, cofinal windows, dyadic ledgers, normalized readback, Real seals, transport, continuation, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, quotient real equality, selected limits, selected representatives, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, modulus normalization, cofinal schedule stability, regularity preservation, Real seal compatibility, and ledger non-escape for BEDC.Derived.BishopCauchyNormalizationUp.

Constructive story: A $\mathsf{BishopCauchyNormalization}^{\uparrow}$ packet is generated from a RegSeqRat source row, CauchyModulusNormalForm row, RegularCauchyCofinalSubsequence row, StreamName windows, DyadicRatCore tolerance ledger, normalized RegSeqRat readback, inherited RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2055

A Concrete Naming Certificate for $\mathsf{InverseFunction}^{\uparrow}$

Following the local inverse function theory of Cartan and Rudin, the $\mathsf{InverseFunction}^{\uparrow}$ packet records only the finite local-inversion rows that a BEDC consumer may inspect after a displayed derivative row, an invertible linear approximation, Newton data, and the regular Cauchy real route have been opened. It consumes the local-solution boundary of chapter 1834, the Newton–Kantorovich route of chapter 1785, the finite step discipline of chapter 223, the regular rational readback route of chapter 301, the finite stream-window surface of chapter 126, and the terminal real-name seal of chapter 38. The chapter names a local certificate route from displayed analytic data to a local bijection window; it does not assert a host inverse function, identify quotient function spaces, select a germ, import ambient Real completeness, or provide bridge verification.

Definition 2055.1 (Inverse function carrier)**.** *An* $\mathsf{InverseFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (F, B, D, L, K, I, J, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed function row, recording the finite graph reads and local classifier for the source and target windows.*
2. *B is the base-point and window row. It records the displayed point x_0, the displayed value y_0, and the source and target neighbourhood names in which the local inverse is read.*
3. *D is the derivative row at the base point and along the displayed window.*
4. *L is the invertible linear approximation row attached to D, together with the finite inverse-witness ledger used for the local inversion estimate.*
5. *K is the* $\mathsf{NewtonKantorovich}^{\uparrow}$ *row that records the contraction neighbourhood and root-boundary handoff for solving $F(x) = y$ near y_0.*
6. *I is the* $\mathsf{ImplicitFunction}^{\uparrow}$ *row that repacks the same local solving data as a graph-window certificate for the equation $F(x) - y = 0$.*
7. *J is the finite* $\mathsf{NewtonIteration}^{\uparrow}$ *step ledger used by K and I before the endpoint is sealed.*
8. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened for source, target, and endpoint observations.*
9. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximations obtained from S.*

10. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed analytic, Newton, implicit, stream, and regular-rational rows have been opened.*

11. *H records componentwise* hsame *transport for the function row, base-point row, derivative row, invertible approximation row, Newton–Kantorovich row, implicit-function row, Newton-iteration ledger, stream windows, regular rational readback, Real seal, replay, provenance, and local naming.*

12. *C records displayed* Cont *replay through*

$$F, B, D, L, J, K, I, S, Q, R \longmapsto \mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}.$$

13. *P is the* Pkg *provenance over* $\mathsf{InverseFunction}^{\uparrow}$, $\mathsf{ImplicitFunction}^{\uparrow}$, $\mathsf{NewtonKantorovich}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, B, D, L, K, I, J, S, Q, R, H, C, P, N$. The carrier has no coordinate for a host inverse function, quotient function space equality, selected inverse germ, global bijection theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.InverseFunctionUp.InverseFunctionTasteGate single carrier alignment`

Theorem 2055.2 (Inverse function NameCert obligations)**.** *For every accepted* $\mathsf{InverseFunction}^{\uparrow}$ *carrier*

$$V = (F, B, D, L, K, I, J, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed function row, base-point and window row, derivative row, invertible approximation row, Newton–Kantorovich row, implicit-function row, Newton-iteration ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name;*

2. *derivative and inverse-witness compatibility, requiring D and L to display the same local linear approximation and inverse ledger used by the solving route;*

3. *Newton route compatibility, requiring J to be the finite step ledger read by K before the root boundary is sealed;*

4. *implicit-function handoff, requiring I to read the same local equation row and graph-window data as the Newton route rather than a second solution source;*

5. *L10 completion-route handoff, requiring S, Q, R to be opened in order through finite StreamName windows, RegSeqRat readbacks, and the terminal Real seal;*

6. *provenance exactness, requiring P and N to source and name only the listed rows and no host inverse function, quotient function space, selected germ, global bijection theorem, ambient Real completeness principle, Lean verification, or bridge credential.*

No inverse-function NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2055.1. The accepted packet displays exactly the function row, base-point and window row, derivative row, invertible approximation row,

Newton–Kantorovich row, implicit-function row, Newton-iteration ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name.

The analytic compatibility obligations are local to D and L. The derivative row supplies the displayed linear data, while the linear approximation row records the inverse-witness ledger used by both the Newton and implicit-function routes.

The solving route is finite. The row J lists the Newton steps, and K reads that ledger as the Newton–Kantorovich contraction and root-boundary handoff. The row I then repacks the same local solving data as a graph-window certificate for $F(x) = y$.

The completion-facing rows are downstream. The stream row S fixes the finite windows, Q supplies the regular rational readback, and R provides the terminal Real seal only after the analytic and solving rows have been opened. The structural rows H, C, P, N transport, replay, source, and name this same route without adding any excluded coordinate. □

Theorem 2055.3 (Inverse function local-bijection handoff)**.** *Every accepted* $\mathsf{InverseFunction}^{\uparrow}$ *read exported as a local bijection window factors through the displayed finite route*

$$\mathsf{ImplicitFunction}^{\uparrow} \longmapsto \mathsf{NewtonKantorovich}^{\uparrow} \longmapsto \mathsf{NewtonIteration}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by I, K, J, S, Q, R and the analytic rows F, B, D, L. The consumer receives one displayed function row, one base-point and window row, one derivative row, one invertible approximation row, one Newton–Kantorovich row, one implicit-function row, one finite NewtonIteration ledger, one StreamName window row, one RegSeqRat readback row, one Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row. The handoff is local window data only; it is not a host inverse function, not quotient function space equality, not a selected germ, not a global bijection theorem, not ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and not bridge verification.*

Proof. Project the accepted carrier $V = (F, B, D, L, K, I, J, S, Q, R, H, C, P, N)$. The local bijection read starts with the displayed function and base-window rows F, B, and the only linear source available to it is the pair D, L.

The solving route then passes through the finite Newton data. The row J exposes the step ledger, K reads that ledger as the Newton–Kantorovich root-boundary route, and I identifies the same local solving data as an implicit-function graph window.

The L10 completion route is the final handoff. The finite windows in S are read back through Q, and the terminal Real data appears only in R. Thus a Real-facing inverse-window consumer cannot bypass the finite StreamName and RegSeqRat route.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no carrier coordinate stores a host inverse function, quotient function space, selected germ, global bijection theorem, ambient completeness theorem, or bridge credential, every exported local-bijection handoff is exhausted by the displayed finite packet. □

Falsifiable prediction: Every $\mathsf{InverseFunction}^{\uparrow}$ local-bijection read factors through one displayed function row, one base-point and window row, one derivative row, one invertible approximation row, one NewtonKantorovich row, one ImplicitFunction row, one finite NewtonIteration ledger, one StreamName finite-window row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: implicitfunction, newtonkantorovich, newtoniteration, regseqrat, streamname, real: $\mathsf{InverseFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local

bijection windows, derivative and inverse-witness rows, finite Newton solving data, implicit graph-window handoff, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}$.

Constructive story: An $\mathsf{InverseFunction}^{\uparrow}$ packet is generated from a displayed function row, a base-point and window row, a derivative row, an invertible approximation row, a NewtonKantorovich row, an ImplicitFunction row, a finite NewtonIteration ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2055.1, Definition 2055.2, and Definition 2055.3 seed the finite local inversion certificate over analytic rows, Newton solving rows, implicit graph-window rows, finite stream windows, regular rational readback, and Real seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InverseFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, quotient function spaces, selected inverse germs, global bijection theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative and inverse-witness compatibility, Newton route compatibility, implicit-function handoff, L10 completion-route handoff, provenance exactness, and local naming for BEDC.Derived.InverseFunctionUp.

Chapter 2056

A Concrete Naming Certificate for $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$

Bishop–Bridges constructive completion treats convergence toward a completed real as a displayed modulus discipline: a completion-facing consumer may open only the finite windows, regular rational readbacks, dyadic tolerances, and real-name seals justified by the exhibited rate. The $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$ packet records that discipline on the L10 Real and Completion spine. It sits after the dyadic rational ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite modulus certificate for completion-facing reads; it does not identify quotient streams, choose limits, invoke countable choice, assert host real equality, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2056.1 (Constructive completion modulus carrier)**.** *A* $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, \mu, W, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy family row, including the displayed observations and precision requests whose completion-facing reads are being certified.*
2. *μ is the unary modulus row assigning to each requested tolerance a finite stage bound for the same source Cauchy family.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened only after the relevant value of μ has been displayed.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic budget consumed by μ, W, and Q.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal exposed only after the modulus, window, readback, and dyadic tolerance rows have been displayed.*
7. *H records componentwise* hsame *transport for the source Cauchy family, unary modulus, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the completion route*

$$S, \mu \longmapsto W \longmapsto Q \longmapsto D \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, \mu, W, Q, D, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveCompletionModulusUp.ConstructiveCompletionModulusTasteGate`
`single carrier alignment`

Theorem 2056.2 (Constructive completion modulus NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$ *carrier*

$$M = (S, \mu, W, Q, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy family, unary modulus, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*

2. *modulus/readback discipline, requiring each completion-facing read to expose the unary modulus value before the corresponding* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback are inspected;*

3. *finite-window completion handoff, requiring the completion-facing read to pass through the displayed window W, readback Q, dyadic ledger D, and Real seal R in that order;*

4. *transport/replay stability, requiring H and C to preserve the same source Cauchy family, modulus value, window, readback, tolerance budget, and seal under componentwise* hsame *transport and displayed* Cont *replay;*

5. *provenance/local naming, requiring P and N to source and name only the displayed rows and to export no quotient stream, selected limit, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No constructive completion modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2056.1. The accepted packet displays exactly the source Cauchy family, unary modulus, finite-window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

The modulus/readback discipline is read from μ, W, and Q. Since the window row is opened only after the relevant modulus value is displayed, the regular rational readback cannot be inspected as an unbounded stream or as a selected completion representative.

The finite-window completion handoff is forced by the route through W, Q, D, R. The dyadic ledger records the tolerance budget before the Real seal is exposed, so the seal is a named terminal row of the finite route rather than an ambient completeness principle.

Transport and replay stability are exactly the content of H and C. They preserve the same source row, modulus value, window, readback, tolerance ledger, and seal, so a replayed packet cannot substitute a different completion witness.

Finally P and N add provenance and local naming only for the displayed route. Because the carrier has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the five obligation rows exhaust the certificate surface. □

Theorem 2056.3 (Constructive completion modulus finite route)**.** *Let*

$$M = (S, \mu, W, Q, D, R, H, C, P, N)$$

be an accepted $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$ *carrier. Every completion-facing read of M factors through the finite route*

$$S, \mu \longmapsto W \longmapsto Q \longmapsto D \longmapsto R \longmapsto H, C, P, N.$$

Equivalently, the read uses only the displayed unary modulus, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *packet, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It exports no quotient stream, selected limit, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Fix a completion-facing read of an accepted carrier. Projection gives the source Cauchy family S and the unary modulus row μ.

The read then opens only the finite window justified by the displayed modulus value. Thus the $\mathsf{StreamName}^{\uparrow}$ row W and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q are downstream of μ and cannot be replaced by an unbounded quotient stream.

The dyadic ledger and Real seal follow the same finite route. The row D records the tolerance budget consumed by the read, and R is exposed only after that budget is displayed.

The structural rows close the route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row preserve and name only S, μ, W, Q, D, R. Hence no completion-facing consumer receives selected limits, countable choice, host real equality, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.ConstructiveCompletionModulusUp.ConstructiveCompletionModulusCarrier finite route`

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2056.1, Definition 2056.2, and Definition 2056.3 seed the finite modulus route for completion-facing reads on the L10 Real and Completion spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ConstructiveCompletionModulusUp.ConstructiveCompletionModulusCarrier\_finite\_route`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, modulus/readback discipline, finite-window completion handoff, transport/replay stability, provenance/local naming, and finite-route nonescape for BEDC.Derived.ConstructiveCompletion

Constructive story: A $\mathsf{ConstructiveCompletionModulus}^{\uparrow}$ packet is generated from a source Cauchy family row, a unary modulus row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2057

A Concrete Naming Certificate for $\mathsf{NestedIntervalConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedIntervalConvergence}^{\uparrow}$ packet records the handoff by which a displayed shrinking closed interval route is consumed as a regular rational completion name before a terminal real seal is read. It sits between the finite nested-packet surface of chapter 320, the completion-facing consumer of chapter 1421, the endpoint-cell discipline of chapter 394, the closed-interval Cauchy route of chapter 1778, and the $\mathsf{RegSeqRat}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$ seal route. The packet separates convergence handoff data from raw nested-interval rows and from the Cantor-intersection consumer: it names the finite diagonal enclosure, the regular-rational readback, and the real-seal nonescape surface without adding point choice, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2057.1 (Nested interval convergence carrier)**.** *A* $\mathsf{NestedIntervalConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (N, L, W, D, E, R, Q, H, C, P, M)$$

with the following displayed rows:

1. *N is an accepted* $\mathsf{NestedInterval}^{\uparrow}$ *packet, carrying the finite endpoint, width, inclusion, schedule, regular-rational, and seal-facing rows already displayed by the nested-interval certificate.*
2. *L is a* $\mathsf{LocatedInterval}^{\uparrow}$ *row binding the visible lower and upper endpoint reads to rational cells and optional dyadic refinements.*
3. *W is the shrinking-window ledger. For each displayed precision request it records a visible nested interval row, its endpoint enclosure, and a dyadic width bound small enough for the requested regular-rational tolerance.*
4. *D is the diagonal selection row. It chooses from N and W a finite window for each displayed request; it does not select a completed intersection point.*
5. *E records the endpoint enclosure rows selected by D and the dependency path back to N, L, W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the selected endpoint enclosures and their displayed width bounds.*
7. *Q is the closed-interval completion handoff row, reading the same selected enclosures through the* $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$ *route before any real-facing seal is exposed.*
8. *H records the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after N, L, W, D, E, R, Q have been displayed.*

9. *C records componentwise* hsame *transport and displayed* Cont *replay for nested packets, located endpoints, shrinking ledgers, diagonal selections, enclosures, regular-rational handoffs, completion handoffs, and seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalConvergence}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *M is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, L, W, D, E, R, Q, H, C, P, M$. It has no coordinate for a raw infinite chain, a chosen intersection point, open-cover compactness, host real equality, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedIntervalConvergenceUp.NestedIntervalConvergenceTasteGate single carrier alignment`

Theorem 2057.2 (Nested interval convergence NameCert obligations)**.** *For every accepted* NestedIntervalConverger *carrier*

$$G = (N, L, W, D, E, R, Q, H, C, P, M),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested-interval packet, located-interval endpoint row, shrinking-window ledger, diagonal selection row, selected endpoint enclosures, regular-rational handoff, closed-interval completion handoff, real seal, transport, replay, provenance, and local naming rows;*

2. *nested-source exactness, requiring every interval read to open N and every endpoint-cell read to open L before the shrinking ledger W is consumed;*

3. *diagonal-window exactness, requiring D to select only a displayed finite window already recorded in N, W for the current precision request;*

4. *enclosure-width exactness, requiring every row of E to carry the selected endpoint enclosure together with the displayed dyadic width bound that justifies the request;*

5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exactness, requiring R to be read only from E and its width ledger;*

6. *closed-interval completion ordering, requiring Q to consume N, L, W, D, E, R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal H is exposed;*

7. *provenance exhaustion, requiring C, P, M to transport, replay, source, and name only the displayed route.*

No convergence obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2057.1. The accepted packet displays the nested source N, located endpoint row L, shrinking ledger W, diagonal selector D, selected enclosures E, regular-rational handoff R, completion handoff Q, terminal seal H, and structural rows C, P, M.

Nested-source exactness and diagonal-window exactness are the first two route constraints. A convergence request must read the finite nested packet and the located endpoint cells before it can inspect W, and the selector D can point only to a window already present in the displayed ledger.

Enclosure-width exactness then identifies the data used by the regular-rational handoff. Each row of E carries the selected endpoint enclosure together with the dyadic width bound that answers the displayed precision request, so R has no hidden source outside those rows.

The completion ordering is downstream of the handoff. The row Q consumes the nested source, located endpoints, shrinking ledger, selector, enclosures, and regular-rational readback before the seal row H becomes visible.

Finally, the transport, provenance, and local naming rows preserve the same finite inventory. Since the carrier contains no other coordinate, the listed obligations exhaust the local NameCert surface and cannot export an infinite intersection point, compactness theorem, host real equality, quotient equality, choice principle, or ambient completeness theorem. □

Theorem 2057.3 (Nested interval convergence carrier admission)**.** *For every accepted* $\mathsf{NestedIntervalConvergence}^{\uparrow}$ *carrier*

$$G = (N, L, W, D, E, R, Q, H, C, P, M),$$

carrier admission projects exactly the nested source N, the located endpoint-cell row L, the shrinking ledger W, the diagonal window selections D, the endpoint enclosures E, the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R, the closed-interval completion handoff Q, the terminal* $\mathsf{Real}^{\uparrow}$ *seal H, componentwise* hsame *transport and displayed* Cont *replay in C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}$ *naming row M.*

No admitted coordinate is available outside these rows.

Proof. Project the accepted carrier through Definition 2057.1. The tuple displays exactly the coordinates $N, L, W, D, E, R, Q, H, C, P, M$, and the classifier compares accepted packets only through those coordinates.

The carrier-admission row of Definition 2057.2 records the same inventory as the local NameCert surface. Hence admission opens the nested source, located endpoint-cell row, shrinking ledger, diagonal selector, selected enclosures, handoff rows, seal row, transport and replay row, provenance row, and local naming row, with no extra projection.

Any selected point, raw infinite chain, host interval-intersection theorem, quotient equality, choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum would require a further carrier coordinate. The definition supplies none, so the admitted packet is exhausted by the displayed rows. □

Theorem 2057.4 (Nested interval convergence diagonal-window exactness)**.** *For every accepted* $\mathsf{NestedIntervalConvergence}^{\uparrow}$ *carrier*

$$G = (N, L, W, D, E, R, Q, H, C, P, M),$$

each diagonal finite-window selection used by a $\mathsf{RegSeqRat}^{\uparrow}$ *consumer is justified by the displayed shrinking ledger W and by a dyadic width bound recorded in the selected endpoint enclosure row E before the readback row R can be consumed.*

Thus R cannot read a diagonal window not already displayed by N, W, D, E for the current precision request.

Proof. Begin with the diagonal selector D. By the diagonal-window row of Definition 2057.2, D may select only a finite window already recorded in the nested packet N and the shrinking ledger W.

The selected enclosure row E supplies the quantitative witness for that choice. Its enclosure-width row carries the endpoint enclosure together with the displayed dyadic width bound, so the precision request is justified before the regular-rational row is visible.

The handoff row R is downstream from E. Combining the diagonal-window, enclosure-width, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows gives the required route

$$N, W, D, E \longmapsto R.$$

No alternative readback remains, because a hidden diagonal choice or an unrecorded width certificate would lie outside the carrier coordinates admitted by Definition 2057.3. □

Theorem 2057.5 (Nested interval convergence provenance exhaustion)**.** *For every accepted* NestedIntervalConverge *carrier*

$$G = (N, L, W, D, E, R, Q, H, C, P, M),$$

every completion-facing or $\mathsf{Real}^{\uparrow}$*-facing consumer factors through the displayed* Pkg *provenance* P *and local* $\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}$ *route* M*, together with the transported and replayed rows in* C*.*

Such a consumer receives no selected point, host interval-intersection theorem, quotient equality of real names, choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the consumer route through the completion-facing row Q or the terminal seal row H. The completion-ordering row of Definition 2057.2 makes both reads downstream from the displayed nested source, located endpoints, shrinking ledger, diagonal selector, selected enclosures, and regular-rational handoff.

The remaining structural data are C, P, M. The provenance-exhaustion row of Definition 2057.2 requires these rows to transport, replay, source, and name only the displayed route, so the consumer cannot bypass the local NameCert surface.

The exclusions follow from carrier admission. A selected point, host interval-intersection theorem, quotient equality, choice schedule, or ambient completeness theorem would have to be stored as a further provenance branch or naming coordinate, but the accepted carrier contains only $N, L, W, D, E, R, Q, H, C, P, M$. □

Theorem 2057.6 (Nested interval convergence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedIntervalConvergence}^{\uparrow}$ *carrier factors its read through the finite route*

$$N, L \longmapsto W \longmapsto D \longmapsto E \longmapsto R.$$

Thus the consumer receives only a displayed nested source packet, located endpoint-cell data, a shrinking-window ledger, a diagonal finite-window selection, selected endpoint enclosures, and the dyadic width bounds used to build the regular-rational readback. It receives no chosen intersection point, no invisible tail of the interval chain, no quotient equality of real names, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Start with the regular-rational row R in Definition 2057.1. The carrier makes R downstream from the selected enclosures E, not from a separate real name or an unrecorded interval tail.

The selected enclosures are themselves downstream from D. By the diagonal-window obligation in Definition 2057.2, the selector can choose only a visible window from N, W for the displayed precision request, with located endpoint-cell data supplied through L.

The dyadic width row in E is the quantitative part of the handoff. It is the only tolerance witness available to R, so the regular-rational consumer receives exactly the finite enclosure and width data recorded in the packet.

No excluded source remains. A chosen intersection point, invisible tail, quotient real equality row, or ambient completeness theorem would require a coordinate outside N, L, W, D, E, R, and the carrier has none. □

Theorem 2057.7 (Nested interval convergence real-seal nonescape)**.** *For every accepted* NestedIntervalConvergence *carrier*

$$G = (N, L, W, D, E, R, Q, H, C, P, M),$$

every $\mathsf{Real}^{\uparrow}$*-facing read factors through the displayed nested packet* N*, located endpoint row* L*, shrinking-window ledger* W*, diagonal selector* D*, selected endpoint enclosures* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R*, closed-interval completion handoff* Q*, transport and replay rows* C*, package provenance* P*, and local naming row* M *before the terminal seal row* H *is consumed.*

The seal route does not claim open-cover compactness, a selected intersection point, uniqueness of the point, host real equality, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through the seal-facing coordinate H. The only route to H named by the carrier passes through the regular-rational handoff R and the closed-interval completion handoff Q.

Use Definition 2057.6 to identify the source of R. The handoff reads only the finite nested source, located endpoints, shrinking ledger, diagonal selector, selected enclosures, and displayed width bounds.

Next use the completion-ordering obligation of Definition 2057.2. The row Q consumes the same selected enclosure route before H is exposed, so a real-facing read cannot bypass the finite completion surface.

The remaining rows C, P, M transport, replay, source, and name this route. Because no carrier coordinate stores compactness, point selection, uniqueness, host equality, quotient equality, countable choice, or ambient completeness, every real-seal read remains inside the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalConvergence}^{\uparrow}$ packet is generated from an accepted NestedIntervalUp packet, one LocatedIntervalUp endpoint-cell row, a shrinking-window ledger, diagonal finite-window selections, selected endpoint enclosures with dyadic width bounds, one RegSeqRatUp handoff, one ClosedIntervalCauchyCompletionUp handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2057.1, Definition 2057.2, Definition 2057.6, and Definition 2057.7 seed the constructive nested-interval convergence handoff from displayed nested packets and located endpoints through shrinking windows, diagonal enclosures, regular-rational readback, closed-interval completion handoff, terminal real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected intersection points, uniqueness of the point, host real equality, quotient equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source exactness, diagonal-window exactness, enclosure-width exactness, RegSeqRat handoff exactness, closed-interval completion ordering, provenance exhaustion, and real-seal nonescape for BEDC.Derived.NestedIntervalConvergenceUp.

Chapter 2058

A Concrete Naming Certificate for $\mathsf{DecimalReal}^{\uparrow}$

Following Bishop and Bridges on real numbers presented by effective approximations, the $\mathsf{DecimalReal}^{\uparrow}$ packet records the finite decimal-expansion readback route before a decimal prefix may be consumed as a regular rational name or as a terminal real seal. It sits beside the signed-binary readback boundary of chapter 1976, while using the finite schedule of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet is a finite means presentation of decimal readback; it is not host real equality, quotient stream equality, decimal uniqueness at dual expansions, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2058.1 (DecimalReal carrier)**.** *A* $\mathsf{DecimalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the signed decimal prefix row, exposing a sign cell, a finite integer part, and a finite base-ten fractional window.*
2. *A is the ambiguity boundary row recording whether the observed window is ordinary, trailing-nine-facing, or trailing-zero-facing, without choosing a quotient representative at a dual decimal expansion.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting exactly the inspected decimal prefix windows and requested precisions.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row obtained by bounding the displayed base-ten tail by a dyadic radius visible to BEDC consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed decimal window schedule and dyadic tolerance ledger.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, A, W, T, R have been displayed.*
7. *H records componentwise* hsame *transport for the signed prefix, ambiguity boundary, schedule, tolerance, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, A, W \longmapsto T \longmapsto R \longmapsto L.$$

9. *P is the* Pkg *provenance over* $\mathsf{DecimalReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SignedBinaryExpansion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, W, T, R, L, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, uniqueness of a decimal expansion at dual tails, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2058.2 (DecimalReal NameCert obligations)**.** *For every accepted* $\mathsf{DecimalReal}^{\uparrow}$ *carrier*

$$D = (S, A, W, T, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DecimalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the signed decimal prefix, ambiguity boundary, finite schedule, dyadic tolerance ledger, regular rational readback, real seal, transport, replay, provenance, and local name;*
2. *decimal-window stability, requiring S and W to expose the same finite sign, integer, and fractional cells before and after the displayed* hsame *transport H;*
3. *dual-tail boundary visibility, requiring A to be inspected before any consumer treats a trailing-nine-facing or trailing-zero-facing window as a terminal decimal presentation;*
4. *dyadic tolerance visibility, requiring T to bound the remaining base-ten tail by a displayed dyadic row before R or L is read;*
5. *handoff order, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume S, A, W, T before the terminal* $\mathsf{Real}^{\uparrow}$ *seal L becomes available;*
6. *provenance locality, requiring C, P, N to replay, package, and name only the listed rows and to export no host equality, quotient stream equality, decimal uniqueness, trichotomy, countable choice, open-cover compactness, or ambient completeness coordinate.*

No DecimalReal obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DecimalRealUp.DecimalRealTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 2058.1. The accepted packet displays exactly the signed prefix row, ambiguity boundary, finite schedule, dyadic tolerance row, regular readback, real seal, and structural rows.

Decimal-window stability is read from S, W, H. The transport row preserves the displayed sign, integer cells, fractional window, and request schedule componentwise, so a consumer cannot silently replace the observed decimal window.

The dual-tail boundary is a separate row. The row A is inspected before any terminal use of a trailing-nine-facing or trailing-zero-facing prefix, and it records the boundary without selecting one quotient representative.

The dyadic tolerance and handoff clauses are ordered. The row T bounds the unseen base-ten tail in the dyadic ledger, the regular rational row R consumes that ledger, and the real seal L is terminal.

Finally C, P, N add only replay, provenance, and local naming. Since the carrier has no coordinate for the excluded principles, the listed rows exhaust the local NameCert surface. □

Theorem 2058.3 (DecimalReal RegSeqRat handoff)**.** *Let*

$$D = (S, A, W, T, R, L, H, C, P, N)$$

be an accepted $\mathsf{DecimalReal}^{\uparrow}$ *carrier. Every regular rational or real-name read exported by* D *factors through the displayed route*

$$S, A, W \longmapsto T \longmapsto R \longmapsto L$$

with structural support H, C, P, N*. Thus a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer receives the signed finite decimal window, the dual-tail boundary row, the inspected schedule, and the dyadic tolerance row as finite BEDC data before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is available.*

The handoff supplies no host real equality, no quotient equality of streams, no uniqueness claim for dual decimal expansions, no trichotomy row, no countable-choice row, no open-cover compactness row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a consumer of the regular rational row R or terminal seal L. By the carrier definition, neither row is independent of the displayed decimal data.

The consumer first reads S through the finite schedule W. This exposes the sign, integer prefix, fractional window, and requested precision as finite rows rather than as an infinite decimal stream.

The ambiguity boundary row A is then read before any terminal decimal interpretation. It records the dual-tail boundary visible in the observed prefix but does not decide quotient equality or uniqueness at trailing tails.

Next the dyadic tolerance row T bounds the unseen base-ten tail. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R consumes that dyadic bound, and only then may the $\mathsf{Real}^{\uparrow}$ seal L be inspected.

The structural rows H, C, P, N transport, replay, package, and name the same route. Because the packet contains none of the excluded coordinates, every exported read remains inside the displayed finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalReal}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DecimalReal}^{\uparrow}}$, seed)

Scope closed: Definition 2058.1, Definition 2058.2, and Definition 2058.3 seed the finite decimal readback route from signed prefix windows through dual-tail boundary rows, stream schedules, dyadic tolerance ledgers, regular rational handoff, Real seals, transport, continuation, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DecimalReal}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.DecimalRealUp.DecimalRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, decimal uniqueness at dual expansions, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, decimal-window stability, dual-tail boundary visibility, dyadic tolerance visibility, handoff order, and provenance locality for BEDC.Derived.DecimalRealUp.

Constructive story: A $\mathsf{DecimalReal}^{\uparrow}$ packet is generated from a signed decimal prefix row, dual-tail ambiguity boundary, StreamName schedule, DyadicRatCore tolerance row, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2059

A Concrete Naming Certificate for $\mathsf{EulerMaclaurin}^{\uparrow}$

Following Euler and Maclaurin, and in the Bishop–Bridges constructive reading, the $\mathsf{EulerMaclaurin}^{\uparrow}$ packet records the finite summation-remainder boundary where a displayed finite partial-sum source is compared with endpoint data, derivative-window rows, Bernoulli-style coefficients, and a dyadic remainder budget before a terminal $\mathsf{Real}^{\uparrow}$ seal is named. It sits beside the summation-by-parts route of chapter 1763, the finite term and partial-sum surface of chapter 1581, the derivative packet of chapter 122, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only the finite BEDC route from inspected terms to a certified remainder budget; it does not import host infinite summation, host integration, selected limits, quotient equality, countable choice, trichotomy, or ambient real completeness.

Definition 2059.1 (Euler–Maclaurin carrier)**.** *An* $\mathsf{EulerMaclaurin}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, A, L, U, D, B, Q, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *or finite partial-sum source row, carrying the inspected term windows, finite summation bounds, and the partial-sum observations that are allowed to enter the formula.*
2. *A is the Abel-summation comparison row, used only as a finite summation-by-parts route for rearranging the inspected partial sums.*
3. *L and U are endpoint* $\mathsf{Real}^{\uparrow}$ *rows for the lower and upper summation endpoints, including the endpoint readback names consumed by the boundary terms.*
4. *D is the derivative-window row, listing the finitely inspected* $\mathsf{Derivative}^{\uparrow}$ *observations and the interval windows on which the remainder estimate is read.*
5. *B is the Bernoulli-style coefficient ledger. It records only the finite coefficient rows requested by the displayed order of the formula.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *remainder-budget row, with positivity, refinement, and split-budget data for the final error request.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes rational approximants for the finite sums, endpoint rows, derivative windows, coefficient ledger, and remainder budget.*

8. *Z is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the finite summation-with-remainder read after S, A, L, U, D, B, Q, R have been displayed.*

9. *H records componentwise* hsame *transport for the series or finite partial-sum source, Abel row, endpoint rows, derivative windows, coefficient ledger, dyadic budget, regular rational readback, terminal real seal, replay row, provenance row, and local naming row.*

10. *C records displayed* Cont *replay through*

$$S, A, L, U, D, B, Q, R \longmapsto Z \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{EulerMaclaurin}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerMaclaurin}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, L, U, D, B, Q, R, Z, H, C, P, N$. It has no coordinate for a host infinite sum, host integral, selected limit, quotient equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EulerMaclaurinUp.EulerMaclaurinTasteGate single carrier alignment`

Theorem 2059.2 (Euler–Maclaurin NameCert obligations)**.** *For every accepted* $\mathsf{EulerMaclaurin}^{\uparrow}$ *carrier*

$$E = (S, A, L, U, D, B, Q, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerMaclaurin}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite series or partial-sum source, Abel row, lower and upper endpoint rows, derivative windows, coefficient ledger, dyadic remainder budget, regular rational readback, terminal real seal, transport, replay, provenance, and local naming row;*

2. *summation-window exposure, requiring every formula request to display the finite term windows and partial-sum bounds in S before the Abel row A is read;*

3. *endpoint readiness, requiring both endpoint* $\mathsf{Real}^{\uparrow}$ *rows L, U to be opened before any boundary term is named;*

4. *derivative and coefficient readiness, requiring the derivative windows D and the finite Bernoulli-style coefficient ledger B to be displayed before the remainder row is accepted;*

5. *dyadic remainder routing, requiring the final remainder request to factor through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positivity, refinement, and split-budget rows in Q;*

6. *rational readback and Real seal handoff, requiring R to expose all finite rational approximants before the terminal* $\mathsf{Real}^{\uparrow}$ *seal Z is named;*

7. *classifier transport, requiring H and C to preserve the same $S, A, L, U, D, B, Q, R, Z$ inventory under* hsame *and displayed* Cont *replay;*

8. *non-escape, requiring P and N to source and name only the listed route, with no hidden host infinite sum, host integral, selected limit, quotient equality, countable choice, trichotomy, ambient real completeness, Lean verification, or bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2059.1. The accepted packet displays the finite source S, Abel row A, endpoint rows L, U, derivative windows D, coefficient ledger B, dyadic budget Q, readback row R, terminal seal Z, and structural rows H, C, P, N.

The summation-window and endpoint obligations are forced by row order. A formula request enters through the finite term and partial-sum source S, may use A only as the displayed summation-by-parts route, and reads the endpoint rows L, U before any boundary term is exposed.

The derivative and coefficient obligations are likewise rowwise. The window row D lists the derivative observations that can participate in the remainder estimate, while B lists only the finitely requested coefficient rows.

The dyadic remainder obligation comes from Q. Its positivity, refinement, and split-budget data are displayed before the terminal estimate is transported to the regular rational readback row R and the real seal Z.

Finally, H and C replay the same inventory under classifier transport, and P, N source and name only that inventory. Since the carrier has no additional coordinate, no unlisted host-analytic or choice-based obligation can be read from the packet. □

Theorem 2059.3 (Euler–Maclaurin remainder handoff)**.** *Every remainder read exported by an accepted* $\mathsf{EulerMaclaurin}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RealSeries}^{\uparrow}, \mathsf{AbelSummation}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Derivative}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the terminal real seal Z can be opened only after the finite source S, Abel row A, endpoint rows L, U, derivative windows D, coefficient ledger B, dyadic budget Q, and regular rational readback row R have all been displayed.

Proof. Let $E = (S, A, L, U, D, B, Q, R, Z, H, C, P, N)$ be accepted. A remainder consumer begins with the finite formula request, so the first visible data are the term windows and partial-sum bounds in S.

The route may then pass through A to express the finite summation-by-parts comparison. The endpoint rows L, U supply the boundary-term seals, while D and B supply the derivative observations and coefficient rows that make the finite correction terms inspectable.

The actual remainder budget is not read until Q is displayed. This dyadic row fixes the positive tolerance, refinements, and budget splits used by the later rational approximants.

The readback row R then exposes rational approximants for each already displayed component, and only afterward does the terminal seal Z become visible. The structural rows H, C, P, N transport and name the same route, so every exported remainder read has the stated factorization. □

Theorem 2059.4 (Euler–Maclaurin Real seal non-escape)**.** *For every accepted* $\mathsf{EulerMaclaurin}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *seal read produced by the packet is either one of the displayed endpoint rows L, U or the terminal seal Z reached through the finite route of Definition 2059.3. No accepted carrier exports a separate* $\mathsf{Real}^{\uparrow}$ *seal through host integration, host infinite summation, selected limits, quotient equality, countable choice, trichotomy, ambient real completeness, Lean verification, or bridge checking.*

Proof. Project an accepted carrier $E = (S, A, L, U, D, B, Q, R, Z, H, C, P, N)$. The only rows whose declared target is $\mathsf{Real}^{\uparrow}$ are the endpoint rows L, U and the terminal seal Z.

The endpoint rows are already displayed as part of the carrier. They are not computed by a host integral or an infinite summation principle; they are named endpoint seals consumed by the finite boundary-term route.

For the terminal seal, apply Definition 2059.3. The seal Z is downstream of S, A, L, U, D, B, Q, R and is therefore reached only after the finite source, boundary, derivative, coefficient, dyadic, and readback rows are visible.

The transport, replay, provenance, and naming rows do not add new real-valued coordinates. Hence there is no remaining path by which a consumer could obtain a detached real seal from an excluded host-analytic or choice-based source. □

Closure status for $\mathsf{NameCert}_{\mathsf{EulerMaclaurin}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EulerMaclaurin}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2059.1, Definition 2059.2, Definition 2059.3, and Definition 2059.4 seed the finite Euler–Maclaurin summation-remainder packet over displayed series or finite partial-sum source, Abel row, endpoint RealUp rows, derivative windows, coefficient ledger, dyadic remainder budget, regular rational readback, terminal RealUp seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EulerMaclaurin}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, host integrals, selected limits, quotient equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, summation-window exposure, endpoint readiness, derivative and coefficient readiness, dyadic remainder routing, rational readback, Real seal handoff, classifier transport, remainder factorization, and non-escape for BEDC.Derived.EulerMaclaurinUp.

Constructive story: An $\mathsf{EulerMaclaurin}^{\uparrow}$ packet is generated from a finite RealSeries or partial-sum source row, Abel summation row, endpoint RealUp rows, derivative-window rows, a Bernoulli-style coefficient ledger, a DyadicRatCore remainder budget, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2060

A Concrete Naming Certificate for $\mathsf{RealUniformEntourageBase}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures, entourage bases, and completion by Cauchy filters, the $\mathsf{RealUniformEntourageBase}^{\uparrow}$ packet specializes chapter 1819 to the Real-induced uniformity of chapter 1624. It supplies the finite entourage-basis certificate consumed by chapter 333 through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger. The packet is a displayed Real-specific basis route; it is not a quotient completion, not a selected-limit construction, and not an assertion of ambient Real completeness.

Definition 2060.1 (Real uniform entourage-base carrier)**.** *A* $\mathsf{RealUniformEntourageBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E_{\mathsf{Real}} = (R, U, B, F, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *source row whose Real metric-ball route supplies every entourage request.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *row induced by R and exposed only through finite entourage data.*
3. *B is the* $\mathsf{EntourageBase}^{\uparrow}$ *row listing the Real-induced basis entourages, their containment witnesses in U, and their finite-intersection refinement witnesses.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row inspected by completion-facing consumers after the Real-induced basis row has been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger, recording admitted Real metric radii, smaller-radius refinements, and the route from each radius request into B.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read by Cauchy-filter consumers after the basis handoff.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S and to the dyadic radius comparisons in D.*
8. *H records componentwise* hsame *transport for the Real uniform source, induced UniformSpace row, EntourageBase row, CauchyFilter handoff, dyadic ledger, finite windows, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real metric-ball request through $R, U, B, F, D, S, Q, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealUniformEntourageBase}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $R, U, B, F, D, S, Q, H, C, P, N$. The carrier has no coordinate for quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealUniformEntourageBaseUp.RealUniformEntourageBaseTasteGate single carrier alignment`

Theorem 2060.2 (Real uniform entourage-base NameCert obligations)**.** *For every accepted* RealUniformEntourageB *carrier*

$$E_{\mathsf{Real}} = (R, U, B, F, D, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBase}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Real uniform source, induced UniformSpace row, Real-induced EntourageBase row, CauchyFilter handoff, DyadicRatCore radius ledger, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local naming row;*
2. *Real metric-ball basis containment, requiring every basis entourage in B to be generated from the Real metric-ball route of R and to carry a displayed containment witness into U;*
3. *finite-intersection refinement, requiring every accepted pair of Real-induced basis rows to refine through a displayed row of B before any Cauchy-filter consumer can inspect it;*
4. *Cauchy-filter handoff, requiring every completion-facing read in F to consume the displayed Real-induced basis row B before reading S or Q;*
5. *classifier transport, requiring H and C to preserve the displayed route R, U, B, F, D, S, Q componentwise under* hsame *and replay it under* Cont*;*
6. *ledger non-escape, requiring D, P, and N to export no quotient completion, selected limit, open-cover compactness, countable-choice selector, trichotomy, or ambient Real completeness principle.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is the finite projection from Definition 2060.1. The projection exposes exactly the Real uniform source, the induced UniformSpace row, the Real-induced EntourageBase row, the CauchyFilter handoff, the DyadicRatCore ledger, finite windows, readback, and the structural rows H, C, P, N.

Real metric-ball basis containment is forced by the source row. A basis request first reads the Real metric-ball route in R, then records the induced entourage in B, and finally displays its containment witness in U.

Finite-intersection refinement is internal to B. The accepted pair of Real-induced basis rows is not refined by a completion object; it is refined by a displayed basis row whose radius comparison is recorded in D.

The Cauchy-filter handoff is downstream of that basis row. A completion-facing consumer reaches F only after B has been replayed, and the finite observations S and Q are read after the handoff.

Transport and replay do not add additional mathematical primitives. The row H preserves each displayed coordinate componentwise, while C repeats the route R, U, B, F, D, S, Q for every accepted request.

Finally, provenance is exhausted by D, P, N. Since quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, and ambient Real completeness are not rows of E_{Real}, the local NameCert surface cannot export them. □

Theorem 2060.3 (Real uniform entourage-base Cauchy-filter handoff)**.** *Every Cauchy-filter consumer of an accepted* $\mathsf{RealUniformEntourageBase}^{\uparrow}$ *carrier reads the Real-induced entourage base only through the displayed rows*

$$\mathsf{RealUniformStructure}^{\uparrow}, \mathsf{UniformSpace}^{\uparrow}, \mathsf{EntourageBase}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{B}$$

Consequently the packet exports no quotient completion, selected limit, open-cover compactness, countable-choice selector, trichotomy, or ambient Real completeness.

Proof. Begin with a Cauchy-filter consumer of E_{Real}. Carrier admission gives the consumer access to the packet only through $R, U, B, F, D, S, Q, H, C, P, N$.

The Real-induced entourage base is therefore read through the displayed basis route. The row R supplies the Real uniform source, U records the induced uniform entourage data, B records the Real-specific basis, and D records the positive-radius ledger used for containment and refinement.

Any Cauchy-filter use is then a downstream read of F. The window row S and the regular rational row Q expose finite observations only after the basis row B and handoff row F have been replayed.

The remaining rows are structural. The BHist, hsame, Cont, Pkg, and NameCert rows transport, replay, source, and name the same finite route, so there is no separate coordinate from which quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, or ambient Real completeness can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBase}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformEntourageBase}^{\uparrow}$ packet is generated from a RealUniformStructure source row, an induced UniformSpace row, a Real-specific EntourageBase row, a CauchyFilter handoff, a DyadicRatCore positive-radius ledger, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformEntourageBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2060.1, Definition 2060.2, and Definition 2060.3 seed the Real-specific entourage-basis route from RealUniformStructure to CauchyFilter consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformEntourageBase}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealUniformEntourageBaseUp.RealUniformEntourageBaseTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a public standard bridge.

Upgrade path: Obligation closure requires carrier admission, Real metric-ball basis containment, finite-intersection refinement, Cauchy-filter handoff, classifier transport, and ledger non-escape proofs for the same packet.

Chapter 2061

A Concrete Naming Certificate for $\mathsf{PositiveRealRadiusScale}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley treatment of uniform structures, the $\mathsf{PositiveRealRadiusScale}^{\uparrow}$ packet names the finite positive-radius interface that the Real uniformity exposes to completion-facing consumers. It sits between the Real uniform structure of chapter 1624 and the Real entourage-base route of chapter 2060: a consumer may ask for smaller positive radii, read the induced entourage row, and hand the resulting finite observation to Cauchy-filter routes without importing selected limits, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2061.1 (PositiveRealRadiusScale carrier)**.** *A* $\mathsf{PositiveRealRadiusScale}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\rho_{\mathsf{Real}} = (U, E, D, S, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *source row whose metric-ball route supplies positive Real-radius requests.*
2. *E is the* $\mathsf{RealUniformEntourageBase}^{\uparrow}$ *row that receives each accepted positive radius as an entourage-basis request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger, recording the admitted dyadic radii, strict positivity witnesses, smaller-radius refinements, and comparison cells used by U and E.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by Real-name and completion consumers when radius requests are read through finite observations.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S and to the dyadic radius comparisons in D.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row that consumes the displayed positive-radius entourage route before completion-facing filters are inspected.*
7. *H records componentwise* hsame *transport for the Real uniform source, Real entourage-base route, dyadic positive-radius ledger, finite windows, rational readback, Cauchy-filter handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a positive-radius request through $U, D, E, S, Q, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{PositiveRealRadiusScale}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{RealUniformEntourageBase}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveRealRadiusScale}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $U, E, D, S, Q, F, H, C, P, N$. The carrier has no coordinate for selected limits, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PositiveRealRadiusScaleUp.PositiveRealRadiusScaleTasteGate single carrier alignment`

Theorem 2061.2 (PositiveRealRadiusScale NameCert obligations)**.** *For every accepted* $\mathsf{PositiveRealRadiusScale}^{\uparrow}$ *carrier*

$$\rho_{\mathsf{Real}} = (U, E, D, S, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PositiveRealRadiusScale}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the RealUniformStructure source row, RealUniformEntourageBase target row, DyadicRatCore positive-radius ledger, StreamName finite windows, RegSeqRat readback, CauchyFilter handoff, transport, replay, provenance, and local naming row;*
2. *positive-radius admission, requiring D to display strict positivity and a dyadic comparison cell before a radius request can be consumed by the Real uniform source U;*
3. *smaller-radius refinement, requiring every accepted radius request to expose a displayed smaller positive radius in D before it is routed to the entourage-base row E;*
4. *finite-window readback, requiring S and Q to record the finite StreamName window and regular rational readback attached to each accepted radius request;*
5. *Cauchy-filter handoff, requiring F to inspect the displayed route U, D, E, S, Q before any completion-facing consumer reads filter data;*
6. *classifier transport, requiring H and C to preserve the displayed route U, D, E, S, Q, F componentwise under* hsame *and replay it under* Cont*;*
7. *ledger non-escape, requiring P and N to export no selected limit, quotient real equality, trichotomy, countable-choice selector, open-cover compactness principle, or ambient Real completeness row.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2061.1. The accepted packet displays exactly the Real uniform source, the Real entourage-base route, the dyadic positive-radius ledger, finite windows, rational readback, Cauchy-filter handoff, and the structural rows H, C, P, N.

Positive-radius admission is read from D. A radius request reaches the Real uniform source U only after the ledger has displayed strict positivity and the dyadic comparison cell attached to that request.

Smaller-radius refinement is also internal to D. The refinement row is then routed to E, so the Real entourage-base consumer receives a displayed smaller positive radius rather than an unrecorded completeness or choice principle.

The finite-window readback rows S and Q attach the radius request to observable StreamName windows and regular rational readback. Thus the Real-name side and the completion-facing side inspect the same finite radius data.

The Cauchy-filter handoff is downstream of the radius and entourage rows. A consumer reaches F only after U, D, E, S, Q have been replayed.

Finally, H, C, P, N add only transport, replay, provenance, and naming. Since selected limits, quotient real equality, trichotomy, countable choice, open-cover compactness, and ambient Real completeness are not rows of ρ_{Real}, the listed obligations exhaust the local NameCert surface. □

Theorem 2061.3 (PositiveRealRadiusScale entourage handoff)**.** *Every Real-uniform or Cauchy-filter consumer of an accepted* $\mathsf{PositiveRealRadiusScale}^{\uparrow}$ *carrier reads the positive-radius scale only through the displayed rows*

$$\mathsf{RealUniformStructure}^{\uparrow}, \mathsf{RealUniformEntourageBase}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow}, \mathsf{BHist}$$

Consequently the packet exports no selected limit, quotient real equality, trichotomy, countable-choice selector, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer of ρ_{Real}. Carrier admission exposes the packet only through $U, E, D, S, Q, F, H, C, P,$

The positive-radius request is first read through the Real uniform source U and the dyadic ledger D. The ledger provides positivity and refinement data before the request can enter the Real entourage-base row E.

Finite observations are then read through S and Q. These rows attach the radius route to StreamName windows and RegSeqRat readback, so the route remains visible to Real-name and completion-facing consumers.

Any Cauchy-filter use is downstream of F. The handoff row receives the already displayed Real uniform and entourage-base route; it does not select a limit or add a quotient equality coordinate.

The remaining rows are structural. The BHist, hsame, Cont, Pkg, and NameCert rows transport, replay, source, and name the same finite route, so there is no separate coordinate from which selected limits, quotient real equality, trichotomy, countable choice, open-cover compactness, or ambient Real completeness can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveRealRadiusScale}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveRealRadiusScale}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2061.1, Definition 2061.2, and Definition 2061.3 seed the finite positive-radius interface from RealUniformStructure through RealUniformEntourageBase to CauchyFilter consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveRealRadiusScale}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PositiveRealRadiusScaleUp.PositiveRealRadiusScaleTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient real equality, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius admission, smaller-radius refinement, finite-window readback, Cauchy-filter handoff, classifier transport, and ledger non-escape for BEDC.Derived.PositiveRealRadiusScaleUp.

Constructive story: A $\mathsf{PositiveRealRadiusScale}^{\uparrow}$ packet is generated from a RealUniformStructure source row, RealUniformEntourageBase route, DyadicRatCore positive-radius ledger, StreamName windows, RegSeqRat readback, CauchyFilter handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{BishopRealLineInterface}^{\uparrow}$ route reads the terminal $\mathsf{Real}^{\uparrow}$ seal through chapter 38, the regular rational readback through chapter 301, the finite stream windows through chapter 126, and the dyadic tolerance rows through chapter 291.

Chapter 2062

A Concrete Naming Certificate for $\mathsf{BishopRealLineInterface}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealLineInterface}^{\uparrow}$ packet records the traditional regular-real interface boundary as finite NameCert data rather than as a host real object. It binds the dyadic endpoint and tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular Cauchy readback of chapter 301, the terminal real-name seal of chapter 38, and the equality-facing comparison boundary of chapter 1820. The packet is an L10 interface surface for Real and Completion consumers; it does not assert completeness of $\mathsf{Real}^{\uparrow}$, quotient stream equality, host real equality, trichotomy, selected limits, countable choice, metric completeness, open-cover compactness, or bridge checking.

Definition 2062.1 (Bishop real-line interface carrier)**.** *A* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, W, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger, recording admitted dyadic endpoints, positive tolerances, and comparison budgets before any real-line consumer reads a value-shaped claim.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only bounded observation windows of the named stream.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row attached to W and to the dyadic tolerance ledger D.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after D, W, R have been displayed.*
5. *Q is the Cauchy-equivalence comparison row for accepted interface packets; it compares endpoint ledgers, finite windows, regular readbacks, and terminal seals through displayed setoid data without adding quotient stream equality or host real equality.*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, terminal Real seal, classifier comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real-line interface request through D, W, R, E, Q, H.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRealLineInterface}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopRealLineInterface}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, Q, H, C, P, N$. The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits, metric completeness, open-cover compactness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopRealLineInterfaceUp.BishopRealLineInterfaceTasteGate single carrier alignment`

Theorem 2062.2 (Bishop real-line interface NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier $I = (D, W, R, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealLineInterface}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission, requiring the packet to expose the dyadic ledger D, stream window W, regular readback R, terminal seal E, comparison row Q, transport row H, continuation row C, package row P, and local naming row N before any consumer can read the interface;*
2. *pointwise* $\mathsf{StreamName}^{\uparrow}$ *stability, requiring every stream observation used by the interface to remain a finite displayed window in W under the componentwise* hsame *transport of H and the replay route of C;*
3. *dyadic tolerance ledger, requiring every real-line request to display its endpoint and tolerance budget in D before reading a stream window, a regular-rational readback, or a comparison row;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring the regular Cauchy row R to be read from the displayed stream windows and dyadic ledger before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E or the Cauchy-equivalence comparison row Q is consumed;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E, P, N to export no* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limit, metric completeness, open-cover compactness, or bridge-checking row.*

No local NameCert obligation is stored outside these five rows.

Proof. Carrier admission is projection from Definition 2062.1. The projection exposes exactly the dyadic ledger D, finite windows W, regular readback R, terminal seal E, comparison row Q, and the structural rows H, C, P, N.

Pointwise stream stability is read through the $\mathsf{StreamName}^{\uparrow}$ surface. The row W contains only bounded observations, and H, C can transport or replay those observations only componentwise, so no global stream equality is introduced.

The dyadic tolerance ledger is the first analytic face. Every value-facing read must pass through D, the public source row of chapter 291, before any finite window or comparison can be consumed.

The regular readback and Real seal are downstream. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ reading of the displayed windows and dyadic budget, while E is reached only after D, W, R have been displayed, matching the Real seal boundary of chapter 38.

Finally, the comparison row is bounded by chapter 1820. It classifies displayed Cauchy data before the seal is read, but it does not add quotient representatives, host real equality, or any excluded completeness principle. □

Theorem 2062.3 (Bishop real-line located interval stream route)**.** *Every located interval datum admitted by a* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier $I = (D, W, R, E, Q, H, C, P, N)$ must factor through the* $\mathsf{StreamName}^{\uparrow}$ *finite windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E before it is consumed as a real-line point. The datum begins as endpoint and tolerance*

information in the dyadic ledger D, is compared only through the Cauchy-equivalence row Q, and is preserved by H, C, P, N as the same displayed interface route.

The route exports no host interval object, Dedekind cut, trichotomy row, quotient equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Unpack the accepted carrier. The located interval datum can enter the interface only through the dyadic endpoint and tolerance ledger D, since no separate host-interval coordinate occurs in I.

Use the finite observation route next. The row W supplies the $\mathsf{StreamName}^{\uparrow}$ windows inspected by the datum, and the row R repacks those windows and the dyadic budget as the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback.

Only then may the terminal seal be consumed. The row E is downstream of D, W, R, and the comparison row Q reads equality-facing information through the displayed Cauchy-equivalence classifier rather than through quotient equality.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no host interval, Dedekind cut, trichotomy, quotient equality, selected representative, choice schedule, or ambient completeness row, none of those objects is exported when the datum is consumed as a real-line point. □

Theorem 2062.4 (Bishop real-line interface four-object route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier reads the interface through the ordered four-object route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer may then inspect Cauchy-equivalence comparison, componentwise transport, continuation replay, package provenance, and local naming data, but it receives no extra route through quotient stream equality, selected limits, countable choice, host real equality, trichotomy, metric completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted carrier $I = (D, W, R, E, Q, H, C, P, N)$. Projection gives a consumer the dyadic row D, the stream-window row W, the regular readback row R, and the terminal seal E in the displayed order of the carrier.

The first object is dyadic. The endpoint and tolerance budget in D is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ chapter, so the route begins with finite dyadic comparisons rather than a host real value.

The second and third objects are the finite stream window and regular rational readback. The row W exposes only bounded observations from $\mathsf{StreamName}^{\uparrow}$, and the row R consumes those observations as the $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy reading.

The fourth object is the terminal Real seal. The row E is reached only after D, W, R have been displayed, exactly as in the $\mathsf{Real}^{\uparrow}$ sibling route.

The remaining rows Q, H, C, P, N compare, transport, replay, source, and name the same route. None of them contains quotient stream equality, selected limits, countable choice, host real equality, trichotomy, metric completeness, open-cover compactness, or ambient Real completeness. □

Theorem 2062.5 (Bishop real-line dyadic-stream sibling route)**.** *Every accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *consumer depends on the sibling route through chapter 291, chapter 126, chapter 301, chapter 38, and chapter 1820. The carrier rows $D, W, R, E, Q, H, C, P, N$ bind that route as dyadic tolerance ledger, finite stream window, regular rational readback, terminal Real seal, displayed Cauchy-equivalence comparison, transport, replay, provenance, and local naming.*

Thus the interface has no direct host-real route, quotient route, trichotomy route, or completeness route outside those sibling chapters.

Proof. Project the carrier rows $D, W, R, E, Q, H, C, P, N$ from Definition 2062.1. The dyadic ledger D is read through chapter 291, and the finite observation row W is read through chapter 126.

The regular readback row R binds the stream window to chapter 301. The terminal seal E is the only Real-facing endpoint and is read through chapter 38.

The equality-facing row Q is then read through chapter 1820. The structural rows H, C, P, N transport, replay, source, and name the same sibling route without adding a second analytic path.

Since no carrier row remains outside $D, W, R, E, Q, H, C, P, N$, an accepted interface consumer receives no direct host-real, quotient, trichotomy, or completeness route. □

Theorem 2062.6 (Bishop real-line interface completion-boundary refusal)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier may read only the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

plus $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison and the structural rows* H, C, P, N*. Therefore the interface cannot export* $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient stream equality, countable choice, host real equality, trichotomy, metric completeness, or open-cover compactness.*

Proof. Project the carrier of Definition 2062.1. The analytic rows are exactly D, W, R, E, Q, and the structural rows are exactly H, C, P, N.

Apply the NameCert obligations of Definition 2062.2. They force dyadic tolerance admission through D, finite stream-window stability through W, regular readback through R, the terminal $\mathsf{Real}^{\uparrow}$ seal through E, and Cauchy-equivalence comparison through Q.

Use the four-object route of Definition 2062.4. A Real or Completion consumer reaches the interface only along $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and then reads comparison, transport, replay, provenance, and local naming data.

Since the carrier has no coordinate beyond those displayed rows, no accepted read can export $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, countable choice, host real equality, trichotomy, metric completeness, or open-cover compactness. □

Theorem 2062.7 (Bishop real-line interface located-limit dependency)**.** *Every Bishop real-line consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier that uses located-limit data must pass through chapter 2387 after the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{BishopLocatedLimit}^{\uparrow}$ *packet supplies the located-limit witness and enclosure row consumed by the interface, while* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *supplies the surrounding real-line packet through* $D, W, R, E, Q, H, C, P, N$*. The dependency exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, metric completeness, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the completion-boundary refusal of Definition 2062.6. It fixes the interface route through dyadic tolerance rows, finite stream windows, regular readback, and the terminal Real seal before any located data can be consumed.

Now read the located-limit witness through chapter 2387. That sibling supplies the located-limit witness and enclosure row on top of the same StreamName, DyadicRatCore, RegSeqRat, and Real-seal route.

The interface rows Q, H, C, P, N then compare, transport, replay, source, and name the same surrounding packet. Since neither the located-limit sibling nor the interface carrier contains a selected limit, quotient stream, choice schedule, host equality, trichotomy, metric-completeness proof, open-cover compactness theorem, or ambient Real-completeness row, the handoff exports none of them. □

Theorem 2062.8 (Bishop real-line interface located-Cauchy handoff)**.** *Every located-Cauchy reader of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier* $I = (D, W, R, E, Q, H, C, P, N)$ *must first read the displayed rows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyEquivalenceSetoid}^{\uparrow}$$

through D, W, R, E, Q *and then pass through the sibling located rows of chapter 1709 and chapter 1850. The handoff exports no selected limit, quotient stream equality, trichotomy, host real equality, countable-choice schedule, metric-completeness assertion, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows $D, W, R, E, Q, H, C, P, N$. The value-facing part is already ordered by Definition 2062.4: a consumer reads the dyadic ledger, finite stream window, regular readback, and terminal Real seal before any comparison row can be inspected.

The comparison row Q remains a Cauchy-equivalence boundary for displayed data. It does not supply a quotient representative or a host equality principle.

A located-Cauchy reader then uses the sibling chapters as the only downstream located surfaces. The interval side is the located interval packet of chapter 1709, and the combined Cauchy side is the Bishop located Cauchy real packet of chapter 1850.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since none of the displayed rows contains a selected limit, quotient equality, trichotomy, host equality, choice schedule, metric-completeness assertion, open-cover compactness, or ambient Real completeness, the handoff exports none of them. □

Theorem 2062.9 (Bishop real-line interface located-order route)**.** *Every located-order consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier* $I = (D, W, R, E, Q, H, C, P, N)$ *must read the dyadic endpoint and tolerance rows* D*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row* Q *before using any order-shaped data. The located-order route is*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto H, C, P, N.$$

It exports no trichotomy, host interval object, selected representative, quotient equality, countable-choice schedule, selected limit, metric-completeness assertion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2062.1. The rows D, W, R, E, Q are the only analytic rows available before the structural transport, replay, provenance, and local naming rows.

Apply Definition 2062.4. Any Real or Completion consumer first reads the dyadic ledger, finite stream windows, regular readback, and terminal Real seal in the displayed order, so located-order data cannot begin from a host interval object or a selected representative.

Use Definition 2062.8 for the comparison boundary. The located-facing read must pass through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row before it can be consumed as order-shaped data, and that row supplies displayed comparison rather than quotient equality or trichotomy.

Finally route the packet through H, C, P, N. These rows transport, replay, source, and name the same finite interface, and the carrier excludes trichotomy, host interval objects, selected representatives, quotient equality, countable choice, selected limits, metric completeness, open-cover compactness, and ambient Real completeness. □

Theorem 2062.10 (Bishop real-line interface RealSeparability handoff)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier* $I = (D, W, R, E, Q, H, C, P, N)$ *may consume the density route only after the displayed interface has exposed*

$$\mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff reads $\mathsf{Rat}^{\uparrow}$ *through the rational density surface of chapter 1809, reads* D, W, R, E *through the interface four-object route, and then packages the same route through* Q, H, C, P, N*. It exports no selected dense point, quotient stream equality, countable-choice selector, host real equality, selected limit, ambient completeness, or detached completeness theorem.*

Proof. Start with the four-object route of Definition 2062.4. It forces every Real or Completion consumer to read the dyadic ledger, finite stream window, regular rational readback, and terminal Real seal through D, W, R, E before any comparison row is inspected.

Next consume the $\mathsf{RealSeparability}^{\uparrow}$ density route through chapter 1809. Its rational dense handoff begins at the displayed $\mathsf{Rat}^{\uparrow}$ source and then uses dyadic tolerances, finite stream windows, regular rational readback, and the Real seal as finite rows rather than as a host dense subset.

Compose these two routes along the shared D, W, R, E interface. The comparison row Q and the structural rows H, C, P, N preserve the same displayed path, so the separability consumer can read density only after $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are visible.

Finally use nonescape from Definition 2062.12. Since the accepted interface has no coordinate for selected dense points, quotient stream equality, countable choice, host equality, selected limits, metric completeness, open-cover compactness, or ambient Real completeness, none of those objects is exported by the handoff. □

Theorem 2062.11 (Bishop real-line interface modulus-comparison handoff)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier* $I = (D, W, R, E, Q, H, C, P, N)$ *must read the interface rows* D, W, R, E, Q *before using a convergence-modulus comparison. The comparison factors through the same route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyEquivalenceSetoid}^{\uparrow}$$

and then reaches the downstream modulus ledger of chapter 2074. It exports no selected limits, quotient stream equality, countable choice, host real equality, or ambient completeness.

Proof. Project the Bishop interface carrier. The rows D, W, R, E, Q expose the dyadic ledger, finite stream window, regular readback, terminal Real seal, and Cauchy-equivalence comparison before any structural row is used.

Apply Definition 2062.4. This forces the value-facing part of the comparison to pass through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order.

Use the Cauchy-equivalence boundary carried by Q. The comparison remains a displayed setoid read before the modulus ledger is consumed, so quotient stream equality and host real equality are not introduced.

Finally hand the route to chapter 2074. The downstream $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger may compare convergence schedules only after the same D, W, R, E, Q route is visible. Since the

Bishop interface carrier has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, or ambient completeness, none of those objects is exported by the handoff. □

Theorem 2062.12 (Bishop real-line interface nonescape)**.** *Every accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier* $I = (D, W, R, E, Q, H, C, P, N)$ *is exhausted by the displayed rows* $D, W, R, E, Q, H, C, P, N$*. Thus the interface exports only dyadic endpoints and tolerances, finite stream windows, regular Cauchy readback, the terminal Real seal, Cauchy-equivalence comparison, transport, continuation, package provenance, and local naming data.*

In particular, the packet exports no $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, host real equality, trichotomy, countable-choice schedule, selected limit, metric-completeness assertion, open-cover compactness principle, or bridge-checking evidence.*

Proof. The carrier tuple lists every coordinate available to a local consumer. Projection from Definition 2062.1 therefore gives exactly $D, W, R, E, Q, H, C, P, N$ and no hidden coordinate.

The mathematical rows are already bounded by the four-object route. The dyadic ledger, finite stream windows, regular readback, and terminal seal are the only value-facing rows; Cauchy-equivalence comparison is a comparison of displayed regular-Cauchy data, not a second equality principle.

The structural rows do not enlarge the packet. The hsame row transports displayed coordinates, the Cont row replays the displayed route, P records provenance for the named interface, and N records the local NameCert row.

Since no excluded object occurs in any carrier coordinate, no accepted read can escape to Real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits, metric completeness, open-cover compactness, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealLineInterface}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealLineInterface}^{\uparrow}$ packet is generated from a DyadicRatCore endpoint and tolerance ledger, finite StreamName windows, RegSeqRat regular readback, a terminal Real seal, Cauchy-equivalence comparison, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealLineInterface}^{\uparrow}}$, obligation)

Scope closed: Definition 2062.1, Definition 2062.2, Definition 2062.4, Definition 2062.10, and Definition 2062.12 seed the traditional constructive real-line interface boundary over the displayed L10 dyadic, stream, regular-readback, Real-seal, comparison, transport, continuation, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealLineInterface}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.BishopRealLineInterfaceUp`

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits, metric completeness, open-cover compactness, or bridge checking.

Upgrade path: Scoped closure requires binding the listed obligations to DyadicRatCore, StreamName, RegSeqRat, RealUp, CauchyEquivalenceSetoid, BHist, hsame, Cont, Pkg, and NameCert rows for BEDC.Derived.BishopRealLineInterfaceUp.

Every $\mathsf{BishopRealLineInterface}^{\uparrow}$ packet consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301 and the terminal $\mathsf{Real}^{\uparrow}$ seal row of chapter 38 before exporting the real-line interface. The sibling route keeps the L10 dependency explicit: finite dyadic and stream rows are read first, the regular rational sequence row is then admitted as the real-line source, and the Real seal is the last value-facing row available to downstream interface consumers.

Theorem 2062.13 (Bishop real-line interface to ordered Cauchy-group forward route)**.** *Every* $\mathsf{OrderedCauchyGroup}^{\uparrow}$ *consumer of chapter 6278 that reads the* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *surface factors through the displayed dyadic endpoint and tolerance ledger, finite stream-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row of the*

present chapter. The ordered Cauchy-group route may consume those rows as the real-line source for order-compatible completion, but it receives no quotient stream equality, selected completed representative, detached completed order, countable-choice schedule, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem from the Bishop interface.*

Proof. Project the carrier $I = (D, W, R, E, Q, H, C, P, N)$ of Definition 2062.1. The only value-facing rows are the dyadic endpoint and tolerance ledger D, stream-window row W, regular readback row R, terminal seal E, and Cauchy-equivalence comparison row Q.

The $\mathsf{OrderedCauchyGroup}^{\uparrow}$ packet reads exactly those rows when it imports a Bishop real-line source for completion. Its own order-compatible completion ledger remains downstream of the displayed $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route in Definition 6278.4.

The structural rows H, C, P, N transport, replay, source, and name the same Bishop interface. Since the interface carrier has no coordinate for quotient equality, selected representatives, detached completed orders, choice schedules, trichotomy, or ambient completeness, none of those objects can be passed to the ordered Cauchy-group consumer. □

Theorem 2062.14 (Bishop real-line interface RealModulus forward route)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer that enters through an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *packet must pass through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *terminal seal, and* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row before the convergence-modulus ledger is read. The route is the same finite interface route of Definition 2062.11 and the reciprocal RealModulus route of Definition 2074.12; it points to the downstream ledger of chapter 2074 without importing quotient streams, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted Bishop interface packet. Its analytic rows expose the dyadic ledger, finite stream windows, regular rational readback, terminal Real seal, and Cauchy-equivalence comparison before any downstream consumer can inspect a modulus ledger.

Apply Theorem 2062.11. A convergence-modulus comparison sourced from the interface is already forced through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyEquivalenceSetoid}$ rows.

Use Theorem 2074.12 on the downstream side. The $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet reads the Bishop interface first and then consumes the same dyadic, stream, regular readback, and Real-seal rows before the convergence-modulus ledger is available.

The two routes therefore meet on the same finite L10 rows before any modulus read. Since neither carrier contains quotient streams, selected limits, countable-choice schedules, host real equality, or ambient Real completeness, none of those objects is exported by the forward route. □

Theorem 2062.15 (Bishop real-line interface regular-completion dependency route)**.** *Every* BishopRealLineInterfac *consumer that reads a completion-facing real-line source through ordered Cauchy-group data must first pass through the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *packet of chapter 1639. The route consumes the displayed dyadic endpoint and tolerance ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, common-tail modulus data, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and componentwise structural rows from that completion packet, then threads the ordered Cauchy-group rows for addition, order, and order-compatible completion through the interface rows* $D, W, R, E, Q, H, C, P, N$.

Thus the Bishop real-line interface depends on regular Cauchy completion as a finite L10 source together with ordered Cauchy-group structure. It does not import a selected completed representative, quotient equality of real names, trichotomy, host real equality, countable choice, metric completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the completion-facing route of chapter 1639. Its accepted packet exposes dyadic tail data, finite stream observations, regular Cauchy readback, common-tail comparison, Real-seal handoff, componentwise transport, continuation replay, provenance, and local naming before any completion consumer may read it.

Next use the four-object route of Definition 2062.4. The interface can read the completion packet only through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, with Cauchy-equivalence comparison and structural rows remaining downstream.

The ordered Cauchy-group data supplies addition, order, and order-compatible completion rows after the same finite real-line source has been displayed. Those rows refine how the source is consumed; they do not add a host completed object or a second equality principle.

Finally apply nonescape from Definition 2062.12. Since the interface carrier has no coordinate for selected representatives, quotient equality, trichotomy, host equality, choice, metric completeness, open-cover compactness, or ambient Real completeness, none of those objects can enter through the regular-completion dependency route. □

Theorem 2062.16 (Bishop real-line interface regular-real seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *carrier must pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier comparison row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row before any public real-line consumer reads it. Equivalently, a public interface read follows*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyEquivalenceSetoid}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N$$

through the carrier rows $D, W, R, Q, E, H, C, P, N$*.*

The route exports no host real equality, quotient stream equality, selected representative, selected limit, trichotomy, countable-choice schedule, metric completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $I = (D, W, R, E, Q, H, C, P, N)$ from Definition 2062.1. The dyadic ledger D is the first value-facing row, and the finite stream window W is the only stream observation row available before readback.

Apply the regular-completion dependency route of Definition 2062.15. It places completion-facing reads behind the regular Cauchy completion packet, whose public surface reaches the interface through dyadic tolerances, finite stream windows, regular readback, common-tail comparison, and the terminal Real seal.

Apply the ordered Cauchy-group forward route of Definition 2062.13. That theorem keeps order-compatible consumers behind the same dyadic, stream, RegSeqRat, Real, and Cauchy-equivalence rows rather than behind a host completed object.

Now use the local four-object route and nonescape theorems of Definition 2062.4 and Definition 2062.12. The comparison row Q can classify only displayed regular-Cauchy data before the seal is read, and H, C, P, N preserve the same route. Therefore every public real-line read reaches the terminal seal only after the RegSeqRat readback, CauchyEquivalenceSetoid comparison, DyadicRatCore tolerance rows, and Real seal have been displayed. □

Chapter 2063

A Concrete Naming Certificate for $\mathsf{LocatedRealField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRealField}^{\uparrow}$ packet records the finite field interface for located Cauchy real names. It sits after the terminal real-name seal of chapter 38, the regular rational source surface of chapter 301, the finite window discipline of chapter 126, the dyadic arithmetic core of chapter 291, the Cauchy real field route of chapter 1717, and the positive apartness surface of chapter 1881. The packet gives Real and Completion consumers one constructive field-facing surface over located Cauchy names; it is not order trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 2063.1 (Located real field carrier)**.** *A* $\mathsf{LocatedRealField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, R, W, D, F, A, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the located real source row. It records the displayed located Cauchy real names and exposes them only through the terminal* $\mathsf{Real}^{\uparrow}$ *seal and the local locatedness data.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source-window row used to read every located real source through regular rational Cauchy data.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite-window discipline for operation, comparison, and apartness reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic ledger. It records the dyadic endpoint operations, positive-radius tolerances, comparison slack, and reciprocal-denominator budgets used by the field surface.*
5. *F is the* $\mathsf{CauchyRealField}^{\uparrow}$ *operation row for zero, one, negation, addition, multiplication, and partial reciprocal reads on the displayed Cauchy real names.*
6. *A is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive-apartness row consumed by reciprocal eligibility and by downstream located-order consumers.*
7. *O is the located comparison row. It records the order and comparison boundary available from the located real presentation without turning it into trichotomy or quotient equality.*
8. *H records componentwise* hsame *transport for the source, regular readback, window, dyadic, field, apartness, comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each field request through $E, R, W, D, F, A, O, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRealField}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealField}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive located-field packet. It has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedRealFieldUp.LocatedRealFieldTasteGate single carrier alignment`

Theorem 2063.2 (Located real field NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealField}^{\uparrow}$ *carrier*

$$L = (E, R, W, D, F, A, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located real source data, regular rational readback row, synchronized finite windows, dyadic tolerance ledger, Cauchy real field row, apartness row, comparison row, transport row, replay row, provenance row, and local naming row;*
2. *operation projection, requiring every zero, one, negation, addition, multiplication, or partial reciprocal read to pass through the displayed* $\mathsf{CauchyRealField}^{\uparrow}$ *row F after the source, window, and dyadic rows are visible;*
3. *apartness and comparison handoff, requiring reciprocal eligibility, positive-gap consumption, and order-facing reads to use the displayed* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A and located comparison row O;*
4. *transport and replay stability, requiring H and C to preserve the same componentwise source, readback, window, dyadic, operation, apartness, comparison, provenance, and local naming rows;*
5. *ledger exhaustion, excluding trichotomy, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking from the local certificate.*

No located real field obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2063.1. The accepted packet displays exactly the located real source row, regular rational readback, synchronized windows, dyadic tolerance ledger, Cauchy real field row, apartness row, comparison row, and structural rows.

The operation projection is inherited from chapter 1717. Its row F is the only place where the packet stores zero, one, negation, addition, multiplication, or reciprocal data, and every such read remains downstream of E, R, W, D.

The apartness and comparison handoff is inherited from chapter 1881 and the located comparison surface. The positive row A supplies the finite apartness boundary used by reciprocal eligibility, while O supplies comparison data without adding a trichotomy coordinate.

Finally, H, C, P, N transport, replay, source, and name only the displayed rows. Since the carrier contains no coordinate for the excluded principles, none can become a local NameCert obligation. □

Theorem 2063.3 (Located real field apartness-field handoff)**.** *Every accepted* $\mathsf{LocatedRealField}^{\uparrow}$ *carrier routes any reciprocal or positive-gap field read through the finite chain*

$$E, R, W, D \longmapsto A, O \longmapsto F \longmapsto H, C, P, N.$$

Thus the field surface consumes located apartness and comparison data before exposing a reciprocal branch, while addition and multiplication remain confined to the same regular rational windows and dyadic arithmetic ledger.

Proof. Start with an accepted carrier $L = (E, R, W, D, F, A, O, H, C, P, N)$. The only located real source is E, and all field reads first pass through the regular rational row R, synchronized window row W, and dyadic ledger D.

For a reciprocal or positive-gap read, the apartness row A must be available before the field row exposes the reciprocal branch. The comparison row O records the located order boundary used by the same consumer, but it does not decide trichotomy or quotient equality.

For addition and multiplication, the field row F uses the same R, W, D route and does not receive separate host arithmetic. These branches therefore remain tied to the located Cauchy real presentation.

The structural rows are downstream of both cases. They preserve the displayed handoff by componentwise transport, replay, provenance, and local naming, and they do not add a selected representative, countable-choice selector, ambient completeness witness, or bridge credential. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealField}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2063.1, Definition 2063.2, and Definition 2063.3 seed the finite located-field route over located real source data, regular rational windows, dyadic tolerances, Cauchy real field operations, positive apartness, located comparison, componentwise transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealField}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.LocatedRealFieldUp`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation projection, apartness-field handoff, comparison confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealFieldUp.

Constructive story: A $\mathsf{LocatedRealField}^{\uparrow}$ packet is generated from located real source data, RegSeqRat readback windows, StreamName finite windows, DyadicRatCore tolerance rows, a CauchyRealField operation row, a CauchyRealApartness positive row, a LocatedRealComparison boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2064

A Concrete Naming Certificate for $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$

Following Bourbaki and Kelley on uniform structures, entourage bases, and Cauchy-filter completion routes, the $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$ packet records the finite Real-side refinement step that sits between the basis surface of chapter 2060 and completion-facing consumers. It reuses the Real uniform source of chapter 1624, the abstract entourage-basis discipline of chapter 1819, and the Cauchy-filter handoff of chapter 333; the radius arithmetic and finite readback are supplied through chapter 291, chapter 126, and chapter 301. The packet is a finite witness surface for Real entourage shrinkage. It is not a quotient completion, not a selected-limit construction, and not an assertion of ambient Real completeness.

Definition 2064.1 (Real uniform entourage-refinement carrier)**.** *A* $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E_{\mathrm{ref}} = (R, U, \epsilon, \delta, B, Y, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *source row whose Real metric-ball route supplies the requested entourage.*
2. *U is the requested Real uniform entourage row, exposed only as a finite request to be refined.*
3. *ϵ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius row naming the request radius for U.*
4. *δ is the smaller positive dyadic radius row, together with the displayed comparison $0 < \delta$ and $2\delta \leq \epsilon$ inside the dyadic ledger.*
5. *B is the* $\mathsf{RealUniformEntourageBase}^{\uparrow}$ *row carrying the Real-induced basis containment witness for the δ-ball entourage inside U.*
6. *Y is the symmetric-shrink row recording that the δ-ball entourage composes with itself inside the requested ϵ-entourage.*
7. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row inspected by completion-facing consumers after the refinement row has been displayed.*
8. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read after F when a consumer asks for a finite Cauchy trace of the refinement.*

9. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to S, ϵ, δ, and the dyadic comparison in the ledger.*

10. *H records componentwise* hsame *transport for the Real uniform source, requested entourage, dyadic radii, basis containment, symmetric shrink, Cauchy-filter handoff, finite windows, readback, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the requested Real entourage through $R, U, \epsilon, \delta, B, Y, F, S, Q, H, P, N$.*

12. *P is the* Pkg *provenance over* $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$, $\mathsf{RealUniformEntourageBase}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformEntourageRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $R, U, \epsilon, \delta, B, Y, F, S, Q, H, C, P, N$. The carrier has no coordinate for quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, bridge checking, Lean verification, or ambient Real completeness.

```
Lean4 ✓  BEDC.Derived.RealUniformEntourageRefinementUp.RealUniformEntourageRefinementTasteGate
single carrier alignment
```

Theorem 2064.2 (Real uniform entourage-refinement NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$ *carrier*

$$E_{\mathrm{ref}} = (R, U, \epsilon, \delta, B, Y, F, S, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealUniformEntourageRefinement}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Real uniform source, requested entourage, dyadic request radius, smaller dyadic radius, basis containment row, symmetric-shrink row, Cauchy-filter handoff, StreamName windows, RegSeqRat readback, transport, replay, provenance, and local naming row;*

2. *requested-entourage grounding, requiring U to be read from the Real uniform source row R and named by the positive dyadic request radius ϵ;*

3. *smaller-radius selection, requiring δ to be a displayed positive dyadic radius with $2\delta \leq \epsilon$ in the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger;*

4. *containment refinement, requiring B to exhibit the δ-ball Real basis entourage as a contained sub-entourage of U;*

5. *symmetric shrink, requiring Y to replay the δ-ball self-composition inside the requested ϵ-entourage;*

6. *Cauchy-filter handoff, requiring every completion-facing read in F to consume B and Y before reading the finite rows S and Q;*

7. *classifier transport, requiring H and C to preserve the displayed route $R, U, \epsilon, \delta, B, Y, F, S, Q$ componentwise under* hsame *and replay it under* Cont*;*

8. *provenance non-escape, requiring P and N to export no quotient completion, selected limit, open-cover compactness, countable-choice selector, trichotomy, bridge-checked claim, Lean-verified claim, or ambient Real completeness principle.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is the finite projection from Definition 2064.1. The projection exposes exactly the Real uniform source, the requested entourage, the two dyadic radius rows, the basis containment row, the symmetric-shrink row, the Cauchy-filter handoff, finite windows, readback, and the structural rows H, C, P, N.

The requested-entourage row is grounded in the Real uniform source. A consumer first reads R, then reads the request U, and then identifies the positive dyadic radius ϵ that names the displayed Real metric-ball request.

The smaller-radius row is internal to the dyadic ledger. The row δ carries the positive-radius witness and the comparison $2\delta \leq \epsilon$, so the containment and symmetric-shrink rows do not need a choice principle or an ambient completeness principle.

Containment refinement is supplied by the Real entourage-base packet. The row B reads the δ-ball basis entourage and displays the witness that this basis row lies inside the requested entourage U.

The symmetric-shrink row is the finite composition witness. It records that two δ-steps remain within the ϵ request, and it is replayed before any completion-facing consumer reaches the Cauchy-filter handoff.

The handoff and readback rows are downstream of the refinement surface. A consumer reads F only after B and Y have been replayed, and the finite observations S and Q expose StreamName windows and RegSeqRat comparisons for that same route.

Finally, transport and provenance add no new mathematical primitive. The row H preserves each displayed coordinate componentwise, C replays the same finite route, and P, N name only the rows already present in E_{ref}. □

Theorem 2064.3 (Real uniform entourage-refinement completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$ *carrier reads a requested Real entourage through a displayed smaller dyadic radius, a Real basis-containment witness, and a symmetric-shrink witness before reaching the Cauchy-filter handoff. Consequently the packet exports no quotient completion, selected limit, open-cover compactness, countable-choice selector, trichotomy, bridge-checked claim, Lean-verified claim, or ambient Real completeness.*

Proof. Begin with a completion-facing consumer of E_{ref}. Carrier admission gives access to the packet only through $R, U, \epsilon, \delta, B, Y, F, S, Q, H, C, P, N$.

The consumer therefore cannot read F directly from the requested entourage. It first reads the Real uniform source R, the requested entourage row U, the request radius ϵ, and the smaller dyadic radius δ with its ledger comparison.

The basis and shrink rows then mediate the handoff. The row B supplies the Real-induced basis containment witness, while Y supplies the symmetric-shrink witness that keeps the doubled δ-step inside the requested ϵ-entourage.

Only after these rows are replayed does the consumer reach the Cauchy-filter row F. The rows S and Q then provide finite StreamName and RegSeqRat readback for the same dyadic route.

The remaining rows are structural. The BHist, hsame, Cont, Pkg, and NameCert coordinates transport, replay, source, and name the finite refinement packet, so no omitted completion, choice, bridge, Lean, or Real-completeness principle can be exported by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformEntourageRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformEntourageRefinement}^{\uparrow}$ packet is generated from a RealUniformStructure source row, a requested Real entourage, a positive dyadic request radius, a smaller dyadic radius, a RealUniformEntourageBase containment row, a symmetric-shrink row, a CauchyFilter handoff, StreamName windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealUniformEntourageRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 2064.1, Definition 2064.2, and Definition 2064.3 seed the Real-specific refinement route from requested entourages to CauchyFilter consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealUniformEntourageRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, open-cover compactness, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, requested-entourage grounding, smaller-radius selection, containment refinement, symmetric shrink, Cauchy-filter handoff, classifier transport, and provenance non-escape proofs for the same packet.

Chapter 2065

A Concrete Naming Certificate for $\mathsf{KuratowskiCompletion}^{\uparrow}$

Following Kuratowski's metric embedding theorem, Kelley-style completion, and Bishop–Bridges constructive analysis, the $\mathsf{KuratowskiCompletion}^{\uparrow}$ packet records the completion route that first embeds a source metric object into displayed distance-function rows and then seals the resulting Cauchy data as finite BEDC completion evidence. It sits after the distance-function surface of chapter 7569, the completion interface of chapter 1506, the universal handoff of chapter 1456, and the separated uniqueness boundary of chapter 1790. The chapter names the L10 metric-completion landing surface without choosing quotient representatives, asserting host function extensionality, invoking countable choice, using open-cover compactness, or assuming ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2065.1 (Kuratowski completion carrier)**.** *A* $\mathsf{KuratowskiCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, E, D, S, A, U, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row, including the finite endpoint, distance, and Cauchy-window requests that will be completed.*
2. *E is the accepted* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *row that turns the source metric observations into displayed distance-function target rows.*
3. *D is the distance-function readback row, exposing the* $\mathsf{Real}^{\uparrow}$ *distance observations used by the embedded Cauchy comparisons.*
4. *S is the completion-seal row for the embedded Cauchy route, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *rather than through a quotient object.*
5. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *extension row that supplies the universal completion handoff for downstream consumers.*
6. *U is the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *uniqueness row, recording that separated readback identifies only the displayed completion route.*
7. *R is the real-distance readback row that returns finite distance observations from the sealed completion surface to the terminal* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source, embedding, distance-function, completion-seal, adjunction, separated-reflection, real-readback, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from source metric reads through Kuratowski distance functions, completion sealing, adjunction extension, separated uniqueness, and real-distance readback.*
10. *P is the* Pkg *provenance over* $\mathsf{KuratowskiCompletion}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host function extensionality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KuratowskiCompletionUp.KuratowskiCompletionTasteGate single carrier alignment`

Theorem 2065.2 (Kuratowski completion NameCert obligations)**.** *For an accepted* $\mathsf{KuratowskiCompletion}^{\uparrow}$ *carrier*

$$K = (M, E, D, S, A, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source metric row, Kuratowski embedding row, distance-function readback row, completion-seal row, adjunction extension row, separated-reflection uniqueness row, real-distance readback row, transport, replay, provenance, and local naming row;*
2. *source metric exposure, requiring every endpoint, distance request, and Cauchy-window request to enter through M before any completion-facing row is read;*
3. *distance-function embedding exposure, requiring E and D to display the Kuratowski distance-function route before completion sealing is available;*
4. *completion-seal exactness, requiring S to bind the embedded Cauchy route as a* $\mathsf{MetricCompletion}^{\uparrow}$ *surface and not as a quotient representative;*
5. *adjunction extension handoff, requiring A to expose every downstream extension consumer only after the same M, E, D, S route has been displayed;*
6. *separated uniqueness readback, requiring U and R to return uniqueness and finite distance reads through separated-reflection and* $\mathsf{Real}^{\uparrow}$ *rows;*
7. *structural exhaustion, requiring H, C, P, N to preserve, replay, source, and name only the listed route.*

No NameCert obligation is stored outside these rows, and none exports quotient completion equality, selected representatives, host function extensionality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2065.1. The accepted packet displays exactly the source, embedding, distance-function, completion-seal, adjunction, separated-reflection, real-readback, and structural rows.

Source metric exposure is read from M. Since every endpoint, distance request, and Cauchy-window request enters before the Kuratowski embedding row, the packet cannot manufacture completion data from an unrecorded metric object.

Distance-function embedding exposure is read from E and D. The embedding row supplies the displayed Kuratowski distance-function surface, while the distance-function readback row records the corresponding $\mathsf{Real}^{\uparrow}$ observations before the completion seal is inspected.

Completion-seal exactness and adjunction extension are read from S and A. The row S seals the embedded Cauchy route as a $\mathsf{MetricCompletion}^{\uparrow}$ object, and the row A exposes extension consumers only downstream of that same finite route.

Separated uniqueness and real-distance readback are then finite row checks. The row U records uniqueness only through the separated-reflection boundary, and R returns distance observations through the displayed real-distance surface.

Finally H, C, P, N add no analytic coordinate. They transport, replay, source, and name the displayed route, so the excluded quotient, representative, host-extensionality, choice, compactness, and ambient-completeness principles have no row from which a consumer could read them. □

Theorem 2065.3 (Kuratowski completion metric-completion handoff)**.** *Let*

$$K = (M, E, D, S, A, U, R, H, C, P, N)$$

be an accepted $\mathsf{KuratowskiCompletion}^{\uparrow}$ *carrier. Every L10 metric-completion consumer of K receives the completion object only through the finite route*

$$M \longmapsto E, D \longmapsto S \longmapsto A \longmapsto U, R \longmapsto H, C, P, N.$$

Thus the handoff from Kuratowski distance functions to metric completion is a displayed NameCert route over source metric rows, distance-function embedding rows, completion-seal rows, adjunction extension rows, separated-reflection uniqueness rows, real-distance readback rows, and structural hsame/Cont/Pkg/NameCert *rows. It is not quotient completion equality, not selected-representative data, not host function extensionality, not countable choice, not open-cover compactness, and not an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Fix an L10 metric-completion consumer of the packet. Projection gives the source metric row M, so the consumer starts from the displayed endpoint, distance, and Cauchy-window requests.

Apply the Kuratowski embedding row E and distance-function readback row D. These rows move the source metric observations into displayed distance-function data before any completion-facing object is available.

The completion-seal row S is therefore attached to the embedded route. Since S is a row of the finite carrier, it gives a $\mathsf{MetricCompletion}^{\uparrow}$ seal and not a quotient class with a selected representative.

The adjunction extension row A supplies the universal completion handoff. Any downstream extension consumer must read through the same M, E, D, S chain before seeing A.

Separated uniqueness and real-distance readback are downstream of that handoff. The rows U and R expose only separated-reflection uniqueness and displayed $\mathsf{Real}^{\uparrow}$ distance observations.

Transport, replay, provenance, and local naming close the route. The rows H, C, P, N preserve the listed data componentwise, so no host function extensionality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2065.1, Definition 2065.2, and Definition 2065.3 seed the finite Kuratowski distance-function route into metric-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host function extensionality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-function embedding exposure, completion-seal exactness, adjunction extension handoff, separated uniqueness readback, real-distance readback, and structural replay for BEDC.Derived.KuratowskiCompletionUp.

Constructive story: A $\mathsf{KuratowskiCompletion}^{\uparrow}$ packet is generated from source metric rows, Kuratowski distance-function embedding rows, completion-seal rows, adjunction extension rows, separated-reflection uniqueness rows, Real distance-readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2066

A Concrete Naming Certificate for $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$

Following Mac Lane on Kleisli categories, Kelley on completion, and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$ packet names the finite L10 boundary where the completion monad's unit-and-bind route is compared with the completion adjunction's universal handoff. It sits after the monad packet of chapter 378, the Kleisli consumer packet of chapter 1845, and the universal-mapping packet of chapter 1456. Its analytic readback is the same finite route through chapter 126, chapter 291, chapter 301, and chapter 38.

The packet is a boundary certificate, not a construction of quotient completion. It records only the displayed rows by which a downstream Real or Completion consumer can see that a Kleisli bind read and an adjunction extension read share the same finite source, window, tolerance, readback, seal, transport, replay, provenance, and local naming route.

Definition 2066.1 (Cauchy completion Kleisli-adjunction carrier)**.** *A* $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, B, L, A, V, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the unit row inherited from* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*, displaying the source packet before any Kleisli consumer or adjunction comparison is read.*
2. *B is the bind row inherited from* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*, displaying the completion-facing source family consumed by a single Kleisli request.*
3. *L is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *consumer row, recording the finite consumer request that reads the unit and bind rows through the same downstream completion surface.*
4. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *comparison row, aligning the Kleisli consumer read with the adjunction's source-side and completion-side observations.*
5. *V is the universal handoff row, exposing the adjunction extension surface only after U, B, L, A have been displayed.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the shared completion read.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same finite window.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the displayed window and tolerance data.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R have been displayed.*

10. *H records componentwise* hsame *transport for the unit, bind, Kleisli consumer, adjunction comparison, universal handoff, window, tolerance, readback, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$U, B \longmapsto L \longmapsto A, V \longmapsto S, D, R, E \longmapsto H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, B, L, A, V, S, D, R, E, H, C, P, N$. The carrier has no coordinate for quotient completions, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionKleisliAdjunctionUp.CauchyCompletionKleisliAdjunctionTasteGate`
`single carrier alignment`

Theorem 2066.2 (Cauchy completion Kleisli-adjunction obligations)**.** *For every accepted* CauchyCompletionKleisliA *carrier*

$$K = (U, B, L, A, V, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unit row, bind row, Kleisli consumer row, adjunction comparison row, universal handoff row, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming row;*

2. *unit and bind exposure, requiring every Kleisli or adjunction consumer to display U and B before it can inspect the shared downstream route;*

3. *Kleisli consumer exactness, requiring L to consume only the displayed unit-and-bind data before opening the adjunction comparison;*

4. *adjunction comparison exactness, requiring A and V to compare the Kleisli consumer with the universal handoff over the same finite source route;*

5. *analytic readback exactness, requiring S, D, R, E to supply the only StreamName, DyadicRatCore, RegSeqRat, and Real rows visible to the consumer;*

6. *structural exhaustion, requiring H, C, P, N to preserve, replay, source, and name the listed route without adding quotient completions, selected limits, countable choice, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2066.1. The accepted packet displays exactly the unit, bind, Kleisli consumer, adjunction comparison, universal handoff, finite analytic readback, and structural rows.

Unit and bind exposure is read from U and B. Since both rows are inherited from the completion monad surface, every consumer starts with the same displayed source packet and source family before any Kleisli or adjunction row is available.

Kleisli exactness is read from L. The row records the consumer request downstream of U and B, so it cannot introduce a second source family, selected limit, or quotient representative before the comparison row is opened.

Adjunction comparison and universal handoff are read from A and V. These rows align the Kleisli consumer with the adjunction boundary only over the displayed U, B, L route, and the handoff is exposed only after that comparison has been recorded.

The analytic readback rows are S, D, R, E. They give the finite StreamName window, dyadic tolerance ledger, regular rational readback, and terminal Real seal used by both sides of the comparison.

Finally H, C, P, N add no analytic or quotient coordinate. They transport, replay, source, and name the same finite route, so the excluded completion, choice, host-equality, ambient-completeness, Lean, and bridge objects have no row from which a consumer could read them. □

Theorem 2066.3 (Cauchy completion Kleisli-adjunction route)**.** *For every accepted* CauchyCompletionKleisliAdjunc *carrier*

$$K = (U, B, L, A, V, S, D, R, E, H, C, P, N),$$

every L10 Completion or Real consumer that compares a Kleisli bind read with an adjunction handoff factors through the finite route

$$U, B \longmapsto L \longmapsto A, V \longmapsto S, D, R, E \longmapsto H, C, P, N.$$

Thus the comparison is a displayed NameCert route over the completion monad, Kleisli consumer, completion adjunction, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, Real seal, hsame *transport,* Cont *replay,* Pkg *provenance, and local naming rows. It is not quotient completion equality, not a selected limit, not countable choice, not host equality, not ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and not a bridge credential.*

Proof. Fix an L10 consumer that compares a Kleisli bind read with an adjunction handoff. Projection of the accepted carrier supplies the unit row U and bind row B, so the comparison begins at the displayed completion monad data.

The Kleisli consumer row L is the next coordinate. It reads only the already displayed unit-and-bind route before the adjunction comparison is opened.

The adjunction comparison and universal handoff are then forced through A and V. These rows compare the Kleisli consumer with the completion adjunction over the same finite source route, so no separate completion object is inserted between them.

The analytic readback is downstream of that comparison. The rows S, D, R, E provide the finite stream window, dyadic tolerance, regular rational readback, and Real seal that both the Kleisli side and the adjunction side must share.

Transport, replay, provenance, and naming close the route. The rows H, C, P, N preserve exactly the listed coordinates, so the consumer receives only the displayed NameCert route and none of the excluded quotient, selection, choice, host-equality, ambient-completeness, or bridge data. □

Theorem 2066.4 (Cauchy completion Kleisli-adjunction Real non-escape)**.** *For every accepted* $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$ *carrier*

$$K = (U, B, L, A, V, S, D, R, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-facing read that uses the Kleisli-adjunction comparison reaches the terminal Real seal only through the displayed route of Definition 2066.3. In particular, the Real read cannot export a quotient completion, selected limit, countable-choice selector, host equality of completion names, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge-checked surface.*

Proof. Start with a Real-facing read of the packet. The route theorem forces the read through U, B, L, A, V before any analytic row can be inspected.

The only analytic rows after that comparison are S, D, R, E. The stream window S, dyadic tolerance row D, and regular rational readback R are displayed before the terminal Real seal E is available.

Now inspect the structural rows. Componentwise transport H and continuation replay C preserve the same finite route, while P and N source and name only the listed completion monad, Kleisli, adjunction, StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Since the carrier contains no quotient-completion coordinate, selected-limit coordinate, countable-choice selector, host-equality row, ambient-completeness theorem, Lean verification row, or bridge row, no Real-facing consumer can export any of those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2066.1, Definition 2066.2, Definition 2066.3, and Definition 2066.4 seed the finite comparison route between Cauchy-completion Kleisli consumers and adjunction handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit and bind exposure, Kleisli consumer exactness, adjunction comparison exactness, universal handoff exactness, analytic readback exactness, structural replay, and Real non-escape for BEDC.Derived.CauchyCompletionKleisliAdjunctionUp.

Constructive story: A $\mathsf{CauchyCompletionKleisliAdjunction}^{\uparrow}$ packet is generated from a completion monad unit row, bind row, Kleisli consumer row, completion adjunction comparison row, universal handoff row, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2067

A Concrete Naming Certificate for $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$ packet records the finite route by which located observations of a real sequence become a Cauchy-criterion read before a terminal real-name seal is consumed. It sits after the finite-window discipline of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy readiness surface of chapter 1590, the Real seal boundary of chapter 38, and the exact completion boundary of chapter 1120. The packet is a located finite-observation criterion for Real and Completion consumers; it does not identify quotient streams, select a limit, invoke countable choice, decide trichotomy, compare host real values, or assert ambient Real completeness.

Definition 2067.1 (Located real Cauchy criterion carrier)**.** *A* $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located source-window row, exposing the finite real-sequence entries, locatedness witnesses, and precision requests inspected by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened from the located source windows in L.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed StreamName windows and no other stream source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-cell row, recording requested radii, split tolerances, late-pair budgets, and dyadic comparison cells.*
5. *Q is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *output row, tying the located source windows, stream windows, regular readback, and dyadic tolerance cells to an accepted Cauchy-criterion read.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, S, R, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the located source windows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance cells, RealCauchyCriterion output, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located precision request through L, S, R, D, Q, E.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCompletionExactBoundaryUp}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, detached completeness, or an ambient Real-completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedRealCauchyCriterionUp.TasteGate.LocatedRealCauchyCriterionTasteGate single carrier alignment`

Theorem 2067.2 (Located real Cauchy criterion NameCert obligations)**.** *For an accepted* LocatedRealCauchyCriter *carrier*

$$K = (L, S, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the located source-window row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance cells, RealCauchyCriterion output, Real seal, transport, replay, provenance, and local naming row;*
2. *located-window exactness, requiring every observed real-sequence entry and locatedness witness to enter through L before any StreamName window or regular rational readback is inspected;*
3. *dyadic tolerance exactness, requiring D to display the requested radius, split tolerance, late-pair budget, and comparison cells used by the criterion;*
4. *CauchyCriterion handoff, requiring Q to read only the displayed route L, S, R, D before the Real seal E is available;*
5. *seal nonescape, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to pass through E rather than through a quotient stream, selected limit, or host real object;*
6. *transport and replay stability, requiring componentwise* hsame *transport H and displayed* Cont *replay C to preserve the same ordered route L, S, R, D, Q, E;*
7. *provenance exhaustion, requiring P and N to name only this displayed route and to export no countable choice, trichotomy, host real equality, detached completeness, or ambient completion principle.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2067.1. The accepted packet displays the located source windows, StreamName windows, regular rational readback, dyadic tolerance cells, CauchyCriterion output, Real seal, and the structural rows H, C, P, N.

Located-window exactness is read from the order of L, S, R. The source observations and locatedness witnesses appear in L before S opens any finite stream window, and R reads only through those displayed windows.

Dyadic tolerance exactness is the role of D. The tolerance row stores the requested radius, split budget, late-pair comparison, and dyadic cells before the CauchyCriterion output row may be consumed.

The handoff to $\mathsf{RealCauchyCriterion}^{\uparrow}$ is then finite. The row Q binds the displayed L, S, R, D route as the only criterion output that can reach the terminal seal E.

Seal nonescape follows from the carrier inventory. The terminal $\mathsf{Real}^{\uparrow}$ row is named as E, while quotient streams, selected limits, and host real objects are not coordinates of the packet.

Transport and replay are structural. Componentwise hsame in H preserves the listed rows, and Cont replay in C repeats the same route from a located precision request through L, S, R, D, Q, E.

Finally, provenance is exhausted by P and N. Since the carrier has no row for countable choice, trichotomy, host real equality, detached completeness, or ambient completion, the local NameCert surface cannot export any such object. □

Theorem 2067.3 (Located real Cauchy criterion terminal seal route)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$ *packet*

$$K = (L, S, R, D, Q, E, H, C, P, N)$$

factors through the displayed finite route

$$L \longmapsto S \longmapsto R \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the terminal real-name seal is reached through located source windows, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells, the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *output, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$ *row. It exports no quotient stream, selected limit, countable choice, trichotomy, host real equality, or ambient Real completeness.*

Proof. Begin with a terminal $\mathsf{Real}^{\uparrow}$ read of an accepted packet. Projection gives the ordered rows $L, S, R, D, Q, E, H, C, P, N$, so the read starts at the located source-window surface rather than at an unbounded host stream.

The StreamName and RegSeqRat stages are downstream of L. The finite windows S are opened from the located source row, and the regular rational readback R is obtained only through those windows.

The dyadic tolerance row D is downstream of that readback. It records the split tolerance and late-pair comparison cells required before the CauchyCriterion output can be accepted.

The CauchyCriterion handoff is Q. By Definition 2067.2, Q may read only L, S, R, D before exposing the terminal Real seal E.

Transport, replay, provenance, and local naming do not add analytic coordinates. The rows H, C, P, N preserve, replay, source, and name the same displayed route.

The excluded objects are absent from the carrier and from the NameCert obligations. Therefore no quotient stream, selected limit, countable choice, trichotomy, host real equality, or ambient Real completeness can be exported by a terminal seal read. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealCauchyCriterion}^{\uparrow}$ packet is generated from located source-window rows, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance cells, a RealCauchyCriterion output row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 2067.1, Definition 2067.2, and Definition 2067.3 seed the located finite-observation route from Real sequence windows to the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, located-window exactness, dyadic tolerance exactness, CauchyCriterion handoff, seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LocatedRealCauchyCriterionUp.

Chapter 2068

A Concrete Naming Certificate for CauchyConvergenceModulusTailMonotonicity$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the CauchyConvergenceModulusTailMonotonicity$^{\uparrow}$ packet records the finite certificate by which a displayed CauchyConvergenceModulus$^{\uparrow}$ tail threshold may be replaced by a later displayed threshold without changing the convergence-modulus route consumed by Real and Completion arguments. It refines the convergence-modulus carrier of chapter 1438, the monotone schedule discipline of chapter 1744, the regular tail-extension surface of chapter 1750, the finite windows of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet gives later L10 Real and Completion consumers a monotone reuse certificate for displayed convergence moduli; it does not import host convergence, selected limits, quotient streams, countable choice, trichotomy, host real equality, or ambient Real completeness.

Definition 2068.1 (Cauchy convergence modulus tail monotonicity carrier)**.** *A* CauchyConvergenceModulusTailMon *carrier is a finite* BHist *packet*

$$K = (K_0, T, W, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *K_0 is the source* CauchyConvergenceModulus$^{\uparrow}$ *packet whose source row, dyadic tolerance row, tail-index schedule, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, transport, replay, provenance, and local name are already displayed.*
2. *T is the monotone tail-threshold comparison row, naming an earlier displayed threshold of K_0, a later displayed threshold, and the finite comparison showing that the later threshold lies inside the same accepted tail.*
3. *W is the* StreamName$^{\uparrow}$ *finite-window reuse row, containing only windows whose indices are admitted by the later displayed threshold in T.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the reused finite windows in W and from the source route of K_0.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, reusing the tolerance cells of K_0 and recording that the later tail request consumes no stronger radius than the displayed convergence-modulus request.*
6. *L is the* RealSequenceLimit$^{\uparrow}$ *handoff row reached only after K_0, T, W, R, D have been displayed.*

7. *H records componentwise* hsame *transport for the source convergence-modulus packet, tail-threshold comparison, finite-window reuse, regular readback, dyadic tolerance ledger, limit handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the earlier convergence-modulus threshold through the later threshold, the reused windows, the regular readback, the dyadic tolerance ledger, and the RealSequenceLimit handoff.*

9. *P is the* Pkg *provenance over* CauchyConvergenceModulusTailMonotonicity$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, CauchyModulusMonotoneNormalization$^{\uparrow}$, RegularCauchyTailMonotonicity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealSequenceLimit$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K_0, T, W, R, D, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, host convergence, ambient Real completeness, or a detached completion object.

Lean4 ✓ `BEDC.Derived.CauchyConvergenceModulusTailMonotonicityUp.TasteGate.CauchyConvergenceModulusTailM`
`single carrier alignment`

Theorem 2068.2 (Cauchy convergence modulus tail monotonicity NameCert obligations)**.** *For an accepted* CauchyConvergenceModulusTailMonotonicity$^{\uparrow}$ *carrier*

$$K = (K_0, T, W, R, D, L, H, C, P, N),$$

the local NameCert$_{\text{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the source* CauchyConvergenceModulus$^{\uparrow}$ *packet, monotone tail-threshold comparison, finite* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RealSequenceLimit$^{\uparrow}$ *handoff, transport, replay, provenance, and local naming row;*

2. *monotone tail extension, requiring the later displayed threshold in T to be read only as a later tail for the same convergence-modulus request carried by K_0;*

3. *tolerance preservation, requiring D to reuse the dyadic tolerance cells displayed by K_0 and to record no stronger tolerance demand;*

4. *finite-window reuse, requiring W and R to inspect only windows and readbacks admitted by the later threshold and the same source route;*

5. *handoff nonescape, requiring L to be downstream of K_0, T, W, R, D and to export no quotient stream, selected limit, host convergence row, host real equality row, or ambient completeness principle;*

6. *transport stability, requiring componentwise* hsame *in H and displayed* Cont *replay in C to preserve the ordered route through the same convergence-modulus consumer surface;*

7. *provenance exhaustion, requiring P and N to name only the rows listed in the carrier and to export no countable choice, trichotomy, detached completion object, standard bridge, or Lean verification claim.*

No tail-monotonicity obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2068.1. The accepted packet displays the source convergence-modulus row, the later-threshold comparison, finite windows, regular readback, tolerance ledger, RealSequenceLimit handoff, and the structural rows H, C, P, N.

Monotone tail extension is the role of T. The row does not choose a new convergence modulus; it reads an earlier displayed threshold from K_0, displays a later threshold for the same request, and records the finite comparison between them.

Tolerance preservation is supplied by D. Since the later threshold is downstream of the same convergence-modulus request, the dyadic cells already displayed by K_0 remain the only tolerance rows available to the consumer.

Finite-window reuse follows from W and R. The windows are opened only beyond the later displayed threshold, and the regular readback is obtained only through those windows and the source route of K_0.

The handoff row L is terminal for this packet. It is reached only after the source packet, later-threshold comparison, windows, readback, and tolerance ledger have been displayed, so quotient streams, selected limits, host convergence, host real equality, and ambient completeness have no route into the handoff.

Finally H, C, P, N preserve, replay, source, and name the same ordered packet. Because no excluded object is a carrier coordinate, provenance exhaustion leaves exactly the listed obligations. □

Theorem 2068.3 (Cauchy convergence modulus tail monotonicity stability)**.** *Let*

$$K = (K_0, T, W, R, D, L, H, C, P, N)$$

be an accepted $\mathsf{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}$ *carrier. Replacing the displayed convergence-modulus threshold in K_0 by the later displayed threshold named by T preserves the same convergence-modulus consumer route:*

$$K_0 \longmapsto T \longmapsto W \longmapsto R \longmapsto D \longmapsto L \longmapsto H, C, P, N.$$

Every Real or Completion consumer that accepted the earlier displayed threshold may therefore read the later threshold through the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RealSequenceLimit}^{\uparrow}$ *route, with no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, host convergence, ambient Real completeness, Lean verification, or standard bridge.*

Proof. Start with the source convergence-modulus packet K_0. Its carrier route already exposes the source row, dyadic tolerance request, tail schedule, finite StreamName windows, RegSeqRat readback, RealSequenceLimit handoff, and structural rows.

Read the comparison row T. It identifies the later threshold as a displayed later tail for the same tolerance request, so the consumer does not receive a second modulus, a choice-selected threshold, or a host convergence predicate.

The finite observations then pass through W and R. Since W contains only windows admitted by the later threshold and R reads only those windows through the same source route, replacing the threshold leaves the consumer on the displayed convergence-modulus path.

The tolerance ledger D keeps the dyadic budget unchanged. The later threshold may discard earlier tail positions, but it does not demand a new tolerance row or a stronger radius.

Finally, the RealSequenceLimit handoff and structural rows are downstream of the same route. The rows L, H, C, P, N preserve, replay, source, and name the replacement, and the carrier has no coordinate for any excluded quotient, choice, host, completeness, Lean, or bridge object. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2068.1, Definition 2068.2, and Definition 2068.3 seed the finite monotone reuse route for displayed Cauchy convergence-modulus tail thresholds.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier exposure, monotone tail extension, tolerance preservation, finite-window reuse, handoff nonescape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CauchyConvergenceModulusTailMonotonicityUp.

Constructive story: A $\mathsf{CauchyConvergenceModulusTailMonotonicity}^{\uparrow}$ packet is generated from a source CauchyConvergenceModulus packet, a monotone tail-threshold comparison, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, RealSequenceLimit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2069

A Concrete Naming Certificate for $\mathsf{JordanVariationDecomposition}^{\uparrow}$

Following Jordan, Rudin, and Folland, the $\mathsf{JordanVariationDecomposition}^{\uparrow}$ packet records the finite signed-variation split visible inside a bounded-variation ledger. It sits over the bounded-variation surface of chapter 398, the real-name seal of chapter 38, the regular rational stream surface of chapter 301, the stream-read discipline of chapter 126, the dyadic observation rows of chapter 291, and the finite integrator handoff of chapter 1637. The packet names only displayed finite partition ledgers and their positive and negative dyadic edge rows; it does not introduce a variation measure, a supremum over all partitions, a quotient function space, countable choice, or a standard bridge.

Definition 2069.1 (Jordan variation decomposition carrier)**.** *A* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, \Pi, X, D, V^{+}, V^{-}, S, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet and its finite partition-variation ledger.*
2. *Π is the displayed ordered partition on which the signed split is read.*
3. *X records the endpoint* $\mathsf{Real}^{\uparrow}$ *rows or their* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name representatives.*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-difference observations attached to adjacent endpoints of* Π*.*
5. *V^{+} is the finite positive-edge dyadic sum ledger.*
6. *V^{-} is the finite negative-edge dyadic sum ledger.*
7. *S records the signed reconstruction row, pairing the two finite ledgers back to the displayed endpoint-difference ledger.*
8. *R records displayed common-refinement comparison rows for transported partitions and transported signed ledgers.*
9. *H records componentwise* hsame *transport for the bounded-variation source, partition, endpoint, dyadic, positive-sum, negative-sum, reconstruction, refinement, package, and naming rows.*
10. *C records displayed* Cont *replay from a bounded-variation request to endpoints, edge observations, signed ledgers, reconstruction, refinement comparison, and downstream exposure.*

11. *P is the* Pkg *provenance over* $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanVariationDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, \Pi, X, D, V^+, V^-, S, R, H, C, P, N$. It has no coordinate for variation measures, host suprema over all partitions, quotient function spaces, countable choice, hidden partition families, or bridge checking.

Lean4 ✓ `BEDC.Derived.JordanVariationDecompositionUp.JordanVariationDecompositionTasteGate`
`single carrier alignment`

Theorem 2069.2 (Jordan variation decomposition NameCert obligations)**.** *For an accepted* JordanVariationDecomp *carrier $J = (B, \Pi, X, D, V^+, V^-, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{JordanVariationDecomposition}^{\uparrow}}$ *surface consists of exactly five rows:*

1. *carrier admission for the displayed tuple $B, \Pi, X, D, V^+, V^-, S, R, H, C, P, N$;*
2. *finite positive-edge and negative-edge ledger formation from the dyadic observations in D;*
3. *signed reconstruction through S, returning the displayed endpoint-difference ledger on Π;*
4. *common-refinement transport through R, H, C for the source bounded-variation packet, endpoint rows, dyadic observations, signed ledgers, and reconstruction row;*
5. *no-escape from variation measures, host suprema over all partitions, quotient function spaces, countable choice, hidden partition families, and bridge checking.*

These rows are read through the finite route

$$B, \Pi, X, D \longmapsto V^+, V^- \longmapsto S \longmapsto R$$

together with H, C, P, N.

Proof. Carrier admission is the projection of Definition 2069.1. The bounded-variation source and displayed partition are present before any signed ledger is formed.

Finite signed ledger formation is edgewise. Each adjacent endpoint pair in Π is read through X, its dyadic difference row is read through D, and that row is placed in exactly one of the positive-edge or negative-edge finite ledgers.

The reconstruction row S then forgets the sign split only after both finite ledgers are visible. Thus the public endpoint-difference read returns to the displayed bounded-variation ledger rather than to a measure-valued object.

For common refinements, use R to transport the partition and edge rows, then replay the same signed split through H and C. Package and naming provenance are carried componentwise, so no extra comparison coordinate is introduced.

The excluded host objects do not occur among the carrier coordinates. The certificate therefore closes only the finite signed-variation ledger and its displayed transport surface. □

Theorem 2069.3 (Jordan variation decomposition bounded-variation route)**.** *Every public read of a* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *packet factors through a displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, a displayed finite partition ledger, endpoint rows from* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, stream reads from* $\mathsf{StreamName}^{\uparrow}$*, dyadic edge observations from* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the finite signed reconstruction row. A consumer such as the* $\mathsf{RiemannStieltjes}^{\uparrow}$ *integrator handoff receives only these finite rows and their visible* BHist, hsame, Cont, Pkg, *and* NameCert *provenance.*

Proof. Start from the source row B. By chapter 398, the bounded-variation read is already restricted to a displayed partition, endpoint names, dyadic edge observations, finite variation ledgers, common-refinement rows, and visible provenance.

Project the carrier of Definition 2069.1. Its additional data are only the positive-edge ledger V^+, the negative-edge ledger V^-, and the reconstruction row S attached to that same displayed partition.

Endpoint reads remain on the real and regular-rational surfaces of chapter 38 and chapter 301. Stream access is supplied by chapter 126, and every edge observation is supplied by chapter 291.

Finally, downstream integration reads through chapter 1637 can consume the signed reconstruction only as a finite integrator ledger. Since the route never ranges over all partitions and never creates a quotient of functions or a measure carrier, the public read remains inside the displayed bounded-variation dependency surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{JordanVariationDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanVariationDecomposition}^{\uparrow}$ packet is generated from a bounded-variation source row, a displayed finite partition, endpoint real or regular-rational names, stream-read windows, dyadic edge observations, positive and negative finite variation ledgers, signed reconstruction, common-refinement transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanVariationDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2069.1, Definition 2069.2, and Definition 2069.3 seed the finite Jordan signed-variation boundary over $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RiemannStieltjes}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanVariationDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close variation measures, host suprema over all partitions, quotient function spaces, countable choice, hidden partition families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed finite-ledger formation, signed reconstruction, common-refinement transport, replay, provenance, and no-escape for BEDC.Derived.JordanVariationDecompositionUp.

Chapter 2070

A Concrete Naming Certificate for $\mathsf{BoundedVariationCompletion}^{\uparrow}$

Following Jordan's bounded-variation decomposition and Bishop–Bridges constructive analysis, the $\mathsf{BoundedVariationCompletion}^{\uparrow}$ packet names the finite completion-facing surface by which bounded-variation data may be consumed as Real/Completion rows. It sits over the bounded-variation carrier of chapter 398, the regulated-function approximation schedule of chapter 1906, the Darboux finite-sum boundary of chapter 1880, the Riemann–Stieltjes tagged-sum handoff of chapter 1637, and the terminal real-name seal of chapter 38. The packet is a finite certificate surface for BV completion reads; it does not introduce host function equality, a quotient of completed functions, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2070.1 (Bounded-variation completion carrier)**.** *A* $\mathsf{BoundedVariationCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, R, D, I, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *V is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its finite partition-variation evidence, endpoint transports, common-refinement rows, and local* $\mathsf{NameCert}_{\mathsf{BoundedVariation}^{\uparrow}}$ *provenance.*
2. *R is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *representative row carrying the displayed step-approximation schedule, tolerance-tail ledger, and finite-sum handoff through the regulated-function surface.*
3. *D is the Darboux ledger row, read through chapter 1880, recording the upper and lower finite-sum windows used by completion-facing approximation requests.*
4. *I is the Riemann–Stieltjes handoff row, read through chapter 1637, connecting the regulated representative and bounded-variation integrator rows to a finite tagged-sum route.*
5. *S records the* $\mathsf{StreamName}^{\uparrow}$ *read windows used to expose the requested finite approximation stages.*
6. *Q records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows by which the stream windows are returned as regular rational names.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded-variation, regulated, Darboux, Riemann–Stieltjes, stream, and regular-rational rows have all been displayed.*

8. *H records componentwise* hsame *transport for the bounded-variation source, regulated representative, Darboux ledger, Riemann–Stieltjes handoff, stream windows, regular-rational readback, real seal, package, and local naming rows.*

9. *C records displayed* Cont *replay from a bounded-variation completion request to the regulated representative, from the representative to finite integration ledgers, from those ledgers to stream and regular-rational readback, and from readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationCompletion}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, R, D, I, S, Q, E, H, C, P, N$. It has no coordinate for a host function equality predicate, quotient completed function, selected completion limit, countable-choice selector, open-cover compactness proof, or ambient completeness witness for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BoundedVariationCompletionUp.BoundedVariationCompletionTasteGate single carrier alignment`

Theorem 2070.2 (Bounded-variation completion NameCert obligations)**.** *For every accepted* $\mathsf{BoundedVariationCompletion}^{\uparrow}$ *carrier $K = (V, R, D, I, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationCompletio}}$ *surface consists of exactly six obligations:*

1. *carrier admission for the displayed tuple $V, R, D, I, S, Q, E, H, C, P, N$;*

2. *variation-Cauchy compatibility, requiring each completion-facing tolerance request to be witnessed by finite bounded-variation partition rows and transported common-refinement ledgers from V;*

3. *regulated representative handoff, requiring R to expose a step-approximation schedule and tolerance-tail row before any completion read reaches D or I;*

4. *integration-ledger compatibility, requiring the Darboux row D and the Riemann–Stieltjes row I to consume the same displayed representative, partition, and bounded-variation evidence;*

5. *classifier transport through H and replay through C for every bounded-variation, regulated, integration, stream, regular-rational, real-seal, package, and naming row;*

6. *ledger nonescape from host function equality, quotient completed functions, selected limits, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

These obligations are read through the finite route

$$V, R \longmapsto D, I \longmapsto S, Q \longmapsto E$$

together with H, C, P, N.

Proof. Carrier admission is the projection of Definition 2070.1. The rows V and R expose the bounded-variation source and regulated representative before any integration or completion-facing row can be read.

Variation-Cauchy compatibility is finite at each tolerance request. The request first selects displayed partition and common-refinement rows from V, then uses the transported endpoint and variation ledgers to bound the requested window without invoking a completed function object.

The regulated representative handoff is supplied by R. Its step schedule and tolerance-tail row must be present before the Darboux and Riemann–Stieltjes ledgers can be exposed, so the integration route is attached to a visible regulated-function approximation rather than to a host quotient of functions.

For integration compatibility, D and I consume the same displayed representative and bounded-variation evidence. The Darboux row records finite upper and lower sum windows, while the Riemann–Stieltjes row records the tagged finite handoff; neither row adds a hidden selector or measure-valued carrier.

Transport and replay are exactly the rows H and C. They move each displayed coordinate componentwise and preserve the route from V, R through D, I, S, Q to E.

The excluded host objects do not occur among the carrier coordinates. Hence the certificate obligations close only the displayed finite Real/Completion packet. □

Theorem 2070.3 (Bounded-variation completion real-seal route)**.** *Every Real/Completion consumer route for an accepted* $\mathsf{BoundedVariationCompletion}^{\uparrow}$ *packet factors through the displayed bounded-variation row* V*, regulated representative row* R*, Darboux and Riemann–Stieltjes rows* D *and* I*,* $\mathsf{StreamName}^{\uparrow}$ *windows* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* Q*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The route exposes only those rows and their visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

No consumer route in this scope uses host function equality, a quotient completed function, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real/Completion consumer request. The continuation row C first replays the bounded-variation source V and the regulated representative R, so the request is already attached to finite partition-variation evidence and a displayed approximation schedule.

The request then reaches the integration ledgers. The Darboux row D supplies finite upper and lower approximation windows, while the Riemann–Stieltjes row I supplies the tagged-sum handoff against the same bounded-variation source and regulated representative.

Only after those rows are present does the stream and regular-rational readback route appear. The windows in S expose finite stream stages, and Q returns the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ rows.

Finally the terminal row E supplies the $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport through H and replay through C preserve this order, while package and naming provenance add no bypass coordinate.

Since the route has displayed every coordinate used by the carrier, no host function equality, quotient completed function, selected limit, countable-choice selector, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness can enter the consumer boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedVariationCompletion}^{\uparrow}$ packet is generated from bounded-variation evidence, regulated representative data, Darboux and Riemann–Stieltjes approximation ledgers, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2070.1, Definition 2070.2, and Definition 2070.3 seed the finite Real/Completion packet over $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BoundedVariationCompletionUp`

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient completed functions, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-Cauchy compatibility, regulated representative handoff, integration-ledger compatibility, classifier transport, ledger nonescape, and field faithfulness for BEDC.Derived.BoundedVariationCompletionUp.

Chapter 2071

A Concrete Naming Certificate for $\mathsf{CauchyTailBound}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyTailBound}^{\uparrow}$ packet isolates the finite evidence that a modulus-controlled Cauchy read stays inside a displayed tail tolerance before any Real-and-Completion consumer reads the terminal seal. It sits between the threshold discipline of chapter 1423, the regular Cauchy criterion of chapter 1611, the dyadic tail envelope of chapter 1659, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later $\mathsf{Real}^{\uparrow}$ and completion-facing arguments a reusable tail-bound certificate; it does not assert quotient stream equality, select a limit, compare host real values, invoke countable choice, use trichotomy, prove open-cover compactness, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2071.1 (Cauchy tail bound carrier)**.** *A* $\mathsf{CauchyTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, S, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row naming the stage from which the Cauchy tail must remain within the requested tolerance.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row inspected only at stages allowed by the threshold row M.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radius, comparison slack, dyadic envelope cells, and threshold-to-window accounting.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed tail window and dyadic ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the threshold, window, tolerance, and regular readback rows have been consumed.*
6. *B is the displayed tail-bound ledger, naming the finite assertion that every inspected pair of tail positions beyond M is contained by the dyadic tolerance recorded in D.*
7. *H records componentwise* hsame *transport for the threshold, tail-window, dyadic ledger, regular readback, Real seal, tail-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto R \longmapsto E \longmapsto B.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, D, R, E, B, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyTailBoundUp.CauchyTailBoundCarrier`
Lean4 ✓ `BEDC.Derived.CauchyTailBoundUp.CauchyTailBoundTasteGate single carrier alignment`

Theorem 2071.2 (Cauchy tail bound NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailBound}^{\uparrow}$ *carrier*

$$T = (M, S, D, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the threshold row, tail-window row, dyadic tolerance ledger, regular readback, Real seal boundary, tail-bound ledger, transport, replay, provenance, and local naming row;*
2. *threshold-window exposure, requiring every inspected tail window in S to expose the relevant* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row M before any pair of tail observations is compared;*
3. *dyadic tolerance accounting, requiring D to record the requested tolerance, comparison slack, and envelope cells consumed by the displayed tail-bound ledger B;*
4. *regular readback, requiring R to be read only from the threshold-approved windows and dyadic tolerance rows already displayed in M, S, D;*
5. *Real seal non-escape, requiring the* $\mathsf{Real}^{\uparrow}$ *seal boundary E to be downstream of M, S, D, R, B and to export no quotient stream, selected limit, host equality, choice, trichotomy, compactness, or ambient completeness row;*
6. *transport, replay, provenance, and local naming, requiring H, C, P, N to preserve, replay, source, and name only the finite rows listed in the carrier.*

No Cauchy tail-bound obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2071.1. The accepted packet displays exactly the threshold, tail-window, dyadic ledger, readback, Real seal, tail-bound ledger, and structural rows.

Threshold-window exposure is read from the pair M, S. Since S is opened only at stages allowed by the threshold row, a consumer cannot compare arbitrary stream positions or import an unbounded quotient stream before the finite modulus evidence is visible.

Dyadic tolerance accounting is supplied by D and B. The dyadic ledger names the requested radius, slack, and envelope cells, while the tail-bound ledger records that the inspected tail pairs consume precisely those displayed tolerance rows.

Regular readback and Real seal non-escape follow from the route through R and E. The readback row is downstream of M, S, D, and the Real seal is downstream of the readback and tail-bound ledger, so the seal cannot export selected limits, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally H, C, P, N add only componentwise transport, displayed replay, provenance, and local naming. Because the carrier has no coordinate for any excluded principle, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyTailBoundUp.CauchyTailBoundCarrier namecert obligations`

Theorem 2071.3 (Cauchy tail bound finite route)**.** *Let*

$$T = (M, S, D, R, E, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyTailBound}^{\uparrow}$ *carrier. Every Real-and-Completion consumer of the tail-bound packet factors through the finite route*

$$M, S, D \longmapsto R \longmapsto B \longmapsto E \longmapsto H, C, P, N.$$

Equivalently, the consumer reads only the displayed $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{StreamName}^{\uparrow}$ *tail window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, tail-bound ledger,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Proof. Fix a Real-and-Completion consumer of an accepted carrier. Projection gives the threshold row M, the finite tail-window row S, and the dyadic tolerance ledger D.

The threshold and window rows force the finite route. The consumer may inspect only those tail positions whose window has been opened after the threshold is displayed, and the dyadic ledger accounts for the requested tolerance before any tail comparison is exported.

The readback and tail-bound ledger are downstream of this data. The row R reads the regular rational observations from S and D, while B records the displayed bound on inspected tail pairs beyond M.

The Real seal is terminal for this packet. Since E is reached only after M, S, D, R, B, and H, C, P, N only transport, replay, source, and name the same route, no quotient stream, selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness escapes through the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailBound}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2071.1, Definition 2071.2, and Definition 2071.3 seed the finite tail-bound route between modulus-controlled Cauchy reads and Real-and-Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailBound}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyTailBoundUp.CauchyTailBoundCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, public closure, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold-window exposure, dyadic tolerance accounting, regular readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailBoundUp.

Constructive story: A $\mathsf{CauchyTailBound}^{\uparrow}$ packet is generated from a RealCauchyModulus threshold row, StreamName tail windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal boundary, a displayed tail-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2072

A Concrete Naming Certificate for $\mathsf{RegularCauchySpread}^{\uparrow}$

Following Brouwer on spreads and Bishop–Bridges on regular Cauchy real names, the $\mathsf{RegularCauchySpread}^{\uparrow}$ packet records the finite spread-law windows from which a $\mathsf{RegSeqRat}^{\uparrow}$ consumer and then a $\mathsf{Real}^{\uparrow}$ seal may be read. It is downstream of the finite-prefix discipline of chapter 1677, the finite request schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal discipline of chapter 38. The packet names finite spread observations and regular Cauchy handoff data; it is not a completed branch, quotient stream, selected limit, countable-choice route, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2072.1 (Regular Cauchy spread carrier)**.** *A* $\mathsf{RegularCauchySpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the spread-law row. It records the decidable admitted-prefix tree and the finite extension checks inherited from the* $\mathsf{BrouwerSpread}^{\uparrow}$ *surface.*
2. *W is the finite window row. Each displayed request is a StreamName-scheduled spread prefix and carries only the observations listed in the packet.*
3. *T is the dyadic tolerance row. It records the requested precision, denominator budget, comparison witnesses, and tolerance refinements through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It turns each accepted finite spread window and tolerance row into the corresponding regular rational observation without selecting an infinite branch.*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the terminal finite-window seal exposed after the* $\mathsf{RegSeqRat}^{\uparrow}$ *row has been displayed and after the dyadic tolerance checks have been replayed.*
6. *H stores componentwise* hsame *transport for the spread law, finite windows, dyadic tolerances, regular rational readback, Real seal, continuation replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a finite precision request through S, W, T, R, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySpread}^{\uparrow}$*,* $\mathsf{BrouwerSpread}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySpread}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $S, W, T, R, L, H, C, P, N$. The carrier has no coordinate for quotient streams, selected branches, countable choice, host real equality, open-cover compactness, ambient Real completeness, or equality of unseen tails.

Lean4 ✓ `BEDC.Derived.RegularCauchySpreadUp.RegularCauchySpreadTasteGate single carrier alignment`

Theorem 2072.2 (Regular Cauchy spread NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySpread}^{\uparrow}$ *carrier*

$$G = (S, W, T, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySpread}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the spread law, finite windows, dyadic tolerance ledger, regular rational readback, Real seal, transport, replay, provenance, and local naming;*
2. *spread-window admissibility, requiring every window in W to be licensed by the finite spread-law row S and by the StreamName schedule visible in the packet;*
3. *dyadic regularity, requiring every precision request in T to provide the finite denominator budget and comparison witnesses needed by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
4. *seal handoff, requiring every public* $\mathsf{Real}^{\uparrow}$ *seal in L to be read only after S, W, T and R have been displayed and replayed;*
5. *structural non-escape, requiring H, C, P, N to preserve the same finite route under* hsame*, replay it under* Cont*, record exact provenance, and export no quotient stream, selected branch, countable-choice selector, host real equality, open-cover compactness, ambient Real completeness, Lean verification, or standard bridge.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchySpreadUp.TasteGate.RegularCauchySpreadTasteGate single carrier alignment`

Proof. Carrier admission is the projection from Definition 2072.1. The accepted packet exposes exactly the rows $S, W, T, R, L, H, C, P, N$.

Spread-window admissibility is internal to S and W. A displayed request first checks the decidable spread-law row, then opens only the finite StreamName-scheduled observations present in the window row.

Dyadic regularity is internal to T and R. The tolerance row supplies the denominator budget, comparison data, and refinements that the regular rational readback uses; no infinite branch or quotient stream is introduced.

The seal handoff is downstream of those rows. A public $\mathsf{Real}^{\uparrow}$ consumer reaches L only after the spread window, tolerance ledger, and $\mathsf{RegSeqRat}^{\uparrow}$ readback have been replayed.

The remaining clauses are structural. The row H transports each displayed coordinate componentwise, C replays the same finite route, and P, N source and name the packet. Since the excluded objects are not coordinates of G, the five rows exhaust the local NameCert surface. □

Theorem 2072.3 (Regular Cauchy spread RegSeqRat–Real handoff)**.** *Every accepted* $\mathsf{RegularCauchySpread}^{\uparrow}$ *packet exposes a* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer through the displayed finite spread windows and dyadic tolerance rows*

$$S, W, T, H, C, P, N,$$

and exposes a downstream $\mathsf{Real}^{\uparrow}$ *seal only through the already displayed regular rational readback row* R *and seal row* L*. Consequently the packet cannot export a quotient stream, selected branch, countable-choice selector, host real equality, open-cover compactness, ambient Real completeness, Lean verification, or standard bridge.*

Proof. Begin with an accepted packet $G = (S, W, T, R, L, H, C, P, N)$. The only route to a regular rational observation is through the spread-law row S, the finite window row W, and the dyadic tolerance row T.

Replay the continuation row C. It checks the admitted spread prefix, opens the finite StreamName-scheduled window, and then reads the denominator budget and comparison witnesses recorded in T.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is therefore a consumer of displayed finite data rather than a selector for an infinite branch. Componentwise transport in H preserves that route under hsame.

The $\mathsf{Real}^{\uparrow}$ seal is later in the packet. A public seal read reaches L only after the regular rational row R has been displayed, and provenance through P, N names the same finite route. There is no separate coordinate from which the excluded quotient, choice, equality, compactness, completeness, verification, or bridge principles can be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySpread}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySpread}^{\uparrow}$ packet is generated from a spread-law row, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySpread}^{\uparrow}}$, seed)

Scope closed: Definition 2072.1, Definition 2072.2, and Definition 2072.3 seed the finite-window route from Brouwer spread laws to regular rational reads and Real seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySpread}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected branches, countable choice, host real equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or standard bridges.

Upgrade path: Obligation closure requires checked carrier admission, spread-window admissibility, dyadic regularity, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.RegularCauchySpreadUp.

Chapter 2073

A Concrete Naming Certificate for $\mathsf{RealCompletionFunctor}^{\uparrow}$

Following Bishop–Bridges constructive real analysis and the Kelley–Bourbaki language of completion as a universal extension surface, the $\mathsf{RealCompletionFunctor}^{\uparrow}$ packet records the terminal functorial exit of the L10 Real and Completion phase. It ties the four finite faces $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ to the existing $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ and $\mathsf{CompletionUniversalProperty}^{\uparrow}$ rows without importing quotient completions, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The packet sits after the dyadic tolerance rows of chapter 291, the finite request windows of chapter 126, the regular rational readbacks of chapter 301, the terminal real seal of chapter 38, the functorial completion surface of chapter 423, and the extension-uniqueness surface of chapter 397.

Definition 2073.1 (RealCompletionFunctor carrier)**.** *A* $\mathsf{RealCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, W, Q, E, F, U, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the inspected dyadic endpoints, radius requests, and comparison rows used by every downstream completion read;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the objectwise requests that the completion route is allowed to inspect;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the selected windows into regular rational Cauchy evidence;*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same D, W, Q route;*
5. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *unit-and-bind row, read only after the four L10 faces D, W, Q, E have been displayed;*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-uniqueness row attached to the same completion functor row and the same terminal real seal;*
7. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, real seal, functor row, universal-property row, replay, provenance, and local naming row;*
8. *C records displayed* Cont *replay of the route*

$$D, W, Q \longmapsto E \longmapsto F \longmapsto U.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionFunctor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionFunctor}^{\uparrow}}$ *ledger.*

The classifier compares accepted packets only through $D, W, Q, E, F, U, H, C, P, N$. It has no coordinate for quotient completion classes, selected limit representatives, countable choice, host-level equality of real names, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RealCompletionFunctorUp.RealCompletionFunctorTasteGate single carrier alignment`

Theorem 2073.2 (RealCompletionFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionFunctor}^{\uparrow}$ *carrier*

$$R = (D, W, Q, E, F, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionFunctor}^{\uparrow}}$ *surface consists exactly of seven finite obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, stream-window row, regular rational readback, terminal real seal, completion-functor row, universal-property row, transport, replay, provenance, and local naming row;*
2. *four-face visibility, requiring every completion consumer to display the D, W, Q, E route before reading the functor or universal-property rows;*
3. *dyadic tolerance exactness, requiring all admitted radius and comparison requests to be read from D through the same displayed windows W;*
4. *stream and regular readback compatibility, requiring the W requests and Q readbacks to name the same finite regular-Cauchy route;*
5. *terminal real seal compatibility, requiring E to be consumed only as the terminal seal of the displayed D, W, Q route;*
6. *functorial handoff, requiring F to supply the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *unit-and-bind surface only after D, W, Q, E have been displayed;*
7. *universal-property exactness, requiring U to expose extension and uniqueness only for the same E, F route and only through H, C, P, N.*

No $\mathsf{RealCompletionFunctor}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2073.1. The accepted packet displays exactly the dyadic, stream-window, regular-readback, terminal-seal, functorial, universal-property, and structural rows.

Four-face visibility is read from D, W, Q, E. The dyadic tolerance row supplies the allowed radius requests, the stream row supplies finite windows, the regular row supplies rational Cauchy readback, and the terminal real row seals that route before the completion functor is inspected.

The functorial handoff is read from F. Since F is downstream of the same four faces, the unit and bind schedules cannot be replaced by a quotient completion object or by a selected host limit.

The universal-property row is then read from U. Its extension and uniqueness data are tied to the displayed real seal and functor row, while H, C, P, N transport, replay, source, and name only those visible rows.

These projections exhaust the carrier coordinates, so no additional obligation can be read from the packet. □

Theorem 2073.3 (RealCompletionFunctor terminal seal non-escape)**.** *Let*

$$R = (D, W, Q, E, F, U, H, C, P, N)$$

be an accepted $\mathsf{RealCompletionFunctor}^{\uparrow}$ *carrier. Every terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer and every completion-facing functor consumer factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow}$$

The route exports no quotient completion, selected limit, countable-choice selector, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden completion object.*

Proof. Start with a terminal-seal or completion-facing consumer and project the accepted carrier. The only terminal real seal is E, and it is named only after the dyadic tolerance row D, stream-window row W, and regular rational readback row Q have been displayed.

The consumer can reach functorial completion data only through F. The row F is therefore a handoff from the four-face L10 route to the existing $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ unit-and-bind surface, not a separate completed carrier.

The extension-uniqueness consumer must then pass through U. This row is attached to the same E, F pair, so existence and uniqueness are read as finite universal-property data over the displayed route.

Finally H, C, P, N preserve only componentwise transport, replay, provenance, and local naming. Since no carrier coordinate names a quotient completion, a selected limit, host equality, countable choice, ambient completeness, or a hidden completion object, the displayed route is the whole exported terminal seal surface. □

Theorem 2073.4 (RealCompletionFunctor obligation closure)**.** *The rows*

$$D, W, Q, E, F, U, H, C, P, N$$

of an accepted $\mathsf{RealCompletionFunctor}^{\uparrow}$ *carrier jointly discharge carrier admission, four-face visibility, dyadic tolerance exactness, stream and regular readback compatibility, terminal real seal compatibility, functorial handoff, universal-property exactness, and ledger non-escape. Consequently every* $\mathsf{RealCompletionFunctor}^{\uparrow}$ *consumer reads the terminal functorial exit only through the displayed dyadic, stream, regular, Real, completion-functor, universal-property, transport, replay, provenance, and local NameCert rows.*

Proof. Carrier admission is Definition 2073.1: the accepted packet has no coordinates beyond $D, W, Q, E, F, U, H, C, P, N$.

The obligation rows are exactly those isolated in Definition 2073.2. They force the four-face route D, W, Q, E to be displayed before the completion functor row F and universal-property row U are read.

Finally apply Definition 2073.3. It identifies the same terminal seal route and excludes quotient completions, selected limits, countable-choice selectors, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, and hidden completion objects from the exported ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionFunctor}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 2073.1, Definition 2073.2, Definition 2073.3, and Definition 2073.4 close the finite functorial exit from $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ into the completion-functor and universal-property rows at obligation strength.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCompletionFunctor}^{\uparrow}}$, theoremChecked)
Lean target: `BEDC.Derived.RealCompletionFunctorUp.RealCompletionFunctorTasteGate\_single\_carrier\_alignment`
Bridge status: none
Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, or a public standard bridge.
Upgrade path: Obligation closure requires checked carrier admission, four-face visibility, dyadic tolerance exactness, stream and regular readback compatibility, terminal real seal compatibility, functorial handoff, universal-property exactness, and ledger non-escape for the RealCompletionFunctor$^{\uparrow}$ packet.
Constructive story: A RealCompletionFunctor$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite-window row, RegSeqRat readback row, terminal RealUp seal, CauchyCompletionFunctor unit-and-bind row, CompletionUniversalProperty extension-uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The RealModulusOfConvergence$^{\uparrow}$ hub reads its convergence-rate row as the finite schedule consumed by chapter 1515 and chapter 301 before any Real$^{\uparrow}$ seal or completion consumer may use it.

Chapter 2074

A Concrete Naming Certificate for $\mathsf{RealModulusOfConvergence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet records the explicit convergence-modulus ledger used by the Real and Completion route. It sits over the dyadic tolerance surface of chapter 291, the finite window service of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite NameCert ledger for modulus-controlled convergence; it is not a proof of Real completeness, not a quotient of streams, and not a choice principle.

The uniqueness-facing consumer is the regular Cauchy route of chapter 3390, which reads the same dyadic tolerance, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal rows before comparing representatives.

Before the Real seal is read, this packet depends explicitly on the finite window route of chapter 126 and the regular rational handoff of chapter 301. Those sibling rows supply the L10 lattice edge that turns a displayed modulus request into pre-seal stream and readback data rather than a detached Real claim.

The completion-facing consumers are chapter 1558 and chapter 1820. The former reads the modulus row as a filter-limit tolerance schedule, while the latter reuses the displayed tolerance, finite-window, and readback rows before equality-facing $\mathsf{Real}^{\uparrow}$ seals are compared. Thus the same finite ledger supplies both completion and equality routes without quotient stream equality, selected representatives, or host real limits.

Definition 2074.1 (Real modulus of convergence carrier)**.** *A* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, W, Q, M, E, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic precision request and the comparison ledger used to judge later approximants.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observable segment of the stream family inspected for that tolerance.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the windowed stream data becomes a regular rational sequence request.*
4. *M is the monotone convergence-modulus ledger, recording that later tolerance requests read a weakly later finite window and never shorten the accepted tail.*

5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff exposed only after D, W, Q, M have been displayed.*
6. *H records componentwise* hsame *transport for tolerance, window, readback, modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a convergence request through D, W, Q, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusOfConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, Q, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 2074.2 (Real modulus of convergence NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *carrier $R = (D, W, Q, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealModulusOfConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window readback, regular rational handoff, monotone modulus-ledger exactness, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N$$

before using the convergence modulus as evidence for a Real-facing seal.

Lean4 ✓ `BEDC.Derived.Real modulus of convergenceUp.RealModulusOfConvergenceCarrier namecert obligations`

Proof. Project the carrier of Definition 2074.1. The dyadic row D supplies the tolerance request, and the window row W is the only finite observation service available to answer it.

The regular rational handoff Q is then read through the monotone ledger M. Monotonicity is a row condition inside M: a later tolerance request must move to a weakly later accepted window, so the packet cannot introduce a detached tail selector.

The Real seal row E is downstream of D, W, Q, M. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so no quotient equality of streams, selected host limit, countable-choice schedule, host real equality, or ambient completeness theorem can be exported by the local classifier. □

Theorem 2074.3 (Real modulus of convergence CauchyModulus handoff)**.** *Every accepted* RealModulusOfConverge *read factors its modulus request through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row before it reaches the terminal Real-facing seal. The route is*

$$\mathsf{CauchyModulus}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Thus every Real or Completion consumer sees the convergence request as Cauchy-modulus data, regular rational readback, dyadic tolerance accounting, the terminal $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert. *It exports no selected tail, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 2074.1. The dyadic tolerance row D, finite-window row W, regular rational readback row Q, and monotone ledger M are the displayed data from which the modulus request is read.

Use the existing Cauchy modulus dependency route. It identifies the request as a $\mathsf{CauchyModulus}^{\uparrow}$ handoff before the regular rational row Q and the dyadic row D are consumed by the Real-facing seal.

Apply Definition 2074.2. The terminal row E is downstream of the displayed modulus, readback, and tolerance rows, and H, C, P, N only transport, replay, source, and name that same route.

Since the carrier has no coordinate for a selected tail, quotient real equality, countable-choice schedule, or ambient Real completeness, the handoff cannot export any of those objects to a Real or Completion consumer. □

Theorem 2074.4 (Real modulus of convergence tail stability)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet. For any later dyadic tolerance request transported through the monotone ledger M, the corresponding read uses a weakly later finite* $\mathsf{StreamName}^{\uparrow}$ *window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff route, and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary before any Real or Completion consumer may use the convergence claim. The routed tail-stability surface is exactly*

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N.$$

It contains no quotient stream equality, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 2074.1. The rows D, W, Q expose the tolerance request, the finite stream window, and the regular rational handoff available before any Real-facing seal is consumed.

Use the monotonicity row inside M. A later dyadic tolerance request may only move the accepted read to a weakly later finite window, so the packet preserves the tail already admitted by the earlier request and never introduces a separate tail selector.

Finally route the resulting read through E, H, C, P, N. The seal row E remains downstream of the displayed tolerance, window, readback, and modulus rows, while H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate supplies quotient stream equality, selected limits, countable choice, or ambient Real completeness, those objects cannot be exported to the Real or Completion consumer. □

Theorem 2074.5 (Real modulus of convergence window composition)**.** *For an accepted* $\mathsf{RealModulusOfConvergence}$ *carrier $R = (D, W, Q, M, E, H, C, P, N)$, each convergence-modulus read composes the dyadic tolerance row D, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The composed pre-seal route is*

$$D, W, Q \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

It exports no quotient equality of streams and makes no choice of a host limit.

Proof. Read the carrier of Definition 2074.1. The dyadic tolerance row D, finite-window row W, and regular rational readback row Q are displayed before the monotone modulus ledger and before the Real-facing seal.

The composition step is local to those three rows. The tolerance request in D determines the comparison scale, W supplies only a finite observed window at that scale, and Q turns that finite window into the regular rational readback consumed by the modulus ledger.

Only after this pre-seal read may the route pass through M and then to E, H, C, P, N. Because the carrier has no coordinate for quotient stream equality or a selected host limit, the composed window route cannot identify stream representatives or choose a limit outside the displayed NameCert data. □

Theorem 2074.6 (Real modulus of convergence monotone-window refinement)**.** *Let $R = (D, W, Q, M, E, H, C, P,$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *carrier, and let two displayed dyadic tolerance requests be related by the monotone ledger M. The later request refines the earlier one only through a weakly later* $\mathsf{StreamName}^{\uparrow}$ *window, preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and reaches the same* $\mathsf{Real}^{\uparrow}$ *seal boundary before any Real or Completion consumer may use the modulus schedule. The refinement route is*

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N.$$

It exports no quotient stream, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 2074.1. The two tolerance requests are displayed through the same dyadic row D, so their comparison is available only through the packet's monotone ledger M.

Use tail stability for the later request and window composition for the resulting read. Theorem 2074.4 forces the later request to use a weakly later finite $\mathsf{StreamName}^{\uparrow}$ window, while Theorem 2074.5 keeps the dyadic row, the window row, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row on the same pre-seal route.

Finally route the refined read through E, H, C, P, N. The visible $\mathsf{Real}^{\uparrow}$ seal boundary is reached only after D, W, Q, M have been displayed, and no carrier coordinate supplies quotient streams, selected limits, countable choice, or ambient Real completeness. □

Theorem 2074.7 (Real modulus RegSeqRat uniqueness handoff)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ and $R' = (D', W', Q', M', E', H', C', P', N')$ be accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packets. If the two packets expose the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the same monotone modulus ledger, then every* $\mathsf{RegSeqRat}^{\uparrow}$ *uniqueness consumer and every* $\mathsf{Real}^{\uparrow}$ *seal consumer reads the same finite convergence route*

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N.$$

Equivalently, the uniqueness-facing consumer cannot distinguish R from R' except through the displayed handoff rows D, W, Q, M and their downstream seal, transport, replay, provenance, and local naming rows. The handoff exports no quotient stream, selected limit, countable-choice schedule, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the two carriers through Definition 2074.1. The hypotheses identify the tolerance row, finite-window row, regular rational readback row, and monotone ledger, so both packets present the same pre-seal route D, W, Q, M to any uniqueness-facing consumer.

Apply Definition 2074.4. Later tolerance requests must use weakly later finite windows and preserve the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff route, so the identified rows cannot split into two independent tails.

Apply Definition 2074.5 and Definition 2074.6. These results keep the dyadic tolerance, finite window, readback row, and monotone ledger on the same route before the Real-facing seal is reached.

The downstream rows E, H, C, P, N are therefore read only as seal, transport, replay, provenance, and local naming for that route. Since the carrier excludes quotient streams, selected limits, countable choice, host Real equality, and ambient completeness, neither a $\mathsf{RegSeqRat}^{\uparrow}$ uniqueness consumer nor a $\mathsf{Real}^{\uparrow}$ seal consumer can read any further distinguishing datum. □

Theorem 2074.8 (Real modulus Cauchy-equivalence setoid factorization)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ and $R' = (D', W', Q', M', E', H', C', P', N')$ be accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packets. If the two packets agree on their displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, their* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, their* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and their monotone modulus ledger, then every equality-facing consumer factors through* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The factorization route is*

$$D, W, Q, M \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto E \longmapsto H, C, P, N.$$

It exports no quotient stream, selected limit, countable-choice schedule, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the two carriers through Definition 2074.1. The hypotheses identify exactly the displayed tolerance, window, readback, and monotone ledger rows, so an equality-facing consumer sees one shared pre-seal route rather than two independent convergence reads.

Apply Definition 2074.2. The Real-facing seal E is downstream of D, W, Q, M, while H, C, P, N only transport, replay, source, and name the same finite route.

Use Definition 2074.7. Since the $\mathsf{RegSeqRat}^{\uparrow}$ readback and monotone ledger are identified, the uniqueness-facing consumer cannot distinguish the packets before the equality boundary is read.

The equality boundary is therefore the displayed setoid route of chapter 1820. Because the carrier contains no quotient streams, selected limits, countable choice, host Real equality, or ambient completeness coordinate, the comparison factors through $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal and exports none of those excluded objects. □

Theorem 2074.9 (Real modulus of convergence CauchyFilterLimit route)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet. Every* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *consumer of R reads the convergence-modulus schedule only through the shared finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The modulus row supplies the filter-limit tolerance schedule, the finite window row supplies the basis inspected by the filter-limit request, and the regular rational readback row supplies the only pre-seal comparison. The route exports no quotient stream, selected host limit, countable-choice schedule, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Theorem 2074.2. Carrier admission exposes the dyadic tolerance row D, finite window row W, regular rational readback row Q, monotone modulus ledger M, Real seal E, and structural rows H, C, P, N.

Apply Theorem 2074.3. The modulus row is first consumed as the tolerance schedule for the filter-limit request, so the request cannot start from a selected host limit or from ambient completeness.

Use Theorem 2074.4 and Theorem 2074.6. These keep the filter-limit tolerance schedule on a weakly later $\mathsf{StreamName}^{\uparrow}$ window while preserving the same $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

Finally apply Theorem 2074.7 and the regular seal handoff of Definition 1558.3. The downstream consumer reaches the $\mathsf{CauchyFilterLimit}^{\uparrow}$ seal only through the same $\mathsf{Real}^{\uparrow}$ boundary and structural rows, and the carrier contains no quotient stream, selected host limit, countable-choice schedule, host Real equality, or ambient completeness coordinate. □

Theorem 2074.10 (Real modulus of convergence tail comparison route)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ and $R' = (D', W', Q', M', E', H', C', P', N')$ be accepted $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packets. Suppose their dyadic tolerance rows and monotone ledgers compare through the same $\mathsf{CauchyModulus}^{\uparrow}$ request, and suppose the later-tail comparison is read only after the displayed $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ handoff rows are exposed. Then every later-tail comparison between R and R' routes through*

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N.$$

The comparison consumes $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order, and it exports no quotient stream equality, selected limit, countable-choice schedule, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum.

Proof. Apply Theorem 2074.3. The shared tolerance and ledger comparison must first be read as a $\mathsf{CauchyModulus}^{\uparrow}$ request before the regular rational row or Real-facing seal can be consumed.

Apply Theorem 2074.4. Once the request is later-tail-facing, the read may only move to weakly later finite $\mathsf{StreamName}^{\uparrow}$ windows while preserving the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff route.

Apply Theorem 2074.6. The monotone ledgers keep the dyadic tolerance, finite window, readback, and modulus rows on one route before the Real-facing seal is reached.

Finally apply Theorem 2074.7. The identified pre-seal rows force every uniqueness-facing and Real-facing consumer to read the same downstream seal, transport, replay, provenance, and local naming rows. Since the carrier contains none of quotient streams, selected limits, countable choice, host Real equality, or ambient completeness, the later-tail comparison cannot export any such datum. □

Theorem 2074.11 (Real modulus of convergence Cauchy-tail exhaustion)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet. Every later-tail comparison accepted by R is exhausted by the $\mathsf{CauchyModulus}^{\uparrow}$ request, the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{Real}^{\uparrow}$ seal, and the structural rows H, C, P, N. No quotient stream equality, selected limit, countable-choice schedule, host Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or extra metric-completion row is exported by the comparison.*

Proof. Project the carrier of Definition 2074.1. The only pre-seal rows available to a later-tail comparison are the dyadic tolerance row D, finite-window row W, regular rational readback row Q, and monotone ledger M.

Apply Theorem 2074.3. The comparison must first be read as a $\mathsf{CauchyModulus}^{\uparrow}$ request, so it cannot begin from a selected limit or from an ambient Real equality.

Use Theorem 2074.4. Later requests move only to weakly later $\mathsf{StreamName}^{\uparrow}$ windows and preserve the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff route.

Apply Theorem 2074.5 and Theorem 2074.6. These keep the dyadic tolerance ledger, finite window, readback row, and monotone comparison on one displayed route before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally apply Theorem 2074.10. The downstream rows E, H, C, P, N supply only the Real-facing seal, transport, replay, provenance, and local naming for that route. Since no carrier coordinate records quotient streams, selected limits, countable choice, host Real equality, ambient

completeness, or a separate metric-completion row, the accepted later-tail comparison is exhausted by the stated data. □

Theorem 2074.12 (Real modulus of convergence Bishop interface forward route)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet. Every read of R sourced from the Bishop real-line interface first consumes chapter 2062, then the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and* $\mathsf{Real}^{\uparrow}$ *seal E before it reads the convergence-modulus ledger M and structural rows H, C, P, N. The route exports no quotient stream, selected limit, countable-choice schedule, host Real equality, detached Bishop real-line object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 2074.1. The only rows available before the modulus ledger are the dyadic tolerance row, finite-window row, regular rational readback row, and terminal Real seal.

Read the source through chapter 2062. That interface already exposes dyadic endpoints and tolerances, finite stream windows, regular readback, and a Real seal before a real-line consumer may use later comparison data.

Now apply the local handoff theorems. The NameCert obligations place $D, W, Q, M, E, H, C, P, N$ in order, while the Cauchy-modulus handoff and tail-stability route keep the modulus request behind the displayed finite window and regular readback.

Thus a Bishop-interface-sourced read reaches M only after the same D, W, Q, E route has been consumed. Since the carrier has no coordinate for quotient streams, selected limits, choice schedules, host equality, detached Bishop objects, or ambient completeness, none of those objects can be exported by the forward route. □

Theorem 2074.13 (Real modulus of convergence tail dominance)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet. If a later displayed request dominates an earlier convergence-modulus request through the monotone ledger M, then every Real or Completion consumer follows the same route*

$$\mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

with a weakly later finite window. The dominant request adds no second choice schedule, no selected tail, no quotient stream equality, no host Real equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 2074.1. The earlier and later requests are both displayed through the dyadic tolerance row D and compared only through the monotone ledger M.

Apply Theorem 2074.3. The request is read as $\mathsf{CauchyModulus}^{\uparrow}$ data before any regular rational handoff or Real-facing seal is consumed.

Use Theorem 2074.4 and Theorem 2074.6. Since the later request dominates the earlier one, the accepted read moves only to a weakly later $\mathsf{StreamName}^{\uparrow}$ window and preserves the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff route.

Finally apply Theorem 2074.10. The downstream consumer reaches the same $\mathsf{Real}^{\uparrow}$ seal through $D, W, Q, M, E, H, C, P, N$. Because the carrier has no coordinate for a second choice schedule, selected tails, quotient stream equality, host Real equality, or ambient completeness, dominance cannot export any such datum. □

Theorem 2074.14 (Real modulus CauchySequenceSpace factorization)**.** *Let $R = (D, W, Q, M, E, H, C, P, N)$ be an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet. Every* $\mathsf{CauchySequenceSpace}^{\uparrow}$*-facing consumer*

of R factors through the same displayed route

$$D, W, Q, M \longmapsto E \longmapsto H, C, P, N.$$

The $\mathsf{CauchySequenceSpace}^{\uparrow}$ *family row, schedule row, and common-window ledger may read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, and the* $\mathsf{Real}^{\uparrow}$ *seal row E only through that route. The factorization exports no quotient stream, selected limit, countable-choice schedule, host Real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or detached metric-completion row.*

Proof. Apply Theorem 2074.2. It exposes exactly the dyadic tolerance row, finite-window row, regular rational readback row, monotone modulus ledger, Real seal, and structural rows available to any downstream consumer.

Apply Theorem 2074.3. The $\mathsf{CauchySequenceSpace}^{\uparrow}$ common-window request therefore reads the displayed packet first as convergence-modulus data, before the regular rational family or Real seal can be consumed.

Use Theorem 2074.4, Theorem 2074.5, and Theorem 2074.6. These keep the family row, schedule row, and common-window ledger on the same D, W, Q, M route with a weakly later finite window when the tolerance request is refined.

Finally apply Theorem 2074.7. The regular rational readback and the Real seal cannot split into a second representative route. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, host Real equality, ambient Real completeness, or a detached metric-completion row, no $\mathsf{CauchySequenceSpace}^{\uparrow}$-facing consumer can export any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusOfConvergence}^{\uparrow}$ Real or Completion consumer factors through one displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one monotone convergence-modulus ledger, one $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before it may use a convergence-modulus claim.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RealModulusOfConvergence}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately supply tolerance comparisons, finite windows, regular rational readback, and terminal Real sealing, while this packet uniquely binds them to a monotone convergence-modulus ledger consumed by the L10 Real and Completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusOfConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat readback row, a monotone convergence-modulus ledger, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusOfConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2074.1, Definition 2074.2, Definition 2074.4, Definition 2074.5, and Definition 2074.6 seed the explicit convergence-modulus ledger over DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusOfConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient stream equality, selected host limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window readback, RegSeqRat handoff, monotone ledger exactness, Real seal handoff, structural transport, and ledger non-escape for BEDC.Derived.RealModulusOfConvergenceUp.

The sibling dependency route for $\mathsf{RealModulusOfConvergence}^{\uparrow}$ is explicit: the carrier consumes $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ tail schedules from chapter 1438, $\mathsf{StreamName}^{\uparrow}$ finite windows from

chapter 126, RegSeqRat$^{\uparrow}$ readback from chapter 301, and terminal Real$^{\uparrow}$ seal rows from chapter 38. In the L10 completion lattice, this places the modulus packet beside the Bishop criterion surface of chapter 3404: both routes expose finite schedules, finite windows, regular rational readback, and Real-facing seals before any completion consumer may read a convergence claim. The route also reads the Cauchy packet through chapter 3413 before its convergence-rate row is available to Completion consumers, and that dependency is through displayed StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows. The Cauchy-modulus source of chapter 292 supplies the finite threshold schedule consumed before RealModulusOfConvergence$^{\uparrow}$ exposes its convergence-rate row.

Theorem 2074.15 (Real modulus of convergence located Cauchy route)**.** *Every* RealModulusOfConvergence$^{\uparrow}$ *consumer that reads a convergence-rate row must pass through the displayed located-Cauchy and completion-facing surfaces before the claim is visible to a completion consumer. The route reads the Cauchy source of chapter 3413, the tail schedule of chapter 1438, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the Bishop completion criterion surface of chapter 3404. It exports no selected limit, quotient Cauchy name, countable-choice schedule, host real equality, or ambient completion theorem.*

Proof. Project the accepted carrier of Definition 2074.1. Its convergence-rate row is available only together with the dyadic tolerance ledger, finite stream window, regular rational handoff, monotone modulus ledger, Real seal, transport, replay, provenance, and local naming rows.

Use the sibling dependency route already displayed for RealModulusOfConvergence$^{\uparrow}$. It places the convergence-rate row after CauchyConvergenceModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, and the Cauchy source route, so the modulus cannot be read as an isolated local counter.

Finally compose with Definition 2074.19. A Bishop completion consumer receives the same finite route before any completion-facing claim is visible, and the carrier has no row for selected limits, quotient names, countable choice, host real equality, or ambient completion. □

Theorem 2074.16 (Real modulus located sequence link)**.** *Any* RealModulusOfConvergence$^{\uparrow}$ *read consumed by a located sequence or Cauchy-modulus chapter must factor through the displayed located Cauchy route of Definition 2074.15, the existing Cauchy modulus dependency route, the regular Cauchy modulus route, and the RegSeqRat consumer route of Definition 2074.17. In particular, a consumer that reaches the located Cauchy sequence surface of chapter 1595 must first read the displayed* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* CauchyConvergenceModulus$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows already exposed by the* RealModulusOfConvergence$^{\uparrow}$ *carrier.*

Proof. Project the accepted RealModulusOfConvergence$^{\uparrow}$ carrier from Definition 2074.1. Its convergence-rate row is not available without the dyadic tolerance ledger, finite stream windows, regular rational readback, monotone modulus ledger, terminal Real seal, transport, replay, provenance, and local naming rows.

Apply the located Cauchy route. It places every completion-facing convergence-rate read after the Cauchy source, tail schedule, stream windows, regular readback, dyadic tolerance ledger, Real seal, and Bishop completion criterion surface.

Now use the Cauchy modulus dependency route and the regular Cauchy modulus route. These routes ensure that a Cauchy-modulus consumer reads the displayed modulus extraction rows before the Real-facing seal can treat the convergence rate as evidence.

Finally apply the RegSeqRat consumer route. A located sequence consumer, including the sibling surface of chapter 1595, therefore reaches the modulus only after the same finite RegSeqRat-

to-Real path has been displayed, and the carrier exports no selected limit, quotient stream equality, countable-choice schedule, host Real equality, or ambient completeness coordinate. □

The Cauchy-modulus dependency route for $\mathsf{RealModulusOfConvergence}^{\uparrow}$ is finite before any Real-facing consumer reads a convergence rate. The carrier consumes modulus rows from chapter 3718, chapter 4071, and the Cauchy modulus surface of chapter 292, then routes the accepted tail data through the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301. Thus the convergence-rate row is not a detached selector: it is read as displayed Cauchy-modulus extraction data before the downstream $\mathsf{Real}^{\uparrow}$ seal of chapter 38 or any Completion consumer can use it. The regular-Cauchy modulus route for $\mathsf{RealModulusOfConvergence}^{\uparrow}$ is read before any terminal Real seal is consumed: a convergence-modulus request first cites the substantive Cauchy modulus sibling of chapter 292, then passes through the RegSeqRat handoff of chapter 301 as displayed regular-Cauchy readback. This dependency keeps the packet in the finite L10 lattice, so the later $\mathsf{Real}^{\uparrow}$ seal receives a routed modulus, stream window, and regular-rational witness rather than a detached host limit, quotient stream equality, countable-choice schedule, or ambient real-completeness premise.

Theorem 2074.17 (Real modulus RegSeqRat consumer route)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer that reaches a Real-facing seal must first route through the displayed consumer surface*

$$\mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{CauchyEquivalenceSetoid}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The route reads regular rational windows, Cauchy-equivalence comparison, dyadic tolerance ledgers, finite stream windows, and the terminal Real seal before any convergence-modulus datum is used as Real-facing evidence. It exports no quotient stream equality, selected host limit, countable-choice schedule, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 2074.1. The visible rows expose dyadic tolerance data, finite stream windows, regular rational readback, a monotone modulus ledger, and a terminal $\mathsf{Real}^{\uparrow}$ seal before the structural rows are consumed.

Apply Definition 2074.8. Equality-facing reads of the modulus packet factor through $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ before the terminal Real seal, so the consumer cannot treat the packet as an isolated local ledger.

Apply Definition 2074.7. The same finite convergence route is read by $\mathsf{RegSeqRat}^{\uparrow}$ uniqueness consumers and by $\mathsf{Real}^{\uparrow}$ seal consumers, with the dyadic ledger and finite stream windows still displayed.

Thus every accepted consumer edge passes through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the seal is used. The exclusion clauses follow from the carrier and the cited factorization and handoff theorems. □

Theorem 2074.18 (Real modulus RegSeqRat-to-Real Cauchy factorization)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer whose seal is read as Cauchy-facing data factors through*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it reaches any $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *consumer. The route exposes regular rational windows, Cauchy-equivalence comparison, dyadic tolerance ledgers, finite stream names, and the terminal Real seal; it exports no quotient stream, selected limit, countable-choice schedule, host* $\mathsf{Real}^{\uparrow}$ *equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 2074.1. The Cauchy-facing seal is available only after the displayed dyadic ledger, finite stream windows, regular rational readback, modulus ledger, terminal Real seal, transport, replay, provenance, and local NameCert rows have been read.

Apply Definition 2074.17. It already forces every Real-facing consumer through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the convergence-modulus datum can be used.

Now compose the Cauchy-equivalence setoid factorization of Definition 2074.8 with the RegSeqRat uniqueness handoff of Definition 2074.7. The first places equality-facing reads before the terminal Real seal, and the second keeps the regular rational route shared by uniqueness and Real-seal consumers.

Thus a $\mathsf{Completion}^{\uparrow}$ or $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer receives the modulus packet only after the RegSeqRat-to-Real Cauchy route has been displayed. Since the carrier has no quotient-stream, selected-limit, countable-choice, host-Real-equality, or ambient-completeness coordinate, none of those principles is exported by the factorization. □

The $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet consumes its substantive route through finite stream windows, regular rational readback, dyadic tolerance rows, and the terminal real-name seal. In particular, chapter 126 supplies the displayed observation windows, chapter 301 supplies the regular rational readback attached to those windows, chapter 291 supplies the positive tolerance ledger, and chapter 38 supplies the bounded $\mathsf{Real}^{\uparrow}$ consumer boundary. The modulus chapter therefore reads convergence data from these sibling certificates before exposing any real-facing rate row.

Theorem 2074.19 (Real modulus Bishop criterion forward link)**.** *Every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet must first display the finite convergence route*

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

This forward link consumes the sibling criterion surface of chapter 3404 and the sibling modulus surface of chapter 1438. The criterion may then consume the local convergence-modulus ledger, transport, replay, provenance, and $\mathsf{NameCert}$ *rows, but it cannot read a selected limit, quotient stream equality, countable-choice schedule, host real equality, detached completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum from the packet.*

Proof. Project the accepted carrier of Definition 2074.1. The carrier exposes the dyadic tolerance row, finite StreamName window, regular-rational readback, monotone convergence-modulus ledger, Real seal, and structural rows before any completion criterion can consume it.

Use the Cauchy-modulus handoff and tail-stability routes of Definition 2074.3 and Definition 2074.4. They force a criterion request through $\mathsf{CauchyModulus}^{\uparrow}$, the displayed $\mathsf{StreamName}^{\uparrow}$ window, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before the Real seal is available.

The dyadic tolerance and Real-seal rows then remain visible by Definition 2074.5. The Bishop completion criterion receives exactly that finite route plus the local transport, replay, provenance, and naming rows, with no hidden selected limit or ambient completion principle. □

Theorem 2074.20 (Real modulus RegularCauchyUniqueness forward route)**.** *Every* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *consumer that compares two Real representatives through a* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *packet first reads the displayed dyadic tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The comparison route is*

$$\mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyU}$$

Thus the uniqueness consumer of chapter 3390 may compare representatives only after the finite modulus lattice has exposed its tolerance, window, readback, and seal data.

Proof. Read the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ carrier as the finite packet whose tolerance, window, readback, monotone ledger, and Real seal rows are already displayed in the local NameCert construction. These rows determine which tail comparison is available to a downstream consumer before any representative comparison is formed.

The $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ consumer then uses the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, the same $\mathsf{StreamName}^{\uparrow}$ finite window, and the same $\mathsf{RegSeqRat}^{\uparrow}$ handoff route before it reaches the $\mathsf{Real}^{\uparrow}$ seal. Therefore the uniqueness comparison is not a direct quotient assertion on Real representatives; it is a routed read through the modulus packet and its finite sibling rows. □

Any $\mathsf{RealModulusOfConvergence}^{\uparrow}$ completion consumer that compares nonzero tail distances must read the positive-distance and zero-refusal route of chapter 3505 before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. The route first displays the $\mathsf{MetricApartness}^{\uparrow}$ metric source, positive-distance witness, zero-distance refusal, separated classifier comparison, completion-facing transport, componentwise hsame data, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row; only then may the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ packet use its dyadic tolerance, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, monotone modulus ledger, and Real seal. Thus nonzero tail comparison is routed through a substantive metric-apartness sibling rather than through host inequality, quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 2075

A Concrete Naming Certificate for $\mathsf{SignedDigitCauchyModulus}^{\uparrow}$

Following Avizienis signed-digit arithmetic and Bishop–Bridges constructive analysis, the SignedDigitCauchyModulu packet names the finite route from signed-digit observation windows to explicit regular Cauchy modulus data. It sits over the signed-digit window surface of chapter 316, the real-facing signed-digit seal of chapter 363, the dyadic endpoint rows of chapter 291, the scheduled finite observations of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the explicit threshold surface of chapter 292. The packet records a displayed modulus for the signed-digit route; it is not a quotient of streams, a selected representative, or an ambient completeness principle.

Definition 2075.1 (SignedDigitCauchyModulus carrier)**.** *A* $\mathsf{SignedDigitCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, E, T, R, L, H, C, P, N)$$

with the following visible coordinates:

1. *S is an accepted* $\mathsf{SignedDigitStream}^{\uparrow}$ *row, carrying signed-digit finite windows, carry ledgers, and normalized prefix data;*
2. *W is the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule of finite signed-digit windows requested by downstream consumers;*
3. *E assigns to every scheduled window a normalized* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row and dyadic radius ledger;*
4. *T is a monotone threshold modulus row: for each requested precision it displays a finite index after which all scheduled endpoint reads satisfy the stated dyadic tolerance;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint rows E and the threshold row T;*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached from the same regular Cauchy readback;*
7. *H records componentwise* hsame *transport for signed-digit windows, endpoint normalization, threshold rows, regular readback rows, and real-seal rows;*
8. *C records the* Cont *replay routes from window requests to normalized endpoints, from endpoints to the threshold check, from the threshold check to R, and from R to L;*
9. *P is the* Pkg *provenance naming* $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDigitReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyModulus}^{\uparrow}$ *as dependency anchors;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDigitCauchyModulus}^{\uparrow}}$ *provenance row.*

The carrier predicate accepts exactly those packets whose signed-digit windows normalize to the displayed dyadic endpoints, whose threshold row is monotone in the requested precision, and whose regular readback and real seal consume that same threshold row through the recorded hsame, Cont, Pkg, *and local* NameCert *data.*

Lean4 ✓ `BEDC.Derived.SignedDigitCauchyModulusUp.SignedDigitCauchyModulusTasteGate single carrier alignment`

Theorem 2075.2 (SignedDigitCauchyModulus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SignedDigitCauchyModulus}^{\uparrow}}$ *seed surface consists of five finite obligations:*

1. *carrier admission: the packet is the finite tuple of Definition 2075.1;*
2. *window normalization: every displayed signed-digit window in W decodes to its named* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row in E;*
3. *threshold monotonicity: the modulus row T is monotone in precision and is read from the same finite window schedule;*
4. *regular readback exactness: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R consumes exactly the endpoint rows E and threshold row T;*
5. *seal and provenance policy: the* $\mathsf{Real}^{\uparrow}$ *seal row L and the rows H, C, P, N transport only the displayed signed-digit, dyadic, threshold, regular readback, and seal data.*

Lean4 ✓ `BEDC.Derived.SignedDigitCauchyModulusUp.SignedDigitCauchyModulusNameCertObligations`

Proof. Project the carrier tuple of Definition 2075.1. Its coordinates already separate the signed-digit source, finite window schedule, dyadic endpoint normalization, threshold modulus, regular readback, real seal, transport, replay, provenance, and local certificate rows.

The window-normalization obligation is the signed-digit finite-window route: S and W determine the finite prefixes, and E records the normalized dyadic endpoints for those prefixes. The threshold obligation is read from T, whose entries are indexed by requested precisions and transported through the same schedule.

The regular readback obligation uses only E and T. The row R therefore cannot read an endpoint outside the displayed signed-digit windows or a modulus outside the displayed threshold ledger.

Finally L, H, C, P, N provide the public seal, componentwise transport, replay, dependency package, and local naming rows. Since these rows are listed as coordinates of the same finite packet, the five obligations exhaust the seed surface. □

Theorem 2075.3 (SignedDigitCauchyModulus regular readback)**.** *Every accepted* $\mathsf{SignedDigitCauchyModulus}^{\uparrow}$ *packet exposes a regular Cauchy readback through its explicit modulus. For each requested precision, the threshold row T supplies a finite bound after which all scheduled signed-digit windows normalize to dyadic endpoint rows in E whose pairwise tolerance is accepted by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R; the* $\mathsf{Real}^{\uparrow}$ *seal row L can read the packet only through this regular readback route.*

Proof. Fix an accepted packet $M = (S, W, E, T, R, L, H, C, P, N)$ and a displayed precision request. The row T gives the corresponding finite threshold, and monotonicity keeps later precision requests on the same or later finite window surface.

Read any two scheduled windows past that threshold through W. The signed-digit row S and the endpoint ledger E normalize both finite prefixes to displayed $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows.

The regular readback row R consumes exactly those endpoint rows together with the threshold row T. Thus the late-pair tolerance required by the regular Cauchy readback is witnessed inside the finite packet.

The real seal row L is downstream of R in the recorded Cont replay. Componentwise hsame transport can move the displayed rows, but it cannot introduce a second modulus, an unobserved endpoint, or a host real limit. Therefore the public readback is the explicit-modulus regular Cauchy readback named by T and R. □

Theorem 2075.4 (SignedDigitCauchyModulus prefix induction)**.** *Let $M = (S, W, E, T, R, L, H, C, P, N)$ be an accepted* SignedDigitCauchyModulus$^{\uparrow}$ *carrier. Each accepted signed-digit prefix normalizes by finite induction on the displayed prefix and carry ledger before it can contribute to the monotone threshold row T and the* RegSeqRat$^{\uparrow}$ *readback row R: the base case reads the empty prefix, the successor case appends one signed digit and normalizes the carried endpoint, and the handoff proceeds through* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$ *without introducing quotient streams, selected representatives, countable choice, or ambient completeness.*

Proof. Use the carrier rows of Definition 2075.1. The rows S, W, E provide the finite prefix schedule and endpoint normalization ledger, while T, R, L, H, C, P, N record threshold replay, regular readback, Real sealing, transport, replay, provenance, and local naming.

In the base case, the empty prefix is read from the scheduled window row W. Its endpoint row in E is the displayed DyadicRatCore$^{\uparrow}$ normalization for that empty window, so the threshold row T receives a finite starting datum.

In the successor case, extend the visible prefix by one signed digit. The signed-digit row S supplies the digit and carry ledger, and E normalizes the carried endpoint to the next displayed dyadic row. Componentwise hsame transport and Cont replay preserve the same prefix ledger rather than choosing a representative of an infinite stream.

After the prefix ledger has been normalized, Definition 2075.3 hands the row to T and R. The terminal seal L can then be read only through the resulting RegSeqRat$^{\uparrow}$ route and the Real$^{\uparrow}$ boundary. Thus every accepted prefix contribution is produced by the displayed finite induction. □

Theorem 2075.5 (SignedDigitCauchyModulus non-escape)**.** *An accepted* SignedDigitCauchyModulus$^{\uparrow}$ *packet gives no countable choice, quotient stream equality, ambient* Real$^{\uparrow}$ *completeness, or general compactness principle. Its consumers can read only the displayed signed-digit windows, normalized dyadic endpoints, monotone threshold row, regular Cauchy readback, real seal boundary, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier coordinates. The only source rows are finite signed-digit windows, scheduled observations, normalized endpoints, one monotone threshold ledger, one regular readback row, one real-seal row, and the structural transport and provenance rows.

Countable choice would require a family of independent selections outside the finite ledger; no such coordinate appears. Quotient stream equality would require an equality object for infinite signed-digit tails; the classifier surface reads only finite windows and normalized endpoint rows.

Ambient Real$^{\uparrow}$ completeness and general compactness would require a completed object or covering principle not present in L, H, C, P, N. The seal row L is merely the terminal consumer of the explicit regular readback and does not export a completion theorem.

Every consumer route therefore factors through the rows listed in Definition 2075.1 and through the readback theorem Definition 2075.3. The excluded principles have no source coordinate from which they could be projected or transported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDigitCauchyModulus}^{\uparrow}$ packet is generated from signed-digit finite windows, normalized $\mathsf{DyadicRatCore}^{\uparrow}$ endpoints, a monotone threshold modulus, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDigitCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2075.1, Definition 2075.2, Definition 2075.3, and Definition 2075.5 seed the finite signed-digit route from normalized windows to explicit regular Cauchy modulus data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDigitCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, countable choice, quotient equality of streams, ambient $\mathsf{Real}^{\uparrow}$ completeness, general compactness, selected representatives, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-digit normalization, monotone threshold replay, regular readback exactness, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.SignedDigitCauchyModulusUp.

Chapter 2076

A Concrete Naming Certificate for $\mathsf{RegularCauchyProductProjection}^{\uparrow}$

Following Bishop and Bridges on finite products of constructive Cauchy real names, the RegularCauchyProductProjec packet records the left and right projection readbacks out of a displayed regular Cauchy product packet. It sits after the finite product carrier of chapter 2009, the product-modulus ledger of **??**, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real algebra and completion consumers a named component-readback route without importing host product equality, quotient stream representatives, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2076.1 (Regular Cauchy product projection carrier)**.** *A* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, L, R, M, W, D, A, B, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *product packet whose component coordinates are being projected.*
2. *L and R are the left and right projection readback rows, each naming the component slot, the source regular Cauchy row, and the projected finite observation route.*
3. *M is the* $\mathsf{CauchyProductModulus}^{\uparrow}$ *ledger that assigns every projection request to the product-facing modulus budget before either component readback is consumed.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the product packet, the left component, the right component, and the product-modulus request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the product window and the two projected component windows.*
6. *A is the left* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from K, M, W, D, L.*
7. *B is the right* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from K, M, W, D, R.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the projected regular rational readbacks A and B have been displayed.*

9. *H stores componentwise* hsame *transport for the product packet, projection readbacks, product-modulus ledger, stream windows, dyadic tolerances, regular rational handoffs, seal, replay, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from a projection request through the product packet, product-modulus ledger, finite windows, dyadic rows, component readbacks, regular rational handoffs, and seal.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$, $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$, $\mathsf{CauchyProductModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, L, R, M, W, D, A, B, S, H, C, P, N$. It has no coordinate for host product equality, quotient stream representatives, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 2076.2 (Regular Cauchy product projection budget)**.** *Let*

$$J = (K, L, R, M, W, D, A, B, S, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *carrier. Every left or right projection request from J factors through the displayed product-modulus budget M, the finite window family W, the dyadic tolerance ledger D, the selected component readback L or R, and the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff A or B before the* $\mathsf{Real}^{\uparrow}$ *seal S is available.*

Proof. Start with the carrier of Definition 2076.1. A projection request first identifies the product packet K and one of the two projection readback rows L or R.

The request then passes through the product-modulus ledger M. This row is the only place where the product-facing precision request is assigned to component budgets, so neither component handoff can be read before M is displayed.

Next the finite window family W and the dyadic tolerance ledger D supply the actual component observations and tolerance rows. The left handoff A consumes the left readback row L, while the right handoff B consumes the right readback row R.

The seal S appears only after those regular rational handoffs. Therefore every projected read follows the route through M, W, D, the selected component readback, and its $\mathsf{RegSeqRat}^{\uparrow}$ handoff before any $\mathsf{Real}^{\uparrow}$ seal can be consumed. □

Theorem 2076.3 (Regular Cauchy product projection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ *carrier*

$$J = (K, L, R, M, W, D, A, B, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyProductProjection}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the product packet, left and right projection readbacks, product-modulus ledger, stream windows, dyadic tolerance rows, regular rational handoffs, Real seal, transport, replay, provenance, and local name;*

2. *projection-budget exactness, requiring every projected precision request to pass through M, W, D before A or B is available;*

3. *left/right readback ordering, requiring L to feed only the left handoff A and R to feed only the right handoff B;*

4. *Real-seal nonescape, requiring S to expose only the displayed projected readback route and no host product equality, quotient stream representative, selected limit, or ambient completeness claim;*

5. *transport, replay, provenance, and local naming stability through H, C, P, N for the same product packet, projection rows, modulus ledger, windows, dyadic tolerances, handoffs, and seal.*

No projection NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyProductProjectionUp.RegularCauchyProductProjectionTasteGate single carrier alignment`

Proof. Carrier admission is the tuple displayed in Definition 2076.1. It lists the product packet, the two projection readbacks, the product-modulus ledger, finite windows, dyadic tolerance rows, handoffs, seal, and structural rows.

Projection-budget exactness is Definition 2076.2. Its factorization route shows that M, W, D stand before the projected handoffs A and B.

The left/right ordering follows by projecting the two readback coordinates. The carrier gives L as the left readback consumed by A and R as the right readback consumed by B, with no third row that can swap or duplicate the roles.

The Real-seal boundary is the projection to S. Since S is downstream of the two regular rational handoffs and no excluded object is a coordinate of the carrier, the nonescape clause follows by inspection.

Finally, H, C, P, N transport, replay, source, and name the same displayed rows. Thus stability is componentwise over the packet already admitted, and the listed five rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ component-facing read factors through a RegularCauchyFiniteProduct packet, left or right projection readback, CauchyProductModulus ledger, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regular-cauchy-finite-product, cauchy-product-modulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a product packet, two ordered projection readbacks, product-modulus budget accounting, synchronized finite windows, dyadic tolerance rows, component handoffs, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductProjection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2076.1, Definition 2076.2, and Definition 2076.3 seed the finite projection route from displayed regular Cauchy products through product-modulus budgets, component readbacks, RegSeqRat handoffs, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductProjection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyProductProjectionUp`

Bridge status: none

Not claimed: This chapter does not close host product equality, quotient stream representatives, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection-budget exactness, left/right readback ordering, Real-seal nonescape, and transport/replay/provenance stability for BEDC.Derived.RegularCauchyProductProjectionUp.

Constructive story: A $\mathsf{RegularCauchyProductProjection}^{\uparrow}$ packet is generated from a RegularCauchyFiniteProduct packet, left and right projection readbacks, a CauchyProductModulus ledger, StreamName windows, DyadicRatCore

tolerance rows, RegSeqRat component handoffs, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2077

A Concrete Naming Certificate for $\mathsf{FastConvergentRealSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Weihrauch-style computable analysis, the $\mathsf{FastConvergentRealSequence}^{\uparrow}$ packet records a real sequence whose convergence is carried by an explicit displayed rate. It sits on the open Real and Completion route after the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the explicit rate surface of chapter 1965, and the terminal real-name seal of chapter 38. It is distinct from the sibling $\mathsf{FastConvergentSeries}^{\uparrow}$ certificate of chapter 367: this chapter names a rate-bearing sequence route, not a partial-sum and tail-sum route.

Definition 2077.1 (Fast convergent real sequence carrier)**.** *A* $\mathsf{FastConvergentRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the stream schedule. It records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which requested sequence positions and precision rows are observed.*
2. *R is the regular rational readback. It stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *observations attached to the scheduled windows and records the regularity checks available to downstream real-name consumers.*
3. *D is the dyadic tolerance ledger. It lists the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, comparison cells, denominator budgets, and refinement rows used by the regular readback.*
4. *M is the explicit convergence-rate row. It gives the finite modulus or rate schedule by which each tolerance request is assigned a displayed tail window.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after S, R, D, and M have been displayed and replayed.*
6. *H stores componentwise* hsame *transport for the stream schedule, regular rational readback, dyadic tolerance ledger, rate row, real seal, continuation replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a finite precision request through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FastConvergentRealSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealConvergenceRate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{FastConvergentRealSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the rows $S, R, D, M, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected limit, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.FastConvergentRealSequenceUp.FastConvergentRealSequenceTasteGate single carrier alignment`

Theorem 2077.2 (Fast real sequence rate exposure)**.** *For every accepted* $\mathsf{FastConvergentRealSequence}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

each precision request reaches the explicit convergence-rate row M before it may read a tail window, dyadic tolerance comparison, regular rational readback, or terminal $\mathsf{Real}^{\uparrow}$ *seal. Thus the rate-bearing sequence surface exposes a displayed modulus route rather than a host convergence predicate.*

Proof. Project the carrier in Definition 2077.1. The row M is a named coordinate of the accepted packet, while S, R, and D are the scheduled observation, regular readback, and tolerance rows to which the rate is applied.

Replay the finite request route stored in C. A precision request first selects the displayed rate threshold in M, then opens the corresponding finite StreamName window in S and the dyadic tolerance cells in D.

Only after those rows have been displayed can the regular rational row R be read as a tail-controlled observation surface. Since the rate is a coordinate of the packet and the route uses only S, R, D, M, H, C, P, N, no host convergence predicate is introduced. □

Theorem 2077.3 (Fast real sequence Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FastConvergentRealSequence}^{\uparrow}$ *packet factors through the displayed route*

$$S, M, D, R, H, C, P, N \longmapsto E.$$

The terminal seal E is therefore downstream of the stream schedule, explicit rate, dyadic tolerance ledger, regular rational readback, transport, replay, provenance, and local NameCert rows.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. The seal row E appears in the carrier only as the terminal handoff coordinate, not as an independent real endpoint.

Use Definition 2077.2. Every precision request reads M before it opens the finite tail window and tolerance ledger. The rows S and D therefore determine the observations and comparisons that the regular rational row R may expose.

The handoff to E is then a projection of the displayed carrier route. Componentwise transport in H, replay in C, provenance in P, and local naming in N bind the same rows to the terminal Real seal. No separate limit selector or ambient completeness coordinate is available. □

Theorem 2077.4 (Fast real sequence non-escape)**.** *An accepted* $\mathsf{FastConvergentRealSequence}^{\uparrow}$ *packet exports only its displayed stream schedule, regular rational readback, dyadic tolerance ledger, convergence-rate row, Real seal handoff, componentwise* hsame *transports, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. It exports no quotient stream equality, selected limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Read the carrier definition. Its complete coordinate list is $S, R, D, M, E, H, C, P, N$, and every classifier comparison is restricted to those rows.

The rate exposure theorem identifies the only convergence control available to a consumer: the displayed row M together with the scheduled windows and tolerance ledger. The Real seal handoff theorem identifies the only terminal real-facing route: the composite through S, M, D, R, H, C, P, N into E.

The excluded objects are not coordinates of the packet and are not produced by the finite replay route. Therefore no consumer of the accepted packet can extract quotient stream equality, a selected limit, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking from this NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastConvergentRealSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{FastConvergentRealSequence}^{\uparrow}$ packet is generated from a finite StreamName schedule, RegSeqRat readback rows, a DyadicRatCore tolerance ledger, an explicit RealConvergenceRate row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastConvergentRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2077.1, Definition 2077.2, Definition 2077.3, and Definition 2077.4 seed the rate-bearing real-sequence route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastConvergentRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, rate-to-tail routing, dyadic tolerance accounting, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.FastConvergentRealSequenceUp.

Chapter 2078

A Concrete Naming Certificate for $\mathsf{PeanoCurve}^{\uparrow}$

Following Peano's space-filling construction, Hilbert-style dyadic subdivision curves, and Bishop–Bridges constructive real analysis, the $\mathsf{PeanoCurve}^{\uparrow}$ packet records a continuous map from the unit interval onto the unit square as finite dyadic observation data. It reads the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the finite stream-window interface of chapter 126, the dyadic endpoint ledger of chapter 291, the graph carrier of chapter 130, and the compact witness surface of chapter 129. The packet names the finite grid readback by which a dyadic input address determines compatible dyadic square addresses; it is not a host-level function space, an uncountable choice principle, a quotient real equality, or an open-cover compactness theorem.

Definition 2078.1 (PeanoCurve finite dyadic subdivision carrier)**.** *A* $\mathsf{PeanoCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, G, S, X, Y, E, R, H, C, Q, N)$$

with the following displayed rows:

1. *I is the interval-address row. It records a finite dyadic interval address, its precision level, and the corresponding* $\mathsf{StreamName}^{\uparrow}$ *observation window before any planar readback is exposed.*
2. *G is the grid-subdivision row. For each displayed interval address in I, it records the finite dyadic square cell selected by the Peano subdivision rule, including the horizontal and vertical dyadic endpoints.*
3. *S is the sibling-continuity row. Adjacent dyadic interval cells are required to map to edge-adjacent or coincident dyadic square cells at the same precision, so the graph can be read through the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface.*
4. *X and Y are the two coordinate readback rows. They turn the horizontal and vertical dyadic endpoints of G into* $\mathsf{RegSeqRat}^{\uparrow}$ *and then* $\mathsf{Real}^{\uparrow}$ *coordinate seals.*
5. *E is the error-budget row. A request for square precision 2^{-n} is answered by an interval precision and a dyadic diameter bound supplied through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
6. *R is the finite surjection readback row. Every displayed dyadic square cell at precision n is paired with at least one interval address whose grid-subdivision row lands inside that cell.*

7. *H records componentwise* hsame *transport for interval addresses, grid cells, sibling-continuity witnesses, coordinate readbacks, error budgets, surjection readbacks, replay data, provenance, and local naming.*

8. *C records displayed* Cont *replay through*

$$I, G, S \longmapsto X, Y, E, R \longmapsto H, Q, N.$$

9. *Q is the* Pkg *provenance over* $\mathsf{PeanoCurve}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PeanoCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, S, X, Y, E, R, H, C, Q, N$. It has no coordinate for a completed function space, arbitrary pointwise choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.PeanoCurveUp.PeanoCurveTasteGate single carrier alignment`

Theorem 2078.2 (PeanoCurve NameCert obligations)**.** *For an accepted* $\mathsf{PeanoCurve}^{\uparrow}$ *carrier*

$$P = (I, G, S, X, Y, E, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeanoCurve}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the interval-address, grid-subdivision, sibling-continuity, coordinate readback, error-budget, finite surjection, transport, replay, provenance, and local naming rows;*

2. *dyadic subdivision exactness, requiring every planar cell read to open I and then G before a coordinate seal is exposed;*

3. *continuity budget exactness, requiring adjacent interval addresses and square-cell diameters to be checked by S and E before the carrier is consumed through* $\mathsf{ContinuousMap}^{\uparrow}$*;*

4. *finite-grid surjection exactness, requiring every displayed dyadic square cell to be met by the readback row R at the requested precision;*

5. *structural exhaustion, requiring H, C, Q, N to transport, replay, source, and name only the listed finite dyadic route.*

No NameCert obligation is stored outside those five rows, and none exports a host-level function space, arbitrary pointwise choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a bridge credential.*

Proof. Carrier admission is projection from Definition 2078.1. The accepted packet displays the interval-address row, grid-subdivision row, sibling-continuity row, coordinate readback rows, error-budget row, finite surjection row, and structural rows.

Dyadic subdivision exactness is read from I and G. A consumer cannot inspect planar coordinates until a finite interval address and its selected dyadic square cell have both been displayed.

Continuity budget exactness is read from S and E. The sibling row checks that adjacent interval cells have compatible square cells, while the error-budget row supplies the dyadic diameter bound requested by the continuous-map consumer.

Finite-grid surjection exactness is read from R. For each displayed square cell at the requested precision, the row supplies an interval address whose grid-subdivision cell lands in that square cell.

Finally H, C, Q, N add no analytic coordinate. They preserve, replay, source, and name the same finite route, so the excluded function-space, choice, quotient, compactness, completeness, verification, and bridge data have no row from which a consumer could read them. □

Theorem 2078.3 (PeanoCurve finite-grid continuous surjection surface)**.** *Let*

$$P = (I, G, S, X, Y, E, R, H, C, Q, N)$$

be an accepted $\mathsf{PeanoCurve}^{\uparrow}$ *carrier. Every compact, continuous, or real-name consumer of* P *receives the Peano curve only through the finite route*

$$I, G, S \longmapsto X, Y, E, R \longmapsto H, C, Q, N.$$

At each requested precision n*, the route exposes dyadic interval addresses, dyadic square cells, coordinate readbacks through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, a dyadic continuity budget, and a finite witness that each displayed square cell has an interval preimage. Thus the packet presents the continuous-surjection surface over finite grids without constructing a host-level uncountable function space or selecting arbitrary real representatives.*

Proof. Fix a consumer request at square precision n. The interval-address row I supplies the finite dyadic input windows that may be inspected, and the grid-subdivision row G assigns each such window to a displayed dyadic square cell.

The coordinate readbacks are then forced through X and Y. Each coordinate endpoint is read as regular rational data and then as a terminal real-name seal, so the planar point is exposed only through the sibling $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ routes.

Continuity is a finite budget check. The row S compares adjacent interval cells, and E records a dyadic diameter bound sufficient for the requested square precision. Therefore the continuous-map consumer reads a modulus-style finite surface rather than a host function graph.

Surjection at the displayed precision is read from R. Every dyadic square cell listed at precision n has a paired interval address, so the finite grid is covered by the readback route.

The structural rows close the handoff. Transport, replay, provenance, and local naming preserve the same finite dyadic route, which gives compact, continuous, and real-name consumers no bypass through function spaces, arbitrary representatives, quotient equality, open-cover compactness, or ambient completeness. □

Theorem 2078.4 (PeanoCurve grid-cell compatibility)**.** *Let*

$$P = (I, G, S, X, Y, E, R, H, C, Q, N)$$

be an accepted $\mathsf{PeanoCurve}^{\uparrow}$ *carrier. Every displayed dyadic square cell read from* G *has coordinate readbacks* X *and* Y *whose* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals are controlled by the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget* E *and the same* $\mathsf{StreamName}^{\uparrow}$ *interval window* I*. Adjacent interval cells use the sibling-continuity row* S *to preserve coordinate compatibility before any compact, continuous, or real-name consumer reads the structural rows* H, C, Q, N*.*

Proof. Project the accepted carrier. The interval-address row I, grid-subdivision row G, sibling-continuity row S, coordinate readback rows X, Y, and dyadic budget row E are all explicit coordinates of the packet.

Fix a displayed square cell in G. Its horizontal and vertical endpoints can be read only through X and Y, and those rows route their regular-rational and real-name seals through the dyadic budget E and the interval window I already attached to that cell.

For adjacent interval cells, the row S records the compatibility check before either coordinate readback is exposed to a downstream consumer. Thus neighboring grid cells preserve the finite edge or coincidence relation required by the Peano subdivision surface.

Finally apply Definition 2078.3. It routes every compact, continuous, and real-name consumer through

$$I, G, S \longmapsto X, Y, E, R \longmapsto H, C, Q, N,$$

so the compatibility witnessed by I, G, S, X, Y, E is preserved by the structural transport, replay, provenance, and local naming rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeanoCurve}^{\uparrow}}$.

Constructive story: A $\mathsf{PeanoCurve}^{\uparrow}$ packet is generated from finite dyadic interval addresses, grid-subdivision cells, sibling-continuity rows, coordinate readbacks through RegSeqRat and Real seals, dyadic error budgets, finite square-cell readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PeanoCurve}^{\uparrow}}$, seed)

Scope closed: Definition 2078.1, Definition 2078.2, Definition 2078.3, and Definition 2078.4 seed the finite dyadic subdivision route for a Peano continuous-surjection packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PeanoCurve}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host-level function spaces, arbitrary pointwise choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic subdivision exactness, continuity budget exactness, finite-grid surjection exactness, and structural replay for BEDC.Derived.PeanoCurveUp.

Chapter 2079

A Concrete Naming Certificate for $\mathsf{SturmSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, a Sturm chain is read as a finite Euclidean division ledger whose endpoint sign rows can be inspected without choosing roots in an ambient ordered field. The $\mathsf{SturmSequence}^{\uparrow}$ packet names that finite sign-variation surface: its polynomial source is the coefficient and pseudo-remainder route of chapter 50, its bracket consumer is chapter 1786, and its Real-facing windows pass through the completion route of chapter 291, chapter 301, and chapter 38. The certificate exposes a chain and its variation counts, not a host real-root theorem.

Definition 2079.1 (SturmSequence Real carrier)**.** *A* $\mathsf{SturmSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, D, Q, E, V, B, W, R, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, carrying the coefficient ledger, the degree bound, the formal derivative row, and the polynomial identity under inspection.*
2. *D is the Euclidean-division ledger for the Sturm chain. It records each displayed quotient, remainder, sign normalization, and terminal-zero decision used to construct the next chain entry.*
3. *Q is the finite chain table, indexed by the entries produced from P and D, with no appeal to factorization over a host field.*
4. *E is the endpoint-evaluation row. It stores the rational or dyadic endpoint windows and the displayed sign rows for every chain entry at each inspected endpoint.*
5. *V is the sign-variation row. It records the deletion of zero signs by displayed neighbor rows and the resulting variation counts at the two endpoints.*
6. *B is the bracket interface row consumed by* $\mathsf{RealRootBracket}^{\uparrow}$*: it stores only the endpoint cells, endpoint signs, and variation-count difference exposed by V.*
7. *W is the dyadic window handoff row through* $\mathsf{DyadicRatCore}^{\uparrow}$*, retaining the requested endpoint precision and comparison rows.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to expose the same endpoint windows at unary precision before any* $\mathsf{Real}^{\uparrow}$ *seal is read.*

9. *H records componentwise* hsame *transport for the polynomial, division, chain, endpoint, variation, bracket, dyadic, regular-rational, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a requested sign-variation read through polynomial inspection, chain construction, endpoint evaluation, zero deletion, variation counting, bracket handoff, and completion-route readback.*

11. *K is the* Pkg *provenance over* $\mathsf{Poly}^{\uparrow}$, $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmSequence}^{\uparrow}}$ *registration row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for a selected root, algebraic closure, real trichotomy, quotient real equality, factorization theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.SturmSequenceUp.TasteGate.SturmSequenceTasteGate single carrier alignment`

Theorem 2079.2 (SturmSequence sign-variation window)**.** *For every accepted* $\mathsf{SturmSequence}^{\uparrow}$ *carrier $S = (P, D, Q, E, V, B, W, R, H, C, K, N)$ and every inspected endpoint pair, the visible sign-variation window factors through the displayed route*

$$P, D \longmapsto Q \longmapsto E \longmapsto V \longmapsto B, W, R, H, C, K, N.$$

The window exposes the finite chain entries, endpoint signs, zero-deletion witnesses, and endpoint variation counts. It exports no selected root, no host factorization theorem, and no $\mathsf{Real}^{\uparrow}$ *equality principle.*

Proof. Open the carrier of Definition 2079.1. The polynomial row P and division ledger D are the only inputs from which the chain table Q is admitted, so the chain cannot be replaced by an external factorization or by a root enumeration.

Endpoint inspection is then recorded in E. Each sign row is attached to a displayed chain entry and a displayed rational or dyadic endpoint window, and the zero rows used by the variation count remain visible.

The variation row V reads those endpoint signs and records the finite zero-deletion and sign-change count. The bracket interface B receives only the endpoint cells and the displayed difference of the two counts.

Finally, W and R expose the same endpoint windows through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, while H, C, K, N provide transport, replay, provenance, and naming. Since no listed coordinate stores a selected root, a factorization theorem, or a quotient real equality proof, the exported window remains finite Sturm-sequence data. □

Theorem 2079.3 (SturmSequence NameCert obligation surface)**.** *For every accepted* $\mathsf{SturmSequence}^{\uparrow}$ *carrier, the local* $\mathsf{NameCert}_{\mathsf{SturmSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the rows listed in Definition 2079.1;*

2. *polynomial-source admission through the coefficient, derivative, degree, and polynomial-identity rows of P;*

3. *chain-construction exactness through the displayed quotient, remainder, sign-normalization, and terminal-zero rows of D and Q;*

4. *endpoint-sign exactness through the endpoint windows and displayed sign rows of E;*

5. *variation-count exactness through the zero-deletion witnesses and sign-change counts of V;*
6. *bracket and completion-route discipline through B, W, R before any* $\mathsf{Real}^{\uparrow}$*-facing consumer is available;*
7. *componentwise transport, continuation replay, provenance, and local naming through H, C, K, N.*

There is no further NameCert obligation for algebraic closure, selected roots, quotient real equality, real trichotomy, factorization, or ambient completeness.

Proof. The carrier row gives the admission surface. Its coordinates name exactly the polynomial source, division ledger, chain table, endpoint signs, variation counts, bracket interface, dyadic and regular-rational readbacks, and structural BEDC rows.

The chain-construction obligations are finite because D and Q list the quotient, remainder, sign-normalization, and terminal-zero rows used to produce the table. No obligation asks for a host factorization or for a completed list of roots.

The endpoint and variation obligations are also finite. The row E attaches each sign to an endpoint window, and the row V records the zero-deletion witnesses and variation counts consumed by the bracket interface.

The completion-route obligations are downstream of that interface. Rows W and R expose the retained endpoint windows through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, while H, C, K, N replay and name the same route. Thus the local certificate has no remaining slot from which to export algebraic closure, selected roots, quotient real equality, real trichotomy, factorization, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{SturmSequence}^{\uparrow}$ read factors through one $\mathsf{Poly}^{\uparrow}$ source row, one Euclidean-division ledger, one finite chain table, endpoint sign rows, a sign-variation row, a $\mathsf{RealRootBracket}^{\uparrow}$ interface row, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, real-root-bracket, dyadicratcore, regseqrat, real: $\mathsf{SturmSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Euclidean-division ledger, finite chain table, endpoint sign rows, zero-deletion witnesses, variation counts, bracket handoff, and completion-route endpoint readback.

Closure status for $\mathsf{NameCert}_{\mathsf{SturmSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2079.1, Definition 2079.2, and Definition 2079.3 seed the finite Sturm-chain route from polynomial division rows through endpoint sign variation, bracket handoff, and dyadic, regular-rational, and Real-facing windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close algebraic root existence, factorization over a host field, selected roots, quotient real equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, chain-construction exactness, endpoint-sign exactness, variation-count exactness, bracket handoff, completion-route discipline, and transport-replay coverage for BEDC.Derived.SturmSequenceUp.

Constructive story: A $\mathsf{SturmSequence}^{\uparrow}$ packet is generated from a $\mathsf{Poly}^{\uparrow}$ source row, a Euclidean-division ledger, a finite chain table, endpoint sign rows, zero-deletion witnesses, sign-variation counts, a $\mathsf{RealRootBracket}^{\uparrow}$ interface row, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2080

A Concrete Naming Certificate for $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$

Bishop–Bridges constructive analysis treats Pringsheim's positive-real-series criterion as a finite route from monotone partial sums and explicit tail budgets to a real endpoint, not as a host appeal to suprema or ambient completeness. The $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ packet sits after the finite windows of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, the real-series Cauchy route of chapter 2014, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the comparison-facing sibling chapter 2023. It records the classical positive-term monotone partial-sum criterion as displayed BEDC certificate data; it does not import host least upper bounds, selected infinite sums, quotient equality of series, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2080.1 (Pringsheim positive real series carrier)**.** *A* $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the positive term-window row. It records exactly the inspected nonnegative real terms and their finite positivity witnesses.*
2. *S is the monotone partial-sum row. It records the displayed partial sums, the successor-step additions, and the monotonicity checks visible to the certificate.*
3. *M is the monotone-bound ledger. It stores the finite upper rows and comparison requests used to turn monotone partial sums into a Cauchy tail budget.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget row. It stores dyadic tolerances, slack radii, denominator bounds, and the finite threshold after which every displayed tail is below the requested budget.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the same partial-sum windows and dyadic budgets as a regular rational sequence before any endpoint seal is exported.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after T, S, M, D, R have displayed the positive terms, monotone partial sums, tail budgets, and regular rational handoff.*
7. *H stores componentwise* hsame *transport for the term windows, partial-sum rows, monotone-bound ledger, dyadic budgets, regular handoff, real seal, replay, provenance, and local naming row.*

8. *C stores displayed* Cont *replay from term windows to partial sums, from partial sums to monotone-bound rows, from those rows to dyadic tail budgets, and from the budgets to the regular handoff and real seal.*

9. *P is the* Pkg *provenance over* PringsheimPositiveRealSeries$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealSeriesCauchyCriterion$^{\uparrow}$, Real$^{\uparrow}$, SeriesLimitComparison$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{PringsheimPositiveRealSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, M, D, R, E, H, C, P, N$. It has no coordinate for a host supremum of partial sums, a selected infinite sum, quotient equality of series, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.PringsheimPositiveRealSeriesUp.PringsheimPositiveRealSeriesTasteGate`
`single carrier alignment`

Theorem 2080.2 (Pringsheim positive real series NameCert obligations)**.** *For every accepted* PringsheimPositiveRealSeries$^{\uparrow}$ *carrier $K = (T, S, M, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{PringsheimPositiveRealSeries}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed positive term windows, monotone partial-sum rows, monotone-bound ledger, dyadic tail budget, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*

2. *positive-window exactness, requiring every positivity read and every term read to pass through T and the finite* StreamName$^{\uparrow}$ *windows named in P;*

3. *monotone partial-sum exactness, requiring every inspected partial sum and successor addition to pass through S before a tail request is admitted;*

4. *tail-budget exactness, requiring M and D to display the finite threshold, upper row, dyadic slack, and tail tolerance consumed by the* RealSeriesCauchyCriterion$^{\uparrow}$ *route;*

5. RegSeqRat$^{\uparrow}$ *handoff, requiring R to read the same partial-sum windows and dyadic budgets before any* Real$^{\uparrow}$ *endpoint seal is available;*

6. *real-seal nonescape, requiring E to remain downstream of T, S, M, D, R and to export no host supremum, selected infinite sum, quotient equality of series, countable choice, trichotomy, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge datum;*

7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Pringsheim positive-series obligation is stored outside these rows.

Proof. Project the accepted packet $K = (T, S, M, D, R, E, H, C, P, N)$. The carrier definition exposes exactly the positive term windows, monotone partial-sum rows, monotone-bound ledger, dyadic budget, regular handoff, real seal, and structural rows.

The positivity and partial-sum data are read first. The row T supplies only displayed term windows, and the row S supplies only displayed finite partial sums and successor additions.

The tail route then passes through M and D. These rows record the upper row, threshold, dyadic slack, and requested tolerance before the RealSeriesCauchyCriterion$^{\uparrow}$ route may be read.

Finally R, E, H, C, P, N hand off, seal, transport, replay, source, and name the same displayed route. Since no excluded analytic, choice, quotient, verification, or bridge object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2080.3 (Pringsheim positive real series tail bound)**.** *Let $K = (T, S, M, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ *carrier. Every completion-facing tail request for the positive real series factors through*

$$T \longmapsto S \longmapsto M, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the finite monotone partial-sum rows and $\mathsf{DyadicRatCore}^{\uparrow}$ *tail budgets determine the Cauchy handoff, not a host least upper bound, selected infinite sum, trichotomy, countable choice, quotient series equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing tail request. The carrier offers no direct infinite-sum coordinate, so the request first reads the finite term windows T and the partial-sum row S.

The monotone-bound ledger M identifies the displayed upper row and the finite threshold attached to the request. The dyadic budget D then supplies the tolerance, slack, and denominator data used by the tail estimate.

The regular handoff R reads exactly those displayed windows and budgets. This is the route made available to the $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ surface before a real endpoint can be sealed.

The seal and structural rows E, H, C, P, N preserve that route. Because the packet has no coordinate for a host supremum, selected sum, quotient equality, trichotomy, choice, or ambient completeness, every tail request is exhausted by the displayed factorization. □

Theorem 2080.4 (Pringsheim positive real series RegSeqRat handoff)**.** *For an accepted* PringsheimPositiveRealSeri *carrier $K = (T, S, M, D, R, E, H, C, P, N)$, every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by R is determined by the same positive term windows, monotone partial-sum rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets displayed in T, S, M, D. No consumer may replace this handoff by a selected limit of partial sums, quotient equality of streams, a trichotomy comparison, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the row R from the accepted packet. By Definition 2080.1, R appears only after the term windows, partial sums, monotone-bound ledger, and dyadic tail budget have been displayed.

Apply Definition 2080.2. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff obligation requires R to read the same partial-sum windows and dyadic budgets before any endpoint seal is available.

Apply Definition 2080.3. Every completion-facing tail request reaches R through T, S, M, D and then exits through E, H, C, P, N.

Therefore R is a local regular-rational handoff row for the displayed finite packet. It cannot be detached as a selected limit, quotient stream equality, trichotomy comparison, choice object, or ambient completeness principle. □

Theorem 2080.5 (Pringsheim positive real series real seal nonescape)**.** *For every accepted* PringsheimPositiveRealSe *carrier, any* $\mathsf{Real}^{\uparrow}$ *consumer of the sealed endpoint must read the positive term windows, monotone partial-sum rows, monotone-bound ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row of Definition 2080.1. The endpoint seal exports no host supremum, selected infinite sum, quotient equality of series, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Proof. Let an accepted endpoint consumer read the row E. The carrier definition places E downstream of T, S, M, D, R, so the consumer first receives the displayed positive terms, monotone partial sums, tail budgets, and regular handoff.

The NameCert obligations force the same order. Positive-window exactness and monotone partial-sum exactness bind the visible series data, tail-budget exactness binds M, D, and the handoff obligation binds R before E can be used.

The structural rows H, C, P, N then transport, replay, source, and name the route. No other coordinate remains in the carrier, so the endpoint seal has no path to a host supremum, selected sum, quotient equality, trichotomy, choice, ambient completeness, Lean verification, or bridge datum. □

Falsifiable prediction: Every $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ completion-facing read factors through positive term windows, monotone partial-sum rows, a monotone-bound ledger, DyadicRatCore tail budgets, RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real-series-cauchy-criterion, real, series-limit-comparison: $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries positivity windows, monotone partial-sum rows, monotone-bound tail budgets, regular rational handoff, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PringsheimPositiveRealSeries}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PringsheimPositiveRealSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2080.1, Definition 2080.2, Definition 2080.3, Definition 2080.4, and Definition 2080.5 seed the finite constructive Pringsheim positive-real-series route from monotone partial sums through dyadic tail budgets to RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PringsheimPositiveRealSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host least upper bounds, selected infinite sums, quotient equality of series, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-window exactness, monotone partial-sum exactness, DyadicRatCore tail-budget handoff, RegSeqRat readback, Real seal nonescape, and structural transport for BEDC.Derived.PringsheimPositiveRealSeriesUp.

Constructive story: A $\mathsf{PringsheimPositiveRealSeries}^{\uparrow}$ packet is generated from positive term windows, monotone partial-sum rows, a monotone-bound ledger, a DyadicRatCore tail-budget row, a RegSeqRat handoff, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2081

A Concrete Naming Certificate for $\mathsf{BishopIntervalHalving}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopIntervalHalving}^{\uparrow}$ packet records the interval-halving route by which a located real interval supplies finite bisection windows, dyadic diameter budgets, regular rational readback, and a terminal real seal. It consumes the located interval surface of chapter 1709, the midpoint and bisection rows of chapter 1712 and chapter 1711, the finite stream discipline of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a constructive bisection certificate for L10 Real and Completion routes; it does not assert open-cover compactness, host interval completeness, quotient real equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2081.1 (Bishop interval halving carrier)**.** *A* $\mathsf{BishopIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, M, B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, carrying the lower and upper endpoint names, locator reads, radius data, and locatedness budget for the interval to be halved.*
2. *M is the* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ *row, exposing the finite midpoint cell used by each halving request.*
3. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *row, recording the retained child interval and the displayed branch decision for the current finite request.*
4. *D is the dyadic diameter budget row. It records the requested precision, the halving depth, the dyadic denominator budget, and the comparison slack showing that the retained interval diameter is below the requested tolerance.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It lists the visible halving windows inspected by the downstream readback, and it contains no completed infinite branch.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained interval windows and their dyadic diameter witnesses.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after I, M, B, D, W, R have been displayed.*

8. *H records componentwise* hsame *transport for the located interval, midpoint, bisection, dyadic budget, finite windows, regular rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$I, M, B, D \longmapsto W \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalHalving}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalHalving}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, B, D, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, an infinite branch selector, quotient stream equality, quotient real equality, host real equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalHalvingUp.TasteGate.BishopIntervalHalvingTasteGate single carrier alignment`

Theorem 2081.2 (Bishop interval halving NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalHalving}^{\uparrow}$ *carrier*

$$K = (I, M, B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalHalving}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval, midpoint row, bisection row, dyadic diameter budget, finite stream windows, regular rational readback, real seal, transport, replay, provenance, and local name;*

2. *source-interval exactness, requiring every endpoint, radius, locator, locatedness, and metric-bracket request to pass through the displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row I;*

3. *midpoint and bisection exactness, requiring every retained child interval to read its midpoint from M and its branch data from B before any downstream window is exposed;*

4. *diameter-budget exactness, requiring D to bind each halving depth to a displayed dyadic tolerance and to the retained interval diameter used for the current request;*

5. *regular readback ordering, requiring W and D to feed R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is consumed;*

6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed route and to export no open-cover compactness, infinite branch selector, quotient stream equality, quotient real equality, host real equality, countable choice, selected limit, or ambient completeness row.*

No Bishop interval halving obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2081.1. The accepted packet displays exactly the located source interval, midpoint row, bisection row, dyadic budget, finite windows, regular readback, seal, and structural rows.

Source-interval exactness is forced by I. Endpoint, radius, locator, locatedness, and metric-bracket reads have no route except the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ coordinate.

Midpoint and bisection exactness are forced by M and B. A retained child interval is available only after the midpoint cell has been displayed and the bisection row has recorded the branch data for the finite request.

The dyadic diameter obligation is internal to D. The row binds each request to a halving depth, denominator budget, comparison slack, and retained-interval diameter bound before any regular rational readback is visible.

The rows W and R then supply the finite-window and $\mathsf{RegSeqRat}^{\uparrow}$ handoff. The terminal seal E is downstream of this readback, so a Real-facing consumer cannot bypass the displayed halving route.

Finally, H, C, P, N only transport, replay, source, and name the same packet. Since the carrier contains no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2081.3 (Bishop interval halving Real seal route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing consumer of an accepted* $\mathsf{BishopIntervalHalving}^{\uparrow}$ *carrier factors through the finite route*

$$I, M, B, D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives only the displayed located interval, midpoint and bisection rows, dyadic diameter budget, finite stream windows, regular rational readback, terminal real seal, componentwise transport, continuation replay, package provenance, and local naming data. It receives no open-cover compactness theorem, completed branch, selected real limit, quotient equality of real names, host real equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (I, M, B, D, W, R, E, H, C, P, N)$. The only source interval coordinate is I, so every interval request begins with the displayed located endpoint and locator data.

The midpoint and bisection rows form the finite halving step. The midpoint row M supplies the child-cell boundary, and the bisection row B records the retained interval for the current request.

The dyadic diameter budget D supplies the quantitative contraction. It records the halving depth and the diameter comparison needed before the finite StreamName windows W can be consumed by the regular rational readback.

The row R is downstream of W and D, so every $\mathsf{RegSeqRat}^{\uparrow}$ read is built from displayed halving windows and dyadic contraction data. The terminal seal E is then read only after R has been exposed.

The structural rows H, C, P, N preserve this route under componentwise transport, replay, provenance, and naming. Since no carrier coordinate stores compactness, a completed branch, a selected real limit, quotient equality, host equality, countable choice, or ambient completeness, no consumer can obtain such data from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalHalving}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalHalving}^{\uparrow}$ packet is generated from a LocatedRealIntervalUp row, a LocatedIntervalMidpointUp row, a LocatedIntervalBisectionUp row, a dyadic diameter budget, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopIntervalHalving}^{\uparrow}}$, seed)

Scope closed: Definition 2081.1, Definition 2081.2, and Definition 2081.3 seed the constructive interval-halving route from located intervals through midpoint and bisection rows, dyadic contraction budgets, finite windows, regular rational readback, terminal real seals, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopIntervalHalving}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, completed branches, selected real limits, quotient equality of real names, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, midpoint and bisection exactness, diameter-budget exactness, regular readback ordering, and real-seal nonescape for BEDC.Derived.BishopIntervalHalvingUp.

Chapter 2082

A Concrete Naming Certificate for $\mathsf{LengthSpace}^{\uparrow}$

Following Burago–Burago–Ivanov and Bridson–Haefliger on length spaces, the $\mathsf{LengthSpace}^{\uparrow}$ packet records the finite path-length ledger by which a metric distance is witnessed through displayed interval paths and displayed segment sums. It sits over the metric source of chapter 57, the continuous-map path surface of chapter 130, the interval address space of chapter 56, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite ledger for length reads; it is not a geodesic existence theorem, a host path space, a quotient path-space construction, an open-cover compactness result, a $\mathsf{Real}^{\uparrow}$ completeness theorem, or a selected-minimizer principle.

Definition 2082.1 (LengthSpace finite path-length carrier)**.** *A* $\mathsf{LengthSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, I, A, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, recording the accepted* $\mathsf{Met}^{\uparrow}$ *carrier, the endpoint identifiers, and the finite source-distance request consumed by the length ledger.*
2. *I is the interval path row, recording a finite* $\mathsf{Interval}^{\uparrow}$ *address family and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *path read used to connect the two endpoint identifiers.*
3. *A is the partition ledger, recording a finite ordered partition of the displayed interval addresses, including adjacent address pairs and endpoint coverage.*
4. *D is the segment-distance ledger, recording the* $\mathsf{Met}^{\uparrow}$ *distance read for each adjacent pair in A after the path row I has supplied its two point reads.*
5. *S is the segment-sum row, recording the finite rational sum of the entries in D and the tolerance split used to compare it with the source-distance request in M.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the finite segment sums as regular rational observations before any terminal real seal is consumed.*
7. *T is the* $\mathsf{Real}^{\uparrow}$ *length-readback row, sealing the regular rational observations from R as a displayed real-name length value for this finite path ledger.*

8. *C records componentwise* hsame *transport for the metric source, interval path row, partition ledger, segment-distance ledger, regular rational readback, real length readback, replay, provenance, and local naming rows.*
9. *P is the displayed* Cont *replay and* Pkg *provenance over* $\mathsf{LengthSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Interval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LengthSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, I, A, D, S, R, T, C, P, N$. It has no coordinate for geodesic existence, host paths, quotient path spaces, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or selected minimizers.*

Lean4 ✓ `BEDC.Derived.LengthSpaceUp.TasteGate.LengthSpaceTasteGate single carrier alignment`

Theorem 2082.2 (LengthSpace path ledger stability)**.** *Let*

$$L = (M, I, A, D, S, R, T, C, P, N)$$

be an accepted $\mathsf{LengthSpace}^{\uparrow}$ *carrier. Every endpoint-distance or length-read consumer of L factors through the displayed finite route*

$$M, I, A, D, S \longmapsto R, T \longmapsto C, P, N.$$

Consequently componentwise hsame *transport of $M, I, A, D, S, R, T, C, P, N$ preserves the same metric source, interval path row, partition ledger, segment-distance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *length seal, with no additional route through host paths, quotient path spaces, open-cover compactness, real completeness, geodesic selection, or minimizing-path choice.*

Proof. Carrier projection gives the ordered rows $M, I, A, D, S, R, T, C, P, N$. A consumer begins with the metric source row M, so the endpoint-distance request is tied to a displayed $\mathsf{Met}^{\uparrow}$ source before any path-length read is available.

The path data are finite. The interval path row I supplies the displayed $\mathsf{ContinuousMap}^{\uparrow}$ reads over $\mathsf{Interval}^{\uparrow}$, and the partition ledger A lists only finitely many adjacent interval addresses. Thus every segment request is bounded by the displayed partition.

The segment-distance ledger D reads each adjacent pair through the metric source, and S folds those entries into one finite segment sum. The readback rows R and T expose that sum first as $\mathsf{RegSeqRat}^{\uparrow}$ data and then as a terminal $\mathsf{Real}^{\uparrow}$ length seal.

Transport and replay are structural. Componentwise hsame in C preserves each listed row, while the displayed provenance in P and the local name in N replay the same finite route.

The excluded objects are absent from the carrier inventory. Since no row stores host paths, quotient path spaces, open covers, ambient completeness, geodesic witnesses, or minimizing-path choices, no consumer can obtain them from the transported packet. □

Theorem 2082.3 (LengthSpace NameCert obligation surface)**.** *For an accepted* $\mathsf{LengthSpace}^{\uparrow}$ *carrier*

$$L = (M, I, A, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LengthSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, interval path row, partition ledger, segment-distance ledger, segment-sum row, regular rational readback, real length readback, transport, replay, provenance, and local naming rows;*

2. *path-source exactness, requiring every displayed path read to pass through M and then I before a segment endpoint can be inspected;*
3. *partition exactness, requiring A to list the finite ordered interval addresses and adjacent pairs consumed by the segment-distance ledger;*
4. *segment-distance exactness, requiring D to read each adjacent pair through the same* $\mathsf{Met}^{\uparrow}$ *source and no other distance object;*
5. *length-readback exactness, requiring S, R, T to expose the finite segment sum through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *before any consumer treats it as a length value;*
6. *structural exhaustion, requiring C, P, N to transport, replay, source, and name only the displayed route M, I, A, D, S, R, T.*

No NameCert obligation is stored outside those rows, and none asserts geodesic existence, host paths, quotient path spaces, open-cover compactness, $\mathsf{Real}^{\uparrow}$ *completeness, or selected minimizers.*

Proof. Carrier admission is projection from Definition 2082.1. The accepted packet displays the metric source, interval path row, partition ledger, segment-distance ledger, segment sum, readback rows, and structural rows.

Path-source exactness is read from M and I. A path endpoint can be inspected only after the source metric row and the displayed interval path row have both been opened.

Partition exactness is the role of A. It supplies the finite ordered interval addresses and adjacent pairs consumed downstream, so the segment-distance ledger has no unlisted path coordinate.

Segment-distance exactness is read from D. Each adjacent pair is measured through the same metric source row M, and no additional distance object is present.

Length-readback exactness is read from S, R, T. The finite segment distances are summed in S, exposed as regular rational observations in R, and sealed as a real-name length value in T.

Finally C, P, N add no analytic coordinate. They preserve, replay, source, and name the same finite route, so the excluded geodesic, host-path, quotient, compactness, completeness, and minimizer data have no row from which a consumer could read them. □

Closure status for $\mathsf{NameCert}_{\mathsf{LengthSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LengthSpace}^{\uparrow}$ packet is generated from a Metric source row, an Interval path row, a finite partition ledger, segment-distance rows, finite segment sums, RegSeqRat and Real length readbacks, componentwise hsame transport, displayed Cont replay with Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LengthSpace}^{\uparrow}}$, seed)

Scope closed: Definition 2082.1, Definition 2082.2, and Definition 2082.3 seed the finite path-length metric ledger surface for downstream path-metric and geodesic consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LengthSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close geodesic existence, host paths, quotient path spaces, open-cover compactness, $\mathsf{Real}^{\uparrow}$ completeness, selected minimizers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, path-source exactness, partition exactness, segment-distance exactness, length-readback exactness, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LengthSpaceUp.

Chapter 2083

A Concrete Naming Certificate for $\mathsf{ScottReal}^{\uparrow}$

Following Scott, Gierz–Hofmann–Keimel–Lawson–Mislove–Scott, and Bishop–Bridges, $\mathsf{ScottReal}^{\uparrow}$ is the finite interval-domain presentation of a real name. It sits after the interval-domain carrier of chapter 1543, the way-below refinement of chapter 1545, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet records the domain-theoretic real presentation as displayed NameCert data, not as quotient real equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2083.1 (ScottReal carrier)**.** *A* $\mathsf{ScottReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, W, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{IntervalDomain}^{\uparrow}$ *row. It records the finite rational interval enclosure surface and the containment checks visible at the current request.*
2. *W is the* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *row. It records the compact-containment and way-below refinements used to move between coarser and sharper interval observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger. It stores dyadic endpoints, radii, slack, denominator budgets, and comparison witnesses for the displayed interval refinements.*
4. *M is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It schedules the finite interval observations consumed by a request without selecting an infinite stream representative.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the same displayed windows and dyadic budgets as regular rational observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is reachable only after the interval-domain, way-below, dyadic, stream-window, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transport for the interval-domain, way-below, dyadic, stream, regular-rational, real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite real-facing request through $I, W, D, M, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ScottReal}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ScottReal}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets through the displayed tuple $I, W, D, M, R, E, H, C, P, N$ and has no coordinate for quotient real equality, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.ScottRealUp.ScottRealTasteGate single carrier alignment`

Theorem 2083.2 (ScottReal NameCert obligations)**.** *For an accepted* $\mathsf{ScottReal}^{\uparrow}$ *carrier*

$$S = (I, W, D, M, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ScottReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the interval-domain row, way-below row, dyadic refinement ledger, stream window, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*
2. *interval-domain coherence, requiring every exposed interval read to come from I and to preserve the displayed endpoint containment and enclosure order;*
3. *way-below compatibility, requiring each refinement in W to be read as compact containment inside the interval-domain row before a sharper interval observation is consumed;*
4. *stream and readback handoff, requiring M and R to consume the same finite interval and dyadic rows before any regular rational observation is exported;*
5. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E to remain downstream of I, W, D, M, R and to export only the displayed real-name seal;*
6. *ledger refusal, requiring D, H, C, P, N to record the displayed refinement route and to export no quotient real equality, no selected limits, no countable choice, no open-cover compactness, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2083.1. The accepted packet exposes exactly the interval-domain, way-below, dyadic, stream, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Interval-domain coherence is read from I. Every public interval observation is a displayed row with endpoints, containment checks, and enclosure order already present in the packet.

Way-below compatibility is read from W against I. A sharper observation may be consumed only after the compact-containment row has shown how it lies inside the coarser interval-domain observation.

The stream and regular-rational handoff is then finite. The row M schedules only the displayed interval observations, and the row R reads those scheduled observations through the dyadic ledger D rather than through a selected infinite stream.

The real seal is last. Since E is downstream of I, W, D, M, R, a real-facing consumer cannot bypass the interval-domain and readback route.

Finally H, C, P, N are structural rows. They transport, replay, source, and name the displayed route, so the excluded quotient, selection, choice, compactness, completeness, verification, and bridge principles have no coordinate from which they could be read. □

Theorem 2083.3 (ScottReal interval-domain handoff)**.** *Let*

$$S = (I, W, D, M, R, E, H, C, P, N)$$

be an accepted $\mathsf{ScottReal}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of* S *factors through the displayed route*

$$I \longmapsto W, D \longmapsto M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the Scott interval-domain presentation reaches real and completion consumers only by passing from interval enclosures to way-below and dyadic refinements, then to finite stream windows, regular rational readback, the real seal, and the structural hsame/Cont/Pkg/NameCert *rows. It does not export quotient real equality, selected limits, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Proof. Fix a real-facing or completion-facing consumer of the accepted packet. The first visible row is the interval-domain surface I, so the consumer begins with displayed finite interval enclosures.

The refinement step reads W and D. The way-below row records compact containment inside the interval-domain surface, while the dyadic ledger supplies the endpoint, radius, slack, and comparison data used by those refinements.

The window step reads M. A consumer receives only finite scheduled interval observations, not an infinite stream representative or a selected limit.

The regular rational step reads R. This row turns the same displayed windows and dyadic budgets into regular rational observations before any terminal real seal is available.

The seal step reads E only after the preceding rows have been displayed. Hence a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer obtains the real presentation through the stated route.

The rows H, C, P, N close the factorization by transport, replay, provenance, and local naming. Since no other coordinate exists in the carrier, the excluded quotient, choice, compactness, completeness, verification, and bridge principles cannot be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScottReal}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ScottReal}^{\uparrow}}$, seed)

Scope closed: Definition 2083.1, Definition 2083.2, and Definition 2083.3 seed the finite interval-domain presentation of real names through way-below refinements, dyadic ledgers, stream windows, regular rational readback, and the terminal real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ScottReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-domain coherence, way-below compatibility, dyadic refinement replay, stream and RegSeqRat handoff, Real seal non-escape, and structural transport for BEDC.Derived.ScottRealUp.

Constructive story: A $\mathsf{ScottReal}^{\uparrow}$ packet is generated from an IntervalDomain row, an IntervalDomainWayBelow row, a DyadicRatCore refinement ledger, a StreamName finite window, a RegSeqRat handoff, a Real seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2084

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is restricted to a displayed tail before any downstream real-name consumer inspects the seal. It sits over the finite observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance discipline of chapter 291, and the terminal real-name boundary of chapter 38. It is the ordinary tail-restriction surface that later cofinal routes, including chapter 1998, can consume as finite BEDC data rather than as host stream reasoning.

Definition 2084.1 (Regular-Cauchy tail-subsequence carrier)**.** *A* $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, W, k, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row, exposing only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and regular rational data admitted by the source certificate.*
2. *W is the tail-window ledger, listing the source windows that remain visible after the cutoff is displayed.*
3. *k is a unary tail cutoff. It is a finite natural-number row, not an infinite selector or choice function.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the source row, the retained windows, the shifted readback, and the terminal real seal.*
5. *R is the shifted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. A request at displayed index n is read from the source regular Cauchy data at the shifted index $k + n$ through the window ledger W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal exposed only after S, W, k, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source row, tail windows, cutoff, tolerance ledger, shifted readback, seal row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite tail read through*

$$S, W, k, D, R \longmapsto E, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, k, D, R, E, H, C, P, N$. It has no coordinate for a quotient stream, selected limit, countable choice principle, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailSubsequenceUp.RegularCauchyTailSubsequenceTasteGate`
`single carrier alignment`

Theorem 2084.2 (Regular-Cauchy tail-subsequence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ *carrier*

$$T = (S, W, k, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSubsequence}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, tail-window ledger, unary cutoff, dyadic tolerance ledger, shifted regular rational readback, real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *tail cutoff exactness, requiring every retained observation to pass through the displayed unary row k and the corresponding retained window in W;*
3. *shifted readback exactness, requiring R to read the displayed tail index n only as the source observation at $k + n$;*
4. *tolerance preservation, requiring D to be the same dyadic ledger consumed by S, W, R and the seal row E;*
5. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed finite tail route.*

No NameCert obligation is stored outside those five rows, and none exports a quotient stream, selected limit, countable choice principle, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

Proof. Carrier admission is projection from Definition 2084.1. The visible analytic rows are exactly the source regular Cauchy row S, tail windows W, unary cutoff k, dyadic ledger D, shifted readback R, and real seal E.

Tail cutoff exactness is read from k and W. A tail observation cannot be inspected until the cutoff row has been displayed and the retained source window has been named in the window ledger.

Shifted readback exactness is read from R. Each displayed tail index n is interpreted by the finite arithmetic row $k + n$, so the packet uses a visible unary shift rather than an external subsequence function.

Tolerance preservation is the projection to D. The source row, retained windows, shifted readback, and real seal all consume that same dyadic ledger, so no detached modulus can alter the tail read.

Finally H, C, P, N are structural rows. They preserve, replay, source, and name only the displayed finite route, leaving no row from which a consumer could extract quotient streams, selected limits, countable choice, host equality, trichotomy, ambient completeness, Lean verification, or bridge data. □

Theorem 2084.3 (Regular-Cauchy tail-subsequence Real seal stability)**.** *Let*

$$T = (S, W, k, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the tail packet factors through*

$$S, W, k, D, R, E, H, C, P, N.$$

Consequently the consumer reads the tail as a finite cutoff, retained-window ledger, shifted regular rational readback, shared dyadic tolerance ledger, and terminal seal. It exports no quotient stream, selected limit, countable choice principle, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ consumer request. The carrier exposes the seal row E only downstream of S, W, k, D, R, so the consumer must first read the source row, retained windows, displayed cutoff, dyadic ledger, and shifted regular rational readback.

The tail operation is finite. The cutoff row k supplies the unary shift, and the window ledger W records the retained source windows; no host subsequence function is available to the consumer.

The shifted readback and tolerance rows then determine the seal-facing data. The row R reads index n through the source observation at $k + n$, while D is the sole dyadic ledger shared by the source, tail windows, readback, and seal.

The structural rows close the route. Componentwise transport, continuation replay, package provenance, and local naming preserve only S, W, k, D, R, E, so the consumer has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailSubsequence}^{\uparrow}$ packet is generated from a source regular Cauchy row, retained StreamName windows, a unary tail cutoff, a shared DyadicRatCore tolerance ledger, shifted RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSubsequence}^{\uparrow}}$, seed)

Scope closed: Definition 2084.1, Definition 2084.2, and Definition 2084.3 seed the finite tail-restriction route for regular Cauchy rational names over displayed StreamName windows, a unary cutoff, dyadic tolerance ledgers, shifted RegSeqRat readback, and the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailSubsequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail cutoff exactness, shifted readback exactness, tolerance preservation, and Real-seal stability for BEDC.Derived.RegularCauchyTailSubsequenceUp.

Chapter 2085

A Concrete Naming Certificate for $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name exposes an explicit precision rate before a $\mathsf{Real}^{\uparrow}$ consumer reads the seal. It sits over the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the generic modulus row of chapter 292, the Real-facing modulus boundary of chapter 1423, and the terminal real-name seal of chapter 38. The packet makes the rate read part of the named certificate rather than a host-side inference from Cauchy convergence prose.

Definition 2085.1 (Regular Cauchy rate extraction carrier)**.** *A* $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, R, M, U, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the source regular Cauchy name and by every downstream precision request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying requested radii, comparison cells, slack rows, and endpoint observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic observations.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row. A requested precision is sent to a displayed finite stage through M before any rate-bearing handoff is available.*
5. *U is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *readback row binding the same S, D, R, M data to the Real-facing modulus surface.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after S, D, R, M, U have all been displayed.*
7. *H records componentwise* hsame *transport for the window, dyadic, regular readback, modulus, Real-facing modulus, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the rate route*

$$S, D, R \longmapsto M \longmapsto U \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRateExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, M, U, E, H, C, P, N$. It has no coordinate for a selected limit, quotient equality of Cauchy streams, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRateExtractionUp.RegularCauchyRateExtractionTasteGate single carrier alignment`

Theorem 2085.2 (Regular Cauchy rate extraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *carrier*

$$X = (S, D, R, M, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRateExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, CauchyModulus threshold, RealCauchyModulus readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *window-readback exactness, requiring R to read only windows displayed in S and tolerance data displayed in D;*
3. *threshold extraction exactness, requiring every precision request to pass through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *row M before the Real-facing modulus row is available;*
4. *Real-modulus compatibility, requiring U to bind the same S, D, R, M route and to expose no independent convergence oracle;*
5. *Real-seal nonescape, requiring E to be downstream of S, D, R, M, U and to export no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness claim;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rate-extraction route.*

No $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2085.1. The accepted packet displays exactly the StreamName, DyadicRatCore, RegSeqRat, CauchyModulus, RealCauchyModulus, Real seal, transport, replay, provenance, and local naming rows.

Window-readback exactness is read from S, D, R. The regular rational readback has access only to the displayed finite windows and dyadic tolerance cells, so it cannot inspect an unbounded stream representative or a quotient equality class.

Threshold extraction exactness is the role of M. A precision request reaches a finite stage only through the displayed $\mathsf{CauchyModulus}^{\uparrow}$ row, and no other carrier coordinate can supply a detached selector.

The Real-facing modulus row U then binds the same source route. Since U is downstream of S, D, R, M, it records the rate as certificate data rather than as an external convergence principle.

Finally, E, H, C, P, N close the finite route. The seal and structural rows preserve only the displayed coordinates, so the excluded objects are absent from every accepted packet and the listed obligations exhaust the local NameCert surface. □

Theorem 2085.3 (Regular Cauchy rate extraction modulus readback)**.** *Let*

$$X = (S, D, R, M, U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *carrier. Every modulus or Real-facing consumer of* X *reads a precision rate through the finite route*

$$S, D, R \longmapsto M \longmapsto U \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed StreamName window row, DyadicRatCore tolerance ledger, RegSeqRat readback, CauchyModulus threshold, RealCauchyModulus handoff, Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *data, and receives no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

Proof. Begin with a modulus or Real-facing read of an accepted packet. Projection exposes the ordered carrier rows $S, D, R, M, U, E, H, C, P, N$.

The readback starts at the finite observation surface. The row S displays the windows, D displays the tolerance ledger, and R displays the regular rational values available to the request.

The precision rate is then read from M. Because M is the only threshold row in the carrier, a consumer cannot replace the rate by countable choice, a hidden selector, or a host convergence argument.

The Real-facing handoff is U, and the terminal seal is E. These rows are downstream of the same finite threshold route, so the rate used by the Real seal is the displayed CauchyModulus threshold already named in the packet.

The structural rows H, C, P, N transport, replay, source, and name that same route. Therefore every modulus or Real-facing consumer factors through the displayed packet and receives none of the excluded escape objects. □

Theorem 2085.4 (Regular Cauchy rate extraction finite-threshold induction)**.** *Let*

$$X = (S, D, R, M, U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *carrier. For any finite extension of displayed precision requests that preserves the* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the extended consumer route adds only replay of the displayed chain*

$$S, D, R \longmapsto M \longmapsto U \longmapsto E \longmapsto H, C, P, N.$$

No extension step introduces a selected limit, quotient stream equality, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

Proof. Induct over the finite ledger of displayed precision requests.

For the empty ledger, the claim is exactly the modulus readback theorem Definition 2085.3. The accepted packet exposes S, D, R first, then routes the precision rate through M, U, E, H, C, P, N.

For the successor ledger, append one displayed request while keeping the same S, D, R rows. The appended request is sent to the finite threshold row M because M is the only carrier coordinate that answers precision-stage questions.

After M is read, the Real-facing transport is forced through U and E. The structural rows H, C, P, N replay, transport, source, and name the same route, so the extension step adds no coordinate outside the displayed packet.

The induction therefore preserves the route invariant at every finite stage. Since every step is a replay of $S, D, R \mapsto M \mapsto U \mapsto E \mapsto H, C, P, N$, none of the excluded escape objects can enter the extended ledger. □

Theorem 2085.5 (Regular Cauchy rate extraction obligation closure)**.** *Definition 2085.1, Definition 2085.2, Definition 2085.3, and Definition 2085.4 jointly close the* $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ *obligation surface for the finite rate-readback route over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

Proof. Project the accepted carrier through Definition 2085.1. This gives exactly the rows $S, D, R, M, U, E, H, C, P$, with no coordinate outside the displayed finite packet.

The NameCert-obligations theorem Definition 2085.2 accounts for carrier admission, window-readback exactness, threshold extraction exactness, Real-modulus compatibility, Real-seal nonescape, and structural stability. These rows cover the carrier, window, threshold, Real-facing handoff, and Real-seal obligations.

The modulus-readback theorem Definition 2085.3 supplies the consumer factorization. Every modulus or Real-facing read passes through $S, D, R \mapsto M \mapsto U \mapsto E \mapsto H, C, P, N$, so the rate is certificate data rather than host-side convergence data.

The finite-threshold induction theorem Definition 2085.4 covers finite precision-ledger growth. Each displayed request reuses the same threshold route and adds no new carrier coordinate.

Together these blocks exhaust the selected obligation surface. The closure therefore introduces no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge credential. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRateExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRateExtraction}^{\uparrow}$ packet is generated from StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a CauchyModulus threshold row, a RealCauchyModulus handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateExtraction}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 2085.1, Definition 2085.2, Definition 2085.3, Definition 2085.4, and Definition 2085.5 close the finite rate-readback obligation surface from regular Cauchy rational data through displayed modulus thresholds, Real-facing modulus rows, finite threshold ledgers, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRateExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, threshold extraction exactness, Real-modulus compatibility, Real-seal nonescape, and structural stability for BEDC.Derived.RegularCauchyRateExtractionUp.

Chapter 2086

A Concrete Naming Certificate for $\mathsf{HausdorffMetric}^{\uparrow}$

Following the Hausdorff–Kuratowski compact-set metric reading, the $\mathsf{HausdorffMetric}^{\uparrow}$ packet records the finite-net route by which two nonempty compact-subset names expose a displayed distance bound without importing an ambient supremum or a choice-selected nearest point. It sits over the compact-metric source of chapter 129, the metric carrier of chapter 57, the finite-net discipline of chapter 127, and the precompact metric surface of chapter 1531. The packet is a NameCert interface for compact-subset distance reads; it is not an open-cover compactness theorem, a host hyperspace construction, a quotient subset equality principle, a supremum principle, or a nearest-point selector.

Definition 2086.1 (Hausdorff metric finite-net carrier)**.** *A* $\mathsf{HausdorffMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, B, M, K, E, D, S, T, C, P, N)$$

with the following displayed rows:

1. *A and B are the two nonempty compact-subset name rows. Each row records the displayed subset identifier, nonempty witness address, compact-subset admission data, and the finite request for which the Hausdorff read is being exposed.*
2. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row consumed by both compact-subset names, including the point-distance reads used by the finite-net comparison.*
3. *K records the two* $\mathsf{CompactMetric}^{\uparrow}$ *rows, one for each compact-subset name, together with the compact-metric admission data needed before a finite net may be inspected.*
4. *E records finite* $\mathsf{ProbeBundle}$ *nets for A and B at a displayed positive* $\mathsf{Rat}^{\uparrow}$ *tolerance, including the mesh budget, net membership rows, and coverage witnesses relative to the compact-metric rows in K.*
5. *D records point-to-set distance witnesses from each displayed net point in the first compact-subset name to the second compact-subset name, and from each displayed net point in the second compact-subset name to the first compact-subset name.*
6. *S records the two symmetric bound rows obtained from D, including the rational slack showing that the two directed finite-net estimates share the same tolerance budget.*

7. *T records componentwise* hsame *transport for the subset names, metric row, compact-metric rows, finite nets, point-to-set witnesses, symmetric bounds, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$A, B, M, K, E \longmapsto D \longmapsto S.$$

9. *P is the* Pkg *provenance over* $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{PrecompactMetric}^{\uparrow}$, ProbeBundle, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, M, K, E, D, S, T, C, P, N$. It has no coordinate for open-cover compactness, host hyperspace membership, quotient subset equality, ambient supremum formation, infimum formation, or a choice-selected nearest point.

Lean4 (definition) `BEDC.Derived.HausdorffMetricUp.HausdorffMetricUp`
Lean4 ✓ `BEDC.Derived.HausdorffMetricUp.HausdorffMetricTasteGate single carrier alignment`

Theorem 2086.2 (HausdorffMetricUp NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMetric}^{\uparrow}$ *carrier*

$$H = (A, B, M, K, E, D, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two nonempty compact-subset names, the ambient metric row, the compact-metric rows, the finite-net rows, the point-to-set witness rows, the symmetric bound rows, transport, replay, provenance, and local naming rows;*

2. *compact-subset net exactness, requiring every displayed* ProbeBundle *net in E to be read through the corresponding compact-subset name and compact-metric row before a point-distance request can be inspected;*

3. *symmetric point-distance bound stability, requiring the two directed estimates in D and S to use the same positive* $\mathsf{Rat}^{\uparrow}$ *tolerance budget and to survive componentwise* hsame *transport in T;*

4. *classifier transport, requiring any accepted comparison of packets to preserve the same subset-name rows, metric source, compact-metric witnesses, finite nets, point-to-set witnesses, and symmetric rational bounds;*

5. *ledger exhaustion, requiring C, P, N to replay, source, and name only the displayed finite route and to export no open-cover compactness theorem, host hyperspace object, quotient subset equality, ambient supremum, infimum, or selected nearest point.*

No Hausdorff metric obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.HausdorffMetricUp.HausdorffMetricCarrier namecert obligations`

Theorem 2086.3 (Hausdorff compact-subset distance bound)**.** *Every Metric-facing, CompactMetric-facing, or hyperspace-facing consumer of an accepted* $\mathsf{HausdorffMetric}^{\uparrow}$ *carrier factors through the finite route*

$$A, B, M, K, E \longmapsto D \longmapsto S \longmapsto T, C, P, N.$$

Consequently the consumer receives only the displayed nonempty compact-subset names, the shared metric row, the compact-metric rows, the finite ProbeBundle *nets at a positive* $\mathsf{Rat}^{\uparrow}$ *tolerance,*

the point-to-set distance witnesses, the symmetric finite-net bounds, componentwise transport, replay, provenance, and the local name. It cannot claim an open-cover compactness theorem, host supremum, quotient subset equality, infimum, or choice-selected nearest point from the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMetric}\uparrow}$.

Constructive story: A $\mathsf{HausdorffMetric}^{\uparrow}$ packet is generated from two nonempty compact-subset names, CompactMetricUp rows, a shared MetricUp source, finite ProbeBundle nets at a positive RatUp tolerance, point-to-set distance witnesses, symmetric bound rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMetric}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2086.1, Definition 2086.2, and Definition 2086.3 seed the finite-net Hausdorff distance interface for nonempty compact-subset names without importing compactness, quotient, supremum, infimum, or nearest-point principles.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMetric}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host hyperspaces, quotient subset equality, ambient suprema or infima, selected nearest points, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-subset net exactness, symmetric point-distance bound stability, classifier transport, and ledger exhaustion for BEDC.Derived.HausdorffMetricUp.

Chapter 2087

A Concrete Naming Certificate for $\mathsf{LowerUpperRealCut}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LowerUpperRealCut}^{\uparrow}$ packet records the finite two-sided located cut surface obtained by reading lower observations through chapter 1446, upper observations through chapter 1447, located gap evidence through chapter 404, and the regular-Cauchy comparison boundary through chapter 1745 before the terminal chapter 38 seal is available.

Definition 2087.1 (Lower-upper real cut carrier)**.** *A* $\mathsf{LowerUpperRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *ledger. It records the finite lower observations and their dyadic comparison budgets.*
2. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *ledger. It records the finite upper observations and their dyadic comparison budgets.*
3. *G is the located gap and apartness row. It binds the lower and upper ledgers to the same displayed located-cut request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the dyadic endpoints, radii, slack, and comparison witnesses used by L, U, G.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It schedules the visible lower-upper reads without selecting an infinite stream representative.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison handoff. It reads the same two-sided windows through the regular-Cauchy comparison boundary.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is reachable only after L, U, G, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the lower ledger, upper ledger, located gap row, dyadic tolerance row, stream window, regular-Cauchy handoff, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for every real-facing read through the same tuple $L, U, G, D, S, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LowerUpperRealCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerUpperRealCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, G, D, S, R, E, H, C, P, N$. It has no coordinate for quotient cuts, host suprema, selected cuts, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LowerUpperRealCutUp.LowerUpperRealCutTasteGate single carrier alignment`

Theorem 2087.2 (Lower-upper real cut NameCert obligations)**.** *For an accepted* $\mathsf{LowerUpperRealCut}^{\uparrow}$ *carrier*

$$K = (L, U, G, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerUpperRealCut}^{\uparrow}}$ *surface consists exactly of the following five finite rows:*

1. *carrier exposure for the lower ledger, upper ledger, located gap row, dyadic tolerance row, stream window, regular-Cauchy handoff, real seal, componentwise transport, replay, provenance, and local naming row;*
2. *lower ledger stability, requiring every lower observation to be read through L, D, S, H, C, P, N and to preserve its displayed dyadic budget under componentwise* hsame *transport;*
3. *upper ledger stability, requiring every upper observation to be read through U, D, S, H, C, P, N and to preserve its displayed dyadic budget under componentwise* hsame *transport;*
4. *located gap exactness, requiring the apartness row G to bind the displayed lower and upper observations to the same finite located-cut request before the regular-Cauchy handoff is consumed;*
5. *ledger non-escape, requiring R and E to remain downstream of L, U, G, D, S and to export no quotient cuts, host suprema, selected cuts, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2087.1. The accepted packet displays exactly the lower ledger, upper ledger, located gap row, dyadic tolerance row, stream window, regular-Cauchy handoff, real seal, and structural rows.

Lower and upper ledger stability are componentwise. The row H transports the visible lower and upper observations, while C replays their dyadic budgets through the same finite window and provenance rows.

Located gap exactness is read from G against L and U. The apartness evidence is a displayed finite row of the packet, so it binds the two one-sided ledgers before a regular-Cauchy comparison can be exported.

Ledger non-escape follows from the row inventory. The handoff R and the seal E sit downstream of L, U, G, D, S, and no coordinate stores a quotient cut, host supremum, selected cut, trichotomy branch, countable-choice selector, or ambient completeness principle. □

Theorem 2087.3 (Lower-upper real cut Cauchy handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{LowerUpperRealCut}^{\uparrow}$ *packet factors through the displayed tuple*

$$L, U, G, D, S, R, E, H, C, P, N$$

and cannot bypass the regular-Cauchy comparison boundary. Thus the two-sided cut surface reaches the terminal real seal only by reading the lower ledger, upper ledger, located gap evidence, dyadic tolerances, finite stream windows, regular-Cauchy handoff, and structural hsame/Cont/Pkg/NameCert *rows.*

Proof. Fix an accepted $\mathsf{LowerUpperRealCut}^{\uparrow}$ carrier. A real-facing consumer first sees the one-sided ledgers L and U together with the dyadic tolerance row D, so every lower or upper observation is a displayed finite read.

The located row G binds those observations to one apartness request. The finite window row S then schedules the same lower-upper evidence without selecting an infinite stream representative.

The comparison handoff is R. Since R is downstream of L, U, G, D, S, a consumer cannot reach the terminal seal before passing through the regular-Cauchy comparison boundary.

The terminal seal E and the rows H, C, P, N only seal, transport, replay, source, and name the displayed route. Because no other carrier coordinate exists, the handoff cannot bypass the boundary or export quotient cuts, host suprema, selected cuts, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerUpperRealCut}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerUpperRealCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2087.1, Definition 2087.2, and Definition 2087.3 seed the finite two-sided located cut surface between one-sided real ledgers and the regular-Cauchy real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerUpperRealCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind construction of $\mathsf{Real}^{\uparrow}$, host supremum, quotient cut equality, selected representative, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier exposure, lower ledger stability, upper ledger stability, located gap exactness, ledger non-escape, and the regular-Cauchy handoff for BEDC.Derived.LowerUpperRealCutUp.

Constructive story: A $\mathsf{LowerUpperRealCut}^{\uparrow}$ packet is generated from LowerReal and UpperReal ledgers, a located gap row, a DyadicRatCore tolerance row, a StreamName finite window, a RegSeqRat comparison handoff, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2088

Located choice sequence NameCert construction

Following Bishop and Bridges, the $\mathsf{LocatedChoiceSequence}^{\uparrow}$ packet records the finite located prefix data by which a choice sequence becomes usable as regular rational evidence. It sits between the prefix surface of chapter 1482, the observation schedule of chapter 126, the finite rate row of chapter 353, and the regular rational handoff of chapter 301. The packet is a bridge of finite means: it names only displayed prefixes, located comparison rows, rate requests, and downstream handoff rows, and it does not identify an infinite host function or a completed real.

Definition 2088.1 (Located choice sequence carrier)**.** *A* $\mathsf{LocatedChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, S, W, A, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier whose visible answers are being read as located data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule naming the finite prefix positions inspected by the consumer.*
3. *W is the prefix-window ledger listing exactly the displayed observations of Q used by this packet.*
4. *A is the answer ledger over W, carrying the displayed* $\mathsf{Nat}^{\uparrow}$ *or rational-code rows supplied by the choice-sequence prefix.*
5. *D is the located comparison ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, recording the inspected lower/upper or apartness comparisons attached to the answers in A.*
6. *M is the* $\mathsf{RegularityModulus}^{\uparrow}$ *row assigning every requested tolerance to a displayed prefix window before any regular rational readback is consumed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, W, A, D, M.*
8. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff R is displayed.*
9. *H stores componentwise* hsame *transport for the choice prefix, schedule, window, answer ledger, located comparisons, modulus row, handoff, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a precision request through the finite prefix window, the located comparison ledger, the regularity modulus row, and the regular rational handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedChoiceSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularityModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedChoiceSequence}^{\uparrow}}$ *row.*

The regularity modulus row is displayed as the finite dependency route

$$\begin{aligned} \textit{precision request} &\longmapsto (S, W) \\ &\longmapsto (A, D) \\ &\longmapsto M \\ &\longmapsto R. \end{aligned}$$

Thus a consumer obtains a regular rational readback only after the finite prefix observations and located comparison rows have supplied the requested modulus witness.

Lean4 ✓ `BEDC.Derived.LocatedChoiceSequenceUp.LocatedChoiceSequenceTasteGate single carrier alignment`

Theorem 2088.2 (Located choice sequence modulus stability)**.** *Let*

$$L = (Q, S, W, A, D, M, R, E, H, C, P, N)$$

be an accepted $\mathsf{LocatedChoiceSequence}^{\uparrow}$ *carrier. If a precision request is transported through the displayed* hsame *row H and replayed through the continuation row C, then the same finite prefix window W, answer ledger A, located comparison ledger D, and regularity modulus row M repack as an accepted* $\mathsf{LocatedChoiceSequence}^{\uparrow}$ *carrier for the transported request. No transported read can inspect a prefix position outside W.*

Proof. Project the carrier of Definition 2088.1. The schedule S, window W, answer ledger A, comparison ledger D, and modulus row M are all displayed coordinates of one finite history.

Transport the precision request and the displayed rows through H. The continuation row C records the replay route from the request to the same finite prefix observations and then to the same regularity modulus row, so the transported read does not need a new tail value.

Repack the transported rows with the existing provenance P and local name N. Since the carrier has no coordinate for a completed host function or hidden tail oracle, the resulting carrier can expose only the prefix positions already listed in W. □

Theorem 2088.3 (Located choice sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedChoiceSequence}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix carrier, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the finite located comparison ledger, and the* $\mathsf{RegularityModulus}^{\uparrow}$ *row before it can read the handoff row R. The regular rational handoff receives exactly the finite route*

$$\mathsf{ChoiceSequence}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegularityModulus}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}$$

and no quotient stream equality, countable-choice selector, or completed real-name equality.

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$ read of L. The carrier first exposes the choice-sequence prefix Q and the stream schedule S; without these rows there is no named finite position at which a prefix answer can be inspected.

The answer ledger A is read only over the finite window W. The located comparison ledger D then records the dyadic comparison data attached to those answers, and the regularity modulus row M assigns the precision request to that displayed window.

Only after these rows are present can the handoff row R be consumed. The listed route is therefore forced by projection of the carrier, and none of the excluded objects appears as a coordinate that could be passed to the $\mathsf{RegSeqRat}^{\uparrow}$ consumer. □

Theorem 2088.4 (Located choice sequence NameCert obligation surface)**.** *The seed* $\mathsf{NameCert}_{\mathsf{LocatedChoiceSequence}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for Definition 2088.1, modulus stability of Definition 2088.2, RegSeqRat handoff of Definition 2088.3, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* N*. The surface records how finite choice-sequence prefix information becomes located regularity data consumable by* $\mathsf{RegSeqRat}^{\uparrow}$ *while exporting no completed host function, countable-choice selector, quotient stream equality,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, open-cover compactness principle, or standard bridge.*

Proof. Carrier admission is the twelve-coordinate packet of Definition 2088.1. It lists the choice prefix, stream schedule, prefix window, answer ledger, located comparison ledger, modulus row, regular rational handoff, optional real seal, and structural rows.

Modulus stability is supplied by Definition 2088.2. It shows that transported precision requests reuse the same displayed prefix window and regularity modulus row rather than reading an undisplayed tail.

The RegSeqRat handoff is supplied by Definition 2088.3. It forces every regular rational consumer through the choice prefix, stream schedule, located comparison data, and regularity modulus before the handoff row is available.

The remaining obligations are componentwise. The rows H, C, P, N transport, replay, source, and name exactly the same displayed coordinates, so there is no additional obligation outside the packet. The excluded host functions, selectors, quotient equalities, completeness principles, compactness principles, and bridge rows are absent from the carrier and therefore absent from the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedChoiceSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedChoiceSequence}^{\uparrow}$ packet is generated from a displayed ChoiceSequence prefix, StreamName schedule, finite answer ledger, DyadicRatCore located comparison rows, a RegularityModulus row, RegSeqRat handoff data, optional RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedChoiceSequence}^{\uparrow}}$, seed)

Scope closed: Definition 2088.1, Definition 2088.2, Definition 2088.3, and Definition 2088.4 seed the finite located choice-sequence route from displayed prefixes through regularity modulus rows into RegSeqRat handoff data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedChoiceSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close completed host functions, countable-choice selectors, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus stability, RegSeqRat handoff, transport, replay, provenance, local naming, and non-escape for the finite located choice-sequence packet.

Chapter 2089

A Concrete Naming Certificate for $\mathsf{DyadicLocatedCut}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicLocatedCut}^{\uparrow}$ packet records the finite dyadic endpoint surface by which located cut data can be read as regular-Cauchy evidence before the terminal real-name seal is consumed. It refines the located-cut route of chapter 404 through the dyadic endpoint ledger of chapter 291, the finite-window discipline of chapter 126, the regular rational handoff of chapter 301, and the terminal chapter 38 seal. It also sits beside the two-sided real cut surface of chapter 2087, with the dyadic endpoint rows made explicit rather than absorbed into general lower and upper ledgers.

Definition 2089.1 (DyadicLocatedCut carrier)**.** *A* $\mathsf{DyadicLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, G, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic endpoint ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$*. It records lower endpoints, requested radii, comparison cells, and endpoint reuse provenance.*
2. *U is the upper dyadic endpoint ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$*. It records upper endpoints, requested radii, comparison cells, and endpoint reuse provenance.*
3. *G is the located gap evidence through* $\mathsf{LocatedCut}^{\uparrow}$*. It binds each displayed lower endpoint and upper endpoint to the same located-cut request.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite precision window. It lists exactly the endpoint reads opened by the requested precision and does not select an infinite representative.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. It reads the endpoint-normalized windows as regular rational data before any real seal is available.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It can be reached only after L, U, G, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the lower endpoint ledger, upper endpoint ledger, located gap row, finite window, regular-Cauchy handoff, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for every consumer read through the same ordered tuple $L, U, G, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicLocatedCut}^{\uparrow}$*,* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LowerUpperRealCut}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicLocatedCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, G, W, R, E, H, C, P, N$. It has no coordinate for quotient cuts, selected suprema, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicLocatedCutUp.TasteGate.DyadicLocatedCutTasteGate single carrier alignment`

Theorem 2089.2 (DyadicLocatedCut NameCert obligations)**.** *For an accepted* $\mathsf{DyadicLocatedCut}^{\uparrow}$ *carrier*

$$K = (L, U, G, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicLocatedCut}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower dyadic endpoint ledger, upper dyadic endpoint ledger, located gap row, finite precision window, regular-Cauchy handoff, terminal real seal, transport, replay, provenance, and local naming row;*
2. *lower dyadic transport, requiring every lower endpoint read to enter through L and to preserve its* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budget under componentwise* hsame *transport;*
3. *upper dyadic transport, requiring every upper endpoint read to enter through U and to preserve its* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budget under componentwise* hsame *transport;*
4. *located-gap exactness, requiring G to bind the displayed lower and upper endpoint rows to one* $\mathsf{LocatedCut}^{\uparrow}$ *request before W or R is consumed;*
5. *regular-Cauchy handoff, requiring R to read only the displayed L, U, G, W route before exposing any* $\mathsf{Real}^{\uparrow}$*-facing datum;*
6. *Real seal non-escape, requiring the terminal row E to remain downstream of L, U, G, W, R and to export no quotient cut, selected supremum, trichotomy branch, countable-choice selector, or ambient completeness principle;*
7. *ledger provenance, requiring P and N to name only the displayed dyadic endpoint route and its transport and replay rows.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DyadicLocatedCutUp.DyadicLocatedCutCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 2089.1. The accepted packet displays the lower endpoint ledger, upper endpoint ledger, located gap row, precision window, regular-Cauchy handoff, real seal, and structural rows H, C, P, N.

Lower and upper dyadic transport are componentwise. The rows L and U store the endpoint comparisons over $\mathsf{DyadicRatCore}^{\uparrow}$, while H transports those endpoint cells and C replays the same displayed reads.

Located-gap exactness is the role of G. Since G is a coordinate of the same packet as L and U, the lower and upper endpoint rows are bound to one located-cut request before a window or regular-Cauchy handoff can be consumed.

The handoff row R is downstream of L, U, G, W. Thus every regular rational read is endpoint-normalized by the displayed ledgers and finite precision window before the terminal real seal is visible.

The terminal row E and the provenance rows P, N add no analytic coordinate. Because the carrier has no row for quotient cuts, selected suprema, trichotomy, countable choice, or ambient completeness, the local NameCert surface cannot export any such object. □

Theorem 2089.3 (DyadicLocatedCut regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicLocatedCut}^{\uparrow}$ *packet factors through the displayed dyadic endpoint ledgers and located-gap row before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, every such consumer reads the route*

$$L, U, G, W, R, E, H, C, P, N$$

with L and U supplying dyadic endpoint-normalized lower and upper evidence, G supplying the located cut gap, W supplying the finite precision window, R supplying the regular-Cauchy handoff, and E supplying the terminal seal.

Proof. Fix an accepted $\mathsf{DyadicLocatedCut}^{\uparrow}$ carrier. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer first sees only the displayed endpoint rows L and U together with the located gap row G, so its input is already normalized by dyadic lower and upper endpoint ledgers.

The finite precision window W is downstream of those rows. It schedules the visible endpoint reads without choosing an infinite stream representative or introducing a host cut object.

The regular-Cauchy handoff is R. Since R is read after L, U, G, W, every regular rational consumer factors through the dyadic endpoint ledgers and the located-gap row.

A $\mathsf{Real}^{\uparrow}$ consumer must pass through E, and E is downstream of R. The remaining rows H, C, P, N only transport, replay, source, and name the same route, so no consumer can bypass the displayed endpoint and located-gap data before reading the terminal seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicLocatedCut}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicLocatedCut}^{\uparrow}}, \mathsf{seed})$

Constructive story: A $\mathsf{DyadicLocatedCut}^{\uparrow}$ packet is generated from lower and upper dyadic endpoint ledgers, located gap evidence, a StreamName finite precision window, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Scope closed: Definition 2089.1, Definition 2089.2, and Definition 2089.3 seed the finite dyadic endpoint route from located cut data to regular-Cauchy readback and the terminal real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicLocatedCut}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DyadicLocatedCutUp`

Bridge status: none

Not claimed: This chapter does not close quotient cuts, selected suprema, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper dyadic transport, located-gap exactness, regular-Cauchy handoff, Real seal non-escape, and ledger provenance for BEDC.Derived.DyadicLocatedCutUp.

Chapter 2090

A Concrete Naming Certificate for $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed rational Cauchy name reaches the real-name seal after its modulus data, regular readback, dyadic tolerance rows, and fast-convergent handoff have all been exposed. It sits after the finite window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the choice-free modulus surface of chapter 2040, the fast real-sequence handoff of chapter 2077, and the terminal real-name seal of chapter 38. The packet is a Bishop-style Cauchy-to-real handoff criterion; it is not a selected limit, quotient equality of Cauchy streams, countable choice, host real completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2090.1 (Bishop Cauchy limit criterion carrier)**.** *A* $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, M, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the finite observation windows and precision requests opened by the Cauchy read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows in S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording endpoint observations, requested radii, split tolerances, slack cells, and comparison budgets for the readback.*
4. *M is the Cauchy-modulus row read through* $\mathsf{CauchyModulusChoice}^{\uparrow}$*; it may be either the displayed selected modulus row or its selected normal-form representative, but in both cases the selected threshold is available only through the finite route named by chapter 2040.*
5. *F is the* $\mathsf{FastConvergentRealSequence}^{\uparrow}$ *handoff row that packages the same windows, readbacks, dyadic tolerances, and modulus threshold as a rate-bearing real-sequence route.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after S, R, D, M, F have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, modulus, fast-sequence handoff, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S \longmapsto R \longmapsto D \longmapsto M \longmapsto F \longmapsto E.$$

9. *P is the* Pkg *provenance over* BishopCauchyLimitCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyModulusChoice$^{\uparrow}$, FastConvergentRealSequence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BishopCauchyLimitCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, F, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected real limit, countable choice, host real completeness, detached convergence, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BishopCauchyLimitCriterionUp.BishopCauchyLimitCriterionTasteGate single carrier alignment`

Theorem 2090.2 (Bishop Cauchy limit criterion NameCert obligations)**.** *For every accepted* BishopCauchyLimitCriterion$^{\uparrow}$ *carrier*

$$K = (S, R, D, M, F, E, H, C, P, N),$$

the local NameCert$_{\text{BishopCauchyLimitCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchyModulusChoice row, FastConvergentRealSequence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *window-readback exactness, requiring R to read only the finite windows displayed by S;*
3. *tolerance exactness, requiring D to display the requested radii, split tolerances, endpoint observations, slack cells, and comparison budgets before any modulus or fast-sequence consumer reads the route;*
4. *modulus-selection nonescape, requiring M to be read through the* CauchyModulusChoice$^{\uparrow}$ *route rather than through countable choice or an external selector;*
5. *fast-sequence compatibility, requiring F to bind the same S, R, D, M rows before the terminal real-name seal is available;*
6. *Real seal nonescape, requiring E to be downstream of S, R, D, M, F and to export no quotient stream, selected limit, detached convergence theorem, host real completeness, or ambient* Real$^{\uparrow}$ *completeness;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed route.*

No Bishop Cauchy limit criterion obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2090.1. The accepted packet displays exactly the window row, regular readback, dyadic tolerance ledger, modulus-choice row, fast-sequence handoff, real seal, and the structural rows.

Window-readback exactness is read from S and R. The regular rational readback is attached to the displayed finite windows, so no quotient stream representative is introduced.

Tolerance exactness is the role of D. Its dyadic radii, slack cells, endpoint observations, and comparison budgets are displayed before the modulus row or the fast-sequence handoff may be consumed.

The modulus-selection condition is local to M. Since M is read through the CauchyModulusChoice$^{\uparrow}$ route, the selected threshold is a named finite row rather than an external choice function.

The fast-sequence handoff is then F. It binds the same windows, readback, dyadic tolerance ledger, and modulus threshold before the terminal Real$^{\uparrow}$ seal can be reached.

Finally, H, C, P, N add only transport, replay, provenance, and local naming. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, detached convergence, host real completeness, or ambient completeness, so the listed rows exhaust the local NameCert surface. □

Theorem 2090.3 (Bishop Cauchy limit criterion modulus seal)**.** *Every accepted* $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$ *carrier*

$$K = (S, R, D, M, F, E, H, C, P, N)$$

reads its Cauchy threshold through the displayed $\mathsf{CauchyModulusChoice}^{\uparrow}$ *row* M *before any* $\mathsf{FastConvergentRealSequen}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the route. Thus the criterion exposes a choice-free modulus seal*

$$S, R, D \longmapsto M \longmapsto F, E$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Start with an accepted carrier and project its rows. The Cauchy threshold row is M, while the observations and tolerances consumed by that threshold are S, R, D.

By Definition 2090.2, M is read only through the $\mathsf{CauchyModulusChoice}^{\uparrow}$ route. Therefore the threshold exposed to downstream consumers is not obtained from countable choice or from a hidden selector.

The fast-sequence handoff F is downstream of the same modulus row. A $\mathsf{FastConvergentRealSequence}^{\uparrow}$ consumer reads the rate-bearing route only after the windows, readback, tolerance ledger, and selected threshold have been displayed.

The real seal E is still later in the route. Structural transport, replay, provenance, and local naming preserve the same displayed rows, so the modulus seal remains choice-free at every consumer boundary. □

Theorem 2090.4 (Bishop Cauchy limit criterion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$ *packet factors through the finite route*

$$S \longmapsto R \longmapsto D \longmapsto M \longmapsto F \longmapsto E \longmapsto H, C, P, N.$$

The Real handoff is therefore a displayed path from StreamName windows through RegSeqRat readback, DyadicRatCore tolerances, the CauchyModulusChoice threshold, the FastConvergentRealSequence route, and the terminal Real seal. It exports no selected limit, quotient equality of Cauchy streams, countable choice, detached convergence, host real completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ or completion-facing read of an accepted packet. Projection exposes the ordered carrier rows $S, R, D, M, F, E, H, C, P, N$.

The readback row R is downstream of the finite windows S, and the dyadic tolerance row D is downstream of the same readback. Thus the consumer cannot start from an unbounded stream representative or a quotient equality class.

The modulus seal theorem places M before both the fast-sequence handoff and the terminal real seal. Hence the Cauchy-to-real route receives a displayed threshold before it can reach F or E.

The terminal row E is the only $\mathsf{Real}^{\uparrow}$ seal coordinate in the packet. The structural rows preserve and name the same finite route, and the excluded principles are absent from the carrier. Therefore every Real or Completion consumer factors through the displayed handoff and receives none of the listed escape objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyLimitCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyLimitCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2090.1, Definition 2090.2, Definition 2090.3, and Definition 2090.4 seed the choice-free Cauchy-to-Real route through StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, CauchyModulusChoice threshold data, FastConvergentRealSequence handoff, Real seal, transport, continuation, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyLimitCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, detached convergence, host real completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, tolerance exactness, modulus-selection nonescape, fast-sequence compatibility, Real seal nonescape, and structural stability for BEDC.Derived.BishopCauchyLi

Constructive story: A $\mathsf{BishopCauchyLimitCriterion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a CauchyModulusChoice threshold row, a FastConvergentRealSequence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2091

A Concrete Naming Certificate for $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is read unchanged through a displayed monotone cofinal reindexing and a finite tail shift before any completion-facing consumer reaches the real-name seal. It sits after the cofinal route surface of chapter 1568, the tail-invariance route of chapter 1578, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet makes reindex invariance visible as BEDC NameCert data rather than as quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2091.1 (Regular-Cauchy reindex-invariance carrier)**.** *A* $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, I, T, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose regular rational Cauchy readback supplies the inspected values and regularity budget.*
2. *I is the accepted* $\mathsf{CauchyReindexing}^{\uparrow}$ *route. It displays a monotone cofinal index map and the finite evidence that every target request is sent to a source request still visible to the source row.*
3. *T is the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *tail route. It records the finite prefix shift needed to align the cofinal route with the retained regular Cauchy windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger opened for the source row, the reindexed route, the shifted tail route, and the seal-facing request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the source row, reindexed windows, tail-shift windows, regular rational readback, and terminal seal.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after Q, I, T, S, D have been displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the reindex route, tail route, windows, dyadic ledger, and regular rational readback have been consumed.*

8. *H records componentwise* hsame *transport for the source row, reindex route, tail route, stream windows, dyadic tolerance ledger, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion-facing request through*

$$Q, I, T, S, D, R \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexInvariance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, I, T, S, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyReindexInvarianceUp.RegularCauchyReindexInvarianceTasteGate`
`single carrier alignment`

Theorem 2091.2 (Regular-Cauchy reindex-invariance NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$ *carrier*

$$J = (Q, I, T, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexInvariance}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, cofinal reindex route, tail-shift route, stream-window ledger, dyadic tolerance ledger, regular rational readback, real seal, componentwise transport, continuation replay, provenance, and local naming rows;*

2. *monotone-cofinal route preservation, requiring every seal-facing request to pass through the displayed* $\mathsf{CauchyReindexing}^{\uparrow}$ *row I before any source observation is inspected;*

3. *tail-shift compatibility, requiring the reindexed request to pass through the displayed* $\mathsf{RegularCauchyShift}^{\uparrow}$ *row T when a finite prefix is discarded;*

4. *dyadic tolerance preservation, requiring D to be the same tolerance ledger consumed by Q, I, T, S, R and the seal row E;*

5. $\mathsf{Real}^{\uparrow}$ *seal handoff, requiring the seal row E to be downstream of I, T, S, D, R and never directly exposed from the source row alone;*

6. *ledger nonescape, requiring H, C, P, N to transport, replay, source, and name only the displayed finite route.*

No NameCert obligation is stored outside these rows, and none exports quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Carrier admission is projection from Definition 2091.1. The visible analytic rows are exactly the source regular Cauchy row Q, cofinal reindex route I, tail-shift route T, stream windows S, dyadic ledger D, readback R, and real seal E.

Monotone-cofinal route preservation is read from I. A seal-facing request cannot inspect a source observation until the displayed reindex route supplies the monotone cofinal source index and its finite admission evidence.

Tail-shift compatibility is read from T. If the cofinal route discards a finite prefix, the packet uses the already displayed $\mathsf{RegularCauchyShift}^{\uparrow}$ row rather than an external tail selector.

Tolerance preservation is the projection to D. The source row, reindex route, tail route, stream windows, readback, and real seal all consume that same dyadic ledger, so no detached modulus can alter the reindexed read.

The seal handoff is the projection to E together with the replay row C. Since E appears only downstream of I, T, S, D, R, every real-facing read follows the displayed route before it reaches the terminal seal.

Finally H, C, P, N are structural rows. They preserve, replay, source, and name only the admitted packet, leaving no coordinate from which quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge data could be read. □

Theorem 2091.3 (Regular-Cauchy reindex-invariance Real seal handoff)**.** *Let*

$$J = (Q, I, T, S, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *read and every completion-facing read from J factors first through the* $\mathsf{CauchyReindexing}^{\uparrow}$ *row I and the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *row T before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and the* $\mathsf{Real}^{\uparrow}$ *seal row E are consumed. Consequently the packet exposes reindex invariance only as the displayed route*

$$Q, I, T, S, D, R \longmapsto E, H, C, P, N$$

and exports no quotient stream, selected limit, countable choice principle, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer request. The carrier exposes the seal row E only downstream of the reindex, tail, window, tolerance, and readback rows, so the consumer must first read I and T before any seal-facing route is available.

The reindex step is finite data. The row I displays the monotone cofinal source request used for the consumer's target request, and the row S records the finite windows in which that request is observed.

The tail step is likewise finite. The row T supplies the $\mathsf{RegularCauchyShift}^{\uparrow}$ route that accounts for the discarded prefix before the readback row R is consumed.

The dyadic and readback rows then determine the seal-facing observation. The ledger D is shared by the source, reindex, tail, window, readback, and seal rows, while R is the only $\mathsf{RegSeqRat}^{\uparrow}$ handoff row available to the consumer.

The structural rows close the route. Transport, replay, provenance, and local naming preserve only Q, I, T, S, D, R, E, so the consumer has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReindexInvariance}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyReindexInvariance}^{\uparrow}$ packet is generated from a RegSeqRat source row, a CauchyReindexing monotone cofinal route, a RegularCauchyShift tail route, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReindexInvariance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2091.1, Definition 2091.2, and Definition 2091.3 seed the finite route by which regular Cauchy rational names are read through displayed monotone cofinal reindexing and finite tail shifts before RegSeqRat readback and RealUp sealing.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyReindexInvariance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-cofinal route preservation, tail-shift compatibility, dyadic tolerance preservation, Real-seal handoff, and ledger nonescape for BEDC.Derived.RegularCauchyReindexInvarianceUp.

Chapter 2092

A Concrete Naming Certificate for $\mathsf{KonigLemma}^{\uparrow}$

Following Koenig's lemma for finitely branching trees, Brouwer's spread reading of prefix trees, and Bishop–Bridges finite evidence, $\mathsf{KonigLemma}^{\uparrow}$ records the L10 finite-prefix boundary where tree and branch arguments may be inspected without choosing an infinite branch. The packet sits after the Boolean prefix surface of chapter 1475, the finite fan boundary of chapter 400, and the barred-spread replay of chapter 1675. It exposes finite frontiers, candidate prefixes, Cantor-space prefix reads, fan/bar handoffs, transport, replay, provenance, and local naming; it does not export weak Koenig principles as global rules, open-cover compactness, quotient streams, selected infinite paths, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2092.1 (KoenigLemma carrier)**.** *A* $\mathsf{KonigLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, W, Q, S, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the finitely branching prefix-tree row. It records the displayed root, finite successor lists, prefix closure checks, and extension rows, with each local branching list read through* $\mathsf{List}^{\uparrow}$ *and each edge decision read through* $\mathsf{Bool}^{\uparrow}$*.*
2. *F is the finite frontier ledger. It lists the currently inspected frontier prefixes, the finite window cut through T, and the local evidence that each retained node either extends inside the window or is recorded as a boundary node for the current request.*
3. *W is the branch-candidate prefix row. It stores a finite compatible chain of prefixes read from T and F, not a completed path and not a choice-selected branch.*
4. *Q is the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix-reading row. It reads every Boolean branch decision through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{Bool}^{\uparrow}$ *observations, and* $\mathsf{List}^{\uparrow}$ *prefix spines.*
5. *S is the* $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *boundary row. It records the finite fan/bar route used when a consumer asks whether the displayed frontier blocks the inspected candidate prefix.*
6. *A is the local classifier row. It compares accepted packets by the displayed tree, frontier, window, candidate-prefix, Cantor-prefix, and fan/bar rows, with no coordinate for quotient path equality or host function extensionality.*

7. *H records componentwise* hsame *transport for the tree, frontier, window, candidate-prefix, Cantor-prefix, fan/bar, classifier, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite tree request through prefix restriction, successor inspection, frontier lookup, candidate-prefix readback, Cantor-space prefix reading, fan/bar boundary use, classifier transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{KonigLemma}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KonigLemma}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these finite rows. It has no coordinate for a selected infinite branch, quotient path space, host function equality, open-cover compactness, weak Koenig principles as ambient rules, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KonigLemmaUp.KonigLemmaTasteGate single carrier alignment`

Theorem 2092.2 (KoenigLemma NameCert obligations)**.** *For every accepted* $\mathsf{KonigLemma}^{\uparrow}$ *carrier*

$$K = (T, F, W, Q, S, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KonigLemma}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the finitely branching prefix-tree row, finite frontier ledger, branch-candidate prefix row, Cantor-space prefix-reading row, fan/bar boundary row, classifier row, transport row, replay row, provenance row, and local naming row;*
2. *classifier transport through the componentwise rows listed in Definition 2092.1;*
3. *prefix-extension stability, requiring every shorter candidate prefix and every displayed one-step extension to reuse the matching rows already present in T, F, W, Q;*
4. *branch-frontier replay, requiring every consumer route that asks for a branch obstruction or branch continuation to pass through the finite frontier ledger and the S fan/bar boundary row;*
5. *non-escape, excluding selected infinite branches, weak Koenig principles as global rules, quotient path spaces, host function equality, open-cover compactness, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside the displayed packet.

Proof. Carrier admission is projection from Definition 2092.1. The accepted packet exposes only the tree, frontier, candidate-prefix, Cantor-prefix, fan/bar, classifier, transport, replay, provenance, and naming rows.

Classifier transport is componentwise. Two accepted packets compare through the displayed prefix-tree and frontier rows first, then through the candidate-prefix row, Cantor-space prefix reads, and fan/bar boundary row. The structural rows H, C, P, N transport and name that same finite comparison.

Prefix-extension stability is inherited from the retained prefix tree. A shorter prefix is a subrow already visible in T and W, while a displayed one-step extension is one of the finite successor rows listed in T and checked against F.

Branch-frontier replay is finite. A consumer asking whether the current window blocks or extends a candidate prefix must read the frontier ledger F and then the fan/bar boundary row S; the route supplies only displayed prefix and boundary evidence.

Coordinate exhaustion gives non-escape. Since the carrier has no slot for a completed path, quotient path equality, host function extensionality, open-cover compactness, weak Koenig rules

as ambient principles, or $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the local certificate. □

Theorem 2092.3 (KoenigLemma Cantor–fan boundary)**.** *Every consumer route from an accepted* $\mathsf{KonigLemma}^{\uparrow}$ *carrier factors through*

$$\mathsf{CantorSpace}^{\uparrow},\ \mathsf{FanTheorem}^{\uparrow},\ \mathsf{BrouwerBarInduction}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

The route may inspect finite tree prefixes, frontier windows, candidate-prefix rows, Cantor-space Boolean prefix reads, and fan/bar boundary rows. It exports no choice-selected infinite branch, quotient path space, host function equality, open-cover compactness, weak Koenig principle as a global rule, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project a consumer route from the accepted carrier of Definition 2092.1. The first visible tree data are finite prefix rows in T and finite frontier rows in F.

The candidate-prefix read is then W. Its Boolean decisions are not host functions; they are read through the $\mathsf{CantorSpace}^{\uparrow}$ row Q, which itself routes through $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{Bool}^{\uparrow}$ observations, $\mathsf{List}^{\uparrow}$ spines, and the surrounding BHist packet.

When the consumer asks whether the current frontier blocks or extends the candidate prefix, the route enters S. That row is exactly the finite $\mathsf{FanTheorem}^{\uparrow}$ and $\mathsf{BrouwerBarInduction}^{\uparrow}$ boundary over the displayed prefix window.

Finally H, C, P, N close the route by hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming. No coordinate remains for a selected infinite branch, quotient path space, host function equality, open-cover compactness, a global weak Koenig rule, or $\mathsf{Real}^{\uparrow}$ completeness, so the stated factorization is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{KonigLemma}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KonigLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2092.1, Definition 2092.2, and Definition 2092.3 seed the finite-prefix tree boundary over Cantor-space prefix reads and fan/bar rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KonigLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected branches, weak Koenig principles as global rules, open-cover compactness, quotient streams, quotient path spaces, host function equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, prefix-extension stability, branch-frontier replay, Cantor-space prefix reading, fan/bar boundary routing, and non-escape for BEDC.Derived.KonigLemmaUp.

Constructive story: A $\mathsf{KonigLemma}^{\uparrow}$ packet is generated from a finitely branching prefix-tree row, a finite frontier ledger, a branch-candidate prefix row, a CantorSpace prefix-reading row, a FanTheorem and BrouwerBarInduction boundary row, classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2093

A Concrete Naming Certificate for $\mathsf{CauchyMetastability}^{\uparrow}$

Following Tao's finite convergence principle and the proof-mining account of metastability, the $\mathsf{CauchyMetastability}^{\uparrow}$ packet records the finite tail-window answer to a Cauchy pressure request before any completed real object is selected. It sits over the stream-window surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the finite modulus discipline of chapter 332. The chapter gives the Real and Completion route a human-standard metastable interface: a displayed control function asks for a finite tail window, and the answer remains finite BEDC data all the way to the $\mathsf{Real}^{\uparrow}$ seal.

Definition 2093.1 (Cauchy metastability carrier)**.** *A* $\mathsf{CauchyMetastability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (p, g, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *p is the precision request. It names the dyadic tolerance that the packet must answer.*
2. *g is the control-function row. It is read only through the finite requests displayed in the packet, not as a countable choice object.*
3. *W is the finite tail-window row. It displays a bound n and the finite interval of indices inspected between n and $g(n)$.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the precision request, the tail-window comparisons, the rational readback, and the terminal real seal.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It supplies the rational values actually compared inside the window named by W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal, exposed only after p, g, W, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the precision request, control row, tail window, tolerance ledger, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay along the finite route*

$$p, g, W, D, Q \longmapsto E, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyMetastability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMetastability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $p, g, W, D, Q, E, H, C, P, N$. It has no coordinate for countable choice, quotient stream equality, a selected limit, host real equality, trichotomy, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.CauchyMetastabilityUp.CauchyMetastabilityTasteGate single carrier alignment`

Theorem 2093.2 (Cauchy metastability NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMetastability}^{\uparrow}$ *carrier*

$$M = (p, g, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMetastability}^{\uparrow}}$ *surface consists exactly of nine finite obligation rows:*

1. *carrier admission for the precision request, control row, tail-window row, tolerance ledger, rational readback, real seal, transport, replay, provenance, and local naming rows;*
2. *control-window exactness, requiring W to answer the displayed control row g at the displayed precision request p;*
3. *dyadic tolerance accounting, requiring every comparison inside W to use the single ledger D;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring each compared value in the finite window to be read from Q;*
5. $\mathsf{Real}^{\uparrow}$ *seal boundary, requiring E to be downstream of p, g, W, D, Q and not an independent completion coordinate;*
6. hsame *transport, requiring accepted comparisons to preserve $p, g, W, D, Q, E, C, P, N$ componentwise;*
7. Cont *replay, requiring every endpoint-facing read to replay the displayed finite route;*
8. Pkg *provenance, requiring the packet to cite only the listed source chapters and kernel objects;*
9. *local naming, requiring N to name precisely the finite metastability packet and no hidden limit data.*

No NameCert obligation is stored outside these rows, and none exports countable choice, quotient stream equality, a selected limit, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2093.1. The accepted packet exposes exactly $p, g, W, D, Q, E, H, C, P, N$, so the local certificate has no detached coordinate for a completed limit or infinite selector.

Control-window exactness is read from g and W. The control row may ask for a tail interval, but the carrier answers it only by the displayed finite window row and the indices named there.

The tolerance and readback obligations are then projections to D and Q. The dyadic ledger is the only tolerance account available to the packet, and the regular rational readback supplies the only values compared inside the named window.

The endpoint boundary is supplied by E together with H, C, P, N. Transport, replay, provenance, and local naming preserve the displayed route from request to window to readback to seal, and they introduce no row from which the excluded principles can be read. □

Theorem 2093.3 (Cauchy metastability finite-tail stability)**.** *Let*

$$M = (p, g, W, D, Q, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyMetastability}^{\uparrow}$ *carrier. Every accepted finite control-window request is answered inside the tail window displayed by* W*: the precision request* p *and control row* g *determine the window, the dyadic ledger* D *accounts for the requested tolerance, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *supplies every compared rational value before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is read. Consequently the packet witnesses finite Cauchy tail stability without countable choice, quotient stream equality, a selected limit, host real equality, trichotomy, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier. The only rows available before the endpoint seal are the precision request p, control row g, tail-window row W, dyadic ledger D, and rational readback Q.

An accepted control-window request is therefore handled inside W. The row g determines which finite interval must be inspected, and W displays the bound and window that answer that request.

The stability comparison is finite. The ledger D records the dyadic tolerance used by every comparison in the window, while Q supplies the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ rational values.

Finally E, H, C, P, N close the endpoint route. Since they only seal, transport, replay, source, and name the displayed finite data, no countable choice principle, quotient stream equality, selected limit, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate can enter the certificate. □

Theorem 2093.4 (Cauchy metastability filter consumer route)**.** *Any* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer reading an accepted* $\mathsf{CauchyMetastability}^{\uparrow}$ *packet receives only the precision request, displayed control-function row, finite tail-window answer,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route exports no quotient stream equality, selected limit, quotient filter, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted metastability carrier through Definition 2093.1. The only rows available are $p, g, W, D, Q, E, H, C, P, N$, namely the request, control row, finite tail window, tolerance ledger, rational readback, endpoint seal, transport, replay, provenance, and local name.

Now route the filter consumer through the CauchyFilter modulus boundary of chapter 333. A filter-facing read may use the finite tail-window answer as Cauchy control data only after the dyadic ledger and RegSeqRat readback rows have been displayed.

The structural rows do not add a second source. The hsame transport, Cont replay, Pkg provenance, and local NameCert row preserve the same finite packet, so no quotient stream equality, selected limit, quotient filter, host equality, choice principle, or ambient completeness coordinate can enter the consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMetastability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyMetastability}^{\uparrow}$ packet is generated from a precision request, a displayed control-function row, a finite tail-window answer, a DyadicRatCore tolerance ledger, RegSeqRat rational readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyMetastability}^{\uparrow}}$, seed)

Scope closed: Definition 2093.1, Definition 2093.2, and Definition 2093.3 seed the finite metastable Cauchy tail-stability route for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyMetastability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient stream equality, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, control-window exactness, dyadic tolerance accounting, RegSeqRat handoff, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMetastabilityUp.

Chapter 2094

A Concrete Naming Certificate for $\mathsf{TanneryTheorem}^{\uparrow}$

Following Bishop and Bridges and the classical theorem of Tannery, the $\mathsf{TanneryTheorem}^{\uparrow}$ packet records the finite dominated-exchange route for a double-indexed real row whose pointwise limit row may be summed after a displayed majorant has supplied tail control. It sits over the real-valued series surface of $\mathsf{RealSeries}^{\uparrow}$ in chapter 1581, the absolute-convergence route of $\mathsf{AbsoluteConvergence}^{\uparrow}$ in chapter 1700, the majorant discipline of $\mathsf{WeierstrassMTest}^{\uparrow}$ in chapter 399, the rational readback row of $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, the finite window interface of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the dyadic tolerance ledger of $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 291, and the terminal real-name seal of $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet names a finite constructive dominated exchange; it does not import host infinite sums, quotient stream equality, countable choice, dominated convergence as an ambient theorem, or ambient real completeness.

Definition 2094.1 (Tannery theorem carrier)**.** *A* $\mathsf{TanneryTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, M, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the double-indexed real row. It records the inspected entries $a_{n,k}$ only through finite real-valued windows and the partial-sum surfaces allowed by* $\mathsf{RealSeries}^{\uparrow}$*.*
2. *L is the pointwise limit row. It records the displayed limiting term l_k for each inspected column together with the finite convergence window that compares $a_{n,k}$ to l_k.*
3. *M is the dominating majorant route. It records the absolute majorant row, the summable majorant handoff, and the domination checks that bind A and L to* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *and* $\mathsf{WeierstrassMTest}^{\uparrow}$*.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family. It stores the row windows, column windows, diagonal comparison windows, and finite cutoffs used by the exchange.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same real entries, limit entries, majorant entries, and partial sums named by A, L, M, W.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the requested precision, tail split, finite head budget, domination slack, and endpoint tolerance used by the exchange.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal. It is available only after the displayed finite row sum, limit-row sum, majorant route, readback, and dyadic ledger have been read.*

8. *H stores componentwise* hsame *transport for A, L, M, W, Q, D, E and for the replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the double-indexed row and pointwise limit row through the majorant route, finite windows, rational readback, dyadic ledger, and endpoint seal.*

10. *P is the* Pkg *provenance over* $\mathsf{TanneryTheorem}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TanneryTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, M, W, Q, D, E, H, C, P, N$. It has no coordinate for a host exchange of limits and sums, a selected infinite row sum, quotient stream equality, countable choice, an ambient dominated convergence theorem, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.TanneryTheoremUp.TanneryTheoremTasteGate single carrier alignment`

Theorem 2094.2 (Tannery theorem NameCert obligations)**.** *For every accepted* $\mathsf{TanneryTheorem}^{\uparrow}$ *carrier*

$$K = (A, L, M, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TanneryTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the double-indexed real row, pointwise limit row, dominating majorant route, finite window family, rational readback, dyadic ledger, real endpoint seal, transport, replay, provenance, and local naming rows;*

2. *pointwise-window exactness, requiring each displayed comparison between $a_{n,k}$ and l_k to pass through A, L, W, Q, D before it can be used in a finite head estimate;*

3. *dominated-row exactness, requiring M to display the absolute majorant row, the summable majorant handoff, and the domination checks for every inspected entry of A and L;*

4. *finite-exchange accounting, requiring W and D to split each requested tolerance into a finite head comparison and a dominated tail comparison;*

5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring Q to read the same rows and windows before any* Real$^{\uparrow}$ *endpoint seal is exported;*

6. Real$^{\uparrow}$ *seal nonescape, requiring E to remain downstream of A, L, M, W, Q, D and to export no host exchange theorem, selected infinite sum, quotient stream equality, countable choice, ambient dominated convergence theorem, Lean verification, or bridge datum;*

7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Tannery dominated-exchange obligation is stored outside these rows.

Proof. Project the accepted packet $K = (A, L, M, W, Q, D, E, H, C, P, N)$. The carrier exposes exactly the double-indexed row, pointwise limit row, majorant route, window family, readback, ledger, endpoint seal, and structural rows.

The pointwise part is finite at every read. A requested column comparison must pass through the row data A, the limit data L, the finite windows W, the rational readback Q, and the tolerance ledger D before it contributes to the finite head estimate.

The dominated part is also displayed. The row M records the majorant, its summable handoff, and the domination checks; the rows W and D record the cutoff and tolerance split used to separate the finite head from the dominated tail.

Finally Q, E, H, C, P, N read back, seal, transport, replay, source, and name the same route. Since no excluded exchange theorem, selected sum, quotient, choice, verification, or bridge coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.TanneryTheoremUp.TanneryTheoremCarrier namecert obligations`

Theorem 2094.3 (Tannery theorem dominated row handoff)**.** *Let $K = (A, L, M, W, Q, D, E, H, C, P, N)$ be an accepted* $\mathsf{TanneryTheorem}^{\uparrow}$ *carrier. Every dominated row comparison exported by the packet factors through*

$$A, L \longmapsto M \longmapsto W, D \longmapsto Q \longmapsto E, H, C, P, N.$$

Thus the majorant route and dyadic tolerance ledger determine the finite handoff from pointwise row convergence to the endpoint-facing series read, not a host dominated convergence theorem, selected infinite sum, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an endpoint-facing dominated row comparison. The carrier has no direct coordinate for exchanging a limit with an infinite sum, so the comparison first reads the displayed row data A and the pointwise limit data L.

The majorant route is then forced. The row M displays the absolute majorant and summable handoff, while W and D provide the finite window cutoff, tolerance split, head comparison, and tail budget.

The row Q reads back the same data as regular rational observations. Only after this readback can E seal the endpoint-facing real row, and H, C, P, N transport, replay, source, and name the route.

Since the carrier offers no host exchange theorem, selected sum, quotient equality, choice object, or ambient completeness coordinate, every dominated row comparison is exhausted by the displayed factorization. □

Lean4 ✓ `BEDC.Derived.TanneryTheoremUp.TanneryTheoremCarrier dominated row handoff`

Theorem 2094.4 (Tannery theorem real-series exchange boundary)**.** *For every accepted* $\mathsf{TanneryTheorem}^{\uparrow}$ *carrier, any* $\mathsf{RealSeries}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the exchange endpoint must read the double-indexed row, pointwise limit row, dominating majorant route, finite stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, endpoint seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row of Definition 2094.1. The exchange boundary exports no host infinite sum, no unrecorded interchange of limit and summation, no quotient stream equality, no countable choice, no ambient dominated convergence theorem, no Lean verification, and no standard bridge.*

Proof. Let a series or real endpoint consumer read the seal row E. By the carrier definition, E is downstream of the row data A, L, the majorant route M, the finite windows W, the readback Q, and the dyadic ledger D.

Apply Definition 2094.2. Pointwise-window exactness binds the finite head comparisons, dominated-row exactness binds the majorant route, and finite-exchange accounting binds the tolerance split before the endpoint seal can be read.

Apply Definition 2094.3. Every dominated comparison reaches the endpoint only through the displayed route from A, L to E, H, C, P, N.

Therefore the consumer receives exactly the finite dominated-exchange packet. It cannot read a host infinite sum, unrecorded interchange theorem, quotient stream equality, countable choice, ambient dominated convergence theorem, Lean verification, or standard bridge from the seal. □

Lean4 ✓ `BEDC.Derived.TanneryTheoremUp.TanneryTheoremCarrier real series exchange boundary`

Falsifiable prediction: Every $\mathsf{TanneryTheorem}^{\uparrow}$ endpoint-facing read factors through a double-indexed real row, pointwise limit row, dominating majorant route, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, absolute-convergence, weierstrassmtest, regseqrat, streamname, dyadicratcore, real: $\mathsf{TanneryTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries double-indexed row data, pointwise limit data, a dominating majorant route, finite exchange windows, regular rational readback, dyadic tolerance accounting, endpoint sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TanneryTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TanneryTheorem}^{\uparrow}$ packet is generated from a double-indexed real row, a pointwise limit row, a dominating majorant route, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp endpoint seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TanneryTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2094.1, Definition 2094.2, Definition 2094.3, and Definition 2094.4 seed the finite dominated-exchange route over the Real and Completion spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TanneryTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, unrecorded interchange of limit and summation, quotient stream equality, countable choice, ambient dominated convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise-window exactness, dominated-row exactness, finite-exchange accounting, RegSeqRat readback, Real seal nonescape, and structural transport for BEDC.Derived.TanneryTheoremUp

Chapter 2095

A Concrete Naming Certificate for $\mathsf{SmythCompletion}^{\uparrow}$

Following Smyth's domain-theoretic completion of quasi-uniform spaces and the constructive completion tradition of Bishop–Bridges, the $\mathsf{SmythCompletion}^{\uparrow}$ packet names the finite asymmetric completion route that reads entourage observations before passing to rounded Cauchy data. It sits on the L10 Real and Completion frontier beside chapter 339, chapter 333, chapter 589, chapter 1506, chapter 1510, and chapter 1883. The packet exports constructive completion evidence without host quotients, selected representatives, `Classical.choice`, `Quot.sound`, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2095.1 (Smyth completion carrier)**.** *A* $\mathsf{SmythCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, Q, F, R, D, M, B, I, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage surface restricted to the asymmetric observations used by the quasi-uniform request.*
2. *Q is the quasi-uniform direction row, recording finite forward-entourage, refinement, and composition observations rather than a symmetric metric equality.*
3. *F is the rounded* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *refinement ledger accepted over Q.*
4. *R is the rounded-refinement row, naming the finite cofinal requests that make the filter or net progressively tighter.*
5. *D is the dyadic and regular readback row through which* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observations reach the terminal* $\mathsf{Real}^{\uparrow}$ *boundary when a real-valued consumer is present.*
6. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row used only when a symmetric metric completion consumer asks for the associated finite route.*
7. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *formal-ball row exposing radius-tagged approximation requests.*
8. *I is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *rounded-ideal row naming the ideal-style closure of the same formal-ball route.*

9. *H records componentwise* hsame *transport for the entourage, quasi-uniform direction, Cauchy refinement, rounded-refinement, readback, metric-completion, formal-ball, rounded-ideal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from quasi-uniform entourage observations through rounded Cauchy refinements, formal balls, rounded ideals, and optional Real-facing readback.*

11. *P is the* Pkg *provenance over* $\mathsf{SmythCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SmythCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for a quotient completion object, a chosen representative, a host equivalence class, a hidden choice functional, `Classical.choice`, `Quot.sound`, *or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 (definition) `BEDC.Derived.SmythCompletionUp.SmythCompletionCarrier`
Lean4 ✓ `BEDC.Derived.SmythCompletionUp.SmythCompletionTasteGate single carrier alignment`

Theorem 2095.2 (Smyth completion NameCert obligations)**.** *For an accepted* $\mathsf{SmythCompletion}^{\uparrow}$ *carrier*

$$S = (U, Q, F, R, D, M, B, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SmythCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed uniform-space entourage row, quasi-uniform direction row, rounded Cauchy-filter or Cauchy-net ledger, rounded-refinement row, dyadic and regular readback row, metric-completion comparison row, formal-ball row, rounded-ideal row, transport, replay, provenance, and local naming row;*

2. *asymmetric entourage exposure, requiring every completion request to pass through U and Q before any rounded Cauchy row is read;*

3. *rounded Cauchy refinement, requiring F and R to display finite cofinal refinements and not a quotient of Cauchy filters or nets;*

4. *formal-ball and rounded-ideal factorization, requiring B and I to carry radius-tagged approximation and rounded ideal closure before a completion-facing consumer receives the packet;*

5. *metric and Real handoff discipline, requiring M and D to be optional downstream readbacks from the same finite route rather than sources of completeness;*

6. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed rows.*

No NameCert obligation is stored outside these rows, and none exports host quotient completion, choice of representatives, `Classical.choice`, `Quot.sound`, *or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2095.1. The accepted packet exposes exactly the uniform-space, quasi-uniform, Cauchy-refinement, rounded-refinement, readback, metric-completion, formal-ball, rounded-ideal, and structural rows.

Asymmetric entourage exposure is forced by the order of U and Q. A completion request first reads finite forward-entourage, refinement, and composition observations; it does not begin from a symmetric metric quotient or from a completed host object.

Rounded Cauchy refinement is then read from F and R. These rows display the finite cofinal filter or net refinements that make the request progressively tighter, so they provide a constructive route rather than a quotient of Cauchy data.

The formal-ball and rounded-ideal rows give the completion interface. The row B supplies radius-tagged approximation requests, and I records the rounded ideal closure of the same route before any completion-facing consumer receives it.

Metric and Real handoffs are downstream checks. The row M compares with $\mathsf{MetricCompletion}^{\uparrow}$ only after the Smyth route has been displayed, while D exposes $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readbacks without asserting ambient Real completeness.

Finally H, C, P, N add no analytic principle. They transport, replay, source, and name the displayed route, so host quotients, selected representatives, `Classical.choice`, and `Quot.sound` have no row from which a consumer could read them. □

Theorem 2095.3 (Smyth completion finite route)**.** *Let*

$$S = (U, Q, F, R, D, M, B, I, H, C, P, N)$$

be an accepted $\mathsf{SmythCompletion}^{\uparrow}$ *carrier. Every completion-facing consumer of* S *receives the Smyth completion only through the finite route*

$$U, Q \longmapsto F, R \longmapsto B, I \longmapsto M, D \longmapsto H, C, P, N.$$

Thus the packet exports a constructive asymmetric completion route from quasi-uniform entourage observations to rounded filter or Cauchy-net refinements, formal balls, rounded ideals, optional metric-completion comparison, and optional Real-facing readback. It does not export host quotient completion equality, a choice of representatives, `Classical.choice`*,* `Quot.sound`*, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Fix a completion-facing consumer of the packet. Projection gives U and Q, so the consumer starts from finite quasi-uniform entourage observations and their directed refinement ledger.

The rounded Cauchy stage reads F and R. Whether the consumer enters through $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyNet}^{\uparrow}$, it receives only the displayed cofinal refinement rows and never a quotient of Cauchy names.

The formal-ball stage reads B and I. Radius-tagged approximation requests are named by $\mathsf{FormalBall}^{\uparrow}$, and the rounded closure is named by $\mathsf{RoundedIdealCompletion}^{\uparrow}$ on the same finite route.

Metric and Real comparisons are optional downstream readbacks. A $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads M only after the rounded route is available, while any $\mathsf{Real}^{\uparrow}$-facing consumer reads D through the displayed dyadic, stream, and regular-rational observations.

Transport, replay, provenance, and local naming close the route. The rows H, C, P, N preserve the listed data componentwise, so no host quotient, representative selector, `Classical.choice`, `Quot.sound`, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{SmythCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SmythCompletion}^{\uparrow}$ packet is generated from quasi-uniform entourage observations, rounded CauchyFilter or CauchyNet refinement rows, dyadic and regular readbacks, MetricCompletion comparison rows, FormalBall approximation rows, RoundedIdealCompletion closure rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmythCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2095.1, Definition 2095.2, and Definition 2095.3 seed the constructive Smyth completion route over finite quasi-uniform and rounded Cauchy data.

Formal verification: FormalStatus(NameCert$_{\mathsf{SmythCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equivalence classes, choice principles, Real$^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, asymmetric entourage exposure, rounded Cauchy refinement, formal-ball and rounded-ideal factorization, metric and Real handoff discipline, transport stability, replay stability, and ledger non-escape for BEDC.Derived.SmythCompletionUp.

Chapter 2096

A Concrete Naming Certificate for $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the packet records refinement of a modulus-controlled real-name read, consuming chapter 885, chapter 1423, chapter 126, chapter 291, chapter 301, and chapter 38. It keeps the stronger precision request inside finite BEDC rows before any downstream Real-facing consumer reads the terminal seal.

Definition 2096.1 (Real Cauchy modulus refinement carrier)**.** *A* $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M_0, M_1, R, S_0, S_1, D_0, D_1, Q, E, H, C, P, N)$$

with the following displayed rows:

1. M_0 *is the coarse* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule already accepted by the Real-facing read.*
2. M_1 *is the refined* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule for the stronger dyadic tolerance request.*
3. R *is the* $\mathsf{CauchyModulusRefinement}^{\uparrow}$ *witness that the refined threshold extends the coarse threshold for the displayed unary precision.*
4. S_0 *is the coarse* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by* M_0*.*
5. S_1 *is the refined* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by* M_1*.*
6. D_0 *is the coarse* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger consumed by the coarse window.*
7. D_1 *is the refined* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger consumed by the refined window.*
8. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row shared by the coarse and refined windows after the refinement witness has been displayed.*
9. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* $M_0, M_1, R, S_0, S_1, D_0, D_1, Q$ *have been consumed.*
10. H *records componentwise* hsame *transport for the two schedules, the refinement witness, both windows, both dyadic ledgers, the readback, the seal, replay, provenance, and local naming rows.*
11. C *records displayed* Cont *replay from each finite precision request through*

$$M_0, M_1, R, S_0, S_1, D_0, D_1, Q \longmapsto E, H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusRefinement}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyModulusRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, open-cover compactness, or a detached Real seal.*

Definition 2096.2 (Real Cauchy modulus refinement classifier)**.** *The* $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$ *classifier compares packets only by componentwise* hsame *transport of the rows*

$$M_0, M_1, R, S_0, S_1, D_0, D_1, Q, E, H, C, P, N.$$

Thus a packet is accepted exactly when its coarse modulus, refined modulus, refinement witness, coarse and refined StreamName windows, dyadic tolerance ledgers, RegSeqRat readback, terminal Real seal, transport row, continuation replay, provenance row, and local NameCert row match a displayed carrier packet component by component. The classifier does not compare packets by quotienting streams, selecting a limit, invoking host real equality, using countable choice, using trichotomy, or importing ambient Real completeness.

Theorem 2096.3 (Real Cauchy modulus refinement NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyModulusRefinement}^{\uparrow}$ *carrier*

$$B = (M_0, M_1, R, S_0, S_1, D_0, D_1, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyModulusRefinement}^{\uparrow}}$ *surface consists exactly of carrier exposure for the displayed packet, coarse-modulus exposure through M_0, refined-modulus exposure through M_1, refinement-witness exposure through R, coarse and refined finite-window exposure through S_0, S_1, dyadic tolerance exposure through D_0, D_1, regular rational readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier tuple in Definition 2096.1. Each named coordinate is a finite row of the same BHist packet, and the last coordinate is the local naming row.

The analytic route is read in the displayed order. A finite precision request first exposes the coarse schedule M_0, the refined schedule M_1, and the witness R that connects them; it then reads the two StreamName windows S_0, S_1, the two dyadic ledgers D_0, D_1, and the shared regular readback Q before the terminal seal E is available.

The structural rows H, C, P, N transport, replay, source, and name only this route. Since the carrier contains no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient Real completeness, none of those principles can be part of the seed obligation surface. □

Lean4 ✓ `BEDC.Derived.RealCauchyModulusRefinementUp.RealCauchyModulusRefinementTasteGate single carrier alignment`

Theorem 2096.4 (Real Cauchy modulus refinement window refinement)**.** *Let*

$$B = (M_0, M_1, R, S_0, S_1, D_0, D_1, Q, E, H, C, P, N)$$

be an accepted RealCauchyModulusRefinement$^{\uparrow}$ *carrier. For every finite unary precision request, the refined read preserves the coarse Real-facing route by the finite passage*

$$M_0, M_1, R, S_0, S_1, D_0, D_1, Q \longmapsto E, H, P, N.$$

The refined window may expose more displayed StreamName data, but it cannot replace the coarse window by quotient stream equality, detach the RegSeqRat readback, select a limit, invoke host real equality, use countable choice, use trichotomy, or call ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealCauchyModulusRefinementUp.RealCauchyModulusRefinementCarrier window refinement`

Proof. Proceed by induction on the finite unary precision request.

In the base case, the request reads M_0, then M_1, and then the displayed refinement witness R. The windows S_0, S_1 and ledgers D_0, D_1 are therefore accessed as named finite rows of the same packet, not as quotient classes or host stream objects.

For the successor step, assume the preceding precision request has already followed the displayed route. The next refined request consumes the next threshold in M_1 together with the witness R, so the refined window S_1 extends the amount of visible finite data while leaving the coarse window S_0 and coarse ledger D_0 attached to the route.

The shared readback row Q is read only after both dyadic ledgers have been displayed, and the terminal seal E is read only after Q. The continuation row C, transported by H and sourced by P, N, replays the same finite route at every induction stage. Hence no selected limit, host real equality, countable-choice principle, trichotomy principle, quotient stream equality, or ambient Real completeness theorem can enter the refinement. □

Theorem 2096.5 (Real Cauchy modulus refinement Real seal nonescape)**.** *No* Real$^{\uparrow}$ *seal read accepted by a* RealCauchyModulusRefinement$^{\uparrow}$ *carrier can bypass the displayed route through the coarse modulus, refined modulus, refinement witness, coarse and refined StreamName windows, dyadic tolerance ledgers, RegSeqRat readback, and local structural rows. Equivalently, every accepted terminal seal read is attached to the packet*

$$B = (M_0, M_1, R, S_0, S_1, D_0, D_1, Q, E, H, C, P, N)$$

and cannot be separated from those rows by quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient Real completeness.

Proof. Use the obligation theorem Definition 2096.3. It exposes the seal row E only after the two modulus rows, the refinement witness, the two StreamName windows, the two dyadic ledgers, and the shared RegSeqRat readback have been displayed.

Now apply the window-refinement theorem Definition 2096.4. The induction on finite unary precision requests preserves the same route when the tolerance request is strengthened, so the refined read cannot detach the terminal seal from the coarse route it refines.

Finally inspect the carrier definition. There is no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient Real completeness, open-cover compactness, or a separate Real seal. A purported seal read that omits one of the displayed rows is therefore not a read accepted by the local classifier. □

Closure status for NameCert$_{\text{RealCauchyModulusRefinement}^{\uparrow}}$ ·

Constructive story: A RealCauchyModulusRefinement$^{\uparrow}$ packet is generated from coarse and refined CauchyModulus schedules, a refinement witness, coarse and refined StreamName windows, dyadic tolerance ledgers, a RegSeqRat readback

row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RealCauchyModulusRefinement}\uparrow}$, seed)

Scope closed: Definition 2096.1, Definition 2096.2, Definition 2096.3, Definition 2096.4, and Definition 2096.5 seed the finite Real-facing modulus-refinement route over displayed StreamName windows, dyadic tolerance ledgers, RegSeqRat readback, and the RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{RealCauchyModulusRefinement}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, componentwise classifier transport, window refinement, and Real-seal nonescape for BEDC.Derived.RealCauchyModulusRefinementUp.

Chapter 2097

A Concrete Naming Certificate for $\mathsf{RegularCauchyCategory}^{\uparrow}$

Following Bishop and Bridges' regular-Cauchy real construction and Mac Lane's categorical language, the $\mathsf{RegularCauchyCategory}^{\uparrow}$ packet records the finite route by which displayed regular-Cauchy objects and tail-compatible morphisms are admitted, identified, composed, and sealed before any completion or universal-property consumer reads them. It sits over the finite observation surface of chapter 126, the rational readback route of chapter 301, the dyadic tolerance discipline of chapter 291, the terminal real-name boundary of chapter 38, the shift route of chapter 1578, the equivalence surface of chapter 1412, and the fast-equivalence comparison of chapter 1656. Its purpose is to expose the categorical object, identity, and composition ledgers as finite BEDC data rather than as host functions, quotient streams, or ambient completeness arguments.

Definition 2097.1 (Regular-Cauchy category carrier)**.** *A* $\mathsf{RegularCauchyCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, M, I, Q, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row exposes a regular-Cauchy name only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *reads,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

2. *M is the morphism ledger. Each morphism row displays its source object, target object, finite window schedule, tolerance-preserving read route, and tail-compatibility evidence.*

3. *I is the identity ledger. For every displayed object row in O, it stores the identity morphism row whose source and target classifiers are that object row and whose reads replay the same finite windows and dyadic tolerances.*

4. *Q is the composition ledger. For every composable pair of displayed morphism rows, it stores the composite source row, middle row, target row, composed window schedule, composed tolerance ledger, and replay route.*

5. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window ledger consumed by object, morphism, identity, and composition rows.*

6. *R is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger, recording each finite rational read used by object rows and by morphism action rows.*

7. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the object tolerances, morphism tolerances, identity tolerances, and composition tolerances.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal ledger exposed only after the corresponding object or morphism route through S, R, D has been displayed.*

9. *H records componentwise* hsame *transport for object rows, morphism rows, identity rows, composition rows, window schedules, rational reads, dyadic ledgers, real seals, replay rows, provenance, and local naming rows.*

10. *C records displayed* Cont *replay for finite categorical reads through*

$$O, M, I, Q, S, R, D, E, H \longmapsto P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, M, I, Q, S, R, D, E, H, C, P, N$. It has no coordinate for a quotient stream, host function equality, selected limit, countable choice principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global category of sets, or external categorical quotient.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCategoryUp.RegularCauchyCategoryTasteGate single carrier alignment`

Theorem 2097.2 (Regular-Cauchy category NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCategory}^{\uparrow}$ *carrier*

$$K = (O, M, I, Q, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCategory}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the object ledger, morphism ledger, identity ledger, composition ledger, StreamName windows, RegSeqRat reads, DyadicRatCore tolerances, Real seals, componentwise transport, continuation replay, provenance, and local naming rows;*

2. *object classifier stability, requiring accepted object rows to be compared only by their displayed windows, rational reads, dyadic tolerances, terminal seals, and local transport;*

3. *morphism classifier stability, requiring accepted morphism rows to preserve the displayed source row, target row, finite window schedule, tolerance ledger, and replay route;*

4. *identity route, requiring every identity row in I to replay the object row it names without adding a quotient stream, selected limit, host equality proof, or ambient completeness argument;*

5. *composition route, requiring every row in Q to display its source morphism, middle object, target morphism, composed window schedule, dyadic tolerance ledger, and* Cont *replay witness;*

6. *RegSeqRat–DyadicRatCore–Real handoff, requiring every categorical read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal to pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row and the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger;*

7. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the rows listed in the carrier.*

No NameCert obligation is stored outside those rows, and none exports quotient stream equality, host function equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, a global category of sets, Lean verification, or bridge checking.*

Proof. Carrier admission is projection from Definition 2097.1. The packet exposes object, morphism, identity, composition, window, readback, tolerance, seal, transport, replay, provenance, and naming rows as finite coordinates of one BHist carrier.

Object classifier stability is read from O, S, R, D, E, H. Since an object row can be inspected only through those coordinates, the classifier has no access to an external stream quotient or a selected limit.

Morphism classifier stability is read from M together with the shared analytic ledgers S, R, D, E. A morphism row must display its source and target object rows and its finite tolerance-preserving read route before it can be accepted.

The identity route is the projection to I. Each identity row is attached to a displayed object row and replays that row's windows, rational reads, dyadic tolerances, and seal, so it cannot introduce a host equality proof or a separate completeness theorem.

The composition route is the projection to Q. A composable pair is accepted only when the two morphism rows, middle object row, composed finite windows, dyadic tolerance ledger, and continuation replay have all been displayed.

The handoff row is S, R, D, E. Any read that reaches a $\mathsf{Real}^{\uparrow}$ seal must first pass through the StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance ledger stored in the same packet.

Finally H, C, P, N are structural rows. They transport, replay, source, and name only the displayed carrier coordinates, leaving no row from which a consumer could extract quotient stream equality, host function equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge data. □

Theorem 2097.3 (Regular-Cauchy category composition route)**.** *Let*

$$K = (O, M, I, Q, S, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyCategory}^{\uparrow}$ *carrier. Every composite morphism read accepted by* K *factors through the displayed source row, middle row, target row, finite window schedule, dyadic tolerance ledger, and* Cont *replay route recorded in* Q, S, R, D, E, H, C*. Consequently the composite exports no quotient stream, host function equality, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted composite morphism read. By the composition obligation in Definition 2097.2, the read cannot be accepted until the two input morphism rows, their shared middle object row, and the resulting source and target object rows have been displayed.

The finite window schedule is then read from S through the composition ledger Q. The schedule records exactly the windows needed by the source morphism, the target morphism, and the composite read; it is not a function space or an infinite selector.

The rational and dyadic data are read next. The row R supplies the regular rational observations, and D supplies the tolerance ledger used by both component morphisms and by the composed route.

The seal-facing step passes through E. A terminal $\mathsf{Real}^{\uparrow}$ seal is visible only after the source, middle, target, window, readback, and tolerance rows have been displayed, so the seal cannot detach from the finite categorical route.

Finally the continuation row C, transported by H and sourced by P, N, replays the displayed composite. Since the carrier contains no coordinate for quotient streams, host function equality, selected limits, countable choice, or ambient Real completeness, none of those principles can be exported by the composite read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCategory}^{\uparrow}$ packet is generated from object rows, morphism rows, identity rows, composition rows, StreamName windows, RegSeqRat reads, DyadicRatCore tolerance ledgers, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2097.1, Definition 2097.2, and Definition 2097.3 seed the finite regular-Cauchy categorical route over displayed object, morphism, identity, composition, StreamName, RegSeqRat, DyadicRatCore, Real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host function equality, selected limits, countable choice, ambient RealUp completeness, a global category of sets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, object classifier transport, morphism classifier transport, identity route, composition route, RegSeqRat-DyadicRatCore-Real handoff, and ledger non-escape for BEDC.Derived.RegularCauchyCategoryUp.

Chapter 2098

A Concrete Naming Certificate for $\mathsf{Goursat}^{\uparrow}$

Following Goursat and constructive complex analysis, the $\mathsf{Goursat}^{\uparrow}$ packet names the finite triangle-subdivision boundary used by the triangle form of the Cauchy integral theorem. It consumes holomorphic disk data from chapter 68, complex analytic bounds from chapter 76, complex limit moduli from chapter 65, complex coordinate histories from chapter 39, and the contour ledger rows of chapter 521. The packet is a finite certificate for one displayed holomorphic triangle and its subdivision ledger; it is not a host path integral theorem, not open-cover compactness, not a selected primitive, and not a quotient equality principle.

Definition 2098.1 (Goursat carrier)**.** *A* $\mathsf{Goursat}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, H, E, S, M, I, C, P, N)$$

with the following displayed rows.

1. *T is the triangle-vertex row. It records three* $\mathsf{Complex}^{\uparrow}$ *vertex histories, their oriented edge histories, and the nondegenerate triangle window inspected by later rows.*
2. *H is the holomorphic disk witness. It is consumed from* $\mathsf{Holomorphic}^{\uparrow}$ *and displays a disk containing the triangle window together with the local modulus and complex analytic bounds needed for the subdivision estimate.*
3. *E is the edge-contour row. It lists the three oriented boundary contour reads and routes each read through the finite contour-integral operation surface.*
4. *S is the subdivision tree. Each node stores a displayed triangle, its four child triangles, the inherited orientation rows, and the parent-to-child boundary ledger.*
5. *M is the modulus and mesh budget. It records the unary refinement level, diameter estimates, perimeter estimates, and the holomorphic error bound used at the leaves.*
6. *I is the integral-cancellation ledger. It records boundary-edge persistence and opposite-edge cancellation for every internal edge of the subdivision tree.*
7. *C records displayed* Cont *replay from the triangle request through the holomorphic disk row, edge-contour rows, subdivision tree, mesh budget, cancellation ledger, provenance, and local naming row.*

8. *P is the* Pkg *provenance over* $\mathsf{Goursat}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{CplxAnalytic}^{\uparrow}$, $\mathsf{CplxLim}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{ContourIntegralOperation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, H, E, S, M, I, C, P, N$. It has no coordinate for a host path integral, open-cover compactness theorem, selected primitive, quotient equality of paths, or classical-choice coordinate.

Lean4 ✓ `BEDC.Derived.GoursatUp.TasteGate.GoursatTasteGate single carrier alignment`

Theorem 2098.2 (Goursat NameCert obligations)**.** *For every accepted* $\mathsf{Goursat}^{\uparrow}$ *carrier*

$$G = (T, H, E, S, M, I, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$ *surface consists exactly of the following obligation rows.*

1. *carrier exposure for the displayed triangle vertices, holomorphic disk witness, edge-contour reads, subdivision tree, mesh budget, cancellation ledger, continuation replay, provenance, and local naming rows;*

2. *classifier transport through the componentwise rows listed in Definition 2098.1;*

3. *subdivision stability, requiring every child triangle in S to inherit its vertex, orientation, holomorphic-disk, and mesh rows from the parent packet;*

4. *boundary-edge cancellation, requiring every internal edge created by the subdivision to appear twice in I with opposite orientations and the same contour-integral read;*

5. *ledger nonescape, excluding host path integrals, open-cover compactness, selected primitives, quotient path equality, and classical-choice coordinates from the local certificate surface.*

No local obligation is stored outside the displayed packet.

Proof. Carrier exposure is projection from Definition 2098.1. The accepted packet displays exactly the triangle, holomorphic disk, contour-edge, subdivision, mesh, cancellation, replay, provenance, and naming rows.

Classifier transport is componentwise. Two accepted packets compare first through the $\mathsf{Complex}^{\uparrow}$ vertex histories in T, then through the holomorphic disk row H, the edge-contour row E, the subdivision tree S, and the mesh and cancellation ledgers M, I. The structural rows C, P, N replay and name that same finite comparison.

Subdivision stability is inherited from the rows stored at each node of S. A child triangle has no independent source of vertices, orientations, disk witnesses, or mesh budgets; each of those entries is read from its parent row and transported through the same hsame comparison.

Boundary-edge cancellation is a ledger property. An internal edge is created only as a shared edge of two sibling triangles, and I records the two appearances with opposite orientations and the same contour-integral row from E. External edges are the original boundary rows and are not cancelled.

Coordinate exhaustion gives ledger nonescape. Since the carrier has no slot for a host path integral, open-cover compactness theorem, selected primitive, quotient path equality, or classical-choice coordinate, none of those objects can be exported by the local $\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$ surface. □

Lean4 ✓ `BEDC.Derived.GoursatUp.GoursatCarrier namecert obligations`

Theorem 2098.3 (Goursat triangle integral boundary)**.** *Let $G = (T, H, E, S, M, I, C, P, N)$ be an accepted* $\mathsf{Goursat}^{\uparrow}$ *carrier. Every accepted triangle-integral read factors through the displayed subdivision tree S and the opposite-edge cancellation ledger I. Its observable route is*

$$T, H, E \longmapsto S \longmapsto M \longmapsto I \longmapsto C, P, N.$$

The read exposes only boundary contour rows, child-triangle rows, leaf mesh estimates, and opposite-edge cancellations. It exposes no host path integral, open-cover compactness argument, selected primitive, quotient equality of paths, or classical-choice coordinate.

Proof. Start with an accepted triangle-integral read. By Definition 2098.1, the read enters through the displayed triangle row T, the holomorphic disk row H, and the edge-contour row E.

The route then passes through S. Each subdivision node records its four children and its inherited orientations, so every internal edge is visible as a row shared by two sibling child triangles.

The mesh budget M supplies the leaf estimates used by the holomorphic disk witness. These estimates are attached to the displayed leaves; they do not select a primitive and do not invoke compactness outside the finite subdivision tree.

The cancellation ledger I removes exactly the internal opposite-edge pairs. The remaining contour reads are the original boundary edges from E, and the structural rows C, P, N replay that finite route. Because no excluded object is a carrier coordinate, the factorization is exhaustive. □

Lean4 ✓ `BEDC.Derived.GoursatUp.GoursatCarrier triangle integral boundary`

Theorem 2098.4 (Goursat holomorphic handoff)**.** *Every* $\mathsf{Holomorphic}^{\uparrow}$ *disk packet consumed by an accepted* $\mathsf{Goursat}^{\uparrow}$ *carrier reaches only the finite triangle-subdivision integral boundary before any downstream Cauchy-integral theorem may use it. The handoff order is*

$$\mathsf{Holomorphic}^{\uparrow},\ \mathsf{CplxAnalytic}^{\uparrow},\ \mathsf{CplxLim}^{\uparrow},\ \mathsf{Complex}^{\uparrow} \longmapsto T, H, E, S, M, I \longmapsto \mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}.$$

Thus a downstream Cauchy-integral consumer may inspect the Goursat boundary only as a finite NameCert row over the displayed triangle, subdivision, mesh, and cancellation data.

Proof. Project the consumed holomorphic disk row H from the carrier. Its domain, modulus, and analytic-bound data are read through the holomorphic, complex analytic, complex limit, and complex coordinate chapters cited above.

The disk row is then restricted by the triangle row T. No downstream consumer can read H directly as a global analytic theorem, because the continuation row first routes it through the edge-contour data E and the subdivision tree S.

The mesh budget M and cancellation ledger I are the last non-structural rows before the local name. They expose finite estimates and opposite-edge cancellation only. The local $\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$ row therefore presents a finite triangle-subdivision boundary, and any later Cauchy-integral theorem must consume that boundary rather than a host path integral or selected primitive. □

Closure status for $\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$, seed)

Scope closed: Definition 2098.1, Definition 2098.2, Definition 2098.3, and Definition 2098.4 seed the finite holomorphic triangle-subdivision integral boundary over contour rows and cancellation ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Goursat}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the full Cauchy integral theorem, Morera theorem, analytic continuation, bridge checking, selected primitives, host contour integrals, open-cover compactness, quotient path equality, classical choice, or Lean verification.

Upgrade path: Obligation closure requires checked carrier exposure, classifier transport, subdivision stability, boundary-edge cancellation, ledger nonescape, triangle integral factorization, and holomorphic handoff for BEDC.Derived.GoursatUp.

Constructive story: A Goursat$^{\uparrow}$ packet is generated from triangle vertex ComplexUp data, a HolomorphicUp disk witness, edge contour rows, a subdivision tree, a modulus and mesh budget, an integral-cancellation ledger, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2099

A Concrete Naming Certificate for $\mathsf{EudoxusReal}^{\uparrow}$

Following Eudoxus and modern almost-homomorphism presentations of the real numbers, the $\mathsf{EudoxusReal}^{\uparrow}$ packet records a real name as an integer-valued line with bounded additive defect. It sits after the integer certificate of chapter 31, the rational-bound surface of chapter 430, the dyadic tolerance ledger of chapter 291, the regular rational sequence surface of chapter 301, and the terminal real-name seal of chapter 38. The chapter keeps the almost-additive function, its defect bound, and its readback windows as displayed finite rows; it is not a quotient construction of reals and does not select limits from host functions.

Definition 2099.1 (Eudoxus real carrier)**.** *An* $\mathsf{EudoxusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, A, B, D, S, Q, R, H, C, P, N)$$

with the following displayed rows.

1. *I is the* $\mathsf{Int}^{\uparrow}$ *source row naming the integer inputs and integer outputs inspected by the packet.*
2. *A is the almost-homomorphism graph row. It records the finite table of inspected pairs together with their displayed additivity comparisons.*
3. *B is the bounded-defect row. It records one integer defect budget and the checked inequalities for every inspected pair in A.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating defect and denominator windows into dyadic error bounds.*
5. *S is the finite window schedule for denominator requests and comparison requests.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed integer graph, defect budget, dyadic ledger, and window schedule.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, A, B, D, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for the integer graph, bounded-defect row, dyadic ledger, schedule, regular rational readback, Real seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each denominator or comparison request through $I, A, B, D, S, Q, R, H, P, N$*

10. *P is the* Pkg *provenance over* $\mathsf{EudoxusReal}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EudoxusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, A, B, D, S, Q, R, H, C, P, N$. It has no coordinate for quotient equality of almost homomorphisms, host functions $\mathsf{Int}^{\uparrow} \to \mathsf{Int}^{\uparrow}$, selected limits, Cauchy-completeness proofs for $\mathsf{Real}^{\uparrow}$, or an ambient ordered-field construction.

Lean4 ✓ `BEDC.Derived.EudoxusRealUp.EudoxusRealTasteGate single carrier alignment`

Theorem 2099.2 (Eudoxus real NameCert obligations)**.** *For every accepted* $\mathsf{EudoxusReal}^{\uparrow}$ *carrier*

$$E = (I, A, B, D, S, Q, R, H, C, P, N),$$

the seed $\mathsf{NameCert}_{\mathsf{EudoxusReal}^{\uparrow}}$ *surface consists exactly of carrier exposure for the displayed integer graph, bounded-defect budget, dyadic tolerance ledger, finite request schedule, regular rational readback, terminal Real seal, componentwise transport, continuation replay, provenance, and local naming rows. Each accepted request is classified only by the inspected integer rows, the bounded-defect inequalities, the dyadic tolerance rows, and the resulting regular rational readback.*

Proof. Project the carrier tuple from Definition 2099.1. The rows I, A, B, D, S, Q, R expose the analytic data, while H, C, P, N transport, replay, source, and name the same packet.

Read a request in dependency order. The integer graph row A is meaningful only over the displayed $\mathsf{Int}^{\uparrow}$ source I; the defect row B bounds the displayed additivity comparisons in A; the dyadic row D turns that bounded defect into finite tolerance information; and the schedule S selects the inspected denominator and comparison windows.

The regular rational row Q is then read from the same displayed rows, and the terminal $\mathsf{Real}^{\uparrow}$ seal R is reached only through Q. Since no excluded object is a coordinate of the carrier, the listed rows exhaust the seed obligation surface. □

Theorem 2099.3 (Eudoxus real bounded-defect readback)**.** *Let*

$$E = (I, A, B, D, S, Q, R, H, C, P, N)$$

be an accepted $\mathsf{EudoxusReal}^{\uparrow}$ *carrier. Every finite read of the almost-homomorphism graph that reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *factors through the bounded-defect route*

$$I, A \longmapsto B \longmapsto D \longmapsto S \longmapsto Q \longmapsto R.$$

Thus a consumer may read a regular rational approximation or a terminal Real seal only after the same integer graph and bounded-defect ledger have been displayed.

Proof. Start with an accepted read of Q or R. By Definition 2099.2, the read enters through the displayed integer source and graph rows I, A.

The bounded-defect row B is the next required row. It records the finite inequalities witnessing that the inspected graph behaves like an additive homomorphism up to the same integer budget on every displayed pair.

The dyadic ledger D converts that budget into tolerance rows, and the schedule S records which denominators and comparisons are inspected. The regular rational row Q is therefore downstream of I, A, B, D, S.

Finally the Real seal R is downstream of Q, and the structural rows H, C, P, N only preserve and name the route. Hence a read skipping the bounded-defect ledger cannot reach either the regular rational readback or the terminal Real seal inside this carrier. □

Theorem 2099.4 (Eudoxus real RegSeqRat comparison)**.** *Every accepted* $\mathsf{EudoxusReal}^{\uparrow}$ *packet determines its* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison surface from the common finite rows*

$$I, \quad A, \quad B, \quad D, \quad S.$$

If two accepted packets are componentwise hsame *on these rows, then their displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows compare through the same dyadic tolerances and denominator windows before either packet exposes its terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Use the classifier in Definition 2099.1. It compares accepted packets componentwise through the integer source, graph, defect, dyadic, and schedule rows before it compares the regular rational readback row.

The bounded-defect row fixes the same finite error budget, and the dyadic ledger translates it through the same tolerance windows. Componentwise hsame on I, A, B, D, S therefore gives the same displayed input data for the $\mathsf{RegSeqRat}^{\uparrow}$ row Q in both packets.

Only after that comparison can a consumer reach the terminal $\mathsf{Real}^{\uparrow}$ seal R. The theorem does not identify quotient classes of almost homomorphisms; it identifies the finite readback route exposed by this NameCert packet. □

Falsifiable prediction: An accepted $\mathsf{EudoxusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ without the displayed $\mathsf{Int}^{\uparrow}$ graph, bounded-defect row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and request schedule refutes the carrier surface.

Independence witness: int, archimedeanreal, dyadicratcore, regseqrat, real: no listed sibling supplies the whole eleven-row packet $E = (I, A, B, D, S, Q, R, H, C, P, N)$, because each sibling contributes only an integer, rational-bound, dyadic-tolerance, regular-rational, or Real-seal surface while this chapter binds the almost-homomorphism graph and bounded-defect route between them.

Closure status for $\mathsf{NameCert}_{\mathsf{EudoxusReal}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EudoxusReal}^{\uparrow}$ packet is generated from an $\mathsf{Int}^{\uparrow}$ source row, an almost-homomorphism graph row, a bounded-defect row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite request schedule, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EudoxusReal}^{\uparrow}}$, seed)

Scope closed: Definition 2099.1, Definition 2099.2, Definition 2099.3, and Definition 2099.4 seed the almost-homomorphism real-name boundary over $\mathsf{Int}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EudoxusReal}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.EudoxusRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, host functions, selected limits, completeness of $\mathsf{Real}^{\uparrow}$, or an ambient ordered-field construction.

Upgrade path: Scoped closure requires binding the displayed bounded-defect readback route as the public route for every accepted Eudoxus real-name consumer.

Chapter 2100

A Concrete Naming Certificate for $\mathsf{HolderInequality}^{\uparrow}$

Following Hardy–Littlewood–Polya and the finite-sum form of classical real analysis, the $\mathsf{HolderInequality}^{\uparrow}$ packet records the BEDC route by which Holder inequality reads over displayed finite real-vector and finite-series windows become available to downstream consumers. It sits after the square-form inequality route of chapter 1684, the partial-sum and tail surface of chapter 1581, and the apartness-from-zero positivity boundary of chapter 1587. The packet is a finite real-analysis inequality certificate; it is not a host real function-space theorem, a quotient stream equality, a selected infinite sum, a countable-choice principle, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 2100.1 (Holder inequality carrier)**.** *A* $\mathsf{HolderInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, W, A, B, C, D, E, F, H, T, P, N)$$

with the following displayed rows:

1. *X and Y are the two finite real-vector or finite-series windows. Each inspected coordinate is exposed through a* $\mathsf{Real}^{\uparrow}$ *endpoint seal and the finite readback rows already used by the surrounding real-analysis packets.*
2. *W is the shared finite observation window. It records the displayed index set, the term reads for X and Y, and the product-row reads used by the inequality-facing consumer.*
3. *A is the conjugate positive exponent ledger. It records the positive exponent rows p and q, the finite certificate that they are conjugate, and the positivity evidence needed to use them as Holder weights.*
4. *B is the* $\mathsf{PositiveReal}^{\uparrow}$ *boundary for every positive exponent, weight, and norm-like scalar consumed by the packet.*
5. *C is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *finite inequality route used for the square-case handoff and for the displayed two-row comparison subpacket.*
6. *D is the* $\mathsf{RealSeries}^{\uparrow}$ *finite partial-sum route. It exposes only the selected finite sums of coordinate powers, products, and comparison rows visible in W.*
7. *E is the dyadic tolerance budget. It stores the finite comparison radii, product slack, exponent-ledger tolerances, and endpoint rounding rows.*
8. *F is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal for the displayed Holder inequality read.*

9. *H records componentwise* hsame *transport for the vector or series windows, conjugate exponent ledger, positivity boundary, Cauchy–Schwarz route, finite-series route, dyadic budget, endpoint seal, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay from any finite coordinate request through W, A, B, C, D, E, F and back to the public inequality boundary.*

11. *P is the* Pkg *provenance over* $\mathsf{HolderInequality}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HolderInequality}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite Holder inequality packet. It has no coordinate for a host real function space, quotient equality of streams, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.HolderInequalityUp.HolderInequalityTasteGate single carrier alignment`

Theorem 2100.2 (Holder inequality NameCert obligations)**.** *For every accepted* $\mathsf{HolderInequality}^{\uparrow}$ *carrier $K = (X, Y, W, A, B, C, D, E, F, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HolderInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed finite windows, shared observation window, conjugate positive exponent ledger, positivity boundary, Cauchy–Schwarz route, finite-series route, dyadic tolerance budget, endpoint seal, transport, replay, provenance, and local naming rows;*

2. *classifier transport, requiring H and T to preserve the same finite windows, exponent rows, positive scalar evidence, square-case comparison route, finite partial-sum rows, tolerance ledger, endpoint seal, and local naming row componentwise;*

3. *conjugate-exponent exactness, requiring the positive exponent data in A and B to be the only source of the Holder exponent relation exported by the packet;*

4. *finite-sum coverage, requiring every product, power, norm-like scalar, and comparison read to pass through W, C, D, E, F before the certificate row is available;*

5. *ledger non-escape, excluding host real function spaces, quotient stream equality, selected infinite sums, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, hidden order principles, and bridge checking.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are the two finite windows X, Y, the shared observation row W, the exponent ledger A, the positivity boundary B, the Cauchy–Schwarz route C, the real-series route D, the dyadic budget E, and the endpoint seal F.

Carrier admission is exactly admission of those displayed rows together with H, T, P, N. The certificate has no additional coordinate from which a host real function-space object, quotient representative, infinite sum, countable-choice selector, or ambient completeness witness could be recovered.

Classifier transport is componentwise. The hsame rows in H keep the windows, exponent rows, positivity evidence, square-case route, finite-series rows, tolerance budget, and endpoint seal aligned, while T replays the same displayed finite route for later consumers.

Conjugate-exponent exactness follows because A is the only exponent-ledger coordinate and B is the only positive-real boundary coordinate available to it. Thus the exported exponent relation

is the displayed positive finite ledger, not a host algebraic theorem imported from outside the packet.

Finite-sum coverage is forced by the carrier order. Every product, power, norm-like scalar, and comparison read must pass through the finite windows, the square-case and series routes, the dyadic tolerance budget, and the endpoint seal before N can name the packet.

The non-escape clause is immediate from the carrier surface. None of host real function spaces, quotient stream equality, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, hidden order principles, or bridge checking is a coordinate of K. □

Theorem 2100.3 (Holder inequality finite-sum route)**.** *For every accepted* $\mathsf{HolderInequality}^{\uparrow}$ *carrier* $K = (X, Y, W, A, B, C, D, E, F, H, T, P, N)$*, every inequality-facing read factors through the displayed finite route*

$$X, Y, W \longmapsto A, B \longmapsto C, D \longmapsto E \longmapsto F \longmapsto H, T, P, N.$$

Consequently the read uses only finite vector or series windows, conjugate positive exponent ledgers, $\mathsf{PositiveReal}^{\uparrow}$ *positivity rows, the square-case comparison route supplied by* $\mathsf{CauchySchwarzReal}^{\uparrow}$*, the partial-sum route supplied by* $\mathsf{RealSeries}^{\uparrow}$*, a dyadic tolerance budget, and a terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal. It supplies no host real function space, quotient stream equality, selected infinite sum, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-checked standard theorem.*

Proof. Start from an accepted carrier. The only displayed coordinate data are X, Y, W, so every product, power, and scalar comparison request begins in the finite windows.

The exponent information is then fixed by A and B. The packet can use the conjugate positive exponent relation only through the displayed ledger and its positive-real boundary.

The inequality route is fixed by C, D, E, F. The square-case comparison is read through the $\mathsf{CauchySchwarzReal}^{\uparrow}$ row, the finite partial sums are read through the $\mathsf{RealSeries}^{\uparrow}$ row, the dyadic tolerance budget records the comparison slack, and the endpoint row seals the resulting $\mathsf{Real}^{\uparrow}$ inequality read.

The rows H, T, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for a host real function space, quotient stream equality, selected infinite sum, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking, every inequality-facing consumer receives only the displayed finite-sum route. □

Falsifiable prediction: Every $\mathsf{HolderInequality}^{\uparrow}$ inequality-facing read factors through finite vector or series windows, a conjugate positive exponent ledger, PositiveRealUp positivity rows, CauchySchwarzRealUp square-case routing, RealSeriesUp partial-sum routing, a dyadic tolerance budget, a RealUp endpoint seal, transport, replay, provenance, and local naming.

Independence witness: cauchy-schwarz-real, realseries, positivereal: $\mathsf{HolderInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite windows, conjugate exponent evidence, positive-real rows, square-case comparison routing, finite partial-sum routing, dyadic tolerance accounting, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HolderInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HolderInequality}^{\uparrow}$ packet is generated from two finite real-vector or finite-series windows, a shared observation window, a conjugate positive exponent ledger, PositiveRealUp positivity rows, CauchySchwarzRealUp and RealSeriesUp finite routes, a dyadic tolerance budget, a RealUp endpoint seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HolderInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2100.1, Definition 2100.2, and Definition 2100.3 seed the finite Holder inequality boundary for displayed RealUp finite-vector and finite-series windows.

Formal verification: FormalStatus(NameCert$_{\text{HolderInequality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real function spaces, quotient stream equality, selected infinite sums, countable choice, ambient Real$^{\uparrow}$ completeness, hidden order principles, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, conjugate-exponent exactness, finite-sum coverage, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.HolderInequalityUp.

Chapter 2101

A Concrete Naming Certificate for $\mathsf{CompactMetricSubspace}^{\uparrow}$

Following Bishop–Bridges compact metric practice and the Kelley subspace reading of finite metric data, the $\mathsf{CompactMetricSubspace}^{\uparrow}$ packet records a compact subspace of a compact metric packet by finite NameCert rows. It sits over the subspace metric interface of chapter 1519, the finite-net compact metric surface of chapter 129, the total-boundedness rows of chapter 127, the complete metric limit rows of chapter 128, and the stream and real readback interfaces of chapter 126, chapter 301, and chapter 38. The packet is a standard compact-subspace certificate for later compact-continuity and completion arguments; it is not an open-cover compactness theorem, a nearest-point selector, a quotient subset equality principle, a trichotomy principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2101.1 (CompactMetricSubspace carrier)**.** *A* $\mathsf{CompactMetricSubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, K, T, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission packet. It records the parent metric row, the displayed admission row, the inherited distance ledger, the restricted replay path, and the local subspace name.*
2. *K is the ambient* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net packet. It supplies only finite net data, mesh budgets, coverage witnesses, and compact-metric admission rows for the parent metric source.*
3. *T records* $\mathsf{TotallyBounded}^{\uparrow}$ *witnesses restricted to admitted points. Each net point exposed to the subspace consumer is checked through the admission row of S before it is used as a covering center.*
4. *L records* $\mathsf{CompleteMetric}^{\uparrow}$ *limit closure for regular Cauchy streams whose displayed observations remain admitted by S and whose parent limit row is admitted by the same subspace predicate.*
5. *W records* $\mathsf{StreamName}^{\uparrow}$ *windows for the subspace Cauchy streams consumed by L, including the finite late-distance windows used by total boundedness and completeness consumers.*
6. *R records* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks of the rational mesh, modulus, and late-distance budgets used by K, T, L, W.*
7. *E records* $\mathsf{Real}^{\uparrow}$ *seals for the displayed distance comparisons and limit-read confirmations that are already routed through S, K, T, L, W, R.*

8. *H records componentwise* hsame *transport for the subspace admission packet, compact metric packet, total-boundedness witnesses, completeness closure row, stream windows, regular rational readbacks, real seals, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$S, K, T, L, W, R, E \longmapsto H, C, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactMetricSubspace}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricSubspace}^{\uparrow}}$ *row.*

The classifier compares packets only through $S, K, T, L, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, choice-selected nearest points, quotient subset equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactMetricSubspaceUp.TasteGate.CompactMetricSubspaceTasteGate single carrier alignment`

Theorem 2101.2 (CompactMetricSubspace NameCert obligations)**.** *For every accepted* $\mathsf{CompactMetricSubspace}^{\uparrow}$ *carrier*

$$Q = (S, K, T, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricSubspace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric subspace packet, ambient compact metric packet, restricted total-boundedness witnesses, complete metric limit-closure row, stream windows, regular rational readbacks, real seals, transport, replay, provenance, and local naming rows;*

2. *subspace admission stability, requiring every compact-net center, stream observation, candidate limit, rational mesh read, and real seal to pass through the* $\mathsf{MetricSubspace}^{\uparrow}$ *admission row before it is exported;*

3. *finite-net restriction, requiring every* $\mathsf{CompactMetric}^{\uparrow}$ *net read used by the subspace packet to be accompanied by admitted centers and the inherited distance route of the metric subspace packet;*

4. *total-bounded and complete-metric compatibility, requiring the* $\mathsf{TotallyBounded}^{\uparrow}$ *witnesses and* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-closure row to name the same restricted distance ledger and the same admitted stream windows;*

5. *classifier transport, requiring* hsame *transport in H to preserve the admission row, finite-net rows, total-boundedness witnesses, completion closure, stream windows, rational readbacks, real seals, replay, provenance, and local name;*

6. *ledger exhaustion, requiring C, P, N to replay, source, and name only the displayed finite route and to export no open-cover compactness, choice-selected nearest point, quotient subset equality, trichotomy principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No CompactMetricSubspace obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2101.1. The tuple displays exactly the subspace packet, ambient compact metric packet, total-boundedness witnesses, completion closure row, stream windows, rational readbacks, real seals, transport, replay, provenance, and local naming rows.

Subspace admission stability is forced by the position of S. A compact-net center, stream observation, candidate limit, rational budget, or real seal is visible to the subspace consumer only after the parent point classifier and the admission row recorded by S have both been replayed.

Finite-net restriction uses K together with T. The ambient $\mathsf{CompactMetric}^{\uparrow}$ row supplies finite nets, while the restricted total-boundedness row exposes only those centers that survive the admission check and the inherited distance route of chapter 1519.

Compatibility of total boundedness and completeness uses L and W. The same admitted stream windows feed the late-distance reads for total boundedness and the limit-closure reads for completeness, so the packet cannot switch to a second distance ledger or a second stream name.

Transport and exhaustion are recorded by H, C, P, N. Since the carrier has no coordinate outside $S, K, T, L, W, R, E, H, C, P, N$, the excluded open-cover, nearest-point, quotient, trichotomy, and ambient real-completeness principles cannot be exported by the local name. □

Theorem 2101.3 (Inherited total-boundedness and completeness)**.** *Let $Q = (S, K, T, L, W, R, E, H, C, P, N)$ be an accepted $\mathsf{CompactMetricSubspace}^{\uparrow}$ carrier. Every subspace compactness read exported by Q factors through the finite route*

$$\mathsf{MetricSubspace}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longrightarrow H, C$$

Consequently a downstream completion or compact-continuity consumer receives total-boundedness witnesses only with admitted centers, completeness witnesses only for admitted regular streams whose limit row is admitted, and distance comparisons only through the inherited $\mathsf{MetricSubspace}^{\uparrow}$ distance ledger. The exported packet contains no open-cover compactness, choice-selected nearest-point data, quotient subset equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Proof. Project the accepted carrier and begin with a total-boundedness read. The $\mathsf{CompactMetric}^{\uparrow}$ row K supplies a finite net at the requested rational budget, and the restricted witness row T filters the displayed centers through the admission row of S before the inherited distance ledger may be read.

For a completeness read, use the $\mathsf{StreamName}^{\uparrow}$ windows in W and the rational readbacks in R to expose the regular Cauchy budgets consumed by L. The limit row is the parent $\mathsf{CompleteMetric}^{\uparrow}$ limit row only after the admission predicate of S accepts it.

The real seals in E confirm the displayed distance and limit comparisons already routed through S, K, T, L, W, R. They do not create a new completeness principle for $\mathsf{Real}^{\uparrow}$; they name the finite comparisons used by the packet.

Finally transport by H and replay by C keep the same rows under classifier comparison, while P, N source and name the route. Thus completion and compact-continuity consumers see exactly the inherited total-bounded and complete-metric data, with the listed nonconstructive and quotient principles absent from the carrier. □

Theorem 2101.4 (Real-seal nonescape)**.** *For an accepted $\mathsf{CompactMetricSubspace}^{\uparrow}$ carrier $Q = (S, K, T, L, W, R, E, H, C, P, N)$, every $\mathsf{Real}^{\uparrow}$ seal exported by E is downstream of an admitted $\mathsf{MetricSubspace}^{\uparrow}$ distance read, a $\mathsf{RegSeqRat}^{\uparrow}$ rational budget read, and the relevant $\mathsf{StreamName}^{\uparrow}$ or finite-net window. Hence a consumer cannot use Q to infer ambient $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient subset equality, a selected nearest point, or open-cover compactness.*

Proof. Read a real seal from E. By carrier admission, the seal is not a standalone real-analysis row; it is indexed by the displayed distance or limit comparison already named by S, K, T, L, W, R.

If the seal concerns a finite-net distance comparison, the route first checks the subspace admission row in S, then reads the compact metric net in K, the restricted total-bounded witness in T, and the rational budget in R.

If the seal concerns a limit comparison, the route first checks the same subspace admission row, then reads the stream window in W, the completeness closure row in L, and the rational readback in R.

The transport, replay, provenance, and naming rows H, C, P, N preserve only these finite dependencies. Since no separate $\mathsf{Real}^{\uparrow}$ completeness coordinate, trichotomy coordinate, quotient row, nearest-point selector, or open-cover compactness row appears in the carrier, none can escape through the real-seal surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricSubspace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricSubspace}^{\uparrow}$ packet is generated from a MetricSubspaceUp admission packet, CompactMetricUp finite-net data, restricted TotallyBoundedUp witnesses, CompleteMetricUp limit closure, StreamNameUp windows, RegSeqRatUp readbacks, RealUp seals, hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricSubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2101.1, Definition 2101.2, Definition 2101.3, and Definition 2101.4 seed the finite compact-subspace interface over metric subspace admission, compact finite nets, total boundedness, completeness, stream windows, rational readbacks, and real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricSubspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected nearest points, quotient subset equality, trichotomy, ambient RealUp completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, subspace admission stability, finite-net restriction, total-bounded and complete-metric compatibility, classifier transport, and ledger exhaustion for BEDC.Derived.CompactMetricSubspaceU

Chapter 2102

A Concrete Naming Certificate for $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$

Following Bishop and Bridges and Kelley completion terminology, the $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ packet records the finite naturality boundary where a displayed completion comparison map commutes with the source unit route and the completed counit route. It sits after the unit carrier of chapter 1527, the counit carrier of chapter 1539, the Bishop comparison surface of chapter 1508, the reflection witness of chapter 403, and the completion-comparison certificate of chapter 1954. Its analytic readback is the same finite route through chapter 126, chapter 291, chapter 301, and chapter 38.

The packet names the two finite naturality squares consumed by Real and Completion interfaces. It does not identify quotient completions, choose stream representatives, assert host function equality, invoke countable choice, import ambient completeness, or certify a Lean-side bridge.

Definition 2102.1 (Cauchy completion comparison naturality carrier)**.** *A* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, U, V, F, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the completion-comparison map row inherited from* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *and* $\mathsf{BishopCompletionComparison}^{\uparrow}$*, displaying the two completion presentations before either square is read.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route row for source admission into the completion-facing presentation.*
3. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *route row for returning an already completed presentation to its displayed completion consumer.*
4. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *witness row, recording that both the unit-facing and counit-facing reads are attached to the same reflection boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row shared by both sides of the comparison map.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source-side and completed-side reads inspect their observations.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared window and readback rows.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the comparison map, unit route, counit route, reflection witness, readback, window, tolerance, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$Q, U, V, F \longmapsto S, D, R \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, U, V, F, R, S, D, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionComparisonNaturalityUp.CauchyCompletionComparisonNaturalityTasteGa
single carrier alignment
```

Theorem 2102.2 (Cauchy completion comparison naturality obligations)**.** *For every accepted* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *carrier*

$$M = (Q, U, V, F, R, S, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion-comparison map, unit route, counit route, reflection witness, StreamName window, DyadicRatCore tolerance, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming row;*
2. *comparison-map exposure, requiring Q to display the two completion presentations before the unit square or counit square is inspected;*
3. *unit-square routing, requiring the source-side read to pass through U and the same Q, F, S, D, R, E rows before any Real-facing seal is public;*
4. *counit-square routing, requiring the completed-side read to pass through V and the same Q, F, S, D, R, E rows before any completion-facing consumer receives a seal;*
5. *reflection stability, requiring F to be the only witness row tying the unit and counit reads to the shared completion boundary;*
6. *structural exhaustion, requiring H, C, P, N to transport, replay, source, and name only the listed rows.*

No obligation is stored outside these rows, and none exports quotient completion equality, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Carrier admission is projection from Definition 2102.1. The accepted tuple displays exactly the completion-comparison map, unit route, counit route, reflection witness, analytic readback rows, and structural rows.

Comparison-map exposure is read from Q. Since Q occurs before the unit and counit routes are inspected, both naturality checks compare the same two completion presentations.

The unit-square obligation is read from U. A source-side consumer first enters through the completion unit and can reach the Real seal only by passing through the displayed comparison, reflection, window, tolerance, and readback rows.

The counit-square obligation is read from V. A completed-side consumer returns through the counit route and is compared against the same Q, F, S, D, R, E route rather than against a separate representative of a completion.

Reflection stability is supplied by F, while H, C, P, N add only componentwise transport, replay, provenance, and local naming. Since the carrier contains no coordinate for quotient equality, selected representatives, host equality, choice, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2102.3 (Cauchy completion comparison naturality unit square)**.** *Let*

$$M = (Q, U, V, F, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *carrier. Every source-side read that compares the completion comparison map with the unit route factors through the finite square*

$$U \longmapsto Q, F \longmapsto S, D, R \longmapsto E.$$

Thus the unit naturality surface is a displayed NameCert route over $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCom}$ $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It is not quotient completion equality, not selected representative equality, not host function equality, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Fix a source-side read. Projection of the accepted carrier gives the unit row U, so the read enters the completion-facing surface through the displayed source admission route.

The comparison then uses Q and F. The comparison row supplies the two completion presentations, and the reflection row records that the unit-facing read is attached to the same completion boundary.

The analytic side of the square is forced through S, D, R, E. The finite window, dyadic tolerance ledger, regular rational readback, and terminal Real seal are the only observation rows available after the comparison and reflection rows.

Finally H, C, P, N preserve, replay, source, and name the same route. No row in the packet can replace the square by quotient equality, selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 2102.4 (Cauchy completion comparison naturality counit square)**.** *Let*

$$M = (Q, U, V, F, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *carrier. Every completed-side read that compares the completion comparison map with the counit route factors through the finite square*

$$V \longmapsto Q, F \longmapsto S, D, R \longmapsto E.$$

Thus the counit naturality surface is a displayed NameCert route over $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It is not quotient completion equality, not selected representative equality, not host function equality, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Fix a completed-side read. Projection of the accepted carrier gives the counit row V, so the read returns from the completed presentation only through the displayed counit route.

The comparison rows are again Q and F. The completion-comparison map row supplies the same two presentations used by the unit square, and the reflection witness ties the completed-side read to the same boundary.

The terminal observation route is S, D, R, E. A completed-side consumer cannot reach the Real seal until the finite StreamName window, DyadicRatCore tolerance ledger, and RegSeqRat readback have all been displayed.

The structural rows H, C, P, N keep this counit square componentwise transported, replayable, sourced, and locally named. Since the carrier has no quotient, representative-choice, host-equality, countable-choice, ambient-completeness, Lean, or bridge coordinate, no completed-side consumer can export one from the packet. □

Theorem 2102.5 (Cauchy completion comparison naturality seal exhaustion)**.** *Let*

$$M = (Q, U, V, F, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *carrier. The unit naturality square and the counit naturality square consume the same finite route*

$$Q, U, V, F \longmapsto S, D, R \longmapsto E \longmapsto H, C, P, N.$$

No quotient completion equality, selected representative, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean bridge, or detached seal can be read outside that route.*

Proof. Carrier admission is projection from Definition 2102.1. The displayed rows are exactly $Q, U, V, F, R, S, D, E, H, C, P, N$, so any naturality consumer must enter through one of these coordinates.

Apply the obligation surface of Definition 2102.2. It requires Q to expose the comparison map, U to route the unit square, V to route the counit square, and F to tie both reads to the same completion-reflection boundary.

The two square theorems then identify the analytic tail. By Definition 2102.3, the source-side read reaches the seal only through S, D, R, E; by Definition 2102.4, the completed-side read reaches the seal through the same rows.

Finally H, C, P, N provide only componentwise transport, replay, provenance, and local naming for that finite route. Since no carrier coordinate stores quotient equality, representative selection, host function equality, countable choice, ambient completeness, a Lean bridge, or a detached seal, neither square can read such data outside the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2102.1, Definition 2102.2, Definition 2102.3, Definition 2102.4, and Definition 2102.5 seed the finite naturality squares tying the completion comparison map to the unit and counit routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-map exposure, unit-square routing, counit-square routing, reflection stability, structural exhaustion, and finite readback exactness for BEDC.Derived.CauchyCompletionComparison

Constructive story: A $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ packet is generated from a completion comparison map row, CauchyCompletionUnit route, CauchyCompletionCounit route, CompletionReflection witness, shared StreamName window, DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2103

A Concrete Naming Certificate for $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the standard Lipschitz continuity principle that a displayed bound transports Cauchy names to Cauchy names, the $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ packet records the finite route by which a regular rational Cauchy name is read through a Lipschitz map and sealed as a Real-facing regular Cauchy output. It consumes the bound-carrying map surface of chapter 410, the input regular rational route of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite transport certificate for Real and Completion consumers; it is not a quotient map construction, a selected limit, host function extensionality, differentiability data, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2103.1 (Regular Cauchy Lipschitz map carrier)**.** *A* $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}}, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row carrying the displayed graph and rational Lipschitz-bound ledger.*
2. *R_{in} is the input* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy name row.*
3. *W_{in} is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened on the input name for the requested tolerance.*
4. *W_{out} is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened on the transported output readings.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound ledger, recording the requested output radius, the bound-scaled input radius, and the comparison budgets used by the Lipschitz inequality.*
6. *R_{out} is the output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after L, R_{in}, W_{in}, W_{out}, and D have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the output regular readback row is exposed.*
8. *H records componentwise* hsame *transport for the Lipschitz map, input readback, both stream windows, dyadic ledger, output readback, Real seal, continuation replay, package provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each finite Real or Completion tolerance request through*

$$L, R_{\text{in}}, W_{\text{in}}, W_{\text{out}}, D, R_{\text{out}} \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* RegularCauchyLipschitzMap$^{\uparrow}$, LipschitzMap$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyLipschitzMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host function extensionality, selected limits, differentiability, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, or a detached output seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLipschitzMapUp.RegularCauchyLipschitzMapTasteGate single carrier alignment`

Definition 2103.2 (Regular Cauchy Lipschitz map classifier)**.** *The* RegularCauchyLipschitzMap$^{\uparrow}$ *classifier compares accepted carriers only through componentwise* hsame *transport of the rows*

$$L, R_{\text{in}}, W_{\text{in}}, W_{\text{out}}, D, R_{\text{out}}, E, H, C, P, N.$$

Thus the Lipschitz bound ledger, input regular name, finite input and output windows, dyadic tolerance conversion, output regular readback, terminal Real seal, continuation replay, provenance, and local NameCert row are all part of the visible comparison surface. The classifier does not compare host functions extensionally, quotient regular streams, select representatives, assert differentiability, or import ambient completeness.

Theorem 2103.3 (Regular Cauchy Lipschitz map NameCert obligations)**.** *For every accepted* RegularCauchyLipschitzMap$^{\uparrow}$ *carrier*

$$T = (L, R_{\text{in}}, W_{\text{in}}, W_{\text{out}}, D, R_{\text{out}}, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLipschitzMap}^{\uparrow}}$ *surface consists exactly of carrier exposure for the displayed packet, source-map exposure through L, input regular-name exposure through R_{in}, finite input-window exposure through W_{in}, finite output-window exposure through W_{out}, dyadic tolerance and bound accounting through D, output regular readback through R_{out}, terminal* Real$^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier tuple in Definition 2103.1. Each listed coordinate is a finite row of the same BHist packet, and the last coordinate is the local NameCert row.

The analytic route is read in displayed order. A tolerance request first exposes the Lipschitz map row L and its bound ledger, then the input regular Cauchy row R_{in}, then the finite windows W_{in} and W_{out}, and then the dyadic conversion ledger D before the output readback R_{out} and terminal seal E are available.

The structural rows H, C, P, N transport, replay, source, and name only this route. Since the carrier contains no coordinate for quotient stream equality, host function extensionality, selected limits, differentiability, countable choice, trichotomy, ambient Real completeness, or a detached output seal, none of those principles can be part of the seed obligation surface. □

Theorem 2103.4 (Regular Cauchy Lipschitz map Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read accepted by a* $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ *carrier factors through the finite transport route*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

Equivalently, an accepted terminal seal read may consume the displayed Lipschitz map, the input regular name, finite input and output stream windows, the dyadic tolerance and bound ledger, and the output regular readback, but it cannot bypass them by quotient stream equality, selected limits, host function extensionality, differentiability data, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the obligation theorem Definition 2103.3. It exposes the terminal seal row E only after the Lipschitz row, input regular-name row, both finite stream windows, dyadic tolerance ledger, and output regular readback have been displayed.

The classifier definition Definition 2103.2 then transports accepted reads only by the componentwise hsame row and replays them only through the displayed Cont route. Thus a Real-facing consumer cannot replace the finite transport route by a host equality of functions or streams.

Finally inspect the carrier definition. There is no coordinate for quotient stream equality, selected limits, host function extensionality, differentiability, countable choice, trichotomy, ambient Real completeness, or a second Real seal. A terminal read omitting one of the displayed transport rows is therefore not accepted by the local classifier. □

Theorem 2103.5 (Regular Cauchy Lipschitz output classifier stability)**.** *Let*

$$T = (L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}}, E, H, C, P, N)$$

and

$$T' = (L', R'_{\mathrm{in}}, W'_{\mathrm{in}}, W'_{\mathrm{out}}, D', R'_{\mathrm{out}}, E', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ *carriers. If the componentwise transport rows agree through* H *on* $L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D$*, then the output rows* R_{out} *and* R'_{out} *agree under the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier before either terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *or* E' *is read.*

Proof. Start from the classifier surface in Definition 2103.2. It compares accepted carriers through componentwise hsame transport of the displayed source rows, so agreement through H fixes the Lipschitz row, input regular name, both stream windows, and dyadic ledger.

Replay the displayed continuation row C. Since C reads output observations only after $L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D$, the same source rows force the same output regular readback route.

The Lipschitz bound ledger D supplies the bound-scaled tolerance conversion. With the same input windows and dyadic accounting, the output readings are compared by the $\mathsf{RegSeqRat}^{\uparrow}$ classifier at R_{out} and R'_{out}.

Only after that classifier agreement is visible may the terminal Real seals E and E' be exposed. Thus the stability claim concerns the output regular Cauchy readback itself, not a later quotient or selected real representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLipschitzMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ packet is generated from a LipschitzMap row, an input RegSeqRat name row, finite StreamName input and output windows, a DyadicRatCore tolerance and bound ledger, an output RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLipschitzMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2103.1, Definition 2103.2, Definition 2103.3, and Definition 2103.4 seed the finite Lipschitz transport route from displayed regular Cauchy input names through dyadic bound accounting to output regular readback and the terminal RealUp seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyLipschitzMap}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host function extensionality, differentiability, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, componentwise classifier transport, Lipschitz tolerance conversion, output regular readback, and Real-seal nonescape for BEDC.Derived.RegularCauchyLipschitzMapUp.

Chapter 2104

A Concrete Naming Certificate for $\mathsf{GromovHausdorff}^{\uparrow}$

Following Gromov's metric-geometry formulation and Hausdorff's compact-subset distance, the $\mathsf{GromovHausdorff}^{\uparrow}$ packet records compact-metric comparison by finite correspondence rows and dyadic distortion bounds. It sits over the metric source of chapter 57, the compact-metric surface of chapter 129, the compact-subset ledger of chapter 1660, the finite-distance interface of chapter 2086, and the distance-function route of chapter 7569. The packet compares two displayed compact metric endpoints; it is not a quotient compactum, a selected optimal correspondence, an open-cover compactness theorem, a host metric-space equality principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2104.1 (Gromov–Hausdorff comparison carrier)**.** *A* $\mathsf{GromovHausdorff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, K_X, K_Y, R, D, H, Q, U, T, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two compact-metric endpoint names. Each endpoint row records the displayed* $\mathsf{Met}^{\uparrow}$ *source, point-name admission data, compact finite-net schedules, and the request for which a comparison certificate is exposed.*
2. *K_X and K_Y are the endpoint* $\mathsf{CompactMetric}^{\uparrow}$ *rows consumed before any correspondence row may be inspected.*
3. *R is the finite correspondence table. Its rows list finitely many endpoint point-name pairs, the two projection-coverage ledgers, and the local inhabitedness witnesses needed for a comparison read.*
4. *D is the dyadic distortion ledger. It records finite* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *bounds comparing the two endpoint distance reads along every displayed pair of correspondence rows.*
5. *H records the* $\mathsf{HausdorffMetric}^{\uparrow}$ *and* $\mathsf{Hyperspace}^{\uparrow}$ *comparison rows used to read the two projection-coverage ledgers as compact-subset distance data rather than as host powersets.*
6. *Q records the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *handoff used when a consumer asks to view the same comparison through distance-function observations.*
7. *U records componentwise* hsame *transport for endpoints, compact-metric rows, correspondence rows, dyadic distortion bounds, Hausdorff and hyperspace comparison rows, Kuratowski handoff, replay, provenance, and local naming rows.*

8. *T records displayed transport of the finite comparison surface across accepted endpoint renamings; it carries only rows already present in $X, Y, K_X, K_Y, R, D, H, Q, U$.*

9. *C records displayed* Cont *replay of the route*

$$X, Y, K_X, K_Y \longmapsto R \longmapsto D \longmapsto H, Q \longmapsto U, T.$$

10. *P is the* Pkg *provenance over* $\mathsf{GromovHausdorff}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GromovHausdorff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K_X, K_Y, R, D, H, Q, U, T, C, P, N$. It has no coordinate for quotient compacta, selected optimal correspondences, open-cover compactness, host metric-space equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or semantic isometry of completed spaces.*

Lean4 ✓ `BEDC.Derived.GromovHausdorffUp.GromovHausdorffTasteGate single carrier alignment`

Theorem 2104.2 (GromovHausdorffUp NameCert obligations)**.** *For every accepted* $\mathsf{GromovHausdorff}^{\uparrow}$ *carrier*

$$G = (X, Y, K_X, K_Y, R, D, H, Q, U, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GromovHausdorff}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *endpoint admission for the two compact-metric names X, Y and their* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *support rows K_X, K_Y;*

2. *correspondence admission for the finite table R, requiring both endpoint projections and local inhabitedness witnesses to be displayed before any distortion read is public;*

3. *dyadic distortion exactness for D, requiring every displayed pair of correspondence rows to read both endpoint distances and the shared dyadic bound through the same finite table;*

4. *compact-subset comparison compatibility for H, requiring the projection-coverage rows to be read through the* $\mathsf{HausdorffMetric}^{\uparrow}$ *and* $\mathsf{Hyperspace}^{\uparrow}$ *surfaces;*

5. *Kuratowski observation compatibility for Q, requiring any distance-function view of the same comparison to pass through the displayed* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *handoff;*

6. *classifier transport through U, T, C, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve endpoints, correspondence rows, distortion bounds, comparison handoffs, provenance, and local name;*

7. *provenance and name exhaustion through P, N, excluding quotient compacta, selected correspondences, open-cover compactness, host metric-space equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and completed-space isometry.*

No Gromov–Hausdorff obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.GromovHausdorffUp.GromovHausdorffCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 2104.1. The endpoint rows X, Y and compact-metric support rows K_X, K_Y are the only compact metric surfaces that a consumer may inspect.

The finite correspondence table R supplies both endpoint projections and the inhabitedness witnesses. The dyadic distortion ledger D then reads every displayed pair of correspondence rows against the two endpoint distance witnesses, so no distortion statement can be formed from data outside the finite table.

The compact-subset comparison row H routes projection coverage through $\mathsf{HausdorffMetric}^{\uparrow}$ and $\mathsf{Hyperspace}^{\uparrow}$, while Q routes distance-function observations through $\mathsf{KuratowskiEmbedding}^{\uparrow}$. These rows are handoffs inside the packet, not ambient compacta or host function spaces.

Finally U, T, C preserve and replay the same route componentwise, and P, N name exactly that finite route. Since the carrier has no row for quotient compacta, selected optimal correspondences, open-cover compactness, host metric-space equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or completed-space isometry, the seven listed rows exhaust the local naming obligations. □

Theorem 2104.3 (Correspondence bridge to compact-metric comparison)**.** *Every* $\mathsf{Met}^{\uparrow}$*-,* $\mathsf{CompactMetric}^{\uparrow}$*-,* $\mathsf{HausdorffMetric}^{\uparrow}$*-,* $\mathsf{Hyperspace}^{\uparrow}$*-, or* $\mathsf{KuratowskiEmbedding}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{GromovHausdorff}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y, K_X, K_Y \longmapsto R \longmapsto D \longmapsto H, Q \longmapsto U, T, C, P, N.$$

Consequently the consumer receives only the displayed compact-metric endpoints, finite correspondence rows, dyadic distortion bounds, compact-subset comparison rows, Kuratowski observation handoff, componentwise transport, replay, provenance, and local name. It cannot derive a quotient compactum, selected optimal correspondence, open-cover compactness proof, host metric-space equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or isometry of completed spaces from the packet.*

Proof. Start with the endpoint rows. Any metric or compact-metric read reaches X, Y, K_X, K_Y first, so the comparison is anchored in displayed point names, distance witnesses, and compact finite-net schedules.

Next inspect the correspondence route. The row R is the only finite relation between the endpoints, and D is the only dyadic distortion ledger attached to it; therefore a compact-metric comparison consumer must pass through R before reading any bound.

The bridge rows then determine the available external views. A Hausdorff or hyperspace consumer reads H, and a Kuratowski-style distance-function consumer reads Q, but both rows consume the same correspondence and distortion data.

The structural rows U, T, C, P, N preserve the route under componentwise transport, replay it, source it, and name it. Because no further coordinate is present, the consumer boundary is exactly the displayed finite comparison surface and nothing beyond it. □

Theorem 2104.4 (Gromov-Hausdorff pointed basepoint forgetful route)**.** *Let a* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *packet feed a* $\mathsf{GromovHausdorff}^{\uparrow}$ *endpoint. The endpoint may be consumed only through the forgotten-basepoint rows*

$$\mathsf{Met}^{\uparrow}, \quad \mathsf{CompleteMetric}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The $\mathsf{GromovHausdorff}^{\uparrow}$ *correspondence cannot inspect, preserve, or compare the basepoint row unless a separate pointed Gromov-Hausdorff carrier supplies an explicit pointed endpoint row.*

Proof. Project the $\mathsf{GromovHausdorff}^{\uparrow}$ carrier of Definition 2104.1. Its endpoint coordinates are X, Y, K_X, K_Y, and its comparison coordinates are $R, D, H, Q, U, T, C, P, N$.

Use Definition 2104.3. Any metric-comparison consumer first reads the ordinary metric and compact-metric endpoint rows, then the correspondence and distortion rows, and finally the Hausdorff, hyperspace, Kuratowski, transport, replay, provenance, and naming rows.

A $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ source may provide such an endpoint only by forgetting its basepoint row and exposing the metric, complete-metric, regular-rational, and real-seal rows. The basepoint route belongs to chapter 4368, not to the ordinary Gromov-Hausdorff carrier.

Since no basepoint coordinate appears among $X, Y, K_X, K_Y, R, D, H, Q, U, T, C, P, N$, the correspondence cannot preserve or compare it. A pointed comparison therefore requires a separate pointed endpoint row. □

Closure status for $\mathsf{NameCert}_{\mathsf{GromovHausdorff}^{\uparrow}}$.

Constructive story: A $\mathsf{GromovHausdorff}^{\uparrow}$ packet is generated from two compact-metric endpoints, finite correspondence rows, dyadic distortion bounds, HausdorffMetricUp and HyperspaceUp comparison rows, a KuratowskiEmbeddingUp observation handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GromovHausdorff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2104.1, Definition 2104.2, and Definition 2104.3 seed the finite correspondence interface for compact-metric comparison without importing quotient, selected-correspondence, open-cover, host-equality, or ambient-completeness principles.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GromovHausdorff}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.GromovHausdorffUp.GromovHausdorffCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close quotient compacta, selected correspondences, open-cover compactness, host metric-space equality, ambient RealUp completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked endpoint admission, correspondence admission, dyadic distortion exactness, compact-subset comparison compatibility, Kuratowski observation compatibility, classifier transport, and provenance/name exhaustion for BEDC.Derived.GromovHausdorffUp.

Chapter 2105

A Concrete Naming Certificate for $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ packet records the finite route by which a 1-Lipschitz, hence nonexpansive, map transports regular Cauchy rational names without enlarging the requested tolerance. It refines the bound-carrying map surface of chapter 410 by fixing the Lipschitz ledger at the nonexpansive unit boundary, and it consumes the input regular rational route of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a precise nonexpansive transport boundary rather than reusing a broader Lipschitz packet with an arbitrary displayed bound.

Definition 2105.1 (Regular Cauchy nonexpansive map carrier)**.** *A* $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M_{\mathrm{src}}, M_{\mathrm{tgt}}, L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}}, E, H, C, P, N)$$

with the following displayed rows:

1. M_{src} *and* M_{tgt} *are the source and target metric rows on which the transport is read.*
2. L *is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row carrying the displayed graph and a bound ledger fixed to the nonexpansive unit boundary.*
3. R_{in} *is the input* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy name row.*
4. W_{in} *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened on the input name for the requested target tolerance.*
5. W_{out} *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened on the transported output readings.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the target radius, the unchanged input radius justified by the unit-bound row, and the comparison cells used by the nonexpansive inequality.*
7. R_{out} *is the output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after* L, R_{in}, W_{in}, W_{out}, *and* D *have been displayed.*
8. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the output regular readback row is exposed.*
9. H *records componentwise* hsame *transport for the metric rows, Lipschitz row, input readback, both stream windows, dyadic ledger, output readback, Real seal, continuation replay, package provenance, and local naming rows.*

10. *C records displayed* Cont *replay from each finite Real or Completion tolerance request through*

$$M_{\mathrm{src}}, M_{\mathrm{tgt}}, L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}} \longmapsto E, H, P, N.$$

11. *P is the* Pkg *provenance over* RegularCauchyNonexpansiveMap$^{\uparrow}$, LipschitzMap$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{RegularCauchyNonexpansiveMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host function extensionality, selected limits, differentiability, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, a detached Lipschitz bound greater than the unit row, or a detached output seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNonexpansiveMapUp.TasteGate.RegularCauchyNonexpansiveMapTasteGate`
`single carrier alignment`

Theorem 2105.2 (Regular Cauchy nonexpansive map NameCert obligations)**.** *For every accepted* RegularCauchyNonexpansiveMap$^{\uparrow}$ *carrier*

$$T = (M_{\mathrm{src}}, M_{\mathrm{tgt}}, L, R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}}, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyNonexpansiveMap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric, target metric, Lipschitz map, input regular name, finite input and output windows, dyadic tolerance ledger, output regular readback, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *nonexpansive bound discipline, requiring the displayed Lipschitz row L to use only the unit-bound ledger and requiring D to route a target tolerance to the same input tolerance rather than to a bound-scaled radius;*
3. *input regular-name exposure, requiring every transported read to begin with the displayed* RegSeqRat$^{\uparrow}$ *row R_{in} and the finite input window W_{in};*
4. *output regular readback, requiring R_{out} to be obtained from the displayed Lipschitz row, windows, and dyadic unit-bound comparisons;*
5. *Real-seal nonescape, requiring E to be downstream of R_{out} and to export no quotient stream equality, selected limit, host function extensionality, differentiability, countable choice, trichotomy, or ambient* Real$^{\uparrow}$ *completeness;*
6. *provenance and local naming, requiring H, C, P, N to transport, replay, source, and name only the displayed nonexpansive route.*

No RegularCauchyNonexpansiveMap$^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2105.1. The accepted packet displays exactly the metric, Lipschitz, regular-name, stream-window, dyadic, output-readback, seal, transport, replay, provenance, and naming coordinates.

The nonexpansive bound discipline is the special role of L and D. The Lipschitz row supplies the map surface, but the accepted bound ledger is fixed at the unit row, so the dyadic tolerance request is replayed without multiplication by a larger rational bound.

Input exposure and output readback are then forced by the route $R_{\mathrm{in}}, W_{\mathrm{in}}, W_{\mathrm{out}}, D, R_{\mathrm{out}}$. The output regular row is visible only after the source regular name, finite windows, and unit-bound comparisons have all been displayed.

Finally E, H, C, P, N close the route. The Real seal is downstream of the output readback, and the structural rows preserve only the displayed coordinates, leaving no carrier position from which an accepted consumer could extract the excluded objects. □

Theorem 2105.3 (Regular Cauchy nonexpansive map Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer read accepted by a* $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ *carrier factors through the finite transport route*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

Equivalently, an accepted terminal seal read may consume the displayed nonexpansive Lipschitz row, the input regular name, finite input and output stream windows, the dyadic unit-bound tolerance ledger, and the output regular readback, but it cannot bypass them by quotient stream equality, selected limits, host function extensionality, differentiability data, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the obligation theorem Definition 2105.2. It exposes the terminal seal row E only after the Lipschitz row with unit-bound ledger, the input regular-name row, both finite stream windows, the dyadic tolerance ledger, and the output regular readback have been displayed.

The displayed Cont route then replays each Real-facing request through the same rows. Since the dyadic ledger records the target tolerance as the input tolerance, the handoff uses the nonexpansive boundary rather than a detached larger Lipschitz constant.

Finally inspect the carrier definition. There is no coordinate for quotient stream equality, selected limits, host function extensionality, differentiability, countable choice, trichotomy, ambient Real completeness, or a second Real seal. A terminal read omitting one of the displayed transport rows is therefore not accepted by the local classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ packet is generated from source and target metric rows, a LipschitzMap row whose bound ledger is fixed at the nonexpansive unit boundary, an input RegSeqRat name row, finite StreamName input and output windows, a DyadicRatCore tolerance ledger, an output RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2105.1, Definition 2105.2, and Definition 2105.3 seed the finite nonexpansive transport route from displayed regular Cauchy input names through unit-bound dyadic accounting to output regular readback and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host function extensionality, differentiability, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nonexpansive bound stability, output readback, Real-seal nonescape, and BEDC.Derived.RegularCauchyNonexpansiveMapUp formalization.

Chapter 2106

A Concrete Naming Certificate for $\mathsf{DyadicValuation}^{\uparrow}$

Following Bourbaki and standard valuation theory, the $\mathsf{DyadicValuation}^{\uparrow}$ packet records the finite dyadic valuation data needed before scale normalization and tolerance comparison may be consumed by the Real and Completion route. It sits over the dyadic arithmetic carrier of chapter 291, the signed mantissa and unary exponent boundary of chapter 1865, the finite scale-domination route of chapter 1672, and the precision handoff surface of chapter 1768. The packet isolates the valuation row as finite data: downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers may read scale order and tolerance bounds without quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2106.1 (Dyadic valuation carrier)**.** *A* $\mathsf{DyadicValuation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, E, Z, V, C, T, H, R, P, N)$$

with the following displayed rows:

1. *M is the signed mantissa row, read through the* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *surface and not through a host integer normal form.*
2. *E is the unary exponent row, recording the denominator scale and successor ledger used by the valuation comparison.*
3. *Z is the zero/nonzero branch row, separating the displayed zero case from the displayed nonzero valuation route without invoking trichotomy.*
4. *V is the valuation row, recording the finite exponent count exposed after M, E, Z have been projected.*
5. *C is the common-scale replay row, transporting compared dyadic packets to the same displayed scale before valuation comparison is read.*
6. *T is the* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *handoff row, exposing the valuation scale to tolerance consumers after the common-scale replay has been displayed.*
7. *H records componentwise* hsame *transport for mantissa, exponent, zero/nonzero branch, valuation, common-scale replay, tolerance handoff, continuation, provenance, and local naming rows.*

8. *R records displayed* Cont *replay from signed dyadic inspection through exponent reading, zero/nonzero branching, valuation exposure, common-scale replay, tolerance handoff, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicValuation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{DyadicArchimedean}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicValuation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, Z, V, C, T, H, R, P, N$. It has no coordinate for quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicValuationUp.DyadicValuationTasteGate single carrier alignment`

Theorem 2106.2 (Dyadic valuation NameCert obligations)**.** *For every accepted* $\mathsf{DyadicValuation}^{\uparrow}$ *carrier $D = (M, E, Z, V, C, T, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicValuation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the signed mantissa, unary exponent, zero/nonzero branch, valuation row, common-scale replay, tolerance handoff, transport, replay, provenance, and local naming rows;*
2. *branch exactness, requiring Z to expose the displayed zero case and the displayed nonzero route before V can be consumed;*
3. *unary valuation induction, requiring V to be read by finite induction over the exponent row E and the branch row Z;*
4. *common-scale stability, requiring C to replay compared dyadic packets at one displayed scale before valuation comparison or tolerance handoff is exported;*
5. *tolerance handoff locality, requiring T to consume only the displayed valuation and common-scale rows;*
6. *classifier transport and replay, requiring H and R to preserve M, E, Z, V, C, T, P, N componentwise;*
7. *ledger non-escape, excluding quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No dyadic valuation obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic rows are M, E, Z, V, C, T, while H, R, P, N transport, replay, source, and name the same finite packet.

Branch exactness is read from Z. The zero side exposes the displayed zero row, and the nonzero side opens the route in which the signed mantissa and unary exponent rows can be inspected. The carrier has no trichotomy row, so this split is only the displayed finite branch.

Unary valuation induction is then performed over E. In the base case the exponent ledger has no successor step, so V records the current valuation row after the branch has been fixed. In the successor step the induction hypothesis is transported through the next exponent row, and the valuation count is extended by that single displayed successor.

The common-scale and tolerance clauses are projections from C and T. The row C is the only common-scale replay, so valuation comparison between packets must pass through it before the tolerance handoff T can be read.

Finally H and R preserve the displayed rows componentwise, while P and N record provenance and local naming. Since the excluded objects do not occur in the carrier, no local NameCert obligation can be stored outside $M, E, Z, V, C, T, H, R, P, N$. □

Theorem 2106.3 (Dyadic valuation common-scale stability)**.** *Let $D = (M, E, Z, V, C, T, H, R, P, N)$ be an accepted* $\mathsf{DyadicValuation}^{\uparrow}$ *carrier. Every valuation comparison exported by D factors through the finite route*

$$M, E, Z, V \longmapsto C \longmapsto H, R, P, N.$$

Thus two accepted replays with componentwise hsame *rows expose the same common-scale valuation surface, and the comparison does not use quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an exported valuation comparison. Carrier admission places the only valuation coordinate at V, and V is downstream of the signed mantissa row M, unary exponent row E, and branch row Z.

The common-scale row C is then the only replay coordinate available for comparing two dyadic packets. It transports the displayed valuation data to a shared scale before any comparison surface is exported.

For stability, read componentwise transport through H and continuation replay through R. These rows preserve M, E, Z, V, C exactly as displayed rows, while P and N only source and name the same route.

Hence an accepted replay exposes the same finite common-scale valuation surface. The conclusion is internal to the packet and has no coordinate for the excluded quotient, host, choice, or completeness principles. □

Theorem 2106.4 (Dyadic valuation tolerance handoff)**.** *Every* $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer named by an accepted* $\mathsf{DyadicValuation}^{\uparrow}$ *carrier reads the route*

$$M, E, Z, V, C \longmapsto T \longmapsto H, R, P, N.$$

Consequently the consumer receives finite valuation and common-scale data before any tolerance-scale comparison is available, with no quotient rational equality, host integer normal form, trichotomy, selected representative, countable-choice row, or $\mathsf{Real}^{\uparrow}$ *completeness witness inserted into the route.*

Proof. Project the tolerance handoff row T. It is the sole row whose purpose is to expose valuation data to $\mathsf{DyadicToleranceScale}^{\uparrow}$ and to the downstream stream, regular-rational, and real-name consumers.

The handoff is downstream of M, E, Z, V, C. Therefore a consumer first reads the signed mantissa, unary exponent, zero/nonzero branch, valuation row, and common-scale replay before it can inspect tolerance-scale data.

The route is preserved by H and replayed by R. Provenance and local naming through P, N source the same packet and do not add a second route to tolerance data.

Thus every listed consumer receives a finite valuation handoff. Since no excluded coordinate is present in the carrier, the handoff cannot import quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicValuation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicValuation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2106.1, Definition 2106.2, Definition 2106.3, and Definition 2106.4 seed the finite dyadic valuation surface from signed mantissa, unary exponent, zero/nonzero branch, valuation row, common-scale replay, tolerance handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicValuation}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DyadicValuationUp`

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, host integer normal forms, trichotomy, selected representatives, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch exactness, unary valuation induction, common-scale stability, tolerance handoff locality, classifier transport, replay, provenance, and local naming for BEDC.Derived.DyadicValuationUp.

Constructive story: A DyadicValuation$^{\uparrow}$ packet is generated from a signed mantissa row, unary exponent row, zero/nonzero branch row, valuation row, common-scale replay, DyadicToleranceScale handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2107

A Concrete Naming Certificate for $\mathsf{Subshift}^{\uparrow}$

Following Lind–Marcus symbolic dynamics and Brouwer–Bishop finite-prefix practice, $\mathsf{Subshift}^{\uparrow}$ records shift-invariant closed symbolic subspaces through finite prefix-language rows over $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$, with $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ as the finite forbidden-word specialization. The packet is a finite certificate surface for symbolic prefix languages; it is not a host infinite sequence, quotient shift space, entropy object, invariant measure, compactness principle, or ambient symbolic-dynamics theorem. Its dependency route is the Cantor prefix surface of chapter 1475, the Baire prefix surface of chapter 334, the finite stream-window route of chapter 126, and the forbidden-ledger specialization of chapter 434.

Definition 2107.1 (Subshift carrier)**.** *A* $\mathsf{Subshift}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (A, W, L, S, C, B, H, K, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet row, naming the visible symbols available to the local prefix language.*
2. *W is the finite window row, recording the inspected word lengths and prefix positions.*
3. *L is the allowed-language ledger, listing the admitted finite prefix rows over A at the windows in W.*
4. *S is the shift-restriction row, recording the finite left and right prefix restrictions that must remain in L whenever a longer accepted row is shifted or truncated.*
5. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *row used when the alphabet row is the displayed binary row.*
6. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *row used when the alphabet row is read as unary-coded countable symbols.*
7. *H records componentwise* hsame *transport for $A, W, L, S, C, B, K, P, N$.*
8. *K records the* Cont *routes from prefix requests to Cantor rows, Baire rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, language-ledger inspection, shift restriction, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{Subshift}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{SubshiftOfFiniteType}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Subshift}^{\uparrow}}$ *row for the finite prefix-language classifier.*

The classifier compares accepted packets only through the displayed prefix rows. It accepts a prefix read exactly when the corresponding finite row is present in L, every recorded shift or restriction is certified by S, and the Cantor, Baire, or stream-name access route is one of the rows named by C, B, K, P, N.

Lean4 ✓ `BEDC.Derived.SubshiftUp.SubshiftTasteGate single carrier alignment`

Theorem 2107.2 (Subshift NameCert obligations)**.** *For every accepted* $\mathsf{Subshift}^{\uparrow}$ *packet* $X = (A, W, L, S, C, B, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Subshift}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the alphabet, window, allowed-language, shift-restriction, Cantor, Baire, transport, continuation, provenance, and local naming rows;*
2. *prefix classifier stability, requiring componentwise* hsame *transport to preserve membership and nonmembership decisions in the finite language ledger L;*
3. *shift-restriction stability, requiring every displayed shift, truncation, and prefix restriction named by S to land again in the admitted ledger row;*
4. *Cantor/Baire handoff, requiring binary reads to pass through the* $\mathsf{CantorSpace}^{\uparrow}$ *row C, countable-symbol reads to pass through the* $\mathsf{BaireSpace}^{\uparrow}$ *row B, and all finite requests to pass through the* $\mathsf{StreamName}^{\uparrow}$ *route named by K;*
5. *finite-type specialization, requiring any finite forbidden-word ledger to be read as the* $\mathsf{SubshiftOfFiniteType}^{\uparrow}$ *instance whose accepted words are exactly the rows of L avoiding the displayed forbidden ledger;*
6. *non-escape, excluding host infinite sequences, quotient shift spaces, entropy, invariant measures, compactness principles, unrecorded branch choice, and ambient symbolic-dynamics theorems.*

No $\mathsf{Subshift}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the accepted carrier. The packet exposes only the rows $A, W, L, S, C, B, H, K, P, N$, so carrier admission is the assertion that those rows form the visible prefix-language certificate.

Prefix classifier stability is componentwise. Transport A, W, L through the displayed hsame rows in H; a membership decision remains available exactly when the transported prefix row is the transported ledger row. There is no second classifier coordinate that could alter the decision.

Shift-restriction stability is read from S. Each displayed shift, truncation, or prefix restriction consumes one accepted finite word row and returns the recorded shorter row in L. The continuation row K replays only those finite moves.

For the handoff clause, split by the source row. Binary prefix reads use C and therefore cite chapter 1475; countable-symbol reads use B and therefore cite chapter 334; every scheduled finite read is opened through the $\mathsf{StreamName}^{\uparrow}$ route named in K.

The finite-type specialization is obtained by taking L to be the list of finite words whose displayed subwords avoid a finite forbidden ledger. This is exactly the local surface of chapter 434, with S recording the same finite restriction checks.

Finally, non-escape follows from absence of coordinates for the excluded objects. A downstream consumer can project only the alphabet, window, language ledger, shift restriction, Cantor and Baire handoff rows, transport, continuation, provenance, and local $\mathsf{NameCert}$ row. □

Theorem 2107.3 (Subshift Cantor/Baire handoff)**.** *Every accepted* $\mathsf{Subshift}^{\uparrow}$ *read factors through the finite prefix route*

$$\mathsf{CantorSpace}^{\uparrow}, \mathsf{BaireSpace}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto A, W, L, S \longmapsto H, K, P, N$$

before any downstream symbolic-dynamics consumer can use the accepted word row. In particular, the consumer receives a displayed Cantor or Baire prefix packet and the scheduled stream-name window before it receives the subshift language decision, and no host infinite sequence, quotient shift space, entropy value, invariant measure, compactness principle, or ambient symbolic-dynamics theorem is inserted into the route.

Proof. Start with an accepted read. Carrier admission places the source access rows at C and B, the finite scheduling row at K, and the language decision rows at A, W, L, S.

If the visible alphabet is binary, the read first opens the $\mathsf{CantorSpace}^{\uparrow}$ row C. If the alphabet is countable-name data, it opens the $\mathsf{BaireSpace}^{\uparrow}$ row B. In both cases the finite request is then scheduled through the $\mathsf{StreamName}^{\uparrow}$ route recorded by K.

Only after that route is displayed can the consumer inspect A, W, L, S. The ledger L supplies the accepted prefix decision, while S supplies the shift or restriction row needed by shift-invariant use.

Transport and replay through H and K preserve the same finite rows, and P, N only source and name the packet. Since the carrier has no coordinate for the excluded objects, every downstream symbolic-dynamics consumer receives exactly the displayed finite prefix route. □

Closure status for $\mathsf{NameCert}_{\mathsf{Subshift}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Subshift}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2107.1, Definition 2107.2, and Definition 2107.3 seed the finite prefix-language surface for shift-invariant symbolic subspaces over Cantor, Baire, and StreamName rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Subshift}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sequences, quotient shift spaces, entropy, invariant measures, compactness principles, unrecorded branch choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix classifier stability, shift-restriction stability, Cantor/Baire handoff, finite-type specialization, and non-escape for BEDC.Derived.SubshiftUp.

Constructive story: A $\mathsf{Subshift}^{\uparrow}$ packet is generated from a finite alphabet row, finite window row, allowed-language ledger, shift-restriction row, Cantor and Baire handoff rows, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 2108

A Concrete Naming Certificate for $\mathsf{BisectionMethod}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BisectionMethod}^{\uparrow}$ packet records the method-level certificate by which located interval data, finite branch ledgers, retained nested windows, stream-name readback, regular rational names, and terminal real seals are exposed before any root-location or intermediate-value consumer can read a Real-facing result. It unifies the dyadic branch surface of chapter 361, the located interval route of chapter 1711, the convergence handoff of chapter 1804, the retained window discipline of chapter 320, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite method certificate for L10 Real and Completion routes; it does not select a root, complete an intersection, assert trichotomy, invoke open-cover compactness, identify quotient real names, or add a choice principle.

Definition 2108.1 (BisectionMethod carrier)**.** *A* $\mathsf{BisectionMethod}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, D, L, W, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row, exposing the endpoint names, locator reads, sign or bracket tests, and the initial tolerance request consumed by the method.*
2. *D is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, recording the finite dyadic midpoint decisions and the branch labels inspected at each visible stage.*
3. *L is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *route, transporting the displayed branch decision to the retained child interval without adding a host-order trichotomy row.*
4. *W is the retained* $\mathsf{NestedInterval}^{\uparrow}$ *window row, listing the finite interval windows that remain available to downstream consumers.*
5. *T is the finite termination and tolerance budget row, binding requested precision to a displayed bisection depth, diameter estimate, and ledger-exhaustion check.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed from W and T, with no completed infinite branch or global selector.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the retained stream windows and their dyadic tolerance witnesses.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after I, D, L, W, T, S, R have been displayed.*

9. *H records componentwise* hsame *transport for located interval source, dyadic branch ledger, located bisection route, retained nested windows, termination budget, stream windows, regular rational readback, real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the method route*

$$I, D, L, W, T \longmapsto S \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{BisectionMethod}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionMethod}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, L, W, T, S, R, E, H, C, P, N$. It has no coordinate for a selected root, completed intersection, trichotomy, open-cover compactness, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BisectionMethodUp.BisectionMethodTasteGate single carrier alignment`

Theorem 2108.2 (BisectionMethod NameCert obligations)**.** *For every accepted* $\mathsf{BisectionMethod}^{\uparrow}$ *carrier*

$$M = (I, D, L, W, T, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionMethod}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the located interval source, dyadic branch ledger, located bisection route, retained nested window, finite termination budget, stream windows, regular rational readback, real seal, transport, replay, provenance, and local name;*

2. *classifier transport, requiring H and C to preserve the displayed method rows componentwise and to replay only the route shown by the carrier;*

3. *branch-retention stability, requiring every retained child interval in W to be justified by the dyadic branch ledger D and the located route L;*

4. *finite-window termination, requiring the budget row T to bind each precision request to a displayed finite depth, diameter witness, and stream-window exposure;*

5. *real-seal nonescape, requiring E to be read only after S and R have displayed finite windows and regular rational readback;*

6. *ledger exhaustion, requiring every exported branch decision, retained window, readback, and seal request to occur in $I, D, L, W, T, S, R, E, H, C, P, N$.*

No BisectionMethod obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2108.1. The accepted packet displays exactly the source interval, branch ledger, located bisection route, retained windows, termination budget, finite stream windows, regular readback, terminal seal, and structural rows.

Classifier transport is forced by H and C. These rows preserve and replay only the displayed coordinates, so an accepted comparison cannot introduce a second route to branch data, tolerance data, readback, or the Real-facing seal.

Branch-retention stability is read from D, L, W. A retained child interval appears in W only after the dyadic branch ledger D has recorded the visible decision and the located route L has transported that decision to the displayed child interval.

Finite-window termination is internal to T. The row binds each request to a finite depth, diameter estimate, and ledger-exhaustion check before the stream window row S is available to the regular rational readback R.

The real-seal clause follows because E is downstream of S and R. A Real-facing consumer must first consume the displayed finite windows and regular rational readback, so the seal cannot be reached through a hidden branch selector or completed intersection.

Finally, P and N only source and name the same packet. Since the carrier contains no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2108.3 (BisectionMethod Real seal route)**.** *Every root-location or intermediate-value consumer of an accepted* $\mathsf{BisectionMethod}^{\uparrow}$ *carrier factors through the finite route*

$$I, D, L, W, T \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before reading any $\mathsf{Real}^{\uparrow}$ *seal. Consequently the consumer receives only the displayed located interval source, dyadic branch ledger, located bisection route, retained nested windows, finite termination budget, stream windows, regular rational readback, terminal real seal, componentwise transport, continuation replay, package provenance, and local naming data. It exports no selected root, completed intersection, trichotomy, open-cover compactness, quotient equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $M = (I, D, L, W, T, S, R, E, H, C, P, N)$. The source interval coordinate is I, so every method request begins with displayed endpoint, locator, bracket, and tolerance data.

The dyadic and located bisection rows form the visible branch step. The ledger D records the branch decision, and the route L transports that decision to the retained interval window exposed by W.

The finite termination row T supplies the quantitative stop condition. It records the requested precision, the finite depth, the diameter witness, and the ledger-exhaustion check before any stream-name window is exported.

The row S is therefore downstream of I, D, L, W, T, and the row R is downstream of S. The terminal real seal E is read only after the regular rational readback has been exposed.

The structural rows H, C, P, N preserve this displayed route under componentwise transport, replay, provenance, and naming. Because no carrier coordinate stores a selected root, completed intersection, trichotomy, compactness principle, quotient equality, choice schedule, or ambient completeness witness, no such object can be exported by a consumer of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionMethod}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionMethod}^{\uparrow}$ packet is generated from a located interval source row, a $\mathsf{DyadicBisection}^{\uparrow}$ branch ledger, a $\mathsf{LocatedIntervalBisection}^{\uparrow}$ route, retained $\mathsf{NestedInterval}^{\uparrow}$ windows, a finite termination and tolerance budget, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BisectionMethod}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2108.1, Definition 2108.2, and Definition 2108.3 seed the finite bisection method route from located interval source data through dyadic branch ledgers, located bisection rows, retained nested windows, termination budgets, stream windows, regular rational readback, terminal real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BisectionMethod}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected roots, completed intersections, trichotomy, open-cover compactness, quotient equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, branch-retention stability, finite-window termination, method-route exactness, real-seal nonescape, and ledger exhaustion for BEDC.Derived.BisectionMethodUp.

Chapter 2109

A Concrete Naming Certificate for $\mathsf{AlgebraicReal}^{\uparrow}$

Following van der Waerden and Bishop–Bridges constructive analysis, the $\mathsf{AlgebraicReal}^{\uparrow}$ packet reads algebraic real data through a polynomial row, finite Sturm-chain sign-variation rows, an isolating rational or dyadic interval, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and a $\mathsf{Real}^{\uparrow}$ seal. It consumes the polynomial source of chapter 50, the Sturm-chain surface of chapter 2079, the finite root-isolation branch of chapter 1534, the interval ledgers of chapter 354 and chapter 306, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet exposes a finite Real-facing certificate for one isolated algebraic window; it is not an algebraic closure construction, a selected host root, a quotient-real equality principle, or an ambient completeness theorem.

Definition 2109.1 (AlgebraicReal carrier)**.** *An* $\mathsf{AlgebraicReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, S, I, D, W, R, E, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, including the coefficient ledger, degree bound, derivative row, and primitive polynomial identity under inspection.*
2. *S is the* $\mathsf{SturmSequence}^{\uparrow}$ *ledger, carrying the Euclidean-division chain, endpoint sign rows, zero-deletion witnesses, and sign-variation counts used by the isolation read.*
3. *I is the* $\mathsf{SturmRootIsolation}^{\uparrow}$ *branch row, exposing the retained rational isolating interval and the displayed variation-count evidence that selected it.*
4. *D is the dyadic refinement row obtained from the retained interval, with endpoint comparison rows and requested precision windows visible as* $\mathsf{DyadicInterval}^{\uparrow}$ *data.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exported from the isolating interval and its dyadic refinements.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D at the requested unary precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the retained algebraic window, reached only after P, S, I, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the polynomial source, Sturm ledger, isolating interval, dyadic refinement, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$P, S \longmapsto I \longmapsto D, W \longmapsto R \longmapsto E.$$

10. *K is the* Pkg *provenance over* $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, $\mathsf{SturmRootIsolation}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AlgebraicReal}^{\uparrow}}$ *registration row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for real-closedness, algebraic closure, selected roots in a host field, quotient real equality, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AlgebraicRealUp.TasteGate.AlgebraicRealTasteGate single carrier alignment`

Theorem 2109.2 (AlgebraicReal NameCert obligations)**.** *For every accepted* $\mathsf{AlgebraicReal}^{\uparrow}$ *carrier*

$$A = (P, S, I, D, W, R, E, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{AlgebraicReal}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed packet of Definition 2109.1;*
2. *classifier transport through the componentwise* hsame *rows in H and the replay rows in C;*
3. *polynomial-source exactness, requiring the coefficient ledger, degree bound, derivative row, and primitive identity of P to be the source consumed by the Sturm ledger S;*
4. *isolation-window exactness, requiring the retained interval I to be selected by the sign-variation rows of S and refined only through the displayed dyadic row D;*
5. *Real-seal nonescape and ledger exhaustion, requiring every read of E to factor through $P, S, I, D, W, R, H, C, K, N$ before a real-facing consumer can inspect the seal.*

No further NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2109.1. The displayed packet lists the polynomial source, Sturm ledger, isolating interval, dyadic refinement, stream windows, regular rational readback, real seal, and structural BEDC rows.

Classifier transport is componentwise. The row H preserves each displayed coordinate by hsame, while C replays the same route from polynomial inspection through real-seal consumption.

Polynomial-source exactness is read before isolation. The Sturm ledger S has access only to the polynomial data in P, so the sign-variation rows cannot be replaced by a host factorization or by a selected list of roots.

Isolation-window exactness is then read from S, I, D. The retained interval I is the branch justified by the displayed variation count, and the dyadic row D only refines that retained interval.

Finally, the rows W, R, E, K, N expose the finite window to stream, regular-rational, and real-facing consumers while preserving package provenance and local naming. Since the carrier has no coordinate for the excluded principles, the five rows exhaust the local NameCert surface. □

Theorem 2109.3 (AlgebraicReal isolating-interval consumer)**.** *For every accepted* $\mathsf{AlgebraicReal}^{\uparrow}$ *carrier $A = (P, S, I, D, W, R, E, H, C, K, N)$, every algebraic-real consumer factors through the displayed* $\mathsf{SturmRootIsolation}^{\uparrow}$ *interval row I before any* $\mathsf{Real}^{\uparrow}$ *seal row E is visible. Thus the consumer receives the polynomial source, Sturm sign-variation ledger, retained rational or dyadic isolating interval, stream windows, regular rational readback, and structural transport rows as finite BEDC data before reaching a real-facing seal.*

Proof. Start with a consumer of the terminal seal E. By the carrier definition, E is not an independent real object; it is stored after the polynomial, Sturm, interval, dyadic, stream, and regular-rational rows.

The polynomial source P feeds the Sturm ledger S. That ledger supplies the endpoint signs and variation-count evidence from which the retained interval I is selected.

Only after I is fixed can the dyadic refinement D, stream windows W, and regular rational readback R be read. The structural rows H, C, K, N transport, replay, package, and name that same route.

Consequently the consumer cannot bypass the $\mathsf{SturmRootIsolation}^{\uparrow}$ interval to inspect E. The seal exposes only the retained algebraic window, and it exports no selected host root, quotient real equality, real trichotomy, algebraic closure, real-closedness, or ambient completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlgebraicReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlgebraicReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2109.1, Definition 2109.2, and Definition 2109.3 seed the finite algebraic-real route from polynomial source rows through Sturm isolation, dyadic refinement, stream windows, regular rational readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlgebraicReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real-closedness, algebraic closure, selected roots, quotient real equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, polynomial-source exactness, isolation-window exactness, and Real-seal nonescape for BEDC.Derived.AlgebraicRealUp.

Constructive story: An $\mathsf{AlgebraicReal}^{\uparrow}$ packet is generated from a polynomial source row, finite Sturm-chain sign-variation rows, a SturmRootIsolation interval row, dyadic refinements, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2110

A Concrete Naming Certificate for $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$

Following Bishop and Bridges' constructive nested-interval treatment, the $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ packet records the finite choice-seal route by which bounded nested intervals are admitted, localized, handed to $\mathsf{CantorIntersection}^{\uparrow}$, read through finite windows, and finally sealed as $\mathsf{Real}^{\uparrow}$ data without exporting a host-selected point. It sits on the Cantor-intersection surface of chapter 1421, the located nested-interval surface of chapter 1713, the convergence handoff of chapter 2057, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet is a human-origin L10 finite route for Real and Completion consumers; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, countable choice, a selected host point, quotient stream equality, trichotomy, open-cover compactness, or host real equality.

Definition 2110.1 (Bounded nested real choice-seal carrier)**.** *A* $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, C, W, R, D, E, H, T, P, N)$$

with the following displayed rows:

1. *I is the bounded nested interval row, exposing the initial bounded source, endpoint names, nested containment ledger, and finite diameter request.*
2. *L is the* $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$ *row, attaching located endpoint comparisons and interval-membership reads to the same nested source.*
3. *C is the* $\mathsf{CantorIntersection}^{\uparrow}$ *handoff row, available only after the bounded source and located rows have been displayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family read from the nested interval route, with one visible window for each finite precision request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and their regular rational approximants.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic endpoint budgets, diameter bounds, and comparison tolerances consumed by W and R.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, C, W, R, D have been exposed.*

8. *H records componentwise* hsame *transport for bounded source data, located nested data, Cantor handoff, windows, readback, tolerances, seal, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay of the bounded nested route*

$$I, L, C, D \longmapsto W \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{NestedIntervalConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, C, W, R, D, E, H, T, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, a selected host point, quotient stream equality, trichotomy, open-cover compactness, or host real equality.*

Lean4 ✓ `BEDC.Derived.BoundedNestedRealChoiceSealUp.BoundedNestedRealChoiceSealTasteGate single carrier alignment`

Theorem 2110.2 (BoundedNestedRealChoiceSeal NameCert obligations)**.** *For every accepted* $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ *carrier*

$$B = (I, L, C, W, R, D, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed bounded source, located nested data, Cantor handoff, finite windows, regular readback, dyadic tolerances, terminal seal, transport, replay, provenance, and local name;*
2. *bounded nested-source exposure, requiring every interval read to pass through I before any located or Cantor-facing row is consumed;*
3. *Cantor handoff, requiring C to be read only after I and L and before any Real-facing seal request;*
4. *finite-window readback, requiring every consumer to pass through W, R, D before reaching E;*
5. *Real seal nonescape, requiring E to remain attached to $I, L, C, W, R, D, H, T, P, N$;*
6. *transport, requiring H to preserve the displayed rows componentwise;*
7. *replay, requiring T to replay only the displayed route from bounded nested data to the terminal seal;*
8. *provenance, requiring P to source the packet from the named BEDC rows listed in the carrier;*
9. *local naming, requiring N to name exactly this finite packet.*

No bounded nested real choice-seal obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2110.1. The accepted packet displays exactly the bounded interval row, located nested row, Cantor handoff, finite windows, regular readback, dyadic tolerances, terminal seal, and structural rows.

The bounded-source and Cantor-handoff obligations are read in order. A consumer first sees I, then the located row L, and only then the handoff row C; the handoff cannot be read as an ambient completion theorem or as a selected point.

The finite-window readback obligation is the route through W, R, D. The StreamName windows expose finite observations, the RegSeqRat row records the regular rational readback of those observations, and the DyadicRatCore row supplies the endpoint and tolerance comparisons used by both.

The structural obligations are H, T, P, N. They transport, replay, source, and name only the displayed coordinates. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2110.3 (BoundedNestedRealChoiceSeal Cantor handoff route)**.** *Every* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ *carrier reads the packet only through the displayed finite route*

$$I, L, C, D \longmapsto W \longmapsto R \longmapsto E \longmapsto H, T, P, N.$$

Thus the consumer receives bounded nested interval data, located nested-interval data, the CantorIntersection handoff, dyadic tolerances, finite StreamName windows, RegSeqRat readback, the terminal Real seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no ambient Real completeness theorem, countable-choice schedule, selected host point, quotient stream equality, trichotomy principle, open-cover compactness proof, or host real equality witness.

Proof. Start with an accepted packet $B = (I, L, C, W, R, D, E, H, T, P, N)$. The only Cantor-facing coordinate in the carrier is C, and the carrier definition places C downstream of the bounded source row I and the located nested row L.

The consumer then follows the finite readback route. The dyadic row D fixes the tolerance and endpoint budget, W exposes the corresponding finite StreamName windows, and R records the regular rational readback of those windows before the terminal seal E can be consumed.

The rows H, T, P, N preserve this route under componentwise transport, continuation replay, package provenance, and local naming. Because no other coordinate is present, a CantorIntersection consumer has no path to a hidden selector, quotient equality, countable choice, trichotomy, compactness, ambient Real completeness, or host real equality. □

Theorem 2110.4 (BoundedNestedRealChoiceSeal nonescape)**.** *An accepted* $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ *packet exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, countable choice, selected host point, quotient stream equality, trichotomy, open-cover compactness, or host real equality. Every exported Real-facing read remains attached to the finite packet*

$$B = (I, L, C, W, R, D, E, H, T, P, N)$$

and to the displayed route through bounded nested data, located nested data, Cantor handoff, finite windows, regular readback, dyadic tolerances, transport, replay, provenance, and local naming.

Proof. Apply Definition 2110.2. The local NameCert surface exposes the terminal seal E only as part of the displayed packet, after I, L, C, W, R, D and before the structural rows H, T, P, N finish the route.

Apply Definition 2110.3. Any CantorIntersection consumer reads the same finite interval, window, tolerance, readback, and seal chain, so it cannot replace the route by a selected host point or by a global completeness principle.

Finally inspect Definition 2110.1. The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, countable choice, selected host point, quotient stream equality, trichotomy, open-cover compactness, or host real equality. Therefore none of those objects can be exported by an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}$ packet is generated from bounded nested interval data, located nested-interval data, a CantorIntersection handoff, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2110.1, Definition 2110.2, Definition 2110.3, and Definition 2110.4 seed the finite nested-interval choice-seal route from bounded and located interval data through Cantor handoff, finite windows, regular readback, dyadic tolerances, terminal Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNestedRealChoiceSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, countable choice, selected host points, quotient stream equality, trichotomy, open-cover compactness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded nested-source exposure, Cantor handoff, finite-window readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BoundedNestedRealChoice

Chapter 2111

A Concrete Naming Certificate for $\mathsf{HurwitzApproximation}^{\uparrow}$

Following Hurwitz and Hardy–Wright, the $\mathsf{HurwitzApproximation}^{\uparrow}$ packet records the finite continued-fraction route to rational approximation with a displayed constant witness. It consumes the continued-fraction source of chapter 342, the convergent-prefix surface of chapter 1777, the Farey-neighbor ledger of chapter 1759, the Stern–Brocot branch route of chapter 1704, the Dirichlet-approximation certificate of chapter 1871, the rational readback boundary of chapter 1668, the regular rational sequence route of chapter 301, and the terminal real-name interface of chapter 38. The packet names a finite approximation certificate for one displayed real input and one requested budget; it is not an infinite continued fraction, a quotient-real equality theorem, a trichotomy principle, a countable-choice selector, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2111.1 (Hurwitz approximation carrier)**.** *A* $\mathsf{HurwitzApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, Q, A, C, F, S, D, R, E, T, U, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *input name, exposed only through finite regular-rational windows and dyadic error budgets.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplying the finite rational windows consumed by the approximation certificate.*
3. *A is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *prefix row. It records a finite list of partial quotients, denominator positivity witnesses, and the displayed stopping index used by the certificate.*
4. *C is the* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *row attached to A, listing the convergent numerator and denominator pairs through the stopping index.*
5. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *neighbor row used to compare adjacent rational windows against the same denominator budget.*
6. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *branch row that replays the same rational interval as a finite left-right word.*
7. *D is the* $\mathsf{DirichletApproximation}^{\uparrow}$ *budget row, recording the denominator threshold, dyadic error request, and finite pigeonhole-style witness that bounds the search space.*

8. *R is the rational-approximation output row. It contains a displayed fraction p/q, the denominator bound, and the dyadic inequality row asserting the Hurwitz-style error estimate for the current budget.*

9. *E records the excluded-principle ledger: no selected infinite expansion, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row may be read from the carrier.*

10. *T records componentwise* hsame *transport for the input name, regular readback, continued-fraction prefix, convergent table, Farey row, Stern–Brocot row, Dirichlet row, output fraction, excluded-principle ledger, replay, provenance, and local naming rows.*

11. *U records displayed* Cont *replay of the finite route*

$$X, Q \longmapsto A \longmapsto C \longmapsto F, S, D \longmapsto R, E.$$

12. *P is the* Pkg *provenance over* $\mathsf{HurwitzApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{ContinuedFractionConvergents}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DirichletApproximation}^{\uparrow}$, RealRationalApproximati BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{HurwitzApproximation}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $X, Q, A, C, F, S, D, R, E, T, U, P, N$. It has no coordinate for selected infinite expansions, quotient real equality, real trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HurwitzApproximationUp.HurwitzApproximationTasteGate single carrier alignment`

Theorem 2111.2 (HurwitzApproximationUp NameCert obligations)**.** *For every accepted* $\mathsf{HurwitzApproximation}^{\uparrow}$ *carrier*

$$H = (X, Q, A, C, F, S, D, R, E, T, U, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HurwitzApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed packet of Definition 2111.1;*

2. *input readback exactness, requiring the* $\mathsf{Real}^{\uparrow}$ *input X to be inspected only through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *windows and dyadic budgets in Q;*

3. *continued-fraction prefix exactness, requiring A and C to expose only the displayed finite prefix and its convergent table through the stopping index;*

4. *neighbor-route compatibility, requiring the Farey row F and Stern–Brocot row S to replay the same rational interval inspected by the convergent table;*

5. *approximation-budget exactness, requiring D and R to display the denominator threshold, chosen fraction, and dyadic error inequality for the current budget;*

6. *excluded-principle nonescape through E, requiring every consumer to refuse selected infinite expansions, quotient real equality, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*

7. *classifier transport and provenance through T, U, P, N, requiring componentwise* hsame *transport, displayed replay, package source, and local name to preserve exactly these rows.*

No Hurwitz-approximation obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.HurwitzApproximationUp.HurwitzApproximationCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 2111.1. The rows X and Q are the only real-facing input interface, so every approximation claim begins with finite rational windows and dyadic budgets rather than quotient-real equality.

The finite continued-fraction data are then read from A and C. Induct over the displayed prefix: the base row exposes the first convergent, and each successor row updates the numerator-denominator pair while preserving the positivity and determinant ledgers carried by the convergent table.

The neighbor rows F and S compare the same displayed rational interval. A finite induction over the Farey adjacency row matches the corresponding Stern–Brocot word, so the branch route cannot import an infinite expansion or a global order split.

The budget rows D and R supply the approximation witness. The dyadic error budget is checked against the finite convergent prefix and the denominator threshold, producing the displayed fraction p/q and its Hurwitz-style inequality for this request.

Finally E records the refused principles, while T, U, P, N transport, replay, source, and name the same finite surface. Since the carrier has no additional coordinate, the seven listed rows exhaust the local NameCert obligations. □

Theorem 2111.3 (Finite continued-fraction handoff)**.** *Every rational-approximation consumer of an accepted* $\mathsf{HurwitzApproximation}^{\uparrow}$ *carrier factors through the finite route*

$$X, Q \longmapsto A \longmapsto C \longmapsto F, S, D \longmapsto R, E, T, U, P, N.$$

Consequently the consumer receives a displayed real-name readback, a finite continued-fraction prefix, its convergent table, Farey and Stern–Brocot comparison rows, a Dirichlet-style dyadic budget row, a rational output fraction, and the excluded-principle ledger before inspecting any approximation claim.

Proof. Start with a consumer of the output fraction in R. By the carrier definition, R is not an independent rational approximation; it is stored after the input readback rows X, Q, the finite continued-fraction prefix A, and the convergent table C.

The handoff from A to C is finite-row induction over the displayed prefix. Each step adds one partial quotient and updates one convergent row, so the consumer never receives an infinite expansion or a selected tail.

The comparison handoff then passes through F, S, D. The Farey and Stern–Brocot rows identify the same finite rational interval, while the Dirichlet row supplies the denominator and dyadic error budget used by the output row.

The structural rows E, T, U, P, N close the public boundary. They record the refused principles, preserve the displayed route under componentwise transport, replay the route, source it, and assign the local NameCert name. Hence the consumer boundary is exactly the finite continued-fraction route and not any ambient real-completeness or choice principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{HurwitzApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HurwitzApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2111.1, Definition 2111.2, and Definition 2111.3 seed the finite continued-fraction route from Real and RegSeqRat readback through convergent prefixes, Farey and Stern–Brocot comparison rows, Dirichlet-style budgets, rational output, excluded-principle refusal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HurwitzApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite expansions, quotient real equality, real trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked input readback exactness, continued-fraction prefix exactness, neighbor-route compatibility, approximation-budget exactness, excluded-principle nonescape, classifier transport, and provenance/name exhaustion for BEDC.Derived.HurwitzApproximationUp.

Constructive story: A HurwitzApproximation$^{\uparrow}$ packet is generated from a RealUp input name, RegSeqRat finite readback, ContinuedFraction finite prefix, convergent table, Farey and SternBrocot comparison rows, a DirichletApproximation budget row, a rational output fraction, an excluded-principle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2112

A Concrete Naming Certificate for $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ packet names the finite located-order test by which two displayed real names are compared through regular Cauchy readbacks, shared windows, dyadic tolerances, positive-gap evidence when present, and located-interval context before any Real-facing consumer reads the result. It sits after the comparison-test surface of chapter 1459, the constructive order surface of chapter 1716, the positive-separation surface of chapter 304, the located interval context of chapter 1709, the terminal real-name seal of chapter 38, the regular-rational readback of chapter 301, the finite stream-window route of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite Bishop-style located order test for $\mathsf{Real}^{\uparrow}$ names; it is not host trichotomy, quotient real equality, a representative selector, countable choice, or ambient real completeness.

Definition 2112.1 (Cauchy real located order test carrier)**.** *A* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, W, D, Q, O, A, L, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two source rows. Each row is read as a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *packet and then sealed as a* $\mathsf{Real}^{\uparrow}$ *name before any located-order verdict is exported.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family. It records the finite observation lengths, synchronised cuts, endpoint readbacks, and comparison windows inspected for both source rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget. It records the requested dyadic scale, positive gap threshold, sign slack, and exhaustion ledger used by the comparison and locatedness rows.*
4. *Q is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *row. It tests the two regular Cauchy readbacks on the shared windows and dyadic budget before the terminal real-name seals are consumed.*
5. *O is the* $\mathsf{CauchyRealOrder}^{\uparrow}$ *row. It stores the oriented order verdict as a finite consequence of X, Y, W, D, Q and does not supply host trichotomy.*
6. *A is an* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-gap row when the displayed windows and dyadic budget separate the two real names.*
7. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *context row. It records the interval enclosure and locatedness data in which the order test is read.*

8. *H records componentwise* hsame *transport for the two source rows, shared windows, dyadic budget, comparison row, order row, positive-gap row when present, located-interval row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each located-order request through $W, D, Q, O, A, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{CauchyRealOrder}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these twelve rows form one finite located-order test packet. It has no coordinate for host trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyRealLocatedOrderTestUp.CauchyRealLocatedOrderTestTasteGate single carrier alignment`

Theorem 2112.2 (Cauchy real located order test NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ *carrier $T = (X, Y, W, D, Q, O, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatedOrd}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular-rational source rows, terminal real-name seals, shared stream windows, dyadic tolerance budget, comparison-test row, order row, positive-gap row when present, located-interval context row, transport, replay, provenance, and local naming row;*
2. *classifier exactness, requiring accepted packets to compare only the displayed $X, Y, W, D, Q, O, A, L, H, C, P, N$ coordinates;*
3. *regular Cauchy comparison readback, requiring Q to inspect the two source rows through the shared windows W and dyadic budget D before the order row is read;*
4. *located order compatibility, requiring O to export an oriented verdict only inside the located interval context L and only after the comparison-test route has been displayed;*
5. *positive-gap visibility, requiring every separating verdict to display its* $\mathsf{ApartnessReal}^{\uparrow}$ *row A as finite dyadic gap evidence over the same W and D;*
6. *nonseparating visibility, requiring every nonseparating read to record the exhausted windows and dyadic slack rather than a quotient equality proof;*
7. *transport and replay stability, requiring H and C to preserve the source, window, tolerance, comparison, order, apartness, located-context, provenance, and local naming rows componentwise;*
8. *ledger exhaustion, excluding host trichotomy, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No located-order test obligation is stored outside these rows.

Proof. Project the carrier $T = (X, Y, W, D, Q, O, A, L, H, C, P, N)$. The projection exposes exactly the two source rows, common window family, dyadic tolerance budget, comparison row, order row, positive-gap row when present, located-interval context row, and structural rows.

The comparison route is finite. The source rows X and Y are inspected through the shared windows W, and the dyadic budget D supplies the scale and slack at which the comparison-test row Q is allowed to read them.

The order and locatedness rows are downstream of this comparison. The row O records only the oriented verdict produced by the displayed comparison route, while L records the interval context in which the verdict is meaningful.

Positive and nonseparating reads use the same ledger. When a positive gap is present, A displays the $\mathsf{ApartnessReal}^{\uparrow}$ separation row. When the gap is not displayed at the requested scale, the packet records the exhausted finite windows and slack; it does not replace that record by quotient equality.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier coordinates stores a host trichotomy, quotient equality, representative selector, choice schedule, completeness datum, Lean theorem, or bridge, no further local obligation can be read from the packet. □

Theorem 2112.3 (Cauchy real located order test finite decision route)**.** *Every accepted* CauchyRealLocatedOrderTes *read factors through the finite route*

$$X, Y \longmapsto W, D \longmapsto Q \longmapsto O, A, L \longmapsto H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *consumer sees the result. Hence the packet exports only the displayed located-order test data, and exports no host trichotomy, quotient real equality, selected representative, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted located-order test carrier. The only source coordinates are X and Y, and the carrier requires each to pass through regular-rational readback and the terminal real-name seal before the public result is exposed.

The shared windows and tolerance budget come next. The finite windows in W select the inspected stream segments, while the dyadic budget D records the comparison scale, requested gap, sign slack, and exhaustion data.

The comparison-test row Q consumes exactly those rows. The oriented order row O, the positive-gap row A when present, and the located-interval context row L are therefore downstream of the displayed comparison route rather than independent Real-level principles.

The structural rows close the route. Componentwise transport in H, replay in C, provenance in P, and local naming in N preserve the same finite packet before a $\mathsf{Real}^{\uparrow}$ consumer can read the verdict. Because the route contains no coordinate for trichotomy, quotient equality, representative selection, countable choice, or completeness, none of those principles is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows sealed as $\mathsf{Real}^{\uparrow}$ names, shared $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance budget, a $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ row, a $\mathsf{CauchyRealOrder}^{\uparrow}$ row, an $\mathsf{ApartnessReal}^{\uparrow}$ positive-gap row when present, a $\mathsf{LocatedRealInterval}^{\uparrow}$ context row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2112.1, Definition 2112.2, and Definition 2112.3 seed the finite located-order test boundary over regular Cauchy readbacks, shared stream windows, a dyadic tolerance budget, comparison-test data, order data, apartness data when present, located-interval context, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.CauchyRealLocatedOrderTestUp covering the twelve-row carrier, local NameCert obligations, and finite decision route.

Chapter 2113

A Concrete Naming Certificate for $\mathsf{KummerTest}^{\uparrow}$

Following Kummer and the Bishop–Bridges constructive-analysis reading of positive series tests, the $\mathsf{KummerTest}^{\uparrow}$ packet records the finite auxiliary-sequence comparison route that turns adjacent-term quotient data into a displayed regular Cauchy tail handoff. It refines the positive $\mathsf{RealSeries}^{\uparrow}$ surface of chapter 1581, sits beside the ratio route of chapter 1461, the comparison route of chapter 1459, and the neighboring defect-test surface of chapter 1719, and it exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet is a finite NameCert boundary for Kummer-style convergence testing; it is not a host infinite sum, selected limit, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 2113.1 (Kummer test carrier)**.** *A* $\mathsf{KummerTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, Q, U, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source* $\mathsf{RealSeries}^{\uparrow}$ *row, exposing only the inspected positive-term windows and partial-sum windows used by the current tail request.*
2. *Q is the adjacent-term quotient row, recording the finite windows on which the inspected ratios of neighboring positive terms are read.*
3. *U is the auxiliary positive comparison sequence row, displaying the finite auxiliary values and positivity witnesses used by the Kummer defect calculation.*
4. *D is the dyadic defect-budget row, assigning each displayed tail request a finite threshold and rational margin for the Kummer inequality on the inspected windows.*
5. *T is the comparison-test handoff row, converting Q, U, D and the source windows in A into the finite majorant route consumed by* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$*.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff has been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
8. *H stores componentwise* hsame *transport for the source, quotient, auxiliary, defect-budget, comparison, readback, seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from each tail request through Q, U, D, T, R, E.*

10. *P is the* Pkg *provenance over* $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{KummerTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Q, U, D, T, R, E, H, C, P, N$. It has no coordinate for host infinite sums, selected limits, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.KummerTestUp.KummerTestTasteGate single carrier alignment`

Theorem 2113.2 (Kummer test NameCert obligations)**.** *For every accepted* $\mathsf{KummerTest}^{\uparrow}$ *carrier*

$$K = (A, Q, U, D, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KummerTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the source* $\mathsf{RealSeries}^{\uparrow}$ *row, adjacent-term quotient row, auxiliary positive comparison sequence row, dyadic defect-budget row, comparison-test handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*

2. *auxiliary-row positivity, requiring every accepted auxiliary read to pass through the displayed finite positivity witnesses in U;*

3. *quotient-window exactness, requiring the quotient data used by the Kummer defect to be read only from the displayed neighboring-term windows A, Q;*

4. *defect-budget stability, requiring the threshold and rational margin in D to be preserved under the accepted tail request and under componentwise transport;*

5. *comparison-test routing, requiring T to be the only row that turns the Kummer defect data into a convergence-test majorant;*

6. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring endpoint-facing data to pass through R and then E after the comparison row has been displayed;*

7. *transport, replay, and provenance discipline, requiring H, C, P, N to transport, replay, source, and name exactly the displayed route and no unrecorded analytic principle.*

Proof. Project the carrier of Definition 2113.1. Carrier exposure is the inventory $A, Q, U, D, T, R, E, H, C, P, N$, so the local certificate has no detached row for an infinite sum, quotient series equality, selected limit, or ambient completeness theorem.

The auxiliary and quotient clauses are read before any convergence handoff. The row U displays only finite auxiliary comparison values together with their positivity witnesses, while A and Q display the source windows and neighboring-term quotient windows used by the Kummer defect.

The defect budget is then forced through D. Its threshold and rational margin are the only data attached to the requested tail, and componentwise hsame transport in H preserves the same threshold, margin, source windows, quotient windows, and auxiliary row.

The comparison route is exactly T. Once Q, U, D have been displayed over the source windows in A, the packet may enter the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ surface; before that row appears, no regular Cauchy conclusion is exported.

Finally, R and E supply the regular rational and Real-facing endpoint route. The replay, provenance, and local naming rows C, P, N preserve this same route, so none of them can introduce host infinite sums, selected limits, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 2113.3 (Kummer test Cauchy tail handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{KummerTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Strea}$$

Concretely, the read consumes source $\mathsf{RealSeries}^{\uparrow}$ *windows and neighboring quotient rows, reads the auxiliary positive comparison sequence and dyadic defect budgets, uses the ratio and Raabe–Duhamel surfaces only as neighboring quotient-defect reference faces, enters the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed stream and dyadic rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 2113.1, the only source data available before the defect estimate are the $\mathsf{RealSeries}^{\uparrow}$ windows in A and the adjacent-term quotient windows in Q.

The Kummer-specific data are then read from U and D. The auxiliary row supplies finite positive comparison values, while the dyadic defect-budget row supplies the threshold and rational margin attached to the requested tail.

The neighboring quotient-test surfaces are reference faces rather than endpoints. The $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ and $\mathsf{RaabeDuhamelTest}^{\uparrow}$ rows identify the surrounding adjacent-term and defect-test discipline, but the accepted $\mathsf{KummerTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ budgets recorded in the carrier, then through R and E. Thus every endpoint-facing read reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only through the displayed comparison, stream, dyadic, readback, seal, transport, replay, provenance, and naming rows. □

Falsifiable prediction: Every accepted $\mathsf{KummerTest}^{\uparrow}$ endpoint-facing read factors through source RealSeries windows, adjacent-term quotient rows, an auxiliary positive comparison sequence, dyadic defect budgets, a RegularCauchyComparisonTest handoff, StreamName and DyadicRatCore display rows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, regularcauchyratiotest, regularcauchycomparisontest, raabeduhameltest, dyadicratcore, streamname, regseqrat, real: $\mathsf{KummerTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent-term quotient windows, an auxiliary positive comparison sequence, dyadic defect budgets, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KummerTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KummerTest}^{\uparrow}$ packet is generated from source RealSeries windows, adjacent-term quotient windows, an auxiliary positive comparison sequence, dyadic defect budgets, a RegularCauchyComparisonTest handoff, StreamName and DyadicRatCore display rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KummerTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2113.1, Definition 2113.2, and Definition 2113.3 seed the finite Kummer convergence-test boundary over displayed source series, quotient, auxiliary positive sequence, defect-budget, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KummerTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier exposure, auxiliary-row positivity, quotient-window exactness, defect-budget stability, comparison-test routing, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal nonescape, and transport, replay, provenance, and local naming discipline for BEDC.Derived.KummerTestUp.

Chapter 2114

A Concrete Naming Certificate for $\mathsf{LocatedMetricCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{LocatedMetricCompletion}^{\uparrow}$ packet records the L10 metric-completion boundary where located source data are carried into finite completion evidence without quotienting Cauchy names. It sits between the located metric source of chapter 1533, the completion packet of chapter 1506, the embedding surface of chapter 1512, the separated uniqueness row of chapter 1411, and the finite readback interfaces of chapter 301 and chapter 38. The packet is a NameCert seed for displayed completion-facing reads over a located metric source; it is not a quotient completion, a representative selector, a countable-choice device, an open-cover compactness principle, host equality, an ambient complete metric space, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 2114.1 (LocatedMetricCompletion carrier)**.** *A* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, F, D, E, R, S, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{LocatedMetric}^{\uparrow}$ *row, including the displayed metric distance, locatedness witness, and finite observation histories used by completion consumers.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row that exposes the completion-facing interface over the same source data.*
3. *F is either a* $\mathsf{CauchyFilter}^{\uparrow}$ *cofinal-evidence row or an empty marker when the displayed completion route uses directed nets.*
4. *D is either a* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row or an empty marker when the displayed completion route uses Cauchy filters.*
5. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row by which located source observations enter the completion-facing surface.*
6. *R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row used for numerical tolerance requests.*
7. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row that identifies displayed limit readings only through the zero-distance route available from the located source and completion packet.*

8. *H records componentwise* hsame *transport for the located source, metric-completion row, Cauchy evidence branch, embedding row, readback row, separatedness row, replay row, provenance row, and local naming row.*

9. *C records the displayed* Cont *replay from located metric observations to the metric-completion interface, Cauchy-filter or Cauchy-net evidence, completion embedding, finite readback, separated comparison, provenance, and local name.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedMetricCompletion}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, ambient complete metric spaces, or detached completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedMetricCompletionUp.LocatedMetricCompletionTasteGate single carrier alignment`

Theorem 2114.2 (LocatedMetricCompletion NameCert obligations)**.** *For an accepted* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *carrier $L = (X, M, F, D, E, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricCompletion}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed located metric source, metric-completion row, selected Cauchy evidence branch, completion embedding, finite readback, separatedness row, transports, replay route, provenance, and local name;*

2. *located-source handoff, requiring every completion-facing read to begin with X before the metric-completion row M or any Cauchy evidence row is consumed;*

3. *cofinal evidence selection, requiring each accepted read to use exactly one visible branch, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F or the* $\mathsf{CauchyNet}^{\uparrow}$ *row D;*

4. *embedding coverage, requiring source observations to reach the completion surface only through the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row E;*

5. *finite readback compatibility, requiring numerical tolerance requests to pass through R before separated comparison or local naming;*

6. *separated uniqueness, requiring equality of displayed completion reads to pass through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row S after the located source, completion interface, Cauchy evidence branch, embedding, and readback have been replayed;*

7. *ledger non-escape, stating that no quotient completion equality, selected representative, countable choice, open-cover compactness, host equality, ambient complete metric space, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem is exported.*

Proof. Carrier admission is projection from Definition 2114.1. The tuple displays the located source, metric-completion row, filter or net branch, embedding row, finite readback, separatedness row, and structural support rows.

The located-source handoff is forced by the order of the replay row C. It reads X first, then M, and only then the selected cofinal evidence branch F or D.

Cofinal evidence selection is local to the carrier. A filter-presented read consumes F while D is empty, and a net-presented read consumes D while F is empty, so the packet exposes one completion route before E and R are inspected.

Embedding coverage follows from the presence of E as the only source-to-completion coordinate. The readback row R handles tolerance requests, and the separatedness row S handles equality of displayed limit readings after the earlier rows have been replayed.

The rows H, C, P, N transport, replay, package, and name exactly that finite route. Since no remaining coordinate names quotient equality, selected representatives, choice, compactness, host equality, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be read from the local certificate. □

Theorem 2114.3 (LocatedMetricCompletion source handoff)**.** *Let $L = (X, M, F, D, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *packet. Every completion-facing read of L factors through the finite route*

$$X \longmapsto M \longmapsto (F \text{ or } D) \longmapsto E \longmapsto R \longmapsto S \longmapsto H, C, P, N,$$

where X is the $\mathsf{LocatedMetric}^{\uparrow}$ *source row, M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, F or D is the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *evidence, E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row, and R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row.*

Proof. Begin with the source row X. A completion-facing consumer first reads the located metric observations, including the displayed distance and locatedness evidence.

The completion interface is then read through M. The cofinal evidence branch is selected next: either the Cauchy-filter row F or the Cauchy-net row D supplies the visible late-window evidence.

The source observation enters completion-facing form only through E. Numerical requests then pass through the finite readback row R, and equality or uniqueness requests pass through the separatedness row S.

Finally, H, C, P, N transport and name the same route. No other coordinate is available, so every completion-facing read has the displayed factorization. □

Theorem 2114.4 (LocatedMetricCompletion branch determinacy)**.** *Let $L = (X, M, F, D, E, R, S, H, C, P, N)$ be an accepted* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *packet. Any accepted completion-facing read determines one Cauchy-evidence branch before embedding or readback is inspected: a filter-presented read consumes the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F with D empty, and a net-presented read consumes the* $\mathsf{CauchyNet}^{\uparrow}$ *row D with F empty. Transport through H and replay through C cannot switch the selected branch after the embedding row E or the readback row R has been read.*

Proof. Project the carrier of Definition 2114.1. The rows F and D are the only visible Cauchy-evidence coordinates, and the carrier description makes each one either an evidence row or the empty marker complementary to the other branch.

Apply Definition 2114.3. Every completion-facing read reaches F or D after the located source row X and the metric-completion row M, but before the embedding row E and the readback row R are available.

Finally apply Definition 2114.5. The support rows H and C transport and replay the already selected finite route; they do not add a second branch coordinate, and no quotient completion, selected representative, choice principle, host equality, ambient complete metric space, or detached $\mathsf{Real}^{\uparrow}$ completeness row is present to replace the selected branch. □

Theorem 2114.5 (LocatedMetricCompletion ledger non-escape)**.** *For an accepted* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *carrier* $L = (X, M, F, D, E, R, S, H, C, P, N)$*, every exported completion read, tolerance read, embedding read, and separated comparison is exhausted by the rows*

$$X, M, F, D, E, R, S, H, C, P, N.$$

Consequently the packet excludes quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, ambient complete metric spaces, and detached completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Completion reads begin at the located source X and the metric-completion row M. They then pass through exactly one displayed Cauchy evidence branch, either F or D.

Embedding reads are confined to E, and numerical tolerance reads are confined to R. The separatedness row S handles comparison of displayed limit readings only after those earlier coordinates have been replayed.

The support rows H, C, P, N preserve and name that same finite ledger. Because the carrier lists no coordinate for quotient equality, representative selection, countable choice, open-cover compactness, host equality, ambient completeness, or $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot be exported by the packet. □

Theorem 2114.6 (LocatedMetricCompletion modulus pair stability)**.** *Let* $L = (X, M, F, D, E, R, S, H, C, P, N)$ *be an accepted* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *packet. Suppose two accepted completion-facing reads share the same located source row* X*, and suppose their displayed Cauchy evidence is carried either by the filter branch* F *or by the directed-window branch* D*, transported through* H*, and replayed through* C*. Then the embedding row* E*, the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row* R*, and the separatedness row* S *preserve the same displayed completion-facing comparison.*

In particular, paired reads with the same source and replayed evidence branch cannot be separated by quotient completion equality, selected representatives, countable choice, host equality, an ambient complete metric space, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2114.1. The two reads share the row X, so their source observations, locatedness evidence, and metric-distance rows are already the same displayed source data.

Apply Definition 2114.3. Each completion-facing read must pass from X to M, then through the visible evidence branch F or D, and then through the embedding and readback rows E and R.

Use the stated transport and replay through H and C. These structural rows preserve the displayed branch evidence while moving it into the completion-facing route, so the comparison reaching E and R is still the one generated from the common source row.

Finally apply Definition 2114.5. The separatedness row S is the only row that can compare displayed completion reads after the source, evidence, embedding, and readback rows have been replayed. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality, ambient complete metric spaces, or detached $\mathsf{Real}^{\uparrow}$ completeness, no external comparison can replace the displayed one. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetricCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2114.1, Definition 2114.2, Definition 2114.3, Definition 2114.5, and Definition 2114.6 seed the finite located metric completion packet over located source data, metric-completion evidence, Cauchy-filter or Cauchy-net branches, completion embedding, finite readback, separated uniqueness, and paired modulus/readback stability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, open-cover compactness, host equality, ambient complete metric spaces, Real$^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, located-source handoff, cofinal evidence selection, embedding coverage, finite readback, separated uniqueness, and ledger non-escape for BEDC.Derived.LocatedMetricCompletionUp.

Constructive story: A LocatedMetricCompletion$^{\uparrow}$ packet is generated from a located metric source, metric-completion row, displayed Cauchy-filter or Cauchy-net evidence, completion embedding, finite readback, separatedness row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 2115

A Concrete Naming Certificate for $\mathsf{KuratowskiCompletionNaturality}^{\uparrow}$

Following Kuratowski, Kelley, and Bishop–Bridges, the $\mathsf{KuratowskiCompletionNaturality}^{\uparrow}$ packet records the finite L10 naturality boundary where a source metric object first enters the Kuratowski completion route and is then compared with the completion-reflection route. It sits over the Kuratowski completion surface of chapter 2065, the reflection witness of chapter 403, the real-reflection seal of **??**, the metric completion interface of chapter 1506, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38.

The packet names the comparison square as finite NameCert data. It does not identify quotient completions, select representatives, assert host function equality, invoke countable choice, import trichotomy, or claim ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2115.1 (Kuratowski completion naturality carrier)**.** *A* $\mathsf{KuratowskiCompletionNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, E, C, F, R, S, D, A, H, T, P, N)$$

with the following displayed rows:

1. *M is the source metric row, carrying the finite endpoint, distance, and Cauchy-window requests inspected by both completion routes.*
2. *E is the* $\mathsf{KuratowskiCompletion}^{\uparrow}$ *row that sends the source metric observations through the Kuratowski embedding and completion seal.*
3. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row reached by the Kuratowski route after the displayed completion seal is exposed.*
4. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *witness row that displays the reflection route for the same source metric observations.*
5. *R is the* $\mathsf{RealCompletionReflection}^{\uparrow}$ *row that returns the reflection route to the public real-completion boundary.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row shared by the Kuratowski and reflection reads.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared finite windows.*

8. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the finite dyadic observations into the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *H records componentwise* hsame *transport for the source metric, Kuratowski completion, metric-completion consumer, reflection witness, real-reflection seal, stream window, dyadic ledger, regular readback, replay, provenance, and local naming rows.*

10. *T records displayed* Cont *replay of the two routes*

$$M \longmapsto E, C \longmapsto S, D, A \quad \textit{and} \quad M \longmapsto F, R \longmapsto S, D, A.$$

11. *P is the* Pkg *provenance over* $\mathsf{KuratowskiCompletionNaturality}^{\uparrow}$, $\mathsf{KuratowskiCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RealCompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiCompletionNaturality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, C, F, R, S, D, A, H, T, P, N$. It has no coordinate for quotient completion equality, host function equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.KuratowskiCompletionNaturalityUp.KuratowskiCompletionNaturalityTasteGate`
`single carrier alignment`

Theorem 2115.2 (Kuratowski completion naturality obligations)**.** *For every accepted* KuratowskiCompletionNatura *carrier*

$$K = (M, E, C, F, R, S, D, A, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiCompletionNaturality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric row, Kuratowski completion row, metric-completion consumer row, completion-reflection witness, real-reflection seal, StreamName window, DyadicRatCore ledger, RegSeqRat readback, transport, replay, provenance, and local naming row;*

2. *Kuratowski-completion exposure, requiring E and C to display the source metric read through the Kuratowski completion route before any completion-facing consumer receives it;*

3. *reflection-square routing, requiring F and R to display the same source metric read through the completion-reflection and real-reflection route before it reaches the shared analytic rows;*

4. *Real-seal nonescape, requiring S, D, A to be the only finite observation path from either route to the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*

5. *transport, replay, and provenance exhaustion, requiring H, T, P, N to transport, replay, source, and name only the listed rows.*

No obligation is stored outside these rows, and none exports quotient completions, host function equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Carrier admission is projection from Definition 2115.1. The accepted packet displays exactly the source metric row, the two completion-facing routes, the shared finite observation rows, and the structural rows.

Kuratowski-completion exposure is read from E and C. The row E places the source metric observations inside the Kuratowski completion route, while C records the metric-completion consumer surface available after that route is displayed.

Reflection-square routing is read from F and R. The reflection witness and real-reflection seal expose the parallel route for the same source metric row before either side can inspect the shared stream, dyadic, and regular-readback rows.

Real-seal nonescape is the row inventory S, D, A. Both routes must pass through the same finite StreamName window, DyadicRatCore tolerance ledger, and RegSeqRat readback before reaching the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, T, P, N add only componentwise transport, replay, provenance, and local naming. Since the carrier contains no quotient, host-equality, representative-choice, countable-choice, trichotomy, ambient-completeness, Lean, or bridge coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2115.3 (Kuratowski completion naturality square)**.** *Let*

$$K = (M, E, C, F, R, S, D, A, H, T, P, N)$$

be an accepted $\mathsf{KuratowskiCompletionNaturality}^{\uparrow}$ *carrier. Every completion-facing consumer comparing the Kuratowski route with the reflection route factors through the finite square*

$$M \longmapsto E, C \longmapsto S, D, A \longmapsto H, T, P, N,$$
$$M \longmapsto F, R \longmapsto S, D, A \longmapsto H, T, P, N.$$

Consequently the consumer receives only the displayed source metric row, Kuratowski completion row, metric-completion consumer row, completion-reflection witness, real-reflection seal, shared finite window, dyadic ledger, regular readback, transport, replay, provenance, and local name. It exports no quotient completion, host function equality, selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Fix a completion-facing consumer of the packet. Projection gives the source row M, so both compared routes begin from the same displayed endpoint, distance, and Cauchy-window requests.

The Kuratowski side then uses E and C. These rows expose the Kuratowski completion route and the metric-completion consumer surface before any terminal real seal is available.

The reflection side uses F and R. These rows expose the completion-reflection route and the real-reflection seal for the same source row, not for a separately chosen representative.

Both sides are forced through S, D, A. The shared finite stream window, dyadic tolerance ledger, and regular rational readback form the only observation route to the public $\mathsf{Real}^{\uparrow}$ seal.

Transport through H, replay through T, provenance through P, and local naming through N preserve exactly that square. Since no excluded coordinate occurs in the carrier, no completion-facing consumer can export quotient completion data, host function equality, a selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiCompletionNaturality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiCompletionNaturality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2115.1, Definition 2115.2, and Definition 2115.3 seed the finite naturality square comparing the Kuratowski completion route with the completion-reflection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiCompletionNaturality}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.KuratowskiCompletionNaturalityUp`

Bridge status: none

Not claimed: This chapter does not close quotient completions, host function equality, selected representatives, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Kuratowski-completion exposure, reflection-square routing, Real-seal nonescape, and structural replay for BEDC.Derived.KuratowskiCompletionNaturalityUp.

Constructive story: A KuratowskiCompletionNaturality$^{\uparrow}$ packet is generated from a source metric row, KuratowskiCompletion route, MetricCompletion consumer, CompletionReflection witness, RealCompletionReflection seal, shared StreamName windows, DyadicRatCore ledger, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2116

A Concrete Naming Certificate for $\mathsf{SignedDistance}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SignedDistance}^{\uparrow}$ packet records the finite route by which a located metric source exposes a signed distance surface without importing a host real-valued distance function. The packet is admitted only as metric, locatedness, apartness, dyadic-budget, regular-readback, and real-seal rows: its metric side is anchored in chapter 57, its located-source side in chapter 2043, its signed separation side in chapter 304, and its L10 readback spine in chapter 291, chapter 301, and chapter 38.

Definition 2116.1 (SignedDistance carrier)**.** *A* $\mathsf{SignedDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, L, A, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row, recording the finite source and target observations, the metric comparison surface, and the triangle-compatible route through which distance requests are read.*
2. *L is the* $\mathsf{LocatedSpace}^{\uparrow}$ *source row, recording the displayed located subset or located family whose boundary is being queried.*
3. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *signed-side or separation row, recording which finite side of the located boundary supplies the positive separation evidence.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, carrying the rational tolerance, sign threshold, and comparison budget used by the signed-distance request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the metric observations, dyadic budget, and signed-side comparisons.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached by the displayed regular readback after M, L, A, D, R have been exposed.*
7. *H records componentwise* hsame *transport for the metric row, located-source row, signed-side row, dyadic budget, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of every signed-distance consumer route through*

$$M, L, A, D, R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{SignedDistance}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSpace}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, A, D, R, E, H, C, P, N$. It has no coordinate for a host distance function, quotient real equality, trichotomy, selected nearest points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SignedDistanceUp.TasteGate.SignedDistanceTasteGate single carrier alignment`

Theorem 2116.2 (SignedDistance NameCert obligations)**.** *For every accepted* $\mathsf{SignedDistance}^{\uparrow}$ *carrier*

$$S = (M, L, A, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDistance}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric distance row, located-source row, signed-side apartness row, dyadic budget, regular rational readback, real seal, transport, replay, provenance, and local naming rows;*

2. *located-boundary stability, requiring every transported or replayed signed-distance request to read the same metric row M, located-source row L, signed-side row A, dyadic budget D, and regular readback R before exposing the real seal E;*

3. *signed-budget exactness, requiring the sign threshold and rational tolerance in D to be the budget read both by the metric comparison in M and by the signed-side separation row A;*

4. *real-seal nonescape, requiring every real-facing or completion-facing consumer to pass through M, L, A, D, R, E and to receive no host distance function, quotient real representative, selected nearest point, trichotomy, or ambient completeness theorem;*

5. *ledger refusal, requiring H, C, P, N to transport, replay, source, and name only the displayed rows, with no Lean verification or bridge credential.*

No signed-distance NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2116.1. The packet displays exactly the metric distance row, the located-source row, the signed-side apartness row, the dyadic budget, the regular readback, the real seal, and the four structural rows.

Located-boundary stability is componentwise. A transported or replayed signed-distance request keeps the same metric row M and located row L, so the boundary being queried cannot be replaced by a host subset or by an unrecorded completion object.

Signed-budget exactness is the shared role of A and D. The signed-side row names the finite separation evidence, while the dyadic row fixes the tolerance and sign threshold read by the metric comparison and the regular rational observation.

For real-seal nonescape, the seal E appears only after M, L, A, D, R have been displayed. A real-facing consumer therefore receives a finite signed-distance route rather than a host distance function, quotient representative, nearest-point selector, trichotomy decision, or completeness theorem.

Finally, H, C, P, N only transport, replay, source, and name the carrier rows. Since the packet contains no Lean verification row and no bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 2116.3 (SignedDistance located boundary)**.** *Every accepted* $\mathsf{SignedDistance}^{\uparrow}$ *carrier*

$$S = (M, L, A, D, R, E, H, C, P, N)$$

binds its signed-distance read to the located boundary named by L*. Any consumer that reads the sign or magnitude surface must first expose the* $\mathsf{Met}^{\uparrow}$ *distance row* M*, the* $\mathsf{LocatedSpace}^{\uparrow}$ *source row* L*, the* $\mathsf{ApartnessReal}^{\uparrow}$ *signed-side row* A*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row* D*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. The exported boundary information is only this finite route and not a selected nearest point, a decidable side choice, a quotient real equality, or an ambient completeness claim.*

Proof. Start with an accepted packet and project its rows from Definition 2116.1. The only boundary-bearing coordinate is the located-source row L, while the metric and signed-side observations are carried by M and A.

The consumer route then opens the dyadic budget D. This budget fixes the finite tolerance and sign threshold used by both the metric comparison and the signed-side separation row.

The regular readback R is the only row that exposes rational observations downstream. It links the displayed metric and signed-side data to the terminal real seal, so no host comparison or hidden side decision can be inserted between the boundary row and the readback.

Thus every sign or magnitude read is bound to the displayed located boundary. The excluded nearest-point, decidable-side, quotient-equality, and ambient-completeness data cannot be exported because they do not occur in the carrier. □

Theorem 2116.4 (SignedDistance Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{SignedDistance}^{\uparrow}$ *carrier*

$$S = (M, L, A, D, R, E, H, C, P, N)$$

factors through the displayed route

$$M, L, A, D, R \longmapsto E.$$

The factorization exports the regular signed-distance seal E *only after the metric row, located-source row, signed-side row, dyadic budget, and regular readback have been displayed, and it exports no host real-valued distance function, quotient real equality, trichotomy, selected nearest point, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the accepted carrier. The real-facing coordinate is the seal E, and its replay row C displays a route whose source rows are M, L, A, D, R.

The metric row M supplies only finite distance observations, while the located row L supplies only the displayed boundary source. The signed-side row A and dyadic budget D then fix the finite sign and tolerance data read by the regular row R.

Only after these rows are visible can the route reach E. Componentwise transport in H, replay in C, provenance in P, and the local row N preserve the same order under classification.

Therefore the $\mathsf{Real}^{\uparrow}$-facing read factors through the displayed finite route. The excluded host function, quotient equality, trichotomy, nearest-point, completeness, Lean-verification, and bridge data have no carrier coordinate and cannot escape through the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDistance}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2116.1, Definition 2116.2, Definition 2116.3, and Definition 2116.4 seed the finite signed-distance route from metric and located boundary data through apartness, dyadic budget, and regular readback rows to the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{SignedDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host distance functions, quotient real equality, trichotomy, selected nearest points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-boundary stability, signed-budget exactness, real-seal nonescape, and ledger refusal for BEDC.Derived.SignedDistanceUp.

Constructive story: A SignedDistance$^{\uparrow}$ packet is generated from a metric distance row, a located-source row, an ApartnessReal signed-side row, a DyadicRatCore budget, a RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2117

Cauchy compact NameCert construction

Following Bishop-style compact metric arguments, the $\mathsf{CauchyCompact}^{\uparrow}$ packet records the finite certificate route by which explicit total boundedness and complete metric data turn a displayed regular Cauchy sequence window into an extracted limit seal. It sits over the compact packet of chapter 60, the compact metric finite-net surface of chapter 129, the complete metric limit rows of chapter 128, the totally bounded completion route of chapter 1573, and the L10 real-name spine of chapter 291, chapter 126, chapter 301, and chapter 38. The chapter names sequential compactness only in the constructive Cauchy-extraction sense supplied by displayed finite nets, regular rational windows, and completeness rows; it is not an open-cover compactness theorem and does not introduce a choice-selected subsequence principle.

Definition 2117.1 (CauchyCompact carrier)**.** *A* $\mathsf{CauchyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, L, T, S, D, E, H, C, P, N)$$

with the following visible rows:

1. *M is the compact metric row. It records the source metric carrier, classifier, finite compact-net admission data, and the compact-metric provenance read from the* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{Compact}^{\uparrow}$ *packets.*
2. *L is the complete metric row. It supplies the displayed limit-witness route for regular Cauchy point streams and the uniqueness route available under the local metric classifier.*
3. *T is the totally bounded finite-net schedule. For each displayed dyadic radius request it records the finite bundle of centers, coverage witnesses, mesh ledger, and replay route consumed by the extraction.*
4. *S is the regular Cauchy sequence window. It records the* $\mathsf{StreamName}^{\uparrow}$ *finite observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity readback, and the late-distance comparisons by which the sequence is admitted as a Cauchy stream in the metric row.*
5. *D is the dyadic radius ledger. It records every* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, refinement, comparison, and tail-budget read used by T and S.*
6. *E is the extracted limit seal. It is the* $\mathsf{Real}^{\uparrow}$*-facing and metric-facing row reached only after M, L, T, S, D have been displayed; it records the limit name, the convergence comparisons, and the local extraction certificate.*

7. *H records componentwise* hsame *transport for the compact metric row, complete metric row, totally bounded schedule, regular Cauchy window, dyadic radius ledger, extracted limit seal, replay, provenance, and local name.*

8. *C records displayed* Cont *replay of the route*

$$M, L, T, S, D \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{TotallyBoundedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, L, T, S, D, E, H, C, P, N$. It has no coordinate for open-cover compactness, a choice-selected subsequence, quotient stream equality, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompactUp.CauchyCompactTasteGate single carrier alignment`

Theorem 2117.2 (CauchyCompact NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompact}^{\uparrow}$ *carrier*

$$K = (M, L, T, S, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier projection for the compact metric row, complete metric row, totally bounded finite-net schedule, regular Cauchy sequence window, dyadic radius ledger, extracted limit seal, transport row, replay row, provenance row, and local name;*

2. *classifier exactness, requiring packet comparison to inspect only the displayed rows $M, L, T, S, D, E, H, C, P, N$ and to preserve their componentwise* hsame *transport;*

3. *Cauchy extraction, requiring every limit read to pass first through the finite-net schedule T, the regular Cauchy window S, the dyadic radius ledger D, and then the complete metric row L before the seal E is exposed;*

4. *transport stability, requiring any transported packet to replay the same compact metric source, total-bounded mesh ledger, complete-metric limit route, regular sequence window, dyadic comparisons, and extracted limit seal;*

5. *ledger confinement, requiring C, P, N to replay, source, and name only the displayed finite route and to export no open-cover compactness, choice-selected sequential compactness, quotient stream equality, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No $\mathsf{CauchyCompact}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier projection is immediate from Definition 2117.1. The accepted packet displays exactly the compact metric row, complete metric row, finite-net schedule, regular Cauchy window, dyadic ledger, extracted limit seal, transport, replay, provenance, and local naming rows.

Classifier exactness is componentwise. The classifier can compare two accepted packets only by reading $M, L, T, S, D, E, H, C, P, N$, and the transport row H records preservation of precisely those coordinates.

For Cauchy extraction, begin with a displayed tolerance request. The dyadic ledger D fixes the radius and tail budget, the total-bounded schedule T exposes the finite centers at that radius, and

the sequence window S supplies the regular Cauchy observations and late-distance comparisons. The complete metric row L is then applied to the same admitted stream route, producing the limit row sealed by E.

Transport stability and ledger confinement follow from H, C, P, N. These rows replay and name only the displayed route, so the excluded open-cover, choice, quotient, host-equality, trichotomy, and ambient real-completeness principles have no carrier coordinate from which they could be read. □

Theorem 2117.3 (CauchyCompact extraction scope)**.** *Let*

$$K = (M, L, T, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompact}^{\uparrow}$ *carrier. Every extracted limit read exported by K factors through the displayed rows*

$$\mathsf{CompleteMetric}^{\uparrow},\quad \mathsf{TotallyBounded}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow}$$

together with the structural rows BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. The extraction reads no open-cover compactness row, no choice-selected subsequence row, no quotient stream equality row, no host real equality row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier. The compact metric input is M, whose finite-net side is read through the $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{TotallyBounded}^{\uparrow}$ surfaces; no open-cover row is present in the carrier.

The sequential input is S together with D. These rows expose the $\mathsf{StreamName}^{\uparrow}$ finite observations, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity readback, and the $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger before any limit request can reach the completion side.

The limit step is L followed by E. The row L is the displayed $\mathsf{CompleteMetric}^{\uparrow}$ route for the same regular Cauchy stream, and E is only the resulting metric and $\mathsf{Real}^{\uparrow}$ seal for that route.

Finally H, C, P, N preserve, replay, source, and name the same dependencies. Since the tuple has no coordinate outside the displayed analytic and structural rows, every exported extraction remains confined to the stated scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompact}^{\uparrow}$ packet is generated from a compact metric row, a complete metric limit row, a totally bounded finite-net schedule, a regular Cauchy sequence window, a dyadic radius ledger, an extracted limit seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2117.1, Definition 2117.2, and Definition 2117.3 seed the finite constructive Cauchy-extraction route from compact metric data, complete metric limits, total-bounded finite nets, regular Cauchy windows, dyadic radii, and real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, choice-based sequential compactness, quotient stream equality, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier projection, classifier exactness, Cauchy extraction, transport stability, and ledger confinement for the $\mathsf{CauchyCompact}^{\uparrow}$ packet.

Chapter 2118

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$ packet names the finite located-comparison route for two regular Cauchy real names before any $\mathsf{Real}^{\uparrow}$ consumer reads a comparison result. It sits over the regular-rational source rows of chapter 301, the shared finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the located comparison surface of chapter 1730, and the Bishop located Cauchy real route of chapter 1850. The packet is a human-origin L10 handoff from regular Cauchy readback to located $\mathsf{Real}^{\uparrow}$ comparison; it is not quotient real equality, trichotomy, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2118.1 (Regular Cauchy located-comparison carrier)**.** *A* $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X_0, X_1, W, R_0, R_1, D, K, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. *X_0 and X_1 are the two regular Cauchy source-name rows. Each is read as a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *packet before either endpoint is sealed as a real name.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family. It records the finite observation lengths, synchronized cuts, endpoint windows, and comparison requests inspected for both sources.*
3. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from X_0, X_1 on the shared windows W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested scale, comparison slack, positive-gap threshold, endpoint enclosures, and exhausted finite windows.*
5. *K is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *evidence row. It reads only the displayed readbacks R_0, R_1, shared windows W, dyadic tolerances D, and the terminal seals E_0, E_1.*
6. *E_0 and E_1 are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows for the two regular Cauchy names. They are reachable only after X_0, X_1, W, R_0, R_1, D have been displayed.*
7. *H records componentwise* hsame *transport for the two source rows, shared windows, readback rows, dyadic tolerance ledger, located-comparison evidence, real seals, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$X_0, X_1, W, D \longmapsto R_0, R_1 \longmapsto E_0, E_1 \longmapsto K.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedComparison}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these thirteen rows. Its classifier compares accepted packets only through $X_0, X_1, W, R_0, R_1, D, K, E_0, E_1, H, C, P, N$ and has no coordinate for quotient real equality, host trichotomy, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedComparisonUp.RegularCauchyLocatedComparisonTasteGate`
`single carrier alignment`

Theorem 2118.2 (Regular Cauchy located-comparison NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$ *carrier*

$$L = (X_0, X_1, W, R_0, R_1, D, K, E_0, E_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy source rows, shared stream windows, two regular rational readbacks, dyadic tolerance ledger, located-comparison evidence row, two terminal real seals, transport, replay, provenance, and local naming row;*
2. *shared-window discipline, requiring the two readback rows R_0, R_1 to be read from the same window family W before any comparison evidence is exported;*
3. *dyadic tolerance discipline, requiring D to supply the requested scale, comparison slack, positive-gap threshold, endpoint enclosures, and exhaustion ledger consumed by the located comparison row;*
4. *real-seal discipline, requiring E_0 and E_1 to remain downstream of the displayed regular Cauchy readbacks and dyadic budget;*
5. *located-comparison exactness, requiring K to compare only the displayed R_0, R_1, W, D, E_0, E_1 route and to expose positive-gap or nonseparating evidence as finite located-comparison data;*
6. *transport and replay stability, requiring H and C to preserve the two source rows, window row, readback rows, tolerance row, comparison row, seal rows, provenance row, and local name row componentwise;*
7. *ledger exhaustion, excluding quotient real equality, host trichotomy, countable choice, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No regular Cauchy located-comparison obligation is stored outside these rows.

Proof. Project the accepted carrier L. The projection exposes exactly the two source rows, the common finite-window family, the two readback rows, the dyadic tolerance ledger, the located-comparison row, the two terminal real seals, and the structural rows.

The comparison route is finite. The shared windows W determine the inspected observations for both source rows, and the two readbacks R_0, R_1 are read at those same windows before any comparison evidence can be exported.

The dyadic ledger is the only tolerance input. It supplies the scale, slack, positive-gap threshold, endpoint enclosures, and exhaustion record that the located-comparison row is allowed to consume.

The real seals remain attached to the route. The rows E_0 and E_1 are read only after the regular Cauchy sources, windows, readbacks, and dyadic budget have been displayed, so the located comparison cannot appeal to detached real representatives.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for quotient real equality, trichotomy, choice, selected limits, completeness, Lean verification, or a bridge, the listed rows exhaust the local NameCert surface. □

Theorem 2118.3 (Regular Cauchy located-comparison real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or located-real consumer of an accepted* $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$ *carrier reads the comparison only through the finite handoff route*

$$X_0, X_1, W, D \longmapsto R_0, R_1 \longmapsto E_0, E_1 \longmapsto K \longmapsto H, C, P, N.$$

Thus the consumer receives two displayed regular Cauchy names, their shared StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledger, terminal RealUp seals, LocatedRealComparison evidence, componentwise transport, continuation replay, package provenance, and local naming. It receives no quotient real equality, host trichotomy, countable-choice schedule, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier. The source rows are X_0 and X_1, and the carrier definition requires both to pass through the common window family W and the dyadic tolerance ledger D.

The readback stage is R_0, R_1. These rows are the only regular rational observations supplied to the later real seals, so a consumer cannot introduce a second readback schedule or an unlisted tolerance.

The seal stage is E_0, E_1. Each seal remains downstream of the corresponding regular Cauchy readback and the shared dyadic budget, placing the comparison inside the displayed Bishop located Cauchy real route rather than inside a quotient-real carrier.

The comparison stage is K. It is the sole located-comparison evidence row and may only consume the displayed readbacks, windows, tolerances, and real seals.

The structural rows H, C, P, N preserve this route under componentwise transport, continuation replay, provenance, and local naming. Because no other coordinate is present, no $\mathsf{Real}^{\uparrow}$ or located-real consumer can read quotient equality, trichotomy, countable choice, a selected limit, or ambient completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedComparison}^{\uparrow}$ packet is generated from two regular Cauchy source-name rows, one shared StreamName window family, two RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a LocatedRealComparison evidence row, two RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2118.1, Definition 2118.2, and Definition 2118.3 seed the finite located-comparison handoff from regular Cauchy readbacks through shared StreamName windows, DyadicRatCore tolerances, RealUp seals, LocatedRealComparison evidence, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, readback exactness, tolerance discipline, Real seal nonescape, located-comparison handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedComparisonUp.

Chapter 2119

A Concrete Naming Certificate for $\mathsf{DyadicDivision}^{\uparrow}$

Following Bishop and Bridges constructive dyadic arithmetic, $\mathsf{DyadicDivision}^{\uparrow}$ names the finite quotient surface used when an exposed dyadic numerator is divided by a displayed nonzero dyadic denominator before regular rational and real-name consumers can read the result. It sits over the dyadic arithmetic carrier of chapter 291, the comparison and order rows of chapter 1623 and chapter 1597, the reciprocal-window budget of chapter 703, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet records division as displayed finite NameCert data; it is not quotient rational equality, a total inverse operation, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2119.1 (Dyadic division carrier)**.** *A* $\mathsf{DyadicDivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, L, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the numerator dyadic packet, read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic surface with its displayed sign, scale, endpoint, and tolerance data.*
2. *B is the nonzero denominator dyadic packet, also read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, whose denominator role is fixed before any reciprocal or quotient row is consumed.*
3. *L is the denominator apartness and bound row. It records the finite comparison and order evidence, supplied by* $\mathsf{DyadicComparison}^{\uparrow}$ *or* $\mathsf{DyadicOrder}^{\uparrow}$*, that keeps B away from the zero boundary and exposes a usable lower bound.*
4. *R is the reciprocal ledger. It is the* $\mathsf{CauchyInverseBudget}^{\uparrow}$*-compatible finite row that converts the displayed nonzero denominator packet and apartness bound into reciprocal-read data without exporting a total inverse operation.*
5. *T is the quotient readback row. It reads the numerator row A and reciprocal ledger R as one displayed dyadic quotient packet before any regular rational handoff is available.*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff together with the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It records the regular rational readback of T and the downstream real-name seal only after A, B, L, R, T have been displayed.*
7. *H records componentwise* hsame *transport for numerator, denominator, apartness, reciprocal, quotient, readback, seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from numerator and denominator inspection through apartness exposure, reciprocal ledger construction, quotient readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *sealing, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicDivision}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicComparison}^{\uparrow}$, $\mathsf{DyadicOrder}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDivision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the tuple $A, B, L, R, T, E, H, C, P, N$. It has no coordinate for quotient rational equality, a total inverse operation, trichotomy, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DyadicDivisionUp.DyadicDivisionTasteGate single carrier alignment`

Theorem 2119.2 (Dyadic division NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDivision}^{\uparrow}$ *carrier $Q = (A, B, L, R, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicDivision}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed numerator packet, nonzero denominator packet, denominator apartness and bound row, reciprocal ledger, quotient readback, regular rational handoff, real seal, transport, replay, provenance, and local naming rows;*
2. *dyadic-source exposure, requiring A and B to be read as* $\mathsf{DyadicRatCore}^{\uparrow}$ *packets before any quotient row is available;*
3. *denominator guard exposure, requiring L to display comparison or order evidence keeping B away from zero before R is consumed;*
4. *reciprocal-ledger locality, requiring R to use only the displayed denominator packet, apartness bound, and reciprocal-window budget named by* $\mathsf{CauchyInverseBudget}^{\uparrow}$*;*
5. *quotient readback exactness, requiring T to consume A and R together and to expose no independent quotient representative;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *non-escape, requiring E to remain downstream of A, B, L, R, T;*
7. *structural replay, requiring H, C, P, N to preserve, replay, source, and name exactly the same finite packet.*

No dyadic division obligation is stored outside these rows.

Proof. Project the carrier of Definition 2119.1. The analytic rows are A, B, L, R, T, E, and the structural rows H, C, P, N transport, replay, source, and name the same tuple.

The source exposure clauses are read from the sibling dyadic surfaces. The numerator and denominator rows are $\mathsf{DyadicRatCore}^{\uparrow}$ packets, while the guard row L is the finite comparison or order evidence supplied by chapter 1623 or chapter 1597.

The reciprocal clause is then local. Since R is downstream of B and L, it has the same guarded shape as the reciprocal budget of chapter 703; no route to R starts from an unguarded denominator.

The quotient row T consumes A and R together. Therefore the quotient read is a displayed dyadic row rather than a selected quotient representative or a host rational equality class.

Finally E is downstream of the quotient readback, and H, C, P, N add only transport, replay, provenance, and local naming. Because none of the excluded principles is a carrier coordinate, no local NameCert obligation can be stored outside $A, B, L, R, T, E, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.DyadicDivisionUp.DyadicDivisionCarrier namecert obligations`

Theorem 2119.3 (Dyadic division nonzero reciprocal route)**.** *Let* $Q = (A, B, L, R, T, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicDivision}^{\uparrow}$ *carrier. Every reciprocal ledger read used by the quotient row factors through*

$$B \longmapsto L \longmapsto R.$$

Thus a denominator packet reaches reciprocal data only after the displayed apartness and bound row has been exposed, and a route omitting L *cannot export a quotient readback.*

Proof. Begin with the denominator row B. Carrier admission places it before the apartness and reciprocal rows, so any denominator-facing reciprocal route starts from a displayed $\mathsf{DyadicRatCore}^{\uparrow}$ packet.

Next read the guard row L. It is the only coordinate that records denominator apartness and a usable lower bound, and it is the only place where the $\mathsf{DyadicComparison}^{\uparrow}$ or $\mathsf{DyadicOrder}^{\uparrow}$ evidence enters the packet.

Now project the reciprocal ledger R. Its construction is downstream of B and L and follows the reciprocal-window budget discipline named by chapter 703.

If L is omitted, the carrier has no row certifying that the denominator stays away from zero. The route therefore remains at the boundary surface and cannot reach the quotient readback row T. □

Theorem 2119.4 (Dyadic division Real readback nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *L10 Real, or Completion consumer of an accepted* $\mathsf{DyadicDivision}^{\uparrow}$ *carrier reads the displayed chain*

$$A, B, L \longmapsto R \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a finite dyadic quotient readback before the regular rational handoff or real seal is available, with no quotient rational equality, total inverse operation, trichotomy, selected representative, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness inserted into the route.*

Proof. Project the quotient row T. It can be consumed only after the numerator packet A and the guarded reciprocal ledger R have been displayed.

Use Definition 2119.3. The reciprocal ledger itself has already passed through the denominator packet B and the apartness row L, so the quotient route contains the complete nonzero-denominator guard.

The regular rational and real-facing row E is terminal relative to T. It hands the displayed quotient row to $\mathsf{RegSeqRat}^{\uparrow}$ and then to the $\mathsf{Real}^{\uparrow}$ seal; it has no second entrance from a host quotient or total inverse operation.

The remaining rows H, C, P, N transport, replay, source, and name the same chain. Since the carrier contains no excluded quotient, choice, trichotomy, or completeness coordinate, every L10 Real or Completion consumer remains inside the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDivision}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDivision}^{\uparrow}$ packet is generated from a numerator DyadicRatCore packet, a nonzero denominator DyadicRatCore packet, a denominator apartness and bound row, a reciprocal ledger, a quotient readback row, a RegSeqRat handoff with terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2119.1, Definition 2119.2, Definition 2119.3, and Definition 2119.4 seed the finite dyadic quotient surface for L10 Real and Completion consumers over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicComparison}^{\uparrow}$, $\mathsf{DyadicOrder}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, total inverse operations, trichotomy, selected representatives, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-source exposure, denominator guard exposure, reciprocal-ledger locality, quotient readback exactness, RegSeqRat and Real non-escape, structural replay, provenance, and local naming for BEDC.Derived.DyadicDivisionUp.

Chapter 2120

A Concrete Naming Certificate for $\mathsf{RegularCauchyAverage}^{\uparrow}$

Following Bishop and Bridges constructive Cauchy real arithmetic, the $\mathsf{RegularCauchyAverage}^{\uparrow}$ packet names finite barycentric averaging of regular rational Cauchy names by displayed dyadic weights, synchronized finite windows, finite summation, scalar normalization, and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits over the regular-rational source rows of chapter 301, the finite-window discipline of chapter 126, the dyadic arithmetic ledger of chapter 291, the pointwise summation surface of chapter 417, the normalization row of chapter 1971, the two-source specialization of chapter 1670, and the terminal real-name boundary of chapter 38. The packet supplies the finite average route needed by Real and Completion consumers; it does not assert quotient streams, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a detached completed-real barycenter.

Definition 2120.1 (RegularCauchyAverage carrier)**.** *A* $\mathsf{RegularCauchyAverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{avg}} = (I, R, W, D, B, S, K, M, E, H, C, P, N)$$

with the following displayed rows:

1. *I is a unary finite index row naming the displayed list of average inputs and the finite arity used by all downstream rows.*
2. *R is the indexed family of component* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family, one finite window for each index in I, all opened at the common precision request.*
4. *D is the family of* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows read from the sources R through the windows W.*
5. *B is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *barycentric weight ledger, recording the finite dyadic weights, their common denominator budget, and the displayed normalization total.*
6. *S is the iterated* $\mathsf{RegularCauchySum}^{\uparrow}$ *aggregation row, summing the weighted component rows in the order named by I.*
7. *K is the* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *normalization row, applying the displayed reciprocal or total-budget scalar from B to the aggregate row S.*
8. *M is an optional* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *two-source specialization row, present only when I displays two equal weights and absent from all other classifier comparisons.*

9. *E is the final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the sum and scalar route before the real-name boundary is opened.*

10. *H records componentwise* hsame *transport for the index row, source family, window family, dyadic observations, weight ledger, sum row, scalar row, optional midpoint row, readback row, replay row, provenance row, and local naming row.*

11. *C records displayed* Cont *replay from every average-facing precision request through $I, R, W, D, B, S, K, M, E, H,$*

12. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAverage}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyScalar}^{\uparrow}$, $\mathsf{RegularCauchyMidpoint}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAverage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, R, W, D, B, S, K, M, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAverageUp.RegularCauchyAverageTasteGate single carrier alignment`

Theorem 2120.2 (RegularCauchyAverage NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAverage}^{\uparrow}$ *carrier*

$$A_{\mathrm{avg}} = (I, R, W, D, B, S, K, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyAverage}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unary finite index row, component* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations and weights,* $\mathsf{RegularCauchySum}^{\uparrow}$ *aggregation row,* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *normalization row, optional* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *specialization row, final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal boundary, and structural rows;*

2. *tuple-window coverage, requiring every indexed component requested by I to be opened through the synchronized finite window family W before the aggregate row is available;*

3. *dyadic weight normalization, requiring B to display the finite weights, the common denominator budget, and the normalization total used by the scalar row;*

4. *sum and scalar budget exactness, requiring S to consume exactly the weighted component rows and K to apply exactly the normalization budget recorded by B;*

5. *readback stability, requiring the final* $\mathsf{RegSeqRat}^{\uparrow}$ *row E to transport through H together with the same index, source, window, dyadic, sum, scalar, optional midpoint, replay, provenance, and naming data;*

6. *Real seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to pass through the final readback row E and its displayed provenance before any real-name seal is exposed;*

7. *provenance and local naming, requiring P and N to name exactly the finite barycentric packet and no external completed-real average.*

No average obligation is stored outside these finite rows, and the surface refuses quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local data.*

Proof. Project the accepted carrier. The analytic coordinates are the finite index row I, the source family R, the synchronized windows W, the dyadic observations D, the barycentric weight ledger

B, the sum row S, the scalar normalization row K, the optional midpoint row M, and the readback row E.

Read the tuple windows before any aggregate is visible. The row I fixes the finite arity, and W opens exactly the component windows used to read D from the $\mathsf{RegSeqRat}^{\uparrow}$ source family R.

Next read the weight and budget rows. The ledger B supplies the dyadic weights, the common denominator budget, and the displayed normalization total. The aggregation row S then consumes the weighted component rows through the $\mathsf{RegularCauchySum}^{\uparrow}$ surface, and the scalar row K applies the normalization through $\mathsf{RegularCauchyScalar}^{\uparrow}$.

The two-source equal-weight case is only a specialization. When M is present, it is the same displayed route through the sibling $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ packet; when it is absent, no classifier coordinate is created for a separate completed-real midpoint.

Finally replay and transport the readback through C, H, P, N. The terminal real-name consumer can reach the seal only after E and its provenance are displayed, so no quotient stream, selected limit, countable-choice selector, trichotomy witness, host real equality, ambient completeness theorem, Lean verification, or bridge credential can be exported by the local NameCert surface. □

Theorem 2120.3 (RegularCauchyAverage barycentric handoff)**.** *Every average read exported by an accepted* $\mathsf{RegularCauchyAverage}^{\uparrow}$ *carrier factors through the finite route*

$$(I, R) \longmapsto W \longmapsto D \longmapsto B \longmapsto S \longmapsto K \longmapsto E$$

before it may be sealed as a $\mathsf{Real}^{\uparrow}$ *read. In particular, the read factors through* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyScalar}^{\uparrow}$ *before the* $\mathsf{Real}^{\uparrow}$ *seal; in the two-source equal-weight case the same route may also be displayed through the* $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ *specialization row. The handoff exports no quotient streams, selected limits, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or detached completed-real barycenter.*

Proof. Start with the accepted packet A_{avg}. The only component sources are the $\mathsf{RegSeqRat}^{\uparrow}$ rows named by R, and the only observations attached to them are opened through the synchronized $\mathsf{StreamName}^{\uparrow}$ windows W and the dyadic rows D.

The barycentric weights are displayed before aggregation. The ledger B fixes the dyadic weights and the normalization budget, so the row S can only be read as the finite $\mathsf{RegularCauchySum}^{\uparrow}$ aggregation of those weighted component rows.

Normalization is downstream of the aggregate. The row K is the $\mathsf{RegularCauchyScalar}^{\uparrow}$ route applying the displayed total-budget scalar to S, and the final $\mathsf{RegSeqRat}^{\uparrow}$ readback E is exposed only after this scalar row is present.

If the finite index has two equal weights, the midpoint row M gives the same consumer boundary through chapter 1670. It does not add a second route to a completed-real average.

The $\mathsf{Real}^{\uparrow}$ seal is therefore terminal. Componentwise transport, continuation replay, package provenance, and the local naming row preserve the route through I, R, W, D, B, S, K, E, leaving no carrier coordinate through which the refused principles or a detached completed-real barycenter could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAverage}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2120.1, Definition 2120.2, and Definition 2120.3 seed the finite barycentric-average surface over unary finite indices, component $\mathsf{RegSeqRat}^{\uparrow}$ rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ weights and budgets, $\mathsf{RegularCauchySum}^{\uparrow}$ aggregation, $\mathsf{RegularCauchyScalar}^{\uparrow}$ normalization, optional $\mathsf{RegularCauchyMidpoint}^{\uparrow}$ specialization, final readback, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAverage}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyAverageUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tuple-window coverage, dyadic weight normalization, sum and scalar budget exactness, readback stability, Real seal non-escape, provenance, and local naming for BEDC.Derived.RegularCauchyAverageUp.

Constructive story: A RegularCauchyAverage$^{\uparrow}$ packet is generated from a unary finite index row, component RegSeqRat sources, synchronized StreamName windows, DyadicRatCore weights and normalization budget, a RegularCauchySum aggregation row, a RegularCauchyScalar normalization row, an optional RegularCauchyMidpoint specialization row, a final RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2121

A Concrete Naming Certificate for $\mathsf{DugundjiExtension}^{\uparrow}$

Following Dugundji's extension theorem and the Kelley–Bourbaki topology tradition, the $\mathsf{DugundjiExtension}^{\uparrow}$ packet records only a finite extension-operator certificate: it names the route by which a closed metric subspace map into a convex target is extended through displayed affine coefficients, not a host-level selection of covers or functions. Its normality side is anchored in chapter 1866, its real-valued seed in chapter 1956, its exported map surface in chapter 130, its closed-domain exposure in chapter 1519, its affine target in chapter 208, its scalar seal in chapter 38, and its componentwise linear handoff in chapter 47. The packet therefore marks the L10 extension-operator boundary where a Tietze-style real extension is repacked as convex-valued continuous extension data while remaining inside finite BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Definition 2121.1 (Dugundji extension carrier)**.** *A* $\mathsf{DugundjiExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, S, V, T, L, G, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the ambient metric/topology source row. It records the* $\mathsf{NormalSpace}^{\uparrow}$ *and metric-topology admission data through which neighborhoods, continuity tests, and closed-set exposure are read.*
2. *S is the closed subspace row. It records the* $\mathsf{MetricSubspace}^{\uparrow}$ *inclusion, the closed-domain classifier, and the local restriction route for the partial map being extended.*
3. *V is the convex target row. It records the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier, the compatible* $\mathsf{VecSpace}^{\uparrow}$ *affine-combination surface, and the target classifier used by convex-valued reads.*
4. *T is the Tietze seed map row. It records the real-valued* $\mathsf{TietzeExtension}^{\uparrow}$ *route used to obtain scalar coordinate extensions before any convex-valued graph is exposed.*
5. *L is the local affine-coefficient ledger. It records the finite coefficient rows, index support, scalar* $\mathsf{Real}^{\uparrow}$ *seals, convex-sum budget, and compatibility with the target row V.*
6. *G is the extension graph row. It records the continuous graph read from ambient points through the coefficient ledger L and the convex target row V.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records the scalar readback used by the Tietze seed and by every displayed affine coefficient.*

8. *H records componentwise* hsame *transport for the ambient source, closed subspace, convex target, Tietze seed, affine-coefficient ledger, extension graph, real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay of every consumer route through*

$$A, S, V, T, L, R \longmapsto G.$$

10. *P is the* Pkg *provenance over* $\mathsf{DugundjiExtension}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{TietzeExtension}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, V, T, L, G, R, H, C, P, N$. It has no coordinate for a host choice function, a selected locally finite cover, function extensionality, quotient map equality, open-cover compactness, or an ambient extension theorem outside the displayed packet.

Lean4 ✓ `BEDC.Derived.DugundjiExtensionUp.DugundjiExtensionTasteGate single carrier alignment`

Theorem 2121.2 (Dugundji extension NameCert obligations)**.** *For every accepted* $\mathsf{DugundjiExtension}^{\uparrow}$ *carrier*

$$D = (A, S, V, T, L, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the ambient metric/topology source, closed subspace, convex target, Tietze seed map, affine-coefficient ledger, extension graph, real seal, transport, replay, provenance, and local name;*

2. *closed-subspace exposure, requiring every restriction read to pass through the* $\mathsf{MetricSubspace}^{\uparrow}$ *row S and the ambient source row A before it can compare the original partial map with the extension graph;*

3. *Tietze seed compatibility, requiring scalar reads in T and R to be the only real-valued extension data consumed by the affine-coefficient ledger L;*

4. *convex affine-combination handoff, requiring every value read of G to be assembled through the* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *rows in V using the displayed coefficient support in L;*

5. *continuous-map export, requiring the exported extension to pass through the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface named by G and replayed by C;*

6. *non-escape from host choice and function extensionality, requiring H, C, P, N to transport, replay, source, and name only the displayed finite rows and to export no selected cover, host choice function, function-extensional equality, quotient map equality, open-cover compactness, or ambient extension theorem.*

No Dugundji extension NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2121.1. The accepted packet displays exactly the ambient source row, closed subspace row, convex target row, Tietze seed row, affine-coefficient ledger, extension graph, real seal, transport, replay, provenance, and local naming rows.

Closed-subspace exposure follows from the order of the carrier rows. The original partial map can be compared with the extension graph only by reading S inside A, so no unrecorded host subset or external restriction operator can be inserted.

Tietze seed compatibility is confined to T and R. These rows provide the scalar extension and scalar seals consumed by L, and the coefficient ledger has no other real-valued input coordinate.

For the affine handoff, the target row V supplies the convex and vector-space operations, while L supplies the finite coefficient support and scalar budget. The graph row G is exposed only after those rows have been displayed.

Finally, H, C, P, N preserve, replay, source, and name the same displayed route. Since the packet contains no coordinate for a selected cover, host choice function, function-extensional equality, quotient map equality, open-cover compactness, or ambient extension theorem, none of those principles can escape through the local NameCert surface. □

Theorem 2121.3 (Dugundji extension affine handoff)**.** *Let*

$$D = (A, S, V, T, L, G, R, H, C, P, N)$$

be an accepted $\mathsf{DugundjiExtension}^{\uparrow}$ *carrier. Every extension read exported by D factors through the displayed route*

$$\mathsf{NormalSpace}^{\uparrow} \longrightarrow \mathsf{TietzeExtension}^{\uparrow} \longrightarrow \mathsf{ConvexSet}^{\uparrow}/\mathsf{VecSpace}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the local $\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}$ *row is exposed. Equivalently, every consumer of the extension graph first reads the normal ambient source A, the closed subspace S, the Tietze seed T, the convex and vector-space target V, the affine ledger L, and the real seal R, then reaches G, C, P, N. The exported data includes no host choice function, no function-extensional equality, no selected cover, no quotient map equality, and no open-cover compactness principle.*

Proof. Project the accepted carrier. The ambient and closed-domain side is A together with S, so the route begins with the normal metric/topology source and its displayed closed subspace.

The scalar seed is T together with the real seal R. Thus any coefficient read entering the affine ledger is already a displayed Tietze-style scalar extension read and not a host-level real-valued function.

The target handoff is V followed by L. The row V supplies the $\mathsf{ConvexSet}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ operations, while L supplies the finite coefficient support and the convex-sum budget used to assemble the value.

Only after these rows are displayed does the graph row G export a $\mathsf{ContinuousMap}^{\uparrow}$ read. Componentwise transport, replay, provenance, and the local row N preserve this order, so every extension read reaches the local NameCert through the stated factorization and through no excluded host principle. □

Theorem 2121.4 (Dugundji extension closed-subspace exposure)**.** *Let*

$$D = (A, S, T, V, L, G, R, H, C, P, N)$$

be an accepted $\mathsf{DugundjiExtension}^{\uparrow}$ *carrier, written in source-first order for the exposure read. Before the extension graph G is public, the packet must display the ambient source row A, the closed subspace row S, the Tietze seed row T, and the convex target row V. Thus every accepted extension read exposes the source, domain restriction, scalar seed, and convex target before it can compare or export the graph row.*

Proof. Project the carrier of Definition 2121.1. Its displayed rows include the ambient metric/topology source, closed subspace, convex target, Tietze seed, affine ledger, graph, real seal, transport, replay, provenance, and local name.

Apply the NameCert obligations of Definition 2121.2. Closed-subspace exposure passes through S inside A, Tietze seed compatibility passes through T, and the convex affine handoff passes through V before the graph row is exposed.

The remaining rows L, R, H, C, P, N preserve and name the same route. They do not supply a second source, a hidden closed-domain comparison, or a detached graph export, so G becomes public only after A, S, T, V have been displayed. □

Theorem 2121.5 (Dugundji extension convex affine non-escape)**.** *For every accepted* $\mathsf{DugundjiExtension}^{\uparrow}$ *carrier*

$$D = (A, S, T, V, L, G, R, H, C, P, N),$$

the affine ledger L and the $\mathsf{Real}^{\uparrow}$ *seal R keep the extension graph G inside displayed convex affine combinations over the target row V. The accepted route excludes host choice, selected covers, quotient map equality, and function extensionality as sources for the graph or for its value comparison.*

Proof. Use the affine handoff of Definition 2121.3. It forces every extension read to pass from the ambient and closed-domain rows through the Tietze seed, convex target, affine ledger, and real seal before reaching the exported graph.

Now apply the closed-subspace exposure theorem Definition 2121.4. Since A, S, T, V are already displayed before G is public, the ledger L can only assemble values from the exposed convex target and the scalar data sealed by R.

No excluded principle is a carrier coordinate. The rows H, C, P, N transport, replay, source, and name the finite packet, but they do not add host choice, selected covers, quotient map equality, or function extensionality to the accepted graph route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2121.1, Definition 2121.2, and Definition 2121.3 seed the finite extension-operator packet from normal ambient data, closed metric subspace exposure, Tietze scalar seeds, convex affine coefficients, continuous-map export, and real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DugundjiExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host-selected cover, host choice function, function-extensional equality, quotient map equality, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires BEDC.Derived.DugundjiExtensionUp to check carrier admission, closed-subspace exposure, Tietze seed compatibility, convex affine-combination handoff, continuous-map export, and non-escape from host choice and function extensionality.

Constructive story: A $\mathsf{DugundjiExtension}^{\uparrow}$ packet is generated from an ambient metric/topology source, a closed subspace row, a convex target row, a Tietze seed map, local affine-coefficient rows, an extension graph, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2122

A Concrete Naming Certificate for $\mathsf{RealIntervalDistance}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Moore–Kearfott–Cloud interval analysis, the $\mathsf{RealIntervalDistance}^{\uparrow}$ packet records the finite point-to-interval distance route by which a displayed real source point, a located real interval, a clamp value, and an absolute-value magnitude are read before any downstream Real or Completion consumer sees a distance. It consumes the clamping surface of chapter 1810, the absolute-value surface of chapter 1608, the located interval surface of chapter 1709, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet gives L10 Real and Completion routes a named finite handoff for distance from a point to a located interval; it is not host order, quotient real equality, a selected closest point theorem, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2122.1 (RealIntervalDistance carrier)**.** *A* $\mathsf{RealIntervalDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, I, K, A, T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source point row. It records the displayed* $\mathsf{Real}^{\uparrow}$ *name of the point whose distance from the interval is requested.*
2. *I is the source interval row. It records the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint names, locator reads, radius data, and locatedness budget for the interval being queried.*
3. *K is the clamp row. It records the* $\mathsf{RealClamping}^{\uparrow}$ *value obtained by clamping the point row X against the displayed interval row I.*
4. *A is the distance-magnitude row. It records the* $\mathsf{RealAbs}^{\uparrow}$ *read of the displayed difference between the source point and the clamp value.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records every dyadic radius, slack comparison, endpoint budget, and absolute-value precision request consumed by K and A.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite observations used to read the point, endpoints, clamp value, and distance magnitude at the requested tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational data exposed by the point, endpoint, clamp, and absolute-value streams.*

8. *H records componentwise* hsame *transport for the point row, interval row, clamp row, absolute-value row, tolerance ledger, stream windows, regular readback, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the route*

$$X, I \longmapsto K \longmapsto A \longmapsto T, S, R.$$

10. *P is the* Pkg *provenance over* $\mathsf{RealIntervalDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalDistance}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these eleven rows form one finite point-to-interval distance packet. It has no coordinate for host order, quotient real equality, a selected closest point, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealIntervalDistanceUp.RealIntervalDistanceTasteGate single carrier alignment`

Theorem 2122.2 (RealIntervalDistance NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalDistance}^{\uparrow}$ *carrier*

$$D = (X, I, K, A, T, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalDistance}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *point, source* $\mathsf{LocatedRealInterval}^{\uparrow}$ *interval,* $\mathsf{RealClamping}^{\uparrow}$ *clamp value,* $\mathsf{RealAbs}^{\uparrow}$ *distance magnitude, dyadic tolerance ledger, stream windows, regular rational readback, transport, replay, provenance, and local name;*

2. *interval exposure, requiring every endpoint, locator, radius, and locatedness request to pass through the displayed interval row I before the clamp row is read;*

3. *clamp handoff, requiring K to consume only the displayed point row X, interval row I, tolerance ledger T, stream windows S, and regular readback row R;*

4. *absolute-distance readback, requiring A to read the magnitude of the displayed difference between X and K through the same T, S, R rows before any distance consumer sees the result;*

5. *tolerance exactness, requiring the dyadic ledger T to record the endpoint, clamp, absolute-value, and consumer precision requests used in the packet;*

6. *transport and replay stability, requiring H and C to preserve the point, interval, clamp, absolute-value, tolerance, window, readback, provenance, and local naming rows componentwise;*

7. *non-escape from host trichotomy, quotient equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No point-to-interval distance obligation is stored outside these rows.

Proof. Project the accepted packet $D = (X, I, K, A, T, S, R, H, C, P, N)$. The projection exposes exactly the point, interval, clamp, absolute-value, tolerance, window, readback, transport, replay, provenance, and naming rows.

The interval and clamp stages are ordered by the displayed route. Endpoint, locator, radius, and locatedness requests begin at I, while the clamp row K is read only after the point row X, the interval row I, and the finite tolerance and readback rows T, S, R have been displayed.

The distance magnitude is downstream of the clamp row. The row A records the $\mathsf{RealAbs}^{\uparrow}$ read of the displayed difference between X and K, using the same dyadic ledger, stream windows, and regular rational readback that admitted the clamp request.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier coordinates stores host trichotomy, quotient real equality, a selected limit or closest point, countable choice, or ambient real completeness, no such principle can be exported by the local NameCert surface. □

Theorem 2122.3 (RealIntervalDistance clamp-absolute handoff)**.** *Every accepted* $\mathsf{RealIntervalDistance}^{\uparrow}$ *distance consumer factors through the displayed route*

$$\mathsf{Real}^{\uparrow}_{\mathrm{source}} \longmapsto \mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{RealClamping}^{\uparrow} \longmapsto \mathsf{RealAbs}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}/\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}$$

Thus the packet exports only the finite clamp-absolute distance handoff and exports no host order, quotient equality, selected closest point, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an accepted distance consumer. The only source point coordinate is X, and the only interval coordinate is I, so the consumer cannot read a distance before those two rows expose their finite real-name and located-interval data.

The clamp coordinate comes next. The row K is the displayed $\mathsf{RealClamping}^{\uparrow}$ output for the point and interval rows, under the precision and readback rows T, S, R.

The absolute-value coordinate supplies the distance magnitude. The row A reads the displayed difference between the source point and the clamp value, again through T, S, R, so the magnitude is tied to the same finite data as the clamp.

The structural rows close the route. Componentwise transport in H, continuation replay in C, provenance in P, and local naming in N preserve exactly the displayed rows, leaving no coordinate for the excluded host, quotient, selection, compactness, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalDistance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealIntervalDistance}^{\uparrow}$ packet is generated from a source RealUp point row, a LocatedRealIntervalUp interval row, a RealClampingUp clamp row, a RealAbsUp distance-magnitude row, a DyadicRatCoreUp tolerance ledger, StreamNameUp finite windows, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealIntervalDistance}^{\uparrow}}$, seed)

Scope closed: Definition 2122.1, Definition 2122.2, and Definition 2122.3 seed the finite point-to-interval distance route over displayed point, located interval, clamp, absolute-value, dyadic tolerance, stream-window, regular-readback, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealIntervalDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host order, quotient equality, selected closest points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exposure, clamp handoff, absolute-distance readback, tolerance exactness, transport stability, and replay confinement for the $\mathsf{RealIntervalDistance}^{\uparrow}$ packet.

Chapter 2123

A Concrete Naming Certificate for $\mathsf{CauchyOscillationCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyOscillationCriterion}^{\uparrow}$ packet records the finite identification between a Cauchy oscillation ledger and a two-point tail-diameter witness. It consumes the oscillation surface of chapter 448, the tail-diameter surface of chapter 1746, the criterion consumers of chapter 392 and chapter 435, the dyadic tolerance row of chapter 291, and the terminal seal discipline of chapter 38. The packet gives RegSeqRat, Real, and Completion consumers one finite criterion object rather than two unrelated tail tests; it is not a real-completeness theorem, a selected limit, quotient stream equality, host real equality, or a choice principle.

Definition 2123.1 (Cauchy oscillation criterion carrier)**.** *A* $\mathsf{CauchyOscillationCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (O, K, A, U, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the accepted* $\mathsf{CauchyOscillation}^{\uparrow}$ *ledger row, exposing the finite tail window, threshold schedule, dyadic tolerance, and oscillation comparisons read by the criterion.*
2. *K is the accepted* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *witness row, exposing the same source tail through one threshold, two visible tail indices, one tolerance, and the displayed diameter bound.*
3. *A is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *consumer row that reads the single-sequence criterion boundary from the shared ledger and diameter witness.*
4. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *family row that reads the same finite route when the criterion is presented as a singleton or fixed finite family.*
5. *Q is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose dyadic budget is used by both the oscillation comparisons in O and the two-point bound in K.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after the common finite criterion route has been accepted.*
7. *H records componentwise* hsame *transport for the oscillation ledger, tail-diameter witness, criterion row, uniform-family row, dyadic tolerance, real seal, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay of the route*

$$O, K, Q \longmapsto A, U \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyOscillationCriterion}^{\uparrow}$, $\mathsf{CauchyOscillation}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyOscillationCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, K, A, U, Q, R, H, C, P, N$. It has no coordinate for a quotient stream, selected limit, host equality, choice datum, ambient Real completeness, or detached completion object.

Lean4 ✓ `BEDC.Derived.CauchyOscillationCriterionUp.CauchyOscillationCriterionTasteGate single carrier alignment`

Theorem 2123.2 (Cauchy oscillation criterion NameCert obligations)**.** *For every accepted* CauchyOscillationCriteri *carrier $X = (O, K, A, U, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyOscillationCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the oscillation ledger, tail-diameter witness, single-sequence criterion consumer, finite-family criterion consumer, shared dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows;*

2. *oscillation-to-diameter projection, requiring every accepted oscillation comparison used by O to expose the same finite source tail, threshold discipline, and dyadic tolerance from which the two visible tail-index comparison in K is read;*

3. *diameter-to-oscillation reconstruction, requiring the two-point bound recorded by K to replay as a finite oscillation-ledger entry over the same displayed source tail and tolerance row;*

4. *consumer handoff through* $\mathsf{CauchyCriterion}^{\uparrow}$ *and* $\mathsf{UniformCauchyCriterion}^{\uparrow}$*, requiring both consumer rows A and U to read the shared O, K, Q route before exposing the terminal* $\mathsf{Real}^{\uparrow}$ *seal row R;*

5. *non-escape from quotient streams, selected limits, host equality, choice, Real completeness, and detached completion data.*

No Cauchy oscillation criterion obligation is stored outside these rows.

Proof. Project the accepted packet $X = (O, K, A, U, Q, R, H, C, P, N)$. The projection exposes exactly one oscillation ledger, one tail-diameter witness, two criterion consumer rows, one shared dyadic tolerance, one Real seal handoff, and the structural transport, replay, provenance, and naming rows.

For the oscillation-to-diameter direction, read the ledger row O through its finite tail window, threshold schedule, dyadic tolerance, and displayed comparisons. The tail-diameter row K uses the same source tail and tolerance but focuses the read to two visible tail indices beyond the displayed threshold, so every two-point comparison consumed by the criterion is a projection of the finite oscillation route.

For the diameter-to-oscillation direction, replay the two visible indices and bound stored in K as one ledger entry over the same source tail. The shared tolerance row Q fixes the dyadic budget, so the reconstructed ledger entry cannot alter the radius, threshold discipline, or source observations.

The consumer rows then read the common route. The $\mathsf{CauchyCriterion}^{\uparrow}$ row A and the $\mathsf{UniformCauchyCriterion}^{\uparrow}$ row U both consume O, K, Q before exposing R, and the replay row C records this order as a displayed continuation.

Finally, H, P, N transport, source, and name only the displayed finite rows. Since the carrier contains no coordinate for quotient streams, selected limits, host equality, choice, Real completeness, or detached completion data, none of those principles can be part of the local obligation surface. □

Theorem 2123.3 (Cauchy oscillation and tail-diameter equivalence)**.** *Every accepted* $\mathsf{CauchyOscillationCriterion}^{\uparrow}$ *packet makes the* $\mathsf{CauchyOscillation}^{\uparrow}$ *ledger and the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *witness the same finite tail-bound route for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers. More explicitly, a consumer that reads the criterion packet through A or U must factor through the common route*

$$O, K, Q \longmapsto A, U \longmapsto R,$$

where O and K expose the same displayed source tail, threshold discipline, dyadic tolerance, and finite two-point comparison data. The equivalence exports no Real completeness, selected limit, quotient stream equality, host equality, choice datum, or detached completion object.

Proof. Start with an accepted packet $X = (O, K, A, U, Q, R, H, C, P, N)$ and a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer route that reads the criterion through A or U. Carrier admission gives the consumer no analytic coordinates except the oscillation ledger O, the tail-diameter witness K, the shared tolerance row Q, and the terminal seal R.

Use the NameCert obligations in Definition 2123.2. The oscillation-to-diameter projection says that the tail-diameter read is obtained by selecting the two visible tail positions inside the finite oscillation ledger. The diameter-to-oscillation reconstruction says that the same two-point bound replays as an oscillation-ledger entry over the same displayed source tail and tolerance.

The consumer handoff fixes the downstream route. Both A and U must read the shared O, K, Q route before exposing R, so the single-sequence criterion and the finite-family criterion cannot distinguish the oscillation ledger from the tail-diameter witness as separate completion data.

The remaining structural rows preserve this identification. Componentwise transport in H and displayed replay in C keep the source tail, threshold discipline, tolerance row, and comparison data aligned, while P and N source and name the same finite packet.

Therefore the oscillation ledger and the tail-diameter witness are the same finite tail-bound route for the displayed RegSeqRat and Real consumers. The non-escape row excludes Real completeness, selected limits, quotient stream equality, host equality, choice, and detached completion objects from the equivalence. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyOscillationCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyOscillationCriterion}^{\uparrow}$ packet is generated from a CauchyOscillationUp ledger, a CauchyTailDiameterUp witness, CauchyCriterionUp and UniformCauchyCriterionUp consumer rows, a shared DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyOscillationCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 2123.1, Definition 2123.2, and Definition 2123.3 seed the finite criterion packet identifying the oscillation ledger with the two-point tail-diameter witness for RegSeqRat and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyOscillationCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, selected limits, quotient stream equality, host equality, choice, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, oscillation-to-diameter projection, diameter-to-oscillation reconstruction, consumer handoff, and non-escape for BEDC.Derived.CauchyOscillationCriterionUp.

Chapter 2124

A Concrete Naming Certificate for $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the metric-completion universal property is read here as a finite extension-and-uniqueness packet over displayed completion data. The packet refines the metric-completion surface of chapter 1506 by binding it to the universal-extension ledger of chapter 397, the regular Cauchy universal route of chapter 1425, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the real-name seal of chapter 38, and the continuous-map interface of chapter 130. It supplies later complete-metric and continuous-extension chapters with an L10 completion route that contains no quotient stream, selected host limit, countable-choice schedule, host function equality, or ambient completeness theorem.

Definition 2124.1 (MetricCompletionUniversalProperty carrier)**.** *A* $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, V, R, W, A, F, E, L, H, C, P, N)$$

with the following visible rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the completion-facing source packet;*
2. *V is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row that supplies extension existence and uniqueness ledgers over displayed completion data;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row used for dense metric observations and finite readback;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the dense source is inspected;*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read after R and W have displayed the requested approximation data;*
6. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *dense-source map row, including the graph and modulus data consumed by the extension route;*
7. *E is the extension ledger from F, R, W, A into a completion-facing* $\mathsf{ContinuousMap}^{\uparrow}$ *row over M;*
8. *L is the uniqueness ledger comparing two extension rows when they agree on the same M, V, R, W, A, F, E coordinates;*

9. *H records componentwise* hsame *transport for the completion row, universal-property row, dense source, window schedule, real seal, continuous-map row, extension ledger, uniqueness ledger, package row, and local naming row;*
10. *C records displayed* Cont *replay of the route*

$$M, V, R, W, A, F \longmapsto E \longmapsto L;$$

11. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionUniversalProperty}^{\uparrow}}$ *row.*

The carrier contains no quotient stream, no selected host limit, no countable-choice schedule, no host function equality, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.MetricCompletionUniversalPropertyUp.MetricCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Definition 2124.2 (MetricCompletionUniversalProperty classifier)**.** *Two accepted* MetricCompletionUniversalProper *carriers*

$$U = (M, V, R, W, A, F, E, L, H, C, P, N)$$

and

$$U' = (M', V', R', W', A', F', E', L', H', C', P', N')$$

classify together exactly when the classifier compares the displayed coordinates componentwise by hsame*, and when the displayed* Cont *replay has the same completion, dense-source, extension, and uniqueness order on both packets. Thus the classifier is the finite route*

$$M, V, R, W, A, F, E, L, H, C, P, N,$$

not equality of quotient completion classes, not host function extensionality, and not a selected-limit principle.

Theorem 2124.3 (MetricCompletionUniversalProperty extension route)**.** *Let $U = (M, V, R, W, A, F, E, L, H, C, P$ be an accepted* $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$ *carrier. Every completion-facing extension exported by U factors through the source completion row M, the universal-property row V, the dense regular rational row R, the finite stream-window schedule W, the terminal real-name seal A, the dense* $\mathsf{ContinuousMap}^{\uparrow}$ *row F, and only then the extension ledger E.*

The exported route is therefore

$$M, V, R, W, A, F \longmapsto E,$$

with the structural rows H, C, P, N recording transport, replay, provenance, and local naming. It is not a quotient-stream construction, not a selected host limit, not an appeal to countable choice, and not an ambient completeness theorem.

Proof. Project the accepted carrier of Definition 2124.1. The completion-facing source is M, and the universal-extension surface is V.

The dense source is read through R and W. The row R supplies regular rational readback, while W supplies the finite windows through which the requested observations are displayed.

The target real-facing boundary is A, and the dense map data are F. The extension ledger E is the only coordinate that repacks those rows as a completion-facing continuous-map read.

The remaining rows H, C, P, N transport, replay, package, and name the same finite route. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, host function equality, or ambient completeness, none of those principles can be exported by the extension route. □

Theorem 2124.4 (MetricCompletionUniversalProperty uniqueness ledger)**.** *Let* $U = (M, V, R, W, A, F, E, L, H, C,$ *and* $U' = (M, V, R, W, A, F, E', L', H', C', P, N')$ *be accepted* $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$ *carriers over the same completion row, universal-property row, dense source, stream-window schedule, real seal, dense continuous-map row, and provenance. If* E *and* E' *agree on every displayed finite read from* R *and* W *after the real-seal handoff through* A*, then the two exported extension rows are compared by the displayed uniqueness ledgers* L *and* L'*.*

The uniqueness conclusion is internal to $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. It is not host equality of maps and not equality of quotient completion representatives.*

Proof. Read the common coordinates from the two accepted packets. The rows M, V, R, W, A, F, P fix the completion source, universal-property surface, dense observations, real seal, dense map, and provenance shared by both extension ledgers.

The hypothesis compares E and E' on every finite read displayed by R and W. Transport those reads through H, H' and replay them through C, C' until both extension rows are presented over the same real-seal coordinate A.

The uniqueness coordinates are exactly L and L'. Applying them gives the required displayed hsame comparison of extension rows.

No carrier coordinate names host function extensionality, quotient equality, a selected limit, countable choice, or ambient completeness. The uniqueness claim therefore remains a ledger comparison over the visible BEDC rows. □

Theorem 2124.5 (MetricCompletionUniversalProperty NameCert obligations)**.** *For the carrier of Definition 2124.1, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier display for the packet* $U = (M, V, R, W, A, F, E, L, H, C, P, N)$*;*
2. *classifier stability by componentwise* hsame *transport and displayed* Cont *replay as in Definition 2124.2;*
3. *extension existence through the finite route of Definition 2124.3;*
4. *uniqueness through the ledger comparison of Definition 2124.4;*
5. *non-escape, requiring every consumer read to remain inside* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

No obligation is stored in a quotient stream, selected host limit, countable-choice schedule, host function equality row, or ambient completeness theorem.

Proof. Carrier display is projection from Definition 2124.1. The accepted packet lists exactly the completion row, universal-property row, dense source, window schedule, real seal, continuous-map row, extension ledger, uniqueness ledger, and structural support rows.

Classifier stability follows from Definition 2124.2. The comparison is componentwise on those displayed coordinates, and the replay data only preserve the same route order.

Extension existence is the obligation of Definition 2124.3. It forces every exported extension to pass through M, V, R, W, A, F before the ledger E is read.

Uniqueness is the obligation of Definition 2124.4. It compares candidate extension rows through L and L' over the same finite dense-source and real-seal reads.

The non-escape row is obtained by inspecting the carrier inventory. Since every exported object is one of the listed BEDC rows or a displayed route between them, the excluded host-side principles have no coordinate from which to enter the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$ packet is generated from a MetricCompletionUp source row, a CompletionUniversalPropertyUp extension row, RegSeqRat and StreamName dense-source reads, a terminal RealUp seal, a ContinuousMapUp dense-map row, extension and uniqueness ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2124.1, Definition 2124.2, Definition 2124.3, Definition 2124.4, and Definition 2124.5 seed the finite metric-completion universal-property route over displayed completion, dense-source, real-seal, continuous-map, extension, and uniqueness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, host function equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier display, classifier stability, extension route, uniqueness ledger, and NameCert non-escape for BEDC.Derived.MetricCompletionUniversalPropertyUp.

Chapter 2125

A Concrete Naming Certificate for $\mathsf{AbelPartialSummation}^{\uparrow}$

Following Abel and Hardy, the $\mathsf{AbelPartialSummation}^{\uparrow}$ packet records the finite summation-by-parts route by which a real series source, a finite coefficient window, and a displayed coefficient-difference ledger determine every endpoint-facing partial-summation read. It sits after the $\mathsf{RealSeries}^{\uparrow}$ carrier of chapter 1581, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite summation-by-parts evidence only through displayed partial sums, adjacent coefficient differences, and boundary terms; it does not import a host infinite sum, a convergence theorem for Abel means, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2125.1 (Abel partial summation carrier)**.** *An* $\mathsf{AbelPartialSummation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, S, \Delta, B, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealSeries}^{\uparrow}$ *source row. It exposes the real-valued term stream only through the finite partial-sum surface already named for real series.*
2. *A is the finite coefficient window. It records the inspected coefficients $a_m, \ldots, a_n$ and the endpoint coefficients used by the summation-by-parts formula.*
3. *S is the finite partial-sum window. It records the displayed partial sums of the source row over the same index interval and the shifted partial sums needed by the coefficient differences.*
4. *Δ is the coefficient-difference ledger. It records every adjacent difference $a_k - a_{k+1}$ used inside the finite summation-by-parts read.*
5. *B is the boundary-term ledger. It records the two endpoint products between displayed coefficients and displayed partial sums, together with the sign and index orientation used by the read.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family for the inspected source terms, coefficients, partial sums, coefficient differences, and boundary terms.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same displayed windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for finite addition, subtraction, multiplication, difference, and boundary-term comparisons.*

9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $R, A, S, \Delta, B, W, Q, D$ have been displayed.*

10. *H records componentwise* hsame *transport for the real series source, coefficient window, partial-sum window, difference ledger, boundary ledger, stream windows, readback, tolerance ledger, endpoint seal, replay, provenance, and local name.*

11. *C records displayed* Cont *replay of the route*

$$R, A, S, \Delta, B, W, Q, D \longmapsto E.$$

12. *P is the* Pkg *provenance over* $\mathsf{AbelPartialSummation}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelPartialSummation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, S, \Delta, B, W, Q, D, E, H, C, P, N$. It has no coordinate for a host infinite sum, an Abel convergence theorem, a hidden coefficient limit, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AbelPartialSummationUp.AbelPartialSummationTasteGate single carrier alignment`

Theorem 2125.2 (Abel partial summation NameCert obligations)**.** *For every accepted* $\mathsf{AbelPartialSummation}^{\uparrow}$ *carrier*

$$K = (R, A, S, \Delta, B, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelPartialSummation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real series source, coefficient window, partial-sum window, coefficient-difference ledger, boundary-term ledger, stream windows, readback, dyadic ledger, endpoint seal, transport, replay, provenance, and local naming rows;*

2. *partial-sum exposure, requiring every series read used by the packet to pass through R and the displayed finite partial-sum window S;*

3. *coefficient-difference stability, requiring Δ to store the adjacent differences read from A and to preserve them under the componentwise* hsame *transport row H;*

4. *boundary ledger accounting, requiring B to display the endpoint products and their signs before any endpoint seal can be exported;*

5. *endpoint non-escape, requiring E to remain downstream of $R, A, S, \Delta, B, W, Q, D$ and to export no host infinite sum, Abel convergence theorem, hidden coefficient limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*

6. *provenance closure, requiring C, P, N to replay, source, and name only the displayed finite route.*

No Abel partial-summation obligation is stored outside these rows.

Proof. Project the accepted packet $K = (R, A, S, \Delta, B, W, Q, D, E, H, C, P, N)$. The carrier exposes exactly the real series source, coefficient window, finite partial sums, coefficient differences, boundary terms, window and readback rows, dyadic ledger, endpoint seal, and structural rows.

The partial-sum exposure is forced by R and S. The source row supplies only the finite series surface of $\mathsf{RealSeries}^{\uparrow}$, and the partial-sum window lists the displayed sums available for the summation-by-parts read.

The coefficient side is finite. The row A displays the inspected coefficients, while Δ records the adjacent differences and B records the endpoint products with their signs and index orientation. The transport row H preserves those same displayed coordinates.

The rows W, Q, D read the source, coefficient, difference, and boundary data through finite windows, regular rational readback, and dyadic tolerance accounting. Only after those rows are present does E expose a $\mathsf{Real}^{\uparrow}$ endpoint seal.

Finally C, P, N replay, source, and name the same finite route. Since the carrier has no excluded coordinate for host summation, Abel convergence, hidden coefficient limits, quotient equality, choice, trichotomy, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2125.3 (Abel partial summation by-parts route)**.** *Every accepted* $\mathsf{AbelPartialSummation}^{\uparrow}$ *packet factors each summation-by-parts read through the displayed finite route*

$$R, A \longmapsto S, \Delta, B \longmapsto W, Q, D \longmapsto E, H, C, P, N.$$

Thus the read reaches the $\mathsf{Real}^{\uparrow}$ *endpoint seal only after the finite partial sums, coefficient-difference ledger, and boundary-term ledger have been displayed. It does not read a host infinite sum, an Abel convergence theorem, a hidden coefficient limit, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an endpoint-facing summation-by-parts request. By Definition 2125.1, the request first projects the source series row R and the coefficient window A.

The finite by-parts data are then fixed. The row S supplies the partial sums of the source row over the displayed interval, Δ supplies the adjacent coefficient differences, and B supplies the endpoint products that close the finite identity.

The readback stage is shared. The rows W, Q, D expose exactly the finite windows, regular rational reads, and dyadic tolerance comparisons needed to combine the displayed partial sums, differences, and boundary terms.

Only after this finite route is displayed can E provide the endpoint seal, and the structural rows H, C, P, N transport, replay, source, and name that same route. Hence every accepted packet factors the summation-by-parts read through the displayed finite ledgers before any $\mathsf{Real}^{\uparrow}$ endpoint seal is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbelPartialSummation}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AbelPartialSummation}^{\uparrow}$ packet is generated from a RealSeriesUp source, a finite coefficient window, a finite partial-sum window, a coefficient-difference ledger, a boundary-term ledger, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AbelPartialSummation}^{\uparrow}}$, seed)

Scope closed: Definition 2125.1, Definition 2125.2, and Definition 2125.3 seed the finite summation-by-parts certificate surface between RealSeriesUp partial sums, coefficient differences, boundary ledgers, and RealUp endpoint seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AbelPartialSummation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite sums, Abel convergence theorems, hidden coefficient limits, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum exposure, coefficient-difference stability, boundary ledger accounting, endpoint non-escape, and provenance closure for BEDC.Derived.AbelPartialSummationUp.

Chapter 2126

A Concrete Naming Certificate for $\mathsf{SilvermanToeplitz}^{\uparrow}$

Following Silverman, Toeplitz, and Hardy on regular matrix summability, the $\mathsf{SilvermanToeplitz}^{\uparrow}$ packet records the finite BEDC certificate by which a row-finite inspection of a summability matrix sends a convergent regular real-name sequence to the same $\mathsf{Real}^{\uparrow}$ endpoint. It refines the matrix-window route of chapter 1682, contains the Cesaro averaging and convergence special cases of chapter 1765 and chapter 1827, reads source and transformed rows through chapter 301, keeps all rational coefficient budgets inside chapter 291, and reaches the terminal seal only through chapter 38. The chapter names the local certificate surface for regular matrices; it does not import a host summability theorem, a completed infinite matrix, quotient equality of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2126.1 (Silverman–Toeplitz carrier)**.** *A* $\mathsf{SilvermanToeplitz}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, B, V, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the matrix coefficient-window row, listing the finite rows, finite column windows, coefficient approximants, truncation thresholds, and row-index requests currently inspected.*
2. *B is the row-sum bound ledger, recording the dyadic bounds used to keep each displayed matrix row uniformly bounded before a transformed read is exposed.*
3. *V is the column-vanishing ledger, recording for each displayed source tolerance the finite column requests whose coefficients are certified small along sufficiently late matrix rows.*
4. *W is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, listing only the finite input-sequence observations demanded by M, B, V.*
5. *T is the transformed finite-row surface, obtained from the displayed coefficients in M, the selected source windows in W, and the dyadic budgets in B, V.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for both the source windows used by the regularity argument and the transformed rows exposed by T.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the matrix windows, bounded-row ledger, column-vanishing ledger, source windows, transformed finite rows, and regular readback have been displayed.*

8. *H stores componentwise* hsame *transport for matrix windows, row-sum bounds, column-vanishing data, source windows, transformed rows, regular readback, real seals, continuation rows, provenance, and local naming.*

9. *C stores displayed* Cont *replay from matrix-row inspection to source-window selection, from row-sum and column-vanishing ledgers to transformed-row construction, from transformed rows to regular readback, and from regular readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SilvermanToeplitz}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SilvermanToeplitz}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, V, W, T, R, E, H, C, P, N$. It has no coordinate for a host summability method, a completed infinite matrix, a selected external limit, quotient equality of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SilvermanToeplitzUp.SilvermanToeplitzTasteGate single carrier alignment`

Theorem 2126.2 (Silverman–Toeplitz NameCert obligations)**.** *For every accepted* $\mathsf{SilvermanToeplitz}^{\uparrow}$ *carrier $S = (M, B, V, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SilvermanToeplitz}^{\uparrow}}$ *surface consists exactly of the following six finite obligation rows:*

1. *carrier admission for the displayed matrix coefficient windows, row-sum bound ledger, column-vanishing ledger, source windows, transformed finite rows, regular readback, real seal, transport, continuation, provenance, and naming rows;*

2. *bounded-row exactness, requiring every row-sum bound used by the transform to be read from B after the corresponding coefficient windows in M have been displayed;*

3. *column-vanishing exactness, requiring every small-column claim to pass through the displayed column ledger V and the same finite matrix-window row M;*

4. *source-to-transform construction, requiring every transformed row to be computed from M, B, V, W before the transformed read reaches T;*

5. *regular readback and real sealing, requiring every completion-facing transformed read to pass through R and then E;*

6. *transport, continuation, provenance, and local naming, requiring H, C, P, N to preserve the same finite route without adding a host summability or ambient completeness coordinate.*

Proof. Project the carrier of Definition 2126.1. Carrier admission is exactly the displayed inventory $M, B, V, W, T, R, E, H, C, P, N$, so the local certificate has no detached coordinate for an infinite matrix or a host summability method.

The bounded-row and column-vanishing clauses are read from B and V only after the inspected coefficient windows in M have been displayed. The selected input observations are the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, and the transformed finite row T is downstream of those same rows.

The regularity handoff is then forced through R, and the endpoint-facing route continues from R to the $\mathsf{Real}^{\uparrow}$ seal E. The structural rows H, C, P, N transport, replay, source, and name this route, and none of them contains an additional analytic principle. □

Theorem 2126.3 (Silverman–Toeplitz regularity route)**.** *Every accepted* $\mathsf{SilvermanToeplitz}^{\uparrow}$ *read that claims preservation of a regular real-name limit factors through the finite route*

$$M \longmapsto (B, V) \longmapsto W \longmapsto T \longmapsto R \longmapsto E \longmapsto (H, C, P, N).$$

The $\mathsf{ToeplitzLemma}^{\uparrow}$ *route supplies the finite matrix-window discipline, the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *and* $\mathsf{CesaroConvergence}^{\uparrow}$ *rows are admitted as special provenance surfaces, and no consumer can bypass* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or the* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with a preservation read from an accepted carrier. The only coefficient data visible to the classifier are the finite windows in M, and the regularity hypotheses for the matrix are represented by the row-sum ledger B and the column-vanishing ledger V.

The source sequence is then inspected only through W. Hence the transformed row cannot be formed before the coefficient, boundedness, vanishing, and source-window data have all been displayed, which forces the route through T.

The transformed sequence reaches any regular real-name consumer through R, and the real endpoint is visible only through E. Provenance may point to the Toeplitz, Hardy–Cesaro, and Cesaro-convergence sibling chapters, but those pointers are package rows in P and do not create a second path around M, B, V, W, T, R, E.

Finally, H, C, P, N preserve and name the same route. Since the carrier has no completed matrix, selected external limit, quotient real-name equality, countable choice, or ambient completeness field, the displayed route exhausts the preservation surface. □

Theorem 2126.4 (Silverman–Toeplitz finite-row regularity)**.** *For every accepted* $\mathsf{SilvermanToeplitz}^{\uparrow}$ *carrier* $S = (M, B, V, W, T, R, E, H, C, P, N)$*, fix a finite transformed row whose coefficient window is displayed in* M*, whose row-sum bound is displayed in* B*, whose column-vanishing ledger is displayed in* V*, whose source window is displayed in* W*, and whose dyadic error budget* D *is read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *provenance in* P*. The row constructed in* T *remains a regular* $\mathsf{RegSeqRat}^{\uparrow}$ *read before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is inspected. Its certificate passes through the sibling anchors* $\mathsf{ToeplitzLemma}^{\uparrow}$*,* $\mathsf{HardyCesaroMean}^{\uparrow}$*, and* $\mathsf{CesaroConvergence}^{\uparrow}$*, but it does not use a host summability theorem or a completed infinite matrix.*

Proof. Induct on the displayed finite matrix rows in M. In the base row, the row-sum ledger B supplies the dyadic bound for the inspected coefficient window, while W supplies exactly the source observations used by that row. The constructed value is therefore a finite rational combination accounted for by D and recorded in T.

For the step, append one coefficient window from M. The row-sum bound remains in B, the small-column evidence remains in V, and the accumulated dyadic error budget is preserved by the $\mathsf{DyadicRatCore}^{\uparrow}$ accounting recorded in P. Thus the enlarged transformed row is still exposed as a regular rational readback candidate.

Apply Definition 2126.3. The finite-row construction is forced through $T \longmapsto R \longmapsto E$, so the read reaches $\mathsf{RegSeqRat}^{\uparrow}$ before any $\mathsf{Real}^{\uparrow}$ seal can be inspected. The $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, and $\mathsf{CesaroConvergence}^{\uparrow}$ entries in P are sibling anchors for this route, not external analytic principles. □

Theorem 2126.5 (Silverman–Toeplitz Real-seal nonescape)**.** *For every accepted* $\mathsf{SilvermanToeplitz}^{\uparrow}$ *carrier, any* $\mathsf{Real}^{\uparrow}$ *endpoint exported by the regular matrix transform is sealed only after the matrix-window rows, row-sum ledger, column-vanishing ledger, source windows, transformed finite rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been exposed. Consequently the Real seal cannot be used to recover a host summability theorem, a detached infinite matrix, a selected source limit, quotient stream equality, countable choice, or ambient completeness.*

Proof. By Definition 2126.1, the seal row is E, and it is downstream of M, B, V, W, T, R. Thus an endpoint-facing consumer has already passed through the finite coefficient windows, the two matrix regularity ledgers, the source windows, the transformed row, and the regular readback.

The remaining rows H, C, P, N only transport, replay, source, and name the same packet. They do not store host analytic data, and the classifier has no coordinate for any excluded principle.

Therefore every exported $\mathsf{Real}^{\uparrow}$ endpoint is a seal for the displayed finite BEDC route rather than a route to host summability, detached infinite matrix data, selected limits, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{SilvermanToeplitz}^{\uparrow}$ preservation read factors through finite matrix coefficient windows, a row-sum bound ledger, a column-vanishing ledger, source StreamName windows, transformed finite rows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: toeplitzlemma, hardycesaromean, cesaroconvergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{SilvermanToeplitz}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries matrix coefficient windows, row-sum bounds, column-vanishing data, source windows, transformed rows, regular readback, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SilvermanToeplitz}^{\uparrow}}$.

Constructive story: A $\mathsf{SilvermanToeplitz}^{\uparrow}$ packet is generated from finite matrix coefficient windows, a row-sum bound ledger, a column-vanishing ledger, source StreamName windows, transformed finite rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SilvermanToeplitz}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2126.1, Definition 2126.2, Definition 2126.3, and Definition 2126.5 seed the regular matrix summability certificate surface from finite matrix windows to Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SilvermanToeplitz}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability theorems, completed infinite matrices, selected source limits, quotient equality of real-name streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-row exactness, column-vanishing exactness, source-to-transform construction, regular readback, Real-seal nonescape, transport and provenance, and TasteGate field faithfulness for BEDC.Derived.SilvermanToeplitzUp.

Chapter 2127

A Concrete Naming Certificate for $\mathsf{SeriesTermTest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SeriesTermTest}^{\uparrow}$ packet records the finite necessary condition for a real series endpoint: adjacent partial-sum differences are displayed term rows that must factor through the same Cauchy-tail threshold before a zero $\mathsf{Real}^{\uparrow}$ seal is read. It sits after the $\mathsf{RealSeries}^{\uparrow}$ carrier of chapter 1581, the real-series Cauchy criterion of chapter 2014, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names only finite displayed term rows, adjacent partial-sum differences, shared tail-threshold evidence, and a zero-seal read; it does not import a host infinite sum, a selected limit, quotient equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2127.1 (Series term test carrier)**.** *A* $\mathsf{SeriesTermTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, A, D, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *T is the real-valued term stream, exposed only through the inspected term rows already available to* $\mathsf{RealSeries}^{\uparrow}$*.*
2. *S is the partial-sum window, listing the displayed partial sums whose adjacent differences are inspected by the current tail request.*
3. *A is the adjacent-difference row, recording the equality of each inspected term row with the corresponding displayed difference between successive partial sums.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, assigning the same tail request a finite threshold and the rational radii used to compare adjacent partial sums.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected terms, partial sums, and adjacent differences.*
6. *Z is the zero* $\mathsf{Real}^{\uparrow}$ *seal reached only after T, S, A, D, R have been displayed through the common Cauchy-tail threshold.*
7. *H stores componentwise* hsame *transport for the term stream, partial-sum window, adjacent-difference row, dyadic ledger, readback, zero seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay of the route*

$$T, S, A, D, R \longmapsto Z.$$

9. *P is the* Pkg *provenance over* $\mathsf{SeriesTermTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeriesTermTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, A, D, R, Z, H, C, P, N$. It has no coordinate for a host infinite sum, selected limit, quotient equality of real streams, countable-choice selector, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeriesTermTestUp.SeriesTermTestTasteGate single carrier alignment`

Theorem 2127.2 (Series term test NameCert obligations)**.** *For every accepted* $\mathsf{SeriesTermTest}^{\uparrow}$ *carrier*

$$K = (T, S, A, D, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeriesTermTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the term stream, partial-sum window, adjacent-difference row, dyadic tolerance ledger, regular rational readback, zero* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *term-row exposure, requiring every inspected term to pass through T and the finite stream windows consumed by the source* $\mathsf{RealSeries}^{\uparrow}$ *row;*
3. *adjacent-difference exactness, requiring A to identify each inspected term row with the displayed difference of neighboring partial sums in S;*
4. *shared tail-threshold routing, requiring the same Cauchy-tail threshold used by* $\mathsf{RealSeriesCauchyCriterion}^{\uparrow}$ *to govern the dyadic ledger D before any zero seal is exported;*
5. *readback and zero-seal nonescape, requiring R and then Z to remain downstream of T, S, A, D;*
6. *provenance closure, requiring H, C, P, N to transport, replay, source, and name only the displayed finite route.*

No series term-test obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (T, S, A, D, R, Z, H, C, P, N)$. The carrier exposes exactly the term stream, partial-sum window, adjacent-difference row, dyadic ledger, readback, zero seal, and structural rows.

The term-row clause is forced by T and S. The source row supplies only displayed terms, while the partial-sum window supplies the two neighboring partial sums needed to read each inspected adjacent difference.

The adjacent-difference row A records the finite equality between the inspected term and the displayed partial-sum difference. The dyadic ledger D attaches those same rows to the shared Cauchy-tail threshold rather than to a separate host convergence assertion.

The readback and zero-seal stage is downstream. The row R reads the displayed term, partial sums, and difference through regular rational data, and Z is available only after that readback has passed through D.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate for host summation, selected limits, quotient equality, choice, trichotomy, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2127.3 (Series term test adjacent-difference route)**.** *Every accepted* $\mathsf{SeriesTermTest}^{\uparrow}$ *packet factors each inspected term read through the displayed finite route*

$$T, S \longmapsto A \longmapsto D, R \longmapsto Z, H, C, P, N.$$

Thus a term row reaches the zero $\mathsf{Real}^{\uparrow}$ *seal only as an adjacent partial-sum difference governed by the shared Cauchy-tail threshold. It does not read a host infinite sum, selected limit, quotient equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an inspected term request. By Definition 2127.1, the request first projects the term stream row T and the finite partial-sum window S.

The adjacent-difference row A then identifies the inspected term with the displayed difference between neighboring partial sums in S. This is a finite row calculation, not an appeal to a completed infinite series.

The shared threshold is read through D. The same tail request that makes the partial sums Cauchy supplies the dyadic tolerance data for the adjacent difference, and the regular rational readback R records the displayed comparison.

Only after this route is displayed can Z provide the zero $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so every accepted term read factors through the adjacent-difference surface before the zero seal is available. □

Theorem 2127.4 (Series term test zero-seal nonescape)**.** *Every zero-seal read from an accepted* $\mathsf{SeriesTermTest}^{\uparrow}$ *carrier factors through*

$$T, S, A, D, R, Z, H, C, P, N.$$

Consequently the packet exports only the necessary-condition statement that displayed terms tend to zero through the same Cauchy-tail threshold used by the partial sums. It exports no host infinite sum, selected limit, quotient equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 2127.3 to place every inspected term on the finite route through T, S, A, D, R. The route already contains the displayed neighboring partial sums and the dyadic threshold data attached to the Cauchy-tail request.

The zero seal Z consumes this route. It is a $\mathsf{Real}^{\uparrow}$ seal for the displayed term rows tending to zero, not a separate completion principle for the whole sequence of partial sums.

The transport, replay, provenance, and local naming rows H, C, P, N preserve the same coordinates. Because the carrier has no coordinate for host infinite sums, selected limits, quotient stream equality, countable choice, trichotomy, or ambient real completeness, none of those principles can escape through the zero-seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeriesTermTest}^{\uparrow}}$.

Constructive story: A $\mathsf{SeriesTermTest}^{\uparrow}$ packet is generated from a real-valued term stream, finite partial-sum windows, adjacent-difference rows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a zero RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeriesTermTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2127.1, Definition 2127.2, Definition 2127.3, and Definition 2127.4 seed the finite necessary-condition certificate route from real-series Cauchy tails to displayed terms tending to zero.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeriesTermTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, adjacent-difference exactness, shared tail-threshold routing, readback and zero-seal nonescape, and transport, replay, provenance, and local naming discipline for BEDC.Derived.SeriesTermTestUp.

Chapter 2128

A Concrete Naming Certificate for $\mathsf{NormalConvergence}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{NormalConvergence}^{\uparrow}$ packet records the finite normal-convergence route by which a real-series family, an absolute majorant, a displayed M-test tail ledger, and a uniform-limit handoff determine every endpoint-facing series read. It sits after the real-series carrier of chapter 1581, the absolute-majorant rows of chapter 1700, the finite majorant tail exactness of chapter 399, the uniform endpoint sealing of chapter 463, the regular rational readback of chapter 301, the stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies the finite dependency surface consumed by Tannery-style series-limit routes such as chapter 2094; it does not import a host uniform convergence theorem, a selected infinite sum, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2128.1 (Normal convergence carrier)**.** *A* $\mathsf{NormalConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (S, A, M, U, R, E, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *family row. It exposes the inspected real-valued term family and its finite partial-sum windows only through the real-series surface.*
2. *A is the* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *majorant row. It records the absolute majorant family and the finite domination comparison visible to the packet.*
3. *M is the* $\mathsf{WeierstrassMTest}^{\uparrow}$ *domination ledger. It displays the finite tail majorant and the exact tail budget used by normal-convergence requests.*
4. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row. It seals the uniform endpoint read only after the majorant and tail ledger have been displayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same finite tails, partial sums, and endpoint requests.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the regular readback and uniform handoff, not from a host infinite-sum coordinate.*
7. *H records componentwise* hsame *transport for the series family, absolute-majorant row, M-test ledger, uniform handoff, regular readback, Real seal, replay, provenance, and local name.*

8. *C records displayed* Cont *replay of the route*

$$S \longmapsto A \longmapsto M \longmapsto U \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* NormalConvergence$^{\uparrow}$, RealSeries$^{\uparrow}$, AbsoluteConvergence$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, UniformLimit$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *L is the local* NameCert$_{\text{NormalConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, M, U, R, E, H, C, P, L$. It has no coordinate for a host uniform convergence theorem, selected infinite sum, hidden pointwise limit, quotient stream equality, countable-choice selector, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NormalConvergenceUp.NormalConvergenceTasteGate single carrier alignment`

Theorem 2128.2 (Normal convergence NameCert obligations)**.** *For every accepted* NormalConvergence$^{\uparrow}$ *carrier*

$$N = (S, A, M, U, R, E, H, C, P, L),$$

the local NameCert$_{\text{NormalConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real-series family row, absolute-majorant row, M-test domination ledger, uniform-limit handoff, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *absolute-majorant exactness, requiring A to display the majorant comparison used by every inspected term family read from S;*
3. *uniform-tail exactness, requiring M to store the finite tail budget that the* WeierstrassMTest$^{\uparrow}$ *route exports to the uniform handoff;*
4. RegSeqRat$^{\uparrow}$ *handoff, requiring R to read the same displayed tail and partial-sum data before any endpoint seal is exported;*
5. Real$^{\uparrow}$ *seal non-escape, requiring E to remain downstream of S, A, M, U, R and to export no host uniform convergence theorem, selected infinite sum, hidden pointwise limit, quotient stream equality, countable choice, trichotomy, or ambient* Real$^{\uparrow}$ *completeness;*
6. *componentwise transport, displayed replay, provenance, and local naming for exactly the rows recorded in H, C, P, L.*

No normal-convergence NameCert obligation is stored outside these rows.

Proof. Project the accepted packet $N = (S, A, M, U, R, E, H, C, P, L)$. The carrier exposes exactly the real-series family, absolute majorant, M-test domination ledger, uniform handoff, regular readback, Real seal, and structural rows.

The majorant side is fixed before any endpoint read. The row S supplies the term-family and partial-sum surface, while A records the absolute-majorant comparison visible for the inspected family.

The finite tail budget is then forced through M. Since M is the displayed WeierstrassMTest$^{\uparrow}$ ledger, every uniform-tail request receives the same majorant exactness row before it can reach U.

The rows U, R, E form the endpoint stage. The uniform handoff exposes only the displayed tail route, the regular rational row reads that same route, and the Real seal is downstream of both.

Finally H, C, P, L transport, replay, source, and name the same packet. Because the carrier contains no excluded coordinate for a host uniform convergence theorem, selected infinite sums, hidden pointwise limits, quotient stream equality, choice, trichotomy, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2128.3 (Normal convergence uniform series handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{NormalConvergence}^{\uparrow}$ *packet factors through the displayed route*

$$S \longmapsto A \longmapsto M \longmapsto U \longmapsto R \longmapsto E \longmapsto H, C, P, L.$$

Thus the read reaches a $\mathsf{Real}^{\uparrow}$ *endpoint only after the real-series family row, absolute-majorant row, M-test tail ledger, uniform handoff, and regular rational readback have all been displayed. It does not read a host uniform convergence theorem, selected infinite sum, hidden pointwise limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an endpoint-facing series request. By Definition 2128.1, the request first projects the family row S and then the absolute-majorant row A.

The normal-convergence control is mediated by M. This row is the finite M-test domination ledger, so it provides the exact tail budget that the route may hand to U.

The handoff to U and R preserves the same displayed data. The uniform-limit row seals only the endpoint-facing uniform request, and the regular rational row reads the same tail and partial-sum route before a Real seal can appear.

The terminal stage is E, H, C, P, L. The Real seal is reached downstream of the displayed route, while transport, replay, provenance, and local naming preserve that same route. Therefore every endpoint-facing read factors through $S, A, M, U, R, E, H, C, P, L$ and cannot bypass the finite normal-convergence certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalConvergence}^{\uparrow}$ packet is generated from a RealSeriesUp family row, an AbsoluteConvergenceUp majorant row, a WeierstrassMTestUp domination ledger, a UniformLimitUp handoff row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NormalConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 2128.1, Definition 2128.2, and Definition 2128.3 seed the finite normal-convergence certificate surface between real-series families, absolute majorants, M-test tail ledgers, uniform handoffs, regular rational readback, and RealUp endpoint seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NormalConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host uniform convergence theorems, selected infinite sums, hidden pointwise limits, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, absolute-majorant exactness, uniform-tail exactness, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.NormalConvergenceUp.

Chapter 2129

A Concrete Naming Certificate for $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$ packet records the finite common-tail filter basis used when several regular Cauchy real names must answer one late-window request before any $\mathsf{Real}^{\uparrow}$ seal is read. It sits after the regular rational source rows of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy-filter route of chapter 333. The packet is a human-origin L10 basis for shared tail windows; it is not a quotient of Cauchy names, a selected limit, a countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2129.1 (Regular Cauchy common-tail filter carrier)**.** *A* $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, \Sigma, T, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite family of regular Cauchy source-name rows. Each source is read only through its displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *surface.*
2. *Σ is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule. It records the finite observation windows, requested tail cut, inspected indices, and synchronization route used for every source in X.*
3. *T is the common-tail basis row. It records the finite upper tail cut, the cofinal refinement requests, and the membership witnesses saying that all later windows are read through the same basis packet.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed sources at the shared windows of Σ and the tail basis T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested radius, tail slack, comparison budget, and exhaustion data consumed by the common-tail test.*
6. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *evidence row. It reads the tail basis T as a finite filter-basis route and exports only the displayed cofinal and stability data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the family, available only after X, Σ, T, R, D, Q have been displayed.*

8. *H records componentwise* hsame *transport for the source family, stream schedule, common-tail basis, readback row, dyadic ledger, Cauchy-filter row, real seal, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the route*

$$X, \Sigma, T, D \longmapsto R \longmapsto Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $X, \Sigma, T, R, D, Q, E, H, C, P, N$ and has no coordinate for quotient Cauchy equality, selected limits, countable choice, host filter completion, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 2129.2 (Regular Cauchy common-tail filter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$ *carrier*

$$F = (X, \Sigma, T, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the regular Cauchy source family, shared stream schedule, common-tail basis, regular rational readback, dyadic tolerance ledger, Cauchy-filter evidence row, terminal real seal, transport, replay, provenance, and local naming row;*

2. *common-tail basis discipline, requiring T to display the finite late cut, cofinal refinement requests, and membership witnesses consumed by every source in X through the same Σ schedule;*

3. *tolerance and readback exactness, requiring R and D to expose only the regular rational observations and dyadic budgets read from the displayed source family, shared windows, and common-tail basis;*

4. *filter and real-seal non-escape, requiring Q and E to remain downstream of X, Σ, T, R, D and to export no quotient Cauchy equality, selected limit, countable-choice schedule, host filter completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*

5. *transport, replay, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed route componentwise.*

No common-tail filter-basis obligation is stored outside these rows.

```
Lean4✓  BEDC.Derived.RegularCauchyCommonTailFilterUp.RegularCauchyCommonTailFilterTasteGate
single carrier alignment
```

Proof. Project the accepted packet $F = (X, \Sigma, T, R, D, Q, E, H, C, P, N)$. The carrier exposes exactly the regular Cauchy source family, shared schedule, common-tail basis, readback, tolerance, Cauchy-filter, real-seal, and structural rows.

The common-tail data are finite. The schedule Σ fixes the inspected windows, while T fixes the late cut and cofinal refinement requests through which every source in X must be read.

The readback and tolerance rows are tied to that same basis. The row R supplies only the regular rational observations visible at the displayed windows, and D supplies only the dyadic radius, slack, comparison budget, and exhaustion data attached to those windows.

The filter and real-seal rows remain downstream. The row Q reads T as a finite Cauchy-filter basis, and E is available only after the source, schedule, tail, readback, tolerance, and filter rows have all been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for quotient Cauchy equality, selected limits, countable choice, host filter completion, ambient real completeness, Lean verification, or a bridge, the five listed rows exhaust the local NameCert surface. □

Theorem 2129.3 (Regular Cauchy common-tail filter real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$ *carrier reads the terminal seal only through the finite route*

$$X, \Sigma, T, D \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed regular Cauchy source family, one shared StreamName schedule, a common-tail basis, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyFilter evidence, the terminal RealUp seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no quotient Cauchy equality, selected limit, countable-choice schedule, host filter completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with an accepted consumer read. The source family is X, and the only admissible finite observation route is the shared schedule Σ together with the common-tail basis T.

The regular rational readback is R. It is read from X through the same Σ and T rows, so the consumer cannot introduce a second tail schedule or a detached representative family.

The dyadic ledger D supplies the precision budget for the shared tail request. The Cauchy-filter evidence Q then reads that budget and tail basis as a finite filter-basis route.

Only after this route is displayed can E expose the $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N preserve the same displayed coordinates under transport, replay, provenance, and local naming.

Because no excluded coordinate occurs in the carrier, the real-seal consumer cannot read quotient Cauchy equality, selected limits, countable choice, host filter completion, or ambient real completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}$ packet is generated from a regular Cauchy source family, a shared StreamName schedule, a common-tail basis row, RegSeqRat readback, a DyadicRatCore tolerance ledger, CauchyFilter evidence, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2129.1, Definition 2129.2, and Definition 2129.3 seed the finite common-tail filter-basis route over shared StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, CauchyFilter evidence, RealUp seals, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonTailFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected limits, countable choice, host filter completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-tail basis discipline, tolerance and readback exactness, filter and Real seal non-escape, transport stability, replay confinement, provenance, and local naming for BEDC.Derived.RegularCauchyCommonTailFilterUp.

Chapter 2130

A Concrete Naming Certificate for $\mathsf{GaussTest}^{\uparrow}$

Following Gauss' positive-series test as presented by Knopp, the $\mathsf{GaussTest}^{\uparrow}$ packet records the finite first-order adjacent-quotient defect route by which a positive $\mathsf{RealSeries}^{\uparrow}$ source exposes a displayed comparison-test tail handoff. It refines the partial-sum windows and tail thresholds of chapter 1581, sits beside the quotient and comparison surfaces of chapter 1461 and chapter 1459, shares neighboring defect rows with chapter 1719 and chapter 2113, and exports endpoints only through chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Gauss-test evidence only through displayed source windows, adjacent quotient windows, unary first-order defect coefficients, dyadic margins, comparison routing, regular rational readback, and terminal Real sealing; it does not import a host infinite sum, selected limit, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2130.1 (Gauss test carrier)**.** *A* $\mathsf{GaussTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, Q, F, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RealSeries}^{\uparrow}$ *row. It exposes only positive-term windows, partial-sum windows, and tail-threshold requests already visible at the real-series boundary.*
2. *Q is the adjacent quotient-window row. It records the inspected neighboring-term ratios and the finite index window on which those quotients are read.*
3. *F is the unary first-order defect coefficient row. It records the displayed coefficient data for the Gauss expansion of the adjacent quotient defect on the inspected tail window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *margin row. It records the rational error margins, threshold witnesses, and finite inequality budgets used to compare the displayed defect coefficient with the required tail bound.*
5. *T is the comparison-test handoff row. It routes S, Q, F, D into the finite majorant or minorant surface consumed by* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$.
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison-test handoff and finite stream windows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*

8. *H stores componentwise* hsame *transport for the source, quotient-window, defect-coefficient, margin, comparison, readback, seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from each tail request through S, Q, F, D, T, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaussTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, F, D, T, R, E, H, C, P, N$. It has no coordinate for a host infinite sum, selected limit, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.GaussTestUp.GaussTestTasteGate single carrier alignment`

Theorem 2130.2 (Gauss test NameCert obligations)**.** *For every accepted* $\mathsf{GaussTest}^{\uparrow}$ *carrier*

$$G = (S, Q, F, D, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GaussTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source* $\mathsf{RealSeries}^{\uparrow}$ *windows, adjacent quotient windows, unary first-order defect coefficient row, dyadic error margins, comparison-test handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *quotient-window exactness, requiring every quotient read used by the Gauss defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *defect-row stability, requiring the coefficient row F and the error margins in D to remain fixed under the accepted tail request and under componentwise* hsame *transport;*
4. *comparison routing, requiring T to be the only row that turns the Gauss defect data into a convergence-test handoff;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring endpoint-facing data to pass through R and then E only after S, Q, F, D, T have been displayed;*
6. *transport, replay, provenance, and naming discipline, requiring H, C, P, N to transport, replay, source, and name exactly the displayed finite route.*

No Gauss-test obligation is stored outside these rows.

Proof. Project the accepted carrier $G = (S, Q, F, D, T, R, E, H, C, P, N)$. The carrier inventory already displays the real-series source, quotient windows, first-order defect coefficient row, dyadic margins, comparison handoff, readback, seal, transport, replay, provenance, and local name.

The quotient and defect clauses are finite. The source row S gives only inspected positive-term and partial-sum windows, while Q records neighboring-term quotients on the same tail window. The coefficient row F stores the unary first-order defect data, and D stores the rational margins and thresholds attached to that read.

The comparison handoff is forced through T. Before T appears, the packet has displayed only source, quotient, defect, and margin rows; after T appears, the endpoint route may proceed to the regular rational readback R and terminal seal E.

Finally H, C, P, N preserve and name this same route. Since the carrier has no excluded coordinate for host summation, selected limits, quotient series equality, choice, trichotomy, ambient

real completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2130.3 (Gauss test tail handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{GaussTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSeries}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes the source $\mathsf{RealSeries}^{\uparrow}$ *windows, the adjacent quotient surface shared with* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$*, the neighboring first-order defect discipline visible beside* $\mathsf{RaabeDuhamelTest}^{\uparrow}$ *and* $\mathsf{KummerTest}^{\uparrow}$*, the dyadic and stream display rows, and the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff before it reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 2130.1, the only source data available at the start are the $\mathsf{RealSeries}^{\uparrow}$ windows in S and the adjacent quotient windows in Q.

The Gauss-specific defect data are then read from F and D. These rows display the unary first-order coefficient data, the rational margins, and the tail threshold needed for the finite comparison request.

The neighboring tests are reference faces for the same quotient-defect lattice. The $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ surface supplies the adjacent quotient discipline, while $\mathsf{RaabeDuhamelTest}^{\uparrow}$ and $\mathsf{KummerTest}^{\uparrow}$ identify the nearby defect rows. The accepted $\mathsf{GaussTest}^{\uparrow}$ route still reaches endpoints only through the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ margins, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host infinite sum, selected limit, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaussTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GaussTest}^{\uparrow}$ packet is generated from source RealSeries windows, adjacent quotient windows, a unary first-order defect coefficient row, DyadicRatCore error margins, a RegularCauchyComparisonTest handoff, StreamName display rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaussTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2130.1, Definition 2130.2, and Definition 2130.3 seed the finite Gauss positive-series test boundary over displayed source series, quotient, first-order defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaussTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient series equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, defect-row stability, comparison routing, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal nonescape, transport, replay, provenance, and local naming discipline for BEDC.Derived.GaussTestUp.

Chapter 2131

A Concrete Naming Certificate for $\mathsf{SamuelCompletion}^{\uparrow}$

Following Samuel on uniform compactification and Bourbaki–Kelley on uniform-space completion, the $\mathsf{SamuelCompletion}^{\uparrow}$ packet records the completion-facing uniform route as finite BEDC data. It sits after the uniform completion surface of chapter 1473, the Cauchy-filter route of chapter 333, the dense uniform embedding boundary of chapter 1651, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the L10 uniform-completion handoff without asserting quotient completion, countable choice, a selected limit, host uniform equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2131.1 (Samuel completion carrier)**.** *A* $\mathsf{SamuelCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, F, D, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite uniform-entourage row. It records the entourages inspected by the completion request and the refinement edges used to compare them.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row. It records the Cauchy-filter data read through the displayed entourage family.*
3. *D is the* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *row. It records the dense source embedding, image-window requests, and pullback evidence used before any completion-facing read is exposed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which dense-source observations and completion-facing requests are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed windows and entourage budgets.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after the displayed uniform, filter, dense-embedding, stream-window, and regular-rational rows have been read.*
7. *H records componentwise* hsame *transport for entourages, Cauchy-filter rows, dense-embedding rows, stream windows, regular readbacks, real seals, replay data, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$U, D, F, W \longmapsto R \longmapsto A.$$

9. *P is the* Pkg *provenance over* $\mathsf{SamuelCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SamuelCompletion}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $U, F, D, W, R, A, H, C, P, N$ and has no coordinate for quotient completion, countable choice, selected limits, host uniform equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SamuelCompletionUp.SamuelCompletionTasteGate single carrier alignment`

Theorem 2131.2 (Samuel completion NameCert obligations)**.** *For the carrier of Definition 2131.1, the local* $\mathsf{NameCert}_{\mathsf{SamuelCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the packet $S = (U, F, D, W, R, A, H, C, P, N)$;*
2. *classifier stability under componentwise* hsame *transport of the uniform-entourage row, Cauchy-filter row, dense-embedding row, stream-window schedule, regular rational readback, real seal, replay row, provenance row, and local naming row;*
3. *displayed* Cont *replay of the route from uniform entourages and dense-source windows through Cauchy-filter data to regular rational readback and the terminal real seal;*
4. *non-escape for every consumer read, requiring the read to remain inside* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

No obligation is stored in a quotient completion, selected limit, countable-choice schedule, host uniform equality row, or ambient completeness theorem.

Proof. Project the accepted carrier. Carrier admission is exactly the display of the ten rows $U, F, D, W, R, A, H, C, P, N$, so there is no additional host object to construct.

Classifier stability is componentwise. The hsame row compares the displayed uniform-entourage, Cauchy-filter, dense-embedding, stream-window, regular-readback, and real-seal coordinates, while the structural rows transport replay, provenance, and local naming over the same coordinates.

The continuation obligation is the displayed route in the carrier. A consumer first reads uniform entourages and dense-source windows, then the Cauchy-filter row, then the regular rational readback, and only then the real seal.

The non-escape clause follows by inspecting the carrier inventory. Since the packet has no coordinate for quotient completion, countable choice, selected limits, host uniform equality, or ambient completeness, no NameCert obligation can export one. □

Theorem 2131.3 (Samuel completion uniform handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{SamuelCompletion}^{\uparrow}$ *carrier reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only after the displayed route through* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, *and* $\mathsf{DenseUniformEmbedding}^{\uparrow}$ *has been read. The handoff route is therefore*

$$U, D, F, W \longmapsto R \longmapsto A \longmapsto H, C, P, N,$$

not a quotient-completion carrier, not a selected limit, not a countable-choice schedule, not host uniform equality, and not an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier and project its uniform-entourage, Cauchy-filter, dense-embedding, and stream-window rows. These rows are upstream of the regular rational readback row in the displayed route.

The regular readback row R is therefore not available as an isolated real-name source. It is read only after the entourage refinements, dense embedding evidence, Cauchy-filter data, and finite windows have been displayed.

The real seal A is downstream of R. Thus every real-facing consumer must traverse the uniform-completion, Cauchy-filter, and dense-embedding route before it can consume a $\mathsf{RegSeqRat}^{\uparrow}$ readback or a terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally the H, C, P, N rows preserve the same route under transport, replay, provenance, and local naming. The excluded host-side principles have no carrier coordinate, so the handoff cannot expose them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SamuelCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SamuelCompletion}^{\uparrow}$ packet is generated from finite uniform-entourage rows, CauchyFilterUp data, DenseUniformEmbeddingUp rows, StreamName windows, RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SamuelCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2131.1, Definition 2131.2, and Definition 2131.3 seed the L10 uniform-completion route from finite entourage and dense-embedding data through Cauchy-filter, StreamName, RegSeqRat, Real seal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SamuelCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, quotient stream equality, selected limits, host uniform equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, displayed continuation replay, uniform handoff, non-escape, provenance, and local naming for BEDC.Derived.SamuelCompletionUp.

Chapter 2132

A Concrete Naming Certificate for $\mathsf{RiemannRearrangement}^{\uparrow}$

Following Riemann's rearrangement theorem and Bishop–Bridges constructive analysis, the $\mathsf{RiemannRearrangement}^{\uparrow}$ packet records the finite certificate surface by which a real series source, a displayed finite schedule, and signed tail ledgers determine every rearrangement-facing endpoint read. It sits after the $\mathsf{RealSeries}^{\uparrow}$ carrier of chapter 1581, the separation supplied by $\mathsf{AbsoluteConvergence}^{\uparrow}$ in chapter 1700, the regular Cauchy comparison discipline of chapter 1459, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the endpoint seal of chapter 38. The packet names only finite rearrangement certificates over displayed rows; it does not import host infinite rearranged sums, selected limits, quotient equality, countable choice, trichotomy, absolute convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2132.1 (Riemann rearrangement carrier)**.** *A* $\mathsf{RiemannRearrangement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, P, M, N, W, Q, D, E, H, C, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row. It exposes the term stream and finite partial-sum surface only through the real-series certificate already named for S.*

2. *R is the finite rearrangement schedule. It records the displayed source indices, their scheduled order, and the finite coverage check that every endpoint-facing read uses exactly the listed schedule entries.*

3. *P is the positive signed tail ledger. It records the displayed positive terms, their scheduled appearances, and the finite partial-sum contributions that remain visible before the endpoint seal.*

4. *M is the negative signed tail ledger. It records the displayed negative terms, their scheduled appearances, and the finite partial-sum contributions that remain visible before the endpoint seal.*

5. *N is the neutral or zero-tail ledger. It records displayed zero terms and neutral entries whose contribution is routed through the schedule without changing the signed tail accounting.*

6. *W is the finite partial-sum window row. It records the source partial sums, scheduled partial sums, and endpoint-facing comparison windows requested by the packet.*

7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed source terms, scheduled terms, signed tail ledgers, partial-sum windows, and endpoint comparisons.*

8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting row for finite addition, subtraction, sign separation, schedule coverage, and endpoint comparison tolerances.*

9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after S, R, P, M, N, W, Q, D have been displayed.*

10. *H records componentwise* hsame *transport for the source row, schedule, signed ledgers, neutral ledger, partial-sum windows, readback, dyadic ledger, endpoint seal, continuation, and local naming row.*

11. *C records displayed* Cont *replay of the route*

$$S, R, P, M, N, W, Q, D \longmapsto E.$$

12. *L is the local* $\mathsf{NameCert}_{\mathsf{RiemannRearrangement}^{\uparrow}}$ *row, sourced over* $\mathsf{RiemannRearrangement}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

The classifier compares accepted packets only through $S, R, P, M, N, W, Q, D, E, H, C, L$. It has no coordinate for a host infinite rearranged sum, a selected limit, quotient equality of series, countable choice, trichotomy, absolute convergence as a conclusion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2132.2 (Riemann rearrangement NameCert obligations)**.** *For every accepted* $\mathsf{RiemannRearrangement}^{\uparrow}$ *carrier*

$$K = (S, R, P, M, N, W, Q, D, E, H, C, L),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannRearrangement}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real series source, finite rearrangement schedule, signed tail ledgers, neutral ledger, partial-sum windows, readback, dyadic ledger, endpoint seal, transport, continuation, and local naming rows;*

2. *schedule exactness, requiring every endpoint-facing read to expose the finite schedule R before it reads a scheduled term or scheduled partial sum;*

3. *positive and negative tail separation, requiring the positive ledger P, negative ledger M, and neutral ledger N to remain distinct under H and to be compared only through the displayed readback row Q and dyadic ledger D;*

4. *partial-sum window stability, requiring the source and scheduled partial-sum windows stored in W to be replayed by C through the same finite route;*

5. *endpoint nonescape, requiring E to remain downstream of S, R, P, M, N, W, Q, D and to export no host infinite rearranged sum, selected limit, quotient equality, countable choice, trichotomy, absolute convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No rearrangement obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RiemannRearrangementUp.RiemannRearrangementTasteGate single carrier alignment`

Proof. Project the accepted packet $K = (S, R, P, M, N, W, Q, D, E, H, C, L)$. The carrier exposes exactly the real series source, finite schedule, signed and neutral ledgers, partial-sum windows, readback, tolerance ledger, endpoint seal, and structural rows.

Schedule exactness is forced by R. Every scheduled term or scheduled partial sum is read through the finite list of displayed source indices, so the packet has no hidden permutation or countable-choice selector.

The signed accounting is separated by construction. The row P contains the positive displayed tail data, the row M contains the negative displayed tail data, and the row N contains neutral entries. The transport row H preserves those coordinates componentwise, while Q and D are the only comparison rows.

The partial-sum window row W and continuation row C replay the same finite route from source terms to scheduled windows. Only after this route is displayed may E expose the $\mathsf{Real}^{\uparrow}$ endpoint seal.

Since the carrier has no excluded coordinate for host infinite rearranged sums, selected limits, quotient equality, countable choice, trichotomy, absolute convergence, or ambient real completeness, the five listed obligations exhaust the local NameCert surface. □

Theorem 2132.3 (Riemann rearrangement positive-negative ledger)**.** *Every endpoint-facing read from an accepted* $\mathsf{RiemannRearrangement}^{\uparrow}$ *carrier must display both signed tail ledgers and the finite rearrangement schedule before the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, each such read factors through*

$$S, R, P, M, W, Q, D \longmapsto E$$

together with H, C, L*, and it may use the neutral row* N *only as displayed zero-tail bookkeeping along the same scheduled route.*

Proof. Begin with an endpoint-facing request from an accepted carrier. By Definition 2132.1, the request first projects the source row S and the finite schedule row R.

The signed ledger stage is then visible. The positive row P and negative row M must both be present before the packet can compare a scheduled partial sum with an endpoint, because the carrier stores the two signed tail surfaces as separate coordinates. The neutral row N can only add displayed zero-tail bookkeeping to that same route.

The finite window and readback stage follows. The row W displays the source and scheduled partial-sum windows, Q reads them through $\mathsf{RegSeqRat}^{\uparrow}$, and D records the dyadic tolerance accounting for the same displayed data.

Only after S, R, P, M, W, Q, D have been displayed can E provide the endpoint seal. The rows H, C, L transport, replay, and name the same route, so no endpoint-facing read can bypass the signed tail ledgers or the finite rearrangement schedule. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRearrangement}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannRearrangement}^{\uparrow}$ packet is generated from a RealSeriesUp source, a finite rearrangement schedule, positive and negative signed tail ledgers, a neutral ledger, finite partial-sum windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannRearrangement}^{\uparrow}}$, seed)

Scope closed: Definition 2132.1, Definition 2132.2, and Definition 2132.3 seed the finite rearrangement certificate surface over RealSeriesUp with separated signed tail ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannRearrangement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host infinite rearranged sums, selected limits, quotient equality, countable choice, trichotomy, absolute convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exactness, positive and negative tail separation, partial-sum window stability, endpoint nonescape, and TasteGate field faithfulness for BEDC.Derived.RiemannRearrangementUp.

Chapter 2133

A Concrete Naming Certificate for $\mathsf{OpenIntervalReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OpenIntervalReal}^{\uparrow}$ packet names an open real interval by finite endpoint seals, located order tests, dyadic tolerance ledgers, and regular Cauchy readbacks before any topology or completion-facing consumer sees membership. It sits after the located interval surface of chapter 1709, the located-order test of chapter 2112, the dyadic tolerance core of chapter 291, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a constructive interface for the open interval (a, b) over displayed $\mathsf{Real}^{\uparrow}$ names; it is not host trichotomy, quotient real equality, a chosen representative, countable choice, ambient real completeness, Lean verification, or bridge checking.

Definition 2133.1 (OpenIntervalReal carrier)**.** *An* $\mathsf{OpenIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, X, L, U, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are endpoint* $\mathsf{Real}^{\uparrow}$ *seal rows. They name the lower and upper endpoints only through displayed regular Cauchy data.*
2. *X is the candidate point* $\mathsf{Real}^{\uparrow}$ *seal row whose open-interval membership is being read.*
3. *L is the left located-order row from* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$*, witnessing that A is located strictly below X at the displayed tolerance.*
4. *U is the right located-order row from* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$*, witnessing that X is located strictly below B at the displayed tolerance.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic radius, positive slack, endpoint comparison budget, and exhaustion ledger shared by L and U.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite endpoint and point windows inspected by the two located-order tests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It reads the same finite windows as regular rational observations before the endpoint and point seals are consumed.*
8. *H records componentwise* hsame *transport for the endpoint seals, candidate seal, order-test rows, dyadic tolerance row, stream windows, readback row, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay from an open-interval membership request through* $A, B, X, L, U, D, S, R, H, P, N$
10. *P is the* Pkg *provenance over* $\mathsf{OpenIntervalReal}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OpenIntervalReal}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these twelve rows form one finite open-interval membership packet. Its classifier reads only the displayed tuple $A, B, X, L, U, D, S, R, H, C, P, N$ and has no coordinate for host trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 2133.2 (OpenIntervalReal NameCert obligations)**.** *For every accepted* $\mathsf{OpenIntervalReal}^{\uparrow}$ *carrier $I = (A, B, X, L, U, D, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OpenIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two endpoint* $\mathsf{Real}^{\uparrow}$ *seals, candidate* $\mathsf{Real}^{\uparrow}$ *seal, two located-order rows, dyadic tolerance ledger, stream windows, regular rational readback, transport, replay, provenance, and local naming row;*
2. *endpoint-seal exactness, requiring A and B to be read only after the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks expose the endpoint data used by the order tests;*
3. *candidate-seal exactness, requiring X to be read through the same window and readback route before a membership verdict is exported;*
4. *located left boundary, requiring L to display the finite order-test route for $A < X$ over the shared dyadic tolerance row D;*
5. *located right boundary, requiring U to display the finite order-test route for $X < B$ over the same dyadic tolerance row D;*
6. *dyadic tolerance coherence, requiring the positive slack and exhaustion rows in D to be the only tolerance data consumed by both boundary tests;*
7. *structural stability, requiring H and C to transport and replay the endpoint, candidate, order-test, tolerance, window, readback, provenance, and local naming rows componentwise;*
8. *ledger refusal, excluding host trichotomy, quotient real equality, selected representatives, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No local NameCert obligation for the open interval is stored outside these rows.

Lean4 ✓ `BEDC.Derived.OpenIntervalRealUp.OpenIntervalRealTasteGate single carrier alignment`

Proof. Project the accepted packet $I = (A, B, X, L, U, D, S, R, H, C, P, N)$. The projection exposes exactly the two endpoint seals, candidate seal, two located-order rows, dyadic tolerance ledger, stream windows, regular rational readback, and structural rows.

The readback route is finite. The stream row S supplies the endpoint and point windows, while the row R reads those same windows as regular rational observations before the $\mathsf{Real}^{\uparrow}$ seals are consumed.

The two boundary tests are then read against the same dyadic ledger. The row L records the displayed route for $A < X$, and the row U records the displayed route for $X < B$; both use the positive slack and exhaustion data stored in D.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since none of the carrier coordinates stores host trichotomy, quotient equality, representative choice, countable

choice, ambient completeness, a Lean theorem, or a bridge, no further local obligation can be read from the packet. □

Theorem 2133.3 (OpenIntervalReal located membership route)**.** *Every accepted* $\mathsf{OpenIntervalReal}^{\uparrow}$ *membership read for* X *between endpoints* A *and* B *factors through the finite route*

$$A, B, X \longmapsto S, R, D \longmapsto L, U \longmapsto H, C, P, N.$$

Thus an open-interval consumer may inspect endpoint seals, candidate seal, finite windows, regular rational readbacks, dyadic tolerance data, and the two located-order rows, but it cannot obtain host trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from the packet.*

Proof. Start with an accepted membership read. The only real-name coordinates are A, B, X, so any endpoint or candidate observation must pass through the displayed $\mathsf{Real}^{\uparrow}$ seals.

The finite observation stage comes next. The row S fixes the inspected windows, the row R reads those windows as regular rational data, and the row D supplies the dyadic tolerance, positive slack, and exhaustion ledger shared by the two boundary tests.

The membership verdict is the conjunction of the two located-order rows. The left row L displays $A < X$, and the right row U displays $X < B$, each over the same readback and tolerance data rather than over a host-order primitive.

The rows H, C, P, N close the route by transport, replay, provenance, and local naming. Because the route has no coordinate for trichotomy, quotient equality, representative selection, countable choice, completeness, verification, or a bridge, those principles are not exported by the membership read. □

Closure status for $\mathsf{NameCert}_{\mathsf{OpenIntervalReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpenIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2133.1, Definition 2133.2, and Definition 2133.3 seed the finite open-interval membership boundary over endpoint RealUp seals, located order-test rows, dyadic tolerance data, StreamName windows, RegSeqRat readback, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpenIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host trichotomy, quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.OpenIntervalRealUp covering the twelve-row carrier, local NameCert obligations, and located membership route.

Constructive story: An $\mathsf{OpenIntervalReal}^{\uparrow}$ packet is generated from lower and upper RealUp endpoint seals, a candidate RealUp seal, left and right CauchyRealLocatedOrderTest rows, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2134

A Concrete Naming Certificate for $\mathsf{MonoidalCompletion}^{\uparrow}$

Following Bourbaki and Kelley on completion and product uniformities, and Bishop–Bridges on constructive metric completion, the $\mathsf{MonoidalCompletion}^{\uparrow}$ packet records the finite BEDC surface on which completion is consumed as a product-compatible reflector. It sits over the metric-completion row of chapter 1506, the binary product handoff of chapter 1540, the functorial completion route of chapter 423, and the reflection boundary of chapter 403. The chapter names only the finite reflector and coherence rows visible to downstream Real and Completion consumers; it does not construct quotient completions, choose representatives, identify host products, assume countable choice, use open-cover compactness, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2134.1 (Monoidal completion carrier)**.** *A* $\mathsf{MonoidalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, U, V, P, A, S, H, C, G, N)$$

with the following visible rows:

1. *M is the metric-completion reflector row. It exposes the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *source-to-completion route and the reflector-facing boundary inherited from chapter 1506.*
2. *U is the unit row. It records the source-object admission into the completion surface before any product consumer may inspect completed data.*
3. *V is the counit row. It records the displayed return route for already completed data and keeps that route inside the completion-reflection boundary.*
4. *P is the binary product row. It is read through the coordinatewise completion product surface of chapter 1540, not through host product equality.*
5. *A is the associativity row. It records that either parenthesization of a finite binary product read uses the same coordinatewise completion, unit, counit, and reflector rows.*
6. *S is the symmetry row. It records that swapping two displayed coordinates swaps only the finite product rows and their associated transport and replay data.*
7. *H records componentwise* hsame *transport for metric-completion, unit, counit, product, associativity, symmetry, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the product-compatible reflector route from metric completion through unit, counit, binary product, associativity, symmetry, and structural rows.*

9. *G is the* Pkg *provenance over* $\mathsf{MonoidalCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionProduct}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MonoidalCompletion}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. It has no coordinate for quotient completion equality, selected representatives, host product equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a host-level symmetric monoidal category.*

Lean4 (definition) `BEDC.Derived.MonoidalCompletionUp.taste gate`

Theorem 2134.2 (Monoidal completion NameCert obligations)**.** *For the carrier of Definition 2134.1, the local* $\mathsf{NameCert}_{\mathsf{MonoidalCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for $Q = (M, U, V, P, A, S, H, C, G, N)$;*
2. *reflector compatibility, requiring the metric-completion reflector row M, unit row U, and counit row V to be consumed through the same completion boundary named in chapter 1506 and chapter 403;*
3. *product compatibility, requiring the binary product row P to factor through the coordinatewise product handoff of chapter 1540;*
4. *finite coherence, requiring associativity row A and symmetry row S to compare only displayed product routes and not host product objects;*
5. *transport and replay stability under componentwise* hsame *transport H and displayed* Cont *replay C;*
6. *provenance exactness, requiring every consumer read to be sourced by G and locally certified by N.*

No obligation is stored in a quotient completion, selected representative, host product equality, countable-choice schedule, open-cover compactness theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or external monoidal-category structure.*

Lean4 ✓ `BEDC.Derived.MonoidalCompletionUp.MonoidalCompletionCarrier namecert obligations`

Proof. Carrier admission is projection from the finite packet in Definition 2134.1. The only displayed coordinates are the reflector row, unit row, counit row, binary product row, associativity row, symmetry row, transport row, replay row, provenance row, and local naming row.

Reflector compatibility is read through the metric-completion and completion-reflection siblings. The M, U, V rows bind source admission, completed-data return, and reflection boundary before any product-compatible consumer can inspect the route.

Product compatibility is then the coordinatewise handoff supplied by chapter 1540. The product row P is a finite route through the displayed coordinate completions and does not add a host product object.

Associativity and symmetry are coherence rows over that finite product route. They compare parenthesized and swapped displays only by reusing the same M, U, V, P, H, C, G, N data, so no quotient equality, selected representative, host product equality, choice principle, compactness theorem, or ambient completeness theorem is available. □

Theorem 2134.3 (Monoidal completion product reflector handoff)**.** *Let $Q = (M, U, V, P, A, S, H, C, G, N)$ be an accepted* $\mathsf{MonoidalCompletion}^{\uparrow}$ *carrier. Every completion-facing product consumer factors*

through the finite route

$$M \longmapsto U, V \longmapsto P \longmapsto A, S \longmapsto H, C, G, N.$$

The handoff uses the metric-completion reflector surface of chapter 1506, the coordinatewise product surface of chapter 1540, the functorial route of chapter 423, and the reflection boundary of chapter 403; it is not a host product completion and not a quotient completion product.

Proof. Project the accepted carrier. The metric-completion reflector row M supplies the completion-facing source boundary, and the unit and counit rows U, V record the two reflector directions available to consumers.

The binary product row P can be consumed only after those reflector rows are displayed. Its coordinatewise content is the same finite product handoff named by the Cauchy-completion product chapter, while the functorial sibling supplies the route discipline for completion-facing maps.

The associativity and symmetry rows A, S are downstream of P. They identify which finite product route is being replayed after a parenthesization or coordinate swap, but they do not identify host product objects.

Finally H, C, G, N preserve the route under transport, replay, provenance, and local certification. Since no further row occurs in Q, the consumer obtains exactly the displayed product-compatible reflector handoff and no quotient completion, selected representative, host product equality, choice principle, compactness theorem, or ambient completeness theorem. □

Theorem 2134.4 (Monoidal completion non-escape)**.** *An accepted* $\mathsf{MonoidalCompletion}^{\uparrow}$ *carrier exports no read outside*

$$(M, U, V, P, A, S, H, C, G, N).$$

In particular, any product-compatible completion consumer must read metric completion, unit, counit, binary product, associativity, symmetry, componentwise transport, displayed continuation replay, provenance, and local naming evidence from the same finite packet.

Proof. Inspect the carrier inventory in Definition 2134.1. Every exported read must be one of the ten displayed rows or a replay of those rows through the local NameCert surface.

If a consumer omits M, it has no metric-completion reflector row. If it omits U or V, it has lost one reflector direction. If it omits P, it has no binary product handoff. If it omits A or S, it has no finite associativity or symmetry comparison for product-compatible reads.

The structural rows are also necessary. Without H, the classifier is not stable under componentwise transport; without C, the route is not replayed as displayed continuation data; without G and N, provenance and local naming are absent.

There is no remaining coordinate. Therefore the packet cannot export quotient completions, selected representatives, host product equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or an external monoidal-category object. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonoidalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MonoidalCompletion}^{\uparrow}$ packet is generated from a metric-completion reflector row, unit row, counit row, binary product row, associativity row, symmetry row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonoidalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2134.1, Definition 2134.2, Definition 2134.3, and Definition 2134.4 seed the finite product-compatible reflector surface for completion through metric-completion, product, functorial, reflection, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MonoidalCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host product equality, countable choice, open-cover compactness, ambient Real$^{\uparrow}$ completeness, or an external monoidal-category structure.

Upgrade path: Obligation closure requires checked carrier admission, reflector compatibility, product compatibility, finite associativity and symmetry coherence, transport stability, replay stability, provenance exactness, and non-escape for BEDC.Derived.MonoidalCompletionUp.

Chapter 2135

A Concrete Naming Certificate for $\mathsf{SmythPowerdomain}^{\uparrow}$

Following Smyth, Plotkin, and Gierz–Hofmann–Keimel–Lawson–Mislove–Scott domain theory, the $\mathsf{SmythPowerdomain}^{\uparrow}$ packet records the finite upper-family powerdomain route over L10 interval-domain cells. It sits after the interval-domain carrier of chapter 1543, the way-below refinement of chapter 1545, the Scott-real presentation of chapter 2083, and the asymmetric completion route of chapter 2095. The packet exports compact upper approximation data for completion-facing consumers without quotient completion, selected representatives, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2135.1 (Smyth powerdomain carrier)**.** *A* $\mathsf{SmythPowerdomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (F, I, W, S, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *F is a finite nonempty upper family of L10 interval-domain cells. Each entry is an accepted* $\mathsf{IntervalDomain}^{\uparrow}$ *cell, and the family is read by upward closure under the displayed interval containment order.*
2. *I is the aggregate* $\mathsf{IntervalDomain}^{\uparrow}$ *support row, recording the interval cells, endpoint containment checks, enclosure order, and finite upper-family membership witnesses.*
3. *W is the* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *support row, recording compact-containment refinements used when a consumer asks for a tighter upper approximation.*
4. *S is the* $\mathsf{ScottReal}^{\uparrow}$ *readback row, exposing Scott-real consumers only after the interval-domain and way-below rows have been displayed.*
5. *C is the* $\mathsf{SmythCompletion}^{\uparrow}$ *handoff row, exposing asymmetric completion consumers only through finite upper-family approximation data.*
6. *H records componentwise* hsame *transport for the finite family, interval-domain support, way-below support, Scott-real readback, Smyth-completion handoff, replay, provenance, classifier, and local naming rows.*
7. *R records displayed* Cont *replay from upper-family requests through $F, I, W, S, C, H, G, Q, N$.*
8. *G is the* Pkg *provenance over* $\mathsf{SmythPowerdomain}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{ScottReal}^{\uparrow}$, $\mathsf{SmythCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *Q is the classifier row comparing accepted packets only through finite upper-family membership, interval containment, way-below refinement, Scott-real readback, Smyth-completion handoff, transport, replay, and provenance.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SmythPowerdomain}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. It has no coordinate for a quotient completion object, a selected representative, countable choice, host real equality, open-cover compactness, a hidden ambient compactness principle, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.SmythPowerdomainUp.SmythPowerdomainTasteGate single carrier alignment`

Theorem 2135.2 (Smyth powerdomain NameCert obligations)**.** *For an accepted* $\mathsf{SmythPowerdomain}^{\uparrow}$ *carrier*

$$P = (F, I, W, S, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SmythPowerdomain}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the finite nonempty upper family, interval-domain support row, way-below support row, Scott-real readback row, Smyth-completion handoff row, transport, replay, provenance, classifier, and local naming row;*

2. *nonempty upper-family exposure, requiring F to display at least one interval-domain cell and to answer every membership request by finite containment data in I;*

3. *way-below refinement compatibility, requiring every requested tightening of an upper approximation to pass through W before it reaches a Scott-real or completion-facing consumer;*

4. *Scott-real handoff discipline, requiring S to read only the interval-domain and way-below rows already exposed by F, I, W;*

5. *Smyth-completion handoff discipline, requiring C to receive finite upper-family approximation data rather than quotient completion data;*

6. *structural exhaustion, requiring H, R, G, Q, N to transport, replay, source, classify, and name only the listed rows.*

No local NameCert obligation is stored outside these rows, and none exports quotient completion, selected representatives, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2135.1. The accepted packet exposes exactly the finite upper family, interval-domain support, way-below support, Scott-real readback, Smyth-completion handoff, transport, replay, provenance, classifier, and local naming rows.

Nonempty upper-family exposure is read from F and I. The row F supplies a finite inhabited list of interval cells, and I supplies the displayed containment and membership witnesses used to read that list upward.

Way-below refinement compatibility is read before any downstream handoff. A tighter upper approximation may be consumed only after W has displayed the compact-containment evidence connecting the requested cell to the interval-domain support.

The Scott-real and Smyth-completion handoffs are downstream rows. The row S reads the same interval and way-below data as a Scott-real approximation surface, while C reads the finite upper-family route as an asymmetric completion approximation.

Finally H, R, G, Q, N add no analytic principle. They transport, replay, source, classify, and name the displayed route, so the excluded quotient, representative selection, choice, host equality, compactness, and completeness principles have no row from which a consumer could read them. □

Theorem 2135.3 (Smyth powerdomain interval-domain handoff)**.** *Let*

$$P = (F, I, W, S, C, H, R, G, Q, N)$$

be an accepted $\mathsf{SmythPowerdomain}^{\uparrow}$ *carrier. Every Scott-real or completion-facing consumer of* P *factors through the displayed finite route*

$$F \longmapsto I \longmapsto W \longmapsto S, C \longmapsto H, R, G, Q, N.$$

Thus the powerdomain packet reaches $\mathsf{ScottReal}^{\uparrow}$ *and* $\mathsf{SmythCompletion}^{\uparrow}$ *consumers only by passing from a finite nonempty upper family of interval-domain cells to interval containment data, way-below refinements, the downstream readback rows, and the structural* hsame/Cont/Pkg/NameCert *rows.*

Proof. Fix a Scott-real or completion-facing consumer of the accepted packet. The first visible object is the finite nonempty family F, so the consumer begins with displayed interval-domain cells rather than with a completed host object.

The interval-domain stage reads I. This row supplies the endpoint containment, enclosure order, and finite membership evidence that make the family an upper approximation surface.

The refinement stage reads W. Compact-containment evidence is displayed before any consumer may request a sharper approximation, so the route stays inside the interval-domain and way-below packets.

The downstream handoff reads S or C. A Scott-real consumer obtains only the displayed interval-domain readback, and a Smyth-completion consumer obtains only the finite upper-family approximation route.

The rows H, R, G, Q, N close the factorization by transport, replay, provenance, classification, and local naming. Since no other coordinate exists in the carrier, the excluded quotient, choice, host equality, compactness, and completeness principles cannot be exported by the handoff. □

Theorem 2135.4 (Smyth powerdomain non-escape)**.** *Let* P *be an accepted* $\mathsf{SmythPowerdomain}^{\uparrow}$ *carrier. Any consumer that asks for quotient completion, selected representatives, countable choice, host real equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness receives no row of* P *from which such data can be read. The only exported data are the finite upper-family carrier, the interval-domain and way-below support rows, the Scott-real and Smyth-completion handoff rows, and the structural transport, replay, provenance, classifier, and local naming rows.*

Proof. Inspect the carrier in Definition 2135.1. Its analytic rows are F, I, W, S, C, and each of them is a finite interval-domain, way-below, Scott-real, or Smyth-completion route.

The structural rows H, R, G, Q, N only preserve, replay, source, compare, and name those displayed routes. They do not introduce a new carrier coordinate or a hidden functional.

Therefore a consumer requesting quotient completion, a selected representative, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness has no matching row to read. The request remains outside the packet interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SmythPowerdomain}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SmythPowerdomain}^{\uparrow}$ packet is generated from a finite nonempty upper family of Interval-Domain cells, aggregate interval-domain support, way-below refinement support, ScottReal readback, SmythCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, classifier comparison, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmythPowerdomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2135.1, Definition 2135.2, Definition 2135.3, and Definition 2135.4 seed the finite upper-family powerdomain route over L10 interval-domain cells.

Formal verification: FormalStatus(NameCert$_{\mathsf{SmythPowerdomain}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite upper-family exposure, way-below refinement compatibility, Scott-real handoff discipline, Smyth-completion handoff discipline, transport stability, replay stability, classifier exhaustion, and ledger non-escape for BEDC.Derived.SmythPowerdomainUp.

Chapter 2136

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$

Following Bishop and Bridges, the regular Cauchy completion is read here as a finite-window completion interface rather than as a quotient object. The $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ packet consumes the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It packages unit and bind rows for the L10 completion route while excluding quotient stream equality, selected limits, host real equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2136.1 (Regular Cauchy completion monad carrier)**.** *A* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, R, D, U, B, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family. It records the source windows, unit windows, bind-request windows, and seal-facing observation cuts inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to W. It contains the regular rational rows read from source windows and the intermediate readbacks produced by unit or bind requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested scale, composition slack, triangle-budget rows, and exhausted dyadic windows used by unit, bind, and seal handoffs.*
4. *U is the unit row. It embeds a displayed regular rational input into the completion interface by copying the relevant window and tolerance rows into a one-stage completion name.*
5. *B is the bind row. It composes an accepted completion name with a displayed finite-window Kleisli request by replaying the source windows, readbacks, and dyadic budget before exposing the composed row.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row. It is reachable only after the displayed W, R, D, U, B route has been read.*
7. *H records componentwise* hsame *transport for windows, readbacks, dyadic ledgers, unit rows, bind rows, seal rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$W, D \longmapsto R \longmapsto U, B \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionMonad}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $W, R, D, U, B, E, H, C, P, N$ and has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionMonadUp.RegularCauchyCompletionMonadTasteGate`
`single carrier alignment`

Theorem 2136.2 (Regular Cauchy completion unit laws)**.** *For every accepted* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *carrier*

$$M = (W, R, D, U, B, E, H, C, P, N),$$

the unit row U is a finite-window identity for the displayed completion interface. A source regular rational read through W, R, D is admitted into the completion packet without changing its inspected windows, readback row, dyadic scale, componentwise transport row, replay row, provenance row, or local naming row.

Proof. Project the accepted carrier M. The projection exposes the finite-window row W, the regular rational readback family R, the dyadic ledger D, and the unit row U before any seal-facing row is visible.

The unit row is checked by finite-window induction on W. In the empty-window case, there is no observation to alter, and the displayed readback and dyadic ledger are the unique empty rows carried by the packet.

For a successor window, the induction hypothesis identifies the previous inspected segment. The next observation is copied by U with the same readback row and the same dyadic scale entry, so the extended window remains the displayed source route rather than a selected limit.

The structural rows H, C, P, N preserve this copied route componentwise. Since the carrier has no quotient stream, host equality, countable-choice selector, or ambient completeness coordinate, the unit law is exactly the finite identity of the displayed rows. □

Theorem 2136.3 (Regular Cauchy completion bind laws)**.** *For every accepted* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *carrier*

$$M = (W, R, D, U, B, E, H, C, P, N),$$

the bind row B composes completion-facing finite-window requests by replaying the source windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances before exporting the composed row. The left-unit, right-unit, and associativity comparisons are therefore comparisons of displayed $W, R, D, U, B, H, C, P, N$ rows, not comparisons of quotient streams or selected limits.*

Proof. Open the bind row B. A bind request is admitted only after the relevant source window in W, the matching readback in R, and the requested dyadic budget in D have been displayed.

For left unit, first read the unit row U. The preceding theorem identifies the unit route with the displayed source window, so applying B to that route consumes the same W, R, D rows as the original request.

For right unit, start from an accepted completion-facing row and then apply the unit request as the second stage. The finite-window induction on W again keeps every inspected observation, readback, and dyadic scale entry unchanged.

For associativity, compare the two finite replays of a three-stage request. Both sides enumerate the same source windows, intermediate readbacks, and dyadic composition ledgers; regrouping changes only the order in which the already displayed bind rows are read.

The transport, replay, provenance, and local naming rows identify these displayed packets componentwise. No step appeals to quotient equality of streams, a selected limit, host real equality, countable choice, or ambient completeness. □

Theorem 2136.4 (Regular Cauchy completion Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *carrier reads the terminal seal only through the finite route*

$$W, D \longmapsto R \longmapsto U, B \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, unit and bind rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no quotient completion, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with an accepted carrier and project the seal row E. The carrier definition places E downstream of the displayed window family W, readback family R, dyadic ledger D, unit row U, and bind row B.

The handoff is therefore finite. A $\mathsf{Real}^{\uparrow}$ consumer must first read the inspected windows, then the regular rational readbacks, then the dyadic tolerances that justify the unit and bind requests.

Only after those rows are present may the seal row E be consumed. The seal is consequently attached to the displayed regular Cauchy completion route rather than to a quotient completion object or a host-selected real representative.

Finally H, C, P, N preserve the route under componentwise transport, continuation replay, provenance, and local naming. Because the carrier contains no excluded coordinate, the terminal handoff cannot export quotient completion, selected limits, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2136.5 (Regular Cauchy completion finite-window induction)**.** *For every accepted* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *carrier*

$$M = (W, R, D, U, B, E, H, C, P, N),$$

induction over the finite $\mathsf{StreamName}^{\uparrow}$ *window row* W *preserves the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the unit row* U*, the bind row* B*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff* E*. The induction route is closed by* H, C, P, N *and cannot introduce quotient stream equality, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use induction on the displayed finite window row W. In the empty-window case, the carrier projection gives the empty readback segment, empty tolerance segment, vacuous unit segment, vacuous bind segment, and no independent seal read. The structural rows H, C, P, N preserve exactly that empty route.

For a successor window, reuse the displayed replay route

$$W, D \longmapsto R \longmapsto U, B \longmapsto E.$$

The induction hypothesis preserves the previous window segment. The new window entry is then read with its dyadic tolerance from D, translated through R, admitted by the unit row U, composed by the bind row B, and finally exposed to the seal row E.

Close the step with the structural rows. Componentwise hsame transport in H, displayed Cont replay in C, Pkg provenance in P, and the local NameCert row N preserve only the rows already displayed in the carrier. Since there is no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness, the induction cannot introduce any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, unit rows, bind rows, a terminal RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2136.1, Definition 2136.2, Definition 2136.3, Definition 2136.4, and Definition 2136.5 seed the finite-window completion interface from StreamName windows through RegSeqRat readbacks, DyadicRatCore tolerances, unit and bind rows, RealUp seal handoff, finite-window induction, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionMonad}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full $\mathsf{Real}^{\uparrow}$ completeness, quotient completion, quotient stream equality, selected limits, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window unit identity, bind composition, tolerance replay, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionMonadUp.

Chapter 2137

Regular Cauchy Tail Sum NameCert Construction

Following Bishop and Bridges-style constructive analysis, the $\mathsf{RegularCauchyTailSum}^{\uparrow}$ packet records the finite tail-sum route by which a displayed dyadic budget, a stream window, and a regular rational sequence read determine a real endpoint seal. It sits between the dyadic tolerance surface of chapter 291, the finite observation windows of chapter 126, the regular rational sequence carrier of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite evidence needed to pass from bounded tail sums to a regular Cauchy read; it does not import quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2137.1 (Regular Cauchy tail-sum carrier)**.** *A* $\mathsf{RegularCauchyTailSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, R, A, W, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-budget ledger. It records the dyadic precision request, the split between initial-window and tail-window tolerances, and the finite comparisons used to absorb the tail.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It records the inspected stream entries, partial sums, and endpoint reads for the requested finite window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row. It records the regular rational sequence read exposed by the same finite window and tolerance budget.*
4. *A is the additivity row. It records the finite addition route for the displayed partial tail sums and the dyadic error terms.*
5. *W is the window-control row. It records the left endpoint, right endpoint, and width certificate for every tail window used by the packet.*
6. *B is the bounded-tail row. It records the displayed tail bound that converts the inspected finite window into a regular Cauchy read.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after D, S, R, A, W, B have been displayed.*
8. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regularity row, additivity row, window-control row, bounded-tail row, endpoint seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay of the route*

$$D, S, R, A, W, B \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, A, W, B, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limiting real, countable choice, ambient completeness, or a host infinite summation principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailSumUp.RegularCauchyTailSumTasteGate single carrier alignment`

Theorem 2137.2 (Regular Cauchy tail-sum window extraction)**.** *For every accepted* $\mathsf{RegularCauchyTailSum}^{\uparrow}$ *carrier*

$$T = (D, S, R, A, W, B, E, H, C, P, N),$$

each tail-sum request factors through a finite BHist/Cont *window whose visible data are exactly the dyadic budget D, stream observations S, regularity readback R, additivity row A, window-control row W, bounded-tail row B, and the structural rows H, C, P, N. No request can read a quotient stream, selected limit, countable-choice witness, ambient completeness proof, or host infinite summation theorem.*

Proof. Project the accepted packet $T = (D, S, R, A, W, B, E, H, C, P, N)$. The carrier definition exposes precisely the dyadic ledger, stream observations, regularity row, finite additivity row, window-control row, bounded-tail row, endpoint seal, and structural rows.

The inspected window is finite. The row D fixes a dyadic tolerance budget, while S lists the stream entries and partial sums actually read inside that budget. The row W records the endpoints and width certificates for the same inspected interval.

The tail stage is also finite. The additivity row A combines only the displayed partial sums and dyadic errors, and the bounded-tail row B records the finite inequality that absorbs all entries beyond the inspected window at the requested precision.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no carrier coordinate stores quotient equality, selected limits, choice, completeness, or host summation, the request cannot escape the displayed finite window. □

Theorem 2137.3 (Regular Cauchy tail-sum regular sequence read)**.** *Every accepted* $\mathsf{RegularCauchyTailSum}^{\uparrow}$ *packet determines its regular rational sequence read through the finite route*

$$D, S, W \longmapsto A, B \longmapsto R.$$

For any two inspected windows with dyadic budgets recorded by D, the corresponding reads in R agree up to the displayed tail budget after transport by H. Thus regularity is witnessed by finite tail-window induction over BHist *and* Cont *rows, not by quotient streams, selected limits, countable choice, or ambient completeness.*

Proof. Begin with two inspected tail windows admitted by the same packet. Their budgets are both read through D, and their stream observations and window widths are read through S and W.

Induct over the finite extension from the shorter displayed window to the longer displayed window. The base case is the common window, where the two reads use the same rows and the same dyadic budget. In the step case, adding one displayed tail entry updates the partial sum through A and tightens the remaining error through B.

The regularity row R receives only the data produced by this finite induction. Componentwise transport in H preserves the dyadic budget, stream observations, additivity row, and bounded-tail row, so the two reads agree up to the displayed tolerance.

The route never asks for a quotient stream representative or a selected limit. It uses only the inspected BHist/Cont windows and the finite dyadic tail budget already present in the packet. □

Theorem 2137.4 (Regular Cauchy tail-sum real seal)**.** *Every accepted* $\mathsf{RegularCauchyTailSum}^{\uparrow}$ *packet reaches its* $\mathsf{Real}^{\uparrow}$ *endpoint seal only through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow} \longmapsto A, W, B \longmapsto E, H, C, P, N.$$

Consequently the packet exports a finite tail-sum seal for later L10 Real and Completion routes, but it exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an endpoint request for the packet. The request must first expose D, S, R because these are the only rows carrying dyadic budgets, stream observations, and regular rational sequence reads.

The finite tail computation then passes through A, W, B. The row A records finite addition of displayed terms and error terms, W records the inspected tail-window widths, and B records the tail bound that converts the finite window into a regular Cauchy read.

Only after those rows are displayed can E provide the $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, P, N preserve and replay the same displayed route, so the endpoint seal is downstream of finite evidence rather than an ambient completeness principle.

Thus the packet supplies a named finite handoff for later L10 Real and Completion consumers while explicitly leaving $\mathsf{Real}^{\uparrow}$ completeness outside the local claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailSum}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailSum}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tail-budget ledger, StreamNameUp observation windows, RegSeqRatUp regular rational readback, a finite additivity row, a window-control row, a bounded-tail row, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2137.1, Definition 2137.2, Definition 2137.3, and Definition 2137.4 seed the finite tail-sum certificate surface between dyadic budgets, stream windows, regular rational reads, and RealUp endpoint seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a host infinite summation principle.

Upgrade path: Obligation closure requires checked carrier admission, finite window extraction, regular sequence readback, tail-bound stability, endpoint non-escape, and provenance closure for BEDC.Derived.RegularCauchyTailSumUp.

Chapter 2138

A Concrete Naming Certificate for $\mathsf{DedekindCutCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DedekindCutCompletion}^{\uparrow}$ packet names the finite comparison surface where located Dedekind-cut evidence may be admitted beside regular Cauchy real names without turning $\mathsf{Real}^{\uparrow}$ into a completed Dedekind quotient. It is positioned after the cut-to-Cauchy boundary of chapter 1745, the regular Cauchy completion route of chapter 1639, the dyadic comparison spine of chapter 1672, the regular rational sequence row of chapter 301, the finite stream interface of chapter 126, and the terminal real-name seal of chapter 38. The chapter records how cut-shaped data enters BEDC as finite rows and is compared only through regularization windows, preserving the refusal to treat $\mathsf{Real}^{\uparrow}$ completeness as already proved.

Definition 2138.1 (Dedekind cut completion carrier)**.** *A* $\mathsf{DedekindCutCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, U, G, W, S, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located lower-cut row. It records finite lower observations together with their downward-use discipline.*
2. *U is the located upper-cut row. It records finite upper observations together with their upward-use discipline.*
3. *G is the dyadic bracket-witness row. It records the displayed dyadic gaps separating admitted lower and upper observations.*
4. *W is the finite regularization-window row. It records the bounded window in which cut evidence is translated into regular Cauchy observations.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row opened by the regularization window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy approximant row read from S under the bracket witnesses in G.*
7. *A is the* $\mathsf{DyadicArchimedean}^{\uparrow}$ *comparison row. It stores the finite Archimedean dyadic scale used to compare bracket witnesses with regular Cauchy tolerances.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is visible only after L, U, G, W, S, R, A have been displayed.*

9. *H records componentwise* hsame *transport for the cut rows, bracket witnesses, regularization windows, stream observations, regular approximants, dyadic comparison row, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a located cut request through $L, U, G, W, S, R, A, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{DedekindCutCompletion}^{\uparrow}$, $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$, BishopRegularCauchyCom DyadicArchimedean$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through finite regularization windows W and the rows displayed downstream of those windows. The carrier has no coordinate for quotient real equality, arbitrary suprema, countable choice, trichotomy, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DedekindCutCompletionUp.DedekindCutCompletionTasteGate single carrier alignment`

Theorem 2138.2 (Located cut regularization)**.** *For every accepted* $\mathsf{DedekindCutCompletion}^{\uparrow}$ *carrier $D = (L, U, G, W, S, R, A, E, H, C, P, N)$, each located cut-side request admitted by L, U, G has a displayed finite regularization route*

$$L, U, G \longrightarrow W \longrightarrow S, R, A \longrightarrow E.$$

The route reads lower and upper cut rows only through the bracket witnesses in G, chooses a bounded regularization window W, exposes stream-window observations in S, reads regular Cauchy approximants in R, compares tolerances through the dyadic Archimedean row A, and reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal E only after those rows are displayed.*

Proof. Carrier projection gives the cut-side rows L, U, G and the regular Cauchy rows W, S, R, A, E. The lower and upper observations are not ambient cuts; they are finite rows whose only displayed meeting point is the dyadic bracket-witness row G.

The finite window row W is therefore the first comparison surface. It bounds the portion of the cut evidence that a consumer may regularize, so the route never needs a global supremum or a selected cut value.

Inside W, the stream row S supplies the inspected observations, the regular sequence row R supplies the regular Cauchy approximants, and the dyadic Archimedean row A supplies the tolerance comparison scale. Only after these rows are visible can the route read the terminal $\mathsf{Real}^{\uparrow}$ seal E. □

Theorem 2138.3 (Dedekind cut completion NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutCompletion}^{\uparrow}$ *carrier $D = (L, U, G, W, S, R, A, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located lower row, located upper row, dyadic bracket witnesses, regularization windows, stream observations, regular Cauchy approximants, dyadic Archimedean comparison row, Real seal, transport, replay, provenance, and local naming row;*

2. *located cut discipline, requiring every lower-row and upper-row read to meet only through the displayed bracket witnesses in G;*

3. *finite regularization, requiring every comparison-facing cut request to choose a displayed window W before reading S, R, A;*

4. *regular Cauchy handoff, requiring stream-window observations, regular approximants, and dyadic tolerance comparisons to be displayed before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is visible;*
5. *classifier stability, requiring componentwise* hsame *transport in* H *and displayed* Cont *replay in* C *to preserve the route* L, U, G, W, S, R, A, E*;*
6. *ledger exhaustion, requiring* P *and* N *to name exactly the displayed packet and no quotient real equality, arbitrary supremum, countable-choice schedule, trichotomy principle, host-completeness theorem, or proof that* $\mathsf{Real}^{\uparrow}$ *is complete.*

No $\mathsf{DedekindCutCompletion}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2138.1. The finite packet exposes the lower row, upper row, bracket witnesses, regularization windows, stream observations, regular approximants, dyadic comparison data, Real seal, and structural rows H, C, P, N.

Located cut discipline follows from the absence of an ambient cut coordinate. Lower and upper observations can be compared only where G supplies a displayed bracket witness.

Finite regularization is forced by the row order. A comparison-facing request must choose W before it can open stream observations, regular Cauchy approximants, or dyadic tolerance comparisons.

The regular Cauchy handoff then reads S, R, A before E. Componentwise hsame transport in H preserves each displayed row, and Cont replay in C repeats the same route for each accepted request.

Finally, provenance and local naming are exhausted by P and N. Since none of the excluded surfaces is a coordinate of D, the local certificate cannot export such a surface as a hidden obligation. □

Theorem 2138.4 (Dedekind cut and Cauchy comparison)**.** *Let* $D = (L, U, G, W, S, R, A, E, H, C, P, N)$ *and* $D' = (L', U', G', W', S', R', A', E', H', C', P', N')$ *be accepted* $\mathsf{DedekindCutCompletion}^{\uparrow}$ *carriers. Any classifier comparison between* D *and* D' *is determined by displayed finite regularization windows and their downstream regular Cauchy rows:*

$$W, S, R, A, H, C, P, N \quad \text{and} \quad W', S', R', A', H', C', P', N'.$$

The comparison may transport located cut evidence from L, U, G *and* L', U', G' *into those windows, but it cannot compare packets by quotient real equality, arbitrary suprema, countable choice, trichotomy, host completeness, or the assertion that* $\mathsf{Real}^{\uparrow}$ *is complete.*

Proof. Start with an accepted classifier comparison. By Definition 2138.1, the only comparison-facing rows are the displayed regularization windows, stream observations, regular approximants, dyadic comparison rows, and structural transport, replay, provenance, and naming rows.

The cut evidence contributes only through L, U, G and L', U', G'. The bracket witnesses in G and G' determine what finite cut-side information enters W and W', but they do not create quotient real equality or a host supremum.

Once the windows are fixed, the comparison is regular Cauchy in form: it reads S, R, A and S', R', A' under the componentwise transport and replay rows. The terminal seals E and E' are reached only after those rows are displayed.

The excluded principles have no coordinates in either carrier. Therefore the classifier compares only through the finite regularization windows and their downstream rows. □

Theorem 2138.5 (Dedekind cut completion window determinacy)**.** *Let* $D = (L, U, G, W, S, R, A, E, H, C, P, N)$ *and* $D' = (L', U', G', W', S', R', A', E', H', C', P', N')$ *be accepted* $\mathsf{DedekindCutCompletion}^{\uparrow}$ *carriers. Suppose the displayed lower rows, upper rows, and bracket-witness rows agree through componentwise transport. Then any accepted classifier comparison between* D *and* D' *must choose matching finite regularization windows before it can read* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicArchimedean}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows.*

Proof. Project both carriers by Definition 2138.1. The only comparison-facing route from located cut data to a real-facing seal passes through the lower row, upper row, bracket witnesses, and then the finite regularization window.

Use the componentwise transport hypothesis on L, U, G and L', U', G'. It makes the cut-side evidence entering W and W' the same displayed evidence, so an accepted classifier route cannot select different window data without leaving the carrier rows.

Apply Definition 2138.4. Once the windows are fixed, every comparison reads the stream rows, regular Cauchy approximants, dyadic Archimedean comparison rows, and terminal real seals downstream of those windows.

The carrier has no coordinate for quotient real equality, arbitrary suprema, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Those principles therefore cannot replace the displayed window choice or break its transport stability. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2138.1, Definition 2138.2, Definition 2138.3, Definition 2138.4, and Definition 2138.5 seed the finite comparison packet admitting located Dedekind-cut evidence through regular Cauchy windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, arbitrary suprema, countable choice, trichotomy, host completeness, Lean verification, bridge checking, or that $\mathsf{Real}^{\uparrow}$ is complete.

Upgrade path: Lean verification requires a theorem for carrier admission, located cut discipline, finite regularization, regular Cauchy handoff, classifier stability, and ledger exhaustion for BEDC.Derived.DedekindCutCompletionUp.

Constructive story: A $\mathsf{DedekindCutCompletion}^{\uparrow}$ packet is generated from located lower and upper cut rows, dyadic bracket witnesses, finite regularization windows, StreamName observations, RegSeqRat approximants, a DyadicArchimedean comparison row, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2139

A Concrete Naming Certificate for EilenbergMooreCompletionAlgebra$^{\uparrow}$

Following Eilenberg and Moore, and the standard monad-algebra treatment of completion monads, the EilenbergMooreCompletionAlgebra$^{\uparrow}$ packet names the finite L10 surface where completion monad data can be consumed as an algebra object rather than as an external quotient completion. It is grounded in the completion monad row of chapter 378, the unit row of chapter 1527, the counit landing of chapter 1539, the idempotent law surface of **??**, and the Kleisli comparison of chapter 2066. Its analytic landing is the metric-completion and real-readback chain through chapter 1506, chapter 301, and chapter 38.

The packet does not construct a quotient of Cauchy names. It records only the finite carrier, law, Kleisli-consumption, and ledger rows by which a downstream Real or CompleteMetric consumer can read the completion monad as an Eilenberg–Moore algebra surface.

Definition 2139.1 (Eilenberg–Moore completion algebra carrier)**.** *An* EilenbergMooreCompletionAlgebra$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, U, K, I, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* CauchyCompletionMonad$^{\uparrow}$ *row, displaying the completion monad source, unit, bind, tolerance, readback, and terminal seal data available to the algebra.*
2. *U is the* CauchyCompletionUnit$^{\uparrow}$ *row, displaying the finite unit admission by which an input name is copied into the completion surface.*
3. *K is the Kleisli or extension row inherited from the* CauchyCompletionKleisliAdjunction$^{\uparrow}$ *comparison, displaying the finite consumer request that reads completion data through the monad route.*
4. *I is the idempotent or multiplication law row from* CauchyCompletionIdempotent$^{\uparrow}$*, recording that a second completion pass is consumed by the same finite completion surface rather than by a selected quotient representative.*
5. *L is the* MetricCompletion$^{\uparrow}$ *landing row, exposing the metric-completion readback that a downstream Real or CompleteMetric consumer may inspect after M, U, K, I have been displayed.*
6. *H records componentwise* hsame *transport for the monad, unit, Kleisli, law, landing, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay of the route*

$$M, U \longmapsto K, I \longmapsto L \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* EilenbergMooreCompletionAlgebra$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionCounit$^{\uparrow}$, CauchyCompletionIdempotent$^{\uparrow}$, CauchyCompletionKleisliAdju MetricCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{EilenbergMooreCompletionAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, K, I, L, H, C, P, N$. The carrier has no coordinate for quotient completion, selected limits, countable choice, host equality of completion names, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EilenbergMooreCompletionAlgebraUp.EilenbergMooreCompletionAlgebraTasteGate`
`single carrier alignment`

Theorem 2139.2 (Eilenberg–Moore completion algebra NameCert obligations)**.** *For every accepted* EilenbergMooreCompletionAlgebra$^{\uparrow}$ *carrier*

$$A = (M, U, K, I, L, H, C, P, N),$$

the local NameCert$_{\text{EilenbergMooreCompletionAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion monad row, unit row, Kleisli or extension row, idempotent or multiplication law row, metric-completion landing row, transport, replay, provenance, and local naming row;*
2. *unit exposure, requiring U to be displayed before any algebra consumer can read the Kleisli or law rows;*
3. *Kleisli consumption, requiring K to consume only the displayed completion monad and unit data before opening the algebra law surface;*
4. *law exactness, requiring I to identify the second completion pass with the same displayed algebra surface through finite idempotent or multiplication rows;*
5. *landing exactness, requiring L to be the only metric-completion landing visible downstream of the monad, unit, Kleisli, and law rows;*
6. *structural exhaustion, requiring H, C, P, N to preserve, replay, source, and name the listed rows without adding quotient completions, selected limits, countable choice, host equality, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2139.1. The accepted packet displays exactly the completion monad, unit, Kleisli or extension, idempotent or multiplication, metric-completion landing, and structural rows.

Unit exposure is read from U. Since U is displayed before the Kleisli and law rows, an algebra consumer starts from the same finite admission route as the completion unit packet.

Kleisli consumption is read from K. The row records a finite consumer request over the displayed monad and unit data, so it cannot introduce a second source family, a selected limit, or a quotient representative before the law row is opened.

Law exactness is read from I. The row identifies the second completion pass with the displayed algebra surface by replaying the idempotent or multiplication route already available in the sibling completion chapters.

Landing exactness is read from L. The metric-completion landing is downstream of M, U, K, I and therefore receives only the displayed algebra route.

Finally H, C, P, N add no analytic, quotient, Lean, or bridge coordinate. They transport, replay, source, and name the same finite route, so no obligation can escape the displayed rows. □

Theorem 2139.3 (Eilenberg–Moore completion algebra law surface)**.** *For every accepted* EilenbergMooreCompletio *carrier*

$$A = (M, U, K, I, L, H, C, P, N),$$

the algebra law surface is the displayed finite route

$$M, U \longmapsto K, I \longmapsto L \longmapsto H, C, P, N.$$

The unit law is read by projecting U against the monad row M, and the multiplication or idempotence law is read by projecting I against the same M, U, K route. Thus the law surface is a NameCert comparison of displayed completion rows, not a quotient-completion equality, selected limit, host equality, ambient completeness theorem, Lean credential, or bridge credential.

Proof. Project the accepted carrier. The monad row M supplies the finite completion route, and U supplies the finite unit admission into that route.

The unit law is therefore a comparison of displayed M and U rows. It asks only whether the unit admission replays the source data carried by the monad surface.

Next inspect K and I. The Kleisli or extension row exposes the consumer request over the displayed monad data, and the idempotent or multiplication row records that applying completion again is consumed by the same algebra surface.

The landing row L is downstream of these law rows. A consumer can see a metric-completion landing only after the unit and multiplication or idempotence reads have already been displayed.

The structural rows H, C, P, N preserve this route componentwise. Since the packet contains no quotient, selection, host-equality, ambient-completeness, Lean, or bridge row, the algebra laws are exactly the displayed NameCert comparisons of $M, U, K, I, L, H, C, P, N$. □

Theorem 2139.4 (Eilenberg–Moore completion algebra unit and multiplication laws)**.** *For every accepted* EilenbergMooreCompletionAlgebra$^{\uparrow}$ *carrier*

$$A = (M, U, K, I, L, H, C, P, N),$$

the public algebra action is available only after the CauchyCompletionMonad$^{\uparrow}$ *row M,* CauchyCompletionUnit$^{\uparrow}$ *row U, Kleisli or extension row K, idempotent or multiplication row I,* MetricCompletion$^{\uparrow}$ *landing row L, and structural rows H, C, P, N have been displayed. The route records the finite unit and multiplication laws of the completion monad without quotient completions, selected representatives, host equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2139.1. The accepted packet exposes exactly the completion monad row M, unit row U, Kleisli or extension row K, idempotent or multiplication row I, metric-completion landing row L, and structural rows H, C, P, N.

Apply the law-surface theorem. Theorem 2139.3 reads the unit law from M, U and the multiplication or idempotence law from M, U, K, I before any landing row is public.

Use the Kleisli-consumption theorem to bind consumers to the same route. Theorem 2139.5 forces a Kleisli or extension request through $M, U, K, I, L, H, C, P, N$, so the algebra action cannot bypass the finite completion-monad rows.

Finally the ledger non-escape theorem excludes the analytic and quotient alternatives. Theorem 2139.6 leaves no coordinate for quotient completion, selected representatives, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2139.5 (Eilenberg–Moore completion algebra Kleisli consumption)**.** *Every L10 Completion consumer that reads an accepted* $\mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow}$ *carrier through a Kleisli or extension request factors through the displayed rows* $M, U, K, I, L, H, C, P, N$*. In particular, the consumer first reads the completion monad and unit rows, then the Kleisli request, then the idempotent or multiplication row, and only then the metric-completion landing and structural rows.*

Proof. Start with an L10 Completion consumer using a Kleisli or extension request. Projection of the accepted carrier supplies M and U before K is available.

The Kleisli row K consumes that displayed completion data. Its source is the finite monad-and-unit route, so the consumer cannot bypass M or U to reach a separate completion object.

The law row I is then read against the same route. It records that the second completion pass is absorbed by the displayed algebra surface rather than exported as a quotient or selected representative.

Only after M, U, K, I have been displayed does the landing row L become visible. The final rows H, C, P, N transport, replay, source, and name the same route, so the consumer factors through exactly the listed packet coordinates. □

Theorem 2139.6 (Eilenberg–Moore completion algebra ledger non-escape)**.** *For every accepted* $\mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow}$ *carrier*

$$A = (M, U, K, I, L, H, C, P, N),$$

the ledger visible to a downstream $\mathsf{Real}^{\uparrow}$ *or CompleteMetric consumer is exhausted by the displayed route of Definition 2139.3. No read of the packet can export a quotient completion, selected limit, countable-choice selector, host equality of completion names, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge-checked surface.*

Proof. Begin with a downstream Real or CompleteMetric read. The law-surface theorem forces the read through M, U, K, I before it can inspect the metric-completion landing.

The landing row L is the only analytic landing coordinate in the carrier. It is attached to the displayed monad, unit, Kleisli, and law rows, and it does not carry an additional quotient or selected-limit field.

Now inspect the structural rows. Componentwise transport H and continuation replay C preserve the same finite route, while P and N source and name only the listed completion-algebra rows.

Because the carrier predicate has no coordinate for quotient completion, selected limits, countable choice, host equality, ambient completeness, Lean verification, or bridge checking, no downstream consumer can export any of those objects from the packet ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EilenbergMooreCompletionAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2139.1, Definition 2139.2, Definition 2139.3, Definition 2139.5, and Definition 2139.6 seed the finite Eilenberg–Moore algebra surface for L10 completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{EilenbergMooreCompletionAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, host equality of completion names, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, Kleisli consumption, law exactness, metric-completion landing exactness, structural replay, and ledger non-escape for BEDC.Derived.EilenbergMooreCompletionAlgebraUp.

Constructive story: An EilenbergMooreCompletionAlgebra$^{\uparrow}$ packet is generated from a Cauchy completion monad row, unit row, Kleisli or extension row, idempotent or multiplication law row, metric-completion landing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2140

A Concrete Naming Certificate for $\mathsf{BishopIntervalEnclosure}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ packet records the finite interval-enclosure surface by which lower and upper endpoint data are read as rational enclosures, dyadic refinements, regular rational readback, and a terminal real-name seal. It sits after the located endpoint carrier of chapter 394, the rational enclosure ledger of chapter 1802, the real interval enclosure boundary of chapter 1625, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name surface of chapter 38. The packet gives real-completion and approximation consumers a named finite route for Bishop-style interval enclosures without importing host intervals, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2140.1 (BishopIntervalEnclosure carrier)**.** *A* $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, U, Q, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower enclosure row. It records the displayed lower endpoint name, lower rational bounds, and lower endpoint admission cells inherited from the* $\mathsf{LocatedInterval}^{\uparrow}$ *surface.*
2. *U is the upper enclosure row. It records the displayed upper endpoint name, upper rational bounds, and upper endpoint admission cells inherited from the same* $\mathsf{LocatedInterval}^{\uparrow}$ *surface.*
3. *Q is the rational interval ledger. It records the* $\mathsf{RationalIntervalEnclosure}^{\uparrow}$ *containment checks tying L and U to one finite interval-enclosure request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and refinement row. It records every dyadic radius, endpoint slack comparison, and refinement request consumed by the lower and upper enclosure rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational data exposed by the displayed lower and upper enclosure rows after the rational and dyadic ledgers have been read.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is reachable only after L, U, Q, D, R have been displayed, and it exposes the real-facing enclosure boundary without selecting quotient representatives.*
7. *H records componentwise* hsame *transport for lower and upper endpoint rows, rational ledger, dyadic refinement row, regular readback, real seal, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$L, U \longmapsto Q \longmapsto D \longmapsto R \longmapsto S.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalEnclosure}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalIntervalEnclosure}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalEnclosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, Q, D, R, S, H, C, P, N$. It has no coordinate for host intervals, selected real endpoints, quotient equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalEnclosureUp.BishopIntervalEnclosureTasteGate single carrier alignment`

Theorem 2140.2 (BishopIntervalEnclosure NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ *carrier*

$$E = (L, U, Q, D, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalEnclosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower enclosure row, upper enclosure row, rational interval ledger, dyadic tolerance and refinement row, regular rational readback, terminal real seal, transport, replay, provenance, and local naming;*
2. *endpoint order, requiring the displayed lower and upper enclosure rows to be admitted through the same located interval request before any rational containment row is consumed;*
3. *containment coverage, requiring Q to record the rational interval checks that place the requested real-facing datum between the lower and upper rows;*
4. *dyadic refinement, requiring D to account for every endpoint slack comparison, tolerance radius, and refinement read used by L, U, Q, R;*
5. *Real seal nonescape, requiring S to remain downstream of L, U, Q, D, R and to export no host interval, selected real endpoint, quotient equality, countable-choice selector, trichotomy branch, or ambient completeness theorem;*
6. *transport and replay stability, requiring H and C to preserve the lower endpoint, upper endpoint, rational ledger, dyadic refinement, regular readback, real seal, provenance, and local naming rows componentwise;*
7. *provenance and local naming, requiring P and N to name only the displayed Bishop interval-enclosure route.*

No Bishop interval-enclosure obligation is stored outside these rows.

Proof. Project the accepted packet $E = (L, U, Q, D, R, S, H, C, P, N)$. The projection exposes exactly the lower enclosure row, upper enclosure row, rational interval ledger, dyadic refinement row, regular rational readback, real seal, transport, replay, provenance, and local naming row.

Endpoint order is internal to the displayed located interval request. The rows L and U are read as paired endpoint-facing data before the rational interval ledger Q is available, so a consumer cannot inspect containment while ignoring either endpoint row.

Containment coverage is the role of Q. The rational ledger binds the lower and upper rows to one interval-enclosure request and records the finite containment checks that are later replayed by C.

Dyadic refinement is exhausted by D. Every tolerance radius, endpoint slack comparison, and refinement read used by L, U, Q, R is a coordinate of D, so no hidden precision source remains for a downstream consumer.

The regular readback and real seal are downstream. Row R reads only the displayed lower, upper, rational, and dyadic rows, and row S is visible only after R.

Finally H, C, P, N transport, replay, source, and name the same route. Since none of the carrier coordinates stores a host interval, selected real endpoint, quotient equality, choice selector, trichotomy branch, or ambient completeness theorem, the local NameCert surface cannot export any of them. □

Theorem 2140.3 (BishopIntervalEnclosure endpoint nonescape)**.** *Every endpoint-facing consumer of an accepted* $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ *packet factors through the displayed rows*

$$L, U, Q, D, R, S, H, C, P, N.$$

Consequently the packet exports only the finite Bishop interval-enclosure route and exports no host interval, selected real endpoint, quotient equality, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an endpoint-facing consumer of an accepted packet. The only lower endpoint access is L, and the only upper endpoint access is U, so the consumer first factors through the paired endpoint rows.

The rational enclosure stage is Q. It records the containment checks attached to the same lower and upper rows, and it provides no independent host interval object.

The dyadic and regular stages are D and R. The row D supplies the finite tolerance and refinement ledger, while R reads regular rational data only after L, U, Q, D have been displayed.

The real-facing boundary is S. Since S is downstream of the endpoint, rational, dyadic, and regular rows, it cannot serve as a selected endpoint or quotient-real witness outside the displayed route.

The structural rows close the factorization. Componentwise transport in H, continuation replay in C, provenance in P, and local naming in N preserve only the same finite route, leaving no coordinate for host intervals, selected real endpoints, quotient equality, countable choice, trichotomy, or $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalEnclosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalEnclosure}^{\uparrow}}, \mathsf{seed})$

Constructive story: A $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ packet is generated from lower and upper enclosure rows, a rational interval ledger, a DyadicRatCore tolerance and refinement row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Scope closed: Definition 2140.1, Definition 2140.2, and Definition 2140.3 seed the finite Bishop interval-enclosure route over displayed endpoint, rational ledger, dyadic refinement, regular readback, Real seal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalEnclosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host intervals, quotient reals, selected real endpoints, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint order, containment coverage, dyadic refinement, Real seal nonescape, transport and replay stability, provenance, and local naming for the $\mathsf{BishopIntervalEnclosure}^{\uparrow}$ packet.

Chapter 2141

A Concrete Naming Certificate for $\mathsf{CofinalTailMap}^{\uparrow}$

Following Kelley-style net and filter cofinal maps and Bishop–Bridges constructive analysis, the $\mathsf{CofinalTailMap}^{\uparrow}$ packet records the finite tail-reindexing surface used when a late request on one directed window must be answered by a displayed later request on another window. It sits after the finite stream interface of chapter 126, the regular rational readback row of chapter 301, the filter-tail route of chapter 333, the Cauchy-net route of chapter 589, the real-series packet of chapter 1581, and the diagonal tail-selector route of chapter 697. The packet isolates the cofinal tail map itself: a finite monotone reindexing and budget row, not a selected subsequence, not a quotient tail, and not an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 2141.1 (CofinalTailMap carrier)**.** *A* $\mathsf{CofinalTailMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, J, U, V, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the source index-window row. It records the finite source tail cuts, inspected source indices, and local order comparisons available to the request.*
2. *J is the target tail-window row. It records the finite target cuts and target requests that must be reached by cofinal reindexing.*
3. *U is the source-tail service row. It names the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, *or* $\mathsf{RealSeries}^{\uparrow}$ *source window whose late request is being transported.*
4. *V is the target-tail service row. It names the corresponding target window after reindexing and records the finite tail preservation data exposed to downstream consumers.*
5. *M is the monotone map row from the displayed source window to the displayed target window. It stores only finite comparisons proving that later source indices are sent to later target indices on the rows under inspection.*
6. *H records componentwise* hsame *transport for the source window, target window, source-tail service, target-tail service, monotone map, cofinality budget, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the reindexing route*

$$I, J, U, M \longmapsto V.$$

8. *P is the* Pkg *provenance over* $\mathsf{CofinalTailMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailMap}^{\uparrow}}$ *row. It includes the finite cofinality-budget row saying that every displayed target tail request in J is reached by some displayed source cut in I through M.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through the displayed source windows, target tail windows, monotone map row, cofinality-budget entry inside N, tail-window preservation data in V, componentwise transport, continuation replay, provenance, and local naming. It has no coordinate for selected subsequences, host functions, quotient tails, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CofinalTailMapUp.CofinalTailMapTasteGate single carrier alignment`

Theorem 2141.2 (CofinalTailMap NameCert obligations)**.** *For every accepted* $\mathsf{CofinalTailMap}^{\uparrow}$ *carrier $T = (I, J, U, V, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalTailMap}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source index windows, target tail windows, source-tail service, target-tail service, monotone map row, transport, replay, provenance, and local naming row;*
2. *monotone transport, requiring every displayed comparison in I to be preserved by M as a displayed comparison in J;*
3. *cofinal reachability, requiring every target tail request retained by J to have a source cut in I whose image under M reaches that request within the budget recorded by N;*
4. *tail-window preservation, requiring V to read only the target windows obtained from U through the displayed map row and cofinality budget;*
5. *classifier stability, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve I, J, U, V, M and the budget row inside N;*
6. *ledger nonescape, requiring P and N to source exactly the displayed packet and no selected subsequence, host function, quotient-tail equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No $\mathsf{CofinalTailMap}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.CofinalTailMapUp.CofinalTailMapCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 2141.1. The accepted packet exposes the source window, target window, two service rows, monotone map, and structural rows H, C, P, N.

Monotone transport is part of the displayed map row M. Since M is finite, every comparison it can use is already named by I and every transported comparison is already named by J.

Cofinal reachability is the budget component of N. For each retained target-tail request, the budget supplies a displayed source cut and records that its image under M reaches the target request.

The target-tail service row V is downstream of this route. It can read a target window only after the source service U, the map row M, and the budget entry in N have been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for selected subsequences, host functions, quotient tails, countable choice, or ambient real completeness, the six listed rows exhaust the local certificate surface. □

Theorem 2141.3 (CofinalTailMap regular-Cauchy handoff)**.** *Let $T = (I, J, U, V, M, H, C, P, N)$ be an accepted* $\mathsf{CofinalTailMap}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *tail request transported through T factors through the displayed source window I, the monotone map row M, the cofinality budget recorded by N, and the target tail window J before it can be read through V. The handoff exports a finite reindexing route for regular-Cauchy and stream-name consumers, and it exports no countable choice, quotient equality, selected subsequence, host function, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Proof. Start with a transported $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{StreamName}^{\uparrow}$ tail request. The source side is the displayed window I together with the source-tail service row U.

The request can move toward the target side only through M. The row M supplies the finite monotone comparisons needed to send the requested source cut to a target cut in J.

Next read the cofinality budget inside N. It records that the displayed target request is reached by a source cut already present in I, so the consumer cannot introduce a separate selected subsequence or choice schedule.

The target-tail service row V is then the only readable target surface. Componentwise transport in H, replay in C, and provenance in P keep the route attached to the same displayed packet.

Therefore the regular-Cauchy or stream-name handoff factors through I, M, N, J, V and contains no quotient tail, host map equality, selected subsequence, countable-choice datum, or completeness principle for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailMap}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2141.1, Definition 2141.2, and Definition 2141.3 seed the finite cofinal tail-reindexing packet over source windows, target windows, monotone map rows, cofinality budgets, tail-window preservation, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subsequences, host functions, quotient tails, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone transport, cofinal reachability, tail-window preservation, classifier stability, ledger nonescape, and the regular-Cauchy handoff for BEDC.Derived.CofinalTailMapUp.

Constructive story: A $\mathsf{CofinalTailMap}^{\uparrow}$ packet is generated from source index windows, target tail windows, source and target tail services, one finite monotone map row, a cofinality-budget row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2142

A Concrete Naming Certificate for $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$ packet records the finite route by which an explicit Archimedean bound on a displayed real name supplies a stable Cauchy-tail threshold envelope for regular rational names. It sits after the explicit bound surface of chapter 430, the threshold modulus of chapter 1754, the comparison route of chapter 1738, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the finite route from a displayed Archimedean row to a regular Cauchy tail threshold; it does not select a least bound, identify quotient streams, invoke trichotomy, use countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2142.1 (Archimedean Cauchy-tail carrier)**.** *An* $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, A, M, E, W, Q, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the displayed* $\mathsf{Real}^{\uparrow}$ *name whose tail request is being bounded. It records only the real-name seal visible through the BEDC carrier surface.*
2. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row. It records the explicit integer or dyadic bound exposed for the displayed real name, together with the finite comparison budget used to read it.*
3. *M is the* $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *threshold row. It records the finite threshold requested by a regular Cauchy tail budget after the bound row has been displayed.*
4. *E is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *comparison row. It records the finite equivalence route that transports the threshold from the displayed regular name to the requested tail formulation.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the inspected finite stream windows, the requested tail cut, and the observation schedule consumed by A, M, E.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational observations read from the same windows and threshold request.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic radius, slack, denominator budget, and comparison witnesses used by the bound and tail rows.*

8. *H records componentwise* hsame *transport for the real name, Archimedean bound, threshold modulus, tail equivalence, stream windows, regular rational readback, dyadic ledger, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the route*

$$R, A, W, D \longmapsto M \longmapsto E \longmapsto Q.$$

10. *P is the* Pkg *provenance over* ArchimedeanCauchyTail$^{\uparrow}$, Real$^{\uparrow}$, ArchimedeanReal$^{\uparrow}$, RegularCauchyTailModulus$^{\uparrow}$, CauchyTailEquivalence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ArchimedeanCauchyTail}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $R, A, M, E, W, Q, D, H, C, P, N$ and has no coordinate for host supremum data, selected least bounds, quotient stream equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ArchimedeanCauchyTailUp.ArchimedeanCauchyTailTasteGate single carrier alignment`

Theorem 2142.2 (Archimedean Cauchy-tail NameCert obligations)**.** *For every accepted* ArchimedeanCauchyTail$^{\uparrow}$ *carrier*

$$T = (R, A, M, E, W, Q, D, H, C, P, N),$$

the local NameCert$_{\text{ArchimedeanCauchyTail}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed real name, Archimedean bound row, tail-threshold modulus, tail-equivalence comparison, stream windows, regular rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming row;*

2. *bound-row exposure, requiring A to display the explicit Archimedean bound for R before any threshold request is answered;*

3. *threshold extraction, requiring M to obtain the regular Cauchy tail threshold only from the displayed bound row, stream windows, and dyadic budget;*

4. *tail-equivalence stability, requiring E to transport the extracted threshold across the displayed Cauchy-tail comparison without adding quotient stream equality or a selected limit;*

5. *ledger nonescape, requiring D, H, C, P, N to preserve the displayed route and to export no host supremum, least-bound selector, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Archimedean-to-tail obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.ArchimedeanCauchyTailUp.ArchimedeanCauchyTailCarrier namecert obligations`

Proof. Project the accepted packet $T = (R, A, M, E, W, Q, D, H, C, P, N)$. The carrier exposes exactly the displayed real name, the Archimedean bound row, the regular Cauchy threshold row, the Cauchy-tail comparison row, the stream-window row, the regular rational readback row, the dyadic ledger, and the structural rows.

The bound exposure is first in the route. A tail request can inspect R only through the bound row A, and A records an explicit displayed bound together with the finite comparison budget that supports it.

The threshold extraction is then local. The row M reads the already displayed bound, the windows in W, and the tolerance data in D to produce a regular Cauchy tail threshold. No detached threshold source is present in the carrier.

The comparison row E transports that same threshold across the displayed Cauchy-tail equivalence. Since E has only the comparison data named in the tuple, it cannot replace the route by quotient stream equality or by a selected limiting representative.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Because the classifier has no coordinate for host supremum data, selected least bounds, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking, the five listed rows exhaust the local NameCert surface. □

Theorem 2142.3 (Archimedean Cauchy-tail threshold envelope)**.** *For every accepted* $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$ *carrier*

$$T = (R, A, M, E, W, Q, D, H, C, P, N),$$

every budgeted tail request first reads the Archimedean bound row A for the displayed real name R and then obtains its Cauchy threshold through the $\mathsf{RegularCauchyTailModulus}^{\uparrow}$ *row M and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *row E. The request may inspect only the shared* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D, and the structural rows H, C, P, N.*

Proof. Begin with a budgeted tail request for the accepted packet. The request names the displayed real row R, but the only row that exposes a bound for R is A.

After A is read, the dyadic budget and finite observation schedule are fixed by D and W. The row M converts exactly those displayed data into the regular Cauchy threshold required by the request.

The row E then transfers the threshold across the displayed Cauchy-tail equivalence. The readback row Q records the regular rational observations reached by the same windows and threshold.

The structural rows H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence the request factors through A, then M and E, and no other threshold source is available. □

Theorem 2142.4 (Archimedean Cauchy-tail nonescape boundary)**.** *Every consumer of an accepted* $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$ *carrier is confined to the displayed route*

$$R, A, W, D \longmapsto M \longmapsto E \longmapsto Q \longmapsto H, C, P, N.$$

Consequently the packet excludes host supremum data, a selected least bound, quotient stream equality, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness from the Archimedean-to-tail handoff.*

Proof. Let a consumer read an accepted carrier. The carrier definition permits the consumer to inspect only $R, A, M, E, W, Q, D, H, C, P, N$.

The real-name and bound part of the route is R, A, W, D. It displays a concrete bound row and the finite windows and dyadic budget needed to read it, but it stores neither a host supremum nor a least-bound selector.

The tail part of the route is M, E, Q. It extracts a regular Cauchy threshold and transports it through the displayed tail-equivalence row before exposing the regular rational readback.

The structural part of the route is H, C, P, N. These rows preserve and source the same finite packet, so they cannot add quotient stream equality, trichotomy, countable choice, or ambient real completeness.

Thus every accepted consumer remains inside the displayed route, and each listed escape principle is absent from the carrier surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTail}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCauchyTail}^{\uparrow}$ packet is generated from a displayed RealUp name, an ArchimedeanRealUp bound row, a RegularCauchyTailModulusUp threshold row, a CauchyTailEquivalenceUp comparison row, StreamNameUp windows, RegSeqRatUp readback, a DyadicRatCoreUp tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTail}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2142.1, Definition 2142.2, Definition 2142.3, and Definition 2142.4 seed the finite bridge from explicit Archimedean bounds to regular Cauchy tail thresholds.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyTail}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host supremum data, selected least bounds, quotient stream equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bound-row exposure, threshold extraction, tail-equivalence stability, ledger nonescape, replay confinement, provenance, and local naming for BEDC.Derived.ArchimedeanCauchyTailUp.

Chapter 2143

A Concrete Naming Certificate for $\mathsf{SternBrocotInterval}^{\uparrow}$

Following Hardy–Wright and Khinchin on Stern–Brocot and continued-fraction rational approximation, the $\mathsf{SternBrocotInterval}^{\uparrow}$ packet records the finite rational-bracketing surface where adjacent Stern–Brocot or Farey data become a named interval before regular rational and real-name consumers read it. It sits beside the branch certificate of chapter 1704, the adjacency ledger of chapter 1759, the dyadic arithmetic core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a finite bracketing NameCert surface for the open Real and Completion phase; it is not a quotient-real equality theorem, a trichotomy principle, a countable-choice selector, a selected representative, or an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2143.1 (Stern–Brocot interval carrier)**.** *A* $\mathsf{SternBrocotInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, R, W, F, S, M, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left rational endpoint row. It is read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *data recording numerator, denominator, sign, scale, and the endpoint tolerance ledger.*
2. *R is the right rational endpoint row, read through the same dyadic rational surface.*
3. *W is the bracket-witness row. It records denominator positivity, endpoint order, the displayed gap budget, and the fact that all later consumers inspect only the interval between L and R.*
4. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *neighbor row when the bracket is supplied by adjacent Farey data.*
5. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *branch row when the same bracket is replayed by a finite left-right Stern–Brocot word.*
6. *M is the mediant refinement row. It records a finite list of mediant insertions, each with the two parent endpoints, the inserted rational, and the child bracket selected for the next row.*
7. *D is the dyadic tolerance row. It compares the displayed rational gap against the finite dyadic budgets requested by downstream stream and regular rational consumers.*
8. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *window handoff row. It records the finite stream observations and the regular rational readback available only after L, R, W, F, S, M, D have been displayed.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only downstream of the bracket, mediant, dyadic, stream, and regular rational rows.*

10. *H records componentwise* hsame *transport for endpoint, bracket, Farey, Stern–Brocot, mediant, dyadic, window, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the finite route*

$$L, R \longmapsto W \longmapsto F, S \longmapsto M, D \longmapsto Q, E, H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{SternBrocotInterval}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, W, F, S, M, D, Q, E, H, C, P, N$. It has no coordinate for quotient real equality, trichotomy, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SternBrocotIntervalUp.SternBrocotIntervalTasteGate single carrier alignment`

Theorem 2143.2 (Stern–Brocot interval NameCert obligations)**.** *For every accepted* $\mathsf{SternBrocotInterval}^{\uparrow}$ *carrier $I = (L, R, W, F, S, M, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SternBrocotInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed rational endpoints, bracket witnesses, Farey neighbor data, Stern–Brocot branch data, mediant refinement ledger, dyadic tolerance ledger, stream and regular rational handoff, Real seal, transport, replay, provenance, and local naming row;*

2. *endpoint exactness, requiring L and R to be read as displayed dyadic rational endpoint rows before any bracket consumer is opened;*

3. *bracket exactness, requiring W to expose denominator positivity, endpoint order, and the finite gap budget before F, S, M, D are consumed;*

4. *Farey and Stern–Brocot route compatibility, requiring F and S to name the same rational bracket when both rows are present;*

5. *mediant refinement locality, requiring every row of M to be generated from the currently displayed endpoints and to select a displayed child bracket;*

6. *window handoff exactness, requiring Q to read only finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows downstream of the bracket and dyadic tolerance rows;*

7. *Real-seal and ledger nonescape, requiring E, H, C, P, N to preserve exactly the displayed packet and to export no quotient real equality, trichotomy, countable choice, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

No Stern–Brocot interval obligation is stored outside these rows.

Proof. Project the carrier of Definition 2143.1. The analytic rows are the two rational endpoints, the bracket witness, the Farey and Stern–Brocot route rows, the mediant ledger, the dyadic tolerance row, the window handoff, and the Real seal.

Endpoint and bracket exactness follow from the row order. A consumer must read L and R as displayed rational packets and then pass through W before any refinement, window, or endpoint-facing row is visible.

The route compatibility clauses are finite. Farey adjacency and the Stern–Brocot word both name a rational interval bounded by the displayed endpoints; comparing their endpoint rows and determinant witness identifies the same bracket without invoking trichotomy or selected representatives.

The mediant and window clauses are then local projections. Each mediant row in M is generated from the current endpoints, while D and Q expose only finite dyadic budgets, stream observations, and regular rational windows.

Finally E, H, C, P, N seal, transport, replay, source, and name the same packet. Since the carrier contains no coordinate for the excluded principles, the seven listed rows exhaust the local certificate surface. □

Theorem 2143.3 (Mediant refinement)**.** *Let $I = (L, R, W, F, S, M, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{SternBrocotInterval}^{\uparrow}$ *carrier. Every mediant refinement row in M factors through the displayed bracket*

$$L, R, W \longmapsto M.$$

At each finite step the inserted rational is the mediant of the currently displayed endpoints, the selected child bracket is again recorded by endpoint rows of the same carrier shape, and the dyadic gap ledger is updated before any $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer may read the result.*

Proof. Begin with the bracket row W. It exposes the endpoint order, denominator positivity, and gap budget for the rational interval named by L and R.

Induct over the finite mediant list in M. The base row is the original bracket itself. For the successor row, the currently displayed endpoints determine the mediant numerator and denominator by finite rational arithmetic, and the row records which of the two child brackets is retained.

After each successor step, the retained endpoints remain displayed rational rows and the gap comparison is passed to D. Thus every refinement step remains inside finite endpoint, mediant, and dyadic budget data.

The window handoff is downstream of those rows. Therefore no stream, regular rational, or Real-facing consumer can inspect a refined interval without first reading the displayed mediant route. □

Theorem 2143.4 (RegSeqRat window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{SternBrocotInterval}^{\uparrow}$ *carrier factors through the finite route*

$$L, R, W, F, S, M, D \longmapsto Q.$$

Consequently the consumer receives endpoint rows, bracket witnesses, the Farey or Stern–Brocot route, finite mediant refinements, and dyadic gap budgets before receiving any regular rational window.

Proof. Project the handoff row Q. By the carrier definition it is downstream of L, R, W, F, S, M, D, so it cannot be opened from a detached stream observation or an independent regular rational sequence.

The bracket rows supply the rational bounds. The route rows F and S explain how those bounds were reached, while the mediant ledger M records the finite refinements that may narrow them.

The dyadic tolerance row D translates the displayed rational gap into the budget format expected by finite stream observations and regular rational windows. Only then does Q expose the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows.

Thus every regular rational consumer reads the bracket as a finite rational interval with a visible refinement and tolerance history, not as a quotient real, a selected representative, or a completeness result. □

Theorem 2143.5 (Real nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-phase consumer of an accepted* $\mathsf{SternBrocotInterval}^{\uparrow}$ *carrier reads the displayed chain*

$$L, R, W, F, S, M, D \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The terminal Real seal is therefore attached to a finite rational bracket, mediant refinement ledger, dyadic tolerance ledger, stream window, regular rational readback, componentwise transport, replay, provenance, and local naming row. It exports no quotient real equality, trichotomy, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a Real-facing request. The carrier exposes the seal row E only after the window handoff row Q has been displayed.

Use Definition 2143.4. The row Q itself is downstream of the endpoint rows, bracket witness, route rows, mediant ledger, and dyadic tolerance ledger, so the seal cannot bypass the finite rational interval data.

The structural rows H, C, P, N preserve the same chain by componentwise transport, displayed replay, provenance, and local naming. They add no additional analytic coordinate.

Since neither E nor the structural rows contain quotient real equality, trichotomy, countable choice, selected representatives, or ambient completeness, every Real and Completion-phase consumer remains inside the displayed NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotInterval}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SternBrocotInterval}^{\uparrow}}$, seed)

Scope closed: Definition 2143.1, Definition 2143.2, Definition 2143.3, Definition 2143.4, and Definition 2143.5 seed the finite rational-bracketing surface from Stern–Brocot and Farey rows through mediant refinement, dyadic tolerance, stream windows, RegSeqRat readback, and Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SternBrocotInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, bracket exactness, Farey and Stern–Brocot route compatibility, mediant refinement locality, RegSeqRat window handoff, Real nonescape, structural transport, provenance, and local naming for BEDC.Derived.SternBrocotIntervalUp.

Constructive story: A $\mathsf{SternBrocotInterval}^{\uparrow}$ packet is generated from left and right rational endpoint rows, a bracket witness row, FareySequence and SternBrocot route rows, a mediant refinement ledger, a DyadicRatCore tolerance row, StreamName and RegSeqRat window handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2144

A Concrete Naming Certificate for $\mathsf{PrincipalFilter}^{\uparrow}$

Following Bourbaki and Kelley on filters, the $\mathsf{PrincipalFilter}^{\uparrow}$ packet records the principal filter generated by one displayed finite base packet. It sits after the finite filter-base surface of chapter 1601, beside the cofinite route of chapter 1564, and before completion-facing consumers read the Cauchy-filter boundary of chapter 333 or the metric-completion route of chapter 1506. The packet is a finite human-origin base case for L10 filter completion routes; it is not quotient filter equality, an ultrafilter principle, a selected limit, countable choice, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2144.1 (PrincipalFilter carrier)**.** *A* $\mathsf{PrincipalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, U, M, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the generator packet. It records the single displayed base element and the finite membership witnesses saying which reads contain it.*
2. *U is the upward-closure row. It records each accepted superset read only by displaying the generator containment already carried by G.*
3. *M is the finite meet and refinement row. It records that two accepted reads refine back to the same generator packet, so their common read stays inside the displayed finite base.*
4. *R is the* $\mathsf{CauchyFilter}^{\uparrow}$ *anchor row. It exposes the principal-filter data only as a finite anchored base for completion-facing consumers.*
5. *H records componentwise* hsame *transport for the generator, upward-closure, finite meet, Cauchy-filter anchor, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the route*

$$G \longmapsto U, M \longmapsto R.$$

7. *P is the* Pkg *provenance over* $\mathsf{PrincipalFilter}^{\uparrow}$*,* $\mathsf{FilterBase}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrincipalFilter}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eight rows. Its classifier compares accepted packets only through G, U, M, R, H, C, P, N and has no coordinate for quotient filter equality, ultrafilters, selected limits, countable choice, ambient completion, $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PrincipalFilterUp.PrincipalFilterTasteGate single carrier alignment`

Theorem 2144.2 (PrincipalFilter NameCert obligation surface)**.** *For every accepted* $\mathsf{PrincipalFilter}^{\uparrow}$ *carrier*

$$F = (G, U, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrincipalFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the generator packet, upward-closure row, finite meet and refinement row, Cauchy-filter anchor row, transport, replay, provenance, and local naming row;*
2. *upward closure, requiring each accepted read to display containment of the same generator packet;*
3. *finite-meet closure, requiring two accepted reads to refine through the displayed generator into an accepted common read;*
4. *generator stability, requiring componentwise* hsame *transport to preserve the generator row before any downstream row is read;*
5. *handoff, requiring every CauchyFilter or MetricCompletion consumer to receive only the finite anchored base row R;*
6. *ledger refusal, excluding quotient filter equality, ultrafilters, selected limits, countable choice, ambient completion, and* $\mathsf{Real}^{\uparrow}$ *completeness from the carrier inventory.*

No principal-filter NameCert obligation is stored outside these rows.

Proof. Project the accepted packet $F = (G, U, M, R, H, C, P, N)$. Carrier admission is exactly the display of these eight rows, so no host filter object is introduced.

The upward-closure row reads every accepted superset through the generator packet G. The finite-meet row then uses the same generator as the common displayed refinement for any two accepted reads.

Generator stability is componentwise. The hsame row first transports G and then transports U, M, R, C, P, N over the same generator coordinate.

The handoff row is R. It is downstream of G, U, M and therefore exposes the principal filter only as a finite anchored base for CauchyFilter and MetricCompletion consumers.

Finally the carrier has no row for quotient filters, ultrafilters, selected limits, countable choice, ambient completion, or real completeness. The ledger refusal follows by inspection of the displayed inventory. □

Theorem 2144.3 (PrincipalFilter refinement handoff)**.** *Every accepted* $\mathsf{PrincipalFilter}^{\uparrow}$ *read factors through the displayed generator packet into the* $\mathsf{FilterBase}^{\uparrow}$ *finite directed-base surface. The exported route is*

$$G \longmapsto U, M \longmapsto R \longmapsto H, C, P, N,$$

where G supplies the single base element, U supplies upward closure, M supplies finite common refinement through that same generator, and R supplies the CauchyFilter anchor. Thus the handoff is a finite FilterBase refinement route, not quotient filter equality, not an ultrafilter, not a selected limit, and not countable choice.

Proof. Start with an accepted read. By carrier admission it has a generator row G, and every upward read in U records containment of that generator.

For two accepted reads, the finite-meet row M returns to the same generator packet. Hence the common refinement is already displayed inside the finite $\mathsf{FilterBase}^{\uparrow}$ surface, rather than supplied by a host filter equality.

The anchor row R exports this directed-base data to the CauchyFilter side. The transport, replay, provenance, and naming rows preserve exactly the same route, so no consumer can bypass the displayed generator. □

Theorem 2144.4 (PrincipalFilter completion anchor)**.** *Every CauchyFilter or MetricCompletion consumer of an accepted* $\mathsf{PrincipalFilter}^{\uparrow}$ *carrier receives the packet only as a finite anchored base:*

$$G, U, M \longmapsto R \longmapsto H, C, P, N.$$

The consumer may read generator containment, upward closure, finite common refinement, the CauchyFilter anchor, componentwise transport, displayed replay, package provenance, and local naming. It receives no quotient filter equality, ultrafilter extension, selected limit, countable-choice schedule, ambient completion, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Let a completion-facing consumer read the packet. The route first displays G, U, M, so the consumer sees the single generator, all upward reads, and the finite common-refinement evidence before any CauchyFilter anchor is available.

The row R is the only completion-facing anchor. It is therefore a finite base handoff to CauchyFilter and MetricCompletion consumers, not a quotient-filter carrier or a selected completion point.

The structural rows H, C, P, N preserve, replay, source, and name that same route. Since the carrier inventory contains no quotient equality, ultrafilter, selected limit, countable-choice schedule, ambient completion, or real-completeness row, the consumer cannot extract any of those principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrincipalFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{PrincipalFilter}^{\uparrow}$ packet is generated from one displayed generator packet, upward-closure reads, finite meet and refinement reads, a CauchyFilter anchor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrincipalFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2144.1, Definition 2144.2, Definition 2144.3, and Definition 2144.4 seed the finite principal-filter base case used by L10 CauchyFilter and MetricCompletion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrincipalFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilters, quotient filters, selected limits, countable choice, ambient completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upward closure, finite-meet closure, generator stability, FilterBase handoff, CauchyFilter and MetricCompletion anchor confinement, ledger refusal, provenance, and local naming for BEDC.Derived.PrincipalFilterUp.

Chapter 2145

A Concrete Naming Certificate for $\mathsf{KuratowskiClosure}^{\uparrow}$

Following Kuratowski's closure-operator axioms and standard point-set topology, the $\mathsf{KuratowskiClosure}^{\uparrow}$ packet records topological closure as a finite BEDC NameCert surface. It sits over the generated-open and neighbourhood rows of chapter 88, the located distance tests of chapter 57, and the certified sequence-limit route of chapter 1571. The packet gives topology, metric, compactness, and sequence-facing consumers a canonical closure-operator object without importing host powersets, arbitrary complements, excluded middle, selected convergent subsequences, or an ambient semantic closure theorem.

Definition 2145.1 (Kuratowski closure carrier)**.** *A* $\mathsf{KuratowskiClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, N, W, Q, H, C, P, L)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row. It records the indexed open rows, finite-intersection rows, union rows, and neighbourhood readback available to closure consumers.*
2. *S is the subset or source-membership row whose closure is being named. It records only displayed membership witnesses over the carrier read by T.*
3. *N is the neighbourhood ledger. Each row records a named neighbourhood of a candidate point, the open-row classifier that admits it, and the finite request asking whether the neighbourhood meets S.*
4. *W is the closure-membership witness row. It records, for each accepted candidate point, the finite responses showing that every displayed neighbourhood request in N reaches a displayed member row of S.*
5. *Q is the* $\mathsf{SequentialClosure}^{\uparrow}$ *comparison row. It is present only as a displayed comparison route from sequence-limit closure evidence to the same neighbourhood ledger, not as a replacement for topological closure.*
6. *H records componentwise* hsame *transport for the topology row, source row, neighbourhood ledger, membership witness rows, sequential comparison row, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay from a closure request through T, S, N, W, Q.*

8. *P is the* Pkg *provenance over* $\mathsf{KuratowskiClosure}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *L is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $T, S, N, W, Q, H, C, P, L$ and has no coordinate for host powersets, arbitrary complements, excluded middle, open-cover compactness, selected subsequences, ambient semantic closure, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.KuratowskiClosureUp.KuratowskiClosureUp`

Theorem 2145.2 (Kuratowski closure NameCert obligations)**.** *For an accepted* $\mathsf{KuratowskiClosure}^{\uparrow}$ *carrier*

$$K = (T, S, N, W, Q, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the ambient topology row, subset or source row, neighbourhood ledger, closure-membership witness rows, sequential-closure comparison row, transport, replay, provenance, and local naming row;*

2. *neighbourhood-meets-source exactness, requiring every displayed closure-membership read to pass through N and W before it can claim that a neighbourhood of the candidate point meets S;*

3. *topology-row stability, requiring finite-intersection and union reads used by the Kuratowski laws to remain inside the displayed* $\mathsf{Topology}^{\uparrow}$ *row T;*

4. *sequential comparison discipline, requiring the row Q to compare certified sequence-limit closure evidence with the same T, S, N, W ledger rather than replacing the topological closure carrier;*

5. *classifier transport, requiring* hsame *comparison to preserve the topology, source, neighbourhood, witness, comparison, replay, provenance, and local naming rows componentwise;*

6. *non-escape, excluding host powersets, arbitrary complement choice, excluded middle, open-cover compactness, selected convergent subsequences, and ambient semantic closure.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2145.1. The accepted packet exposes exactly the ambient topology row, source-membership row, neighbourhood ledger, closure-membership witness rows, sequential comparison row, and structural rows H, C, P, L.

Neighbourhood-meets-source exactness follows from the route N, W. A closure-membership read may answer a candidate point only after the named neighbourhood request in N and the displayed source-meeting response in W have both been read.

Topology-row stability is inherited from T. The finite-intersection and union rows used to express the Kuratowski closure laws are the rows already displayed by the topology packet, so no host powerset or complement surface is added.

The comparison row Q is downstream of the same ledger. It can translate certified sequence-limit evidence into neighbourhood-meeting evidence only through T, S, N, W, and therefore cannot supply arbitrary subsequence choice or quotient stream equality.

Finally H, C, P, L preserve and name only the displayed coordinates. Since the excluded host objects have no carrier row, no replay edge, and no provenance entry, the listed obligations exhaust the local NameCert surface. □

Theorem 2145.3 (Kuratowski closure idempotence)**.** *Let*

$$K = (T, S, N, W, Q, H, C, P, L)$$

be an accepted $\mathsf{KuratowskiClosure}^{\uparrow}$ *carrier, and let* $\overline{S}$ *denote the closure-membership row named by* W *inside the same packet. The closure packet generated from* $\overline{S}$ *over the unchanged topology row* T *is classified by* $\mathsf{msame}_{\mathsf{KuratowskiClosure}^{\uparrow}}$ *with the original packet* K*. Equivalently, applying the displayed closure operation twice reads the same finite neighbourhood ledger, the same membership witness rows, the same sequential-comparison route, and the same transport, replay, provenance, and local naming rows.*

Proof. Start from the accepted carrier K. A first closure read exposes the neighbourhood ledger N and the witness rows W over the source row S.

For the second closure read, the source row is the displayed closure-membership row $\overline{S}$. A neighbourhood request for a point of the second closure is answered by replaying the same neighbourhood row in N and then using the witness row in W that already certifies a meeting point for the first closure read.

The Kuratowski closure law used here is therefore idempotence of the displayed neighbourhood-meeting ledger: the second pass introduces no new topology row, no new source classifier, and no new witness coordinate. It only reuses the finite T, N, W rows.

Componentwise transport through H and replay through C identify the two packets. Provenance and local naming through P, L are unchanged, and the sequential comparison row Q remains attached to the same ledger. Hence the two closure packets are classified by $\mathsf{msame}_{\mathsf{KuratowskiClosure}^{\uparrow}}$. □

Theorem 2145.4 (Kuratowski closure finite-union handoff)**.** *Let*

$$K = (T, S, N, W, Q, H, C, P, L)$$

be an accepted $\mathsf{KuratowskiClosure}^{\uparrow}$ *carrier whose source row* S *is displayed as the binary union of two source-membership rows* S_0 *and* S_1 *through the finite-union rows of the ambient* $\mathsf{Topology}^{\uparrow}$ *packet* T*. Every closure read of that binary-union source factors through the displayed finite-union rows of* T*, the two source-membership rows* S_0, S_1*, the neighbourhood ledger* N*, the paired witness rows in* W*, and the existing transport, replay, provenance, and local naming rows* H, C, P, L*. The handoff introduces no host powerset, arbitrary complement surface, excluded-middle branch, selected subsequence, or ambient semantic closure object.*

Proof. Project the accepted carrier K. The only topology operations visible to a closure consumer are the finite-intersection and union rows already stored in T, and the source being closed is the displayed binary-union row built from S_0 and S_1.

For a neighbourhood request in N, use the finite-union row from T before reading witnesses. The request is answered by one of the two displayed source routes: either the witness row in W meets the S_0 side, or it meets the S_1 side. Both routes remain paired with the same neighbourhood ledger and therefore use the same closure-membership classifier.

Transport and replay are then inherited componentwise. The rows H and C identify the topology row, the two source-membership rows, the neighbourhood ledger, and the paired witnesses, while P and L preserve provenance and the local $\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}$ name. Since no carrier coordinate stores host powersets or complement choices, the binary-union handoff is exhausted by the displayed $T, S_0, S_1, N, W, H, C, P, L$ rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2145.1, Definition 2145.2, Definition 2145.3, and Definition 2145.4 seed the finite topological closure-operator packet over displayed topology, subset, finite-union, neighbourhood, witness, sequential-comparison, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, arbitrary complements, excluded middle, open-cover compactness, selected convergent subsequences, ambient semantic closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood-meets-source exactness, topology-row stability, finite-union handoff, sequential comparison discipline, classifier transport, idempotence, and non-escape for BEDC.Derived.KuratowskiClosureUp.KuratowskiClosureNameCert_obligations.

Constructive story: A $\mathsf{KuratowskiClosure}^{\uparrow}$ packet is generated from a TopologyUp row, a subset or source-membership row, neighbourhood ledgers, closure-membership witness rows, a SequentialClosureUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2146

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTailComparison}^{\uparrow}$ packet records the finite common-tail comparison surface used when two regular Cauchy real names must answer an equality or endpoint-comparison request through one displayed late window. It sits over the regular rational rows of chapter 301, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the route-order handoff of chapter 1568, the zero-tail witness surface of chapter 380, and the terminal seal of chapter 38. The packet gives later $\mathsf{Real}^{\uparrow}$ equality and endpoint-comparison chapters a single finite comparison object rather than repeated route-order prose; it is not quotient stream equality, a selected limit, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2146.1 (Regular Cauchy tail-comparison carrier)**.** *A* $\mathsf{RegularCauchyTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, W, D, I, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy source rows. Each is read only through its displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *surface before a real seal or comparison answer is available.*
2. *W is the common* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It records the shared late cut, inspected indices, synchronized endpoint windows, and comparison requests used for both sources.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested equality or endpoint-comparison radius, comparison slack, dyadic enclosure budget, and exhausted finite windows.*
4. *I is the* $\mathsf{CauchyReindexing}^{\uparrow}$ *route row. It records the finite reindexing handoff that places the X and Y observations in the same route order before any comparison read is exported.*
5. *Z is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail handoff row. It is present exactly when an equality request is being read, and it records the displayed tail-difference route used to certify that the two late windows collapse to a null sequence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary for the comparison read. It becomes available only after X, Y, W, D, I and, for equality reads, Z have been displayed.*
7. *H records componentwise* hsame *transport for the two source rows, shared window row, dyadic ledger, reindexing route, zero-tail row, real seal, replay, provenance, and local naming row.*

8. *C records displayed* Cont *replay of the route*

$$X, Y, W, D \longmapsto I \longmapsto Z \longmapsto E.$$

For a non-equality endpoint comparison, the Z stage is the displayed empty equality branch rather than a hidden quotient-equality read.

9. *P is the* Pkg *provenance over* RegularCauchyTailComparison$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyReindexing$^{\uparrow}$, NullSequence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{RegularCauchyTailComparison}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $X, Y, W, D, I, Z, E, H, C, P, N$ and has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailComparisonUp.RegularCauchyTailComparisonTasteGate single carrier alignment`

Theorem 2146.2 (Regular Cauchy tail-comparison NameCert obligations)**.** *For every accepted* RegularCauchyTailComparison$^{\uparrow}$ *carrier*

$$T = (X, Y, W, D, I, Z, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyTailComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy source rows, common stream-window row, dyadic tolerance ledger, reindexing route, zero-tail handoff row, terminal real seal, transport, replay, provenance, and local naming row;*

2. *common-window discipline, requiring both source rows to be read through the same W row before either equality or endpoint-comparison evidence is exported;*

3. *dyadic tolerance exactness, requiring D to provide the requested radius, slack, enclosure budget, and finite-window exhaustion consumed by the comparison read;*

4. *route-order discipline, requiring I to align the displayed X and Y observations before any* Real$^{\uparrow}$ *seal boundary or equality branch can be read;*

5. *zero-tail confinement, requiring Z to appear only for equality requests and to record the displayed* NullSequence$^{\uparrow}$ *handoff rather than quotient stream equality;*

6. *seal-boundary discipline, requiring E to remain downstream of X, Y, W, D, I and the equality branch Z when that branch is requested;*

7. *transport, replay, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise;*

8. *non-escape, excluding quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No regular Cauchy tail-comparison obligation is stored outside these rows.

Proof. Project the accepted packet $T = (X, Y, W, D, I, Z, E, H, C, P, N)$. The carrier exposes exactly the two source rows, one shared finite-window row, the dyadic tolerance ledger, the route-order handoff, the zero-tail branch, the terminal real seal, and the structural rows.

The comparison route starts with a common window. The row W fixes the late cut and inspected indices for both X and Y, so a comparison read cannot introduce two unrelated StreamName schedules.

The dyadic and route-order data are then forced. The row D supplies the precision budget read by the comparison, and I reorders the displayed observations into the common Cauchy route before the seal row can be reached.

For equality reads, the only equality-specific coordinate is Z. It records a $\mathsf{NullSequence}^{\uparrow}$ zero-tail handoff from the displayed late-window differences; for endpoint comparisons, the same carrier exposes the empty equality branch and therefore no quotient stream-equality coordinate.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite rows. Since none of the excluded host objects has a carrier coordinate, replay edge, provenance entry, or local naming row, the listed obligations exhaust the local NameCert surface. □

Theorem 2146.3 (Regular Cauchy tail-comparison common-window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *equality or endpoint-comparison read of an accepted* $\mathsf{RegularCauchyTailComparison}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y, W, D \longmapsto I \longmapsto Z \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives two displayed regular Cauchy rows, one shared StreamName finite window, DyadicRatCore tolerance accounting, a CauchyReindexing route-order handoff, the NullSequence zero-tail branch exactly for equality reads, the terminal RealUp seal boundary, componentwise transport, continuation replay, package provenance, and local naming. It receives no quotient stream equality, selected limit, host real equality, trichotomy, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted equality or endpoint-comparison read. The source rows are X and Y, and the carrier requires both rows to be observed through the common finite-window row W.

The dyadic ledger D fixes the requested radius, slack, enclosure budget, and exhausted windows attached to that shared observation. The route-order row I then aligns the two displayed regular Cauchy observations before any real seal is visible.

When the read is an equality request, the zero-tail branch Z supplies the displayed null-sequence handoff from the aligned tail differences. When the read is an endpoint comparison, the carrier uses the displayed empty equality branch, so the comparison still passes through the same route without adding a quotient-equality coordinate.

Only after the shared window, tolerance ledger, reindexing route, and relevant zero-tail branch have been displayed can E expose the $\mathsf{Real}^{\uparrow}$ seal boundary. The structural rows H, C, P, N preserve that same finite route under componentwise transport, continuation replay, provenance, and local naming.

Induction over the displayed route order proves that no later consumer can skip a row: the base stage reads X, Y, W, D, the reindexing step reads I, the equality-branch step reads Z when requested, and the terminal step reads E, H, C, P, N. Because there is no coordinate outside this route, the consumer cannot obtain quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailComparison}^{\uparrow}$ packet is generated from two regular Cauchy source rows, one shared StreamName finite-window row, a DyadicRatCore tolerance ledger, a CauchyReindexing route row, a NullSequence

zero-tail handoff when equality is requested, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailComparison}\uparrow}$, seed)

Scope closed: Definition 2146.1, Definition 2146.2, and Definition 2146.3 seed the finite common-tail comparison route over shared StreamName windows, RegSeqRat source rows, DyadicRatCore tolerances, CauchyReindexing handoff, NullSequence equality branch, RealUp seal boundary, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailComparison}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-window discipline, dyadic tolerance exactness, route-order discipline, zero-tail confinement, seal-boundary discipline, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailComparisonUp.

Chapter 2147

A Concrete Naming Certificate for $\mathsf{CauchyTailIntersection}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyTailIntersection}^{\uparrow}$ packet records the finite common tail-window intersection retained after two displayed regular-Cauchy windows have been compared and synchronized. It sits after the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the tail-comparison route of chapter 387, and the shared threshold surface of chapter 560. The packet names the retained intersection carrier itself: a finite row that can be handed to later $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers without choosing a quotient stream, selecting a limit, or invoking ambient completeness.

Definition 2147.1 (CauchyTailIntersection carrier)**.** *A* $\mathsf{CauchyTailIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, W_0, W_1, I, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular-Cauchy source rows whose finite tail windows are being compared.*
2. *W_0 and W_1 are the displayed* $\mathsf{StreamName}^{\uparrow}$ *tail-window rows retained from the two sources after the sibling comparison and meet packets have fixed the visible cuts.*
3. *I is the retained common finite tail-window intersection. It stores exactly the inspected cuts lying in both displayed tails, together with their local order and membership evidence.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the retained intersection. It records the dyadic precision requests and budget comparisons used by the common tail rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It reads the retained intersection through the regular rational sequence interface and exposes no independent stream or quotient coordinate.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row. It is visible only after the intersection, dyadic ledger, and regular readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the source rows, windows, retained intersection, dyadic ledger, regular readback, real handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$X, Y, W_0, W_1 \longmapsto I \longmapsto D \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailIntersection}^{\uparrow}$, $\mathsf{CauchyTailComparison}^{\uparrow}$, $\mathsf{RegularCauchyTailMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailIntersection}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these twelve rows. Its classifier compares accepted packets only through the two source rows, the two displayed windows, the retained intersection, the dyadic ledger, the regular readback, the real handoff, componentwise transport, continuation replay, provenance, and local naming. It has no coordinate for quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyTailIntersectionUp.CauchyTailIntersectionTasteGate single carrier alignment`

Theorem 2147.2 (CauchyTailIntersection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailIntersection}^{\uparrow}$ *carrier*

$$T = (X, Y, W_0, W_1, I, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailIntersection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two source rows, two displayed windows, retained intersection, dyadic ledger, regular readback, real handoff, transport, replay, provenance, and local naming row;*
2. *comparison admission, requiring W_0 and W_1 to be supplied by the sibling* $\mathsf{CauchyTailComparison}^{\uparrow}$ *and* $\mathsf{RegularCauchyTailMeet}^{\uparrow}$ *routes before I can be read;*
3. *intersection exactness, requiring I to retain only cuts displayed in both windows and to retain every displayed cut certified by both windows inside the finite packet;*
4. *tolerance compatibility, requiring D to read only dyadic tolerance requests attached to the retained intersection;*
5. *readback exactness, requiring R to be the only* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the retained intersection and E to be the only* $\mathsf{Real}^{\uparrow}$ *handoff downstream of R;*
6. *structural exhaustion, requiring H, C, P, N to preserve, replay, source, and name the listed rows without exporting quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2147.1. The accepted packet displays exactly the two source rows, two tail windows, retained intersection, dyadic ledger, regular readback, real handoff, and structural rows.

Comparison admission is read from the sibling routes named in the provenance row P. The comparison and meet packets expose the two compatible finite tail windows before the local intersection row I can be inspected.

Intersection exactness is a projection of I. Since I is a finite row, membership is checked only against the displayed cuts in W_0 and W_1, and every retained cut carries both displayed membership entries.

Tolerance compatibility is read from D. The dyadic ledger is downstream of I, so every tolerance request is attached to an already retained common cut rather than to a hidden tail oracle.

Readback exactness is read from R and E. The regular readback consumes I and D, and the real handoff is visible only after that readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Because the carrier has no quotient, selected-limit, choice, host-equality, trichotomy, or completeness coordinate, the six listed obligation rows exhaust the local certificate surface. □

Theorem 2147.3 (CauchyTailIntersection window meet)**.** *Let* $T = (X, Y, W_0, W_1, I, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyTailIntersection}^{\uparrow}$ *carrier. Every consumer read of a retained common tail cut factors through the displayed route*

$$X, Y \longmapsto W_0, W_1 \longmapsto I.$$

The row I *is therefore the unique retained finite intersection surface available to the packet, and no consumer can replace it by an independent tail, quotient stream, selected limit, host equality proof, trichotomy branch, or ambient completeness principle.*

Proof. Start with a consumer read of a retained common tail cut. Projection of the accepted carrier supplies the source rows X, Y and the displayed windows W_0, W_1 before the intersection row is available.

The sibling comparison and meet routes inside P account for the two displayed windows. Their role is to expose finite compatible tail cuts, not to export a completed tail object.

Now inspect I. A cut is readable from I only with membership evidence from both W_0 and W_1, so the consumer cannot bypass either displayed window.

The transport and replay rows H and C preserve the same route through hsame and Cont. The local naming row N names that retained intersection and no second intersection coordinate.

Thus every retained common cut factors through X, Y, W_0, W_1, I, and the packet exports no independent tail, quotient stream, selected limit, host equality, trichotomy branch, or ambient completeness principle. □

Theorem 2147.4 (CauchyTailIntersection RegSeqRat handoff)**.** *Let* $T = (X, Y, W_0, W_1, I, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyTailIntersection}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the retained common tail-window intersection factors through*

$$I \longmapsto D \longmapsto R \longmapsto E$$

after the source-window route X, Y, W_0, W_1 *has been displayed. The handoff exports the finite retained intersection, dyadic tolerance ledger, regular rational readback, and real-name boundary, and it exports no quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.*

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer request. The source-window route has already displayed X, Y, W_0, W_1 and the retained common intersection I.

The dyadic ledger D is the next visible row. It reads tolerance requests only from the retained intersection, so it cannot introduce a separate budget schedule or unbounded tail selector.

The regular readback R consumes I through D. Since R is the only $\mathsf{RegSeqRat}^{\uparrow}$ row in the carrier, the consumer has no alternate regular-rational channel.

The real handoff E is downstream of R. It seals the displayed finite route for $\mathsf{Real}^{\uparrow}$ consumers but does not assert a host real equality or completeness theorem.

Componentwise transport, continuation replay, package provenance, and local naming preserve this same chain. Therefore every regular or real-facing read factors through I, D, R, E and exports none of the excluded objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailIntersection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailIntersection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2147.1, Definition 2147.2, Definition 2147.3, and Definition 2147.4 seed the finite retained common tail-window intersection carrier for regular-Cauchy and real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailIntersection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison admission, intersection exactness, tolerance compatibility, regular readback exactness, real handoff exactness, and structural exhaustion for BEDC.Derived.CauchyTailIntersection

Constructive story: A $\mathsf{CauchyTailIntersection}^{\uparrow}$ packet is generated from two regular-Cauchy source rows, two displayed finite tail windows, one retained common intersection row, a dyadic tolerance ledger, a regular rational readback, a real-name handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2148

A Concrete Naming Certificate for $\mathsf{YonedaCompletion}^{\uparrow}$

Following Lawvere's enriched metric spaces and the Isbell–Yoneda completion tradition, the $\mathsf{YonedaCompletion}^{\uparrow}$ packet records a presheaf-style completion route as finite BEDC NameCert data. It sits over the formal-ball surfaces of chapter 1510 and chapter 2012, the metric source of chapter 57, the Cauchy-net route of chapter 589, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic radius ledger of chapter 291, and the real-name seal of chapter 38. The packet gives completion consumers a finite Yoneda-style handoff without quotient completion, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2148.1 (Yoneda completion carrier)**.** *A* $\mathsf{YonedaCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (M, B, F, D, S, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, displayed distance observations, and finite comparison histories read by completion consumers.*
2. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *radius-tagged observation row over M.*
3. *F is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, exposing the rounded formal-ball completion route available before any Yoneda-facing read is exported.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording positive-radius witnesses, dyadic refinements, strict comparisons, and radius-sum budgets used by B and F.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for center observations, radius requests, and presheaf-profile probes.*
6. *Q is the* $\mathsf{CauchyNet}^{\uparrow}$ *comparison row. It records the directed-window route through which a finite Yoneda profile can hand off to ordinary Cauchy data.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the retained stream windows and dyadic budgets.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal, reachable only through the displayed D, S, R route and not as an ambient completeness principle.*
9. *H records componentwise* hsame *transport for metric sources, formal balls, formal-ball completion rows, dyadic ledgers, stream windows, Cauchy-net comparisons, regular readbacks, real seals, replay data, provenance, and local naming.*

10. *C records displayed* Cont *replay of the route*

$$M, B, F, D, S \longmapsto Q, R \longmapsto A.$$

11. *P is the* Pkg *provenance over* $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{YonedaCompletion}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these twelve rows. Its classifier compares accepted packets only through $M, B, F, D, S, Q, R, A, H, C, P, N$ and has no coordinate for quotient completion, selected limit representatives, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.YonedaCompletionUp.TasteGate.YonedaCompletionTasteGate single carrier alignment`

Theorem 2148.2 (Yoneda completion NameCert obligations)**.** *For an accepted* $\mathsf{YonedaCompletion}^{\uparrow}$ *carrier*

$$Y = (M, B, F, D, S, Q, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{YonedaCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source row, formal-ball row, formal-ball completion row, dyadic radius ledger, stream-window schedule, Cauchy-net comparison row, regular rational readback, real seal, transport, replay, provenance, and local naming row;*
2. *Yoneda-profile exposure, requiring every completion-facing profile to be read as a finite formal-ball family over M, B, F before any Cauchy-net or real-facing handoff is available;*
3. *radius and window exactness, requiring each formal-ball radius, refinement, and comparison used by the profile to be accounted for in D and inspected through S;*
4. *Cauchy handoff discipline, requiring Q to compare the finite profile with a displayed* $\mathsf{CauchyNet}^{\uparrow}$ *route rather than replacing it with a quotient of Cauchy names;*
5. *regular rational and real readback discipline, requiring R and A to be downstream of the same D, S, Q ledger;*
6. *classifier transport, requiring H and C to preserve the displayed route componentwise and to replay only the listed coordinates;*
7. *non-escape, excluding quotient completion, selected limits, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2148.1. The accepted packet exposes exactly the metric, formal-ball, formal-ball completion, dyadic, stream-window, Cauchy-net comparison, regular readback, real seal, and structural rows.

Yoneda-profile exposure is forced by the order of the displayed rows. A completion-facing profile first reads M, B, F as finite formal-ball data, so it cannot begin from a quotient completion object or a selected representative.

Radius and window exactness follows from D and S. Every radius comparison and every inspected center observation used by the profile has a dyadic budget and a finite stream-window address before the profile can be consumed.

The Cauchy handoff is the row Q. It compares the same finite profile with a displayed directed-window route, and therefore supplies no quotient of Cauchy names and no countable-choice selector.

The regular rational and real readbacks are downstream. The rows R and A can be read only after the dyadic, stream-window, and Cauchy-net comparison rows have been displayed.

Finally H, C, P, N transport, replay, source, and name only the listed rows. Since the excluded host principles have no coordinate in the carrier, they cannot appear as local NameCert obligations. □

Theorem 2148.3 (Yoneda completion formal-ball density)**.** *Let*

$$Y = (M, B, F, D, S, Q, R, A, H, C, P, N)$$

be an accepted $\mathsf{YonedaCompletion}^{\uparrow}$ *carrier. Every displayed Yoneda profile exported by* Y *is densely probed by finite* $\mathsf{FormalBall}^{\uparrow}$ *data: for each retained profile request there is a formal-ball row in* B*, a formal-ball completion read in* F*, a dyadic radius budget in* D*, and a stream-window address in* S *whose replay through* C *reaches the same profile request. Thus density is recorded as finite formal-ball coverage, not as a selected limit and not as an ambient completion theorem.*

Proof. Fix a retained profile request. By carrier admission the request is part of the displayed $\mathsf{YonedaCompletion}^{\uparrow}$ packet, so its only profile-facing coordinates are M, B, F, D, S before any downstream readback occurs.

The formal-ball row B supplies the radius-tagged probe over the metric source M. The formal-ball completion row F records that the same probe participates in the rounded completion route.

The rows D and S bind the probe to finite data. The radius is accounted for by the dyadic ledger D, and the inspected center observation has a stream-window address in S.

Replay through C identifies this finite formal-ball coverage with the retained profile request. The real-facing rows R, A and the Cauchy-net comparison row Q are downstream of the same coverage route, so they cannot supply a selected limit or ambient completeness principle. □

Theorem 2148.4 (Yoneda completion regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{YonedaCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$M, B, F, D, S \longmapsto Q \longmapsto R \longmapsto A.$$

Consequently the packet hands finite Yoneda-profile data to regular Cauchy consumers only through the displayed $\mathsf{FormalBall}^{\uparrow}$*,* $\mathsf{FormalBallCompletion}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyNet}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. It does not export quotient completion, selected limits, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier and project its readback rows. A $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer can reach R or A only by following the replay route named in Definition 2148.1.

The first stage of that route is formal-ball data over the metric source. The rows M, B, F expose the metric observations, formal-ball probes, and formal-ball completion reads.

The second stage binds those probes to finite budgets. The dyadic ledger D and stream schedule S record the radius comparisons and inspected windows used by the profile.

The Cauchy comparison row Q then supplies the ordinary directed-window handoff. Since Q is downstream of M, B, F, D, S, it cannot replace the finite profile by a quotient Cauchy object or a selected limit.

Finally R and A are read from that same route. Componentwise transport and replay through H and C, together with provenance and local naming through P, N, preserve the factorization and expose no additional completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{YonedaCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{YonedaCompletion}^{\uparrow}$ packet is generated from a displayed MetricUp source, FormalBallUp probes, FormalBallCompletionUp rounded reads, DyadicRatCore radius ledgers, StreamName windows, CauchyNet comparisons, RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YonedaCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2148.1, Definition 2148.2, Definition 2148.3, and Definition 2148.4 seed the finite Yoneda completion route over formal-ball profiles, directed Cauchy comparison, regular rational readback, real sealing, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YonedaCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Yoneda-profile exposure, radius and window exactness, Cauchy handoff discipline, regular rational and real readback discipline, classifier transport, formal-ball density, and non-escape for BEDC.Derived.YonedaCompletionUp.

Chapter 2149

A Concrete Naming Certificate for $\mathsf{MetricCompletionReflector}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop–Bridges on metric and uniform completion, the $\mathsf{MetricCompletionReflector}^{\uparrow}$ packet records the metric-specialized reflector surface for the open L10 completion route. It sits after the metric-completion carrier of chapter 1506, the uniform-completion route of chapter 1473, the separated metric boundary of chapter 1411, and the broader reflector surface of chapter 1585. The packet exposes completion reflection through finite metric, uniform, separated, Cauchy-reflector, real-seal, transport, replay, provenance, and local NameCert rows without importing quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Definition 2149.1 (MetricCompletionReflector carrier)**.** *A* $\mathsf{MetricCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, U, S, K, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row. It records the displayed metric-completion source, Cauchy evidence, completion-facing reads, and local object boundary inspected by the reflector.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *comparison row. It records the uniform completion route that agrees with the metric completion on the displayed entourages and Cauchy data.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row. It supplies the zero-distance uniqueness boundary used before any completed read may be treated as reflector-stable.*
4. *K is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row. It records the unit, counit, idempotent replay, universal-property, and reflection surface from which the metric-specialized reflector is read.*
5. *A is the real-seal handoff row. It records the terminal* $\mathsf{Real}^{\uparrow}$*-facing windows reached by metric-completion consumers only after M, U, S, K have been displayed.*
6. *H records componentwise* hsame *transport for metric completion, uniform completion, separated metric evidence, the Cauchy reflector row, the real-seal handoff, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$M, U, S, K \longmapsto A \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionReflector}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionReflector}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $M, U, S, K, A, H, C, P, N$ and has no coordinate for quotient completion, selected representatives, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.MetricCompletionReflectorUp.MetricCompletionReflectorTasteGate single carrier alignment`

Theorem 2149.2 (MetricCompletionReflector NameCert obligations)**.** *For an accepted* $\mathsf{MetricCompletionReflector}^{\uparrow}$ *carrier*

$$R = (M, U, S, K, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric-completion row, uniform-completion comparison row, separated metric boundary, Cauchy reflector row, real-seal handoff, transport, replay, provenance, and local naming row;*
2. *metric-reflector source exactness, requiring every reflector-facing read to project M before the uniform comparison row or Cauchy reflector row can be consumed;*
3. *uniform compatibility, requiring U to compare only the displayed metric entourages and Cauchy data already available through M;*
4. *separated uniqueness, requiring S before any completed read is reused by the reflector as a stable metric-completion object;*
5. *reflector route compatibility, requiring K to supply the unit, counit, idempotent replay, universal-property, and reflection reads over the same M, U, S, A surface;*
6. *classifier transport, requiring* hsame *comparison to preserve the metric, uniform, separated, Cauchy-reflector, real-seal, replay, provenance, and local naming rows componentwise;*
7. *non-escape, excluding quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, and bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2149.1. The accepted packet exposes exactly the metric-completion, uniform-completion, separated metric, Cauchy-reflector, real-seal, transport, replay, provenance, and local naming rows.

Metric-reflector source exactness starts at M. A consumer cannot inspect the uniform comparison row U or the Cauchy reflector row K until the metric-completion source and completion-facing route in M have been displayed.

Uniform compatibility is confined to U. It compares the displayed metric entourages and Cauchy data already named by M, so it cannot introduce a separate host uniform completion or a selected completed point.

Separated uniqueness is supplied by S. The reflector may reuse a completed read only after the separated metric boundary has classified the zero-distance evidence visible in the same packet.

Finally K, A, H, C, P, N preserve the same finite route. The Cauchy reflector row supplies unit, counit, idempotent, universal-property, and reflection faces over M, U, S, A, while the structural

rows transport, replay, source, and name only displayed coordinates. The excluded host principles have no carrier row, so the listed obligations exhaust the local NameCert surface. □

Theorem 2149.3 (MetricCompletionReflector real-seal handoff)**.** *Let*

$$R = (M, U, S, K, A, H, C, P, N)$$

be an accepted $\mathsf{MetricCompletionReflector}^{\uparrow}$ *carrier. Every real-facing reflector consumer reaches the terminal seal row* A *only through the finite route*

$$M \longmapsto U, S \longmapsto K \longmapsto A \longmapsto H, C, P, N.$$

The handoff specializes the broader $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *surface to the displayed metric-completion and separated-metric rows. It does not export quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with an accepted carrier R. The metric-completion row M is the only completion-object entry point, so every real-facing reflector consumer first reads the displayed metric-completion route.

The uniform and separated rows are then read as constraints on the same completion data. The row U supplies the uniform-completion comparison over the displayed entourages, and S supplies the separated metric boundary before reuse of a completed read.

The Cauchy reflector row K consumes those rows as its metric-specialized input. Its unit, counit, idempotent replay, universal-property, and reflection faces are therefore attached to the same M, U, S surface rather than to a detached quotient object.

The real-seal row A is downstream of this reflector route. Componentwise transport by H, displayed replay by C, provenance by P, and local naming by N preserve that order, so no excluded host-side principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionReflector}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionReflector}^{\uparrow}$ packet is generated from a MetricCompletionUp row, UniformCompletionUp comparison row, SeparatedMetricUp boundary, CauchyCompletionReflectorUp row, RealUp-facing seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2149.1, Definition 2149.2, and Definition 2149.3 seed the metric-specialized reflector surface for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MetricCompletionReflectorUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-reflector source exactness, uniform compatibility, separated uniqueness, reflector route compatibility, real-seal handoff, classifier transport, and non-escape for BEDC.Derived.MetricCompletionReflectorUp.

Chapter 2150

A Concrete Naming Certificate for $\mathsf{VietorisRipsComplex}^{\uparrow}$

Following Vietoris and Rips on finite threshold complexes, the $\mathsf{VietorisRipsComplex}^{\uparrow}$ packet records the BEDC surface that turns finite metric data into a simplex ledger. It sits over the metric distance interface of chapter 57, the finite net exposure of chapter 1767, the rational radius ledgers of chapter 291 and chapter 37, and the finite face consumer of chapter 238. The packet gives compact-metric approximation and persistent-homology consumers a finite threshold complex without importing host topology, open covers, selected dense subsets, quotient realization, or ambient homology.

Definition 2150.1 (Vietoris–Rips complex carrier)**.** *A* $\mathsf{VietorisRipsComplex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, V, \epsilon, D, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row. It records only the finite point histories and displayed distance witnesses available to the metric consumer.*
2. *V is the finite vertex or net ledger, read through the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *surface when the source data come from an approximation row.*
3. *ϵ is the threshold row. It is displayed through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *rational-radius ledgers before any simplex test can compare distances with the threshold.*
4. *D records pairwise distance witnesses for the displayed vertices. Each row names two vertices from V, the metric-distance witness read from X, and the comparison of that witness with ϵ.*
5. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *handoff row. It records exactly the finite simplex and face rows whose simplex predicate is accepted by threshold tests in D.*
6. *H records componentwise* hsame *transport for the metric source, vertex ledger, threshold row, pairwise distance ledger, simplicial handoff, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay from a simplex request through V, ϵ, D before the request reaches S.*
8. *P is the* Pkg *provenance over* $\mathsf{VietorisRipsComplex}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{VietorisRipsComplex}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $X, V, \epsilon, D, S, H, C, P, N$ and has no coordinate for host topology, open-cover compactness, selected dense subsets, quotient geometric realization, ambient homology, Lean proof search, or bridge checking.

Lean4 ✓ `BEDC.Derived.VietorisRipsComplexUp.VietorisRipsComplexUpTasteGate single carrier alignment`

Theorem 2150.2 (Vietoris–Rips complex NameCert obligations)**.** *For an accepted* $\mathsf{VietorisRipsComplex}^{\uparrow}$ *carrier*

$$R = (X, V, \epsilon, D, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VietorisRipsComplex}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source row, finite vertex or net ledger, threshold row, pairwise distance ledger, simplicial handoff row, transport, replay, provenance, and local naming;*
2. *threshold simplex exactness, requiring a displayed simplex of S to be accepted only when every displayed pair of its vertices has a row in D whose metric-distance witness is bounded by ϵ;*
3. *finite-net exposure, requiring vertex reads imported from* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *to remain inside the ledger V before any distance row or simplex row can use them;*
4. $\mathsf{SimplicialComplex}^{\uparrow}$ *compatibility, requiring the simplex and face rows handed to S to be closed under displayed face restriction because every face removes pairwise threshold requests rather than creating new ones;*
5. *classifier transport, requiring* hsame *comparison to preserve the metric source, vertex ledger, threshold row, pairwise distance ledger, simplicial handoff, replay, provenance, and local naming rows componentwise;*
6. *non-escape, excluding host topology, open covers, selected dense subsets, quotient geometric realization, ambient homology, and any simplex not routed through the finite threshold ledger.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2150.1. The accepted packet exposes exactly the metric source, vertex ledger, threshold row, pairwise distance ledger, simplicial handoff, and structural rows H, C, P, N.

Threshold simplex exactness is the route V, ϵ, D, S. A simplex read first lists its vertices in V, then reads the rational threshold ϵ, then checks every displayed pair through the corresponding metric-distance row in D before the simplex row is handed to S.

Finite-net exposure is inherited from V. When the vertices are supplied by a $\mathsf{FiniteEpsilonNet}^{\uparrow}$ packet, the Rips carrier reads them as ledger entries rather than as a selected dense subset or host-level finite set.

Compatibility with $\mathsf{SimplicialComplex}^{\uparrow}$ follows because every face request removes vertices from an already accepted simplex. The remaining pairwise requests are a subledger of the same D rows, so the face row can be handed to S without creating a new metric or threshold coordinate.

Finally H, C, P, N preserve and name only the displayed coordinates. Since topology, open covers, quotient realization, ambient homology, and unrouted simplex predicates have no carrier row, the listed obligations exhaust the local NameCert surface. □

Theorem 2150.3 (Vietoris–Rips simplex threshold handoff)**.** *Let*

$$R = (X, V, \epsilon, D, S, H, C, P, N)$$

be an accepted $\mathsf{VietorisRipsComplex}^{\uparrow}$ *carrier. Every simplex read exported to the* $\mathsf{SimplicialComplex}^{\uparrow}$ *consumer factors through the finite route*

$$V \longmapsto \epsilon \longmapsto D \longmapsto S \longmapsto H, C, P, N.$$

Consequently the simplicial consumer can use only the displayed finite metric threshold rows. It cannot use host topology, open-cover compactness, selected dense subsets, quotient geometric realization, ambient homology, or a simplex predicate detached from D*.*

Proof. Start with a simplex read. Projection gives the vertex ledger V, but the packet exposes no host finite set, no selected dense subset, and no geometric realization at this stage.

The read must next pass through the threshold row ϵ and the pairwise distance ledger D. For each displayed pair of vertices, D supplies the metric-distance witness from X and the finite comparison with ϵ.

Only after those rows have been read is the simplex handed to S. Face reads are handled by the same route because restricting a simplex can only delete pairwise threshold requests already present in D.

Transport, replay, provenance, and local naming are exactly H, C, P, N. They preserve the displayed route and add no topology, cover, representative choice, realization quotient, homology group, or detached simplex classifier. Hence every simplex read factors through the stated finite threshold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{VietorisRipsComplex}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VietorisRipsComplex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2150.1, Definition 2150.2, and Definition 2150.3 seed the finite metric threshold complex packet over displayed metric, net, rational threshold, pairwise distance, simplicial handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VietorisRipsComplex}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.VietorisRipsComplexUp`

Bridge status: none

Not claimed: This chapter does not close persistent homology, Cech complexes, open-cover compactness, quotient geometric realization, selected dense subsets, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold simplex exactness, finite-net exposure, SimplicialComplex compatibility, classifier transport, simplex handoff factorization, and non-escape for BEDC.Derived.VietorisRipsComplexUp.

Constructive story: A $\mathsf{VietorisRipsComplex}^{\uparrow}$ packet is generated from a MetricUp source row, a finite vertex or FiniteEpsilonNet ledger, a DyadicRatCore or Rat threshold row, pairwise distance witnesses, a SimplicialComplex handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2151

A Concrete Naming Certificate for $\mathsf{RegularCauchyTranslation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTranslation}^{\uparrow}$ packet records constant translation of a regular Cauchy real name as displayed finite BEDC data. It sits beside the two-source addition surface of chapter 417, the sign-flip route of chapter 459, the rational scaling route of chapter 465, the dyadic arithmetic core of chapter 291, the finite observation discipline of chapter 126, the regular rational source boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers one finite route for reading $x + c$ through the displayed source and constant rows, without importing host real addition, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2151.1 (Regular Cauchy translation carrier)**.** *A* $\mathsf{RegularCauchyTranslation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T_c = (X, Q, W_X, W_Q, D_X, D_Q, A, E, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row for the regular Cauchy real name being translated.*
2. *Q is the constant row, displayed either as a dyadic endpoint row or as a rational row whose dyadic readings are supplied through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *W_X and W_Q are synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows. The row W_X opens the requested source observation, while W_Q opens the stationary finite window for the constant row at the same precision request.*
4. *D_X and D_Q are the translated dyadic endpoint rows, recording the source endpoint read from X and the constant endpoint read from Q in the common dyadic ledger.*
5. *A is the* $\mathsf{RegularCauchySum}^{\uparrow}$ *handoff row adding the source regular name to the constant regular row generated from Q.*
6. *E is the finite error ledger combining source tolerance, constant display tolerance, dyadic endpoint addition tolerance, and the output tolerance consumed by the sum handoff.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the displayed sum route.*
8. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $X, Q, W_X, W_Q, D_X, D_Q, A, E, R$ have been displayed.*

9. *H records componentwise* hsame *transport for the source, constant, synchronized windows, dyadic endpoint rows, sum handoff, finite ledger, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from any translation-facing precision request through $X, Q, W_X, W_Q, D_X, D_Q, A$, before the seal row Z is exposed.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTranslation}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyNegation}^{\uparrow}$, $\mathsf{RegularCauchyScale}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTranslation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Q, W_X, W_Q, D_X, D_Q, A, E, R, Z, H, C, P, N$. The carrier has no coordinate for host real addition, quotient stream equality, a selected representative, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTranslationUp.RegularCauchyTranslationTasteGate single carrier alignment`

Theorem 2151.2 (Regular Cauchy translation NameCert obligations)**.** *For every accepted* RegularCauchyTranslatio *carrier*

$$T_c = (X, Q, W_X, W_Q, D_X, D_Q, A, E, R, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTranslation}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *source-and-constant admission, requiring the source* $\mathsf{RegSeqRat}^{\uparrow}$ *row X and the displayed constant row Q to be present before any translated read is exposed;*

2. *synchronized-window exposure, requiring W_X and W_Q to open the source and constant observations at the same finite precision request;*

3. *dyadic endpoint and ledger exactness, requiring D_X, D_Q, E to account for the source endpoint, constant endpoint, endpoint addition, and output tolerance used by the translation;*

4. *sum-route handoff, requiring the translated readback R to pass through the* $\mathsf{RegularCauchySum}^{\uparrow}$ *row A and through no host addition operation;*

5. *structural naming, requiring Z, H, C, P, N to expose the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, package provenance, and local* NameCert *row for exactly this finite translation packet.*

No translation obligation is stored outside these five rows, and the surface refuses host real addition, quotient stream equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local data.*

Proof. Project the accepted carrier. The source row X, constant row Q, synchronized windows W_X, W_Q, dyadic endpoint rows D_X, D_Q, sum handoff A, finite ledger E, readback row R, seal row Z, and structural rows H, C, P, N are the whole visible packet.

Read the source and constant windows first. The row W_X supplies the finite observation of the source regular name, while W_Q supplies the stationary constant observation at the same precision request. Thus the source endpoint and constant endpoint are compared only after both windows are displayed.

Next read the dyadic and budget rows. The rows D_X and D_Q supply the endpoint data consumed by the $\mathsf{RegularCauchySum}^{\uparrow}$ handoff, and E records the source tolerance, constant tolerance, endpoint-addition tolerance, and output tolerance before R is available.

Finally replay the route through the structural rows. Since A is the only addition handoff and every seal-facing read must pass through R before Z, no quotient stream, selected representative, host addition operation, countable-choice selector, trichotomy witness, ambient completeness theorem, Lean verification, or bridge credential can be read from the local packet. □

Theorem 2151.3 (Regular Cauchy translation real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RegularCauchyTranslation}^{\uparrow}$ *carrier factors through the displayed constant row and sum route*

$$(X, Q) \longmapsto (W_X, W_Q) \longmapsto (D_X, D_Q) \longmapsto A \longmapsto E \longmapsto R \longmapsto Z.$$

Thus the consumer sees the source regular row, the constant dyadic or rational row, synchronized StreamName windows, translated dyadic endpoint rows, the $\mathsf{RegularCauchySum}^{\uparrow}$ *handoff, the finite error ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It receives no host real addition, quotient stream equality, selected representative, countable-choice witness, trichotomy decision, ambient completeness coordinate, Lean verification, or bridge checking.*

Proof. Begin with a Real-facing read of the accepted packet. The carrier exposes the seal row Z only downstream of the readback row R, so the read must first pass through the regular readback route.

The readback route is fixed by the constant row and the sum handoff. The source row X and constant row Q are opened through synchronized windows W_X, W_Q, their dyadic endpoint rows are read as D_X, D_Q, and the displayed $\mathsf{RegularCauchySum}^{\uparrow}$ row A is then the only route forming the translated regular name.

The finite ledger precedes the seal. The row E accounts for source, constant, endpoint-addition, and output tolerances before R and then Z can be consumed.

The structural rows preserve this factorization. Componentwise hsame transport, Cont replay, Pkg provenance, and local naming do not add another Real-facing coordinate, so every accepted seal read factors through the displayed constant row and sum route and cannot escape to the refused principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTranslation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTranslation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2151.1, Definition 2151.2, and Definition 2151.3 seed the finite constant-translation surface over one $\mathsf{RegSeqRat}^{\uparrow}$ source row, a displayed dyadic or rational constant row, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, a $\mathsf{RegularCauchySum}^{\uparrow}$ handoff, finite error ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTranslation}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyTranslationUp.RegularCauchyTranslationTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close host real addition, quotient stream equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-and-constant admission, synchronized-window exposure, dyadic endpoint and ledger exactness, sum-route handoff, Real seal non-escape, componentwise transport, continuation replay, provenance, and local naming for BEDC.Derived.RegularCauchyTranslationUp.

Constructive story: A $\mathsf{RegularCauchyTranslation}^{\uparrow}$ packet is generated from a RegSeqRat source row, a displayed dyadic or rational constant row, synchronized StreamName windows, translated DyadicRatCore endpoint rows, a RegularCauchySum handoff, a finite error ledger, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2152

A Concrete Naming Certificate for $\mathsf{RoundedIntervalDomain}^{\uparrow}$

Following Scott domain theory and Bishop–Bridges interval approximation of constructive reals, the $\mathsf{RoundedIntervalDomain}^{\uparrow}$ packet records the finite rounded dyadic interval-domain route by which regular-Cauchy real names are approximated before the terminal real-name seal is exposed. It refines the plain interval-domain carrier of chapter 1543 by forcing every endpoint, containment, and refinement read through rounded dyadic data from chapter 291, finite stream windows from chapter 126, regular rational readback from chapter 301, and the displayed real-name boundary of chapter 38. The packet gives Real and Completion consumers a finite rounded approximation surface; it is not a Dedekind-cut construction, an open-cover compactness principle, trichotomy, a quotient representative selector, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2152.1 (Rounded interval-domain carrier)**.** *A* $\mathsf{RoundedIntervalDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, B, J, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the rounded lower endpoint row. It records a dyadic rational endpoint, its displayed rounding radius, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison data used to read it.*
2. *U is the rounded upper endpoint row. It records the paired dyadic rational endpoint, its displayed rounding radius, and the same dyadic arithmetic surface.*
3. *B is the rounded containment row. It records lower-below-upper evidence, slack inequalities, and the fact that the exact real-name request lies inside the displayed rounded dyadic interval.*
4. *J is the refinement witness row. It records a finite refinement step from a parent rounded interval to a child rounded interval whose endpoints and radii are again displayed by L, U, B-shaped data.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation window. It records the finite indices at which endpoint and containment observations are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational values read from the same observation window after the dyadic endpoint and containment rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is reachable only downstream of L, U, B, J, W, R.*
8. *H records componentwise* hsame *transport for endpoints, containment, refinement, stream windows, regular readback, Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the finite route*

$$L, U \longmapsto B \longmapsto J \longmapsto W \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RoundedIntervalDomain}^{\uparrow}$, $\mathsf{IntervalDomain}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RoundedIntervalDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, B, J, W, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, open-cover compactness, trichotomy, quotient representatives, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RoundedIntervalDomainUp.RoundedIntervalDomainUp`
Lean4 ✓ `BEDC.Derived.RoundedIntervalDomainUp.RoundedIntervalDomainTasteGate single carrier alignment`

Theorem 2152.2 (Rounded interval-domain NameCert obligations)**.** *For every accepted* $\mathsf{RoundedIntervalDomain}^{\uparrow}$ *carrier*

$$I = (L, U, B, J, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RoundedIntervalDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed rounded lower endpoint, rounded upper endpoint, containment, refinement, stream window, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *rounded-refinement stability, requiring every child interval named by J to keep dyadic endpoints, rounding radii, containment evidence, and observation windows inside the displayed packet route;*
3. hsame *transport, requiring H to preserve endpoints, radii, containment checks, refinement witnesses, stream windows, regular readbacks, Real seals, replay rows, provenance, and local naming componentwise;*
4. *containment classifier reflexivity on a displayed interval, symmetry for equality of two displayed rounded presentations with the same endpoint and radius rows, and transitivity along a finite chain of displayed refinement witnesses;*
5. *ledger non-escape, requiring C and P to replay and source only the rows $L, U, B, J, W, R, E, H, C, P, N$;*
6. *Real seal compatibility, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer of E to read the rounded dyadic endpoint ledger, containment row, refinement witness, stream window, and regular rational readback before the seal is inspected.*

No rounded interval-domain obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2152.1. The projection exposes exactly the rounded endpoints, containment, refinement witness, observation window, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Rounded-refinement stability reads L, U, B, J, W. A child interval in J is generated from displayed parent endpoint rows and retains displayed rounding radii and containment evidence before any later stream or regular rational consumer is available.

Transport is componentwise. The row H moves each displayed endpoint, radius, containment check, refinement witness, window, readback, seal, replay row, provenance row, and naming row to its transported counterpart, so no detached analytic coordinate appears during transport.

The containment classifier clauses use only the rounded presentation. Reflexivity is the displayed interval compared with itself, symmetry is available when two presentations expose the same rounded endpoint and radius rows, and transitivity is concatenation of the finite refinement witnesses recorded by J.

Finally, ledger non-escape and Real seal compatibility follow from the replay order in C and the source row P. The terminal seal E is downstream of dyadic endpoint stability, containment, refinement, finite observation, and regular readback, and the carrier has no row for any excluded principle. □

Theorem 2152.3 (Rounded interval-domain Cauchy readback)**.** *Let*

$$I_0, I_1, I_2, \ldots$$

be a nested family of accepted $\mathsf{RoundedIntervalDomain}^{\uparrow}$ *packets whose refinement rows display shrinking dyadic radii and whose observation windows are compatible under* hsame*. Then the public* $\mathsf{Real}^{\uparrow}$ *seal read from the nested rounded dyadic interval route is the same seal exposed by its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route. The identification uses only* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint stability,* $\mathsf{StreamName}^{\uparrow}$ *pointwise observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, and* $\mathsf{Real}^{\uparrow}$ *classifier transport.*

Proof. First read the dyadic endpoint data. Each packet exposes rounded lower and upper endpoints together with radii, and the refinement row identifies the later endpoint rows as contained rounded refinements of the earlier rows. Shrinking radii make the endpoint ledger stable for every displayed precision request.

Next read the stream windows. Compatibility under hsame means that a fixed finite observation request is eventually served by the same displayed window row, so pointwise $\mathsf{StreamName}^{\uparrow}$ observation is independent of the chosen later packet.

The regular rational readback is then diagonal. For each displayed precision request, choose a packet far enough along the nested route that its radius and window rows cover the request, and read the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ row. The previous two steps ensure that this readback is stable under further displayed refinements.

Finally compare the Real seals. The seal row in each packet is downstream of the same dyadic endpoint, stream-window, and regular-readback route, and $\mathsf{Real}^{\uparrow}$ classifier transport applies along the displayed hsame rows. Hence the nested rounded interval route and the diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback expose the same public $\mathsf{Real}^{\uparrow}$ seal without using quotient representatives, trichotomy, open-cover compactness, Dedekind-cut completeness, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RoundedIntervalDomain}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoundedIntervalDomain}^{\uparrow}}, \mathsf{seed})$

Constructive story: A $\mathsf{RoundedIntervalDomain}^{\uparrow}$ packet is generated from rounded lower and upper dyadic endpoint rows, rounded containment evidence, a refinement witness, StreamName observation windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Scope closed: Definition 2152.1, Definition 2152.2, and Definition 2152.3 seed the finite rounded dyadic interval-domain route from endpoint rounding through refinement, finite observation, regular rational readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoundedIntervalDomain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut completeness, open-cover compactness, trichotomy, quotient representatives, selected limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rounded-refinement stability, hsame transport, containment classifier laws for displayed rounded presentations, ledger non-escape, Real seal compatibility, and Cauchy readback for BEDC.Derived.RoundedIntervalDomainUp.

Chapter 2153

A Concrete Naming Certificate for $\mathsf{CantorCylinderBasis}^{\uparrow}$

Following Brouwer's finite-prefix reading of choice sequences and the Bishop–Bridges cylinder terminology for Cantor space, the $\mathsf{CantorCylinderBasis}^{\uparrow}$ packet records the basic clopen cylinder surface of $\mathsf{CantorSpace}^{\uparrow}$ as finite NameCert data. It sits after the Cantor-space carrier of chapter 1475, the unary finite-window interface of chapter 126, the two-endpoint discipline of chapter 32, and the ordered prefix spine of chapter 36. The packet names prefix cylinders by displayed finite routes; it is not a host function extensionality principle, a quotient of infinite streams, a completed infinite product, a fan-theorem compactness argument, a countable-choice selector, or an appeal to $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2153.1 (Cantor cylinder basis carrier)**.** *A* $\mathsf{CantorCylinderBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (C, S, W, B, L, R, H, K, P, N)$$

with the following visible rows:

1. *C is the source* $\mathsf{CantorSpace}^{\uparrow}$ *packet whose public observations are read only through finite prefix windows;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the requested finite prefix positions;*
3. *W is the finite cylinder-window row, including the displayed prefix and the shorter windows used by downstream refinement reads;*
4. *B is the* $\mathsf{Bool}^{\uparrow}$ *endpoint ledger assigning visible Boolean rows to the positions of W;*
5. *L is the* $\mathsf{List}^{\uparrow}$ *spine row ordering the prefix entries and recording their finite length;*
6. *R is the refinement ledger saying which shorter or longer displayed prefix rows are being compared inside the same cylinder family;*
7. *H is the componentwise* hsame *transport row for C, S, W, B, L, R;*
8. *K is the displayed* Cont *replay row for cylinder reads and prefix refinements already accepted by the packet;*
9. *P is the* Pkg *provenance over* $\mathsf{CantorCylinderBasis}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}$ *row.*

The carrier has no coordinate for host-level maps $\mathsf{Nat}^{\uparrow} \to \mathsf{Bool}^{\uparrow}$*, quotient stream equality, completed infinite products, fan-theorem compactness, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorCylinderBasisUp.CantorCylinderBasisTasteGate single carrier alignment`

Definition 2153.2 (Cantor cylinder basis classifier)**.** *Two accepted* $\mathsf{CantorCylinderBasis}^{\uparrow}$ *carriers*

$$Z = (C, S, W, B, L, R, H, K, P, N) \quad \textit{and} \quad Z' = (C', S', W', B', L', R', H', K', P', N')$$

are classified as the same basis element exactly when their displayed $\mathsf{StreamName}^{\uparrow}$ *schedules select the same finite prefix length and their* $\mathsf{Bool}^{\uparrow}$ *rows agree componentwise along the ordered* $\mathsf{List}^{\uparrow}$ *spine. The comparison is transported only through* H, H' *and replayed only through* K, K'*. It does not compare host functions, quotient streams, infinite tails, or completed product coordinates.*

Theorem 2153.3 (Cantor cylinder basis NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}$ *surface of Definition 2153.1 consists exactly of the finite carrier admission, classifier transport, prefix refinement, StreamName–Bool scope, continuation replay, provenance, and non-escape obligations displayed by*

$$C, S, W, B, L, R, H, K, P, N.$$

Every obligation is checked by finite prefix data over the sibling $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*, and* $\mathsf{List}^{\uparrow}$ *packets.*

Proof. Carrier admission is projection of the ten displayed rows in Definition 2153.1. The source Cantor packet supplies the finite observation context, while S, W, B, L expose the requested prefix as StreamName windows, Boolean endpoints, and a List spine.

Classifier transport uses Definition 2153.2. Once the finite prefix length and componentwise Boolean rows are fixed, H transports the same finite rows and K replays them without adding any host-level stream coordinate.

The remaining obligations are inspection of R, H, K, P, N. Prefix refinement is recorded as a finite relation between displayed prefix rows, continuation replay is confined to the accepted rows, and provenance keeps the route inside the listed sibling chapters.

Non-escape follows because no carrier coordinate supplies host function extensionality, quotient stream equality, completed infinite products, fan-theorem compactness, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2153.4 (Cantor cylinder basis prefix refinement)**.** *Let* $Z = (C, S, W, B, L, R, H, K, P, N)$ *be an accepted* $\mathsf{CantorCylinderBasis}^{\uparrow}$ *carrier. If a displayed prefix row in* R *extends another displayed prefix row in the same carrier, then the longer cylinder read factors through the shorter cylinder read together with the additional finite* $\mathsf{Bool}^{\uparrow}$ *rows on the suffix. If the rows are incompatible at a visible position, the classifier rejects the shared basis element at that same finite position.*

Proof. Read the two prefix rows from R. The shorter row and the longer row are both subwindows of the displayed cylinder-window row W, so their positions are already scheduled by the same $\mathsf{StreamName}^{\uparrow}$ row S.

In the extension case, the common part is ordered by the same initial segment of the $\mathsf{List}^{\uparrow}$ spine L. The suffix is a finite list of additional $\mathsf{Bool}^{\uparrow}$ rows from B, and replay through K appends only those visible rows to the already accepted shorter read.

In the incompatible case, some displayed position carries different $\mathsf{Bool}^{\uparrow}$ endpoint rows. The classifier of Definition 2153.2 reads that position directly through the shared finite spine and rejects equality there, without comparing infinite tails or host functions. □

Theorem 2153.5 (Cantor cylinder basis StreamName-Bool scope)**.** *Every public read exported by an accepted* $\mathsf{CantorCylinderBasis}^{\uparrow}$ *carrier reaches downstream countable-name consumers only through the finite route*

$$\mathsf{CantorSpace}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{List}^{\uparrow},\ \mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}.$$

The route exposes prefix-cylinder membership, refinement, and incompatibility only as displayed BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *data. It does not export host function extensionality, quotient streams, completed infinite products, fan-theorem compactness, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a public consumer read of Z. The Cantor source C can be read only through the finite prefix discipline of chapter 1475; the requested positions are then exposed by the $\mathsf{StreamName}^{\uparrow}$ schedule S and the cylinder window W.

The visible values are exactly the $\mathsf{Bool}^{\uparrow}$ rows in B, ordered by the $\mathsf{List}^{\uparrow}$ spine L. Refinement information is read from R, transported by H, and replayed by K before provenance and local naming are recorded in P, N.

Thus every public route remains inside the displayed finite rows. The excluded objects are not supplied by the carrier, classifier, or refinement ledger, so no downstream consumer can obtain them from the cylinder-basis packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2153.1, Definition 2153.2, Definition 2153.3, Definition 2153.4, and Definition 2153.5 seed the finite prefix-cylinder basis surface over CantorSpace, StreamName, Bool, and List rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorCylinderBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close host function extensionality, quotient stream equality, completed infinite products, fan-theorem compactness, countable choice, Real completeness, or a standard bridge.

Upgrade path: The next step is obligationClosure: carrier admission, classifier transport, prefix refinement, StreamName-Bool routing, and non-escape must be discharged as local rows.

Constructive story: The construction starts from a finite CantorSpace prefix, reads the selected positions through StreamName windows, records Bool endpoints along a List spine, and exposes only hsame, Cont, Pkg, and local NameCert rows to downstream consumers.

Chapter 2154

A Concrete Naming Certificate for $\mathsf{SamuelCompactification}^{\uparrow}$

Following Samuel on uniform compactification and Bourbaki–Kelley on uniform spaces, the $\mathsf{SamuelCompactification}^{\uparrow}$ packet records the compactification-facing uniform route as finite BEDC data. It consumes the completion route of chapter 2131, the uniform completion surface of chapter 1473, the dense uniform embedding boundary of chapter 1651, the compact metric boundary of chapter 129, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the bounded-uniform-readout boundary without importing a host compactification space, a selected ultrafilter, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 2154.1 (Samuel compactification carrier)**.** *A* $\mathsf{SamuelCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, B, D, K, W, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform-entourage source row. It records the finite entourage requests and refinement witnesses from which every compactification-facing read begins.*
2. *B is the bounded uniformly continuous readout row. It records only displayed bounded readouts whose uniform continuity has been inspected through U.*
3. *D is the dense uniform embedding row. It records the source embedding, dense image windows, and pullback evidence required before a compact boundary read can be exposed.*
4. *K is the compact-metric boundary row. It records the compact metric recipient, boundary-window requests, and finite compactness witnesses consumed by compactification-facing observers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which dense-source observations, bounded readouts, and compact-boundary requests are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the displayed stream windows, bounded-readout budgets, and compact-boundary tolerances.*
7. *H records componentwise* hsame *transport for uniform entourages, bounded readouts, dense embeddings, compact-boundary rows, stream windows, regular readbacks, replay data, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$U, B, D, W \longmapsto R \longmapsto K.$$

9. *P is the* Pkg *provenance over* $\mathsf{SamuelCompactification}^{\uparrow}$, $\mathsf{SamuelCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SamuelCompactification}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $U, B, D, K, W, R, H, C, P, N$ and has no coordinate for a host compactification space, selected ultrafilters, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SamuelCompactificationUp.SamuelCompactificationTasteGate single carrier alignment`

Theorem 2154.2 (Samuel compactification NameCert obligations)**.** *For the carrier of Definition 2154.1, the local* $\mathsf{NameCert}_{\mathsf{SamuelCompactification}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the packet $S = (U, B, D, K, W, R, H, C, P, N)$;*
2. *bounded-readout stability, requiring every bounded uniformly continuous read to remain tied to the displayed entourage row U and the displayed readout row B under componentwise* hsame *transport;*
3. *dense-source handoff, requiring every compactification-facing consumer to pass through the dense uniform embedding row D before reading a compact boundary;*
4. *compact-boundary sealing, requiring compact-boundary observations to be exposed only through K after the stream-window schedule W and regular rational readback row R have been replayed;*
5. *ledger non-escape, excluding host compactification spaces, selected ultrafilters, quotient stream equality, countable-choice schedules, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No obligation is stored outside the displayed carrier rows.

Proof. Carrier admission is projection from Definition 2154.1. The accepted packet displays exactly the uniform source, bounded readout, dense embedding, compact boundary, stream-window, regular readback, transport, replay, provenance, and local naming rows.

Bounded-readout stability is componentwise. The row H compares U and B together, so a readout cannot be transported without the entourage data that witnesses its uniform continuity.

Dense-source handoff is the placement of D before the boundary row in every replay route. A compactification-facing consumer therefore reads the displayed dense embedding and its image-window evidence before it can request a compact boundary observation.

Compact-boundary sealing follows from the route $U, B, D, W \longmapsto R \longmapsto K$. Boundary observations are downstream of the finite windows and regular rational readback, so they cannot appear as an independent host compactification coordinate.

Finally, P and N name only the displayed rows. Since the excluded host-side principles have no carrier coordinate, no replay edge, and no provenance entry, the five obligations exhaust the local NameCert surface. □

Theorem 2154.3 (Samuel compactification uniform handoff)**.** *Every compactification-facing read exported by an accepted* $\mathsf{SamuelCompactification}^{\uparrow}$ *carrier factors through* $\mathsf{SamuelCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *More precisely, the read begins with the uniform-entourage source row U, inspects bounded uniformly continuous readouts through B, passes through the dense uniform embedding row D, replays finite windows through W, reads regular rational data through R, and only then reaches the compact-metric boundary row K named by P, N.*

Proof. Start with an accepted carrier and project its displayed rows. The provenance row P records the allowed sibling surfaces, namely the Samuel and uniform completion routes of chapter 2131 and chapter 1473, the dense embedding route of chapter 1651, the compact metric boundary of chapter 129, the stream-window and regular rational readback routes of chapter 126 and chapter 301, and the real-name seal of chapter 38.

The replay row C orders those surfaces. A compactification-facing read first consumes U, B, D, W, then reaches R, and only after that reaches K.

The transport row H preserves the same order under hsame, and N names no row outside the displayed packet. Hence every exported read factors through the listed BEDC surfaces and cannot expose a host compactification space, selected ultrafilter, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking. □

Theorem 2154.4 (Samuel compactification finite-cover boundary)**.** *Every finite-cover or compact-boundary read of an accepted* $\mathsf{SamuelCompactification}^{\uparrow}$ *packet must pass through* $\mathsf{SamuelCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The boundary exports no selected ultrafilters, quotient compactification points, countable choice, host compact spaces, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the rows of Definition 2154.1. A finite-cover or compact-boundary read can see only the uniform source row U, bounded readout row B, dense embedding row D, compact-metric boundary row K, stream windows W, regular readback R, transport H, replay C, provenance P, and local naming N.

Apply the uniform handoff of Definition 2154.3. It orders the read through Samuel completion, uniform completion, dense uniform embedding, compact metric boundary, stream windows, regular rational readback, and the Real seal before any compactification-facing boundary is public.

The finite-cover boundary is therefore a displayed route through $U, B, D, W, R, K, H, C, P, N$. Since the carrier has no coordinate for selected ultrafilters, quotient compactification points, countable choice, host compact spaces, or ambient Real completeness, no compact-boundary consumer can import those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SamuelCompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{SamuelCompactification}^{\uparrow}$ packet is generated from a uniform-entourage source row, bounded uniformly continuous readouts, DenseUniformEmbeddingUp rows, CompactMetricUp boundary rows, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SamuelCompactification}^{\uparrow}}$, seed)

Scope closed: Definition 2154.1, Definition 2154.2, and Definition 2154.3 seed the L10 compactification route from finite uniform-entourage and bounded-readout data through dense embedding, stream-window, regular readback, compact-metric boundary, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SamuelCompactification}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactification spaces, selected ultrafilters, quotient stream equality, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-readout stability, dense-source handoff, compact-boundary sealing, ledger non-escape, provenance, and local naming for BEDC.Derived.SamuelCompactificationUp.

Chapter 2155

A Concrete Naming Certificate for $\mathsf{UniformConvergenceSeries}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{UniformConvergenceSeries}^{\uparrow}$ packet records the finite handoff from uniform Cauchy control of real-series partial sums to a uniform-limit surface. It sits over the real-series packet of chapter 1581, the uniform Cauchy series packet of chapter 1736, the uniform-limit endpoint packet of chapter 463, the majorant-test route of chapter 399, the finite stream windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives later power-series, real-function, and continuity-transfer chapters a named theorem-level boundary without treating $\mathsf{UniformCauchySeries}^{\uparrow}$ itself as an infinite-sum object.

Definition 2155.1 (Uniform convergence series carrier)**.** *A* $\mathsf{UniformConvergenceSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, S, W, D, Q, L, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the real-valued term-family row. Each summand is read only through displayed finite windows before it is admitted to a partial-sum comparison.*
2. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row. It records finite series windows, displayed partial sums, and the shared real-series tail threshold.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row used by the partial sums and by the uniform tail request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested radius, dyadic slack, finite addition budget, and exhausted comparison windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed partial sums.*
6. *L is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *ledger. It records the single uniform tail threshold that works across the displayed parameter window.*
7. *M is the optional* $\mathsf{WeierstrassMTest}^{\uparrow}$ *majorant route. When present, it supplies the finite domination and majorant-tail rows used to populate L; when absent, L is supplied directly by the uniform Cauchy series packet.*
8. *E is the* $\mathsf{UniformLimit}^{\uparrow}$ *endpoint row. It becomes visible only after S, W, D, Q, L and the applicable majorant route M have been displayed.*

9. *H records componentwise* hsame *transport for the term-family, real-series, stream-window, dyadic, regular-rational, uniform-Cauchy, majorant, uniform-limit, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$A, S, W, D, Q \longmapsto L \longmapsto M \longmapsto E.$$

If the majorant route is not used, the M coordinate is the displayed empty majorant branch rather than a hidden summability principle.

11. *P is the* Pkg *provenance over* $\mathsf{UniformConvergenceSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{WeierstrassMTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceSeries}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these twelve rows. Its classifier compares accepted packets only through $A, S, W, D, Q, L, M, E, H, C, P, N$ and has no coordinate for host infinite sums, selected limits, quotient equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.UniformConvergenceSeriesUp.UniformConvergenceSeriesTasteGate single carrier alignment`

Theorem 2155.2 (Uniform convergence series NameCert obligations)**.** *For every accepted* UniformConvergenceSeri *carrier*

$$U = (A, S, W, D, Q, L, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformConvergenceSeries}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the term-family row, real-series partial-sum row, stream schedule, dyadic tolerance ledger, regular-rational readback, uniform Cauchy ledger, optional majorant route, uniform-limit endpoint, transport, replay, provenance, and local naming row;*

2. *partial-sum coverage, requiring every convergence request to read the displayed* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row S before reaching any limit endpoint;*

3. *uniform-tail exactness, requiring the single threshold row L to answer all displayed parameter-window requests through the same dyadic tolerance ledger D and schedule row W;*

4. *regular readback discipline, requiring Q to be the only regular-rational route by which the finite partial sums are handed to the uniform-limit endpoint;*

5. *majorant confinement, requiring M to be either the displayed* $\mathsf{WeierstrassMTest}^{\uparrow}$ *domination route or the displayed empty majorant branch, never a host summability assertion;*

6. *endpoint factorization, requiring E to remain downstream of S, W, D, Q, L and the applicable M branch;*

7. *transport, replay, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise;*

8. *non-escape, excluding host infinite sums, selected limits, quotient equality, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No uniform-convergence-series obligation is stored outside these rows.

Proof. Project the accepted packet $U = (A, S, W, D, Q, L, M, E, H, C, P, N)$. The carrier exposes exactly the term-family row, the real-series partial-sum row, the shared stream schedule, the dyadic ledger, the regular readback, the uniform Cauchy ledger, the optional majorant route, the uniform-limit endpoint, and the structural rows.

The convergence request first consumes the real-series partial sums. The row S supplies the finite partial-sum windows, while W fixes the displayed schedule and D fixes the dyadic tolerance budget used by the comparison.

Uniformity is supplied by L. The same threshold row must answer every displayed parameter-window request, so a consumer cannot replace the packet by pointwise convergence data or by a separately chosen limit at each parameter.

The regular readback and majorant branch are then confined. The row Q is the only route into the regular-rational handoff, and M either records the finite Weierstrass majorant route or records that no majorant branch is used.

Finally E, H, C, P, N seal, transport, replay, source, and name that same finite route. Since host infinite sums, selected limits, quotient equality, countable choice, open-cover compactness, and ambient real completeness have no carrier coordinate or replay edge, the listed obligations exhaust the local NameCert surface. □

Theorem 2155.3 (Uniform convergence series RealSeries handoff)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$ *endpoint read of an accepted* $\mathsf{UniformConvergenceSeries}^{\uparrow}$ *carrier factors through the finite route*

$$A, S, W, D, Q \longmapsto L \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed real-valued term family, $\mathsf{RealSeries}^{\uparrow}$ *partial-sum windows,* $\mathsf{StreamName}^{\uparrow}$ *schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, one* $\mathsf{UniformCauchySeries}^{\uparrow}$ *tail ledger, the applicable* $\mathsf{WeierstrassMTest}^{\uparrow}$ *branch, the* $\mathsf{UniformLimit}^{\uparrow}$ *endpoint row, componentwise transport, continuation replay, package provenance, and local naming. It receives no host infinite sum, selected limit, quotient equality, countable-choice schedule, open-cover compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted endpoint read. The term-family row A is visible only through the real-series partial-sum row S, and those partial sums are inspected along the schedule row W with the tolerance accounting recorded in D.

The regular-rational readback is then forced through Q. It is the unique displayed handoff from the finite partial sums to the uniform Cauchy comparison surface, so no quotient equality of series or selected infinite sum is introduced.

The uniform Cauchy step is the row L. It supplies the common tail threshold that works for every displayed parameter-window request; if a majorant proof is used, the row M records exactly the Weierstrass domination and majorant-tail rows that populate L.

Only after this route is displayed can E expose the $\mathsf{UniformLimit}^{\uparrow}$ endpoint. The structural rows H, C, P, N preserve the same route under componentwise transport, continuation replay, provenance, and local naming.

Induction over the displayed route order proves the factorization. The base stage reads A, S, W, D, Q, the uniform-tail stage reads L, the majorant branch reads M, and the terminal stage reads E, H, C, P, N. There is no coordinate outside this route from which a consumer could extract a host infinite sum, selected limit, quotient equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformConvergenceSeries}^{\uparrow}$ packet is generated from a real-valued term-family row, RealSeries partial-sum windows, StreamName schedules, a DyadicRatCore tolerance ledger, RegSeqRat readback, one UniformCauchySeries tail ledger, an optional WeierstrassMTest majorant route, a UniformLimit endpoint row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2155.1, Definition 2155.2, and Definition 2155.3 seed the finite theorem-level route from RealSeries partial sums and UniformCauchySeries tail control to the UniformLimit endpoint surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum coverage, uniform-tail exactness, regular readback discipline, majorant confinement, endpoint factorization, transport, replay, provenance, and local naming for BEDC.Derived.UniformConvergenceSeriesUp.

Chapter 2156

A Concrete Naming Certificate for $\mathsf{MetricCompletionMonad}^{\uparrow}$

Following Mac Lane's monad formulation together with Kelley and Bishop-style metric completion, the $\mathsf{MetricCompletionMonad}^{\uparrow}$ packet records the L10 unit-and-flattening interface for displayed metric completions. It consumes the completion carrier of chapter 1506, the extension route of chapter 2124, the finite completion monad surface of chapter 378, and the regular-window monad of chapter 2136. The packet gives later completion functoriality and complete-metric consumers a single monadic handoff without quotient Cauchy-name equality, selected representatives, countable choice, or an ambient completeness assumption.

Definition 2156.1 (MetricCompletionMonad carrier)**.** *A* $\mathsf{MetricCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Q, U, F, V, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, metric-distance evidence, and finite observation histories consumed by the completion interface.*
2. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion-object row. It names the displayed Cauchy-filter or Cauchy-net completion packet over X.*
3. *U is the unit-embedding row. It inserts a displayed point of X into Q by copying the source metric observations into the completion-facing route.*
4. *F is the multiplication or flattening row. It flattens a completion object whose entries are themselves displayed completion rows by replaying their finite Cauchy evidence through the same metric source.*
5. *V is the* $\mathsf{MetricCompletionUniversalProperty}^{\uparrow}$ *replay row. It records that unit and flattening consumers may be read through the displayed extension-and-uniqueness surface, rather than through a host quotient of Cauchy names.*
6. *S is the separated-comparison row. It compares two monadic handoffs only through the zero-distance classifier route already displayed by the completion packet.*
7. *H records componentwise* hsame *transport for the metric source, completion object, unit row, flattening row, universal-property replay, separated comparison, package row, and local naming row.*

8. *C records displayed* Cont *replay of the route*

$$X, Q \longmapsto U, F \longmapsto V \longmapsto S.$$

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionMonad}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, MetricCompletionUniversalProp CauchyCompletionMonad$^{\uparrow}$, RegularCauchyCompletionMonad$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionMonad}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $X, Q, U, F, V, S, H, C, P, N$ and has no coordinate for quotient Cauchy-name equality, selected representatives, countable choice, host completion equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricCompletionMonadUp.MetricCompletionMonadTasteGate single carrier alignment`

Theorem 2156.2 (MetricCompletionMonad NameCert obligations)**.** *For an accepted* $\mathsf{MetricCompletionMonad}^{\uparrow}$ *carrier*

$$M = (X, Q, U, F, V, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionMonad}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed metric source X and completion object Q;*
2. *monadic row display for the unit embedding U and flattening route F;*
3. *universal-property replay through V and separated comparison through S;*
4. *structural stability by componentwise* hsame *transport H and displayed* Cont *replay C;*
5. *provenance and local naming through P and N, with no exported quotient, selector, choice, or ambient-completeness coordinate.*

Proof. Project the accepted carrier of Definition 2156.1. The projection exposes the metric source, completion object, unit row, flattening row, universal-property replay, separated comparison, transport, replay, provenance, and local naming rows.

The five obligations are grouped from those displayed rows. The first two obligations read the carrier and monadic data, the third reads the extension and comparison surfaces, the fourth reads stability, and the fifth reads provenance and local naming.

Because the carrier contains no additional coordinate, a consumer cannot obtain quotient Cauchy-name equality, a selected representative, countable choice, host completion equality, or ambient completeness from the certificate. □

Theorem 2156.3 (MetricCompletionMonad unit and multiplication handoff)**.** *For every accepted* $\mathsf{MetricCompletionMonad}^{\uparrow}$ *carrier $M = (X, Q, U, F, V, S, H, C, P, N)$, the unit row U and multiplication row F hand completion-facing consumers through the displayed metric-completion rows. The left-unit, right-unit, and associativity comparisons are comparisons of the finite route*

$$X, Q, U, F, V, S, H, C, P, N,$$

not comparisons of quotient Cauchy names or selected representatives.

Proof. The unit handoff starts with X and Q. The row U copies the displayed source observations into the completion object before any universal-property or separated-comparison read is available.

The multiplication handoff starts with a completion row whose entries are themselves displayed completion rows. The row F flattens that packet by replaying the same metric source and completion-object rows, then routes the result through V and S.

Left unit, right unit, and associativity are therefore checked by comparing the displayed route before and after replay. The rows H, C, P, N preserve the route componentwise and record that no quotient equality, selected representative, countable-choice schedule, or ambient completeness witness is present. □

Theorem 2156.4 (MetricCompletionMonad non-escape)**.** *Every public consumer route exported by an accepted* $\mathsf{MetricCompletionMonad}^{\uparrow}$ *carrier is exhausted by the source metric row, completion-object row, unit embedding, flattening route, universal-property replay, separated comparison,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The packet exports no quotient Cauchy-name equality, selected representatives, countable choice, host completion equality, or ambient completeness theorem.*

Proof. Read an accepted carrier through its displayed tuple. The only rows available to a public consumer are $X, Q, U, F, V, S, H, C, P, N$.

The monadic rows U and F are downstream of the metric source and completion object. The universal-property row V and separated-comparison row S are downstream of those same monadic rows, so they cannot introduce an independent completion object.

Finally, the structural rows transport, replay, package, and name the same finite packet. Since none of the excluded principles appears as a row of the carrier, no public route can export such a principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionMonad}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionMonad}^{\uparrow}$ packet is generated from a MetricUp source row, a MetricCompletionUp completion-object row, unit embedding, multiplication or flattening, MetricCompletionUniversalProperty replay, separated comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionMonad}^{\uparrow}}$, seed)

Scope closed: Definition 2156.1, Definition 2156.2, Definition 2156.3, and Definition 2156.4 seed the finite monadic completion interface over displayed metric source, completion object, unit, flattening, universal-property, separated-comparison, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionMonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy-name equality, selected representatives, countable choice, host completion equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, five-row NameCert surface, unit and multiplication handoff, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionMonadUp.

Chapter 2157

A Concrete Naming Certificate for $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$

Following Bourbaki and Kelley on first-countability and sequential closure, the $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ packet records the finite route by which a local countable base supplies Frechet-tail neighbourhood tests and then exposes a displayed sequential-closure witness. It sits after the local-basis surface of chapter 2053, the cofinite-tail interface of chapter 1564, the sequence-closure packet of chapter 1571, the real sequence limit handoff of chapter 1413, the finite windows of chapter 126, the regular rational readback rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names only the displayed route from countable local bases to sequence-limit closure evidence; it does not identify host points, choose subsequences, quotient filters, prove open-cover compactness, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2157.1 (First-countable sequential-closure carrier)**.** *A* $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, T, Q, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FirstCountableSpace}^{\uparrow}$ *local-basis row. It records the accepted point-window, the local countable neighbourhood ledger, and the topology classifier that every closure request must read.*
2. *T is the* $\mathsf{FrechetFilter}^{\uparrow}$ *tail-neighbourhood row. It turns the local countable request into a cofinite-tail budget before any sequence window is inspected.*
3. *Q is the* $\mathsf{SequentialClosure}^{\uparrow}$ *witness row. It names the displayed source subset, candidate closure point, neighbourhood tests, and metric tests supplied to the closure consumer.*
4. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row, available only when the sequential witness is read through a real-valued sequence limit.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the local basis enumeration, the Frechet threshold, the inspected sequence entries, and the limit handoff.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed metric and real-sequence windows into regular rational data.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the neighbourhood radius, Frechet-tail slack, sequence-window budget, and real-limit comparison scale.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, T, Q, L, W, R, D have been displayed.*
9. *H records componentwise* hsame *transport for the local-basis, Frechet-tail, sequential-closure, real-sequence-limit, stream-window, regular-rational, dyadic-tolerance, Real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the route*

$$B, T \longmapsto Q \longmapsto L, W, R, D \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$, $\mathsf{FirstCountableSpace}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FirstCountableSequentialClosure}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, selected subsequences, quotient filters, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached closure operator.*

Lean4 ✓ `BEDC.Derived.FirstCountableSequentialClosureUp.FirstCountableSequentialClosureTasteGate single carrier alignment`

Theorem 2157.2 (First-countable sequential-closure NameCert obligations)**.** *For every accepted* $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ *carrier*

$$F = (B, T, Q, L, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FirstCountableSequentialClosure}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the first-countable local basis, Frechet tail, sequential-closure witness, real-sequence-limit handoff, stream windows, regular-rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming row;*
2. *local-basis coverage, requiring every closure-neighbourhood request to read the displayed* $\mathsf{FirstCountableSpace}^{\uparrow}$ *row B before it can ask for sequence data;*
3. *Frechet-tail exactness, requiring the cofinite-tail row T to supply the tail threshold used by the same neighbourhood request;*
4. *sequential-witness exposure, requiring Q to be the only row that exports the displayed sequence-closure witness downstream of B, T;*
5. *real-limit handoff, requiring L, W, R, D to be consumed before the terminal Real seal E is exposed;*
6. *transport and replay stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise;*
7. *non-escape, excluding host point equality, selected subsequences, quotient filters, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and detached host closure.*

No first-countable sequential-closure obligation is stored outside these rows.

Proof. Project the accepted carrier $F = (B, T, Q, L, W, R, D, E, H, C, P, N)$. The projection exposes exactly the local first-countable basis, the Frechet tail, the sequential-closure witness, the real-sequence-limit handoff, the stream windows, the regular readback, the dyadic ledger, the Real seal, and the structural rows.

The closure request first consumes the local basis row B. This row fixes the displayed point-window and the countable neighbourhood ledger, so the request cannot replace the local basis by a host family of neighbourhoods.

The Frechet tail row T then supplies the cofinite threshold for the same request. Once the threshold is displayed, the sequential witness row Q reads sequence entries through that tail budget rather than through a selected subsequence or quotient filter.

The real-valued limit side is downstream of the witness. The handoff row L, window ledger W, regular readback R, and dyadic ledger D must all be read before the terminal Real seal E becomes visible.

Finally H, C, P, N transport, replay, source, and name only that finite route. Since host point equality, selected subsequences, quotient filters, open-cover compactness, countable choice, ambient real completeness, and detached host closure have no carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2157.3 (First-countable sequential-closure route)**.** *Let*

$$F = (B, T, Q, L, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ *carrier. Every consumer that reads the displayed closure witness factors through the finite route*

$$B, T \longmapsto Q \longmapsto L, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a $\mathsf{FirstCountableSpace}^{\uparrow}$ *local basis, a* $\mathsf{FrechetFilter}^{\uparrow}$ *tail-neighbourhood row, a* $\mathsf{SequentialClosure}^{\uparrow}$ *witness, a* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no host point equality, selected subsequence, quotient filter, open-cover compactness principle, countable-choice row, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or detached closure operator.*

Proof. Start with an accepted consumer read of the closure witness. Carrier admission gives access to the witness row Q only together with the local-basis row B and the Frechet-tail row T.

Replay the route through C. The consumer first reads the point-window and local neighbourhood ledger from B, then the cofinite threshold and tail-neighbourhood surface from T, and only then reaches the displayed sequential witness Q.

The real-limit data remain downstream. The rows L, W, R, D expose the real sequence limit, finite windows, regular rational readback, and dyadic tolerance accounting before the Real seal E is reached.

Transport by H, provenance by P, and local naming by N preserve that same order. No remaining carrier row can export host point equality, a selected subsequence, quotient filter equality, open-cover compactness, countable choice, ambient Real completeness, or detached host closure, so every accepted consumer factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FirstCountableSequentialClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{FirstCountableSequentialClosure}^{\uparrow}$ packet is generated from a FirstCountableSpace local-basis row, a FrechetFilter tail-neighbourhood row, a SequentialClosure witness row, a RealSequenceLimit handoff row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirstCountableSequentialClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2157.1, Definition 2157.2, and Definition 2157.3 seed the finite route from local countable bases and Frechet tails to displayed sequential-closure witnesses.

Formal verification: FormalStatus(NameCert$_{\text{FirstCountableSequentialClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host point equality, selected subsequences, quotient filters, open-cover compactness, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-basis coverage, Frechet-tail exactness, sequential-witness exposure, real-limit handoff, transport, replay, provenance, and local naming for BEDC.Derived.FirstCountableSequentialClo

Chapter 2158

A Concrete Naming Certificate for $\mathsf{QuasiMetric}^{\uparrow}$

Following Lawvere, Nachbin, and Kelley, the $\mathsf{QuasiMetric}^{\uparrow}$ packet records the finite directed-distance surface that sits beside the symmetric metric certificate of chapter 57, the directed entourage carrier of chapter 2334, the filter route of chapter 333, the net route of chapter 589, and the completion consumer of chapter 1473. Its Real-valued distance rows read the terminal seal of chapter 38 through the dyadic tolerance source of chapter 291, so completion-facing consumers see an oriented distance ledger before any uniform-entourage handoff is exposed. The chapter names only that directed NameCert surface; it does not assert symmetry, separatedness, metric completeness, open-cover compactness, quotient distance, selected limits, or a standard bridge.

Definition 2158.1 (QuasiMetric carrier surface)**.** *A* $\mathsf{QuasiMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source carrier row. It names the traditional quasi-metric source and the finite observation context in which directed distance requests are admitted.*
2. *X is the point carrier row. It records the admitted point names and their classifier, but it does not identify points by zero distance.*
3. *D is the directed $\mathsf{Real}^{\uparrow}$-valued distance row. It records oriented requests $d(x, y)$ and keeps the reverse request $d(y, x)$ as a separate row when that reverse request is present.*
4. *Z is the zero-on-diagonal row. It records $d(x, x) = 0$ as a displayed boundary condition on directed distance requests.*
5. *T is the forward triangle ledger. It records the directed comparison from $d(x, z)$ through the ordered pair of rows $d(x, y)$ and $d(y, z)$.*
6. *B is the directed-ball row. It records finite requests of the form $B^{+}(x, \varepsilon)$ by displaying the center, radius, and incoming point through D, Z, T.*
7. *F is the $\mathsf{CauchyFilter}^{\uparrow}$ row. It records filter-base Cauchy reads whose directed balls are supplied by B and whose comparisons remain ordered.*
8. *R is the $\mathsf{CauchyNet}^{\uparrow}$ row. It records net-tail Cauchy reads whose eventual directed-ball comparisons are supplied by B.*

9. *E is the uniform-reflection row. It records the finite handoff from directed balls to* $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, *and completion-facing entourage requests without installing symmetry as primitive data.*

10. *H records componentwise* hsame *transport for source, point-carrier, directed distance, zero diagonal, triangle, directed-ball, Cauchy filter, Cauchy net, uniform reflection, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a consumer request through point admission, directed distance readback, zero diagonal or triangle evidence, directed-ball inspection, Cauchy filter or net readback, uniform reflection, provenance, and local naming.*

12. *P is the* Pkg *provenance over* $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, X, D, Z, T, B, F, R, E, H, C, P, N$. It has no coordinate for symmetry, separated point identity, metric completeness, open-cover compactness, quotient distance, selected limits, quotient completion, or bridge checking.

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricTasteGate single carrier alignment`

Theorem 2158.2 (QuasiMetric NameCert obligation surface)**.** *For every accepted* $\mathsf{QuasiMetric}^{\uparrow}$ *carrier*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the source, point carrier, directed* $\mathsf{Real}^{\uparrow}$*-valued distance row, zero diagonal row, forward triangle row, directed-ball row, Cauchy filter row, Cauchy net row, uniform-reflection row, componentwise transport, displayed replay, provenance, and local name;*

2. *asymmetric distance readback, requiring every exported distance request to keep the ordered pair of point rows visible before the* $\mathsf{Real}^{\uparrow}$ *distance seal is read;*

3. *zero diagonal stability, requiring self-distance to be stored in Z rather than imported from a host metric law;*

4. *oriented triangle stability, requiring each triangle comparison to be displayed in T with its ordered endpoints;*

5. *directed-ball handoff, requiring every ball consumer to pass through B after D, Z, T have been displayed;*

6. *Cauchy filter and Cauchy net routing, requiring filter-base and net-tail comparisons to read F, R through the directed-ball row;*

7. *classifier transport, requiring accepted packets to move only by the componentwise* hsame *row H and the replay row C;*

8. *uniform-reflection handoff, requiring every quasi-uniform, uniform, or completion-facing request to pass through E after the directed balls and Cauchy rows have been displayed;*

9. *ledger non-escape, excluding symmetry, separatedness, metric completeness, open-cover compactness, quotient distance, quotient completion, selected limits, and any standard bridge.*

No quasi-metric NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier namecert obligations`

Proof. Project the carrier of Definition 2158.1. The tuple lists exactly the source, point carrier, directed distance, zero diagonal, forward triangle, directed-ball, Cauchy filter, Cauchy net, uniform-reflection, transport, replay, provenance, and local naming rows.

Asymmetric distance readback is forced by D. A request for $d(x, y)$ keeps x and y in that order, and a request for $d(y, x)$ is a distinct displayed row unless the carrier separately records a comparison between them.

The zero diagonal and triangle obligations are the displayed rows Z and T. Thus reflexive zero and forward triangle comparison are local quasi-metric data, not imported symmetry or host metric structure.

Directed-ball reads use B only after D, Z, T have been displayed. The Cauchy filter and net rows F, R read those directed balls rather than a quotient completion object.

Classifier transport is componentwise through H, while C replays the same ordered route for consumers. The uniform-reflection handoff uses E only after the directed distance, ball, and Cauchy rows have been read.

The remaining exclusions follow from the carrier inventory. Since no coordinate names symmetry, separated identity, metric completeness, open-cover compactness, quotient distance, quotient completion, selected limits, or a standard bridge, none of those data can be exported by the local certificate. □

Theorem 2158.3 (QuasiMetric directed ball handoff)**.** *Let*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N)$$

be an accepted $\mathsf{QuasiMetric}^{\uparrow}$ *packet. Every* $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer that asks for a distance-presented directed ball reaches it only through the finite route*

$$S, X, D, Z, T \longmapsto B \longmapsto F, R, E \longmapsto H, C, P, N.$$

The handoff may reflect to quasi-uniform, uniform, Cauchy-filter, Cauchy-net, or completion-facing rows, but it receives no symmetry theorem, no separated point identity, no metric completeness, no open-cover compactness, no quotient distance, no quotient completion, and no standard bridge.

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier directed ball handoff`

Proof. Start with the accepted carrier. The consumer first reads the source and point rows S, X, then the directed distance ledger D.

The diagonal and triangle rows Z, T are read before any ball-facing handoff. They supply the quasi-metric boundary conditions needed to form directed balls without turning the directed distance into a symmetric metric.

Only after those rows are displayed does the consumer reach B. Filter and net consumers then read F, R, while quasi-uniform, uniform, and completion-facing consumers read the reflection row E.

The support rows H, C, P, N transport, replay, package, and name the same finite route. Therefore the handoff exports finite directed-distance and directed-ball data, not symmetry, separatedness, metric completeness, open-cover compactness, quotient distance, quotient completion, or bridge evidence. □

Theorem 2158.4 (QuasiMetric non-escape boundary)**.** *Every accepted* $\mathsf{QuasiMetric}^{\uparrow}$ *read used by* $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *factors through the displayed rows of Definition 2158.1. In particular, a metric or completion consumer cannot bypass the directed-distance row* D*, the forward triangle ledger* T*, the directed-ball row* B*, the Cauchy rows* F, R*, and the uniform-reflection row* E *by importing symmetry, quotient completion, selected limits, host metric equality,* `Classical.choice`*, or* `Quot.sound`*.*

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier nonescape`

Proof. Apply Definition 2158.2. The NameCert surface stores all quasi-metric obligations in $S, X, D, Z, T, B, F, R, E, H, C, P, N$ and stores none outside those rows.

Apply Definition 2158.3. Every directed-ball, Cauchy, quasi-uniform, uniform, and completion-facing consumer reaches the public route only after D, Z, T, B, F, R, E have been displayed.

A metric-facing consumer may read the sibling metric surface of chapter 57 only as a reflected reader named by E or P. Since the carrier has no primitive coordinate for symmetry, quotient completion, selected limits, host equality, `Classical.choice`, or `Quot.sound`, those exits remain unavailable. □

Theorem 2158.5 (QuasiMetric scope tightening)**.** *Let*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N)$$

be an accepted $\mathsf{QuasiMetric}^{\uparrow}$ *packet. Its directed-distance scope is exactly the displayed route through source, point carrier, directed* $\mathsf{Real}^{\uparrow}$ *distance, zero diagonal, forward triangle, directed balls, Cauchy filter, Cauchy net, uniform reflection, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. Consequently every* $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of* Q *is bounded by the finite route*

$$S, X, D, Z, T \longmapsto B \longmapsto F, R, E \longmapsto H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier scope tightening`

Proof. Apply the obligation surface of Definition 2158.2. It identifies the carrier rows $S, X, D, Z, T, B, F, R, E, H, C,$ as the full local NameCert inventory and places directed distance, zero diagonal, and forward triangle data before every public ball or completion-facing read.

Next use the directed-ball handoff of Definition 2158.3. It routes quasi-uniform, uniform, Cauchy-filter, Cauchy-net, and completion-facing consumers through B, F, R, E only after the directed-distance prefix has been displayed.

Finally apply the non-escape boundary of Definition 2158.4. Since the carrier has no coordinate for symmetry, separated point identity, metric completeness, open-cover compactness, quotient distance, quotient completion, selected limits, host metric equality, `Classical.choice`, or `Quot.sound`, the displayed row route is not merely sufficient but exact for the scoped directed-distance surface. □

Theorem 2158.6 (QuasiMetric directed-ball triangle refinement)**.** *Let*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N)$$

be an accepted $\mathsf{QuasiMetric}^{\uparrow}$ *packet. If two displayed directed-ball reads in* B *have compatible middle endpoint, with the first read routed from* x *to* y *and the second from* y *to* z*, then their composed*

directed-ball read is exposed only through the asymmetric triangle row T, preserving the source row S, point rows X, directed-distance ledger D, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier directed ball triangle refinement`

Proof. Project the carrier surface of Definition 2158.1. The two directed-ball reads have no source other than S, X, D, Z, T, B, and the middle endpoint compatibility is a displayed point-row match inside X rather than a separated point identity.

Use the oriented triangle obligation in Definition 2158.2. It requires each triangle comparison to be stored in T with ordered endpoints, so composing the x to y and y to z ball reads can only read the displayed asymmetric triangle witness.

The handoff theorem Definition 2158.3 then routes the composed directed ball through B and onward to any Cauchy, uniform, or completion-facing consumer. The support rows H, C, P, N transport, replay, package, and name the same directed-distance witnesses, while Definition 2158.4 excludes symmetry, quotient completion, selected limits, host metric equality, `Classical.choice`, and `Quot.sound`. □

Theorem 2158.7 (QuasiMetric directed-distance scope upgrade)**.** *Let*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N)$$

be an accepted $\mathsf{QuasiMetric}^{\uparrow}$ *packet. Every accepted directed-distance read used by* $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *must pass through the displayed rows $S, X, D, Z, T, B, F, R, E, H, C, P, N$ of Definition 2158.1. The scope is bound to the finite* BHist *carrier, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* NameCert *naming, Real-valued distance readback, dyadic tolerance comparison, and directed-ball routing.*

Consequently the directed-distance route is a scoped BEDC surface, not a host metric object: no consumer receives symmetry, separated point identity, quotient distance, selected completion limit, ambient metric completeness, open-cover compactness, `Classical.choice`, `Quot.sound`, *or bridge evidence from the* $\mathsf{QuasiMetric}^{\uparrow}$ *packet.*

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier directed distance scope upgrade`

Proof. Start with Definition 2158.2. It identifies $S, X, D, Z, T, B, F, R, E, H, C, P, N$ as the complete local NameCert inventory and places the Real-valued directed-distance row D, zero row Z, and forward triangle row T before every public read.

Next apply Definition 2158.3. Directed-ball consumers reach B only after S, X, D, Z, T have been displayed, and Cauchy, quasi-uniform, uniform, and completion-facing consumers then pass through F, R, E before the support rows H, C, P, N are read.

Use Definition 2158.4 to close the non-escape boundary. It rules out host symmetry, separated equality, quotient completion, selected limits, `Classical.choice`, and `Quot.sound` as alternate exits from the packet.

Finally invoke Definition 2158.6. The composed directed-ball comparisons read the asymmetric triangle row T and preserve the same BHist, hsame, Cont, Pkg, and NameCert support. Thus every listed consumer receives exactly the directed-distance scope carried by the displayed rows. □

Theorem 2158.8 (QuasiMetric public certificate)**.** *For every accepted* $\mathsf{QuasiMetric}^{\uparrow}$ *packet*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N),$$

the public directed-distance certificate is the assembled row route

$\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, Rea

It is obtained by combining the carrier surface of Definition 2158.1, the NameCert obligations of Definition 2158.2, the directed-ball handoff of Definition 2158.3, the non-escape boundary of Definition 2158.4, the directed-ball triangle refinement of Definition 2158.6, and the directed-distance scope upgrade of Definition 2158.7. The certificate exports directed $\mathsf{Real}^{\uparrow}$*-valued distance, zero diagonal, forward triangle, directed-ball, Cauchy, uniform-reflection, transport, replay, provenance, and local naming rows, but it exports no symmetry, separatedness, quotient completion, selected limits, host equality,* `Classical.choice`*,* `Quot.sound`*, or bridge evidence.*

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier public certificate`

Proof. First read the carrier surface of Definition 2158.1. It displays the source and point rows S, X, the directed $\mathsf{Real}^{\uparrow}$ distance row D, the zero and forward-triangle rows Z, T, the directed-ball row B, Cauchy filter and net rows F, R, the uniform-reflection row E, and the structural rows H, C, P, N.

Next apply the NameCert obligation surface of Definition 2158.2. That theorem identifies carrier admission, directed distance readback, zero diagonal stability, forward triangle stability, directed-ball handoff, Cauchy routing, classifier transport, uniform reflection, and ledger non-escape as the exact local obligations.

Use Definition 2158.3 and Definition 2158.6 to assemble the public route. Directed balls are read only after S, X, D, Z, T, composed directed-ball comparisons read the asymmetric triangle row T, and Cauchy, quasi-uniform, uniform, and completion-facing consumers pass through B, F, R, E before the support rows are read.

Finally use Definition 2158.4 and Definition 2158.7. They close the route as a public certificate over the displayed rows and rule out alternate exits through symmetry, separatedness, quotient completion, selected limits, host equality, `Classical.choice`, `Quot.sound`, or bridge evidence. □

Theorem 2158.9 (QuasiMetric public NameCert export)**.** *For every accepted* $\mathsf{QuasiMetric}^{\uparrow}$ *packet*

$$Q = (S, X, D, Z, T, B, F, R, E, H, C, P, N),$$

the public directed-distance certificate of Definition 2158.8 is exactly the local $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$ *export over the source row S, point carrier X, directed* $\mathsf{Real}^{\uparrow}$ *distance row D, zero row Z, forward triangle row T, directed-ball row B, Cauchy filter row F, Cauchy net row R, uniform-reflection row E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N. The export records dyadic tolerance comparison and directed-ball routing for the Real-valued distance surface, but it exports no symmetry, separated equality, quotient completion, selected limit, ambient completeness, open-cover compactness, or bridge evidence.*

Lean4 ✓ `BEDC.Derived.QuasiMetricUp.QuasiMetricCarrier public namecert export`

Proof. First apply Definition 2158.8. It assembles the public directed-distance certificate from the carrier, obligation surface, directed-ball handoff, triangle refinement, scope route, and non-escape boundary.

Next use Definition 2158.7. It keeps every Real- or dyadic-facing distance read inside $S, X, D, Z, T, B, F, R, E, H$ so the public certificate has no hidden parameter transport outside the displayed packet.

Finally apply Definition 2158.4. Since symmetry, separated equality, quotient completion, selected limits, host equality, `Classical.choice`, `Quot.sound`, and bridge evidence are absent

from the carrier, the public certificate is precisely the local $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$ export over the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasiMetric}^{\uparrow}$ packet is generated from a source row, a point carrier, a directed RealUp-valued distance row, a zero diagonal row, a forward triangle row, a directed-ball row, Cauchy filter and Cauchy net rows, a uniform-reflection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$, publicClosed)

Scope closed: Definition 2158.8 is the exported public directed-distance route assembled from the displayed source, point, Real-valued distance, zero diagonal, forward triangle, directed-ball, Cauchy, uniform-reflection, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{QuasiMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close symmetry, separatedness, metric completeness, open-cover compactness, quotient distance, quotient completion, selected limits, standard bridge evidence, Lean verification, or bridge checking.

Upgrade path: Public closure requires checked carrier admission, directed distance readback, zero diagonal stability, forward triangle stability, directed-ball handoff, Cauchy routing, uniform reflection, classifier transport, and ledger non-escape for BEDC.Derived.QuasiMetricUp.

Chapter 2159

A Concrete Naming Certificate for LeibnizAlternatingRemainder$^{\uparrow}$

Following Leibniz and Bishop–Bridges constructive alternating-series analysis, the LeibnizAlternatingRemainder$^{\uparrow}$ packet records the finite route by which an alternating real series exposes a next-term remainder budget before any completion-facing consumer may read a regular rational handoff or a real seal. It sits after the finite partial-sum surface of RealSeries$^{\uparrow}$ in chapter 1581, the convergence-test route of AlternatingSeriesTest$^{\uparrow}$ in chapter 1483, the null-tail surface of NullSequence$^{\uparrow}$ in chapter 380, the finite observation windows of StreamName$^{\uparrow}$ in chapter 126, the regular rational readback of RegSeqRat$^{\uparrow}$ in chapter 301, the dyadic tolerance ledger of DyadicRatCore$^{\uparrow}$ in chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38. It also names the finite completion-facing row used by the local Leibniz streams of RealAnalytic$^{\uparrow}$ in chapter 74: the analytic chapter may display unary Leibniz partial sums, while this packet isolates the remainder certificate consumed by Real and Completion routes.

Definition 2159.1 (Leibniz alternating remainder carrier)**.** *A* LeibnizAlternatingRemainder$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, \Sigma, M, T, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* RealSeries$^{\uparrow}$ *source row. It exposes the real-valued summand stream and the displayed partial sums only through the finite series surface already named for real series.*
2. *Σ is the alternating-sign ledger. It records the parity row assigning the sign of every inspected summand and the adjacent partial-sum direction used by the remainder estimate.*
3. *M is the monotone null-tail row. It records nonnegative magnitude reads, the displayed monotonicity comparisons on the inspected tail, and the* NullSequence$^{\uparrow}$ *request ensuring that the next omitted magnitude can be made smaller than any requested dyadic tolerance.*
4. *T is the* AlternatingSeriesTest$^{\uparrow}$ *handoff row. It records that the displayed source, sign ledger, monotone magnitude row, and tail schedule are accepted by the alternating-series-test surface before any remainder read is exported.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *remainder-budget ledger. For each precision request it records the dyadic tolerance, the selected cutoff, the next omitted magnitude, and the finite comparison certifying that the remainder budget is bounded by that next magnitude.*

6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family for summands, signs, magnitudes, partial sums, next omitted terms, dyadic comparisons, and endpoint-facing reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic remainder budgets.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Σ, M, T, D, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the source, sign ledger, monotone null-tail row, alternating-test handoff, dyadic budget, stream windows, regular readback, real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay of the route*

$$S, \Sigma, M, T, D, W \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAnalytic}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingRemainder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, \Sigma, M, T, D, W, R, E, H, C, P, N$. It has no coordinate for a selected infinite sum, quotient stream equality, host real equality, countable choice, trichotomy outside the displayed dyadic comparisons, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an analytic endpoint not routed through the listed rows.*

Lean4 ✓ `BEDC.Derived.LeibnizAlternatingRemainderUp.LeibnizAlternatingRemainderTasteGate single carrier alignment`

Theorem 2159.2 (Leibniz alternating remainder NameCert obligations)**.** *For every accepted* $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ *carrier*

$$L = (S, \Sigma, M, T, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingRemainder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the twelve displayed rows;*
2. $\mathsf{RealSeries}^{\uparrow}$ *source exposure, requiring every summand and partial-sum read to pass through S;*
3. *alternating-sign exposure, requiring Σ to bind every inspected partial-sum step to its finite parity row;*
4. *monotone null-tail exactness, requiring M to display the nonnegative magnitude row, the observed monotone comparisons, and the* $\mathsf{NullSequence}^{\uparrow}$ *tail request used by the remainder budget;*
5. *alternating-test compatibility, requiring T to consume only S, Σ, M and the displayed tail schedule;*
6. *dyadic remainder-budget exactness, requiring D to pair every requested tolerance with a cutoff, next omitted magnitude, and comparison row;*
7. *finite-window and regular-readback compatibility, requiring W and R to expose the same summand, partial-sum, omitted-term, and budget data;*

8. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring* E *to remain downstream of* S, Σ, M, T, D, W, R*;*

9. *componentwise transport, displayed replay, provenance, and local naming for the same route through* H, C, P, N*.*

No Leibniz remainder obligation is stored outside these rows.

Proof. Project the accepted packet $L = (S, \Sigma, M, T, D, W, R, E, H, C, P, N)$. The carrier exposes exactly the source, sign, magnitude, alternating-test, dyadic-budget, window, readback, seal, and structural rows.

The source and sign obligations come first. The row S provides only finite $\mathsf{RealSeries}^{\uparrow}$ term and partial-sum observations, while Σ attaches each inspected summand to the parity ledger used by the alternating partial sums.

The tail obligations are finite. The row M displays the nonnegative magnitude reads, their monotone comparisons on the inspected tail, and the null-sequence request. The handoff row T then records that these displayed rows enter the $\mathsf{AlternatingSeriesTest}^{\uparrow}$ surface.

The remainder budget is fixed by D. For each requested tolerance, D names a cutoff and the next omitted magnitude, and it records the dyadic comparison that bounds the remainder by that displayed magnitude. The rows W and R expose the same finite observations as stream windows and regular rational readback.

Finally E, H, C, P, N seal, transport, replay, source, and name that route. Since no excluded object occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2159.3 (Leibniz alternating remainder tail bound)**.** *For every accepted* $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ *carrier*

$$L = (S, \Sigma, M, T, D, W, R, E, H, C, P, N),$$

each endpoint-facing remainder request factors through the finite route

$$S, \Sigma, M, T \longmapsto D, W \longmapsto R.$$

The dyadic row D *therefore binds the absolute remainder budget to the next omitted displayed magnitude read from* M*, and the regular readback* R *may be consumed only after the same cutoff, sign ledger, monotone null-tail row, and stream window have been displayed.*

Proof. Begin with an endpoint-facing remainder request. The carrier first exposes the real-series source S and the alternating-sign ledger Σ, so the request sees only displayed summands and their finite parity data.

Read the monotone null-tail row M and the alternating-test handoff T. These rows provide the displayed nonnegative magnitudes, the observed decrease comparisons, the null-tail request, and the test compatibility needed for the selected cutoff.

The dyadic ledger D then pairs the requested tolerance with that cutoff and the next omitted magnitude. Because the monotone tail and alternating sign have already been displayed, the classical Leibniz estimate is represented here as a finite BEDC comparison row rather than as an external convergence theorem.

The rows W and R expose the same observations as finite windows and regular rational readback. Thus the readback cannot bypass the source, sign, null-tail, test, or budget rows while still serving the remainder request. □

Theorem 2159.4 (Leibniz alternating remainder Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealAnalytic}^{\uparrow}$*-facing read from an accepted* $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ *carrier reaches its terminal seal only through*

$$S, \Sigma, M, T, D, W, R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports a finite remainder certificate for Real and Completion consumers, including the local Leibniz stream rows used by $\mathsf{RealAnalytic}^{\uparrow}$*, but it exports no selected infinite sum, quotient stream equality, host real equality, countable-choice selector, unrestricted trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or analytic endpoint detached from the displayed remainder route.*

Proof. Use Definition 2159.3 to place every remainder request behind the source, sign, monotone null-tail, alternating-test, dyadic-budget, window, and regular-readback rows.

The terminal row E is a $\mathsf{Real}^{\uparrow}$ seal for that already displayed route. It is not a selected value of an infinite alternating sum, and it does not assert host completeness for arbitrary real streams.

For a $\mathsf{RealAnalytic}^{\uparrow}$ consumer, the same boundary applies. The analytic chapter may read unary Leibniz partial-sum rows, but the completion-facing endpoint must still pass through the displayed remainder budget and regular readback before it reaches E.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the carrier has no coordinate for the excluded principles or for a detached analytic endpoint, none can escape through the seal boundary. □

Falsifiable prediction: Every $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ endpoint-facing read factors through a RealSeriesUp source, alternating-sign ledger, monotone NullSequenceUp tail row, AlternatingSeriesTestUp handoff, DyadicRatCoreUp remainder budget, StreamNameUp windows, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, alternating-series-test, nullsequence, streamname, regseqrat, dyadicratcore, real, real-analytic: $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the next-omitted-term remainder budget, alternating parity, monotone null-tail schedule, regular readback, Real sealing, analytic-stream consumer boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LeibnizAlternatingRemainder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeibnizAlternatingRemainder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2159.1, Definition 2159.2, Definition 2159.3, and Definition 2159.4 seed the finite Leibniz alternating-remainder certificate surface from displayed real-series partial sums and monotone null tails through dyadic next-omitted-term budgets to regular readback and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeibnizAlternatingRemainder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient stream equality, host real equality, countable choice, unrestricted trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, detached analytic endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-ledger exactness, monotone null-tail exactness, dyadic next-omitted-term budget stability, RegSeqRat readback, Real seal non-escape, analytic-stream boundary compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LeibnizAlternatingRemainderUp.

Constructive story: A $\mathsf{LeibnizAlternatingRemainder}^{\uparrow}$ packet is generated from a RealSeriesUp source, alternating-sign ledger, monotone NullSequenceUp tail row, AlternatingSeriesTestUp handoff, DyadicRatCoreUp remainder budget, StreamNameUp windows, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2160

A Concrete Naming Certificate for $\mathsf{PeanoContinuum}^{\uparrow}$

Following Hahn and Mazurkiewicz, the $\mathsf{PeanoContinuum}^{\uparrow}$ packet records a compact, connected, locally connected metric continuum as finite BEDC NameCert data. It sits over the compact-metric source of chapter 129, the chain-connected route of chapter 2050, the complete separable metric boundary of chapter 1735, the Polish-space readback of chapter 1490, and the continuous-map surface of chapter 59. The packet names the finite continuum interface needed by interval-image and compact-metric consumers without importing host point-set topology, quotient point equality, open-cover compactness, countable choice, or the full Hahn–Mazurkiewicz theorem.

Definition 2160.1 (Peano continuum carrier)**.** *A* $\mathsf{PeanoContinuum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, K, S, L, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *X is the compact-metric source row, including the displayed metric carrier, finite compactness data, total-boundedness ledger, and complete metric handoff read through chapter 129.*
2. *K is the connectedness row. It is either a directly displayed connectedness witness or a* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *route whose finite chain ledgers are admitted as the connectedness-facing evidence for the same metric source.*
3. *S is the complete-separable and Polish readback row. It records the common dense-window and completion boundary consumed by chapter 1735 and chapter 1490.*
4. *L is the finite local-connectedness refinement ledger. For each displayed compact-window request, it records a finite family of connected neighbourhood rows, inclusion refinements, overlap certificates, and radius comparisons over the source metric.*
5. *I is the interval and continuum readback row. It records the finite interval-source windows and continuum-facing observations that later consumers read before asking for an interval image or a compact continuum boundary.*
6. *R is the continuous readback row, exposing only the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus rows that connect the interval windows in I to the compact-metric source row X.*
7. *H records componentwise* hsame *transport for the compact-metric source, connectedness route, complete-separable and Polish readback, local-connectedness ledger, interval and continuum readback, continuous-map rows, provenance, and local naming row.*

8. *C records displayed* Cont *replay of the route*

$$X, K, S, L, I \longmapsto R \longmapsto H, Q, N.$$

9. *Q is the* Pkg *provenance over* PeanoContinuum$^{\uparrow}$, CompactMetric$^{\uparrow}$, ChainConnectedMetric$^{\uparrow}$, CompleteSeparableMe PolishSpace$^{\uparrow}$, ContinuousMap$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{PeanoContinuum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, S, L, I, R, H, C, Q, N$. The carrier has no coordinate for host point-set topology, open-cover compactness, quotient point equality, countable choice, a selected space-filling map, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PeanoContinuumUp.PeanoContinuumTasteGate single carrier alignment`

Theorem 2160.2 (Peano continuum NameCert obligations)**.** *For every accepted* PeanoContinuum$^{\uparrow}$ *carrier*

$$P = (X, K, S, L, I, R, H, C, Q, N),$$

the local NameCert$_{\text{PeanoContinuum}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact-metric source, connectedness route, complete-separable and Polish readback, local-connectedness ledger, interval and continuum readback, continuous readback, transport, replay, provenance, and local naming row;*

2. *compact-metric handoff, requiring every metric, compactness, finite-net, and completion-facing read to pass through the single source row X;*

3. *connectedness visibility, requiring K to expose either the displayed connectedness evidence or the finite* ChainConnectedMetric$^{\uparrow}$ *route before a continuum-facing consumer may read the packet;*

4. *local-connectedness refinement, requiring L to record finite connected-neighbourhood refinements, inclusion rows, overlap certificates, and radius comparisons for each displayed compact-window request;*

5. *continuous-image readback nonescape, requiring I and R to be displayed before any interval-image or continuum read is exported;*

6. *provenance exhaustion, requiring H, C, Q, N to transport, replay, source, and name exactly the rows listed above and no host point-set topology, open-cover compactness, quotient point equality, countable-choice selector, full Hahn–Mazurkiewicz theorem, Lean verification, or bridge credential.*

No Peano-continuum obligation is stored outside these rows.

Proof. Project the accepted carrier $P = (X, K, S, L, I, R, H, C, Q, N)$. The row list in Definition 2160.1 is the whole visible packet, so carrier admission is projection from the displayed finite BHist data.

The compact-metric handoff is forced by the source coordinate. Every metric, compactness, finite-net, and completion-facing request first reads X, and the complete-separable and Polish row S is only a readback of that same displayed source.

Connectedness is visible through K. If the packet uses direct connectedness evidence, the row is read as displayed data; if it uses chain-connectedness, the finite ChainConnectedMetric$^{\uparrow}$ route supplies the endpoint, chain, radius, and tolerance ledgers over the same source X.

Local connectedness is confined to L. Each refinement request is answered by a finite ledger of connected neighbourhood rows, inclusion refinements, overlap certificates, and radius comparisons, so no host open-set lattice or point-set topology is introduced.

The interval and continuous-image rows are downstream. A consumer must read I and then R before receiving an interval-image or continuum observation, and R exposes only the graph and modulus rows admitted by the continuous-map surface.

Finally H, C, Q, N transport, replay, source, and name the same rows componentwise. Since the refused principles have no carrier coordinate and no replay edge, the listed obligations exhaust the local NameCert surface. □

Theorem 2160.3 (Peano continuum locally connected handoff)**.** *Every continuum-facing consumer read of an accepted* $\mathsf{PeanoContinuum}^{\uparrow}$ *carrier factors through the finite route*

$$X \longmapsto K \longmapsto S \longmapsto L \longmapsto I \longmapsto R \longmapsto H, C, Q, N.$$

Thus the consumer receives the compact-metric row, connectedness row, finite local-connectedness refinement ledger, interval and continuum readback, continuous readback, componentwise transport, continuation replay, package provenance, and local naming. It receives no host point-set topology, open-cover compactness, quotient point equality, countable choice, selected space-filling map, full Hahn–Mazurkiewicz theorem, Lean verification, or bridge checking.

Proof. Start with a continuum-facing read of the accepted packet. The compact-metric source row X is the first admissible row because every later interval, local-connectedness, and completion-facing observation is measured over that source.

The connectedness and complete-separable boundaries are then fixed. The row K supplies the displayed connectedness or chain-connectedness route, while S records the complete-separable and Polish readback for the same metric data rather than for a second host space.

Next read the local-connectedness ledger L. Its finite refinement rows give the connected neighbourhood, inclusion, overlap, and radius-comparison data required by the consumer before the interval readback is exposed.

The interval and continuous readback rows complete the functional handoff. The row I supplies the interval and continuum observations, and R supplies the continuous-map graph and modulus rows through which those observations are replayed.

The structural rows close the route. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming preserve exactly this dependency order, so every consumer read factors through the compact-metric row, connectedness row, finite local-connectedness ledger, continuous readback, transport, replay, provenance, and local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeanoContinuum}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PeanoContinuum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2160.1, Definition 2160.2, and Definition 2160.3 seed the finite compact, connected, locally connected metric continuum packet over compact-metric, chain-connected, complete-separable, Polish, continuous-map, local-connectedness, interval-readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PeanoContinuum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host point-set topology, open-cover compactness, quotient point equality, countable choice, selected space-filling maps, a full Hahn–Mazurkiewicz theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric handoff, connectedness visibility, local-connectedness refinement, continuous-image readback nonescape, provenance exhaustion, and locally connected handoff for BEDC.Derived.PeanoContinuumUp.

Constructive story: A $\mathsf{PeanoContinuum}^{\uparrow}$ packet is generated from a compact-metric source row, connectedness or ChainConnectedMetric evidence, CompleteSeparableMetric and PolishSpace readback, a finite local-connectedness

refinement ledger, interval and continuum readback rows, ContinuousMap graph and modulus rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2161

A Concrete Naming Certificate for CodonWindowSpectralInverse$^{\uparrow}$

The CodonWindowSpectralInverse$^{\uparrow}$ packet realizes the codon-window spectral inverse vision of subsection 9462.28.22 as finite BEDC NameCert data. It sits beside the stop-face suffix packet of chapter 1374, the finite bit-path carrier of chapter 346, the zero-spine arithmetic route of chapter 277, and the window-six dynamics bridge of subsection 9462.28.20. Its role is a finite inverse packet: it records the codon cube, two-axis atlas, local face spectra, stop-face reconstruction row, adjacency spectra, reassignment spectra, transport, replay, provenance, and local naming without identifying the biological code table with a BEDC kernel primitive.

Definition 2161.1 (Codon-window spectral inverse carrier)**.** *A* CodonWindowSpectralInverse$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, A, F, H, R, T, Q, E, K, D, P, N)$$

with the following displayed rows:

1. *C is the codon-cube row. It records the finite Q_6 carrier, the three two-bit base windows, and the displayed* BitVector$^{\uparrow}$ *paths used to read a codon window.*
2. *A is the two-axis atlas row. It records the* $\mathrm{Y/R}$ *and* $\mathrm{W/S}$ *readback for* $\mathrm{U, C, A, G}$ *and keeps RNA/DNA spelling as bridge data rather than as a kernel equality.*
3. *F is the local face enumeration row. It records the finite Q_2 faces, the selected stop/trp face, and the face labels consumed by the inverse spectrum.*
4. *H is the local face spectrum row. It records multiplicity and class-pattern rows for each displayed face named by F.*
5. *R is the Hamming adjacency spectrum row. It records one-bit adjacency boundaries, class changes, and local control-tile comparisons over the same Q_6 carrier.*
6. *T is the reassignment spectrum row. It records finite comparison rows across displayed biological bridge tables or controlled perturbation rows.*
7. *Q is the stop-face row. It records the* $\mathrm{U+RR}$ *square, the family-suffix coordinates* $00, 01, 10, 11$, *and the retained $X_2 = \{00, 01, 10\}$ stop candidates.*
8. *E records componentwise* hsame *transport for codon-cube, atlas, face, spectrum, adjacency, reassignment, stop-face, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay from a spectral inverse request through the cube, atlas, face enumeration, local spectra, adjacency spectra, reassignment spectra, stop-face row, provenance, and local name.*

10. *D is the inverse-decision ledger. It records which finite rows are used to constrain a local coding mechanism and refuses any hidden global code-table reconstruction row.*

11. *P is the* Pkg *provenance over* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$, $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$, $\mathsf{BitVector}^{\uparrow}$, $\mathsf{AxisAdd}^{\uparrow}$, *the codon-window-six and DNA-window-six bridge rows,* BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CodonWindowSpectralInverse}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, A, F, H, R, T, Q, E, K, D, P, N$. It has no coordinate for a host genetic-code table, a selected global inverse, biological universality, quotient codon equality, probabilistic inference, or a choice principle.

Lean4 ✓ `BEDC.Derived.CodonWindowSpectralInverseUp.TasteGate.CodonWindowSpectralInverseTasteGate single carrier alignment`

Theorem 2161.2 (Codon-window spectral inverse NameCert obligations)**.** *For every accepted* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *carrier*

$$S = (C, A, F, H, R, T, Q, E, K, D, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CodonWindowSpectralInverse}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the codon cube, two-axis atlas, local face enumeration, local face spectra, Hamming adjacency spectra, reassignment spectra, stop-face row, transport, replay, inverse-decision ledger, provenance, and local naming row;*

2. *cube exactness, requiring every codon-window read to pass through the finite Q_6 row C and its displayed* $\mathsf{BitVector}^{\uparrow}$ *coordinates;*

3. *atlas compatibility, requiring all biological spelling reads to pass through A before any face or spectrum row is consumed;*

4. *local spectrum coverage, requiring F, H, R, T to expose the finite face, multiplicity, adjacency, and reassignment rows used by the inverse packet;*

5. *stop-face reconstruction, requiring Q to display the* $\mathrm{U}+\mathrm{RR}$ *face and the retained X_2 suffix rows before a stop-candidate read is exported;*

6. *transport and replay stability, requiring E and K to move only the displayed rows and preserve the same carrier inventory under* hsame *and* Cont *replay;*

7. *provenance and non-escape, requiring D, P, N to record the consumed sibling packets and to exclude host genetic-code tables, selected global inverses, biological universality, quotient codon equality, probabilistic inference, and choice.*

No codon-window spectral inverse obligation is stored outside these rows.

Proof. Project the accepted carrier $S = (C, A, F, H, R, T, Q, E, K, D, P, N)$. The tuple inventory is the whole visible packet, so carrier admission is projection from the displayed finite BHist data.

Cube exactness is forced by C. Every codon-window request reads the finite Q_6 carrier and its three two-bit $\mathsf{BitVector}^{\uparrow}$ windows before any biological or spectral classifier row is available.

Atlas compatibility is confined to A. The RNA/DNA spelling and the two-axis readback are bridge rows over the finite cube, not kernel equalities and not a host code table.

The local spectrum rows are F, H, R, T. They enumerate the finite faces, attach local face spectra, record one-bit adjacency boundaries, and compare reassignment rows, so the inverse packet can constrain local mechanisms without selecting a global table.

The stop-face row Q supplies the finite $\mathrm{U} + \mathrm{RR}$ square and the retained X_2 suffix rows. Transport through E and replay through K preserve these rows componentwise, while D, P, N record the consumed route and local name. □

Theorem 2161.3 (Codon-window stop-face reconstruction route)**.** *Let $S = (C, A, F, H, R, T, Q, E, K, D, P, N)$ be an accepted* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *carrier. Every stop-face reconstruction read exported by S factors through the finite route*

$$C, A, F, H, R, T \longmapsto Q \longmapsto E, K, D, P, N.$$

The exported read receives the codon cube, two-axis atlas, local face enumeration, local face spectra, Hamming adjacency spectra, reassignment spectra, stop-face row, componentwise transport, replay, inverse-decision ledger, provenance, and local name. It receives no host genetic-code table, selected global inverse, biological universality claim, quotient codon equality, probabilistic inference row, or choice principle.

Proof. Start with the accepted packet and read C and A. These rows fix the finite codon carrier and the two-axis spelling bridge before any spectral data are visible.

Next read F, H, R, T. The local faces, local spectra, Hamming adjacency boundaries, and reassignment comparisons are all finite rows over the same displayed cube, so no global code-table search is introduced.

The stop-face row Q then identifies the $\mathrm{U} + \mathrm{RR}$ square and the X_2 suffix subset. This is the only row through which a stop-face reconstruction may pass.

Finally the structural rows E, K, D, P, N transport, replay, audit, source, and name the same route. Since the refused principles have no carrier coordinate and no replay edge, the stop-face reconstruction read exports only the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ stop-face or local inverse read factors through one finite codon-cube row, one two-axis atlas row, finite local face spectra, Hamming adjacency spectra, reassignment spectra, one stop-face row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Empirical comparison may reject a chosen biological bridge by showing that its local face spectra, adjacency spectra, or reassignment spectra do not match observed genetic-code table rows.

Independence witness: stopcodonzeckendorfsuffix, bitvector, axisadd, codon-window-six-bridge, dna-window-six-dynamics-bridge: $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the codon cube, two-axis atlas, local face enumeration, local face spectra, Hamming adjacency spectra, reassignment spectra, stop-face reconstruction row, inverse-decision ledger, transport, replay, provenance, and local naming.

The Lean follow-up file `lean4/BEDC/Derived/CodonWindowSpectralInverseUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{CodonWindowSpectralInverse}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{CodonWindowSpectralInverse}^{\uparrow}$ instances, with field-projection proofs preserving $C, A, F, H, R, T, Q, E, K, D, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{CodonWindowSpectralInverse}^{\uparrow}}$.

Constructive story: A CodonWindowSpectralInverse$^{\uparrow}$ packet is generated from a codon-cube row, two-axis atlas row, local face enumeration, local face spectra, Hamming adjacency spectra, reassignment spectra, stop-face row, componentwise hsame transport, displayed Cont replay, inverse-decision ledger, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CodonWindowSpectralInverse}^{\uparrow}}$, seed)

Scope closed: Definition 2161.1, Definition 2161.2, and Definition 2161.3 seed the finite codon-window spectral inverse packet over codon cube, atlas, face spectra, adjacency spectra, reassignment spectra, stop-face, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CodonWindowSpectralInverse}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host genetic-code tables, selected global inverses, biological universality, quotient codon equality, probabilistic inference, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cube exactness, atlas compatibility, local spectrum coverage, stop-face reconstruction, transport and replay stability, provenance non-escape, Nontrivial CodonWindowSpectralInverseUp, and FieldFaithful CodonWindowSpectralInverseUp for BEDC.Derived.CodonWindowSpectralInverseUp.

Chapter 2162

A Concrete Naming Certificate for $\mathsf{RealApproximationScheme}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealApproximationScheme}^{\uparrow}$ packet records a constructive real approximation interface as finite BEDC NameCert data. It sits between the stream observations of chapter 126, the dyadic tolerance rows of chapter 291, the regular-sequence readback of chapter 301, and the sealed real boundary of chapter 38. The packet names the approximation route needed by later Real, metric, and continuous-map certificates without importing host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Definition 2162.1 (Real approximation scheme carrier)**.** *A* $\mathsf{RealApproximationScheme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It records the finite schedule of requested observations, the visible stream-name windows, and the order in which a consumer may read them.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic radii, comparison cells, shrink steps, and budget rows attached to each displayed observation request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational sequence data that consumes S and D before any real-facing seal is exposed.*
4. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records only the finite real-boundary interface reached through the regular readback, not a host real value.*
5. *H records componentwise* hsame *transport for the stream observations, dyadic tolerance rows, regular readback, real seal row, continuation route, package provenance, and local naming row.*
6. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto R \longmapsto Q \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{RealApproximationScheme}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApproximationScheme}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, D, R, Q, H, C, P, N. The carrier has no coordinate for host real equality, quotient streams, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RealApproximationSchemeUp.RealApproximationSchemeTasteGate single carrier alignment`

Theorem 2162.2 (Real approximation scheme NameCert obligations)**.** *For every accepted* RealApproximationScher *carrier*

$$A = (S, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealApproximationScheme}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream observations, dyadic tolerance ledger, regular readback, real seal boundary, transport, replay, provenance, and local naming row;*
2. *stream observability, requiring every approximation request to enter through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in S;*
3. *dyadic tolerance exactness, requiring every precision demand to be witnessed by a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, shrink, comparison, or budget row in D;*
4. *regular readback routing, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume the stream and dyadic rows before a real-facing consumer may read the packet;*
5. *real seal nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *seal row Q to be reached only through R and never through a host real representative;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the rows listed above and no host real equality, quotient stream, selected limit, trichotomy, ambient completeness, Lean verification, or bridge credential.*

No real-approximation-scheme obligation is stored outside these rows.

Proof. Project the accepted carrier $A = (S, D, R, Q, H, C, P, N)$. The definition lists the whole visible packet, so carrier admission is just the displayed finite BHist data together with its transport, replay, provenance, and naming rows.

The observation side is forced by S. Every approximation request first reads the $\mathsf{StreamName}^{\uparrow}$ windows and schedule before any tolerance or readback row can be consumed.

The precision side is forced by D. The dyadic radii, shrink steps, comparison cells, and budget rows are the only tolerance coordinates available to the packet, so a consumer cannot replace them by an ambient metric or a host real estimate.

The real-facing route passes through R and then Q. The $\mathsf{RegSeqRat}^{\uparrow}$ row reads the stream and dyadic data as a regular rational sequence, and the $\mathsf{Real}^{\uparrow}$ seal row exposes only the finite boundary supplied by that readback.

Finally H, C, P, N transport, replay, source, and name the same rows componentwise. Since the refused principles have no coordinate and no replay edge, the listed obligations exhaust the local NameCert surface. □

Theorem 2162.3 (Real approximation scheme RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{RealApproximationScheme}^{\uparrow}$ *carrier factors through the finite route*

$$S, D \longmapsto R \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer receives the $\mathsf{StreamName}^{\uparrow}$ *observation windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. It receives no host real equality, quotient stream, selected limit, trichotomy, ambient completeness, Lean verification, or bridge checking.*

Proof. Start with a consumer read of the accepted packet. The stream observation row S supplies the finite request schedule and visible windows, so the read begins with a named $\mathsf{StreamName}^{\uparrow}$ surface rather than an unbounded stream.

Next read the dyadic tolerance row D. The row provides the $\mathsf{DyadicRatCore}^{\uparrow}$ radii, shrink steps, comparison cells, and budget entries attached to the same observations, so the read has a finite precision ledger before it reaches a regular sequence.

The handoff is then the row R. It packages the observed stream and tolerance data as the $\mathsf{RegSeqRat}^{\uparrow}$ readback consumed by later real-facing certificates.

The real boundary is last. The row Q records only the $\mathsf{Real}^{\uparrow}$ seal reached through R, and the structural rows H, C, P, N preserve exactly this route by componentwise transport, continuation replay, provenance, and local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApproximationScheme}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApproximationScheme}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2162.1, Definition 2162.2, and Definition 2162.3 seed the finite approximation packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApproximationScheme}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream observability, dyadic tolerance exactness, regular readback routing, real seal nonescape, and provenance exhaustion for BEDC.Derived.RealApproximationSchemeUp.

Constructive story: A $\mathsf{RealApproximationScheme}^{\uparrow}$ packet is generated from StreamName observation windows, DyadicRatCore tolerance rows, RegSeqRat readback, a Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2163

A Concrete Naming Certificate for $\mathsf{SqueezeTheorem}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SqueezeTheorem}^{\uparrow}$ packet records the constructive comparison sandwich as a finite lower-middle-upper $\mathsf{Real}^{\uparrow}$ comparison packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. It sits after the real seal of chapter 38, the regular rational rows of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the displayed sequence-limit route of chapter 1413. The packet gives L10 Real and Completion consumers one named sandwich surface without importing host limits, quotient streams, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2163.1 (Squeeze theorem carrier)**.** *A* $\mathsf{SqueezeTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, M, U, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L, M, and U are the lower, middle, and upper source rows. Each source is read only through the same displayed sequence-limit and real-comparison route before any conclusion is exported.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the inspected tail indices, lower-middle comparison windows, middle-upper comparison windows, and limit-window requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the split tolerance used to compare the lower and upper bounds with the common limit row and the middle row with the displayed sandwich windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled lower, middle, upper, and limit observations.*
5. *E is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *and terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It becomes available only after L, M, U, W, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the three source rows, window row, dyadic ledger, rational readback row, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay of the finite route*

$$L, M, U, W, D, Q \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{SqueezeTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{SqueezeTheorem}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $L, M, U, W, D, Q, E, H, C, P, N$ and has no coordinate for host sequence limits, quotient stream equality, trichotomy, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2163.2 (Squeeze theorem NameCert obligations)**.** *For every accepted* $\mathsf{SqueezeTheorem}^{\uparrow}$ *carrier*

$$S = (L, M, U, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SqueezeTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower, middle, and upper source rows, finite window row, dyadic tolerance ledger, regular-rational readback row, displayed sequence-limit and real-seal row, transport, replay, provenance, and local naming row;*
2. *sandwich-window discipline, requiring every lower-middle and middle-upper comparison to be read through the same scheduled W row before the middle limit can be exported;*
3. *dyadic tolerance exactness, requiring D to record the tolerance split consumed by the two bounding convergence reads and the middle comparison read;*
4. *rational readback discipline, requiring Q to expose the scheduled* $\mathsf{RegSeqRat}^{\uparrow}$ *observations for the lower, middle, upper, and limit rows before the terminal seal E is available;*
5. *real-seal nonescape, requiring E to remain downstream of the displayed sequence-limit route and forbidding direct host-limit or completeness reads;*
6. *transport, replay, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise;*
7. *ledger refusal, excluding host sequence limits, quotient stream equality, trichotomy, countable choice, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No squeeze theorem obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SqueezeTheoremUp.SqueezeTheoremCarrier namecert obligations`

Proof. Project the accepted packet $S = (L, M, U, W, D, Q, E, H, C, P, N)$. The carrier exposes the three source rows, one shared finite-window row, one dyadic tolerance ledger, one regular-rational readback row, one terminal sequence-limit and real-seal row, and the structural rows.

The comparison route starts with the common window. The row W fixes which lower-middle, middle-upper, and limit windows are inspected, so the middle source cannot use an unrelated comparison schedule.

The dyadic and rational data are then forced. The row D supplies the tolerance split, and Q supplies the scheduled $\mathsf{RegSeqRat}^{\uparrow}$ observations attached to that split before any seal is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite rows. Since none of the excluded host objects has a carrier coordinate, replay edge, provenance entry, or local naming row, the listed obligations exhaust the local NameCert surface. □

Theorem 2163.3 (Squeeze theorem comparison sandwich)**.** *Every accepted* $\mathsf{SqueezeTheorem}^{\uparrow}$ *carrier exposes the middle real-sequence limit only through the finite route*

$$L, M, U, W, D, Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed lower-middle-upper comparison sandwich, a shared $\mathsf{StreamName}^{\uparrow}$ *tail-window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise transport, continuation replay, package provenance, and local naming. It receives no host sequence limit, quotient stream equality, trichotomy, countable choice, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The comparison sandwich consumes the regular-name enclosure of chapter 1772 and the dyadic lower/upper tolerance route of chapter 3627 before the $\mathsf{RealSequenceLimit}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows are exposed.

Proof. Start with an accepted squeeze-theorem read. The source rows are L, M, and U, and the carrier requires all comparison evidence to be observed through the shared finite-window row W.

The dyadic ledger D fixes the tolerance split for the bounding convergence reads and the middle comparison read. The regular-rational row Q then exposes the scheduled lower, middle, upper, and limit observations before the real seal can be reached.

Only after those rows have been displayed can E expose the $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff and terminal $\mathsf{Real}^{\uparrow}$ seal boundary. The structural rows H, C, P, N preserve that same finite route under componentwise transport, continuation replay, provenance, and local naming.

Induction over the displayed route order proves that no later consumer can skip a row: the base stage reads L, M, U, W, D, Q, the terminal step reads E, and the structural step reads H, C, P, N. Because there is no coordinate outside this route, the consumer cannot obtain host sequence limits, quotient stream equality, trichotomy, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the packet. □

Theorem 2163.4 (Squeeze theorem real-seal nonescape)**.** *An accepted* $\mathsf{SqueezeTheorem}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the lower row, middle row, upper row, shared StreamName window row, dyadic tolerance ledger, RegSeqRat readback row, and RealSequenceLimit handoff. Consequently the real seal cannot be used as a shortcut to host limit existence, quotient stream equality, trichotomy, countable choice, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row E from an accepted carrier. By carrier admission, E is not an independent coordinate; it is downstream of L, M, U, W, D, Q and records only the sequence-limit and real-seal boundary reached from those displayed rows.

The replay row C has no branch that starts at E. It replays the route from the three source rows through the window schedule, tolerance ledger, and rational readback before the seal can be consumed.

The transport and provenance rows H and P preserve the same dependency order. The local name N therefore names only the displayed sandwich packet, so no host limit, quotient stream, trichotomy, countable-choice selector, selected representative, or ambient completeness witness can escape through the seal. □

Theorem 2163.5 (Squeeze theorem Real seal handoff)**.** *A* $\mathsf{SqueezeTheorem}^{\uparrow}$ *consumer reaches the middle-row* $\mathsf{Real}^{\uparrow}$ *seal only by first displaying the lower and upper* $\mathsf{RealSequenceLimit}^{\uparrow}$ *rows, the middle sequence row, the shared* $\mathsf{StreamName}^{\uparrow}$ *tail windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff. The resulting route is*

$$L, M, U, W, D, Q \longmapsto E \longmapsto H, C, P, N.$$

It refuses host limits, quotient streams, trichotomy, countable choice, and ambient Real completeness.

Proof. Begin with the comparison sandwich of Definition 2163.3. It forces the lower, middle, and upper rows to pass through one shared finite route L, M, U, W, D, Q before any seal row is available.

Apply the real-seal nonescape theorem Definition 2163.4. The terminal seal E cannot be reached until the shared tail windows, dyadic tolerance ledger, regular rational readbacks, and displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff have all been displayed.

The structural rows H, C, P, N then transport, replay, source, and name the same route. Since neither theorem exposes host limits, quotient streams, trichotomy, countable choice, or ambient Real completeness, the handoff to the middle-row $\mathsf{Real}^{\uparrow}$ seal exports none of those objects. □

Theorem 2163.6 (Squeeze theorem zero-envelope route)**.** *If an accepted* $\mathsf{SqueezeTheorem}^{\uparrow}$ *carrier has lower and upper source rows both read through the displayed zero* $\mathsf{RealSequenceLimit}^{\uparrow}$ *envelope, then every middle-row* $\mathsf{Real}^{\uparrow}$ *seal still factors through* W, D, Q, E, H, C, P, N*. It cannot be read as host zero-limit equality, quotient stream equality, or ambient completeness.*

Proof. Project the accepted carrier $S = (L, M, U, W, D, Q, E, H, C, P, N)$. The lower and upper rows share the same displayed zero envelope, but the carrier still exposes the middle row only through the comparison data and terminal seal rows.

Specialize the comparison sandwich of Definition 2163.3 to the case where L and U read the same zero $\mathsf{RealSequenceLimit}^{\uparrow}$ envelope. The shared envelope fixes the bounding route; it does not remove the finite window row W, dyadic ledger D, or regular-rational readback Q.

Apply the real-seal nonescape theorem Definition 2163.4. The middle-row seal reaches $\mathsf{Real}^{\uparrow}$ only after the displayed tolerance and readback route has been consumed.

Thus the zero-envelope squeeze route is certified by finite dyadic tolerance accounting and $\mathsf{RegSeqRat}^{\uparrow}$ readback. It is not a host zero-limit equality, a quotient stream equality, or an appeal to ambient completeness. □

Theorem 2163.7 (Squeeze theorem Cauchy-real field route)**.** *Let* $S = (L, M, U, W, D, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{SqueezeTheorem}^{\uparrow}$ *zero-envelope carrier whose lower, middle, and upper rows are read through the* $\mathsf{CauchyRealField}^{\uparrow}$ *surface. Then the middle-row* $\mathsf{Real}^{\uparrow}$ *seal is reached only after the common* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyRealField}^{\uparrow}$*, order, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows are displayed. The route does not claim* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, or host limits.*

Proof. Begin with the zero-envelope route of Definition 2163.6. It already forces the middle-row seal to pass through the shared window row W, dyadic ledger D, regular readback row Q, terminal seal row E, and structural rows H, C, P, N.

Read the lower, middle, and upper rows through the $\mathsf{CauchyRealField}^{\uparrow}$ chapter chapter 1717. That surface supplies the regular-Cauchy addition, product, reciprocal, and order rows as displayed field-facing data before a real-facing consumer can inspect the result.

Finally pass through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The seal is downstream of the displayed regular readback, stream windows, dyadic tolerance accounting, Cauchy-real field rows, order rows, transport, replay, provenance, and local naming.

Because the proof composes only those displayed chapter surfaces, it adds no Real completeness principle, no trichotomy split, and no host limit object. □

Closure status for $\mathsf{NameCert}_{\mathsf{SqueezeTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SqueezeTheorem}^{\uparrow}$ packet is generated from lower, middle, and upper source rows, one shared StreamName tail-window schedule, a DyadicRatCore tolerance ledger, RegSeqRat readbacks, a RealSequenceLimit handoff, a terminal RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SqueezeTheorem}\uparrow}$, seed)

Scope closed: Definition 2163.1, Definition 2163.2, Definition 2163.3, and Definition 2163.4 seed the finite constructive comparison sandwich over StreamName, RegSeqRat, DyadicRatCore, RealSequenceLimit, and RealUp rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SqueezeTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host sequence limits, quotient stream equality, trichotomy, countable choice, selected representatives, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, sandwich-window discipline, dyadic tolerance exactness, rational readback discipline, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SqueezeTheoremUp.

Chapter 2164

A Concrete Naming Certificate for $\mathsf{FrechetCompletion}^{\uparrow}$

Following Frechet on metric convergence, Bourbaki and Kelley on filters and nets, and Bishop–Bridges on constructive completion, the $\mathsf{FrechetCompletion}^{\uparrow}$ packet records the L10 filter-completion surface as finite BEDC NameCert data. It consumes the completion source of chapter 1506, the filter route of chapter 333, the net route of chapter 589, and the separated-limit boundary of chapter 1612. It also reads finite stream windows from chapter 126, regular rational readbacks from chapter 301, and terminal real seals from chapter 38. The carrier names only finite filter, net, sequential, separated, transport, replay, provenance, and local naming rows; it does not import quotient completions, representative choice, countable choice, a host complete space, or a global convergence selector.

Definition 2164.1 (FrechetCompletion carrier)**.** *A* $\mathsf{FrechetCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, L, N, S, Q, R, H, C, P, K)$$

with the following displayed rows:

1. *A is the* $\mathsf{CauchyFilter}^{\uparrow}$ *admission row, recording the finite filter-base observations and Cauchy refinement checks visible to the packet.*
2. *L is the* $\mathsf{CauchyNet}^{\uparrow}$ *handoff row, displaying the directed finite net observations that read the same completion route without selecting a cofinal subnet.*
3. *N is the* $\mathsf{MetricCompletion}^{\uparrow}$ *neighbour row, exposing only the local completion-facing boundary consumed by later L10 routes.*
4. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *separation row, recording the finite equality and apartness boundary used to prevent two completion reads from being merged by an undisplayed quotient.*
5. *Q is the paired* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *sequential readback row, carrying finite windows and regular rational approximants for consumers that ask for a sequence-like presentation.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed distance and approximation reads.*
7. *H records componentwise* hsame *transport for the Cauchy filter, Cauchy net, metric-completion neighbour, separated-completion boundary, stream window, regular rational readback, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$A, L, N, S, Q, R \longmapsto H, P, K.$$

9. *P is the* Pkg *provenance over* $\mathsf{FrechetCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *K is the local* $\mathsf{NameCert}_{\mathsf{FrechetCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, N, S, Q, R, H, C, P, K$. It has no coordinate for quotient completion equality, a chosen representative, a countable-choice schedule, a host complete metric space, a selected cofinal subnet, or an ambient convergence oracle.

Lean4 ✓ `BEDC.Derived.FrechetCompletionUp.FrechetCompletionTasteGate single carrier alignment`

Theorem 2164.2 (FrechetCompletion NameCert obligations)**.** *For every accepted* $\mathsf{FrechetCompletion}^{\uparrow}$ *carrier*

$$F = (A, L, N, S, Q, R, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{FrechetCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row,* $\mathsf{CauchyNet}^{\uparrow}$ *row,* $\mathsf{MetricCompletion}^{\uparrow}$ *neighbour row,* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *sequential readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public completion read must pass through those rows in the order displayed by C.*

Proof. Project the accepted carrier in Definition 2164.1. The visible mathematical rows are the Cauchy filter row A, Cauchy net row L, local metric-completion neighbour row N, separated-completion boundary row S, sequential readback row Q, and real seal row R.

The replay row C orders these rows before transport and naming are read. Thus a public completion-facing consumer first sees the filter and net routes, then the local completion and separation boundaries, and only then the sequential and real readbacks.

The rows H, P, K transport, source, and name exactly this surface. Since the carrier has no coordinate for quotient completion equality, representative choice, countable choice, a selected cofinal subnet, or a host complete space, the listed rows exhaust the local NameCert obligations. □

Theorem 2164.3 (FrechetCompletion filter handoff)**.** *Let $F = (A, L, N, S, Q, R, H, C, P, K)$ be an accepted* $\mathsf{FrechetCompletion}^{\uparrow}$ *carrier. Every downstream completion consumer that enters through the filter side receives a finite handoff through*

$$A \longmapsto L \longmapsto N \longmapsto S \longmapsto Q \longmapsto R \longmapsto H, C, P, K.$$

The handoff exposes the Cauchy filter row, the matching Cauchy net row, the metric-completion neighbour row, the separated-completion boundary, the sequential readback, the real seal, transport, replay, provenance, and local naming, and it exports no quotient completion, no selected representative, and no choice principle.

Proof. Start with a completion consumer entering through A. The carrier makes A a displayed Cauchy-filter row, so the consumer cannot obtain completion data before reading the finite filter-base observations and Cauchy refinement checks.

The net and local completion rows then fix the same route. The row L records the visible net handoff, and N exposes only the local $\mathsf{MetricCompletion}^{\uparrow}$ neighbour boundary rather than a completed host point.

Separation and sequential readback are downstream checks. The row S prevents the route from being collapsed by an undisplayed quotient, while Q and R expose finite stream windows, regular rational approximants, and real seals for consumers that need a sequence-like read.

The structural rows close the handoff. Componentwise transport, displayed replay, provenance, and local naming preserve exactly the route shown above, so no quotient completion, representative selector, or countable-choice coordinate is available. □

Theorem 2164.4 (FrechetCompletion sequential readback)**.** *Every sequence-like read of an accepted* $\mathsf{FrechetCompletion}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows after the filter, net, metric-completion, and separated-completion rows have been read. Consequently the sequential presentation is a readback of the finite completion route, not a replacement by a host Cauchy sequence quotient or a selected limit.*

Proof. Inspect the accepted carrier. The only sequence-facing coordinate is Q, and it is declared as the paired $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ readback row. The only terminal value-facing coordinate is the $\mathsf{Real}^{\uparrow}$ seal row R.

The replay row C places Q and R after the Cauchy filter, Cauchy net, metric-completion neighbour, and separated-completion boundary rows. Thus a sequence-like consumer reads finite windows and rational approximants only as a downstream readback of the displayed completion route.

Because no coordinate stores a host Cauchy sequence quotient, selected limit, representative equality, or ambient completeness theorem, the sequential presentation cannot replace the finite route named by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2164.1, Definition 2164.2, Definition 2164.3, and Definition 2164.4 seed the finite filter-completion packet over Cauchy filter, Cauchy net, metric-completion, separated-completion, sequential readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, representative choice, countable choice, selected cofinal subnets, host complete spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter handoff, net handoff, metric-completion neighbour routing, separated-completion boundary routing, sequential readback, real seal discipline, transport stability, provenance, and local naming for BEDC.Derived.FrechetCompletionUp.

Constructive story: A $\mathsf{FrechetCompletion}^{\uparrow}$ packet is generated from finite CauchyFilter, CauchyNet, MetricCompletion, SeparatedCompletion, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 2165

A Concrete Naming Certificate for $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$

The $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ packet realizes the selector-ledger branch of chapter 9399 as finite NameCert data. It refines the one-step boundary selector of chapter 685, the finite selector ledger of chapter 494, the observer-history identity surface of chapter 479, and the inscription event boundary of chapter 519. Its carrier stores the current history, selected mark, displayed Ext route, selector witness, transport, replay, provenance, and local naming in one packet so that an empty-boundary branch is generated by a visible row rather than by a hidden chooser.

Definition 2165.1 (EmptyBoundarySelectorLedger carrier)**.** *An* $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (H, M, R, X, S, L, T, C, P, N)$$

with the following displayed rows:

1. *H is the current* Hist *row whose empty boundary is being inspected.*
2. *M is the selected-mark row, exposing the mark enacted at that boundary.*
3. *R is the resulting* Hist *row after the selected mark has been enacted.*
4. *X is the displayed* $\mathsf{Ext}(H, M, R)$ *route row.*
5. *S is the selector-witness row, recording that M is the enacted local branch for this packet and not a value returned by an undisplayed choice function.*
6. *L is the non-reset ledger row. It records the finite selector ledger continuing from H through M to R, preserving prior history instead of returning to* e*.*
7. *T records componentwise* hsame *transport for H, M, R, X, S, L and for the local naming row.*
8. *C records displayed* Cont *replay of the route*

$$H, M, R, X, S, L \longmapsto T, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$, $\mathsf{EmptyBoundarySelector}^{\uparrow}$, $\mathsf{SelectorLedger}^{\uparrow}$, $\mathsf{ObserverHistIdentity}^{\uparrow}$, $\mathsf{InscriptionEvent}^{\uparrow}$, BHist, Ext, hsame, Cont, *and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, M, R, X, S, L, T, C, P, N$. It has no coordinate for observer substance, global branch choice, a reset-to-Empty state, host trace equality, an unrecorded inscription source, or a truth oracle.

Lean4 ✓ `BEDC.Derived.EmptyBoundarySelectorLedgerUp.EmptyBoundarySelectorLedgerTasteGate single carrier alignment`

Theorem 2165.2 (EmptyBoundarySelectorLedger NameCert obligations)**.** *For every accepted* $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ *carrier*

$$E = (H, M, R, X, S, L, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the current* Hist *row, selected-mark row, resulting* Hist *row, displayed* Ext *route, selector witness, non-reset ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every branch read must expose both the selected mark M and the route row X before the ledger row L is consumed.*

Proof. Project the accepted carrier in Definition 2165.1. The current history H, selected mark M, resulting history R, extension route X, selector witness S, and ledger L are the complete branch-facing inventory.

The selected mark and extension route are read before the ledger. Thus a consumer cannot treat L as an unexplained sequence of choices; it receives the enacted local mark and the displayed Ext route that produced the next history.

The rows T, C, P, N transport, replay, source, and name exactly these displayed rows. Since the carrier has no coordinate for observer substance, global choice, reset-to-Empty semantics, host trace equality, or truth oracles, the listed rows exhaust the NameCert obligations. □

Theorem 2165.3 (EmptyBoundarySelectorLedger branch separation)**.** *Let $E = (H, M, R, X, S, L, T, C, P, N)$ and $E' = (H, M', R', X', S', L', T', C', P', N')$ be accepted* $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ *carriers over the same current history row. If the selected-mark rows M and M' are not classified together by the displayed transport rows, then the packets cannot be identified by their ledger rows alone. The branch distinction must pass through the selected mark,* Ext *route, selector witness, and non-reset ledger rows.*

Proof. Begin with the two accepted packets. Each exposes the current history row, selected mark, result row, extension route, selector witness, ledger row, transport, replay, provenance, and local name.

If M and M' are not classified together by the displayed transport rows, then the branch-facing coordinates already differ before the ledger row is read. The route rows X and X' and selector witnesses S and S' are attached to those selected marks.

Reading only L and L' would erase the branch evidence that the carrier explicitly stores. The classifier therefore preserves the distinction through M, X, S, L and their transported routes, rather than collapsing the packets to a hidden ledger equality. □

Theorem 2165.4 (EmptyBoundarySelectorLedger non-reset replay)**.** *Every public replay of an accepted* $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ *packet factors through the current history row H before it reaches the selected mark M, displayed extension route X, selector witness S, and ledger row L. Consequently the packet records continuation from the current history, not a reset to* e *and not a fresh Empty-as-chooser event.*

Proof. Inspect the replay row C. It displays a route whose first row is the current history H, followed by the selected mark, result row, extension route, selector witness, and ledger.

Because L is declared as a non-reset ledger row, it records the finite selector ledger continuing from H through M to R. The source history is therefore part of the public route, not external memory.

The structural rows preserve that route componentwise. No replay edge starts from e unless H itself is the displayed empty history, and no row supplies a hidden chooser at the boundary. □

Falsifiable prediction: Deleting the selected-mark row makes the packet unable to distinguish the two Ext branches at the empty boundary.

Independence witness: empty-fable-machine, observer-hist-identity, inscriptionevent, finitewindowobserver: the carrier is not bijective to these siblings because it jointly stores the current Hist, selected mark, Ext route, selector witness, and non-reset ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{EmptyBoundarySelectorLedger}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EmptyBoundarySelectorLedger}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2165.1, Definition 2165.2, Definition 2165.3, and Definition 2165.4 seed the finite empty-boundary selector ledger over current history, selected mark, Ext route, selector witness, non-reset ledger, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EmptyBoundarySelectorLedger}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Nontrivial or FieldFaithful Lean instances, bridge checking, observer substance, global branch choice, reset-to-Empty semantics, host trace equality, or truth-oracle semantics.

Upgrade path: Lean follow-up must provide Nontrivial and FieldFaithful instances for BEDC.Derived.EmptyBoundarySelectorLedgerUp while preserving the finite selected-mark and non-reset-ledger route.

Constructive story: An $\mathsf{EmptyBoundarySelectorLedger}^{\uparrow}$ packet is generated from current Hist, selected mark, result Hist, Ext route, selector witness, non-reset ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2166

A Concrete Naming Certificate for $\mathsf{FiniteVitaliCover}^{\uparrow}$

Following Vitali's finite covering argument, Bishop–Bridges located interval reasoning, and the dyadic-radius bookkeeping used by constructive real analysis, the $\mathsf{FiniteVitaliCover}^{\uparrow}$ packet records the finite subcover-selection surface as BEDC NameCert data. It sits after the located interval carrier of chapter 394, the dyadic rational core of chapter 291, and the real-value seal of chapter 38. The packet names finite interval rows, radius rows, a disjoint selected subfamily, shadow-coverage rows, componentwise transport, replay, provenance, and local naming only; it does not import measure, arbitrary open-cover compactness, countable choice, or host set equality.

Definition 2166.1 (FiniteVitaliCover carrier)**.** *A* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, R, S, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite located-interval row. It lists the candidate intervals by their* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoints and records the displayed membership windows through which a later real consumer may inspect each interval.*
2. *R is the dyadic radius row. It records the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius attached to each interval, together with the finite comparison windows used to order candidate radii.*
3. *S is the selected subfamily row. It records a finite list of chosen interval indices and the pairwise disjointness checks between the selected located intervals.*
4. *W is the shadow-coverage row. It records, for every candidate interval not selected, a selected interval with comparable or larger dyadic radius whose controlled enlargement covers the candidate's displayed located window.*
5. *H records componentwise* hsame *transport for the located-interval row, dyadic radius row, selected disjoint subfamily row, shadow-coverage row, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay of the finite route*

$$I, R \longmapsto S, W \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{FiniteVitaliCover}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{FiniteVitaliCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through I, R, S, W, H, C, P, N. It has no coordinate for Lebesgue measure, a countable choice sequence, open-cover compactness, a host set equality predicate, or a global family of balls.

Lean4 ✓ `BEDC.Derived.FiniteVitaliCoverUp.FiniteVitaliCoverTasteGate single carrier alignment`

Theorem 2166.2 (FiniteVitaliCover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *carrier*

$$V = (I, R, S, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteVitaliCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite located-interval row, dyadic radius row, selected disjoint subfamily row, shadow-coverage row, component-wise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The shadow row must witness that every unselected candidate interval is covered by the controlled enlargement of a selected interval whose dyadic radius is comparable or larger, and every public cover read must pass through the route displayed by C.*

Lean4 ✓ `BEDC.Derived.FiniteVitaliCoverUp.FiniteVitaliCoverCarrier namecert obligations`

Proof. Project the accepted carrier in Definition 2166.1. The only mathematical data exposed to a cover consumer are the located interval row I, dyadic radius row R, selected subfamily row S, and shadow-coverage row W.

The selected row records pairwise disjointness directly, while the shadow row records the finite witness from each unselected interval to a selected controlling interval. Thus the usual finite Vitali conclusion is read as a bounded finite shadow of the selected subfamily rather than as a measure-theoretic covering theorem.

The rows H, C, P, N transport, replay, source, and name exactly this finite surface. Since the carrier contains no coordinate for measure, countable choice, arbitrary open-cover compactness, host set equality, or an infinite ball family, the listed obligations exhaust the local NameCert surface. □

Theorem 2166.3 (Finite Vitali cover Real-completion subcover route)**.** *Every Real and Completion consumer of an accepted* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *packet factors through the finite located-interval row I, dyadic radius row R, selected finite subfamily row S, and shadow-coverage row W before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. The route supplies finite shadow subcover evidence only and exports no measure estimate, arbitrary open-cover compactness, countable choice, host set equality, or global ball family.*

Proof. Project the accepted carrier

$$V = (I, R, S, W, H, C, P, N)$$

from Definition 2166.1. The rows I, R, S, W are the only cover-facing rows available before the structural transport, replay, provenance, and naming rows are read.

Use Definition 2166.2. The selected row S supplies the finite disjoint subfamily, while the shadow row W supplies, for each unselected candidate, a selected controlling interval with comparable or larger dyadic radius.

Route every Real-facing read through located intervals and dyadic radii. The located row I exposes the interval windows, the radius row R exposes the dyadic comparisons, and C replays this finite path before any downstream consumer may inspect a terminal real seal.

The ledger is nonescaping because H, C, P, N transport, replay, source, and name only the displayed rows. Since no row stores a measure estimate, arbitrary open-cover compactness, countable choice, host set equality, or a global ball family, the route exports exactly the finite shadow subcover evidence stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVitaliCover}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteVitaliCover}\uparrow}$, seed)

Scope closed: Definition 2166.1 and Definition 2166.2 seed the finite Vitali-cover packet over located interval, dyadic radius, selected disjoint subfamily, shadow coverage, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteVitaliCover}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure estimates, open-cover compactness, countable choice, host set equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic radius ordering, selected-subfamily disjointness, shadow coverage, componentwise transport, provenance, and local naming for BEDC.Derived.FiniteVitaliCoverUp.

Constructive story: A $\mathsf{FiniteVitaliCover}^{\uparrow}$ packet is generated from finite LocatedInterval, DyadicRatCore, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 2167

A Concrete Naming Certificate for $\mathsf{StarRefinementUniformCover}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces and star refinements of uniform covers, the $\mathsf{StarRefinementUniformCover}^{\uparrow}$ packet records the finite cover-refinement surface needed by the L10 completion route. It sits after the uniform-space surface of chapter 339, the entourage-base surface of chapter 1819, the filter-base surface of chapter 1601, the rounded-ideal completion surface of chapter 1883, and the metric-completion surface of chapter 1506. The packet names displayed cover and star-refinement rows only; it does not import open-cover compactness, selected refinements, choice, or ambient completeness.

Definition 2167.1 (StarRefinementUniformCover carrier)**.** *A* $\mathsf{StarRefinementUniformCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, E, F, R, M, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform-cover row. It records the finite cover family read from the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage interface and the membership windows through which a downstream completion consumer may inspect it.*
2. *E is the* $\mathsf{EntourageBase}^{\uparrow}$ *row. It supplies the entourage-basis refinement data whose square or composite containment witnesses the star-refinement relation.*
3. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row. It records the directed finite base route that transports cover refinements to Cauchy and completion consumers.*
4. *R is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *row. It records the rounded-ideal completion consumer that may request the refinement as a formal-ball neighbourhood boundary.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row. It records the metric-completion consumer that may read the same refinement through metric entourage or ball data.*
6. *H records componentwise* hsame *transport for the uniform cover, entourage base, filter base, rounded-ideal completion, metric completion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$U, E, F \longmapsto R, M \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{StarRefinementUniformCover}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{EntourageBase}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{StarRefinementUniformCover}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $U, E, F, R, M, H, C, P, N$ and has no coordinate for open-cover compactness, a choice-selected refinement, a host star operation, or ambient completeness.

Lean4 ✓ `BEDC.Derived.StarRefinementUniformCoverUp.StarRefinementUniformCoverTasteGate single carrier alignment`

Theorem 2167.2 (StarRefinementUniformCover NameCert obligations)**.** *For an accepted* StarRefinementUniformC *carrier*

$$S = (U, E, F, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StarRefinementUniformCover}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the uniform-cover row, entourage-base row, filter-base row, rounded-ideal completion row, metric-completion row, transport, replay, provenance, and local naming row;*
2. *star-refinement containment, requiring the displayed entourage basis in E to witness that every star of the refined cover lies inside a displayed member of the cover row U;*
3. *filter-base handoff, requiring the directed base row F to preserve the same refinement relation for Cauchy and completion consumers;*
4. *rounded-ideal completion handoff, requiring R to read the refinement only as a formal-ball neighbourhood boundary already routed through U, E, F;*
5. *metric-completion handoff, requiring M to read the same refinement only through the displayed metric entourage or ball data already routed through U, E, F;*
6. *classifier transport, requiring* hsame *comparison to preserve the uniform-cover, entourage-base, filter-base, rounded-ideal, metric-completion, replay, provenance, and local naming rows componentwise;*
7. *non-escape, excluding open-cover compactness, choice, selected refinements, host star operations, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2167.1. The accepted packet exposes exactly the uniform-cover, entourage-base, filter-base, rounded-ideal completion, metric-completion, transport, replay, provenance, and local naming rows.

Star-refinement containment is read from E against U. The entourage-basis row supplies the finite refinement witness, and the cover row supplies the displayed members into which each star is contained.

The completion handoffs then consume the same route. The filter-base row F carries directed refinement data to Cauchy-style consumers, while R and M read the route as rounded-ideal and metric-completion neighbourhood data.

Finally H, C, P, N preserve the route componentwise. Since the carrier has no row for compactness of arbitrary open covers, selected refinements, choice, a host star operation, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2167.3 (StarRefinementUniformCover completion handoff)**.** *Let*

$$S = (U, E, F, R, M, H, C, P, N)$$

be an accepted $\mathsf{StarRefinementUniformCover}^{\uparrow}$ *carrier. Every completion-facing consumer of the star-refinement row factors through the finite route*

$$U \longmapsto E \longmapsto F \longmapsto R, M \longmapsto H, C, P, N.$$

The handoff supplies $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *with a displayed refinement row for uniform-cover arguments. It does not export open-cover compactness, choice, selected refinements, a host star operation, or ambient completeness.*

Proof. Begin with the accepted carrier S. The uniform-cover row U is the only cover entry point, so every completion-facing read first exposes the finite cover family and its membership windows.

The entourage-base row E then supplies the star-refinement witness. Its containment data refine the cover row already displayed by U, and the filter-base row F preserves that relation as a directed finite base route.

The completion consumers are downstream of this route. The rounded-ideal row R reads the refinement as a formal-ball neighbourhood boundary, while the metric-completion row M reads it through displayed metric entourage or ball data.

Transport by H, replay by C, provenance by P, and naming by N preserve the same finite route. No excluded host-side principle is available to any completion consumer. □

Theorem 2167.4 (StarRefinementUniformCover filter-base determinacy)**.** *Let*

$$S = (U, E, F, R, M, H, C, P, N)$$

be an accepted $\mathsf{StarRefinementUniformCover}^{\uparrow}$ *carrier. Any completion-facing consumer of a star-refinement row must preserve the same directed* $\mathsf{FilterBase}^{\uparrow}$ *route* F *when passing to* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *through* R *or to* $\mathsf{MetricCompletion}^{\uparrow}$ *through* M*. No selected refinement, host star operation, or ambient completeness row can replace* F*.*

Proof. Project the carrier of Definition 2167.1. The only directed completion-facing base row is F, and it is stored between the uniform cover U, the entourage-base refinement witness E, and the two completion consumer rows R, M.

Apply Definition 2167.2. Its filter-base handoff obligation requires the directed base row F to preserve the same refinement relation for Cauchy and completion consumers.

Finally apply Definition 2167.3. Both the rounded-ideal row R and the metric-completion row M are downstream of U, E, F and the structural rows H, C, P, N. Since the carrier has no selected refinement, host star operation, or ambient completeness coordinate, no completion consumer can replace the displayed filter-base route. □

Closure status for $\mathsf{NameCert}_{\mathsf{StarRefinementUniformCover}^{\uparrow}}$.

Constructive story: A $\mathsf{StarRefinementUniformCover}^{\uparrow}$ packet is generated from a uniform-cover row, an EntourageBaseUp refinement witness, a FilterBaseUp directed base route, RoundedIdealCompletionUp and MetricCompletionUp consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StarRefinementUniformCover}^{\uparrow}}$, seed)

Scope closed: Definition 2167.1, Definition 2167.2, and Definition 2167.3 seed the uniform-cover star-refinement surface for L10 completion routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StarRefinementUniformCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice, selected refinements, host star operations, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, star-refinement containment, filter-base handoff, rounded-ideal completion handoff, metric-completion handoff, classifier transport, and non-escape for BEDC.Derived.StarRefinementUniformCo

Chapter 2168

A Concrete Naming Certificate for $\mathsf{InverseLimitMetric}^{\uparrow}$

Following Bourbaki and Kelley on inverse limits of metric systems, the $\mathsf{InverseLimitMetric}^{\uparrow}$ packet records the L10 completion-neighbour obtained from finite projection rows rather than from a host inverse-limit object. It consumes the metric surface of chapter 57, the finite window discipline of chapter 126, the regular rational readbacks of chapter 301, the real-valued distance seals of chapter 38, and the completion-facing boundary of chapter 1506. The certificate names only finite diagram observations, projection compatibility, and local completion-neighbour data; it does not assert a host inverse limit, quotient completion equality, selected representatives, countable choice, or ambient completeness.

Definition 2168.1 (InverseLimitMetric carrier)**.** *An* $\mathsf{InverseLimitMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, \Pi, W, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite diagram row, listing the visible metric nodes, displayed bonding arrows, and nodewise* $\mathsf{Met}^{\uparrow}$ *evidence consumed by the packet.*
2. *Π is the projection row. Each projection observation names its source node, target node, finite bonding path, and compatibility check against the diagram row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to address finite node histories and finite projection histories.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for distance comparisons inside the displayed nodes and along displayed projection paths.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row. It seals the real-valued distance read after the finite window and regular readback rows have been displayed.*
6. *L is the local completion-neighbour row. It records the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff used by later completion consumers, without quotienting Cauchy names or selecting a representative point.*
7. *H records componentwise* hsame *transport for diagram rows, projection rows, stream windows, regular readbacks, real distance seals, local completion-neighbour rows, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$D \longmapsto \Pi \longmapsto W \longmapsto Q \longmapsto R \longmapsto L.$$

9. *P is the* Pkg *provenance over* $\mathsf{InverseLimitMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{InverseLimitMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, \Pi, W, Q, R, L, H, C, P, N$. It has no coordinate for a host inverse-limit carrier, quotient completion equality, selected representatives, countable-choice selectors, or an ambient complete metric space.

Lean4 (definition) `BEDC.Derived.InverseLimitMetricUp.InverseLimitMetricUp`

Theorem 2168.2 (InverseLimitMetric NameCert obligations)**.** *For every accepted* $\mathsf{InverseLimitMetric}^{\uparrow}$ *carrier $I = (D, \Pi, W, Q, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InverseLimitMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed diagram rows, finite projection rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *distance seals, local* $\mathsf{MetricCompletion}^{\uparrow}$ *neighbour handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read must pass through those rows in the order displayed by C.*

Proof. Project the accepted carrier in Definition 2168.1. The only available mathematical rows are the finite diagram D, projection data Π, stream windows W, regular readbacks Q, real distance seals R, and local completion-neighbour row L.

The replay row C orders those rows before the structural support is consumed. Thus a public read first sees finite diagram and projection data, then finite windows and regular readbacks, then the real distance seal and the completion-neighbour handoff.

The rows H, P, N transport, source, and name exactly this surface. Since the carrier has no coordinate for a host inverse-limit object, quotient completion equality, selected representatives, choice, or ambient completeness, the listed obligations exhaust the local certificate. □

Theorem 2168.3 (InverseLimitMetric projection stability)**.** *Let $I = (D, \Pi, W, Q, R, L, H, C, P, N)$ be an accepted* $\mathsf{InverseLimitMetric}^{\uparrow}$ *carrier. If a consumer transports I componentwise along the* hsame *rows recorded in H, then every finite projection compatibility check still factors through*

$$D, \Pi, W, Q, R, L, H, C, P, N.$$

In particular, transported projection rows cannot be replaced by a host inverse-limit point, a selected completion representative, a countable-choice schedule, or an ambient completeness witness.

Proof. The projection row Π is named as a coordinate of the carrier, and the transport row H stores componentwise hsame data for that coordinate together with the diagram, windows, readbacks, distance seals, and completion-neighbour row.

After transport, the only projection evidence available is the transported instance of Π over the transported diagram row D. The replay row C still routes finite projection observations through W, Q, R, L before provenance and local naming are read.

No coordinate remains from which a host inverse-limit point, chosen representative, choice schedule, or ambient complete metric space could be recovered. Therefore projection compatibility is stable exactly along the finite NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{InverseLimitMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InverseLimitMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2168.1, Definition 2168.2, and Definition 2168.3 seed the finite inverse-limit metric packet over diagram rows, projection rows, stream windows, regular readbacks, real distance seals, local completion-neighbour evidence, transport, replay, provenance, and naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InverseLimitMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse limits, quotient completion equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite projection compatibility, StreamName window routing, RegSeqRat readback, RealUp distance sealing, MetricCompletionUp neighbour handoff, transport stability, provenance, and local naming for BEDC.Derived.InverseLimitMetricUp.

Constructive story: An $\mathsf{InverseLimitMetric}^{\uparrow}$ packet is generated from finite metric diagram rows, projection rows, StreamName windows, RegSeqRat readbacks, RealUp distance seals, local MetricCompletionUp neighbour data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 2169

A Concrete Naming Certificate for $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$ packet records the finite boundary between regular Cauchy rational names and real-name seals. It sits after the regular rational sequence surface of chapter 301, the finite stream naming surface of chapter 126, the dyadic tolerance ledger of chapter 291, the real-name packet of chapter 38, and the uniform embedding surface of chapter 1662. The certificate lets completion consumers see the regular-name evidence and the terminal $\mathsf{Real}^{\uparrow}$ handoff while refusing to identify classifier equality with quotient-class equality.

Definition 2169.1 (RegularCauchyQuotientRefusal carrier)**.** *A* $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to name finite denominator and radius requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite observation schedule inspected at the requested tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row, read only through D and S.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row available after the regular rational name has been displayed.*
5. *U is the* $\mathsf{RealUniformEmbedding}^{\uparrow}$ *row, recording that the real-facing handoff is read through the displayed uniform embedding boundary rather than through quotient equality of Cauchy names.*
6. *H records componentwise* hsame *transport for tolerance ledgers, stream windows, regular rational names, real seals, uniform embedding rows, replay, provenance, and local naming.*
7. *C records the displayed* Cont *replay of the route*

$$D, S, R \longmapsto E, U \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient streams, host real equality, selected representatives, countable choice, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyQuotientRefusalUp.RegularCauchyQuotientRefusalTasteGate single carrier alignment`

Theorem 2169.2 (RegularCauchyQuotientRefusal NameCert obligations)**.** *For an accepted* RegularCauchyQuotien *carrier*

$$Q = (D, S, R, E, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the dyadic tolerance ledger, finite stream window, regular rational name, real seal, uniform embedding row, componentwise transport, replay, provenance, and local naming row;*
2. *regular-name exposure, requiring every real-facing consumer to read D, S, and R before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *row;*
3. *classifier non-escape, requiring any comparison between two accepted packets to use only componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows;*
4. *real-seal handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *consumer to pass through E and U without receiving quotient-stream equality, host real equality, selected representatives, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2169.1. The displayed packet lists the tolerance ledger, window schedule, regular rational name, real-facing seal, uniform embedding row, and the structural support rows.

Regular-name exposure is the dependency order of the carrier. A consumer cannot reach the real-seal row E until the finite dyadic request, stream window, and regular rational readback have all been displayed.

Classifier non-escape is componentwise. The rows H and C transport and replay only the displayed coordinates, while P and N record provenance and local naming for those coordinates.

The excluded principles are absent from the carrier. Since no field stores quotient streams, host real equality, selected representatives, countable choice, or a $\mathsf{Real}^{\uparrow}$ completeness witness, no NameCert obligation can export such data. □

Theorem 2169.3 (RegularCauchyQuotientRefusal classifier non-escape)**.** *Let $Q = (D, S, R, E, U, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$ *packet. Every classifier comparison exported by Q is exhausted by the rows*

$$D, S, R, E, U, H, C, P, N.$$

In particular, equality of two displayed classifier reads cannot be promoted to quotient equality of regular Cauchy streams, host equality of reals, a selected representative, a countable-choice schedule, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the classifier through the carrier rows of Definition 2169.1. The rows D, S, R expose the finite regular-name evidence before any real-facing comparison is available.

The rows E and U then record the terminal real seal and uniform embedding boundary. They are handoff rows, not quotient constructors, so they do not add a coordinate for equality of Cauchy streams or equality in a host real object.

Transport and replay remain componentwise through H and C. Provenance and local naming through P and N close the route at the same displayed rows.

Because no further row exists, the classifier cannot manufacture the excluded quotient, representative, choice, host-equality, or completeness data downstream. □

Theorem 2169.4 (RegularCauchyQuotientRefusal RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$ *packet factors through the finite route*

$$D \longmapsto S \longmapsto R \longmapsto E, U \longmapsto H, C, P, N.$$

Thus the consumer may read the dyadic tolerance ledger, finite stream window, regular rational name, terminal real seal, real-uniform-embedding boundary, componentwise transports, replay, provenance, and local NameCert *row. It receives no quotient stream, host real equality, selected representative, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project an accepted carrier Q. The only route to a real-facing read begins with the dyadic tolerance row D, then the finite stream window S, then the regular rational name row R.

The terminal seal rows are downstream of that route. Row E records the $\mathsf{Real}^{\uparrow}$-facing handoff, and row U records the uniform embedding boundary through which the handoff is read.

The structural rows preserve the same route. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming do not supply a quotient coordinate or a selected representative.

Therefore every $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed finite handoff and none of the excluded quotient, host-equality, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite window, RegSeqRat regular rational name, RealUp seal, RealUniformEmbedding boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2169.1, Definition 2169.2, Definition 2169.3, and Definition 2169.4 seed the finite regular-Cauchy-to-real handoff boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyQuotientRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter refuses quotient streams, host real equality, selected representatives, countable choice, and completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier non-escape, and RealUp seal handoff for BEDC.Derived.RegularCauchyQuotientRefusalUp.

Chapter 2170

A Concrete Naming Certificate for $\mathsf{SeparatedCompletionReflector}^{\uparrow}$

Following Bishop–Bridges on constructive metric completion and Kelley on separated uniform reflection, the $\mathsf{SeparatedCompletionReflector}^{\uparrow}$ packet records the L10 boundary where a displayed completion route may be read as separated before later complete-metric and real-completion consumers use it. It sits after the metric-completion carrier of chapter 1506, the separated metric boundary of chapter 1411, the uniform-completion route of chapter 1473, the completion universal-property row of chapter 1473, the Cauchy-filter surface of chapter 333, the Cauchy-net surface of chapter 589, the regular-rational readback surface of chapter 301, and the real seal of chapter 38. The reflector consumes the separated-completion row route of chapter 1612 when it turns metric-completion reads into separated reusable completion data. The packet is a finite reflector boundary, not a quotient completion, selected-limit principle, ambient complete space, host-equality bridge, or countable-choice schedule.

Definition 2170.1 (SeparatedCompletionReflector carrier)**.** *A* $\mathsf{SeparatedCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, U, Q, F, D, T, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row exposing the displayed metric-completion source and completion-facing read route.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row supplying the zero-distance classifier boundary before any completed read is treated as reusable.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row comparing the same Cauchy data through the displayed uniform route.*
4. *Q is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *handoff row read only after M, S, U have been displayed.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch row when the completion evidence is filter-presented.*
6. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *branch row when the completion evidence is net-presented.*
7. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance-readback row used by numerical consumers.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by real-completion consumers.*

9. *H records componentwise* hsame *transport for the completion, separation, uniform, universal-property, Cauchy, tolerance, real-seal, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$M, S, U, F, D, T \longmapsto Q \longmapsto A \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* SeparatedCompletionReflector$^{\uparrow}$, MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, CauchyFilter$^{\uparrow}$, CauchyNet$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{SeparatedCompletionReflector}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these twelve rows. Its classifier compares accepted reflector packets only through $M, S, U, Q, F, D, T, A, H, C, P, N$ and has no coordinate for quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.SeparatedCompletionReflectorUp.SeparatedCompletionReflectorTasteGate`
`single carrier alignment`

Theorem 2170.2 (SeparatedCompletionReflector NameCert obligations)**.** *For an accepted* SeparatedCompletionRe *carrier*

$$R = (M, S, U, Q, F, D, T, A, H, C, P, N),$$

the local NameCert$_{\text{SeparatedCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric-completion, separated-metric, uniform-completion, universal-property, Cauchy-filter, Cauchy-net, regular-rational, real-seal, transport, replay, provenance, and local naming rows;*

2. *separated-source exactness, requiring every reflector-facing completion read to project M and then S before U, Q, A can be consumed;*

3. *branch visibility, requiring every Cauchy read to use the displayed filter branch F or net branch D before tolerance data in T may be read;*

4. *universal handoff locality, requiring Q to consume only the displayed M, S, U, F, D, T surface and to export only the real-seal route A plus structural rows;*

5. *classifier transport, requiring* hsame *comparison to preserve the completion, separation, uniform, universal-property, Cauchy, tolerance, real-seal, replay, provenance, and local naming rows componentwise;*

6. *non-escape, excluding quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, and bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2170.1. The accepted packet exposes exactly the metric-completion, separated-metric, uniform-completion, universal-property, Cauchy-branch, tolerance, real-seal, transport, replay, provenance, and local naming rows.

Separated-source exactness starts at M and S. A consumer cannot read the uniform route U, the universal-property handoff Q, or the terminal real seal A until the displayed completion source and separated zero-distance classifier boundary have both been exposed.

Branch visibility is confined to F, D, T. The packet records whether the Cauchy evidence is filter-presented or net-presented before any regular-rational tolerance row can be inspected.

Finally Q, A, H, C, P, N preserve the same finite route. The universal-property row consumes only the displayed completion, separation, uniform, Cauchy, and tolerance rows; the real-seal row is downstream of that handoff; and the structural rows transport, replay, source, and name only displayed coordinates. Since no excluded host principle has a carrier row, the listed obligations exhaust the local NameCert surface. □

Theorem 2170.3 (SeparatedCompletionReflector universal handoff)**.** *Let*

$$R = (M, S, U, Q, F, D, T, A, H, C, P, N)$$

be an accepted $\mathsf{SeparatedCompletionReflector}^{\uparrow}$ *carrier. Every complete-metric or real-completion consumer reaches the universal-property row Q and the terminal real-seal row A only through the finite route*

$$M \longmapsto S, U \longmapsto F, D, T \longmapsto Q \longmapsto A \longmapsto H, C, P, N.$$

The handoff refines the $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *surfaces by inserting the* $\mathsf{SeparatedMetric}^{\uparrow}$ *boundary before universal extension data are exported. It does not export quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with an accepted carrier R. The metric-completion row M is the only completion-object entry point, and the separated metric row S is the only row that authorizes reuse of completed reads through zero-distance classifier evidence.

The uniform row U and Cauchy branch rows F, D are then read as constraints on the same completion data. The tolerance row T supplies finite regular-rational readback before the universal-property row is allowed to export a consumer-facing extension route.

The row Q is therefore downstream of the displayed completion, separation, uniform, Cauchy, and tolerance data. The real-seal row A and the structural rows H, C, P, N preserve that order, so the handoff reaches consumers without introducing any excluded host-side principle. □

Theorem 2170.4 (SeparatedCompletionReflector non-escape)**.** *Every consumer of an accepted* $\mathsf{SeparatedCompletionReflector}^{\uparrow}$ *carrier may inspect completion reflection only through the displayed rows*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow},\ \mathsf{CompletionUniversalProperty}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyNe}$$

No consumer route can read a quotient completion, selected limit, host equality of completed points, countable-choice schedule, ambient complete object, Lean verification certificate, or standard bridge from this packet.

Proof. The non-escape claim is field exclusion. The carrier of Definition 2170.1 lists the only rows admitted by the packet, and the NameCert obligations of Definition 2170.2 require every accepted read to project through those rows.

Any route asking for a quotient completion, selected limit, host equality, countable choice, ambient completeness, Lean verification, or standard bridge would require an additional coordinate. Since no such coordinate occurs in R, the local classifier has no accepted read for that request. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCompletionReflector}^{\uparrow}$ packet is generated from MetricCompletionUp, SeparatedMetricUp, UniformCompletionUp, CompletionUniversalPropertyUp, CauchyFilterUp, CauchyNetUp, RegSeqRatUp, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2170.1, Definition 2170.2, Definition 2170.3, and Definition 2170.4 seed the separated reflector boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflector}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.SeparatedCompletionReflectorUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, host equality, countable choice, ambient completeness, Lean verification beyond the encoded carrier, or bridge checking.

Upgrade path: Obligation closure requires checked separated-source exactness, Cauchy-branch visibility, universal handoff locality, classifier transport, and non-escape for BEDC.Derived.SeparatedCompletionReflectorUp.

Chapter 2171

A Concrete Naming Certificate for $\mathsf{LocalTimeFiber}^{\uparrow}$

The $\mathsf{LocalTimeFiber}^{\uparrow}$ packet realizes the observer-local reading of section 9462.22 as finite NameCert data. It sits between the observer-history identity surface of chapter 479, the local time-stream claims of Definition 9462.592, the no-global-clock boundary of Definition 9462.602, the local clock budget of chapter 665, and the synchronization certificate of chapter 1189. The packet exposes observer-local Hist growth as finite source, Cont-step, transport, refusal, provenance, and NameCert rows rather than as a global time axis.

Definition 2171.1 (LocalTimeFiber carrier)**.** *A* $\mathsf{LocalTimeFiber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, S, K, B, R, T, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{ObserverHistIdentity}^{\uparrow}$ *source row naming the observer as a displayed history surface.*
2. *S is the hist-time-stream row recording the local sequence of* Hist *constructor growth for that observer.*
3. *K is the local-clock-budget row bounding which local steps may be read by a consumer.*
4. *B is the* $\mathsf{SynchronizationCertificate}^{\uparrow}$ *boundary row, present only as a certified comparison surface and not as a global clock.*
5. *R is the refusal row excluding a global present, absolute clock, detached observer frame, and host equality of time fibers.*
6. *T records componentwise* hsame *transport for observer identity, local stream, local budget, synchronization boundary, refusal, provenance, and local naming.*
7. *C records displayed* Cont *replay of the observer-local step route*

$$O, S, K \longmapsto B, R \longmapsto T, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocalTimeFiber}^{\uparrow}$*,* $\mathsf{ObserverHistIdentity}^{\uparrow}$*, hist-time-stream, local-clock-budget,* $\mathsf{SynchronizationCertificate}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalTimeFiber}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted local time fibers only through $O, S, K, B, R, T, C, P, N$ and has no coordinate for a global clock, absolute present, detached observer substance, unbudgeted step selector, host equality of histories, Lean verification beyond the encoded carrier, or bridge checking.

Lean4 ✓ `BEDC.Derived.LocalTimeFiberUp.TasteGate.LocalTimeFiberTasteGate single carrier alignment`

Theorem 2171.2 (LocalTimeFiber NameCert obligations)**.** *For an accepted* $\mathsf{LocalTimeFiber}^{\uparrow}$ *carrier*

$$L = (O, S, K, B, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocalTimeFiber}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the observer-history source, local Hist time stream, local clock budget, synchronization boundary, refusal, transport, replay, provenance, and local naming rows;*
2. *observer-time-step exactness, requiring every time read to project the observer-history row O before it projects a local step from S;*
3. *hist-time-stream locality, requiring each step read to stay within the displayed local stream rather than a shared external index;*
4. *local-clock-budget boundedness, requiring K before any consumer can request a local step window;*
5. *synchronizationcertificate confinement, allowing B only as a finite comparison certificate and never as a global clock or absolute present;*
6. *observerhistidentity transport, requiring* hsame *comparison to preserve the observer, stream, budget, synchronization, refusal, replay, provenance, and local naming rows componentwise;*
7. *non-escape, excluding a global clock, detached observer frame, host equality of histories, unbudgeted local-step selection, Lean verification beyond the encoded carrier, and bridge checking.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2171.1. The accepted packet exposes exactly the observer-history source, local stream, budget, synchronization boundary, refusal, transport, replay, provenance, and local naming rows.

Observer-time-step exactness starts at O. A consumer cannot inspect a local time step until the observer-history source has been displayed, so the time fiber is attached to an observer history rather than to a detached clock carrier.

The stream and budget obligations are confined to S and K. The stream row supplies local Hist growth, and the budget row bounds which local windows may be read by downstream consumers.

The synchronization and refusal obligations are supplied by B and R. A synchronization certificate may compare displayed local fibers, but the refusal row blocks any reading of that comparison as a global present, absolute clock, detached frame, or host equality.

Finally T, C, P, N preserve the same finite route. Since no excluded host-side time principle has a carrier row, the listed obligations exhaust the local NameCert surface. □

Theorem 2171.3 (LocalTimeFiber no-global-clock boundary)**.** *Every consumer of an accepted* $\mathsf{LocalTimeFiber}^{\uparrow}$ *carrier reads observer-local time only through*

$\mathsf{ObserverHistIdentity}^{\uparrow}$, *hist-time-stream*, *local-clock-budget*, $\mathsf{SynchronizationCertificate}^{\uparrow}$, BHist, hsame, Cont, Pkg, N

It cannot read a global clock, absolute present, detached observer frame, host equality of histories, unbudgeted local-step selector, Lean verification beyond the encoded carrier, or standard bridge from this packet.

Proof. The boundary is field exclusion. The carrier of Definition 2171.1 lists the only rows admitted by the packet, and the NameCert obligations of Definition 2171.2 force every accepted read to project through those rows.

A global clock, absolute present, detached observer frame, host equality of histories, unbudgeted step selector, or standard bridge would require an additional coordinate. Since no such coordinate occurs in L, the local classifier has no accepted read for that request. □

Falsifiable prediction: A LocalTimeFiberUp read fails whenever a consumer asks for a global clock, absolute present, detached observer frame, host equality of histories, or unbudgeted local-step selector not displayed in the local fiber packet.

Independence witness: observer-time-step, hist-time-stream, local-clock-budget, observerhistidentity, synchronizationcertificate: none is bijective to $\mathsf{LocalTimeFiber}^{\uparrow}$ because the local fiber jointly carries observer source, local Hist stream, budget, synchronization boundary, refusal, transport, replay, provenance, and local NameCert rows, while each sibling exposes only one surface of that route.

The Lean follow-up must provide `Nontrivial LocalTimeFiberUp` and `FieldFaithful LocalTimeFiberUp` in `lean4/BEDC/Derived/LocalTimeFiberUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalTimeFiber}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalTimeFiber}^{\uparrow}$ packet is generated from an ObserverHistIdentityUp source row, local Hist time stream, local clock budget, SynchronizationCertificateUp boundary, global-clock refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalTimeFiber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2171.1, Definition 2171.2, and Definition 2171.3 seed the observer-local time-fiber surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalTimeFiber}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.LocalTimeFiberUp`

Bridge status: none

Not claimed: This chapter does not close a global clock, absolute present, detached observer frame, host equality of histories, unbudgeted step selection, Lean verification beyond the encoded carrier, or bridge checking.

Upgrade path: Obligation closure requires checked observer-time-step exactness, hist-time-stream locality, local-clock-budget boundedness, synchronizationcertificate confinement, observerhistidentity transport, and non-escape for BEDC.Derived.LocalTimeFiberUp.

Chapter 2172

A Concrete Naming Certificate for $\mathsf{CauchyInterleavingProduct}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyInterleavingProduct}^{\uparrow}$ packet records the finite product route obtained by alternating two regular Cauchy observation schedules before exposing the pointwise product readback. It sits between the interleaving surface of chapter 414, the product budget surface of chapter 461, and the root Cauchy product route of chapter 464. Its dependencies are the dyadic arithmetic core of chapter 291, the finite stream windows of chapter 126, the regular rational sequence route of chapter 301, and the terminal real seal of chapter 38.

Definition 2172.1 (CauchyInterleavingProduct carrier)**.** *A* $\mathsf{CauchyInterleavingProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, S_A, S_B, \sigma, M, D_A, D_B, D, E, R, I, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source names whose scheduled rational observations may be inspected;*
2. *S_A and S_B are the two* $\mathsf{StreamName}^{\uparrow}$ *windows used to expose the selected source observations;*
3. *σ is the alternating selector row, with even reads attached to A and odd reads attached to B;*
4. *M is the shared modulus and product-budget ledger recording the selected source precisions, endpoint magnitude bounds, and output tolerance;*
5. *D_A and D_B are the selected* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows obtained from the two scheduled windows;*
6. *D is the displayed dyadic product row for $D_A \cdot D_B$ at the common exponent named by M;*
7. *E is the finite product-error ledger attached to the dyadic product row;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *product readback whose scheduled point row is D with tolerance budget E;*
9. *I is the interleaving provenance row binding R to the alternating selector, both schedules, and the two source names;*
10. *H stores componentwise* hsame *transports for the source names, schedules, selector, modulus ledger, dyadic endpoint rows, product ledger, readback, and interleaving provenance;*

11. *C stores the displayed* Cont *routes for selecting a source side, reading the two windows needed by the product, forming the dyadic endpoint product, checking the product budget, and exposing the readback;*
12. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingProduct}^{\uparrow}}$ *row.*

The carrier is a finite observation certificate. It stores no quotient stream, selected real limit, ambient completeness object, or host-level multiplication law over completed reals.

Lean4 ✓ `BEDC.Derived.CauchyInterleavingProductUp.CauchyInterleavingProductTasteGate single carrier alignment`

Theorem 2172.2 (CauchyInterleavingProduct NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingProduct}^{\uparrow}}$ *surface for the carrier of Definition 2172.1 consists exactly of carrier admission, selector-window coverage, dyadic product formation, finite product-budget exactness, regular readback handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every consumer route for Q factors through*

$$A, B, S_A, S_B, \sigma, M, D_A, D_B, D, E, R, I, H, C, P, N.$$

No obligation exports quotient stream equality, a selected limit, host real equality, ambient Real completeness, or completed-real multiplication.

Proof. Carrier admission is projection from the displayed tuple Q. The two source names, two schedules, selector, modulus ledger, dyadic endpoint rows, product row, product-error ledger, readback row, and interleaving provenance are explicit coordinates.

Selector-window coverage follows the same even-and-odd partition used by the regular Cauchy interleaving packet. Product formation then reads the selected windows through the dyadic core and records the output tolerance in the finite product ledger.

The readback row R is downstream of those displayed rows. The transport, continuation, package, and naming coordinates attach the same finite route to every accepted consumer, so no omitted quotient, selected limit, host equality, completeness, or completed multiplication row can enter the local NameCert surface. □

Theorem 2172.3 (CauchyInterleavingProduct diagonal product route)**.** *For every accepted* CauchyInterleavingProd *carrier Q, the diagonal product route first reads the alternating selector σ, then exposes the two scheduled windows S_A, S_B at the common product request, then forms the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product row D from D_A and D_B, and only then exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R. The route is diagonal because both selected source observations are checked against the same ledger M before the product row is accepted.*

Proof. Project the selector, schedule, and modulus coordinates of Q. The selector chooses the active interleaving side, while the common product request still displays both windows needed to form the product.

Read D_A and D_B through S_A, S_B under the ledger M. The dyadic product row D is then a finite core row at the displayed common exponent, and the product-error row E records the tolerance assigned to that row.

Finally repack D and E as the regular readback R with interleaving provenance I. Since the route never leaves the displayed tuple, the product is diagonal over the finite schedules rather than a multiplication of selected completed limits. □

Theorem 2172.4 (CauchyInterleavingProduct nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{CauchyInterleavingProduct}^{\uparrow}$ *carrier must pass through the regular readback* R*, the interleaving provenance* I*, the dyadic product row* D*, the product-error ledger* E*, the two source schedules, the shared ledger* M*, and the structural rows* H, C, P, N*. The packet exposes no route to quotient stream equality, selected host limits, ambient completeness, or completed-real multiplication.*

Proof. The real-facing row is downstream of the regular readback. By the carrier definition, R is not available without the source schedules, dyadic endpoint product, and finite product-error ledger.

The interleaving provenance row I keeps the readback attached to the alternating selector and to the two source names. The structural rows H, C, P, N preserve this route under transport and consumer replay.

No field of Q names a quotient stream, a selected limit, an ambient completeness witness, or a completed-real multiplication operation. Thus every real-facing consumer remains inside the finite interleaving-product certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingProduct}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyInterleavingProduct}^{\uparrow}$ packet is generated from two regular Cauchy source names, two stream windows, an alternating selector, a shared modulus and product-budget ledger, dyadic endpoint rows, a dyadic product row, a product-error ledger, a regular readback, interleaving provenance, transport, replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2172.1, Definition 2172.2, Definition 2172.3, and Definition 2172.4 bind the finite product/interleaving certificate for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected real limits, ambient completeness, host real equality, or completed-real multiplication.

Upgrade path: Lean closure requires carrier admission, selector-window coverage, dyadic product formation, product-budget exactness, regular readback handoff, nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyInterleavingProductUp.

Chapter 2173

A Concrete Naming Certificate for $\mathsf{DenseExtensionMetric}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{DenseExtensionMetric}^{\uparrow}$ packet records the finite BEDC surface for extending a uniformly continuous map from a dense metric source into a displayed complete metric target. It consumes the metric-completion route of chapter 1506, the unit admission route of chapter 1527, the complete target boundary of chapter 128, and the naming spine of chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a constructive dense-extension certificate over displayed rows; it is not quotient completion equality, not a selected host limit, not countable choice, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 2173.1 (Dense extension metric carrier)**.** *A* $\mathsf{DenseExtensionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, J, \omega, M, Y, S, R, D, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, including the source point histories and distance-read interface inspected before completion data are consumed.*
2. *J is the dense embedding row, recording the displayed source-to-completion admission through the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *and the dense image boundary used by completion consumers.*
3. *ω is the uniform-continuity modulus row for the source map on dense points, carrying the finite tolerance conversion from source metric reads to target metric reads.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row through which completion-facing source names are read.*
5. *Y is the* $\mathsf{CompleteMetric}^{\uparrow}$ *target row, carrying the displayed complete-metric limit interface for the transported image stream.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window route for the dense source names and transported image names.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route for rational regularity checks attached to S.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic budgets consumed by X, J, ω, M, Y, S, R.*
9. *Q is the* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after the stream, regular-rational, and dyadic routes have been displayed.*

10. *T is the extension transport row, assigning each admitted completion-side source name a target read through Y and recording agreement with the dense source map on the image of J.*

11. *H is the componentwise* hsame *transport ledger for the source metric, dense embedding, modulus, completion handoff, complete target, stream, rational, dyadic, real seal, and extension transport rows.*

12. *C is the displayed* Cont *replay route forcing every extension read to pass through*

$$X, J, \omega \longmapsto M, S, R, D \longmapsto Y, Q, T.$$

13. *P is the* Pkg *provenance over* $\mathsf{DenseExtensionMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseExtensionMetric}^{\uparrow}}$ *row.*

No coordinate of the carrier stores quotient completion equality, selected representatives, countable-choice schedules, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DenseExtensionMetricUp.DenseExtensionMetricTasteGate single carrier alignment`

Theorem 2173.2 (Dense extension metric NameCert obligations)**.** *For an accepted* $\mathsf{DenseExtensionMetric}^{\uparrow}$ *carrier $E = (X, J, \omega, M, Y, S, R, D, Q, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseExtensionMetric}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source metric, dense embedding, modulus, metric-completion handoff, complete target, stream windows, regular readbacks, dyadic ledger, real seal, extension transport, componentwise transport, replay, provenance, and local naming rows;*

2. *dense-entry exposure, requiring every completion-side source read to enter through J before M is consumed;*

3. *modulus transport, requiring every target Cauchy request to use ω after the source distance reads in X and before the complete-target row Y is queried;*

4. *completion handoff, requiring source completion names to pass through M, S, R, D, Q before an extension value is offered;*

5. *target completeness handoff, requiring the transported image stream to reach a target read only through Y and T;*

6. *dense agreement, requiring T to restrict to the original dense-source map on the image carried by J;*

7. *ledger non-escape, stating that H, C, P, N export no quotient completion, selected representative, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 2173.1. The tuple lists the source metric, dense embedding, modulus, completion handoff, complete target, stream, rational, dyadic, real, transport, replay, provenance, and local naming rows.

Dense-entry exposure uses J as the only carrier row connecting source points to completion-facing names. Thus a completion-side source read must pass through the displayed dense embedding before M can be consumed.

Modulus transport uses ω between the source metric row X and the target complete-metric row Y. The modulus row is the only tolerance-conversion coordinate in the packet, so every transported image request is routed through it.

Completion handoff is the ordered read of M, S, R, D, Q. The metric-completion row supplies the completion-facing source name, while the stream, regular-rational, dyadic, and real-seal rows provide the finite readback surface required before the target extension row is reached.

The complete-target and dense-agreement clauses are then projections to Y and T. The target read is accepted only through the displayed complete metric row, and T records agreement on the dense image carried by J.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since no excluded host-side object appears as a coordinate, the listed obligations exhaust the local certificate surface. □

Theorem 2173.3 (Dense extension metric completion route)**.** *Let $E = (X, J, \omega, M, Y, S, R, D, Q, T, H, C, P, N)$ be an accepted* $\mathsf{DenseExtensionMetric}^{\uparrow}$ *carrier. Every completion-side extension read exported by E factors through*

$$X, J, \omega \longmapsto M, S, R, D, Q \longmapsto Y, T \longmapsto H, C, P, N.$$

Consequently the packet exports a displayed dense-extension handoff from source metric rows to complete target rows, not quotient completion equality, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-side source read. The carrier first exposes the source metric row X and the dense embedding row J, so the request enters the completion interface through displayed dense-source data.

The modulus row ω is then consumed before target convergence can be queried. This fixes the finite tolerance conversion used to transport source Cauchy evidence into the target metric row.

The completion handoff is read through M, S, R, D, Q. These rows provide the metric-completion surface, finite windows, regular rational readbacks, dyadic budgets, and terminal real seal for the source-side request.

Only then can the complete target row Y and extension transport row T be consumed. The structural rows H, C, P, N preserve that route componentwise, so no quotient completion, selected limit, choice schedule, host equality, or $\mathsf{Real}^{\uparrow}$ completeness theorem is introduced. □

Theorem 2173.4 (Dense extension metric downstream non-escape)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenseExtensionMetric}^{\uparrow}$ *carrier receives only the rows named in Definition 2173.1. In particular, downstream consumers receive no quotient completion equality, selected representative, countable-choice schedule, host equality of completed objects, or completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. A $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives M only after the source metric row X, the dense embedding row J, and the modulus row ω have been displayed. The stream, regular-rational, dyadic, and real-seal consumers are confined to S, R, D, Q along the same route.

A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives Y only after the source completion handoff and finite readback rows have been fixed. The extension transport row T records the target read and dense agreement, but it has no coordinate for a host completion object or a chosen representative.

The structural rows H, C, P, N preserve and name this route. Since every downstream read is a projection from the same finite packet, the excluded quotient, choice, host-equality, and ambient-completeness data cannot escape to any listed consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{DenseExtensionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseExtensionMetric}^{\uparrow}$ packet is generated from a source metric row, dense embedding row, uniform-continuity modulus row, MetricCompletion handoff, CompleteMetric target row, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, extension transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseExtensionMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2173.1, Definition 2173.2, Definition 2173.3, and Definition 2173.4 seed the finite dense-extension theorem route over metric completion and complete target rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseExtensionMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected limits, countable choice, host equality of completed objects, open-cover compactness, Lean verification, bridge checking, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-entry exposure, modulus transport, completion handoff, complete-target handoff, dense agreement, transport replay, and ledger non-escape for BEDC.Derived.DenseExtensionMetricUp

Chapter 2174

A Concrete Naming Certificate for $\mathsf{CauchyApproximationScheme}^{\uparrow}$

Following Bishop and Bridges constructive approximation of Cauchy names, the $\mathsf{CauchyApproximationScheme}^{\uparrow}$ packet records the finite schedule that lets a consumer request dyadic precision, read a finite window of a name, and hand the resulting regular rational data toward the real boundary. It sits between the dyadic tolerance core of chapter 291, the stream observation surface of chapter 126, the regular rational sequence readback of chapter 301, the convergence schedule of chapter 1438, the net-facing carrier of chapter 589, and the sealed real interface of chapter 38. The chapter packages the approximation schedule before quotient completion, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness is available.

Definition 2174.1 (Cauchy approximation scheme carrier)**.** *A* $\mathsf{CauchyApproximationScheme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, M, R, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic radii, comparison cells, shrink rows, and precision budgets available to the approximation consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It records the finite name windows whose entries may be inspected at each displayed precision demand.*
3. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule row. It assigns to each tolerance request a finite window boundary that is allowed to justify a Cauchy comparison.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed dyadic ledger, stream windows, and schedule.*
5. *W is the* $\mathsf{CauchyNet}^{\uparrow}$ *compatibility row. It records the net-facing approximation window reached from the same finite schedule.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records only the boundary interface reached through R and W, not a host real representative.*
7. *H records componentwise* hsame *transport for the tolerance ledger, stream windows, convergence schedule, regular readback, net window, real seal, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$D, S, M \longmapsto R, W \longmapsto Q \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationScheme}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the ten displayed rows. It has no coordinate for quotient completion, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationSchemeUp.CauchyApproximationSchemeTasteGate single carrier alignment`

Theorem 2174.2 (Cauchy approximation scheme NameCert obligations)**.** *For every accepted* $\mathsf{CauchyApproximationScheme}^{\uparrow}$ *carrier*

$$A = (D, S, M, R, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyApproximationScheme}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, stream observation windows, convergence schedule, regular readback, net window, real seal, transport, replay, provenance, and local naming row;*
2. *dyadic precision exposure, requiring every tolerance demand to be witnessed by a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, shrink row, comparison cell, or budget entry in D;*
3. *stream-window exposure, requiring every inspected name value to enter through the finite* $\mathsf{StreamName}^{\uparrow}$ *row S;*
4. *modulus routing, requiring M to bind each displayed precision request to a finite Cauchy window before R, W, or Q is read;*
5. *regular and net handoff, requiring R and W to consume the same displayed D, S, M data before any real-facing boundary is exposed;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the rows listed above and no quotient completion, selected limit, countable-choice row, trichotomy row, host real equality, or ambient completeness principle.*

No Cauchy-approximation-scheme obligation is stored outside these rows.

Proof. Project the carrier tuple $A = (D, S, M, R, W, Q, H, C, P, N)$. The carrier definition lists every coordinate visible to the local certificate, so admission is just the displayed finite packet with its transport, replay, provenance, and naming rows.

The precision request first reads D and S. The dyadic ledger supplies the tolerance data, while the stream row supplies only finite windows of the Cauchy name.

The schedule row M then fixes the Cauchy window that may justify the request. Since M is displayed before R, W, Q, no later consumer can replace the finite window by a selected limit or an ambient completeness assertion.

Finally R and W are the regular-sequence and net-facing handoffs, and Q is only the real seal reached after those handoffs. The rows H, C, P, N preserve and name that same route, so the listed obligations exhaust the NameCert surface. □

Theorem 2174.3 (Cauchy approximation scheme window stability)**.** *Let*

$$A = (D, S, M, R, W, Q, H, C, P, N)$$

and

$$A' = (D', S', M', R', W', Q', H', C', P', N')$$

be accepted $\mathsf{CauchyApproximationScheme}^{\uparrow}$ *packets. If the displayed* hsame *rows identify the dyadic tolerance ledgers, stream windows, and convergence schedules* D, S, M *with* D', S', M' *componentwise, then every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer sees the same finite approximation window from* A *and* A'*. The consumer can distinguish packets only through the displayed route*

$$D, S, M \longmapsto R, W \longmapsto Q$$

together with its transport, continuation replay, provenance, and local naming rows.

Proof. Start with the NameCert obligation theorem. Every consumer-facing read of the packet must pass through the dyadic ledger, stream windows, and convergence schedule before it reaches a regular readback, net window, or real seal.

Apply the componentwise hsame hypotheses to those first three rows. They identify the tolerance request, the inspected stream window, and the scheduled Cauchy boundary in the two packets.

The handoff rows R, W, Q are read only after the identified data, and the structural rows H, C, P, N and H', C', P', N' transport, replay, source, and name that same route. Thus the consumer-visible approximation window is stable under the displayed transport, with no additional coordinate available to refine the comparison. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationScheme}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationScheme}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2174.1, Definition 2174.2, and Definition 2174.3 seed the finite Cauchy approximation schedule over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationScheme}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic precision exposure, stream-window exposure, modulus routing, regular and net handoff, window stability, and provenance exhaustion for BEDC.Derived.CauchyApproximationSchemeU

Constructive story: A $\mathsf{CauchyApproximationScheme}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, a Cauchy convergence schedule, RegSeqRat readback, CauchyNet compatibility, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2175

A Concrete Naming Certificate for $\mathsf{DyadicCompactness}^{\uparrow}$

Following Bishop and Bridges on constructive compactness, the $\mathsf{DyadicCompactness}^{\uparrow}$ packet records the finite dyadic net surface for a located closed bounded real interval before any Real or Completion consumer reads a compactness-shaped seal. It sits after the interval-net carrier of chapter 1250, the dyadic cell surface of chapter 306, the rational interval ledger of chapter 354, the finite stream-window interface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The chapter names finite-net compactness data only; it does not assert Heine–Borel open-cover compactness, select subcovers, quotient streams, host real equality, host limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2175.1 (Dyadic compactness carrier)**.** *A* $\mathsf{DyadicCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, Q, D, M, Z, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the located closed bounded interval-net row supplied by* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$*, including lower and upper located endpoint windows;*
2. *Q is the finite* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger attached to the located endpoint row;*
3. *D is the finite* $\mathsf{DyadicInterval}^{\uparrow}$ *cell ledger refining the rational cells of Q;*
4. *M is the unary mesh scale naming the dyadic precision at which coverage is read;*
5. *Z is the* $\mathsf{ProbeBundle}$ *center ledger, with one displayed center for each accepted dyadic cell;*
6. *F is the finite coverage ledger saying that each displayed interval read enters a rational cell in Q, its dyadic refinement in D, and the corresponding center row in Z;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect endpoints, cells, and centers at mesh scale M;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same endpoint, cell, and center windows;*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, Q, D, M, Z, F, S, R have been displayed;*
10. *H records componentwise* hsame *transport for located endpoints, rational cells, dyadic cells, mesh scale, centers, coverage rows, stream windows, regular-rational readbacks, and the Real seal;*

11. *C records displayed* Cont *replay from the closed bounded interval net through rational cells, dyadic cells, mesh scale, center rows, finite coverage, StreamName windows, RegSeqRat readback, and RealUp sealing;*
12. *P is the* Pkg *provenance over* $\mathsf{DyadicCompactness}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCompactness}^{\uparrow}}$ *row.*

The carrier exports no open cover, chosen finite subcover, quotient stream representative, host real equality, selected host limit, hidden compactness principle, or ambient Real-completeness row.

Lean4 ✓ `BEDC.Derived.DyadicCompactnessUp.DyadicCompactnessTasteGate single carrier alignment`

Theorem 2175.2 (Dyadic compactness NameCert obligations)**.** *For the carrier of Definition 2175.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (L, Q, D, M, Z, F, S, R, E, I$ closed bounded interval-net admission through L, rational interval cell admission through Q, dyadic cell admission through D, unary mesh admission through M, ProbeBundle center admission through Z, finite coverage admission through F, StreamName and RegSeqRat readback through S and R, RealUp sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier row by row. The first block displays a closed bounded interval net, rational interval cells, dyadic refinements, a unary mesh scale, ProbeBundle centers, and a finite coverage ledger.

Read the analytic handoff only after the finite rows have been fixed. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks inspect endpoints, cells, and centers at the named mesh scale, and the $\mathsf{Real}^{\uparrow}$ seal is downstream of those windows.

The structural rows close the NameCert surface. The transport row H, replay row C, package row P, and local name N preserve only the displayed finite packet, so no further obligation can enter from an open cover, a chosen subcover, a quotient stream, a host real equality test, a selected limit, or ambient completeness. □

Theorem 2175.3 (Dyadic compactness finite-net coverage)**.** *Every accepted containment read of a* $\mathsf{DyadicCompactness}^{\uparrow}$ *carrier factors through the finite route*

$$L, Q, D, M, Z, F, H, C, P, N.$$

The read consumes the closed bounded interval-net row, the rational interval cell ledger, the dyadic refinement ledger, the unary mesh scale, the matching ProbeBundle center row, and the finite coverage row. It receives no open-cover compactness principle, hidden subcover choice, quotient stream representative, host real equality, selected host limit, or ambient Real-completeness row.

Proof. Induct over the finite mesh cells displayed by Q and D. In the base cell, the located endpoint read is admitted by L, refined by the first dyadic cell in D, and routed to its displayed ProbeBundle center in Z at mesh scale M.

For the successor step, keep the previously admitted cells fixed and read the next rational cell with its dyadic refinement. The coverage row F appends only that displayed cell-center route, while H transports the same finite rows componentwise.

Finally replay the accumulated route through C and package it through P, N. Since the carrier has no coordinate for open covers, subcover choices, quotient stream representatives, host real equality, selected limits, or ambient completeness, the induction cannot leave the finite dyadic net. □

Theorem 2175.4 (Dyadic compactness Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{DyadicCompactness}^{\uparrow}$ *packet reads the compactness data through the finite route*

$$L, Q, D, M, Z, F, S, R, E, H, C, P, N.$$

The Real seal row E is exposed only after the closed bounded interval net, rational cells, dyadic cells, unary mesh, ProbeBundle centers, finite coverage rows, StreamName windows, and RegSeqRat readbacks have been displayed. Consequently the downstream Real and Completion route receives a finite dyadic net packet and receives no Heine–Borel open-cover theorem, hidden finite-subcover selection, quotient stream equality, selected host real limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness premise.*

Proof. Start with Definition 2175.2. It places E after the finite rows L, Q, D, M, Z, F, S, R, so a Real-facing consumer cannot inspect a seal before the mesh, center, coverage, and readback data have been displayed.

Use Definition 2175.3 for the containment side. The coverage read is already confined to rational cells, dyadic refinements, a unary mesh scale, and ProbeBundle centers, so replacing it by an open cover or by a chosen finite subcover would require a row absent from the carrier.

The remaining structural rows H, C, P, N transport, replay, source, and name the same finite packet. Therefore every Real and Completion consumer receives the displayed dyadic net route and no quotient stream equality, selected host real limit, host real equality, or ambient completeness premise. □

Falsifiable prediction: Every Real or Completion read of a DyadicCompactnessUp packet factors through the displayed finite dyadic net route and not through an open-cover compactness row.

Independence witness: closedboundedintervalnet, dyadicinterval, rationalinterval, streamname, regseqrat, real: DyadicCompactnessUp is not bijective to any listed sibling because it jointly carries a closed bounded interval-net row, rational cells, dyadic cells, a unary mesh scale, ProbeBundle centers, finite coverage rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCompactness}^{\uparrow}$ packet is generated from a closed bounded interval-net row, rational interval cells, dyadic cells, a unary mesh scale, ProbeBundle centers, finite coverage rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2175.1, Definition 2175.2, Definition 2175.3, and Definition 2175.4 bind finite dyadic net compactness for the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCompactness}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DyadicCompactnessUp`

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, Heine–Borel compactness, open-cover compactness, hidden finite-subcover choice, quotient streams, host real equality, selected host limits, or ambient Real completeness.

Upgrade path: Upgrade to scoped finite-net obligations by proving carrier admission, finite-net coverage, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.DyadicCompactnessUp.

Chapter 2176

A Concrete Naming Certificate for $\mathsf{LocatedBoundedRealSet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ packet records the finite located-membership, boundedness, and approximation rows that a supremum-facing consumer may inspect before any least-upper-bound principle is invoked. It sits over the closed interval boundary of chapter 394, the dyadic rational core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the finite source rows consumed by the located supremum route of chapter 1427 and the dyadic supremum boundary of chapter 1653; it does not assert that every bounded located subset has a supremum.

Definition 2176.1 (Located bounded real set carrier)**.** *A* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the located-membership window row. It records the finite membership and apartness requests that have been opened for the displayed real-name family.*
2. *I is the closed interval bound row, carried through* $\mathsf{LocatedInterval}^{\uparrow}$*. It supplies the lower and upper endpoint windows that bound every accepted member row.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *approximation ledger. It records the lower and upper dyadic approximants used by membership, interval containment, and supremum-facing reads.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected member windows, interval endpoint windows, and dyadic comparison windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed member observations and interval endpoint observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located-membership, interval-bound, dyadic, stream-window, and regular-readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the membership window, interval bound, dyadic approximation ledger, stream windows, regular readbacks, real seals, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each set-facing or supremum-facing request through $M, I, D, W, R, E, H, P,$*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSet}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $M, I, D, W, R, E, H, C, P, N$ and has no coordinate for Heine–Borel compactness, arbitrary least-upper-bound completeness, trichotomy, countable choice, host set equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedBoundedRealSetUp.LocatedBoundedRealSetCarrier displayed route`

Theorem 2176.2 (Located bounded real set NameCert obligations)**.** *For every accepted* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *carrier*

$$B = (M, I, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located-membership window, the closed interval bound, the dyadic approximation ledger, the stream windows, the regular readbacks, the real seal, transport, replay, provenance, and local naming row;*
2. *located-membership exposure, requiring every membership or apartness request to enter through the displayed row M and through one of the finite windows named by W;*
3. *interval-bound exposure, requiring every accepted member read to retain the* $\mathsf{LocatedInterval}^{\uparrow}$ *row I before a boundedness conclusion is exported;*
4. *dyadic approximation exactness, requiring every lower or upper approximation read to use the dyadic scales and comparison rows recorded in D;*
5. *regular readback handoff, requiring R to be the only route from the displayed dyadic and stream-window data into regular rational observations;*
6. *real-seal non-escape, requiring E to remain downstream of M, I, D, W, R and to export no completed supremum or host set object;*
7. *transport, continuation, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise.*

No located-bounded-set obligation is stored outside those rows.

Proof. Project the accepted packet $B = (M, I, D, W, R, E, H, C, P, N)$. The carrier exposes the located-membership row, the interval bound, the dyadic ledger, the finite windows, the regular readbacks, and the real seal as the whole analytic surface.

A membership read starts with M and W. The row M supplies the displayed membership or apartness question, while W fixes the finite real-name and endpoint windows in which that question is read.

The boundedness read then passes through I and D. The $\mathsf{LocatedInterval}^{\uparrow}$ row supplies the closed interval boundary, and the dyadic ledger supplies the lower and upper approximation rows used to compare the inspected member data with that boundary.

The regular and real-facing rows are downstream. The row R is the only regular rational handoff, and the row E is exposed only after the membership, interval, dyadic, and window rows have been retained.

Finally H, C, P, N transport, replay, source, and name the same packet. Since Heine–Borel compactness, arbitrary least-upper-bound completeness, trichotomy, countable choice, host set equality, and $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate or replay edge, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.LocatedBoundedRealSetUp.LocatedBoundedRealSetCarrier namecert obligations`

Theorem 2176.3 (Located bounded real set interval-bound handoff)**.** *Every* $\mathsf{LocatedSupremum}^{\uparrow}$ *or* $\mathsf{DyadicSupremum}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *carrier factors through the finite route*

$$M, I \longmapsto D, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives located membership windows, the $\mathsf{LocatedInterval}^{\uparrow}$ *closed bound,* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower and upper approximation rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, package provenance, and local naming. It receives no Heine–Borel compactness, arbitrary least-upper-bound completeness, trichotomy, countable-choice schedule, host set equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.LocatedBoundedRealSetUp.LocatedBoundedRealSetCarrier interval bound handoff`

Proof. Start with a supremum-facing read of the accepted carrier. The packet contains no supremum coordinate, so the read must first enter through the located-membership row M and the closed interval bound row I.

The approximation stage is finite. The dyadic row D records the lower and upper comparison scales, and the stream row W names exactly the inspected member and endpoint windows used by those comparisons.

The readback and seal rows then fix the public handoff. The row R translates the displayed dyadic and stream-window data into regular rational observations, and E seals only that observed finite boundary.

The structural rows preserve this route for both sibling consumers. A $\mathsf{LocatedSupremum}^{\uparrow}$ reader obtains its located family, upper-bound, and lower-approximation source rows from the displayed package, while a $\mathsf{DyadicSupremum}^{\uparrow}$ reader obtains the dyadic bound and approximation rows from D, W, R. No structural row can introduce compactness, completeness, trichotomy, countable choice, or host set equality because those data are absent from the carrier. □

Theorem 2176.4 (Located bounded real set non-escape)**.** *Let* $B = (M, I, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *carrier. Every real-facing, interval-facing, or supremum-facing read exported by* B *is exhausted by the rows* $M, I, D, W, R, E, H, C, P, N$*. In particular, the packet exports a finite bounded located-set source and not a Heine–Borel theorem, an arbitrary least-upper-bound theorem, a trichotomy decision, a countable-choice selector, a host set equality witness, or a* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the accepted packet coordinate by coordinate. The only set-facing coordinate is M, the only bound coordinate is I, the only approximation coordinate is D, and the only finite observation coordinate is W.

The only handoff to regular rational and real-name consumers is R followed by E. Any read that skips these rows has not followed the displayed carrier route, and any read that stays on the route can expose only the rows already named in the packet.

Transport, continuation replay, provenance, and naming do not enlarge the analytic surface. They preserve the same finite rows under hsame, Cont, Pkg, and $\mathsf{NameCert}$, so no escaped compactness, completeness, trichotomy, choice, or host set equality coordinate remains to be read. □

Lean4 ✓ `BEDC.Derived.LocatedBoundedRealSetUp.LocatedBoundedRealSetCarrier nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2176.1, Definition 2176.2, Definition 2176.3, and Definition 2176.4 seed the finite bounded located-set source over LocatedInterval, DyadicRatCore, StreamName, RegSeqRat, Real, LocatedSupremum, and DyadicSupremum consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, arbitrary least-upper-bound completeness, trichotomy, countable choice, host set equality, $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked located-membership exposure, interval-bound exposure, dyadic approximation exactness, regular readback handoff, real-seal non-escape, componentwise transport, continuation replay, provenance, and local naming for BEDC.Derived.LocatedBoundedRealSetUp.

Constructive story: A $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ packet is generated from located-membership windows, a $\mathsf{LocatedInterval}^{\uparrow}$ closed bound, $\mathsf{DyadicRatCore}^{\uparrow}$ lower and upper approximation rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2177

A Concrete Naming Certificate for $\mathsf{ViscositySolution}^{\uparrow}$

Following Crandall–Ishii–Lions viscosity-solutions theory, the $\mathsf{ViscositySolution}^{\uparrow}$ packet records the weak PDE solution interface by finite contact-test data rather than by ambient differentiability of the unknown. It sits over the Sobolev carrier of chapter 440, the Lipschitz-map modulus surface of chapter 410, the metric observation boundary of chapter 57, and the real seal of chapter 38. The packet names the finite viscosity subsolution and supersolution obligation surface needed by PDE-facing consumers without importing existence, uniqueness, comparison principles, classical differentiability, or ambient PDE regularity.

Definition 2177.1 (Viscosity solution carrier)**.** *A* $\mathsf{ViscositySolution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (D, M, U, F, T_-, T_+, B, L, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain row. It records the metric observation windows, boundary faces, interior-window tags, and localization radii read through the* $\mathsf{Met}^{\uparrow}$ *surface.*
2. *M is the regularity row. It exposes the Sobolev-facing and Lipschitz-facing bounds used to keep test functions local to the same displayed domain row.*
3. *U is the candidate scalar row. It records the observed real values, semicontinuity side, boundary trace windows, and finite comparison requests through the* $\mathsf{Real}^{\uparrow}$ *seal.*
4. *F is the finite PDE operator row. It records only the inspected operator values $F(x, r, p, X)$ at named contact jets and has no coordinate for a global host PDE algebra.*
5. *T_- is the subsolution test-function row. Each entry contains a smooth-test name, a local touching-above certificate, the inspected first and second jet data, and the resulting inequality check.*
6. *T_+ is the supersolution test-function row. Each entry contains a smooth-test name, a local touching-below certificate, the inspected first and second jet data, and the resulting inequality check.*
7. *B is the boundary compatibility row. It records boundary-window reads, admissible trace comparisons, and the refusal of uninspected boundary regularity.*

8. *L is the finite localization ledger. It ties every test-function contact, operator read, and boundary read back to the same displayed domain and tolerance windows.*

9. *R records* $\mathsf{Real}^{\uparrow}$ *readback for the candidate values, operator values, comparison inequalities, and boundary traces.*

10. *H records componentwise* hsame *transport for the domain, regularity, candidate, operator, subsolution tests, supersolution tests, boundary, localization, real-readback, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the finite route*

$$D, M, U, F, T_-, T_+, B, L, R \longmapsto H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{ViscositySolution}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{ViscositySolution}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, U, F, T_-, T_+, B, L, R, H, C, P, N$. The carrier has no coordinate for solution existence, uniqueness, comparison principles, classical differentiability of the candidate, global PDE regularity, host function equality, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.ViscositySolutionUp.ViscositySolutionUp`

Theorem 2177.2 (Viscosity solution NameCert obligations)**.** *For every accepted* $\mathsf{ViscositySolution}^{\uparrow}$ *carrier*

$$V = (D, M, U, F, T_-, T_+, B, L, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ViscositySolution}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the domain, regularity, candidate scalar, PDE operator, subsolution test, supersolution test, boundary, localization, real-readback, transport, replay, provenance, and local naming rows;*

2. *common-domain discipline, requiring every Sobolev, Lipschitz, metric, candidate-value, operator, and boundary read to pass through the same displayed domain row D;*

3. *contact-test exactness, requiring each entry of T_- and T_+ to record the touching side, localization window, inspected jet data, operator value, and inequality sign for that contact;*

4. *boundary compatibility, requiring B to expose only the named trace windows and comparison rows before any boundary-facing consumer may read the packet;*

5. *localization stability, requiring L to bind every contact test and boundary read to the same finite metric windows and tolerance rows;*

6. *real-readback nonescape, requiring R to expose only the displayed candidate values, operator values, comparison inequalities, and trace rows;*

7. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no solution existence theorem, uniqueness theorem, comparison principle, classical differentiability witness, global PDE regularity, host function equality, Lean verification, or bridge credential.*

No viscosity-solution obligation is stored outside these rows.

Proof. Project the accepted packet $V = (D, M, U, F, T_-, T_+, B, L, R, H, C, P, N)$. The carrier definition lists the entire visible packet, so carrier admission is projection from the displayed finite BHist data.

The common-domain discipline is forced by the source row D. The Sobolev and Lipschitz regularity row M, the scalar candidate row U, the operator row F, and the boundary row B all refer back to the same metric observation windows and localization radii.

The contact obligations are exactly the two test rows. A subsolution read must pass through T_- and therefore displays a touching-above certificate, inspected jet data, the operator value, and the required inequality. A supersolution read is the dual read through T_+ with touching-below data and its displayed inequality.

Boundary and localization data are finite ledgers. The row B exposes only named trace comparisons, while L binds each interior or boundary contact to the same domain windows and tolerance data.

Finally R, H, C, P, N read back, transport, replay, source, and name those rows componentwise. Since the excluded analytic principles have no carrier coordinate and no replay edge, the listed rows exhaust the local NameCert surface. □

Theorem 2177.3 (Viscosity solution test-function boundary)**.** *Every PDE-facing consumer read of an accepted* $\mathsf{ViscositySolution}^{\uparrow}$ *carrier factors through the finite route*

$$D, M, U, F, T_-, T_+, B, L, R \longmapsto H, C, P, N.$$

Thus the consumer receives the displayed metric domain windows, Sobolev and Lipschitz regularity rows, candidate real-value observations, finite PDE-operator values, subsolution and supersolution contact-test ledgers, boundary trace comparisons, localization data, real-readback rows, componentwise transport, continuation replay, package provenance, and local naming. It receives no solution existence theorem, uniqueness theorem, comparison principle, classical differentiability of the candidate, ambient PDE regularity, or host function equality.

Proof. Start with a PDE-facing read of the accepted packet. The domain row D is the first admissible source because every contact test, boundary trace, and operator value is localized over its metric observation windows.

The regularity and candidate rows are then fixed. The row M supplies only Sobolev-facing and Lipschitz-facing bounds, while U supplies the observed real values and semicontinuity side used by the contact tests.

The operator and test-function rows form the viscosity boundary. The row F supplies only inspected operator values at named contact jets, and the rows T_- and T_+ expose the touching side, jet data, and inequality signs required for subsolution and supersolution reads.

Boundary traces cannot bypass the contact route. The row B contributes named trace comparisons, and L ties them to the same localization windows before R exposes the corresponding real-readback inequalities.

The structural rows H, C, P, N preserve exactly this dependency order. Therefore every consumer read factors through the finite route above and cannot obtain existence, uniqueness, comparison, differentiability, global regularity, or host equality data from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ViscositySolution}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ViscositySolution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2177.1, Definition 2177.2, and Definition 2177.3 seed the finite viscosity-solution packet over metric domain windows, Sobolev and Lipschitz regularity rows, real candidate observations, PDE operator reads, contact-test ledgers, boundary traces, localization, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ViscositySolution}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close solution existence, uniqueness, comparison principles, classical differentiability, ambient PDE regularity, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-domain discipline, contact-test exactness, boundary compatibility, localization stability, real-readback nonescape, and provenance exhaustion for BEDC.Derived.ViscositySolutionUp

Constructive story: A ViscositySolution$^{\uparrow}$ packet is generated from metric domain windows, Sobolev and Lipschitz regularity rows, candidate RealUp observations, finite PDE-operator values, subsolution and supersolution contact-test ledgers, boundary trace comparisons, localization rows, RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2178

A Concrete Naming Certificate for $\mathsf{PreReal}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy construction of the reals, the $\mathsf{PreReal}^{\uparrow}$ packet records the unsealed regular-Cauchy name boundary between regular rational sequences and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits over the dyadic tolerance ledger of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the real-name boundary of chapter 38. The packet exposes the finite name data available before a Real seal is consumed; it is not quotient stream equality, a selected limit, host real equality, countable choice, or completeness.

Definition 2178.1 (PreReal carrier)**.** *A* $\mathsf{PreReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (D, S, R, H, C, K, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording finite precision requests, denominator windows, and dyadic radius bookkeeping.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row opened by the same tolerance request, recording the finite schedule of inspected name entries.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, read only through D and S, and carrying the displayed Cauchy regularity evidence available before a real seal is consumed.*
4. *H records componentwise* hsame *transport for tolerance, observation, regularity, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay of the route*

$$D, S \longmapsto R \longmapsto H, K, N.$$

6. *K is the* Pkg *provenance over* $\mathsf{PreReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected representatives, host real equality, countable choice, a selected limit, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PreRealUp.PreRealTasteGate single carrier alignment`

Theorem 2178.2 (PreReal NameCert obligations)**.** *For an accepted* $\mathsf{PreReal}^{\uparrow}$ *carrier*

$$P = (D, S, R, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance, StreamName observation, RegSeqRat regularity, componentwise transport, replay, provenance, and local naming rows;*
2. *tolerance-window exposure, requiring every PreReal read to start from the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *and the matching* $\mathsf{StreamName}^{\uparrow}$ *row* S*;*
3. *regularity handoff, requiring the opened finite name window to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before any Real-facing boundary is named;*
4. *transport and replay stability, requiring* H *and* C *to preserve the same displayed rows componentwise;*
5. *provenance non-escape, requiring* K *and* N *to export no quotient streams, selected representatives, host real equality, countable choice, selected limit, or completeness principle.*

No PreReal obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2178.1. The packet lists the tolerance row, finite observation row, regularity row, transport, replay, provenance, and local naming row, so admission has no additional source.

The tolerance-window exposure obligation reads D and S first. These two rows bind the same finite precision request, so a PreReal consumer cannot skip directly to a selected limit or quotient stream.

The regularity handoff is row R. It consumes the opened finite window through the regular rational sequence surface, and therefore presents only regular Cauchy name data to any later Real-facing boundary.

Finally H, C, K, N transport, replay, source, and name exactly the displayed route. Since the carrier contains no coordinate for the excluded host objects or choice principles, the listed rows exhaust the local NameCert surface. □

Theorem 2178.3 (PreReal constant dyadic route)**.** *Every constant dyadic name admitted by a* $\mathsf{PreReal}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{PreReal}^{\uparrow}.$$

For a carrier $P = (D, S, R, H, C, K, N)$*, the route reads the constant dyadic tolerance data from* D*, opens the matching finite StreamName observation row* S*, records regularity in* R*, and then consumes only* H, C, K, N *as structural support for the local* $\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}$ *row.*

Proof. Project the accepted packet $P = (D, S, R, H, C, K, N)$. The constant dyadic request first reads D, because this is the only tolerance row exposed by the carrier.

The matching StreamName observation row is S. It records the finite schedule at which the constant dyadic data are inspected, and it supplies no selected limit or quotient coordinate.

The regularity evidence is then read from R. A constant dyadic name satisfies the regular rational route by the displayed tolerance and observation rows, so the PreReal packet receives a regular name rather than host real data.

The remaining rows H, C, K, N preserve, replay, source, and name the same route. Thus the constant dyadic route reaches $\mathsf{PreReal}^{\uparrow}$ without exporting quotient streams, selected limits, host real equality, countable choice, or Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}$.

Constructive story: A $\mathsf{PreReal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName finite observation row, RegSeqRat regularity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any RealUp seal is consumed.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2178.1, Definition 2178.2, and Definition 2178.3 seed the unsealed regular-Cauchy name boundary between RegSeqRat rows and RealUp seal consumption.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PreReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter refuses quotient streams, selected limits, host real equality, countable choice, selected representatives, and completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exposure, regularity handoff, transport and replay stability, provenance non-escape, and constant dyadic routing for BEDC.Derived.PreRealUp.

Chapter 2179

A Concrete Naming Certificate for $\mathsf{BishopSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopSequence}^{\uparrow}$ packet records a constructive sequence of real-facing approximants as finite BEDC NameCert data. It sits beside the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the explicit convergence row of chapter 332, and the terminal real-name seal of chapter 38. The packet keeps the sequence surface visible to later Real and Completion consumers before any $\mathsf{Real}^{\uparrow}$ seal is read; it does not quotient streams, choose limits, invoke countable choice, or assert ambient completeness.

Definition 2179.1 (Bishop sequence carrier)**.** *A* $\mathsf{BishopSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, M, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It records the finite sequence indices, inspected windows, and read schedule exposed to the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational sequence values read from the same windows.*
3. *M is the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row. It records the displayed threshold discipline by which later tail requests are answered.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic radii, shrink steps, denominator budgets, and comparison cells attached to S, R, M.*
5. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row. It records only the finite real-facing boundary reached after the regular readback and modulus rows have been displayed.*
6. *H records componentwise* hsame *transport for stream observations, regular readback, convergence modulus, dyadic tolerance ledger, real seal data, continuation replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto R, M \longmapsto Q \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, M, D, Q, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BishopSequenceUp.BishopSequenceTasteGate single carrier alignment`

Theorem 2179.2 (Bishop sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopSequence}^{\uparrow}$ *carrier*

$$B = (S, R, M, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream windows, regular rational readback, convergence modulus, dyadic tolerance ledger, real seal boundary, transport, replay, provenance, and local naming row;*
2. *observation visibility, requiring every sequence index read to pass through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows in S;*
3. *regular readback, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume the same finite observations before any real-facing consumer reads the packet;*
4. *modulus accountability, requiring the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *row M to expose the threshold used by any tail or completion request;*
5. *tolerance exactness, requiring every threshold and readback comparison to be witnessed by a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *entry in D;*
6. *real seal confinement, requiring Q to expose only the finite* $\mathsf{Real}^{\uparrow}$ *boundary reached through R, M, D;*
7. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the rows listed above and no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, ambient completeness, Lean verification, or bridge credential.*

No Bishop-sequence obligation is stored outside these rows.

Proof. Project the accepted packet $B = (S, R, M, D, Q, H, C, P, N)$. The carrier definition lists every visible row, so admission is exactly the displayed finite BHist data together with its transport, replay, provenance, and naming rows.

The observation and readback rows are forced first. The schedule and finite windows live in S, and the regular rational values live in R, so a consumer has no route to an unbounded stream or to a host sequence representative.

The convergence discipline is the row M together with the dyadic ledger D. Every threshold request must read the displayed modulus row and the displayed tolerance entries before it can be presented to a completion-facing consumer.

The real-facing boundary is last. The row Q records only the $\mathsf{Real}^{\uparrow}$ seal reached through the regular readback, convergence modulus, and tolerance ledger.

Finally H, C, P, N preserve exactly this route under componentwise transport, continuation replay, provenance, and local naming. Since the refused principles have no coordinate and no replay edge, the listed obligations exhaust the local NameCert surface. □

Theorem 2179.3 (Bishop sequence StreamName–Real route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{BishopSequence}^{\uparrow}$ *carrier factors through the finite route*

$$S, D \longmapsto R, M \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer receives the $\mathsf{StreamName}^{\uparrow}$ *observation windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{ModulusOfConvergence}^{\uparrow}$ *threshold row, the* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It receives no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Start with a consumer read of the accepted carrier. The read begins at S, where the packet displays the finite sequence indices, observation windows, and schedule.

The tolerance row D is read with the same windows. It supplies the dyadic radii, shrink steps, denominator budgets, and comparison cells needed for both regular readback and convergence thresholds.

The middle handoff is R, M. The $\mathsf{RegSeqRat}^{\uparrow}$ row exposes the regular rational sequence data, while the $\mathsf{ModulusOfConvergence}^{\uparrow}$ row exposes the threshold discipline used by tail and completion requests.

The real boundary is then Q. It is reachable only after R, M, D have been displayed, so a $\mathsf{Real}^{\uparrow}$ consumer reads a seal rather than a selected host limit.

The structural rows H, C, P, N keep the same route stable under transport, replay, provenance, and local naming. Therefore every listed consumer factors through the displayed finite route and cannot access any excluded principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2179.1, Definition 2179.2, and Definition 2179.3 seed the finite constructive sequence packet over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ModulusOfConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observation visibility, regular readback, modulus accountability, tolerance exactness, real seal confinement, provenance exhaustion, and local naming for BEDC.Derived.BishopSequenceUp.

Constructive story: A $\mathsf{BishopSequence}^{\uparrow}$ packet is generated from StreamName observation windows, RegSeqRat readback, a ModulusOfConvergence threshold row, a DyadicRatCore tolerance ledger, RealUp seal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2180

A Concrete Naming Certificate for $\mathsf{CauchyRealEquality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRealEquality}^{\uparrow}$ packet records equality of constructive Cauchy reals as a finite regular-name classifier surface. It sits after the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the sealed real boundary of chapter 38. The packet lets Real-facing consumers compare two displayed regular Cauchy names without importing quotient stream equality, host real equality, selected representatives, countable choice, or an ambient completeness theorem.

Definition 2180.1 (Cauchy real equality carrier)**.** *A* $\mathsf{CauchyRealEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L_0, L_1, S_0, S_1, R_0, R_1, D, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two* $\mathsf{Real}^{\uparrow}$ *seal rows. They record only the terminal real-name boundaries reached by the two compared regular names.*
2. S_0 *and* S_1 *are the paired* $\mathsf{StreamName}^{\uparrow}$ *window rows. They expose the finite observation schedules inspected on the two sides.*
3. R_0 *and* R_1 *are the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows. Each readback consumes its own stream window before the corresponding* $\mathsf{Real}^{\uparrow}$ *seal is available.*
4. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic precision requests and comparison budgets used to test the two regular readbacks at the same requested tolerance.*
5. H *records componentwise* hsame *transport for both real seals, both stream windows, both regular readbacks, the dyadic ledger, continuation replay, provenance, and local naming.*
6. C *records displayed* Cont *replay of the comparison route*

$$S_0, S_1, D \longmapsto R_0, R_1 \longmapsto L_0, L_1 \longmapsto H, P, N.$$

7. P *is the* Pkg *provenance over* $\mathsf{CauchyRealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealEquality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. The carrier has no coordinate for quotient stream equality, host real equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyRealEqualityUp.TasteGate.CauchyRealEqualityTasteGate single carrier alignment`

Theorem 2180.2 (Cauchy real equality NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealEquality}^{\uparrow}$ *carrier*

$$E = (L_0, L_1, S_0, S_1, R_0, R_1, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealEquality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{Real}^{\uparrow}$ *seals, paired stream windows, paired regular readbacks, shared dyadic tolerance ledger, transport, replay, provenance, and local naming row;*
2. *two-sided window exposure, requiring every equality read to inspect* S_0 *and* S_1 *before reaching either regular readback;*
3. *regular-readback comparison, requiring* R_0 *and* R_1 *to be compared only at tolerance requests named by* D*;*
4. *Real-seal nonescape, requiring* L_0 *and* L_1 *to be read only after the paired regular readbacks and never as host real representatives;*
5. *provenance exhaustion, requiring* H, C, P, N *to transport, replay, source, and name exactly the displayed rows and no quotient stream equality, host real equality, selected representative, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge credential.*

No Cauchy-real-equality obligation is stored outside these rows.

Proof. Project the accepted carrier. Its displayed rows are the whole finite packet, so carrier admission is exactly the listed pair of real seals, the two stream windows, the two regular rational readbacks, the shared dyadic tolerance ledger, and the four structural rows.

The comparison first reads the windows S_0 and S_1. Those rows name the finite observations available on each side, so an equality consumer cannot begin from an unbounded stream.

The readbacks R_0 and R_1 are then compared only through D. The dyadic ledger supplies the common precision budget and prevents the comparison from appealing to an ambient metric or a host real value.

Finally L_0, L_1, H, C, P, N close the route. The real seals are terminal rows reached through the regular readbacks, while transport, replay, provenance, and local naming preserve the same coordinates componentwise. Since the refused principles have no coordinate and no replay edge, the listed obligations exhaust the local certificate surface. □

Theorem 2180.3 (Cauchy real equality RegSeqRat classifier handoff)**.** *Every classifier comparison exported by an accepted* $\mathsf{CauchyRealEquality}^{\uparrow}$ *packet factors through the finite route*

$$S_0, S_1, D \longmapsto R_0, R_1 \longmapsto L_0, L_1 \longmapsto H, C, P, N.$$

Thus the consumer receives paired $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the two* $\mathsf{Real}^{\uparrow}$ *seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It receives no quotient equality of streams, host real equality, selected representative, countable-choice schedule, or ambient completeness witness.*

Proof. Start with an exported classifier comparison. The source rows are the paired stream windows S_0 and S_1, so the comparison is anchored at finite observation schedules rather than at total streams.

Next read the shared dyadic ledger D and the paired regular readbacks R_0, R_1. The ledger supplies the tolerance requests, and the readbacks supply the only rational sequence data visible to the classifier.

The terminal real-facing data are L_0 and L_1. These rows are seals reached through R_0 and R_1, not host real values and not quotient representatives.

The structural rows H, C, P, N preserve the same route by componentwise transport, replay, provenance, and local naming. Therefore the classifier handoff is exactly the displayed RegSeqRat-to-Real route and cannot export any of the excluded equality or choice data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealEquality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealEquality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2180.1, Definition 2180.2, and Definition 2180.3 seed the finite equality-classifier packet over two Real seals, two StreamName windows, two RegSeqRat readbacks, one DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealEquality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-sided window exposure, regular-readback comparison, Real-seal nonescape, and provenance exhaustion for BEDC.Derived.CauchyRealEqualityUp.

Constructive story: A $\mathsf{CauchyRealEquality}^{\uparrow}$ packet is generated from two RealUp seal rows, paired StreamName observation windows, paired RegSeqRat regular readbacks, a shared DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2181

A Concrete Naming Certificate for MetaCICResidualSubstitutionBoundary$^{\uparrow}$

The MetaCICResidualSubstitutionBoundary$^{\uparrow}$ packet realizes the residual-substitution compatibility boundary of chapter 9378 as finite NameCert data. It sits beside the confluence audit packet of chapter 735, the closed-normal fragment of chapter 737, and the substitution audit surface of chapter 1102. The downstream parallel-diamond frontier of chapter 4238 consumes this packet specifically as the visible residual-substitution premise row while retaining the conditional boundary. The packet names exactly the row needed by the conditional parallel-diamond handoff without turning the checked fragments into unconditional MetaCIC confluence.

Definition 2181.1 (MetaCIC residual-substitution boundary carrier)**.** *A* MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, S, A, J, N, F, L, H, C, P, K)$$

with the following displayed rows:

1. *R is the residual beta-branch row. It records the two residual branches whose componentwise join is blocked without substitution compatibility.*
2. *S is the substitution-compatibility boundary row. It names the* BetaParallelSubstitutionCompatibility *input as a required row, not as a theorem produced by the packet.*
3. *A is the application-branch audit row. It records the beta-application shape where the substitution boundary is consumed by the parallel diamond route.*
4. *J is the conditional join row. It records the route from S and A to the parallel-diamond handoff while keeping the premise visible.*
5. *N is the closed-normal fragment row, read through chapter 737 as an available sibling fragment rather than as a discharge of S.*
6. *F is the confluence audit packet row, read through chapter 735 as the larger audit context that consumes this boundary.*
7. *L is the substitution audit ledger, read through chapter 1102, recording which closed substitution routes are audited before a consumer cites S.*
8. *H records componentwise* hsame *transport for residual branches, substitution boundary, application audit, conditional join, sibling fragment, audit packet, ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$R, S, A, L \longmapsto J \longmapsto F, N \longmapsto H, P, K.$$

10. *P is the* Pkg *provenance over* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*, MetaCIC beta reduction,* BetaParallel*,* BetaStarStep*, closedness rows, the named sibling packets,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *K is the local* NameCert$_{\text{MetaCICResidualSubstitutionBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate proving BetaParallelSubstitutionCompatibility*, no coordinate proving full parallel diamond without S, no coordinate proving arbitrary closed Church–Rosser, and no coordinate for host substitution semantics.*

Theorem 2181.2 (MetaCIC residual-substitution boundary NameCert obligations)**.** *For every accepted* MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *carrier*

$$B = (R, S, A, J, N, F, L, H, C, P, K),$$

the local NameCert$_{\text{MetaCICResidualSubstitutionBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the residual beta branches, substitution-compatibility boundary, application-branch audit, conditional join, closed-normal fragment, confluence audit packet, substitution audit ledger, transport, replay, provenance, and local naming row;*
2. *residual-row exposure, requiring a consumer to read the beta residual branches R before citing the boundary;*
3. *substitution-boundary visibility, requiring S to remain a named premise row whenever the conditional join J is exported;*
4. *sibling separation, requiring N and F to be read as contextual fragments and audit packets rather than as proofs of S;*
5. *ledger exactness, requiring L to record the audited closed substitution rows visible to the boundary;*
6. *provenance exhaustion, requiring H, C, P, K to transport, replay, source, and name exactly the displayed rows and no proof of* BetaParallelSubstitutionCompatibility*, no premise-free parallel diamond, no arbitrary closed Church–Rosser theorem, and no host substitution semantics.*

No residual-substitution-boundary obligation is stored outside these rows.

Proof. Project the accepted carrier $B = (R, S, A, J, N, F, L, H, C, P, K)$. The carrier definition lists the whole finite packet, so carrier admission is projection from the displayed rows.

The residual branch data are read first. Row R records the beta branches whose join requires substitution compatibility, and row A records the application shape where the branch interaction occurs.

The boundary row S remains visible through the conditional join J. Since S is a premise row rather than a produced theorem, the packet can expose the conditional handoff but cannot erase the input.

Rows N and F keep the siblings separate. The closed-normal fragment and the confluence audit packet are useful context, but neither contains a projection that proves the residual substitution boundary row.

The ledger row L records the audited closed substitution data that may be cited before the boundary is consumed. Finally H, C, P, K transport, replay, source, and name exactly those rows. With no coordinate for the refused endpoint claims, the listed obligations exhaust the local NameCert surface. □

Theorem 2181.3 (MetaCIC residual-substitution premise preservation)**.** *Let $B = (R, S, A, J, N, F, L, H, C, P, K)$ be an accepted* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *carrier. Any consumer of B that exports a parallel-diamond handoff or a closed Church–Rosser handoff must keep the substitution row S visible as a premise through the route*

$$R, S, A, L \longmapsto J \longmapsto F, N \longmapsto H, P, K.$$

Thus the consumer may export only a conditional handoff whose displayed route still contains S; it does not obtain $\mathsf{BetaParallelSubstitutionCompatibility}$ *as a theorem of the packet.*

Proof. Project the carrier of Definition 2181.1. The rows R, S, A, L are the only entry surface for the residual beta branches, the substitution-compatibility boundary, the application audit, and the substitution audit ledger.

Apply Definition 2181.2. Its substitution-boundary visibility clause requires S to remain a named premise whenever the conditional join row J is exported, while its ledger exactness clause keeps L attached to the audited closed substitution rows.

The sibling separation rows then route the handoff through the confluence audit packet F and the closed-normal fragment N without turning either sibling into a proof of S. The confluence audit map may consume the conditional route, but the visible premise row remains part of the exported packet.

Finally H, P, K transport, source, and name the same finite route. Since the carrier has no coordinate proving $\mathsf{BetaParallelSubstitutionCompatibility}$, a parallel-diamond or closed Church–Rosser consumer can receive only the premise-preserving handoff displayed above. □

Theorem 2181.4 (MetaCIC residual-substitution closed-face consumer)**.** *Let $B = (R, S, A, J, N, F, L, H, C, P, K)$ be an accepted* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *carrier. Any closed-face confluence consumer or subject-reduction consumer of B must keep the substitution row S and the closed-face audit ledger L visible through the route*

$$R, S, A, L \longmapsto J \longmapsto F, N \longmapsto H, P, K.$$

The sibling confluence-audit packet F and closed-normal row N may be used as context for the closed face, but neither sibling discharges S or turns the route into unrestricted substitution compatibility.

Proof. Start with Definition 2181.2. Its substitution-boundary visibility clause keeps S named whenever the conditional join J is exported, and its ledger exactness clause keeps L attached to the audited closed substitution rows.

Apply Definition 2181.3. It routes any parallel-diamond or closed Church–Rosser handoff through

$$R, S, A, L \longmapsto J \longmapsto F, N \longmapsto H, P, K.$$

The closed-face consumer may inspect the sibling confluence-audit row F and the closed-normal row N only after the visible premise route has reached J. Those rows provide closed-face context, while S remains the named substitution-compatibility premise.

Finally H, C, P, K transport, replay, source, and name the same route. Since the carrier has no coordinate proving the substitution row, no closed-face consumer can erase S or obtain unrestricted substitution compatibility from the sibling context. □

Falsifiable prediction: A packet lacking the residual-substitution row is rejected even if it carries closed-normal confluence rows.

Independence witness: metacicconfluenceauditpacket, metacicconfluencenormalfragment, metacicredexfrontier, closedsubstitutionboundary, substitutionboundaryaudit: the residual-substitution boundary carrier is not bijective to any sibling because it carries the dedicated premise row S coupled to the conditional join row J and the residual beta-branch row R.

The proposed Lean follow-up supplies Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/MetaCICRe` by giving two explicit boundary packets, one with the residual-substitution row present and one refused by that row, and by matching on the fields $R, S, A, J, N, F, L, H, C, P, K$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}}$ ·

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2181.1 and Definition 2181.2 seed the finite residual-substitution boundary packet over residual beta branches, the visible substitution premise row, application audit, conditional join, sibling fragments, substitution ledger, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close BetaParallelSubstitutionCompatibility, premise-free parallel diamond, arbitrary closed Church–Rosser, host substitution semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-row exposure, substitution-boundary visibility, sibling separation, ledger exactness, provenance exhaustion, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICResidualSubstitutionBoundaryUp.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ packet is generated from residual beta branches, a visible substitution-compatibility premise row, application audit data, conditional join data, closed-normal and confluence-audit sibling rows, a substitution audit ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2182

A Concrete Naming Certificate for $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$

The $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ packet realizes the MetaCIC decidability audit map of chapter 9379 as a finite checker seal. It sits beside the subject-reduction obstruction of chapter 9377 and the confluence audit map of chapter 9378: those maps expose preservation and residual-substitution boundaries, while this packet records the closed typing-checker surface that can be consumed without promoting bounded normal equality into unrestricted beta-conversion decidability.

Definition 2182.1 (Closed type-checker seal carrier)**.** *A* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, T, Q, E, N, R, O, H, K, P, L)$$

with the following displayed rows:

1. *C is the closed input row, carrying the represented closed MetaCIC term and the displayed closed context boundary;*
2. *T is the* `inferTypeCtxClosed` *checker row, including the soundness and completeness surfaces for the represented* `HasType` *relation;*
3. *Q is the structural* `Term.eq` *and* `checkSameTerm` *row used by the checker;*
4. *E is the checker-exactness row tying successful checking to the represented typing judgement and tying represented typing back to checker success;*
5. *N is the bounded normal-equality row over* `normalizeBounded` *and* `decideClosedBetaConv`*;*
6. *R is the finished-normalization boundary row required before a bounded-normal comparison is read as a statement about named finished normal forms;*
7. *O is the non-escape row excluding unrestricted closed beta-conversion decidability and any standalone closedness-decision theorem not displayed in the packet;*
8. *H records componentwise* hsame *transport across the closed input, checker, equality, exactness, bounded-normal, boundary, obstruction, replay, provenance, and local naming rows;*
9. *K records displayed* Cont *routes for closed typing checks, structural equality readback, bounded-normal comparison, finished-boundary replay, and non-escape readback;*

10. *P is the* Pkg *provenance over the decidability audit map, the checker seal of chapter 747, the decidability audit packet of chapter 746, the bounded normal-equality checker of chapter 752, the residual-substitution boundary of chapter 2181,* BHist, hsame, Cont, *and* NameCert *rows;*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{ClosedTypeCheckerSeal}^{\uparrow}}$ *row.*

The carrier exposes closed typing-checker exactness and bounded normal-equality data only through the displayed rows. It has no coordinate for unrestricted beta-conversion decidability, no hidden normalization endpoint, and no host-side closed-term oracle.

Theorem 2182.2 (Closed type-checker seal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ClosedTypeCheckerSeal}^{\uparrow}}$ *surface consists of six obligation rows:*

1. *carrier: formation of the finite packet of Definition 2182.1;*
2. *checker exactness: the* `inferTypeCtxClosed` *row exposes soundness and completeness for the represented typing relation through T and E;*
3. *equality readback: structural term equality and same-term checking are read only through Q;*
4. *bounded normal equality: bounded-normal comparison is read only through N and the finished-normalization boundary row R;*
5. *non-escape: unrestricted closed beta-conversion decidability and standalone closedness decidability are refused by O;*
6. *ledger exactness: every transport, replay, provenance, and local naming read factors through H, K, P, L.*

Proof. Project the carrier tuple of Definition 2182.1. The visible checker data are C, T, Q, E, N, R, O, and the structural replay data are H, K, P, L.

The checker-exactness row uses the positive part of chapter 9379. The closed wrapper `inferTypeCtxClosed` supplies the typing-checker soundness and completeness surface, while E records that this surface is the exact public route for represented typing.

The equality and bounded-normal rows stay separate. Structural term equality and same-term checking are consumed through Q, while bounded normal equality is consumed through N and becomes a finished-normal statement only when R is displayed.

The non-escape row O records the missing endpoint. Because chapter 9377 and chapter 9378 keep their own preservation and residual-substitution boundaries visible, this seal cannot erase those boundaries by citing a checker row.

Finally H, K, P, L transport, replay, source, and name exactly the displayed rows. No additional carrier coordinate exists, so the listed rows exhaust the local NameCert obligations. □

Theorem 2182.3 (Closed type-checker seal bounded endpoint boundary)**.** *Every accepted* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *carrier permits downstream consumers to read checked closed typing, structural term equality, same-term checking, bounded normal equality, finished-normalization consequences when their witness row is displayed, and the corresponding transport, replay, provenance, and naming rows. It does not permit a consumer to export unrestricted closed beta-conversion decidability without a finished-normalization witness.*

Proof. Read the positive checker routes first. Rows C, T, Q, E expose the closed input boundary, the closed type-checker result, the structural equality row, and the exactness relation between checking and represented typing.

Next read the bounded-normal route. Row N compares bounded normalizer outputs, and row R is the only route that turns the bounded comparison into a finished-normal statement.

The obstruction row O is part of the same carrier. It refuses any consumer route that omits R while claiming unrestricted closed beta-conversion decidability.

Thus the seal separates checked typing decidability from the bounded conversion boundary. The public endpoint is the finite checker packet, not a global conversion oracle. □

Theorem 2182.4 (Closed type-checker seal non-escape)**.** *The public boundary of* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *exports only the finite rows*

$$C, T, Q, E, N, R, O, H, K, P, L.$$

It exports no subject-reduction theorem, no confluence theorem, no residual-substitution discharge, no unrestricted closed beta-conversion decider, no hidden normalization endpoint, and no host equality oracle.

Proof. Inspect the accepted carrier. Its checker-facing rows are C, T, Q, E, N, R, O, and its structural rows are H, K, P, L.

Apply Definition 2182.2. Every public read projects through the carrier, checker-exactness, equality-readback, bounded-normal, non-escape, or ledger-exactness rows.

Apply Definition 2182.3. The bounded-normal route reaches a finished-normal statement only when the witness row is displayed, and the same route refuses unrestricted beta-conversion decidability without that witness.

Since no coordinate names subject reduction, confluence, residual-substitution discharge, a hidden normalizer endpoint, or a host equality oracle, none of those objects can escape the seal. □

Theorem 2182.5 (Closed type-checker seal residual-substitution forward link)**.** *Every accepted* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *packet that exports checked closed typing and bounded normal-equality data to a residual-substitution or confluence-facing consumer must pass through the displayed residual-substitution boundary and the finished-normalization row R. Without that row the packet cannot export subject reduction, confluence, residual-substitution discharge, unrestricted closed beta-conversion decidability, or global normalization.*

Proof. Project the carrier rows of Definition 2182.1. The checker-facing export is carried by C, T, Q, E, N, while the finished-normalization boundary is the separate row R and the non-escape row is O.

Apply Definition 2182.3. It permits bounded normal equality to become a finished-normal statement only when the witness row is displayed, so a consumer that omits R receives checked typing and bounded comparison data but no normalization endpoint.

Apply Definition 2182.4. The same carrier has no coordinate for subject reduction, confluence, residual-substitution discharge, unrestricted beta-conversion decidability, or global normalization.

The residual-substitution boundary of chapter 2181 is therefore the sibling route through which a confluence-facing consumer must pass before reading this seal as part of a residual-substitution argument. The seal forwards bounded checker data to that route; it does not replace the route by a global normalization theorem. □

Theorem 2182.6 (Closed type-checker seal substitution-frontier handoff)**.** *Every accepted* $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ *packet used by a confluence-facing consumer must expose the closed input row C, the* `inferTypeCtxClosed` *exactness rows T, E, the bounded-normal row N, the finished-normalization boundary row R, and the residual-substitution boundary before any confluence or subject-reduction read is consumed. The packet may feed the residual-substitution route of chapter 2181 and the conditional confluence map*

of chapter 9378, but it does not discharge residual substitution, subject reduction, full confluence, unrestricted beta-conversion decidability, or global normalization.

Proof. Project the carrier of Definition 2182.1. The closed input, checker, exactness, bounded-normal, finished-boundary, and non-escape rows are displayed as C, T, E, N, R, O.

Apply the non-escape theorem Definition 2182.4. It prevents the seal from exporting subject reduction, confluence, residual-substitution discharge, unrestricted beta-conversion decidability, hidden normalization endpoints, or host equality.

Then route the confluence-facing read through chapter 2181 and chapter 9378. Those sibling surfaces keep the residual-substitution frontier visible before any confluence or subject-reduction consumer can cite the checker seal. □

Falsifiable prediction: The seal has no carrier route exporting unrestricted closed beta-conversion decidability without a finished-normalization witness.

Independence witness: metacicdecidableboundary, metacicdecidableauditpacket, metacicdecidablecheckerseal, metacic-residual-substitution-boundary: the closed type-checker seal is not bijective to any listed sibling because it binds the closed input row, inferTypeCtxClosed exactness row, structural equality readback, bounded-normal row, finished-normalization boundary, and non-escape row in one finite carrier.

The Lean follow-up supplies Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/ClosedTypeChecker` by exhibiting distinct accepted and refused closed checker packets and by matching on the rows $C, T, Q, E, N, R, O, H, K, P, L$.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedTypeCheckerSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$ packet is generated from a closed input row, inferTypeCtxClosed checker row, structural Term.eq and checkSameTerm row, checker-exactness row, bounded normal-equality row, finished-normalization boundary row, non-escape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedTypeCheckerSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2182.1, Definition 2182.2, Definition 2182.3, and Definition 2182.4 seed the finite closed type-checker seal and its bounded conversion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedTypeCheckerSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close subject reduction, confluence, residual-substitution discharge, unrestricted closed beta-conversion decidability, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inferTypeCtxClosed exactness routing, equality readback, bounded-normal boundary routing, non-escape, and TasteGate nontriviality with field faithfulness for BEDC.Derived.ClosedTypeChec

Chapter 2183

A Concrete Naming Certificate for $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ packet records a bounded located sequence of real names as finite BEDC NameCert data before any compactness, supremum, or completeness principle is available. It sits after the real seal of chapter 38, the located real boundary of **??**, the regular rational readback surface of chapter 301, the finite stream-name discipline of chapter 126, and the dyadic rational core of chapter 291. The packet gives L10 Real and Completion consumers one named sequence-boundary object whose rows can later feed limit, Cauchy, and boundedness arguments without importing Heine–Borel compactness, sequential compactness, host suprema, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2183.1 (Located bounded real sequence carrier)**.** *A* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, L, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected-index row. It records the finite sequence positions and tail windows opened by the current consumer.*
2. *L is the located-real row. It records the* $\mathsf{LocatedReal}^{\uparrow}$ *membership and apartness windows attached to the inspected entries.*
3. *U is the boundedness row. It records the displayed lower and upper real-boundary windows that bound every inspected entry.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger. It records the dyadic radii, shrink steps, lower-bound comparisons, upper-bound comparisons, and tolerance budgets used by L and U.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row. It names the finite stream windows for the inspected indices, located-real reads, bound reads, and dyadic comparisons.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records the regular rational observations obtained from W and D before any real-facing seal is exported.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the real-name boundary reached through L, U, D, W, R.*

8. *H records componentwise* hsame *transport for the index row, located-real row, boundedness row, dyadic ledger, stream windows, regular readbacks, real seal, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay of the finite route*

$$I, L, U \longmapsto D, W \longmapsto R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSequence}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $I, L, U, D, W, R, E, H, C, P, N$ and has no coordinate for Heine–Borel compactness, sequential compactness, host suprema, countable choice, selected subsequences, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedBoundedRealSequenceUp.TasteGate.LocatedBoundedRealSequenceTasteGate single carrier alignment`

Theorem 2183.2 (Located bounded real sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ *carrier*

$$S = (I, L, U, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected-index row, located-real row, boundedness row, dyadic comparison ledger, stream observation row, regular rational readback row, real seal, transport, replay, provenance, and local naming row;*

2. *located-entry exposure, requiring every inspected sequence entry to pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *windows in L and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W;*

3. *boundedness exposure, requiring every lower-bound or upper-bound read to be displayed in U and compared through the dyadic ledger D;*

4. *dyadic exactness, requiring every tolerance, shrink, and comparison step used by an inspected entry to be recorded in D;*

5. *regular readback discipline, requiring R to be the only route from the displayed stream and dyadic rows into regular rational observations;*

6. *real-seal nonescape, requiring E to remain downstream of I, L, U, D, W, R and to export no host supremum, compactness theorem, selected subsequence, or ambient completeness witness;*

7. *transport, continuation, provenance, and naming stability, requiring H, C, P, N to preserve and source exactly the displayed rows componentwise.*

No located-bounded-real-sequence obligation is stored outside these rows.

Proof. Project the accepted packet $S = (I, L, U, D, W, R, E, H, C, P, N)$. The carrier exposes the inspected indices, located-real entry windows, boundedness windows, dyadic comparison ledger, stream observations, regular readbacks, and terminal real seal as the whole analytic surface.

An entry read starts with I, L, W. The row I fixes the finite positions under inspection, L supplies the located-real windows for those positions, and W names the stream windows through which the observations are read.

The boundedness read then passes through U and D. The row U supplies the displayed lower and upper real-boundary windows, while D supplies the dyadic radii, shrink steps, and comparison cells used to relate the inspected entries to those bounds.

The readback and seal rows are downstream. The row R is the only regular rational handoff from the displayed stream and dyadic data, and the row E is exposed only after the index, located-entry, boundedness, dyadic, stream, and readback rows have been retained.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since compactness, selected subsequences, host suprema, countable choice, and $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate or replay edge, the listed rows exhaust the local NameCert surface. □

Theorem 2183.3 (Located bounded real sequence Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read of an accepted* $\mathsf{LocatedBoundedRealSequence}^{\uparrow}$ *carrier factors through the finite route*

$$I, L, U \longmapsto D, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives inspected sequence indices, located real entry windows, displayed lower and upper real-boundary windows, $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison rows, finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. It receives no Heine–Borel compactness, sequential compactness, host supremum, countable-choice schedule, selected subsequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a real-facing or regular-rational consumer read of the accepted packet. The carrier contains no global sequence object outside the displayed finite rows, so the read first enters through the inspected-index row I and the located-entry row L.

The bounded sequence boundary is then finite. The row U records the displayed lower and upper real-boundary windows, and D records the dyadic comparisons that bind the inspected entries to those windows. The row W names the exact stream observations used by both the entry reads and the bound reads.

The public handoff is R followed by E. The $\mathsf{RegSeqRat}^{\uparrow}$ row translates the displayed stream and dyadic rows into regular rational observations, and the $\mathsf{Real}^{\uparrow}$ seal exposes only the real-name boundary reached through that readback.

The structural rows preserve the same dependency order. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row can preserve or source the route, but they cannot introduce compactness, selected subsequences, host suprema, countable choice, or completeness data absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2183.1, Definition 2183.2, and Definition 2183.3 seed the finite located bounded real sequence packet over $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedBoundedRealSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, sequential compactness, host suprema, countable choice, selected subsequences, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked located-entry exposure, boundedness exposure, dyadic exactness, regular readback discipline, real-seal nonescape, componentwise transport, continuation replay, provenance, and local naming for BEDC.Derived.LocatedBoundedRealSequenceUp.

Constructive story: A LocatedBoundedRealSequence$^{\uparrow}$ packet is generated from inspected sequence indices, LocatedReal$^{\uparrow}$ entry windows, displayed lower and upper Real$^{\uparrow}$ bound windows, DyadicRatCore$^{\uparrow}$ comparison rows, finite StreamName$^{\uparrow}$ observation windows, RegSeqRat$^{\uparrow}$ readbacks, a terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2184

A Concrete Naming Certificate for $\mathsf{VitaliCover}^{\uparrow}$

Following Vitali and Bishop–Bridges constructive analysis, the $\mathsf{VitaliCover}^{\uparrow}$ packet records the finite ledger by which a displayed interval family offers disjoint candidates for later measure-facing arguments. It sits after the dyadic interval surface of chapter 306, the finite epsilon-net route of chapter 1767, the measure interface of chapter 92, and the real-name seal of chapter 38. The packet names finite interval selection data only; it does not assert the full Vitali covering theorem, measure completion, countable selection, open-cover compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2184.1 (VitaliCover carrier)**.** *A* $\mathsf{VitaliCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, D, E, F, M, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval-family row. Each listed candidate is read through the* $\mathsf{DyadicInterval}^{\uparrow}$ *interface and carries its displayed endpoints and radius ledger.*
2. *D is the disjointness ledger. It records the finite pairwise separation checks used to certify that selected intervals do not overlap.*
3. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row. It records the finite demand points or cells whose local cover requests are visible to the packet.*
4. *F is the finite selection row. It records the chosen subfamily, its ordering, and the dependency of each chosen interval on a displayed request in E.*
5. *M is the* $\mathsf{Measure}^{\uparrow}$ *handoff row. It records only the finite additive ledger exposed to a measure-facing consumer after I, D, E, F have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary row. It records endpoint and radius reads through real-name seals rather than host real values.*
7. *H records componentwise* hsame *transport for the interval family, disjointness ledger, epsilon-net row, selection row, measure handoff, real boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$I, D, E, F \longmapsto M, R \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{VitaliCover}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{VitaliCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, E, F, M, R, H, C, P, N$. It has no coordinate for an infinite subcover, a countable selector, completed measure, host open-cover compactness, or quotient equality of real endpoints.

Lean4 ✓ `BEDC.Derived.VitaliCoverUp.VitaliCoverTasteGate single carrier alignment`

Theorem 2184.2 (VitaliCover NameCert obligations)**.** *For every accepted* $\mathsf{VitaliCover}^{\uparrow}$ *carrier*

$$V = (I, D, E, F, M, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VitaliCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval family, disjointness ledger, epsilon-net witnesses, finite selection row, measure handoff, real boundary, transport, replay, provenance, and local naming row;*
2. *interval locality, requiring every candidate interval to be read through the* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint and radius data in I;*
3. *finite request coverage, requiring each selected interval in F to answer a displayed request in the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row E;*
4. *disjointness exactness, requiring the pairwise ledger D to certify all overlaps refused by the selected finite subfamily;*
5. *measure handoff, requiring the* $\mathsf{Measure}^{\uparrow}$ *row M to consume only the selected disjoint finite family and its displayed dyadic endpoint data;*
6. *real-boundary nonescape, requiring the* $\mathsf{Real}^{\uparrow}$ *row R to expose only endpoint and radius seals already routed through I, D, E, F;*
7. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the rows above.*

No obligation row supplies the full Vitali covering theorem, measure completion, countable selection, open-cover compactness, quotient real equality, or ambient real completeness.

Proof. Project the carrier in Definition 2184.1. The packet exposes only the interval family I, disjointness ledger D, epsilon-net requests E, finite selection row F, measure handoff M, real boundary R, and the structural rows H, C, P, N.

The locality and request-coverage obligations are read before any measure-facing consumer is reached. Candidate intervals enter through $\mathsf{DyadicInterval}^{\uparrow}$, and every selected interval is tied to a displayed finite request in $\mathsf{FiniteEpsilonNet}^{\uparrow}$.

The disjointness ledger is then the only selected-subfamily compatibility row. It records pairwise finite separation checks, so a later consumer may not replace it by a host almost-disjoint family or an infinite extraction principle.

Finally M and R consume the same finite route, while H, C, P, N transport, replay, source, and name it. Since no other coordinate is available, the listed rows exhaust the NameCert surface. □

Theorem 2184.3 (VitaliCover finite selection handoff)**.** *Let*

$$V = (I, D, E, F, M, R, H, C, P, N)$$

be an accepted $\mathsf{VitaliCover}^{\uparrow}$ *carrier. Every measure-facing read of the finite disjoint selection factors through the route*

$$I \longmapsto D \longmapsto E \longmapsto F \longmapsto M, R \longmapsto H, C, P, N.$$

The handoff supplies $\mathsf{Measure}^{\uparrow}$ *with a selected finite disjoint interval ledger and supplies* $\mathsf{Real}^{\uparrow}$ *with the endpoint and radius seals used by that ledger. It does not export the full Vitali covering theorem, measure completion, countable selection, open-cover compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a measure-facing read of V. The candidate intervals are first read through I, so their endpoint and radius information remains inside the dyadic interval surface.

The pairwise compatibility check is then D, and the finite request source is E. The selected row F can therefore name only intervals whose requests and disjointness witnesses have already been displayed.

The consumer handoff is downstream of those rows. The $\mathsf{Measure}^{\uparrow}$ row M reads the finite additive ledger for the selected disjoint family, and the $\mathsf{Real}^{\uparrow}$ row R supplies the endpoint and radius seals used by that ledger.

Transport by H, replay by C, provenance by P, and naming by N preserve the same finite route. No excluded infinite or completed covering principle has a carrier coordinate from which it could be exported. □

Theorem 2184.4 (VitaliCover Besicovitch finite-overlap boundary)**.** *Let*

$$V = (I, D, E, F, M, R, H, C, P, N)$$

be an accepted $\mathsf{VitaliCover}^{\uparrow}$ *carrier. Every measure-facing read of its accepted finite selection may be routed through a bounded finite-overlap shadow ledger when the selected intervals are compared against the finite Vitali request rows and the finite epsilon-net request rows of chapter 2166 and chapter 1767.*

The shadow ledger is bounded entirely by the displayed rows $I, D, E, F, M, R, H, C, P, N$*:* I *supplies the finite interval candidates,* D *supplies pairwise disjointness for selected rows,* E *and* F *supply finite request coverage and selected-subfamily membership, and* M *is the only* $\mathsf{Measure}^{\uparrow}$*-facing additive read. The boundary exports no measure completion, countable selection, open-cover compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier along Definition 2184.1. The finite interval rows I are the only source of candidate intervals, and the selected rows of F can be read only after their finite request witnesses in E have been displayed.

The bounded-overlap shadow is therefore not an additional covering theorem. It is the finite ledger obtained by comparing the selected rows in F against the finite Vitali and epsilon-net request surfaces named by chapter 2166 and chapter 1767.

Use D as the pairwise disjointness boundary for the selected subfamily. Since every selected interval is checked against that ledger before the measure handoff, a measure-facing consumer can count overlaps only through the finite shadow already exposed by I, D, E, F.

Finally route the additive read through M and the endpoint seals through R, with H, C, P, N transporting, replaying, sourcing, and naming the same rows. By chapter 92, this is a finite measure-facing ledger, not a completion, countable-selection, compactness, quotient-real, or ambient-completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{VitaliCover}^{\uparrow}$ measure-facing read factors through a finite DyadicInterval family, a pairwise disjointness ledger, a FiniteEpsilonNet request row, a finite selection row, a Measure handoff, a Real endpoint boundary, transport, continuation, provenance, and local naming.

Independence witness: dyadicinterval, finite-epsilon-net, measure, real: $\mathsf{VitaliCover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval candidates, finite request coverage, pairwise disjointness, selected-subfamily ordering, measure handoff, real endpoint seals, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{VitaliCover}^{\uparrow}}$.

Constructive story: A $\mathsf{VitaliCover}^{\uparrow}$ packet is generated from a finite DyadicInterval family, pairwise disjointness ledger, FiniteEpsilonNet request row, finite selection row, Measure handoff, Real endpoint boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VitaliCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2184.1, Definition 2184.2, and Definition 2184.3 seed the finite disjoint interval selection ledger for measure-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VitaliCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full Vitali covering theorem, measure completion, countable selection, open-cover compactness, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite request coverage, disjointness exactness, measure handoff, real-boundary nonescape, and provenance exhaustion for BEDC.Derived.VitaliCoverUp.

Chapter 2185

A Concrete Naming Certificate for $\mathsf{FastCauchySequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{FastCauchySequence}^{\uparrow}$ packet records a standard constructive source object for the L10 Real and Completion route. It sits after the dyadic arithmetic ledger of chapter 291, the finite window surface of chapter 126, and the regular rational readback surface of chapter 301. The packet names a fast Cauchy sequence as finite BEDC carrier data; it is not a quotient stream, selected limit, trichotomy principle, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2185.1 (Fast Cauchy sequence carrier)**.** *A* $\mathsf{FastCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, M, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger. It stores the dyadic terms visible at each requested index, the positive dyadic tolerance rows, and the comparison windows used by the fast Cauchy test.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It specifies which term observations and tolerance requests are visible to downstream consumers.*
3. *M is the explicit fast Cauchy modulus ledger. For every displayed request level k, it records a finite budget row forcing all scheduled later windows to be within the dyadic tolerance named by D at level k.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the same dyadic observations, stream windows, and modulus ledger.*
5. *H stores componentwise* hsame *transport for D, S, M, R, C, P, N.*
6. *C stores displayed* Cont *replay from dyadic observations through stream windows and modulus checks to the regular rational handoff row.*
7. *P is the* Pkg *provenance over* $\mathsf{FastCauchySequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted fast Cauchy sequence packets only through D, S, M, R, H, C, P, N. It has no coordinate for quotient streams, selected limits, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchySequenceUp.FastCauchySequenceTasteGate single carrier alignment`

Theorem 2185.2 (Fast Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchySequence}^{\uparrow}$ *carrier $F = (D, S, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FastCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic observation exposure, finite window exposure, explicit fast Cauchy modulus ledger exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside D, S, M, R, H, C, P, N.*

Lean4 ✓ `BEDC.Derived.FastCauchySequenceUp.FastCauchySequenceCarrier namecert obligations`

Proof. Project the carrier of Definition 2185.1. The analytic rows are exactly the dyadic observation ledger D, finite stream windows S, explicit modulus ledger M, and regular rational handoff R.

The fast Cauchy obligation is the defining use of M. Each displayed request level reads its dyadic tolerance from D, its visible windows from S, and its later-window comparison budget from M.

The handoff obligation is then the projection to R. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the displayed list exhausts the local NameCert surface. □

Theorem 2185.3 (Fast Cauchy sequence RegSeqRat handoff)**.** *For every accepted* $\mathsf{FastCauchySequence}^{\uparrow}$ *carrier $F = (D, S, M, R, H, C, P, N)$, any* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer reaches the readback row R only through*

$$D \longmapsto S \longmapsto M \longmapsto R.$$

Thus the route from fast Cauchy sequence data to regular rational data must display the $\mathsf{DyadicRatCore}^{\uparrow}$ *observations,* $\mathsf{StreamName}^{\uparrow}$ *windows, and explicit fast Cauchy modulus ledger before the handoff is consumed.*

Proof. Start with a consumer asking for regular rational readback from the packet. The only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate is R, and the carrier places it downstream of D, S, M.

The dyadic row D supplies the observed terms and tolerances, while S selects the finite windows that may be inspected. The modulus ledger M binds those windows to the fast Cauchy comparison budget.

Transport, replay, provenance, and local naming preserve that route without adding an alternate source for readback data. Hence the consumer reaches R only after the displayed dyadic, stream, and modulus rows are present. □

Theorem 2185.4 (Fast Cauchy sequence prefix-window tightening)**.** *For every accepted* $\mathsf{FastCauchySequence}^{\uparrow}$ *carrier $F = (D, S, M, R, H, C, P, N)$ and every displayed request level k, any finite* $\mathsf{StreamName}^{\uparrow}$ *window retained by S for the k-request reads its later-window comparison budget from the same modulus row M and dyadic tolerance row D before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R is available. Consequently the handoff exposes no selected limit, quotient-stream equality, trichotomy, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier rows from Definition 2185.1. The retained finite window is a row of S, while the tolerance visible at request level k is stored in the dyadic ledger D.

The row M is the displayed fast Cauchy modulus ledger. It is the only carrier coordinate that binds later-window comparisons to the tolerance selected at the same request level, so the k-window cannot draw a separate budget from R or from an external completion surface.

Now compose this projection with the route of Definition 2185.3. Since every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches R only after $D \longmapsto S \longmapsto M$ is present, the finite prefix-window tightening is consumed before the handoff and carries no additional limit, quotient, trichotomy, choice, or $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 2185.5 (Fast Cauchy sequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FastCauchySequence}^{\uparrow}$ *packet reaches the terminal Real seal only through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row D, the* $\mathsf{StreamName}^{\uparrow}$ *window row S, the explicit modulus ledger M, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R. The handoff exports no selected limit, quotient stream equality, countable-choice selector, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier in Definition 2185.1. The only Real-facing route available to the packet begins with D, S, M and then reaches the regular readback row R.

Apply the RegSeqRat handoff theorem Definition 2185.3. It places every read of R downstream of the dyadic observations, stream windows, and explicit modulus ledger.

Use the prefix-window tightening theorem Definition 2185.4. It rules out any separate selected limit, quotient stream equality, countable-choice selector, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness row before the handoff is consumed. The terminal seal therefore carries only the displayed finite route through D, S, M, R. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchySequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FastCauchySequence}^{\uparrow}$ packet is generated from DyadicRatCore observations, StreamName windows, an explicit fast Cauchy modulus ledger, RegSeqRat handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2185.1, Definition 2185.2, Definition 2185.3, Definition 2185.4, and Definition 2185.5 seed the finite fast Cauchy sequence source object for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic observation exposure, finite window exposure, explicit modulus-ledger exposure, RegSeqRat handoff, transport, replay, provenance, and local naming for BEDC.Derived.FastCauchySequenceUp.

Chapter 2186

A Concrete Naming Certificate for NormalFamily$^{\uparrow}$

Following Montel and Arzela–Ascoli, the NormalFamily$^{\uparrow}$ packet names the finite compact-continuity surface used to read a normal family of maps from a compact metric source into a complete metric target. It consumes the ArzelaAscoli$^{\uparrow}$ handoff of chapter 412, the compact source rows of chapter 129, the complete target rows of chapter 128, the continuous-map family rows of chapter 130, and the pointwise image rows of TotallyBounded$^{\uparrow}$ as finite carrier data.

The chapter records the normal-family object as a naming certificate surface, not as an ambient compact-open topology, a selected convergent subsequence, or a choice principle. Every consumer read must pass through the displayed family ledger, compact source net, complete target ledger, equicontinuity rows, pointwise total-bounded image rows, finite ProbeBundle net data, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Definition 2186.1 (Normal-family carrier)**.** *A* NormalFamily$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (F, X, Y, M, E, I, Q, H, C, P, L)$$

with the following visible coordinates:

1. *F is the finite family ledger whose entries are* ContinuousMap$^{\uparrow}$ *rows from the source row X to the target row Y;*
2. *X is the compact source row, read through* CompactMetric$^{\uparrow}$ *and its finite* ProbeBundle *net surface;*
3. *Y is the complete target row, read through* CompleteMetric$^{\uparrow}$ *and its displayed limit-witness ledgers;*
4. *M is the compact-continuity handoff row supplied by* ArzelaAscoli$^{\uparrow}$ *for the same source, target, and family ledger;*
5. *E records equicontinuity rows binding target tolerances to finite source-radius rows uniformly over F;*
6. *I records pointwise* TotallyBounded$^{\uparrow}$ *image rows for the displayed target values over the bundled source probes;*
7. *Q records the finite* ProbeBundle *net data used by the compact source and by the Arzela–Ascoli handoff;*

8. *H records componentwise* hsame *transport for the family ledger, compact source, complete target, equicontinuity rows, pointwise image rows, probe rows, and handoff row;*
9. *C records the displayed* Cont *replay routes for compact-metric, complete-metric, continuous-map, total-bounded, and Arzela–Ascoli consumers;*
10. *P is the* Pkg *provenance over* $\mathsf{NormalFamily}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, ProbeBundle, hsame, *and* Cont *rows;*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{NormalFamily}^{\uparrow}}$ *row.*

The packet contains no host function-space carrier, open-cover compactness row, selected subsequence, ambient holomorphic-function principle, or choice-selected cluster point.

Lean4 (definition) `BEDC.Derived.NormalFamilyUp.TasteGate.NormalFamilyUp`
Lean4 ✓ `BEDC.Derived.NormalFamilyUp.TasteGate.NormalFamilyTasteGate single carrier alignment`

Theorem 2186.2 (Normal-family NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{NormalFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for the packet of Definition 2186.1; classifier transport of the family ledger, compact source, complete target, Arzela–Ascoli handoff, equicontinuity rows, pointwise total-bounded image rows, finite probe rows, and displayed provenance; and non-escape through the recorded* hsame, Cont, Pkg, *and local* NameCert *rows.*

Proof. Carrier admission is the displayed tuple $N = (F, X, Y, M, E, I, Q, H, C, P, L)$. Each analytic coordinate is visible before the local certificate may be read: the family ledger F, compact source X, complete target Y, Arzela–Ascoli handoff M, equicontinuity rows E, pointwise image rows I, and probe rows Q.

Classifier transport is componentwise. The $\mathsf{ContinuousMap}^{\uparrow}$ rows transport the family graph data, the $\mathsf{CompactMetric}^{\uparrow}$ row transports source finite-net data, the $\mathsf{CompleteMetric}^{\uparrow}$ row transports target limit ledgers, and the $\mathsf{TotallyBounded}^{\uparrow}$ rows transport pointwise image bundles.

The remaining rows H, C, P, L preserve the same coordinates under hsame transport, replay them through displayed Cont routes, record their Pkg provenance, and name the packet locally. Since no omitted coordinate names a host function space, open-cover compactness, selected subsequence, or choice-selected cluster point, the certificate exports only the displayed normal-family surface. □

Theorem 2186.3 (Normal-family Arzela-Ascoli handoff)**.** *Every compact-continuity consumer reading an accepted* $\mathsf{NormalFamily}^{\uparrow}$ *packet factors through the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *row M, the compact source row X, the complete target row Y, the continuous-map family ledger F, the equicontinuity rows E, the pointwise* $\mathsf{TotallyBounded}^{\uparrow}$ *image rows I, the finite* ProbeBundle *data Q, and the displayed* hsame, Cont, Pkg, *and* NameCert *provenance.*

Consequently the handoff supplies the finite data needed for the constructive normal-family reading and supplies no ambient compact-open topology, open-cover compactness principle, selected subsequence, or host choice principle.

Proof. Project the Arzela–Ascoli coordinate M. Its role is only to consume the same compact source, complete target, family ledger, equicontinuity rows, pointwise image rows, and finite probe rows already displayed in the $\mathsf{NormalFamily}^{\uparrow}$ carrier.

Next read X, Y, F, E, I, Q. These rows provide the compact source net, the complete target ledger, the continuous-map family entries, the uniform equicontinuity witnesses, the pointwise total-bounded image bundles, and the probe data that the handoff needs.

Finally attach H, C, P, L. They ensure that every downstream read uses the transported and replayed carrier rows rather than an ambient function-space object. The absence of any further coordinate rules out open-cover compactness, selected subsequences, and host choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalFamily}^{\uparrow}$ packet is generated from a finite family ledger, compact source row, complete target row, Arzela-Ascoli handoff row, equicontinuity rows, pointwise total-bounded image rows, finite ProbeBundle net data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2186.1, Definition 2186.2, and Definition 2186.3 seed the finite normal-family certificate over ArzelaAscoliUp, CompactMetricUp, CompleteMetricUp, ContinuousMapUp, TotallyBoundedUp, ProbeBundle, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient compact-open topology, open-cover compactness, selected subsequences, holomorphic compactness, or host choice.

Upgrade path: Obligation closure requires checked carrier projection, classifier transport, Arzela-Ascoli handoff confinement, provenance replay, and local naming for the $\mathsf{NormalFamily}^{\uparrow}$ packet.

Chapter 2187

A Concrete Naming Certificate for $\mathsf{MetricCoveringNumber}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MetricCoveringNumber}^{\uparrow}$ packet records the quantitative finite-net count attached to a displayed metric tolerance. It sits between the metric source of chapter 57, the finite epsilon-net route of chapter 1767, the total-bounded surface of chapter 127, the compact-metric consumer of chapter 129, and the dyadic tolerance core of chapter 291. The packet names the covering-number boundary as finite BEDC data, not as a host finite set, a minimal-cardinality proof, or an open-cover compactness argument.

Definition 2187.1 (Metric covering-number carrier)**.** *A* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, \varepsilon, P, C, D, B, H, R, S, N)$$

with the following visible rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier, distance classifier, and metric-law ledger read by the packet.*
2. *ε is the positive tolerance row, read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with the displayed positivity evidence used by coverage consumers.*
3. *P is the closed* $\mathsf{ProbeBundle}$ *of centers admitted for the selected tolerance.*
4. *C is the center-carrier ledger, recording for every accepted center of P the corresponding point-classifier admission in X.*
5. *D is the distance and coverage ledger pairing each inspected source-point history with an admitted center and a metric-distance row bounded by ε.*
6. *B is the unary budget row naming the number of center entries exposed by P and replayed by the coverage ledger.*
7. *H records componentwise* hsame *transport for the metric source, tolerance row, center bundle, center-carrier ledger, coverage ledger, budget row, replay row, provenance row, and local naming row.*
8. *R records displayed* Cont *replay from a covering-number request through $X, \varepsilon, P, C, D, B, H, S, N$.*
9. *S is the* Pkg *provenance over* $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCoveringNumber}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, \varepsilon, P, C, D, B, H, R, S, N$. It has no coordinate for an ambient finite subset, host enumeration, least-cardinality selector, open-cover refinement, sequential compactness proof, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCoveringNumberUp.MetricCoveringNumberTasteGate single carrier alignment`

Theorem 2187.2 (Metric covering-number NameCert obligations)**.** *For every accepted* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *carrier $K = (X, \varepsilon, P, C, D, B, H, R, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCoveringNumber}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, positive tolerance, center bundle, center-carrier ledger, coverage-distance ledger, unary budget, transport, replay, provenance, and local naming rows;*
2. *center visibility, requiring each center counted by B and used by D to enter through the same displayed* $\mathsf{ProbeBundle}$ *row P and center-carrier ledger C;*
3. *coverage exactness, requiring every inspected point of X to be paired by D with a displayed distance row bounded by ε;*
4. *budget exactness, requiring the exported covering number to count exactly the visible center entries of P used by the coverage ledger;*
5. *tolerance transport, requiring the* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row to remain fixed under H before any consumer reads the count;*
6. *ledger nonescape, excluding ambient finite subsets, host enumerations, least-cardinality selectors, open-cover compactness, sequential compactness, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No metric covering-number obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 2187.1. The analytic rows are exactly the metric source X, positive tolerance ε, center bundle P, center-carrier ledger C, coverage-distance ledger D, and unary budget row B.

Center visibility and coverage exactness are rowwise. A center can be counted only after it is visible in P and admitted by C as a point of X, and a source point is covered only through one of the distance rows displayed by D.

The budget row then names the quantitative boundary. Because B counts the same visible entries that P, C, D use, a consumer cannot replace the packet by a smaller host set, an external enumeration, or a minimal-cardinality argument.

The transport and replay rows preserve the same route. The row H moves no coordinate beyond $X, \varepsilon, P, C, D, B, R, S, N$, and R replays covering-number requests through the same ordered data. The refusal clauses follow from absence of coordinates for open covers, selected compactness proofs, countable choice, sequential compactness, and ambient completeness. □

Theorem 2187.3 (Metric covering-number finite-net handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer reading an accepted* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *carrier factors through the finite route*

$$X, \varepsilon \longmapsto P, C, D, B \longmapsto H, R, S, N.$$

Consequently the consumer receives the finite center bundle, positive tolerance row, coverage witnesses, and unary covering budget before any total-bounded or compact-metric conclusion is available. The handoff exports no ambient finite subset, host enumeration, least-cardinality selector,

open-cover compactness proof, sequential compactness proof, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a total-bounded or compact-metric consumer of K. By Definition 2187.1, the consumer can enter the packet only through the displayed metric source X and the displayed tolerance row ε.

The finite-net data are opened next. The closed bundle P names the centers, C admits those centers as points of X, and D supplies the coverage-distance rows. The budget row B then records the quantitative count of exactly those center entries.

Finally H, R, S, N package the same route under classifier transport, continuation replay, provenance, and local naming. Applying Definition 2187.2, the consumer cannot bypass the finite-net route through a host finite subset, an open-cover compactness proof, or a choice-selected enumeration. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCoveringNumber}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCoveringNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2187.1, Definition 2187.2, and Definition 2187.3 seed the quantitative finite-net packet over displayed metric sources, positive rational or dyadic tolerances, center bundles, coverage ledgers, unary budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCoveringNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimal covering numbers, total boundedness, compact metric completeness, open-cover compactness, sequential compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center visibility, coverage exactness, budget exactness, tolerance transport, ledger nonescape, and finite-net consumer handoff for BEDC.Derived.MetricCoveringNumberUp.

Constructive story: A $\mathsf{MetricCoveringNumber}^{\uparrow}$ packet is generated from a MetricUp source row, a positive RatUp or DyadicRatCoreUp tolerance, a closed ProbeBundle of centers, a center-carrier ledger, distance and coverage rows, a unary center budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2188

A Concrete Naming Certificate for $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$ packet names the complete finite-net route by which compact metric data and continuous-map pointwise moduli yield a uniform-modulus ledger. It consumes the compact metric source of chapter 129, the continuous-map rows of chapter 130, the metric triangle surface of chapter 57, the finite rational lower-bound fold of chapter 619, the uniform-modulus consumer row of chapter 369, and the rational tolerance source of chapter 37. The scheme records the Heine–Cantor proof as one BEDC packet, with no appeal to open-cover compactness, a host modulus selector, or choice.

Definition 2188.1 (Compact uniform-continuity scheme carrier)**.** *A* $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, F, E, P, M, Q, T, G, H, R, S, N)$$

with the following visible rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite* $\mathsf{ProbeBundle}$ *net surface used at the requested tolerance.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose triangle ledger reads the final estimate.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row from X to Y.*
4. *E is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row, including the rational subdivision data used for the source and target triangle estimates.*
5. *P is the finite center bundle supplied by the compact metric row.*
6. *M is the pointwise-modulus ledger at the centers of P.*
7. *Q is the subordinate-radius ledger binding each center radius to the pointwise-modulus precision at that center.*
8. *T is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound fold over the finite radius ledger.*
9. *G is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row produced from E, T, M, Q and the metric triangle ledgers of X and Y.*
10. *H records componentwise* hsame *transport for the compact source, target metric, continuous map, tolerance row, center bundle, pointwise rows, subordinate radii, finite fold, uniform-modulus handoff, replay, provenance, and local naming.*

11. *R records displayed* Cont *replay from a uniform-modulus request through* $X, Y, F, E, P, M, Q, T, G, H, S, N$.
12. *S is the* Pkg *provenance over* $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityScheme}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, Y, F, E, P, M, Q, T, G, H, R, S, N$. It contains no open-cover compactness coordinate, selected Lebesgue-number proof, host modulus function, countable-choice schedule, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuitySchemeUp.CompactUniformContinuitySchemeTasteGate`
`single carrier alignment`

Theorem 2188.2 (Compact uniform-continuity scheme NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$ *carrier* $U = (X, Y, F, E, P, M, Q, T, G, H, R, S, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityScheme}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed compact source, target metric, continuous-map ledger, positive rational tolerance, finite center bundle, pointwise-modulus rows, subordinate-radius ledger, finite lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local naming rows;*
2. *source lock, requiring the compact metric row X and continuous-map source row in F to expose the same metric carrier before any center or modulus row is read;*
3. *finite-net extraction, requiring the centers in P and coverage rows in X to be the only compactness data consumed by the scheme;*
4. *pointwise-modulus binding, requiring each row of M and Q to attach to a displayed center of P and to the same rational tolerance row E;*
5. *rational-fold exactness, requiring T to be the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the displayed subordinate radii rather than an external minimum selector;*
6. *uniform-modulus handoff, requiring G to be obtained from the finite fold, pointwise rows, and source and target metric triangle ledgers;*
7. *ledger nonescape, excluding open-cover compactness, selected Lebesgue numbers, host modulus functions, choice, sequential compactness, and ambient completeness as local certificate data.*

No compact uniform-continuity scheme obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 2188.1. The analytic rows are the compact source X, target metric Y, continuous-map ledger F, rational tolerance E, center bundle P, pointwise rows M, subordinate radii Q, rational fold T, and uniform-modulus handoff G.

The source lock is forced before any compact-continuity read. The compact metric row and the continuous-map source row are the same displayed metric carrier, so the center rows and pointwise-modulus rows cannot be read over different sources.

The finite-net and pointwise-modulus obligations are rowwise. The bundle P supplies the centers, M supplies the local modulus rows at those centers, and Q attaches each subordinate radius to the corresponding local precision.

The lower-bound fold T is the quantitative step. Since it is carried by $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ over the displayed $\mathsf{Rat}^{\uparrow}$ radii, the scheme exports a classifier-stable precision history rather than a host-selected minimum.

Finally G reads the source and target triangle ledgers through X and Y and packages the result as a $\mathsf{UniformModulus}^{\uparrow}$ row. The structural rows H, R, S, N replay only the displayed route, so open-cover compactness, selected Lebesgue numbers, host modulus functions, choice, and ambient completeness have no coordinate from which to enter the local certificate. □

Theorem 2188.3 (Compact uniform-continuity scheme handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer reading an accepted* $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$ *carrier factors through the finite route*

$$X, F, E \longmapsto P, M, Q, T \longmapsto G, H, R, S, N.$$

Thus the consumer receives compact finite-net data, continuous-map pointwise-modulus rows, rational subordinate-radius rows, the finite lower-bound fold, and the metric-triangle estimate before the uniform-modulus row is public. The handoff exports no open-cover compactness, selected Lebesgue number, host modulus selector, countable-choice schedule, sequential compactness argument, or ambient completeness theorem.

Proof. Begin with a uniform-modulus consumer of U. By Definition 2188.1, the consumer first reads the compact source X, the continuous-map row F, and the rational tolerance E.

The finite compactness and continuity data are opened next. The center bundle P comes from the compact metric row, the pointwise rows M come from the continuous-map ledger at those centers, and the subordinate-radius ledger Q binds the two surfaces to the same tolerance request.

The lower-bound fold T then computes the tolerance-only precision history over the displayed finite rational radius ledger. The handoff row G applies the metric triangle rows in the source and target metrics and records the result as a $\mathsf{UniformModulus}^{\uparrow}$ row.

The transport, replay, provenance, and local naming rows expose exactly this finite route. Applying Definition 2188.2, the consumer cannot replace it by an open-cover proof, a selected Lebesgue number, an external modulus function, countable choice, sequential compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityScheme}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityScheme}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2188.1, Definition 2188.2, and Definition 2188.3 seed the compact-uniform-continuity proof scheme over compact metric finite nets, continuous-map pointwise moduli, rational subordinate radii, finite lower-bound folds, metric triangle ledgers, uniform-modulus handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityScheme}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, host modulus functions, sequential compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source lock, finite-net extraction, pointwise-modulus binding, rational-fold exactness, uniform-modulus handoff, and ledger nonescape for BEDC.Derived.CompactUniformContinuitySchem

Constructive story: A $\mathsf{CompactUniformContinuityScheme}^{\uparrow}$ packet is generated from CompactMetricUp source finite-net rows, a target MetricUp row, ContinuousMapUp pointwise-modulus rows, a positive RatUp tolerance, subordinate center radii, a FiniteNetMinimumFoldUp lower-bound fold, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2189

A Concrete Naming Certificate for $\mathsf{BanachAlgebra}^{\uparrow}$

Following Bourbaki and Gelfand–Naimark, the $\mathsf{BanachAlgebra}^{\uparrow}$ packet records the classical complete normed algebra interface as finite BEDC NameCert data. It sits between the Banach carrier of chapter 90, the multiplication surface of chapter 43, the norm rows of chapter 89, the completion witnesses of chapter 128 and chapter 95, and the C*-algebra consumer of chapter 211. The packet names a complete normed algebra row before involution, spectrum, representation, or C^*-identity data are available.

Definition 2189.1 (Banach algebra carrier)**.** *A* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, N, B, Q, M, H, C, P, L)$$

with the following displayed rows:

1. *R is the* $\mathsf{Ring}^{\uparrow}$ *multiplication row, including product, additive carrier compatibility, distributivity, associativity, and unit data when the source algebra displays a unit.*
2. *N is the* $\mathsf{Norm}^{\uparrow}$ *norm row over the same carried additive carrier, including non-negativity, zero exactness, homogeneity, triangle, and classifier transport for norm values.*
3. *B is the* $\mathsf{Banach}^{\uparrow}$ *complete normed carrier row. It supplies the vector-additive carrier, scalar action, norm-induced metric, and regular Cauchy limit surface used by the algebra packet.*
4. *Q is the product-norm control row. For carried endpoints x, y, it records the finite ledger comparing the norm of the displayed product with the product of the displayed norms:*

$$\nu(x \cdot y) \leq \nu(x)\nu(y).$$

5. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion seal. It states that completion-facing reads of the norm-induced metric land back in the same algebra carrier and may inspect multiplication only through the displayed product row R and product-control row Q.*
6. *H records componentwise* hsame *transport for the RingUp multiplication row, NormUp norm row, BanachUp complete carrier row, completion seal, product-control ledger, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$\mathsf{Ring}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Banach}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{BanachAlgebra}^{\uparrow}.$$

8. *P is the* Pkg *provenance over* $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier compares accepted packets only through $R, N, B, Q, M, H, C, P, L$ and has no coordinate for involution, spectrum, Gelfand transform, weak-operator closure, Hilbert-space representation, host algebra equality, quotient completion, selected limits, or countable choice.

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraTasteGate single carrier alignment`

Theorem 2189.2 (Banach algebra complete product carrier)**.** *For an accepted* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier $A = (R, N, B, Q, M, H, C, P, L)$, every algebra product consumer first reads the displayed* BHist *rows for* $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, *and multiplication before it may inspect a Cauchy product. The complete product carrier is therefore the* Pkg*-named surface transported by* hsame, *replayed by* Cont, *and locally sealed by* $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$*; a* $\mathsf{CStarAlg}^{\uparrow}$ *consumer sees only that complete normed product boundary before adding involution data.*

Proof. Project the accepted packet onto $R, N, B, Q, M, H, C, P, L$. The multiplication row is R, the norm row is N, the complete Banach carrier row is B, and the complete-metric seal is M.

A Cauchy product read must pass through these rows in this order. The product is not available from $\mathsf{CompleteMetric}^{\uparrow}$ alone, and the Cauchy limit service is not available from $\mathsf{Ring}^{\uparrow}$ alone; the Banach algebra carrier exposes their joint surface.

The rows H, C, P, L then transport, replay, source, and name the same surface. Thus the $\mathsf{CStarAlg}^{\uparrow}$ endpoint receives a complete normed product carrier, not an involutive or spectral object supplied by the root packet. □

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraCompleteProductCarrier`

Theorem 2189.3 (Banach algebra norm submultiplicativity ledger)**.** *For an accepted* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier $A = (R, N, B, Q, M, H, C, P, L)$, submultiplicative norm control is the displayed* BHist *ledger row Q over the* $\mathsf{Ring}^{\uparrow}$ *multiplication route and the* $\mathsf{Norm}^{\uparrow}$ *route. It is not a host inequality oracle: any read of $\nu(x \cdot y) \leq \nu(x)\nu(y)$ must cite the product row R, the norm row N, the ledger row Q, the* hsame *transport row H, the* Cont *replay row C, the* Pkg *provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$ *row L.*

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraNormSubmultiplicativityLedger`

Proof. The comparison row Q is part of the accepted packet, so norm control is projected rather than inferred from an ambient ordered field oracle. It displays the product named by R and the norm values named by N.

Transport through H keeps the product endpoint and norm endpoints aligned over the same Banach carrier B. Replay through C preserves the route from $\mathsf{Ring}^{\uparrow}$ and $\mathsf{Norm}^{\uparrow}$ into $\mathsf{BanachAlgebra}^{\uparrow}$ before the comparison is exported.

The provenance row P and local naming row L seal that ledger as BEDC data. A $\mathsf{CStarAlg}^{\uparrow}$ consumer may reuse the ledger, but it must add its own involution and C^*-identity rows outside this theorem. □

Theorem 2189.4 (Banach algebra Cauchy product stability)**.** *Let $A = (R, N, B, Q, M, H, C, P, L)$ be an accepted* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier. Cauchy input rows transported by* hsame *through H and replayed by* Cont *through C remain inside the complete product carrier named by B, M, Q, P, L. In*

particular, a completion-facing product read cannot leave the $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, *and* $\mathsf{Norm}^{\uparrow}$ *sibling rows before the local* $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$ *exports data to a* $\mathsf{CStarAlg}^{\uparrow}$ *consumer.*

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraCauchyProductStability`

Proof. Start with Cauchy input rows read through the norm-induced metric of B and the completion seal M. The carrier requires these rows to return to the same Banach algebra surface before multiplication is replayed.

The product row R and norm row N are then read through the product-control ledger Q. Componentwise hsame transport through H prevents a transported Cauchy row from changing the displayed product or norm route.

Finally, Cont replay through C and provenance through P, L export only the complete product carrier. Hence the read remains inside the $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, and $\mathsf{Norm}^{\uparrow}$ sibling surface until a downstream $\mathsf{CStarAlg}^{\uparrow}$ packet supplies additional rows. □

Theorem 2189.5 (Banach algebra NameCert obligations)**.** *For an accepted* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier*

$$A = (R, N, B, Q, M, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed* $\mathsf{Ring}^{\uparrow}$ *multiplication row,* $\mathsf{Norm}^{\uparrow}$ *norm row,* $\mathsf{Banach}^{\uparrow}$ *complete normed carrier row, product-control row, completion seal, transport, replay, provenance, and local naming row;*
2. *algebra-norm compatibility, requiring the product row R and norm row N to share the carried additive carrier supplied by the Banach row B;*
3. *submultiplicative product control, requiring each product read to pass through the ledger Q before a norm estimate can be exported;*
4. *completion stability, requiring every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *read to return to the same carried algebra surface before multiplication is replayed;*
5. *classifier transport, requiring H and C to preserve the product, norm, Banach, completion, and product-control coordinates componentwise;*
6. *consumer non-escape, allowing* $\mathsf{CStarAlg}^{\uparrow}$ *to consume the complete normed algebra surface while excluding involution, C^*-identity, spectrum, representation, and host star-algebra data from the root packet.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2189.1. The accepted packet exposes exactly the multiplication, norm, Banach, product-control, completion, transport, replay, provenance, and local naming rows.

Algebra-norm compatibility is forced by the shared carrier. The RingUp multiplication row and NormUp row are read over the same additive carrier already packaged by the BanachUp row, so a product read and a norm read cannot move to different representatives without passing through the same hsame transport.

Submultiplicative control is the row Q. Every product estimate has to name the product from R, the norm values from N, and the comparison endpoint displayed by Q before it can be exported.

Completion stability is the row M. A completion-facing consumer first reads the $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ seal, and the seal returns to the same Banach carrier before multiplication is replayed.

Finally H, C, P, L transport, replay, source, and name only the listed rows. Since the excluded operator-algebra and host-completion principles have no coordinate in the carrier, they cannot appear as local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraNameCertObligations`

Theorem 2189.6 (Banach algebra completion-product nonescape)**.** *Let*

$$A = (R, N, B, Q, M, H, C, P, L)$$

be an accepted $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier. Every completion-facing product read exported by A factors through the finite route*

$$\mathsf{Ring}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Banach}^{\uparrow} \longmapsto Q \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{BanachAlgebra}^{\uparrow}.$$

Consequently a $\mathsf{CStarAlg}^{\uparrow}$ *consumer may cite the complete normed algebra surface only as displayed multiplication, norm, completeness, product-control, transport, replay, provenance, and local naming data. The route exports no involution, no C^*-identity, no spectrum, no representation, no weak-operator closure, no quotient completion, and no selected limit.*

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraCompletionProductNonescape`

Proof. Start with an accepted carrier and project its completion and product rows. The product row R and norm row N are already tied to the complete Banach row B by the carrier predicate.

The product-control row Q is the only place where multiplication and norm estimates meet. Thus a product read cannot pass directly from the RingUp row into a completion consumer; it must first expose the norm comparison ledger.

The completion seal M then checks the same norm-induced metric surface through $\mathsf{CompleteMetric}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$. Because M returns to the displayed Banach carrier, completion does not add a quotient product or a selected representative.

Replay through C and transport through H preserve the route componentwise, while P and L name only that finite surface. A $\mathsf{CStarAlg}^{\uparrow}$ consumer can therefore use the complete normed algebra boundary as a dependency, but its involution and C^*-identity rows must be supplied by the C*-algebra packet itself. □

Theorem 2189.7 (Banach algebra public export surface)**.** *Every complete normed algebra consumer of an accepted* $\mathsf{BanachAlgebra}^{\uparrow}$ *carrier reads only the displayed multiplication row, norm row, Banach completeness row, product-control ledger, completion seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$ *row. In carrier notation the public export is the finite surface*

$$A = (R, N, B, Q, M, H, C, P, L).$$

It does not export involution, a C^*-identity, spectrum, representation data, weak-operator closure, quotient completion, selected limits, host algebra equality, or any ambient normed-algebra oracle.

Lean4 ✓ `BEDC.Derived.BanachAlgebraUp.BanachAlgebraPublicExportSurface`

Proof. Project an accepted carrier onto $R, N, B, Q, M, H, C, P, L$. The earlier carrier, product, submultiplicativity, stability, obligation, and nonescape theorems show that every complete product read is already routed through these rows.

A public consumer therefore receives the multiplication from R, the norm from N, completeness from B and M, and product control from Q. The rows H, C, P, L transport, replay, source, and name that same finite surface.

No excluded operator-algebra or host-completion coordinate appears in the carrier predicate. Since there is no row for such data, the public export cannot expose it. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachAlgebra}^{\uparrow}$ packet is generated from a displayed RingUp multiplication row, a NormUp norm row over the same carried additive carrier, a BanachUp complete normed carrier row, a product-control ledger, a CompleteMetricUp and MetricCompletionUp completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}, \mathsf{publicClosed})$

Scope closed: Definition 2189.1, Definition 2189.5, Definition 2189.3, Definition 2189.6, and Definition 2189.7 display the finite complete normed algebra public surface over RingUp multiplication, NormUp norm, BanachUp completeness, completion seals, product control, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim involution, the C^*-identity, spectrum, representation theory, weak-operator closure, host algebra equality, quotient completion, selected limits, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, complete product carrier routing, submultiplicativity ledger exposure, Cauchy product stability, NameCert obligation coverage, completion-product nonescape, and public export coverage for BEDC.Derived.BanachAlgebraUp.

Chapter 2190

A Concrete Naming Certificate for $\mathsf{MooreSmithCompletion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{MooreSmithCompletion}^{\uparrow}$ packet records the directed-net completion route as finite BEDC NameCert data. It sits after the directed-index substrate of chapter 1607, the Cauchy-net surface of chapter 589, and the Moore–Smith convergence boundary of chapter 1521; it is read by the metric completion interface of chapter 1506, the uniform completion boundary of chapter 1473, and the separated comparison row of chapter 1411. Its numerical readback is only the finite regular-rational and real-name route of chapter 301 and chapter 38. The packet names the completion-facing net interface without selecting limits, quotienting nets, assuming ambient completeness, invoking countable choice, or asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2190.1 (Moore–Smith completion carrier)**.** *A* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, C, G, Q, S, R, A, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index substrate, including the finite comparison windows and common upper-window witnesses used by the net route.*
2. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row over D, recording the displayed Cauchy windows before any completion-facing read occurs.*
3. *G is the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *convergence row, tying the same directed windows to a visible boundary request.*
4. *Q is the completion boundary row read by* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumers.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance comparison row available when two displayed boundary reads must be compared.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite tolerance readback row attached to the retained net windows.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after D, C, G, Q, S, R have been displayed.*
8. *H records componentwise* hsame *transport for directed indices, Cauchy-net tails, convergence rows, completion-boundary reads, separated comparison rows, readback rows, replay data, provenance, and local naming.*

9. *K records displayed* Cont *replay from directed windows through Cauchy-net evidence, Moore–Smith convergence, completion-boundary reads, separated comparison, regular rational readback, and real-name sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, a quotient net, a host completion object, ambient completeness, countable choice, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MooreSmithCompletionUp.MooreSmithCompletionTasteGate single carrier alignment`

Definition 2190.2 (Moore–Smith completion classifier)**.** *Two accepted* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *packets are classified as the same displayed completion read exactly when their directed substrates, Cauchy-net tails, Moore–Smith convergence rows, completion-boundary rows, separated comparison rows, regular rational readbacks, and terminal real seals are related componentwise by the displayed* hsame *transports and replayed by the same* Cont, Pkg, *and* NameCert *route. The classifier compares boundary reads through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance row when such a comparison is requested; it does not compare host completion points, quotient classes of nets, or selected representatives.*

Theorem 2190.3 (Moore–Smith completion NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *carrier $M = (D, C, G, Q, S, R, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed substrate, Cauchy-net tail row, Moore–Smith convergence row, completion boundary row, separated comparison row, finite readback row, real seal, transport, replay, provenance, and local naming row;*

2. *directed-net coverage, requiring every completion-facing request to begin with the displayed* $\mathsf{DirectedSet}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *rows;*

3. *convergence boundary exposure, requiring G and Q to be read before any metric or uniform completion consumer receives the packet;*

4. *separated comparison discipline, requiring all comparison of two displayed boundary reads to pass through S rather than host equality or quotient equality;*

5. *readback discipline, requiring every numerical tolerance request to pass through R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal A is inspected;*

6. *structural transport, requiring H, K, P, N to preserve and name only the displayed route;*

7. *ledger non-escape, excluding selected limits, quotient nets, ambient completeness, countable choice, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2190.1. The accepted packet exposes exactly the directed-index, Cauchy-net, convergence, boundary, separatedness, readback, real-seal, and structural rows.

Directed-net coverage follows from the first two rows. A completion-facing request cannot reach G, Q, S, R, A until the directed substrate D and Cauchy-net evidence C have been displayed.

The convergence and completion boundary obligations are the roles of G and Q. These rows place the Moore–Smith convergence read before the metric or uniform completion consumer receives any boundary data.

Separated comparison and readback discipline are local to S and R. The classifier of Definition 2190.2 uses S for zero-distance comparison, and every numerical read uses R before the terminal $\mathsf{Real}^{\uparrow}$ seal A.

Finally H, K, P, N transport, replay, source, and name only the listed coordinates. Since the excluded completion principles have no carrier coordinate, they cannot appear as local NameCert obligations. □

Theorem 2190.4 (Moore–Smith net filter completion route)**.** *Let $M = (D, C, G, Q, S, R, A, H, K, P, N)$ be an accepted* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *carrier, and let a completion-facing L10 read of M display the same directed tail windows as a* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *bridge. Then every Cauchy-net read exported by M factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow} \longmapsto \mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The completion consumer may inspect a limit boundary only after the directed-index rows D, the retained finite tail windows in C, the Cauchy-filter bridge, the $\mathsf{StreamName}^{\uparrow}$ *window schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal A have all been displayed through H, K, P, N. The route exports no quotient equality of nets or filters, selected limit, host completion point, countable-choice schedule, or ambient completeness theorem.*

Proof. Project the accepted carrier. The rows D and C expose the directed index substrate and the Cauchy-net tail evidence before the convergence row G or completion boundary row Q is read.

Now route the displayed tail windows through the Cauchy-filter bridge. The bridge hypothesis says that the same finite windows are served by the Cauchy-filter completion criterion surface, so the read passes from the net tail row into $\mathsf{CauchyFilter}^{\uparrow}$ data without quotienting nets or filters.

The L10 numerical handoff is then forced through finite names. The bridge supplies the $\mathsf{StreamName}^{\uparrow}$ inspection schedule, while the Moore–Smith carrier retains the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R and the terminal $\mathsf{Real}^{\uparrow}$ seal A as downstream coordinates of the same route.

Finally the structural rows H, K, P, N preserve, replay, source, and name only these displayed coordinates. Since no carrier coordinate contains a selected limit, quotient equality, host completion point, choice schedule, or ambient completeness theorem, a completion consumer can read a limit boundary only through the route in the statement. □

Theorem 2190.5 (Moore–Smith completion net boundary)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *packet reads the boundary through the finite route*

$$D \longmapsto C \longmapsto G \longmapsto Q.$$

Thus the completion boundary is exposed only after the displayed $\mathsf{DirectedSet}^{\uparrow}$ *index substrate, the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row, and the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *convergence row have been read. The packet provides no selected Moore–Smith limit, quotient of nets, ambient complete object, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a completion-facing consumer and project the carrier rows. The directed substrate D names the finite index windows and upper-window witnesses, while C records the Cauchy-net tail evidence over those windows.

The convergence row G is the next boundary-facing row. It ties the same directed windows to the Moore–Smith convergence test before any completion boundary is exported.

The row Q is therefore downstream of D, C, G. Any $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{UniformCompletion}^{\uparrow}$ consumer reads the boundary from Q with the same transport, replay, provenance, and local name recorded by H, K, P, N.

No carrier coordinate stores a selected limit, a quotient net, an ambient complete object, a countable-choice schedule, or a $\mathsf{Real}^{\uparrow}$ completeness theorem. The exported boundary is exactly the displayed directed-net route. □

Theorem 2190.6 (Moore–Smith completion nonescape)**.** *For an accepted* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *carrier* $M = (D, C, G, Q, S, R, A, H, K, P, N)$*, every comparison, tolerance read, and terminal real-name read exported by* M *is exhausted by*

$$D, C, G, Q, S, R, A, H, K, P, N.$$

Consequently the packet cannot introduce selected limits, quotient nets, ambient completeness, countable choice, host equality of completion points, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Comparison requests use the classifier of Definition 2190.2. That classifier compares displayed completion reads through componentwise transport and the separated zero-distance row S.

Tolerance requests use R. Since R is downstream of D, C, G, Q, S, each numerical read remains attached to the same directed-net completion route.

Terminal real-name reads use A only after the regular rational readback has been displayed. Thus A is a seal on the same finite route, not an ambient real-completeness principle.

The structural rows H, K, P, N preserve, replay, source, and name only these coordinates. Since no other completion or choice data is present in the carrier, the listed rows exhaust the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCompletion}^{\uparrow}$ packet is generated from a DirectedSetUp substrate, CauchyNetUp tail evidence, MooreSmithConvergenceUp boundary evidence, MetricCompletionUp and UniformCompletionUp completion reads, SeparatedMetricUp zero-distance comparison, RegSeqRatUp readback, a RealUp seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2190.1, Definition 2190.2, Definition 2190.3, Definition 2190.4, Definition 2190.5, and Definition 2190.6 seed the finite directed-net completion route between Moore–Smith convergence and metric or uniform completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient nets, ambient completeness, countable choice, host equality of completion points, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, directed-net coverage, convergence boundary exposure, separated comparison discipline, finite readback discipline, transport preservation, and nonescape for BEDC.Derived.MooreSmithCompletionUp.

Chapter 2191

A Concrete Naming Certificate for $\mathsf{OpenBall}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{OpenBall}^{\uparrow}$ packet records the open metric neighbourhood around a displayed centre as finite BEDC NameCert data. It sits after the metric source of chapter 57, the real-name and dyadic-radius routes of chapter 38, the located-distance discipline of chapter 1533, and the boundary comparison supplied by chapter 1451. The packet gives topology and continuity consumers a reusable neighbourhood row without importing host set openness, quotient point equality, selected radii, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2191.1 (OpenBall carrier)**.** *An* $\mathsf{OpenBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, C, R, X, D, B, H, Q, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the point classifier, metric-distance operation, symmetry, triangle, and zero-distance discipline available from the displayed metric certificate.*
2. *C is the centre point row, classified as a point of the carrier named by M.*
3. *R is the positive radius row, read either as a* $\mathsf{Real}^{\uparrow}$ *radius name or as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger sealed into the* $\mathsf{Real}^{\uparrow}$ *order surface.*
4. *X is the point row for the point whose neighbourhood membership is being tested.*
5. *D is the metric-distance witness row for the displayed distance value $d(C, X)$ over the metric source M.*
6. *B is the distance-below-radius ledger, recording the constructive strict comparison*

$$d(C, X) <_{\mathsf{R}} R.$$

7. *H records componentwise* hsame *transport for the metric source, centre, radius, point, distance witness, strict comparison, replay data, provenance, and local name.*
8. *Q records displayed* Cont *replay of distance evaluation, Real or dyadic radius readback, strict comparison, boundary comparison with* $\mathsf{MetricClosedBall}^{\uparrow}$*, and neighbourhood handoff to later topology and continuity consumers.*

9. *P is the* Pkg *provenance over* $\mathsf{OpenBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}$ *row.*

The classifier accepts exactly the rows $M, C, R, X, D, B, H, Q, P, N$. It compares accepted packets only through the displayed metric classifier, radius readback, strict ordered-real comparison, structural transports, replay route, provenance, and local naming row.

Lean4 ✓ `BEDC.Derived.OpenBallUp.OpenBallTasteGate single carrier alignment`

Theorem 2191.2 (OpenBall NameCert obligations)**.** *For an accepted* $\mathsf{OpenBall}^{\uparrow}$ *carrier*

$$O = (M, C, R, X, D, B, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, centre-point admission, positive radius admission through the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *route, point admission, distance-witness readback, strict distance-below-radius comparison, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and ledger non-escape for the displayed rows $M, C, R, X, D, B, H, Q, P, N$.*

Proof. Carrier admission projects the finite packet. The metric source row M supplies the carried point classifier and distance operation, while C and X are admitted only as points over that source.

Radius admission reads R through the displayed real or dyadic route. A dyadic radius is consumed only after its finite ledger is sealed into the ordered $\mathsf{Real}^{\uparrow}$ comparison surface.

The distance and membership obligations are the rows D and B. The row D names the distance value $d(C, X)$, and B records the strict comparison $d(C, X) <_{\mathsf{R}} R$ that makes the point an open-neighbourhood member.

Finally H, Q, P, N transport, replay, source, and name only these rows. Since no other coordinate is present, the packet cannot export host openness, quotient point equality, selected radii, compactness, or ambient real completeness. □

Theorem 2191.3 (OpenBall metric neighbourhood handoff)**.** *Every topology or continuity consumer that reads an* $\mathsf{OpenBall}^{\uparrow}$ *neighbourhood receives the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto (C, R, X) \longmapsto D \longmapsto B \longmapsto \mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}.$$

The handoff factors through the metric source of chapter 57, the located-distance discipline of chapter 1533, and the boundary comparison of chapter 1451; it does not supply a host open set, a generated topology theorem, a compactness argument, a selected limit, or quotient equality of metric points.

Proof. Start with a consumer request for an open metric neighbourhood. The packet first exposes M, so the request is tied to the displayed metric carrier and its point classifier before any distance comparison is read.

The centre, radius, and tested point are then the rows C, R, X. The located metric route supplies the distance witness D, and the strict ledger B turns that witness into the membership comparison.

Boundary comparison with $\mathsf{MetricClosedBall}^{\uparrow}$ is available only as a comparison route in Q. The open packet keeps the strict row as the exported neighbourhood witness; the closed-ball chapter remains the sibling non-strict boundary, not a replacement for the open membership comparison.

Thus topology and continuity consumers receive exactly the finite neighbourhood route named by N and replayed by Q. The carrier contains no coordinate from which host openness, generated topology closure, compactness, selected limits, or quotient point equality can be recovered. □

Theorem 2191.4 (OpenBall real-radius boundary)**.** *For an accepted* $\mathsf{OpenBall}^{\uparrow}$ *packet* $O = (M, C, R, X, D, B, H, Q, P, N)$*, every radius-sensitive read is exhausted by the positive* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* R*, the distance witness* D*, and the strict comparison ledger* B*. The packet does not assert completeness of* $\mathsf{Real}^{\uparrow}$*, trichotomy of real order, a selected rational approximation schedule, or equality of host real values.*

Proof. Inspect the radius row R. If it is read as a real radius, the packet uses only the displayed ordered-real comparison needed by B; if it is read through the dyadic route, the dyadic ledger is first sealed into the same comparison surface.

The distance witness D supplies the other endpoint of the comparison. The membership row is therefore the strict ledger B, not a separate real-number construction or a host subset predicate.

Transport through H and replay through Q preserve this finite comparison route. Provenance P and local naming N identify the route as an $\mathsf{OpenBall}^{\uparrow}$ certificate, so no real-completeness theorem, trichotomy oracle, selected approximation schedule, or host equality of real values is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OpenBall}^{\uparrow}$ packet is generated from a MetricUp source row, a centre point row, a positive RealUp or DyadicRatCoreUp radius row, a point row, a MetricDistanceWitness row, a strict distance-below-radius ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}$, seed)

Scope closed: Definition 2191.1, Definition 2191.2, Definition 2191.3, and Definition 2191.4 seed the finite open-neighbourhood route over metric distance and positive real or dyadic radius data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OpenBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host set openness, generated topology closure, compactness, quotient equality for metric points, selected radii, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, positive-radius admission, distance-witness readback, strict comparison transport, neighbourhood handoff, real-radius boundary, and ledger non-escape for BEDC.Derived.OpenBallUp.

Chapter 2192

A Concrete Naming Certificate for $\mathsf{MetricCompletionFunctor}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop-style completion functoriality, the $\mathsf{MetricCompletionFunctor}^{\uparrow}$ packet records the L10 route that transports uniformly continuous metric maps across the displayed completion surface. It specializes the metric-completion carrier of chapter 1506, the functorial transport surface of chapter 426, and the Cauchy-name functor packet of chapter 423. Its cofinal evidence is read through the Cauchy-filter branch of chapter 333 or the Cauchy-net branch of chapter 589; the packet itself adds no quotient completion, selected representative, host completed metric space, or ambient completeness principle.

Definition 2192.1 (MetricCompletionFunctor carrier)**.** *A* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M_0, M_1, G, E, T, H, C, P, N)$$

with the following visible rows:

1. *M_0 is the source* $\mathsf{MetricCompletion}^{\uparrow}$ *row, including the displayed metric source, its Cauchy-filter or Cauchy-net cofinal evidence, finite readback, and separated-limit comparison row.*
2. *M_1 is the target* $\mathsf{MetricCompletion}^{\uparrow}$ *row with the same kind of displayed completion evidence and readback surface.*
3. *G is the metric-map row whose finite modulus and graph ledger send source observations to target observations before completion-facing reads are taken.*
4. *E is the extension row inherited from the* $\mathsf{CompletionFunctor}^{\uparrow}$ *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *surfaces, routing the displayed map through the two completion rows.*
5. *T is the functoriality ledger comparing identity and composition reads by finite* hsame *transport over M_0, M_1, G, and E.*
6. *H records componentwise* hsame *transport for the two metric-completion rows, the map row, the extension row, the functoriality ledger, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$M_0, G, M_1 \longmapsto E \longmapsto T, H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctor}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected limits, host function equality, a host completed metric space, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricCompletionFunctorUp.MetricCompletionFunctorTasteGate single carrier alignment`

Theorem 2192.2 (MetricCompletionFunctor NameCert obligations)**.** *For an accepted* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *carrier* $F = (M_0, M_1, G, E, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, source and target metric-completion exactness, map-modulus compatibility, extension-ledger exactness, identity and composition functoriality, classifier transport, and non-escape of quotient completions, selected limits, host function equality, countable choice, and ambient completeness. Every obligation is read from the displayed rows* $M_0, M_1, G, E, T, H, C, P, N$ *and their finite* hsame *and* Cont *routes.*

Proof. Carrier admission is projection from Definition 2192.1. The accepted packet exposes exactly the two metric-completion rows, the displayed map ledger, the extension row, functoriality ledger, transport rows, replay rows, provenance, and local naming row.

The source and target exactness clauses require every completion-facing read to pass through M_0 or M_1 before the extension row is inspected. The map-modulus clause keeps the graph ledger G attached to the finite source observations and target observations already visible in those rows.

Extension exactness is the coordinate E. It is read from the existing completion-functor and Cauchy-completion-functor surfaces, so no second extension route or host function equality is available.

Identity and composition are the public roles of T, transported componentwise by H and replayed through C. The provenance and local naming rows then exhaust the packet, leaving no coordinate for quotient completion equality, selected representatives, countable choice, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.MetricCompletionFunctorUp.MetricCompletionFunctorCarrier namecert obligations`

Theorem 2192.3 (MetricCompletionFunctor L10 route)**.** *Every L10 completion-route consumer of an accepted* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *packet factors through the finite chain*

$$M_0, G, M_1 \longmapsto E \longmapsto T, H, C, P, N.$$

The route first reads the source $\mathsf{MetricCompletion}^{\uparrow}$ *packet, then the displayed map ledger, then the target* $\mathsf{MetricCompletion}^{\uparrow}$ *packet, and only then the extension and functoriality rows supplied by the completion-functor surfaces. It therefore connects* $\mathsf{MetricCompletion}^{\uparrow}$ *to* $\mathsf{CompletionFunctor}^{\uparrow}$ *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *by displayed finite data rather than by a quotient completion object or a host-level completed metric functor.*

Proof. Project the carrier of Definition 2192.1. The first completion-facing coordinate is M_0, so the consumer begins with the source metric-completion evidence, including its filter or net branch and finite readback row.

The next analytic coordinate is the map ledger G. Its finite modulus and graph rows move displayed source observations to target observations before any extension read is taken.

The target completion row M_1 supplies the completion-facing landing surface. The extension row E then repacks the displayed map through the completion-functor and Cauchy-completion-functor interfaces, while the functoriality ledger T records identity and composition comparisons as finite transport.

The rows H, C, P, N preserve and name the same route. Since the carrier has no quotient, selected-limit, host-function-equality, or ambient-completeness coordinate, every L10 consumer receives only the displayed finite route. □

Theorem 2192.4 (MetricCompletionFunctor map-modulus compatibility)**.** *For an accepted* MetricCompletionFunc *carrier* $F = (M_0, M_1, G, E, T, H, C, P, N)$*, every consumer read of map-modulus compatibility is forced through the displayed map row* G *of Definition 2192.1. The source observations come from* M_0*, the target observations land in* M_1*, and the modulus row is read before the extension row* E *can be consumed. No consumer may replace this finite row by host uniform continuity or a function-space equality.*

Proof. Read the carrier coordinates. The only row that relates source observations to target observations is G, and it is displayed between the source and target completion rows M_0 and M_1.

The extension row E is downstream of that graph and modulus data. A consumer therefore cannot request an extension read until the finite modulus and graph ledger in G have already been exposed.

Transport and replay are supplied by H and C, with provenance and naming through P and N. Since no host function-space row exists in the carrier, the compatibility claim is exactly the visible finite row. □

Theorem 2192.5 (MetricCompletionFunctor extension-ledger exactness)**.** *For an accepted* MetricCompletionFunct *carrier* $F = (M_0, M_1, G, E, T, H, C, P, N)$*, every extension read is exactly the displayed row* E *named in Definition 2192.1. The row* E *must be read after the source completion row* M_0*, the metric-map row* G*, and the target completion row* M_1*, and it may only export the completion handoff already visible through* $\mathsf{CompletionFunctor}^{\uparrow}$ *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rows.*

Proof. Project the route in Definition 2192.3. The extension coordinate is not an independent host map; it is the displayed row encountered after the two completion packets and the map ledger have been read.

Exactness means that a consumer cannot choose another extension witness or compare two extensions by host function equality. The only usable extension data are the entries of E, transported by H and replayed by C.

The provenance row P records the source of the completion-functor handoff, and N names the local certificate. These rows exhaust the extension ledger visible to downstream completion consumers. □

Theorem 2192.6 (MetricCompletionFunctor identity and composition functoriality)**.** *For an accepted* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *carrier* $F = (M_0, M_1, G, E, T, H, C, P, N)$*, every identity or composition read is forced through the functoriality ledger* T *displayed in Definition 2192.1. Identity reads compare the visible completion row with its own displayed map ledger, and composition reads compare two displayed map rows only after their extension rows have been read. The comparison is finite* hsame *transport, not host equality of completed maps.*

Proof. The route theorem places T after the extension row E. Thus identity and composition are not prerequisites for constructing a host functor; they are ledger reads over already displayed completion and extension data.

For identity, T records the finite transport comparing the source completion row with the identity map ledger. For composition, T records the finite transport comparing the two-stage displayed map read with the displayed composite read after their extension rows are available.

The rows H, C, P, N transport, replay, source, and name those comparisons. Because the carrier has no host completed-map equality coordinate, the functoriality surface is exactly the finite ledger T. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionFunctor}^{\uparrow}$ packet is generated from source and target MetricCompletionUp rows, a displayed metric-map ledger, a completion-functor extension row, a functoriality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2192.1, Definition 2192.2, and Definition 2192.3 seed the functorial L10 transport route between metric completion and the existing completion-functor surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host completed metric spaces, host function equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target metric-completion exactness, map-modulus compatibility, extension-ledger exactness, functoriality, classifier transport, and non-escape for BEDC.Derived.MetricCompletionFunctorUp.

Chapter 2193

A Concrete Naming Certificate for $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$

Following Bishop and Bridges, strongly extensional maps on real names are read as graph certificates that preserve apartness and RegSeqRat classifier data. The $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$ packet connects the regular Cauchy real-name surface of chapter 38, the rational readback discipline of chapter 301, the finite observation windows of chapter 126, and the dyadic tolerance ledger of chapter 291. It records only displayed finite graph rows and transport obligations; it is not a host real function, a quotient-function equality, a trichotomy principle, an open-cover compactness object, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2193.1 (Strongly extensional real-map carrier)**.** *A* $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, G, A, S, R, D, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *name row, including its regular Cauchy presentation and finite observation boundary.*
2. *Y is the target* $\mathsf{Real}^{\uparrow}$ *name row reached after the displayed graph route is consumed.*
3. *G is the finite graph certificate, pairing inspected source windows with inspected target windows.*
4. *A is the apartness-preservation ledger: every displayed apartness witness between source names is routed to a displayed apartness witness between target names.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the graph certificate before any regular rational readback is taken.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier-readback row attached to the source and target windows in S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the displayed rational approximants and apartness margins.*
8. *H records componentwise* hsame *transport for the two real-name rows, graph row, apartness ledger, stream windows, rational readbacks, dyadic tolerances, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the route*

$$X, G, S, R, D \longmapsto Y, A, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{StronglyExtensionalRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, G, A, S, R, D, H, C, P, N$. It has no coordinate for host functions, quotient equality of real names, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.StronglyExtensionalRealMapUp.StronglyExtensionalRealMapTasteGate single carrier alignment`

Theorem 2193.2 (Strongly extensional real-map NameCert obligations)**.** *For every accepted* $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$ *carrier $F = (X, Y, G, A, S, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StronglyExtensionalRealMap}^{\uparrow}}$ *surface consists exactly of carrier exposure, source and target real-name exposure, graph-window exactness, apartness-preservation exposure,* $\mathsf{StreamName}^{\uparrow}$ *window discipline,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance service, classifier transport, and non-escape of host functions, quotient equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every obligation is read from $X, Y, G, A, S, R, D, H, C, P, N$ and their displayed* hsame *and* Cont *routes.*

Proof. Project the carrier of Definition 2193.1. The visible inventory is exactly the two real-name rows, graph row, apartness ledger, stream windows, regular rational readbacks, dyadic tolerance ledger, transport, replay, provenance, and local naming.

Graph-window exactness is forced through G and S: each inspected source window is paired with a displayed target window before any target real-name row is consumed. The apartness-preservation obligation is the row A, and it records only displayed apartness witnesses rather than a global host-function property.

Regular rational classifier data then pass through R, while dyadic tolerance comparisons pass through D. Transport by H and replay by C preserve the same finite route, and P, N source and name it locally. Since no carrier coordinate names a host function, quotient equality, trichotomy, compact cover, or ambient completeness theorem, none can be exported by the local certificate. □

Theorem 2193.3 (Strongly extensional real-map RegSeqRat transport)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$ *carrier factors through the finite chain*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the graph row G, apartness ledger A, and structural rows H, C, P, N. If two displayed source reads are classifier-equal on the same finite regular Cauchy window, then their transported target reads are compared only by the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ *row and dyadic tolerance ledger in the carrier. The theorem does not assert equality of host functions or equality of ambient real-number quotients.*

Proof. Start from the accepted carrier $F = (X, Y, G, A, S, R, D, H, C, P, N)$. A regular-rational consumer first reads the source real-name row X and then the stream windows S selected by the graph certificate G.

The classifier readback is the row R. Any comparison of source reads on the same finite regular Cauchy window must therefore pass through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ classifier data before reaching a target read.

The dyadic tolerance ledger D supplies the numeric comparison margins, while the apartness ledger A records the strongly extensional direction on displayed witnesses. The structural

rows H, C, P, N transport, replay, source, and name the same route, so the transported target comparison remains a finite RegSeqRat-and-dyadic read rather than a host-function equality or quotient-real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{StronglyExtensionalRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{StronglyExtensionalRealMap}^{\uparrow}$ packet is generated from source and target RealUp name rows, a finite graph certificate, an apartness-preservation ledger, StreamName windows, RegSeqRat classifier readback, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StronglyExtensionalRealMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2193.1, Definition 2193.2, and Definition 2193.3 seed the finite graph-certificate route for strongly extensional maps on regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StronglyExtensionalRealMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, host function equality, quotient equality of real names, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph-window exactness, apartness-preservation exposure, StreamName window discipline, RegSeqRat classifier transport, DyadicRatCore tolerance service, and non-escape for BEDC.Derived.StronglyExtensionalRealMapUp.

Chapter 2194

A Concrete Naming Certificate for $\mathsf{EulerTransform}^{\uparrow}$

Following Euler and Hardy on divergent series and Bishop–Bridges constructive analysis, the $\mathsf{EulerTransform}^{\uparrow}$ packet records finite Euler acceleration for alternating real series as a displayed BEDC route. It consumes the source-series surface of chapter 1581, the alternating-sign route of chapter 1483, the Abel and Dirichlet neighboring handoffs of chapter 1496 and chapter 1497, the summable tail budget of chapter 1799, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet supplies a finite acceleration boundary for Real and Completion series consumers; it is not a host infinite sum, selected limit, quotient series equality, classical convergence transfer, ambient completeness theorem, Lean verification, or bridge checking.

Definition 2194.1 (Euler transform carrier)**.** *An* $\mathsf{EulerTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, A, D, T, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RealSeries}^{\uparrow}$ *row, exposing only the inspected real-series windows and partial-sum windows used by the current alternating-tail request.*
2. *A is the alternating-sign row, recording the finite parity and sign windows consumed by the* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *surface.*
3. *D is the finite forward-difference triangle, displaying the finitely many iterated difference rows used by the Euler acceleration step.*
4. *T is the transformed tail-window row, naming the accelerated tail windows obtained from S, A, D.*
5. *B is the summable tail and modulus budget, requiring the transformed windows to be served through* $\mathsf{SummableTailModulus}^{\uparrow}$ *before any endpoint read is available.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff reached only after the transformed tail windows and the summable budget have been displayed.*
7. *H stores componentwise* hsame *transport for the source, alternating, difference, transformed-window, budget, readback, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a tail request through D, T, B, R.*
9. *P is the* Pkg *provenance over* $\mathsf{EulerTransform}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EulerTransform}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, D, T, B, R, H, C, P, N$. It has no coordinate for host infinite sums, selected limits, quotient series equality, classical convergence transfer, ambient $\mathsf{Real}^\uparrow$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EulerTransformUp.EulerTransformTasteGate single carrier alignment`

Theorem 2194.2 (Euler transform NameCert obligations)**.** *For every accepted* $\mathsf{EulerTransform}^\uparrow$ *carrier*

$$E = (S, A, D, T, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EulerTransform}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the source* $\mathsf{RealSeries}^\uparrow$ *row, alternating-sign row, finite forward-difference triangle, transformed tail windows, summable tail budget,* $\mathsf{RegSeqRat}^\uparrow$ *handoff, transport, replay, provenance, and local naming row;*
2. *alternating-window exactness, requiring A to supply the only sign and parity data used by the acceleration step;*
3. *difference-triangle stability, requiring each displayed forward-difference row in D to be read from the same finite source windows in S and transported componentwise by H;*
4. *transformed-tail coherence, requiring T to name only windows obtained from the displayed difference triangle and alternating route;*
5. *budget service, requiring B to provide the summable tail and modulus boundary before the regular rational handoff R can be consumed;*
6. *neighboring-test discipline, requiring Abel and Dirichlet rows to appear only as provenance faces for the same tail-window route and not as independent endpoint seals;*
7. *nonescape, requiring H, C, P, N to transport, replay, source, and name exactly this route and no unrecorded analytic principle.*

Proof. Project the carrier of Definition 2194.1. The visible inventory is $S, A, D, T, B, R, H, C, P, N$, so the local certificate has no detached row for a host sum, selected limit, quotient equality, classical convergence transfer, or ambient completeness theorem.

The alternating and difference data are read before any transformed endpoint is available. The row A supplies the finite sign and parity windows, while D displays only the forward-difference triangle computed from the same inspected source windows in S.

The transformed-tail row is then forced through T. It can name only acceleration windows obtained from S, A, D, and the summable tail budget B is the only row that turns those windows into a modulus-bearing endpoint service.

Finally, R, H, C, P, N supply regular readback, transport, replay, provenance, and local naming for the same route. The Abel and Dirichlet faces appear only in the provenance of the shared tail-window discipline, so no independent endpoint seal or stronger convergence principle is exported. □

Theorem 2194.3 (Euler transform alternating acceleration handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{EulerTransform}^\uparrow$ *carrier factors through the ordered route*

$$\mathsf{RealSeries}^\uparrow \longmapsto \mathsf{AlternatingSeriesTest}^\uparrow \longmapsto \mathsf{SummableTailModulus}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The Abel and Dirichlet rows may be read only as neighboring provenance for the same transformed tail-window route. No endpoint-facing consumer reaches $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *before the finite forward-difference triangle and summable tail budget have been displayed.*

Proof. Start from the accepted carrier $E = (S, A, D, T, B, R, H, C, P, N)$. The only source row is S, and the only alternating information is the sign-window row A.

The acceleration handoff is carried by D and T. The finite difference triangle is read from the displayed source windows, and the transformed tail windows are the only rows passed downstream.

The budget row B supplies the summable tail and modulus service for those transformed windows. Only after that service appears may the route enter R and then the terminal $\mathsf{Real}^{\uparrow}$ read. Since Abel and Dirichlet data occur only inside the provenance row P, they cannot replace the displayed alternating, difference, budget, or regular-rational rows. □

Theorem 2194.4 (Euler transform Real seal nonescape)**.** *For every accepted* $\mathsf{EulerTransform}^{\uparrow}$ *carrier* $E = (S, A, D, T, B, R, H, C, P, N)$*, any consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *endpoint seal without the displayed source-series windows, alternating-sign row, finite forward-difference triangle, transformed tail windows, summable tail budget, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff has left the local* $\mathsf{NameCert}_{\mathsf{EulerTransform}^{\uparrow}}$ *surface.*

Proof. Read the obligation rows in Definition 2194.2. Endpoint-facing data must pass through carrier exposure, alternating-window exactness, difference-triangle stability, transformed-tail coherence, budget service, regular readback, transport, replay, provenance, and naming.

The only Real-facing route named by the carrier is downstream of R. Therefore a consumer that omits any of S, A, D, T, B, R is asking for a row absent from the carrier and absent from the local certificate. The rows H, C, P, N preserve the same route rather than adding host sums, selected limits, quotient equality, classical convergence transfer, ambient completeness, Lean verification, or bridge data. □

Falsifiable prediction: Every accepted $\mathsf{EulerTransform}^{\uparrow}$ endpoint-facing read factors through source RealSeries windows, alternating-sign windows, a finite forward-difference triangle, transformed tail windows, a SummableTailModulus budget, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, alternating-series-test, abel-test, dirichlet-test, summable-tail-modulus, regseqrat, real: $\mathsf{EulerTransform}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, alternating signs, finite forward differences, transformed tail windows, summable tail budgets, regular rational readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerTransform}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EulerTransform}^{\uparrow}$ packet is generated from source RealSeries windows, alternating-sign rows, a finite forward-difference triangle, transformed tail windows, a SummableTailModulus budget, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EulerTransform}^{\uparrow}}$, seed)

Scope closed: Definition 2194.1, Definition 2194.2, Definition 2194.3, and Definition 2194.4 seed the finite Euler acceleration boundary over displayed source series, alternating signs, forward differences, transformed tail budgets, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EulerTransform}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.EulerTransformUp.EulerTransformTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient series equality, classical convergence transfer, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier exposure, alternating-window exactness, difference-triangle stability, transformed-tail coherence, summable budget service, regular-rational handoff, Real seal nonescape, and transport, replay, provenance, and local naming discipline for BEDC.Derived.EulerTransformUp.

Chapter 2195

A Concrete Naming Certificate for BishopCompactnessCriterion$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the BishopCompactnessCriterion$^{\uparrow}$ packet records the compactness criterion for metric spaces as a finite-net plus complete-limit BEDC route. It consumes the compact metric surface of chapter 129, the total-bounded finite-net rows of chapter 127, the complete-metric limit-readback rows of chapter 128, and the completion-facing landing surface of chapter 1506. The packet gives L10 compact metric consumers a constructive criterion surface without invoking open-cover compactness, host sequential compactness, countable choice, or an ambient selected completion.

Definition 2195.1 (Bishop compactness criterion carrier)**.** *A* BishopCompactnessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, F, C, L, Q, H, R, P, N)$$

with the following displayed rows:

1. *M is the* CompactMetric$^{\uparrow}$ *source row, exposing the metric carrier and the public compact-metric certificate surface.*
2. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row, including the requested tolerance, finite* ProbeBundle *centers, membership constructors, coverage distances, and rational comparison ledgers.*
3. *F is the finite-net handoff row reading the centers and coverage witnesses from T as the compactness criterion input rather than as an external subcover.*
4. *C is the* CompleteMetric$^{\uparrow}$ *row for regular Cauchy streams over the same metric carrier.*
5. *L is the limit-readback row supplied by C, including the stream name, modulus, displayed limit history, late-observation thresholds, metric-distance witnesses, and rational comparison ledgers.*
6. *Q is the* MetricCompletion$^{\uparrow}$ *handoff row used by completion-facing consumers of the same finite-net and complete-limit data.*
7. *H records componentwise* hsame *transport for M, T, F, C, L, Q and the local certificate row.*
8. *R records displayed* Cont *replay from a compactness request through finite-net extraction, complete-limit readback, metric-completion handoff, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* BishopCompactnessCriterion$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* TotallyBounded$^{\uparrow}$*,* CompleteMetric$^{\uparrow}$*,* MetricCompletion$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactnessCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, T, F, C, L, Q, H, R, P, N$. It has no coordinate for open-cover compactness, a selected subsequence, countable choice, quotient completion equality, host compactness, or an ambient complete metric space.

Lean4 ✓ `BEDC.Derived.BishopCompactnessCriterionUp.BishopCompactnessCriterionTasteGate single carrier alignment`

Theorem 2195.2 (Bishop compactness criterion NameCert obligations)**.** *For every accepted* BishopCompactnessCri *carrier*

$$K = (M, T, F, C, L, Q, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompactnessCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the compact metric source, total-bounded finite-net row, finite-net handoff, complete-metric row, limit-readback row, metric-completion handoff, transport, replay, provenance, and local naming rows;*
2. *compact-source lock, requiring M to be the common metric carrier read by both the* $\mathsf{TotallyBounded}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *faces;*
3. *finite-net exactness, requiring T and F to expose only the finite* ProbeBundle *centers, membership constructors, coverage distances, and rational comparison rows supplied by total boundedness;*
4. *complete-limit readback, requiring C and L to expose regular Cauchy stream, modulus, limit, threshold, distance-witness, and rational-bound rows over the same metric source;*
5. *metric-completion handoff, requiring completion-facing consumers to read Q only after the finite-net and complete-limit rows have been displayed;*
6. *nonescape, requiring H, R, P, N to transport, replay, source, and name exactly this route and no open-cover, choice, quotient, host compactness, or ambient-completion datum.*

Proof. Project the carrier of Definition 2195.1. The visible inventory is $M, T, F, C, L, Q, H, R, P, N$, so every local obligation must be read from those rows.

The compact-source lock is forced first. The finite-net row and the complete-limit row both attach to the metric carrier named by M; otherwise the criterion would compare total boundedness and completeness over different sources.

The finite-net obligations are the rows T and F. They expose the tolerance, centers, membership constructors, coverage distances, and rational ledgers already provided by $\mathsf{TotallyBounded}^{\uparrow}$, and they do not create an open cover or a selected finite subcover.

The complete-limit obligations are the rows C and L. They expose the regular Cauchy name, modulus, displayed limit, threshold histories, metric-distance witnesses, and rational comparison ledgers already provided by $\mathsf{CompleteMetric}^{\uparrow}$.

Finally Q, H, R, P, N provide the completion-facing handoff, transport, replay, provenance, and naming for the same route. Since no other coordinate occurs in the carrier, the excluded open-cover, choice, quotient, host compactness, and ambient-completion principles cannot enter the local certificate. □

Theorem 2195.3 (Bishop compactness criterion net-completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCompactnessCriterion}^{\uparrow}$ *carrier factors through the ordered route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

The finite-net side is read through T, F*, the complete-limit side is read through* C, L*, and the terminal completion handoff is* Q*. No consumer reaches* $\mathsf{MetricCompletion}^{\uparrow}$ *from this packet before the finite-net and complete-limit rows have both been displayed over the same compact metric source.*

Proof. Begin with the compact metric source row M. It names the metric carrier that both the total-bounded and complete-metric faces must use.

The finite-net side is read next through T and F. These rows expose the finite ProbeBundle centers and coverage ledgers needed by the constructive compactness criterion.

The complete-limit side is read through C and L. These rows expose the regular Cauchy limit-readback data over the same metric source, including the modulus, displayed limit, late thresholds, and metric-distance witnesses.

Only after those two faces are visible does Q supply the metric-completion handoff. The transport, replay, provenance, and local naming rows preserve this order, so a completion-facing consumer receives no selected representative, quotient completion equality, open-cover compactness, countable-choice schedule, or host complete-space datum. □

Theorem 2195.4 (Bishop compactness criterion compact metric scope)**.** *The* $\mathsf{BishopCompactnessCriterion}^{\uparrow}$ *packet closes exactly the constructive compact-metric criterion surface obtained from a displayed compact metric carrier, total-bounded finite-net data, complete-metric limit-readback data,* hsame *transport,* Cont *replay,* Pkg *provenance, local* NameCert *rows, and the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff. It does not claim open-cover compactness, host sequential compactness, a selected convergent subsequence, countable choice, quotient completion equality, or completeness of an ambient metric space outside the displayed rows.*

Proof. Apply Definition 2195.2. The obligation surface lists the compact source, finite-net face, complete-limit face, completion handoff, and structural support rows as the entire local certificate.

Apply Definition 2195.3. It fixes the only completion-facing route and requires the finite-net and complete-limit rows to be displayed before the $\mathsf{MetricCompletion}^{\uparrow}$ handoff is consumed.

The remaining scope claim is boundary discipline. Open covers, host sequential compactness, selected subsequences, countable choice, quotient completion equality, and ambient complete spaces all require coordinates absent from K. The rows H, R, P, N preserve and name the displayed route rather than adding any of those data, so the compact metric scope is exactly the stated constructive criterion surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactnessCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCompactnessCriterion}^{\uparrow}$ packet is generated from a CompactMetricUp source row, TotallyBoundedUp finite-net data, CompleteMetricUp limit-readback data, a MetricCompletionUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompactnessCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 2195.1, Definition 2195.2, Definition 2195.3, and Definition 2195.4 seed the constructive compactness criterion over compact metric source rows, total-bounded finite nets, complete-metric limit readback, metric-completion handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompactnessCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host sequential compactness, selected subsequences, countable choice, quotient completion equality, ambient complete metric spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source lock, finite-net exactness, complete-limit readback, metric-completion handoff, and nonescape for BEDC.Derived.BishopCompactnessCriterionUp.

Chapter 2196

A Concrete Naming Certificate for $\mathsf{DiniContinuity}^{\uparrow}$

Following Dini's monotone convergence theorem and Bishop–Bridges constructive compactness, the $\mathsf{DiniContinuity}^{\uparrow}$ packet records the finite surface by which a monotone family of continuous maps on a compact metric source yields a uniform tolerance handoff. It consumes the compact source rows of chapter 129, the map-continuity boundary of chapter 130, the uniform-limit target of chapter 463, the pointwise-to-uniform budget discipline of chapter 369, and the rational tolerance ledger of chapter 37. The packet does not choose a host supremum, import classical open-cover compactness, or identify completed real functions outside the displayed NameCert rows.

Definition 2196.1 (Dini continuity carrier)**.** *A* $\mathsf{DiniContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, M, L, R, V, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It supplies the carried metric source, compact finite-cover interface, and cover-refinement ledger used by the tolerance request.*
2. *M is the monotone-family row. It records the displayed family indices, pointwise comparison order, and the assertion that later family rows move monotonically toward the limiting row on the carried source.*
3. *L is the pointwise convergence row. It records, for each carried probe, the finite observation route by which the family reaches the limiting value within the requested rational tolerance.*
4. *R is the positive rational tolerance budget, read through chapter 37; it stores the requested tolerance, the finite subdivisions used in the cover argument, and the budget comparisons needed by the handoff.*
5. *V is a finite* $\mathsf{ProbeBundle}$ *cover extracted from the compact source row. Each probe cell carries a local continuity witness, a monotone-family comparison witness, and a rational budget slice.*
6. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row. It records that the finite cover and tolerance budget have been assembled into one uniform read for the limiting map.*
7. *H records componentwise* hsame *transport for compact source data, monotone family data, pointwise convergence rows, rational budgets, cover probes, uniform-limit handoff, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}, \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{ProbeBundle} \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto \mathsf{DiniContinuity}$$

9. *P is the* Pkg *provenance over* $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $K, M, L, R, V, U, H, C, P, N$ and has no coordinate for host suprema, classical excluded-middle compactness, selected subcovers outside the finite ProbeBundle*, quotient real-function equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.DiniContinuityUp.DiniContinuityTasteGate single carrier alignment`

Theorem 2196.2 (Dini continuity NameCert obligation surface)**.** *For an accepted* $\mathsf{DiniContinuity}^{\uparrow}$ *carrier*

$$D = (K, M, L, R, V, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the compact metric source row, monotone family row, pointwise convergence row, rational tolerance budget, finite probe cover, uniform-limit handoff, transport, replay, provenance, and local naming row;*
2. *monotone-family visibility, requiring every family comparison used by the packet to be read from M before a pointwise convergence row from L can be applied;*
3. *compact finite-cover extraction, requiring the uniform request to be reduced through the* ProbeBundle *cover V supplied by the compact source row K;*
4. *uniform tolerance budgeting, requiring the positive rational request and every cover-cell budget slice to be read through R before the* $\mathsf{UniformModulus}^{\uparrow}$ *and* $\mathsf{UniformLimit}^{\uparrow}$ *handoff U is exported;*
5. *ledger non-escape, requiring H, C, P, N to preserve exactly the rows K, M, L, R, V, U and to export no host supremum, no external subcover selection, no quotient function equality, and no ambient completeness principle.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2196.1. The accepted packet exposes exactly the compact source, monotone family, pointwise convergence, rational budget, finite probe cover, uniform-limit handoff, transport, replay, provenance, and local naming rows.

Monotone-family visibility is forced by the row order. A pointwise convergence read from L can only be used after the relevant family comparison has been displayed by M, so the packet cannot replace monotone convergence by an unrecorded host limit argument.

Compactness enters only through K and V. The compact source row supplies the finite-cover interface, and the ProbeBundle row stores the actual cover cells that the uniform request inspects.

The tolerance ledger is R. Each local continuity or monotonicity estimate reads a rational slice before the uniform-limit handoff U is exported, so the final tolerance bound is assembled from finite rows.

Finally H, C, P, N transport, replay, source, and name only the listed coordinates. Since the excluded host principles have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$ obligations. □

Theorem 2196.3 (Dini continuity uniform-Cauchy handoff)**.** *For an accepted* $\mathsf{DiniContinuity}^{\uparrow}$ *carrier*

$$D = (K, M, L, R, V, U, H, C, P, N),$$

every uniform-Cauchy consumer of D must read compact finite-cover extraction through K, V, pointwise convergence through L, monotone-family visibility through M, rational tolerance budgeting through R, and only then the $\mathsf{UniformLimit}^{\uparrow}$ *handoff U. The exported route contains no host supremum, selected subcover, quotient function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier into its displayed rows. The compact source row K supplies the finite-cover interface, while V records the actual $\mathsf{ProbeBundle}$ cells inspected by the consumer.

The convergence and monotonicity reads are separate obligations. A pointwise convergence row from L may be used only for a displayed probe, and the monotone-family comparison supporting that read must already be visible through M.

The tolerance handoff then reads the positive rational budget R. Each cover cell receives its finite budget slice before the packet exports the $\mathsf{UniformLimit}^{\uparrow}$ row U, so the consumer receives a uniform-Cauchy route assembled from finite compact-cover and rational-budget data.

Finally H, C, P, N transport, replay, source, and name exactly the rows K, M, L, R, V, U. Since the carrier has no coordinate for host suprema, selected subcovers outside V, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the exported handoff cannot read any such principle. □

Theorem 2196.4 (Dini continuity compact finite-cover route)**.** *Every accepted* $\mathsf{DiniContinuity}^{\uparrow}$ *carrier routes a monotone continuous-family tolerance request through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *finite* $\mathsf{ProbeBundle}$ *cover rows,* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *before any* $\mathsf{Real}^{\uparrow}$*-facing uniform conclusion is read. In particular, the compact source and finite cover are displayed before the rational budget, the continuous-map rows expose pointwise monotone visibility, and the shared tolerance is then handed through the uniform-modulus and uniform-limit rows without reading a host supremum, selected minimum radius outside the packet, open-cover compactness, or ambient real completeness.*

Proof. Project the carrier rows of Definition 2196.1. The compact source row K and finite $\mathsf{ProbeBundle}$ cover row V are visible before the rational tolerance budget R is consumed.

Use the continuous-map rows carried by M and L for pointwise monotone visibility. These rows expose only displayed family comparisons and probe-wise convergence data, so they cannot substitute a host supremum or a detached pointwise limit.

Route the shared tolerance through R, then through the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{UniformLimit}^{\uparrow}$ handoff row U. The selected cover cells and their local radius data are already stored in V, so the minimum-radius and finite-cover steps remain finite packet data.

The structural rows H, C, P, N preserve exactly this compact-cover route. Therefore every Real-facing uniform conclusion stays downstream of the compact source, finite cover, rational budget, continuous-family rows, and uniform handoff, with no noncarried compactness or completeness principle available. □

Theorem 2196.5 (Dini continuity monotone cover budget)**.** *Let*

$$D = (K, M, L, R, V, U, H, C, P, N)$$

be an accepted $\mathsf{DiniContinuity}^{\uparrow}$ *carrier. Refining the finite* $\mathsf{ProbeBundle}$ *cover row V while preserving the compact source row K, the monotone continuous-family row M, the pointwise convergence rows L, and the positive rational tolerance budget R preserves the* $\mathsf{UniformModulus}^{\uparrow}$ *and* $\mathsf{UniformLimit}^{\uparrow}$

handoff U. The refined route cannot export a host supremum, selected minimum outside the packet, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 2196.1. The old and refined covers are both finite ProbeBundle rows over the same compact source row K, so each refined cell reads its source and local radius through the displayed cover ledger.

Compare the old and refined finite cover rows componentwise. The refinement does not change the monotone-family row M, the pointwise convergence rows L, or the positive rational budget R; it only replaces a cover cell by finitely many displayed subcells whose provenance remains inside V.

Replay the same rational tolerance budget through the monotone rows. The monotone comparison keeps the local estimates ordered, and the pointwise rows remain attached to the displayed probes of the refined finite cover.

Apply Definition 2196.4 to the refined cover. The $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{UniformLimit}^{\uparrow}$ handoff still reads only $K, M, L, R, V, U, H, C, P, N$, so no host supremum, selected external minimum, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness is introduced. □

Theorem 2196.6 (Dini continuity uniform-window tightening)**.** *For an accepted* $\mathsf{DiniContinuity}^{\uparrow}$ *carrier*

$$D = (K, M, L, R, V, U, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$*-facing uniform-window request must pass through the compact source row K, the continuous-map family data carried by M, the pointwise convergence row L, the monotone visibility row M, the rational budget R, the finite* ProbeBundle *cover V, the* $\mathsf{UniformModulus}^{\uparrow}$ *and* $\mathsf{UniformLimit}^{\uparrow}$ *handoff U, and then the* hsame, Cont, Pkg, *and local* NameCert *rows. The tightened window exports no host supremum, open-cover compactness, quotient function equality, selected minimum radius outside V, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier definition of Definition 2196.1. The accepted packet displays the compact source, continuous-family data, pointwise convergence, monotone visibility, rational budget, finite cover, handoff, and structural certificate rows.

The uniform-Cauchy handoff of Definition 2196.3 fixes the tolerance order. A consumer must read the rational budget and the finite cover before the uniform-limit row can be exported.

The compact finite-cover route of Definition 2196.4 fixes the compactness boundary. The cover cells come from the displayed ProbeBundle row rather than from an arbitrary open-cover principle or a selected global radius.

Finally the NameCert obligation surface of Definition 2196.2 repacks the route through H, C, P, N. Since the excluded Real-facing principles have no carrier coordinate, the tightened uniform window remains a finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiniContinuity}^{\uparrow}$ packet is generated from a compact metric source row, a monotone continuous-family row, pointwise convergence rows, a positive rational tolerance budget, a finite ProbeBundle cover, a UniformLimitUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 2196.1 and Definition 2196.2 seed the finite compact-metric monotone-continuity certificate surface for Dini-style uniform handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiniContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness beyond the displayed finite ProbeBundle, quotient real-function equality, host suprema, ambient completeness, selected subcovers outside the carrier, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-family visibility, compact finite-cover extraction, uniform tolerance budgeting, and ledger non-escape for BEDC.Derived.DiniContinuityUp.

Chapter 2197

A Concrete Naming Certificate for $\mathsf{GroebnerBasis}^{\uparrow}$

Following Buchberger's finite basis construction, the $\mathsf{GroebnerBasis}^{\uparrow}$ packet names the reduction certificate that turns a finite polynomial generator list into a displayed leading-term and S-polynomial completion surface. It consumes the normalized coefficient-list and operation rows of chapter 50 together with the membership and absorption rows of chapter 81. The chapter records a finite reduction trace, not a host polynomial-ring quotient, algebraic closure, selected term order outside the ledger, or global ideal-membership decision procedure.

Definition 2197.1 (Groebner basis carrier)**.** *A* $\mathsf{GroebnerBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, I, O, L, S, R, F, H, C, K, N)$$

with the following displayed rows:

1. *P is the polynomial source row. It records the ambient* $\mathsf{Poly}^{\uparrow}$ *coefficient ledger, the finite variable list, the displayed addition and multiplication rows, and the normal-form syntax used by every polynomial in the packet.*
2. *I is the ideal-generator row. It lists the carried generator polynomials and the* $\mathsf{Ideal}^{\uparrow}$ *predicate row that certifies their ambient ideal membership and closure under displayed polynomial multiplication.*
3. *O is the monomial-order row. It is a finite comparison ledger for the monomials inspected by this packet, including well-founded descent evidence for the listed reduction steps and no coordinate for an external global ordering oracle.*
4. *L is the leading-term ledger. For each displayed polynomial it records the leading monomial, leading coefficient, leading term, and divisibility comparison against the inspected reducers.*
5. *S is the S-polynomial row. It records each critical pair, common multiple row, multiplier row, subtraction row, and the source polynomials used to form the displayed S-polynomial.*
6. *R is the finite reduction trace. It lists the ordered reducer choices, multiplier rows, subtraction steps, strictly descending leading monomial witnesses, and terminal remainder for every inspected S-polynomial.*
7. *F is the normal-form row. It records exactly which inspected reductions terminate at zero and which terminal remainders remain visible as refusal rows for stronger ideal-membership claims.*

8. *H records componentwise* hsame *transport for polynomial sources, ideal generators, monomial-order comparisons, leading-term rows, S-polynomial rows, reduction traces, normal-form rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a requested reduction certificate through polynomial inspection, ideal-generator lookup, leading-term comparison, S-polynomial construction, finite reduction, and normal-form readback.*
10. *K is the* Pkg *provenance over* $\mathsf{GroebnerBasis}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host quotient ring, a hidden global monomial-order oracle, algebraic closure, a completed ideal-membership algorithm, selected normal forms outside F, or Lean-side verification.

Lean4 ✓ `BEDC.Derived.GroebnerBasisUp.GroebnerBasisTasteGate single carrier alignment`

Theorem 2197.2 (Groebner basis NameCert obligations)**.** *For every accepted* $\mathsf{GroebnerBasis}^{\uparrow}$ *carrier $G = (P, I, O, L, S, R, F, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the polynomial source, ideal-generator list, monomial-order ledger, leading-term ledger, S-polynomial row, finite reduction trace, normal-form row, transport, replay, provenance, and local naming;*
2. *polynomial-source compatibility, requiring every polynomial used by I, L, S, R, F to be read through the same* $\mathsf{Poly}^{\uparrow}$ *coefficient and operation rows;*
3. *ideal-source compatibility, requiring every generator in I and every reducer used by R to be carried by the displayed* $\mathsf{Ideal}^{\uparrow}$ *predicate surface;*
4. *monomial-order and leading-term exposure, requiring O and L to display every comparison and leading-term read used by the reduction trace;*
5. *S-polynomial exactness, requiring S to display the two source polynomials, common multiple, multiplier rows, and subtraction row for each critical pair;*
6. *reduction-trace exactness, requiring R to display each reducer choice, multiplier, subtraction, and strictly descending leading-monomial witness until the terminal row in F is reached;*
7. *transport, continuation, provenance, and naming, requiring H, C, K, N to replay only the displayed rows of the same packet.*

No Groebner-basis obligation is stored outside these rows.

Proof. Open the carrier of Definition 2197.1. The rows P and I give the only polynomial and ideal sources. Every later row must project back to those two sources before it can be read as part of the packet.

The monomial-order and leading-term obligations are finite because O and L list the comparisons actually used in the displayed reductions. The certificate never asks for an unbounded ordering oracle; it asks only for the inspected comparison rows and their descent witnesses.

For each critical pair, the row S displays the common multiple, the two multipliers, and the subtraction that forms the S-polynomial. The reduction row R then records the finite sequence of reducer choices and subtraction steps, and each step carries its strictly descending leading-monomial witness from O and L.

The terminal row F records whether the inspected S-polynomial reduces to zero or leaves a visible remainder. Zero terminal rows support the displayed basis obligation. Nonzero terminal rows remain refusal data, so the certificate does not promote the packet to a complete ideal-membership decision procedure.

Finally, componentwise transport in H, continuation replay in C, package provenance in K, and local naming in N preserve exactly the rows already listed. Since the carrier has no coordinate for host quotient rings, hidden ordering oracles, algebraic closure, or selected normal forms outside F, no additional NameCert obligation can be read from the packet. □

Definition 2197.3 (Groebner basis finite reduction carrier)**.** *A* $\mathsf{GroebnerBasis}^{\uparrow}$ *finite reduction carrier is a displayed refinement*

$$G_{\mathrm{red}} = (P, A, I, L, S, T, F, H, C, K, N)$$

of the carrier in Definition 2197.1. The row P is the finite $\mathsf{Poly}^{\uparrow}$ *row over the coefficient* $\mathsf{Ring}^{\uparrow}$ *and* $\mathsf{CommRing}^{\uparrow}$ *surfaces, A records the finite polynomial list inspected by the packet, I is the displayed* $\mathsf{Ideal}^{\uparrow}$ *generator source, L is the leading-term ledger for the inspected rows, S is the finite S-polynomial ledger, T is the reduction-trace ledger, and F is the terminal normal-form or refusal ledger. The row H transports P, A, I, L, S, T, F by componentwise* hsame*, C replays the same finite trace by* Cont*, K records* Pkg *provenance over* $\mathsf{GroebnerBasis}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{Ideal}^{\uparrow}$*,* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{CommRing}^{\uparrow}$*, and* $\mathsf{HilbertBasis}^{\uparrow}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$ *naming row.*

The classifier of G_{red} compares two accepted packets only by matching these rows and their displayed descent witnesses. It has no coordinate for quotient ideal equality, a selected infinite basis, a hidden term-order oracle, a completed ideal-membership decision procedure, or a normal form chosen outside F.

Theorem 2197.4 (Groebner basis finite reduction NameCert obligations)**.** *For every accepted finite reduction carrier*

$$G_{\mathrm{red}} = (P, A, I, L, S, T, F, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$ *surface consists exactly of finite polynomial-row admission through P and A, ideal-generator admission through I, leading-term exposure through L, S-polynomial construction through S, reduction-trace exactness through T, terminal normal-form or refusal exposure through F, componentwise transport through H, displayed replay through C, provenance through K, and local naming through N.*

Proof. Project the refinement to the carrier of Definition 2197.1. The rows P, A, I, L, S, T, F are the finite algebraic rows visible before transport or replay is applied, and every row is read through the existing $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, and $\mathsf{CommRing}^{\uparrow}$ surfaces.

Induct over the displayed reduction ledger T. The base row is the S-polynomial displayed in S with its leading-term data from L. In the step case, the reducer, multiplier, subtraction, and descent witness are all rows of T, so the next polynomial remains inside the finite polynomial ledger and no hidden basis element is introduced.

At the terminal row, F records either a zero normal form or a visible refusal remainder. The zero case exports the finite reduction certificate; the refusal case blocks any stronger ideal-membership or completed-basis claim.

The structural rows H, C, K, N then transport, replay, source, and name the same finite ledger. Since none of these rows contains quotient ideal equality, an infinite basis selector, or a normal form outside F, the listed rows exhaust the finite reduction obligations. □

Theorem 2197.5 (Groebner basis Buchberger handoff)**.** *Every Buchberger-style read of an accepted* $\mathsf{GroebnerBasis}^{\uparrow}$ *finite reduction carrier factors through the finite handoff*

$$\mathsf{Poly}^{\uparrow},\ \mathsf{Ring}^{\uparrow},\ \mathsf{CommRing}^{\uparrow},\ \mathsf{Ideal}^{\uparrow},\ L,\ S,\ T,\ F,\ \mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}.$$

The handoff supplies the reduction rows that a downstream $\mathsf{HilbertBasis}^{\uparrow}$ *or polynomial-normal-form consumer may inspect. It does not supply quotient ideal equality, a selected infinite basis, a global ideal-membership decision procedure, or a normal-form oracle outside the displayed terminal ledger.*

Proof. Start with the finite reduction carrier of Definition 2197.3. The polynomial, coefficient-ring, commutative-ring, and ideal rows give the only algebraic sources available to the handoff.

For each critical pair, the S-polynomial row S points to the leading-term data in L and to the ordered reduction ledger T. Induction over T shows that every subtraction step is justified by a displayed reducer and a displayed descent witness, so the handoff remains finite.

Finally read the terminal row F. A zero terminal row is the Buchberger reduction certificate exported to downstream consumers, while a nonzero terminal row is refusal data. The $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$ naming row, together with H, C, K, preserves that distinction and prevents the handoff from being read as a quotient equality or an infinite-basis selection. □

Theorem 2197.6 (Groebner basis reduction certificate)**.** *Let* $G = (P, I, O, L, S, R, F, H, C, K, N)$ *be an accepted* $\mathsf{GroebnerBasis}^{\uparrow}$ *carrier. For every S-polynomial row displayed in* S*, the reduction certificate exported to downstream polynomial-normal-form and ideal-membership consumers factors through the finite route*

$$P, I, O, L, S \longmapsto R \longmapsto F, H, C, K, N.$$

If the corresponding terminal row in F *is zero, the inspected critical-pair reduction is certified by displayed polynomial operations and ideal-generator membership rows. If the terminal row is nonzero, the same row is retained as refusal data for any claim that would require a completed Groebner basis beyond the displayed packet.*

Proof. Fix an inspected S-polynomial row in S. Its source polynomials, common multiple, multipliers, and subtraction row are already displayed over the polynomial source P and the ideal-generator source I.

Read the ordered reduction trace in R. Each step chooses a displayed reducer, checks divisibility against the current leading term using O and L, forms the multiplier row through P, subtracts the corresponding multiple, and records the descent witness needed for the next step.

Because R is finite, the route reaches the terminal row recorded in F. If that row is zero, the certificate exports exactly the displayed zero-reduction trace and the ideal-generator membership needed by downstream consumers. If it is not zero, the terminal polynomial remains visible as a refusal row, so a consumer cannot silently read a stronger ideal-membership or normal-form claim.

The structural rows H, C, K, N then transport, replay, source, and name the same reduction route. They add no hidden basis completion, quotient-ring equality, external term ordering, algebraic closure, or selected normal form outside the displayed trace. □

Falsifiable prediction: Every $\mathsf{GroebnerBasis}^{\uparrow}$ read factors through one $\mathsf{Poly}^{\uparrow}$ source row, one $\mathsf{Ideal}^{\uparrow}$ generator row, one finite monomial-order comparison ledger, one leading-term ledger, one S-polynomial table, one finite reduction trace, one normal-form or refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Independence witness: polynomial, ideal: $\mathsf{GroebnerBasis}^{\uparrow}$ is not bijective to either listed sibling because it jointly carries critical-pair formation, leading-term comparison, finite reduction traces, and terminal normal-form or refusal rows over the shared polynomial-and-ideal source.

Closure status for $\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2197.1, Definition 2197.3, Definition 2197.2, Definition 2197.4, Definition 2197.5, and Definition 2197.6 seed the finite Groebner reduction packet over displayed $\mathsf{Poly}^{\uparrow}$ and $\mathsf{Ideal}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroebnerBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global ideal-membership decision procedure, a quotient-ring normal-form theorem, algebraic closure, a host monomial-order oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial and ideal source compatibility, leading-term exposure, S-polynomial exactness, reduction-trace exactness, terminal-row discipline, and transport-replay coverage for BEDC.Derived.GroebnerBasisUp.

Constructive story: A $\mathsf{GroebnerBasis}^{\uparrow}$ packet is generated from a polynomial source row, an ideal-generator list, a monomial-order comparison ledger, a leading-term ledger, S-polynomial rows, finite reduction traces, terminal normal-form or refusal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2198

A Concrete Naming Certificate for $\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}$

Following Kelley-style nets and the standard Hausdorff uniqueness principle, the $\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}$ packet records the finite BEDC route by which two accepted limit reads of the same Cauchy net are identified only at a displayed separated zero-distance row. It sits downstream of the directed-window Cauchy-net surface of chapter 589, the zero-distance classifier of chapter 1411, the completion-facing handoff of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives completion consumers a uniqueness boundary without importing quotient nets, selected limit representatives, host equality, countable choice, or ambient completeness.

Definition 2198.1 (CauchyNetHausdorffUniqueness carrier)**.** *A* $\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, L_0, L_1, S, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window route, including the index preorder, late-window Cauchy rows, and finite metric-distance reads for the same net;*
2. *L_0 and L_1 are two displayed limit reads over D, written through the finite routes*

$$D \longmapsto L_0$$

and

$$D \longmapsto L_1;$$

3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *or Hausdorff-metric separation row that displays zero distance between the two limit reads after both are read from the same directed-window route;*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, recording that completion-facing consumers see only the Cauchy-net route, the two displayed limit reads, and the separated comparison row;*
5. *H records componentwise* hsame *transport for $D, L_0, L_1, S, M, C, P, N$;*
6. *C records displayed* Cont *replay through the directed Cauchy windows, the two limit reads, the separated comparison, and the metric-completion handoff;*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

8. N is the local $\mathsf{NameCert}_{\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}}$ *row.*
The carrier contains no quotient-net equality, no selected representative limit, no host equality of points, no countable-choice schedule, and no ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyNetHausdorffUniquenessUp.CauchyNetHausdorffUniquenessTasteGate`
`single carrier alignment`

Theorem 2198.2 (CauchyNetHausdorffUniqueness NameCert obligations)**.** *For the carrier of Definition 2198.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}}$ *surface consists exactly of carrier admission for $U = (D, L_0, L_1, S, M, H, C, P, N)$, the shared* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window route D, Hausdorff separation through S, endpoint uniqueness through the displayed zero-distance comparison,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

The obligations do not include quotienting nets, choosing a canonical limit, comparing host equality, importing countable choice, or asserting ambient completeness.

Proof. Carrier admission is projection from Definition 2198.1. The rows $D, L_0, L_1, S, M, H, C, P, N$ are the complete displayed packet.

The shared Cauchy-net route is the first obligation. Both limit reads are attached to the same directed-window row D, so a consumer cannot compare endpoints before reading the same Cauchy evidence.

The Hausdorff step is the row S. It is read only after L_0 and L_1 have both been displayed over D, and it identifies the endpoints through separated zero-distance evidence rather than through host equality.

The rows M, H, C, P, N preserve and expose that same route. The metric-completion handoff, transport, replay, provenance, and local naming rows add no coordinate for quotient nets, selected representatives, choice, or ambient completeness. □

Theorem 2198.3 (CauchyNetHausdorff limit separation)**.** *Let*

$$U = (D, L_0, L_1, S, M, H, C, P, N)$$

be an accepted $\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}$ *packet. If L_0 and L_1 are two accepted limit reads over the same* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window route D, then the packet identifies them only through the displayed separated zero-distance row S, transported by H and replayed by C into the same* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff M.*

Thus two accepted limit reads of the same directed-window Cauchy route are equal for the exported classifier exactly at the $\mathsf{SeparatedMetric}^{\uparrow}$ *row. The conclusion is not quotient-net equality, not host point equality, and not a selected-limit principle.*

Proof. Read the accepted packet through the carrier definition. The two limit rows L_0 and L_1 share the same source coordinate D, so every comparison begins with the displayed $\mathsf{CauchyNet}^{\uparrow}$ directed-window evidence.

The only comparison row is S. It records separated zero-distance evidence for the two displayed limit reads and is therefore the unique route by which the exported classifier identifies them.

Transport and replay do not change the comparison target. The rows H and C move and replay D, L_0, L_1, S componentwise, while M, P, N record the completion handoff, provenance, and local name.

Since no remaining coordinate contains quotient-net equality, host equality, a selected representative, choice, or ambient completeness, the uniqueness conclusion stays at the displayed separated row. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}$ packet is generated from one displayed CauchyNet directed-window route, two displayed limit reads over that route, a separated zero-distance comparison row, a MetricCompletion handoff row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2198.1, Definition 2198.2, and Definition 2198.3 seed the finite Cauchy-net Hausdorff uniqueness route over directed windows, displayed limit reads, separated zero-distance evidence, and MetricCompletion handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNetHausdorffUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient-net equality, selected representatives, host equality, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shared Cauchy-net route stability, separated zero-distance comparison, MetricCompletion handoff exactness, and ledger non-escape for BEDC.Derived.CauchyNetHausdorffUniquenessUp.

Chapter 2199

A Concrete Naming Certificate for $\mathsf{CantorCompleteMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorCompleteMetric}^{\uparrow}$ packet records the nested closed-ball criterion for reading a complete-metric point from finite BEDC data. It sits after the complete-metric surface of chapter 128, the completion route of chapter 1506, the Cauchy-filter interface of chapter 333, the nested-interval spine of chapter 320, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names a constructive closed-ball handoff; it does not import open-cover compactness, countable choice, quotient completions, selected limits, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2199.1 (CantorCompleteMetric carrier)**.** *A* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{ccm}} = (M, B, F, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence obligations are the only complete-metric data available to the packet.*
2. *B is the nested closed-ball row, listing a descending finite window of closed metric balls with each later ball displayed inside the earlier ball.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested ball spine and read only through the displayed ball inclusions.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, storing the requested tolerances and the visible shrink bounds for the ball radii.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the inspected nested-ball prefix.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the numerical radius and center schedule needed by the completion route.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, F, D, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for the metric source, nested closed balls, Cauchy filter, dyadic radius ledger, finite windows, regular readback, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from closed-ball inclusion through Cauchy-filter generation, dyadic radius shrinkage, finite windows, regular readback, and the terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorCompleteMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for an open cover, a countable-choice selector, a quotient completion, a selected limit, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CantorCompleteMetricUp.CantorCompleteMetricTasteGate single carrier alignment`

Theorem 2199.2 (CantorCompleteMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *carrier*

$$K_{\mathrm{ccm}} = (M, B, F, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorCompleteMetric}^{\uparrow}}$ *surface consists exactly of carrier admission, nested closed-ball inclusion, radius shrinkage, Cauchy-filter generation, finite-window exposure, regular-rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows displayed in Definition 2199.1. It has no obligation row for open-cover compactness, countable choice, quotient completions, selected limits, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier K_{ccm}. The row M supplies only the complete-metric source already named by chapter 128, while B supplies the nested closed-ball spine and F repacks that spine as the Cauchy-filter surface of chapter 333.

The radius data are read next. The dyadic ledger D records the visible shrink bounds, and the finite window W determines which closed balls are inspected at the requested tolerance.

The numerical readback and seal are downstream. The row Q exposes the regular-rational schedule used by chapter 301, and the row E seals that read as the $\mathsf{Real}^{\uparrow}$ output only after the nested-ball and filter rows have been displayed.

The structural rows H, C, P, N preserve the same route. Since no row contains an open cover, a countable-choice selector, a quotient completion, a selected limit, or full $\mathsf{Real}^{\uparrow}$ completeness data, none of those objects can become a local NameCert obligation. □

Theorem 2199.3 (CantorCompleteMetric nested closed-ball handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *carrier factors through the finite route*

$$M, B \longmapsto F, D, W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus a later $\mathsf{CompactMetric}^{\uparrow}$ *or Real-and-Completion consumer receives the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, descending closed-ball window,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff, dyadic radius ledger, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *rows. The route exports no open-cover compactness, countable choice, quotient completion, selected limit, or full* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a completion-facing consumer request. The request can inspect the packet only through the accepted carrier, so the first visible data are the complete-metric row M and the nested closed-ball spine B.

The handoff then passes through the generated filter and radius ledger. The row F records the Cauchy-filter surface coming from the nested balls, while D and W bind each tolerance request to a displayed finite window.

The regular-rational readback follows the same finite route. The row Q is read after the ball, filter, radius, and window rows, and E seals only that displayed read as the Real-facing output.

Finally H, C, P, N transport, replay, source, and name the route. No step opens an open cover, chooses countably many centers, quotients a completion, selects a host limit, or invokes full $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2199.1, Definition 2199.2, and Definition 2199.3 seed the finite nested closed-ball criterion for complete-metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested closed-ball inclusion, radius shrinkage, Cauchy-filter generation, finite-window exposure, regular readback, Real sealing, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.CantorCompleteMetricUp.

Constructive story: A $\mathsf{CantorCompleteMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, descending closed-ball window, CauchyFilter handoff, DyadicRatCore radius ledger, StreamName window, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2200

A Concrete Naming Certificate for $\mathsf{UniformLimitTheorem}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformLimitTheorem}^{\uparrow}$ packet records the finite NameCert route by which a uniformly convergent sequence of continuous maps exports continuity of its limit. It consumes the continuous-map rows of chapter 130, the uniform-limit ledger of chapter 463, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seals of chapter 38, and the metric source and target rows of chapter 57. The packet names the theorem handoff as finite certificate data; it is not an ambient function-space topology, a quotient stream equality, a selected host limit, or a global completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 2200.1 (Uniform-limit theorem carrier)**.** *A* $\mathsf{UniformLimitTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, L, W, R, E, X, Y, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence ledger. Each entry records a continuous map from the displayed metric source X to the displayed metric target Y.*
2. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *modulus ledger binding target tolerances to a finite tail stage and to the shared source-radius information used by the limit read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family exposing exactly the finite values of the sequence and of the limit candidate needed for the displayed tolerance request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W after the uniform-limit budget in L has fixed the tail.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows and regular-rational readbacks have been displayed.*
6. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the local source-radius requests consumed by the continuity rows in F.*
7. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including the target tolerance ledger consumed by L, R, E.*
8. *H records componentwise* hsame *transport for $F, L, W, R, E, X, Y, C, P, N$.*
9. *C records displayed* Cont *replay from a limit-continuity request through X, Y, F, L, W, R, E and then through the structural rows.*

10. *P is the* Pkg *provenance over* UniformLimitTheorem$^\uparrow$, ContinuousMap$^\uparrow$, UniformLimit$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, Met$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformLimitTheorem}^\uparrow}$ *row.*

The carrier has no coordinate for an ambient topology on maps, a host-level selected limit function, quotient stream equality, countable choice, or completeness of the target metric space beyond the displayed rows.

Lean4 ✓ `BEDC.Derived.UniformLimitTheoremUp.UniformLimitTheoremTasteGate single carrier alignment`

Theorem 2200.2 (Uniform-limit theorem NameCert obligations)**.** *For an accepted* UniformLimitTheorem$^\uparrow$ *carrier $U = (F, L, W, R, E, X, Y, H, C, P, N)$, the local* NameCert$_{\text{UniformLimitTheorem}^\uparrow}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed tuple over the* ContinuousMap$^\uparrow$ *sequence,* UniformLimit$^\uparrow$ *ledger, finite* StreamName$^\uparrow$ *windows,* RegSeqRat$^\uparrow$ *readbacks,* Real$^\uparrow$ *seals, and source and target* Met$^\uparrow$ *rows;*
2. *uniform-limit visibility, requiring every limit-continuity read to pass through the modulus ledger L before any window or real-seal row is exposed;*
3. *continuity inheritance, requiring the source-radius row read from a member of F and the tail bound read from L to be jointly visible before the limit row is named continuous;*
4. *finite-window discipline, requiring every sequence value and limit candidate used by the handoff to be read through W and then through R;*
5. Real$^\uparrow$*-seal nonescape, requiring the terminal real seal E to appear only after L, W, R and the target metric row Y have been displayed;*
6. *transport and replay stability through the componentwise* hsame *row H and displayed* Cont *row C;*
7. *provenance exhaustion through P and the local naming row N.*

Proof. Carrier admission is projection from Definition 2200.1. It exposes the continuous-map sequence, the uniform-limit ledger, finite windows, regular readbacks, real seals, metric rows, transport, replay, provenance, and local naming rows.

Uniform-limit visibility is the role of L. A target tolerance first selects a displayed tail stage and the source-radius information needed for the limit read. Only then may the route inspect the finite windows W and regular readbacks R.

Continuity inheritance uses F together with L. The chosen member of the continuous sequence supplies the local source-radius row, while the uniform-limit ledger supplies the tail comparison that transfers the target tolerance to the limit candidate.

The Real-facing end of the route is W, R, E. The windows expose only finite observations, the readback row repacks those observations as regular rational data, and the Real$^\uparrow$ seal appears after these rows and the target metric row have been displayed.

Finally H, C, P, N transport, replay, package, and name the same route. Since the carrier lacks ambient map topology, quotient stream equality, selected host limits, choice, and target-completeness coordinates, no further obligation can be read from the local certificate. □

Theorem 2200.3 (Uniform limit preserves continuity as a BEDC handoff)**.** *Every continuity read of the limit named by an accepted* UniformLimitTheorem$^\uparrow$ *carrier factors through the displayed route*

$$X, Y \longmapsto F, L \longmapsto W, R \longmapsto E, H, C, P, N.$$

Thus the source and target $\mathsf{Met}^{\uparrow}$ *rows are read first, the* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence and* $\mathsf{UniformLimit}^{\uparrow}$ *ledger supply the continuity and tail witnesses, the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows expose only finite readbacks, and the* $\mathsf{Real}^{\uparrow}$ *seal is reached only at the end of the route. No consumer of the packet receives an ambient function-space topology, a selected host limit, quotient stream equality, countable choice, or target completeness not present in the carrier.*

Proof. Begin with a continuity request for the limit. The only source and target data available are the metric rows X and Y, so the request first reads the source-radius and target-tolerance surfaces recorded there.

Next the route reads F and L. The sequence row F provides the continuous-map witness at the displayed tail stage, while L provides the uniform-limit budget that lets the same target tolerance be compared with the limit candidate.

The finite observation stage is W, R. The selected sequence value and the limit candidate are exposed through StreamName windows and then through RegSeqRat readbacks; no quotient stream or host function value is available.

The terminal stage is E, H, C, P, N. The Real seal is reached only after the finite readbacks have been fixed, and the structural rows transport, replay, package, and name the same finite route. This proves that continuity of the limit is a BEDC handoff through the displayed uniform-limit ledger rather than an appeal to an ambient theorem outside the packet. □

Theorem 2200.4 (Uniform-limit regular-Cauchy modulus handoff)**.** *Let* $U = (F, L, W, R, E, X, Y, H, C, P, N)$ *be an accepted* $\mathsf{UniformLimitTheorem}^{\uparrow}$ *carrier. Every limit-continuity modulus read of* U *passes from the source and target* $\mathsf{Met}^{\uparrow}$ *rows* X, Y *through the* $\mathsf{ContinuousMap}^{\uparrow}$ *row* F*, the* $\mathsf{UniformLimit}^{\uparrow}$ *ledger* L*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

The route L, W, R, E *is the only modulus-to-real route carried by the packet. It exports no ambient function-space theorem, quotient stream equality, selected host limit, countable-choice schedule, or target-completeness principle beyond the displayed metric, window, readback, and real-seal rows.*

Proof. Project the carrier of Definition 2200.1. The only metric data are X, Y, the only continuity data are stored in F, and the only uniform convergence modulus is the finite ledger L.

Apply Definition 2200.2. Its visibility and finite-window clauses force a limit-continuity request to read the uniform-limit budget before inspecting W, then to recover regular rational data through R, and only then to consume the real seal E.

Finally use Definition 2200.3. It identifies the same route as the BEDC continuity handoff, so L, W, R, E exhaust the modulus-to-real path for RealUp-sealed uniform-limit arguments. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformLimitTheorem}^{\uparrow}$ packet is generated from a ContinuousMapUp sequence ledger, a UniformLimitUp modulus ledger, finite StreamNameUp windows, RegSeqRatUp readbacks, RealUp seals, source and target MetricUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformLimitTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 2200.1, Definition 2200.2, and Definition 2200.3 seed the finite theorem handoff from ContinuousMapUp and UniformLimitUp rows through StreamNameUp, RegSeqRatUp, RealUp, and MetricUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformLimitTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ambient function-space topology, quotient stream equality, selected host limits, countable choice, or target completeness beyond the displayed rows.

Upgrade path: Obligation closure requires checked carrier admission, uniform-limit visibility, continuity inheritance, finite-window discipline, Real-seal nonescape, transport and replay stability, and provenance exhaustion for BEDC.Derived.UniformLimitTheorem

Chapter 2201

A Concrete Naming Certificate for $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ packet isolates the explicit rational and dyadic bound surface used when a displayed $\mathsf{Real}^{\uparrow}$ name is read by order-facing or completion-facing consumers. It sits between the real-name seal of chapter 38, the regular Cauchy handoff of chapter 301, the stream window of chapter 126, the rational carrier of chapter 37, the dyadic presentation of chapter 291, and the Archimedean bound packet of chapter 430. The schema records the finite ledger through which a bound is inspected; it does not assert a supremum, trichotomy, quotient equality of real names, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2201.1 (Real Archimedean bound schema carrier)**.** *A* $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, Q, D, S, G, B, H, C, P, N)$$

with the following visible rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name whose bound route is being displayed.*
2. *Q is the* $\mathsf{Rat}^{\uparrow}$ *row naming the rational bound used by order and completion consumers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *presentation of the same bound at the dyadic inspection surface.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window through which finite reads of R are exposed.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff connecting the stream window to regular rational approximants.*
6. *B is the finite bound ledger comparing the inspected rational approximants with the rational and dyadic bound rows.*
7. *H stores componentwise* hsame *transports for the real name, rational row, dyadic row, stream window, regular-sequence handoff, bound ledger, route rows, provenance, and local naming.*
8. *C stores the displayed* Cont *route from a consumer request through $R, Q, D, S, G, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{ArchimedeanReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanBoundSchema}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, Q, D, S, G, B, H, C, P, N$. No host supremum, selected least bound, trichotomy oracle, quotient equality of real names, or ambient completeness datum is a coordinate of the carrier.

Lean4 ✓ `BEDC.Derived.RealArchimedeanBoundSchemaUp.RealArchimedeanBoundSchemaTasteGate` single carrier alignment

Theorem 2201.2 (Real Archimedean bound schema obligations)**.** *For every accepted* RealArchimedeanBoundSchem *carrier $A = (R, Q, D, S, G, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanBoundSchema}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real name, rational bound row, dyadic presentation, stream window, regular rational handoff, finite bound ledger, transport rows, continuation routes, provenance, and local naming;*
2. *rational-dyadic coherence, requiring Q and D to be read as one bound witness for the same displayed real name;*
3. *window/readback compatibility, requiring every finite inspection of R to pass through S and then through the regular rational handoff G before the ledger B is consulted;*
4. *ledger non-escape, excluding host suprema, selected least bounds, trichotomy, quotient equality, and ambient completeness from the local certificate;*
5. *consumer handoff, requiring each* $\mathsf{Real}^{\uparrow}$ *order-facing, locatedness-facing, or completion-facing read to receive $Q, D, S, G, B, H, C, P, N$ before using the displayed bound.*

No Real Archimedean bound schema obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 2201.1. The bound rows are Q and D, the real-name inspection rows are S and G, and the finite comparison surface is B.

Rational-dyadic coherence is componentwise: both bound rows are coordinates of the same packet and are transported by the same H row. The continuation row C gives a single route from a consumer request to those two rows, so the rational and dyadic presentations cannot be separated by the classifier.

Window/readback compatibility follows from the route shape. A finite read of R first opens the stream window S, then the regular rational handoff G, and only then reaches the bound ledger B.

The remaining clauses are the non-escape and handoff boundary. Since the carrier has no coordinate for a supremum, selected least bound, trichotomy oracle, quotient equality, or completeness theorem, the only public bound route is the displayed $Q, D, S, G, B, H, C, P, N$ ledger. □

Theorem 2201.3 (Real Archimedean rational-dyadic handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *order-facing, locatedness-facing, or completion-facing consumer of an accepted* $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ *carrier reads the rational row Q and the dyadic row D through the same finite route*

$$R \longmapsto S, G \longmapsto B \longmapsto Q, D, H, C, P, N.$$

Thus the consumer receives one rational-dyadic bound surface, with the stream window, regular rational readback, bound ledger, transport, continuation, provenance, and local naming rows already displayed. The packet exports no supremum, trichotomy, quotient equality of real names, selected least bound, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer of the accepted carrier A. By Definition 2201.1, the consumer can inspect the real name R only through the window S and the regular rational handoff G.

The bound ledger B is then the only comparison row connecting those finite reads to the rational row Q and the dyadic row D. The structural rows H, C, P, N transport, replay, record provenance for, and name exactly that route.

Apply Definition 2201.2. Its coherence, compatibility, non-escape, and handoff clauses force order-facing, locatedness-facing, and completion-facing consumers to use the same S, G, B, H, C, P, N route for both Q and D. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanBoundSchema}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArchimedeanBoundSchema}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2201.1, Definition 2201.2, and Definition 2201.3 seed the explicit rational and dyadic bound schema between displayed $\mathsf{Real}^{\uparrow}$ names, $\mathsf{Rat}^{\uparrow}$ rows, $\mathsf{DyadicRatCore}^{\uparrow}$ presentations, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoffs, finite bound ledgers, structural transport, route replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArchimedeanBoundSchema}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close suprema, trichotomy, quotient equality of real names, selected least bounds, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-dyadic coherence, window/readback compatibility, ledger non-escape, and consumer handoff for BEDC.Derived.RealArchimedeanBoundSchemaUp.

Constructive story: A $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ packet is generated from a RealUp name, RatUp and DyadicRatCoreUp bound rows, a StreamNameUp inspection window, a RegSeqRatUp readback handoff, a finite bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2202

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ packet realizes the finite extractor route named in chapter 9237: compact finite-net rows and continuous-map pointwise-modulus rows are folded over a finite ProbeBundle of rational radii to produce a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff. It consumes the compact metric surface of chapter 129, the continuous-map pointwise rows of chapter 130, the rational lower-bound machinery of chapter 619, the uniform-modulus consumer row of chapter 369, the rational carrier of chapter 37, and the ProbeBundle finite-net selectors of chapter 914. The packet records the finite fold itself; it does not import open-cover compactness, sequential compactness, a choice-selected Lebesgue number, or an ambient completeness theorem.

Definition 2202.1 (Compact uniform modulus extractor carrier)**.** *A* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, T, P, L, M, U, H, C, G, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the finite* ProbeBundle *of centers at the requested rational tolerance.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus row over the same compact metric source.*
3. *T is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row requested by the downstream uniform-modulus consumer.*
4. *P is the finite* ProbeBundle *readback row listing the centers and coverage radii supplied by K.*
5. *L is the center-indexed lower-bound ledger whose entries are rational radii certified by the pointwise-modulus rows of F.*
6. *M is the finite rational-fold row that takes the lower bound of the displayed L entries without a host minimum selector.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from T, P, L, M and the source and target metric triangle ledgers.*
8. *H stores componentwise* hsame *transport for the compact source, continuous map, tolerance row, ProbeBundle readback, lower-bound ledger, rational fold, uniform-modulus handoff, replay, provenance, and local naming.*

9. *C stores displayed* Cont *replay from a uniform-modulus request through $K, F, T, P, L, M, U, H, G, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, F, T, P, L, M, U, H, C, G, N$. No open cover, selected subcover, sequential compactness argument, host lower-bound function, choice-selected Lebesgue number, or ambient completeness coordinate is part of the extractor.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusExtractorUp.CompactUniformModulusExtractorTasteGate`
`single carrier alignment`

Theorem 2202.2 (Compact uniform modulus extractor obligations)**.** *For every accepted* CompactUniformModulusE *carrier $E = (K, F, T, P, L, M, U, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtractor}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric finite-net row, continuous-map pointwise-modulus row, rational tolerance row, ProbeBundle readback, lower-bound ledger, rational fold, uniform-modulus handoff, transport, replay, provenance, and local naming;*
2. *compact-source lock, requiring K and the source row of F to name the same metric carrier before the center bundle P is read;*
3. *pointwise-radius binding, requiring every entry of L to be attached to a center of P and to a displayed pointwise-modulus row of F;*
4. *rational-fold exactness, requiring M to be the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over L rather than a host minimum selector;*
5. *uniform-modulus handoff, requiring U to be read only after T, P, L, M and the metric triangle rows have been displayed;*
6. *structural transport, requiring* hsame, Cont, Pkg, *and local* NameCert *rows to replay the same finite extractor route;*
7. *ledger non-escape, excluding open-cover compactness, selected Lebesgue numbers, sequential compactness, choice, and ambient completeness from the local packet.*

No compact uniform modulus extractor obligation is stored outside these rows.

Proof. Project the carrier of Definition 2202.1. The compact source is K, the pointwise-modulus source is F, the requested tolerance is T, and the finite center bundle is P.

The source lock is required before the fold begins. The center rows in P and the pointwise-modulus rows in F are read over the same compact metric carrier, so the lower-bound ledger L cannot mix centers from one source with moduli from another.

The finite extractor step is the transition from L to M. Since M is a rational fold over the displayed finite ledger, it is classifier-stable data in the packet rather than a host-selected minimum.

Finally U is read only after the fold and the triangle rows are visible. The transport, replay, provenance, and local naming rows follow this same route, so open-cover compactness, selected Lebesgue numbers, sequential compactness, choice, and ambient completeness have no coordinate from which to enter the certificate. □

Theorem 2202.3 (Compact uniform modulus finite-fold handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer reading an accepted* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ *carrier factors through the finite route*

$$K, F, T \longmapsto P, L, M \longmapsto U, H, C, G, N.$$

Thus the consumer receives compact finite-net rows, continuous-map pointwise-modulus rows, the $\mathsf{Rat}^{\uparrow}$ *lower-bound fold over the displayed* ProbeBundle *radii, and the structural handoff rows before any tolerance-only modulus is public. The packet exports no open-cover compactness, selected subcover, selected Lebesgue number, host minimum selector, sequential compactness argument, choice principle, or ambient completeness theorem.*

Proof. Begin with a uniform-modulus consumer of E. The carrier of Definition 2202.1 exposes the compact metric source K, continuous-map row F, and rational tolerance T as the only input rows.

The finite center data are then read through P. Each center contributes a pointwise-modulus radius to L, and the fold row M computes the displayed rational lower bound over that finite ledger.

The handoff row U packages the folded radius as a tolerance-only uniform modulus, while H, C, G, N transport, replay, record provenance for, and name the same route. Applying Definition 2202.2, no consumer can replace this finite path by an open-cover proof, selected subcover, selected Lebesgue number, host minimum selector, sequential compactness argument, choice principle, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtractor}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2202.1, Definition 2202.2, and Definition 2202.3 seed the finite ProbeBundle rational-fold extractor from compact metric finite nets and continuous-map pointwise moduli to UniformModulusUp handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusExtractor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected Lebesgue numbers, sequential compactness, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source lock, pointwise-radius binding, rational-fold exactness, uniform-modulus handoff, structural transport, and ledger non-escape for BEDC.Derived.CompactUniformModulusExtr

Constructive story: A $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, a RatUp tolerance, a finite ProbeBundle center ledger, a rational lower-bound fold, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2203

A Concrete Naming Certificate for $\mathsf{DyadicIntermediateValue}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicIntermediateValue}^{\uparrow}$ packet records the finite dyadic bisection surface that sits between a displayed intermediate-value sign route and the later real-name seal. It is a neighbour of the $\mathsf{IntermediateValue}^{\uparrow}$ route of chapter 1402: the packet consumes the finite branch discipline of chapter 361, the endpoint carrier of chapter 394, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names the finite bisection certificate consumed by Real and Completion phases; it does not select a root, quotient real names, decide real trichotomy, invoke open-cover compactness, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2203.1 (Dyadic intermediate-value carrier)**.** *A* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, S, B, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint packet, including lower and upper endpoint windows, rational cells, and dyadic refinements visible to the bisection route.*
2. *S is the displayed sign-change ledger inherited from an* $\mathsf{IntermediateValue}^{\uparrow}$ *sign route; it records only the inspected endpoint signs and the midpoint signs used by the finite branch prefix.*
3. *B is the* $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger, retaining the finite sequence of dyadic child intervals whose displayed signs preserve the intermediate-value boundary.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the dyadic radius, midpoint arithmetic, and width comparison used to stop the branch prefix at the requested precision.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family for the retained endpoint names and midpoint observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W after the dyadic tolerance row Q has fixed the finite window.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row available only after I, S, B, Q, W, R have been displayed.*

8. *H records componentwise* hsame *transport for the interval, sign ledger, bisection ledger, dyadic tolerance, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$I, S \longmapsto B, Q \longmapsto W, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntermediateValue}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $I, S, B, Q, W, R, E, H, C, P, N$ and has no coordinate for a selected root, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicIntermediateValueUp.DyadicIntermediateValueTasteGate single carrier alignment`

Theorem 2203.2 (Dyadic intermediate-value NameCert obligations)**.** *For an accepted* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *carrier*

$$D = (I, S, B, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicIntermediateValue}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed located interval, sign ledger, bisection ledger, dyadic tolerance row, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *sign-ledger visibility, requiring every endpoint or midpoint sign consumed by the packet to pass through I and S before a branch row in B is read;*

3. *dyadic branch exactness, requiring every retained child interval in B to expose its midpoint arithmetic and width comparison through Q;*

4. *finite-window discipline, requiring every retained endpoint read to pass through W and then through R before the Real-facing row E is exposed;*

5. *transport and replay stability through H and C;*

6. *provenance exhaustion through P and the local naming row N.*

No $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2203.1. The projection exposes exactly the located interval, sign ledger, dyadic branch ledger, tolerance row, finite windows, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Sign-ledger visibility reads I and S before B. Endpoint and midpoint sign rows are therefore attached to the displayed located interval cells before any retained child interval is named.

Dyadic branch exactness reads B together with Q. Each branch step carries a midpoint and a width comparison through the dyadic arithmetic row, so the finite prefix cannot be replaced by an unrecorded host search.

The finite-window and Real-facing stages are W, R, E. The retained endpoints are first observed through StreamName windows, then repacked as regular rational data, and only then offered to the Real seal row.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for selected roots, quotient real equality, trichotomy, open-cover compactness, or ambient completeness, those principles cannot appear as local obligations. □

Theorem 2203.3 (Dyadic intermediate-value bisection route)**.** *Every branch read exported by an accepted* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *packet factors through*

$$\mathsf{IntermediateValue}^{\uparrow} \longmapsto I, S \longmapsto B, Q \longmapsto W, R.$$

Thus the packet consumes the intermediate-value sign route only as displayed finite sign data, refines it through a $\mathsf{DyadicBisection}^{\uparrow}$ *branch ledger and a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and exposes retained windows only through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*.*

Proof. Begin with an exported branch read. The only sign data visible to the packet are the rows I and S, so the read first passes through the located interval endpoint carrier and the displayed sign-change ledger inherited from the neighbouring intermediate-value route.

Next the retained child interval is read from B. The dyadic tolerance row Q supplies the midpoint arithmetic and width comparison attached to the branch prefix, so the branch read remains a finite dyadic bisection route.

Finally the retained endpoints are exposed through W and R. The StreamName windows record the finite observations, and the RegSeqRat row repacks them as regular rational readback. No selected root, quotient real, trichotomy decision, compactness argument, or ambient completeness theorem is used by the branch route. □

Theorem 2203.4 (DyadicIntermediateValue finite branch determinacy)**.** *Let*

$$D = (I, S, B, Q, W, R, E, H, C, P, N)$$

be an accepted $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *carrier. Every accepted branch row in B is determined by the displayed located interval row I, sign ledger S, dyadic tolerance row Q,* $\mathsf{StreamName}^{\uparrow}$ *windows W, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route R before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The rows H, C, P, N transport, replay, source, and name this same finite determination.*

Consequently the packet exports no selected root, quotient stream, quotient real, trichotomy decision, unrecorded host search, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier $D = (I, S, B, Q, W, R, E, H, C, P, N)$. The branch ledger B is read only after the located interval row I and sign ledger S have displayed the endpoint and midpoint sign data.

Apply Definition 2203.3. It identifies the branch read as a finite route through $\mathsf{IntermediateValue}^{\uparrow}$, I, S, then B, Q, then W, R. Thus each accepted branch row is fixed by the displayed sign data, dyadic tolerance row, StreamName windows, and RegSeqRat readback before the Real-facing seal appears.

Finally apply Definition 2203.5. The handoff to E is downstream from I, S, B, Q, W, R and is preserved by H, C, P, N, so no hidden search or root-selection coordinate can be inserted after the branch row. □

Theorem 2203.5 (Dyadic intermediate-value Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *carrier receives the finite route*

$$I, S, B, Q, W, R, E, H, C, P, N.$$

The handoff reads the located endpoint packet, sign ledger, dyadic bisection prefix, dyadic tolerance row, finite StreamName windows, and RegSeqRat readback before the Real seal. It exports a finite bisection certificate, not a selected root, quotient real construction, trichotomy principle, open-cover compactness argument, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier $D = (I, S, B, Q, W, R, E, H, C, P, N)$. A Real-facing consumer can reach the seal only through the displayed continuation row C, whose route lists the interval, sign, branch, tolerance, window, and readback rows before E.

The sign and branch rows are finite. The rows I, S, B, Q bind the endpoint signs, midpoint signs, retained child intervals, and dyadic width comparisons before any readback row is visible.

The readback stage is W followed by R. StreamName exposes only the finite retained endpoint observations, and RegSeqRat turns those observations into the regular rational data consumed by the Real boundary.

The structural rows H, C, P, N preserve and name the same route. Therefore the Real seal handoff is exactly the displayed finite bisection certificate, with no available coordinate for selected roots, quotient reals, trichotomy, open-cover compactness, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntermediateValue}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicIntermediateValue}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint packet, an $\mathsf{IntermediateValue}^{\uparrow}$ sign ledger, a $\mathsf{DyadicBisection}^{\uparrow}$ branch prefix, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2203.1, Definition 2203.2, Definition 2203.3, Definition 2203.4, and Definition 2203.5 seed the finite dyadic bisection certificate between the intermediate-value sign route and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected roots, quotient reals, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-ledger visibility, dyadic branch exactness, finite-window discipline, Real-seal handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.DyadicIntermediateValu

Chapter 2204

A Concrete Naming Certificate for $\mathsf{Egorov}^{\uparrow}$

Following Egorov and Bishop–Bridges constructive measure theory, the $\mathsf{Egorov}^{\uparrow}$ packet records almost-everywhere convergence as finite exceptional-set and finite-window uniformity data rather than as a host measure-theoretic selector. It sits after the measurable carrier discipline of chapter 92, the probability-space ledger of chapter 184, the random-variable preimage surface of chapter 185, the stream observation windows of chapter 126, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The carrier keeps every convergence claim inside a displayed finite budget and exports no selected exceptional set, ambient almost-everywhere equality predicate, or completeness principle.

The exceptional-window row is a measure-window sibling of the finite cover ledger in chapter 2166 and forms a later consumer boundary for maximal-function domination routes such as chapter 3581. This dependency is only finite exceptional-window and ledger routing; it is not measure completion, selected null-set construction, or host almost-everywhere equality.

Definition 2204.1 (Egorov carrier)**.** *An* $\mathsf{Egorov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, \Omega, F, X, S, R, A, W, U, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Measure}^{\uparrow}$ *row whose measurable-set ledger supplies the finite exceptional-set budget.*
2. *Ω is the* $\mathsf{ProbSpace}^{\uparrow}$ *normalization row used to read the budget as a probability-bound request.*
3. *F is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *rows whose pointwise convergence boundary is being inspected.*
4. *X is the limiting* $\mathsf{RandomVar}^{\uparrow}$ *row, kept as a named random-variable packet rather than as a host almost-everywhere equivalence class.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the family F and the limit row X.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route that turns each observed real value into a regular-rational comparison surface.*
7. *A is the finite exceptional-window row. It records the displayed measurable approximants, their finite measure budget, and the containment checks used by later uniformity reads.*

8. *W is the finite convergence window of indices, precision requests, and admissible observation sites outside A.*
9. *U is the uniformity row asserting that all reads in W meet the displayed regular-rational tolerance route outside the recorded exceptional window.*
10. *L is the measure ledger connecting A to M and Ω without selecting a countable exceptional set.*
11. *H records componentwise* hsame *transport for the measure, probability, random-variable, stream, regular-rational, exceptional-window, uniformity, ledger, replay, provenance, and local-name rows.*
12. *C records displayed* Cont *replay of the finite route*

$$M, \Omega, F, X, S, R, A, W, U, L \longmapsto H, P, N.$$

13. *P is the* Pkg *provenance over* $\mathsf{Egorov}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, \Omega, F, X, S, R, A, W, U, L, H, C, P, N$. It has no coordinate for full measure convergence theorems, countable choice, selected exceptional sets, host almost-everywhere equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EgorovUp.EgorovTasteGate single carrier alignment`

Theorem 2204.2 (Egorov NameCert obligations)**.** *For every accepted* $\mathsf{Egorov}^{\uparrow}$ *carrier*

$$E = (M, \Omega, F, X, S, R, A, W, U, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, probability row, finite random-variable family, limit random-variable row, finite stream observation schedule, regular-rational readback route, exceptional-window row, finite convergence window, uniformity row, measure ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public convergence read must pass through A, W, U, L before it can be exported as a local name.*

Proof. Project the accepted carrier in Definition 2204.1. The only convergence-facing rows are the exceptional-window row A, the finite convergence window W, the uniformity row U, and the measure ledger L.

The rows M, Ω, F, X, S, R supply the measurable, probabilistic, random-variable, stream, and regular-rational dependency surface on which those four rows are interpreted. The rows H, C, P, N transport, replay, source, and name exactly the same finite route.

Because no row stores a selected countable exceptional set, a host almost-everywhere equality predicate, a full measure convergence theorem, or an ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.EgorovUp.EgorovCarrier namecert obligations`

Theorem 2204.3 (Egorov finite-window uniformity)**.** *For every accepted* $\mathsf{Egorov}^{\uparrow}$ *carrier E, any public uniform-convergence read of E is determined by the displayed finite convergence window W together with the regular-rational readback route R and the uniformity row U. Refining the observation schedule through the same R, W, U leaves the public verdict unchanged outside the exceptional window A.*

Proof. The finite convergence window W lists the only indices, precision requests, and observation sites that the classifier may inspect. Each such request is evaluated through the regular-rational readback row R and then compared by the uniformity row U.

If the observation schedule is refined while the same R, W, U remain displayed, the classifier receives the same finite comparison data for every public read. The exceptional-window row A fixes the sites that are not part of the uniformity read, so the verdict outside A is unchanged. □

Lean4 ✓ `BEDC.Derived.EgorovUp.Egorov finite window uniformity`

Theorem 2204.4 (Egorov measure ledger nonescape)**.** *For every accepted* $\mathsf{Egorov}^{\uparrow}$ *carrier* E*, the measure information exported by the packet is exactly the finite ledger* L *attached to the exceptional-window row* A *and the ambient rows* M, Ω*. No consumer of the local* $\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$ *may replace* L *by a selected exceptional set, a countable null-set union, host almost-everywhere equality, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Lean4 ✓ `BEDC.Derived.EgorovUp.Egorov measure ledger nonescape`

Proof. The measure-facing coordinates of the carrier are M, Ω, A, L. The ledger L records the finite budget and containment checks for the displayed exceptional window A against the ambient measure and probability rows.

All replay into the public certificate passes through C, whose source includes L but contains no row for a selected exceptional set, countable null-set union, host almost-everywhere equality, or completeness of $\mathsf{Real}^{\uparrow}$. Thus the local name can export only the finite measure ledger already present in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$, obligation)

Scope closed: Definition 2204.1, Definition 2204.2, Definition 2204.3, and Definition 2204.4 seed the finite almost-everywhere uniformity packet over measure, probability, random-variable, stream, regular-rational, exceptional-window, uniformity, ledger, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Egorov}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full measure convergence theorems, countable choice, selected exceptional sets, host almost-everywhere equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exceptional-window budget exactness, finite-window uniformity, measure-ledger nonescape, componentwise transport, provenance, and local naming for BEDC.Derived.EgorovUp.

Constructive story: An $\mathsf{Egorov}^{\uparrow}$ packet is generated from Measure, ProbSpace, RandomVar, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 2205

A Concrete Naming Certificate for $\mathsf{UniformEntourageSpace}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces as entourage systems, the $\mathsf{UniformEntourageSpace}^{\uparrow}$ packet records the finite BEDC surface where diagonal, symmetry, refinement, and composition entourages are visible before any L10 completion consumer reads the route. It sits between the uniform-completion carrier of chapter 1473, the Cauchy-entourage request layer of chapter 1791, the real uniform structure of chapter 1624, and the metric completion boundary of chapter 1506. The packet names the entourage-side uniformity interface for completion handoff; it is not a quotient uniform space, a selected limit, an open-cover compactness theorem, countable choice, or ambient completeness.

Definition 2205.1 (Uniform entourage-space carrier)**.** *A* $\mathsf{UniformEntourageSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, D, S, R, C, L, T, K, P, N)$$

with the following displayed rows:

1. *E is the finite entourage-family row. It records the displayed binary neighborhoods used by every later Cauchy-entourage and completion-facing read.*
2. *D is the diagonal row, requiring each accepted entourage request to expose the finite diagonal witness it uses.*
3. *S is the symmetry row, recording the displayed reversal witness for the same entourage request.*
4. *R is the refinement row, recording how a smaller displayed entourage is chosen inside a larger displayed request without using a host filter quotient.*
5. *C is the composition row, recording the finite two-step entourage whose composite lies inside the requested entourage.*
6. *L is the L10 completion handoff row linking the displayed entourage data to* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyEntourageFilter}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumers.*
7. *T records componentwise* hsame *transport for the entourage family, diagonal, symmetry, refinement, composition, handoff, provenance, and local naming rows.*
8. *K records displayed* Cont *replay of the route*

$$E, D, S, R, C \longmapsto L \longmapsto T, P, N.$$

9. *P is the* Pkg *provenance over* UniformEntourageSpace$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CauchyEntourageFilter$^{\uparrow}$, RealUniformStructure$^{\uparrow}$, MetricCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{UniformEntourageSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $E, D, S, R, C, L, T, K, P, N$. The carrier has no coordinate for quotient uniform spaces, selected limits, open-cover compactness, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.UniformEntourageSpaceUp.UniformEntourageSpaceTasteGate single carrier alignment`

Theorem 2205.2 (Uniform entourage-space NameCert obligations)**.** *For every accepted* UniformEntourageSpace$^{\uparrow}$ *carrier*

$$U = (E, D, S, R, C, L, T, K, P, N),$$

the local NameCert$_{\text{UniformEntourageSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the entourage family, diagonal, symmetry, refinement, composition, L10 handoff, transport, replay, provenance, and local naming rows;*
2. *diagonal, symmetry, refinement, and composition exactness, requiring every uniformity read to remain inside the displayed rows D, S, R, C attached to the same entourage-family row E;*
3. *completion handoff, requiring each* UniformCompletion$^{\uparrow}$, CauchyEntourageFilter$^{\uparrow}$, RealUniformStructure$^{\uparrow}$, *and* MetricCompletion$^{\uparrow}$ *consumer named by L or P to read the displayed entourage route before any completion-facing boundary is available;*
4. *transport and replay stability through componentwise* hsame *in T and displayed* Cont *replay in K;*
5. *provenance exhaustion, requiring P and N to export no quotient uniform space, selected limit, open-cover compactness theorem, countable-choice selector, host equality, ambient completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2205.1. The projection exposes the entourage-family row, diagonal row, symmetry row, refinement row, composition row, L10 handoff, transport, replay, provenance, and local naming rows.

The uniformity obligations are componentwise. Diagonal reads are confined to D, symmetry reads to S, refinement reads to R, and composition reads to C, all over the same displayed entourage family E.

The completion handoff is the row L. A consumer for UniformCompletion$^{\uparrow}$, CauchyEntourageFilter$^{\uparrow}$, RealUniformStructure$^{\uparrow}$, or MetricCompletion$^{\uparrow}$ reaches the completion corridor only after the displayed route E, D, S, R, C has been replayed.

Finally T, K, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient, selected-limit, compactness, choice, host-equality, completeness, verification, or bridge coordinate, these rows exhaust the local certificate surface. □

Theorem 2205.3 (Uniform entourage-space completion handoff)**.** *Every L10 completion consumer of an accepted* UniformEntourageSpace$^{\uparrow}$ *carrier factors through the displayed route*

$$E, D, S, R, C \longmapsto L \longmapsto T, K, P, N.$$

Thus the consumer first reads the finite entourage family together with its diagonal, symmetry, refinement, and composition witnesses, and only then reaches the completion-facing handoff row. No consumer receives a quotient uniform space, selected limit, open-cover compactness theorem, countable-choice selector, host equality, or ambient completeness principle from this packet.

Proof. Project the accepted carrier $U = (E, D, S, R, C, L, T, K, P, N)$. The only uniformity data available to a consumer are the displayed entourage row and its four structural witnesses.

The handoff row L is downstream of those witnesses. It can name $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyEntourageFilter}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ consumers only as readers of the already displayed entourage route.

The structural rows preserve the same order. Componentwise hsame in T and displayed Cont replay in K cannot introduce a new quotient, selected-limit, compactness, choice, host-equality, or completeness coordinate.

Therefore every completion-facing read factors through the stated finite route and remains an entourage-surface handoff rather than an ambient completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEntourageSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEntourageSpace}^{\uparrow}$ packet is generated from a finite entourage-family row, diagonal row, symmetry row, refinement row, composition row, L10 completion handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEntourageSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2205.1, Definition 2205.2, and Definition 2205.3 seed the Kelley–Bourbaki entourage surface feeding UniformCompletion, CauchyEntourageFilter, RealUniformStructure, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEntourageSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient uniform spaces, selected limits, open-cover compactness, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, diagonal exactness, symmetry exactness, refinement exactness, composition exactness, completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformEntourageSpaceUp.

Chapter 2206

A Concrete Naming Certificate for $\mathsf{RealPrefixDiagnostic}^{\uparrow}$

Following Bishop–Bridges regular Cauchy real names and the finite-readback route of chapter 9238, the $\mathsf{RealPrefixDiagnostic}^{\uparrow}$ packet records the prefix-window diagnostic surface used before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits after the finite cylinder interface of chapter 1685, the observation windows of chapter 126, the regular rational source rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite diagnostic object for the Real-completeness route; it is not a proof of $\mathsf{Real}^{\uparrow}$ completeness, a quotient of Cauchy names, a selected host limit, countable choice, trichotomy, or host real equality.

Definition 2206.1 (Real prefix-diagnostic carrier)**.** *A* $\mathsf{RealPrefixDiagnostic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, W, S, R, D, E, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{RealBaireCylinder}^{\uparrow}$ *prefix-cylinder row naming the finite branch of the real-name read.*
2. *W is the prefix-window budget row, recording the bounded observation depth and the precision request that opened the diagnostic.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row inspected inside the same finite window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the inspected StreamName entries.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row comparing the requested precision with the regular rational readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after B, W, S, R, D have been replayed.*
7. *A is the diagnostic answer row. It records pass, refine-window, or reject-at-prefix outcomes as finite rows rather than as host real decisions.*
8. *H records componentwise* hsame *transport for prefix cylinders, window budgets, StreamName rows, RegSeqRat readbacks, dyadic tolerances, Real seals, diagnostic answers, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the route*

$$B, W \longmapsto S \longmapsto R \longmapsto D \longmapsto E, A, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RealPrefixDiagnostic}^{\uparrow}$, $\mathsf{RealBaireCylinder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealPrefixDiagnostic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $B, W, S, R, D, E, A, H, C, P, N$. The carrier has no coordinate for a completed real object, quotient Cauchy equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RealPrefixDiagnosticUp.RealPrefixDiagnosticTasteGate single carrier alignment`

Theorem 2206.2 (Real prefix-diagnostic NameCert obligations)**.** *For every accepted* $\mathsf{RealPrefixDiagnostic}^{\uparrow}$ *carrier*

$$Q = (B, W, S, R, D, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealPrefixDiagnostic}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the RealBaireCylinder prefix, prefix-window budget, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, diagnostic answer, transport, replay, provenance, and local naming rows;*
2. *prefix-window exactness, requiring every diagnostic request to read B and W before any StreamName entry is inspected;*
3. *regular readback discipline, requiring each inspected window to pass through S and R before the dyadic tolerance row D compares the requested precision;*
4. *Real-seal boundary, requiring the* $\mathsf{Real}^{\uparrow}$ *row E to be read only after the finite prefix, stream, regular-rational, and dyadic rows have been displayed;*
5. *diagnostic-answer confinement, requiring A to report only finite pass, refine-window, or reject-at-prefix rows and no host real decision;*
6. *transport, replay, and provenance exhaustion through H, C, P, N.*

No RealPrefixDiagnostic obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2206.1. The tuple displays the prefix cylinder, finite window budget, StreamName observations, RegSeqRat readback, dyadic tolerance, Real seal, diagnostic answer, and structural rows.

The prefix-window obligation reads B and W first. These rows bind the finite branch and depth of the diagnostic, so no later row can inspect an unbounded stream or a host real representative.

The regular readback obligation is the route through S, R, D. StreamName entries are inspected only within the displayed window, the regular rational row repacks that finite observation, and the dyadic row compares it with the requested tolerance.

The Real-seal boundary is row E, and the diagnostic answer is row A. The answer can report finite success, a request for a deeper window, or rejection at the displayed prefix, but it cannot decide equality of host reals or select a completed limit.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no completed-real, quotient, selected-limit, choice, trichotomy, host-equality, verification, or bridge coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealPrefixDiagnosticUp.RealPrefixDiagnosticCarrier namecert obligations`

Theorem 2206.3 (Real prefix-diagnostic Real-seal readback)**.** *Every Real-completeness route consumer of an accepted* $\mathsf{RealPrefixDiagnostic}^{\uparrow}$ *packet factors its finite diagnostic read through*

$$B, W \longmapsto S \longmapsto R \longmapsto D \longmapsto E, A, H, C, P, N.$$

Thus prefix-cylinder data are inspected before StreamName rows, StreamName rows before RegSeqRat readback, RegSeqRat rows before dyadic tolerance comparison, and the $\mathsf{Real}^{\uparrow}$ *seal only after those finite rows are displayed. The packet exports no completed* $\mathsf{Real}^{\uparrow}$ *proof, quotient Cauchy class, selected host limit, countable-choice schedule, trichotomy, or host real equality.*

Proof. Begin with a finite diagnostic read for the Real-completeness route. The only prefix data exposed by the carrier are B and W, so the consumer first fixes the displayed cylinder and bounded observation depth.

The stream and rational readback stages then follow. The row S exposes only the StreamName entries inside that window, while R repacks those entries as regular rational readback data.

The tolerance check is row D. It compares the requested precision with the displayed readback before the Real seal is available.

Only then can the route read E and the diagnostic answer A. Transport through H, replay through C, and provenance through P, N preserve this order, so the consumer receives a finite prefix diagnostic rather than a completed-real or choice-based object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealPrefixDiagnostic}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealPrefixDiagnostic}^{\uparrow}$ packet is generated from a RealBaireCylinder prefix row, finite prefix-window budget, StreamName row, RegSeqRat readback, DyadicRatCore tolerance row, RealUp seal row, diagnostic answer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealPrefixDiagnostic}^{\uparrow}}$, seed)

Scope closed: Definition 2206.1, Definition 2206.2, and Definition 2206.3 seed the finite prefix-window diagnostic route before a RealUp seal is read.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealPrefixDiagnostic}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient Cauchy equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, prefix-window exactness, regular readback discipline, dyadic tolerance comparison, Real-seal boundary, diagnostic-answer confinement, transport, replay, provenance, and local naming for BEDC.Derived.RealPrefixDiagnosticUp.

Chapter 2207

A Concrete Naming Certificate for $\mathsf{BesselInequality}^{\uparrow}$

Following Riesz–Fischer/Hilbert-space treatments of Bessel's inequality, the $\mathsf{BesselInequality}^{\uparrow}$ packet records the finite BEDC surface where an orthogonal family, coefficient ledger, partial-square-sum bound, and norm handoff are visible before any Parseval, Fourier-series, orthogonal-projection, or Hilbert-completeness consumer reads the route. It sits beside the Hilbert carrier of chapter 91, the inner-product classifier of chapter 86, the norm surface of chapter 89, and the vector-space reduct of chapter 47. The packet names the inequality boundary only; it is not Parseval completeness, an orthogonal expansion theorem, a projection theorem, a selected orthonormal basis, or a host summability principle.

Definition 2207.1 (Bessel inequality carrier)**.** *A* $\mathsf{BesselInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (H, I, N, V, O, C, S, L, T, R, P, Q)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row naming the ambient complete inner-product space.*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same carried vector endpoints, including the coefficient readback operation $\langle x, e\rangle_H$.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row over the same carrier, including the squared-norm comparison used by the inequality bound.*
4. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *reduct containing the zero, addition, scalar-action, finite-combination, and classifier rows used by the coefficient spine.*
5. *O is the orthogonal-family row, recording a finite displayed family $e_0, \ldots, e_{k-1}$ whose pairwise inner products classify as zero and whose norm rows classify as unit rows when the normalized form is requested.*
6. *C is the coefficient ledger row, recording the finite coefficient list $\langle x, e_i\rangle_H$ for a carried vector x and every displayed family position $i < k$.*
7. *S is the partial-square-sum row, recording the finite sum $\sum_{i<k} |\langle x, e_i\rangle_H|^2$ through the real scalar surface read from I and N.*

8. *L is the Hilbert-bound handoff row, recording the comparison*

$$\sum_{i<k} |\langle x, e_i\rangle_H|^2 \le \|x\|_H^2$$

as a displayed finite inequality row rather than as a host Hilbert-space theorem.

9. *T records componentwise* hsame *transport for Hilbert carriers, inner-product rows, norm rows, vector-space rows, orthogonal-family rows, coefficient ledgers, partial sums, handoff comparisons, replay, provenance, and local naming.*

10. *R records displayed* Cont *replay of the route*

$$H, I, N, V, O, C, S \longmapsto L \longmapsto T, P, Q.$$

11. *P is the* Pkg *provenance over* $\mathsf{BesselInequality}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *Q is the local* $\mathsf{NameCert}_{\mathsf{BesselInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $H, I, N, V, O, C, S, L, T, R, P, Q$. The carrier has no coordinate for Parseval equality, a complete orthonormal basis, projection minimization, Fourier convergence, choice of basis, host summability, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BesselInequalityUp.BesselInequalityTasteGate single carrier alignment`

Theorem 2207.2 (Bessel inequality NameCert obligations)**.** *For every accepted* $\mathsf{BesselInequality}^{\uparrow}$ *carrier*

$$B = (H, I, N, V, O, C, S, L, T, R, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{BesselInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the Hilbert, inner-product, norm, vector-space, orthogonal-family, coefficient-ledger, partial-square-sum, Hilbert-bound handoff, transport, replay, provenance, and local naming rows;*

2. *dependency coherence, requiring the* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, *and* $\mathsf{VecSpace}^{\uparrow}$ *rows to read the same carried vector endpoints and the same classifier route;*

3. *orthogonality exactness, requiring O to display the finite family positions and the pairwise zero inner-product rows used by the coefficient computation;*

4. *coefficient and square-sum exactness, requiring every coefficient in C and every summand in S to be read from the displayed inner-product and norm surfaces;*

5. *Hilbert-bound handoff, requiring the inequality row L to be downstream of O, C, S and upstream of any Parseval, Fourier-series, orthogonal-projection, or Hilbert-completeness consumer;*

6. *transport, replay, and provenance exhaustion through T, R, P, Q.*

No BesselInequality obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 2207.1. The tuple displays the Hilbert carrier, the inner-product and norm rows, the vector-space reduct, the orthogonal family, the coefficient ledger, the square-sum row, and the structural rows.

The dependency rows are read over the same carried endpoints. Thus the coefficient operation from $\mathsf{InnerProduct}^{\uparrow}$, the norm comparison from $\mathsf{Norm}^{\uparrow}$, and the finite linear-combination support from $\mathsf{VecSpace}^{\uparrow}$ remain tied to the ambient $\mathsf{Hilbert}^{\uparrow}$ carrier.

Orthogonality exactness is confined to O. Every coefficient in C is then obtained by reading the inner-product row against one displayed orthogonal-family entry, and S repacks those finitely many coefficient norms as the displayed partial square sum.

The bound row L is downstream of those finite rows. Since T, R, P, Q only transport, replay, source, and name the same route, the packet exports the Bessel inequality handoff and no Parseval equality, basis selection, projection minimizer, Fourier convergence, host summability, Lean verification, or bridge checking. □

Theorem 2207.3 (Bessel inequality Hilbert-bound handoff)**.** *Every downstream consumer of an accepted* $\mathsf{BesselInequality}^{\uparrow}$ *packet that uses the Hilbert bound factors through the displayed route*

$$H, I, N, V, O, C, S \longmapsto L \longmapsto T, R, P, Q.$$

Thus the consumer first reads the shared $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*, and* $\mathsf{VecSpace}^{\uparrow}$ *dependency surface, then the finite orthogonal family and coefficient square-sum ledger, and only then reaches the inequality row. The packet exports no Parseval equality, complete orthonormal basis, projection theorem, Fourier convergence theorem, selected basis, or host summability principle.*

Proof. Begin with an accepted packet $B = (H, I, N, V, O, C, S, L, T, R, P, Q)$. The shared dependency surface is visible first: H gives the Hilbert carrier, I gives coefficient readback, N gives the norm comparison, and V gives the finite vector-space operations.

The finite orthogonal-family and coefficient stages then follow. The row O supplies the displayed family and its orthogonality witnesses, C records the coefficient list for the carried vector, and S records the finite partial square sum.

Only after those rows are displayed can the consumer read L. Transport through T, replay through R, and provenance through P, Q preserve this order, so the downstream object receives a Hilbert-bound handoff rather than a stronger completeness, basis, projection, or Fourier-series theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BesselInequality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BesselInequality}^{\uparrow}$ packet is generated from a Hilbert carrier row, inner-product row, norm row, vector-space reduct, finite orthogonal-family row, coefficient-ledger row, partial-square-sum row, Hilbert-bound handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BesselInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2207.1, Definition 2207.2, and Definition 2207.3 seed the finite Hilbert-space coefficient-bound surface consumed by Parseval, Fourier-series, orthogonal-projection, and Hilbert-completeness chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BesselInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Parseval equality, complete orthonormal bases, projection minimization, Fourier convergence, selected bases, host summability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, dependency coherence, orthogonality exactness, coefficient and square-sum exactness, Hilbert-bound handoff, transport, replay, provenance, and local naming for BEDC.Derived.BesselInequalityUp.

Chapter 2208

A Concrete Naming Certificate for $\mathsf{RegularFilterRealization}^{\uparrow}$

Following Bishop and Bridges and the Bourbaki–Kelley filterbase terminology, the $\mathsf{RegularFilterRealization}^{\uparrow}$ packet records the L10 finite route by which a displayed Cauchy filterbase is read as regular rational Cauchy data before a $\mathsf{Real}^{\uparrow}$ seal is admitted. It sits over the finite Cauchy filter surface of chapter 333, the basis factorization of chapter 1501, the regular filter package of chapter 1445, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a realization boundary, not a completeness principle: it names the finite rows consumed by the route from Cauchy filters to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seals without selecting quotient filters or host limits.

Definition 2208.1 (RegularFilterRealization carrier)**.** *A* $\mathsf{RegularFilterRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, A, G, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *finite-filterbase row, including its visible base packets, refinement relation, and Cauchy pressure ledger.*
2. *A is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *factorization row that exposes the selected finite basis windows used by the realization.*
3. *G is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *obligation package binding the filterbase, basis, stream, dyadic, regular readback, and seal rows into one route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened by the basis and consumed by the regular readback.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the same windows, including endpoint and radius comparison rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from A, S, D through the regular Cauchy filter package G.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, A, G, S, D, R have been displayed.*
8. *H records componentwise* hsame *transport for B, A, G, S, D, R, E and for the structural rows below.*

9. *C records the displayed* Cont *replay from filterbase refinements to basis windows, from windows and dyadic rows to the regular rational readback, and from the readback to the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularFilterRealization}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularFilterRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven coordinates and their displayed transports. It has no coordinate for quotient filters, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Theorem 2208.2 (RegularFilterRealization NameCert obligations)**.** *For every accepted* $\mathsf{RegularFilterRealization}^{\uparrow}$ *carrier $F = (B, A, G, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularFilterRealization}^{\uparrow}}$ *surface consists exactly of:*

1. *CauchyFilter source exposure through B;*
2. *CauchyFilterBasis factorization through A;*
3. *RegularCauchyFilter obligation packaging through G;*
4. *StreamName finite-window exposure through S;*
5. *DyadicRatCore tolerance exactness through D;*
6. *RegSeqRat readback through R;*
7. *RealUp seal handoff through E;*
8. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

No obligation is discharged by a quotient filter, a selected host limit, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Proof. Project the carrier of Definition 2208.1. The visible coordinates are precisely $B, A, G, S, D, R, E, H, C, P, N$, so every local obligation must be read from those rows.

The source and basis side is finite. The row B supplies the Cauchy filterbase data, while A exposes the basis windows used by the realization. These rows place the construction in the same filterbase surface as chapter 333 and its basis factorization.

The regular readback is then forced through G, S, D, R. The package G binds the RegularCauchyFilter route, S lists the requested finite windows, D records their dyadic tolerance ledger, and R is the only regular rational readback row.

Finally E, H, C, P, N close the seal-facing certificate. Since the excluded quotient, choice, host-equality, completeness, and bridge coordinates are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularFilterRealizationUp.RegularFilterRealizationTasteGate single carrier alignment`

Theorem 2208.3 (RegularFilterRealization RegSeqRat factorization)**.** *Let $F = (B, A, G, S, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularFilterRealization}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by F factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqR}$$

Concretely, the read first exposes B, then the basis row A, then the obligation package G, then the finite windows S and dyadic ledger D, and only then reaches the readback row R through the recorded H, C, P, N data. The factorization exports no quotient filter, selected limit, countable-choice selector, host real equality, or ambient completeness coordinate.

Proof. Start from the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate in the carrier, namely R. By definition of F, this row is not stored independently of the filterbase data; it is downstream from B, A, G, S, D.

The filterbase is first read through B, and the displayed basis windows are then read through A. The obligation package G is the regular Cauchy filter row that ties those windows to the stream and dyadic data.

The finite readback step uses S and D. The schedule S lists exactly the requested windows, and D records the dyadic tolerance cells for those same windows. Thus the readback R is determined by displayed regular data rather than by a host-selected limit.

The remaining rows H, C, P, N transport, replay, package, and name the same route. No carrier coordinate remains for quotient filters, choice, host real equality, or ambient completeness, so none can be exported by the factorization. □

Theorem 2208.4 (RegularFilterRealization Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularFilterRealization}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row named in Definition 2208.3. The consumer may inspect the source filterbase B, the basis row A, the regular filter package G, the stream windows S, the dyadic ledger D, the regular readback R, and the structural rows H, C, P, N before reading the seal E. It receives no quotient-filter equality, selected host limit, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or checked bridge credential from the packet.*

Proof. Project the seal row E from the accepted carrier. The definition places E after the source filterbase, basis factorization, regular filter package, window schedule, dyadic ledger, and regular rational readback.

Apply Definition 2208.3 to the preceding readback. The Real-facing consumer must therefore pass through R, and R itself has already passed through B, A, G, S, D.

The structural rows do not add another seal route. They transport, replay, package, and name the same finite handoff. Since E is the only $\mathsf{Real}^{\uparrow}$ coordinate and the excluded principles are absent from the carrier, every seal read factors through the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularFilterRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularFilterRealization}^{\uparrow}$ packet is generated from a CauchyFilter finite-filterbase row, a CauchyFilterBasis factorization row, a RegularCauchyFilter obligation package, StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal row, componentwise transport, displayed replay, package provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularFilterRealization}^{\uparrow}}$, seed)

Scope closed: Definition 2208.1, Definition 2208.2, Definition 2208.3, and Definition 2208.4 bind the finite realization route from Cauchy filters through basis windows and regular rational readback to the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularFilterRealization}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host real equality, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, CauchyFilter source exposure, CauchyFilterBasis factorization, RegularCauchyFilter package handoff, StreamName window exposure, DyadicRatCore tolerance exactness, RegSeqRat readback, and RealUp seal handoff for the same packet.

Chapter 2209

A Concrete Naming Certificate for $\mathsf{SchurMatrixSummability}^{\uparrow}$

Following Schur's matrix test and Hardy's summability treatment, the $\mathsf{SchurMatrixSummability}^{\uparrow}$ packet records the finite BEDC certificate by which a bounded matrix kernel sends regular real-name source windows through finite transformed rows without escaping the L10 readback discipline. It refines the matrix-window route of chapter 1682 and the regular matrix surface of chapter 2126, contains the Cesaro rows of chapter 1765 and chapter 1827 as named provenance surfaces, reads source and transformed rows through chapter 301, keeps rational coefficient and tolerance budgets inside chapter 291, and reaches the endpoint only through chapter 38. The chapter names the finite bounded-kernel summability certificate; it does not import a host Schur theorem, a completed infinite matrix, quotient equality of real-name streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2209.1 (Schur matrix summability carrier)**.** *A* $\mathsf{SchurMatrixSummability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, B, C, W, T, R, E, H, Q, P, N)$$

with the following displayed rows:

1. *M is the matrix coefficient-window row, listing finite matrix rows, finite column windows, coefficient approximants, truncation thresholds, and row requests currently inspected.*
2. *B is the row-bound ledger, recording the dyadic bounds that control the displayed row kernels before any transformed finite row is exposed.*
3. *C is the column-tail ledger, recording the displayed tail requests whose coefficient mass is certified small beyond the inspected finite column windows.*
4. *W is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the source observations allowed to feed the displayed matrix windows.*
5. *T is the transformed finite-row surface constructed from the coefficient windows in M, the row-bound ledger B, the column-tail ledger C, and the source windows W.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for both the source-window reads and the transformed rows exposed by T.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, B, C, W, T, R have been displayed.*

8. *H stores componentwise* hsame *transport for matrix windows, row bounds, column tails, source windows, transformed rows, regular readback, real seals, continuation rows, provenance, and local naming.*

9. *Q stores displayed* Cont *replay from matrix-row inspection to source-window selection, from row-bound and column-tail ledgers to transformed-row construction, from transformed rows to regular readback, and from readback to the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{SchurMatrixSummability}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{SilvermanToeplitz}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurMatrixSummability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, B, C, W, T, R, E, H, Q, P, N$. It has no coordinate for a host Schur theorem, a completed infinite matrix, quotient stream equality, selected external limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SchurMatrixSummabilityUp.SchurMatrixSummabilityTasteGate single carrier alignment`

Theorem 2209.2 (Schur matrix summability NameCert obligations)**.** *For every accepted* $\mathsf{SchurMatrixSummability}^{\uparrow}$ *carrier $K = (M, B, C, W, T, R, E, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SchurMatrixSummability}^{\uparrow}}$ *surface consists exactly of the following seven finite obligation rows:*

1. *carrier admission for the displayed matrix coefficient windows, row-bound ledger, column-tail ledger, source windows, transformed finite rows, regular readback, real seal, transport, continuation, provenance, and naming rows;*

2. *row-bound exactness, requiring every bounded-kernel claim used by the transform to be read from B after the corresponding coefficient windows in M have been displayed;*

3. *column-tail exactness, requiring every tail-smallness claim to pass through the displayed column-tail ledger C and the same finite matrix-window row M;*

4. *source-window transform construction, requiring every transformed row in T to be computed from M, B, C, W before any transformed read is exposed;*

5. *regular readback, requiring source-window and transformed-row reads to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R with dyadic budgets supplied by the same finite ledger data;*

6. *Real-seal nonescape, requiring the endpoint row E to remain downstream of M, B, C, W, T, R and to export no host summability or ambient completeness field;*

7. *ledger preservation, requiring H, Q, P, N to preserve the same finite route under componentwise transport, continuation replay, package provenance, and local naming.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 2209.1. Carrier admission is exactly the displayed inventory $M, B, C, W, T, R, E, H, Q, P, N$, so the local certificate has no detached coordinate for an infinite matrix, a host Schur theorem, or an external completion principle.

The bounded-kernel rows are read before the transform. The inspected coefficient windows in M expose the finite row and column data, the row-bound ledger B controls the displayed row kernels, and the column-tail ledger C records the tail-smallness requests that remain after the finite windows have been chosen.

The source and transform rows are then forced. The only source observations are the $\mathsf{StreamName}^{\uparrow}$ windows W, and the transformed finite-row surface T is constructed from M, B, C, W. This is the Schur-bounded counterpart of the Toeplitz and Silverman–Toeplitz matrix routes referenced by P.

Finally the readback and seal rows close the route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads both source and transformed windows, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of that readback. The structural rows H, Q, P, N transport, replay, source, and name the same packet, so the seven listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurMatrixSummability}^{\uparrow}}$.

Constructive story: A $\mathsf{SchurMatrixSummability}^{\uparrow}$ packet is generated from finite matrix coefficient windows, a row-bound ledger, a column-tail ledger, source StreamName windows, transformed finite rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurMatrixSummability}^{\uparrow}}$, seed)

Scope closed: Definition 2209.1 and Definition 2209.2 seed the bounded-kernel matrix summability certificate surface from finite matrix rows to L10 regular real-name readback and Real-facing seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurMatrixSummability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Schur theorems, completed infinite matrices, selected source limits, quotient equality of real-name streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row-bound exactness, column-tail exactness, source-window transform construction, regular readback, Real-seal nonescape, ledger preservation, and TasteGate field faithfulness for BEDC.Derived.SchurMatrixSummabilityUp.

Chapter 2210

A Concrete Naming Certificate for $\mathsf{WeakDerivative}^{\uparrow}$

Following Evans and standard distribution theory, the $\mathsf{WeakDerivative}^{\uparrow}$ packet records the finite test-action surface by which a distribution-facing derivative is named without importing host differentiation. It sits after the distribution endpoint interface of chapter 186, the distribution-space test-action interface of chapter 8023, the Sobolev weak-gradient ledger of chapter 440, the real seal of chapter 38, the regular rational readback surface of chapter 301, and the finite stream windows of chapter 126. The chapter records only the finite carrier, its naming obligations, and the Sobolev handoff boundary.

Definition 2210.1 (Weak derivative carrier)**.** *A* $\mathsf{WeakDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, D, E, Q, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the test-action row, listing the finite test probes, their displayed support windows, the sign convention for the chosen coordinate direction, and the source action values requested from the input object.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *row for the input distribution or locally integrable object, read only through the test actions named in A.*
3. *E is the candidate derivative row, giving a second distribution-facing action on the same displayed tests.*
4. *Q is the shared* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every displayed test action and candidate derivative action.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family that selects the finite test probes, readback windows, and comparison tolerances visible to this packet.*
6. *H stores componentwise* hsame *transport for the test-action, distribution, derivative, readback, window, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay for the route from test selection to distribution action, from distribution action to candidate derivative action, and from both actions to the shared readback row.*

8. *P is the* Pkg *provenance over* WeakDerivative$^\uparrow$, Distribution$^\uparrow$, Sobolev$^\uparrow$, Real$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{WeakDerivative}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $A, D, E, Q, W, H, C, P, N$. It has no coordinate for a host derivative operator, a completed test-function space, quotient equality of distributions, density of smooth functions, integration by parts outside the displayed tests, or ambient Sobolev completeness.

Lean4 ✓ `BEDC.Derived.WeakDerivativeUp.WeakDerivativeTasteGate single carrier alignment`

Theorem 2210.2 (Weak derivative NameCert obligations)**.** *For every accepted* WeakDerivative$^\uparrow$ *carrier $K = (A, D, E, Q, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakDerivative}^\uparrow}$ *surface consists exactly of carrier admission for the nine displayed rows, test-action coverage through A, input distribution access through D, candidate derivative access through E, shared* Real$^\uparrow$ *and* RegSeqRat$^\uparrow$ *readback through Q, finite window selection through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 2210.1. The test-facing data available to the classifier are exactly the selected tests in A, the distribution action in D, the candidate derivative action in E, the readback row Q, and the stream windows W.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the classifier has no coordinate for a host derivative operator, completed test-function space, quotient equality, density theorem, or ambient Sobolev completeness, those objects cannot discharge a local NameCert obligation.

Thus the obligation surface is exactly the displayed finite packet. □

Theorem 2210.3 (Weak derivative Sobolev handoff)**.** *Every* Sobolev$^\uparrow$ *consumer read exported by an accepted* WeakDerivative$^\uparrow$ *carrier factors through the finite route*

$$A \longmapsto D, E \longmapsto Q \longmapsto W \longmapsto H, C, P, N.$$

The handoff supplies only the displayed test-action ledger, distribution and candidate derivative actions, real and regular-rational readback, finite stream windows, transport, replay, provenance, and local naming. It does not supply a completed Sobolev space, quotient equality of functions, density of smooth tests, trace theory, compact embedding, or a host weak-gradient operator.

Proof. Start with a Sobolev-facing request for weak-derivative data. The carrier first fixes the test-action row A, so the request is constrained to the displayed finite probes and support windows.

The rows D and E then expose the input distribution action and the candidate derivative action on those same tests. The shared readback row Q and the window row W are the only places where real or regular-rational observations can be inspected.

Finally read H, C, P, N. These rows preserve and name the same finite route for the Sobolev$^\uparrow$ handoff, and no absent analytic object can be recovered from the carrier. □

Theorem 2210.4 (Weak derivative distribution-space boundary)**.** *Any* WeakDerivative$^\uparrow$ *read consuming a* DistributionSpace$^\uparrow$ *packet must expose the distribution-space test-action row, the candidate derivative action,* RegSeqRat$^\uparrow$ *and* Real$^\uparrow$ *scalar readback,* StreamName$^\uparrow$ *windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before that read can be used by* Sobolev$^\uparrow$.

Proof. Start with a weak-derivative read whose input action is supplied by a $\mathsf{DistributionSpace}^{\uparrow}$ packet. By Definition 8023.3, the distribution-space packet first exposes finite test windows and a scalar action ledger before any derivative-facing boundary can be consumed.

Read the $\mathsf{WeakDerivative}^{\uparrow}$ carrier next. The candidate derivative action is the row E, and the shared readback row Q supplies the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar boundary for both the input action and the candidate derivative action.

The finite windows remain visible through W, while H, C, P, N transport, replay, source, and name the same route. Applying Definition 2210.3 then forces the Sobolev-facing consumer to pass through those rows before any handoff is accepted. □

Falsifiable prediction: Every $\mathsf{WeakDerivative}^{\uparrow}$ Sobolev-facing read factors through a displayed test-action row, distribution action row, candidate derivative row, shared RealUp and RegSeqRatUp readback, StreamName windows, hsame transport, Cont replay, package provenance, and local naming row.

Independence witness: distribution, sobolev, real, regseqrat, streamname: $\mathsf{WeakDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the test-action ledger, the input distribution action, the candidate derivative action, shared readback, finite windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakDerivative}^{\uparrow}$ packet is generated from finite test-action rows, an input DistributionUp row, a candidate derivative row, shared RealUp and RegSeqRatUp readback, StreamName windows, componentwise transports, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2210.1, Definition 2210.2, Definition 2210.3, and Definition 2210.4 bind the finite weak-action surface over $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DistributionSpace}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed test-function spaces, quotient equality of distributions, density of smooth functions, integration by parts beyond displayed tests, Sobolev completeness, compact embedding, trace theory, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, test-action exactness, readback stability, Sobolev handoff nonescape, and TasteGate field faithfulness for BEDC.Derived.WeakDerivativeUp.

Chapter 2211

A Concrete Naming Certificate for $\mathsf{UniformCompleteSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{UniformCompleteSpace}^{\uparrow}$ packet records the displayed completeness property of a uniform-space carrier as finite Cauchy-filter convergence data. It sits after the completion interfaces of chapter 1506 and chapter 1473, reads the Cauchy-filter and Cauchy-net surfaces of chapter 333 and chapter 589, and keeps separated comparison available through chapter 1411. The packet generalizes the metric-completion route by recording convergence of displayed uniform Cauchy data without constructing quotient completions, choosing representatives, invoking countable choice, importing open-cover compactness, or asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2211.1 (Uniform complete-space carrier)**.** *A* $\mathsf{UniformCompleteSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, F, N, M, C, Q, R, E, H, T, P, A)$$

with the following displayed rows:

1. *S is the underlying uniform-space row, carrying the finite entourage filterbase, diagonal, refinement, symmetry, and composition data inspected by completion-facing consumers.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row admitted over S.*
3. *N is the* $\mathsf{CauchyNet}^{\uparrow}$ *row giving the directed-window presentation of the same Cauchy demand when a net-style consumer is used.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row available when the uniform data is read through a displayed metric-completion presentation.*
5. *C is the* $\mathsf{UniformCompletion}^{\uparrow}$ *comparison row available when the uniform data is read through the existing uniform-completion presentation.*
6. *Q is the convergence boundary row asserting that the displayed Cauchy filter has a completion-facing limit route inside the packet.*
7. *R is the regular rational readback row exposed only after S, F, N, M, C, Q have been displayed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for numerical readback consumers.*
9. *H records componentwise* hsame *transport for the uniform surface, Cauchy data, completion comparisons, convergence boundary, readback, seal, continuation, provenance, and local naming rows.*

10. *T records displayed* Cont *replay from a finite entourage request through Cauchy-filter or Cauchy-net data, completion comparison, convergence boundary, readback, and seal.*
11. *P is the* Pkg *provenance over* $\mathsf{UniformCompleteSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformCompleteSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient completions, selected representatives, countable choice, open-cover compactness, host uniform equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCompleteSpaceUp.UniformCompleteSpaceTasteGate single carrier alignment`

Theorem 2211.2 (Uniform complete-space NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompleteSpace}^{\uparrow}$ *carrier* $U = (S, F, N, M, C, Q, R, E, H, T, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCompleteSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed uniform-space row, Cauchy-filter row, Cauchy-net row, metric-completion comparison, uniform-completion comparison, convergence boundary, regular readback, Real seal, transport, replay, provenance, and local naming row;*
2. *Cauchy request factorization, requiring every accepted completion-facing read to begin with the same finite entourage data in S and then pass through either F or N before any completion comparison is exposed;*
3. *completion comparison discipline, requiring M and C to compare only the displayed metric and uniform completion presentations of the same Cauchy route;*
4. *Cauchy-filter seal routing, requiring Q to remain downstream of S, F, N, M, C and upstream of only R, E, H, T, P, A;*
5. *separated comparison non-escape, allowing separated reads only through the provenance named by P and never through host equality or quotient completion equality;*
6. *ledger non-escape, excluding quotient completions, selected representatives, countable choice, open-cover compactness, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2211.1. The projection exposes exactly $S, F, N, M, C, Q, R, E, H, T, P, A$.

Cauchy request factorization begins with the finite entourage surface S. The request is then read through the displayed filter row F or the directed net row N, so a completion-facing consumer receives Cauchy evidence before it sees any completion comparison.

The completion comparison rows M and C are downstream of that Cauchy route. They compare the metric-completion and uniform-completion presentations of the same displayed request, and the convergence boundary Q is sealed only after those presentations have been read.

Finally R, E, H, T, P, A provide readback, terminal sealing, componentwise transport, replay, provenance, and local naming for the same route. Since no listed row is a quotient completion, selected representative, choice schedule, open-cover compactness certificate, or proof of $\mathsf{Real}^{\uparrow}$ completeness, the local NameCert surface cannot export any of those objects. □

Theorem 2211.3 (Uniform complete-space Cauchy-filter seal)**.** *Let* $U = (S, F, N, M, C, Q, R, E, H, T, P, A)$ *be an accepted* $\mathsf{UniformCompleteSpace}^{\uparrow}$ *carrier. Every Cauchy-filter convergence read factors through the displayed route*

$$S \longmapsto F, N \longmapsto M, C \longmapsto Q \longmapsto R, E \longmapsto H, T, P, A.$$

Thus the packet seals the finite Cauchy-filter route for uniform completeness without adding quotient completions, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Cauchy-filter convergence read. The uniform-space row S fixes the finite entourage demand before any Cauchy or completion row is inspected.

The Cauchy data is then displayed by F and, when a directed presentation is needed, by N. Both rows remain tied to the same request from S.

The comparison rows M and C read the metric-completion and uniform-completion presentations of that displayed Cauchy route. The boundary row Q is downstream of those comparisons, so the convergence seal is attached to the visible filter and net data rather than to a host completed object.

The readback and terminal rows R and E are exposed only after Q, while H, T, P, A transport, replay, source, and name the same route. The carrier therefore has no row from which quotient completions, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompleteSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompleteSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2211.1, Definition 2211.2, and Definition 2211.3 seed the finite uniform completeness packet over displayed uniform, Cauchy-filter, Cauchy-net, metric-completion, uniform-completion, separated-comparison, readback, and seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompleteSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, open-cover compactness, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy request factorization, completion comparison discipline, Cauchy-filter seal routing, separated comparison non-escape, and ledger non-escape for BEDC.Derived.UniformCompleteSpac

Constructive story: A $\mathsf{UniformCompleteSpace}^{\uparrow}$ packet is generated from a uniform-space row, CauchyFilter and CauchyNet rows, MetricCompletion and UniformCompletion comparison rows, a convergence boundary, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2212

A Concrete Naming Certificate for MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$

The MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ packet realizes the finite critical-pair boundary suggested by chapter 9378. It sits between the redex-frontier surface of chapter 1008, the closed-normal confluence fragment of chapter 737, the bounded equality checker of chapter 752, and the subject-reduction route classifier of chapter 785. The critical-pair table is a consumer of chapter 1008, and the common-reduct table is bounded by chapter 737 before any bounded equality checker is read. The packet names displayed critical-pair and common-reduct rows; it is not a global confluence theorem.

Falsifiable prediction: Any accepted MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ packet exports only displayed critical-pair and common-reduct rows, together with redex-frontier, normal-fragment, bounded-checker, route-classifier, transport, replay, provenance, and local naming data. It exports no global confluence theorem.

Independence witness: MetaCICRedexFrontier$^{\uparrow}$, MetaCICConfluenceNormalFragment$^{\uparrow}$, BoundedNormalEqualityCh and SubjectReductionRouteClassifier$^{\uparrow}$: these siblings respectively expose redex-frontier rows, closed-normal confluence fragments, bounded normal-equality checking, and subject-reduction route classification; none is bijective with a packet that jointly records critical-pair enumeration and common-reduct boundary rows.

Definition 2212.1 (MetaCIC confluence critical-pair boundary carrier)**.** *A* MetaCICConfluenceCriticalPairBoundary *carrier is a finite* BHist *packet*

$$B = (R, K, J, E, D, S, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the* MetaCICRedexFrontier$^{\uparrow}$ *row from which beta-redex, application, lambda-domain, and Pi-domain frontier cells are read.*
2. *K is the critical-pair table row, listing the finite overlapping frontier cases inspected by the packet.*
3. *J is the common-reduct table row, listing the displayed join candidates for the accepted critical-pair rows.*
4. *E is the* BoundedNormalEqualityChecker$^{\uparrow}$ *row used to compare bounded normal forms when a join candidate is checked by bounded normal equality.*

5. *D is the* $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ *row used for closed normal and atom-shaped source cases.*
6. *S is the* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$ *separation row, recording that subject-reduction discharge routes are named separately from confluence critical-pair rows.*
7. *U is the boundary row stating that the packet does not prove arbitrary closed-term Church–Rosser, full parallel diamond, or residual-substitution compatibility.*
8. *H records componentwise* hsame *transport for the frontier, critical-pair table, common-reduct table, bounded-checker, normal-fragment, route-classifier, boundary, provenance, and local naming rows.*
9. *C records displayed* Cont *replay by which audit consumers read the critical-pair table and common-reduct table before any bounded checker or normal-fragment handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICRedexFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a global confluence theorem, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or unchecked subject-reduction discharge.

Theorem 2212.2 (MetaCIC confluence critical-pair boundary obligations)**.** *For every accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier* $B = (R, K, J, E, D, S, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConf}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the redex-frontier row, critical-pair table, common-reduct table, bounded equality checker, closed-normal fragment, subject-reduction route classifier, boundary row, transport, replay, provenance, and local naming row;*
2. *critical-pair display, requiring each accepted pair to be read from the finite frontier cases in R and K;*
3. *common-reduct display, requiring each accepted join candidate to be read from J before any bounded checker or normal-fragment row is consumed;*
4. *bounded and normal handoff, requiring E and D to consume only the displayed critical-pair and common-reduct rows;*
5. *subject-reduction separation, requiring S to keep typing-discharge routes outside the confluence boundary;*
6. *ledger non-escape, requiring U, P, N to export no global confluence theorem, arbitrary closed-term Church–Rosser, full parallel diamond, or residual-substitution compatibility.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2212.1. The projection exposes exactly $R, K, J, E, D, S, U, H, C, P, N$.

The critical-pair display obligation reads R and K first. Thus the packet can only enumerate finite frontier overlaps that are explicitly recorded in the carrier.

The common-reduct display obligation is downstream of that enumeration. The row J supplies the join candidates before the bounded checker E or the closed-normal fragment D may be consulted.

The subject-reduction row S separates typing-discharge routes from the confluence boundary, and U records the non-claim surface. Since the remaining rows H, C, P, N only transport, replay, source, and name the displayed route, the packet cannot export global confluence or residual-substitution compatibility. □

Theorem 2212.3 (MetaCIC confluence critical-pair common-reduct boundary)**.** *Let $B = (R, K, J, E, D, S, U, H, C$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Every confluence audit read from B factors through the displayed route*

$$R \longmapsto K \longmapsto J \longmapsto E, D \longmapsto S, U \longmapsto H, C, P, N.$$

Consequently the packet supplies a finite critical-pair and common-reduct boundary, not a proof of arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or subject-reduction discharge.

Proof. Start with an audit consumer of the packet. The consumer first reads the redex-frontier row R and then the critical-pair table K, so the inspected cases are finite displayed rows.

The common-reduct table J is read next. Any bounded normal-equality check or closed-normal confluence fragment receives only the join candidates already displayed by J.

The route-classifier row S and boundary row U then fix the non-claim surface. They keep subject-reduction discharge and residual-substitution compatibility outside the packet.

Finally H, C, P, N transport, replay, source, and name the same finite route. No row in the carrier is a global confluence theorem, full parallel diamond, or unchecked subject-reduction proof. □

Theorem 2212.4 (MetaCIC critical-pair subject-reduction separation)**.** *Let $B = (R, K, J, E, D, S, U, H, C, P, N)$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Subject-reduction information can be exposed from B only through the displayed row S. The row S is a boundary classifier: it keeps typing-discharge routes outside the critical-pair table K and the common-reduct table J. Therefore no confluence audit consumer may treat critical-pair joins as subject-reduction discharge.*

Proof. Project the carrier of Definition 2212.1. The projection displays the route-classifier row S separately from the redex-frontier row R, the critical-pair table K, and the common-reduct table J.

Apply Definition 2212.2. Its row inventory names subject-reduction separation as an obligation attached to S, while the critical-pair display and common-reduct display obligations are attached to K and J.

Now apply Definition 2212.3. The confluence audit route reads R, K, J before it reaches E, D and then S, U, so the typing-discharge boundary is downstream of the common-reduct rows rather than hidden inside them.

Thus a consumer may inspect subject-reduction information only by reading S as the displayed boundary classifier. The finite critical-pair and common-reduct rows remain confluence-boundary data, not subject-reduction discharge. □

Theorem 2212.5 (MetaCIC confluence critical-pair local join)**.** *Let $B = (R, K, J, E, D, S, U, H, C, P, N)$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Any displayed critical pair listed by K may be consumed as joinable only through a corresponding common-reduct row in J, then*

through the bounded normal-equality checker row E or the closed-normal fragment row D. It may not be consumed as arbitrary closed-term Church–Rosser evidence, full parallel diamond evidence, or residual-substitution compatibility.

Proof. Project the carrier of Definition 2212.1. The critical-pair table is the row K, and the only row that can name a join candidate for an entry of K is the common-reduct table J.

Apply Definition 2212.3. It forces every confluence audit read through R, K, J before the bounded checker E or closed-normal fragment D is consumed.

The rows S and U record the non-claim separation. They keep subject-reduction discharge, arbitrary closed-term Church–Rosser, full parallel diamond, and residual-substitution compatibility outside the local critical-pair packet, while H, C, P, N only transport, replay, source, and name the displayed finite route. □

Theorem 2212.6 (MetaCIC critical-pair local-join coverage)**.** *Let $B = (R, K, J, E, D, S, U, H, C, P, N)$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Every critical-pair row accepted in K and consumed as locally joinable has its corresponding join candidate displayed in J. Verification of that candidate factors through either the bounded normal-equality row E or the closed-normal fragment row D before the subject-reduction separation row S can be read.*

Proof. Project the carrier of Definition 2212.1. The projection separates the critical-pair table K, the common-reduct table J, the bounded normal-equality checker E, the closed-normal fragment D, and the subject-reduction separation row S.

Apply Definition 2212.2 and Definition 2212.3. The obligations attach critical-pair display to K and common-reduct display to J, while the common-reduct boundary forces the verification route through E or D before any downstream boundary row is consumed.

Finally apply Definition 2212.4 and Definition 2212.5. Local joinability is therefore a finite K, J, E, D coverage claim, with S read only as the later subject-reduction boundary and not as a hidden proof of global confluence. □

Theorem 2212.7 (MetaCIC critical-pair residual-join boundary)**.** *Let $B = (R, K, J, E, D, S, U, H, C, P, N)$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Residual-style join evidence can be exposed from B only through the displayed route*

$$K, J, E, D, S, U, H, C, P, N.$$

The route records finite critical-pair rows, common-reduct rows, bounded normal-equality or closed-normal checks, subject-reduction separation, boundary data, transport, replay, provenance, and local naming. It does not supply residual-substitution compatibility, a global parallel diamond, or an unchecked confluence theorem.

Proof. Project the carrier of Definition 2212.1. The critical-pair table K and common-reduct table J are the only rows from which a residual-style join read can begin.

Use the common-reduct boundary of Definition 2212.3. It forces a consumer to read the common-reduct table before any bounded normal-equality row E or closed-normal fragment row D is used.

Apply the subject-reduction separation theorem Definition 2212.4. The row S keeps typing-discharge routes outside the critical-pair and common-reduct tables, so residual-style join evidence cannot be treated as subject-reduction discharge.

Finally apply the local-join coverage theorem Definition 2212.6. Local joinability is exhausted by the finite K, J, E, D coverage route, and the rows U, H, C, P, N only record the boundary,

transport, replay, source, and name. Residual-substitution compatibility and a global parallel diamond are therefore outside the accepted packet. □

Theorem 2212.8 (MetaCIC critical-pair residual-substitution obstruction route)**.** *Let $B = (R, K, J, E, D, S, U, H,$ be an accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carrier. Any residual-substitution-shaped consumer of B must route through*

$$K, J, E, D, S, U, H, C, P, N.$$

It may read residual-style data only as an obstruction boundary for the local critical-pair packet, not as residual-substitution compatibility, global parallel diamond evidence, or full confluence.

Proof. Project the carrier of Definition 2212.1. The projection exposes the critical-pair table K, common-reduct table J, bounded checker E, closed-normal fragment D, subject-reduction boundary S, non-claim boundary U, and structural rows H, C, P, N.

Use the common-reduct boundary of Definition 2212.3. Any consumer that starts from a critical-pair row must pass through J before it may consult E or D.

Apply Definition 2212.4 and Definition 2212.6. These results keep subject-reduction discharge separated at S and exhaust local joinability by the finite K, J, E, D coverage route.

Finally apply Definition 2212.7. The residual read is already classified as boundary data, and no carrier row supplies residual-substitution compatibility or a global parallel diamond. Therefore a residual-substitution-shaped consumer can only read an obstruction boundary through $K, J, E, D, S, U, H, C, P, N$. □

Theorem 2212.9 (MetaCIC confluence critical-pair field-faithful prediction)**.** *Let $B = (R, K, J, E, D, S, U, H, C, P$ and $B' = (R', K', J', E', D', S', U', H', C', P', N')$ be accepted* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *carriers. If their displayed field projections agree componentwise on $R, K, J, E, D, S, U, H, C, P, N$, then they expose the same critical-pair and common-reduct boundary route. In particular, the critical-pair table K and common-reduct table J cannot be erased or replaced by aliases of $R, E, D, S, U, H, C, P, N$ without changing the public packet classifier.*

Proof. Project both carriers through Definition 2212.1. Componentwise equality of the displayed fields identifies the same redex-frontier row, the same critical-pair table, the same common-reduct table, the same bounded and normal checker rows, and the same boundary classifier route.

Apply Definition 2212.3. The confluence audit route must read R, K, J before it reaches E, D, S, U, so agreement of all displayed projections fixes the same critical-pair and common-reduct boundary route.

Finally apply Definition 2212.5. A displayed critical pair is joinable only through its corresponding row in J. Erasing K or J would remove a field used by the local join route, so the public classifier changes rather than preserving a field-faithful packet. □

Theorem 2212.10 (MetaCIC confluence critical-pair nontrivial witness surface)**.** *There are two classifier-separated* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *packet surfaces: one whose K, J boundary rows are empty and whose U row records that no displayed critical-pair join is available, and one with a displayed critical-pair row in K together with a corresponding common-reduct row in J. The separation is witnessed by the K, J, U fields and is visible to the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}}$ *classifier.*

Proof. Read the carrier definition. The fields K and J are independent displayed rows: K lists inspected critical pairs, and J lists the corresponding common-reduct candidates.

For the empty surface, the classifier reads empty K, J rows together with the boundary row U that denies a displayed local join route. For the inhabited surface, the classifier reads one critical-pair entry in K and the matching common-reduct entry in J before U records the bounded boundary route.

Apply Definition 2212.3 and Definition 2212.5. The local join claim cannot be consumed without the displayed K, J rows, so the empty and inhabited surfaces are separated by K, J, U rather than by an external confluence principle. □

The Lean obligation for this packet is to provide `Nontrivial MetaCICConfluenceCriticalPairBoundaryUp` and `FieldFaithful MetaCICConfluenceCriticalPairBoundaryUp` TasteGate instances when the carrier is formalized. Those instances must check that the critical-pair table and common-reduct table are genuine fields of the carrier, not aliases for the global confluence boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2212.1, Definition 2212.2, and Definition 2212.3 seed the finite MetaCIC critical-pair and common-reduct boundary over redex-frontier, normal-fragment, bounded-checker, route-classifier, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, subject-reduction discharge, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, critical-pair display, common-reduct display, bounded and normal handoff, subject-reduction separation, ledger non-escape, and TasteGate nontriviality and field faithfulness for BEDC.Derived.MetaCICConfluenceCriticalPairBoundaryUp.

Constructive story: A $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ packet is generated from a MetaCICRedexFrontier row, a finite critical-pair table, a common-reduct table, BoundedNormalEqualityChecker and MetaCICConfluenceNormalFragment rows, a SubjectReductionRouteClassifier separation row, a boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2213

A Concrete Naming Certificate for $\mathsf{GeodesicMetric}^{\uparrow}$

Following Burago–Burago–Ivanov and Bridson–Haefliger on geodesic metric spaces, the $\mathsf{GeodesicMetric}^{\uparrow}$ packet records a displayed finite segment witness inside a metric carrier. It sits over the metric source of chapter 57, reads closed-ball membership through chapter 1451, and remains adjacent to the completion and compactness frontiers of chapter 128, **??**, and chapter 2035. The packet names a finite segment handoff for later compactness and completion consumers; it is not a host path space, not a selected geodesic object, not a quotient path-equality principle, and not an ambient completeness assertion.

Definition 2213.1 (GeodesicMetric carrier)**.** *A* $\mathsf{GeodesicMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, E, T, S, A, C, R, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, displayed endpoint-distance request, and the metric comparison rows used by every segment read.*
2. *E records the two endpoint histories over M, together with the endpoint classifier rows used to transport the segment request.*
3. *T is the finite segment-parameter row, a bounded rational or dyadic address ledger for the visible subsegment reads.*
4. *S is the displayed segment witness row: for the endpoint pair in E, it lists the finite intermediate point reads and the source-distance observations through M.*
5. *A is the distance-additivity or midpoint-budget ledger. It records either that the displayed subsegment distances add to the endpoint distance within the visible tolerance budget, or that the displayed midpoint windows split the endpoint budget into the two certified subreads requested by the consumer.*
6. *C records componentwise* hsame *transport for the metric row, endpoint rows, parameter row, segment witness, additivity or midpoint ledger, replay, provenance, and local naming rows.*
7. *R is the displayed* Cont *replay row, requiring every segment read to be regenerated from M, E, T, S, A, C before any consumer may inspect it.*

8. *P is the* Pkg *provenance over* $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, T, S, A, C, R, P, N$. It has no coordinate for host path spaces, selected geodesics, quotient path equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.GeodesicMetricUp.GeodesicMetricTasteGate single carrier alignment`

Theorem 2213.2 (GeodesicMetric NameCert obligations)**.** *For an accepted* $\mathsf{GeodesicMetric}^{\uparrow}$ *carrier*

$$G = (M, E, T, S, A, C, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint compatibility over the displayed* $\mathsf{Met}^{\uparrow}$ *row, finite parameter admission, segment-witness exactness, additivity or midpoint-budget exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 2213.1. No obligation row asserts a host path object, a globally selected geodesic, quotient equality of paths, or completeness of the ambient metric space.*

Proof. Carrier admission is projection from Definition 2213.1. The accepted packet displays the metric source, the two endpoint rows, the finite parameter ledger, the segment witness, and the additivity or midpoint-budget row before the transport and naming rows are read.

Endpoint compatibility is metric-local. The row E may be transported only along the point classifier exposed by M, so every endpoint-distance request remains inside the same $\mathsf{Met}^{\uparrow}$ source.

Segment exactness is finite. The rows T and S list only the visible parameter windows, intermediate point reads, and distance observations required by the current witness. The row A then records the visible additivity or midpoint-budget comparison through those same distance observations.

Transport and replay are structural. Componentwise hsame in C preserves the metric source, endpoints, parameter row, segment witness, and budget ledger, while R, P, and N replay and name exactly that finite route.

The excluded objects are absent from the carrier inventory. Since no row stores host paths, selected geodesics, quotient path equality, or ambient completeness, no local NameCert obligation can export them. □

Theorem 2213.3 (GeodesicMetric segment handoff)**.** *Every completion-facing, compactness-facing, or closed-ball-facing read of an accepted* $\mathsf{GeodesicMetric}^{\uparrow}$ *carrier factors through the finite route*

$$M, E, T, S, A \longmapsto C, R, P, N.$$

Thus a later $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, *or* $\mathsf{DoublingMetric}^{\uparrow}$ *consumer receives only the displayed metric row, endpoint rows, finite segment-parameter row, segment witness, additivity or midpoint-budget ledger, componentwise transport, replay, provenance, and local naming rows. The handoff does not import host path spaces, selected geodesics, quotient path equality, or ambient completeness.*

Proof. Project the accepted carrier. The metric source is M, and every endpoint or closed-ball query first reads the endpoint rows in E through that source.

The segment data are T, S, A. The parameter row T bounds the visible requests, S supplies the displayed intermediate point and distance observations, and A exposes the finite additivity or midpoint-budget comparison that a consumer may use.

The remaining rows do not enlarge the analytic object. The transport row C preserves the same coordinates componentwise, and the replay, provenance, and naming rows R, P, N require the consumer to regenerate the same finite route before accepting the segment handoff.

Therefore completion, properness, doubling, and closed-ball consumers can use the displayed witness only through those rows. The carrier contains no path-space object, selected geodesic, path quotient, or completeness row from which a stronger conclusion could be read. □

Theorem 2213.4 (GeodesicMetric path-length handoff)**.** *Every* $\mathsf{LengthSpace}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{GeodesicMetric}^{\uparrow}$ *carrier*

$$G = (M, E, T, S, A, C, R, P, N)$$

must read the $\mathsf{Met}^{\uparrow}$ *source, endpoint rows, finite segment-parameter row, displayed segment witness, additivity or midpoint-budget ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$ *row before treating the segment as a path-length witness. The handoff reaches chapter 2082 only through those displayed rows and does not import host path spaces, quotient path equality, selected geodesics, or ambient completeness.*

Proof. Project the accepted carrier. Its metric source is M, its endpoint data are E, and its finite segment route is the row sequence T, S, A.

Apply Definition 2213.3. The segment handoff already forces every consumer to read the finite parameter row, displayed segment witness, additivity or midpoint-budget ledger, and the structural rows C, R, P, N before using the witness.

The $\mathsf{LengthSpace}^{\uparrow}$ consumer of chapter 2082 therefore receives a path-length witness only as this finite metric segment route. Since the carrier stores no host path space, path quotient, selected global geodesic, or ambient completeness row, none of those stronger objects can enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GeodesicMetric}^{\uparrow}$ packet is generated from a MetricUp source, endpoint rows, a finite segment-parameter row, a displayed segment witness, an additivity or midpoint-budget ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$, seed)

Scope closed: Definition 2213.1, Definition 2213.2, and Definition 2213.3 seed the finite segment-witness surface for later metric closed-ball, completion, properness, and doubling consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GeodesicMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host path spaces, selected geodesics, quotient path equality, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint compatibility, finite parameter admission, segment-witness exactness, additivity or midpoint-budget exactness, transport stability, replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.GeodesicMetricUp.

Chapter 2214

A Concrete Naming Certificate for CauchySeriesProduct$^{\uparrow}$

Following Hardy and Knopp on products of series, the CauchySeriesProduct$^{\uparrow}$ packet records the constructive product route by which two accepted real-series source rows expose a Cauchy product only through finite convolution windows, tail budgets, regular rational readback, and a real seal. It consumes the finite partial-sum discipline of RealSeries$^{\uparrow}$ in chapter 1581, the convolution ledger of CauchyConvolution$^{\uparrow}$ in chapter 409, the fast product surface of FastSeriesProduct$^{\uparrow}$ in chapter 368, the absolute convergence route of AbsoluteConvergence$^{\uparrow}$ in chapter 1700, and the majorant tail accounting of CauchyMajorantSeries$^{\uparrow}$ in chapter 1774. The packet names the finite certificate by which a product series is admitted; it does not import a host theorem on Cauchy products, selected infinite sums, quotient equality of series, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 2214.1 (Cauchy series product carrier)**.** *A* CauchySeriesProduct$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, C, B, M, R, E, H, P, N)$$

with the following displayed rows:

1. *S and T are the two* RealSeries$^{\uparrow}$ *source rows. They expose only inspected term windows, finite partial sums, and source tail requests already admitted by the real-series packet.*
2. *C is the finite* CauchyConvolution$^{\uparrow}$ *window row. It records the inspected convolution diagonals, product-prefix windows, and finite coefficient multiplications used by the current request.*
3. *B is the product-budget row. It is supplied either by the* AbsoluteConvergence$^{\uparrow}$ *route or by the* CauchyMajorantSeries$^{\uparrow}$ *route, and it records the displayed tail bounds needed to control the omitted convolution terms.*
4. *M is the* FastSeriesProduct$^{\uparrow}$ *handoff row. It repacks the displayed source windows and the convolution row as the finite product surface consumed by downstream readers.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed product partial sums and for every source-window read used by C and M.*
6. *E is the* Real$^{\uparrow}$ *seal reached only after S, T, C, B, M, R have been displayed.*
7. *H stores componentwise* hsame *transport for the two source rows, convolution windows, product budgets, fast-product handoff, regular readback, real seal, provenance, and local naming row.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchySeriesProduct}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyConvolution}^{\uparrow}$, $\mathsf{FastSeriesProduct}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{CauchyMajorantSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySeriesProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, C, B, M, R, E, H, P, N$. It has no coordinate for a host Cauchy product theorem, an unbounded convolution sum, a selected real limit, quotient equality of series, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySeriesProductUp.TasteGate.CauchySeriesProductTasteGate single carrier alignment`

Theorem 2214.2 (Cauchy series product NameCert obligations)**.** *For every accepted* $\mathsf{CauchySeriesProduct}^{\uparrow}$ *carrier $K = (S, T, C, B, M, R, E, H, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySeriesProduct}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed* $\mathsf{RealSeries}^{\uparrow}$ *source rows, finite convolution windows, product budgets, fast-product handoff, regular readback, real seal, transport, provenance, and local naming row;*
2. *source-window exactness, requiring every product term inspected by C to be read from the displayed finite windows of S and T;*
3. *convolution-window exactness, requiring every product-prefix row to pass through the finite* $\mathsf{CauchyConvolution}^{\uparrow}$ *row C before the fast-product handoff is exposed;*
4. *tail-budget exactness, requiring each omitted convolution contribution to be bounded by the displayed* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *or* $\mathsf{CauchyMajorantSeries}^{\uparrow}$ *budget row B;*
5. *regular readback, requiring every product partial-sum read to pass through R after C, B, M have been displayed;*
6. *Real-seal nonescape, requiring the endpoint row E to remain downstream of the finite readback and to export no host product theorem, selected infinite sum, quotient equality, or ambient completeness;*
7. *provenance and naming preservation, requiring H, P, N to preserve the same finite route under transport, package sourcing, and local certificate comparison.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 2214.1. The two source rows S, T expose only the finite real-series windows and partial-sum data named by the source packets, so the product certificate begins with displayed series data rather than a host infinite sequence.

The convolution and product rows are then forced. The row C records the finite Cauchy-convolution windows, and M repacks those windows as the fast product handoff. Every inspected product term is therefore read through S, T, C, M.

The tail budget is separate but still finite. The row B supplies either an absolute-convergence budget or a majorant-series budget, so omitted convolution terms are controlled by displayed tail rows before any regular readback is exported.

Finally the readback and seal close the route. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the product partial sums, the $\mathsf{Real}^{\uparrow}$ seal E is downstream of that readback, and H, P, N transport, source, and name the same packet. The listed rows therefore exhaust the local NameCert surface. □

Theorem 2214.3 (Cauchy series product RealSeries handoff)**.** *Every endpoint-facing read from an accepted* $\mathsf{CauchySeriesProduct}^{\uparrow}$ *carrier factors through the two source* $\mathsf{RealSeries}^{\uparrow}$ *rows* S, T*, the finite* $\mathsf{CauchyConvolution}^{\uparrow}$ *window row* C*, the displayed absolute or majorant tail budget* B*, the* $\mathsf{FastSeriesProduct}^{\uparrow}$ *handoff row* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural rows* H, P, N*. A consumer receives no host multiplication theorem for series, no selected infinite convolution sum, no quotient equality of term streams, no countable-choice selector, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with the source rows. The carrier admits only the two displayed $\mathsf{RealSeries}^{\uparrow}$ packets, so every product read first chooses the finite source windows visible in S and T.

Next pass through the product route. The convolution row C builds the inspected Cauchy-product windows, the budget row B controls the omitted tails by absolute or majorant evidence, and the fast-product row M records the finite handoff visible to downstream product consumers.

Finally read the endpoint. The row R supplies the regular rational readback of the displayed product partial sums, and E is the only real-facing seal. Since the structural rows H, P, N preserve this same route, the handoff cannot expose any unlisted analytic object. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySeriesProduct}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySeriesProduct}^{\uparrow}$ packet is generated from two RealSeries source rows, finite convolution windows, an absolute or majorant tail budget, a FastSeriesProduct handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySeriesProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2214.1, Definition 2214.2, and Definition 2214.3 seed the finite constructive product-of-series certificate from two RealSeries source rows through convolution windows, tail budgets, regular readback, and RealUp endpoint sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySeriesProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Cauchy product theorems, selected infinite convolution sums, quotient equality of series, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window exactness, convolution-window exactness, tail-budget exactness, regular readback, Real-seal nonescape, provenance preservation, and TasteGate field faithfulness for BEDC.Derived.CauchySeriesProductUp.

Chapter 2215

A Concrete Naming Certificate for RegulatedUniformApproximation$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the RegulatedUniformApproximation$^{\uparrow}$ packet separates the finite step-schedule data used to approximate a regulated function uniformly from the later integration interface. It sits between the regulated-function surface of chapter 1554, the dyadic step-function rows of chapter 428, the uniform-limit route of chapter 463, the closed interval domain of chapter 1448, and the terminal real seal of chapter 38. Its role is to expose a finite approximation route, not to assert integration, host function-space equality, or ambient completeness.

Definition 2215.1 (Regulated uniform approximation carrier)**.** *A* RegulatedUniformApproximation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, A, T, U, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* ClosedUnitInterval$^{\uparrow}$ *domain row, including the interval endpoints and order data consumed by the approximation schedule;*
2. *A is the finite* DyadicStepFunction$^{\uparrow}$ *step ledger, recording the displayed step partitions, carried values, and dyadic tolerances;*
3. *T is the uniform tail schedule, naming the finite threshold and tolerance comparisons used to read the approximation as a uniform route;*
4. *U is the* UniformLimit$^{\uparrow}$ *handoff row that consumes A and T before any later regulated-function or integration-facing consumer sees the packet;*
5. *E is the* Real$^{\uparrow}$ *seal row read through the regular rational windows and* RegSeqRat$^{\uparrow}$ *readback data attached to the displayed step values;*
6. *H stores componentwise* hsame *transports for the interval, step ledger, uniform schedule, uniform-limit route, real seal, provenance, and local name;*
7. *C stores the displayed* Cont *replay routes from a domain request through the selected step row, the uniform schedule, the handoff row, and the real seal;*
8. *P is the* Pkg *provenance over* RegulatedUniformApproximation$^{\uparrow}$, RegulatedRealFunction$^{\uparrow}$, DyadicStepFunction$^{\uparrow}$, UniformLimit$^{\uparrow}$, ClosedUnitInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedUniformApproximation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a Darboux sum, Riemann sum, regulated integral, quotient of functions, open-cover compactness proof, selected real representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegulatedUniformApproximationUp.RegulatedUniformApproximationTasteGate`
`single carrier alignment`

Theorem 2215.2 (Regulated uniform approximation NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedUniformApproximation}^{\uparrow}$ *packet $R = (I, A, T, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedUniformApproximation}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed interval, step ledger, uniform schedule, handoff route, real seal, transport, replay, provenance, and local name;*
2. *step-ledger exactness, requiring every approximation read to use the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *rows in A over the* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *row I;*
3. *uniform approximation stability, requiring the tail schedule T to control all later step reads at the displayed tolerance;*
4. $\mathsf{UniformLimit}^{\uparrow}$ *handoff, requiring every regulated-function or integration-facing consumer to factor through U before reaching the real seal row E;*
5. *real-seal non-escape, requiring every terminal read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal data displayed by E.*

No local obligation is stored outside those five rows.

Proof. Carrier admission is projection from Definition 2215.1. The accepted packet exposes only the rows $I, A, T, U, E, H, C, P, N$.

The step-ledger row is governed by I and A. A domain read first enters the closed interval row and then selects one of the displayed dyadic step-function rows; there is no host function value outside this finite ledger.

The uniform side is governed by T and U. The tail schedule records the common tolerance route, and the uniform-limit handoff consumes that schedule before any downstream consumer reaches E.

The remaining rows transport, replay, source, and name the same packet. Since no coordinate stores integration, function quotients, compactness, selected representatives, or ambient completeness, no NameCert obligation can export them. □

Theorem 2215.3 (Regulated uniform approximation step-ledger exactness)**.** *Every approximation read of an accepted* $\mathsf{RegulatedUniformApproximation}^{\uparrow}$ *packet factors through the* $\mathsf{ClosedUnitInterval}^{\uparrow}$ *row I and one displayed* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row in the ledger A. The read obtains its partition, carried value, and dyadic tolerance data from that finite row, not from a host function-space representative.*

Proof. Read the carrier definition. The only approximation coordinate is A, and it is scoped by the interval row I.

The continuation row C sends a domain request to I before selecting a step row in A. Componentwise transport in H preserves the same interval and selected step row under classifier comparison.

Thus an approximation read has no route around I and A. Any value or tolerance not displayed by the selected dyadic step-function row is outside the accepted packet. □

Theorem 2215.4 (Regulated uniform approximation stability)**.** *If two accepted reads of a* RegulatedUniformApprox *packet use step rows past the threshold named by the uniform tail schedule T, then their comparison is routed through the same displayed tolerance schedule and the same* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row U. The stability claim is therefore a finite schedule claim over A, T, U, not an ambient completeness claim about all real-valued functions.*

Proof. Start with the schedule row T. It is the only coordinate that names common thresholds for the step ledger A.

The handoff row U consumes exactly that schedule as a uniform-limit route. Therefore any comparison of sufficiently late displayed steps must be replayed through T and then through U.

The carrier omits ambient function-space limits and completeness data. Stability is consequently confined to the displayed schedule and its uniform-limit handoff. □

Theorem 2215.5 (Regulated uniform approximation UniformLimit handoff)**.** *Every later* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *consumer, and every later integration-facing consumer that reads this packet through a regulated-function surface, must factor through*

$$I, A, T \longmapsto U \longmapsto E.$$

The handoff supplies the closed interval row, dyadic step ledger, uniform schedule, $\mathsf{UniformLimit}^{\uparrow}$ *route, and terminal* $\mathsf{Real}^{\uparrow}$ *seal, but it supplies no Darboux, Riemann, or regulated-integral row.*

Proof. Use step-ledger exactness to place every approximation read at I and A. Use uniform approximation stability to place every sufficiently late comparison at T and U.

The carrier then exposes E only after the handoff through U. Structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming.

Because no integration coordinate occurs in $R = (I, A, T, U, E, H, C, P, N)$, downstream integration chapters may consume this handoff as input data but cannot read integration as part of the approximation packet itself. □

Theorem 2215.6 (Regulated uniform approximation real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegulatedUniformApproximation}^{\uparrow}$ *packet reaches the terminal seal only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback data attached to E after the interval, step-ledger, uniform schedule, and* $\mathsf{UniformLimit}^{\uparrow}$ *rows have been read. It exports no selected real representative, quotient function equality, countable-choice schedule, integration row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier. The only terminal real-facing coordinate is E, and its provenance row records the regular rational readback and real seal dependency rather than a host real value.

The preceding theorems force every read of E to pass through I, A, T and U. The structural rows do not add analytic content; they transport, replay, source, and name the same finite packet.

Therefore any route requiring selected representatives, quotient equality, choice, integration, compactness, or ambient completeness has no coordinate in the accepted carrier and is rejected at the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedUniformApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedUniformApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2215.1, Definition 2215.2, Definition 2215.3, Definition 2215.4, Definition 2215.5, and Definition 2215.6 seed the finite step-schedule approximation surface over interval, dyadic step, uniform tail, uniform-limit, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedUniformApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Darboux integration, Riemann integration, regulated integrals, host function-space equality, open-cover compactness, selected real representatives, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, step-ledger exactness, uniform approximation stability, UniformLimit handoff, Real-seal non-escape, and field faithfulness for the same packet.

Constructive story: A RegulatedUniformApproximation$^{\uparrow}$ packet is generated from a ClosedUnitInterval domain row, a finite DyadicStepFunction step ledger, a uniform tail schedule, a UniformLimit route, RegSeqRat readback and Real seal data, componentwise transport, displayed continuation replay, package provenance, and local naming.

Chapter 2216

A Concrete Naming Certificate for $\mathsf{CauchyLocatedness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyLocatedness}^{\uparrow}$ packet records locatedness as finite witness data for regular Cauchy real names. It sits over the regular rational source of chapter 301, the finite stream-window schedule of chapter 126, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the quotient-respecting comparison discipline of chapter 1820. The packet exposes locatedness witnesses without replacing them by Dedekind cuts, host real equality, selected limits, countable choice, trichotomy, or ambient completeness.

Definition 2216.1 (Cauchy locatedness carrier)**.** *A* $\mathsf{CauchyLocatedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, W, D, Q, V, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular Cauchy source row, read as a* $\mathsf{RegSeqRat}^{\uparrow}$ *name before any* $\mathsf{Real}^{\uparrow}$ *consumer receives a sealed real-name boundary.*
2. *W is the finite observation window row, a* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the inspected Cauchy stages, rational intervals, and comparison requests.*
3. *D is the dyadic tolerance request row, carried by* $\mathsf{DyadicRatCore}^{\uparrow}$ *and recording the requested scale, positive margin, endpoint enclosure, and exhausted window ledger.*
4. *Q is the Cauchy-equivalence comparison row, tying the witness to the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *surface so equivalent regular Cauchy names replay the same finite locatedness route.*
5. *V is the displayed located witness row. For each requested dyadic tolerance and rational test window, it records the finite inside-or-apart evidence available from R, W, D, Q.*
6. *H records componentwise* hsame *transport for the regular source, stream windows, dyadic request, equivalence comparison, located witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from R, W, D into Q, from Q into the witness row V, and from V into the terminal* $\mathsf{Real}^{\uparrow}$ *readback boundary.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyLocatedness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $R, W, D, Q, V, H, C, P, N$. It has no coordinate for Dedekind cuts, host real equality, selected limits, countable choice, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyLocatednessUp.CauchyLocatednessTasteGate single carrier alignment`

Theorem 2216.2 (Cauchy locatedness window stability)**.** *Let $L = (R, W, D, Q, V, H, C, P, N)$ be an accepted* $\mathsf{CauchyLocatedness}^\uparrow$ *carrier. If a consumer refines the displayed* $\mathsf{StreamName}^\uparrow$ *observation window while preserving the same dyadic tolerance request and Cauchy-equivalence comparison row, then the located witness row replays through H and C to the same local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^\uparrow}$ *surface. The refined read exposes only the regular Cauchy source, finite windows, dyadic request, equivalence comparison, displayed located witness, and structural rows.*

Proof. Project the accepted carrier. The only rows available to a window-refining consumer are the regular source R, stream-window schedule W, dyadic request D, equivalence comparison Q, located witness V, and structural rows H, C, P, N.

Refining the observation window changes only which finite prefix of W is inspected. The dyadic request row D and equivalence comparison row Q remain the same request data, so the witness row V is read by replaying the displayed route rather than by selecting a new real representative.

Componentwise hsame in H transports the source, window, dyadic, comparison, and witness rows. The Cont row C then replays the route from R, W, D into Q and from Q into V before any terminal $\mathsf{Real}^\uparrow$ read is exposed.

Thus the refined read has the same local NameCert surface. Since no carrier coordinate stores host equality, Dedekind cuts, selected limits, countable choice, trichotomy, or completeness, window refinement cannot introduce any of those objects. □

Theorem 2216.3 (Cauchy locatedness NameCert obligation)**.** *For every accepted* $\mathsf{CauchyLocatedness}^\uparrow$ *carrier $L = (R, W, D, Q, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the regular Cauchy source row, finite observation windows, dyadic tolerance request, Cauchy-equivalence comparison row, located witness row, transport, replay, provenance, and local naming row;*
2. *witness exactness, requiring V to read only the displayed R, W, D, Q route and to return finite inside-or-apart evidence for the requested dyadic tolerance;*
3. *equivalence stability, requiring Q to keep equivalent regular Cauchy names on the same locatedness route before the terminal* $\mathsf{Real}^\uparrow$ *boundary is read;*
4. *window stability, requiring refinements of W to replay through H and C as in Definition 2216.2;*
5. *ledger exhaustion, excluding Dedekind cuts, host real equality, selected limits, countable choice, trichotomy, and ambient completeness.*

No locatedness obligation is stored outside these rows.

Proof. Project the carrier L. The projection exposes exactly the regular Cauchy source, finite window schedule, dyadic request ledger, equivalence comparison, located witness, componentwise transport, continuation replay, provenance, and local naming row.

The witness obligation is finite. The row V consumes the displayed source, windows, dyadic tolerance, and equivalence comparison, then records the inside-or-apart evidence visible at that request.

The equivalence obligation is carried by Q. It prevents a consumer from changing the locatedness witness by switching to an equivalent regular Cauchy name outside the displayed comparison route.

The stability obligation is Definition 2216.2. Window refinements replay through the same transport and continuation rows, so the local NameCert surface remains the one named by N.

Finally, P records only the listed sibling certificates and N names only this packet. Since the carrier has no other coordinate, the five obligations exhaust the local $\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLocatedness}^{\uparrow}$ packet is generated from a regular Cauchy source row, finite StreamName observation windows, a DyadicRatCore tolerance request, a CauchyEquivalenceSetoid comparison row, a displayed located witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2216.1, Definition 2216.2, and Definition 2216.3 seed the finite locatedness witness packet for regular Cauchy real names over RegSeqRat, StreamName, DyadicRatCore, RealUp, and CauchyEquivalenceSetoid rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocatedness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, host real equality, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean declaration for BEDC.Derived.CauchyLocatednessUp covering the nine-row carrier, window stability, and local NameCert obligations.

Chapter 2217

A Concrete Naming Certificate for $\mathsf{SobolevEmbedding}^{\uparrow}$

Following Adams and Fournier on Sobolev spaces, the $\mathsf{SobolevEmbedding}^{\uparrow}$ packet records the finite BEDC naming surface for an embedding assertion read from a Sobolev source certificate into a normed, Banach, Hilbert, metric, or compact-metric target certificate. It is a recognised analysis neighbour of chapter 440, chapter 89, chapter 90, chapter 91, chapter 57, chapter 129, and chapter 38. The chapter does not import a host function space; it names the displayed dependency rows that later PDE and functional-analysis certificates can cite.

Definition 2217.1 (Sobolev embedding carrier)**.** *A* $\mathsf{SobolevEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, N, E, T, R, L, P)$$

with the following displayed rows:

1. *D records the domain regularity and scalar* $\mathsf{Real}^{\uparrow}$ *boundary on which the source observations are read;*
2. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, including the visible weak-derivative, integrability, and base-metric rows supplied by chapter 440;*
3. *N is the target norm row, read through* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Banach}^{\uparrow}$*, or* $\mathsf{Hilbert}^{\uparrow}$ *according to the named target surface;*
4. *E records the exponent and regularity comparison rows that license the displayed embedding estimate;*
5. *T is the target readback row, which may point to normed, Banach, Hilbert, metric, or compact-metric evidence but never to a hidden host function;*
6. *R stores the componentwise* hsame *transport for the domain, Sobolev source, norm target, exponent rows, target readback, ledger, and provenance rows;*
7. *L is the finite ledger stating the norm-bound budget and the source-to-target observations consumed by the embedding read;*
8. *P is the* Pkg *provenance over* $\mathsf{SobolevEmbedding}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, together with the local naming row.*

The carrier has no coordinate for a host function representative, almost-everywhere quotient equality, Rellich compactness, full PDE regularity, open-cover compactness, countable choice, or a selected subsequence.

Lean4 ✓ `BEDC.Derived.SobolevEmbeddingUp.SobolevEmbeddingTasteGate single carrier alignment`

Theorem 2217.2 (Sobolev embedding NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{SobolevEmbedding}^{\uparrow}}$ *surface for the carrier of Definition 2217.1 consists exactly of the following finite obligations:*

1. *carrier admission: an accepted packet exposes only* $Q = (D, S, N, E, T, R, L, P)$ *and admits no hidden analytic coordinate;*
2. *classifier stability: two accepted packets compare only through componentwise* hsame *transport of the domain, Sobolev source, norm target, exponent, target readback, ledger, provenance, and naming rows;*
3. *derivative-row transport: the weak-derivative observations inherited from* S *remain attached to the same embedding read whenever* R *transports the displayed rows;*
4. *norm-bound ledger preservation: the estimate recorded in* L *is preserved under the same displayed transport and cannot be strengthened by an unrecorded bound;*
5. *target readback: every target observation factors through* T *and through the named* $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{CompactMetric}^{\uparrow}$ *surface;*
6. *ledger non-escape: no consumer can recover Rellich compactness, full PDE regularity, open-cover compactness, host function equality, choice, or a selected representative from the packet.*

Proof. Carrier admission is projection from the eight displayed rows of Q. The source and target data are visible certificates, and the local packet contains no row for host functions or external compactness.

For classifier stability, compare two packets row by row using the stored hsame routes in R. The Sobolev source, norm target, exponent rows, target readback, ledger, provenance, and local naming row are transported componentwise.

Derivative-row transport and norm-bound preservation are ledger facts. The derivative observations are those already exposed by S, while the bound is the row stored in L; the replay can reread those rows but cannot create an additional derivative or a sharper estimate.

Finally, target readback is forced through T and the named neighbouring certificates. Since the carrier has no coordinate for the excluded analytic principles, ledger exhaustion gives the non-escape boundary. □

Theorem 2217.3 (Sobolev embedding finite-energy localization)**.** *Every accepted* $\mathsf{SobolevEmbedding}^{\uparrow}$ *packet exposes only the displayed local Sobolev, norm, metric, exponent, transport, and ledger data in* D, S, N, E, T, R, L, P*. The finite source rows may localize an energy or derivative budget to the named embedding read, but they do not assert Rellich compactness, full PDE regularity, open-cover compactness, host function equality, countable choice, or a selected representative.*

Proof. Inspect a downstream consumer of the embedding packet. It can read the domain row D, the Sobolev source row S, the norm target row N, the exponent and regularity rows E, the target readback T, the transport ledger R, the norm-bound ledger L, and the provenance row P.

This is enough to localize the finite energy or weak-derivative budget named by the source rows. It is not enough to produce compactness, regularity, host equality, choice, or representatives, because none of those objects appears as a row of the carrier or as an output of the componentwise transport. □

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{SobolevEmbedding}^{\uparrow}$ packet is generated from a domain-regularity row, a $\mathsf{Sobolev}^{\uparrow}$ source row, a displayed norm or functional-analysis target row, exponent and regularity comparison rows, target readback, componentwise hsame transport, a finite norm-bound ledger, provenance, and local naming evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobolevEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2217.1, Definition 2217.2, and Definition 2217.3 seed the finite embedding-read surface over Sobolev, Norm, Banach, Hilbert, Metric, CompactMetric, and Real neighbours.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobolevEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Rellich compactness, full PDE regularity, open-cover compactness, host function equality, countable choice, selected representatives, or a Lean theorem.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the same displayed Sobolev embedding packet.

Chapter 2218

A Concrete Naming Certificate for $\mathsf{RellichKondrachov}^{\uparrow}$

Following Rellich and Kondrachov, in the form used by Adams–Fournier for Sobolev spaces, the $\mathsf{RellichKondrachov}^{\uparrow}$ packet records the finite compact-embedding route from a bounded Sobolev window into a compact metric target window. It sits after the weak-derivative and Sobolev embedding surfaces of chapter 2210 and chapter 2217, and it cites the Sobolev carrier of chapter 440, the compact and complete metric witnesses of chapter 129 and chapter 128, and the scalar readback of chapter 38. The chapter names only the finite subsequence-window ledger and the compactness route visible to BEDC consumers.

Definition 2218.1 (Rellich–Kondrachov carrier)**.** *A* $\mathsf{RellichKondrachov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, W, S, E, C, M, T, L, H, P, N)$$

with the following displayed rows:

1. *D records the bounded domain, boundary regularity, scalar* $\mathsf{Real}^{\uparrow}$ *readback, and finite coordinate windows in which the compact-embedding statement is inspected.*
2. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *row, read through chapter 2210, that fixes the weak-action observations used by the Sobolev source.*
3. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row from chapter 440, including the derivative-order, integrability, and boundedness ledger visible to the compactness route.*
4. *E is the* $\mathsf{SobolevEmbedding}^{\uparrow}$ *comparison row from chapter 2217; it records the exponent inequality and the embedding read without asserting compactness by itself.*
5. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *target row from chapter 129, restricted to the finite sequential compactness windows named by this packet.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-readback row from chapter 128, used only to read the displayed limit-window tests.*
7. *T stores the finite tolerance scale and subsequence-window schedule, listing for each requested observation window the bounded Sobolev source window, the target metric tolerance, and the compact-metric witness window.*
8. *L is the compact-embedding ledger. It records that the route*

$$W, S, E, D \longmapsto C, M, T$$

is the only available path from bounded Sobolev data to compact target observations.

9. *H stores componentwise* hsame *transport for $D, W, S, E, C, M, T, L, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RellichKondrachov}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{SobolevEmbedding}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RellichKondrachov}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected subsequence, open-cover compactness, quotient function space, host function representative, countable choice, full PDE regularity, trace theory, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the displayed real readbacks.*

Lean4 ✓ `BEDC.Derived.RellichKondrachovUp.RellichKondrachovTasteGate single carrier alignment`

Theorem 2218.2 (Rellich–Kondrachov NameCert obligations)**.** *For every accepted* $\mathsf{RellichKondrachov}^{\uparrow}$ *carrier $R = (D, W, S, E, C, M, T, L, H, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RellichKondrachov}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, weak-derivative access through W, Sobolev source access through S, embedding comparison access through E, compact-metric target access through C, complete-metric Cauchy readback through M, finite subsequence-window scheduling through T, compact-route ledger preservation through L, componentwise* hsame *transport through H, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from the displayed packet of Definition 2218.1. The analytic inputs are only the weak-derivative, Sobolev, embedding, compact-metric, complete-metric, and real readback rows named in $D, W, S, E, C, M, T, L, H, P, N$.

The obligation surface is therefore exhausted row by row. The weak action is read through W, the Sobolev boundedness data through S, the exponent and embedding comparison through E, and the compact target windows through C and M.

The subsequence evidence is not a chosen subsequence. It is the finite schedule in T together with the ledger L, so a consumer may replay a requested compactness window but cannot select an infinite object outside the packet.

Finally, H, P, N transport, source, and name the same rows. Since no remaining coordinate contains open-cover compactness, quotient function spaces, host representatives, choice, trace theory, or ambient real completeness, none of those objects can discharge a local NameCert clause. □

Lean4 ✓ `BEDC.Derived.RellichKondrachovUp.RellichKondrachovCarrier namecert obligations`

Theorem 2218.3 (Rellich–Kondrachov finite compact-embedding route)**.** *Every consumer read exported by an accepted* $\mathsf{RellichKondrachov}^{\uparrow}$ *carrier factors through the displayed route*

$$\begin{aligned}&\mathsf{WeakDerivative}^{\uparrow}, \mathsf{Sobolev}^{\uparrow}, \mathsf{SobolevEmbedding}^{\uparrow}, \mathsf{Real}^{\uparrow}\\ &\quad\longmapsto \mathsf{CompactMetric}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}\\ &\quad\longmapsto T, L, H, P, N.\end{aligned}$$

The route supplies only finite compactness windows: a bounded Sobolev source row, an embedding comparison row, a compact-metric witness window, a complete-metric readback window, and the tolerance ledger that relates them. It supplies no selected subsequence, open-cover compactness principle, quotient function space, host representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with a consumer request for compact embedding data. The carrier first exposes the weak-derivative and Sobolev source rows, so the request is restricted to the derivative order, integrability, boundedness, and scalar readbacks displayed in D, W, S.

Next the request must pass through the embedding comparison row E. This row locates the source and target exponents, but by construction it contains no compactness coordinate beyond the route recorded in this carrier.

The target compactness read is then forced through C, M, T. The compact-metric row gives the finite witness window, the complete-metric row gives the Cauchy readback available inside that window, and the schedule T records the tolerance and observation indices used by the route.

It remains only to replay the route. The ledger L states that all compact-embedding observations factor through the displayed rows, while H, P, N transport, source, and name the same finite packet. The absent analytic principles are not recoverable from these rows. □

Falsifiable prediction: Every $\mathsf{RellichKondrachov}^{\uparrow}$ compact-embedding read factors through displayed WeakDerivativeUp, SobolevUp, SobolevEmbeddingUp, CompactMetricUp, CompleteMetricUp, RealUp, subsequence-window schedule, compact-route ledger, transport, provenance, and local naming rows.

Independence witness: weak-derivative, sobolev, sobolev-embedding, compactmetric, complete-metric, real: $\mathsf{RellichKondrachov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite compact-embedding route, target compactness windows, complete-metric readbacks, subsequence-window schedule, route ledger, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RellichKondrachov}^{\uparrow}}$.

Constructive story: A $\mathsf{RellichKondrachov}^{\uparrow}$ packet is generated from bounded Sobolev and weak-derivative rows, an embedding comparison row, compact-metric and complete-metric target windows, scalar readback, a finite subsequence-window schedule, compact-route ledger, componentwise transport, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RellichKondrachov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2218.1, Definition 2218.2, and Definition 2218.3 seed the finite compact-embedding route over WeakDerivative, Sobolev, SobolevEmbedding, CompactMetric, CompleteMetric, and Real neighbours.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RellichKondrachov}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subsequences, open-cover compactness, quotient function spaces, host representatives, trace theory, full PDE regularity, countable choice, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the same finite Rellich–Kondrachov compact-embedding packet.

Chapter 2219

A Concrete Naming Certificate for $\mathsf{AscoliDiagonal}^{\uparrow}$

Following Arzela–Ascoli and Bishop–Bridges constructive analysis, the $\mathsf{AscoliDiagonal}^{\uparrow}$ packet records the finite-window diagonal extraction surface used when compact source nets, equicontinuity ledgers, and complete-metric target rows are already displayed. It consumes chapter 412, chapter 413, and chapter 129 as finite carrier inputs; it does not import a choice-selected subsequence, compact-open topology, or ambient sequence space.

Definition 2219.1 (AscoliDiagonal carrier)**.** *An* $\mathsf{AscoliDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, E, Y, \mathcal{W}, R, S, H, C, P, N)$$

with the following visible rows:

1. *X records compact source-net rows from the* $\mathsf{CompactMetric}^{\uparrow}$ *source, including finite probe windows and their coverage witnesses;*
2. *E records the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *ledger that binds each target tolerance request to a displayed source-radius row uniformly over the family ledger;*
3. *Y records the* $\mathsf{CompleteMetric}^{\uparrow}$ *target rows through which diagonal pointwise reads are sealed;*
4. *$\mathcal{W}$ records finite diagonal* $\mathsf{StreamName}^{\uparrow}$ *windows, with each window naming the source-net points and family indices currently inspected;*
5. *R records* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the target-distance and tolerance witnesses visible on $\mathcal{W}$;*
6. *S records the* $\mathsf{Real}^{\uparrow}$ *seal that repacks the window readbacks into the target-side regular Cauchy boundary;*
7. *H records componentwise* hsame *transports for $X, E, Y, \mathcal{W}, R, S$ and the structural rows;*
8. *C records displayed* Cont *replay through compact source nets, equicontinuity rows, target complete-metric rows, StreamName windows, RegSeqRat readback, and the Real seal;*
9. *P is the* Pkg *provenance over* $\mathsf{AscoliDiagonal}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AscoliDiagonal}^{\uparrow}}$ *row.*

The carrier exports only finite-window diagonal data. It has no coordinate for an unbounded subsequence selector, an ambient compact-open topology, or a choice principle over all family members.

Lean4 ✓ `BEDC.Derived.AscoliDiagonalUp.AscoliDiagonalTasteGate single carrier alignment`

Theorem 2219.2 (AscoliDiagonal NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{AscoliDiagonal}^\uparrow}$ *surface consists exactly of carrier admission for the packet of Definition 2219.1; classifier transport for compact source-net rows, equicontinuity-family ledgers, complete-metric target rows, finite* $\mathsf{StreamName}^\uparrow$ *diagonal windows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks, and the* $\mathsf{Real}^\uparrow$ *seal; finite-window replay through* H, C, P, N*; and non-escape from unbounded subsequence selection, ambient function-space topology, and open-cover compactness.*

Proof. Carrier admission is projection from Definition 2219.1. The compact source rows, family equicontinuity rows, target completion rows, finite diagonal windows, readback rows, Real seal, transport rows, continuation routes, provenance, and local name are all displayed.

Classifier transport is componentwise. The source rows are transported through the compact-metric finite nets, the family rows through the EquicontinuityFamily ledger, and the target rows through the complete-metric and Real readback boundaries.

Finite-window replay uses $\mathcal{W}$ and R. Each consumer sees only the currently displayed diagonal window, its RegSeqRat readback rows, and the Real seal S carried through H, C, P, N.

The non-escape clauses follow by inspecting the carrier. No coordinate names an unbounded subsequence selector, ambient compact-open topology, or open-cover compactness object. □

Theorem 2219.3 (AscoliDiagonal finite-window extraction)**.** *Every public extraction read from an accepted* $\mathsf{AscoliDiagonal}^\uparrow$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{EquicontinuityFamily}^\uparrow,\ \mathsf{CompleteMetric}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow,\ \mathsf{NameCert}_{\mathsf{AscoliDiagonal}}$$

The route first reads compact source nets, then equicontinuity-family modulus rows, then target complete-metric rows, then finite diagonal StreamName windows, then RegSeqRat readback and the Real seal. It exports a finite-window diagonal extraction surface and no choice-selected subsequence or compact-open topological completion.

Proof. Start with an accepted carrier $D = (X, E, Y, \mathcal{W}, R, S, H, C, P, N)$. The compact source row X supplies the finite source-net windows, and the equicontinuity row E supplies the shared modulus ledger for those windows.

The target row Y supplies the complete-metric readback surface. Each diagonal inspection is then restricted to the finite $\mathsf{StreamName}^\uparrow$ window $\mathcal{W}$, so the consumer sees only the displayed source-net points and family indices.

The RegSeqRat row R records the finite readback values for that window, and the Real seal S repacks the readback as the visible target boundary. Transport by H and replay through C preserve the same rows, while P and N record provenance and local naming.

Thus extraction is finite-window diagonal replay. Since the carrier contains no unbounded subsequence selector, no ambient compact-open topology, and no topological completion row, none of those objects is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{AscoliDiagonal}^\uparrow}$**.**

Constructive story: An $\mathsf{AscoliDiagonal}^\uparrow$ packet is generated from compact source-net rows, equicontinuity-family modulus rows, complete-metric target rows, finite StreamName diagonal windows, RegSeqRat readback rows, a Real seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AscoliDiagonal}^\uparrow}$, seed)

Scope closed: Definition 2219.1, Definition 2219.2, and Definition 2219.3 seed the finite-window diagonal extraction packet over compact source nets, equicontinuity ledgers, and complete-metric target rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AscoliDiagonal}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a choice-selected subsequence, ambient compact-open topology, open-cover compactness, sequential compactness, or a standard bridge.

Upgrade path: Obligation closure requires Lean carrier admission, classifier transport, finite-window extraction, readback sealing, and non-escape rows.

Chapter 2220

A Concrete Naming Certificate for $\mathsf{CauchyTailControl}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailControl}^{\uparrow}$ packet records the finite threshold surface by which a regular Cauchy name answers a requested tail tolerance before any Real or Completion consumer reads the result. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence route of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 1506. The packet is the finite tail-threshold control interface for regular Cauchy names; it does not assert a selected limit, quotient equality of names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2220.1 (CauchyTailControl carrier)**.** *A* $\mathsf{CauchyTailControl}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, W, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite precision-request row. It records the displayed tolerance request whose tail threshold is being certified.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened for that request. It records the finite observations available before the threshold is accepted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the dyadic radius, comparison rows, slack budget, and denominator discipline used by the tail check.*
4. *B is the tail-threshold row. It records the finite bound after which every displayed pair of regular rational reads is controlled by the tolerance in D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the windows and dyadic ledger under the same threshold row before a regular Cauchy name can be exported.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal. It is available only after Q, W, D, B, R have displayed the precision request, windows, tolerance ledger, tail threshold, and regular handoff.*
7. *H records componentwise* hsame *transport for the request, window, tolerance, threshold, handoff, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each precision request through $Q, W, D, B, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailControl}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailControl}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, W, D, B, R, E, H, C, P, N$. It has no coordinate for a chosen limit, quotient equality of streams, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2220.2 (CauchyTailControl NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailControl}^{\uparrow}$ *carrier $T = (Q, W, D, B, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailControl}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed precision request, finite stream window, dyadic tolerance ledger, tail-threshold row, regular rational handoff, terminal seal, and structural rows H, C, P, N;*
2. *request exactness, requiring every tail-control read to expose Q before any threshold or regular handoff row is available;*
3. *window and tolerance accountability, requiring W and D to provide the inspected observations, dyadic radius, comparison rows, slack budget, and denominator discipline used by the threshold;*
4. *tail-threshold exactness, requiring B to name the finite bound after which the displayed regular rational reads are controlled by the requested tolerance;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to consume the same W, D, B rows before the terminal* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal E may be read;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No CauchyTailControl obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyTailControlUp.CauchyTailControlTasteGate single carrier alignment`

Proof. Project the accepted packet $T = (Q, W, D, B, R, E, H, C, P, N)$. The carrier definition exposes exactly the request row, stream window, dyadic ledger, tail threshold, regular handoff, terminal seal, and structural rows.

The request and window data are read first. The row Q fixes the finite tolerance request, while W supplies only the observations admitted for that request.

The dyadic and threshold rows then determine the tail control. The row D carries the radius, comparison, slack, and denominator data, and B records the finite bound tied to that ledger.

Finally R, E, H, C, P, N hand off, seal, transport, replay, source, and name the same displayed route. Since the excluded limit, quotient, choice, order, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2220.3 (CauchyTailControl RegSeqRat handoff)**.** *Let $T = (Q, W, D, B, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyTailControl}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or completion-facing consumer of the packet factors through the displayed tail-control route*

$$Q \longmapsto W \longmapsto D \longmapsto B \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a finite precision request, finite stream observations, dyadic tolerance arithmetic, an explicit tail threshold, regular rational readback, terminal sealing, transport, replay, provenance, and local naming before any Real or Completion datum is available. The

route cannot use a selected limit, quotient equality of names, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$ or completion-facing read of T. By Definition 2220.1, the read has no direct access to a limit, quotient, or ambient completion object; it can inspect only the displayed rows.

The first part of the route is fixed by the request and observation rows. The request Q names the tolerance, W supplies the finite observations, and D supplies the dyadic arithmetic used to compare the observed tail.

The tail-control step is the row B. It binds the finite threshold to the same request, observations, and dyadic ledger, so the later regular handoff cannot be detached from the displayed tail budget.

The row R gives the regular rational readback only after Q, W, D, B have been displayed, and E, H, C, P, N seal, transport, replay, source, and name the same route. Therefore every accepted consumer factors through the displayed tail-control route and has no path to the excluded objects. □

Falsifiable prediction: Every $\mathsf{CauchyTailControl}^{\uparrow}$ Real or Completion read factors through a precision request, StreamName finite windows, a DyadicRatCore tolerance ledger, an explicit tail threshold, RegSeqRat handoff, terminal sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletion: $\mathsf{CauchyTailControl}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the request row, observation window, dyadic tolerance ledger, tail-threshold row, regular handoff, terminal seal, transport, continuation, provenance, and local naming as one finite control packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailControl}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailControl}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2220.1, Definition 2220.2, and Definition 2220.3 seed the finite tail-threshold control interface consumed by regular Cauchy names, Real seals, and MetricCompletion-facing readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailControl}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of names, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exactness, StreamName window accountability, DyadicRatCore tolerance accountability, tail-threshold exactness, RegSeqRat handoff, terminal seal nonescape, and structural transport for BEDC.Derived.CauchyTailControlUp.

Constructive story: A $\mathsf{CauchyTailControl}^{\uparrow}$ packet is generated from a precision request, StreamName finite windows, a DyadicRatCore tolerance ledger, an explicit tail-threshold row, a RegSeqRat handoff, a RealUp or completion-facing seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2221

A Concrete Naming Certificate for $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$

Following Kelley and Bourbaki on the compact-open topology, the $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ packet records the finite BEDC surface by which a compact source and a target real seal expose a reusable function-space carrier. It reads the compact-open sup-metric boundary of chapter 1903, the continuous-map graph rows of chapter 130, and the uniform-tail endpoint route of chapter 463. The packet names the carried compact-open function-space data; it does not install an ambient topology on all host functions, quotient function equality, open-cover compactness, selected limits, or pointwise-only convergence.

Definition 2221.1 (Compact-open function-space carrier)**.** *A* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, S, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite source-window and compact-net inspection rows consumed by compact-open requests.*
2. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *graph ledger whose graph rows expose the admissible maps out of X.*
3. *S is the* $\mathsf{CompactOpenMetric}^{\uparrow}$ *subbasic-radius row, including the finite source-window, target-distance, and radius-budget coordinates used to compare graph rows.*
4. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *tail ledger recording the displayed uniform-tail route by which a function-space endpoint may be read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the sampled graph values and tail comparisons are observed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows and tail ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback has been displayed.*
8. *H records componentwise* hsame *transport for the compact source row, continuous-map graph rows, compact-open radius rows, uniform-tail ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming row.*

9. *C records displayed* Cont *replay from a compact-open function-space request through graph inspection, subbasic-radius comparison, uniform-tail readback, stream windows, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactOpenMetric}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenFunctionSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, S, U, W, R, E, H, C, P, N$. No carrier coordinate names quotient function equality, an unrestricted function-space topology, open-cover compactness, selected limits, pointwise-only convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactOpenFunctionSpaceUp.CompactOpenFunctionSpaceTasteGate single carrier alignment`

Theorem 2221.2 (Compact-open function-space NameCert obligations)**.** *For every accepted* CompactOpenFunctio *carrier $K = (X, F, S, U, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenFunctionSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact source row, continuous-map graph ledger, compact-open subbasic-radius row, uniform-tail ledger, stream-window row, regular readback, real seal, transport, replay, provenance, and local naming row;*

2. *graph visibility, requiring every function-space read to enter through the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *graph ledger F before it can inspect values;*

3. *subbasic-radius stability, requiring every compact-open comparison to use the same* $\mathsf{CompactOpenMetric}^{\uparrow}$ *row S and the finite source-window and radius-budget coordinates stored there;*

4. *uniform-limit handoff, requiring the endpoint read to pass through the* $\mathsf{UniformLimit}^{\uparrow}$ *tail ledger U, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R before reaching the* $\mathsf{Real}^{\uparrow}$ *seal E;*

5. *non-escape, requiring H, C, P, N to preserve the same displayed rows and to export no quotient function equality, unrestricted function-space topology, open-cover compactness, selected limit, pointwise-only convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No function-space NameCert obligation is stored outside those rows.

Proof. Project the carrier in Definition 2221.1. The visible analytic coordinates are exactly the compact source row, continuous-map graph ledger, compact-open subbasic-radius row, uniform-tail ledger, stream windows, regular readback, real seal, and structural rows.

Graph visibility is forced by the coordinate F. A function-space read has no independent map coordinate, so every value inspection first passes through the displayed $\mathsf{ContinuousMap}^{\uparrow}$ graph rows.

The compact-open comparison is forced by S. Its finite source windows and radius budgets are the only subbasic-radius data in the packet, hence classifier transport can preserve compact-open comparisons only by transporting the same row through H and replaying it through C.

The endpoint handoff is the displayed route U, W, R, E. The uniform-tail ledger supplies the comparison schedule, the stream-window row exposes the finite observations, the regular readback records the rational name, and the real seal is reached only after those rows are visible.

Finally inspect P and N. Their source list contains no quotient-function, ambient-topology, open-cover, selected-limit, pointwise-only, or ambient-completeness coordinate, so the listed rows exhaust the local NameCert surface. □

Theorem 2221.3 (Compact-open function-space uniform-convergence handoff)**.** *Every uniform-convergence consumer of an accepted* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{ContinuousMap}^{\uparrow}, \mathsf{CompactOpenMetric}^{\uparrow}, \mathsf{UniformLimit}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Thus the consumer receives continuous-map graph data, compact-open subbasic-radius comparisons, a uniform-tail ledger, finite stream windows, regular rational readback, and a real seal before it treats the endpoint as function-space data. The route exports no quotient function equality, selected host limit, pointwise-only convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (X, F, S, U, W, R, E, H, C, P, N)$. A consumer asking for uniform convergence cannot reach E directly, because the real seal row is downstream of the uniform-tail and readback coordinates.

The graph part of the request first reads F, and compact-open comparison then reads S. This supplies the finite source-window and radius-budget data needed to interpret the request as a compact-open comparison rather than a pointwise-only observation.

The limiting endpoint is then routed through U, W, R, E. The uniform-tail ledger gives the shared tail schedule, the stream windows expose finite observations, the regular readback records the rational name, and the real seal provides the endpoint boundary.

The structural rows H, C, P, N preserve exactly this route. Since the packet contains no quotient equality, selected limit, pointwise-only convergence, or ambient completeness row, none can be exported by the handoff. □

Theorem 2221.4 (Compact-open function-space nonescape)**.** *An accepted* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ *carrier exposes only the rows of Definition 2221.1. In particular, any certified read of its function-space endpoint must factor through the compact source row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph ledger, the* $\mathsf{CompactOpenMetric}^{\uparrow}$ *subbasic-radius row, the* $\mathsf{UniformLimit}^{\uparrow}$ *tail ledger, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *seal. It cannot replace that route by open-cover compactness, an unrestricted function-space topology, quotient equality of maps, a selected host limit, pointwise-only convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier has exactly the rows $X, F, S, U, W, R, E, H, C, P, N$. A certified endpoint read must therefore enter through one of these rows and replay through C while preserving the same components by H.

The only graph coordinate is F, the only compact-open comparison coordinate is S, and the only endpoint route is U, W, R, E. Hence every endpoint read factors through the stated finite route.

The excluded principles are absent from the carrier and from the provenance row P. The local naming row N therefore has no source from which to certify any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenFunctionSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMapUp graph rows, a CompactOpenMetricUp subbasic-radius row, a UniformLimitUp tail ledger, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenFunctionSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2221.1, Definition 2221.2, Definition 2221.3, and Definition 2221.4 seed the compact-open function-space carrier and its finite uniform-convergence handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactOpenFunctionSpace}\uparrow}$, encodedDef)

Lean target: `BEDC.Derived.CompactOpenFunctionSpaceUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of maps, unrestricted function-space topology, open-cover compactness, selected host limits, pointwise-only convergence, ambient Real$^{\uparrow}$ completeness, Lean-checked obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph visibility, subbasic-radius stability, uniform-limit handoff, and non-escape proofs for the same finite packet.

Chapter 2222

A Concrete Naming Certificate for $\mathsf{CauchyRegularityModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyRegularityModulus}^{\uparrow}$ packet records a displayed regularity modulus for Cauchy criteria before a $\mathsf{Real}^{\uparrow}$ or Completion consumer is allowed to read a terminal seal. It sits over the Cauchy surface of chapter 392, the uniform criterion of chapter 435, the finite observation windows of chapter 126, the regular rational sequence route of chapter 301, the dyadic arithmetic ledger of chapter 291, and the real-name seal of chapter 38. The packet names the finite modulus route from criterion data to regular sequence handoff; it does not select a limit or quotient stream representative.

Definition 2222.1 (CauchyRegularityModulus carrier)**.** *A* $\mathsf{CauchyRegularityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, U, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *row naming the local Cauchy criterion request.*
2. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row that records the shared criterion surface consumed by the same request.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the criterion request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the displayed dyadic radius, comparison rows, and denominator discipline used by the modulus.*
5. *R is the regularity-modulus row. It records the finite threshold assignment that makes the criterion readable as a regular sequence route.*
6. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row available only after K, U, W, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the criterion, uniform criterion, window, dyadic ledger, modulus, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the criterion request through $K, U, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRegularityModulus}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRegularityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, U, W, D, R, E, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRegularityModulusUp.CauchyRegularityModulusTasteGate single carrier alignment`

Theorem 2222.2 (CauchyRegularityModulus NameCert obligations)**.** *For every accepted* CauchyRegularityModulu *carrier $M = (K, U, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRegularityModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Cauchy criterion, uniform criterion, stream window, dyadic ledger, regularity modulus, regular rational handoff, transport, replay, provenance, and local naming rows;*
2. *criterion exactness, requiring K and U to be exposed before any modulus or regular rational handoff is available;*
3. *window accountability, requiring W to list only the finite observations admitted by the criterion request;*
4. *dyadic tolerance accountability, requiring D to provide the radius, comparison, slack, and denominator data used by the regularity row;*
5. *modulus exactness, requiring R to name the finite threshold assignment that turns the Cauchy and uniform criterion rows into a regular sequence route;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring E to consume the same K, U, W, D, R rows before any* $\mathsf{Real}^{\uparrow}$ *seal may be read;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No CauchyRegularityModulus obligation is stored outside these rows.

Proof. Project the accepted packet $M = (K, U, W, D, R, E, H, C, P, N)$. The carrier definition exposes exactly the criterion, uniform criterion, window, dyadic ledger, regularity modulus, handoff, and structural rows.

The criterion rows are read first. The row K names the Cauchy request, while U records the uniform criterion surface available to the same request.

The observation and arithmetic rows then determine the finite modulus. The row W lists the admitted finite windows, and D carries the dyadic tolerance data used by the threshold assignment.

Finally R, E, H, C, P, N record the modulus, handoff, transport, replay, source, and local name for the same displayed route. Since the excluded selected-limit, quotient, choice, host-equality, trichotomy, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2222.3 (CauchyRegularityModulus RegSeqRat handoff)**.** *Let $M = (K, U, W, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyRegularityModulus}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing consumer of the packet factors through the displayed regularity route*

$$K \longmapsto U \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives Cauchy criterion data, uniform criterion data, finite stream observations, dyadic tolerance arithmetic, an explicit regularity modulus, regular rational readback, transport, replay, provenance, and local naming before any Real-facing seal is available.

Proof. Begin with a $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or Completion-facing read of M. By Definition 2222.1, the read has no direct access to a selected limit, quotient stream equality, or ambient completeness object.

The criterion part of the route is fixed by K and U. These rows display the local Cauchy request and the uniform criterion surface before any finite modulus row can be consumed.

The finite data part is fixed by W and D. The stream window supplies the observed entries, and the dyadic ledger supplies the tolerance arithmetic that the modulus row must use.

The row R gives the explicit regularity modulus, after which E provides the regular rational handoff. The rows H, C, P, N transport, replay, source, and name the same route, so the consumer has no path to selected limits, quotient equality, choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{CauchyRegularityModulus}^{\uparrow}$ Real or Completion read factors through CauchyCriterion, UniformCauchyCriterion, StreamName, DyadicRatCore, explicit regularity modulus, RegSeqRat handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming rows.

Independence witness: cauchycriterion, uniformcauchycriterion, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyRegularityModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the criterion, uniform criterion, finite-window, dyadic-tolerance, explicit-modulus, regular-handoff, transport, continuation, provenance, and local naming rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRegularityModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRegularityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2222.1, Definition 2222.2, and Definition 2222.3 seed the finite regularity-modulus route from Cauchy criteria to regular rational handoff and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRegularityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exactness, StreamName window accountability, DyadicRatCore tolerance accountability, regularity-modulus exactness, RegSeqRat handoff, terminal seal nonescape, and structural transport for BEDC.Derived.CauchyRegularityModulusUp.

Constructive story: A $\mathsf{CauchyRegularityModulus}^{\uparrow}$ packet is generated from CauchyCriterion and UniformCauchyCriterion rows, StreamName finite windows, a DyadicRatCore tolerance ledger, an explicit regularity-modulus row, a RegSeqRat handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2223

A Concrete Naming Certificate for FiniteObservationConfluence$^{\uparrow}$

The FiniteObservationConfluence$^{\uparrow}$ packet realizes the typed-example audit surface of chapter 9382 as finite confluence-facing observation data. It sits between the typed examples gallery of chapter 756, the coverage matrix of chapter 923, the finite observation-budget selector of chapter 782, the bounded normal equality checker of chapter 752, and the confluence critical-pair boundary of chapter 2212. The packet records selected typed examples, reduction observations, normal-form comparisons, and a critical-pair boundary handoff; it is not a theorem of global confluence.

Falsifiable prediction: Every accepted FiniteObservationConfluence$^{\uparrow}$ audit read exposes typed-example rows, finite observation-budget rows, reduction observations, bounded normal-equality rows, confluence-boundary handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it can be cited by a confluence audit consumer.

Independence witness: typedexamplegallery, typedexamplecoveragematrix, finiteobservation-budgetselector, boundednormalequalitychecker, metacic-confluence-critical-pair-boundary: FiniteObservationConflu is not bijective to any listed sibling because it jointly carries typed-example selection, finite budget, reduction observation, normal comparison, confluence-boundary handoff, transport, replay, provenance, and local naming rows.

Definition 2223.1 (FiniteObservationConfluence carrier)**.** *A* FiniteObservationConfluence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, B, R, E, Q, J, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-example row read from the MetaCIC gallery and coverage-matrix surface.*
2. *B is the* FiniteObservationBudgetSelector$^{\uparrow}$ *row bounding which example observations are admitted.*
3. *R is the reduction-observation row, recording the displayed beta, application, projection, or eliminator computation inspected by the packet.*
4. *E is the* BoundedNormalEqualityChecker$^{\uparrow}$ *row used when the observation is compared by bounded normal equality.*
5. *Q is the* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ *handoff row, recording the critical-pair and common-reduct boundary available to the same observation.*

6. *J is the confluence-observation ledger row tying the typed example, budget, reduction observation, bounded checker, and critical-pair boundary to one finite audit route.*
7. *H records componentwise* hsame *transport for the typed-example, budget, reduction, bounded-checker, boundary, ledger, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the typed example through $T, B, R, E, Q, J, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationConfluence}^{\uparrow}$, $\mathsf{FiniteObservationBudgetSelector}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationConfluence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, R, E, Q, J, H, C, P, N$. It has no coordinate for arbitrary closed-term Church–Rosser, full parallel diamond, strong normalisation, subject reduction, residual-substitution compatibility, or unchecked conversion decidability.

Lean4 ✓ `BEDC.Derived.FiniteObservationConfluenceUp.FiniteObservationConfluenceTasteGate single carrier alignment`

Theorem 2223.2 (FiniteObservationConfluence NameCert obligations)**.** *For every accepted* FiniteObservationConf *carrier $F = (T, B, R, E, Q, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationConfluence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the typed-example row, finite observation budget, reduction observation, bounded equality checker, confluence-boundary handoff, observation ledger, transport, replay, provenance, and local naming row;*
2. *typed-example exactness, requiring T to name an accepted gallery or coverage-matrix example before any reduction observation is read;*
3. *budget accountability, requiring B to bound the visible observation route for the selected example;*
4. *reduction-observation exactness, requiring R to display the beta, application, projection, or eliminator computation actually inspected by the packet;*
5. *bounded-comparison handoff, requiring E to consume only the displayed observation rows;*
6. *confluence-boundary handoff, requiring Q to read only the finite critical-pair and common-reduct boundary available to the same observation;*
7. *ledger non-escape, requiring J, H, C, P, N to transport, replay, source, and name the same finite route without exporting global confluence, subject reduction, strong normalisation, residual-substitution compatibility, or unchecked conversion decidability.*

No FiniteObservationConfluence obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2223.1. The projection exposes exactly $T, B, R, E, Q, J, H, C, P, N$.

The typed-example and budget rows are read first. The row T names the gallery example, and B restricts the observation route to a finite budget.

The reduction observation is then displayed by R. The bounded checker E may compare only the rows already exposed by the typed-example, budget, and reduction observation.

The confluence-boundary handoff Q is similarly downstream of the finite observation. It may read the critical-pair and common-reduct boundary, but it does not turn the packet into a global confluence theorem.

Finally J, H, C, P, N ledger, transport, replay, source, and name the same route. Since the excluded metatheorems are not coordinates of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2223.3 (FiniteObservationConfluence MetaCIC handoff)**.** *Let $F = (T, B, R, E, Q, J, H, C, P, N)$ be an accepted* $\mathsf{FiniteObservationConfluence}^{\uparrow}$ *carrier. Every MetaCIC confluence-audit consumer of F factors through the displayed route*

$$T \longmapsto B \longmapsto R \longmapsto E \longmapsto Q \longmapsto J \longmapsto H, C, P, N.$$

Consequently typed-example checker rows, reduction observations, and confluence-boundary handoff are consumed as one finite observation packet, not as a claim of arbitrary confluence.

Proof. Start with a confluence-audit consumer of F. The carrier gives no direct coordinate for arbitrary Church–Rosser, full parallel diamond, subject reduction, strong normalisation, or residual-substitution compatibility.

The consumer must first read the typed-example row T and the finite budget row B. These rows identify the concrete example and the finite observation surface.

The reduction row R and bounded checker row E then supply the inspected computation and comparison data. They are tied to the same budget and cannot range over arbitrary closed terms.

The critical-pair boundary row Q is read only after the finite observation is displayed. The ledger and structural rows J, H, C, P, N then record, transport, replay, source, and name the same route, proving the stated factorization. □

The Lean TasteGate surface for this packet must provide `Nontrivial FiniteObservationConfluenceUp` and `FieldFaithful FiniteObservationConfluenceUp`. The field-faithfulness check must keep the typed-example row, finite budget row, reduction-observation row, bounded-checker row, confluence-boundary handoff row, and ledger row as distinct carrier fields rather than aliases for a global confluence theorem.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationConfluence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationConfluence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2223.1, Definition 2223.2, and Definition 2223.3 seed the finite typed-example observation route from checker rows and reduction observations to confluence-boundary handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationConfluence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary closed-term Church–Rosser, full parallel diamond, subject reduction, strong normalisation, residual-substitution compatibility, unchecked conversion decidability, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, typed-example exactness, budget accountability, reduction-observation exactness, bounded-comparison handoff, confluence-boundary handoff, ledger non-escape, and TasteGate nontriviality and field faithfulness for BEDC.Derived.FiniteObservationConfluenceUp.

Constructive story: A $\mathsf{FiniteObservationConfluence}^{\uparrow}$ packet is generated from typed-example rows, a finite observation-budget selector row, reduction-observation rows, a bounded normal-equality checker row, a MetaCIC confluence critical-pair boundary handoff, a confluence-observation ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2224

A Concrete Naming Certificate for $\mathsf{OscillationModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OscillationModulus}^{\uparrow}$ packet records the finite route by which local oscillation control is read as explicit modulus data before a Real or uniform-continuity consumer treats the bound as a continuity certificate. It sits between the Real oscillation interface of chapter 1839, the function-specific modulus route of chapter 1631, and the uniform boundary of chapter 391. The packet is an L10 NameCert surface for converting displayed oscillation rows into displayed tolerance-window rows; it does not assert Real completeness, compactness, quotient streams, selected limits, host equality, countable choice, or bridge-axis evidence.

Definition 2224.1 (Oscillation-modulus carrier)**.** *An* $\mathsf{OscillationModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M_{\mathrm{osc}} = (O, U, K, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{RealOscillation}^{\uparrow}$ *row exposing the local oscillation ledger and its Real-facing readback route.*
2. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *boundary row that receives one explicit source-window budget for each requested target tolerance.*
3. *K is the* $\mathsf{ModulusContinuity}^{\uparrow}$ *row converting the oscillation bound into the displayed tolerance-to-window schedule.*
4. *D is the dyadic tolerance ledger shared by the oscillation estimate and the modulus request.*
5. *W is the finite source-window row selected by K and inspected by the oscillation ledger O.*
6. *R is the terminal Real-facing readback reached after O, K, D, W have been displayed.*
7. *H records componentwise* hsame *transport for the oscillation row, uniform-continuity row, modulus row, tolerance ledger, source window, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each target tolerance request through D, K, W, O, R and into the uniform-continuity boundary U.*
9. *P is the* Pkg *provenance over* $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{RealOscillation}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, U, K, D, W, R, H, C, P, N$ and the displayed routes among those rows. It has no coordinate for Real completeness, compactness, quotient streams, selected limits, host equality, countable choice, or bridge-axis evidence.

Lean4 ✓ `BEDC.Derived.OscillationModulusUp.OscillationModulusTasteGate single carrier alignment`

Theorem 2224.2 (Oscillation-modulus NameCert obligations)**.** *For every accepted* $\mathsf{OscillationModulus}^{\uparrow}$ *carrier* $M_{\mathrm{osc}} = (O, U, K, D, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OscillationModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Real oscillation row, uniform-continuity boundary row, modulus-continuity row, dyadic tolerance ledger, source window, Real readback, transport, replay, provenance, and local naming rows;*
2. *oscillation-to-modulus exposure, requiring every requested target tolerance read from D to pass through K before a source window W is opened;*
3. *oscillation ledger compatibility, requiring O to inspect the same D, W, R route that K exports as explicit modulus data;*
4. *uniform boundary handoff, requiring U to consume only the displayed modulus schedule K and the oscillation readback O, R;*
5. *ledger non-escape, requiring H, C, P, N to export no Real completeness, compactness, quotient streams, selected limits, host equality, countable choice, or bridge-axis evidence.*

No local obligation is stored outside those rows.

Theorem 2224.3 (Oscillation-modulus Real oscillation handoff)**.** *Every Real-facing read of an accepted* $\mathsf{OscillationModulus}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RealOscillation}^{\uparrow},\ \mathsf{ModulusContinuity}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg}$$

Thus the read receives local oscillation control only as an explicit tolerance-window schedule and receives no Real completeness principle, compactness principle, quotient stream equality, selected limit, host equality, or countable-choice schedule.

Theorem 2224.4 (Oscillation-modulus uniform-continuity boundary)**.** *Let* $M_{\mathrm{osc}} = (O, U, K, D, W, R, H, C, P, N)$ *be an accepted* $\mathsf{OscillationModulus}^{\uparrow}$ *carrier. Every* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer of U reads the public modulus through D, K, W, O, R before any uniform-continuity conclusion is exposed. Consequently the consumer receives an explicit finite tolerance-window route grounded in the oscillation ledger, not a compactness theorem, selected modulus oracle, quotient completion, host equality, ambient Real completeness, countable choice, or bridge certificate.*

Falsifiable prediction: Every $\mathsf{OscillationModulus}^{\uparrow}$ consumer read factors through one RealOscillation row, one ModulusContinuity row, one UniformContinuity boundary row, one DyadicRatCore tolerance ledger, one StreamName source window, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert, and the terminal RealUp seal.

Independence witness: real-oscillation, moduluscontinuity, uniformcontinuity: $\mathsf{OscillationModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an oscillation ledger, an explicit tolerance-to-window schedule, a uniform-consumer boundary, a shared dyadic-window-readback route, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{OscillationModulus}^{\uparrow}$ packet is generated from a RealOscillation row, a UniformContinuity boundary row, a ModulusContinuity tolerance-to-window row, a dyadic tolerance ledger, a finite source window, a Real-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2224.1, Definition 2224.2, Definition 2224.3, and Definition 2224.4 seed the finite handoff from Real oscillation control to explicit modulus data and uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, compactness, quotient streams, selected limits, host equality, countable choice, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, oscillation-to-modulus exposure, oscillation ledger compatibility, uniform boundary handoff, transport and replay preservation, provenance exhaustion, and local naming for future BEDC.Derived.OscillationModulusUp.

Chapter 2225

A Concrete Naming Certificate for $\mathsf{SeparableMetricSpace}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{SeparableMetricSpace}^{\uparrow}$ packet records the traditional countable-dense metric interface as finite BEDC rows. It sits between the metric carrier of chapter 57, the finite observation schedule of chapter 126, the regular rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet names a displayed dense enumeration and its approximation obligations; it is not quotient completion, not a selected-representative principle, not countable choice, not open-cover compactness, and not completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2225.1 (SeparableMetricSpace carrier)**.** *A* $\mathsf{SeparableMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, D, E, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row, including the source point histories, metric-read classifier, and finite distance-query surface.*
2. *D is the countable dense enumeration row, represented as a* $\mathsf{StreamName}^{\uparrow}$ *schedule whose displayed entries are inspected only through finite windows.*
3. *E is the density evidence row: for every displayed point request and positive rational tolerance request visible in M, it stores a finite witness index in D together with the corresponding distance-read budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the requested dense entries and the finite witness indices.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance bounds attached to the metric requests.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, D, E, W, R have been displayed.*
7. *H is the componentwise* hsame *transport ledger for metric, dense stream, density evidence, finite windows, regular rational readbacks, real seals, replay, provenance, and local naming rows.*
8. *C is the displayed* Cont *replay route forcing every density read to pass through*

$$M, D \longmapsto E, W, R \longmapsto Q \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{SeparableMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableMetricSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, selected representatives, countable choice, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeparableMetricSpaceUp.SeparableMetricSpaceTasteGate single carrier alignment`

Theorem 2225.2 (SeparableMetricSpace NameCert obligations)**.** *For an accepted* $\mathsf{SeparableMetricSpace}^{\uparrow}$ *carrier* $S = (M, D, E, W, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableMetricSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric row, dense stream row, density evidence row, finite StreamName windows, RegSeqRat readbacks, Real seal, transport, replay, provenance, and local naming rows;*
2. *dense-stream visibility, requiring every dense witness to be read through D and W before it can answer a metric approximation request;*
3. *approximation evidence, requiring every displayed point-and-tolerance request to pass through E and then through the metric row M and rational readback row R;*
4. *real-seal order, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *readback Q to remain downstream of the metric, stream, density, and regular-rational rows;*
5. *transport stability, requiring H and C to preserve the same displayed metric and dense-stream coordinates componentwise;*
6. *ledger non-escape, stating that P and N export no quotient completion, selected representative, countable-choice schedule, open-cover compactness argument, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 2225.1. The tuple exposes exactly the metric carrier, dense stream, density evidence, finite windows, regular rational readbacks, real seal, transport, replay, provenance, and local naming rows.

Dense-stream visibility uses D together with the finite window row W. A consumer can inspect only the displayed dense entries selected by the StreamName schedule, so the countable dense surface is a named stream row rather than a host enumeration principle.

Approximation evidence uses E between the metric row M and the rational readback row R. The witness index and distance budget are stored as finite rows before any terminal real seal is read.

The real-seal, transport, and non-escape clauses follow from the remaining coordinates. The seal row Q is downstream of M, D, E, W, R, while H, C, P, N preserve, replay, source, and name only those displayed rows. Since no excluded principle is a coordinate of the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 2225.3 (SeparableMetricSpace countable-dense stream handoff)**.** *Let $S = (M, D, E, W, R, Q, H, C, P, N$ be an accepted* $\mathsf{SeparableMetricSpace}^{\uparrow}$ *carrier. Every downstream* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of its countable-dense interface factors through*

$$M, D \longmapsto E, W, R \longmapsto Q \longmapsto H, C, P, N.$$

Consequently the handoff exports a displayed countable dense stream with finite approximation evidence, not quotient completion, selected representatives, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Start with a downstream dense-approximation read. The metric row M supplies the point and distance-query surface, while the dense stream row D supplies only finite entries selected through the StreamName window W.

The density evidence row E then chooses the displayed witness index already present in D and attaches the rational budget read through R. This is the point at which the countable-dense route becomes visible to completion and Real-facing consumers.

The terminal seal Q and the structural rows H, C, P, N preserve that route without adding any host-side principle. Thus every consumer receives the same finite dense-stream handoff and none of the excluded completion, choice, compactness, or Real-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparableMetricSpace}^{\uparrow}$ packet is generated from a MetricUp row, a StreamName countable-dense schedule, finite density evidence, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparableMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2225.1, Definition 2225.2, and Definition 2225.3 seed the countable-dense metric interface over displayed metric, stream, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparableMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, open-cover compactness, host real equality, Lean verification, bridge checking, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-stream visibility, approximation evidence, real-seal order, transport stability, and ledger non-escape for BEDC.Derived.SeparableMetricSpaceUp.

Chapter 2226

A Concrete Naming Certificate for $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$

The $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$ packet realizes the candidate-mediated boundary of chapter 9383 as a finite obstruction socket. It is narrower than the normalization router of chapter 1108: the socket records exactly where public strong-normalization reads must carry Candidate, ClosedCandidate, or CandidateSet evidence, and where the frontier refuses an evidence-free closed typed term route.

Definition 2226.1 (MetaCIC normal audit socket carrier)**.** *A* $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, K, L, S, O, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed term row: it records the empty-context typed term whose public normalization read is being audited.*
2. *K is the Candidate evidence row consumed by the direct candidate-to-strong-normalization route.*
3. *L is the ClosedCandidate evidence row consumed by the closed-candidate projection route.*
4. *S is the CandidateSet evidence row consumed by the candidate-set projection route.*
5. *O is the obstruction row recording that the public route is blocked when K, L, S are all absent.*
6. *F is the frontier row tying the socket to* $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, *and* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *consumers.*
7. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, obstruction, frontier, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from candidate evidence to strong-normalization reads and from obstruction rows to refusal reads.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$, *Candidate, ClosedCandidate, CandidateSet, closed typed term rows, obstruction rows, frontier rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalAuditSocket}^{\uparrow}}$ *row.*

The carrier has no coordinate for unconditional strong normalization of every closed typed term, evidence-free application normalization, host reduction normalization, quotient reduction equality, or a normalization oracle outside the displayed socket.

Theorem 2226.2 (MetaCIC normal audit socket NameCert obligations)**.** *For an accepted* MetaCICNormalAuditSoc *carrier* $A = (T, K, L, S, O, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalAuditSocket}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the closed typed term row, Candidate row, ClosedCandidate row, CandidateSet row, obstruction row, frontier row, componentwise transport, replay, provenance, and local naming rows;*
2. *positive route admission, requiring any public strong-normalization read to consume at least one of* K, L, S *before leaving the socket;*
3. *obstruction exactness, requiring* O *to refuse the public route when the closed typed term row* T *has no displayed Candidate, ClosedCandidate, or CandidateSet evidence;*
4. *frontier handoff, requiring* F *to expose the obstruction to the MetaCIC critical path, normalization frontier, normal-form consistency seal, and decidable-boundary consumers;*
5. *transport and replay stability, requiring* H *and* C *to preserve the same evidence-or-obstruction branch componentwise;*
6. *ledger non-escape, stating that* P *and* N *export no evidence-free strong-normalization theorem, no quotient reduction equality, and no host normalization oracle.*

Proof. Carrier admission is projection from Definition 2226.1. The tuple lists the closed typed term, three evidence rows, obstruction row, frontier row, transport, replay, provenance, and local naming coordinates.

For the positive route, inspect K, L, S. Each is a displayed candidate-mediated evidence row, and the socket has no other coordinate from which a public strong-normalization read can be obtained.

For the refusal route, inspect O. When the three evidence rows are absent from the displayed route, O is the only normalization-facing row attached to the closed typed term, so the public route records obstruction rather than a theorem.

The frontier handoff and stability clauses follow through F, H, C, P, N. These rows expose, transport, replay, source, and name the same evidence-or-obstruction branch, so no evidence-free closed-term normalization principle can escape the socket. □

Lean4 ✓ `BEDC.Derived.MetaCICNormalAuditSocketUp.MetaCICNormalAuditSocketTasteGate single carrier alignment`

Theorem 2226.3 (MetaCIC normal audit socket frontier handoff)**.** *Let* $A = (T, K, L, S, O, F, H, C, P, N)$ *be an accepted* $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$ *carrier. Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, *or* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *consumer receives the finite route*

$$T \longmapsto K, L, S \text{ or } O \longmapsto F \longmapsto H, C, P, N.$$

Thus the public normalization route is preserved as candidate-mediated evidence or as an explicit obstruction socket, not as a hidden unconditional strong-normalization theorem.

Proof. Start with the closed typed term row T. A consumer can leave the socket on the positive side only by reading one of the evidence rows K, L, S, corresponding to Candidate, ClosedCandidate, or CandidateSet evidence.

If those evidence rows are absent from the displayed route, the obstruction row O is read instead. This keeps the public route candidate-mediated and blocks the evidence-free closed-term claim at the same boundary named by the normalization audit map.

The frontier row F then hands the branch to the four downstream consumers, while H, C, P, N preserve the finite route. Since neither branch contains a host normalization oracle or quotient reduction equality, the handoff records an auditable socket rather than a global theorem. □

Theorem 2226.4 (MetaCIC critical path normal audit socket handoff)**.** *Every* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *consumer that uses the* $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$ *boundary must preserve the closed typed term row T, the Candidate row K, the ClosedCandidate row L, the CandidateSet row S, the obstruction row O, the frontier row F, the transport row H, the replay row C, the provenance row P, and the local* NameCert *row N before any closed-normal consistency read is admitted.*

Thus a closed-normal consistency consumer can read the socket only as a candidate-mediated branch or as an explicit obstruction branch. It cannot erase Candidate, ClosedCandidate, CandidateSet, obstruction, frontier, transport, replay, provenance, or naming data into an evidence-free normalization theorem.

Proof. Begin with the frontier handoff of Definition 2226.3. It is already the only displayed route from the socket to $\mathsf{MetaCICCriticalPath}^{\uparrow}$.

On the positive side, preserve K, L, S together with T. A critical-path read that later reaches a closed-normal consistency boundary must still be able to identify which candidate row supported the route.

On the obstructed side, preserve O together with F. This keeps the missing-evidence branch visible at the critical-path boundary rather than converting it into a hidden normalization claim.

Finally carry H, C, P, N along both branches. These rows are the transport, replay, provenance, and naming coordinates that make the socket auditable before any consistency read is consumed. □

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPathUp.MetaCICCriticalPathNormalAuditSocketHandoff`

Falsifiable prediction: Deleting Candidate, ClosedCandidate, and CandidateSet evidence from the carrier blocks the public strong-normalization route at the obstruction row O.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, normalformconsistencyseal, metacicdecidableboundary: the socket carries the closed typed term, three candidate-evidence rows, obstruction row, frontier row, transport, replay, provenance, and local NameCert rows, so no listed sibling is row-bijective with it.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalAuditSocket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalAuditSocket}^{\uparrow}$ packet is generated from a closed typed term row, Candidate evidence, ClosedCandidate evidence, CandidateSet evidence, an obstruction row, a frontier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalAuditSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2226.1, Definition 2226.2, and Definition 2226.3 seed the finite candidate-mediated normalization obstruction socket.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalAuditSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim evidence-free strong normalization, unconditional application normalization, host normalization, quotient reduction equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-route admission, obstruction exactness, frontier handoff, transport replay, and ledger non-escape for BEDC.Derived.MetaCICNormalAuditSocketUp.

Chapter 2227

A Concrete Naming Certificate for $\mathsf{RegularCauchyTightening}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyTightening}^{\uparrow}$ packet records the finite operation that turns a regular Cauchy presentation toward an explicit fast schedule before any real-name or completion consumer reads it. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the fast witness surface of chapter 2185, and the terminal real-name seal of chapter 38. The packet is a finite tightening interface: it does not assert quotient equality of streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2227.1 (Regular Cauchy tightening carrier)**.** *A* $\mathsf{RegularCauchyTightening}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, F, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the requested dyadic radius, comparison rows, denominator discipline, and slack budget available to the tightening test.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite stream observations inspected before a tighter schedule is accepted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row. It exposes the regular rational presentation whose tolerance route is being tightened.*
4. *F is the* $\mathsf{FastCauchySequence}^{\uparrow}$ *witness row. It records the explicit fast schedule exported after D, S, R, T have been displayed.*
5. *T is the tightening-schedule row. It binds each displayed tolerance request to the stronger finite budget used by the fast witness.*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream window, regular readback, fast witness, tightening schedule, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from D, S, R, T to the fast witness row F and then through H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTightening}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FastCauchySequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTightening}^{\uparrow}}$ *row.*

The classifier compares accepted tightening packets only through $D, S, R, F, T, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTighteningUp.RegularCauchyTighteningTasteGate single carrier alignment`

Theorem 2227.2 (Regular Cauchy tightening NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTightening}$ *carrier $U = (D, S, R, F, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTightening}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance ledger, stream window, regular readback, fast witness, tightening schedule, transport, replay, provenance, and local naming rows;*
2. *classifier stability under componentwise agreement on $D, S, R, F, T, H, C, P, N$;*
3. *modulus tightening, requiring T to bind every displayed tolerance request to a stronger finite budget over the same dyadic ledger and stream window;*
4. *fast-regular handoff, requiring F to be exported only after the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and tightening rows have been read;*
5. *real-seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$*-facing read to pass through the finite packet without importing quotient equality, selected limits, countable choice, trichotomy, or ambient completeness.*

No RegularCauchyTightening obligation is stored outside $D, S, R, F, T, H, C, P, N$.

Proof. Project the accepted packet $U = (D, S, R, F, T, H, C, P, N)$. The carrier definition exposes exactly the dyadic tolerance ledger, stream window, regular readback, fast witness, tightening schedule, and structural rows.

Classifier stability is componentwise. Agreement of the displayed rows fixes the dyadic budget, visible observations, regular presentation, fast witness, schedule, transport, replay, provenance, and local naming data inspected by the classifier.

The tightening obligation is the row T. It reads the tolerance information in D, the finite observations in S, and the regular Cauchy presentation in R, then supplies the stronger budget consumed by the fast witness row F.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded quotient, choice, order, limit-selection, or ambient-completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2227.3 (Regular Cauchy tightening fast-regular handoff)**.** *Let $U = (D, S, R, F, T, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTightening}^{\uparrow}$ *carrier. Every exported fast-regular witness factors through the displayed route*

$$D \longmapsto S \longmapsto R \longmapsto T \longmapsto F \longmapsto H, C, P, N.$$

Consequently the exported witness can use only the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and* $\mathsf{FastCauchySequence}^{\uparrow}$ *witness rows displayed by the packet.*

Proof. Start with a consumer reading the fast witness coordinate. By Definition 2227.1, the only fast Cauchy output row is F, and it is paired with the same dyadic, stream, regular, and tightening rows.

The dyadic row D supplies the tolerance arithmetic, S supplies the finite window, and R supplies the regular rational readback. The schedule row T binds these three surfaces to the stronger budget demanded by the exported fast witness.

The structural rows H, C, P, N preserve the displayed route without introducing another source of fast witness data. Thus every accepted export factors through the named dyadic, stream, regular, tightening, and fast rows. □

Theorem 2227.4 (Regular Cauchy tightening real-seal boundary)**.** *An accepted* $\mathsf{RegularCauchyTightening}^{\uparrow}$ *packet cannot export quotient stream equality, selected limits, countable choice, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness. Any* $\mathsf{Real}^{\uparrow}$*-facing consumer receives only the finite dyadic ledger, stream window, regular readback, tightening schedule, fast witness, transport, replay, provenance, and local naming rows displayed in* $D, S, R, F, T, H, C, P, N$.

Proof. Inspect the carrier of Definition 2227.1. Its analytic coordinates are D, S, R, F, T, and its structural coordinates are H, C, P, N.

None of these coordinates names a quotient on streams, a selected limit, a choice functional, a trichotomy principle, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. The provenance row records the real-name seal only as a downstream boundary for the finite packet.

Therefore a real-facing consumer can replay the displayed tightening route, but it cannot extract any of the excluded ambient principles from the packet. □

Falsifiable prediction: Every $\mathsf{RegularCauchyTightening}^{\uparrow}$ fast-regular export factors through a DyadicRatCore tolerance ledger, StreamName finite window, RegSeqRat regular readback, tightening schedule, FastCauchySequence witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, fastcauchysequence, real: RegularCauchyTightenir is not bijective to any listed sibling because it jointly carries the tolerance ledger, observation window, regular readback, explicit fast witness, tightening schedule, transport, continuation, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTightening}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTightening}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2227.1, Definition 2227.2, Definition 2227.3, and Definition 2227.4 seed the finite tightening interface from regular Cauchy readback to explicit fast schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTightening}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient completeness, quotient reals, quotient stream equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, modulus tightening, fast-regular handoff, real-seal non-escape, and structural transport rows for BEDC.Derived.RegularCauchyTighteningUp.

Constructive story: A $\mathsf{RegularCauchyTightening}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite window, RegSeqRat regular readback, tightening schedule, FastCauchySequence witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 2228

A Concrete Naming Certificate for $\mathsf{UniformContinuityModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformContinuityModulus}^{\uparrow}$ packet records the explicit tolerance-to-radius surface used by uniformly continuous maps between metric carriers. It sits between the source and target metric certificates of chapter 57, the pointwise continuous-map interface of chapter 130, the consumer modulus surface of chapter 369, and the compact finite-net production route of chapter 1610. Its role is to expose a named finite carrier for continuity moduli, not to assert open-cover compactness, selected real representatives, quotient equality of functions, or ambient completeness.

The compact finite-net specialization may supply the displayed source radius through chapter 4118. This chapter treats the tolerance-radius row as explicit modulus data: it records the δ supplied to the consumer and does not read compactness, choice, selected finite subcovers, or open-cover evidence.

Definition 2228.1 (Uniform continuity modulus carrier)**.** *A* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, F, \varepsilon, \delta, M, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the displayed source-distance classifier and the source-side tolerance comparison used by the modulus.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including the displayed target-distance classifier and the target triangle ledger consumed by later estimates.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose graph and pointwise-continuity reads are constrained by the same source and target metric rows.*
4. *ε is the positive target tolerance history requested by the consumer.*
5. *δ is the positive source radius history selected for that target tolerance.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row receiving the displayed implication from source closeness below δ to target closeness below ε.*
7. *H records componentwise* hsame *transport for $X, Y, F, \varepsilon, \delta, M$ and for the local classifier row.*
8. *C records the displayed* Cont *replay routes by which a consumer reads the same source metric, target metric, continuous-map, tolerance, radius, and modulus rows.*

9. *P is the* Pkg *provenance over* UniformContinuityModulus$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{UniformContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying $X, Y, F, \varepsilon, \delta, M, H, C, P, N$. It has no coordinate for a compactness theorem, selected finite net, open-cover argument, quotient function equality, host function extensionality, or ambient completion principle.

Lean4 ✓ `BEDC.Derived.UniformContinuityModulusUp.UniformContinuityModulusTasteGate single carrier alignment`

Theorem 2228.2 (Uniform continuity modulus NameCert obligations)**.** *For every accepted* UniformContinuityModu *carrier $U = (X, Y, F, \varepsilon, \delta, M, H, C, P, N)$, the local* NameCert$_{\text{UniformContinuityModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric, target metric, continuous-map row, target tolerance, source radius, uniform-modulus handoff, transport, replay, provenance, and local name;*
2. *tolerance-radius exactness, requiring ε and δ to be read before the* UniformModulus$^{\uparrow}$ *consumer row M is available;*
3. *continuous-map accountability, requiring every function read to pass through the displayed* ContinuousMap$^{\uparrow}$ *row F and the source and target metric rows X, Y;*
4. *componentwise transport and replay stability under H and C for the same finite rows;*
5. *ledger refusal, stating that compactness, selected finite nets, open covers, quotient function equality, host extensionality, and completion data are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the carrier in Definition 2228.1. It exposes exactly the source and target metrics, continuous-map row, tolerance, radius, uniform-modulus handoff, and structural rows.

The target tolerance and source radius are read before the consumer row. Thus the UniformModulus$^{\uparrow}$ handoff can inspect only the displayed implication from source closeness below δ to target closeness below ε.

The row F supplies the continuous-map boundary, while H, C, P, N transport, replay, source, and name the same packet. Since compactness, finite-net selection, open covers, quotient equality, host extensionality, and completion data are absent from the carrier, no NameCert obligation can export them. □

Theorem 2228.3 (ContinuousMap handoff)**.** *Every* ContinuousMap$^{\uparrow}$ *consumer of an accepted* UniformContinuityModulus$^{\uparrow}$ *carrier factors through the displayed route*

$$X, Y, F, \varepsilon, \delta \longmapsto M \longmapsto H, C, P, N.$$

The consumer receives the source metric row, target metric row, continuous-map row, requested target tolerance, selected source radius, uniform-modulus handoff, transport, replay, provenance, and local naming. It receives no full host function, compactness theorem, open-cover proof, quotient equality of functions, or ambient completeness witness.

Proof. Start with a ContinuousMap$^{\uparrow}$ read of the packet. By Definition 2228.1, the only function-facing coordinate is F, and it is stored beside the metric rows X, Y and the tolerance-radius pair ε, δ.

The consumer handoff is the row M. It can be reached only after the tolerance and radius rows have been displayed, and the rows H, C, P, N keep the same route under transport, replay, provenance, and local naming.

Therefore every continuous-map consumer sees a finite modulus boundary over the displayed metric and continuous-map rows. The excluded host function, compactness, open-cover, quotient-equality, and completeness objects have no coordinate in the accepted packet. □

Theorem 2228.4 (Composition stability)**.** *Let* $U_1 = (X, Y, F, \varepsilon_1, \delta_1, M_1, H_1, C_1, P_1, N_1)$ *and* $U_2 = (Y, Z, G, \varepsilon_2, \delta_2, M_2, H_2, C_2, P_2, N_2)$ *be accepted* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *carriers with the target metric row of* U_1 *matching the source metric row of* U_2 *under the displayed* hsame *transport. A composed consumer may form its public modulus only by reading the outer target tolerance in* U_2*, the source radius selected by* U_2 *as a target tolerance for* U_1*, the source radius selected by* U_1*, and the two displayed* $\mathsf{UniformModulus}^{\uparrow}$ *handoff rows. The composed route exports no hidden compactness selector, host function composition theorem, quotient-map equality, or completeness witness.*

Proof. Read the second carrier first. Its target tolerance ε_2 selects the source radius δ_2 through M_2, and the shared metric row identifies δ_2 as the tolerance request consumed by the first carrier.

Read the first carrier at that displayed tolerance. Its source radius δ_1 and handoff row M_1 provide the source-side control needed before the outer handoff M_2 is applied.

The componentwise transports H_1, H_2 and replay rows C_1, C_2 keep both packets on the same displayed metric boundary. Since neither packet contains compactness, host function composition, quotient equality, or completeness data, the composed consumer cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{UniformContinuityModulus}^{\uparrow}$ consumer factors through source and target MetricUp rows, a ContinuousMapUp row, a positive target tolerance, a positive source radius, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: continuousmap, uniformmodulus, heinecantormodulus, metric: UniformContinuityModul is not bijective to any listed sibling because it jointly carries the source metric, target metric, ContinuousMap row, target tolerance, source radius, UniformModulus consumer handoff, transport, replay, provenance, and local NameCert rows as one finite modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuityModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2228.1, Definition 2228.2, Definition 2228.3, and Definition 2228.4 seed the finite explicit-modulus surface over source metric, target metric, ContinuousMap, tolerance-radius, and UniformModulus consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, open-cover arguments, selected finite nets, quotient function equality, host function extensionality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-radius exactness, ContinuousMap handoff, composition stability, and field faithfulness for BEDC.Derived.UniformContinuityModulusUp.

Constructive story: A $\mathsf{UniformContinuityModulus}^{\uparrow}$ packet is generated from a source MetricUp row, a target MetricUp row, a ContinuousMap row, a positive target tolerance, a positive source radius, a UniformModulus consumer row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2229

A Concrete Naming Certificate for $\mathsf{ClosedIntervalApproximation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedIntervalApproximation}^{\uparrow}$ packet records the finite endpoint approximation surface for a closed real interval before any compactness or real-completeness principle is available. It sits after the endpoint carrier of chapter 1591, the finite net route of chapter 1250, the rational cover ledger of chapter 1882, the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is an endpoint-window approximation certificate over displayed rows, not Heine–Borel compactness, open-cover compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2229.1 (Closed interval approximation carrier)**.** *A* $\mathsf{ClosedIntervalApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, K, Q, D, S, R, L, U, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted closed bounded interval row whose lower and upper endpoint data are being approximated.*
2. *K is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *row naming the finite mesh and cell route available over I.*
3. *Q is the* $\mathsf{RationalIntervalCover}^{\uparrow}$ *ledger recording the rational endpoint windows and containment tests inspected by the approximation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording dyadic radii, endpoint slack comparisons, and mesh refinements used by K and Q.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row specifying which endpoint observations are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and dyadic tolerance requests.*
7. *L and U are the displayed lower and upper approximation rows delivered to Real and Completion consumers.*
8. *W is the window-coherence ledger saying that later endpoint windows refine the same I, K, Q, D, S, R, L, U route.*

9. *H records componentwise* hsame *transport for the interval row, finite net row, rational cover ledger, dyadic tolerance ledger, stream window, regular readback, lower approximation, upper approximation, coherence ledger, continuation row, provenance row, and local naming row.*

10. *C records displayed* Cont *replay of the route*

$$I \longmapsto K \longmapsto Q, D \longmapsto S, R \longmapsto L, U, W.$$

11. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalApproximation}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{RationalIntervalCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalApproximation}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite endpoint approximation packet. It has no coordinate for a host interval, a selected real endpoint, quotient real equality, trichotomy, an open-cover subcover, Heine–Borel compactness, or Real completeness.

Lean4 ✓ `BEDC.Derived.ClosedIntervalApproximationUp.ClosedIntervalApproximationTasteGate single carrier alignment`

Theorem 2229.2 (Closed interval approximation NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalApproximation}^{\uparrow}$ *carrier*

$$A = (I, K, Q, D, S, R, L, U, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedIntervalApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval row, finite net row, rational cover ledger, dyadic tolerance ledger, stream window, regular readback, lower approximation, upper approximation, coherence, transport, replay, provenance, and local naming rows;*

2. *endpoint source compatibility, requiring K, Q, D, S, R to be read from the same accepted closed interval row I;*

3. *rational and dyadic exactness, requiring Q and D to account for every endpoint containment, tolerance radius, and refinement used to compute L and U;*

4. *finite-window readback, requiring S and R to expose only the inspected endpoint windows before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed;*

5. *approximation coherence, requiring W, H, C to preserve the same endpoint approximation route under later finite-window replay and componentwise transport;*

6. *Real and Completion handoff, allowing downstream consumers to read L and U only through the displayed I, K, Q, D, S, R, W route;*

7. *boundary refusal, excluding Heine–Borel compactness, open-cover compactness, quotient real equality, host endpoint selection, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No closed-interval approximation obligation is stored outside these rows.

Proof. Project the accepted carrier. The projection exposes exactly the interval source, finite net, rational cover ledger, dyadic tolerance ledger, finite stream window, regular rational readback,

lower and upper approximation rows, coherence ledger, transport, replay, provenance, and local naming.

Endpoint source compatibility follows because K, Q, D, S, R are coordinates of the same carrier and are all attached to I. A consumer cannot replace I by a host interval or use a second endpoint source without leaving the displayed packet.

Rational and dyadic exactness are the rows Q and D. They account for the containment tests, dyadic radii, slack comparisons, and mesh refinements used by the lower and upper approximation rows, so no hidden precision source remains for L or U.

The finite-window readback is the route through S and R. These rows expose only the inspected endpoint observations and their regular rational readbacks before a Real-facing consumer can read the approximation.

Finally W, H, C, P, N preserve and name the same route. Since no carrier coordinate stores a host interval, selected real endpoint, quotient real equality, trichotomy branch, open-cover subcover, Heine–Borel compactness theorem, or ambient completeness theorem, the local NameCert surface cannot export any of them. □

Theorem 2229.3 (Closed interval approximation Real-window handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{ClosedIntervalApproximation}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{RationalIntervalCover}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamNam}$$

The consumer receives displayed lower and upper endpoint approximation rows with a coherence ledger, not a host closed interval, not quotient real equality, not a selected endpoint pair, not Heine–Borel or open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with an accepted carrier $A = (I, K, Q, D, S, R, L, U, W, H, C, P, N)$. The only interval source visible to a consumer is I, and the only finite mesh source over it is K.

The rational and dyadic stage is then forced. The ledger Q supplies rational endpoint windows and containment tests, while D supplies dyadic tolerances and refinement data attached to the same interval and mesh rows.

The Real-facing stage passes through finite observations. The window row S and readback row R expose the inspected endpoint data before L and U are delivered as lower and upper approximation rows.

The coherence and structural rows close the handoff. The row W ties later windows to the same route, while H, C, P, N transport, replay, source, and name it. Thus a Real or Completion consumer can read only the finite approximation route and cannot obtain host interval equality, selected endpoints, quotient real equality, trichotomy, compactness, or ambient completeness from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalApproximation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClosedIntervalApproximation}^{\uparrow}$ packet is generated from a closed bounded interval row, a ClosedBoundedIntervalNet row, a RationalIntervalCover ledger, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, lower and upper approximation rows, a coherence ledger, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2229.1, Definition 2229.2, and Definition 2229.3 seed the finite endpoint approximation surface over ClosedBoundedInterval, ClosedBoundedIntervalNet, RationalIntervalCover, DyadicRatCore, StreamName, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, open-cover compactness, quotient real equality, selected endpoints, trichotomy, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint source compatibility, rational and dyadic exactness, finite-window readback, approximation coherence, Real and Completion handoff, and boundary refusal for BEDC.Derived.ClosedIntervalApproximationUp.

Chapter 2230

A Concrete Naming Certificate for $\mathsf{AbelDirichletComparison}^{\uparrow}$

Following Hardy and Bishop–Bridges, the $\mathsf{AbelDirichletComparison}^{\uparrow}$ packet records the finite comparison surface shared by the Abel and Dirichlet convergence tests over the $\mathsf{RealSeries}^{\uparrow}$ corridor. It sits after the real-series source of chapter 1581, the Abel-test face of chapter 1496, the Dirichlet-test face of chapter 1497, the uniform Cauchy handoff of chapter 1736, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet names a single shared-tail route for downstream Tauberian, summability, and real-series consumers; it is not a selected series sum, a quotient equality of partial-sum streams, a countable-choice selector, host completeness, or a replacement for either test.

Definition 2230.1 (Abel–Dirichlet comparison carrier)**.** *An* $\mathsf{AbelDirichletComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, D, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing the real-valued term stream and the finite partial-sum windows read by the comparison.*
2. *A is the* $\mathsf{AbelTest}^{\uparrow}$ *face row, exposing the bounded partial-sum row, monotone null coefficient ledger, and finite Abel summation-by-parts data visible to the packet.*
3. *D is the* $\mathsf{DirichletTest}^{\uparrow}$ *face row, exposing the bounded partial-sum row, coefficient stream, null-tail ledger, and finite Dirichlet tail schedule visible to the packet.*
4. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *shared-tail row, displaying the common tail request and the finite threshold through which both test faces hand off to later Cauchy-style consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed source windows, test-face windows, and shared-tail comparison.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after S, A, D, U, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, Abel face, Dirichlet face, shared-tail row, regular readback, endpoint seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a consumer request through S to the finite source windows, through A and D to the two test faces, through U to the shared tail, through R to regular rational readback, and then through E to the endpoint seal.*

9. *P is the* Pkg *provenance over* $\mathsf{AbelDirichletComparison}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelDirichletComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a chosen infinite sum, an equality quotient on partial-sum streams, a choice-selected convergence witness, an ambient completeness theorem, or an independent test result outside the named faces.

Lean4 ✓ `BEDC.Derived.AbelDirichletComparisonUp.AbelDirichletComparisonTasteGate single carrier alignment`

Theorem 2230.2 (Abel–Dirichlet comparison NameCert obligations)**.** *For every accepted* $\mathsf{AbelDirichletComparison}^{\uparrow}$ *carrier $K = (S, A, D, U, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbelDirichletComparison}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed ten-row packet;*
2. $\mathsf{RealSeries}^{\uparrow}$ *source exposure through S, preserving the finite term and partial-sum windows consumed by the comparison;*
3. *Abel-face exposure through A, preserving the bounded partial-sum, monotone null coefficient, and summation-by-parts rows already named by the Abel-test packet;*
4. *Dirichlet-face exposure through D, preserving the bounded partial-sum, coefficient stream, null-tail, and tail-schedule rows already named by the Dirichlet-test packet;*
5. *shared-tail exactness through U, requiring both faces to hand off to the same* $\mathsf{UniformCauchySeries}^{\uparrow}$ *tail request before a later consumer may read a Cauchy-style conclusion;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *endpoint exposure through R and E;*
7. *componentwise transport, displayed replay, provenance, and local naming through exactly H, C, P, N.*

These are finite-row obligations over the sibling surfaces of chapter 1581, chapter 1496, chapter 1497, chapter 1736, chapter 301, and chapter 38.

Proof. Project the accepted packet. The carrier exposes exactly the real-series source row, Abel-test face, Dirichlet-test face, shared-tail row, regular readback, endpoint seal, componentwise transports, continuation replay, provenance, and local naming row.

Read the analytic faces. The row A can contribute only the Abel-test finite summation-by-parts route already visible through its carrier, while D can contribute only the Dirichlet-test finite tail route already visible through its carrier. The comparison packet does not add a third convergence argument.

The common obligation is the row U. It forces both faces to meet at one $\mathsf{UniformCauchySeries}^{\uparrow}$ shared-tail request before R and E can be read by downstream consumers.

Finally read H, C, P, N. These structural rows transport, replay, source, and name the same route, so the seven listed obligations exhaust the local certificate surface. □

Theorem 2230.3 (Abel–Dirichlet shared-tail route)**.** *Every downstream consumer that compares Abel and Dirichlet evidence through an accepted* $\mathsf{AbelDirichletComparison}^{\uparrow}$ *carrier must factor through*

$$S \longmapsto A, D \longmapsto U \longmapsto R \longmapsto E$$

together with H, C, P, N. Thus Tauberian, Euler-transform, and summability-style consumers receive a finite shared-tail handoff from the $\mathsf{RealSeries}^{\uparrow}$ *corridor rather than two unrelated informal references to the Abel and Dirichlet rows.*

Proof. Start with a downstream comparison read. The source row S is the only $\mathsf{RealSeries}^{\uparrow}$ entry point, so the requested term and partial-sum data are first confined to the displayed real-series windows.

The two test faces are then read in parallel. The Abel face A and Dirichlet face D retain their own finite coefficient and bounded-partial-sum data, but neither face may export an endpoint seal directly from the comparison packet.

The row U is the unique tail-facing bridge between those faces and later Cauchy-style consumers. A consumer therefore reaches regular rational readback R and the endpoint seal E only after the shared $\mathsf{UniformCauchySeries}^{\uparrow}$ row has been displayed.

Transport, replay, provenance, and local naming through H, C, P, N preserve that same finite route. Since the carrier has no coordinate for selected sums, quotient stream equality, countable choice, host completeness, or a separate convergence theorem, no downstream consumer can bypass the shared-tail handoff. □

Falsifiable prediction: An accepted $\mathsf{AbelDirichletComparison}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ endpoint seal, or downstream summability handoff without first reading the $\mathsf{RealSeries}^{\uparrow}$ source row, both test faces, and the $\mathsf{UniformCauchySeries}^{\uparrow}$ shared-tail row refutes the carrier surface.

Independence witness: realseries, abel-test, dirichlet-test, uniform-cauchy-series, regseqrat, real: no listed sibling supplies the whole ten-row packet $K = (S, A, D, U, R, E, H, C, P, N)$, because each sibling contributes only a source, one test face, a shared-tail handoff, a readback, or a seal while this chapter binds the comparison with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelDirichletComparison}^{\uparrow}}$.

Constructive story: An $\mathsf{AbelDirichletComparison}^{\uparrow}$ packet is generated from a $\mathsf{RealSeries}^{\uparrow}$ source row, $\mathsf{AbelTest}^{\uparrow}$ and $\mathsf{DirichletTest}^{\uparrow}$ face rows, one $\mathsf{UniformCauchySeries}^{\uparrow}$ shared-tail row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelDirichletComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2230.1, Definition 2230.2, and Definition 2230.3 seed the finite comparison surface over $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelDirichletComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected series sums, quotient equality of partial-sum streams, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface and shared-tail route.

Chapter 2231

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedCompletion}^{\uparrow}$ packet records the located completion landing object reached when displayed Cauchy data carries an explicit locatedness service before the L10 Real and Completion routes consume it. It sits after the located Cauchy evidence of chapter 2216, the located completion seal of chapter 1805, the metric-completion interface of chapter 1506, and the real Cauchy modulus boundary of chapter 1423. The packet supplies a named constructive completion surface for located Cauchy data; it does not assert quotient equality of Cauchy names, select representatives, use trichotomy, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2231.1 (Bishop located completion carrier)**.** *A* $\mathsf{BishopLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, S, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{CauchyLocatedness}^{\uparrow}$ *row. It records the located Cauchy evidence available before any completion-facing consumer reads a seal.*
2. *S is the* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *row. It seals the located Cauchy data as a completion-facing object without quotienting names.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route. It records the displayed metric-completion source through which the located packet can be consumed by L10 completion readers.*
4. *R is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *row. It records the real-facing modulus or threshold discipline attached to the same Cauchy data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or completion seal exposed only after L, S, M, R have been displayed.*
6. *H records componentwise* hsame *transport for locatedness, seal, metric-completion, real-modulus, terminal seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through L, S, M, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletion}^{\uparrow}$, $\mathsf{CauchyLocatedness}^{\uparrow}$, $\mathsf{LocatedCompletionSeal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, S, M, R, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionUp.BishopLocatedCompletionTasteGate single carrier alignment`

Theorem 2231.2 (Bishop located completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletion}$ *carrier $B = (L, S, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located Cauchy row, located completion seal, metric-completion route, real Cauchy modulus row, terminal seal, and structural rows H, C, P, N;*
2. *locatedness exposure, requiring every completion-facing read to pass through L before the located seal or metric-completion route is inspected;*
3. *seal exactness, requiring S to package the same located Cauchy data without adding quotient names or representative selection;*
4. *completion compatibility, requiring M to consume only the displayed metric-completion route already available from the located Cauchy data;*
5. *real-modulus compatibility, requiring R to expose the same real Cauchy modulus or threshold discipline before the terminal* $\mathsf{Real}^{\uparrow}$ *or completion seal E is read;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Bishop located completion obligation is stored outside these rows.

Proof. Project the carrier of Definition 2231.1. The visible packet consists exactly of the located Cauchy row L, located seal S, metric-completion route M, real-modulus row R, terminal seal E, and structural rows H, C, P, N.

The locatedness row is read first. The chapter chapter 2216 supplies the sibling surface for located Cauchy evidence, and the carrier gives no route to the seal before that row is displayed.

The seal and completion rows then keep the same data on the L10 route. The seal row S is the located-completion boundary of chapter 1805, while M routes the packet through the metric-completion interface of chapter 1506.

Finally the real-modulus and structural rows close the local certificate. The row R records the threshold discipline named in chapter 1423, and E, H, C, P, N expose, transport, replay, source, and name the same route. Since the excluded quotient, selection, trichotomy, choice, host-equality, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the local $\mathsf{NameCert}$ surface. □

Theorem 2231.3 (Bishop located completion nonescape)**.** *Every accepted* $\mathsf{BishopLocatedCompletion}^{\uparrow}$ *Real or completion read factors through the displayed rows $L, S, M, R, E, H, C, P, N$ of Definition 2231.1. The read may inspect the* $\mathsf{CauchyLocatedness}^{\uparrow}$ *row, the* $\mathsf{LocatedCompletionSeal}^{\uparrow}$ *row, the* $\mathsf{MetricCompletion}^{\uparrow}$ *route, the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row, the terminal seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row, but it cannot project quotient completion equality, selected representatives, host equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2231.1. The only visible analytic rows are L, S, M, R, E, and the only structural rows are H, C, P, N.

Use Definition 2231.2. It requires completion-facing reads to pass through locatedness, located sealing, metric-completion compatibility, real-modulus compatibility, terminal sealing, componentwise transport, replay, provenance, and local naming in the displayed order.

The refused objects have no row in the carrier. Hence quotient completion equality, selected representatives, host equality, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be projected from an accepted Bishop located completion read. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedCompletion}^{\uparrow}$ Real or Completion read factors through $\mathsf{CauchyLocatedness}^{\uparrow}$, $\mathsf{LocatedCompletionSeal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, terminal sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchylocatedness, locatedcompletionseal, metriccompletion, realcauchymodulus: $\mathsf{BishopLocatedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located Cauchy evidence, located completion sealing, metric-completion routing, real-modulus threshold discipline, terminal sealing, transport, continuation, provenance, and local naming as one finite completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2231.1, Definition 2231.2, and Definition 2231.3 seed the finite located-completion surface over $\mathsf{CauchyLocatedness}^{\uparrow}$, $\mathsf{LocatedCompletionSeal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host real equality, trichotomy, countable choice, public bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, locatedness exposure, seal exactness, metric-completion compatibility, real-modulus compatibility, and ledger non-escape over the same displayed packet.

Constructive story: The construction starts from located Cauchy data, seals it as a located completion object, routes it through the L10 metric-completion interface, and exposes the real Cauchy modulus only along the same finite NameCert path.

Chapter 2232

A Concrete Naming Certificate for $\mathsf{MetricCofinalCompletion}^{\uparrow}$

Following Bishop–Bridges constructive metric completion and Kelley's net and filter language, the $\mathsf{MetricCofinalCompletion}^{\uparrow}$ packet records the finite cofinal-evidence surface that lets a completion consumer compare Cauchy-filter and Cauchy-net presentations before reading a Real-facing or separated-metric readback. It sits beside the metric source of chapter 57, the Cauchy-filter branch of chapter 333, the Cauchy-net branch of chapter 589, the regular rational and real readbacks of chapter 301 and chapter 38, the separated comparison of chapter 1411, the uniform completion route of chapter 1473, and the metric completion surface of chapter 1506.

Definition 2232.1 (Metric cofinal completion carrier)**.** *A* $\mathsf{MetricCofinalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, F, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row. It records the metric carrier, distance observation, and finite request boundary consumed by the completion-facing route.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch. It records filter-neighborhood evidence and the finite Cauchy request surface over X.*
3. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *branch. It records directed-index evidence, net values, and the finite Cauchy request surface over the same X.*
4. *R is the readback row. It records either the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback needed by regular rational names or the* $\mathsf{Real}^{\uparrow}$ *seal reached from that readback.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row. It records the equality-facing separated metric comparison available after both cofinal branches have been displayed.*
6. *H records componentwise* hsame *transport for the metric source, filter branch, net branch, readback, separated comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$X, F, D \longmapsto R, S \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{MetricCofinalCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{MetricCofinalCompletion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion, selected representatives, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCofinalCompletionUp.MetricCofinalCompletionTasteGate single carrier alignment`

Theorem 2232.2 (Metric cofinal completion NameCert obligations)**.** *For an accepted* $\mathsf{MetricCofinalCompletion}^{\uparrow}$ *carrier* $Q = (X, F, D, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCofinalCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exactness, Cauchy-filter branch exposure, Cauchy-net branch exposure, cofinal comparison of the two branches, readback exactness through* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*, separated comparison through* $\mathsf{SeparatedMetric}^{\uparrow}$*, transport and replay stability, provenance, local naming, and non-escape of quotient completion, selected representatives, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2232.1. The metric source is the row X, and it is the only row that supplies the underlying metric request boundary.

The Cauchy-filter and Cauchy-net branches are the rows F and D. They are both read over X, so a completion-facing consumer can compare their request surfaces without introducing a host quotient or a selected representative.

The readback and separated comparison are the rows R and S. The row R keeps regular rational and real-name evidence as displayed readback data, while S records separated-metric comparison after the two Cauchy branches have been exposed.

The rows H, C, P, N transport, replay, source, and name the same route. Since no other coordinate exists, the excluded host-side completion principles cannot enter the accepted packet. □

Theorem 2232.3 (Metric cofinal completion filter-net cofinality)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCofinalCompletion}^{\uparrow}$ *packet factors the Cauchy-filter and Cauchy-net branches through the same metric source row:*

$$X \longmapsto F, D \longmapsto R, S.$$

Thus the packet records cofinal evidence between the $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *presentations before any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer can inspect the route.*

Proof. Project the accepted carrier and read X first. Both branch rows name Cauchy evidence over this same metric source, so neither branch can change the metric carrier or distance request boundary.

The filter branch F supplies neighborhood evidence, and the net branch D supplies directed-index evidence. Cofinality is recorded by requiring both rows to be visible before the readback row R or the separated comparison row S is consumed.

The downstream completion chapters read only the displayed route. The uniform and metric completion surfaces may consume $X, F, D, R, S, H, C, P, N$, but the packet gives them no separate quotient object, representative selector, choice principle, compact open-cover argument, or ambient completeness seal. □

Theorem 2232.4 (Metric cofinal completion nonescape)**.** *Every accepted* $\mathsf{MetricCofinalCompletion}^{\uparrow}$ *read remains inside the finite route*

$$X, F, D, R, S, H, C, P, N.$$

In particular, no consumer may export quotient completion equality, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ *completeness merely from the cofinal filter-net evidence.*

Proof. The carrier inventory of Definition 2232.1 lists the whole packet. Rows X, F, D, R, S give the metric source, two Cauchy branches, finite readback, and separated comparison.

Rows H and C preserve that inventory under hsame transport and Cont replay. Rows P and N record provenance and local naming for the same displayed route.

None of the excluded principles has a coordinate in the packet or a replay edge out of it. Hence any accepted consumer remains inside the finite cofinal-evidence surface. □

Falsifiable prediction: Every accepted $\mathsf{MetricCofinalCompletion}^{\uparrow}$ completion-facing read displays one metric source row, one Cauchy-filter branch, one Cauchy-net branch, one regular-rational or real readback row, one separated comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it can be consumed by a completion chapter.

Independence witness: metric, cauchyfilter, cauchynet, regseqrat, real, separatedmetric, uniformcompletion, metriccompletion: $\mathsf{MetricCofinalCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a shared MetricUp source, filter and net Cauchy branches, readback, separated comparison, transport, replay, provenance, and local naming as one finite cofinal-evidence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCofinalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCofinalCompletion}^{\uparrow}$ packet is generated from a MetricUp source row, CauchyFilterUp branch, CauchyNetUp branch, RegSeqRatUp or RealUp readback, SeparatedMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCofinalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2232.1, Definition 2232.2, Definition 2232.3, and Definition 2232.4 seed the finite cofinal-evidence interface between Cauchy-filter and Cauchy-net completion presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCofinalCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter and net branch exactness, readback exactness, separated comparison, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.MetricCofinalCompletionUp.

Chapter 2233

A Concrete Naming Certificate for $\mathsf{PicardErrorEstimate}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, this packet records the a posteriori Picard error estimate as finite BEDC NameCert data. It sits between the map-level contraction surface of chapter 1418, the finite iteration route of chapter 395, and the fixed-point endpoint discipline of chapter 396. The packet names only the displayed residual, contraction ratio, Cauchy handoff, and endpoint seal rows; it does not select a host fixed point or import ambient completeness outside the carried rows.

Definition 2233.1 (Picard error-estimate carrier)**.** *A* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, d, T, \lambda, I, r, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the complete metric row whose carried histories contain the displayed Picard iterates.*
2. *d is the metric-distance row used to read adjacent iterate distances and endpoint-facing error estimates.*
3. *T is the contraction-map row, read through the* $\mathsf{ContractionMapping}^{\uparrow}$ *packet and never as an untracked host function.*
4. *λ is the rational contraction-ratio ledger with a displayed row below the unit bound.*
5. *I is the adjacent Picard-iterate ledger supplied by* $\mathsf{PicardContraction}^{\uparrow}$*, recording the visible rows x_k, x_{k+1} and the application route $x_{k+1} = T(x_k)$.*
6. *r is the residual distance row, reading $d(x_{k+1}, x_k)$ from the adjacent ledger.*
7. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row obtained from the residual row and the rational ratio ledger.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that exposes the regular rational name consumed by the completion route.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing error-bound row reached only after r, λ, M, R have been displayed.*
10. *H stores componentwise* hsame *transport for the complete metric, distance, map, ratio, Picard ledger, residual, modulus, handoff, seal, continuation, provenance, and local naming rows.*

11. *C stores displayed* Cont *replay from contraction application to adjacent residual read, from residual and ratio to the modulus row, and from the modulus row to the regular and real seal rows.*

12. *P is the* Pkg *provenance over* $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardErrorEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, d, T, \lambda, I, r, M, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient equality, ambient completeness, global nonlinear analysis, or choice-selected endpoint.

Lean4 ✓ `BEDC.Derived.PicardErrorEstimateUp.PicardErrorEstimateTasteGate single carrier alignment`

Theorem 2233.2 (Picard error-estimate NameCert obligations)**.** *For every accepted* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *carrier $Q = (X, d, T, \lambda, I, r, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardErrorEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, contraction-map compatibility through T, rational-ratio visibility through λ, adjacent Picard ledger access through I, residual-distance read-back through r, Cauchy-modulus construction through M, regular-rational handoff through R,* $\mathsf{Real}^{\uparrow}$ *seal access through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 2233.1. The only metric and contraction data available to the classifier are the complete-metric row X, the distance row d, the contraction-map row T, and the rational ratio ledger λ.

The adjacent Picard ledger I exposes the visible pair whose residual distance is read by r. The modulus row M is downstream of that residual and ratio data, and the handoff row R is downstream of the modulus row.

The endpoint-facing row E is therefore a seal after the displayed residual, ratio, modulus, and regular-rational handoff. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local certificate surface. □

Theorem 2233.3 (Picard a posteriori error-bound route)**.** *Every endpoint error read from an accepted* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *packet factors through the internal classifier route*

$$I \longmapsto r \longmapsto \lambda \longmapsto M \longmapsto R \longmapsto E$$

together with X, d, T, H, C, P, N. Thus the a posteriori error estimate is read first from the adjacent residual row and the contraction ratio before any Cauchy, regular-rational, or real seal row is exposed. The route supplies no selected host fixed point, quotient equality, ambient completeness principle, or choice-selected endpoint.

Proof. Start with an endpoint error request. The carrier permits such a request only after the adjacent Picard ledger I has displayed the two consecutive iterates and the contraction application route through T.

The residual row r reads the distance between those adjacent iterates through d. The ratio row λ is the only contraction-bound coordinate, so the a posteriori estimate can use no other numerical source before the Cauchy-modulus row M is formed.

The regular handoff R and the seal row E are downstream of M. Componentwise transport and replay through H and C, with provenance and naming through P and N, preserve this same route and do not create a host fixed-point coordinate. □

Falsifiable prediction: Every $\mathsf{PicardErrorEstimate}^{\uparrow}$ endpoint error read factors through one complete metric row, one contraction-map row, one rational ratio below one, one adjacent Picard ledger, one residual distance row, one CauchyModulus row, one RegSeqRat handoff, one RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: contractionmapping, picardcontraction, banachfixedpoint: $\mathsf{PicardErrorEstimate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the residual distance row and the ratio-to-modulus error-bound route before the completion and endpoint seal rows are read.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardErrorEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{PicardErrorEstimate}^{\uparrow}$ packet is generated from a complete metric row, a metric-distance row, a contraction-map row, a rational contraction ratio below one, an adjacent Picard ledger, a residual distance row, a CauchyModulus row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardErrorEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2233.1, Definition 2233.2, and Definition 2233.3 bind the finite a posteriori contraction error-bound route through the displayed residual, ratio, modulus, handoff, and seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardErrorEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, selected fixed points, quotient equality, ambient completeness, bridge checking, Lean verification, or general nonlinear analysis.

Upgrade path: Obligation closure requires checked carrier admission, residual readback, ratio-to-modulus routing, regular-rational handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PicardErrorEstimateUp.

Chapter 2234

A Concrete Naming Certificate for $\mathsf{CodonSixBitWindow}^{\uparrow}$

The $\mathsf{CodonSixBitWindow}^{\uparrow}$ packet realizes the codon source atlas of subsection 9462.28.17 as finite BEDC NameCert data. It reads each base through a two-coordinate $\mathsf{Bool}^{\uparrow}$ atlas, packages the three base positions through the product and list surfaces of chapter 34 and chapter 36, compares no-11 suffix constraints against chapter 401, and hands the stop-face suffix row to chapter 1374. The packet records the six-coordinate source window and Flat6 readback; it does not identify source-space equality with the biological translation classifier.

Definition 2234.1 (Codon six-bit window carrier)**.** *A* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, A, L, F, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *coordinate row. It records the two binary axes used to read each base in the codon source atlas.*
2. *A is the three-position atlas row. It records the first, second, and third base windows as ordered pairs of Bool coordinates.*
3. *L is the* $\mathsf{List}^{\uparrow}$ *spine row. It records the ordered six-coordinate window and preserves position order for downstream reads.*
4. *F is the Flat6 readback row. It flattens the three two-coordinate base rows into one six-coordinate source-space vector.*
5. *S is the suffix-handoff row. It records the comparison with* $\mathsf{FibonacciCube}^{\uparrow}$ *no-11 suffix discipline and the handoff to* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *for stop-face reads.*
6. *H records componentwise* hsame *transport for Bool coordinates, product pairs, list spine, Flat6 readback, suffix handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$B, A, L \longmapsto F, S \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{FibonacciCube}^{\uparrow}$, $\mathsf{StopCodonZeckendorfSu}$ *the codon-window-six bridge,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CodonSixBitWindow}^{\uparrow}}$ *row.*
The classifier compares packets through $B, A, L, F, S, H, C, P, N$. It has no coordinate for biological translation-table identity, amino-acid classification, stochastic codon usage, quotient codon equality, or a selected global code table.

Lean4 ✓ `BEDC.Derived.CodonSixBitWindowUp.CodonSixBitWindowTasteGate single carrier alignment`

Theorem 2234.2 (Codon six-bit window NameCert obligations)**.** *For an accepted* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *carrier $W = (B, A, L, F, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CodonSixBitWindow}^{\uparrow}}$ *surface consists exactly of carrier admission, Bool-coordinate exactness, three-position atlas order, List spine preservation, Flat6 readback exactness, Fibonacci-cube suffix comparison, StopCodonZeckendorfSuffix handoff, transport and replay stability, provenance, local naming, and non-escape of biological translation-table identity or amino-acid classification.*

Proof. Carrier admission is projection from Definition 2234.1. The Bool-coordinate row B and three-position atlas row A expose exactly six source coordinates, grouped as three ordered base windows.

The list spine row L keeps the six coordinates ordered, and the Flat6 row F is the only readback that forgets the three-pair grouping while preserving the same source-space coordinates.

The suffix row S compares the final two-coordinate suffix with the finite no-11 discipline of chapter 401 and the stop-face handoff of chapter 1374. This is a source-window handoff, not a biological translation classifier.

The rows H, C, P, N transport, replay, source, and locally name the same route. Since no translation-table or amino-acid row exists in the carrier, those classifiers cannot be exported by the accepted packet. □

Theorem 2234.3 (Codon six-bit Flat6 readback)**.** *Every accepted* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-space read factors through*

$$B, A, L \longmapsto F$$

before it can reach the suffix-handoff row S or the structural rows H, C, P, N. Thus Flat6 equality is equality of the displayed six-coordinate source window, not equality of biological translation classifiers.

Proof. Read B to obtain the Bool coordinate axes and A to group them into three base positions. The product surface preserves the two-coordinate base rows before any flattening occurs.

Read L to preserve the ordered six-coordinate spine. The Flat6 row F is then the unique route that exposes the same coordinates as one source-space vector.

Only after F is displayed may a consumer inspect the suffix handoff S or the structural rows H, C, P, N. Therefore any packet identifying Flat6 source-space equality with biological translation classification leaves the accepted carrier surface. □

Falsifiable prediction: Any $\mathsf{CodonSixBitWindow}^{\uparrow}$ packet that identifies Flat6 source-space equality with the biological translation classifier fails the classifier row, because the accepted carrier has only Bool-coordinate, product/list spine, Flat6 readback, suffix-handoff, transport, replay, provenance, and local naming rows.

Independence witness: stopcodonzeckendorfsuffix, codon_window_spectral_inverse, fibonaccicube, bool, list: $\mathsf{CodonSixBitWindow}^{\uparrow}$ is not bijective to any listed sibling because the explicit six-coordinate Flat6 row and suffix-handoff row jointly preserve source-window equality while refusing biological translation-classifier identity.

The Lean TasteGate follow-up is `BEDC.Derived.CodonSixBitWindowUp.TasteGate`, with explicit Nontrivial $\mathsf{CodonSixBitWindow}^{\uparrow}$ and FieldFaithful $\mathsf{CodonSixBitWindow}^{\uparrow}$ instances preserving $B, A, L, F, S, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{CodonSixBitWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{CodonSixBitWindow}^{\uparrow}$ packet is generated from BoolUp coordinate rows, a three-position product atlas, an ordered ListUp spine, Flat6 readback, FibonacciCubeUp suffix comparison, StopCodonZeckendorfSuffixUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CodonSixBitWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2234.1, Definition 2234.2, and Definition 2234.3 seed the finite six-coordinate source-window packet and its Flat6 readback boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CodonSixBitWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological translation-table identity, amino-acid classification, stochastic codon usage, quotient codon equality, selected global code tables, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Bool-coordinate exactness, product/list spine preservation, Flat6 readback, suffix handoff, nontriviality, and field faithfulness for BEDC.Derived.CodonSixBitWindowUp.TasteGate.

Chapter 2235

A Concrete Naming Certificate for $\mathsf{JensenInequality}^{\uparrow}$

Following Jensen and Hardy–Littlewood–Polya, the $\mathsf{JensenInequality}^{\uparrow}$ packet records the finite convex-combination route by which a convex function row, finitely many affine weights, and displayed real comparisons produce the Jensen inequality surface. It sits over the feasible affine spine of chapter 208, the vector carrier of chapter 47, the scalar and comparison rows of chapter 38, the rational coefficient source of chapter 37, and the endpoint comparison discipline of chapter 1674. The packet names the finite certificate route for Jensen-style convex analysis; it does not assert host convex functions, quotient equality of real names, ambient completeness, or a global optimization principle.

Definition 2235.1 (Jensen inequality carrier)**.** *A* $\mathsf{JensenInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (F, W, A, X, Y, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the convex-function source row, recording the displayed domain, codomain, convexity ledger, and function-read route.*
2. *W is the finite affine-weight row, recording the rational coefficients, nonnegative coefficient checks, and finite-sum-to-one row read through* $\mathsf{Rat}^{\uparrow}$*.*
3. *A is the affine-spine row, recording the* $\mathsf{ConvexSet}^{\uparrow}$ *route that keeps the weighted point inside the displayed convex carrier.*
4. *X is the finite input-family row, recording the finitely many points consumed by the affine spine and their* $\mathsf{VecSpace}^{\uparrow}$ *source reads.*
5. *Y is the convex-image row, recording the finite image reads $F(x_i)$ and the weighted image expression.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, recording the displayed inequality between the function value at the affine point and the weighted image expression.*
7. *Q is the* $\mathsf{RealMinMax}^{\uparrow}$ *endpoint row, recording the comparison and bounding data used when a consumer reads the displayed inequality.*
8. *H records componentwise* hsame *transport for F, W, A, X, Y, R, Q and the local naming row.*

9. *C records displayed* Cont *replay from the source rows through the affine spine, comparison rows, endpoint row, provenance, and local name.*

10. *P is the* Pkg *provenance over* $\mathsf{JensenInequality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{RealMinMax}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JensenInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, A, X, Y, R, Q, H, C, P, N$. It has no coordinate for a host convex function, quotient equality of functions or real names, a selected supremum, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.JensenInequalityUp.JensenInequalityTasteGate single carrier alignment`

Theorem 2235.2 (Jensen inequality NameCert obligations)**.** *For every accepted* $\mathsf{JensenInequality}^{\uparrow}$ *carrier $J = (F, W, A, X, Y, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{JensenInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the convex-function row, affine-weight row, affine-spine row, input-family row, image row, real-comparison row, endpoint row, structural rows H, C, P, and local naming row N; nonnegative rational coefficient accountability through W; convex-set affine-spine accountability through A; real comparison exactness through R, Q; and componentwise transport, replay, provenance, and naming for the same route.*

Proof. Project the accepted packet $J = (F, W, A, X, Y, R, Q, H, C, P, N)$. The carrier definition exposes exactly the convex-function source, affine weights, affine spine, input family, image family, comparison, endpoint, transport, replay, provenance, and naming rows.

The coefficient and affine-spine rows are read before the inequality row. Thus W supplies the nonnegative rational coefficients and finite sum row, while A supplies the displayed $\mathsf{ConvexSet}^{\uparrow}$ route that keeps the affine point in the accepted carrier.

The image and comparison rows then record the Jensen surface. The rows Y, R, Q carry the function images, real comparison, and endpoint comparison data, and H, C, P, N preserve and name that route. Since the excluded host convex-function and quotient-equality objects are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2235.3 (Jensen inequality affine-spine route)**.** *Every accepted* $\mathsf{JensenInequality}^{\uparrow}$ *carrier routes its displayed inequality through*

$$W, X \longmapsto A \longmapsto F, Y \longmapsto R, Q \longmapsto H, C, P, N.$$

Consequently a consumer must read the finite rational weights, input family, $\mathsf{ConvexSet}^{\uparrow}$ *affine-spine data, convex-function source row, finite image row,* $\mathsf{Real}^{\uparrow}$ *comparison row,* $\mathsf{RealMinMax}^{\uparrow}$ *endpoint row, componentwise transport, replay, provenance, and local naming before it can cite the Jensen inequality surface.*

Proof. Start with a consumer read of the inequality. By Definition 2235.1, the only affine-combination data are the coefficient row W, input-family row X, and affine-spine row A.

The convex-function and image rows are downstream of that affine route. The row F supplies the convexity ledger, and Y supplies the displayed image family over the same finite weights.

Finally the comparison rows R, Q expose the real inequality only after those source rows have been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route, so no consumer can bypass the affine spine by invoking an ambient convex-function theorem. □

Theorem 2235.4 (Jensen inequality public export)**.** *The public export of an accepted* $\mathsf{JensenInequality}^{\uparrow}$ *packet is exactly the finite row family* $F, W, A, X, Y, R, Q, H, C, P, N$ *and the dependency reads through* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{RealMinMax}^{\uparrow}$*. It exports no host convex function, quotient real equality, selected representative of a real value, open-cover compactness, countable choice, or ambient completeness principle.*

Proof. Use Definition 2235.2 to identify the local NameCert surface. It lists the carrier rows and structural rows as the complete obligation family.

Then use Definition 2235.3. It orders every consumer read through the rational weights, vector inputs, convex-set affine spine, convex-function source, image family, real comparison, endpoint comparison, transport, replay, provenance, and local naming rows.

Combining the obligation surface with the ordered route leaves precisely the listed public export. The absent host objects have no carrier coordinate and therefore cannot be exported by the packet. □

Falsifiable prediction: Every $\mathsf{JensenInequality}^{\uparrow}$ consumer reads a convex-function source row, finite rational nonnegative weights, a ConvexSet affine-spine row, vector input rows, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the Jensen inequality surface is public.

Independence witness: convexset, real, rat, vecspace, realminmax: $\mathsf{JensenInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries convex-function source, finite affine weights, affine-spine membership, image rows, comparison rows, endpoint rows, transport, replay, provenance, and local NameCert data as one finite inequality packet.

Closure status for $\mathsf{NameCert}_{\mathsf{JensenInequality}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JensenInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2235.1, Definition 2235.2, Definition 2235.3, and Definition 2235.4 seed the finite Jensen inequality route over convex-function, affine-weight, ConvexSet, VecSpace, Real, Rat, and RealMinMax rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JensenInequality}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.JensenInequalityUp.JensenInequalityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host convex functions, quotient equality of real names or functions, selected extrema, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite affine-weight accountability, ConvexSet affine-spine exactness, Real comparison nonescape, endpoint comparison handoff, and public export scope for BEDC.Derived.JensenInequalityU

Constructive story: A $\mathsf{JensenInequality}^{\uparrow}$ packet is generated from a convex-function source row, finite rational affine weights, a ConvexSet affine-spine row, VecSpace input rows, finite image rows, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2236

A Concrete Naming Certificate for $\mathsf{FiniteObservationFold}^{\uparrow}$

The $\mathsf{FiniteObservationFold}^{\uparrow}$ packet realizes the bounded MetaCIC checker-output surface of section 9379.4 as finite NameCert data. It sits beside the critical-path route of chapter 765, the bounded evaluator of chapter 818, the typed endpoint boundary of chapter 562, the consistency seal of chapter 740, and the export refusal gate of chapter 1319. The packet separates checked finite observations from any claim of global closed-term conversion decidability.

Definition 2236.1 (Finite observation fold carrier)**.** *A* $\mathsf{FiniteObservationFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, T, R, N, S, G, H, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source row, recording the current bounded decidability frontier and the open global-conversion boundary.*
2. *T is the* $\mathsf{FiniteTraceEvaluator}^{\uparrow}$ *observation row, recording the finite checker trace, bounded inputs, accepted checkpoint reads, and validation boundary.*
3. *R is the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *endpoint row, recording the typed normal-form boundary reached by the bounded observation.*
4. *N is the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *seal row, recording the closed-normal consistency data available at the displayed endpoint.*
5. *S is the finite fold row, recording the ordered aggregation of checker-output observations from T, R, N under the same critical-path source K.*
6. *G is the* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *refusal row, recording that the finite fold cannot be exported as global beta-conversion decidability or as a checked target outside the displayed bounded rows.*
7. *H records componentwise* hsame *transport for K, T, R, N, S, G and the local naming row.*
8. *C records displayed* Cont *replay from the critical-path source through finite trace observations, typed endpoint, consistency seal, fold row, refusal gate, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationFold}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{FiniteTraceEvaluator}^{\uparrow}$, $\mathsf{TypedReductionNormalForm}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. L is the local $\mathsf{NameCert}_{\mathsf{FiniteObservationFold}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, R, N, S, G, H, C, P, L$. It has no coordinate for unbounded evaluation, global conversion decidability, closed subject reduction, full consistency, host validation, or an exported checked target beyond the displayed finite observation fold.

Lean4 ✓ `BEDC.Derived.FiniteObservationFoldUp.FiniteObservationFoldTasteGate single carrier alignment`

Theorem 2236.2 (Finite observation fold NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationFold}^{\uparrow}$ *carrier $O = (K, T, R, N, S, G, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationFold}^{\uparrow}}$ *surface consists exactly of carrier admission for the critical-path source, finite trace observation, typed normal-form endpoint, normal-form consistency seal, finite fold row, cannot-claim export gate, structural rows H, C, P, and local naming row L; observation exactness through T; endpoint accountability through R, N; finite-fold ordering through S; export refusal through G; and componentwise transport, replay, provenance, and naming for the same bounded route.*

Proof. Project the accepted packet $O = (K, T, R, N, S, G, H, C, P, L)$. The carrier definition exposes exactly the critical-path, trace, endpoint, seal, fold, refusal, transport, replay, provenance, and naming rows.

The finite observation rows are read before any fold is public. The trace row T records bounded checker output, while R and N record the typed endpoint and consistency seal available for that bounded output.

The fold row S aggregates only those displayed observations, and the refusal row G blocks export as a stronger global decision procedure. Since the excluded unbounded and global claims are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2236.3 (Finite observation fold vision handoff)**.** *Every* $\mathsf{FiniteObservationFold}^{\uparrow}$ *consumer receives the route*

$$K \longmapsto T \longmapsto R, N \longmapsto S \longmapsto G \longmapsto H, C, P, L.$$

Thus the MetaCIC decidability reading is a finite checker-output packet over bounded observations, typed normal-form endpoints, normal-form consistency seals, refusal export rows, componentwise transport, displayed replay, provenance, and local naming. It is not a global closed-term beta-conversion decider.

Proof. Start from a consumer of the fold. The critical-path row K fixes the frontier and keeps the global conversion boundary visible.

The finite trace row T supplies the bounded observations. The endpoint row R and seal row N are read only as data attached to those bounded observations, not as an unbounded normalisation theorem.

The fold row S then aggregates the displayed observations, and the export gate G refuses any checked-target reading stronger than that finite packet. The structural rows H, C, P, L preserve this route under transport, replay, provenance, and naming. □

Falsifiable prediction: Every $\mathsf{FiniteObservationFold}^{\uparrow}$ consumer reads MetaCICCriticalPath, FiniteTraceEvaluator, TypedReductionNormalForm, NormalFormConsistencySeal, CannotClaimExportGate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any bounded checker-output packet is public.

Independence witness: MetaCICCriticalPath, FiniteTraceEvaluator, TypedReductionNormalForm, NormalFormConsistencySeal, CannotClaimExportGate: none is bijective to FiniteObservationFoldUp because the fold jointly carries critical-path source, finite trace observation, typed endpoint, normal-form consistency seal, bounded aggregation, export refusal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationFold}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationFold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2236.1, Definition 2236.2, and Definition 2236.3 seed the finite bounded-observation fold over MetaCIC critical-path, finite-trace, typed-normal-form, normal-form-consistency, and cannot-claim export rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationFold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global beta-conversion decidability, unbounded evaluation, closed subject reduction, full MetaCIC consistency, host validation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite observation exactness, typed-endpoint accountability, consistency-seal handoff, fold ordering, export refusal, and route preservation for BEDC.Derived.FiniteObservationFoldUp.

Constructive story: A $\mathsf{FiniteObservationFold}^{\uparrow}$ packet is generated from a MetaCICCriticalPath source row, a FiniteTraceEvaluator observation row, a TypedReductionNormalForm endpoint row, a NormalFormConsistencySeal row, a finite fold row, a CannotClaimExportGate refusal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2237

A Concrete Naming Certificate for $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley separated-reflection route for uniform and metric completions, the $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$ packet records the traditional passage from pre-separated pseudometric evidence to a separated metric completion as finite NameCert data. It starts from the pre-separated distance surface of chapter 1614, routes zero-distance evidence through the separated classifier of chapter 2004, and hands the resulting separated metric surface to the completion boundary of chapter 1506. Its numerical readback uses the finite windows of chapter 126, the regular rational route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet names the L10 completion bridge without quotient primitives, host equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2237.1 (SeparatedPseudometricCompletion carrier)**.** *A* $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P, Z, Q, M, S, R, D, E, H, C, K, N)$$

with the following displayed rows:

1. *P is the source* $\mathsf{Pseudometric}^{\uparrow}$ *packet. It supplies point admission, displayed distance witnesses, dyadic tolerance rows, stream windows, regular rational readback, a real-distance seal, and zero-distance boundary data before separation.*
2. *Z is the displayed zero-distance row. It records the finite evidence consumed by the separated classifier and does not identify point rows by host equality.*
3. *Q is the* $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ *classifier output. It is the only row that turns the zero-distance evidence in Z into separated metric point-classifier evidence.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion handoff. It consumes the separated metric surface exposed by Q and supplies the completion-facing route without adding quotient completion equality.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before any regular rational or real-distance readback is inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger used by the source pseudometric and by the completion-facing readback.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D and the separated classifier route have been displayed.*
9. *H records componentwise* hsame *transport for the pseudometric, zero-distance, separated classifier, completion handoff, finite-window, regular rational, dyadic, real-seal, continuation, package, and naming rows.*
10. *C records displayed* Cont *replay from a completion request through $P, Z, Q, M, S, R, D, E, H, K, N$ in that order.*
11. *K is the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedQuotientMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedPseudometricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient primitives, host equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SeparatedPseudometricCompletionUp.SeparatedPseudometricCompletionTasteGate`
`single carrier alignment`

Theorem 2237.2 (SeparatedPseudometricCompletion NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$ *carrier $A = (P, Z, Q, M, S, R, D, E, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedPseud}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source* $\mathsf{Pseudometric}^{\uparrow}$ *packet, zero-distance row,* $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ *classifier output,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *zero-distance separation, requiring every point-identity read from Z to pass through the separated classifier output Q before a metric or completion consumer may use it;*
3. *completion handoff, requiring M to consume the separated metric surface from Q rather than the pre-separated pseudometric row P directly;*
4. *numerical readback ordering, requiring S, R, D, E to be displayed in order before any completion-facing real-distance seal is exported;*
5. *transport and replay stability, requiring H and C to preserve the same route under componentwise* hsame *transport and displayed* Cont *replay;*
6. *provenance exactness, requiring K and N to name exactly the displayed BEDC dependencies and the local certificate row;*
7. *non-escape, excluding quotient primitives, host equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No separated pseudometric completion NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2237.1. The tuple lists the source pseudometric packet, zero-distance evidence, separated classifier output, completion handoff, finite windows, regular rational readback, dyadic ledger, real seal, transport, replay, provenance, and local naming rows.

Zero-distance separation follows from the order P, Z, Q. The source pseudometric row exposes zero-distance evidence, the row Z records the displayed evidence, and the row Q is the first coordinate that can export separated point-classifier evidence.

The completion handoff is then forced through M. Since M is downstream of Q, a completion-facing consumer cannot read point identity directly from the pre-separated pseudometric row.

Numerical readback is the displayed route S, R, D, E. Finite windows are opened first, regular rational data and dyadic comparisons are read next, and the terminal real seal is reached only after those rows have been displayed.

Transport and replay stability are supplied by H and C. They move the listed rows componentwise and replay the same finite route for consumers, so the certificate remains inside the displayed dependency surface.

The provenance and non-escape clauses are inventory checks on K, N and on the fields of A. None of the excluded principles occurs among the carrier coordinates, hence none can be part of the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.SeparatedPseudometricCompletionUp.SeparatedPseudometricCompletionCarrier namecert obligations`

Theorem 2237.3 (SeparatedPseudometricCompletion completion route)**.** *Let $A = (P, Z, Q, M, S, R, D, E, H, C, K$ be an accepted* $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$ *packet. Every completion-facing consumer factors through the finite route*

$$\mathsf{Pseudometric}^{\uparrow} \to Z \to \mathsf{SeparatedQuotientMetric}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicR}$$

Thus the consumer receives a separated metric completion handoff only after the zero-distance evidence has passed through the separated classifier and the numerical readback rows have been displayed. The packet exports no direct route from pre-separated pseudometric evidence to completion equality, host equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Start from an accepted packet A. The source row P supplies pre-separated distance evidence, and the row Z records the zero-distance boundary as displayed data.

The replay row C sends that evidence through Q before the completion handoff M can be read. This makes the separated classifier output the only route from zero distance to the metric point surface consumed by completion.

After M, a completion-facing numerical request reads the windows S, the regular rational row R, the dyadic ledger D, and finally the real seal E. The transport row H and the provenance rows K, N name the same displayed chain.

The excluded objects are absent from the carrier. There is no quotient primitive, host equality field, selected representative, selected limit, countable-choice schedule, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness row, so no consumer can obtain one from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedPseudometricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$ packet is generated from a source $\mathsf{Pseudometric}^{\uparrow}$ packet, displayed zero-distance evidence, a $\mathsf{SeparatedQuotientMetric}^{\uparrow}$ classifier output, a $\mathsf{MetricCompletion}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedPseudometricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2237.1, Definition 2237.2, and Definition 2237.3 seed the L10 route from pre-separated pseudometric evidence through separated classification into metric completion.

Formal verification: FormalStatus(NameCert$_{\text{SeparatedPseudometricCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient primitives, host equality, selected representatives, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance separation, completion handoff, numerical readback ordering, transport and replay stability, provenance exactness, and non-escape for BEDC.Derived.SeparatedPseudometricCo

Chapter 2238

A Concrete Naming Certificate for $\mathsf{CauchyProductCommutativity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyProductCommutativity}^{\uparrow}$ packet names the finite source-window symmetry behind Cauchy-product commutativity before any completed-real multiplication claim is available. It sits over the product packet of chapter 464, the regular product route of chapter 461, the finite stream-window surface of chapter 126, the dyadic product ledger of chapter 291, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet records commutativity as a finite displayed swap of product-source rows; it does not import host multiplication, quotient streams, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2238.1 (CauchyProductCommutativity carrier)**.** *A* $\mathsf{CauchyProductCommutativity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P_{xy}, P_{yx}, W_x, W_y, D_{xy}, D_{yx}, R_{xy}, R_{yx}, E_{xy}, E_{yx}, H, C, Q, N)$$

with the following displayed rows:

1. P_{xy} *and* P_{yx} *are the original and swapped* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows.*
2. W_x *and* W_y *are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows used by both source orders.*
3. D_{xy} *and* D_{yx} *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-observation rows, one for each order of the displayed dyadic inputs.*
4. R_{xy} *and* R_{yx} *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows reached from the two dyadic product observations.*
5. E_{xy} *and* E_{yx} *are the* $\mathsf{Real}^{\uparrow}$ *seal rows exposed only after the paired windows, dyadic product rows, and regular handoffs have been displayed.*
6. H *records componentwise* hsame *transport for the two source rows, paired windows, dyadic product rows, regular handoffs, Real seals, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from every product-commutativity request through* $P_{xy}, P_{yx}, W_x, W_y, D_{xy}, D_{yx}, R_{x}$
8. Q *is the* Pkg *provenance over* $\mathsf{CauchyProductCommutativity}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCommutativity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for host multiplication, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 2238.2 (CauchyProductCommutativity NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductCommutativity}^{\uparrow}$ *carrier*

$$K = (P_{xy}, P_{yx}, W_x, W_y, D_{xy}, D_{yx}, R_{xy}, R_{yx}, E_{xy}, E_{yx}, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCommutativity}^{\uparrow}}$ *surface consists exactly of carrier admission for the original and swapped* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *product observations, two* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, two* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is read through the sibling routes named in chapter 464, chapter 461, chapter 126, chapter 291, chapter 301, and chapter 38; no obligation is stored in host multiplication or completed-real algebra.*

Proof. Project the accepted carrier. The two product-source rows P_{xy} and P_{yx} expose the same two stream windows in opposite order, and the carrier definition leaves no third source row for a product consumer.

Read the dyadic product observations. The rows D_{xy} and D_{yx} are the only points at which the two window observations are multiplied, and their symmetry is the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ product-row symmetry rather than a host multiplication law.

Pass to the regular and real-facing rows. The handoffs R_{xy}, R_{yx} and seals E_{xy}, E_{yx} are downstream of the same windows and dyadic observations. The structural rows H, C, Q, N transport, replay, source, and name this finite packet, so the listed obligations exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.CauchyProductCommutativityUp.CauchyProductCommutativityCarrier namecert obligations`

Theorem 2238.3 (CauchyProductCommutativity window swap)**.** *Swapping the paired* $\mathsf{StreamName}^{\uparrow}$ *windows of an accepted* $\mathsf{CauchyProductCommutativity}^{\uparrow}$ *carrier swaps the two* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows and the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *product observations while preserving the same finite route to the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, the* $\mathsf{Real}^{\uparrow}$ *seal rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.CauchyProductCommutativityUp.CauchyProductCommutativity window swap`

Proof. The carrier stores the two windows as the shared source of both product orders. Exchanging W_x and W_y therefore exchanges P_{xy} with P_{yx} and D_{xy} with D_{yx}.

The downstream rows are componentwise over the displayed source order. The regular handoffs and Real seals are read after the dyadic product rows, and H, C, Q, N preserve the same dependency route after the exchange.

No host multiplication or quotient-stream equality is used in the swap. The statement is only the finite window exchange already visible inside the packet. □

Theorem 2238.4 (CauchyProductCommutativity Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProductCommutativity}^{\uparrow}$ *packet factors through the original and swapped* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-observation rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows before either seal row* E_{xy} *or* E_{yx} *is exposed. Thus the packet gives a non-quotient commutativity certificate for later Real algebra routes while excluding host multiplication, quotient streams, selected limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a seal-facing read. By the carrier definition, the only seal rows are E_{xy} and E_{yx}, and both are downstream of the displayed product-source rows, windows, dyadic product observations, and regular handoffs.

Use the window-swap theorem to identify the two routes as opposite readings of the same finite source-window packet. The structural rows then replay and name the same dependency route for any Real-facing consumer.

Because the carrier exposes no host multiplication, quotient-stream product, selected limit, trichotomy row, or ambient completeness coordinate, the Real seal boundary cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCommutativity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCommutativity}^{\uparrow}$ packet is generated from original and swapped $\mathsf{CauchyProduct}^{\uparrow}$ source rows, paired $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ product-observation rows, $\mathsf{RegSeqRat}^{\uparrow}$ classifier rows, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCommutativity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2238.1, Definition 2238.2, Definition 2238.3, and Definition 2238.4 seed the finite source-window commutativity certificate for the Cauchy product route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCommutativity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host multiplication, quotient streams, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window swapping, dyadic product-row symmetry, regular handoff preservation, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCommutativityUp.

Chapter 2239

A Concrete Naming Certificate for $\mathsf{CauchyProductAssociativity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyProductAssociativity}^{\uparrow}$ packet records the finite threefold source-window route by which two parenthesized Cauchy-product readings expose the same regular-rational product boundary before any completed-real multiplication law is available. It sits after the product-modulus route of **??**, the bounded-product route of chapter 1529, the source-swap route of chapter 2238, the regular multiplication route of chapter 1702, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names associativity as displayed BEDC route data over finite windows and dyadic product ledgers; it does not import host multiplication, quotient streams, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2239.1 (Cauchy product associativity carrier)**.** *A* $\mathsf{CauchyProductAssociativity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (P_{(xy)z}, P_{x(yz)}, W_x, W_y, W_z, M_{xy}, M_{yz}, B_{xy}, B_{yz}, D_{(xy)z}, D_{x(yz)}, R_{(xy)z}, R_{x(yz)}, E_{(xy)z}, E_{x(yz)}, H, C, Q, N)$$

with the following displayed rows:

1. $P_{(xy)z}$ *and* $P_{x(yz)}$ *are the two parenthesized* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows.*
2. W_x, W_y, W_z *are the three* $\mathsf{StreamName}^{\uparrow}$ *finite source-window rows shared by both parenthesizations.*
3. M_{xy} *and* M_{yz} *are the* $\mathsf{CauchyProductModulus}^{\uparrow}$ *rows that allocate the two adjacent product budgets before the outer product row is read.*
4. B_{xy} *and* B_{yz} *are the* $\mathsf{CauchyProductBounded}^{\uparrow}$ *rows that keep the adjacent intermediate product reads inside displayed finite magnitude ledgers.*
5. $D_{(xy)z}$ *and* $D_{x(yz)}$ *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *threefold product ledgers for the left- and right-associated product observations.*
6. $R_{(xy)z}$ *and* $R_{x(yz)}$ *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows reached from the two threefold dyadic product ledgers.*
7. $E_{(xy)z}$ *and* $E_{x(yz)}$ *are the* $\mathsf{Real}^{\uparrow}$ *seal rows exposed only after the shared windows, adjacent modulus rows, bounded-product rows, dyadic ledgers, and regular-rational handoffs have been displayed.*

8. *H records componentwise* hsame *transport for the two parenthesized product rows, shared windows, modulus rows, bounded rows, dyadic ledgers, regular handoffs, Real seals, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from every associativity-facing request through the three shared windows, the adjacent product budgets, the bounded-product ledgers, the two dyadic threefold ledgers, and the two regular and Real-facing handoffs.*
10. *Q is the* Pkg *provenance over* $\mathsf{CauchyProductAssociativity}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductModulus}^{\uparrow}$, $\mathsf{CauchyProductBounded}^{\uparrow}$, $\mathsf{CauchyProductCommutativity}^{\uparrow}$, $\mathsf{RegularCauchyMultiplication}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductAssociativity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. The carrier has no coordinate for host multiplication, quotient stream equality, selected product limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyProductAssociativityUp.CauchyProductAssociativityTasteGate single carrier alignment`

Theorem 2239.2 (Cauchy product associativity NameCert obligations)**.** *For every accepted* CauchyProductAssocia *carrier*

$$A = (P_{(xy)z}, P_{x(yz)}, W_x, W_y, W_z, M_{xy}, M_{yz}, B_{xy}, B_{yz}, D_{(xy)z}, D_{x(yz)}, R_{(xy)z}, R_{x(yz)}, E_{(xy)z}, E_{x(yz)}, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductAssociativity}^{\uparrow}}$ *surface consists exactly of carrier admission for the two parenthesized* $\mathsf{CauchyProduct}^{\uparrow}$ *rows, the three shared* $\mathsf{StreamName}^{\uparrow}$ *source windows, adjacent* CauchyProductModulus *budgets, adjacent* $\mathsf{CauchyProductBounded}^{\uparrow}$ *ledgers, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *threefold product ledgers, two* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, two* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is read through the sibling routes named in the opening dependency paragraph; no obligation is stored in host multiplication or completed-real algebra.*

Proof. Project the accepted carrier. The only source windows are W_x, W_y, W_z, and both parenthesized product rows are downstream of those three windows.

Read the adjacent product-budget rows. The modulus rows M_{xy}, M_{yz} allocate the finite tolerances, while the bounded-product rows B_{xy}, B_{yz} keep the intermediate product observations inside displayed magnitude ledgers before either outer product read is exposed.

Now inspect the two dyadic ledgers. The rows $D_{(xy)z}$ and $D_{x(yz)}$ are the only coordinates where the three source windows become product observations, and the regular rows $R_{(xy)z}, R_{x(yz)}$ and seal rows $E_{(xy)z}, E_{x(yz)}$ are downstream of them.

Finally H, C, Q, N transport, replay, source, and name the same finite packet. Since no omitted row contains host multiplication, quotient stream equality, selected limits, trichotomy, choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 2239.3 (Cauchy product associativity threefold route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchyProductAssociativity}^{\uparrow}$ *carrier factors through the shared finite route*

$$W_x, W_y, W_z \longmapsto M_{xy}, M_{yz} \longmapsto B_{xy}, B_{yz} \longmapsto D_{(xy)z}, D_{x(yz)} \longmapsto R_{(xy)z}, R_{x(yz)}.$$

Thus the two parenthesized product reads consume the same three source windows, adjacent modulus budgets, and bounded-product ledgers before the regular-rational product boundary is public.

Proof. Begin with a regular-rational consumer. By Definition 2239.1, the only regular classifier rows are $R_{(xy)z}$ and $R_{x(yz)}$.

Those rows are not source data. Each is downstream of a threefold dyadic product ledger, and both dyadic ledgers are downstream of the same source windows together with the adjacent modulus and bounded-product rows.

The two parenthesized source rows therefore determine different readings of one displayed three-window route. No regular-rational consumer can bypass $W_x, W_y, W_z, M_{xy}, M_{yz}, B_{xy}, B_{yz}, D_{(xy)z}, D_{x(yz)}$ and still read either classifier row from this packet. □

Theorem 2239.4 (Cauchy product associativity Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{CauchyProductAssociativity}^{\uparrow}$ *carrier factors through the threefold route of Definition 2239.3 before either seal row* $E_{(xy)z}$ *or* $E_{x(yz)}$ *is exposed. Hence the packet supplies the certificate-level associativity surface needed by later* $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *multiplication consumers, while exporting no host real multiplication law, quotient stream equality, selected product limit, trichotomy, countable choice, or ambient completeness theorem.*

Proof. Start from a seal-facing read. The carrier exposes exactly the two seal rows $E_{(xy)z}$ and $E_{x(yz)}$, and each appears after the corresponding regular-rational classifier row.

Apply Definition 2239.3. Both classifier rows are downstream of the same three source windows, adjacent modulus budgets, bounded-product ledgers, and dyadic threefold product observations.

The structural rows H, C, Q, N preserve and name this displayed route. Since the packet has no coordinate for host multiplication, quotient streams, selected limits, trichotomy, choice, or ambient completeness, no Real-facing read can export any of those objects. □

Falsifiable prediction: Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer of a $\mathsf{CauchyProductAssociativity}^{\uparrow}$ packet factors through three shared source windows, adjacent CauchyProductModulus budgets, adjacent CauchyProductBounded ledgers, two dyadic threefold product rows, and the two regular-rational handoffs before either parenthesized product seal is consumed.

Independence witness: cauchyproductmodulus, cauchyproductbounded, cauchyproductcommutativity, regularcauchymultiplication, regseqrat, real: $\mathsf{CauchyProductAssociativity}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds three source windows, two adjacent product budgets, two bounded-product ledgers, two parenthesized product observations, two regular handoffs, two Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductAssociativity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductAssociativity}^{\uparrow}$ packet is generated from two parenthesized CauchyProduct rows, three shared StreamName windows, adjacent CauchyProductModulus budgets, adjacent CauchyProductBounded ledgers, two DyadicRatCore threefold product rows, two RegSeqRat classifier rows, two RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductAssociativity}^{\uparrow}}$, seed)

Scope closed: Definition 2239.1, Definition 2239.2, Definition 2239.3, and Definition 2239.4 seed the finite threefold product associativity packet for the regular-Cauchy product route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductAssociativity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real multiplication, quotient stream equality, selected product limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjacent product-budget handoff, bounded-product ledger preservation, threefold dyadic product routing, regular handoff preservation, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductAssociativityUp.

Chapter 2240

A Concrete Naming Certificate for $\mathsf{FiniteWindowPullback}^{\uparrow}$

The $\mathsf{FiniteWindowPullback}^{\uparrow}$ packet realizes the transported stationary diagonal seal route of Definition 9238.18 as finite NameCert data. It pulls each $\mathsf{Real}^{\uparrow}$ finite-window seal read back through an explicitly displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row. The packet is designed as the concrete object later Lean work can formalize before TasteGate.lean supplies the required Nontrivial and FieldFaithful instances.

The pullback route is a joint consumer of the finite-window carrier of chapter 126, the regular readback of chapter 301, the dyadic ledger of chapter 291, and the seal row of chapter 38. Its carrier stores those sibling surfaces together so a seal-facing request cannot be treated as an isolated packet detached from the L10 source chain.

Definition 2240.1 (FiniteWindowPullback carrier)**.** *A* $\mathsf{FiniteWindowPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by the seal-facing request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed after the displayed stream window.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger attached to the same finite window.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row whose public read is pulled back to S, R, D.*
5. *H records componentwise* hsame *transport for the window, readback, ledger, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a seal-facing precision request through S, R, D, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowPullback}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, R, D, E, H, C, P, N. It has no coordinate for a detached Real value, quotient Cauchy class, selected limit, trichotomy, ambient completeness witness, Lean instance, or bridge check.

Lean4 ✓ `BEDC.Derived.FiniteWindowPullbackUp.FiniteWindowPullbackTasteGate single carrier alignment`

Theorem 2240.2 (FiniteWindowPullback NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowPullback}^{\uparrow}$ *carrier* $F = (S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every* $\mathsf{Real}^{\uparrow}$ *finite-window seal read admitted by the packet factors through exactly one displayed pullback packet* F*.*

Proof. Project the carrier. The only analytic rows are the displayed window S, readback R, dyadic ledger D, and seal row E.

The seal-facing read is downstream of those rows by construction. If a second admitted pullback packet exposes the same public seal read with the same window, readback, and dyadic ledger, constructor inversion on the finite BHist packet identifies its structural rows with H, C, P, N.

Thus each admitted finite-window seal read factors through one displayed packet, and no detached Real value, quotient class, selected limit, trichotomy row, ambient completeness witness, Lean instance, or bridge check can enter the local NameCert surface. □

Theorem 2240.3 (FiniteWindowPullback stationary diagonal route)**.** *The transported stationary diagonal seal exhaustion of Definition 9238.18 is realized by a* $\mathsf{FiniteWindowPullback}^{\uparrow}$ *packet whenever its public finite-window seal read is restricted to the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{Real}^{\uparrow}$ *seal rows. The route preserves the same* hsame*,* Cont*,* Pkg*, and local* NameCert *rows after pullback.*

Proof. Begin with the transported stationary diagonal seal row. The cited exhaustion theorem says that the public read is already determined by the finite stream window, regular readback, dyadic observation ledger, and Real seal.

Package those four readings as S, R, D, E. The structural rows H, C, P, N record transport, replay, provenance, and local naming without adding another analytic coordinate.

The resulting packet is exactly a $\mathsf{FiniteWindowPullback}^{\uparrow}$ carrier, so the stationary diagonal route has been pulled back to finite BEDC data. □

Theorem 2240.4 (FiniteWindowPullback StreamName route)**.** *Every accepted* $\mathsf{FiniteWindowPullback}^{\uparrow}$ *seal-facing read must pass through chapter 126, chapter 301, chapter 291, and chapter 38 before the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}$ *row is read. Equivalently, the read order exposes the* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* E *before the transport, replay, provenance, and local naming rows* H, C, P, N*.*

Proof. Project the finite carrier $F = (S, R, D, E, H, C, P, N)$ of Definition 2240.1. The first four rows are precisely the sibling surfaces for the stream window, regular readback, dyadic ledger, and Real seal.

The remaining rows cannot supply an alternative analytic route. The row H only transports the displayed components, C only replays the displayed request, P only records provenance, and N only names the already assembled local packet.

Thus any accepted seal-facing read reaches the local $\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}$ row only after passing through the four sibling chapters and their displayed rows. □

Falsifiable prediction: Every $\mathsf{Real}^{\uparrow}$ finite-window seal read admitted by a $\mathsf{FiniteWindowPullback}^{\uparrow}$ packet factors through exactly one displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert pullback packet.

Independence witness: streamname, regseqrat, dyadicratcore, real, and cauchylimitseal: no listed sibling is bijective to $\mathsf{FiniteWindowPullback}^{\uparrow}$ because this packet jointly stores the finite window, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows for one pullback route.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowPullback}^{\uparrow}$ packet is generated from a $\mathsf{StreamName}^{\uparrow}$ finite-window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{DyadicRatCore}^{\uparrow}$ observation ledger, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2240.1, Definition 2240.2, and Definition 2240.3 seed the finite pullback packet for Real seal reads over the transported stationary diagonal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close detached Real values, quotient Cauchy classes, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniqueness of the displayed pullback packet, stationary diagonal seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowPullbackUp.

Chapter 2241

A Concrete Naming Certificate for $\mathsf{SequentialUniformCompletion}^{\uparrow}$

Following Kelley on uniform spaces and Bishop–Bridges on constructive sequence completion, the $\mathsf{SequentialUniformCompletion}^{\uparrow}$ packet records the finite interface by which a displayed sequential Cauchy-completion route is offered to uniform-completion consumers. It sits between the sequence route of chapter 1683, the uniform-completion consumer of chapter 1473, the metric-completion surface of chapter 1506, the Cauchy-filter branch of chapter 333, and the Cauchy-net branch of chapter 589. The packet names only a finite handoff surface: it does not quotient Cauchy sequences, choose representatives, invoke countable choice, use open-cover compactness, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2241.1 (Sequential uniform completion carrier)**.** *A* $\mathsf{SequentialUniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, F, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SequentialCompletion}^{\uparrow}$ *route. It records the finite stream schedule, regular readback, dyadic tolerance ledger, Cauchy criterion or limit-seal row, and terminal real-seal handoff already accepted by the sequence-completion packet.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row. It records that the sequential route is read as a metric-completion-facing surface before any uniform consumer inspects it.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch extracted from the same displayed sequence route. It records the filter-side neighborhood evidence visible to the uniform-completion surface.*
4. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *branch extracted from the same displayed sequence route. It records the directed-window evidence visible to the uniform-completion surface.*
5. *H records componentwise* hsame *transport for the sequential route, metric-completion row, filter branch, net branch, replay, provenance, and local naming row.*
6. *C records displayed* Cont *replay of the handoff*

$$S \longmapsto M, F, D \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{SequentialCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CauchyNet}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{SequentialUniformCompletion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion, representative choice, countable choice, open-cover compactness, selected uniform limits, host equality of completions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialUniformCompletionUp.SequentialUniformCompletionTasteGate single carrier alignment`

Theorem 2241.2 (Sequential uniform completion NameCert obligations)**.** *For every accepted* $\mathsf{SequentialUniformCompletion}^{\uparrow}$ *carrier* $U = (S, M, F, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialUniformCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, sequence-route exposure, metric-completion handoff, Cauchy-filter branch exposure, Cauchy-net branch exposure, componentwise transport, displayed replay, provenance, local naming, and non-escape of quotient completions, representative choice, countable choice, open-cover compactness, selected uniform limits, host completion equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2241.1. The first projected row is S, so every consumer begins with the same finite $\mathsf{SequentialCompletion}^{\uparrow}$ route rather than with a quotient completion or a selected representative.

The metric and uniform-facing rows are then read through M, F, D. The row M supplies the metric-completion handoff, while F and D expose the filter and net presentations consumed by $\mathsf{UniformCompletion}^{\uparrow}$.

Rows H, C, P, N transport, replay, source, and name that same packet. Since the carrier inventory contains no other coordinate, the excluded completion principles cannot be exported by an accepted read. □

Theorem 2241.3 (Sequential uniform completion branch handoff)**.** *Every uniform-completion-facing read of an accepted* $\mathsf{SequentialUniformCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto M, F, D \longmapsto H, C, P, N.$$

Thus the sequential completion surface is handed to the metric-completion, Cauchy-filter, and Cauchy-net consumers before any $\mathsf{UniformCompletion}^{\uparrow}$ *read can use it, and the handoff exports no quotient completion, representative choice, countable choice, open-cover compactness, selected uniform limit, host completion equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the accepted carrier $U = (S, M, F, D, H, C, P, N)$. The continuation row C makes S the entry point, so the route starts with the already displayed sequence-completion packet.

The next rows are M, F, D. They give the metric-completion surface and the two standard uniform-completion branches over the same sequential route, which prevents a consumer from replacing the route by a hidden quotient object or selected limiting point.

Finally, H, P, N preserve the same handoff under transport, provenance, and local naming. No additional coordinate remains from which a choice principle, compactness principle, completion equality, or ambient real completeness claim could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialUniformCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialUniformCompletion}^{\uparrow}$ packet is generated from one $\mathsf{SequentialCompletion}^{\uparrow}$ route, one $\mathsf{MetricCompletion}^{\uparrow}$ consumer row, one $\mathsf{CauchyFilter}^{\uparrow}$ branch, one $\mathsf{CauchyNet}^{\uparrow}$ branch, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2241.1, Definition 2241.2, and Definition 2241.3 seed the finite sequence-to-uniform completion handoff through sequential, metric, filter, and net completion rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SequentialUniformCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, representative choice, countable choice, open-cover compactness, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-route exposure, metric-completion handoff, filter and net branch exactness, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.SequentialUniformCompletionUp.

Chapter 2242

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$

Following Bourbaki and Kelley on metric completion and separated uniform spaces, the HausdorffCompletionUniversa packet records the separated universal-property surface for completion. It sits between the metric completion carrier of chapter 1506, the separated metric boundary of chapter 1411, the uniform completion route of chapter 1473, the extension-and-uniqueness packet of chapter 397, and the quotient-free separated reflection boundary of **??**. The packet names the finite handoff by which dense source data extend into a separated target without importing quotient completion, representative choice, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2242.1 (HausdorffCompletionUniversal carrier)**.** *A* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, Q, S, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{MetricCompletion}^{\uparrow}$ *row. It records the metric source, completion-facing seal, and finite request boundary consumed by the universal-property route.*
2. *Q is the completion seal row. It records the completed object as a displayed seal, not as a host quotient of Cauchy names.*
3. *S is the separated target row. It records the* $\mathsf{SeparatedMetric}^{\uparrow}$ *or Hausdorff-facing target exactness consumed by uniqueness.*
4. *D is the dense-source handoff row. It records the source inclusion and dense read through which maps out of the source are inspected.*
5. *E is the extension row. It records the finite* Cont *route that extends the dense-source map into the separated target.*
6. *L is the uniqueness row. It records the displayed separatedness comparison showing that two extensions with the same dense-source reads are* hsame*-identified.*
7. *H records componentwise* hsame *transport for the source metric, completion seal, separated target, dense handoff, extension, uniqueness, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$M, Q, D \longmapsto S, E, L \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* HausdorffCompletionUniversal$^{\uparrow}$, MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, HausdorffReflection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{HausdorffCompletionUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, S, D, E, L, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host equality of completed points, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUniversalUp.HausdorffCompletionUniversalTasteGate`
`single carrier alignment`

Theorem 2242.2 (HausdorffCompletionUniversal NameCert obligations)**.** *For an accepted* HausdorffCompletionUn *carrier $U = (M, Q, S, D, E, L, H, C, P, N)$, the local* NameCert$_{\text{HausdorffCompletionUniversal}^{\uparrow}}$ *surface consists exactly of carrier admission; separated target exactness through S; dense-source handoff through D; extension existence through E; uniqueness through the separatedness ledger L; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance and local naming through P, N; and non-escape from quotient completion, representative choice, host equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2242.1. The accepted packet exposes exactly the metric-completion source, completion seal, separated target, dense handoff, extension, uniqueness, transport, replay, provenance, and local naming rows.

Separated target exactness is the row S. A uniqueness read cannot inspect host equality of completed points; it can only read the separated target row and the displayed comparison ledger carried by L.

The extension surface is finite. The dense-source handoff D supplies the source reads, and the extension row E repacks those reads through a displayed Cont route into the separated target.

Transport and replay preserve the same packet. Rows H and C move only the displayed coordinates M, Q, S, D, E, L, P, N, while P and N record provenance and local naming for that route. Since no excluded host-side coordinate exists in the carrier, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted HausdorffCompletionUniversal$^{\uparrow}$ consumer reads a MetricCompletion source row, a completion seal row, a SeparatedMetric target row, a dense-source handoff row, an extension row, a uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before citing the separated universal property.

Independence witness: metriccompletion, separatedmetric, uniformcompletion, completionuniversalproperty, hausdorffreflection: HausdorffCompletionUniversal$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a completion seal, separated target exactness, dense-source handoff, extension, uniqueness, transport, replay, provenance, and local naming as one finite universal-property packet.

Closure status for NameCert$_{\text{HausdorffCompletionUniversal}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{HausdorffCompletionUniversal}^{\uparrow}}$, seed)

Scope closed: Definition 2242.1 and Definition 2242.2 seed the separated universal-property surface for completion over MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, HausdorffReflection$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{HausdorffCompletionUniversal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host equality of completed points, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated target exactness, dense-source handoff, extension existence, uniqueness through separatedness, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.HausdorffCompletionUniversalUp.

Constructive story: A HausdorffCompletionUniversal$^{\uparrow}$ packet is generated from a MetricCompletion source row, completion seal, separated target row, dense-source handoff, extension row, uniqueness row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2243

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$ packet records the finite route by which regular Cauchy filter data are refined through cofinal subsequence windows before a regular rational readback and terminal real seal are consumed. It sits between the filter surface of chapter 333, the cofinal extraction route of chapter 405, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The refinement is the filter-facing handoff consumed by chapter 1506 and chapter 1473; it is not a quotient completion principle, selected representative, countable-choice schedule, or ambient real-completeness claim.

Definition 2243.1 (Regular Cauchy filter refinement carrier)**.** *A* $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its finite filterbase requests and Cauchy pressure ledger.*
2. *S is the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *refinement row selecting the cofinal subsequence windows used to read the filter without changing its source surface.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed by S and consumed by the dyadic and regular rational readback.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the same windows, including the displayed comparison cells used by the regularity check.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the cofinal windows and dyadic tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, S, W, D, R have been displayed.*
7. *H records componentwise* hsame *transport for F, S, W, D, R, E and for the structural rows below.*
8. *C records displayed* Cont *replay from filter refinement to stream windows, from windows and dyadic cells to regular rational readback, and from readback to the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{CofinalSubsequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterRefinement}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for quotient filters, quotient completions, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterRefinementUp.RegularCauchyFilterRefinementTasteGate`
`single carrier alignment`

Theorem 2243.2 (Regular Cauchy filter refinement obligations)**.** *For every accepted* RegularCauchyFilterRefinemen *carrier* $Q = (F, S, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, classifier transport through* H, C, P, N*, cofinal filter refinement through* F *and* S*, dyadic-window readback through* W, D, R*, and* $\mathsf{Real}^{\uparrow}$ *non-escape through the seal row* E*. No obligation is discharged by quotient completion, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2243.1. The accepted packet lists exactly the filter source, cofinal refinement, stream windows, dyadic tolerance ledger, regular readback, Real seal, and structural rows.

Classifier transport is componentwise. The row H transports the displayed data, C replays the same refinement route, and P, N source and name that route without adding a second classifier surface.

Cofinal filter refinement is read through F and S. The source filter stays visible at F, while the cofinal subsequence row S selects the windows used by the refinement before any regular rational or Real-facing consumer is reached.

Dyadic-window readback is then forced through W, D, R. The stream row exposes finite windows, the dyadic row records their tolerance cells, and the regular rational row is the only readback coordinate downstream of them.

Finally E is the only Real-facing seal. Since the excluded quotient, representative, choice, host-equality, and ambient-completeness principles have no coordinate in the carrier, they cannot be exported by the local NameCert surface. □

Theorem 2243.3 (Regular Cauchy filter refinement handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$ *packet reads the filter refinement only through*

$$F, S, W, D, R, E, H, C, P, N.$$

The consumer may inspect the source $\mathsf{CauchyFilter}^{\uparrow}$ *row, the* $\mathsf{CofinalSubsequence}^{\uparrow}$ *refinement row, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It receives no quotient completion, selected representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the displayed carrier. The refinement route exposes F before any cofinal window is read, so completion-facing consumers do not begin from a quotient filter or selected representative.

Apply the cofinal row S and then the finite window row W. These rows align the filter request surface with the cofinal subsequence interface of chapter 405 before the readback reaches dyadic tolerance data.

The readback step uses D and R. The dyadic ledger supplies finite comparison cells, and the regular rational readback is the only numerical row exported before the seal E.

The completion consumers of chapter 1506 and chapter 1473 can therefore read only the displayed route and its structural rows H, C, P, N. No absent quotient, representative, choice, or ambient completeness coordinate can be recovered downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterRefinement}^{\uparrow}$ packet is generated from a CauchyFilter source row, CofinalSubsequence refinement row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2243.1, Definition 2243.2, and Definition 2243.3 seed the finite filter-refinement route from CauchyFilter data through cofinal windows and dyadic readback to completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, cofinal filter refinement, dyadic-window readback, RealUp non-escape, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyFilterRefinementUp.

Chapter 2244

A Concrete Naming Certificate for $\mathsf{RealSeriesRatioTest}^{\uparrow}$

Following d'Alembert and Cauchy's ratio-test tradition, the $\mathsf{RealSeriesRatioTest}^{\uparrow}$ packet records a finite convergence-test surface over the $\mathsf{RealSeries}^{\uparrow}$ corridor rather than a host limit of ratios. It consumes successive real-series term rows from chapter 1581, regular rational readback from chapter 301, finite stream windows from chapter 126, dyadic ratio-bound rows, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a displayed eventual-ratio bound for downstream Real and Completion consumers; it does not choose an infinite ratio limit, compare quotient real values, select a convergence witness by countable choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2244.1 (RealSeriesRatioTest carrier)**.** *A* $\mathsf{RealSeriesRatioTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing the inspected term stream and finite partial-sum windows.*
2. *T is the successive-term row, listing adjacent nonzero term windows a_n and a_{n+1} only through displayed real-name observations.*
3. *B is the dyadic ratio-bound row. It records a dyadic $q < 1$, a finite threshold, and the displayed comparison that every inspected tail ratio is bounded by q.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to read the adjacent terms and the bound comparisons.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the term windows, ratio-bound rows, and partial-sum windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, T, B, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the real-series source, adjacent-term row, dyadic bound, finite windows, regular readback, endpoint seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a consumer request through S, T, B, W, R and then to E, H, P, N.*

9. *P is the* Pkg *provenance over* $\mathsf{RealSeriesRatioTest}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *dyadic bound data,* BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSeriesRatioTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a host ratio limit, quotient equality of real-name streams, a selected infinite sum, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RealSeriesRatioTestUp.RealSeriesRatioTestTasteGate single carrier alignment`

Theorem 2244.2 (RealSeriesRatioTest NameCert obligations)**.** *For every accepted* $\mathsf{RealSeriesRatioTest}^{\uparrow}$ *carrier* $K = (S, T, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSeriesRatioTest}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure, successive-term window exactness, dyadic ratio-bound exactness,* $\mathsf{StreamName}^{\uparrow}$ *finite-window service,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every convergence-test consumer must read the finite route*

$$S, T, B, W \longmapsto R \longmapsto E$$

together with H, C, P, N before using the packet as ratio-test evidence.

Proof. Project the accepted carrier. The source row S provides only the displayed real-series terms and partial-sum windows already exposed by the $\mathsf{RealSeries}^{\uparrow}$ packet.

The successive-term row T and dyadic bound row B are the analytic core of the carrier. They name adjacent tail terms, the dyadic bound $q < 1$, and the finite comparisons visible in the selected windows; they do not name a limit of ratios.

The rows W and R supply the finite observation schedule and regular rational readback through which the comparisons are inspected. The endpoint row E can be read only after those finite rows have been displayed.

Finally H, C, P, N transport, replay, source, and locally name the same route. Since no omitted row carries quotient real equality, selected sums, countable choice, or ambient completeness, the listed obligations exhaust the local certificate surface. □

Falsifiable prediction: An accepted $\mathsf{RealSeriesRatioTest}^{\uparrow}$ packet that reaches a $\mathsf{Real}^{\uparrow}$ convergence seal without displaying adjacent term windows, one dyadic ratio bound below 1, finite $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback fails the carrier surface.

Independence witness: RealSeries, RegSeqRat, StreamName, Real, dyadic ratio-bound rows: none is bijective to $\mathsf{RealSeriesRatioTest}^{\uparrow}$, because the listed siblings separately supply source terms, finite windows, readback, or endpoint sealing while this packet binds them to one eventual-ratio-test route with transport, replay, provenance, and local naming.

The Lean TasteGate follow-up is `BEDC.Derived.RealSeriesRatioTestUp`, with explicit Nontrivial RealSeriesRat and FieldFaithful $\mathsf{RealSeriesRatioTest}^{\uparrow}$ instances preserving $S, T, B, W, R, E, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSeriesRatioTest}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSeriesRatioTest}^{\uparrow}$ packet is generated from a $\mathsf{RealSeries}^{\uparrow}$ source row, successive real-term windows, one dyadic ratio-bound row below one, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSeriesRatioTest}^{\uparrow}}$, seed)

Scope closed: Definition 2244.1 and Definition 2244.2 seed the finite ratio-test carrier and its NameCert obligation package over $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealSeriesRatioTest}\uparrow}$, theoremChecked)
Lean target: `BEDC.Derived.RealSeriesRatioTestUp.RealSeriesRatioTestTasteGate\_single\_carrier\_alignment`
Bridge status: none
Not claimed: This chapter does not claim host ratio limits, quotient real equality, selected infinite sums, countable choice, Real completeness, or bridge checking.
Upgrade path: A bridge route can relate the ten-row carrier gate to downstream ratio-test consumer obligations.

Chapter 2245

A Concrete Naming Certificate for $\mathsf{CodonTransitionMatrix}^{\uparrow}$

The $\mathsf{CodonTransitionMatrix}^{\uparrow}$ packet realizes the finite-state stop-boundary prototype of subsection 9462.28.24 as concrete BEDC NameCert data. It consumes the six-coordinate source-window carrier of chapter 2234, the stop-suffix surface of chapter 1374, finite state rows for scan, elongate, terminate, readthrough, repair, and mutate states, and local transition rows for UAA, UAG, UGA, and UGG. The packet records a standard stop-boundary transition matrix; it does not identify source-space equality with biological translation identity, choose a stochastic model, or collapse the tryptophan row into termination.

Definition 2245.1 (CodonTransitionMatrix carrier)**.** *A* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, Q, F, D, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the six-coordinate codon readback.*
2. *Q is the finite-state row listing scan, elongate, terminate, readthrough, repair, and mutate states as displayed classifier entries.*
3. *F is the stop-face row naming the local* UAA*,* UAG*,* UGA*, and* UGG *suffix reads inside the codon-window surface.*
4. *D is the deterministic transition row. It records*

$$\begin{aligned}
\delta(\mathsf{elongate}, \mathrm{UAA}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UAG}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UGA}) &= \mathsf{terminate},\\
\delta(\mathsf{elongate}, \mathrm{UGG}) &= \mathsf{elongate\text{-}Trp}.
\end{aligned}$$

5. *T is the transition-exactness row requiring each displayed stop-face transition to agree with D and requiring the* UGG *row to remain non-terminating.*
6. *H records componentwise* hsame *transport for source windows, state rows, stop-face rows, deterministic transition rows, exactness rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a codon-window request through W, Q, F, D, T and then to H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, *stop-face and codon-cube boundary rows, local stop-boundary rows,* BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CodonTransitionMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for stochastic readthrough probabilities, a global genetic-code table, amino-acid classifier identity, source-space quotient equality, or a biological inevitability claim.

Lean4 ✓ `BEDC.Derived.CodonTransitionMatrixUp.CodonTransitionMatrixTasteGate single carrier alignment`

Theorem 2245.2 (CodonTransitionMatrix NameCert obligations)**.** *For every accepted* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *carrier $M = (W, Q, F, D, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CodonTransitionMatrix}^{\uparrow}}$ *surface consists exactly of source-window admission through W, finite-state admission through Q, stop-face row admission through F, deterministic transition display through D, transition exactness through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. In particular, the three standard stop rows terminate while the* UGG *local suffix row remains an elongation row.*

Proof. Project the accepted carrier. The row W confines all codon reads to the six-coordinate source window, and Q supplies the only states that can be used by the matrix.

The stop-face row F selects the four local suffix reads named by the prototype. The deterministic row D then displays exactly the three terminating rows and the one UGG elongation row.

The exactness row T is the classifier check: a packet mapping UGG to terminate cannot satisfy the displayed matrix. The structural rows H, C, P, N transport, replay, source, and locally name the same finite route.

No stochastic probability, global table, amino-acid classifier, quotient source equality, or biological inevitability row appears in the carrier. Hence the listed rows exhaust the NameCert obligation surface. □

Theorem 2245.3 (CodonTransitionMatrix stop-boundary exactness)**.** *Every accepted* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *read factors through*

$$W, Q, F \longmapsto D \longmapsto T$$

together with H, C, P, N. Therefore a transition matrix that maps the UGG *local suffix row to* terminate *is outside the accepted stop-boundary carrier.*

Proof. Begin with a transition read. The source-window row W supplies the codon window, and the finite-state row Q supplies the state names; no transition can be inspected before both rows are displayed.

The stop-face row F narrows the read to the four local suffix rows. The deterministic row D then fixes the three stop transitions and the UGG elongation transition.

Finally T checks exactness against the displayed matrix. Since H, C, P, N only preserve the same route, changing the UGG row to terminate changes D and breaks T. □

Falsifiable prediction: A transition matrix that maps the UGG local suffix row to the terminate state fails the standard stop-boundary transition row.

Independence witness: CodonSixBitWindow, CodonCubeBoundary, CodonStopFace, CodonSixBitWindow, LocalStopBoundary-style rows: none is bijective to $\mathsf{CodonTransitionMatrix}^{\uparrow}$, because those siblings supply source windows, boundary vocabulary, stop-face rows, or local boundary style while this packet binds them to one deterministic state-transition matrix with exactness, transport, replay, provenance, and local naming.

The Lean TasteGate file for `BEDC.Derived.CodonTransitionMatrixUp` must provide explicit $\mathsf{Nontrivial\ CodonTransitionMatrix}^{\uparrow}$ and $\mathsf{FieldFaithful\ CodonTransitionMatrix}^{\uparrow}$ instances preserving $W, Q, F, D, T, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{CodonTransitionMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{CodonTransitionMatrix}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source-window row, finite state rows, stop-face suffix rows, deterministic transition rows for UAA, UAG, UGA, and UGG, a transition-exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CodonTransitionMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2245.1, Definition 2245.2, and Definition 2245.3 seed the finite stop-boundary transition matrix over codon windows and local stop-face transition rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CodonTransitionMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim stochastic readthrough modeling, global genetic-code table identity, amino-acid classifier identity, source-space quotient equality, biological inevitability, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the nine-row carrier and prove the deterministic transition exactness surface for BEDC.Derived.CodonTransitionMatrixUp.

Chapter 2246

A Concrete Naming Certificate for $\mathsf{MartingaleConvergence}^{\uparrow}$

Following Doob, the $\mathsf{MartingaleConvergence}^{\uparrow}$ packet names the finite convergence-preparation surface supplied by upcrossing control over displayed martingale windows. It sits after the probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, and Doob-upcrossing packets of chapter 184, chapter 185, chapter 188, chapter 190, chapter 344, chapter 383, chapter 1629, and chapter 1630. The chapter records the finite NameCert surface consumed by later convergence work; it does not identify paths up to almost-sure quotient equality or import measure completion.

2246.1 Finite Convergence Carrier

Definition 2246.1 (Martingale convergence finite packet)**.** *A* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (\Omega, X, E, M, T, O, U, D, S, R, L, A, H, K, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *row naming the probability context.*
2. X *is the* $\mathsf{RandomVar}^{\uparrow}$ *endpoint-family row for the displayed martingale window.*
3. E *is the* $\mathsf{CondExp}^{\uparrow}$ *row exposing the successor faces used by the martingale law.*
4. M *is the consumed* $\mathsf{Martingale}^{\uparrow}$ *finite adapted-sequence packet.*
5. T *is the* $\mathsf{StoppingTime}^{\uparrow}$ *finite index row for bounded stopping reads.*
6. O *is the* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window row.*
7. U *is the* $\mathsf{Upcrossing}^{\uparrow}$ *crossing-count ledger for fixed lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoints.*
8. D *is the* $\mathsf{DoobUpcrossing}^{\uparrow}$ *inequality row bounding the displayed crossing-count ledger.*
9. S *is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for finite endpoint prefixes.*
10. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational regularity row for displayed approximation windows.*
11. L *is the* $\mathsf{Real}^{\uparrow}$ *lower and upper boundary endpoint row, with the comparison row carried as data.*

12. *A is the convergence-admission ledger recording which finite windows, boundary pairs, rational tolerances, and upcrossing bounds are available to downstream consumers.*
13. *H records componentwise* hsame *transport for the probability, endpoint, conditional-expectation, martingale, stopping, upcrossing, schedule, rational, real-boundary, admission, provenance, and naming rows.*
14. *K records the displayed* Cont *routes from finite endpoint prefixes through bounded stopped windows, upcrossing counts, Doob bounds, and convergence-admission rows.*
15. *P is the* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, $\mathsf{Upcrossing}^{\uparrow}$, $\mathsf{DoobUpcrossing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
16. *N is the local* $\mathsf{NameCert}_{\mathsf{MartingaleConvergence}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through these displayed rows. In particular, it has no coordinate for almost-sure quotient equality, completed sigma-algebras, Brownian paths, unbounded stopping-time choice, or a bridge theorem.

Lean4 ✓ `BEDC.Derived.MartingaleConvergenceUp.MartingaleConvergenceTasteGate single carrier alignment`

2246.2 Certificate Obligations

Theorem 2246.2 (Martingale convergence NameCert obligations)**.** *For the carrier of Definition 2246.1, the local* $\mathsf{NameCert}_{\mathsf{MartingaleConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the rows*

$$\Omega, X, E, M, T, O, U, D, S, R, L, A, H, K, P, N,$$

martingale endpoint exposure through X, E, M, bounded stopped-window exposure through T, O, finite upcrossing and Doob-bound exposure through U, D, schedule and rational-approximation exposure through S, R, L, convergence-admission exactness through A, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.

Proof. Project the packet of Definition 2246.1. The carrier lists exactly the probability, endpoint, conditional-expectation, martingale, stopping, optional-stopping, upcrossing, Doob-bound, schedule, rational, real-boundary, admission, transport, replay, provenance, and naming rows.

The martingale-facing rows are X, E, M. They supply finite endpoint reads and conditional-expectation faces only through the displayed $\mathsf{Martingale}^{\uparrow}$ packet.

The stopping-facing rows are T, O. They expose bounded stopped windows and do not select an unbounded stopping-time family.

The crossing-facing rows are U, D. They consume the finite crossing-count ledger and the finite Doob-bound row, so the convergence packet does not recompute an inequality outside the accepted upcrossing certificates.

The remaining rows S, R, L, A, H, K, P, N record finite prefix scheduling, rational approximation, boundary endpoints, admission, transport, replay, provenance, and naming. Since the carrier contains no quotient-equality, measure-completion, Brownian-path, unbounded-choice, or bridge coordinate, those objects are outside the local certificate surface. □

Definition 2246.3 (Martingale convergence bounded real window)**.** *For an accepted* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier*

$$C = (\Omega, X, E, M, T, O, U, D, S, R, L, A, H, K, P, N),$$

the bounded real window is the finite endpoint packet displayed by L, S, R*: the row* L *names the lower and upper* $\mathsf{Real}^{\uparrow}$ *boundary endpoints, the row* S *names the finite prefix windows in which those endpoints are inspected, and the row* R *names the regular rational readbacks used to compare the endpoints with the martingale values. The window is bounded only when the same* L, S, R *rows are consumed before the upcrossing rows* U, D *and before the convergence-admission row* A*.*

Theorem 2246.4 (Martingale convergence upcrossing window lock)**.** *Every convergence-admission read of* A *in an accepted* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier first passes through the upcrossing rows* U, D *together with the bounded real window of Definition 2246.3, the bounded optional-stopping rows* T, O*, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. The lock supplies a finite convergence-preparation boundary over bounded endpoints and bounded stopping windows; it supplies no almost-sure quotient equality, no unbounded stopping theorem, no measure completion, and no ambient convergence theorem.*

Proof. Open the carrier of Definition 2246.1. The row A is the convergence-admission ledger, and its available inputs are exactly the displayed probability, endpoint, martingale, stopping, upcrossing, schedule, readback, boundary, transport, replay, provenance, and naming rows of the same packet.

Apply the NameCert obligation theorem, Definition 2246.2. It identifies bounded stopped-window exposure through T, O, finite upcrossing and Doob-bound exposure through U, D, schedule and rational-approximation exposure through S, R, L, and convergence-admission exactness through A.

The bounded real window is therefore not an external endpoint object. It is the same L, S, R packet that the upcrossing and Doob rows consume before A can admit a convergence-facing read.

Finally, the carrier contains no coordinate for almost-sure quotient equality, completed measure spaces, unbounded stopping-time choice, or ambient convergence. Hence the admission row can only lock the finite bounded-window route through $T, O, U, D, L, S, R, H, K, P, N$. □

Theorem 2246.5 (Martingale convergence upcrossing route)**.** *Every accepted* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier consumes upcrossing information only through the displayed finite handoff from* $\mathsf{Upcrossing}^{\uparrow}$ *and* $\mathsf{DoobUpcrossing}^{\uparrow}$*: lower and upper* $\mathsf{Real}^{\uparrow}$ *boundary rows, finite martingale endpoint rows, lower and upper decision ledgers, bounded* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window rows, crossing-count ledger, and the Doob-bound row. No convergence consumer may replace that handoff by an ambient limiting event, almost-sure quotient, or unbounded stopping theorem.*

Proof. Start with the accepted carrier and project U and D. The row U is the crossing-count ledger supplied by chapter 1629; the row D is the Doob-bound surface supplied by chapter 1630.

The boundary rows and endpoint reads are displayed in L and X, while the bounded stopped-window route is displayed in T and O. The replay row K records the finite path from endpoint prefixes to stopped windows, crossing counts, and Doob bounds.

Thus a downstream consumer receives exactly the finite upcrossing handoff rows carried by $U, D, L, X, T, O, H, K, P, N$. The carrier has no coordinate from which an ambient limiting event, almost-sure quotient, or unbounded stopping theorem could be read. □

Theorem 2246.6 (Martingale convergence finite Cauchy admission)**.** *For an accepted* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier, the convergence-admission row* A *may export a finite Cauchy-preparation ledger only after*

the bounded endpoint rows L, $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Upcrossing}^{\uparrow}$ *row U, and* $\mathsf{DoobUpcrossing}^{\uparrow}$ *row D have all been displayed. The exported ledger is a finite bounded-window admission route over the same martingale, stopping, optional-stopping, upcrossing, schedule, rational-readback, and Real-boundary rows; it does not claim almost-sure convergence, measure completion, unbounded stopping-time choice, Brownian convergence, or an ambient convergence theorem.*

Proof. Open the carrier of Definition 2246.1. The row A is the only convergence-admission ledger, and its available endpoint and approximation inputs are the displayed L, S, R rows of the same packet.

Apply Definition 2246.2. It identifies finite upcrossing and Doob-bound exposure through U, D, schedule and rational-approximation exposure through S, R, L, and convergence-admission exactness through A.

Use the bounded real window of Definition 2246.3 and the lock theorem Definition 2246.4. They force any admission read to pass through bounded endpoint rows, finite prefix scheduling, rational readback, and bounded upcrossing control before A is consumed.

Finally apply Definition 2246.5. The upcrossing input to the admission ledger is exactly the displayed $\mathsf{Upcrossing}^{\uparrow}$ and $\mathsf{DoobUpcrossing}^{\uparrow}$ handoff, so the exported Cauchy-preparation ledger remains finite and cannot add almost-sure convergence, measure completion, unbounded stopping-time choice, Brownian convergence, or an ambient convergence theorem. □

Theorem 2246.7 (Martingale convergence obligation closure handoff)**.** *For an accepted* $\mathsf{MartingaleConvergence}^{\uparrow}$ *carrier*

$$C = (\Omega, X, E, M, T, O, U, D, S, R, L, A, H, K, P, N),$$

any obligation-closure read of the convergence-admission ledger A must first expose the martingale endpoint rows X, E, M, the bounded stopping rows T, O, the upcrossing and Doob rows U, D, and the schedule, readback, and boundary rows S, R, L. The exported object is only a finite convergence-preparation ledger over those rows together with H, K, P, N; it does not claim almost-sure convergence, measure completion, unbounded stopping-time choice, Brownian convergence, or an ambient convergence theorem.

Proof. Open the accepted carrier and apply Definition 2246.2. The obligation surface for A is already bounded by the displayed endpoint, conditional-expectation, martingale, stopping, optional-stopping, upcrossing, Doob-bound, schedule, readback, real-boundary, transport, replay, provenance, and naming rows.

Use the bounded real window of Definition 2246.3. It identifies the boundary and approximation side as the same L, S, R rows that must be displayed before the upcrossing route can be consumed.

Apply Definition 2246.4. This forces any read of A through the bounded stopping rows T, O, the upcrossing rows U, D, and the schedule and readback rows S, R, L.

Finally use Definition 2246.6. The only export left after those exposures is the finite convergence-preparation ledger over the same displayed packet. Since no carrier row names almost-sure quotient equality, measure completion, unbounded stopping-time choice, Brownian convergence, or an ambient convergence theorem, the obligation handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{MartingaleConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MartingaleConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2246.1, Definition 2246.2, Definition 2246.3, Definition 2246.4, and Definition 2246.5 seed the finite convergence-preparation packet over martingale windows, bounded stopping, upcrossing ledgers, Doob bounds, finite prefix schedules, rational approximation rows, and Real boundary endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MartingaleConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close almost-sure quotient equality, measure completion, unbounded choice of stopping times, Brownian convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, martingale endpoint exposure, bounded stopped-window exposure, upcrossing handoff exactness, Doob-bound consumption, finite schedule accountability, rational approximation routing, and non-escape for BEDC.Derived.MartingaleConvergenceUp.

Constructive story: A MartingaleConvergence$^{\uparrow}$ packet is generated from probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, upcrossing, Doob-upcrossing, stream-name, regular-rational, Real-boundary, transport, continuation, provenance, and local NameCert rows.

Chapter 2247

A Concrete Naming Certificate for $\mathsf{ObservationResidualFit}^{\uparrow}$

The $\mathsf{ObservationResidualFit}^{\uparrow}$ packet realizes the residual surface between finite observational fit and open physical refutation in section 9414.13. It sits beside $\mathsf{OpenFitPacket}^{\uparrow}$ from chapter 1137, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$ from **??**, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$ from chapter 1124, FiniteObservationRefutat from chapter 1169, and the finite refutation boundary of chapter 508. Those siblings expose finite fit, physical-signature comparison, model audit, observed defeat, and refutation boundaries; this packet binds the observed-signature, model-signature, residual-row, refutation-surface, open-continuation, transport, replay, provenance, and local naming rows into one finite carrier.

2247.1 Residual Carrier

Definition 2247.1 (Observation residual fit carrier)**.** *An* $\mathsf{ObservationResidualFit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, M, R, E, L, T, C, H, P, N)$$

with the following displayed rows:

1. *O is the observed-signature row for the finite trace and observation bundle.*
2. *M is the model-signature row for the same finite trace and observation bundle.*
3. *R is the residual row recording the displayed mismatch budget between O and M.*
4. *E is the finite refutation-surface row, read through the local failure boundary when the residual separates the signatures.*
5. *L is the open-continuation ledger recording undefeated continuation status when the residual remains inside the admitted tolerance.*
6. *T records componentwise* hsame *transport for observed signatures, model signatures, residual rows, refutation surfaces, continuation ledgers, package provenance, and local naming.*
7. *C records displayed* Cont *replay from finite observation through model comparison, residual extraction, refutation-surface reading, open continuation, provenance, and local naming.*
8. *H is the handoff row to* $\mathsf{OpenFitPacket}^{\uparrow}$, $\mathsf{RealityConstrainedSignatureFit}^{\uparrow}$, $\mathsf{RealityConstrainedModelAudit}^{\uparrow}$, $\mathsf{FiniteObservationRefutation}^{\uparrow}$, *and the finite refutation boundary.*

9. *P is the* Pkg *provenance over* ObservationResidualFit$^{\uparrow}$, OpenFitPacket$^{\uparrow}$, RealityConstrainedSignatureFit$^{\uparrow}$, RealityConstrainedModelAudit$^{\uparrow}$, FiniteObservationRefutation$^{\uparrow}$, FiniteRefutationBoundary$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{ObservationResidualFit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, M, R, E, L, T, C, H, P, N$. It has no coordinate for final-fit oracles, global physical truth, host model adequacy, unbounded observation histories, or a hidden refutation outside the displayed residual row.

Lean4 ✓ `BEDC.Derived.ObservationResidualFitUp.ObservationResidualFitTasteGate single carrier alignment`

2247.2 Residual Obligations

Theorem 2247.2 (Observation residual fit NameCert obligations)**.** *For every accepted* ObservationResidualFit$^{\uparrow}$ *carrier $F = (O, M, R, E, L, T, C, H, P, N)$, the local* NameCert$_{\text{ObservationResidualFit}^{\uparrow}}$ *surface consists exactly of observed-signature exposure through O, model-signature exposure through M, residual exactness through R, finite refutation-surface exposure through E, open-continuation ledgering through L, componentwise transport through T, displayed replay through C, sibling handoff through H, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 2247.1. The rows O and M expose the finite observed and model-generated signatures over one observation bundle and one finite trace.

The residual row R is the only mismatch budget visible to the packet. If the residual separates the two signatures, the refutation surface E records the finite defeat route; otherwise the open-continuation ledger L records the admitted continuation status.

The rows T, C, H, P, N transport, replay, hand off, package, and name the same finite packet. Since no remaining coordinate stores a final-fit oracle, global physical truth predicate, host adequacy judgment, unbounded observation history, or detached refutation, the listed rows exhaust the local NameCert obligations. □

Theorem 2247.3 (Observation residual fit sibling route)**.** *Every public residual read from an accepted* ObservationResidualFit$^{\uparrow}$ *packet must factor through the finite-fit row of* OpenFitPacket$^{\uparrow}$*, the signature-comparison row of* RealityConstrainedSignatureFit$^{\uparrow}$*, the audit row of* RealityConstrainedModelAudit$^{\uparrow}$*, the observed-defeat row of* FiniteObservationRefutation$^{\uparrow}$*, and the finite refutation boundary when the residual is separating. A route that drops R, E, or L is not an accepted residual-fit route.*

Proof. Read the sibling handoff row H. It names the open-fit packet, signature-fit packet, model-audit packet, finite-observation-refutation packet, and finite refutation boundary as the only external surfaces consumed by the residual packet.

The residual row R is computed from the displayed signature rows O and M. The refutation row E and open-continuation row L are then mutually visible outcomes of that finite residual read: separation exposes a refutation surface, while admitted mismatch remains an open continuation.

The transport, replay, provenance, and naming rows T, C, P, N preserve the same route. Therefore any consumer that drops R, E, or L has left the accepted residual-fit packet, even if it still cites open fit, signature fit, model audit, observed defeat, or refutation language. □

Falsifiable prediction: Every accepted ObservationResidualFit$^{\uparrow}$ consumer displays observed-signature, model-signature, residual, finite-refutation, open-continuation, handoff, transport, replay, provenance, and local naming rows before any residual-fit read is public.

Independence witness: OpenFitPacket, RealityConstrainedSignatureFit, RealityConstrained-ModelAudit, FiniteObservationRefutation, FiniteRefutationBoundary: none is bijective to $\mathsf{ObservationResidualFit}^{\uparrow}$ because the residual packet jointly carries observed signature, model signature, mismatch budget, refutation surface, open continuation, handoff, transport, replay, provenance, and local NameCert rows.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ ObservationResidualFit}^{\uparrow}$ and $\mathsf{FieldFaithful\ ObservationResidua}$ instances preserving $O, M, R, E, L, T, C, H, P, N$ in `lean4/BEDC/Derived/ObservationResidualFitUp/TasteGat`

Closure status for $\mathsf{NameCert}_{\mathsf{ObservationResidualFit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservationResidualFit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2247.1, Definition 2247.2, and Definition 2247.3 seed the finite residual-fit packet over observed signatures, model signatures, residual budgets, finite refutation surfaces, open continuation ledgers, sibling handoffs, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservationResidualFit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close final-fit oracles, global physical truth, host model adequacy, unbounded observation histories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observed-signature exposure, model-signature exposure, residual exactness, refutation-surface handoff, open-continuation ledgering, and sibling-route preservation for BEDC.Derived.ObservationResidualFitUp.

Constructive story: An $\mathsf{ObservationResidualFit}^{\uparrow}$ packet is generated from finite observed-signature, model-signature, residual-row, refutation-surface, open-continuation, handoff, transport, replay, provenance, and local NameCert rows.

Chapter 2248

A Concrete Naming Certificate for $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$

Following Bishop and Bridges on constructive located intervals, the $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ packet records a finite dyadic partition of a displayed located $\mathsf{Real}^{\uparrow}$ interval as NameCert data. It sits after the located endpoint surface of chapter 394, the dyadic refinement surface of chapter 306, the rational interval cell ledger of chapter 354, the finite StreamName windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite mesh ledger for closed-bounded interval reads; it does not import open-cover compactness, a host interval partition, quotient real equality, selected endpoint values, countable choice, or ambient completeness.

Definition 2248.1 (Located real interval partition carrier)**.** *A* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (J, M, Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the source* $\mathsf{LocatedInterval}^{\uparrow}$ *row whose lower and upper endpoint names have already exposed located real endpoint windows.*
2. *M is the finite dyadic mesh row, listing the ordered mesh cells and their common precision.*
3. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *witness family assigning each mesh cell a rational containment row inside the located interval ledger.*
4. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement family attached to the rational cells of Q.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for the endpoint and mesh-cell reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and dyadic mesh precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the located endpoint, rational cell, dyadic mesh, window, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the located interval row, dyadic mesh cells, rational witnesses, dyadic refinements, stream windows, regular readbacks, real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a partition read through $J, M, Q, D, S, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed endpoint, mesh, rational-cell, dyadic-refinement, window, readback, seal, transport, replay, provenance, and local naming rows. It has no coordinate for a host partition object, an open-cover subcover, quotient real equality, selected completed endpoints, countable choice, or ambient compactness.

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalPartitionUp.LocatedRealIntervalPartitionTasteGate`
`single carrier alignment`

Theorem 2248.2 (Located real interval partition NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *carrier* $T = (J, M, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalPartition}}$ *surface consists exactly of carrier admission, located endpoint exposure through J, finite dyadic mesh exposure through M, rational interval witness exposure through Q, dyadic cell refinement through D, finite StreamName window exposure through S, RegSeqRat readback through R, terminal Real-seal exposure through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Every partition read of the located interval must therefore pass through the displayed finite mesh and rational witness rows before any StreamName, RegSeqRat, or Real seal row is consumed.

Proof. Project the carrier of Definition 2248.1. The tuple lists the only rows available to a partition consumer: the source located interval, mesh cells, rational witnesses, dyadic refinements, stream windows, regular readbacks, real seals, transport, replay, provenance, and local naming rows.

The located endpoint stage is J. By the located interval certificate of chapter 394, endpoint reads enter through displayed located real windows and not through chosen completed endpoint values.

The partition stage is M, Q, D. The dyadic mesh row gives the finite cell order and common precision, the rational interval witness family records containment for each cell, and the dyadic refinement family attaches the mesh cells to those rational witnesses.

The readback and seal stage is S, R, E. A consumer reaches the finite StreamName windows only after the mesh and rational witness rows are visible, then reads the regular rational rows and finally the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name exactly this route. Since no host partition object, open-cover subcover, quotient real equality, chosen endpoint, choice schedule, or compactness row occurs in the carrier, the displayed NameCert surface is exhaustive. □

Theorem 2248.3 (Located real interval partition mesh handoff)**.** *Every accepted* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *partition read hands its mesh to downstream closed-bounded interval consumers through the finite route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{RationalInterval}^{\uparrow}, \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes the located endpoint rows, dyadic mesh cells, rational containment witnesses, finite windows, regular readbacks, and terminal real seals, together with hsame, Cont, Pkg, *and local* NameCert *provenance. It exports no open-cover compactness principle, no countable-choice subnet, no quotient interval, and no ambient complete ordered field row.*

Proof. Start with an accepted carrier $T = (J, M, Q, D, S, R, E, H, C, P, N)$. The only source interval row is J, so every endpoint and containment read begins at the displayed located interval surface.

The mesh handoff is stored in M, Q, D. The dyadic mesh row M gives the finite partition cells, the rational witness family Q certifies their containment in the located interval ledger, and the dyadic refinement family D attaches the cells to the rational interval witnesses.

The downstream read is stored in S, R, E. StreamName windows are opened at the displayed mesh precision, RegSeqRat rows read back the same windows, and the Real seal is reached only after those rows have been displayed.

Finally H, C, P, N preserve the handoff under componentwise transport and replay. There is no additional coordinate in Definition 2248.1, so a closed-bounded interval consumer receives exactly the displayed finite mesh route and none of the excluded compactness, choice, quotient, or ambient-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalPartition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2248.1, Definition 2248.2, and Definition 2248.3 seed the finite dyadic partition packet for located Real interval reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient interval equality, selected endpoint values, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh-cell coverage, rational interval witnesses, dyadic refinement, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealIntervalPartitionUp.

Constructive story: A $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ packet is generated from a LocatedIntervalUp row, finite dyadic mesh cells, RationalIntervalUp witnesses, DyadicIntervalUp refinements, StreamNameUp windows, RegSeqRatUp readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2249

A Concrete Naming Certificate for $\mathsf{NestedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges and Cantor nested interval reasoning, the $\mathsf{NestedIntervalSelection}^{\uparrow}$ packet records the finite route by which a displayed nested interval chain supplies a regular Cauchy selection surface without selecting a completed point. It sits after the finite nested interval packet of chapter 320, the located interval boundary of chapter 394, the closed bounded interval endpoint carrier of chapter 1591, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names one L10 finite selection route; it does not import countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or an external real-point selector.

Definition 2249.1 (Nested interval selection carrier)**.** *A* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, L, B, W, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{NestedInterval}^{\uparrow}$ *finite chain whose visible intervals are being read.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row that records the locatedness tests available for each displayed endpoint window.*
3. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint row anchoring the lower and upper interval bounds.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite interval prefix inspected by the consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed endpoint windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording endpoint widths, midpoint tolerances, and the Cauchy budget requested of the selected row.*
7. *Q is the finite selector row saying that the selected regular-rational approximant is read from the displayed window W and ledger D.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, L, B, W, R, D, Q have been displayed.*
9. *H records componentwise* hsame *transport for all carrier rows.*
10. *C records displayed* Cont *replay of the route*

$$I, L, B \longmapsto W, D \longmapsto R, Q \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalSelection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalSelection}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite selection packet. The classifier compares accepted packets only by componentwise agreement of $I, L, B, W, R, D, Q, E, H, C, P, N$ under the displayed hsame *and* Cont *routes; it has no coordinate for a chosen intersection point, quotient stream equality, countable choice, trichotomy, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.NestedIntervalSelectionUp.NestedIntervalSelectionTasteGate single carrier alignment`

Theorem 2249.2 (Nested interval selection NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (I, L, B, W, R, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, nested-chain exposure through I, located endpoint exposure through L, closed bounded endpoint exposure through B, finite window exposure through W, regular rational readback through R, dyadic tolerance accounting through D, selector-row exactness through Q, real-seal handoff through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The projection exposes the nested interval source, located endpoint tests, closed bounded endpoint anchor, finite window, regular readback, dyadic tolerance ledger, selector row, real seal, transport, continuation, provenance, and local naming rows.

The analytic obligations are exactly the visible rows. The nested-chain row gives the finite interval source, the located and closed bounded rows give endpoint admissibility, the window and dyadic rows give the inspected prefix and tolerance budget, and the regular readback plus selector row give the selected approximant.

The structural obligations are H, C, P, N. They transport, replay, source, and name the same finite route. Since the carrier contains no chosen point, quotient real equality, countable-choice function, trichotomy row, or real-completeness theorem, no further local NameCert obligation can be read. □

Theorem 2249.3 (Nested interval selection Cauchy route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *packet reads the Cauchy-facing route*

$$\mathsf{NestedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

through the displayed rows $I, L, B, W, D, R, Q, E, H, C, P, N$. The selected regular-rational approximant is determined by the finite window and tolerance ledger, not by countable choice, quotient real equality, a selected nested-intersection point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source interval rows. The nested interval packet I supplies the displayed finite chain, while L and B provide located and closed bounded endpoint data attached to the same visible prefix.

Next read the Cauchy budget. The window row W chooses the inspected finite prefix, and the dyadic ledger D records the endpoint widths and tolerances used by the selector row Q. This is a finite BHist route, so the consumer receives no external sequence of choices.

Finally pass through the readback and seal rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies the regular-rational approximant, and E may be read as a $\mathsf{Real}^{\uparrow}$ seal only after I, L, B, W, D, R, Q are displayed. The rows H, C, P, N preserve and name that route without adding quotient equality or a completeness principle. □

Theorem 2249.4 (Nested interval selection non-escape)**.** *No accepted* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *packet exports a selected point of an infinite nested intersection, a quotient real equality proof, a countable-choice selector, trichotomy for endpoint comparison, an independent real carrier, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public read factors through the finite rows* $I, L, B, W, R, D, Q, E, H, C, P, N$ *of Definition 2249.1.*

Proof. Project the carrier and inspect its coordinates. The only analytic rows are the nested chain, located interval data, closed bounded endpoint data, finite observation window, regular rational readback, dyadic tolerance ledger, selector row, and real seal.

The selector row is finite: it is read from W and D, then passed to R and E. It never stores a function choosing an element from every nested chain and never identifies two real names by quotient equality.

The remaining rows are structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row can only preserve the displayed packet. Therefore any public read outside $I, L, B, W, R, D, Q, E, H, C, P, N$ is unavailable, and the listed host-level or completeness principles cannot escape from the certificate. □

Falsifiable prediction: Every $\mathsf{NestedIntervalSelection}^{\uparrow}$ read factors through a nested interval source, located endpoint row, closed bounded endpoint row, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, finite selector row, RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: nestedinterval, locatedinterval, closedboundedinterval, regseqrat, real: $\mathsf{NestedIntervalSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite nested-chain source, endpoint locatedness, closed bounded endpoint anchoring, a displayed window, regular readback, dyadic tolerance accounting, a selector row, seal handoff, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalSelection}^{\uparrow}$ packet is generated from a nested interval row, located interval row, closed bounded interval endpoint row, finite StreamName window, RegSeqRat readback row, DyadicRatCore tolerance ledger, finite selector row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2249.1, Definition 2249.2, Definition 2249.3, and Definition 2249.4 seed the finite nested-interval selection surface over NestedInterval, LocatedInterval, ClosedBoundedInterval, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient real equality, trichotomy, a selected nested-intersection point, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Cauchy route exactness, non-escape, and TasteGate field faithfulness for BEDC.Derived.NestedIntervalSelectionUp.

Chapter 2250

A Concrete Naming Certificate for $\mathsf{DenseCauchyEmbedding}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley on dense uniform embeddings, the $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ packet names the finite source-to-completion interface used when a dense metric source is admitted into a Cauchy completion. It sits after the metric source of chapter 57, the regular dense Cauchy route of chapter 1453, the completion embedding surface of chapter 1512, the sequence-comparison ledger of chapter 1412, and the metric-completion boundary of chapter 1506. The packet exposes density as finite NameCert data rather than as a quotient completion, selected limit, host equality of completion points, countable-choice selector, or ambient completeness principle.

Definition 2250.1 (Dense Cauchy embedding carrier)**.** *A* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, D, Q, J, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the source carrier, classifier, distance witness, and finite observation histories.*
2. *D is the displayed dense source row, recording the finite approximation schedule that lets completion requests be read back through source observations.*
3. *Q is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *evidence row, exposing the Cauchy comparison route used to identify source approximants before completion consumers read them.*
4. *J is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row that inserts source observations into the completion boundary.*
5. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row consumed only after M, D, Q, J have been displayed.*
6. *H records componentwise* hsame *transport for the metric source, dense source row, Cauchy evidence, completion embedding, metric-completion handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto D \longmapsto Q \longmapsto J \longmapsto L.$$

8. *P is the* Pkg *provenance over* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseCauchyEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host equality of completed points, countable choice, ambient metric completeness, or bridge credentials.

Lean4 ✓ `BEDC.Derived.DenseCauchyEmbeddingUp.DenseCauchyEmbeddingTasteGate single carrier alignment`

Theorem 2250.2 (Dense Cauchy embedding NameCert obligations)**.** *For an accepted* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ *carrier $E = (M, D, Q, J, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseCauchyEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric row, dense source row, Cauchy equivalence or net evidence, completion embedding, metric-completion handoff, transport, replay, provenance, and local naming rows;*

2. *dense-source visibility, requiring every completion-facing approximation request to expose D before it can be read through J or L;*

3. *Cauchy coherence, requiring the displayed Q row to compare the source approximants through* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *before the completion embedding is consumed;*

4. *embedding locality, requiring J to insert only the source observations displayed by M, D, Q into the completion-facing boundary;*

5. *classifier stability, requiring transported packets with the same displayed M, D, Q, J, L route to be identified through the carried* hsame *and* Cont *rows;*

6. *ledger non-escape, excluding quotient completion equality, selected limits, host equality, countable choice, ambient completeness, and bridge checking.*

Proof. Carrier admission is projection from Definition 2250.1. The tuple lists exactly the metric source, dense source, Cauchy evidence, completion embedding, metric-completion handoff, transport, replay, provenance, and local naming rows.

Dense-source visibility is the projection to D. A completion-facing request cannot reach the embedding row J or the completion row L until the displayed dense source approximation route has been named.

Cauchy coherence is the projection to Q. Whether the local evidence is sequential or net-presented, the packet requires the approximants to be compared before their image is treated as a completion-facing entry.

Embedding locality is the dependency order in C. The continuation row replays only the route from M, D, Q through J to L, while H, P, N transport, source, and name that same finite packet. Since no excluded host-side datum is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2250.3 (Dense Cauchy embedding metric completion handoff)**.** *Let $E = (M, D, Q, J, L, H, C, P, N)$ be an accepted* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ *carrier. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of E reads the completion boundary only through the finite route*

$$M \longmapsto D \longmapsto Q \longmapsto J \longmapsto L \longmapsto H, C, P, N.$$

Thus the dense-source embedding interface is a displayed BEDC handoff from $\mathsf{Met}^{\uparrow}$ *source rows through* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, *and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *evidence into* $\mathsf{MetricCompletion}^{\uparrow}$, *and not a hidden quotient completion or selected-limit principle.*

Proof. Start with an accepted carrier. The source row M supplies the metric observations, and the dense source row D records which finite source approximants answer the completion-facing request.

The Cauchy evidence row Q is read next. It is the only row that compares the selected approximants before they enter the completion boundary, so a completion consumer cannot bypass it and still consume this packet.

The embedding row J then inserts those displayed observations into the completion-facing route. Only after M, D, Q, J have been fixed does the metric-completion row L become available.

Finally H, C, P, N preserve, replay, source, and name the same finite route. The packet therefore exports the metric-completion handoff without adding quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or bridge credentials. □

Falsifiable prediction: An accepted $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ consumer that reaches $\mathsf{MetricCompletion}^{\uparrow}$ without the displayed $\mathsf{Met}^{\uparrow}$ source, dense source row, Cauchy evidence, and $\mathsf{CompletionEmbedding}^{\uparrow}$ route refutes the carrier surface.

Independence witness: metric, regular-cauchy-dense-embedding, completionembedding, cauchy-sequenceequivalence, cauchynet, metriccompletion: no listed sibling supplies the whole packet $E = (M, D, Q, J, L, H, C, P, N)$, because each sibling contributes only a source metric, dense Cauchy route, embedding row, Cauchy comparison row, or completion boundary while this chapter binds them with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseCauchyEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ packet is generated from a MetricUp source row, displayed dense source data, CauchySequenceEquivalenceUp or CauchyNetUp evidence, a CompletionEmbeddingUp handoff, a MetricCompletionUp consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2250.1, Definition 2250.2, and Definition 2250.3 seed the dense Cauchy embedding interface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed points, countable choice, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source visibility, Cauchy coherence, embedding locality, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.DenseCauchyEmbeddingUp.

Chapter 2251

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$

Following Bishop–Bridges and Kelley-style completion theory, the $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ packet records the finite universal-property surface for regular Cauchy completion. It sits after the dyadic tolerance core of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric-completion handoff of chapter 1506, the uniform-completion route of chapter 1473, and the separated-metric comparison boundary of chapter 1411. The packet names only the displayed extension and uniqueness route available to completion consumers; it does not assert quotient completion, selected representatives, ambient completeness, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2251.1 (Regular Cauchy completion universal-property carrier)**.** *A* RegularCauchyCompletionUniversalP *carrier is a finite* BHist *packet*

$$U = (D, W, Q, R, M, F, S, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the regular Cauchy readback.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that makes the source observations available.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row read through D and W.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the same displayed regular Cauchy route.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row consumed by metric-completion readers.*
6. *F is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row consumed by uniform-completion readers.*
7. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row used by uniqueness of extensions.*
8. *E is the extension-existence row: it records the displayed route from a regular Cauchy source read through D, W, Q, R into the completion-facing target read through M, F, S.*
9. *L is the uniqueness ledger: it compares two extension rows over the same D, W, Q, R, M, F, S route by separatedness rather than by host equality.*
10. *H records componentwise* hsame *transport for $D, W, Q, R, M, F, S, E, L$ and the local naming row.*

11. *C records displayed* Cont *replay of the route*

$$D, W, Q, R \longmapsto M, F, S \longmapsto E, L, H, P, N.$$

12. *P is the* Pkg *provenance over* RegularCauchyCompletionUniversalProperty$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, MetricCompletion$^{\uparrow}$, UniformCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

13. *N is the local* NameCert$_{\text{RegularCauchyCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, Q, R, M, F, S, E, L, H, C, P, N$. It has no coordinate for quotient completion, representative selection, host equality of completed points, hidden completeness of the target, countable choice, or completeness of Real$^{\uparrow}$.

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionUniversalPropertyUp.RegularCauchyCompletionUniversalPropert
single carrier alignment

Theorem 2251.2 (Regular Cauchy completion universal-property NameCert obligations)**.** *For every accepted* RegularCauchyCompletionUniversalProperty$^{\uparrow}$ *carrier $U = (D, W, Q, R, M, F, S, E, L, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of carrier admission; dyadic tolerance exposure through D; finite stream exposure through W; regular rational source exposure through Q; terminal real sealing through R; metric and uniform completion handoff through M, F; separated comparison through S; extension existence through E; uniqueness through L; componentwise transport through H; displayed replay through C; provenance through P; local naming through N; and non-escape from quotient completion, representative selection, host equality, hidden target completeness, countable choice, or* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2251.1. The accepted packet exposes exactly the dyadic tolerance row, stream-window row, regular rational source, real seal, metric-completion handoff, uniform-completion handoff, separated comparison row, extension row, uniqueness ledger, transport, replay, provenance, and local name.

The regular source route is finite. Rows D, W, Q provide the dyadic, stream, and regular rational data before the terminal real seal R is read, so the extension row cannot begin from a hidden completed point.

The completion-facing route is also finite. Rows M, F, S record the metric, uniform, and separated target surfaces before E and L are inspected, so existence and uniqueness are read through displayed completion data rather than through a quotient or a selected representative.

Rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded host-side coordinate exists in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2251.3 (Regular Cauchy completion universal-property core)**.** *Every extension-and-uniqueness read of an accepted* RegularCauchyCompletionUniversalProperty$^{\uparrow}$ *packet factors through*

$$D, W, Q, R \longmapsto M, F, S \longmapsto E, L, H, C, P, N.$$

The route first exposes dyadic tolerance, stream windows, regular rational source data, and the terminal real seal; then it exposes metric completion, uniform completion, and separated comparison rows; and only then exports extension existence and uniqueness. No accepted read exports quotient completion, selected representatives, ambient completeness, countable choice, host equality of completed points, or Real$^{\uparrow}$ *completeness.*

Proof. Read the accepted carrier $U = (D, W, Q, R, M, F, S, E, L, H, C, P, N)$. Its continuation row C displays the order in which the regular Cauchy source route is made available before the completion-facing rows are inspected.

The first stage is D, W, Q, R. It uses the dyadic tolerance ledger, the finite stream schedule, and the regular rational source to reach the real seal; none of these rows supplies a quotient object or a selected limit.

The second stage is M, F, S. The metric and uniform completion rows are consumed together with the separated comparison row, so the uniqueness ledger can compare extensions by separatedness inside the packet.

The final stage is E, L, H, C, P, N. Extension existence and uniqueness are transported, replayed, sourced, and named by the displayed structural rows. Because the carrier has no additional coordinate, every accepted universal-property read factors through the stated route. □

Theorem 2251.4 (Regular Cauchy completion extension uniqueness handoff)**.** *For every accepted* $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ *packet* $U = (D, W, Q, R, M, F, S, E, L, H, C, P, N)$*, extension existence and extension uniqueness are public only through the ordered rows*

$$D, W, Q, R, M, F, S, E, L, H, C, P, N.$$

The source side first reads dyadic tolerance, stream windows, regular rational source data, and the terminal real seal; the completion side then reads metric completion, uniform completion, and separated comparison rows; and the extension row E *and uniqueness ledger* L *are exported only after that displayed route is present.*

Proof. Project the accepted carrier of Definition 2251.1. Its rows already place the source route D, W, Q, R before the completion-facing route M, F, S.

Apply Definition 2251.2. The local NameCert surface exposes extension existence only through E and exposes uniqueness only through L, with componentwise transport, continuation replay, provenance, and local naming supplied by H, C, P, N.

Use Definition 2251.3 to identify the ordered route. The metric and uniform completion rows provide the completion handoff, while the separated comparison row supplies the equality test used by the uniqueness ledger. Hence a consumer receives extension existence and uniqueness only through the displayed packet. □

Theorem 2251.5 (Regular Cauchy completion separated nonescape)**.** *No accepted* RegularCauchyCompletionUniver *consumer can replace the separated comparison row* S *by quotient completion, representative selection, host equality of completed points, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or hidden target completeness. Any public uniqueness read is confined to the displayed separated comparison row together with* $D, W, Q, R, M, F, E, L, H, C, P, N$*.*

Proof. Inspect the carrier coordinates of Definition 2251.1. The only row assigned to comparison of completed extension values is S, and the only row assigned to uniqueness is L.

Apply Definition 2251.4. It places S before E and L in the completion-facing handoff, so uniqueness has no independent source outside the separated comparison boundary.

Finally use the non-escape clause of Definition 2251.2. Quotient completion, representative selection, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and hidden target completeness have no carrier coordinate, no continuation route, and no provenance row in U. □

Theorem 2251.6 (Regular Cauchy completion Bishop handoff)**.** *Every accepted* RegularCauchyCompletionUniversal *packet*

$$U = (D, W, Q, R, M, F, S, E, L, H, C, P, N)$$

whose dense source, Cauchy filter, separated uniqueness, and extension rows are read by a BishopRegularCauchyComp *consumer factors through chapter 1639, chapter 301, chapter 126, and chapter 38 before it exports a universal arrow. The exported arrow is therefore a finite L10 completion-facing route through the displayed dyadic tolerance, stream window, regular rational source, Real seal, metric completion, uniform completion, separated comparison, extension, uniqueness, transport, replay, provenance, and local naming rows.*

The route exports no quotient completion, selected representative, countable-choice selector, host equality of completed points, hidden target completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier row of Definition 2251.1. The dense source and Cauchy filter read are visible only through D, W, Q, R, so the Bishop consumer first opens the dyadic, stream, RegSeqRat, and Real chapters named in the statement.

Next apply Definition 2251.3 and Definition 2251.4. These route the completion-facing part through M, F, S before E and L are exported, which is exactly the metric, uniform, and separated handoff expected by the $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ surface.

Finally use Definition 2251.5. Since the separated comparison row is the only admissible uniqueness boundary, the universal arrow cannot be reconstructed from quotient completion, selected representatives, countable choice, host equality, hidden completeness, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence or uniqueness while omitting any of the dyadic tolerance, stream-window, regular rational source, real seal, metric-completion, uniform-completion, separated comparison, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletion, uniform-completion, separatedmetric: none is bijective because $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ jointly carries tolerance, finite observation, regular source, terminal seal, metric handoff, uniform handoff, separated comparison, extension, uniqueness, transport, replay, provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2251.1, Definition 2251.2, and Definition 2251.3 seed the finite regular-Cauchy universal-property surface over dyadic, stream, regular rational, real, metric-completion, uniform-completion, separated-metric, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, ambient completeness, countable choice, host equality of completed points, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular source exposure, completion handoff, separated uniqueness, structural transport, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyCompletio

Constructive story: A $\mathsf{RegularCauchyCompletionUniversalProperty}^{\uparrow}$ packet is generated from dyadic tolerance data, finite stream windows, regular rational source rows, a terminal Real seal, metric and uniform completion rows, a separated comparison row, extension and uniqueness rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2252

A Concrete Naming Certificate for $\mathsf{LandauSymbol}^{\uparrow}$

Following Landau and Hardy, the $\mathsf{LandauSymbol}^{\uparrow}$ packet records eventual finite-threshold comparison as BEDC data rather than as host asymptotic notation. It sits over the finite window service of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the modulus-controlled real-name route of chapter 1423. The packet gives Real and Completion consumers a named threshold comparison row without importing limit notation, quotient streams, selected tails, trichotomy, or ambient completeness.

Definition 2252.1 (Landau symbol carrier)**.** *A* $\mathsf{LandauSymbol}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, W, B, Q, E, H, C, P, N)$$

with the following visible rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row naming the inspected stream family and the finite windows used to read the candidate function.*
2. *T is the comparison* $\mathsf{StreamName}^{\uparrow}$ *row naming the matching finite windows for the bounding stream.*
3. *W is the threshold witness row, consisting of one displayed unary threshold and the proof ledger that every accepted tail request is beyond it.*
4. *B is the eventual-bound ledger, recording the displayed finite comparison between the two streams for all windows admitted after W.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row through which both stream reads enter regular rational comparison.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source stream, comparison stream, threshold witness, eventual-bound ledger, and RegSeqRat handoff have been displayed.*
7. *H records componentwise* hsame *transport for the source stream, comparison stream, threshold witness, eventual-bound ledger, RegSeqRat handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Landau comparison request through $S, T, W, B, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LandauSymbol}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LandauSymbol}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, W, B, Q, E, H, C, P, N$. It has no coordinate for a host limit, quotient stream equality, selected infinite tail, external asymptotic notation, trichotomy of completed reals, or ambient completeness.

Lean4 (definition) `BEDC.Derived.LandauSymbolUp.LandauSymbolUp`

Theorem 2252.2 (Landau symbol NameCert obligations)**.** *For every accepted* $\mathsf{LandauSymbol}^{\uparrow}$ *carrier $L = (S, T, W, B, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LandauSymbol}^{\uparrow}}$ *surface consists exactly of carrier admission, threshold monotonicity, eventual-bound exactness, RegSeqRat and Real handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Threshold monotonicity says that every later finite window served by the source and comparison streams remains beyond the displayed threshold W. Eventual-bound exactness says that the bound read by a consumer is precisely the comparison ledger B after the route*

$$S, T, W \longmapsto B \longmapsto Q \longmapsto E$$

has been displayed. Ledger non-escape excludes host limit notation, quotient stream equality, selected tails, trichotomy, and ambient completeness from the packet.

Proof. Project the carrier of Definition 2252.1. The two stream rows S, T supply the only observable source and comparison windows, and W is the only threshold coordinate.

The eventual comparison is read through B. Threshold monotonicity is the row condition that a later accepted request still factors through the same displayed threshold, so no separate tail selector can enter the route.

The handoff rows Q and E place the comparison inside RegSeqRat readback before any Real-facing consumer sees a seal. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so there is no coordinate from which host asymptotic notation, quotient stream equality, selected tails, trichotomy, or ambient completeness can be exported. □

Falsifiable prediction: Every accepted $\mathsf{LandauSymbol}^{\uparrow}$ Real-facing consumer factors through displayed source and comparison $\mathsf{StreamName}^{\uparrow}$ windows, one threshold witness, one eventual-bound ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before it may use the comparison as asymptotic evidence.

Independence witness: streamname, regseqrat, real, realcauchymodulus: no listed sibling is bijective to $\mathsf{LandauSymbol}^{\uparrow}$, because the siblings respectively supply finite windows, regular rational readback, terminal real sealing, or modulus-controlled real-name service, while this packet uniquely binds two stream rows to one finite threshold and one eventual-bound ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{LandauSymbol}^{\uparrow}}$.

Constructive story: A $\mathsf{LandauSymbol}^{\uparrow}$ packet is generated from source and comparison StreamName rows, a threshold witness, an eventual-bound ledger, RegSeqRat handoff, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LandauSymbol}^{\uparrow}}$, seed)

Scope closed: Definition 2252.1 and Definition 2252.2 seed the finite threshold-comparison carrier over StreamName, RegSeqRat, Real, and RealCauchyModulus rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LandauSymbol}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host asymptotic notation, quotient stream equality, selected infinite tails, trichotomy, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold monotonicity, eventual-bound exactness, RegSeqRat and Real handoff, structural transport, and ledger non-escape for BEDC.Derived.LandauSymbolUp.

Chapter 2253

A Concrete Naming Certificate for $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$

The $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ packet realizes the compact-uniform-continuity route of chapter 9237 as a finite extractor surface: dyadic compactness rows provide a finite net and dyadic radius ledger, metric triangle rows transport the estimate, continuous-map rows supply pointwise moduli, and the resulting route hands a tolerance-only radius to $\mathsf{UniformModulus}^{\uparrow}$. It sits beside the compact metric source of chapter 129, the metric triangle surface of chapter 57, the pointwise graph rows of chapter 130, the Lebesgue-radius ledger of chapter 452, and the uniform-modulus consumer of chapter 369. The packet names the finite extractor route only; it does not assert open-cover compactness, sequential compactness, a host Lebesgue number, or an ambient completeness theorem.

Definition 2253.1 (Dyadic compactness extractor carrier)**.** *A* $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, M, T, R, F, L, U, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the compact carrier and the displayed ProbeBundle center family.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *triangle row for the source and target metric estimates used by the extractor.*
3. *T is the dyadic radius row naming the finite precision scale attached to the accepted compact net.*
4. *R is the finite-net readback row that associates every accepted source point request with a displayed center and dyadic radius from T.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying pointwise modulus data at each displayed center.*
6. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *ledger row that records the finite lower-radius comparison consumed by the uniform route.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from K, M, T, R, F, L.*
8. *H records componentwise* hsame *transport for compact source, metric triangle, dyadic radius, finite-net readback, continuous-map modulus, Lebesgue ledger, uniform-modulus handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-uniform request through $K, M, T, R, F, L, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCompactnessExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, T, R, F, L, U, H, C, P, N$. It has no coordinate for an open cover, selected subcover, sequential compactness argument, host Lebesgue number, host minimum selector, choice principle, or ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.DyadicCompactnessExtractorUp.DyadicCompactnessExtractorUp`

Theorem 2253.2 (Dyadic compactness extractor NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ *carrier $D = (K, M, T, R, F, L, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicCompactnessExtract}}$ *surface consists exactly of carrier admission, compact finite-net exposure, dyadic radius exactness, metric triangle compatibility, continuous-map pointwise-modulus exposure, finite Lebesgue-ledger readback, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every compact-uniform consumer must read the displayed finite route*

$$K, M, T, R, F, L \longmapsto U \longmapsto H, C, P, N$$

before exporting a uniform modulus.

Proof. Project the carrier row by row. The compact source K supplies the finite net, while T and R record the dyadic radius and the finite center selected for a source request.

The metric triangle row M and continuous-map row F provide the only estimate data used by the extractor. The Lebesgue ledger L is then the finite lower-radius comparison that makes the resulting radius uniform over the displayed net.

The handoff row U is read only after those finite rows have been exposed. The structural rows H, C, P, N preserve the same route, so no open cover, selected subcover, sequential compactness argument, host Lebesgue number, host minimum selector, choice principle, or ambient completeness theorem can enter the certificate. □

Theorem 2253.3 (Dyadic compactness extractor vision handoff)**.** *The finite extractor packet $D = (K, M, T, R, F, L, U, H, C, P, N)$ realizes the compact-uniform-continuity handoff of chapter 9237: any accepted consumer factors through a displayed finite net, one dyadic radius row, the metric triangle row, the continuous-map modulus row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *before it exports a* $\mathsf{UniformModulus}^{\uparrow}$ *row. The exported row is therefore a BEDC finite-radius ledger, not a host modulus selector.*

Proof. Start with a compact-uniform consumer. The carrier forces the consumer to read K, T, R as the compact finite-net and dyadic radius data before any uniform-radius output is visible.

Next the route reads the metric and continuity evidence through M and F. The finite Lebesgue ledger L is the displayed comparison tying those pointwise rows to one compact-net radius.

The uniform-modulus row U is downstream of this finite chain. Since H, C, P, N only transport, replay, source, and name the same packet, the handoff is exactly the BEDC route described in the vision chapter and not an external modulus selector. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ compact-uniform consumer factors through a displayed finite net, dyadic radius row, metric triangle row, continuous-map

modulus row, hsame transport, Cont replay, Pkg provenance, and local NameCert before exporting a uniform modulus.

Independence witness: compactmetric, metric, continuousmap, finitelebesguenumber, uniformmodulus: $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the compact finite-net source, metric triangle estimates, dyadic radius ledger, pointwise continuousmap modulus rows, finite Lebesgue-radius readback, and UniformModulus handoff; each sibling omits at least one of these extractor coordinates.

The Lean TasteGate follow-up is `BEDC.Derived.DyadicCompactnessExtractorUp`, with Nontrivial DyadicComp and FieldFaithful $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ instances preserving $K, M, T, R, F, L, U, H, C, P, N$ as separate real extractor rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCompactnessExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCompactnessExtractor}^{\uparrow}$ packet is generated from a CompactMetric finite-net row, a Metric triangle row, a dyadic radius row, finite-net readback, ContinuousMap pointwise modulus data, a FiniteLebesgueNumber ledger, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCompactnessExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2253.1, Definition 2253.2, and Definition 2253.3 seed the finite dyadic compactness extractor route behind compact-uniform-continuity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCompactnessExtractor}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.DyadicCompactnessExtractorUp.DyadicCompactnessExtractorUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, sequential compactness, host Lebesgue numbers, host minimum selectors, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, dyadic-radius exactness, metric-triangle compatibility, continuous-map modulus exposure, finite Lebesgue-ledger readback, uniform-modulus handoff, structural transport, and ledger non-escape for BEDC.Derived.DyadicCompactnessExtractorUp.

Chapter 2254

A Concrete Naming Certificate for $\mathsf{CofinalDyadicMesh}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CofinalDyadicMesh}^{\uparrow}$ packet names the finite selector that refines an arbitrary Real- or completion-facing observation request to a dyadic mesh window. It sits between the dyadic mesh discipline of chapter 358, the endpoint ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal Real seal of chapter 38. The chapter records the selector as finite NameCert data; it does not assert a quotient real equality, selected limit, or countable-choice subnet.

Definition 2254.1 (Cofinal dyadic mesh carrier)**.** *A* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite observation request coming from a Real or regular Cauchy consumer.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic endpoint and tolerance ledger available to answer Q.*
3. *M is the* $\mathsf{DyadicMesh}^{\uparrow}$ *mesh row selected from D and refined far enough for the request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the selected mesh depth.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after M, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the request, dyadic ledger, mesh, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from Q through D, M, S, R, E, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{CofinalDyadicMesh}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalDyadicMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, M, S, R, E, H, C, P, N$. It has no coordinate for a hidden subnet, quotient Cauchy class, host real equality, selected limit, countable-choice schedule, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.CofinalDyadicMeshUp.CofinalDyadicMeshTasteGate single carrier alignment`

Theorem 2254.2 (Cofinal dyadic mesh NameCert obligations)**.** *For every accepted* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *carrier* $K = (Q, D, M, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalDyadicMesh}^{\uparrow}}$ *surface consists exactly of carrier admission, request exposure through* Q*, dyadic endpoint and tolerance exposure through* D*, selected mesh exposure through* M*, finite-window exposure through* S*, regular readback through* R*, terminal Real-seal exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*. Every Real or regular Cauchy finite observation admitted by the packet must pass through this displayed dyadic mesh selector before reading* S, R, E*.*

Proof. Project the carrier. The request row Q is the only incoming observation demand, while D and M display the dyadic endpoint ledger and the selected mesh depth used to answer it.

The finite observation then proceeds through S, R, E. The stream window S is opened at the selected mesh depth, the regular readback R consumes that window, and the terminal Real seal E is read only after those rows are visible.

The structural rows H, C, P, N transport, replay, package, and name exactly the displayed route. Since no hidden subnet, quotient class, host equality, selected limit, choice schedule, or ambient completeness row appears in the carrier, every admitted observation factors through the finite dyadic selector. □

Falsifiable prediction: If the selected mesh row M is removed, a Real or regular Cauchy finite observation can no longer be forced to pass through a displayed dyadic refinement before the StreamName, RegSeqRat, and Real seal rows are read.

Independence witness: dyadicmesh, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{CofinalDyadicMesh}^{\uparrow}$ because this packet stores the observation request, dyadic endpoint ledger, selected mesh depth, finite stream window, regular readback, terminal Real seal, transport, replay, provenance, and local NameCert row as one selector route.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalDyadicMesh}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CofinalDyadicMesh}^{\uparrow}$ packet is generated from a finite observation request, a $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint and tolerance ledger, a selected $\mathsf{DyadicMesh}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ finite-window data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalDyadicMesh}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2254.1 and Definition 2254.2 seed the finite dyadic-mesh selector feeding Real and RegSeqRat finite observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalDyadicMesh}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CofinalDyadicMeshUp.CofinalDyadicMeshTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient real equality, selected limits, hidden subnets, countable-choice schedules, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector cofinality, finite-window routing, regular readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CofinalDyadicMeshUp.

Chapter 2255

A Concrete Naming Certificate for $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ packet records the finite criterion by which displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ data produces Cauchy behaviour for a sequence of constructive real names. The packet reads the sequence and candidate limit through the $\mathsf{RealSequenceLimit}^{\uparrow}$ surface of chapter 1413, the finite name windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, the nonquotient comparison boundary of chapter 1404, and the terminal real-name seal of chapter 38. It is a displayed finite tail-window criterion, not a quotient stream theorem, selected representative principle, countable-choice argument, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 2255.1 (RealSequenceCauchyCriterion carrier)**.** *A* $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, S, R, D, T, E, H, C, P, N)$$

with the following visible rows:

1. *L is the accepted* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row naming the displayed sequence, candidate limit, and finite convergence classifier;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that lists the inspected sequence windows and limit windows for each requested tolerance;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled sequence and limit observations;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the tolerance rows and rational comparison cells used by the tail-window test;*
5. *T is the finite Cauchy tail-window classifier: each accepted tolerance request is assigned one displayed tail index and the paired windows used to compare two later sequence reads through R and D;*
6. *E is the* $\mathsf{RegularCauchyEquivalence}^{\uparrow}$ *comparison boundary that records the two sequence reads as a nonquotient regular-Cauchy comparison before any* $\mathsf{Real}^{\uparrow}$ *handoff;*
7. *H stores componentwise* hsame *transports for L, S, R, D, T, E and the local naming rows;*
8. *C stores displayed* Cont *routes exposing the limit row, window schedule, regular rational readback, dyadic ledger, Cauchy classifier, and comparison boundary in that order;*

9. *P is the* Pkg *provenance over* RealSequenceLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyEquivalence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* NameCert$_{\text{RealSequenceCauchyCriterion}^{\uparrow}}$ *row.*

The carrier asserts only that these rows form one displayed limit-to-Cauchy packet. It has no field for quotient streams, selected representatives of real names, countable choice over tail bounds, trichotomy, or an ambient completeness theorem for Real$^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RealSequenceCauchyCriterionUp.RealSequenceCauchyCriterionTasteGate single carrier alignment`

Theorem 2255.2 (RealSequenceCauchyCriterion NameCert obligations)**.** *For every accepted* RealSequenceCauchy *carrier $K = (L, S, R, D, T, E, H, C, P, N)$, the local* NameCert$_{\text{RealSequenceCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, limit-row exposure through L, finite window exposure through S, regular rational readback through R, dyadic tolerance comparison through D, Cauchy tail-window classification through T, nonquotient regular-Cauchy comparison through E, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for quotient streams, selected representatives, countable choice, trichotomy, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2255.1. The accepted packet exposes the limit row, stream schedule, regular rational readback, dyadic ledger, Cauchy classifier, comparison boundary, transport, replay, provenance, and local naming rows.

The limit-row exposure is finite because L is consumed only through the scheduled windows S. For each requested tolerance, the convergence classifier already present in RealSequenceLimit$^{\uparrow}$ supplies a displayed tail window comparing later sequence reads to the candidate limit.

The Cauchy classifier T reads two later sequence windows through the same RegSeqRat$^{\uparrow}$ row R and the same DyadicRatCore$^{\uparrow}$ ledger D. The comparison first factors each sequence read through the displayed limit row and then composes the two finite dyadic comparisons inside the recorded tolerance budget.

The comparison boundary E prevents the route from becoming quotient equality of real streams. Rows H, C, P, N transport, replay, source, and name the same packet, so the local certificate has no coordinate from which a selected representative, a choice function for all tails, a trichotomy oracle, or ambient completeness of Real$^{\uparrow}$ can be exported. □

Theorem 2255.3 (RealSequenceCauchyCriterion limit-Cauchy handoff)**.** *Every* RealSequenceLimit$^{\uparrow}$ *display accepted by a* RealSequenceCauchyCriterion$^{\uparrow}$ *carrier yields a Cauchy tail-window route through finite* RegSeqRat$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *rows. Concretely, for each requested tolerance in the classifier row T, the carrier exposes a tail index and two later sequence windows whose regular rational readbacks are compared by the dyadic ledger before the comparison is handed to* RegularCauchyEquivalence$^{\uparrow}$ *and any downstream* Real$^{\uparrow}$ *consumer.*

Thus a displayed sequence-limit packet is enough to serve Cauchy behaviour at the finite BEDC window level, without quotient streams, selected representatives, countable choice, real trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier $K = (L, S, R, D, T, E, H, C, P, N)$. The row L supplies the displayed sequence-limit packet, while S lists the finite sequence and limit windows that may be inspected.

For a tolerance request, read the tail entry of T. It selects a displayed tail index and two later sequence windows inside the same schedule S. Their rational observations are read through R, and the required tolerance cells are read through D.

Apply the sequence-limit obligations of Definition 1413.2 to both later windows. Each window is compared to the same candidate limit through the recorded dyadic tolerance ledger. Combining those two finite comparisons gives the Cauchy tail-window comparison recorded by T.

Finally pass the comparison through the nonquotient boundary E before any $\mathsf{Real}^{\uparrow}$ handoff. Since all reads are coordinates of K and all replay goes through C, the route introduces no quotient stream, representative selector, countable choice, trichotomy decision, or ambient completeness row. □

Theorem 2255.4 (RealSequenceCauchyCriterion tail-window stability)**.** *Let*

$$K = (L, S, R, D, T, E, H, C, P, N)$$

and

$$K' = (L', S', R', D', T', E', H', C', P', N')$$

be accepted $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ *packets. If the two packets are connected componentwise by* hsame *transport on* $L, S, R, D, T, E, H, C, P, N$*, then every Cauchy tail-window route exposed by* K *transports to a Cauchy tail-window route exposed by* K'*. The transported route keeps the limit row, stream schedule, regular rational readback, dyadic tolerance ledger, Cauchy classifier, nonquotient comparison boundary, transport, replay, provenance, and local name visible, and it introduces no quotient streams, selected representatives, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project both carriers through Definition 2255.1. The componentwise hsame assumptions identify the rows $L, S, R, D, T, E, H, C, P, N$ with their primed counterparts while preserving the carrier coordinates that expose the tail-window route.

Use the NameCert obligations of Definition 2255.2 to transport each displayed row. The limit row, finite schedule, regular rational readback, dyadic ledger, Cauchy classifier, and comparison boundary are moved as rows of the packet, so no representative selector or quotient stream is introduced.

Apply the handoff theorem Definition 2255.3 after transport. The same tolerance request now reads the primed tail index and primed later sequence windows through S', R', and D', then hands the comparison to E' before any $\mathsf{Real}^{\uparrow}$ consumer.

The remaining coordinates H', C', P', N' record transport, replay, provenance, and local naming for the transported route. Since neither packet contains a row for countable choice, trichotomy, quotient streams, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the stability route cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSequenceCauchyCriterion}^{\uparrow}$ packet is generated from a displayed RealSequenceLimit row, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a Cauchy tail-window classifier, a RegularCauchyEquivalence boundary, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSequenceCauchyCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 2255.1, Definition 2255.2, and Definition 2255.3 seed the finite displayed-limit to Cauchy tail-window criterion over StreamName, RegSeqRat, DyadicRatCore, RegularCauchyEquivalence, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSequenceCauchyCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected real-name representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, limit-row exposure, finite Cauchy tail-window classification, regular rational and dyadic comparison exactness, nonquotient comparison handoff, structural transport, and ledger refusal for BEDC.Derived.RealSequenceCauchyCriterionUp.

Chapter 2256

A Concrete Naming Certificate for $\mathsf{CompletionUnitTriangle}^{\uparrow}$

Following Bishop–Bridges completion and Mac Lane adjunction triangle identities, the $\mathsf{CompletionUnitTriangle}^{\uparrow}$ packet records the finite L10 surface at which a completion-facing read returns through the same unit, counit, and adjunction rows that introduced it. It sits after the unit packet of chapter 1527, the adjunction packet of chapter 1456, and the regular universal surface of chapter 1425. The packet names the triangle identity as a route through displayed finite rows; it does not form quotient completions, compare host equality of completed objects, choose representatives, assume countable choice, or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2256.1 (Completion unit triangle carrier)**.** *A* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (U, K, A, E, R, S, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row sending the source regular name into the completion-facing boundary.*
2. *K is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning an already displayed completion read to the terminal real-facing route.*
3. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row binding U and K as the unit and counit faces of one adjunction packet.*
4. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row through which completion consumers read the same unit-facing datum.*
5. *R is the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *regular-Cauchy universal row attached to the same source name and extension ledger.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row supplying the visible stream segment of the source name.*
7. *D is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for the same finite window.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that fixes the displayed precision used by S and D.*
9. *H stores componentwise* hsame *transport for the unit, counit, adjunction, universal-property, regular-Cauchy, stream, readback, dyadic, terminal Real, replay, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay for the triangle route: unit first, universal extension second, regular-Cauchy readback third, counit return fourth, and terminal Real seal last.*

11. *P is the* Pkg *provenance over* $\mathsf{CompletionUnitTriangle}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionUnitTriangle}^{\uparrow}}$ *row.*

The carrier exposes a triangle-identity route only as a finite replay path through the named rows. It carries no quotient completion object, no host equality of completed carriers, no selected representative, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionUnitTriangleUp.CompletionUnitTriangleTasteGate single carrier alignment`

Theorem 2256.2 (Completion unit triangle NameCert obligations)**.** *For an accepted* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *carrier* $T = (U, K, A, E, R, S, D, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionUnitTriangle}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the unit, counit, adjunction, universal-property, regular-Cauchy, stream, regular-rational readback, dyadic, transport, replay, provenance, and local naming rows;*

2. *unit and counit visibility, requiring every triangle read to cite U before any completion-facing extension row and to cite K before any return to the terminal real seal;*

3. *triangle-route identity, requiring the adjunction row A to bind the unit and counit faces to the same displayed completion read;*

4. *classifier stability, requiring transported triangle evidence to replay through H and C before a completion-facing consumer reads it;*

5. *ledger non-escape, requiring the package and local naming rows to export no quotient completion, host equality, selected representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 2256.1. The tuple lists exactly the unit, counit, adjunction, universal-property, regular-Cauchy, stream, readback, dyadic, transport, replay, provenance, and local naming rows.

Unit and counit visibility are the first two completion faces of the tuple. A completion-facing read enters through U, and a read that returns to the Real boundary exits through K.

The triangle-route identity is recorded by A together with the replay row C. The adjunction row binds the unit and counit faces to one displayed read, while C forces the same route through E, R, S, D, Q, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any consumer sees it.

Classifier stability is componentwise transport through H followed by the same replay through C. Ledger non-escape is the inventory statement for P and N: the carrier contains no quotient completion, host equality, selected representative, countable-choice schedule, or ambient completeness row. □

Theorem 2256.3 (Completion unit triangle identity route)**.** *Every accepted* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *triangle read factors through the same unit, counit, adjunction, universal-extension, regular-Cauchy, stream, dyadic, and Real seal rows before any completion-facing consumer reads it. Concretely, the exported route is exhausted by*

$\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, I

No accepted read may replace one of these rows by quotient-completion equality, host equality, a selected representative, countable choice, or an ambient completeness principle.

Proof. Read the carrier of Definition 2256.1. The only analytic rows available to a triangle consumer are U, K, A, E, R, S, D, Q and the terminal $\mathsf{Real}^{\uparrow}$ seal named by the transport, replay, and provenance rows.

Apply the NameCert obligations of Definition 2256.2. Unit and counit visibility force the read to enter through U and return through K, while triangle-route identity forces both faces through the same adjunction row A.

The universal-extension and regular-Cauchy rows E and R then keep the completion-facing read attached to the same regular name. The stream, readback, and dyadic rows S, D, Q display the finite windows and tolerances consumed by the terminal $\mathsf{Real}^{\uparrow}$ seal.

The package and local naming rows contain no further completion object or host equality coordinate. Therefore every accepted triangle read is exhausted by the displayed route and cannot export quotient completions, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUnitTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionUnitTriangle}^{\uparrow}$ packet is generated from a completion unit row, counit row, adjunction row, universal-property row, regular-Cauchy universal row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionUnitTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2256.1, Definition 2256.2, and Definition 2256.3 seed the finite triangle-identity packet over unit, counit, adjunction, universal-property, regular-Cauchy, StreamName, RegSeqRat, DyadicRatCore, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionUnitTriangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, host equality of completed objects, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit and counit visibility, triangle-route identity, classifier stability, and ledger non-escape for BEDC.Derived.CompletionUnitTriangleUp.

Chapter 2257

A Concrete Naming Certificate for $\mathsf{TotalVariationFunction}^{\uparrow}$

Following Jordan's decomposition theorem and the total-variation functions used in Rudin and Folland for bounded-variation analysis, the $\mathsf{TotalVariationFunction}^{\uparrow}$ packet records the cumulative variation route as finite BEDC data. It consumes the finite partition-variation surface of chapter 398, the positive and negative variation rows of chapter 2069, the terminal endpoint surfaces of chapter 38 and chapter 301, the dyadic tolerance ledger of chapter 291, the regulated-function handoff of chapter 1554, and the Riemann–Stieltjes consumer boundary of chapter 1637. The packet names cumulative variation as a displayed finite-prefix ledger, not as a host supremum over all partitions or as a quotient of real-valued functions.

Definition 2257.1 (Total variation function carrier)**.** *A* $\mathsf{TotalVariationFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, L, J, X, D, G, S, I, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet whose finite partition ledgers supply all edge-variation reads.*
2. *L is the partition-prefix ledger, listing the finite initial subpartitions used to read cumulative variation from the left endpoint to a displayed endpoint.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, exposing positive and negative variation ledgers for the same finite prefixes.*
4. *X records endpoint rows through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, including the left endpoint, the queried endpoint, and the terminal seal used by downstream consumers.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-observation ledger for endpoint differences, sign separation, and finite prefix sums.*
6. *G is the graph row for the cumulative total-variation function, read only as the finite sum of the positive and negative prefix ledgers displayed by J.*
7. *S is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer row, recording the finite regulated-function handoff obtained from the bounded-variation and cumulative-variation ledgers.*
8. *I is the* $\mathsf{RiemannStieltjes}^{\uparrow}$ *consumer row, recording that a Stieltjes-facing read of the integrator first consumes B, L, J, X, D, G, S.*

9. *H records componentwise* hsame *transport for the bounded-variation source, prefix ledger, Jordan rows, endpoint rows, dyadic observations, graph row, consumer rows, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a consumer request through the finite prefix ledger, dyadic edge observations, Jordan split, cumulative graph row, and regulated or Stieltjes handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TotalVariationFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a completed variation measure, a supremum over all partitions, host function equality, quotient real-valued functions, selected representatives, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.TotalVariationFunctionUp.TotalVariationFunctionTasteGate single carrier alignment`

Theorem 2257.2 (Total variation function NameCert obligations)**.** *For every accepted* $\mathsf{TotalVariationFunction}^{\uparrow}$ *carrier* $T = (B, L, J, X, D, G, S, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TotalVariationFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, finite partition-prefix ledger,* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *positive and negative variation rows,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, cumulative graph row,* $\mathsf{RegulatedFunction}^{\uparrow}$ *and* $\mathsf{RiemannStieltjes}^{\uparrow}$ *consumer rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every cumulative variation read exported by the packet factors through

$$\mathsf{BoundedVariation}^{\uparrow},\ L,\ \mathsf{JordanVariationDecomposition}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ G,\ \mathsf{RegulatedFunction}^{\uparrow}$$

No local obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2257.1. The projection exposes exactly $B, L, J, X, D, G, S, I, H, C, P, N$.

The cumulative variation read starts with the bounded-variation source B and the finite prefix ledger L. The Jordan row J then separates each displayed edge into positive and negative variation rows, while X and D supply the endpoint names and dyadic observations used by those finite sums.

The graph row G stores only the resulting finite prefix sum. Downstream regulated-function and Stieltjes consumers must read S or I after B, L, J, X, D, G have been displayed, and the structural rows H, C, P, N only transport, replay, source, and name that same route.

Since no carrier coordinate stores all partitions at once, host function equality, selected representatives, a variation measure, choice, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2257.3 (Total variation function regulated handoff)**.** *Let* $T = (B, L, J, X, D, G, S, I, H, C, P, N)$ *be an accepted* $\mathsf{TotalVariationFunction}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedFunction}^{\uparrow}$ *or* $\mathsf{RiemannStieltjes}^{\uparrow}$ *consumer read of the cumulative total-variation function first reads the displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *source, then the finite partition-prefix ledger, then the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *positive and negative variation rows, then the* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, and only then reaches the graph row G and consumer rows S, I.*

Consequently the handoff supplies a finite cumulative-variation route, not a completed variation measure, quotient function object, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a regulated-function or Stieltjes consumer of T. By the carrier definition, the only consumer rows are S and I, and both are downstream of the graph row G.

The graph row is itself downstream of the finite route B, L, J, X, D: the bounded-variation packet supplies the edge ledgers, the prefix ledger selects the displayed initial subpartitions, the Jordan row splits positive and negative variation, and the endpoint and dyadic rows provide the named observations for the finite sums.

The replay row C records the same ordering for every accepted consumer request, while H, P, N preserve transport, provenance, and local naming. Thus no consumer can bypass the finite prefix route through a host variation measure, quotient function equality, choice principle, or ambient completeness claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotalVariationFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{TotalVariationFunction}^{\uparrow}$ packet is generated from a $\mathsf{BoundedVariation}^{\uparrow}$ source packet, finite partition-prefix ledgers, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ positive and negative variation rows, $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ endpoint rows, $\mathsf{DyadicRatCore}^{\uparrow}$ edge observations, cumulative graph data, $\mathsf{RegulatedFunction}^{\uparrow}$ and $\mathsf{RiemannStieltjes}^{\uparrow}$ consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalVariationFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2257.1, Definition 2257.2, and Definition 2257.3 seed the finite cumulative total-variation route over bounded-variation, Jordan-split, endpoint, dyadic, regulated-function, and Stieltjes consumer rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalVariationFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed variation measures, suprema over all partitions, quotient function equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-prefix exactness, Jordan positive and negative variation routing, endpoint and dyadic observation routing, regulated-function handoff, Stieltjes handoff, transport, replay, provenance, and local naming for BEDC.Derived.TotalVariationFunctionUp.

Chapter 2258

A Concrete Naming Certificate for $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$

The $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$ packet realizes the GroundCompiler audit-map boundary of chapter 9407 as one finite route object. It sits beside the event-flow route of chapter 830, the audit-map family surface of chapter 811, the local MetaCIC critical-pair boundary of chapter 2212, and the finite-window pullback of chapter 2240. The packet stores the rows by which an audit consumer sees event flow, MetaCIC boundary references, no-hidden-input evidence, transport, replay, provenance, and local naming without treating a GroundCompiler report as a MetaCIC theorem.

Falsifiable prediction: Every theorem, generated recognizer, compiler report, or MetaCIC-facing export admitted by a $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$ packet is accompanied by displayed event-flow, no-hidden-input, boundary, transport, replay, provenance, and local NameCert rows; no hidden input or MetaCIC theorem is exported without those rows.

Independence witness: Nearest siblings: groundcompilereventflowauditroute, metacicauditmapfamily, metacic-confluence-critical-pair-boundary, and finite-window-pullback. The audit-route packet is not bijective to any one sibling because it jointly carries event-flow routing, MetaCIC-boundary references, no-hidden-input separation, transport, replay, provenance, and local naming for one GroundCompiler export path.

Definition 2258.1 (GroundCompiler audit route carrier)**.** *A* $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (E, M, B, Q, T, R, V, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}$ *event-flow row, including the displayed channel, generated-recognizer, certificate-gate, and non-evidence boundary.*
2. *M is the* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *boundary row naming which MetaCIC audit-map family entries are referenced by the GroundCompiler route.*
3. *B is the critical-pair boundary row, read through* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *when the route mentions confluence-facing MetaCIC data.*
4. *Q is the no-hidden-input row, stating that every accepted export path is sourced by displayed event-flow and channel rows rather than host compiler state, source prose, reports, metrics, or motifs.*

5. *T is the transport row connecting event-flow export names with the referenced MetaCIC boundary names without converting either side into the other.*
6. *R is the replay row for generated recognizer, certificate-gate, report-boundary, MetaCIC-boundary, and no-hidden-input reads.*
7. *V is the* $\mathsf{FiniteWindowPullback}^{\uparrow}$ *local-window row used when an audit route reads a finite observation window before exposing a certificate-facing export.*
8. *H records componentwise* hsame *transport for event-flow, MetaCIC-family, critical-pair-boundary, no-hidden-input, route-transport, replay, finite-window, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from event-flow input through channel readback, generated-recognizer mediation, no-hidden-input separation, optional MetaCIC-boundary reference, finite-window pullback, and certificate-gate export.*
10. *P is the* Pkg *provenance over* $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$, $\mathsf{GroundCompilerEventFlowAuditRoute}^{\uparrow}$, $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{FiniteWindowPullback}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditRoute}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for hidden compiler input, unchecked host reports, proof evidence extracted from prose, substitution of a GroundCompiler route for a MetaCIC theorem, or substitution of a MetaCIC theorem for a GroundCompiler certificate gate.

Lean4 ✓ `BEDC.Derived.GroundCompilerAuditRouteUp.GroundCompilerAuditRouteTasteGate single carrier alignment`

Theorem 2258.2 (GroundCompiler audit route NameCert obligations)**.** *For every accepted* GroundCompilerAuditR *carrier $A = (E, M, B, Q, T, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditRoute}^{\uparrow}}$ *surface consists exactly of carrier admission for the event-flow route, MetaCIC audit-map-family boundary, critical-pair boundary reference, no-hidden-input row, route-transport row, replay row, finite-window pullback row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

An accepted audit export may be consumed only through the displayed route

$$E \longmapsto Q \longmapsto T, R \longmapsto M, B, V \longmapsto H, C, P, N.$$

No GroundCompiler report, motif, metric, host compiler state, source prose, or MetaCIC theorem is exported outside that route.

Proof. Carrier admission is projection from Definition 2258.1. The projection exposes exactly $E, M, B, Q, T, R, V, H, C, P, N$.

The event-flow row E is read before any export-facing boundary. The no-hidden-input row Q then separates displayed channel readback from host compiler state, reports, metrics, motifs, and source prose.

The transport and replay rows T, R connect the GroundCompiler route to optional MetaCIC-family, critical-pair, and finite-window rows without replacing any one row by another. The structural rows H, C, P, N transport, replay, source, and name the same finite path.

Since no carrier coordinate converts reports or MetaCIC boundary references into proof evidence, the listed rows exhaust the local NameCert surface. □

Theorem 2258.3 (GroundCompiler audit route MetaCIC boundary)**.** *Let $A = (E, M, B, Q, T, R, V, H, C, P, N)$ be an accepted* $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$ *carrier. Any consumer that reads a MetaCIC-facing reference from A must first read the displayed GroundCompiler event-flow row E, the no-hidden-input row Q, and the route-transport row T; it then reads the* $\mathsf{MetaCICAuditMapFamily}^{\uparrow}$ *row M or the* $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$ *row B as a boundary reference only.*

Consequently the packet does not turn a GroundCompiler audit route into subject reduction, normalization, confluence, closed consistency, or arbitrary MetaCIC theorem export; it also does not turn a MetaCIC boundary reference into a GroundCompiler certificate-gate export without the displayed event-flow and no-hidden-input rows.

Proof. Start with a MetaCIC-facing consumer of A. The carrier definition gives access to MetaCIC data only through M and B.

The NameCert obligations force the consumer to read E, Q, T before those boundary rows. Thus the GroundCompiler event-flow source and no-hidden-input separation are already displayed when a MetaCIC reference is inspected.

The replay row R and structural rows H, C, P, N preserve this ordering. They do not contain constructors for subject reduction, normalization, confluence, closed consistency, or arbitrary theorem export, and they do not replace the certificate gate of E by the MetaCIC boundary rows M or B. □

Future Lean TasteGate work for this packet must supply `Nontrivial GroundCompilerAuditRouteUp` and `FieldFaithful GroundCompilerAuditRouteUp` instances, so the event-flow, no-hidden-input, MetaCIC-boundary, finite-window, transport, replay, provenance, and local naming rows are checked as genuine carrier fields.

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerAuditRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerAuditRoute}^{\uparrow}$ packet is generated from a GroundCompiler event-flow audit route, MetaCIC audit-map-family boundary row, critical-pair boundary row, no-hidden-input row, route-transport row, replay row, finite-window pullback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerAuditRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2258.1, Definition 2258.2, and Definition 2258.3 seed the finite GroundCompiler audit route over event-flow, MetaCIC-boundary, no-hidden-input, transport, replay, finite-window, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerAuditRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compiler correctness, proof reconstruction from reports or source prose, subject reduction, normalization, confluence, closed consistency, arbitrary MetaCIC theorem export, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, event-flow exposure, no-hidden-input separation, MetaCIC-boundary routing, finite-window routing, transport, replay, provenance, local naming, and TasteGate nontriviality and field faithfulness for BEDC.Derived.GroundCompilerAuditRouteUp.

Chapter 2259

A Concrete Naming Certificate for $\mathsf{CauchyIntegralTest}^{\uparrow}$

Following Cauchy's integral-test tradition and the constructive real analysis of Bishop and Bridges, the $\mathsf{CauchyIntegralTest}^{\uparrow}$ packet records a finite comparison route from nonnegative monotone real-function data to a regular Cauchy series-tail seal. It consumes the $\mathsf{Real}^{\uparrow}$ readback surface of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ rational approximation corridor of chapter 301, finite observation schedules from chapter 126, and integral-window rows from chapter 1460 and chapter 1526. The packet names the textbook tail comparison as finite NameCert data; it does not add measure theory, Lebesgue integration, countable choice, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 2259.1 (CauchyIntegralTest carrier)**.** *A* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, I, S, W, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Real}^{\uparrow}$ *nonnegative monotone function row. It records finite real-name observations of the sampled function and the displayed monotone comparison between inspected tail points.*
2. *I is the integral tail-window row. It records Darboux or Riemann interval sums over inspected tails, together with the finite comparison between consecutive tail windows.*
3. *S is the regular Cauchy series-tail row. It records partial-sum tail windows and the displayed comparison to the integral tail windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row selecting the finite function samples, interval windows, and series-tail windows used by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to function values, integral bounds, and series tails.*
6. *H records componentwise* hsame *transport for the function row, integral-window row, series-tail row, stream schedule, regular rational readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a consumer request through F, I, S, W, R and then to H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyIntegralTest}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *regular series-tail data,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for a host improper integral, a Lebesgue integral, an arbitrary choice of tail witnesses, a hidden completeness theorem, or quotient equality of real functions.

Definition 2259.2 (CauchyIntegralTest classifier)**.** *Two accepted* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *packets classify together exactly when their displayed rows are componentwise transported by* hsame*: the nonnegative monotone* $\mathsf{Real}^{\uparrow}$ *observations match, the integral tail-window rows match, the regular Cauchy series-tail rows match, the* $\mathsf{StreamName}^{\uparrow}$ *schedules and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks match, and the replay, provenance, and local naming rows match. No additional classifier row compares external functions, host integrals, or completed real sums.*

Theorem 2259.3 (CauchyIntegralTest NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *carrier* $K = (F, I, S, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralTest}^{\uparrow}}$ *surface consists exactly of carrier admission, nonnegative monotone* $\mathsf{Real}^{\uparrow}$ *observation, Darboux or Riemann tail-window exposure, regular Cauchy series-tail exposure,* $\mathsf{StreamName}^{\uparrow}$ *finite observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The row F exposes only the nonnegative monotone real-name observations needed for the integral-test comparison, while the row I exposes the inspected integral tail windows.

The row S records the series-tail windows and their comparison to I. The rows W and R are the only routes through which the finite observations and rational readbacks are inspected.

Finally H, C, P, N transport, replay, source, and locally name the same displayed route. Since no row carries a host improper integral, a measure-theoretic integral, a choice functional, or a completeness theorem for $\mathsf{Real}^{\uparrow}$, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyIntegralTestUp.CauchyIntegralTestCarrier namecert obligations`

Theorem 2259.4 (CauchyIntegralTest tail boundary)**.** *Every accepted* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *tail comparison factors through the finite route*

$$F, W \longmapsto I, R \longmapsto S$$

before it can be used as convergence-test evidence. Thus the packet exports only a displayed tail-window boundary, not an ambient theorem that all monotone real functions have improper integrals.

Proof. Read the function samples through F and the selected observation schedule through W. These rows determine the inspected monotone tail windows.

The integral row I and readback row R expose the finite Darboux or Riemann bounds attached to those windows. Only after those rows are visible may the series-tail row S be compared against them.

Therefore a consumer that bypasses F, W, I, R or treats the route as a host improper-integral theorem leaves the accepted carrier surface. □

Theorem 2259.5 (CauchyIntegralTest series handoff)**.** *The* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *packet hands convergence evidence to the regular Cauchy series-tail corridor only through the displayed row* S *together with the readback rows* W *and* R*. It does not choose a completed series sum, assert* $\mathsf{Real}^{\uparrow}$ *completeness, or identify integral tails with real limits outside the packet.*

Proof. The handoff target is the regular Cauchy series-tail row S. The finite windows and rational approximations that make S readable are supplied by W and R.

The carrier contains no row for a selected infinite sum and no row for a global completeness theorem. The packet therefore supplies a convergence-test handoff to the series-tail corridor, while leaving completed sums and completeness principles outside the named surface. □

Theorem 2259.6 (CauchyIntegralTest monotone-tail induction)**.** *Let $K = (F, I, S, W, R, H, C, P, N)$ be an accepted* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *carrier. A monotone nonnegative tail window accepted by the carrier can be advanced by one finite observation step while preserving the integral-window row I, the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows W, R, and the* $\mathsf{Real}^{\uparrow}$ *seal route carried by the same structural rows. The induction step exports no Lebesgue measure, selected improper integral value, countable choice, quotient equality of functions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 2259.1. The function row F records the monotone nonnegative observations, and the stream row W records the finite observation schedule that may be extended by one displayed tail point.

Apply the tail boundary theorem of Definition 2259.4. The advanced tail point is read through F, W before the integral-window row I and the readback row R are inspected, so the new observation remains inside the same finite comparison corridor.

Use the series handoff of Definition 2259.5. The series-tail row S receives convergence evidence only through W, R, and the structural rows H, C, P, N transport, replay, source, and name that same handoff after the one-step advance.

Because the carrier has no coordinate for Lebesgue measure, selected improper integral values, countable choice, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the induction step preserves exactly the displayed BEDC route and exports none of those principles. □

Falsifiable prediction: An accepted $\mathsf{CauchyIntegralTest}^{\uparrow}$ packet that reaches a series-tail seal without displaying the nonnegative monotone $\mathsf{Real}^{\uparrow}$ row, a Darboux or Riemann integral tail-window row, finite $\mathsf{StreamName}^{\uparrow}$ observations, and $\mathsf{RegSeqRat}^{\uparrow}$ readback fails the carrier surface.

Independence witness: Real, RegSeqRat, StreamName, DarbouxIntegral, RiemannIntegral, regular series-tail rows: none is bijective to $\mathsf{CauchyIntegralTest}^{\uparrow}$, because the listed siblings separately supply real observations, finite schedules, rational readback, integral windows, or series-tail evidence while this packet binds them to one constructive integral-test route with transport, replay, provenance, and local naming.

The Lean TasteGate follow-up is `BEDC.Derived.CauchyIntegralTestUp`, with explicit Nontrivial CauchyIntegra and FieldFaithful $\mathsf{CauchyIntegralTest}^{\uparrow}$ instances preserving $F, I, S, W, R, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntegralTest}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntegralTest}^{\uparrow}$ packet is generated from a nonnegative monotone RealUp observation row, a Darboux or Riemann integral tail-window row, a regular Cauchy series-tail row, finite StreamNameUp observations, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntegralTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2259.1, Definition 2259.2, Definition 2259.3, Definition 2259.4, and Definition 2259.5 seed the finite integral-test carrier, classifier, tail boundary, and series handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntegralTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure theory, Lebesgue integration, selected improper integrals, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, tail-window boundary, series-tail handoff, nontriviality, and field faithfulness for BEDC.Derived.CauchyIntegralTestUp.

Chapter 2260

A Concrete Naming Certificate for $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$

The $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ packet records the traditional genetic-code four-base alphabet as a conservative finite readback over BEDC finite-data carriers. It uses the two-endpoint $\mathsf{Bool}^{\uparrow}$ surface of chapter 32 and the componentwise product discipline of chapter 34, while its source vision is the DNA four-phase bridge of subsection 9462.28.16. The packet names a finite external alphabet bridge; it does not prove a molecular-biology theorem, identify BEDC channel glyphs with bases, or turn biological bases into BEDC primitives.

Definition 2260.1 (DNAFourPhaseAlphabetBridge carrier)**.** *A* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *endpoint row. It supplies the two finite axes used by the readback.*
2. *Q is the* $\mathsf{Prod}^{\uparrow}$ *quadrant row. It records the four displayed pairs of Bool endpoints without adding a fifth phase or quotienting any pair.*
3. *A is the external alphabet row. It names the target-side symbols* A, C, G, *and* T *only as readback labels.*
4. *E is the bridge-exactness row. It records that each displayed quadrant has exactly one external alphabet label and that each external label is read from exactly one displayed quadrant.*
5. *H records componentwise* hsame *transport for the Bool axes, product quadrant, external alphabet label, exactness, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the route*

$$B, Q \longmapsto A, E \longmapsto H, P, N.$$

7. *P is the* Pkg *provenance over* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, *the source vision bridge,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets through B, Q, A, E, H, C, P, N. It has no coordinate for molecular identity, biological translation correctness, mutation dynamics, channel-glyph equality, or a global genetic-code theorem.

Lean4 ✓ `BEDC.Derived.DNAFourPhaseAlphabetBridgeUp.DNAFourPhaseAlphabetBridgeUpTasteGate single carrier alignment`

Definition 2260.2 (DNAFourPhaseAlphabetBridge classifier)**.** *Two accepted* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ *packets classify together exactly when their Bool endpoint rows, product quadrant rows, external alphabet rows, exactness rows, transport rows, replay rows, provenance rows, and local naming rows classify componentwise by* hsame*. The external alphabet labels are compared only as the target-side labels recorded by the bridge row; they are not BEDC source constructors.*

Theorem 2260.3 (DNAFourPhaseAlphabetBridge NameCert obligations)**.** *For every accepted* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ *carrier* $D = (B, Q, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}}$ *surface consists exactly of Bool endpoint admission, product quadrant formation, four-label external alphabet readback, quadrant-label exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DNAFourPhaseAlphabetBridgeUp.DNAFourPhaseAlphabetBridgeTasteGate single carrier alignment`

Proof. Project the accepted carrier. The row B supplies only the two finite Bool axes, and the row Q forms their product quadrants through the existing product discipline.

The row A names the four external alphabet labels, while E records the finite readback exactness between quadrants and labels. These rows are target-side readbacks and do not create source constructors for biological bases.

Finally H, C, P, N transport, replay, source, and locally name the same route. Since the carrier contains no molecular identity row, no translation-correctness row, and no channel-glyph identification row, the listed obligations exhaust the local certificate surface. □

Theorem 2260.4 (DNAFourPhaseAlphabetBridge readback boundary)**.** *Every accepted* DNAFourPhaseAlphabetB *readback factors through*

$$B, Q \longmapsto A, E$$

before it can be used as four-phase alphabet evidence. Thus the packet exports a conservative finite readback boundary, not an identification between BEDC channel glyphs and DNA bases.

Proof. Read B to obtain the two finite axes and Q to obtain the four displayed product quadrants. These rows are internal finite-data rows.

Only after those rows are displayed may the external alphabet row A and exactness row E be inspected. The bridge therefore keeps internal finite carriers and external biological labels in separate rows.

Any consumer that treats the labels as BEDC channel glyphs or as a molecular-biology theorem bypasses the accepted carrier boundary. □

Theorem 2260.5 (DNAFourPhaseAlphabetBridge quadrant exhaustion)**.** *Every accepted* DNAFourPhaseAlphabet *carrier* $D = (B, Q, A, E, H, C, P, N)$ *exposes exactly four* $\mathsf{Bool}^{\uparrow} \times \mathsf{Bool}^{\uparrow}$ *product quadrants through* Q*. Each displayed quadrant is read by exactly one of the external labels in* A*, and every such label is consumed only through the exactness, transport, replay, provenance, and local naming rows* E, H, C, P, N*.*

The exhaustion exports no biological translation theorem, no molecular identity, and no identity between BEDC channel glyphs and the external labels A, C, G, *and* T.

Proof. Project the accepted carrier. The row B supplies only the two finite Bool axes, and the product row Q records their four displayed ordered pairs.

Use the classifier of Definition 2260.2. It compares packets by the Bool endpoint rows, product quadrant rows, external alphabet rows, exactness rows, transport rows, replay rows, provenance rows, and local naming rows, so there is no hidden fifth quadrant or quotient of two quadrants.

The external alphabet row A supplies exactly the four target-side labels, while E records the one-to-one finite readback between those labels and the quadrants in Q. The remaining rows H, C, P, N are the only downstream consumers of that readback.

Since no row of the carrier records molecular identity, biological translation correctness, mutation dynamics, or channel-glyph equality, the theorem is a finite exhaustion statement about the accepted bridge and not an exported biology or glyph-identification theorem. □

Falsifiable prediction: An accepted $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ packet that identifies BEDC channel glyphs with A, C, G, or T fails the classifier row, because the accepted carrier records only Bool endpoint axes, product quadrants, external alphabet labels, exactness, transport, replay, provenance, and local naming.

Independence witness: Bool, Prod, channel genetic-code bridge: $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ is not bijective to any listed sibling because it binds two finite Bool axes and product quadrant formation to an external four-label readback while preserving the separation between BEDC finite carriers and biological alphabet labels.

The Lean TasteGate follow-up is `BEDC.Derived.DNAFourPhaseAlphabetBridgeUp`, with explicit $\mathsf{Nontrivial\ DNAFourPhaseAlphabetBridge}^{\uparrow}$ and $\mathsf{FieldFaithful\ DNAFourPhaseAlphabetBridge}^{\uparrow}$ instances preserving B, Q, A, E, H, C, P, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ packet is generated from BoolUp endpoint axes, a ProdUp quadrant row, a four-label external DNA alphabet readback row, bridge exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2260.1, Definition 2260.2, Definition 2260.3, and Definition 2260.4 seed the finite four-phase alphabet bridge and its readback boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim molecular-biology theorems, biological translation correctness, channel-glyph identification, mutation dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, quadrant-label exactness, readback boundary, nontriviality, and field faithfulness for BEDC.Derived.DNAFourPhaseAlphabetBridgeUp.

Chapter 2261

A Concrete Naming Certificate for $\mathsf{UniformSequentialContinuity}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformSequentialContinuity}^{\uparrow}$ packet records the finite route by which one displayed uniform modulus transports regular Cauchy sequence data through a continuous map before the $\mathsf{Real}^{\uparrow}$ seal is read. It sits between the map surface of chapter 130, the uniform-modulus row of chapter 369, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38. The packet names a uniform sequential handoff only; it does not assert host sequential continuity, function extensionality, quotient stream equality, countable choice, selected limits, or ambient real completeness.

Definition 2261.1 (Uniform sequential continuity carrier)**.** *A* $\mathsf{UniformSequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, S, W_X, W_Y, R_X, R_Y, D_X, D_Y, E_X, E_Y, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the displayed source-to-target map and its local classifier interface.*
2. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *row. It converts every requested target tolerance into a source tolerance before any target sequence read is consumed.*
3. *S is the source sequence surface exposed only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. *W_X and W_Y are the source and image finite-window rows. The row W_Y is read only by applying F to the windows selected by W_X.*
5. *R_X and R_Y are the source and target* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows attached to the displayed windows.*
6. *D_X and D_Y are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers. The target ledger D_Y records the requested precision and the source ledger D_X records the modulus-returned precision.*
7. *E_X and E_Y are the source and target* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the corresponding stream, regular-rational, and dyadic rows have been displayed.*
8. *H records componentwise* hsame *transport for the map row, modulus row, stream surface, window rows, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a target tolerance request through $M, D_X, W_X, R_X, F, W_Y, R_Y, D_Y$ before E_Y is exposed.*
10. *P is the* Pkg *provenance over* UniformSequentialContinuity$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformSequentialContinuity}^\uparrow}$ *row.*

The classifier compares accepted packets only through the displayed tuple $F, M, S, W_X, W_Y, R_X, R_Y, D_X, D_Y, E_X, E$ The carrier has no coordinate for host sequential continuity, function extensionality, quotient stream equality, countable choice, selected limits, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformSequentialContinuityUp.UniformSequentialContinuityTasteGate single carrier alignment`

Theorem 2261.2 (Uniform sequential continuity NameCert obligations)**.** *For every accepted* UniformSequentialContinuity$^\uparrow$ *carrier*

$$U = (F, M, S, W_X, W_Y, R_X, R_Y, D_X, D_Y, E_X, E_Y, H, C, P, N),$$

the local NameCert$_{\text{UniformSequentialContinuity}^\uparrow}$ *surface consists exactly of carrier admission, displayed continuous-map use, uniform-modulus tolerance conversion, source and image finite-window exposure, source and target* RegSeqRat$^\uparrow$ *handoff, dyadic tolerance accounting, source and target* Real$^\uparrow$ *seal ordering, componentwise transport, displayed replay, provenance, local naming, and non-escape from host sequential continuity, function extensionality, quotient stream equality, countable choice, selected limits, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 2261.1. The projection exposes the displayed map, uniform modulus, source stream surface, finite windows, regular readbacks, dyadic ledgers, Real seals, and structural rows.

A target tolerance request first reads the modulus row M. This produces the source-side dyadic tolerance in D_X before any image-window read occurs, so the route cannot start from a detached target limit or a host sequential-continuity theorem.

The source sequence is then inspected through S, W_X, R_X, D_X, E_X. The image sequence is obtained by applying the displayed map row F to the same finite source windows, producing W_Y and then the target handoff through R_Y, D_Y, E_Y.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since no excluded object is a carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 2261.3 (Uniform sequential continuity Cauchy handoff)**.** *Every accepted* UniformSequentialContinuity$^\uparrow$ *packet preserves regular Cauchy sequence data through the displayed uniform modulus before either* Real$^\uparrow$ *seal is consumed. Equivalently, every target-seal-facing route factors through*

$$D_Y \longmapsto M \longmapsto D_X \longmapsto W_X \longmapsto R_X \qquad \textit{and} \qquad W_X, F \longmapsto W_Y \longmapsto R_Y \longmapsto E_Y$$

together with S, E_X, H, C, P, N.

Proof. Start with the target precision request recorded in D_Y. The uniform-modulus row M converts that request into the source precision recorded in D_X, so the source sequence is inspected at a tolerance already tied to the target demand.

The source route then reads the finite stream windows W_X from S and the regular rational handoff R_X before the source seal row E_X can be used. This is the displayed regular Cauchy data available to the map row.

Apply the continuous-map row F to the displayed source windows. The image windows W_Y are therefore downstream of the same source inspection, and the target regular-rational handoff R_Y precedes the target seal E_Y.

The rows H, C, P, N preserve the route under transport, replay, provenance, and local naming. Thus no Real-facing consumer can bypass the uniform modulus, the source regular Cauchy handoff, or the target regular Cauchy handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSequentialContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformSequentialContinuity}^{\uparrow}$ packet is generated from a displayed ContinuousMap row, a UniformModulus row, source and image StreamName windows, source and target RegSeqRat readbacks, DyadicRatCore tolerance ledgers, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSequentialContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2261.1, Definition 2261.2, and Definition 2261.3 seed the finite uniform sequential continuity handoff from a modulus-controlled source regular Cauchy route to the target Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSequentialContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sequential continuity, function extensionality, quotient stream equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displayed map use, uniform-modulus tolerance conversion, source and image window exactness, RegSeqRat handoff, dyadic ledger accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.UniformSequentialContinuityUp.

Chapter 2262

A Concrete Naming Certificate for $\mathsf{SmithNormalForm}^{\uparrow}$

Following Smith's determinantal reduction and Newman's normal-form treatment, the $\mathsf{SmithNormalForm}^{\uparrow}$ packet records the finite row-and-column certificate by which an integer or principal-ideal matrix is reduced to a displayed diagonal divisibility chain. It consumes the matrix carrier of chapter 49, the finite module-row discipline of chapter 46, the ideal membership surface of chapter 81, and the Dedekind-domain naming rows of chapter 172. The chapter names a finite certificate for diagonal reduction and divisibility bookkeeping; it does not claim quotient-module classification, a choice-selected basis, a host principal-ideal-domain theorem, or equality outside displayed row and column operations.

Definition 2262.1 (Smith normal form carrier)**.** *A* $\mathsf{SmithNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, R, C, D, V, E, U, H, T, P, N)$$

with the following displayed rows:

1. *M is the source matrix row, recording the finite matrix entries, row and column bounds, coefficient carrier, and matrix-operation ledger inherited from* $\mathsf{Mat}^{\uparrow}$*.*
2. *R is the elementary row-operation trace. It lists each row swap, row scaling by a displayed unit, and row addition by a displayed multiple, together with the matrix row it transforms.*
3. *C is the elementary column-operation trace, with the same visible data for column swaps, unit scalings, and displayed multiple additions.*
4. *D is the diagonal readback row, listing the terminal diagonal entries and the zero-off-diagonal witnesses exposed after the combined R, C trace.*
5. *V is the divisibility-chain row. It records, for every adjacent pair of nonzero diagonal entries, the displayed ideal-membership or divisibility witness that the earlier diagonal entry divides the later one.*
6. *E is the exactness row. It records that every intermediate matrix is obtained from the previous one by the displayed elementary operation and that the terminal readback is the matrix named by D.*
7. *U is the unit-and-ideal ledger, exposing the unit rows, principal ideal rows, and Dedekind-compatible ideal containment checks used by R, C, V.*

8. *H records componentwise* hsame *transport for matrix entries, row traces, column traces, diagonal readback, divisibility witnesses, exactness rows, unit and ideal ledgers, continuation rows, provenance, and naming.*
9. *T records displayed* Cont *replay from source matrix inspection through elementary-operation playback, diagonal readback, divisibility verification, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{SmithNormalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, $\mathsf{Dedekind}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SmithNormalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient modules, selected external bases, hidden principal ideal choices, ambient noetherian arguments, or host equality outside the displayed elementary-operation trace.

Lean4 ✓ `BEDC.Derived.SmithNormalFormUp.SmithNormalFormTasteGate single carrier alignment`

Definition 2262.2 (Smith normal form classifier)**.** *Two accepted* $\mathsf{SmithNormalForm}^{\uparrow}$ *packets*

$$S = (M, R, C, D, V, E, U, H, T, P, N) \quad \textit{and} \quad S' = (M', R', C', D', V', E', U', H', T', P', N')$$

are classifier-equivalent exactly when the eleven displayed rows match componentwise under the local hsame *surface: the source matrices match through the same* $\mathsf{Mat}^{\uparrow}$ *carrier, the elementary row and column traces match step by step, the terminal diagonal rows agree entrywise, the divisibility witnesses in V, V' name the same adjacent divisibility obligations, the exactness and unit-ideal rows agree, and the transport, continuation, provenance, and naming rows replay the same finite data. The classifier ignores no additional ambient module quotient, selected basis, or external diagonal-form uniqueness theorem.*

Theorem 2262.3 (Smith normal form divisibility chain)**.** *For every accepted* $\mathsf{SmithNormalForm}^{\uparrow}$ *carrier $S = (M, R, C, D, V, E, U, H, T, P, N)$, any exported diagonal normal-form claim factors through the displayed divisibility route*

$$M, R, C \longmapsto D, E, U \longmapsto V, H, T, P, N.$$

In particular, the claim that the diagonal entries form a Smith chain is readable only from the adjacent divisibility witnesses listed in V and the unit-and-ideal ledger U.

Proof. Open the carrier of Definition 2262.1. The source matrix row M and the elementary traces R, C are the only rows that can produce a terminal matrix for this packet.

The diagonal row D is read only after the exactness row E has replayed every elementary operation. The unit-and-ideal ledger U supplies the visible unit rows and ideal-containment checks that justify the divisibility witnesses.

Finally, V lists the adjacent divisibility witnesses for the terminal diagonal entries. Since H, T, P, N transport, replay, source, and name only those rows, any downstream Smith-chain claim must pass through the displayed route and cannot appeal to an external normal-form theorem. □

Theorem 2262.4 (Smith row-column equivalence)**.** *For every accepted* $\mathsf{SmithNormalForm}^{\uparrow}$ *carrier $S = (M, R, C, D, V, E, U, H, T, P, N)$, the source matrix named by M and the diagonal matrix named by D are equivalent only by the finite row and column traces R and C together with the exactness row E. No quotient-module equality, selected basis identification, or ambient isomorphism theorem is exported by the packet.*

Proof. Read the elementary traces in order. Each row of R and each row of C names one displayed elementary operation and the matrix row it transforms, so the trace itself supplies the equivalence witness between successive matrices.

The exactness row E ties those local witnesses to the terminal diagonal readback D. Thus the only exported equivalence is the finite playback from M through R, C to D.

The classifier of Definition 2262.2 compares the same rows componentwise. Therefore the packet has no remaining coordinate from which quotient-module equality, selected bases, or an ambient classification theorem could be read. □

Theorem 2262.5 (Smith normal form NameCert obligations)**.** *For every accepted* $\mathsf{SmithNormalForm}^{\uparrow}$ *carrier* $S = (M, R, C, D, V, E, U, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SmithNormalForm}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source matrix, row trace, column trace, diagonal readback, divisibility witnesses, exactness row, unit-and-ideal ledger, transport, continuation, provenance, and local naming;*
2. *matrix-source compatibility, requiring every entry read by* R, C, D, E, V, U *to be supplied by the displayed* $\mathsf{Mat}^{\uparrow}$ *source row* M*;*
3. *elementary-operation exactness, requiring every row and column operation to name its source row, target row, coefficient or unit witness, and resulting matrix entry updates;*
4. *diagonal readback exactness, requiring* D *to expose every diagonal entry and every zero-off-diagonal witness after the finite trace;*
5. *divisibility-chain exactness, requiring* V *and* U *to display the unit, ideal-membership, and containment evidence for each adjacent diagonal divisibility claim;*
6. *module, ideal, and Dedekind provenance, requiring the finite rows used by* M, R, C, D, V, E, U *to be sourced through the cited* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{Ideal}^{\uparrow}$*, and* $\mathsf{Dedekind}^{\uparrow}$ *packets when those algebraic surfaces are invoked;*
7. *ledger preservation, requiring* H, T, P, N *to preserve exactly the displayed rows under componentwise transport, continuation replay, package provenance, and local naming.*

No Smith-normal-form obligation is stored outside these rows.

Proof. Project the carrier rows. Carrier admission gives precisely $M, R, C, D, V, E, U, H, T, P, N$, so a local certificate cannot obtain extra data from a host module quotient, selected basis, or hidden principal-ideal argument.

Matrix-source compatibility is forced because every row and column operation reads entries from M. Elementary-operation exactness is forced because R and C list each operation, its displayed coefficient or unit row, and the resulting matrix updates.

The diagonal and divisibility obligations are then separate visible checks. The row D exposes the terminal diagonal entries and zero-off-diagonal witnesses, while V and U expose adjacent divisibility, unit, and ideal-containment evidence.

Finally, the provenance row P records exactly which $\mathsf{Mat}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{Ideal}^{\uparrow}$, and $\mathsf{Dedekind}^{\uparrow}$ surfaces are being consumed. The structural rows H, T, N transport, replay, and name the same finite packet. These rows exhaust the local NameCert surface. □

Falsifiable prediction: Every $\mathsf{SmithNormalForm}^{\uparrow}$ read factors through one $\mathsf{Mat}^{\uparrow}$ source row, one elementary row-operation trace, one elementary column-operation trace, one diagonal readback

row, one adjacent divisibility ledger, one exactness row, one unit-and-ideal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: matrix, module, ideal, dedekind: $\mathsf{SmithNormalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries elementary row and column playback, diagonal readback, adjacent divisibility witnesses, and unit-and-ideal exactness as one finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SmithNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{SmithNormalForm}^{\uparrow}$ packet is generated from a finite MatrixUp source row, elementary row and column traces, diagonal readback, adjacent divisibility witnesses, exactness rows, unit-and-ideal ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmithNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2262.1, Definition 2262.2, Definition 2262.3, Definition 2262.4, and Definition 2262.5 seed the finite Smith normal form certificate surface over displayed matrix, module, ideal, and Dedekind rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SmithNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-module classification, selected external bases, hidden principal-ideal choices, ambient noetherian arguments, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, elementary-operation playback, diagonal readback exactness, divisibility-chain exactness, unit-and-ideal ledger preservation, and TasteGate field faithfulness for BEDC.Derived.SmithNormalFormUp.

Chapter 2263

A Concrete Naming Certificate for $\mathsf{RadonMeasure}^{\uparrow}$

Following Radon and standard constructive measure theory, the $\mathsf{RadonMeasure}^{\uparrow}$ packet records regular finite measure values on compact metric and open-ball cover rows as finite BEDC data. It is anchored in chapter 92, chapter 129, chapter 57, chapter 2191, chapter 2166, and chapter 186. The chapter exposes a regular-measure surface for measure-facing analysis without importing host sigma-additivity, choice-selected approximants, quotient measure equality, or ambient standard-model measure theory.

2263.1 Radon Measure Carrier

Definition 2263.1 (RadonMeasure carrier)**.** *A* $\mathsf{RadonMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, M, O, K, V, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *surfaces.*
2. *M is the base finite measure row, read through the* $\mathsf{Measure}^{\uparrow}$ *event-value boundary.*
3. *O is the open-ball locator row naming the open-ball cover requests visible to the packet.*
4. *K is the compact inner approximant ledger for finite regular lower reads.*
5. *V is the finite cover outer approximant ledger, routed through finite Vitali cover rows when cover refinement is requested.*
6. *D is the distribution-facing value row exposed to downstream* $\mathsf{Distribution}^{\uparrow}$ *consumers.*
7. *H records componentwise* hsame *transport for $X, M, O, K, V, D, C, P, N$.*
8. *C records displayed* Cont *replay from compact metric source and base measure row through open-ball localization, compact inner approximation, finite cover outer approximation, distribution value, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{OpenBall}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RadonMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, O, K, V, D, H, C, P, N$. It has no coordinate for host sigma-additivity, choice-selected compact approximants, quotient equality of measures, an ambient measurable-space completion, or a standard-model measure object outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.RadonMeasureUp.RadonMeasureTasteGate single carrier alignment`

2263.2 Regularity Obligations

Theorem 2263.2 (RadonMeasure NameCert obligations)**.** *For every accepted* $\mathsf{RadonMeasure}^{\uparrow}$ *carrier* $R = (X, M, O, K, V, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RadonMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission through* X *and* M*, regularity classifier stability through* O, K, V*, inner approximation exactness through* K*, outer approximation exactness through* V*, finite-cover ledger visibility through the* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *handoff, distribution-facing value exposure through* D*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. It refuses host sigma-additivity, choice-selected approximants, quotient measure equality, and ambient standard-model measure theory as public coordinates.*

Proof. Project the carrier of Definition 2263.1. The source row X and measure row M are the only admitted compact metric and finite measure inputs.

The locator row O, compact inner ledger K, and finite cover outer ledger V are the only regularity classifier rows. Inner and outer exactness are therefore read from the displayed approximation ledgers rather than from a host regularity theorem.

The rows D, H, C, P, N expose distribution values, componentwise transport, replay, provenance, and local naming for the same finite packet. Since no remaining coordinate stores host sigma-additivity, choice-selected approximants, quotient measure equality, or an ambient standard-model measure object, the listed rows exhaust the local NameCert obligations. □

Theorem 2263.3 (RadonMeasure compact regularity route)**.** *Every accepted* $\mathsf{RadonMeasure}^{\uparrow}$ *packet routes a compact metric finite-cover request through the open-ball locator row* O*, the compact inner approximant ledger* K*, the finite Vitali cover outer ledger* V*, and the base measure value row* M *before any distribution consumer can read the value row* D*. A request that bypasses one of* O, K, V, M *is not an accepted RadonMeasure compact-regularity route.*

Proof. Read the replay row C and the provenance row P. They bind compact metric source data to open-ball localization, compact inner approximation, finite cover outer approximation, and base measure valuation before the distribution-facing row is exposed.

The transport row H preserves the same sequence componentwise. Thus a consumer can read D only after the route has displayed O, K, V, M inside the same packet.

Dropping any one of O, K, V, M leaves no finite row witnessing the corresponding localization, inner read, outer read, or measure value. Such a route is outside the accepted carrier. □

Lean4 ✓ `BEDC.Derived.RadonMeasureUp.RadonMeasureCarrier compact regularity route`

Theorem 2263.4 (RadonMeasure weak consumer boundary)**.** *Downstream* $\mathsf{Distribution}^{\uparrow}$ *and* $\mathsf{RieszRepresentation}^{\uparrow}$ *consumers of an accepted* $\mathsf{RadonMeasure}^{\uparrow}$ *packet read only the displayed regular measure packet* $R = (X, M, O, K, V, D, H, C, P, N)$*. They cannot infer host measure equality, unlisted integration theorems, choice-selected compact approximants, ambient sigma-additivity, or any representation theorem not displayed by the packet rows.*

Proof. The carrier exposes downstream access through D together with the replay, transport, provenance, and local naming rows. These rows record only finite regular-measure data over the displayed compact metric, open-ball, compact approximant, finite cover, and base measure inputs.

No carrier coordinate names host measure equality, an integration theorem, a selector for compact approximants, ambient sigma-additivity, or a representation theorem. Therefore a distribution or Riesz-facing consumer receives the weak displayed packet and no stronger host-theoretic consequence. □

Closure status for $\mathsf{NameCert}_{\mathsf{RadonMeasure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RadonMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2263.1, Definition 2263.2, Definition 2263.3, and Definition 2263.4 seed the finite regular-measure packet over compact metric sources, base measure rows, open-ball localization, compact inner approximants, finite cover outer approximants, distribution values, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RadonMeasure}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RadonMeasureUp`

Bridge status: none

Not claimed: This chapter does not close host sigma-additivity, quotient measure equality, selected approximants, ambient standard-model measure theory, Lean verification of the regularity obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regularity classifier stability, inner and outer approximation exactness, finite-cover ledger visibility, weak consumer boundary, and compact regularity routing for BEDC.Derived.RadonMeasureUp.

Constructive story: A $\mathsf{RadonMeasure}^{\uparrow}$ packet is generated from compact metric source, base finite measure, open-ball locator, compact inner approximant, finite cover outer approximant, distribution-facing value, componentwise transport, replay, provenance, and local NameCert rows.

Chapter 2264

A Concrete Naming Certificate for $\mathsf{OrthogonalPolynomial}^{\uparrow}$

Following Szego and Chihara on orthogonal polynomials, the $\mathsf{OrthogonalPolynomial}^{\uparrow}$ packet records the finite recurrence and moment ledgers by which a displayed polynomial family is named without passing through a host function space. It consumes the polynomial carrier of chapter 50, the real endpoint discipline of chapter 38, the finite matrix and linear-map rows of chapter 49 and chapter 48, and the coefficient-energy boundary of chapter 1940. Sturm-style sign and root ledgers from chapter 2079 may be cited by consumers that need interval localization, but the present packet names only the recurrence, moment, and normalization surface.

Definition 2264.1 (Orthogonal polynomial carrier)**.** *An* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (F, M, R, N, Z, E, H, C, P, A)$$

with the following displayed rows:

1. *F is the polynomial-family row, listing finitely many* $\mathsf{Poly}^{\uparrow}$ *rows $p_0, \ldots, p_k$ with degree, coefficient, and leading-coefficient witnesses.*
2. *M is the moment and inner-product ledger. It records the finite moment rows read by every displayed product $p_i p_j$ and the classifier row saying which inner products classify as zero or as the displayed norm value.*
3. *R is the three-term recurrence row, displaying the coefficients a_n, b_n, c_n and the finite polynomial equality ledger for $p_{n+1} = (a_n x + b_n)p_n - c_n p_{n-1}$ at each accepted index.*
4. *N is the normalization row, exposing norm-squared entries, nonzero leading rows, and the scaling witnesses used when a normalized family is requested.*
5. *Z is the finite Gram or Jacobi matrix row through which matrix and linear-map consumers read recurrence coefficients.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row for coefficient, moment, norm, and recurrence-coefficient entries.*
7. *H records componentwise* hsame *transport for polynomial rows, moment ledgers, recurrence rows, normalization rows, matrix rows, endpoint seals, continuation rows, provenance, and naming.*

8. *C records displayed* Cont *replay from coefficient inspection through moment evaluation, recurrence playback, normalization readback, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{OrthogonalPolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Parseval}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *A is the local* $\mathsf{NameCert}_{\mathsf{OrthogonalPolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host measure space, quotient polynomial ring, selected infinite limit, complete function space, or ambient spectral theorem.

Lean4 (definition) `BEDC.Derived.OrthogonalPolynomialUp.OrthogonalPolynomialUp`
Lean4 ✓ `BEDC.Derived.OrthogonalPolynomialUp.OrthogonalPolynomialTasteGate single carrier alignment`

Theorem 2264.2 (Orthogonal polynomial NameCert obligations)**.** *For every accepted* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *carrier* $O = (F, M, R, N, Z, E, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{OrthogonalPolynomial}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the polynomial-family, moment, recurrence, normalization, matrix, endpoint-seal, transport, replay, provenance, and local naming rows;*
2. *polynomial-source compatibility, requiring every recurrence and moment read to use only the* $\mathsf{Poly}^{\uparrow}$ *rows listed in F;*
3. *moment-ledger exactness, requiring each displayed inner product in M to name its finite moment rows and its resulting* $\mathsf{Real}^{\uparrow}$ *value;*
4. *orthogonality exactness, requiring the zero inner-product rows in M to be the only exported orthogonality witnesses;*
5. *normalization exactness, requiring N to display every norm, nonzero leading row, and scaling witness consumed by a normalized read;*
6. *recurrence exactness, requiring R to display each recurrence coefficient and the polynomial equality ledger that reconstructs the next family row from the previous two rows;*
7. *matrix and handoff visibility, requiring Z to expose the finite Jacobi or Gram row consumed by* $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Parseval}^{\uparrow}$, *or* $\mathsf{SturmSequence}^{\uparrow}$ *consumers without adding an external operator;*
8. *ledger preservation, requiring H, C, P, A to preserve exactly the displayed rows under componentwise transport, continuation replay, package provenance, and local naming.*

No orthogonal-polynomial obligation is stored outside these rows.

Proof. Project the carrier of Definition 2264.1. The family row F supplies the only polynomials that can be inspected, and the moment ledger M supplies the only inner-product values attached to products of those rows.

The orthogonality and normalization obligations are then finite ledger checks. A zero inner-product claim must point to the appropriate row of M, while a normalized read must pass through the norm and scaling witnesses displayed by N.

The recurrence obligation is likewise internal to R. Each accepted step lists its coefficients and the polynomial equality ledger reconstructing p_{n+1} from p_n and p_{n-1}, so no quotient polynomial equality or host function-space equality is available.

Finally, Z exposes the finite matrix row used by linear-algebra, Parseval, or Sturm consumers, and H, C, P, A transport, replay, source, and name the same rows. These projections exhaust the local NameCert surface. □

Theorem 2264.3 (Orthogonal polynomial three-term recurrence handoff)**.** *Every recurrence claim exported by an accepted* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *packet factors through the displayed route*

$$F, M, N \longmapsto R, Z, E \longmapsto H, C, P, A.$$

Thus a downstream spectral approximation or finite Jacobi-matrix consumer receives the polynomial family, moment ledger, normalization rows, recurrence coefficients, matrix readback, and endpoint seals, but receives no host function-space equality, no quotient polynomial object, and no selected infinite limiting family.

Proof. Read the recurrence row R in index order. Each entry names the preceding two polynomial rows from F, the displayed coefficients, and the equality ledger that constructs the next polynomial row.

The moment and normalization rows M, N provide the orthogonality and scaling side conditions needed to treat the recurrence as the orthogonal-polynomial recurrence rather than as an arbitrary polynomial recursion. The endpoint row E seals the real coefficient and moment reads before they leave the packet.

The matrix row Z repacks the same coefficients as a finite Gram or Jacobi row for $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ consumers, while $\mathsf{Parseval}^{\uparrow}$ and $\mathsf{SturmSequence}^{\uparrow}$ consumers can inspect only the displayed coefficient-energy or sign ledgers sourced from those same rows.

Since H, C, P, A preserve and name no data beyond F, M, R, N, Z, E, every exported handoff factors through the displayed route and cannot appeal to a host function space, quotient polynomial equality, or a selected limit. □

Falsifiable prediction: Every $\mathsf{OrthogonalPolynomial}^{\uparrow}$ read factors through one polynomial-family row, one moment and inner-product ledger, one three-term recurrence row, one normalization row, one finite Gram or Jacobi matrix row, one $\mathsf{Real}^{\uparrow}$ endpoint-seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, real, matrix, linearmap, parseval, sturmsequence: $\mathsf{OrthogonalPolynomial}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite polynomial family, moment ledger, recurrence coefficients, normalization rows, and Jacobi or Gram readback as one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OrthogonalPolynomial}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OrthogonalPolynomial}^{\uparrow}$ packet is generated from a finite PolynomialUp family, moment and inner-product ledger, three-term recurrence ledger, normalization rows, finite Gram or Jacobi matrix readback, RealUp endpoint seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrthogonalPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2264.1, Definition 2264.2, and Definition 2264.3 seed the finite orthogonal-polynomial certificate surface over displayed polynomial, moment, recurrence, normalization, matrix, and real endpoint rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrthogonalPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host measure spaces, infinite orthogonal systems, quotient polynomial equality, complete function-space equality, selected limits, spectral approximation theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, moment-ledger exactness, recurrence playback, normalization exactness, matrix handoff, endpoint sealing, and TasteGate field faithfulness for BEDC.Derived.OrthogonalPolynomialUp.

Chapter 2265

A Concrete Naming Certificate for $\mathsf{BanachDual}^{\uparrow}$

Following Banach and Bourbaki, the $\mathsf{BanachDual}^{\uparrow}$ packet records the continuous scalar-functional surface of a displayed Banach carrier as finite NameCert data. It reads its source through the Banach carrier of chapter 90, the normed-space rows of chapter 1950, the bounded-linear-operator surface of chapter 1644, the scalar boundary of chapter 38, and the functional-analysis interface of chapter 94. The packet is the scalar-functional neighbour of the Banach and operator certificates, not a host-level function space.

Definition 2265.1 (BanachDual carrier)**.** *For a displayed* $\mathsf{Banach}^{\uparrow}$ *carrier* C*, a* $\mathsf{BanachDual}^{\uparrow}(C)$ *carrier is a finite* BHist *packet*

$$\Phi = (C, L, K, H, R, P, N)$$

with the following displayed rows:

1. C *is the source* $\mathsf{Banach}^{\uparrow}$ *carrier, including its* $\mathsf{NormedSpace}^{\uparrow}$ *rows and completeness handoff.*
2. L *is a* $\mathsf{BoundedLinearOperator}^{\uparrow}(C, \mathsf{Real}^{\uparrow})$ *row whose target is the scalar singleton interface supplied by* $\mathsf{Real}^{\uparrow}$*.*
3. K *is the operator-norm bound endpoint read through the same* $\mathsf{Real}^{\uparrow}$ *scalar surface as* L*.*
4. H *records componentwise* hsame *transport for the source carrier, scalar-functional row, norm-bound row, replay row, provenance row, and local naming row.*
5. R *records displayed* Cont *replay from a scalar-functional request through* C, L, K, H, P, N*.*
6. P *is the* Pkg *provenance over* $\mathsf{BanachDual}^{\uparrow}$*,* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{NormedSpace}^{\uparrow}$*,* $\mathsf{BoundedLinearOperator}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FunctionalAnalysis}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
7. N *is the local* $\mathsf{NameCert}_{\mathsf{BanachDual}^{\uparrow}}$ *row.*

The classifier compares accepted dual rows only through the displayed source carrier, scalar-functional operator row, norm-bound endpoint, transport, replay, provenance, and local naming rows. It has no coordinate for host function extensionality, quotient dual spaces, Hahn–Banach separation data, selected representatives, or bridge checking.

Lean4 ✓ `BEDC.Derived.BanachDualUp.BanachDualTasteGate single carrier alignment`

Theorem 2265.2 (BanachDual NameCert obligations)**.** *For every accepted* $\mathsf{BanachDual}^{\uparrow}(C)$ *carrier* $\Phi = (C, L, K, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachDual}^{\uparrow}}$ *surface consists exactly of source*

Banach admission, scalar-functional bounded-operator admission, RealUp norm-bound exactness, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A scalar-functional consumer can inspect the source carrier C, the operator row L, the bound endpoint K, and the structural rows H, R, P, N, but cannot replace them by a host function, quotient dual object, chosen representative, or Hahn–Banach witness.*

Proof. Project the carrier rows. The only analytic coordinates are the source Banach row C, the scalar-functional bounded-operator row L, and the RealUp bound endpoint K.

The structural rows H, R, P, N preserve those coordinates through classifier transport, replay, provenance, and local naming. Because the target of L is the displayed scalar interface, every scalar-functional read is already routed through $\mathsf{BoundedLinearOperator}^{\uparrow}(C, \mathsf{Real}^{\uparrow})$.

No further row is present in the carrier. Hence host function extensionality, quotient dual spaces, Hahn–Banach strong forms, selected representatives, and bridge checks are outside the local NameCert surface. □

Theorem 2265.3 (BanachDual operator handoff)**.** *Every accepted* $\mathsf{BanachDual}^{\uparrow}(C)$ *scalar-functional read factors through the displayed route*

$$\mathsf{Banach}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

The handoff first reads the source Banach carrier C, then the scalar-valued bounded operator row L, then the RealUp norm-bound endpoint K, and only then the local transport, replay, provenance, and naming rows H, R, P, N.

Proof. Begin with an accepted scalar-functional read. By Definition 2265.1, the read is carried by $\Phi = (C, L, K, H, R, P, N)$.

The row L is already a $\mathsf{BoundedLinearOperator}^{\uparrow}(C, \mathsf{Real}^{\uparrow})$ row, so it exposes the Banach source, normed-space obligations, and scalar target before any local dual row is consumed. The endpoint bound K is a RealUp row attached to that same operator surface.

Replaying the request through H, R, P, N preserves the displayed route and adds no host function, quotient dual object, selected representative, or bridge check. □

Falsifiable prediction: Every accepted $\mathsf{BanachDual}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{Banach}^{\uparrow}$ source, a $\mathsf{BoundedLinearOperator}^{\uparrow}(C, \mathsf{Real}^{\uparrow})$ row, a $\mathsf{Real}^{\uparrow}$ norm-bound endpoint, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: banach, normedspace, boundedlinearoperator, real, and functionalanalysis: no listed sibling is bijective to $\mathsf{BanachDual}^{\uparrow}$ because this packet stores the scalar-valued bounded-operator face together with the source Banach carrier, norm-bound endpoint, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachDual}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachDual}^{\uparrow}$ packet is generated from a displayed $\mathsf{Banach}^{\uparrow}$ source carrier, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}(C, \mathsf{Real}^{\uparrow})$ row, a $\mathsf{Real}^{\uparrow}$ norm-bound endpoint, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachDual}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2265.1, Definition 2265.2, and Definition 2265.3 seed the scalar-functional carrier surface over the displayed Banach and bounded-operator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachDual}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient dual spaces, Hahn–Banach strong form, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-functional bounded-operator routing, norm-bound exactness, transport, replay, provenance, and local naming for BEDC.Derived.BanachDualUp.

Chapter 2266

A Concrete Naming Certificate for $\mathsf{IntervalBisectionTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{IntervalBisectionTree}^{\uparrow}$ packet records a finite tree of interval-refinement decisions for the Real and Completion route. It packages the branch ledger of chapter 361, the endpoint-cell surface of chapter 394, the finite tree discipline of **??**, the retained-window route of chapter 320, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The carrier is finite bisection-tree data for inspected interval prefixes, not an open-cover compactness principle, an infinite selected branch, a quotient real, trichotomy, countable choice, or ambient Real completeness.

Definition 2266.1 (IntervalBisectionTree carrier)**.** *An* $\mathsf{IntervalBisectionTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, D, B, Q, W, S, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedInterval}^{\uparrow}$ *row carrying the visible endpoint cells, located endpoint windows, rational containment rows, and Real-facing seal coordinates.*
2. *D is the finite* $\mathsf{DyadicBisection}^{\uparrow}$ *decision ledger, recording exactly the inspected left-or-right child choices and their dyadic scales.*
3. *B is the* $\mathsf{DyadicIntervalTree}^{\uparrow}$ *finite-tree row whose nodes are the interval cells visible at the requested depth.*
4. *Q records the containment and refinement rows that attach every child cell in B to its parent interval and to the retained branch prefix in D.*
5. *W is the retained* $\mathsf{NestedInterval}^{\uparrow}$ *window route exported from the finite tree after the inspected refinements have been applied.*
6. *S records the* $\mathsf{StreamName}^{\uparrow}$ *windows by which a downstream consumer requests finite endpoint or retained-window observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal route reached only after I, D, B, Q, W, S have been displayed.*

8. *H records componentwise* hsame *transport for the interval source, branch ledger, finite tree, containment rows, retained windows, stream windows, readback and seal rows, replay rows, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from a bisection-tree request through the same rows without adding an external branch selector.*
10. *P is the* Pkg *provenance over* $\mathsf{IntervalBisectionTree}^{\uparrow}$, $\mathsf{DyadicBisection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalTree}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalBisectionTree}^{\uparrow}}$ *row.*

The carrier predicate asserts only that these rows form one finite interval bisection-tree packet. It has no coordinate for open-cover compactness, selected infinite branches, countable choice, quotient real equality, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.IntervalBisectionTreeUp.IntervalBisectionTreeTasteGate single carrier alignment`

Theorem 2266.2 (IntervalBisectionTree NameCert obligations)**.** *For every accepted* $\mathsf{IntervalBisectionTree}^{\uparrow}$ *carrier* $T = (I, D, B, Q, W, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntervalBisectionTree}^{\uparrow}}$ *surface is exhausted by carrier admission, source located-interval exposure, dyadic branch-ledger exposure, finite tree-node exposure, parent-child containment rows, retained nested-window routing, stream-window requests, regular rational readback, Real-facing seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may inspect exactly those rows and no host open cover, selected branch, countable-choice witness, quotient-real representative, trichotomy row, or ambient completeness object.*

Proof. Project the carrier of Definition 2266.1. The analytic coordinates are the source interval row I, the branch ledger D, the finite tree row B, the containment rows Q, the retained nested route W, the stream windows S, and the readback and seal route R.

The local certificate obligations are then structural over those rows. Transport through H is componentwise, replay through C rereads only displayed entries, provenance through P names the listed sibling packets, and N names the resulting local certificate.

Since no other coordinate is present, the excluded compactness, branch-selection, choice, quotient, trichotomy, and completeness principles cannot be projected from an accepted packet. □

Theorem 2266.3 (IntervalBisectionTree Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{IntervalBisectionTree}^{\uparrow}$ *packet factors through the finite route*

$$I, D, B, Q \longmapsto W \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The consumer first reads the located interval and finite bisection-tree data, then the retained nested window, then the stream and regular-rational readback rows, and only then the Real-facing seal. The handoff exports no selected branch, quotient real, countable-choice row, trichotomy premise, open-cover compactness witness, or ambient Real-completeness principle.

Proof. Start with an accepted packet $T = (I, D, B, Q, W, S, R, H, C, P, N)$. The source and refinement stage is I, D, B, Q: it exposes the located interval row, finite branch decisions, finite tree nodes, and parent-child containment rows.

The retained-window stage is W. It is the only nested-interval route supplied by the packet, so a downstream read cannot skip the finite tree and ask for an independently selected branch.

The endpoint stage is S, R. Stream requests are finite, regular rational readback is displayed, and the Real-facing seal is downstream of those rows. Transport, replay, provenance, and local naming then preserve the same route and add no host-side principle. □

Falsifiable prediction: Every accepted $\mathsf{IntervalBisectionTree}^{\uparrow}$ consumer reads a located interval source, a finite dyadic branch ledger, finite interval-tree nodes, containment rows, a retained nested-window route, finite stream windows, regular rational readback, a Real-facing seal, transport, replay, provenance, and local naming.

Independence witness: dyadicbisection, locatedinterval, dyadicintervaltree, nestedinterval, streamname, regseqrat, real: no listed sibling is bijective to $\mathsf{IntervalBisectionTree}^{\uparrow}$ because this packet jointly carries source interval data, finite bisection-tree nodes, retained nested windows, readback and seal routing, and local NameCert structure.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalBisectionTree}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalBisectionTree}^{\uparrow}$ packet is generated from a located interval source, a finite dyadic branch ledger, finite interval-tree nodes, parent-child containment rows, a retained NestedInterval route, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalBisectionTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2266.1, Definition 2266.2, and Definition 2266.3 seed the finite bisection-tree route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalBisectionTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient reals, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bisection-tree exposure, retained-window routing, Real handoff, non-escape, and local NameCert obligations for BEDC.Derived.IntervalBisectionTreeUp.

Chapter 2267

A Concrete Naming Certificate for $\mathsf{MeagreSet}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{MeagreSet}^{\uparrow}$ packet records the finite BEDC surface for first-category evidence inside a displayed complete-metric Baire space. It sits next to the dense-open packet of chapter 1542, reads the prefix windows of chapter 334, and consumes the completeness rows of chapter 128 through the metric-completion interface of chapter 1506. The packet names displayed nowhere-dense covers by finite windows and regular-rational readbacks; it does not assert a host set-theoretic category theorem, a countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2267.1 (MeagreSet carrier)**.** *A* $\mathsf{MeagreSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, Q, K, W, R, E, T, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix packet supplying the finite observation windows in which basic balls, dense-open complements, and tested points are named.*
2. *Q is the displayed* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{MetricCompletion}^{\uparrow}$ *completeness route over the same metric source, including the limit-witness and completion-facing rows needed by Baire-category consumers.*
3. *K is the finite nowhere-dense cover ledger. Each listed row names a closed-frontier inspection, an empty-interior certificate for a displayed basic ball request, and the complementary dense-open row consumed downstream.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect the cover rows and the tested point name.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the radii, shrinkage requests, and comparison tolerances opened by W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the displayed regular-rational readback has been opened.*
7. *T is the* $\mathsf{BaireCategory}^{\uparrow}$ *downstream route sending the complementary dense-open rows of K to the finite dense-open intersection packet.*
8. *H stores componentwise* hsame *transport for B, Q, K, W, R, E, T and the structural rows.*

9. *C stores displayed* Cont *replay of the prefix windows, completeness route, nowhere-dense cover ledger, readback rows, real seals, and Baire-category route.*
10. *P is the* Pkg *provenance over* $\mathsf{MeagreSet}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MeagreSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary host subset, a selected point outside a countable union, a quotient completion, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MeagreSetUp.MeagreSetTasteGate single carrier alignment`

Theorem 2267.2 (MeagreSet NameCert obligations)**.** *For an accepted* $\mathsf{MeagreSet}^{\uparrow}$ *carrier* $M = (B, Q, K, W, R, E, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MeagreSet}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the Baire-space prefix, complete-metric and metric-completion route, nowhere-dense cover ledger, finite stream windows, regular-rational readbacks, real seal, Baire-category downstream route, transport, replay, provenance, and local naming row;*
2. *nowhere-dense ledger exposure, requiring each row of K to provide a displayed basic-ball request, an empty-interior certificate for the covered fragment, and the complementary dense-open row read by T;*
3. *window-readback compatibility, requiring W, R, E to open finite point and radius observations before any cover or dense-open decision is exported;*
4. *Baire-category consumer compatibility, requiring T to consume only the complementary dense-open rows already displayed by K over the same B, Q, W, R, E prefix;*
5. *ledger non-escape, excluding arbitrary host subsets, selected outside points, countable-choice schedules, quotient completion equality, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2267.1. The tuple lists exactly the prefix, completeness, cover, window, readback, seal, downstream, transport, replay, provenance, and local naming rows.

The nowhere-dense ledger is the row K. Each entry is opened only as a finite basic-ball inspection together with an empty-interior certificate and a complementary dense-open row; no host subset predicate is exported.

Window-readback compatibility follows from the order W, R, E. The finite stream window opens the relevant point and radius observations, the regular-rational row supplies the tolerance readback, and the real seal is reached only after those rows are displayed.

The downstream consumer route is T. It reads the complementary dense-open rows already named in K and hands them to the $\mathsf{BaireCategory}^{\uparrow}$ packet over the same prefix and completeness surface. The support rows H, C, P, N transport, replay, source, and name that same route.

No remaining carrier coordinate names a selected point outside a cover, a countable-choice schedule, quotient completion equality, open-cover compactness, ambient completeness, or completeness of $\mathsf{Real}^{\uparrow}$. The listed obligations exhaust the local NameCert surface. □

Theorem 2267.3 (MeagreSet BaireCategory consumer route)**.** *Let* $M = (B, Q, K, W, R, E, T, H, C, P, N)$ *be an accepted* $\mathsf{MeagreSet}^{\uparrow}$ *packet. Every* $\mathsf{BaireCategory}^{\uparrow}$ *consumer of M factors through the displayed route*

$$B, Q, K, W, R, E \longmapsto T.$$

The consumer receives the Baire-space prefix, the complete-metric and metric-completion route, the finite nowhere-dense cover ledger, stream windows, regular-rational readbacks, real seals, and the complementary dense-open handoff. It receives no arbitrary subset classifier, selected generic point, quotient completion, countable-choice selector, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with the carrier inventory. The prefix row B and completeness route Q determine the displayed complete-metric surface on which the cover rows can be interpreted.

Read the cover ledger K through the finite windows W. Each inspected cover row is tied to the same point and ball observations, and each radius or shrinkage request is read back through R before reaching the real seal E.

The route T then repacks the complementary dense-open rows for the $\mathsf{BaireCategory}^{\uparrow}$ consumer. This handoff is exactly the finite dense-open schedule required by the neighboring Baire-category packet; it is not a host intersection or a selection operation.

Finally use the structural rows H, C, P, N. They preserve the same route under componentwise transport, replay it as displayed continuation data, record its provenance, and name it locally. Since the carrier has no arbitrary subset classifier, selected generic point, quotient completion, choice schedule, open-cover compactness row, or ambient completeness row, no consumer can read one from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MeagreSet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MeagreSet}^{\uparrow}$ packet is generated from a BaireSpace prefix packet, CompleteMetric and MetricCompletion completeness route, finite nowhere-dense cover ledger, StreamName windows, RegSeqRat readbacks, Real seal, BaireCategory downstream route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MeagreSet}^{\uparrow}}$, seed)

Scope closed: Definition 2267.1, Definition 2267.2, and Definition 2267.3 seed the finite meagre-set certificate surface as a Baire-category neighbor of the complete-metric route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MeagreSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host set-theoretic category theorem, selected generic points, countable choice, quotient completion equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nowhere-dense ledger exposure, window-readback compatibility, Baire-category consumer compatibility, and ledger non-escape for the same finite packet.

Chapter 2268

A Concrete Naming Certificate for $\mathsf{DirichletUniformConvergence}^{\uparrow}$

Following Dirichlet and Bishop–Bridges constructive analysis, the $\mathsf{DirichletUniformConvergence}^{\uparrow}$ packet records uniform convergence by bounded partial sums, a monotone-null control row, and finite tail windows before any completion-facing consumer reads a real seal. It sits after the uniform Cauchy series route of chapter 1736, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the uniform-limit route of chapter 463, and the ratio-test comparison surface of chapter 2244. The chapter names the finite Dirichlet tail-control certificate; it is not a quotient series equality, a selected sum, countable choice, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 2268.1 (Dirichlet uniform convergence carrier)**.** *A* $\mathsf{DirichletUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(S, B, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the displayed series source row, exposing only the finite term and partial-sum windows consumed by the packet.*
2. *B is the bounded partial-sum row, recording the visible uniform bound for the source partial sums over the inspected domain.*
3. *M is the monotone-null control row, recording the coefficient schedule and its displayed tail decrease to the null row.*
4. *W is the finite tail-window schedule that pairs a requested tolerance with the bounded-sum windows and monotone-null controls inspected for that tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source windows, partial-sum windows, monotone-null controls, and tail schedule.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has displayed the scheduled tail comparison.*
7. *H records componentwise* hsame *transports for the source, bounded-row, monotone-null row, tail schedule, readback, seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *routes from any consumer request through S, B, M, W, R, E in that order.*

9. *P is the* Pkg *provenance over* $\mathsf{DirichletUniformConvergence}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformConvergence}^{\uparrow}}$ *row.*

The classifier admits only finite packets with these ten rows. No coordinate stores a quotient of series streams, a chosen sum, a countable-choice selector, trichotomy, or a completeness witness for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DirichletUniformConvergenceUp.DirichletUniformConvergenceTasteGate single carrier alignment`

Theorem 2268.2 (Dirichlet uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{DirichletUniformConvergence}^{\uparrow}$ *carrier* $(S, B, M, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirichletUniformConvergence}^{\uparrow}}$ *surface consists of five finite rows:*

1. *carrier admission for the displayed source, bounded partial-sum row, monotone-null control row, finite tail schedule, readback, seal, transports, continuations, provenance, and local naming;*
2. *bounded-source control, requiring every tail read to use the row B before a uniform estimate is exposed;*
3. *monotone-null control, requiring every coefficient read to use the row M and the schedule W before a tail threshold is exposed;*
4. *uniform tail handoff, requiring the comparison to pass through the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *route and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is consumed;*
5. *ledger refusal, stating that quotient series equality, selected sums, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are absent from the carrier.*

Proof. Project the carrier of Definition 2268.1. The source, bounded partial-sum row, monotone-null row, finite tail schedule, readback, seal, transport, continuation, provenance, and local naming are exactly the displayed packet coordinates.

The bounded-source row is read before any tail estimate because B is the only coordinate that records a uniform partial-sum bound. The monotone-null row is read before any coefficient estimate because M is the only coordinate that records the displayed null-control schedule.

The schedule W binds these two rows to one finite tail request. A downstream consumer therefore reaches the uniform Cauchy series surface of chapter 1736, the regular rational readback of chapter 301, and the real seal of chapter 38 only through the listed route.

The refusal row is an inventory check on the carrier. Since none of the ten coordinates is a quotient series equality, selected sum, countable-choice selector, trichotomy row, or ambient completeness witness, the local certificate exports none of those principles. □

Theorem 2268.3 (Dirichlet uniform tail stability)**.** *Let $(S, B, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{DirichletUniformConvergence}^{\uparrow}$ *carrier. The bounded partial-sum row B and monotone-null row M transported by the componentwise* hsame *ledger H and replayed by the displayed* Cont *routes C induce the same uniform tail ledger W before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with a consumer read of a uniform tail comparison. By Definition 2268.1, the only available bounded partial-sum data are in B, and the only available monotone-null data are in M.

Apply the obligation surface of Definition 2268.2. It requires both rows to be read through the finite schedule W before the regular rational readback R or the real seal E can be reached.

The transport row H is componentwise, so it carries the displayed source, bounded row, monotone-null row, and schedule without changing which tail windows are named. The continuation row C replays the same ordered route from source through B, M, W, R, E.

Thus transported and replayed consumers inspect the same bounded-partial-sum and monotone-null data at the same scheduled tail windows. The tail ledger is stable under the displayed hsame and Cont rows, and no quotient series equality, selected sum, countable-choice selector, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate can intervene. □

Falsifiable prediction: An accepted $\mathsf{DirichletUniformConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, or $\mathsf{UniformLimit}^{\uparrow}$ handoff without first reading the source row, bounded partial-sum row, monotone-null row, and finite tail schedule refutes the carrier surface.

Independence witness: uniform-cauchy-series, real, regseqrat, uniformlimit, real-series-ratio-test: none is bijective to $\mathsf{DirichletUniformConvergence}^{\uparrow}$ because this packet jointly carries a series source, bounded partial sums, monotone-null controls, one finite tail schedule, regular readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletUniformConvergence}^{\uparrow}$ packet is generated from a displayed series source, bounded partial-sum row, monotone-null control row, finite tail-window schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2268.1, Definition 2268.2, and Definition 2268.3 seed the finite Dirichlet uniform convergence tail-control surface over $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{UniformLimit}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove carrier admission, bounded-source control, monotone-null control, uniform tail handoff, and ledger refusal for the same finite packet.

Chapter 2269

A Concrete Naming Certificate for $\mathsf{RealRootIsolation}^{\uparrow}$

Following Sturm root isolation and Bishop–Bridges constructive real analysis, the $\mathsf{RealRootIsolation}^{\uparrow}$ packet records a complete polynomial-root isolation surface as finite BEDC NameCert data. It composes the sign-variation route of chapter 1534, the retained-window route of chapter 1407, the real-polynomial evaluation surface of chapter 1728, the dyadic interval carrier of chapter 306, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names an isolating certificate for one displayed real polynomial window; it does not select a host algebraic root, invoke algebraic closure, quotient real equality, trichotomy, countable choice, or ambient completeness.

Definition 2269.1 (Real root isolation carrier)**.** *A* $\mathsf{RealRootIsolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, E, S, B, D, T, L, H, C, Q, N)$$

with the following displayed rows:

1. *P is the polynomial source row. It records the finite coefficient ledger and the polynomial name read from the source* $\mathsf{Poly}^{\uparrow}$ *or* $\mathsf{RealPolynomial}^{\uparrow}$ *surface.*
2. *E is the real-polynomial evaluation row. It records the endpoint evaluation windows, sign rows, and requested precision route over the* $\mathsf{RealPolynomial}^{\uparrow}$ *packet.*
3. *S is the Sturm or sign-variation row. It records the displayed Sturm-chain sign table or the finite sign-variation ledger used to certify the accepted root count in each inspected cell.*
4. *B is the bisection refinement row. It stores the branch prefix, retained endpoint signs, and discarded-branch ledgers supplied by the* $\mathsf{BisectionRootIsolation}^{\uparrow}$ *surface.*
5. *D is the dyadic interval tree row. It records the ordered dyadic cells, parent-child containment rows, endpoint windows, and the final retained isolating interval.*
6. *T is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the retained endpoint names and by the downstream real-facing consumer of the isolating window.*
7. *L is the local classifier row. It admits two carriers together exactly when P, E, S, B, D, T expose the same polynomial source, evaluation windows, variation ledger, bisection branch, dyadic interval tree, and seal route.*
8. *H records componentwise* hsame *transport for polynomial source, evaluation, sign-variation, bisection, dyadic interval tree, real seal, classifier, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a root-isolation request through P, E, S, B, D, T, L, H in that order.*

10. *Q is the* Pkg *provenance over* $\mathsf{RealRootIsolation}^{\uparrow}$, $\mathsf{SturmRootIsolation}^{\uparrow}$, $\mathsf{BisectionRootIsolation}^{\uparrow}$, $\mathsf{RealPolynomial}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealRootIsolation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores a selected host root, a factorization theorem, a global algebraic closure object, quotient real equality, real trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealRootIsolationUp.RealRootIsolationTasteGate single carrier alignment`

Theorem 2269.2 (Real root isolation NameCert obligations)**.** *For every accepted* $\mathsf{RealRootIsolation}^{\uparrow}$ *carrier $R = (P, E, S, B, D, T, L, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealRootIsolation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the polynomial source, evaluation, sign-variation, bisection, dyadic interval tree, real seal, classifier, transport, replay, provenance, and local naming rows;*

2. *polynomial-source exactness, requiring every coefficient and polynomial-name read to pass through P;*

3. *evaluation exactness, requiring endpoint signs and precision windows to pass through the* $\mathsf{RealPolynomial}^{\uparrow}$ *evaluation row E;*

4. *Sturm or sign-variation exactness, requiring every accepted root-count read to be one of the finite rows displayed by S;*

5. *bisection refinement exactness, requiring the retained branch and discarded-branch evidence to pass through B;*

6. *dyadic interval handoff, requiring the retained interval to be exported only through the dyadic interval tree row D and the* $\mathsf{Real}^{\uparrow}$ *seal row T;*

7. *componentwise transport, replay, provenance, classifier stability, and local naming through L, H, C, Q, N.*

No obligation outside these rows is part of the packet.

Proof. Project the accepted carrier of Definition 2269.1. The tuple inventory already lists the whole finite packet, so carrier admission is exhausted by the rows $P, E, S, B, D, T, L, H, C, Q, N$.

The polynomial source row P fixes the coefficient ledger and polynomial name. Endpoint evaluation cannot be read directly from a host real polynomial; it passes through E, where the displayed precision windows and sign rows are recorded.

The root-count evidence is then confined to S. Whether the row is read as a Sturm-chain variation ledger or as an equivalent finite sign-variation table, every accepted count is a displayed row over the same polynomial and interval endpoints.

The bisection row B chooses the retained branch only after that sign-variation evidence is visible. The dyadic interval tree D records the parent-child containment and retained endpoint windows, and T seals the same retained interval for $\mathsf{Real}^{\uparrow}$ consumers.

The classifier L compares only these visible rows. Since H, C, Q, N merely transport, replay, source, and name the same route, no hidden host root, factorization theorem, algebraic closure object, quotient equality, trichotomy, choice, or completeness witness can be exported. □

Theorem 2269.3 (Real root isolation dyadic interval handoff)**.** *Every accepted* $\mathsf{RealRootIsolation}^{\uparrow}$ *read handed to a downstream interval or real-seal consumer factors through*

$$P, E, S, B \longmapsto D, T \longmapsto L, H, C, Q, N.$$

Thus the consumer receives a displayed polynomial source row, real-polynomial evaluation row, Sturm or sign-variation row, bisection refinement row, dyadic interval tree row, $\mathsf{Real}^{\uparrow}$ *seal row, classifier row, componentwise transport, replay, provenance, and local naming. It receives no selected host root, algebraic factorization, quotient real equality, trichotomy, countable choice, or ambient completeness principle.*

Proof. Start with a downstream read of the isolating interval. The polynomial source and endpoint evaluation are fixed by P and E, so the read cannot inspect a retained interval before the source polynomial and its endpoint signs are displayed.

Next read S and B. The sign-variation ledger supplies the accepted root-count evidence, and the bisection row records the retained branch together with the discarded-branch evidence.

The dyadic interval tree D is then the only export row for the retained cell. The real seal T packages that same endpoint window for $\mathsf{Real}^{\uparrow}$ consumers.

Finally L, H, C, Q, N compare, transport, replay, source, and locally name the route. Since none of those structural rows adds an analytic object, the handoff remains the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{RealRootIsolation}^{\uparrow}$ consumer that reaches a $\mathsf{DyadicInterval}^{\uparrow}$ retained cell or $\mathsf{Real}^{\uparrow}$ seal without first reading the polynomial source row, real-polynomial evaluation row, Sturm or sign-variation row, and bisection refinement row refutes the carrier surface.

Independence witness: sturmrootisolation, bisectionrootisolation, real-polynomial, dyadicinterval, real: none is bijective to $\mathsf{RealRootIsolation}^{\uparrow}$ because this packet jointly carries the polynomial source, evaluation row, sign-variation ledger, bisection refinement, dyadic interval tree, real seal, classifier, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealRootIsolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RealRootIsolation}^{\uparrow}$ packet is generated from a polynomial source row, real-polynomial evaluation row, Sturm or sign-variation row, bisection refinement row, dyadic interval tree row, $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealRootIsolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2269.1, Definition 2269.2, and Definition 2269.3 seed the finite real-root isolation packet over Sturm, bisection, dyadic interval, real-polynomial evaluation, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealRootIsolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host root selection, algebraic factorization, algebraic closure, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove source exactness, evaluation exactness, sign-variation exactness, bisection refinement, dyadic interval handoff, and structural stability for BEDC.Derived.RealRootIsolationUp.

Chapter 2270

A Concrete Naming Certificate for $\mathsf{SpectralStabilization}^{\uparrow}$

The $\mathsf{SpectralStabilization}^{\uparrow}$ packet realizes the killed bit-flip closure-chain spectral-radius and survival-potential ledger of subsection 9462.28.21 as finite BEDC NameCert data. It consumes the six-coordinate codon source window of chapter 2234, the stop-boundary transition surface of chapter 2245, the local inverse spectrum of chapter 2161, and the stop-suffix carrier of chapter 1374. The packet records a finite spectral stabilization surface for displayed killed one-bit dynamics; it does not identify the biological code table with a kernel primitive, choose a global stochastic model, or import an analytic Perron theorem outside the displayed finite ledger.

Definition 2270.1 (Spectral stabilization carrier)**.** *A* $\mathsf{SpectralStabilization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (W, M, I, K, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the finite Q_6 codon-window carrier and the six bit coordinates used by the killed bit-flip walk.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row exposing the local transition vocabulary for scan, elongate, terminate, readthrough, repair, and mutate states.*
3. *I is the* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *row exposing local face spectra, Hamming adjacency spectra, reassignment spectra, and stop-face reconstruction data.*
4. *K is the killed bit-flip matrix ledger. It records, for each displayed support in the closure chain, the finite transient matrix entries whose off-support moves are absorbed.*
5. *R is the spectral-radius row. It records the displayed one-step retention, Perron-root approximation row, and survival-potential ledger used by Definition 9462.1389.*
6. *L is the closure-chain ledger for $\mathcal{R}$, $\overline{\mathcal{R}}$, $\mathrm{Med}_1(\mathcal{R})$, and M, including the displayed vertex and internal-edge counts.*
7. *U is the stop-suffix and anchor handoff row, read through* $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ *and the finite stop-start core named by the source vision.*
8. *H records componentwise* hsame *transport for source windows, transition rows, inverse spectra, killed matrices, spectral-radius ledgers, closure-chain rows, stop-suffix handoffs, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a spectral-stabilization request through* W, M, I, K, R, L, U, H.
10. *P is the* Pkg *provenance over* $\mathsf{SpectralStabilization}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$, $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$, *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralStabilization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, M, I, K, R, L, U, H, C, P, N$. It has no coordinate for a hidden biological table, a selected global stochastic process, analytic spectral theory outside the finite matrix ledger, quotient codon equality, probabilistic inference, or a choice principle.

Lean4 (definition) `BEDC.Derived.SpectralStabilizationUp.SpectralStabilizationUp`

Theorem 2270.2 (Spectral stabilization NameCert obligations)**.** *For every accepted* $\mathsf{SpectralStabilization}^{\uparrow}$ *carrier $Z = (W, M, I, K, R, L, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpectralStabilization}^{\uparrow}}$ *surface consists exactly of source-window admission through W, transition admission through M, inverse-spectrum admission through I, killed-matrix entry exactness through K, spectral-radius and survival-potential ledger display through R, closure-chain support display through L, stop-suffix handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 2270.1. The source-window, transition, inverse-spectrum, killed-matrix, spectral-radius, closure-chain, stop-suffix, transport, replay, provenance, and local naming rows are exactly the displayed coordinates.

The killed matrix ledger K can be read only after the codon-window row W supplies the six bit coordinates and the transition row M supplies the local state vocabulary. The inverse-spectrum row I supplies the finite face and adjacency spectra used by the same matrix rows.

The spectral-radius row R is not an external analytic theorem. It is a displayed finite ledger containing the one-step retention, Perron-root approximation, and survival-potential data for the supports named by L.

The stop-suffix row U routes the anchor and stop-start core through the existing suffix packet. The structural rows H, C, P, N transport, replay, source, and name the same finite inventory, so no hidden biological table, global stochastic process, quotient codon equality, probabilistic inference row, or choice principle is exported. □

Theorem 2270.3 (Spectral stabilization window handoff)**.** *Every accepted* $\mathsf{SpectralStabilization}^{\uparrow}$ *spectral-radius or survival-potential read factors through*

$$W, M, I \longmapsto K, L, R, U \longmapsto H, C, P, N.$$

Thus a consumer receives the codon source window, transition vocabulary, inverse-spectrum row, killed bit-flip matrix entries, closure-chain support ledger, spectral-radius and survival-potential rows, stop-suffix handoff, transport, replay, provenance, and local naming. It receives no hidden biological table, selected global stochastic model, analytic spectral theorem outside the displayed matrix ledger, quotient codon equality, probabilistic inference row, or choice principle.

Proof. Start with a spectral-radius or survival-potential read. The codon-window row W fixes the finite six-coordinate source, while M fixes the transition vocabulary and I fixes the local spectrum rows.

The killed bit-flip matrix ledger K is then computed over the displayed supports in L. Its entries are finite rows: moves inside a support remain transient, and moves outside that support are absorbed.

The row R stores the one-step retention, spectral-radius approximation, and survival-potential data for the same support chain. The row U attaches the stop-suffix and anchor handoff without adding a global code table.

Finally H, C, P, N preserve and name the same packet. Since no additional coordinate exists, the handoff remains finite and cannot export the refused principles. □

Falsifiable prediction: Every accepted $\mathsf{SpectralStabilization}^{\uparrow}$ read of a spectral radius, one-step retention, or survival-potential row factors through one $\mathsf{CodonSixBitWindow}^{\uparrow}$ source window, one $\mathsf{CodonTransitionMatrix}^{\uparrow}$ transition vocabulary, one $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ inverse-spectrum row, one killed bit-flip matrix ledger, one closure-chain support ledger, one $\mathsf{StopCodonZeckendorfSuffix}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: codon-six-bit-window, codon-transition-matrix, codon-window-spectral-inverse, stopcodonzeckendorfsuffix: $\mathsf{SpectralStabilization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries killed matrix entries, closure-chain supports, spectral-radius rows, survival-potential rows, stop-suffix handoff, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.SpectralStabilizationUp` must provide explicit Nontrivial $\mathsf{SpectralStabilization}^{\uparrow}$ and FieldFaithful $\mathsf{SpectralStabilization}^{\uparrow}$ instances preserving $W, M, I, K, R, L, U, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralStabilization}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralStabilization}^{\uparrow}$ packet is generated from a codon source-window row, transition vocabulary row, inverse-spectrum row, killed bit-flip matrix ledger, closure-chain support ledger, spectral-radius and survival-potential rows, stop-suffix handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralStabilization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2270.1, Definition 2270.2, and Definition 2270.3 seed the finite spectral-stabilization packet over codon windows, transition matrices, inverse spectra, killed bit-flip matrices, closure-chain support rows, and stop-suffix handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralStabilization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological table identity, selected global stochastic modeling, analytic spectral theory outside finite matrix ledgers, quotient codon equality, probabilistic inference, choice, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove source-window admission, transition admission, inverse-spectrum admission, killed-matrix exactness, spectral-ledger display, closure-chain support display, stop-suffix handoff, Nontrivial SpectralStabilizationUp, and FieldFaithful SpectralStabilizationUp for BEDC.Derived.SpectralStabilizationUp.

Chapter 2271

A Concrete Naming Certificate for $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$

Following Stone, Engelking, and Willard on zero-dimensional and totally disconnected spaces, the $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$ packet records the finite BEDC surface by which a displayed topology admits no nontrivial connected component rows. It sits over the generated-open classifier of chapter 88, the compact finite-cover boundary of chapter 60, the separated-neighborhood surface of chapter 1925, and the clopen-ledger bridge of chapter 261. The packet names clopen separation and component-collapse rows without importing ultrafilters, host point equality, quotient spaces, selected components, open-cover compactness beyond the displayed compact row, or excluded middle.

Definition 2271.1 (TotallyDisconnectedSpace carrier)**.** *A* $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, K, Hd, x, y, U, V, L, Z, R, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Topology}^{\uparrow}$ *source row, including open predicates, finite neighborhood histories, and the generated-open classifier used by topology consumers.*
2. *K is the optional* $\mathsf{Compact}^{\uparrow}$ *support row read by compact Stone-style consumers; it is a displayed compactness interface, not an ambient open-cover principle.*
3. *Hd is the* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row exposing disjoint open-neighborhood data for distinct displayed point names.*
4. *x, y are point-name rows in the same topology source.*
5. *U, V are clopen separation rows: U contains x, V contains y, and each row carries both open and complement-open evidence in O.*
6. *L is the clopen ledger row connecting finite clopen observations with the Stone-duality clopen surface of chapter 261.*
7. *Z is the component-collapse row stating that every displayed connected-subpacket read through O has at most one point-name class under the local classifier.*
8. *R is the refusal row excluding ultrafilter points, quotient components, host equality, selected component representatives, and excluded-middle separation.*

9. *C records displayed* Cont *replay from a point-pair query through O, Hd, U, V, L, Z, R.*
10. *P is the* Pkg *provenance over* $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{StoneDuality}^{\uparrow}$, BHist, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyDisconnectedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a completed Stone space, ultrafilter selection, quotient-component equality, host subset equality, or a choice-selected clopen basis.

Lean4 ✓ `BEDC.Derived.TotallyDisconnectedSpaceUp.TotallyDisconnectedSpaceTasteGate single carrier alignment`

Theorem 2271.2 (TotallyDisconnectedSpace NameCert obligations)**.** *For every accepted* TotallyDisconnectedSpace *carrier*

$$T = (O, K, Hd, x, y, U, V, L, Z, R, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TotallyDisconnectedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, optional compact support, Hausdorff separation row, point names, clopen separation rows, clopen ledger, component-collapse row, refusal row, replay, provenance, and local naming row;*
2. *clopen separation, requiring every separated pair of displayed point names to be routed through clopen rows U, V inside O;*
3. *component collapse, requiring every connected-subpacket read through the topology source to factor through the singleton classifier row Z;*
4. *Stone-ledger compatibility, requiring the clopen ledger L to expose only the finite clopen rows consumed by the Stone-duality surface;*
5. *nonescape, requiring all topology, compact, Hausdorff, and Stone-duality consumers to read only $O, K, Hd, x, y, U, V, L, Z, R, C, P, N$.*

No totally disconnected space obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2271.1. The accepted tuple displays the topology source, compact support row, Hausdorff separation data, point names, clopen rows, clopen ledger, component-collapse row, refusal row, replay, provenance, and local naming data.

The clopen separation row is local to O. A separated pair first reads the $\mathsf{HausdorffSpace}^{\uparrow}$ row Hd for disjoint neighborhoods and then refines the displayed neighborhoods to the clopen rows U, V. The proof uses the rows as carrier fields; it does not select a global clopen basis.

Component collapse is exactly the role of Z. Any connected-subpacket consumer must pass through the same topology source and the same clopen ledger before it can compare point names, so the only exported comparison is the singleton classifier row.

The Stone-ledger and nonescape clauses are inventory checks on L, R, C, P, N. Since no ultrafilter point, quotient component, host equality row, selected representative, or excluded-middle branch occurs in the carrier, none can be exported by the local certificate. □

Theorem 2271.3 (TotallyDisconnectedSpace clopen separation route)**.** *Let $T = (O, K, Hd, x, y, U, V, L, Z, R, C, P$ be an accepted* $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$ *packet. Every Stone-duality or compact-Hausdorff consumer that asks for total disconnectedness factors first through the topology row O, then through*

the Hausdorff separation row Hd, then through the displayed clopen separation rows U, V, and finally through the clopen ledger L and component-collapse row Z. The route exports no ultrafilter point, quotient component, host point equality, selected clopen basis, or nonconstructive separation principle.

Proof. Start from the topology row O and the point names x, y. The Hausdorff row Hd supplies only displayed neighborhood separation, so a total-disconnectedness consumer must still use the clopen rows U, V before it can read a separation as a clopen separation.

The clopen ledger L records the finite rows consumed by Stone-duality-style arguments. It keeps the consumer inside the displayed topology classifier and does not build a point space or an ultrafilter representation.

Finally Z supplies the component-collapse comparison. Replay through C and naming through P, N preserve the same route, while the refusal row R excludes quotient components, host equality, selected clopen bases, and nonconstructive separation branches. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyDisconnectedSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyDisconnectedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2271.1, Definition 2271.2, and Definition 2271.3 seed the finite topology, compact, Hausdorff, clopen-ledger, and component-collapse surface for totally disconnected spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyDisconnectedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilter existence, quotient components, host point equality, selected clopen bases, open-cover compactness beyond displayed compact rows, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, clopen separation, component collapse, Stone-ledger compatibility, and nonescape for BEDC.Derived.TotallyDisconnectedSpaceUp.

Constructive story: A $\mathsf{TotallyDisconnectedSpace}^{\uparrow}$ packet is generated from a topology row, optional compact support, Hausdorff separated-neighborhood data, displayed point names, clopen separation rows, a clopen ledger, component-collapse evidence, replay, provenance, and a local NameCert row.

Chapter 2272

A Concrete Naming Certificate for MetaCICFrontierDependencyMatrix$^{\uparrow}$

The MetaCICFrontierDependencyMatrix$^{\uparrow}$ packet realizes the MetaCIC open-frontier ledger as finite matrix data. It places the four subject-reduction discharge rows beside the obstruction rows for dependent codomain, confluence, normalization, decidability, and audit-map consumers, so downstream chapters can cite a bounded dependency matrix rather than an informal checklist. The packet reads the open-problem ledger of section 9386.2, the subject-reduction discharge socket of chapter 719, the normalization boundary of chapter 892, and the route classifier of chapter 785.

Falsifiable prediction: Every accepted MetaCICFrontierDependencyMatrix$^{\uparrow}$ row factors through exactly one named positive support row or exactly one named obstruction row before any audit-map consumer reads it.

Independence witness: metacicopenproblemledger, subjectreductiondischargesocket, metacicnormalizationfrontier, subjectreductionrouteclassifier: the open ledger lists frontier questions, the discharge socket carries four positive subject-reduction rows, the normalization frontier names normalization obstructions, and the route classifier separates subject-reduction routes; none is bijective to the finite matrix packet that records positive rows, obstruction rows, and audit consumers together.

Definition 2272.1 (MetaCIC frontier dependency matrix carrier)**.** *A* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, A, L, P, Q, N, D, J, R, H, C, K, E)$$

with the following displayed rows:

1. B, A, L, P *are the four positive subject-reduction discharge rows: beta substitution, application-argument stability, lambda-domain subject reduction, and Pi-domain subject reduction.*
2. Q *is the dependent-codomain obstruction row separating application-argument stability from a closed subject-reduction theorem.*
3. N *is the MetaCIC normalization frontier row, including the reduction and normal-form boundaries read by normalization consumers.*
4. D *is the decidability frontier row for bounded checking and closed type-checking consumers.*
5. J *is the confluence and critical-pair obstruction row read by audit-map consumers.*

6. *R is the route-classifier row assigning each matrix row to positive support, obstruction, or audit consumer.*
7. *H records componentwise* hsame *transport for all positive, obstruction, classifier, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an audit-map request through a selected matrix row and its route-classifier entry.*
9. *K is the* Pkg *provenance over* $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *E is the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these finite rows. It has no coordinate for a closed subject-reduction theorem, global confluence, full normalization, total decidability, or an audit-map result that bypasses the route classifier.

Theorem 2272.2 (MetaCIC frontier dependency matrix obligations)**.** *For every accepted* MetaCICFrontierDepende *carrier $M = (B, A, L, P, Q, N, D, J, R, H, C, K, E)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the four positive subject-reduction rows, dependent-codomain obstruction row, normalization frontier, decidability frontier, confluence boundary, route classifier, transport, replay, provenance, and local naming row;*
2. *row polarity, requiring each matrix row to be classified by R as positive support, obstruction, or audit consumer;*
3. *support factorization, requiring every positive row to factor through one of B, A, L, P before subject-reduction consumers may use it;*
4. *obstruction factorization, requiring every negative frontier row to factor through exactly one of Q, N, D, J before audit-map consumers may cite it;*
5. *nonescape, excluding closed subject reduction, global confluence, full normalization, total decidability, and audit-map closure outside the displayed rows.*

No frontier dependency matrix obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2307.1. The tuple displays the positive subject-reduction rows, obstruction rows, classifier, transport, replay, provenance, and local naming row.

Row polarity is the job of R. Each matrix entry must be read with its route-classifier tag before any consumer treats it as support or obstruction.

Support factorization then follows by exhaustion of the positive rows. The only positive coordinates are B, A, L, P, so a subject-reduction consumer cannot obtain a fifth support row from the matrix.

Obstruction factorization is the matching exhaustion over Q, N, D, J. These rows name the dependent-codomain, normalization, decidability, and confluence boundaries, and the replay row C preserves their route tags for audit consumers.

The nonescape clause inspects K, E and the carrier fields. Since no closed subject-reduction theorem, global confluence theorem, full normalization theorem, total decidability theorem, or completed audit-map result appears among the coordinates, none is exported by the packet. □

Theorem 2272.3 (MetaCIC frontier audit-map route)**.** *Every audit-map consumer of an accepted* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *packet first reads the route classifier R, then either one positive support row among B, A, L, P or one obstruction row among Q, N, D, J, and only then replays the selected dependency through H, C, K, E. Thus the matrix is finite row data for frontier accounting, not a proof of closed subject reduction, confluence, normalization, decidability, or audit-map completion.*

Proof. An audit-map request begins at R, because the carrier gives no untagged dependency coordinate. The classifier chooses the polarity and the named row before the consumer can inspect the row content.

In the positive case, row exhaustion gives one of B, A, L, P. In the obstruction case, row exhaustion gives one of Q, N, D, J. The two cases are disjoint because the route classifier stores their tags before replay.

The structural rows H, C, K, E preserve and name the selected matrix entry. They do not add a closed theorem or a completed audit-map result, so the consumer remains inside finite frontier accounting. □

Theorem 2272.4 (MetaCIC frontier dependency matrix substitution support)**.** *Every beta-substitution or application-argument consumer of an accepted* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *packet first factors through one of the positive support rows B, A, L, P. Every dependent-codomain, normalization, decidability, or confluence consumer first factors through one of the obstruction rows Q, N, D, J. The packet therefore supplies a finite substitution-support route for subject-reduction discharge, but it exports no closed subject-reduction theorem, no global normalization theorem, no total decidability theorem, and no confluence theorem.*

Proof. Project the carrier of Definition 2307.1. The substitution-facing support rows are exactly B, A, L, P, with B carrying beta substitution and A carrying application-argument stability.

The remaining frontier consumers are separated by row polarity. Dependent-codomain reads pass through Q, normalization reads through N, decidability reads through D, and confluence or critical-pair reads through J.

Exhaust the route classifier R. Since every row is tagged before replay through H, C, K, E, a consumer cannot cross from a positive support row into a closed global theorem, and an obstruction row cannot be read as a completed MetaCIC result. □

Closure status for NameCert$_{\text{MetaCICFrontierDependencyMatrix}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICFrontierDependencyMatrix}^{\uparrow}}$, seed)

Scope closed: Definition 2307.1, Definition 2307.2, and Definition 2307.3 seed the finite MetaCIC frontier matrix over positive support rows, obstruction rows, route-classifier tags, and audit-map consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICFrontierDependencyMatrix}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close subject reduction, confluence, normalization, decidability, audit-map completion, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, row polarity, support factorization, obstruction factorization, audit-map route replay, and nonescape for BEDC.Derived.MetaCICFrontierDependencyMatrixUp.

Constructive story: A MetaCICFrontierDependencyMatrix$^{\uparrow}$ packet is generated from four positive subject-reduction rows, named obstruction rows, a route classifier, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2273

A Concrete Naming Certificate for $\mathsf{LiouvilleNumber}^{\uparrow}$

Beginning from Liouville's rational-approximation construction, the $\mathsf{LiouvilleNumber}^{\uparrow}$ packet records a nontrivial $\mathsf{Real}^{\uparrow}$ name by finite approximation rows rather than by an ambient transcendence theorem. It reads rational approximants through the rational-approximation sibling chapter 1668, stores tolerance budgets in the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, consumes finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, and seals the resulting regular real name through chapter 38. The packet is a finite NameCert route for Liouville-style approximation data, not a proof of full transcendence theory, decimal expansion uniqueness, countable choice over approximants, or completeness of all $\mathsf{Real}^{\uparrow}$ streams.

Definition 2273.1 (LiouvilleNumber carrier)**.** *A* $\mathsf{LiouvilleNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, D, S, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational-approximant row: for each accepted finite precision request it displays the inspected rational approximant and the denominator-growth witness used by the Liouville construction.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic budgets against which the rational errors are compared.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that lists exactly which approximant windows and tolerance windows are opened by the certificate.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, turning the scheduled rational approximants into a regular rational sequence before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *A is the Liouville approximation classifier, pairing each displayed denominator-growth row with the dyadic comparison that witnesses an error below the recorded budget.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, admitting the regular rational readback as the named real object after the finite approximation route has been checked.*
7. *H stores componentwise* hsame *transport for Q, D, S, R, A, E and the local naming rows.*
8. *C stores displayed* Cont *replay from a real-name request through the approximant row, tolerance ledger, finite window schedule, regular rational readback, approximation classifier, and real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{LiouvilleNumber}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LiouvilleNumber}^{\uparrow}}$ *row.*

The carrier asserts only that these displayed rows form one Liouville-style real-name certificate. It has no coordinate for decimal expansion uniqueness, unbounded choice of approximants outside the displayed windows, a blanket irrationality theorem for all reals, a transcendence theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LiouvilleNumberUp.LiouvilleNumberTasteGate single carrier alignment`

Theorem 2273.2 (LiouvilleNumber NameCert obligations)**.** *For every accepted* $\mathsf{LiouvilleNumber}^{\uparrow}$ *carrier* $L = (Q, D, S, R, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LiouvilleNumber}^{\uparrow}}$ *surface consists exactly of carrier admission, rational-approximant exposure through* Q*, dyadic tolerance comparison through* D*, finite window scheduling through* S*, regular rational readback through* R*, Liouville approximation classification through* A*,* $\mathsf{Real}^{\uparrow}$ *seal exposure through* E*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for decimal uniqueness, unbounded choice, general irrationality, transcendence, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 2273.1. The accepted packet exposes exactly the rational approximant row, dyadic ledger, finite window schedule, regular rational readback, approximation classifier, real seal, transport, replay, provenance, and local naming rows.

The approximation rows are finite because Q and S list only the inspected windows. Each accepted precision request is matched with the denominator-growth witness and error budget recorded in D and A.

The regular rational readback R converts those scheduled approximants into the row consumed by the real-name surface. The handoff to E therefore occurs only after the finite Liouville approximation route has been replayed.

Rows H, C, P, N transport, replay, source, and name the same packet. Since no other coordinate exists, the certificate cannot export decimal expansion uniqueness, an off-window choice function, a general irrationality theorem, a transcendence theorem, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 2273.3 (LiouvilleNumber finite approximation handoff)**.** *Every real-name consumer of an accepted* $\mathsf{LiouvilleNumber}^{\uparrow}$ *carrier factors through the finite route*

$$Q \longmapsto D \longmapsto S \longmapsto R \longmapsto A \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet supplies a Liouville-style $\mathsf{Real}^{\uparrow}$ *object from rational approximants, dyadic tolerance budgets, finite stream windows, and regular rational readback, without claiming decimal expansion uniqueness, unbounded choice, transcendence, or ambient real completeness.*

Proof. Start with an accepted carrier $L = (Q, D, S, R, A, E, H, C, P, N)$. A real-name request must read the rational approximant row Q and the dyadic budget row D before it can inspect the window schedule S.

The regular rational readback R is then forced by the displayed windows. The classifier A records the Liouville approximation inequality for the same inspected rows, so the real seal E receives only finite approximation evidence.

The transport, replay, provenance, and naming rows preserve that route. They add no coordinate for off-window choice, decimal uniqueness, transcendence, or a general completeness theorem, so every downstream consumer remains inside the finite NameCert packet. □

Theorem 2273.4 (LiouvilleNumber finite-window approximation induction)**.** *Let $L = (Q, D, S, R, A, E, H, C, P, N$ be an accepted* $\mathsf{LiouvilleNumber}^{\uparrow}$ *carrier, and let S' be a finite* $\mathsf{StreamName}^{\uparrow}$ *extension of the inspected window schedule S whose extra requests are witnessed by rows already routed through the rational-approximant row Q, the dyadic budget row D, the regular rational readback row R, the approximation classifier row A, and the* $\mathsf{Real}^{\uparrow}$ *seal row E. Then every real-name read over the extended finite window still factors through*

$$Q, D, S', R, A, E, H, C, P, N.$$

The extension preserves the same Liouville-style route by finite-window induction, and it still refuses off-window choice, decimal expansion uniqueness, transcendence, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The base window is exactly the finite schedule S, and Definition 2273.3 gives the route through $Q, D, R, A, E, H, C, P, N$ for every inspected request in that schedule.

For the induction step, add one displayed request to the window. The hypothesis on S' supplies the matching rational approximant in Q, dyadic tolerance budget in D, regular rational readback in R, approximation classifier in A, and real seal in E before the transport, replay, provenance, and naming rows are consulted.

Iterating the finite step over the added window entries gives the displayed route for all of S'. Since the induction only appends inspected rows and never adds a coordinate outside $Q, D, S', R, A, E, H, C, P, N$, the packet still has no off-window choice function, decimal uniqueness principle, transcendence theorem, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LiouvilleNumber}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LiouvilleNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2273.1, Definition 2273.2, Definition 2273.3, and Definition 2273.4 seed the finite Liouville-style real-name certificate over rational approximants, dyadic tolerance budgets, finite StreamName windows, RegSeqRat readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LiouvilleNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, decimal expansion uniqueness, countable choice over approximants, a general irrationality theorem, transcendence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, rational-approximant exposure, dyadic comparison exactness, finite window scheduling, regular rational readback, Real seal handoff, structural transport, replay, provenance, and nonescape for BEDC.Derived.LiouvilleNumberUp.

Constructive story: A $\mathsf{LiouvilleNumber}^{\uparrow}$ packet is generated from rational approximant rows, DyadicRatCore tolerance budgets, finite StreamName windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Chapter 2274

A Concrete Naming Certificate for CandidateSNConfluenceBoundary$^{\uparrow}$

The CandidateSNConfluenceBoundary$^{\uparrow}$ packet realizes the normalization and confluence frontier of chapter 9386 as one finite MetaCIC NameCert row. It sits beside the critical-path carrier of chapter 765, the normalisation frontier, the redex frontier, and the bounded normal-equality checker, and records exactly where candidate-mediated strong-normalization evidence can be consumed before confluence claims must stop. The packet names candidate evidence, bounded normal checks, critical-pair or redex-frontier handoff rows, and nonescape rows; it is not a closed subject-reduction theorem, arbitrary Church–Rosser theorem, or closed consistency theorem.

Falsifiable prediction: Any consumer of an accepted CandidateSNConfluenceBoundary$^{\uparrow}$ packet still cannot export closed subject reduction or arbitrary Church–Rosser unless it also supplies the displayed subject-reduction discharge rows, confluence-boundary rows, and candidate-interface admission rows of the carrier.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacicredexfrontier, boundednormalequalitychecker: the critical path records the global frontier, the normalization frontier records candidate-mediated strong-normalization boundaries, the redex frontier records local reduction cases, and the bounded checker records finite normal-equality comparison; none is bijective to the packet that stores all four rows with one confluence nonescape boundary.

Definition 2274.1 (CandidateSNConfluenceBoundary carrier)**.** *A* CandidateSNConfluenceBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, N, R, E, S, J, U, H, C, P, L)$$

with the following displayed rows:

1. *K is the MetaCIC critical-path row naming the subject-reduction, normalization, confluence, decidability, and consistency frontier being read.*
2. *N is the candidate strong-normalization row, admitting only terms already inside the candidate interface.*
3. *R is the redex-frontier row through which local reduction and critical-pair cases are inspected.*
4. *E is the bounded normal-equality checker row used when a finite normal comparison is requested.*
5. *S is the subject-reduction discharge row, recording which subject-reduction inputs are present and which remain outside the packet.*

6. *J is the confluence-boundary row, recording displayed join, common-reduct, or normal-fragment handoffs without promoting them to arbitrary Church–Rosser.*
7. *U is the nonescape row excluding closed subject reduction, arbitrary closed confluence, full closed consistency, and candidate-free strong normalization.*
8. *H records componentwise* hsame *transport for K, N, R, E, S, J, U and the local naming rows.*
9. *C records displayed* Cont *replay from a consumer request through candidate admission, redex-frontier inspection, bounded normal comparison, subject-reduction discharge, and confluence-boundary handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, MetaCICNormalizationFro $\mathsf{MetaCICRedexFrontier}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}}$ *row.*

The carrier compares accepted packets only by these rows. It has no coordinate for closed subject reduction, arbitrary Church–Rosser, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, or a completed MetaCIC audit map.

Theorem 2274.2 (CandidateSNConfluenceBoundary NameCert obligations)**.** *For every accepted* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *carrier $B = (K, N, R, E, S, J, U, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CandidateSNConfluence}}$ *surface consists exactly of carrier admission for the critical-path, candidate-SN, redex-frontier, bounded-checker, subject-reduction discharge, confluence-boundary, nonescape, transport, replay, provenance, and local naming rows; candidate-interface admission through N; redex-frontier inspection through R; bounded normal-equality comparison through E; explicit subject-reduction discharge through S; confluence-boundary handoff through J; and ledger refusal through U for closed subject reduction, arbitrary Church–Rosser, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, and completed audit-map closure.*

Proof. Carrier admission is projection from Definition 2274.1. The accepted packet exposes the critical-path row, candidate strong-normalization row, redex frontier, bounded checker, subject-reduction discharge row, confluence-boundary row, nonescape row, transport, replay, provenance, and local naming row.

Candidate-mediated strong normalization is read only through N. This keeps the strong-normalization evidence inside the candidate interface named by the packet.

Confluence consumption then passes through the displayed redex-frontier and bounded-checker rows R and E, followed by the discharge and boundary rows S and J. These rows distinguish local join or normal-fragment evidence from arbitrary closed Church–Rosser.

The nonescape row U and structural rows H, C, P, L preserve the same finite boundary. Since no coordinate provides closed subject reduction, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, or completed audit-map closure, none can be exported by the packet. □

Theorem 2274.3 (CandidateSNConfluenceBoundary checked obligation route)**.** *For every accepted* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *carrier $B = (K, N, R, E, S, J, U, H, C, P, L)$, the rows K, N, R, E, S, J, U, H are exactly the checked field inventory consumed by the Lean TasteGate alignment target. Consequently any SN-to-confluence read remains candidate-mediated and cannot become closed subject reduction, arbitrary Church–Rosser, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, or completed audit-map closure.*

Lean4 ✓ BEDC.Derived.CandidateSNConfluenceBoundaryUp.CandidateSNConfluenceBoundaryTasteGate single carrier alignment

Proof. Project the carrier of Definition 2274.1. The available rows are exactly the critical-path row K, candidate strong-normalization row N, redex-frontier row R, bounded normal-equality checker E, subject-reduction discharge row S, confluence-boundary row J, nonescape row U, transport row H, replay row C, provenance row P, and local naming row L.

The checked obligation route reads the same inventory. Candidate admission is mediated by N, confluence-facing inspection is mediated by R, E, S, J, and the nonescape boundary is mediated by U.

The structural rows H, C, P, L transport, replay, source, and name only that displayed packet. Since the carrier contains no row for closed subject reduction, arbitrary Church–Rosser, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, or completed audit-map closure, none can be exported by the checked route. □

Theorem 2274.4 (Candidate-mediated SN confluence boundary route)**.** *Every MetaCIC consumer of an accepted* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *packet factors through candidate admission, redex-frontier inspection, bounded normal-equality comparison, subject-reduction discharge, and confluence-boundary handoff before any conclusion is replayed:*

$$K \longmapsto N \longmapsto R, E \longmapsto S, J \longmapsto U \longmapsto H, C, P, L.$$

Consequently the packet is a finite boundary witness between candidate-mediated strong normalization and confluence claims, not evidence for closed subject reduction, arbitrary Church–Rosser, candidate-free strong normalization, or closed consistency.

Proof. Start with the carrier $B = (K, N, R, E, S, J, U, H, C, P, L)$. The critical-path row K locates the consumer at the MetaCIC frontier, but the first positive normalization input is still the candidate row N.

Any confluence-facing read must then inspect the local frontier rows R and E. These rows expose redex cases and bounded normal comparisons only at the finite window recorded by the packet.

The discharge row S and confluence-boundary row J determine which subject-reduction and join inputs are actually present. The nonescape row U records that all stronger endpoints remain outside the carrier.

Finally H, C, P, L transport, replay, source, and name the same finite route. They do not add closed subject reduction, arbitrary Church–Rosser, candidate-free strong normalization, or closed consistency evidence. □

Theorem 2274.5 (CandidateSNConfluenceBoundary Real-phase handoff)**.** *Any Real-phase consumer using an accepted* $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ *packet must pass through the candidate admission row N, redex frontier row R, bounded normal-equality row E, subject-reduction discharge row S, confluence-boundary row J, nonescape row U, and the L10 source faces before reaching a* $\mathsf{Real}^{\uparrow}$*-facing audit conclusion. The resulting handoff is the finite route*

$$K \longmapsto N \longmapsto R, E \longmapsto S, J \longmapsto U \longmapsto H, C, P, L$$

with the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *faces read only as downstream Real and Completion source surfaces, not as closed subject reduction, arbitrary Church–Rosser, candidate-free strong normalization, or completed MetaCIC audit-map closure.*

Proof. Project the carrier $B = (K, N, R, E, S, J, U, H, C, P, L)$. The only positive candidate-mediated normalization input is the admission row N, and the only confluence-facing local data are the redex frontier R and bounded normal-equality checker E.

Apply Definition 2274.4. It forces every MetaCIC consumer through N, then through R, E, and then through the subject-reduction discharge and confluence-boundary rows S, J before the nonescape row U is replayed.

A Real-phase audit can therefore attach the L10 source faces only after that boundary route is visible. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ reads are downstream source surfaces for a completion-facing audit, while H, C, P, L transport, replay, source, and name the same finite route.

Because U excludes closed subject reduction, arbitrary Church–Rosser, candidate-free strong normalization, and completed MetaCIC audit-map closure, the handoff does not promote the packet beyond the displayed boundary. □

The Lean follow-up for this packet is to provide `Nontrivial CandidateSNConfluenceBoundaryUp` and `FieldFaithful CandidateSNConfluenceBoundaryUp` instances in `TasteGate.lean`. Those instances must check that candidate admission, redex-frontier inspection, bounded equality checking, subject-reduction discharge, and confluence-boundary handoff are genuine carrier fields rather than aliases for a closed consistency endpoint.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2274.1, Definition 2274.2, Definition 2274.4, and Definition 2274.5 seed the finite MetaCIC packet separating candidate-mediated strong normalization from confluence, subject-reduction endpoints, and Real-phase audit handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, closed subject reduction, arbitrary Church–Rosser, unrestricted parallel diamond, candidate-free strong normalization, closed consistency, or completed MetaCIC audit-map closure.

Upgrade path: Obligation closure requires checked carrier admission, candidate-interface admission, redex-frontier inspection, bounded equality checking, subject-reduction discharge, confluence-boundary handoff, nonescape, structural transport, replay, and provenance for BEDC.Derived.CandidateSNConfluenceBoundaryUp.

Constructive story: A $\mathsf{CandidateSNConfluenceBoundary}^{\uparrow}$ packet is generated from MetaCIC critical-path rows, candidate strong-normalization evidence, redex-frontier rows, bounded normal-equality checker rows, subject-reduction discharge rows, confluence-boundary rows, nonescape data, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Chapter 2275

A Concrete Naming Certificate for $\mathsf{CoveringSpacePathLifting}^{\uparrow}$

Following Hatcher and Spanier on covering spaces, the $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ packet records the finite BEDC surface by which a displayed base path admits a lifted segment route once an initial point over the base endpoint is fixed. It sits over the homotopy path rows of chapter 156, the circle source of chapter 58, the indexed-open topology surface of chapter 88, the continuous-map carrier of chapter 130, and the metric observation boundary of chapter 57. The packet names path lifting as finite window and sheet data; it does not import a universal cover, monodromy action, quotient path space, chosen global sheet, or host uniqueness theorem.

Definition 2275.1 (CoveringSpacePathLifting carrier)**.** *A* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\gamma, \pi, e_0, U, S, \Sigma, E, H, C, P, N)$$

with the following displayed rows:

1. *γ is the base path row, read through* $\mathsf{Homotopy}^{\uparrow}$ *and the displayed* $\mathsf{S}^{1\uparrow}$ *or interval endpoint surface when the path is closed or endpoint-parametrized.*
2. *π is the covering projection row, a* $\mathsf{ContinuousMap}^{\uparrow}$ *packet from a displayed total-space topology to the base* $\mathsf{Topology}^{\uparrow}$ *row.*
3. *e_0 is the initial lift point row, together with the endpoint comparison showing that $\pi(e_0)$ is the initial endpoint of γ under the base classifier.*
4. *U is the evenly covered window ledger. Each base-path window carries a topology index, its lifted sheet window, and the local projection-readback row witnessing that the window is evenly covered.*
5. *S is the lifted segment ledger. It lists the finite lifted path segments, their endpoint gluing rows, and the* Cont *reads showing that their projection follows the corresponding base-path windows.*
6. *Σ is the sheet classifier. It compares lifted segments by the initial point, the evenly covered window row, and the segment gluing ledger, not by a host-level choice of sheet.*
7. *E is the endpoint seal. It records the final lifted endpoint after all segment windows have been glued and exposes only the projected endpoint comparison back to the base path.*

8. *H records componentwise* hsame *transport for the base path, projection, initial point, window ledger, lifted segment ledger, sheet classifier, endpoint seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a path-lifting request through $\gamma, \pi, e_0, U, S, \Sigma, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CoveringSpacePathLifting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a universal covering space, a chosen global sheet, a completed path quotient, a monodromy action, a fundamental-group representation, or a host uniqueness principle.

Lean4 ✓ `BEDC.Derived.CoveringSpacePathLiftingUp.TasteGate.CoveringSpacePathLiftingTasteGate`
`single carrier alignment`

Theorem 2275.2 (CoveringSpacePathLifting NameCert obligations)**.** *For every accepted* CoveringSpacePathLifting *carrier*

$$L = (\gamma, \pi, e_0, U, S, \Sigma, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CoveringSpacePathLifting}^{\uparrow}}$ *surface consists exactly of carrier admission, base-path admission through* $\mathsf{Homotopy}^{\uparrow}$, *covering-projection admission through* $\mathsf{ContinuousMap}^{\uparrow}$ *over* $\mathsf{Topology}^{\uparrow}$, *initial endpoint exactness, evenly covered window exactness, lifted segment gluing, sheet-classifier stability, endpoint seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every path-lifting consumer can inspect $\gamma, \pi, e_0, U, S, \Sigma, E, H, C, P, N$, but cannot replace them by a universal cover, host path quotient, selected global sheet, monodromy action, or fundamental-group representation.*

Proof. Carrier admission is projection from Definition 2275.1. The accepted tuple displays the base path, covering projection, initial lift point, evenly covered windows, lifted segment ledger, sheet classifier, endpoint seal, transport, replay, provenance, and local naming rows.

The base and projection obligations are inherited by reading γ through $\mathsf{Homotopy}^{\uparrow}$ and π through $\mathsf{ContinuousMap}^{\uparrow}$ over the displayed $\mathsf{Topology}^{\uparrow}$ rows. The metric observation boundary may be used only as the finite $\mathsf{Met}^{\uparrow}$ readback already named in the provenance row.

Initial endpoint exactness, window exactness, and segment gluing are the roles of e_0, U, S. Each lifted segment is attached to a visible evenly covered window, and the gluing row records the next endpoint before the following window is consumed.

The sheet classifier and endpoint seal are then inventory checks on Σ and E. Since H, C, P, N only transport, replay, source, and name the same finite rows, the local certificate has no remaining place to store a universal cover, global sheet selector, quotient path, monodromy action, or fundamental-group representation. □

Theorem 2275.3 (CoveringSpacePathLifting unique lift route)**.** *Let $L = (\gamma, \pi, e_0, U, S, \Sigma, E, H, C, P, N)$ and $L' = (\gamma, \pi, e_0, U, S', \Sigma', E', H', C', P', N')$ be accepted* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ *packets with the same displayed base path, covering projection, initial lift point, and evenly covered window ledger. If the lifted segment ledgers are both admitted by the same sheet classifier route over U, then every endpoint and intermediate segment read of L and L' factors through the same finite route*

$$\gamma,\ \pi,\ e_0,\ U,\ \Sigma,\ E,\ H,\ C,\ P,\ N.$$

Thus uniqueness of the lift is recorded as sheet-classifier determinacy over the displayed window ledger, not as host path equality or a selected global sheet.

Proof. Start with the common rows γ, π, e_0, U. They fix the base path, the projection surface, the initial lifted endpoint, and the sequence of evenly covered windows visible to both packets.

Read the first lifted segment through the sheet classifier. Because both packets use the same initial lift point and the same first evenly covered window, Σ admits only the displayed segment row compatible with the local projection readback.

Proceed along the finite lifted segment ledger. At each next window, the previous endpoint is already sealed by the gluing row, so the same sheet-classifier route determines the next segment read. This is a finite induction over the listed windows of U, not an appeal to a completed path space.

The final endpoint rows E and E' are therefore read through the same sequence of segment and sheet rows. Replaying and transporting by H, C, P, N preserves that route and exports no global sheet selector, quotient path equality, monodromy action, or fundamental-group representation. □

Theorem 2275.4 (CoveringSpacePathLifting monodromy boundary)**.** *Let $L = (\gamma, \pi, e_0, U, S, \Sigma, E, H, C, P, N)$ be an accepted* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ *packet. Any attempted loop-consumer read from L must factor through the displayed base path γ, covering projection π, initial lift point e_0, evenly covered window ledger U, lifted segment ledger S, sheet classifier Σ, endpoint seal E, transport H, replay C, provenance P, and local NameCert row N. Consequently the packet can expose endpoint and sheet-classifier reads along the displayed lifting route, but it cannot export a monodromy action, a fundamental-group representation, a host path quotient, or a universal-cover carrier.*

Proof. Project the accepted carrier using Definition 2275.1. The only rows available to a consumer are the displayed base path, projection, initial lift point, window ledger, lifted segments, sheet classifier, endpoint seal, transport, replay, provenance, and local naming.

Apply Definition 2275.2. The NameCert surface admits inspection of those rows and refuses replacement by a universal cover, host path quotient, selected global sheet, monodromy action, or fundamental-group representation.

Apply Definition 2275.3 to the lifted segment route whenever the loop consumer asks for endpoint or intermediate segment reads along the displayed window ledger. The theorem identifies the finite sheet-classifier route through U, Σ, E, H, C, P, N and does not add a completed path space.

Thus every accepted read remains a path-lifting read through the finite carrier. Exporting monodromy or a representation would require an additional coordinate that stores loop composition on path classes, and no such coordinate is present in L. □

Closure status for $\mathsf{NameCert}_{\mathsf{CoveringSpacePathLifting}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoveringSpacePathLifting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2275.1, Definition 2275.2, Definition 2275.3, and Definition 2275.4 seed the finite base-path, covering-projection, evenly covered window, lifted segment, sheet-classifier, endpoint-seal, and loop-consumer boundary surface for path lifting.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoveringSpacePathLifting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close universal covers, monodromy actions, fundamental-group representations, host path quotients, selected global sheets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, lifted segment gluing, sheet-classifier determinacy, endpoint seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CoveringSpacePathLiftingUp.

Constructive story: A $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ packet is generated from a base path, a covering projection, an initial lifted point, evenly covered base windows, lifted segment rows, a sheet classifier, an endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2276

A Concrete Naming Certificate for $\mathsf{CoveringHomotopyLifting}^{\uparrow}$

Hatcher and Spanier treat the covering homotopy lifting property as the homotopy-level continuation of path lifting over evenly covered neighborhoods. The $\mathsf{CoveringHomotopyLifting}^{\uparrow}$ packet records that classical object as finite BEDC sheet-window data: it consumes the path-lifting route of chapter 2275, the homotopy rows of chapter 156, the circle and interval endpoint surfaces of chapter 58, the topology and continuous-map carriers of chapter 88 and chapter 130, and the finite metric observation boundary of chapter 57. It names homotopy lifting as a rectangle of base-time windows and sheet seals; it does not import a universal cover, quotient path space, monodromy action, fundamental-group representation, or global sheet choice.

Definition 2276.1 (CoveringHomotopyLifting carrier)**.** *A* $\mathsf{CoveringHomotopyLifting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, \pi, \tilde{f}_0, I, U, W, \Sigma, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the base homotopy row, read through* $\mathsf{Homotopy}^{\uparrow}$ *from a displayed source surface and time interval into the base* $\mathsf{Topology}^{\uparrow}$ *row.*
2. *π is the covering projection row, a* $\mathsf{ContinuousMap}^{\uparrow}$ *packet from a displayed total-space topology to the base topology.*
3. *$\tilde{f}_0$ is the initial lifted slice row, together with the endpoint comparison showing that $\pi \circ \tilde{f}_0$ is the initial time-slice of F.*
4. *I is the source-time cell ledger. It lists the finite domain cells and adjacent time strips on which the homotopy request is inspected.*
5. *U is the evenly covered base-window ledger. Each cell-time window has a named base open, a lifted sheet window, and a local projection-readback row.*
6. *W is the lifted homotopy window ledger. It records the lifted cell strips, their face-gluing rows, and the displayed* Cont *reads showing that projection follows F on each window.*
7. *Σ is the sheet-window classifier. It compares lifted strips by the initial slice, the evenly covered window row, and face gluing, not by host-level equality of lifted maps.*
8. *E is the terminal slice seal. It records the final lifted time-slice and exposes only the projected comparison back to the base homotopy.*

9. *H records componentwise* hsame *transport for the base homotopy, projection, initial slice, source-time cells, evenly covered windows, lifted windows, sheet classifier, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a homotopy-lifting request through $F, \pi, \tilde{f}_0, I, U, W, \Sigma, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CoveringHomotopyLifting}^{\uparrow}$, $\mathsf{CoveringSpacePathLifting}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CoveringHomotopyLifting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a universal covering space, quotient path space, selected global sheet, monodromy action, fundamental-group representation, host homotopy equality, or ambient uniqueness theorem.

Lean4 ✓ `BEDC.Derived.CoveringHomotopyLiftingUp.CoveringHomotopyLiftingTasteGate single carrier alignment`

Theorem 2276.2 (CoveringHomotopyLifting NameCert obligations)**.** *For every accepted* CoveringHomotopyLifting *carrier*

$$Q = (F, \pi, \tilde{f}_0, I, U, W, \Sigma, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CoveringHomotopyLifting}^{\uparrow}}$ *surface consists exactly of carrier admission, base-homotopy admission through* $\mathsf{Homotopy}^{\uparrow}$, *covering-projection admission through* $\mathsf{ContinuousMap}^{\uparrow}$ *over* $\mathsf{Topology}^{\uparrow}$, *initial slice exactness, source-time cell coverage, evenly covered window exactness, lifted window gluing, sheet-window classifier stability, terminal slice readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every homotopy-lifting consumer can inspect $F, \pi, \tilde{f}_0, I, U, W, \Sigma, E, H, C, P, N$, but cannot replace them by a universal cover, quotient path space, selected global sheet, monodromy action, or fundamental-group representation.*

Lean4 ✓ `BEDC.Derived.CoveringHomotopyLiftingUp.CoveringHomotopyLiftingNameCert obligations`

Proof. Carrier admission is projection from Definition 2276.1. The accepted tuple displays the base homotopy, covering projection, initial lifted slice, source-time cell ledger, evenly covered windows, lifted window ledger, sheet classifier, terminal seal, transport, replay, provenance, and local naming rows.

The base and projection obligations are inherited by reading F through $\mathsf{Homotopy}^{\uparrow}$ and π through $\mathsf{ContinuousMap}^{\uparrow}$ over the displayed $\mathsf{Topology}^{\uparrow}$ rows. The prior path-lifting certificate of chapter 2275 supplies the one-dimensional boundary route on each cell edge; the present carrier stores only the finite cell-time extension data needed to glue those routes across adjacent strips.

Initial slice exactness, source-time coverage, evenly covered window exactness, and lifted window gluing are the roles of $\tilde{f}_0, I, U, W$. Each lifted strip is attached to a visible evenly covered window, and each face-gluing row records the next boundary before the adjacent time strip is consumed.

The sheet-window classifier and terminal seal are then inventory checks on Σ and E. Since H, C, P, N only transport, replay, source, and name the same finite rows, the local certificate has no remaining place to store a universal cover, quotient path space, global sheet selector, monodromy action, or fundamental-group representation. □

Theorem 2276.3 (CoveringHomotopyLifting sheet-window determinacy)**.** *Let*

$$Q = (F, \pi, \tilde{f}_0, I, U, W, \Sigma, E, H, C, P, N)$$

and

$$Q' = (F, \pi, \tilde{f}_0, I, U, W', \Sigma', E', H', C', P', N')$$

be accepted $\mathsf{CoveringHomotopyLifting}^{\uparrow}$ *packets with the same displayed base homotopy, covering projection, initial lifted slice, source-time cell ledger, and evenly covered base-window ledger. If both lifted window ledgers are admitted by the same sheet-window classifier route over* I *and* U*, then every terminal slice and intermediate strip read of* Q *and* Q' *factors through the same finite route*

$$F,\ \pi,\ \tilde{f}_0,\ I,\ U,\ \Sigma,\ E,\ H,\ C,\ P,\ N.$$

Thus homotopy lifting determinacy is recorded as sheet-window determinacy over displayed cell-time windows, not as host equality of lifted homotopies, quotient path-space equality, or a selected global sheet.

Proof. Start with the common rows $F, \pi, \tilde{f}_0, I, U$. They fix the base homotopy, the projection surface, the initial lifted slice, the source-time cell schedule, and the sequence of evenly covered base windows visible to both packets.

Read the first lifted cell strip through the sheet-window classifier. Because both packets use the same initial slice and the same first evenly covered window, Σ admits only the displayed strip row compatible with the local projection readback.

Proceed over the finite source-time cell ledger. At each adjacent face, the previous strip boundary is already sealed by the gluing row, so the same sheet-window classifier route determines the next lifted strip read. This is a finite induction over the listed cells of I and windows of U, not an appeal to a completed path space.

The final slice rows E and E' are therefore read through the same sequence of window and sheet rows. Replaying and transporting by H, C, P, N preserves that route and exports no global sheet selector, quotient path-space equality, monodromy action, or fundamental-group representation. □

Closure status for $\mathsf{NameCert}_{\mathsf{CoveringHomotopyLifting}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoveringHomotopyLifting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2276.1, Definition 2276.2, and Definition 2276.3 seed the finite base-homotopy, covering-projection, source-time cell, evenly covered window, lifted strip, sheet-classifier, and terminal-slice surface for homotopy lifting.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoveringHomotopyLifting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close universal covers, quotient path spaces, monodromy actions, fundamental-group representations, host homotopy equality, selected global sheets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-time coverage, window exactness, lifted strip gluing, sheet-window determinacy, terminal slice readback, transport, replay, provenance, and local naming for BEDC.Derived.CoveringHomotopyLiftingUp.

Constructive story: A $\mathsf{CoveringHomotopyLifting}^{\uparrow}$ packet is generated from a base homotopy, a covering projection, an initial lifted slice, source-time cells, evenly covered base windows, lifted strip rows, a sheet-window classifier, a terminal slice seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2277

A Concrete Naming Certificate for $\mathsf{NestedIntervalTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{NestedIntervalTheorem}^{\uparrow}$ packet records the constructive nested closed interval theorem as finite BEDC route data. It consumes the retained-window surface of chapter 320, the dyadic endpoint and tolerance ledger of chapter 291, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the interval-existence handoff without asserting quotient reals, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2277.1 (Nested interval theorem carrier)**.** *A* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{NestedInterval}^{\uparrow}$ *row, including the inspected closed rational or dyadic interval windows, endpoint-order rows, width rows, and displayed containment witnesses.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger that records the endpoint comparisons and the diameter-to-tolerance rows used by the retained windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule requested from the lower and upper endpoint rows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite interval windows and dyadic tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval, dyadic, stream, regular readback, Real seal, replay, provenance, and naming rows.*
7. *C records the displayed* Cont *replay from nested interval data through dyadic tolerance control, stream windows, regular readback, and Real seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalTheorem}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite host intersection object, a selected quotient representative, countable choice, trichotomy, a Cauchy-completion quotient, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.NestedIntervalTheoremUp`

Theorem 2277.2 (Nested interval theorem NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *carrier* $K = (I, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed nine-row packet,* $\mathsf{NestedInterval}^{\uparrow}$ *source exposure through* I*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance exactness through* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E*, and the structural transport, replay, provenance, and naming rows* H, C, P, N*.*

No obligation may be discharged by a host interval intersection, selected point, countable choice, trichotomy, quotient real, or ambient completeness principle.

Proof. Project the carrier of Definition 2308.1. The rows $I, D, S, R, E, H, C, P, N$ are the only coordinates visible to the local classifier.

The interval source is I. It exposes only the finite nested-window surface already available from $\mathsf{NestedInterval}^{\uparrow}$: endpoint order, width, containment, and retained-window rows.

The completion-facing route is finite. The ledger D reads the dyadic endpoint and tolerance comparisons, S lists the requested stream windows, R repacks those windows as a regular rational readback, and E is downstream of those rows.

The structural rows do not add mathematical principles. The transports H, continuations C, provenance P, and local certificate N preserve and name the same packet, so the excluded host intersection, choice, quotient, trichotomy, and completeness coordinates remain absent. □

Lean4 ✓ `BEDC.Derived.NestedIntervalTheoremUp.NestedIntervalTheoremTasteGate single carrier alignment`

Theorem 2277.3 (Nested interval theorem Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads the nested interval row I*, the dyadic endpoint and tolerance ledger* D*, the stream-window schedule* S*, the regular-rational row* R*, and only then the Real seal row* E *with its transports, continuations, provenance, and local naming.*

Proof. Start with the Real-facing read. Since E is the only Real seal coordinate in the carrier, every such read must pass through the rows that make E visible.

The source row is I. It supplies the finite nested closed interval windows and containment witnesses, while D supplies the dyadic endpoint comparisons and diameter-to-tolerance rows read from the same windows.

The name construction then passes through S and R. The finite stream schedule selects the requested endpoint windows, and the regular rational row records the corresponding Cauchy readback.

Finally H, C, P, N transport, replay, source, and name the same route. Because the carrier contains no coordinate for a host intersection object, selected point, quotient real, countable choice, trichotomy, or ambient completeness, the displayed handoff is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NestedIntervalTheorem}^{\uparrow}$ packet is generated from a NestedInterval row, a DyadicRatCore endpoint and tolerance ledger, StreamName finite windows, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 2308.1, Definition 2308.2, and Definition 2308.3 seed the finite constructive nested closed interval route from displayed nested windows to RegSeqRat readback and Real seal handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.NestedIntervalTheoremUp`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient reals, selected host intersections, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window exposure, dyadic endpoint and tolerance exactness, StreamName window exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalTheoremUp.

The routed HellyTheorem$^{\uparrow}$ chapter consumes chapter 208 and exports its finite subfamily ledger to chapter 2347 before the child files spell out the dependency route.

Chapter 2278

A Concrete Naming Certificate for $\mathsf{HellyTheorem}^{\uparrow}$

Following Helly and the finite-dimensional convexity tradition, the $\mathsf{HellyTheorem}^{\uparrow}$ packet records finite convex-intersection data as a BEDC naming surface. It consumes the convex-family rows of chapter 208, the finite witness rows of chapter 209, the ambient affine and vector rows of chapter 206 and chapter 47, and the compact finite-read discipline of chapter 129. The chapter names only finite subfamily ledgers, displayed intersection witnesses, transport, continuation replay, provenance, and local naming; it does not assert a host separation theorem, choice-selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

The $\mathsf{HellyTheorem}^{\uparrow}$ packet also consumes $\mathsf{ConvexSet}^{\uparrow}$ carrier rows and the finite-subfamily rows of chapter 1792 before any intersection conclusion is exposed.

The finite convex-geometry route is visible before the Helly certificate is exported: $\mathsf{HellyTheorem}^{\uparrow}$ reads each family member through the $\mathsf{ConvexSet}^{\uparrow}$ carrier of chapter 208, then uses the Caratheodory-style finite convex-hull rows of chapter 2347 as the support ledger that keeps the intersection test finite.

The finite subfamily ledger exported by $\mathsf{HellyTheorem}^{\uparrow}$ is consumed by chapter 2347 as the boundary that keeps convex-hull support rows finite before Caratheodory support reduction is read. That ledger is not available from the ambient affine row alone: it is read only after the $\mathsf{ConvexSet}^{\uparrow}$ membership rows of chapter 208 and the $\mathsf{Polytope}^{\uparrow}$ finite witness rows of chapter 209 expose the subfamily boundary that the Caratheodory packet consumes.

The same finite subfamily ledger is a reusable dependency for convex-hull and Caratheodory consumers: chapter 2347 may read the $\mathsf{HellyTheorem}^{\uparrow}$ row only after the $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$ surfaces named above have fixed the displayed finite family boundary.

The finite closed-family indexing boundary is supplied by $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ through chapter 1792; $\mathsf{HellyTheorem}^{\uparrow}$ then specializes that boundary by adding affine ambient rows, vector displacement rows, convex-family membership, and polytope witnesses for the Helly subfamily ledger.

The concrete dependency route is horizontal rather than ambient: each finite family member is first read as a $\mathsf{ConvexSet}^{\uparrow}$ row through chapter 208, then the finite subfamily ledger S is checked against the finite-intersection carrier rows of chapter 1792, and only then may the full-family row I be exposed to a Helly consumer.

2278.1 Finite Convex Intersection Carrier

Definition 2278.1 (HellyTheorem carrier)**.** *A* $\mathsf{HellyTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, V, F, W, S, I, T, C, P, N)$$

with the following displayed rows:

1. *A is the finite-dimensional* $\mathsf{AffineSpace}^{\uparrow}$ *ambient row.*
2. *V is the associated* $\mathsf{VecSpace}^{\uparrow}$ *displacement row used by the affine coordinates.*
3. *F is a finite family of* $\mathsf{ConvexSet}^{\uparrow}$ *rows in the ambient affine space.*
4. *W is the* $\mathsf{Polytope}^{\uparrow}$ *finite witness ledger for displayed simplex, face, or hull witnesses used by the subfamily checks.*
5. *S is the finite subfamily intersection ledger, recording each inspected subfamily and its displayed nonempty intersection witness.*
6. *I is the resulting full-family intersection row exposed by the packet after the finite ledger has been replayed.*
7. *T records componentwise* hsame *transport for $A, V, F, W, S, I, C, P, N$.*
8. *C records displayed* Cont *replay from ambient affine data through convex-family rows, finite polytope witnesses, subfamily intersection checks, the full-family intersection row, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HellyTheorem}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, V, F, W, S, I, T, C, P, N$. It has no coordinate for host convex equality, ambient standard-model convexity, choice-selected points, separation or compactness theorems not displayed by the finite rows, or an infinite family of constraints.

Lean4 ✓ `BEDC.Derived.HellyTheoremUp.HellyTheoremTasteGate single carrier alignment`

2278.2 Helly NameCert Surface

Theorem 2278.2 (HellyTheorem NameCert obligations)**.** *For every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *carrier*

$$H = (A, V, F, W, S, I, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the ambient affine row, vector displacement row, finite convex-family row, finite polytope witness ledger, subfamily intersection ledger, full-family intersection row, transport, replay, provenance, and local naming;*
2. *ambient compatibility, requiring the* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *rows to be the only coordinates through which convex sets and polytope witnesses are read;*
3. *convex-family exposure, requiring each displayed family member to pass through a* $\mathsf{ConvexSet}^{\uparrow}$ *row before it enters any subfamily intersection check;*

4. *finite witness exactness, requiring every inspected subfamily intersection and every exported full-family intersection row to be backed by the displayed* $\mathsf{Polytope}^{\uparrow}$ *witness ledger;*
5. *subfamily ledger stability, requiring the finite intersection ledger* S *to be transported componentwise by* T *and replayed by* C *before* I *is exposed;*
6. *nonescape, excluding host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, infinite-family claims, Lean verification, and bridge checking.*

No HellyTheorem obligation is stored outside these rows.

Proof. Project the accepted carrier. The ambient coordinates are A and V, the convex input is the finite family row F, and the finite witness coordinates are W, S, I.

Ambient compatibility is forced because the carrier has no second affine or vector coordinate. Every convex-family read therefore passes through F over the same A, V boundary.

The witness ledger W and the subfamily ledger S are the only rows that record inspected finite intersections. Transport T and replay C preserve those rows componentwise before the exported row I is read.

The remaining coordinates are provenance and local naming. Since no coordinate stores host convex equality, a choice-selected point, a separation theorem, an unlisted compactness principle, an infinite-family assertion, Lean verification, or bridge data, those objects cannot become local NameCert obligations. □

Theorem 2278.3 (Helly finite-intersection route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet routes a finite convex-intersection request through*

$$\mathsf{AffineSpace}^{\uparrow}, \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{Polytope}^{\uparrow} \longmapsto S \longmapsto I.$$

The route exposes ambient affine data, vector displacement data, convex-family membership, finite polytope witnesses, and the finite subfamily ledger before the full-family intersection row is available. A request bypassing one of these rows is outside the accepted $\mathsf{HellyTheorem}^{\uparrow}$ *carrier.*

Proof. Read the carrier of Definition 2309.1. The first two coordinates bind the ambient affine and vector rows, and the convex-family coordinate F is interpreted only over that ambient boundary.

The polytope witness row W and the finite subfamily ledger S supply the displayed intersection checks. The replay row C presents them before the full-family row I is exposed.

Thus a consumer reaches I only along the stated route. If any one of the ambient, convex-family, polytope-witness, or subfamily-ledger rows is missing, the packet has no finite coordinate from which the requested intersection row can be read. □

Theorem 2278.4 (Helly finite-intersection forward link)**.** *Every finite convex-intersection request accepted by a* $\mathsf{HellyTheorem}^{\uparrow}$ *packet consumes the finite closed-family indexing discipline of chapter 1792 before the full-family intersection row* I *is exposed. The* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *sibling supplies the finite-family ledger boundary, while* $\mathsf{HellyTheorem}^{\uparrow}$ *supplies the convex and affine specialization over* $A, V, F, W, S, I, T, C, P, N$.

Proof. Project the carrier of Definition 2309.1. The ambient and specialization rows are A, V, F, W, the finite subfamily ledger is S, the full-family intersection row is I, and T, C, P, N transport, replay, source, and name the same packet.

Apply Definition 2309.3. It requires the request to pass through the finite subfamily ledger S before I is available, so a consumer cannot read the full-family row directly from the ambient affine or convex-family rows.

The finite intersection property sibling provides the closed finite-family indexing boundary named in chapter 1792. Helly contributes the convex, affine, vector, and polytope-witness specialization attached to that boundary, rather than replacing it by a second indexing principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 2309.1, Definition 2309.2, and Definition 2309.3 seed the finite convex-intersection packet over affine ambient rows, vector displacement rows, convex-family rows, polytope witness ledgers, subfamily intersection ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, infinite-family Helly claims, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ambient compatibility, convex-family exposure, finite witness exactness, subfamily ledger stability, and finite-intersection routing for BEDC.Derived.HellyTheoremUp.

Constructive story: A $\mathsf{HellyTheorem}^\uparrow$ packet is generated from affine ambient data, vector displacement data, finite convex-family rows, finite polytope witness rows, finite subfamily intersection ledgers, a full-family intersection row, componentwise transport, continuation replay, provenance, and local NameCert rows.

The finite subfamily ledger exported by $\mathsf{HellyTheorem}^\uparrow$ is consumed by chapter 2347 through a concrete dependency route: first the convex-family membership rows are read from chapter 208, then the finite witness ledger is read from chapter 209, then the Helly full-family intersection row is exposed, and only after those rows are fixed may the Caratheodory support-reduction packet read a bounded support surface. This route records a downstream use of $\mathsf{HellyTheorem}^\uparrow$ without adding a theorem environment to the hub.

Theorem 2278.5 (Helly Caratheodory forward dependency)**.** *Every* $\mathsf{HellyTheorem}^\uparrow$ *finite-subfamily read that is exported to a convex-hull consumer must first consume the* $\mathsf{CaratheodoryConvexHull}^\uparrow$ *support-reduction route of chapter 2347 and the convex-hull carrier rows already displayed in Definition 2309.12. Only after those rows are visible may the packet export the Helly finite-intersection route of Definition 2309.18. The dependency is a finite forward read through the displayed convex-family, polytope, Caratheodory, and local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$ *rows, not a host separation theorem, selected intersection point, or ambient convexity principle.*

Proof. Project the existing Helly dependency route. The convex-family rows and finite witness rows are displayed before the Helly subfamily ledger is inspected.

Next read the Caratheodory convex-hull surface. The support-reduction packet consumes the displayed subfamily ledger only as finite bounded-support data, so the convex-hull carrier remains upstream of the Helly finite-intersection export.

Finally repack the rows into the local Helly NameCert route. Since no row names a host separation theorem, selected intersection point, or ambient convexity principle, the exported dependency is exactly the finite forward route named in the statement. □

Theorem 2278.6 (Helly Caratheodory sibling route)**.** *The* $\mathsf{HellyTheorem}^\uparrow$ *finite-subfamily ledger is available to the* $\mathsf{CaratheodoryConvexHull}^\uparrow$ *consumer of chapter 2347 only after the* $\mathsf{ConvexSet}^\uparrow$, $\mathsf{Polytope}^\uparrow$, $\mathsf{AffineSpace}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, *and* $\mathsf{CompactMetric}^\uparrow$ *rows expose the finite convex-family boundary. The consumed route is the displayed finite ledger through those rows and the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$ *packet; it is not a detached convexity principle, a selected intersection point, or a host support-reduction theorem.*

Proof. Project the Helly ledger as a finite subfamily packet. Its convex-family membership, polytope witness, affine and vector-space coordinates, and compactmetric boundary rows must be visible before a convex-hull consumer can read the finite-intersection surface.

Then pass the exposed ledger to the Caratheodory consumer. The consumer receives bounded-support data from the displayed packet, so the route remains tied to the same local NameCert rows.

No additional coordinate names host convexity, selected intersection points, or an external support theorem. The sibling route is therefore the finite dependency stated above. □

Theorem 2278.7 (Helly Caratheodory finite-dependency audit)**.** *Any* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily ledger exported to* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *must display* $\mathsf{ConvexSet}^{\uparrow}$ *membership,* $\mathsf{Polytope}^{\uparrow}$ *finite witnesses,* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *ambient rows, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read boundary, and the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row before the Caratheodory bounded-support surface is consumed. The route exports no host separation theorem, selected intersection point, quotient convex-set equality, or ambient convexity principle.*

Proof. Start with the forward dependency theorem, Definition 2309.5. It places the convex-family and finite-witness rows upstream of any convex-hull consumer.

Apply the sibling route theorem, Definition 2309.6. It refines the dependency surface to $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and the local Helly naming row before the Caratheodory surface is read.

The carrier non-escape boundary in chapter 2309 excludes host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, and infinite-family claims. Hence the Caratheodory handoff is exactly the finite dependency audit stated above. □

Lemma 2278.8 (Caratheodory–Helly finite dependency route)**.** *Every* $\mathsf{HellyTheorem}^{\uparrow}$ *consumer that uses Caratheodory-style convex support data must first expose the finite convex-family row of chapter 208, the finite witness and face rows of chapter 209, and the bounded-support handoff of chapter 2347. Only after those rows are visible may the Helly packet read its subfamily-intersection ledger and export the full-family intersection row through* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$.

Proof. Project the $\mathsf{HellyTheorem}^{\uparrow}$ carrier from Definition 2309.1. The convex-family coordinate is displayed before the subfamily ledger can be inspected, and the finite witness rows are carried separately from the ambient affine data.

The Caratheodory convex-hull packet of chapter 2347 then supplies only bounded support rows. It does not select an ambient intersection point or add a host separation theorem.

Thus the dependency lattice is finite: convex-family rows and polytope witnesses feed the Caratheodory support route, and that route feeds the Helly subfamily ledger before the full-family row is exported. □

Theorem 2278.9 (Helly Caratheodory convex-hull sibling route)**.** *Every finite-intersection consumer that reads an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet must read the convex-family row through the* $\mathsf{ConvexSet}^{\uparrow}$ *surface of chapter 208, the Caratheodory convex-hull route of chapter 2347, the finite closed-family boundary of chapter 1792, and the Helly carrier of chapter 2309 before the finite intersection witness is exported. The route is a dependency-lattice projection; it does not export an isolated convex-family oracle, ambient separation theorem, quotient convex-hull equality, or selected intersection point.*

Proof. Project the Helly carrier in chapter 2309. Its finite intersection witness is downstream of the convex-family row, finite closed-family indexing, affine or convex support rows, transport, replay, provenance, and local naming.

The convex-family row is first read through the $\mathsf{ConvexSet}^{\uparrow}$ sibling of chapter 208. The finite support and convex-hull read then passes through the Caratheodory route of chapter 2347, which records the displayed finite-combination dependency rather than an ambient hull operator.

Finally the finite-family ledger is read through chapter 1792. These three sibling routes are already dependency rows for the Helly packet, so composing them before the witness export is only a projection through the existing carrier and lattice.

Since no listed row carries an isolated convex-family oracle, separation theorem, quotient hull equality, or selected intersection point, the exported Helly witness remains the displayed finite-intersection route. □

Theorem 2278.10 (Helly finite-subfamily ledger route)**.** *Every* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily ledger consumed by* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *must read* $\mathsf{ConvexSet}^{\uparrow}$ *membership rows,* $\mathsf{Polytope}^{\uparrow}$ *finite witness rows,* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *ambient rows, and* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read boundaries before exporting the bounded support surface. The ledger route is therefore*

$$\mathsf{AffineSpace}^{\uparrow}, \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{Polytope}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CaratheodoryConvexHull}^{\uparrow}.$$

It exposes only the named finite subfamily data and does not introduce a host separation theorem, a selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

Proof. Project the existing $\mathsf{HellyTheorem}^{\uparrow}$ carrier of Definition 2309.1. Its ambient rows are A, V, its convex-family row is F, and its finite witness and subfamily rows are W, S, I.

Follow the sibling route already displayed by the chapter: $\mathsf{ConvexSet}^{\uparrow}$ supplies membership rows, $\mathsf{Polytope}^{\uparrow}$ supplies finite witnesses, $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ supply the ambient affine and displacement rows, and $\mathsf{CompactMetric}^{\uparrow}$ supplies the finite-read boundary. The $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ consumer can read a bounded support surface only after those rows have been displayed.

No carrier coordinate stores host separation, a selected intersection point, quotient convex-set equality, or ambient standard-model convexity. The exported ledger is therefore exactly the finite sibling route stated above. □

Proposition 2278.11 (Helly convex-hull sibling dependency)**.** *The* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection certificate supplies its convex-hull dependency only through the finite subfamily ledger consumed by chapter 2347. Thus any Caratheodory convex-hull readback using* $\mathsf{HellyTheorem}^{\uparrow}$ *must display the convex-family classifier, finite witness ledger, full-family intersection row, and local NameCert provenance before the Caratheodory finite-generation route may expose bounded support data.*

Proof. Use Definition 2309.10. It already forces the Helly ledger through the convex-family, polytope, affine, vector, and compact finite-read rows before the Caratheodory convex-hull surface is available.

The sibling chapter chapter 2347 then consumes that ledger as its Helly handoff row. Since neither the Helly carrier nor the Caratheodory convex-hull carrier stores a host separation theorem, selected intersection point, or quotient convex-hull equality, the dependency remains the displayed finite-generation route and nothing larger. □

Consequently every $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ or finite convex-family consumer may read $\mathsf{HellyTheorem}^{\uparrow}$ only after the $\mathsf{ConvexSet}^{\uparrow}$ membership rows of chapter 208, the $\mathsf{Polytope}^{\uparrow}$ finite witness rows, the $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ ambient rows, and the $\mathsf{CompactMetric}^{\uparrow}$ finite-read boundary have been

displayed. The consumption route then reaches chapter 2347 as a finite subfamily ledger, not as a host convexity theorem or an uncarried intersection selector. The finite subfamily ledger of $\mathsf{HellyTheorem}^{\uparrow}$ is read through the horizontal convex lattice before any convex-hull consumer may use it. The route first fixes the convex-family rows of chapter 208, the finite witness rows of chapter 209, the affine and vector ambient rows of chapter 206 and chapter 47, and the support-reduction boundary of chapter 2347; only after those rows are displayed may the Helly finite subfamily ledger be consumed as convex-hull input.

Convex-hull and Caratheodory dependency. The $\mathsf{HellyTheorem}^{\uparrow}$ packet consumes its finite convex-family carrier through the convex-set surface of chapter 208 and its affine-combination reduction through the Caratheodory convex-hull surface of chapter 2347 before it exports an intersection certificate. Thus the Helly read is not an isolated incidence claim: it first exposes the convex-family rows, then the Caratheodory finite-combination rows, and only then packages the common-intersection row through its local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ surface.

Theorem 2278.12 (Helly theorem convex-hull forward link)**.** *Every convex-hull consumer of an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet must read the displayed finite family row, full-family intersection witness, Caratheodory-style finite subfamily row,* $\mathsf{ConvexSet}^{\uparrow}$ *classifier rows, and local* $\mathsf{NameCert}$ *provenance before it uses the Helly packet. The forward link factors through the Helly construction of chapter 2309, the Caratheodory dependency route in Definition 2309.10, and the convex-hull surface of chapter 2347. It supplies no full compact convexity theorem and no host separation theorem.*

Proof. Project the Helly carrier. Its finite family row is F, its full-family intersection witness is I, and its local naming and provenance are carried by P, N.

Use the existing Caratheodory forward dependency route. That route reads $\mathsf{ConvexSet}^{\uparrow}$ membership rows, $\mathsf{Polytope}^{\uparrow}$ finite witnesses, affine and vector ambient rows, and compact finite-read boundaries before a Caratheodory convex-hull consumer can read a bounded support surface.

The present forward link names the same boundary for convex-hull consumers. The consumer may use the Helly packet only after the finite family, intersection witness, finite subfamily ledger, convex-set classifier rows, and local NameCert provenance have all been displayed.

Since the Helly carrier has no coordinate for compact convexity as a host theorem and no coordinate for a separation theorem, neither principle is exported by this link. □

Theorem 2278.13 (Helly convex-hull forward scope)**.** *The finite-intersection row of an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet is a forward scope for the* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *consumer of chapter 2347. A convex-hull membership request may consume the Helly row only after the convex-set and polytope sibling packets have displayed their finite family, face-boundary, support, and coefficient ledgers. The handoff exports a finite subfamily certificate for convex-hull reasoning; it does not export ambient separation, selected minimal supports, quotient hull equality, host convexity, or Real completeness.*

Theorem 2278.14 (Helly convex-family dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily read must display the* $\mathsf{ConvexSet}^{\uparrow}$ *family rows and the* $\mathsf{Polytope}^{\uparrow}$ *finite witness ledger before the subfamily ledger S of Definition 2309.1 is inspected. The family row consumes the convex carrier of chapter 208 before the finite-intersection ledger is read. The ambient* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *rows fix the affine boundary for those convex sets, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read discipline bounds the inspected family window, and any* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *consumer receives only the displayed subfamily ledger after those rows are visible. The route exports no host separation*

theorem, choice-selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

Proof. Project the carrier of Definition 2309.1. The convex-family coordinate is F, the finite witness coordinate is W, the ambient coordinates are A, V, and the inspected subfamily coordinate is S.

The row F is read through $\mathsf{ConvexSet}^{\uparrow}$, while W is read through $\mathsf{Polytope}^{\uparrow}$. The rows A, V keep those reads in the displayed affine and vector ambient surface, and the compact finite-read boundary fixes the inspected finite window before S is replayed.

A Caratheodory convex-hull consumer may therefore read the Helly handoff only after the convex-family, polytope, affine, vector, and compact rows have been displayed. Since the structural rows only transport, replay, source, and name this packet, no host separation theorem, selected point, quotient convex-set equality, or ambient standard model is exported. □

Theorem 2278.15 (Helly convex-geometry dependency lattice)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection read consumes the convex-geometry lattice*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{Polytope}^{\uparrow},\ \mathsf{CaratheodoryConvexHull}^{\uparrow}$$

before it exposes its local intersection row. The $\mathsf{ConvexSet}^{\uparrow}$ *surface of chapter 208 supplies the membership and convex-family rows, the* $\mathsf{Polytope}^{\uparrow}$ *surface of chapter 209 supplies the finite witness rows, and the* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *surface of chapter 2347 supplies the bounded-support handoff consumed by convex-hull readers. A Helly packet that omits one of these rows is outside the accepted sibling route.*

Proof. Project the carrier in Definition 2309.1. Its convex-family coordinate is F, its finite witness coordinate is W, and its subfamily and full-family intersection coordinates are S, I.

The row F is read through the $\mathsf{ConvexSet}^{\uparrow}$ classifier, so no finite subfamily can be inspected before the convex-family membership rows have been displayed. The row W is the finite $\mathsf{Polytope}^{\uparrow}$ witness ledger, so every inspected subfamily intersection is backed by the polytope witness surface before S is replayed.

The convex-hull consumer then reads the same finite subfamily ledger through $\mathsf{CaratheodoryConvexHull}^{\uparrow}$. Since the carrier has no alternate row for a host separation theorem, selected intersection point, quotient convex-set equality, or ambient standard-model convexity, the three named sibling surfaces form the whole dependency lattice for the Helly read. □

Consequently $\mathsf{HellyTheorem}^{\uparrow}$ is not an isolated hub: its public convex-geometry use is routed through $\mathsf{ConvexSet}^{\uparrow}$ membership rows, $\mathsf{Polytope}^{\uparrow}$ finite witness rows, and $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ bounded-support rows before any local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ intersection certificate is consumed.

Proposition 2278.16 (Helly convex-geometry sibling lattice route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection packet consumes the convex-set membership rows of chapter 208, the polytope finite witness rows of chapter 209, and the convex-hull support surface of chapter 2347 through the same finite intersection data before any local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row is read. The route is a sibling dependency lattice over displayed finite subfamilies, not a host convexity theorem or a choice-selected intersection point.*

Proof. Project the carrier of Definition 2309.1. The convex-family row F supplies only $\mathsf{ConvexSet}^{\uparrow}$ membership data, while the witness row W supplies the $\mathsf{Polytope}^{\uparrow}$ finite face, simplex, or hull witnesses used by inspected subfamilies.

The finite subfamily ledger S is then the shared handoff. A convex-hull consumer reads it through $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ as bounded support data, and the Helly packet reads the same ledger before exposing the full-family row I.

Since T, C, P, N only transport, replay, source, and name the displayed rows, they cannot introduce a second convex-hull carrier, an ambient separation argument, or a selected intersection point. Thus the sibling lattice is exactly the finite route through $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, the Caratheodory convex-hull surface, and the local Helly ledger. □

Consequently the $\mathsf{HellyTheorem}^{\uparrow}$ hub records a concrete convex-geometry sibling dependency: finite intersection data is read only after the membership, finite witness, and convex-hull support rows have been displayed inside the same BHist, hsame, Cont, Pkg, and NameCert surface. The route is the same finite sibling lattice, not a host convexity theorem.

Theorem 2278.17 (Helly convex polytope lattice route)**.** *Any accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection packet reads its convex family through* $\mathsf{ConvexSet}^{\uparrow}$*, its finite cell witnesses through* $\mathsf{Polytope}^{\uparrow}$*, and its ambient coordinates through* $\mathsf{AffineSpace}^{\uparrow}$ *before it exposes the local intersection ledger. In carrier notation, the public route first displays the convex-family row F, the polytope witness row W, and the affine ambient row A of Definition 2309.1; only then may the subfamily ledger S and full-family row I be consumed. The route exports no host convex-set equality, selected intersection point, separation theorem, quotient affine coordinate, or ambient standard-model convexity.*

Proof. Project the Helly carrier of Definition 2309.1. Its ambient row is A, its convex-family row is F, and its finite witness row is W.

The convex-family row is read through chapter 208. Thus an accepted packet displays the $\mathsf{ConvexSet}^{\uparrow}$ membership and convexity rows before a subfamily can be inspected.

The finite cell witnesses are read through chapter 209, while the ambient coordinates are read through chapter 206. These rows bind the finite cells to the same affine boundary before the ledger rows S and I are exposed.

Finally inspect the remaining coordinates T, C, P, N. They transport, replay, source, and name the same finite packet, and they add no host convex equality, selected intersection point, separation theorem, quotient affine coordinate, or ambient standard-model convexity. The stated convex-polytope-affine route therefore exhausts the public Helly read. □

Theorem 2278.18 (Helly theorem convex-polytope dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily read displays the* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{AffineSpace}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and compact finite-read rows before the finite-intersection ledger is consumed. The route may inspect the Helly family row, finite subfamily row, and intersection ledger only after those sibling rows are present; it exports no selected intersection point, ambient separation theorem, quotient affine coordinate, or host compactness oracle.*

Proof. Project the Helly carrier of Definition 2309.1. The convex-family row, finite witness row, ambient row, and finite-intersection ledger are explicit packet coordinates, so a consumer read starts from displayed rows rather than from a host theorem about arbitrary convex families.

Use the sibling dependency files already attached to this chapter. The convex set dependency route supplies the $\mathsf{ConvexSet}^{\uparrow}$ membership and convexity rows, while the convex-polytope lattice route supplies the $\mathsf{Polytope}^{\uparrow}$ finite-cell witnesses and the $\mathsf{AffineSpace}^{\uparrow}$ ambient boundary.

The finite convex lattice route adds the $\mathsf{VecSpace}^{\uparrow}$ scalar row and the compact finite-read rows needed before a finite subfamily can be consumed. With those rows displayed, the Helly ledger reads only the finite-intersection surface carried by the packet.

Finally replay the local transport, provenance, and naming coordinates. They preserve the same finite route and do not add a selected point, separation theorem, quotient affine coordinate, or host compactness oracle. Hence the convex-polytope dependency route is exhausted before the finite-intersection ledger is read. □

Theorem 2278.19 (Helly convex-set dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite convex-intersection read first consumes the convex carrier of chapter 208 and the finite polytope witness surface of chapter 209. Only after those two sibling rows are displayed may the Helly intersection row expose its finite subfamily ledger, common-intersection row, transport, replay, provenance, and local naming. The route exports no host separation theorem, selected intersection point, quotient convex-set equality, unrestricted finite-cover oracle, or ambient standard-model convexity.*

Proof. Project the Helly carrier of Definition 2309.1. The convex-family row is consumed through the convex-set certificate, and the finite witness row is consumed through the polytope certificate.

After those sibling rows are visible, the Helly subfamily and intersection rows can be read as finite BHist data. The structural rows transport, replay, source, and name the same route, so the carrier has no coordinate for a host separation theorem, selected point, quotient convex-set equality, or unrestricted cover oracle. □

The finite-intersection certificates used by $\mathsf{HellyTheorem}^{\uparrow}$ enter through the same convex-row surface as chapter 208 and the finite hull route of chapter 2347. A Helly carrier can therefore read its displayed family of convex sets, witness subfamilies, and intersection rows as finite BHist data before it exports the Helly conclusion, rather than importing an ambient convexity theorem or an unrestricted finite-cover oracle.

Theorem 2278.20 (Helly convex dependency anchor)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet consumes finite convex-family rows and finite-intersection rows before it exports the Helly* $\mathsf{NameCert}$ *surface. The convex-family rows are read through the* $\mathsf{ConvexSet}^{\uparrow}$ *membership surface of chapter 208; the finite-intersection rows are read through the subfamily ledger of chapter 1792. Only after both rows are displayed may the packet expose its full-family intersection witness, transport rows, replay rows, provenance, and local naming.*

Proof. Project the carrier of Definition 2309.1. Its family coordinate cannot be read as an ambient convexity assertion; each member is inspected through the displayed convex-set rows.

Use the finite-intersection dependency route already recorded for the Helly packet. That route keeps the subfamily ledger visible before the full-family intersection row is exported.

The remaining rows only transport, replay, source, and name the same finite surface. Hence the Helly export is anchored in the convex-family and finite-intersection siblings rather than in a detached kernel-level theorem. □

Theorem 2278.21 (Helly finite convex-family budget)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-family read must expose* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{AffineSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows before it exports a finite intersection witness. In particular, the convex-set membership rows of chapter 208, the finite witness rows of chapter 209, the compact finite-read rows of chapter 60, and the ambient affine and vector rows of chapter 206 and chapter 47 form the displayed dependency route for the local intersection ledger.*

Proof. Project the $\mathsf{HellyTheorem}^{\uparrow}$ carrier from Definition 2309.1. The convex-family coordinate F is read through the $\mathsf{ConvexSet}^{\uparrow}$ surface, the witness coordinate W is read through $\mathsf{Polytope}^{\uparrow}$, and

the ambient rows A, V are read through $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ before the finite subfamily ledger S can be inspected.

The compact finite-read discipline supplies the bounded inspection surface for those finite subfamilies. Transport, continuation replay, provenance, and local naming are then carried by the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Therefore the exported finite intersection witness is downstream of the sibling convex, polytope, compact, vector, and affine chapters rather than a detached host convexity fact. □

The $\mathsf{HellyTheorem}^{\uparrow}$ packet consumes the finite-intersection lattice of chapter 1792 before any Helly carrier can export an intersection witness. The route also reads the convex carrier rows of chapter 208 and the compact finite-cover boundary of chapter 129; together with the local sibling route of Definition 2309.10, these rows force the witness to be a finite NameCert consequence rather than a kernel-only intersection assertion. The finite-family carrier for $\mathsf{HellyTheorem}^{\uparrow}$ reads the finite-intersection route of chapter 1792 together with the $\mathsf{ConvexSet}^{\uparrow}$ surface and the convex-family coordinate F before it can export a Helly subfamily witness. This bridge keeps the witness surface dependent on displayed finite-intersection and convex-family data rather than on an isolated kernel assertion.

Chapter 2279

A Concrete Naming Certificate for $\mathsf{TemperedDistribution}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{TemperedDistribution}^{\uparrow}$ packet records the finite observable face of a tempered distribution as a continuous linear action on Schwartz-test windows. The chapter connects the distribution row of chapter 186, the finite Fourier observation surface of chapter 124, the weak-derivative route of chapter 2210, the Sobolev comparison surface of chapter 440, and the terminal real readback of chapter 38. It names only the BEDC certificate surface for these finite rows; it does not import a completed Schwartz space, a host distribution equality relation, quotient dual spaces, or a global topological-vector-space completeness theorem.

2279.1 Carrier definition

Definition 2279.1 (Tempered distribution carrier)**.** *A* $\mathsf{TemperedDistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, G, A, F, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite Schwartz-test window row. It records the named test functions visible to the packet, their finite derivative orders, polynomial weights, support windows when present, and the requested observation indices.*
2. *G is the polynomial-growth ledger. It records the displayed seminorm bounds used to admit each test row and each derivative row.*
3. *A is the* $\mathsf{Distribution}^{\uparrow}$ *action row. It records the finite list of test-window evaluations and their* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoints.*
4. *F is the* $\mathsf{Fourier}^{\uparrow}$ *coefficient and test-transform row. It records finite transform observations for the same test windows and the coefficient windows consumed by Fourier-facing reads.*
5. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *and* $\mathsf{Sobolev}^{\uparrow}$ *compatibility row. It records integration-by-parts signs, derivative-order shifts, and Sobolev-window comparisons that are explicitly displayed in the packet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational approximation windows of the action and transform endpoints.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the displayed readback route.*
8. *L is the local classifier row. It admits two packets together exactly when their Schwartz-test windows, polynomial-growth ledgers, distribution-action rows, Fourier rows, weak-derivative and Sobolev rows, rational readbacks, and real seals match componentwise.*
9. *H records componentwise* hsame *transport for all visible rows.*
10. *C records displayed* Cont *replay from a tempered-distribution request through $S, G, A, F, W, Q, E, L, H$ in that order.*
11. *P is the* Pkg *provenance row over* $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores a completed Schwartz space, an unrestricted continuous dual, host equality of distributions, quotient test-function equality, selected representative, countable choice schedule, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.TemperedDistributionUp.TemperedDistributionTasteGate single carrier alignment`

2279.2 The certificate

Theorem 2279.2 (Tempered distribution NameCert obligations)**.** *For every accepted* $\mathsf{TemperedDistribution}^{\uparrow}$ *carrier $T = (S, G, A, F, W, Q, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Schwartz-test window, polynomial-growth ledger, distribution-action, Fourier, weak-derivative and Sobolev compatibility, rational readback, real seal, classifier, transport, replay, provenance, and local naming rows;*
2. *action exactness, requiring every displayed evaluation of the distribution against a test window to pass through A after the corresponding test row in S and growth ledger in G have been opened;*
3. *polynomial-growth stability, requiring tightened derivative or weight requests to replay the same ledger G before any action or transform endpoint is read;*
4. *Fourier compatibility, requiring every finite transform or coefficient read to pass through F and then through the same Q, E readback route as the action row;*
5. *weak-derivative and Sobolev compatibility, requiring every derivative-order shift, sign row, or Sobolev-window comparison to pass through W before the packet is exported;*
6. *componentwise* hsame, *displayed* Cont *replay,* Pkg *provenance, and local* NameCert *closure through L, H, C, P, N.*

No obligation outside these rows is part of the certificate.

Proof. Project the carrier of Definition 2310.1. The row inventory gives the whole finite packet, so carrier admission is exhausted by $S, G, A, F, W, Q, E, L, H, C, P, N$.

The Schwartz-test row S fixes which tests and derivative orders are visible. The growth ledger G must be read before any action endpoint, because it is the row that admits the displayed seminorm and polynomial-weight data.

The action row A then supplies only finite evaluations against those admitted tests. The rational and real endpoints of those evaluations pass through Q and E, so there is no detached real seal or host distribution value.

The Fourier and derivative routes are likewise internal rows. The Fourier row F records finite coefficient and test-transform observations, while W records the weak-derivative signs and Sobolev-window comparisons. Transport, replay, provenance, and naming rows compare and repack the same data, and none of them adds a completed test-function space, unrestricted dual, quotient equality, selected representative, choice schedule, or ambient completeness theorem. □

Theorem 2279.3 (Tempered distribution Fourier handoff)**.** *Every accepted* $\mathsf{TemperedDistribution}^{\uparrow}$ *read exported to a Fourier-facing consumer factors through*

$$S, G, A \longmapsto F \longmapsto W, Q, E \longmapsto L, H, C, P, N.$$

Thus the consumer receives the displayed Schwartz-test window, polynomial-growth ledger, distribution-action row, finite Fourier coefficient and test-transform row, weak-derivative and Sobolev compatibility row, rational readback, real seal, classifier, componentwise transport, replay, provenance, and local naming. It receives no completed Schwartz topology, host distribution equality, quotient dual object, selected representative, countable choice schedule, Plancherel theorem, inversion theorem, or ambient completeness principle.

Proof. Start with a Fourier-facing read. The test row S and growth ledger G must be visible before the packet can expose a distribution action, because the finite transform request is attached to the same admitted test windows.

Read A next. It supplies the finite action endpoints against the admitted tests and records the source $\mathsf{Distribution}^{\uparrow}$ route for those endpoints.

The handoff to F is then a finite-observation handoff: coefficient windows and test-transform rows are exposed as $\mathsf{Fourier}^{\uparrow}$ packages, not as host transforms of a completed function space. The compatibility row W records exactly the weak-derivative signs and Sobolev comparisons that are part of the same request.

Finally Q, E, L, H, C, P, N read back, compare, replay, source, and name the same route. Since none of these rows contains a completed Schwartz topology, host equality, quotient dual, selected representative, choice schedule, Plancherel theorem, inversion theorem, or ambient completeness witness, the Fourier consumer receives only the displayed finite certificate. □

Falsifiable prediction: Every accepted $\mathsf{TemperedDistribution}^{\uparrow}$ consumer that reaches a Fourier coefficient, weak-derivative sign, Sobolev comparison, or $\mathsf{Real}^{\uparrow}$ seal without first reading the finite Schwartz-test window and polynomial-growth ledger refutes the carrier surface.

Independence witness: distribution, fourier, weakderivative, sobolev, regseqrat, real: none is bijective to $\mathsf{TemperedDistribution}^{\uparrow}$ because this packet jointly carries finite Schwartz-test windows, polynomial-growth ledgers, distribution actions, Fourier coefficient and transform rows, weak-derivative and Sobolev compatibility rows, rational readbacks, real seals, classifier, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$.

Constructive story: A $\mathsf{TemperedDistribution}^{\uparrow}$ packet is generated from finite Schwartz-test windows, polynomial-growth ledger rows, displayed $\mathsf{Distribution}^{\uparrow}$ actions, finite $\mathsf{Fourier}^{\uparrow}$ coefficient and test-transform observations, $\mathsf{WeakDerivative}^{\uparrow}$ and $\mathsf{Sobolev}^{\uparrow}$ compatibility rows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2310.1, Definition 2310.2, and Definition 2310.3 seed the finite tempered-distribution interface over Schwartz-test windows, polynomial-growth ledgers, distribution actions, Fourier observations, weak-derivative and Sobolev compatibility, rational readbacks, and real seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{TemperedDistribution}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed Schwartz spaces, unrestricted continuous duals, quotient test-function equality, host distribution equality, selected representatives, countable choice, Plancherel, inversion, or ambient completeness.

Upgrade path: A formal route can encode the finite carrier and prove the NameCert obligation surface and Fourier handoff theorem.

Chapter 2280

A Concrete Naming Certificate for $\mathsf{DirichletKernel}^{\uparrow}$

Following Zygmund and Katznelson on trigonometric series, the $\mathsf{DirichletKernel}^{\uparrow}$ packet records the finite symmetric Fourier kernel that evaluates one displayed partial-sum window before any convergence-facing consumer may read a series or limit handoff. It sits over the Fourier coefficient interface of chapter 124, the positive averaging sibling $\mathsf{FejerKernel}^{\uparrow}$ of chapter 1939, the complex arithmetic carrier of chapter 39, the circle-source samples of chapter 58, and the real-series partial-sum corridor of chapter 1581. The chapter names a finite trigonometric partial-sum kernel; it does not import host pointwise limits, ambient Fourier convergence, quotient function spaces, countable choice, or a selected infinite sum.

Definition 2280.1 (DirichletKernel carrier)**.** *A* $\mathsf{DirichletKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, A, S, Z, F, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the frequency-window row, recording the symmetric finite interval of integer frequencies and the endpoint truncation bound read by the partial-sum request.*
2. *A is the trigonometric coefficient ledger, recording the displayed coefficient attached to each frequency in W together with its opposite-frequency mate.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *evaluation route, exposing only the finite circle samples and phase rows at which the kernel is inspected.*
4. *Z is the* $\mathsf{Complex}^{\uparrow}$ *summation row, recording the displayed complex additions and multiplications used to accumulate the finite trigonometric polynomial.*
5. *F is the* $\mathsf{Fourier}^{\uparrow}$ *partial-sum handoff row, presenting the same frequency window and coefficient ledger to a Fourier partial-sum consumer.*
6. *R is the* $\mathsf{RealSeries}^{\uparrow}$ *and* $\mathsf{CplxSer}^{\uparrow}$ *provenance row, recording which real or complex partial-sum corridor may read the displayed finite sum.*
7. *H records componentwise* hsame *transport for frequency windows, coefficient ledgers, circle evaluations, complex summation, Fourier handoff, series provenance, replay, package provenance, and local naming.*

8. *C records displayed* Cont *replay from a phase and coefficient request through W, A, S, Z, F, R in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CplxSer}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, A, S, Z, F, R, H, C, P, N$. It has no coordinate for a host pointwise limit, an external convergence theorem, an ambient quotient of functions, countable choice, or a selected infinite-series value.

Lean4 ✓ `BEDC.Derived.DirichletKernelUp.TasteGate.DirichletKernelTasteGate single carrier alignment`

Theorem 2280.2 (DirichletKernel NameCert obligations)**.** *For every accepted* $\mathsf{DirichletKernel}^{\uparrow}$ *carrier $K = (W, A, S, Z, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed frequency window, trigonometric coefficient ledger, circle-evaluation route, complex summation row, Fourier partial-sum handoff, real and complex series provenance, transport, replay, package provenance, and local naming;*
2. *finite-window stability, requiring every transported or replayed kernel read to preserve the same symmetric frequency interval and endpoint truncation bound in W;*
3. *coefficient symmetry, requiring the ledger A to pair each displayed frequency with its opposite-frequency mate before the summation row is consumed;*
4. *convolution and summation handoff, requiring the* $\mathsf{Complex}^{\uparrow}$ *row Z and the* $\mathsf{Fourier}^{\uparrow}$ *row F to expose only the finite trigonometric polynomial determined by W, A, S;*
5. $\mathsf{S}^{1\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, *and* $\mathsf{CplxSer}^{\uparrow}$ *routing, requiring every phase, arithmetic, and partial-sum consumer to factor through S, Z, F, R;*
6. *transport and replay stability, requiring H and C to preserve the ordered route through the same carrier coordinates;*
7. *ledger refusal, requiring the packet to export no host pointwise limit, external Fourier convergence principle, quotient function equality, countable-choice selector, or selected infinite-series value.*

No Dirichlet-kernel obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DirichletKernelUp.DirichletKernelCarrier namecert obligations`
↪ variant `BEDC.Derived.DirichletKernelUp.DirichletKernelNameCertObligations`

Proof. Carrier admission is projection from Definition 2311.1. The displayed tuple exposes exactly the frequency window, coefficient ledger, circle evaluation, complex summation, Fourier handoff, series provenance, transport, replay, package provenance, and local naming rows.

Finite-window stability is read from W. Since the carrier stores one symmetric frequency interval and one endpoint truncation bound, every transported or replayed kernel read names the same finite window before any coefficient or phase row is inspected.

Coefficient symmetry is read from A. The ledger pairs displayed frequencies with their opposite-frequency mates, so the summation row may form the finite trigonometric polynomial without appealing to a host Fourier basis or an ambient function-space quotient.

The handoff rows are then forced through S, Z, F, R. Circle samples enter through $\mathsf{S}^{1\uparrow}$, arithmetic accumulation enters through $\mathsf{Complex}^{\uparrow}$, the partial-sum consumer enters through $\mathsf{Fourier}^{\uparrow}$, and series-facing provenance enters only through the named real and complex series corridors.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Because the packet has no coordinate for pointwise limits, external convergence, quotient function equality, countable choice, or selected infinite sums, the refusal clause is exhausted by the carrier inventory. □

Falsifiable prediction: Every accepted $\mathsf{DirichletKernel}^{\uparrow}$ partial-sum read factors through a frequency window, trigonometric coefficient ledger, $\mathsf{S}^{1\uparrow}$ evaluation route, $\mathsf{Complex}^{\uparrow}$ summation row, $\mathsf{Fourier}^{\uparrow}$ handoff, $\mathsf{RealSeries}^{\uparrow}$ or $\mathsf{CplxSer}^{\uparrow}$ provenance, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fourier, fejer-kernel, complex, s1, realseries: $\mathsf{DirichletKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a symmetric frequency window, opposite-frequency coefficient ledger, circle evaluation route, complex finite summation row, Fourier partial-sum handoff, real and complex series provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletKernel}^{\uparrow}$ packet is generated from a symmetric finite frequency window, a trigonometric coefficient ledger, displayed $\mathsf{S}^{1\uparrow}$ phase samples, a $\mathsf{Complex}^{\uparrow}$ finite summation row, a $\mathsf{Fourier}^{\uparrow}$ partial-sum handoff, $\mathsf{RealSeries}^{\uparrow}$ and $\mathsf{CplxSer}^{\uparrow}$ provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2311.1 and Definition 2311.2 seed the finite Dirichlet-kernel partial-sum surface for Fourier and series consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host pointwise Fourier convergence, quotient function equality, selected infinite sums, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window stability, coefficient symmetry, summation handoff, routing, transport, replay, provenance, and local naming for the same finite packet.

Chapter 2281

A Concrete Naming Certificate for $\mathsf{BishopCauchyReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyReal}^{\uparrow}$ packet records a modulus-equipped Cauchy real presentation as finite BEDC certificate data before the open Real and Completion route reads the terminal seal. It sits after the regular rational readback surface of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus discipline of chapter 292, and the terminal real-name boundary of chapter 38. The packet names the constructive presentation of real data by displayed Cauchy approximants and explicit moduli; it does not make $\mathsf{Real}^{\uparrow}$ a quotient carrier, assert ambient completeness, choose a representative limit, or import countable choice.

This carrier supplies the concrete ledger row used by the Real completeness vision of chapter 9238: every Real-facing read is routed through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, explicit $\mathsf{CauchyModulus}^{\uparrow}$ schedule, $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal. The vision therefore reads this packet as finite completion data, not as quotient-stream equality, host real equality, a selected representative, or ambient completeness.

Definition 2281.1 (Bishop Cauchy real carrier)**.** *A* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row that supplies the rational approximants read by the packet.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that records the finite tolerance-to-stage ledger used to inspect Q.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which a consumer asks for approximation data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic row used to compare requested precisions with the displayed rational approximants.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, M, S, D have been displayed.*
6. *H stores componentwise* hsame *transport for the regular source, modulus, stream schedule, dyadic ledger, real seal, replay, provenance, and local naming rows.*

7. *C stores displayed* Cont *replay of the route*

$$Q, M, S, D \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, S, D, R, H, C, P, N$. It has no coordinate for quotient equality of Cauchy streams, selected host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRealUp.TasteGate.BishopCauchyRealTasteGate single carrier alignment`

Theorem 2281.2 (Bishop Cauchy real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier $B = (Q, M, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$ *surface consists exactly of carrier admission, regular rational source exposure through Q, explicit Cauchy modulus exposure through M, finite stream-window routing through S, dyadic tolerance comparison through D, terminal real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape from quotient stream equality, selected representatives, host limit equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRealUp.BishopCauchyRealNameCertObligations`

Proof. Carrier admission is projection from Definition 2313.1. The accepted packet exposes exactly the regular rational source row, the modulus row, the stream-window schedule, the dyadic tolerance ledger, the real seal, and the structural rows.

The Cauchy presentation is read through Q, M, S, D. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular rational approximants, the $\mathsf{CauchyModulus}^{\uparrow}$ row supplies the displayed tolerance-to-stage ledger, $\mathsf{StreamName}^{\uparrow}$ bounds each read to finite windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance arithmetic.

Only after those rows are visible does the terminal $\mathsf{Real}^{\uparrow}$ seal R appear. Transport, replay, provenance, and naming preserve the same route through H, C, P, N. Since the excluded completion principles are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2281.3 (Bishop Cauchy real RegSeqRat handoff)**.** *Every regular-rational read of an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Thus a Real-facing consumer receives a modulus-controlled regular Cauchy presentation rather than a quotient Cauchy class, a chosen representative, a host limit, or a completeness principle.

Proof. Start with the displayed carrier $B = (Q, M, S, D, R, H, C, P, N)$. The source row Q is the only regular rational source available to the classifier.

The read cannot bypass M, S, D. The modulus row records which finite stages answer a requested tolerance, the stream row restricts the answer to visible windows, and the dyadic row records the endpoint arithmetic used by the tolerance comparison.

The seal row R is downstream of that route. The structural rows then transport, replay, source, and name the same packet, so every exported regular-rational read has the stated factorization and no hidden completion coordinate. □

Theorem 2281.4 (Bishop Cauchy real regular readback seal)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing seal read from an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier* $B = (Q, M, S, D, R, H, C, P, N)$ *must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*. The terminal seal* R *may then be read only as the endpoint of the displayed route*

$$Q, M, S, D \longmapsto R \longmapsto H, C, P, N.$$

Thus a Real-facing consumer receives the regular-readback-to-seal dependency as finite BEDC data, not as a quotient completion, a chosen representative, a host limit equality, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the carrier of Definition 2313.1. The rows Q, M, S, D precede the terminal seal R and are the only displayed source, modulus, window, and tolerance coordinates available to a Real-facing read.

Apply the obligation surface of Definition 2313.2. It identifies carrier admission, source exposure, modulus exposure, stream-window routing, dyadic comparison, terminal sealing, transport, replay, provenance, and local naming as the complete local NameCert surface.

Finally use the RegSeqRat handoff of Definition 2313.3. It forces every regular-rational read through the same $Q, M, S, D, R, H, C, P, N$ route, so the $\mathsf{Real}^{\uparrow}$ seal is terminal for that route and cannot hide quotient completion or host real equality. □

Theorem 2281.5 (Bishop Cauchy real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier is public only as the endpoint of the displayed route*

$$Q, M, S, D \longmapsto R.$$

Consequently the packet exposes a constructive real presentation for the open Real and Completion route while leaving quotient completion, ambient real completeness, selected limits, host real equality, trichotomy, and countable choice outside the exported surface.

Proof. Inspect the carrier order. The terminal seal R is stored after the regular source, modulus, stream-window, and dyadic tolerance rows.

The continuation row C replays that order for consumers. A Real-facing read therefore must display the regular rational approximants, the modulus ledger, the finite observation window, and the dyadic tolerance comparison before the seal can be consumed.

The remaining structural rows H, P, N preserve and name the same boundary. Since no quotient, selected limit, host equality, trichotomy, choice, or ambient completeness coordinate occurs in the packet, the seal is exactly the displayed endpoint of the constructive Cauchy presentation. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyReal}^{\uparrow}$ real-facing read factors through a displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{CauchyModulus}^{\uparrow}$ ledger, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, cauchymodulus, streamname, dyadicratcore, real: $\mathsf{BishopCauchyReal}^{\uparrow}$ is not bijective to any listed sibling because it binds a modulus-controlled Cauchy real presentation, finite observation windows, dyadic tolerance arithmetic, terminal real sealing, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2313.1, Definition 2313.2, Definition 2313.3, and Definition 2313.5 seed the modulus-equipped Cauchy real presentation surface for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopCauchyReal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, ambient Real$^{\uparrow}$ completeness, selected limits, host real equality, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-controlled regular source exposure, finite stream-window routing, dyadic tolerance readback, real-seal boundary, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.BishopCauchyRealUp.

Constructive story: A BishopCauchyReal$^{\uparrow}$ packet is generated from a regular rational source, an explicit Cauchy modulus, finite stream windows, dyadic tolerance rows, a terminal Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2282

A Concrete Naming Certificate for $\mathsf{PointwiseLimit}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki's function-limit terminology, the $\mathsf{PointwiseLimit}^{\uparrow}$ packet records an indexwise limit route for displayed real-valued maps without importing a uniform ledger or a host function-space object. It reads continuous approximants from chapter 130, keeps the limiting observations separate from the $\mathsf{UniformLimit}^{\uparrow}$ surface of chapter 463, schedules finite observations through chapter 126, checks regular rational readbacks through chapter 301, seals values only at chapter 38, keeps source and target distance rows visible through chapter 57, and supplies the generic pointwise-limit sibling read consumed by chapter 1604 and chapter 2196.

Definition 2282.1 (PointwiseLimit carrier)**.** *A* $\mathsf{PointwiseLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Y, F, I, S, Q, R, E, H, C, K, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *context row naming the points at which the family is inspected.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *value-seal context for the observed outputs.*
3. *F is the finite ledger of displayed maps, including the* $\mathsf{ContinuousMap}^{\uparrow}$ *rows when the source family is continuous.*
4. *I is the point-index row saying that each convergence request is attached to one displayed source point and one requested tolerance.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite approximant windows inspected for that point and tolerance.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected output windows.*
7. *R is the pointwise convergence ledger asserting that the selected windows are adequate for the displayed point and tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after X, Y, F, I, S, Q, R have been displayed.*
9. *H records componentwise* hsame *transport for the source, target, map ledger, point-index row, schedule, readback, convergence ledger, value seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a point-and-tolerance request through the finite schedule, readback, pointwise ledger, and RealUp seal.*

11. *K is the* Pkg *provenance over* $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PointwiseLimitUp.PointwiseLimitTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.PointwiseLimitUp.PointwiseLimitUp`

Theorem 2282.2 (PointwiseLimit NameCert obligations)**.** *For every accepted* $\mathsf{PointwiseLimit}^{\uparrow}$ *carrier* $P = (X, Y, F, I, S, Q, R, E, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric context, target metric and RealUp value context, displayed map ledger, point-index row, stream schedule, regular rational readback, pointwise convergence ledger, terminal value seal, transport, replay, provenance, and local naming row;*

2. *point-index visibility, requiring each convergence request to display its source point and requested tolerance through I before any schedule in S is read;*

3. *finite-window scheduling, requiring the requested approximant windows to be selected by S and checked by Q before the pointwise convergence row R can be consumed;*

4. *Real-seal routing, requiring the value row E to be downstream of X, Y, F, I, S, Q, R rather than of an ambient function-space limit;*

5. *sibling handoff compatibility, allowing* $\mathsf{BaireOneFunction}^{\uparrow}$ *and* $\mathsf{DiniContinuity}^{\uparrow}$ *consumers to read the pointwise route while keeping* $\mathsf{UniformLimit}^{\uparrow}$ *as a separate stronger ledger;*

6. *nonescape from uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, and completeness of* $\mathsf{Real}^{\uparrow}$.

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2314.1. The accepted packet exposes exactly the source and target metric rows, map ledger, point-index row, finite schedule, regular rational readback, convergence ledger, value seal, transport, replay, provenance, and local naming row.

Point-index visibility is forced by I. A consumer must name the displayed source point and tolerance before it can inspect the corresponding schedule, so the packet does not contain an unindexed host function limit.

Finite-window scheduling is then read through S and Q. The schedule selects only finite approximant windows, and the regular rational row checks those windows before R records the pointwise convergence ledger for the displayed point.

The RealUp value seal E is downstream of this route. The sibling handoff rows in K name $\mathsf{BaireOneFunction}^{\uparrow}$ and $\mathsf{DiniContinuity}^{\uparrow}$ as consumers of the pointwise route, while $\mathsf{UniformLimit}^{\uparrow}$ remains a separate stronger ledger. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local surface. □

Theorem 2282.3 (PointwiseLimit Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{BaireOneFunction}^{\uparrow}$*, or* $\mathsf{DiniContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseLimit}^{\uparrow}$ *packet reads the limiting value through the finite route*

$$X, Y, F, I, S, Q, R \longmapsto E.$$

The route first displays the source and target metric rows, then the map ledger and point-index row, then the StreamName schedule, RegSeqRat readback, and pointwise convergence ledger, and only then exposes the RealUp value seal. It exports no uniform convergence ledger, continuity-preservation theorem, host function equality, quotient function-space object, choice-selected pointwise limit, or RealUp completeness principle.

Proof. Start with a consumer request for the value seal E. By Definition 2314.1, E is a coordinate of the same finite packet as X, Y, F, I, S, Q, R, and the replay row C records the only displayed route into it.

The source and target rows X, Y are read first, followed by the displayed map ledger F and the point-index request I. The finite schedule S and regular rational readback Q then supply the observed approximant windows for that point and tolerance.

The convergence row R certifies only this pointwise route before E is exposed. Transport, replay, provenance, and local naming through H, C, K, N preserve that order for $\mathsf{BaireOneFunction}^{\uparrow}$ and $\mathsf{DiniContinuity}^{\uparrow}$ consumers, while no carrier coordinate supplies the excluded stronger objects. □

Falsifiable prediction: Every accepted $\mathsf{PointwiseLimit}^{\uparrow}$ value-seal read factors through a displayed point-index row and finite StreamName and RegSeqRat windows before any RealUp value is exposed.

Independence witness: continuousmap, uniformlimit, streamname, regseqrat, baireonefunction: PointwiseLimitUp is not bijective to any listed sibling because it jointly carries a source-point request, an approximant ledger, finite schedule and readback rows, a pointwise convergence ledger, a RealUp value seal, and explicit refusal of the UniformLimitUp ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PointwiseLimit}^{\uparrow}$ packet is generated from source and target MetricUp rows, a displayed map ledger, a point-index row, StreamName schedules, RegSeqRat readbacks, a pointwise convergence ledger, a RealUp value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2314.1, Definition 2314.2, and Definition 2314.3 seed the finite pointwise-limit certificate over displayed source points, approximant windows, regular rational readbacks, and RealUp value seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, point-index visibility, finite-window scheduling, Real-seal routing, sibling handoff compatibility, and nonescape for BEDC.Derived.PointwiseLimitUp.

Chapter 2283

A Concrete Naming Certificate for $\mathsf{RealLogarithm}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLogarithm}^{\uparrow}$ packet names logarithm on positive real data as a finite inverse-route certificate for $\mathsf{RealExponential}^{\uparrow}$. The construction records located positive brackets, dyadic interval refinement, exponential comparison rows, and real-equality endpoint checks; it does not import a host logarithm, analytic continuation, quotient real equality, selected infinite bisection branch, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2283.1 (RealLogarithm carrier)**.** *A* $\mathsf{RealLogarithm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, I, B, E, A, O, C, R, H, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *row naming the positive input real and its located apartness from* 0*.*
2. *I is the* $\mathsf{OpenIntervalReal}^{\uparrow}$ *row carrying the positive bracket interval on which the inverse search is performed.*
3. *B is the finite dyadic bracket ledger, with each row recording a rational interval, midpoint comparison request, and retained positive subinterval.*
4. *E is the* $\mathsf{RealExponential}^{\uparrow}$ *inverse-route row used to compare displayed exponentials against the positive input.*
5. *A is the* $\mathsf{RealAbs}^{\uparrow}$ *control row bounding comparison error and bracket width during the finite read.*
6. *O is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *row that classifies the displayed real comparisons used by the bracket ledger.*
7. *C is the displayed* Cont *route from positive input and bracket rows through exponential comparisons to the endpoint classifier.*
8. *R is the real classifier row, read through* hsame *transport and the displayed comparison rows, that identifies the logarithm endpoint with the inverse route.*
9. *H is the componentwise* hsame *transport row for $P, I, B, E, A, O, C, R, K, N$.*

10. *K is the* Pkg *provenance over* $\mathsf{RealLogarithm}^{\uparrow}$, $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{OpenIntervalReal}^{\uparrow}$, $\mathsf{RealExponential}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$ *row.*

The classifier accepts only packets whose positive input, bracket ledger, exponential inverse-route rows, absolute-error controls, Bishop comparison rows, transport, continuation, provenance, and local naming are all displayed. It has no coordinate for a host logarithm or for a hidden infinite choice of bisection branch.

Lean4 ✓ `BEDC.Derived.RealLogarithmUp.RealLogarithmTasteGate single carrier alignment`

Theorem 2283.2 (RealLogarithm NameCert obligations)**.** *For every accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *carrier* $L = (P, I, B, E, A, O, C, R, H, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$ *surface consists exactly of carrier exposure for the positive real row, the open positive interval, the dyadic bracket ledger, the* $\mathsf{RealExponential}^{\uparrow}$ *inverse-route comparisons, the* $\mathsf{RealAbs}^{\uparrow}$ *error controls, the* $\mathsf{BishopRealComparison}^{\uparrow}$ *classifier rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Any public logarithm read must pass through those rows before it can consume an endpoint value.*

Proof. Project the accepted carrier to the displayed rows $P, I, B, E, A, O, C, R, H, K, N$. The positivity and interval rows give the only admissible domain data, while the dyadic ledger records the finite bracket observations that a consumer may inspect.

The inverse comparisons are then forced through E, A, O. The exponential row supplies the only comparison with the candidate endpoint, the absolute-value row supplies the displayed error control, and the Bishop comparison row classifies the retained side of the bracket.

Finally, H, C, K, N preserve the same rows under transport, replay, provenance, and naming. Since no coordinate stores a host logarithm, an analytic-continuation datum, or a selected infinite branch, the local NameCert surface is exactly the displayed inverse-route surface. □

Theorem 2283.3 (RealLogarithm exponential inverse route)**.** *Every public read of an accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *packet factors through the positive* $\mathsf{Real}^{\uparrow}$ *input row, the located open bracket row, the finite dyadic bracket ledger, the* $\mathsf{RealExponential}^{\uparrow}$ *comparison row, the* $\mathsf{RealAbs}^{\uparrow}$ *error-control row, and the* $\mathsf{BishopRealComparison}^{\uparrow}$ *classifier row before the logarithm endpoint is consumed. Consequently the packet names logarithm as the constructive inverse route of exponential on positive real data, not as a host transcendental primitive.*

Proof. Start with a public logarithm read. The carrier gives the positive input row P and the bracket row I, so the read begins inside a located positive real interval rather than in an ambient total real function space.

The finite bracket ledger B supplies the inspected dyadic intervals. Each retained bracket is justified only by the exponential comparison row E, the absolute-error row A, and the Bishop comparison row O.

The endpoint read is then routed through C and R, with H, K, N preserving the same displayed packet under transport, provenance, and naming. Thus the logarithm value is consumed only after the exponential inverse route has been displayed, and no hidden host log or selected infinite branch is exported. □

Theorem 2283.4 (RealLogarithm sibling dependency route)**.** *Every public logarithm read of an accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *packet first consumes the positive-domain row from chapter 2029, the bracket row from chapter 2133, the inverse comparison row from chapter 1816, and the error and*

classifier rows from chapter 1608 and chapter 1381. The route is a sibling dependency lattice for the displayed packet rows, not a host logarithm or ambient transcendental primitive.

Proof. Project the carrier $L = (P, I, B, E, A, O, C, R, H, K, N)$. The rows P and I are the positive-domain and open-bracket sources, so the logarithm read begins by consuming the sibling surfaces for located positive reals and open real intervals.

The inverse-route data then passes through E, A, O. The row E is the substantive $\mathsf{RealExponential}^{\uparrow}$ sibling dependency: it supplies the comparison route from chapter 1816, while the absolute-value and Bishop-comparison rows supply the error and classifier ledgers used by the finite bracket read.

Finally the structural rows C, R, H, K, N replay, transport, source, and name exactly that lattice of sibling rows. Since no carrier coordinate stores a host log or an ambient analytic primitive, the theorem records dependency routing rather than a host logarithm construction. □

Falsifiable prediction: Every $\mathsf{RealLogarithm}^{\uparrow}$ public read factors through a located positive real row, an open positive interval row, a dyadic bracket ledger, a RealExponential inverse-route row, a RealAbs error-control row, a BishopRealComparison classifier row, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: real-exponential, real-analytic, located-positive-real, open-interval-real, real-abs, bishop-real-comparison: $\mathsf{RealLogarithm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries positive-domain admission, finite inverse brackets, exponential comparison rows, absolute-error controls, Bishop comparison classification, endpoint transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLogarithm}^{\uparrow}$ packet is generated from a located positive real input, an open positive interval, finite dyadic bracket rows, RealExponential inverse-route comparisons, RealAbs error controls, BishopRealComparison classifier rows, componentwise transport, displayed continuation, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2315.1, Definition 2315.2, and Definition 2315.3 seed the constructive inverse-route surface for logarithm on positive real data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host logarithms, analytic continuation, quotient real equality, selected infinite bisection branches, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-domain exposure, bracket refinement, exponential inverse-route comparison, error-control, classifier nonescape, and TasteGate field faithfulness for BEDC.Derived.RealLogarithmUp.

Chapter 2284

A Concrete Naming Certificate for $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$

Following Bourbaki and Kelley on uniform completions and Bishop–Bridges on constructive completion, the $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ packet records the finite functorial handoff from a located uniform source to the regular Cauchy real seal. It sits after the located uniform completion surface of chapter 1692, the functorial completion boundary of chapter 1892, the regular rational readback of chapter 301, the finite observation windows of chapter 126, and the terminal real-name seal of chapter 38. The packet names a quotient-free L10 completion functor: it exposes source rows, located Cauchy rows, finite readback windows, real-seal handoff, transport, replay, provenance, and local naming, but no quotient completion, selected representative, countable-choice branch, host equality of limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2284.1 (Located Cauchy completion functor carrier)**.** *A* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, L, G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *source row, carrying the located uniform completion request before any real-facing consumer inspects a seal.*
2. *L is the located Cauchy service row extracted from U, recording the positive-radius and finite-window evidence used to answer completion requests.*
3. *G is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *action row, recording the finite functorial map from source requests to completion-facing requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window schedule through which the consumer asks for finite data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window schedule.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after U, L, G, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the located source, functor row, observation window, regular readback, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through*

$$U, L, G \longmapsto S, R, E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient Cauchy classes, representative selection, host limit equality, countable choice, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyCompletionFunctorUp.LocatedCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 2284.2 (Located Cauchy completion functor NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier* $F = (U, L, G, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletio}}$ *surface consists exactly of carrier admission, located uniform source exposure, located Cauchy service exposure, functorial completion action, finite observation-window routing, regular rational readback, terminal real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient completions, selected representatives, host limit equality, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2316.1. The displayed packet fixes the source row U, located service row L, functor row G, window schedule S, readback row R, terminal seal E, and structural rows H, C, P, N.

The located source is read first. The row U is the located uniform completion surface cited in chapter 1692, and L records the finite located Cauchy service used by the functor row. Thus no consumer reaches the completion functor before located evidence is displayed.

The functorial action then routes the request through G and into the finite observation rows. The schedule S and readback R are exactly the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ interfaces named above, so the handoff is a finite readback route rather than a quotient of all Cauchy names.

Finally E, H, C, P, N expose the real seal, transport the same components, replay the route, record provenance, and name the packet. Since the excluded completion principles are absent from the carrier, no local obligation can be stored outside the displayed rows. □

Theorem 2284.3 (Located Cauchy completion functor real-seal handoff)**.** *Every real-facing read of an accepted* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier factors through the route*

$\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, N

The terminal $\mathsf{Real}^{\uparrow}$ *seal is therefore available only after the located source row, functorial completion row, finite observation window, and regular rational readback have been displayed. The handoff exports no quotient stream equality, selected limit, countable-choice branch, host real equality, or ambient completeness datum.*

Proof. Start with an accepted carrier $F = (U, L, G, S, R, E, H, C, P, N)$. The continuation row C makes U, L, G the entry surface for every completion request, so the route cannot begin at a terminal real object.

The finite readback phase is then forced by S and R. These rows are the displayed $\mathsf{StreamName}^{\uparrow}$ observation window and $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the real seal E is stored only after them in the carrier.

Componentwise transport by H, replay by C, provenance by P, and naming by N preserve the same displayed route. Since no quotient stream equality, selected limit, choice branch, host real

equality, or ambient completeness coordinate occurs in the packet, every real-facing read factors through the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ completion read factors through $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locateduniformcompletion, uniformcompletionfunctor, regseqrat, streamname, real: $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds located source evidence, functorial completion action, finite observation windows, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2316.1, Definition 2316.2, and Definition 2316.3 seed the finite quotient-free completion-functor handoff through located uniform source rows, functor rows, finite observation windows, regular readback, and terminal real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, open-cover compactness, host limit equality, Lean verification, public bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, functorial action, finite-window readback, real-seal handoff, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.LocatedCauchyCompletionFunctorUp.

Constructive story: A $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ packet is generated from a located uniform completion source, a functorial completion row, finite observation windows, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2285

A Concrete Naming Certificate for $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$

The $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ packet realizes the wobble-quotient geometry branch around Definition 9462.1440 as finite trigger rows. It reads the quotient skeleton, spectral reconstruction, product decomposition, and occupancy rows of the wobble quotient geometry file, then packages them as a concrete NameCert surface beside the hash boundary of chapter 242, the spectral sequence surface of chapter 178, the network-flow route of chapter 233, and the Boolean endpoint discipline of chapter 32. The packet is a finite trigger carrier for spectral geometry consumers; it is not a host eigenvector computation, a biological assertion, a quotient-space semantic equality, or an unbounded graph search.

Falsifiable prediction: Every accepted $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ trigger read factors through wobble quotient rows, spectral hinge rows, trigger energy rows, hsame transport, Cont replay, and Pkg provenance before any downstream consumer treats the trigger as active.

Independence witness: hash, spectralseq, networkflow, bool: $\mathsf{Hash}^{\uparrow}$ records digest rows rather than quotient geometry, $\mathsf{SpectralSeq}^{\uparrow}$ records filtration pages rather than wobble trigger energy, $\mathsf{NetworkFlow}^{\uparrow}$ records capacity routes rather than quotient-fiber hinges, and $\mathsf{Bool}^{\uparrow}$ records endpoint tags rather than the spectral trigger packet; none is bijective to $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$.

Definition 2285.1 (Wobble quotient spectral trigger carrier)**.** *A* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, S, E, H, C, P, B, R, N)$$

with the following displayed rows:

1. *Q is the wobble quotient row, recording the finite f_3 quotient vertices, the one-hole quotient closure, and the quotient-product lift surface.*
2. *S is the spectral hinge row, recording the finite adjacency spectrum readback used by the quotient spectral reconstruction.*
3. *E is the trigger energy row, recording the accepted threshold comparison for the killed bit-flip spectral radius and its finite trigger tag.*
4. *H records componentwise* hsame *transport for quotient, spectral, energy, continuation, provenance, Boolean tag, route, and local naming rows.*

5. *C records displayed* Cont *replay from a consumer trigger request through*

$$Q, S, E \longmapsto H, P, B, R, N.$$

6. *P is the* Pkg *provenance over* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, $\mathsf{Hash}^{\uparrow}$, $\mathsf{SpectralSeq}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

7. *B is the* $\mathsf{Bool}^{\uparrow}$ *trigger tag row, exposing the finite accept or refuse endpoint for the trigger request.*

8. *R is the route row assigning the read to quotient-geometry support, spectral-hinge support, trigger-energy support, or refusal.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for host spectral computation, ambient quotient equality, biological interpretation, infinite graph traversal, or a trigger that bypasses route classification.

Theorem 2285.2 (Wobble quotient spectral trigger NameCert obligations)**.** *For every accepted* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *carrier* $T = (Q, S, E, H, C, P, B, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$ *surface consists exactly of carrier admission, quotient-row exposure, spectral-hinge exposure, trigger-energy exposure, route classification, Boolean trigger tagging, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of host spectral computation, ambient quotient equality, biological interpretation, infinite graph traversal, and unclassified trigger reads.*

Proof. Carrier admission projects the displayed tuple of Definition 2317.1. The quotient row Q, spectral row S, trigger energy row E, route row R, Boolean row B, and structural rows H, C, P, N are the only visible coordinates.

The quotient row is read before the spectral hinge. This keeps the trigger tied to the finite wobble quotient skeleton and prevents a consumer from substituting an ambient quotient-space equality.

The spectral hinge and trigger energy rows are then read through S and E. Route classification by R decides whether the request is quotient support, spectral support, energy support, or refusal before the Boolean tag B is exported.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no host spectral computation, biological interpretation, infinite graph traversal, or unclassified trigger coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2285.3 (Wobble quotient spectral trigger handoff)**.** *Every accepted trigger read of a* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *packet factors through the route*

$$\mathsf{Hash}^{\uparrow},\ \mathsf{SpectralSeq}^{\uparrow},\ \mathsf{NetworkFlow}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The quotient row, spectral hinge row, trigger energy row, route row, and Boolean trigger tag are therefore visible before a downstream consumer can treat the trigger as accepted. The route exports no host eigenvector data, ambient quotient equality, biological assertion, infinite graph search, or trigger tag outside the displayed carrier.

Proof. Start with an accepted carrier $T = (Q, S, E, H, C, P, B, R, N)$. The quotient, spectral, and energy rows are the only support rows available to the trigger request, while R records how the request is classified.

The Boolean tag is read only after route classification. Thus an accepted tag in B cannot be separated from the quotient, spectral, and energy evidence that produced it.

Componentwise transport by H, continuation replay by C, provenance by P, and local naming by N preserve that same finite route. Since the excluded host and ambient objects are absent from the carrier, every accepted trigger read factors through the displayed rows. □

Theorem 2285.4 (Wobble quotient spectral trigger TasteGate obligations)**.** *The TasteGate surface for* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *requires a nontriviality row and a field-faithfulness row:*

$$\mathsf{Nontrivial}\,\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow} \quad \textit{and} \quad \mathsf{FieldFaithful}\,\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}.$$

The nontriviality row separates accepted trigger tags from refusal tags, and the field-faithfulness row requires every accepted tag to retain its quotient-row, spectral-hinge, trigger-energy, route, transport, replay, provenance, and local naming support.

Lean4 ✓ `BEDC.Derived.WobbleQuotientSpectralTriggerUp.wobbleQuotientSpectralTriggerChapterTasteGate`

Proof. Nontriviality is read from the Boolean trigger row B together with the route classifier R. An accepted tag and a refusal tag are distinct only because the carrier displays both the tag and the route that licensed it.

Field faithfulness is row preservation. A field-faithful read may transport the packet through H and replay it through C, but it cannot erase Q, S, E, R, P, N while keeping an accepted trigger tag.

Thus the required TasteGate rows are exactly the nontrivial trigger distinction and the preservation of the displayed support fields. □

Closure status for $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2317.1, Definition 2317.2, Definition 2317.3, and Definition 2317.4 seed the finite quotient-trigger surface through quotient rows, spectral hinge rows, trigger energy rows, route classification, Boolean trigger tags, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host spectral computation, biological semantics, ambient quotient equality, infinite graph traversal, Lean verification, public bridge checking, or field-faithful TasteGate implementation.

Upgrade path: Obligation closure requires checked carrier admission, quotient exposure, spectral hinge exposure, trigger-energy exposure, route classification, Boolean tagging, support-field preservation, and TasteGate instances for BEDC.Derived.WobbleQuotientSpectralTriggerUp.

Constructive story: A $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ packet is generated from wobble quotient rows, spectral hinge rows, trigger energy rows, Boolean trigger tags, route classification, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2286

A Concrete Naming Certificate for $\mathsf{CantorIntersectionTheorem}^{\uparrow}$

Following Cantor's nested intersection theorem and Bishop–Bridges constructive analysis, the $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ packet records the theorem-shaped L10 boundary where a displayed sequence of nested located intervals is consumed through finite windows before any Real-facing seal is read. It sits downstream of the finite compactness surface of chapter 1584, the nested carrier of chapter 320, the located endpoint row of chapter 394, the dyadic interval ledger of chapter 306, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter names the finite nested-intersection theorem route only; it does not introduce a point selector, quotient real equality, open-cover compactness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2286.1 (Cantor intersection theorem carrier)**.** *A* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *finite-window compactness row consumed as the theorem source.*
2. *I is the* $\mathsf{NestedInterval}^{\uparrow}$ *row naming the inspected nested interval prefix.*
3. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-location row for the same displayed prefix.*
4. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *row carrying dyadic endpoint boxes, inclusion checks, and mesh comparisons.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule read at the displayed precisions.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the located dyadic windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, I, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the theorem source, interval rows, windows, readback, seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite theorem request through Q, I, L, D, W, R, E in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$ *row.*

The classifier admits only packets with these eleven rows. No coordinate stores open-cover compactness, a selected intersection point, quotient real equality, countable choice, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 2286.2 (Cantor intersection theorem NameCert obligations)**.** *For every accepted* CantorIntersectionTheor *carrier $K = (Q, I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed nested-interval compactness row, nested prefix, located endpoints, dyadic interval ledger, stream windows, regular rational readback, real seal, transports, continuations, provenance, and local naming;*
2. *source coherence, requiring Q to be consumed through the same I, L, D, W, R, E route exposed by the finite-window compactness packet;*
3. *located mesh exactness, requiring L and D to govern the same interval prefix before any* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *row is read;*
4. *theorem-route sealing, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through Q, I, L, D, W, R before E, H, C, P, N are exposed;*
5. *ledger non-escape, excluding open-cover compactness, selected intersection points, quotient real equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 2318.1. The local naming surface has exactly the displayed rows $Q, I, L, D, W, R, E, H, C, P, N$.

The source coherence row is inherited by reading Q through chapter 1584. That source already exposes the nested prefix, located endpoint row, dyadic interval row, stream windows, regular rational readback, and real seal boundary as finite data.

The located mesh exactness row binds L and D to the same prefix before W and R can be consumed. The theorem-route sealing row then forces any Real-facing read to pass through the finite source and mesh route before it reaches E and the structural rows H, C, P, N.

Since no row in the carrier is an open-cover compactness proof, point selector, quotient-real equality witness, countable-choice schedule, trichotomy row, or ambient completeness theorem, the ledger non-escape obligation is exactly the absence of those coordinates. □

Theorem 2286.3 (Cantor intersection theorem nested route)**.** *Let $K = (Q, I, L, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier. Every theorem-facing read of the nested intersection boundary factors through*

$$\mathsf{NestedIntervalCompactness}^{\uparrow},\ \mathsf{NestedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in that order, followed only by componentwise transport, continuation replay, package provenance, and the local naming row.

Lean4 ✓ `BEDC.Derived.CantorIntersectionTheoremUp.CantorIntersectionTheoremTasteGate single carrier alignment`

Proof. Start with a theorem-facing read. By Definition 2318.2, the read first consumes the finite-window source row Q and its exposed nested prefix I.

Next the same obligation surface requires located mesh exactness. The located endpoint row L and dyadic interval row D therefore govern the inspected prefix before the stream windows W or regular rational readback R are visible.

Finally the theorem-route sealing row sends the read through R to the real seal E, and only then to H, C, P, N. The carrier has no other coordinate, so the displayed seven-object route exhausts the nested-intersection theorem boundary. □

Theorem 2286.4 (Cantor intersection theorem Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *packet receives a finite nested-window theorem route through* $Q, I, L, D, W, R, E, H, C, P, N$*. The consumer receives no open-cover compactness theorem, no selected intersection point, no quotient real equality principle, no countable-choice selector, no trichotomy row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Apply Definition 2318.3. A Real-facing consumer reaches the terminal seal only after the finite-window compactness row, nested prefix, located endpoints, dyadic boxes, stream windows, and regular rational readback have all been displayed.

The structural rows H, C, P, N preserve and name that route; they do not add analytic content. The source coherence and theorem-route sealing obligations of Definition 2318.2 keep the Real seal downstream of the same finite data.

Thus the consumer has no coordinate from which to read open-cover compactness, a selected point of the full intersection, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The exported boundary is the finite nested-window theorem route and nothing stronger. □

Theorem 2286.5 (CantorIntersection diameter-zero route)**.** *Let* $K = (Q, I, L, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier. If the nested interval packet carries a displayed dyadic diameter ledger tending to zero, then every Real-facing seal read factors through* $\mathsf{NestedIntervalCompactness}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *by first reading the nested prefix* I*, located endpoint row* L*, dyadic interval and diameter rows in* D*, stream windows* W*, regular rational readback* R*, and terminal seal* E*. The route exports no selected intersection point, ambient completeness claim, quotient real equality, countable choice, trichotomy, or open-cover compactness.*

Proof. Project the carrier of Definition 2318.1. The theorem source row Q exposes the same nested interval prefix I that the diameter-zero read inspects.

Source coherence from Definition 2318.2 forces the nested prefix to be read before the Real-facing seal route is opened. The displayed dyadic diameter ledger is part of the D row, so it is tied to the same prefix and the same located endpoint row L.

Located mesh exactness then governs the zero-diameter rows. A consumer may inspect the finite $\mathsf{StreamName}^{\uparrow}$ windows W and the $\mathsf{RegSeqRat}^{\uparrow}$ readback R only after the located and dyadic rows have displayed the required mesh comparisons.

Finally, theorem-route sealing and Definition 2318.4 send the read through E and then only through H, C, P, N. None of those structural rows can add a selected point, ambient completeness principle, quotient equality, countable selector, trichotomy row, or open-cover compactness theorem. □

Falsifiable prediction: An accepted $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without first reading the nested-interval compactness row, nested prefix, located endpoint row, dyadic interval ledger, stream windows, and regular rational readback refutes the carrier surface.

Independence witness: nestedintervalcompactness, nestedinterval, locatedinterval, dyadicinterval, streamname, regseqrat, real: none is bijective to $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ because this packet jointly carries the theorem-source row, the finite nested prefix, located and dyadic endpoint rows, stream windows, regular readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ packet is generated from a NestedIntervalCompactness row, a nested interval prefix, located endpoint rows, dyadic interval boxes, StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2318.1, Definition 2318.2, Definition 2318.3, and Definition 2318.4 seed the finite Cantor nested-intersection theorem route over the L10 nested-interval boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected intersection points, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove source coherence, located mesh exactness, theorem-route sealing, and ledger non-escape for the same finite packet.

Chapter 2287

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$

Following Bishop–Bridges and Bourbaki-style completion theory, the $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ packet records the finite universal-property surface for the L10 completion interface. It sits after the metric-completion carrier of chapter 1506, the uniform-completion route of chapter 1473, the functorial completion handoff of chapter 423, and the separated-metric comparison boundary of chapter 1411. The packet names the dense-source, extension, and uniqueness rows that completion consumers may cite without importing quotients, selected limits, choice, or ambient completeness.

Definition 2287.1 (Cauchy completion universal property carrier)**.** *A* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, K, T, F, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source row, recording the source object together with the finite density witness consumed by completion readers.*
2. *K is the completion carrier row, read through* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff data.*
3. *T is the complete separated target row, recording the target surface whose separatedness supports uniqueness.*
4. *F is the source map row from D into T before completion is invoked.*
5. *E is the extension map row from K into T, tied to F along the dense source row.*
6. *L is the uniqueness classifier, comparing two extension rows over the same D, K, T, F surface by separatedness rather than by quotient equality.*
7. *H records componentwise* hsame *transport for D, K, T, F, E, L and the local naming row.*
8. *C records displayed* Cont *replay of dense-source handoff, extension existence, and uniqueness reads.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, K, T, F, E, L, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host limit choice, ambient real completeness, or countable choice.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalPropertyUp.CauchyCompletionUniversalPropertyTasteGate single carrier alignment`

Theorem 2287.2 (CauchyCompletionUniversalProperty NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *carrier* $U = (D, K, T, F, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of five obligation rows: carrier exposure for the displayed tuple; dense-source handoff through D and the source map F; extension existence through the completion rows K, T, E; uniqueness up to the classifier L over the same dense-source and completion route; and ledger non-escape from choice, quotient equality, selected limits, or host limit principles.*

Proof. Carrier exposure is projection from Definition 2319.1. The tuple displays the dense source, completion carrier, complete separated target, source map, extension row, uniqueness classifier, transport, replay, provenance, and local naming row.

The dense-source handoff uses D and F. A consumer must first read the source and its finite density route before it can ask for an extension into the target.

Extension existence is carried by K, T, E. The completion carrier is read through the metric, uniform, and functorial completion rows, while T supplies the complete separated target surface. The extension row E is therefore attached to displayed completion data rather than to a selected host limit.

Uniqueness is carried by L. Since L compares extension rows over the same D, K, T, F surface and uses separatedness from T, it does not require quotient equality or representative selection.

The ledger boundary is fixed by H, C, P, N. These rows transport, replay, source, and name the same displayed packet, and no excluded host-side principle appears as a coordinate of the carrier. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalPropertyUp.CauchyCompletionUniversalPropertyCarrier namecert obligations`

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence or uniqueness while omitting the dense source, completion carrier, complete separated target, source map, extension row, uniqueness classifier, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: metriccompletion, uniformcompletion, cauchycompletionfunctor, separatedmetric: none is bijective because $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ jointly carries dense source, completion carrier, complete separated target, source map, extension, uniqueness classifier, transport, replay, provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ packet is generated from a dense source row, a completion carrier row, a complete separated target row, a source map, an extension row, a uniqueness classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$, seed)

Scope closed: Definition 2319.1 and Definition 2319.2 seed the finite L10 Cauchy completion universal-property surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, dense-source handoff, extension existence, uniqueness classification, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyCompletionUniver

Chapter 2288

A Concrete Naming Certificate for $\mathsf{CandidateSetNormalization}^{\uparrow}$

The $\mathsf{CandidateSetNormalization}^{\uparrow}$ packet realizes the candidate-set reading of chapter 9383 as finite NameCert data. It records candidate evidence, closed-term projection, Pi and application adequacy, no-infinite-reduction, obstruction, transport, continuation, provenance, and local naming rows. The packet is deliberately a candidate-set surface: it does not claim unconditional strong normalization for every closed typed MetaCIC term. Its candidate-evidence and bounded-normal checking route is read through the sibling surfaces of chapter 892 and chapter 752.

Definition 2288.1 (MetaCIC candidate-set normalization carrier)**.** *A* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, A, I, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite candidate-set evidence row, including candidate membership, soundness, and beta-closure evidence for the accepted read.*
2. *T is the closed typed term projection row, recording the empty-context typed term whose normalization read is supplied by K.*
3. *A is the application and Pi adequacy row, tying application reads and binder reads back to the same candidate-set evidence.*
4. *I is the no-infinite-reduction row obtained from the candidate-set strong-normalization route.*
5. *O is the obstruction row recording that a closed-term normalization read is refused when K is absent.*
6. *H records componentwise* hsame *transport for K, T, A, I, O and the local naming row.*
7. *C records displayed* Cont *replay from candidate-set evidence to strong normalization, closed-term projection, application and Pi adequacy, and no-infinite-reduction reads.*
8. *P is the* Pkg *provenance over* $\mathsf{CandidateSetNormalization}^{\uparrow}$, $\mathsf{MetaCICRedexFrontier}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontie}$ $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, T, A, I, O, H, C, P, N$. It contains no coordinate for evidence-free closed-term normalization, componentwise application normalization, host reduction normalization, quotient reduction equality, subject-reduction discharge, or a global normalization oracle.

Lean4 (definition) `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationUp`
Lean4 ✓ `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationTasteGate single carrier alignment`

Definition 2288.2 (Candidate-set normalization frontier filter)**.** *For an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier $S = (K, T, A, I, O, H, C, P, N)$, its public frontier filter is the finite* BHist *subpacket*

$$\mathsf{Frontier}(S) = (K, T, A, I, O, H, C, P, N)$$

read in the displayed order: candidate-set evidence K, closed-term projection T, adequacy row A, no-infinite-reduction row I, obstruction row O, transport H, replay C, provenance P, and local naming N. A consumer reaches the redex-facing normalization frontier only through this subpacket, so the filter contains no coordinate for candidate-free strong normalization, closed subject reduction, arbitrary confluence, host conversion equality, or a global normalizer.

Theorem 2288.3 (MetaCIC candidate-set normalization obligations)**.** *For every accepted* $\mathsf{CandidateSetNormalizati}$ *carrier $S = (K, T, A, I, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$ *surface consists exactly of finite candidate-set evidence through K; closed-term projection through T; application and Pi adequacy through A; no-infinite-reduction through I; obstruction when the candidate-set row is absent through O; componentwise transport through H; displayed replay through C; provenance through P; local naming through N; and non-escape from unconditional MetaCIC strong normalization.*

Proof. Carrier admission is projection from Definition 2320.1. The tuple displays the candidate-set evidence, closed typed term projection, adequacy row, no-infinite-reduction row, obstruction row, transport, replay, provenance, and local name.

The positive route begins at K. Candidate-set membership, soundness, and beta-closure are read before T, A, or I can export a normalization-facing conclusion.

The closed-term projection is T, and the application and Pi adequacy route is A. Both rows consume the same candidate-set evidence rather than a componentwise strong-normalization shortcut.

The no-infinite-reduction row I is downstream of that evidence. If K is absent, O records the obstruction, so the packet refuses an evidence-free theorem over all closed typed terms.

Finally H, C, P, N transport, replay, source, and name the same finite branch. Since no global normalization oracle or quotient reduction equality is present in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2288.4 (Candidate-set normalization redex non-escape)**.** *Every redex-frontier consumer of an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier must pass through the frontier filter $K, T, A, I, O, H, C, P, N$ of Definition 2320.2. It exports no candidate-free strong normalization, no closed subject-reduction theorem, and no arbitrary confluence principle.*

Proof. Project the accepted carrier and then read its frontier filter. The first row is K, so the consumer starts from candidate-set evidence rather than from a redex-normalization oracle.

The closed-term row T and adequacy row A are downstream of the same candidate evidence. They can expose closed projection and application or Pi adequacy only through that evidence.

The no-infinite-reduction row I and obstruction row O complete the normalization-facing branch. If the candidate evidence is absent, O records refusal instead of supplying a candidate-free theorem.

Finally H, C, P, N transport, replay, source, and name the same branch. Since the filter has no subject-reduction, confluence, or global-normalizer coordinate, no consumer can export those conclusions from this packet. □

Theorem 2288.5 (Candidate-set normalization candidate-closed handoff)**.** *For every accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier, the handoff from candidate-set evidence to the closed-term projection remains mediated by* $K, T, A, I, O, H, C, P, N$*. In particular, the closed projection row* T *is not a candidate-free closed normalization theorem and cannot be reused without the candidate-set evidence row* K*.*

Proof. Read the carrier as $S = (K, T, A, I, O, H, C, P, N)$. The closed-term projection row is named as T, but its admissibility is part of the same finite packet as K.

The adequacy and no-infinite-reduction rows A and I also consume the candidate-set route. They do not create a detached closed-term theorem, because the obstruction row O records refusal when K is absent.

Transport and replay through H and C, with provenance and naming through P and N, preserve the same candidate-mediated route. Thus the closed projection remains a handoff inside the frontier filter rather than a standalone normalization principle. □

Theorem 2288.6 (MetaCIC candidate-set normalization idempotence)**.** *Let* $S = (K, T, A, I, O, H, C, P, N)$ *be an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier whose candidate-set packet has already been normalized through the displayed frontier-dependency rows. Applying the same normalization route again exposes the same accepted candidate rows, the same closed-term projection, the same application and Pi adequacy rows, and no additional spectral-trigger row.*

Proof. Use the carrier and obligation surface of Definition 2320.1 and Definition 2320.3. The first normalization pass can read only $K, T, A, I, O, H, C, P, N$, and the accepted candidate-set evidence is already attached to the displayed no-infinite-reduction and obstruction rows.

The frontier-dependency rows of chapter 2307 determine which candidate rows may be exposed before the closed-term projection and adequacy rows are replayed. Running the same route again therefore reuses the same dependency rows rather than introducing a new candidate branch.

Finally compare with the spectral-trigger surface of chapter 2317. A trigger row would have to be present as an additional obstruction or replay coordinate, but the $\mathsf{CandidateSetNormalization}^{\uparrow}$ classifier contains no such coordinate beyond O and C. The second pass is therefore idempotent on accepted candidate rows and creates no further spectral-trigger row. □

Falsifiable prediction: No accepted closed-term strong-normalization read through $\mathsf{CandidateSetNormalization}^{\uparrow}$ bypasses candidate evidence or candidate-set evidence.

Independence witness: metacicredexfrontier, metacicnormalizationfrontier, bounded-normal-equality-checker, metacicopenproblemledger: none is bijective because $\mathsf{CandidateSetNormalization}^{\uparrow}$ jointly carries candidate-set evidence, closed-term projection, application and Pi adequacy, no-infinite-reduction, obstruction, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CandidateSetNormalization}^{\uparrow}$ packet is generated from finite candidate-set evidence, a closed typed term projection, application and Pi adequacy, a no-infinite-reduction row, an obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CandidateSetNormalization}\uparrow}$, seed)
Scope closed: Definition 2320.1 and Definition 2320.3 seed the finite candidate-set normalization surface.
Formal verification: FormalStatus(NameCert$_{\text{CandidateSetNormalization}\uparrow}$, encodedDef)
Lean target: `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationUp`
Bridge status: none
Not claimed: This chapter does not claim unconditional MetaCIC strong normalization, evidence-free application normalization, subject reduction, host normalization, quotient reduction equality, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked candidate-set evidence, closed-term projection, application and Pi adequacy, no-infinite-reduction, obstruction exactness, transport replay, and TasteGate field faithfulness for BEDC.Derived.CandidateSetNormalizationUp.

Chapter 2289

A Concrete Naming Certificate for $\mathsf{CompletionComparison}^{\uparrow}$

Following Bishop–Bridges metric completion and Kelley uniform-space completion, the $\mathsf{CompletionComparison}^{\uparrow}$ packet records the finite comparison surface shared by metric, filter, net, regular-Cauchy, separated, and uniform completion presentations. It sits after the metric-completion carrier of chapter 1506, the Cauchy-filter carrier of chapter 333, the Cauchy-net carrier of chapter 589, the regular-Cauchy comparison carrier of Definition 411.1, the separated-metric carrier of chapter 1411, and the uniform-completion carrier of chapter 1473. The chapter gives downstream completion consumers one named comparison surface rather than requiring them to choose a quotient-completion presentation or a selected representative prose route.

Definition 2289.1 (Completion comparison carrier)**.** *A* $\mathsf{CompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, T, R, S, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, carrying the metric completion presentation before any comparison consumer reads a completion point.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, carrying the filter presentation over the same finite tolerance requests.*
3. *T is the* $\mathsf{CauchyNet}^{\uparrow}$ *source row, carrying the net presentation and its directed observation rows.*
4. *R is the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row, comparing the sequence-style regular Cauchy route with the filter and net routes through displayed tolerance windows.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row used to make equality and uniqueness reads pass through separated comparison data rather than host equality.*
6. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row, recording the uniform completion presentation consumed by entourage-facing readers.*
7. *H records componentwise* hsame *transport for the metric, filter, net, regular-comparison, separated, uniform, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the comparison route*

$$M, F, T, R \longmapsto S, U \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionComparison}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, choice of a convergent subnet, host equality of limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionComparisonUp.CompletionComparisonTasteGate single carrier alignment`

Theorem 2289.2 (Completion comparison NameCert obligations)**.** *For every accepted* $\mathsf{CompletionComparison}^{\uparrow}$ *carrier* $K = (M, F, T, R, S, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission, metric completion source exposure, Cauchy-filter source exposure, Cauchy-net source exposure, regular-Cauchy comparison exposure, separated-metric equality routing, uniform-completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient completion equality, selected representatives, choice-selected subnets, host equality of limits, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2321.1. The accepted packet exposes exactly the metric row M, filter row F, net row T, regular comparison row R, separated row S, uniform row U, and structural rows H, C, P, N.

The source presentations are read before any equality or uniqueness comparison. Rows M, F, T, R keep metric, filter, net, and regular-Cauchy evidence inside the same finite packet, so a consumer cannot replace the displayed comparison by a quotient object or a selected representative.

The separated and uniform rows then control the target-facing read. Row S supplies the separated comparison boundary, and row U supplies the uniform completion handoff. Rows H, C, P, N transport, replay, source, and name the same finite route.

Since the excluded host-side principles are absent from the carrier, every local obligation is one of the displayed rows. □

Theorem 2289.3 (Completion comparison filter-net correspondence)**.** *Every filter-net completion read of an accepted* $\mathsf{CompletionComparison}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow},\ \mathsf{RegularCauchyComparison}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow}$$

In particular, a consumer comparing Cauchy filters with Cauchy nets must read the shared metric row, the regular-Cauchy comparison row, the separated-metric row, and the uniform-completion handoff from Definition 2321.1; it cannot import quotient equality of completions, a selected representative sequence, a choice-selected subnet, a host limit theorem, or ambient completeness.

Proof. Begin with an accepted carrier $K = (M, F, T, R, S, U, H, C, P, N)$. The filter row F and net row T occur in the same packet as the metric row M, so their comparison begins from displayed finite completion data.

The regular comparison row R supplies the common tolerance-window route. The separated row S is the only equality-facing component, and the uniform row U is the only entourage-facing completion handoff.

Finally H, C, P, N preserve that route under transport, replay, provenance, and local naming. No extra completion object appears in K, so every filter-net read factors through the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{CompletionComparison}^{\uparrow}$ consumer comparing completion presentations factors through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metriccompletion, cauchyfilter, cauchynet, regularcauchycomparison, separatedmetric, uniformcompletion: $\mathsf{CompletionComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric, filter, net, regular-Cauchy, separated, uniform, transport, replay, provenance, and local NameCert rows in one comparison packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2321.1, Definition 2321.2, and Definition 2321.3 seed the finite comparison surface tying metric, filter, net, regular-Cauchy, separated, and uniform completion rows to one NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, choice-selected subnets, host limit equality, open-cover compactness, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-row exposure, filter-net correspondence, separated equality routing, uniform handoff, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CompletionComparisonUp.

Constructive story: A $\mathsf{CompletionComparison}^{\uparrow}$ packet is generated from metric completion, Cauchy-filter, Cauchy-net, regular-Cauchy comparison, separated-metric, uniform-completion, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2290

A Concrete Naming Certificate for $\mathsf{CofinalCompactInterval}^{\uparrow}$

Following Bishop and Bridges constructive compact-interval arguments, the $\mathsf{CofinalCompactInterval}^{\uparrow}$ packet names the finite cofinal window by which a closed bounded interval request is refined to displayed locator, dyadic, stream, regular-rational, and Real-facing rows. It sits after the closed bounded interval source of chapter 1591, the compact interval locator of chapter 1891, the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives downstream interval compactness, Cousin-style finite refinement, and Heine–Borel interval consumers a named finite-window route without importing open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2290.1 (Cofinal compact interval carrier)**.** *A* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing lower and upper located endpoints, endpoint order, rational containment, dyadic containment, and the base interval naming surface.*
2. *L is the* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *row, recording the located comparison and tolerance-to-window service used to refine interval requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying the dyadic tolerances, endpoint cells, and finite mesh comparisons demanded by I and L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the cofinal dyadic depth chosen for the displayed request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the same interval, locator, dyadic, and stream rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the interval source, locator, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-interval request through $I, L, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CofinalCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{CompactIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$ *row.*

The classifier admits only packets with these ten rows. No coordinate stores open-cover compactness, a selected subcover, quotient real equality, countable choice, trichotomy, host interval equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CofinalCompactIntervalUp.TasteGate.CofinalCompactIntervalTasteGate single carrier alignment`

Theorem 2290.2 (Cofinal compact interval NameCert obligations)**.** *For every accepted* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *carrier* $K = (I, L, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, compact locator, dyadic tolerance ledger, stream windows, regular rational readback, real seal, transports, continuations, provenance, and local naming;*
2. *interval-source exactness, requiring every endpoint, containment, and interval-local request to enter through I before the locator row is read;*
3. *locator cofinality, requiring L and D to convert any displayed compact-interval request into a finite dyadic window deep enough for the requested observation;*
4. *real-handoff ordering, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through I, L, D, W, R before E, H, C, P, N are exposed;*
5. *ledger non-escape, excluding open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, host interval equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 2322.1. The local naming surface has exactly the displayed rows $I, L, D, W, R, E, H, C, P, N$.

The interval-source exactness row is supplied by reading I through chapter 1591. It exposes the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, finite-window, regular-readback, and Real-seal boundary as displayed interval data.

The locator cofinality row is then read through chapter 1891 and the dyadic tolerance ledger of chapter 291. These rows force the selected finite window to be a displayed refinement of the same interval request before W or R can be consumed.

Finally the real-handoff ordering row sends every Real-facing read through W and R before E and the structural rows H, C, P, N are exposed. Since no coordinate of the carrier is an open cover, selected subcover, quotient-real equality witness, countable-choice schedule, trichotomy row, host interval equality, or ambient completeness theorem, the ledger non-escape obligation is exactly the absence of those rows. □

Theorem 2290.3 (Cofinal compact interval Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, interval-compactness, Cousin-refinement, or Heine–Borel-style consumer admitted by an accepted* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *packet reads the compact interval request through*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{CompactIntervalLocator}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in that order. The handoff exposes a finite cofinal window and its real seal; it exposes no open-cover compactness row, selected subcover, quotient real equality, countable-choice selector, trichotomy decision, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an admitted consumer. By Definition 2322.2, the read first enters through the closed bounded interval row I, so the endpoint and containment data are fixed before any compact-window refinement is available.

Next the same obligation surface requires locator cofinality. The locator row L and dyadic ledger D therefore choose a displayed finite dyadic window for the same request before the stream row W and regular readback R are visible.

The real-handoff ordering row then sends the read through R to the real seal E, followed only by transport, replay, provenance, and the local naming row. Because the carrier has no other coordinate, the displayed six-object route exhausts the consumer-facing handoff and excludes the listed compactness, choice, quotient, order, host, and completeness data. □

Falsifiable prediction: An accepted $\mathsf{CofinalCompactInterval}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without first reading the closed bounded interval row, compact interval locator, dyadic tolerance ledger, finite stream window, and regular rational readback refutes the carrier surface.

Independence witness: closedboundedinterval, compact-interval-locator, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{CofinalCompactInterval}^{\uparrow}$ because this packet jointly carries the interval source, compact locator, cofinal dyadic window, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalCompactInterval}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a CompactIntervalLocatorUp row, a DyadicRatCoreUp tolerance ledger, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2322.1, Definition 2322.2, and Definition 2322.3 seed the finite cofinal compact-interval route from closed bounded intervals through locator and dyadic-window rows to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, host interval equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exactness, locator cofinality, finite-window routing, regular readback, Real-seal handoff, structural transport, replay, provenance, and local naming for BEDC.Derived.CofinalCompactIntervalUp.

Chapter 2291

A Concrete Naming Certificate for InverseMappingTheorem$^{\uparrow}$

Following Banach and Rudin functional analysis, the InverseMappingTheorem$^{\uparrow}$ packet records the finite BEDC certificate for reading a bijective bounded linear operator between Banach rows as a bounded inverse row. It consumes the Banach source and target certificates of chapter 90, the bounded-operator surface of chapter 1644, the open-ball image route of chapter 1538, and the graph-read boundary of chapter 1537. The packet exposes only displayed rows: Banach endpoints, a bijective bounded-linear-operator row, an open-mapping image witness, an inverse-window ledger, transport, replay, provenance, and local naming.

Definition 2291.1 (Inverse mapping theorem carrier)**.** *An* InverseMappingTheorem$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, T, B, I, H, C, P, N)$$

with the following visible rows:

1. *X is the source* Banach$^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, and completion ledgers used by source-window consumers.*
2. *Y is the target* Banach$^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, and completion ledgers used by target-window consumers.*
3. *T is the* BoundedLinearOperator$^{\uparrow}$ *row from X to Y, carrying the endpoint action, linearity ledger, norm-bound endpoint, and bounded-operator provenance over the same Banach endpoints.*
4. *B is the bijectivity ledger. Its injective side records that equal target observations of T factor through the source classifier of X, and its surjective side records that every displayed target window has a scheduled source preimage window.*
5. *I is the* OpenMappingTheorem$^{\uparrow}$ *image-row witness. It records the finite open-ball image data and the bounded-inverse-facing source-window handoff supplied by the open-mapping packet for the same X, Y, T, B rows.*
6. *H records the inverse-window ledger: target tolerance requests, source preimage windows, norm-bound comparisons for the inverse row, and the finite dependencies on B and I.*
7. *C records componentwise* hsame *transport for X, Y, T, B, I, H and the structural rows.*

8. *P records displayed* Cont *replay from a bounded-inverse request through the Banach rows, the bounded operator row, the bijectivity ledger, the open-mapping image witness, the inverse-window ledger, and the closed-graph boundary.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InverseMappingTheorem}^{\uparrow}}$ *row, with provenance over* $\mathsf{InverseMappingTheorem}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OpenMappingTheorem}^{\uparrow}$, $\mathsf{ClosedGraphTheorem}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

The carrier has no coordinate for a host inverse function, quotient operator space, selected inverse values outside the displayed windows, or completeness beyond the Banach source and target rows.

Theorem 2291.2 (Inverse mapping theorem NameCert obligations)**.** *For an accepted* $\mathsf{InverseMappingTheorem}^{\uparrow}$ *carrier* $Q = (X, Y, T, B, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{InverseMappingTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Banach source row, Banach target row, bounded-linear-operator row, bijectivity ledger, open-mapping image-row witness, inverse-window ledger, componentwise transport, replay, provenance, and local naming;*

2. *endpoint agreement, requiring T, B, and I to read the same displayed X and Y Banach rows before any inverse-window request is admitted;*

3. *bijective operator exposure, requiring inverse-facing consumers to read injective comparison and surjective preimage data through the bounded-operator row T and the finite ledger B;*

4. *open-mapping handoff, requiring every bounded-inverse norm estimate to pass through the* OpenMappingTheorem *image-row witness I before the inverse-window ledger H is public;*

5. *closed-graph compatibility, requiring graph-facing consumers to route the displayed inverse graph through chapter 1537 and the same X, Y, T, B, I, H rows;*

6. *structural replay, requiring* hsame, Cont, Pkg, *and local* NameCert *rows to preserve the same bounded-inverse route;*

7. *ledger non-escape, excluding host inverse functions, quotient operator spaces, selected inverse values outside the displayed windows, and ambient completeness beyond X and Y.*

No inverse mapping theorem obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.InverseMappingTheoremUp.InverseMappingTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 2291.1. The endpoint data are X and Y, the bounded operator is T, the bijectivity data are B, the open-mapping witness is I, and the inverse-window ledger is H.

Endpoint agreement is a rowwise condition. Since T, B, and I are all carried over the same displayed Banach endpoints, an inverse-facing consumer cannot mix an operator row from one source with an image-row witness from another target.

The bounded-inverse estimate is read in two stages. First B supplies the displayed injective and surjective windows for T. Then I supplies the open-ball image row from the $\mathsf{OpenMappingTheorem}^{\uparrow}$ packet, and only after that can H expose a source-window request and an inverse norm-bound comparison.

Closed-graph compatibility is the same displayed route viewed through graph reads. The closed-graph chapter supplies only the graph-facing boundary, so it cannot replace the Banach, bounded-operator, bijectivity, open-mapping, or inverse-window rows.

Finally C, P, N transport, replay, record provenance for, and name exactly this finite route. The excluded host inverse, quotient operator, selected-value, and ambient-completeness principles have no coordinate in the tuple. □

Closure status for $\mathsf{NameCert}_{\mathsf{InverseMappingTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InverseMappingTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2291.1 and Definition 2291.2 seed the finite bounded-inverse certificate over Banach source and target rows, bounded linear operators, bijectivity ledgers, open-mapping image witnesses, inverse-window ledgers, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InverseMappingTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, quotient operator spaces, selected inverse values outside displayed windows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint agreement, bijective operator exposure, open-mapping handoff, closed-graph compatibility, structural replay, and ledger non-escape for BEDC.Derived.InverseMappingTheoremUp.

Constructive story: An $\mathsf{InverseMappingTheorem}^{\uparrow}$ packet is generated from displayed Banach endpoints, a BoundedLinearOperatorUp row, bijectivity ledgers, an OpenMappingTheoremUp image-row witness, inverse-window norm-bound data, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2292

A Concrete Naming Certificate for CompactUniformContinuityModulus$^{\uparrow}$

Following Bishop–Bridges/Heine–Cantor constructive compactness, the CompactUniformContinuityModulus$^{\uparrow}$ packet records the concrete tolerance-only modulus extracted from finite compact-net data and displayed pointwise continuity rows. It realizes the finite-net proof route of chapter 9237 through the compact metric surface of chapter 129, the graph and pointwise-radius surface of chapter 130, the downstream UniformModulus$^{\uparrow}$ interface of chapter 369, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names the modulus object itself, not an ambient open-cover theorem or a host-chosen Lebesgue number.

Definition 2292.1 (Compact uniform continuity modulus carrier)**.** *A* CompactUniformContinuityModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, F, M, D, R, H, C, P, N)$$

with the following visible rows:

1. *K is the compact finite-net row, including the displayed center bundle and covering radii used by the compact source.*
2. *F is the* ContinuousMap$^{\uparrow}$ *graph row together with the pointwise modulus rows attached to the centers named by K.*
3. *M is the selected rational lower-bound fold over the centerwise source radii supplied by F.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested target tolerance, the centerwise radius comparisons, the lower-bound fold, and the final source tolerance.*
5. *R is the* Real$^{\uparrow}$ *seal boundary used only to read rational-radius and target-distance comparisons after the dyadic ledger has been displayed.*
6. *H records componentwise* hsame *transport for the compact source, pointwise modulus rows, lower-bound fold, dyadic ledger, real seal boundary, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a tolerance request through K, F, M, D, R, H, P, N to the downstream* UniformModulus$^{\uparrow}$ *row.*
8. *P is the* Pkg *provenance over* CompactUniformContinuityModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, host continuity, quotient stream equality, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityModulusUp.CompactUniformContinuityModulusTasteGate`
`single carrier alignment`

Theorem 2292.2 (Compact uniform continuity modulus NameCert obligations)**.** *For an accepted* $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$ *carrier* $Q = (K, F, M, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinui}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the compact finite-net row, pointwise modulus rows, selected rational lower-bound fold, dyadic tolerance ledger, real-seal boundary, transport, replay, provenance, and local naming;*
2. *source lock, requiring K and the source of F to name the same compact metric carrier before any center radius or graph row is read;*
3. *local modulus coverage, requiring every center selected by K to have a corresponding pointwise row in F;*
4. *lower-bound fold exactness, requiring the source tolerance to be computed through M and D from the displayed finite radius ledger rather than by a host minimum or selected Lebesgue number;*
5. *handoff exactness, requiring the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer to read only the tolerance row transported through H, C, P, N after the* $\mathsf{Real}^{\uparrow}$ *boundary R has sealed the rational comparisons.*

No compact uniform continuity modulus obligation is stored outside these rows.

Proof. Read the accepted carrier of Definition 2292.1. The compact source row K supplies only a finite center bundle and covering radii, while the continuous-map row F supplies graph reads and pointwise continuity radii over the same source.

The pointwise modulus rows in F attach radius information to the displayed centers. Thus each center used by the compact-net row has an explicit continuity row, and no center outside the finite bundle can affect the extracted modulus.

The selected rational lower-bound fold is carried by M and checked through D. It compares the requested target tolerance with the centerwise radii and records the finite route to the source tolerance. The $\mathsf{Real}^{\uparrow}$ seal boundary R is read only after these dyadic comparisons have been displayed.

The downstream $\mathsf{UniformModulus}^{\uparrow}$ row is therefore a tolerance-only modulus obtained from finite packet data. The transport, replay, provenance, and local naming rows preserve this route, so open-cover compactness, selected subcovers, host continuity, quotient stream equality, choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in the certificate. □

Theorem 2292.3 (Lebesgue number closed-interval compactness handoff)**.** *Every accepted* $\mathsf{CompactUniformContinu}$ *packet* $Q = (K, F, M, D, R, H, C, P, N)$ *hands a tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *row to compact-continuous consumers through finite-net coverage in K, pointwise modulus rows in F, the selected rational lower-bound fold M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance transport in D, the* $\mathsf{Real}^{\uparrow}$ *seal boundary R, and the structural rows H, C, P, N. The handoff exports no open-cover compactness theorem, selected subcover, host-continuity principle, choice coordinate, quotient stream equality, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the finite-net row K. It displays the centers and coverage radii that replace an open-cover compactness read at the packet boundary.

Read the pointwise rows F over those centers. The selected rational lower-bound fold M then combines the displayed source radii, and the dyadic ledger D transports the requested target tolerance through that fold.

Finally read the seal and structural rows. The $\mathsf{Real}^{\uparrow}$ boundary R seals only the rational comparisons already displayed by D, while H, C, P, N transport, replay, source, and name the same route. Hence the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer receives exactly this tolerance-only row and no excluded compactness, continuity, choice, quotient, or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2292.1, Definition 2292.2, and Definition 2292.3 seed the finite modulus extraction certificate over compact finite-net rows, pointwise modulus rows, selected rational lower-bound folds, DyadicRatCore tolerance ledgers, RealUp seal boundaries, UniformModulus handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host continuity, choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source lock, local modulus coverage, lower-bound fold exactness, DyadicRatCore transport, Real seal boundary, UniformModulus handoff, transport and replay stability, and TasteGate field faithfulness for BEDC.Derived.CompactUniformContinuityModulusUp.

Constructive story: A $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMapUp pointwise-radius rows, selected rational lower-bound folds, DyadicRatCoreUp tolerance ledgers, RealUp comparison seals, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2293

A Concrete Naming Certificate for $\mathsf{PoissonKernel}^{\uparrow}$

Following Poisson and classical Fourier analysis, the $\mathsf{PoissonKernel}^{\uparrow}$ packet records the finite positive harmonic-analysis kernel by which a displayed radius row and angle-window row are read before any boundary or Abel-type consumer may compare it with Fejer and Dirichlet kernels. It sits beside the positive averaging surface $\mathsf{FejerKernel}^{\uparrow}$ of chapter 1939, the partial-sum surface $\mathsf{DirichletKernel}^{\uparrow}$ of chapter 2311, the terminal real-name seal of chapter 38, the regular rational sequence route of chapter 301, and the dyadic arithmetic ledger of chapter 291. The chapter names a finite positive kernel boundary; it does not import host harmonic functions, quotient function equality, analytic completion, countable choice, or a selected boundary limit.

Definition 2293.1 (PoissonKernel carrier)**.** *A* $\mathsf{PoissonKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, A, D, M, V, F, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the radius row, recording the displayed rational or dyadic radius request together with the comparison ledger asserting it lies inside the finite admissible radial window.*
2. *A is the angle-window row, recording the finite circle or phase samples at which the kernel is inspected.*
3. *D is the denominator row, recording the displayed expression $1 - 2r\cos(\theta) + r^2$ through finite real, regular-rational, and dyadic comparison rows.*
4. *M is the numerator row, recording the displayed expression $1 - r^2$ with the same radius ledger as R.*
5. *V is the kernel-value readback row, exposing only the finite quotient read determined by M, D, A, R.*
6. *F is the* $\mathsf{FejerKernel}^{\uparrow}$ *comparison row, recording the positive-kernel averaging corridor through which Fejer-facing consumers may compare displayed finite windows.*
7. *S is the* $\mathsf{DirichletKernel}^{\uparrow}$ *comparison row, recording the partial-sum corridor through which Dirichlet-facing consumers may compare displayed finite windows.*
8. *H records componentwise* hsame *transport for the radius, angle-window, denominator, numerator, value, comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a boundary request through R, A, D, M, V, F, S in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{PoissonKernel}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PoissonKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, D, M, V, F, S, H, C, P, N$. It has no coordinate for host harmonic-function equality, analytic completion, quotient boundary functions, countable choice, or a selected radial limit.

Lean4 ✓ `BEDC.Derived.PoissonKernelUp.TasteGate.PoissonKernelTasteGate single carrier alignment`

Theorem 2293.2 (PoissonKernel NameCert obligations)**.** *For every accepted* $\mathsf{PoissonKernel}^{\uparrow}$ *carrier $K = (R, A, D, M, V, F, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PoissonKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for the radius row, angle-window row, denominator row, numerator row, kernel-value readback, Fejer comparison row, Dirichlet comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public kernel read must first expose the same radius and angle rows, then the denominator and numerator ledgers, then the value readback, and only then the Fejer or Dirichlet comparison route. No Poisson-kernel obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2293.1. The displayed tuple exposes exactly the radius, angle-window, denominator, numerator, value, Fejer comparison, Dirichlet comparison, transport, replay, provenance, and local naming rows.

The value route is finite because V is read only after R, A, D, M have been displayed. The denominator and numerator ledgers share the radius row, so a consumer cannot replace the radial budget while preserving the accepted packet.

The comparison rows are downstream. Fejer-facing reads pass through F, Dirichlet-facing reads pass through S, and both routes inherit the same radius, angle, numerator, denominator, and value rows before any handoff is exposed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for host harmonic functions, analytic completion, quotient boundary functions, countable choice, or selected radial limits, the refusal clause is exhausted by the carrier inventory. □

Theorem 2293.3 (PoissonKernel Fejer and Dirichlet handoff)**.** *Every accepted* $\mathsf{PoissonKernel}^{\uparrow}$ *comparison read factors through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{FejerKernel}^{\uparrow},\ \mathsf{DirichletKernel}^{\uparrow}.$$

The radius and angle rows determine the displayed numerator, denominator, and kernel-value readback before either the positive Fejer corridor or the partial-sum Dirichlet corridor is exposed. The route exports no host boundary convergence theorem, analytic completion, quotient function equality, countable-choice selector, or selected limiting harmonic extension.

Proof. Let $K = (R, A, D, M, V, F, S, H, C, P, N)$ be accepted. Carrier projection exposes the real, regular-rational, and dyadic arithmetic rows through R, A, D, M, V before any comparison row can be inspected.

Replay C. A comparison request first reads the radius row and angle window, then the denominator and numerator ledgers, and then the kernel-value readback. Thus both comparison routes receive the same finite kernel value rather than a host function.

The Fejer handoff is exactly F, and the Dirichlet handoff is exactly S. Componentwise transport through H preserves the same displayed rows, while P, N record provenance and local naming. The excluded analytic and choice-theoretic objects are absent from the carrier, so no comparison read can export them. □

Falsifiable prediction: Every accepted $\mathsf{PoissonKernel}^{\uparrow}$ boundary-kernel read factors through a radius row, angle-window row, denominator row, numerator row, kernel-value readback, Fejer comparison row, Dirichlet comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: fejer-kernel, dirichlet-kernel, real, regseqrat, dyadicratcore: $\mathsf{PoissonKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a radius row, angle-window row, denominator ledger, numerator ledger, kernel-value readback, Fejer comparison, Dirichlet comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoissonKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{PoissonKernel}^{\uparrow}$ packet is generated from a radius row, angle-window row, denominator row, numerator row, kernel-value readback, Fejer and Dirichlet comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoissonKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2293.1, Definition 2293.2, and Definition 2293.3 seed the finite positive Poisson-kernel surface for harmonic-analysis consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoissonKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host harmonic-function equality, analytic completion, quotient boundary functions, countable choice, selected radial limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius exactness, denominator and numerator stability, value readback, Fejer and Dirichlet handoff, transport, replay, provenance, local naming, Nontrivial PoissonKernelUp, and FieldFaithful PoissonKernelUp for BEDC.Derived.PoissonKernelUp.

Chapter 2294

A Concrete Naming Certificate for $\mathsf{ConstructiveGreenFunction}^{\uparrow}$

Following Courant–Hilbert and Evans, the $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ packet records a finite boundary-kernel object for Dirichlet-style Green data without importing host harmonic-function equality or analytic completion. It sits beside the positive boundary kernel of chapter 2293, the finite PDE relation rows of chapter 194, the weak-trace surface of chapter 440, and the metric carrier of chapter 57. The packet names only displayed domain, boundary, relation, trace, distance, transport, replay, provenance, and local naming rows.

Definition 2294.1 (ConstructiveGreenFunction carrier)**.** *A* $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, B, R, K, S, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain row, recording the finite carrier of interior observations and the boundary-access ledger used by the Dirichlet request.*
2. *B is the boundary row, recording the named boundary samples and their finite incidence relation with the domain row.*
3. *R is the* $\mathsf{Pde}^{\uparrow}$ *relation row, recording the elliptic or boundary-value relation through displayed derivative and boundary-data rows.*
4. *K is the* $\mathsf{PoissonKernel}^{\uparrow}$ *comparison row, exposing the positive boundary-kernel corridor used when the Green packet is compared with a Poisson–Dirichlet read.*
5. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *trace row, recording the weak-trace and integrability data consumed by boundary-facing PDE requests.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row, recording the displayed interior-to-boundary distance, diagonal refusal, and comparison ledger.*
7. *H records componentwise* hsame *transport for the domain, boundary, PDE relation, Poisson comparison, Sobolev trace, metric distance, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-kernel request through $D, B, R, K, S, M, H, P, N$ in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveGreenFunction}^{\uparrow}$, $\mathsf{PoissonKernel}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveGreenFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host harmonic-function equality, analytic Green operators, quotient boundary functions, selected fundamental solutions, countable choice, or completed solution spaces.

Lean4 ✓ `BEDC.Derived.ConstructiveGreenFunctionUp.ConstructiveGreenFunctionTasteGate single carrier alignment`

Theorem 2294.2 (ConstructiveGreenFunction NameCert obligations)**.** *For every accepted* ConstructiveGreenFunct *carrier $G = (D, B, R, K, S, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveGreenFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the domain row, boundary row, PDE relation row, Poisson-kernel comparison row, Sobolev trace row, metric distance row, componentwise transport, displayed replay, provenance, and local naming row;*
2. *boundary-kernel exactness, requiring every Green read to fix D and B before it may inspect either the PDE relation row R or the Poisson comparison row K;*
3. *trace compatibility, requiring the Sobolev trace row S to consume the same boundary samples as B and the same PDE relation endpoints as R;*
4. *metric separation, requiring the distance row M to be the only source for interior-to-boundary distance comparisons and diagonal refusals;*
5. *transport and replay stability, requiring H and C to preserve the ordered route D, B, R, K, S, M under componentwise* hsame *and displayed continuation;*
6. *provenance and name exhaustion, requiring P and N to export no host harmonic-function equality, analytic completion, quotient boundary function, selected fundamental solution, countable-choice selector, or completed solution space.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2294.1. The displayed tuple exposes exactly the domain, boundary, PDE relation, Poisson comparison, Sobolev trace, metric distance, transport, replay, provenance, and local naming rows.

For boundary-kernel exactness, replay a request through C. The request first fixes the domain row D and boundary row B; only then can it inspect the PDE relation row R or the Poisson comparison row K.

Trace compatibility is read from B, R, S. The Sobolev trace row consumes the same boundary samples and the same derivative or relation endpoints already displayed by the PDE row, so it does not introduce a separate host trace operator.

Metric separation is the role of M. Interior-to-boundary comparisons and diagonal refusals are read from the displayed distance row before any downstream Green or Dirichlet consumer can cite them.

Finally, H, C, P, N transport, replay, source, and name the same finite route. The excluded analytic and choice-theoretic objects are absent from the carrier inventory, so the local certificate cannot export them. □

Theorem 2294.3 (ConstructiveGreenFunction Poisson–Dirichlet handoff)**.** *Every Poisson–Dirichlet consumer of an accepted* $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{Pde}^{\uparrow} \longmapsto \mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{PoissonKernel}^{\uparrow}.$$

The route first reads the displayed distance and boundary incidence rows, then the PDE relation row, then the Sobolev trace row, and only then the Poisson-kernel comparison. It exports no host harmonic-function equality, analytic completion, quotient boundary function, selected Green operator, countable-choice selector, or completed solution space.

Proof. Start with an accepted carrier $G = (D, B, R, K, S, M, H, C, P, N)$. Carrier projection exposes M as the only metric distance source and exposes D, B as the only domain and boundary incidence rows.

Replay the continuation row C. A Dirichlet request reads the displayed distance and boundary data before it reaches the PDE relation row R, so the relation is consumed as finite BEDC data rather than as a host differential equation.

The Sobolev trace row S is then read over the same boundary samples and relation endpoints. This supplies the trace-facing ledger needed by the boundary consumer without adding a completed Sobolev quotient or selected representative.

Only after this trace route is fixed can the Poisson comparison row K be inspected. Transport through H and provenance through P, N preserve that ordered route, and the excluded analytic objects are not coordinates of the carrier. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ boundary-kernel read factors through domain rows, boundary rows, PDE relation rows, PoissonKernel comparison, Sobolev trace data, MetricUp distance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: poisson-kernel, pde, sobolev, metric: $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries domain and boundary rows, PDE relation rows, Poisson comparison, Sobolev trace data, metric distance rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveGreenFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveGreenFunction}^{\uparrow}$ packet is generated from domain and boundary rows, PDE relation rows, PoissonKernel comparison, Sobolev trace data, MetricUp distance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveGreenFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2294.1, Definition 2294.2, and Definition 2294.3 seed the finite boundary-kernel certificate surface linking Poisson kernels, PDE relations, Sobolev trace data, and metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveGreenFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host harmonic-function equality, analytic Green operators, quotient boundary functions, selected fundamental solutions, countable choice, completed solution spaces, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, boundary-kernel exactness, trace compatibility, metric separation, Poisson–Dirichlet handoff, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveGreenFunctionUp.

Chapter 2295

A Concrete Naming Certificate for $\mathsf{FiniteModulusDiagonal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ packet records the finite diagonal extraction surface needed by L10 Real-and-Completion consumers. It sits over the modulus-controlled Cauchy read of chapter 1423, the finite stream-window surface of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal boundary of chapter 38. The packet names a finite diagonal route before any quotient stream equality, selected limit, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is available.

Definition 2295.1 (Finite modulus diagonal carrier)**.** *A* $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, S, T, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{RealCauchyModulus}^{\uparrow}$ *modulus packet that schedules each requested dyadic precision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger selected by M for the diagonal read.*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the finite diagonal comparisons inspected on those windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the scheduled finite windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, S, T, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus packet, window ledger, tolerance ledger, readback row, seal row, continuation row, provenance row, and local naming row.*
7. *C records the displayed* Cont *replay from a finite diagonal request through M, S, T, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusDiagonal}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a host-selected diagonal subsequence, countable choice, trichotomy, a detached limit, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusDiagonalUp.FiniteModulusDiagonalTasteGate single carrier alignment`

Theorem 2295.2 (Finite modulus diagonal NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{FiniteModulusDiagonal}^{\uparrow}}$ *surface for Definition 2295.1 consists exactly of carrier exposure for D, modulus exposure through M, finite-window exposure through S, dyadic diagonal tolerance exposure through T, regular rational readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. Each accepted finite diagonal request is classified only by the displayed route through the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *packet, the* $\mathsf{StreamName}^{\uparrow}$ *window ledger, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row.*

Proof. Project the carrier tuple $D = (M, S, T, Q, E, H, C, P, N)$. The analytic rows are exactly the modulus packet, stream windows, dyadic tolerance ledger, regular readback, and real seal.

Read the structural rows next. The witnesses H, C, P, N transport, replay, source, and name the same finite route, so the local certificate has no hidden analytic coordinate.

The excluded principles are absent from the tuple in Definition 2295.1. Hence the listed rows exhaust the seed obligation surface. □

Theorem 2295.3 (Finite modulus diagonal window cofinality)**.** *Let $D = (M, S, T, Q, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ *packet. For any two finite diagonal precision requests whose second request refines the first, the window selected by M and displayed in S for the second request is cofinal over the earlier window. It preserves the earlier* $\mathsf{StreamName}^{\uparrow}$ *observations, keeps the earlier* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance entries inside T, and leaves the associated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and terminal* $\mathsf{Real}^{\uparrow}$ *seal E attached to the same carrier route.*

Proof. Read the underlying modulus packet M through chapter 1423. Refining a finite precision request may enlarge the displayed StreamName window, but it does not erase the earlier scheduled window.

The dyadic ledger row T records both the earlier tolerance entries and the refined entries as finite packet data. The readback row Q is consumed only after this ledger is displayed, so the refined request preserves the earlier rational readback route.

Finally replay C and transport by H. The terminal seal row E remains the same displayed $\mathsf{Real}^{\uparrow}$ handoff, and the provenance and local naming rows P, N keep the cofinal route inside the accepted carrier. □

Theorem 2295.4 (Finite modulus diagonal Real-seal handoff)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ *packet factors through*

$$\mathsf{RealCauchyModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Equivalently, the diagonal consumer reaches E only after the finite modulus row M, window ledger S, tolerance ledger T, and regular readback row Q have been displayed and replayed by H, C, P, N.

Proof. Project the accepted carrier. The row E is not a detached real value; it is a named coordinate after M, S, T, Q in the finite packet.

Use the cofinal window theorem Definition 2295.3. Any stronger diagonal request preserves the earlier finite window and readback route before it reaches the terminal seal.

The rows H, C, P, N only transport, replay, source, and name that route. Therefore every accepted Real-seal read factors through the displayed modulus-window-ledger-readback handoff. □

Lean4 ✓ `BEDC.Derived.FiniteModulusDiagonalUp.FiniteModulusDiagonalRealSealHandoff`

Theorem 2295.5 (Finite modulus diagonal ledger nonescape)**.** *No downstream L10 consumer of an accepted* $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ *packet can bypass the displayed ledger* $M, S, T, Q, E, H, C, P, N$*. In particular, the packet does not expose a quotient stream, selected diagonal subsequence, host real equality, countable-choice row, unbounded stream read, detached limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Inspect the carrier in Definition 2295.1. The only analytic coordinates are the modulus packet, finite stream windows, dyadic tolerance ledger, regular rational readback, and terminal real seal.

The NameCert obligation theorem Definition 2295.2 says that every accepted local read is classified through those coordinates and the structural rows H, C, P, N.

Thus a consumer route omitting the displayed ledger is not a read of the accepted packet. The excluded principles have no carrier coordinate and cannot be introduced by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusDiagonal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteModulusDiagonal}^{\uparrow}$ packet is generated from a RealCauchyModulus schedule, StreamName finite windows, a DyadicRatCore diagonal tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2295.1, Definition 2295.2, Definition 2295.3, Definition 2295.4, and Definition 2295.5 seed the finite modulus diagonal carrier over $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected diagonal subsequences, host real equality, countable choice, unbounded stream reads, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formalization should encode the carrier rows and prove the finite-window cofinality and Real-seal handoff over the displayed BEDC packet.

Chapter 2296

A Concrete Naming Certificate for $\mathsf{MedianLeakageSpectrum}^{\uparrow}$

The $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ packet realizes the active median leakage table of Definition 9462.1336 and the direct leakage spectrum of Definition 9462.1344 as finite BEDC NameCert data. It consumes the six-coordinate source window of chapter 2234, the finite codon transition vocabulary of chapter 2245, the stabilization boundary of chapter 2270, the conditioned survival-chain route of **??**, and the host-primitive refusal surface of chapter 1157. The packet names a finite leakage spectrum; it does not identify median closure with biological translation identity, choose a stochastic model, or import host primitive enumeration.

Definition 2296.1 (MedianLeakageSpectrum carrier)**.** *A* $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, T, A, M, I, G, D, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the finite six-bit codon carrier and coordinatewise median readback.*
2. *T is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *transition-vocabulary row naming the finite scan, elongate, terminate, readthrough, repair, and mutate states available to the consumer.*
3. *A is the active-triple row, recording the* 286 *distinct active triples and the closed versus leaking split* $264 + 22$.
4. *M is the median-output row, recording the coordinatewise majority output attached to each active triple.*
5. *I is the inactive-output row, listing the six direct inactive outputs* UUG, AAG, UGG, ACA, AUG, *and* UCG.
6. *G is the centrality ledger, recording the median centrality counts for active and inactive outputs together with the direct leakage multiplicities.*
7. *D is the direct-output multiplicity row, recording the displayed multiplicities* $8, 4, 4, 2, 2, 2$ *for the six inactive outputs.*
8. *H records componentwise* hsame *transport for the source window, transition vocabulary, active triples, median outputs, inactive outputs, centrality ledger, direct multiplicities, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a leakage request through W, T, A, M, I, G, D in that order.*
10. *P is the* Pkg *provenance over* MedianLeakageSpectrum$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, CodonTransitionMatrix$^{\uparrow}$, SpectralStabilization$^{\uparrow}$, DoobSurvivalChain$^{\uparrow}$, HostPrimitiveLeakage$^{\uparrow}$, *Definition 9462.1336, Definition 9462.1344,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MedianLeakageSpectrum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, T, A, M, I, G, D, H, C, P, N$. It has no coordinate for host primitive enumeration, biological translation identity, stochastic model selection, quotient codon equality, or an unrecorded median table.

Lean4 ✓ `BEDC.Derived.MedianLeakageSpectrumUp.MedianLeakageSpectrumUp single carrier alignment`

Theorem 2296.2 (MedianLeakageSpectrum NameCert obligations)**.** *For every accepted* MedianLeakageSpectrum$^{\uparrow}$ *carrier $L = (W, T, A, M, I, G, D, H, C, P, N)$, the local* NameCert$_{\text{MedianLeakageSpectrum}^{\uparrow}}$ *surface consists exactly of carrier admission for the codon source window, transition vocabulary, active-triple row, median-output row, inactive-output row, centrality ledger, direct-output multiplicity row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every leakage multiplicity read must expose A, M, I, G, D over the same W, T source before any downstream spectral, survival-chain, or host-primitive refusal consumer may inspect it. No median-leakage obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2296.1. The tuple displays the source window, transition vocabulary, active triples, median outputs, inactive outputs, centrality counts, direct multiplicities, transport, replay, provenance, and local naming rows.

The spectrum route is ordered. A consumer first reads W, T, then the active-triple ledger A, then the median-output row M, then the inactive-output row I, and then the centrality and multiplicity rows G, D.

The downstream dependencies are consequently consumers rather than hidden data. SpectralStabilization$^{\uparrow}$ and DoobSurvivalChain$^{\uparrow}$ may read the displayed spectrum only after the finite leakage rows are present, while HostPrimitiveLeakage$^{\uparrow}$ records that no host primitive enumeration may replace them.

Finally, H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for biological translation identity, stochastic model selection, quotient codon equality, host primitive enumeration, or an unrecorded median table, the refusal clause is exhausted by the carrier inventory. □

Theorem 2296.3 (MedianLeakageSpectrum direct output route)**.** *Every accepted* MedianLeakageSpectrum$^{\uparrow}$ *direct-output read factors through*

$$\text{CodonSixBitWindow}^{\uparrow} \longmapsto \text{CodonTransitionMatrix}^{\uparrow} \longmapsto \text{SpectralStabilization}^{\uparrow} \longmapsto \text{DoobSurvivalChain}^{\uparrow} \longmapsto \text{HostPrimi}$$

The read exposes the active-triple ledger, median-output ledger, inactive-output row, centrality ledger, and direct-output multiplicities before it reaches any spectral stabilization, survival-chain, or host-primitive refusal consumer. In particular, a leakage multiplicity may not be read from D without the rows A, M, I, G that certify the finite table from which it came.

Proof. Let $L = (W, T, A, M, I, G, D, H, C, P, N)$ be accepted. Carrier projection first exposes the codon source window and finite transition vocabulary, so the six-bit median readback is tied to the same finite codon surface throughout the packet.

Replay C. The leakage request passes through the active triples, median outputs, inactive outputs, centrality ledger, and direct multiplicity row in sequence. Thus D cannot be consumed unless A, M, I, G have already been displayed.

The remaining siblings consume the displayed route. $\mathsf{SpectralStabilization}^{\uparrow}$ reads the finite spectrum boundary, $\mathsf{DoobSurvivalChain}^{\uparrow}$ reads the conditioned survival-chain corridor, and $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ refuses any replacement by host primitive enumeration. Transport through H and provenance through P, N preserve the same dependency route. □

Falsifiable prediction: Every accepted consumer that reads a $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ leakage multiplicity without the active-triple row, median-output row, inactive-output row, and centrality ledger refutes the packet.

Independence witness: codon-six-bit-window, codon-transition-matrix, spectral-stabilization, doob-survival-chain, hostprimitiveleakage: $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries active triples, median outputs, inactive leakage rows, centrality counts, direct-output multiplicities, transport, replay, provenance, and local naming.

The Lean follow-up for `BEDC.Derived.MedianLeakageSpectrumUp` must supply explicit Nontrivial MedianLeaka and FieldFaithful $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ instances preserving $W, T, A, M, I, G, D, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianLeakageSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{MedianLeakageSpectrum}^{\uparrow}$ packet is generated from a CodonSixBitWindow source row, CodonTransitionMatrix vocabulary row, active triples, median outputs, inactive leakage rows, centrality counts, direct-output multiplicities, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MedianLeakageSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2296.1, Definition 2296.2, and Definition 2296.3 seed the finite median-leakage spectrum surface for codon-window, transition, spectral, survival-chain, and host-primitive refusal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MedianLeakageSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological translation identity, stochastic model selection, quotient codon equality, host primitive enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, active-triple exactness, median-output exactness, inactive-output exactness, centrality and multiplicity ledgers, transport, replay, provenance, local naming, Nontrivial MedianLeakageSpectrumUp, and FieldFaithful MedianLeakageSpectrumUp for BEDC.Derived.MedianLeakageSpectrumUp.

Chapter 2297

A Concrete Naming Certificate for $\mathsf{PositiveCone}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{PositiveCone}^{\uparrow}$ packet isolates the positive cone over $\mathsf{Real}^{\uparrow}$ as finite apartness-from-zero data with algebra-facing closure rows. It refines the $\mathsf{PositiveReal}^{\uparrow}$ carrier of chapter 1587, the $\mathsf{ApartnessReal}^{\uparrow}$ separation surface of chapter 304, the $\mathsf{LocatedApartness}^{\uparrow}$ route of chapter 1380, and the finite-name substrates $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ used by chapter 38.

2297.1 Carrier and cone packet

Definition 2297.1 (PositiveCone carrier)**.** *A* $\mathsf{PositiveCone}^{\uparrow}$ *carrier is a finite packet* $(R, P, A, D, W, Q, H, C, K, N)$ *with the following displayed rows:*

1. R*, a sealed* $\mathsf{Real}^{\uparrow}$ *row for the real name whose cone membership is being classified;*
2. P*, a* $\mathsf{PositiveReal}^{\uparrow}$ *row naming the same real through finite positivity evidence;*
3. A*, an* $\mathsf{ApartnessReal}^{\uparrow}$ *apartness-to-zero row whose left endpoint is the zero real row and whose right endpoint is* R*;*
4. D*, a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger consumed by* A *and* P*;*
5. W*, a finite* $\mathsf{StreamName}^{\uparrow}$ *observation window shared by the zero row, the real row, and the positivity packet;*
6. Q*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled rational observations of* R *on* W*;*
7. H*, the componentwise* hsame *rows for the real seal, positivity row, apartness row, radius ledger, window, and readback;*
8. C*, the displayed* Cont *replay routes keeping the finite observations attached to the transported packet;*
9. K*, the* Pkg *provenance rows for the real seal, positive-real row, apartness ledger, dyadic radius, stream window, regular-rational readback, and replay surface;*
10. N*, the local* $\mathsf{NameCert}$ *row binding these nine rows as one positive cone certificate.*

The carrier contains no order decision coordinate, no quotient stream equality, no selected limit, and no ambient completeness witness. Cone membership is the finite compatibility of the displayed

real, positive-real, apartness, radius, window, readback, transport, continuation, provenance, and local certificate rows.

Lean4 ✓ `BEDC.Derived.PositiveConeUp.PositiveConeTasteGate single carrier alignment`

2297.2 NameCert obligations

Theorem 2297.2 (PositiveCone NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{PositiveCone}^{\uparrow}}$ *surface for the carrier of Definition 2297.1 consists of six finite obligations:*

1. *carrier admission: the displayed rows* $(R, P, A, D, W, Q, H, C, K, N)$ *occur in one finite* BHist *packet;*
2. *classifier transport: accepted cone carriers compare only by componentwise transport of the real seal, positive-real row, apartness row, radius ledger, finite window, readback, replay, provenance, and local certificate rows;*
3. *zero-exclusion: the apartness-to-zero row* A *and the positive-real row* P *share the same sealed real endpoint* R *and the same finite observation window* W*;*
4. *positive-radius exactness: the positive dyadic radius used by the apartness row and by the positive-real row is exactly the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger* D*;*
5. *addition and multiplication closure surface: algebra consumers receive only the finite cone rows required to transport positivity across displayed addition-facing and multiplication-facing packet reads;*
6. *ledger non-escape: downstream consumers can inspect only the ten carrier rows and their* hsame*,* Cont*,* Pkg*, and* NameCert *witnesses.*

These obligations certify the positive cone as a finite NameCert packet over the real-name substrate. They do not certify trichotomy, a host total order, quotient equality of streams, selected limits, or ambient real completeness.

Proof. Project the carrier of Definition 2297.1. The projection exposes the sealed real row, the positive-real row, the apartness-to-zero row, the positive dyadic radius, the finite observation window, the regular-rational readback, transport, replay, provenance, and local certificate rows.

Carrier admission and classifier transport are componentwise. The hsame and Cont rows keep the real seal, positive-real packet, apartness packet, radius ledger, stream window, and readback attached to the same finite observation surface after transport.

Zero-exclusion and positive-radius exactness are read from the shared rows. The apartness endpoint and the positive-real carrier both name R, while D is the only positive dyadic radius ledger available to either positivity read.

The closure surface is therefore only a ledger interface for later algebraic consumers. Since the packet contains no coordinate for order decisions, quotient representatives, chosen limits, or completeness witnesses, every consumer remains confined to the displayed finite evidence. □

Theorem 2297.3 (PositiveCone apartness-order handoff)**.** *Every accepted* $\mathsf{PositiveCone}^{\uparrow}$ *read factors through the displayed* $\mathsf{PositiveReal}^{\uparrow}$*,* $\mathsf{ApartnessReal}^{\uparrow}$*, and* $\mathsf{LocatedApartness}^{\uparrow}$ *rows, together with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite window, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of Definition 2297.1. Consequently a later order, metric-radius, or real-algebra consumer receives cone membership only as finite apartness-radius evidence with visible* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

No accepted read exports trichotomy, host order, quotient stream equality, selected limits, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted cone carrier and project its positive-real coordinate P. By Definition 2297.1, P shares the real endpoint R, the radius ledger D, the finite window W, and the readback row Q with the apartness-to-zero coordinate A.

Read the packet through the surrounding siblings. The $\mathsf{PositiveReal}^{\uparrow}$ row supplies positivity as apartness from zero, the $\mathsf{ApartnessReal}^{\uparrow}$ row exposes the finite separation packet, and the $\mathsf{LocatedApartness}^{\uparrow}$ route supplies the located handoff used by downstream metric-radius consumers.

Transport is again componentwise. The hsame, Cont, Pkg, and NameCert rows move the positive-real, apartness, radius, window, and readback coordinates together, so every transported read still factors through the displayed finite surfaces.

The carrier has no row for totality, trichotomy, quotient equality, selected limits, or real completeness. Therefore the handoff is an order-facing interface only through finite apartness-radius evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveCone}^{\uparrow}}$.

Constructive story: A $\mathsf{PositiveCone}^{\uparrow}$ packet is generated from a sealed real row, a positive-real row, an apartness-to-zero row, one positive dyadic radius ledger, one finite observation window, one regular-rational readback, transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2297.1, Definition 2297.2, and Definition 2297.3 name the seed carrier, obligation surface, and finite apartness-order handoff for positive cone packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy of constructive reals, a host total order, quotient stream equality, selected limits, ambient real completeness, or a standard bridge.

Upgrade path: To reach obligationClosure, the carrier, six obligation rows, and apartness-order handoff must be checked as finite BEDC packet obligations.

Chapter 2298

A Concrete Naming Certificate for $\mathsf{ContextFreePumpingLemma}^{\uparrow}$

Following Bar-Hillel–Perles–Shamir and Hopcroft–Ullman, the $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ packet records the finite derivation-window evidence behind the context-free pumping lemma as NameCert data. It consumes the grammar surface of chapter 356, the regular-language comparison boundary of chapter 351, the finite automaton prefix surface of chapter 1016, the word rows of chapter 36, and the bound rows of chapter 28. The packet names the pumping certificate, not host language equality, infinite parse trees, grammar quotients, or a semantic closure theorem over all languages.

Definition 2298.1 (Context-free pumping lemma carrier)**.** *A* $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, W, D, A, E, I, Y, H, C, R, Q, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *grammar packet, including terminal alphabet, nonterminal carrier, start row, and production ledger.*
2. *W is the terminal* $\mathsf{List}^{\uparrow}$ *word row whose length comparison is read through* $\mathsf{Nat}^{\uparrow}$*.*
3. *D is the finite parse or derivation ledger yielding W from G.*
4. *A is the repeated-nonterminal row, recording the two derivation-prefix positions and the common nonterminal exposed inside the finite window.*
5. *E is the decomposition row, displaying the word factorization $W = uvxyz$ with the repeated-nonterminal window attached to v, x, y.*
6. *I is the pump-index row, read as a* $\mathsf{Nat}^{\uparrow}$ *input for the requested finite replay.*
7. *Y is the yielded-word row, displaying the pumped terminal word $uv^I xy^I z$ as a* $\mathsf{List}^{\uparrow}$ *row.*
8. *H records componentwise* hsame *transport for grammar rows, derivation prefixes, repeated-nonterminal evidence, decomposition rows, pump indices, and yielded words.*
9. *C records displayed* Cont *replay from a derivation prefix and pump index through the finite pump window.*
10. *R is the regular-language or finite-prefix consumer boundary, used only to compare the yielded word with sibling language-recognition packets.*

11. *Q is the* Pkg *provenance over* $\mathsf{ContextFreePumpingLemma}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FinitePrefixAutomaton}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContextFreePumpingLemma}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $G, W, D, A, E, I, Y, H, C, R, Q, N$. It contains no coordinate for host language equality, infinite derivation trees, grammar quotient semantics, countable choice over derivations, or semantic membership in an ambient language set.

Theorem 2298.2 (Context-free pumping lemma NameCert obligations)**.** *For every accepted* ContextFreePumpingL *carrier $P = (G, W, D, A, E, I, Y, H, C, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContextFreePumpingLemma}^{\uparrow}}$ *surface consists exactly of finite grammar admission through G; terminal-word and length admission through W; derivation-ledger admission through D; repeated-nonterminal exposure through A; decomposition through E; pump-index reading through I; yielded-word display through Y; componentwise transport through H; replay through C; sibling language-recognition boundary through R; provenance through Q; and local naming through N.*

Lean4 ✓ `BEDC.Derived.ContextFreePumpingLemmaUp.ContextFreePumpingLemmaTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 2298.1. The finite packet displays the grammar, word, derivation ledger, repeated nonterminal, decomposition, pump index, yielded word, transport, replay, consumer boundary, provenance, and local name.

The positive route begins at the finite derivation ledger D. The repeated nonterminal row A is read inside that ledger before the decomposition row E may expose the factors u, v, x, y, z.

The pump index I is then replayed through C to produce the yielded word row Y. The structural rows H, Q, N preserve and name that same finite route, while R only offers a sibling boundary for later regular-language or finite-prefix consumers.

Since the classifier has no coordinate for host language equality, infinite derivation trees, grammar quotient semantics, or semantic language membership, the listed rows exhaust the local NameCert obligations. □

Theorem 2298.3 (Context-free pumping window boundary)**.** *Let $P = (G, W, D, A, E, I, Y, H, C, R, Q, N)$ be an accepted* $\mathsf{ContextFreePumpingLemma}^{\uparrow}$ *carrier. Every pumpable-yield read of P factors through the finite derivation-prefix rows D, A, E, I, Y before reaching transport, replay, provenance, local naming, or a sibling language-recognition boundary. No accepted read factors through host language equality, infinite derivation trees, grammar quotient semantics, or semantic membership in an ambient language set.*

Proof. Project the carrier of Definition 2298.1. The only rows capable of producing a pumped yield are the finite derivation ledger D, the repeated-nonterminal row A, the decomposition row E, the pump-index row I, and the yielded-word row Y.

Transport and replay preserve this displayed window. The hsame row H changes no grammar or word coordinate, and the Cont row C replays the same derivation prefix at the requested index.

The provenance and local naming rows close the packet, while R only places the yielded word at a regular-language or finite-prefix boundary. Because the absent host and quotient semantics are not carrier coordinates, they cannot supply another pumpable-yield route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContextFreePumpingLemma}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContextFreePumpingLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2298.1, Definition 2298.2, and Definition 2298.3 seed the finite context-free pumping certificate over grammar, word, derivation-prefix, repeated-nonterminal, decomposition, pump-index, yielded-word, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ContextFreePumpingLemma}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ContextFreePumpingLemmaUp.ContextFreePumpingLemmaTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the full pumping lemma as a Lean theorem, host language equality, infinite derivation trees, grammar quotient semantics, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, repeated-nonterminal exposure, decomposition exactness, pump replay, yielded-word boundary, and ledger non-escape for BEDC.Derived.ContextFreePumpingLemmaUp.

Constructive story: A ContextFreePumpingLemma$^{\uparrow}$ packet is generated from finite grammar data, a terminal word, a finite derivation ledger, repeated-nonterminal evidence, a decomposition row, a pump index, a yielded word, componentwise hsame transport, Cont replay, sibling-recognition provenance, and local naming.

Chapter 2299

A Concrete Naming Certificate for $\mathsf{CompactUniformModulus}^{\uparrow}$

Following Bishop–Bridges compact metric continuity, the $\mathsf{CompactUniformModulus}^{\uparrow}$ packet realizes the finite-net fold named in chapter 9237. It consumes the metric source of chapter 57, the compact finite-cover surface of chapter 60, the pointwise continuity rows of chapter 59, the output interface of chapter 369, and the rational comparison rows of chapter 37. The packet names a finite uniform-modulus ledger, not an open-cover compactness theorem, selected Lebesgue number, sequential compactness principle, or host modulus function.

Definition 2299.1 (Compact uniform modulus carrier)**.** *A* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, K, F, L, Q, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the distance and comparison reads used by the compact and continuity surfaces.*
2. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, displaying the finite center bundle and covering radius ledger over X.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus row, attaching a source radius to each displayed center and requested target tolerance.*
4. *L is the local finite-net fold ledger, matching each center from K with the corresponding pointwise row from F.*
5. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row computing the finite lower bound used as the source tolerance.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row produced from the fold ledger and rational comparison.*
7. *H records componentwise* hsame *transport for the metric, compact, continuity, fold, rational, output, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a target-tolerance request through $X, K, F, L, Q, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, K, F, L, Q, U, H, C, P, N$. It has no coordinate for selected subcovers, selected Lebesgue numbers, sequence extraction, countable choice, host minima, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CompactUniformModulusUp.CompactUniformModulusUp`
Lean4 ✓ `BEDC.Derived.CompactUniformModulusUp.CompactUniformModulusTasteGate single carrier alignment`

Theorem 2299.2 (Compact uniform modulus NameCert obligations)**.** *For an accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier $M = (X, K, F, L, Q, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, compact finite-net row, pointwise continuity row, local fold ledger, rational comparison row, uniform-modulus output, transport, replay, provenance, and local naming; source lock between X, K, F; finite coverage of every center named by K; rational lower-bound exactness through Q; output admission through U; and ledger non-escape from selected subcovers, selected Lebesgue numbers, sequence extraction, choice, host minima, and ambient completeness.*

Proof. Project the carrier of Definition 2299.1. The source metric row X is shared by the compact row K and the continuity row F, so the fold cannot change the underlying metric carrier.

The finite-net fold is displayed by L. It pairs each center recorded in K with a pointwise continuity radius from F, giving a finite ledger rather than a host-selected subcover or sequence argument.

The rational comparison row Q computes the lower bound consumed by the output row U. Transport, replay, provenance, and local naming preserve this route, so the absent compactness and choice principles have no coordinate in the NameCert surface. □

Theorem 2299.3 (Compact finite net to uniform modulus)**.** *Let $M = (X, K, F, L, Q, U, H, C, P, N)$ be an accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier. Every uniform-modulus read exported by M factors through the finite route*

$$X, K, F, L, Q, U, H, C, P, N.$$

The route first reads the compact finite-net rows and pointwise continuity rows, then folds their displayed rational bounds into Q, then exposes the $\mathsf{UniformModulus}^{\uparrow}$ *row U. It never passes through selected subcovers, selected Lebesgue numbers, sequential compactness, host modulus functions, countable choice, or ambient complete metric spaces.*

Proof. The carrier definition displays the only modulus-producing coordinates. A read cannot reach U before it has fixed the common source metric X, the finite-net row K, and the pointwise continuity row F.

The fold ledger L and rational comparison row Q provide the finite lower-bound computation. Thus the output is obtained from named rational data, not from an external minimum or selected Lebesgue number.

Finally H, C, P, N transport, replay, source, and name the same route. Since the excluded principles do not occur among the carrier rows, no accepted uniform-modulus read can factor through them. □

Theorem 2299.4 (Compact uniform modulus finite-net extraction)**.** *Let $M = (X, K, F, L, Q, U, H, C, P, N)$ be an accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier. Every finite-net extraction read exported by M exposes the same finite* ProbeBundle *lower-bound fold consumed by chapter 4118: the read must pass through X, K, F, L, Q before the* $\mathsf{UniformModulus}^{\uparrow}$ *output row U can be consumed. No selected*

subcover, selected Lebesgue number, host minimum, sequence extraction, or ambient completeness row is available to the extraction.

Proof. Project the carrier of Definition 2299.1. The rows $X, K, F, L, Q, U, H, C, P, N$ are the only displayed rows available to an accepted finite-net extraction read.

Use Definition 2299.2. It fixes source lock among X, K, F, finite coverage through K, rational lower-bound exactness through Q, and non-escape from selected compactness data.

Apply Definition 2299.3. The finite fold forces the read through K, F, L, Q before the output row U is visible, matching the lower-bound fold consumed by chapter 4118.

It remains to rule out alternate extraction data. The carrier classifier has no coordinate for a selected subcover, selected Lebesgue number, host minimum, sequence extraction, or ambient completeness row, so row exhaustion prevents any accepted extraction read from leaving the displayed finite route. □

Theorem 2299.5 (Compact uniform modulus RealSeparability handoff)**.** *A* $\mathsf{RealSeparability}^{\uparrow}$ *or compact-metric consumer may read an accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier* $M = (X, K, F, L, Q, U, H, C, P, N$ *only through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{Compact}^{\uparrow},\ \mathsf{Continuous}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The read exposes the metric source, compact finite-net row, pointwise continuity row, finite fold ledger, rational comparison row, uniform-modulus output, transport, replay, provenance, and local naming. It does not expose a selected Lebesgue number, selected subcover, host modulus function, sequence extraction principle, or ambient compact-metric completeness theorem.

Proof. Project the carrier of Definition 2299.1. The rows $X, K, F, L, Q, U, H, C, P, N$ give the metric source, compact finite-net row, pointwise continuity row, local fold, rational comparison, output, transport, replay, provenance, and local naming.

Use the obligation surface of Definition 2299.2. It fixes source lock, finite coverage, rational lower-bound exactness, output admission, and non-escape from selected compactness data as local NameCert obligations.

Finally apply the finite-net fold theorem Definition 2299.3. The fold forces every exported uniform-modulus read through $X, K, F, L, Q, U, H, C, P, N$, so a RealSeparability or compact-metric consumer receives only the displayed finite route and no selected Lebesgue number or host modulus. □

Definition 2299.6 (Compact uniform modulus Real route ledger)**.** *The* $\mathsf{CompactUniformModulus}^{\uparrow}$ *Real route ledger is the finite route*

$$X, K, F, L, Q, U, H, C, P, N$$

from compact finite-net rows and $\mathsf{Continuous}^{\uparrow}$ *pointwise moduli through* $\mathsf{Rat}^{\uparrow}$ *lower-bound folds into the* $\mathsf{UniformModulus}^{\uparrow}$ *output row and finally into* $\mathsf{Real}^{\uparrow}$ *consumer reads. It records only MetricUp, CompactUp, ContinuousUp, UniformModulusUp, RatUp, BHist, hsame, Cont, Pkg, and NameCert rows; it has no coordinate for a selected Lebesgue number, host modulus function, sequential compactness principle, selected subcover, or ambient Real completeness.*

Theorem 2299.7 (Compact uniform modulus Real route exhaustion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing compact uniform modulus read factors through the ledger of Definition 2299.6. Equivalently, the read passes through* $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg,

and NameCert *rows before it reaches a Real consumer. It exposes no selected Lebesgue number, host modulus function, sequential compactness argument, selected subcover, or ambient Real completeness.*

Proof. Project the carrier of Definition 2299.1. Its rows $X, K, F, L, Q, U, H, C, P, N$ are exactly the metric source, compact finite-net data, pointwise continuity data, local fold ledger, rational comparison row, uniform-modulus output, transport, replay, provenance, and local naming.

Use the NameCert obligations of Definition 2299.2. They bind source lock, finite coverage, rational lower-bound exactness, output admission, and ledger non-escape to those rows, so a consumer cannot bypass the finite packet.

Apply Definition 2299.3. The fold routes every exported modulus through the compact finite-net row, the pointwise continuity row, and the rational lower-bound computation before the $\mathsf{UniformModulus}^{\uparrow}$ output is visible.

Finally use Definition 2299.5. Its handoff already identifies the Real-facing route through the same finite rows. Replacing the RealSeparability consumer by a $\mathsf{Real}^{\uparrow}$ consumer changes only the terminal read, not the finite ledger that supplies it. □

Theorem 2299.8 (Compact uniform modulus vision route)**.** *Every accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier* $M = (X, K, F, L, Q, U, H, C, P, N)$ *realizes the finite* ProbeBundle *lower-bound fold described by chapter 9237. The route passes through* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{Continuous}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it exposes no selected Lebesgue number, selected subcover, sequence extraction, choice principle, host modulus, or ambient completeness datum.*

Proof. Start with Definition 2299.3. It forces every exported uniform-modulus read through the compact finite-net row, the pointwise continuity row, the fold ledger, the rational comparison row, and the output row.

Use the Real route ledger of Definition 2299.6. That ledger identifies the BEDC row names as $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Finally apply Definition 2299.7. The exhaustion theorem rules out selected Lebesgue numbers, selected subcovers, sequence extraction, host moduli, and ambient completeness rows, so the displayed route is exactly the finite ProbeBundle lower-bound fold named by the vision chapter. □

Theorem 2299.9 (Compact uniform continuity Lebesgue-number factorization)**.** *Let* $M = (X, K, F, L, Q, U, H, C,$ *be an accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier. Whenever the compact finite-net row* K *and pointwise-continuity row* F *are presented as a finite cover, every exported* $\mathsf{UniformModulus}^{\uparrow}$ *row* U *factors through the finite Lebesgue-number route of chapter 2301. The factorization is through displayed finite rows: compact finite-net data, pointwise-radius cover rows, rational lower-bound ledger, and the uniform-modulus output route. It does not pass through a selected host Lebesgue number, selected subcover, host minimum, sequence extraction, or ambient completeness theorem.*

Proof. Start with the compact finite-net fold of Definition 2299.3. The exported uniform-modulus read must pass through $X, K, F, L, Q, U, H, C, P, N$, so the finite cover data from K and the pointwise-radius data from F are already displayed before U is visible.

When K and F are presented as a finite cover, read the finite Lebesgue-number route of chapter 2301. Its positive-radius ledger consumes compact metric and metric rows, the displayed cover or pointwise-radius family, and the finite $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ radius row before handing the result to $\mathsf{UniformModulus}^{\uparrow}$.

The rational lower-bound ledger Q is the comparison row by which the compact uniform-modulus packet consumes that finite positive-radius data. Thus the output route factors through

the same displayed finite rows used by the Lebesgue-number packet, and no selected host Lebesgue number is introduced.

Finally apply Definition 2299.2. The non-escape row excludes selected Lebesgue numbers, selected subcovers, host minima, sequence extraction, and ambient completeness, so the factorization remains internal to the finite BEDC ledger. □

Theorem 2299.10 (Compact uniform continuity vision marker route)**.** *Every accepted* $\mathsf{CompactUniformModulus}^{\uparrow}$ *carrier realizes the concrete route named by the compact-uniform-continuity vision marker for chapter 2299. The route factors through* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold,* $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, with no selected host Lebesgue number, selected subcover, sequence compactness argument, external modulus selector, or ambient function-space equality.*

Proof. Read the carrier rows of Definition 2299.1 through the compact finite-net route of Definition 2299.3. The compact row and metric source combine as the $\mathsf{CompactMetric}^{\uparrow}$ access surface consumed by compact-uniform readers, while the pointwise continuity row supplies the $\mathsf{ContinuousMap}^{\uparrow}$ modulus data.

The rational comparison row is the $\mathsf{Rat}^{\uparrow}$ lower-bound fold. By Definition 2299.9, this fold is precisely the finite radius route through which the $\mathsf{UniformModulus}^{\uparrow}$ output is read.

Finally apply Definition 2299.2 and Definition 2299.7. They keep the route inside the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows and exclude selected host compactness, host modulus choice, and ambient equality coordinates. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2299.1, Definition 2299.2, and Definition 2299.3 seed the finite uniform-modulus packet over metric, compact finite-net, pointwise continuity, local fold, rational comparison, output, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected Lebesgue numbers, sequential compactness, host modulus functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source lock, finite-net coverage, pointwise-modulus coverage, rational lower-bound exactness, uniform-modulus output, and ledger non-escape for BEDC.Derived.CompactUniformModulusUp.

Constructive story: A $\mathsf{CompactUniformModulus}^{\uparrow}$ packet is generated from a MetricUp source, CompactUp finite-net rows, ContinuousUp pointwise rows, a local finite fold ledger, RatUp comparisons, a UniformModulusUp output, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2300

A Concrete Naming Certificate for $\mathsf{MichaelSelection}^{\uparrow}$

Following Michael's selection theorem in the paracompact closed-convex reading, the $\mathsf{MichaelSelection}^{\uparrow}$ packet records the finite BEDC surface by which a lower-semicontinuous closed-convex value assignment exports local continuous selection data. It sits over the paracompact cover ledger of chapter 1963, the lower-semicontinuous epigraph and value-row surface of chapter 1536, the convex target rows of chapter 208, the continuous-map handoff of chapter 130, and the metric observation boundary of chapter 57. The packet names only a finite selection-surface ledger: domain cover and paracompact refinement rows, lower-semicontinuous closed-convex value rows, local finite selection rows, continuous-map handoff, componentwise transport, replay, provenance, and local naming. It does not import an arbitrary choice function, a host global selector, quotient equality of maps, Banach-space completeness, or ambient point-set topology.

Definition 2300.1 (Michael selection carrier)**.** *A* $\mathsf{MichaelSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, V, K, L, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain row carrying the* $\mathsf{Paracompact}^{\uparrow}$ *cover and refinement ledger used by the selection surface.*
2. *V is the value-assignment row read through* $\mathsf{LowerSemicontinuous}^{\uparrow}$*, exposing finite lower-semicontinuity windows for the closed-valued relation.*
3. *K is the closed-convex target row, read through* $\mathsf{ConvexSet}^{\uparrow}$ *together with the metric observation rows used by the target carrier.*
4. *L is the local finite selection ledger, matching each displayed cover cell from D with a value row from V and a convex target witness from K.*
5. *S is the finite stitching row that records compatibility of the local selection ledger on displayed cover refinements.*
6. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row exported only after D, V, K, L, S have been displayed.*
7. *H records componentwise* hsame *transport for the domain, value, convex-target, local selection, stitching, handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the route*

$$D, V, K, L, S \longmapsto F \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{MichaelSelection}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MichaelSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, V, K, L, S, F, H, C, P, N$. It has no coordinate for an arbitrary selector, a host-level partition of unity, quotient equality of continuous maps, global choice over all fibres, ambient Banach completeness, or point-set topological semantics.

Lean4 ✓ `BEDC.Derived.MichaelSelectionUp.MichaelSelectionTasteGate single carrier alignment`

Theorem 2300.2 (Michael selection NameCert obligations)**.** *For every accepted* $\mathsf{MichaelSelection}^{\uparrow}$ *carrier $M = (D, V, K, L, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MichaelSelection}^{\uparrow}}$ *surface consists exactly of carrier admission, paracompact domain-cover exposure through D, lower-semicontinuous closed-value exposure through V, closed-convex target exposure through K, local finite selection ledger exposure through L, displayed stitching through S, continuous-map handoff through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape from arbitrary choice, host selectors, quotient map equality, ambient Banach completeness, and point-set topological semantics.*

Proof. Project the carrier of Definition 2300.1. The rows $D, V, K, L, S, F, H, C, P, N$ are the whole packet, so carrier admission and the visible obligation rows are read directly from the displayed tuple.

The analytic surface is the finite route D, V, K, L, S. The paracompact row supplies the cover and refinement ledger, the lower-semicontinuous row supplies value-window exposure, the convex row supplies closed-convex target witnesses, and the local ledger records the finite selection data over displayed cover cells.

Only after those rows are present may the continuous-map row F be read. The structural rows H, C, P, N transport, replay, source, and name the same route, so the excluded choice and host-topology principles have no carrier coordinate and cannot be obtained from the local NameCert. □

Theorem 2300.3 (Michael selection continuous handoff boundary)**.** *Every continuous-map consumer of an accepted* $\mathsf{MichaelSelection}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Paracompact}^{\uparrow},\ \mathsf{LowerSemicontinuous}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The $\mathsf{ContinuousMap}^{\uparrow}$ *handoff is terminal for the packet: it is reached only after the paracompact cover row, lower-semicontinuous closed-convex value row, local finite selection ledger, and stitching row have been displayed.*

Proof. Inspect the carrier order in Definition 2300.1. The continuous-map row F occurs downstream of the domain, value, convex-target, local selection, and stitching rows.

The NameCert obligations of Definition 2300.2 identify those rows as the public selection surface. A consumer therefore cannot cite F without also exposing D, V, K, L, S and the structural rows H, C, P, N.

The absent coordinates are exactly the host selector and choice principles excluded by the carrier. Hence the handoff exports a finite BEDC selection surface rather than an arbitrary global selection theorem. □

Falsifiable prediction: Every accepted $\mathsf{MichaelSelection}^{\uparrow}$ continuous-map read factors through a displayed $\mathsf{Paracompact}^{\uparrow}$ domain cover row, $\mathsf{LowerSemicontinuous}^{\uparrow}$ closed-value row, $\mathsf{ConvexSet}^{\uparrow}$ target row, local finite selection ledger, stitching row, terminal $\mathsf{ContinuousMap}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: paracompact, lowersemicontinuous, convexset, continuousmap, metric: $\mathsf{MichaelSelection}^{\uparrow}$ is not bijective to any listed sibling because it binds domain cover refinement, lower-semicontinuous closed-convex values, local finite selection rows, stitching, continuous-map export, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MichaelSelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MichaelSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2300.1, Definition 2300.2, and Definition 2300.3 seed the finite Michael selection surface over paracompact cover rows, lower-semicontinuous closed-convex value rows, local finite selection rows, stitching, continuous-map handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MichaelSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary selection, host partition-of-unity construction, quotient map equality, ambient Banach completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paracompact cover exposure, lower-semicontinuous closed-convex value exposure, finite selection-ledger exactness, continuous-map handoff boundary, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.MichaelSelectionUp.

Constructive story: A $\mathsf{MichaelSelection}^{\uparrow}$ packet is generated from a paracompact domain cover row, a lower-semicontinuous closed-value row, a closed-convex target row, local finite selection rows, stitching rows, a continuous-map handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2301

A Concrete Naming Certificate for LebesgueNumberLemma$^{\uparrow}$

Following Lebesgue's finite-radius lemma for compact metric covers, the LebesgueNumberLemma$^{\uparrow}$ packet records the theorem-facing route that turns a compact metric finite-cover request into a positive radius ledger. It refines, rather than duplicates, the FiniteLebesgueNumber$^{\uparrow}$ surface of chapter 452: the latter is the finite radius ledger, while this packet names the compact-cover-to-ledger lemma route consumed by compact uniform continuity. It sits over the compact metric source of chapter 129, the metric observation rows of chapter 57, the continuous-map pointwise-radius surface of chapter 130, the FiniteLebesgueNumber$^{\uparrow}$ positive-radius ledger, and the UniformModulus$^{\uparrow}$ consumer of chapter 369.

Definition 2301.1 (Lebesgue number lemma carrier)**.** *A* LebesgueNumberLemma$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, F, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, including the finite-net and located coverage data used by the cover request.*
2. *M is the* Met$^{\uparrow}$ *distance-observation row through which cover radii and ball-containment checks are read.*
3. *F is the displayed finite cover or pointwise-radius family row, read through* ContinuousMap$^{\uparrow}$ *when the consumer is compact uniform continuity.*
4. *R is the positive* FiniteLebesgueNumber$^{\uparrow}$ *radius ledger produced for the displayed finite cover row.*
5. *U is the downstream* UniformModulus$^{\uparrow}$ *handoff row that may consume R after the compact and metric rows have been exposed.*
6. *H records componentwise* hsame *transport for K, M, F, R, U, P, N.*
7. *C records displayed* Cont *replay from compact metric finite nets and metric ball rows to the finite Lebesgue-number ledger and then to the uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* LebesgueNumberLemma$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LebesgueNumberLemma}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, M, F, R, U, H, C, P, N$. It has no coordinate for an arbitrary open-cover compactness theorem, selected subcover choice, host real minima, external Lebesgue numbers, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LebesgueNumberLemmaUp.LebesgueNumberLemmaTasteGate single carrier alignment`

Theorem 2301.2 (Lebesgue number lemma NameCert obligations)**.** *For every accepted* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *carrier $L = (K, M, F, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, metric observation row, displayed finite cover or pointwise-radius family, finite Lebesgue-number positive radius ledger, uniform-modulus handoff, componentwise transport, continuation replay, package provenance, and local naming; compact-source lock between K and M; cover-to-ledger exactness through F and R; uniform-modulus handoff through U; and non-escape from arbitrary open covers, selected subcovers, host minima, external Lebesgue numbers, and ambient completeness.*

Proof. Project the carrier of Definition 2301.1. The displayed tuple contains exactly the compact source, metric observation, finite cover, positive radius ledger, uniform handoff, and structural rows.

The finite radius is not selected externally. It is the row R of the carrier, read after the compact metric and metric rows have exposed the finite coverage data for F.

The rows H, C, P, N transport, replay, source, and name this same route. Since no host cover or external minimum coordinate exists in the carrier, the local NameCert surface exports only the finite BEDC lemma packet. □

Theorem 2301.3 (Lebesgue number lemma uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *carrier reaches* $\mathsf{UniformModulus}^{\uparrow}$ *only through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{FiniteLebesgueNumber}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The theorem-facing row is the route from compact finite-cover data to the positive finite Lebesgue-number ledger R; the ledger itself remains the sibling surface of chapter 452.

Proof. Read K, M, F first. These rows display the compact metric source, the metric ball observations, and the finite cover or pointwise-radius family that generates the cover request.

Then read R. By the obligations of Definition 2301.2, this is the only positive radius ledger exported by the packet, and it is exactly the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ row for the displayed cover data.

Finally the uniform-modulus handoff U may consume that ledger through H, C, P, N. The route therefore refines the existing finite-ledger packet without creating a synonym for it. □

Theorem 2301.4 (Lebesgue number lemma finite-net route)**.** *Every finite-net read used by an accepted* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *packet factors through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{Met}^{\uparrow}$ *ball-observation row, the displayed finite cover or pointwise-radius family, the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger, and the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. It cannot be replaced by arbitrary open-cover compactness or a selected subcover.*

Proof. Project the carrier rows of Definition 2301.1. A finite-net read is visible only through $K, M, F, R, U, H, C, P, N$.

The compact source row K supplies the finite-net and located coverage data, while M supplies the metric ball observations used to test the displayed cover or pointwise-radius row F. The positive radius ledger R is then the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ row for that displayed data.

The existing NameCert obligations in Definition 2301.2 and the uniform-modulus handoff of Definition 2301.3 compose this route before U is available to compact-uniform-continuity consumers.

Since arbitrary open-cover compactness and selected subcovers are not coordinates of $K, M, F, R, U, H, C, P, N$, no finite-net read in the packet can be replaced by those host principles. □

Theorem 2301.5 (Lebesgue number lemma obligation-closure package)**.** *The carrier, NameCert obligations, uniform-modulus handoff, and finite-net route of an accepted* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *packet assemble the full obligation surface over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *In carrier terms this surface is exactly the projection of*

$$K, M, F, R, U, H, C, P, N,$$

and it excludes arbitrary open-cover compactness and selected subcovers.

Proof. Start with the carrier of Definition 2301.1. Its rows $K, M, F, R, U, H, C, P, N$ are the compact metric source, metric observation, displayed cover or pointwise-radius family, finite Lebesgue-number ledger, uniform-modulus handoff, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations theorem, Definition 2301.2. It records carrier admission, compact-source lock, cover-to-ledger exactness, uniform-modulus handoff, transport, replay, provenance, and naming over precisely those rows.

Use the uniform-modulus handoff and finite-net route, Definition 2301.3 and Definition 2301.4. They keep every compact-uniform-continuity and finite-net consumer on the same displayed route through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, and $\mathsf{UniformModulus}^{\uparrow}$.

Since neither arbitrary open-cover compactness nor selected subcovers occur as coordinates of $K, M, F, R, U, H, C, P, N$, the obligation package closes the displayed BEDC surface without exporting those host principles. □

Falsifiable prediction: Every accepted $\mathsf{LebesgueNumberLemma}^{\uparrow}$ compact-uniform-continuity read must expose a $\mathsf{CompactMetric}^{\uparrow}$ source row, $\mathsf{Met}^{\uparrow}$ ball rows, a $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-radius family when present, a $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ positive radius ledger, a terminal $\mathsf{UniformModulus}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, metric, continuousmap, finite-lebesgue-number, uniformmodulus: $\mathsf{LebesgueNumberLemma}^{\uparrow}$ is not bijective to any listed sibling because it binds the compact-cover-to-positive-radius theorem route between the compact metric source and the finite Lebesgue-number ledger before a UniformModulus consumer can read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberLemma}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2301.1, Definition 2301.2, and Definition 2301.3 seed the finite compact-cover-to-positive-radius lemma route consumed by compact uniform continuity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host minima, external Lebesgue numbers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source lock, cover-to-ledger exactness, uniform-modulus handoff, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LebesgueNumberLemmaUp.

Constructive story: A LebesgueNumberLemma$^{\uparrow}$ packet is generated from a compact metric source row, metric ball observations, a finite cover or pointwise-radius family, a FiniteLebesgueNumber positive radius ledger, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2302

A Concrete Naming Certificate for $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$

Following Banach and Steinhaus, the $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ packet records the finite NameCert surface by which a pointwise bounded family of bounded linear operators exposes one uniform operator-norm ledger. It sits over the Banach source row of chapter 90, the bounded-operator family surface of chapter 1644, the scalar and norm rows of chapter 38 and chapter 1500, and the finite observation discipline already used by Banach-facing consumer packets.

Definition 2302.1 (Uniform boundedness principle carrier)**.** *A* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, F, W, L, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *source row, including the normed-space and complete-metric observations used by the operator family.*
2. *F is the finite family row of* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *packets over the same Banach source and displayed target Banach rows.*
3. *W is the pointwise boundedness window ledger: every inspected source endpoint and every inspected operator row has a displayed* $\mathsf{Real}^{\uparrow}$ *bound read through the family surface.*
4. *L is the single uniform norm-bound ledger, read through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{NormedRealLine}^{\uparrow}$*, and attached to the same family row F.*
5. *R records the Banach source rows, bounded-linear-operator rows, pointwise windows, and uniform bound as one route rather than as unrelated observations.*
6. *H records componentwise* hsame *transport for B, F, W, L, R, P, N.*
7. *C records displayed* Cont *replays from Banach source reads to operator-family reads, from pointwise boundedness windows to the uniform norm-bound ledger, and from the Real-facing bound to the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$*,* $\mathsf{Banach}^{\uparrow}$*,* $\mathsf{BoundedLinearOperator}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{NormedRealLine}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, F, W, L, R, H, C, P, N$. It has no coordinate for an arbitrary family of host operators, a selected point outside the displayed windows, a quotient equality of functions, countable choice, or an ambient completeness theorem beyond the Banach row already carried in B.

Lean4 ✓ `BEDC.Derived.UniformBoundednessPrincipleUp.UniformBoundednessPrincipleTasteGate single carrier alignment`

Theorem 2302.2 (Uniform boundedness NameCert obligations)**.** *For every accepted* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *carrier $U = (B, F, W, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach source, bounded-operator family, pointwise boundedness windows, uniform norm-bound ledger, route row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming; source lock between the Banach row and every operator-family member; pointwise-window exhaustion through W; uniform-bound exactness through L; and non-escape from host operator families, uninspected points, quotient function equality, countable choice, and ambient completeness outside B.*

Proof. Project the carrier of Definition 2302.1. The rows $B, F, W, L, R, H, C, P, N$ are the whole packet, so each local obligation is read from the displayed tuple.

The source lock is the route row R. It ties the $\mathsf{Banach}^{\uparrow}$ source used by F to the pointwise boundedness windows in W and to the uniform norm-bound ledger L.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier has no coordinate for the excluded host principles, no accepted consumer can recover them from the local NameCert surface. □

Theorem 2302.3 (Uniform boundedness Banach handoff)**.** *Every Banach-facing consumer of an accepted* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *carrier factors through the ordered route*

$$\mathsf{Banach}^{\uparrow},\quad \mathsf{BoundedLinearOperator}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{NormedRealLine}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The exported datum is the uniform operator-norm ledger L attached to the family row F, not a host-level Banach-Steinhaus theorem, an external operator-family selector, or an unrestricted completeness principle.

Proof. Read the carrier order in Definition 2302.1. A consumer first sees the Banach source B and bounded-operator family F, then the pointwise boundedness windows W, and only then the uniform norm-bound ledger L.

Apply the NameCert obligations of Definition 2302.2. They require the pointwise windows and uniform ledger to be tied by R and transported only through H, C, P, N.

Thus every downstream Banach read receives the displayed finite route and no additional host operator family, countable choice principle, quotient function equality, or ambient completeness row. □

Theorem 2302.4 (Uniform boundedness window-to-ledger dependency)**.** *Every public uniform-bound read of an accepted* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *carrier must pass through the Banach source row of chapter 90, the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *family row of chapter 1644, the pointwise windows W, the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{NormedRealLine}^{\uparrow}$ *bound ledger L, and the route row R before it reaches the local* $\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}$ *row. In particular, for $U = (B, F, W, L, R, H, C, P, N)$, the ledger L is not public as an isolated real bound; it is readable only through the dependency route B, F, W, R and the structural replay rows H, C, P, N.*

Proof. Project the carrier of Definition 2302.1. The tuple displays $B, F, W, L, R, H, C, P, N$, so a read of L occurs inside the same packet that carries the Banach source, the bounded-operator family, and the pointwise windows.

The route row R is the dependency witness. It records the Banach source rows, bounded-linear-operator rows, pointwise windows, and uniform bound as one route rather than unrelated observations, so L cannot be read without first exposing B, F, W, R.

The structural rows H, C, P, N preserve and replay this route before the local NameCert is reached. Thus a public uniform-bound read is a window-to-ledger dependency read, not a direct export of an isolated host-level Banach-Steinhaus conclusion. □

Theorem 2302.5 (Uniform boundedness pointwise-to-uniform route)**.** *For every accepted* UniformBoundednessPrin *carrier $U = (B, F, W, L, R, H, C, P, N)$, any public uniform bound for the bounded-operator family F over the Banach source B must be obtained from the displayed pointwise windows W over that same B and F, then routed through the uniform ledger L, the dependency row R, componentwise transport H, continuation replay C, package provenance P, and the local naming row N. No accepted read may export L as a uniform bound from a different source, a different operator family, an uninspected pointwise window, quotient equality of maps, countable choice, or ambient Banach completeness outside the carried row B.*

Proof. Project the carrier of Definition 2302.1. The rows B, F, W are the only source, family, and pointwise-window coordinates available to the packet, while L is the only uniform norm-bound ledger.

Apply Definition 2302.4. It forces every public read of L through the same Banach source, bounded-operator family, pointwise windows, and dependency row R before the local certificate is reached.

The structural rows H, C, P, N then transport, replay, source, and name that same route. Since the carrier has no coordinate for alternate sources, alternate families, hidden pointwise witnesses, quotient map equality, countable choice, or ambient completeness outside B, the public uniform bound is exactly the displayed pointwise-to-uniform route. □

Falsifiable prediction: Every accepted $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ consumer must expose a $\mathsf{Banach}^{\uparrow}$ source row, a $\mathsf{BoundedLinearOperator}^{\uparrow}$ family row, pointwise boundedness windows, one $\mathsf{Real}^{\uparrow}$ and $\mathsf{NormedRealLine}^{\uparrow}$ uniform norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the uniform bound is public.

Independence witness: banach, boundedlinearoperator, real, normed-real-line: $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a Banach source, a bounded-operator family, pointwise boundedness windows, a single uniform norm-bound ledger, route data, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}$ **.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2302.1, Definition 2302.2, and Definition 2302.3 seed the finite Banach-Steinhaus route through Banach source rows, bounded-operator family rows, pointwise boundedness windows, and one uniform norm-bound ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformBoundednessPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary host operator families, selected pointwise witnesses outside the displayed windows, quotient equality of maps, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source lock, pointwise-window exhaustion, uniform-bound exactness, Banach handoff boundary, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.UniformBoundednessPrincipleUp.

Constructive story: A UniformBoundednessPrinciple$^{\uparrow}$ packet is generated from a Banach source row, a bounded-linear-operator family row, pointwise boundedness windows, a uniform RealUp and NormedRealLine norm-bound ledger, a route row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2303

A Concrete Naming Certificate for $\mathsf{MetrizableCompletion}^{\uparrow}$

Following Kelley and Bourbaki on metrizable topological spaces and completion, the $\mathsf{MetrizableCompletion}^{\uparrow}$ packet records the finite BEDC route by which a topology equipped with a displayed metrization can be read by the existing metric-completion surface of chapter 1506. It sits after the $\mathsf{Topology}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ carrier rows, consumes the $\mathsf{MetricCompletion}^{\uparrow}$ branch and the $\mathsf{CompleteMetric}^{\uparrow}$ seal, and exposes only the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback path needed for completion-facing consumers.

Definition 2303.1 (MetrizableCompletion carrier)**.** *A* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, M, Q, K, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *row, including the open-set classifier, neighborhood interface, and finite continuity rows visible to the packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *metrization row on the same carried histories, with a displayed comparison that says the topology read through T is the topology generated by the metric read through M.*
3. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *branch consumed from chapter 1506, including its Cauchy-filter or Cauchy-net evidence route.*
4. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row for the completed metric-facing object, read only after Q has exposed the displayed completion branch.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row used to name the finite observation windows inspected by the completion route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for numerical tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint-seal row used when distance or convergence witnesses are compared.*
8. *H records componentwise* hsame *transport for the topology, metrization, metric-completion branch, complete-metric seal, stream, readback, endpoint, replay, provenance, and local naming rows.*

9. *C records the displayed* Cont *replay from topology admission through metrization, completion-branch admission, complete-metric sealing, stream/readback inspection, and endpoint comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{MetrizableCompletion}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, a selected representative of a Cauchy name, host topological equality, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetrizableCompletionUp.MetrizableCompletionTasteGate single carrier alignment`

Theorem 2303.2 (MetrizableCompletion NameCert obligations)**.** *For an accepted* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier $Z = (T, M, Q, K, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology and metrization rows, compatibility of the topology row with the metric-generated topology, admission of the metric-completion branch Q, sealing by the complete-metric row K, stream/readback readiness through S and R, endpoint comparison through E, componentwise transport through H, replay through C, provenance through P, and ledger non-escape for the excluded host-side principles.*

Proof. Carrier admission is projection from Definition 2323.1. The tuple lists the topology, metrization, metric-completion branch, complete-metric seal, stream, readback, endpoint, transport, replay, provenance, and local naming rows.

The topology-metrization compatibility is local to T and M. A consumer may read open-set or neighborhood data only through the displayed metric generator row, and the packet gives no separate host-topology equality coordinate.

The completion-facing part then passes through Q and K. The row Q is exactly the metric-completion interface of chapter 1506; K seals the resulting read as complete-metric evidence only after that branch is displayed.

Finally S, R, and E provide the finite numerical route. Transport, replay, provenance, and local naming preserve the same displayed packet, so quotient completion equality, selected representatives, countable choice, open-cover compactness, host topological equality, and $\mathsf{Real}^{\uparrow}$ completeness cannot be exported. □

Theorem 2303.3 (MetrizableCompletion Cauchy seal)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier factors through the finite Cauchy seal*

$$T \longmapsto M \longmapsto Q \longmapsto K \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

In particular, a consumer cannot read the topological completion before the metrization row, cannot read complete-metric evidence before the $\mathsf{MetricCompletion}^{\uparrow}$ *branch, and cannot compare convergence data before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed.*

Proof. Start with the topology row T. The packet admits it only together with the metrization row M, so a topological completion request first becomes a metric-facing request over the same carried histories.

Next apply the metric-completion branch. The row Q is the sole route to completion-facing data, and chapter 1506 already forces such data through a displayed Cauchy-filter or Cauchy-net branch.

The complete-metric seal K is downstream of Q. It does not select limits or quotient Cauchy names; it records that the displayed completion branch has the complete-metric evidence needed by the consumer.

The numerical seal is read last. Stream windows S precede regular-rational readback R, and the terminal endpoint comparison E is the only $\mathsf{Real}^{\uparrow}$ row offered to convergence comparisons. The transport, replay, provenance, and local naming rows retain this order, so no omitted host-side principle can enter the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetrizableCompletion}^{\uparrow}$ packet is generated from a topology row, a metrizing metric row, the metric-completion branch, a complete-metric seal, StreamName windows, RegSeqRat readback, a terminal RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2323.1, Definition 2323.2, and Definition 2323.3 seed the finite metrizable-completion surface over topology, metric, completion, complete-metric, stream, readback, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host topological equality, open-cover compactness, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, topology-metrization compatibility, metric-completion branch routing, complete-metric sealing, Cauchy-seal exactness, and ledger non-escape over the same displayed packet.

Chapter 2304

A Concrete Naming Certificate for $\mathsf{EffectiveMetricCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion, the $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ packet records the finite BEDC route by which located metric data is completed through explicit approximation schedules rather than quotient names or selected representatives. It sits between the metric-completion surface of chapter 1506, the located metric surface of chapter 1533, and the Real-and-Completion route through chapter 126, chapter 301, and chapter 38.

Definition 2304.1 (Effective metric completion carrier)**.** *An* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (M, L, A, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *branch consumed from chapter 1506, read only as a displayed completion route.*
2. *L is the* $\mathsf{LocatedMetric}^{\uparrow}$ *source row from chapter 1533, including the located distance comparisons visible to the packet.*
3. *A is an effective approximation and modulus ledger, listing finite precision requests, response windows, and the monotone schedule by which later requests refine earlier windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the finite observation windows inspected by the completion route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational regular-sequence data emitted by the approximation ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row used for distance and convergence comparisons after S and R have been displayed.*
7. *H records componentwise* hsame *transport for the completion branch, located metric source, approximation ledger, stream windows, readback, seal, replay, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from located metric admission through effective approximation, stream inspection, readback, and terminal Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{EffectiveMetricCompletion}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, a selected Cauchy representative, host metric equality, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.EffectiveMetricCompletionUp.EffectiveMetricCompletionTasteGate single carrier alignment`

Theorem 2304.2 (EffectiveMetricCompletion NameCert obligations)**.** *For an accepted* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ *carrier* $Y = (M, L, A, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EffectiveMetricCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion branch and located metric source, effective approximation ledger admission, stream-window readiness, regular-rational readback, terminal Real sealing, componentwise transport, displayed replay, provenance, local naming, and ledger non-escape for the excluded host-side principles.*

Proof. Carrier admission is projection from Definition 2304.1. The packet lists the completion branch, located metric source, approximation ledger, stream windows, readback, Real seal, transport, replay, provenance, and local naming rows.

The effective approximation row A is downstream of both M and L. A consumer must therefore read the completion request through the displayed metric-completion branch and the located metric source before it can inspect any finite precision window.

The numerical route is then forced through S, R, E. Stream names provide the finite windows, regular rational sequences provide the readback, and the Real row supplies only the terminal seal for comparisons already exposed by the earlier rows.

Finally H, C, P, N preserve the same displayed route under transport, replay, provenance, and local naming. Since the carrier has no remaining row, quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, and $\mathsf{Real}^{\uparrow}$ completeness cannot be exported. □

Theorem 2304.3 (EffectiveMetricCompletion Real-seal handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ *carrier factors through the finite handoff*

$$M \longmapsto L \longmapsto A \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

In particular, the terminal $\mathsf{Real}^{\uparrow}$ *seal is read only after the metric-completion branch, located metric source, effective approximation ledger, StreamName windows, and RegSeqRat readback have all been displayed.*

Proof. Start with the completion branch M and located source L. The carrier exposes no completion-facing request unless those two rows have already been displayed in the same finite history packet.

Next route through the approximation ledger A. Its finite modulus entries bind each requested precision to a displayed window, so the consumer cannot choose a hidden representative or import a countable-choice schedule.

The handoff to Real data proceeds through S and R. The stream row names the inspected window, and the regular rational row supplies the finite readback consumed by the terminal Real seal.

It remains to check non-escape. Transport, replay, provenance, and local naming keep the same row order, so route exhaustion leaves no coordinate for quotient completion equality, selected representatives, host equality, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveMetricCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ packet is generated from a metric-completion branch, a located metric source, an effective approximation and modulus ledger, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2304.1, Definition 2304.2, and Definition 2304.3 seed the finite effective metric completion surface over located metric data, explicit approximation schedules, stream windows, rational readback, Real sealing, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, approximation-ledger routing, stream-window readiness, RegSeqRat readback coherence, Real-seal handoff, and ledger non-escape over the same displayed packet.

Chapter 2305

A Concrete Naming Certificate for $\mathsf{CompletionTower}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley's separated uniform completion route, the $\mathsf{CompletionTower}^{\uparrow}$ packet records the finite tower read in which a source metric is completed, read through a second completion request, and flattened by displayed idempotence rather than by quotient equality or selected representatives. It sits after the metric-completion surface of chapter 1506, the idempotent Cauchy-completion row of **??**, the uniform-completion boundary of chapter 1473, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a finite route for repeated completion reads without importing an ambient completed-object equality.

Definition 2305.1 (Completion tower carrier)**.** *A* $\mathsf{CompletionTower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, F, S, D, I, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row, carrying the displayed distance and tolerance requests before any completion consumer reads the tower.*
2. *F is the first* $\mathsf{MetricCompletion}^{\uparrow}$ *row produced from M.*
3. *S is the second completion-request row, recording the attempted completion of the already completed presentation.*
4. *D is the dense embedding row from the source metric into the first completion row.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *flattening row, recording that the second completion request reuses the first completion surface.*
6. *Q is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row, making equality-facing reads pass through separated data rather than host equality of completed points.*
7. *R is the regular readback and Real-facing route through* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.
8. *H records componentwise* hsame *transport for the metric, completion, embedding, flattening, separated comparison, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M \longmapsto F \longmapsto S \longmapsto D, I, Q, R.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompletionTower}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionTower}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionTowerUp.CompletionTowerTasteGate single carrier alignment`

Theorem 2305.2 (Completion tower NameCert obligations)**.** *For every accepted* $\mathsf{CompletionTower}^{\uparrow}$ *carrier* $T = (M, F, S, D, I, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionTower}^{\uparrow}}$ *surface consists exactly of source metric exposure, first completion exposure, second completion-request exposure, dense embedding, idempotence flattening, separated comparison, regular and Real-facing readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from quotient completion equality, selected representatives, countable choice, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2305.1. The source metric row M, first completion row F, second completion-request row S, dense embedding row D, idempotence row I, separated comparison row Q, and readback row R are the only analytic coordinates in the packet.

The tower route is then read in order. The first completion consumer reads F from M, the second request is displayed as S, and flattening is forced through I rather than through an equality of completed objects. The dense embedding row D and separated comparison row Q keep source and equality-facing reads visible.

Finally H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Since the excluded host-side principles are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted repeated-completion read factors through the source metric, first completion, second completion request, dense embedding, idempotence flattening, separated comparison, regular readback, transport, replay, provenance, and local naming rows.

Independence witness: metriccompletion, cauchy-completion-idempotent, uniformcompletion, regseqrat, real: $\mathsf{CompletionTower}^{\uparrow}$ is not bijective to any one sibling because it jointly carries first completion, second completion request, dense embedding, flattening, separated comparison, readback, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionTower}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompletionTower}^{\uparrow}}$, seed)

Scope closed: Definition 2305.1 and Definition 2305.2 seed the finite repeated-completion tower surface over displayed metric, completion, dense-embedding, idempotence, separated-comparison, readback, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompletionTower}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completed objects, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, repeated-completion flattening, dense-embedding routing, separated comparison, readback stability, transport and replay non-escape, and TasteGate field faithfulness for BEDC.Derived.CompletionTowerUp.

Constructive story: A $\mathsf{CompletionTower}^{\uparrow}$ packet is generated from a source metric row, first completion row, second completion request, dense embedding, idempotence flattening, separated comparison, uniform and regular readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2306

A Concrete Naming Certificate for $\mathsf{CoreStateGraph}^{\uparrow}$

The $\mathsf{CoreStateGraph}^{\uparrow}$ packet realizes the strict partial-aware core-state graph of Definition 9462.1810 as finite NameCert data. It records the 22 state rows, the single-coordinate edge ledger, the 9 component ledger, observed module-generator rows, transport, replay, provenance, and local naming surface used by the DNA-window boundary analysis. When the finite alphabet surface is needed, the packet consumes the displayed alphabet and readback rows of chapter 2260 before adding strict edges and component ledgers. The packet is a finite graph-state ledger for accepted reads, not a mechanistic evolutionary model, not a hidden distance oracle, and not a collapse of ambiguous table rows.

Definition 2306.1 (Core state graph carrier)**.** *A* $\mathsf{CoreStateGraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, E, K, M, T, H, C, P, N)$$

with the following displayed rows:

1. *V is the vertex ledger of the 22 partial-aware core states, retaining ambiguous rows as separate labels.*
2. *E is the strict edge ledger, adding an edge exactly when two state vectors differ in one core coordinate.*
3. *K is the component ledger with 9 connected components and the displayed component-size row.*
4. *M is the observed module-generator row for UGA, UAA/UAG, AGR, AUA, and AAA support moves.*
5. *T is the table-group provenance row tying vertices and components to the comparison-family table identifiers.*
6. *H records componentwise* hsame *transport for V, E, K, M, T and the local naming row.*
7. *C records displayed* Cont *replay of accepted graph reads through vertex, edge, component, module-generator, and table-group rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CoreStateGraph}^{\uparrow}$*, the originating DNA-window strict graph vision,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CoreStateGraph}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, K, M, T, H, C, P, N$. It has no coordinate for collapsed ambiguous rows, unlisted graph edges, a connectedness assertion, a hidden biological mechanism, or a host-side graph oracle.

Lean4 ✓ `BEDC.Derived.CoreStateGraphUp.CoreStateGraphTasteGate single carrier alignment`

Theorem 2306.2 (Core state graph NameCert obligations)**.** *For every accepted* $\mathsf{CoreStateGraph}^{\uparrow}$ *carrier* $G = (V, E, K, M, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CoreStateGraph}^{\uparrow}}$ *surface consists exactly of vertex-ledger exposure, strict edge-ledger exposure, component-ledger exposure, module-generator exposure, table-group provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from collapsed ambiguity, unlisted edges, hidden mechanisms, connectedness import, and host-side graph computation.*

Proof. Project the carrier of Definition 2306.1. The finite graph data are exactly the vertex row V, edge row E, component row K, module-generator row M, and table-group row T.

The strict graph read of Definition 9462.1810 is recovered by reading V before E, then reading K as the component ledger generated from that edge relation. The module-generator row M is separate from the strict edge row and therefore cannot add hidden edges.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since no other coordinate appears in the carrier, accepted reads cannot collapse ambiguous states, import a connectedness theorem, or consult a host graph oracle. □

Theorem 2306.3 (Core state graph finite acceptance route)**.** *Every accepted* $\mathsf{CoreStateGraph}^{\uparrow}$ *read factors through the finite route*

$$V, E, K, M, T \longmapsto H, C, P, N.$$

Thus the read preserves the 22*-state ledger, the strict edge ledger, the* 9*-component ledger, and the module-generator row as separate displayed coordinates.*

Proof. Begin with the carrier projection. The graph-facing rows V, E, K specify the partial-aware state space, strict adjacency relation, and connected-component ledger.

Next read M and T. These rows record module-generator support and table-group provenance, but neither row rewrites the strict edge relation. The structural rows H, C, P, N then transport, replay, source, and name only the same finite coordinates.

The route is therefore exhausted by the displayed packet. A consumer that removes any one of V, E, K, M, T loses a named graph or provenance coordinate, so no accepted read can factor through a smaller hidden graph. □

Falsifiable prediction: Every accepted read of the strict core-state graph factors through the 22-state ledger, strict edge ledger, 9-component ledger, module-generator row, table-group provenance, transport, replay, provenance, and local naming rows.

Independence witness: Nearby finite graph and state-ledger siblings: codon-transition-matrix, dna-four-phase-alphabet-bridge, finite-observation-fold, and finite-observation-confluence. The $\mathsf{CoreStateGraph}^{\uparrow}$ packet is not bijective to any one sibling because it carries the partial-aware vertex ledger, strict edge ledger, component ledger, module-generator support, and table-group provenance simultaneously.

Later Lean work should place Nontrivial $\mathsf{CoreStateGraph}^{\uparrow}$ and FieldFaithful $\mathsf{CoreStateGraph}^{\uparrow}$ instances in the TasteGate surface. Nontriviality must distinguish carriers with the same vertex ledger but different strict edge or component rows, while field faithfulness must preserve $V, E, K, M, T, H, C, P, N$ separately.

Closure status for $\mathsf{NameCert}_{\mathsf{CoreStateGraph}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoreStateGraph}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2306.1, Definition 2306.2, and Definition 2306.3 seed the finite strict graph surface over state, edge, component, module-generator, table-group, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoreStateGraph}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a mechanistic evolutionary model, graph connectedness, hidden distance computation, collapsed ambiguous table rows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, strict edge exposure, component-ledger exposure, module-generator separation, finite route exactness, and TasteGate field faithfulness for BEDC.Derived.CoreStateGraphUp.

Constructive story: A $\mathsf{CoreStateGraph}^{\uparrow}$ packet is generated from a partial-aware vertex ledger, strict edge ledger, component ledger, module-generator support row, table-group provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2307

A Concrete Naming Certificate for MetaCICFrontierDependencyMatrix$^{\uparrow}$

The MetaCICFrontierDependencyMatrix$^{\uparrow}$ packet realizes the MetaCIC open-frontier ledger as finite matrix data. It places the four subject-reduction discharge rows beside the obstruction rows for dependent codomain, confluence, normalization, decidability, and audit-map consumers, so downstream chapters can cite a bounded dependency matrix rather than an informal checklist. The packet reads the open-problem ledger of section 9386.2, the subject-reduction discharge socket of chapter 719, the normalization boundary of chapter 892, and the route classifier of chapter 785.

Falsifiable prediction: Every accepted MetaCICFrontierDependencyMatrix$^{\uparrow}$ row factors through exactly one named positive support row or exactly one named obstruction row before any audit-map consumer reads it.

Independence witness: metacicopenproblemledger, subjectreductiondischargesocket, metacicnormalizationfrontier, subjectreductionrouteclassifier: the open ledger lists frontier questions, the discharge socket carries four positive subject-reduction rows, the normalization frontier names normalization obstructions, and the route classifier separates subject-reduction routes; none is bijective to the finite matrix packet that records positive rows, obstruction rows, and audit consumers together.

Definition 2307.1 (MetaCIC frontier dependency matrix carrier)**.** *A* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, A, L, P, Q, N, D, J, R, H, C, K, E)$$

with the following displayed rows:

1. B, A, L, P *are the four positive subject-reduction discharge rows: beta substitution, application-argument stability, lambda-domain subject reduction, and Pi-domain subject reduction.*
2. Q *is the dependent-codomain obstruction row separating application-argument stability from a closed subject-reduction theorem.*
3. N *is the MetaCIC normalization frontier row, including the reduction and normal-form boundaries read by normalization consumers.*
4. D *is the decidability frontier row for bounded checking and closed type-checking consumers.*
5. J *is the confluence and critical-pair obstruction row read by audit-map consumers.*

6. *R is the route-classifier row assigning each matrix row to positive support, obstruction, or audit consumer.*
7. *H records componentwise* hsame *transport for all positive, obstruction, classifier, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an audit-map request through a selected matrix row and its route-classifier entry.*
9. *K is the* Pkg *provenance over* $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *E is the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these finite rows. It has no coordinate for a closed subject-reduction theorem, global confluence, full normalization, total decidability, or an audit-map result that bypasses the route classifier.

Theorem 2307.2 (MetaCIC frontier dependency matrix obligations)**.** *For every accepted* MetaCICFrontierDepende *carrier $M = (B, A, L, P, Q, N, D, J, R, H, C, K, E)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the four positive subject-reduction rows, dependent-codomain obstruction row, normalization frontier, decidability frontier, confluence boundary, route classifier, transport, replay, provenance, and local naming row;*
2. *row polarity, requiring each matrix row to be classified by R as positive support, obstruction, or audit consumer;*
3. *support factorization, requiring every positive row to factor through one of B, A, L, P before subject-reduction consumers may use it;*
4. *obstruction factorization, requiring every negative frontier row to factor through exactly one of Q, N, D, J before audit-map consumers may cite it;*
5. *nonescape, excluding closed subject reduction, global confluence, full normalization, total decidability, and audit-map closure outside the displayed rows.*

No frontier dependency matrix obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2307.1. The tuple displays the positive subject-reduction rows, obstruction rows, classifier, transport, replay, provenance, and local naming row.

Row polarity is the job of R. Each matrix entry must be read with its route-classifier tag before any consumer treats it as support or obstruction.

Support factorization then follows by exhaustion of the positive rows. The only positive coordinates are B, A, L, P, so a subject-reduction consumer cannot obtain a fifth support row from the matrix.

Obstruction factorization is the matching exhaustion over Q, N, D, J. These rows name the dependent-codomain, normalization, decidability, and confluence boundaries, and the replay row C preserves their route tags for audit consumers.

The nonescape clause inspects K, E and the carrier fields. Since no closed subject-reduction theorem, global confluence theorem, full normalization theorem, total decidability theorem, or completed audit-map result appears among the coordinates, none is exported by the packet. □

Theorem 2307.3 (MetaCIC frontier audit-map route)**.** *Every audit-map consumer of an accepted* $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$ *packet first reads the route classifier R, then either one positive support row among B, A, L, P or one obstruction row among Q, N, D, J, and only then replays the selected dependency through H, C, K, E. Thus the matrix is finite row data for frontier accounting, not a proof of closed subject reduction, confluence, normalization, decidability, or audit-map completion.*

Proof. An audit-map request begins at R, because the carrier gives no untagged dependency coordinate. The classifier chooses the polarity and the named row before the consumer can inspect the row content.

In the positive case, row exhaustion gives one of B, A, L, P. In the obstruction case, row exhaustion gives one of Q, N, D, J. The two cases are disjoint because the route classifier stores their tags before replay.

The structural rows H, C, K, E preserve and name the selected matrix entry. They do not add a closed theorem or a completed audit-map result, so the consumer remains inside finite frontier accounting. □

Theorem 2307.4 (MetaCIC frontier dependency matrix substitution support)**.** *Every beta-substitution or application-argument consumer of an accepted* $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$ *packet first factors through one of the positive support rows B, A, L, P. Every dependent-codomain, normalization, decidability, or confluence consumer first factors through one of the obstruction rows Q, N, D, J. The packet therefore supplies a finite substitution-support route for subject-reduction discharge, but it exports no closed subject-reduction theorem, no global normalization theorem, no total decidability theorem, and no confluence theorem.*

Proof. Project the carrier of Definition 2307.1. The substitution-facing support rows are exactly B, A, L, P, with B carrying beta substitution and A carrying application-argument stability.

The remaining frontier consumers are separated by row polarity. Dependent-codomain reads pass through Q, normalization reads through N, decidability reads through D, and confluence or critical-pair reads through J.

Exhaust the route classifier R. Since every row is tagged before replay through H, C, K, E, a consumer cannot cross from a positive support row into a closed global theorem, and an obstruction row cannot be read as a completed MetaCIC result. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2307.1, Definition 2307.2, and Definition 2307.3 seed the finite MetaCIC frontier matrix over positive support rows, obstruction rows, route-classifier tags, and audit-map consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close subject reduction, confluence, normalization, decidability, audit-map completion, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, row polarity, support factorization, obstruction factorization, audit-map route replay, and nonescape for BEDC.Derived.MetaCICFrontierDependencyMatrixUp.

Constructive story: A $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$ packet is generated from four positive subject-reduction rows, named obstruction rows, a route classifier, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2308

A Concrete Naming Certificate for $\mathsf{NestedIntervalTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{NestedIntervalTheorem}^{\uparrow}$ packet records the constructive nested closed interval theorem as finite BEDC route data. It consumes the retained-window surface of chapter 320, the dyadic endpoint and tolerance ledger of chapter 291, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the interval-existence handoff without asserting quotient reals, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2308.1 (Nested interval theorem carrier)**.** *A* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{NestedInterval}^{\uparrow}$ *row, including the inspected closed rational or dyadic interval windows, endpoint-order rows, width rows, and displayed containment witnesses.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger that records the endpoint comparisons and the diameter-to-tolerance rows used by the retained windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule requested from the lower and upper endpoint rows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite interval windows and dyadic tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval, dyadic, stream, regular readback, Real seal, replay, provenance, and naming rows.*
7. *C records the displayed* Cont *replay from nested interval data through dyadic tolerance control, stream windows, regular readback, and Real seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalTheorem}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite host intersection object, a selected quotient representative, countable choice, trichotomy, a Cauchy-completion quotient, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.NestedIntervalTheoremUp`

Theorem 2308.2 (Nested interval theorem NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *carrier* $K = (I, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed nine-row packet,* $\mathsf{NestedInterval}^{\uparrow}$ *source exposure through* I*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance exactness through* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E*, and the structural transport, replay, provenance, and naming rows* H, C, P, N*.*

No obligation may be discharged by a host interval intersection, selected point, countable choice, trichotomy, quotient real, or ambient completeness principle.

Proof. Project the carrier of Definition 2308.1. The rows $I, D, S, R, E, H, C, P, N$ are the only coordinates visible to the local classifier.

The interval source is I. It exposes only the finite nested-window surface already available from $\mathsf{NestedInterval}^{\uparrow}$: endpoint order, width, containment, and retained-window rows.

The completion-facing route is finite. The ledger D reads the dyadic endpoint and tolerance comparisons, S lists the requested stream windows, R repacks those windows as a regular rational readback, and E is downstream of those rows.

The structural rows do not add mathematical principles. The transports H, continuations C, provenance P, and local certificate N preserve and name the same packet, so the excluded host intersection, choice, quotient, trichotomy, and completeness coordinates remain absent. □

Lean4 ✓ `BEDC.Derived.NestedIntervalTheoremUp.NestedIntervalTheoremTasteGate single carrier alignment`

Theorem 2308.3 (Nested interval theorem Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads the nested interval row I*, the dyadic endpoint and tolerance ledger* D*, the stream-window schedule* S*, the regular-rational row* R*, and only then the Real seal row* E *with its transports, continuations, provenance, and local naming.*

Proof. Start with the Real-facing read. Since E is the only Real seal coordinate in the carrier, every such read must pass through the rows that make E visible.

The source row is I. It supplies the finite nested closed interval windows and containment witnesses, while D supplies the dyadic endpoint comparisons and diameter-to-tolerance rows read from the same windows.

The name construction then passes through S and R. The finite stream schedule selects the requested endpoint windows, and the regular rational row records the corresponding Cauchy readback.

Finally H, C, P, N transport, replay, source, and name the same route. Because the carrier contains no coordinate for a host intersection object, selected point, quotient real, countable choice, trichotomy, or ambient completeness, the displayed handoff is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NestedIntervalTheorem}^{\uparrow}$ packet is generated from a NestedInterval row, a DyadicRatCore endpoint and tolerance ledger, StreamName finite windows, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 2308.1, Definition 2308.2, and Definition 2308.3 seed the finite constructive nested closed interval route from displayed nested windows to RegSeqRat readback and Real seal handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{NestedIntervalTheorem}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.NestedIntervalTheoremUp`

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient reals, selected host intersections, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window exposure, dyadic endpoint and tolerance exactness, StreamName window exposure, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalTheoremUp.

The routed HellyTheorem$^{\uparrow}$ chapter consumes chapter 208 and exports its finite subfamily ledger to chapter 2347 before the child files spell out the dependency route.

Chapter 2309

A Concrete Naming Certificate for $\mathsf{HellyTheorem}^{\uparrow}$

Following Helly and the finite-dimensional convexity tradition, the $\mathsf{HellyTheorem}^{\uparrow}$ packet records finite convex-intersection data as a BEDC naming surface. It consumes the convex-family rows of chapter 208, the finite witness rows of chapter 209, the ambient affine and vector rows of chapter 206 and chapter 47, and the compact finite-read discipline of chapter 129. The chapter names only finite subfamily ledgers, displayed intersection witnesses, transport, continuation replay, provenance, and local naming; it does not assert a host separation theorem, choice-selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

The $\mathsf{HellyTheorem}^{\uparrow}$ packet also consumes $\mathsf{ConvexSet}^{\uparrow}$ carrier rows and the finite-subfamily rows of chapter 1792 before any intersection conclusion is exposed.

The finite convex-geometry route is visible before the Helly certificate is exported: $\mathsf{HellyTheorem}^{\uparrow}$ reads each family member through the $\mathsf{ConvexSet}^{\uparrow}$ carrier of chapter 208, then uses the Caratheodory-style finite convex-hull rows of chapter 2347 as the support ledger that keeps the intersection test finite.

The finite subfamily ledger exported by $\mathsf{HellyTheorem}^{\uparrow}$ is consumed by chapter 2347 as the boundary that keeps convex-hull support rows finite before Caratheodory support reduction is read. That ledger is not available from the ambient affine row alone: it is read only after the $\mathsf{ConvexSet}^{\uparrow}$ membership rows of chapter 208 and the $\mathsf{Polytope}^{\uparrow}$ finite witness rows of chapter 209 expose the subfamily boundary that the Caratheodory packet consumes.

The same finite subfamily ledger is a reusable dependency for convex-hull and Caratheodory consumers: chapter 2347 may read the $\mathsf{HellyTheorem}^{\uparrow}$ row only after the $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{CompactMetric}^{\uparrow}$ surfaces named above have fixed the displayed finite family boundary.

The finite closed-family indexing boundary is supplied by $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ through chapter 1792; $\mathsf{HellyTheorem}^{\uparrow}$ then specializes that boundary by adding affine ambient rows, vector displacement rows, convex-family membership, and polytope witnesses for the Helly subfamily ledger.

The concrete dependency route is horizontal rather than ambient: each finite family member is first read as a $\mathsf{ConvexSet}^{\uparrow}$ row through chapter 208, then the finite subfamily ledger S is checked against the finite-intersection carrier rows of chapter 1792, and only then may the full-family row I be exposed to a Helly consumer.

2309.1 Finite Convex Intersection Carrier

Definition 2309.1 (HellyTheorem carrier)**.** *A* $\mathsf{HellyTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, V, F, W, S, I, T, C, P, N)$$

with the following displayed rows:

1. *A is the finite-dimensional* $\mathsf{AffineSpace}^{\uparrow}$ *ambient row.*
2. *V is the associated* $\mathsf{VecSpace}^{\uparrow}$ *displacement row used by the affine coordinates.*
3. *F is a finite family of* $\mathsf{ConvexSet}^{\uparrow}$ *rows in the ambient affine space.*
4. *W is the* $\mathsf{Polytope}^{\uparrow}$ *finite witness ledger for displayed simplex, face, or hull witnesses used by the subfamily checks.*
5. *S is the finite subfamily intersection ledger, recording each inspected subfamily and its displayed nonempty intersection witness.*
6. *I is the resulting full-family intersection row exposed by the packet after the finite ledger has been replayed.*
7. *T records componentwise* hsame *transport for $A, V, F, W, S, I, C, P, N$.*
8. *C records displayed* Cont *replay from ambient affine data through convex-family rows, finite polytope witnesses, subfamily intersection checks, the full-family intersection row, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{HellyTheorem}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, V, F, W, S, I, T, C, P, N$. It has no coordinate for host convex equality, ambient standard-model convexity, choice-selected points, separation or compactness theorems not displayed by the finite rows, or an infinite family of constraints.

Lean4 ✓ `BEDC.Derived.HellyTheoremUp.HellyTheoremTasteGate single carrier alignment`

2309.2 Helly NameCert Surface

Theorem 2309.2 (HellyTheorem NameCert obligations)**.** *For every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *carrier*

$$H = (A, V, F, W, S, I, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the ambient affine row, vector displacement row, finite convex-family row, finite polytope witness ledger, subfamily intersection ledger, full-family intersection row, transport, replay, provenance, and local naming;*
2. *ambient compatibility, requiring the* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *rows to be the only coordinates through which convex sets and polytope witnesses are read;*
3. *convex-family exposure, requiring each displayed family member to pass through a* $\mathsf{ConvexSet}^{\uparrow}$ *row before it enters any subfamily intersection check;*

4. *finite witness exactness, requiring every inspected subfamily intersection and every exported full-family intersection row to be backed by the displayed* $\mathsf{Polytope}^{\uparrow}$ *witness ledger;*
5. *subfamily ledger stability, requiring the finite intersection ledger* S *to be transported componentwise by* T *and replayed by* C *before* I *is exposed;*
6. *nonescape, excluding host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, infinite-family claims, Lean verification, and bridge checking.*

No HellyTheorem obligation is stored outside these rows.

Proof. Project the accepted carrier. The ambient coordinates are A and V, the convex input is the finite family row F, and the finite witness coordinates are W, S, I.

Ambient compatibility is forced because the carrier has no second affine or vector coordinate. Every convex-family read therefore passes through F over the same A, V boundary.

The witness ledger W and the subfamily ledger S are the only rows that record inspected finite intersections. Transport T and replay C preserve those rows componentwise before the exported row I is read.

The remaining coordinates are provenance and local naming. Since no coordinate stores host convex equality, a choice-selected point, a separation theorem, an unlisted compactness principle, an infinite-family assertion, Lean verification, or bridge data, those objects cannot become local NameCert obligations. □

Theorem 2309.3 (Helly finite-intersection route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet routes a finite convex-intersection request through*

$$\mathsf{AffineSpace}^{\uparrow}, \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{Polytope}^{\uparrow} \longmapsto S \longmapsto I.$$

The route exposes ambient affine data, vector displacement data, convex-family membership, finite polytope witnesses, and the finite subfamily ledger before the full-family intersection row is available. A request bypassing one of these rows is outside the accepted $\mathsf{HellyTheorem}^{\uparrow}$ *carrier.*

Proof. Read the carrier of Definition 2309.1. The first two coordinates bind the ambient affine and vector rows, and the convex-family coordinate F is interpreted only over that ambient boundary.

The polytope witness row W and the finite subfamily ledger S supply the displayed intersection checks. The replay row C presents them before the full-family row I is exposed.

Thus a consumer reaches I only along the stated route. If any one of the ambient, convex-family, polytope-witness, or subfamily-ledger rows is missing, the packet has no finite coordinate from which the requested intersection row can be read. □

Theorem 2309.4 (Helly finite-intersection forward link)**.** *Every finite convex-intersection request accepted by a* $\mathsf{HellyTheorem}^{\uparrow}$ *packet consumes the finite closed-family indexing discipline of chapter 1792 before the full-family intersection row* I *is exposed. The* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$ *sibling supplies the finite-family ledger boundary, while* $\mathsf{HellyTheorem}^{\uparrow}$ *supplies the convex and affine specialization over* $A, V, F, W, S, I, T, C, P, N$.

Proof. Project the carrier of Definition 2309.1. The ambient and specialization rows are A, V, F, W, the finite subfamily ledger is S, the full-family intersection row is I, and T, C, P, N transport, replay, source, and name the same packet.

Apply Definition 2309.3. It requires the request to pass through the finite subfamily ledger S before I is available, so a consumer cannot read the full-family row directly from the ambient affine or convex-family rows.

The finite intersection property sibling provides the closed finite-family indexing boundary named in chapter 1792. Helly contributes the convex, affine, vector, and polytope-witness specialization attached to that boundary, rather than replacing it by a second indexing principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 2309.1, Definition 2309.2, and Definition 2309.3 seed the finite convex-intersection packet over affine ambient rows, vector displacement rows, convex-family rows, polytope witness ledgers, subfamily intersection ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, infinite-family Helly claims, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ambient compatibility, convex-family exposure, finite witness exactness, subfamily ledger stability, and finite-intersection routing for BEDC.Derived.HellyTheoremUp.

Constructive story: A $\mathsf{HellyTheorem}^\uparrow$ packet is generated from affine ambient data, vector displacement data, finite convex-family rows, finite polytope witness rows, finite subfamily intersection ledgers, a full-family intersection row, componentwise transport, continuation replay, provenance, and local NameCert rows.

The finite subfamily ledger exported by $\mathsf{HellyTheorem}^\uparrow$ is consumed by chapter 2347 through a concrete dependency route: first the convex-family membership rows are read from chapter 208, then the finite witness ledger is read from chapter 209, then the Helly full-family intersection row is exposed, and only after those rows are fixed may the Caratheodory support-reduction packet read a bounded support surface. This route records a downstream use of $\mathsf{HellyTheorem}^\uparrow$ without adding a theorem environment to the hub.

Theorem 2309.5 (Helly Caratheodory forward dependency)**.** *Every* $\mathsf{HellyTheorem}^\uparrow$ *finite-subfamily read that is exported to a convex-hull consumer must first consume the* $\mathsf{CaratheodoryConvexHull}^\uparrow$ *support-reduction route of chapter 2347 and the convex-hull carrier rows already displayed in Definition 2309.12. Only after those rows are visible may the packet export the Helly finite-intersection route of Definition 2309.18. The dependency is a finite forward read through the displayed convex-family, polytope, Caratheodory, and local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$ *rows, not a host separation theorem, selected intersection point, or ambient convexity principle.*

Proof. Project the existing Helly dependency route. The convex-family rows and finite witness rows are displayed before the Helly subfamily ledger is inspected.

Next read the Caratheodory convex-hull surface. The support-reduction packet consumes the displayed subfamily ledger only as finite bounded-support data, so the convex-hull carrier remains upstream of the Helly finite-intersection export.

Finally repack the rows into the local Helly NameCert route. Since no row names a host separation theorem, selected intersection point, or ambient convexity principle, the exported dependency is exactly the finite forward route named in the statement. □

Theorem 2309.6 (Helly Caratheodory sibling route)**.** *The* $\mathsf{HellyTheorem}^\uparrow$ *finite-subfamily ledger is available to the* $\mathsf{CaratheodoryConvexHull}^\uparrow$ *consumer of chapter 2347 only after the* $\mathsf{ConvexSet}^\uparrow$, $\mathsf{Polytope}^\uparrow$, $\mathsf{AffineSpace}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, *and* $\mathsf{CompactMetric}^\uparrow$ *rows expose the finite convex-family boundary. The consumed route is the displayed finite ledger through those rows and the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^\uparrow}$ *packet; it is not a detached convexity principle, a selected intersection point, or a host support-reduction theorem.*

Proof. Project the Helly ledger as a finite subfamily packet. Its convex-family membership, polytope witness, affine and vector-space coordinates, and compactmetric boundary rows must be visible before a convex-hull consumer can read the finite-intersection surface.

Then pass the exposed ledger to the Caratheodory consumer. The consumer receives bounded-support data from the displayed packet, so the route remains tied to the same local NameCert rows.

No additional coordinate names host convexity, selected intersection points, or an external support theorem. The sibling route is therefore the finite dependency stated above. □

Theorem 2309.7 (Helly Caratheodory finite-dependency audit)**.** *Any* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily ledger exported to* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *must display* $\mathsf{ConvexSet}^{\uparrow}$ *membership,* $\mathsf{Polytope}^{\uparrow}$ *finite witnesses,* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *ambient rows, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read boundary, and the local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row before the Caratheodory bounded-support surface is consumed. The route exports no host separation theorem, selected intersection point, quotient convex-set equality, or ambient convexity principle.*

Proof. Start with the forward dependency theorem, Definition 2309.5. It places the convex-family and finite-witness rows upstream of any convex-hull consumer.

Apply the sibling route theorem, Definition 2309.6. It refines the dependency surface to $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and the local Helly naming row before the Caratheodory surface is read.

The carrier non-escape boundary in chapter 2309 excludes host convex-set equality, choice-selected intersection points, separation theorems, unlisted compactness principles, and infinite-family claims. Hence the Caratheodory handoff is exactly the finite dependency audit stated above. □

Lemma 2309.8 (Caratheodory–Helly finite dependency route)**.** *Every* $\mathsf{HellyTheorem}^{\uparrow}$ *consumer that uses Caratheodory-style convex support data must first expose the finite convex-family row of chapter 208, the finite witness and face rows of chapter 209, and the bounded-support handoff of chapter 2347. Only after those rows are visible may the Helly packet read its subfamily-intersection ledger and export the full-family intersection row through* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$.

Proof. Project the $\mathsf{HellyTheorem}^{\uparrow}$ carrier from Definition 2309.1. The convex-family coordinate is displayed before the subfamily ledger can be inspected, and the finite witness rows are carried separately from the ambient affine data.

The Caratheodory convex-hull packet of chapter 2347 then supplies only bounded support rows. It does not select an ambient intersection point or add a host separation theorem.

Thus the dependency lattice is finite: convex-family rows and polytope witnesses feed the Caratheodory support route, and that route feeds the Helly subfamily ledger before the full-family row is exported. □

Theorem 2309.9 (Helly Caratheodory convex-hull sibling route)**.** *Every finite-intersection consumer that reads an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet must read the convex-family row through the* $\mathsf{ConvexSet}^{\uparrow}$ *surface of chapter 208, the Caratheodory convex-hull route of chapter 2347, the finite closed-family boundary of chapter 1792, and the Helly carrier of chapter 2309 before the finite intersection witness is exported. The route is a dependency-lattice projection; it does not export an isolated convex-family oracle, ambient separation theorem, quotient convex-hull equality, or selected intersection point.*

Proof. Project the Helly carrier in chapter 2309. Its finite intersection witness is downstream of the convex-family row, finite closed-family indexing, affine or convex support rows, transport, replay, provenance, and local naming.

The convex-family row is first read through the $\mathsf{ConvexSet}^{\uparrow}$ sibling of chapter 208. The finite support and convex-hull read then passes through the Caratheodory route of chapter 2347, which records the displayed finite-combination dependency rather than an ambient hull operator.

Finally the finite-family ledger is read through chapter 1792. These three sibling routes are already dependency rows for the Helly packet, so composing them before the witness export is only a projection through the existing carrier and lattice.

Since no listed row carries an isolated convex-family oracle, separation theorem, quotient hull equality, or selected intersection point, the exported Helly witness remains the displayed finite-intersection route. □

Theorem 2309.10 (Helly finite-subfamily ledger route)**.** *Every* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily ledger consumed by* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *must read* $\mathsf{ConvexSet}^{\uparrow}$ *membership rows,* $\mathsf{Polytope}^{\uparrow}$ *finite witness rows,* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *ambient rows, and* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read boundaries before exporting the bounded support surface. The ledger route is therefore*

$$\mathsf{AffineSpace}^{\uparrow}, \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{ConvexSet}^{\uparrow} \longmapsto \mathsf{Polytope}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CaratheodoryConvexHull}^{\uparrow}.$$

It exposes only the named finite subfamily data and does not introduce a host separation theorem, a selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

Proof. Project the existing $\mathsf{HellyTheorem}^{\uparrow}$ carrier of Definition 2309.1. Its ambient rows are A, V, its convex-family row is F, and its finite witness and subfamily rows are W, S, I.

Follow the sibling route already displayed by the chapter: $\mathsf{ConvexSet}^{\uparrow}$ supplies membership rows, $\mathsf{Polytope}^{\uparrow}$ supplies finite witnesses, $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ supply the ambient affine and displacement rows, and $\mathsf{CompactMetric}^{\uparrow}$ supplies the finite-read boundary. The $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ consumer can read a bounded support surface only after those rows have been displayed.

No carrier coordinate stores host separation, a selected intersection point, quotient convex-set equality, or ambient standard-model convexity. The exported ledger is therefore exactly the finite sibling route stated above. □

Proposition 2309.11 (Helly convex-hull sibling dependency)**.** *The* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection certificate supplies its convex-hull dependency only through the finite subfamily ledger consumed by chapter 2347. Thus any Caratheodory convex-hull readback using* $\mathsf{HellyTheorem}^{\uparrow}$ *must display the convex-family classifier, finite witness ledger, full-family intersection row, and local NameCert provenance before the Caratheodory finite-generation route may expose bounded support data.*

Proof. Use Definition 2309.10. It already forces the Helly ledger through the convex-family, polytope, affine, vector, and compact finite-read rows before the Caratheodory convex-hull surface is available.

The sibling chapter chapter 2347 then consumes that ledger as its Helly handoff row. Since neither the Helly carrier nor the Caratheodory convex-hull carrier stores a host separation theorem, selected intersection point, or quotient convex-hull equality, the dependency remains the displayed finite-generation route and nothing larger. □

Consequently every $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ or finite convex-family consumer may read $\mathsf{HellyTheorem}^{\uparrow}$ only after the $\mathsf{ConvexSet}^{\uparrow}$ membership rows of chapter 208, the $\mathsf{Polytope}^{\uparrow}$ finite witness rows, the $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ ambient rows, and the $\mathsf{CompactMetric}^{\uparrow}$ finite-read boundary have been

displayed. The consumption route then reaches chapter 2347 as a finite subfamily ledger, not as a host convexity theorem or an uncarried intersection selector. The finite subfamily ledger of $\mathsf{HellyTheorem}^{\uparrow}$ is read through the horizontal convex lattice before any convex-hull consumer may use it. The route first fixes the convex-family rows of chapter 208, the finite witness rows of chapter 209, the affine and vector ambient rows of chapter 206 and chapter 47, and the support-reduction boundary of chapter 2347; only after those rows are displayed may the Helly finite subfamily ledger be consumed as convex-hull input.

Convex-hull and Caratheodory dependency. The $\mathsf{HellyTheorem}^{\uparrow}$ packet consumes its finite convex-family carrier through the convex-set surface of chapter 208 and its affine-combination reduction through the Caratheodory convex-hull surface of chapter 2347 before it exports an intersection certificate. Thus the Helly read is not an isolated incidence claim: it first exposes the convex-family rows, then the Caratheodory finite-combination rows, and only then packages the common-intersection row through its local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ surface.

Theorem 2309.12 (Helly theorem convex-hull forward link)**.** *Every convex-hull consumer of an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet must read the displayed finite family row, full-family intersection witness, Caratheodory-style finite subfamily row,* $\mathsf{ConvexSet}^{\uparrow}$ *classifier rows, and local* $\mathsf{NameCert}$ *provenance before it uses the Helly packet. The forward link factors through the Helly construction of chapter 2309, the Caratheodory dependency route in Definition 2309.10, and the convex-hull surface of chapter 2347. It supplies no full compact convexity theorem and no host separation theorem.*

Proof. Project the Helly carrier. Its finite family row is F, its full-family intersection witness is I, and its local naming and provenance are carried by P, N.

Use the existing Caratheodory forward dependency route. That route reads $\mathsf{ConvexSet}^{\uparrow}$ membership rows, $\mathsf{Polytope}^{\uparrow}$ finite witnesses, affine and vector ambient rows, and compact finite-read boundaries before a Caratheodory convex-hull consumer can read a bounded support surface.

The present forward link names the same boundary for convex-hull consumers. The consumer may use the Helly packet only after the finite family, intersection witness, finite subfamily ledger, convex-set classifier rows, and local NameCert provenance have all been displayed.

Since the Helly carrier has no coordinate for compact convexity as a host theorem and no coordinate for a separation theorem, neither principle is exported by this link. □

Theorem 2309.13 (Helly convex-hull forward scope)**.** *The finite-intersection row of an accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet is a forward scope for the* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *consumer of chapter 2347. A convex-hull membership request may consume the Helly row only after the convex-set and polytope sibling packets have displayed their finite family, face-boundary, support, and coefficient ledgers. The handoff exports a finite subfamily certificate for convex-hull reasoning; it does not export ambient separation, selected minimal supports, quotient hull equality, host convexity, or Real completeness.*

Theorem 2309.14 (Helly convex-family dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily read must display the* $\mathsf{ConvexSet}^{\uparrow}$ *family rows and the* $\mathsf{Polytope}^{\uparrow}$ *finite witness ledger before the subfamily ledger S of Definition 2309.1 is inspected. The family row consumes the convex carrier of chapter 208 before the finite-intersection ledger is read. The ambient* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *rows fix the affine boundary for those convex sets, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-read discipline bounds the inspected family window, and any* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *consumer receives only the displayed subfamily ledger after those rows are visible. The route exports no host separation*

theorem, choice-selected intersection point, quotient convex-set equality, or ambient standard-model convexity.

Proof. Project the carrier of Definition 2309.1. The convex-family coordinate is F, the finite witness coordinate is W, the ambient coordinates are A, V, and the inspected subfamily coordinate is S.

The row F is read through $\mathsf{ConvexSet}^{\uparrow}$, while W is read through $\mathsf{Polytope}^{\uparrow}$. The rows A, V keep those reads in the displayed affine and vector ambient surface, and the compact finite-read boundary fixes the inspected finite window before S is replayed.

A Caratheodory convex-hull consumer may therefore read the Helly handoff only after the convex-family, polytope, affine, vector, and compact rows have been displayed. Since the structural rows only transport, replay, source, and name this packet, no host separation theorem, selected point, quotient convex-set equality, or ambient standard model is exported. □

Theorem 2309.15 (Helly convex-geometry dependency lattice)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection read consumes the convex-geometry lattice*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{Polytope}^{\uparrow},\ \mathsf{CaratheodoryConvexHull}^{\uparrow}$$

before it exposes its local intersection row. The $\mathsf{ConvexSet}^{\uparrow}$ *surface of chapter 208 supplies the membership and convex-family rows, the* $\mathsf{Polytope}^{\uparrow}$ *surface of chapter 209 supplies the finite witness rows, and the* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *surface of chapter 2347 supplies the bounded-support handoff consumed by convex-hull readers. A Helly packet that omits one of these rows is outside the accepted sibling route.*

Proof. Project the carrier in Definition 2309.1. Its convex-family coordinate is F, its finite witness coordinate is W, and its subfamily and full-family intersection coordinates are S, I.

The row F is read through the $\mathsf{ConvexSet}^{\uparrow}$ classifier, so no finite subfamily can be inspected before the convex-family membership rows have been displayed. The row W is the finite $\mathsf{Polytope}^{\uparrow}$ witness ledger, so every inspected subfamily intersection is backed by the polytope witness surface before S is replayed.

The convex-hull consumer then reads the same finite subfamily ledger through $\mathsf{CaratheodoryConvexHull}^{\uparrow}$. Since the carrier has no alternate row for a host separation theorem, selected intersection point, quotient convex-set equality, or ambient standard-model convexity, the three named sibling surfaces form the whole dependency lattice for the Helly read. □

Consequently $\mathsf{HellyTheorem}^{\uparrow}$ is not an isolated hub: its public convex-geometry use is routed through $\mathsf{ConvexSet}^{\uparrow}$ membership rows, $\mathsf{Polytope}^{\uparrow}$ finite witness rows, and $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ bounded-support rows before any local $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ intersection certificate is consumed.

Proposition 2309.16 (Helly convex-geometry sibling lattice route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection packet consumes the convex-set membership rows of chapter 208, the polytope finite witness rows of chapter 209, and the convex-hull support surface of chapter 2347 through the same finite intersection data before any local* $\mathsf{NameCert}_{\mathsf{HellyTheorem}^{\uparrow}}$ *row is read. The route is a sibling dependency lattice over displayed finite subfamilies, not a host convexity theorem or a choice-selected intersection point.*

Proof. Project the carrier of Definition 2309.1. The convex-family row F supplies only $\mathsf{ConvexSet}^{\uparrow}$ membership data, while the witness row W supplies the $\mathsf{Polytope}^{\uparrow}$ finite face, simplex, or hull witnesses used by inspected subfamilies.

The finite subfamily ledger S is then the shared handoff. A convex-hull consumer reads it through $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ as bounded support data, and the Helly packet reads the same ledger before exposing the full-family row I.

Since T, C, P, N only transport, replay, source, and name the displayed rows, they cannot introduce a second convex-hull carrier, an ambient separation argument, or a selected intersection point. Thus the sibling lattice is exactly the finite route through $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, the Caratheodory convex-hull surface, and the local Helly ledger. □

Consequently the $\mathsf{HellyTheorem}^{\uparrow}$ hub records a concrete convex-geometry sibling dependency: finite intersection data is read only after the membership, finite witness, and convex-hull support rows have been displayed inside the same BHist, hsame, Cont, Pkg, and NameCert surface. The route is the same finite sibling lattice, not a host convexity theorem.

Theorem 2309.17 (Helly convex polytope lattice route)**.** *Any accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-intersection packet reads its convex family through* $\mathsf{ConvexSet}^{\uparrow}$*, its finite cell witnesses through* $\mathsf{Polytope}^{\uparrow}$*, and its ambient coordinates through* $\mathsf{AffineSpace}^{\uparrow}$ *before it exposes the local intersection ledger. In carrier notation, the public route first displays the convex-family row F, the polytope witness row W, and the affine ambient row A of Definition 2309.1; only then may the subfamily ledger S and full-family row I be consumed. The route exports no host convex-set equality, selected intersection point, separation theorem, quotient affine coordinate, or ambient standard-model convexity.*

Proof. Project the Helly carrier of Definition 2309.1. Its ambient row is A, its convex-family row is F, and its finite witness row is W.

The convex-family row is read through chapter 208. Thus an accepted packet displays the $\mathsf{ConvexSet}^{\uparrow}$ membership and convexity rows before a subfamily can be inspected.

The finite cell witnesses are read through chapter 209, while the ambient coordinates are read through chapter 206. These rows bind the finite cells to the same affine boundary before the ledger rows S and I are exposed.

Finally inspect the remaining coordinates T, C, P, N. They transport, replay, source, and name the same finite packet, and they add no host convex equality, selected intersection point, separation theorem, quotient affine coordinate, or ambient standard-model convexity. The stated convex-polytope-affine route therefore exhausts the public Helly read. □

Theorem 2309.18 (Helly theorem convex-polytope dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-subfamily read displays the* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{AffineSpace}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and compact finite-read rows before the finite-intersection ledger is consumed. The route may inspect the Helly family row, finite subfamily row, and intersection ledger only after those sibling rows are present; it exports no selected intersection point, ambient separation theorem, quotient affine coordinate, or host compactness oracle.*

Proof. Project the Helly carrier of Definition 2309.1. The convex-family row, finite witness row, ambient row, and finite-intersection ledger are explicit packet coordinates, so a consumer read starts from displayed rows rather than from a host theorem about arbitrary convex families.

Use the sibling dependency files already attached to this chapter. The convex set dependency route supplies the $\mathsf{ConvexSet}^{\uparrow}$ membership and convexity rows, while the convex-polytope lattice route supplies the $\mathsf{Polytope}^{\uparrow}$ finite-cell witnesses and the $\mathsf{AffineSpace}^{\uparrow}$ ambient boundary.

The finite convex lattice route adds the $\mathsf{VecSpace}^{\uparrow}$ scalar row and the compact finite-read rows needed before a finite subfamily can be consumed. With those rows displayed, the Helly ledger reads only the finite-intersection surface carried by the packet.

Finally replay the local transport, provenance, and naming coordinates. They preserve the same finite route and do not add a selected point, separation theorem, quotient affine coordinate, or host compactness oracle. Hence the convex-polytope dependency route is exhausted before the finite-intersection ledger is read. □

Theorem 2309.19 (Helly convex-set dependency route)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite convex-intersection read first consumes the convex carrier of chapter 208 and the finite polytope witness surface of chapter 209. Only after those two sibling rows are displayed may the Helly intersection row expose its finite subfamily ledger, common-intersection row, transport, replay, provenance, and local naming. The route exports no host separation theorem, selected intersection point, quotient convex-set equality, unrestricted finite-cover oracle, or ambient standard-model convexity.*

Proof. Project the Helly carrier of Definition 2309.1. The convex-family row is consumed through the convex-set certificate, and the finite witness row is consumed through the polytope certificate.

After those sibling rows are visible, the Helly subfamily and intersection rows can be read as finite BHist data. The structural rows transport, replay, source, and name the same route, so the carrier has no coordinate for a host separation theorem, selected point, quotient convex-set equality, or unrestricted cover oracle. □

The finite-intersection certificates used by $\mathsf{HellyTheorem}^{\uparrow}$ enter through the same convex-row surface as chapter 208 and the finite hull route of chapter 2347. A Helly carrier can therefore read its displayed family of convex sets, witness subfamilies, and intersection rows as finite BHist data before it exports the Helly conclusion, rather than importing an ambient convexity theorem or an unrestricted finite-cover oracle.

Theorem 2309.20 (Helly convex dependency anchor)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *packet consumes finite convex-family rows and finite-intersection rows before it exports the Helly* $\mathsf{NameCert}$ *surface. The convex-family rows are read through the* $\mathsf{ConvexSet}^{\uparrow}$ *membership surface of chapter 208; the finite-intersection rows are read through the subfamily ledger of chapter 1792. Only after both rows are displayed may the packet expose its full-family intersection witness, transport rows, replay rows, provenance, and local naming.*

Proof. Project the carrier of Definition 2309.1. Its family coordinate cannot be read as an ambient convexity assertion; each member is inspected through the displayed convex-set rows.

Use the finite-intersection dependency route already recorded for the Helly packet. That route keeps the subfamily ledger visible before the full-family intersection row is exported.

The remaining rows only transport, replay, source, and name the same finite surface. Hence the Helly export is anchored in the convex-family and finite-intersection siblings rather than in a detached kernel-level theorem. □

Theorem 2309.21 (Helly finite convex-family budget)**.** *Every accepted* $\mathsf{HellyTheorem}^{\uparrow}$ *finite-family read must expose* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{AffineSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows before it exports a finite intersection witness. In particular, the convex-set membership rows of chapter 208, the finite witness rows of chapter 209, the compact finite-read rows of chapter 60, and the ambient affine and vector rows of chapter 206 and chapter 47 form the displayed dependency route for the local intersection ledger.*

Proof. Project the $\mathsf{HellyTheorem}^{\uparrow}$ carrier from Definition 2309.1. The convex-family coordinate F is read through the $\mathsf{ConvexSet}^{\uparrow}$ surface, the witness coordinate W is read through $\mathsf{Polytope}^{\uparrow}$, and

the ambient rows A, V are read through $\mathsf{AffineSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ before the finite subfamily ledger S can be inspected.

The compact finite-read discipline supplies the bounded inspection surface for those finite subfamilies. Transport, continuation replay, provenance, and local naming are then carried by the displayed BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows. Therefore the exported finite intersection witness is downstream of the sibling convex, polytope, compact, vector, and affine chapters rather than a detached host convexity fact. □

The $\mathsf{HellyTheorem}^{\uparrow}$ packet consumes the finite-intersection lattice of chapter 1792 before any Helly carrier can export an intersection witness. The route also reads the convex carrier rows of chapter 208 and the compact finite-cover boundary of chapter 129; together with the local sibling route of Definition 2309.10, these rows force the witness to be a finite NameCert consequence rather than a kernel-only intersection assertion. The finite-family carrier for $\mathsf{HellyTheorem}^{\uparrow}$ reads the finite-intersection route of chapter 1792 together with the $\mathsf{ConvexSet}^{\uparrow}$ surface and the convex-family coordinate F before it can export a Helly subfamily witness. This bridge keeps the witness surface dependent on displayed finite-intersection and convex-family data rather than on an isolated kernel assertion.

Chapter 2310

A Concrete Naming Certificate for $\mathsf{TemperedDistribution}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{TemperedDistribution}^{\uparrow}$ packet records the finite observable face of a tempered distribution as a continuous linear action on Schwartz-test windows. The chapter connects the distribution row of chapter 186, the finite Fourier observation surface of chapter 124, the weak-derivative route of chapter 2210, the Sobolev comparison surface of chapter 440, and the terminal real readback of chapter 38. It names only the BEDC certificate surface for these finite rows; it does not import a completed Schwartz space, a host distribution equality relation, quotient dual spaces, or a global topological-vector-space completeness theorem.

2310.1 Carrier definition

Definition 2310.1 (Tempered distribution carrier)**.** *A* $\mathsf{TemperedDistribution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, G, A, F, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite Schwartz-test window row. It records the named test functions visible to the packet, their finite derivative orders, polynomial weights, support windows when present, and the requested observation indices.*
2. *G is the polynomial-growth ledger. It records the displayed seminorm bounds used to admit each test row and each derivative row.*
3. *A is the* $\mathsf{Distribution}^{\uparrow}$ *action row. It records the finite list of test-window evaluations and their* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoints.*
4. *F is the* $\mathsf{Fourier}^{\uparrow}$ *coefficient and test-transform row. It records finite transform observations for the same test windows and the coefficient windows consumed by Fourier-facing reads.*
5. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *and* $\mathsf{Sobolev}^{\uparrow}$ *compatibility row. It records integration-by-parts signs, derivative-order shifts, and Sobolev-window comparisons that are explicitly displayed in the packet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational approximation windows of the action and transform endpoints.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the displayed readback route.*
8. *L is the local classifier row. It admits two packets together exactly when their Schwartz-test windows, polynomial-growth ledgers, distribution-action rows, Fourier rows, weak-derivative and Sobolev rows, rational readbacks, and real seals match componentwise.*
9. *H records componentwise* hsame *transport for all visible rows.*
10. *C records displayed* Cont *replay from a tempered-distribution request through $S, G, A, F, W, Q, E, L, H$ in that order.*
11. *P is the* Pkg *provenance row over* $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores a completed Schwartz space, an unrestricted continuous dual, host equality of distributions, quotient test-function equality, selected representative, countable choice schedule, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.TemperedDistributionUp.TemperedDistributionTasteGate single carrier alignment`

2310.2 The certificate

Theorem 2310.2 (Tempered distribution NameCert obligations)**.** *For every accepted* $\mathsf{TemperedDistribution}^{\uparrow}$ *carrier $T = (S, G, A, F, W, Q, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Schwartz-test window, polynomial-growth ledger, distribution-action, Fourier, weak-derivative and Sobolev compatibility, rational readback, real seal, classifier, transport, replay, provenance, and local naming rows;*
2. *action exactness, requiring every displayed evaluation of the distribution against a test window to pass through A after the corresponding test row in S and growth ledger in G have been opened;*
3. *polynomial-growth stability, requiring tightened derivative or weight requests to replay the same ledger G before any action or transform endpoint is read;*
4. *Fourier compatibility, requiring every finite transform or coefficient read to pass through F and then through the same Q, E readback route as the action row;*
5. *weak-derivative and Sobolev compatibility, requiring every derivative-order shift, sign row, or Sobolev-window comparison to pass through W before the packet is exported;*
6. *componentwise* hsame, *displayed* Cont *replay,* Pkg *provenance, and local* NameCert *closure through L, H, C, P, N.*

No obligation outside these rows is part of the certificate.

Proof. Project the carrier of Definition 2310.1. The row inventory gives the whole finite packet, so carrier admission is exhausted by $S, G, A, F, W, Q, E, L, H, C, P, N$.

The Schwartz-test row S fixes which tests and derivative orders are visible. The growth ledger G must be read before any action endpoint, because it is the row that admits the displayed seminorm and polynomial-weight data.

The action row A then supplies only finite evaluations against those admitted tests. The rational and real endpoints of those evaluations pass through Q and E, so there is no detached real seal or host distribution value.

The Fourier and derivative routes are likewise internal rows. The Fourier row F records finite coefficient and test-transform observations, while W records the weak-derivative signs and Sobolev-window comparisons. Transport, replay, provenance, and naming rows compare and repack the same data, and none of them adds a completed test-function space, unrestricted dual, quotient equality, selected representative, choice schedule, or ambient completeness theorem. □

Theorem 2310.3 (Tempered distribution Fourier handoff)**.** *Every accepted* $\mathsf{TemperedDistribution}^{\uparrow}$ *read exported to a Fourier-facing consumer factors through*

$$S, G, A \longmapsto F \longmapsto W, Q, E \longmapsto L, H, C, P, N.$$

Thus the consumer receives the displayed Schwartz-test window, polynomial-growth ledger, distribution-action row, finite Fourier coefficient and test-transform row, weak-derivative and Sobolev compatibility row, rational readback, real seal, classifier, componentwise transport, replay, provenance, and local naming. It receives no completed Schwartz topology, host distribution equality, quotient dual object, selected representative, countable choice schedule, Plancherel theorem, inversion theorem, or ambient completeness principle.

Proof. Start with a Fourier-facing read. The test row S and growth ledger G must be visible before the packet can expose a distribution action, because the finite transform request is attached to the same admitted test windows.

Read A next. It supplies the finite action endpoints against the admitted tests and records the source $\mathsf{Distribution}^{\uparrow}$ route for those endpoints.

The handoff to F is then a finite-observation handoff: coefficient windows and test-transform rows are exposed as $\mathsf{Fourier}^{\uparrow}$ packages, not as host transforms of a completed function space. The compatibility row W records exactly the weak-derivative signs and Sobolev comparisons that are part of the same request.

Finally Q, E, L, H, C, P, N read back, compare, replay, source, and name the same route. Since none of these rows contains a completed Schwartz topology, host equality, quotient dual, selected representative, choice schedule, Plancherel theorem, inversion theorem, or ambient completeness witness, the Fourier consumer receives only the displayed finite certificate. □

Falsifiable prediction: Every accepted $\mathsf{TemperedDistribution}^{\uparrow}$ consumer that reaches a Fourier coefficient, weak-derivative sign, Sobolev comparison, or $\mathsf{Real}^{\uparrow}$ seal without first reading the finite Schwartz-test window and polynomial-growth ledger refutes the carrier surface.

Independence witness: distribution, fourier, weakderivative, sobolev, regseqrat, real: none is bijective to $\mathsf{TemperedDistribution}^{\uparrow}$ because this packet jointly carries finite Schwartz-test windows, polynomial-growth ledgers, distribution actions, Fourier coefficient and transform rows, weak-derivative and Sobolev compatibility rows, rational readbacks, real seals, classifier, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}$.

Constructive story: A $\mathsf{TemperedDistribution}^{\uparrow}$ packet is generated from finite Schwartz-test windows, polynomial-growth ledger rows, displayed $\mathsf{Distribution}^{\uparrow}$ actions, finite $\mathsf{Fourier}^{\uparrow}$ coefficient and test-transform observations, $\mathsf{WeakDerivative}^{\uparrow}$ and $\mathsf{Sobolev}^{\uparrow}$ compatibility rows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TemperedDistribution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2310.1, Definition 2310.2, and Definition 2310.3 seed the finite tempered-distribution interface over Schwartz-test windows, polynomial-growth ledgers, distribution actions, Fourier observations, weak-derivative and Sobolev compatibility, rational readbacks, and real seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{TemperedDistribution}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed Schwartz spaces, unrestricted continuous duals, quotient test-function equality, host distribution equality, selected representatives, countable choice, Plancherel, inversion, or ambient completeness.

Upgrade path: A formal route can encode the finite carrier and prove the NameCert obligation surface and Fourier handoff theorem.

Chapter 2311

A Concrete Naming Certificate for $\mathsf{DirichletKernel}^{\uparrow}$

Following Zygmund and Katznelson on trigonometric series, the $\mathsf{DirichletKernel}^{\uparrow}$ packet records the finite symmetric Fourier kernel that evaluates one displayed partial-sum window before any convergence-facing consumer may read a series or limit handoff. It sits over the Fourier coefficient interface of chapter 124, the positive averaging sibling $\mathsf{FejerKernel}^{\uparrow}$ of chapter 1939, the complex arithmetic carrier of chapter 39, the circle-source samples of chapter 58, and the real-series partial-sum corridor of chapter 1581. The chapter names a finite trigonometric partial-sum kernel; it does not import host pointwise limits, ambient Fourier convergence, quotient function spaces, countable choice, or a selected infinite sum.

Definition 2311.1 (DirichletKernel carrier)**.** *A* $\mathsf{DirichletKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, A, S, Z, F, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the frequency-window row, recording the symmetric finite interval of integer frequencies and the endpoint truncation bound read by the partial-sum request.*
2. *A is the trigonometric coefficient ledger, recording the displayed coefficient attached to each frequency in W together with its opposite-frequency mate.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *evaluation route, exposing only the finite circle samples and phase rows at which the kernel is inspected.*
4. *Z is the* $\mathsf{Complex}^{\uparrow}$ *summation row, recording the displayed complex additions and multiplications used to accumulate the finite trigonometric polynomial.*
5. *F is the* $\mathsf{Fourier}^{\uparrow}$ *partial-sum handoff row, presenting the same frequency window and coefficient ledger to a Fourier partial-sum consumer.*
6. *R is the* $\mathsf{RealSeries}^{\uparrow}$ *and* $\mathsf{CplxSer}^{\uparrow}$ *provenance row, recording which real or complex partial-sum corridor may read the displayed finite sum.*
7. *H records componentwise* hsame *transport for frequency windows, coefficient ledgers, circle evaluations, complex summation, Fourier handoff, series provenance, replay, package provenance, and local naming.*

8. *C records displayed* Cont *replay from a phase and coefficient request through W, A, S, Z, F, R in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{DirichletKernel}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CplxSer}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, A, S, Z, F, R, H, C, P, N$. It has no coordinate for a host pointwise limit, an external convergence theorem, an ambient quotient of functions, countable choice, or a selected infinite-series value.

Lean4 ✓ `BEDC.Derived.DirichletKernelUp.TasteGate.DirichletKernelTasteGate single carrier alignment`

Theorem 2311.2 (DirichletKernel NameCert obligations)**.** *For every accepted* $\mathsf{DirichletKernel}^{\uparrow}$ *carrier $K = (W, A, S, Z, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the displayed frequency window, trigonometric coefficient ledger, circle-evaluation route, complex summation row, Fourier partial-sum handoff, real and complex series provenance, transport, replay, package provenance, and local naming;*
2. *finite-window stability, requiring every transported or replayed kernel read to preserve the same symmetric frequency interval and endpoint truncation bound in W;*
3. *coefficient symmetry, requiring the ledger A to pair each displayed frequency with its opposite-frequency mate before the summation row is consumed;*
4. *convolution and summation handoff, requiring the* $\mathsf{Complex}^{\uparrow}$ *row Z and the* $\mathsf{Fourier}^{\uparrow}$ *row F to expose only the finite trigonometric polynomial determined by W, A, S;*
5. $\mathsf{S}^{1\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, *and* $\mathsf{CplxSer}^{\uparrow}$ *routing, requiring every phase, arithmetic, and partial-sum consumer to factor through S, Z, F, R;*
6. *transport and replay stability, requiring H and C to preserve the ordered route through the same carrier coordinates;*
7. *ledger refusal, requiring the packet to export no host pointwise limit, external Fourier convergence principle, quotient function equality, countable-choice selector, or selected infinite-series value.*

No Dirichlet-kernel obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.DirichletKernelUp.DirichletKernelCarrier namecert obligations`
↪ variant `BEDC.Derived.DirichletKernelUp.DirichletKernelNameCertObligations`

Proof. Carrier admission is projection from Definition 2311.1. The displayed tuple exposes exactly the frequency window, coefficient ledger, circle evaluation, complex summation, Fourier handoff, series provenance, transport, replay, package provenance, and local naming rows.

Finite-window stability is read from W. Since the carrier stores one symmetric frequency interval and one endpoint truncation bound, every transported or replayed kernel read names the same finite window before any coefficient or phase row is inspected.

Coefficient symmetry is read from A. The ledger pairs displayed frequencies with their opposite-frequency mates, so the summation row may form the finite trigonometric polynomial without appealing to a host Fourier basis or an ambient function-space quotient.

The handoff rows are then forced through S, Z, F, R. Circle samples enter through $\mathsf{S}^{1\uparrow}$, arithmetic accumulation enters through $\mathsf{Complex}^{\uparrow}$, the partial-sum consumer enters through $\mathsf{Fourier}^{\uparrow}$, and series-facing provenance enters only through the named real and complex series corridors.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Because the packet has no coordinate for pointwise limits, external convergence, quotient function equality, countable choice, or selected infinite sums, the refusal clause is exhausted by the carrier inventory. □

Falsifiable prediction: Every accepted $\mathsf{DirichletKernel}^{\uparrow}$ partial-sum read factors through a frequency window, trigonometric coefficient ledger, $\mathsf{S}^{1\uparrow}$ evaluation route, $\mathsf{Complex}^{\uparrow}$ summation row, $\mathsf{Fourier}^{\uparrow}$ handoff, $\mathsf{RealSeries}^{\uparrow}$ or $\mathsf{CplxSer}^{\uparrow}$ provenance, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fourier, fejer-kernel, complex, s1, realseries: $\mathsf{DirichletKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a symmetric frequency window, opposite-frequency coefficient ledger, circle evaluation route, complex finite summation row, Fourier partial-sum handoff, real and complex series provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletKernel}^{\uparrow}$ packet is generated from a symmetric finite frequency window, a trigonometric coefficient ledger, displayed $\mathsf{S}^{1\uparrow}$ phase samples, a $\mathsf{Complex}^{\uparrow}$ finite summation row, a $\mathsf{Fourier}^{\uparrow}$ partial-sum handoff, $\mathsf{RealSeries}^{\uparrow}$ and $\mathsf{CplxSer}^{\uparrow}$ provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2311.1 and Definition 2311.2 seed the finite Dirichlet-kernel partial-sum surface for Fourier and series consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host pointwise Fourier convergence, quotient function equality, selected infinite sums, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window stability, coefficient symmetry, summation handoff, routing, transport, replay, provenance, and local naming for the same finite packet.

Chapter 2312

A Concrete Naming Certificate for $\mathsf{CauchyCut}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCut}^{\uparrow}$ packet records the finite bridge by which a located lower–upper cut presentation is read as a regular Cauchy real name before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits after the located-cut surface of chapter 404, the dyadic arithmetic carrier of chapter 291, the finite stream-window surface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed cut-to-stream route; it does not identify Dedekind cuts, quotient regular names, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2312.1 (Cauchy cut carrier)**.** *A* $\mathsf{CauchyCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the located lower-cut row, recording inhabited rounded lower evidence and its rational observation boundary through chapter 404.*
2. *U is the located upper-cut row, recording inhabited rounded upper evidence and the displayed separation from L.*
3. *M is the dyadic Cauchy modulus ledger: for each displayed precision row it supplies a finite dyadic bracket window whose lower and upper endpoints are read from L and U.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to ask for the bracket midpoint at a requested precision.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row extracted from the midpoint stream and the modulus ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row recording endpoint comparison, midpoint formation, and tolerance shrinking for the displayed bracket.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after L, U, M, S, Q, D have been displayed.*
8. *H stores componentwise* hsame *transport for the cut rows, modulus ledger, stream schedule, regular source, dyadic row, real seal, replay, provenance, and local naming row.*

9. *C stores displayed* Cont *replay of the route*

$$L, U, M, S, D \longmapsto Q \longmapsto R.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, M, S, Q, D, R, H, C, P, N$. It has no coordinate for host cut equality, Dedekind completeness, quotient stream equality, a selected limiting real, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCutUp.CauchyCutTasteGate single carrier alignment`

Theorem 2312.2 (Cauchy cut carrier admission)**.** *For every accepted* $\mathsf{CauchyCut}^{\uparrow}$ *carrier $K = (L, U, M, S, Q, D, R, H, C, P, N)$, the local carrier surface is exactly the displayed lower located-cut row L, upper located-cut row U, dyadic modulus ledger M, stream-window row S, regular rational source row Q, dyadic arithmetic row D, terminal real seal R, componentwise transport H, displayed replay C, provenance P, and local naming row N.*

Proof. Carrier admission is projection from Definition 2312.1. The tuple displays the lower and upper located-cut rows before any stream or regular-rational read can occur.

The remaining rows are structural or downstream rows attached to that same carrier. Since the classifier has no other coordinate, an accepted packet can expose no carrier data outside the listed rows. □

Theorem 2312.3 (Cauchy cut stream extraction)**.** *Every stream read exported by an accepted* $\mathsf{CauchyCut}^{\uparrow}$ *packet factors through the displayed route*

$$L, U, M, D \longmapsto S \longmapsto Q.$$

Thus the extracted regular Cauchy name is obtained from located lower–upper cut brackets and dyadic midpoint arithmetic, not from a quotient stream, selected representative, host real limit, or Dedekind-cut equality.

Proof. Start with the lower and upper cut rows L and U. The modulus ledger M chooses only a finite displayed bracket for the requested precision, and the dyadic row D records the midpoint and tolerance calculation inside $\mathsf{DyadicRatCore}^{\uparrow}$.

The stream row S schedules exactly those finite bracket reads. The regular rational row Q is downstream of S and therefore inherits the same cut and dyadic dependencies. No quotient stream or selected limit appears in the carrier. □

Theorem 2312.4 (Cauchy cut modulus stability)**.** *If an accepted* $\mathsf{CauchyCut}^{\uparrow}$ *carrier is transported componentwise by its H row, the dyadic Cauchy modulus ledger remains attached to the same lower–upper cut bracket route. Later precision reads may refine the displayed bracket through M and D, but they cannot replace it by an independent stream selector or by an ambient completeness principle.*

Proof. The transport row H acts componentwise on L, U, M, S, Q, D, R and the local naming row. In particular, a transported packet still exposes the same modulus ledger between the located cut rows and the stream schedule.

Refining a precision read uses M again and then replays the dyadic arithmetic row D. Since neither H nor C contains a coordinate for a separate selector or completeness theorem, the modulus route is stable under the displayed transport. □

Theorem 2312.5 (Cauchy cut classifier transport)**.** *The classifier for* $\mathsf{CauchyCut}^{\uparrow}$ *transports acceptance only through* $L, U, M, S, Q, D, R, H, C, P, N$*. Consequently every consumer that reads the packet through chapter 404, chapter 291, chapter 126, chapter 301, and chapter 38 receives the same displayed cut-to-regular-name route.*

Proof. Inspect the classifier clause in Definition 2312.1. It compares accepted packets only by the listed carrier fields and therefore cannot branch on hidden cut equality, host real equality, or quotient stream data.

The cited sibling chapters name the only public read points for those fields: located cuts for L, U, dyadic arithmetic for M, D, finite streams for S, regular rational names for Q, and the real seal for R. Transport through H, C, P, N preserves that dependency chain for every consumer. □

Theorem 2312.6 (Cauchy cut real seal nonescape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{CauchyCut}^{\uparrow}$ *carrier is public only as the endpoint of the displayed route*

$$L, U, M, S, D, Q \longmapsto R.$$

No consumer of the packet can extract a completed real by bypassing the located-cut rows, the dyadic Cauchy modulus ledger, the finite stream window, or the regular rational source row.

Proof. The seal row R occurs after L, U, M, S, Q, D in the carrier. The replay row C records the same order, so a consumer must display the located cut rows, the modulus ledger, the stream schedule, the dyadic arithmetic, and the regular rational source before reading the real seal.

The provenance and naming rows P, N attach that route to the local $\mathsf{NameCert}_{\mathsf{CauchyCut}^{\uparrow}}$ surface. Since the carrier excludes selected limits, quotient regular names, Dedekind completeness, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness, the real seal cannot escape the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCut}^{\uparrow}$ read that exports a $\mathsf{Real}^{\uparrow}$ seal factors through displayed located lower–upper cut rows, a dyadic Cauchy modulus ledger, finite stream windows, a regular rational source, dyadic arithmetic, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locatedcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchyCut}^{\uparrow}$ is not bijective to any listed sibling because it binds lower–upper located-cut evidence, a dyadic modulus ledger, stream scheduling, regular-rational extraction, terminal real sealing, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCut}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCut}^{\uparrow}}$, seed)

Scope closed: Definition 2312.1, Definition 2312.2, Definition 2312.3, Definition 2312.4, Definition 2312.5, and Definition 2312.6 seed the located-cut-to-regular-Cauchy-name bridge surface for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind-cut completeness, quotient regular-name equality, selected limits, host real equality, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-to-stream extraction, modulus stability, classifier transport, real-seal nonescape, provenance discipline, and TasteGate field faithfulness for BEDC.Derived.CauchyCutUp.

Constructive story: A $\mathsf{CauchyCut}^{\uparrow}$ packet is generated from displayed lower and upper located-cut rows, a dyadic Cauchy modulus ledger, finite stream windows, a regular rational extraction row, dyadic arithmetic, a terminal Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2313

A Concrete Naming Certificate for $\mathsf{BishopCauchyReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyReal}^{\uparrow}$ packet records a modulus-equipped Cauchy real presentation as finite BEDC certificate data before the open Real and Completion route reads the terminal seal. It sits after the regular rational readback surface of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus discipline of chapter 292, and the terminal real-name boundary of chapter 38. The packet names the constructive presentation of real data by displayed Cauchy approximants and explicit moduli; it does not make $\mathsf{Real}^{\uparrow}$ a quotient carrier, assert ambient completeness, choose a representative limit, or import countable choice.

This carrier supplies the concrete ledger row used by the Real completeness vision of chapter 9238: every Real-facing read is routed through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, explicit $\mathsf{CauchyModulus}^{\uparrow}$ schedule, $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal. The vision therefore reads this packet as finite completion data, not as quotient-stream equality, host real equality, a selected representative, or ambient completeness.

Definition 2313.1 (Bishop Cauchy real carrier)**.** *A* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row that supplies the rational approximants read by the packet.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row that records the finite tolerance-to-stage ledger used to inspect Q.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which a consumer asks for approximation data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic row used to compare requested precisions with the displayed rational approximants.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, M, S, D have been displayed.*
6. *H stores componentwise* hsame *transport for the regular source, modulus, stream schedule, dyadic ledger, real seal, replay, provenance, and local naming rows.*

7. *C stores displayed* Cont *replay of the route*

$$Q, M, S, D \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, S, D, R, H, C, P, N$. It has no coordinate for quotient equality of Cauchy streams, selected host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRealUp.TasteGate.BishopCauchyRealTasteGate single carrier alignment`

Theorem 2313.2 (Bishop Cauchy real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier $B = (Q, M, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$ *surface consists exactly of carrier admission, regular rational source exposure through Q, explicit Cauchy modulus exposure through M, finite stream-window routing through S, dyadic tolerance comparison through D, terminal real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape from quotient stream equality, selected representatives, host limit equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRealUp.BishopCauchyRealNameCertObligations`

Proof. Carrier admission is projection from Definition 2313.1. The accepted packet exposes exactly the regular rational source row, the modulus row, the stream-window schedule, the dyadic tolerance ledger, the real seal, and the structural rows.

The Cauchy presentation is read through Q, M, S, D. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular rational approximants, the $\mathsf{CauchyModulus}^{\uparrow}$ row supplies the displayed tolerance-to-stage ledger, $\mathsf{StreamName}^{\uparrow}$ bounds each read to finite windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance arithmetic.

Only after those rows are visible does the terminal $\mathsf{Real}^{\uparrow}$ seal R appear. Transport, replay, provenance, and naming preserve the same route through H, C, P, N. Since the excluded completion principles are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2313.3 (Bishop Cauchy real RegSeqRat handoff)**.** *Every regular-rational read of an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Thus a Real-facing consumer receives a modulus-controlled regular Cauchy presentation rather than a quotient Cauchy class, a chosen representative, a host limit, or a completeness principle.

Proof. Start with the displayed carrier $B = (Q, M, S, D, R, H, C, P, N)$. The source row Q is the only regular rational source available to the classifier.

The read cannot bypass M, S, D. The modulus row records which finite stages answer a requested tolerance, the stream row restricts the answer to visible windows, and the dyadic row records the endpoint arithmetic used by the tolerance comparison.

The seal row R is downstream of that route. The structural rows then transport, replay, source, and name the same packet, so every exported regular-rational read has the stated factorization and no hidden completion coordinate. □

Theorem 2313.4 (Bishop Cauchy real regular readback seal)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing seal read from an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier* $B = (Q, M, S, D, R, H, C, P, N)$ *must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* Q*, the* $\mathsf{CauchyModulus}^{\uparrow}$ *ledger* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*. The terminal seal* R *may then be read only as the endpoint of the displayed route*

$$Q, M, S, D \longmapsto R \longmapsto H, C, P, N.$$

Thus a Real-facing consumer receives the regular-readback-to-seal dependency as finite BEDC data, not as a quotient completion, a chosen representative, a host limit equality, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the carrier of Definition 2313.1. The rows Q, M, S, D precede the terminal seal R and are the only displayed source, modulus, window, and tolerance coordinates available to a Real-facing read.

Apply the obligation surface of Definition 2313.2. It identifies carrier admission, source exposure, modulus exposure, stream-window routing, dyadic comparison, terminal sealing, transport, replay, provenance, and local naming as the complete local NameCert surface.

Finally use the RegSeqRat handoff of Definition 2313.3. It forces every regular-rational read through the same $Q, M, S, D, R, H, C, P, N$ route, so the $\mathsf{Real}^{\uparrow}$ seal is terminal for that route and cannot hide quotient completion or host real equality. □

Theorem 2313.5 (Bishop Cauchy real seal boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{BishopCauchyReal}^{\uparrow}$ *carrier is public only as the endpoint of the displayed route*

$$Q, M, S, D \longmapsto R.$$

Consequently the packet exposes a constructive real presentation for the open Real and Completion route while leaving quotient completion, ambient real completeness, selected limits, host real equality, trichotomy, and countable choice outside the exported surface.

Proof. Inspect the carrier order. The terminal seal R is stored after the regular source, modulus, stream-window, and dyadic tolerance rows.

The continuation row C replays that order for consumers. A Real-facing read therefore must display the regular rational approximants, the modulus ledger, the finite observation window, and the dyadic tolerance comparison before the seal can be consumed.

The remaining structural rows H, P, N preserve and name the same boundary. Since no quotient, selected limit, host equality, trichotomy, choice, or ambient completeness coordinate occurs in the packet, the seal is exactly the displayed endpoint of the constructive Cauchy presentation. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyReal}^{\uparrow}$ real-facing read factors through a displayed $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{CauchyModulus}^{\uparrow}$ ledger, $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, cauchymodulus, streamname, dyadicratcore, real: $\mathsf{BishopCauchyReal}^{\uparrow}$ is not bijective to any listed sibling because it binds a modulus-controlled Cauchy real presentation, finite observation windows, dyadic tolerance arithmetic, terminal real sealing, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2313.1, Definition 2313.2, Definition 2313.3, and Definition 2313.5 seed the modulus-equipped Cauchy real presentation surface for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopCauchyReal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, ambient Real$^{\uparrow}$ completeness, selected limits, host real equality, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-controlled regular source exposure, finite stream-window routing, dyadic tolerance readback, real-seal boundary, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.BishopCauchyRealUp.

Constructive story: A BishopCauchyReal$^{\uparrow}$ packet is generated from a regular rational source, an explicit Cauchy modulus, finite stream windows, dyadic tolerance rows, a terminal Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2314

A Concrete Naming Certificate for $\mathsf{PointwiseLimit}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki's function-limit terminology, the $\mathsf{PointwiseLimit}^{\uparrow}$ packet records an indexwise limit route for displayed real-valued maps without importing a uniform ledger or a host function-space object. It reads continuous approximants from chapter 130, keeps the limiting observations separate from the $\mathsf{UniformLimit}^{\uparrow}$ surface of chapter 463, schedules finite observations through chapter 126, checks regular rational readbacks through chapter 301, seals values only at chapter 38, keeps source and target distance rows visible through chapter 57, and supplies the generic pointwise-limit sibling read consumed by chapter 1604 and chapter 2196.

Definition 2314.1 (PointwiseLimit carrier)**.** *A* $\mathsf{PointwiseLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Y, F, I, S, Q, R, E, H, C, K, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *context row naming the points at which the family is inspected.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *value-seal context for the observed outputs.*
3. *F is the finite ledger of displayed maps, including the* $\mathsf{ContinuousMap}^{\uparrow}$ *rows when the source family is continuous.*
4. *I is the point-index row saying that each convergence request is attached to one displayed source point and one requested tolerance.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite approximant windows inspected for that point and tolerance.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected output windows.*
7. *R is the pointwise convergence ledger asserting that the selected windows are adequate for the displayed point and tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after X, Y, F, I, S, Q, R have been displayed.*
9. *H records componentwise* hsame *transport for the source, target, map ledger, point-index row, schedule, readback, convergence ledger, value seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a point-and-tolerance request through the finite schedule, readback, pointwise ledger, and RealUp seal.*

11. *K is the* Pkg *provenance over* $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.PointwiseLimitUp.PointwiseLimitTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.PointwiseLimitUp.PointwiseLimitUp`

Theorem 2314.2 (PointwiseLimit NameCert obligations)**.** *For every accepted* $\mathsf{PointwiseLimit}^{\uparrow}$ *carrier* $P = (X, Y, F, I, S, Q, R, E, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric context, target metric and RealUp value context, displayed map ledger, point-index row, stream schedule, regular rational readback, pointwise convergence ledger, terminal value seal, transport, replay, provenance, and local naming row;*

2. *point-index visibility, requiring each convergence request to display its source point and requested tolerance through I before any schedule in S is read;*

3. *finite-window scheduling, requiring the requested approximant windows to be selected by S and checked by Q before the pointwise convergence row R can be consumed;*

4. *Real-seal routing, requiring the value row E to be downstream of X, Y, F, I, S, Q, R rather than of an ambient function-space limit;*

5. *sibling handoff compatibility, allowing* $\mathsf{BaireOneFunction}^{\uparrow}$ *and* $\mathsf{DiniContinuity}^{\uparrow}$ *consumers to read the pointwise route while keeping* $\mathsf{UniformLimit}^{\uparrow}$ *as a separate stronger ledger;*

6. *nonescape from uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, and completeness of* $\mathsf{Real}^{\uparrow}$.

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2314.1. The accepted packet exposes exactly the source and target metric rows, map ledger, point-index row, finite schedule, regular rational readback, convergence ledger, value seal, transport, replay, provenance, and local naming row.

Point-index visibility is forced by I. A consumer must name the displayed source point and tolerance before it can inspect the corresponding schedule, so the packet does not contain an unindexed host function limit.

Finite-window scheduling is then read through S and Q. The schedule selects only finite approximant windows, and the regular rational row checks those windows before R records the pointwise convergence ledger for the displayed point.

The RealUp value seal E is downstream of this route. The sibling handoff rows in K name $\mathsf{BaireOneFunction}^{\uparrow}$ and $\mathsf{DiniContinuity}^{\uparrow}$ as consumers of the pointwise route, while $\mathsf{UniformLimit}^{\uparrow}$ remains a separate stronger ledger. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local surface. □

Theorem 2314.3 (PointwiseLimit Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{BaireOneFunction}^{\uparrow}$*, or* $\mathsf{DiniContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseLimit}^{\uparrow}$ *packet reads the limiting value through the finite route*

$$X, Y, F, I, S, Q, R \longmapsto E.$$

The route first displays the source and target metric rows, then the map ledger and point-index row, then the StreamName schedule, RegSeqRat readback, and pointwise convergence ledger, and only then exposes the RealUp value seal. It exports no uniform convergence ledger, continuity-preservation theorem, host function equality, quotient function-space object, choice-selected pointwise limit, or RealUp completeness principle.

Proof. Start with a consumer request for the value seal E. By Definition 2314.1, E is a coordinate of the same finite packet as X, Y, F, I, S, Q, R, and the replay row C records the only displayed route into it.

The source and target rows X, Y are read first, followed by the displayed map ledger F and the point-index request I. The finite schedule S and regular rational readback Q then supply the observed approximant windows for that point and tolerance.

The convergence row R certifies only this pointwise route before E is exposed. Transport, replay, provenance, and local naming through H, C, K, N preserve that order for $\mathsf{BaireOneFunction}^{\uparrow}$ and $\mathsf{DiniContinuity}^{\uparrow}$ consumers, while no carrier coordinate supplies the excluded stronger objects. □

Falsifiable prediction: Every accepted $\mathsf{PointwiseLimit}^{\uparrow}$ value-seal read factors through a displayed point-index row and finite StreamName and RegSeqRat windows before any RealUp value is exposed.

Independence witness: continuousmap, uniformlimit, streamname, regseqrat, baireonefunction: PointwiseLimitUp is not bijective to any listed sibling because it jointly carries a source-point request, an approximant ledger, finite schedule and readback rows, a pointwise convergence ledger, a RealUp value seal, and explicit refusal of the UniformLimitUp ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PointwiseLimit}^{\uparrow}$ packet is generated from source and target MetricUp rows, a displayed map ledger, a point-index row, StreamName schedules, RegSeqRat readbacks, a pointwise convergence ledger, a RealUp value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2314.1, Definition 2314.2, and Definition 2314.3 seed the finite pointwise-limit certificate over displayed source points, approximant windows, regular rational readbacks, and RealUp value seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close uniform convergence, continuity preservation, host function equality, quotient function spaces, choice-selected pointwise limits, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, point-index visibility, finite-window scheduling, Real-seal routing, sibling handoff compatibility, and nonescape for BEDC.Derived.PointwiseLimitUp.

Chapter 2315

A Concrete Naming Certificate for $\mathsf{RealLogarithm}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLogarithm}^{\uparrow}$ packet names logarithm on positive real data as a finite inverse-route certificate for $\mathsf{RealExponential}^{\uparrow}$. The construction records located positive brackets, dyadic interval refinement, exponential comparison rows, and real-equality endpoint checks; it does not import a host logarithm, analytic continuation, quotient real equality, selected infinite bisection branch, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2315.1 (RealLogarithm carrier)**.** *A* $\mathsf{RealLogarithm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, I, B, E, A, O, C, R, H, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{LocatedPositiveReal}^{\uparrow}$ *row naming the positive input real and its located apartness from* 0*.*
2. *I is the* $\mathsf{OpenIntervalReal}^{\uparrow}$ *row carrying the positive bracket interval on which the inverse search is performed.*
3. *B is the finite dyadic bracket ledger, with each row recording a rational interval, midpoint comparison request, and retained positive subinterval.*
4. *E is the* $\mathsf{RealExponential}^{\uparrow}$ *inverse-route row used to compare displayed exponentials against the positive input.*
5. *A is the* $\mathsf{RealAbs}^{\uparrow}$ *control row bounding comparison error and bracket width during the finite read.*
6. *O is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *row that classifies the displayed real comparisons used by the bracket ledger.*
7. *C is the displayed* Cont *route from positive input and bracket rows through exponential comparisons to the endpoint classifier.*
8. *R is the real classifier row, read through* hsame *transport and the displayed comparison rows, that identifies the logarithm endpoint with the inverse route.*
9. *H is the componentwise* hsame *transport row for $P, I, B, E, A, O, C, R, K, N$.*

10. *K is the* Pkg *provenance over* $\mathsf{RealLogarithm}^{\uparrow}$, $\mathsf{LocatedPositiveReal}^{\uparrow}$, $\mathsf{OpenIntervalReal}^{\uparrow}$, $\mathsf{RealExponential}^{\uparrow}$, $\mathsf{RealAbs}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$ *row.*

The classifier accepts only packets whose positive input, bracket ledger, exponential inverse-route rows, absolute-error controls, Bishop comparison rows, transport, continuation, provenance, and local naming are all displayed. It has no coordinate for a host logarithm or for a hidden infinite choice of bisection branch.

Lean4 ✓ `BEDC.Derived.RealLogarithmUp.RealLogarithmTasteGate single carrier alignment`

Theorem 2315.2 (RealLogarithm NameCert obligations)**.** *For every accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *carrier* $L = (P, I, B, E, A, O, C, R, H, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$ *surface consists exactly of carrier exposure for the positive real row, the open positive interval, the dyadic bracket ledger, the* $\mathsf{RealExponential}^{\uparrow}$ *inverse-route comparisons, the* $\mathsf{RealAbs}^{\uparrow}$ *error controls, the* $\mathsf{BishopRealComparison}^{\uparrow}$ *classifier rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Any public logarithm read must pass through those rows before it can consume an endpoint value.*

Proof. Project the accepted carrier to the displayed rows $P, I, B, E, A, O, C, R, H, K, N$. The positivity and interval rows give the only admissible domain data, while the dyadic ledger records the finite bracket observations that a consumer may inspect.

The inverse comparisons are then forced through E, A, O. The exponential row supplies the only comparison with the candidate endpoint, the absolute-value row supplies the displayed error control, and the Bishop comparison row classifies the retained side of the bracket.

Finally, H, C, K, N preserve the same rows under transport, replay, provenance, and naming. Since no coordinate stores a host logarithm, an analytic-continuation datum, or a selected infinite branch, the local NameCert surface is exactly the displayed inverse-route surface. □

Theorem 2315.3 (RealLogarithm exponential inverse route)**.** *Every public read of an accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *packet factors through the positive* $\mathsf{Real}^{\uparrow}$ *input row, the located open bracket row, the finite dyadic bracket ledger, the* $\mathsf{RealExponential}^{\uparrow}$ *comparison row, the* $\mathsf{RealAbs}^{\uparrow}$ *error-control row, and the* $\mathsf{BishopRealComparison}^{\uparrow}$ *classifier row before the logarithm endpoint is consumed. Consequently the packet names logarithm as the constructive inverse route of exponential on positive real data, not as a host transcendental primitive.*

Proof. Start with a public logarithm read. The carrier gives the positive input row P and the bracket row I, so the read begins inside a located positive real interval rather than in an ambient total real function space.

The finite bracket ledger B supplies the inspected dyadic intervals. Each retained bracket is justified only by the exponential comparison row E, the absolute-error row A, and the Bishop comparison row O.

The endpoint read is then routed through C and R, with H, K, N preserving the same displayed packet under transport, provenance, and naming. Thus the logarithm value is consumed only after the exponential inverse route has been displayed, and no hidden host log or selected infinite branch is exported. □

Theorem 2315.4 (RealLogarithm sibling dependency route)**.** *Every public logarithm read of an accepted* $\mathsf{RealLogarithm}^{\uparrow}$ *packet first consumes the positive-domain row from chapter 2029, the bracket row from chapter 2133, the inverse comparison row from chapter 1816, and the error and*

classifier rows from chapter 1608 and chapter 1381. The route is a sibling dependency lattice for the displayed packet rows, not a host logarithm or ambient transcendental primitive.

Proof. Project the carrier $L = (P, I, B, E, A, O, C, R, H, K, N)$. The rows P and I are the positive-domain and open-bracket sources, so the logarithm read begins by consuming the sibling surfaces for located positive reals and open real intervals.

The inverse-route data then passes through E, A, O. The row E is the substantive $\mathsf{RealExponential}^{\uparrow}$ sibling dependency: it supplies the comparison route from chapter 1816, while the absolute-value and Bishop-comparison rows supply the error and classifier ledgers used by the finite bracket read.

Finally the structural rows C, R, H, K, N replay, transport, source, and name exactly that lattice of sibling rows. Since no carrier coordinate stores a host log or an ambient analytic primitive, the theorem records dependency routing rather than a host logarithm construction. □

Falsifiable prediction: Every $\mathsf{RealLogarithm}^{\uparrow}$ public read factors through a located positive real row, an open positive interval row, a dyadic bracket ledger, a RealExponential inverse-route row, a RealAbs error-control row, a BishopRealComparison classifier row, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: real-exponential, real-analytic, located-positive-real, open-interval-real, real-abs, bishop-real-comparison: $\mathsf{RealLogarithm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries positive-domain admission, finite inverse brackets, exponential comparison rows, absolute-error controls, Bishop comparison classification, endpoint transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLogarithm}^{\uparrow}$ packet is generated from a located positive real input, an open positive interval, finite dyadic bracket rows, RealExponential inverse-route comparisons, RealAbs error controls, BishopRealComparison classifier rows, componentwise transport, displayed continuation, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2315.1, Definition 2315.2, and Definition 2315.3 seed the constructive inverse-route surface for logarithm on positive real data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLogarithm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host logarithms, analytic continuation, quotient real equality, selected infinite bisection branches, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-domain exposure, bracket refinement, exponential inverse-route comparison, error-control, classifier nonescape, and TasteGate field faithfulness for BEDC.Derived.RealLogarithmUp.

Chapter 2316

A Concrete Naming Certificate for $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$

Following Bourbaki and Kelley on uniform completions and Bishop–Bridges on constructive completion, the $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ packet records the finite functorial handoff from a located uniform source to the regular Cauchy real seal. It sits after the located uniform completion surface of chapter 1692, the functorial completion boundary of chapter 1892, the regular rational readback of chapter 301, the finite observation windows of chapter 126, and the terminal real-name seal of chapter 38. The packet names a quotient-free L10 completion functor: it exposes source rows, located Cauchy rows, finite readback windows, real-seal handoff, transport, replay, provenance, and local naming, but no quotient completion, selected representative, countable-choice branch, host equality of limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2316.1 (Located Cauchy completion functor carrier)**.** *A* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, L, G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{LocatedUniformCompletion}^{\uparrow}$ *source row, carrying the located uniform completion request before any real-facing consumer inspects a seal.*
2. *L is the located Cauchy service row extracted from U, recording the positive-radius and finite-window evidence used to answer completion requests.*
3. *G is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *action row, recording the finite functorial map from source requests to completion-facing requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window schedule through which the consumer asks for finite data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window schedule.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after U, L, G, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the located source, functor row, observation window, regular readback, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through*

$$U, L, G \longmapsto S, R, E \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient Cauchy classes, representative selection, host limit equality, countable choice, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyCompletionFunctorUp.LocatedCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 2316.2 (Located Cauchy completion functor NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier* $F = (U, L, G, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletio}}$ *surface consists exactly of carrier admission, located uniform source exposure, located Cauchy service exposure, functorial completion action, finite observation-window routing, regular rational readback, terminal real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient completions, selected representatives, host limit equality, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2316.1. The displayed packet fixes the source row U, located service row L, functor row G, window schedule S, readback row R, terminal seal E, and structural rows H, C, P, N.

The located source is read first. The row U is the located uniform completion surface cited in chapter 1692, and L records the finite located Cauchy service used by the functor row. Thus no consumer reaches the completion functor before located evidence is displayed.

The functorial action then routes the request through G and into the finite observation rows. The schedule S and readback R are exactly the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ interfaces named above, so the handoff is a finite readback route rather than a quotient of all Cauchy names.

Finally E, H, C, P, N expose the real seal, transport the same components, replay the route, record provenance, and name the packet. Since the excluded completion principles are absent from the carrier, no local obligation can be stored outside the displayed rows. □

Theorem 2316.3 (Located Cauchy completion functor real-seal handoff)**.** *Every real-facing read of an accepted* $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ *carrier factors through the route*

$\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, N

The terminal $\mathsf{Real}^{\uparrow}$ *seal is therefore available only after the located source row, functorial completion row, finite observation window, and regular rational readback have been displayed. The handoff exports no quotient stream equality, selected limit, countable-choice branch, host real equality, or ambient completeness datum.*

Proof. Start with an accepted carrier $F = (U, L, G, S, R, E, H, C, P, N)$. The continuation row C makes U, L, G the entry surface for every completion request, so the route cannot begin at a terminal real object.

The finite readback phase is then forced by S and R. These rows are the displayed $\mathsf{StreamName}^{\uparrow}$ observation window and $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the real seal E is stored only after them in the carrier.

Componentwise transport by H, replay by C, provenance by P, and naming by N preserve the same displayed route. Since no quotient stream equality, selected limit, choice branch, host real

equality, or ambient completeness coordinate occurs in the packet, every real-facing read factors through the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ completion read factors through $\mathsf{LocatedUniformCompletion}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locateduniformcompletion, uniformcompletionfunctor, regseqrat, streamname, real: $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it binds located source evidence, functorial completion action, finite observation windows, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2316.1, Definition 2316.2, and Definition 2316.3 seed the finite quotient-free completion-functor handoff through located uniform source rows, functor rows, finite observation windows, regular readback, and terminal real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, open-cover compactness, host limit equality, Lean verification, public bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, functorial action, finite-window readback, real-seal handoff, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.LocatedCauchyCompletionFunctorUp.

Constructive story: A $\mathsf{LocatedCauchyCompletionFunctor}^{\uparrow}$ packet is generated from a located uniform completion source, a functorial completion row, finite observation windows, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2317

A Concrete Naming Certificate for $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$

The $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ packet realizes the wobble-quotient geometry branch around Definition 9462.1440 as finite trigger rows. It reads the quotient skeleton, spectral reconstruction, product decomposition, and occupancy rows of the wobble quotient geometry file, then packages them as a concrete NameCert surface beside the hash boundary of chapter 242, the spectral sequence surface of chapter 178, the network-flow route of chapter 233, and the Boolean endpoint discipline of chapter 32. The packet is a finite trigger carrier for spectral geometry consumers; it is not a host eigenvector computation, a biological assertion, a quotient-space semantic equality, or an unbounded graph search.

Falsifiable prediction: Every accepted $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ trigger read factors through wobble quotient rows, spectral hinge rows, trigger energy rows, hsame transport, Cont replay, and Pkg provenance before any downstream consumer treats the trigger as active.

Independence witness: hash, spectralseq, networkflow, bool: $\mathsf{Hash}^{\uparrow}$ records digest rows rather than quotient geometry, $\mathsf{SpectralSeq}^{\uparrow}$ records filtration pages rather than wobble trigger energy, $\mathsf{NetworkFlow}^{\uparrow}$ records capacity routes rather than quotient-fiber hinges, and $\mathsf{Bool}^{\uparrow}$ records endpoint tags rather than the spectral trigger packet; none is bijective to $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$.

Definition 2317.1 (Wobble quotient spectral trigger carrier)**.** *A* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, S, E, H, C, P, B, R, N)$$

with the following displayed rows:

1. *Q is the wobble quotient row, recording the finite f_3 quotient vertices, the one-hole quotient closure, and the quotient-product lift surface.*
2. *S is the spectral hinge row, recording the finite adjacency spectrum readback used by the quotient spectral reconstruction.*
3. *E is the trigger energy row, recording the accepted threshold comparison for the killed bit-flip spectral radius and its finite trigger tag.*
4. *H records componentwise* hsame *transport for quotient, spectral, energy, continuation, provenance, Boolean tag, route, and local naming rows.*

5. *C records displayed* Cont *replay from a consumer trigger request through*

$$Q, S, E \longmapsto H, P, B, R, N.$$

6. *P is the* Pkg *provenance over* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, $\mathsf{Hash}^{\uparrow}$, $\mathsf{SpectralSeq}^{\uparrow}$, $\mathsf{NetworkFlow}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

7. *B is the* $\mathsf{Bool}^{\uparrow}$ *trigger tag row, exposing the finite accept or refuse endpoint for the trigger request.*

8. *R is the route row assigning the read to quotient-geometry support, spectral-hinge support, trigger-energy support, or refusal.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for host spectral computation, ambient quotient equality, biological interpretation, infinite graph traversal, or a trigger that bypasses route classification.

Theorem 2317.2 (Wobble quotient spectral trigger NameCert obligations)**.** *For every accepted* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *carrier* $T = (Q, S, E, H, C, P, B, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$ *surface consists exactly of carrier admission, quotient-row exposure, spectral-hinge exposure, trigger-energy exposure, route classification, Boolean trigger tagging, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of host spectral computation, ambient quotient equality, biological interpretation, infinite graph traversal, and unclassified trigger reads.*

Proof. Carrier admission projects the displayed tuple of Definition 2317.1. The quotient row Q, spectral row S, trigger energy row E, route row R, Boolean row B, and structural rows H, C, P, N are the only visible coordinates.

The quotient row is read before the spectral hinge. This keeps the trigger tied to the finite wobble quotient skeleton and prevents a consumer from substituting an ambient quotient-space equality.

The spectral hinge and trigger energy rows are then read through S and E. Route classification by R decides whether the request is quotient support, spectral support, energy support, or refusal before the Boolean tag B is exported.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no host spectral computation, biological interpretation, infinite graph traversal, or unclassified trigger coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2317.3 (Wobble quotient spectral trigger handoff)**.** *Every accepted trigger read of a* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *packet factors through the route*

$$\mathsf{Hash}^{\uparrow},\ \mathsf{SpectralSeq}^{\uparrow},\ \mathsf{NetworkFlow}^{\uparrow},\ \mathsf{Bool}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The quotient row, spectral hinge row, trigger energy row, route row, and Boolean trigger tag are therefore visible before a downstream consumer can treat the trigger as accepted. The route exports no host eigenvector data, ambient quotient equality, biological assertion, infinite graph search, or trigger tag outside the displayed carrier.

Proof. Start with an accepted carrier $T = (Q, S, E, H, C, P, B, R, N)$. The quotient, spectral, and energy rows are the only support rows available to the trigger request, while R records how the request is classified.

The Boolean tag is read only after route classification. Thus an accepted tag in B cannot be separated from the quotient, spectral, and energy evidence that produced it.

Componentwise transport by H, continuation replay by C, provenance by P, and local naming by N preserve that same finite route. Since the excluded host and ambient objects are absent from the carrier, every accepted trigger read factors through the displayed rows. □

Theorem 2317.4 (Wobble quotient spectral trigger TasteGate obligations)**.** *The TasteGate surface for* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *requires a nontriviality row and a field-faithfulness row:*

$$\mathsf{Nontrivial}\,\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow} \quad \textit{and} \quad \mathsf{FieldFaithful}\,\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}.$$

The nontriviality row separates accepted trigger tags from refusal tags, and the field-faithfulness row requires every accepted tag to retain its quotient-row, spectral-hinge, trigger-energy, route, transport, replay, provenance, and local naming support.

Lean4 ✓ `BEDC.Derived.WobbleQuotientSpectralTriggerUp.wobbleQuotientSpectralTriggerChapterTasteGate`

Proof. Nontriviality is read from the Boolean trigger row B together with the route classifier R. An accepted tag and a refusal tag are distinct only because the carrier displays both the tag and the route that licensed it.

Field faithfulness is row preservation. A field-faithful read may transport the packet through H and replay it through C, but it cannot erase Q, S, E, R, P, N while keeping an accepted trigger tag.

Thus the required TasteGate rows are exactly the nontrivial trigger distinction and the preservation of the displayed support fields. □

Closure status for $\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2317.1, Definition 2317.2, Definition 2317.3, and Definition 2317.4 seed the finite quotient-trigger surface through quotient rows, spectral hinge rows, trigger energy rows, route classification, Boolean trigger tags, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host spectral computation, biological semantics, ambient quotient equality, infinite graph traversal, Lean verification, public bridge checking, or field-faithful TasteGate implementation.

Upgrade path: Obligation closure requires checked carrier admission, quotient exposure, spectral hinge exposure, trigger-energy exposure, route classification, Boolean tagging, support-field preservation, and TasteGate instances for BEDC.Derived.WobbleQuotientSpectralTriggerUp.

Constructive story: A $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ packet is generated from wobble quotient rows, spectral hinge rows, trigger energy rows, Boolean trigger tags, route classification, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2318

A Concrete Naming Certificate for $\mathsf{CantorIntersectionTheorem}^{\uparrow}$

Following Cantor's nested intersection theorem and Bishop–Bridges constructive analysis, the $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ packet records the theorem-shaped L10 boundary where a displayed sequence of nested located intervals is consumed through finite windows before any Real-facing seal is read. It sits downstream of the finite compactness surface of chapter 1584, the nested carrier of chapter 320, the located endpoint row of chapter 394, the dyadic interval ledger of chapter 306, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter names the finite nested-intersection theorem route only; it does not introduce a point selector, quotient real equality, open-cover compactness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2318.1 (Cantor intersection theorem carrier)**.** *A* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{NestedIntervalCompactness}^{\uparrow}$ *finite-window compactness row consumed as the theorem source.*
2. *I is the* $\mathsf{NestedInterval}^{\uparrow}$ *row naming the inspected nested interval prefix.*
3. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-location row for the same displayed prefix.*
4. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *row carrying dyadic endpoint boxes, inclusion checks, and mesh comparisons.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule read at the displayed precisions.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the located dyadic windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, I, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the theorem source, interval rows, windows, readback, seal, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite theorem request through Q, I, L, D, W, R, E in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$, $\mathsf{NestedIntervalCompactness}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$ *row.*

The classifier admits only packets with these eleven rows. No coordinate stores open-cover compactness, a selected intersection point, quotient real equality, countable choice, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 2318.2 (Cantor intersection theorem NameCert obligations)**.** *For every accepted* CantorIntersectionTheor *carrier $K = (Q, I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed nested-interval compactness row, nested prefix, located endpoints, dyadic interval ledger, stream windows, regular rational readback, real seal, transports, continuations, provenance, and local naming;*
2. *source coherence, requiring Q to be consumed through the same I, L, D, W, R, E route exposed by the finite-window compactness packet;*
3. *located mesh exactness, requiring L and D to govern the same interval prefix before any* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *row is read;*
4. *theorem-route sealing, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through Q, I, L, D, W, R before E, H, C, P, N are exposed;*
5. *ledger non-escape, excluding open-cover compactness, selected intersection points, quotient real equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 2318.1. The local naming surface has exactly the displayed rows $Q, I, L, D, W, R, E, H, C, P, N$.

The source coherence row is inherited by reading Q through chapter 1584. That source already exposes the nested prefix, located endpoint row, dyadic interval row, stream windows, regular rational readback, and real seal boundary as finite data.

The located mesh exactness row binds L and D to the same prefix before W and R can be consumed. The theorem-route sealing row then forces any Real-facing read to pass through the finite source and mesh route before it reaches E and the structural rows H, C, P, N.

Since no row in the carrier is an open-cover compactness proof, point selector, quotient-real equality witness, countable-choice schedule, trichotomy row, or ambient completeness theorem, the ledger non-escape obligation is exactly the absence of those coordinates. □

Theorem 2318.3 (Cantor intersection theorem nested route)**.** *Let $K = (Q, I, L, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier. Every theorem-facing read of the nested intersection boundary factors through*

$$\mathsf{NestedIntervalCompactness}^{\uparrow},\ \mathsf{NestedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in that order, followed only by componentwise transport, continuation replay, package provenance, and the local naming row.

Lean4 ✓ `BEDC.Derived.CantorIntersectionTheoremUp.CantorIntersectionTheoremTasteGate single carrier alignment`

Proof. Start with a theorem-facing read. By Definition 2318.2, the read first consumes the finite-window source row Q and its exposed nested prefix I.

Next the same obligation surface requires located mesh exactness. The located endpoint row L and dyadic interval row D therefore govern the inspected prefix before the stream windows W or regular rational readback R are visible.

Finally the theorem-route sealing row sends the read through R to the real seal E, and only then to H, C, P, N. The carrier has no other coordinate, so the displayed seven-object route exhausts the nested-intersection theorem boundary. □

Theorem 2318.4 (Cantor intersection theorem Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *packet receives a finite nested-window theorem route through* $Q, I, L, D, W, R, E, H, C, P, N$*. The consumer receives no open-cover compactness theorem, no selected intersection point, no quotient real equality principle, no countable-choice selector, no trichotomy row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Apply Definition 2318.3. A Real-facing consumer reaches the terminal seal only after the finite-window compactness row, nested prefix, located endpoints, dyadic boxes, stream windows, and regular rational readback have all been displayed.

The structural rows H, C, P, N preserve and name that route; they do not add analytic content. The source coherence and theorem-route sealing obligations of Definition 2318.2 keep the Real seal downstream of the same finite data.

Thus the consumer has no coordinate from which to read open-cover compactness, a selected point of the full intersection, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The exported boundary is the finite nested-window theorem route and nothing stronger. □

Theorem 2318.5 (CantorIntersection diameter-zero route)**.** *Let* $K = (Q, I, L, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *carrier. If the nested interval packet carries a displayed dyadic diameter ledger tending to zero, then every Real-facing seal read factors through* $\mathsf{NestedIntervalCompactness}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *by first reading the nested prefix* I*, located endpoint row* L*, dyadic interval and diameter rows in* D*, stream windows* W*, regular rational readback* R*, and terminal seal* E*. The route exports no selected intersection point, ambient completeness claim, quotient real equality, countable choice, trichotomy, or open-cover compactness.*

Proof. Project the carrier of Definition 2318.1. The theorem source row Q exposes the same nested interval prefix I that the diameter-zero read inspects.

Source coherence from Definition 2318.2 forces the nested prefix to be read before the Real-facing seal route is opened. The displayed dyadic diameter ledger is part of the D row, so it is tied to the same prefix and the same located endpoint row L.

Located mesh exactness then governs the zero-diameter rows. A consumer may inspect the finite $\mathsf{StreamName}^{\uparrow}$ windows W and the $\mathsf{RegSeqRat}^{\uparrow}$ readback R only after the located and dyadic rows have displayed the required mesh comparisons.

Finally, theorem-route sealing and Definition 2318.4 send the read through E and then only through H, C, P, N. None of those structural rows can add a selected point, ambient completeness principle, quotient equality, countable selector, trichotomy row, or open-cover compactness theorem. □

Falsifiable prediction: An accepted $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without first reading the nested-interval compactness row, nested prefix, located endpoint row, dyadic interval ledger, stream windows, and regular rational readback refutes the carrier surface.

Independence witness: nestedintervalcompactness, nestedinterval, locatedinterval, dyadicinterval, streamname, regseqrat, real: none is bijective to $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ because this packet jointly carries the theorem-source row, the finite nested prefix, located and dyadic endpoint rows, stream windows, regular readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ packet is generated from a NestedIntervalCompactness row, a nested interval prefix, located endpoint rows, dyadic interval boxes, StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2318.1, Definition 2318.2, Definition 2318.3, and Definition 2318.4 seed the finite Cantor nested-intersection theorem route over the L10 nested-interval boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected intersection points, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the eleven-row carrier and prove source coherence, located mesh exactness, theorem-route sealing, and ledger non-escape for the same finite packet.

Chapter 2319

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$

Following Bishop–Bridges and Bourbaki-style completion theory, the $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ packet records the finite universal-property surface for the L10 completion interface. It sits after the metric-completion carrier of chapter 1506, the uniform-completion route of chapter 1473, the functorial completion handoff of chapter 423, and the separated-metric comparison boundary of chapter 1411. The packet names the dense-source, extension, and uniqueness rows that completion consumers may cite without importing quotients, selected limits, choice, or ambient completeness.

Definition 2319.1 (Cauchy completion universal property carrier)**.** *A* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, K, T, F, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source row, recording the source object together with the finite density witness consumed by completion readers.*
2. *K is the completion carrier row, read through* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff data.*
3. *T is the complete separated target row, recording the target surface whose separatedness supports uniqueness.*
4. *F is the source map row from D into T before completion is invoked.*
5. *E is the extension map row from K into T, tied to F along the dense source row.*
6. *L is the uniqueness classifier, comparing two extension rows over the same D, K, T, F surface by separatedness rather than by quotient equality.*
7. *H records componentwise* hsame *transport for D, K, T, F, E, L and the local naming row.*
8. *C records displayed* Cont *replay of dense-source handoff, extension existence, and uniqueness reads.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, K, T, F, E, L, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host limit choice, ambient real completeness, or countable choice.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalPropertyUp.CauchyCompletionUniversalPropertyTasteGate single carrier alignment`

Theorem 2319.2 (CauchyCompletionUniversalProperty NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *carrier* $U = (D, K, T, F, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of five obligation rows: carrier exposure for the displayed tuple; dense-source handoff through D and the source map F; extension existence through the completion rows K, T, E; uniqueness up to the classifier L over the same dense-source and completion route; and ledger non-escape from choice, quotient equality, selected limits, or host limit principles.*

Proof. Carrier exposure is projection from Definition 2319.1. The tuple displays the dense source, completion carrier, complete separated target, source map, extension row, uniqueness classifier, transport, replay, provenance, and local naming row.

The dense-source handoff uses D and F. A consumer must first read the source and its finite density route before it can ask for an extension into the target.

Extension existence is carried by K, T, E. The completion carrier is read through the metric, uniform, and functorial completion rows, while T supplies the complete separated target surface. The extension row E is therefore attached to displayed completion data rather than to a selected host limit.

Uniqueness is carried by L. Since L compares extension rows over the same D, K, T, F surface and uses separatedness from T, it does not require quotient equality or representative selection.

The ledger boundary is fixed by H, C, P, N. These rows transport, replay, source, and name the same displayed packet, and no excluded host-side principle appears as a coordinate of the carrier. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalPropertyUp.CauchyCompletionUniversalPropertyCarrier namecert obligations`

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence or uniqueness while omitting the dense source, completion carrier, complete separated target, source map, extension row, uniqueness classifier, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: metriccompletion, uniformcompletion, cauchycompletionfunctor, separatedmetric: none is bijective because $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ jointly carries dense source, completion carrier, complete separated target, source map, extension, uniqueness classifier, transport, replay, provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ packet is generated from a dense source row, a completion carrier row, a complete separated target row, a source map, an extension row, a uniqueness classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$, seed)

Scope closed: Definition 2319.1 and Definition 2319.2 seed the finite L10 Cauchy completion universal-property surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, dense-source handoff, extension existence, uniqueness classification, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyCompletionUniver

Chapter 2320

A Concrete Naming Certificate for $\mathsf{CandidateSetNormalization}^{\uparrow}$

The $\mathsf{CandidateSetNormalization}^{\uparrow}$ packet realizes the candidate-set reading of chapter 9383 as finite NameCert data. It records candidate evidence, closed-term projection, Pi and application adequacy, no-infinite-reduction, obstruction, transport, continuation, provenance, and local naming rows. The packet is deliberately a candidate-set surface: it does not claim unconditional strong normalization for every closed typed MetaCIC term. Its candidate-evidence and bounded-normal checking route is read through the sibling surfaces of chapter 892 and chapter 752.

Definition 2320.1 (MetaCIC candidate-set normalization carrier)**.** *A* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, A, I, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite candidate-set evidence row, including candidate membership, soundness, and beta-closure evidence for the accepted read.*
2. *T is the closed typed term projection row, recording the empty-context typed term whose normalization read is supplied by K.*
3. *A is the application and Pi adequacy row, tying application reads and binder reads back to the same candidate-set evidence.*
4. *I is the no-infinite-reduction row obtained from the candidate-set strong-normalization route.*
5. *O is the obstruction row recording that a closed-term normalization read is refused when K is absent.*
6. *H records componentwise* hsame *transport for K, T, A, I, O and the local naming row.*
7. *C records displayed* Cont *replay from candidate-set evidence to strong normalization, closed-term projection, application and Pi adequacy, and no-infinite-reduction reads.*
8. *P is the* Pkg *provenance over* $\mathsf{CandidateSetNormalization}^{\uparrow}$, $\mathsf{MetaCICRedexFrontier}^{\uparrow}$, MetaCICNormalizationFrontie $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICOpenProblemLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, T, A, I, O, H, C, P, N$. It contains no coordinate for evidence-free closed-term normalization, componentwise application normalization, host reduction normalization, quotient reduction equality, subject-reduction discharge, or a global normalization oracle.

Lean4 (definition) `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationUp`
Lean4 ✓ `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationTasteGate single carrier alignment`

Definition 2320.2 (Candidate-set normalization frontier filter)**.** *For an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier $S = (K, T, A, I, O, H, C, P, N)$, its public frontier filter is the finite* BHist *subpacket*

$$\mathsf{Frontier}(S) = (K, T, A, I, O, H, C, P, N)$$

read in the displayed order: candidate-set evidence K, closed-term projection T, adequacy row A, no-infinite-reduction row I, obstruction row O, transport H, replay C, provenance P, and local naming N. A consumer reaches the redex-facing normalization frontier only through this subpacket, so the filter contains no coordinate for candidate-free strong normalization, closed subject reduction, arbitrary confluence, host conversion equality, or a global normalizer.

Theorem 2320.3 (MetaCIC candidate-set normalization obligations)**.** *For every accepted* $\mathsf{CandidateSetNormalizati}$ *carrier $S = (K, T, A, I, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$ *surface consists exactly of finite candidate-set evidence through K; closed-term projection through T; application and Pi adequacy through A; no-infinite-reduction through I; obstruction when the candidate-set row is absent through O; componentwise transport through H; displayed replay through C; provenance through P; local naming through N; and non-escape from unconditional MetaCIC strong normalization.*

Proof. Carrier admission is projection from Definition 2320.1. The tuple displays the candidate-set evidence, closed typed term projection, adequacy row, no-infinite-reduction row, obstruction row, transport, replay, provenance, and local name.

The positive route begins at K. Candidate-set membership, soundness, and beta-closure are read before T, A, or I can export a normalization-facing conclusion.

The closed-term projection is T, and the application and Pi adequacy route is A. Both rows consume the same candidate-set evidence rather than a componentwise strong-normalization shortcut.

The no-infinite-reduction row I is downstream of that evidence. If K is absent, O records the obstruction, so the packet refuses an evidence-free theorem over all closed typed terms.

Finally H, C, P, N transport, replay, source, and name the same finite branch. Since no global normalization oracle or quotient reduction equality is present in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2320.4 (Candidate-set normalization redex non-escape)**.** *Every redex-frontier consumer of an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier must pass through the frontier filter $K, T, A, I, O, H, C, P, N$ of Definition 2320.2. It exports no candidate-free strong normalization, no closed subject-reduction theorem, and no arbitrary confluence principle.*

Proof. Project the accepted carrier and then read its frontier filter. The first row is K, so the consumer starts from candidate-set evidence rather than from a redex-normalization oracle.

The closed-term row T and adequacy row A are downstream of the same candidate evidence. They can expose closed projection and application or Pi adequacy only through that evidence.

The no-infinite-reduction row I and obstruction row O complete the normalization-facing branch. If the candidate evidence is absent, O records refusal instead of supplying a candidate-free theorem.

Finally H, C, P, N transport, replay, source, and name the same branch. Since the filter has no subject-reduction, confluence, or global-normalizer coordinate, no consumer can export those conclusions from this packet. □

Theorem 2320.5 (Candidate-set normalization candidate-closed handoff)**.** *For every accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier, the handoff from candidate-set evidence to the closed-term projection remains mediated by* $K, T, A, I, O, H, C, P, N$*. In particular, the closed projection row* T *is not a candidate-free closed normalization theorem and cannot be reused without the candidate-set evidence row* K*.*

Proof. Read the carrier as $S = (K, T, A, I, O, H, C, P, N)$. The closed-term projection row is named as T, but its admissibility is part of the same finite packet as K.

The adequacy and no-infinite-reduction rows A and I also consume the candidate-set route. They do not create a detached closed-term theorem, because the obstruction row O records refusal when K is absent.

Transport and replay through H and C, with provenance and naming through P and N, preserve the same candidate-mediated route. Thus the closed projection remains a handoff inside the frontier filter rather than a standalone normalization principle. □

Theorem 2320.6 (MetaCIC candidate-set normalization idempotence)**.** *Let* $S = (K, T, A, I, O, H, C, P, N)$ *be an accepted* $\mathsf{CandidateSetNormalization}^{\uparrow}$ *carrier whose candidate-set packet has already been normalized through the displayed frontier-dependency rows. Applying the same normalization route again exposes the same accepted candidate rows, the same closed-term projection, the same application and Pi adequacy rows, and no additional spectral-trigger row.*

Proof. Use the carrier and obligation surface of Definition 2320.1 and Definition 2320.3. The first normalization pass can read only $K, T, A, I, O, H, C, P, N$, and the accepted candidate-set evidence is already attached to the displayed no-infinite-reduction and obstruction rows.

The frontier-dependency rows of chapter 2307 determine which candidate rows may be exposed before the closed-term projection and adequacy rows are replayed. Running the same route again therefore reuses the same dependency rows rather than introducing a new candidate branch.

Finally compare with the spectral-trigger surface of chapter 2317. A trigger row would have to be present as an additional obstruction or replay coordinate, but the $\mathsf{CandidateSetNormalization}^{\uparrow}$ classifier contains no such coordinate beyond O and C. The second pass is therefore idempotent on accepted candidate rows and creates no further spectral-trigger row. □

Falsifiable prediction: No accepted closed-term strong-normalization read through $\mathsf{CandidateSetNormalization}^{\uparrow}$ bypasses candidate evidence or candidate-set evidence.

Independence witness: metacicredexfrontier, metacicnormalizationfrontier, bounded-normal-equality-checker, metacicopenproblemledger: none is bijective because $\mathsf{CandidateSetNormalization}^{\uparrow}$ jointly carries candidate-set evidence, closed-term projection, application and Pi adequacy, no-infinite-reduction, obstruction, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateSetNormalization}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CandidateSetNormalization}^{\uparrow}$ packet is generated from finite candidate-set evidence, a closed typed term projection, application and Pi adequacy, a no-infinite-reduction row, an obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CandidateSetNormalization}\uparrow}$, seed)
Scope closed: Definition 2320.1 and Definition 2320.3 seed the finite candidate-set normalization surface.
Formal verification: FormalStatus(NameCert$_{\text{CandidateSetNormalization}\uparrow}$, encodedDef)
Lean target: `BEDC.Derived.CandidateSetNormalizationUp.CandidateSetNormalizationUp`
Bridge status: none
Not claimed: This chapter does not claim unconditional MetaCIC strong normalization, evidence-free application normalization, subject reduction, host normalization, quotient reduction equality, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked candidate-set evidence, closed-term projection, application and Pi adequacy, no-infinite-reduction, obstruction exactness, transport replay, and TasteGate field faithfulness for BEDC.Derived.CandidateSetNormalizationUp.

Chapter 2321

A Concrete Naming Certificate for $\mathsf{CompletionComparison}^{\uparrow}$

Following Bishop–Bridges metric completion and Kelley uniform-space completion, the $\mathsf{CompletionComparison}^{\uparrow}$ packet records the finite comparison surface shared by metric, filter, net, regular-Cauchy, separated, and uniform completion presentations. It sits after the metric-completion carrier of chapter 1506, the Cauchy-filter carrier of chapter 333, the Cauchy-net carrier of chapter 589, the regular-Cauchy comparison carrier of Definition 411.1, the separated-metric carrier of chapter 1411, and the uniform-completion carrier of chapter 1473. The chapter gives downstream completion consumers one named comparison surface rather than requiring them to choose a quotient-completion presentation or a selected representative prose route.

Definition 2321.1 (Completion comparison carrier)**.** *A* $\mathsf{CompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, T, R, S, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, carrying the metric completion presentation before any comparison consumer reads a completion point.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, carrying the filter presentation over the same finite tolerance requests.*
3. *T is the* $\mathsf{CauchyNet}^{\uparrow}$ *source row, carrying the net presentation and its directed observation rows.*
4. *R is the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *row, comparing the sequence-style regular Cauchy route with the filter and net routes through displayed tolerance windows.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row used to make equality and uniqueness reads pass through separated comparison data rather than host equality.*
6. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row, recording the uniform completion presentation consumed by entourage-facing readers.*
7. *H records componentwise* hsame *transport for the metric, filter, net, regular-comparison, separated, uniform, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the comparison route*

$$M, F, T, R \longmapsto S, U \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionComparison}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, choice of a convergent subnet, host equality of limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionComparisonUp.CompletionComparisonTasteGate single carrier alignment`

Theorem 2321.2 (Completion comparison NameCert obligations)**.** *For every accepted* $\mathsf{CompletionComparison}^{\uparrow}$ *carrier* $K = (M, F, T, R, S, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission, metric completion source exposure, Cauchy-filter source exposure, Cauchy-net source exposure, regular-Cauchy comparison exposure, separated-metric equality routing, uniform-completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient completion equality, selected representatives, choice-selected subnets, host equality of limits, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2321.1. The accepted packet exposes exactly the metric row M, filter row F, net row T, regular comparison row R, separated row S, uniform row U, and structural rows H, C, P, N.

The source presentations are read before any equality or uniqueness comparison. Rows M, F, T, R keep metric, filter, net, and regular-Cauchy evidence inside the same finite packet, so a consumer cannot replace the displayed comparison by a quotient object or a selected representative.

The separated and uniform rows then control the target-facing read. Row S supplies the separated comparison boundary, and row U supplies the uniform completion handoff. Rows H, C, P, N transport, replay, source, and name the same finite route.

Since the excluded host-side principles are absent from the carrier, every local obligation is one of the displayed rows. □

Theorem 2321.3 (Completion comparison filter-net correspondence)**.** *Every filter-net completion read of an accepted* $\mathsf{CompletionComparison}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow},\ \mathsf{RegularCauchyComparison}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow}$$

In particular, a consumer comparing Cauchy filters with Cauchy nets must read the shared metric row, the regular-Cauchy comparison row, the separated-metric row, and the uniform-completion handoff from Definition 2321.1; it cannot import quotient equality of completions, a selected representative sequence, a choice-selected subnet, a host limit theorem, or ambient completeness.

Proof. Begin with an accepted carrier $K = (M, F, T, R, S, U, H, C, P, N)$. The filter row F and net row T occur in the same packet as the metric row M, so their comparison begins from displayed finite completion data.

The regular comparison row R supplies the common tolerance-window route. The separated row S is the only equality-facing component, and the uniform row U is the only entourage-facing completion handoff.

Finally H, C, P, N preserve that route under transport, replay, provenance, and local naming. No extra completion object appears in K, so every filter-net read factors through the stated rows. □

Falsifiable prediction: Every accepted $\mathsf{CompletionComparison}^{\uparrow}$ consumer comparing completion presentations factors through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metriccompletion, cauchyfilter, cauchynet, regularcauchycomparison, separatedmetric, uniformcompletion: $\mathsf{CompletionComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric, filter, net, regular-Cauchy, separated, uniform, transport, replay, provenance, and local NameCert rows in one comparison packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2321.1, Definition 2321.2, and Definition 2321.3 seed the finite comparison surface tying metric, filter, net, regular-Cauchy, separated, and uniform completion rows to one NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, choice-selected subnets, host limit equality, open-cover compactness, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-row exposure, filter-net correspondence, separated equality routing, uniform handoff, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CompletionComparisonUp.

Constructive story: A $\mathsf{CompletionComparison}^{\uparrow}$ packet is generated from metric completion, Cauchy-filter, Cauchy-net, regular-Cauchy comparison, separated-metric, uniform-completion, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2322

A Concrete Naming Certificate for $\mathsf{CofinalCompactInterval}^{\uparrow}$

Following Bishop and Bridges constructive compact-interval arguments, the $\mathsf{CofinalCompactInterval}^{\uparrow}$ packet names the finite cofinal window by which a closed bounded interval request is refined to displayed locator, dyadic, stream, regular-rational, and Real-facing rows. It sits after the closed bounded interval source of chapter 1591, the compact interval locator of chapter 1891, the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives downstream interval compactness, Cousin-style finite refinement, and Heine–Borel interval consumers a named finite-window route without importing open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2322.1 (Cofinal compact interval carrier)**.** *A* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing lower and upper located endpoints, endpoint order, rational containment, dyadic containment, and the base interval naming surface.*
2. *L is the* $\mathsf{CompactIntervalLocator}^{\uparrow}$ *row, recording the located comparison and tolerance-to-window service used to refine interval requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger carrying the dyadic tolerances, endpoint cells, and finite mesh comparisons demanded by I and L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the cofinal dyadic depth chosen for the displayed request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback attached to the same interval, locator, dyadic, and stream rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the interval source, locator, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-interval request through $I, L, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CofinalCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{CompactIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$ *row.*

The classifier admits only packets with these ten rows. No coordinate stores open-cover compactness, a selected subcover, quotient real equality, countable choice, trichotomy, host interval equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CofinalCompactIntervalUp.TasteGate.CofinalCompactIntervalTasteGate single carrier alignment`

Theorem 2322.2 (Cofinal compact interval NameCert obligations)**.** *For every accepted* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *carrier* $K = (I, L, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, compact locator, dyadic tolerance ledger, stream windows, regular rational readback, real seal, transports, continuations, provenance, and local naming;*
2. *interval-source exactness, requiring every endpoint, containment, and interval-local request to enter through I before the locator row is read;*
3. *locator cofinality, requiring L and D to convert any displayed compact-interval request into a finite dyadic window deep enough for the requested observation;*
4. *real-handoff ordering, requiring every* $\mathsf{Real}^{\uparrow}$ *read to pass through I, L, D, W, R before E, H, C, P, N are exposed;*
5. *ledger non-escape, excluding open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, host interval equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 2322.1. The local naming surface has exactly the displayed rows $I, L, D, W, R, E, H, C, P, N$.

The interval-source exactness row is supplied by reading I through chapter 1591. It exposes the lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, finite-window, regular-readback, and Real-seal boundary as displayed interval data.

The locator cofinality row is then read through chapter 1891 and the dyadic tolerance ledger of chapter 291. These rows force the selected finite window to be a displayed refinement of the same interval request before W or R can be consumed.

Finally the real-handoff ordering row sends every Real-facing read through W and R before E and the structural rows H, C, P, N are exposed. Since no coordinate of the carrier is an open cover, selected subcover, quotient-real equality witness, countable-choice schedule, trichotomy row, host interval equality, or ambient completeness theorem, the ledger non-escape obligation is exactly the absence of those rows. □

Theorem 2322.3 (Cofinal compact interval Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, interval-compactness, Cousin-refinement, or Heine–Borel-style consumer admitted by an accepted* $\mathsf{CofinalCompactInterval}^{\uparrow}$ *packet reads the compact interval request through*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{CompactIntervalLocator}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in that order. The handoff exposes a finite cofinal window and its real seal; it exposes no open-cover compactness row, selected subcover, quotient real equality, countable-choice selector, trichotomy decision, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an admitted consumer. By Definition 2322.2, the read first enters through the closed bounded interval row I, so the endpoint and containment data are fixed before any compact-window refinement is available.

Next the same obligation surface requires locator cofinality. The locator row L and dyadic ledger D therefore choose a displayed finite dyadic window for the same request before the stream row W and regular readback R are visible.

The real-handoff ordering row then sends the read through R to the real seal E, followed only by transport, replay, provenance, and the local naming row. Because the carrier has no other coordinate, the displayed six-object route exhausts the consumer-facing handoff and excludes the listed compactness, choice, quotient, order, host, and completeness data. □

Falsifiable prediction: An accepted $\mathsf{CofinalCompactInterval}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ without first reading the closed bounded interval row, compact interval locator, dyadic tolerance ledger, finite stream window, and regular rational readback refutes the carrier surface.

Independence witness: closedboundedinterval, compact-interval-locator, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{CofinalCompactInterval}^{\uparrow}$ because this packet jointly carries the interval source, compact locator, cofinal dyadic window, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalCompactInterval}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a CompactIntervalLocatorUp row, a DyadicRatCoreUp tolerance ledger, StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2322.1, Definition 2322.2, and Definition 2322.3 seed the finite cofinal compact-interval route from closed bounded intervals through locator and dyadic-window rows to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient real equality, countable choice, trichotomy, host interval equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exactness, locator cofinality, finite-window routing, regular readback, Real-seal handoff, structural transport, replay, provenance, and local naming for BEDC.Derived.CofinalCompactIntervalUp.

Chapter 2323

A Concrete Naming Certificate for $\mathsf{MetrizableCompletion}^{\uparrow}$

Following Kelley and Bourbaki on metrizable topological spaces and completion, the $\mathsf{MetrizableCompletion}^{\uparrow}$ packet records the finite BEDC route by which a topology equipped with a displayed metrization can be read by the existing metric-completion surface of chapter 1506. It sits after the $\mathsf{Topology}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ carrier rows, consumes the $\mathsf{MetricCompletion}^{\uparrow}$ branch and the $\mathsf{CompleteMetric}^{\uparrow}$ seal, and exposes only the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback path needed for completion-facing consumers.

Definition 2323.1 (MetrizableCompletion carrier)**.** *A* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (T, M, Q, K, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *row, including the open-set classifier, neighborhood interface, and finite continuity rows visible to the packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *metrization row on the same carried histories, with a displayed comparison that says the topology read through T is the topology generated by the metric read through M.*
3. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *branch consumed from chapter 1506, including its Cauchy-filter or Cauchy-net evidence route.*
4. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *seal row for the completed metric-facing object, read only after Q has exposed the displayed completion branch.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row used to name the finite observation windows inspected by the completion route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for numerical tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint-seal row used when distance or convergence witnesses are compared.*
8. *H records componentwise* hsame *transport for the topology, metrization, metric-completion branch, complete-metric seal, stream, readback, endpoint, replay, provenance, and local naming rows.*

9. *C records the displayed* Cont *replay from topology admission through metrization, completion-branch admission, complete-metric sealing, stream/readback inspection, and endpoint comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{MetrizableCompletion}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, a selected representative of a Cauchy name, host topological equality, countable choice, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetrizableCompletionUp.MetrizableCompletionTasteGate single carrier alignment`

Theorem 2323.2 (MetrizableCompletion NameCert obligations)**.** *For an accepted* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier $Z = (T, M, Q, K, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology and metrization rows, compatibility of the topology row with the metric-generated topology, admission of the metric-completion branch Q, sealing by the complete-metric row K, stream/readback readiness through S and R, endpoint comparison through E, componentwise transport through H, replay through C, provenance through P, and ledger non-escape for the excluded host-side principles.*

Proof. Carrier admission is projection from Definition 2323.1. The tuple lists the topology, metrization, metric-completion branch, complete-metric seal, stream, readback, endpoint, transport, replay, provenance, and local naming rows.

The topology-metrization compatibility is local to T and M. A consumer may read open-set or neighborhood data only through the displayed metric generator row, and the packet gives no separate host-topology equality coordinate.

The completion-facing part then passes through Q and K. The row Q is exactly the metric-completion interface of chapter 1506; K seals the resulting read as complete-metric evidence only after that branch is displayed.

Finally S, R, and E provide the finite numerical route. Transport, replay, provenance, and local naming preserve the same displayed packet, so quotient completion equality, selected representatives, countable choice, open-cover compactness, host topological equality, and $\mathsf{Real}^{\uparrow}$ completeness cannot be exported. □

Theorem 2323.3 (MetrizableCompletion Cauchy seal)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetrizableCompletion}^{\uparrow}$ *carrier factors through the finite Cauchy seal*

$$T \longmapsto M \longmapsto Q \longmapsto K \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

In particular, a consumer cannot read the topological completion before the metrization row, cannot read complete-metric evidence before the $\mathsf{MetricCompletion}^{\uparrow}$ *branch, and cannot compare convergence data before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been displayed.*

Proof. Start with the topology row T. The packet admits it only together with the metrization row M, so a topological completion request first becomes a metric-facing request over the same carried histories.

Next apply the metric-completion branch. The row Q is the sole route to completion-facing data, and chapter 1506 already forces such data through a displayed Cauchy-filter or Cauchy-net branch.

The complete-metric seal K is downstream of Q. It does not select limits or quotient Cauchy names; it records that the displayed completion branch has the complete-metric evidence needed by the consumer.

The numerical seal is read last. Stream windows S precede regular-rational readback R, and the terminal endpoint comparison E is the only $\mathsf{Real}^{\uparrow}$ row offered to convergence comparisons. The transport, replay, provenance, and local naming rows retain this order, so no omitted host-side principle can enter the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetrizableCompletion}^{\uparrow}$ packet is generated from a topology row, a metrizing metric row, the metric-completion branch, a complete-metric seal, StreamName windows, RegSeqRat readback, a terminal RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2323.1, Definition 2323.2, and Definition 2323.3 seed the finite metrizable-completion surface over topology, metric, completion, complete-metric, stream, readback, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetrizableCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host topological equality, open-cover compactness, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, topology-metrization compatibility, metric-completion branch routing, complete-metric sealing, Cauchy-seal exactness, and ledger non-escape over the same displayed packet.

Chapter 2324

A Concrete Naming Certificate for $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ packet records the shrinking closed interval branch of the Real and Completion route as finite BEDC certificate data. It sits after the retained-window surface of chapter 320, the bisection discipline of chapter 1489, the constructive nested interval packet of chapter 2308, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate names the diameter-zero descent route through displayed closed intervals; it does not import compactness, choice, quotient stream equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2324.1 (Nested interval diameter-zero carrier)**.** *A* $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, J, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested closed interval source row, read from* $\mathsf{NestedInterval}^{\uparrow}$*, including closed endpoint rows, containment witnesses, and retained-window evidence.*
2. *J is the interval-halving and descent row, read from* $\mathsf{IntervalHalving}^{\uparrow}$*, recording the chosen finite bisection side, endpoint refinement, and containment descent for each inspected stage.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter budget row recording dyadic radii, endpoint comparisons, and the finite inequality checks showing the inspected widths are below the requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the nested closed interval source, halving descent, and dyadic diameter budget.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, J, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, halving descent, diameter budget, regular readback, Real seal, replay, provenance, and naming rows.*
7. *C records the displayed* Cont *replay*

$$I, J, D \longmapsto R \longmapsto E$$

through which Real and Completion consumers read the packet.

8. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{IntervalHalving}^{\uparrow}$, $\mathsf{NestedIntervalTheorem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalDiameterZero}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, J, D, R, E, H, C, P, N$. It has no coordinate for Heine–Borel compactness, open-cover compactness, a selected intersection point by choice, quotient stream equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedIntervalDiameterZeroUp.NestedIntervalDiameterZeroTasteGate single carrier alignment`

Theorem 2324.2 (Nested interval diameter-zero NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ *carrier $Z = (I, J, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalDiameterZero}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed nine-row packet, nested closed interval source exposure through I, interval-halving descent exposure through J, dyadic diameter-budget exactness through D, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, local naming through N, and non-escape from the excluded compactness, choice, quotient, trichotomy, and ambient-completeness principles.*

Proof. Project the carrier of Definition 2324.1. The rows $I, J, D, R, E, H, C, P, N$ are the whole certificate surface, so every local obligation must be read from one of those rows.

The constructive interval data are I, J, D. The source row I supplies the closed nested intervals, J records the displayed halving and descent route, and D records the dyadic diameter budget that makes each requested tolerance visible as finite endpoint arithmetic.

The completion-facing read is then forced through R and E. The regular rational row R is assembled from the same nested source, descent, and diameter budget, and the Real seal E is downstream of that readback.

Finally the structural rows H, C, P, N transport, replay, source, and name the same packet. Since no row contains compactness, a selected point, quotient stream equality, trichotomy, or ambient completeness, the listed obligations exhaust the local NameCert. □

Theorem 2324.3 (Nested interval diameter-zero Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{IntervalHalving}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads the nested closed interval source I, the halving descent row J, the dyadic diameter budget D, the regular rational readback R, and only then the Real seal E with its transport, replay, provenance, and local naming rows.

Proof. Begin with a Real-facing consumer. The only Real seal coordinate in the carrier is E, and the carrier places E downstream of I, J, D, R.

The source and descent rows provide the shrinking interval data. Row I exposes the nested closed interval windows, while row J records the interval-halving route that preserves containment at the inspected stages.

The diameter-zero condition is read through D. Its dyadic budget compares widths with the requested tolerance and supplies the arithmetic needed for the $\mathsf{RegSeqRat}^{\uparrow}$ readback R.

The structural rows then preserve the route into E. Thus the Real seal is available only through the displayed finite path, and the packet cannot be repacked as Heine–Borel compactness, an open-cover argument, a choice-selected point, quotient stream equality, trichotomy, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ Real-facing read factors through a nested closed interval source, an interval-halving descent row, a dyadic diameter budget, a RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: nestedinterval, intervalhalving, nested-interval-theorem, dyadicratcore, regseqrat, real: $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ is not bijective to any listed sibling because it binds closed interval nesting, halving descent, diameter-zero dyadic budgets, regular rational readback, Real sealing, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalDiameterZero}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalDiameterZero}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2324.1, Definition 2324.2, and Definition 2324.3 seed the finite shrinking closed interval route from nested interval source and halving descent through dyadic diameter budgets to RegSeqRat readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalDiameterZero}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel, open-cover compactness, a selected intersection point by choice, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested closed interval exposure, interval-halving descent stability, dyadic diameter-budget exactness, RegSeqRat readback, Real-seal non-escape, transport and replay stability, provenance, and local naming for BEDC.Derived.NestedIntervalDiameterZeroUp.

Constructive story: A $\mathsf{NestedIntervalDiameterZero}^{\uparrow}$ packet is generated from a nested closed interval source row, an interval-halving descent row, a dyadic diameter budget, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2325

A Concrete Naming Certificate for ClosedParallelDiamondPremiseLedger$^{\uparrow}$

The ClosedParallelDiamondPremiseLedger$^{\uparrow}$ packet realizes the closed parallel diamond premise boundary of chapter 9378 as finite BEDC data. It sits beside the confluence audit packet of chapter 735, the closed-normal seal of chapter 741, the residual-substitution boundary of chapter 2181, and the parallel confluence audit of chapter 744. The packet records the premise ledger consumed by the conditional MetaCIC confluence bridge without asserting the missing diamond proof.

Definition 2325.1 (Closed parallel diamond premise ledger carrier)**.** *A* ClosedParallelDiamondPremiseLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, S, D, C, N, R, A, H, K, Q, G)$$

with the following displayed rows:

1. *P is the closed parallel-step source row for the two displayed* BetaParallel *branches.*
2. *S is the residual-substitution premise row, read through* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*, that keeps substitution compatibility visible as an input.*
3. *D is the closed parallel diamond premise row naming* BetaParallelDiamondClosed *as a ledger entry rather than as a theorem proved by this packet.*
4. *C is the conditional closed-star confluence handoff row consumed by the MetaCIC confluence bridge after S and D are present.*
5. *N is the closed-normal seal row read through* ClosedNormalConfluenceSeal$^{\uparrow}$ *for the unconditional normal-source fragment.*
6. *R is the confluence audit packet row read through* MetaCICConfluenceAuditPacket$^{\uparrow}$ *as the larger consumer of the conditional boundary.*
7. *A is the parallel confluence audit row read through* ParallelConfluenceAudit$^{\uparrow}$ *to keep the parallel-step, substitution, diamond, and star-confluence handoffs in one audit map.*
8. *H records componentwise* hsame *transport for the premise, sibling, audit, handoff, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay*

$$P, S, D \longmapsto C \longmapsto R, A, N$$

for consumers of the conditional confluence bridge.

10. *Q is the* Pkg *provenance over* ClosedParallelDiamondPremiseLedger$^{\uparrow}$, MetaCICConfluenceAuditPacket$^{\uparrow}$, ClosedNormalConfluenceSeal$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, ParallelConfluenceAudit$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *G is the local* NameCert$_{\text{ClosedParallelDiamondPremiseLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, S, D, C, N, R, A, H, K, Q, G$. It has no coordinate proving BetaParallelDiamondClosed*, no proof of* BetaParallelSubstitutionCompatibility*, no arbitrary closed Church–Rosser theorem, no host residual semantics, and no unchecked bridge discharge.*

Lean4 (definition) `BEDC.Derived.ClosedParallelDiamondPremiseLedgerUp.ClosedParallelDiamondPremiseLedgerUp`

Theorem 2325.2 (Closed parallel diamond premise ledger obligations)**.** *For every accepted* ClosedParallelDiamondP *carrier $L = (P, S, D, C, N, R, A, H, K, Q, G)$, the local* NameCert$_{\text{ClosedParallelDiamondPremiseLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet, closed parallel-step source exposure through P, residual-substitution premise visibility through S, closed parallel diamond premise visibility through D, conditional closed-star handoff visibility through C, closed-normal fragment separation through N, confluence audit consumption through R, parallel audit consumption through A, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through Q, local naming through G, and non-escape from premise-free closed confluence.*

Proof. Project the carrier of Definition 2325.1. The rows $P, S, D, C, N, R, A, H, K, Q, G$ are the whole ledger, so every local obligation is read from a displayed row.

The premise boundary is S, D. Row S keeps residual substitution compatibility visible, and row D names the closed parallel diamond premise as a ledger entry rather than as a discharged theorem.

The consumer-facing bridge row is C. It may be read only after the premise rows are present, and the audit rows R and A record that this conditional handoff is consumed by the confluence audit map and the parallel audit map.

The closed-normal row N remains a sibling fragment rather than a replacement for the premise ledger. The structural rows H, K, Q, G transport, replay, source, and name the same finite packet, so no hidden proof of the diamond, substitution compatibility, or arbitrary closed Church–Rosser can enter the certificate. □

Theorem 2325.3 (Closed parallel diamond premise conditional boundary)**.** *Every conditional closed-star confluence read through an accepted* ClosedParallelDiamondPremiseLedger$^{\uparrow}$ *carrier $L = (P, S, D, C, N, R, A, H, K, Q, G)$ must expose both the residual-substitution premise row S and the closed parallel diamond premise row D. The packet may be cited only as the finite route*

$$P, S, D, C, N, R, A, H, K, Q, G$$

for a conditional handoff; it is not an unconditional diamond proof, an unconditional confluence proof, or a premise-free closed Church–Rosser certificate.

Proof. Project the carrier of Definition 2325.1. The rows $P, S, D, C, N, R, A, H, K, Q, G$ are displayed explicitly, and the conditional handoff row C is downstream of the premise rows S and D.

Apply the obligation theorem Definition 2325.2. It states that residual-substitution premise visibility and closed parallel diamond premise visibility are part of the exact local NameCert surface, alongside the closed parallel-step source, sibling audit rows, transport, replay, provenance, and local naming.

Therefore a consumer reading closed-star confluence through this packet must carry S and D as premises. Since the carrier has no row proving either premise and no row for arbitrary closed confluence, the packet cannot be cited as an unconditional diamond or confluence proof. □

Theorem 2325.4 (Closed parallel diamond premise vision handoff)**.** *Every confluence-vision consumer of* $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$ *must read the closed parallel-step source row P, residual-substitution premise row S, closed parallel diamond premise row D, conditional closed-star handoff row C, closed-normal fragment row N, confluence audit row R, and parallel audit row A before any closed-star confluence claim is available. The packet therefore refuses premise-free closed confluence: it exposes the vision handoff only as the displayed route*

$$P, S, D \longmapsto C \longmapsto N, R, A.$$

Proof. Use the carrier definition Definition 2325.1. Its rows P, S, D, C, N, R, A are displayed separately, so a vision consumer cannot collapse the source, premise, conditional handoff, closed-normal fragment, and audit consumers into a single unnamed confluence fact.

Apply Definition 2325.2. The local NameCert surface includes closed parallel-step source exposure, residual-substitution premise visibility, closed parallel diamond premise visibility, conditional handoff visibility, closed-normal fragment separation, confluence audit consumption, and parallel audit consumption.

Thus the confluence-vision marker is read through the same finite ledger as the concrete chapter. Since the carrier has no row proving BetaParallelDiamondClosed, no row proving residual-substitution compatibility, and no row for arbitrary closed Church–Rosser, the handoff cannot be used as premise-free closed confluence. □

Falsifiable prediction: A conditional MetaCIC confluence bridge that consumes ClosedParallelDiamondPremiseLed must display closed parallel-step source rows, residual-substitution premise rows, the closed parallel diamond premise row, closed-star handoff rows, closed-normal seal rows, confluence audit rows, parallel audit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metacicconfluenceauditpacket, closednormalconfluenceseal, metacicresidualsubstitutionboundary, parallelconfluenceaudit: $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed parallel-step source, the residual-substitution premise row, the closed parallel diamond premise row, the conditional closed-star handoff, sibling audit consumers, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2325.1 and Definition 2325.2 seed the finite closed parallel diamond premise ledger consumed by the conditional MetaCIC confluence bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.ClosedParallelDiamondPremiseLedgerUp.ClosedParallelDiamondPremiseLedgerUp`

Bridge status: none

Not claimed: This chapter does not prove BetaParallelDiamondClosed, BetaParallelSubstitutionCompatibility, arbitrary closed Church–Rosser, host residual semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed parallel-step exposure, residual-substitution premise visibility, closed parallel diamond premise visibility, conditional handoff exactness, sibling audit separation, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.ClosedParallelDiamondPremiseLedgerUp.

Constructive story: A ClosedParallelDiamondPremiseLedger$^{\uparrow}$ packet is generated from closed parallel-step source rows, residual-substitution premise rows, a closed parallel diamond premise row, a conditional closed-star confluence handoff, closed-normal seal rows, confluence audit rows, parallel audit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2326

A Concrete Naming Certificate for $\mathsf{UniformConvergenceSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformConvergenceSpace}^{\uparrow}$ packet records uniform convergence over displayed real-valued function packets by finite domain rows, function-family windows, uniform tail ledgers, regular rational readbacks, and terminal real seals. It consumes the uniform-limit route of chapter 463, the graph surface of chapter 130, the terminal real-name seal of chapter 38, the regular rational handoff of chapter 301, the finite window discipline of chapter 126, and the completion boundary of chapter 1506. The packet names the convergence-space interface itself, not a quotient function space, an ambient supremum norm, a host-selected limit function, or an external completeness principle.

Definition 2326.1 (Uniform convergence space carrier)**.** *A* $\mathsf{UniformConvergenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(D, F, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the displayed domain row, exposing only the finite observation coordinates and equality tests consumed by the function packets.*
2. *F is the displayed real-valued function family row, read through* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{ContinuousMap}^{\uparrow}$ *graph data over the same domain row.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule that binds each finite domain observation to the function-family window inspected for a requested tolerance.*
4. *T is the uniform tail ledger, assigning each requested tolerance to a single family-wide threshold and recording the accepted window comparisons beyond that threshold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same domain observations, function windows, and uniform tail ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row reached only after the regular rational readback has displayed the scheduled tail comparisons.*
7. *H records componentwise* hsame *transports for the domain row, function family, window schedule, uniform tail ledger, readback, endpoint seal, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *routes from any convergence-space consumer request through D, F, W, T, R, E in that order.*

9. *P is the* Pkg *provenance over* $\mathsf{UniformConvergenceSpace}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, MetricCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceSpace}^{\uparrow}}$ *row.*

The classifier admits only finite packets with these ten rows. No coordinate stores a quotient function space, ambient supremum metric, selected limit function, pointwise-only convergence witness, choice principle, or external completeness theorem.

Lean4 ✓ `BEDC.Derived.UniformConvergenceSpaceUp.UniformConvergenceSpaceTasteGate single carrier alignment`

Theorem 2326.2 (Uniform convergence space NameCert obligations)**.** *For every accepted* UniformConvergenceSpac *carrier $(D, F, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformConvergenceSpace}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed domain row, function-family row,* $\mathsf{StreamName}^{\uparrow}$ *window schedule, uniform tail ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *endpoint seal, transports, continuations, provenance, and local naming;*

2. *uniform-window classifier stability, requiring transported and replayed consumers to inspect the same domain observations, function-family windows, and scheduled tolerance rows before a convergence-space read is exposed;*

3. *tail-modulus exactness, requiring every endpoint-facing read to use the single family-wide threshold recorded in T rather than a pointwise threshold or an ambient supremum argument;*

4. *endpoint-seal handoff, requiring the route from D, F, W, T through R and then E before any* $\mathsf{Real}^{\uparrow}$ *endpoint row is consumed;*

5. *ledger non-escape, excluding quotient function spaces, ambient supremum metrics, selected limit functions, pointwise-only convergence witnesses, choice, and external completeness.*

Proof. Project the carrier of Definition 2326.1. The domain row, function-family row, window schedule, uniform tail ledger, readback, endpoint seal, transport, continuation, provenance, and local naming are exactly the displayed packet coordinates.

Uniform-window classifier stability is componentwise. The row H transports the domain observations, function windows, and tail ledger without changing which finite windows are named, while C replays the same route from the consumer request through D, F, W, T, R, E.

Tail-modulus exactness follows from the role of T. A requested tolerance is assigned one family-wide threshold before the readback row R can be consumed, so a pointwise threshold or an ambient supremum argument has no carrier coordinate.

The endpoint-seal handoff is forced by the packet order. The regular rational readback R is the only row between the uniform tail ledger and the endpoint seal E, and the $\mathsf{Real}^{\uparrow}$ endpoint is exposed only after that readback has displayed the scheduled comparisons.

The refusal clause is an inventory check on the carrier. Since none of the ten coordinates is a quotient function space, ambient supremum metric, selected limit function, pointwise-only convergence witness, choice row, or external completeness theorem, the local certificate exports none of those principles. □

Falsifiable prediction: An accepted $\mathsf{UniformConvergenceSpace}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ endpoint seal, $\mathsf{UniformLimit}^{\uparrow}$ handoff, or $\mathsf{MetricCompletion}^{\uparrow}$ boundary without first reading the domain row, function-family row, $\mathsf{StreamName}^{\uparrow}$ window schedule, and uniform tail ledger refutes the carrier surface.

Independence witness: uniformlimit, continuousmap, real, regseqrat, streamname, metriccompletion: none is bijective to $\mathsf{UniformConvergenceSpace}^{\uparrow}$ because this packet jointly carries a domain row, displayed real-valued function-family row, StreamName window schedule, uniform tail ledger, regular readback, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2326.1 and Definition 2326.2 seed the finite uniform convergence space certificate over displayed domain rows, real-valued function-family windows, uniform tail ledgers, regular rational readbacks, endpoint seals, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient function spaces, ambient supremum metrics, selected limit functions, pointwise-only convergence witnesses, choice, external completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-window classifier stability, tail-modulus exactness, endpoint-seal handoff, and ledger non-escape for BEDC.Derived.UniformConvergenceSpaceUp.

Constructive story: A $\mathsf{UniformConvergenceSpace}^{\uparrow}$ packet is generated from a displayed domain row, displayed real-valued function-family row, StreamNameUp window schedule, uniform tail ledger, RegSeqRatUp readback, RealUp endpoint seal, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2327

A Concrete Naming Certificate for $\mathsf{LocatedRealIntervalHull}^{\uparrow}$

Following Moore interval analysis and Bishop-style located real data, the $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ packet names the finite hull read that encloses a displayed family of $\mathsf{Real}^{\uparrow}$ names by located lower and upper endpoint rows. It sits after the terminal real-name seal of chapter 38, the regular rational source of chapter 301, the finite observation window of chapter 126, the dyadic tolerance ledger of chapter 291, the endpoint-ledger interface of chapter 394, and the rational readback surface of chapter 1668. The packet records a finite enclosure hull for Real and Completion consumers; it does not choose representatives, quotient real equality, an ambient least interval, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2327.1 (Located real interval hull carrier)**.** *A* $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (R, S, Q, D, A, J, L, U, E, T, C, P, N)$$

with the following displayed rows:

1. *R is the finite family of* $\mathsf{Real}^{\uparrow}$ *names whose common enclosure is being read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the displayed real names.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readbacks at the requested precision.*
5. *A is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *row recording the rational approximants read from Q under the tolerance ledger D.*
6. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row whose displayed lower and upper bounds form the candidate hull.*
7. *L and U are the lower and upper endpoint ledger rows read through J.*
8. *E is the enclosure ledger saying that every inspected approximation in A lies between L and U with the displayed dyadic tolerance.*
9. *T records componentwise* hsame *transport for the real-name family, stream windows, regular readbacks, dyadic tolerance, rational approximants, located interval row, endpoint rows, enclosure ledger, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a finite hull request through $R, S, Q, D, A, J, L, U, E, T, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntervalHull}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalHull}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It has no coordinate for a host least interval, selected real representatives, quotient equality of real names, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalHullUp.TasteGate.LocatedRealIntervalHullTasteGate single carrier alignment`

Theorem 2327.2 (Located real interval hull NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ *carrier* $H = (R, S, Q, D, A, J, L, U, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalHull}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real-name family, stream windows, regular readbacks, dyadic tolerance, rational approximants, located interval, endpoint ledgers, enclosure ledger, transport, replay, provenance, and local naming rows;*
2. *source compatibility, requiring S, Q, D, A to be read from the same finite family of* $\mathsf{Real}^{\uparrow}$ *names named by R;*
3. *endpoint admission, requiring J, L, U to be displayed located-interval rows rather than host-selected endpoints;*
4. *hull enclosure exactness, requiring E to bind every inspected rational approximation in A between L and U under the tolerance ledger D;*
5. *replay stability, requiring T and C to preserve the same hull route under componentwise transport and later finite-window reads;*
6. *ledger refusal, excluding host least intervals, selected representatives, quotient real equality, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedRealIntervalHullUp.LocatedRealIntervalHullNamecertObligations`
Lean4 ✓ `BEDC.Derived.LocatedRealIntervalHullUp.LocatedRealIntervalHullCarrier namecert obligations`

Proof. Project the carrier of Definition 2327.1. The projection exposes the real-name family, finite stream windows, regular rational readbacks, dyadic tolerance ledger, rational approximation row, located interval endpoint row, enclosure ledger, and the structural rows T, C, P, N.

Source compatibility follows from the displayed route through chapter 126, chapter 301, chapter 291, and chapter 1668. The rows S, Q, D, A are all attached to the same family R, so a consumer cannot insert a second real-name source.

Endpoint admission is the row J together with L and U. These rows are the visible located interval endpoints supplied by chapter 394, not host-selected real representatives.

The hull enclosure clause is the ledger E. It compares only the displayed rational approximation rows against the displayed endpoint rows at the dyadic tolerance named by D.

Finally T, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host least interval, selected representatives, quotient real equality, trichotomy, countable choice, or ambient completeness, the local NameCert surface cannot export any of those objects. □

Theorem 2327.3 (Located real interval hull finite endpoint readback)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ *carrier factors through the finite endpoint route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RealRationalApproximation}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow}.$$

The consumer receives the displayed lower and upper endpoint rows L, U *only together with the finite windows, readbacks, tolerance ledger, rational approximation row, enclosure ledger, transport, replay, provenance, and local naming data. It receives no quotient real equality, no selected representative family, no host least hull, and no completeness witness.*

Proof. Start with an accepted carrier $H = (R, S, Q, D, A, J, L, U, E, T, C, P, N)$. The only source of Real-facing data is the displayed family R, and the only observations opened from it are the finite windows S.

The readback stage is Q, D, A. Regular rational rows are read through the windows, the dyadic ledger fixes the tolerance budget, and the rational approximation row records the inspected approximants.

The endpoint stage is J, L, U, E. The located interval row supplies the lower and upper endpoint ledgers, and the enclosure ledger binds the approximants to those endpoints at the displayed tolerance.

The structural rows then preserve the route. Componentwise transport, continuation replay, provenance, and the local NameCert row do not add another endpoint source, so every consumer receives exactly the finite endpoint readback surface and none of the excluded quotient, selection, host-hull, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalHull}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalHull}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2327.1, Definition 2327.2, and Definition 2327.3 seed the finite located interval hull route over displayed $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, and $\mathsf{LocatedInterval}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalHull}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, host least intervals, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, endpoint admission, hull enclosure exactness, replay stability, and ledger refusal for BEDC.Derived.LocatedRealIntervalHullUp.

Constructive story: A $\mathsf{LocatedRealIntervalHull}^{\uparrow}$ packet is generated from displayed RealUp names, StreamNameUp windows, RegSeqRatUp readbacks, a DyadicRatCoreUp tolerance ledger, a RealRationalApproximationUp approximation row, a LocatedIntervalUp endpoint row, lower and upper endpoint ledgers, an enclosure ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2328

A Concrete Naming Certificate for $\mathsf{LipschitzConstant}^{\uparrow}$

Following Bishop and Bridges and standard metric-space analysis, a Lipschitz constant is useful in constructive arguments only when the constant, the source and target metrics, and the finite tolerance conversion route are displayed together. The $\mathsf{LipschitzConstant}^{\uparrow}$ packet names that surface as a finite L10 object. It refines the bound ledger already consumed by chapter 410, reads metric rows from chapter 57, reads the displayed scalar through chapter 38, and hands the resulting modulus to chapter 369. Its downstream consumers include $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ surfaces; it is not a host real inequality, a quotient of maps, a differentiability claim, or an ambient completeness theorem.

Definition 2328.1 (Lipschitz constant carrier)**.** *A* $\mathsf{LipschitzConstant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(K, X, Y, D, F, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the displayed* $\mathsf{Real}^{\uparrow}$ *constant row, read as a nonnegative finite bound rather than as an ambient host scalar.*
2. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
3. *D records the dyadic distance-comparison ledger: every source-distance tolerance request and every target-distance bound check is routed through displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons.*
4. *F is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row whose graph and bound use the same K, X, Y, D data.*
5. *M is the finite modulus handoff row into* $\mathsf{UniformModulus}^{\uparrow}$ *and* $\mathsf{UniformContinuity}^{\uparrow}$*, sending a requested target tolerance to a displayed source tolerance through K and D.*
6. *H records componentwise* hsame *transport for the constant, metric rows, dyadic comparisons, source map, modulus handoff, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from any continuity, completion, or map consumer through K, X, Y, D, F, M in that order.*
8. *P is the* Pkg *provenance over* $\mathsf{LipschitzConstant}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LipschitzMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzConstant}^\uparrow}$ *row.*

The classifier accepts only packets with these ten rows. It has no coordinate for a detached host real bound, pointwise-only continuity, quotient map equality, differentiability, selected limits, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LipschitzConstantUp.LipschitzConstantTasteGate single carrier alignment`

Theorem 2328.2 (Lipschitz constant modulus handoff)**.** *For every accepted* $\mathsf{LipschitzConstant}^\uparrow$ *carrier* $(K, X, Y, D, F, M, H, C, P, N)$*, every* $\mathsf{UniformContinuity}^\uparrow$ *or* $\mathsf{UniformModulus}^\uparrow$ *consumer that reads the constant must first read the displayed* $\mathsf{Real}^\uparrow$ *row* K*, the source and target* $\mathsf{Met}^\uparrow$ *rows* X, Y*, the dyadic comparison ledger* D*, and the source* $\mathsf{LipschitzMap}^\uparrow$ *row* F*. The exported modulus row* M *is therefore a finite handoff from Lipschitz control to uniform continuity, not an independent continuity principle.*

Proof. Project the carrier of Definition 2328.1. The only constant row is K, and the only metric rows are X and Y, so any consumer that mentions the bound must read those displayed rows.

The dyadic comparison ledger D supplies the tolerance arithmetic. It records how a target tolerance request is converted into a source-distance request under the displayed bound, so the request cannot bypass the finite $\mathsf{DyadicRatCore}^\uparrow$ comparisons.

The source map row F attaches that arithmetic to a $\mathsf{LipschitzMap}^\uparrow$ packet. The modulus row M is consumed only after K, X, Y, D, F have been displayed, and H and C preserve the same route under transport and replay.

Thus the uniform-continuity read is exactly the finite handoff named by M. No carrier coordinate supplies pointwise-only continuity, detached host estimates, quotient function equality, differentiability, selected limits, choice, or ambient completeness. □

Theorem 2328.3 (Lipschitz constant NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{LipschitzConstant}^\uparrow}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed constant, source metric, target metric, dyadic comparison, source map, modulus, transport, continuation, provenance, and local naming rows;*
2. *nonnegative-bound exposure, requiring* K *to be read as the displayed* $\mathsf{Real}^\uparrow$ *constant before any target-distance estimate is consumed;*
3. *dyadic comparison exactness, requiring all tolerance conversions to pass through* D*;*
4. *modulus handoff, requiring the* $\mathsf{UniformContinuity}^\uparrow$ *and* $\mathsf{UniformModulus}^\uparrow$ *reads to pass through* K, X, Y, D, F, M*;*
5. *ledger non-escape, excluding detached host constants, quotient map equality, differentiability, selected limits, countable choice, and ambient metric or real completeness.*

Proof. Carrier admission is immediate from Definition 2328.1; the packet exposes exactly the ten listed rows.

The bound-exposure and dyadic-exactness obligations follow from the positions of K and D. A distance estimate has no row to inspect except the displayed real constant and the dyadic comparison ledger attached to it.

The handoff obligation is Definition 2328.2. It routes every uniform-continuity or uniform-modulus consumer through the same finite rows before M is exported.

The refusal clause is an inventory check on the carrier. Since none of the displayed rows is a host constant, a quotient map, differentiability data, selected-limit data, choice data, or completeness evidence, the local NameCert exports none of those principles. □

Theorem 2328.4 (Lipschitz constant uniform continuity route)**.** *Let* $(K, X, Y, D, F, M, H, C, P, N)$ *be an accepted* $\mathsf{LipschitzConstant}^{\uparrow}$ *carrier. Every* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer that uses the displayed Lipschitz constant factors through*

$\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, BHist, hsame, Cont, Pkg, Na

In particular, the constant cannot be consumed as a parameter echo detached from the finite modulus route.

Proof. Start with the accepted carrier and apply Definition 2328.3. Any consumer sees only the displayed constant, metric rows, dyadic comparisons, source map, modulus handoff, transport, replay, provenance, and naming rows.

For $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{UniformContinuity}^{\uparrow}$ consumers, Definition 2328.2 forces the read through K, X, Y, D, F, M before continuity information is exposed.

For $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ consumers, the same route is followed by the downstream regular-readback or completion-facing surface. The constant is still tied to the source map and dyadic tolerance ledger before any real or completion seal can be inspected.

Therefore the displayed route is the only available consumer path. The packet offers no detached parameter copy, host real inequality, quotient map equality, selected limit, countable-choice schedule, differentiability row, or ambient completeness evidence. □

Definition 2328.5 (Mean-value inequality carrier)**.** *A mean-value inequality carrier over a* $\mathsf{LipschitzConstant}^{\uparrow}$ *packet is a finite* BHist *packet*

$$V = (X, Y, G, L, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source and target reads use X and Y.*
3. *L is an accepted* $\mathsf{LipschitzConstant}^{\uparrow}$ *row binding the same source metric, target metric, graph, and displayed constant.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for source-distance reads, target-distance reads, and multiplication by the displayed Lipschitz ledger.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal reached only after X, Y, G, L, D have all been displayed.*
6. *H records componentwise* hsame *transport for the two metric rows, continuous graph, Lipschitz row, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a source-distance request through the Lipschitz handoff to the target-distance bound.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}$ *row for this consequence packet.*

The classifier accepts only packets whose source and target metric rows, continuous graph row, Lipschitz constant row, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows are displayed together. It has no coordinate for a differentiability theorem, tangent vector, selected maximum point, quotient map equality, countable choice, or ambient Real completeness.

Theorem 2328.6 (Mean-value inequality Lipschitz handoff)**.** *For every accepted mean-value inequality carrier $V = (X, Y, G, L, D, R, H, C, P, N)$, every target-distance read through Y is bounded by the displayed Lipschitz ledger in L multiplied by the corresponding source-distance read through X. The comparison is consumed through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D and sealed only by the displayed* $\mathsf{Real}^{\uparrow}$ *row R; it is not a differentiability claim or an ambient host inequality.*

Proof. Project the carrier of Definition 2328.5. The only available source-distance and target-distance rows are X and Y, and the only graph row connecting them is the displayed $\mathsf{ContinuousMap}^{\uparrow}$ row G.

Apply Definition 2328.4 to the $\mathsf{LipschitzConstant}^{\uparrow}$ row L. That route forces the consumer through the displayed constant, the same metric rows, the source map binding, and the finite modulus handoff before any continuity information is read.

The dyadic ledger D supplies the arithmetic comparison: the target-distance read is checked against the displayed Lipschitz ledger times the source-distance read. The Real seal R records only that routed comparison, while H, C, P, N transport, replay, source, and name the same packet. Since no row supplies derivatives, selected extrema, quotient equality, choice, or ambient completeness, the consequence is exactly the Lipschitz handoff. □

Falsifiable prediction: An accepted $\mathsf{LipschitzConstant}^{\uparrow}$ consumer that reaches $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegularCauchyLipschitzMap}^{\uparrow}$, or $\mathsf{MetricCompletion}^{\uparrow}$ without first reading K, X, Y, D, F, M refutes the carrier surface.

Independence witness: lipschitzmap, metric, real, uniformmodulus, uniformcontinuity: none is bijective to $\mathsf{LipschitzConstant}^{\uparrow}$ because this packet jointly carries the displayed constant, the two metric rows, dyadic tolerance comparisons, source map binding, modulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzConstant}^{\uparrow}}$.

Origin: human

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzConstant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2328.1, Definition 2328.2, Definition 2328.3, and Definition 2328.4 seed the finite Lipschitz-constant certificate over displayed Real, Metric, dyadic comparison, LipschitzMap, modulus, uniform-continuity, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzConstant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim detached host constants, quotient map equality, differentiability, selected limits, countable choice, ambient metric completeness, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nonnegative-bound exposure, dyadic comparison exactness, modulus handoff, and ledger non-escape for BEDC.Derived.LipschitzConstantUp.

Constructive story: A $\mathsf{LipschitzConstant}^{\uparrow}$ packet is generated from one displayed RealUp constant row, source and target MetricUp rows, a DyadicRatCore comparison ledger, a LipschitzMap source row, a UniformModulus and UniformContinuity handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Chapter 2329

A Concrete Naming Certificate for $\mathsf{PoincareInequality}^{\uparrow}$

Following Poincare and Bishop–Bridges, the $\mathsf{PoincareInequality}^{\uparrow}$ packet names the finite carrier surface by which a bounded-domain or zero-mean Sobolev read yields a norm-control estimate. It is a downstream consumer of the weak-derivative route in chapter 440, the metric and real readback surfaces of chapter 57 and chapter 38, and the normed target ledger used by functional-analysis chapters.

Definition 2329.1 (Poincare inequality carrier)**.** *A* $\mathsf{PoincareInequality}^{\uparrow}$ *carrier is a finite packet*

$$Q = (D, B, G, N, K, H, C, P, L)$$

with the following displayed rows.

1. *D is the domain row, read through* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording the bounded or otherwise admissible observation surface on which the inequality is asserted.*
2. *B is the boundary or zero-mean condition row. It records the displayed trace, anchored-boundary, or mean-zero constraint that prevents constant functions from escaping the estimate.*
3. *G is the gradient or derivative-energy row supplied by* $\mathsf{Sobolev}^{\uparrow}$*, exposing the weak-gradient magnitude before any norm estimate is consumed.*
4. *N is the norm row, read through* $\mathsf{Norm}^{\uparrow}$ *and the scalar* $\mathsf{Real}^{\uparrow}$ *seal, naming the measured size controlled by the gradient route.*
5. *K is the constant ledger. It records a finite Poincare constant, its dependency on the displayed domain and boundary rows, and the fact that no hidden compactness theorem or selected minimizer is imported.*
6. *H is componentwise* hsame *transport for D, B, G, N, K.*
7. *C is the displayed* Cont *replay that sends a Sobolev admissibility read through the boundary row, derivative-energy row, norm row, and constant ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{PoincareInequality}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, completion-facing rows,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{PoincareInequality}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.PoincareInequalityUp.PoincareInequalityTasteGate single carrier alignment`

Theorem 2329.2 (Poincare inequality NameCert obligations)**.** *For an accepted carrier* $Q = (D, B, G, N, K, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{PoincareInequality}^{\uparrow}}$ *surface consists of five obligations.*

1. *Carrier admission displays the domain, boundary or zero-mean condition, Sobolev derivative-energy row, norm row, constant ledger, transports, replay, provenance, and local naming row.*
2. *Sobolev dependency requires every derivative-energy read to factor through the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative packet of chapter 440.*
3. *Norm and scalar readback require the measured side of the estimate to pass through* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before any completion-facing consumer can read the bound.*
4. *Constant-ledger exactness requires the Poincare constant to depend only on the displayed domain and boundary rows, not on a host compactness theorem, selected representative, quotient of functions, or hidden minimizer.*
5. *Transport and replay preserve the same finite route under componentwise* hsame *and displayed* Cont *paths, including the completion-facing rows used by downstream metric or normed-space consumers.*

Thus the packet exports a finite NameCert route for the inequality estimate, not a full Sobolev embedding theorem, compactness principle, quotient function space, or host analytic proof.

Proof. Project the carrier coordinates. The rows D, B, G, N, K expose exactly the domain, boundary or zero-mean condition, derivative-energy source, norm target, and constant ledger needed for the estimate.

The Sobolev dependency is read through G. Since G is a displayed $\mathsf{Sobolev}^{\uparrow}$ row, any weak-derivative or gradient-energy consumer must pass through the sibling certificate before the norm row is inspected.

The norm and scalar readback are then forced through N and the $\mathsf{Real}^{\uparrow}$ seal. The constant ledger K records only the visible dependency on D and B, so a hidden minimizer, compactness theorem, quotient representative, or selected function cannot enter the route.

Finally H, C, P, L repack the same finite path under transport, continuation replay, provenance, and local naming. The completion-facing rows stay downstream of the displayed domain, Sobolev, norm, and scalar evidence. □

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareInequality}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2329.1 and Definition 2329.2 seed the finite Poincare inequality route over $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, completion-facing rows, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareInequality}^{\uparrow}}, \mathsf{encodedDef})$

Lean target: `BEDC.Derived.PoincareInequalityUp`

Bridge status: none

Not claimed: This chapter does not close compact Sobolev embedding, trace theory, quotient equality of functions, selected minimizers, host compactness, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, Sobolev dependency, norm and scalar readback, constant-ledger exactness, componentwise transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{PoincareInequality}^{\uparrow}$ packet is generated from a displayed domain row, boundary or zero-mean row, Sobolev derivative-energy row, norm row, finite constant ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2330

A Concrete Naming Certificate for $\mathsf{TypedExampleReductionTrace}^{\uparrow}$

The $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ packet realizes the MetaCIC typed-example gallery as a finite NameCert carrier. It reads the audit map of chapter 9382 as a concrete route connecting a displayed typed term, its type, a selected reduction step, the checker result, closedness evidence, transport, replay, provenance, and local naming. The packet consumes the bounded MetaCIC critical-path and confluence-boundary surfaces through chapter 765 and chapter 2274 before the checker result row is read.

Before the checker and reduction rows are public, the packet also consumes the closed context seal of chapter 722. That seal supplies the finite context row read by Z, so the displayed type A, reduction step R, and checker result K are not interpreted over an open or implicit environment. The subject-reduction frontier is then reached through chapter 718 or chapter 716: those packets provide the boundary rows saying which beta-substitution and subject-reduction obligations must be visible before the typed-example trace may treat R and K as a coherent public row.

Definition 2330.1 (TypedExampleReductionTrace carrier)**.** *A* $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ *carrier is a finite packet*

$$T = (E, A, R, K, Z, H, C, P, L)$$

with the following displayed rows.

1. *E is the MetaCIC typed-example term row, naming one Church, identity, application, beta-redex, projection, Pi, constant, or sort-context example from the gallery.*
2. *A is the displayed type row accepted for E by the represented typing relation.*
3. *R is the reduction-step row, recording the selected beta or computation step when the gallery example has one.*
4. *K is the checker-result row, linking the displayed term and type to the option-valued checker or inference result used by the decidability gallery.*
5. *Z is the closedness row, exposing the finite context seal needed before the reduction or checker result is read.*
6. *H is componentwise* hsame *transport for E, A, R, K, Z.*
7. *C is the displayed* Cont *replay from typed-example admission through reduction, checker acceptance, closedness, and downstream audit use.*

8. *P is the* Pkg *provenance over* $\mathsf{TypedExampleReductionTrace}^{\uparrow}$*, the MetaCIC typed-example gallery, checker rows, reduction rows, closedness rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. *L is the local* $\mathsf{NameCert}_{\mathsf{TypedExampleReductionTrace}^{\uparrow}}$ *row.*

Theorem 2330.2 (TypedExampleReductionTrace NameCert obligations)**.** *For an accepted carrier $T = (E, A, R, K, Z, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{TypedExampleReductionTrace}^{\uparrow}}$ *surface consists of carrier admission, typed-term and type visibility, reduction-step visibility, checker-result agreement, closedness exposure, componentwise transport, continuation replay, provenance, and local naming. Every read factors through the finite route*

$$E \to A \to R \to K \to Z \to H, C, P, L.$$

The packet does not claim subject reduction, strong normalization, confluence, global conversion decidability, or an arbitrary closed-term theorem.

Proof. Carrier admission projects the rows of Definition 2330.1. The term row E and type row A expose the gallery example before any computation or checker result is consumed.

The reduction row R and checker row K are then read as displayed MetaCIC evidence. They record a selected worked-example computation and its checker-facing result; they do not range over all terms or all reductions.

Closedness is exposed by Z before transport and replay. This keeps the route tied to the finite context seal required by the gallery and prevents the carrier from asserting an unbounded subject-reduction principle.

Finally H, C, P, L preserve the same row-local route under hsame transport, continuation replay, provenance, and local naming. The public read remains a finite typed-example trace rather than a global metatheorem. □

Falsifiable prediction: Every accepted $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ read factors through a displayed MetaCIC typed-example term, type, reduction, checker-result, closedness, hsame, Cont, Pkg, and NameCert route.

Independence witness: metacic-subject-reduction-obstruction, betasubstitutionpreservation, subjectreductiondischarge, closedtermcontextseal: none is bijective with $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ because the packet jointly carries a typed-example term, type row, reduction row, checker-result row, closedness row, transport, replay, provenance, and local naming.

The Lean TasteGate work for this packet must provide explicit Nontrivial and FieldFaithful instances for $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ preserving $E, A, R, K, Z, H, C, P, L$.

Closure status for $\mathsf{NameCert}_{\mathsf{TypedExampleReductionTrace}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TypedExampleReductionTrace}^{\uparrow}}$, seed)

Scope closed: Definition 2330.1 and Definition 2330.2 seed the finite typed-example reduction-trace route over MetaCIC typed terms, displayed types, reduction rows, checker results, closedness, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TypedExampleReductionTrace}^{\uparrow}}$, encodedDef)

Lean target: `BEDC.Derived.TypedExampleReductionTraceUp`

Bridge status: none

Not claimed: This chapter does not close subject reduction, strong normalization, confluence, global conversion decidability, arbitrary closed-term normalization, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, typed-term visibility, type visibility, reduction-step visibility, checker-result agreement, closedness exposure, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{TypedExampleReductionTrace}^{\uparrow}$ packet is generated from a displayed MetaCIC typed-example term, its type row, a reduction-step row, a checker-result row, a closedness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2331

A Concrete Naming Certificate for $\mathsf{UniformEntourageBasis}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, a uniformity can be read through a directed family of entourages whose finite refinements control diagonal, symmetry, and composition requests. The $\mathsf{UniformEntourageBasis}^{\uparrow}$ packet records that basis surface as finite BEDC NameCert data. It sits between the ambient uniform-space interface of chapter 339, the separated boundary of chapter 1876, the complete-uniform consumer of chapter 1588, the Cauchy-filter route of chapter 333, and the metric basis model of chapter 57. The packet gives L10 completion consumers an explicit finite entourage-basis carrier rather than an implicit convention about available uniform neighborhoods.

Definition 2331.1 (Uniform entourage basis carrier)**.** *A* $\mathsf{UniformEntourageBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, B, D, R, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-source row, carrying the displayed source histories and point classifier read by the ambient* $\mathsf{UniformSpace}^{\uparrow}$ *packet.*
2. *B is the finite entourage-basis ledger, listing the named basic entourage rows available to completion, Cauchy-filter, separated-uniform, and metric consumers.*
3. *D is the diagonal ledger, recording that each listed basic entourage admits the displayed diagonal point pairs from X.*
4. *R is the pairwise refinement ledger, recording for each two displayed basic entourages a further listed basic entourage refined into both.*
5. *S is the symmetry ledger, recording for each listed basic entourage a displayed basic entourage whose reversed point-pair reads refine the original row.*
6. *Q is the composition ledger, recording for each listed basic entourage a displayed basic entourage whose two-step relational composition refines the original row.*
7. *H records componentwise* hsame *transport for the point source, basis ledger, diagonal rows, pairwise refinement rows, symmetry rows, and composition rows.*
8. *C records displayed* Cont *replay from any completion or Cauchy-filter consumer request through X, B, D, R, S, Q in that order.*

9. *P is the* Pkg *provenance over* $\mathsf{UniformEntourageBasis}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformEntourageBasis}^{\uparrow}}$ *row.*

The carrier exports no quotient completion, host equality of points, selected limit, countable-choice schedule, open-cover compactness principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness row, Lean verification, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.UniformEntourageBasisUp.UniformEntourageBasisTasteGate single carrier alignment`

Theorem 2331.2 (Uniform entourage basis NameCert obligations)**.** *For the carrier of Definition 2331.1, the local* $\mathsf{NameCert}_{\mathsf{UniformEntourageBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (X, B, D, R, S, Q, H, C, P, N)$*, point-source admission through* X*, finite basic-entourage admission through* B*, diagonal admission through* D*, pairwise common-refinement admission through* R*, symmetry admission through* S*, composition admission through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The point-source row and finite basis ledger are displayed before any uniform consumer can ask for a Cauchy, separated, complete, or metric-facing read.

The obligation rows are the uniformity laws restricted to the named basis. The diagonal row D, pairwise refinement row R, symmetry row S, and composition row Q certify exactly the finite requests needed to treat B as an entourage basis.

The structural rows preserve that basis surface. The transport row H moves all six data rows componentwise, the replay row C forces consumers through the same route, the package row P records the sibling dependencies, and the local name N certifies no additional completion or choice coordinate. □

Theorem 2331.3 (Uniform entourage basis refinement stability)**.** *Every accepted* $\mathsf{UniformEntourageBasis}^{\uparrow}$ *packet keeps completion-facing refinement reads inside the finite route*

$$X, B, D, R, S, Q, H, C, P, N.$$

A request for a smaller common entourage is discharged by the pairwise refinement row R*; a request for a reversed entourage is discharged by* S*; and a request for a two-step entourage whose composition remains inside a displayed target entourage is discharged by* Q*. The resulting read is transported by* H*, replayed by* C*, sourced by* P*, and named by* N*, with no quotient completion, host equality, selected limit, countable choice, open-cover compactness, or ambient Real completeness exposed.*

Proof. Start with a completion-facing or Cauchy-filter request. The consumer can inspect only the point source X and the finite basis ledger B before asking for a refinement row.

There are three refinement forms. Pairwise narrowing is the row R, reversed-point stability is the row S, and two-step composition stability is the row Q; the diagonal row D supplies the identity-pair side condition used by all three forms.

Finally apply the structural route. Componentwise hsame transport keeps the selected basis rows attached to the same point source, Cont replay prevents the consumer from skipping the displayed rows, and the Pkg and local NameCert rows name only this finite basis packet. □

Falsifiable prediction: Every L10 completion read that consumes a UniformEntourageBasisUp packet must factor through the displayed point source, finite basis ledger, diagonal rows, pairwise refinement rows, symmetry rows, composition rows, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: uniformspace, separateduniformspace, completeuniformspace, cauchyfilter, metric: UniformEntourageBasisUp is not bijective to any listed sibling because it jointly carries a point-source row, a finite entourage-basis ledger, diagonal rows, pairwise refinement rows, symmetry rows, composition rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEntourageBasis}^{\uparrow}}$ **.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEntourageBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2331.1, Definition 2331.2, and Definition 2331.3 seed the finite entourage-basis carrier over point-source rows, basic entourage ledgers, diagonal rows, pairwise refinement rows, symmetry rows, composition rows, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEntourageBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, host equality, selected limits, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-law stability, completion-consumer nonescape, and TasteGate field faithfulness for BEDC.Derived.UniformEntourageBasisUp.

Constructive story: A $\mathsf{UniformEntourageBasis}^{\uparrow}$ packet is generated from a point-source row, finite entourage-basis ledger, diagonal rows, pairwise refinement rows, symmetry rows, composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2332

A Concrete Naming Certificate for $\mathsf{UniformCompletionMonad}^{\uparrow}$

Following Bourbaki and Kelley on uniform completion, the $\mathsf{UniformCompletionMonad}^{\uparrow}$ packet records the completion unit and idempotent multiplication route as finite NameCert data over chapter 1473, chapter 426, chapter 2256, **??**, and chapter 1506. It names the monadic completion surface as a ledger of displayed rows; it does not identify quotient completions, select representatives, compare host equality, use choice, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2332.1 (Uniform completion monad carrier)**.** *A* $\mathsf{UniformCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (U, F, T, I, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row naming the separated completion route over the source uniform presentation.*
2. *F is the* $\mathsf{CompletionFunctor}^{\uparrow}$ *row that transports displayed uniformly continuous maps through the same completion route.*
3. *T is the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *row recording the unit-return triangle for source data entering the completion boundary.*
4. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *row recording that a repeated completion read contributes no additional public completion object.*
5. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row serving metric consumers that read the same completion-facing surface.*
6. *R is the finite route ledger that orders the unit leg before the functorial read and the idempotent multiplication leg after the repeated completion read.*
7. *H stores componentwise* hsame *transport for the uniform completion, functor, unit triangle, idempotent, metric completion, route ledger, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for unit admission, functorial transport, repeated completion, idempotent multiplication, metric readback, and terminal naming.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionMonad}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUnitTriangle}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionMonad}^{\uparrow}}$ *row.*

The carrier exposes the monad only as the displayed unit route and idempotent multiplication route through these rows. It carries no quotient completion object, no selected representative, no host equality of completed uniform spaces, no choice schedule, and no completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformCompletionMonadUp.UniformCompletionMonadTasteGate single carrier alignment`

Theorem 2332.2 (Uniform completion monad NameCert obligations)**.** *For an accepted* $\mathsf{UniformCompletionMonad}^{\uparrow}$ *carrier* $M = (U, F, T, I, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionMonad}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the uniform completion, completion functor, unit triangle, idempotent, metric completion, route ledger, transport, replay, provenance, and local naming rows;*
2. *unit-route visibility, requiring every source-to-completion read to pass through U, F, and T before any metric completion consumer reads it;*
3. *idempotent multiplication visibility, requiring every repeated completion read to pass through I after the repeated functorial read has been displayed;*
4. *classifier stability, requiring transported monad evidence to replay through H and C before it is exported;*
5. *ledger non-escape, requiring P and N to export no quotient completion, selected representative, host equality, choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 2332.1. The tuple lists exactly the uniform completion, functor, unit triangle, idempotent, metric completion, route ledger, transport, replay, provenance, and local naming rows.

Unit-route visibility is the first completion face of the tuple. A source datum enters through U, is transported through F, and returns through the triangle row T before the metric completion row Q can be read.

Idempotent multiplication visibility is the repeated-completion face. The repeated read is ordered by R and replayed by C so that I is the only multiplication row available to consumers.

Classifier stability is componentwise transport through H followed by the same replay through C. Ledger non-escape is the inventory statement for P and N: neither row contains a quotient completion, selected representative, host equality coordinate, choice schedule, or ambient completeness row. □

Theorem 2332.3 (Uniform completion monad unit route)**.** *Every accepted* $\mathsf{UniformCompletionMonad}^{\uparrow}$ *unit read factors through the displayed route*

$$\mathsf{UniformCompletion}^{\uparrow},\ \mathsf{CompletionFunctor}^{\uparrow},\ \mathsf{CompletionUnitTriangle}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow}.$$

The route is a finite $\mathsf{NameCert}$ *ledger over the carrier rows of Definition 2332.1; it is not a quotient-completion map, a selected representative, a host equality of completed objects, a choice principle, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Read the carrier of Definition 2332.1. The only rows available before a metric completion consumer reads a unit datum are U, F, T, Q together with the route, transport, replay, provenance, and naming rows.

Apply the unit-route visibility clause of Definition 2332.2. It orders the source-to-completion read through U, F, and T before exposing Q.

The package and local naming rows contain no further completion object or host equality coordinate. Therefore the exported unit route is exhausted by the displayed finite ledger and cannot supply quotient completions, selected representatives, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2332.4 (Uniform completion monad idempotent multiplication)**.** *Every accepted* UniformCompletionMon *repeated completion read factors through the displayed route*

$$\mathsf{UniformCompletion}^{\uparrow},\ \mathsf{CompletionFunctor}^{\uparrow},\ \mathsf{CauchyCompletionIdempotent}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow}.$$

The multiplication face records idempotence as reuse of the same finite completion ledger, not as equality of host completion objects.

Proof. Read the carrier rows U, F, I, Q, R, C from Definition 2332.1. They are the only rows that can serve a repeated completion consumer.

Apply the idempotent multiplication visibility clause of Definition 2332.2. The route ledger R orders the repeated functorial read, and the replay row C forces the multiplication face through I before Q is exposed.

Since P and N contain no quotient completion or host equality coordinate, idempotence is available only as the finite reuse of the displayed completion ledger. No selected representative, choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is exported. □

Theorem 2332.5 (Uniform completion monad ledger non-escape)**.** *Every consumer route exported by an accepted* $\mathsf{UniformCompletionMonad}^{\uparrow}$ *carrier is exhausted by the unit route of Definition 2332.3, the idempotent multiplication route of Definition 2332.4, the classifier transport row H, the replay row C, the provenance row P, and the local naming row N. No exported route can obtain quotient completions, selected representatives, host equality, choice,* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking from the packet.*

Proof. The carrier has only the rows displayed in Definition 2332.1. The unit route and the idempotent multiplication route account for the two completion-facing monad reads.

Apply the classifier stability and ledger non-escape clauses of Definition 2332.2. Transported evidence must pass through H and C, while P and N only record provenance and local naming.

Thus every consumer route is already one of the displayed finite routes. The packet contains no row from which quotient completions, selected representatives, host equality, choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionMonad}^{\uparrow}$ packet is generated from a uniform completion row, completion functor row, completion unit triangle row, completion idempotent row, metric completion row, finite route ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2332.1, Definition 2332.2, Definition 2332.3, Definition 2332.4, and Definition 2332.5 seed the finite monadic completion packet over unit, functorial, idempotent, metric, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionMonad}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.UniformCompletionMonadUp`

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host equality, choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-route visibility, idempotent multiplication visibility, classifier stability, and ledger non-escape for BEDC.Derived.UniformCompletionMonadUp.

Chapter 2333

A Concrete Naming Certificate for $\mathsf{DirichletSeriesTest}^{\uparrow}$

Following Dirichlet and Bishop–Bridges constructive analysis, the $\mathsf{DirichletSeriesTest}^{\uparrow}$ packet records the finite cancellation surface for a real series whose partial sums are bounded and whose coefficient stream is monotone null. It consumes the $\mathsf{RealSeries}^{\uparrow}$ root surface of chapter 1581, the uniform Cauchy handoff of chapter 1736, the real sequence limit interface of chapter 1413, the regular rational rows of chapter 301, the stream windows of chapter 126, the dyadic tolerance core of chapter 291, and the endpoint seal of chapter 38. The packet names convergence only through finite Abel-summation windows and shared Cauchy thresholds; it does not import host infinite sums, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2333.1 (Dirichlet series test carrier)**.** *A* $\mathsf{DirichletSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, U, L, Q, W, D, E, A, B, V, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealSeries}^{\uparrow}$ *carrier of Definition 1581.1, including finite term access, partial-sum windows, the shared threshold, and the endpoint seal route.*
2. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff row consumed only after the Dirichlet cancellation surface is displayed.*
3. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *limit-interface row for the coefficient stream, read only as a null ledger and not as a selected series limit.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coefficient comparisons.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for term, coefficient, and Abel-difference indices.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal inherited from the real-series row.*
8. *A is the bounded partial-sum ledger.*
9. *B is the monotone null coefficient ledger.*
10. *V is the Abel-summation finite-difference variation ledger.*
11. *H, C, P, N are componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{DirichletSeriesTest}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.DirichletSeriesTestUp.DirichletSeriesTestTasteGate single carrier alignment`

Theorem 2333.2 (Dirichlet series test NameCert obligations)**.** *For every accepted* $\mathsf{DirichletSeriesTest}^{\uparrow}$ *carrier* $K = (R, U, L, Q, W, D, E, A, B, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirichletSeriesTest}^{\uparrow}}$ *surface consists exactly of five obligations: carrier admission for the displayed rows, bounded partial-sum ledger visibility through* A*, monotone null coefficient ledger visibility through* B *and* L*, Abel summation route visibility through* V*, and tail-cancellation Cauchy readiness through* D, E, U, H, C, P, N*.*

Proof. Project the carrier of Definition 2333.1. Each listed obligation names one displayed row group of the finite packet and no external analytic object.

Use the $\mathsf{RealSeries}^{\uparrow}$ Dirichlet cancellation surface of Definition 1581.44. It supplies the bounded partial-sum, monotone null coefficient, Abel-difference, dyadic tolerance, shared threshold, and endpoint rows already required by the carrier.

The support rows H, C, P, N preserve, replay, source, and name the same finite route. Since the packet has no host infinite sum, selected limit, quotient equality, countable-choice schedule, or ambient completeness coordinate, the five obligations exhaust the accepted surface. □

Theorem 2333.3 (Dirichlet series test Abel summation route)**.** *Every accepted* $\mathsf{DirichletSeriesTest}^{\uparrow}$ *carrier exports its cancellation claim through the finite Abel-summation route*

$$R \longmapsto A \longmapsto (B, L, Q, W) \longmapsto V \longmapsto D \longmapsto E \longmapsto U.$$

The route first reads bounded partial sums from the $\mathsf{RealSeries}^{\uparrow}$ *row, then reads monotone null coefficients through* $\mathsf{RealSequenceLimit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{StreamName}^{\uparrow}$*, then reads Abel finite differences through* V*, and only then reaches dyadic tolerance, endpoint sealing, and the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *handoff.*

Proof. Start with the bounded partial-sum and variation route of Definition 1581.45. It places the finite partial-sum ledger before coefficient variation and dyadic tolerance.

Apply the tail-cancellation theorem Definition 1581.46. The endpoint seal is downstream of the bounded partial-sum ledger, coefficient ledger, Abel-difference ledger, dyadic tolerance ledger, and shared threshold.

Finally use the Cauchy-readiness theorem Definition 1581.47. The $\mathsf{UniformCauchySeries}^{\uparrow}$ row is a consumer of the same finite cancellation surface, so the handoff cannot bypass the displayed Abel route. □

Theorem 2333.4 (Dirichlet series test Cauchy consumer surface)**.** *The downstream Cauchy consumer of an accepted* $\mathsf{DirichletSeriesTest}^{\uparrow}$ *carrier receives only*

$\mathsf{RealSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist,

Its convergence read is the tail-cancellation Cauchy readiness of Definition 1581.47 and the Abel route of Definition 2333.3; no stronger host-side convergence principle is exported.

Proof. Read the carrier inventory in Definition 2333.1. The visible rows are exactly the real-series carrier, uniform Cauchy handoff, real sequence limit interface, regular rational readback, stream windows, dyadic tolerance, endpoint seal, cancellation ledgers, and support rows.

Use Definition 2333.2. It partitions those rows into carrier admission, bounded partial sums, monotone null coefficients, Abel summation, and Cauchy readiness.

The named consumer route then follows from Definition 2333.3. Because every step is a displayed finite row, the consumer cannot recover a host infinite sum, selected series limit, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{DirichletSeriesTest}^{\uparrow}$ convergence read factors through a RealSeries bounded partial-sum ledger, a monotone null coefficient ledger, Abel finite differences, dyadic tolerance, a shared Cauchy threshold, an endpoint RealUp seal, uniform Cauchy handoff, transport, continuation, provenance, and local naming.

Independence witness: realseries, uniform-cauchy-series, real-sequence-limit, regseqrat, streamname, dyadicratcore, real: $\mathsf{DirichletSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded partial sums, monotone null coefficients, Abel finite differences, Cauchy readiness, endpoint sealing, uniform handoff, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletSeriesTest}^{\uparrow}}$.

Constructive story: A $\mathsf{DirichletSeriesTest}^{\uparrow}$ packet is generated from a $\mathsf{RealSeries}^{\uparrow}$ carrier, $\mathsf{UniformCauchySeries}^{\uparrow}$ handoff, $\mathsf{RealSequenceLimit}^{\uparrow}$ coefficient limit interface, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{Real}^{\uparrow}$ endpoint seal, bounded partial-sum ledger, monotone null coefficient ledger, Abel finite-difference variation ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2333.1, Definition 2333.2, Definition 2333.3, and Definition 2333.4 seed the constructive Dirichlet series test surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, selected limits, quotient equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded partial-sum ledger visibility, monotone null coefficient visibility, Abel finite-difference accounting, Cauchy readiness, and TasteGate field faithfulness for BEDC.Derived.DirichletSeriesTestUp.

The $\mathsf{QuasiUniformSpace}^{\uparrow}$ packet routes its directed entourage surface through the UniformSpace sibling of chapter 339 and the UniformCompletion dependency.

Chapter 2334

A Concrete Naming Certificate for $\mathsf{QuasiUniformSpace}^{\uparrow}$

Following Nachbin and Kelley, the $\mathsf{QuasiUniformSpace}^{\uparrow}$ packet records directed uniform entourage data as finite NameCert evidence before any symmetric uniformity or quotient completion is read. It sits on the L10 Real and Completion route beside chapter 339, chapter 333, chapter 589, chapter 1473, and chapter 1876. The packet exposes the asymmetric substrate used by completion-facing filters and nets without asserting inverse entourages, separated quotient completion, selected limits, open-cover compactness, `Classical.choice`, or `Quot.sound`.

At the spine level, the reflected row E is a downstream reader of chapter 339. Completion-facing consumers route onward through chapter 1473, so the directed quasi-uniform carrier exposes the sibling dependency before any reflected uniform completion data is consumed.

The actual carrier relationship is asymmetric: U and D are the primitive directed entourage and direction-ledger rows, while the $\mathsf{UniformSpace}^{\uparrow}$ sibling of chapter 339 and the metric-facing route through chapter 57 can be read only through the reflected row E after those directed rows are visible. Thus a metric specialization consumes a displayed symmetric reflection of quasi-uniform data; it is not a replacement for the quasi-uniform carrier and it does not add inverse entourages to U.

Definition 2334.1 (QuasiUniformSpace carrier)**.** *A* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, D, F, N, E, H, C, P, L)$$

with the following displayed rows:

1. *U is the directed entourage row, recording finite diagonal, refinement, and forward-composition observations.*
2. *D is the direction ledger, recording which requests are read forward and which are transported only after a displayed reverse request exists.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *read row, exposing finite filter-base requests whose Cauchy comparisons are evaluated through U and D.*
4. *N is the* $\mathsf{CauchyNet}^{\uparrow}$ *read row, exposing directed net tails and common refinements through the same forward-entourage ledger.*
5. *E is the uniform reflection row, naming the optional* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *consumer only as a downstream reflection of the directed data.*

6. *H records componentwise* hsame *transport for the directed entourage, direction ledger, Cauchy-filter read, Cauchy-net read, uniform reflection, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from directed entourage observations through filter or net reads and then to any reflected uniform consumer.*
8. *P is the* Pkg *provenance over* $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{QuasiUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for symmetric uniformity as primitive data, a separated quotient completion, a selected limit, open-cover compactness, countable choice, `Classical.choice`*, or* `Quot.sound`*.*

Theorem 2334.2 (Quasi-uniform entourage obligations)**.** *For an accepted* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier*

$$Q = (U, D, F, N, E, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{QuasiUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, directed diagonal exposure, forward-refinement stability, forward-composition replay, direction-ledger compatibility, Cauchy-filter and Cauchy-net read discipline, optional uniform reflection, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $U, D, F, N, E, H, C, P, L$.*

Proof. Carrier admission is projection from Definition 2334.1. The accepted packet exposes exactly the directed entourage row, direction ledger, Cauchy-filter row, Cauchy-net row, uniform reflection row, transport row, replay row, provenance row, and local naming row.

Directed diagonal exposure, forward-refinement stability, and forward-composition replay are read from U. The row records the finite entourage observations used by the packet, so a consumer cannot replace them with a symmetric entourage filter.

The direction discipline is read from D. A forward request may be replayed as stored, while a reverse request is accepted only when the ledger displays it as a separate finite observation.

The Cauchy boundary rows are then projected from F and N. Filter-base and net-tail reads are evaluated through the displayed U, D route, and therefore cannot bypass the quasi-uniform substrate.

Finally E, H, C, P, L add only reflection, transport, replay, provenance, and naming. They do not introduce inverse entourages, separated quotient completion, selected limits, open-cover compactness, choice, `Classical.choice`, or `Quot.sound`. □

Theorem 2334.3 (Cauchy boundary for directed entourages)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *consumer read of Q factors through the displayed route*

$$U, D \longmapsto F, N \longmapsto H, C, P, L.$$

Thus the Cauchy boundary sees only directed entourage observations, their direction ledger, the displayed filter or net read row, and the structural rows. It does not see a quotient of Cauchy filters, a selected convergent subnet, a symmetric uniformity, or a host completion object.

Proof. Fix a Cauchy-facing consumer of Q. Projection gives U and D, so the consumer first receives the finite directed entourage observations and the ledger explaining which directions have been displayed.

If the consumer is filter-based, it reads F. The finite filter-base requests and Cauchy comparisons in F are accepted only as requests over U, D.

If the consumer is net-based, it reads N. The finite tail and common-refinement observations in N use the same directed ledger, so no selected subnet or limit representative is exported.

Transport, replay, provenance, and local naming close the route through H, C, P, L. These structural rows preserve the displayed data and provide no coordinate for quotient filter equality, selected convergence, symmetric uniformity, or a completed host object. □

Theorem 2334.4 (Uniform reflection route for quasi-uniform data)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Any* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *consumer named by E or P reads only a reflected route downstream of the directed data:*

$$U, D \longmapsto F, N \longmapsto E \longmapsto H, C, P, L.$$

The reflection route can record a symmetric or separated consumer as a reader, but it does not promote symmetric uniformity, quotient completion, or separatedness to primitive coordinates of the $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier.*

Proof. Project the optional reflection row E. Its uniform-space, uniform-completion, and separated-uniform-space entries are downstream consumers of the directed route, because the carrier presents U, D, F, N before E.

The displayed Cont row C replays the same order. A reflected consumer can inspect the directed entourage observations, then the Cauchy-filter or Cauchy-net boundary rows, and only then the reflected uniform request.

The provenance row P names the same dependency order. It may cite $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, or $\mathsf{SeparatedUniformSpace}^{\uparrow}$ as consumers, but the local naming row L still names the packet as $\mathsf{QuasiUniformSpace}^{\uparrow}$ with directed entourage data as its primitive surface.

Componentwise hsame transport in H preserves these rows without adding an inverse-entourage axiom. Hence symmetric uniformity, separated quotient completion, and selected limits remain outside the carrier. □

Theorem 2334.5 (Uniform completion sibling route for quasi-uniform data)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of Q must first read the directed entourage row U, the direction ledger D, the* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *rows F, N, then the reflection row E and the structural rows H, C, P, L. Equivalently, the completion-facing route factors through*

$$U, D \longmapsto F, N \longmapsto E \longmapsto H, C, P, L$$

before it can use the uniform-completion surface of chapter 1473. The Cauchy-filter and Cauchy-net siblings of chapter 333 and chapter 589 are therefore mandatory readers, while uniform completion remains a reflected consumer rather than primitive data in the quasi-uniform carrier.

Proof. Start with Definition 2334.3. It forces every Cauchy-facing read to inspect U, D before it reaches the filter row F or the net row N.

Apply Definition 2334.4. A uniform-completion consumer can be named only by the reflected row E or by provenance in P, and both are downstream of the directed Cauchy boundary.

The structural rows H, C, P, L then transport, replay, source, and name the same route. Since $\mathsf{UniformCompletion}^{\uparrow}$ is reached through this reflected path, no carrier coordinate promotes quotient completion, separatedness, selected limits, or symmetric uniformity to primitive quasi-uniform data. □

Theorem 2334.6 (Metric reflection lattice route for quasi-uniform data)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Whenever Q is reflected through a symmetric or separated row and then read by a* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer, the route must first expose*

$$U, D, F, N, E, H, C, P, L$$

before any metric-space or metric-completion surface is available. Thus the metric-facing lattice route factors through the Cauchy boundary and uniform reflection rows of Q before it reaches the metric certificate of chapter 57 or the metric-completion sibling of chapter 1506.

Proof. Start with Definition 2334.3. It forces every completion-facing read to open the directed entourage row U, the direction ledger D, and the Cauchy filter and net rows F, N before a reflected consumer can inspect completion data.

Next apply Definition 2334.4. A symmetric or separated row is present only as a reflected reader in E or as provenance in P, so metric structure is downstream of the quasi-uniform route rather than a primitive coordinate of the carrier.

Finally use the sibling route of Definition 2334.5. A metric-completion consumer may read the reflected completion surface only after H, C, P, L transport, replay, source, and name the displayed rows. Since none of those rows stores quotient completion equality, a selected limit, or a host metric completion object, those alternate routes remain unavailable. □

The completion lattice therefore has a mandatory sibling passage: any quasi-uniform completion consumer must first enter the uniform-completion surface of chapter 1473, and only after that reflected uniform row is displayed can metric reflection reach the metric-completion consumer of chapter 1506. This dependency prevents a directed entourage packet from being read directly as metric completion data.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{QuasiUniformSpace}^{\uparrow}$ packet is generated from directed entourage observations, a direction ledger, CauchyFilter and CauchyNet read rows, optional uniform reflection, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2334.1, Definition 2334.2, Definition 2334.3, Definition 2334.4, Definition 2334.5, and Definition 2334.6 seed the directed uniform completion substrate over finite quasi-uniform data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiUniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close symmetric uniformity, quotient completion, selected limits, open-cover compactness, choice principles, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed entourage exposure, direction-ledger compatibility, Cauchy-filter and Cauchy-net read discipline, uniform reflection non-escape, transport stability, replay stability, and ledger non-escape for BEDC.Derived.QuasiUniformSpaceUp.

The quasi-uniform completion route also has a distance-presented sibling edge. A consumer that asks for a directed ball, rather than a primitive entourage, must first read the $\mathsf{QuasiMetric}^{\uparrow}$ carrier of chapter 2158; only after the directed distance, zero diagonal, forward triangle, directed-ball, and Cauchy rows have been displayed may the consumer reflect to $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, or $\mathsf{Met}^{\uparrow}$.

Theorem 2334.7 (QuasiMetric forward link for quasi-uniform consumers)**.** *Every* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *completion-facing consumer that asks for a distance-presented directed ball factors through the* $\mathsf{QuasiMetric}^{\uparrow}$ *carrier of chapter 2158 before reflecting to* $\mathsf{UniformSpace}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$*. The reflected quasi-uniform route may consume directed-ball and Cauchy rows, but it does not turn those rows into primitive symmetry, quotient completion, selected limits, or host metric equality.*

Proof. Apply Definition 2158.3. The distance-presented directed-ball request first reads the $\mathsf{QuasiMetric}^{\uparrow}$ source, point, directed distance, zero diagonal, forward triangle, directed-ball, Cauchy filter, Cauchy net, uniform-reflection, transport, replay, provenance, and local naming rows.

Only after that route is displayed can the quasi-uniform carrier of Definition 2334.1 receive a directed entourage or completion-facing request. The reflected $\mathsf{UniformSpace}^{\uparrow}$ or $\mathsf{Met}^{\uparrow}$ consumer is therefore downstream of the quasi-metric route, not a replacement for it.

The non-escape clause of Definition 2158.4 excludes primitive symmetry, quotient completion, selected limits, and host equality. The quasi-uniform packet adds no coordinate that reintroduces them. □

Theorem 2334.8 (Uniform-space sibling route for quasi-uniform data)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier whose forward and reverse entourage ledgers agree as displayed finite rows. Then every symmetric quasi-uniform packet routes to the* $\mathsf{UniformSpace}^{\uparrow}$ *sibling surface of chapter 339 through the same finite entourage,* hsame*-transport,* Cont*-replay,* Pkg*, and* NameCert *rows.*

The route is local to the displayed carrier:

$$U, D \longmapsto E \longmapsto H, C, P, L.$$

It does not claim completion, Hausdorff separation, host topological equality, selected limits, or quotient uniformity.

Proof. Project the carrier of Definition 2334.1. The directed entourage row U and direction ledger D are displayed before the reflected row E.

The agreement assumption says that the forward and reverse entourage ledgers are the same finite data when read as a symmetric packet. Thus the reflected row E can expose the $\mathsf{UniformSpace}^{\uparrow}$ sibling surface without adding primitive symmetric uniformity to the quasi-uniform carrier.

The structural rows H, C, P, L transport, replay, source, and name the same finite route. Since no further coordinate is present, the theorem records only the local sibling passage and leaves completion, Hausdorff separation, host topological equality, selected limits, and quotient uniformity outside the packet. □

Theorem 2334.9 (Directed uniform-space dependency for quasi-uniform packets)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Any route from* Q *to the* $\mathsf{UniformSpace}^{\uparrow}$ *sibling of chapter 339 must first display the directed entourage row* U *and the direction ledger* D*. Only after those rows are visible may the reflected row* E *name a symmetric uniform-space reader; the quasi-uniform packet itself still exports directed entourage data rather than primitive symmetric uniformity, quotient completion, selected limits, or host equality.*

Proof. Project the carrier of Definition 2334.1. The directed entourage row U and direction ledger D precede every Cauchy-facing, reflected-uniform, transport, replay, provenance, or naming row.

Apply the uniform reflection route of Definition 2334.4. The reflected row E names $\mathsf{UniformSpace}^{\uparrow}$ only as a downstream reader of the displayed directed data.

Finally use the uniform-space sibling route of Definition 2334.8. It identifies the sibling passage through U, D, E, H, C, P, L, so the dependency edge to chapter 339 is real, but it does not promote symmetric uniformity or completion data to primitive quasi-uniform coordinates. □

Theorem 2334.10 (QuasiUniformSpace entourage-direction sibling route)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. A directed-entourage read of* Q *sits between the distance rows of chapter 57 and chapter 2158, the symmetric-entourage rows of chapter 339, the convergence routes of chapter 333 and chapter 589, and the completion consumers of chapter 1473. The directed entourage classifier is not a symmetric metric, selected limit, quotient completion, or open-cover compactness theorem.*

Proof. Project the carrier of Definition 2334.1. The rows U and D expose directed entourage data and the direction ledger before the filter, net, reflection, transport, replay, provenance, and naming rows can be consumed.

Use Definition 2334.7. Distance-presented directed balls must first pass through the $\mathsf{QuasiMetric}^{\uparrow}$ route, while metric-facing consumers remain reflected readers rather than primitive quasi-uniform coordinates.

Apply Definition 2334.8 and Definition 2334.9. Symmetric $\mathsf{UniformSpace}^{\uparrow}$ data are downstream of the displayed directed rows and cannot replace them.

Finally use Definition 2334.3 and Definition 2334.12. The $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$ routes read the same directed boundary, so completion-facing consumers receive no selected limit, quotient completion, symmetric metric, or open-cover compactness coordinate. □

Theorem 2334.11 (Quasi-uniform forward uniform-space link)**.** *Let*

$$Q = (U, D, F, N, E, H, C, P, L)$$

be an accepted $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier whose forward and reverse entourage requests are both displayed in the directed ledger* D*. Then every symmetric uniform consumer of* Q *factors through the* $\mathsf{UniformSpace}^{\uparrow}$ *sibling of chapter 339 before the completion-facing source row consumed by* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *in chapter 1588 is available.*

The route is

$$U, D \longmapsto E \longmapsto H, C, P, L,$$

where E *reads the reflected* $\mathsf{UniformSpace}^{\uparrow}$ *row only after both directions have been displayed. The resulting completion source is therefore a finite reflected uniform row, not primitive symmetric data inside the quasi-uniform carrier and not a selected quotient completion.*

Proof. Project the carrier of Definition 2334.1. The directed entourage row U and the ledger D are read before the reflected row E, so a symmetric consumer must see both the forward and reverse displayed requests before it can cite a uniform-space row.

Use the uniform reflection route of Definition 2334.4. Once both directions are present, E supplies the reflected $\mathsf{UniformSpace}^{\uparrow}$ sibling surface of chapter 339, while H, C, P, L only transport, replay, source, and name that same reflected route.

Finally read the complete-uniform-space carrier of chapter 1588. Its completion source row consumes a displayed $\mathsf{UniformSpace}^{\uparrow}$ entourage surface, so the quasi-uniform packet reaches it only through the reflected uniform row and not through primitive symmetric uniformity, quotient completion, or a selected limit. □

Theorem 2334.12 (Quasi-uniform uniform-completion dependency route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of a* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *packet first reads the directed entourage row, the direction ledger, the* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *boundary rows, and the reflected* $\mathsf{UniformSpace}^{\uparrow}$ *row before it reaches the completion surface of chapter 1473. The route is mediated by the Cauchy-filter and Cauchy-net siblings of chapter 333 and chapter 589, so uniform completion is a downstream reader of directed quasi-uniform data rather than a primitive coordinate of the carrier.*

Proof. Project the carrier of Definition 2334.1. The directed entourage row and direction ledger are exposed before any filter, net, or reflected uniform consumer is visible.

The $\mathsf{CauchyFilter}^{\uparrow}$ and $\mathsf{CauchyNet}^{\uparrow}$ rows then bind completion-facing reads to the displayed directed boundary. The reflected $\mathsf{UniformSpace}^{\uparrow}$ row can be read only after those boundary rows have been displayed.

The reflected symmetric entourage row is precisely the sibling route of chapter 339. A completion consumer therefore cannot bypass that uniform-space witness when it reads the $\mathsf{UniformCompletion}^{\uparrow}$ surface.

Thus every uniform-completion request reaches chapter 1473 through the same directed entourage, direction-ledger, Cauchy-filter, Cauchy-net, and reflected-uniform route, and no quotient completion or selected limit is promoted to primitive quasi-uniform data. □

The quasi-uniform reflection lattice keeps completion-facing consumers downstream of directed entourage data. A $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{SeparatedUniformSpace}^{\uparrow}$ reader may appear in the reflected row of a $\mathsf{QuasiUniformSpace}^{\uparrow}$ packet, but it first reads the directed entourage rows before it can consume the uniform completion surface of chapter 1473 or the separated sibling surface.

Theorem 2334.13 (Uniform completion reflection lattice for quasi-uniform data)**.** *Let* $Q = (U, D, F, N, E, H, C, P, L)$ *be an accepted* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *carrier. Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *consumer named by* E *or* P *first reads the directed entourage row* U *and direction ledger* D*, then the reflection row* E*, before any completion-facing or separated-uniform surface is available. Thus the reflected route through chapter 339 and chapter 1473 remains downstream of finite quasi-uniform data.*

Proof. Project the carrier row. The directed entourage row U and direction ledger D are displayed before the reflection row E, so a reflected consumer cannot begin with symmetric or separated completion data.

Use the uniform reflection route of Definition 2334.4. It names $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, and $\mathsf{SeparatedUniformSpace}^{\uparrow}$ only as downstream readers of the directed rows.

The structural rows H, C, P, L transport, replay, source, and name the same order. Hence a completion-facing or separated-uniform consumer receives a reflected route rather than a primitive

quotient completion, selected limit, or separatedness coordinate inside the quasi-uniform carrier. □

Chapter 2335

A Concrete Naming Certificate for $\mathsf{VolterraIntegralOperator}^{\uparrow}$

Following Volterra and the standard constructive treatment of integral equations, the $\mathsf{VolterraIntegralOperator}^{\uparrow}$ packet records the finite operator surface by which a displayed kernel and an input history produce a causal integral history before any completion-facing consumer may read a fixed-point or convergence route. It sits over the public integral row of chapter 123, the continuous-map graph row of chapter 130, the endpoint seal of chapter 38, the compact source-window discipline of chapter 129, and the bounded-operator boundary of chapter 1644. The packet names the constructive Volterra operator boundary; it does not import a Fredholm inverse, global compactness theorem, selected solution operator, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2335.1 (Volterra integral operator carrier)**.** *A* $\mathsf{VolterraIntegralOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, T, X, Y, G, M, Q, S, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite interval-window row naming the inspected initial segment of the source history.*
2. *T is the time-order row, recording only pairs of displayed times with the triangular relation $s \leq t$ inside I.*
3. *X is the input-history row read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{UniformModulus}^{\uparrow}$ *windows.*
4. *Y is the output-history row whose values are exposed only as* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
5. *G is the causal kernel-window row, visible as a* $\mathsf{ContinuousMap}^{\uparrow}$ *graph only on the triangular domain determined by T.*
6. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row for the input history, kernel graph, and finite Riemann-window estimates used by the operator.*
7. *Q is the quadrature-window row, storing finite tagged subintervals and dyadic error budgets for the displayed integral reads.*
8. *S is the* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{RealSeries}^{\uparrow}$ *finite-sum row that records the partial-sum approximation of the integral window.*
9. *R is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row for completion-facing consumers of the output history.*

10. *L is the successive-approximation ledger, storing only the finite Picard-window route $X \mapsto Y$ through G, M, Q, S, R.*
11. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal for each displayed output value.*
12. *H records componentwise* hsame *transport for the interval, time-order, input, output, kernel, modulus, quadrature, series, completion, ledger, endpoint, replay, provenance, and local naming rows.*
13. *C records displayed* Cont *replay from an accepted operator request through the same rows.*
14. *P is the* Pkg *provenance over* $\mathsf{VolterraIntegralOperator}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite Volterra operator packet. It has no coordinate for a Fredholm inverse, global compactness theorem, selected solution operator, host function extensionality, quotient equality of functions, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Theorem 2335.2 (Volterra integral operator NameCert obligations)**.** *For every accepted* $\mathsf{VolterraIntegralOperator}^{\uparrow}$ *carrier $K = (I, T, X, Y, G, M, Q, S, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval window, triangular time-order row, input-history row, output-history row, kernel graph row, uniform-modulus row, quadrature-window row, RealSeries finite-sum row, UniformCompletion handoff row, successive-approximation ledger, Real endpoint seal, componentwise transport, displayed continuation replay, package provenance, and local naming. Every accepted operator request is classified only by the displayed finite rows.*

Lean4 ✓ `BEDC.Derived.VolterraIntegralOperatorUp.VolterraIntegralOperatorCarrier namecert obligations`

Proof. Project the accepted carrier. The analytic rows are $I, T, X, Y, G, M, Q, S, R, L, E$, and the structural support rows are H, C, P, N.

Carrier admission reads those rows in the displayed order. The input and kernel rows are continuous and uniformly controlled only through X, G, M, the integral window is approximated only through Q, S, and the completion-facing handoff is the row R before the endpoint seal E is read.

Classifier stability is componentwise. The hsame row transports each finite coordinate, while C replays the same operator route for later consumers and P, N source and name it.

Since the carrier has no field for a host function space, a quotient function, a selected fixed point, a selected limit, countable choice, ambient real completeness, or a standard bridge, none of those principles can be exported by the local naming certificate. □

Theorem 2335.3 (Volterra triangular causality)**.** *For every accepted* $\mathsf{VolterraIntegralOperator}^{\uparrow}$ *carrier, the output value at a displayed time t is read only through source times s with $s \leq t$ recorded in the triangular row T. In BEDC rows, every causal read factors through*

$$I, T \longmapsto X, G, M \longmapsto Q, S \longmapsto Y, E \longmapsto H, C, P, N.$$

Thus a downstream consumer cannot inspect future input-history rows, replace the triangular domain by a host function-space domain, or identify two histories by external function extensionality.

Proof. Start with a displayed time request. The interval row I fixes the finite time window, and the triangular row T permits only source indices already recorded with $s \leq t$.

The kernel and input rows are then read through X, G, M. The uniform-modulus row controls the finite estimates over the same triangular domain, so no future time coordinate is introduced by continuity data.

The integral read is discharged by the quadrature and RealSeries rows Q, S, followed by the output and endpoint rows Y, E. Transport, replay, provenance, and naming preserve this same finite route.

The excluded host function-space and extensionality principles are absent from every carrier coordinate, so the displayed triangular route is the entire causality surface. □

Theorem 2335.4 (Volterra integral operator causal nonescape)**.** *Every accepted* $\mathsf{VolterraIntegralOperator}^{\uparrow}$ *packet must expose its source interval row, causal kernel-window row,* $\mathsf{ContinuousMap}^{\uparrow}$ *row,* $\mathsf{Integral}^{\uparrow}$ *row, terminal* $\mathsf{Real}^{\uparrow}$ *output seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}$ *row before any consumer reads the output history. The accepted read is the finite route*

$$I, T, G, X, S, E, H, C, P, N,$$

with compact source discipline supplied only through $\mathsf{CompactMetric}^{\uparrow}$ *rows and bounded operator discipline supplied only through* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows. It exports no Fredholm inverse, global compactness theorem, selected solution operator, quotient function equality, choice-selected representative, or ambient solution-space object.*

Proof. Start with carrier admission. The source interval and triangular time-order rows I, T determine the finite causal window, and the kernel row G is readable only on that window.

Next read the analytic rows through the displayed dependencies. The input and kernel data reach consumers through $\mathsf{ContinuousMap}^{\uparrow}$ rows, the integral window is exposed through the $\mathsf{Integral}^{\uparrow}$ and finite-sum row S, and the output is sealed only at the terminal $\mathsf{Real}^{\uparrow}$ row E.

The structural rows then preserve the same packet. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row attach the causal read to H, C, P, N before any consumer can use the output.

The compact and bounded-linear-operator references are dependency boundaries, not hidden theorems. Since the packet has no coordinate for a Fredholm inverse, global compactness theorem, selected solution operator, quotient function equality, selected representative, or ambient solution space, none of those objects can escape through the Volterra output. □

Theorem 2335.5 (Volterra successive approximation surface)**.** *The successive-approximation ledger of an accepted* $\mathsf{VolterraIntegralOperator}^{\uparrow}$ *carrier exposes only finite Picard windows. Each displayed Picard step reads the previous history through* X*, the kernel through* G*, the shared modulus through* M*, the quadrature and finite RealSeries rows through* Q, S*, the completion-facing handoff through* R*, and the endpoint seal through* E*. The exported surface is therefore*

$$X, G, M, Q, S, R, E, L, H, C, P, N,$$

not a Banach fixed-point theorem, a selected limit of iterates, a quotient equality of histories, countable choice over iteration indices, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the Picard ledger L. By definition it stores only displayed approximation windows, each tied to the same input, kernel, modulus, quadrature, finite-sum, completion, and endpoint rows.

For one step, the previous history is consumed through X and the triangular kernel through G. The shared modulus row M and quadrature row Q supply the finite estimate data, while S records the partial-sum approximation used by the integral read.

The completion-facing row R is a handoff boundary rather than a fixed-point principle. The endpoint seal E is downstream of the finite rows, and H, C, P, N transport, replay, source, and name exactly that surface.

Thus successive approximation is present as finite BEDC ledger data only. No selected limit, Banach fixed point, quotient-history equality, countable-choice selector, or ambient completeness coordinate is available to the consumer. □

Falsifiable prediction: Every $\mathsf{VolterraIntegralOperator}^{\uparrow}$ operator read factors through an interval window, triangular time-order row, continuous-map input and kernel rows, an IntegralUp read, a shared UniformModulusUp ledger, finite quadrature windows, RealSeriesUp partial-sum rows, a UniformCompletionUp handoff, a RealUp endpoint seal, transport, continuation, provenance, and local naming.

Independence witness: integral, continuousmap, real, compactmetric, boundedlinearoperator, uniformmodulus, uniformcompletion, realseries: $\mathsf{VolterraIntegralOperator}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries triangular causality, causal kernel-window data, input and output histories, integral rows, quadrature windows, finite series approximations, completion handoff, Picard-window ledgers, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{VolterraIntegralOperator}^{\uparrow}$ packet is generated from an interval window, triangular time-order row, continuous-map input-history row, causal kernel-window row, IntegralUp row, shared UniformModulusUp row, finite quadrature row, RealSeriesUp partial-sum row, UniformCompletionUp handoff row, successive-approximation ledger, RealUp endpoint seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2335.1, Definition 2335.2, Definition 2335.3, Definition 2335.4, and Definition 2335.5 seed the finite Volterra operator boundary for triangular constructive integral-equation reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VolterraIntegralOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Banach fixed point, Fredholm inversion, global compactness, selected solution operators, host function extensionality, quotient equality of functions, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, triangular causality, quadrature-window exactness, finite RealSeries approximation, UniformCompletion handoff, successive-approximation ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.VolterraIntegralOperatorUp.

Chapter 2336

A Concrete Naming Certificate for $\mathsf{NeighborhoodFunction}^{\uparrow}$

Following Brouwer's neighbourhood-function analysis and Bishop-style constructive continuity, the $\mathsf{NeighborhoodFunction}^{\uparrow}$ packet records the finite-observation interface by which a functional on choice or Cantor data is read through inspected prefixes before any fan-theorem or fan-functional consumer treats it as continuous. It sits between the finite stream windows of chapter 126, the binary-prefix surface of chapter 1475, the bar and spread law rows of chapter 400 and chapter 1532, and the modulus-facing handoff of chapter 1525. It also reads choice-sequence schedules through chapter 1482. The chapter names a finite NameCert carrier for neighbourhood-function reads; it does not admit host functions, external function extensionality, countable choice, selected infinite branches, or an ambient continuity theorem as primitive data.

Definition 2336.1 (NeighborhoodFunction carrier)**.** *A* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (W, B, A, V, O, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that records which prefixes are inspected.*
2. *B is the* $\mathsf{CantorSpace}^{\uparrow}$ *or* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row read through the schedule W.*
3. *A is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admissibility row for the displayed prefixes.*
4. *V is the local value row, recording the first accepted value attached to an inspected prefix.*
5. *O is the bar-observation row consumed by* $\mathsf{FanTheorem}^{\uparrow}$ *and* $\mathsf{FanFunctional}^{\uparrow}$ *handoffs.*
6. *L is the finite prefix ledger, storing only prefix extension, prefix agreement, and value-stability entries.*
7. *H records componentwise* hsame *transport for the schedule, prefix, spread-law, value, bar-observation, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a neighbourhood request through W, B, A, V, O, L in that order.*
9. *P is the* Pkg *provenance over* $\mathsf{NeighborhoodFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, A, V, O, L, H, C, P, N$. It has no coordinate for a host functional, an infinite branch selector, external function extensionality, a global continuity theorem, countable choice, or a completed function space.

Lean4 ✓ `BEDC.Derived.NeighborhoodFunctionUp.NeighborhoodFunctionTasteGate single carrier alignment`

Theorem 2336.2 (NeighborhoodFunction NameCert obligations)**.** *For every accepted* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *carrier* $U = (W, B, A, V, O, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream schedule, Cantor or choice-sequence prefix row, spread-law admissibility row, local value row, bar-observation row, finite prefix ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted neighbourhood-function read must expose those rows before it may be consumed by* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{FanFunctional}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{SpreadLaw}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *siblings.*

Proof. Project the accepted carrier. The visible analytic data are the stream schedule W, the displayed prefix row B, the spread-law row A, the local value row V, the bar-observation row O, and the prefix ledger L.

Run finite prefix induction over the inspected $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{CantorSpace}^{\uparrow}$ reads. The empty prefix exposes only the initial schedule and admissibility rows. The extension step appends one displayed prefix entry, checks admissibility through A, and records the first value row V when the ledger permits it.

Prefix agreement is transported by H. Componentwise hsame keeps the compared schedules, prefixes, admissibility rows, values, and bar observations aligned, while C replays the same request order for downstream consumers.

Finally P, N source and name the packet. Since no host functional, infinite branch selector, external extensionality proof, countable-choice schedule, completed function space, or ambient continuity theorem appears in the tuple, the NameCert obligations are exhausted by the displayed rows. □

Theorem 2336.3 (NeighborhoodFunction fan-functional handoff)**.** *Every* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *handoff to* $\mathsf{FanFunctional}^{\uparrow}$ *factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{CantorSpace}^{\uparrow}\ \textit{or}\ \mathsf{ChoiceSequence}^{\uparrow},\ \mathsf{SpreadLaw}^{\uparrow},\ \mathsf{FanTheorem}^{\uparrow},\ \mathsf{FanFunctional}^{\uparrow}.$$

The handoff reads only inspected prefixes, prefix agreement under hsame*, displayed continuation replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}$ *ledger before the fan-functional consumer may ask for a modulus-facing value.*

Proof. Start with the finite schedule row W. A fan-functional request can inspect only the prefixes already exposed through B and admitted by the spread-law row A.

Apply finite prefix induction again. At each step, the ledger L either records that no value has yet been exposed or records the first value row V stable under the displayed prefix agreement.

The bar-observation row O is then the only handoff row available to the fan theorem and fan functional siblings. Transport through H and replay through C preserve the same prefix route, while P, N provide the package and local naming ledger.

Thus the handoff is finite-observation data. It does not export a host functional, selected infinite branch, external function extensionality proof, countable-choice schedule, completed function space, or unrecorded modulus. □

Closure status for $\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{NeighborhoodFunction}^{\uparrow}$ packet is generated from a StreamName finite-window schedule, a CantorSpace or ChoiceSequence prefix row, a SpreadLaw admissibility row, a local value row, a FanTheorem bar-observation row, a finite prefix ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2336.1, Definition 2336.2, and Definition 2336.3 seed the finite neighbourhood-function interface for prefix-based fan and choice-sequence reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeighborhoodFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function extensionality, global continuity, selected infinite branches, countable choice, completed function spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-induction exactness, prefix-agreement transport, bar-observation routing, fan-functional handoff, and ledger non-escape for BEDC.Derived.NeighborhoodFunctionUp.

Chapter 2337

A Concrete Naming Certificate for MetaCICNormalizationWindowConfluence$^{\uparrow}$

The MetaCICNormalizationWindowConfluence$^{\uparrow}$ packet realizes the candidate-mediated normalization-window and confluence-boundary branch of chapter 9383 as finite NameCert data. It consumes the candidate-mediated strong-normalization boundary of chapter 2274 and the local critical-pair boundary of chapter 2212 before exporting its normalization-window confluence boundary. It packages candidate evidence, closed-candidate evidence, candidate-set rows, Pi and application adequacy rows, typed example reduction traces, closed-normal rows, transport, replay, provenance, and local naming before any consumer treats a MetaCIC normalization read as accepted. The packet is not a global strong-normalization theorem, not a closed confluence theorem, and not a subject-reduction closure theorem.

The packet also consumes the confluence-audit boundary of chapter 735, the normalization frontier of chapter 892, and the bridge surface of chapter 1190. These siblings provide the conditional confluence rows, candidate-mediated normalization frontier, and bridge-facing nonescape boundary that this chapter packages into one finite normalization-window carrier.

Definition 2337.1 (MetaCIC normalization-window confluence carrier)**.** *A* MetaCICNormalizationWindowConfluen *carrier is a finite* BHist *packet*

$$M = (T, R, K, Q, S, A, B, D, E, Z, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed MetaCIC term-window row naming the finite expression and context fragment being audited.*
2. *R is the typed reduction-trace row, read through the local typed-example reduction surface.*
3. *K is the candidate-evidence row for the audited term window.*
4. *Q is the closed-candidate evidence row for closed subwindows.*
5. *S is the candidate-set membership row.*
6. *A is the Pi-adequacy row for dependent function windows.*
7. *B is the application-adequacy row for application windows.*
8. *D is the closed-normal row reached after the accepted candidate-mediated route.*

9. *E is the confluence-boundary row, recording the refusal to promote local window agreement to global closed confluence.*
10. *Z is the decidable-boundary row, storing the exact finite boundary at which beta-step inspection is admitted.*
11. *H records componentwise* hsame *transport for term windows, traces, candidate rows, adequacy rows, closed-normal rows, boundary rows, replay, provenance, and naming.*
12. *C records displayed* Cont *replay from a normalization-window request through $T, R, K, Q, S, A, B, D, E, Z$ in that order.*
13. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}$*, candidate evidence, closed-candidate evidence, candidate sets, Pi adequacy, application adequacy, typed example reduction traces, closed-normal rows,* BHist, hsame, Cont, *and* NameCert*.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, K, Q, S, A, B, D, E, Z, H, C, P, N$. It has no coordinate for global subject reduction, global strong normalization, global confluence, proof irrelevance, host conversion equality, or a closed consistency theorem.

Lean4 ✓ `BEDC.Derived.MetaCICNormalizationWindowConfluenceUp.MetaCICNormalizationWindowConfluenceTasteGa`
`single carrier alignment`

Theorem 2337.2 (MetaCIC normalization-window confluence NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}$ *carrier $M = (T, R, K, Q, S, A, B, D, E, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}}$ *surface consists exactly of carrier admission for the typed term window, typed reduction trace, candidate evidence, closed-candidate evidence, candidate-set membership, Pi adequacy, application adequacy, closed-normal row, confluence-boundary row, decidable beta-step boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Lean TasteGate work for this packet must supply Nontrivial and FieldFaithful instances for the carrier rows and classifier projections before any bridge-checked reading is available.*

Proof. Project the accepted carrier. The typed expression and reduction rows are T, R, the candidate-mediated normalization rows are K, Q, S, A, B, D, and the boundary rows are E, Z.

Carrier admission first checks the candidate evidence consumed by the read. Closed subwindows must pass through Q, candidate-set membership through S, Pi windows through A, and application windows through B before the closed-normal row D is exposed.

Transport is componentwise through H. It preserves each candidate, adequacy, closed-normal, and boundary row without replacing them by host conversion equality or a global confluence premise.

Replay through C fixes the order of inspection, while P, N source and name the packet. The TasteGate burden is therefore precise: Nontrivial and FieldFaithful instances must witness that these displayed rows are not collapsed by the classifier or by the public carrier surface. □

Theorem 2337.3 (MetaCIC normalization-window confluence nonescape)**.** *Every accepted* MetaCICNormalizationW
read factors through candidate evidence, closed-candidate evidence, candidate-set membership, Pi adequacy, application adequacy, and closed-normal rows before the confluence-boundary row may be consumed. Consequently the packet exports no global strong-normalization theorem, no global confluence theorem, no subject-reduction closure theorem, no proof-irrelevance principle, no host conversion equality, and no closed consistency theorem.

Proof. Begin with a normalization-window request. The trace row R can be inspected only after the typed term window T is fixed.

The request then enters the candidate-mediated route. Candidate evidence K, closed-candidate evidence Q, candidate-set membership S, Pi adequacy A, and application adequacy B are read before the closed-normal row D.

Only after that route is displayed may the confluence-boundary row E and the decidable beta-step boundary Z be consumed. Componentwise transport, continuation replay, provenance, and naming preserve the same finite row order.

Since the carrier has no field for global subject reduction, global strong normalization, global confluence, proof irrelevance, host conversion equality, or closed consistency, no accepted read can escape to those principles. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}$ read factors through candidate evidence, closed-candidate evidence, candidate-set membership, Pi adequacy, application adequacy, and closed-normal rows before any confluence-boundary row is consumed.

Independence witness: metacicconfluenceauditpacket, metacicnormalizationfrontier, typed-example-reduction-trace, metacicdecidableboundary, betastepboundary: none is bijective to MetaCICNormalizationWindowC because this packet jointly carries typed term windows, reduction traces, candidate rows, closed-candidate rows, candidate-set rows, Pi and application adequacy, closed-normal rows, confluence and beta-step boundaries, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}$ packet is generated from a typed term window, typed reduction trace, candidate evidence, closed-candidate evidence, candidate-set membership, Pi adequacy, application adequacy, closed-normal rows, confluence-boundary rows, beta-step boundary rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2337.1, Definition 2337.2, and Definition 2337.3 seed the finite candidate-mediated normalization-window and confluence-boundary carrier.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationWindowConfluence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, global confluence, subject reduction, proof irrelevance, host conversion equality, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate route exactness, Pi and application adequacy preservation, closed-normal routing, confluence-boundary non-escape, beta-step boundary exactness, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICNormalizationWindowConfluenceUp.

Chapter 2338

A Concrete Naming Certificate for $\mathsf{SturmComparison}^{\uparrow}$

Following Sturm comparison theory and Bishop–Bridges constructive analysis, the $\mathsf{SturmComparison}^{\uparrow}$ packet records a finite comparison-neighbour surface for two displayed scalar ODE rows. It consumes the local-flow source of chapter 193, the continuous-map handoff of chapter 130, the finite observation windows of chapter 126, the regular rational readbacks of chapter 301, the Sturm root-isolation sibling of chapter 1534, and the terminal real-name seal of chapter 38. The packet names a local zero-order comparison certificate; it does not assert global ODE completeness, host solution equality, excluded-middle sign trichotomy, quotient real equality, or a selected classical solution curve.

Definition 2338.1 (Sturm comparison carrier)**.** *A* $\mathsf{SturmComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O_0, O_1, K, Z, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O_0 and O_1 are the two scalar* $\mathsf{Ode}^{\uparrow}$ *comparison-equation rows. Each row records its local time window, coefficient readbacks, initial endpoint data, and solution-window request through the displayed* $\mathsf{Ode}^{\uparrow}$ *local-flow surface.*
2. *K is the coefficient-order ledger. It records the finite comparison inequalities between the two coefficient packets, including the time window on which the order is licensed and the continuous-map rows that transport coefficient observations.*
3. *Z is the finite zero-window ledger. It records the inspected intervals, endpoint signs or nonzero certificates, retained zero-neighbour rows, and the local interlacing request used by the comparison read.*
4. *W stores the* $\mathsf{StreamName}^{\uparrow}$ *windows for coefficient observations, solution observations, and zero-window endpoints.*
5. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks attached to the windows in W.*
6. *E stores the* $\mathsf{Real}^{\uparrow}$ *seals for the coefficient-order rows, endpoint observations, and zero-window boundaries after O_0, O_1, K, Z, W, R have been displayed.*
7. *H records componentwise* hsame *transport for both ODE rows, coefficient-order ledgers, zero-window rows, stream windows, regular readbacks, real seals, continuation replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a comparison request through the two ODE rows, coefficient order, zero windows, readbacks, seals, transport, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{SturmComparison}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SturmRootIsolation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for a host solution space, global existence theorem, solution equality, trichotomy oracle, quotient real stream, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.SturmComparisonUp.TasteGate.SturmComparisonTasteGate single carrier alignment`

Theorem 2338.2 (Sturm comparison NameCert obligations)**.** *For every accepted* $\mathsf{SturmComparison}^{\uparrow}$ *carrier* $S = (O_0, O_1, K, Z, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SturmComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the two displayed* $\mathsf{Ode}^{\uparrow}$ *comparison rows, coefficient-order ledger, finite zero-window ledger, stream windows, regular readbacks, real seals, transport, replay, provenance, and local naming;*
2. *coefficient-order exactness, requiring every comparison inequality to pass through K and through the continuous-map rows named by the coefficient readbacks;*
3. *zero-window exactness, requiring every zero-neighbour and interlacing read to pass through Z before any* $\mathsf{Real}^{\uparrow}$ *seal is exported;*
4. *readback confinement, requiring the windows W and readbacks R to mediate all coefficient, solution, and endpoint observations;*
5. *componentwise* hsame *transport and displayed* Cont *replay through H and C;*
6. *provenance and local naming through P and N.*

No comparison obligation outside these rows is part of the packet.

Proof. Project the accepted carrier of Definition 2338.1. The two ODE rows O_0, O_1 supply the only local-flow source data, so a comparison read cannot inspect a solution before the displayed ODE windows and endpoint rows have been opened.

The coefficient-order surface is K. Its entries are finite Real-facing reads transported through continuous-map rows and the window/readback pair W, R, so the order relation is a certificate row rather than a host comparison of functions.

The zero-window surface is Z. It records the inspected windows, retained endpoint signs or nonzero witnesses, and the interlacing request. The $\mathsf{Real}^{\uparrow}$ seals in E are downstream of Z, hence zero-order information cannot bypass the finite zero ledger.

Finally H, C, P, N transport, replay, source, and name the same route. Since none of these structural rows adds a host solution space, quotient stream, trichotomy principle, or ambient completeness witness, the listed rows exhaust the local NameCert obligations. □

Theorem 2338.3 (Sturm comparison zero-order handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or zero-window consumer of an accepted* $\mathsf{SturmComparison}^{\uparrow}$ *packet reads the route*

$$O_0, O_1 \longmapsto K \longmapsto Z \longmapsto W, R \longmapsto E \longmapsto H, C, P, N$$

before it may use the comparison conclusion. Thus the consumer receives two displayed $\mathsf{Ode}^{\uparrow}$ *comparison rows, coefficient-order ledgers, finite zero-window rows, stream windows, regular readbacks, real seals, componentwise transport, displayed continuation replay, provenance, and local naming. It receives no global ODE completeness theorem, host solution equality, excluded-middle sign trichotomy, quotient real equality, or selected classical solution curve.*

Proof. Begin with a consumer request for a comparison conclusion. The only source rows available are the two $\mathsf{Ode}^{\uparrow}$ rows O_0, O_1, so the request is already confined to displayed local-flow data.

Read the coefficient order through K. The order ledger is then tied to the finite observation rows W, R, which expose coefficient and endpoint values through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rather than through host real functions.

Read the zero-window ledger Z. The retained windows and interlacing request are the only zero-order data visible to the consumer, and the real seals E are produced only after those windows have been displayed.

The rows H, C, P, N repack the same route as transport, replay, provenance, and local naming. Therefore every accepted handoff is the finite comparison packet itself, with no separate classical comparison theorem or ambient completeness principle exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SturmComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{SturmComparison}^{\uparrow}$ packet is generated from two displayed $\mathsf{Ode}^{\uparrow}$ comparison rows, coefficient-order ledgers, finite zero-window rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{Real}^{\uparrow}$ seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2338.1, Definition 2338.2, and Definition 2338.3 seed the finite comparison-neighbour packet over displayed ODE rows, coefficient order, zero windows, readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global ODE completeness, host solution equality, excluded-middle sign trichotomy, quotient real equality, selected classical solution curves, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-order exactness, zero-window exactness, readback confinement, consumer handoff, and structural stability for BEDC.Derived.SturmComparisonUp.

Chapter 2339

A Concrete Naming Certificate for $\mathsf{MaterialityOrbitEscape}^{\uparrow}$

The $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ packet realizes the materiality-orbit-escape branch of subsection 9462.27.7 as finite observational-context, feedback-prefix, identity-closure, orbit-rule, separation, transport, and local NameCert rows. It consumes the finite observation-window discipline of chapter 126, the observer-facing direction rows of chapter 1047, the closed-observation sibling of chapter 616, and the audit-facing report surface of chapter 1279. The packet names an indexed ledger judgment; it does not introduce context-free materiality, context-free free will, global observer substance, completed feedback streams, or a universal prediction oracle.

Definition 2339.1 (Materiality orbit escape carrier)**.** *A* $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, F, I, R, S, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the observational-context row. It stores the observer history, probe pattern, candidate interface, identity classifier, state classifier, finite rule family, and ledger provenance named by the originating vision.*
2. *F is the feedback-prefix row. It stores a finite response prefix and the finite observation windows by which the prefix is read.*
3. *I is the identity-closure row. It records the displayed classifier witnesses that keep the prefix traceable as one observed datum.*
4. *R is the rule-family row. It stores the currently available ledger-certified orbit rules and their source certificates.*
5. *S is the separation row. It records the positive separation witnesses used to refute orbit closure for each rule in R.*
6. *T is the indexed judgment row. It records whether the accepted read is material appearance through a closing rule or freedom-like appearance through certified escape from the displayed finite family.*
7. *H records componentwise* hsame *transport for observational context, feedback prefix, identity closure, rule family, separation rows, indexed judgment, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an observer-indexed request through O, F, I, R, S, T, H.*

9. *P is the* Pkg *provenance over* $\mathsf{MaterialityOrbitEscape}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ObserverDirection}^{\uparrow}$, ClosedObservationSyste $\mathsf{SocketReport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MaterialityOrbitEscape}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite indexed packet. It has no coordinate for a context-free materiality predicate, context-free free-will predicate, global observer, completed feedback stream, hidden rule family, or universal predictor.

Lean4 ✓ `BEDC.Derived.MaterialityOrbitEscapeUp.MaterialityOrbitEscapeTasteGate single carrier alignment`

Theorem 2339.2 (Materiality orbit escape NameCert obligations)**.** *For every accepted* $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ *carrier $M = (O, F, I, R, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MaterialityOrbitEscape}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the observational context, feedback prefix, identity-closure witnesses, finite rule family, separation witnesses, indexed judgment, transport, replay, provenance, and local naming;*
2. *context exactness, requiring every observer, probe, candidate interface, classifier, rule-family, and ledger read to pass through O;*
3. *prefix exactness, requiring every feedback response to pass through the finite prefix row F;*
4. *identity-closure exactness, requiring the sameness of the observed datum to be witnessed by I before any material or freedom-like judgment is exposed;*
5. *rule-family and separation exactness, requiring material appearance to read a displayed closing rule from R and freedom-like appearance to read a positive separation row from S for each rule in R;*
6. *structural transport, continuation replay, provenance, and local naming through H, C, P, N.*

No obligation outside these rows is part of the packet.

Proof. Project the accepted carrier of Definition 2339.1. The observational context row O fixes the observer history, probe pattern, candidate interface, classifiers, finite rule family, and ledger source. Thus no judgment can be read independently of its context.

The feedback prefix is F, and identity closure is I. A material or freedom-like read must first expose the finite response prefix and the classifier witnesses that keep it traceable as one datum.

The rule-family row R and separation row S distinguish the two public cases. Material appearance reads a closing rule from R; freedom-like appearance reads, for each displayed rule in R, a positive separation witness in S.

Finally H, C, P, N only transport, replay, source, and name the same finite route. They add no global observer, completed stream, context-free predicate, hidden rule family, or universal predictor, so the listed rows exhaust the local NameCert obligations. □

Theorem 2339.3 (Materiality orbit escape rule-family nonescape)**.** *Every public* $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ *read is indexed by the same observational context and finite rule family that appear in its carrier. A consumer may project a material-appearance read only through $O, F, I, R, T, H, C, P, N$ with an orbit-closing rule in R, and may project a freedom-like read only through $O, F, I, R, S, T, H, C, P, N$ with a displayed separation witness for each rule in R. The packet exports no judgment after replacing R by a hidden larger family, deleting I, or erasing the observer and prefix indices.*

Proof. Read the public judgment row T. Its source coordinates are the context O, prefix F, identity-closure row I, and displayed finite rule family R.

In the material case, T records a rule from R together with the orbit-closing ledger for the prefix. Removing R or replacing it by a hidden family changes the source row and therefore changes the carrier.

In the freedom-like case, T records certified escape from the same family R by using the separation row S. Each rule admitted by R must have a corresponding positive separation witness before the public read is exposed.

The structural rows H, C, P, N preserve this indexing under transport and replay. Hence the consumer cannot de-index the judgment, drop identity closure, or read escape relative to an undisplayed rule family. □

Falsifiable prediction: An accepted $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ packet is refuted by any public freedom-like read that omits identity closure, changes the displayed finite rule family, or lacks a positive separation witness for one of that family's rules.

Independence witness: observerdirection, closedobservationsystem, socketreport, streamname: none is bijective to $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ because this packet jointly carries observer-indexed context, finite feedback prefix, identity closure, displayed rule family, separation witnesses, indexed judgment, transport, replay, provenance, and local naming.

The future Lean TasteGate file for this packet must provide explicit $\mathsf{Nontrivial\ MaterialityOrbitEscape}^{\uparrow}$ and $\mathsf{FieldFaithful\ MaterialityOrbitEscape}^{\uparrow}$ instances. The nontrivial witness must separate a material-closing read from an escape read with the same observer and prefix but different rule-family/separation rows, and field faithfulness must preserve $O, F, I, R, S, T, H, C, P, N$ as distinct carrier coordinates.

Closure status for $\mathsf{NameCert}_{\mathsf{MaterialityOrbitEscape}^{\uparrow}}$.

Constructive story: A $\mathsf{MaterialityOrbitEscape}^{\uparrow}$ packet is generated from an observational-context row, finite feedback-prefix row, identity-closure witnesses, finite rule-family rows, positive separation rows, an indexed material-or-escape judgment, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaterialityOrbitEscape}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2339.1, Definition 2339.2, and Definition 2339.3 seed the finite observer-indexed materiality and orbit-escape packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaterialityOrbitEscape}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close context-free materiality, context-free free will, global observer substance, completed feedback streams, hidden rule families, universal prediction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, context exactness, prefix exactness, identity-closure exactness, rule-family and separation exactness, nonescape, and TasteGate field faithfulness for BEDC.Derived.MaterialityOrbitEscapeU

The $\mathsf{DominatedConvergence}^{\uparrow}$ hub routes its spine through chapter 92, chapter 123, chapter 3447, and chapter 2094 before the limit-of-integrals readback.

Chapter 2340

A Concrete Naming Certificate for DominatedConvergence$^{\uparrow}$

Following Lebesgue's convergence theorem and Bishop-style constructive integration, the DominatedConvergence$^{\uparrow}$ packet records the finite visible route by which pointwise limit data, an integrable majorant, and integral readback rows are retained before any limit-of-integrals consumer reads the conclusion. It sits after the measure surface of chapter 92, the public integral row of chapter 123, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the series-style bounded handoff of chapter 2094. The lower-bound side is kept compatible with chapter 3447: FatouLemma$^{\uparrow}$ supplies the nonnegative liminf route, while this packet adds the displayed integrable majorant and limit-window rows needed for dominated integral readback. The packet names dominated convergence as finite NameCert data; it does not assert ambient measure-theoretic completeness, quotient function equality, host almost-everywhere equality, selected representatives, or an unbounded convergence theorem.

Definition 2340.1 (Dominated convergence carrier)**.** *A* DominatedConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, I, F, G, L, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* Measure$^{\uparrow}$ *source row, exposing only the finite measure requests that the convergence packet may inspect.*
2. *I is the* Integral$^{\uparrow}$ *readback row for the displayed integrands and the majorant.*
3. *F is the finite family row for the indexed integrands.*
4. *G is the integrable majorant row, read through the same measure and integral surfaces as I.*
5. *L is the pointwise-limit row, recording only the displayed convergence windows used by the packet.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for rational integral approximants and tolerance requests.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row reached after the measure, integral, majorant, limit, and readback rows have been displayed.*
8. *T is the sibling handoff row to* TanneryTheorem$^{\uparrow}$*, used only as a bounded finite-sum and series-style domination boundary.*

9. *H records componentwise* hsame *transports for M, I, F, G, L, R, E, T.*
10. *C is the displayed* Cont *replay of the same route.*
11. *P is the* Pkg *provenance over* DominatedConvergence$^{\uparrow}$, Measure$^{\uparrow}$, Integral$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, TanneryTheorem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{DominatedConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for ambient measure completion, quotient function equality, host almost-everywhere equality, selected representatives, countable choice, or a detached completeness theorem.

Lean4 ✓ `BEDC.Derived.DominatedConvergenceUp.DominatedConvergenceTasteGate single carrier alignment`

Theorem 2340.2 (Dominated convergence NameCert obligations)**.** *For every accepted* DominatedConvergence$^{\uparrow}$ *carrier $D = (M, I, F, G, L, R, E, T, H, C, P, N)$, the local* NameCert$_{\text{DominatedConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the measure row, integral row, indexed integrand family, majorant, pointwise-limit row, regular rational readback, Real seal, Tannery handoff, transport, replay, provenance, and local naming;*
2. *majorant visibility, requiring every domination read to pass through G together with the same* Measure$^{\uparrow}$ *and* Integral$^{\uparrow}$ *rows used by I;*
3. *limit-window visibility, requiring every pointwise convergence read to pass through the displayed row L before any integral-limit read is available;*
4. *readback exactness, requiring integral approximants and tolerance requests to pass through R before the* Real$^{\uparrow}$ *seal row E is consumed;*
5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No dominated-convergence obligation is stored outside D.

Proof. Project the carrier of Definition 2340.1. The projection exposes exactly the measure row, integral row, integrand-family row, majorant row, pointwise-limit row, regular rational readback, Real seal, Tannery handoff, and structural rows.

Majorant visibility is the row G together with M and I. A domination read therefore uses the same finite measure and integral surfaces as the integral readback, rather than a hidden host boundedness or function-equality predicate.

Limit-window visibility is the row L. The theorem permits a consumer to inspect only the displayed convergence windows before reaching the integral-limit read, and it does not add a coordinate for ambient almost-everywhere equality.

The readback and seal route is $R \longmapsto E$. The regular rational approximants and tolerance requests are displayed before the terminal real-name seal is available. The rows H, C, P, N transport, replay, source, and name this same packet, so the listed obligations exhaust the local surface. □

Theorem 2340.3 (Dominated convergence Fatou forward dependency)**.** *Any accepted* DominatedConvergence$^{\uparrow}$ *packet that consumes the Fatou forward row must keep the dominating bound, finite measurable-window rows, Fatou lower-bound route, and* Real$^{\uparrow}$ *endpoint seals visible before it exports the dominated-convergence conclusion. Concretely, the read factors through the displayed majorant row G, the*

finite $\mathsf{Measure}^{\uparrow}$ *and* $\mathsf{Integral}^{\uparrow}$ *rows* M, I*, the pointwise-limit window row* L*, the* $\mathsf{FatouLemma}^{\uparrow}$ *lower-bound boundary named by Definition 2340.7, the regular rational readback* R*, the terminal seal* E*, and the structural rows* H, C, P, N*. It cannot use measure completion, selected representatives, quotient equality of functions, host almost-everywhere equality, or countable choice.*

Proof. Project the carrier of Definition 2340.1. The dominating bound is the row G, the finite measurable-window data are exposed only through M, I, F, L, and the endpoint route is $R \longmapsto E$.

Apply the NameCert obligations of Definition 2340.2. Majorant visibility keeps every domination read on G together with M, I, limit-window visibility keeps every pointwise read on L, and readback exactness keeps the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of R.

Use the local Fatou forward route Definition 2340.7. It allows the Fatou-facing consumer to receive only the displayed measure, integral, majorant, pointwise-limit, regular rational, Real seal, Tannery handoff, transport, replay, provenance, and local naming rows.

Therefore the dominated-convergence conclusion is exported only after the majorant, finite measurable windows, Fatou lower-bound boundary, regular readback, endpoint seals, and structural rows are all visible. Since those rows contain no measure completion, selected representative, quotient function equality, host almost-everywhere equality, or countable-choice coordinate, the forward dependency cannot import any of them. □

Theorem 2340.4 (Dominated convergence majorant handoff)**.** *Every* $\mathsf{Integral}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DominatedConvergence}^{\uparrow}$ *carrier* $D = (M, I, F, G, L, R, E, T, H, C, P, N)$ *reads the majorant side of the packet only through the displayed route*

$$M \longmapsto I \longmapsto G \longmapsto L \longmapsto R \longmapsto E \longmapsto T \longmapsto H, C, P, N.$$

Thus the consumer first receives the measure row, integral readback row, integrable majorant, pointwise-limit windows, regular rational readback, Real seal, bounded Tannery handoff, componentwise transport, displayed replay, provenance, and local naming before it can use the dominated-convergence conclusion.

Proof. Start with an integral-facing or Real-facing read of D. By Definition 2340.1, the only measure and integral coordinates are M and I, and the only domination coordinate is G.

The limit part is forced through L and R. The row L displays the pointwise-limit windows, while R records the regular rational readback and tolerance requests before the terminal seal E is reached.

The bounded handoff is T. It points to the finite domination discipline already visible in $\mathsf{TanneryTheorem}^{\uparrow}$ without importing the ambient dominated convergence theorem as a hidden premise.

Finally, H, C, P, N preserve and name the same route. Since the carrier lacks coordinates for measure completion, quotient function equality, host almost-everywhere equality, selected representatives, and countable choice, every consumer factors through the displayed majorant handoff. □

Theorem 2340.5 (Dominated convergence measure-integral route)**.** *Every dominated-convergence consumer of an accepted* $\mathsf{DominatedConvergence}^{\uparrow}$ *carrier* $D = (M, I, F, G, L, R, E, T, H, C, P, N)$ *factors through the* $\mathsf{Measure}^{\uparrow}$ *row,* $\mathsf{Integral}^{\uparrow}$ *row, majorant row, pointwise-limit row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{TanneryTheorem}^{\uparrow}$ *handoff before it consumes the conclusion. The route is anchored by chapter 92, chapter 123, chapter 301, chapter 38, and chapter 2094, with* H, C, P, N *preserving and naming exactly those rows.*

Proof. Project the carrier of Definition 2340.1. The first available analytic rows are M, I, G, L, R, E, T, and the structural rows are H, C, P, N.

The measure and integral stages are M and I. They bind the consumer to the finite measure and integral surfaces before any domination or limit row is read.

The domination and convergence stages are G and L. The majorant row is read through the same measure and integral surfaces, while the pointwise-limit row exposes only the displayed convergence windows.

The readback and sibling handoff are R, E, T. The regular rational row and terminal real seal are consumed before the Tannery handoff, and the transport, replay, provenance, and naming rows preserve this route without adding host almost-everywhere equality, selected representatives, or ambient completeness. □

Theorem 2340.6 (Dominated convergence Fatou–Tannery nonescape)**.** *Every lower-bound or bounded-series consumer of an accepted* $\mathsf{DominatedConvergence}^{\uparrow}$ *carrier* $D = (M, I, F, G, L, R, E, T, H, C, P, N)$ *reads the Fatou nonnegative route and the Tannery bounded handoff only through the displayed rows*

$$M, I, G, L, R, E, T, H, C, P, N.$$

In particular, the consumer may use $\mathsf{FatouLemma}^{\uparrow}$ *only as the nonnegative lower-bound boundary compatible with* M, I, G, L, R, E*, and may use* $\mathsf{TanneryTheorem}^{\uparrow}$ *only through the bounded handoff row* T*. It cannot import host almost-everywhere equality, selected representatives, quotient function equality, ambient measure completion, or a detached completion theorem at the Fatou–Tannery boundary.*

Proof. Project the carrier of Definition 2340.1. The carrier exposes the majorant row G, the pointwise-limit windows L, the regular rational readback R, the Real seal E, the Tannery handoff T, and the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 2340.2. Majorant visibility forces every domination read through G together with M and I, while limit-window visibility forces every convergence read through L before integral-limit readback is available.

Use the majorant handoff of Definition 2340.4 and the measure-integral route of Definition 2340.5. These two routes place the lower-bound side next to the $\mathsf{FatouLemma}^{\uparrow}$ nonnegative boundary and place the bounded-series side next to the $\mathsf{TanneryTheorem}^{\uparrow}$ row T without adding any new carrier coordinate.

It follows that the Fatou-facing and Tannery-facing reads are exhausted by $M, I, G, L, R, E, T, H, C, P, N$. Since host almost-everywhere equality, selected representatives, quotient function equality, ambient measure completion, and detached completeness are absent from that list, none can escape through this boundary. □

Falsifiable prediction: Every $\mathsf{DominatedConvergence}^{\uparrow}$ integral-facing or Real-facing read factors through displayed Measure, Integral, majorant, pointwise-limit, RegSeqRat readback, Real seal, Tannery handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: measure, integral, regseqrat, real, tannery-theorem: $\mathsf{DominatedConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries measure source, integral readback, indexed integrand family, integrable majorant, pointwise-limit windows, regular rational approximants, Real seal, bounded Tannery handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DominatedConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DominatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2340.1, Definition 2340.2, and Definition 2340.4 seed the finite dominated-convergence route through Measure$^{\uparrow}$, Integral$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and TanneryTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{DominatedConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient measure-theoretic completeness, quotient function equality, host almost-everywhere equality, selected representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, majorant visibility, limit-window visibility, regular rational readback exactness, Real seal handoff, transport and replay stability, and nonescape for BEDC.Derived.DominatedConvergenceUp.

Constructive story: A DominatedConvergence$^{\uparrow}$ packet is generated from a displayed Measure row, Integral readback row, indexed integrand-family row, integrable majorant row, pointwise-limit windows, RegSeqRat readback, terminal Real seal, bounded Tannery handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 2340.7 (Dominated convergence Fatou forward route)**.** *Every* FatouLemma$^{\uparrow}$ *consumer using chapter 2340 receives only the displayed* Measure$^{\uparrow}$ *row sourced from chapter 92,* Integral$^{\uparrow}$ *row, integrable majorant row, pointwise-limit row,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal,* TanneryTheorem$^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert$_{\text{DominatedConverge}}$ *rows of an accepted* DominatedConvergence$^{\uparrow}$ *packet. The forward route supplies a dominated upper-side boundary compatible with chapter 3447; it does not export ambient measure completion, quotient function equality, host almost-everywhere equality, selected representatives, countable choice, or a detached monotone-convergence theorem.*

Proof. Project the accepted carrier from Definition 2340.1. The consumer can see only the measure row, integral row, indexed family row, majorant row, pointwise-limit row, regular rational readback, Real seal, Tannery handoff, and the structural rows.

Apply the NameCert obligation theorem Definition 2340.2. Majorant visibility, limit-window visibility, readback exactness, and structural stability force every dominated-convergence read to remain inside those displayed rows.

Use the majorant handoff Definition 2340.4. It orders the route through Measure$^{\uparrow}$, Integral$^{\uparrow}$, the majorant, pointwise-limit windows, RegSeqRat$^{\uparrow}$ readback, Real$^{\uparrow}$ seal, TanneryTheorem$^{\uparrow}$ handoff, and H, C, P, N before any conclusion is consumed.

The FatouLemma$^{\uparrow}$ chapter chapter 3447 supplies the lower-bound side as its own finite NameCert route. Therefore the forward dependency from dominated convergence to Fatou receives only the displayed rows listed above, with no ambient measure completion, quotient equality, selected representative, countable-choice, or detached monotone-convergence data. □

Theorem 2340.8 (Dominated convergence Fatou–Tannery sibling lattice)**.** *Every* DominatedConvergence$^{\uparrow}$ *consumer that reaches* FatouLemma$^{\uparrow}$ *must pass through the displayed majorant row, pointwise-limit row,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, and* TanneryTheorem$^{\uparrow}$ *handoff row before the Fatou-side lower-bound boundary is consumed. The sibling lattice is therefore pinned to chapter 3447 and chapter 2094:* FatouLemma$^{\uparrow}$ *supplies the liminf-facing floor, while* TanneryTheorem$^{\uparrow}$ *supplies the bounded finite-exchange handoff. It does not export ambient measure completion, quotient function equality, selected representatives, countable choice, host almost-everywhere equality, or a detached monotone-convergence theorem.*

Proof. Project the accepted DominatedConvergence$^{\uparrow}$ carrier from Definition 2340.1. The only rows available to a Fatou-facing consumer are the displayed measure, integral, family, majorant, pointwise-limit, regular-readback, Real-seal, Tannery-handoff, transport, replay, provenance, and local naming rows.

Apply Definition 2340.7. It already places the Fatou-side route behind the majorant, pointwise-limit, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and TanneryTheorem$^{\uparrow}$ rows.

Now use the sibling anchors. The Fatou chapter chapter 3447 supplies the lower-bound comparison, and the Tannery chapter chapter 2094 supplies the bounded exchange row. Since both are reached only through the displayed lattice, no omitted host theorem or quotient equality can enter the DominatedConvergence packet. □

The horizontal lattice route for $\mathsf{DominatedConvergence}^{\uparrow}$ reads the lower-bound boundary from $\mathsf{FatouLemma}^{\uparrow}$ in chapter 3447 and the bounded handoff from $\mathsf{TanneryTheorem}^{\uparrow}$ in chapter 2094 before exporting its $\mathsf{Real}^{\uparrow}$ integral conclusion. Thus the dominated-convergence packet keeps its Fatou-side liminf floor and Tannery-side bounded exchange route visible as sibling rows, rather than as an ambient measure-completion theorem, a selected representative, quotient function equality, or a detached host convergence principle.

Theorem 2340.9 (Dominated convergence measure-integral sibling lattice)**.** *Every* $\mathsf{DominatedConvergence}^{\uparrow}$ *consumer must pass through the displayed sibling lattice before any dominated-convergence conclusion is read:*

1. *the* $\mathsf{Measure}^{\uparrow}$ *row of chapter 92 supplies the finite measurable-window source;*
2. *the* $\mathsf{Integral}^{\uparrow}$ *row of chapter 123 supplies the integral readback over that same source;*
3. *the* $\mathsf{FatouLemma}^{\uparrow}$ *row supplies only the nonnegative lower-bound boundary;*
4. *the* $\mathsf{TanneryTheorem}^{\uparrow}$ *row supplies only the bounded finite-exchange handoff;*
5. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows supply the regular rational readback and terminal Real seal.*

Thus the route records measure, integral, Fatou, Tannery, regular-readback, and Real-seal dependencies as visible sibling data. It does not export ambient measure completion, quotient function equality, selected representatives, host almost-everywhere equality, or countable choice.

The dependency lattice for $\mathsf{DominatedConvergence}^{\uparrow}$ consumes the measure carrier of chapter 92, the integral readback surface of chapter 123, and the lower-bound route of chapter 3447 before any dominated integral conclusion is exported. Thus the packet keeps the measure row, integral row, Fatou boundary, majorant row, convergence windows, and endpoint readback visible as sibling dependencies, with no ambient measure completion, selected representative, quotient function equality, host almost-everywhere equality, or countable-choice route.

Chapter 2341

A Concrete Naming Certificate for $\mathsf{CauchyNetRegularization}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Moore–Smith nets, the $\mathsf{CauchyNetRegularization}^{\uparrow}$ packet records the finite regularization surface by which directed Cauchy-net evidence is repacked before $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumers read it. It sits after the directed-window packet of chapter 589, the regularization route of chapter 376, the finite-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the completion landing surface of chapter 1506, and the terminal real-name seal of chapter 38. The packet is a finite carrier for regularized directed evidence; it is not a quotient of nets, not a selected subnet, not a host completion theorem, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2341.1 (CauchyNetRegularization carrier)**.** *A* $\mathsf{CauchyNetRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, S, R, T, A, E, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row whose inspected vertices and common upper-window witnesses provide the source evidence.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule exposing the finite observation windows read from D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled observations.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic radii, rational comparison cells, and late-window bounds used by the regularization.*
5. *A is the finite regularization row: for each displayed tolerance request it chooses a named regular window already justified by the directed common-window witness in D and the tolerance ledger T.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the route D, S, R, T, A has been displayed.*
7. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row available only after the same regularized route has been displayed.*
8. *H stores componentwise* hsame *transport for the directed evidence, schedules, readbacks, tolerances, regularization row, seal row, completion handoff, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from directed windows to scheduled observations, regular readback, dyadic tolerance checks, regularized windows, and then to the real and metric-completion consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyNetRegularization}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient-net equality, selected representatives, cofinal choice functions, host metric completeness, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyNetRegularizationUp.CauchyNetRegularizationTasteGate single carrier alignment`

Theorem 2341.2 (CauchyNetRegularization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyNetRegularization}^{\uparrow}$ *carrier $G = (D, S, R, T, A, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyNetRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, directed-window evidence through D, finite schedule exposure through S, regular rational readback through R, dyadic tolerance and comparison exactness through T, regularized-window selection through A, terminal real sealing through E, metric-completion handoff through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 2341.1. The accepted packet exposes the directed-window evidence, schedule, readback, tolerance ledger, regularization row, two consumer rows, and the structural rows H, C, P, N.

Read the evidence route in order. The directed row D supplies only displayed finite common-window witnesses; the schedule S and readback row R then expose the observations that are available to a regularization consumer. The dyadic row T records the tolerance and comparison cells attached to those observations.

The regularization row A is therefore not an independent selector. It chooses only among windows already justified by D, S, R, T, and the terminal rows E and M may be read only after that route has been displayed.

Rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no quotient-net, selected-subnet, cofinal-choice, host-completeness, open-cover, or ambient real-completeness coordinate, none of those principles is part of the local NameCert surface. □

Theorem 2341.3 (CauchyNetRegularization ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchyNetRegularization}^{\uparrow}$ *packet factors through exactly the regularized directed route*

$$D \longmapsto S \longmapsto R \longmapsto T \longmapsto A \longmapsto (E, M)$$

together with H, C, P, N. Consequently the consumer sees finite Cauchy-net evidence regularized into displayed StreamName, RegSeqRat, and DyadicRatCore rows before reaching either terminal surface, and it receives no quotient net, selected subnet, cofinal choice function, host completion theorem, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the source row D. By Definition 589.2, Cauchy-net evidence is available only as displayed directed-window data with common-window witnesses.

Next read the regularization route. The scheduled windows S, regular readback R, and dyadic tolerance ledger T are the only rows from which A may form a regularized window. Thus every regularized observation is tied to the same finite directed witness data rather than to an external subnet or host net.

Finally inspect the two terminal rows. The real seal E and metric-completion handoff M are downstream of A and share the same transport, replay, provenance, and local naming rows. Removing any row from $D, S, R, T, A, H, C, P, N$ breaks the displayed route, so the ledger is exact for the stated consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyNetRegularization}^{\uparrow}$ packet is generated from displayed CauchyNet evidence, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a regularized-window ledger, Real and MetricCompletion consumer rows, componentwise transport, displayed continuation replay, package provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2341.1, Definition 2341.2, and Definition 2341.3 seed the finite regularization route from directed Cauchy-net evidence to Real and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyNetRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-net equality, selected subnets, cofinal choice, host completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, directed-window evidence exposure, regularized-window exactness, Real seal routing, MetricCompletion handoff, componentwise transport, continuation replay, provenance, and local naming for BEDC.Derived.CauchyNetRegularizationUp.

Chapter 2342

A Concrete Naming Certificate for ChoiceFreeRegularCauchyModulus$^{\uparrow}$

Following Bishop and Bridges on constructive real analysis with explicit moduli, the ChoiceFreeRegularCauchyModul packet records the finite L10 route by which a regular Cauchy modulus is read from displayed dyadic tolerances and regular rational windows before any Real or Completion consumer sees a seal. It sits after the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the Real seal of chapter 38, and the extraction discipline of chapter 641. The packet exposes a modulus for displayed requests only; it is not countable choice, a quotient stream equality, a selected limit, host real equality, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 2342.1 (Choice-free regular Cauchy modulus carrier)**.** *A* ChoiceFreeRegularCauchyModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, W, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, displaying the dyadic precision request, slack conversion, comparison cell, and local radius ledger consumed by the modulus read.*
2. *W is the* StreamName$^{\uparrow}$ *finite-window row, displaying exactly the source observations inspected for that tolerance request.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the windows in W, exposing the regular rational comparisons used by the modulus claim.*
4. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after D, W, R and the extracted modulus row have been displayed.*
5. *K is the local extraction row, read through the sibling* CauchyModulusExtraction$^{\uparrow}$ *surface and recording the displayed unary threshold that serves the request.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, window, readback, Real seal, extraction row, continuation route, provenance, and local naming row.*
7. *C records displayed* Cont *replay from a Real or Completion tolerance request through D, W, R, K, E and then through the structural rows.*

8. *P is the* Pkg *provenance over* ChoiceFreeRegularCauchyModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusExtraction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{ChoiceFreeRegularCauchyModulus}^{\uparrow}}$ *row.*

The no-choice ledger is explicit: the carrier stores no countable-choice functional, no quotient stream representative, no selected limit, no trichotomy oracle, no host real equality test, and no ambient completion principle.

Lean4 ✓ `BEDC.Derived.ChoiceFreeRegularCauchyModulusUp.ChoiceFreeRegularCauchyModulusTasteGate single carrier alignment`

Theorem 2342.2 (Choice-free regular Cauchy modulus obligations)**.** *For every accepted* ChoiceFreeRegularCauchyM *packet*

$$M = (D, W, R, E, K, H, C, P, N),$$

the local NameCert$_{\text{ChoiceFreeRegularCauchyModulus}^{\uparrow}}$ *obligations are exactly the following finite rows:*

1. *carrier admission for the dyadic tolerance row, finite StreamName window, RegSeqRat readback, extracted modulus row, Real seal, transport, replay, provenance, and local naming data;*

2. *dyadic tolerance visibility, requiring every modulus request to expose the precision, slack, comparison, and radius data in D before K is read;*

3. *finite-window coverage, requiring the source observations used by K to be the displayed windows in W;*

4. *regular readback stability, requiring the comparison data exported to Real and Completion consumers to pass through the* RegSeqRat$^{\uparrow}$ *row R before the terminal seal E;*

5. *extraction locality, requiring K to be read through the sibling* CauchyModulusExtraction$^{\uparrow}$ *surface over the same D, W, R route;*

6. *no-choice exhaustion, requiring every accepted consumer to read only $D, W, R, E, K, H, C, P, N$ and therefore to receive no countable choice, quotient stream equality, selected limit, trichotomy, host real equality, or ambient completeness coordinate.*

Proof. Carrier admission is projection from Definition 2342.1. The packet displays the tolerance request, finite windows, regular rational readback, extraction row, Real seal, and structural rows as one finite BHist object.

For a modulus request, the dyadic row D is read first. It fixes the requested tolerance and the finite comparison budget available to the consumer. The StreamName row W then fixes the inspected source windows, and R records the regular rational readback from those windows.

The extraction row K is local to that route. It is not a choice functional over all future requests; it records the displayed threshold obtained from the visible D, W, R rows through the CauchyModulusExtraction$^{\uparrow}$ surface.

Finally the seal row E and the structural rows H, C, P, N transport and replay the same route. Since the no-choice ledger lists every absent coordinate, no countable-choice functional, quotient stream equality, selected limit, trichotomy oracle, host equality test, or ambient completeness principle can be exported by the local NameCert surface. □

Closure status for NameCert$_{\text{ChoiceFreeRegularCauchyModulus}^{\uparrow}}$**.**

Constructive story: A ChoiceFreeRegularCauchyModulus$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName finite windows, RegSeqRat readback, a CauchyModulusExtraction row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{ChoiceFreeRegularCauchyModulus}\uparrow}$, seed)

Scope closed: Definition 2342.1 and Definition 2342.2 seed the finite no-choice modulus route over dyadic tolerance, window, readback, extraction, and Real seal rows.

Formal verification: FormalStatus(NameCert$_{\text{ChoiceFreeRegularCauchyModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, countable choice, quotient stream equality, selected limits, trichotomy, host real equality, ambient Real$^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance visibility, finite-window coverage, regular readback stability, extraction locality, no-choice exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.ChoiceFreeRegularCauchyModulusUp.

Chapter 2343

A Concrete Naming Certificate for $\mathsf{MinkowskiInequality}^{\uparrow}$

Following Hardy–Littlewood–Polya and standard normed-space analysis, the $\mathsf{MinkowskiInequality}^{\uparrow}$ packet records the finite RealUp route by which a triangle-inequality read over displayed vector or finite-series windows becomes available to downstream norm and functional-analysis consumers. It sits over the Holder inequality handoff of chapter 2100, the square-case boundary of chapter 1684, the finite partial-sum surface of chapter 1581, the norm rows of chapter 89, and the endpoint seals of chapter 38. The packet is a finite inequality consumer; it is not a host function-space norm theorem, a quotient stream equality, a selected infinite sum, a countable-choice principle, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 2343.1 (Minkowski inequality carrier)**.** *A* $\mathsf{MinkowskiInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, A, B, U, V, C, D, E, F, H, T, P, N)$$

with the following displayed rows:

1. *X and Y are the two finite vector or finite-series windows whose inspected coordinates are exposed through* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *Z is the displayed sum window, carrying exactly the coordinatewise or partial-sum rows read from X and Y.*
3. *A and B are the two* $\mathsf{Norm}^{\uparrow}$ *rows for the displayed inputs, including only the finite norm reads visible in X and Y.*
4. *U is the* $\mathsf{Norm}^{\uparrow}$ *row for the displayed sum window Z.*
5. *V is the* $\mathsf{HolderInequality}^{\uparrow}$ *handoff row used to compare the mixed finite products that appear in the triangle-inequality read.*
6. *C is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *square-case boundary, available when the exponent and norm rows specialize to the displayed square route.*
7. *D is the* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row, exposing only the finite sums of coordinates, powers, and comparison rows used by the carrier.*
8. *E is the dyadic tolerance budget for finite addition, norm comparison, Holder handoff slack, square-case slack, and endpoint rounding.*

9. *F is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal for the displayed Minkowski inequality read.*

10. *H records componentwise* hsame *transport for the finite windows, norm rows, Holder handoff, Cauchy–Schwarz boundary, RealSeries partial sums, dyadic budget, endpoint seal, replay, provenance, and local naming rows.*

11. *T records displayed* Cont *replay from any accepted coordinate or partial-sum request through $X, Y, Z, A, B, U, V, C, D, E, F$.*

12. *P is the* Pkg *provenance over* $\mathsf{MinkowskiInequality}^{\uparrow}$, $\mathsf{HolderInequality}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite Minkowski inequality packet. It has no coordinate for host function-space norms, quotient stream equality, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.MinkowskiInequalityUp.MinkowskiInequalityTasteGate single carrier alignment`

Theorem 2343.2 (Minkowski inequality NameCert obligations)**.** *For every accepted* $\mathsf{MinkowskiInequality}^{\uparrow}$ *carrier $K = (X, Y, Z, A, B, U, V, C, D, E, F, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the two displayed input windows, the displayed sum window, the two input norm rows, the sum norm row, the Holder handoff row, the Cauchy–Schwarz square-case boundary, the RealSeries partial-sum row, the dyadic tolerance budget, the endpoint seal, componentwise transport, displayed replay, package provenance, and local naming. Every accepted inequality request is classified only by the displayed finite windows, norm rows, Holder handoff, square-case boundary, finite partial sums, dyadic budget, and endpoint seal.*

Proof. Project the accepted carrier. The analytic coordinates are the finite windows X, Y, Z, the norm rows A, B, U, the Holder handoff V, the square-case boundary C, the partial-sum row D, the dyadic budget E, and the endpoint seal F.

Carrier admission reads exactly those displayed rows together with H, T, P, N. There is no carrier coordinate from which a host function-space norm, quotient representative, selected infinite sum, countable-choice selector, or ambient completeness witness could be recovered.

Classifier stability is componentwise. The hsame row transports the same finite windows, norm rows, handoff rows, partial-sum data, tolerance budget, and endpoint seal, while T replays the same finite route for later consumers.

Thus the local $\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}$ row names the displayed inequality packet and no additional analytic theorem. □

Theorem 2343.3 (Minkowski inequality Holder handoff)**.** *For every accepted* $\mathsf{MinkowskiInequality}^{\uparrow}$ *carrier $K = (X, Y, Z, A, B, U, V, C, D, E, F, H, T, P, N)$, every triangle-inequality-facing read factors through the finite route*

$$X, Y, Z \longmapsto A, B, U \longmapsto V \longmapsto C, D \longmapsto E \longmapsto F \longmapsto H, T, P, N.$$

Consequently the read uses only finite vector or finite-series windows, the two input norm rows and sum norm row, the $\mathsf{HolderInequality}^{\uparrow}$ *mixed-product handoff, the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *square-case boundary, the* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row, a dyadic tolerance budget, and a terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal. It supplies no host function-space norm, quotient stream equality, selected infinite sum, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-checked standard theorem.*

Proof. Start from the displayed windows. The carrier lets a consumer inspect only the input windows X, Y and the sum window Z before any norm comparison is available.

The norm comparison is then fixed by A, B, U and by the Holder handoff row V. The square-case boundary C records the Cauchy–Schwarz specialization, while the RealSeries row D records the finite partial sums used in the comparison.

The dyadic budget E and endpoint seal F complete the Real-facing read. The support rows H, T, P, N transport, replay, source, and name the same route, so no excluded host-side principle can enter the accepted packet. □

Falsifiable prediction: Every $\mathsf{MinkowskiInequality}^{\uparrow}$ triangle-inequality read factors through finite vector or series windows, two input norm rows, a sum norm row, HolderInequalityUp handoff, CauchySchwarzRealUp square-case boundary, RealSeriesUp partial-sum routing, a dyadic tolerance budget, a RealUp endpoint seal, transport, replay, provenance, and local naming.

Independence witness: holder-inequality, cauchy-schwarz-real, realseries, norm, real: $\mathsf{MinkowskiInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite windows, three norm rows, Holder mixed-product routing, Cauchy–Schwarz square-case routing, RealSeries partial-sum routing, dyadic tolerance accounting, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{MinkowskiInequality}^{\uparrow}$ packet is generated from two finite vector or finite-series input windows, a displayed sum window, two input norm rows, a sum norm row, HolderInequalityUp handoff, CauchySchwarzRealUp square-case boundary, RealSeriesUp partial-sum rows, a dyadic tolerance budget, a RealUp endpoint seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2343.1, Definition 2343.2, and Definition 2343.3 seed the finite Minkowski inequality boundary for displayed RealUp finite-vector and finite-series windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinkowskiInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space norms, quotient stream equality, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, norm-row exactness, Holder handoff, square-case boundary routing, finite-sum coverage, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.MinkowskiInequalityUp.

Chapter 2344

A Concrete Naming Certificate for $\mathsf{StoneCechCompactification}^{\uparrow}$

Following Gillman–Jerison and Engelking on compactifications, the $\mathsf{StoneCechCompactification}^{\uparrow}$ packet records the finite BEDC surface by which bounded continuous probes from a displayed completely regular source are read through compact Hausdorff target seals. It sits over the generated-open topology rows of chapter 88, the compact-metric witness discipline of chapter 129, the graph and continuity surface of chapter 130, the endpoint-stable limiting route of chapter 463, and the real-name seal of chapter 38. The packet names finite bounded-continuous-probe windows and compact-target seal rows only; it does not construct an ambient function algebra, choose ultrafilters, identify quotient extensions, or assert an external compactification theorem.

Definition 2344.1 (Stone–Cech finite probe carrier)**.** *A* $\mathsf{StoneCechCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, O, B, W, F, K, T, U, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-object row, read only through the displayed topology classifier of* $\mathsf{Topology}^{\uparrow}$*.*
2. *O is the finite generated-open row for the inspected source neighborhoods and separation tests.*
3. *B is the bounded-real range row, recording the finite real-name bounds consumed by bounded continuous probes.*
4. *W is the finite probe-window row, listing the bounded continuous probes whose values are inspected.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and continuity row for each probe in W.*
6. *K is the compact-target seal row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *or the displayed compact target surface available to the surrounding route.*
7. *T is the compact Hausdorff target-separation row for finite pairs of target observations.*
8. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row for replaying finite probe windows through stable target seals.*
9. *R is the* $\mathsf{Real}^{\uparrow}$ *value row attached to the bounded target observations.*
10. *H records componentwise* hsame *transport for source topology, bounded range, probe window, continuous-map graph, compact-target seal, target separation, uniform handoff, real-value, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a finite bounded-probe request through O, B, W, F, K, T, U, R.*
12. *P is the* Pkg *provenance over* $\mathsf{StoneCechCompactification}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{StoneCechCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, O, B, W, F, K, T, U, R, H, C, P, N$. It has no coordinate for ultrafilter choice, ambient bounded-function algebra, quotient extension equality, host compactification points, excluded middle, or external compact Hausdorff completion.

Lean4 ✓ `BEDC.Derived.StoneCechCompactificationUp.StoneCechCompactificationTasteGate single carrier alignment`

Theorem 2344.2 (Stone–Cech compactification NameCert obligations)**.** *For every accepted* StoneCechCompactifica *carrier*

$$S = (X, O, B, W, F, K, T, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StoneCechCompactification}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source topology row, generated-open inspections, bounded-real range row, finite probe window, continuous-map graph rows, compact-target seal, target separation rows, uniform handoff, real-value rows, transport, replay, provenance, and naming;*
2. *probe-window classifier, requiring every bounded continuous probe read to factor through W, F, B, R and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows;*
3. *compact-target handoff, requiring K, T to expose only the displayed compact target and target-separation rows before any compactification-facing consumer reads the seal;*
4. *transport and replay, requiring H, C to preserve the same finite packet under* hsame *comparison and displayed* Cont *observation replay;*
5. *ledger non-escape, requiring every topology, compactness, continuous-map, uniform-limit, and real-value consumer to read through $X, O, B, W, F, K, T, U, R, H, C, P, N$ rather than through ultrafilters, host compactification points, quotient extension equality, or an ambient bounded-function algebra.*

No Stone–Cech compactification obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2344.1. The accepted tuple exposes exactly the source topology, generated-open, bounded-range, probe-window, continuous-map, compact-target, target-separation, uniform-handoff, real-value, transport, replay, provenance, and local naming rows listed in the statement.

The probe-window classifier is finite. A probe request first selects an entry of W, then reads its graph and continuity data through F, its bounded range through B, and its displayed real values through R. No function algebra or selected infinite probe family is part of the carrier.

The compact-target handoff is read from K, T, U. The compact target seal and its finite separation rows are displayed before the uniform replay row can export a compactification-facing observation, so the handoff remains inside the target seal.

Transport, replay, provenance, and naming are supplied by H, C, P, N. Since the carrier has no coordinate for ultrafilter choice, quotient extension equality, host compactification points, excluded middle, or an external compact Hausdorff completion, the listed rows exhaust the local certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{StoneCechCompactification}^{\uparrow}}$.

Constructive story: A $\mathsf{StoneCechCompactification}^{\uparrow}$ packet is generated from a source topology row, generated-open inspections, bounded-real range data, finite bounded-continuous-probe windows, continuous-map graph rows, compact-target seals, target-separation rows, uniform replay, Real value rows, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneCechCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2344.1 and Definition 2344.2 seed the finite bounded-continuous-probe surface for compactification consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneCechCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ultrafilter existence, ambient bounded-function algebra, quotient extension equality, host compactification points, excluded middle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, probe-window classifier, compact-target handoff, transport, replay, provenance, local naming, and non-escape for the same finite packet.

Chapter 2345

A Concrete Naming Certificate for $\mathsf{UniformCompletionSeparationKernel}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley's separated uniform reflection, the $\mathsf{UniformCompletionSeparationKernel}^{\uparrow}$ packet names the finite interface that lets a completion-facing consumer pass from displayed Cauchy data to a separated comparison boundary. It sits between the metric completion surface of chapter 1506, the uniform completion route of chapter 1473, and the zero-distance exactness boundary of chapter 1411.

Definition 2345.1 (UniformCompletionSeparationKernel carrier)**.** *A* $\mathsf{UniformCompletionSeparationKernel}^{\uparrow}$ *carrier is a tuple*

$$K = (M, U, S, F, N, R, H, C, P, L)$$

with the following displayed rows.

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row read by metric completion consumers before any separated comparison is requested.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row recording the uniform-completion handoff surface over the same displayed Cauchy data.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance row used as the exact separated-kernel boundary.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch, exposing finite filter evidence for completion-facing reads.*
5. *N is the* $\mathsf{CauchyNet}^{\uparrow}$ *branch, exposing finite net evidence over the same completion route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback row through which terminal precision claims are sealed.*
7. *H is componentwise* hsame *transport for M, U, S, F, N, R.*
8. *C is the displayed* Cont *replay from Cauchy evidence to the completion handoff and separated comparison.*
9. *P is the* Pkg *provenance row over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionSeparationKernel}^{\uparrow}}$ *row.*

Theorem 2345.2 (UniformCompletionSeparationKernel NameCert obligations)**.** *For an accepted carrier $K = (M, U, S, F, N, R, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionSeparationKernel}^{\uparrow}}$ *obligation surface consists of carrier admission, uniform-completion handoff, separated-kernel exactness,*

filter and net branch visibility, RegSeqRat or Real seal nonescape, and componentwise transport. Every completion-facing read must pass through the displayed route

$$M \to U \to S \to F, N \to R \to H, C, P, L.$$

The packet exports no quotient completion, selected representative, countable choice schedule, host equality, open-cover compactness theorem, or Real completeness row.

Proof. Carrier admission is read directly from Definition 2345.1. The tuple displays the metric-completion source, the uniform-completion handoff, the separated zero-distance boundary, both Cauchy evidence branches, the terminal readback row, transport, replay, provenance, and local naming.

The uniform-completion handoff uses U only after M has been projected. Thus a completion-facing consumer cannot bypass the metric-completion source or insert a separate quotient-completion object before the uniform route is exposed.

Separated-kernel exactness is the role of S. Any comparison of two displayed completion reads is forced through the $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance row, so the packet never supplies host equality or selected representatives.

The filter and net branches are visible as the separate rows F and N. They can be inspected as finite BEDC evidence, but neither branch introduces countable choice, an unbounded selector, open-cover compactness, or an ambient completeness theorem.

Finally R, H, C, P, L seal the route componentwise. The RegSeqRat or Real readback is reached only after the displayed completion and separated-boundary rows, while transport, replay, provenance, and local naming preserve the same finite route for every consumer. □

Theorem 2345.3 (UniformCompletionSeparationKernel filter-net branch visibility)**.** *Let* $K = (M, U, S, F, N, R, H, C, P, L)$ *be an accepted* $\mathsf{UniformCompletionSeparationKernel}^{\uparrow}$ *carrier. Every completion-facing read from* K *exposes both the* $\mathsf{CauchyFilter}^{\uparrow}$ *branch row* F *and the* $\mathsf{CauchyNet}^{\uparrow}$ *branch row* N *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row* R *is consumed.*

The read therefore factors through the displayed $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, and* $\mathsf{SeparatedMetric}^{\uparrow}$ *route before it reaches either terminal seal. It cannot be replaced by a hidden quotient completion, selected limit row, host equality, or one-branch completion presentation.*

Lean4 ✓ `BEDC.Derived.UniformCompletionSeparationKernelUp.TasteGate.UniformCompletionSeparationKernelFil`

Proof. Project the accepted carrier. The coordinates M, U, S expose the metric-completion source, the uniform-completion handoff, and the separated zero-distance boundary before any terminal readback is available.

The branch rows are separate coordinates. The filter branch F and the net branch N are both displayed before R, so a completion-facing consumer cannot consume the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal while hiding one of the two Cauchy presentations.

Now use the ledger order recorded by H, C, P, L. Componentwise transport and continuation replay preserve the route through M, U, S, F, N before R, while provenance and local naming record that route as the only public packet surface.

Ledger non-escape excludes any hidden quotient completion, selected limit, host equality, or one-branch substitute. Such data are absent from the carrier coordinates and cannot be reconstructed from the displayed NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionSeparationKernel}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionSeparationKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2345.1 and Definition 2345.2 seed the separated uniform-completion interface over MetricCompletion$^{\uparrow}$, UniformCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, CauchyFilter$^{\uparrow}$, CauchyNet$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\text{UniformCompletionSeparationKernel}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.UniformCompletionSeparationKernelUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, open-cover compactness, or Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, uniform-completion handoff, separated-kernel exactness, filter and net branch visibility, seal nonescape, transport, replay, provenance, and local naming.

Constructive story: A UniformCompletionSeparationKernel$^{\uparrow}$ packet is generated from MetricCompletionUp, UniformCompletionUp, SeparatedMetricUp, CauchyFilterUp, CauchyNetUp, RegSeqRatUp or RealUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2346

A Concrete Naming Certificate for $\mathsf{CauchyDedekindBridge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyDedekindBridge}^{\uparrow}$ packet records the finite bridge surface by which regular Cauchy names can be compared against located-cut-facing data before any completion principle for $\mathsf{Real}^{\uparrow}$ is available. It sits between the regular rational name route of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the located-cut comparison boundary of chapter 404, the Bishop real comparison route of chapter 1381, and the terminal real-name seal of chapter 38. The chapter gives Phase Real and Completion consumers one named handoff between Cauchy-style and cut-facing reads without asserting quotient real equality or ambient completeness.

Definition 2346.1 (Cauchy-Dedekind bridge carrier)**.** *A* $\mathsf{CauchyDedekindBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational Cauchy name before any located-cut-facing comparison is read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to expose only the finite source and comparison observations requested by the packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the endpoint scales, error budgets, and refinement requests consumed by the bridge.*
4. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *comparison row. It records lower-side and upper-side located evidence only through the displayed dyadic and stream windows.*
5. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *classification row, comparing the regular Cauchy source with the located-cut-facing row through finite rational brackets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, W, D, L, B have been consumed.*
7. *H records componentwise* hsame *transport for the regular source, stream windows, dyadic tolerances, located comparison, Bishop comparison, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the bridge route*

$$R, W, D \longmapsto L, B \longmapsto E, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyDedekindBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDedekindBridge}^{\uparrow}}$ *row.*

The classifier for accepted packets reads only these rows. It has no coordinate for quotient real equality, a Dedekind-cut construction as primitive, countable choice, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDedekindBridgeUp.CauchyDedekindBridgeTasteGate single carrier alignment`

Theorem 2346.2 (Cauchy-Dedekind bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDedekindBridge}^{\uparrow}$ *carrier* $K = (R, W, D, L, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyDedekindBridge}^{\uparrow}}$ *surface consists exactly of carrier admission, regular-source exposure, finite-window exposure, dyadic-tolerance exposure, located-cut comparison exposure, Bishop comparison classification, real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of quotient real equality, Dedekind-cut construction as primitive, countable choice, trichotomy, selected representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2346.1. The accepted packet exposes exactly the regular rational source row R, window schedule W, dyadic tolerance row D, located-cut comparison row L, Bishop comparison row B, terminal real seal E, and structural rows H, C, P, N.

The comparison rows are read in order. A consumer first reads R, W, D as regular Cauchy evidence at finite precision, then reads L as located-cut-facing evidence and B as the finite Bishop comparison classification. None of those rows contains a host quotient of real names or a selected cut representative.

The handoff to $\mathsf{Real}^{\uparrow}$ is therefore delayed until the displayed comparison route has been consumed. Rows H, C, P, N transport, replay, source, and name the same finite route, so the excluded principles have no carrier coordinate from which they could be recovered. □

Theorem 2346.3 (Cauchy-Dedekind bridge real-comparison handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* BishopRealCompariso *facing consumer of an accepted* $\mathsf{CauchyDedekindBridge}^{\uparrow}$ *packet factors through the displayed route*

$\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, Nam

In particular, a consumer comparing a regular Cauchy name with located-cut-facing data must read the finite window schedule and dyadic tolerance ledger before the comparison row and before the terminal real seal.

Proof. Begin with an accepted carrier $K = (R, W, D, L, B, E, H, C, P, N)$. The source row R is visible only through the window schedule W and tolerance ledger D, so the consumer has no unscheduled access to a regular Cauchy stream.

The located-cut-facing read is then supplied by L and classified by B. These two rows compare only the displayed rational brackets and located evidence made available by R, W, D.

Finally the real seal E is reached through the same route and preserved by H, C, P, N. Thus the handoff is a finite bridge from regular Cauchy data to located comparison data, not a quotient real equality theorem, a primitive Dedekind-cut construction, a countable-choice selector, trichotomy, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyDedekindBridge}^{\uparrow}$ consumer comparing a regular Cauchy name with located-cut-facing data factors through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$,

LocatedCut$^{\uparrow}$, BishopRealComparison$^{\uparrow}$, the terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, locatedcut, bishoprealcomparison, real: CauchyDedekindBridge$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regular Cauchy source, finite windows, dyadic tolerances, located-cut comparison, Bishop comparison classification, real seal, transport, replay, provenance, and local NameCert rows in one bridge packet.

Closure status for NameCert$_{\text{CauchyDedekindBridge}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchyDedekindBridge}^{\uparrow}}$, seed)

Scope closed: Definition 2346.1, Definition 2346.2, and Definition 2346.3 seed the finite bridge surface between regular Cauchy names and located-cut-facing comparison rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyDedekindBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, Dedekind-cut construction as primitive, countable choice, trichotomy, selected representatives, Lean verification, bridge checking, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, regular-source exposure, finite-window exposure, dyadic-tolerance exactness, located-cut comparison exposure, Bishop comparison classification, real-seal non-escape, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyDedekindBridgeUp.

Constructive story: A CauchyDedekindBridge$^{\uparrow}$ packet is generated from a RegSeqRat source, StreamName finite windows, DyadicRatCore tolerances, LocatedCut comparison evidence, BishopRealComparison classification, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2347

A Concrete Naming Certificate for $\mathsf{CaratheodoryConvexHull}^{\uparrow}$

Following Caratheodory and standard finite-dimensional convex geometry, the $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ packet records finite convex-hull certificates as BEDC naming data. It sits over the ambient affine carrier of chapter 206, the convex membership surface of chapter 208, the finite face and witness rows of chapter 209, and the finite subfamily handoff of chapter 2309. Before the support rows are reduced, $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ consumes the finite-subfamily ledger supplied by chapter 2309 as the displayed boundary for the bounded convex-family read. The packet names only finite support rows, coefficient ledgers, hull-membership witnesses, boundary faces, transport, replay, provenance, and local naming; it does not assert a host separation theorem, a selected minimal simplex, quotient convex-hull equality, or ambient real-vector completeness.

Definition 2347.1 (Caratheodory convex-hull carrier)**.** *A* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, C, S, W, F, L, B, Y, H, T, P, N)$$

with the following displayed rows:

1. *A is the finite-dimensional* $\mathsf{AffineSpace}^{\uparrow}$ *ambient row through which every point and affine combination is read;*
2. *C is the* $\mathsf{ConvexSet}^{\uparrow}$ *source row for the displayed finite generator family;*
3. *S is a finite support row, listing the selected generator histories and exposing the support bound used by the Caratheodory read;*
4. *W is the coefficient ledger, recording nonnegative scalar rows, the finite-sum-one row, and the displayed affine-combination replay;*
5. *F is the* $\mathsf{Polytope}^{\uparrow}$ *face-boundary row that stores the simplex, face, or exposed-hull witness used by the finite support;*
6. *L is the local hull classifier, accepting exactly those point rows that are replayed from S, W, F inside the ambient affine row A and the convex source row C;*
7. *B is the Helly handoff row naming the finite subfamily boundary of chapter 2309;*
8. *Y is the exported hull-membership row, available only after A, C, S, W, F, L, B have been displayed;*

9. *H is the componentwise* hsame *transport for ambient, convex, support, coefficient, face, Helly, and membership rows;*
10. *T is the displayed* Cont *route that replays the finite affine-combination certificate without adding host convexity data;*
11. *P is the* Pkg *provenance over* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{HellyTheorem}^{\uparrow}$, BHist, hsame, Cont, *and local* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryConvexHull}^{\uparrow}}$ *row.*

The carrier exposes no ambient convex-hull operator, no quotient equality of hulls, no selected minimal support, no host dimension theorem, and no standard bridge row.

Lean4 ✓ `BEDC.Derived.CaratheodoryConvexHullUp.CaratheodoryConvexHullTasteGate single carrier alignment`

Theorem 2347.2 (Caratheodory convex-hull NameCert obligations)**.** *For the carrier of Definition 2347.1, the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryConvexHull}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient affine row, convex source row, finite support row, coefficient ledger, polytope face-boundary row, local hull classifier, Helly handoff row, exported membership row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Classifier stability is tested only by replaying the same finite support, coefficient ledger, face boundary, and Helly handoff through transported ambient and convex rows. Ledger exactness says that every accepted membership read exposes precisely $A, C, S, W, F, L, B, Y, H, T, P, N$ and no external convex-hull datum.*

Theorem 2347.3 (Caratheodory convex-hull Helly handoff)**.** *Every accepted* $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *packet routes a finite convex-hull membership request through*

$$\mathsf{AffineSpace}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{Polytope}^{\uparrow},\ \mathsf{HellyTheorem}^{\uparrow}.$$

The affine and convex rows fix the ambient source, the finite support and coefficient ledger display the affine combination, the polytope face row records the boundary witness, and the Helly row records the finite subfamily boundary consumed by downstream convexity arguments. A consumer that omits any of those rows is outside the accepted $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ *carrier.*

Closure status for $\mathsf{NameCert}_{\mathsf{CaratheodoryConvexHull}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CaratheodoryConvexHull}^{\uparrow}$ packet is generated from finite-dimensional affine ambient data, a convex source row, a finite support row, nonnegative coefficient and finite-sum-one ledgers, a polytope face-boundary witness, a local hull classifier, a Helly handoff row, an exported membership row, componentwise transport, continuation replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CaratheodoryConvexHull}^{\uparrow}}$, seed)

Scope closed: Definition 2347.1, Definition 2347.2, and Definition 2347.3 bind the finite support, coefficient, polytope-face, Helly-handoff, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CaratheodoryConvexHull}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host convex-hull equality, minimal-support selection, ambient dimension theorems, Real completeness, separation, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, support replay, coefficient exactness, face-boundary routing, Helly handoff, transport, provenance, and local naming for BEDC.Derived.CaratheodoryConvexHullUp.

Chapter 2348

A Concrete Naming Certificate for $\mathsf{RealApartnessSign}^{\uparrow}$

Bishop–Bridges constructive apartness/sign information reads the sign of a real-name presentation only through positive finite evidence: a negative side, a positive side, or a displayed zero-adjacent refusal to claim either side at the requested precision. The $\mathsf{RealApartnessSign}^{\uparrow}$ packet names that finite sign-read surface between the apartness interface of chapter 304, the separated completion boundary of chapter 1567, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It supplies Real and Completion consumers with a concrete NameCert dependency for located sign comparison; it is not trichotomy, host real order, quotient equality of Cauchy streams, a selected limit, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2348.1 (RealApartnessSign carrier)**.** *A* $\mathsf{RealApartnessSign}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, A, W, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the inspected real-name row, read either as a terminal* $\mathsf{Real}^{\uparrow}$ *seal or as its* $\mathsf{RegSeqRat}^{\uparrow}$ *source before the seal handoff.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *evidence row comparing X with the displayed zero real-name boundary.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window opened for the sign request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback visible on that window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and endpoint ledger used to compare the readback with zero at the requested precision.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after A, W, R, D have been displayed.*
7. *L records the located comparison row: negative-side evidence, positive-side evidence, or a bounded zero-adjacent refusal at the current dyadic tolerance.*
8. *H records componentwise* hsame *transport for the inspected real row, apartness row, window row, readback row, dyadic ledger, seal row, located comparison row, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from the sign request through apartness inspection, finite window readback, dyadic comparison, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealApartnessSign}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{RealApartnessCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessSign}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, total order, quotient stream equality, host real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealApartnessSignUp.TasteGate.RealApartnessSignTasteGate single carrier alignment`

Theorem 2348.2 (RealApartnessSign obligation surface)**.** *For every accepted* $\mathsf{RealApartnessSign}^{\uparrow}$ *carrier* $S = (X, A, W, R, D, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealApartnessSign}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the inspected real row, apartness row, finite window, regular readback, dyadic ledger, real seal, located comparison row, transport, replay, provenance, and local naming rows;*
2. *apartness anchoring, requiring the sign read to inspect X against the zero boundary through the displayed* $\mathsf{ApartnessReal}^{\uparrow}$ *row A before a positive or negative side is exposed;*
3. *finite-window readback, requiring every sign request to pass through W, R, and D before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E can be consumed;*
4. *located comparison exactness, requiring L to record exactly the displayed negative-side evidence, positive-side evidence, or zero-adjacent refusal available at the current dyadic tolerance;*
5. *componentwise stability, requiring H and C to transport and replay only the displayed rows $X, A, W, R, D, E, L, P, N$;*
6. *ledger exhaustion, requiring the packet to export no trichotomy decision, total order, quotient stream equality, host real equality, selected representative, countable-choice schedule, or ambient completeness principle.*

No local sign obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2348.1. The accepted tuple displays exactly the inspected real row, apartness row, finite window, regular rational readback, dyadic ledger, terminal seal, located comparison row, and structural support.

Apartness anchoring is forced by the position of A. A positive-side or negative-side read cannot be named before the comparison with the zero boundary is visible as $\mathsf{ApartnessReal}^{\uparrow}$ evidence.

The finite-window readback discipline is the role of W, R, D, E. The sign request opens a finite StreamName window, reads it as a regular rational row, compares it through the dyadic ledger, and only then exposes the Real seal.

Located comparison exactness follows from L. The row records the finite sign information available at the requested tolerance, including the zero-adjacent refusal case when neither side has been witnessed.

Finally H, C, P, N transport, replay, package, and name only the displayed packet. Since the carrier contains no further coordinate, it cannot export trichotomy, total order, quotient equality, host equality, choice, or ambient completeness. □

Theorem 2348.3 (RealApartnessSign located comparison route)**.** *Every located comparison read exported by an accepted* $\mathsf{RealApartnessSign}^{\uparrow}$ *packet factors through the finite route*

$$X, A, W, R, D \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real or Completion consumer can read sign information only after the apartness-to-zero row, finite StreamName window, regular rational readback, and dyadic tolerance ledger have all been displayed. The route may expose positive-side evidence, negative-side evidence, or a zero-adjacent refusal at the current tolerance, but it exports no total sign decision for arbitrary real names.

Proof. Start with the accepted carrier of Definition 2348.1. Projection gives the single displayed route $X, A, W, R, D, E, L, H, C, P, N$.

Read the apartness row first. The sign request is anchored by A, so any positive or negative side must be evidence against the zero boundary rather than a host-level order claim.

Next replay the finite observation path through W, R, D. The StreamName window, RegSeqRat readback, and DyadicRatCore tolerance ledger determine which finite comparison row is available.

The located comparison row L is then consumed before the terminal seal E and the structural rows H, C, P, N. Since L has only the three displayed alternatives, the consumer receives finite sign information without a global trichotomy principle. □

Theorem 2348.4 (RealApartnessSign real seal boundary)**.** *For an accepted* $\mathsf{RealApartnessSign}^{\uparrow}$ *carrier* $S = (X, A, W, R, D, E, L, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *is available to downstream Real and Completion consumers only through the same sign-read route that displays* $A, W, R, D, L, H, C, P, N$ *In particular, the seal cannot be read as a sign decision detached from apartness evidence, finite windows, regular readback, or the dyadic tolerance ledger.*

Proof. Apply Definition 2348.2. The finite-window readback and located comparison obligations place W, R, D, L before any consumer can use the seal row.

Apply Definition 2348.3. The displayed route already forces apartness anchoring through A and then passes through the finite observation and dyadic comparison rows before reaching the seal.

Finally use ledger exhaustion from the obligation surface. The carrier has no coordinate for a detached sign oracle, host order, quotient stream equality, selected representative, countable-choice schedule, or ambient completeness theorem. The Real seal is therefore read only as the end of the displayed sign packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessSign}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApartnessSign}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2348.1, Definition 2348.2, Definition 2348.3, and Definition 2348.4 seed the finite constructive sign-read surface over ApartnessReal, RealApartnessCompletion, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApartnessSign}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, total order, quotient stream equality, host real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, apartness anchoring, finite-window readback, located comparison exactness, componentwise stability, seal boundary, and ledger exhaustion proofs for BEDC.Derived.RealApartnessSignUp.

Constructive story: A $\mathsf{RealApartnessSign}^{\uparrow}$ packet is generated from an inspected RealUp row, an ApartnessReal comparison with zero, a StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, located comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{ArensFortSpace}^{\uparrow}$ packet is read through the topology, observable two-point, metric-generated topology, and finite stream-window sibling surfaces of chapter 88, chapter 1860, chapter 1655, and chapter 126.

Chapter 2349

A Concrete Naming Certificate for $\mathsf{ArensFortSpace}^{\uparrow}$

Following Arens and Fort, the $\mathsf{ArensFortSpace}^{\uparrow}$ packet records the classical grid space with one non-isolated accumulation point as finite BEDC topology data. It sits beside the indexed-open carrier of chapter 88, the two-point observable handoff of chapter 1860, the metric-generated topology boundary of chapter 1655, and the finite observation windows of chapter 126. The packet is a topology-generated boundary object: it names neighbourhood filters and observable truth reads without importing a metric presentation, a host powerset of opens, choice, or a completed ambient space.

Definition 2349.1 (Arens–Fort space carrier)**.** *An* $\mathsf{ArensFortSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathsf{af}} = (G, S, T, O, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite grid-point ledger. It records the distinguished point ∞ and the finitely inspected isolated points (m, n), together with the local classifier that separates ∞ from the displayed grid rows.*
2. *S is the special-point neighbourhood filter ledger. A row of S names a finite stem and a tail threshold for each displayed column, so a neighbourhood of ∞ contains all but finitely many points in every named column and contains ∞ itself.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open carrier generated by the isolated singleton rows of G and the special-point neighbourhood rows of S.*
4. *O is the observable truth handoff into* $\mathsf{SierpinskiSpace}^{\uparrow}$*: an open-membership read is exposed only as a carried observable truth row after the relevant isolated singleton or special-neighbourhood witness has been displayed.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect finitely many grid coordinates, special-neighbourhood thresholds, and observable truth requests.*
6. *H records componentwise* hsame *transport for grid rows, special-neighbourhood filters, indexed opens, observable truth rows, finite windows, replay rows, provenance, and the local certificate.*
7. *C records displayed* Cont *replay from a public open-membership request through W, then through G or S, then through T, and finally through the observable handoff O.*

8. *P is the* Pkg *provenance over* $\mathsf{ArensFortSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ArensFortSpace}^{\uparrow}}$ *row.*

The classifier admits packets only through these nine rows. It has no coordinate for a host powerset of grid points, a metric ball basis, a selected convergent sequence, an ambient compactness or completeness witness, or a choice principle.

Theorem 2349.2 (Arens–Fort NameCert obligations)**.** *For every accepted* $\mathsf{ArensFortSpace}^{\uparrow}$ *carrier* $A_{\mathsf{af}} = (G, S, T, O, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensFortSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed grid ledger, special-point neighbourhood filter, indexed-open carrier, observable truth handoff, finite windows, transports, continuations, provenance, and local naming row;*

2. *isolated-point stability, requiring each displayed grid point (m, n) to remain open by its singleton row under the componentwise* hsame *transport and displayed* Cont *replay;*

3. *special-point neighbourhood exactness, requiring every open read at ∞ to exhibit the finite stem and column thresholds named by S before the read is admitted by T;*

4. *topology-open transport, requiring the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row to move only through the isolated singleton constructors and special-neighbourhood constructors named in G and S;*

5. *observable-open handoff, requiring every* $\mathsf{SierpinskiSpace}^{\uparrow}$ *truth read to factor through the same open-membership witness already carried by T and inspected by W;*

6. *ledger non-escape, excluding host powersets, metric-generated ball bases, selected convergent sequences, ambient compactness, ambient completeness, choice, and any open classifier outside $G, S, T, O, W, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.ArensFortSpaceUp.ArensFortSpaceCarrier semantic name certificate`

Proof. Carrier admission is projection from Definition 2349.1. The packet exposes only the grid ledger G, the special-neighbourhood filter S, the indexed topology row T, the observable handoff O, the finite windows W, and the structural rows H, C, P, N.

Isolated-point stability is read directly from the singleton rows of G. Transport along H preserves the displayed grid coordinate, and replay along C keeps the singleton-open request attached to the same coordinate before T admits it as an open.

For the special point, the only neighbourhood source is S. Thus every open read at ∞ must present the displayed finite stem and column-threshold data before the topology row T can expose the neighbourhood as open.

Topology-open transport follows by constructor analysis on T. Its generators are exactly the isolated singleton rows and the special-neighbourhood rows, and the replay route never introduces a metric-ball basis or a host powerset of points.

The observable handoff is forced by O and W. A $\mathsf{SierpinskiSpace}^{\uparrow}$ truth value is readable only after the same finite window and the same carried open-membership witness have been displayed.

Finally, the non-escape clause is an inventory check on the carrier. Since metric balls, selected sequences, compactness, completeness, choice, host powersets, and detached open classifiers are not coordinates of A_{af}, no local NameCert obligation can export them. □

Falsifiable prediction: An accepted $\mathsf{ArensFortSpace}^{\uparrow}$ consumer that reads a $\mathsf{SierpinskiSpace}^{\uparrow}$ observable truth value at the special point without first displaying a finite $\mathsf{StreamName}^{\uparrow}$ window, the corresponding special-neighbourhood threshold row, and the $\mathsf{Topology}^{\uparrow}$ open-membership witness refutes the carrier surface.

Independence witness: topology, sierpinski-space, metric-topology, streamname: none is bijective to $\mathsf{ArensFortSpace}^{\uparrow}$ because this packet jointly carries isolated grid rows, special-point neighbourhood filters, indexed topology opens, observable truth handoff, finite stream windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensFortSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensFortSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2349.1 and Definition 2349.2 seed the finite Arens–Fort topology certificate over isolated grid rows, special-point neighbourhood filters, indexed-open rows, observable truth handoff, finite windows, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensFortSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim metrizability, compactness, completeness, choice, host powersets, selected convergent sequences, ambient topology equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, isolated-point stability, special-point neighbourhood exactness, topology-open transport, observable-open handoff, and ledger non-escape for BEDC.Derived.ArensFortSpaceUp.

Constructive story: An $\mathsf{ArensFortSpace}^{\uparrow}$ packet is generated from isolated grid-point rows, special-point neighbourhood filter rows, a TopologyUp indexed-open carrier, a SierpinskiSpaceUp observable truth handoff, finite StreamNameUp windows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2350

A Concrete Naming Certificate for $\mathsf{CauchyContractionOrbit}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, a $\mathsf{CauchyContractionOrbit}^{\uparrow}$ packet names the finite Picard-orbit ledger used by constructive contraction fixed-point routes before an endpoint is sealed. It sits between the map-level contraction packet of chapter 1418, the one-step Picard route of chapter 395, the modulus surface of chapter 292, the completion consumer of chapter 128, the regular rational sequence handoff of chapter 301, and the final real-number seal of chapter 38. The packet records the finite orbit and ratio data that make a Cauchy-modulus route visible without selecting a fixed point.

Definition 2350.1 (CauchyContractionOrbit carrier)**.** *A* $\mathsf{CauchyContractionOrbit}^{\uparrow}$ *carrier is a finite packet*

$$O = (X, T, I, L, M, R, E, H, C, P, N)$$

with the following displayed rows.

1. *X is the metric carrier and distance-read row inherited from the contraction-map surface.*
2. *T is the self-map graph row whose accepted applications generate the displayed Picard iterates.*
3. *I is the finite Picard-iterate ledger, recording the initial point and the visible rows $x_0, x_1, \ldots, x_n$ produced by repeated graph reads of T.*
4. *L is the strict rational contraction ledger, storing a nonnegative bound below one and the adjacent-distance comparisons attached to the iterate ledger.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *route extracted from the finite orbit ledger and the strict ratio ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row used when the orbit is read as a regular rational sequence approximation.*
7. *E is the* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row, exposing where a completion argument may later consume the orbit without being supplied by it.*
8. *H is componentwise* hsame *transport for the metric carrier, self-map graph, iterate ledger, ratio ledger, modulus route, regular-sequence handoff, completion consumer, replay, provenance, and local naming rows.*
9. *C is the displayed* Cont *replay from contraction-map admission through Picard iteration, ratio comparison, Cauchy-modulus handoff, regular-sequence handoff, and completion-facing use.*

10. *P is the* Pkg *provenance over* ContractionMapping$^{\uparrow}$, PicardContraction$^{\uparrow}$, CauchyModulus$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyContractionOrbit}^{\uparrow}}$ *row.*

The carrier exposes a finite orbit-to-modulus surface. It contains no coordinate for a selected fixed point, host limit, quotient stream, ambient completeness proof, countable-choice principle, or Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyContractionOrbitUp.CauchyContractionOrbitTasteGate single carrier alignment`

Theorem 2350.2 (CauchyContractionOrbit NameCert obligations)**.** *For an accepted* CauchyContractionOrbit$^{\uparrow}$ *carrier $O = (X, T, I, L, M, R, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyContractionOrbit}^{\uparrow}}$ *surface consists of carrier admission, contraction-map visibility, finite Picard-iterate ledger visibility, strict ratio-ledger stability, Cauchy-modulus exposure,* RegSeqRat$^{\uparrow}$ *and* CompleteMetric$^{\uparrow}$ *handoff visibility,* Real$^{\uparrow}$ *seal separation, componentwise transport, continuation replay, provenance, and local naming for the finite packet of Definition 2350.1.*

Proof. Carrier admission projects the displayed tuple. The metric row X, graph row T, finite Picard ledger I, and strict ratio ledger L give the contraction-map and iterate data read by the certificate.

The Cauchy-modulus row M is exposed only after those finite rows are visible. The handoff rows R and E record where regular rational sequence and complete-metric consumers may later read the orbit, while the Real$^{\uparrow}$ entry in the provenance row remains a seal boundary rather than an endpoint proof.

Transport, replay, provenance, and local naming are componentwise. The rows H, C, P, N preserve exactly the carrier coordinates listed above, so the certificate exports the orbit-to-modulus surface without adding host limits, quotient streams, countable choice, or real completeness. □

Theorem 2350.3 (CauchyContractionOrbit Cauchy-modulus handoff)**.** *Every accepted* CauchyContractionOrbit$^{\uparrow}$ *carrier exposes a* CauchyModulus$^{\uparrow}$ *route before any* Real$^{\uparrow}$ *seal is read. The finite orbit ledger I and strict ratio ledger L determine the modulus row M, and every regular-sequence or complete-metric consumer must pass through*

$$X,\ T,\ I,\ L,\ M,\ R,\ E,\ H,\ C,\ P,\ N.$$

Thus the packet supplies a Cauchy-modulus handoff for the constructive contraction route, not a selected fixed point, host limit, quotient stream, ambient completeness theorem, countable-choice construction, or Real$^{\uparrow}$ *completeness proof.*

Proof. Read the finite Picard ledger I. It contains only displayed iterate rows generated from the self-map graph T, so each downstream distance comparison is anchored in the same metric carrier X.

Read the strict ratio ledger L. Its below-one rational bound and adjacent-distance comparisons are the data used by the modulus row M; no infinite stream or host limit is consulted.

The rows R and E are therefore downstream handoff rows. They may consume the already exposed modulus surface for regular rational sequence or complete-metric purposes, but neither row turns the packet into a real-number endpoint or a fixed-point selector.

Finally H, C, P, N transport and replay the same finite route. Since the Real$^{\uparrow}$ dependency appears only in the provenance and seal boundary, the Cauchy-modulus route is available before any real-completeness seal is read. □

Falsifiable prediction: Every accepted $\mathsf{CauchyContractionOrbit}^{\uparrow}$ read factors through a contraction-map row, finite Picard-iterate ledger, strict ratio ledger, CauchyModulus row, RegSeqRat handoff, CompleteMetric consumer row, hsame transport, Cont replay, Pkg provenance, and NameCert row before any RealUp endpoint seal.

Independence witness: contractionmapping, picardcontraction, cauchymodulus, completemetric, regseqrat, real: none is bijective with $\mathsf{CauchyContractionOrbit}^{\uparrow}$ because the packet jointly stores the map graph, finite orbit ledger, strict ratio ledger, modulus extraction row, regular-sequence handoff, completion-facing consumer row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContractionOrbit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContractionOrbit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2350.1, Definition 2350.2, and Definition 2350.3 seed the finite contraction-orbit packet that exposes a Cauchy-modulus handoff before endpoint sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContractionOrbit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected fixed points, host limits, quotient streams, ambient completeness, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-map visibility, Picard-ledger visibility, strict ratio stability, Cauchy-modulus exposure, RegSeqRat and CompleteMetric handoff visibility, Real seal separation, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{CauchyContractionOrbit}^{\uparrow}$ packet is generated from a metric carrier, self-map graph, finite Picard ledger, strict ratio ledger, CauchyModulus route, RegSeqRat handoff, CompleteMetric consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 2351

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ packet records regulated-integral values through displayed Cauchy windows. It reads the regulated-integral surface of chapter 1557, the Riemann–Stieltjes sibling route of chapter 1637, the dyadic step rows of chapter 428, the regular rational readback of chapter 301, the finite stream windows of chapter 126, and the terminal real-name boundary of chapter 38. The packet names integral values by finite Cauchy-read obligations, not by host function equality, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2351.1 (Regulated Cauchy integral carrier)**.** *A* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, T, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the regulated integrand and regulated-integral row, read through* $\mathsf{RegulatedIntegral}^{\uparrow}$ *and carrying only the displayed finite integration data.*
2. *T is the tagged partition row, including the tags, cells, endpoint reads, and the sibling compatibility surface with* $\mathsf{RiemannStieltjes}^{\uparrow}$ *when an integrator row is present.*
3. *A is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximation ledger used to compute the finite integral sums on the displayed partition.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *Cauchy window schedule listing the precision windows actually requested by a consumer of the integral value.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite sums in A through the requested windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after G, T, A, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the integrand, partition, step approximation, window schedule, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from regulated-integral reads to tagged partitions, from tagged partitions to dyadic step sums, from step sums to Cauchy windows, from windows to* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and from readback to the* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, T, A, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, selected limiting partitions, countable choice, or an ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.RegulatedCauchyIntegralUp.TasteGate.RegulatedCauchyIntegralTasteGate`
`single carrier alignment`

Theorem 2351.2 (Regulated Cauchy integral NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *carrier $K = (G, T, A, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-integral source exposure through G, tagged-partition exposure through T, dyadic step-sum exposure through A, Cauchy-window scheduling through W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, provenance through P, local naming through N, and non-escape from host function equality, quotient streams, selected limiting partitions, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2351.1. The accepted packet exposes exactly the regulated-integral source, tagged partition, dyadic step approximation, Cauchy window schedule, regular readback, real seal, and structural rows.

The integration read is finite. The row G supplies the regulated-integral source, T supplies the tagged partition data, and A supplies the dyadic step sums visible at the requested tolerances.

The Cauchy read is downstream. The row W records the requested windows, R records the regular rational readback from those windows, and E is the only Real-facing seal. Since the excluded principles have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2351.3 (Regulated Cauchy integral window readback exactness)**.** *Let $K = (G, T, A, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *packet. Every displayed Cauchy window in W reads the integral value through the same dyadic step approximation ledger A over the same tagged partition T and determines the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row R. The readback is exact at the packet boundary: no consumer can replace the displayed window route by quotient stream equality, a host limiting partition, countable choice, or host real equality.*

Lean4 ✓ `BEDC.Derived.RegulatedCauchyIntegralUp.RegulatedCauchyIntegralWindowReadbackExactness`

Proof. Start with a displayed window in W. The carrier gives no detached stream coordinate, so the window can be read only through the continuation route from G to T and from T to A.

The dyadic step ledger fixes the finite sum read. Componentwise transport through H and replay through C carry the same tagged cells, values, and tolerance rows into the regular rational readback coordinate R.

Thus every accepted window reaches the same R by the displayed route. The absent quotient, host-limit, choice, and host-equality coordinates cannot alter the readback at the packet boundary. □

Theorem 2351.4 (Regulated Cauchy integral Real seal non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *carrier is available only as the endpoint of the route*

$$\mathsf{RegulatedIntegral}^{\uparrow},\ \mathsf{RiemannStieltjes}^{\uparrow},\ \mathsf{DyadicStepFunction}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, a Real-facing consumer must first read the regulated-integral source G, tagged partition T, dyadic step approximation A, Cauchy window schedule W, and regular readback R before it can read E. The packet exports no host function equality, quotient stream, countable-choice selector, selected limiting partition, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegulatedCauchyIntegralUp.RegulatedCauchyIntegralCarrier real seal nonescape`

Proof. Project the seal coordinate of Definition 2351.1. The row E is downstream from G, T, A, W, R and is transported and replayed only by H and C.

Apply Definition 2351.2. It identifies the regulated-integral, partition, dyadic step, window, readback, seal, transport, replay, provenance, and naming rows as the complete local certificate surface.

Apply Definition 2351.3. The regular readback row is fixed by the displayed Cauchy windows over the same finite integration ledger. Therefore the Real seal cannot be reached through any omitted host or completion coordinate. □

Falsifiable prediction: Every accepted $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ value read factors through a displayed $\mathsf{RegulatedIntegral}^{\uparrow}$ source, tagged partition, $\mathsf{DyadicStepFunction}^{\uparrow}$ approximation, $\mathsf{StreamName}^{\uparrow}$ Cauchy window schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regulatedintegral, riemannstieltjes, dyadicstepfunction, streamname, regseqrat, real: $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ is not bijective to any listed sibling because it binds regulated integral source data, tagged partitions, dyadic step approximations, Cauchy windows, regular rational readback, Real sealing, transport, replay, provenance, and local NameCert rows into one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ packet is generated from a regulated-integral source row, tagged partition row, dyadic step approximation ledger, Cauchy window schedule, RegSeqRat readback, Real seal, componentwise transport, displayed continuation replay, package provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedCauchyIntegral}^{\uparrow}}$, seed)

Scope closed: Definition 2351.1, Definition 2351.2, Definition 2351.3, and Definition 2351.4 seed the regulated-integral Cauchy-read boundary over tagged partitions, dyadic step approximations, StreamName windows, RegSeqRat readback, and Real seal data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedCauchyIntegral}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient stream completion, selected limiting partitions, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-integral source exposure, tagged-partition exposure, dyadic step-sum exposure, Cauchy-window readback exactness, Real-seal non-escape, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.RegulatedCauchyIntegralUp.

Chapter 2352

A Concrete Naming Certificate for $\mathsf{SpectralRadius}^{\uparrow}$

Following Gelfand and standard Banach-algebra spectral theory, the $\mathsf{SpectralRadius}^{\uparrow}$ packet records the finite BEDC surface by which a displayed Banach-algebra element exposes operator-power norm-growth rows and resolvent-test rows before any spectrum, maximal ideal space, or host algebra quotient is available. It is a textbook neighbour of the operator-algebra frontier: it reads the complete normed algebra carrier of chapter 2189, the bounded-operator route of chapter 1644, the public operator-norm carrier of chapter 3603 for $\mathsf{OperatorNorm}^{\uparrow}$, the stability obligation of Definition 94.3, and the compact-operator dependency of chapter 1952.

Definition 2352.1 (Spectral radius carrier)**.** *A* $\mathsf{SpectralRadius}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, a, W, Q, G, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *row carrying multiplication, norm, completion, product control, transport, replay, provenance, and local naming.*
2. *a is the carried algebra element whose powers are inspected.*
3. *W is the finite operator-power window. For each displayed natural index in the packet it records the product row for a^n and the corresponding* $\mathsf{OperatorNorm}^{\uparrow}$ *or algebra-norm endpoint.*
4. *Q is the norm-growth comparison ledger. It records the rational or* $\mathsf{Real}^{\uparrow}$ *bound rows used to compare the observed power norms with the requested growth threshold.*
5. *G is the Gelfand-style radius readback row. It is only a finite readback from the displayed windows W and comparisons Q; it is not a limit of all powers.*
6. *E is the resolvent-test row. For a displayed scalar-window request it records the candidate inverse-check surface and the refusal row saying that no inverse is exported unless the local Banach-algebra ledger displays the needed multiplication and unit witnesses.*
7. *H records componentwise* hsame *transport for A, a, W, Q, G, E and the structural rows.*
8. *C records displayed* Cont *replay of the route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto W, Q, G, E \longmapsto \mathsf{SpectralRadius}^{\uparrow}.$$

9. *P is the* Pkg *provenance over* $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralRadius}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through the Banach-algebra source, the displayed element, the finite power-window rows, norm-growth comparisons, resolvent-test rows, transport, replay, provenance, and local naming. It has no coordinate for a completed spectrum, the Gelfand transform, maximal ideals, host operator equality, quotient completion, selected approximate eigenvectors, or choice.

Lean4 ✓ `BEDC.Derived.SpectralRadiusUp.SpectralRadiusTasteGate single carrier alignment`

Theorem 2352.2 (Spectral radius NameCert obligations)**.** *For an accepted* $\mathsf{SpectralRadius}^{\uparrow}$ *carrier*

$$S = (A, a, W, Q, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralRadius}^{\uparrow}}$ *surface consists exactly of carrier admission, Banach-algebra source compatibility, finite power-window stability, norm-growth readback, resolvent-test boundary, classifier transport, continuation replay, provenance, and local naming. A downstream operator-algebra consumer may inspect the displayed power norms and the displayed resolvent-test outcomes, but it receives no completed spectrum, no spectral theorem, no compact-operator eigenvalue list, no maximal ideal space, and no host equality of operators.*

Proof. Project the carrier of Definition 2352.1. The first two rows fix the Banach-algebra source and the carried element, while W, Q, G, E are the only analytic rows available to a consumer.

Power-window stability is componentwise. Transport through H moves each displayed power product and norm endpoint along the Banach-algebra classifier, and replay through C returns to the same finite window before any radius readback is public.

Norm-growth readback uses Q and G. These rows compare only the displayed power norms already present in W with the requested growth threshold. They do not quantify over all powers or select a host limit.

The resolvent boundary is the row E. It records tests against displayed scalar windows and local inverse-check rows, but it cannot export an inverse or a spectrum unless those rows are already visible in the Banach-algebra ledger.

Finally P and N name only this finite surface. The excluded spectral, representation-theoretic, quotient, choice, and host-equality objects have no coordinate in the packet, so they cannot be recovered by a downstream read. □

Theorem 2352.3 (Spectral radius resolvent boundary)**.** *Every resolvent-facing read of an accepted* $\mathsf{SpectralRadius}^{\uparrow}$ *packet factors through the same displayed Banach-algebra source row A, carried element row a, finite power-window row W, norm-growth ledger Q, radius readback row G, resolvent-test row E, and structural rows H, C, P, N. In particular, a resolvent request cannot replace the finite readback by a completed spectrum, a Gelfand-transform equality, a compact-operator eigenvalue theorem, selected approximate eigenvectors, quotient completion equality, or countable choice.*

Proof. Start with an accepted packet and the resolvent request. Carrier admission exposes E only after the Banach-algebra row A, element row a, power windows W, and norm-growth ledger Q have been displayed.

The radius readback G is therefore already tied to the finite windows. Any scalar test entering E must use the same route through C and the same componentwise transport through H.

The sibling operator rows add no hidden coordinate. The bounded-operator surface supplies action and norm-bound ledgers, the operator-norm obligation supplies the scalar bound row, and the compact-operator chapter supplies only compactness schedules when separately present.

Since the packet has no field for a completed spectrum, Gelfand transform, eigenvector selector, quotient equality, or choice schedule, the resolvent-facing read remains a boundary over the displayed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{SpectralRadius}^{\uparrow}$ resolvent-facing read exposes the Banach-algebra source, element row, finite power-window row, norm-growth ledger, radius readback row, resolvent-test row, componentwise transport, replay, provenance, and local naming before any spectral-radius conclusion is public.

Independence witness: banachalgebra, boundedlinearoperator, operatornorm, compactoperator: BanachAlgebraUp lacks the finite power-window and resolvent-test rows; BoundedLinearOperatorUp lacks the Banach-algebra product-power ledger; OperatorNormUp supplies scalar norm bounds without naming the radius packet; CompactOperatorUp supplies image-net schedules without the resolvent boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralRadius}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralRadius}^{\uparrow}$ packet is generated from a BanachAlgebraUp source, a carried element, finite operator-power norm-growth rows, a radius readback row, a resolvent-test row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralRadius}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2352.1, Definition 2352.2, Definition 2352.3, the falsifiable prediction, and the independence witness seed the finite spectral-radius route over BanachAlgebraUp, BoundedLinearOperatorUp, OperatorNormUp, CompactOperatorUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralRadius}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the spectral theorem, compact-operator eigenvalue decomposition, Gelfand representation, maximal ideal space, completed spectrum, quotient completion, selected approximate eigenvectors, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite power-window stability, norm-growth readback, resolvent-test boundary, classifier transport, and consumer non-escape for BEDC.Derived.SpectralRadiusUp.

Chapter 2353

A Concrete Naming Certificate for $\mathsf{ConnectedSpace}^{\uparrow}$

Following Hausdorff and Bishop–Bridges topology, the $\mathsf{ConnectedSpace}^{\uparrow}$ packet records connectedness as a finite refusal surface for displayed clopen separations over $\mathsf{Topology}^{\uparrow}$. It sits beside the topology carrier of chapter 88, the metric and real consumers of chapter 57 and chapter 38, the continuity route of chapter 59, the circle boundary of chapter 58, and the chain-connected metric comparison of $\mathsf{ChainConnectedMetric}^{\uparrow}$. The certificate names the topological connectedness interface without adding excluded-middle separation, host subset extensionality, open-cover compactness, or a selected global point.

Definition 2353.1 (ConnectedSpace carrier)**.** *A* $\mathsf{ConnectedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, S, F, M, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-set classifier row, including the visible open indices, membership ledger, finite meet rows, union rows, and boundary opens inspected by connectedness consumers.*
2. *O is the clopen candidate row. It records a displayed open row together with the complementary open row that would form a separation if both sides were inhabited in the accepted packet.*
3. *S is the clopen-separation attempt row. It stores the finite attempt to split the carrier into two disjoint nonempty clopen pieces and marks the route as refused unless the attempt collapses to the empty or whole boundary row already present in T.*
4. *F is the* $\mathsf{Continuous}^{\uparrow}$ *pullback row. It records only displayed inverse-image opens and their transported clopen attempts for continuous consumers.*
5. *M is the optional* $\mathsf{Met}^{\uparrow}$ *consumer row, used when a metric-generated topology requests connectedness data through balls and finite open ledgers.*
6. *R is the optional* $\mathsf{Real}^{\uparrow}$ *consumer row, used by real-line and interval routes after the topology and pullback rows have been displayed.*
7. *Q is the* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *comparison row. It records that a metric chain-connected witness may feed the topological refusal surface only through the visible metric, topology, and real rows.*

8. *H records componentwise* hsame *transport for the topology classifier, clopen attempt, pullback, optional metric and real consumers, chain-connected comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a connectedness read through*

$$T, O, S, F, M, R, Q \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the S^1 circle source,* $\mathsf{ChainConnectedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConnectedSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for excluded-middle decomposition of subsets, host subset extensionality, quotient topology equality, open-cover compactness, global choice of a point, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ConnectedSpaceUp.ConnectedSpaceTasteGate single carrier alignment`

Theorem 2353.2 (ConnectedSpace carrier admission)**.** *For every accepted* $\mathsf{ConnectedSpace}^{\uparrow}$ *carrier*

$$K = (T, O, S, F, M, R, Q, H, C, P, N),$$

the local carrier admission row is exhausted by the displayed topology classifier, clopen attempt, continuous pullback, optional metric and real consumers, chain-connected comparison, componentwise transport, replay, provenance, and local naming rows. No connectedness consumer may inspect a clopen separation before those rows are present.

Proof. Project the carrier of Definition 2353.1. The projection exposes exactly the rows $T, O, S, F, M, R, Q, H, C, P,$

The topology row is read first, because every clopen attempt is expressed as a pair of displayed open rows over the same $\mathsf{Topology}^{\uparrow}$ classifier. The optional metric, real, circle, and chain-connected consumers are downstream rows; they do not add a second connectedness carrier.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier definition contains no further coordinate, carrier admission is precisely the displayed finite packet. □

Theorem 2353.3 (ConnectedSpace classifier stability)**.** *Let $K = (T, O, S, F, M, R, Q, H, C, P, N)$ and $K' = (T', O', S', F', M', R', Q', H', C', P', N')$ be accepted* $\mathsf{ConnectedSpace}^{\uparrow}$ *carriers. If the topology classifier rows are* hsame*, the clopen-attempt rows are transported by the displayed component of H, and the consumer rows F, M, R, Q match componentwise, then every connectedness read accepted for K is accepted for K' through the same replay and provenance route.*

Proof. Use the componentwise transport row in H. It carries the topology classifier first, so the open indices, membership ledger, finite meets, unions, and boundary opens remain the same visible topology surface.

Transport the clopen-attempt row next. The attempt remains a displayed open-complement pair over the transported topology classifier, and its refusal status remains attached to the same empty-or-whole boundary alternatives.

The rows F, M, R, Q are transported after the classifier and attempt rows. Thus continuous, metric, real, circle, and chain-connected consumers receive the same finite route, and C, P, N replay, source, and name that route for K'. □

Theorem 2353.4 (ConnectedSpace clopen-separation refusal)**.** *For an accepted* $\mathsf{ConnectedSpace}^{\uparrow}$ *carrier* $K = (T, O, S, F, M, R, Q, H, C, P, N)$*, every clopen separation attempt in* S *is refused unless the displayed clopen row in* O *collapses through* T *to the empty boundary open or to the whole-carrier boundary open. The refusal is a finite ledger row and is not an appeal to excluded middle over arbitrary subsets.*

Proof. Read the attempt row S together with the clopen row O. Both rows are indexed by the open-set classifier T, so the only admissible alternatives are the displayed boundary opens already present in the topology packet.

If the candidate is the empty boundary, the attempted separation has no left inhabited component. If the candidate is the whole-carrier boundary, the complementary side has no displayed inhabited component.

All other attempts remain refused by the local row S. The refusal is stored as finite NameCert data over T, O, H, C, P, N, so it does not decide arbitrary host subsets or import a global separation principle. □

Theorem 2353.5 (ConnectedSpace continuous-pullback preservation)**.** *Let* $K = (T, O, S, F, M, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{ConnectedSpace}^{\uparrow}$ *carrier. Any* $\mathsf{Continuous}^{\uparrow}$ *consumer that pulls a clopen attempt back along a displayed map must read the inverse-image row through* F *and then reuse the same refusal route* T, O, S, H, C, P, N*. Hence the pullback of a refused clopen separation remains refused inside the connectedness packet.*

Proof. Project the pullback row F. It stores only displayed inverse images of open rows from the topology classifier and their transported clopen attempts.

Apply the refusal theorem to the transported attempt. The pullback candidate is still read as a clopen row over T and can pass only if it collapses to the displayed empty or whole boundary row.

The replay and provenance rows do not create a new separation object. They only route the continuous consumer through F and back to T, O, S, H, P, N, so the refused status is preserved. □

Theorem 2353.6 (ConnectedSpace ledger nonescape)**.** *Every* $\mathsf{ConnectedSpace}^{\uparrow}$ *read exported from an accepted carrier*

$$K = (T, O, S, F, M, R, Q, H, C, P, N)$$

factors through $\mathsf{Topology}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, *optional* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers, the* S^1 *circle source,* $\mathsf{ChainConnectedMetric}^{\uparrow}$ *comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. It exports no excluded-middle clopen split, no host subset extensionality, no quotient topology equality, no open-cover compactness, and no selected global point.*

Proof. Combine carrier admission with the classifier-stability and clopen-refusal theorems. These place every accepted read inside the finite rows $T, O, S, F, M, R, Q, H, C, P, N$.

The continuous-pullback theorem handles the only consumer route that can manufacture a new clopen attempt. The attempt is immediately returned to the same refusal ledger.

Since the optional metric, real, circle, and chain-connected rows are consumer handoffs rather than generators of a host separation principle, the listed route exhausts the export surface and none of the refused objects can escape the ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConnectedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConnectedSpace}^{\uparrow}$ packet is generated from a Topology open-set classifier, a displayed clopen-attempt row, a finite refusal ledger, continuous pullback rows, optional Metric and Real consumer rows, a circle-source

handoff, a ChainConnectedMetric comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ConnectedSpace}\uparrow}$, seed)

Scope closed: Definition 2353.1, Definition 2353.2, Definition 2353.3, Definition 2353.4, Definition 2353.5, and Definition 2353.6 seed the finite connectedness refusal surface over topology, continuity, metric, real, circle, and chain-connected consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConnectedSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close excluded-middle separation, host subset extensionality, quotient topology equality, open-cover compactness, selected points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, clopen-separation refusal, continuous-pullback preservation, and ledger nonescape for BEDC.Derived.ConnectedSpaceUp.

Chapter 2354

A Concrete Naming Certificate for $\mathsf{PointwiseContinuityModulus}^{\uparrow}$

Following Bishop–Bridges pointwise continuity on metric spaces, the $\mathsf{PointwiseContinuityModulus}^{\uparrow}$ packet records the local radius surface that a displayed continuous map supplies at a single named source point and target tolerance. It sits between the metric source of chapter 57, the graph and pointwise continuity rows of chapter 130, the rational tolerance rows of chapter 37, the global modulus interface of chapter 2228, and the compact finite-net extraction of chapter 2292. The packet names one point-indexed local radius ledger; it is not a uniform modulus, not a compactness principle, not a selected Lebesgue number, and not host semantic continuity.

Definition 2354.1 (Pointwise continuity modulus carrier)**.** *A* $\mathsf{PointwiseContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Y, F, x, \epsilon, \delta, R, U, K, H, C, Q, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose point classifier and distance comparisons supply the neighbourhood side of the local read.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance comparisons supply the tolerance side.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from X to Y, including the displayed pointwise continuity evidence read at the named point.*
4. *x is the finite source-point name at which the local radius is certified.*
5. *ϵ is the requested* $\mathsf{Rat}^{\uparrow}$ *target tolerance row.*
6. *δ is the offered* $\mathsf{Rat}^{\uparrow}$ *source radius row.*
7. *R is the local radius ledger tying x, ϵ, δ to the pointwise continuity row carried by F.*
8. *U is the optional* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *comparison row, read only when a downstream consumer asks whether the local radius is compatible with a global modulus surface.*
9. *K is the optional* $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$ *extraction row, read only when compact finite-net folding consumes the point-indexed radius surface.*
10. *H records componentwise* hsame *transport for the two metric rows, graph row, point row, tolerance row, radius row, comparison rows, replay, provenance, and local naming.*

11. *C records displayed* Cont *replay from a target-tolerance request through the local radius ledger and onward to any global or compact modulus consumer.*

12. *Q is the* Pkg *provenance over* PointwiseContinuityModulus$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Rat$^{\uparrow}$, UniformContinuityModu CompactUniformContinuityModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* NameCert$_{\text{PointwiseContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, Y, F, x, \epsilon, \delta, R, U, K, H, C, Q, N$. It has no coordinate for a host neighbourhood system, a selected global modulus function, compact open-cover data, sequence extraction, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.PointwiseContinuityModulusUp.PointwiseContinuityModulusTasteGate single carrier alignment`

Theorem 2354.2 (Pointwise continuity modulus NameCert obligations)**.** *For an accepted* PointwiseContinuityModu *carrier $P = (X, Y, F, x, \epsilon, \delta, R, U, K, H, C, Q, N)$, the local* NameCert$_{\text{PointwiseContinuityModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, source-target metric lock through X and Y, graph compatibility through F, point-indexed tolerance visibility through x and ϵ, radius exactness through δ and R, optional global-modulus compatibility through U, optional compact-extraction compatibility through K, structural transport through H, displayed replay through C, provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.PointwiseContinuityModulusUp.PointwiseContinuityModulusCarrier namecert obligations`

Proof. Project the carrier of Definition 2354.1. The only source of a local radius is the displayed tuple x, ϵ, δ, R attached to the continuous-map row F over the two metric rows X and Y.

The ledger R binds the requested target tolerance to the offered source radius at the named point. Thus the packet records a local point-indexed read, not a uniform radius for all source points and not a host-level continuity assertion.

The optional rows U and K are downstream comparison surfaces. A UniformContinuityModulus$^{\uparrow}$ consumer may read U only after the local radius ledger has been displayed, and the compact extraction route of chapter 2292 may read K only as a compact finite-net consumer of the same point-indexed ledger.

Rows H, C, Q, N transport, replay, source, and name the same finite route. Since no host neighbourhood system, selected global modulus function, compact open-cover datum, sequence extraction principle, countable-choice row, or ambient-completeness row occurs in the carrier, the listed obligations exhaust the NameCert surface. □

Theorem 2354.3 (Pointwise radius handoff to uniform extraction)**.** *Every accepted* PointwiseContinuityModulus$^{\uparrow}$ *carrier exports a local-radius read through the finite route*

Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Rat$^{\uparrow}$, UniformContinuityModulus$^{\uparrow}$, CompactUniformContinuityModulus$^{\uparrow}$, BHist, hsame, Co

A downstream uniform-modulus or compact-uniform-modulus consumer receives the point row x, the requested target tolerance ϵ, the offered source radius δ, and the ledger R before it may consult either U or K. The route exports no selected global modulus function, no open-cover compactness, no selected Lebesgue number, no sequence extraction principle, and no ambient complete metric space.

Proof. Start with the obligation surface of Definition 2354.2. It fixes the two metric rows, the continuous-map graph row, the point row, the requested tolerance, the offered radius, and the local ledger as the only pointwise source of modulus information.

Now read a uniform consumer. The row U is available only as a comparison against the already displayed pointwise ledger, so it cannot choose a radius independently of x, ϵ, δ, R.

Finally read a compact extraction consumer. The compact-uniform-continuity route of chapter 2292 consumes pointwise radius rows center by center before folding them into a compact finite-net output. Therefore K is downstream of the same local ledger, and the excluded host compactness and global selection principles have no coordinate in the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseContinuityModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseContinuityModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2354.1, Definition 2354.2, and Definition 2354.3 seed the finite local-radius route over metric rows, ContinuousMap graph rows, Rat tolerance rows, UniformContinuityModulus comparison rows, CompactUniformContinuityModulus extraction rows, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseContinuityModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global uniform modulus, compact finite-net extraction, selected Lebesgue numbers, host continuity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric lock, point-indexed radius exactness, UniformContinuityModulus compatibility, CompactUniformContinuityModulus handoff, structural replay, and ledger non-escape for BEDC.Derived.PointwiseContinuityModulusUp.

Constructive story: A $\mathsf{PointwiseContinuityModulus}^{\uparrow}$ packet is generated from source and target MetricUp rows, a ContinuousMapUp graph row, one named source point, a RatUp target tolerance, a RatUp source radius, a local radius ledger, optional UniformContinuityModulusUp and CompactUniformContinuityModulusUp comparison rows, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2355

A Concrete Naming Certificate for $\mathsf{CompleteSeparatedMetric}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley-style separated metric completion, the $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ packet records the standard landing object whose completion route has both displayed limit existence and zero-distance uniqueness. It sits after the completion surface of chapter 1506, the separated comparison boundary of chapter 1411, the complete-limit witness of chapter 128, the finite Cauchy-name equality boundary of chapter 1412, and the terminal real-name seal of chapter 38. The packet names the constructive complete separated metric object without adding quotient completion classes, selected representatives, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2355.1 (CompleteSeparatedMetric carrier)**.** *A* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, C, S, E, R, H, K, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, including the displayed metric carrier, completion handoff, and finite tolerance readback surface.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-existence row, recording that every accepted regular Cauchy point name exposes a displayed limit, modulus, and convergence-distance ledger.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance identity row, recording that two displayed carried observations with real zero distance are identified by the point classifier.*
4. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *boundary row by which two Cauchy presentations of the same completion-facing point are compared before any limit equality is read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *zero-distance readback row used to consume metric-distance witnesses only through the terminal real-name seal.*
6. *H records componentwise* hsame *transport for the metric source, complete-limit row, separatedness row, Cauchy-equivalence boundary, real readback, replay row, provenance row, and local naming row.*
7. *K records the displayed* Cont *replay from Cauchy observations through completion handoff, limit existence, zero-distance comparison, and separated classifier identity.*

8. *P is the* Pkg *provenance over* MetricCompletion$^{\uparrow}$, CompleteMetric$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, CauchySequenceEquivalenc Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{CompleteSeparatedMetric}^{\uparrow}}$ *row.*

The carrier is the paired complete-and-separated completion surface as finite BEDC data. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of points, or a theorem asserting ambient completeness of Real$^{\uparrow}$.

Theorem 2355.2 (CompleteSeparatedMetric NameCert obligations)**.** *For an accepted* CompleteSeparatedMetric$^{\uparrow}$ *carrier $Q = (X, C, S, E, R, H, K, P, N)$, the local* NameCert$_{\text{CompleteSeparatedMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the* MetricCompletion$^{\uparrow}$ *source row,* CompleteMetric$^{\uparrow}$ *limit-existence row,* SeparatedMetric$^{\uparrow}$ *zero-distance identity row,* CauchySequenceEquivalence$^{\uparrow}$ *boundary row,* Real$^{\uparrow}$ *readback row, componentwise transport, continuation replay, provenance row, and local naming row;*

2. *classifier stability, stating that the accepted classifier route is preserved by the* hsame *transports in H and replayed by K through the same finite rows;*

3. *completion handoff, stating that any completion-facing consumer first reads the source completion row X and the complete-limit row C before seeing a displayed limit;*

4. *separated uniqueness, stating that two displayed limits for the same Cauchy presentation are compared through E, read as zero through R, and then identified only through the separatedness row S;*

5. *ledger non-escape, stating that the packet exports no quotient completion object, no selected representative, no countable-choice schedule, no host equality of completion points, and no ambient* Real$^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CompleteSeparatedMetricUp.CompleteSeparatedMetricTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 2355.1. The tuple lists the completion source, complete-limit witness, separated identity route, Cauchy-equivalence boundary, real zero-distance readback, transport, replay, provenance, and local naming rows.

Classifier stability follows from the componentwise transport row H together with the replay row K. Transport moves each displayed row to a row of the same kind, and replay consumes the transported rows in the same order as the accepted classifier route.

Completion handoff is the role of X and C. A consumer cannot inspect a limit until the metric-completion source has exposed its completion-facing carrier and the complete-metric row has exposed a displayed limit witness for the regular Cauchy presentation.

Separated uniqueness uses the equivalence boundary E to compare two presentations before the limit comparison is read. The real readback row R supplies the zero-distance observation, and S is the only exported bridge from zero distance to classifier identity.

The ledger statement is an inventory check on the rows of Q. None of the displayed coordinates is a quotient completion class, selected representative, countable-choice schedule, host equality proof, or ambient completeness theorem for Real$^{\uparrow}$. □

Theorem 2355.3 (CompleteSeparatedMetric limit uniqueness)**.** *Let $Q = (X, C, S, E, R, H, K, P, N)$ be an accepted* CompleteSeparatedMetric$^{\uparrow}$ *carrier. If two displayed limit reads arise from the same*

completion-facing Cauchy point-name packet, then their agreement is witnessed only through the separated metric classifier row S and the zero-distance readback row R after the Cauchy-equivalence boundary row E has compared the presentations. The packet exports classifier identity through this displayed route; it does not export host equality of points, quotient completion equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteSeparatedMetricUp.CompleteSeparatedMetricLimitUniqueness`

Proof. Project the complete and separated rows of the carrier. The complete-limit row C supplies the two displayed limit reads, while the separatedness row S is the only coordinate that can turn a zero-distance comparison into classifier identity.

Use the boundary row E to compare the two Cauchy presentations of the same point-name packet. This keeps the comparison inside the finite Cauchy-equivalence surface before any limit identity is read.

The zero-distance witness is then consumed through the $\mathsf{Real}^{\uparrow}$ readback row R. Applying the separated classifier route S yields agreement of the displayed limit reads as BEDC classifier identity.

Finally, H, K, P, N transport, replay, source, and name this same route. Since the carrier has no coordinate for host equality, quotient completion equality, selected representatives, countable choice, or ambient Real completeness, no stronger uniqueness object is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ packet is generated from a MetricCompletion source row, a CompleteMetric limit-existence row, a SeparatedMetric zero-distance identity row, a CauchySequenceEquivalence boundary row, a RealUp readback row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2355.1 and Definition 2355.2 state the finite complete separated metric certificate surface over displayed completion, complete-limit, separatedness, Cauchy-equivalence, real readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteSeparatedMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient completion equality, selected representatives, countable choice, host equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, completion handoff, separated uniqueness, and ledger non-escape for the same finite packet.

Chapter 2356

A Concrete Naming Certificate for $\mathsf{MetricProjection}^{\uparrow}$

Following Hilbert projection theory and constructive convex analysis, the $\mathsf{MetricProjection}^{\uparrow}$ packet records the nearest-point surface for a displayed closed convex subset of a Hilbert carrier as finite BEDC data. It sits after the Hilbert source of chapter 91, the closed convex source of chapter 208, the metric distance ledger of chapter 57, the complete metric limit surface of chapter 128, and the real-name readback of chapter 38. The packet names the projection route without importing a host nearest-point selector, quotient equality of points, countable choice, trichotomy, or ambient real completeness.

Definition 2356.1 (MetricProjection carrier)**.** *A* $\mathsf{MetricProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (H, C, D, I, W, E, T, R, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, including the inner-product carrier, norm ledger, and induced metric route.*
2. *C is the closed convex source row, recording the admitted subset, convex-combination stability, and closedness boundary over the Hilbert source.*
3. *D is the metric distance ledger, exposing only displayed* $\mathsf{Met}^{\uparrow}$ *distance comparisons between the external point, subset witnesses, and candidate endpoints.*
4. *I is the located infimum budget for the squared-distance values over the displayed closed convex source.*
5. *W is the approximating minimizing window, consisting of finite candidates, tolerances, and the monotone comparison schedule used to approach the located infimum.*
6. *E is the projection endpoint row, recording the carried subset point admitted as the limit of the minimizing window.*
7. *T records componentwise* hsame *transport for $H, C, D, I, W, E, R, P, N$.*
8. *R records the displayed* Cont *replay from distance comparisons through located infimum, minimizing windows, endpoint admission, and closest-point reads.*
9. *P is the* Pkg *provenance over* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricProjection}^\uparrow}$ *row.*

The local classifier compares accepted carriers only by these displayed rows. It has no coordinate for a host projection map, selected representative, quotient equality of Hilbert points, ambient choice principle, or global completeness theorem outside the named sources.

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionTasteGate single carrier alignment`

Theorem 2356.2 (MetricProjection NameCert obligations)**.** *For an accepted* $\mathsf{MetricProjection}^\uparrow$ *carrier* $Q = (H, C, D, I, W, E, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricProjection}^\uparrow}$ *surface consists exactly of carrier exposure for the Hilbert source, closed convex source, metric distance ledger, located infimum budget, approximating minimizing window, projection endpoint, componentwise transport, replay route, package provenance, and local naming row. Every projection-facing consumer must read the endpoint through* $H, C, D, I, W, E, T, R, P, N$ *in that order of dependency, and no consumer receives a host nearest-point selector, quotient representative, trichotomy decision, countable-choice schedule, or ambient* $\mathsf{Real}^\uparrow$ *completeness theorem from the packet.*

Proof. Carrier exposure is projection from Definition 2356.1. The tuple lists the Hilbert source, closed convex source, metric distance ledger, located infimum budget, minimizing window, endpoint, transport, replay, provenance, and local naming row as its only coordinates.

The projection route is finite. Distance reads pass through D, lower-bound and approximation reads pass through I and W, and endpoint admission passes through E after the closed convex source row C and complete metric source are displayed.

The remaining rows preserve the same route. Transport T moves only the displayed coordinates, replay R consumes them in the stated dependency order, and P, N record provenance and local naming. Since the carrier lacks rows for host selectors, quotient representatives, trichotomy, countable choice, and ambient real completeness, those objects cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrier namecert obligations`

Theorem 2356.3 (MetricProjection closest-point nonescape)**.** *Every accepted closest-point read exported by a* $\mathsf{MetricProjection}^\uparrow$ *carrier factors through the displayed located infimum budget* I*, approximating minimizing window* W*, and projection endpoint row* E *over the Hilbert, closed convex, metric, complete metric, and real readback sources. Thus a consumer can use the carried endpoint only as a finite BEDC projection endpoint, not as a host nearest-point selector or an unconstrained choice of representative.*

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrier closest point nonescape`

Proof. Begin with an accepted closest-point read. By Definition 2356.2, the read reaches the local naming row only through $H, C, D, I, W, E, T, R, P, N$.

The rows I and W provide the located lower bound and finite approximation route for squared distances. The endpoint row E is admitted only after the minimizing window is replayed over the closed convex source and the complete metric limit surface.

Consequently the exported point is tied to the displayed finite route. Changing the endpoint by appealing to a host projection map, a selected representative, or a choice principle is not a move inside the carrier, so such data cannot escape through the $\mathsf{MetricProjection}^\uparrow$ packet. □

The endpoint row of a $\mathsf{MetricProjection}^\uparrow$ packet consumes $\mathsf{LocatedSet}^\uparrow$ only after the located distance route has displayed the parent $\mathsf{Met}^\uparrow$ row, MetricSubspace admission, located-distance ledger, rational windows, $\mathsf{Real}^\uparrow$ seal, and $\mathsf{LocatedMetric}^\uparrow$ handoff; see chapter 1541.

Theorem 2356.4 (MetricProjection endpoint separation from LocatedSet coordinates)**.** *A projection endpoint exported by an accepted* $\mathsf{MetricProjection}^{\uparrow}$ *carrier cannot be read as a* $\mathsf{LocatedSet}^{\uparrow}$ *coordinate. It is supplied only as the consumer endpoint row* E *after the* $\mathsf{LocatedSet}^{\uparrow}$ *route has displayed the parent metric row, MetricSubspace admission, located-distance ledger, rational windows,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff.*

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrier locatedset endpoint separation`

Proof. Use the carrier route of Definition 2356.1. The endpoint row E appears after the Hilbert, closed convex, metric, located-infimum, and minimizing-window rows have been displayed, and it is replayed only through T, R, P, N as a projection-facing endpoint.

The LocatedSet route supplies the sibling distance data before this endpoint is read. Its parent metric row, MetricSubspace admission, located-distance ledger, rational windows, $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{LocatedMetric}^{\uparrow}$ handoff are source rows for the projection consumer, not coordinates of the endpoint itself.

Therefore a consumer cannot invert the endpoint row into a $\mathsf{LocatedSet}^{\uparrow}$ coordinate. The only accepted read is the forward projection route: first the located-set distance surface, then the $\mathsf{MetricProjection}^{\uparrow}$ endpoint row E as the consumer endpoint. □

Theorem 2356.5 (MetricProjection carrier stability obligation)**.** *Let* $Q = (H, C, D, I, W, E, T, R, P, N)$ *and* $Q' = (H', C', D', I', W', E', T', R', P', N')$ *be accepted* $\mathsf{MetricProjection}^{\uparrow}$ *carriers whose displayed Hilbert source, closed convex source, metric distance ledger, located infimum budget, minimizing window, endpoint row, transport, replay, provenance, and local naming rows agree componentwise through the classifier of Definition 2356.1. Then every accepted projection-facing read through* Q *transports to the corresponding read through* Q' *without adding a host nearest-point selector, quotient representative, trichotomy decision, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Use the carrier inventory of Definition 2356.1. The classifier compares exactly $H, C, D, I, W, E, T, R, P, N$, so any transported read must preserve those rows componentwise.

Apply Definition 2356.2. The local NameCert route reads carrier admission, distance ledgers, located infimum budget, minimizing window, endpoint, transport, replay, provenance, and naming in the displayed order.

Finally apply Definition 2356.3. Since the endpoint can escape only through I, W, E and the structural rows, classifier stability cannot introduce a selector, quotient representative, trichotomy decision, countable-choice schedule, or ambient completeness row. □

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrier obligation carrier stability`

Theorem 2356.6 (MetricProjection located-infimum replay obligation)**.** *For an accepted* $\mathsf{MetricProjection}^{\uparrow}$ *carrier* $Q = (H, C, D, I, W, E, T, R, P, N)$*, every closest-point replay that consumes the located infimum budget must first read the Hilbert row* H*, closed convex row* C*, metric distance row* D*, and located infimum row* I*, then replay the minimizing window* W *before the endpoint row* E *can be admitted. The replay is the route displayed by Definition 2356.2 and remains inside the LocatedSet endpoint-separation boundary of Definition 2356.4.*

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionLocatedInfimumReplayObligation`

Proof. Start from the NameCert obligations of Definition 2356.2. They place $H, C, D, I, W, E, T, R, P, N$ in dependency order, with distance comparison and infimum data preceding the endpoint.

The located infimum row I is not an endpoint. It supplies the lower-bound and approximation budget that W replays through finite minimizing windows over the displayed closed convex source.

Use Definition 2356.4. The LocatedSet route contributes parent metric, subspace, distance, rational-window, Real-seal, and located-metric source rows before the projection endpoint is read, so the replay cannot collapse into a LocatedSet coordinate or hidden closest-point selector. □

Theorem 2356.7 (MetricProjection endpoint admission obligation)**.** *For an accepted* $\mathsf{MetricProjection}^{\uparrow}$ *carrier* $Q = (H, C, D, I, W, E, T, R, P, N)$*, the endpoint row* E *is admitted only after the minimizing-window row* W *has replayed the located-infimum route over* H, C, D, I *and after the LocatedSet source route has supplied its endpoint-separation boundary. Any consumer of* E *must then pass through* T, R, P, N *before using the endpoint as a projection-facing row.*

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrierEndpointAdmissionCertificate`

Proof. Project the carrier and apply Definition 2356.6. The rows H, C, D, I, W are the only route by which the endpoint can be reached from the Hilbert, convex, metric, and located-infimum data.

Next apply Definition 2356.4. It separates the projection endpoint from the LocatedSet coordinate route and forces the source handoff to remain upstream of E.

Finally use Definition 2356.3. The endpoint consumer must pass through transport, replay, provenance, and local naming rows T, R, P, N, so endpoint admission remains a finite BEDC packet boundary and not a host projection principle. □

Theorem 2356.8 (MetricProjection public projection certificate)**.** *An accepted* $\mathsf{MetricProjection}^{\uparrow}$ *packet exports a single public projection certificate only through the rows* $H, C, D, I, W, E, T, R, P, N$*. The certificate exposes the Hilbert source, closed convex source, metric distance ledger, located infimum budget, minimizing window, endpoint admission, componentwise transport, replay, provenance, and local naming rows, and it exports no host selector, quotient equality, choice schedule, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the accepted packet through Definition 2356.1. Its public rows are exactly $H, C, D, I, W, E, T, R, P, N$, so any projection-facing export must keep the Hilbert, convex, metric, infimum, window, endpoint, transport, replay, provenance, and naming coordinates visible.

Apply Definition 2356.2 and Definition 2356.7. These force the endpoint row to appear only after the located-infimum and minimizing-window route, and then through transport, replay, provenance, and local naming.

Finally use Definition 2356.3. Since the carrier lacks rows for selectors, quotient equality, choice schedules, trichotomy, and ambient completeness, the public export is the displayed projection certificate and nothing stronger. □

Lean4 ✓ `BEDC.Derived.MetricProjectionUp.MetricProjectionCarrier public projection certificate`

Closure status for $\mathsf{NameCert}_{\mathsf{MetricProjection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricProjection}^{\uparrow}$ packet is generated from a Hilbert source, closed convex source, metric distance ledger, located infimum budget, approximating minimizing window, projection endpoint, componentwise transport, continuation replay, package provenance, and local naming row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricProjection}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 2356.1, Definition 2356.2, Definition 2356.3, Definition 2356.4, Definition 2356.5, Definition 2356.6, Definition 2356.7, and Definition 2356.8 scope $\mathsf{MetricProjection}^{\uparrow}$ over $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, located-infimum replay, endpoint admission, public projection export, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricProjection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host nearest-point selector, quotient equality of Hilbert points, countable choice, trichotomy, or ambient RealUp completeness.

Upgrade path: The theory axis is mature; formal verification and bridge verification remain pending on their independent axes.

Chapter 2357

A Concrete Naming Certificate for $\mathsf{BorelCantelli}^{\uparrow}$

Following Borel and Cantelli, the $\mathsf{BorelCantelli}^{\uparrow}$ packet records the finite event-sequence ledger used by the first Borel–Cantelli route before any downstream convergence or martingale chapter may cite eventual absence of repeated hits. It sits over the probability-space carrier of chapter 184, the random-variable observation surface of chapter 185, the distribution readback surface of chapter 186, and the independence ledger of chapter 187. The chapter names the summable-event finite-window boundary; it does not import countable choice, a host sigma-algebra completion, an almost-sure quotient, or an infinite limiting theorem as hidden data.

Definition 2357.1 (BorelCantelli carrier)**.** *A* $\mathsf{BorelCantelli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (\Omega, E, W, S, T, U, L, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the probability carrier from which all event reads are taken;*
2. E *is the event-sequence row, recording the finite family of displayed event names and their membership readbacks;*
3. W *is the finite union-window row, recording the initial and tail windows through which repeated-hit requests are inspected;*
4. S *is the partial probability-sum row for the event probabilities visible in* E*;*
5. T *is the tail-sum bound row, recording the displayed finite tail budget used by the first-lemma route;*
6. U *is the finite union-bound ledger, exposing the route from event-window membership through probability subadditivity to the displayed tail bound;*
7. L *is the tail-event ledger, recording that every repeated-hit read is witnessed only through the displayed windows and tail budgets;*
8. H *records componentwise* hsame *transport for the probability source, event sequence, union windows, partial sums, tail bounds, union-bound ledger, tail-event ledger, replay, provenance, and local naming rows;*

9. *C records the displayed* Cont *replay from an event request through Ω, E, W, S, T, U, L in that order;*
10. *P is the* Pkg *provenance over* $\mathsf{BorelCantelli}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelCantelli}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $\Omega, E, W, S, T, U, L, H, C, P, N$. It has no coordinate for a completed measurable universe, a quotient almost-sure equality, an infinite choice of tail witnesses, or the second Borel–Cantelli independence converse.

Lean4 ✓ `BEDC.Derived.BorelCantelliUp.BorelCantelliTasteGate single carrier alignment`

Theorem 2357.2 (BorelCantelli NameCert obligations)**.** *For every accepted* $\mathsf{BorelCantelli}^{\uparrow}$ *carrier $B = (\Omega, E, W, S, T, U, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BorelCantelli}^{\uparrow}}$ *surface consists exactly of carrier admission for the probability source row, event-sequence row, finite union-window row, partial probability-sum row, tail-sum bound row, finite union-bound ledger, tail-event ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public Borel–Cantelli read must expose the same probability source and event sequence before it may inspect a finite window, a partial sum, a tail budget, or a repeated-hit ledger. No Borel–Cantelli obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 2357.1. The displayed tuple exposes exactly the probability source, event sequence, finite windows, partial sums, tail bounds, union-bound ledger, tail-event ledger, transport, replay, provenance, and local naming rows.

The event route is finite because each repeated-hit read first passes through E and W. The probability estimate is then read through S, T, U, so a consumer cannot replace the displayed summability budget while preserving the accepted packet.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for a host measurable completion, quotient almost-sure equality, infinite witness choice, or independence converse, the NameCert obligations are exhausted by the displayed rows. □

Theorem 2357.3 (BorelCantelli first-lemma finite-sum route)**.** *Every accepted* $\mathsf{BorelCantelli}^{\uparrow}$ *first-lemma read factors through the finite route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{Independence}^{\uparrow}.$$

The probability source and event-sequence rows determine each finite union window and partial probability sum before the tail-event ledger is exposed. The route exports no countable-choice selector, no completed almost-sure quotient, no infinite sigma-algebra completion, and no second-lemma independence converse.

Proof. Let $B = (\Omega, E, W, S, T, U, L, H, C, P, N)$ be accepted. Carrier projection exposes the probability-space row Ω and the event-sequence row E before any window or sum can be inspected.

Replay C. A first-lemma request first reads the finite union window W, then the partial probability-sum row S, then the displayed tail budget T, and then the finite union-bound ledger U. Thus the tail-event ledger L receives only finite-window and finite-sum data.

The sibling probability, random-variable, distribution, and independence surfaces supply the named source rows and comparison ledgers, but the first-lemma route uses no independence converse. Componentwise transport through H preserves the same rows, while P, N record provenance and local naming. The excluded infinite and quotient objects are absent from the carrier, so no first-lemma read can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{BorelCantelli}\uparrow}$.

Constructive story: A $\mathsf{BorelCantelli}^{\uparrow}$ packet is generated from a probability source row, event-sequence row, finite union-window row, partial probability-sum row, tail-sum bound row, finite union-bound ledger, tail-event ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelCantelli}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2357.1, Definition 2357.2, and Definition 2357.3 seed the finite event-sequence and first-lemma route over probability windows and tail sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelCantelli}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sigma-algebra completion, quotient almost-sure equality, countable-choice tail selection, the second Borel–Cantelli lemma, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, event-window exactness, partial-sum stability, tail-sum bound routing, union-bound ledger exactness, tail-event ledger non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BorelCantelliUp.

Chapter 2358

A Concrete Naming Certificate for $\mathsf{YoungInequality}^{\uparrow}$

Following Hardy–Littlewood–Polya and the finite conjugate-exponent form of real analysis, the $\mathsf{YoungInequality}^{\uparrow}$ packet records the BEDC route by which a product estimate is read before it is consumed by Holder and Minkowski inequality packets. It sits between the positive-real exponent surface of chapter 1587, the finite Holder handoff of chapter 2100, and the triangle-inequality consumer route of chapter 2343. The packet is a finite two-row real-analysis certificate; it is not a host convexity theorem, an ambient integration theorem, a selected infinite sum, a quotient stream equality, or bridge checking.

Definition 2358.1 (Young inequality carrier)**.** *A* $\mathsf{YoungInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (A, B, E, R, S, V, M, D, F, H, C, P, N)$$

with the following visible coordinates:

1. *A and B are the two displayed* $\mathsf{Real}^{\uparrow}$ *rows whose product is bounded by the Young inequality read.*
2. *E is the conjugate exponent and weight ledger, recording positive exponent rows p, q, the conjugacy certificate, and the displayed weights $1/p$ and $1/q$ used in the finite product comparison.*
3. *R and S are the* $\mathsf{PositiveReal}^{\uparrow}$ *positivity rows for the two real inputs and for the exponent weights needed by the comparison.*
4. *V is the* $\mathsf{HolderInequality}^{\uparrow}$ *handoff row that reads the same conjugate ledger as the one-point precursor of a finite Holder sum.*
5. *M is the* $\mathsf{MinkowskiInequality}^{\uparrow}$ *consumer route, recording where the same finite product estimate may be used inside later triangle-inequality normalization.*
6. *D is the dyadic tolerance budget for positive-real comparison, exponent-weight rounding, product estimation, Holder handoff slack, Minkowski consumer slack, and endpoint rounding.*
7. *F is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal for the displayed Young inequality read.*
8. *H records componentwise* hsame *transport for the real rows, conjugate ledger, positivity rows, Holder handoff, Minkowski consumer route, dyadic budget, endpoint seal, replay, provenance, and local naming rows.*

9. *C records the displayed* Cont *routes for real-row readback, positivity replay, exponent-weight replay, Holder handoff, Minkowski consumer routing, dyadic-budget readback, and endpoint sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{YoungInequality}^{\uparrow}$, $\mathsf{HolderInequality}^{\uparrow}$, $\mathsf{MinkowskiInequality}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite Young inequality packet. It has no coordinate for ambient convexity, host integration, selected infinite sums, quotient equality of streams, or a standard bridge.

Lean4 ✓ `BEDC.Derived.YoungInequalityUp.YoungInequalityTasteGate single carrier alignment`

Theorem 2358.2 (Young inequality NameCert obligations)**.** *For every accepted* $\mathsf{YoungInequality}^{\uparrow}$ *carrier* $Y = (A, B, E, R, S, V, M, D, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the two displayed* $\mathsf{Real}^{\uparrow}$ *rows, conjugate exponent and weight exactness,* $\mathsf{PositiveReal}^{\uparrow}$ *positivity readback, Holder handoff compatibility, Minkowski consumer routing, dyadic tolerance accounting, endpoint sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted product-bound request is classified only by those finite rows.*

Proof. Project the accepted carrier. The analytic coordinates are the two real rows A, B, the conjugate ledger E, the positivity rows R, S, the Holder handoff V, the Minkowski consumer route M, the dyadic budget D, and the endpoint seal F.

The structural rows H, C, P, N then transport, replay, source, and name exactly that finite packet. No row introduces a host convexity principle, an integration theorem, a selected infinite sum, or an ambient completeness assertion.

Thus the local $\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}$ row names the displayed product-bound packet and no additional analytic theorem. □

Theorem 2358.3 (Young inequality Holder–Minkowski handoff)**.** *For every accepted* $\mathsf{YoungInequality}^{\uparrow}$ *carrier* $Y = (A, B, E, R, S, V, M, D, F, H, C, P, N)$*, every Holder-facing or Minkowski-facing read factors through the finite route*

$$A, B, E, R, S, V, M, D, F, H, C, P, N.$$

Consequently the read uses only two displayed $\mathsf{Real}^{\uparrow}$ *rows, the conjugate exponent and weight ledger,* $\mathsf{PositiveReal}^{\uparrow}$ *positivity rows, the* $\mathsf{HolderInequality}^{\uparrow}$ *handoff, the* $\mathsf{MinkowskiInequality}^{\uparrow}$ *consumer route, a dyadic tolerance budget, and a terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal. It supplies no host convexity theorem, no ambient integration theorem, no selected infinite sum, no quotient stream equality, and no bridge-checked standard theorem.*

Proof. The source of the product comparison is the pair A, B together with the positivity rows R, S. The conjugate ledger E fixes the displayed weights and the finite conjugacy certificate used by the Young inequality read.

The Holder handoff V is then a consumer of the same conjugate ledger, not an independent source of exponent data. The Minkowski route M records only where the bounded product estimate can be consumed later, so it does not add a new triangle-inequality theorem to the Young packet.

Finally D and F account for tolerance and endpoint sealing, while H, C, P, N transport, replay, source, and name the route. Hence every downstream read passes through the displayed finite packet. □

Falsifiable prediction: Every $\mathsf{YoungInequality}^{\uparrow}$ product-bound read factors through two RealUp rows, a conjugate exponent and weight ledger, PositiveRealUp positivity rows, HolderInequalityUp handoff, MinkowskiInequalityUp consumer routing, a dyadic tolerance budget, a RealUp endpoint seal, transport, replay, provenance, and local naming.

Independence witness: holder-inequality, minkowski-inequality, positivereal: $\mathsf{YoungInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two RealUp rows, conjugate exponent and weight data, positivity rows, Holder handoff, Minkowski consumer routing, dyadic tolerance accounting, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungInequality}^{\uparrow}$ packet is generated from two $\mathsf{Real}^{\uparrow}$ rows, a conjugate exponent and weight ledger, $\mathsf{PositiveReal}^{\uparrow}$ positivity rows, $\mathsf{HolderInequality}^{\uparrow}$ handoff, $\mathsf{MinkowskiInequality}^{\uparrow}$ consumer routing, a dyadic tolerance budget, a $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2358.1, Definition 2358.2, and Definition 2358.3 seed the finite Young inequality boundary for displayed RealUp product estimates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim host convexity, ambient integration, selected infinite sums, quotient stream equality, bridge checking, or a completed standard theorem for $\mathsf{YoungInequality}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, conjugate-exponent exactness, positivity readback, Holder handoff, Minkowski consumer routing, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.YoungInequalityUp.

Chapter 2359

A Concrete Naming Certificate for PicardIteration$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Banach fixed-point route, the PicardIteration$^{\uparrow}$ packet records the finite contraction-iteration ledger consumed before Picard–Lindelof and fixed-point endpoint packets read a completed limit. It sits between the contraction-map surface of chapter 1418, the one-step Picard route of chapter 395, the complete-metric consumer of chapter 128, the Cauchy-modulus surface of chapter 292, the regular-rational handoff of chapter 301, the real seal of chapter 38, and the fixed-point endpoint discipline of chapter 396. The packet names the finite orbit, residual distances, modulus handoff, and seal boundary without selecting a fixed point or importing host function extensionality.

Definition 2359.1 (PicardIteration carrier)**.** *A* PicardIteration$^{\uparrow}$ *carrier is a finite packet*

$$P = (X, T, x_0, I, r, M, R, E, H, C, Q, N)$$

with the following displayed rows.

1. *X is the complete metric carrier and distance-read row available to the contraction route.*
2. *T is the contraction self-map graph row, including the displayed contraction-bound row used by adjacent iterate comparisons.*
3. *x_0 is the initial-point row from which the orbit is generated.*
4. *I is the finite iterate ledger, recording the visible rows $x_0, x_1, \ldots, x_n$ obtained by repeated reads of T.*
5. *r is the residual-distance row, storing the adjacent distances and contraction-ratio comparisons read from the iterate ledger.*
6. *M is the* CauchyModulus$^{\uparrow}$ *row extracted from I and r.*
7. *R is the* RegSeqRat$^{\uparrow}$ *handoff row used when the orbit is consumed as a regular rational approximation stream.*
8. *E is the* Real$^{\uparrow}$ *seal boundary recording where endpoint-facing consumers may read a real seal after the regular handoff.*
9. *H is componentwise* hsame *transport for the metric carrier, self-map graph, initial point, iterate ledger, residual row, modulus row, regular handoff, real seal boundary, replay, provenance, and local naming rows.*

10. *C is the displayed* Cont *replay from contraction-map admission through Picard iteration, residual comparison, Cauchy-modulus extraction, regular-sequence handoff, and real-seal separation.*
11. *Q is the* Pkg *provenance over* $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}$ *row.*

The carrier exposes a finite contraction-iteration route. It contains no coordinate for a Banach fixed point, a selected limit, quotient equality of completed reals, host function extensionality, or an ambient completeness theorem beyond the displayed complete-metric row.

Lean4 ✓ `BEDC.Derived.PicardIterationUp.PicardIterationTasteGate single carrier alignment`

Theorem 2359.2 (PicardIteration NameCert obligations)**.** *For an accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier* $P = (X, T, x_0, I, r, M, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}$ *surface consists of carrier admission, complete-metric visibility, contraction-map visibility, initial-point visibility, finite iterate-ledger visibility, residual-distance stability, Cauchy-modulus exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff visibility,* $\mathsf{Real}^{\uparrow}$ *seal separation, componentwise transport, continuation replay, provenance, and local naming for the finite packet of Definition 2359.1.*

Proof. Carrier admission projects the displayed tuple. The rows X, T, x_0, I expose the complete metric, contraction graph, initial point, and finite orbit before any convergence-facing row is consumed.

The residual-distance row r is then read from the same finite iterate ledger. Its adjacent-distance and contraction-ratio comparisons determine the modulus row M; no infinite stream, chosen endpoint, or host limit is consulted.

The handoff rows R and E remain downstream boundaries. They mark where a regular rational sequence consumer and a real-seal consumer may read the orbit after the finite modulus route has been displayed, but they do not add quotient equality or a selected real value.

Finally H, C, Q, N transport, replay, source, and name exactly these rows. The certificate therefore exports the contraction-iteration surface and not a Banach fixed-point theorem. □

Theorem 2359.3 (PicardIteration contraction route)**.** *Every accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier factors its iterate and residual reads through*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardContraction}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow}.$$

The self-map row T and initial row x_0 generate the finite ledger I, the adjacent-distance comparisons in r are read through the contraction route, and the modulus row M is visible only after those finite rows have been displayed. No iterate read bypasses the contraction-map graph, the Picard-step ledger, or the complete-metric distance row.

Proof. Start with the contraction-map surface. The graph row T is the only source for the displayed successor rows in I, and the complete-metric row X is the only source for the distance comparisons used by r.

The Picard-step route is finite. Each successor row in I is obtained by replaying the same accepted graph row, so the residual-distance comparisons remain attached to the same contraction certificate rather than to an ambient function space.

The modulus row M is downstream of those reads. It is therefore a Cauchy-modulus handoff for a displayed contraction ledger, not an endpoint selector, quotient-equality claim, or host extensionality principle. □

Theorem 2359.4 (PicardIteration completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier passes through the displayed rows*

$$X,\ T,\ x_0,\ I,\ r,\ M,\ R,\ E,\ H,\ C,\ Q,\ N$$

before any fixed-point or ODE packet may consume the iteration. The route supplies a $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and a* $\mathsf{Real}^{\uparrow}$ *seal boundary for downstream* $\mathsf{BanachFixedPoint}^{\uparrow}$ *and* $\mathsf{PicardLindelof}^{\uparrow}$ *consumers; it does not supply a Banach fixed point, selected limit, quotient equality, host function extensionality, or ambient completeness beyond the named* $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. First expose the finite orbit and residual rows. The rows I and r are generated inside the displayed metric and contraction route, so a completion-facing consumer must first read X, T, x_0, I, r.

Next expose the Cauchy-modulus and regular-sequence handoff. The row M records the finite modulus extracted from the residual ledger, and R is the only regular rational sequence boundary available to downstream consumers.

Finally expose the real-seal boundary and local certificate rows. The row E marks where a real-facing consumer may read a seal after the regular handoff, while H, C, Q, N preserve the same finite route under transport, replay, provenance, and local naming. Since no coordinate stores a selected endpoint or host function equality, downstream fixed-point and ODE packets must supply their own endpoint obligations. □

Theorem 2359.5 (PicardIteration Cauchy-modulus chain)**.** *For every accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier*

$$P = (X, T, x_0, I, r, M, R, E, H, C, Q, N),$$

any completion-facing consumer that reads the $\mathsf{RegSeqRat}^{\uparrow}$ *row R must first read the residual row r and the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M. Thus the regular handoff is generated by displayed contraction residuals through the finite iterate ledger, not by a selected endpoint or by ambient completeness of the metric space.*

Proof. Project the carrier of Definition 2359.1. The rows I, r, M, R occur in the displayed packet order, and the residual row r is the only source for the contraction-distance data used by the modulus row M.

Apply Definition 2359.3. It forces the reads of I, r, M to pass through the contraction graph, finite Picard ledger, complete-metric distance row, and Cauchy-modulus extraction in that order.

Now apply Definition 2359.4. The regular handoff row R is downstream of M, and the real-seal row E is downstream of R. Since no carrier coordinate stores a chosen endpoint or an ambient completeness witness, every completion-facing read of R is generated by the displayed residual-to-modulus chain. □

Theorem 2359.6 (PicardIteration RegSeqRat tail handoff)**.** *For every accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier $P = (X, T, x_0, I, r, M, R, E, H, C, Q, N)$, the residual-distance row r together with the* $\mathsf{CauchyModulus}^{\uparrow}$ *row M determines the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R available to* $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{PicardContraction}^{\uparrow}$ *consumers. This handoff exports no selected fixed point, quotient equality, or ambient completeness beyond the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Proof. Project the carrier of Definition 2359.1. The rows I, r, M, R occur as displayed coordinates of the same finite packet, so a downstream consumer cannot detach the regular handoff from the finite orbit and residual-distance rows.

Apply Definition 2359.3. It routes the residual comparisons through the contraction graph, finite Picard ledger, complete-metric distance row, and Cauchy-modulus extraction.

Next apply Definition 2359.4 and Definition 2359.5. These theorems place R downstream of r and M, and upstream of the real-seal row E consumed by fixed-point chapters.

Thus $\mathsf{BanachContraction}^{\uparrow}$ and $\mathsf{PicardContraction}^{\uparrow}$ may consume a named regular rational tail boundary, but they receive no selected endpoint, quotient equality, host function extensionality, or ambient completeness from the Picard packet. □

Theorem 2359.7 (PicardIteration endpoint uniqueness boundary)**.** *Two endpoint reads from the same accepted* $\mathsf{PicardIteration}^{\uparrow}$ *carrier can be compared only after exposing the same complete-metric row, contraction graph, initial point, finite Picard ledger, residual-distance row,* $\mathsf{CauchyModulus}^{\uparrow}$ *route,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}$ *row. The comparison is therefore a boundary over the displayed packet* $P = (X, T, x_0, I, r, M, R, E, H, C, Q, N)$*; it is not read from ambient completeness, host function extensionality, quotient equality of endpoints, or an uncarried fixed-point selector.*

Proof. Start with the contraction route of Definition 2359.3. It fixes the complete-metric row X, contraction graph T, initial point x_0, finite Picard ledger I, residual-distance row r, and Cauchy-modulus row M before any endpoint-facing data are consumed.

Apply the completion handoff of Definition 2359.4. It places the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R, $\mathsf{Real}^{\uparrow}$ seal E, and structural rows H, C, Q, N downstream of the same displayed contraction ledger.

Use the Cauchy-modulus chain and RegSeqRat tail handoff, Definition 2359.5 and Definition 2359.6. They force both endpoint reads to pass through the same residual-to-modulus-to-tail route before a real-facing seal is visible.

Thus an endpoint comparison can only compare reads that have exposed the same carrier rows. Since the carrier has no coordinate for selected fixed points, quotient endpoint equality, host function extensionality, or ambient completeness beyond X, uniqueness remains a displayed boundary obligation for downstream fixed-point and ODE consumers. □

Falsifiable prediction: Every accepted $\mathsf{PicardIteration}^{\uparrow}$ read factors through one complete metric carrier, one contraction-map row, one initial-point row, one finite iterate ledger, one residual-distance row, one CauchyModulus row, one RegSeqRat handoff, one RealUp seal boundary, hsame transport, Cont replay, Pkg provenance, and NameCert row before any fixed-point or ODE endpoint is consumed.

Independence witness: contractionmapping, picardcontraction, completemetric, cauchymodulus, regseqrat, real, banachfixedpoint: none is bijective with $\mathsf{PicardIteration}^{\uparrow}$ because the packet jointly stores the contraction graph, initial point, finite iteration ledger, residual-distance row, modulus extraction row, regular handoff, real-seal boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2359.1, Definition 2359.2, Definition 2359.3, and Definition 2359.4 seed the finite contraction-iteration packet consumed before fixed-point and ODE endpoint rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardIteration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Banach fixed points, selected limits, quotient equality, host function extensionality, ambient completeness beyond the displayed CompleteMetric row, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, complete-metric visibility, contraction-map visibility, initial-point visibility, finite iterate-ledger visibility, residual-distance stability, Cauchy-modulus exposure, RegSeqRat handoff visibility, Real seal separation, transport, replay, provenance, and local naming.

Constructive story: A PicardIteration$^{\uparrow}$ packet is generated from a complete metric carrier, contraction self-map graph, initial point, finite Picard ledger, residual-distance row, CauchyModulus route, RegSeqRat handoff, RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2360

A Concrete Naming Certificate for $\mathsf{SubstrateTraceInvariant}^{\uparrow}$

The $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ packet realizes the cellular manifest and audit route of chapter 9390 as finite NameCert data. It reads the substrate not as an unbounded physical claim but as a bounded manifest over cellular rules, trace windows, locality cells, continuation rows, audit classifiers, transport, replay, provenance, and local naming. The packet is BEDC-native: it packages the audit invariant that a displayed cellular trace remains accountable to its manifest rows before any downstream trust or universality reading is consumed.

The manifest/audit invariant is consumed through the cellular trust substrate row of chapter 937, the trust manifest surface of chapter 940, the bounded-universal-trace handoff of chapter 944, and the certificate-machine acceptance surface of chapter 3355. These siblings can read the trace only after the manifest, audit classifier, bounded trace, locality, rule replay, transport, continuation, provenance, and local naming rows have been displayed.

Definition 2360.1 (SubstrateTraceInvariant carrier)**.** *A* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ *carrier is a finite packet*

$$S = (M, A, T, W, L, R, U, H, C, P, N)$$

with the following displayed rows.

1. *M is the manifest row, naming the finite cellular rule table, boundary convention, alphabet, origin cell, and public audit handle.*
2. *A is the audit classifier row, recording which manifest entries must be visible before a trace row is trusted.*
3. *T is the bounded trace row, storing the finite time-window and cell-window observations under the manifest.*
4. *W is the window-consistency row, checking that each observed cell in T lies inside the displayed bounded window.*
5. *L is the locality-cell row, exposing the finite neighborhood reads used by each transition check.*
6. *R is the rule-application row, replaying the cellular transition table against the locality rows.*
7. *U is the bounded-universal-trace handoff row, marking where a continuation automaton or cellular trust substrate may consume the trace after audit.*

8. *H is componentwise* hsame *transport for the manifest, audit classifier, bounded trace, window check, locality cells, rule applications, continuation handoff, replay, provenance, and local naming rows.*
9. *C is the displayed* Cont *replay from manifest admission through local trace checking, rule replay, audit classification, and continuation handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$*, cellular trust substrate rows, cellular automaton rows, bounded-universal-trace rows, locality-cell rows, continuation-automaton rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SubstrateTraceInvariant}^{\uparrow}}$ *row.*

The carrier exposes a finite manifest/audit invariant. It has no coordinate for an unbounded cellular universe, a metaphysical substrate commitment, a completed universal simulator, a host trace equality principle, or a choice-selected infinite orbit.

Lean4 (definition) `BEDC.Derived.SubstrateTraceInvariantUp.SubstrateTraceInvariantUp`

Theorem 2360.2 (SubstrateTraceInvariant NameCert obligations)**.** *For an accepted* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ *carrier $S = (M, A, T, W, L, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SubstrateTraceInvariant}^{\uparrow}}$ *surface consists of carrier admission, manifest visibility, audit-classifier visibility, bounded-trace visibility, window consistency, locality-cell visibility, rule-application replay, bounded-universal-trace handoff separation, componentwise transport, continuation replay, provenance, and local naming for the finite packet of Definition 2360.1.*

Proof. Carrier admission projects the displayed tuple. The manifest row M and audit row A are read before the trace row, so the trace cannot be trusted without its finite public manifest and classifier.

The bounded trace row T is then checked by W, L, R. Window consistency keeps every observation inside the displayed cell-time window, locality cells expose the finite neighborhood reads, and the rule-application row replays the manifest rule table against those local rows.

The handoff row U is downstream of the audit route. It permits a trust-substrate or continuation-automaton consumer to cite the trace only after the manifest and local replay rows are visible.

Finally H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. The packet therefore exports a finite audit invariant and not an unbounded substrate theorem. □

Lean4 ✓ `BEDC.Derived.SubstrateTraceInvariantUp.SubstrateTraceInvariantTasteGate single carrier alignment`

Theorem 2360.3 (SubstrateTraceInvariant manifest route)**.** *Every accepted* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ *trace read factors through*

$$M,\ A,\ T,\ W,\ L,\ R,\ U,\ H,\ C,\ P,\ N.$$

In particular, the bounded trace is visible only after the manifest and audit classifier are displayed, and every transition read is replayed through the locality-cell and rule-application rows. The route exposes no unbounded orbit, host trace equality, completed simulator, or independent substrate commitment.

Proof. Start with manifest admission. The rows M and A determine which finite rule, boundary, alphabet, and audit obligations govern the trace.

Now read the displayed trace. The row T is paired with the window row W, so only observations inside the finite manifest window can enter the route.

Finally replay the local dynamics. The locality row L and rule row R supply the finite transition checks, and U, H, C, P, N only hand off, transport, replay, source, and name that checked route. Since no carrier coordinate stores an infinite orbit or completed simulator, the manifest route is exhausted by the listed rows. □

Theorem 2360.4 (SubstrateTraceInvariant audit handoff)**.** *Every audit-facing consumer of an accepted* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ *packet receives a trace only through the manifest route of Definition 2360.3. The handoff to cellular trust substrates, cellular automata, bounded-universal traces, locality cells, and continuation automata is therefore mediated by the same finite manifest, audit classifier, bounded trace, window check, locality replay, rule replay, transport, continuation, provenance, and local naming rows.*

Proof. First expose the local manifest route. By Definition 2360.3, any trace read has already passed through M, A, T, W, L, R.

Next expose the handoff row. The row U records the bounded downstream use of the trace; it does not create another trace, enlarge the window, or add an independent cellular substrate.

Finally carry the route through H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same audited rows, so every downstream consumer receives the finite invariant rather than an unbounded substrate claim. □

Theorem 2360.5 (SubstrateTraceInvariant sibling-lattice route)**.** *Every* $\mathsf{CertificateMachine}^{\uparrow}$ *or cellular trust substrate consumer of an accepted* $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ *packet must first read the displayed manifest row, audit classifier, bounded trace, window check, locality-cell row, rule-application replay, bounded-universal-trace handoff, componentwise transport, continuation replay, provenance, and local* $\mathsf{NameCert}$ *row. Thus certificate-machine acceptance and refusal data, trust-substrate manifest reads, bounded trace consumers, and continuation consumers receive the trace only through the sibling lattice cited above, not through an unbounded cellular universe, a completed simulator, host trace equality, or a choice-selected infinite orbit.*

Proof. Start from the accepted carrier $S = (M, A, T, W, L, R, U, H, C, P, N)$. The manifest route theorem forces every trace read through M, A, T, W, L, R before any downstream handoff is visible.

The row U is the only downstream handoff coordinate. It names the bounded-universal-trace, cellular trust, and continuation-facing use of the audited trace, while the opening dependency paragraph places the certificate-machine consumer after the same manifest and audit rows.

Finally H, C, P, N preserve, replay, source, and name the displayed route. Since the carrier has no coordinate for an unbounded orbit, completed simulator, host trace equality, or selected infinite path, the sibling consumers can receive only the finite trace invariant. □

Falsifiable prediction: Every accepted $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ read factors through a manifest row, audit classifier, bounded trace, window consistency check, locality-cell row, rule-application row, bounded-universal-trace handoff, hsame transport, Cont replay, Pkg provenance, and NameCert row before any cellular trust or continuation consumer reads the trace.

Independence witness: cellulartrustsubstrate, cellularautomaton, bounded-universal-trace, localitycell, continuationautomaton: none is bijective with $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ because the

packet jointly stores a manifest, audit classifier, bounded trace, window consistency row, locality-cell row, rule-application row, downstream handoff, transport, replay, provenance, and local naming.

The Lean TasteGate work for this packet must later provide explicit Nontrivial and FieldFaithful instances preserving $M, A, T, W, L, R, U, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{SubstrateTraceInvariant}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstrateTraceInvariant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2360.1, Definition 2360.2, Definition 2360.3, and Definition 2360.4 seed the finite cellular manifest/audit invariant packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstrateTraceInvariant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an unbounded cellular universe, metaphysical substrate commitment, completed universal simulator, host trace equality, choice-selected infinite orbit, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, manifest visibility, audit-classifier visibility, bounded-trace visibility, window consistency, locality-cell visibility, rule-application replay, bounded-universal-trace handoff separation, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{SubstrateTraceInvariant}^{\uparrow}$ packet is generated from a finite cellular manifest, audit classifier, bounded trace, window consistency row, locality-cell row, rule-application row, bounded-universal-trace handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2361

A Concrete Naming Certificate for $\mathsf{IsbellTightSpan}^{\uparrow}$

Following Isbell's injective-envelope construction and the Lawvere metric-space reading, the $\mathsf{IsbellTightSpan}^{\uparrow}$ packet records a finite distance-envelope boundary for a displayed metric source. It sits after the metric observation rows of chapter 57, beside the Yoneda profile route of chapter 2148, and before the completion-facing handoff of chapter 1506. The packet names only finite extremal-distance rows, one-point extension witnesses, separated comparison, completion and Real readback, transport, replay, provenance, and local NameCert data; it does not assert a host injective hull, selected hyperconvex hull, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2361.1 (Isbell tight-span carrier)**.** *An* $\mathsf{IsbellTightSpan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, F, E, O, S, Q, L, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the finite point classifier and distance observations read from chapter 57.*
2. *F is the finite distance-envelope row. It records the retained functions f from displayed source points to displayed* $\mathsf{Real}^{\uparrow}$ *distance bounds, together with the inequalities $f(x) + f(y) \geq d(x, y)$ visible in the metric row.*
3. *E is the extremal-tightness row, listing the finite witnesses that no retained envelope coordinate can be lowered without breaking a displayed distance-envelope inequality.*
4. *O is the one-point extension witness row. It records finite extension tests saying that a candidate point joined to the source by the envelope distances preserves the metric triangle rows already displayed by M and F.*
5. *S is the separated comparison row: two envelopes are compared only by their displayed distance rows and by the separated zero-distance test consumed by completion readers.*
6. *Q is the completion handoff row toward* $\mathsf{MetricCompletion}^{\uparrow}$*, read only after M, F, E, O, S have been displayed.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *readback row for the retained envelope distances and separated comparison distances.*

8. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *windows used to read the displayed real distance rows.*
9. *H stores componentwise* hsame *transport for the metric source, envelope inequalities, extremal witnesses, one-point extension tests, separated comparison, completion handoff, Real readback, and local naming rows.*
10. *C stores displayed* Cont *replay from metric observations to envelope inequalities, from envelopes to one-point extension tests, and from separated comparison to completion and Real readback.*
11. *P is the* Pkg *provenance over* $\mathsf{IsbellTightSpan}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{IsbellTightSpan}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, E, O, S, Q, L, R, H, C, P, N$. It has no coordinate for a host injective envelope, a selected global hull point, a quotient completion, an open-cover compactness principle, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.IsbellTightSpanUp.IsbellTightSpanTasteGate single carrier alignment`

Theorem 2361.2 (Isbell tight-span NameCert obligations)**.** *For every accepted* $\mathsf{IsbellTightSpan}^{\uparrow}$ *carrier $I = (M, F, E, O, S, Q, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IsbellTightSpan}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, finite distance-envelope inequalities, extremal-tightness witnesses, one-point extension witnesses, separated comparison, completion handoff,* $\mathsf{Real}^{\uparrow}$ *readback,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the packet must inspect these rows before treating the tight-span boundary as a completion neighbour.*

Proof. Project the carrier of Definition 2361.1. The metric source row M and distance-envelope row F are the only places where source distances and envelope inequalities are displayed.

The extremal row E and one-point extension row O record the injective-envelope tests as finite witnesses. The separated comparison row S and completion handoff row Q are downstream of those witnesses, so a completion consumer cannot bypass them.

Finally read L, R, H, C, P, N. These rows provide Real readback, finite regular and stream windows, transport, replay, provenance, and local naming for the same route. Since no other classifier coordinate exists, the listed rows exhaust the NameCert surface. □

Theorem 2361.3 (Isbell tight-span metric handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{IsbellTightSpan}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow} \longmapsto F \longmapsto E \longmapsto O \longmapsto S \longmapsto Q, L, R \longmapsto H, C, P, N.$$

Thus the packet supplies a finite injective-envelope boundary adjacent to metric completion: it keeps metric-source observations, envelope inequalities, one-point extension witnesses, separated comparison, completion readback, and Real readback visible, while exporting no host injective hull, selected limit, quotient completion, countable-choice selector, or ambient completeness principle.

Proof. Start with the source metric row. The carrier allows a completion-facing read only after the finite distance-envelope inequalities in F and extremal-tightness witnesses in E have been displayed.

Next read the one-point extension and separated comparison rows. The row O records the finite triangle-preservation tests for adding an envelope point, and S records the separated comparison used by completion consumers.

The handoff rows Q, L, R then expose only completion-facing and Real-facing readback for the same finite packet. Transport, replay, provenance, and local naming by H, C, P, N preserve that route and do not add any selected hull point or ambient completeness witness. □

Falsifiable prediction: Every accepted IsbellTightSpanUp completion-facing read factors through finite distance-envelope inequalities and one-point extension witnesses before any MetricCompletionUp or RealUp readback is exposed.

Independence witness: metric, metriccompletion, yonedacompletion: IsbellTightSpanUp is not bijective to any listed sibling because it jointly carries distance-envelope inequalities, extremal-tightness witnesses, one-point extension tests, separated comparison, completion handoff, Real readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{IsbellTightSpan}^{\uparrow}}$.

Constructive story: An $\mathsf{IsbellTightSpan}^{\uparrow}$ packet is generated from a MetricUp source row, finite distance-envelope inequalities, extremal-tightness witnesses, one-point extension tests, separated comparison, MetricCompletionUp handoff, RealUp readback, RegSeqRat and StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IsbellTightSpan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2361.1, Definition 2361.2, and Definition 2361.3 display the finite tight-span envelope route adjacent to metric completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IsbellTightSpan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host injective envelopes, selected hyperconvex hulls, quotient completion, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires Lean carrier admission, distance-envelope inequalities, extremal witnesses, one-point extension tests, separated comparison, and completion handoff for BEDC.Derived.IsbellTightSpanUp.

Chapter 2362

A Concrete Naming Certificate for $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the Arzela–Ascoli equicontinuity route, the $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ packet records the family-level handoff by which a compact finite-net ledger supplies one displayed source-radius row for a finite family of continuous maps. It sits between the compact-uniform-continuity route of chapter 388, the compact-metric finite-net surface of chapter 129, the graph and pointwise-modulus rows of chapter 130, and the downstream output interface of chapter 369. The certificate names the reusable family-level finite-net route, not an open-cover compactness theorem, a host function-space topology, a selected global modulus function, or an ambient Arzela–Ascoli compactness principle.

Definition 2362.1 (Finite-net equicontinuity carrier)**.** *A* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, R, M, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-metric finite-net row read from* $\mathsf{CompactMetric}^{\uparrow}$*, including the finite center* $\mathsf{ProbeBundle}$ *and its coverage radius ledger.*
2. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows sharing the same source and target metric surfaces.*
3. *B records the center-indexed pointwise bounds for every displayed family member at every center of the finite* $\mathsf{ProbeBundle}$*.*
4. *R is the rational tolerance and source-radius ledger used to compare the center bounds and the compact finite-net radius.*
5. *M is the family-level finite-net modulus row produced by folding K, F, B, and R.*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, exposing the resulting source-radius row as a reusable modulus boundary.*
7. *H records componentwise* hsame *transports for the compact-net, family, bound, rational, modulus, output, route, provenance, and local naming rows.*
8. *C records the displayed* Cont *routes by which compact-uniform-continuity, continuous-map, compact-metric, equicontinuity, and uniform-modulus consumers read the same packet.*

9. *P is the* Pkg *provenance over* FiniteNetEquicontinuity$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{FiniteNetEquicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, F, B, R, M, U, H, C, P, N$. It has no coordinate for host function equality, selected subcovers, selected global moduli, open-cover compactness, or ambient function-space compactness.

Lean4 (definition) `BEDC.Derived.FiniteNetEquicontinuityUp.FiniteNetEquicontinuityUp`
Lean4 ✓ `BEDC.Derived.FiniteNetEquicontinuityUp.FiniteNetEquicontinuityTasteGate single carrier alignment`

Theorem 2362.2 (Finite-net equicontinuity obligation surface)**.** *For an accepted* FiniteNetEquicontinuity$^{\uparrow}$ *carrier $E = (K, F, B, R, M, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteNetEquicontinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-metric finite-net row, finite family ledger, center-indexed bound rows, rational tolerance ledger, family-level modulus row, uniform-modulus handoff row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every downstream read must preserve the same finite* ProbeBundle *centers, the same family ledger, the same rational radius comparison, and the same local NameCert row.*

Proof. Project the carrier of Definition 2362.1. The finite-net data are exactly K, the family data are exactly F, and the center-indexed estimates are exactly B.

The rational ledger R compares the compact-net radius with the source-radius rows needed by the displayed family estimates. Folding those rows gives M, and the handoff row U exposes that folded radius only as a UniformModulus$^{\uparrow}$ boundary.

Transport, replay, provenance, and local naming are the rows H, C, P, N. Since no further coordinate exists, the certificate surface has no path to host function equality, selected subcovers, selected global moduli, open-cover compactness, or ambient function-space compactness. □

Theorem 2362.3 (Finite-net equicontinuity uniform-modulus handoff)**.** *Every* UniformModulus$^{\uparrow}$ *read exported from an accepted* FiniteNetEquicontinuity$^{\uparrow}$ *packet factors through the finite route*

$$K, F, B, R, M, U, H, C, P, N.$$

The route first consumes compact finite-net centers from CompactMetric$^{\uparrow}$*, then reads the finite* ContinuousMap$^{\uparrow}$ *family rows and center-indexed bounds, then folds the rational radius ledger into the family-level modulus row M, and only then exposes the output row U. The handoff does not export open-cover compactness, an Arzela–Ascoli compactness theorem, host function extensionality, selected global moduli, or a function-space topology.*

Proof. Begin with the finite-net row K. It supplies the displayed ProbeBundle centers and coverage radius through the compact-metric certificate, so the route starts from finite carrier data rather than from an open cover.

Next read the family ledger F and the center-indexed rows B. Each family member is consumed through its ContinuousMap$^{\uparrow}$ row, and the estimates are attached only to centers already named by K.

Finally use the rational ledger R to form M and expose U. The hsame, Cont, Pkg, and local NameCert rows preserve the same finite packet during transport and replay, so the exported uniform-modulus row is exactly the finite-net family handoff and not an ambient compactness or function-space assertion. □

Theorem 2362.4 (Compact finite-net coverage row)**.** *Let $E = (K, F, B, R, M, U, H, C, P, N)$ be an accepted* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *carrier. Every accepted read of E exposes the compact finite-net row K before any source-family radius row is read from F, B, or R. Thus the inspected* $\mathsf{ProbeBundle}$ *centers and their compact-metric coverage ledger are visible before the family estimates can be folded toward M or exported through U.*

Proof. Project the carrier of Definition 2362.1. The row K is the only coordinate carrying compact-metric finite-net centers and the associated coverage ledger, while F, B, R are downstream rows for family membership, center-indexed estimates, and rational radius comparison.

Apply Definition 2362.2. It requires every downstream read to preserve the same finite $\mathsf{ProbeBundle}$ centers and the same rational radius comparison. Apply Definition 2362.3. The handoff route begins at K and reaches U only after the family rows and rational ledger have been consumed.

Since the carrier has no coordinate for open-cover compactness, selected moduli, host function-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no consumer can bypass K by appealing to one of those external surfaces. □

Theorem 2362.5 (Finite family row stability)**.** *Let $E = (K, F, B, R, M, U, H, C, P, N)$ be an accepted* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *carrier. The finite family row F and bound ledger B are transported componentwise only by the* hsame *row H, and every replay of those rows to compact-metric, continuous-map, equicontinuity, or uniform-modulus consumers factors through the displayed* Cont *routes in C.*

Proof. Project the carrier of Definition 2362.1. The family ledger is F, the center-indexed bound ledger is B, the transport coordinate is H, and the displayed replay coordinate is C.

Use Definition 2362.2. Its obligation surface names componentwise hsame transport and displayed Cont replay as part of the local $\mathsf{NameCert}_{\mathsf{FiniteNetEquicontinuity}^{\uparrow}}$ surface, with no additional transport coordinate.

Use Definition 2362.3. The route to U reads F and B after K and before R, M, U, so the same family and bound ledgers are preserved during the handoff. The carrier excludes open-cover compactness, selected moduli, host function-space compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so those surfaces cannot supply an alternate replay path. □

Theorem 2362.6 (Rational radius folding order)**.** *Let $E = (K, F, B, R, M, U, H, C, P, N)$ be an accepted* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *carrier. The rational radius data R fold into the uniform-modulus handoff row U only through the family-level modulus row M, and only after the compact-net row K, finite family row F, and bound ledger B have been displayed.*

Proof. Project the carrier of Definition 2362.1. The row R is the rational tolerance and source-radius ledger, M is the family-level finite-net modulus row, and U is the downstream $\mathsf{UniformModulus}^{\uparrow}$ handoff row.

Apply Definition 2362.2. It places carrier admission, compact finite-net coverage, finite family data, bound rows, rational comparison, modulus formation, and uniform-modulus output in the same local NameCert surface.

Apply Definition 2362.3. The exported route is exactly $K, F, B, R, M, U, H, C, P, N$, so R cannot reach U before K, F, B are displayed and cannot skip the fold row M. Because the packet has no coordinate for open-cover compactness, selected moduli, host function-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no external compactness or function-space argument can replace this finite fold. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteNetEquicontinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteNetEquicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2362.1, Definition 2362.2, and Definition 2362.3 seed the family-level finite-net equicontinuity route over compact-metric centers, continuous-map family rows, rational radius ledgers, uniform-modulus handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteNetEquicontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, selected global moduli, open-cover compactness, Arzela–Ascoli compactness, host function extensionality, function-space topology, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net coverage, finite family row stability, rational radius folding, uniform-modulus handoff, and non-escape for BEDC.Derived.FiniteNetEquicontinuityUp.

Constructive story: A $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ packet is generated from CompactMetricUp finite-net centers, ContinuousMapUp family rows, center-indexed estimates, rational tolerance histories, a family-level finite-net modulus row, a UniformModulusUp output row, componentwise hsame transport, Cont replay, provenance, and local naming.

Chapter 2363

A Concrete Naming Certificate for $\mathsf{LaguerrePolya}^{\uparrow}$

Following Laguerre and Polya's real-rooted polynomial limit theory, the $\mathsf{LaguerrePolya}^{\uparrow}$ packet records a function only through polynomial approximants with displayed real roots, coefficient-window convergence, and a real-power-series endpoint. It consumes the finite real-polynomial evaluator of chapter 1728, the coefficient and radius-window surface of chapter 1583, the root-bracket boundary of chapter 1786, the terminal real-name seal of chapter 38, and the finite readback rows of chapter 301 and chapter 126. The packet names the constructive Laguerre–Polya route as BEDC certificate data; it does not import a host entire-function primitive, a selected infinite product, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2363.1 (Laguerre–Polya carrier)**.** *A* $\mathsf{LaguerrePolya}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, R, S, Q, W, E, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the approximant-family row. It records a finite inspection of* $\mathsf{RealPolynomial}^{\uparrow}$ *names, their coefficient windows, degree bounds, evaluation requests, and the index window used by the local certificate.*
2. *B is the real-root evidence row. For every inspected approximant it stores the finite* $\mathsf{RealRootBracket}^{\uparrow}$ *data witnessing the displayed real-root route for the roots relevant to that inspection.*
3. *R is the coefficient-readback row. It records* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks for the approximant coefficients, center coefficients, bracket endpoints, and requested real output names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It stores the finite windows used to compare polynomial coefficients, root brackets, partial evaluations, and endpoint power-series coefficients.*
5. *Q is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *endpoint row. It exposes the limiting coefficient stream, radius or subradius window, partial-sum route, and endpoint seal available to Real-facing consumers.*
6. *W is the convergence-window ledger. It records the finite Cauchy comparison between inspected polynomial coefficient windows and the endpoint power-series windows at the requested precision.*
7. *E is the evaluation-compatibility row. It records that each displayed polynomial evaluation from A and each endpoint read from Q use the same real-name, readback, and window discipline.*

8. *T is the real-root limit route row. It records how the root-bracket data B is retained through the finite coefficient-window convergence ledger before the endpoint series is exposed.*

9. *H records componentwise* hsame *transport for approximant rows, root brackets, readbacks, stream windows, endpoint power-series rows, convergence windows, evaluation compatibility, the root-limit route, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a Laguerre–Polya request through polynomial approximants, real-root brackets, regular readbacks, stream windows, endpoint power-series rows, and the root-limit route.*

11. *P is the* Pkg *provenance over* $\mathsf{LaguerrePolya}^{\uparrow}$, $\mathsf{RealPolynomial}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RealRootBracket}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LaguerrePolya}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, R, S, Q, W, E, T, H, C, P, N$. It has no coordinate for host entire functions, selected infinite products, quotient function spaces, non-displayed zero sets, countable-choice selectors, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LaguerrePolyaUp.LaguerrePolyaTasteGate single carrier alignment`

Theorem 2363.2 (Laguerre–Polya NameCert obligations)**.** *For every accepted* $\mathsf{LaguerrePolya}^{\uparrow}$ *carrier*

$$L = (A, B, R, S, Q, W, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LaguerrePolya}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approximant family, real-root brackets, regular readbacks, stream windows, endpoint power-series row, convergence-window ledger, evaluation compatibility, real-root limit route, transport, replay, provenance, and local naming rows;*

2. *approximant discipline, requiring A to expose only* $\mathsf{RealPolynomial}^{\uparrow}$ *names with finite coefficient windows and displayed evaluation requests;*

3. *real-root discipline, requiring B to route every inspected approximant root through* $\mathsf{RealRootBracket}^{\uparrow}$ *rows before it can contribute to the limiting packet;*

4. *endpoint discipline, requiring Q to expose the limiting object through* $\mathsf{RealPowerSeries}^{\uparrow}$ *coefficient, radius-window, partial-sum, and endpoint rows;*

5. *convergence discipline, requiring W and E to compare approximant rows and endpoint rows only through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readbacks;*

6. *limit-route discipline, requiring T to retain the bracketed real-root information through the displayed coefficient-window convergence route;*

7. *non-escape, requiring H, C, P, N to export only the displayed route and no host entire function, selected infinite product, quotient function equality, non-displayed zero set, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Laguerre–Polya obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2363.1. The accepted packet displays exactly the approximant family, root-bracket rows, readback rows, stream windows, endpoint power-series row, convergence ledger, evaluation compatibility row, root-limit route, and structural rows.

The approximant and root disciplines are the first stage. The row A gives only finite $\mathsf{RealPolynomial}^{\uparrow}$ data, while B attaches the relevant roots to $\mathsf{RealRootBracket}^{\uparrow}$ evidence before any limiting route is available.

The endpoint and convergence disciplines are the second stage. The row Q exposes the limiting object as a $\mathsf{RealPowerSeries}^{\uparrow}$ endpoint, while W and E compare approximants and endpoint reads through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The limit-route discipline is the final analytic step. The row T records that the displayed real-root evidence survives exactly along the coefficient-window convergence route already stored in W and E.

Finally H, C, P, N transport, replay, source, and name the same finite data. Since no excluded analytic object appears as a carrier coordinate, the local NameCert surface has no obligation row for such an object. □

Theorem 2363.3 (Laguerre–Polya real-root limit route)**.** *Every Real-facing read from an accepted* $\mathsf{LaguerrePolya}^{\uparrow}$ *carrier factors through*

$$A, B \longmapsto R, S \longmapsto Q, W, E \longmapsto T \longmapsto H, C, P, N.$$

Thus the consumer first reads finite $\mathsf{RealPolynomial}^{\uparrow}$ *approximants and their* $\mathsf{RealRootBracket}^{\uparrow}$ *evidence, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *comparison windows, then the* $\mathsf{RealPowerSeries}^{\uparrow}$ *endpoint and convergence ledger, and only then the retained real-root limit route. The handoff exports a displayed Laguerre–Polya certificate, not a host entire-function primitive or selected zero set.*

Proof. Start with an accepted packet $L = (A, B, R, S, Q, W, E, T, H, C, P, N)$. The only approximant coordinates are A and B, so a consumer cannot read a limiting function before it has inspected finite real-polynomial rows and their displayed real-root brackets.

The comparison stage is R, S, W, E. These rows bind coefficient reads, bracket endpoints, real-name windows, and approximant-versus-endpoint comparisons to regular rational readbacks and finite stream windows.

The endpoint stage is Q. It exposes the limiting object only as the $\mathsf{RealPowerSeries}^{\uparrow}$ endpoint row named by the same comparison windows, so the endpoint is not a host entire function or quotient function-space element.

The retained root route is T, followed by H, C, P, N. These rows preserve, replay, package, and name the displayed data, and therefore the Real-facing consumer receives exactly the finite Laguerre–Polya route listed in the statement. □

Falsifiable prediction: Every $\mathsf{LaguerrePolya}^{\uparrow}$ read factors through finite $\mathsf{RealPolynomial}^{\uparrow}$ approximants, $\mathsf{RealRootBracket}^{\uparrow}$ root evidence, $\mathsf{RegSeqRat}^{\uparrow}$ coefficient readbacks, $\mathsf{StreamName}^{\uparrow}$ comparison windows, one $\mathsf{RealPowerSeries}^{\uparrow}$ endpoint row, a convergence-window ledger, evaluation compatibility, a retained real-root limit route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real-polynomial, realpowerseries, real-root-bracket, real, regseqrat, streamname: $\mathsf{LaguerrePolya}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries real-rooted polynomial approximants, displayed root brackets, coefficient-window convergence, endpoint power-series data, evaluation compatibility, root-limit routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LaguerrePolya}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{LaguerrePolya}\uparrow}$, seed)

Scope closed: Definition 2363.1, Definition 2363.2, and Definition 2363.3 seed the finite Laguerre–Polya route from real-rooted polynomial approximants through coefficient-window convergence to a real-power-series endpoint.

Formal verification: FormalStatus(NameCert$_{\text{LaguerrePolya}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host entire functions, selected infinite products, quotient function spaces, non-displayed zero sets, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, approximant discipline, real-root discipline, endpoint discipline, convergence discipline, limit-route discipline, and non-escape for BEDC.Derived.LaguerrePolyaUp.

Constructive story: A LaguerrePolya$^{\uparrow}$ packet is generated from finite RealPolynomial$^{\uparrow}$ approximants, RealRootBracket$^{\uparrow}$ root evidence, RegSeqRat$^{\uparrow}$ coefficient and endpoint readbacks, StreamName$^{\uparrow}$ comparison windows, a RealPowerSeries$^{\uparrow}$ endpoint row, convergence-window and evaluation-compatibility ledgers, a retained real-root limit route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2364

A Concrete Naming Certificate for $\mathsf{FiniteAutomaton}^{\uparrow}$

Following Hopcroft and Ullman, the $\mathsf{FiniteAutomaton}^{\uparrow}$ packet records the finite control surface of a deterministic finite automaton as displayed BEDC data rather than as a quotient of accepted languages. It sits between the accepted-language export of chapter 351, the finite word-spine constraints used by chapter 2298, and the bounded transition-trace discipline of chapter 280.

The packet is intentionally smaller than the language chapter and smaller than the machine chapter. It names one automaton presentation, one finite input window, one run trace, one endpoint classifier, and the transport and provenance rows needed by finite consumers.

Definition 2364.1 (FiniteAutomaton carrier)**.** *A* $\mathsf{FiniteAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, \Sigma, \delta, q_0, F, w, \rho, e, \chi, H, C, P, N)$$

with the following displayed rows:

1. *Q is a finite state carrier row, read through the same finite-row discipline as the state component of chapter 351;*
2. *Σ is the alphabet-window row naming exactly the symbols inspected by the finite input word;*
3. *δ is the transition table row whose displayed entries consume a state row and a symbol row and return the next state row;*
4. *q_0 is the start-state row inside Q;*
5. *F is the accepting-state row inside Q;*
6. *w is the finite input window as a displayed word spine over Σ;*
7. *ρ is the run-trace ledger beginning at q_0 and reading w one symbol at a time through δ;*
8. *e is the endpoint row of ρ;*
9. *χ is the classifier row saying that the endpoint is accepted exactly when e is displayed inside F;*
10. *H stores componentwise* hsame *transports for the state carrier, alphabet window, transition table, input window, run trace, endpoint, and classifier rows;*
11. *C stores the* Cont *replay routes that let a consumer re-read the input window, the transition table entry used at each step, the run endpoint, and the classifier row;*

12. *P stores the* Pkg *provenance naming the finite-row, word-spine, regular-language, and Turing-machine comparison surfaces consumed by the packet;*
13. *N stores the local* $\mathsf{NameCert}_{\mathsf{FiniteAutomaton}^{\uparrow}}$ *rows.*

The carrier predicate is $\mathsf{FiniteAutomatonPacket}(A)$*. It exports no nondeterministic branch choice, minimization quotient, regular-expression equivalence, host language set, infinite automaton, or unbounded computation trace.*

Lean4 ✓ `BEDC.Derived.FiniteAutomatonUp.TasteGate.FiniteAutomatonTasteGate single carrier alignment`

Theorem 2364.2 (FiniteAutomaton NameCert obligations)**.** *For every accepted* $\mathsf{FiniteAutomaton}^{\uparrow}$ *carrier*

$$A = (Q, \Sigma, \delta, q_0, F, w, \rho, e, \chi, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteAutomaton}^{\uparrow}}$ *surface consists of carrier admission, deterministic transition replay, endpoint classifier stability,* hsame *transport,* Cont *replay, and* Pkg *provenance for the rows in Definition 2364.1. A consumer may project the finite state carrier, alphabet window, transition table, start state, accepting-state row, input window, run trace, endpoint, classifier, and local naming rows, but it obtains no language quotient, hidden recognizer state, minimization datum, regular-expression conversion, or unbounded machine trace.*

Proof. Project the accepted carrier A. The finite rows $Q, \Sigma, \delta, q_0, F, w$ name the automaton presentation and the inspected input window, while ρ, e, χ name the run trace, endpoint, and acceptance classifier.

Deterministic replay is obtained by reading the word spine in w. At the empty window the trace endpoint is the start-state row q_0. At a cons window, the displayed transition-table entry in δ supplies the unique next state row, and the Cont replay route passes the remaining window to the same ledger.

Transport and provenance are exactly the rows H, C, P, N. Since those rows only compare and replay the displayed carrier coordinates, the certificate has no coordinate from which a quotient language, minimized automaton, hidden state, regular-expression conversion, or unbounded computation trace could be read. □

Theorem 2364.3 (FiniteAutomaton deterministic replay exactness)**.** *Let* $A = (Q, \Sigma, \delta, q_0, F, w, \rho, e, \chi, H, C, P, N)$ *be an accepted* $\mathsf{FiniteAutomaton}^{\uparrow}$ *carrier. Every accepted replay of the run trace ρ factors through the displayed start-state row q_0, the finite input window w, the transition table δ, the component-wise* hsame *rows H, the* Cont *replay rows C, and the same package provenance P, N. In particular, two replays with the same displayed rows have the same endpoint state and the same classifier row χ.*

Proof. Induct on the displayed input window w. For the empty window, carrier admission identifies the run endpoint with q_0, and the classifier row reads only whether this displayed state lies in F.

For a cons window, project the current state and head symbol. The transition table row has one displayed entry for that pair, so the next state of the replay is fixed. The Cont row then hands the tail window to the induction hypothesis with the transported state, transition, and classifier rows.

The hsame and Pkg rows only preserve the same displayed coordinates. Therefore no alternate endpoint can be produced from minimization, a language quotient, a hidden recognizer state, or an external computation trace. □

Theorem 2364.4 (FiniteAutomaton regular-language consumer)**.** *Every public* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that reads a displayed deterministic finite-automaton presentation can factor its finite recognition read through a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *packet with the same state carrier, alphabet window, transition table, input word, run trace, endpoint classifier,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The factorization preserves the accepted-word endpoint exposed by chapter 351 and remains below the bounded trace surface of chapter 280.*

The factorization does not assert that all presentations of a regular language are equivalent, does not minimize the automaton, and does not translate the packet into a context-free or Turing-machine object. The context-free pumping consumer in chapter 2298 may read only the finite word-window and replay rows displayed here.

Proof. Begin with the public finite-automaton rows already exposed by the $\mathsf{RegularLanguage}^{\uparrow}$ consumer: finite alphabet, finite state carrier, start state, accepting-state row, transition table, input word, deterministic run, and accepted endpoint. These rows are exactly the first nine coordinates of Definition 2364.1.

Next copy the consumer's visible transport, continuation, and provenance rows into H, C, P, N. The deterministic replay exactness theorem then identifies the accepted endpoint with the replay endpoint of the $\mathsf{FiniteAutomaton}^{\uparrow}$ packet.

The resulting factorization is a projection to a smaller finite presentation. It therefore preserves the regular-language accepted-word endpoint while excluding minimization, language quotienting, context-free grammar conversion, and unbounded Turing-machine execution, none of which are carrier rows of $\mathsf{FiniteAutomaton}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteAutomaton}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteAutomaton}^{\uparrow}$ packet is generated from finite state rows, an alphabet window, a transition table, a start-state row, an accepting-state row, a finite input window, a deterministic run trace, an endpoint classifier, componentwise hsame transports, Cont replay routes, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteAutomaton}^{\uparrow}}$, obligation)

Scope closed: Definition 2364.1, Definition 2364.2, Definition 2364.3, and Definition 2364.4 bind the finite automaton carrier, local NameCert surface, deterministic replay route, and regular-language consumer factorization.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteAutomaton}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close minimization, regular-expression conversion, equivalence of all automaton presentations, nondeterministic branching, or machine-checked bridge evidence.

Upgrade path: Scoped closure requires checked Lean rows for the carrier, deterministic replay exactness, and the regular-language consumer factorization.

Chapter 2365

A Concrete Naming Certificate for $\mathsf{SorgenfreyPlane}^{\uparrow}$

Following Sorgenfrey and Kelley, the $\mathsf{SorgenfreyPlane}^{\uparrow}$ packet records the product of two lower-limit lines as finite NameCert data over the already named one-dimensional lower-limit and product-topology surfaces. It sits after the lower-limit line of chapter 3363, the product-topology route of chapter 2000, the indexed-open carrier of chapter 88, and the observable-open classifier of chapter 1860. The packet names the lower-limit product boundary without importing host metrizability, a powerset topology on pairs, selected neighbourhood bases, quotient equality of real pairs, or an ambient set-theoretic plane.

Definition 2365.1 (Sorgenfrey plane carrier)**.** *A* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L_X, L_Y, Q, O_X, O_Y, B, S, H, C, R, N)$$

with the following displayed rows:

1. L_X *and* L_Y *are the two* $\mathsf{SorgenfreyLine}^{\uparrow}$ *source rows, each exposing lower-limit interval windows only through the finite carrier of Definition 3363.1.*
2. Q *is the* $\mathsf{ProductTopology}^{\uparrow}$ *product row, reading paired histories through component topology rows and rectangular subbasis data.*
3. O_X *and* O_Y *are the two component* $\mathsf{Topology}^{\uparrow}$ *indexed-open rows exported by the lower-limit line carriers.*
4. B *is the lower-limit rectangle ledger, recording basic rows of the form* $[a, b) \times [c, d)$ *as paired component windows rather than as a host powerset of* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$.
5. S *is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row used when a consumer asks whether a displayed rectangle or generated product open has been observed.*
6. H *records componentwise* hsame *transport for both lower-limit line rows, the product-topology row, the component topology rows, the rectangle ledger, the observable-open row, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from a product observation through component lower-limit windows, rectangular subbasis membership, generated product-topology admission, observable-open readback, transport, provenance, and local naming.*

8. *R is the* Pkg *provenance over* $\mathsf{SorgenfreyPlane}^{\uparrow}$, $\mathsf{SorgenfreyLine}^{\uparrow}$, $\mathsf{ProductTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SorgenfreyPlane}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It stores no host plane, no metric witness, no second-countable base, no powerset topology, no quotient real-pair equality, no selected open family, and no hidden product carrier outside the displayed $\mathsf{ProductTopology}^{\uparrow}$ *route.*

Theorem 2365.2 (Sorgenfrey plane NameCert obligations)**.** *For every accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier*

$$P = (L_X, L_Y, Q, O_X, O_Y, B, S, H, C, R, N),$$

the local $\mathsf{NameCert}_{\mathsf{SorgenfreyPlane}^{\uparrow}}$ *obligations are exactly the displayed finite product rows:*

1. *component lower-limit admission through L_X and L_Y;*
2. *product-topology admission through Q, with the two topology rows O_X, O_Y exposed before any product read is consumed;*
3. *rectangle-ledger exactness through B, requiring every basic lower-limit product neighbourhood to be a paired half-open window row;*
4. *observable-open readback through S, so generated product opens are inspected through finite truth rows rather than host propositions;*
5. *transport, replay, provenance, and local naming through H, C, R, N.*

These obligations do not assert metrizability, normality, second countability, host separation theorems, powerset topology on pairs, quotient real-pair equality, selected bases, Lean verification, or bridge checking.

Proof. Project the accepted carrier. The only component sources are the lower-limit line rows L_X, L_Y, and each component topology read must pass through O_X, O_Y before the product row is consumed.

The product route is then fixed by Q. Its rectangular subbasis access is refined by B, whose entries are paired half-open windows and not arbitrary subsets of a host plane.

Observable-open requests pass through S, so membership reads return displayed finite truth data. The structural rows H, C, R, N only transport, replay, source, and name the same route. Since no row stores a metric, a selected basis, a host powerset, quotient real-pair equality, or an ambient plane, the listed rows exhaust the local NameCert obligations. □

Theorem 2365.3 (Sorgenfrey plane product-separation route)**.** *Every separation-facing read exported by an accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier factors through the finite route*

$$L_X, L_Y, O_X, O_Y, Q, B, S, H, C, R, N.$$

The route may distinguish product observations by component lower-limit windows and observable-open truth rows, but it cannot claim a host metric, a normality theorem, a powerset topology, or a selected countable base for the classical Sorgenfrey plane.

Proof. Start with a separation-facing request. The carrier has no direct separation coordinate, so the request must first read one or both component lower-limit line rows and their topology exports.

The product row Q then pairs those component reads. Any product-open evidence must pass through the rectangle ledger B, whose entries are generated from the two lower-limit component windows.

The observable-open row S records the finite truth readback for the generated product observation. Transport and replay through H, C, together with provenance and local naming through R, N, preserve that same componentwise route. Because the carrier lacks coordinates for host metrics, normality proofs, powerset opens, selected countable bases, and quotient real-pair equality, no such claim can be exported by the product-separation route. □

Falsifiable prediction: An accepted $\mathsf{SorgenfreyPlane}^{\uparrow}$ consumer that obtains a product-open or separation-facing read without first exposing both lower-limit component rows, the product-topology row, the rectangle ledger, and the observable-open row refutes the carrier surface.

Independence witness: sorgenfrey-line, product-topology, topology, sierpinski-space: no listed sibling supplies the whole eleven-row packet $P = (L_X, L_Y, Q, O_X, O_Y, B, S, H, C, R, N)$, because the lower-limit line supplies one-dimensional half-open windows, product topology supplies paired rectangular generation, topology supplies indexed-open rows, and Sierpinski space supplies observable-open truth readback, while this chapter binds those rows into one product lower-limit NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SorgenfreyPlane}^{\uparrow}}$.

Constructive story: A $\mathsf{SorgenfreyPlane}^{\uparrow}$ packet is generated from two SorgenfreyLineUp source rows, a ProductTopologyUp row, two TopologyUp component indexed-open rows, a lower-limit rectangle ledger, a SierpinskiSpaceUp observable-open row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SorgenfreyPlane}^{\uparrow}}$, seed)

Scope closed: Definition 2365.1, Definition 2365.2, and Definition 2365.3 seed the finite lower-limit product surface over SorgenfreyLine, ProductTopology, Topology, SierpinskiSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SorgenfreyPlane}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host metrizability, normality, second countability, powerset topology on pairs, quotient real-pair equality, selected bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked component lower-limit admission, product-topology admission, rectangle-ledger exactness, observable-open readback, transport stability, replay stability, and non-escape for BEDC.Derived.SorgenfreyPlaneUp

Theorem 2365.4 (Sorgenfrey plane lower-limit product dependency route)**.** *Every accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *read that exposes plane-level separation, product-open, or nonnormality-facing data must first consume the two lower-limit line rows of chapter 3363 and the product-topology row of chapter 2000. In the carrier notation of Definition 2365.1, the rows* L_X, L_Y, O_X, O_Y, Q, B *are dependency data for the plane-level read; they are not replaceable by a host plane, a selected family of open rectangles, or a powerset topology on* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 2365.1. The two one-dimensional source rows L_X, L_Y are read before any component topology rows O_X, O_Y or rectangle ledger row B is available.

Next read the product-topology row Q. It packages paired lower-limit windows as product evidence, so a plane-level product-open or separation-facing consumer receives the rectangular route only after both component lower-limit sources have been displayed.

The observable-open, transport, replay, provenance, and naming rows S, H, C, R, N preserve this dependency order. Since the carrier has no coordinate for an ambient host plane, selected rectangle family, powerset topology, or quotient equality of real pairs, every accepted plane-level read remains downstream of the lower-limit line and product-topology rows. □

Theorem 2365.5 (Sorgenfrey plane topology sibling route)**.** *Every accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier that exposes a plane topology read is downstream of the indexed-open carrier of chapter 88, the lower-limit line rows of chapter 3363, and the product-topology classifier of chapter 2000. In the notation of Definition 2365.1, the rows* L_X, L_Y, O_X, O_Y, Q, B *show that the carrier is not*

kernel-isolated: it consumes lower-limit topology and product topology classifier rows before any product-open or separation-facing read is public.

Proof. Project the carrier of Definition 2365.1. The component rows L_X, L_Y first expose lower-limit line data, and the rows O_X, O_Y are the topology rows read from those component sources.

Next read the product-topology row Q and the rectangle ledger B. Together with Definition 2365.4, these rows route every plane-level topology observation through paired lower-limit windows and the displayed product-topology classifier.

The structural rows S, H, C, R, N preserve the same dependency order for observable-open readback, transport, replay, provenance, and local naming. Since the carrier has no standalone topology coordinate outside O_X, O_Y, Q, B, no accepted plane topology read can bypass the lower-limit and product-topology siblings. □

Theorem 2365.6 (Sorgenfrey plane TopologyUp dependency)**.** *Every accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier that exposes its plane topology consumes the lower-limit topology and product topology rows through the* $\mathsf{Topology}^{\uparrow}$ *carrier/classifier route of chapter 88, together with the lower-limit source of chapter 3363 and the product classifier of chapter 2000. In the notation of Definition 2365.1, the rows* O_X, O_Y, Q, B *are the visible dependency surface for that read.*

Proof. Project the carrier rows O_X, O_Y, Q, B. The component topology rows O_X, O_Y are the $\mathsf{Topology}^{\uparrow}$ reads exported by the two lower-limit line sources, and Q is the displayed product-topology row.

Use Definition 2365.5 and Definition 2365.4. They force every plane-level topology observation to pass through the component lower-limit windows and then through the product classifier before the rectangle ledger B is public.

Thus the topology dependency is a concrete sibling route through $\mathsf{Topology}^{\uparrow}$, $\mathsf{SorgenfreyLine}^{\uparrow}$, and $\mathsf{ProductTopology}^{\uparrow}$, not a standalone host topology on a plane. □

Theorem 2365.7 (SorgenfreyPlane product-topology forward link)**.** *Every* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *product-open or separation-facing read consumes the* $\mathsf{SorgenfreyLine}^{\uparrow}$ *component rows through chapter 3363, the* $\mathsf{ProductTopology}^{\uparrow}$ *product row through chapter 2000, and the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row through chapter 1860. This route is a concrete sibling edge for the product lower-limit lattice: component half-open windows are paired by the product row and inspected by the observable-open row, with no assertion of metrizability, normality, selected bases, host powerset topology, quotient equality of real pairs, or ambient set-theoretic plane.*

Proof. Project the carrier of Definition 2365.1. The plane packet has two lower-limit component rows, the product topology row, the component indexed-open rows, the rectangle ledger, the observable-open row, transport, replay, provenance, and local naming data.

A product-open read first passes through the $\mathsf{SorgenfreyLine}^{\uparrow}$ component lower-limit rows and then through the $\mathsf{ProductTopology}^{\uparrow}$ row. Only after those rows are visible may the rectangle ledger be read as plane-level product data.

A separation-facing read follows the same route and then exposes the $\mathsf{SierpinskiSpace}^{\uparrow}$ observable-open row. Since every route remains inside the displayed component, product, observable-open, transport, replay, provenance, and local naming rows, no plane-level read bypasses the component lower-limit or product-topology dependencies. □

Theorem 2365.8 (Sorgenfrey plane topology sibling lattice)**.** *Every accepted* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *carrier consumes its topology dependency lattice through the lower-limit component route and the product-topology route before any plane-level open, rectangle, or separation-facing read is exported.*

In the notation of Definition 2365.1, the rows L_X, L_Y, O_X, O_Y, Q, B exhibit the dependency surface: L_X, L_Y name the two lower-limit sources, O_X, O_Y expose their topology rows, Q is the product-topology classifier, and B records the rectangle basis. Thus the carrier depends on the topology sibling of Definition 2365.5, the lower-limit product route of Definition 2365.4, and the product lattice route of Definition 2365.7; it is not a kernel-only packet.

Proof. Project the carrier rows named in Definition 2365.1. The lower-limit line entries L_X, L_Y are read before the component topology rows O_X, O_Y, so the plane topology cannot be formed without the two displayed line sources.

Next read Q and B. The product-topology row Q combines the component topology data, and the rectangle ledger B records the finite product basis used by plane-level reads.

Finally apply the existing sibling routes. Theorems 2365.5, 2365.4, and 2365.7 keep every exported topology or separation read on the displayed lower-limit and product-topology lattice. No additional kernel coordinate can supply a standalone topology for the plane. □

Chapter 2366

A Concrete Naming Certificate for $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$

Following Bishop and Bridges on nested intervals and regular Cauchy real names, the RegularCauchyCantorIntersectio packet records the finite route by which a descending sequence of closed dyadic windows with regular-Cauchy diameter control produces a named intersection witness before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits over the dyadic arithmetic core of chapter 291, the stream-window schedule of chapter 126, the regular rational readback of chapter 301, the real-name boundary of chapter 38, and the tail-estimate packet of chapter 1487. The carrier stores nested-window evidence and a regular Cauchy witness route; it does not import choice, quotient streams, open-cover compactness, selected host limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2366.1 (RegularCauchyCantorIntersection carrier)**.** *A* $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, W, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested closed-window ledger, listing the inspected lower and upper dyadic endpoints and the inclusion check from each visible window to its predecessor;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that records which interval stages and precision requests the packet exposes;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for endpoint comparison, diameter bounds, and midpoint slack at each inspected stage;*
4. *T is the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *tail-control row, stating that late windows have the displayed dyadic diameter bound for the requested precision;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name readback obtained from the nested-window and tail-control rows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after I, W, D, T, R have been replayed;*
7. *H records componentwise* hsame *transport for the nested-window, stream, dyadic, tail-estimate, regular-name, and real-seal rows;*
8. *C records displayed* Cont *replay for the nested-window route and downstream witness handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCantorIntersection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, W, D, T, R, E, H, C, P, N$. There is no coordinate for a countable-choice selector, quotient equality of streams, open-cover compactness, a host-level selected point, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCantorIntersectionUp.RegularCauchyCantorIntersectionTasteGate`
`single carrier alignment`

Theorem 2366.2 (RegularCauchyCantorIntersection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ *carrier $K = (I, W, D, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCantorInter}}$ *surface consists exactly of carrier admission, nested-window inclusion, dyadic endpoint and diameter exactness, tail-estimate handoff, regular-Cauchy readback, real-seal nonescape, componentwise transport, displayed replay, package provenance, local naming, and refusal of choice selectors, quotient streams, open-cover compactness, host selected limits, trichotomy, and ambient real completeness.*

Proof. Project the displayed carrier. The only interval data are the nested ledger I, the visible stage schedule W, and the dyadic endpoint ledger D, so carrier admission and nested-window inclusion are read from those rows.

The tail-control and regular-name obligations are T and R. The tail-estimate row supplies the diameter budget for late windows, and the regular-name row reads that budget as the Cauchy witness consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

The real-seal row E is downstream of the nested-window and regular-name rows. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the same route and add no selector, quotient, compactness, host limit, trichotomy, or ambient-completeness coordinate. □

Theorem 2366.3 (RegularCauchyCantorIntersection nested-window route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ *carrier factors through the finite route*

$$I, W \longmapsto D \longmapsto T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the intersection witness is read only after the nested closed-window ledger, finite stream schedule, dyadic endpoint and diameter checks, regular tail estimate, and regular rational name have been displayed. The route supplies a constructive regular-Cauchy witness and does not supply countable choice, quotient-stream equality, open-cover compactness, a selected host limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 2366.1. The rows I and W expose only the inspected closed windows and the finite schedule of stages visible to the request.

Read the dyadic ledger D next. It supplies endpoint comparison and diameter bounds for the same windows, so the consumer cannot bypass the dyadic layer and jump directly to a real point.

Apply the tail-estimate row T and the regular-name row R. Together they turn the shrinking-window evidence into a displayed regular Cauchy readback before the real seal E is available.

Finally inspect H, C, P, N. These structural rows transport, replay, source, and name the same finite route. Since the carrier has no selector, quotient, open-cover, host-limit, trichotomy, or ambient-completeness row, every accepted consumer remains inside the displayed nested-window route. □

Theorem 2366.4 (Regular Cauchy Cantor intersection tail schedule)**.** *For every accepted* RegularCauchyCantorInte *carrier*

$$K = (I, W, D, T, R, E, H, C, P, N),$$

the intersection witness is readable only after a finite $\mathsf{StreamName}^{\uparrow}$ *stage schedule has exposed the inspected nested windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and diameter checks have been replayed, the* $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ *row has handed its tail budget to* $\mathsf{RegSeqRat}^{\uparrow}$*, and the resulting regular readback has reached the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The same read must also pass through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCantorIntersection}^{\uparrow}}$ *row. No quotient stream, unbounded schedule, selected host point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can shorten that tail schedule.*

Proof. Project the accepted carrier. The rows $I, W, D, T, R, E, H, C, P, N$ are the whole displayed packet, and the finite stage information is stored only in the $\mathsf{StreamName}^{\uparrow}$ row W together with the nested-window ledger I.

Use the NameCert obligations of Definition 2366.2. They force endpoint exactness and diameter control through the dyadic ledger D, and they place the tail-estimate handoff in T before any regular rational readback is consumed.

Now apply the nested-window route of Definition 2366.3. It orders the accepted read as

$$I, W \longmapsto D \longmapsto T \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the schedule is finite because every displayed stage is named by W before the route reaches R or E. Since the carrier has no row for quotient streams, host selected points, unbounded schedules, or ambient completeness, the accepted intersection witness is readable only through the finite tail schedule. □

Falsifiable prediction: A $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ read is refuted whenever a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the intersection witness without displaying the nested closed-window ledger, the finite StreamName schedule, the DyadicRatCore endpoint and diameter checks, the RegularCauchyTailEstimate row, the RegSeqRat readback, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, regularcauchytailestimate: none alone is bijective with $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ because the packet jointly records nested closed-window inclusion, dyadic diameter control, finite stage scheduling, tail-estimate handoff, regular-name readback, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCantorIntersection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyCantorIntersection}^{\uparrow}$ packet is generated from a nested closed-window ledger, a finite StreamName stage schedule, a DyadicRatCore endpoint and diameter ledger, a RegularCauchyTailEstimate row, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyCantorIntersection}^{\uparrow}}$, seed)

Scope closed: Definition 2366.1, Definition 2366.2, and Definition 2366.3 seed the nested closed-window route from regular Cauchy tail control to Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyCantorIntersection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient streams, open-cover compactness, selected host limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-window inclusion, dyadic endpoint exactness, tail-estimate handoff, regular-name readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCantorIntersectionUp.

Chapter 2367

A Concrete Naming Certificate for $\mathsf{HausdorffUniformCompletion}^{\uparrow}$

Following Kelley on uniform spaces and Bishop–Bridges constructive completion, the $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ packet records the separated uniform-completion boundary required when a completion route must expose both Cauchy-filter data and Hausdorff uniqueness data as finite rows. It sits between the uniform completion carrier of chapter 1473, the separated completion seal of chapter 402, the Cauchy-filter source of chapter 333, the Cauchy-net witness surface of chapter 589, and the separated uniform-space packet of chapter 1876. The packet names the point at which the L10 completion route may read separatedness explicitly rather than treating it as implicit in uniform completion.

Definition 2367.1 (Hausdorff uniform completion carrier)**.** *A* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, D, U, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row fixing the completion-facing filter request.*
2. *D is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed witness row extracted from the same request.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *universal-extension row, read only after F and D have displayed the completion source.*
4. *S is the Hausdorff and separated classifier row, read through* $\mathsf{HausdorffCompletion}^{\uparrow}$ *and the separated uniform-space boundary, which compares two completion reads without introducing a quotient representative.*
5. *H records componentwise* hsame *transport for F, D, U, S, C, P, N.*
6. *C records displayed* Cont *replay from completion consumers through the same finite route.*
7. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one separated uniform-completion packet. It contains no coordinate for a quotient completion object, selected representative, host equality of completed points, countable choice, or ambient complete uniform space.

Lean4 ✓ BEDC.Derived.HausdorffUniformCompletionUp.HausdorffUniformCompletionTasteGate single carrier alignment

Theorem 2367.2 (Hausdorff uniform completion NameCert obligations)**.** *For an accepted* HausdorffUniformComple *carrier* $K = (F, D, U, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformCompletion}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *carrier admission for the displayed Cauchy-filter row, Cauchy-net row, uniform-completion universal row, separated classifier row, transport row, replay row, provenance row, and local naming row;*
2. *filter-net exactness, requiring every completion-facing read to pass through* F *and then through the finite directed witness row* D *before* U *is available;*
3. *separated classifier stability, requiring* S *to compare completed reads only through the displayed* $\mathsf{HausdorffCompletion}^{\uparrow}$ *and separated uniform-space rows;*
4. *transport and replay stability, requiring* H *and* C *to preserve the same row order under componentwise sameness and displayed continuation replay;*
5. *ledger non-escape, requiring* P *and* N *to name no quotient completion, selected representative, host equality, countable-choice schedule, or ambient complete uniform-space coordinate.*

No separated uniform-completion obligation is stored outside these finite rows.

Proof. Carrier admission is projection from Definition 2367.1. The projected coordinates are exactly the filter row F, net row D, universal row U, separated classifier row S, and structural rows H, C, P, N.

Filter-net exactness reads the same finite route as Definition 1473.7. The filter row fixes the request, and the directed-net row supplies the finite witness before any universal-extension comparison is visible.

Separated classifier stability is the role of S. The comparison can cite the Hausdorff completion boundary of chapter 402 and the separated uniform-space packet of chapter 1876, but it receives no quotient class or selected representative.

The transport and replay rows H and C preserve that same route. Since P and N package and name only the displayed packet, every excluded object is outside the carrier and outside the continuation ledger. □

Theorem 2367.3 (Hausdorff uniform completion separation handoff)**.** *Every separated uniform-completion uniqueness read from an accepted* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *carrier*

$$K = (F, D, U, S, H, C, P, N)$$

must first expose the $\mathsf{CauchyFilter}^{\uparrow}$ *source row* F*, the* $\mathsf{CauchyNet}^{\uparrow}$ *directed witness row* D*, the* $\mathsf{UniformCompletion}^{\uparrow}$ *universal row* U*, the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separation row inside* S*, and the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *classifier inside the same separated row. The read therefore cannot export quotient completion classes, selected representatives, countable choice, or host equality.*

Proof. Project the carrier of Definition 2367.1. The rows F, D, U, S, H, C, P, N are the only coordinates available to a separated uniform-completion consumer.

The separated classifier S is downstream of the filter row F, directed witness row D, and universal-completion row U. Thus uniqueness can be read only after the Cauchy-filter request, Cauchy-net witness, and uniform-completion universal row have already been displayed.

Finally read H, C, P, N as transport, replay, provenance, and name preservation for the same route. These structural rows preserve the separated handoff; they do not introduce quotient completion classes, selected representatives, countable-choice schedules, or host equality. □

Theorem 2367.4 (Hausdorff uniform completion filter-net coincidence)**.** *Let $K = (F, D, U, S, H, C, P, N)$ be an accepted* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *carrier presenting both the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F and the* $\mathsf{CauchyNet}^{\uparrow}$ *row D for the same uniform-completion request U. Every separated uniqueness read from either presentation factors through the same route*

$$S, H, C, P, N.$$

Thus the filter side and the net side cannot create a second completion point, quotient filter/net equality, a selected representative, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 2367.1. The accepted packet contains one Cauchy-filter source row F, one Cauchy-net witness row D, one uniform-completion request row U, and the shared separated and structural rows S, H, C, P, N.

Use the NameCert obligations of Definition 2367.2. Filter-net exactness places both presentations before U, while separated classifier stability makes S the only row from which uniqueness can be read.

Finally apply the separation handoff of Definition 2367.3. Since both sides reach the same S, H, C, P, N route, neither presentation exports a second completion point, quotient equality between filters and nets, selected representatives, host equality, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer reads $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, and a Hausdorff separated classifier row before citing separated uniform-completion uniqueness.

Independence witness: cauchyfilter, cauchynet, uniformcompletion, hausdorffcompletion, separateduniformspace: $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the filter request, directed witness, universal-extension route, separated classifier, transport, replay, provenance, and local NameCert rows as one boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2367.1 and Definition 2367.2 seed the separated uniform-completion boundary over displayed Cauchy-filter, Cauchy-net, uniform-completion, Hausdorff classifier, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient completion equality, selected representatives, host equality, countable choice, or ambient complete uniform-space existence.

Upgrade path: Obligation closure requires checked carrier admission, filter-net exactness, separated classifier stability, transport and replay stability, and ledger non-escape for BEDC.Derived.HausdorffUniformCompletionUp.

Constructive story: A $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ packet is generated from CauchyFilter, CauchyNet, UniformCompletion, HausdorffCompletion, SeparatedUniformSpace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2368

A Concrete Naming Certificate for $\mathsf{WrrStatePhaseSpace}^{\uparrow}$

The $\mathsf{WrrStatePhaseSpace}^{\uparrow}$ packet realizes the WRR hotspot state-vector branch of chapter 9462 and the source analysis in subsection 9462.28.21 as finite NameCert data. It sits beside the codon-window carrier of chapter 2234, the transition matrix of chapter 2245, the core state graph of chapter 2306, and the finite observation fold of chapter 2236. Those siblings expose codon windows, transition rows, graph compression, and finite observation folding; this packet binds the WRR hotspot state vector, invariant anchors, active support, reassignment preorder, table provenance, transport, replay, and local naming as one concrete finite phase-space object.

Definition 2368.1 (WRR state phase-space carrier)**.** *A* $\mathsf{WrrStatePhaseSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, A, V, R, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the state-vector ledger recording the eight-coordinate C_{WRR} assignment vector of Definition 9462.1794.*
2. *A is the invariant-anchor ledger for the* UGG *and* AAG *coordinates of Definition 9462.1796.*
3. *V is the active-support row naming the six moving WRR coordinates of Definition 9462.1797.*
4. *R is the reassignment-preorder row recording the finite inclusions of Definition 9462.1802.*
5. *T is the table-provenance row linking each state vector to its curated genetic-code table identifier and comparison-family ledger.*
6. *H records componentwise* hsame *transport for the state vector, anchors, active support, preorder, table provenance, replay, package, and local naming rows.*
7. *C records displayed* Cont *replay from a hotspot-state consumer through S, A, V, R, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{WrrStatePhaseSpace}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{CoreStateGraph}^{\uparrow}$, $\mathsf{FiniteObservationFold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WrrStatePhaseSpace}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite WRR hotspot state phase space. It does not assert a biochemical time parameter, an evolutionary tree, a complete biological state space, or a dynamics law outside the displayed comparison-family ledger.

Theorem 2368.2 (WRR state phase-space NameCert obligations)**.** *For an accepted* $\mathsf{WrrStatePhaseSpace}^{\uparrow}$ *carrier* $W = (S, A, V, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WrrStatePhaseSpace}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the state-vector ledger, invariant-anchor ledger, active-support row, reassignment-preorder row, table-provenance row, componentwise transport, replay, package provenance, and local naming row;*
2. *state-vector exactness, requiring every hotspot-state read to use the coordinate order fixed by* S *and the WRR cube source rather than an unnamed codon projection;*
3. *anchor and support stability, requiring* A *and* V *to keep invariant anchors separate from active reassignment coordinates under* H *and* C*;*
4. *preorder provenance, requiring each inclusion in* R *to be sourced by finite table identifiers in* T *and replayed only through the same comparison-family ledger;*
5. *ledger non-escape, requiring* P *and* N *to export no temporal dynamics, no causal reassignment rule, no evolutionary tree, no unobserved-state impossibility claim, and no context-free translation model.*

Lean follow-up must provide $\mathsf{Nontrivial\ WrrStatePhaseSpace}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ WrrStatePhaseSpace}^{\uparrow}$ *instances in* BEDC.Derived.WrrStatePhaseSpaceUp.TasteGate *for this displayed finite carrier.*

Proof. Carrier admission is projection from Definition 2368.1. The tuple displays exactly the state-vector, anchor, support, preorder, table-provenance, transport, replay, provenance, and naming rows.

State-vector exactness reads the WRR coordinate order from the source definition and stores it in S. The carrier has no second codon projection row, so a hotspot-state consumer cannot silently change the coordinate order.

Anchor and support stability use the split between A and V. The invariant anchors remain named separately from the active support, while H and C transport and replay the same split rather than merging it into a single untyped state list.

Preorder provenance is the role of R and T. Each displayed inclusion is tied to finite table identifiers before it is packaged, so the preorder is an observed comparison-family row and not a temporal or causal law.

The remaining claim is ledger non-escape. Since temporal dynamics, causal reassignment mechanisms, evolutionary-tree structure, exhaustive biological state space, and ribosome-context models are not carrier coordinates, no NameCert consumer can read them from W. □

Lean4 ✓ `BEDC.Derived.WrrStatePhaseSpaceUp.WrrStatePhaseSpaceTasteGate single carrier alignment`

Falsifiable prediction: Every accepted $\mathsf{WrrStatePhaseSpace}^{\uparrow}$ consumer reads a WRR state-vector ledger, invariant-anchor ledger, active-support row, reassignment-preorder row, table-provenance row, hsame transport, Cont replay, Pkg provenance, and local naming before citing hotspot phase-space compression.

Independence witness: codon-six-bit-window, codon-transition-matrix, core-state-graph, finite-observation-fold; $\mathsf{WrrStatePhaseSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state vectors, invariant anchors, active support, reassignment preorder, table provenance, transport, replay, package provenance, and local NameCert rows as one hotspot phase-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WrrStatePhaseSpace}^{\uparrow}}$ **.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WrrStatePhaseSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2368.1 and Definition 2368.2 seed the finite WRR hotspot phase-space surface over state-vector, invariant-anchor, active-support, reassignment-preorder, table-provenance, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{WrrStatePhaseSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, temporal biological dynamics, causal reassignment mechanisms, evolutionary-tree reconstruction, exhaustive biological state-space closure, or context-free translation dynamics.

Upgrade path: Obligation closure requires checked carrier admission, state-vector exactness, anchor and support stability, preorder provenance, ledger non-escape, and TasteGate nontriviality plus field faithfulness for BEDC.Derived.WrrStatePhaseSpaceUp.

Constructive story: A WrrStatePhaseSpace$^{\uparrow}$ packet is generated from WRR state-vector, invariant-anchor, active-support, reassignment-preorder, table-provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2369

A Concrete Naming Certificate for $\mathsf{SigmaCompact}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{SigmaCompact}^{\uparrow}$ packet records a space by a displayed compact exhaustion rather than by an open-cover compactness principle or a choice-selected cover. It sits as an L10 neighbour of the local compactness certificate of chapter 1726, the proper closed-ball compactness route of **??**, the center-radius window of chapter 1451, the compact metric witness of chapter 129, and the stream-indexed naming surface of chapter 126. The packet is finite at every public read: a consumer sees a named exhaustion prefix, its compact rows, and the replay ledger that justifies the prefix, not a completed countable family supplied by an ambient host.

Definition 2369.1 (SigmaCompact carrier)**.** *A* $\mathsf{SigmaCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, E, K, B, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric or topological source row whose points and comparison data are already carried by finite BEDC histories.*
2. *E is a* $\mathsf{StreamName}^{\uparrow}$ *exhaustion-prefix row. Its displayed entries name compact windows $E(0), \ldots, E(m)$ and a monotone inclusion ledger between consecutive entries.*
3. *K assigns a* $\mathsf{CompactMetric}^{\uparrow}$ *witness to each displayed exhaustion entry $E(i)$.*
4. *B is the optional* $\mathsf{MetricClosedBall}^{\uparrow}$ *boundary row for those entries that arise from center-radius metric windows.*
5. *L is the optional* $\mathsf{LocallyCompact}^{\uparrow}$ *support row, used only when a local compact neighbourhood certificate is the source of one displayed exhaustion window.*
6. *H records componentwise* hsame *transport for the source row, exhaustion prefix, compact witnesses, closed-ball boundary rows, local compact support rows, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite exhaustion request through X, E, K, B, L, H and into a compactness, properness, completion, or stream consumer.*
8. *P is the* Pkg *provenance over* $\mathsf{SigmaCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{SigmaCompact}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these nine rows. It has no coordinate for open-cover compactness, countable choice, a host-selected exhaustion, quotient point equality, or an ambient locally compact or proper metric structure beyond the displayed handoff rows.

Theorem 2369.2 (SigmaCompact NameCert obligations)**.** *For every accepted* $\mathsf{SigmaCompact}^\uparrow$ *carrier* $S = (X, E, K, B, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SigmaCompact}^\uparrow}$ *surface consists exactly of carrier admission, exhaustion-prefix indexing, compact witness transport for each displayed prefix entry, closed-ball boundary handoff when a row B is present, locally compact support handoff when a row L is present, provenance non-escape, and local naming for the rows $X, E, K, B, L, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.SigmaCompactUp.SigmaCompactTasteGate single carrier alignment`

Proof. Project the carrier of Definition 2369.1. Carrier admission reads the source row, the finite stream prefix, the compact witnesses attached to that prefix, the optional boundary and local compact rows, and the structural rows H, C, P, N.

Exhaustion-prefix indexing is finite. Induct over the displayed list $E(0), \ldots, E(m)$. At each entry the compact witness is the corresponding row in K, and the inclusion into the next displayed window is one of the ledger entries already present in E.

The handoff rows are exact. A closed-ball consumer can read B only for entries that display a $\mathsf{MetricClosedBall}^\uparrow$ boundary, and a local compactness consumer can read L only for entries that display the corresponding $\mathsf{LocallyCompact}^\uparrow$ support.

Provenance and naming are exhausted by P and N. Since the carrier has no coordinate for open covers, countable choice, host-selected exhaustions, quotient point equality, or ambient properness, none of those principles can occur in the local NameCert surface. □

Theorem 2369.3 (SigmaCompact proper metric exhaustion)**.** *Every accepted* $\mathsf{ProperMetric}^\uparrow$ *packet that displays a finite radius-window prefix through* $\mathsf{StreamName}^\uparrow$ *induces a* $\mathsf{SigmaCompact}^\uparrow$ *read over the same metric source by taking each shown window to its* $\mathsf{MetricClosedBall}^\uparrow$ *row and its attached* $\mathsf{CompactMetric}^\uparrow$ *witness. The induced read exports only*

$$X, E, K, B, L, H, C, P, N,$$

with L empty unless a local compactness support row is displayed, and it does not export open-cover compactness, a completed countable exhaustion, selected centers outside the prefix, quotient equality, countable choice, or ambient completeness.

Proof. Start with the proper metric carrier of **??**. For each displayed radius-window entry in the stream prefix, the proper metric packet already exposes a closed-ball row and a compact witness for that same row.

Package the displayed prefix as E. The compact witnesses become K, the closed-ball rows become B, and the optional local compact support is copied into L only when it is already shown. Componentwise transport and replay are inherited as H and C.

The resulting tuple has exactly the carrier form of Definition 2369.1. Its provenance names the proper metric source and the compact, closed-ball, local compact, and stream-name siblings, so a downstream consumer cannot extend the prefix, select hidden centers, or import an open-cover or choice principle. □

Lean4 ✓ `BEDC.Derived.SigmaCompactUp.SigmaCompactCarrier proper metric exhaustion`

Theorem 2369.4 (SigmaCompact compact-exhaustion sibling route)**.** *Every* $\mathsf{SigmaCompact}^{\uparrow}$ *consumer of a displayed compact exhaustion reads the sibling rows* $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *only through the carrier route*

$$X, E, K, B, L, H, C, P, N.$$

The finite exhaustion prefix E *names the displayed windows,* K *supplies the compact witnesses for those windows,* B *and* L *are read only when the corresponding closed-ball or local compact support rows are displayed, and* H, C, P, N *transport, replay, source, and name that same route. The read exports no open-cover compactness, completed countable exhaustion, countable choice, quotient equality, or ambient completeness.*

Proof. Apply Definition 2369.1. The only public coordinates of an accepted compact-exhaustion packet are the source row, stream prefix, compact witnesses, optional closed-ball and local compact rows, and the structural rows H, C, P, N.

Apply Definition 2369.2. It fixes the local NameCert surface to those displayed rows, so a consumer cannot read a locally compact, proper metric, closed-ball, compact metric, or stream-name sibling except by projecting the corresponding coordinate of the carrier.

Finally apply Definition 2369.3. The proper metric route supplies closed-ball and compact witnesses only for the displayed finite prefix. Since neither the carrier nor the proper metric handoff contains an open-cover compactness proof, completed countable exhaustion, countable-choice schedule, quotient equality, or ambient completeness row, the sibling route cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.SigmaCompactUp.SigmaCompactCompactExhaustionSiblingRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{SigmaCompact}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SigmaCompact}^{\uparrow}$ packet is generated from a source row, a finite StreamName exhaustion prefix, CompactMetric witnesses for displayed entries, optional MetricClosedBall and LocallyCompact handoff rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SigmaCompact}^{\uparrow}}$, seed)

Scope closed: Definition 2369.1, Definition 2369.2, and Definition 2369.3 seed the finite compact-exhaustion interface between local compactness, proper metric windows, compact metric witnesses, and stream-indexed names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SigmaCompact}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, a completed countable exhaustion, countable choice, quotient point equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exhaustion-prefix indexing, compact witness projection, proper metric handoff, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.SigmaCompactUp.

Chapter 2370

A Concrete Naming Certificate for $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$

Following Blaschke and Kelley, the $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$ packet names the finite selection surface for a sequence of compact subsets of a compact metric source. It sits over the compact-subset ledger of chapter 1660, the finite-net distance interface of chapter 2086, the compact metric source of chapter 129, the directed convergence route of chapter 1667, and the uniform Cauchy-net boundary of chapter 1657. The packet records finite Hausdorff-distance selection data; it is not a host compactness theorem, an open-cover argument, an ambient powerset quotient, a choice-selected subnet, or an external sequential compactness principle.

Definition 2370.1 (Blaschke selection compact-hyperspace carrier)**.** *A* $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, S, D, W, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, including its* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, finite-net, total-boundedness, and completeness witnesses.*
2. *S is the sequence of compact-subset* $\mathsf{Hyperspace}^{\uparrow}$ *rows over X, with each compact-subset presentation read only through the displayed hyperspace carrier.*
3. *D is the* $\mathsf{HausdorffMetric}^{\uparrow}$ *distance-ledger family comparing the compact-subset rows of S at each requested finite resolution.*
4. *W is the finite Cauchy-subnet window ledger, recording the selected index window, cofinality evidence, Hausdorff-distance requests, and stabilization bounds.*
5. *E is the extracted compact-subset endpoint row, displayed as a* $\mathsf{Hyperspace}^{\uparrow}$ *compact-subset name over the same source X.*
6. *R is the dyadic and* $\mathsf{Real}^{\uparrow}$ *tolerance readback showing that the window ledgers in W and the endpoint row E answer the same finite Hausdorff-distance requests.*
7. *H records componentwise* hsame *transport for the compact metric source, compact-subset sequence, Hausdorff distance ledgers, Cauchy-subnet windows, endpoint row, tolerance readback, provenance, and local naming rows.*

8. *C records displayed* Cont *replay through*

$$X, S, D \longmapsto W \longmapsto E, R.$$

9. *P is the* Pkg *provenance over* BlaschkeSelectionCompactHyperspace$^{\uparrow}$, Hyperspace$^{\uparrow}$, HausdorffMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, NetConvergence$^{\uparrow}$, CauchyUniformNet$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BlaschkeSelectionCompactHyperspace}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $X, S, D, W, E, R, H, C, P, N$. It has no coordinate for open-cover compactness, host powersets, quotient subset equality, selected choice functions, or an unbounded subnet object.

Lean4 ✓ `BEDC.Derived.BlaschkeSelectionCompactHyperspaceUp.BlaschkeSelectionCompactHyperspaceTasteGate single carrier alignment`

Theorem 2370.2 (Blaschke selection compact-hyperspace NameCert obligations)**.** *For every accepted* BlaschkeSelectionCompactHyperspace$^{\uparrow}$ *carrier*

$$B = (X, S, D, W, E, R, H, C, P, N),$$

the local NameCert$_{\text{BlaschkeSelectionCompactHyperspace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source row X, the compact-subset sequence S, the HausdorffMetric distance ledgers D, the finite Cauchy-subnet windows W, the endpoint row E, tolerance readback R, transport, replay, provenance, and local naming rows;*

2. *compact-subset sequence exactness, requiring every displayed member of S to be admitted through* Hyperspace$^{\uparrow}$ *over the same compact metric source X before any selection window can inspect it;*

3. *finite Hausdorff Cauchy-window stability, requiring the ledgers in D and W to share the same finite resolution requests and to route their distance reads through* HausdorffMetric$^{\uparrow}$*;*

4. *endpoint readback stability, requiring E and R to expose the selected compact-subset endpoint only by the displayed* Real$^{\uparrow}$ *tolerance comparisons and componentwise* hsame *transport;*

5. *provenance and name exhaustion, requiring C, P, N to replay, source, and name only the displayed finite route and to export no host compactness theorem, powerset quotient, selected choice function, open-cover proof, or unbounded subnet object.*

No Blaschke selection compact-hyperspace obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2370.1. The packet displays the compact metric source, compact-subset sequence, Hausdorff distance ledgers, Cauchy-subnet windows, endpoint row, tolerance readback, transports, replay, provenance, and local naming row as its only coordinates.

The compact-subset sequence is exact because every member of S is read through the Hyperspace$^{\uparrow}$ surface over the same source X. The Hausdorff reads are exact because D records only finite HausdorffMetric$^{\uparrow}$ ledgers attached to those displayed compact-subset names.

The selection step is the finite-net replay through W. Each selected window exposes a cofinal finite index range and a stabilization ledger, and the route through NetConvergence$^{\uparrow}$ and CauchyUniformNet$^{\uparrow}$ keeps the window evidence inside displayed convergence data.

The endpoint and tolerance readback are then projected through E and R. The endpoint is a compact-subset row over X, while the $\mathsf{Real}^{\uparrow}$ readback states exactly the finite Hausdorff-distance tolerances checked by the window ledgers.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for host compactness, powerset quotients, choice-selected functions, open-cover proofs, or unbounded subnet objects, the five listed obligation rows exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}}$.

Constructive story: A $\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}$ packet is generated from a CompactMetricUp source, a sequence of HyperspaceUp compact-subset rows, HausdorffMetricUp distance ledgers, finite Cauchy-subnet windows, an endpoint compact-subset row, RealUp tolerance readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2370.1 and Definition 2370.2 seed the finite compact-hyperspace selection surface through Hyperspace, HausdorffMetric, CompactMetric, NetConvergence, and CauchyUniformNet rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BlaschkeSelectionCompactHyperspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host compactness, open-cover compactness, powerset quotients, choice-selected functions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-subset sequence exactness, finite Hausdorff Cauchy-window stability, endpoint readback stability, and provenance/name exhaustion for BEDC.Derived.BlaschkeSelectionCompactHyp

Chapter 2371

A Concrete Naming Certificate for $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$

The $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ packet realizes the observation-boundary route implicit in chapter 9390 as finite NameCert data. It sits between the cellular trust substrate of chapter 940, the cellular rule and orbit carrier of chapter 934, the bounded trace boundary of chapter 944, the local read cell of chapter 509, and the continuation automaton surface of chapter 482. Its distinguishing datum is the finite boundary at which a local rule, a bounded orbit window, a manifest row, and an audit replay jointly authorize an observation.

Falsifiable prediction: Removing either the local-rule row or the boundary-manifest row breaks accepted observation replay: without the local rule two windows with different update tables can share the same text observation, and without the manifest the audit route has no finite boundary against which to replay the observation.

Independence witness: cellulartrustsubstrate, cellularautomaton, boundeduniversaltrace, localitycell, continuationautomaton: none is bijective to $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ because each omits at least one of the joint boundary ingredients: local transition table, bounded orbit window, boundary manifest, accepted observation row, or audit replay route.

Definition 2371.1 (Cellular substrate observation-boundary carrier)**.** *A* $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, W, M, B, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the local cellular rule row, including alphabet, neighbourhood arity, boundary convention, and complete transition table.*
2. *W is the bounded orbit-window row, including the initial configuration, finite time interval, displayed cell coordinates, and trace bounds.*
3. *M is the boundary-manifest row recording the accepted observation fields, replay bounds, text schema, and local read discipline.*
4. *B is the accepted observation-boundary row, pairing the displayed local cells with the manifest fields and the orbit-window coordinates they are allowed to inspect.*
5. *A is the audit-replay row, replaying each boundary read from R through W and checking it against M and B.*

6. *H records componentwise* hsame *transport for the local rule, orbit window, manifest, boundary row, audit replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through*

$$R, W, M, B \longmapsto A.$$

8. *P is the* Pkg *provenance over* $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{LocalityCell}^{\uparrow}$, $\mathsf{ContinuationAutomaton}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unbounded orbit, hidden update table, host evaluator identity, detached manifest schema, quotient stream equality, or an all-time induction principle.

Lean4 ✓ `BEDC.Derived.CellularSubstrateObservationBoundaryUp.CellularSubstrateObservationBoundaryTasteGa`
`single carrier alignment`

Theorem 2371.2 (Cellular trust substrate vision route)**.** *For every accepted* $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ *carrier*

$$O = (R, W, M, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}}$ *surface gives a finite route from a local rule and bounded orbit window to a trusted observation boundary. A consumer receives exactly the local-rule row R, bounded orbit-window row W, boundary-manifest row M, accepted observation-boundary row B, audit-replay row A, componentwise transport H, displayed replay C, provenance P, and local naming row N, and receives no unbounded orbit, hidden update table, detached manifest, host evaluator, quotient stream, or all-time induction principle.*

Proof. Carrier admission is projection from Definition 2371.1. The packet displays the local rule, orbit window, manifest, boundary row, audit replay, transport, continuation, provenance, and naming rows as its complete coordinate list.

The finite route starts at R and W. The rule row supplies the transition table and boundary convention, while the orbit-window row fixes the initial configuration, time interval, and cell coordinates that may be inspected.

The observation boundary is then exposed by M and B. The manifest states the accepted schema and replay bounds, and the boundary row ties each observation field to the local cells and orbit coordinates it is allowed to read.

The audit replay A checks the route back through R, W, M, B. Componentwise transport through H and displayed replay through C preserve the same boundary route under accepted comparison.

Finally P and N source and name only this displayed packet. Since no row carries an unbounded orbit, hidden update table, detached manifest, host evaluator, quotient stream, or all-time induction principle, the trusted observation boundary is exactly the finite NameCert route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ packet is generated from a local cellular rule, bounded orbit window, boundary manifest, accepted observation-boundary row, audit replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2371.1 and Definition 2371.2 seed the finite route from local rule and bounded orbit window to a trusted observation boundary.

Formal verification: FormalStatus(NameCert$_{\text{CellularSubstrateObservationBoundary}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CellularSubstrateObservationBoundaryUp.CellularSubstrateObservationBoundaryTasteGate\_s`

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, host evaluator correctness, quotient streams, cellular completeness, all-time induction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-rule accountability, bounded orbit-window exactness, manifest-boundary replay, and provenance/name exhaustion for BEDC.Derived.CellularSubstrateObservationBoundaryUp; Lean follow-up must provide Nontrivial and FieldFaithful TasteGate instances.

Chapter 2372

A Concrete Naming Certificate for $\mathsf{SequentialCompact}^{\uparrow}$

Following Bishop and Bridges constructive compactness and Brouwer choice-sequence practice, the $\mathsf{SequentialCompact}^{\uparrow}$ packet records sequential compactness only as finite subsequence-window data. It sits over the compact metric source of chapter 129, the Baire-space inspection discipline of chapter 334, the finite stream interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet gives compact-metric and Baire-space completion consumers a displayed subsequence route; it is not a host sequential compactness theorem, not countable choice, and not a quotient stream principle.

Definition 2372.1 (SequentialCompact carrier)**.** *A* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, B, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite cover and metric inspection data bound the sequence consumer.*
2. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *cylinder discipline row used when a downstream completion route reads finite choice-sequence windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, recording the inspected indices and finite request schedule.*
4. *W is the finite subsequence-window row: selected indices, selected source windows, and the displayed monotonicity evidence for the selected index list.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected windows and their rational tail budgets.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback is accepted.*
7. *H records componentwise* hsame *transports for K, B, S, W, R, E.*
8. *C records displayed* Cont *replays from compact-metric source reads to stream windows, Baire cylinders, subsequence windows, regular rational readbacks, and the terminal real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, S, W, R, E, H, C, P, N$. It has no coordinate for an externally chosen subsequence, quotient equality of streams, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier local namecert surface`

Theorem 2372.2 (SequentialCompact NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$ *surface consists exactly of finite rows for carrier admission, compact-metric source exposure, Baire-cylinder exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by host sequential compactness, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.TasteGate.SequentialCompactTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 2372.1. The accepted packet exposes exactly the compact metric source, Baire cylinder row, stream window, subsequence window, regular rational readback, Real seal, transport, continuation, provenance, and local naming rows.

The compactness and choice-sequence reads are finite. The $\mathsf{CompactMetric}^{\uparrow}$ row supplies only displayed metric and cover data, while the $\mathsf{BaireSpace}^{\uparrow}$ row supplies only finite cylinder access before any selected window is consumed.

The subsequence evidence is the row W. Its selected indices, selected source windows, and monotonicity evidence are packet data, and the hsame and Cont rows transport and replay those same data without adding a second selection source.

The readback then passes through R and E. Since neither row has a coordinate for host sequential compactness, countable choice, quotient stream equality, an external subsequence, or ambient Real completeness, the local NameCert surface is exhausted by the displayed finite rows. □

Theorem 2372.3 (SequentialCompact obligation readiness)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the checked TasteGate carrier-alignment marker above makes the next obligation surface exactly the finite-window classifier exposure for $K, B, S, W, R, E, H, C, P, N$, subsequence-window monotonicity in W, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, and refusal of any host subsequence selector, quotient stream equality, countable choice, open-cover compactness theorem, or ambient Real completeness row.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier obligation readiness`

Proof. Start from Definition 2372.2 and the displayed TasteGate carrier-alignment marker attached to it. The checked alignment pins the TasteGate carrier to the same tuple $K, B, S, W, R, E, H, C, P, N$ used by the paper surface.

The finite-window classifier exposure is therefore not an additional carrier. It is the componentwise classifier read of the existing compact-metric source, Baire cylinder row, stream window,

subsequence window, regular readback, real seal, transport, continuation, provenance, and local naming rows.

The remaining rows are forced by dependency order. The subsequence-window monotonicity obligation is the monotonicity evidence inside W; the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is R; and the terminal $\mathsf{Real}^{\uparrow}$ nonescape row is E.

Because none of the aligned carrier coordinates contains a host subsequence selector, quotient stream equality, countable choice, open-cover compactness theorem, or ambient real-completeness principle, those refusals are part of the exact obligation surface for the next closure step. □

Theorem 2372.4 (SequentialCompact finite-window cluster obligation)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the finite-window cluster obligation binds the $\mathsf{CompactMetric}^{\uparrow}$ *row* K*,* $\mathsf{BaireSpace}^{\uparrow}$ *row* B*,* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, and* $\mathsf{Real}^{\uparrow}$ *seal row* E *to the selected finite-window route* W *inside the same packet. A cluster-facing consumer must read* $K, B, S, W, R, E, H, C, P, N$ *in the displayed surface and receives no host subsequence choice, quotient stream equality, open-cover compactness theorem, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactFormalTargetScope`

Proof. Start from the obligation readiness surface. It already identifies $K, B, S, W, R, E, H, C, P, N$ as the finite rows that can be consumed by the next closure step.

The selected cluster data are the finite row W. The compact metric source K, Baire cylinder discipline B, and stream-name schedule S are read before W, while R and E carry the regular rational and real-name endpoint.

The structural rows H, C, P, N preserve and name that same route. Since no coordinate of the packet names host subsequence choice, quotient stream equality, open-cover compactness, or ambient Real completeness, the cluster obligation stays finite-window local. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactFiniteWindowClusterObligation`

Theorem 2372.5 (SequentialCompact StreamName subsequence route)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier* $Q = (K, B, S, W, R, E, H, C, P, N)$*, every subsequence route read by a* $\mathsf{StreamName}^{\uparrow}$ *consumer factors through the source-window row* S *and the selected finite-window row* W *before it reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The route is a displayed finite-window cluster route, not a host subsequence selector, quotient stream equality, countable-choice witness, or open-cover compactness principle.*

Proof. Read S as the finite source-window schedule. The selected indices and selected windows are then displayed in W, so the route is already internal to the packet before any readback occurs.

Next pass through R and E. These rows expose regular rational readback and the terminal real seal, but they do not add stream quotienting or ambient completeness.

Finally use transport, replay, provenance, and local naming. The rows H, C, P, N keep the same S, W, R, E route visible to the consumer and do not create a separate selection principle. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactStreamNameSubsequenceRoute`

Theorem 2372.6 (SequentialCompact Real cluster nonescape)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier* $Q = (K, B, S, W, R, E, H, C, P, N)$*, any cluster-facing read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal row* E *must first expose the compact metric source* K*, the Baire cylinder row* B*, the stream-name row*

S, the selected finite-window row W, and the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The seal exports only the displayed cluster route and no ambient Real completeness, hidden Real limit selector, quotient stream equality, host subsequence choice, or open-cover compactness.*

Proof. The carrier places E downstream of the finite route. A consumer reaches it only after the compact metric, Baire, stream-name, selected-window, and regular rational rows have been displayed.

Apply the readiness theorem to identify the exclusions. It refuses host subsequence selectors, quotient stream equality, countable choice, open-cover compactness, and ambient Real completeness as rows of the packet.

Thus the real seal is a terminal observation of the finite-window cluster route. It cannot be used as a separate completeness theorem or as a hidden limit-selection device. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier real cluster nonescape`

Theorem 2372.7 (SequentialCompact public cluster export)**.** *Every public cluster-facing consumer of an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet receives exactly the route*

$$K, B, S, W, R, E, H, C, P, N.$$

Here K is the $\mathsf{CompactMetric}^{\uparrow}$ *source row, B is the* $\mathsf{BaireSpace}^{\uparrow}$ *cylinder row, S and W are the* $\mathsf{StreamName}^{\uparrow}$ *source-window and selected-window rows, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The export is public only as this finite route: it refuses host subsequence choice, quotient stream equality, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactPublicClusterExport`

Proof. Start with the obligation readiness and finite-window cluster obligation. They expose the accepted carrier only through $K, B, S, W, R, E, H, C, P, N$ and keep the cluster obligation inside the displayed finite windows.

Next use the $\mathsf{StreamName}^{\uparrow}$ subsequence route. The rows S and W name the source and selected windows before any $\mathsf{RegSeqRat}^{\uparrow}$ readback can be consumed, so no host selector or quotient stream equality is introduced.

Finally apply the Real cluster nonescape theorem together with the public finite cluster route. The readback row R precedes the seal E, and H, C, P, N only transport, replay, source, and name the same finite packet. Thus the public consumer receives exactly the displayed route and none of the refused choice, compactness, quotient, or ambient-completeness principles. □

Theorem 2372.8 (SequentialCompact root obligation Baire window)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every root obligation that consumes selected finite windows must expose the $\mathsf{BaireSpace}^{\uparrow}$ *cylinder row B before the selected-window row W. The selected indices and selected source windows remain data of W inside the same* BHist *packet, transported by H, replayed by C, and named by N; they are not supplied by an external subsequence selector or by countable choice.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompact root obligation baire window`

Proof. Start from Definition 2372.3. It identifies the next obligation surface with the displayed rows $K, B, S, W, R, E, H, C, P, N$ and with the finite-window classifier exposure of those rows.

Apply Definition 2372.16. The handoff order places the compact-metric source K, the Baire cylinder row B, and the stream-window row S before the selected-window row W is consumed.

The structural rows then preserve that order. Transport H, replay C, provenance P, and local naming N keep the selected finite windows inside the displayed packet, so the root obligation cannot start from an external selector or a countable-choice schedule. □

Theorem 2372.9 (SequentialCompact root obligation Real seal)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every root obligation that reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal* E *must first consume* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*. The seal is therefore downstream of* K, B, S, W, R *and of the structural rows* H, C, P, N*, and it cannot be used as an ambient Real-completeness theorem, selected Real limit, quotient stream equality, or host subsequence principle.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier root obligation real seal`

Proof. Use Definition 2372.3 to read the root obligation surface. The readiness theorem names $\mathsf{RegSeqRat}^{\uparrow}$ handoff through R and $\mathsf{Real}^{\uparrow}$ seal nonescape through E as rows of the same finite carrier.

Apply Definition 2372.16. It routes selected finite windows through W before regular rational readback in R and before the terminal seal in E.

Finally use the existing nonescape route, including Definition 2372.20. The row E remains an endpoint of the displayed handoff, not a source of ambient completeness, selected limits, quotient equality, or host subsequence data. □

Theorem 2372.10 (SequentialCompact root obligation nonescape)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the root obligation surface consists only of the displayed rows $K, B, S, W, R, E, H, C, P, N$*. It has no row for quotient stream equality, externally selected subsequences, countable choice, open-cover compactness, selected Real limits, host sequential compactness, or ambient Real completeness.*

Proof. Project the carrier and apply Definition 2372.2. The local certificate surface is exactly the finite tuple $K, B, S, W, R, E, H, C, P, N$ with compact-metric source exposure, Baire-cylinder exposure, stream-window inspection, selected-window monotonicity, regular readback, Real seal nonescape, transport, replay, provenance, and local naming.

Use Definition 2372.3. It states that the aligned root surface refuses host subsequence selectors, quotient stream equality, countable choice, open-cover compactness, and ambient Real completeness as obligation rows.

The finite-subsequence handoff keeps the same refusal through the downstream route. Since H, C, P, N only preserve and name the displayed rows, the root obligation surface cannot export any of the excluded host or quotient principles. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRootObligationNonescape`

Theorem 2372.11 (SequentialCompact Baire cylinder exposure)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

any compact-metric completion consumer must read the $\mathsf{BaireSpace}^{\uparrow}$ *cylinder row* B *before it reads the selected finite-window row* W*. The row* B *is part of the same carrier surface as* $K, S, W, R, E, H, C, P, N$*, and the consumer may not replace it by a host choice-sequence branch, open-cover compactness theorem, countable-choice selector, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactBaireCylinderExposure`

Proof. Start from Definition 2372.2. The obligation surface includes Baire-cylinder exposure as a named row of the accepted packet, so B is visible before any downstream read can treat selected windows as evidence.

Apply Definition 2372.16. Its handoff order places $\mathsf{BaireSpace}^{\uparrow}$ before $\mathsf{StreamName}^{\uparrow}$ and before the finite subsequence-window row. Therefore a compact-metric completion consumer cannot inspect W without first passing through B.

Finally use Definition 2372.3. The readiness surface refuses host subsequence selection, quotient stream equality, countable choice, open-cover compactness, and ambient Real completeness, so the Baire exposure remains the displayed finite cylinder row. □

Theorem 2372.12 (SequentialCompact window transport)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the componentwise hsame *transport row* H *preserves the source stream row* S*, selected-window row* W*, regular readback row* R*, and terminal seal row* E *before any completion consumer reads the route. Transport cannot introduce a second selector, quotient stream equality, countable-choice witness, open-cover compactness theorem, or ambient real-completeness coordinate.*

Proof. Project the carrier in Definition 2372.1. The row H records componentwise transport exactly for the displayed analytic rows K, B, S, W, R, E.

Use Definition 2372.2 to identify the local certificate surface. Transport is one of its structural rows and acts on the existing carrier coordinates rather than on a hidden host sequence.

Apply Definition 2372.16. Since S, W, R, E are read in that order by completion consumers, the transported route preserves the same order and exports none of the excluded host principles. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier window transport`

Theorem 2372.13 (SequentialCompact selector refusal)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

there is no row naming a host subsequence selection function, countable-choice principle, quotient stream equality, or open-cover compactness theorem. Every selected-window read is the displayed finite row W *after the compact-metric source row* K*, Baire cylinder row* B*, and stream-window row* S *have been exposed.*

Proof. Begin with the carrier definition. The tuple contains W as finite selected-window data, but it contains no coordinate for a host selector, countable-choice object, quotient stream equality, or open-cover compactness theorem.

Apply Definition 2372.2. The local NameCert surface is exhausted by the displayed rows $K, B, S, W, R, E, H, C, P, N$, so no additional selector row can be supplied by the certificate.

Finally use Definition 2372.16. Any completion-facing consumer reaches W only through the finite handoff order, and the subsequent readback and seal rows preserve that refusal. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactSelectorRefusal`

Theorem 2372.14 (SequentialCompact public finite cluster route)**.** *Every public cluster read exported by an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet factors through the finite row route*

$$K, B, S, W, R, E, H, C, P, N.$$

The consumer reads $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{BaireSpace}^{\uparrow}$ *rows before the selected finite-window row, reads* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and receives no host subsequence choice, countable choice, quotient stream equality, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with selector refusal, Definition 2372.13. It keeps every selected-window read inside the displayed row W after the compact-metric source K, Baire cylinder B, and stream-window row S have been exposed.

Next use the finite subsequence handoff, Definition 2372.16. The public cluster route therefore reads K, B, S, W in order before any regular-rational or Real-facing conclusion can be inspected.

Finally apply Definition 2372.6. The RegSeqRat readback row R precedes the terminal seal E, while H, C, P, N only transport, replay, source, and name the same finite packet. Since none of these rows contains host subsequence choice, countable choice, quotient stream equality, open-cover compactness, or ambient Real completeness, the public cluster read is exactly the stated finite route. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactPublicFiniteClusterRoute`

Theorem 2372.15 (SequentialCompact public NameCert route)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every public $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$ *consumer must read the same finite rows* $K, B, S, W, R, E, H, C, P, N$ *through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* $\mathsf{BaireSpace}^{\uparrow}$ *cylinder row,* $\mathsf{StreamName}^{\uparrow}$ *source-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before using the local NameCert row* N*. The public route is a finite* BHist *route over those displayed rows; it does not provide host subsequence choice, quotient stream equality, open-cover compactness, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactPublicNameCertCarrierRows`

Proof. Start with the finite public cluster surface of Definition 2372.14. It already forces public cluster reads through $K, B, S, W, R, E, H, C, P, N$ and refuses the host selector, quotient, choice, open-cover, and Real-completeness exits.

Next use the source and window boundaries. The $\mathsf{StreamName}^{\uparrow}$ subsequence route of Definition 2372.5, the Baire exposure of Definition 2372.11, and the root selected-window route of Definition 2372.30 place B, S, W before any public NameCert consumer can use selected-window data.

Now read the downstream seal route. Apply Definition 2372.6, Definition 2372.33, and Definition 2372.10. They force the RegSeqRat row R and terminal Real seal E to remain finite-window

observations, not hidden subsequence objects, quotient stream equalities, open-cover compactness arguments, countable-choice schedules, or ambient completeness principles.

Finally use H, C, P, N only as transport, replay, provenance, and local naming rows. Since the earlier rows have already exhausted the public carrier surface, the local NameCert names exactly the displayed finite route and exports no additional public principle. □

Theorem 2372.16 (SequentialCompact finite subsequence handoff)**.** *Every accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet consumed by a compact-metric or Baire-space completion route must expose the finite handoff*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{BaireSpace}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

in that order: compact source data and Baire cylinder data are read before stream windows, stream windows are read before the displayed subsequence-window row, and the selected windows are read before the regular rational handoff and Real seal. The handoff exports no host subsequence selector, quotient stream equality, open-cover compactness theorem, countable choice, or ambient Real completeness row.

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier finite subsequence handoff`

Proof. Project the carrier $Q = (K, B, S, W, R, E, H, C, P, N)$. The rows K and B are the only compact-metric and Baire-space sources available to the consumer.

Next read the stream and subsequence windows. The row S gives finite source-window access, and the row W gives the selected finite windows and monotonicity evidence over those source windows.

Finally apply the readback and seal rows. The row R consumes only the selected windows from W, and the row E seals only the accepted regular rational readback. The transport, continuation, provenance, and NameCert rows preserve this same handoff, so no host subsequence selector, quotient stream equality, open-cover compactness theorem, countable choice, or ambient Real completeness can be exported. □

Theorem 2372.17 (SequentialCompact RegSeqRat readback route)**.** *Any accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must first expose the selected finite windows in* W *and then the regular rational readback row* R*. No consumer may replace* R *by a host subsequence limit, quotient stream equality, countable-choice selector, open-cover compactness theorem, or ambient Real completeness.*

Proof. Start from the NameCert obligations, Definition 2372.2. They identify W and R as displayed carrier rows, not as hidden host selectors or quotient-stream data.

Apply the finite subsequence handoff, Definition 2372.16. It orders the completion-facing route so that selected windows are exposed before regular rational readback and before the terminal RealUp seal.

Use window transport and selector refusal, Definition 2372.12 and Definition 2372.13. Transport preserves the displayed rows, while the refusal theorem leaves no coordinate for a host subsequence limit, quotient stream equality, countable-choice selector, open-cover compactness theorem, or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier regseqrat readback route`

Theorem 2372.18 (SequentialCompact selector monotonicity)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the selected finite windows exposed to any compact-metric or Baire-space completion consumer are monotone exactly because the row W displays the selected index list, the selected source windows, and the monotonicity evidence for that list. No consumer may replace W by an externally selected subsequence, countable-choice function, quotient stream selector, or host sequential compactness theorem.

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactSelectorMonotonicity`

Proof. Project the carrier from Definition 2372.1. The only coordinate that stores selected windows is W, and its displayed data include the selected indices, selected source windows, and monotonicity evidence.

Apply Definition 2372.16. Any completion consumer reads K, B, S before it reaches W, and the subsequent rows R, E, H, C, P, N preserve the same finite handoff.

Thus selector monotonicity is not imported from a host choice principle. It is the projection of the finite row W, transported by H and replayed by C without adding any second selector coordinate. □

Theorem 2372.19 (SequentialCompact RegSeqRat handoff exactness)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ *handoff available after the finite subsequence-window row W has been displayed. Every rational-tail budget, regular readback, or downstream completion request that depends on selected windows must pass through R before it can reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E.*

Proof. Read the handoff order in Definition 2372.16. The route reaches $\mathsf{StreamName}^{\uparrow}$ through S, then the selected finite windows through W, and only then the regular rational readback through R.

The carrier has no alternate regular rational coordinate. The rows H and C can transport and replay R, while P and N record provenance and naming for the same displayed row.

Consequently exactness of the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is a projection claim about R. A consumer that asks for rational-tail data after W receives that row and cannot bypass it through quotient stream equality, a selected limit, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRegSeqRatHandoffExactness`

Theorem 2372.20 (SequentialCompact Real seal nonescape)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}$ *row E is reachable only after the compact metric source K, Baire cylinder row B, stream-window row S, subsequence-window row W, regular rational handoff R, and structural rows H, C, P, N have been displayed. The seal exports no ambient Real completeness, selected limit, quotient equality of streams, countable choice, or host sequential compactness theorem.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCarrier real seal nonescape`

Proof. Start with the carrier projection. The row E is listed after R and is described as the $\mathsf{Real}^{\uparrow}$ seal reached after the regular rational readback is accepted.

Use Definition 2372.18 to keep the selected finite windows inside W. Then use Definition 2372.19 to force every rational-tail read from those windows through R.

It remains only to inspect the structural rows. Transport, replay, provenance, and naming preserve the displayed route K, B, S, W, R, E; they do not add completeness, quotient, or choice coordinates. Hence the real seal is a nonescaping endpoint of the finite handoff, not an ambient Real-completeness principle. □

Theorem 2372.21 (SequentialCompact window extraction scope)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every finite extraction request must expose the compact metric source row K, the Baire cylinder row B, the stream-window row S, and the selected finite-window row W before any $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal is available. The extraction scope exports no host subsequence selector, countable choice, quotient stream equality, open-cover compactness theorem, selected Real limit, or ambient Real completeness.*

Proof. Begin with Definition 2372.11. It places the Baire cylinder row B before selected-window data can be consumed, so an extraction request cannot start from a detached selector.

Next apply Definition 2372.16. The route reads K, B, S, W in finite order, and the selected source windows are packet rows rather than an infinite subsequence object.

Use Definition 2372.18. It keeps the selected indices and selected windows inside W, while H, C, P, N only transport, replay, source, and name the same displayed rows.

The later rows R and E are downstream of the window route. Hence the extraction scope is finite and does not export the excluded host selector, choice, quotient, compactness, selected-limit, or completeness principles. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactWindowExtractionScope`

Theorem 2372.22 (SequentialCompact Baire RegSeqRat exhaustion)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every regular rational readback requested after a Baire cylinder exposure must pass through the row R. The request first displays B, then the stream and selected-window rows S, W, and only then reads R before the seal E; no host subsequence selector, countable choice, quotient stream equality, open-cover compactness theorem, or ambient Real completeness is exported.

Proof. Read the Baire exposure through Definition 2372.11. It supplies the cylinder row B as displayed finite data before selected windows are available.

Apply Definition 2372.21. The request reaches the selected windows through K, B, S, W, so the regular rational readback is tied to the same finite route.

Now use Definition 2372.19. It identifies R as the only $\mathsf{RegSeqRat}^{\uparrow}$ handoff after W, and the structural rows do not add another readback coordinate.

Therefore the Baire-to-RegSeqRat route is exhausted by B, S, W, R and cannot be replaced by a host selector, a choice schedule, quotient stream equality, open-cover compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactBaireRegSeqRatExhaustion`

Theorem 2372.23 (SequentialCompact Real seal window nonescape)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is a window endpoint: it may be read only after the Baire cylinder row* B*, stream-window row* S*, selected-window row* W*, and regular rational readback row* R *have been displayed. The seal exports no host subsequence selector, countable choice, quotient stream equality, open-cover compactness theorem, selected Real limit, or ambient Real completeness.*

Proof. Start with Definition 2372.21. It forces every extraction request to expose B, S, W as finite packet rows before downstream readback can begin.

Then apply Definition 2372.22. Every regular rational request after that exposure passes through R, so the seal cannot be reached from a detached stream or selector.

Finally use Definition 2372.20. The row E is downstream of R and of the structural rows H, C, P, N, all of which preserve the displayed route.

Thus the Real seal remains a finite-window endpoint. It does not provide a host subsequence selector, countable choice, quotient stream equality, open-cover compactness, selected Real limit, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRealSealWindowNonescape`

Theorem 2372.24 (SequentialCompact cylinder monotone window)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every $\mathsf{BaireSpace}^{\uparrow}$ *cylinder exposure consumed by a compact-metric or Baire-space route is tied to the monotone selected-window row* W*. The consumer may inspect* B *and then* W*, transported by* H *and replayed by* C*, but it may not replace that route by a host choice-sequence branch, a nonmonotone selector, quotient stream equality, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Baire exposure theorem Definition 2372.11. It places the cylinder row B before any downstream read of selected windows, so the consumer reaches the window surface through the displayed carrier order.

Apply Definition 2372.16 and Definition 2372.18. The handoff places W after K, B, S, and the selector monotonicity theorem identifies the monotone selected indices and selected source windows as data stored inside W.

Finally use Definition 2372.20. The later readback and seal rows preserve the same finite route, so no consumer can bypass the Baire cylinder and monotone selected-window boundary through a host selector or an ambient compactness principle. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCylinderMonotoneWindow`

Theorem 2372.25 (SequentialCompact regular tail nonescape)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every tail request made after the selected finite-window row W *must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before it can reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The request exports no quotient stream equality, selected limit, countable-choice schedule, host sequential compactness theorem, open-cover compactness theorem, or ambient Real completeness row.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRegularTailNonescape`

Proof. Read W through Definition 2372.18. The selected finite windows and their monotonicity evidence are packet data, not a host subsequence object.

Next apply Definition 2372.19. It identifies R as the only regular rational handoff available after W, so every rational-tail budget or regular readback request must use that row.

Now apply Definition 2372.20. The seal row E is reachable only after R and the structural rows have been displayed, hence the tail request cannot escape through quotient streams, selected limits, countable choice, host compactness, or ambient completeness. □

Theorem 2372.26 (SequentialCompact Real seal consumer boundary)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every compact-metric, Baire-space, regular-rational, stream-window, or Real-facing consumer of the terminal seal can inspect only the rows $K, B, S, W, R, E, H, C, P, N$*. In particular, the consumer sees no row for external subsequence selection, quotient stream equality, open-cover compactness, countable choice, selected Real limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier from Definition 2372.1. Its visible coordinates are exactly $K, B, S, W, R, E, H, C, P, N$, and the local NameCert obligation theorem identifies those rows as the certificate surface.

Use Definition 2372.16 for the consumer order. Compact-metric and Baire reads pass through K and B, stream-window reads pass through S and W, regular rational reads pass through R, and the Real-facing endpoint is E.

Finally combine Definition 2372.19 with Definition 2372.20. They force the regular tail through R and the Real seal through E, while the structural rows preserve the displayed route and do not add excluded host or quotient data. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRealSealConsumerBoundary`

Theorem 2372.27 (SequentialCompact finite subsequence obligation)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every finite subsequence obligation read from Q *is a* BHist *obligation over the displayed rows* $K, B, S, W, R, E, H, C, P, N$*. The selected-window row* W *must be transported by the componentwise* hsame *row* H*, replayed through the displayed* Cont *row* C*, and named by the local* $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$ *row* N *before any compact-metric, stream-window, regular-rational, or Real-facing consumer can use it. The obligation refuses host subsequence selectors, countable choice, quotient stream equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactFiniteSubsequenceObligation`

Proof. Project the carrier from Definition 2372.1. The only row that stores selected subsequence information is W, and the carrier records it as finite selected indices, selected source windows, and monotonicity evidence.

Use Definition 2372.16. The handoff forces the consumer to reach W only after the compact-metric, Baire-cylinder, and stream-window rows have been displayed.

Now read the structural rows. The hsame row H transports the selected-window data componentwise, the Cont row C replays the same finite route, and the local $\mathsf{NameCert}$ row N names that displayed packet. None of these rows contains a coordinate for a host selector, countable-choice schedule, quotient stream equality, or ambient Real completeness. □

Theorem 2372.28 (SequentialCompact cluster carrier stability)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

any cluster-window read exposed by Q *is stable only along the same* BHist *carrier tuple. If a consumer changes presentation through* hsame*, the cluster read remains the transported* W, R, E *route replayed by* Cont *and named by* $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$*; it does not become a host subsequence selector, a countable-choice witness, a quotient stream equality, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactClusterCarrierStability`

Proof. Start with Definition 2372.12. The row H preserves the stream-window, selected-window, regular-readback, and terminal-seal rows before any completion consumer reads the route.

The cluster read is therefore carrier-local. It can use W for selected finite windows, R for regular rational readback, and E for the terminal seal, but each of these rows remains inside the same tuple Q.

Finally apply Definition 2372.26. A Real-facing or compact-metric consumer can inspect only $K, B, S, W, R, E, H, C, P, N$, so no presentation change can export the excluded host selector, choice, quotient, or ambient completeness data. □

Theorem 2372.29 (SequentialCompact compactmetric consumer scope)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every $\mathsf{CompactMetric}^{\uparrow}$ *consumer that uses the sequential-compactness packet has scope exactly the finite* BHist *rows* $K, B, S, W, R, E, H, C, P, N$*. It may read compact-metric source data, Baire cylinders, stream windows, finite subsequence windows, regular rational readback, the Real seal,* hsame *transport,* Cont *replay, package provenance, and the local* NameCert *row, but it may not request host subsequence selectors, countable choice, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCompactMetricConsumerScope`

Proof. Read the compact-metric source through K. The carrier definition gives no other compactness coordinate, and Definition 2372.11 places the Baire row B before selected windows are consumed.

Next use Definition 2372.27. It binds the finite subsequence obligation to W, H, C, N, so the compact-metric consumer receives displayed packet data rather than an external selector.

The downstream rows are R and E. By Definition 2372.19 and Definition 2372.20, those rows provide regular readback and the terminal seal without exporting quotient stream equality, countable choice, host subsequence selection, or ambient Real completeness. □

Theorem 2372.30 (SequentialCompact root selected-window route)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every root consumer that requests a finite selected subsequence window must expose the Baire-cylinder row B*, the StreamName source-window row* S*, and the selected-window row* W *before any compact-metric or Real-facing conclusion is read. The request is a finite* BHist *route through* K, B, S, W, H, C, P, N*, not a host subsequence selector, countable-choice schedule, quotient stream equality, or open-cover compactness argument.*

Proof. Project the carrier from Definition 2372.1. The rows that can supply selected-window data are B, S, W together with the compact-metric source K and the structural rows H, C, P, N.

Apply Definition 2372.16. It forces every subsequence handoff to pass through the finite selected-window row W after the compact-metric, Baire-cylinder, and stream-window data have been displayed.

Finally use Definition 2372.29. A root compact-metric consumer sees only $K, B, S, W, R, E, H, C, P, N$, so the selected-window route cannot be replaced by a host selector, countable choice, quotient stream equality, or open-cover compactness. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRootSelectedWindowRoute`

Theorem 2372.31 (SequentialCompact root RegSeqRat exposure)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

any root consumer that has read a selected finite window through W must read the resulting regular-rational tail data through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *row R before it can inspect the terminal seal E. The exposure route is $K, B, S, W, R, H, C, P, N$ and exports no quotient stream equality, selected limit, countable-choice subsequence, host sequential compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the selected-window route of Definition 2372.30. It places every root window request inside the displayed rows K, B, S, W, H, C, P, N.

Apply Definition 2372.19. The regular rational readback available after a selected window is exactly the row R, so a consumer cannot bypass R by treating the selected observations as quotient stream data or as a chosen limit.

The structural rows then replay the same finite route. The rows H, C, P, N transport, replay, source, and name R together with the selected-window data, and none of them contains countable choice, host sequential compactness, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRootRegSeqRatExposure`

Theorem 2372.32 (SequentialCompact root Real seal refusal)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

no root consumer may inspect the terminal $\mathsf{Real}^{\uparrow}$ *seal row E until the compact-metric source, Baire-cylinder row, StreamName source-window row, selected-window row, RegSeqRat readback row, componentwise transport, displayed replay, provenance, and local naming rows have been displayed. Thus the Real seal cannot serve as a shortcut for host sequential compactness, selected limits, countable choice, quotient stream equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRootRealSealRefusal`

Proof. First apply Definition 2372.31. It forces every root selected-window read to pass through R before the terminal seal can be considered.

Now use Definition 2372.20. The seal row E is only the terminal row of the displayed finite route; it does not introduce host sequential compactness, selected limits, countable choice, quotient stream equality, open-cover compactness, or ambient Real completeness.

Finally apply Definition 2372.26. A Real-facing consumer can inspect only $K, B, S, W, R, E, H, C, P, N$, so no root consumer obtains a seal shortcut outside the finite NameCert carrier. □

Theorem 2372.33 (SequentialCompact Baire real-window admission)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every Baire-window admission consumed by a Real-facing root reader must expose the Baire cylinder row B*, the StreamName source-window row* S*, the selected finite-window row* W*, the RegSeqRat readback row* R*, and the terminal Real seal row* E *in that order. The admitted window is a displayed* BHist *route through* $K, B, S, W, R, E, H, C, P, N$*, not a host subsequence selector, countable-choice branch, quotient stream equality, open-cover compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 2372.1. The only rows available for a Baire-window request are the compact-metric source K, the Baire cylinder row B, the stream source row S, and the selected finite-window row W.

Apply Definition 2372.30. It requires every root selected-window request to display B, S, W before any compact-metric or Real-facing conclusion is read.

Now use Definition 2372.31 and Definition 2372.32. These force the regular rational readback R and terminal seal E to remain downstream of the finite Baire and stream windows.

The rows H, C, P, N only transport, replay, source, and name that route. Since no row stores a host selector, choice branch, quotient equality, open-cover proof, or ambient completeness witness, the admission is exactly the displayed BHist window route. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactBaireRealWindowAdmission`

Theorem 2372.34 (SequentialCompact RegSeqRat cluster handoff)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every cluster read exported to a completion or Real-facing consumer must pass through the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *after the finite selected-window row* W *and before the terminal seal* E*. The cluster handoff is therefore the finite route* $K, B, S, W, R, E, H, C, P, N$*; it is not a selected subsequence object, a quotient stream class, a countable-choice schedule, or a host compactness conclusion.*

Proof. Start with Definition 2372.28. A cluster-window read remains the transported W, R, E route inside the same carrier tuple, even after presentation transport through hsame.

Apply Definition 2372.33. The row W is available only after the Baire and stream windows have been displayed, and the terminal seal E is available only after the regular rational readback route has appeared.

The handoff to a completion consumer therefore goes through R. By Definition 2372.19, R is the only regular-rational row exported by the packet, so a selected subsequence object or quotient stream class has no carrier coordinate from which to replace it.

Finally, H, C, P, N preserve the same route under transport, replay, provenance, and naming. This exhausts the cluster handoff surface seen by downstream consumers. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRegSeqRatClusterHandoff`

Theorem 2372.35 (SequentialCompact compactmetric root surface)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the root CompactMetric$^{\uparrow}$ *consumer receives only the finite compact-metric carrier row* K*, Baire and stream window rows* B, S, W*, RegSeqRat and Real seal rows* R, E*, and the structural rows* H, C, P, N*. In particular, compact-metric consumers may use the finite cluster-carrier and seal rows recorded by* Q*, but they may not import sequential compactness as a primitive, choose subsequences externally, identify quotient streams, invoke countable choice, or replace the route by an open-cover compactness theorem.*

Proof. Read the compact-metric consumer scope from Definition 2372.29. It restricts every such consumer to the rows $K, B, S, W, R, E, H, C, P, N$.

Use Definition 2372.34. Any cluster read that leaves the selected-window surface must pass through R before reaching the terminal seal E, so the compact-metric consumer receives finite cluster-carrier data rather than a selected subsequence object.

Apply Definition 2372.33. The Baire and stream windows remain displayed before Real-facing use, and the seal row cannot shortcut the window route.

The remaining structural rows are transport, replay, provenance, and local naming rows only. They carry no primitive sequential compactness, external selection, quotient stream equality, countable-choice schedule, or open-cover compactness proof, so the root surface is exactly the finite carrier-and-seal interface stated above. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCompactMetricRootSurface`

Theorem 2372.36 (SequentialCompact root obligation surface)**.** *For an accepted* SequentialCompact$^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

every root obligation read for the L10 compactness corridor must expose the CompactMetric$^{\uparrow}$ *source row* K*, the* BaireSpace$^{\uparrow}$ *cylinder row* B*, the* StreamName$^{\uparrow}$ *source-window row* S*, the finite selected-window row* W*, the* RegSeqRat$^{\uparrow}$ *readback row* R*, and the terminal* Real$^{\uparrow}$ *seal row* E *before transport, replay, provenance, and local naming are consulted. The obligation is a finite* BHist *surface over* $K, B, S, W, R, E, H, C, P, N$*, not countable choice, quotient stream equality, external subsequence selection, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRootObligationSurface`

Proof. Project the carrier of Definition 2372.1. The root reader sees only the compact-metric, Baire, stream, finite-window, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.

Apply Definition 2372.30 and Definition 2372.31. These force the selected-window read through B, S, W and then through the regular-rational row R before any Real-facing conclusion is available.

Apply Definition 2372.20. The seal row E cannot supply host sequential compactness, a selected subsequence object, quotient stream equality, countable choice, or ambient Real completeness. The remaining rows H, C, P, N preserve the same finite route. □

Theorem 2372.37 (SequentialCompact compactmetric handoff)**.** *Every* CompactMetric$^{\uparrow}$ *consumer that receives an accepted* SequentialCompact$^{\uparrow}$ *packet receives the compact-metric row* K *together with the Baire, StreamName, selected-window, RegSeqRat, Real-seal, transport, replay, provenance, and local naming rows* $B, S, W, R, E, H, C, P, N$*. The handoff permits downstream compact and completion chapters to consume finite carrier-and-window obligations, but it does not export a primitive sequential compactness theorem, an externally selected subsequence, quotient stream equality, countable choice, open-cover compactness, or ambient Real completeness.*

Proof. Use Definition 2372.29. It already restricts compact-metric consumers to the rows $K, B, S, W, R, E, H, C, P,$

Apply Definition 2372.36. The new root surface records the same rows as an obligation boundary, so the compact-metric handoff is a finite packet read rather than a hidden theorem about all sequences.

Finally use Definition 2372.34 and Definition 2372.26. They route cluster and Real-facing reads through R and E without introducing selected representatives, host stream equality, choice schedules, or ambient completion. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactCompactMetricHandoff`

Theorem 2372.38 (SequentialCompact RegSeqRat stream scope)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exported by an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet is scoped by the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row* S *and finite selected-window row* W*. A downstream completion or Real-facing consumer may read* R *only as a regular-rational readback of the finite stream window* S, W*; it may not replace the route by quotient stream equality, an externally chosen subsequence, countable choice, host real equality, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRegSeqRatStreamScope`

Proof. Read the source-window scope from Definition 2372.30. The stream source S and selected finite window W are displayed before the RegSeqRat row can be used.

Apply Definition 2372.31. It makes R the exposed regular-rational readback row for root consumers, not a selector or quotient-class object.

Use Definition 2372.33 to keep the Baire and Real-facing route attached to B, S, W, R, E. Since the carrier has no row for quotient stream equality, external subsequence selection, choice, host real equality, or ambient completeness, the RegSeqRat stream scope is exactly the displayed finite-window route. □

Theorem 2372.39 (SequentialCompact obligation closure)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the compact-metric row K*, Baire row* B*, StreamName row* S*, finite selected-window row* W*, RegSeqRat row* R*, terminal Real seal row* E*, and structural rows* H, C, P, N *together supply the full* $\mathsf{obligation}$ *surface for the* $\mathsf{SequentialCompact}^{\uparrow}$ *packet. Every compact-metric, Baire-space, RegSeqRat, or Real-facing consumer reads a finite* BHist *obligation over exactly these rows, with no host sequential compactness, countable choice, quotient stream equality, open-cover compactness, external subsequence selector, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactObligationClosure`

Proof. Start with Definition 2372.3. It identifies the next obligation surface with the displayed rows $K, B, S, W, R, E, H, C, P, N$ and refuses host subsequence selectors, quotient stream equality, countable choice, open-cover compactness, and ambient Real completeness.

The finite-window obligations are then exhausted by the local root and selector theorems. Apply Definition 2372.4, Definition 2372.10, and Definition 2372.13. These keep the selected-window data inside W and rule out any second selector or quotient-stream coordinate.

Next read the downstream handoffs. Use Definition 2372.19, Definition 2372.20, Definition 2372.36, and Definition 2372.38. They force every RegSeqRat read through R and every terminal Real-facing read through E after the displayed finite windows have appeared.

Finally use the compact-metric and structural consumer boundaries, Definition 2372.35 and Definition 2372.37. The rows H, C, P, N transport, replay, source, and name the same finite packet, so the obligation surface is closed exactly over $K, B, S, W, R, E, H, C, P, N$. □

Theorem 2372.40 (SequentialCompact public export no-choice boundary)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *carrier*

$$Q = (K, B, S, W, R, E, H, C, P, N),$$

the public SequentialCompact consumer receives only the finite cluster window W*, the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row* K*, the* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *window rows* B, S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the transport, replay, provenance, and local* NameCert *rows* H, C, P, N*. This public export is the already displayed finite route; it does not add a choice-selected subsequence, quotient stream, open-cover compactness theorem, host sequential compactness principle, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactPublicExport`

Proof. Start with Definition 2372.39. It closes the obligation surface over exactly the rows $K, B, S, W, R, E, H, C, P$, and excludes host compactness, countable choice, quotient stream equality, external subsequence selection, and ambient Real completeness.

Use Definition 2372.4 and Definition 2372.14. The public cluster read is the finite selected-window route through W, not an externally selected subsequence object.

Apply Definition 2372.15 and Definition 2372.37. These route public naming and compact-metric consumers through the same $K, B, S, W, R, E, H, C, P, N$ packet.

Finally use Definition 2372.38. It keeps every regular-rational readback scoped by the displayed stream and selected-window rows before the Real seal is consumed. Since no additional row is present in the carrier, the public export boundary is exactly the no-choice finite route stated above. □

Theorem 2372.41 (SequentialCompact formal target scope)**.** *The formal target for* $\mathsf{SequentialCompact}^{\uparrow}$ *is exactly the finite cluster route*

$$K, B, S, W, R, E, H, C, P, N$$

exported by Definition 2372.40: the $\mathsf{CompactMetric}^{\uparrow}$ *root surface* K*, the* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *window rows* B, S*, the selected finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and the structural* hsame*,* Cont*,* Pkg*, and local* NameCert *rows* H, C, P, N*. It excludes selected subsequence objects outside* W*, open-cover compactness, countable-choice schedules, quotient-stream equality, and ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactFormalTargetRouteCertificate`

Proof. Start with the public export boundary of Definition 2372.40. It states the visible consumer surface as exactly the finite cluster window, compact-metric handoff, Baire and StreamName rows, RegSeqRat readback, Real seal, and structural rows.

The RegSeqRat handoff and stream scope are already fixed by Definition 2372.19 and Definition 2372.38. These keep the readback row R downstream of B, S, W and upstream of the Real seal E.

The compact-metric root surface and Real seal boundary are supplied by Definition 2372.35, Definition 2372.37, and Definition 2372.20. Together they prevent the formal target from becoming an open-cover compactness claim or an ambient completeness claim.

Finally, Definition 2372.13 and Definition 2372.4 keep subsequence information inside the displayed finite-window row W. Thus the formalization queue receives the same finite Baire/StreamName/RegSeqRat/ cluster route and none of the excluded choice or completeness principles. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactFormalTargetStructuralRowsCertificate`

Theorem 2372.42 (SequentialCompact Real subsequence Cauchy handoff)**.** *For an accepted* $\mathsf{SequentialCompact}^{\uparrow}$ *packet, any Real-facing cluster consumer that reads the selected finite-window row must factor through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the* $\mathsf{Real}^{\uparrow}$ *seal as a Cauchy readback handoff. The consumer receives the displayed rows* S, W, R, E *from the finite-window route, not a host subsequence object, countable-choice schedule, quotient stream equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the public export of Definition 2372.40. It makes the selected finite-window row W, stream row S, regular-rational readback row R, and terminal Real seal E part of the same public packet.

Use the handoff exactness of Definition 2372.19 together with the stream scope of Definition 2372.38. These rows force every regular-rational readback to pass through the displayed $\mathsf{StreamName}^{\uparrow}$ source and selected finite window before it can be consumed downstream.

Finally apply Definition 2372.20. The Real-facing boundary reads E only after the S, W, R route is visible, so the handoff is a Cauchy readback route through finite data and not an additional subsequence, choice, quotient, selected-limit, or completeness object. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactRealSubsequenceCauchyHandoff`

Theorem 2372.43 (SequentialCompact standard bridge scope)**.** *Any standard-facing bridge for* $\mathsf{SequentialCompact}^{\uparrow}$ *may read only the displayed compact-metric row* K, *the* $\mathsf{BaireSpace}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *window rows* B, S, *the selected finite-window row* W, *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R, *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E, *and the structural* hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows* H, C, P, N *of the public packet. The bridge may not export host sequential compactness, a choice-selected subsequence, quotient stream equality, open-cover compactness, countable-choice schedules, selected limits outside* W, *or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 2372.40. It identifies the public consumer surface as exactly the finite route through $K, B, S, W, R, E, H, C, P, N$.

Use Definition 2372.19. The regular-rational readback row stays downstream of the displayed stream and selected finite-window rows, so a bridge cannot replace it by quotient stream equality or a host-selected subsequence.

Finally apply Definition 2372.20. The Real-facing seal is consumed only after the finite StreamName and RegSeqRat route is present. Hence the standard-facing bridge is scoped to the displayed finite Baire/StreamName/RegSeqRat/Real boundary and exports none of the excluded classical compactness or completeness principles. □

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactStandardBridgeScope`

Theorem 2372.44 (SequentialCompact mature public bridge package)**.** *The mature public package for* $\mathsf{SequentialCompact}^{\uparrow}$ *is exactly the finite route*

$$K, B, S, W, R, E, H, C, P, N$$

assembled by Definition 2372.40, Definition 2372.41, Definition 2372.42, and Definition 2372.43. It exports the compact-metric source row, Baire and StreamName window rows, selected finite-window row, RegSeqRat readback, terminal Real seal, componentwise transport, replay, provenance, and local naming, with the same exclusions: no host sequential compactness, choice-selected subsequence outside W, quotient stream equality, open-cover compactness, countable-choice schedule, selected limit outside the displayed route, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean credential, or bridge checking.*

Lean4 ✓ `BEDC.Derived.SequentialCompactUp.SequentialCompactMaturePublicBridgePackage`

Proof. Start with Definition 2372.40. It gives the public consumer exactly the finite cluster route through $K, B, S, W, R, E, H, C, P, N$.

Use Definition 2372.41 and Definition 2372.42. They keep the formal target and Real-facing Cauchy handoff on the same StreamName, selected-window, RegSeqRat, and Real-seal rows.

Finally apply Definition 2372.43. The standard-facing bridge may read only the displayed public packet and inherits the same exclusions, so the mature paper package is exactly the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 2372.1, Definition 2372.2, Definition 2372.15, Definition 2372.39, Definition 2372.40, Definition 2372.43, and Definition 2372.44 close the public finite subsequence-window route and its standard-facing bridge scope for compact-metric, Baire-space, RegSeqRat, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequential compactness, countable choice, quotient stream equality, open-cover compactness, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Mature closure requires Lean verification and bridge checking for the same public finite-window route; those remain separate verification-axis work.

Constructive story: A $\mathsf{SequentialCompact}^{\uparrow}$ packet is generated from CompactMetric source data, BaireSpace cylinder data, StreamName source windows, displayed finite subsequence windows, RegSeqRat readback, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 2373

A Concrete Naming Certificate for $\mathsf{SeparatedMetricCompletion}^{\uparrow}$

Following Bishop–Bridges and Kelley, the $\mathsf{SeparatedMetricCompletion}^{\uparrow}$ packet names the finite completion boundary obtained after zero-distance identifications are made visible rather than quotient-hidden. It sits between the metric completion source of chapter 1506, the separated metric classifier of chapter 1411, the Cauchy-filter route of chapter 333, the Cauchy-net route of chapter 589, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes a separated completion handoff for L10 Real and Completion consumers; it is not a quotient completion, selected representative principle, host equality theorem, countable-choice selector, or ambient completeness assertion.

Definition 2373.1 (SeparatedMetricCompletion carrier)**.** *A* $\mathsf{SeparatedMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, B, Q, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row naming the completion request and its metric evidence.*
2. *B is the selected Cauchy branch row, either the* $\mathsf{CauchyFilter}^{\uparrow}$ *route or the* $\mathsf{CauchyNet}^{\uparrow}$ *route, together with its displayed observation schedule.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback handoff extracted from the same finite Cauchy branch.*
4. *R records the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier used to identify when two completion reads name the same separated point.*
5. *Z is the local zero-distance comparison ledger tying R to the source metric row M and the branch row B.*
6. *H records componentwise* hsame *transport for the metric-completion source, Cauchy branch, readback, zero-distance classifier, comparison ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through the selected branch, regular readback, Real seal, zero-distance classifier, and separated handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{SeparatedMetricCompletion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completion classes, selected representatives, host equality, countable choice, ambient completeness, or an unrecorded completion object.

Lean4 ✓ `BEDC.Derived.SeparatedMetricCompletionUp.SeparatedMetricCompletionTasteGate single carrier alignment`

Theorem 2373.2 (SeparatedMetricCompletion NameCert obligations)**.** *For every accepted* SeparatedMetricComple *carrier $S = (M, B, Q, R, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedMetricCompletion}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *source, selected Cauchy branch, regular readback, Real seal, zero-distance classifier, comparison ledger, transport, continuation, provenance, and local naming rows;*
2. *source-first exposure, requiring every separated completion read to consume M before it reads the branch row B or the readback row Q;*
3. *branch exactness, requiring B to be one displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *route and requiring Q to be read from that same route;*
4. *separation exactness, requiring the zero-distance classifier R and comparison ledger Z to mediate all same-point reads before a consumer treats the handoff as separated;*
5. *ledger refusal, excluding quotient completion classes, selected representatives, host equality, countable-choice selectors, ambient completeness, and detached completion objects.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 2373.1. Its public rows are exactly $M, B, Q, R, Z, H, C, P, N$, so carrier admission exposes a metric-completion source, one Cauchy branch, the readback handoff, the zero-distance classifier, and the structural rows.

The source and branch obligations follow by replaying C. A separated completion read first consumes M, then chooses the displayed branch row B, and only then reads the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff in Q.

The separation obligation is carried by R and Z. These rows are the only coordinates that compare completion reads as zero-distance data, while H, P, and N transport, package, and name the same displayed route.

Every excluded object is absent from the carrier. Quotient classes, selected representatives, host equality, countable choice, ambient completeness, and detached completion objects therefore cannot supply an additional local obligation. □

Theorem 2373.3 (SeparatedMetricCompletion MetricCompletion handoff)**.** *Let $S = (M, B, Q, R, Z, H, C, P, N)$ be an accepted* $\mathsf{SeparatedMetricCompletion}^{\uparrow}$ *packet. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer that reads the separated handoff factors through the finite route*

$$M \longmapsto B \longmapsto Q \longmapsto R, Z \longmapsto H, C, P, N.$$

Thus the completion source is read before the Cauchy-filter or Cauchy-net branch, the Real-facing readback is read from that same branch, and the $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier mediates separated equality before the packet exits to downstream L10 consumers.*

Proof. Start with an accepted separated completion read. By carrier admission, the only completion source available to the consumer is the displayed row M.

Replay the selected branch. The row B records whether the finite Cauchy evidence is filter-presented or net-presented, and the readback row Q is attached to that same branch through the continuation row C.

Now apply the separated classifier rows. The row R supplies the $\mathsf{SeparatedMetric}^{\uparrow}$ zero-distance classifier, and Z records its compatibility with the source metric and the selected branch. Hence same-point use is mediated by displayed zero-distance data rather than by host equality.

Finally pass through the structural rows H, C, P, N. They preserve and name the route already displayed, so the handoff cannot introduce quotient completion classes, selected representatives, countable choice, ambient completeness, or a detached completion object. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedMetricCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2373.1, Definition 2373.2, and Definition 2373.3 seed the finite separated completion boundary between metric completion evidence, Cauchy-filter or Cauchy-net presentation, regular readback, Real sealing, and zero-distance classification.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion classes, selected representatives, host equality, countable-choice selectors, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, branch exactness, separation exactness, MetricCompletion handoff, and ledger refusal proofs for the same packet.

Constructive story: A $\mathsf{SeparatedMetricCompletion}^{\uparrow}$ packet is generated from a MetricCompletionUp source row, a displayed CauchyFilterUp or CauchyNetUp branch, RegSeqRatUp readback, a RealUp seal, a SeparatedMetricUp zero-distance classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2374

A Concrete Naming Certificate for $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$

Following Riesz–Fischer/Hilbert-space projection theorem treatments, the $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$ packet records the finite theorem-facing surface of orthogonal projection data without importing the full classical projection theorem as a host principle. It sits over the Hilbert carrier of chapter 91, the inner-product rows of chapter 86, the norm surface of $\mathsf{Norm}^{\uparrow}$, the Bessel-bound route of chapter 2207, and the vector-space operations of chapter 47. The singleton projection row of Definition 91.25 is the checked sibling surface: this chapter names the general finite packet that a theorem-facing projection consumer must read before any closed-subspace or minimizer claim can be exported.

Definition 2374.1 (Orthogonal projection theorem carrier)**.** *An* $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(H, V, I, Nm, B, A, Q, R, O, T, C, P, N)$$

whose visible rows are:

1. *H is the displayed* $\mathsf{Hilbert}^{\uparrow}$ *source carrier, read through its finite* $\mathsf{NameCert}_{\mathsf{Hilbert}^{\uparrow}}$ *rows;*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *reduct carrying addition, subtraction, scalar action, and the displayed affine subspace operations used by the projection row;*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the Hilbert source, and Nm is the associated* $\mathsf{Norm}^{\uparrow}$ *row used to state the distance-minimizing comparison;*
4. *B is the* $\mathsf{BesselInequality}^{\uparrow}$ *dependency row that bounds the finite orthogonal comparison ledger;*
5. *A is the displayed closed-affine constraint row for the target subspace surface visible to the packet;*
6. *Q is the candidate projection endpoint row;*
7. *R is the residual row $x - Q$ formed through the* $\mathsf{VecSpace}^{\uparrow}$ *operations;*
8. *O is the orthogonality classifier asserting that the residual row is orthogonal to the displayed subspace directions through I;*
9. *T is the minimality comparison row stating that scheduled norm comparisons through Nm factor through the same residual and orthogonality ledger;*

10. *C, P, N are the consumer replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{OrthogonalProjectionTheorem}^{\uparrow}}$ *rows. The carrier has no coordinate for a host closed-subspace type, no choice-selected minimizer, no quotient Hilbert equality, no global Riesz representation theorem, and no ambient completeness argument beyond the displayed* $\mathsf{Hilbert}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.OrthogonalProjectionTheoremUp.OrthogonalProjectionTheoremTasteGate single carrier alignment`

Theorem 2374.2 (Orthogonal projection theorem NameCert obligations)**.** *For an accepted* OrthogonalProjectionTh *carrier* $(H, V, I, Nm, B, A, Q, R, O, T, C, P, N)$*, the local NameCert obligation surface consists of the following finite rows:*

1. *carrier admission for the Hilbert source H, vector-space reduct V, inner-product row I, norm row Nm, Bessel-bound row B, displayed constraint row A, candidate projection Q, residual R, orthogonality classifier O, minimality comparison T, replay C, provenance P, and naming row N;*
2. *dependency admission through* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{BesselInequality}^{\uparrow}$, *and* $\mathsf{VecSpace}^{\uparrow}$*;*
3. *residual formation, stating that R is built from the displayed vector-space operations and the candidate endpoint Q;*
4. *orthogonality classification, stating that O reads only the inner-product rows visible through I and the finite directions named by A;*
5. *Bessel-controlled comparison, stating that B bounds the finite orthogonal ledger before T records the norm-minimality comparison;*
6. *singleton compatibility, stating that the Hilbert singleton projection row of Definition 91.25 is an admitted special case of the same residual, orthogonality, and norm-readback pattern;*
7. *nonescape, stating that the packet exports no host projection operator, no selected minimizer outside the displayed rows, no closed-subspace completeness theorem, and no quotient Hilbert equality.*

Proof. Project the carrier of Definition 2374.1. Its coordinates list exactly the Hilbert source, vector-space reduct, inner product, norm, Bessel ledger, constraint row, candidate endpoint, residual, orthogonality classifier, norm comparison, replay, provenance, and local name.

The dependency rows are finite because each is read from a named sibling surface. The $\mathsf{Hilbert}^{\uparrow}$ row supplies the source carrier, $\mathsf{InnerProduct}^{\uparrow}$ supplies the orthogonality observations, $\mathsf{Norm}^{\uparrow}$ supplies the distance comparison, $\mathsf{BesselInequality}^{\uparrow}$ supplies the finite bound ledger, and $\mathsf{VecSpace}^{\uparrow}$ supplies the algebraic residual operations.

Residual formation and orthogonality are then local rows. The residual R is formed by the displayed vector operations, while O can inspect only inner-product entries named by I and directions named by A. The minimality row T reads the same norm row Nm after the Bessel ledger has fixed the finite orthogonal comparison surface.

For the sibling check, specialize to the Hilbert singleton projection row of Definition 91.25. There the candidate endpoint, residual, norm readback, and inner-product readback are all the displayed singleton zero rows, so the general packet pattern collapses to the existing finite certificate surface.

Since every row is one of these displayed coordinates, no host projection operator, chosen minimizer, quotient equality, or external closed-subspace theorem can be read from the packet. □

Theorem 2374.3 (Orthogonal projection theorem Hilbert handoff)**.** *Every Hilbert-facing consumer of an* $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$ *packet must read the displayed Hilbert source* H*, vector-space reduct* V*, inner-product row* I*, norm row* Nm*, Bessel-bound row* B*, constraint row* A*, candidate projection* Q*, residual row* R*, orthogonality classifier* O*, and norm-minimality comparison* T *before exporting a projection-theorem conclusion. The handoff is therefore a finite NameCert route through* $\mathsf{Hilbert}^{\uparrow}$*,* $\mathsf{InnerProduct}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{BesselInequality}^{\uparrow}$*, and* $\mathsf{VecSpace}^{\uparrow}$*, not a claim of the full host projection theorem.*

Proof. Start with a consumer asking for projection-theorem data over a displayed Hilbert source. By Definition 2374.1, the only available Hilbert data are the rows H, V, I, Nm and the finite theorem-facing rows $B, A, Q, R, O, T, C, P, N$.

Apply the obligation surface of Definition 2374.2. It forces residual formation through V, orthogonality through I, bounded comparison through B, and norm-minimality readback through Nm before the consumer can reach the local name.

The checked singleton sibling in Definition 91.25 shows that this route has a concrete Hilbert projection instance, but it does not promote the packet to a global closed-subspace projection theorem. The handoff exports only the displayed finite rows and excludes host choice, quotient Hilbert equality, and non-displayed completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{OrthogonalProjectionTheorem}^{\uparrow}}$.

Constructive story: An orthogonal-projection-theorem packet is generated from a displayed Hilbert source, vector-space reduct, inner-product and norm rows, a Bessel-bound ledger, a displayed constraint row, a candidate projection endpoint, residual formation, orthogonality classification, norm-minimality comparison, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrthogonalProjectionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2374.1, Definition 2374.2, and Definition 2374.3 declare the finite theorem-facing carrier and Hilbert handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrthogonalProjectionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host closed-subspace projection operator, choice-selected minimizers, quotient Hilbert equality, or the full classical projection theorem.

Upgrade path: Obligation closure requires checked carrier admission, dependency admission, residual formation, orthogonality classification, Bessel-controlled comparison, singleton compatibility, and consumer nonescape for the same finite packet.

Chapter 2375

A Concrete Naming Certificate for $\mathsf{VietorisTopology}^{\uparrow}$

Following Vietoris and Kelley, the $\mathsf{VietorisTopology}^{\uparrow}$ packet records the finite subbasic-hit and subbasic-miss topology read on compact-subset names. It sits over the compact-subset surface of chapter 1660, the finite-net distance route of chapter 2086, the compact metric source of chapter 129, and the ambient topology carrier of chapter 88. The packet names the lower-hit and upper-miss generators as BEDC rows; it is not a host powerset topology, not subset equality, not a choice-selected point, and not open-cover compactness.

Definition 2375.1 (Vietoris topology carrier)**.** *A* $\mathsf{VietorisTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (H, K, L, U, M, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hyperspace}^{\uparrow}$ *source row, recording the compact-subset naming surface and the finite Hausdorff-distance ledger through which compact-subset requests are read.*
2. *K is the compact-subset row over the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source, including the finite subset-net schedules and nonempty compact-subset admission data that the hyperspace route exposes.*
3. *L is the lower-hit row. It records subbasic generators of the form "the compact-subset name hits a displayed open row" through finite witness windows routed by H and K.*
4. *U is the upper-miss row. It records subbasic generators of the form "the compact-subset name is covered by a displayed open row" through finite miss certificates and compact-metric net checks.*
5. *M is the metric and finite-net compatibility row, comparing the lower-hit and upper-miss requests with the* $\mathsf{HausdorffMetric}^{\uparrow}$ *finite-net distance route and the* $\mathsf{CompactMetric}^{\uparrow}$ *source schedules.*
6. *T is the topology transport row from the displayed subbasic hit and miss generators into the ambient* $\mathsf{Topology}^{\uparrow}$ *carrier.*
7. *C records displayed* Cont *replay of the route*

$$H, K, L, U, M \longmapsto T.$$

8. *P is the* Pkg *provenance over* VietorisTopology$^{\uparrow}$, Hyperspace$^{\uparrow}$, HausdorffMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, Topology$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{VietorisTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, K, L, U, M, T, C, P, N$. It has no coordinate for host subset equality, choice-selected points, open-cover compactness, quotient compacta, or ambient powerset topology.

Lean4 ✓ `BEDC.Derived.VietorisTopologyUp.VietorisTopologyTasteGate single carrier alignment`

Theorem 2375.2 (VietorisTopologyUp NameCert obligations)**.** *For every accepted* VietorisTopology$^{\uparrow}$ *carrier*

$$V = (H, K, L, U, M, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VietorisTopology}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the hyperspace source, compact-subset row, lower-hit row, upper-miss row, finite-net compatibility row, topology transport row, replay row, provenance row, and local naming row;*

2. *lower-hit exactness, requiring every hit generator in L to be read through the displayed hyperspace and compact-subset rows before a witness window is exposed;*

3. *upper-miss exactness, requiring every miss generator in U to be read through the same compact-metric net schedules before a cover read is exposed;*

4. *finite-net compatibility, requiring M to preserve the HausdorffMetric and CompactMetric tolerance ledgers attached to the hit and miss rows;*

5. hsame *and* Cont *transport, requiring T and C to carry the subbasic rows into* Topology$^{\uparrow}$ *without adding a hidden classifier surface;*

6. *ledger nonescape through P and N, excluding host subset equality, choice-selected points, open-cover compactness, quotient compacta, and ambient powerset topology.*

No Vietoris topology obligation is stored outside these rows.

Closure status for $\mathsf{NameCert}_{\mathsf{VietorisTopology}^{\uparrow}}$**.**

Constructive story: A VietorisTopology$^{\uparrow}$ packet is generated from a HyperspaceUp source, compact-subset rows, lower-hit generators, upper-miss generators, HausdorffMetric and CompactMetric finite-net compatibility, topology transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VietorisTopology}^{\uparrow}}$, seed)

Scope closed: Definition 2375.1 and Definition 2375.2 seed the finite lower-hit and upper-miss topology surface over hyperspace, HausdorffMetric, CompactMetric, and Topology rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{VietorisTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host subset equality, choice-selected points, open-cover compactness, quotient compacta, ambient powerset topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-hit exactness, upper-miss exactness, finite-net compatibility, topology transport, replay, provenance, local naming, and ledger nonescape for BEDC.Derived.VietorisTopologyUp.

Chapter 2376

A Concrete Naming Certificate for $\mathsf{LocatedRegSeqRat}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedRegSeqRat}^{\uparrow}$ packet records a located regular sequence of rational approximants as finite BEDC NameCert data. It sits after the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback surface of chapter 301, the located real boundary of **??**, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with a nonquotient route from displayed regular sequence data to located real-name handoff, without importing selected limits, quotient stream equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2376.1 (LocatedRegSeqRat carrier)**.** *A* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, Q, W, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, exposing finite dyadic scales and comparison rows used by later location requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, naming the finite windows at which rational approximants may be read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence source row, carrying rational readback and regularity evidence for the scheduled windows.*
4. *Q is the located comparison row, recording for each displayed dyadic tolerance the finite branch that locates the regular sequence relative to the requested rational cut.*
5. *W is the tail-window row, binding the location read to a finite suffix of the scheduled regular sequence.*
6. *H records componentwise* hsame *transport for D, S, R, Q, W, K, P, N.*
7. *K records displayed* Cont *replay from a location request through the dyadic, stream, regular-sequence, comparison, tail-window, transport, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegSeqRat}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $D, S, R, Q, W, H, K, P, N$. It has no coordinate for quotient stream equality, a selected limit representative, countable choice, trichotomy, host rational order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRegSeqRatUp.LocatedRegSeqRatTasteGate single carrier alignment`

Theorem 2376.2 (LocatedRegSeqRat NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier $L = (D, S, R, Q, W, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegSeqRat}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite stream-window exposure, regular rational readback, located comparison classification, tail-window binding, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and ledger refusal for quotient streams, selected limit representatives, countable choice, trichotomy, host rational order, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 2376.1. The packet exposes the dyadic ledger D, the observation schedule S, the regular rational row R, the located comparison row Q, and the tail-window row W before the transport, replay, provenance, and local naming rows can be consumed.

The NameCert obligations are precisely these projections. The dyadic and stream rows determine which finite tolerance and observation windows are available; the regular rational row supplies the readback; the located comparison and tail-window rows state where the location request is decided.

Transport and replay are exhausted by H and K. Since the classifier has no additional coordinate beyond $D, S, R, Q, W, H, K, P, N$, the local naming row cannot import quotient stream equality, selected limit representatives, countable choice, trichotomy, host rational order, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2376.3 (LocatedRegSeqRat tail-window stability)**.** *Every transported* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *location read preserves the same finite tail-window route. If a consumer requests a dyadic tolerance through D, the transported packet must still read the scheduled stream windows S, the regular rational source R, the located comparison row Q, and the tail-window row W before it can inspect the local naming row N.*

Proof. Project the transport row H from the carrier. It acts componentwise on the dyadic ledger, stream schedule, regular rational source, located comparison row, tail-window row, replay, provenance, and local naming rows.

The replay row K then repeats the same finite route after transport. A tolerance request first reads D, then the scheduled windows S, then the regular rational source R, and then the located comparison row Q bound to the suffix named by W.

No transported read can bypass this suffix because the local naming row is not exposed until the displayed route has been replayed. Thus tail-window stability is internal to the carrier surface. □

Lean4 ✓ `BEDC.Derived.LocatedRegSeqRatUp.LocatedRegSeqRatCarrier tail window stability`

Theorem 2376.4 (LocatedRegSeqRat RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *packet factors through the located regular-sequence carrier before the terminal real-name seal is read. The handoff exposes only $D, S, R, Q, W, H, K, P, N$ and the sibling* $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *certificate surfaces, so it cannot supply a quotient completion, selected limit, countable-choice schedule, trichotomy decision, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Proof. Begin with a Real-facing read. The carrier forces the read through the dyadic tolerance ledger, finite stream schedule, regular rational source, located comparison classifier, and tail-window row before it reaches provenance or local naming.

The sibling $\mathsf{LocatedReal}^{\uparrow}$ boundary consumes the displayed location row and not a host cut predicate. The $\mathsf{Real}^{\uparrow}$ seal is therefore terminal for the displayed route rather than a source of further completeness data.

Since no classifier coordinate stores quotient completion, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the handoff exports only the finite located regular-sequence certificate surface. □

Theorem 2376.5 (LocatedRegSeqRat modulus consumer route)**.** *Every accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *packet used by a Real or Completion consumer factors through its displayed locatedness row, finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{Real}^{\uparrow}$ *seal before any consumer-visible located read is exported. The locatedness row is downstream of finite windows and cannot be replaced by quotient stream equality, a selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier as $L = (D, S, R, Q, W, H, K, P, N)$. The carrier exposes the dyadic tolerance ledger D, the finite window schedule S, the regular rational readback row R, the located comparison row Q, and the tail-window row W before provenance or local naming can be consumed.

Apply Definition 2376.2 to identify the displayed location surface and Definition 2376.4 to route Real-facing reads through the terminal $\mathsf{Real}^{\uparrow}$ seal. The existing handoff makes the consumer pass through the same finite rows rather than through an external completion principle.

The locatedness row is therefore downstream of finite windows, regular rational readback, and dyadic tolerance data. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, no consumer-visible located read can bypass this modulus route. □

Theorem 2376.6 (LocatedRegSeqRat Bishop located-completion handoff)**.** *Every* $\mathsf{BishopLocatedCompletion}^{\uparrow}$ *read sourced from an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *packet must pass through the displayed DyadicRatCore tolerance ledger, StreamName window schedule, RegSeqRat readback row, located comparison row, finite tail-window row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *before a completion-facing located seal is exposed. The completion consumer therefore reads the route*

$$\mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{LocatedReal}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BishopLocatedCompletion}^{\uparrow}$$

only through the finite rows $D, S, R, Q, W, H, K, P, N$ *of Definition 2376.1.*

Proof. Project the accepted carrier as $L = (D, S, R, Q, W, H, K, P, N)$. By Definition 2376.2, the local NameCert surface exposes the dyadic tolerance ledger, stream windows, regular rational readback, located comparison row, tail-window row, transport, replay, provenance, and local naming as the complete set of readable rows.

Apply Definition 2376.3 to keep the same finite suffix after transport. The completion-facing consumer cannot move the located comparison away from the displayed tail-window route, because every transported location request must replay D, S, R, Q, W before it can inspect local naming.

Apply Definition 2376.4 and Definition 2376.5. These results force Real-facing and Completion-facing reads through the located row, the finite StreamName window, the RegSeqRat readback, the

DyadicRatCore tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any consumer-visible located read is exported.

The $\mathsf{BishopLocatedCompletion}^{\uparrow}$ consumer therefore receives a located completion seal only after the same finite rows have been displayed. Since the $\mathsf{LocatedRegSeqRat}^{\uparrow}$ carrier contains no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, none of those objects can be inserted into the handoff. □

Theorem 2376.7 (LocatedRegSeqRat explicit-modulus handoff)**.** *Every explicit-modulus consumer of an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier* $L = (D, S, R, Q, W, H, K, P, N)$ *must read the dyadic tolerance ledger* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, located comparison row* Q*, and tail-window row* W *before any Real-facing or completion-facing located seal is exported. The handoff therefore exposes an explicit finite route from tolerance and window data to the located seal, not quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $L = (D, S, R, Q, W, H, K, P, N)$. The explicit-modulus consumer can inspect only the displayed dyadic ledger, stream schedule, regular rational source, located comparison, and tail-window rows before the structural rows are replayed.

Apply Definition 2376.3. It keeps every transported location request on the same finite suffix route through D, S, R, Q, W.

Apply Definition 2376.5 and Definition 2376.6. These handoffs require the Real-facing or completion-facing located seal to be exported only after the finite tolerance ledger, finite window schedule, regular sequence readback, located comparison, and tail-window row have been displayed.

Since no carrier coordinate stores quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, an explicit-modulus consumer cannot bypass this finite route. □

Theorem 2376.8 (LocatedRegSeqRat finite location decision)**.** *Every located comparison row* Q *of an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier is decided only after the dyadic tolerance ledger* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and tail-window row* W *have been displayed. Thus a location decision is a finite BEDC route through* D, S, R, W, Q*, not a host trichotomy decision, selected limit comparison, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *order test.*

Proof. Project the accepted carrier $L = (D, S, R, Q, W, H, K, P, N)$. The carrier lists the dyadic tolerance ledger, stream schedule, regular rational row, located comparison row, and tail-window row as displayed data before transport, replay, provenance, or local naming can be read.

The comparison row Q is downstream of D, S, R, W. The tolerance to be decided is supplied by D, the finite observation windows by S, the rational readback by R, and the finite suffix binding by W.

The structural rows H, K, P, N only preserve and replay this displayed route. Since no coordinate stores trichotomy, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ order, the located decision remains a finite comparison over the displayed rows. □

Theorem 2376.9 (LocatedRegSeqRat tail-modulus transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier preserves the same dyadic tolerance and finite* $\mathsf{StreamName}^{\uparrow}$ *schedule for every transported tail-window read. In particular, a transported comparison cannot replace the displayed tolerance ledger* D *or schedule* S *with a new modulus source outside the carrier.*

Proof. Start with the transport row H of $L = (D, S, R, Q, W, H, K, P, N)$. By carrier projection, H acts componentwise on the dyadic ledger, stream schedule, regular rational readback, located comparison, tail-window, replay, provenance, and local naming rows.

Apply Definition 2376.3. It already forces every transported location request to replay D, S, R, Q, W before the local naming row is inspected.

The same route preserves the tolerance and schedule used by the original tail-window read. If a transported packet changed the dyadic tolerance source or finite StreamName schedule, it would leave the componentwise transport row and therefore would not be the accepted transported $\mathsf{LocatedRegSeqRat}^{\uparrow}$ packet. □

Theorem 2376.10 (LocatedRegSeqRat window-modulus fusion)**.** *For every accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier* $L = (D, S, R, Q, W, H, K, P, N)$*, finite location decisions and tail-modulus transport share one displayed route through the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any Real-facing consumer reads locatedness. The fused route exports no detached location test and no detached modulus source.*

Proof. Apply Definition 2376.8. A location decision reads the dyadic tolerance ledger D, finite StreamName schedule S, regular rational row R, tail-window row W, and comparison row Q as one finite BEDC route.

Apply Definition 2376.9. The transported tail-window read preserves the same tolerance ledger and finite schedule, so the modulus side cannot replace D or S by a source outside the carrier.

Finally apply Definition 2376.11. The $\mathsf{Real}^{\uparrow}$ seal is terminal for the displayed route $D, S, R, Q, W, H, K, P, N$. Therefore the location decision and transported modulus read fuse before the Real seal is consumed, and neither can export a separate host order test, selected limit, quotient stream equality, or ambient completeness principle. □

Theorem 2376.11 (LocatedRegSeqRat Real seal nonescape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *carrier is terminal for the displayed finite route. It cannot export quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness to a Real-facing or Completion-facing consumer.*

Proof. Use Definition 2376.4. Every Real-facing consumer first factors through the dyadic ledger, finite stream schedule, regular rational source, located comparison, tail-window row, transport, replay, provenance, and local naming.

Use Definition 2376.5 and Definition 2376.6. These handoffs keep Real-facing and Completion-facing reads downstream of the finite rows $D, S, R, Q, W, H, K, P, N$.

The terminal seal is therefore a boundary row, not a source of additional host principles. Since the carrier has no coordinate for quotient stream equality, selected limits, trichotomy, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the seal. □

Theorem 2376.12 (LocatedRegSeqRat RealEquality handoff)**.** *Every equality-facing consumer of an accepted* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *packet must read the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, located comparison row, tail-window row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before it can invoke the* $\mathsf{RealEquality}^{\uparrow}$ *classifier boundary. The handoff exports no quotient stream equality, selected limit, trichotomy, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 2376.10. The located decision and tail-modulus read share the same finite route through the dyadic ledger, StreamName schedule, RegSeqRat readback, located comparison row, and tail-window row.

Apply Definition 2376.11. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of those displayed rows and cannot export a quotient, selected limit, trichotomy row, countable-choice schedule, host equality, or ambient completeness principle.

Finally use the classifier and NameCert boundary of Definition 1636.2 and Definition 1636.3. A $\mathsf{RealEquality}^{\uparrow}$ read can inspect equality only after the same finite windows, regular readbacks, dyadic tolerance ledger, and Real-seal boundary are visible, so the accepted $\mathsf{LocatedRegSeqRat}^{\uparrow}$ packet reaches RealEquality through the displayed L10 rows and no hidden equality source. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegSeqRat}\uparrow}$.

Constructive story: A $\mathsf{LocatedRegSeqRat}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName observation windows, RegSeqRat readbacks, a located comparison row, a finite tail-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRegSeqRat}\uparrow}, \mathsf{seed})$

Scope closed: Definition 2376.1, Definition 2376.2, Definition 2376.3, and Definition 2376.4 seed the finite route from displayed regular rational sequence data to located Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRegSeqRat}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, location classification, tail-window stability, RealUp handoff, and ledger refusal for BEDC.Derived.LocatedRegSeqRatUp.

Chapter 2377

A Concrete Naming Certificate for $\mathsf{FiniteRealCover}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{FiniteRealCover}^{\uparrow}$ packet records the finite ball-cover data used by metric and compact-metric arguments without importing arbitrary open-cover compactness or a choice-selected subcover. It sits over the $\mathsf{Met}^{\uparrow}$ surface of chapter 57, the $\mathsf{Real}^{\uparrow}$ distance seal of chapter 38, the dyadic radius arithmetic of chapter 291, the total-bounded witness interface of chapter 127, and the compact-metric consumer route of chapter 129. The packet names only finite ball rows, dyadic radii, coverage witnesses, refinement ledgers, componentwise transport, replay, provenance, and local naming evidence.

Definition 2377.1 (Finite real cover carrier)**.** *A* $\mathsf{FiniteRealCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, R, D, B, W, E, H, C, P, N)$$

with the following displayed coordinates:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *row whose distance observations are sealed through* $\mathsf{Real}^{\uparrow}$*;*
2. *R is a finite row of* $\mathsf{Real}^{\uparrow}$ *or metric source probes to be covered;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, with each radius recorded as a positive dyadic tolerance;*
4. *B is the finite ball row assigning each listed center and dyadic radius to its displayed metric-distance witness;*
5. *W is the coverage witness row, pairing each inspected source probe in R with an admitted ball in B and its distance bound;*
6. *E is the refinement ledger, recording how a finer finite ball row reuses or refines the displayed centers, dyadic radii, and coverage witnesses;*
7. *H is the componentwise* hsame *transport row for the metric, probe, radius, ball, coverage, refinement, provenance, and local naming coordinates;*
8. *C is the displayed* Cont *replay route for metric distance reads, radius reads, coverage reads, and refinement reads;*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealCover}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for an arbitrary open cover, a host finite-subcover selection function, a powerset compactness theorem, a quotient real equality, or an ambient completeness object.

Definition 2377.2 (Finite real cover classifier)**.** *Two accepted* $\mathsf{FiniteRealCover}^{\uparrow}$ *packets*

$$F = (M, R, D, B, W, E, H, C, P, N)$$

and

$$F' = (M', R', D', B', W', E', H', C', P', N')$$

are classified together exactly when the stored hsame *rows transport* M *to* M'*,* R *to* R'*,* D *to* D'*,* B *to* B'*,* W *to* W'*,* E *to* E'*,* P *to* P'*, and* N *to* N'*, and the displayed* Cont *rows replay the same metric-distance, dyadic-radius, coverage, and refinement route on both packets. The classifier does not compare host subsets, arbitrary cover families, selected subcovers, completed metric spaces, or quotient real representatives.*

Theorem 2377.3 (Finite real cover NameCert obligations)**.** *For the carrier of Definition 2377.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteRealCover}^{\uparrow}}$ *surface consists exactly of carrier admission for* $F = (M, R, D, B, W, E, H, C, P, N)$ *classifier stability by Definition 2377.2, metric-distance witness admission through* M*, dyadic radius admission through* D*, finite ball admission through* B*, coverage witness admission through* W*, refinement-ledger admission through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.FiniteRealCoverUp.FiniteRealCoverCarrier namecert obligations`

Proof. Project the carrier. Its rows display the metric source, inspected probes, dyadic radii, ball witnesses, coverage witnesses, refinement ledger, transport, replay, provenance, and local name as one finite packet.

Use the classifier definition for stability. It compares only the displayed rows and transports them componentwise through hsame, then replays the same route through C.

The omitted open-cover, subcover-choice, quotient, and completeness objects are not coordinates of F. Coordinate exhaustion therefore leaves exactly the listed NameCert obligations. □

Theorem 2377.4 (Finite real cover refinement stability)**.** *Any accepted refinement read of a* $\mathsf{FiniteRealCover}^{\uparrow}$ *packet must pass through the displayed refinement ledger* E*. If a finer packet refines* F*, the read transports the metric row* M*, dyadic radius ledger* D*, ball row* B*, and coverage row* W *through the stored* hsame *entries and replays the same* Cont *route before the finer coverage witnesses are exposed. It cannot replace the ledger by a host cover refinement, arbitrary finite-subcover choice, completed metric compactness result, or quotient real equality.*

Proof. Start with a refinement-facing consumer. Carrier projection gives access only to the rows $M, R, D, B, W, E, H, C, P, N$.

The refinement ledger E is the only row whose role is to relate the displayed balls, dyadic radii, and coverage witnesses to a finer finite ball row. Transport through H preserves those coordinates componentwise, and replay through C preserves the metric and dyadic reads.

Since no host refinement operation or subcover selector is stored in the packet, the accepted refinement remains the finite ledger route described above. □

Theorem 2377.5 (Finite real cover total-bounded handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer citing an accepted* $\mathsf{FiniteRealCover}^{\uparrow}$ *packet receives a finite handoff through*

$$M \longmapsto D \longmapsto B \longmapsto W \longmapsto E.$$

The handoff supplies the metric-distance witness, dyadic tolerance, finite ball row, coverage witness, and refinement ledger needed by total-bounded and compact-metric reads. It supplies no general open-cover compactness theorem, no choice-selected subcover, no ambient powerset cover object, no independent complete-metric carrier, and no quotient real representative.

Proof. Read the consumer through the carrier. The metric row M gives the distance source, D gives the dyadic tolerance budget, B gives the finite ball witnesses, and W gives the coverage witnesses.

For refinement-sensitive consumers, apply refinement stability. The ledger E keeps finer covers inside the same finite ball and dyadic-radius route before any total-bounded or compact-metric handoff is exported.

The listed excluded objects are not stored in the carrier, classifier, transport, replay, provenance, or local NameCert rows. Therefore the total-bounded and compact-metric handoff is exhausted by the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteRealCover}^{\uparrow}$ packet is generated from a metric source row, Real-valued distance probes, dyadic radius rows, finite ball rows, coverage witnesses, a refinement ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRealCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope records the finite carrier, classifier, NameCert obligations, refinement stability, and total-bounded handoff for explicit metric ball covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRealCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, choice-selected subcovers, hyperspace compactness, quotient real equality, or ambient metric completion.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, coverage-witness nonescape, refinement-ledger stability, and total-bounded handoff for BEDC.Derived.FiniteRealCoverUp.

Chapter 2378

A Concrete Naming Certificate for $\mathsf{RealNullSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealNullSequence}^{\uparrow}$ packet records a displayed real-valued sequence whose finite convergence surface is tied to the zero $\mathsf{Real}^{\uparrow}$ row. It sits after the real-name seal of chapter 38, the finite-window discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the displayed sequence-limit interface of chapter 1413. The packet gives later sequence-algebra and Cauchy-criterion consumers a finite zero-limit route without importing host sequence equality, quotient limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2378.1 (RealNullSequence carrier)**.** *A* $\mathsf{RealNullSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, 0, W, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *S is a displayed sequence of* $\mathsf{Real}^{\uparrow}$ *names, each read only through the finite packet rows made visible to this carrier.*
2. *0 is the displayed zero* $\mathsf{Real}^{\uparrow}$ *row, supplied as the candidate limit and not as a host-level constant outside the packet.*
3. *W stores the* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to inspect sequence entries and the zero row at the requested tolerances.*
4. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
5. *D stores the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, slack rows, and comparison budgets used by every scheduled read.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *convergence classifier row, specialized to the displayed sequence S and the zero row 0.*
7. *H records componentwise* hsame *transport for sequence rows, zero row, windows, readbacks, tolerances, convergence classifier, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from sequence windows through regular rational readback, dyadic tolerance comparison, and the zero-limit classifier.*
9. *P is the* Pkg *provenance over* $\mathsf{RealNullSequence}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealNullSequence}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, 0, W, R, D, L, H, C, P, N$. It has no coordinate for host sequence equality, quotient limits, countable-choice schedules, trichotomy decisions, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 2378.2 (RealNullSequence NameCert obligations)**.** *For every accepted* $\mathsf{RealNullSequence}^\uparrow$ *carrier $Z = (S, 0, W, R, D, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealNullSequence}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed* $\mathsf{Real}^\uparrow$ *sequence row, zero* $\mathsf{Real}^\uparrow$ *row, finite* $\mathsf{StreamName}^\uparrow$ *windows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerances,* $\mathsf{RealSequenceLimit}^\uparrow$ *classifier, transport, replay, provenance, and local naming rows;*
2. *zero-row exposure, requiring every accepted limit read to expose the displayed zero* $\mathsf{Real}^\uparrow$ *row before the sequence-limit classifier is consumed;*
3. *finite-window convergence, requiring all convergence evidence to pass through the scheduled W, R, D, L rows;*
4. *transport and replay stability, requiring* hsame *transport and* Cont *replay to preserve the same sequence, zero, window, readback, tolerance, and classifier rows;*
5. *refusal of host sequence equality, quotient limits, countable choice, trichotomy, and ambient completeness as packet coordinates.*

No zero-limit obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 2378.1. The accepted packet displays exactly the sequence row, zero row, finite windows, regular readbacks, dyadic tolerances, convergence classifier, transport, replay, provenance, and local name.

The analytic read is then forced by the row order. The sequence row S and zero row 0 are inspected only through the scheduled windows W, readbacks R, and tolerance ledger D, and the convergence classifier L is the only row that can certify approach to the displayed zero.

Transport and replay stability follow from the structural rows H and C. They preserve the same displayed rows componentwise and replay the same finite route before the local name is consumed.

Finally inspect the carrier inventory. Since host sequence equality, quotient limits, countable choice, trichotomy, and ambient $\mathsf{Real}^\uparrow$ completeness are not coordinates of Z, none can become a local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.RealNullSequenceUp.RealNullSequenceCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{RealNullSequence}^\uparrow}$.

Constructive story: A $\mathsf{RealNullSequence}^\uparrow$ packet is generated from a displayed RealUp sequence, a zero RealUp row, finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a RealSequenceLimit convergence classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealNullSequence}^\uparrow}$, seed)

Scope closed: Definition 2378.1 and Definition 2378.2 seed the finite zero-limit route for displayed real sequences.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealNullSequence}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close sequence algebra, Cauchy criteria, quotient limits, host sequence equality, countable choice, trichotomy, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-row exposure, finite-window convergence, transport stability, replay stability, and refusal rows for BEDC.Derived.RealNullSequenceUp.

Chapter 2379

A Concrete Naming Certificate for $\mathsf{FanModulusBoundary}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges constructive analysis, the $\mathsf{FanModulusBoundary}^{\uparrow}$ packet records the boundary at which a finite fan-bar depth becomes a compact-uniform-continuity modulus handoff. It sits over the finite bar packet of chapter 400, the bar-to-window schedule of chapter 1707, the compact-uniform-continuity consumer of chapter 388, and the uniform-modulus surface of chapter 369. The packet uses ProbeBundle routes for finite branch windows and finite compact-source probes; it is not open-cover compactness, not countable choice, not a selected branch, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2379.1 (Fan modulus boundary carrier)**.** *A* $\mathsf{FanModulusBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, M, W, Q, U, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *row, recording the binary fan, decidable bar classifier, finite depth ledger, and finite branch-prefix witness window.*
2. *M is the* $\mathsf{BishopFanModulus}^{\uparrow}$ *row, exposing the bar-depth-to-window-modulus schedule through StreamName, RegSeqRat, DyadicRatCore, and Real readback rows.*
3. *W is the boundary-window row. It records the finite branch-prefix window that may be handed to compact-uniform-continuity consumers without selecting an infinite branch.*
4. *Q is the* ProbeBundle *route row, aligning finite fan-prefix probes with the compact finite-net probes used by the compact-uniform-continuity source.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, recording the tolerance-only modulus schedule exported from the finite fan-bar boundary.*
6. *R is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *readback row, recording that the exported modulus is consumed only as a compact finite-net and pointwise-modulus handoff.*
7. *H records componentwise* hsame *transport for the fan row, Bishop fan modulus row, boundary window, ProbeBundle route, UniformModulus row, compact-uniform readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$F, M, W, Q \longmapsto U \longmapsto R.$$

9. *P is the* Pkg *provenance over* FanModulusBoundary$^{\uparrow}$, FanTheorem$^{\uparrow}$, BishopFanModulus$^{\uparrow}$, CompactUniformContinu UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{FanModulusBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, M, W, Q, U, R, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FanModulusBoundaryUp.FanModulusBoundaryTasteGate single carrier alignment`

Theorem 2379.2 (FanModulusBoundaryUp NameCert obligations)**.** *For every accepted* FanModulusBoundary$^{\uparrow}$ *carrier*

$$B = (F, M, W, Q, U, R, H, C, P, N),$$

the local NameCert$_{\text{FanModulusBoundary}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite fan row, Bishop fan modulus row, boundary window, ProbeBundle route, UniformModulus consumer, compact-uniform readback, transport, replay, provenance, and local naming rows;*

2. *fan-bar depth exactness, requiring every finite depth used by W to be read through the displayed* FanTheorem$^{\uparrow}$ *bar classifier and witness window;*

3. *Bishop modulus boundary exactness, requiring the window schedule in M to be the only Real and Completion-facing modulus source exposed to U;*

4. *ProbeBundle compatibility, requiring Q to align finite fan-prefix probes with compact finite-net probes before the compact-uniform-continuity readback R is public;*

5. hsame *and* Cont *transport through H and C, preserving the same fan-bar, modulus, ProbeBundle, UniformModulus, and compact-uniform rows under replay;*

6. *ledger nonescape through P and N, excluding open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, and ambient* Real$^{\uparrow}$ *completeness.*

No fan modulus boundary obligation is stored outside these rows.

Closure status for NameCert$_{\text{FanModulusBoundary}^{\uparrow}}$.

Constructive story: A FanModulusBoundary$^{\uparrow}$ packet is generated from a FanTheoremUp finite bar row, BishopFanModulusUp bar-to-window schedule, finite boundary windows, ProbeBundle compatibility, UniformModulus handoff, CompactUniformContinuity readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FanModulusBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 2379.1 and Definition 2379.2 seed the finite fan-bar modulus boundary consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: FormalStatus(NameCert$_{\text{FanModulusBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected branches, selected limits, quotient stream equality, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-bar depth exactness, Bishop modulus boundary exactness, ProbeBundle compatibility, UniformModulus handoff, compact-uniform readback, transport, replay, provenance, local naming, and ledger nonescape for BEDC.Derived.FanModulusBoundaryUp.

Chapter 2380

A Concrete Naming Certificate for $\mathsf{PhysicalInduction}^{\uparrow}$

The $\mathsf{PhysicalInduction}^{\uparrow}$ packet realizes the open-continuation discipline of section 9414.25 as a finite audit-ledger induction carrier. It sits beside the physical model audit of chapter 1129, the ontic source surface of chapter 1125, the state-space model carrier of chapter 329, and the observer-budget support packet of chapter 1121. Those sibling rows supply audit comparison, reality-facing source, model-state presentation, and observer-bounded access; this packet binds an observation row to a hypothesis row, falsifier ledger, step transport row, and local naming row before any induction read is accepted.

Definition 2380.1 (PhysicalInduction carrier)**.** *A* $\mathsf{PhysicalInduction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, H, A, S, B, F, T, R, P, N)$$

with the following displayed rows:

1. *O is the observation row, read through the* $\mathsf{OnticState}^{\uparrow}$ *source and the* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *support boundary.*
2. *H is the hypothesis row, read through the* $\mathsf{StateSpaceModel}^{\uparrow}$ *model-state presentation and the displayed comparison bundle.*
3. *A is the* $\mathsf{PhysicalModelAudit}^{\uparrow}$ *row recording the accepted audit comparison between observation and hypothesis data.*
4. *S is the step row naming the finite transition from the current observed prefix to the next licensed prediction read.*
5. *B is the observer-budget row specifying which bounded observation support may inspect the step.*
6. *F is the falsifier ledger, naming finite observations, signature mismatches, or budget failures that defeat the proposed step.*
7. *T records componentwise* hsame *transport for observation, hypothesis, audit, step, budget, falsifier, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from observation and hypothesis rows through the audit row and into the finite step boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{PhysicalInduction}^{\uparrow}$, $\mathsf{PhysicalModelAudit}^{\uparrow}$, $\mathsf{OnticState}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{ObserverBudgetSupport}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhysicalInduction}^{\uparrow}}$ *row.*

The carrier has no coordinate for a final induction oracle, total future fit, host physical truth, unbounded observation, detached model equality, or an induction step outside the displayed audit and falsifier ledger.

Lean4 ✓ `BEDC.Derived.PhysicalInductionUp.PhysicalInductionTasteGate single carrier alignment`

Theorem 2380.2 (PhysicalInduction NameCert obligations)**.** *For every accepted* $\mathsf{PhysicalInduction}^{\uparrow}$ *carrier* $I = (O, H, A, S, B, F, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PhysicalInduction}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the observation row, hypothesis row, physical-model audit row, finite step row, observer-budget row, falsifier ledger, transport, replay, provenance, and local naming row;*
2. *observation exposure, requiring O to be read through* $\mathsf{OnticState}^{\uparrow}$ *and* $\mathsf{ObserverBudgetSupport}^{\uparrow}$ *before a hypothesis step is consumed;*
3. *hypothesis and audit exposure, requiring H and A to display the state-space model row and audit comparison used by the proposed step;*
4. *falsifier-ledger exposure, requiring F to name the finite observations, signature mismatches, or support failures that defeat the step;*
5. *step transport stability, requiring T, R, P, N to preserve the same observation, hypothesis, audit, budget, and falsifier rows under accepted replay.*

No accepted physical-induction read may bypass any one of these rows.

Proof. Carrier admission is projection from Definition 2380.1. The accepted packet displays the observation, hypothesis, audit, step, observer-budget, falsifier, transport, replay, provenance, and local naming rows as its complete surface.

The observation and hypothesis rows are read before the step row. The observation row is bounded by the ontic-source and observer-budget siblings, while the hypothesis row is bounded by the displayed state-space model presentation.

The audit and falsifier rows are the decisive middle boundary. The audit row states which comparison licenses the step, and the falsifier ledger states which finite observations or support failures defeat it.

Finally the structural rows T, R, P, N transport, replay, source, and name the same packet. Since no further coordinate exists, an accepted induction read cannot detach the step from its observation, hypothesis, audit, budget, or falsifier ledger. □

Falsifiable prediction: Every accepted $\mathsf{PhysicalInduction}^{\uparrow}$ read must expose an observation row, hypothesis row, falsifier ledger, step transport row, and local NameCert row; a read omitting any one of these rows is outside the carrier.

Independence witness: physical-model-audit, ontic-state, state-space-model, observer-budget-support: none is bijective because $\mathsf{PhysicalInduction}^{\uparrow}$ binds observation, hypothesis, audit, observer budget, falsifier ledger, step transport, replay, provenance, and local NameCert rows in one packet. PhysicalModelAuditUp audits a comparison without isolating the induction step; OnticStateUp exposes the source without the hypothesis row; StateSpaceModelUp exposes model state without the falsifier ledger; ObserverBudgetSupportUp bounds access without licensing a hypothesis step.

The Lean follow-up must provide `Nontrivial PhysicalInductionUp` and `FieldFaithful PhysicalInductionUp` instances in `lean4/BEDC/Derived/PhysicalInductionUp/TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{PhysicalInduction}^{\uparrow}}$.

Constructive story: A $\mathsf{PhysicalInduction}^{\uparrow}$ packet is generated from observation, hypothesis, physical-model audit, finite step, observer-budget, falsifier-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhysicalInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2380.1 and Definition 2380.2 seed physical induction as a finite observation-step NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhysicalInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert a final induction oracle, total future fit, host physical truth, unbounded observation, Lean verification, or bridge checking for $\mathsf{PhysicalInduction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, observation exposure, hypothesis and audit exposure, falsifier-ledger exposure, step transport stability, nontriviality, and field faithfulness for BEDC.Derived.PhysicalInductionUp.

Chapter 2381

A Concrete Naming Certificate for $\mathsf{CompletionIdempotence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompletionIdempotence}^{\uparrow}$ packet records the finite reflector surface by which a displayed completion request over an already completed name is read back through the same completion boundary. It sits over the Cauchy completion functor of chapter 423, the metric completion surface of chapter 1506, the separated completion seal of chapter 1612, and the uniform completion route of chapter 1473. The packet is a reuse certificate for completed names; it does not construct a second quotient object, select representatives, assert host equality, invoke choice, or claim an ambient completeness theorem.

Definition 2381.1 (Completion idempotence carrier)**.** *A* $\mathsf{CompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, R, Q, W, D, G, A, C, P, N)$$

with the following displayed rows:

1. *F is the first completion seal, naming the completed boundary already available to downstream consumers.*
2. *R is the second completion request row, recording that a consumer asks to complete the already sealed boundary.*
3. *Q is the Cauchy readback row, requiring every second request to read the same finite Cauchy evidence used by F.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the finite windows through which the repeated completion request is observed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger tying the first seal and the second request to the same dyadic comparison budget.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, keeping regular rational readback attached to the displayed Cauchy data.*
7. *A is the classifier transport row, recording componentwise* hsame *movement for the first seal, second request, Cauchy readback, window, tolerance, and handoff rows.*
8. *C is the displayed* Cont *replay row from the second completion request through R, Q, W, D, G, A back to the first completion seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionIdempotence}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a second completion object, quotient class, selected representative, host equality, choice principle, detached Cauchy witness, or unrecorded completeness assertion.

Lean4 ✓ `BEDC.Derived.CompletionIdempotenceUp.CompletionIdempotenceTasteGate single carrier alignment`

Theorem 2381.2 (CompletionIdempotence NameCert obligations)**.** *For every accepted* $\mathsf{CompletionIdempotence}^{\uparrow}$ *carrier* $I = (F, R, Q, W, D, G, A, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompletionIdempotence}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the first completion seal, second completion request, Cauchy readback, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, classifier transport, continuation replay, provenance, and local naming row;*
2. *first-seal visibility, requiring F to be read before the second request is admitted;*
3. *repeated-request exactness, requiring R and Q to reuse the Cauchy data already attached to the first seal;*
4. *window and tolerance stability, requiring W, D, G to keep finite observation windows, dyadic tolerances, and regular rational handoff on the same displayed route;*
5. *classifier and replay closure, requiring A, C, P, N to transport, replay, source, and name only the displayed rows;*
6. *nonescape, excluding a second completion object, quotient class, selected representative, host equality, choice principle, detached Cauchy witness, and unrecorded completeness assertion.*

No completion idempotence obligation is stored outside these rows.

Proof. Project the carrier of Definition 2381.1. The only rows available to the local certificate are $F, R, Q, W, D, G, A, C, P, N$.

The first-seal and repeated-request obligations follow by reading F before R. The Cauchy readback row Q attaches the second request to the same displayed evidence rather than to a fresh completion object.

The window, tolerance, and handoff obligations are carried by W, D, G. They keep the observed StreamName windows, DyadicRatCore tolerances, and RegSeqRat readback on the route already named by the first seal.

The structural rows A, C, P, N transport, replay, source, and name the displayed packet. Since every excluded object is absent from the tuple, the certificate cannot export a second completion object, quotient class, selected representative, host equality, choice principle, detached Cauchy witness, or unrecorded completeness assertion. □

Theorem 2381.3 (Completion idempotence stable reflection)**.** *Let* $I = (F, R, Q, W, D, G, A, C, P, N)$ *be an accepted* $\mathsf{CompletionIdempotence}^{\uparrow}$ *carrier. Every downstream completion consumer that reads the second request factors through the finite route*

$$F \longmapsto R, Q \longmapsto W, D, G \longmapsto A, C, P, N.$$

Thus a completed name can be reused through its first completion seal, Cauchy readback, finite StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, classifier transport, continuation replay, provenance, and local NameCert row, without constructing or naming a second completion object.

Proof. Start from an accepted repeated-completion read. Carrier admission exposes the first seal F and the second request R as rows of the same finite packet.

The Cauchy readback row Q ties the second request to the evidence already carried by F. The rows W, D, G then preserve finite observation windows, dyadic tolerances, and regular rational handoff along that same route.

Finally the classifier transport A, continuation replay C, provenance P, and local naming row N package the route for downstream consumers. None of these rows adds a second completion object, so the reflection is stable by exhaustion of the displayed carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionIdempotence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionIdempotence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2381.1, Definition 2381.2, and Definition 2381.3 seed the finite completion-idempotence reflector over first completion seals, repeated requests, Cauchy readback, StreamName windows, DyadicRatCore tolerances, RegSeqRat handoff, classifier transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionIdempotence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, quotient completion classes, selected representatives, host equality, choice principles, detached Cauchy witnesses, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, repeated-request exactness, window and tolerance stability, classifier transport, replay closure, and nonescape for BEDC.Derived.CompletionIdempotenceUp.

Constructive story: A $\mathsf{CompletionIdempotence}^{\uparrow}$ packet is generated from a first completion seal, a second completion request, Cauchy readback, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat handoff, classifier transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2382

A Concrete Naming Certificate for $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$

The $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ packet realizes the confluence audit-map boundary of chapter 9378 as finite frontier critical-pair data. It reads the critical-pair boundary of chapter 2212, the frontier dependency matrix of chapter 2307, the closed-normal fragment of chapter 737, the bounded normal-equality checker of chapter 752, and the subject-reduction route classifier of chapter 785. The packet names local frontier joins for audit consumers; it does not assert global confluence, arbitrary closed-term Church–Rosser, residual-substitution compatibility, or unchecked subject-reduction discharge.

Falsifiable prediction: Every accepted $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ packet routes confluence audit reads through redex-frontier, finite critical-pair, local-join, bounded-checker, substitution-boundary, transport, replay, provenance, and local naming rows.

Independence witness: metacic-confluence-critical-pair-boundary, metacic-frontier-dependency-matrix, metacicconfluencenormalfragment, bounded-normal-equality-checker, and subject-reduction-route-classifier: these siblings respectively expose common-reduct boundary rows, frontier dependency accounting, closed-normal fragments, bounded normal-form comparison, and typing-route separation; none is bijective with the packet that jointly records redex-frontier critical pairs, local joins, substitution-boundary refusal, transport, replay, provenance, and local naming.

Definition 2382.1 (MetaCIC frontier critical-pair carrier)**.** *A* $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, F, J, E, U, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the redex-frontier row read from the finite MetaCIC frontier cases.*
2. *F is the finite critical-pair table row, listing the frontier overlaps accepted by the packet.*
3. *J is the local-join row, recording displayed join candidates for each accepted frontier critical pair.*
4. *E is the* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$ *row used when a local join is checked by bounded normal equality.*
5. *U is the substitution-boundary row, recording that residual-substitution compatibility and arbitrary closed substitution are not supplied by the packet.*

6. *S is the subject-reduction route-classifier row, keeping typing discharge separate from confluence audit reads.*
7. *H records componentwise* hsame *transport for frontier, critical-pair, local-join, bounded-checker, substitution-boundary, route-classifier, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an audit-map request through the frontier table, local join, bounded checker, substitution boundary, and route classifier.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$, $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}}$ *row.*

The carrier contains no coordinate for global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, or an audit-map consumer that bypasses the displayed frontier table.

Lean4 ✓ `BEDC.Derived.MetaCICFrontierCriticalPairUp.MetaCICFrontierCriticalPairTasteGate single carrier alignment`

Theorem 2382.2 (MetaCIC frontier critical-pair NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ *carrier* $K = (R, F, J, E, U, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the redex-frontier row, finite critical-pair table, local-join row, bounded-checker row, substitution-boundary row, route-classifier row, transport, replay, provenance, and local naming row;*
2. *frontier enumeration, requiring every accepted critical pair to be read from R and F before a local join is consulted;*
3. *local-join exactness, requiring J to display the join candidate before the bounded checker E is used;*
4. *substitution-boundary visibility, requiring U to remain visible whenever the audit-map boundary is cited;*
5. *route separation, requiring S to keep subject-reduction discharge outside the confluence critical-pair route;*
6. *nonescape, excluding global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, and audit-map reads outside $R, F, J, E, U, S, H, C, P, N$.*

No frontier critical-pair obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 2382.1. The projection exposes exactly $R, F, J, E, U, S, H, C, P, N$.

Frontier enumeration reads R and F first. Therefore the packet can inspect only finite frontier overlaps that appear in the displayed critical-pair table.

The local-join obligation reads J before the bounded checker E. This keeps bounded normal-equality checking downstream of a displayed join candidate rather than turning the checker into a global confluence proof.

The boundary and route-separation obligations are carried by U and S. They keep substitution compatibility and subject-reduction discharge outside the packet's confluence route.

The structural rows H, C, P, N transport, replay, source, and name only the displayed route. Since the excluded confluence and typing claims are absent from the carrier, the local NameCert surface is exhausted by the stated rows. □

Theorem 2382.3 (MetaCIC frontier critical-pair route)**.** *Every confluence audit-map read from an accepted* $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ *packet factors through the finite route*

$$R \longmapsto F \longmapsto J \longmapsto E \longmapsto U, S \longmapsto H, C, P, N.$$

Thus the packet supplies a redex-frontier, finite critical-pair, local-join, bounded-checker, substitution-boundary, transport, replay, provenance, and local naming handle for the audit map, without claiming global confluence or discharging the substitution boundary.

Proof. An audit-map read begins at the redex-frontier row R and the finite critical-pair table F, because no unfrontiered confluence coordinate appears in the carrier.

For an accepted pair, the local-join row J is read before the bounded checker E. The checker is therefore attached to displayed local joins, not to arbitrary closed terms.

The substitution-boundary row U and route-classifier row S remain visible before the structural rows package the read. The rows H, C, P, N preserve and name the same finite route, so the packet remains a critical-pair frontier handle rather than a global confluence theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2382.1, Definition 2382.2, and Definition 2382.3 seed the finite MetaCIC frontier critical-pair route over redex-frontier rows, critical-pair tables, local joins, bounded checking, substitution-boundary refusal, route separation, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, frontier enumeration, local-join exactness, substitution-boundary visibility, route separation, and nonescape for BEDC.Derived.MetaCICFrontierCriticalPairUp.

Constructive story: A $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$ packet is generated from redex-frontier rows, finite critical-pair rows, local-join rows, a bounded-checker row, a substitution-boundary row, a subject-reduction route-classifier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2383

A Concrete Naming Certificate for $\mathsf{FixedPointIteration}^{\uparrow}$

Following Banach's fixed-point iteration and Bishop–Bridges constructive analysis, the $\mathsf{FixedPointIteration}^{\uparrow}$ packet records the finite orbit ledger read between the contraction-map surface of chapter 1418, the one-step Picard route of chapter 395, the complete-metric handoff of chapter 128, the metric source of chapter 57, and the endpoint fixed-point certificate of chapter 396. The packet names displayed iterate rows, residual budgets, transport, continuation, provenance, and local naming before any completed endpoint row is consumed.

Definition 2383.1 (FixedPointIteration carrier)**.** *A* $\mathsf{FixedPointIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, d, T, x_0, I, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the admitted point classifier for the iteration domain.*
2. *d is the distance row used to read finite adjacent-iterate distances inside X.*
3. *T is the self-map graph row supplied by the* $\mathsf{ContractionMapping}^{\uparrow}$ *surface and read one application at a time.*
4. *x_0 is the initial-point row from which the finite orbit is generated.*
5. *I is the finite iterate ledger, listing displayed rows $x_0, x_1, \ldots, x_n$ with application witnesses that each x_{k+1} is read from T at x_k.*
6. *R is the residual budget row, recording the finite distance and contraction-ratio estimates available for the displayed orbit window.*
7. *H records componentwise* hsame *transport for the metric source, distance row, self-map graph, initial point, iterate ledger, residual budget, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for extending a finite orbit window by one certified application row and for handing the finite residual rows to completion-facing consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FixedPointIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected fixed point, a completed limit, host function equality, or ambient completeness outside the displayed completion handoff.

Lean4 ✓ `BEDC.Derived.FixedPointIterationUp.FixedPointIterationTasteGate single carrier alignment`

Theorem 2383.2 (FixedPointIteration NameCert obligations)**.** *For an accepted* $\mathsf{FixedPointIteration}^{\uparrow}$ *carrier* $Q = (X, d, T, x_0, I, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FixedPointIteration}^{\uparrow}}$ *surface consists exactly of carrier admission, initial-point admission, self-map graph readability, finite iterate-ledger formation, residual-budget exposure, componentwise transport, continuation replay, package provenance stability, and local naming for the carrier of Definition 2383.1. No obligation is stored outside those displayed rows.*

Proof. Carrier admission projects the rows $X, d, T, x_0, I, R, H, C, P, N$ from the finite packet. The initial-point and self-map obligations read x_0 and T against the metric-source classifier before any iterate is formed.

The iterate-ledger obligation then checks each displayed application row in I. The residual-budget obligation reads only the finite adjacent-distance and contraction-ratio rows in R, so it does not require a completed limit object.

Transport and replay are supplied by H and C, while P and N record provenance and local naming. Since the carrier has no other coordinate, the public NameCert surface is exactly the listed finite iteration-orbit surface. □

Theorem 2383.3 (FixedPointIteration residual budget)**.** *Every accepted* $\mathsf{FixedPointIteration}^{\uparrow}$ *carrier exposes a residual budget only for the displayed finite orbit window* I*. The row* R *may be consumed as finite evidence for adjacent-distance control, tail-radius comparison over the displayed window, and contraction-ratio bookkeeping, but it is not a completed Cauchy sequence, not a selected endpoint, and not a proof of host fixed-point equality.*

Proof. Project the carrier through Definition 2383.1. The only residual coordinate is R, and it is attached to the finite iterate ledger I rather than to an infinite sequence object.

Read the contraction source through chapter 1418 and the one-step route through chapter 395. These surfaces supply adjacent application and distance comparisons for displayed rows only.

Thus a consumer may use R as finite residual bookkeeping for the orbit window. Any completed endpoint, Cauchy-limit object, or fixed-point equality must be supplied by a separate completion or fixed-point packet. □

Theorem 2383.4 (FixedPointIteration complete-metric handoff)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{BanachFixedPoint}^{\uparrow}$ *consumer of a* $\mathsf{FixedPointIteration}^{\uparrow}$ *packet must pass through the finite route*

$$X, d, T, x_0, I, R \longmapsto H, C, P, N.$$

The iteration packet supplies the orbit and residual rows consumed by completion-facing certificates, but the limit witness, endpoint seal, and classifier-fixed endpoint row remain outside $\mathsf{FixedPointIteration}^{\uparrow}$ *until the complete-metric and Banach fixed-point packets provide them.*

Proof. Start with an accepted carrier. The metric source X, distance row d, self-map row T, and initial point x_0 generate only the displayed ledger I and residual row R.

The continuation row C replays this finite route for a completion-facing consumer, and the transport row H preserves the same componentwise readings under the packet classifier. The provenance and naming rows P, N record that this route is the one exported by the packet.

Completeness and endpoint fixedness are therefore not read from $\mathsf{FixedPointIteration}^{\uparrow}$. They enter only when a $\mathsf{CompleteMetric}^{\uparrow}$ or $\mathsf{BanachFixedPoint}^{\uparrow}$ packet consumes the displayed finite orbit and supplies its own limit or endpoint rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FixedPointIteration}^{\uparrow}}$.

Constructive story: A $\mathsf{FixedPointIteration}^{\uparrow}$ packet is generated from a MetricUp source row, a distance row, a ContractionMapping self-map graph, an initial point, a finite iterate ledger, a residual budget, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FixedPointIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2383.1, Definition 2383.2, Definition 2383.3, and Definition 2383.4 seed the finite iteration-orbit packet over metric, contraction-map, Picard-step, complete-metric, fixed-point, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FixedPointIteration}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FixedPointIterationUp.FixedPointIterationTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close selected fixed points, completed limits, host fixed-point equality, ambient completeness outside the carried handoff, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, iterate-ledger formation, residual-budget exposure, completion handoff, transport stability, replay stability, provenance stability, and local naming for BEDC.Derived.FixedPointIterationUp.

Chapter 2384

A Concrete Naming Certificate for $\mathsf{RaikovCompletion}^{\uparrow}$

Following Raikov and the Bourbaki–Kelley treatment of two-sided uniform completion, the $\mathsf{RaikovCompletion}^{\uparrow}$ packet records the completion route for a separated uniform group as finite BEDC NameCert data. It sits after the entourage carrier of chapter 339, the separated comparison boundary of chapter 1876, the Cauchy-filter surface of chapter 333, and the completion handoff of chapter 1473. The packet names the two-sided Cauchy-filter route without quotient representatives, selected limits, host equality of completion points, trichotomy, countable choice, or ambient complete-group structure.

Definition 2384.1 (Raikov completion carrier)**.** *A* $\mathsf{RaikovCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, U, S, L, R, F, C, H, P, N)$$

with the following displayed rows:

1. *G is the source group row, read only through the finite multiplication, inverse, and unit observations needed by uniform consumers.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row on the source, carrying both left and right translate requests as displayed finite entourage data.*
3. *S is the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *comparison row, recording that the diagonal boundary is separated before completion-facing reads are exported.*
4. *L is the left Cauchy-filter ledger and R is the right Cauchy-filter ledger, each read through the* $\mathsf{CauchyFilter}^{\uparrow}$ *surface.*
5. *F is the two-sided compatibility row asserting that the same displayed filter packet satisfies the left and right Cauchy requests after common finite refinement.*
6. *C is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row consumed by completion-facing readers only after U, S, L, R, F have been displayed.*
7. *H records componentwise* hsame *transport for the group, uniformity, separated comparison, two Cauchy-filter ledgers, compatibility, handoff, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RaikovCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RaikovCompletion}^{\uparrow}}$ *row.*
The carrier predicate is exactly the assertion that these rows form one two-sided uniform completion packet. It has no coordinate for quotient representatives, selected limits, host equality, trichotomy, choice, or an ambient completed group.

Lean4 (definition) `BEDC.Derived.RaikovCompletionUp.RaikovCompletionUp`
Lean4 ✓ `BEDC.Derived.RaikovCompletionUp.RaikovCompletionTasteGate single carrier alignment`

Theorem 2384.2 (Raikov completion NameCert obligations)**.** *For every accepted* $\mathsf{RaikovCompletion}^{\uparrow}$ *carrier* $K = (G, U, S, L, R, F, C, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RaikovCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, left and right Cauchy-filter visibility, two-sided common-refinement compatibility, separated-boundary visibility, uniform-completion handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Every obligation is read through the finite rows

$$G, U, S, L, R, F, C, H, P, N.$$

No obligation is stored as a quotient completion, a chosen representative, a selected limit, host equality of completion points, trichotomy, countable choice, or an ambient complete-group principle.

Proof. Project the carrier of Definition 2384.1. The projection exposes the source group row, uniform entourage row, separated comparison row, left and right Cauchy-filter ledgers, two-sided compatibility row, completion handoff, transport row, provenance row, and local naming row.

The left and right Cauchy obligations are exactly the rows L and R. The compatibility row F is downstream of both, so a completion-facing consumer cannot read only one side of the uniform structure and call it a Raikov route.

The separated boundary is the row S, and the completion handoff is the row C. The replay data require the route through U, S, L, R, F before C is available, while H, P, N preserve transport, provenance, and local naming for the same finite packet.

Since the carrier has no quotient, representative-selection, host-equality, trichotomy, choice, or ambient complete-group coordinate, none of the listed obligations can export such data. □

Theorem 2384.3 (Raikov completion two-sided Cauchy-filter route)**.** *Every completion-facing read of an accepted* $\mathsf{RaikovCompletion}^{\uparrow}$ *packet factors through the finite two-sided route*

$$\mathsf{UniformSpace}^{\uparrow} \longrightarrow \mathsf{SeparatedUniformSpace}^{\uparrow} \longrightarrow (\mathsf{CauchyFilter}^{\uparrow}_{\mathrm{left}}, \mathsf{CauchyFilter}^{\uparrow}_{\mathrm{right}}) \longrightarrow \mathsf{UniformCompletion}^{\uparrow}.$$

Equivalently, a consumer first reads the displayed uniform entourage row U, then the separated comparison row S, then the paired left and right Cauchy-filter ledgers L, R together with their compatibility row F, and only then the completion handoff row C. The route exports no quotient representative, chosen limit, host equality of completion points, trichotomy, countable choice, or ambient complete-group object.

Proof. Start with an accepted carrier $K = (G, U, S, L, R, F, C, H, P, N)$. The only uniform data available to a completion-facing consumer is the displayed row U, and the only separated comparison data is the displayed row S.

The Cauchy-filter part of the route is two-sided. The left ledger L and right ledger R are both required before the compatibility row F can be read, so the packet does not expose a one-sided completion boundary.

The completion handoff is the row C. By Definition 2384.2, C is downstream of the uniform, separated, left-filter, right-filter, and compatibility rows, with the same hsame, Cont, Pkg, and local NameCert provenance.

Thus every completion-facing read factors through the displayed two-sided Cauchy-filter route. The exclusions follow from the carrier inventory in Definition 2384.1. □

Closure status for $\mathsf{NameCert}_{\mathsf{RaikovCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RaikovCompletion}^{\uparrow}$ packet is generated from a source group row, UniformSpaceUp entourage row, SeparatedUniformSpaceUp comparison row, left and right CauchyFilterUp ledgers, a two-sided compatibility row, UniformCompletionUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RaikovCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2384.1, Definition 2384.2, and Definition 2384.3 seed the two-sided uniform completion route over UniformSpace, SeparatedUniformSpace, CauchyFilter, and UniformCompletion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RaikovCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, uniqueness of completion points, selected representatives, host equality, trichotomy, countable choice, ambient complete-group structure, or bridge checking.

Upgrade path: Formal verification remains the route from the paper carrier and two-sided handoff rows to checked theorem, axiom-clean, and bridge-checked status for BEDC.Derived.RaikovCompletionUp.

Chapter 2385

A Concrete Naming Certificate for $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$

Following Bishop and Bridges on compact intervals and located real-number approximation, the $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ packet records the finite dyadic-net data available for a located closed interval. It sits over the real-name endpoint seal of chapter 38, the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the closed bounded interval carrier of chapter 1591, and the compact-metric finite-net consumer of chapter 129. The packet names the constructive finite-net substitute for interval compactness; it does not assert unrestricted open-cover compactness or a general Heine–Borel theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$.

Definition 2385.1 (Heine–Borel located interval carrier)**.** *A* $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, E, R, S, D, F, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing lower and upper endpoint names, interval membership, and boundedness witnesses.*
2. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row, read only through the finite endpoint windows attached to I.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row naming the requested positive approximation scale and its finite readback.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected endpoint and mesh windows before any coverage handoff.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision ledger, recording the dyadic mesh, endpoint-radius comparisons, and inclusion checks for every displayed cell.*
6. *F is the finite-net row, consisting of finitely many dyadic centers, their radius budget, and the cell-coverage witnesses obtained from I, E, R, S, D.*
7. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row read after the finite-net row has been displayed.*
8. *H records componentwise* hsame *transport for the interval, endpoint seal, tolerance, window, dyadic, finite-net, compact-metric handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from interval endpoint inspection through dyadic subdivision, finite-net coverage, compact-metric handoff, and local naming.*

10. *P is the* Pkg *provenance over* HeineBorelLocatedInterval$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HeineBorelLocatedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, E, R, S, D, F, K, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcover functionals, quotient real equality, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineBorelLocatedIntervalUp.TasteGate.HeineBorelLocatedIntervalTasteGate`
`single carrier alignment`

Theorem 2385.2 (Heine–Borel located interval finite-net obligation)**.** *For every accepted* HeineBorelLocatedInterva *carrier $L = (I, E, R, S, D, F, K, H, C, P, N)$, the finite-net obligation is exactly the route*

$$I, E, R, S, D \longmapsto F \longmapsto K, H, C, P, N.$$

Thus a compactness-facing consumer may read only the displayed closed bounded interval source, endpoint seals, regular rational tolerance, finite stream windows, dyadic subdivision ledger, finite dyadic centers, compact-metric handoff, transport, replay, provenance, and local NameCert rows. No read exports unrestricted open-cover compactness, a selected finite subcover outside F, quotient equality of real names, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 2385.1. The interval source, endpoint seal, tolerance row, finite windows, and dyadic ledger are the only rows from which a net can be constructed.

The finite-net row F is read after those rows. Its centers and radius budget are displayed dyadic data, and each coverage witness is tied to the same endpoint, tolerance, stream-window, and dyadic comparison surface.

Finally K, H, C, P, N hand the displayed net to compact-metric consumers while preserving transport, replay, provenance, and local naming. Since no carrier coordinate stores arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient completeness, the local obligation surface exports none of those data. □

Theorem 2385.3 (Heine–Borel located interval compact-metric handoff)**.** *Every* CompactMetric$^{\uparrow}$ *consumer of an accepted* HeineBorelLocatedInterval$^{\uparrow}$ *packet must first display the finite dyadic-net row F together with its source rows I, E, R, S, D and structural rows H, C, P, N. The handoff is a finite located closed-interval coverage certificate over* ClosedBoundedInterval$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, *and* DyadicRatCore$^{\uparrow}$*; it is not an open-cover compactness theorem, a selected subcover principle, a quotient-completion statement, or an ambient completeness assertion.*

Proof. Start from the finite-net obligation of Definition 2385.2. It forces the compact-metric consumer to read the interval, endpoint, tolerance, window, dyadic, and finite-net rows before K is available.

Read the compact-metric handoff row K next. This row packages only the displayed centers, mesh radius, and cell-coverage witnesses already stored in F; it does not add a host compactness theorem or a cover selector.

The transport and replay rows H, C keep the same finite dyadic data stable under accepted componentwise reads, while P, N bind provenance and local naming. Hence the handoff remains a finite located interval certificate and cannot export open-cover compactness, selected subcovers, quotient completion, or ambient completeness. □

Falsifiable prediction: A compact-metric consumer that reads $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ without displaying the closed bounded interval, Real endpoint seals, regular rational tolerance, stream windows, DyadicRatCore subdivision ledger, finite dyadic net, compact-metric handoff, transport, replay, provenance, and local NameCert rows refutes the carrier surface.

Independence witness: closedboundedinterval, real, regseqrat, streamname, dyadicratcore, compactmetric: no listed sibling is bijective to $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ because this packet binds located closed-interval endpoints, tolerance readback, finite windows, dyadic subdivision, finite-net coverage, compact-metric handoff, transport, replay, provenance, and local naming in one interval-coverage interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelLocatedInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, RealUp endpoint seals, a RegSeqRatUp tolerance row, StreamNameUp finite windows, a DyadicRatCoreUp subdivision ledger, a finite dyadic-net row, a CompactMetricUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelLocatedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2385.1, Definition 2385.2, and Definition 2385.3 seed the finite located closed-interval net interface for compact-metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelLocatedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted open-cover compactness, general Heine–Borel compactness for subsets of $\mathsf{Real}^{\uparrow}$, selected subcovers, quotient real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sealing, tolerance readback, finite-window replay, dyadic subdivision, finite-net coverage, compact-metric handoff, transport, provenance, and local naming for BEDC.Derived.HeineBorelLocatedIntervalUp.

Chapter 2386

A Concrete Naming Certificate for DyadicCauchyModulusLattice$^{\uparrow}$

Following Bishop and Bridges, the DyadicCauchyModulusLattice$^{\uparrow}$ packet records the finite lattice surface by which dyadic tolerance requests, Cauchy convergence moduli, finite stream windows, regular rational readback, and Real seal handoff are compared without quotient streams or selected limits. It sits over the dyadic comparison ledger of chapter 291, the convergence-modulus threshold route of chapter 1438, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal boundary of chapter 38. The packet is a directed Real and Completion input surface; it is not a quotient completion object, not a host modulus function, and not a choice principle.

Definition 2386.1 (Dyadic Cauchy modulus lattice carrier)**.** *A* DyadicCauchyModulusLattice$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, M, W, Q, R, J, H, C, P, N)$$

with the following visible rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row naming the dyadic precision requests and their comparison ledger.*
2. *M is the* CauchyConvergenceModulus$^{\uparrow}$ *threshold row assigning accepted tail windows to the dyadic requests.*
3. *W is the* StreamName$^{\uparrow}$ *finite-window row used by the modulus threshold.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed through the window.*
5. *R is the* Real$^{\uparrow}$ *seal handoff row, available only after D, M, W, Q have been displayed.*
6. *J is the directed lattice row, recording finite meet and weak-later refinement comparisons between displayed dyadic-modulus requests.*
7. *H records componentwise* hsame *transport for tolerance, modulus, window, readback, seal, lattice, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a dyadic precision request through $D, M, W, Q, R, J, H, P, N$.*
9. *P is the* Pkg *provenance over* DyadicCauchyModulusLattice$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyModulusLattice}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, M, W, Q, R, J, H, C, P, N$. It has no coordinate for quotient stream equality, selected limit, host real equality, ambient Real completeness, countable choice, or a host-level modulus function.

Lean4 ✓ `BEDC.Derived.DyadicCauchyModulusLatticeUp.DyadicCauchyModulusLatticeTasteGate single carrier alignment`

Theorem 2386.2 (Dyadic Cauchy modulus lattice NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ *carrier $L = (D, M, W, Q, R, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyModulusLattice}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, convergence-modulus threshold exposure, finite-window readback, regular rational handoff, directed lattice comparison, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$D, M, J \longmapsto W, Q \longmapsto R \longmapsto H, C, P, N$$

before using the lattice as evidence for a dyadic Cauchy modulus comparison.

Proof. Project the carrier of Definition 2386.1. The rows D and M expose the dyadic request and the convergence-modulus threshold assigned to that request, while J records only finite directed comparisons between such displayed rows.

The finite observation stage is then forced through W and Q. The window row supplies the inspected $\mathsf{StreamName}^{\uparrow}$ segment, and the readback row supplies the $\mathsf{RegSeqRat}^{\uparrow}$ route consumed by later Real and Completion arguments.

Finally read R, H, C, P, N. The Real seal row is downstream of the displayed dyadic, modulus, lattice, window, and readback rows, while the structural rows transport, replay, source, and name the same finite route. Since no carrier coordinate supplies quotient streams, selected limits, countable choice, host real equality, ambient Real completeness, or a host modulus function, the listed obligations exhaust the local naming surface. □

Theorem 2386.3 (Dyadic Cauchy modulus lattice tail refinement)**.** *Let $L = (D, M, W, Q, R, J, H, C, P, N)$ be an accepted* $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ *packet. If two dyadic tolerance requests are related by the directed lattice row J, then the refined request is served by a weakly later* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *threshold, a weakly later* $\mathsf{StreamName}^{\uparrow}$ *finite window, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary. The refinement route is*

$$D, M, J \longmapsto W, Q \longmapsto R \longmapsto H, C, P, N.$$

It does not select a stream representative, choose a limit, or invoke ambient Real completeness.

Proof. Read the dyadic tolerance row D together with the directed lattice row J. A refinement in J can compare only displayed dyadic-modulus requests, so the refined request remains inside the finite carrier rows.

Use the threshold row M to pass from the refined dyadic request to a weakly later Cauchy convergence threshold. The finite-window row W and readback row Q then preserve the same $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route, changing only by moving to a weakly later accepted window.

The Real-facing endpoint is still the row R, followed by H, C, P, N. Because the route has no quotient stream coordinate, selected-limit coordinate, choice schedule, or ambient completeness coordinate, the tail refinement is exactly the displayed finite lattice refinement. □

Theorem 2386.4 (Dyadic Cauchy modulus lattice common-refinement exhaustion)**.** *Let* $L = (D, M, W, Q, R, J, H, C, P, N)$ *be an accepted* $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ *packet. If two dyadic tolerance requests are accepted by the same packet, then any common refinement visible to a Real or Completion consumer must be a displayed row of the directed lattice component* J*. After that refinement is chosen inside* J*, every consumer still factors through the same route*

$$D, M, J \longmapsto W, Q \longmapsto R \longmapsto H, C, P, N.$$

Thus common-refinement reads cannot pass through quotient streams, selected limits, host modulus functions, or ambient Real completeness.

Proof. Project the carrier of Definition 2386.1. The only component that compares dyadic tolerance requests is J, and it records finite meet and weak-later comparisons between requests already displayed by D and served by M.

Apply Definition 2386.3 to the first request and then to the second request. Both refinements remain on weakly later $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ thresholds, finite $\mathsf{StreamName}^{\uparrow}$ windows, the same $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the same downstream $\mathsf{Real}^{\uparrow}$ seal boundary.

Now use Definition 2386.5. The Real-facing handoff is available only after the dyadic tolerance, convergence-modulus, lattice, window, and readback rows have been displayed, and the structural rows H, C, P, N preserve that same finite route. Since the carrier has no quotient-stream, selected-limit, host-modulus, or ambient-completeness coordinate, the displayed common refinement in J exhausts the consumer-visible comparison. □

Theorem 2386.5 (Dyadic Cauchy modulus lattice Real seal handoff)**.** *Every accepted* $\mathsf{DyadicCauchyModulusLattice}$ *packet exposes its* $\mathsf{Real}^{\uparrow}$ *seal only after the dyadic tolerance row, convergence-modulus row, directed lattice row, finite* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have all been displayed. Thus any Real or Completion consumer using the lattice must factor through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{CauchyConvergenceModulus}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as carrier rows of the same BHist *packet, with transport, replay, provenance, and local naming supplied by* H, C, P, N*.*

Proof. Start from the accepted carrier and read the obligation route of Definition 2386.2. The row R is not an independent Real datum; it is the seal handoff attached to the already displayed dyadic request, convergence threshold, lattice comparison, window, and regular readback.

Apply tail refinement when the consumer asks for a stronger tolerance. Theorem 2386.3 keeps the request on the same weak-later modulus, window, readback, and seal route, so the endpoint cannot fork into a selected limit or quotient stream.

The structural rows H, C, P, N then transport, replay, source, and name the same handoff. No host real equality, ambient Real completeness theorem, countable choice principle, or host-level modulus function is available from the carrier, so the displayed rows are the whole Real seal boundary. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ Real or Completion consumer factors through one displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ threshold row, one directed lattice row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before it may use a dyadic modulus comparison.

Independence witness: dyadicratcore, cauchy-convergence-modulus, streamname, regseqrat, real: $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately supply tolerance comparison, convergence thresholds, finite windows, regular rational readback, and terminal Real sealing, while this packet uniquely binds them to a directed lattice row consumed by Real and Completion arguments.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyModulusLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyModulusLattice}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a CauchyConvergenceModulus threshold row, a directed lattice row, a StreamName finite-window row, a RegSeqRat readback row, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyModulusLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2386.1, Definition 2386.2, Definition 2386.3, and Definition 2386.5 seed the directed dyadic Cauchy modulus lattice over DyadicRatCore, CauchyConvergenceModulus, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyModulusLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient stream equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed lattice comparison, dyadic tolerance exposure, convergence-modulus threshold exposure, finite-window readback, RegSeqRat handoff, Real seal handoff, structural transport, and non-escape for BEDC.Derived.DyadicCauchyModulusLatticeUp.

Chapter 2387

A Concrete Naming Certificate for $\mathsf{BishopLocatedLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedLimit}^{\uparrow}$ packet records the located-limit surface of a displayed constructive real sequence as finite NameCert data. It sits after the finite window schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located-limit interface of chapter 1420, the located-real enclosure route of **??**, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion route a traditional constructive limit certificate without quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2387.1 (Bishop located-limit carrier)**.** *A* $\mathsf{BishopLocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite windows inspected for the limit request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same schedule and tolerance data.*
4. *L is the located-limit witness row, recording the displayed modulus and enclosure data for the requested tail.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, D, R, L have been displayed.*
6. *H records componentwise* hsame *transport for the schedule, tolerance, readback, located-limit, and Real-seal rows.*
7. *C records displayed* Cont *replay from a precision request through the schedule, tolerance ledger, readback, located-limit witness, and Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedLimitUp.BishopLocatedLimitTasteGate single carrier alignment`

Theorem 2387.2 (Bishop located-limit NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedLimit}^{\uparrow}$ *carrier* $B = (S, D, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedLimit}^{\uparrow}}$ *surface consists exactly of StreamName schedule exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, located-limit witness display, Real-seal ordering, componentwise transport, continuation replay, package provenance, and local naming. Every Real or Completion consumer must read the finite route*

$$S, D \longmapsto R \longmapsto L \longmapsto E$$

together with H, C, P, N *before treating the packet as a Bishop located limit.*

Proof. Project the carrier of Definition 2387.1. The tuple exposes exactly the schedule, tolerance, readback, located-limit, Real-seal, transport, replay, provenance, and local naming rows.

The constructive source is finite. The rows S and D name the inspected windows and tolerances, and R reads those same windows as regular rational data rather than as host sequence equality.

The limit side is then read through L and E. The row L supplies the located witness for the displayed request, and the row E is downstream from that witness, so a Real-facing consumer cannot bypass the located route.

Transport through H and replay through C preserve the same ordering, while P and N record provenance and local naming. Since the carrier contains no quotient stream equality, selected limit, countable-choice schedule, trichotomy decision, or ambient-completeness row, the listed projections exhaust the NameCert obligations. □

Theorem 2387.3 (Bishop located-limit regular-Cauchy readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{BishopLocatedLimit}^{\uparrow}$ *carrier* $B = (S, D, R, L, E, H, C, P, N)$ *must read, in order, the* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{LocatedLimit}^{\uparrow}$ *witness* L*, the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure route carried by the same displayed limit data, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, followed by the structural rows* H, C, P, N*. In particular, the consumer cannot bypass the regular-Cauchy boundary* R *by appealing directly to a host located limit, quotient stream equality, selected real, countable choice, trichotomy, or ambient Real completeness.*

Proof. Project the carrier of Definition 2387.1. The schedule and tolerance rows S, D precede the readback row R, and the located-limit and seal rows L, E are downstream of that readback.

Apply Definition 2387.2. The theorem states that every Real or Completion consumer must read the finite route $S, D \longmapsto R \longmapsto L \longmapsto E$ together with the structural rows before treating the packet as a Bishop located limit.

The $\mathsf{LocatedReal}^{\uparrow}$ enclosure is therefore read only as part of the same located-limit witness and Real-seal path. Since no carrier coordinate stores a direct host limit, quotient stream equality, selected real, choice schedule, trichotomy decision, or ambient completeness theorem, the regular-Cauchy readback R is the mandatory boundary. □

Theorem 2387.4 (Bishop located-limit tail-modulus induction)**.** *Let* $B = (S, D, R, L, E, H, C, P, N)$ *be an accepted* $\mathsf{BishopLocatedLimit}^{\uparrow}$ *carrier. Any later located-limit precision request is discharged by induction over the same finite* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, located-limit witness* L*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, with* H, C, P, N *preserving the route.*

Thus a later request may narrow the displayed tolerance window, but it cannot replace the route by quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient Real completeness.

Proof. Project the carrier rows of Definition 2387.1. The later request has access only to the schedule S, tolerance ledger D, readback row R, located-limit witness L, Real seal E, and the structural rows.

Use the displayed schedule and tolerance ordering. A later precision request is a later finite window inside the same $\mathsf{StreamName}^{\uparrow}$ schedule and is budgeted by the same $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before the $\mathsf{RegSeqRat}^{\uparrow}$ readback can be inspected.

Induct over the finite request schedule. The initial request is the regular-Cauchy readback route of Definition 2387.3; the step narrows the tolerance row and reuses the same readback, located witness, and seal rows.

Finally repack through H, C, P, N. Since every step stays inside the displayed finite route, no quotient stream equality, selected host limit, countable choice, trichotomy, or ambient Real completeness is exported. □

Theorem 2387.5 (Bishop located-limit window determinacy)**.** *For every accepted* $\mathsf{BishopLocatedLimit}^{\uparrow}$ *carrier* $B = (S, D, R, L, E, H, C, P, N)$*, any two Real or Completion consumer reads that classify the same precision request must factor through the same finite route*

$$S, D \longmapsto R \longmapsto L \longmapsto E$$

and the same structural rows H, C, P, N*. No such read may switch to quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient Real completeness.*

Proof. Project the carrier of Definition 2387.1. The only rows available to the classifier are the displayed schedule, tolerance ledger, regular rational readback, located-limit witness, terminal Real seal, transport, replay, provenance, and local naming data.

Apply Definition 2387.2. Any Real or Completion consumer must read the route $S, D \longmapsto R \longmapsto L \longmapsto E$ together with H, C, P, N before treating the packet as a Bishop located limit.

Apply Definition 2387.3. The readback row R is the mandatory $\mathsf{RegSeqRat}^{\uparrow}$ boundary, so two reads of the same precision request cannot replace it by host located-limit equality or by a selected Real value.

It remains only to compare the available coordinates. Since the carrier has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, or ambient Real completeness, there is no second route through which either read can classify the request. Both reads are therefore determined by the same finite window route and by the same structural rows. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedLimit}^{\uparrow}$ read used by a Real or Completion consumer displays one $\mathsf{StreamName}^{\uparrow}$ schedule, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one located-limit witness, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: streamname, dyadicratcore, regseqrat, locatedlimit, locatedreal, real: $\mathsf{BishopLocatedLimit}^{\uparrow}$ is not bijective to any listed sibling because it binds the Bishop located-limit witness to the same finite schedule, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedLimit}^{\uparrow}$ packet is generated from a StreamName finite schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a located-limit witness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 2387.1 and Definition 2387.2 seed the finite Bishop located-limit route over StreamName, DyadicRatCore, RegSeqRat, LocatedLimit, LocatedReal, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient stream equality, selected host limits, quotient real equality, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exposure, tolerance accounting, regular readback ordering, located-limit witness display, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedLimitUp.

Chapter 2388

A Concrete Naming Certificate for $\mathsf{MetricEmbedding}^{\uparrow}$

Following Bourbaki and Kelley on metric embeddings, the $\mathsf{MetricEmbedding}^{\uparrow}$ packet records finite control data for an injective metric-route interface without requiring equality of all distances. It sits between the metric source of chapter 57, the graph and replay surface of chapter 130, the separation consumer of chapter 1411, and the stricter distance-equality interface of chapter 439. The packet is weaker than $\mathsf{IsometricEmbedding}^{\uparrow}$: it exposes displayed graph-distance control and separated-completion handoff rows, but it does not certify distance preservation, quotient completion classes, selected representatives, or host function equality.

Definition 2388.1 (MetricEmbedding carrier)**.** *A* $\mathsf{MetricEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, F, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
3. *F is the finite graph row displaying the source-to-target relation used by consumers;*
4. *D is the graph-distance control row, recording source distance reads, target distance reads, and the permitted control comparison between them;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and distance-read seal attached to the comparisons in D;*
6. *S is the separated-metric handoff row for consumers that need point separation after reading the graph;*
7. *H stores componentwise* hsame *transports for X, Y, F, D, R, S and for the structural rows below;*
8. *C stores the displayed* Cont *replay from source metric reads to graph reads, target metric reads, distance-control reads, and separated-consumer reads;*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}$ *row.*

The carrier contains no host function object, no quotient map, no selected completion representative, no ambient completion theorem, and no $\mathsf{IsometricEmbedding}^{\uparrow}$ *distance-equality certificate.*

Lean4 (definition) `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier`

Theorem 2388.2 (MetricEmbedding carrier and classifier obligations)**.** *For every accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *carrier* $E = (X, Y, F, D, R, S, H, C, P, N)$*, the local carrier and classifier surface consists exactly of the displayed ten-row packet. Carrier admission reads the source metric row, target metric row, graph row, graph-distance control row, real-distance seal, separated handoff, transport, replay, provenance, and local naming rows. Classifier transport compares accepted packets componentwise through the* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg*, and* $\mathsf{NameCert}$ *surfaces, and it has no coordinate for host function equality, quotient maps, selected representatives, or isometric distance equality.*

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.TasteGate.MetricEmbeddingTasteGate single carrier alignment`

Proof. Project the carrier definition. The accepted packet exposes exactly $X, Y, F, D, R, S, H, C, P, N$, so carrier admission is the finite BHist row inventory and not an external embedding object.

Classifier transport is componentwise on the same rows. The source and target metric classifiers compare X and Y, the continuous-map graph surface compares F, the distance-control and real-seal rows compare D and R, and the separated handoff compares S only after the graph and metric rows have been displayed.

The structural rows do not add analytic coordinates. They transport, replay, package, and name the same packet, so host functions, quotient maps, selected completion representatives, and the stricter $\mathsf{IsometricEmbedding}^{\uparrow}$ equality surface cannot be read from this carrier. □

Theorem 2388.3 (MetricEmbedding graph-distance control)**.** *Let* $E = (X, Y, F, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *packet. Every distance-control request first reads a source metric observation in* X*, then the displayed graph row* F*, then the target metric observation in* Y*, and only then the comparison row* D *with its* $\mathsf{Real}^{\uparrow}$ *seal* R*. The request receives the named graph-distance control data, not an isometric equality of source and target distances and not a detached continuous map.*

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier graph distance control`
↪ variant `BEDC.Derived.MetricEmbeddingUp.GraphDistanceControl.MetricEmbeddingGraphDistance control`
↪ variant `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingGraphDistanceControl`

Proof. Start with a distance-control request. Carrier projection forces the request to pass through the source metric row X and the graph row F before it can identify the corresponding target metric row Y.

Read the comparison row D only after those observations have been displayed. The seal R records the finite real-distance and tolerance data attached to this comparison, while H and C keep the source, graph, target, and real rows aligned.

The packet has no coordinate asserting equality of all source and target distances. Therefore the exported datum is the displayed graph-distance control route, weaker than the $\mathsf{IsometricEmbedding}^{\uparrow}$ distance-equality surface and independent of any host-level continuous map object. □

Theorem 2388.4 (MetricEmbedding separated-completion handoff)**.** *Every* $\mathsf{SeparatedMetric}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *packet factors through*

$$X \longmapsto F \longmapsto Y \longmapsto D \longmapsto R \longmapsto S$$

with the stored transport, replay, provenance, and local naming rows. The handoff may use graph-distance control to supply separated metric data, but it does not export quotient completion classes, selected representatives, ambient completion existence, or isometric distance preservation.

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier separated completion handoff`

Proof. Project the separated handoff row S. It is downstream from the displayed source metric row, graph row, target metric row, graph-distance control row, and real seal, so a separated consumer cannot read it before those rows have been exhibited.

Apply the graph-distance control theorem. The route to S uses the finite comparison data D, R attached to graph observations, not a host function and not an isometric equality certificate.

Finally read the structural rows H, C, P, N. They record transport, continuation replay, provenance, and local naming for the same displayed packet. Since quotient completion classes, selected representatives, ambient completion theorems, and isometric preservation are absent from the carrier, separated-completion consumers receive only the finite handoff route above. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricEmbedding}^{\uparrow}$ packet is generated from displayed source and target metric rows, a graph row, a graph-distance control row, a RealUp seal, a separated-metric handoff row, componentwise transport, displayed replay, package provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}, \mathsf{obligation})$

Scope closed: The obligation scope enumerates carrier admission, classifier transport, graph-distance control, separated-completion handoff, and nonescape obligations for metric embeddings weaker than isometric embeddings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient completion classes, selected representatives, ambient completion existence, isometric distance preservation, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checking the MetricSpace, SeparatedMetric, BHist, hsame, Cont, Pkg, and NameCert rows against the same finite packet.

Chapter 2389

A Concrete Naming Certificate for $\mathsf{AffineContraction}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, an $\mathsf{AffineContraction}^{\uparrow}$ packet names an affine self-map whose linear part carries a displayed contraction bound. It refines the map-level contraction surface of chapter 1418, reads its bounded-distance comparison through chapter 410, and hands Cauchy approximation data to chapter 1506 and chapter 1473. The packet records the affine decomposition and the contraction ledger without selecting a fixed point, assuming ambient completeness, or identifying host affine maps.

Definition 2389.1 (AffineContraction carrier)**.** *An* $\mathsf{AffineContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N)$$

with the following displayed rows:

1. X, d *are the metric carrier and distance-read rows used by the contraction and completion consumers.*
2. $V, +, 0$ *are the affine-coordinate rows naming the carried additive chart in which the affine decomposition is read.*
3. L *is the linear-part graph row, with displayed application reads inside the same carried chart.*
4. b *is the translation row.*
5. T *is the affine self-map row, read only through the finite graph equation* $T(x) = L(x) + b$.
6. G *is the graph-exactness row tying each accepted read of* T *to the displayed reads of* L, b, *and* $+$.
7. λ *is the rational contraction-bound ledger, with nonnegative and strict unit-bound rows for the linear part.*
8. M *is the induced Lipschitz and contraction-modulus row consumed by metric-completion and uniform-completion readers.*
9. H *stores componentwise* hsame *transport for the metric, affine-coordinate, linear-part, translation, self-map, graph, bound, modulus, replay, provenance, and local naming rows.*
10. C *stores displayed* Cont *replay for affine application, graph exactness, contraction-bound readback, modulus extraction, and completion-facing handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{AffineContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$ *row.*

The carrier exposes an affine contraction surface. It contains no coordinate for a selected fixed point, a completed endpoint, quotient equality of maps, ambient completeness, choice, or bridge checking.

Lean4 ✓ `BEDC.Derived.AffineContractionUp.AffineContractionTasteGate single carrier alignment`

Theorem 2389.2 (AffineContraction NameCert obligations)**.** *For an accepted* $\mathsf{AffineContraction}^{\uparrow}$ *carrier* $A = (X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$ *surface consists of carrier admission, affine-coordinate visibility, linear-part graph visibility, translation visibility, self-map graph exactness, rational contraction-bound stability, induced Lipschitz-modulus visibility, metric-completion and uniform-completion handoff visibility, componentwise classifier transport, continuation replay, provenance, and local naming for the finite packet of Definition 3340.1.*

Proof. Carrier admission is projection from the displayed tuple. The rows $X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N$ are exactly the rows available to the local certificate.

Affine-coordinate visibility reads the additive chart rows $V, +, 0$. Linear-part and translation visibility then read L and b before the self-map row T can be consumed.

Graph exactness is the row G. It forces each accepted self-map application to replay through $L(x)$, b, and the displayed addition row, so the affine map is not identified by host function equality.

The contraction-bound and modulus obligations are carried by λ and M. Since λ stores the strict unit-bound ledger for the linear part, the induced modulus is available to contraction and completion consumers without selecting a fixed point.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. These rows export no ambient completeness proof, choice principle, endpoint seal, quotient equality, or bridge-checked standardization. □

Theorem 2389.3 (AffineContraction metric-completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{AffineContraction}^{\uparrow}$ *carrier must pass through*

$$X,\ d,\ V,\ +,\ 0,\ L,\ b,\ T,\ G,\ \lambda,\ M,\ H,\ C,\ P,\ N.$$

The handoff supplies a displayed affine self-map, graph-exact affine decomposition, rational contraction ledger, and induced modulus. It does not supply a selected fixed point, completed endpoint, quotient equality of affine maps, ambient completeness theorem, choice principle, or bridge-checked standard object.

Proof. Read the carrier of Definition 3340.1. The only completion-facing rows are the metric rows X, d, the affine decomposition rows $V, +, 0, L, b, T, G$, the contraction ledger λ, and the induced modulus row M.

The graph row G orders the affine read before any completion consumer sees the self-map. The consumer therefore receives a finite route from x to $L(x) + b$, not a host affine function.

The strict unit-bound row in λ and the induced modulus row M are the only contraction data exported to chapter 1506 and chapter 1473. A completion route may consume those rows to build Cauchy approximation data, but the endpoint seal and completeness proof remain outside this packet.

Transport and replay are exhausted by H and C, while P and N only record provenance and local naming. Thus the handoff is precisely the finite affine-contraction surface and cannot provide the excluded endpoint, quotient, choice, completeness, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3340.1, Definition 3340.2, and Definition 3340.3 seed the affine contraction packet over affine decomposition, contraction ledger, and completion-facing modulus handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected fixed points, completed endpoints, quotient equality of affine maps, ambient completeness, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-coordinate visibility, linear-part graph visibility, translation visibility, self-map graph exactness, rational bound stability, modulus visibility, completion handoff visibility, transport, replay, provenance, and local naming for BEDC.Derived.AffineContractionUp.

Constructive story: An $\mathsf{AffineContraction}^{\uparrow}$ packet is generated from a metric carrier, affine-coordinate rows, linear-part graph, translation row, affine self-map graph, rational contraction-bound ledger, induced modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 2390

A Concrete Naming Certificate for $\mathsf{MetricClosure}^{\uparrow}$

Following Kelley on metric topology and Bishop–Bridges constructive metric spaces, the $\mathsf{MetricClosure}^{\uparrow}$ packet names the finite closure interface by which metric tests, neighbourhood tests, located-set rows, finite StreamName windows, RegSeqRat readbacks, and Real seals expose a candidate limit as closure evidence. It sits over the metric source of chapter 57, the topology surface of chapter 88, the located-set interface of chapter 1541, the sequence-limit handoff of chapter 1571, the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a named BEDC surface for metric closure reads; it is not a host topological closure operator, not an arbitrary subsequence selector, not open-cover compactness, not quotient equality, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2390.1 (MetricClosure carrier)**.** *A* $\mathsf{MetricClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, S, L, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the finite metric comparison surface.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row naming the neighbourhood tests compatible with M.*
3. *S is the subset or* $\mathsf{LocatedSet}^{\uparrow}$ *row whose admitted points are being closed.*
4. *L is the candidate limit row, read only through the same metric and topological tests.*
5. *U is the finite neighbourhood and metric-test ledger, pairing each displayed neighbourhood request with the matching metric tolerance request.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the displayed approximating points and the candidate limit.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the same windows, tolerances, and metric-comparison requests.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, T, S, L, U, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the metric, topology, subset or located-set, candidate limit, test ledger, window, readback, and Real seal rows.*

10. *C records displayed* Cont *replay from neighbourhood requests to metric tests, then through StreamName windows, RegSeqRat readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose metric row, topology row, subset or located-set row, candidate limit, paired test ledger, finite windows, rational readbacks, Real seal, transport, replay, provenance, and local naming are all displayed. It has no coordinate for a host closure operator, arbitrary subsequence choice, open-cover compactness, quotient equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 2390.2 (MetricClosure NameCert obligations)**.** *For every accepted* $\mathsf{MetricClosure}^{\uparrow}$ *carrier* $K = (M, T, S, L, U, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient metric row, the compatible topology row, the subset or located-set row, the candidate limit row, the paired neighbourhood and metric-test ledger, StreamName windows, RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public closure read must pass through those rows before it can consume the candidate limit.*

Proof. Project the accepted carrier to the displayed rows $M, T, S, L, U, W, R, E, H, C, P, N$. The metric, topology, subset, and candidate rows give the only admissible source data for a closure read.

The ledger row U then binds each neighbourhood request to a finite metric-test request. The StreamName row W and RegSeqRat row R provide the visible windows and rational readbacks for that same request before the Real seal E is exposed.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Since no row stores a host closure operator, an arbitrary subsequence, an open-cover object, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the NameCert obligations are exhausted by the listed rows. □

Lean4 ✓ `BEDC.Derived.MetricClosureUp.MetricClosureCarrier namecert obligations`

Theorem 2390.3 (MetricClosure sequential-closure handoff)**.** *Every* $\mathsf{SequentialClosure}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricClosure}^{\uparrow}$ *packet reads the candidate limit through the finite route*

$$M, T, S, L, U, W, R, E, H, C, P, N.$$

The consumer receives a displayed metric row, a compatible topology row, a subset or located-set row, a candidate limit, paired neighbourhood and metric-test rows, finite StreamName windows, RegSeqRat readbacks, a RealUp seal, transport, replay, provenance, and local naming. It receives no ambient topological closure operator, arbitrary subsequence selector, open-cover compactness principle, quotient-equality row, or RealUp completeness theorem.

Proof. Start with a sequential-closure consumer. The carrier exposes the candidate limit L only together with the common metric row M, topology row T, and source row S, so the consumer cannot detach the limit from the displayed metric/topology interface.

Use the test ledger U to align the neighbourhood read with the metric tolerance read. The finite window row W, regular rational row R, and Real seal E then give the same handoff surface used by the $\mathsf{SequentialClosure}^{\uparrow}$ packet of chapter 1571.

The rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Thus the handoff is the finite metric-closure surface itself, not a stronger host closure, subsequence-choice, compactness, quotient, or completeness principle. □

Falsifiable prediction: Every $\mathsf{MetricClosure}^{\uparrow}$ public read factors through a Metric row, a Topology row, a subset or LocatedSet row, a candidate-limit row, paired neighbourhood and metric-test ledgers, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: metric, topology, locatedset, sequentialclosure, streamname, regseqrat, real: $\mathsf{MetricClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric/topology compatibility surface, a source subset or located-set row, a candidate limit, paired neighbourhood and metric tests, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricClosure}^{\uparrow}$ packet is generated from a Metric row, a compatible Topology row, a subset or LocatedSet row, a candidate limit, paired neighbourhood and metric-test rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3342.1, Definition 3342.2, and Definition 3342.3 seed the finite metric/topology closure interface for displayed candidate-limit reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topological closure, arbitrary subsequence choice, open-cover compactness, quotient equality, $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric/topology compatibility, located-source exposure, paired test-ledger stability, StreamName and RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricClosureUp.

Chapter 2391

A Concrete Naming Certificate for $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ packet records the finite equivalence between the Cauchy criterion read and an explicit convergence-modulus schedule before any completion or real-name consumer sees a terminal seal. It sits between the criterion surface of chapter 392, the convergence-modulus schedule of chapter 1438, the finite windows of chapter 126, the regular readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite-tail equivalence object for L10 Real and Completion consumers; it is not a quotient-stream equality, selected limit, countable-choice schedule, host convergence theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2391.1 (CauchyCriterionEquivalence carrier)**.** *A* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence and source-window row, exposing only the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows of the sequence under discussion.*
2. *Q is the criterion tail request row, naming the finite late-pair request consumed by the* $\mathsf{CauchyCriterion}^{\uparrow}$ *surface.*
3. *M is the convergence-modulus schedule row, naming the finite threshold data consumed by the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *surface.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic radius, split tolerances, slack cells, and comparison budget shared by Q and M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed source windows and tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source windows, criterion request, modulus schedule, dyadic tolerance ledger, regular readback, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from source windows to criterion requests, from criterion requests to modulus schedules, from both finite surfaces to dyadic tolerance reads, and from the tolerance-readback route to the real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, M, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice selection of thresholds, host real equality, detached convergence proofs, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2391.2 (CauchyCriterionEquivalence NameCert obligations)**.** *For every accepted* CauchyCriterionEquival *carrier*

$$K = (S, Q, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$ *surface consists exactly of the bidirectional finite criterion/modulus obligations over the displayed packet. A criterion-facing read of Q may expose a convergence-modulus schedule only by reading the same source windows S, dyadic tolerance ledger D, regular readback R, real seal E, and structural rows H, C, P, N. Conversely, a modulus-facing read of M may expose the corresponding Cauchy-criterion tail request only through those same displayed rows. Neither direction exports quotient stream equality, a selected limit, countable choice over thresholds, host convergence, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or an ambient completion principle.*

Proof. Project the carrier of Definition 3343.1. The criterion request Q and the convergence-modulus schedule M are separate displayed rows, but they share the source-window row S, the dyadic tolerance row D, the regular readback R, the real seal E, and the structural rows.

For the criterion-to-modulus direction, read a finite late-pair request through Q. The continuation row C permits replay only along the displayed route from S through D and R before the schedule row M is exposed, so the modulus threshold is a packet-local finite row rather than an external selector.

For the modulus-to-criterion direction, read a displayed threshold in M. The same tolerance ledger D and readback R supply the late-pair comparison consumed by Q, and the transports in H preserve all rows componentwise under the local classifier.

Finally, the real seal E is downstream of both finite surfaces, while P and N only source and name the same packet. Since the carrier contains no quotient stream, selected limit, countable-choice schedule, host convergence theorem, detached real-completeness row, or ambient completion coordinate, the two finite directions exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionEquivalenceUp.CauchyCriterionEquivalenceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ packet is generated from source sequence windows, a criterion tail request, a convergence-modulus schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 3343.1 and Definition 3343.2 seed the finite bidirectional criterion/modulus surface over $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable-choice schedules, host convergence, detached $\mathsf{Real}^{\uparrow}$ completeness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus projection, modulus-to-criterion projection, shared tolerance-readback exactness, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyCriterionEquivalenceUp.

Chapter 2392

A Concrete Naming Certificate for $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$

Following Barendregt and Coquand-style metatheory, the $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ packet records the finite BEDC witness surface by which MetaCIC closedness preservation is consumed before any subject-reduction theorem is claimed. It sits after the audit rows of chapter 786, the closure-trace packet of chapter 795, and the dependent-codomain transport packet of chapter 804. The packet gives downstream closed-normal and subject-reduction routes a single human-lineage certificate for closedness preservation without treating closedness as typing preservation.

Definition 2392.1 (MetaCIC closure-preservation witness carrier)**.** *A* $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (A, T, D, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ *row that names the shift, substitution, beta-step, and finite beta-chain closedness-preservation audit chain;*
2. *T is the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *row that records the accepted closure-preservation theorem rows and their finite replay surface;*
3. *D is the* $\mathsf{DependentCodomainClosurePreservation}^{\uparrow}$ *row that exposes codomain closedness transport without discharging dependent-codomain inversion;*
4. *L is the ledger row stating that the witness is a closedness-preservation route, not a typing-preservation or subject-reduction route;*
5. *H records componentwise* hsame *transport for A, T, D, L, P, N;*
6. *C records displayed* Cont *replay routes from the witness to the audit, trace, and dependent-codomain rows;*
7. *P is the* Pkg *provenance over MetaCIC closure preservation, the three sibling packets,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}$ *row.*

The carrier contains no typing context, no conversion equality, no subject-reduction inhabitant, no normalization theorem, no confluence theorem, and no route from closedness preservation to closed consistency.

Lean4 ✓ BEDC.Derived.MetaCICClosurePreservationWitnessUp.MetaCICClosurePreservationWitnessTasteGate
single carrier alignment

Theorem 2392.2 (MetaCIC closure-preservation witness audit chain)**.** *For the carrier of Definition 3344.1, every accepted witness read factors through the chain*

$$\mathsf{ClosurePreservationAuditWitness}^{\uparrow} \longrightarrow \mathsf{MetaCICClosureTrace}^{\uparrow} \longrightarrow \mathsf{DependentCodomainClosurePreservation}^{\uparrow}.$$

The first row supplies the closedness-preservation audit inventory, the second row supplies the finite accepted-declaration trace, and the third row supplies the dependent-codomain closedness transport boundary. The chain exports only closedness-preservation data and its displayed ledger.

Proof. Project the carrier W. The first three rows are A, T, D, so any consumer reaches the audit witness, closure trace, and dependent-codomain transport surface before it can read the ledger.

The audit row A fixes the shift, substitution, beta-step, and beta-star closedness rows. The trace row T records those rows as accepted MetaCIC closure-preservation declarations, while the dependent-codomain row D records the codomain closedness transport boundary.

Finally inspect L, H, C, P, N. The ledger marks the route as closedness-only, and the transport, replay, provenance, and local naming rows preserve only the displayed chain. No coordinate adds typing preservation or subject reduction, so the audit chain has exactly the stated scope. □

Theorem 2392.3 (MetaCIC closure-preservation witness non-escape)**.** *Every public read of an accepted* $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ *carrier remains inside*

$$A, T, D, L, H, C, P, N.$$

A consumer may cite the $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ *audit row, the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *accepted trace row, the* $\mathsf{DependentCodomainClosurePreservation}^{\uparrow}$ *codomain-transport row, the closedness-only ledger, componentwise transports, displayed replay, provenance, and local naming. It cannot extract subject reduction, type preservation, conversion, confluence, normalization, closed consistency, or a hidden MetaCIC kernel soundness premise.*

Proof. Carrier projection leaves exactly the finite tuple A, T, D, L, H, C, P, N. The three witness rows are already named as closedness-preservation surfaces, and the ledger row records that no typing-preservation route is being exported.

Replay through C and transport through H keep the same rows visible to downstream consumers. Provenance P identifies the sibling packets and the MetaCIC closure-preservation route, while N names the local certificate.

For any stronger request, inspect the carrier inventory. There is no coordinate for typing contexts, conversion equality, confluence, normalization, closed consistency, or kernel soundness. Thus every public read remains confined to the displayed closedness-preservation witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ packet is generated from the closure-preservation audit witness, MetaCIC closure trace, dependent-codomain closure-preservation transport row, closedness-only ledger, componentwise transports, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3344.1, Definition 3344.2, and Definition 3344.3 bind the finite witness surface joining audit rows, accepted trace rows, and dependent-codomain closedness transport.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove subject reduction, typing preservation, dependent-codomain inversion, confluence, normalization, closed consistency, Lean verification of the witness packet, or a public schema bridge.

Upgrade path: Obligation closure requires checked carrier admission, audit-chain exactness, dependent-codomain boundary preservation, closedness-only ledger exactness, provenance, local naming, and non-escape for the same finite witness packet.

Chapter 2393

A Concrete Naming Certificate for $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ packet names the finite tail-comparison route for positive decreasing $\mathsf{RealSeries}^{\uparrow}$ terms. It sits next to the real-series surface of chapter 1581, the displayed limit interface of chapter 1413, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, and the terminal real-name seal of chapter 38. It refines the dyadic-block test packet of chapter 1478 by making the $\mathsf{RealSeries}^{\uparrow}$ tail carrier itself visible to Real and Completion consumers before any infinite sum or completed endpoint is read.

Definition 2393.1 (CauchyCondensationRealSeries carrier)**.** *A* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, M, B, G, W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive decreasing* $\mathsf{RealSeries}^{\uparrow}$ *source row. Its terms are admitted only through finite term windows, and its decreasing evidence is read as adjacent finite comparisons.*
2. *M is the monotone-tail ledger, recording positivity, decreasing transport, and the finite tail cut from which the comparison begins.*
3. *B is the dyadic block-window ledger. It lists the inspected powers of two, the associated block endpoints, and the finite intervals of original terms visible at each requested tolerance.*
4. *G is the condensed-term row, displaying the term $2^n a_{2^n}$ on the same dyadic indices as B and tying it to the original* $\mathsf{RealSeries}^{\uparrow}$ *term windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule used for the original terms, block endpoints, condensed terms, and partial-sum windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every displayed original term, condensed term, and block partial sum.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and block-comparison ledger. It records the finite inequalities that bound each inspected original tail block above and below by the matching condensed-term rows.*

8. *R is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *route for the partial-sum and condensed-partial-sum windows that the packet exposes to downstream limit consumers.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, M, B, G, W, Q, D, R have been displayed.*
10. *H records componentwise* hsame *transport for the source, monotone-tail ledger, dyadic blocks, condensed terms, schedules, readbacks, tolerance ledger, limit route, Real seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a tail request through the monotone ledger, dyadic blocks, condensed terms, stream schedules, regular readbacks, dyadic comparisons, limit route, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, B, G, W, Q, D, R, E, H, C, P, N$. It has no coordinate for host infinite sums, quotient real equality, countable choice, trichotomy, selected endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationRealSeriesUp.CauchyCondensationRealSeriesTasteGate`
`single carrier alignment`

Theorem 2393.2 (CauchyCondensationRealSeries obligations)**.** *For every accepted* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *carrier $K = (A, M, B, G, W, Q, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed thirteen-row packet; positive decreasing* $\mathsf{RealSeries}^{\uparrow}$ *source exposure through A and M; dyadic block-window coverage through B; condensed-term formation through G;* $\mathsf{StreamName}^{\uparrow}$ *scheduling through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q;* $\mathsf{DyadicRatCore}^{\uparrow}$ *block comparison through D;* $\mathsf{RealSequenceLimit}^{\uparrow}$ *partial-sum routing through R;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3345.1. The accepted packet exposes exactly the source, monotone-tail ledger, dyadic block windows, condensed terms, stream schedules, regular readbacks, dyadic comparison ledger, limit route, Real seal, and structural rows.

The source obligations are finite. Positivity and decreasing behavior are read from M only on the term windows selected by W, and the original term observations are read back through Q before any block comparison is formed.

The condensation obligations are carried by B, G, D. The dyadic ledger B supplies the finite block endpoints, G forms the corresponding condensed terms, and D records the block inequalities that compare the original tail with the condensed tail on those displayed windows.

The consumer route is then R followed by E. The limit-facing row R receives only the partial-sum and condensed-partial-sum windows already displayed, and the Real seal E is reached only after that route has passed through the regular and dyadic ledgers.

Finally H, C, P, N transport, replay, source, and name the same packet. Since host infinite sums, quotient real equality, countable choice, trichotomy, selected endpoints, and ambient completeness are not coordinates of K, they cannot appear as local NameCert obligations. □

Theorem 2393.3 (CauchyCondensationRealSeries Cauchy route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *packet factors its tail-comparison read through*

the finite route

$$A, M \longmapsto B, G \longmapsto W, Q, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the original positive decreasing $\mathsf{RealSeries}^{\uparrow}$ *tail and the condensed tail are compared only by displayed dyadic block windows, scheduled stream reads, regular rational readbacks, and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *partial-sum route before any* $\mathsf{Real}^{\uparrow}$ *seal is visible.*

Proof. Start with the accepted carrier. The source row A and monotone ledger M are the only rows from which a consumer can inspect the positive decreasing real-series tail.

Read the dyadic block rows next. The ledger B names the finite block windows, while G names the condensed terms on the same dyadic indices. No consumer can read a condensed comparison without preserving both rows.

Now pass through W, Q, D. The schedule W fixes the finite term and partial-sum windows, Q gives the regular rational readback for those windows, and D records the dyadic inequalities that compare each displayed original block with its condensed row.

The partial-sum route R is therefore already a displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ route before the seal E is reached. The remaining rows H, C, P, N preserve and name that same finite route, so the handoff supplies no host infinite sum, quotient real equality, countable-choice schedule, trichotomy split, selected endpoint, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3345.1, Definition 3345.2, and Definition 3345.3 seed the finite tail-comparison carrier for positive decreasing $\mathsf{RealSeries}^{\uparrow}$ terms, dyadic block windows, condensed terms, finite schedules, regular readbacks, dyadic comparison ledgers, limit routing, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, monotone-tail exposure, dyadic block-window coverage, condensed-term formation, schedule and readback stability, dyadic comparison, limit routing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationRealSeriesUp.

Constructive story: A $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ packet is generated from a positive decreasing RealSeries source row, a monotone-tail ledger, dyadic block windows, condensed terms, StreamName schedules, RegSeqRat readbacks, DyadicRatCore comparison rows, a RealSequenceLimit route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2394

A Concrete Naming Certificate for VariationOfConstants$^{\uparrow}$

Following Coddington–Levinson and Hartman on ordinary differential equations, the VariationOfConstants$^{\uparrow}$ packet records the finite inhomogeneous linear-ODE handoff by which a homogeneous local flow, a forcing row, and a Gronwall control row determine the displayed solution update without importing a host solution operator.

The packet sits next to the ODE source surface of chapter 193, the Picard–Lindelof existence route of chapter 1888, the local-flow endpoint packet of chapter 1899, the Gronwall comparison surface of chapter 2034, and the real endpoint seal of chapter 38. It names the standard variation-of-constants route as a BEDC certificate over displayed rows rather than as an ambient formula in a function space.

Definition 2394.1 (Variation-of-constants carrier)**.** *A* VariationOfConstants$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (O, P, L, G, R, H, C, Q, N)$$

with the following visible rows:

1. *O is the* Ode$^{\uparrow}$ *inhomogeneous linear source row, displaying the time window, initial state row, coefficient row, forcing row, and endpoint request.*
2. *P is the* PicardLindelof$^{\uparrow}$ *existence-and-uniqueness route read on the same local window.*
3. *L is the* PicardLindelofLocalFlow$^{\uparrow}$ *homogeneous local-flow row, naming the propagator surface used before the forcing contribution is accumulated.*
4. *G is the* GronwallInequality$^{\uparrow}$ *comparison row that bounds the accumulated forcing and the difference between any two displayed variation-of-constants reads.*
5. *R is the* Real$^{\uparrow}$ *endpoint and error-seal row for the finite observation of the homogeneous part, forcing integral ledger, and output state.*
6. *H stores componentwise* hsame *transport for the ODE source, Picard route, local-flow row, Gronwall row, real endpoint seal, replay row, provenance row, and local name.*
7. *C stores the displayed* Cont *replay from the ODE source through the Picard route, homogeneous flow row, forcing ledger, Gronwall comparison, and real endpoint seal.*
8. *Q is the* Pkg *provenance over* VariationOfConstants$^{\uparrow}$, Ode$^{\uparrow}$, PicardLindelof$^{\uparrow}$, PicardLindelofLocalFlow$^{\uparrow}$, GronwallInequality$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by the displayed source, Picard route, homogeneous-flow row, forcing ledger as replayed by C, Gronwall comparison, real endpoint seal, transport, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.VariationOfConstantsUp.VariationOfConstantsTasteGate single carrier alignment`

Theorem 2394.2 (Variation-of-constants NameCert obligations)**.** *For the carrier of Definition 3346.1, the local* $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$ *surface consists exactly of carrier admission for $V = (O, P, L, G, R, H, C, Q, N)$ ODE source admission through O, Picard–Lindelof route visibility through P, homogeneous local-flow visibility through L, Gronwall comparison visibility through G, endpoint sealing through R, componentwise transport through H, displayed replay through C, provenance through Q, and local naming through N. The packet exports no host fundamental matrix, selected global solution curve, quotient path equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier. The fields $O, P, L, G, R, H, C, Q, N$ are the only available rows, so every local naming obligation must be read from those fields.

The source and existence rows come first. The ODE row O supplies the inhomogeneous linear problem, and P records the Picard–Lindelof route on the same displayed window. The homogeneous row L is therefore not a detached fundamental matrix; it is the local-flow packet already named by chapter 1899.

The forcing contribution is read only by replay. The continuation row C accumulates the forcing ledger after the homogeneous flow row has been displayed, and the comparison row G bounds the accumulated read through the Gronwall surface of chapter 2034.

Finally R, H, Q, N seal, transport, source, and name the same finite route. Since no carrier coordinate stores a host solution operator, selected global curve, quotient path equality, choice schedule, or ambient completeness theorem, none of those objects can become a NameCert obligation. □

Theorem 2394.3 (Variation-of-constants Picard–Gronwall handoff)**.** *Every accepted* $\mathsf{VariationOfConstants}^{\uparrow}$ *carrier factors each inhomogeneous local-ODE read through the finite route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{PicardLindelof}^{\uparrow} \longmapsto \mathsf{PicardLindelofLocalFlow}^{\uparrow} \longmapsto \mathsf{GronwallInequality}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Picard–Lindelof rows supply the displayed local-flow surface, the variation-of-constants replay reads the forcing ledger on that same surface, and the Gronwall row bounds the endpoint and comparison error before the $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Begin with O and P. They fix the ODE source row and the local Picard–Lindelof existence route on one displayed time window.

Read the homogeneous local flow next. The row L is consumed on the same window and supplies the propagator surface used by the forcing replay; no separate host fundamental matrix is opened.

The forcing step is the continuation route C. It records how the forcing ledger is accumulated after L, and it sends the resulting comparison request to the Gronwall row G.

The endpoint is sealed only after that comparison. The row R gives the visible $\mathsf{Real}^{\uparrow}$ endpoint and error seal, while H, Q, N keep transport, provenance, and local naming tied to the same packet. Hence an inhomogeneous local-ODE consumer sees exactly the Picard–Gronwall route above and cannot bypass it through a selected global solution curve or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3346.1, Definition 3346.2, and Definition 3346.3 seed the finite inhomogeneous ODE route over ODE source, Picard–Lindelof, local-flow, Gronwall, Real endpoint, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fundamental matrices, global ODE existence, quotient path equality, selected solution curves, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard–Lindelof route visibility, local-flow visibility, forcing replay exactness, Gronwall comparison visibility, endpoint sealing, transport, replay, provenance, and local naming for BEDC.Derived.VariationOfConstantsUp.

Constructive story: A $\mathsf{VariationOfConstants}^{\uparrow}$ packet is generated from one inhomogeneous Ode source row, one PicardLindelof route, one PicardLindelofLocalFlow homogeneous-flow row, one GronwallInequality comparison row, one Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2395

A Concrete Naming Certificate for CompletionUniformFunctorialEnvelope$^{\uparrow}$

Following Bishop–Bridges constructive completion and Mac Lane functorial language, the CompletionUniformFuncto packet records the finite envelope that lets a completion-facing consumer read the uniform-completion route of chapter 1473, the metric-completion carrier of chapter 1506, the functorial completed-map handoff of chapter 1937, and the extension-uniqueness surface of chapter 397 through one local NameCert boundary. Its real-facing side remains attached to the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38; it does not add a quotient completion, selected representative, countable-choice schedule, or ambient real-completeness coordinate.

Definition 2395.1 (CompletionUniformFunctorialEnvelope carrier)**.** *A* CompletionUniformFunctorialEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, M, F, L, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformCompletion$^{\uparrow}$ *row, exposing the entourage-facing completion presentation and the displayed Cauchy data that it consumes.*
2. *M is the* MetricCompletion$^{\uparrow}$ *row, exposing the metric completion carrier, its distance-readback boundary, and the finite tolerance handoff aligned with U.*
3. *F is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row, recording the completed-map transport and functorial composition ledger over the same source and target completion rows.*
4. *L is the* CompletionUniversalProperty$^{\uparrow}$ *row, recording extension existence and uniqueness over the completion surface presented by U, M, F.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row for the source, target, and completed-map observations inspected by the packet.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed stream windows, metric tolerances, and real-facing observations.*
7. *A is the terminal* Real$^{\uparrow}$ *seal row reached only after S, R, U, M, F, L have been displayed.*
8. *H records componentwise* hsame *transport for the uniform-completion, metric-completion, functoriality, universal-property, stream-window, regular-readback, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through the uniform and metric rows, through the functorial completed-map row, through the universal-property row, and finally through the stream, regular-readback, and Real-seal boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, F, L, S, R, A, H, C, P, N$. It has no coordinate for quotient completions, selected representatives, countable choice, host completed spaces, function extensionality outside the displayed hsame *rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionUniformFunctorialEnvelopeUp.TasteGate.CompletionUniformFunctorialEnvelop`
`single carrier alignment`

Theorem 2395.2 (CompletionUniformFunctorialEnvelope NameCert obligations)**.** *For every accepted* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ *carrier $E = (U, M, F, L, S, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; uniform-completion exposure through U; metric-completion exposure through M; completed-map functoriality through F; extension and uniqueness exposure through L; finite stream-window exposure through S; regular rational readback through R; terminal real-name sealing through A; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3347.1. The accepted packet exposes exactly the uniform-completion row, metric-completion row, functoriality row, universal-property row, stream windows, regular readbacks, Real seal, and structural rows.

The completion obligations are read in the order U, M, F, L. The uniform row supplies the entourage-facing completion presentation, the metric row supplies the distance and tolerance boundary, the functoriality row supplies completed-map transport, and the universal-property row supplies extension and uniqueness over those same completion coordinates.

The real-facing obligations are downstream of the completion envelope. The stream row S and regular readback R expose only finite observations, and the Real seal A is available only after the completion and functorial rows have already been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since quotient completions, selected representatives, countable choice, host completed spaces, external function extensionality, and ambient real completeness are not coordinates of E, they cannot appear as local NameCert obligations. □

Theorem 2395.3 (CompletionUniformFunctorialEnvelope consumer handoff)**.** *Every completion-facing or real-facing consumer of an accepted* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ *packet factors its read through the finite route*

$$U, M \longmapsto F \longmapsto L \longmapsto S, R, A \longmapsto H, C, P, N.$$

Thus a consumer can use uniform completion, metric completion, functorial completed maps, and extension uniqueness only as rows of one displayed envelope, and it reaches $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data only through the same finite packet.*

Proof. Start with an accepted carrier. The completion-facing side exposes U and M first, so the uniform and metric presentations are read from the same displayed packet rather than from independent host completions.

The functoriality row F then transports completed-map reads over those completion coordinates. The universal-property row L can expose extension and uniqueness only after that functorial route has been named.

The handoff to real-facing consumers is S, R, A. These rows provide the finite stream observations, regular rational readback, and terminal Real seal visible to downstream clients, with no selected representative or quotient-completion coordinate inserted between them.

The structural rows H, C, P, N preserve this route under transport, replay it as a displayed continuation, record provenance, and name the local certificate. Consequently every exported read factors through the displayed envelope and supplies no countable-choice schedule, host completed space, external function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ packet is generated from UniformCompletion, MetricCompletion, CauchyCompletionFunctoriality, CompletionUniversalProperty, StreamName, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3347.1, Definition 3347.2, and Definition 3347.3 seed the finite envelope binding uniform completion, metric completion, functorial completed maps, universal extension, finite stream windows, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host completed spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion exposure, functorial completed-map transport, universal-property exposure, stream-window and regular-readback stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionUniformFunctorialEnvelopeUp.

Chapter 2396

A Concrete Naming Certificate for $\mathsf{DarbouxProperty}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxProperty}^{\uparrow}$ packet records the finite intermediate-value route for a displayed real-valued map whose interval image cannot jump over a visible value row. It sits after the real-name boundary of chapter 38, the pointwise graph and modulus rows of chapter 130, the bisection handoff of chapter 1422, the regular rational readback of chapter 301, and the dyadic tolerance core of chapter 291. The chapter names the Darboux property as finite BEDC carrier data, not as a host theorem about all real functions or a choice-selected intermediate point.

Definition 2396.1 (DarbouxProperty carrier)**.** *A* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source interval row, exposing the inspected endpoint cells and orientation data used by the property request.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the graph window, pointwise modulus, and metric-distance witnesses for the displayed map.*
3. *A and B are the endpoint value rows, read as* $\mathsf{Real}^{\uparrow}$ *names attached to the two endpoint cells of I.*
4. *Y is the target value row whose placement between the displayed endpoint values is tested only through finite real-name windows.*
5. *S is the sign or bracket row for the residual comparison $F(-) - Y$, expressed through dyadic and regular rational windows rather than host real subtraction.*
6. *M is the finite modulus budget row selecting the precision at which endpoint, target, and midpoint reads are inspected.*
7. *R is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route that retains a branch whose residual signs still bracket the target value row.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained branch windows and requested tolerances.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal offered to a downstream intermediate-value consumer after the displayed route has been read.*

10. *H stores componentwise* hsame *transport for interval, map, endpoint value, target value, residual bracket, modulus, bisection, regular readback, seal, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a Darboux request through the same finite rows.*
12. *P is the* Pkg *provenance over* $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$ *row.*

The carrier exposes a finite bracketing certificate for target-value passage. It has no coordinate for host real equality, quotient real equality, a selected intermediate point, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.DarbouxPropertyUp.TasteGate.DarbouxPropertyTasteGate single carrier alignment`

Theorem 2396.2 (DarbouxProperty NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$ *surface consists of carrier admission, endpoint and target value visibility, residual bracketing visibility, finite modulus control, intermediate-value bisection handoff, regular rational readback, terminal real seal exposure, componentwise transport, continuation replay, provenance, and local naming for the displayed packet of Definition 3348.1.*

Proof. Project the carrier of Definition 3348.1. The interval row I, continuous-map row F, endpoint rows A, B, and target row Y are the only rows from which a Darboux request may be read.

The residual bracket is the row S. It is inspected at the precision selected by M, so the endpoint and target comparison remains inside finite dyadic and regular rational windows.

The bisection row R is then the only route to a retained branch. Its output is read back through Q before the real-name seal E is exposed.

The structural rows H, C, P, N transport, replay, source, and name that same packet. Since no coordinate stores host equality, quotient equality, a selected point, trichotomy, ambient completeness, or open-cover compactness, none of those objects can appear in the local NameCert surface. □

Theorem 2396.3 (DarbouxProperty intermediate-value route)**.** *Every target-value consumer of an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier factors through*

$\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, Nam

In particular, a request for a displayed value between the endpoint reads first reaches the continuous-map and residual-bracket rows, then the finite bisection route, then the regular rational readback, and only then the real-name seal.

Proof. Start from $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$. The continuation row C exposes the interval, map, endpoint, and target rows before any branch row is visible.

The residual bracket S is measured at the budget M, with dyadic tolerance data supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ component named in provenance. This gives a finite sign or comparison surface for $F(-) - Y$.

The branch handoff is exactly R. It is the $\mathsf{IntermediateValueBisection}^{\uparrow}$ route, so the retained branch is visible only through the same finite bisection ledger used by the constructive intermediate-value surface.

The retained branch is then read through Q and sealed by E. The remaining rows H, C, P, N keep the route attached to one BEDC packet, so the consumer cannot bypass the route by appealing to a host intermediate-value theorem or a selected point. □

Theorem 2396.4 (DarbouxProperty real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier is downstream of the endpoint value rows, the target value row, the residual bracket row, the modulus budget, the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. The export is a named finite real-seal handoff, not host real equality, quotient real equality, a selected intermediate point, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Proof. Read the route to the seal. The real-facing row E appears after the endpoint and target rows A, B, Y, the residual bracket S, the budget M, the bisection route R, and the regular readback Q in the displayed carrier.

The transport and replay rows H and C preserve those dependencies componentwise. Provenance P and local naming N record that the exposed seal belongs to the $\mathsf{DarbouxProperty}^{\uparrow}$ packet.

Therefore every real-facing read is exhausted by the finite rows already listed. There is no carrier field through which host equality, quotient equality, a selected intermediate point, trichotomy, ambient completeness, or open-cover compactness could be exported. □

Theorem 2396.5 (DarbouxProperty bisection target carrier)**.** *Let* $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier. Every target-value bisection read of* D *is carried by the finite row family*

$$I, F, A, B, Y, S, M, R.$$

The row R *may be inspected only after the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* F*, the endpoint* $\mathsf{Real}^{\uparrow}$ *rows* A, B*, the target* $\mathsf{Real}^{\uparrow}$ *row* Y*, the residual-bracket row* S*, and the finite modulus budget* M *have been displayed. Thus the bisection target is a BEDC packet over displayed residual windows, not a host-selected intermediate point.*

Proof. Project the carrier of Definition 3348.1. The only rows preceding the bisection route are the interval row, continuous-map row, endpoint and target value rows, residual bracket, and modulus budget.

The continuation row C preserves this read order. A target-value request therefore reaches R only after the finite residual comparison has been formed from F, A, B, Y, S, M.

The structural rows H, P, N transport, source, and name the same route. Since no coordinate stores a selected point or a host intermediate-value theorem, the bisection target remains the displayed carrier route. □

Theorem 2396.6 (DarbouxProperty residual-bracket stability)**.** *For an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier* $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$*, extending the inspected bisection window through the same modulus budget preserves the residual bracket read. The comparison row* S *is transported by* H*, replayed by* C*, and consumed by the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route* R *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance data. No branch extension may replace* S *by real trichotomy or by host real subtraction.*

Proof. Begin with the residual bracket S at the precision selected by M. Its endpoint and target reads are already attached to the displayed $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

When the bisection window is extended, the replay row C requires the next read to use the same modulus budget and dyadic tolerance route. Componentwise transport in H moves the endpoint, target, and residual rows together, so the sign or comparison data remains tied to the same packet.

The bisection row R consumes exactly that transported residual bracket. Because the carrier has no trichotomy coordinate and no host subtraction coordinate, stability is only the finite persistence of S along the displayed branch route. □

Theorem 2396.7 (DarbouxProperty RegSeqRat readback)**.** *Let $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *seal exposed by E is preceded by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q of the branch retained by R. The readback uses the dyadic tolerance data named in* $\mathsf{DyadicRatCore}^{\uparrow}$ *and the regular rational schedule of* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal real-name row is public.*

Proof. Read the rows following the bisection handoff. The route R retains the displayed branch, while Q is the only row that converts the retained branch windows into regular rational data.

The dyadic budget already used by S and M supplies the tolerance side of the readback. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q then presents the requested rational windows before the terminal row E is available.

Finally, H, C, P, N keep the readback attached to the same carrier and NameCert. The real seal is therefore a finite readback boundary over Q, not an assertion of host real equality, selected convergence, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DarbouxProperty}^{\uparrow}$ packet is generated from an interval row, a ContinuousMap row, endpoint Real rows, a target Real row, a finite residual-bracket row, a modulus budget, an IntermediateValueBisection route, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$, seed)

Scope closed: Definition 3348.1, Definition 3348.2, Definition 3348.3, Definition 3348.4, Definition 3348.5, Definition 3348.6, and Definition 3348.7 seed the finite Darboux target-value route and its real-seal boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient real equality, a selected intermediate point, trichotomy, ambient RealUp completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-bracket stability, intermediate-value bisection handoff, regular rational readback, and real-seal nonescape for BEDC.Derived.DarbouxPropertyUp.

Chapter 2397

A Concrete Naming Certificate for $\mathsf{LocatedCompleteMetric}^{\uparrow}$

Following Bishop and Bridges on constructive metric completeness, the $\mathsf{LocatedCompleteMetric}^{\uparrow}$ packet records a complete metric object whose Cauchy-limit service is located at every finite precision request. It sits after the completion surface of chapter 1506, the zero-distance classifier boundary of chapter 1411, the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, and the finite observation schedule of chapter 126. The packet names a constructive complete-metric carrier with displayed location and limit witnesses; it is not a quotient completion class, a selected representative, countable choice, host equality of points, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2397.1 (LocatedCompleteMetric carrier)**.** *A* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, R, Q, W, A, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing the metric carrier, completion handoff, and finite tolerance surface.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row, requiring zero-distance comparisons to pass through the displayed separated metric boundary.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row, used only after finite distance observations have been routed through regular rational readback.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for Cauchy approximants, limit budgets, and distance witnesses.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite observation schedule for Cauchy data and locatedness requests.*
6. *A records componentwise* hsame *transport for the metric-completion source, separated classifier, real seal, regular rational readback, stream windows, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a finite Cauchy request through W, Q, R and then through the separated classifier S over the completion source M.*

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$ *row.*

The carrier exports a displayed complete-metric service only through these rows. It has no coordinate for quotient completion equality, selected limits, a countable-choice schedule, host equality of carried points, or an unrestricted completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCompleteMetricUp.LocatedCompleteMetricTasteGate single carrier alignment`

Theorem 2397.2 (LocatedCompleteMetric NameCert obligation surface)**.** *For every accepted* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier* $L = (M, S, R, Q, W, A, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the metric-completion source, separated classifier, real distance seal, regular rational readback, stream-window schedule, componentwise transport, continuation replay, provenance, and local naming row;*

2. *located request admission, requiring every precision request to be read through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W before any limit witness is exposed;*

3. *regular readback stability, requiring Cauchy approximants, limit budgets, and distance witnesses to pass through Q before the* $\mathsf{Real}^{\uparrow}$ *seal R is consumed;*

4. *separated classifier stability, requiring zero-distance uniqueness for displayed limits to be compared through S over the same completion source M;*

5. *ledger non-escape, excluding quotient completion classes, selected representatives, countable-choice schedules, host equality of carried points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3349.1. The displayed tuple contains exactly the rows $M, S, R, Q, W, A, C, P, N$, so carrier admission and local naming are exhausted by those coordinates.

A located request must enter through W. That row supplies the finite observation schedule, and Q turns the scheduled Cauchy approximants and requested budgets into regular rational readback before any terminal real-distance seal is visible.

The separated comparison is then confined to S over the same metric-completion source M. Transport through A and replay through C preserve the displayed route, while P records that no excluded quotient, selector, choice, host-equality, or ambient-completeness coordinate is exported. □

Theorem 2397.3 (LocatedCompleteMetric Cauchy-limit witness)**.** *Let $L = (M, S, R, Q, W, A, C, P, N)$ be an accepted* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier. Every Cauchy-limit witness served by L factors through the finite route*

$$W \longmapsto Q \longmapsto R \longmapsto M, S \longmapsto A, C, P, N.$$

Thus the witness is a located finite-readback service over the displayed metric-completion source and separated classifier, not a selected host limit, quotient representative, choice sequence, or ambient completeness assertion.

Proof. Begin with a Cauchy-limit request against L. The only observation schedule in the carrier is W, so the request first fixes a finite stream window.

The regular rational row Q supplies the finite readback for the Cauchy approximants and the requested limit budget. The real-distance seal R is read after that regular handoff, so a downstream consumer receives only displayed distance information.

The completion source M and separated classifier S package the witness as a complete-metric service and compare any two displayed limits through zero-distance classification. The structural rows A, C, P, N transport, replay, source, and name the same route, so the excluded host-limit and quotient data have no row from which to be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3349.1, Definition 3349.2, and Definition 3349.3 enumerate the carrier, NameCert obligation surface, and Cauchy-limit witness obligations over MetricCompletion, SeparatedMetric, Real, RegSeqRat, StreamName, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, selected representatives, countable choice, host equality of carried points, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located request admission, regular readback stability, separated classifier stability, and ledger non-escape for BEDC.Derived.LocatedCompleteMetricUp.

Constructive story: A $\mathsf{LocatedCompleteMetric}^{\uparrow}$ packet is generated from a MetricCompletion source row, a SeparatedMetric classifier row, a RealUp distance seal, RegSeqRat readback, StreamName windows, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2398

A Concrete Naming Certificate for $\mathsf{CauchyCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley-style completion interfaces, the $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ packet records the finite handoff surface between metric-completion, uniform-completion, named-window, dyadic-tolerance, regular-readback, and Real-seal consumers. It sits over the metric completion route of chapter 1506, the uniform completion route of chapter 1473, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names the completion boundary without importing quotient stream equality, selected limits, ambient completeness, countable choice, or host completion operators.

Definition 2398.1 (Cauchy completion boundary carrier)**.** *A* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the metric Cauchy completion request and its displayed finite source route.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row naming the uniform completion request and its entourage-facing handoff route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the metric and uniform completion reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to budget Cauchy comparisons, entourage requests, and Real-facing seal reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same finite windows and dyadic tolerances.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after M, U, S, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric-completion, uniform-completion, window, dyadic, readback, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing read through $M, U, S, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$ *row.*

The classifier accepts only packets whose metric-completion row, uniform-completion row, shared finite windows, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming are displayed in this order. It has no coordinate for quotient streams, selected representatives, selected limits, ambient completeness, countable choice, or host completion operators.

Lean4 ✓ `BEDC.Derived.CauchyCompletionBoundaryUp.CauchyCompletionBoundaryTasteGate single carrier alignment`

Theorem 2398.2 (Cauchy completion boundary NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *carrier* $B = (M, U, S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion row, uniform-completion row, shared finite-window row, dyadic tolerance ledger, regular rational readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-boundary read must consume the rows* $M, U, S, D, R, E, H, C, P, N$ *before it exports a Real-facing completion handoff, and no obligation is stored outside those rows.*

Proof. Project the accepted carrier. The projection exposes the metric-completion request, uniform-completion request, shared windows, dyadic tolerances, regular readback, Real seal, transport, replay, provenance, and local name as one finite packet.

Read stability from the row order. A consumer that asks for a completion-facing handoff must pass through the shared $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before R or E can be inspected, so the metric and uniform routes cannot escape into a host completion object.

Finally apply coordinate exhaustion. Quotient streams, selected representatives, selected limits, ambient completeness, countable choice, and host completion operators are not coordinates of B, and the hsame, Cont, Pkg, and NameCert rows preserve the displayed packet rather than adding new analytic data. □

Theorem 2398.3 (Cauchy completion boundary finite handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer citing an accepted* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *packet receives only the finite handoff*

$$M \longmapsto U \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The handoff supplies the metric-completion route, uniform-completion route, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming. It supplies no quotient stream equality, selected representative, selected limit, countable-choice schedule, ambient completeness theorem, or host completion operator.

Proof. Begin with a consumer of the boundary packet. Carrier projection gives the rows $M, U, S, D, R, E, H, C, P, N$ and no other completion-facing coordinate.

The shared observation route is finite. The row S stores the inspected windows, D stores the dyadic budget, R stores the regular readback, and E is terminal with respect to the displayed Real-facing seal.

The structural rows then keep the same route under transport and replay. Since the excluded quotient, choice, limit, completeness, and host-completion objects are absent from the carrier, every exported handoff is exhausted by the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ packet is generated from MetricCompletion and UniformCompletion rows, shared StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionBoundary}\uparrow}$, seed)

Scope closed: Definition 3350.1, Definition 3350.2, and Definition 3350.3 seed the finite completion-boundary handoff for MetricCompletion, UniformCompletion, and RealUp consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, countable choice, ambient completion, host completion operators, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, dyadic tolerance routing, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionBoundary

Chapter 2399

A Concrete Naming Certificate for $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$

Following Kelley and Bourbaki on metric completion, the $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ packet records the standard dense embedding of a metric carrier into its completion as finite BEDC data. It reads the metric source of chapter 57, the completion row of chapter 1506, the separated comparison surface of chapter 1411, the uniform completion handoff of chapter 1473, and the regular rational readback of chapter 301. The packet gives completion-uniqueness and universal-property consumers a visible dense-image route without quotient representatives, selected limits, countable choice, host equality of completion classes, or an ambient completion object.

Definition 2399.1 (MetricCompletionDenseEmbedding carrier)**.** *A* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, C, E, S, U, R, H, K, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Metric}^{\uparrow}$ *source row, carrying the source carrier, distance classifier, and finite metric reads inherited from chapter 57.*
2. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, carrying the completion-facing Cauchy surface of chapter 1506.*
3. *E is the dense-image row, recording the displayed embedding route from a source point row in X to its completion name in C and the finite witness rows saying that every inspected completion neighbourhood is reached by a source image.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row from chapter 1411, used only after the source and completion rows have been displayed.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row from chapter 1473, recording the uniform completion consumer route that reads the same dense image surface.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row from chapter 301, exposing the finite regular-rational observations attached to completion-neighbourhood and distance-budget reads.*
7. *H records componentwise* hsame *transport for $X, C, E, S, U, R, K, P, N$.*
8. *K records displayed* Cont *replay from source metric reads through completion names, dense-image witnesses, separated comparisons, uniform completion handoff, and regular rational readback.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$*, the five sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient representatives, selected Cauchy limits, countable choice, host equality of completion classes, or a detached completion uniqueness theorem.

Lean4 ✓ `BEDC.Derived.MetricCompletionDenseEmbeddingUp.TasteGate.MetricCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 2399.2 (MetricCompletionDenseEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *carrier* $D = (X, C, E, S, U, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionDense}}$ *surface consists exactly of carrier admission for the source metric row, completion row, dense-image row, separated comparison row, uniform completion handoff, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must consume* X, C, E *before it can read* S, U, R, H, K, P, N*.*

Proof. Project the carrier of Definition 3351.1. The rows X, C, E expose the source metric, completion carrier, and dense image surface as displayed finite packet data.

The comparison and handoff rows are downstream. The separated comparison S reads the same source and completion rows, the uniform completion handoff U follows the same dense-image route, and the regular rational readback R exposes only the finite observation data attached to those reads.

Finally H, K, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient representatives, selected limits, countable choice, or host completion equality, the listed rows exhaust the local NameCert obligations. □

Theorem 2399.3 (MetricCompletionDenseEmbedding handoff)**.** *Every completion-uniqueness, separated-completion, or uniform-completion consumer of an accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{Metric}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{DenseImage}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}.$$

The exported handoff is the displayed dense embedding of the metric source into its completion; it is not a quotient-representative selector, an ambient completion theorem, a host equality comparison, or a choice principle.

Proof. Start with a completion-facing consumer. By Definition 3351.2, the consumer first reads the source metric row X, the completion row C, and the dense-image row E.

The sibling chapters then determine the downstream route. The separated metric comparison is read through chapter 1411, the uniform handoff through chapter 1473, and the regular rational windows through chapter 301.

The structural rows H, K, P, N preserve that same finite route under transport, replay, provenance, and naming. Because the carrier has no row for representatives, selected limits, countable choice, host equality, or an ambient completion theorem, every accepted handoff remains the displayed dense embedding surface. □

Theorem 2399.4 (MetricCompletionDenseEmbedding universal-property route)**.** *Every completion universal-property read using the dense image of an accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$

packet factors through the source metric row, completion row, dense image row, separated comparison row, uniform handoff row, RegSeqRat readback row, componentwise transport, displayed replay, provenance, and local NameCert *row. The route exports the source-to-completion dense embedding needed by universal-property consumers, not a quotient-representative selector, a selected limit, countable choice, host equality of completion classes, or an ambient completion theorem.*

Proof. Use Definition 3351.2 to read the carrier surface. A universal-property consumer first receives the source metric row, completion row, and dense image row, so its dense-source request is tied to displayed packet data.

Then apply the handoff of Definition 3351.3. The separated comparison, uniform completion handoff, and RegSeqRat readback rows are the only downstream completion-facing rows available to the consumer.

Finally the transport, replay, provenance, and local naming rows preserve that same route. Since no excluded host-side principle is a carrier coordinate, the universal-property read factors through the dense embedding surface alone. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ consumer that reaches a completion-uniqueness or uniform-completion handoff while omitting the source metric row, completion row, dense-image row, separated comparison row, or regular rational readback refutes the packet.

Independence witness: metric, metriccompletion, separatedmetric, uniformcompletion, regseqrat: no listed sibling is bijective with $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ because the packet jointly carries source metric data, completion data, dense-image witnesses, separated comparison, uniform handoff, regular rational readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a metric source row, metric-completion row, dense-image row, separated comparison row, uniform completion handoff, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3351.1, Definition 3351.2, and Definition 3351.3 seed the standard dense embedding route from a metric carrier into its completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected Cauchy limits, countable choice, host equality of completion classes, completion uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-image exactness, separated comparison handoff, uniform completion handoff, regular rational readback, and TasteGate field faithfulness for BEDC.Derived.MetricCompletionDenseEmb

Chapter 2400

A Concrete Naming Certificate for ClosedPiDomainSubjectReduction$^{\uparrow}$

The ClosedPiDomainSubjectReduction$^{\uparrow}$ packet realizes the Pi-domain branch of the closed MetaCIC subject-reduction frontier as a finite NameCert carrier. It sits beside the four-row discharge socket of chapter 719, the discharge ledger of chapter 720, the beta-substitution packet of chapter 716, the closed substitution boundary of chapter 538, and the typed beta congruence trigger of chapter 760. The packet names the Pi-domain obligation as displayed closed term data; it does not export an unconditional closed subject-reduction theorem.

Definition 2400.1 (ClosedPiDomainSubjectReduction carrier)**.** *A* ClosedPiDomainSubjectReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, D, D', C, Y, Z, B, H, R, Q, N)$$

with the following displayed rows:

1. *T is the displayed closed Pi term row, naming the source term $\Pi D.\, C$ and its closed context boundary.*
2. *D is the original domain row, carrying the domain annotation inspected before the congruence step.*
3. *D' is the reduced domain row reached by the displayed beta step from D.*
4. *C is the codomain row, read under the binder induced by the displayed domain.*
5. *Y is the typing row, recording the source typing judgement for $\Pi D.\, C$ and the target typing judgement demanded for $\Pi D'.\, C$.*
6. *Z is the closedness row, requiring the Pi term, original domain, reduced domain, and codomain reads to remain inside the same closed MetaCIC boundary.*
7. *B is the beta-step and typed congruence row, consuming the displayed Pi-domain beta step and the trigger surface of chapter 760.*
8. *H records componentwise* hsame *transport for $T, D, D', C, Y, Z, B, R, Q, N$.*
9. *R records displayed* Cont *replay from the source Pi term through the domain step, typing row, closedness row, and subject-reduction handoff.*

10. *Q is the* Pkg *provenance over chapter 719, chapter 720, chapter 716, chapter 538, chapter 760,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying the closed Pi term, both domain rows, the codomain row, the typing row, the closedness row, the Pi-domain beta step, transport, replay, provenance, and local naming. It has no coordinate for a blanket closed subject-reduction theorem, a host conversion rule, erased derivation semantics, quotient syntax equality, or an untracked binder context.

Lean4 ✓ `BEDC.Derived.ClosedPiDomainSubjectReductionUp.ClosedPiDomainSubjectReductionTasteGate`
`single carrier alignment`

Theorem 2400.2 (ClosedPiDomainSubjectReduction obligations)**.** *For every accepted* ClosedPiDomainSubjectRed *carrier $P = (T, D, D', C, Y, Z, B, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed closed Pi term, original domain, reduced domain, codomain, typing row, closedness row, beta-step row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The Pi-domain preservation read is accepted only when B displays the domain beta step and Y, Z display the corresponding typing and closedness rows.*

Proof. Project the carrier of Definition 3352.1. The rows T, D, D', C display the Pi term and the two domain readings, while Y and Z display the typing and closedness boundaries attached to the same context.

The beta-step row B is then read as the Pi-domain congruence trigger. It is not interchangeable with the beta-substitution row, the application-argument row, or the lambda-domain row of the subject-reduction socket, because the target branch changes the displayed Pi domain while preserving the codomain under the induced binder.

The structural rows H, R, Q, N transport, replay, source, and name only that finite packet. Since no row stores a host conversion rule, erased derivation semantics, quotient syntax equality, or a closed subject-reduction theorem, the displayed rows exhaust the local NameCert obligations. □

Theorem 2400.3 (ClosedPiDomainSubjectReduction handoff)**.** *Every subject-reduction discharge ledger read that consumes the Pi-domain branch from an accepted* $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ *packet factors through the displayed route*

$$\Pi D.\, C,\ D,\ D',\ C,\ Y,\ Z,\ B,\ H,\ R,\ Q,\ N.$$

The handoff supplies the Pi-domain obligation row to the subject-reduction discharge socket and ledger; it does not supply the beta-substitution row, application-argument row, lambda-domain row, or an unconditional closed subject-reduction theorem.

Proof. Start with a subject-reduction consumer of the Pi-domain branch. The socket and ledger chapters, chapter 719 and chapter 720, require a distinct Pi-domain row before the parameterised subject-reduction route can be read.

The present carrier supplies that row only through T, D, D', C, Y, Z, B. These rows display the source Pi term, the original and reduced domains, the codomain, the typing judgement, the closedness boundary, and the beta-step trigger.

Finally H, R, Q, N preserve and name the same finite handoff. The beta-substitution preservation packet, closed substitution boundary, and typed beta congruence trigger remain sibling inputs rather than hidden substitute proofs, so the handoff is exactly the Pi-domain row and nothing stronger. □

Falsifiable prediction: Any accepted $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ consumer reaching a subject-reduction conclusion without the displayed Pi-domain beta step and typing row refutes the carrier.

Independence witness: metacic-subject-reduction-obligation, betasubstitutionpreservation, closed-substitutionboundary, typedbetacongruencetrigger: none is bijective with $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ because this packet jointly carries the closed Pi term, original domain, reduced domain, codomain, typing row, closedness row, Pi-domain beta step, transport, replay, provenance, and local naming.

The Lean TasteGate file for $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ must provide explicit Nontrivial ClosedPiDomainSu and $\mathsf{FieldFaithful}\ \mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ instances preserving $T, D, D', C, Y, Z, B, H, R, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ packet is generated from a closed Pi term row, original domain row, reduced domain row, codomain row, typing row, closedness row, beta-step row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3352.1, Definition 3352.2, and Definition 3352.3 seed the finite Pi-domain subject-reduction obligation packet for the closed MetaCIC discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta-substitution preservation, application-argument stability, lambda-domain preservation, unconditional closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Pi-domain beta-step exactness, typing-row exposure, closedness-row exposure, subject-reduction ledger handoff, and TasteGate field faithfulness for BEDC.Derived.ClosedPiDomainSubjectRed

Chapter 2401

A Concrete Naming Certificate for $\mathsf{FiniteTraceInductionGap}^{\uparrow}$

Following the Bishop and Brouwer distinction between finite verification and an induction schema, the $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ packet records the finite-observation side of Definition 9393.4 and the non-internal-induction boundary of Definition 9393.5. It sits beside the finite-trace GAP socket of chapter 504, the transcendental induction socket of chapter 498, and the closed-observation ledger of chapter 616. The packet names a finite verification trace together with the GAP socket exposed by any attempted universal closure; it does not install a substrate-internal universal quantifier, dependent induction schema, host recursor, or future-observation oracle.

Definition 2401.1 (Finite trace induction gap carrier)**.** *A* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, O, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite verification trace row, listing only the observed indices and their closed-generation witness records.*
2. *O is the observed predicate row, storing the finite family of predicate reads actually checked by the trace.*
3. *A is the attempted induction-closure row, naming the requested passage from the finite trace to a universal claim without admitting the passage as a carrier coordinate.*
4. *Q is the GAP socket row exposed when A asks for substrate-internal induction beyond the displayed trace.*
5. *H records componentwise* hsame *transport for the finite trace, observed predicate rows, attempted closure row, GAP socket, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each observed predicate read through T, O, A, Q, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$, $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, $\mathsf{TranscendentalInductionSocket}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, BHist, Gap, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$ *row.*

The classifier accepts only packets whose finite trace, observed predicate rows, attempted induction-closure row, GAP socket, transport, replay, provenance, and local naming are all displayed. It has

no coordinate for a universal quantifier over unobserved indices, a substrate-internal induction schema, a host dependent recursor, an infinite observation ledger, or a future-observation oracle.

Lean4 ✓ BEDC.Derived.FiniteTraceInductionGapUp.FiniteTraceInductionGapTasteGate single carrier alignment

Theorem 2401.2 (Finite trace induction gap NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *carrier* $G = (T, O, A, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite trace row, observed predicate rows, attempted induction-closure row, GAP socket row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that asks for induction closure must expose* A *and then consume* Q *as a GAP socket; no substrate-internal universal statement is exported by the packet.*

Proof. Project the accepted carrier. The projection exposes the finite trace T, the observed predicate rows O, the attempted closure row A, the GAP socket Q, and the structural rows H, C, P, N.

Read the induction request through A. If the consumer stays within the finite trace, it can use only the displayed rows T and O. If it asks for all unobserved indices, the next displayed row is the GAP socket Q, not a universal witness.

The transport and replay rows preserve that boundary. Since no universal quantifier, dependent induction schema, host recursor, infinite ledger, or future-observation oracle is a coordinate of G, the local NameCert obligations are exhausted by the finite trace and GAP route. □

Theorem 2401.3 (Finite trace induction gap non-escape)**.** *Every induction-facing consumer of an accepted* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *packet factors through*

$$T \longmapsto O \longmapsto A \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer may read the observed finite trace and the displayed predicate witnesses, but any attempted export of a universal induction claim reaches only the GAP socket. The packet exports no substrate-internal universal quantifier, no closed dependent recursor, no infinite observation object, and no promise over future observations.

Proof. Start with the carrier rows of Definition 3353.1. The trace row T and observed predicate row O give exactly the finite witness records already present in the closed observational system.

The attempted closure row A names the requested induction passage. It is deliberately followed by the GAP socket row Q, so a consumer that asks for unobserved indices receives a visible boundary rather than a hidden induction rule.

Finally replay and transport the same route through C and H, then read provenance and local naming through P, N. The excluded universal, recursor, infinite-ledger, and future-observation objects are absent from the carrier, so the displayed GAP route is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ packet is generated from a finite verification trace, observed predicate rows, an attempted induction-closure row, a GAP socket, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$, seed)

Scope closed: Definition 3353.1, Definition 3353.2, and Definition 3353.3 seed the finite verification trace and induction-GAP boundary for closed observational systems.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a substrate-internal universal quantifier, dependent induction schema, host recursor, infinite observation ledger, future-observation oracle, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-trace visibility, observed-predicate stability, induction-GAP routing, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTraceInductionGapUp.

Chapter 2402

A Concrete Naming Certificate for $\mathsf{OvertReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OvertReal}^{\uparrow}$ packet records positive information about real data as finite witnessable contact, not as a host open-set membership predicate. It sits over the real-name boundary of chapter 38, the located source interval of chapter 1709, the finite interval-cover route of chapter 1885, and the one-sided observation rows of chapter 1446 and chapter 1447. The packet supplies an L10 positive-information interface for downstream real and interval consumers without claiming classical open-cover compactness, arbitrary open-set membership, trichotomy, selected points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2402.1 (OvertReal carrier)**.** *An* $\mathsf{OvertReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, I, C, L, U, W, A, H, T, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row naming the real datum that may be observed only through the displayed real-name route.*
2. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, exposing endpoint names, radius data, locator reads, locatedness moduli, and metric-bracket rows around the inspected real datum.*
3. *C is the* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row, recording a finite located cover request through rational intervals, dyadic mesh data, refinement rows, finite windows, and readback data.*
4. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *positive lower-observation row. It records finite inhabited lower rational information that may witness nonempty contact from below.*
5. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *positive upper-observation row. It records finite upper-bound information that keeps the contact request located rather than host-open.*
6. *W is the finite witness row naming the particular cover cell, interval bracket, lower read, and upper read used by the positive-information claim.*
7. *A is the anti-escape ledger saying that the packet exports only witnessable contact and finite cover handoff, not arbitrary open-set membership or open-cover compactness.*
8. *H records componentwise* hsame *transport for $R, I, C, L, U, W, A, P, N$.*
9. *T records displayed* Cont *replay from the real seal to the located interval, from the interval to the cover request, and from the cover request to the lower and upper contact rows.*

10. *P is the* Pkg *provenance over* $\mathsf{OvertReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, I, C, L, U, W, A, H, T, P, N$. It has no coordinate for host open sets, detached membership predicates, arbitrary open covers, compactness extraction, trichotomy, selected representatives, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.OvertRealUp.OvertRealTasteGate single carrier alignment`

Theorem 2402.2 (OvertReal NameCert obligations)**.** *For every accepted* $\mathsf{OvertReal}^{\uparrow}$ *carrier*

$$O = (R, I, C, L, U, W, A, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$ *surface consists exactly of the finite positive-information obligations over the displayed packet:*

1. *real-source admission, requiring every real-facing read to pass through the terminal* $\mathsf{Real}^{\uparrow}$ *row R;*
2. *located-interval admission, requiring endpoint, radius, locator, locatedness, and metric-bracket requests to pass through I;*
3. *finite-cover handoff, requiring every positive contact claim to cite the displayed* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row C;*
4. *one-sided observation exactness, requiring the lower and upper contact reads to be exactly L and U;*
5. *witness-row exactness, requiring the contact witness to be the finite row W and not a selected host point;*
6. *transport, replay, provenance, and local naming stability through H, T, P, N;*
7. *non-escape through A, excluding host open-set membership, arbitrary open-cover compactness, trichotomy, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 3354.1. The real datum is visible only through R, and the only interval information attached to that datum is the located row I.

The positive-information route next passes through C. Because C is a $\mathsf{LocatedIntervalCover}^{\uparrow}$ row, the cover request is finite, located, and routed through displayed rational and dyadic refinement data before any contact claim is available.

The one-sided rows L and U provide the observable lower and upper information. The witness row W binds these reads to one displayed cover cell and interval bracket, so the packet never replaces positive contact by a selected point or by detached host membership.

Finally H, T, P, N transport, replay, source, and name the same rows, while A records the forbidden exits. Since no other coordinate occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.OvertRealUp.OvertRealCarrier semantic name certificate`

Theorem 2402.3 (OvertReal positive-cover handoff)**.** *Every* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *consumer that asks whether an accepted* $\mathsf{OvertReal}^{\uparrow}$ *packet has positive contact with a displayed located interval factors through*

$$R, I, C, L, U, W, A, H, T, P, N.$$

The handoff exposes the source $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *bracket, the finite* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *request, the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *one-sided contact rows, the finite witness row, and the structural transport, replay, provenance, and local naming rows. It exports no host open-set membership predicate, no open-cover compactness theorem, no selected point of contact, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the accepted carrier O. The target consumer is cover-facing, so its first available cover row is the displayed coordinate C.

Reading C forces the consumer to retain the source interval row I and the terminal real seal R, because the cover route is attached to the located interval around the real datum rather than to an external open-set predicate.

The contact claim is then witnessed by L, U, W. The lower and upper rows provide the one-sided positive information, and W records the finite cover cell and bracket in which that information is consumed.

The rows A, H, T, P, N add no extra carrier coordinate. They only block escape, transport the displayed packet, replay the displayed route, record provenance, and name the local certificate. Thus every positive-cover handoff factors through exactly the stated finite surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$.

Constructive story: An $\mathsf{OvertReal}^{\uparrow}$ packet is generated from a terminal RealUp seal, a LocatedRealIntervalUp source interval, a LocatedIntervalCoverUp finite cover request, LowerRealUp and UpperRealUp one-sided contact rows, a finite witness row, an anti-escape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$, seed)

Scope closed: Definition 3354.1, Definition 3354.2, and Definition 3354.3 seed the finite positive-information surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim classical open-set membership, arbitrary open-cover compactness, selected points of contact, trichotomy, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, real-source exactness, located-interval exactness, finite-cover handoff, one-sided contact exactness, witness-row exactness, transport stability, replay stability, and non-escape for BEDC.Derived.OvertRealUp.

Chapter 2403

A Concrete Naming Certificate for $\mathsf{CertificateMachine}^{\uparrow}$

The $\mathsf{CertificateMachine}^{\uparrow}$ packet realizes the certificate-machine branch of subsection 9462.28.23 as finite BEDC carrier data. It names the surface on which a machine-like object is admitted only through a finite kernel row, accepted-observation rows, branch-refusal rows, audit replay, provenance, and local NameCert evidence. The packet is not an ordinary Turing-machine identity claim, not a semantic truth oracle, not a host execution trace, and not a replacement for the kernel and NameCert obligations it records.

Definition 2403.1 (CertificateMachine carrier)**.** *A* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, O, B, A, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite certificate-kernel row. It lists the admitted constructors, candidate rows, acceptance stamps, and refusal slots visible to this machine surface.*
2. *O is the accepted-observation row, recording the finite observations the machine has admitted as certificate-bearing rows.*
3. *B is the branch-refusal row, recording rejected or unlicensed branches together with the local reason each branch cannot be read as accepted data.*
4. *A is the audit-replay row, recording the finite replay route from generated candidate through accepted observation or branch refusal.*
5. *R is the replay-result row, naming the public result that may be consumed after the audit replay finishes.*
6. *S is the soundness-boundary row saying that the machine exports certificate acceptance and refusal data, not a host truth predicate or an external operational semantics.*
7. *H records componentwise* hsame *transport for K, O, B, A, R, S, P, N.*
8. *C records displayed* Cont *replay from kernel row to observation or refusal row, from that row to audit replay, and from audit replay to the public result.*
9. *P is the* Pkg *provenance over* $\mathsf{CertificateMachine}^{\uparrow}$*, finite certificate kernels, accepted observations, branch refusals, audit replay,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, B, A, R, S, H, C, P, N$. It has no coordinate for host machine equality, ordinary Turing-machine identity, semantic truth, unbounded execution logs, unlicensed branch acceptance, or external proof authority.

Lean4 ✓ `BEDC.Derived.CertificateMachineUp.CertificateMachineTasteGate single carrier alignment`

Theorem 2403.2 (CertificateMachine NameCert obligations)**.** *For every accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier*

$$M = (K, O, B, A, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *kernel admission, requiring every candidate, constructor, acceptance stamp, and refusal slot to pass through K;*
2. *accepted-observation exactness, requiring every admitted observation to be one of the rows displayed in O;*
3. *branch-refusal exactness, requiring every rejected branch read to pass through B with its local refusal reason;*
4. *audit-replay determinacy, requiring every public result to be replayed through A before R is consumed;*
5. *soundness-boundary non-escape, requiring S to exclude semantic truth, ordinary Turing-machine identity, host execution logs, and external proof authority;*
6. *transport, continuation, provenance, and local naming stability through H, C, P, N.*

Lean follow-up must provide $\mathsf{Nontrivial}(\mathsf{CertificateMachine}^{\uparrow})$ *and* $\mathsf{FieldFaithful}(\mathsf{CertificateMachine}^{\uparrow})$ *instances before the object may be consumed as a nondegenerate field-facing certificate machine.*

Proof. Project the carrier of Definition 3355.1. The only kernel coordinate is K, so every candidate and acceptance stamp begins from that finite certificate-kernel row.

Accepted observations and refusals then split through O and B. The accepted side records only admitted observation rows, while the refusal side records branch-local reasons for rejection.

The audit route passes through A before the public result R is available. Thus a consumer cannot bypass replay by citing a host execution log or an unbounded operational trace.

Finally S, H, C, P, N bind the boundary, transport the same rows, replay the displayed route, record provenance, and name the local certificate. Since no coordinate adds semantic truth, ordinary Turing-machine identity, unlicensed branch acceptance, or external proof authority, the listed obligations exhaust the NameCert surface. □

Theorem 2403.3 (CertificateMachine audit-replay non-escape)**.** *Every public read of an accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier factors through*

$$K, O, B, A, R, S, H, C, P, N.$$

An accepted observation reaches the public result only by the route K, O, A, R, and a rejected branch reaches the public refusal surface only by the route K, B, A, R. Both routes keep the soundness boundary S visible and preserve the same transport, continuation, provenance, and local naming rows. No read exports host machine equality, semantic truth, unbounded execution, unlicensed branch acceptance, or an external proof oracle.

Proof. Start from the displayed carrier M. There are exactly two observation-facing routes after the kernel row: the accepted-observation row O and the branch-refusal row B.

In the accepted case, continuation replay through C sends K and O into the audit row A before the public result R can be read. In the refusal case, the same replay discipline sends K and B into A before the refusal result is visible.

The soundness boundary S is carried with both routes. It excludes the stronger readings that would identify the packet with an ordinary Turing machine, a semantic truth oracle, a host execution log, or an external proof source.

The remaining rows H, P, N only transport, source, and name the packet. Hence every public read remains inside the displayed finite surface. □

Theorem 2403.4 (CertificateMachine branch-refusal determinacy)**.** *Let $M = (K, O, B, A, R, S, H, C, P, N)$ be an accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *packet. Every refused branch of M is determined by the finite kernel row K, the branch-refusal row B, the audit-replay row A, and the soundness-boundary row S. Before the refused branch can be consumed as a MetaCIC audit row, its read must pass through the accepted-kernel surface of chapter 549, the axiom-query ledger of chapter 654, and the closed-generator audit surface of chapter 709.*

Proof. Project the carrier of Definition 3355.1. The refusal side has only the kernel row K, the branch-refusal row B, the audit replay row A, and the soundness boundary S before the public result row can be read.

Apply audit-replay non-escape, Definition 3355.3. A rejected branch reaches the public refusal surface only by the route K, B, A, R, with S still visible as the boundary against host truth, unbounded execution, and external proof authority.

The MetaCIC sibling lattice supplies the concrete audit route for those rows. Kernel admission is read through chapter 549, proof-supply cleanliness through chapter 654, and closed-generator replay through chapter 709.

Thus the refused branch is determined by the displayed finite rows of the same packet and by the named sibling ledgers. No separate kernel-only or host-execution route can consume it as a MetaCIC audit row. □

Falsifiable prediction: Every $\mathsf{CertificateMachine}^{\uparrow}$ read factors through one finite certificate-kernel row, one accepted-observation row or one branch-refusal row, one audit-replay row, one replay-result row, one soundness-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: kernelacceptancewitness, axiomqueryledger, inscriptionevent, observerfilter, closedgeneratoraudit: $\mathsf{CertificateMachine}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries kernel admission, accepted observations, branch refusals, audit replay, result exposure, and soundness-boundary refusal as one finite machine packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateMachine}^{\uparrow}$ packet is generated from a finite certificate-kernel row, accepted-observation rows, branch-refusal rows, audit replay, replay result exposure, a soundness-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3355.1, Definition 3355.2, and Definition 3355.3 seed the finite machine surface over certificate kernels, accepted observations, branch refusals, audit replay, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ordinary Turing-machine identity, semantic truth, host execution semantics, unbounded trace replay, unlicensed branch acceptance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, kernel exactness, accepted-observation exactness, branch-refusal exactness, audit-replay determinacy, soundness-boundary non-escape, transport stability, replay stability, and the Nontrivial(CertificateMachine$^{\uparrow}$) and FieldFaithful(CertificateMachine$^{\uparrow}$) Lean instances for BEDC.Derived.CertificateMachineUp.

Chapter 2404

A Concrete Naming Certificate for GeneratingFunction$^{\uparrow}$

Following Wilf and Stanley, a GeneratingFunction$^{\uparrow}$ packet records a sequence by its coefficient stream and the finite algebraic windows through which coefficient extraction is read. It sits after the unary index discipline of chapter 28, the finite spine discipline of chapter 36, the stream-window interface of chapter 126, the real-series carrier of chapter 1581, and the coefficient-and-radius surface of chapter 1583. The packet is a traditional finite-combinatorics object in BEDC form: it exports coefficient readback and Cauchy-product handoff, not host analytic convergence or equality of functions.

Definition 2404.1 (GeneratingFunction carrier)**.** *A* GeneratingFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, Q, L, Pd, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the coefficient source row, indexed through* Nat$^{\uparrow}$ *and exposed only by finite* StreamName$^{\uparrow}$ *windows.*
2. *W is the coefficient-window ledger, listing the finite prefixes and requested coefficient indices visible to the packet.*
3. *Q is the coefficient readback row, turning each displayed window into a* RegSeqRat$^{\uparrow}$ *or finite combinatorial count row before any series consumer reads it.*
4. *L is the finite* List$^{\uparrow}$ *polynomial-prefix row, carrying the inspected initial polynomial truncations and their coefficient positions.*
5. *Pd is the coefficientwise classifier row for equality-facing reads of displayed prefixes.*
6. *M is the product handoff row, exposing the Cauchy convolution window used by* CauchySeriesProduct$^{\uparrow}$ *and* RealPowerSeries$^{\uparrow}$ *consumers.*
7. *H stores componentwise* hsame *transport for coefficient source, windows, readbacks, finite prefixes, classifier rows, product handoff, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from a coefficient request through the stream window, readback, polynomial prefix, classifier, and product-handoff rows.*
9. *P is the* Pkg *provenance over* GeneratingFunction$^{\uparrow}$*,* Nat$^{\uparrow}$*,* List$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* RealSeries$^{\uparrow}$*,* CauchySeriesProduct$^{\uparrow}$*,* RealPowerSeries$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^\uparrow}$ *row.*
The carrier contains no coordinate for a host analytic function, quotient series equality, selected convergence radius, countable-choice coefficient family, or ambient $\mathsf{Real}^\uparrow$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.GeneratingFunctionUp.GeneratingFunctionTasteGate single carrier alignment`

Theorem 2404.2 (GeneratingFunction NameCert obligations)**.** *For every accepted* $\mathsf{GeneratingFunction}^\uparrow$ *carrier* $K = (A, W, Q, L, Pd, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^\uparrow}$ *surface consists exactly of five finite rows: carrier admission for the displayed coefficient packet; coefficient-window readback through* A, W, Q, L*; coefficientwise classifier stability through* Pd, H, C*; Cauchy and product handoff through* M*; and ledger refusal for host functions, quotient series equality, analytic convergence, selected radii, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 3356.1. The accepted packet exposes only coefficient source, finite windows, readback rows, list-polynomial prefixes, classifier data, product handoff, transport, replay, provenance, and naming rows.

The coefficient-window readback is finite. A consumer first reads a $\mathsf{Nat}^\uparrow$ index request, then uses W to select a finite $\mathsf{StreamName}^\uparrow$ window, Q to read the coefficient data, and L to place the result in the displayed polynomial prefix.

Classifier stability is componentwise. The row Pd compares displayed prefixes only after the same windows and readbacks have been transported by H and replayed by C.

The Cauchy/product handoff is the row M. It exports exactly the finite convolution window needed by product and power-series consumers, so no host function equality, quotient series equality, analytic convergence theorem, selected radius, countable-choice family, or ambient completeness row enters the certificate. □

Theorem 2404.3 (GeneratingFunction coefficient readback)**.** *Every coefficient-facing consumer of an accepted* $\mathsf{GeneratingFunction}^\uparrow$ *packet factors through*

$$\mathsf{Nat}^\uparrow,\ A \longmapsto W \longmapsto Q, L \longmapsto Pd, H, C, P, N.$$

If the consumer asks for a product coefficient, the route additionally passes through the finite convolution ledger M *before any* $\mathsf{CauchySeriesProduct}^\uparrow$ *or* $\mathsf{RealPowerSeries}^\uparrow$ *handoff is visible. Thus the public readback is a finite coefficient-window certificate, not a host function evaluation, quotient equality of formal series, analytic convergence claim, selected infinite sum, or ambient* $\mathsf{Real}^\uparrow$ *completeness route.*

Proof. Begin with a displayed coefficient request. The only available index route is the $\mathsf{Nat}^\uparrow$ row named in the provenance, and the only available coefficient source is A.

The request then opens the selected finite window W. The readback row Q supplies the coefficient data, and the list-polynomial row L places the coefficient in a finite prefix rather than in an infinite host series.

For product-facing reads, use M after the same window and readback rows. The convolution comparison is therefore a displayed finite ledger handed to $\mathsf{CauchySeriesProduct}^\uparrow$ or $\mathsf{RealPowerSeries}^\uparrow$, not an equality of analytic functions.

The remaining rows Pd, H, C, P, N preserve and name this route. Since the carrier has no coordinate for host function evaluation, quotient series equality, analytic convergence, selected infinite sums, countable choice, or ambient completeness, every coefficient read is exhausted by the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{GeneratingFunction}^{\uparrow}$ packet is generated from a Nat-indexed coefficient source, finite StreamName coefficient windows, RegSeqRat or finite-count readbacks, List polynomial prefixes, a coefficientwise classifier, a Cauchy-product handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3356.1, Definition 3356.2, and Definition 3356.3 seed the finite coefficient-window certificate for generating functions and their product-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host analytic functions, quotient series equality, analytic convergence, selected radii, selected infinite sums, countable choice, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window readback, coefficientwise classifier stability, Cauchy/product handoff, and ledger refusal for BEDC.Derived.GeneratingFunctionUp.

Chapter 2405

A Concrete Naming Certificate for $\mathsf{MetricIdentification}^{\uparrow}$

Following Bourbaki and Kelley on the identification of zero-distance points in a pseudometric space, the $\mathsf{MetricIdentification}^{\uparrow}$ packet records the finite BEDC route from displayed pseudometric distance rows to a separated metric-facing classifier. It sits between the pseudometric and metric rows of chapter 57, the zero-distance classifier boundary of chapter 1411, the completion consumer of chapter 1506, and the terminal real-distance seal of chapter 38. The packet names the identification surface without forming quotient primitives, host equality classes, selected representatives, or ambient completion points.

Definition 2405.1 (MetricIdentification carrier)**.** *A* $\mathsf{MetricIdentification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, M, Z, S, C, T, R, H, K, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *source row, exposing point histories, distance histories, and pseudometric comparison ledgers.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *target row whose distance reads are allowed only after the identification classifier has accepted the source comparison.*
3. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance row, recording the displayed real-distance seal used to compare source points at distance zero.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier handoff, requiring every equality-facing read to pass through the separated metric boundary.*
5. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer boundary, available only after P, M, Z, S have displayed the finite identification route.*
6. *T records componentwise* hsame *transport for source points, distance rows, zero-distance witnesses, separated classifier rows, completion handoff rows, provenance rows, and local naming.*
7. *R records displayed* Cont *replay from a finite pseudometric comparison request through the zero-distance row and separated classifier to the metric and completion consumers.*
8. *H records the zero-distance history ledger: every accepted comparison supplies the same* BHist *source pair, the displayed real-distance row Z, and the classifier row by which the pair becomes identified for S.*

9. *K is the* Pkg *provenance over* $\mathsf{MetricIdentification}^{\uparrow}$, $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Q is the quotient-refusal ledger, stating that no primitive quotient object, host equality class, selected representative, or uninspected completion point is a carrier coordinate.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose source pseudometric, target metric, zero-distance Real row, separated classifier handoff, completion boundary, transport, replay, zero-distance history ledger, provenance, refusal ledger, and local naming are displayed. It compares accepted packets through those rows and through no host quotient relation.

Lean4 ✓ `BEDC.Derived.MetricIdentificationUp.MetricIdentificationTasteGate single carrier alignment`

Theorem 2405.2 (MetricIdentification NameCert obligations)**.** *For every accepted* $\mathsf{MetricIdentification}^{\uparrow}$ *carrier $I = (P, M, Z, S, C, T, R, H, K, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$ *surface consists exactly of carrier admission for the pseudometric source, metric target, Real zero-distance row, separated metric classifier handoff, metric-completion consumer boundary, componentwise* hsame *transport, displayed* Cont *replay, zero-distance history ledger, package provenance, quotient-refusal ledger, and local naming. Every public identification read must pass through the zero-distance history row H and the separated classifier row S before any metric or completion consumer can read it.*

Proof. Project the accepted carrier of Definition 3357.1. The displayed analytic rows are exactly P, M, Z, S, C, H, and the structural rows are exactly T, R, K, Q, N.

The zero-distance comparison is therefore not a host equality assertion. It is the row H, read over the source pseudometric row P and the Real seal Z, then transported through T and replayed through R.

The separated metric row S is the first equality-facing consumer of that history. The metric row M and completion boundary C can only consume the result after S has displayed the classifier handoff. Since Q records the absence of quotient primitives, representatives, host equality classes, and uninspected completion points, the listed rows exhaust the local NameCert obligations. □

Theorem 2405.3 (MetricIdentification completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricIdentification}^{\uparrow}$ *packet reads zero-distance identification through*

$$\mathsf{Pseudometric}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{SeparatedMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{MetricCompletion}^{\uparrow}.$$

The handoff is a finite classifier route from source pseudometric distance rows to a separated metric boundary. It is not a quotient completion, not a selected representative scheme, and not host equality of carried points.

Proof. Begin with a completion-facing read. The carrier exposes the completion boundary only as the row C, so the request must first inspect the pseudometric source P, the Real zero-distance row Z, and the zero-distance history ledger H.

The separated classifier S then supplies the equality-facing boundary. Only after that boundary is displayed can the metric row M and completion boundary C be consumed by the downstream route.

The structural rows T, R, K, N preserve the same displayed packet under transport, continuation replay, provenance, and naming. The refusal row Q excludes quotient primitives and selected representatives, so the completion handoff is exactly the finite route shown above. □

Falsifiable prediction: Every accepted $\mathsf{MetricIdentification}^{\uparrow}$ completion-facing read must expose a $\mathsf{Pseudometric}^{\uparrow}$ source row, a $\mathsf{Real}^{\uparrow}$ zero-distance row, a $\mathsf{SeparatedMetric}^{\uparrow}$ classifier handoff, a $\mathsf{Met}^{\uparrow}$ target row, a $\mathsf{MetricCompletion}^{\uparrow}$ consumer boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, quotient-refusal data, and local naming.

Independence witness: pseudometric, metric, real, separatedmetric, metriccompletion: $\mathsf{MetricIdentification}^{\uparrow}$ is not bijective to any listed sibling because it binds the zero-distance Real row and separated classifier handoff before the metric and completion consumers may read an identified source comparison.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3357.1, Definition 3357.2, and Definition 3357.3 seed the zero-distance identification route from pseudometric rows to separated metric and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient construction, selected representatives, host equality of points, completion existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance ledger exactness, separated classifier handoff, completion consumer nonescape, transport and replay stability, provenance exhaustion, quotient-refusal faithfulness, and TasteGate field faithfulness for BEDC.Derived.MetricIdentificationUp.

Constructive story: A $\mathsf{MetricIdentification}^{\uparrow}$ packet is generated from a Pseudometric source row, a Metric target row, a Real zero-distance row, a SeparatedMetric classifier handoff, a MetricCompletion consumer boundary, componentwise hsame transport, displayed Cont replay, zero-distance history ledgers, Pkg provenance, quotient-refusal data, and a local NameCert row.

Chapter 2406

A Concrete Naming Certificate for EquicontinuityFiniteNetSelector$^{\uparrow}$

Following the Arzela–Ascoli finite-net reading in Bishop constructive analysis, the EquicontinuityFiniteNetSelector$^{\uparrow}$ packet records the finite selector surface that turns an equicontinuous family over a compact metric carrier into a displayed net-indexed modulus handoff. It sits over the family ledger of chapter 413, the compact source of chapter 129, the pointwise graph rows of chapter 130, the finite net surface of chapter 1767, and the uniform-modulus consumer of chapter 369. The packet selects from finite displayed rows only; it is not an open-cover compactness principle, not a host function-space topology, not a choice operator, and not a selector over external points.

Definition 2406.1 (Equicontinuity finite-net selector carrier)**.** *An* EquicontinuityFiniteNetSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, K, G, N, R, L, U, H, C, P, A)$$

with the following displayed rows:

1. *F is the* EquicontinuityFamily$^{\uparrow}$ *source row, exposing the shared family ledger and its pointwise tolerance-radius data.*
2. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing compact finite-net and located coverage data for the same source metric.*
3. *G is the* ContinuousMap$^{\uparrow}$ *graph row family whose pointwise modulus rows are inspected by the selector.*
4. *N is the* FiniteEpsilonNet$^{\uparrow}$ *row, carrying the displayed* ProbeBundle *of centers and the finite coverage ledger for the requested tolerance scale.*
5. *R is the positive rational radius ledger attached to the selected centers and family members.*
6. *L is the finite lower-bound row that compares the selected radii and exports the precision candidate to the modulus consumer.*
7. *U is the* UniformModulus$^{\uparrow}$ *handoff row, available only after F, K, G, N, R, L have displayed the finite selector route.*
8. *H records componentwise* hsame *transport for the family, compact metric, continuous-map, finite-net, radius, lower-bound, uniform-modulus, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact finite-net request through the family modulus rows and the lower-bound selector to the uniform-modulus consumer.*

10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *A is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose family, compact metric, continuous graph, finite net, radius ledger, lower-bound row, uniform-modulus handoff, transport, replay, provenance, and local naming are displayed in these coordinates. It has no coordinate for a host open cover, a global choice function, a function-space topology, an external finite subset, or a selected point outside the ProbeBundle *row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityFiniteNetSelectorUp.EquicontinuityFiniteNetSelectorTasteGate`
`single carrier alignment`

Theorem 2406.2 (Equicontinuity finite-net selector NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ *carrier* $E = (F, K, G, N, R, L, U, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetS}}$ *surface consists exactly of carrier admission for the equicontinuity-family row, compact metric row, continuous-map graph family, finite epsilon-net row, radius ledger, finite lower-bound row, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The selector is valid only when the finite-net centers and the family modulus rows share the same compact source metric before the uniform-modulus row is consumed.*

Proof. Project the accepted carrier of Definition 3358.1. The displayed source rows are F, K, G, N, the selector rows are R, L, U, and the structural rows are H, C, P, A.

The finite centers are read only from N. The family modulus information is read only from F and G, with K supplying the compact metric source used by the coverage ledger.

The lower-bound row L then compares the displayed radius rows R and exports a precision candidate to U. Transport through H and replay through C preserve the same row alignment, while P and A record provenance and local naming. No remaining coordinate can supply a host cover, a choice function, a function-space topology, or an external finite subset. □

Theorem 2406.3 (Equicontinuity finite-net selector uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ *packet reads the selector through*

$$\mathsf{EquicontinuityFamily}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{FiniteEpsilonNet}^{\uparrow},\quad \mathsf{UniformModulus}^{\uparrow}.$$

The exported precision is the finite lower-bound row over the displayed net centers and family radius ledger; it is not an externally chosen modulus and not an open-cover compactness theorem.

Proof. Start with the uniform-modulus request. The carrier exposes the consumer boundary only as U, so the request must first inspect the equicontinuity family F, the compact metric source K, the continuous graph family G, and the finite epsilon-net row N.

The radius ledger R is then read at the displayed net centers and family members. The lower-bound selector L is a finite row over those displayed radii, so the precision candidate handed to U remains inside the packet.

The transport, replay, provenance, and naming rows H, C, P, A preserve this finite route. Thus every consumer sees the selector as a compact finite-net and equicontinuity-family handoff, not as a host-level selector. □

Falsifiable prediction: Every accepted EquicontinuityFiniteNetSelector$^{\uparrow}$ uniform-modulus read must expose an EquicontinuityFamily$^{\uparrow}$ source row, a CompactMetric$^{\uparrow}$ compact source row, ContinuousMap$^{\uparrow}$ graph rows, a FiniteEpsilonNet$^{\uparrow}$ ProbeBundle coverage row, finite radius and lower-bound rows, a UniformModulus$^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: equicontinuityfamily, compactmetric, continuousmap, finite-epsilon-net, uniformmodulus: EquicontinuityFiniteNetSelector$^{\uparrow}$ is not bijective to any listed sibling because it binds the family modulus rows to a compact finite-net lower-bound selector before a UniformModulus consumer can read a tolerance-only precision row.

Closure status for NameCert$_{\text{EquicontinuityFiniteNetSelector}^{\uparrow}}$ **.**

Theory closure: TheoryGate(NameCert$_{\text{EquicontinuityFiniteNetSelector}^{\uparrow}}$, seed)

Scope closed: Definition 3358.1, Definition 3358.2, and Definition 3358.3 seed the finite selector route from equicontinuity-family and compact-net rows to a uniform-modulus consumer.

Formal verification: FormalStatus(NameCert$_{\text{EquicontinuityFiniteNetSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Arzela–Ascoli compactness, function-space topology, host finite subsets, choice of representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source lock, finite-net center visibility, family radius exactness, lower-bound selector stability, uniform-modulus handoff, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.EquicontinuityFiniteNetSelectorUp.

Constructive story: An EquicontinuityFiniteNetSelector$^{\uparrow}$ packet is generated from an EquicontinuityFamily source row, a CompactMetric source row, ContinuousMap graph rows, a FiniteEpsilonNet ProbeBundle row, finite radius and lower-bound rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2407

A Concrete Naming Certificate for $\mathsf{RealBolzano}^{\uparrow}$

Following Bolzano and Bishop–Bridges constructive analysis, the $\mathsf{RealBolzano}^{\uparrow}$ packet records sign-change bracketing as finite Real-and-Completion route data. It sits over the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the interval comparison surfaces of chapter 1591, chapter 411, and chapter 1786. The packet supplies a displayed bracket for later intermediate-value and root-isolation consumers without importing real trichotomy, unrestricted intermediate value, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2407.1 (RealBolzano carrier)**.** *A* $\mathsf{RealBolzano}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *a and b are endpoint real-name rows, each read through the* $\mathsf{Real}^{\uparrow}$ *seal route and exposed only as finite regular windows.*
2. *L and U are the left and right located sign witnesses, recording displayed apartness-from-zero evidence at the inspected endpoint windows.*
3. *S is the sign-change ledger, requiring the two endpoint sign witnesses to be opposed at the requested dyadic precision.*
4. *D is the dyadic bracket ledger, storing the interval endpoints, midpoint requests, tolerance radii, and width comparisons used by the finite bisection route.*
5. *J is the comparison row, carrying the ordered interval and nonescape checks that keep every retained bracket inside the displayed endpoint interval.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row assembled from the retained bracket windows and dyadic tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the endpoint rows, located sign witnesses, bracket ledger, comparison row, and regular handoff have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint, sign, bracket, comparison, regular handoff, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite sign-change request through $a, b, L, U, S, D, J, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealBolzano}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *the closed bounded interval route, the regular Cauchy comparison route,* $\mathsf{RealRootBracket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $a, b, L, U, S, D, J, R, E, H, C, P, N$. It has no coordinate for real trichotomy, unrestricted intermediate value, a selected zero, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2407.2 (RealBolzano NameCert obligations)**.** *For every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *carrier*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for endpoint real rows, located sign witnesses, dyadic bracket ledger, comparison row, regular handoff, real seal boundary, transport, replay, provenance, and local naming;*
2. *endpoint sign exposure, requiring L and U to be read after the endpoint real-name rows a, b and to expose only located apartness data at displayed precision;*
3. *bracket-refinement stability, requiring each retained interval in D to preserve the displayed sign-change ledger S and to decrease the bracket width by dyadic comparison;*
4. *comparison nonescape, requiring J to keep every retained bracket inside the original displayed closed bounded interval;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing, requiring R to be assembled from the retained bracket windows and dyadic tolerance ledger rather than from a selected root;*
6. $\mathsf{Real}^{\uparrow}$ *seal routing, requiring E to remain downstream of a, b, L, U, S, D, J, R;*
7. *componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N for the same rows;*
8. *refusal of real trichotomy, unrestricted intermediate value, selected roots, quotient real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No RealBolzano obligation is stored outside these rows.

Proof. Project the carrier of Definition 3359.1. The endpoint rows a, b are the only real-name sources, and the located sign rows L, U expose sign information only at displayed precision.

The bracket route is finite. The ledger S records the sign-change request, while D stores the dyadic interval, midpoint, tolerance, and width-comparison data used by each retained bracket. The comparison row J keeps each retained interval inside the original closed bounded interval.

The completion-facing rows are downstream of that bracket route. The row R assembles the regular rational handoff from the retained bracket windows, and the seal E is read only after a, b, L, U, S, D, J, R have been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet rows. Since no excluded order, selection, quotient, or completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealBolzanoUp.RealBolzanoTasteGate single carrier alignment`

Theorem 2407.3 (RealBolzano bracket seal route)**.** *Every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *sign-change read passes through endpoint real rows, located sign witnesses, the sign-change ledger, dyadic bracket ledger, comparison row, regular handoff, and terminal Real seal along the finite route*

$$a, b, L, U, S, D, J, R, E, H, C, P, N.$$

The route records bracket-to-seal data for later root-bracketing consumers, but it does not provide unrestricted intermediate value, a selected root, real trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3359.1. The endpoint rows a, b are the only real sources, and the located sign witnesses L, U are read from those endpoint windows at displayed precision.

Use Definition 3359.2. Its obligation rows place the sign-change ledger S before the dyadic bracket ledger D, keep each retained bracket inside the displayed interval through J, and assemble the regular handoff R from retained bracket windows rather than from a selected root.

The same obligation surface requires the Real seal E to remain downstream of a, b, L, U, S, D, J, R, with H, C, P, N transporting and naming only that finite packet. Since no excluded endpoint principle occurs in the carrier, the bracket-to-seal route exports only displayed sign-change and bracketing data. □

Theorem 2407.4 (RealBolzano bracket-refinement stability)**.** *Let*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealBolzano}^{\uparrow}$ *carrier. Replacing one retained dyadic bracket row in* D *by a displayed midpoint refinement that preserves the opposed located sign witnesses and passes the comparison row* J *yields another accepted* $\mathsf{RealBolzano}^{\uparrow}$ *packet with the same downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Project the carrier of Definition 3359.1. The changed coordinate is a single retained bracket row inside D; the endpoint rows a, b, the located sign witnesses L, U, the sign-change ledger S, and the comparison row J remain displayed packet data.

Use Definition 3359.2. Its dependency order keeps S before D and J before R, so the midpoint replacement is admitted only when the opposed located sign witnesses are still present and the refined bracket still passes the displayed comparison row.

Transport the changed bracket row componentwise through H and replay the same finite request through C. The provenance row P and local naming row N continue to name the packet with the refined bracket ledger.

Finally apply Definition 3359.3. The regular handoff R is still assembled from the retained bracket windows, and the terminal Real seal E remains downstream of a, b, L, U, S, D, J, R. No selected root, real trichotomy, quotient equality, unrestricted intermediate value, or ambient completeness coordinate is introduced by the refinement. □

Theorem 2407.5 (RealBolzano regular-Cauchy bracket readback)**.** *Every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *bracket-refinement read reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the endpoint* $\mathsf{Real}^{\uparrow}$ *rows, located sign witnesses, the sign-change ledger, dyadic bracket ledger, comparison nonescape row, and displayed bracket windows. The terminal* $\mathsf{Real}^{\uparrow}$ *seal remains downstream of that regular-Cauchy readback, and the packet exports no unrestricted intermediate value theorem, selected root, trichotomy, quotient equality, or ambient Real completeness.*

Proof. Start with the NameCert obligations of Definition 3359.2. They force endpoint sign exposure through a, b, L, U, record the sign-change ledger in S, and keep every retained bracket inside the displayed interval through the comparison row J.

Apply the bracket seal route of Definition 3359.3. It orders every sign-change read through $a, b, L, U, S, D, J, R, E, H, C, P, N$, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is assembled before the terminal Real seal E can be consumed.

Now use bracket-refinement stability from Definition 3359.4. Replacing a retained bracket row changes only the displayed dyadic bracket data inside D while preserving the same sign witnesses, comparison nonescape row, regular handoff, and downstream seal route.

Thus the regular-Cauchy bracket readback reaches $\mathsf{RegSeqRat}^{\uparrow}$ through the finite endpoint, sign, ledger, comparison, and window rows, with E, H, C, P, N only sealing, transporting, replaying, sourcing, and naming the route. None of the excluded host principles occurs as a carrier coordinate. □

Theorem 2407.6 (RealBolzano sign-change bisection)**.** *Let*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealBolzano}^{\uparrow}$ *carrier. Any displayed midpoint refinement that keeps one opposed located sign pair and satisfies the comparison row J yields a retained bracket row whose* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff remains assembled from D and whose* $\mathsf{Real}^{\uparrow}$ *seal remains downstream of R. The bisection step does not export unrestricted intermediate value, a selected root, trichotomy, quotient real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 3359.1. The midpoint refinement changes only the retained bracket data inside D, while the endpoint rows, located sign witnesses, sign-change ledger, comparison row, regular handoff, and Real seal remain displayed coordinates of the same packet.

Apply Definition 3359.2. The obligations admit the refined bracket only when one opposed located sign pair is still displayed and the comparison row J keeps the retained interval inside the original bracket surface.

Use Definition 3359.4. It transports the refined bracket through H, C, P, N and preserves the downstream handoff and seal route. Then apply Definition 3359.5: the $\mathsf{RegSeqRat}^{\uparrow}$ row R is still assembled from the dyadic bracket ledger D, and the terminal $\mathsf{Real}^{\uparrow}$ seal E remains downstream of R.

Since the carrier has no unrestricted intermediate-value row, selected-root coordinate, trichotomy principle, quotient real equality, or ambient completeness theorem, the theorem records only the constructive finite bisection step visible to later root-bracketing consumers. □

Theorem 2407.7 (RealBolzano located-bracket nonescape)**.** *Every located sign-change bracket exported by an accepted* $\mathsf{RealBolzano}^{\uparrow}$ *carrier*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

remains inside the endpoint rows a, b, located sign witnesses L, U, sign-change ledger S, dyadic bracket ledger D, comparison row J, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N. The packet exports no unrestricted intermediate value theorem, selected root, trichotomy, quotient real equality, or ambient Real completeness.*

Proof. Use the NameCert obligations of Definition 3359.2. Endpoint sign exposure must pass through a, b, L, U, and comparison nonescape is recorded by the row J for every retained dyadic bracket in D.

Apply the bracket seal route of Definition 3359.3. It orders every sign-change read through $a, b, L, U, S, D, J, R, E, H, C, P, N$, so the regular readback and terminal Real seal remain downstream of the located bracket data.

Now use the sign-change bisection theorem Definition 3359.6. A retained midpoint bracket may be exported only when it preserves an opposed located sign pair and satisfies the comparison row J.

Thus each exported located bracket stays inside the displayed endpoint, sign, ledger, comparison, readback, seal, and structural rows. Since the carrier has no unrestricted intermediate-value, selected-root, trichotomy, quotient-equality, or ambient-completeness coordinate, none is exported. □

Falsifiable prediction: Every accepted $\mathsf{RealBolzano}^{\uparrow}$ read factors through two endpoint Real rows, two located sign witnesses, one sign-change ledger, one dyadic bracket ledger, one comparison row, one RegSeqRat handoff, one terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any intermediate-value or root-bracketing consumer may inspect it.

Independence witness: real, regseqrat, dyadicratcore, closedboundedinterval, regular-cauchy-comparison, real-root-bracket: $\mathsf{RealBolzano}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint real rows, located sign witnesses, a sign-change ledger, dyadic bracket refinement, comparison nonescape, regular handoff, real seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3359.1 and Definition 3359.2 seed the constructive sign-change bracketing packet over endpoint Real rows, located sign witnesses, dyadic bracket ledgers, comparison nonescape, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, unrestricted intermediate value, selected roots, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exposure, bracket-refinement stability, comparison nonescape, RegSeqRat handoff routing, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealBolzanoUp.

Constructive story: A $\mathsf{RealBolzano}^{\uparrow}$ packet is generated from endpoint Real rows, located sign witnesses, a sign-change ledger, dyadic bracket refinement, comparison rows, a RegSeqRat handoff, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2408

A Concrete Naming Certificate for $\mathsf{CofinalFilterBase}^{\uparrow}$

Following Kelley and Bourbaki on cofinal filter bases, the $\mathsf{CofinalFilterBase}^{\uparrow}$ packet records the finite comparison interface by which one displayed filter base may replace another in Cauchy and completion consumers. It sits between the displayed base surface of chapter 1601, the Cauchy-filter route of chapter 333, the uniform completion route of chapter 1473, the metric completion route of chapter 1506, and the Cauchy-filter completion route of chapter 1569. The packet names cofinal replacement without quotient filters, selected representatives, selected limits, ultrafilters, Zorn arguments, or ambient completeness.

Definition 2408.1 (CofinalFilterBase carrier)**.** *A* $\mathsf{CofinalFilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, T, \rho, R, F, U, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{FilterBase}^{\uparrow}$ *row whose displayed packets may be used after replacement.*
2. *T is the target* $\mathsf{FilterBase}^{\uparrow}$ *row whose requests are issued by a downstream consumer.*
3. *ρ is the cofinal map row. For every target request it names a source base packet lying inside the requested target packet.*
4. *R is the containment and refinement witness ledger, recording the source-to-target containment reads, common-refinement reads, and replay edges used by ρ.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, recording that the Cauchy-filter consumer reads the same cofinal replacement route.*
6. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer row, exposing the entourage-facing completion route after cofinal replacement.*
7. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row, exposing the metric-completion route after cofinal replacement.*
8. *H records componentwise* hsame *transport for $S, T, \rho, R, F, U, M, C, P, N$.*
9. *C records displayed* Cont *replay from a target base request through ρ, through the source base packet, and through the refinement ledger exposed to downstream consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CofinalFilterBase}^{\uparrow}$*, the five sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient filters, ultrafilters, Zorn arguments, selected representatives, selected limits, countable choice, host filter equality, or ambient completion.

Lean4 ✓ `BEDC.Derived.CofinalFilterBaseUp.CofinalFilterBaseTasteGate single carrier alignment`

Theorem 2408.2 (CofinalFilterBase NameCert obligations)**.** *For every accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *carrier* $B = (S, T, \rho, R, F, U, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$ *surface consists exactly of carrier admission for the source and target* $\mathsf{FilterBase}^{\uparrow}$ *rows, cofinal map row, containment and refinement ledger, CauchyFilter handoff, UniformCompletion consumer row, MetricCompletion consumer row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted target request must be witnessed by a displayed source packet through* ρ *and* R *before any Cauchy or completion consumer can read the replacement.*

Proof. Project the carrier of Definition 3360.1. The rows S and T expose the two displayed filter bases, while ρ and R store the only containment and refinement data connecting them.

The consumer rows are downstream of that comparison. The CauchyFilter row F, UniformCompletion row U, and MetricCompletion row M may read the replacement only after the target request has been replayed through ρ and the ledger R.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since quotient filters, ultrafilters, Zorn arguments, selected representatives, selected limits, countable choice, host filter equality, and ambient completion are absent from the carrier, no additional NameCert obligation is available. □

Theorem 2408.3 (CofinalFilterBase CauchyFilter handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *packet factors through*

$$T \longmapsto \rho \longmapsto S \longmapsto R \longmapsto F, H, C, P, N.$$

Thus a Cauchy-filter request against the target base is answered by a displayed source base packet and its containment ledger before the Cauchy row is read. The handoff supplies no quotient filter, ultrafilter, selected representative, selected limit, or choice schedule.

Proof. Begin with a Cauchy-filter consumer. The request is a target-base read through T, so the carrier forces the request to pass first through the cofinal map row ρ.

Apply Definition 3360.2. The row ρ names a source packet in S, and R records the containment and common-refinement reads that justify the replacement.

The handoff row F is read only after those rows. Transport by H and replay by C preserve the same displayed route, while P and N expose the packet as finite provenance and local naming. No excluded quotient, ultrafilter, selected representative, selected limit, or choice coordinate remains. □

Theorem 2408.4 (CofinalFilterBase completion nonescape)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer citing an accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *packet receives only the displayed cofinal replacement route from target requests to source base packets. The completion-facing read is exhausted by* $S, T, \rho, R, F, U, M, H, C, P, N$ *and cannot escape to quotient filters, selected limits, host filter equality, ultrafilters, Zorn arguments, countable choice, or ambient completeness.*

Proof. Start from a completion-facing request. By Definition 3360.1, the only completion rows are U and M, and the Cauchy-filter completion consumer must pass through the same Cauchy handoff row F.

Apply Definition 3360.3. The target request is first replayed through ρ into the source base row S, then through the containment and refinement ledger R.

The sibling completion chapters named in the opening paragraph therefore receive a displayed cofinal-base route, not a detached completion principle. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming, so no quotient, selected limit, host equality, ultrafilter, Zorn argument, choice schedule, or ambient completeness object is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalFilterBase}^{\uparrow}$ packet is generated from source and target FilterBase rows, a cofinal map from target requests to source base packets, containment and refinement witnesses, CauchyFilter handoff, UniformCompletion and MetricCompletion consumer rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3360.1, Definition 3360.2, Definition 3360.3, and Definition 3360.4 seed the cofinal-base replacement interface between FilterBase, CauchyFilter, and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host filter equality, ultrafilters, Zorn arguments, countable choice, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal map exactness, containment replay, CauchyFilter handoff, UniformCompletion and MetricCompletion consumer nonescape, and ledger-refusal proofs for BEDC.Derived.CofinalFilterBaseUp.

Chapter 2409

A Concrete Naming Certificate for $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$

Following Bishop and Bridges and the constructive bounded-variation treatment, the $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ packet records absolute continuity on a displayed real interval as finite partition control, variation budget, derivative/integral ledger, and real-name handoff data rather than as a host function quotient or a measure-completed derivative theorem. It sits after the regulated-function carrier of chapter 1906, the bounded-variation ledger of chapter 398, the integral surface of chapter 123, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The function-level carrier consumes the epsilon-delta finite partition surface of chapter 6066 before derivative, integral, bounded-variation, or Real-seal ledgers are read. The packet gives Real and Completion consumers a finite route for interval variation requests before any ambient measure algebra, selected derivative, host function equality, or completed L^1 object is available.

Definition 2409.1 (AbsoluteContinuousFunction carrier)**.** *An* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, F, M, P, V, D, L, T, W, Q, R, H, C, S, N)$$

with the following displayed rows:

1. *I is the source interval row, exposing the endpoint real-name windows, orientation data, and compact subinterval request on which the function is inspected.*
2. *F is the displayed real-valued function row, read only through finite input windows and output* $\mathsf{Real}^{\uparrow}$ *seals.*
3. *M is the modulus row assigning, for each dyadic error budget, a finite interval-length budget used by the absolute-continuity test.*
4. *P is the finite disjoint-partition ledger. It lists the inspected interval family, total displayed length, endpoint windows, and the corresponding value-jump windows.*
5. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *budget row asserting that the finite sum of visible jumps along P is controlled by the error budget selected by M.*
6. *D is the derivative ledger row. It records only displayed derivative-window requests and their local variation accounting.*

7. *L is the* $\mathsf{Integral}^{\uparrow}$ *handoff row for the interval derivative or variation-density request consumed by integration-facing clients.*
8. *T is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *approximation row used to expose stepwise endpoint behavior before the gauge row is read.*
9. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the interval, partition, derivative, integral, regulated approximation, and seal observations consumed by the packet.*
10. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for interval lengths, output jumps, regulated approximants, and integral estimates.*
11. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by downstream Real and Completion consumers after the partition, regulated, gauge, regular, and dyadic rows have been displayed.*
12. *H stores componentwise* hsame *transport for interval, function, modulus, partition, variation, derivative, integral, regulated, stream, regular readback, Real seal, replay, provenance, and local naming rows.*
13. *C stores displayed* Cont *replay from an absolute-continuity request through the same finite rows.*
14. *S is the* Pkg *provenance over* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, quotient real functions, measure completion, a selected derivative, countable additivity, ambient variation space completion, or a completed L^1 representative.

```
Lean4✓  BEDC.Derived.AbsolutelyContinuousFunctionUp.AbsolutelyContinuousFunctionTasteGate
single carrier alignment
```

Theorem 2409.2 (AbsoluteContinuousFunction NameCert obligations)**.** *For every accepted* AbsolutelyContinuousF *carrier $A = (I, F, M, P, V, D, L, T, W, Q, R, H, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$ *surface consists exactly of carrier admission; partition-refinement stability through P and W; variation-budget exactness through V; derivative/integral ledger handoff through D, L; regulated approximation through T; StreamName and RegSeqRat readback through W, Q; Real seal nonescape through R; componentwise transport, displayed replay, provenance, and local naming through H, C, S, N.*

Proof. Project the accepted carrier of Definition 3361.1. The visible rows are exactly $I, F, M, P, V, D, L, T, W, Q, R,$ so every local naming obligation must be read from one of those rows.

The source and control obligations are I, F, M, P, V. The interval and function rows determine the inspected endpoint windows, while the modulus row chooses the finite length budget. The partition and variation ledgers then record the only admissible test that small total displayed length forces small total displayed jump.

The analytic handoff obligations are D, L, T, W, Q, R. The derivative ledger records the local window request, the integral row carries the integration-facing handoff, the regulated row exposes stepwise endpoint behavior, and the stream and regular rows provide the finite numerical readbacks before the terminal Real seal is visible.

Finally H, C, S, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host function equality, quotient real functions, measure completion, a selected derivative, countable additivity, ambient variation space completion, or a completed L^1 representative, none of those objects can appear in the local NameCert surface. □

Theorem 2409.3 (AbsoluteContinuousFunction bounded-variation handoff)**.** *Every bounded-variation or fundamental-theorem consumer reads an accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packet only through the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route*

$$I, F, M \longmapsto P, V \longmapsto D, L, T \longmapsto W, Q \longmapsto R, H, C, S, N.$$

Thus a consumer reaches the $\mathsf{Integral}^{\uparrow}$ *row only after the absolute-continuity partition budget and* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger have been displayed, and reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after regulated approximation, StreamName windows, and regular rational readback have also been displayed.*

Proof. Start with the interval, function, and modulus rows. They fix the displayed real interval, the finite function windows, and the dyadic budget used by the absolute-continuity request.

The partition route is next. The row P lists the inspected disjoint intervals and their total displayed length, and V records the corresponding finite jump bound. No gauge-facing read is available before these rows have been consumed.

Only then does the route reach the derivative, integral, and regulated rows. The derivative ledger D records the local request, the $\mathsf{Integral}^{\uparrow}$ row L receives the integration-facing handoff, and the $\mathsf{RegulatedFunction}^{\uparrow}$ row T supplies the stepwise approximation boundary.

The sibling dependency is exact: the bounded-variation row is read through chapter 398, the regulated approximation through chapter 1906, the integral handoff through chapter 123, and the terminal numerical route through chapter 126, chapter 301, and chapter 38.

The terminal read is numerical and sealed. Rows W and Q provide finite observation windows and regular rational comparisons for the displayed estimates, while R, H, C, S, N seal, transport, replay, source, and name the packet. Hence the exported handoff is finite BEDC route data, not a host derivative theorem, measure-completed integral, quotient function equality, or selected L^1 representative. □

Theorem 2409.4 (AbsoluteContinuousFunction partition-modulus stability)**.** *Let* $A = (I, F, M, P, V, D, L, T, W, Q$ *and* $A' = (I', F', M, P', V', D', L', T', W', Q', R', H', C', S', N')$ *be accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packets sharing the same modulus row* M*. If their partition ledgers* P *and* P' *are related componentwise by* hsame *through the transport row* H*, then the variation-budget row* V *exports the same absolute-continuity jump bound before any derivative row, integral handoff, or Real seal is read.*

Proof. Project the two accepted carriers through Definition 3361.1. The shared row M fixes the dyadic length budget used by both absolute-continuity requests, while P and P' list the corresponding displayed disjoint interval families.

Use the componentwise hsame data in H to compare the partition ledgers. Since the comparison is rowwise over the displayed endpoint windows and total-length entries, the same partition budget is presented to the variation row on both sides.

Now read the variation-budget obligation from Definition 3361.2. The row V is upstream of D, L, T, W, Q, R, so the equality of jump bounds is obtained before derivative requests, integral handoffs, regulated approximations, numerical readbacks, or Real seals can contribute extra data. □

Theorem 2409.5 (AbsoluteContinuousFunction Real and Completion consumer boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packet factors through the displayed rows*

$$M \longmapsto P \longmapsto V \longmapsto T \longmapsto W \longmapsto Q \longmapsto R$$

in that order before it exports an endpoint, variation, or integral-facing real read. Such a consumer cannot replace the route by host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, or ambient L^1 completion.

Proof. Begin with the control rows. The modulus row M chooses the finite length budget, the partition ledger P displays the inspected intervals, and the variation row V records the corresponding finite jump bound.

Next compose with the bounded-variation handoff of Definition 3361.3. It places the regulated approximation row T before the StreamName and RegSeqRat reads, so a Real or Completion consumer must pass through the displayed approximation boundary before any terminal numerical seal is available.

Finally read W, Q, R. The finite observation windows, regular rational comparisons, and Real seal are the only consumer-facing numerical route, and the carrier has no coordinate for host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, or ambient L^1 completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3361.1, Definition 3361.2, and Definition 3361.3 seed the finite absolute-continuity route over interval, function, modulus, partition, bounded-variation, derivative, integral, regulated, StreamName, regular, Real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, ambient variation space completion, completed L^1 representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition coverage, variation-budget exactness, derivative and integral handoff, regulated approximation, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteContinuousFunctionUp.

Constructive story: An $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ packet is generated from an interval row, displayed real-valued function row, modulus budget, disjoint-partition ledger, bounded-variation budget, derivative ledger, Integral handoff, RegulatedFunction approximation, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2410

A Concrete Naming Certificate for $\mathsf{EberleinSmulian}^{\uparrow}$

Following Eberlein and Smulian, the $\mathsf{EberleinSmulian}^{\uparrow}$ packet records the weak-compactness-to-sequential-window route as finite NameCert data. It sits beside the sequential compactness carrier of chapter 2372, the compact metric surface of chapter 129, the metric row of chapter 57, and the Cauchy sequence space boundary of chapter 406. The packet is a traditional functional-analysis reconstruction in the BEDC carrier language: it stores finite weak-test ledgers, subsequence-selection rows, compactness handoffs, and terminal naming evidence without asserting unrestricted weak compactness, countable choice, Banach-space dual completeness, or host-level subsequence selection.

Definition 2410.1 (Eberlein–Smulian finite weak-window carrier)**.** *An* $\mathsf{EberleinSmulian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, H, K, M, Q, W, U, L, E, T, C, P, N)$$

with the following displayed rows:

1. *F is the functional-analysis source row naming the bounded linear-test context and the finite weak-test family inspected by the packet.*
2. *H is the Hilbert or Banach-facing source row that supplies the normed carrier, bounded sequence cells, and finite comparison windows.*
3. *K is the compactness handoff row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{SequentialCompact}^{\uparrow}$ *sibling surfaces when a finite compact window is displayed.*
4. *M is the metric row used to compare finite windows and to route Cauchy-style estimates through the sibling metric certificate.*
5. *Q is the Cauchy sequence equivalence boundary: two displayed subsequence windows are compared only through finite regular-window reads and not through quotient sequence equality.*
6. *W is the weak-test ledger listing the finitely many linear tests, tolerances, and observation windows requested by the consumer.*
7. *U is the subsequence-selection row recording the displayed indices retained at each finite weak-test window.*
8. *L is the compactness handoff ledger that repacks the retained subsequence windows as a compact or sequentially compact local packet.*

9. *E is the terminal weak-cluster or convergence seal offered to downstream consumers after the ledgers have been read.*
10. *T stores componentwise* hsame *transports for $F, H, K, M, Q, W, U, L, E$.*
11. *C stores the displayed* Cont *replay routes from weak tests to retained subsequence windows and from compactness handoff to the terminal seal.*
12. *P is the* Pkg *provenance over the source rows, weak-test ledger, subsequence rows, compactness handoff, transport, replay, and local naming.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only through these rows. It has no coordinate for a global weak topology, an unrestricted functional separator, a choice function selecting subsequences from arbitrary sequences, a quotient of sequence tails, or an ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.EberleinSmulianUp.EberleinSmulianTasteGate single carrier alignment`

Theorem 2410.2 (Eberlein–Smulian seed obligations)**.** *For every accepted* $\mathsf{EberleinSmulian}^{\uparrow}$ *carrier $S = (F, H, K, M, Q, W, U, L, E, T, C, P, N)$, the local NameCert surface consists of:*

1. *carrier admission for the finite weak-window packet of Definition 3362.1;*
2. *weak-test exposure through the finite ledger W over the displayed source rows F and H;*
3. *subsequence-window exposure through U, with retained indices visible before any cluster or convergence seal is exported;*
4. *compactness handoff through K and L, routed through the sibling compact and sequential compactness certificates rather than through open-cover compactness;*
5. *metric and Cauchy-window comparison through M and Q before any equality or convergence read is consumed;*
6. *componentwise transport, replay, provenance, and local naming through T, C, P, N.*

These obligations do not include Banach-space weak compactness in full generality, unrestricted linear-functional choice, countable choice, quotient sequence equality, or ambient completeness.

Proof. Project the carrier rows of Definition 3362.1. The rows $F, H, K, M, Q, W, U, L, E, T, C, P, N$ are the only data visible to the local classifier.

The weak-test side is finite. The ledger W lists the requested tests and tolerances, and the source rows F, H bind those tests to the displayed normed or Hilbert-facing carrier before a subsequence row is read.

The subsequence side is also finite. The row U records the retained indices for the inspected windows, while M and Q keep metric comparison and Cauchy-window comparison inside displayed rows.

The compactness handoff is downstream from those ledgers. The rows K and L repack the retained windows through the compact metric and sequential compactness sibling surfaces, and E is exposed only after T, C, P, N transport, replay, source, and name the same packet. Since the excluded principles are absent from the carrier, they cannot discharge a local NameCert obligation. □

Theorem 2410.3 (Eberlein–Smulian nonescape boundary)**.** *Every consumer of an accepted* $\mathsf{EberleinSmulian}^{\uparrow}$ *packet factors through the finite route*

$$W \longmapsto U \longmapsto (K, L) \longmapsto (M, Q) \longmapsto E$$

with componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$ *evidence. Thus a weak-cluster or sequential-compactness read cannot bypass the weak-test ledger, the retained subsequence windows, the compactness handoff, or the metric and Cauchy-window comparison rows.*

The boundary exports no unrestricted weak topology, no selected subsequence for arbitrary input streams, no quotient equality of sequence tails, no open-cover compactness theorem, and no ambient completeness principle.

Proof. Start with an accepted packet S. A consumer reaches the terminal seal E only by following the continuation row C, so its first readable data are the weak-test ledger W and the displayed source rows F, H.

The retained windows are then read through U. The indices are visible finite rows, and the comparison rows M, Q keep the resulting tail and equality checks inside displayed metric and Cauchy-window data.

Next read the compactness handoff. The rows K, L connect the retained windows to the compact metric and sequential compactness sibling certificates before any weak-cluster or convergence seal can be consumed.

Finally, T, C, P, N bind the route to one NameCert packet. Since the carrier contains none of the excluded global objects, the consumer has no escape route around the displayed finite ledgers. □

Closure status for $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EberleinSmulian}^{\uparrow}$ packet is generated from displayed functional-analysis and Hilbert or Banach-facing source rows, finite weak-test ledgers, subsequence-selection windows, compactness handoff rows, metric and Cauchy-window comparisons, componentwise transport, continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$, seed)

Scope closed: Definition 3362.1, Definition 3362.2, and Definition 3362.3 bind the seed finite weak-window carrier and its nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim full Eberlein–Smulian equivalence, unrestricted weak compactness, Banach-space dual completeness, countable choice, quotient sequence equality, host-level subsequence selection, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-test ledger exposure, subsequence-window exposure, compactness handoff, metric and Cauchy-window comparison, transport replay, provenance, and local naming for the same finite packet.

Chapter 2411

A Concrete Naming Certificate for $\mathsf{SorgenfreyLine}^{\uparrow}$

Following Sorgenfrey and Kelley, the $\mathsf{SorgenfreyLine}^{\uparrow}$ packet records the lower-limit line as a finite naming certificate over the already named real and topology surfaces. It is a one-sided neighbourhood interface adjacent to the $\mathsf{Real}^{\uparrow}$ carrier of chapter 38, the local open-neighbourhood discipline of chapter 88, and the stream readback route of chapter 126. The packet also reads lower-limit interval witnesses through the located interval surface of chapter 1709 and regular rational approximation rows, but it does not identify the BEDC carrier with a host topology on the classical real line.

Definition 2411.1 (SorgenfreyLine carrier)**.** *A* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, I, W, A, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *source row for the point being observed.*
2. *I is a* $\mathsf{LocatedRealInterval}^{\uparrow}$ *lower-limit interval row displaying half-open windows $[a, b)$ around the source point through located endpoint, radius, and bracket data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that fixes the finite precision and finite list of lower-limit neighbourhood reads exposed to a consumer.*
4. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row repacking the rational endpoint observations of I and W.*
5. *L is the lower-limit topology ledger saying that neighbourhood access is by displayed half-open interval windows and not by an arbitrary host-open predicate.*
6. *H records componentwise* hsame *transports for R, I, W, A, L, C, P, N.*
7. *C records displayed* Cont *replay from the real source row to the located half-open interval row, from that row to the stream window, and from the window to the regular rational readback.*
8. *P is the* Pkg *provenance over* $\mathsf{SorgenfreyLine}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *row.*

The carrier stores no host topological space, no quotient real presentation, no selected point of a neighbourhood basis, no choice functional, and no completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SorgenfreyLineUp.SorgenfreyLineTasteGate single carrier alignment`

Theorem 2411.2 (SorgenfreyLine NameCert obligations)**.** *For every accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier*

$$S = (R, I, W, A, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *obligations are exactly the finite lower-limit rows displayed by the carrier:*

1. *real-source admission, requiring every point read to pass through the* $\mathsf{Real}^{\uparrow}$ *row* R*;*
2. *half-open interval admission, requiring every neighbourhood read to expose the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row* I *as a lower-limit window* $[a, b)$*;*
3. *stream-window exactness, requiring the finite precision and finite neighbourhood inspection route to be the row* W*;*
4. *regular-rational readback, requiring endpoint and bracket observations to repack through* A *before any downstream real-facing consumer reads them;*
5. *topology-ledger exhaustion, requiring the topology-facing claim to use the ledger* L *rather than an ambient open-set predicate;*
6. *transport, replay, provenance, and local naming stability through* H, C, P, N*.*

These obligations do not assert host topology, quotient real equality, selected neighbourhood choices, countable choice, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier. The only point row is R, the only lower-limit interval row is I, and the only finite observation schedule is the stream window W.

The endpoint and bracket reads of the half-open window are then repacked by the regular rational row A. The ledger L records that topology-facing access is by displayed lower-limit windows, so a consumer cannot replace the route by an arbitrary host-open predicate.

The rows H, C, P, N only transport, replay, source, and name the displayed route. Since the carrier has no coordinate for quotient reals, selected neighbourhood choices, countable choice, trichotomy, host topology, or real completeness, none of those principles can be part of the local NameCert obligations. □

Theorem 2411.3 (SorgenfreyLine lower-limit window nonescape)**.** *Every accepted lower-limit neighbourhood read of an accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier*

$$S = (R, I, W, A, L, H, C, P, N)$$

must pass through the $\mathsf{Real}^{\uparrow}$ *source row* R*, the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *half-open interval row* I*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* A*, the lower-limit topology ledger* L*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *row* N*. The read exports no host topology, quotient real equality, selected neighbourhood basis, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 3363.1. Its only point source is R, its only lower-limit neighbourhood row is the half-open $\mathsf{LocatedRealInterval}^{\uparrow}$ row I, and its only finite inspection schedule is the $\mathsf{StreamName}^{\uparrow}$ row W.

The endpoint observations of the half-open window then repack through A, and the topology-facing claim is exhausted by the ledger L. Apply Theorem 3363.2 to identify the remaining transport, replay, provenance, and local naming rows as H, C, P, N.

Because no other row occurs in the carrier, a lower-limit neighbourhood read cannot escape into host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2411.4 (SorgenfreyLine lower-limit topology handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, *or product-topology consumer of an accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *packet*

$$S = (R, I, W, A, L, H, C, P, N)$$

receives the lower-limit neighbourhood read only through $R, I, W, A, L, H, C, P, N$*. The row* I *provides the located half-open interval witness, and the row* L *provides the lower-limit topology ledger. The downstream* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *product route may read this one-dimensional boundary, but it receives no host topology, quotient real equality, selected neighbourhood basis, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the accepted carrier of Definition 3363.1. The real source is R, the located half-open interval row is I, the finite window and rational readback rows are W and A, and the topology ledger is L.

Apply Definition 3363.2 and Definition 3363.3. Together they identify the structural rows H, C, P, N and force every lower-limit neighbourhood read to stay inside the displayed route.

The $\mathsf{SorgenfreyPlane}^{\uparrow}$ consumer therefore receives the line component only as this lower-limit topology handoff. Since no carrier row stores host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, or ambient real completeness, the product route cannot import those objects from the line certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SorgenfreyLine}^{\uparrow}$ packet is generated from a RealUp source row, a LocatedRealIntervalUp lower-limit half-open interval row, a StreamName finite window, a RegSeqRat readback row, a lower-limit topology ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3363.1 and Definition 3363.2 seed the lower-limit neighbourhood surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, half-open interval admission, stream-window exactness, regular-rational readback, topology-ledger exhaustion, transport stability, replay stability, and non-escape for BEDC.Derived.SorgenfreyLineUp.

Chapter 2412

A Concrete Naming Certificate for $\mathsf{CauchyTailDiagonal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyTailDiagonal}^{\uparrow}$ packet records the finite diagonal-tail handoff by which two regular rational tails are read through one selected diagonal window before a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer inspects a seal. It sits between the diagonal extraction route of chapter 1773, the interleaving surface of chapter 414, the finite tail seal of chapter 559, the finite windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet is a finite handoff surface for Real and completion routes; it does not select an ambient limit, quotient stream, countable-choice schedule, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness.

Definition 2412.1 (CauchyTailDiagonal carrier)**.** *A* $\mathsf{CauchyTailDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, \delta, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *tail rows whose late observations are to be compared.*
2. *δ is the finite diagonal selector row, read through* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *and synchronized with the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *handoff.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the inspected late indices for X, Y, and the diagonal row δ.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the paired tail requests, diagonal slack, and finite comparison budget.*
5. *E is the* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, Y, δ, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the two tail rows, diagonal selector, window family, dyadic ledger, seal row, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from paired tail requests through the selected diagonal window, dyadic tolerance ledger, finite tail seal, and terminal real handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyTailDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}$ $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, Y, \delta, W, D, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable-choice selectors, host equality of real names, trichotomy, or ambient completion.

Lean4 ✓ `BEDC.Derived.CauchyTailDiagonalUp.CauchyTailDiagonalTasteGate single carrier alignment`

Theorem 2412.2 (CauchyTailDiagonal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailDiagonal}^\uparrow$ *carrier $K = (X, Y, \delta, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$ *surface consists exactly of carrier admission for the ten displayed rows, paired* $\mathsf{RegSeqRat}^\uparrow$ *tail admission through X, Y, diagonal selector exactness through δ, finite* $\mathsf{StreamName}^\uparrow$ *window coverage through W,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance accounting through D, finite tail seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3364.1. The only tail sources are X and Y, and the only diagonal selector is the row δ displayed by the packet.

Read the finite windows and tolerance ledger next. The row W names exactly the late observations inspected by the two tails and the diagonal selector, while D records the dyadic tolerances and slack needed by that finite comparison.

The seal row E is downstream of those rows. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the local certificate has no remaining coordinate from which a quotient stream, selected limit, countable-choice selector, host real equality, trichotomy row, or ambient completion principle could be exported. □

Theorem 2412.3 (CauchyTailDiagonal Real handoff)**.** *Every* $\mathsf{Real}^\uparrow$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailDiagonal}^\uparrow$ *carrier must read the finite route*

$$X, Y, \delta, W, D, E, H, C, P, N$$

before any Cauchy-tail conclusion is exported. The handoff first displays the two $\mathsf{RegSeqRat}^\uparrow$ *tails, then the diagonal selector governed by* $\mathsf{RegularCauchyDiagonalExtraction}^\uparrow$ *and* $\mathsf{RegularCauchyInterleaving}^\uparrow$*, then the selected* $\mathsf{StreamName}^\uparrow$ *windows and* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledger, and only then the* $\mathsf{FiniteTailDiagonalSeal}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *seal row.*

Proof. Begin with an accepted carrier and apply Definition 3364.2. The consumer-facing data are exhausted by the displayed packet $X, Y, \delta, W, D, E, H, C, P, N$.

Use the diagonal and interleaving coordinates in order. The selector row δ names the diagonal observations, and the interleaving handoff ensures that both tail rows are read on the same finite window family before the dyadic budget is consumed.

Finally read the finite tail seal. Since E is downstream of the paired tail rows, selected windows, and tolerance ledger, a Real or completion-facing consumer receives a finite diagonal-tail handoff rather than a quotient stream, selected limit, countable-choice schedule, host equality proof, trichotomy decision, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$.

Constructive story: A $\mathsf{CauchyTailDiagonal}^\uparrow$ packet is generated from two RegSeqRat tail rows, one finite diagonal selector, StreamName windows, a DyadicRatCore tolerance ledger, a FiniteTailDiagonalSeal and RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$, seed)

Scope closed: Definition 3364.1, Definition 3364.2, and Definition 3364.3 seed the finite diagonal-tail handoff over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality of real names, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal selector exactness, paired tail window coverage, dyadic tolerance accounting, finite tail seal handoff, Real handoff, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyTailDiagonalUp.

Chapter 2413

A Concrete Naming Certificate for SeparatedLimitUniqueness$^{\uparrow}$

Following Bishop and Bridges constructive metric-space analysis, the SeparatedLimitUniqueness$^{\uparrow}$ packet records the familiar uniqueness of metric limits in a separated metric space as finite BEDC certificate data. It sits over the separated classifier of chapter 1411, the complete-metric limit service of chapter 128, the Cauchy-sequence comparison route of chapter 1412, the terminal real seal of chapter 38, the regular rational readback surface of chapter 301, and the finite observation schedule of chapter 126. The packet names only the displayed zero-distance handoff between two limit rows; it is not a quotient completion equality, a selected representative theorem, countable choice, host equality of sequences, or an ambient completeness principle for Real$^{\uparrow}$.

Definition 2413.1 (SeparatedLimitUniqueness carrier)**.** *A* SeparatedLimitUniqueness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L_0, L_1, Q, M, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *L_0 and L_1 are* CompleteMetric$^{\uparrow}$ *limit rows for the same displayed Cauchy datum, each exposing only the finite limit-service surface available in the complete metric packet.*
2. *Q is the common* CauchySequenceEquivalence$^{\uparrow}$ *comparison route through which the two limit reads are compared before any endpoint classifier is inspected.*
3. *M is the* Met$^{\uparrow}$ *distance-witness row carrying the displayed distance comparison between the two candidate limits.*
4. *Z is the* Real$^{\uparrow}$ *zero-distance row, read only after the distance witness has passed through finite regular rational and stream-window readback.*
5. *S is the* SeparatedMetric$^{\uparrow}$ *classifier handoff saying that the zero-distance row is the only route by which the two displayed limit reads may be identified.*
6. *H records componentwise* hsame *transport for $L_0, L_1, Q, M, Z, S, C, P, N$.*
7. *C records displayed* Cont *replay from the two limit rows through the common Cauchy comparison, the metric distance witness, the zero-distance row, and the separated classifier.*
8. *P is the* Pkg *provenance over* CompleteMetric$^{\uparrow}$, CauchySequenceEquivalence$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L_0, L_1, Q, M, Z, S, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2413.2 (SeparatedLimitUniqueness NameCert obligations)**.** *For every accepted* SeparatedLimitUniquenes *carrier*

$$U = (L_0, L_1, Q, M, Z, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *two-limit admission, requiring both displayed limit reads to pass through the* $\mathsf{CompleteMetric}^{\uparrow}$ *rows L_0 and L_1;*
2. *common Cauchy comparison, requiring the two limit rows to be compared through the single* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route Q;*
3. *distance-witness exactness, requiring the metric comparison to be the displayed* $\mathsf{Met}^{\uparrow}$ *distance row M;*
4. *zero-distance sealing, requiring the zero comparison to be the* $\mathsf{Real}^{\uparrow}$ *row Z reached after the finite readback route;*
5. *separated classifier handoff, requiring identification of the two limit reads to pass through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row S;*
6. *transport, replay, provenance, and local naming stability through H, C, P, N for the same rows;*
7. *refusal of quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No SeparatedLimitUniqueness obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SeparatedLimitUniquenessUp.SeparatedLimitUniquenessTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3365.1. The two candidate limits are visible only as the displayed complete-metric limit rows L_0 and L_1, so the certificate has no separate host endpoint from which to read equality.

The common comparison route is Q. It aligns the two limit reads as Cauchy data before the distance witness is inspected, keeping the route inside the finite $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ surface.

The metric comparison then passes through M and reaches the zero-distance row Z. The row Z is a terminal $\mathsf{Real}^{\uparrow}$ seal over the finite readback route, not a detached assertion of equality in a host real line.

Finally the separated classifier row S is consumed. Transport through H, replay through C, provenance through P, and local naming through N preserve the same displayed path, so the listed rows exhaust the NameCert obligations. □

Theorem 2413.3 (SeparatedLimitUniqueness zero-distance handoff)**.** *Every consumer that identifies two accepted* $\mathsf{CompleteMetric}^{\uparrow}$ *limit reads through an accepted* $\mathsf{SeparatedLimitUniqueness}^{\uparrow}$ *packet must factor the identification through*

$$L_0, L_1, Q, M, Z, S, H, C, P, N.$$

The handoff reads the common Cauchy comparison Q, the $\mathsf{Met}^{\uparrow}$ *distance witness M, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, and then the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier S before any named limit uniqueness claim is visible. It exports no quotient sequence equality, selected representative, host equality of metric points, countable-choice schedule, trichotomy split, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier U. A uniqueness-facing consumer first sees only the two complete-metric rows L_0 and L_1 and the shared comparison route Q.

The metric witness M is downstream of that route. Thus the consumer cannot compare the endpoints by host equality or by selecting representatives before the finite Cauchy comparison has been displayed.

The zero-distance row Z is then read as a $\mathsf{Real}^{\uparrow}$ seal, and S translates that displayed zero-distance evidence into the separated classifier handoff. The structural rows H, C, P, N transport, replay, source, and name the same route.

Therefore every limit-uniqueness read is the stated finite zero-distance handoff, with no additional quotient, choice, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3365.1, Definition 3365.2, and Definition 3365.3 seed the finite separated-limit uniqueness packet over complete-metric limit rows, Cauchy-sequence comparison, metric distance witnesses, Real zero-distance rows, separated classifiers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common Cauchy comparison, distance-witness exactness, zero-distance sealing, separated classifier handoff, transport stability, replay stability, and non-escape for BEDC.Derived.SeparatedLimitUniquenessUp.

Constructive story: A $\mathsf{SeparatedLimitUniqueness}^{\uparrow}$ packet is generated from two CompleteMetricUp limit rows, a common CauchySequenceEquivalenceUp comparison route, a MetricUp distance witness, a RealUp zero-distance row, a SeparatedMetricUp classifier handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2414

A Concrete Naming Certificate for $\mathsf{CertificateMachineBranch}^{\uparrow}$

The $\mathsf{CertificateMachineBranch}^{\uparrow}$ packet realizes the branch-separation side of subsection 9462.28.23 as finite BEDC certificate data. It sits downstream of the finite certificate machine surface in chapter 3355, but it records a narrower packet: a kernel spine, an admission gate, a derivation-strength record, a cannot-claim ledger boundary, and explicit branch-separation rows for allegory and biological bridge candidates. The packet is not a plain Turing-machine identity claim, not a biological semantics claim, not a semantic truth oracle, and not an unbounded execution trace.

Definition 2414.1 (CertificateMachineBranch carrier)**.** *A* $\mathsf{CertificateMachineBranch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, A, D, L, G, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite kernel-spine row, naming the constructors, candidate rows, acceptance stamps, and refusal slots inherited from the BEDC certificate kernel.*
2. *A is the admission-gate row. It records the finite evidence by which a branch candidate is accepted, refused, or kept outside the public certificate surface.*
3. *D is the derivation-strength record, naming the declared strength at which the accepted branch may be read.*
4. *L is the cannot-claim boundary ledger, excluding semantic truth, plain Turing-machine identity, unbounded execution, biological identity, and external proof authority.*
5. *G is the allegory-branch row, recording candidate symbolic patterns only as pre-admission branch material.*
6. *R is the biological bridge row, recording DNA-facing or external natural branches only as bridge candidates, not as kernel primitives.*
7. *S is the branch-separation row saying that allegory, DNA, and certificate-kernel reads remain distinct unless the admission gate names a finite accepted route.*
8. *H records componentwise* hsame *transport for $K, A, D, L, G, R, S, C, P, N$.*
9. *C records displayed* Cont *replay from kernel spine through admission gate, derivation-strength record, cannot-claim boundary, and branch-separation rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CertificateMachineBranch}^{\uparrow}$, $\mathsf{CertificateMachine}^{\uparrow}$, *finite certificate kernels, admission gates, derivation-strength records, cannot-claim ledgers, allegory branches, biological bridge branches,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$ *row.*

The classifier compares accepted branch packets only through $K, A, D, L, G, R, S, H, C, P, N$. It has no coordinate for host machine equality, semantic truth, biological identity, unbounded execution logs, unlicensed allegory admission, or external proof authority.

Lean4 ✓ `BEDC.Derived.CertificateMachineBranchUp.TasteGate.CertificateMachineBranchTasteGate single carrier alignment`

Theorem 2414.2 (CertificateMachineBranch NameCert obligations)**.** *For every accepted* CertificateMachineBranch *carrier*

$$B = (K, A, D, L, G, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *kernel-spine admission, requiring every branch candidate, constructor, acceptance stamp, and refusal slot to pass through K;*
2. *admission-gate exactness, requiring every accepted branch read to cite the finite row A;*
3. *derivation-strength exactness, requiring the public strength of the branch read to be the displayed row D;*
4. *cannot-claim boundary non-escape, requiring L to exclude semantic truth, plain Turing-machine identity, unbounded execution, biological identity, and external proof authority;*
5. *allegory and biological branch separation, requiring G, R, S to keep symbolic-pattern and DNA-facing candidates distinct from kernel primitives unless admitted by A;*
6. *transport, continuation, provenance, and local naming stability through H, C, P, N.*

Lean TasteGate formalization must include explicit $\mathsf{Nontrivial}(\mathsf{CertificateMachineBranch}^{\uparrow})$ *and* $\mathsf{FieldFaithful}(\mathsf{Certifica}$ *instances before the packet may be consumed as a nondegenerate field-facing branch certificate.*

Proof. Project the carrier of Definition 3366.1. The branch candidate first meets the kernel-spine row K and the admission-gate row A, so no public branch read can arise from an unlicensed symbolic or biological source.

The derivation-strength row D and cannot-claim ledger L then determine the public reading strength and the forbidden exits. These rows keep certificate acceptance separate from semantic truth, host machine equality, unbounded execution, and biological identity.

The rows G, R, S provide the branch inventory. They distinguish allegory candidates from DNA-facing bridge candidates and from kernel primitives, and they expose no equality between those sources unless the admission gate records the finite route.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 2414.3 (CertificateMachineBranch separation non-escape)**.** *Every accepted public read of a* $\mathsf{CertificateMachineBranch}^{\uparrow}$ *packet factors through*

$$K, A, D, L, G, R, S, H, C, P, N.$$

The read first traverses the finite kernel spine K, the admission gate A, the derivation-strength record D, and the cannot-claim boundary L; only then may it inspect the allegory row G, biological bridge row R, and branch-separation row S. It exports no plain Turing-machine identity, biological semantics, host execution trace, unlicensed branch acceptance, or semantic truth predicate.

Proof. Begin with the accepted carrier B. The only kernel-facing row is K, and the only acceptance row is A, so a public read cannot skip directly to an allegory or DNA-facing branch.

The row D fixes the derivation strength carried by the accepted branch. The row L remains visible during the replay, so the consumer receives the stated boundary rather than a stronger semantic or biological claim.

The rows G, R, S are read after that boundary. They record branch inventory and separation, not an identification of allegory, biology, and kernel primitives.

The remaining rows H, C, P, N preserve the same finite route by transport, continuation replay, provenance, and local naming. Hence every accepted read stays inside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{CertificateMachineBranch}^{\uparrow}$ branch read factors through a finite kernel spine, an admission gate, a derivation-strength record, a cannot-claim boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before any branch-separation row may be consumed.

Independence witness: finitewitnessroute, gapclosureboundary, finitetraceevaluator, kernelacceptancetrace: $\mathsf{CertificateMachineBranch}^{\uparrow}$ is not bijective to any listed sibling because none contains the joint admission-gate plus allegory/DNA branch-separation inventory together with derivation strength, cannot-claim boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateMachineBranch}^{\uparrow}$ packet is generated from a finite kernel spine, an admission gate, a derivation-strength record, a cannot-claim boundary, allegory and biological bridge branch rows, branch-separation evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3366.1, Definition 3366.2, and Definition 3366.3 seed the finite branch packet over certificate-machine kernels, admission gates, derivation-strength records, cannot-claim ledgers, allegory rows, biological bridge rows, branch separation, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim plain Turing-machine identity, semantic truth, biological semantics, host execution traces, unlicensed branch acceptance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked kernel-spine admission, admission-gate exactness, derivation-strength exactness, cannot-claim boundary non-escape, branch-separation exactness, transport stability, replay stability, and the $\mathsf{Nontrivial}(\mathsf{CertificateMachineBranch}^{\uparrow})$ and $\mathsf{FieldFaithful}(\mathsf{CertificateMachineBranch}^{\uparrow})$ Lean instances for BEDC.Derived.CertificateMac

Chapter 2415

A Concrete Naming Certificate for $\mathsf{SylvesterInertia}^{\uparrow}$

Following Sylvester's law of inertia and finite-dimensional quadratic-form theory, the $\mathsf{SylvesterInertia}^{\uparrow}$ packet records congruence-invariant positive, negative, and null sign-count rows. It reads the field source of chapter 45, the matrix presentation of chapter 49, the bilinear form row of chapter 146, and the inner-product substrate of chapter 86. The packet names finite inertia data as a displayed NameCert carrier; it does not assert analytic diagonalization, spectral theory, choice-selected bases, or completeness.

Definition 2415.1 (Sylvester inertia carrier)**.** *A* $\mathsf{SylvesterInertia}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, M, B, Q, D, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Field}^{\uparrow}$ *source row over which the scalar signs and matrix entries are read.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the finite square coordinate table used by the form.*
3. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *row naming the symmetric bilinear form carried by the matrix presentation.*
4. *Q is the quadratic-form row obtained by reading the diagonal values of B on displayed coordinate vectors.*
5. *D is the finite diagonalization-witness row, recording the congruence table that exposes diagonal form entries without choosing an ambient basis outside the packet.*
6. *S is the sign-count ledger, listing the displayed positive, negative, and null diagonal entries as the inertia triple.*
7. *H records componentwise* hsame *transport for the field source, matrix presentation, bilinear form, quadratic form, diagonal witness, sign-count ledger, continuation replay, provenance, and local naming.*
8. *C records* Cont *replay from a finite congruence request through F, M, B, Q, D into the sign-count ledger S.*
9. *P is the* Pkg *provenance over* $\mathsf{SylvesterInertia}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$ *row naming the displayed inertia packet.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for a spectral theorem, analytic diagonalization process, choice-selected basis, quotient form equality, or completeness principle.

Lean4 ✓ `BEDC.Derived.SylvesterInertiaUp.SylvesterInertiaTasteGate single carrier alignment`

Theorem 2415.2 (Sylvester inertia NameCert obligations)**.** *For every accepted* $\mathsf{SylvesterInertia}^{\uparrow}$ *carrier* $K = (F, M, B, Q, D, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the field source, matrix presentation, bilinear form, quadratic form, diagonal witness, sign-count ledger, transport, continuation replay, provenance, and local naming row;*
2. *form coherence, requiring* M*,* B*, and* Q *to describe the same displayed finite quadratic-form packet over* F*;*
3. *diagonal witness exactness, requiring* D *to expose only congruence steps whose source and target rows are recorded in* M, B, Q*;*
4. *inertia ledger readback, requiring* S *to count exactly the displayed positive, negative, and null diagonal entries exposed by* D*;*
5. *non-escape, excluding spectral decomposition, analytic diagonalization, basis choice outside the packet, quotient form equality, and completeness arguments.*

No local NameCert obligation is stored outside those five rows.

Proof. Project the carrier of Definition 3367.1. Carrier admission exposes exactly $F, M, B, Q, D, S, H, C, P, N$.

The form-coherence row binds the matrix table M, bilinear row B, and quadratic row Q over the same field source F. The diagonal witness row D is then readable only as a finite congruence ledger for that displayed form data.

The sign-count ledger S reads the diagonal entries exposed by D and separates them into positive, negative, and null cells. Finally H, C, P, N transport, replay, source, and name only that finite packet, so the excluded analytic and choice-based coordinates have no NameCert row. □

Theorem 2415.3 (Sylvester inertia congruence invariance)**.** *If two accepted* $\mathsf{SylvesterInertia}^{\uparrow}$ *packets are related by a displayed congruence replay through their rows* $F, M, B, Q, D, H, C, P, N$*, then they preserve the displayed sign-count ledger* S*. In particular, congruent accepted packets export the same positive, negative, and null inertia counts through* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$*.*

Proof. Begin with two accepted carriers and a displayed congruence replay between them. By Definition 3367.2, the replay must pass through the field source, matrix presentation, bilinear form, quadratic form, and diagonal witness rows before the sign-count ledger is read.

The diagonal witness exactness row forces the congruence table to expose the same diagonal-form cells on both sides of the accepted comparison. The inertia ledger readback row then counts those displayed cells as positive, negative, or null inside S.

Because the carrier has no row for an external basis selector, analytic diagonalizer, quotient equality, or spectral argument, the exported comparison is exactly the componentwise ledger comparison. Thus the positive, negative, and null counts are preserved. □

Closure status for $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3367.1, Definition 3367.2, and Definition 3367.3 seed the finite inertia carrier, five-row NameCert obligation surface, and congruence-invariant sign-count ledger.

Formal verification: FormalStatus(NameCert$_{\mathsf{SylvesterInertia}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the spectral theorem, analytic diagonalization, choice-selected bases, quotient form equality, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, form coherence, diagonal witness exactness, inertia ledger readback, and non-escape for the same finite Sylvester inertia packet.

Constructive story: A SylvesterInertia$^{\uparrow}$ packet is generated from a field source, matrix presentation, bilinear form, quadratic-form row, diagonalization witness, positive-negative-null sign-count ledger, componentwise transport, continuation replay, provenance, and local NameCert row.

Chapter 2416

A Concrete Naming Certificate for $\mathsf{FundamentalGroup}^{\uparrow}$

Following Hatcher and Spanier on the fundamental group, the $\mathsf{FundamentalGroup}^{\uparrow}$ packet names the loop-composition group at a displayed basepoint by finite BEDC rows. It sits beside the homotopy surface of chapter 156, the algebraic group surface of chapter 41, and the path-lifting surface of chapter 2275. The certificate records based loops, loop homotopies, concatenation, inverse, identity, and covering-lift handoff ledgers; it does not import quotient classes of loops or a host topological theorem.

Definition 2416.1 ($\mathsf{FundamentalGroup}^{\uparrow}$ finite carrier)**.** *A* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient topological source row and x_0 is the displayed basepoint row.*
2. *L is the finite based-loop row family. Each loop starts and ends at x_0 and is read through the path-facing part of* $\mathsf{Homotopy}^{\uparrow}$*.*
3. *Q is the loop-homotopy row family. It compares based loops by endpoint-fixed* $\mathsf{Homotopy}^{\uparrow}$ *certificates, not by an ambient quotient of maps.*
4. *M is the concatenation ledger. It records finite composable loop pairs, their concatenated loop row, and the endpoint-gluing readback.*
5. *I is the inverse ledger. It records the reversed loop row and the homotopy rows witnessing the two cancellation routes back to the identity loop row.*
6. *E is the identity loop row at x_0 and its left and right unit ledgers for M.*
7. *U is the covering-lift handoff row. It exposes exactly the based-loop data consumable by chapter 2275: base path, initial lift point request, lifted segment route, sheet classifier, and endpoint seal.*
8. *H records componentwise* hsame *transport for $X, x_0, L, Q, M, I, E, U, C, P, N$.*
9. *C records displayed* Cont *replay for loop readback, loop-homotopy readback, concatenation, inverse, identity, and covering-lift handoff rows.*

10. *P is the* Pkg *provenance over* $\mathsf{FundamentalGroup}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{CoveringSpacePathLifting}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. Thus the group object is carried as a finite based-loop and homotopy ledger over BHist, *not as a quotient set of host loops.*

Lean4 ✓ `BEDC.Derived.FundamentalGroupUp.FundamentalGroupTasteGate single carrier alignment`

Theorem 2416.2 ($\mathsf{FundamentalGroup}^{\uparrow}$ NameCert obligations)**.** *For every accepted* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier*

$$\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$ *surface consists exactly of carrier admission, basepoint admission, based-loop admission through* $\mathsf{Homotopy}^{\uparrow}$*, endpoint-fixed loop-homotopy comparison, concatenation ledger exactness, inverse ledger exactness, identity-loop exactness, covering-lift handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{Group}^{\uparrow}$ *consumer can inspect $L, Q, M, I, E, H, C, P, N$ as the finite algebraic surface, while every covering consumer can inspect U as the finite handoff route; neither consumer receives quotient classes of loops.*

Proof. Carrier admission is projection from Definition 3368.1. The rows $X, x_0, L, Q, M, I, E, U, H, C, P, N$ already list the base space, basepoint, loops, homotopies, operations, transport, replay, provenance, and local naming data.

The group obligation is read from M, I, E over the classifier Q. Concatenation uses the finite endpoint-gluing row of M, inverse uses the reversed-loop row and cancellation ledgers of I, and the unit laws use E. Each comparison is routed through endpoint-fixed $\mathsf{Homotopy}^{\uparrow}$ rows, so the algebraic equality surface is the finite loop-homotopy classifier.

The covering obligation is independent of any global universal cover. The handoff row U exposes a based loop as the base path required by chapter 2275, together with exactly the initial lift request, lifted segment route, sheet classifier, and endpoint seal that a path-lifting packet can read.

Since H, C, P, N only transport, replay, source, and name the displayed rows, there is no remaining coordinate through which a downstream consumer could demand a host loop quotient, a completed path space, or an uncarried fundamental-group representation. □

Theorem 2416.3 ($\mathsf{FundamentalGroup}^{\uparrow}$ covering path route)**.** *Let* $\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N)$ *be an accepted* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier. Any* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ *consumer of a based loop from* Π *factors through the handoff row U, the original based-loop row in L, the endpoint-fixed loop-homotopy classifier Q, the componentwise transport H, the displayed replay C, the provenance P, and the local name N. The consumer may observe the lifted endpoint seal and sheet-classifier route supplied by the covering packet, but it cannot replace the loop row by a quotient class or a fundamental-group representation.*

Proof. Choose an accepted based loop row from L. Its initial and final endpoint rows are both the displayed basepoint x_0, and its admissibility is already read through the $\mathsf{Homotopy}^{\uparrow}$ path surface.

Project the handoff row U. By definition, U contains the base-path presentation, the initial lift request, and the finite route slots consumed by $\mathsf{CoveringSpacePathLifting}^{\uparrow}$. The path-lifting packet therefore receives the loop only as a carried base path with visible endpoint data.

Transport along H and replay along C. These rows preserve the same based-loop row, loop-homotopy comparison, provenance, and local name, so any covering endpoint seal remains tied to the finite packet Π.

The route ends at the covering packet's own sheet classifier and endpoint seal. No step introduces a quotient of loops, a universal-cover action, or a host representation theorem; all handoff information is finite BHist data with NameCert provenance. □

Closure status for $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3368.1, Definition 3368.2, and Definition 3368.3 seed the finite basepoint, loop, loop-homotopy, concatenation, inverse, identity, covering-handoff, transport, replay, provenance, and local naming surface for the fundamental group.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FundamentalGroupUp.FundamentalGroupTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close van Kampen theorems, monodromy representations, universal covers, quotient path spaces, host loop equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, based-loop admission, loop-homotopy comparison, operation ledgers, covering handoff, transport, replay, provenance, and local naming for BEDC.Derived.FundamentalGroupUp.

Constructive story: A $\mathsf{FundamentalGroup}^{\uparrow}$ packet is generated from a displayed basepoint, finite based-loop rows, endpoint-fixed loop-homotopy rows, concatenation, inverse, identity, covering-lift handoff, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 2417

A Concrete Naming Certificate for $\mathsf{PartitionOfUnity}^{\uparrow}$

Following Kelley and Bourbaki on partitions of unity, the $\mathsf{PartitionOfUnity}^{\uparrow}$ packet records the finite BEDC surface by which a displayed locally finite open cover receives compatible bump rows and finite-support weight rows. It sits over the topological source of chapter 88, the support-window discipline of chapter 1726, the refinement ledger of chapter 1963, the chart-facing consumer surface of chapter 96, and the separation-function route of **??**. The packet names only displayed finite rows for local-to-global gluing; it does not select arbitrary host functions, assert compactness of open covers, or identify analytic functions extensionally.

Definition 2417.1 (Partition of unity carrier)**.** *A* $\mathsf{PartitionOfUnity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, L, Q, R, W, F, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the open-set operations and point-neighborhood reads on which all cover and support data are interpreted.*
2. *L is the* $\mathsf{LocallyCompact}^{\uparrow}$ *support-window row, recording the compactly supported local windows used by each bump row without exporting an open-cover compactness principle.*
3. *Q is the* $\mathsf{Paracompact}^{\uparrow}$ *locally finite cover ledger, including the displayed refinement cells, finite neighborhood witnesses, and assignment back to the original open-cover request.*
4. *R is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *bump-row ledger, recording the separation-compatible construction of each nonnegative bump row inside the support window supplied by L and the locally finite cell supplied by Q.*
5. *W is the finite-support weight row, listing the finitely many bump rows visible at each displayed point-neighborhood read.*
6. *F is the displayed sum-to-one row, recording that the finitely supported weight rows add to the unit row on the covered region.*
7. *S is the manifold-facing handoff row, exposing exactly the partition rows consumable by* $\mathsf{Manifold}^{\uparrow}$ *chart gluing and by local-to-global analysis packets.*
8. *H records componentwise* hsame *transport for the topology source, support windows, locally finite cover ledger, bump rows, weight rows, sum row, manifold handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a partition request through $T, L, Q, R, W, F, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-selected function family, arbitrary subcover selection, open-cover compactness theorem, unlogged chart choice, or extensional equality of analytic functions.

Lean4 ✓ `BEDC.Derived.PartitionOfUnityUp.PartitionOfUnityTasteGate single carrier alignment`

Theorem 2417.2 (Partition of unity NameCert obligations)**.** *For every accepted* $\mathsf{PartitionOfUnity}^{\uparrow}$ *carrier*

$$U = (T, L, Q, R, W, F, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, support windows, locally finite cover ledger, bump-row ledger, finite-support weights, sum-to-one row, manifold handoff, transport, replay, provenance, and local naming;*
2. *support-window exactness, requiring every bump row to read its support through the displayed* $\mathsf{LocallyCompact}^{\uparrow}$ *row L and not through a host compactness claim;*
3. *locally finite cover exactness, requiring every weight row in W to be indexed only by the refinement cells and finite neighborhood witnesses displayed by Q;*
4. *Urysohn-compatible bump exactness, requiring every bump row in R to pass through the* $\mathsf{UrysohnLemma}^{\uparrow}$ *separation-function route before it can contribute to a weight row;*
5. *finite-support and sum exactness, requiring W and F to display the finite set of nonzero weights at each point-neighborhood read and the corresponding sum-to-one row;*
6. *manifold handoff exactness, requiring S to export only the rows $T, L, Q, R, W, F, H, C, P, N$ needed by* $\mathsf{Manifold}^{\uparrow}$ *chart gluing and local-to-global consumers;*
7. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve exactly the listed rows.*

No local partition-of-unity obligation is stored outside these seven rows.

Proof. Carrier admission is projection from Definition 3369.1. The accepted tuple displays exactly the topology source, local compact support windows, paracompact locally finite ledger, Urysohn-compatible bump rows, finite-support weights, sum row, manifold handoff, transport, replay, provenance, and local naming.

Support-window exactness is read from L, while locally finite cover exactness is read from Q. The row L supplies only the compactly supported local windows already displayed in the packet, and Q supplies only the refinement cells, assignments, and finite neighborhood witnesses of the paracompact ledger.

The bump-row gate is R. Each bump row is admitted through the $\mathsf{UrysohnLemma}^{\uparrow}$ separation-function route, so a weight row cannot be introduced by selecting an arbitrary host function.

The weight and sum gates are W and F. The row W lists the finite set of nonzero weights visible at each displayed neighborhood read, and F records the displayed sum-to-one row over that finite support.

Finally, S, H, C, P, N only hand off, transport, replay, source, and name the same finite packet. Thus a manifold or local-to-global consumer can read the partition surface without receiving a host-selected function family, arbitrary subcover, open-cover compactness theorem, unlogged chart choice, or extensional equality of analytic functions. □

Theorem 2417.3 (Partition of unity locally finite cover route)**.** *Let*

$$U = (T, L, Q, R, W, F, S, H, C, P, N)$$

be an accepted $\mathsf{PartitionOfUnity}^{\uparrow}$ *carrier. Every weight row in* W *and every sum-to-one row in* F *must first read the* $\mathsf{Paracompact}^{\uparrow}$ *locally finite cover ledger* Q *before reaching the* $\mathsf{UrysohnLemma}^{\uparrow}$ *bump rows* R*, the finite-support surface* W*, the sum row* F*, and the* $\mathsf{Manifold}^{\uparrow}$ *handoff* S*. The route does not select arbitrary subcovers or import open-cover compactness.*

Proof. Project the carrier of Definition 3369.1. The row Q is the only locally finite cover ledger in the packet, and it records the displayed refinement cells, finite neighbourhood witnesses, and assignment back to the original cover request.

Project the analytic rows R, W, F, S. The bump row R reads its separation and support data after the locally finite cell is known, the finite-support row W lists only the weights visible over that displayed locally finite neighbourhood, and the sum-to-one row F adds exactly those listed weights.

The handoff row S exports this same route to $\mathsf{Manifold}^{\uparrow}$ consumers. By Definition 3369.2, no local partition-of-unity obligation is stored outside the carrier rows, so neither arbitrary subcover selection nor open-cover compactness can replace Q. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3369.1 and Definition 3369.2 seed the finite topology, local compactness, paracompact refinement, Urysohn bump, weight, sum-to-one, manifold handoff, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-selected function families, arbitrary subcover selection, open-cover compactness, analytic function extensionality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-window exactness, locally finite cover exactness, Urysohn-compatible bump exactness, finite-support sum exactness, manifold handoff exactness, transport stability, replay stability, and non-escape proofs for BEDC.Derived.PartitionOfUnityUp.

Constructive story: A $\mathsf{PartitionOfUnity}^{\uparrow}$ packet is generated from a $\mathsf{Topology}^{\uparrow}$ source row, $\mathsf{LocallyCompact}^{\uparrow}$ support windows, a $\mathsf{Paracompact}^{\uparrow}$ locally finite cover ledger, $\mathsf{UrysohnLemma}^{\uparrow}$ bump rows, finite-support weights, a displayed sum-to-one row, a $\mathsf{Manifold}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2418

A Concrete Naming Certificate for $\mathsf{Bornology}^{\uparrow}$

Following Bourbaki and Kelley on bornologies, the $\mathsf{Bornology}^{\uparrow}$ packet records the finite bounded-family interface that sits between the bounded-subset certificate of chapter 1757, the directed base surface of chapter 1601, the uniform entourage surface of chapter 339, and the Cauchy-family specialization of chapter 1896. It names bounded families by displayed hereditary and finite-union rows before any completion, uniform, or Real-facing consumer can ask for a selected bound, a quotient family, countable choice, host suprema, or ambient completeness.

Definition 2418.1 (Bornology carrier)**.** *A* $\mathsf{Bornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, S, U, E, D, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded-family source row. It lists the finite family requests that may be read as bounded data.*
2. *S is the subset-heredity ledger, recording that an accepted subfamily request is read only through a displayed parent family already named by F.*
3. *U is the finite-union ledger, recording binary and finite union requests as displayed rows over previously accepted bounded families.*
4. *E is the singleton and source-exposure row, naming the basic bounded packets available before hereditary or union closure is inspected.*
5. *D is the consumer handoff row for* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *readers, and for the* $\mathsf{CauchyBornology}^{\uparrow}$ *specialization of the same bounded-family surface.*
6. *H records componentwise* hsame *transport for F, S, U, E, D, C, P, N.*
7. *C records displayed* Cont *replay from bounded-family admission through subset heredity, finite union, source exposure, consumer handoff, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{Bornology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyTasteGate single carrier alignment`

Theorem 2418.2 (Bornology NameCert obligations)**.** *For every accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier* $B = (F, S, U, E, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-family source exposure, subset-heredity ledger visibility, finite-union ledger visibility, singleton and source exposure,* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *consumer handoff,* $\mathsf{CauchyBornology}^{\uparrow}$ *specialization handoff, componentwise transport, displayed continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyNameCertObligations`

Proof. Carrier admission projects the displayed packet. The source row F and singleton/source row E expose the basic bounded-family entries before any closure rule is read.

Subset heredity is confined to S, and finite union is confined to U. These ledgers are rows of the same packet, so a consumer cannot replace them by a host powerset, a selected bound, or a quotient family.

The handoff row D then presents the same bounded-family surface to $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, and $\mathsf{CauchyBornology}^{\uparrow}$ consumers. The structural rows H, C, P, N transport, replay, source, and name only the displayed packet. □

Theorem 2418.3 (Bornology bounded-family union stability)**.** *Let* $B = (F, S, U, E, D, H, C, P, N)$ *be an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier. Every accepted finite union of displayed bounded-family reads is certified by the finite-union ledger* U *after the source row* F *and singleton/source row* E *have been exposed, and every accepted subfamily of such a union is routed back through the subset-heredity ledger* S*. The route exports no host supremum, selected bound, quotient family, countable-choice selector, open-cover compactness principle, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyBoundedFamilyUnionStability`

Proof. Read the carrier rows. Source and singleton exposure are the only entry points for basic bounded-family data, so a union request must first cite displayed rows from F and E.

The finite-union ledger U records the admitted union row. A later subfamily request cannot inspect that union through a host family predicate; it must pass through S, whose parent row is already displayed inside the packet.

Transport and replay through H and C preserve the same ledgers, while P and N package only the named dependencies and local certificate. Therefore the union-stability route remains finite and contains none of the excluded host principles. □

Theorem 2418.4 (Bornology CauchyBornology handoff)**.** *Every* $\mathsf{CauchyBornology}^{\uparrow}$ *consumer of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier reads the general bounded-family surface through*

$$F, E \longmapsto S, U \longmapsto D \longmapsto H, C, P, N.$$

Thus the Cauchy specialization receives bounded-family source rows, singleton exposure, subset heredity, finite-union stability, consumer handoff, transport, replay, provenance, and local naming before it may open rate budgets, regular rational readback, Real seals, or completion-facing data.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyCauchyBornologyHandoff`

Proof. Start with a $\mathsf{CauchyBornology}^{\uparrow}$ consumer. The only general bornology data available to it are the displayed source and singleton rows F, E.

The closure properties are read next. Subset heredity is supplied by S, and finite union is supplied by U, so the consumer receives the ordinary bornological closure rows before any Cauchy-specific budget or Real-facing row can be inspected.

The handoff is then the row D, preserved by H and replayed by C. Since P and N name only the general bounded-family packet and its Cauchy specialization, the route adds no selected bound, quotient family, countable-choice selector, host supremum, or ambient completeness premise. □

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyCarrier cauchy bornology handoff`

Theorem 2418.5 (Bornology obligation closure route)**.** *Every accepted* $\mathsf{Bornology}^{\uparrow}$ *obligation read exposes exactly the route*

$$F, S, U, E, D, H, C, P, N.$$

The rows are the bounded-family source, subset-heredity ledger, finite-union ledger, singleton and source exposure, $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *evidence. The route exports no host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyObligationClosureRoute`

Proof. The carrier of Definition 3370.1 supplies the visible rows $F, S, U, E, D, H, C, P, N$. The NameCert obligations of Definition 3370.2 identify those rows as carrier admission, bounded-family source exposure, heredity, finite union, source exposure, consumer handoff, transport, replay, provenance, and local naming.

The bounded-family union stability theorem Definition 3370.3 keeps union and subfamily reads inside U and S after the source rows F, E have been exposed. The CauchyBornology handoff of Definition 3370.4 routes every Cauchy consumer through the same displayed surface before any Cauchy-specific budget can be opened.

Together these results exhaust the finite obligation route. Since none of the carrier rows names host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness, no accepted obligation read can export those omitted principles. □

Theorem 2418.6 (Bornology scoped bounded-family route)**.** *Every scoped bounded-family consumer of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier*

$$B = (F, S, U, E, D, H, C, P, N)$$

must read the route

$$F, S, U, E, D, H, C, P, N$$

through the displayed $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before using the bounded-family surface. The route exports no host supremum, selected bound, quotient family, countable-choice selector, open-cover compactness principle, or ambient completeness premise.*

Proof. Start with the NameCert obligations of Definition 3370.2. They expose the bounded-family source row F, the subset-heredity ledger S, the finite-union ledger U, the singleton/source row E, the consumer handoff row D, and the structural rows H, C, P, N as the complete local surface.

Use bounded-family union stability, Definition 3370.3. It keeps every finite union and subfamily read inside U and S after F, E have been displayed, so the bounded-family consumer cannot replace the packet by a host family or selected bound.

Next apply the CauchyBornology handoff of Definition 3370.4. It forces completion-facing bounded-family reads through the same F, E, D, H, C, P, N surface before any Cauchy-specific budget is opened.

Finally compose with the obligation closure route Definition 3370.5. The scoped consumer therefore reads exactly the displayed carrier route through $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with no additional supremum, choice, quotient, compactness, or completeness coordinate. □

Theorem 2418.7 (Bornology scope kernel grounding)**.** *The bounded-family source, subset-heredity, finite-union, consumer handoff, transport, replay, provenance, and local NameCert rows of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier are scoped exactly over* $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Every scoped consumer must read the displayed route*

$$F, S, U, E, D, H, C, P, N$$

through those kernel and sibling rows, and the route adds no host supremum, selected bound, quotient family, countable-choice selector, compactness principle, or ambient completeness premise.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyScopeKernelGrounding`

Proof. Start with the NameCert obligations of Definition 3370.2. They expose carrier admission, bounded-family source rows, subset heredity, finite union, singleton/source exposure, consumer handoff, componentwise transport, displayed replay, provenance, and local naming as the local $\mathsf{Bornology}^{\uparrow}$ surface.

Use bounded-family union stability, Definition 3370.3. It keeps heredity and finite-union reads inside the displayed S and U ledgers after F and E have been exposed, so a scoped consumer cannot replace the packet by host suprema, selected bounds, quotient families, or choice data.

Apply the CauchyBornology handoff of Definition 3370.4. The handoff row D presents the same bounded-family surface to $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, and $\mathsf{CauchyBornology}^{\uparrow}$ consumers before any Cauchy-specific or completion-facing data are opened.

Finally compose the obligation closure route Definition 3370.5 with the scoped bounded-family route Definition 3370.6. The resulting kernel grounding is exactly the route through $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with no extra compactness or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3370.1, Definition 3370.2, Definition 3370.3, Definition 3370.4, Definition 3370.5, Definition 3370.6, and Definition 3370.7 bind the finite bounded-family NameCert surface over bounded-set, filter-base, uniform-space, and Cauchy-bornology consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host suprema, selected bounds, quotient families, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure asks for a public bounded-family export theorem and later Lean verification for BEDC.Derived.BornologyUp.

Constructive story: A Bornology$^{\uparrow}$ packet is generated from bounded-family source rows, subset-heredity ledgers, finite-union ledgers, singleton and source exposure, FilterBaseUp or UniformSpaceUp handoff rows, CauchyBornologyUp specialization, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2419

A Concrete Naming Certificate for $\mathsf{MetricAnnulus}^{\uparrow}$

Following Kelley on metric neighbourhoods and Bishop–Bridges constructive metric spaces, the $\mathsf{MetricAnnulus}^{\uparrow}$ packet records the shell between an inner exclusion radius and an outer containment radius as finite BEDC certificate data. It sits over the metric source of chapter 57, the metric-ball surface of chapter 1450, the open-ball exclusion route of chapter 2191, the closed-ball containment route of chapter 1451, and the real-name radius seal of chapter 38. The packet is a metric shell neighbourhood certificate, not a host subset, not quotient point equality, not a selected-radius principle, not compactness, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2419.1 (MetricAnnulus carrier)**.** *A* $\mathsf{MetricAnnulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, c, r_0, r_1, O, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the finite distance-comparison surface.*
2. *c is the centre row, read only through the same metric source.*
3. *r_0 is the inner radius row, exposed as a positive or nonnegative* $\mathsf{Real}^{\uparrow}$ *radius seal according to the local metric-ball convention.*
4. *r_1 is the outer radius row, exposed through the same* $\mathsf{Real}^{\uparrow}$ *radius discipline.*
5. *O is the* $\mathsf{OpenBall}^{\uparrow}$ *exclusion row, recording that points in the shell are outside the displayed inner open ball around c with radius r_0.*
6. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *containment row, recording that points in the shell remain inside the displayed closed ball around c with radius r_1.*
7. *H records componentwise* hsame *transport for the metric source, centre, two radius rows, open-ball exclusion row, closed-ball containment row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a shell-membership request through the radius-order check, the open-ball exclusion test, and the closed-ball containment test.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{OpenBall}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$ *row.*

The carrier accepts only packets whose metric source, centre, inner radius, outer radius, exclusion row, containment row, transport, replay, provenance, and local name are all displayed in the same finite ledger.

Theorem 2419.2 (MetricAnnulus NameCert obligations)**.** *For every accepted* $\mathsf{MetricAnnulus}^{\uparrow}$ *carrier* $K = (M, c, r_0, r_1, O, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source and centre, visibility of the ordered inner and outer radius rows, inner-boundary exclusion through the* $\mathsf{OpenBall}^{\uparrow}$ *row, outer-boundary containment through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row, classifier transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every shell-neighbourhood read is confined to those rows and cannot open a host subset, quotient point equality, selected-radius choice, compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted packet to the displayed rows $M, c, r_0, r_1, O, B, H, C, P, N$. The metric source and centre are the only source coordinates, and the two radius rows are the only numerical boundary coordinates available to a shell read.

The inner boundary is read through O: a point must avoid the displayed open ball around c at radius r_0. The outer boundary is read through B: the same point remains inside the displayed metric closed ball around c at radius r_1.

The transport row H and replay row C keep the classifier and continuation route componentwise on this packet. The provenance and local name rows P, N then certify that every later consumer reaches the shell only through the metric source, centre, radius rows, exclusion row, and containment row already displayed.

Since the carrier has no field for a host subset, quotient equality on points, a selected-radius operator, compactness data, or a completeness theorem for $\mathsf{Real}^{\uparrow}$, the ledger cannot escape to those principles. □

Lean4 ✓ `BEDC.Derived.MetricAnnulusUp.MetricAnnulusCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricAnnulus}^{\uparrow}$ packet is generated from a Metric source row, a centre row, inner and outer Real radius rows, an OpenBall exclusion row, a MetricClosedBall containment row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$, seed)

Scope closed: Definition 3371.1 and Definition 3371.2 seed the finite metric shell certificate for displayed inner exclusion and outer containment reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host subsets, quotient point equality, selected radii, compactness, theorem-level Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, radius-order visibility, inner-boundary exclusion, outer-boundary containment, classifier transport, Cont replay, Pkg provenance, and ledger nonescape for BEDC.Derived.MetricAnnulusUp.

Chapter 2420

A Concrete Naming Certificate for RealArchimedeanProperty$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RealArchimedeanProperty$^{\uparrow}$ packet records the finite BEDC surface by which a displayed Real$^{\uparrow}$ name receives a natural-number bound and the corresponding rational and dyadic comparison rows. It sits next to the RealArchimedeanBoundSchema$^{\uparrow}$ neighbour of chapter 2201, but it names the property-level handoff: every bound read starts from the Real$^{\uparrow}$ name of chapter 38, passes through Nat$^{\uparrow}$ of chapter 28, Rat$^{\uparrow}$ of chapter 37, DyadicRatCore$^{\uparrow}$ of chapter 291, StreamName$^{\uparrow}$ of chapter 126, and RegSeqRat$^{\uparrow}$ of chapter 301, and only then exposes an Archimedean bound ledger. The packet is a finite naming certificate for that route, not a host real-order axiom, trichotomy principle, quotient-real equality, or completeness theorem.

Definition 2420.1 (RealArchimedeanProperty carrier)**.** *A* RealArchimedeanProperty$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, n, q, d, S, G, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the* Real$^{\uparrow}$ *name whose finite real observations are being bounded.*
2. *n is the* Nat$^{\uparrow}$ *bound row, recording the natural-number index used by the bound.*
3. *q is the* Rat$^{\uparrow}$ *comparison row naming the rational face of the same bound.*
4. *d is the* DyadicRatCore$^{\uparrow}$ *comparison row naming the dyadic face of the same bound.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected before the comparison is read.*
6. *G is the* RegSeqRat$^{\uparrow}$ *readback row that turns the inspected stream window into a regular rational Cauchy observation.*
7. *B is the Archimedean bound ledger, compatible with the* RealArchimedeanBoundSchema$^{\uparrow}$ *route of chapter 2201.*
8. *H records componentwise* hsame *transport for the real name, natural bound, rational row, dyadic row, stream window, readback row, bound ledger, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a bound request through S, G, q, d, and n.*
10. *P is the* Pkg *provenance over* RealArchimedeanProperty$^{\uparrow}$, Real$^{\uparrow}$, Nat$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealArchimedeanBoundSchema$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$ *row.*
The carrier accepts only packets whose real observation, bound index, rational and dyadic comparisons, stream inspection, regular rational readback, bound ledger, transport, replay, provenance, and local name are all displayed in the same finite certificate.

Theorem 2420.2 (RealArchimedeanProperty NameCert obligations)**.** *For every accepted* RealArchimedeanPropert *carrier $A = (R, n, q, d, S, G, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Real}^{\uparrow}$ *name,* $\mathsf{Nat}^{\uparrow}$ *bound row,* $\mathsf{Rat}^{\uparrow}$ *comparison row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, Archimedean bound ledger, componentwise transport through H, displayed continuation replay through C,* Pkg *provenance through P, and local naming through N. No obligation may read a host real-order axiom, trichotomy oracle, quotient-real equality, selected unbounded search, or ambient completeness theorem.*

Proof. Project the accepted carrier. The rows $R, n, q, d, S, G, B, H, C, P, N$ are the only coordinates available to the local certificate, so carrier admission and local naming are finite ledger facts.

The observation route begins with R and the inspected stream window S. The readback row G turns that finite window into the rational Cauchy surface already accepted by $\mathsf{RegSeqRat}^{\uparrow}$.

The bound route then reads q, d, and n as the rational, dyadic, and natural faces of the same displayed bound ledger B. The transport and replay rows preserve exactly these coordinates, and the provenance row names the sibling chapters from which they are read.

Since the carrier has no coordinate for host order, trichotomy, quotient equality, an unbounded search principle, or completeness, none of those principles can enter the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealArchimedeanPropertyUp.RealArchimedeanPropertyTasteGate single carrier alignment`

Theorem 2420.3 (RealArchimedeanProperty natural-rational bound handoff)**.** *Every order-facing or approximation-facing consumer of an accepted* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *packet reaches the natural bound row only through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Nat}^{\uparrow}.$$

The rational and dyadic comparison rows are read before the natural-number bound is exported, and the same handoff is compatible with the $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ *ledger of chapter 2201.*

Proof. Begin with a consumer requesting a bound for the displayed real name. The continuation row C first routes the request through the finite $\mathsf{StreamName}^{\uparrow}$ window S and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row G.

The comparison rows q and d are then read as rational and dyadic witnesses for the same inspected real observation. Only after these rows are displayed does the natural-number row n become available as the bound face of the ledger B.

The natural bound row is consumed by finite unary induction over the displayed bound ledger. The base step reads the initial ledger face, and each successor step reuses the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, rational comparison, and $\mathsf{DyadicRatCore}^{\uparrow}$ comparison rows. The induction exports only the displayed Nat/Rat/Dyadic/Real route; it does not introduce unbounded search, trichotomy, host order, quotient equality, or completeness.

The compatibility with $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ is structural: both packets expose the same real, stream, regular readback, rational, dyadic, and bound-ledger coordinates. The property packet adds the explicit $\mathsf{Nat}^{\uparrow}$ handoff without adding host order or completeness data. □

Theorem 2420.4 (RealArchimedeanProperty ledger nonescape)**.** *For an accepted* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *packet, every Archimedean-property read factors through the displayed rows* $R, n, q, d, S, G, B, H, C, P, N$*. In particular, the packet exports no hidden real supremum, no total order trichotomy, no quotient equality of real names, no unbounded search over naturals, no selected least bound, and no completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Inspect the carrier inventory. The only real-facing row is R, the only finite observation rows are S and G, the only comparison rows are q and d, and the only natural bound row is n.

The replay row C can only traverse those displayed coordinates, while H transports them componentwise and P, N package the same route as provenance and local naming. Thus any Archimedean-property read must pass through the finite ledger described in the carrier.

Because no row stores a supremum, trichotomy decision, quotient equality test, unbounded natural search, selected least-bound operator, or completeness theorem, the packet cannot export any of those structures. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealArchimedeanProperty}^{\uparrow}$ packet is generated from a RealUp name, NatUp bound row, RatUp comparison row, DyadicRatCoreUp comparison row, StreamNameUp finite-window row, RegSeqRatUp readback row, Archimedean bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3372.1, Definition 3372.2, Definition 3372.3, and Definition 3372.4 seed the finite property-level bound route over Real, StreamName, RegSeqRat, Rat, DyadicRatCore, Nat, and RealArchimedeanBoundSchema rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real order, trichotomy, quotient equality of real names, unbounded search, least-bound selection, Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, natural-rational-dyadic bound compatibility, finite-window readback, transport stability, replay stability, and ledger nonescape for BEDC.Derived.RealArchimedeanPropertyUp.

Chapter 2421

A Concrete Naming Certificate for $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ packet records the monotone-convergence route that is available when a bounded monotone real sequence is handed to a located supremum surface. It sits between the source sequence packet of chapter 1434, the located upper-bound and lower-approximation packet of chapter 1427, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite handoff from monotone sequence data to a located supremum read; it does not import arbitrary completeness of the real line.

Definition 2421.1 (Located monotone convergence carrier)**.** *A* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, B, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *source row, exposing the inspected sequence schedule, finite stream windows, regular readbacks, real seals, boundedness ledger, and monotonicity ledger.*
2. *B is the monotone bounded ledger read from M, recording the common upper-bound surface and the finite comparison rows that keep each inspected sequence value below it.*
3. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, displaying the upper-bound and lower-approximation ledgers supplied to the supremum packet.*
4. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *precision-window row through which the sequence observations, upper-bound ledger, and lower-approximation requests are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windowed sequence observations and the located-supremum approximation ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested dyadic precision, tail threshold, and comparison radii used to relate a late monotone term to the located supremum surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, U, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the monotone source, bounded ledger, located-supremum handoff, precision windows, regular readback, dyadic tolerance ledger, real seal, continuation route, provenance, and local naming row.*

9. *C records displayed* Cont *replay from a convergence request through $M, B, U, W, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* LocatedMonotoneConvergence$^{\uparrow}$, MonotoneRealSequence$^{\uparrow}$, LocatedSupremum$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocatedMonotoneConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through

$$M, B, U, W, R, D, E, H, C, P, N.$$

It has no coordinate for arbitrary real completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, or a selected ambient limit outside the displayed located-supremum handoff.

Lean4 ✓ `BEDC.Derived.LocatedMonotoneConvergenceUp.LocatedMonotoneConvergenceTasteGate single carrier alignment`

Theorem 2421.2 (Located monotone convergence NameCert obligations)**.** *For every accepted* LocatedMonotoneConvergence$^{\uparrow}$ *carrier*

$$L = (M, B, U, W, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{LocatedMonotoneConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the monotone source, bounded ledger, located-supremum handoff, shared precision window, regular readback, dyadic tolerance ledger, real seal, transports, replay, provenance, and local naming;*
2. *monotone bounded ledger admission, requiring B to be read from the same* MonotoneRealSequence$^{\uparrow}$ *packet M that supplies the sequence schedule and monotonicity ledger;*
3. *located-supremum handoff, requiring U to expose upper-bound and lower-approximation rows through the* LocatedSupremum$^{\uparrow}$ *surface before any convergence read is exported;*
4. RegSeqRat$^{\uparrow}$ *readback, requiring R to repack the inspected stream windows and approximation witnesses rather than a hidden real representative;*
5. *dyadic tolerance replay, requiring D to bind each requested precision to a finite tail threshold and comparison radius;*
6. Real$^{\uparrow}$ *seal non-escape, requiring E to remain downstream of M, B, U, W, R, D together with H, C, P, N.*

No obligation may call on arbitrary Real$^{\uparrow}$ *completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, or an ambient selected limit.*

Proof. Project the carrier of Definition 3373.1. Carrier admission exposes exactly the rows $M, B, U, W, R, D, E, H, C$ so the only sequence source is the accepted MonotoneRealSequence$^{\uparrow}$ row M.

Read the order-facing rows next. The bounded ledger B is attached to the same inspected schedule and monotonicity ledger carried by M, while U is the only located-supremum handoff row and therefore the only place where upper-bound and lower-approximation information can be read.

The convergence-facing rows are downstream of these order rows. The shared window W, regular readback R, and dyadic tolerance ledger D supply the finite precision request, readback route, tail threshold, and comparison radius before the real seal E is exposed.

Finally the structural rows H, C, P, N transport, replay, source, and name the same packet. Since the excluded completeness, order, quotient, choice, compactness, and selected-limit coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Theorem 2421.3 (Located monotone convergence supremum readback)**.** *Let $L = (M, B, U, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing convergence read exported by L factors through the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row U and its displayed upper-bound and lower-approximation ledgers, then through the shared* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. Thus the read is a located-supremum factorization of the monotone source M, not an appeal to ambient completeness of arbitrary real subsets.*

Lean4 ✓ `BEDC.Derived.LocatedMonotoneConvergenceUp.LocatedMonotoneConvergenceCarrier supremum readback`

Proof. Start with a Real-facing convergence consumer. By Definition 3373.1, the only terminal real coordinate is E, and it is displayed after the monotone source, bounded ledger, located-supremum handoff, shared window, regular readback, and dyadic tolerance ledger.

Apply Definition 3373.2. Its located-supremum handoff row forces the upper-bound and lower-approximation reads through U, while its readback and tolerance rows force the same request through W, R, D before E can be read.

The packet M supplies only the bounded monotone source surface, and H, C, P, N can only transport, replay, source, and name the displayed route. Therefore every exported convergence read factors through $\mathsf{LocatedSupremum}^{\uparrow}$ and not through an ambient completeness principle. □

Theorem 2421.4 (Located monotone convergence bounded source route)**.** *Let $L = (M, B, U, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet whose source row is supplied by an accepted* $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *packet. Then every convergence read first consumes the source packet's monotonicity ledger, boundedness ledger, and* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff before it may inspect the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row U, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The readback is therefore a bounded-monotone-source route followed by a located-supremum route, not an ambient completeness shortcut.*

Proof. Project the carrier of Definition 3373.1. The rows M and B are the only source-facing rows available before the located-supremum handoff U and the readback rows W, R, D, E.

When M is supplied by $\mathsf{MonotoneBoundedSequence}^{\uparrow}$, apply Definition 3407.4. The source-side read must expose the monotonicity ledger, boundedness ledger, and $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff through the same finite source surface before a located convergence consumer can inspect later rows.

Now apply Definition 3373.3. After the bounded source route is fixed, every $\mathsf{Real}^{\uparrow}$-facing convergence read factors through U, then through the shared window and readback rows W, R, D, and finally through the terminal seal E.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and naming. Since neither carrier has a coordinate for arbitrary real completeness or a selected ambient limit, the readback cannot bypass the bounded-source route. □

Theorem 2421.5 (Located monotone convergence source induction)**.** *Let*

$$L = (M, B, U, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing convergence read exported by* L *is generated by induction over the displayed finite rows* $M, B, U, W, R, D, E, H, C, P, N$*. The base rows are the monotone source* M *and boundedness ledger* B*. The step rows are the located-supremum handoff* U*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*,* Cont *replay* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}$ *row* N*. The induction exposes no selected limit, quotient representative, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the finite carrier displayed in Definition 3373.1. Its only rows are $M, B, U, W, R, D, E, H, C, P, N$, so an induction on accepted convergence reads has base information only from the monotone source and boundedness ledger.

Apply Definition 3373.2. The NameCert obligations force carrier admission, bounded ledger admission, located-supremum handoff, readback, tolerance replay, and Real seal non-escape to occur through those same displayed rows.

The readback steps are fixed by Definition 3373.3. After the base rows are admitted, each Real-facing read must pass through U, W, R, D, E in order, with H, C, P, N only transporting, replaying, sourcing, and naming the route.

If the source is supplied by a bounded monotone packet, Definition 3373.4 identifies the same base rows as the monotonicity ledger, boundedness ledger, and finite source handoff before any later convergence row can be inspected.

Thus the induction covers every exported convergence read using exactly the displayed finite rows. Since the carrier and its NameCert obligations contain no coordinate for a selected limit, quotient representative, countable choice, trichotomy, or ambient Real completeness, none can appear in the generated read. □

Falsifiable prediction: Every accepted $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ convergence read factors through a MonotoneRealSequence source, a boundedness ledger, a LocatedSupremum handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real-facing convergence consumer may inspect it.

Independence witness: monotone-real-sequence, locatedsupremum, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded monotone sequence source and a located-supremum handoff before the Real-facing convergence read is sealed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3373.1, Definition 3373.2, Definition 3373.3, Definition 3373.4, and Definition 3373.5 seed the finite handoff from bounded monotone real sequence data to a located-supremum Real-facing convergence read.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, monotone bounded ledger admission, located-supremum handoff, RegSeqRat readback, dyadic tolerance replay, and Real seal non-escape for BEDC.Derived.LocatedMonotoneConvergenceU

Constructive story: A $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ packet is generated from a MonotoneRealSequence source, a monotone bounded ledger, a LocatedSupremum handoff, a StreamName precision window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2422

A Concrete Naming Certificate for $\mathsf{BaireSteinhaus}^{\uparrow}$

Following Baire, Steinhaus, and Banach–Steinhaus, the $\mathsf{BaireSteinhaus}^{\uparrow}$ packet records the finite BEDC handoff by which a Baire-category witness, a complete metric source, norm rows, functional-analysis probes, and a uniform-boundedness consumer are named in one certificate. The packet sits between the Baire-category surface of chapter 1542, the complete-metric surface of chapter 128, the norm surface of chapter 89, the functional-analysis source of chapter 94, and the uniform-boundedness boundary of chapter 1835. It is a traditional analysis reconstruction in the BEDC carrier language: the certificate names the finite pointwise-bounded handoff without asserting host Banach-space completeness, countable choice, quotient equality of function spaces, or an unrestricted uniform-boundedness theorem.

Definition 2422.1 (Baire–Steinhaus carrier)**.** *A* $\mathsf{BaireSteinhaus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, R, F, U, W, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open row read by the pointwise-bounded argument.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row supplying the complete metric carrier on which the Baire row is interpreted.*
3. *R is the* $\mathsf{Norm}^{\uparrow}$ *row used to read boundedness witnesses as finite norm estimates.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *source row naming the family of linear probes and the finite test windows visible to the packet.*
5. *U is the* $\mathsf{UniformBoundedness}^{\uparrow}$ *consumer row that may receive a uniform-bound request only after B, M, R, F have been displayed.*
6. *W is the pointwise-boundedness ledger, recording the inspected point, operator family window, norm bound, and finite comparison tolerance.*
7. *E is the Baire–Steinhaus handoff row from the pointwise ledger W to the uniform-boundedness consumer U.*
8. *H records componentwise* hsame *transport for B, M, R, F, U, W, E.*
9. *C records displayed* Cont *replay from a pointwise-bounded request through the Baire, complete-metric, norm, functional-analysis, and handoff rows.*

10. *P is the* Pkg *provenance over* BaireSteinhaus$^{\uparrow}$, BaireCategory$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Norm$^{\uparrow}$, FunctionalAnalysis$^{\uparrow}$, UniformBoundedness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$ *row.*

The carrier accepts only packets whose Baire, complete-metric, norm, functional-analysis, pointwise-boundedness, uniform-boundedness, transport, replay, provenance, and local naming rows are displayed in the same finite certificate.

Lean4 ✓ `BEDC.Derived.BaireSteinhausUp.BaireSteinhausTasteGate single carrier alignment`

Theorem 2422.2 (Baire–Steinhaus NameCert obligations)**.** *For every accepted* BaireSteinhaus$^{\uparrow}$ *carrier $S = (B, M, R, F, U, W, E, H, C, P, N)$, the local* NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* BaireCategory$^{\uparrow}$ *row,* CompleteMetric$^{\uparrow}$ *row,* Norm$^{\uparrow}$ *row,* FunctionalAnalysis$^{\uparrow}$ *row,* UniformBoundedness$^{\uparrow}$ *consumer row, pointwise-boundedness ledger, Baire–Steinhaus handoff, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. No obligation may read host Banach-space completeness, countable choice, quotient equality of function spaces, a selected dense point, or an unrestricted uniform-boundedness theorem.*

Proof. Project the carrier of Definition 3374.1. The rows $B, M, R, F, U, W, E, H, C, P, N$ are the only coordinates available to the local certificate, so every obligation is a finite row obligation.

The Baire-category side is read through B over the complete metric row M. The norm and functional-analysis rows R and F supply only the displayed finite observations used by the pointwise-boundedness ledger W.

The handoff row E is then the only route from W to the uniform-boundedness consumer U. Transport, replay, provenance, and local naming preserve that same route, and the carrier contains no coordinate for the excluded host-level principles. □

Theorem 2422.3 (Baire–Steinhaus pointwise-bounded handoff)**.** *Every pointwise-boundedness read accepted by a* BaireSteinhaus$^{\uparrow}$ *packet factors through the finite route*

$$\text{BaireCategory}^{\uparrow},\ \text{CompleteMetric}^{\uparrow},\ \text{Norm}^{\uparrow},\ \text{FunctionalAnalysis}^{\uparrow} \longmapsto \text{UniformBoundedness}^{\uparrow}.$$

The pointwise ledger W is exposed before the handoff row E can feed U, and the UniformBoundedness$^{\uparrow}$ *consumer receives only that displayed handoff. It receives no host Banach-space completeness theorem, no global operator-family supremum, no countable-choice selector, no quotient equality of function spaces, and no selected dense point.*

Proof. Begin with a pointwise-bounded request. The continuation row C first routes the request through the Baire row B and the complete-metric row M, so the category argument is anchored in the displayed metric source.

Next the request reads the norm row R and functional-analysis row F. These rows expose the finite operator-family windows and norm estimates stored in W, not a host family of all operators.

Finally the handoff row E repacks the displayed pointwise ledger for the uniform-boundedness consumer U. Since H, P, N only transport, source, and name the same finite rows, the consumer cannot recover any excluded completeness, choice, quotient, supremum, or selected-point principle. □

Closure status for NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$**.**

Constructive story: A BaireSteinhaus$^{\uparrow}$ packet is generated from a BaireCategory row, CompleteMetric row, Norm row, FunctionalAnalysis source row, UniformBoundedness consumer row, pointwise-boundedness ledger, handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BaireSteinhaus}\uparrow}$, seed)

Scope closed: Definition 3374.1, Definition 3374.2, and Definition 3374.3 seed the finite pointwise-boundedness handoff over BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, and UniformBoundedness rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BaireSteinhaus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Banach-space completeness, countable choice, quotient equality of function spaces, selected dense points, global operator-family suprema, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise-boundedness ledger exposure, Baire-category routing, norm and functional-analysis compatibility, uniform-boundedness handoff, transport replay, provenance, and local naming for BEDC.Derived.BaireSteinhausUp.

Chapter 2423

A Concrete Naming Certificate for $\mathsf{BanachSteinhaus}^{\uparrow}$

Following Banach and Steinhaus, the $\mathsf{BanachSteinhaus}^{\uparrow}$ packet records the traditional theorem-level surface behind the uniform boundedness principle as finite NameCert data. It is the neighbour of the Baire handoff in chapter 3374 and the operator-family certificate in chapter 2302: the packet names the theorem route from Baire-category completeness data to a uniform boundedness read without importing a host Banach-space theorem, countable choice, quotient equality of function spaces, or an unrecorded operator-family supremum.

Definition 2423.1 (Banach–Steinhaus carrier)**.** *A* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, M, N, F, U, L, W, H, C, P, K)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *row that supplies the category witness used by the theorem-level route.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over the displayed source, keeping the Baire read tied to a complete metric carrier.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row through which pointwise and uniform boundedness are read as norm estimates.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row naming the linear-probe family and the finite observation windows visible to the packet.*
5. *U is the* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *row that receives the theorem-level uniform-bound read after the Baire, completeness, norm, and functional-analysis rows have been displayed.*
6. *L is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *family row over the same source and target surfaces.*
7. *W is the pointwise-boundedness-to-uniform-bound ledger, recording the inspected source windows, operator rows, norm bounds, and the single uniform operator-bound output row.*
8. *H records componentwise* hsame *transport for B, M, N, F, U, L, W.*
9. *C records displayed* Cont *replay from pointwise boundedness requests through the Baire-category, complete-metric, norm, functional-analysis, bounded-operator, and uniform-boundedness rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BanachSteinhaus}^{\uparrow}$, $\mathsf{BaireSteinhaus}^{\uparrow}$, $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *K is the local* $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host-level Banach-space completeness, countable choice, quotient equality of maps, an unrestricted operator-family selector, or a theorem object outside the finite route.

Lean4 ✓ `BEDC.Derived.BanachSteinhausUp.BanachSteinhausTasteGate single carrier alignment`

Theorem 2423.2 (Banach–Steinhaus NameCert obligations)**.** *For every accepted* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier* $T = (B, M, N, F, U, L, W, H, C, P, K)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire-category row, complete-metric row, norm row, functional-analysis row, uniform-boundedness-principle row, bounded-linear-operator family row, pointwise-to-uniform ledger, componentwise transport, displayed replay, provenance, and local naming; Baire-source compatibility between B and M; operator-family compatibility between F and L; uniform-bound exactness through W and U; and non-escape from host completeness, countable choice, quotient equality of maps, unrestricted operator-family selectors, and theorem objects outside the displayed finite packet.*

Proof. Project the carrier of Definition 3375.1. The tuple $B, M, N, F, U, L, W, H, C, P, K$ is the whole accepted packet, so carrier admission and the named local rows are direct reads from the finite history.

The Baire-source compatibility is the shared route through B and M. The category row is interpreted only over the complete metric row displayed in the same packet, so a consumer cannot detach the Baire read from the carried source.

The operator-family compatibility is the shared route through F and L. The functional-analysis row names the finite probe windows, and the bounded-linear-operator row records the operator-family data over those same windows.

The uniform-bound exactness is the ledger W feeding the uniform-boundedness-principle row U. The transport, replay, provenance, and naming rows H, C, P, K certify that the same displayed route is preserved.

Since no coordinate names the excluded host-level objects, the local NameCert surface exports only the finite Banach–Steinhaus route. □

Theorem 2423.3 (Banach–Steinhaus Baire handoff)**.** *Every theorem-level read of an accepted* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier factors through the ordered handoff*

$$\mathsf{BaireCategory}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{UniformBoundednessPr}$$

The $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *row receives the ledger W only after the Baire and complete-metric rows have been exposed through the same* BHist *packet. Thus the packet is a theorem-level neighbour of chapter 3374 and chapter 2302, not a replacement for either sibling and not a host Banach–Steinhaus theorem outside BEDC data.*

Proof. Read the pointwise-boundedness request through the continuation row C. The first part of the route exposes the Baire-category row B over the complete-metric row M, matching the Baire side of the handoff named in chapter 3374.

Next read the norm row N, the functional-analysis row F, and the bounded-linear-operator family row L. These rows provide the finite operator windows and norm estimates that can be assembled into the pointwise-to-uniform ledger W.

Finally feed W to the uniform-boundedness-principle row U, the same theorem-level consumer named in chapter 2302. The rows H, P, K transport, source, and name this route without adding an external theorem object.

The exported handoff is therefore exactly the displayed finite route, with no host completeness principle, countable-choice selector, quotient equality of function spaces, or unrestricted family supremum. □

Falsifiable prediction: Every accepted $\mathsf{BanachSteinhaus}^{\uparrow}$ packet must expose $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows before any theorem-level uniform-bound read is public.

Independence witness: baire-steinhaus, uniform-boundedness-principle, boundedlinearoperator, norm, complete-metric: $\mathsf{BanachSteinhaus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the Baire-category source, complete-metric source, norm read, functional-analysis probe family, bounded-operator family, pointwise-to-uniform ledger, theorem-level uniform-boundedness consumer, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3375.1, Definition 3375.2, and Definition 3375.3 seed the finite theorem-level Banach–Steinhaus route over BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, UniformBoundednessPrinciple, BoundedLinearOperator, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Banach-space completeness, countable choice, quotient equality of maps, unrestricted operator-family selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-source compatibility, operator-family compatibility, uniform-bound ledger exactness, Baire handoff stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.BanachSteinhausUp.

Constructive story: A $\mathsf{BanachSteinhaus}^{\uparrow}$ packet is generated from BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, UniformBoundednessPrinciple, BoundedLinearOperator, pointwise-to-uniform ledger, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2424

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflection}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley reflection terminology, the $\mathsf{CauchyCompletionReflection}^{\uparrow}$ packet records the finite BEDC boundary where Cauchy completion data reflect through the displayed unit and counit routes before reaching the Real seal. It reads the dense unit route of chapter 1527, the completed-object return route of chapter 1539, the comparison surface of chapter 1954, the broader completion-reflection boundary of chapter 403, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a nonquotient reflection certificate for displayed completion routes; it does not identify quotient completions, choose stream representatives, assert host equality, invoke countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2424.1 (CauchyCompletionReflection carrier)**.** *A* $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row admitting source data into the completion-facing presentation.*
2. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning completed data to a displayed completion consumer.*
3. *Q is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row comparing the unit/universal-property presentation with the functorial metric-completion presentation.*
4. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *row tying the unit and counit routes to the same reflection boundary.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected before any terminal seal can be read.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the stream windows.*
7. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row shared by the unit, counit, comparison, and reflection routes.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, A have been displayed.*

9. *H records componentwise* hsame *transport for U, V, Q, F, S, D, A, E and the structural rows.*

10. *C records displayed* Cont *replay of the route*

$$U, V, Q, F \longmapsto S, D, A \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflection}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectionUp.CauchyCompletionReflectionTasteGate single carrier alignment`

Theorem 2424.2 (CauchyCompletionReflection NameCert obligations)**.** *For every accepted* CauchyCompletionRefl *carrier*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unit route, counit route, comparison row, reflection row, stream windows, dyadic ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*

2. *unit exposure, requiring source admission to pass through U before a completion-facing read can inspect Q or F;*

3. *counit exposure, requiring completed-object return to pass through V before the same Q, F reflection boundary is read;*

4. *comparison agreement, requiring the unit route and counit route to use the shared* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row Q before the finite analytic route is public;*

5. *reflection stability, requiring F to be the only reflection witness tying U, V, Q to the terminal seal;*

6. *finite seal routing, requiring all Real-facing reads to pass through S, D, A, E and then through H, C, P, N;*

7. *non-escape, requiring the packet to source and name no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 3376.1. The accepted tuple exposes exactly the unit, counit, comparison, reflection, analytic readback, and structural rows listed there.

For the unit side, the source row is read through U before any completion-facing comparison is available. For the counit side, the completed-object return is read through V before the same comparison and reflection boundary can be inspected.

The comparison and reflection obligations use Q and F. The comparison row identifies the displayed completion presentations, while F records that both the unit and counit reads are attached to the same completion-reflection boundary.

Finally the finite seal route is forced through S, D, A, E. Induction on the finite $\mathsf{StreamName}^{\uparrow}$ windows in S shows that each inspected window carries only its dyadic tolerance row and regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is reached. The transport, replay, provenance, and naming rows H, C, P, N preserve only these displayed coordinates, so no excluded completion or choice principle is exported. □

Theorem 2424.3 (CauchyCompletionReflection unit-counit triangle)**.** *Let*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier. Every consumer that compares the unit route with the counit route factors through the finite triangle*

$$U, V \longmapsto Q, F \longmapsto S, D, A \longmapsto E.$$

The triangle is a displayed NameCert route over $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It is not quotient completion equality, not selected representative equality, not host equality of completed objects, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier. A unit-facing read has access to U, and a counit-facing read has access to V; neither route contains a terminal seal on its own.

The two reads meet only at Q and F. The comparison row supplies the shared completion presentation, and the reflection row records the boundary that both the unit and counit routes must use.

The remaining analytic data are S, D, A, E. Induction on the finite stream-window family shows that every triangle consumer sees exactly the window row, attached dyadic tolerance, regular rational readback, and Real seal before the structural rows H, C, P, N replay and name the route. Since no other coordinate is present, the excluded quotient, representative, host-equality, choice, and completeness data cannot be read. □

Theorem 2424.4 (CauchyCompletionReflection Real-seal handoff)**.** *For an accepted* $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier* $R = (U, V, Q, F, S, D, A, E, H, C, P, N)$, *every Real-facing completion-reflection consumer receives the terminal seal* E *only through the route*

$$U, V, Q, F \longmapsto S, D, A \longmapsto E \longmapsto H, C, P, N.$$

Thus the handoff to $\mathsf{Real}^{\uparrow}$ *is mediated by the sibling* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows and by the displayed* $\mathsf{CompletionReflection}^{\uparrow}$ *boundary. It exports no quotient completion equality, selected representative, host equality, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean bridge, or detached seal.*

Proof. Start with a Real-facing consumer. Projection of the accepted carrier gives the only available completion-reflection rows: U, V, Q, F and the finite analytic route S, D, A, E.

The consumer cannot read E before the sibling route has been displayed. The stream windows in S determine the finite observations, D supplies their tolerances, and A supplies the regular rational readback.

After E is reached, H and C transport and replay the same route, while P and N source and name it. No carrier coordinate stores a quotient completion, selected representative, host equality, choice schedule, compactness argument, ambient real-completeness theorem, Lean bridge, or detached seal, so no Real-facing handoff can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflection}^{\uparrow}$ packet is generated from CauchyCompletionUnit and CauchyCompletionCounit routes, a CauchyCompletionComparison row, a CompletionReflection witness, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3376.1, Definition 3376.2, Definition 3376.3, and Definition 3376.4 seed the finite nonquotient reflection boundary for L10 completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, counit exposure, comparison agreement, reflection stability, finite seal routing, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletion

Chapter 2425

A Concrete Naming Certificate for $\mathsf{CauchyRealArchimedean}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyRealArchimedean}^{\uparrow}$ packet records the finite Archimedean witness consumed by L10 Cauchy-real readers before an order, located-comparison, or completion-facing consumer opens a real-name seal. It sits over the regular rational source rows of chapter 301, the dyadic scale rows of chapter 291, the terminal real-name surface of chapter 38, the Cauchy-real presentation route of chapter 1511, and the rational-bound witness surface of chapter 430. The packet names a displayed finite bound route; it is not a host least-bound principle, not quotient equality of real names, not a choice-selected integer part, and not ambient real completeness.

Definition 2425.1 (CauchyRealArchimedean carrier)**.** *A* $\mathsf{CauchyRealArchimedean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, D, E, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose finite regular-rational observations are the only Cauchy-real reads admitted by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the dyadic tolerance, comparison budget, and rational arithmetic used to inspect G.*
3. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the displayed route through G and D.*
4. *W is the Archimedean bound witness row, read as the rational or dyadic bound surface supplied by chapter 430 for the sealed Cauchy-real name.*
5. *H records componentwise* hsame *transport for G, D, E, W, C, P, N.*
6. *C records displayed* Cont *replay from regular-rational windows through dyadic scale inspection, the terminal real seal, and the Archimedean witness row:*

$$G \longmapsto D \longmapsto E \longmapsto W.$$

7. *P is the* Pkg *provenance over* $\mathsf{CauchyRealArchimedean}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealPresentation}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through G, D, E, W, H, C, P, N. It has no coordinate for a host supremum, a selected least bound, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2425.2 (CauchyRealArchimedean NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealArchimedean}^{\uparrow}$ *carrier $A = (G, D, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed regular-rational source, dyadic scale, real seal, Archimedean witness, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring comparisons of accepted packets to inspect matching G, D, E, W rows through the componentwise* hsame *data in H;*
3. *dyadic-scale exactness, requiring every bound or comparison request to pass through the displayed scale row D before the real seal E is consumed;*
4. *Archimedean witness exposure, requiring the bound witness W to be read after the same G, D, E route and before any order or located-comparison consumer uses it;*
5. *continuation, provenance, and naming stability through C, P, N.*

Proof. Project the carrier in Definition 3378.1. The regular-rational observations, dyadic scale data, real-name seal, bound witness, transport, replay, provenance, and local name are the only displayed coordinates.

The analytic route is supplied by the sibling Cauchy-real chapters. The $\mathsf{Real}^{\uparrow}$ seal is read through chapter 38, the regular rational source through chapter 301, the scale row through chapter 291, and the presentation face through chapter 1511.

The Archimedean part is then the displayed witness row. Apply the rational-bound surface of chapter 430: it supplies a finite bound route for the same real-name seal, while this carrier records that route as W downstream of G, D, E.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no field stores a host supremum, selected least bound, quotient stream equality, countable-choice schedule, host equality row, or ambient completeness object, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyRealArchimedeanUp.CauchyRealArchimedeanCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealArchimedean}^{\uparrow}$ packet is generated from displayed $\mathsf{RegSeqRat}^{\uparrow}$ source rows, $\mathsf{DyadicRatCore}^{\uparrow}$ scale rows, a terminal $\mathsf{Real}^{\uparrow}$ seal row, an Archimedean bound witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$, seed)

Scope closed: Definition 3378.1 and Definition 3378.2 seed the finite Archimedean witness route for Cauchy-real names over RegSeqRat, DyadicRatCore, Real, CauchyRealPresentation, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected least bounds, integer-part selection, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic-scale exactness, Archimedean witness exposure, replay stability, provenance stability, and local naming for the same displayed packet.

Chapter 2426

A Concrete Naming Certificate for $\mathsf{CauchyCompletionOperator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCompletionOperator}^{\uparrow}$ packet names the finite operator surface that turns displayed Cauchy data into a completion-facing handoff without selecting quotient representatives. It sits between the metric completion carrier of chapter 1506, the Cauchy carrier of chapter 3413, the StreamName finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the completion boundary of chapter 3350, the uniform completion route of chapter 1473, the separated limit surface of chapter 2355, and the terminal real-name seal of chapter 38. The operator is therefore a finite NameCert object over visible StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, completion-boundary rows, and Real-facing seals, not a host completion class or a choice of limits.

Definition 2426.1 (CauchyCompletionOperator carrier)**.** *A* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing the metric Cauchy request and its displayed finite tolerance route;*
2. *B is the* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *row, recording the handoff from metric and uniform completion reads to finite-window and Real-facing consumers;*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row, recording the entourage-facing completion presentation consumed by the same operator;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule of finite windows inspected by the completion request;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the scheduled windows;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the Cauchy request, readback, and seal boundary;*
7. *Q is the* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *separated-limit row that receives the operator handoff only through the visible completion boundary;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, U, S, R, D, Q have been displayed;*

9. *H records componentwise* hsame *transport for the source, boundary, uniform, window, readback, dyadic, separated-limit, Real-seal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from an operator request through $M, B, U, S, R, D, Q, E, H, P, N$;*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionOperator}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompleteSeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$ *row.*

The classifier accepts only packets whose operator handoff is exhausted by $M, B, U, S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorTasteGate single carrier alignment`

Theorem 2426.2 (CauchyCompletionOperator NameCert obligation surface)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier $O = (M, B, U, S, R, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperat}}$ *surface consists exactly of carrier admission for the displayed twelve-row packet; metric-completion exposure through M; completion-boundary exposure through B; uniform-completion exposure through U; finite-window exposure through S; regular-rational readback through R; dyadic-tolerance exposure through D; separated-limit handoff through Q; terminal Real sealing through E; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperator namecert obligation surface`

↪ variant `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCarrier namecert obligation surface`

Proof. Project the carrier of Definition 3379.1. The accepted packet exposes exactly the metric source row, completion-boundary row, uniform-completion row, finite windows, regular readback, dyadic ledger, separated-limit row, Real seal, and structural rows.

The operator obligations are read in the displayed order. The metric and uniform completion requests first pass through the boundary row B; the scheduled windows S, readback R, and dyadic ledger D then supply the finite evidence needed by both completion and Real-facing consumers.

The separated-limit and Real-seal rows are downstream of the same packet. The row Q is reached only through the visible boundary route, and the terminal seal E is reached only after the finite windows, readback, and tolerance ledger have already been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient real completeness are not coordinates of O, they cannot appear as local NameCert obligations. □

Theorem 2426.3 (CauchyCompletionOperator functorial handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet factors through the finite handoff*

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

Thus the operator may be replayed by later functorial completion surfaces only as a displayed route from metric and uniform completion requests through common finite windows, dyadic tolerances,

regular readback, separated-limit handoff, Real sealing, transport, replay, provenance, and local naming.

Proof. Start with an accepted carrier. The completion-facing rows M and U are not independent host completions; they are source rows of the same finite packet.

The boundary row B aligns those source rows before any consumer reaches the scheduled windows. The rows S, R, D then expose the common finite observations, regular rational readback, and dyadic tolerance ledger used by downstream completion requests.

The rows Q and E are terminal for the separated-limit and Real-facing sides of the handoff. They do not insert a quotient completion or a selected representative between the finite evidence and the terminal consumer.

The structural rows H, C, P, N preserve this route under transport, replay it as a displayed continuation, record provenance, and name the local certificate. Therefore any later functorial completion surface can consume only the displayed operator route. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorFunctorialHandoff`

Theorem 2426.4 (CauchyCompletionOperator triangle consumption route)**.** *Every accepted* CauchyCompletionOp *packet consumed by* CauchyCompletionFunctoriality$^{\uparrow}$ *or* ***??*** *for an identity or composition triangle read must route through*

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

The triangle read cannot bypass the completion-boundary row B, the finite StreamName$^{\uparrow}$ *windows S, the* RegSeqRat$^{\uparrow}$ *readback R, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, or the separated-limit handoff Q. It exports no quotient completion equality, selected representative, countable-choice schedule, host equality, selected limit, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorTriangleConsumptionRoute`

Proof. Project the carrier of Definition 3379.1. The only available rows are $M, B, U, S, R, D, Q, E, H, C, P, N$, so an identity or composition triangle consumer has no detached completion endpoint to inspect.

Apply Definition 3379.3. It places the metric and uniform source rows before B, then forces every completion-facing read through the finite windows, regular readback, dyadic tolerance ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming.

The downstream consumers are the functorial completion surface of chapter 1937 and the separated completion functor of chapter 4094. Both may consume the operator packet only through the displayed route, so their triangle reads inherit the same completion-boundary and finite-window order.

The exclusion clause follows from the carrier inventory. None of quotient completion equality, selected representatives, selected limits, countable choice, host equality, or ambient Real completeness is a coordinate of the accepted packet. □

Theorem 2426.5 (CauchyCompletionOperator finite-window extraction)**.** *Every accepted* CauchyCompletionOpera *packet exposes its completion-facing request through a finite extraction window before any separated-limit or Real endpoint row is available. The extracted window consists exactly of the* StreamName$^{\uparrow}$ *schedule S, the* RegSeqRat$^{\uparrow}$ *readback R, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, the completion-boundary row B, and the structural rows H, C, P, N that transport, replay, source, and name the same finite packet.*

Consequently a consumer may reach Q or E only through the displayed route

$$B, S, R, D, H, C, P, N \longrightarrow Q, E.$$

No selected limit, quotient completion equality, host completion operator, host real equality, or countable-choice schedule is exposed by the extraction.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorWindowExtraction`

Proof. Project the carrier of Definition 3379.1. The only rows between the metric and uniform source requests and the terminal consumers are the completion boundary, finite StreamName schedule, regular readback, dyadic ledger, separated-limit row, Real seal, and structural support.

Apply Definition 3379.2. Its obligation surface forces local naming to inspect B, S, R, D before Q or E can be consumed, while H, C, P, N only transport, replay, source, and name the displayed packet.

Use Definition 3379.3. The handoff order places the finite window and dyadic ledger between the completion source rows and the terminal rows, so the extraction is determined by the visible window data rather than by any host completion object.

The exclusion clause follows from the same carrier inventory. The packet has no coordinate for a selected limit, quotient completion equality, host completion operator, host real equality, or countable-choice schedule, so none can be read during extraction. □

Theorem 2426.6 (CauchyCompletionOperator ledger nonescape)**.** *No accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet exports quotient completion equality, selected representatives, selected limits, countable-choice schedules, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness data. Every consumer route to* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *must first read the packet rows* $M, B, U, S, R, D, Q, E, H, C, P, N$.

Proof. Inspect the carrier inventory in Definition 3379.1. Its analytic coordinates are the metric source, completion boundary, uniform source, finite windows, regular readback, dyadic ledger, separated-limit handoff, and Real seal.

The structural coordinates are only hsame, Cont, Pkg, and NameCert rows. They move, replay, source, and name the displayed packet; they do not add analytic data.

The excluded quotient, representative, limit, choice, host-operator, and ambient-completeness objects are absent from both inventories. Hence every separated-limit or Real-facing read is forced through the visible finite ledger and cannot escape the operator packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorLedgerNonescape`

Theorem 2426.7 (CauchyCompletionOperator Real-seal nonescape)**.** *Let T be any Real-facing consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier* $O = (M, B, U, S, R, D, Q, E, H, C, P, N)$. *If T reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, then the read first factors through the completion-boundary row B, the finite* $\mathsf{StreamName}^{\uparrow}$ *window S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the displayed transport, replay, provenance, and local naming rows H, C, P, N.*

The terminal seal therefore exports only the displayed finite operator packet. It does not expose Real completeness, quotient limits, selected representatives, choice, host equality, or an ambient completion endpoint.

Proof. Begin with Definition 3379.5. It isolates the finite window and tolerance ledger that every terminal consumer must pass through before Q or E can be inspected.

Next apply Definition 3379.3. The Real-facing side of the handoff receives E only after B, S, R, D and the structural rows have already fixed the displayed route.

Finally use Definition 3379.6. Since the packet exports no quotient completion equality, selected representative, selected limit, choice schedule, host completion operator, or ambient Real-completeness coordinate, the terminal Real seal cannot escape the finite operator packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorRealSealNonescape`

Theorem 2426.8 (CauchyCompletionOperator obligation surface)**.** *Every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

exposes exactly the following obligation surface: the $\mathsf{MetricCompletion}^{\uparrow}$ *row* M*, the* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *row* B*, the* $\mathsf{UniformCompletion}^{\uparrow}$ *row* U*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *handoff* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$ *row* N*.*

The surface refuses quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission and local naming come from Definition 3379.2. That theorem projects exactly the twelve displayed rows $M, B, U, S, R, D, Q, E, H, C, P, N$ from the accepted carrier and assigns each row its local NameCert role.

The completion-facing order is supplied by Definition 3379.3. It places the metric and uniform source rows before the boundary row, then routes every later completion read through the finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming rows.

The finite inspection window is fixed by Definition 3379.5. It prevents Q or E from being consumed before B, S, R, D, H, C, P, N have been displayed.

Finally apply Definition 3379.6 and Definition 3379.7. These two nonescape clauses exclude quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient $\mathsf{Real}^{\uparrow}$ completeness from both the ledger and the terminal Real seal. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorPublicObligationSurface`

Theorem 2426.9 (CauchyCompletionOperator obligation triad)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N),$$

the local NameCert obligations decompose exactly into the following three faces:

1. *finite-window extraction, reading the completion-boundary row* B*, StreamName window* S*, RegSeqRat readback* R*, DyadicRatCore ledger* D*, and structural rows* H, C, P, N *before any terminal read;*
2. *functorial handoff, routing the metric and uniform source rows* M, U *through* B*, then through* S, R, D*, and only then to* Q, E, H, C, P, N*;*
3. *ledger nonescape, requiring every separated-limit or Real-facing consumer to read only the displayed rows* $M, B, U, S, R, D, Q, E, H, C, P, N$ *and no quotient completion equality, selected representative, selected limit, countable-choice schedule, host completion operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorObligationTriad`

Proof. The finite-window face is exactly Definition 3379.5. It isolates B, S, R, D, H, C, P, N as the finite evidence a consumer must inspect before Q or E is available.

The handoff face is exactly Definition 3379.3. It places M, U on the source side, routes them through the boundary row B, and sends every completion-facing consumer through the common finite window before the separated-limit and Real-seal rows.

The nonescape face combines Definition 3379.6 with Definition 3379.7. The first theorem prevents any consumer from leaving the displayed ledger, and the second theorem applies the same refusal to terminal Real-seal reads. Together they exhaust the obligation surface recorded in Definition 3379.8. □

Theorem 2426.10 (CauchyCompletionOperator obligation window route)**.** *Every accepted* CauchyCompletionOper *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

satisfies the completion-operator obligation sequence

$$M, U \longrightarrow B \longrightarrow S, R, D \longrightarrow Q, E \longrightarrow H, C, P, N.$$

The metric and uniform source rows M, U are read first; the completion-boundary row B mediates the two source reads; the rows S, R, D expose the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows; the rows Q, E are terminal separated-limit and Real-seal consumers; and the rows H, C, P, N only transport, replay, source, and name the same packet.*

Proof. Start from the carrier inventory in Definition 3379.1. An accepted packet has exactly the displayed metric source, completion boundary, uniform source, finite window, regular readback, dyadic tolerance, separated-limit, Real-seal, transport, replay, provenance, and naming rows.

Apply Definition 3379.2. The local NameCert surface reads the metric and uniform source rows as obligations of the same carrier, and the boundary row B is the displayed mediator between those source reads and every later completion-facing consumer.

Use Definition 3379.5. Before either terminal row is available, the operator exposes the finite $\mathsf{StreamName}^{\uparrow}$ schedule S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, together with the structural rows that keep the extraction attached to the same packet.

Finally invoke Definition 3379.7. The terminal Real read can only occur after the boundary, finite window, readback, dyadic ledger, and structural rows have fixed the route. The separated-limit row Q is consumed on the same downstream side of the packet, while H, C, P, N merely transport, replay, source, and name what the carrier already displays. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorObligationWindowRoute`

Theorem 2426.11 (CauchyCompletionOperator universal factorization)**.** *Every completion-universal or functorial completion consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

first reads the finite route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

If the same consumer also cites a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row or a* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row, that row is admitted only as a downstream read of the same boundary, finite-window, separated-limit, Real-seal, transport, replay, provenance, and local naming route. The consumer receives no host universal property, quotient completion, selected representative, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness shortcut around O.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorUniversalFactorization`

Proof. Start with a completion-universal or functorial completion consumer of O. By Definition 3379.3, the consumer already reads the metric and uniform source rows through the displayed boundary row before it can inspect finite windows, regular readback, dyadic tolerances, separated-limit handoff, Real seal, transport, replay, provenance, or local naming.

Next apply Definition 3379.10. It fixes the same order as an obligation sequence: M, U precede B; B precedes S, R, D; the terminal rows Q, E are downstream of those finite observations; and H, C, P, N only preserve and name the displayed packet.

Now consider an additional $\mathsf{CompletionUniversalProperty}^{\uparrow}$ or $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ citation. Such a citation can be read only after the rows needed by the operator route have been displayed, because the operator exports completion-facing data solely through the boundary and window route just identified. The cited universal or functorial row is therefore a later consumer of O, not an independent source of completion evidence.

Finally use Definition 3379.6 and Definition 3379.7. These exclude quotient completion equality, selected representatives, selected limits, countable-choice schedules, host completion operators, ambient Real completeness, and terminal-seal escapes from the packet. Hence no universal-property or functorial completion consumer can bypass the finite CauchyCompletionOperator route. □

Theorem 2426.12 (CauchyCompletionOperator separated-limit factorization)**.** *Every* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *consumer reading row Q from an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

must first read the finite route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto H, C, P, N.$$

Thus Q is a downstream separated-limit handoff of the displayed finite-window route, not an independent completion witness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorSeparatedLimitFactorization`

Proof. Start with the obligation surface of Definition 3379.2. It exposes Q only as one row of the accepted carrier, after the metric source, completion-boundary, uniform source, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance rows have already been displayed.

Apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. These two results force the separated-limit consumer through M, U, then B, then S, R, D, before the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same packet.

Finally use ledger nonescape, Definition 3379.6. Since quotient completion equality, selected limits, host completion operators, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, row Q cannot supply an independent completion object. It is only the separated-limit handoff downstream of the finite StreamName, RegSeqRat, and DyadicRatCore route. □

Theorem 2426.13 (CauchyCompletionOperator idempotent complete carrier)**.** *Let $O = (M, B, U, S, R, D, Q, E, H$ be an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet whose source and target completion rows are already separated, and whose transport row is the identity replay over the displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then any second application of the completion operator factors through the same finite observation windows, the same regular rational readback, the same dyadic tolerance ledger, and the same terminal Real seal.*

Thus the second application is read as an idempotent complete-carrier replay of O, not as a new quotient stream equality, selected limit, countable-choice schedule, host completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorIdempotentCompleteCarrier`

Proof. Start from the carrier of Definition 3379.1. The only rows available to either completion read are the metric and uniform source rows, the boundary row, the StreamName window, the RegSeqRat readback, the DyadicRatCore ledger, the separated-limit handoff, the Real seal, and the structural replay rows.

Apply the universal factorization theorem, Definition 3379.11. It forces every completion-universal or functorial consumer of the first application through $M, U, B, S, R, D, Q, E, H, C, P, N$ before any completion-facing handoff is available.

Next apply separated-limit factorization, Definition 3379.12. Since the source and target completion rows are already separated and the transport row is the identity replay over the displayed limit-seal, window, readback, dyadic, and Real rows, the second application has no additional separated-limit coordinate to inspect.

Finally use the scoped obligation closure of Definition 3379.14. The closed scope already exhausts carrier admission, finite-window routing, terminal Real sealing, transport, replay, provenance, and local naming. The repeated operator read therefore reuses the same finite observation windows and terminal seal without introducing quotient stream equality, a selected limit, countable choice, host completion data, or ambient Real completeness. □

Theorem 2426.14 (CauchyCompletionOperator scoped obligation closure)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N),$$

the metric and uniform source rows, completion-boundary row, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, separated-limit handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *rows jointly discharge the* obligation *surface for the operator. Every scope is routed through the displayed BEDC primitives and sibling chapters, and no scope is supplied by quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorScopedObligationClosure`

Proof. Start with the obligation surface and functorial handoff, Definition 3379.2 and Definition 3379.3. These expose the metric source, uniform source, completion boundary, finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, and structural rows in the displayed carrier order.

Apply finite-window extraction, ledger nonescape, and Real-seal nonescape from Definition 3379.5, Definition 3379.6, and Definition 3379.7. They force terminal separated-limit and Real-facing reads through the same finite window and prevent escape to quotient, choice, host, or ambient-completeness data.

Finally use the obligation triad, obligation-window route, universal factorization, and separated-limit factorization, Definition 3379.9, Definition 3379.10, Definition 3379.11, and Definition 3379.12. These rows exhaust carrier admission, source routing, terminal handoff, transport, replay, provenance, and local naming, so the scoped obligation surface is closed exactly at obligation. □

Theorem 2426.15 (CauchyCompletionOperator composition uniqueness route)**.** *Let O_0 and O_1 be accepted* CauchyCompletionOperator$^\uparrow$ *packets with the same* MetricCompletion$^\uparrow$, CauchyCompletionBoundary$^\uparrow$, UniformCompletion$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, CompleteSeparatedMetric$^\uparrow$, Real$^\uparrow$, hsame, Cont, Pkg, *and* NameCert *rows. Then every completion-facing composition read induced by O_0 follows the same finite operator route as the read induced by O_1.*

Equivalently, agreement of the displayed rows forces agreement of the completion-facing handoff route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N,$$

without asserting quotient completion equality, selected limits, host completion operators, countable choice, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCompositionUniquenessRoute`

Proof. Start with the functorial handoff of Definition 3379.3. It forces any composition-facing read of either packet to pass through the same metric source, uniform source, completion boundary, finite window, regular readback, dyadic tolerance ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming rows.

Next apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. Since O_0 and O_1 agree on the displayed StreamName, RegSeqRat, DyadicRatCore, boundary, and structural rows, both composition reads expose the same finite observation window before Q or E is available.

Use universal factorization and separated-limit factorization, Definition 3379.11 and Definition 3379.12. These theorems route the universal, functorial, and separated-limit consumers through the already displayed packet rows, so no second completion-facing route can be generated from data outside the common row family.

Finally invoke scoped obligation closure, Definition 3379.14. The closed scope exhausts the admissible carrier, handoff, window, seal, transport, replay, provenance, and naming obligations. Therefore two accepted packets with the same displayed rows induce the same completion-facing route, while quotient completion equality, selected limits, host completion operators, countable choice, and ambient Real$^\uparrow$ completeness remain outside the claim. □

Theorem 2426.16 (CauchyCompletionOperator public export)**.** *The public* CauchyCompletionOperator$^\uparrow$ *export is the finite route*

$$M, B, U, S, R, D, Q, E, H, C, P, N$$

from metric and uniform completion requests through StreamName$^\uparrow$ *windows,* RegSeqRat$^\uparrow$ *readback,* DyadicRatCore$^\uparrow$ *tolerances,* CompleteSeparatedMetric$^\uparrow$ *handoff, and the terminal* Real$^\uparrow$ *seal. Every downstream completion consumer must cite this displayed route before using the operator, rather than quotient completion equality, host completion equality, a selected representative, a selected limit, countable choice, or ambient* Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorPublicExport`

Proof. Start with the obligation surface and functorial handoff, Definition 3379.2 and Definition 3379.3. They expose the source metric row, boundary row, uniform row, finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, and structural rows in the displayed carrier order.

Apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. These theorems force every completion-facing read through S, R, D before either the separated handoff Q or terminal seal E is available.

Use ledger nonescape, universal factorization, and separated-limit factorization, Definition 3379.6, Definition 3379.11, and Definition 3379.12. They keep the operator inside the finite dyadic and separated route, so no quotient, selected limit, host completion, countable-choice, or ambient-completeness coordinate can be read as a public export.

Finally invoke scoped obligation closure, Definition 3379.14. Since the scoped carrier surface already exhausts admission, finite-window routing, terminal Real sealing, transport, replay, provenance, and local naming, the public export is exactly the displayed finite route for downstream completion consumers. □

Theorem 2426.17 (Complete metric limit uniqueness outbound reference)**.** *Every public* CauchyCompletionOperat *export that later asserts uniqueness of a separated complete-metric limit must consume chapter 5983 after the public operator route of Definition 3379.16. The operator route exports the finite completion handoff only; it exports no selected limit, quotient representative, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the public export theorem. It binds every downstream completion consumer to the displayed route through $M, B, U, S, R, D, Q, E, H, C, P, N$ before the operator can be used.

A later uniqueness assertion is a separate separated complete-metric comparison. It therefore consumes the sibling route chapter 5983 after the operator export has supplied its completion-facing handoff.

Finally apply the operator nonescape clauses already used in the public export proof. They exclude selected limits, quotient representatives, countable-choice schedules, host completion objects, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the operator route, so uniqueness is imported only from the downstream sibling and not from an extra coordinate of $\mathsf{CauchyCompletionOperator}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCompleteMetricLimitUniquenessOu`

Theorem 2426.18 (CauchyCompletionOperator bridge route)**.** *Every bridge-facing consumer of* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *must read the displayed operator through the functorial handoff, finite-window extraction, universal factorization, separated-limit factorization, and public export rows before treating it as a standard completion bridge. Concretely, the consumer first reads the metric and uniform source rows through* M, U *and the completion-boundary row* B*; then it reads the finite window, regular readback, and dyadic ledger rows* S, R, D*; then it reads the universal and separated-limit handoffs through* Q *and the terminal Real seal* E*; and only then may it replay transport, provenance, and local naming through* H, C, P, N*.*

This route is a bridge-readiness obligation on the paper surface. It does not assert bridgeChecked*, does not replace the existing public closure status, and does not add quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the functorial handoff, Definition 3379.3. It fixes the order in which metric and uniform source rows are routed through the displayed completion-boundary row before any terminal completion consumer is available.

Apply finite-window extraction, Definition 3379.5. This forces a bridge-facing consumer to inspect the $\mathsf{StreamName}^{\uparrow}$ window S, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and structural rows H, C, P, N before reading Q or E.

Use universal factorization and separated-limit factorization, Definition 3379.11 and Definition 3379.12. These theorems keep completion-universal, functorial, and separated-limit consumers inside the same displayed route.

Finally apply public export, Definition 3379.16. It binds downstream completion consumers to the finite public route. Since bridge checking is not claimed by that theorem or by the closurestatus block below, the two axes remain separate. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorBridgeRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionOperator}^{\uparrow}$ packet is generated from MetricCompletion, CauchyCompletionBoundary, UniformCompletion, StreamName, RegSeqRat, DyadicRatCore, CompleteSeparatedMetric, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 3379.14 closes the scoped obligation surface over $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompleteSeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows already exhibited by Definition 3379.1, Definition 3379.2, Definition 3379.5, Definition 3379.9, Definition 3379.10, Definition 3379.11, Definition 3379.12, and Definition 3379.16; the public export theorem binds downstream completion consumers to the same finite route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No paper-side theory upgrade remains; bridge checking remains a separate formal-axis obligation unless a standard bridge theorem is supplied.

Chapter 2427

A Concrete Naming Certificate for $\mathsf{RealCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealCauchyCompleteness}^{\uparrow}$ packet records the classical regular-Cauchy completeness route as finite BEDC NameCert data. It sits over the dyadic radius ledger of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the Cauchy-limit seal of chapter 386, and the terminal real-name seal of chapter 38. The packet exposes a displayed completion handoff for regular Cauchy streams; it does not assert that $\mathsf{Real}^{\uparrow}$ already carries ambient completeness.

Definition 2427.1 (Real Cauchy completeness carrier)**.** *A* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule of finite windows used to inspect the Cauchy stream.*
2. *M is the explicit Cauchy-modulus row, exposing the threshold at which requested dyadic tolerances are met.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for positive radii, halving steps, comparison budgets, and tolerance addition.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback assembled from the retained windows and the modulus row.*
5. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row consuming the same S, M, D, R route before any real-facing read is exported.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the limit-seal handoff.*
7. *H records componentwise* hsame *transport for $S, M, D, R, L, E, C, P, N$.*
8. *C records displayed* Cont *replay of the diagonal route*

$$S, M, D \longmapsto R \longmapsto L \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyCompleteness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *row.*
The classifier accepts only packets that display the finite windows, explicit modulus, dyadic budgets, diagonal readback, limit-seal handoff, real seal, transport, replay, provenance, and local naming. It has no coordinate for quotient streams, selected limits, countable choice, open-cover compactness, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletenessUp.RealCauchyCompletenessTasteGate single carrier alignment`

Theorem 2427.2 (Real Cauchy completeness NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier* $K = (S, M, D, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream windows, explicit Cauchy modulus, dyadic tolerance ledger, diagonal regular readback, CauchyLimitSeal handoff, Real seal, componentwise transport, displayed replay, provenance, and local naming. The completion-facing read is accepted only when the modulus row M is exposed before the diagonal readback R, the handoff row L consumes the same S, M, D, R route, and every* $\mathsf{Real}^{\uparrow}$ *consumer reaches E through that displayed route.*

Proof. Carrier admission is projection from Definition 3380.1. The rows S, M, D display the finite stream windows, the explicit Cauchy modulus, and the dyadic tolerance ledger before any completion-facing read is available.

The diagonal limit handoff is then assembled in R. The row R reads the retained windows through the modulus thresholds and dyadic budgets, so the construction is a displayed regular rational readback rather than a quotient of streams or a selected host limit.

Classifier stability follows from H and C. Componentwise hsame transport preserves the same window, modulus, dyadic, readback, seal, provenance, and naming rows, while displayed Cont replay keeps the route ordered as $S, M, D \to R \to L \to E$.

The ledger rows P, N expose no extra coordinate. Since quotient streams, selected limits, countable choice, open-cover compactness, host convergence, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, every accepted completion read remains inside the displayed finite route. □

Theorem 2427.3 (Real Cauchy completeness seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The read therefore supplies a finite Cauchy-completeness handoff to real-name consumers without adding quotient stream equality, a selected limit, countable choice, or the claim that $\mathsf{Real}^{\uparrow}$ *is already complete.*

Proof. Start with an accepted carrier $K = (S, M, D, R, L, E, H, C, P, N)$. By the carrier definition, the only real-facing endpoint is E, and E is downstream of the limit-seal row L.

The row L consumes the diagonal readback R. That readback is itself downstream of the finite stream schedule S, the explicit modulus M, and the dyadic ledger D.

Thus any $\mathsf{Real}^{\uparrow}$ consumer reaches the seal only through S, M, D, R, L, E and the structural rows H, C, P, N. The excluded host-side data have no carrier coordinate, so they cannot be introduced by the handoff. □

Theorem 2427.4 (Real Cauchy completeness uniform-tail seal)**.** *Let* $K = (S, M, D, R, L, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier. Suppose a consumer reads K through a* $\mathsf{UniformConvergence}^{\uparrow}$

shared threshold whose tail schedule dominates the explicit Cauchy-modulus row M and whose Cauchy-facing handoff is served by the $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ *route of chapter 3876. Then every* $\mathsf{Real}^{\uparrow}$ *endpoint read factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, the* $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ *handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L before the Real seal E is exposed.*

Proof. Project the carrier rows from Definition 3380.1. The only endpoint row is E, and Definition 3380.3 places E downstream of S, M, D, R, L.

The shared threshold is read through the $\mathsf{UniformConvergence}^{\uparrow}$ tail surface of chapter 3417. The dominance assumption says that this tail schedule cannot be consumed before it covers the explicit modulus row M already present in K.

The Cauchy-facing handoff is then the $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ route. It binds the same uniform tail row to a finite Cauchy comparison before the regular readback R and the limit-seal row L are available to the endpoint consumer.

Thus the row order is S, M, D first, then the shared uniform tail and Cauchy-criterion handoff, then R, L, E. Since the carrier has no quotient coordinate, choice coordinate, or second endpoint row, every $\mathsf{Real}^{\uparrow}$ read follows this displayed route. □

Theorem 2427.5 (Real Cauchy completeness filtered minimality)**.** *Let $K = (S, M, D, R, L, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier. Every filtered or completion-service consumer that reads K through the* $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ *route of chapter 8666 or the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *service boundary of chapter 1606 must preserve exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S, explicit modulus row M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L,* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *row P, and local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *row N before exporting a completion-facing Real read. Replacing this route by quotient streams, selected limits, countable choice, host convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is outside the carrier.*

Proof. Project the inventory of rows from Definition 3380.1. It gives the consumer exactly $S, M, D, R, L, E, H, C, P,$ with no coordinate for quotient stream equality, a chosen limit, countable choice, host convergence, or an ambient completeness theorem.

The terminal Real read is constrained by Definition 3380.3. Any endpoint must pass through the displayed stream windows, dyadic tolerance ledger, regular readback, and limit-seal handoff before reaching E.

For filtered consumers, Definition 8666.3 supplies the finite synchronization principle. Its window induction reads the filter and regular Cauchy rows against the same dyadic tolerances and preserves the shared transport, replay, provenance, and local naming rows, so the filtered route cannot bypass $S, M, D, R, L, E, H, C, P, N$.

For completion-service consumers, the $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ chapter records the least displayed service boundary for completion reads, with Lean touchpoint

BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityTasteGate_single_carrier_alignm

Reading K through that boundary therefore preserves the Real Cauchy route as the service-facing input rather than adding an external completion principle. □

Falsifiable prediction: Any accepted $\mathsf{RealCauchyCompleteness}^{\uparrow}$ consumer reaching a $\mathsf{Real}^{\uparrow}$ read without the displayed StreamName windows, explicit Cauchy modulus, DyadicRatCore ledger, RegSeqRat diagonal readback, and CauchyLimitSeal row refutes the carrier.

Independence witness: DyadicRatCore, StreamName, RegSeqRat, CauchyLimitSeal, and Real are not individually bijective with $\mathsf{RealCauchyCompleteness}^{\uparrow}$ because the packet carries the ordered completion route across all five surfaces together with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyCompleteness}^{\uparrow}$ packet is generated from finite StreamName windows, an explicit Cauchy-modulus row, a DyadicRatCore tolerance ledger, a RegSeqRat diagonal readback, a CauchyLimitSeal handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3380.1, Definition 3380.2, Definition 3380.3, Definition 3380.4, and Definition 3380.5 seed the finite constructive Cauchy-completeness route through StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, UniformConvergence, UniformConvergenceCauchyCriterion, filtered completion, minimal completion service, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealCauchyCompletenessUp.RealCauchyCompletenessTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, host convergence, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, explicit Cauchy-modulus exposure, diagonal limit handoff, classifier stability, and ledger non-escape for BEDC.Derived.RealCauchyCompletenessUp.

Chapter 2428

A Concrete Naming Certificate for ClosedLamDomainSubjectReduction$^{\uparrow}$

The ClosedLamDomainSubjectReduction$^{\uparrow}$ packet realizes the lambda-domain branch of the closed MetaCIC subject-reduction frontier as a finite NameCert carrier. It sits beside the four-row discharge socket of chapter 719, the discharge ledger of chapter 720, the beta-substitution packet of chapter 716, the closed substitution boundary of chapter 538, the typed beta congruence trigger of chapter 760, and the Pi-domain sibling of chapter 3352. The packet names the lambda-domain obligation as displayed closed term data; it does not export an unconditional closed subject-reduction theorem.

Definition 2428.1 (ClosedLamDomainSubjectReduction carrier)**.** *A* ClosedLamDomainSubjectReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, D, D', B, Y, Z, G, H, R, Q, N)$$

with the following displayed rows:

1. *T is the displayed closed lambda term row, naming the source term $\lambda D.\, B$ and its closed context boundary.*
2. *D is the original domain row inspected before the congruence step.*
3. *D' is the reduced domain row reached by the displayed beta step from D.*
4. *B is the body row, read under the binder induced by the displayed domain.*
5. *Y is the typing row, recording the source typing judgement for $\lambda D.\, B$ and the target typing judgement demanded for $\lambda D'.\, B$.*
6. *Z is the closedness row, requiring the lambda term, original domain, reduced domain, and body reads to remain inside the same closed MetaCIC boundary.*
7. *G is the beta-step and typed congruence row, consuming the displayed lambda-domain beta step and the trigger surface of chapter 760.*
8. *H records componentwise* hsame *transport for $T, D, D', B, Y, Z, G, R, Q, N$.*
9. *R records displayed* Cont *replay from the source lambda term through the domain step, typing row, closedness row, and subject-reduction handoff.*

10. *Q is the* Pkg *provenance over chapter 719, chapter 720, chapter 716, chapter 538, chapter 760, chapter 3352,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying the closed lambda term, both domain rows, the body row, the typing row, the closedness row, the lambda-domain beta step, transport, replay, provenance, and local naming. It has no coordinate for a blanket closed subject-reduction theorem, a host conversion rule, erased derivation semantics, quotient syntax equality, or an untracked binder context.

Theorem 2428.2 (ClosedLamDomainSubjectReduction obligations)**.** *For every accepted* ClosedLamDomainSubject *carrier $L = (T, D, D', B, Y, Z, G, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed closed lambda term, original domain, reduced domain, body row, typing row, closedness row, beta-step row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The lambda-domain preservation read is accepted only when G displays the domain beta step and Y, Z display the corresponding typing and closedness rows.*

Proof. Project the carrier of Definition 3381.1. The rows T, D, D', B display the lambda term, the two domain readings, and the body read, while Y and Z display the typing and closedness boundaries attached to the same context.

The beta-step row G is then read as the lambda-domain congruence trigger. It is not interchangeable with the beta-substitution row, the application-argument row, or the Pi-domain row of the subject-reduction socket, because the target branch changes the displayed lambda domain while preserving the body under the induced binder.

The structural rows H, R, Q, N transport, replay, source, and name only that finite packet. Since no row stores a host conversion rule, erased derivation semantics, quotient syntax equality, or a closed subject-reduction theorem, the displayed rows exhaust the local NameCert obligations. □

Theorem 2428.3 (ClosedLamDomainSubjectReduction handoff)**.** *Every subject-reduction discharge ledger read that consumes the lambda-domain branch from an accepted* $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ *packet factors through the displayed route*

$$\lambda D.\, B,\ D,\ D',\ B,\ Y,\ Z,\ G,\ H,\ R,\ Q,\ N.$$

The handoff supplies the lambda-domain obligation row to the subject-reduction discharge socket and ledger; it does not supply the beta-substitution row, application-argument row, Pi-domain row, or an unconditional closed subject-reduction theorem.

Proof. Start with a subject-reduction consumer of the lambda-domain branch. The socket and ledger chapters, chapter 719 and chapter 720, require a distinct lambda-domain row before the parameterised subject-reduction route can be read.

The present carrier supplies that row only through T, D, D', B, Y, Z, G. These rows display the source lambda term, the original and reduced domains, the body, the typing judgement, the closedness boundary, and the beta-step trigger.

Finally H, R, Q, N preserve and name the same finite handoff. The beta-substitution preservation packet, closed substitution boundary, typed beta congruence trigger, and Pi-domain packet remain sibling inputs rather than hidden substitute proofs, so the handoff is exactly the lambda-domain row and nothing stronger. □

Falsifiable prediction: Any accepted $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ consumer reaching a subject-reduction conclusion without the displayed lambda-domain beta step and typing rows refutes the carrier.

Independence witness: ClosedPiDomainSubjectReduction, BetaSubstitutionPreservation, ClosedSubstitutionBoundary, and TypedBetaCongruenceTrigger are not bijective with $\mathsf{ClosedLamDomainSubjectReduction}$ because this packet jointly carries the closed lambda term, original domain, reduced domain, body row, typing row, closedness row, lambda-domain beta step, transport, replay, provenance, and local naming.

The Lean TasteGate file for $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ClosedLamDom}$ and $\mathsf{FieldFaithful\ ClosedLamDomainSubjectReduction}^{\uparrow}$ instances preserving $T, D, D', B, Y, Z, G, H, R, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ packet is generated from a closed lambda term row, original domain row, reduced domain row, body row, typing row, closedness row, beta-step row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3381.1, Definition 3381.2, and Definition 3381.3 seed the finite lambda-domain subject-reduction obligation packet for the closed MetaCIC discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta-substitution preservation, application-argument stability, Pi-domain preservation, unconditional closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lambda-domain beta-step exactness, typing-row exposure, closedness-row exposure, subject-reduction ledger handoff, and TasteGate field faithfulness for BEDC.Derived.ClosedLamDomainSub

Chapter 2429

A Concrete Naming Certificate for CauchyCompletenessCriterion$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyCompletenessCriterion$^{\uparrow}$ packet records the finite Cauchy-criterion route by which a metric source, an explicit Cauchy modulus, named stream windows, regular rational readback, dyadic tolerance accounting, and a terminal Real$^{\uparrow}$ seal are presented to completion-facing consumers. It sits over the metric completion surface of chapter 1506, the regular rational rows of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet gives later completion chapters a single criterion object to cite instead of restating the Cauchy-modulus handoff locally.

Definition 2429.1 (CauchyCompletenessCriterion carrier)**.** *A* CauchyCompletenessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, \mu, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, carrying the named carrier, distance observations, and completion-facing request inherited by the metric completion route.*
2. *μ is the explicit Cauchy modulus row, displayed as finite threshold data for each inspected tolerance request.*
3. *S is the* StreamName$^{\uparrow}$ *window family naming the finite tail observations read from the source stream and the modulus thresholds.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from those windows.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger recording the rational budget, tail slack, and comparison bounds used by the readback.*
6. *E is the* Real$^{\uparrow}$ *seal row, visible only after M, μ, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, modulus, windows, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing Cauchy request through $M, \mu, S, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* CauchyCompletenessCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the ten displayed rows. It has no coordinate for quotient completions, selected limits, countable-choice schedules, trichotomy decisions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessCriterionUp.CauchyCompletenessCriterionTasteGate single carrier alignment`

Theorem 2429.2 (CauchyCompletenessCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletenessC}$ *carrier* $Q = (M, \mu, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, explicit Cauchy modulus, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must consume* M, μ, S, R, D *before it can inspect* E, H, C, P, N*.*

Proof. Project the carrier of Definition 3382.1. The rows M and μ expose the metric source and the Cauchy modulus as finite packet data.

Read the observation rows next. The row S stores exactly the inspected tail windows, R records the regular rational readback from those windows, and D stores the dyadic tolerance budget attached to the same finite observations.

Finally inspect the seal and structural rows. Since E, H, C, P, N transport, replay, source, and name the already displayed route, the certificate has no remaining coordinate from which a quotient completion, selected limit, countable-choice schedule, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem could be exported. □

Theorem 2429.3 (CauchyCompletenessCriterion Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or metric-completion consumer citing an accepted* $\mathsf{CauchyCompletenessCriterion}^{\uparrow}$ *packet receives only the finite route*

$$M, \mu, S, R, D, E, H, C, P, N.$$

The handoff first displays the metric source and Cauchy modulus, then the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then the* $\mathsf{Real}^{\uparrow}$ *seal together with transport, replay, provenance, and local naming.*

Proof. Begin with an accepted carrier and apply Definition 3382.2. The consumer-facing data are exhausted by the displayed rows $M, \mu, S, R, D, E, H, C, P, N$.

The criterion route is ordered. A completion-facing read must first expose the metric source and explicit modulus, then read finite stream windows and regular rational observations, and then consume the dyadic tolerance ledger.

The Real seal is terminal with respect to this route. Because quotient completions, selected limits, countable-choice schedules, trichotomy decisions, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, every exported handoff is the displayed finite criterion packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessCriterion}^{\uparrow}$ packet is generated from a metric source row, explicit Cauchy modulus, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3382.1, Definition 3382.2, and Definition 3382.3 seed the Bishop-style Cauchy criterion handoff for Real and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyCompletenessCriterionUp.CauchyCompletenessCriterionTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, trichotomy, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, finite window coverage, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyCompletenessCriterionUp.

Chapter 2430

A Concrete Naming Certificate for $\mathsf{TailModulusLimitHandoff}^{\uparrow}$

Following the Bishop-style diagonal Cauchy proof route in section 9238.1, the $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ packet records the finite tail-modulus handoff used by the diagonal regularity theorem of Definition 9238.4. It reads the finite windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet gives Real-completion proof targets a concrete landing for the shared tail threshold consumed by diagonal-limit consumers.

Definition 2430.1 (TailModulusLimitHandoff carrier)**.** *A* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\mu, \tau, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. μ *is the Cauchy modulus row for the regular rational sequence of regular rational sequences.*
2. τ *is the finite tail selector row naming the shared threshold used by the diagonal limit read.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected at the selected tail threshold.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the diagonal representative and the two tail observations compared by the proof.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the endpoint legs, middle modulus leg, and rational slack used by the diagonal estimate.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* μ, τ, S, R, D *have been displayed.*
7. H *records componentwise* hsame *transport for the modulus, tail selector, windows, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a diagonal-limit request through* $\mu, \tau, S, R, D, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient streams, selected limits, countable-choice schedules, trichotomy decisions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TailModulusLimitHandoffUp.TailModulusLimitHandoffTasteGate single carrier alignment`

Theorem 2430.2 (TailModulusLimitHandoff NameCert obligations)**.** *For every accepted* TailModulusLimitHandoff *carrier* $T = (\mu, \tau, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy modulus, finite tail selector,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The diagonal-limit consumer must read* μ, τ, S, R, D *before it can export a Real-facing seal through* E, H, C, P, N*.*

Proof. Project the carrier of Definition 3383.1. The rows μ and τ display the modulus and the shared tail threshold used by the diagonal-limit proof.

Read the finite observations. The row S names the inspected windows, R records the regular rational readback, and D stores the dyadic tolerance ledger for the endpoint legs and the middle modulus leg.

The seal and structural rows add no analytic coordinate. The rows E, H, C, P, N seal, transport, replay, source, and name the displayed finite route, so quotient streams, selected limits, countable-choice schedules, trichotomy decisions, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the certificate. □

Theorem 2430.3 (TailModulusLimitHandoff diagonal handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *packet receives the finite diagonal route*

$$\mu, \tau, S, R, D, E, H, C, P, N.$$

The handoff supplies the Cauchy modulus, shared tail selector, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming needed by the diagonal regularity route of Definition 9238.4.*

Proof. Start with a Real-completion consumer. By Definition 3383.2, the consumer first reads the modulus and tail selector, then the finite windows, readback row, and dyadic tolerance ledger.

The diagonal route is therefore finite before the Real seal is visible. The seal row E depends on the displayed rows μ, τ, S, R, D, and the structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming.

Since no carrier coordinate stores quotient streams, selected limits, countable-choice schedules, trichotomy decisions, or ambient completeness, every exported diagonal handoff is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ packet is generated from a Cauchy modulus row, finite tail selector, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 3383.1, Definition 3383.2, and Definition 3383.3 seed the finite tail-modulus handoff used by the diagonal regularity proof route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.TailModulusLimitHandoffUp.TailModulusLimitHandoffTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail selector exactness, finite window coverage, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.TailModulusLimitHandoffUp.

Chapter 2431

A Concrete Naming Certificate for $\mathsf{BorelSet}^{\uparrow}$

Following Bourbaki and Bishop–Bridges topology, the $\mathsf{BorelSet}^{\uparrow}$ packet names the finite boundary by which a topological or metric real-facing source exposes a generated set predicate without importing measure theory. It sits after the open-neighbourhood interface of chapter 88, the metric source of chapter 57, the terminal real-name seal of chapter 38, the finite observation route of chapter 126, and the regular rational readback surface of chapter 301. The packet records generator admission, finite constructor replay, and sibling provenance; it does not claim a measure, sigma-additivity, choice-selected codes, or classical completeness of the full Borel hierarchy.

Definition 2431.1 (Borel set carrier)**.** *A* $\mathsf{BorelSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, O, M, G, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-space row read either through the* $\mathsf{Topology}^{\uparrow}$ *interface of chapter 88 or through a metric presentation supplied by chapter 57.*
2. *O is the open-generator ledger naming basic open rows visible to the source topology.*
3. *M is the real-facing membership query surface, routed through* $\mathsf{Real}^{\uparrow}$ *names from chapter 38 when the source is a real or metric-real object.*
4. *G is the finite generator tree recording the displayed complements, finite unions, and finite intersections that the current consumer is allowed to replay.*
5. *R is the readback row tying finite membership observations to* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational names through chapter 126 and chapter 301.*
6. *H stores componentwise* hsame *transport for the source, generator, membership, readback, continuation, provenance, and local naming rows.*
7. *C stores the displayed* Cont *replay from a source query through open generators, finite constructor rows, and readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, O, M, G, R, H, C, P, N$. It has no coordinate for measure values, countable additivity, a choice-selected code for an arbitrary hierarchy branch, or a completed classical Borel universe.

Lean4 ✓ `BEDC.Derived.BorelSetUp.BorelSetTasteGate single carrier alignment`

Theorem 2431.2 (Borel set NameCert obligations)**.** *For the carrier of Definition 3384.1, the local* $\mathsf{NameCert}_{\mathsf{BorelSet}^{\uparrow}}$ *surface consists exactly of source-space admission, open-generator admission, real-facing membership queries when the source is real or metric-real, finite generator-tree replay, StreamName and RegSeqRat readback, componentwise transport, displayed continuation replay, package provenance, and local naming. Every accepted Borel-set query must read the source topology or metric row before it reads O, G, R, H, C, P, N.*

Proof. Project the carrier $B = (X, O, M, G, R, H, C, P, N)$. The source row X and open-generator ledger O are the only rows that introduce topological information.

Read the finite constructor tree. Complements, finite unions, and finite intersections are replayed only through G, and the membership-facing observations are tied to M and R before a consumer can compare two packets.

The structural rows add no measure-theoretic content. The rows H, C, P, N transport, replay, source, and name the displayed route, so the listed rows exhaust the seed NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.BorelSetUp.BorelSetCarrier namecert obligations`

Theorem 2431.3 (Borel set generator stability)**.** *Let $B = (X, O, M, G, R, H, C, P, N)$ and*

$$B' = (X', O', M', G', R', H', C', P', N')$$

be accepted $\mathsf{BorelSet}^{\uparrow}$ *carriers. If the displayed source row, open-generator ledger, finite generator tree, membership surface, and readback row agree componentwise through the packet's* hsame *rows, then every topology-facing, metric-facing, or real-facing consumer sees the same generated set boundary from B and B'. The comparison uses only the rows $X, O, M, G, R, H, C, P, N$ and $X', O', M', G', R', H', C', P', N'$.*

Proof. Start with the NameCert obligations. Any consumer read first passes through the source-space row and open-generator ledger, then through the finite generator tree and readback rows.

Apply the displayed componentwise transports. The hypotheses identify the rows that determine the generator route, while H', C', P', N' preserve, replay, source, and name the same finite route for B'.

Thus the consumer-visible boundary is stable under the stated transport. No measure value, countable-additivity proof, hierarchy-completeness object, or choice-selected branch can refine the comparison, because none is a coordinate of either carrier. □

Falsifiable prediction: A $\mathsf{BorelSet}^{\uparrow}$ consumer that reaches a measure value, a countable-additivity row, or a choice-selected hierarchy code from the displayed packet refutes the carrier surface.

Independence witness: topology, metric, real, streamname, regseqrat: no listed sibling supplies the whole nine-row packet $B = (X, O, M, G, R, H, C, P, N)$, because each sibling contributes only source topology, metric control, real-name sealing, finite observation, or rational readback data while this chapter binds the generated-set boundary with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3384.1, Definition 3384.2, and Definition 3384.3 seed the finite generated-set boundary over topology, metric, real-name, StreamName, RegSeqRat, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure theory, sigma-additivity, countable choice, choice-selected Borel codes, classical hierarchy completeness, open-cover compactness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, generator stability, readback preservation, transport, and ledger-refusal proofs for BEDC.Derived.BorelSetUp.

Constructive story: A $\mathsf{BorelSet}^{\uparrow}$ packet is generated from a topology or metric source row, an open-generator ledger, real-facing membership reads, a finite generator tree, StreamName and RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2432

A Concrete Naming Certificate for $\mathsf{OuterMeasure}^{\uparrow}$

Following Caratheodory and Bishop–Bridges constructive measure theory, the $\mathsf{OuterMeasure}^{\uparrow}$ packet records the finite cover-valued boundary from which measurable-event and measure rows can later be read. It sits between the generated-set surface of chapter 3384, the membership discipline of chapter 3668, the real-name endpoint seal of chapter 38, the finite observation windows of chapter 126, and the measure carrier of chapter 92. The packet names only displayed set rows, empty-set valuation, monotonicity comparisons, finite cover subadditivity, Caratheodory-measurability boundary tests, transport, replay, provenance, and local naming; it does not assert a completed measure space, a chosen sigma-algebra closure, countable choice of covers, quotient equality of subsets, or ambient measure completion.

Definition 2432.1 (OuterMeasure carrier)**.** *An* $\mathsf{OuterMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, S, Z, M, F, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the source object whose displayed subsets are read through* $\mathsf{SetLike}^{\uparrow}$ *and, when the source is real-facing, through the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set boundary.*
2. *S is the displayed subset-family row naming the finite set requests visible to the packet.*
3. *Z is the empty-set value row, comparing the outer value of the displayed empty subset with the zero endpoint on the* $\mathsf{Real}^{\uparrow}$ *surface.*
4. *M is the monotonicity row, recording a finite inclusion witness $A \subseteq B$ together with the comparison $\mu^*(A) \leq \mu^*(B)$ on the displayed real endpoints.*
5. *F is the finite displayed cover subadditivity row, assigning a finite cover list $U_0, \ldots, U_k$ of a displayed subset A to the visible comparison*

$$\mu^*(A) \leq \mu^*(U_0) + \cdots + \mu^*(U_k).$$

6. *B is the Caratheodory-measurability boundary row: for each displayed test set T and candidate measurable set E, it records the two-sided split comparison between $\mu^*(T)$ and the displayed sum of the inside and outside rows $T \cap E$ and $T \setminus E$.*

7. *H stores componentwise* hsame *transport for the source row, subset-family row, value rows, inclusion and finite-cover ledgers, boundary row, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a subset request through empty-set value, monotonicity, finite cover subadditivity, boundary testing, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, Z, M, F, B, H, C, P, N$. It has no coordinate for a completed measure, a host sigma-algebra, a selected cover family, quotient subset equality, or a hidden countable-additivity row.

Lean4 ✓ `BEDC.Derived.OuterMeasureUp.OuterMeasureTasteGate single carrier alignment`

Theorem 2432.2 (OuterMeasure NameCert obligations)**.** *For the carrier of Definition 3385.1, the local* $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source and subset-family rows, empty-set zero readback, monotonicity under displayed inclusion, finite displayed cover subadditivity, Caratheodory-measurability boundary testing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted outer-measure consumer must read the source subset through S, expose value comparisons through Z, M, F, B, and replay those rows through H, C, P, N before any* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *sibling row is exposed.*

Proof. Project the carrier $O = (X, S, Z, M, F, B, H, C, P, N)$. The source row X and subset-family row S are the only admitted entry points for displayed set requests, so carrier admission is exhausted by those rows.

Read the value ledgers. The row Z supplies the empty-set zero comparison, M supplies monotonicity under a displayed inclusion witness, and F supplies the finite cover subadditivity comparison for a named finite cover list.

The boundary row B is the only place where a Caratheodory-measurability test is visible. It records the split comparison for displayed test sets and candidates, but it does not choose a completed sigma-algebra or assert ambient measure completion.

Finally replay the structural rows. Componentwise transport through H, continuation replay through C, provenance through P, and local naming through N preserve the same finite route. Since no further coordinate is present, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2432.3 (OuterMeasure finite premeasure route)**.** *Every* $\mathsf{OuterMeasure}^{\uparrow}$ *packet exports a finite pre-measure route*

$$S \longmapsto Z, M, F, B \longmapsto H, C, P, N$$

that downstream $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, *and* $\mathsf{MeasurableFunction}^{\uparrow}$ *consumers may cite before reading measure-completion language. The route exposes empty-set zero, displayed inclusion monotonicity, finite cover subadditivity, and Caratheodory boundary rows, not a host measure space or a choice-selected completion.*

Proof. Start with an accepted subset request in S. The empty, inclusion, finite cover, and boundary rows are the only value-bearing rows available to the packet.

The finite cover row F keeps the route on a displayed list of cover sets, while the boundary row B tests only the named split of a displayed test set by a candidate measurable set. These two rows provide the pre-measure boundary consumed by later measure-facing chapters.

Replay through H, C, P, N. The provenance row records the allowed sibling surfaces, and the continuation row exposes them in the same order. Thus a downstream consumer can cite the finite pre-measure route without importing completed measure language or selected cover choices. □

Falsifiable prediction: An $\mathsf{OuterMeasure}^{\uparrow}$ consumer that obtains a completed measure, a chosen sigma-algebra closure, countable choice of covers, quotient equality of subsets, or a $\mathsf{Measure}^{\uparrow}$ countable-additivity row without first reading $S, Z, M, F, B, H, C, P, N$ refutes the carrier surface.

Independence witness: measure, borelset, setlike, real, streamname: no listed sibling supplies the whole ten-row packet $O = (X, S, Z, M, F, B, H, C, P, N)$, because each sibling contributes only measure values, generated-set membership, set-like membership, real endpoints, or finite observation windows while this chapter binds outer valuation, finite cover subadditivity, boundary testing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3385.1, Definition 3385.2, and Definition 3385.3 seed the finite outer-measure surface over displayed subsets, empty-set value, inclusion monotonicity, finite covers, Caratheodory boundary rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed measure spaces, chosen sigma-algebra closures, countable choice of covers, quotient subset equality, MeasureUp countable additivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, empty-set zero, monotonicity, finite cover subadditivity, Caratheodory boundary testing, transport, provenance, and no-escape for BEDC.Derived.OuterMeasureUp.

Constructive story: An $\mathsf{OuterMeasure}^{\uparrow}$ packet is generated from a displayed source row, subset-family row, empty-set value row, inclusion monotonicity row, finite cover subadditivity row, Caratheodory boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{LocallyConvexSpace}^{\uparrow}$ reads the vector-space, topological-vector, seminormed-space, and functional-analysis sibling rows through chapter 47, chapter 205, chapter 5868, and chapter 94 before the child files route the finite local-convexity carrier.

Chapter 2433

A Concrete Naming Certificate for $\mathsf{LocallyConvexSpace}^{\uparrow}$

Following Bourbaki and Treves on locally convex topological vector spaces, the $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet records the finite BEDC surface where a vector-space carrier, a topological-vector-space carrier, a finite seminorm ledger, and a convex balanced neighbourhood ledger are read before any functional-analysis consumer can cite local convexity. It sits beside the vector-space source of chapter 47, the topological-vector-space source of chapter 205, the norm comparison surface of chapter 89, and the functional-analysis consumer frontier of chapter 94. The packet names the locally convex carrier only through displayed finite rows; it is not a Hahn–Banach principle, not a choice-selected dual, and not an ambient weak-star compactness theorem.

The seminorm and convex-balanced-neighbourhood ledgers consume the sibling rows in a fixed order: chapter 47 supplies the algebraic source, chapter 205 supplies the topological-vector continuity route, chapter 89 supplies the optional norm comparison row, and chapter 94 supplies the downstream consumer boundary. These links keep local convexity as a displayed finite dependency surface rather than an orphan analytic predicate.

Downstream $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumers use $\mathsf{LocallyConvexSpace}^{\uparrow}$ only through the finite seminorm ledger and the convex-balanced-neighbourhood ledger displayed here. The consumer chapter of chapter 94 may cite local convexity only after those two rows have been exposed with the vector-space, topological-vector-space, transport, replay, provenance, and local naming rows.

Definition 2433.1 (Locally convex space carrier)**.** *A* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (V, T, S, B, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, including the additive carrier, scalar action, and vector-space laws used by downstream linear consumers.*
2. *T is the* $\mathsf{TopVecSpace}^{\uparrow}$ *source row, recording the topology and the displayed continuity of addition and scalar multiplication.*
3. *S is the finite seminorm-family ledger, listing the seminorm names, their vector-space source, triangle and scalar rows, and the generated neighbourhood probes accepted in the packet.*
4. *B is the convex balanced neighbourhood ledger, recording the finite neighbourhood rows certified as convex, balanced, and compatible with the topology read through T.*

5. *M is the* $\mathsf{Norm}^{\uparrow}$ *comparison row, used only when a named normed source is present and only as a comparison into the seminorm ledger rather than as a replacement for local convexity.*
6. *H records componentwise* hsame *transport for V, T, S, B, M and the local naming row.*
7. *C records displayed* Cont *replay through the vector source, topological-vector-space source, seminorm ledger, convex balanced neighbourhood ledger, norm comparison, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, T, S, B, M, H, C, P, N$. It contains no coordinate for an arbitrary dual selector, a host locally convex topology predicate, an unrestricted separation theorem, weak-star compactness, quotient completion, or a global choice principle.

Theorem 2433.2 (Locally convex space NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier $L = (V, T, S, B, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *surface consists exactly of vector-space source exposure through V; topological-vector-space source exposure through T; finite seminorm-family evidence through S; convex balanced neighbourhood evidence through B; optional norm comparison through M; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.LocallyConvexSpaceUp.LocallyConvexSpaceCarrier namecert obligations`

Proof. Project the accepted carrier in Definition 3386.1. The rows $V, T, S, B, M, H, C, P, N$ are exactly the vector, topology, seminorm, neighbourhood, norm-comparison, transport, replay, provenance, and naming coordinates.

The vector and topological-vector rows are consumed first. They supply the additive carrier, scalar action, topology, and continuity obligations before any seminorm or neighbourhood evidence can be read.

The local-convexity evidence is then split across S and B. The seminorm ledger records finite probe families, while the neighbourhood ledger records the convex balanced neighbourhood rows generated from those probes.

Finally M, H, C, P, N compare, transport, replay, source, and name the same finite packet. Since no dual selector, separation theorem, weak-star compactness row, or host topology predicate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2433.3 (Locally convex space seminorm-filter stability)**.** *Let $L = (V, T, S, B, M, H, C, P, N)$ be an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier. Any consumer that reads a finite seminorm probe from S and then passes to the neighbourhood filter generated by B must still factor through the same rows $V, T, S, B, M, H, C, P, N$. The handoff exports no candidate-free normed-space structure, no selected continuous dual, and no ambient weak topology beyond the displayed finite filter rows.*

Proof. Begin with a seminorm probe in S. Its vector-space source is the same V read by the carrier, and its topological compatibility is checked against the same T used to admit the packet.

Pass from the probe to the neighbourhood side through B. The rows in B record convexity, balancedness, and filter membership as displayed finite evidence, so the handoff remains inside the packet rather than introducing a host filter object.

The comparison row M may relate the probe to a norm row, but it is only a comparison coordinate. Transport and replay through H and C, with provenance and naming through P

and N, preserve the same route. Thus any accepted seminorm-filter read remains mediated by $V, T, S, B, M, H, C, P, N$ and cannot export a selected dual or ambient weak topology. □

Theorem 2433.4 (Locally convex space seminorm-family obligation)**.** *Let $L = (V, T, S, B, M, H, C, P, N)$ be an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier. Any accepted consumer that reads the finite seminorm family S must expose the seminormed-space comparison row of chapter 5868 before using those probes. Any weak-star-facing consumer may pass through chapter 5192 only after the displayed seminorm, convex-balanced neighbourhood, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows S, B, H, C, P, N have been read together with the source rows V, T.*

Proof. Project the accepted carrier $L = (V, T, S, B, M, H, C, P, N)$. The row S is the finite seminorm-family ledger, while V and T are the vector-space and topological-vector-space sources that make each seminorm probe admissible.

Route S through the seminormed-space sibling named in chapter 5868. That sibling supplies the comparison surface for finite family indices, zero, triangle, scalar, and induced-neighbourhood reads before a local-convexity consumer may use the probes as neighbourhood data.

For weak-star-facing consumers, keep the route downstream of the local convexity packet. The weak-star topology chapter can be cited only after S, B, H, C, P, N and the sources V, T are displayed, so the weak-star use is a finite observation route over the local-convexity packet rather than a selected-dual principle. □

Theorem 2433.5 (Locally convex space topological-vector seminorm route)**.** *Every local-convex consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier factors first through the topological-vector-space source of chapter 205, the vector-space source of chapter 47, and the norm-comparison surface of chapter 89. Only after those sibling rows have been displayed may a functional-analysis consumer read the finite seminorm ledger S and the convex balanced neighbourhood ledger B as local convexity evidence.*

Proof. Project the accepted carrier $L = (V, T, S, B, M, H, C, P, N)$. The rows T, V, and M are exactly the displayed topological-vector, vector, and norm-comparison coordinates named by the sibling chapters.

The seminorm and neighbourhood rows do not replace those siblings. They are read after V and T fix the algebraic and topological source, and after M records whether a norm comparison is present.

Thus a functional-analysis consumer reaches local convexity only through the ordered route $T, V, M, S, B, H, C, P, N$. No Hahn–Banach principle, selected dual, ambient weak topology, or host locally convex predicate is exported by this sibling handoff. □

Theorem 2433.6 (Locally convex space forward citation)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that cites an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet for seminorm or convex-balanced-neighbourhood data must first read the sibling rows named in the carrier route:* $\mathsf{VecSpace}^{\uparrow}$ *for algebraic source,* $\mathsf{TopVecSpace}^{\uparrow}$ *for topological-vector continuity,* $\mathsf{Norm}^{\uparrow}$ *for optional comparison, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *for the downstream consumer boundary. The local convexity citation is therefore a horizontal dependency through those displayed rows, not an isolated analytic predicate.*

Proof. Read the carrier of Definition 3386.1. Its rows V, T, M are the vector-space, topological-vector-space, and norm-comparison coordinates, while S and B are the seminorm and convex-balanced-neighbourhood ledgers consumed downstream.

The functional-analysis endpoint is named by the provenance and consumer boundary already recorded in the chapter opening. A consumer cannot detach S or B from the source rows that make them meaningful.

Thus every forward citation travels through $V, T, M, S, B, H, C, P, N$. The route records a real sibling dependency and exports no stand-alone local-convexity predicate outside the finite NameCert packet. □

Theorem 2433.7 (Locally convex space functional-analysis anchor)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must read the same seminorm family S, topological-vector route T, convex-balanced filter-neighbourhood rows B, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row N before any functional-analytic closure is exposed. The consumer boundary of chapter 94 may cite local convexity only through this finite route, not through a detached locally convex topology predicate, selected dual, Hahn–Banach principle, or weak-star compactness theorem.*

Proof. Project the carrier in Definition 3386.1. The rows S, T, B, H, C, P, N are explicit coordinates of the accepted packet, with S naming the seminorm family, T naming the topological-vector source, and B naming the finite filter-neighbourhood evidence.

Apply the forward-citation theorem Definition 3386.6. It already forces functional-analysis consumers to pass through the vector, topological-vector, norm-comparison, seminorm, neighbourhood, transport, replay, provenance, and local naming rows.

The present theorem records the same route at the functional-analysis boundary. Since the carrier has no coordinate for a detached topology predicate, selected dual, separation principle, or weak-star compactness theorem, no functional-analytic closure can be exposed before those displayed rows are read. □

Falsifiable prediction: No accepted $\mathsf{LocallyConvexSpace}^{\uparrow}$ consumer can cite local convexity unless it first exposes the vector-space source, topological-vector-space source, finite seminorm-family ledger, convex balanced neighbourhood ledger, transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: topvecspace, vecspace, norm, functionalanalysis: none is bijective with $\mathsf{LocallyConvexSpace}^{\uparrow}$ because this carrier alone binds finite seminorm probes to convex balanced neighbourhood rows while retaining the vector, topology, comparison, transport, replay, provenance, and local NameCert coordinates.

The Banach–Alaoglu and weak-star compactness packets read $\mathsf{LocallyConvexSpace}^{\uparrow}$ through its displayed seminorm and topological-vector rows before any dual-ball compactness surface is exposed. In particular, chapter 3420 and chapter 6401 consume the locally convex source only after the seminorm-family route of chapter 5868 has made neighborhoods, bounded dual probes, and transport rows visible; neither compactness packet may treat $\mathsf{LocallyConvexSpace}^{\uparrow}$ as a hidden kernel premise detached from those finite rows.

Theorem 2433.8 (Locally convex Banach-seminorm route)**.** *Every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet that is consumed by a Banach-facing functional-analysis row must first expose the* $\mathsf{SeminormedSpace}^{\uparrow}$ *family route of chapter 5868 and the* $\mathsf{TopVecSpace}^{\uparrow}$ *linear-topology route of chapter 205. A* $\mathsf{Banach}^{\uparrow}$ *read from chapter 90 is then available only through displayed bounded completion, complete-metric limit, or bounded-linear consumer rows already carried by the downstream packet. It does not replace the locally convex carrier by a host Banach space, selected norm, selected dual, or ambient completeness principle.*

Proof. Project the accepted locally convex carrier through Definition 3386.1. The rows S and B provide the finite seminorm family and convex balanced neighbourhood ledger, while T provides the topological-vector operation-continuity surface.

Read the sibling sources next. The seminorm row is anchored by $\mathsf{SeminormedSpace}^{\uparrow}$, and the linear-topology row is anchored by $\mathsf{TopVecSpace}^{\uparrow}$; these rows are visible before any functional-analysis consumer may cite local convexity.

When a Banach-facing consumer is also present, the $\mathsf{Banach}^{\uparrow}$ row is downstream of those displayed routes. It contributes only the bounded completion or limit rows named by the consumer packet, so the locally convex read remains mediated by $V, T, S, B, M, H, C, P, N$ and cannot export a detached Banach carrier or selected norm. □

Theorem 2433.9 (Locally convex topological-vector dependency anchor)**.** *The* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *dependency surface ties every seminorm-family read to the* $\mathsf{TopVecSpace}^{\uparrow}$ *source before Banach completeness is consumed. Thus a downstream* $\mathsf{Banach}^{\uparrow}$ *or bounded-linear row may use locally convex data only after the topological-vector operations, seminorm family, convex balanced neighbourhood ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows have all been read from the same finite carrier.*

Proof. Start with the topological-vector coordinate T. It supplies the vector addition and scalar multiplication continuity route through chapter 205, so every seminorm probe in S is interpreted against a displayed linear topology.

The neighbourhood ledger B is then read from the same carrier. Its convex balanced rows are generated from the visible seminorm and topological-vector data, rather than from an ambient locally convex host predicate.

Finally the structural rows H, C, P, N transport, replay, source, and name the route. Any Banach-facing consumer of chapter 90 receives only that already displayed route plus its own bounded completion or limit rows; the locally convex packet supplies no separate completeness, selected dual, or host equality coordinate. □

Theorem 2433.10 (Locally convex compact-regular-function dependency route)**.** *Every compact-regular real-function or functional-analysis consumer that cites the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the topological-vector-space source of chapter 205, the seminorm ledger, and the convex-balanced neighbourhood ledger from the source certificate. The route lets compact regular real-function packets use* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *as a source for functional-analysis neighbourhood control, while the exported rows remain* $V, T, S, B, M, H, C, P, N$ *and do not become a selected dual, an open-cover compactness proof, or an ambient host locally convex predicate.*

Proof. Project the source carrier of Definition 3386.1. The topological-vector-space row T is the displayed route through chapter 205, while S and B are the only seminorm and convex-balanced neighbourhood rows exposed to downstream consumers.

A compact-regular real-function packet may therefore cite the locally convex source only after those rows are visible. Functional-analysis consumers receive the same route through transport, replay, provenance, and local naming in H, C, P, N.

Since no row stores a selected dual, open-cover compactness proof, or host locally convex predicate, the dependency is a finite cross-reference route rather than an additional compactness or duality principle. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet supplies the finite seminorm and convex-neighbourhood rows consumed by the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495 before that carrier reaches its Baire-category handoff through chapter 3570 and chapter 3535.

Thus a $\mathsf{LocallyConvexSpace}^{\uparrow}$ read exports local convexity to the Frechet/Baire downstream lattice only through displayed seminorm, neighbourhood, transport, replay, provenance, and local naming rows; it exports no Hahn–Banach separation, infinite seminorm family, selected metric, or ambient completion.

Frechet dependency route. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet supplies the finite seminorm-family rows and convex balanced neighbourhood-filter rows consumed by the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495. The same read also consumes the $\mathsf{TopVecSpace}^{\uparrow}$ operation-continuity surface of chapter 205 and the $\mathsf{MetricCompletion}^{\uparrow}$ handoff of chapter 1506 before complete locally convex replay is available. Thus the dependency is horizontal: $\mathsf{LocallyConvexSpace}^{\uparrow}$ provides the local convexity evidence, while $\mathsf{FrechetSpace}^{\uparrow}$ repacks that evidence with topological-vector operations, metrizability, and completion-facing data without importing an ambient complete locally convex host, a selected metric, quotient completion equality, countable choice, or an infinite seminorm enumeration.

Frechet and topological-vector sibling lattice. The sibling lattice around $\mathsf{LocallyConvexSpace}^{\uparrow}$ has two mandatory dependency edges. A $\mathsf{FrechetSpace}^{\uparrow}$ read cites chapter 3495 for the metric-completion and metrizability rows, while a $\mathsf{TopVecSpace}^{\uparrow}$ read cites chapter 205 for the vector-space topology, addition-continuity, and scalar-continuity rows. A locally convex consumer may therefore export seminorm-family evidence only after those sibling rows have displayed the finite seminorm ledger, metric completion handoff, and vector-space topology surface; it does not bypass either sibling into an ambient locally convex host.

Frechet and topological-vector crosslink. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier depends on the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205 for the compatibility between vector addition, scalar multiplication, and the displayed topology before its seminorm and convex-balanced neighbourhood rows can be read as local convexity. The same packet feeds the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495 by adding countable seminorm-family, metrizable replay, and completion-facing constraints over that topological-vector surface. Thus the route from $\mathsf{TopVecSpace}^{\uparrow}$ through $\mathsf{LocallyConvexSpace}^{\uparrow}$ to $\mathsf{FrechetSpace}^{\uparrow}$ is a finite NameCert dependency chain, not a host-level assertion of an ambient locally convex complete vector space.

Theorem 2433.11 (Locally convex space forward lattice anchor)**.** *Every forward dependency read of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet factors through the displayed lattice*

$$\mathsf{VecSpace}^{\uparrow},\ \mathsf{TopVecSpace}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ S,\ B,\ H,\ C,\ P,\ N.$$

Here S is the finite seminorm ledger and B is the convex-balanced neighbourhood ledger. The sibling consumer of chapter 94 may cite local convexity only after these rows have been exposed, so the forward lattice records a finite NameCert dependency rather than an isolated analytic predicate.

Proof. Project the carrier of Definition 3386.1. Its source rows name $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{TopVecSpace}^{\uparrow}$, while the optional comparison row names $\mathsf{Norm}^{\uparrow}$ and the downstream boundary names $\mathsf{FunctionalAnalysis}^{\uparrow}$.

The local-convexity evidence is then read from S and B. The seminorm ledger supplies the finite probe family, and the neighbourhood ledger supplies the convex-balanced rows attached to the same topological-vector source.

Finally transport, replay, provenance, and naming are supplied by H, C, P, N. Repacking these rows gives the forward lattice above and leaves no coordinate for a detached locally convex predicate, selected dual, Hahn–Banach row, or weak-star compactness theorem. □

Every $\mathsf{LocallyConvexSpace}^{\uparrow}$ read that is consumed by $\mathsf{FunctionalAnalysis}^{\uparrow}$ or $\mathsf{FrechetSpace}^{\uparrow}$ must expose the convex-neighbourhood and seminorm-window surface before any downstream linear-topology or complete-metrizable row is read. The route passes from the local carrier of chapter 3386 through the $\mathsf{FrechetSpace}^{\uparrow}$ dependency of chapter 3495 and the functional-analysis consumers cited by chapter 94; it exports no host locally convex space, selected infinite seminorm family, quotient completion, or ambient completeness coordinate.

Theorem 2433.12 (Locally convex functional-analysis forward link)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that cites the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the vector-space source of chapter 47, the topological-vector-space source of chapter 205, the norm-comparison row of chapter 89, and the functional-analysis boundary of chapter 94 through the displayed seminorm ledger S and convex-balanced neighbourhood ledger B. The downstream Frechet route of chapter 3495 and weak-star route of chapter 5192 read* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *only through those same S and B rows, not through a selected dual or host locally convex predicate. The route exports finite local-convexity evidence only through $V, T, S, B, M, H, C, P, N$; it does not export a selected dual, a Hahn–Banach principle, weak-star compactness, or an ambient locally convex topology predicate.*

Proof. Project the carrier of Definition 3386.1. Its rows V, T, M are the displayed vector, topological-vector, and norm-comparison sources, while S and B are the only local-convexity ledgers.

The functional-analysis consumer can therefore reach local convexity only after those source rows have been read through the seminorm and convex-balanced neighbourhood data. The Frechet and weak-star consumers inherit the same restriction: they may inspect S and B, but they receive no selected dual, host locally convex predicate, or detached topology witness. Transport, replay, provenance, and naming through H, C, P, N preserve the same finite route, so no dual selector, Hahn–Banach row, weak-star compactness row, or host topology predicate is introduced. □

Theorem 2433.13 (Locally convex functional-analysis dependency route)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the seminorm family row S, the* $\mathsf{TopVecSpace}^{\uparrow}$ *transport row T, the finite convex-balanced neighbourhood ledger B, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row N. The consumer boundary of chapter 94 may then cite the packet only through those visible rows, never through a selected dual, a host locally convex topology predicate, a Hahn–Banach principle, quotient equality of functionals, or an ambient completeness theorem.*

Proof. Project the accepted carrier of Definition 3386.1. Its locally convex data are the seminorm row S and the convex-balanced neighbourhood ledger B, while the linear-topology compatibility is supplied by the $\mathsf{TopVecSpace}^{\uparrow}$ row T.

The downstream $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer can therefore replay local convexity only after S, T, B are visible, and only with the transport, replay, provenance, and naming rows H, C, P, N attached. The functional-analysis chapter of chapter 94 receives that finite route as a dependency surface, so no selected dual, host topology predicate, quotient equality, or completeness principle is introduced. □

The top-level $\mathsf{LocallyConvexSpace}^{\uparrow}$ hub exposes its sibling lattice through chapter 205, chapter 208, chapter 5868, and chapter 94. A locally convex read therefore enters the dossier through displayed topological-vector operations, convex-neighbourhood evidence, finite seminorm-family rows, and functional-analysis consumers before any downstream packet treats $V, T, S, B, M, H, C, P, N$ as a reusable source surface. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ dependency lattice is read through its displayed seminorm-family and topological-vector rows rather than through a detached host predicate.

A consumer reaches the packet by first reading $\mathsf{Seminorm}^{\uparrow}$ and $\mathsf{SeminormedSpace}^{\uparrow}$ of chapter 5868 for seminorm evidence, $\mathsf{TopVecSpace}^{\uparrow}$ of chapter 205 for addition and scalar-continuity transport, $\mathsf{ConvexSet}^{\uparrow}$ for convex balanced neighbourhood rows, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ for the downstream citation boundary. No route exports a selected dual, Hahn–Banach principle, weak-star compactness theorem, or ambient locally convex topology outside the finite NameCert packet.

Theorem 2433.14 (Locally convex neighbourhood sibling route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *neighbourhood read first consumes the* $\mathsf{TopVecSpace}^{\uparrow}$ *vector/topology rows of chapter 205 and the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite seminorm rows of chapter 5868. Only after those sibling rows are visible may the packet expose convex balanced neighbourhood data through the local rows* B, H, C, P, N*.*

Proof. Project the carrier of Definition 3386.1. The topological-vector row T records the linear topology and operation-continuity route, while the seminorm ledger S records the finite probes that generate the accepted neighbourhood reads.

Read those rows through their sibling chapters. The $\mathsf{TopVecSpace}^{\uparrow}$ route supplies the vector/topology source, and the $\mathsf{SeminormedSpace}^{\uparrow}$ route supplies the finite seminorm comparison surface before any convex balanced neighbourhood can be used.

The neighbourhood row B is therefore downstream of T and S. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route and do not introduce a detached locally convex host predicate. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ seminorm and convex-neighbourhood ledgers consume the sibling lattice directly: $\mathsf{VecSpace}^{\uparrow}$ supplies the linear operations through chapter 47, $\mathsf{TopVecSpace}^{\uparrow}$ supplies continuity of addition and scalar multiplication through chapter 205, $\mathsf{Norm}^{\uparrow}$ supplies scalar-bound readbacks through chapter 89, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ supplies downstream replay through chapter 94. The $\mathsf{NormedSpace}^{\uparrow}$ row of chapter 1950 is the single-norm specialization whose scalar-bound readbacks are consumed before the finite seminorm-family route is generalized to $\mathsf{LocallyConvexSpace}^{\uparrow}$. The seminormed-space row of chapter 5868 and the topological-vector-space row are the finite routes through which the carrier reads convex-neighbourhood and seminorm-control data. A local-convexity consumer therefore reads those sibling rows as part of the finite ledger route before citing the packet. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier consumes the finite seminorm-family ledger only after the topological-vector classifier surface of chapter 205 is visible. The same read is then available to the functional-analysis route of chapter 94: consumers inspect the S seminorm row and the B convex-balanced neighbourhood row as displayed NameCert data, not as a host locally convex predicate or an unbounded family of seminorms. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier reads its separating seminorm family through the finite family-seminorm packet of chapter 5868 and reads topology compatibility through the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205 together with the complete-metrizable consumer route of chapter 3495. The seminorm rows generate the displayed locally convex neighbourhood basis before the B convex-balanced neighbourhood ledger can be used, so a downstream consumer reaches local convexity through S and B as BEDC NameCert rows rather than through an ambient locally convex predicate. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ dependency lattice reads the $\mathsf{Seminorm}^{\uparrow}$ family through the finite family route of chapter 5868 and the $\mathsf{TopVecSpace}^{\uparrow}$ carrier of chapter 205 before any convex-neighbourhood route is available. Thus the local-convexity packet exposes seminorm-control rows and topological-vector continuity rows as finite sibling data, then passes them to S, B, H, C, P, N for transport, replay, provenance, and local naming rather than treating local convexity as a kernel-only predicate. Every $\mathsf{LocallyConvexSpace}^{\uparrow}$ consumer reads seminorm-family rows through the finite seminormed-space carrier of chapter 5868 and reads topological-vector operations through chapter 205 before any convex-neighbourhood or continuous-linear functional route is used. Thus

the local-convexity handoff remains the displayed route through $V, T, S, B, M, H, C, P, N$, not a host locally convex predicate detached from its seminorm and topological-vector-space sources. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet reads its neighbourhood basis only after the finite seminorm-family row and the $\mathsf{TopVecSpace}^{\uparrow}$ linear-topology row have been displayed together. The seminorm ledger is anchored by chapter 5868, while the ambient additive-topological continuity source is anchored by chapter 205; downstream functional-analysis consumers therefore use the locally convex surface through $V, T, S, B, M, H, C, P, N$, not through a detached host predicate. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier reads its separating seminorm family through the finite family ledger of chapter 5868 and its ambient additive-topological structure through the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205. A local-convexity consumer therefore reaches the S seminorm row only after the topological-vector operations and continuity rows have been displayed, and the resulting packet remains the finite route $V, T, S, B, M, H, C, P, N$ rather than a host locally convex predicate.

Theorem 2433.15 (Locally convex space sibling dependency route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer that exports a locally convex read must first expose the sibling route through the substantive dependency surfaces of chapter 205 for the* $\mathsf{TopVecSpace}^{\uparrow}$ *topology row, chapter 47 for the* $\mathsf{VecSpace}^{\uparrow}$ *or module algebra row, the finite local seminorm ledger carried by this packet, and chapter 89 for the optional* $\mathsf{Norm}^{\uparrow}$ *comparison row when present, together with the structural* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The exported read is therefore a horizontal dependency over the displayed vector, topology, seminorm, norm-comparison, transport, replay, provenance, and naming surfaces; it is not a standalone host predicate for local convexity.*

Proof. Project the accepted carrier of Definition 3386.1. Its rows $V, T, S, B, M, H, C, P, N$ already order the vector, topology, seminorm, neighbourhood, comparison, transport, replay, provenance, and naming data.

The sibling route reads T and V before any locally convex probe is exported. The seminorm ledger S supplies finite local convexity probes, and the comparison row M supplies a norm route only when such a row is displayed by the packet.

The structural rows H, C, P, N then transport, replay, source, and name the same finite route. Since no independent host predicate occurs in the carrier, every locally convex read remains tied to the sibling dependency surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet is generated from a vector-space source, a topological-vector-space source, a finite seminorm-family ledger, a convex balanced neighbourhood ledger, optional norm comparison rows, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3386.1, Definition 3386.2, Definition 3386.3, and Definition 3386.15 seed the finite locally convex carrier over $V, T, S, B, M, H, C, P, N$ and its sibling dependency route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice-selected duals, ambient weak-star compactness, spectral theory, unrestricted barrelledness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked vector, topological-vector, seminorm, convex balanced neighbourhood, norm comparison, transport, replay, provenance, sibling dependency, and local naming rows for BEDC.Derived.LocallyConvexSpaceUp.

Theorem 2433.16 (Locally convex space sibling route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer must read the* $\mathsf{Seminorm}^{\uparrow}$ *evidence carried by the finite seminorm-family route of chapter 5868, the* $\mathsf{TopVecSpace}^{\uparrow}$ *topology row of chapter 205, and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer boundary of chapter 94 before treating the locally convex carrier as available. The local carrier is therefore a horizontal dependency through displayed seminorm, topological-vector, functional-analysis, transport, replay, provenance, and naming rows, not a detached host predicate.*

Proof. Project the accepted carrier of Definition 3386.1. The seminorm ledger, topological-vector source, convex-neighbourhood rows, and structural rows are all visible coordinates of the same finite packet.

The seminorm evidence is read first through the finite family route cited by the seminormed-space chapter, and the topology row is read through $\mathsf{TopVecSpace}^{\uparrow}$. Only after those sibling rows are displayed may a $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer use the packet as a locally convex source.

The remaining rows transport, replay, source, and name the same dependency surface. Since no coordinate stores a selected dual, Hahn–Banach principle, weak-star compactness theorem, or ambient locally convex topology predicate, the consumer receives only the displayed sibling route. □

Theorem 2433.17 (Locally convex TopVecSpace anchor)**.** *Every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet consumes the* $\mathsf{TopVecSpace}^{\uparrow}$ *carrier of chapter 205 and the* $\mathsf{ConvexSet}^{\uparrow}$ *neighbourhood rows of chapter 208 before any seminorm ledger, local-base read, or functional-analysis handoff is available. The local convexity read is therefore a sibling route through displayed vector-topology and convex-neighbourhood data, not an isolated host predicate.*

Proof. Project the carrier of Definition 3386.1. The row T is the topological-vector-space source, while B is the convex balanced neighbourhood ledger.

The row T must be read first because it supplies the topology, addition-continuity ledger, and scalar-continuity ledger used by every neighbourhood request. The row B is then read through $\mathsf{ConvexSet}^{\uparrow}$ as the displayed convex-neighbourhood evidence.

Only after T and B are visible may the finite seminorm ledger S or a local-base consumer inspect the packet. Transport, replay, provenance, and local naming keep the same route, so the chapter records a horizontal dependency through $\mathsf{TopVecSpace}^{\uparrow}$ and $\mathsf{ConvexSet}^{\uparrow}$ rather than a kernel-only local-convexity predicate. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier consumes the $\mathsf{TopVecSpace}^{\uparrow}$ carrier and operation-continuity transport route of chapter 205 together with the $\mathsf{Seminorm}^{\uparrow}$ family rows read through the finite seminormed-space surface of chapter 5868. Banach, Frechet, and weak-star consumers may inspect locally convex data only after those topological-vector and seminorm ledgers have displayed S, B, H, C, P, N as NameCert rows, so the route exports no detached locally convex host predicate, selected dual, Hahn–Banach principle, or ambient completeness theorem.

Theorem 2433.18 (Locally convex topological-vector-space dependency route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer must read the* $\mathsf{TopVecSpace}^{\uparrow}$ *linear-topology row of chapter 205, the* $\mathsf{ConvexSet}^{\uparrow}$ *neighbourhood row of chapter 208, and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *continuity-facing row of chapter 94 before treating the packet as locally convex. The route binds local convexity to displayed linear topology, convex neighbourhood, finite seminorm, continuity, transport, replay, provenance, and local naming rows; it is not a standalone host predicate for locally convex spaces.*

Proof. Project the carrier of Definition 3386.1. Its rows T, S, B already separate the topological-vector-space source, finite seminorm ledger, and convex balanced neighbourhood ledger.

The linear-topology row is read first through $\mathsf{TopVecSpace}^{\uparrow}$. The convex-neighbourhood row is then read through $\mathsf{ConvexSet}^{\uparrow}$ as the displayed source for the neighbourhood evidence in B.

Only after those rows are visible may the $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer inspect continuity-facing data. Transport, replay, provenance, and local naming preserve the same finite route, so no independent locally convex host predicate, selected dual, Hahn–Banach row, or weak-star compactness row is exported. □

Weak-star topology and weak-star compactness consumers read the $\mathsf{LocallyConvexSpace}^{\uparrow}$ source row before finite bounded-linear-functional windows: chapter 5192 uses the displayed weak-observation probes, and chapter 6401 consumes the same locally convex surface before the finite dual-ball compactness handoff. The route keeps weak-star reads tied to vector, topology, seminorm, convex-neighbourhood, transport, replay, provenance, and local naming rows rather than to a host locally convex predicate. The $\mathsf{ForwardSocketLedger}^{\uparrow}$ hub reads its forward request only after the supply ledger of chapter 528 and the far-end boundary of chapter 1061.

Chapter 2434

A Concrete Naming Certificate for $\mathsf{ForwardSocketLedger}^{\uparrow}$

The $\mathsf{ForwardSocketLedger}^{\uparrow}$ packet realizes the forward-socket family boundary of chapter 9401 as finite NameCert rows. It sits beside the supply-socket accounting of chapter 528, the apophatic socket classifier of chapter 536, the far-end boundary packet of chapter 1061, and the local inscription carrier of chapter 610. Its role is to make the forward read auditable before any consumer cites the apophatic far-end boundary.

The dependency route consumed before the forward socket export passes through the supply ledger of chapter 528 and the far-end boundary of chapter 1061; the forward packet only exports after those ledger and far-end rows are visible.

Definition 2434.1 (Forward socket ledger carrier)**.** *A* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, R, A, G, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the socket-kind row, recording which forward socket is being read and how it is classified among the accepted socket forms.*
2. *R is the requested supply-shape row, recording the requested external shape without supplying it as a kernel object.*
3. *A is the non-internalization marker, recording that the requested far-end supply is a boundary and not a generated state, record, rule, or* BHist *constructor.*
4. *G is the* GAP *ledger row, binding the refusal of internal supply to the visible socket-use and supply-ledger routes.*
5. *T is the apophatic far-end boundary row, naming the far end only as the common boundary read by the forward socket family.*
6. *H records componentwise* hsame *transport for K, R, A, G, T and the local naming row.*
7. *C records displayed* Cont *replay through socket kind, requested supply shape, non-internalization,* GAP *ledger, far-end boundary, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ForwardSocketLedger}^{\uparrow}$*,* $\mathsf{SupplySocketLedger}^{\uparrow}$*,* $\mathsf{ApophaticSocket}^{\uparrow}$*,* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$*,* $\mathsf{InscriptionPoint}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *row.*
The carrier contains no coordinate for T *as a kernel object, no hidden requested-supply witness, no socket selector outside K, no object-level equality between the far end and a local state, and no host fixed-point principle.*

Theorem 2434.2 (Forward socket ledger NameCert obligations)**.** *For every accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier $F = (K, R, A, G, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *surface consists exactly of socket-kind exposure through K; requested supply-shape exposure through R; non-internalization evidence through A;* GAP *ledger exposure through G; apophatic far-end boundary exposure through T; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 3387.1. The tuple lists socket kind, requested supply shape, non-internalization, gap ledger, far-end boundary, transport, replay, provenance, and local naming as explicit rows.

The socket can be read only after K and R expose its kind and requested shape. The rows A and G then mark the request as boundary supply rather than as an internal object.

The far-end boundary row T is downstream of those rows. Transport, replay, provenance, and naming through H, C, P, N preserve the same finite route. Since no hidden selector, requested-supply witness, internal far-end object, or host fixed-point principle occurs in the carrier, these rows exhaust the local NameCert surface. □

Theorem 2434.3 (Forward socket ledger apophatic non-escape)**.** *Every accepted read of the apophatic far-end boundary through* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *must first expose $K, R, A, G, T, H, C, P, N$. It cannot cite the far end as an internal object, a generated state, a selected supply witness, a socket-family fixed point outside the ledger, or an equality between* T *and a local inscription point.*

Proof. Start with the rows K and R. They identify the forward socket and its requested supply shape before any far-end boundary is available.

The rows A and G are the non-internalization gate. They state that the requested supply remains a GAP-ledgered boundary rather than a kernel-side value.

Only then may T be read as the apophatic far-end boundary, and H, C, P, N only transport, replay, source, and name that same finite route. Thus any consumer that skips $K, R, A, G, T, H, C, P, N$ has no accepted route to the far end. □

Falsifiable prediction: Every accepted read through $\mathsf{ForwardSocketLedger}^{\uparrow}$ must expose socket kind, requested supply shape, non-internalization marker, GAP ledger, apophatic far-end boundary, transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before any consumer can cite the far-end boundary.

Independence witness: supplysocketledger, apophaticsocket, apophaticfarendsocket, inscriptionpoint: none is bijective with $\mathsf{ForwardSocketLedger}^{\uparrow}$ because this carrier alone binds socket kind, requested supply shape, non-internalization, GAP ledger, far-end boundary, transport, replay, provenance, and local NameCert rows before a far-end read is accepted.

Closure status for $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ForwardSocketLedger}^{\uparrow}$ packet is generated from socket-kind exposure, requested supply-shape exposure, non-internalization evidence, a GAP ledger row, an apophatic far-end boundary row, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3387.1, Definition 3387.2, and Definition 3387.3 seed the finite forward-socket ledger over $K, R, A, G, T, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not internalize T, select a hidden supply witness, close a consciousness theorem, provide a host fixed-point theorem, claim object-level equality with a local inscription point, give Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked socket-kind, requested-supply, non-internalization, GAP-ledger, far-end boundary, transport, replay, provenance, and local naming rows for BEDC.Derived.ForwardSocketLedgerUp.

Theorem 2434.4 (Forward socket ledger MetaCIC candidate-discharge route)**.** *Let $F = (K, R, A, G, T, H, C, P, N)$ be an accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier. Any forward socket export consumed by the MetaCIC candidate-discharge frontier of chapter 765 must expose the forward rows $K, R, A, G, T, H, C, P, N$ before the MetaCIC discharge socket D can be read. The forward rows are dependency data for the socket consumer; they are not a concealed subject-reduction discharge and do not fill D by themselves.*

Proof. Project the accepted carrier of Definition 3387.1. The rows $K, R, A, G, T, H, C, P, N$ expose the socket kind, requested shape, non-internalization marker, gap ledger, far-end boundary, transport, replay, provenance, and local naming before any export is available.

Now read the candidate-discharge frontier of chapter 765. Its packet keeps the discharge socket D visible while candidate-mediated strong-normalization, confluence-facing, bounded-conversion, and source-readiness rows are consumed.

A MetaCIC consumer that receives a forward socket export therefore receives dependency data already ordered by the forward carrier. Since the forward packet contains no subject-reduction certificate row and no coordinate that identifies the far-end boundary with a typing proof, the export can only feed the visible socket route.

Thus the MetaCIC discharge socket may read the forward rows as dependency surface, but D remains a separate socket row until an explicit subject-reduction certificate is displayed. □

Theorem 2434.5 (Forward socket ledger SocketUse dependency route)**.** *Let $F = (K, R, A, G, T, H, C, P, N)$ be an accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier. Any accepted forward socket export that consumes the far-end boundary row T factors through the visible socket-use ledger of chapter 1044 before the local* $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *row is read. In particular, the export route must first expose the supply ledger of chapter 528, the far-end boundary of chapter 1061, the socket-use event rows $A, S, Q, V, R, H, C, P, N$ of* $\mathsf{SocketUseLedger}^{\uparrow}$*, and only then the forward rows $K, R, A, G, T, H, C, P, N$.*

Proof. Project the accepted carrier of Definition 3387.1. The forward packet presents the requested supply shape, non-internalization marker, GAP ledger, and far-end boundary as visible rows rather than as an internal object.

Use the supply-socket and far-end sibling routes cited in the carrier provenance. The supply ledger accounts for the requested shape and refusal route, while the apophatic far-end socket keeps the boundary row non-internalized.

The visible consumption of that boundary is the role of $\mathsf{SocketUseLedger}^{\uparrow}$. Its event rows expose the marked use and refusal ledger before an audit consumer reaches a local socket-use NameCert row. Therefore a forward export that cites T must pass through that event surface before N may name the forward packet. □

Chapter 2435

A Concrete Naming Certificate for $\mathsf{SuccessiveApproximation}^{\uparrow}$

Picard–Banach successive approximation reads an iterated approximation schedule through finite budgets before any completed limit claim is exposed. The $\mathsf{SuccessiveApproximation}^{\uparrow}$ packet sits on the L10 route from the convergence-modulus schedule of chapter 1438, the displayed limit handoff of chapter 1413, the regular rational readback of chapter 301, the finite observation windows of chapter 126, and the dyadic tolerance ledger of chapter 291. It gives contraction, Picard, and fixed-point consumers a finite iteration-budget surface rather than a host limit, countable-choice schedule, quotient sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2435.1 (Successive approximation carrier)**.** *A* $\mathsf{SuccessiveApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, T, W, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite iteration row, listing the inspected Picard approximants and the displayed application of the step operator at each row;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-budget ledger attached to the inspected rows;*
3. *T is the finite tail schedule that records which iteration indices are compared at each displayed budget;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to read each displayed approximant and its successor;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and dyadic budgets;*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row, visible only after I, D, T, W, Q have been displayed;*
7. *H records componentwise* hsame *transport for iteration rows, budgets, tail schedules, windows, readbacks, and the limit handoff;*
8. *C records displayed* Cont *replay for consumers of the same finite schedule;*
9. *P is the* Pkg *provenance over* $\mathsf{SuccessiveApproximation}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$ *row.*

The packet contains no host sequence limit, no countable-choice selector for tails, no quotient stream equality, no trichotomy split, and no ambient completeness coordinate.

Theorem 2435.2 (Successive approximation NameCert obligations)**.** *For every accepted* $\mathsf{SuccessiveApproximation}^{\uparrow}$ *carrier* $K = (I, D, T, W, Q, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite iteration rows, budget-classifier stability through* D, T, W, Q*, convergence-modulus handoff through* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$*,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *nonescape through* L*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, local naming through* N*, and refusal of host limits and countable choice. Every fixed-point, Picard, or contraction consumer receives only this displayed schedule before reading a terminal real-facing row.*

Proof. Project the carrier of Definition 3388.1. The iteration row I, dyadic budget ledger D, tail schedule T, finite windows W, and regular rational readbacks Q are the only approximation rows available to the packet.

Use the sibling convergence route. The convergence-modulus schedule of chapter 1438 consumes the same dyadic budgets and tail rows, while chapter 1413 accepts the resulting limit handoff only after $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks have been displayed.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the same finite schedule, so the packet cannot export a host limit, countable-choice tail selector, quotient sequence equality, trichotomy decision, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.SuccessiveApproximationUp.SuccessiveApproximationCarrier namecert obligations`

Falsifiable prediction: Every fixed-point consumer of a SuccessiveApproximationUp packet reads a finite iteration budget before any RealSequenceLimitUp handoff is visible.

Independence witness: cauchy-convergence-modulus, real-sequence-limit, regseqrat, streamname, dyadicratcore: SuccessiveApproximationUp is not bijective to any listed sibling because it jointly carries iteration rows, dyadic error budgets, tail schedules, finite windows, readback, limit handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SuccessiveApproximation}^{\uparrow}$ packet is generated from finite Picard iteration rows, dyadic error budgets, tail schedules, StreamName windows, RegSeqRat readbacks, a RealSequenceLimit handoff, transport, replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 3388.1 and Definition 3388.2 declare the finite Picard-style iteration-budget surface for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host limits, countable choice, quotient sequence equality, fixed-point existence, contraction uniqueness, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, budget classifier stability, convergence-modulus handoff, RealSequenceLimit nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SuccessiveApproximationUp.

Chapter 2436

A Concrete Naming Certificate for $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is admitted into the $\mathsf{Real}^{\uparrow}$ seal. It sits after the regular-rational source of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the dense-embedding route of chapter 1453, and the terminal real-name seal of chapter 38. The packet exposes the source-to-seal admission route without importing quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2436.1 (RealCauchyDenseEmbedding carrier)**.** *A* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, J, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, exposing the regular Cauchy rational name and its displayed modulus discipline;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the finite source entries requested by a real-name consumer;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested precision, comparison budget, and slack rows used by the admission route;*
4. *J is the* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *route row that names the dense source-to-real embedding discipline;*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, J have been displayed;*
6. *H records componentwise* hsame *transport for the regular source, windows, dyadic ledger, dense-embedding route, Real seal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a Real-admission request through R, W, D, J, S, H;*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a selected representative of a real name, countable choice over approximation stages, trichotomy, a host completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyDenseEmbeddingUp.RealCauchyDenseEmbeddingTasteGate single carrier alignment`

Theorem 2436.2 (RealCauchyDenseEmbedding NameCert obligations)**.** *For every accepted* RealCauchyDenseEmb *carrier* $E = (R, W, D, J, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular-rational source* R*, finite window exposure through* W*, dyadic tolerance accounting through* D*, dense-embedding routing through* J*, Real-seal handoff through* S*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. A* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer must read* R, W, D, J *before the seal row* S *is available.*

Proof. Project the carrier of Definition 3389.1. The regular source, finite windows, dyadic tolerance ledger, dense-embedding route, Real seal, transport, replay, provenance, and local naming rows are exactly the displayed coordinates.

The source side is finite. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular Cauchy rational name, the $\mathsf{StreamName}^{\uparrow}$ row opens only the requested windows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the tolerance ledger used by those windows.

The dense-embedding row J then mediates admission to the Real seal. The row S is downstream of R, W, D, J, and the structural rows H, C, P, N preserve and name the same route, so the certificate surface cannot expose quotient real equality, selected representatives, trichotomy, countable choice, or ambient completeness. □

Theorem 2436.3 (RealCauchyDenseEmbedding Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyDenseEmbedding}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Nam}$$

The terminal $\mathsf{Real}^{\uparrow}$ *row is only a seal for the displayed regular Cauchy rational source and its finite tolerance windows. It is not a source of quotient real equality, selected representatives, countable choice, trichotomy, host completion, or ambient real completeness.*

Proof. Start with the obligation surface of Definition 3389.2. A consumer cannot read the Real seal until the regular source, finite windows, dyadic tolerance ledger, and dense-embedding route have already been displayed.

The sibling rows enforce the direction of the read. $\mathsf{RegSeqRat}^{\uparrow}$ supplies the source name, $\mathsf{StreamName}^{\uparrow}$ supplies only finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the precision ledger, and $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ supplies the source-to-seal route.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no refused host-side principle is a coordinate of E, the Real row remains a seal and not an escape principle. □

Falsifiable prediction: Every accepted $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ consumer reads one $\mathsf{RegSeqRat}^{\uparrow}$ source row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and one $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ route before any $\mathsf{Real}^{\uparrow}$ seal is exposed.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-dense-embedding, real: $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries

source regularity, finite windows, dyadic tolerance accounting, dense-embedding routing, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegularCauchyDenseEmbedding route, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3389.1, Definition 3389.2, and Definition 3389.3 seed the finite regular-Cauchy rational source-to-Real seal admission packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, trichotomy, host completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity, finite window exposure, dyadic tolerance accounting, dense-embedding routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyDenseEmbeddingUp.

$\mathsf{RegularCauchyUniqueness}^{\uparrow}$ reads uniqueness through chapter 301, chapter 126, chapter 291, chapter 1412, and chapter 38 before any completion consumer cites the seal.

Chapter 2437

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniqueness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ packet records the textbook uniqueness route for regular Cauchy names as finite BEDC data. It sits after the regular-rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the comparison discipline of chapter 1412, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Real-completion consumers such as chapter 3579 use $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ to make the $\mathsf{Real}^{\uparrow}$ seal independent of the chosen $\mathsf{RegSeqRat}^{\uparrow}$ representative. The chapter only seeds the finite route from equality data for regular Cauchy names to the Real seal boundary.

Definition 2437.1 (Regular Cauchy uniqueness carrier)**.** *A* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, S, T, D, E, R, H, C, P, N)$$

with the following visible rows:

1. *X and Y are paired* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy name rows;*
2. *S and T are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the two names are observed at the same requested tolerance;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the displayed radius, comparison budget, and rational-distance rows used by the uniqueness route;*
4. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row tying the paired windows to one finite equivalence decision;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the paired names, windows, dyadic tolerance ledger, and comparison row have been displayed;*
6. *H stores componentwise* hsame *transport for X, Y, S, T, D, E, R and the structural rows;*
7. *C stores displayed* Cont *replay of the route*

$$X, Y \longmapsto S, T, D \longmapsto E \longmapsto R;$$

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by the displayed rows $X, Y, S, T, D, E, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, ambient Real completeness, or an unrecorded equality proof outside the packet.

Lean4 ✓ `BEDC.Derived.RegularCauchyUniquenessUp.RegularCauchyUniquenessTasteGate single carrier alignment`

Theorem 2437.2 (Regular Cauchy uniqueness NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier*

$$U = (X, Y, S, T, D, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$ *surface consists exactly of carrier admission for the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *names, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under the same displayed rows; tolerance exactness through D; transport invariance through H; ledger visibility through C and P; and Real-seal nonescape through R.*

Proof. Carrier admission is projection from Definition 3390.1. The projection exposes exactly the two regular Cauchy names, their finite windows, the dyadic tolerance ledger, the comparison row, the seal row, and the structural rows.

Classifier stability follows because the classifier is componentwise on the same displayed rows. Tolerance exactness is the projection of D, and ledger visibility is the fact that every consumer route in C reads X, Y, S, T, D, E, R before the provenance and naming rows.

Transport invariance is supplied by H: each transported accepted carrier keeps the same row roles. The Real-seal boundary is downstream from E, so no consumer can read R without first passing through the finite comparison packet. □

Theorem 2437.3 (Regular Cauchy uniqueness Real-seal obligations)**.** *Every accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet that exposes a* $\mathsf{Real}^{\uparrow}$ *seal must display the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and one* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row E before the seal row R is available. The seal obligation is therefore the ordered finite route*

$$X, Y \longmapsto S, T, D \longmapsto E \longmapsto R,$$

preserved by H, C, P, N and not by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3390.1. The projection exposes exactly the paired regular Cauchy rows, paired stream windows, dyadic tolerance ledger, comparison row, and terminal seal row.

Use Definition 3390.4 to bind the comparison to the single row D. The windows S, T and names X, Y are opened at that same tolerance source before E is read.

Now use Definition 3390.7. It places every $\mathsf{Real}^{\uparrow}$-facing read downstream from E and excludes quotient stream equality, selected limits, host real equality, countable choice, and ambient completeness from the seal route. The structural rows H, C, P, N transport, replay, source, and name that same finite obligation surface. □

Theorem 2437.4 (Regular Cauchy uniqueness tolerance exactness)**.** *Every accepted comparison in a* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet reads its tolerance data through the single displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D. The paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T and paired* $\mathsf{RegSeqRat}^{\uparrow}$ *names*

X, Y are opened at that row's radius and comparison budget, and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row E consumes no tolerance source outside D.*

Consequently the uniqueness route exports no hidden metric estimate, quotient-stream equality, selected limit, or ambient Real-completeness argument.

Proof. Project the carrier and inspect the tolerance coordinate. The only dyadic radius and rational-distance data are stored in D.

The windows S, T are tied to that coordinate by the continuation route C. Thus the two regular Cauchy names are observed through the same finite tolerance request before the comparison row is read.

The comparison row E is downstream from S, T, D. Since no other tolerance coordinate is present in the carrier, an accepted comparison cannot be justified by an external estimate or by quotient equality of streams. □

Theorem 2437.5 (Regular Cauchy uniqueness transport invariance)**.** *Let $U = (X, Y, S, T, D, E, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier. Componentwise* hsame *transport of the paired names, windows, tolerance ledger, comparison row, seal row, continuation route, provenance row, and local naming row produces another accepted carrier with the same uniqueness route. The transported classifier still reads only $X, Y, S, T, D, E, R, H, C, P, N$.*

Proof. Read the transport row H. Its components move the two $\mathsf{RegSeqRat}^{\uparrow}$ names, the two $\mathsf{StreamName}^{\uparrow}$ windows, the dyadic tolerance row, the comparison row, and the seal row together.

Replay the route through C after transport. The same ordered route from the paired names to the windows, tolerance, comparison row, and seal row remains visible.

Repack the transported rows with the same provenance and local naming data. Since the classifier has no coordinate outside the displayed rows, the transported packet is accepted by the same finite rule. □

Theorem 2437.6 (Regular Cauchy uniqueness tail-shift stability)**.** *Let $U = (X, Y, S, T, D, E, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier whose paired* $\mathsf{RegSeqRat}^{\uparrow}$ *windows have the same* $\mathsf{Real}^{\uparrow}$ *seal row R and the same uniqueness tolerance ledger D. For any finite* $\mathsf{StreamName}^{\uparrow}$ *tail shift of the displayed windows S, T, the shifted windows preserve the uniqueness handoff through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row, and* $\mathsf{Real}^{\uparrow}$ *seal boundary, with H, C, P, N carrying the transported route.*

Proof. Project the carrier rows. The paired names X, Y, windows S, T, dyadic tolerance ledger D, comparison row E, and seal row R are all displayed before any Real-facing read is available.

Apply Definition 3390.4. The tail-shifted windows are still read at the same displayed tolerance source D, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback and comparison row cannot import an external estimate.

Use Definition 3390.5 for the shifted windows and their transported structural rows. The same route through X, Y, S, T, D, E, R remains accepted after the finite reindexing.

Finally apply Definition 3390.7. Since the seal is downstream from the same comparison row and the carrier has no quotient stream equality, selected limit, countable-choice, ambient-completeness, or host-equality coordinate, the finite tail shift preserves the uniqueness handoff. □

Theorem 2437.7 (Regular Cauchy uniqueness Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet factors through the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row E, and only then the seal row R, with H, C, P, N preserving the same route.*

Thus the Real seal cannot escape to quotient stream equality, a selected limit, countable choice, ambient Real completeness, or any unrecorded host equality proof.

Proof. Start from a Real-facing read and project the continuation route C. The route lists the paired names, paired windows, dyadic tolerance row, and comparison row before the seal row R.

Use Definition 3390.4 to pin the comparison to D. Then use Definition 3390.5 to see that transport and replay preserve the same ordered finite rows.

The provenance and naming rows P, N attach the route to the local packet. Because the carrier contains no quotient stream, selected limit, countable-choice, ambient-completeness, or host-equality coordinate, no Real-facing consumer can read such data from the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ packet is generated from paired $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ comparison row, a terminal $\mathsf{Real}^{\uparrow}$ seal, transport data, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3390.1, Definition 3390.2, Definition 3390.3, Definition 3390.4, Definition 3390.5, and Definition 3390.7 name the L10 regular-Cauchy uniqueness route from finite RegSeqRat equality data through StreamName and DyadicRatCore tolerances to the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier surface, classifier stability, Real-seal obligations, tolerance exactness, transport invariance, ledger visibility, and Real-seal nonescape rows for the displayed uniqueness packet.

The downstream $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer of chapter 3579 reads $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ through the carrier of Definition 3390.1, the tolerance exactness row of Definition 3390.4, and the Real seal nonescape row of Definition 3390.7. This route keeps the $\mathsf{Real}^{\uparrow}$ seal independent of the chosen $\mathsf{RegSeqRat}^{\uparrow}$ representative: the completion witness consumes the displayed uniqueness comparison before the seal row is available, and it receives no quotient stream, selected limit, countable choice, or ambient completeness coordinate from the uniqueness packet.

Theorem 2437.8 (Regular Cauchy uniqueness classifier lock)**.** *Let*

$$U = (X, Y, S, T, D, E, R, H, C, P, N)$$

and

$$U' = (X', Y', S', T', D', E', R', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carriers. If a consumer reads the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows R and R' as the same regular Cauchy limit, then that read must factor through the displayed Cauchy-equivalence classifier before either seal is consumed.*

Concretely, the read passes through the paired $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y and X', Y', the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, T and S', T', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers D, D', the comparison rows E, E', componentwise* hsame *transport H, H', displayed* Cont *replay C, C',* Pkg *provenance P, P', and the local* NameCert *rows N, N'. It receives no host quotient stream equality, no selected representatives, no countable choice, and no unrecorded equality proof outside the local packet rows.*

Proof. Project both accepted carriers through Definition 3390.1. The projection exposes the paired regular Cauchy names, finite stream windows, dyadic tolerance ledgers, comparison rows, terminal real seals, and structural rows on both sides.

Invoke the tolerance exactness row of Definition 3390.4. It binds each comparison read to the displayed dyadic ledger before the comparison row can be consumed, so the two seal reads cannot be compared without first exposing the same kind of finite tolerance source.

The sibling route then passes through chapter 1820 before chapter 38: the comparison rows E, E' are consumed as Cauchy-equivalence classifier data, and only after that classifier read are the terminal Real seal rows available.

Next use the completion-consumer route following the chapter closure block. That route says every $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer reads $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ through the carrier, tolerance exactness, and Real seal nonescape rows before any $\mathsf{Real}^{\uparrow}$ seal is available.

Finally apply Definition 3390.7. Since the carriers contain no quotient stream, selected representative, countable-choice, or host-equality coordinate, the only available equality route is the displayed classifier route through E, E' together with $H, H', C, C', P, P', N, N'$. The read is therefore locked to the local Cauchy-equivalence classifier before either terminal seal is consumed. □

$\mathsf{RegularCauchyUniqueness}^{\uparrow}$ reads the completion-uniqueness boundary of chapter 1782 after the $\mathsf{RegSeqRat}^{\uparrow}$ comparison row and before any $\mathsf{Real}^{\uparrow}$-facing terminal seal. This dependency keeps the uniqueness route on displayed finite regular-Cauchy completion rows, and it rules out quotient stream equality, selected representatives, ambient real equality, or any choice of a preferred limit witness. The $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ route belongs to the Real classifier lattice: before any terminal uniqueness consumer compares two real-name seals, it reads the source $\mathsf{RegSeqRat}^{\uparrow}$ rows of chapter 301, inspects equality through the $\mathsf{RealNameClassifier}^{\uparrow}$ boundary of chapter 1517, and then consumes the $\mathsf{Real}^{\uparrow}$ seal rows already displayed by the carrier. This order keeps the terminal comparison attached to regular-rational source data and to the classifier inspection route rather than to quotient stream equality, selected representatives, host real equality, or ambient completeness.

Proof. Project the carrier rows and read the comparison from left to right. The regular Cauchy source rows provide the two names, the classifier boundary supplies the finite equality inspection, and the Real seal rows are consumed only after those entries have been displayed, so no uniqueness consumer receives an unrecorded terminal comparison. □

Chapter 2438

A Concrete Naming Certificate for $\mathsf{DyadicMeshComposition}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicMeshComposition}^{\uparrow}$ packet records the finite common-refinement route used when two displayed dyadic meshes are composed before a Real-and-Completion consumer reads a closed interval net. It sits between the dyadic mesh surface of chapter 358, the interval-net route of chapter 1250, the dyadic rational arithmetic surface of chapter 291, and the stream-window readbacks of chapter 126 and chapter 301. The packet is a finite naming certificate for mesh composition; it is not an open-cover compactness theorem, quotient stream equality, selected limit, host real equality, or ambient completeness primitive.

Definition 2438.1 (DyadicMeshComposition carrier)**.** *A* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M_0, M_1, R, Q, H, C, P, N)$$

with the following displayed rows:

1. M_0 *is the source* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the first finite ordered dyadic mesh and its inspected cell ledger.*
2. M_1 *is the target* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the second finite ordered dyadic mesh and its inspected cell ledger.*
3. R *is the common-refinement relation row, listing for each displayed source cell the finite family of target cells that refines it and recording the reverse containment reads needed by downstream consumers.*
4. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger storing the endpoint comparisons, radius splits, and dyadic slack witnesses used by the refinement relation.*
5. H *records componentwise* hsame *transport for* M_0, M_1, R, Q, C, P, N.
6. C *records displayed* Cont *replay from a mesh-composition request through the two mesh rows, the common-refinement relation, and the dyadic bound ledger.*
7. P *is the* Pkg *provenance over* $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an infinite mesh, selected subcover, quotient equality of streams, host real equality, hidden point chooser, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2438.2 (DyadicMeshComposition NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier* $T = (M_0, M_1, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{DyadicMesh}^{\uparrow}$ *rows, common-refinement relation exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over the mesh, interval-net, stream, readback, history, transport, continuation, and naming rows, and local naming. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *consumer of the packet must read* M_0, M_1, R, Q *before it can use the structural rows* H, C, P, N*.*

Proof. Project the carrier of Definition 3391.1. The rows M_0 and M_1 expose the two finite dyadic mesh ledgers, while R records the displayed common-refinement relation between their cells.

Read the cell-bound ledger next. The row Q supplies the dyadic endpoint comparisons and radius splits that witness each refinement edge in R, so the relation is checked through $\mathsf{DyadicRatCore}^{\uparrow}$ data rather than through a host-level order on real intervals.

The remaining rows add only structure. The transport row H preserves the displayed coordinates, C replays the finite route, P records provenance for the named dependencies, and N names the local certificate. These projections exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.DyadicMeshCompositionUp.DyadicMeshCompositionNameCertObligations`

Theorem 2438.3 (DyadicMeshComposition refinement transitivity)**.** *Let*

$$T = (M_0, M_1, R, Q, H, C, P, N)$$

be an accepted $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier. If a displayed cell of* M_0 *refines through a finite* R*-row into displayed cells of* M_1*, and those target cells are later inspected by a second accepted dyadic mesh row, then the composite inspection is again represented by finite refinement rows over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger. The composite exports no infinite refinement tree, quotient cell equality, hidden selector, or ambient completeness coordinate.*

Proof. Induct over the finite BHist rows displaying the refinement relation R. The empty row has no cell to export, so the composite route is the empty displayed refinement.

For the step row, project the current source cell and the finite family of target cells named by R. The ledger Q supplies the endpoint comparisons and radius splits for that row, while the induction hypothesis composes the remaining displayed rows.

Transport and replay preserve the constructed route. Componentwise hsame in H keeps the source mesh, target mesh, refinement row, and dyadic ledger aligned, and the Cont row C replays the same finite sequence of inspections. Thus the composite remains a finite mesh-refinement packet with no extra analytic coordinate. □

Theorem 2438.4 (DyadicMeshComposition closed-interval-net handoff)**.** *Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *consumer that uses a* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *packet reads the interval net through the displayed mesh rows* M_0, M_1*, the common-refinement relation* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger* Q*, and the structural rows* H, C, P, N*. The handoff supplies the finite mesh-composition route needed by closed bounded interval nets without adding open-cover compactness, selected subcovers, quotient stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted closed-interval-net consumer. By chapter 1250, the consumer already reads interval coverage through finite dyadic mesh rows and their rational and dyadic cell ledgers.

Now insert the mesh-composition packet. The rows M_0, M_1, R, Q identify the finite common-refinement route, and Definition 3391.2 shows that this route is the complete local certificate surface before the structural rows are consumed.

Finally apply the refinement transitivity theorem. It keeps successive mesh inspections inside the displayed dyadic ledger and finite BHist induction. The closed-interval-net handoff is therefore exhausted by the stated rows and cannot expose any of the excluded compactness, quotient, selector, host-equality, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshComposition}^{\uparrow}$ packet is generated from two DyadicMesh rows, a finite common-refinement relation, a DyadicRatCore cell-bound ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3391.1, Definition 3391.2, Definition 3391.3, and Definition 3391.4 seed the finite common-refinement route consumed by closed bounded interval nets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient stream equality, infinite refinement trees, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement exactness, DyadicRatCore cell-bound accounting, transport stability, replay stability, closed-interval-net handoff, and provenance exhaustion for BEDC.Derived.DyadicMeshCompositionUp.

Chapter 2439

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ packet names the finite operator surface that turns a regular Cauchy family and its displayed modulus into a Real-facing limit seal. It sits after the tail-modulus handoff of chapter 3383 and beside the completion operator of chapter 3379; the originating proof route is the regular-Cauchy construction of section 9238.1. The packet gives Phase Real and Completion a single operator-level object over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not another handoff-only certificate.

Definition 2439.1 (RegularCauchyLimitOperator carrier)**.** *A* $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\mu, \tau, S, R, D, H_0, E, H, C, P, N)$$

with the following displayed rows:

1. μ *is the modulus-input row for the regular Cauchy family over* $\mathsf{RegSeqRat}^{\uparrow}$.
2. τ *is the finite tail selector row that turns a requested tolerance into the shared tail threshold used by the diagonal read.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected at the selected threshold and at the diagonal representative.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected family member, the diagonal representative, and the comparison witnesses read by the operator.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording endpoint tolerances, the middle modulus leg, slack accounting, and the final requested bound.*
6. H_0 *is the* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *route imported from chapter 3383.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* μ, τ, S, R, D, H_0 *have been displayed.*
8. H *records componentwise* hsame *transport for the modulus input, tail selector, windows, regular readbacks, dyadic ledger, handoff route, Real seal, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay from a limit-operator request through* $\mu, \tau, S, R, D, H_0, E, H, P, N$.

10. *P is the* Pkg *provenance over* RegularCauchyLimitOperator$^{\uparrow}$, TailModulusLimitHandoff$^{\uparrow}$, CauchyCompletionOperat StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyLimitOperator}^{\uparrow}}$ *row.*

The classifier accepts only packets whose operator read is exhausted by these rows. It has no coordinate for quotient streams, selected host limits, countable-choice schedules, trichotomy decisions, host completion operators, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 2439.2 (RegularCauchyLimitOperator NameCert obligations)**.** *For every accepted* RegularCauchyLimitO *carrier $L = (\mu, \tau, S, R, D, H_0, E, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyLimitOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the modulus input, finite tail selector,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* TailModulusLimitHandoff$^{\uparrow}$ *route, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion-facing consumer must read μ, τ, S, R, D, H_0 before the Real seal row E is available.*

Proof. Project the carrier of Definition 3392.1. The rows μ and τ expose the constructive modulus request and the selected finite tail threshold.

Read the finite analytic rows. The window row S, readback row R, and tolerance ledger D supply the inspected StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ data used by the diagonal estimate.

The handoff route H_0 imports the already displayed tail-modulus route of chapter 3383. The Real seal E is downstream of that route, and H, C, P, N only transport, replay, source, and name the same finite packet.

Since quotient streams, selected host limits, countable-choice schedules, trichotomy decisions, host completion operators, and ambient completeness are absent from the carrier inventory, they cannot become local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitOperatorUp.RegularCauchyLimitOperatorNameCertObligations`

Theorem 2439.3 (RegularCauchyLimitOperator diagonal Real seal)**.** *Every Real-completion consumer of an accepted* RegularCauchyLimitOperator$^{\uparrow}$ *packet receives the finite operator route*

$$\mu, \tau, S, R, D, H_0 \longmapsto E \longmapsto H, C, P, N.$$

The route factors the diagonal regularity construction of section 9238.1 through the tail-modulus handoff of chapter 3383 and the completion-facing operator discipline of chapter 3379. It exports a Real$^{\uparrow}$ *seal only as a displayed consequence of the modulus input, selected tail, finite windows, regular readback, dyadic tolerance ledger, and handoff route.*

Proof. Start with an accepted limit-operator packet. By Definition 3392.2, the consumer first reads the modulus input, the tail selector, the finite windows, the regular-rational readback, and the dyadic tolerance ledger.

The tail-modulus row H_0 then packages those reads in the handoff form already isolated by Definition 3383.3. This is the finite route consumed by the diagonal regularity argument before any terminal real name can be sealed.

The Real row E therefore has no independent host-limit coordinate. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming, so the exported diagonal seal is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ packet is generated from a regular Cauchy modulus input, finite tail selector, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, TailModulusLimitHandoff route, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3392.1, Definition 3392.2, and Definition 3392.3 seed the finite operator surface from regular Cauchy modulus data to a Real-facing diagonal seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-input exactness, finite tail selector coverage, StreamName window coverage, RegSeqRat readback, DyadicRatCore tolerance accounting, TailModulusLimitHandoff compatibility, Real seal nonescape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyLimitOperatorUp.

Chapter 2440

A Concrete Naming Certificate for DnaWindowSixBoundaryTile$^{\uparrow}$

The DnaWindowSixBoundaryTile$^{\uparrow}$ packet realizes the finite stop/Trp and Ile/Met $X_2 \sqcup \{11\}$ boundary-tile surface of Definition 9462.1154 and Definition 9462.1156 as BEDC NameCert data. It consumes the six-coordinate codon source window of chapter 2234, the standard-code transition rows of chapter 2245, the stop-suffix packet of chapter 1374, and the spectral stabilization route of chapter 2270. The packet records the two boundary tiles as finite classifier rows, not as a hidden biological table or an ambient stochastic model.

Definition 2440.1 (DnaWindowSixBoundaryTile carrier)**.** *A* DnaWindowSixBoundaryTile$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, M, B, I, S, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates of the relevant codon windows;*
2. *M is the* CodonTransitionMatrix$^{\uparrow}$ *row exposing the standard-code bridge rows used to read stop, tryptophan, isoleucine, methionine, and initiation contexts;*
3. *B is the finite boundary-tile classifier row, recording the stop/Trp tile* $\mathrm{U} + \mathrm{RR} + X_2/11$ *and the Ile/Met tile* $\mathrm{AU} + X_2/11$*;*
4. *I is the Ile/Met initiation handoff row, distinguishing the Met amino-acid row from the canonical start bridge row when initiation context is present;*
5. *S is the* StopCodonZeckendorfSuffix$^{\uparrow}$ *stop-suffix handoff row for the stop-side tile;*
6. *L is the* SpectralStabilization$^{\uparrow}$ *sibling route row used by finite consumers that compare the same stop-face boundary with spectral support data;*
7. *H records componentwise* hsame *transport for source windows, transition rows, boundary tiles, initiation handoff, stop suffix, spectral route, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from a boundary-tile request through W, M, B, I, S, L, H;*
9. *P is the* Pkg *provenance over* DnaWindowSixBoundaryTile$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, CodonTransitionMatrix$^{\uparrow}$, StopCodonZeckendorfSuffix$^{\uparrow}$, SpectralStabilization$^{\uparrow}$, *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}}$ *row.*
The classifier accepts only packets whose stop/Trp and Ile/Met boundary reads are exhausted by $W, M, B, I, S, L, H, C, P, N$. It has no coordinate for a hidden codon table, global biological semantics, probabilistic inference, quotient codon equality, or a selected stochastic process.

Lean4 (definition) `BEDC.Derived.DnaWindowSixBoundaryTileUp.DnaWindowSixBoundaryTileUp`

Theorem 2440.2 (DnaWindowSixBoundaryTile NameCert obligations)**.** *For every accepted* DnaWindowSixBound *carrier $T = (W, M, B, I, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}}$ *surface consists exactly of source-window admission through W, standard-code bridge-row admission through M, boundary-tile classifier exactness through B, initiation handoff through I, stop-suffix handoff through S, spectral-stabilization sibling routing through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3393.1. The source window, transition bridge, boundary classifier, initiation handoff, stop suffix, spectral sibling route, transport, replay, provenance, and local naming rows are the only displayed coordinates.

The stop/Trp tile and the Ile/Met tile are finite classifier reads. The row W supplies the six coordinate positions, M supplies the standard-code bridge rows, and B records the two $X_2 \sqcup \{11\}$ partitions.

The rows I, S, L keep the initiation, stop-suffix, and spectral consumers attached to the same boundary-tile surface. Finally H, C, P, N transport, replay, source, and name that surface, so no hidden codon table, global biological semantics, probabilistic inference, quotient codon equality, or selected stochastic process is exported. □

Theorem 2440.3 (DnaWindowSixBoundaryTile consumer nonescape)**.** *Every accepted consumer of a* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *packet reads the route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto B, I, S, L \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Thus the six-coordinate codon window and standard-code bridge rows are read before any stop/Trp, Ile/Met, initiation, stop-suffix, or spectral-stabilization consumer can use the boundary-tile classifier. The route exports no hidden biological table, global semantics, probabilistic inference, quotient codon equality, or selected stochastic model.

Proof. Use Definition 3393.2. The boundary classifier row B is downstream of the source-window row W and standard-code bridge row M.

The initiation, stop-suffix, and spectral rows are also downstream. They may consume the displayed boundary-tile classifier, but they do not replace the six-coordinate source window or the standard-code bridge rows.

The structural rows H, C, P, N preserve and name the same finite route. Since the carrier contains no row for the refused external principles, every accepted consumer remains inside the displayed packet. □

Falsifiable prediction: Any accepted $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ consumer omitting the six-coordinate $\mathsf{CodonSixBitWindow}^{\uparrow}$ source window or the standard-code $\mathsf{CodonTransitionMatrix}^{\uparrow}$ bridge rows refutes the packet.

Independence witness: codon-six-bit-window, codon-transition-matrix, stopcodonzeckendorf-suffix, spectral-stabilization: $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two boundary-tile classifier rows, initiation handoff, stop-suffix routing, spectral sibling routing, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.DnaWindowSixBoundaryTileUp` must provide explicit Nontrivial DnaWindowSixBoundaryTile$^{\uparrow}$ and FieldFaithful DnaWindowSixBoundaryTile$^{\uparrow}$ instances preserving $W, M, B, I, S, L, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$ **.**

Constructive story: A DnaWindowSixBoundaryTile$^{\uparrow}$ packet is generated from codon six-bit windows, standard-code bridge rows, finite stop/Trp and Ile/Met boundary-tile classifiers, initiation handoff, stop-suffix handoff, spectral sibling routing, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$, seed)

Scope closed: Definition 3393.1, Definition 3393.2, and Definition 3393.3 seed the finite boundary-tile packet over codon windows and standard-code bridge rows.

Formal verification: FormalStatus(NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological table identity, global biological semantics, probabilistic inference, quotient codon equality, selected stochastic modeling, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-tile classifier exactness, initiation handoff, stop-suffix handoff, spectral sibling routing, transport, replay, provenance, local naming, Nontrivial DnaWindowSixBoundaryTileUp, and FieldFaithful DnaWindowSixBoundaryTileUp for BEDC.Derived.DnaWindowSixBoundaryTileUp.

Chapter 2441

A Concrete Naming Certificate for $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ packet records the finite route from located endpoint signs through dyadic bisection windows to the $\mathsf{Real}^{\uparrow}$ seal. It sits next to the sign-bracket certificate of chapter 1402 and the dyadic branch certificate of chapter 2203, but it keeps the located real output as a sealed finite read rather than a selected root or a quotient real.

The packet is an L10 Real and Completion neighbour. Its source cells are located interval endpoints, its branch rows are dyadic, and its consumer boundary is the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route used by the regular-real seal.

Definition 2441.1 (Located real intermediate-value carrier)**.** *A* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, S, B, Q, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, including the lower endpoint cell, upper endpoint cell, located interval comparison data, and the dyadic endpoint refinements consumed by the branch route;*
2. *S is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign ledger, recording the accepted negative-side and positive-side endpoint reads and the displayed midpoint sign rows carried along the route of Definition 1402.11;*
3. *B is the* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *branch data, retaining the finite sequence of accepted dyadic child intervals and the sign boundary that travels with each branch row;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose radius and midpoint arithmetic determine the displayed stopping window;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the retained endpoints and midpoint observations;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows after Q fixes the finite tolerance request;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, S, B, Q, W, R have been displayed;*

8. *H records componentwise* hsame *transport for the located interval, sign ledger, dyadic branch data, tolerance row, window rows, readback row, seal row, replay row, provenance row, and local naming row;*
9. *C records displayed* Cont *replay of the route*

$$I, S \longmapsto B, Q \longmapsto W, R \longmapsto E;$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. It has no coordinate for a selected root, quotient real, real trichotomy decision, open-cover compactness argument, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LocatedRealIntermediateValueUp.LocatedRealIntermediateValueTasteGate`
`single carrier alignment`

Theorem 2441.2 (Located real intermediate-value NameCert obligations)**.** *For an accepted* LocatedRealIntermedia *carrier*

$$L = (I, S, B, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$ *surface consists of carrier admission for the displayed located interval, sign ledger, dyadic branch, tolerance, window, readback, seal, transport, replay, provenance, and naming rows; sign-ledger visibility through I and S; dyadic branch exactness through B and Q; finite-window readback through W and R; Real-seal exposure through E; and structural stability through H, C, P, N.*

Every obligation is internal to the displayed finite packet. In particular, the certificate cannot be discharged by selecting a root, passing to a quotient real, deciding trichotomy, invoking open-cover compactness, or using an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 3394.1. The projection exposes exactly the located interval, sign, branch, tolerance, window, readback, seal, transport, replay, provenance, and local naming rows.

Sign-ledger visibility follows by reading S only over the located endpoint row I. The dyadic branch stage then reads B together with the tolerance row Q, so each accepted child interval is tied to displayed midpoint arithmetic and a visible sign boundary.

The finite-window stage reads W and then R. The real-facing row E is downstream of those windows, while H, C, P, N transport, replay, source, and name the same route.

Since none of the carrier coordinates stores a root selector, quotient-real equality, trichotomy oracle, compactness datum, or Real-completeness principle, none of these objects can serve as a local obligation. □

Theorem 2441.3 (Located real sign-branch route)**.** *Every Real-facing read from an accepted* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ *packet factors through*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{IntermediateValue}^{\uparrow} \longmapsto \mathsf{DyadicIntermediateValue}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The route reads the located endpoint row I, the sign ledger S, the branch data B, the tolerance row Q, the windows W, the readback row R, and only then the seal row E, with H, C, P, N preserving the same finite packet.

Proof. Project the accepted carrier $L = (I, S, B, Q, W, R, E, H, C, P, N)$. The endpoint and sign stages are I followed by S, so the read begins with located signs rather than a host predicate over all real points.

Apply the dyadic bisection route of Definition 2203.3. The branch row B and tolerance row Q determine the finite child-interval route before any window data is consumed.

Next read the finite windows and regular-rational readback. The rows W and R are upstream of the seal row E, so a Real-facing consumer reaches the seal only through the displayed finite route.

The structural rows H, C, P, N transport, replay, source, and name that same route. Therefore no selected root, quotient real, real trichotomy decision, open-cover compactness argument, or ambient completeness principle can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint row, an $\mathsf{IntermediateValue}^{\uparrow}$ sign ledger, a $\mathsf{DyadicIntermediateValue}^{\uparrow}$ branch row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3394.1, Definition 3394.2, and Definition 3394.3 seed the finite located-sign route from dyadic branch data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected root, quotient real, real trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-ledger visibility, dyadic branch exactness, finite-window readback, Real-seal exposure, transport and replay stability, and provenance exhaustion for BEDC.Derived.LocatedRealIntermedia

Chapter 2442

A Concrete Naming Certificate for $\mathsf{LocatedRealLine}^{\uparrow}$

Following Bishop and Bridges, the located real line is presented as a finite interface over located real names, apartness evidence, and regular Cauchy readback, rather than as a host real-line object or quotient completion. The $\mathsf{LocatedRealLine}^{\uparrow}$ packet binds the located-name surface of chapter 1933, the positive-separation evidence of chapter 304, the terminal real seal of chapter 38, the regular rational source route of chapter 301, and the finite observation windows of chapter 126. It also reads the dyadic endpoint and tolerance ledgers of chapter 291 so L10 real-completion routes can cite one non-isolated line interface.

Definition 2442.1 (LocatedRealLine carrier)**.** *A* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Lambda = (L, A, Q, R, S, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *row presenting the point-facing located real-name surface.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *row presenting positive finite-window separation evidence available to line-facing order and metric consumers.*
3. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same regular rational source route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular Cauchy readback used by L and Q.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation schedule naming the windows inspected by the line interface.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and tolerance ledger used for located windows, apartness gaps, and regular Cauchy bounds.*
7. *H records componentwise* hsame *transport for L, A, Q, R, S, D and for the structural replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a line request through $L, A, Q, R, S, D, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealLine}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite located real-line surface. It contains no quotient real equality, trichotomy decision, host real-line primitive, ambient completeness theorem, selected limit, countable-choice schedule, or bridge credential.

Lean4 (definition) `BEDC.Derived.LocatedRealLineUp.LocatedRealLineUp`

Theorem 2442.2 (LocatedRealLine NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier* $\Lambda = (L, A, Q, R, S, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located-real row, apartness row, real seal, regular rational source, stream schedule, dyadic ledger, transport, replay, provenance, and local naming rows;*
2. *classifier stability, requiring sameness to be componentwise transport of* L, A, Q, R, S, D *through the displayed* hsame *and* Cont *routes;*
3. *located point exactness, requiring every point read to pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *row* L *and its scheduled regular Cauchy source rows* R, S, D*;*
4. *apartness visibility, requiring every positive gap exposed to an order or metric consumer to pass through the* $\mathsf{ApartnessReal}^{\uparrow}$ *row* A *and the same finite dyadic ledger* D*;*
5. *real-seal compatibility, requiring the* $\mathsf{Real}^{\uparrow}$ *row* Q *to retain the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *source surface seen by* L*;*
6. *non-escape, excluding quotient real equality, trichotomy, host real-line primitives, ambient completeness, selected limits, countable choice, Lean verification, and bridge checking as local data.*

No located real-line obligation is stored outside these rows.

Proof. Project the carrier of Definition 3395.1. The analytic rows are precisely the located-real row L, apartness row A, real seal Q, regular rational source R, stream schedule S, and dyadic ledger D.

Classifier stability is componentwise. The rows H and C transport and replay the six analytic rows, while P and N source and name the same displayed packet.

Located point exactness follows by reading L through chapter 1933. Apartness visibility follows by reading A through chapter 304. In both cases the only line-facing observations available are the scheduled regular Cauchy windows and dyadic ledgers already stored in R, S, D.

The seal row Q is terminal and shares the same R, S source route. Since no remaining coordinate contains quotient equality, trichotomy, a host real line, ambient completeness, selected limits, or countable choice, the listed rows exhaust the local certificate. □

Theorem 2442.3 (LocatedRealLine regular Cauchy real handoff)**.** *Every L10 real-completion consumer of an accepted* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier receives the finite route*

$$L, A, R, S, D \longmapsto Q \longmapsto H, C, P, N.$$

The handoff first exposes located real-name data, apartness-gap data, scheduled regular rational readback, stream windows, and dyadic tolerance accounting. Only after those rows are displayed does it expose the terminal $\mathsf{Real}^{\uparrow}$ *seal and the structural transport, replay, provenance, and local naming rows.*

Proof. Start with an accepted carrier $\Lambda = (L, A, Q, R, S, D, H, C, P, N)$. By Definition 3395.2, any consumer must read the point surface, positive-gap surface, regular Cauchy source, stream schedule, and dyadic ledger before the seal row is visible.

The located-real component L supplies the point-facing readback through the finite windows of chapter 126 and the regular source rows of chapter 301. The apartness component A supplies positive finite-window gap evidence through chapter 304.

The real seal Q is then the same terminal seal route cited by chapter 38. The dyadic ledger D remains the only endpoint, radius, and tolerance account available from chapter 291.

Repacking through H, C, P, N preserves exactly this displayed route. The carrier has no coordinate for quotient real equality, trichotomy, a host real-line primitive, ambient completeness, selected limits, countable choice, or bridge data, so the L10 handoff is exhausted by the finite packet above. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealLine}^{\uparrow}$ packet is generated from a LocatedReal row, an ApartnessReal row, a RealUp seal, RegSeqRat readback, StreamName windows, a DyadicRatCore ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3395.1, Definition 3395.2, and Definition 3395.3 seed the Bishop-style located real-line interface for L10 real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, host real-line primitives, ambient completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located point exactness, apartness visibility, regular Cauchy readback, dyadic ledger accounting, Real seal compatibility, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LocatedRealLineUp.

The $\mathsf{LocatedUpperBound}^{\uparrow}$ hub routes the reusable upper-bound witness between **??**, chapter 38, and the downstream chapter 1427 consumer.

Chapter 2443

A Concrete Naming Certificate for $\mathsf{LocatedUpperBound}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedUpperBound}^{\uparrow}$ packet isolates the finite upper-bound witness that a Real or Completion consumer may inspect before any least-upper-bound or completeness principle is available. It sits over the located-cut surface of chapter 404, the located-real surface of **??**, the comparison discipline of chapter 1381, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the downstream located-supremum consumer of chapter 1427. The chapter names upper-bound data as a reusable NameCert surface, rather than keeping it only as an internal row of the supremum packet.

Definition 2443.1 (LocatedUpperBound carrier)**.** *A* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the located family row, read through either a* $\mathsf{LocatedCut}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *surface.*
2. *U is the upper-bound ledger row. It records the displayed bound name, the comparison windows that certify the source family lies below that bound, and the dyadic tolerances attached to those comparisons.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule listing the finite windows inspected by the upper-bound ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the bound name and by every comparison window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located source, upper-bound ledger, finite windows, and regular readback have been displayed.*
6. *H records componentwise* hsame *transport for F, U, W, R, E and for the structural replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an upper-bound request through F, U, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite upper-bound surface. It contains no lower-approximation ledger, selected supremum, trichotomy row, quotient-cut carrier, host real equality test, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge credential.*

Definition 2443.2 (LocatedUpperBound classifier)**.** *The* $\mathsf{LocatedUpperBound}^{\uparrow}$ *classifier accepts exactly those finite packets* $B = (F, U, W, R, E, H, C, P, N)$ *whose upper-bound reads pass through the displayed source row* F*, comparison ledger* U*, finite request schedule* W*, regular rational readback* R*, and terminal real-name seal* E*. Classifier sameness is componentwise sameness of these rows through* H *and replay through* C*; no consumer-facing classifier may identify two packets by host real equality, quotient-cut equality, trichotomy, a selected supremum, countable choice, or an ambient completeness principle.*

Theorem 2443.3 (LocatedUpperBound NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier* $B = (F, U, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located source, upper-bound ledger, stream schedule, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *classifier stability, requiring upper-bound sameness to be componentwise transport of* F, U, W, R, E *through the displayed* hsame *and* Cont *routes;*
3. *source coverage, requiring every bounded source read to pass through the located-cut or located-real surface named by* F*;*
4. *comparison exactness, requiring every upper-bound assertion to pass through the* $\mathsf{BishopRealComparison}^{\uparrow}$ *discipline named by* U *at a finite window listed in* W*;*
5. *regular readback compatibility, requiring the bound name and each comparison window to use the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed;*
6. *non-escape, excluding lower-approximation cofinality, selected suprema, trichotomy, quotient cuts, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local data.*

No upper-bound obligation is stored outside these rows.

Proof. Project the carrier of Definition 3396.1. The analytic rows are precisely the source row F, upper-bound ledger U, finite window row W, regular readback row R, and terminal real seal E.

Classifier stability is componentwise. The rows H and C transport and replay the five analytic rows, while P and N source and name the same displayed packet.

Source coverage follows by reading F through the located-cut and located-real sibling surfaces. Comparison exactness follows by reading U through the Bishop-style comparison row at a window listed in W.

The readback row R is shared by the bound name and every comparison window before the seal E is exposed. Since no remaining coordinate contains lower-approximation cofinality, selected suprema, trichotomy, quotient cuts, host equality, countable choice, ambient completeness, Lean verification, or bridge data, the listed rows exhaust the local certificate. □

Theorem 2443.4 (LocatedUpperBound real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier receives the finite route*

$$F, U, W, R \longmapsto E \longmapsto H, C, P, N.$$

The handoff first exposes the located source, upper-bound comparison ledger, finite request schedule, and regular rational readback. Only after those rows are displayed does it expose the terminal $\mathsf{Real}^{\uparrow}$ *seal and the structural transport, replay, provenance, and local naming rows.*

Proof. Start with an accepted carrier $B = (F, U, W, R, E, H, C, P, N)$. By Definition 3396.3, any Real-facing consumer must read the source row, comparison ledger, finite window schedule, and regular readback before the real seal is available.

The source row F supplies the located-cut or located-real family interface. The ledger U supplies the upper-bound comparison rows through the Bishop-style comparison discipline, and W records the finite windows inspected for those comparisons.

The readback row R is the sole route into regular rational real-name data, and E is the only terminal real seal exposed by the carrier. Repacking through H, C, P, N preserves this order of disclosure.

Because the carrier has no lower-approximation coordinate, selected supremum, trichotomy row, quotient-cut carrier, host equality test, countable-choice schedule, ambient completeness theorem, or bridge data, the displayed route exhausts every $\mathsf{Real}^{\uparrow}$-facing handoff. □

Theorem 2443.5 (LocatedUpperBound supremum route)**.** *Every* $\mathsf{LocatedSupremum}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier may use the packet only as the upstream upper-bound face. The consumer receives* $F, U, W, R, E, H, C, P, N$ *as the finite upper-bound witness, while lower-approximation cofinality, selected supremum data, and completion-facing least-upper-bound closure remain obligations of the downstream* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet.*

Proof. Inspect the carrier $B = (F, U, W, R, E, H, C, P, N)$. Its only order-facing analytic row is the upper-bound ledger U, and that row is tied to the located source F, the finite windows W, the regular readback R, and the real seal E.

Apply Definition 3396.4. It shows that any real-facing read exported from the upper-bound packet reaches E only through the displayed upper-bound route.

The downstream supremum packet in chapter 1427 additionally requires lower-approximation and cofinality rows. Those rows are absent from Definition 3396.1, so a $\mathsf{LocatedSupremum}^{\uparrow}$ consumer can read this packet only as its reusable upstream upper-bound witness. □

Theorem 2443.6 (LocatedUpperBound real-certificate route)**.** *Every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet exposes its Real-facing certificate only along the finite route from the located source row, upper-bound comparison ledger,* $\mathsf{StreamName}^{\uparrow}$ *request schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. This route is stable under the displayed* hsame *transports and* Cont *replay rows, and no Real-facing consumer can read the certificate before those rows have been displayed.*

Proof. Project the carrier of Definition 3396.1. Its Real-facing analytic rows are exactly the located source F, upper-bound comparison ledger U, finite request schedule W, regular rational readback R, and terminal real seal E.

Apply Definition 3396.3. Carrier admission, source coverage, comparison exactness, and regular readback compatibility require every upper-bound certificate read to pass through those five analytic rows before the structural rows are consumed.

Now apply Definition 3396.4. It identifies the displayed route

$$F, U, W, R \longmapsto E \longmapsto H, C, P, N$$

as the only Real-facing handoff exported by the packet.

The transport row H and continuation row C are componentwise rows of the same carrier, so they preserve and replay the route rather than adding a second certificate source. Since the carrier has no selected supremum, lower-approximation ledger, quotient-cut equality, host real equality, countable-choice schedule, ambient completeness theorem, Lean verification row, or bridge credential, the stated route exhausts the reusable Real-facing certificate. □

Theorem 2443.7 (LocatedUpperBound comparison exactness)**.** *Every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet routes each upper-bound comparison through the located-cut, located-real, Bishop-real-comparison,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already displayed in the carrier before any supremum-facing consumer may read it. In particular, the upper-bound ledger U cannot be consumed as a detached order assertion: it must be read with the located source F, finite request schedule W, regular rational readback R, terminal seal E, and the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 3396.1. The rows F, U, W, R, E are the analytic comparison route, while H, C, P, N transport, replay, source, and name the same accepted packet.

Apply Definition 3396.4. It forces every Real-facing read to expose F, U, W, R before the terminal seal E is available.

Apply Definition 3396.5. A $\mathsf{LocatedSupremum}^{\uparrow}$ consumer receives this packet only as its upstream upper-bound witness, so it may not import lower-approximation cofinality or selected supremum data from the upper-bound carrier.

Apply Definition 3396.6. The route ties the upper-bound ledger to the finite request schedule and regular rational readback before any Real-facing certificate is consumed. Since the carrier has no selected supremum, trichotomy row, quotient-cut equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle, no supremum consumer can bypass the displayed comparison rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUpperBound}^{\uparrow}$ packet is generated from a located source row, an upper-bound comparison ledger, a StreamName finite-window schedule, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3396.1, Definition 3396.2, Definition 3396.3, Definition 3396.4, and Definition 3396.5 seed the reusable upper-bound witness for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected suprema, trichotomy, quotient cuts, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, source coverage, comparison exactness, regular readback compatibility, Real seal handoff, supremum routing, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LocatedUpperBoundUp.

Theorem 2443.8 (LocatedSupremum upstream upper-bound face)**.** *The downstream* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet of chapter 1427 consumes an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet only as its upstream upper-bound face. Concretely, the route supplied by Definition 3396.5 offers the finite rows $F, U, W, R, E, H, C, P, N$ as the displayed upper-bound witness; lower-approximation cofinality, least-upper-bound exactness, and supremum-facing closure remain rows of* $\mathsf{LocatedSupremum}^{\uparrow}$ *rather than rows of* $\mathsf{LocatedUpperBound}^{\uparrow}$.

Proof. Read Definition 3396.5. It identifies the accepted upper-bound carrier as a reusable witness for the source family, the upper-bound ledger, finite windows, regular readback, the Real seal, and the structural transport, replay, provenance, and local naming rows.

Now compare the downstream consumer in chapter 1427. Its supremum certificate requires lower-approximation and cofinality data in addition to an upstream upper-bound face. Those additional rows are not coordinates of Definition 3396.1.

Thus every handoff from $\mathsf{LocatedUpperBound}^{\uparrow}$ into $\mathsf{LocatedSupremum}^{\uparrow}$ is exactly the forward dependency through $F, U, W, R, E, H, C, P, N$. The supremum packet must supply the remaining supremum-specific rows itself. $\square$

Chapter 2444

A Concrete Naming Certificate for $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$

Following Bourbaki and Kelley on uniform embeddings and completion, the $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ packet records the finite BEDC route by which an embedding into a separated completion is consumed without quotient representatives or selected limits. It sits after the metric-completion carrier of chapter 1506, the separated comparison surface of chapter 1411, the uniform-completion route of chapter 1473, the Cauchy-name comparison of chapter 1412, and the terminal seal boundary of chapter 38. The packet names the embedding interface as displayed rows over completion input, separatedness, uniform continuity, Cauchy-equivalence stability, and Real-facing readback; it does not assert a quotient completion, choose representatives, or import ambient completeness.

Definition 2444.1 (UniformEmbeddingCompletion carrier)**.** *A* $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, U, Q, R, T, W, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row supplying the completion-facing metric request and its displayed finite tolerance route.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row requiring the target-side comparison to be separated before the embedding read is public.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row exposing the entourage-facing completion presentation consumed by the embedding.*
4. *Q is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row identifying when two source Cauchy names have the same displayed completion read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the completion, separatedness, uniform, and Cauchy-equivalence rows have been displayed.*
6. *T records the finite source-window schedule inspected by the embedding before a target read is accepted.*
7. *W records the finite target-window schedule produced by the embedding route.*
8. *A records the approximation ledger saying that matching source windows and tolerances transport to matching target windows.*

9. *H records componentwise* hsame *transport for M, S, U, Q, R, T, W, A and the structural rows.*
10. *C records displayed* Cont *replay of the route*

$$M, S, U, Q, T, A \longmapsto W \longmapsto R.$$

11. *P is the* Pkg *provenance over* UniformEmbeddingCompletion$^{\uparrow}$, MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CauchySequenceEquivalence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{UniformEmbeddingCompletion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose embedding-to-completion read is exhausted by $M, S, U, Q, R, T, W, A, H, C,$ It has no coordinate for quotient representatives, selected limits, host equality of completions, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformEmbeddingCompletionUp.UniformEmbeddingCompletionTasteGate single carrier alignment`

Theorem 2444.2 (UniformEmbeddingCompletion NameCert obligations)**.** *For every accepted* UniformEmbeddingCompletion$^{\uparrow}$ *carrier*

$$E = (M, S, U, Q, R, T, W, A, H, C, P, N),$$

the local NameCert$_{\text{UniformEmbeddingCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed twelve-row packet; metric-completion exposure through M; separated-target exposure through S; uniform-completion exposure through U; Cauchy-equivalence stability through Q; terminal Real sealing through R; finite source-window exposure through T; finite target-window exposure through W; approximation transport through A; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3397.1. The accepted packet exposes exactly the completion source, separated target, uniform completion, Cauchy-equivalence comparison, terminal seal, source and target windows, approximation ledger, and structural rows listed there.

The analytic obligations are read in the displayed order. The completion request passes through M and U, the separated target boundary is supplied by S, and the source comparison is stabilized by Q before the terminal Real-facing row R can be inspected.

The embedding content is the finite window handoff T, A, W. The source windows in T and their approximation ledger A determine the target windows in W; no step inserts a quotient representative or a selected completion point.

Finally H, C, P, N transport, replay, source, and name the same route. Since the excluded quotient, choice, host-equality, and ambient-completeness objects are absent from the carrier, they cannot appear as local NameCert obligations. □

Theorem 2444.3 (UniformEmbeddingCompletion universal route)**.** *Every completion-facing consumer of an accepted* UniformEmbeddingCompletion$^{\uparrow}$ *packet factors through the finite route*

$$M, S, U, Q, T, A \longmapsto W, R \longmapsto H, C, P, N.$$

Thus a uniformly embedded source can be read by completion consumers only through displayed metric-completion input, separatedness, uniform-completion structure, Cauchy-equivalence stability, finite source windows, approximation transport, finite target windows, the terminal Real seal, transport, replay, provenance, and local naming.

Proof. Start with an accepted carrier. The source completion row M and uniform-completion row U provide only completion-facing requests, while the separated row S restricts target comparison to the displayed separated boundary.

The Cauchy-equivalence row Q supplies the only equality-facing source comparison. It is read before the target windows are public, so a consumer cannot replace it by quotient-stream equality or host completion equality.

The universal route is the finite handoff T, A, W. The source-window schedule in T is transported by the approximation ledger A to the target-window schedule W, and the Real-facing read reaches R only after those rows have been displayed.

The structural rows H, C, P, N preserve, replay, source, and name exactly this packet. Hence every completion-facing consumer factors through the displayed route and cannot read a selected limit, quotient representative, countable-choice schedule, host completion equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ packet is generated from MetricCompletion, SeparatedMetric, UniformCompletion, CauchySequenceEquivalence, Real, finite source windows, finite target windows, approximation transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3397.1, Definition 3397.2, and Definition 3397.3 seed the finite nonquotient route for uniformly embedded completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformEmbeddingCompletionUp.UniformEmbeddingCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, host equality of completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-target exposure, uniform-completion exposure, Cauchy-equivalence stability, finite window transport, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformEmbeddingCompletionUp.

Chapter 2445

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and Bourbaki on reflective subobjects, the $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ packet records the RegSeqRat-specific completion reflector boundary as finite BEDC naming data. It sits after the finite observation rows of chapter 126, the regular rational source of chapter 301, the completed real-name surface of chapter 360, the monadic completion route of chapter 378, the idempotence boundary of chapter 2381, and the extension surface of chapter 397. The packet names only the finite reflector rows visible to RegSeqRat consumers; it does not identify quotient completions, select limits, assert host equality of completed objects, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2445.1 (RegularCauchyCompletionReflector carrier)**.** *A* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, E, U, I, P, H, C, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source regular Cauchy name and every completion-facing observation cut.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, exposing the regular rational readback and its displayed tolerance discipline before completion is requested.*
3. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *completed real-name row reached from S, Q through the displayed completion surface rather than through a host quotient carrier.*
4. *U is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit and bind route specialized to the same S, Q, E rows.*
5. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *boundary row recording that a second completion pass reuses the same completed surface.*
6. *P is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row for consumers whose maps are read through the displayed regular Cauchy source and the same completed real-name row.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, completed real-name rows, monad routes, idempotence boundaries, universal-property rows, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a RegSeqRat completion request through*

$$S, Q \longmapsto E \longmapsto U, I, P \longmapsto H, N.$$

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$ *row, with provenance read from the displayed* Pkg *rows over* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

The classifier compares accepted packets only through $S, Q, E, U, I, P, H, C, N$. It has no coordinate for quotient completion equality, selected limit representatives, host equality of completed objects, countable choice, detached $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionReflectorUp.RegularCauchyCompletionReflectorTasteGate`
`single carrier alignment`

Theorem 2445.2 (RegularCauchyCompletionReflector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier*

$$R = (S, Q, E, U, I, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for stream windows, RegSeqRat readbacks, completed real-name rows, monad routes, idempotence boundaries, universal-property extension rows, transport, replay, provenance, and local naming;*
2. *source regularity, requiring every completion-facing read to expose S and Q before E is available;*
3. *completion admission, requiring the completed row E to be read through the displayed* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *surface;*
4. *monad handoff, requiring unit and bind consumers to use U over the same S, Q, E rows;*
5. *idempotence stability, requiring repeated completion reads to reuse I and the same completed row E;*
6. *universal-property compatibility, requiring every extension consumer to read P over the same source, completion, monad, and idempotence rows;*
7. *nonescape, excluding quotient completions, selected limits, host equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. The stream window row, regular rational readback row, completed real-name row, monad route, idempotence boundary, universal-property row, transport, replay, and local naming row are exactly the displayed coordinates of R.

The source side is forced by S and Q. A completion-facing consumer has no completed row to inspect until the finite StreamName windows and the RegSeqRat readback have been displayed.

The completed row E is then read through the RealCauchyCompletion surface. The monad row U, idempotence row I, and universal-property row P all refer back to the same S, Q, E route, so none of them can introduce a separate completion object.

Finally H, C, N preserve and name the same finite packet. Since the carrier has no coordinate for quotient completions, selected limits, host equality, countable choice, detached Real completeness, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 2445.3 (RegularCauchyCompletionReflector unit handoff)**.** *Let $R = (S, Q, E, U, I, P, H, C, N)$ be an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier. Every unit-facing completion request factors through the finite route*

$$S \longmapsto Q \longmapsto E \longmapsto U \longmapsto H, C, N.$$

The handoff copies the displayed RegSeqRat source into the completed real-name surface by finite windows and regular readbacks. It does not supply quotient stream equality, a selected limit, host equality, countable choice, or ambient completeness.

Proof. Start with the source rows. The finite windows in S determine which parts of the regular rational source may be inspected, and Q supplies the matching RegSeqRat readback.

The completed row E is downstream of those source rows. A unit-facing request therefore reaches completion only after the finite source route has been displayed.

The monad row U records the unit handoff on that same route, and H, C, N transport, replay, and name it componentwise. No excluded host-side principle is available as a row of the packet. □

Theorem 2445.4 (RegularCauchyCompletionReflector idempotence boundary)**.** *For an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier $R = (S, Q, E, U, I, P, H, C, N)$, a repeated completion read is classified by the same completed real-name row E and the same idempotence boundary I. The second read may replay S, Q, U, P, H, C, N, but it has no independent completed carrier and no quotient equality coordinate.*

Proof. Read the carrier coordinates. The only completed real-name coordinate is E, and the only idempotence coordinate is I.

A repeated completion request can therefore replay the source windows, RegSeqRat readbacks, monad handoff, universal-property row, transport, continuation, and local naming. It cannot branch to a second completed object because no such coordinate appears in R.

Thus idempotence is a boundary on the displayed completion surface, not an assertion of host quotient equality or selected-limit equality. □

Theorem 2445.5 (RegularCauchyCompletionReflector ledger nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *packet reads the completion reflector through*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionMonad}^{\uparrow}, \mathsf{CompletionIdempotence}^{\uparrow}, \mathsf{Co}$$

The ledger exposes no quotient completion, selected limit representative, host equality of completed objects, countable-choice schedule, detached $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge-checking row.*

Proof. Apply Definition 3398.2. The public read begins with S, Q, then reaches the completed real-name row E, and only then consumes the monad, idempotence, and universal-property rows U, I, P.

The sibling chapters fix the same dependency route: chapter 301 supplies the regular rational source, chapter 360 supplies the completed real-name surface, and chapter 378 supplies the completion monad handoff.

The structural rows H, C, N preserve and name only that displayed route. Hence no refused host-side principle can escape through the local ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readbacks, a RealCauchyCompletion completed row, a CauchyCompletionMonad handoff, a CompletionIdempotence boundary, a CompletionUniversalProperty extension row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3398.1, Definition 3398.2, Definition 3398.3, Definition 3398.4, and Definition 3398.5 seed the RegSeqRat-specific completion reflector packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality of completed objects, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity, completion admission, unit handoff, idempotence boundary, universal-property compatibility, ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionReflectorUp.

Chapter 2446

A Concrete Naming Certificate for $\mathsf{SchwartzFunction}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzFunction}^{\uparrow}$ packet records the finite test-function carrier that exposes rapidly decreasing smooth windows to Fourier and tempered-distribution consumers. It sits over the weak-derivative route of chapter 2210, the finite Fourier observation surface of chapter 124, the tempered-distribution handoff of chapter 2310, the local support and gluing surface of chapter 3369, and the scalar readback of chapter 38. The chapter names only the finite BEDC carrier for test windows, derivative orders, polynomial weights, Fourier-readiness rows, and real seals; it does not import a completed Schwartz topology, host smooth functions, unrestricted seminorm families, Plancherel, inversion, or quotient test-function equality.

2446.1 Carrier definition

Definition 2446.1 (Schwartz function carrier)**.** *A* $\mathsf{SchwartzFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, G, F, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite derivative-observation row. It records the named weak-derivative requests, multi-order labels, sign conventions, and local derivative windows read through chapter 2210.*
2. *W is the finite window row. It records the compact or bounded windows on which pointwise reads, derivative reads, and support restrictions are visible.*
3. *G is the polynomial-weight ledger. It records the displayed weight exponents and rational bounds certifying rapid decay for the derivative orders requested in D.*
4. *F is the Fourier-readiness row. It records the finite transform and coefficient windows exported to chapter 124.*
5. *T is the tempered-distribution test handoff row. It exports the test-window inventory consumed by chapter 2310.*
6. *R is the scalar readback row through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, exposing only the finite rational and real endpoints needed by derivative, weight, and transform observations.*

7. *L is the local classifier row. It admits two packets together exactly when their derivative observations, windows, polynomial-weight ledgers, Fourier rows, tempered-distribution test rows, and scalar readbacks match componentwise.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from a Schwartz-test request through D, W, G, F, T, R, L, H in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$ *row.*

The classifier compares only these finite rows. It has no coordinate for a completed Schwartz space, an infinite seminorm topology, host smooth-function equality, arbitrary support selection, countable choice, Plancherel, inversion, or quotient test-function equality.

Lean4 ✓ `BEDC.Derived.SchwartzFunctionUp.SchwartzFunctionTasteGate single carrier alignment`

2446.2 The certificate

Theorem 2446.2 (Schwartz function NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *carrier*

$$S = (D, W, G, F, T, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for derivative observations, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming;*
2. *derivative-window exactness, requiring every derivative read to pass through D and W before any weighted or Fourier-facing read is exposed;*
3. *rapid-decay exactness, requiring every displayed polynomial-weight bound to be read from G for a derivative order already named in D;*
4. *Fourier-readiness exactness, requiring every finite transform or coefficient request to pass through F and then through the same scalar readback row R;*
5. *tempered-distribution test exactness, requiring every distribution-facing test row to pass through T after the derivative and weight ledgers have been opened;*
6. hsame, Cont, Pkg, *and local* NameCert *stability for the rows $D, W, G, F, T, R, L, H, C, P, N$.*

No obligation outside these rows is part of the certificate.

Proof. Carrier admission is projection from Definition 3399.1. The accepted tuple displays the derivative observations, local windows, polynomial-weight ledger, Fourier-readiness row, tempered-distribution test handoff, scalar readback, classifier, transport, replay, provenance, and local naming.

The derivative gate is D together with W. A consumer cannot ask for a weighted or transform observation until the finite derivative order and the window on which it is read have both been displayed.

The rapid-decay gate is G. Its bounds are attached to the displayed derivative orders, so the proof uses no infinite seminorm family and no host smooth-function equality.

The Fourier and distribution handoffs are F and T. These rows export only the finite transform windows and test rows already admitted by D, W, G, and the endpoints pass through R before any scalar seal is read. The remaining rows L, H, C, P, N compare, transport, replay, source, and name the same finite inventory. □

Theorem 2446.3 (Schwartz function Fourier stability)**.** *Every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *packet exported to a Fourier-facing consumer factors through*

$$D, W, G \longmapsto F \longmapsto R \longmapsto L, H, C, P, N.$$

Thus the consumer receives derivative windows, polynomial-weight ledgers, finite transform and coefficient windows, scalar readbacks, classifier, transport, replay, provenance, and local naming. It receives no completed Schwartz topology, Plancherel theorem, inversion theorem, quotient test-function equality, selected representative, countable choice, or ambient completeness principle.

Proof. Start with a Fourier-facing read. The derivative row D and window row W must be visible because each finite transform request is attached to a displayed test window and derivative order.

Read the polynomial-weight ledger G next. It is the only row that certifies the rapid-decay bound attached to the requested derivative order, so there is no appeal to an infinite seminorm topology.

The handoff to F is then finite. It records coefficient and transform windows consumable by chapter 124, and its scalar endpoints pass through R before the classifier, transport, replay, provenance, and naming rows are read. □

Theorem 2446.4 (Schwartz function tempered-distribution handoff)**.** *Every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *packet exported to a tempered-distribution consumer factors through the displayed derivative row, local window row, polynomial-weight ledger, test handoff row, scalar readback row, classifier, transport, replay, provenance, and local naming:*

$$D, W, G \longmapsto T \longmapsto R \longmapsto L, H, C, P, N.$$

The tempered-distribution consumer therefore receives only the finite test-function rows required by chapter 2310; it receives no completed test-function space, unrestricted continuous dual, quotient test-function equality, host distribution equality, selected representative, or countable-choice schedule.

Proof. Project the accepted carrier of Definition 3399.1. The distribution-facing handoff row T is stored after the derivative, window, and polynomial-weight rows, so it cannot be read without those finite tests already being displayed.

The scalar route remains R. It exposes only the rational and real endpoints needed by the finite test row and does not identify host functions or distributions.

Finally L, H, C, P, N repack the same data as a local NameCert surface. Because no row contains a completed Schwartz topology, unrestricted dual, quotient equality, selected representative, or choice schedule, the handoff is exactly the finite test carrier consumed by the tempered-distribution packet. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzFunction}^{\uparrow}$ consumer that reaches a Fourier window, tempered-distribution test row, weak-derivative observation, or $\mathsf{Real}^{\uparrow}$ scalar seal without first reading the derivative window and polynomial-weight ledger refutes the carrier surface.

Independence witness: weakderivative, fourier, tempereddistribution, partitionofunity, regseqrat, real: none is bijective to $\mathsf{SchwartzFunction}^{\uparrow}$ because this packet jointly carries derivative observations, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzFunction}^{\uparrow}$ packet is generated from finite derivative-observation rows, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, a local classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3399.1, Definition 3399.2, Definition 3399.3, and Definition 3399.4 seed the finite Schwartz-test carrier over derivative windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Schwartz spaces, infinite seminorm topologies, host smooth-function equality, arbitrary support selection, quotient test-function equality, Plancherel, inversion, unrestricted continuous duals, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exactness, rapid-decay exactness, Fourier-readiness exactness, tempered-distribution test exactness, scalar readback stability, transport stability, replay stability, and non-escape proofs for BEDC.Derived.SchwartzFunctionUp.

Chapter 2447

A Concrete Naming Certificate for $\mathsf{AsymptoticEquivalence}^{\uparrow}$

Following Hardy on divergent series and Bishop–Bridges constructive analysis, an $\mathsf{AsymptoticEquivalence}^{\uparrow}$ packet names tail-equivalence data for real-series and regular-Cauchy comparison routes by finite thresholds, displayed tail windows, and terminal Real-facing seals. It sits after the partial-sum and tail-budget surface of chapter 1581, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a reusable comparison certificate for two displayed tails; it is not a quotient of series, not a selected sum, not a host limit, and not an ambient completeness principle.

Definition 2447.1 (Asymptotic equivalence carrier)**.** *An* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are the two* $\mathsf{RealSeries}^{\uparrow}$ *tail-source rows whose partial-sum or term tails are compared.*
2. R_0, R_1 *are the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows used by the two tail names.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for ratio, difference, and tail-budget inspections.*
4. T *is the finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window schedule assigning each requested precision a pair of inspected tail windows.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison-seal row reached after the two readbacks and dyadic ledger have been consumed.*
6. H *records componentwise* hsame *transport for the two series rows, two regular readbacks, dyadic ledger, stream windows, terminal seal, continuation, provenance, and local naming rows.*
7. C *records the displayed* Cont *replay from each precision request through* $T, D, R_0, R_1, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$ *row.*

The carrier exposes only finite tail-comparison data. It has no coordinate for a selected infinite sum, quotient equality of series, host limit selection, trichotomy, countable choice, ambient Real completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.AsymptoticEquivalenceUp.AsymptoticEquivalenceTasteGate single carrier alignment`

Theorem 2447.2 (Asymptotic equivalence NameCert obligations)**.** *For an accepted* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier* $A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$ *surface consists exactly of carrier exposure, two real-series tail-source rows, two regular-Cauchy readback rows, the shared dyadic tolerance ledger, the stream-indexed tail-window schedule, the terminal Real comparison seal, componentwise classifier transport, continuation replay, package provenance, and local naming. Each accepted precision request is classified by the finite tail window in* T *and the tolerance ledger in* D *before either regular readback may expose the terminal row* E*.*

Proof. Project the carrier tuple in Definition 3400.1. The analytic rows are $S_0, S_1, R_0, R_1, D, T, E$, and the structural rows are H, C, P, N.

For a precision request, first read the scheduled finite windows in T and the dyadic tolerance ledger D. These rows determine which tails of the two $\mathsf{RealSeries}^{\uparrow}$ sources and which regular-Cauchy readbacks may be inspected.

The classifier transport is componentwise through H, while C replays the same route from the request to the two readbacks and then to E. The provenance row P and local naming row N record only those displayed dependencies, so the NameCert surface cannot import a selected sum, quotient equality, host limit, choice schedule, or ambient completeness proof. □

Theorem 2447.3 (Asymptotic equivalence tail transfer)**.** *Let* $A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier. If a Real or Completion consumer reads a finite tail estimate for* S_0 *through the schedule* T *and ledger* D*, then the same accepted packet exposes the matching finite tail estimate for* S_1 *through the componentwise readback rows* R_0, R_1 *and the terminal comparison seal* E*. The transfer is stable under* hsame *transport of the carrier rows and exports no selected real representative, quotient stream equality, host convergence principle, or completed infinite tail.*

Proof. Start with the accepted schedule row T. It stores a pair of finite tail windows for each precision request, so a consumer that reads the first source tail has already displayed the matching second-source window.

Use the dyadic ledger D to compare the two inspected tails. The readback rows R_0 and R_1 then expose the regular-Cauchy data attached to the two windows, and the terminal row E records the Real-facing comparison seal reached after both readbacks are displayed.

Transport along H preserves the two series rows, two readbacks, ledger, windows, and terminal seal component by component. Replaying C therefore gives the same tail-transfer route to every downstream consumer, while the excluded selected representatives, quotient equalities, host convergence principles, and completed infinite tails remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AsymptoticEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RealSeries}^{\uparrow}$ tail-source rows, two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, a shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ tail windows, a terminal $\mathsf{Real}^{\uparrow}$ comparison seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3400.1, Definition 3400.2, and Definition 3400.3 seed the finite tail-equivalence route over $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient equality of series, host limits, trichotomy, countable choice, ambient Real completeness, bridge checking, or a Lean theorem.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite tail-transfer and NameCert obligation surfaces.

Chapter 2448

A Concrete Naming Certificate for $\mathsf{RieszLemma}^{\uparrow}$

Following Riesz's lemma in normed linear spaces, the $\mathsf{RieszLemma}^{\uparrow}$ packet records the finite BEDC witness by which a proper closed subspace and a displayed separation tolerance expose a normed-space separation row. It sits over the norm carrier of chapter 89, the inner-product comparison surface of chapter 86, the metric observation row of chapter 57, the complete-metric boundary of chapter 128, and the Banach carrier of chapter 90. The certificate names a finite separation witness surface; it is not a quotient subspace construction, not a selected best approximation, and not an ambient compactness or completeness theorem.

Definition 2448.1 (Riesz lemma finite separation carrier)**.** *A* $\mathsf{RieszLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, U, A, B, D, S, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the normed-space source row, read through* $\mathsf{Norm}^{\uparrow}$ *and the vector-space carrier already visible to the packet.*
2. *Y is the displayed proper closed subspace row, named as finite membership and closure evidence rather than as a quotient carrier.*
3. *U is the unit normalization row recording the inspected vector window and its norm-one certificate.*
4. *A is the separation tolerance row, with the requested dyadic margin read through the local* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
5. *B is the ball-avoidance row recording the displayed finite comparisons with the subspace window.*
6. *D is the metric-distance observation row, routed through* $\mathsf{Met}^{\uparrow}$ *and the norm-induced distance surface.*
7. *S is the separation witness row that names the finite vector witness after U, A, B, D have been read.*
8. *K is the completeness or Banach-facing source row, available only as a dependency witness from* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{Banach}^{\uparrow}$ *and not as a selection principle.*
9. *H stores componentwise* hsame *transports for the displayed rows.*

10. *C stores the displayed* Cont *replay route from source and subspace rows to the witness row.*
11. *P is the* Pkg *provenance over the normed source, subspace ledger, tolerance, avoidance, distance, witness, transport, replay, and local naming rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for quotient subspace distance, best-approximation selection, closed-ball compactness, countable choice, or ambient Banach-space completeness.

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaTasteGate single carrier alignment`

Theorem 2448.2 (Riesz lemma NameCert obligations)**.** *For every accepted* $\mathsf{RieszLemma}^{\uparrow}$ *carrier* $R = (X, Y, U, A, B, D, S, K, H, C, P, N)$*, the local NameCert surface consists of:*

1. *carrier admission for the finite normed-separation packet of Definition 3401.1;*
2. *source-row exposure through X, with the norm and metric rows read through chapter 89 and chapter 57;*
3. *subspace exposure through Y, with closure information displayed as finite ledger data rather than quotient subspace distance;*
4. *unit and tolerance exposure through U and A before any avoidance or separation row is visible;*
5. *classifier stability through B, D, S, where ball avoidance, metric observation, and separation witness rows are compared componentwise;*
6. *ledger nonescape through K, H, C, P, N, so completeness, transport, replay, provenance, and naming remain inside the same finite packet.*

These obligations do not include a quotient normed space, a nearest-point projection, a selected distance minimizer, countable choice, compactness of closed balls, or an unrestricted Banach-space separation principle.

Proof. Project the carrier rows of Definition 3401.1. The source, subspace, unit, tolerance, avoidance, distance, witness, completeness-source, transport, replay, provenance, and naming rows are the only data visible to the local classifier.

The normed-separation part is finite. The source row X supplies the normed carrier, Y supplies the displayed subspace ledger, and U, A fix the unit normalization and dyadic tolerance before any witness can be read.

The classifier stability part is read through B, D, S. These rows compare avoidance and distance observations componentwise, so the separation witness cannot be replaced by quotient distance or by a selected best approximation.

Finally K, H, C, P, N bind the route to one local certificate. Since the excluded objects are absent from the carrier, no NameCert obligation can escape through them. □

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaCarrier namecert obligations`

Theorem 2448.3 (Riesz lemma normed separation witness)**.** *Every consumer of an accepted* $\mathsf{RieszLemma}^{\uparrow}$ *packet that asks for the Riesz separation witness factors through*

$$X, Y, U, A, B, D, S, K, H, C, P, N.$$

The read consumes the normed source, closed-subspace ledger, unit normalization, tolerance row, ball-avoidance row, metric-distance observation, separation witness row, completeness-source row,

componentwise transport, replay, provenance, and local naming before any witness is exported. The route exports no quotient subspace distance, selected minimizer, host projection, compactness principle, countable choice, or ambient Banach-space completeness coordinate.

Proof. Begin with the NameCert obligations of Definition 3401.2. They place X, Y, U, A before the avoidance and distance rows, so the requested separation witness is grounded in displayed normed-space and subspace data.

Next read B, D, S. The avoidance row, metric observation row, and separation witness row are compared by the local classifier, which makes the witness stable under componentwise equality of the displayed packet.

Finally use the ledger rows K, H, C, P, N. They record that completeness dependencies, transport, continuation replay, provenance, and local naming stay inside the same finite BHist packet. Hence the consumer receives a finite normed separation witness rather than any of the excluded quotient, choice, projection, compactness, or ambient-completeness objects. □

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaCarrier normed separation witness`

Closure status for $\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszLemma}^{\uparrow}$ packet is generated from a normed-space source row, closed-subspace ledger, unit normalization row, dyadic tolerance, ball-avoidance row, metric-distance observation, separation witness, completeness-source dependency, componentwise transport, continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$, seed)

Scope closed: Definition 3401.1, Definition 3401.2, and Definition 3401.3 bind the finite normed-space separation carrier and its witness nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim quotient subspace distance, nearest-point projection, selected distance minimizers, compactness of closed balls, countable choice, full Banach-space separation, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability for the avoidance and distance rows, ledger nonescape, and local NameCert coverage for the displayed normed separation packet.

Chapter 2449

A Concrete Naming Certificate for $\mathsf{SimplicialApproximation}^{\uparrow}$

Following Brouwer, Alexander, and Spanier on the classical simplicial approximation theorem, the $\mathsf{SimplicialApproximation}^{\uparrow}$ packet records the finite combinatorial route by which a displayed continuous map is replaced by a simplicial map after a displayed subdivision. It sits beside the finite complex surface of chapter 238, the polytope realization boundary of chapter 209, the graph and modulus rows of chapter 130, the Brouwer fixed-point route of chapter 2030, and the deformation ledger of chapter 156. The packet names only the finite subdivision, vertex-image, star-containment, and handoff rows visible to BEDC consumers; it does not import host topology, open-cover compactness, quotient homotopy classes, or choice-selected approximating maps.

Definition 2449.1 (Simplicial approximation carrier)**.** *A* $\mathsf{SimplicialApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, L, S, V, F, T, B, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the source* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, including the finite simplex inventory, face relation, and dimension grading.*
2. *L is the target* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, read through the same finite-complex surface.*
3. *S is the subdivision ledger for K, naming the subdivided complex, the face projection back to K, and the carried simplex-refinement relation.*
4. *V is the vertex-image ledger assigning each vertex of the subdivision to a vertex of L.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map from the polytope realization of K to the polytope realization of L.*
6. *T is the* $\mathsf{Polytope}^{\uparrow}$ *realization row, exposing the barycentric and face-containment data used to read stars and simplex interiors.*
7. *B is the finite star-containment ledger asserting that every subdivided simplex of S is sent by V into a target simplex whose open-star row contains the corresponding F-image window.*
8. *M is the Brouwer-facing handoff row, recording the finite approximation surface consumed by* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *arguments that pass through a combinatorial subdivision.*
9. *H records the* $\mathsf{Homotopy}^{\uparrow}$ *handoff row witnessing that the simplicial map encoded by V, B is homotopic to the displayed continuous map along the visible polytope and star rows.*

10. *C records componentwise* Cont *replay from $K, L, S, V, F, T, B, M, H$ through the local certificate rows.*
11. *P is the* Pkg *provenance over* SimplicialApproximation$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, Polytope$^{\uparrow}$, ContinuousMap$^{\uparrow}$, BrouwerFixedPoint$^{\uparrow}$, Homotopy$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient topological space, an arbitrary open cover, a selected infinite subdivision process, a quotient homotopy class, or a host proof of compactness.

Lean4 ✓ `BEDC.Derived.SimplicialApproximationUp.SimplicialApproximationTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.SimplicialApproximationUp.SimplicialApproximationUpTasteGate single carrier alignment`

Theorem 2449.2 (SimplicialApproximationUp NameCert obligations)**.** *For every accepted* SimplicialApproximation *carrier*

$$A = (K, L, S, V, F, T, B, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$ *surface consists exactly of source-complex admission, target-complex admission, subdivision admission, vertex-image admissibility, continuous-map readback, polytope realization readback, star-containment exactness, Brouwer-facing finite-topology handoff, Homotopy-facing deformation handoff, displayed continuation replay, package provenance, and local naming. No obligation is stored outside the rows of Definition 3402.1.*

Proof. Project the carrier. The source and target finite topological data are exactly K and L, while S refines only the displayed source complex and cannot introduce a new ambient space.

The map data are split into two finite ledgers. The continuous row F supplies the BEDC-visible graph and modulus surface, and the vertex-image row V supplies the candidate simplicial map. The polytope row T is the only realization surface on which both can be compared.

Star-containment exactness is therefore the boundary row B. It checks each subdivided simplex against a target simplex and records the finite containment needed by the classical simplicial approximation route.

The remaining rows do not add new analytic content. The Brouwer handoff M, Homotopy handoff H, continuation replay C, provenance P, and local name N expose where the finite approximation may be consumed. Since no other coordinate exists, the listed rows exhaust the NameCert surface. □

Theorem 2449.3 (Simplicial approximation Brouwer handoff)**.** *Every* BrouwerFixedPoint$^{\uparrow}$ *or* Homotopy$^{\uparrow}$ *consumer of an accepted* SimplicialApproximation$^{\uparrow}$ *carrier receives only the finite route*

$$K, L, S, V, F, T, B \longmapsto M, H \longmapsto C, P, N.$$

The route first exposes the source and target simplicial complexes, the displayed subdivision, the vertex-image ledger, the continuous-map row, the polytope realization row, and the star-containment ledger. Only then may a consumer read the Brouwer-facing finite-topology handoff or the Homotopy-facing deformation handoff.

Proof. Start with an accepted carrier $A = (K, L, S, V, F, T, B, M, H, C, P, N)$. By Definition 3402.2, the consumer must read K, L, S, V, F, T, B before any handoff row is visible.

For a Brouwer-facing consumer, M is the only row naming the finite approximation surface. It is already indexed by the displayed subdivision and star-containment ledger, so it cannot replace the finite route by an open-cover compactness argument or a selected host approximant.

For a Homotopy-facing consumer, H is the only deformation row. It relates the simplicial map encoded by V, B to the continuous row F through the visible polytope data T, rather than through a quotient of maps.

Repacking through C, P, N preserves exactly this finite route. Thus both downstream consumers see the same displayed subdivision, vertex, star, realization, and handoff rows, with no additional host-topology coordinate. □

Falsifiable prediction: A $\mathsf{SimplicialApproximation}^{\uparrow}$ consumer that reaches a Brouwer fixed-point or homotopy conclusion without reading the subdivision row S, vertex-image row V, polytope realization row T, and star-containment row B refutes the carrier surface.

Independence witness: simplicialcomplex, polytope, continuousmap, brouwer-fixed-point, homotopy: no listed sibling supplies the whole twelve-row packet $A = (K, L, S, V, F, T, B, M, H, C, P, N)$, because the siblings contribute finite complexes, realization data, continuous graph rows, fixed-point consumers, or deformation ledgers separately while this chapter binds them into one subdivision-and-star certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimplicialApproximation}^{\uparrow}$ packet is generated from source and target SimplicialComplex rows, a subdivision ledger, a vertex-image ledger, a ContinuousMap row, a Polytope realization row, a star-containment ledger, BrouwerFixedPoint and Homotopy handoff rows, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3402.1, Definition 3402.2, and Definition 3402.3 seed the finite subdivision, vertex-image, star-containment, Brouwer handoff, and Homotopy handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topology, open-cover compactness, quotient homotopy classes, selected infinite subdivision processes, general Brouwer fixed-point existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subdivision stability, vertex-image admissibility, star-containment exactness, Brouwer handoff, Homotopy handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SimplicialApproximationUp.

Chapter 2450

A Concrete Naming Certificate for $\mathsf{MooreSmithNet}^{\uparrow}$

Following Moore–Smith and Kelley nets, the $\mathsf{MooreSmithNet}^{\uparrow}$ packet records the finite directed-net carrier that completion and convergence chapters can inspect before any limit, quotient net, or ambient completeness assertion is named. It sits over the directed-index substrate of chapter 1607, the ordinary net-convergence route of chapter 1667, the uniform-completion consumer of chapter 1473, and the filter-base surface of chapter 1601. The packet is a completion-side neighbour for the L10 directed evidence corridor; it supplies a finite net row, not a selected subnet, not a cofinal-choice operator, and not a host-level limit theorem.

Definition 2450.1 (Moore–Smith net carrier)**.** *A* $\mathsf{MooreSmithNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, V, T, F, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index surface, naming finite index vertices, displayed preorder comparisons, and upper-window witnesses consumed by every net read.*
2. *V is the net-value window, recording the finite values visible at the indices named by D.*
3. *T is the tail ledger, recording the finite late-index requests and the displayed upper-window witnesses that answer them.*
4. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained from the displayed tails of D and T.*
5. *U is the completion-facing consumer row for* $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *reads that inspect the same directed tail surface.*
6. *H records componentwise* hsame *transport for the directed index row, value window, tail ledger, filter-base handoff, completion consumer row, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a net request through D, V, T, F, U.*
8. *P is the* Pkg *provenance over* $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$ *row.*

The carrier exports only this finite directed-net surface. It has no coordinate for quotient nets, selected limits, cofinal choice functions, ambient completeness, or host equality of net values.

Lean4 ✓ `BEDC.Derived.MooreSmithNetUp.MooreSmithNetTasteGate single carrier alignment`

Theorem 2450.2 (Moore–Smith net NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *carrier* $M = (D, V, T, F, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed-index row, value window, tail ledger, filter-base handoff, completion consumer row, transport, replay, provenance, and local naming row;*
2. *directed-tail coverage, requiring every late-index request to read an upper-window witness from* D *and* T *before any value in* V *is consumed;*
3. *filter-base exposure, requiring the* $\mathsf{FilterBase}^{\uparrow}$ *row* F *to be assembled only from the displayed directed tails;*
4. *convergence and completion handoff, requiring* $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumers to inspect the same* D, V, T, F *route before reaching* U*;*
5. *ledger refusal, stating that quotient nets, selected limits, cofinal-choice functions, ambient completeness, and host equality are not packet coordinates.*

Proof. Project the accepted carrier of Definition 3403.1. Its analytic rows are D, V, T, F, U, and its structural rows are H, C, P, N.

Directed-tail coverage is read from D and T. A consumer first names a displayed index request, then reads the upper-window witness in the directed set, and only then inspects the corresponding value window V.

The filter-base row F is downstream of the same directed tails. Since F has no independent source coordinate, the filter-facing consumer receives only the finite tail family already displayed by D, T, V.

The completion row U is reached after this route. Componentwise transport through H and replay through C preserve the same rows, while P and N name them as the local certificate surface. No listed coordinate can export quotient nets, selected limits, cofinal choice, ambient completeness, or host equality. □

Theorem 2450.3 (Moore–Smith net Cauchy completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet that needs Cauchy-style completion data first passes through the directed tails and filter-base handoff of chapter 3794 before the broader completion boundary of chapter 2190 may be exposed. The route reads* $D, V, T, F, U, H, C, P, N$ *as finite net data, Cauchy-filter data, and Moore–Smith completion boundary data, and it does not introduce selected limits, quotient nets, cofinal choice, ambient completeness, or host equality.*

Proof. Start with the obligations of Definition 3403.2. They force every completion-facing read to inspect the directed-index row D, value window V, tail ledger T, and filter-base handoff F before reaching the completion consumer row U.

The Cauchy completion neighbour receives exactly that tail and filter route. Thus a reader of chapter 3794 consumes displayed Moore–Smith net data as a Cauchy-style completion surface rather than as a selected limit.

Only after this Cauchy-filter route is visible can the Moore–Smith completion boundary of chapter 2190 be named. Componentwise transport, replay, provenance, and local naming remain H, C, P, N, so the route cannot add quotient nets, cofinal choice, ambient completeness, or host equality. □

Theorem 2450.4 (Moore–Smith net uniform-completion handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ V,\ T \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}.$$

The handoff exposes directed indices, displayed value windows, finite tail requests, filter-base data, and convergence-facing replay only through $D, V, T, F, U, H, C, P, N$*. It does not supply a quotient of nets, a selected completion point, a cofinal-choice function, host equality of values, or ambient uniform completeness.*

Proof. Begin with the NameCert obligations of Definition 3403.2. The directed-tail coverage row forces the consumer to read D and T before any value window can be used.

Next apply the filter-base exposure row. The handoff to $\mathsf{FilterBase}^{\uparrow}$ is assembled from those displayed tails, so it cannot introduce an external filter, ultrafilter, or quotient of net tails.

The convergence and completion handoff row then sends the same finite route to $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers. Transport and replay remain componentwise through H and C, while P, N record provenance and naming for the same packet. Thus the uniform-completion read is exhausted by the displayed directed-net carrier and cannot import selected limits, cofinal choice, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithNet}^{\uparrow}$ packet is generated from a DirectedSetUp index row, displayed net-value windows, a finite tail ledger, a FilterBaseUp handoff row, NetConvergenceUp and UniformCompletionUp consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$, seed)

Scope closed: Definition 3403.1, Definition 3403.2, and Definition 3403.4 seed the finite Moore–Smith net carrier consumed by completion-side directed evidence.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient nets, cofinal choice, ambient completeness, host equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail coverage, filter-base exposure, convergence handoff, uniform-completion handoff, transport preservation, and ledger refusal for the same finite packet.

Chapter 2451

A Concrete Naming Certificate for $\mathsf{BishopCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionCriterion}^{\uparrow}$ packet records the explicit criterion by which a displayed regular Cauchy route is allowed to enter a completion boundary without quotienting names or selecting limits. It sits between the modulus schedule of chapter 1438, the finite window surface of chapter 126, the regular rational readback of chapter 301, the metric completion packet of chapter 1506, the uniform completion packet of chapter 1473, and the terminal real-name seal of chapter 38. The packet is a finite NameCert criterion for completion consumers; it is not a quotient-completion construction, not a selected-representative principle, not countable choice, not open-cover compactness, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2451.1 (Bishop completion criterion carrier)**.** *A* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, W, R, K, U, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row recording the explicit tail schedule inspected by the criterion.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which source observations are read at the displayed precision requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and tolerance schedule.*
4. *K is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row reached after M, W, R have been displayed.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row reached by the same finite completion route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for Real-facing consumers of the criterion.*
7. *H records componentwise* hsame *transport for the modulus schedule, stream windows, regular readback, metric and uniform completion handoffs, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through M, W, R, K, U, E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$ *row naming the displayed criterion and its permitted consumer reads.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionCriterionUp.BishopCompletionCriterionTasteGate single carrier alignment`

Theorem 2451.2 (Bishop completion criterion NameCert obligations)**.** *For every accepted* BishopCompletionCriteri *carrier $B = (M, W, R, K, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed modulus schedule, stream windows, regular rational readback, metric-completion handoff, uniform-completion handoff, Real seal, transport, continuation replay, provenance, and local naming row;*
2. *explicit Cauchy-modulus exposure, requiring M to be read before any completion boundary K or U is available;*
3. *finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring every completion-facing observation to pass through W and then R at the displayed precision requests;*
4. *completion-seal handoff, requiring K, U, and E to remain downstream of M, W, R and upstream only of the structural rows H, C, P, N;*
5. *ledger non-escape, refusing quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Carrier admission is tuple projection from Definition 3404.1. The accepted packet displays exactly $M, W, R, K, U, E, H, C, P, N$.

The analytic route first reads the explicit modulus schedule M, then the finite windows W, and then the regular rational readback R. The completion rows K and U and the Real seal E are downstream of that displayed route.

Finally H, C, P, N transport, replay, source, and name the same finite criterion. Since quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier row, none can be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3404.1 and Definition 3404.2 seed the finite Bishop completion criterion route from explicit Cauchy moduli through StreamName, RegSeqRat, MetricCompletion, UniformCompletion, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopCompletionCriterionUp.BishopCompletionCriterionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion, selected limits, countable choice, open-cover compactness, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, explicit modulus exposure, finite StreamName and RegSeqRat readback, completion-seal handoff, and ledger non-escape for BEDC.Derived.BishopCompletionCriterionUp.

Constructive story: A $\mathsf{BishopCompletionCriterion}^{\uparrow}$ packet is generated from a CauchyConvergenceModulus tail schedule, finite StreamName windows, RegSeqRat readback, MetricCompletion and UniformCompletion handoffs, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2452

A Concrete Naming Certificate for $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ packet records the finite criterion by which two regular-Cauchy views may be interleaved before a completion-facing consumer reads either view. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, the subsequence criterion of chapter 1552, and the product interleaving route of chapter 2172. The packet is a displayed interleaving criterion for completion consumers; it is not a quotient stream, not a choice-selected limit, and not an ambient completeness coordinate.

Definition 2452.1 (Cauchy interleaving criterion carrier)**.** *A* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows whose observations are to be interleaved.*
2. D_0 *and* D_1 *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers attached to the two windows.*
3. T *is the finite selector schedule recording which side is read at each displayed stage.*
4. W *is the common window-extraction row obtained from the two scheduled sources and their dyadic ledgers.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reconstructed from the extracted interleaving windows.*
6. E *is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after* $S_0, S_1, D_0, D_1, T, W, Q$ *have been displayed.*
7. H *records componentwise* hsame *transport for the two stream windows, two dyadic ledgers, selector schedule, extracted window, readback row, Real seal, continuation, provenance, and local naming rows.*
8. C *records displayed* Cont *replay through the same interleaving route.*
9. P *is the* Pkg *provenance over* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient streams, selected host limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterleavingCriterionUp.CauchyInterleavingCriterionTasteGate single carrier alignment`

Theorem 2452.2 (Cauchy interleaving criterion NameCert obligations)**.** *For every accepted* CauchyInterleavingCrit *carrier* $I = (S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, two-sided finite-window exposure, two dyadic tolerance ledgers, finite selector admission, common window extraction, regular readback handoff, Real-seal boundary exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer reads only this displayed route and receives no quotient stream, selected representative, host convergence predicate, countable-choice selector, or ambient completeness witness.*

Proof. Project the carrier of Definition 3405.1. The rows $S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N$ are the complete displayed packet.

The analytic rows first expose the two finite stream windows and their dyadic tolerance ledgers. The selector schedule T then chooses the displayed side at each stage, and the common extraction row W records the finite window actually read by the interleaving consumer.

The regular readback Q and Real seal E are downstream of that extracted route. The structural rows H, C, P, N transport, replay, source, and name the same finite criterion, so no quotient stream or choice-selected limit can enter the local NameCert surface. □

Theorem 2452.3 (Cauchy interleaving criterion window extraction)**.** *Every accepted* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *packet admits a finite window extraction route*

$$(S_0, S_1, D_0, D_1, T) \longmapsto W \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The extracted window W *is read from the two displayed* $\mathsf{StreamName}^{\uparrow}$ *rows using only the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers and the finite selector schedule* T*. Hence downstream* $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopCompletionCriterion}^{\uparrow}$, *and regular-Cauchy completion consumers cannot replace the route by a quotient of streams, a selected limit, or an ambient completeness coordinate.*

Proof. Begin with the NameCert obligations of Definition 3405.2. They force both source windows and both dyadic ledgers to be visible before the selector schedule can be consumed.

The selector schedule is finite, so each displayed stage chooses one of the two already exposed windows. The common extraction row W records exactly those finite choices, and the regular readback row Q is reached only after W has been displayed.

Finally the Real seal and structural rows replay the same route. Because the carrier has no coordinate for quotient streams, selected representatives, countable choice, host real equality, or ambient completeness, a completion-facing consumer receives only the finite extraction route shown above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two DyadicRatCore tolerance ledgers, a finite selector schedule, an extracted interleaving window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3405.1, Definition 3405.2, and Definition 3405.3 seed the finite interleaving criterion route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyInterleavingCriterion}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyInterleavingCriterionUp.CauchyInterleavingCriterionTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or a completed-real binary operation.

Upgrade path: Obligation closure requires carrier admission, selector-window coverage, dyadic ledger compatibility, regular readback handoff, Real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyInterleavingCriterionUp.

Chapter 2453

A Concrete Naming Certificate for $\mathsf{LocatedRealModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealModulus}^{\uparrow}$ packet records the finite modulus surface that lets a located real answer bounded precision requests without becoming a quotient real or a trichotomy oracle. It sits over the located-real carrier of chapter 1933, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the density-support route of chapter 1809. It also refines the convergence-modulus discipline of chapter 1438 for the Phase Real and Completion context of chapter 9238.

Definition 2453.1 (LocatedRealModulus carrier)**.** *A* $\mathsf{LocatedRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *row whose located interval and endpoint enclosure data answer the precision request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger recording the requested radius, endpoint slack, and comparison budget.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from the located real at the displayed precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, D, W, R have been displayed.*
6. *H stores componentwise* hsame *transport for the located-real row, dyadic ledger, finite window, regular readback, real seal, replay route, provenance, and local naming row.*
7. *C stores the displayed* Cont *replay from a precision request through L, D, W, R to the seal row E and to downstream completion consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealModulus}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, trichotomy, a selected limiting representative, countable choice, host completeness, or an ambient order oracle.

Lean4 ✓ `BEDC.Derived.LocatedRealModulusUp.LocatedRealModulusTasteGate single carrier alignment`

Theorem 2453.2 (LocatedRealModulus NameCert obligations)**.** *For an accepted* $\mathsf{LocatedRealModulus}^{\uparrow}$ *carrier* $M = (L, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, located-real row exposure, dyadic precision-ledger exactness, finite stream-window coverage, regular rational readback compatibility, real-seal non-escape, componentwise transport, continuation replay, provenance, and local naming for the packet of Definition 3406.1.*

Proof. Carrier admission is projection from the displayed tuple. The analytic rows are L, D, W, R, E, and the structural rows are H, C, P, N.

The located-real row L supplies the enclosure surface, while D records the radius and comparison budget for the request. The stream row W and readback row R are opened only at that displayed budget, so the window coverage and rational readback are tied to the same dyadic ledger.

The seal row E is downstream of those reads. Transport and replay preserve the same order through H and C, while P and N add provenance and naming without introducing quotient equality, trichotomy, selected limits, choice, or ambient completeness. □

Theorem 2453.3 (LocatedRealModulus dyadic-window coverage)**.** *Every precision request accepted by a* $\mathsf{LocatedRealModulus}^{\uparrow}$ *packet must display the dyadic ledger* D *before opening the finite stream window* W*, and must display the window* W *before reading the regular rational row* R*. Thus a* $\mathsf{RealSeparability}^{\uparrow}$ *or* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *consumer receives a covered dyadic-window route*

$$D \longmapsto W \longmapsto R$$

inside the same BHist *packet, not a detached density choice, an infinite selector, or a host convergence proof.*

Proof. Start with the carrier inventory of Definition 3406.1. The dyadic ledger precedes the finite-window and readback rows in the accepted packet.

Use the local obligations of Definition 3406.2. They require dyadic precision exactness, window coverage, and regular rational readback compatibility to be certified for the same packet.

The separability and convergence-modulus consumers can therefore cite the displayed route only as finite coverage data. Since the packet contains no density selector, infinite choice schedule, quotient equality, or host convergence coordinate, none can be exported with the covered window. □

Theorem 2453.4 (LocatedRealModulus real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Real and Completion consumer of an accepted* $\mathsf{LocatedRealModulus}^{\uparrow}$ *packet must pass through*

$$L, D, W, R, E, H, C, P, N$$

in that order before the real seal E *is read. The handoff preserves the located-real enclosure, dyadic precision ledger, stream window, and regular readback as visible rows, and it exports no host real equality, trichotomy, selected representative, countable-choice schedule, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the located-real row first. It supplies the enclosure and locatedness data inherited from chapter 1933.

Next read the dyadic-window route of Definition 3406.3. The precision ledger, finite stream window, and regular rational readback are already fixed before the seal row is available.

Finally replay the handoff through C and transport it through H. The provenance and local naming rows record the same packet, so the terminal $\mathsf{Real}^{\uparrow}$ seal is a downstream read of finite modulus data rather than an appeal to quotient completion, choice, trichotomy, selected representatives, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealModulus}^{\uparrow}$ packet is generated from a displayed LocatedRealUp row, a DyadicRatCore precision ledger, a StreamName finite-window row, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3406.1, Definition 3406.2, Definition 3406.3, and Definition 3406.4 seed the finite located-modulus route over LocatedReal, DyadicRatCore, StreamName, RegSeqRat, RealUp, RealSeparability, and CauchyConvergenceModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, trichotomy, selected limits, countable choice, host completeness, or an ambient RealUp completeness theorem.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-window coverage, regular readback compatibility, real-seal handoff, transport stability, continuation replay, provenance, and local naming for BEDC.Derived.LocatedRealModulusUp.

Chapter 2454

A Concrete Naming Certificate for $\mathsf{MonotoneBoundedSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ packet records the finite carrier surface for a monotone sequence of constructive real names together with an explicit boundedness ledger. It sits between the $\mathsf{Real}^{\uparrow}$ rows of chapter 38, the bounded sequence discipline of chapter 1415, the limit interface of chapter 1413, the tail-modulus handoff of chapter 1438, and the located monotone convergence route of chapter 3373. The packet names the data available before a constructive monotone-convergence consumer asks for a limit; it does not smuggle a completed limit into the source carrier.

Definition 2454.1 (MonotoneBoundedSequence carrier)**.** *A* $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Real}^{\uparrow}$ *source-row family for the sequence values, read only through the finite sequence schedule admitted by the packet.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger recording the inspected indices and observation windows for each displayed sequence value.*
3. *R is the real-readback row that ties each scheduled observation in W to the corresponding* $\mathsf{Real}^{\uparrow}$ *source row in S.*
4. *O is the monotonicity ledger, listing the finite comparison rows by which the packet certifies that later scheduled values do not move below earlier scheduled values.*
5. *B is the boundedness ledger, read through the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *surface, which records the common real bound and the finite comparisons keeping every inspected source row below that bound.*
6. *T is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff row exposed to downstream convergence consumers when the monotonicity and boundedness ledgers have been inspected together.*
7. *H records componentwise* hsame *transport for the source rows, stream windows, real readbacks, monotonicity ledger, boundedness ledger, modulus handoff, replay route, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a convergence request through $S, W, R, O, B, T, H, P, N$.*

9. *P is the* Pkg *provenance over* MonotoneBoundedSequence$^\uparrow$, Real$^\uparrow$, BoundedRealSequence$^\uparrow$, RealSequenceLimit$^\uparrow$, CauchyConvergenceModulus$^\uparrow$, LocatedMonotoneConvergence$^\uparrow$, StreamName$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{MonotoneBoundedSequence}^\uparrow}$ *row.*

The classifier compares accepted packets only through

$$S, W, R, O, B, T, H, C, P, N.$$

It has no coordinate for quotient limits, countable choice, trichotomy, selected representatives, or ambient Real$^\uparrow$ *completeness.*

Theorem 2454.2 (MonotoneBoundedSequence NameCert obligations)**.** *For every accepted* MonotoneBoundedSequ *carrier*

$$M = (S, W, R, O, B, T, H, C, P, N),$$

the local NameCert$_{\text{MonotoneBoundedSequence}^\uparrow}$ *surface consists exactly of the finite rows for carrier admission, source-row scheduling, stream-window readback, monotonicity-ledger admission, boundedness-ledger admission,* CauchyConvergenceModulus$^\uparrow$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each obligation reads the same* Real$^\uparrow$ *source rows and the same* StreamName$^\uparrow$ *windows before the boundedness ledger or modulus handoff is exposed. No obligation may appeal to quotient limits, countable choice, trichotomy, selected representatives, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the carrier in Definition 3407.1. Carrier admission exposes precisely the rows $S, W, R, O, B, T, H, C, P, N$, so every source value is read as a scheduled Real$^\uparrow$ row through the displayed stream windows.

The order-facing rows are then forced to use the same source schedule. The monotonicity ledger O compares only scheduled values from S, W, R, while the boundedness ledger B is the BoundedRealSequence$^\uparrow$ read of the same scheduled source rows against a common bound.

The convergence-facing handoff comes last. The CauchyConvergenceModulus$^\uparrow$ row T may be exposed only after the monotonicity and boundedness ledgers have both been inspected, and the RealSequenceLimit$^\uparrow$ and LocatedMonotoneConvergence$^\uparrow$ consumers can read only that displayed handoff.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, choice, trichotomy, representative-selection, and ambient-completeness coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.MonotoneBoundedSequenceUp.TasteGate.MonotoneBoundedSequenceTasteGate single carrier alignment`

Theorem 2454.3 (MonotoneBoundedSequence scheduled-window factorization)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted MonotoneBoundedSequence$^\uparrow$ *carrier. Every consumer that reads the monotonicity ledger O, boundedness ledger B, and* CauchyConvergenceModulus$^\uparrow$ *row T must factor those reads through the same* RealSequenceSpace$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, *and* Real$^\uparrow$ *schedule S, W, R before any* LocatedSupremum$^\uparrow$ *or* LocatedMonotoneConvergence$^\uparrow$ *read is exposed. The factorization exports one displayed finite sequence schedule, not quotient limits, selected representatives, countable choice, trichotomy, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the carrier in Definition 3407.1. The rows S, W, R are the only sequence-value route available to the packet: S lists the source rows, W records the finite stream windows, and R ties those windows to the real readback surface.

Apply Definition 3407.2. It forces the order-facing rows to consume the same source schedule, so O cannot compare a different sequence surface and B cannot introduce a second boundedness schedule.

The modulus row T is downstream of that shared route. Reading it as a $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff therefore uses the same RealSequenceSpace, StreamName, RegSeqRat, and Real rows before any located-supremum or located-monotone-convergence consumer is allowed to inspect the packet.

Since the carrier has no coordinate for quotient limits, selected representatives, countable choice, trichotomy, or ambient real completeness, the shared-window route cannot export any of those objects. □

Theorem 2454.4 (MonotoneBoundedSequence finite limit boundary)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *or* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer of M must first read the* $\mathsf{Real}^{\uparrow}$ *source rows,* $\mathsf{StreamName}^{\uparrow}$ *windows, monotonicity ledger, boundedness ledger,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *through the same finite surface. The consumer receives a limit-boundary route for monotone bounded data, not a quotient limit, countable-choice-selected tail, trichotomy decision, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a limit-facing consumer. By Definition 3407.1, the only available sequence data are the scheduled $\mathsf{Real}^{\uparrow}$ source rows S, their $\mathsf{StreamName}^{\uparrow}$ windows W, and the readback row R.

Apply Definition 3407.2. The monotonicity ledger O and boundedness ledger B are attached to that same scheduled source surface, and the modulus row T is available only after both ledgers have been displayed.

The $\mathsf{RealSequenceLimit}^{\uparrow}$ route may therefore inspect a finite tail-boundary surface, and the $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ route may inspect the same bounded monotone handoff before reaching its located-supremum data. Neither route can replace the packet by a completed real limit or a host-level sequence theorem.

The structural rows H, C, P, N preserve this boundary under transport, replay, provenance, and naming. The absent quotient-limit, choice, trichotomy, representative, and ambient-completeness coordinates remain absent for every downstream consumer. □

Theorem 2454.5 (MonotoneBoundedSequence located convergence source)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer of M must read the same source rows S, stream windows W, real readback row R, monotonicity ledger O, boundedness ledger B,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff T, transport row H, replay row C, provenance row P, and local naming row N before any* $\mathsf{LocatedSupremum}^{\uparrow}$ *or terminal* $\mathsf{Real}^{\uparrow}$ *seal row is exposed. Thus the located monotone convergence route consumes the displayed bounded monotone source surface rather than a second limit object, quotient representative, choice-selected tail, or ambient completeness theorem.*

Proof. Project the carrier of Definition 3407.1. The tuple exposes exactly the rows $S, W, R, O, B, T, H, C, P, N$, so a located convergence consumer has no other source coordinate to inspect.

Apply Definition 3407.2. The monotonicity ledger, boundedness ledger, and $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff are all attached to the same scheduled source rows and stream windows.

Use Definition 3407.4. It places every $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ read downstream of the displayed source rows, order ledgers, modulus handoff, transport, replay, provenance, and local name.

The located monotone convergence packet can therefore consume only this source route before it reaches a $\mathsf{LocatedSupremum}^{\uparrow}$ handoff or a terminal real seal. Since the carrier has no coordinate for a second limit object, quotient representative, choice-selected tail, or ambient completeness theorem, none can be exposed by the handoff. □

Theorem 2454.6 (MonotoneBoundedSequence Cauchy-modulus frontier)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every completion-facing consumer that uses the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff row* T *must first expose the shared* $\mathsf{Real}^{\uparrow}$ *source rows* S*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the readback row* R*, the monotonicity ledger* O*, the boundedness ledger* B*, the transport row* H*, the replay row* C*, the provenance row* P*, and the local* $\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}$ *row* N*. The handoff therefore exports a finite Cauchy-modulus frontier for the displayed bounded monotone source, not a quotient limit, choice-selected tail, trichotomy decision, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 3407.1. The row T is not an independent limit object; it is one coordinate of the displayed tuple $S, W, R, O, B, T, H, C, P, N$.

Apply Definition 3407.2. The $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff is admitted only through the same source rows, stream windows, readback, monotonicity ledger, and boundedness ledger that support the local naming surface.

Use Definition 3407.4 and Definition 3407.5. Together they place every completion-facing or located-convergence read downstream of the displayed carrier surface and before any terminal limit seal.

The structural rows H, C, P, N preserve that route under componentwise transport, replay, provenance, and local naming. Since the carrier has no coordinate for quotient limits, choice-selected tails, trichotomy decisions, selected representatives, or ambient real completeness, the modulus frontier cannot export any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ consumer that claims access to a convergence boundary must display the Real source rows, StreamName windows, monotonicity ledger, boundedness ledger, CauchyConvergenceModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert before any RealSequenceLimit or LocatedMonotoneConvergence read is inspected.

Independence witness: boundedrealsequence, realsequencelimit, cauchyconvergencemodulus, locatedmonotoneconvergence, real: $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a monotonicity ledger, boundedness ledger, and finite modulus handoff before any limit-facing consumer may read the source sequence.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3407.1, Definition 3407.2, Definition 3407.4, and Definition 3407.5 seed the finite source carrier, local obligation surface, limit-boundary route, and located-convergence source handoff for monotone bounded real sequences.

Formal verification: FormalStatus(NameCert$_{\text{MonotoneBoundedSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient limits, countable choice, trichotomy, selected representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, monotonicity-ledger admission, boundedness-ledger admission, CauchyConvergenceModulus handoff, transport replay, provenance, and local naming for BEDC.Derived.MonotoneBoundedSequenc

Constructive story: A MonotoneBoundedSequence$^{\uparrow}$ packet is generated from Real source rows, StreamName windows, real readback, monotonicity and boundedness ledgers, a CauchyConvergenceModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2455

A Concrete Naming Certificate for $\mathsf{AiryFunction}^{\uparrow}$

Following Airy and Olver on the classical Airy differential equation, the $\mathsf{AiryFunction}^{\uparrow}$ packet records the finite certificate route by which Airy-function reads are named through a displayed scalar ODE, coefficient recurrences, real-series windows, and terminal real-name seals. It consumes the real endpoint discipline of chapter 38, the finite partial-sum and tail surface of chapter 1581, the ODE source of chapter 193, and the Picard local-flow route of chapter 1888. The packet is not a host entire-function primitive, an asymptotic expansion theorem, a selected global solution object, or an ambient analytic-completeness claim.

Definition 2455.1 (Airy function carrier)**.** *An* $\mathsf{AiryFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, I, K, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the scalar ODE row for the Airy equation $y'' - xy = 0$, read through the local* $\mathsf{Ode}^{\uparrow}$ *source and its displayed derivative, vector-field, and state-window ledgers.*
2. *I is the initial-data row. It records the chosen Airy branch by finitely displayed values for $y(0)$ and $y'(0)$, including the branch tags that distinguish the standard* Ai *and* Bi *reads when such a distinction is requested.*
3. *K is the coefficient-recurrence row. It records the finite recurrence ledger by which coefficient windows for the Airy power series are generated from D and I.*
4. *T is the* $\mathsf{RealSeries}^{\uparrow}$ *term-stream row, listing the finite partial-sum windows, tail requests, and shared threshold rows used by the requested Airy read.*
5. *S is the local Picard or ODE nonescape row, ensuring that the inspected series and flow windows stay inside the displayed scalar state window supplied by* $\mathsf{PicardLindelof}^{\uparrow}$*.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coefficients, partial sums, initial values, and endpoint requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the ODE, initial-data, recurrence, series, nonescape, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the ODE row, initial data, recurrence ledger, series windows, nonescape row, readbacks, Real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from an Airy request through the ODE row, initial data, coefficient recurrence, real-series windows, Picard nonescape, regular readbacks, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{AiryFunction}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$ *row.*

The carrier exports only this finite route. It has no coordinate for a host entire function, global analytic continuation, asymptotic sector decomposition, selected zero set, quotient function equality, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.AiryFunctionUp.AiryFunctionTasteGate single carrier alignment`

Theorem 2455.2 (Airy function NameCert obligations)**.** *For every accepted* $\mathsf{AiryFunction}^{\uparrow}$ *carrier* $A = (D, I, K, T, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed ODE, initial-data, recurrence, real-series, nonescape, readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *ODE compatibility, requiring D to be the only scalar Airy equation row read by the packet;*
3. *initial-data compatibility, requiring I to supply the branch-determining values consumed by the recurrence ledger;*
4. *recurrence exactness, requiring K to generate every inspected coefficient window from D and I before any series read is exposed;*
5. *real-series visibility, requiring T to expose partial sums, tail thresholds, and endpoint requests only through the* $\mathsf{RealSeries}^{\uparrow}$ *surface;*
6. *local-flow nonescape, requiring S to bind the inspected ODE and series windows to the Picard local-flow route;*
7. *endpoint sealing, requiring R and E to read every requested coefficient, partial sum, and Airy endpoint through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows;*
8. *structural preservation, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no external analytic object.*

No Airy-function obligation is stored outside these rows.

Proof. Project the carrier of Definition 3408.1. The ODE row D and initial-data row I are visible before any recurrence or series row can be inspected, so the requested branch is fixed inside the packet.

The coefficient and series obligations are then finite ledger checks. The recurrence row K supplies the inspected coefficient windows, and the $\mathsf{RealSeries}^{\uparrow}$ row T supplies only the partial sums, tail thresholds, and endpoint requests listed in the packet.

The local-flow and endpoint rows close the analytic route. The nonescape row S keeps the inspected windows inside the displayed Picard or ODE state window, while R and E provide regular rational readback and the terminal real-name seal.

Finally H, C, P, N transport, replay, source, and name the same finite data. Since the carrier has no coordinate for a host entire function, asymptotic sector, selected zero set, quotient function equality, or ambient completeness theorem, the local NameCert surface has no obligation row for such an object. □

Theorem 2455.3 (Airy function real-series handoff)**.** *Every Real-facing read from an accepted* $\mathsf{AiryFunction}^{\uparrow}$ *carrier factors through*

$$D, I \longmapsto K \longmapsto T, S \longmapsto R, E \longmapsto H, C, P, N.$$

Thus a consumer receives the displayed Airy ODE row, branch initial data, coefficient recurrence ledger, finite real-series windows, local nonescape row, regular readback, and terminal real seal, but receives no host entire-function primitive or global analytic continuation datum.

Proof. Start with an accepted packet $A = (D, I, K, T, S, R, E, H, C, P, N)$. The only ODE and branch data available to a consumer are D and I, so a Real-facing read begins at the displayed Airy equation and its carried initial values.

The recurrence row K is the next forced coordinate. It generates the inspected coefficient windows from the ODE and branch data before the $\mathsf{RealSeries}^{\uparrow}$ row T exposes partial sums or tail thresholds.

The nonescape and endpoint rows are downstream. The row S keeps the displayed series and local-flow windows inside the accepted ODE route, and R, E provide the regular readback and terminal $\mathsf{Real}^{\uparrow}$ seal.

The structural rows H, C, P, N preserve and name only that route. Hence the handoff cannot introduce a host entire function, a global continuation object, an asymptotic sector decomposition, a selected zero set, or quotient function equality. □

Falsifiable prediction: Every $\mathsf{AiryFunction}^{\uparrow}$ read factors through one displayed Airy ODE row, one branch initial-data row, one coefficient-recurrence ledger, one $\mathsf{RealSeries}^{\uparrow}$ partial-sum and tail surface, one Picard nonescape row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: real, realseries, ode, picard-lindelof: $\mathsf{AiryFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an Airy ODE row, branch initial data, coefficient recurrence, real-series windows, local-flow nonescape, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AiryFunction}^{\uparrow}$ packet is generated from a displayed Airy ODE row, branch initial data, coefficient-recurrence ledger, RealSeriesUp partial-sum and tail rows, Picard nonescape data, RegSeqRat readbacks, RealUp endpoint seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$, seed)

Scope closed: Definition 3408.1, Definition 3408.2, and Definition 3408.3 seed the finite Airy-function certificate surface over displayed ODE, branch, recurrence, series, nonescape, readback, and real endpoint rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host entire functions, global analytic continuation, asymptotic expansions, selected zero sets, quotient function equality, ambient analytic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ODE compatibility, branch-data compatibility, recurrence exactness, real-series visibility, nonescape, endpoint sealing, and structural preservation for BEDC.Derived.AiryFunctionUp.

Chapter 2456

A Concrete Naming Certificate for $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ packet names the finite comparison surface between a Cauchy presentation of the real line and a located lower-upper presentation. It sits between the Cauchy path of chapter 38, chapter 301, chapter 126, and chapter 291, and the located path of **??**, chapter 404, chapter 1446, and chapter 1447. The packet compares finite observation windows only; it does not add quotient cuts, host real equality, countable choice, trichotomy, or an ambient completeness principle.

Definition 2456.1 (Cauchy-located equivalence carrier)**.** *A* $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (C, L, U, W, R, D, E, H, Ct, P, N)$$

with the following rows:

1. *C is the displayed* $\mathsf{Real}^{\uparrow}$ *Cauchy seal, read through the regular rational and stream-name surface of* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*;*
2. *L is the located lower-observation row, read through* $\mathsf{LowerReal}^{\uparrow}$ *and the lower side of* $\mathsf{LocatedCut}^{\uparrow}$*;*
3. *U is the located upper-observation row, read through* $\mathsf{UpperReal}^{\uparrow}$ *and the upper side of* $\mathsf{LocatedCut}^{\uparrow}$*;*
4. *W is the shared finite-window ledger comparing Cauchy tails, dyadic tolerances, lower observations, and upper observations;*
5. *R records the regular-sequence readback from C into the window ledger;*
6. *D records the dyadic-rational tolerance and endpoint rows consumed by W;*
7. *E records the two handoff routes: Cauchy data to located observations, and located observations back to a Cauchy seal;*
8. *H stores componentwise* hsame *transport for all displayed rows;*
9. *Ct stores the displayed* Cont *replay routes used by the two handoffs and by shared-window comparison;*
10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, L, U, W, R, D, E, H, Ct, P, N$. No coordinate stores quotient-cut equality, host real equality, selected limits, trichotomy, countable choice, or a global completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyLocatedRealEquivalenceUp.TasteGate.CauchyLocatedRealEquivalenceTasteGate single carrier alignment`

Theorem 2456.2 (Cauchy-located equivalence NameCert obligations)**.** *For every accepted* CauchyLocatedRealEqui *carrier $Q = (C, L, U, W, R, D, E, H, Ct, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the Cauchy seal, lower and upper located observations, shared-window ledger, regular readback, dyadic tolerance row, handoff row, transport, continuation, provenance, and local naming row;*
2. *Cauchy-to-located handoff, requiring reads from C through R, D, W, E, H, Ct before lower and upper observations are exported as L, U;*
3. *located-to-Cauchy handoff, requiring compatible lower and upper observations in L, U to pass through the same shared windows W, D, E, H, Ct before the Cauchy seal C is read;*
4. *classifier stability under shared finite windows, requiring agreement on W, R, D, E, H, Ct to preserve both directions of comparison;*
5. $\mathsf{Real}^{\uparrow}$*-seal nonescape, excluding quotient cuts, host real equality, selected limits, trichotomy, countable choice, and ambient completeness from the exported comparison.*

No obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3409.1. The row list displays exactly the Cauchy seal, the lower and upper located observations, the shared-window ledger, the regular and dyadic readback rows, the handoff row, and the structural rows H, Ct, P, N.

For the Cauchy-to-located handoff, read C through the regular sequence row R and the dyadic row D. The shared-window ledger W is the only place where Cauchy tail information is compared with lower and upper observations, and E records the finite route from that comparison to L, U.

For the located-to-Cauchy handoff, begin with compatible reads in L and U. The same ledger W and dyadic row D supply the finite tolerance comparison, while E and Ct replay the route back to the Cauchy seal C.

Classifier stability is componentwise. Agreement on the shared-window, readback, tolerance, handoff, transport, and continuation rows preserves the two handoff routes because no other classifier coordinate is available.

Finally, the provenance and naming rows mention only the listed sibling chapters and BEDC support rows. Since quotient cuts, host real equality, selected limits, trichotomy, countable choice, and ambient completeness are not coordinates of Q, the $\mathsf{Real}^{\uparrow}$ seal cannot export them through this packet. □

Theorem 2456.3 (Cauchy-located equivalence finite handoff)**.** *Every accepted* $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ *carrier $Q = (C, L, U, W, R, D, E, H, Ct, P, N)$ routes comparison between the Cauchy* $\mathsf{Real}^{\uparrow}$ *seal and the located* $\mathsf{LowerReal}^{\uparrow}$-$\mathsf{UpperReal}^{\uparrow}$ *observations through the finite surface*

$$C, L, U, W, R, D, E, H, Ct, P, N.$$

The route may be consumed by $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, *and* $\mathsf{UpperReal}^{\uparrow}$ *readers, but it exports no quotient-cut equality, host real equality, selected limit, trichotomy, countable choice, or ambient completeness.*

Proof. Start with the carrier obligations. They force both directions of comparison to use W, R, D, E, H, Ct, so a consumer cannot jump directly from a Cauchy seal to a located cut or from a located cut to a Cauchy seal.

The Cauchy siblings supply the source rows C, R, D, while the located siblings supply L, U and their cut-facing reads. The handoff row E is the only coordinate that compares the two sides, and the continuation row Ct is the only replay route.

Nonescape follows from the final obligation of Definition 3409.2. Since the forbidden data are absent from the carrier, the finite handoff cannot export them to either side of the comparison. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ packet is generated from a Cauchy $\mathsf{Real}^{\uparrow}$ seal, lower and upper located observations, shared finite windows, regular and dyadic readback rows, two handoff routes, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3409.1, Definition 3409.2, and Definition 3409.3 seed the finite comparison surface between Cauchy and located real presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cuts, host real equality, selected limits, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, both handoff directions, shared-window classifier stability, and Real-seal nonescape for BEDC.Derived.CauchyLocatedRealEquivalenceUp.

$\mathsf{Cauchy}^{\uparrow}$ is an L10 route whose regular-sequence, finite-window, dyadic-tolerance, and terminal seal reads are supplied through chapter 301, chapter 126, chapter 291, and chapter 38.

Chapter 2457

A Concrete Naming Certificate for $\mathsf{CauchyRealLocator}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRealLocator}^{\uparrow}$ packet records the finite locator surface by which a displayed Cauchy real answers Real-facing precision and side-location requests. It refines the modulus route of chapter 1423 and chapter 292, reads finite observations through chapter 126, uses the regular rational handoff of chapter 301, records dyadic tolerances through chapter 291, and reaches the terminal seal of chapter 38. The packet is a traditional constructive-analysis locator object reconstructed as finite NameCert data; it does not select a host limit, identify quotient streams, invoke countable choice, decide trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2457.1 (Cauchy real locator carrier)**.** *A* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row. It records the displayed RealCauchyModulus and CauchyModulus threshold data used for every locator request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the locator at the requested precision and side test.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the positive dyadic scale, comparison cells, and slack budget for the displayed windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows into regular rational data before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *locator seal. It records only the Real-facing locator boundary delivered after M, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic ledger, regular readback, terminal locator seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a locator request through M, S, D, Q, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$*, RealCauchyModulus, CauchyModulus,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $M, S, D, Q, R, H, C, P, N$. It has no coordinate for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 2457.2 (Cauchy real locator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier*

$$L = (M, S, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the modulus schedule, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal Real locator seal, transport, replay, provenance, and local naming;*
2. *modulus exactness, requiring every locator request to use the displayed row M rather than an independent threshold, host convergence theorem, or selected limit;*
3. *window and tolerance compatibility, requiring the inspected windows in S to be compared only through the dyadic ledger D;*
4. *RegSeqRat readback, requiring the rational data consumed by the locator seal to pass through the displayed row Q;*
5. *route ordering, requiring R to be read only after M, S, D, Q have been displayed;*
6. *nonescape, excluding host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, and bridge checking.*

No local locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3410.1. The accepted packet displays exactly the modulus schedule, finite windows, dyadic ledger, regular readback, terminal locator seal, and structural rows.

Modulus exactness is forced by the single coordinate M. Every locator request reads its threshold from this row, so the packet has no second convergence witness, hidden host completeness theorem, or selected limit coordinate.

Window and tolerance compatibility are supplied by S and D. The stream row exposes only finite windows, while the dyadic row records the positive scale, comparison cells, and slack budget used to interpret those windows.

The readback and ordering clauses follow from Q and R. The regular rational row is the only rational readback coordinate, and the terminal Real locator seal occurs downstream of M, S, D, Q.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no row for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2457.3 (Cauchy real locator modulus route)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier factors each locator request through the displayed finite route*

$$M \longmapsto S, D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Thus the request reaches the $\mathsf{Real}^{\uparrow}$ *locator seal only through the visible modulus schedule, finite stream windows, dyadic tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Proof. Begin with an accepted carrier $L = (M, S, D, Q, R, H, C, P, N)$ and a displayed locator request. The only threshold coordinate available to the request is M, so the modulus schedule is read first.

The finite observation step is next. The request opens the window row S and interprets its tolerance through D, whose dyadic ledger records the positive scale and comparison cells.

The rational handoff is then forced through Q. No independent stream or host rational approximation is present in the carrier, so the locator request reaches the Real-facing boundary only after the regular rational readback has been displayed.

Finally the terminal row R is consumed and the structural rows H, C, P, N preserve the same route. This proves the stated factorization and excludes a bypass through selected limits, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Theorem 2457.4 (Cauchy real locator Real seal nonescape)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *locator-seal read reaches* R *only after the displayed rows* M, S, D, Q *have been consumed. The seal exports no host Real equality, quotient stream representative, selected limit, countable-choice selector, trichotomy branch, open-cover compactness witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Inspect the terminal coordinate of the carrier. The only Real-facing locator seal is R, and it is listed after the modulus schedule, stream-window, dyadic-ledger, and regular-readback rows.

Apply the obligation surface of Definition 3410.2. It requires route ordering through M, S, D, Q before R can be read by a Real-facing consumer.

The structural rows preserve this boundary. The hsame transports and Cont replay keep the same displayed packet, while Pkg provenance and local naming record only that route.

Because no excluded coordinate appears in L, the terminal seal cannot export host Real equality, quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. Any consumer asking for one of those principles must leave the local carrier. □

Theorem 2457.5 (Cauchy real locator four-object exit)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *Real-facing locator read factors through exactly the four L10 exit objects named by Definition 3412.8: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *locator seal. In carrier coordinates, any accepted read of* $L = (M, S, D, Q, R, H, C, P, N)$ *uses* S, D, Q, R *as the four object faces and uses* M, H, C, P, N *only as the modulus schedule, transport, replay, provenance, and local naming support. It exports no quotient stream equality, host Real equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier tuple. The row S is the finite $\mathsf{StreamName}^{\uparrow}$ window, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and R is the terminal $\mathsf{Real}^{\uparrow}$ locator seal.

The modulus route theorem Definition 3410.3 fixes the order of consumption. A locator request reads the schedule M, opens the finite windows S, interprets them through the dyadic ledger D, passes through the regular readback Q, and only then reaches R.

The Real-seal nonescape theorem Definition 3410.4 supplies the boundary condition for the terminal read. It rules out every Real-facing bypass that would identify quotient streams, appeal to host Real equality, choose a limit, branch by trichotomy, use open covers, or import ambient completeness.

Finally compare this carrier projection with Definition 3412.8. The four object faces are exactly S, D, Q, R, while M, H, C, P, N supply only schedule, transport, replay, provenance, and naming support. Hence an accepted locator read has the claimed four-object L10 exit route and no additional exported coordinate. □

Theorem 2457.6 (Cauchy real locator induction route)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every locator-side read available to induction is obtained by the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

The induction step reads finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal rows without deciding trichotomy and without choosing a host limit.*

Proof. Project the carrier of Definition 3410.1. The modulus schedule M is the only threshold source, so a locator-side induction step begins by reading the same schedule used in the modulus route.

The finite observation is then S interpreted through D. The stream row exposes only bounded windows, while the dyadic row supplies the tolerance scale and comparison cells needed to use those windows.

The rational and Real-facing parts follow through Q and R. The modulus route theorem Definition 3410.3 fixes this order, and the four-object exit theorem Definition 3410.5 identifies S, D, Q, R as the four L10 exit faces.

Finally H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the carrier has no coordinate for trichotomy or selected host limits, the induction route cannot decide a side by trichotomy or choose a limit outside the displayed locator data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3410.1, Definition 3410.2, Definition 3410.3, Definition 3410.4, Definition 3410.5, and Definition 3410.6 seed the finite locator route over Cauchy modulus schedules, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and the Real locator seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, window and tolerance compatibility, RegSeqRat readback, route ordering, nonescape, and local NameCert registration for BEDC.Derived.CauchyRealLocatorUp.

Constructive story: A $\mathsf{CauchyRealLocator}^{\uparrow}$ packet is generated from a RealCauchyModulusUp and CauchyModulusUp schedule row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp locator seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2458

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 2458.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2458.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 2458.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 2458.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2458.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 2458.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 2458.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2459

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2459.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through $X, Y, W, D, A, M, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2459.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation budgets,* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route,* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 2459.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 2459.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 2459.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 2459.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Cauchy$^{\uparrow}$ gateway reads its completion-facing source through chapter 126, chapter 301, chapter 291, and chapter 38 before any terminal seal is public.

Theorem 2459.7 (Cauchy L10 sibling route)**.** *The Cauchy packet family exported by this hub is consumed through the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

Concretely, the finite-window surface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the completion boundary of chapter 1506 are sibling inputs to the Cauchy carriers rather than hidden ambient structure.

Proof. Read any Cauchy carrier exported by the hub through its finite observation rows. The stream coordinate supplies the inspected windows, the regular-rational coordinate supplies the numerical readback, and the dyadic coordinate supplies the tolerance budget.

The Real-facing seal is reached only after those rows have been displayed. When a completion consumer is present, it receives the same displayed route through the metric-completion boundary.

Thus the hub's Cauchy packet family is attached to the Real and MetricCompletion siblings by visible L10 rows, with no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient completeness coordinate added by the route. □

The Cauchy$^{\uparrow}$ route consumes the L10 sibling chapters before the real locator and sequence algebra inputs are read. In particular, the regular-rational source of chapter 301, the finite stream windows of chapter 126, and the terminal seal of chapter 38 are already part of the displayed dependency spine for every downstream Cauchy locator or algebra consumer. The completion-side consumer in chapter 1937 reads the same RegSeqRat, StreamName, and RealUp spine before exposing its functorial completion route.

Theorem 2459.8 (Cauchy L10 four-object seal route)**.** *Every accepted* Cauchy$^{\uparrow}$ *Real-facing seal route used by the current L10 phase factors through the four displayed sibling faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Concretely, the finite window family of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38 are the only Real-facing sibling rows exported by the route. The supporting rows are the displayed BHist *carrier data, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *registration.*

Proof. Read the route through the existing CauchyRealLocator carrier and its Real-seal nonescape theorem. The carrier exposes a finite StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, and a terminal RealUp locator seal before any Real-facing consumer can read the packet.

Project those four named faces. The StreamName row supplies only inspected finite windows, the DyadicRatCore row supplies the tolerance and comparison budget, the RegSeqRat row supplies the rational readback, and the RealUp row supplies the terminal seal.

The structural rows preserve this factorization. The BHist data name the carried packet, hsame transports the displayed coordinates, Cont replays the same route, Pkg records provenance, and NameCert records the local accepted surface.

Since the carrier has no coordinate for quotient streams, host equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the Cauchy Real-facing seal route. □

Chapter 2460

A Concrete Naming Certificate for $\mathsf{Cauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{Cauchy}^{\uparrow}$ packet records the ordinary constructive Cauchy-real carrier as finite BEDC naming data. It sits above the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational sequence surface of chapter 301, and the terminal seal of chapter 38. The packet is not a quotient construction of reals; it is a finite route by which a modulus, rational readback, transport, replay, provenance, and local naming row become visible before any $\mathsf{Real}^{\uparrow}$ consumer can read the Cauchy object.

Definition 2460.1 (Cauchy carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the finite windows of rational approximants inspected by the carrier.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the positive dyadic scale, comparison cells, and slack conversions used to interpret each requested window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational readback consumed downstream.*
4. *M is the modulus row, recording the displayed threshold assignment used by Cauchy requests and refusing any hidden convergence theorem.*
5. *Q is the modulus/readback compatibility row, requiring the stream window, dyadic tolerance, regular rational sequence, and modulus rows to agree at every displayed precision request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after S, D, R, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-sequence, modulus, compatibility, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy request through $S, D, R, M, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $S, D, R, M, Q, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyUp.CauchyCarrier`

Theorem 2460.2 (Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *carrier*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 3413.1.*

Proof. Carrier admission is projection from Definition 3413.1. The accepted packet displays precisely the stream, dyadic, regular-sequence, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

The stability and exactness obligations are rowwise. The stream row supplies only finite windows, the dyadic row supplies only displayed tolerance cells, the regular-sequence row supplies only rational readback, and the modulus row supplies only the visible threshold assignment.

The compatibility row Q binds these four data rows before any $\mathsf{Real}^{\uparrow}$ seal can be consumed. The structural rows H, C, P, N then transport, replay, source, and name the same route without adding quotient equality, selected limits, host real equality, choice, trichotomy, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCarrier namecert obligations`

Theorem 2460.3 (Cauchy modulus readback route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer route factors through the displayed path*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a precision request reads finite $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the explicit modulus row, the compatibility row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal-facing boundary.*

Proof. Begin with an accepted carrier $X = (S, D, R, M, Q, E, H, C, P, N)$ and a displayed precision request. The only finite observations available to the request are the windows in S, and their tolerance scale is read through D.

The request next reads the explicit modulus row M. The compatibility row Q is the only coordinate that relates the chosen window, tolerance, threshold, and regular-rational readback, so the route cannot bypass it.

After Q is displayed, the consumer may read the $\mathsf{RegSeqRat}^{\uparrow}$ row R and then the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row E. The rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming.

Since the carrier has no quotient, selected-limit, host equality, countable-choice, trichotomy, open-cover, or ambient-completeness coordinate, the displayed route is the whole public Cauchy readback surface. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusReadbackRoute`

Theorem 2460.4 (Cauchy StreamName RegSeqRat window)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet exposes its* $\mathsf{RegSeqRat}^{\uparrow}$ *row only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and modulus/readback compatibility row. Thus a regular rational read of* $X = (S, D, R, M, Q, E, H, C, P, N)$ *must factor through* S, D, M, Q *before it can inspect* R*.*

Proof. Project the carrier of Definition 3413.1. The row S is the only finite observation surface, D is the only displayed tolerance surface, and M is the only modulus row available to the packet.

Apply Definition 3413.2 to identify Q as the compatibility row binding stream windows, dyadic tolerances, regular rational readback, and modulus data. Then apply Definition 3413.3: the displayed route reads S, D, M and Q before R.

The carrier has no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient completeness coordinate. Hence the stated window and compatibility rows exhaust the public access path to $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 2460.5 (Cauchy finite-tail criterion)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

be an accepted $\mathsf{Cauchy}^{\uparrow}$ *carrier. Suppose every displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window admitted by* S *and every dyadic tolerance cell admitted by* D *satisfy the tail bound named by the modulus row* M*, and suppose the compatibility row* Q *identifies those windows with the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of* X *is routed through the same finite tail ledger*

$$S, D, M \longmapsto Q \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The criterion exports no selected limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows available for a tail read are the finite StreamName window row S, the DyadicRatCore tolerance row D, the modulus row M, the compatibility row Q, the RegSeqRat readback row R, and the terminal seal row E.

Induct over the displayed window list retained by S. The empty window has no readback outside the recorded tolerance ledger. In the successor step, the new inspected position is checked against the dyadic cell in D and the threshold supplied by M, so the same finite tail bound remains attached to the extended window.

Apply Definition 3413.4. It forces any access to R through the displayed S, D, M, Q route. Apply Definition 3413.3 to identify Q as the point where the window, tolerance, modulus, and readback rows are jointly inspected.

Finally use Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal E is downstream of the regular rational readback, so no Real-facing consumer can bypass the finite tail ledger. Since the carrier has no selected-limit, quotient-stream, countable-choice, trichotomy, or ambient-completeness coordinate, the displayed finite induction exhausts the criterion. □

Definition 2460.6 (Cauchy finite-window modulus carrier)**.** *The finite-window modulus carrier associated to an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet is the displayed handoff surface*

$$W = (S, D, M, Q, R, E, H, C, P, N)$$

read from the carrier of Definition 3413.1. Its rows have the following roles:

1. *S records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by a tail request.*

2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells and slack conversions attached to those windows.*
3. *M records the explicit modulus threshold row used to accept later-window requests.*
4. *Q records the compatibility row tying each displayed window, dyadic tolerance, modulus threshold, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
5. *R records the regular-rational readback exposed only after S, D, M, Q have been displayed.*
6. *E records the downstream* $\mathsf{Real}^{\uparrow}$ *seal-facing row.*
7. *H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming for the same finite route.*

The carrier admits only these displayed coordinates. It has no coordinate for quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2460.7 (Cauchy finite-window tail induction)**.** *Let $W = (S, D, M, Q, R, E, H, C, P, N)$ be the finite-window modulus carrier of Definition 3413.6. Every accepted tail request obtained by extending the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row preserves the dyadic tolerance row D, the modulus threshold row M, the compatibility row Q, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. Thus the induction route is*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E$$

with H, C, P, N transporting, replaying, sourcing, and naming the same displayed handoff.

Proof. Project the underlying accepted Cauchy carrier through Definition 3413.1 and read it in the window/modulus order specified by Definition 3413.6. The only tail-request rows available are $S, D, M, Q, R, E, H, C, P, N$.

Induct over the displayed $\mathsf{StreamName}^{\uparrow}$ windows retained by S. The empty window has no tolerance request outside D and no readback outside the route named by Definition 3413.4.

For the successor step, the new inspected window is accepted only through the dyadic cell in D and the threshold supplied by M. Apply Definition 3413.5 to keep the same finite tail ledger attached to the extended request.

The compatibility row Q is then the unique displayed row where the window, dyadic tolerance, modulus threshold, and regular readback are jointly inspected. Apply Definition 3413.3 to place R after Q and E after R.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Since none of the rows supplies quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the induction cannot expose any such principle. □

Theorem 2460.8 (Cauchy RegSeqRat handoff obligation)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the finite-window modulus carrier of Definition 3413.6 and the tail-induction route of Definition 3413.7. In particular, the consumer reads S, D, M, Q before R, and any Real-facing use reads E only after that* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Begin with an accepted Cauchy packet and project its rows by Definition 3413.1. The carrier supplies finite stream windows, dyadic tolerances, a modulus row, a compatibility row, regular-rational readback, a terminal Real-facing seal, and structural transport, replay, provenance, and naming rows.

Apply Definition 3413.4. It forces access to the $\mathsf{RegSeqRat}^{\uparrow}$ row through the displayed S, D, M, Q route, so no consumer can inspect R independently of the finite window and tolerance data.

Next apply Definition 3413.5 and Definition 3413.7. These identify the finite tail ledger and show that accepted extensions preserve the dyadic tolerance, modulus threshold, compatibility, and regular readback rows before the Real seal is exposed.

Finally use Definition 3413.3. The structural rows H, C, P, N transport and name the same handoff, while the carrier contains no quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ consumer factors through the stated finite-window induction surface. □

Theorem 2460.9 (Cauchy RegSeqRat Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal real seal. In the notation of Definition 3413.1, the seal-facing route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

It does not expose quotient streams, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $X = (S, D, R, M, Q, E, H, C, P, N)$ from Definition 3413.1. The seal row E is declared terminal and seal-facing; it is not an independent source of Cauchy data.

Use Definition 3413.3 to order the displayed read. The route reaches R after S, D, M, Q and reaches E only after R has been displayed.

Finally apply Definition 3413.2. The rows H, C, P, N transport, replay, source, and name the same route, so they cannot insert quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient completeness between $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 2460.10 (Cauchy modulus transport stability)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

and

$$X' = (S', D', R', M', Q', E', H', C', P', N')$$

be accepted $\mathsf{Cauchy}^{\uparrow}$ *packets. Suppose the stream row, dyadic tolerance row, modulus row, modulus/readback compatibility row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, replay row, provenance row, and local naming row are transported componentwise by the displayed* hsame *data in* H *and* H'*, and suppose both packets are replayed through the same displayed* Cont *route. Then the public Cauchy readback route*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

is the same BEDC route as

$$S', D', M' \longmapsto Q' \longmapsto R' \longmapsto E' \longmapsto H', C', P', N'.$$

In particular, componentwise transport of the modulus/readback surface cannot introduce quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, or ambient completeness.

Proof. Project both accepted packets through Definition 3413.1. The only visible coordinates are the stream, dyadic tolerance, regular-rational readback, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

Apply Definition 3413.3 to X and to X'. The two projected routes must read their displayed stream, dyadic, and modulus rows before their compatibility rows, and the shared Cont replay identifies these reads through the componentwise hsame data.

Next use Definition 3413.4. Access to R and R' is forced through the transported S, D, M, Q and S', D', M', Q' rows, so no other regular-rational readback path is available.

Finally apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal remains downstream of the transported $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the structural rows H, C, P, N and H', C', P', N' only transport, replay, source, and name that path. Since neither carrier contains quotient streams, selected limits, host equality, choice, trichotomy, or ambient completeness, the two displays name the same BEDC route. □

Theorem 2460.11 (Cauchy uniform-modulus Real seal)**.** *For an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

any uniform-modulus consumer of the explicit modulus row M *that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must read* S, D, M, Q *before* R*, and* R *before* E*. The route contains no quotient stream equality, selected limit, countable choice, trichotomy, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 3413.1. The explicit modulus row M is packaged with the finite stream window S, the dyadic tolerance row D, the compatibility row Q, the regular rational readback R, and the terminal seal-facing row E.

Apply Definition 3413.3. It forces a precision request to read S, D, M and then Q before any regular-rational row can be inspected.

Next apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of R, so a uniform-modulus consumer cannot reach the seal before the regular rational readback has been displayed.

Finally use Definition 3413.10. Componentwise transport and the displayed Cont replay preserve the same route for every accepted consumer, and the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2460.12 (Cauchy root filter-completion anchor)**.** *Every Cauchy root consumer that first reaches the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *surface of chapter 1569 must already have read the Cauchy core route, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Project the root hub through its Cauchy core child. The core route exposes the stream-window, tolerance, readback, transport, replay, provenance, and naming rows before any completion-facing consumer can inspect the root packet.

Now read the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ chapter. Its completion handoff consumes the same finite windows and regular rational readback before the terminal Real seal is visible, so the root consumer cannot enter through a detached filter object.

The structural rows preserve the path. Componentwise hsame transport and displayed Cont replay keep the consumer inside the Cauchy root packet, while Pkg provenance and the local $\mathsf{NameCert}$ row record that no quotient filter, selected limit, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported. □

Theorem 2460.13 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 2460.14 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream

consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 2460.15 (Cauchy finite-window modulus pullback)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *modulus read pulls back to the same finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any consumer can inspect a Cauchy value. In the notation of Definition 3413.1, the pullback route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No quotient stream equality, selected limit, host real equality, countable-choice coordinate, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is available along this route.*

Proof. Project the accepted carrier through Definition 3413.1. The finite stream window S, dyadic tolerance row D, modulus row M, compatibility row Q, regular-rational readback R, and terminal Real seal E are the only rows available before the structural rows are read.

Apply Definition 3413.2. It identifies the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ surface with stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility, Real seal ordering, transport, replay, provenance, and local naming.

Now use Definition 3413.3. A precision request first reads S, D, M, Q, then reaches R, and only then reaches E. The route is sharpened by Definition 3413.4, which keeps the $\mathsf{RegSeqRat}^{\uparrow}$ row behind the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, modulus, and compatibility rows.

Finally apply Definition 3413.9. The terminal Real seal is reached only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback has been displayed. Thus every accepted modulus read pulls back to the same four L10 sibling rows before a consumer sees a Cauchy value, and none of the excluded quotient, selected-limit, choice, order, compactness, or completeness coordinates can be inserted. □

Theorem 2460.16 (Regular-Cauchy modulus pullback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of the* $\mathsf{Cauchy}^{\uparrow}$ *finite-window modulus route obtains a dyadic tolerance schedule that is cofinal with the route displayed in Definition 3413.15. In the notation of Definition 3413.1, the consumer may read*

$$S, D, M, Q \longmapsto R \longmapsto E$$

only as a regular-rational handoff: the $\mathsf{Real}^{\uparrow}$ *seal is reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, not before it. The route exports no quotient stream equality, selected limit, host real equality, choice principle, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3413.15. It fixes the finite route from the StreamName window S, dyadic tolerance row D, modulus row M, and compatibility row Q into the regular-rational readback R, and only then into the terminal Real seal E.

Project the route at the $\mathsf{RegSeqRat}^{\uparrow}$ boundary. A RegSeqRat-facing consumer reads the dyadic tolerance row D through the same finite windows S and compatibility row Q, so its tolerance requests are bounded by a cofinal schedule already displayed by the Cauchy modulus route.

Transport the schedule through the regular-rational classifier row. The readback row R is the first row that presents the schedule as regular-rational data, and the Real seal E is reachable only after that readback has been displayed.

Thus the pullback is cofinal at the RegSeqRat surface and terminal at the Real seal. Since the carrier has no coordinate for quotient stream equality, selected limits, host equality, choice,

trichotomy, or Real completeness, none of those principles can be inserted into the consumer route. □

Theorem 2460.17 (Cauchy finite-window pullback uniqueness)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing Cauchy modulus consumer accepted by the* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the same finite route*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

displayed in Definition 3413.15. The consumer cannot replace this route by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows visible to a modulus consumer are the finite StreamName window, dyadic tolerance ledger, modulus row, compatibility row, RegSeqRat readback, Real seal, and the structural rows.

Apply Definition 3413.2. It identifies the local NameCert surface with precisely those rows, so there is no second carrier coordinate from which an alternative Real-facing route could be read.

Use Definition 3413.3. A precision request reaches R only after S, D, M, Q have been displayed, and it reaches the terminal seal E only after the regular-rational readback has been displayed.

Finally apply Definition 3413.15 and Definition 3413.16. The first theorem fixes the finite pullback route to the L10 sibling rows, while the second shows that every RegSeqRat-facing consumer receives the same cofinal handoff. Hence any Real-facing Cauchy modulus consumer factors through the displayed route and gains no quotient, selected-limit, choice, host-equality, or completeness coordinate. □

Every completion-facing consumer of the $\mathsf{Cauchy}^{\uparrow}$ hub reads the finite Cauchy packet through chapter 301 and chapter 38 before any Real seal is exposed to completion rows. The route first displays the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the dyadic tolerance ledger, then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal, and only then lets Cauchy completion consumers inspect the same BHist, hsame, Cont, Pkg, and local NameCert rows. Thus the hub records a structural dependency route, not quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 2460.18 (Cauchy RegSeqRat Real forward dependency)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *source-window read consumed by a Real-facing downstream chapter must pass through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is exposed. The route reads only displayed finite windows, regular rational tail data, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier route in Definition 3413.1. Its source-window reads are displayed before the regular rational readback and terminal real seal rows are available.

Use the regular readback and real-seal routes already named by the Cauchy child files. They order downstream consumption through $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$, with the structural rows transporting and naming the same finite windows.

Since the hub exposes only child-file surfaces and no independent carrier coordinate, a Real-facing consumer can cite $\mathsf{Cauchy}^{\uparrow}$ only through that forward dependency route. The read therefore reaches chapter 301 and chapter 38 before any Real-facing seal is exposed. □

Theorem 2460.19 (Cauchy located-uniform filter forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *route whose explicit modulus row is consumed by a located or uniform Cauchy filter completion must pass through the displayed carrier route*

$$S, D, M, Q, R, E, H, C, P, N$$

of Definition 3413.1, then through chapter 1661 or chapter 1694, before any $\mathsf{Real}^{\uparrow}$ *seal-facing completion row is exported. The forward route has no quotient stream equality, selected limit, countable choice, metric-completeness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host equality.*

Proof. Start with the explicit modulus readback of the Cauchy carrier. By Definition 3413.3, a precision request reads the finite StreamName row S, the DyadicRatCore row D, the explicit modulus row M, and the compatibility row Q before it can inspect the regular rational row R.

Apply Definition 3413.9. The seal-facing row E is downstream of R, so a Real-facing completion consumer cannot bypass the regular rational readback.

Now use Definition 3413.11. Any uniform-modulus consumer of M that reaches a $\mathsf{Real}^{\uparrow}$ seal preserves the same ordering S, D, M, Q before R and R before E.

The remaining completion-neighbour branch is supplied only by the sibling chapters chapter 1661 and chapter 1694. These chapters consume the already displayed Cauchy route as located or uniform filter evidence; neither adds quotient stream equality, selected limits, choice, metric-completeness data, ambient completeness, or host equality to the core Cauchy carrier. □

Theorem 2460.20 (Cauchy route to CauchyCriterion)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *hub's finite route is consumed by chapter 392 through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exports no detached completion carrier, quotient sequence equality, selected modulus outside the displayed rows, countable-choice schedule, or host completeness principle.*

Proof. Read the hub through its existing finite siblings. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows provide the only Cauchy data visible to downstream criteria.

The structural rows hsame, Cont, Pkg, and NameCert transport, replay, source, and name that same finite route. Therefore the Cauchy criterion consumer receives exactly the displayed BEDC rows and no external completion or choice object. □

Theorem 2460.21 (Cauchy hub Bishop completion forward route)**.** *The regular-source material routed by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by chapter 1639 through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The forward route exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness principle.*

Proof. Read the Cauchy hub through its regular-source children. Those children expose dyadic tolerance arithmetic, finite stream windows, regular rational readback, and the terminal real seal as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The Bishop regular Cauchy completion chapter consumes exactly that L10-facing route. Its completion read is mediated by the same finite BHist carrier discipline, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Since the hub route contains no detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness coordinate, the forward edge to the Bishop completion packet is precisely the displayed finite dependency route. □

Chapter 2461

A Concrete Naming Certificate for $\mathsf{FiniteSimplicialChain}^{\uparrow}$

Following Eilenberg–Steenrod and the standard finite presentation of simplicial homology, the $\mathsf{FiniteSimplicialChain}^{\uparrow}$ packet records the BEDC route from a finite $\mathsf{SimplicialComplex}^{\uparrow}$ carrier to the chain groups and boundary rows consumed by $\mathsf{Homology}^{\uparrow}$. It sits over the simplex inventory of chapter 238, the finite enumeration surface of chapter 112, and the additive carrier rows of chapter 42; it supplies $\mathsf{Homology}^{\uparrow}$ and $\mathsf{PersistentHom}^{\uparrow}$ with a displayed finite chain-complex source rather than an implicit classical complex.

Definition 2461.1 (Finite simplicial chain carrier)**.** *A* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *carrier over a finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$C = (K, F, A, G, B, Z, H, T, P, N)$$

with the following displayed rows:

1. *K is the source* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, including the finite simplex inventory, face relation, and dimension grading.*
2. *F is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration row for oriented simplex generators in each displayed dimension.*
3. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *row for finite integer combinations of oriented simplex generators, with addition and inverse read componentwise.*
4. *G is the grading row assigning each chain generator to its simplex dimension.*
5. *B is the boundary ledger sending an oriented n-simplex to the alternating finite face sum in dimension $n-1$.*
6. *Z is the square-zero cancellation ledger, pairing each codimension-two face occurrence with the opposite occurrence obtained by deleting the two vertices in the other order.*
7. *H records componentwise* hsame *transport for simplex inventories, generator enumerations, additive rows, grading rows, boundary rows, cancellation rows, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from K, F, A, G, B, Z, H to the chain-complex source read by* $\mathsf{Homology}^{\uparrow}$*.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSimplicialChain}^{\uparrow}$*,* $\mathsf{SimplicialComplex}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{AbGroup}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite chain module, a host quotient group, a selected orientation convention outside F, an ambient topological realization, or a homology computation not routed through B and Z.

Lean4 ✓ `BEDC.Derived.FiniteSimplicialChainUp.TasteGate.FiniteSimplicialChainTasteGate single carrier alignment`

Theorem 2461.2 (Finite simplicial chain boundary square-zero)**.** *For every accepted* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *carrier $C = (K, F, A, G, B, Z, H, T, P, N)$, the displayed boundary row B composes with itself to the zero row of the* $\mathsf{AbGroup}^{\uparrow}$ *chain carrier. Equivalently, every* $\mathsf{Homology}^{\uparrow}$ *consumer receives a chain-complex source whose boundary-square obstruction is discharged by the finite cancellation ledger Z before cycles and boundaries are compared.*

Proof. Project the carrier of Definition 3414.1. The simplex source K and finite generator enumeration F make every boundary expansion a finite face list, and the additive row A is the only group operation available for collecting those face terms.

Apply the boundary ledger B once to an oriented simplex. The result is the alternating sum of its displayed codimension-one faces. Applying B again expands each face into codimension-two faces, still inside the finite enumeration supplied by F.

The cancellation ledger Z pairs the term obtained by deleting vertex i and then vertex j with the term obtained by deleting j and then i. The two terms have the same codimension-two face row under hsame and opposite signs in the $\mathsf{AbGroup}^{\uparrow}$ carrier, so they add to the zero row.

Since all codimension-two occurrences are exhausted by these pairs, the second boundary has no unpaired summand. The continuation, provenance, and local naming rows T, P, N repack the same finite cancellation into the $\mathsf{Homology}^{\uparrow}$ source surface, so no host chain module or external quotient argument is used. □

Falsifiable prediction: A $\mathsf{Homology}^{\uparrow}$ or $\mathsf{PersistentHom}^{\uparrow}$ consumer using $\mathsf{FiniteSimplicialChain}^{\uparrow}$ must display the source $\mathsf{SimplicialComplex}^{\uparrow}$ row, finite generator enumeration, additive chain carrier, boundary ledger, square-zero cancellation ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: simplicialcomplex, finset, abgroup, homology: $\mathsf{FiniteSimplicialChain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries simplex inventory, oriented generator enumeration, additive chain rows, boundary rows, square-zero cancellation, Homology-facing replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSimplicialChain}^{\uparrow}$ packet is generated from a finite SimplicialComplex row, FinSet generator enumeration, AbGroup additive chain rows, grading, a finite alternating boundary ledger, codimension-two cancellation pairs, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$, seed)

Scope closed: Definition 3414.1 and Definition 3414.2 seed the finite simplicial chain carrier and its square-zero boundary route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteSimplicialChainUp.TasteGate.FiniteSimplicialChainTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close singular chains, infinite chain complexes, host quotient homology, selected geometric realization, or bridge checking.

Upgrade path: Obligation closure requires finite generator enumeration, additive chain stability, boundary exactness, square-zero cancellation, Homology handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.FiniteSimplicialChainU

Chapter 2462

A Concrete Naming Certificate for $\mathsf{RealModulusCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealModulusCriterion}^{\uparrow}$ packet records the finite criterion by which an explicit Cauchy modulus makes a regular rational stream ready for a $\mathsf{Real}^{\uparrow}$ convergence read. It sits between the convergence schedule of chapter 1438, the sequence-limit consumer of chapter 1413, the regular rational names of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38.

Definition 2462.1 (Real modulus criterion carrier)**.** *A* $\mathsf{RealModulusCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, Q, W, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row naming the explicit tail schedule.*
2. S *is the sequence-source row inspected by the modulus.*
3. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback exposed by the scheduled tails.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used to read each tail comparison.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the modulus requests.*
6. L *is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer row reached only after* M, S, Q, W, D *have been displayed.*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *convergence-readiness seal exported to real-name consumers.*
8. H *records componentwise* hsame *transport for the analytic rows and the structural rows.*
9. C *records displayed* Cont *replay from the tolerance request through the modulus, windows, regular readback, sequence-limit consumer, and real seal.*
10. P *is the* Pkg *provenance over* $\mathsf{RealModulusCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. N *is the local* $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completion, selected representatives, countable choice, trichotomy, host sequence convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealModulusCriterionUp.RealModulusCriterionTasteGate single carrier alignment`

Theorem 2462.2 (Real modulus criterion NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$ *surface for Definition 3415.1 consists exactly of carrier exposure for R, explicit modulus exposure through M, sequence-source exposure through S, regular rational readback through Q, finite window exposure through W, dyadic tolerance exposure through D, sequence-limit handoff through L, real convergence-readiness through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $R = (M, S, Q, W, D, L, E, H, C, P, N)$. The analytic route is finite: a tolerance request reads the modulus row M, the source row S, the regular rational row Q, the window row W, and the dyadic row D before it reaches L and E.

The rows H, C, P, N preserve and name that same route. Componentwise hsame transport keeps each analytic coordinate in its displayed sibling surface, while Cont replay records the order in which the consumer reads the modulus, windows, regular name, sequence-limit row, and real seal.

No excluded object is a row of the carrier. Therefore the listed rows exhaust the seed NameCert obligations for the explicit real-modulus criterion. □

Theorem 2462.3 (Real modulus criterion completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealModulusCriterion}^{\uparrow}$ *packet must pass through*

$$\mathsf{CauchyConvergenceModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RealSequenceLimit}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in the displayed carrier order before treating the packet as convergence-ready for real-name use. The handoff supplies a finite criterion for explicit Cauchy-modulus convergence; it does not supply quotient completion, a selected limiting representative, countable choice, trichotomy, or an ambient completeness theorem.

Proof. Start with the carrier definition. The modulus row M and dyadic ledger D identify the tolerance request, while W and Q expose the finite tail windows and regular rational readback needed by the request.

Next read the sequence-limit and real-seal rows. The row L is reached only after the modulus, window, dyadic, and regular readback rows have been displayed, and E is the resulting $\mathsf{Real}^{\uparrow}$ readiness boundary.

Finally apply the structural rows. Transport, replay, provenance, and local naming can only preserve the displayed route, so a completion-facing consumer cannot bypass the criterion through quotient completion, representative selection, countable choice, trichotomy, host convergence, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusCriterion}^{\uparrow}$ packet is generated from an explicit CauchyConvergenceModulus row, a sequence source, RegSeqRat readback, finite StreamName windows, DyadicRatCore tolerance rows, a RealSequenceLimit consumer row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3415.1, Definition 3415.2, and Definition 3415.3 declare the finite explicit-modulus criterion connecting $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, trichotomy, host sequence convergence, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, explicit modulus exposure, sequence-limit handoff, real-seal nonescape, and completion-facing criterion readiness.

Chapter 2463

A Concrete Naming Certificate for $\mathsf{CauchyFilterCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyFilterCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed filter base becomes acceptable Cauchy-filter data for the L10 completion route. It sits after the finite $\mathsf{FilterBase}^{\uparrow}$ interface of chapter 1601, the Cauchy-filter surface of chapter 333, and the completion-facing routes of chapter 1506 and chapter 1473; it never replaces them by quotient filters, choice-selected limits, or host completeness.

Definition 2463.1 (Cauchy filter criterion carrier)**.** *A* $\mathsf{CauchyFilterCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, M, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row naming the finite base packets and directed-refinement witnesses that may be inspected by a completion consumer.*
2. *K is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row recording that the displayed base is Cauchy only through finite entourage or ball requests.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row reached only after B, K, W, R, D, E have been displayed.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row exposing the same Cauchy-filter route to entourage-facing completion consumers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used when the criterion is read sequentially.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the sequential windows and tolerance requests.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds every displayed Cauchy request.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal read by real-name consumers after the filter, window, regular-rational, and dyadic rows have been displayed.*
9. *H records componentwise* hsame *transport for the filter-base, Cauchy-filter, completion, sequential, tolerance, real-seal, and structural rows.*
10. *C records displayed* Cont *replay from a filter-base request through Cauchy-filter evidence, sequential readback, tolerance accounting, completion handoff, and real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCriterion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, M, U, W, R, D, E, H, C, P, N$. There is no coordinate for quotient filter equality, selected representatives, countable choice, host-side limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterCriterionUp.CauchyFilterCriterionTasteGate single carrier alignment`

Theorem 2463.2 (Cauchy filter criterion NameCert obligations)**.** *For the carrier of Definition 3416.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$ *surface consists exactly of carrier exposure for F, finite filter-base exposure through B, Cauchy-filter exposure through K, metric-completion handoff through M, uniform-completion handoff through U, sequential window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, real-name sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $F = (B, K, M, U, W, R, D, E, H, C, P, N)$. The base row B supplies only the finite directed interface of chapter 1601, and the Cauchy row K reads that interface before any completion-facing coordinate is visible.

The sequential rows then bind the criterion to the real-name route. The window row W, regular readback row R, and dyadic tolerance row D are displayed before the real seal E or either completion handoff can be consumed.

Finally the structural rows H, C, P, N transport, replay, source, and name exactly these coordinates. Since quotient filters, selected representatives, countable choice, host limits, and ambient completeness are not rows of the carrier, no further NameCert obligation remains. □

Theorem 2463.3 (Cauchy filter criterion real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCriterion}^{\uparrow}$ *carrier must pass through*

$$\mathsf{FilterBase}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in the displayed carrier order before reading the completion handoff rows M or U. The handoff supplies a finite Cauchy-filter criterion for the completion route; it does not supply quotient filters, choice-selected limits, selected representatives, countable choice, host completeness, or an ambient real-limit operator.

Proof. Start with the finite base route. The consumer first reads B and can narrow requests only through the displayed directed-refinement surface supplied by Definition 1601.3.

Next read the Cauchy and sequential rows. The row K witnesses the Cauchy-filter condition over the same base, while W, R, D translate inspected requests into finite windows, regular rational readback, and dyadic tolerance accounting.

The terminal seal and completion handoffs are downstream of those rows. Thus E, M, U can only expose the real-name and completion-facing boundaries after the displayed filter-base criterion has been replayed by C and transported by H.

No excluded principle is carried by P or N. The packet therefore feeds the L10 real and completion route without quotient filters, selected limits, countable choice, host completeness, or a hidden ambient limit object. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCriterion}^{\uparrow}$ packet is generated from a FilterBaseUp row, a CauchyFilterUp row, MetricCompletionUp and UniformCompletionUp handoff rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3416.1, Definition 3416.2, and Definition 3416.3 seed the finite criterion by which a displayed FilterBase and CauchyFilter route may feed Real, MetricCompletion, and UniformCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyFilterCriterionUp.CauchyFilterCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient filters, choice-selected limits, selected representatives, countable choice, host completeness, bridge checking, or an ambient real-limit operator.

Upgrade path: Obligation closure requires Cauchy-filter criterion exposure, completion handoff nonescape, real-seal readiness, and ledger-refusal proofs for BEDC.Derived.CauchyFilterCriterionUp.

Chapter 2464

A Concrete Naming Certificate for $\mathsf{UniformConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformConvergence}^{\uparrow}$ packet records uniform tail control before any $\mathsf{UniformLimit}^{\uparrow}$ consumer reads a $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the pointwise limiting surface of chapter 2314, the uniform-limit route of chapter 463, the continuous-map graph rows of chapter 130, and the terminal real-name seal of chapter 38. The chapter names the generic finite tail ledger used by later completion and limit consumers; it is not a selected limit, a supremum norm construction, countable choice, quotient equality of functions, host completeness, or open-cover compactness.

Definition 2464.1 (Uniform convergence carrier)**.** *A* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed function-family or sequence source. Each source entry is read only through the finite windows named by the packet.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window ledger. It records the parameter windows, index windows, and tolerance-request windows inspected by the consumer.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed finite comparisons attached to F and W.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has displayed the scheduled comparisons.*
5. *T is the uniform tail schedule. For each requested tolerance row it records one threshold row shared by every displayed observation window in W.*
6. *H records componentwise* hsame *transport for the source, window ledger, readback, real seal, tail schedule, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from any limit-facing consumer request through F, W, T, Q, R in that order.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier has no coordinate for a selected limiting function, a host supremum norm, quotient equality of functions, countable choice, ambient completeness, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.UniformConvergenceUp.TasteGate.UniformConvergenceTasteGate single carrier alignment`

Theorem 2464.2 (Uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier*

$$U = (F, W, Q, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source row, finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seals, uniform tail schedule, transports, continuations, provenance, and local naming;*
2. *uniform tail coverage, requiring every requested tolerance row to read the single threshold recorded in T before any limiting read is exposed;*
3. *real-seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *seal in R to remain downstream of the readback row Q and the schedule row T;*
4. $\mathsf{UniformLimit}^{\uparrow}$ *handoff, requiring each endpoint-facing consumer to factor through $F, W, Q, R, T, H, C, P, N$ before forming a limiting read;*
5. *transport and replay stability, requiring H and C to preserve the displayed source, window, readback, seal, and tail rows componentwise;*
6. *excluded rows, ruling out selected limits, host supremum norms, quotient equality of functions, countable choice, ambient completeness, open-cover compactness, and any hidden pointwise-only threshold.*

No uniform-convergence obligation is stored outside these rows.

Proof. Project the accepted packet $U = (F, W, Q, R, T, H, C, P, N)$. The carrier exposes exactly the displayed source, finite observation windows, regular rational readbacks, real seals, uniform tail schedule, and structural rows.

Uniformity is supplied by T. A tolerance request first selects a single threshold row in that ledger, and the same row must serve every displayed observation window in W before any readback is admitted.

The real seal is then downstream of the regular rational route. The row Q is the only readback channel attached to the scheduled comparisons, and the seal row R becomes visible only after Q has displayed those comparisons through the tail schedule.

The endpoint handoff follows by the displayed route order. A $\mathsf{UniformLimit}^{\uparrow}$ consumer receives F, W, T, Q, R and then the structural rows H, C, P, N; there is no carrier coordinate from which it can read a selected limit, host supremum norm, quotient equality, choice row, ambient completeness theorem, open-cover compactness argument, or pointwise-only threshold. □

Theorem 2464.3 (Uniform convergence UniformLimit handoff)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier reads*

$$F, W, Q, R, T, H, C, P, N$$

before forming the limiting read. Thus the limiting route is forced through the displayed source, finite observation windows, regular rational readbacks, real seals, uniform tail schedule, componentwise transport, continuation replay, provenance, and local naming. It receives no selected limit, host supremum norm, quotient function equality, countable-choice schedule, ambient completeness theorem, open-cover compactness principle, or hidden pointwise-only threshold.

Proof. Start with a limit-facing consumer request. By Definition 3417.1, the request first reads the source row F through the finite observation windows W.

The uniform tail schedule T then supplies the common threshold for the requested tolerance. The readback row Q is attached to that scheduled comparison, and the real seal row R is reachable only after the readback has displayed it.

Finally H, C, P, N transport, replay, source, and name the same route. Since every coordinate available to the consumer occurs in the displayed packet, the limiting read cannot be formed from a selected limit, host supremum norm, quotient function equality, countable-choice schedule, ambient completeness theorem, open-cover compactness principle, or hidden pointwise-only threshold. □

Falsifiable prediction: An accepted $\mathsf{UniformConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{UniformLimit}^{\uparrow}$ endpoint or $\mathsf{Real}^{\uparrow}$ seal without first reading $F, W, Q, R, T, H, C, P, N$ refutes the carrier surface.

Independence witness: streamname, regseqrat, pointwise-limit, uniformlimit, continuousmap, real: none is bijective to $\mathsf{UniformConvergence}^{\uparrow}$ because this packet jointly carries a displayed source, finite observation windows, regular readbacks, real seals, one uniform tail schedule, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3417.1, Definition 3417.2, and Definition 3417.3 seed the finite uniform tail ledger over displayed sources, StreamName windows, RegSeqRat readbacks, Real seals, and UniformLimit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformConvergenceUp.TasteGate.UniformConvergenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim selected limits, host supremum norms, quotient equality of functions, countable choice, ambient completeness, open-cover compactness, or bridge checking.

Upgrade path: Obligation closure requires uniform tail coverage, real-seal non-escape, UniformLimit handoff, transport and replay stability, and excluded-row coverage for BEDC.Derived.UniformConvergenceUp.

Constructive story: A $\mathsf{UniformConvergence}^{\uparrow}$ packet is generated from a displayed function-family or sequence source, finite StreamName observation windows, RegSeqRat readbacks, RealUp seals, one uniform tail schedule, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Chapter 2465

A Concrete Naming Certificate for $\mathsf{BishopApproximationSequence}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{BishopApproximationSequence}^{\uparrow}$ packet records a finite naming surface for approximation sequences whose visible observations are dyadic schedules, finite stream windows, and regular-rational readbacks. It sits between the dyadic arithmetic of chapter 291, the finite stream surface of chapter 126, the regular-rational route of chapter 301, the terminal real-name seal of chapter 38, and the convergence-modulus discipline of chapter 1438. The packet gives Real and Completion consumers a named approximation-sequence interface without adding quotient sequence equality, countable choice, trichotomy, a selected host limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2465.1 (BishopApproximationSequence carrier)**.** *A* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, W, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bracket row, storing the dyadic precision requests and the finite arithmetic witnesses used by the approximation schedule;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, storing the finite sequence windows whose entries are inspected by the consumer;*
3. *W is the window-synchronization row that binds the dyadic tolerances in D to the stream observations in S;*
4. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row, storing the finite modulus witnesses used to compare later windows with earlier requested tolerances;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the accepted windows as regular rational readback data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, W, M, R have been displayed;*
7. *H records the componentwise* hsame *transports for the dyadic row, stream row, window row, modulus row, regular handoff, real seal, continuation routes, provenance, and local naming row;*
8. *C records the displayed* Cont *routes that replay tolerance reads, stream-window reads, modulus reads, regular-rational handoff, and real-seal readback;*

9. *P is the* Pkg *provenance over* $\mathsf{BishopApproximationSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$ *row.*

The accepted carrier stores no quotient sequence equality, no selected infinite subsequence, no host real equality test, no countable-choice functional, no trichotomy row, and no ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.BishopApproximationSequenceUp.BishopApproximationSequenceTasteGate single carrier alignment`

Theorem 2465.2 (BishopApproximationSequence NameCert obligations)**.** *For an accepted packet $A = (D, S, W, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$ *obligations are exactly the following finite rows:*

1. *carrier admission: the accepted object is the displayed packet with dyadic tolerances, finite stream windows, synchronized window data, convergence modulus, regular-rational handoff, real seal, transports, continuations, provenance, and local naming;*

2. *dyadic-window coherence: every precision request in D is attached to a displayed window in S through the synchronization row W;*

3. *modulus coverage: every accepted tolerance request reads a finite witness in M before any regular-rational or real-name observation is exported;*

4. *regular handoff stability: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R reads only the synchronized windows and modulus witnesses carried by D, S, W, M;*

5. *classifier transport: the componentwise transports in H preserve the dyadic row, stream row, synchronization row, modulus row, regular handoff, real seal, continuation replay, provenance, and local naming row;*

6. *ledger exhaustion: every Real or Completion consumer must pass through one of $D, S, W, M, R, E, H, C, P, N$ before reading any approximation-sequence observation.*

No obligation may call on quotient sequence equality, a selected host limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier row by row. Carrier admission is immediate from the displayed BHist packet, and dyadic-window coherence reads only the tolerance row D, stream row S, and synchronization row W.

For modulus coverage, inspect a tolerance request in D. The row M supplies the finite convergence witness attached to that request, and the route into R is available only after the synchronized window has been displayed.

Classifier transport and ledger exhaustion are structural. The transport row H moves only the displayed coordinates, while C replays only the displayed tolerance, window, modulus, regular handoff, and seal routes. Since the carrier contains no quotient equality, selected limit, choice functional, trichotomy row, host equality test, or completeness coordinate, none of those principles can enter the local obligations. □

Theorem 2465.3 (BishopApproximationSequence RegSeqRat handoff)**.** *Let $A = (D, S, W, M, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *packet. Every regular-rational read exported by A factors through a displayed dyadic tolerance in D, a finite stream window in S, the synchronization*

row W, and a convergence-modulus witness in M before the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R is available. The route is stable under the transports in H and the replay routes in C.*

Proof. Begin with a regular-rational consumer of the packet. The carrier makes R the only regular-rational handoff coordinate, and R is downstream of the dyadic, stream-window, synchronization, and modulus rows.

Apply Definition 3418.2. Its dyadic-window coherence and modulus coverage rows force every tolerance request to pass through D, S, W, M before a regular read is exposed.

The transports in H preserve each displayed coordinate, and the continuations in C replay only the same route. Thus the regular-rational read cannot detach from the finite approximation schedule or use quotient sequence equality, countable choice, trichotomy, host equality, selected limits, or ambient completeness. □

Theorem 2465.4 (BishopApproximationSequence Real seal nonescape)**.** *Let $A = (D, S, W, M, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *packet. No* $\mathsf{Real}^{\uparrow}$*-facing consumer can bypass the dyadic tolerance row D, finite stream row S, synchronization row W, convergence-modulus row M, regular handoff R, terminal seal E, transports H, replays C, provenance P, and local name N to obtain quotient sequence equality, a selected host limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier. The only terminal real-name coordinate is E, and it is displayed after the dyadic tolerance row, finite stream row, synchronization row, modulus row, and regular-rational handoff.

Use Definition 3418.3. Every regular-rational approximation read has already passed through D, S, W, M before reaching R, so the route into E is a finite route through the named rows.

The rows H and C can transport and replay only displayed coordinates, while P and N record the same package and local name. Because the carrier contains none of the excluded objects, the Real-facing seal cannot export quotient equality, selected limits, choice, trichotomy, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopApproximationSequence}^{\uparrow}$ packet is generated from dyadic tolerances, finite stream windows, a synchronization row, a convergence-modulus row, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3418.1, Definition 3418.2, Definition 3418.3, and Definition 3418.4 present the finite approximation-sequence certificate surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient sequence equality, selected host limits, countable choice, trichotomy, host real equality, arbitrary least-upper-bound completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, dyadic-window coherence, modulus coverage, regular handoff stability, classifier transport, ledger exhaustion, and real-seal nonescape for the displayed packet.

Chapter 2466

A Concrete Naming Certificate for $\mathsf{LaxMilgram}^{\uparrow}$

Following Lax and Milgram and the variational treatment in Evans, the $\mathsf{LaxMilgram}^{\uparrow}$ packet records the finite BEDC surface for coercive variational solvability. It is read as a neighbour of the Hilbert and Sobolev certificates of chapter 91 and chapter 440, with its pairing row anchored in the bilinear-form chapter chapter 146 and its bounded action surface anchored in chapter 1644. The packet names only the displayed variational handoff; it does not add a choice-selected solution operator or an ambient PDE existence theorem.

Definition 2466.1 (Lax-Milgram carrier)**.** *A* $\mathsf{LaxMilgram}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, B, F, C, S, T, P, N)$$

with the following displayed rows:

1. *H is the Hilbert/Sobolev domain row, read through the* $\mathsf{Hilbert}^{\uparrow}$ *surface and, when a weak-form PDE consumer is present, through the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative ledger;*
2. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *row naming the bilinear pairing together with its boundedness and coercivity evidence as finite classifier rows;*
3. *F is the bounded functional row, read through the same bounded-linear-operator surface used by* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *consumers;*
4. *C records the coercivity and boundedness constants as displayed ledger entries rather than as real-number selectors outside the packet;*
5. *S is the solution handoff row: it exposes the single variational response row certified by the equation $B(S, v) = F(v)$ for displayed test rows v in H;*
6. *T stores the* hsame *transport and* Cont *replay rows for the domain, bilinear form, bounded functional, constants, and solution handoff;*
7. *P is the* Pkg *provenance over* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a choice-selected global inverse, a quotient representative, a completed PDE solution space beyond the cited Sobolev row, or a host theorem asserting existence outside the displayed variational packet.

Lean4 ✓ `BEDC.Derived.LaxMilgramUp.LaxMilgramTasteGate single carrier alignment`

Theorem 2466.2 (Lax-Milgram NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$ *surface for the carrier of Definition 3419.1 consists of exactly five finite obligations:*

1. *domain admission, requiring H to be read through the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row and any weak-form consumer through the* $\mathsf{Sobolev}^{\uparrow}$ *ledger;*
2. *bilinear-form admission, requiring B to expose only the* $\mathsf{BilinForm}^{\uparrow}$ *pairing row together with boundedness and coercivity ledger entries;*
3. *functional admission, requiring F to be read through the bounded functional surface supplied by* $\mathsf{BoundedLinearOperator}^{\uparrow}$*;*
4. *variational handoff exactness, requiring every solution-facing read to pass through S and the displayed equation row $B(S, v) = F(v)$ for admitted test rows v;*
5. *transport and replay stability, requiring T, P, N to preserve the same domain, bilinear, functional, constant, and handoff rows under componentwise* hsame*, displayed* Cont*, provenance, and local naming.*

Proof. Project the carrier rows. The domain, bilinear form, functional, constant ledger, handoff, transport, provenance, and local naming evidence are the full visible inventory of K.

Use the cited sibling certificates for each analytic dependency. The Hilbert and Sobolev rows supply the domain-facing reads, the bilinear-form row supplies the pairing and its boundedness/coercivity classifier, and the bounded-linear-operator surface supplies the functional row.

Finally inspect a solution-facing consumer. It can reach only the handoff row S and the displayed variational equation against admitted test rows. The transport, replay, provenance, and naming rows preserve that finite route and add no external selector. □

Theorem 2466.3 (Lax-Milgram variational handoff)**.** *Every PDE, Sobolev, Hilbert, or variational consumer of an accepted* $\mathsf{LaxMilgram}^{\uparrow}$ *packet receives only the displayed route*

$$H \longmapsto B \longmapsto F \longmapsto C \longmapsto S \longmapsto T, P, N.$$

The consumer may use the certified variational response row and its equation against admitted test rows, but it receives no choice-selected solution operator, no quotient representative, no host inverse map, no ambient compactness principle, and no PDE solution space beyond the finite Sobolev or Hilbert rows already cited.

Proof. Begin with the NameCert obligations. They force any consumer to read the domain, bilinear form, functional, constant ledger, and handoff rows in the order displayed by the carrier.

The Sobolev and Hilbert dependencies identify the domain surface, while the bilinear-form and bounded-linear-operator dependencies identify the pairing and functional surfaces. These reads are certificate rows, not host analytic constructions.

It remains to inspect the handoff. The row S is available only as the variational response named by $B(S, v) = F(v)$ for displayed tests v. Since no other solution coordinate is present in the carrier, the listed selector, quotient, inverse, compactness, and ambient PDE objects cannot be exported. □

Theorem 2466.4 (Lax-Milgram energy-minimizer handoff)**.** *An accepted* $\mathsf{LaxMilgram}^{\uparrow}$ *packet exposes its variational solution row as an energy-minimizer handoff only through the Hilbert/Sobolev*

domain row of chapter 91 and chapter 440, the $\mathsf{BilinForm}^{\uparrow}$ *coercive pairing row of chapter 146, and the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *functional row of chapter 1644. The handoff is the finite route*

$$H \longmapsto B \longmapsto F \longmapsto C \longmapsto S,$$

with T, P, N *preserving transport, replay, provenance, and local naming. It exports no selected inverse operator, no quotient representative of a solution class, and no ambient PDE existence theorem.*

Proof. Project the carrier of Definition 3419.1. The only domain coordinate is H, so an energy read starts from the Hilbert row and, for weak-form consumers, the Sobolev ledger named by the sibling chapters.

The energy pairing is then forced through B and C. The bilinear-form sibling supplies the coercive pairing row, while the constant ledger records the displayed boundedness and coercivity data. No inverse operator is stored in these rows.

The forcing term is the row F supplied through the bounded-linear-operator surface. The variational response row S can therefore be read as an energy-minimizer handoff only after the domain, pairing, constant, and functional rows have been displayed.

Finally T, P, N transport, source, and name the same finite route. Since the carrier has no coordinate for a selected inverse operator, quotient representative, or ambient PDE existence theorem, none of those objects can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$.

Constructive story: A $\mathsf{LaxMilgram}^{\uparrow}$ packet is generated from a Hilbert or Sobolev domain row, a bilinear-form row with boundedness and coercivity ledgers, a bounded functional row, a solution handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$, seed)

Scope closed: Definition 3419.1, Definition 3419.2, and Definition 3419.3 seed the finite variational certificate surface consumed by Sobolev, Hilbert, PDE, and bounded-operator routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a choice-selected solution operator, quotient representative, host inverse map, ambient compactness principle, or full PDE existence theorem.

Upgrade path: Obligation closure requires checked carrier admission, bilinear boundedness and coercivity rows, bounded functional admission, solution handoff exactness, transport, replay, provenance, and local naming for the displayed packet.

Chapter 2467

A Concrete Naming Certificate for $\mathsf{BanachAlaoglu}^{\uparrow}$

Following Banach and Alaoglu, the $\mathsf{BanachAlaoglu}^{\uparrow}$ packet records the finite weak-star dual-ball window that functional-analysis consumers can inspect before any unrestricted compactness theorem is available. It sits over the Banach source row of chapter 90, the bounded-linear-functional surface supplied by chapter 1644, the scalar norm readback of chapter 1500, the terminal real seal of chapter 38, and the finite product compactness handoff of **??**. The chapter names only finite windows in the weak-star dual ball; it does not claim unrestricted Tychonoff compactness, ultrafilter arguments, host weak-star compactness, quotient function spaces, or selected cluster points.

The weak-star window also reads the locally convex seminorm-neighborhood discipline of chapter 3386 before any dual-ball compactness route is exposed.

Definition 2467.1 (Banach-Alaoglu finite weak-star dual-ball carrier)**.** *A* $\mathsf{BanachAlaoglu}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, L, W, S, R, F, T, C, P, N)$$

with the following displayed rows:

1. *B is the Banach source row, read through the displayed* $\mathsf{Banach}^{\uparrow}$ *carrier and its norm, metric, regular-Cauchy, and completion ledgers.*
2. *L is the bounded-linear functional family row, whose members are read as* $\mathsf{BoundedLinearOperator}^{\uparrow}(B, \mathsf{Real}^{\uparrow})$ *coordinates with displayed operator bounds.*
3. *W is the finite weak-star seminorm-window row, listing the finitely many source probes and scalar tolerances allowed in the present observation.*
4. *S is the dual-ball bound row, recording the shared scalar bound on the displayed functionals through* $\mathsf{NormedRealLine}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readbacks.*
5. *R is the Real and normed-real readback row for all scalar values, scalar bounds, and seminorm tolerances exposed by the window.*
6. *F is the finite-product handoff row, sending the displayed scalar coordinate window to the finite* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$ *compactness surface.*
7. *T records componentwise* hsame *transport for the Banach source, functional family, weak-star window, scalar readbacks, finite-product handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a weak-star window request through $B, L, W, S, R, F, T, P, N$.*
9. *P is the* Pkg *provenance over* BanachAlaoglu$^{\uparrow}$, Banach$^{\uparrow}$, BoundedLinearOperator$^{\uparrow}$, NormedRealLine$^{\uparrow}$, Real$^{\uparrow}$, TychonoffFiniteProduct$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BanachAlaoglu}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted Tychonoff compactness, ultrafilter or choice arguments, host weak-star compactness, quotient function spaces, selected cluster points, ambient dual-space compactness, Lean verification, or standard bridge data.

Lean4 ✓ `BEDC.Derived.BanachAlaogluUp.BanachAlaogluTasteGate single carrier alignment`

Theorem 2467.2 (Banach-Alaoglu NameCert obligations)**.** *For the carrier of Definition 3420.1, the local* NameCert$_{\text{BanachAlaoglu}^{\uparrow}}$ *surface consists of exactly six finite obligations:*

1. *Banach source admission, requiring B to be read through the displayed source row of chapter 90;*
2. *bounded-linear-functional admission, requiring each coordinate of L to pass through the bounded-linear-operator surface of chapter 1644;*
3. *finite weak-star window admission, requiring W to expose only finitely many source probes and scalar tolerances;*
4. *scalar readback admission, requiring S and R to read all bounds and scalar values through chapter 1500 and chapter 38;*
5. *finite-product handoff exactness, requiring every compactness-facing read to pass through F and the finite product surface of* **??***;*
6. *transport, replay, provenance, and local naming stability, requiring T, C, P, N to preserve the same finite route under componentwise* hsame*, displayed* Cont*, package provenance, and local naming.*

Proof. Project the carrier rows. The displayed Banach source, bounded-linear functional family, finite weak-star window, scalar bound and readback rows, finite-product handoff, transport, replay, provenance, and local naming rows exhaust the visible inventory of A.

The source and functional obligations are read from the cited sibling certificates. The Banach row supplies the normed source data, and the bounded-linear-operator row supplies the finite bounded-functional surface into the scalar real line.

The weak-star content is only the finite row W. Its probes and tolerances are read through S and R, so every scalar comparison remains inside the displayed normed-real and Real readbacks.

Finally inspect a compactness-facing consumer. It may reach only the finite-product row F after the displayed scalar coordinates have been exposed. Since no unrestricted product, ultrafilter, quotient function space, or selected cluster-point row is present, the certificate closes only the finite weak-star window. □

Theorem 2467.3 (Banach-Alaoglu finite dual-ball handoff)**.** *Every accepted* BanachAlaoglu$^{\uparrow}$ *packet exports the route*

$$B, L, W, S, R \longmapsto F \longmapsto T, C, P, N$$

to any duality, compactness, or functional-analysis consumer. The consumer receives a finite weak-star dual-ball window and the corresponding finite product compactness handoff, but receives no unrestricted Tychonoff compactness theorem, no ultrafilter or choice argument, no host weak-star compactness claim, no quotient function space, and no selected cluster point.

Proof. Begin with an accepted packet and a consumer request. The request first reads the Banach source row B and the bounded-linear functional row L, so the functional data is already confined to displayed source and scalar surfaces.

The finite weak-star window W then selects the only probes and scalar tolerances that may be inspected. Rows S and R translate those displayed values and bounds through the normed-real and Real readback interfaces.

Only after these rows are present can the consumer reach F. That row is the finite-product handoff, not an unrestricted product theorem or a compactness assertion over an entire host dual ball.

The structural rows T, C, P, N preserve and replay the same finite route. Because the carrier contains no selector, quotient function space, ultrafilter, or unrestricted compactness coordinate, the exported handoff is exactly the finite weak-star dual-ball window named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachAlaoglu}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachAlaoglu}^{\uparrow}$ packet is generated from a Banach source row, a bounded-linear functional family row, finite weak-star seminorm windows, Real and normed-real scalar readbacks, a finite-product handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachAlaoglu}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3420.1, Definition 3420.2, and Definition 3420.3 seed the finite weak-star dual-ball window consumed by later compactness and duality routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachAlaoglu}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted Tychonoff compactness, ultrafilter or choice arguments, host weak-star compactness, quotient function spaces, selected cluster points, ambient dual-space compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-linear functional family admission, finite weak-star window exactness, scalar readback admission, finite-product handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.BanachAlaogluUp.

Chapter 2468

A Concrete Naming Certificate for $\mathsf{MonotoneCauchySequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{MonotoneCauchySequence}^{\uparrow}$ packet records the finite carrier surface for a monotone regular rational source whose tail windows are already prepared for constructive real completion. It sits between the $\mathsf{Real}^{\uparrow}$ rows of chapter 38, the regular sequence discipline of $\mathsf{RegSeqRat}^{\uparrow}$, the $\mathsf{StreamName}^{\uparrow}$ window ledgers, the dyadic comparison rows of $\mathsf{DyadicRatCore}^{\uparrow}$, and the limit-facing boundary of chapter 1413. The packet names the Cauchy tail surface available before a completion consumer asks for a completed real.

Definition 2468.1 (MonotoneCauchySequence carrier)**.** *A* $\mathsf{MonotoneCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, R, D, O, T, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for the inspected sequence windows.*
2. *W is the stream-window ledger recording the finite index intervals and tail requests visible to the packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that ties each scheduled window to a regular rational approximation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row used to compare rational approximants inside common dyadic windows.*
5. *O is the monotonicity ledger, listing the finite comparison rows by which later inspected approximants do not move below earlier inspected approximants.*
6. *T is the tail-threshold row, assigning each requested precision a displayed tail window and dyadic comparison bound.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row that exposes the regular Cauchy source to real-name consumers without choosing a host limit.*
8. *H records componentwise* hsame *transport for the stream schedule, window ledger, regular readback, dyadic comparisons, monotonicity ledger, tail threshold, real seal, replay route, provenance, and local naming row.*
9. *C records displayed* Cont *replay from a convergence request through $S, W, R, D, O, T, A, H, P, N$.*

10. *P is the* Pkg *provenance over* MonotoneCauchySequence$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealSequenceLimit$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MonotoneCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through

$$S, W, R, D, O, T, A, H, C, P, N.$$

It has no coordinate for quotient streams, selected host limits, host real equality, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchySequenceUp.MonotoneCauchySequenceTasteGate single carrier alignment`

Theorem 2468.2 (MonotoneCauchySequence NameCert obligations)**.** *For every accepted* MonotoneCauchySequenc *carrier*

$$M = (S, W, R, D, O, T, A, H, C, P, N),$$

the local NameCert$_{\text{MonotoneCauchySequence}^{\uparrow}}$ *surface consists exactly of the finite rows for carrier admission, stream-window scheduling, regular rational readback, dyadic comparison, monotonicity-ledger admission, tail-threshold admission,* Real$^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each obligation reads the same* StreamName$^{\uparrow}$ *windows and the same* RegSeqRat$^{\uparrow}$ *source rows before the tail threshold or real seal is exposed. No obligation may appeal to quotient streams, selected host limits, host real equality, countable choice, trichotomy, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 3421.1. Carrier admission exposes precisely the rows $S, W, R, D, O, T, A, H, C, P, N$, so every inspected approximation is read as a scheduled regular rational row through the displayed stream windows.

The order-facing and Cauchy-facing rows use the same source schedule. The monotonicity ledger O compares only approximants exposed by S, W, R, D, while the tail-threshold row T assigns each requested precision a finite window and dyadic comparison bound over the same readback surface.

The real-facing seal comes after those rows. The Real$^{\uparrow}$ seal A may be exposed only from the regular rational source, dyadic comparison row, monotonicity ledger, and tail threshold already present in the carrier, and a RealSequenceLimit$^{\uparrow}$ consumer can read only that displayed seal.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, host-limit, host-equality, choice, trichotomy, and ambient-completeness coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Theorem 2468.3 (MonotoneCauchySequence tail-window boundary)**.** *Let*

$$M = (S, W, R, D, O, T, A, H, C, P, N)$$

be an accepted MonotoneCauchySequence$^{\uparrow}$ *carrier. Every* RealSequenceLimit$^{\uparrow}$ *or completion-facing consumer of M must first read the* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *comparisons, monotonicity ledger, tail-threshold row,* Real$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *through the same finite surface. The consumer receives a monotone Cauchy tail-window route, not a quotient stream, selected host limit, host real equality, countable-choice tail, trichotomy decision, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Proof. Start with a limit-facing or completion-facing consumer. By Definition 3421.1, the only available sequence data are the scheduled $\mathsf{StreamName}^{\uparrow}$ windows S, W, the regular rational readback row R, and the dyadic comparison row D.

Apply Definition 3421.2. The monotonicity ledger O and tail-threshold row T are attached to that same scheduled source surface, and the $\mathsf{Real}^{\uparrow}$ seal A is available only after both the dyadic comparison and tail threshold have been displayed.

The $\mathsf{RealSequenceLimit}^{\uparrow}$ route may therefore inspect a finite monotone Cauchy boundary before reaching any real-name consumer. A completion-facing route receives the same boundary through the real seal, not a completed quotient stream or selected host limit.

The structural rows H, C, P, N preserve this boundary under transport, replay, provenance, and naming. The absent quotient-stream, host-limit, host-equality, choice, trichotomy, and ambient-completeness coordinates remain absent for every downstream consumer. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneCauchySequence}^{\uparrow}$ consumer that claims access to a completion or limit boundary must display the StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, monotonicity ledger, tail threshold, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert before any RealSequenceLimit or completion read is inspected.

Independence witness: real, regseqrat, streamname, dyadicratcore, realsequencelimit: MonotoneCauchySequence is not bijective to any listed sibling because it jointly carries monotonicity, regular rational readback, dyadic tail thresholds, and a real seal before any limit-facing consumer may read the source sequence.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3421.1, Definition 3421.2, and Definition 3421.3 seed the finite source carrier, local obligation surface, and tail-window boundary for monotone Cauchy sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, stream-window scheduling, regular readback, dyadic comparison, monotonicity-ledger admission, tail-threshold admission, real sealing, transport replay, provenance, and local naming for BEDC.Derived.MonotoneCauchySequenceUp.

Constructive story: A $\mathsf{MonotoneCauchySequence}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, monotonicity and tail-threshold ledgers, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2469

A Concrete Naming Certificate for MetaCICCandidateSetSNHandoff$^{\uparrow}$

The MetaCICCandidateSetSNHandoff$^{\uparrow}$ packet realizes the candidate-set strong-normalization handoff of chapter 9383 as finite NameCert data. It is narrower than the candidate-set normalization surface of chapter 2320: this packet records exactly the route from Candidate and CandidateSet membership evidence to a closed typed term strong-normalization export, with beta-closure and no-infinite-reduction kept visible. It sits beside the MetaCIC critical path of chapter 765, the closed substitution boundary of chapter 527, the finite trace evaluator of chapter 818, the kernel category surface of chapter 488, and the halting distinction packet of chapter 491.

Definition 2469.1 (MetaCIC candidate-set SN handoff carrier)**.** *A* MetaCICCandidateSetSNHandoff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, S, T, B, E, N, R, C, P, L)$$

with the following displayed rows:

1. *A is the Candidate row, naming the reducibility-candidate evidence admitted by the normalization audit map.*
2. *S is the CandidateSet membership row, recording finite membership, soundness, and the selected candidate-set witness consumed by the handoff.*
3. *T is the closed typed term evidence row, recording the empty-context term, its type, and the closedness evidence to which the candidate-set row is attached.*
4. *B is the beta-closure row, exposing the one-step beta stability supplied by the same CandidateSet route.*
5. *E is the strong-normalization export row, read only after A, S, T, B have all been displayed.*
6. *N is the no-infinite-reduction projection row obtained from the exported strong-normalization evidence.*
7. *R records componentwise* hsame *transport for candidate, candidate-set, closed-term, beta-closure, export, no-infinite, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a closed-term request through CandidateSet membership, beta closure, strong-normalization export, and no-infinite-reduction projection.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$*, Candidate, CandidateSet, closed typed term evidence, beta-closure evidence, strong-normalization export,* BHist, hsame, Cont, Pkg*, and* NameCert *rows.*

10. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, T, B, E, N, R, C, P, L$. It has no coordinate for candidate-free strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, or a proof-producing oracle outside the displayed handoff.

Theorem 2469.2 (MetaCIC candidate-set SN handoff obligations)**.** *For every accepted* MetaCICCandidateSetSNHa *carrier $H = (A, S, T, B, E, N, R, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission; Candidate evidence through A; CandidateSet membership, soundness, and witness exposure through S; closed typed term evidence through T; beta-closure stability through B; strong-normalization export through E; no-infinite-reduction projection through N; componentwise transport through R; displayed replay through C; provenance through P; and local naming through L.*

Proof. Project the accepted carrier of Definition 3422.1. The visible rows are exactly $A, S, T, B, E, N, R, C, P, L$.

The positive route begins with A and S. Candidate evidence and CandidateSet membership are displayed before the closed typed term row can be read as a normalization input.

The closed-term and beta rows are T and B. They keep the empty-context typing, closedness, and one-step beta stability attached to the same CandidateSet witness.

The export rows are E and N. Strong normalization is exported only after A, S, T, B are present, and no-infinite-reduction is only a projection from that export. The transport, replay, provenance, and naming rows preserve the same handoff, so no excluded host route can appear in the NameCert surface. □

Theorem 2469.3 (MetaCIC candidate-set SN export nonescape)**.** *Every public strong-normalization or no-infinite-reduction read exported by an accepted* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *packet factors through*

$$A, S \longmapsto T, B \longmapsto E, N \longmapsto R, C, P, L.$$

Thus a closed typed term consumer receives the route only as CandidateSet membership, closed-term, beta-closure, and SN export NameCert data, not as an unconditional theorem over all closed MetaCIC terms.

Proof. Read the source rows A and S. They name the candidate evidence and CandidateSet membership that the normalization audit map requires for the public closed-term route.

Next read T and B. The closed typed term row and beta-closure row attach the term and its local reduction stability to the same finite evidence route.

Finally read E, N, R, C, P, L. The strong-normalization export and no-infinite projection are downstream of the candidate-set source, while transport, replay, provenance, and local naming preserve that order. Since the carrier has no coordinate for candidate-free strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, or a proof-producing oracle, no consumer can escape the displayed route. □

Theorem 2469.4 (MetaCIC candidate-set SN export factorization)**.** *For every accepted* MetaCICCandidateSetSNH *carrier*

$$H = (A, S, T, B, E, N, R, C, P, L),$$

each strong-normalization export row factors through the displayed order

$$A \longmapsto S \longmapsto T \longmapsto B \longmapsto E \longmapsto N \longmapsto R \longmapsto C \longmapsto P \longmapsto L.$$

Thus the export is mediated first by Candidate evidence, then CandidateSet membership, closed typed term evidence, beta-closure stability, strong-normalization export, no-infinite projection, transport, replay, provenance, and local NameCert rows.

Proof. Project the accepted carrier of Definition 3422.1. The rows are displayed as $A, S, T, B, E, N, R, C, P, L$, and the local classifier compares accepted packets only through those coordinates.

The candidate-mediated source is A followed by S. The closed typed term row T and beta-closure stability row B can be read only as part of the same finite CandidateSet handoff.

The export itself is then E, with N only as the no-infinite-reduction projection from that export. The nonescape theorem Definition 3422.3 rules out a candidate-free or host-oracle route for these public reads.

Finally the structural rows R, C, P, L transport, replay, source, and name the same route. This gives the stated factorization and keeps the strong-normalization export inside the local NameCert surface. □

Theorem 2469.5 (MetaCIC candidate-set SN TasteGate route)**.** *Every strong-normalization export from an accepted* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *carrier*

$$H = (A, S, T, B, E, N, R, C, P, L)$$

passes through Candidate evidence A, CandidateSet membership S, closed typed term evidence T, beta-closure stability B, no-infinite-reduction projection N, componentwise transport R, replay C, provenance P, and the local NameCert row L. These rows are exactly the field projections that a future TasteGate field-faithful instance for $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *must preserve when it exposes the strong-normalization export E.*

```
Lean4 ✓  BEDC.Derived.MetaCICCandidateSetSNHandoffUp.MetaCICCandidateSetSNHandoffTasteGate
single carrier alignment
```

Proof. Project the accepted carrier using Definition 3422.1. The field inventory is $A, S, T, B, E, N, R, C, P, L$, and the classifier compares accepted packets only through those displayed coordinates.

Apply Definition 3422.4. It orders the SN export through Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure, export, no-infinite projection, transport, replay, provenance, and local naming.

The TasteGate field-faithful requirement therefore has no additional field to preserve and no license to merge fields. Preserving the handoff means preserving each of $A, S, T, B, E, N, R, C, P, L$ as a displayed projection.

Since the carrier has no coordinate for candidate-free strong normalization, host evaluator normalization, quotient conversion equality, or an external proof oracle, the same field-faithful route is the only route available to any SN export consumer. □

Falsifiable prediction: A closed typed term read that reaches $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ strong-normalization export without displaying CandidateSet membership, closed typed term evidence, beta-closure stability, and the SN export row fails the carrier surface.

Independence witness: metaciccriticalpath, closedtermsubstitutionboundary, finitetraceevaluator, kernelcategory, haltingdistinction: none is bijective because $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$

binds Candidate evidence, CandidateSet membership, closed typed term evidence, beta closure, SN export, no-infinite projection, transport, replay, provenance, and local NameCert rows in one finite handoff packet.

The Lean follow-up is `BEDC.Derived.MetaCICCandidateSetSNHandoffUp`. Its TasteGate surface must provide Nontrivial MetaCICCandidateSetSNHandoff$^{\uparrow}$ and FieldFaithful MetaCICCandidateSetSNHandoff$^{\uparrow}$ instances that keep $A, S, T, B, E, N, R, C, P, L$ as separate displayed fields.

Closure status for NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$.

Constructive story: A MetaCICCandidateSetSNHandoff$^{\uparrow}$ packet is generated from Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure evidence, strong-normalization export, no-infinite-reduction projection, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 3422.1, Definition 3422.2, Definition 3422.3, and Definition 3422.4 seed the finite candidate-set strong-normalization handoff surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unconditional MetaCIC strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure stability, SN export, no-infinite projection, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MetaCICCandidateSetSNHandoffUp.

Chapter 2470

A Concrete Naming Certificate for $\mathsf{FilterConvergence}^{\uparrow}$

Following Kelley and Bourbaki on filter convergence, the $\mathsf{FilterConvergence}^{\uparrow}$ packet records the finite convergence interface needed by the L10 completion route. It sits between the finite $\mathsf{CauchyFilter}^{\uparrow}$ surface of chapter 333, the regular directed-tail surface of chapter 1445, the window discipline of chapter 126, the ambient metric interface of chapter 57, and the terminal real-name seal of chapter 38. The packet names convergence toward a displayed target point through visible tail events and tolerance rows; it does not quotient filters, select host limits, invoke countable choice, hide cofinal tails, or assume ambient real completeness.

Definition 2470.1 (FilterConvergence carrier)**.** *A* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, X, T, V, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the visible* $\mathsf{CauchyFilter}^{\uparrow}$ *row. It stores the finite filterbase requests and refinement evidence from which convergence may be read.*
2. *X is the target point row over* $\mathsf{Met}^{\uparrow}$*. It records the displayed point classifier, distance observations, and metric-law ledger used by every neighborhood request.*
3. *T is the finite tail-event schedule. Each row names a filter event requested around the target point and the finite stage at which that event is inspected.*
4. *V is the Metric/Real tolerance row. It matches the metric ball or entourage demand around X with the rational and real-facing tolerance data read by the convergence request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after the filter row, target point row, tail schedule, and tolerance row have all been displayed.*
6. *H records componentwise* hsame *transport for B, X, T, V, E and for the structural rows below.*
7. *C records displayed* Cont *replay from filterbase refinement to tail events, from target-point neighborhoods to tolerance windows, and from tolerance windows to the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{FilterConvergence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, X, T, V, E, H, C, P, N$. There is no coordinate for quotient filters, selected host limits, hidden cofinal tails, countable-choice schedules, host equality of target points, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.FilterConvergenceUp.FilterConvergenceTasteGate single carrier alignment`

Theorem 2470.2 (FilterConvergence NameCert obligations)**.** *For every accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier $F = (B, X, T, V, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, visible filter exposure through B, target-point exposure through X, finite tail-event exposure through T, Metric/Real tolerance readback through V, terminal Real seal handoff through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the convergence route

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

without adding quotient filters, selected limits, hidden cofinal tails, countable choice, host target equality, or ambient real completeness.

Proof. Project the carrier of Definition 3423.1. The packet exposes exactly the visible filter row, target point row, tail-event schedule, tolerance row, Real seal, and structural rows.

The filter and target rows are read first. The row B supplies only finite filterbase requests, while X supplies the metric target point and its displayed neighborhood observations.

The tail schedule and tolerance row then bind convergence to finite data. The schedule T lists the filter events inspected around X, and V ties each such event to the Metric and Real tolerance windows consumed by the request.

Finally E, H, C, P, N seal, transport, replay, source, and name the same ordered route. Since the excluded objects are absent from the carrier, no further local NameCert obligation can be exported. □

Theorem 2470.3 (FilterConvergence tail stability)**.** *Let $F = (B, X, T, V, E, H, C, P, N)$ be an accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier. Any two displayed tail events in T that are read around the same target point row X admit a later displayed filterbase refinement inside B whose Metric/Real tolerance data is still the row V and whose terminal seal is still E.*

Consequently tail refinement is stable under componentwise hsame *transport, displayed* Cont *replay, package provenance, and local naming through H, C, P, N. The stability claim does not use quotient filters, selected host limits, countable choice, hidden cofinal tails, host target equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with two tail events in the finite schedule T. Because both events are carrier rows, they are already tied to the same visible filterbase row B and the same target point row X.

Use the directed refinement surface read from B. The later filterbase row is displayed inside the packet, so the proof never passes to an external cofinal family or quotient filter.

Read the tolerance row next. The row V is the only Metric/Real tolerance interface attached to the scheduled tail events, and H, C transport and replay that same interface to the later refinement.

The Real seal is therefore unchanged. Since E is downstream from B, X, T, V, and P, N only source and name those coordinates, the excluded choice, limit, hidden-tail, equality, and completeness principles cannot enter the tail-stability route. □

Theorem 2470.4 (FilterConvergence Real seal handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *packet must read the displayed convergence route*

$$B \longmapsto X \longmapsto T \longmapsto V \longmapsto E$$

before the Real seal is available. In BEDC terms, the consumer passes through the visible $\mathsf{CauchyFilter}^{\uparrow}$ *row, the target* $\mathsf{Met}^{\uparrow}$ *point row, the* $\mathsf{StreamName}^{\uparrow}$ *tail-event schedule, and the Metric/Real tolerance row before it reaches* $\mathsf{Real}^{\uparrow}$*.*

The handoff supplies a finite convergence interface for $\mathsf{RegularCauchyFilter}^{\uparrow}$ *and downstream completion-limit uniqueness arguments. It supplies no quotient filter, selected host limit, countable-choice schedule, hidden cofinal tail, host equality of target points, or ambient real-completeness principle.*

Proof. Start with the visible filter row B. A consumer can inspect only the finite filterbase requests and refinements recorded by the packet.

Next read the target and schedule rows. The row X supplies the metric target point and neighborhood observations, while T records the finite tail events that answer those neighborhood requests.

The tolerance and seal rows are downstream. The row V matches those tail events with the Metric and Real tolerance windows, and E seals the compatible readback only after that route has been replayed.

Thus every Real-facing consumer receives the same ordered route through $B, X, T, V, E, H, C, P, N$. Since no excluded principle is a carrier coordinate, the handoff is a finite convergence certificate rather than an ambient completeness assertion. □

Theorem 2470.5 (Filter convergence directed-net index)**.** *Any* $\mathsf{FilterConvergence}^{\uparrow}$ *packet that consumes* $\mathsf{MooreSmithNet}^{\uparrow}$ *data must expose the* $\mathsf{DirectedSet}^{\uparrow}$ *index row and the* $\mathsf{FilterBase}^{\uparrow}$ *handoff before any* $\mathsf{NetConvergence}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer reads convergence. The convergence route therefore passes through displayed directed indices, directed tails, and filterbase refinement data, not through host cofinal selectors, quotient net indices, hidden tail predicates, or ambient convergence relations.*

Proof. Project the filter-convergence carrier of Definition 3423.1. The visible filter row B, target row X, tail schedule T, tolerance row V, and seal row E are the only convergence-facing coordinates.

Read the Moore–Smith net carrier through chapter 3403. Its displayed route exposes a $\mathsf{DirectedSet}^{\uparrow}$ index row before the $\mathsf{FilterBase}^{\uparrow}$ handoff and before any $\mathsf{NetConvergence}^{\uparrow}$ or $\mathsf{UniformCompletion}^{\uparrow}$ consumer is named.

Apply the DirectedSet filter-net handoff of Definition 1607.4. It forces the finite index row, preorder comparisons, finite request windows, and upper-window witnesses to be visible before filter or net convergence data is consumed.

Since neither the Moore–Smith route nor the filter-convergence carrier contains host cofinal selectors, quotient net indices, hidden tail predicates, or ambient convergence relations, those objects cannot enter through filter notation. The convergence read remains the displayed route through $B, X, T, V, E, H, C, P, N$ after the directed-index and filterbase rows have been exposed. □

Falsifiable prediction: Every accepted $\mathsf{FilterConvergence}^{\uparrow}$ read factors through the visible $\mathsf{CauchyFilter}^{\uparrow}$ row, target $\mathsf{Met}^{\uparrow}$ point row, finite tail-event schedule, Metric/Real tolerance row, $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any completion-facing consumer reads convergence data.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, metric, real: $\mathsf{FilterConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a visible filter row, target point row, finite tail-event schedule, Metric/Real tolerance row, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterConvergence}^{\uparrow}$ packet is generated from a visible CauchyFilter row, a target Metric point row, a finite StreamName tail-event schedule, a Metric/Real tolerance row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3423.1, Definition 3423.2, Definition 3423.3, and Definition 3423.4 seed the finite convergence interface from visible filter rows and target metric neighborhoods to Real-facing seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected host limits, countable choice, hidden cofinal tails, host target equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible filter exposure, target-point exposure, finite tail-event exposure, Metric/Real tolerance readback, Real seal handoff, transport stability, replay stability, and ledger non-escape for BEDC.Derived.FilterConvergenceUp.

Chapter 2471

A Concrete Naming Certificate for $\mathsf{BishopCompleteMetric}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley completion language, the $\mathsf{BishopCompleteMetric}^{\uparrow}$ packet records the complete metric object reached when displayed Cauchy-filter data, regular rational readback, separated metric identity, and terminal real sealing are assembled as one finite L10 NameCert route. It sits after the metric-completion carrier of chapter 1506, the complete-metric service of chapter 128, the Cauchy-filter evidence of chapter 333, the regular rational readback of chapter 301, the Real classifier seal of chapter 38, and the separated metric classifier of chapter 1411. The packet exposes a named constructive complete metric surface; it does not assert host quotient completion equality, selected representatives, countable choice, trichotomy, or ambient real completeness outside the displayed rows.

Definition 2471.1 (Bishop complete metric carrier)**.** *A* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, F, R, S, C, E, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, including the displayed metric carrier, completion handoff, and finite tolerance readback surface.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *evidence row, recording the Cauchy-filter route from which completion-facing point names are admitted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational windows used by Real-facing consumers of the same completion data.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *evidence row, recording the zero-distance classifier boundary available for completed point equality.*
5. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording the displayed limit, modulus, and convergence-distance ledger for accepted regular Cauchy point names.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after M, F, R, S, C have been displayed.*
7. *H records componentwise* hsame *transport for the metric-completion, Cauchy-filter, regular-readback, separatedness, complete-metric, Real-seal, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from a complete-metric consumer through $M, F, R, S, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, S, C, E, H, K, P, N$. It has no coordinate for quotient completion equality, selected representatives, host metric equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompleteMetricUp.BishopCompleteMetricTasteGate single carrier alignment`

Theorem 2471.2 (Bishop complete metric NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier $B = (M, F, R, S, C, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion source, Cauchy-filter evidence, regular rational readback, separated metric classifier, complete-metric service, terminal Real seal, componentwise transport, displayed continuation replay, provenance, and local naming rows. Every complete-metric read is therefore routed through $M, F, R, S, C, E, H, K, P, N$, and no Bishop complete metric obligation is stored outside those rows.*

Proof. Project the carrier of Definition 3424.1. The visible packet consists exactly of $M, F, R, S, C, E, H, K, P, N$, so carrier admission, transport, continuation replay, provenance, and local naming are exhausted by those displayed rows.

Read the analytic dependencies in dependency order. The metric-completion source M admits the completion route, F records the Cauchy-filter evidence feeding that route, R supplies regular rational readback for Real-facing observations, S supplies the separated zero-distance classifier, and C supplies the complete-metric limit service.

The terminal seal E is downstream of those rows, while H, K, P, N preserve the same route by transport, replay, provenance, and naming. Since quotient completion equality, representatives, host equality, countable choice, trichotomy, and ambient real completeness have no carrier coordinate, they cannot appear as local obligations. □

Theorem 2471.3 (Bishop complete metric completion handoff)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier must pass through the ordered route*

$$M, F, R, S, C, E, H, K, P, N$$

before reading the terminal complete metric seal. The handoff exposes metric-completion data, Cauchy-filter evidence, regular rational readback, separated metric identity, complete-metric limit service, Real sealing, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, but it exports no quotient completion equality, selected representative, countable-choice selector, host metric equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from a completion-facing consumer of the accepted carrier. The only completion source row is M, and the only Cauchy evidence row feeding it is F.

Move next through the readback and classifier boundary. The row R exposes the regular rational windows used by Real-facing reads, while S is the separated metric route by which zero-distance comparisons are admitted.

Finally read C and E, then replay the structural rows H, K, P, N. The absence of quotient, representative, choice, host-equality, trichotomy, and ambient-completeness coordinates in Definition 3424.1 forces every accepted handoff to remain on the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompleteMetric}^{\uparrow}$ completion or Real read factors through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metriccompletion, completemetric, cauchyfilter, regseqrat, real, separatedmetric: $\mathsf{BishopCompleteMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the metric-completion source, Cauchy-filter evidence, regular rational readback, separated metric classifier, complete-metric service, terminal Real seal, transport, continuation, provenance, and local naming as one finite complete-metric packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3424.1, Definition 3424.2, and Definition 3424.3 seed the complete-metric surface over $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host metric equality, countable choice, trichotomy, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter exposure, regular rational readback, separated classifier exactness, complete-metric service compatibility, and completion-handoff nonescape over the same displayed packet.

Constructive story: The construction starts from a metric-completion source, records Cauchy-filter evidence and regular rational readback, passes through separated metric identity and complete-metric limit service, and exposes the terminal Real seal only along the same finite NameCert route.

Chapter 2472

A Concrete Naming Certificate for FredholmAlternative$^{\uparrow}$

Following Fredholm, Riesz, and compact-operator treatments of linear equations, the FredholmAlternative$^{\uparrow}$ packet records the finite BEDC surface for the kernel–range obstruction alternative attached to a displayed compact perturbation. It is read after the compact-operator certificate of chapter 1952, the Banach and Hilbert carrier rows of chapter 90 and chapter 91, the bounded-operator action surface of chapter 1644, the separation witness of chapter 3401, and the variational handoff of chapter 3419. The packet names only this displayed finite alternative route; it does not import a host Fredholm theorem, a quotient index equality, a choice-selected inverse, or an ambient spectral decomposition.

Definition 2472.1 (Fredholm alternative carrier)**.** *A* FredholmAlternative$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

with the following visible rows:

1. *S is the source endpoint row, read through the* Banach$^{\uparrow}$ *and, when an orthogonal witness is present,* Hilbert$^{\uparrow}$ *certificate surfaces;*
2. *T is the target endpoint row, read through the same Banach or Hilbert certificate discipline;*
3. *B is the* BoundedLinearOperator$^{\uparrow}$ *row naming the displayed linear action and norm-bound ledger between S and T;*
4. *K is the* CompactOperator$^{\uparrow}$ *row naming the compact perturbation witness by finite image-net schedules and tail moduli;*
5. *L is the* LaxMilgram$^{\uparrow}$ *or bounded-action handoff row used by variational consumers that read the same operator boundary;*
6. *V is the* RieszLemma$^{\uparrow}$ *separation row available when a range-obstruction witness is exposed;*
7. *R is the kernel witness row, recording the finite nonzero-kernel or kernel-refusal evidence attached to the displayed operator equation;*
8. *A is the range-obstruction witness row, recording the finite cokernel-facing obstruction, separation functional, or refusal of surjectivity over the same endpoints;*
9. *H stores componentwise* hsame *transport for the endpoint, bounded-action, compactness, variational, separation, kernel, and obstruction rows;*

10. *C stores displayed* Cont *replay for the same rows;*
11. *P is the* Pkg *provenance over* FredholmAlternative$^{\uparrow}$, CompactOperator$^{\uparrow}$, Banach$^{\uparrow}$, Hilbert$^{\uparrow}$, BoundedLinearOperato RieszLemma$^{\uparrow}$, LaxMilgram$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host inverse, quotient-space equality, index formula, selected basis of the kernel, selected complement of the range, or external spectral theorem.

Lean4 ✓ `BEDC.Derived.FredholmAlternativeUp.FredholmAlternativeTasteGate single carrier alignment`

Theorem 2472.2 (Fredholm alternative NameCert obligations)**.** *For the carrier of Definition 3425.1, the local* NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ *surface consists of exactly six finite obligations:*

1. *endpoint admission through the displayed* Banach$^{\uparrow}$ *or* Hilbert$^{\uparrow}$ *source and target rows;*
2. *bounded-action admission through the* BoundedLinearOperator$^{\uparrow}$ *row B;*
3. *compact-perturbation admission through the* CompactOperator$^{\uparrow}$ *row K;*
4. *alternative-witness exposure, requiring the kernel witness R or the range-obstruction witness A to be read as displayed finite rows over the same endpoints;*
5. *separation and variational compatibility, requiring V and L to support only the displayed obstruction or handoff route already present in the carrier;*
6. *transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier rows. The endpoint, bounded-action, compactness, variational, separation, kernel, obstruction, transport, replay, provenance, and local naming entries are the full visible inventory of F.

Read the analytic dependencies next. The Banach or Hilbert rows determine the endpoints, the bounded-operator row determines the action, and the compact-operator row supplies the finite compactness witness. These rows name BEDC certificate data rather than a host operator theorem.

The alternative witness is then local to the displayed packet. If the kernel side is exposed, it is exposed through R; if the range-obstruction side is exposed, it is exposed through A together with the separation row V. The variational row L is only a handoff through the same bounded-action surface.

Finally H, C, P, N preserve this finite route. Since the carrier contains no additional inverse, quotient, index, complement, or spectral coordinate, the six obligations exhaust the local NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ surface. □

Theorem 2472.3 (Fredholm alternative kernel–cokernel handoff)**.** *Let*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

be an accepted FredholmAlternative$^{\uparrow}$ *packet. Every consumer that reads a kernel-facing conclusion or a range-obstruction conclusion must pass through the same endpoint rows S, T, bounded-action row B, compact-operator row K, separation or variational handoff rows V, L, transport row H, replay row C, provenance row P, and local naming row N. The handoff exposes the displayed kernel witness R or the displayed range-obstruction witness A; it exposes no host inverse, quotient equality, Fredholm index, selected kernel basis, selected range complement, or ambient spectral theorem.*

Proof. Begin with the NameCert obligations of Definition 3425.2. They force endpoint admission, bounded-action admission, compact-perturbation admission, and witness exposure to be read from one finite packet.

If a consumer follows the kernel side, the only available kernel-facing row is R, and it is transported by the same H, C, P, N rows as the operator and compactness data. If a consumer follows the range-obstruction side, the obstruction row is A, with the separation support V and the variational handoff L read through the same endpoints.

Thus the two readings are not detached alternatives living over different carriers. They are the two witness exits of one finite route over $S, T, B, K, L, V, H, C, P, N$. The excluded inverse, quotient, index, basis, complement, and spectral objects are absent from the carrier and therefore cannot be exported by the handoff. □

Theorem 2472.4 (Fredholm alternative compact obstruction scope)**.** *Let*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

be an accepted $\mathsf{FredholmAlternative}^{\uparrow}$ *packet. Every compact-obstruction read exposed by F factors through the endpoint rows S, T, the bounded-action row B, the compact row K, the Riesz separation row V, the Lax–Milgram handoff row L, and exactly one displayed witness exit, either R or A. The transport, replay, provenance, and local naming rows H, C, P, N preserve that route. The read exports no host inverse, quotient index, selected complement, or ambient spectral theorem.*

Proof. Project the finite carrier rows of F. The only analytic rows available to a compact-obstruction consumer are the endpoints S, T, the bounded action B, the compact witness K, the separation row V, and the variational handoff L.

Apply the local NameCert obligations of Definition 3425.2. They force any obstruction witness to be read from the displayed alternative-witness clause, so the consumer must use one of the two witness exits already present in the packet.

Use the kernel–cokernel handoff of Definition 3425.3. The kernel-facing route exits through R, while the range-obstruction route exits through A together with the same compact, separation, and variational rows. In both cases H, C, P, N transport and replay only the displayed finite route.

Since the carrier contains no row for a host inverse, quotient index, selected complement, or ambient spectral theorem, none of these objects can be exported by the compact-obstruction scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}$.

Constructive story: A $\mathsf{FredholmAlternative}^{\uparrow}$ packet is generated from Banach or Hilbert endpoint rows, a bounded-operator row, a compact-operator row, variational and separation handoff rows, kernel and range-obstruction witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3425.1, Definition 3425.2, and Definition 3425.3 seed the finite Fredholm-alternative certificate surface consumed by compact-operator, Banach, Hilbert, bounded-operator, Riesz-separation, and Lax-Milgram routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Fredholm theorem, quotient-space equality, index formula, selected inverse, selected kernel basis, selected range complement, or standard bridge.

Upgrade path: Obligation closure requires checked endpoint admission, bounded-action admission, compact-perturbation admission, kernel witness exposure, range-obstruction exposure, separation and variational compatibility, transport, replay, provenance, and local naming for the displayed packet.

Chapter 2473

A Concrete Naming Certificate for $\mathsf{CousinCover}^{\uparrow}$

Following Cousin and Bishop-style gauge integration, the $\mathsf{CousinCover}^{\uparrow}$ packet names the reusable finite gauge-cover carrier below the tagged-subdivision conclusion of chapter 1861. It separates the cover data consumed by $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, and compact interval finite-cover arguments from the lemma-level subdivision packet. The carrier reads located interval rows from chapter 1711, finite cover and radius boundaries from chapter 1596 and chapter 452, and Real-facing names through chapter 126, chapter 301, and chapter 38.

Definition 2473.1 (Cousin cover carrier)**.** *A* $\mathsf{CousinCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, G, T, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval row, including named endpoints, endpoint order, containment tests, and the interval boundary inspected by finite bisection and cover consumers.*
2. *G is the positive gauge row. Each displayed tag request carries a positive local radius read through rational and Real-facing witnesses before any cover cell is admitted.*
3. *T is the tagged cover-cell row. Each entry records a tag, a located interval cell, containment in I, local admission by the radius in G, and the finite handoff shape read by* $\mathsf{CousinLemma}^{\uparrow}$*.*
4. *M is the mesh and refinement ledger recording endpoint comparisons, adjacency, overlap permissions, and the finite refinement relation between a cover row and any smaller admitted cover row.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposed for endpoint, tag, gauge-radius, cell, and mesh observations in I, G, T, M.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the displayed windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite windows and regular rational readbacks in W, R have been displayed.*
8. *H records componentwise* hsame *transport for interval, gauge, tagged cover, mesh, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a gauge-cover request through $I, G, T, M, W, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, T, M, W, R, E, H, C, P, N$. It has no coordinate for general open-cover compactness, a choice-selected subcover, a host real completeness theorem, quotient stream equality, or an external interval object.

Lean4 ✓ `BEDC.Derived.CousinCoverUp.CousinCoverTasteGate single carrier alignment`

Theorem 2473.2 (Cousin cover NameCert obligations)**.** *For every accepted* $\mathsf{CousinCover}^{\uparrow}$ *carrier*

$$Q = (I, G, T, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, positive gauge, tagged cover cells, mesh and refinement ledger, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *gauge locality, requiring each tagged cover cell in T to read its radius from G and to prove containment in that local gauge window;*
3. *refinement coverage, requiring M to record that the finite cell family covers the displayed interval row and that every accepted refinement remains inside the same interval and gauge rows;*
4. *finite tagged-cover handoff, requiring downstream tagged-subdivision consumers to read the cover cells through T, M, W, R, E before reaching H, C, P, N;*
5. *ledger non-escape, excluding general open-cover compactness, selected subcovers, host interval equality, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3426.1. The tuple displays the located interval, gauge, tagged cover cells, refinement ledger, finite window schedule, readback, terminal Real seal, and structural rows.

Gauge locality is read from G and T. A cell cannot be admitted unless its tag has a displayed positive radius and the cell containment test is recorded inside that local gauge window.

Refinement coverage is read from M. The ledger states both that the finite cell family covers the interval row I and that any smaller cover cell remains attached to the same interval and gauge source.

The tagged-cover handoff is the ordered route through T, M, W, R, E. The cover cells and their mesh ledger are exposed before the StreamName windows, regular rational readbacks, and terminal Real seal can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs in the carrier, quotient stream equality, selected subcovers, open-cover compactness, countable choice, host interval equality, and ambient Real completeness cannot be projected from an accepted packet. □

Theorem 2473.3 (Cousin cover gauge refinement handoff)**.** *Every* $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, *or interval finite-cover consumer of an accepted* $\mathsf{CousinCover}^{\uparrow}$ *packet reads the cover through the ordered carrier*

$$I, G, T, M, W, R, E, H, C, P, N.$$

Thus the reusable cover surface is the finite gauge-cover packet over located intervals, tagged cells, mesh refinement, StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal. No consumer receives a detached open cover, a choice-selected refinement, host interval equality, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a downstream consumer read. Carrier admission exposes the located interval row I and the positive gauge row G before any cover cell can be inspected.

The tagged cover row T and the mesh ledger M then provide the finite refinement surface. This is the part consumed by $\mathsf{CousinLemma}^{\uparrow}$ when it forms a tagged subdivision, and by $\mathsf{GaugeIntegral}^{\uparrow}$, Darboux, Riemann, and interval finite-cover consumers when they need a gauge-local finite cover.

The Real-facing portion of the read passes through W, R, E. These rows expose only finite StreamName windows, RegSeqRat readback, and the terminal Real seal attached to the interval, tag, gauge-radius, cell, and mesh observations already displayed.

The remaining rows H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the carrier has no detached open-cover or host-completeness coordinate, the consumer handoff is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CousinCover}^{\uparrow}$ packet is generated from a located interval row, a positive gauge row, finite tagged cover cells, mesh and refinement ledgers, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3426.1, Definition 3426.2, and Definition 3426.3 seed the reusable finite gauge-cover route from located intervals and positive gauges through tagged cover cells, mesh refinement, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, selected subcovers, host interval equality, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge locality, refinement coverage, finite tagged-cover handoff, and ledger non-escape for the Cousin cover packet.

Chapter 2474

A Concrete Naming Certificate for $\mathsf{RealUniformTailComparison}^{\uparrow}$

Following Bishop and Bridges constructive real sequence/tail-modulus discipline, the $\mathsf{RealUniformTailComparison}^{\uparrow}$ packet records a finite uniform comparison surface for two inspected real-sequence tails. It sits between the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window route of chapter 126, the dyadic tolerance ledger of chapter 291, the sequence-limit consumer of chapter 1413, and the limsup-facing envelope of chapter 768. The packet supplies a named L10 boundary for consumers that compare two tails under one displayed tolerance budget; it does not choose a limit, quotient streams, decide trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2474.1 (Real uniform tail comparison carrier)**.** *A* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S_A, S_B, W_A, W_B, Q_A, Q_B, D, T, L, \Lambda, H, C, P, N)$$

with the following displayed rows:

1. *S_A and S_B are the two real-sequence source rows whose tails are compared.*
2. *W_A and W_B are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows exposed for the two inspected tails.*
3. *Q_A and Q_B are the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback rows attached to those windows.*
4. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every comparison request.*
5. *T is the finite tail-budget row naming the common index floor and precision request.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer row that may read the comparison only after the windows, readbacks, ledger, and tail budget have been displayed.*
7. *Λ is the* $\mathsf{Limsup}^{\uparrow}$ *consumer row that may read the same comparison as an upper-tail envelope without selecting a subsequential limit.*
8. *H records componentwise* hsame *transport for the two source routes, the shared ledger, the tail budget, and the two consumer rows.*
9. *C records displayed* Cont *replay from a tail-comparison request through the windows, regular readbacks, dyadic ledger, tail budget, sequence-limit row, and limsup row.*
10. *P is the* Pkg *provenance over* $\mathsf{RealUniformTailComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$ *row.*
The carrier contains no coordinate for quotient stream equality, a selected limiting representative, countable choice, trichotomy of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformTailComparisonUp.RealUniformTailComparisonTasteGate single carrier alignment`

Theorem 2474.2 (Real uniform tail comparison NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$ *surface for Definition 3427.1 consists exactly of source exposure for S_A, S_B, finite-window exposure for W_A, W_B, regular readback exposure for Q_A, Q_B, shared dyadic tolerance exposure for D, common tail-budget exposure for T, sequence-limit handoff through L, limsup handoff through Λ, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $U = (S_A, S_B, W_A, W_B, Q_A, Q_B, D, T, L, \Lambda, H, C, P, N)$. The comparison route is finite: a request reads the two source rows, the two StreamName windows, the two regular rational readbacks, the shared dyadic tolerance ledger, and the common tail budget before it reaches either consumer row.

The structural rows preserve that same route. Componentwise hsame transport keeps the two tail presentations and the shared ledger in their displayed sibling surfaces, while Cont replay records the order in which the request reaches the sequence-limit and limsup consumers.

No excluded object is a row of the carrier. Therefore the listed rows exhaust the seed NameCert obligations for the uniform tail-comparison packet. □

Theorem 2474.3 (Real uniform tail comparison RegSeqRat handoff)**.** *Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *or* $\mathsf{Limsup}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *packet must read the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows Q_A, Q_B through the displayed windows W_A, W_B and the shared dyadic ledger D before using the common tail budget T. The handoff exports a finite comparison of regular rational tails; it does not export quotient stream equality, a selected limit, countable choice, trichotomy, or ambient real completeness.*

Proof. Start with the carrier definition. The two regular rational readbacks Q_A, Q_B are attached to the two finite window rows W_A, W_B, and both sides share the single dyadic tolerance ledger D.

The common tail-budget row T is displayed after those rows. Hence a sequence-limit or limsup consumer cannot inspect the comparison budget before the two regular rational tail presentations and their shared tolerance surface are available.

The remaining rows H, C, P, N transport, replay, package, and name this same finite route. They add no quotient stream, selected limit, choice operator, trichotomy decision, or ambient completeness coordinate. □

Theorem 2474.4 (Real uniform tail comparison real-seal boundary)**.** *Every real-facing read exported by an accepted* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *packet factors through the displayed source rows, StreamName windows, RegSeqRat readbacks, shared DyadicRatCore ledger, common tail budget, and the two consumer rows L, Λ before any* $\mathsf{Real}^{\uparrow}$ *seal can be used downstream. The packet therefore supplies a boundary for tail comparison under a common tolerance budget, not a proof of selected convergence, quotient equality of streams, real trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The only real-facing rows named by the packet are reached through the displayed sequence-limit and limsup consumers. Those consumers are positioned after $S_A, S_B, W_A, W_B, Q_A, Q_B, D, T$ in the carrier.

Transport and continuation replay preserve that order componentwise. Package provenance and local naming can expose the route, but they cannot insert a hidden real seal or a selected representative before the finite comparison rows.

Thus every downstream $\mathsf{Real}^{\uparrow}$ read is bounded by the displayed tail-comparison route, and the stated nonclaims follow from the absence of the excluded coordinates in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformTailComparison}^{\uparrow}$ packet is generated from two real-sequence source rows, two StreamName finite-window rows, two RegSeqRat readbacks, one shared DyadicRatCore tolerance ledger, one common tail budget, RealSequenceLimit and Limsup consumer rows, componentwise transport, displayed continuation replay, package provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3427.1, Definition 3427.2, Definition 3427.3, and Definition 3427.4 seed the finite L10 tail-comparison route over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient streams, real trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, uniform tail-budget exposure, RegSeqRat handoff, real-seal boundary, and consumer nonescape for BEDC.Derived.RealUniformTailComparisonUp.

Chapter 2475

A Concrete Naming Certificate for $\mathsf{CauchyChainCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyChainCriterion}^{\uparrow}$ packet records the finite criterion surface by which a Cauchy-space consumer reads a chain of adjacent rational approximations before reaching a Real endpoint. It sits between the $\mathsf{CauchySpace}^{\uparrow}$ interface of chapter 1847, the selected-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite chain criterion, not a quotient completion, selected limit, host equality, countable-choice schedule, or ambient completeness principle.

Definition 2475.1 (CauchyChainCriterion carrier)**.** *A* $\mathsf{CauchyChainCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchySpace}^{\uparrow}$ *route that admits the finite filter-uniform source read.*
2. *L is the finite chain-link ledger, listing adjacent approximation links and their requested tolerances.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family from which each link endpoint is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and adjacent links.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the finite sums and refinements needed along the chain.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, L, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the source, links, windows, readbacks, tolerance ledger, endpoint seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the Cauchy-space source through the chain links, selected windows, regular readbacks, dyadic tolerance ledger, and endpoint seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyChainCriterion}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, W, R, D, E, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyChainCriterionUp.CauchyChainCriterionUp`
Lean4 ✓ `BEDC.Derived.CauchyChainCriterionUp.CauchyChainCriterionTasteGate single carrier alignment`

Theorem 2475.2 (CauchyChainCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyChainCriterion}^{\uparrow}$ *carrier* $K = (S, L, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, adjacent-link coverage through* L*, tail-chain transitivity through the displayed dyadic ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*, and non-escape for quotient completion, selected limits, host equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3428.1. Carrier admission exposes the Cauchy-space source S, the link ledger L, the selected windows W, the regular readback R, the tolerance ledger D, and the endpoint seal E as named finite rows.

Adjacent-link coverage is read from L and W: each link endpoint must be present in the selected window family before the corresponding regular rational readback may be inspected.

Tail-chain transitivity is the displayed use of D. The dyadic ledger records the finite tolerance sums and refinements connecting successive links, so a tail-chain consumer receives a finite ledger rather than an external convergence theorem.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is forced through R before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be read. The structural rows H, C, P, N transport, replay, source, and name the same route, and the omitted host-side objects have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyChainCriterion}^{\uparrow}$ packet is generated from a $\mathsf{CauchySpace}^{\uparrow}$ source route, finite adjacent-link ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3428.1 and Definition 3428.2 seed the finite Cauchy-space chain criterion from $\mathsf{CauchySpace}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host real equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjacent-link coverage, tail-chain transitivity, regular-name handoff, endpoint-seal routing, and non-escape proofs for BEDC.Derived.CauchyChainCriterionUp.

Chapter 2476

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$

Following Riemann, Darboux, and Bishop–Bridges constructive analysis, the ClosedBoundedIntervalDarbouxPartition packet records the finite partition surface needed to run Darboux and Riemann integration over one displayed closed bounded interval. It binds the base interval of chapter 1591, the located partition row of chapter 2248, the finite Darboux-sum boundary of chapter 1880, the Darboux integration packet of chapter 1460, and the tagged-mesh consumer of chapter 1526. The packet supplies a closed-interval-specific Darboux partition certificate; it is not an open-cover compactness theorem, a selected tag family, a quotient function-equality principle, a measure-completion object, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 2476.1 (Closed bounded interval Darboux partition carrier)**.** *A* ClosedBoundedIntervalDarbouxPartition *carrier is a finite* BHist *packet*

$$K = (I, M, L, U, S, D, J, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row, carrying lower and upper located endpoints, endpoint order, rational and dyadic containment ledgers, finite windows, readbacks, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
2. *M is the* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row over the same interval endpoints, exposing the finite cell mesh, cell adjacency, dyadic refinement, rational-cell witnesses, and located endpoint readbacks;*
3. *L and U are lower and upper Darboux bracket rows read through* $\mathsf{DarbouxSum}^{\uparrow}$ *over the cells of M;*
4. *S is the shared Darboux upper-lower sum surface, recording the finite lower sum, finite upper sum, and the visible gap ledger;*
5. *D is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *handoff row attached to the same interval, mesh, lower bracket, upper bracket, and gap ledger;*
6. *J is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer handoff row, read only after the Darboux row D and the partition row M have been displayed;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback and seal row reached from the finite gap ledger;*

8. *H records componentwise* hsame *transport of the interval, partition, Darboux brackets, sum gap, Darboux handoff, Riemann handoff, and real-seal rows;*
9. *C records displayed* Cont *replay from interval endpoints to partition cells, from cells to Darboux sums, and from Darboux gap data to the integration handoff rows;*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected tag families, quotient function equality, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Its public object is the finite closed-interval partition route from I through $M, L, U, S, D, J, R, H, C, P, N$.*

```
Lean4 ✓  BEDC.Derived.ClosedBoundedIntervalDarbouxPartitionUp.TasteGate.ClosedBoundedIntervalDarbouxPart
single carrier alignment
```

Theorem 2476.2 (Closed bounded interval Darboux partition NameCert obligations)**.** *For the carrier of Definition 3429.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, M, L, U, S, D, J, R, H, C, P, N)$; source-interval exactness through I; located partition exactness through M; lower and upper Darboux bracket exposure through L, U; finite Darboux-sum exactness and gap visibility through S; Darboux handoff through D; Riemann handoff through J;* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *sealing through R; componentwise transport through H; continuation replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier. The interval row I fixes the lower and upper located endpoints, endpoint order, containment ledgers, finite readbacks, and terminal real seal before any partition row can be consumed.

Next read the located partition row M. Since M is attached to the same endpoint and containment data, every Darboux cell and refinement read remains a finite closed-interval read rather than a host interval partition.

The lower and upper bracket rows L, U and the sum row S are then inspected through the Darboux-sum surface. They record finite lower and upper sums over displayed cells, and the gap ledger is visible before either integration handoff row can be used.

Finally the handoff rows D, J, R and the structural rows H, C, P, N preserve and name exactly this route. None of these rows contains open-cover compactness, a selected tag family, quotient function equality, measure completion, or ambient real completeness, so the listed obligations exhaust the NameCert surface. □

Theorem 2476.3 (Closed bounded interval Darboux partition refinement stability)**.** *Let $K = (I, M, L, U, S, D, J, R, H, C, P, N)$ and $K' = (I', M', L', U', S', D', J', R', H', C', P', N')$ be accepted* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *carriers classified together by componentwise* hsame *transport. If M' is read as a displayed common refinement of M over the transported closed bounded interval row, then the transported lower bracket, upper bracket, finite sum gap, Darboux handoff, Riemann handoff, and* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal are all read through the same finite refinement route.*

The refinement therefore preserves the Darboux partition certificate without adding selected tags, quotient equality of integrands, measure completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source interval row. Componentwise transport carries I to I', including endpoint order, containment ledgers, finite windows, readbacks, and the real seal, before any refined cell can be read.

Read the common refinement through M'. Its cells are located relative to the transported endpoints and containment ledgers, so each refined cell points back to the same closed bounded interval route.

The Darboux bracket and sum rows transport next. The rows L', U', S' compute lower and upper sums only from the displayed refined cells and carried values, hence the visible gap is preserved as a finite Darboux comparison.

The integration handoff rows D', J', R' are downstream of that comparison. Since H' and C' replay the same interval-to-partition-to-gap route and P', N' only source and name it, refinement stability cannot introduce the excluded host-side objects. □

Theorem 2476.4 (Closed bounded interval Darboux partition Riemann handoff)**.** *Every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of a* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *packet receives the closed interval only through I, the located partition only through M, the Darboux lower and upper bracket rows only through L, U, S, D, and the real-facing seal only through R. The handoff row J may place a tagged-mesh consumer over that displayed partition route, but it does not supply an independent tag-selection family, quotient equality of integrands, measure completion, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 3429.2. It makes the interval, partition, bracket, sum, gap, Darboux handoff, Riemann handoff, seal, transport, replay, provenance, and local naming rows the entire public surface.

Apply Definition 3429.3. Any consumer-facing refinement of the partition must preserve the same endpoint, containment, cell, bracket, sum, and gap route before the handoff rows are read.

Now compare with the Riemann handoff boundary of Definition 1526.4. The Riemann consumer may add its own tag-containment and finite-sum rows, but those rows sit downstream of the displayed closed-interval Darboux partition route.

Thus J is a consumer handoff for the finite route already named by $I, M, L, U, S, D, R, H, C, P, N$. It cannot export an independent selected tag family, quotient integrand equality, measure-completion object, open-cover compactness proof, or ambient real-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ packet is generated from a closed bounded interval row, a located real interval partition row, lower and upper Darboux bracket rows, a finite Darboux-sum gap ledger, Darboux and Riemann integration handoff rows, a RegSeqRat/Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3429.1, Definition 3429.2, Definition 3429.3, and Definition 3429.4 seed the closed bounded interval Darboux partition boundary over $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected tag families, quotient function equality, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located partition exactness, Darboux bracket exposure, finite gap preservation, Riemann handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.ClosedBoundedInterv

Chapter 2477

A Concrete Naming Certificate for $\mathsf{FiniteCoverRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet records the constructive finite refinement step used by compact metric and completion-facing consumers when two displayed finite covers must be compared. It sits over the compact metric finite-net surface of chapter 129, the dyadic common-refinement ledger of chapter 1877, and the finite real ball-cover surface of chapter 2377. The packet names only a finite refinement route through displayed cover rows; it is not open-cover compactness, not a selected Lebesgue number, and not countable choice.

Definition 2477.1 (Finite cover refinement carrier)**.** *A* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, D, F, Q, U, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite-net and metric-distance data bound the cover comparison.*
2. *D is the* $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *common-refinement ledger comparing the two dyadic cover presentations seen by the consumer.*
3. *F is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row naming the finite real-ball cover, its coverage witnesses, and its refinement ledger.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *tolerance row used to read radii and overlap bounds.*
5. *U is the finite* $\mathsf{ProbeBundle}$ *row listing the inspected centers, witnesses, and retained members.*
6. *W is the refinement witness row, pairing each retained probe with a member of the common finite refinement and recording the displayed inclusion or radius comparison.*
7. *H is the componentwise* hsame *transport row for the compact metric, dyadic refinement, finite real cover, rational tolerance, probe bundle, witness, provenance, and local naming coordinates.*
8. *C is the displayed* Cont *replay route for compact-metric reads, dyadic cover reads, finite real-cover reads, rational-radius reads, and refinement-witness reads.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicCoverRefinement}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$ *evidence row.*

The carrier has no coordinate for an arbitrary open cover, a host finite-subcover selector, a selected Lebesgue number, a countable-choice sequence, quotient real equality, or ambient compactness.

Theorem 2477.2 (Finite cover refinement NameCert obligations)**.** *For the carrier of Definition 3430.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (K, D, F, Q, U, W, H, C, P, N)$*, compact-metric source exposure through* K*, dyadic common-refinement exposure through* D*, finite real-cover exposure through* F*, rational tolerance exposure through* Q*, finite probe-bundle exposure through* U*, refinement-witness exposure through* W*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every compact-metric, finite-real-cover, or completion-facing consumer of this packet must therefore read the displayed route

$$K \longmapsto D \longmapsto F \longmapsto Q \longmapsto U \longmapsto W \longmapsto (H, C, P, N)$$

before it can compare the two finite covers. The route exports no open-cover compactness theorem, no selected Lebesgue number, no countable-choice selector, no quotient representative, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier tuple. The rows K, D, F, Q, U, W are the only analytic coordinates, and the rows H, C, P, N transport, replay, source, and name that same finite packet.

Read the dependency chain in order. The compact-metric row supplies the finite source data, the dyadic refinement row supplies the common-refinement ledger, the finite real-cover row supplies coverage witnesses, and the rational and probe rows expose the finite tolerance window in which W is meaningful.

The structural rows create no additional compactness principle. Since the carrier contains no arbitrary open cover, selected Lebesgue number, countable-choice sequence, quotient representative, or ambient completeness row, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.FiniteCoverRefinementUp.FiniteCoverRefinementTasteGate single carrier alignment`

Theorem 2477.3 (Finite cover refinement compact handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{FiniteRealCover}^{\uparrow}$*, or regular-completion consumer citing an accepted* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *packet receives only the finite comparison route displayed in Definition 3430.2. In particular, the consumer may compare retained cover members and dyadic radius rows through* D, F, Q, U, W*, but it cannot replace that comparison by open-cover compactness, a selected Lebesgue number, countable choice, a host subcover selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligation theorem. It places every cover comparison on the displayed route from the compact metric source through the dyadic and finite real-cover rows.

The witness row W is meaningful only after Q and U have exposed the finite tolerance and probe rows. Componentwise transport and continuation replay preserve those coordinates without adding a second comparison route.

The excluded objects are absent from the carrier, provenance, and local naming rows. Hence the compact-facing handoff is exactly the finite refinement packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a DyadicCoverRefinementUp common-refinement ledger, a FiniteRealCoverUp coverage row, RatUp tolerance rows, finite ProbeBundle rows, a refinement witness, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 3430.1, Definition 3430.2, and Definition 3430.3 define the finite refinement packet and its compact-facing handoff over displayed finite cover rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, countable choice, host subcover selection, quotient real equality, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement witness exactness, finite-cover transport, compact-facing handoff, and ledger refusal for BEDC.Derived.FiniteCoverRefinementUp.

Chapter 2478

A Concrete Naming Certificate for $\mathsf{LocatedIntervalCompactness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ packet records the finite-cover compactness step available for a displayed located real interval when the cover has already been reduced to explicit dyadic mesh data. It sits over the located endpoint surface of chapter 394, the dyadic rational arithmetic of chapter 291, the terminal real-name rows of chapter 38, the finite-net compact metric surface of chapter 129, and the finite refinement route of chapter 3430. The packet names only endpoint rows, mesh cells, finite cover rows, refinement witnesses, compact-metric handoff, transport, replay, provenance, and local naming evidence; it is not a general Heine–Borel theorem and does not import arbitrary open-cover compactness.

Definition 2478.1 (Located interval compactness carrier)**.** *A* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, E, M, F, W, K, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing the lower and upper endpoint names, rational cell ledger, dyadic endpoint refinements, finite endpoint windows, and the downstream* $\mathsf{Real}^{\uparrow}$ *seals.*
2. *E is the endpoint exposure row that records the two endpoint rows as the boundary read consumed by the finite mesh.*
3. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, listing a finite dyadic subdivision with the endpoint cells and mesh-step witnesses used by the interval packet.*
4. *F is the finite cover row, containing only displayed finite cover members indexed by the mesh cells.*
5. *W is the refinement witness row, pairing each mesh cell with a retained finite cover member and the displayed inclusion or overlap comparison.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row read after the mesh and finite cover rows have been displayed.*
7. *H records componentwise* hsame *transport for the located interval, endpoint exposure, mesh, finite cover, refinement witness, compact-metric handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from endpoint inspection through dyadic mesh construction, finite cover lookup, refinement witness replay, compact-metric handoff, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, selected subcover functional, sequential compactness principle, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalCompactnessUp.LocatedIntervalCompactnessTasteGate single carrier alignment`

Theorem 2478.2 (Located interval compactness carrier exposure)**.** *For every accepted* LocatedIntervalCompactness *packet*

$$I = (L, E, M, F, W, K, H, C, P, N),$$

the local certificate exposes exactly the located endpoint rows, the dyadic mesh row, the finite cover row, the refinement witness, the compact-metric handoff, componentwise transport, displayed replay, provenance, and local NameCert rows. Every finite-cover compactness read factors through

$$L \longmapsto E \longmapsto M \longmapsto F \longmapsto W \longmapsto K \longmapsto (H, C, P, N)$$

before any compact-metric or completion-facing consumer can use the packet. The exported route contains no general open-cover compactness theorem, no sequential compactness principle, no choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. Project the carrier of Definition 3431.1. The interval source is the row L, and the only endpoint-facing data available to the compactness route are the displayed endpoint rows named by E.

The finite construction then proceeds by dyadic mesh induction over M. The base cell reads the endpoint exposure row and the corresponding finite cover entry in F. The induction step appends the next displayed dyadic cell, reads its cover member from the same finite cover row, and records the retained overlap or inclusion comparison in W.

Apply the finite refinement route of chapter 3430. The witness row W is the only refinement surface exported to the compact-metric handoff K, so a compact-facing consumer receives a finite mesh-and-cover comparison rather than an arbitrary open cover.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for a selected subcover functional, sequential compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be exported by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ packet is generated from a LocatedIntervalUp endpoint row, an endpoint exposure row, a finite DyadicRatCoreUp mesh, finite cover rows, refinement witnesses, a CompactMetricUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 3431.1 and Definition 3431.2 seed the finite located-interval compactness surface over endpoint rows, dyadic mesh rows, finite cover rows, refinement witnesses, compact-metric handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, arbitrary open-cover compactness, sequential compactness, choice, quotient real equality, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh induction, finite cover refinement, compact-metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalCompactnessUp.

Chapter 2479

A Concrete Naming Certificate for $\mathsf{LimitComparisonTest}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Hardy's classical comparison-test presentation, the $\mathsf{LimitComparisonTest}^{\uparrow}$ packet records the finite route by which a source sequence is compared to a positive comparator sequence before a real-facing conclusion is exposed. It is the sequence-level sibling of chapter 2023: $\mathsf{SeriesLimitComparison}^{\uparrow}$ consumes the resulting comparison surface for series tails, while $\mathsf{LimitComparisonTest}^{\uparrow}$ names the underlying ratio-test carrier over $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet exposes finite ratio-control data; it does not import a host quotient limit, quotient stream equality, real trichotomy, countable choice, or ambient completeness.

Definition 2479.1 (Limit comparison test carrier)**.** *A* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the inspected finite windows of the sequence being tested.*
2. *T is the comparator* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the matched finite windows of the positive comparator sequence.*
3. *W_S and W_T are the source and comparator observation schedules used by downstream sequence and series consumers.*
4. *Q_S and Q_T are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the displayed source and comparator windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the ratio slack, positivity margin, comparison radius, and endpoint tolerance consumed by the test.*
6. *R is the eventual positive ratio ledger. It displays a finite threshold after which comparator observations are positive and source-to-comparator comparisons remain inside the dyadic band recorded in D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, comparator windows, regular readbacks, dyadic ledger, and ratio ledger have been displayed.*

8. *H stores componentwise* hsame *transport for the source windows, comparator windows, readback rows, dyadic ledger, ratio ledger, real seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a limit-comparison request through $S, T, W_S, W_T, Q_S, Q_T, D, R$ and then to the real-facing seal E.*
10. *P is the* Pkg *provenance over* $\mathsf{LimitComparisonTest}^{\uparrow}$, $\mathsf{SeriesLimitComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N$. It has no coordinate for a host quotient limit, quotient stream equality, a selected infinite tail, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 2479.2 (Limit comparison test NameCert obligations)**.** *For every accepted* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source windows, comparator windows, readback rows, dyadic tolerance ledger, eventual positive ratio ledger, real seal, transport, replay, provenance, and local naming rows;*
2. *source-window exactness, requiring every inspected source observation to pass through S, W_S, Q_S, D before the ratio ledger is read;*
3. *comparator-window exactness, requiring every inspected comparator observation to pass through T, W_T, Q_T, D before positivity or ratio control is used;*
4. *eventual positivity, requiring R to display the finite threshold and positive comparator margin used by all later comparisons;*
5. *ratio-band exactness, requiring R and D to display the dyadic lower and upper comparison bounds before any real-facing seal is consumed;*
6. *real-seal nonescape, requiring E to be downstream of the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and ratio-ledger rows and to export no host quotient limit, quotient stream equality, selected tail, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge datum;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same finite route through H, C, P, N.*

No limit-comparison-test obligation is stored outside these rows.

Proof. Project the accepted packet $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$. The carrier definition exposes the source and comparator windows, their regular readbacks, the shared dyadic ledger, the eventual positive ratio ledger, and the structural rows.

The source side and comparator side are read separately. Source observations must pass through S, W_S, Q_S, D, while comparator observations must pass through T, W_T, Q_T, D.

The ratio ledger R then supplies the finite positivity threshold and dyadic comparison band. Since these are displayed rows, a consumer cannot replace them by a host quotient limit or by an undisplayed infinite tail.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Because no excluded analytic or choice object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.LimitComparisonTestUp.LimitComparisonTestTasteGate single carrier alignment`

Theorem 2479.3 (Limit comparison test Cauchy handoff)**.** *Let* $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier. Every Cauchy-facing or series-facing read of the source route factors through*

$$T, W_T, Q_T, D \longmapsto R \longmapsto S, W_S, Q_S, D \longmapsto E, H, C, P, N.$$

Thus comparator control is handed to the source route only through the displayed eventual positive ratio ledger and dyadic tolerance budget. The handoff exports finite BEDC comparison data, not a host quotient limit, quotient stream equality, selected tail, countable choice, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Cauchy-facing or series-facing request for the source route. The comparator data available to the carrier are exactly T, W_T, Q_T, D, so the request first reads the displayed comparator windows, readback row, and tolerance ledger.

The ledger R supplies the finite threshold, positivity margin, and dyadic ratio band that connect later comparator observations to later source observations. This is a packet row rather than a host convergence assertion.

The request then returns to the source rows S, W_S, Q_S, D. These rows expose the finite source windows and regular rational readbacks that a downstream $\mathsf{SeriesLimitComparison}^{\uparrow}$ or Cauchy consumer may inspect.

The real seal and structural rows E, H, C, P, N preserve that finite route. Since the carrier has no coordinate for quotient limits, quotient streams, choice, trichotomy, or ambient completeness, every such handoff is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$.

Constructive story: A $\mathsf{LimitComparisonTest}^{\uparrow}$ packet is generated from source and comparator StreamName windows, RegSeqRat readback rows, a DyadicRatCore tolerance ledger, an eventual positive ratio ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$, seed)

Scope closed: Definition 3432.1, Definition 3432.2, and Definition 3432.3 seed the finite constructive limit-comparison route from positive comparator windows through ratio budgeting to source-route RealUp sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient limits, quotient stream equality, selected tails, countable choice, real trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and comparator window exactness, eventual positive ratio-ledger stability, DyadicRatCore and RegSeqRat budget handoff, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.LimitComparisonTestUp.

Chapter 2480

A Concrete Naming Certificate for $\mathsf{DenjoyIntegral}^{\uparrow}$

Following Denjoy, Henstock, Kurzweil, and Bishop–Bridges, the $\mathsf{DenjoyIntegral}^{\uparrow}$ packet records the constructive integration neighbour of $\mathsf{GaugeIntegral}^{\uparrow}$ that keeps exceptional variation data visible before any real-valued integral read is exported. It sits beside the gauge-partition route of chapter 1999, the public integral row of chapter 123, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Denjoy-style handoff for one displayed integrand window; it is not measure completion, a quotient of functions, a host almost-everywhere equality theorem, a selected primitive, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2480.1 (Denjoy integral carrier)**.** *A* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (G, I, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *source row, read through the finite gauge-partition route of chapter 1999.*
2. *I is the public* $\mathsf{Integral}^{\uparrow}$ *row for the same displayed integrand and interval window.*
3. *V is the exceptional-variation ledger, recording the finite variation exclusions, controlled subintervals, and refusal rows that prevent the handoff from becoming a hidden measure-completion argument.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational integral approximants and tolerance requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, I, V, R have been displayed.*
6. *H records componentwise* hsame *transport for the gauge source, integral row, exceptional-variation ledger, regular readback, real seal, continuation replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the same route.*
8. *P is the* Pkg *provenance over* $\mathsf{DenjoyIntegral}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$ *row.*

The carrier has no coordinate for measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.DenjoyIntegralUp.DenjoyIntegralUp`

Theorem 2480.2 (Denjoy integral NameCert obligations)**.** *For every accepted* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier* $D = (G, I, V, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed gauge-integral source, public integral row, exceptional-variation ledger, RegSeqRat readback, Real seal, hsame transport, Cont replay, provenance, and local naming;*
2. *gauge compatibility, requiring every partition-facing request to pass through* G *and the sibling route of chapter 1999;*
3. *public integral visibility, requiring the exported integral read to pass through* I *before any regular-rational or Real-facing row is consumed;*
4. *exceptional-variation accountability, requiring every exceptional subwindow or variation refusal to be displayed in* V *rather than hidden in a measure-completion or almost-everywhere premise;*
5. *readback exactness, requiring rational approximants and tolerance requests to pass through* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is available;*
6. *structural stability, requiring* H, C, P, N *to transport, replay, source, and name only the rows already listed.*

No Denjoy-integral obligation is stored outside D*.*

Proof. Project the carrier of Definition 3433.1. The displayed packet exposes exactly the gauge source, public integral row, exceptional-variation ledger, regular readback, real seal, and structural rows.

Gauge compatibility is the row G. It routes every partition-facing read through the same finite gauge data used by chapter 1999, so a consumer cannot replace the packet by a selected family of gauges or by an ambient compactness theorem.

Integral visibility and exceptional-variation accountability are the rows I and V. The public integral row records the exported integral surface, while V keeps exceptional variation and refusal data as displayed finite ledger entries.

The regular readback and real seal are downstream of those rows. The row R supplies rational approximants and tolerance requests before E is exposed, and H, C, P, N preserve only that route. Since the carrier contains no measure-completion, quotient-function, host almost-everywhere, selected-primitive, choice, or completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2480.3 (Denjoy integral gauge handoff)**.** *Every* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier* $D = (G, I, V, R, E, H, C, P, N)$ *must read the finite route*

$$G \longmapsto I \longmapsto V \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The handoff therefore exposes the gauge-integral source, public integral row, exceptional-variation ledger, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming before any Denjoy integral value is consumed.

Proof. Start with an integral-facing, regular-rational-facing, or Real-facing read of D. By Definition 3433.1, the only gauge source is G and the only public integral coordinate is I.

The exceptional side is forced through V. A consumer must inspect the displayed variation ledger and refusal rows before the handoff reaches rational approximants or a real-name seal.

The route into the real-facing value is $R \longmapsto E$. The regular rational readback records the finite approximants and tolerance requests, and the terminal seal can be replayed only through H and C with provenance P and local name N.

Consequently the Denjoy packet cannot export a value by bypassing $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$, and it cannot import measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, or ambient real completeness as hidden data. □

Falsifiable prediction: Every $\mathsf{DenjoyIntegral}^{\uparrow}$ integral-facing or Real-facing read factors through displayed GaugeIntegral, Integral, exceptional-variation ledger, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: gauge-integral, integral, regseqrat, real: $\mathsf{DenjoyIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a gauge source, public integral row, exceptional-variation ledger, regular rational approximants, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3433.1, Definition 3433.2, and Definition 3433.3 seed the finite Denjoy-integration route through $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge compatibility, public integral visibility, exceptional-variation accountability, regular rational readback exactness, Real seal handoff, transport stability, and replay stability for BEDC.Derived.DenjoyIntegralUp.

Constructive story: A $\mathsf{DenjoyIntegral}^{\uparrow}$ packet is generated from a displayed GaugeIntegral source, public Integral row, exceptional-variation ledger, RegSeqRat readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2481

A Concrete Naming Certificate for $\mathsf{DecidableBetaFrontier}^{\uparrow}$

The $\mathsf{DecidableBetaFrontier}^{\uparrow}$ packet realizes the bounded MetaCIC beta-audit window named in chapter 9386 as finite NameCert data. It sits beside the literal lambda-calculus substrate of chapter 200, the beta-redex frontier of chapter 1008, the decidability audit packet of chapter 746, and the GroundCompiler audit map of chapter 836. The packet records a bounded audit frontier for beta checking; it does not claim full normalization, arbitrary closed beta-conversion decidability, closed subject reduction, or global Church–Rosser.

Definition 2481.1 (Decidable beta frontier carrier)**.** *A* $\mathsf{DecidableBetaFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, L, W, Q, A, G, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC audit-source row, reading the open-problem and decidability boundaries of chapter 9386.*
2. *L is the* $\mathsf{LambdaCalc}^{\uparrow}$ *syntax and beta-step row, anchored in chapter 200.*
3. *W is the bounded beta-frontier window: it fixes the finite depth, fuel, closedness observations, and normal-output comparisons that the packet may inspect.*
4. *Q is the bounded checker row, read through the finite decidability surface of chapter 746.*
5. *A is the audit refusal row, recording why the bounded window is not a proof of full normalization or unrestricted beta-conversion decidability.*
6. *G is the GroundCompiler provenance row, read through chapter 836 only as compiler-layer audit data.*
7. *R is the redex-frontier handoff row, read through chapter 1008.*
8. *H records componentwise* hsame *transport for the audit source, lambda syntax row, bounded window, checker row, refusal row, compiler provenance, frontier handoff, continuation replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the bounded audit route.*
10. *P is the* Pkg *provenance over* $\mathsf{DecidableBetaFrontier}^{\uparrow}$*, MetaCIC audit rows,* $\mathsf{LambdaCalc}^{\uparrow}$*, bounded checker rows, GroundCompiler audit rows, beta-frontier rows,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DecidableBetaFrontier}^{\uparrow}}$ *row.*

The carrier has no coordinate for full normalization, arbitrary closed conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, host evaluator equality, or hidden compiler evidence.

Theorem 2481.2 (Decidable beta frontier NameCert obligations)**.** *For every accepted* $\mathsf{DecidableBetaFrontier}^{\uparrow}$ *carrier* $B = (M, L, W, Q, A, G, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DecidableBetaFrontier}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the MetaCIC audit source, LambdaCalc syntax row, bounded beta window, checker row, refusal row, GroundCompiler provenance, redex-frontier handoff, transport, replay, provenance, and local naming;*
2. *bounded-window exactness, requiring every beta-check request to name the finite depth, fuel, closedness observations, and normal-output comparisons in W;*
3. *checker accountability, requiring Q to expose only bounded checker facts already visible at the decidability audit surface;*
4. *frontier refusal, requiring A to record the non-claim against full normalization, unrestricted conversion decidability, subject reduction, and global confluence;*
5. *compiler-layer provenance, requiring G to be read only as a GroundCompiler audit row and not as hidden semantic evidence;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No decidable beta frontier obligation is stored outside B.

Proof. Project the carrier of Definition 3434.1. The displayed packet exposes the MetaCIC audit source, lambda syntax row, bounded window, checker row, refusal row, compiler provenance, redex-frontier handoff, and structural rows.

Bounded-window exactness is the row W. A beta-check request cannot enter the packet unless its finite depth, fuel, closedness observations, and normal-output comparisons have been fixed as displayed data.

Checker accountability is the row Q, and frontier refusal is the row A. The checker row may report bounded facts, while the refusal row keeps full normalization, unrestricted conversion decidability, subject reduction, and global confluence outside the carrier.

The compiler and redex rows are G and R. They provide audit provenance and frontier routing, not semantic shortcuts. Since H, C, P, N only transport, replay, source, and name displayed rows, the local obligations are exhausted by the finite packet. □

Theorem 2481.3 (Decidable beta frontier MetaCIC handoff)**.** *Every MetaCIC,* $\mathsf{LambdaCalc}^{\uparrow}$*, or GroundCompiler-facing consumer of an accepted* $\mathsf{DecidableBetaFrontier}^{\uparrow}$ *carrier* $B = (M, L, W, Q, A, G, R, H, C, P,$ *must read the bounded route*

$$M, L \longmapsto W \longmapsto Q \longmapsto A \longmapsto G, R \longmapsto H, C, P, N.$$

The handoff exposes the MetaCIC audit source, LambdaCalc syntax, bounded beta window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, transport, replay, provenance, and local naming before any beta-frontier conclusion is consumed.

Proof. Start with a MetaCIC, LambdaCalc, or GroundCompiler-facing read of B. The carrier gives the audit source M and lambda syntax row L as the only source coordinates.

The finite audit boundary is $W \longmapsto Q$. The bounded window fixes the observed beta frontier, and the checker row reads only that finite window.

The refusal and provenance rows are then forced into the route. The row A records the non-claims, while G and R expose GroundCompiler audit provenance and redex-frontier handoff without adding hidden semantic evidence.

Finally H, C, P, N replay and name the same packet. Therefore no consumer can turn the bounded frontier into full normalization, unrestricted beta-conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, host evaluator equality, or a compiler fact absent from the displayed rows. □

Theorem 2481.4 (Decidable beta frontier TasteGate boundary)**.** *The Lean-facing* TasteGate.lean *site for* DecidableBetaFrontier$^{\uparrow}$ *must provide explicit* Nontrivial DecidableBetaFrontier$^{\uparrow}$ *and* FieldFaithful DecidableBet *instances for the displayed rows* $M, L, W, Q, A, G, R, H, C, P, N$*. These instances preserve the distinction between bounded checker evidence, redex-frontier handoff, refusal ledger, and compiler-layer provenance before any checked carrier-alignment target may cite* BEDC.Derived.DecidableBetaFrontierUp*.*

Lean4 ✓ `BEDC.Derived.DecidableBetaFrontierUp.DecidableBetaFrontierTasteGate single carrier alignment`

Proof. Read the carrier rows of Definition 3434.1. The bounded window, checker row, refusal ledger, compiler provenance, and redex-frontier handoff have different consumer roles and cannot be collapsed without changing the NameCert surface.

Nontriviality rules out the degenerate packet in which the frontier has no distinguishable bounded audit row. Field faithfulness requires each displayed coordinate $M, L, W, Q, A, G, R, H, C, P, N$ to survive transport as the same named row.

Thus the later Lean TasteGate site must keep those rows separate before it records a carrier-alignment theorem for the packet. □

Falsifiable prediction: Every DecidableBetaFrontier$^{\uparrow}$ beta-frontier read factors through displayed MetaCIC audit source, LambdaCalc syntax, bounded window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: MetaCIC, LambdaCalc, GroundCompiler: DecidableBetaFrontier$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded beta-window data, checker accountability, refusal rows, compiler-layer provenance, redex-frontier handoff, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 3434.1, Definition 3434.2, Definition 3434.3, and Definition 3434.4 seed the finite bounded beta-frontier route over MetaCIC, LambdaCalc, GroundCompiler, and beta-audit rows.

Formal verification: FormalStatus(NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full normalization, unrestricted closed beta-conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window exactness, checker accountability, frontier refusal, compiler-layer provenance, redex-frontier handoff, TasteGate nontriviality, and field faithfulness for BEDC.Derived.DecidableBetaFrontierUp.

Constructive story: A DecidableBetaFrontier$^{\uparrow}$ packet is generated from a MetaCIC audit source, LambdaCalc syntax row, bounded beta window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, component-wise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2482

A Concrete Naming Certificate for $\mathsf{HardyInequality}^{\uparrow}$

Following Hardy, Littlewood, and Polya on Hardy's classical inequality, the $\mathsf{HardyInequality}^{\uparrow}$ packet records the finite weighted-average surface needed by BEDC real-series consumers. It sits between the $\mathsf{RealSeries}^{\uparrow}$ summation interface of chapter 1581, the positive rational weight surface of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names Hardy-style partial-sum control as finite RealSeries data; it does not import host analytic inequalities, choose infinite sums, assume monotone convergence, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2482.1 (Hardy inequality carrier)**.** *A* $\mathsf{HardyInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, W, S, Q, D, U, E, T, J, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row. It stores the finite real-series prefix whose Hardy averages are inspected.*
2. *W is the* $\mathsf{Rat}^{\uparrow}$ *positive-weight row. Each inspected denominator and coefficient is displayed as a rational-positive entry, not as a host real scalar.*
3. *S is the partial-sum ledger. Its rows name the inspected prefix sums and weighted averages over the same finite windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected averages.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every comparison between a weighted average and its controlling series prefix.*
6. *U is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule. It records which finite prefix, denominator, and tolerance request are open at each read.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the RealSeries source, rational weights, partial sums, readback, dyadic tolerance, and window schedule have all been displayed.*
8. *T records componentwise* hsame *transport for A, W, S, Q, D, U, E.*
9. *J records displayed* Cont *replay from a Hardy comparison request through the window schedule, partial-sum ledger, readback row, dyadic tolerance ledger, and Real seal.*

10. *K is the* Pkg *provenance over* $\mathsf{HardyInequality}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *P records the finite boundary predicate saying that each displayed average reads only entries already present in the source prefix and positive-weight row.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an infinite summation operator, a choice of limiting representative, a host analytic norm, countable choice, quotient equality of streams, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.HardyInequalityUp.TasteGate.HardyInequalityTasteGate single carrier alignment`

Theorem 2482.2 (Hardy inequality NameCert obligations)**.** *For every accepted* $\mathsf{HardyInequality}^{\uparrow}$ *carrier* $H = (A, W, S, Q, D, U, E, T, J, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure through A, positive* $\mathsf{Rat}^{\uparrow}$ *weight exposure through W, finite partial-sum exposure through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through D,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through U, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through T, displayed replay through J, package provenance through K, boundary-predicate exposure through P, and local naming through N.*

These obligations bind the finite Hardy route

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

without adding host analytic inequalities, infinite sums, selected limits, countable choice, quotient stream equality, or ambient real completeness.

Proof. Project the carrier of Definition 3435.1. Every accepted read begins with the displayed RealSeries source row and the positive rational weight row, so the weighted averages are finite BEDC rows rather than host-level analytic expressions.

The partial-sum ledger and window schedule are read next. The row S lists only prefix sums and weighted averages whose indices are opened by U, and the boundary row P requires each such average to use entries already present in A and weights already present in W.

Readback and tolerance data then enter through Q and D. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular rational approximants for the inspected averages, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the finite tolerance ledger for each comparison.

Finally E, T, J, K, N seal, transport, replay, source, and name the same ordered route. Since the excluded objects are absent from the carrier, no additional Hardy obligation can be exported by the local NameCert surface. □

Theorem 2482.3 (Hardy inequality partial-sum boundary)**.** *Let $H = (A, W, S, Q, D, U, E, T, J, K, P, N)$ be an accepted* $\mathsf{HardyInequality}^{\uparrow}$ *packet. Every Hardy average or partial-sum comparison exported by H factors through a displayed* $\mathsf{StreamName}^{\uparrow}$ *window in U, a finite* $\mathsf{RealSeries}^{\uparrow}$ *prefix in A, a positive* $\mathsf{Rat}^{\uparrow}$ *weight row in W, the partial-sum ledger S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before any* $\mathsf{Real}^{\uparrow}$ *seal E is available downstream.*

Consequently the packet records Hardy's classical inequality only as finite weighted-prefix control. It does not assert convergence of the whole series, import an external L^p inequality, choose a limiting representative, quotient streams, invoke countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the window row U. A Hardy request is accepted only after it names a finite prefix, denominator, and tolerance request, so no read can inspect a term outside the displayed window.

The source and weight rows supply the numeric data for that request. The $\mathsf{RealSeries}^{\uparrow}$ row A gives the inspected prefix, while the $\mathsf{Rat}^{\uparrow}$ row W gives the positive rational weights used by the displayed average.

The partial-sum ledger S and readback row Q then expose the finite average as regular rational data. The dyadic ledger D is the only tolerance surface available for comparing that readback with the controlling prefix.

The Real seal E is downstream from all of these rows, and T, J, K, N only transport, replay, source, and name the same route. Thus every exported comparison is bounded by the finite prefix data, and the excluded analytic and completeness principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{HardyInequality}^{\uparrow}$ read exposes a $\mathsf{RealSeries}^{\uparrow}$ prefix, positive $\mathsf{Rat}^{\uparrow}$ weights, a finite partial-sum ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, displayed continuation replay, package provenance, boundary-predicate exposure, and local NameCert before any Hardy comparison is exported.

Independence witness: realseries, rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{HardyInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a weighted partial-sum ledger and a boundary predicate tying Hardy averages to finite RealSeries prefixes and positive rational weights.

Closure status for $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{HardyInequality}^{\uparrow}$ packet is generated from a RealSeries source prefix, positive Rat weights, a finite partial-sum ledger, RegSeqRat readback, a DyadicRatCore tolerance ledger, StreamName windows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, a finite boundary predicate, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3435.1, Definition 3435.2, and Definition 3435.3 seed the finite Hardy weighted-prefix route over displayed $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic Hardy inequalities, convergence of infinite sums, selected limits, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, positive-weight exposure, partial-sum boundary exposure, readback handoff, tolerance ledger nonescape, Real seal handoff, and transport stability for BEDC.Derived.HardyInequalityUp.

Chapter 2483

A Concrete Naming Certificate for $\mathsf{CommutativeCStar}^{\uparrow}$

Following Gelfand–Naimark and the Gelfand duality theorem, the $\mathsf{CommutativeCStar}^{\uparrow}$ packet records the finite algebraic source boundary by which a unital commutative C*-algebra can be handed to $\mathsf{GelfandDuality}^{\uparrow}$ before any completed spectrum, maximal ideal space, or host Gelfand transform is public. It refines the root $\mathsf{CStarAlg}^{\uparrow}$ surface of chapter 211 by adding unit and commutativity rows over the same $\mathsf{Banach}^{\uparrow}$ and $\mathsf{Ring}^{\uparrow}$ dependency rows, and it keeps the spectrum route through $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as displayed downstream rows rather than as an imported duality theorem.

Definition 2483.1 (Commutative CStar carrier)**.** *A* $\mathsf{CommutativeCStar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, B, R, I, M, E, Q, L, X, T, Y, H, C, P, N)$$

with the following displayed rows:

1. *A is the root* $\mathsf{CStarAlg}^{\uparrow}$ *row, carrying the Banach source, RingUp multiplication source, involution row, norm-square row, transport, replay, provenance, and local naming already required by Definition 211.3.*
2. *B is the consumed* $\mathsf{Banach}^{\uparrow}$ *row read from the same source history as A.*
3. *R is the consumed* $\mathsf{Ring}^{\uparrow}$ *row read from the same source history as A.*
4. *I is the displayed unit row for the multiplication surface of R.*
5. *M is the commutativity classifier row, recording that every displayed multiplication comparison used by the packet is symmetric over the same finite carrier.*
6. *E is the involution compatibility row, requiring the displayed involution of A to preserve the unit and commute with the displayed multiplication rows.*
7. *Q is the norm-square row, preserving the C^* identity $\|a^*a\| = \|a\|^2$ on the displayed algebra rows.*
8. *L is the character-facing ledger. It lists the finite algebraic rows that a later character or evaluation consumer may inspect, but it does not supply characters, a spectrum, or a transform.*
9. *X is the spectrum-facing* $\mathsf{GelfandDuality}^{\uparrow}$ *request row, recording that characters and evaluations must be supplied by the downstream spectrum packet.*

10. *T is the* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *boundary row for the displayed spectrum carrier.*
11. *Y is the* $\mathsf{Real}^{\uparrow}$ *scalar-readback row through which norm and evaluation values are sealed before topology-facing reads consume them.*
12. *H records componentwise* hsame *transport for $A, B, R, I, M, E, Q, L, X, T, Y$ and the structural rows.*
13. *C records displayed* Cont *replay of the source route*

$$\mathsf{CStarAlg}^{\uparrow}, \mathsf{Banach}^{\uparrow}, \mathsf{Ring}^{\uparrow} \longmapsto I, M, E, Q, L \longmapsto X, T, Y \longmapsto \mathsf{CommutativeCStar}^{\uparrow}.$$

14. *P is the* Pkg *provenance over* $\mathsf{CommutativeCStar}^{\uparrow}$, $\mathsf{CStarAlg}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through the root C-algebra row, the shared Banach and RingUp dependency rows, the unit row, the commutativity classifier, involution compatibility, norm-square compatibility, the character-facing ledger, the downstream spectrum request, the compact-topology boundary, the Real scalar seal, transport, replay, provenance, and local naming. It has no coordinate for a completed spectrum, maximal ideal space, host Gelfand transform, host algebra equality, quotient completion, selected character, or choice.*

Lean4 ✓ `BEDC.Derived.CommutativeCStarUp.TasteGate.CommutativeCStarTasteGate single carrier alignment`

Theorem 2483.2 (Commutative CStar NameCert obligations)**.** *For every accepted* $\mathsf{CommutativeCStar}^{\uparrow}$ *carrier*

$$U = (A, B, R, I, M, E, Q, L, X, T, Y, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$ *surface consists exactly of carrier admission, root CStarAlg source compatibility, Banach and RingUp source agreement, unit visibility, commutativity classifier stability, involution compatibility, norm-square compatibility, character-facing ledger visibility, downstream GelfandDuality request visibility, Compact and Topology boundary visibility, Real scalar sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

A downstream consumer may inspect these algebraic rows as the unital commutative C-source boundary. It receives no completed spectrum, no host Gelfand transform, no maximal ideal space, no host equality of algebras, no quotient completion, no selected character, and no choice principle.*

Proof. Project the carrier of Definition 3436.1. The projection exposes the root CStarAlg row A, the shared Banach and RingUp rows B, R, the unit and commutativity rows I, M, the involution and norm-square rows E, Q, the character-facing ledger L, the spectrum request X, the compact-topology boundary T, the Real scalar seal Y, and the structural rows H, C, P, N.

Source compatibility is inherited from the root certificate of Definition 211.3 and the Banach–Ring boundary of Definition 211.7. The present packet adds only the displayed unit and commutativity rows over that same finite source.

Stability is componentwise. Transport through H moves the root C*-algebra row, unit row, commutativity classifier, involution compatibility, norm-square row, ledger entries, spectrum request, compact-topology boundary, and Real scalar seal together; replay through C returns the same displayed algebraic source before any character-facing read is public.

Finally P and N name only this finite surface. Since a completed spectrum, host Gelfand transform, maximal ideal space, host equality, quotient completion, selected characters, and choice have no coordinate in the carrier, they cannot be recovered from the local NameCert surface. □

Theorem 2483.3 (Commutative CStar Gelfand spectrum boundary)**.** *Every* $\mathsf{GelfandDuality}^{\uparrow}$ *algebra-source or spectrum-facing read of an accepted* $\mathsf{CommutativeCStar}^{\uparrow}$ *packet factors through the same root CStarAlg row A, shared* $\mathsf{Banach}^{\uparrow}$ *row B, shared* $\mathsf{Ring}^{\uparrow}$ *row R, unit row I, commutativity classifier M, involution row E, norm-square row Q, character-facing ledger L, spectrum request row X, compact-topology boundary row T, Real scalar seal row Y, and structural rows H, C, P, N.*

Consequently $\mathsf{GelfandDuality}^{\uparrow}$ *can cite* $\mathsf{CommutativeCStar}^{\uparrow}$ *as its unital commutative C*-source boundary while the spectrum side remains a downstream* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *read. The packet does not treat the root* $\mathsf{CStarAlg}^{\uparrow}$ *chapter as already commutative and does not import a completed spectrum, host transform, maximal ideal space, quotient, host equality, selected character, or choice data.*

Proof. Begin with a $\mathsf{GelfandDuality}^{\uparrow}$ algebra-source read. The root C*-algebra dependency is the row A, and Definition 211.5 requires unit and commutativity rows to be displayed before that dependency can be consumed as a unital commutative source.

The rows I and M supply exactly that added algebraic surface. The rows E and Q keep the involution and norm-square laws attached to the same Banach and RingUp dependencies B, R already carried by A.

The character-facing ledger L is only a ledger of available algebraic rows. The downstream row X records that character and evaluation data belong to the $\mathsf{GelfandDuality}^{\uparrow}$ packet, while T and Y record the compact-topological spectrum boundary and Real scalar seal that such a packet may consume.

Transport, replay, provenance, and naming are then handled by H, C, P, N. Since these structural rows preserve only the displayed algebraic packet and its downstream boundary rows, the $\mathsf{GelfandDuality}^{\uparrow}$ source read remains the finite $\mathsf{CommutativeCStar}^{\uparrow}$ boundary and cannot collapse into host-level duality data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CommutativeCStar}^{\uparrow}$ packet is generated from a root CStarAlgUp row, shared BanachUp and RingUp dependency rows, a unit row, a commutativity classifier, involution compatibility, norm-square compatibility, a character-facing algebraic ledger, downstream GelfandDuality request visibility, Compact and Topology boundary rows, a Real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3436.1, Definition 3436.2, and Definition 3436.3 seed the finite unital commutative C*-source boundary over CStarAlgUp, BanachUp, RingUp, GelfandDualityUp, CompactUp, TopologyUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed spectrum, maximal ideal space, host Gelfand transform, host algebra equality, quotient completion, selected character, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, unit visibility, commutativity stability, involution and norm-square compatibility, character-ledger visibility, compact-topology boundary routing, Real scalar sealing, and Gelfand spectrum non-escape for BEDC.Derived.CommutativeCStarUp.

Chapter 2484

A Concrete Naming Certificate for MetaCICCriticalPairLedger$^{\uparrow}$

The MetaCICCriticalPairLedger$^{\uparrow}$ packet realizes a finite audit-map ledger for MetaCIC critical pairs. It reads the confluence critical-pair boundary of chapter 2212, the frontier critical-pair route of chapter 2382, the frontier dependency matrix of chapter 2307, the bounded normal-equality checker of chapter 752, and the subject-reduction route classifier of chapter 785. The packet stores critical-pair entries as a ledger of finite rows rather than as a global confluence theorem.

Falsifiable prediction: Every accepted MetaCICCriticalPairLedger$^{\uparrow}$ packet exposes a finite critical-pair table, local join ledger, bounded-checker handoff, route-separation row, audit-map replay row, provenance, and local naming before any confluence-facing consumer can read it.

Independence witness: metacic-confluence-critical-pair-boundary, metacic-frontier-critical-pair, metacic-frontier-dependency-matrix, bounded-normal-equality-checker, subject-reduction-route-classifier: these siblings respectively expose common-reduct boundary rows, frontier local-join rows, dependency accounting, bounded normal comparison, and typing-route separation; none is bijective with the packet that jointly records the ledger index, entry table, local joins, checker handoff, route refusal, audit replay, provenance, and local naming.

Definition 2484.1 (MetaCIC critical-pair ledger carrier)**.** *A* MetaCICCriticalPairLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ *source row.*
2. *F is the* MetaCICFrontierCriticalPair$^{\uparrow}$ *frontier route row.*
3. *D is the* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *dependency-accounting row.*
4. *K is the ledger key row, indexing the finite critical-pair entries accepted by the packet.*
5. *J is the local-join ledger row, assigning displayed join candidates to the accepted ledger entries.*
6. *E is the* BoundedNormalEqualityChecker$^{\uparrow}$ *handoff row used only after K and J have been displayed.*
7. *S is the* SubjectReductionRouteClassifier$^{\uparrow}$ *separation row, keeping typing discharge outside the critical-pair ledger.*

8. *A is the audit-map replay row, recording the finite readback path from an audit request through B, F, D, K, J, E, S.*
9. *H records componentwise* hsame *transport for all source, ledger, handoff, route-separation, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the ledger route.*
11. *P is the* Pkg *provenance over* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$, $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, SubjectReductionRouteClassifier$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, or an audit-map read outside the finite ledger.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPairLedgerUp.MetaCICCriticalPairLedgerTasteGate single carrier alignment`

Theorem 2484.2 (MetaCIC critical-pair ledger NameCert obligations)**.** *For every accepted* MetaCICCriticalPairLed *carrier*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$ *surface consists exactly of carrier admission, critical-pair boundary compatibility, frontier route compatibility, dependency-accounting visibility, finite ledger-key display, local-join ledger display, bounded-checker handoff, subject-reduction route separation, audit-map replay, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

No obligation is stored outside those rows, and no consumer receives global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, or a hidden audit-map credential.

Proof. Project the carrier of Definition 3437.1. The projection exposes exactly $B, F, D, K, J, E, S, A, H, C, P, N$.

Boundary and frontier compatibility are read first from B and F. The dependency-accounting row D then records which frontier and checker dependencies are visible before any ledger entry can be consumed.

The ledger rows K and J are finite. A bounded-checker handoff through E is available only after the entry key and local join have been displayed, and the route-separation row S keeps subject-reduction discharge outside the ledger.

The audit replay A and structural rows H, C, P, N transport, replay, source, and name only the displayed ledger route. Since the excluded confluence and typing claims have no carrier coordinate, the NameCert surface is exhausted by the stated rows. □

Theorem 2484.3 (MetaCIC critical-pair ledger audit route)**.** *Every MetaCIC confluence audit-map read of an accepted* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *packet factors through the finite route*

$$B, F, D \longmapsto K, J \longmapsto E, S \longmapsto A, H, C, P, N.$$

Thus the packet concretizes the audit-map critical-pair surface as a ledger of displayed rows, not as a proof of global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or subject-reduction discharge.

Proof. An audit-map consumer first reads the critical-pair boundary B, frontier route F, and dependency-accounting row D. These rows determine which finite critical-pair entries are in scope.

The ledger key K and local-join ledger J are then read together. The bounded normal-equality checker E receives only these displayed entries and join candidates.

The route-separation row S remains visible before the audit replay A and structural rows package the read. Therefore the consumer receives a finite ledger route, while the excluded global confluence and subject-reduction claims remain outside the packet. □

Theorem 2484.4 (MetaCIC critical-pair ledger local-join induction)**.** *For every accepted* MetaCICCriticalPairLedge *carrier*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N),$$

induction over the finite ledger-key row K preserves the route

$$B, F, D \longmapsto K, J \longmapsto E, S \longmapsto A, H, C, P, N$$

for each displayed critical-pair entry. Each induction step reads the local join from J before the bounded-checker handoff E and the subject-reduction separation row S, and no step upgrades the ledger to global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or unchecked subject-reduction discharge.

Proof. The base key is read after the source rows B, F, D. By Definition 3437.1, the first ledger entry has no join coordinate except the displayed local-join row J.

For the induction step, assume the route has been preserved for the earlier ledger keys. The next key is another finite entry of K, so the packet again exposes its join candidate through J before any bounded-checker handoff through E is available.

The separation row S remains downstream of the local join and checker rows. Applying Definition 3437.3, the audit replay can therefore continue only through A, H, C, P, N after the route $B, F, D \longmapsto K, J \longmapsto E, S$ has been displayed.

No excluded confluence principle is present in the carrier. Componentwise transport, continuation replay, provenance, and local naming preserve the finite ledger route without adding global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or unchecked subject-reduction discharge. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ packet is generated from confluence critical-pair boundary rows, frontier critical-pair rows, dependency-accounting rows, finite ledger keys, local-join rows, bounded-checker handoffs, subject-reduction route separation, audit-map replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3437.1, Definition 3437.2, Definition 3437.3, the falsifiable prediction, and the independence witness seed the finite MetaCIC critical-pair ledger over boundary, frontier, dependency, checker, route-separation, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary compatibility, frontier compatibility, dependency visibility, ledger display, checker handoff, route separation, audit replay, and nonescape for BEDC.Derived.MetaCICCriticalPairLedg

Chapter 2485

A Concrete Naming Certificate for $\mathsf{RankNullity}^{\uparrow}$

Following the standard finite-dimensional linear-algebra theorem over a field, the $\mathsf{RankNullity}^{\uparrow}$ packet records the BEDC surface at which a linear map decomposes its source ledger into kernel and image dimensions. It sits after the field carrier of chapter 45, the vector-space ledger of chapter 47, the linear-map packet of chapter 48, and the matrix presentation of chapter 49. The chapter names the rank-nullity certificate as finite NameCert data; it does not add basis choice, quotient-vector-space construction, or spectral structure.

Definition 2485.1 (Rank-nullity carrier)**.** *A* $\mathsf{RankNullity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, S, T, K, I, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LinMap}^{\uparrow}$ *code row, recording the displayed map whose kernel and image ledgers are being read.*
2. *S is the finite* $\mathsf{VecSpace}^{\uparrow}$ *source ledger over the displayed* $\mathsf{Field}^{\uparrow}$ *row, including its finite dimension token.*
3. *T is the finite* $\mathsf{VecSpace}^{\uparrow}$ *target ledger over the same field row, including the codomain dimension context used by the image row.*
4. *K is the kernel row, exposing exactly the source vectors whose L-readback reaches the zero row of T.*
5. *I is the image row, exposing exactly the target vectors reached by an L-readback from an accepted source row.*
6. *A is the rank row, naming the finite dimension token attached to I.*
7. *U is the nullity row, naming the finite dimension token attached to K.*
8. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same map and the same finite source and target ledgers.*
9. *H stores componentwise* hsame *transport for the map, source, target, kernel, image, rank, nullity, matrix presentation, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a rank-nullity request through $L, S, T, K, I, A, U, M, H$ and back to the local certificate row.*

11. *P is the* Pkg *provenance over* $\mathsf{RankNullity}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary non-singleton function extensionality, quotient dimensions, basis selection, spectral theory, Lean verification, or bridge checking.

Theorem 2485.2 (Rank-nullity NameCert obligations)**.** *For every accepted* $\mathsf{RankNullity}^{\uparrow}$ *carrier*

$$R = (L, S, T, K, I, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the linear-map code, finite source ledger, finite target ledger, kernel row, image row, rank row, nullity row, matrix presentation, transport, replay, provenance, and local naming;*
2. *map-source-target coherence, requiring L to be read only against the finite* $\mathsf{VecSpace}^{\uparrow}$ *ledgers S, T over the displayed* $\mathsf{Field}^{\uparrow}$ *row;*
3. *kernel exactness, requiring K to contain precisely the source rows mapped by L to the zero row of T;*
4. *image exactness, requiring I to contain precisely the target rows reached by L from an admitted source row in S;*
5. *finite dimension accounting, requiring A and U to read the displayed dimension tokens of I and K and to satisfy the source-dimension ledger encoded by the packet;*
6. *matrix compatibility, requiring M to present the same L, S, T route without adding an external basis-choice row;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rank-nullity rows.*

No rank-nullity obligation is stored outside R.

Proof. Project the carrier of Definition 3438.1. The tuple lists the linear-map code, source and target vector-space ledgers, kernel, image, rank, nullity, matrix presentation, and the structural rows.

The coherence, kernel, image, and matrix obligations are exactly the rows L, S, T, K, I, M. The source and target ledgers supply the finite vector-space context; the kernel and image rows are read only through the displayed map; and the matrix row is a presentation of the same route rather than an additional choice principle.

The dimension obligation is stored in A, U together with the finite source ledger S. Since H, C, P, N only transport, replay, source, and name rows already present in the carrier, there is no remaining location for a quotient dimension, basis choice, spectral datum, or external function-extensionality witness. □

Lean4 ✓ `BEDC.Derived.RankNullityUp.RankNullityTasteGate single carrier alignment`

Theorem 2485.3 (Rank-nullity kernel-image ledger)**.** *Every accepted* $\mathsf{RankNullity}^{\uparrow}$ *packet factors a finite-dimensional linear-map read through the displayed route*

$$\mathsf{LinMap}^{\uparrow} \longrightarrow \mathsf{VecSpace}^{\uparrow}_{\mathrm{src}} \times \mathsf{VecSpace}^{\uparrow}_{\mathrm{tgt}} \longrightarrow (\ker, \mathrm{im}) \longrightarrow (\mathrm{rank}, \mathrm{nullity}) \longrightarrow \mathsf{Mat}^{\uparrow}.$$

Along this route the rank row A and nullity row U account for the finite source-dimension token of S by the kernel-image ledger, and every exported rank-nullity read preserves the same rows $L, S, T, K, I, A, U, M, H, C, P, N$.

Proof. Start with an accepted carrier $R = (L, S, T, K, I, A, U, M, H, C, P, N)$. Projection exposes the map code, source and target ledgers, kernel and image rows, rank and nullity rows, matrix presentation, and structural rows.

The rows K and I are the only kernel and image coordinates in the packet. Thus any consumer reading a kernel-image decomposition must use the L-route from S to T rather than a detached quotient or ambient function graph.

The rows A and U then record the finite dimension accounting for I and K. The matrix row M presents the same map over the same finite ledgers, while H, C, P, N preserve the row order under transport, replay, provenance, and local naming.

Therefore the exported read is exactly the displayed kernel-image ledger and cannot export arbitrary non-singleton function extensionality, quotient dimensions, basis selection, spectral theory, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3438.1, Definition 3438.2, and Definition 3438.3 seed the finite rank-nullity carrier over LinearMap, VecSpace, Matrix, and Field rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary non-singleton function extensionality, quotient dimensions, choice of bases, spectral theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, map-source-target coherence, kernel exactness, image exactness, finite dimension accounting, matrix compatibility, transport stability, and replay stability for BEDC.Derived.RankNullityUp.

Constructive story: A $\mathsf{RankNullity}^{\uparrow}$ packet is generated from a LinearMap code row, finite source and target VecSpace ledgers, kernel and image rows, rank and nullity rows, a Matrix presentation row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2486

A Concrete Naming Certificate for $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$

The $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ packet realizes the endpoint bridge isolated by the MetaCIC decidability frontier in chapter 9386. It sits between the bounded comparison boundary of chapter 749, the candidate-mediated normalisation frontier of chapter 892, and the closed-normal endpoint seal of chapter 740. The packet records how a bounded normal-output comparison may be consumed only after finished-normal evidence is displayed; it does not claim unrestricted closed beta-conversion decidability.

Definition 2486.1 (Bounded normal conversion bridge carrier)**.** *A* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, O, E, S, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *row, exposing the bounded normal-equality checker and its refusal boundary.*
2. *F is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row, exposing the candidate-mediated route by which normal computation is allowed to reach a finished endpoint.*
3. *O is the bounded normal-output row, naming the finite fuel, closedness window, and two normalizer outputs inspected by the checker.*
4. *E is the finished-normal evidence row, requiring the compared outputs to be closed normal forms before conversion evidence is exported.*
5. *S is the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *endpoint row, recording that the finished-normal endpoint can be consumed only at the normal-form boundary.*
6. *R is the refusal row, recording that a read bypassing either O or E is not a conversion bridge read.*
7. *H stores componentwise* hsame *transport for the decidability boundary, normalisation frontier, bounded output, finished evidence, normal-form seal, refusal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a conversion request through D, F, O, E, S, R, H and back to the local certificate row.*

9. *P is the* Pkg *provenance over* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global conversion decider, fuel-erased normalization, quotient equality over represented terms, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.BoundedNormalConversionBridgeUp.BoundedNormalConversionBridgeUp`

Theorem 2486.2 (Bounded normal conversion bridge NameCert obligations)**.** *For every accepted* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *carrier*

$$B = (D, F, O, E, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the decidability boundary, normalisation frontier, bounded normal-output row, finished-normal evidence row, endpoint seal, refusal row, transport, replay, provenance, and local naming;*
2. *bounded-output visibility, requiring every conversion read to cite the finite output row O before checker equality is consumed;*
3. *finished-normal endpoint visibility, requiring E to certify that the compared outputs are finished closed normal forms before the normal-form seal S is read;*
4. *frontier compatibility, requiring D and F to be read together so bounded decidability is not separated from the normalisation route that supplies its endpoint evidence;*
5. *refusal stability, requiring R to reject any read that bypasses bounded normal output or finished-normal evidence;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rows.*

No bounded normal conversion bridge obligation is stored outside B.

Proof. Project the carrier of Definition 3439.1. The tuple exposes the decidability boundary, normalisation frontier, bounded output, finished evidence, normal-form seal, refusal, and structural rows.

The bounded-output and finished-endpoint obligations are the rows O and E. A conversion request first fixes the finite normalizer outputs and then supplies evidence that those outputs are finished closed normal forms.

The frontier compatibility obligation is stored in D, F, S. The bounded checker is read through the decidability boundary, the endpoint evidence is routed through the normalisation frontier, and the normal-form seal records the finished endpoint without creating a global conversion decision procedure.

The refusal and structural rows R, H, C, P, N preserve that order. Since the carrier contains no coordinate for fuel-erased normalization, quotient equality, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 2486.3 (Bounded normal conversion endpoint handoff)**.** *Every accepted* BoundedNormalConversionBridge *conversion read factors through the finite route*

$$\mathsf{MetaCICDecidableBoundary}^{\uparrow} \longrightarrow \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longrightarrow \mathsf{NormalFormConsistencySeal}^{\uparrow}$$

with bounded normal outputs and finished closed-normal evidence displayed before any conversion conclusion is exported.

Proof. Let $B = (D, F, O, E, S, R, H, C, P, N)$ be an accepted carrier. Projection exposes $D, F, O, E, S, R, H, C, P, N$ as the only rows available to a conversion read.

The row O fixes the bounded normalizer outputs inspected by the decidability boundary D. The row E then supplies the finished-normal evidence required before those outputs can be consumed as closed normal forms.

The row F supplies the normalisation-frontier route, and S places the endpoint at the normal-form consistency seal. The refusal row R rejects any path that omits O or E, while H, C, P, N preserve the same row order under transport, replay, provenance, and local naming.

Thus the packet supplies an endpoint handoff from bounded normal comparison to finished normal forms, not a global conversion decider or a fuel-erased normalization theorem. □

Falsifiable prediction: Any conversion read that bypasses either bounded normal output or finished-normal evidence refutes the $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ carrier.

Independence witness: metacicdecidableboundary, metacicnormalizationfrontier, normalformconsistencyseal, subjectreductiondischargeledger, metacicconfluenceauditpacket: the bridge carrier is not bijective to any listed sibling because it binds bounded normal-output comparison, finished-normal evidence, endpoint sealing, refusal, transport, replay, provenance, and local naming in one conversion-facing packet.

The Lean TasteGate follow-up for this chapter must provide explicit $\mathsf{Nontrivial\ BoundedNormalConversionBridge}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundedNormalConversionBridge}^{\uparrow}$ instances preserving $D, F, O, E, S, R, H, C, P, N$ as separate displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3439.1, Definition 3439.2, and Definition 3439.3 seed the bounded normal conversion bridge from decidability rows to finished normal-form endpoint evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted closed beta-conversion decidability, fuel-erased normalization, quotient equality over represented terms, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-output visibility, finished-normal endpoint visibility, frontier compatibility, refusal stability, TasteGate nontriviality, and field faithfulness for BEDC.Derived.BoundedNormalConversio

Constructive story: A $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ packet is generated from MetaCICDecidableBoundary, MetaCICNormalizationFrontier, bounded normal-output, finished-normal evidence, NormalFormConsistencySeal, refusal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2487

A Concrete Naming Certificate for $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ packet records the finite comparison step used when two displayed dyadic or regular-rational windows must be compared before the terminal real seal is consumed. It sits over the dyadic tolerance rows of chapter 291, the finite observation windows of chapter 126, the regular-rational readback route of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the Bishop-style real comparison surface of chapter 1381. The packet names only a finite dyadic comparison surface; it is not real trichotomy, quotient-real equality, a selected limit, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 2487.1 (DyadicComparisonPrinciple carrier)**.** *A* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D_0, D_1, W, R, O, H, C, P, N)$$

with the following displayed rows:

1. *D_0 and D_1 are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows whose dyadic numerators, scales, tolerance ledgers, and descent witnesses are being compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule recording which finite observation windows of the two names are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the scheduled windows into regular rational observations without selecting quotient representatives.*
4. *O is the finite comparison outcome row. It records one displayed dyadic relation among the inspected rows, or a displayed overlap certificate when the current window is not yet separating.*
5. *H records componentwise* hsame *transports for D_0, D_1, W, R, and O.*
6. *C records the displayed* Cont *replay routes by which downstream consumers inspect the two dyadic rows, the shared window, the regular-rational readback, and the finite comparison outcome.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for global real trichotomy, quotient-real equality, a selected limit, countable choice, a host order decision procedure, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicComparisonPrincipleUp.DyadicComparisonPrincipleUpTasteGate single carrier alignment`

Theorem 2487.2 (DyadicComparisonPrinciple NameCert obligations)**.** *For every accepted* DyadicComparisonPrinc *carrier* $K = (D_0, D_1, W, R, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the two dyadic rows, shared finite window schedule, regular-rational readback, finite comparison outcome, componentwise transport, displayed continuation replay, provenance, and local naming rows.*

Every $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{BishopRealComparison}^{\uparrow}$*-facing consumer of this packet must therefore pass through the displayed route*

$$D_0, D_1 \longmapsto W \longmapsto R \longmapsto O \longmapsto H, C, P, N$$

before it can read a dyadic comparison. The route exports no real trichotomy, no quotient equality of real names, no selected limit, no countable-choice schedule, and no ambient completeness coordinate.

Proof. Project the carrier of Definition 3440.1. The analytic rows are exactly the two dyadic endpoints, the shared finite window schedule, the regular-rational readback, and the finite outcome row.

Read the comparison in route order. The dyadic endpoint rows supply the tolerance ledgers, the window schedule names the finite observations, and the regular-rational row supplies the readback surface on which the outcome row is meaningful.

The transport and replay rows preserve those displayed coordinates componentwise. Provenance and local naming then source the same finite packet for downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{BishopRealComparison}^{\uparrow}$ consumers.

The excluded principles are absent from the carrier. Since no coordinate stores quotient equality, trichotomy, a selected limit, countable choice, a host decision procedure, or ambient completeness, none of them can be exported by the local certificate. □

Theorem 2487.3 (DyadicComparisonPrinciple trichotomy refusal)**.** *An accepted* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ *packet exports finite dyadic comparison data only through* $D_0, D_1, W, R, O, H, C, P, N$*. In particular, the outcome row* O *may name a displayed separating relation or an overlap certificate for the inspected window, but it does not decide a global trichotomy for arbitrary* $\mathsf{Real}^{\uparrow}$ *names, identify quotient real equality, choose a limit representative, provide countable choice, or close ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the finite outcome row O. Its inputs are the two dyadic rows and the shared finite window route named by W and R, so it is a statement about the inspected window rather than about all real-name extensions.

Use Definition 3440.2. Every consumer read of the outcome must pass through the displayed carrier rows before reaching any $\mathsf{Real}^{\uparrow}$ or $\mathsf{BishopRealComparison}^{\uparrow}$-facing surface.

The packet therefore separates finite dyadic comparison from global order structure. Quotient real equality, selected limits, countable choice, host trichotomy, and real completeness have no row in the carrier and cannot be recovered from the provenance or local naming data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ packet is generated from two displayed DyadicRatCoreUp rows, a shared StreamName finite-window schedule, a RegSeqRat readback row, a finite comparison outcome or overlap certificate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DyadicComparisonPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 3440.1, Definition 3440.2, and Definition 3440.3 define the finite dyadic comparison packet and its trichotomy refusal boundary.

Formal verification: FormalStatus(NameCert$_{\text{DyadicComparisonPrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, global real trichotomy, quotient-real equality, selected limits, countable choice, host order decision procedures, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite outcome transport, Real-facing handoff, BishopRealComparison-facing handoff, and non-escape theorems over BEDC.Derived.DyadicComparisonPrincipleUp.

Chapter 2488

A Concrete Naming Certificate for $\mathsf{CauchyQuotientCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, this packet records the finite quotient-read criterion for regular Cauchy real names. It sits over the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. Its algebra-facing dependencies are the displayed product carrier of chapter 461 and the synchronized sufficiency criterion of chapter 1586; the packet names quotient reads only through these finite rows and never through host division of completed real objects.

Definition 2488.1 (Cauchy quotient criterion carrier)**.** *A* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, Z, W, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A is the numerator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose quotient read is requested.*
2. *B is the denominator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, accompanied only by displayed finite nonzero-window evidence in the dyadic ledger.*
3. *Z is the proposed quotient* $\mathsf{RegSeqRat}^{\uparrow}$ *row, read as a candidate regular rational name rather than as a selected representative of a quotient stream.*
4. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for A, B, Z, and the product check.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the requested quotient tolerance, the finite denominator-separation budget, and the residual comparison budget for $B \cdot Z$ against A.*
6. *P is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route for the displayed product of B and Z.*
7. *K is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row that accepts the product route before any regular-rational or real-facing quotient read is exposed.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the accepted quotient candidate after the product residual has been checked.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, P, K, R have been displayed.*
10. *H records componentwise* hsame *transport for the source, candidate, window, dyadic, product, criterion, handoff, seal, continuation, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a quotient tolerance request through W, D, P, K, R, E and then through the structural rows.*
12. *L is the* Pkg *provenance over* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, Z, W, D, P, K, R, E, H, C, L, N$. It has no coordinate for quotient streams, selected representatives, host real division, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 2488.2 (Cauchy quotient criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier $Q = (A, B, Z, W, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the numerator, denominator, candidate quotient, synchronized windows, dyadic quotient ledger, product route, product criterion, regular-rational handoff, real seal, transport, continuation, provenance, and local name; finite denominator-window coverage through W and D; product-residual exactness through P and K;* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R;* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E; and refusal of quotient streams, selected representatives, host real division, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

Proof. Carrier admission is projection from Definition 3441.1. The accepted packet displays the numerator row A, denominator row B, candidate quotient row Z, synchronized finite windows, dyadic quotient ledger, product route, criterion row, regular handoff, real seal, and the structural rows.

The denominator and residual checks are finite. The row D stores the denominator-separation budget and the product residual budget for the same windows named by W, while P and K route the displayed product $B \cdot Z$ through the regular Cauchy product surface and its criterion.

The consumer-facing rows are downstream. The handoff R is available only after the product residual has passed through W, D, P, K, and the seal row E is available only after R has been displayed. The rows H, C, L, N transport, replay, source, and name this same route.

Since the carrier has no coordinate for quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyQuotientCriterionUp.CauchyQuotientCriterionTasteGate single carrier alignment`

Theorem 2488.3 (Cauchy quotient criterion Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing quotient read of an accepted* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier factors through the displayed finite route*

$$A, B, Z \longmapsto W \longmapsto D \longmapsto P \longmapsto K \longmapsto R \longmapsto E$$

together with the stored hsame *transport H,* Cont *replay C,* Pkg *provenance L, and local* NameCert *row N. The route exposes a quotient-read criterion for regular Cauchy real names, not quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing quotient consumer. By the carrier definition, the only $\mathsf{Real}^{\uparrow}$ seal row is E, and it is downstream of the regular-rational handoff R.

The handoff R is not primitive quotient data. It is admitted only after the synchronized window row W and dyadic ledger D have displayed the denominator-separation and residual

budgets, and only after P and K have routed the product check through $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{CauchyProductCriterion}^{\uparrow}$.

The structural rows preserve the same route. Componentwise transport H, replay C, provenance L, and local naming N can only re-expose the finite packet already displayed.

Thus every real-facing quotient read reaches E through the named finite chain and cannot import quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ packet is generated from numerator, denominator, and candidate quotient RegSeqRat rows, synchronized StreamName windows, a DyadicRatCore quotient ledger, a RegularCauchyProduct route, a CauchyProductCriterion sufficiency row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3441.1, Definition 3441.2, and Definition 3441.3 seed the finite quotient-read route for regular Cauchy real names through synchronized windows, dyadic denominator and residual budgets, product checking, RegSeqRat handoff, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyQuotientCriterionUp.CauchyQuotientCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, host real division, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite denominator-window coverage, product-residual exactness, RegSeqRat handoff, Real-seal nonescape, transport stability, continuation replay, and ledger non-escape for BEDC.Derived.CauchyQuotientCriterionUp.

Chapter 2489

A Concrete Naming Certificate for $\mathsf{PicardFixedPointModulus}^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive analysis, the $\mathsf{PicardFixedPointModulus}^{\uparrow}$ packet records the finite tail modulus that a fixed-point consumer reads before any endpoint is selected. It sits after the contraction-map surface of chapter 1418, the one-step Picard route of chapter 395, the separate complete-metric consumer row of chapter 128, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a constructive modulus handoff for Picard tails; it is not a selected fixed point, not a completed orbit endpoint, not host fixed-point equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2489.1 (PicardFixedPointModulus carrier)**.** *A* $\mathsf{PicardFixedPointModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, I, L, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *source packet, including its metric carrier, self-map graph, rational contraction-bound ledger, and one-step Picard route.*
2. *I is the* $\mathsf{PicardContraction}^{\uparrow}$ *iterate ledger that records the inspected Picard approximants and adjacent contraction comparisons.*
3. *L is the strict-ratio tail ledger, listing the finite bound powers and tolerance requests used to answer tail estimates.*
4. *T is the modulus threshold row. Each entry names a displayed stage after which the Picard tail comparison is admitted for the requested tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the Picard orbit as regular rational sequence data only after K, I, L, T have been displayed.*
6. *E is the* $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal boundary reached by completion and Real-facing consumers after the regular readback route has been exposed.*
7. *H records componentwise* hsame *transport for the contraction source, Picard ledger, tail ledger, threshold row, regular readback, seal boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from contraction-map admission through Picard iteration, tail-threshold extraction, regular readback, and completion-facing seal.*

9. *P is the* Pkg *provenance over* PicardFixedPointModulus$^\uparrow$, ContractionMapping$^\uparrow$, PicardContraction$^\uparrow$, CompleteMetric$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{PicardFixedPointModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through $K, I, L, T, R, E, H, C, P, N$. It has no coordinate for endpoint selection, completed orbit equality, quotient stream equality, host function extensionality, countable choice, trichotomy, ambient Real$^\uparrow$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardFixedPointModulusUp.PicardFixedPointModulusTasteGate single carrier alignment`

Theorem 2489.2 (PicardFixedPointModulus NameCert obligations)**.** *For every accepted* PicardFixedPointModulus *carrier*

$$M = (K, I, L, T, R, E, H, C, P, N),$$

the local NameCert$_{\text{PicardFixedPointModulus}^\uparrow}$ *surface consists exactly of carrier admission, contraction-source visibility through K, Picard-iterate visibility through I, strict-ratio tail accounting through L, threshold extraction through T,* RegSeqRat$^\uparrow$ *readback through R, completion and Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the fixed-point modulus route without adding endpoint selection, completed orbit equality, quotient stream equality, host function extensionality, countable choice, trichotomy, or ambient Real$^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 3442.1. The displayed rows $K, I, L, T, R, E, H, C, P, N$ are the entire local surface, so carrier admission is exactly the admission of those rows.

The contraction and Picard data are read first. The row K supplies the contraction-map packet named by chapter 1418, while I supplies the Picard iterate route named by chapter 395.

The modulus row is extracted from finite tail accounting. The strict-ratio ledger L lists the bound powers and tolerance requests, and T records the displayed threshold entries consumed by a fixed-point modulus request.

Only then can regular readback and completion-facing seals be consumed. The row R exposes the RegSeqRat$^\uparrow$ route, while E marks the later CompleteMetric$^\uparrow$ and Real$^\uparrow$ boundary. Since endpoint selection and ambient completeness are not carrier coordinates, the local NameCert obligations stop at this displayed modulus handoff. □

Theorem 2489.3 (PicardFixedPointModulus tail handoff)**.** *Every fixed-point or completion-facing consumer of an accepted* PicardFixedPointModulus$^\uparrow$ *packet must read*

$$\text{ContractionMapping}^\uparrow,\ \text{PicardContraction}^\uparrow,\ L,\ T,\ \text{RegSeqRat}^\uparrow,\ \text{CompleteMetric}^\uparrow,\ \text{Real}^\uparrow$$

in the order displayed by Definition 3442.1. The handoff supplies a finite Picard-tail modulus for downstream fixed-point consumers; it does not select a fixed point, identify a completed orbit endpoint, assert host fixed-point equality, or import ambient completeness.

Proof. Begin with the contraction source and Picard ledger. A consumer first reads K and I, so each iterate and adjacent contraction comparison remains anchored in the same displayed contraction packet.

Next read the finite tail accounting. The rows L and T expose the strict-ratio budget and the requested threshold; no infinite tail selector or completed endpoint is available at this stage.

The regular rational and completion-facing rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R receives only the displayed modulus route, and the $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Real}^{\uparrow}$ seal boundary E is visible only after that route has been replayed.

Thus the packet supplies a fixed-point modulus handoff rather than an endpoint selection principle. The omitted host-side principles have no carrier row and cannot be exported by H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{PicardFixedPointModulus}^{\uparrow}$ packet is generated from a ContractionMapping source packet, a PicardContraction iterate ledger, a strict-ratio tail ledger, a displayed threshold row, a RegSeqRat readback, a CompleteMetric/Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3442.1, Definition 3442.2, and Definition 3442.3 seed the finite Picard tail-modulus handoff from contraction and Picard data to regular readback and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim endpoint selection, completed orbit equality, quotient stream equality, host fixed-point equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-source visibility, Picard-iterate visibility, strict-ratio tail accounting, threshold extraction, regular readback, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PicardFixedPointModulusUp.

Chapter 2490

A Concrete Naming Certificate for $\mathsf{RectifiableCurve}^{\uparrow}$

Following Burago–Burago–Ivanov metric geometry and Bishop–Bridges constructive analysis, the $\mathsf{RectifiableCurve}^{\uparrow}$ packet records rectifiability by finite partition-length ledgers over a displayed metric path. It consumes the metric carrier of chapter 57, the continuous-path graph surface of chapter 130, the terminal real-name seal of chapter 38, the dyadic length rows of chapter 291, and the finite variation boundary of chapter 398. The packet names only the partition-level length certificate; it is not a completed path space, not an arclength supremum, and not quotient equality of curves.

Definition 2490.1 (Rectifiable curve carrier)**.** *A* $\mathsf{RectifiableCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, d, \gamma, I, \Pi, L, R, H, C, P, N)$$

with the following displayed rows:

1. X, d *is the accepted* $\mathsf{Met}^{\uparrow}$ *carrier and distance row.*
2. γ *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for a path from the interval row* I *into* X*.*
3. I *is the finite interval row supplying the ordered endpoint and subdivision domain.*
4. Π *is the ordered finite partition ledger over* I*, including adjacency and refinement witnesses.*
5. L *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-length sum row obtained by reading* $d(\gamma(t_i), \gamma(t_{i+1}))$ *on the displayed partition edges.*
6. R *is the common-refinement row comparing two displayed partition ledgers and their induced dyadic length sums.*
7. H *records componentwise* hsame *transport of the metric carrier, path graph, interval row, partition ledger, dyadic length sum, common-refinement row, provenance, and local naming row.*
8. C *records displayed* Cont *replay for metric reads, path evaluation, interval subdivision, edge-distance readback, length summation, and refinement comparison.*
9. P *is the* Pkg *provenance row over* $\mathsf{RectifiableCurve}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^{\uparrow}}$ *row.*

The carrier has no coordinate for a supremum over all partitions, quotient equality of paths, completed path spaces, host arclength measures, or bridge evidence. Its public object is the finite partition-length route from $X, d, \gamma, I, \Pi, L, R$ through H, C, P, N.

Lean4 ✓ `BEDC.Derived.RectifiableCurveUp.TasteGate.RectifiableCurveTasteGate single carrier alignment`

Theorem 2490.2 (Rectifiable curve NameCert obligations)**.** *For the carrier of Definition 3443.1, the local* $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$ *surface consists of exactly five obligation rows:*

1. *carrier admission for the finite packet $K = (X, d, \gamma, I, \Pi, L, R, H, C, P, N)$;*
2. *common-refinement classification for displayed partition ledgers through R;*
3. *dyadic edge-length exactness for the sum row L over the adjacent edges of Π;*
4. *componentwise* hsame *and* Cont *transport for metric reads, path graph reads, partition reads, edge-distance readbacks, and length-sum replay;*
5. *exclusion of arclength suprema, quotient path equality, completed path spaces, and bridge evidence from the local certificate.*

Every metric, variation, or path-length consumer must therefore read rectifiability through the displayed finite partition row and its dyadic length ledger before any completion-facing object can be considered.

Proof. Project the carrier. The rows $X, d, \gamma, I, \Pi, L, R$ are the only geometric coordinates, while H, C, P, N only transport, replay, source, and name that same finite packet.

Read the partition route in order. The interval row I supplies the ordered domain, the path graph γ supplies endpoint reads in the metric carrier, and the partition ledger Π lists the adjacent edge reads consumed by L.

Common refinement is finite. The row R compares two displayed partition ledgers by refinement witnesses and compares only their induced dyadic edge-length sums, so the classifier never ranges over all partitions.

The structural rows preserve this boundary. Since no carrier coordinate names an arclength supremum, quotient path equality, completed path space, or bridge datum, the five listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$**.**

Constructive story: A $\mathsf{RectifiableCurve}^\uparrow$ packet is generated from a MetricUp carrier and distance row, a ContinuousMapUp path graph, a finite interval row, an ordered partition ledger, a DyadicRatCoreUp edge-length sum, a common-refinement row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$, seed)

Scope closed: Definition 3443.1 and Definition 3443.2 name the finite partition-length carrier and its five-row NameCert obligation surface over $\mathsf{Met}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, and $\mathsf{BoundedVariation}^\uparrow$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an arclength supremum, quotient equality of paths, completed path spaces, host measure construction, ambient metric completion, or bridge evidence.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement classification, dyadic edge-length exactness, transport, replay, provenance, and ledger non-escape for BEDC.Derived.RectifiableCurveUp.

Chapter 2491

A Concrete Naming Certificate for $\mathsf{HeineCantor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{HeineCantor}^{\uparrow}$ packet records compact-metric uniform-modulus extraction as finite NameCert data. It reads the finite-net surface of chapter 129, the endpoint mesh and refinement rows of chapter 3431, the pointwise modulus rows of chapter 130, and the target consumer interface of chapter 369. The interval-specialized consumer also factors compact finite-net rows through the interval mesh and refinement rows of chapter 3868 before it exports the uniform-modulus consumer route. The chapter names the constructive route from compact finite data and local continuity rows to one displayed uniform modulus, without importing open-cover compactness, a selected subcover functional, trichotomy on host reals, or ambient completeness.

Definition 2491.1 (Heine-Cantor carrier)**.** *A* $\mathsf{HeineCantor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite* $\mathsf{ProbeBundle}$ *net rows at the source precision scales consumed by the modulus extraction.*
2. *Y is the target metric row whose distance ledger receives the final uniform estimate.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from X to Y, including the pointwise modulus rows attached to listed net centers.*
4. *E is the positive target precision row, read as a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint history when the consumer presents a real tolerance.*
5. *N is the compact finite-net row selected from X for the scale determined by E.*
6. *R lists the local pointwise radius rows for the centers of N, with rational lower-bound readbacks through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the local rational classifier.*
7. *L is the* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *mesh and refinement handoff row when the finite-net data is produced from located interval endpoints.*
8. *M is the exported source modulus row, obtained by the finite fold over the radius rows R.*
9. *T is the target estimate row, combining compact-net coverage, two pointwise estimates around a selected center, and the target metric triangle route.*

10. *H records componentwise* hsame *transport for the compact source, target metric, continuous-map, precision, finite-net, radius, located-interval, modulus, and target-estimate rows.*
11. *C records the displayed* Cont *replay route from compact data and pointwise continuity data to the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer.*
12. *P records* Pkg *provenance over* $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, *and* Cont *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}$ *row.*

The carrier exposes only compact finite-net rows, continuous-map modulus rows, located mesh/refinement rows, endpoint readbacks, componentwise transport, replay, provenance, and local naming. It has no coordinate for arbitrary open covers, selected center choice, host real equality, quotient completion, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineCantorUp.HeineCantorTasteGate single carrier alignment`

Theorem 2491.2 (Heine-Cantor compact uniform modulus)**.** *For every accepted* $\mathsf{HeineCantor}^{\uparrow}$ *carrier*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A),$$

the exported source row M is a $\mathsf{UniformModulus}^{\uparrow}$ *witness for the continuous map row F at target precision E. Any consumer of the uniform-modulus witness must factor through the finite route*

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

Thus compact-metric finite nets, $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows,* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *mesh and refinement data,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint readbacks, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *ledger suffice to export the compact uniform modulus.*

Proof. Project the carrier of Definition 3444.1. The target precision row E determines the compact source scale, and the $\mathsf{CompactMetric}^{\uparrow}$ row X supplies the finite net row N at that scale.

Read the continuous-map row F at each center of N. The pointwise radius rows are stored in R, and their rational lower-bound readbacks are attached to the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ endpoint rows already present in the packet.

When the source net is produced from an interval presentation, the row L supplies the located mesh and finite refinement data of chapter 3431. This row is a handoff into the compact finite-net route, not a general open-cover compactness principle.

Fold the finite family of radius rows R into the single source row M. The fold is finite because it ranges only over the centers listed by N, and componentwise transport in H preserves the compact source, continuous-map, endpoint, radius, and folded-modulus rows.

Finally use the target estimate row T. Compact-net coverage places two source points near a listed center, the pointwise rows in R estimate their images at that center, and the target metric ledger combines the estimates at precision E. Replay by C then presents M to the $\mathsf{UniformModulus}^{\uparrow}$ consumer, while P and A record the provenance and local naming ledger. □

Theorem 2491.3 (Heine-Cantor finite-net radius fold)**.** *Let*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A)$$

be an accepted $\mathsf{HeineCantor}^{\uparrow}$ *carrier. The row M is obtained only by a finite fold over the pointwise radius rows R indexed by the finite net N. When located interval data is present, the row L supplies*

the mesh and refinement handoff before the fold is read. Consequently every $\mathsf{UniformModulus}^{\uparrow}$ *consumer of* K *must read* N, R, L *before it reads the folded modulus row* M *and the target estimate row* T*.*

Proof. Project the carrier of Definition 3444.1. The finite-net row is N, the pointwise radius inventory is R, the located mesh and refinement handoff is L, and the exported modulus row is M.

Apply Definition 3444.2. Its route forces every compact uniform-modulus export through

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

The fold is finite because its indices are exactly the centers listed by the $\mathsf{ProbeBundle}$ net row N. The located interval row L refines the mesh when such data is used, but it does not replace the net-indexed radius list by open-cover compactness or selected subcovers. Thus any consumer must read N, R, L before the folded row M and the estimate row T are visible. □

Theorem 2491.4 (Heine-Cantor Dini window handoff)**.** *A* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *consumer may use the* $\mathsf{HeineCantor}^{\uparrow}$ *compact uniform modulus only through the finite route*

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

Along this route the compact-window modulus rows supplied by the Heine–Cantor packet feed the Dini finite-window schedule of chapter 1436 without open-cover compactness, selected subcovers, countable choice, quotient real equality, or host real trichotomy.

Proof. Start with the Heine–Cantor carrier of Definition 3444.1. The compact source X, continuous-map row F, and target precision row E determine the finite-net and pointwise-radius rows N, R, L.

Fold those rows into the exported modulus and target estimate rows M, T. By Definition 3444.2, this is the only route by which a compact uniform modulus is presented to a $\mathsf{UniformModulus}^{\uparrow}$ consumer.

Now read the Dini packet through chapter 1436. Its finite-window schedule consumes compact-domain rows, continuous-map family rows, windows, readbacks, and Real seals; the Heine–Cantor route supplies the compact-window modulus part of that inventory while preserving the same finite-net discipline.

The structural rows H, C, P, A then transport, replay, source, and name the handoff. Since neither carrier contains an open-cover, selected-subcover, choice, quotient-equality, or trichotomy coordinate, the Dini consumer receives only the finite compact-window modulus route. □

Theorem 2491.5 (Heine-Cantor compact-modulus selector handoff)**.** *Let* $K = (X, Y, F, E, N, R, L, M, T, H, C, P,$ *be an accepted* $\mathsf{HeineCantor}^{\uparrow}$ *packet. Suppose the compact source row* X*, the total-bounded finite-net row exposed by* N*, the continuous-map row* F*, the dyadic tolerance row read from* E*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal all match an accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *packet. Then every* $\mathsf{UniformModulus}^{\uparrow}$ *export from* K *factors through Definition 4105.3 before the row* M *is visible to a consumer. The handoff exports no open-cover compactness, countable choice, host function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the Heine–Cantor carrier of Definition 3444.1. Its compact source, continuous-map, tolerance, finite-net, readback, seal, transport, replay, provenance, and local naming rows are exactly the rows available to a uniform-modulus consumer.

Use the assumed row match with the $\mathsf{CompactModulusSelector}^{\uparrow}$ packet. The compact source row, total-bounded finite-net row, continuous-map row, dyadic tolerance ledger, finite windows,

regular readbacks, and Real seal are then the rows named by the selector carrier before its uniform-modulus export is visible.

Apply Definition 4105.3. It forces the selector export to pass through the finite route from compact metric data, total-bounded nets, pointwise continuity, dyadic comparison, finite windows, regular readback, and the Real seal before the $\mathsf{UniformModulus}^{\uparrow}$ row can be consumed.

Transport the resulting route back along the Heine–Cantor rows H, C, P, A. The exported row M is therefore a Heine–Cantor presentation of the selector's finite-net modulus route, and no excluded open-cover, choice, host-equality, or ambient-completeness coordinate is introduced. □

Theorem 2491.6 (Heine-Cantor compact uniform route scope)**.** *Every accepted* $\mathsf{HeineCantor}^{\uparrow}$ *consumer that exports a uniform modulus must first read* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus rows,* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ *radius-extraction rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and terminal* $\mathsf{Real}^{\uparrow}$ *seals. The export route is therefore an L10 finite dependency route through the displayed rows of Definition 3444.1, Definition 3444.2, Definition 3444.3, Definition 3444.4, and Definition 3444.5; it refuses open-cover compactness, selected moduli, quotient function equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 3444.1. Its compact source row X, continuous-map row F, target precision row E, finite-net row N, radius rows R, located handoff row L, exported modulus row M, and target estimate row T expose the compact finite route rather than an open-cover surface.

Apply the compact uniform modulus theorem and the finite-net radius fold, Definition 3444.2 and Definition 3444.3. They force every export through compact finite nets, pointwise radius rows, finite radius folding, and the target estimate ledger before the structural rows can replay the modulus.

Next use the Dini window handoff and compact-modulus selector handoff, Definition 3444.4 and Definition 3444.5. These handoffs identify the downstream L10 reads as finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and Real seals.

The extractor row is therefore read only as finite radius extraction attached to those displayed dependencies. Since none of the cited rows contains an open cover, selected modulus, quotient function equality, or ambient completeness coordinate, the consumer receives exactly the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HeineCantor}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise modulus rows, LocatedIntervalCompactness mesh/refinement rows, RealUp and RegSeqRat endpoint readbacks, a finite radius fold, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3444.1 and Definition 3444.2 seed the compact finite-net route from pointwise continuity data to a UniformModulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, choice, host real equality, trichotomy, quotient completion, ambient Real completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net extraction, finite radius fold, target estimate handoff, transport, replay, provenance, and local naming for BEDC.Derived.HeineCantorUp.

Chapter 2492

A Concrete Naming Certificate for $\mathsf{RealUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{RealUniformSpace}^{\uparrow}$ packet records the finite L10 interface at which real endpoints, regular rational readback, stream windows, dyadic tolerance ledgers, and metric distance witnesses are assembled into a local uniformity surface. It sits beside the $\mathsf{Real}^{\uparrow}$ endpoint certificate of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{Met}^{\uparrow}$ distance witness route of chapter 57, and the regular Cauchy uniformity route of chapter 1628. The packet supplies a finite uniformity interface for Real and Completion consumers; it does not import host uniform spaces, quotient completions, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2492.1 (RealUniformSpace carrier)**.** *A* $\mathsf{RealUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, Q, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row, listing the endpoint names whose uniform comparison is allowed to be read.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the endpoint names.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected for the displayed endpoint and readback rows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the inspected radii, refinements, and finite comparison budgets.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-witness row for the same displayed windows and dyadic tolerances.*
6. *E is the* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *entourage route reached only after R, Q, W, D, M have been displayed.*
7. *H records componentwise* hsame *transport for the endpoint, readback, window, tolerance, metric, entourage, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniformity request through $R, Q, W, D, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by componentwise displayed transport through these rows. It contains no host uniform-space carrier, quotient-completion equality, selected representative, open-cover compactness datum, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Theorem 2492.2 (RealUniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformSpace}^{\uparrow}$ *carrier* $U = (R, Q, W, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, endpoint access through* R*, regular rational readback through* Q*, finite window selection through* W*, dyadic tolerance accounting through* D*, metric distance witnessing through* M*, regular Cauchy entourage routing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3445.1. Each public obligation is attached to one displayed row, so carrier admission supplies precisely the endpoint, readback, window, tolerance, metric, entourage, transport, replay, provenance, and naming data.

The transport row is componentwise. Moving a packet along H changes only the displayed representatives inside their hsame classes and cannot create a host uniform-space carrier or a quotient-completion equality.

Replay by C follows the same route, and P, N source and name that route locally. Since no accepted row stores a selected representative, countable-choice schedule, open-cover compactness datum, or ambient completeness witness, the NameCert surface is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.RealUniformSpaceUp.RealUniformSpaceTasteGate single carrier alignment`

Theorem 2492.3 (RealUniformSpace Cauchy-entourage route)**.** *Every* $\mathsf{RealUniformSpace}^{\uparrow}$ *consumer that asks for a Cauchy entourage comparison must traverse*

$$R \longmapsto Q \longmapsto W \longmapsto D \longmapsto M \longmapsto E$$

before using H, C, P, N*. Thus the real-uniform handoff reaches* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *only after the* $\mathsf{Real}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{Met}^{\uparrow}$ *distance witnesses have been displayed.*

Proof. Begin with a Cauchy-entourage request. The carrier admits endpoint rows only through R, so the request first exposes the real endpoints and then reads their regular rational names through Q.

The finite windows are then selected by W. Tolerance comparisons are recorded by D, and the metric-facing distance witnesses in M are read against the same windows and dyadic ledgers.

Only after those rows are visible may the route reach the entourage row E. The structural rows H, C, P, N transport, replay, source, and name this finite route, so no host uniformity, quotient completion, selected representative, or ambient completeness principle enters the handoff. □

Falsifiable prediction: Every $\mathsf{RealUniformSpace}^{\uparrow}$ Cauchy-entourage read factors through RealUp endpoint rows, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, MetricUp distance witnesses, a RegularCauchyUniformity entourage route, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore, metric, regular-cauchy-uniformity: $\mathsf{RealUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, regular readback, finite stream windows, dyadic tolerance ledgers, metric distance witnesses, regular Cauchy entourage routing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformSpace}^{\uparrow}$ packet is generated from RealUp endpoint rows, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, MetricUp distance witnesses, a RegularCauchyUniformity entourage route, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3445.1, Definition 3445.2, and Definition 3445.3 bind the finite real-uniform certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealUniformSpaceUp.RealUniformSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host uniform spaces, quotient completions, selected representatives, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Cauchy-entourage routing, metric tolerance accounting, and TasteGate field faithfulness for BEDC.Derived.RealUniformSpaceUp.

Chapter 2493

A Concrete Naming Certificate for $\mathsf{BishopRealApartnessTopology}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ packet records the finite neighbourhood surface obtained when Bishop real names are read through apartness-positive and located comparison evidence before any topology-facing consumer sees an open-neighbourhood ledger. It sits beside the terminal real-name seal of chapter 38, the positive separation surface of chapter 304, the located real packet of **??**, the topology interface of chapter 88, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. The packet exports a constructive topology route for one displayed Bishop real neighbourhood; it exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.

Definition 2493.1 (Bishop real apartness topology carrier)**.** *A* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, L, T, W, Q, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name row, read only through the terminal real-name seal selected for the packet.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-evidence row, recording the displayed apartness radius, endpoint readbacks, and positive separation ledger.*
3. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, recording the located left/right comparison windows used by the same neighbourhood request.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *consumer row, admitting only the topology-facing neighbourhood ledger generated from A, L, W, Q, O.*
5. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the real name, apartness evidence, located comparison, and neighbourhood ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations inspected by R, A, L, W.*
7. *O is the open-neighbourhood ledger generated from the apartness-positive evidence and located comparison rows before T is read.*
8. *H records componentwise* hsame *transport for the real name, apartness evidence, located comparison, topology row, stream schedule, regular readback, open-neighbourhood ledger, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Bishop real neighbourhood request through R, A, L, W, Q, O and then to the topology consumer row T.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealApartnessTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, L, T, W, Q, O, H, C, P, N$. It has no coordinate for trichotomy, host open sets, quotient real equality, countable choice, ambient completeness, or a topology object detached from the displayed open-neighbourhood ledger.

Lean4 ✓ `BEDC.Derived.BishopRealApartnessTopologyUp.BishopRealApartnessTopologyTasteGate single carrier alignment`

Theorem 2493.2 (Bishop real apartness topology NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ *carrier $B = (R, A, L, T, W, Q, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealApartnessTopol}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the* $\mathsf{Real}^{\uparrow}$ *name,* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-evidence row,* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row,* $\mathsf{Topology}^{\uparrow}$ *consumer row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, open-neighbourhood ledger, transport, replay, provenance, and local naming;*

2. *apartness-positive visibility, requiring the positive radius, endpoint readbacks, and separation ledger to be displayed in A before any neighbourhood is generated;*

3. *located comparison visibility, requiring every comparison used by the neighbourhood request to pass through the located row L and the shared W, Q windows;*

4. *topology admission, requiring the topology-facing consumer to read only the generated ledger O through T;*

5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No Bishop real apartness topology obligation is stored outside B, and the packet exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 3446.1. The displayed packet exposes exactly the real name, apartness-positive evidence, located comparison row, topology consumer row, stream schedule, regular readback, open-neighbourhood ledger, and structural rows.

Apartness-positive visibility is the row A. It carries the positive radius, endpoint readbacks, and separation ledger before the open-neighbourhood ledger is generated, so no trichotomy or host inequality row is needed.

Located comparison visibility is the row L together with W and Q. These rows align the comparison request with the finite stream window and regular-rational readback used by the same real name.

Topology admission is forced through O and T. The ledger O is generated only after the apartness and located rows have been displayed, and T admits that ledger as the topology-facing surface.

Finally H, C, P, N preserve only the displayed rows by transport, replay, provenance, and local naming. Since the carrier has no coordinate for host open sets, quotient equality, choice, completeness, or detached topology data, the listed rows exhaust the local NameCert surface. □

Theorem 2493.3 (Bishop real apartness topology Real seal)**.** *Every* Real$^{\uparrow}$*-facing or* Topology$^{\uparrow}$*-facing consumer of an accepted* BishopRealApartnessTopology$^{\uparrow}$ *carrier must pass through the route*

$$R, A, L, W, Q, O, H, C, P, N$$

before reading the topology row T. The handoff exposes the real name, apartness-positive evidence, located comparison rows, finite stream schedule, regular rational readback, open-neighbourhood ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.*

Proof. Start with a Real-facing or topology-facing read of the accepted carrier. The only real-name source is R, and its finite observations are read through W and Q.

The neighbourhood evidence is forced through A and L. The apartness-positive row supplies the displayed separation ledger, while the located comparison row supplies the comparison windows used by the same request.

The topology consumer can read only after O has been generated. Thus T receives a ledger assembled from the displayed real, apartness, locatedness, stream, and regular-readback rows, not a host open set or quotient-open representative.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier contains no trichotomy, quotient equality, countable-choice, host-open, or completeness coordinate, every accepted handoff remains on the displayed finite route. □

Falsifiable prediction: Every accepted BishopRealApartnessTopology$^{\uparrow}$ Real-facing or topology-facing read factors through Real$^{\uparrow}$, ApartnessReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, Topology$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, open-neighbourhood ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: real, apartnessreal, locatedreal, topology, streamname, regseqrat: BishopRealApartnessTo is not bijective to any listed sibling because it jointly carries a real-name seal, apartness-positive evidence, located comparison rows, a topology-facing neighbourhood ledger, finite stream windows, regular rational readback, transport, replay, provenance, and local naming as one constructive topology packet.

Closure status for NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$, seed)

Scope closed: Definition 3446.1, Definition 3446.2, and Definition 3446.3 seed the finite Bishop real apartness topology route through Real$^{\uparrow}$, ApartnessReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, Topology$^{\uparrow}$, StreamName$^{\uparrow}$, and RegSeqRat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopRealApartnessTopologyUp.BishopRealApartnessTopologyTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close trichotomy, host open sets, quotient real equality, countable choice, Lean theorem checking, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, apartness-positive visibility, located comparison visibility, topology admission, Real seal handoff, transport stability, and replay stability for BEDC.Derived.BishopRealApartnessTopologyUp.

Constructive story: The construction starts from a Bishop real name, records apartness-positive and located comparison evidence over finite StreamName and RegSeqRat windows, generates an open-neighbourhood ledger, and exposes the topology-facing consumer only along the same finite NameCert route.

Chapter 2494

A Concrete Naming Certificate for $\mathsf{FatouLemma}^{\uparrow}$

Following Fatou and Lebesgue's convergence theory, the $\mathsf{FatouLemma}^{\uparrow}$ packet records the finite lower-semicontinuity route from nonnegative measurable families to an integral lower-bound seal. It depends on the measure carrier of chapter 92, the integral readback surface of chapter 123, the regular rational tolerance interface of chapter 301, the Real seal of chapter 38, and the dominated-convergence boundary of chapter 2340. The packet is a lower-bound certificate only: it stores the rows needed to name Fatou's inequality inside BEDC, but it does not construct ambient measure completion, almost-everywhere quotient equality, quotient function equality, countable choice, or a full monotone convergence theorem.

Definition 2494.1 (FatouLemma carrier)**.** *A* $\mathsf{FatouLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, I, A, L, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row naming the finite measurable-domain data and the measurable-set ledger read by the functions below.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row used for every lower-bound comparison in the packet.*
3. *A is the nonnegative family row: each displayed function in the sequence is admitted as a measurable nonnegative integrand over the same M.*
4. *L is the liminf window row, storing the finite tail windows whose lower envelope is consumed by the Fatou comparison.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row for the tail-window lower bounds.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *lower-bound seal asserting that the named liminf integral is bounded above by the displayed lower limit of integral readbacks.*
7. *T records the transport from the family rows through the liminf windows to the integral and Real readbacks.*
8. *H records componentwise* hsame *transport for M, I, A, L, Q, R, T and the structural rows below.*
9. *C records the displayed* Cont *replay from Measure source rows to integral reads, from nonnegative family rows to liminf windows, and from regular rational lower bounds to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FatouLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for an ambient completed measure algebra, host almost-everywhere equality, quotient function equality, a countable-choice selector for tails, or a monotone-convergence theorem.

Lean4 ✓ `BEDC.Derived.FatouLemmaUp.FatouLemmaTasteGate single carrier alignment`

Theorem 2494.2 (FatouLemma NameCert obligations)**.** *For every accepted* $\mathsf{FatouLemma}^{\uparrow}$ *carrier* $F = (M, I, A, L, Q, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FatouLemma}^{\uparrow}}$ *surface consists exactly of Measure source exposure through* M*, Integral readback through* I*, nonnegative family admission through* A*, liminf window exposure through* L*, RegSeqRat tolerance readback through* Q*, Real lower-bound sealing through* R*, transport through* T, H*, continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Thus every accepted Fatou comparison is a finite lower-bound packet over the displayed rows. No obligation may be discharged by ambient measure completion, host almost-everywhere equality, quotient function equality, countable choice, or a monotone convergence theorem.

Proof. Project the carrier of Definition 3447.1. The rows $M, I, A, L, Q, R, T, H, C, P, N$ are the only coordinates visible to the local classifier.

The analytic source rows are read first. The measure row M supplies the measurable-domain data, the integral row I supplies the readback surface, and the family row A records nonnegativity for each displayed function.

The lower-bound route is finite. The liminf window row L lists the tail windows, the regular-rational row Q records the tolerance readbacks for those windows, and the Real row R seals the comparison after the same data has been transported and replayed through T, H, C.

The package and local naming rows P, N source exactly this route. Since the excluded completion, equality, quotient, choice, and monotone-convergence coordinates are absent from the carrier, they cannot discharge a NameCert obligation. □

Theorem 2494.3 (FatouLemma liminf handoff)**.** *Let* $F = (M, I, A, L, Q, R, T, H, C, P, N)$ *be an accepted* $\mathsf{FatouLemma}^{\uparrow}$ *packet. Every liminf consumer factors through the finite route*

$$\mathsf{Measure}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto L \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads the Measure source row M*, the Integral readback row* I*, the nonnegative family row* A*, the liminf windows in* L*, and the RegSeqRat tolerance row* Q *before the lower-bound seal* R *is visible.*

Proof. Begin with the displayed liminf window row. Because L is part of the accepted carrier, each requested tail window is tied to the same Measure source M and the same Integral readback I.

Use the nonnegative family row A to keep every tail comparison inside the displayed measurable family. The row Q then records the regular rational tolerances used by those finite windows.

Finally read the Real seal. The rows T, H, C transport and replay the same lower-bound route into R, while P, N package and name it. No external tail selector, quotient equality, or completion principle is available to replace this handoff. □

Theorem 2494.4 (FatouLemma dominated-convergence boundary)**.** *The* FatouLemma$^{\uparrow}$ *packet supplies the lower-bound side needed by the dominated-convergence route of chapter 2340: it exposes the nonnegative family row, liminf window row, RegSeqRat tolerance readback, Integral readback, and Real lower-bound seal as finite NameCert data over the same Measure source.*

It does not export the domination majorant, almost-everywhere convergence quotient, two-sided limit equality, or global convergence conclusion of DominatedConvergence$^{\uparrow}$*. Those remain obligations of the dominated-convergence packet, whose boundary also cites the human-known comparison tradition around* TanneryTheorem$^{\uparrow}$*.*

Proof. Project the Fatou carrier and retain only the rows consumed by a dominated-convergence lower-bound reader: $M, I, A, L, Q, R, T, H, C, P, N$.

Apply Definition 3447.3. The lower-bound read reaches the Real seal only after the Measure source, Integral readback, nonnegative family row, liminf windows, and regular rational tolerance row have all been displayed.

The dominated-convergence packet has additional surface: domination data, convergence equality data, and a two-sided limit conclusion. Since none of those coordinates appears in Definition 3447.1, this chapter supplies only the Fatou lower-bound boundary and not the full dominated-convergence theorem. □

Closure status for NameCert$_{\text{FatouLemma}^{\uparrow}}$**.**

Constructive story: A FatouLemma$^{\uparrow}$ packet is generated from a Measure source row, an Integral readback row, a nonnegative family row, a liminf window row, RegSeqRat tolerance readback, a Real lower-bound seal, transport, continuation replay, package provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{FatouLemma}^{\uparrow}}$, seed)

Scope closed: Definition 3447.1, Definition 3447.2, Definition 3447.3, and Definition 3447.4 bind the finite lower-bound packet and its dominated-convergence boundary.

Formal verification: FormalStatus(NameCert$_{\text{FatouLemma}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FatouLemmaUp.FatouLemmaTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim ambient measure completion, host almost-everywhere equality, quotient function equality, countable choice, a full monotone convergence theorem, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Measure source exposure, Integral readback, nonnegative family admission, liminf window exposure, RegSeqRat tolerance readback, and Real lower-bound sealing for the same packet.

Chapter 2495

A Concrete Naming Certificate for $\mathsf{RadonNikodym}^{\uparrow}$

Following Radon's derivative of measures, Nikodym's theorem, and the standard measure-theoretic presentation of absolute continuity, the $\mathsf{RadonNikodym}^{\uparrow}$ packet records a finite density route from one displayed measure row to another. It sits after the measurable-event carrier of chapter 92, the integral readback surface of chapter 123, the regular rational tolerance interface of chapter 301, the Real seal of chapter 38, and the lower-bound comparison boundary of chapter 3447. The packet names the rows needed to consume a density witness inside BEDC; it is not ambient measure completion, quotient almost-everywhere equality, countable choice, or an imported Radon–Nikodym theorem.

Definition 2495.1 (RadonNikodym carrier)**.** *A* $\mathsf{RadonNikodym}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *M_μ is the base* $\mathsf{Measure}^{\uparrow}$ *row, exposing the measurable-event family, finite-additivity rows, and Real-valued measure readbacks.*
2. *M_ν is the target* $\mathsf{Measure}^{\uparrow}$ *row on the same displayed event family.*
3. *A is the absolute-continuity ledger: every event read as μ-null carries a displayed route to a ν-null comparison before any density row is consumed.*
4. *D is the candidate density row, stored as a nonnegative measurable integrand over the shared event family.*
5. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row assigning each displayed event E the integral of D over E.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance ledger for the finite simple-function and event-window approximants used by the density comparison.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal comparing the target-measure readback $\nu(E)$ with the integral readback of D on each requested event window.*
8. *H records componentwise* hsame *transport for $M_\mu, M_\nu, A, D, I, Q, S$ and the structural rows below.*
9. *C records displayed* Cont *replay from measure rows through absolute-continuity ledgers, density windows, integral readback, regular rational tolerance, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{RadonNikodym}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FatouLemma}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed measure algebras, quotient almost-everywhere equality, countable-choice selectors for representatives, or a host Radon–Nikodym theorem.

Lean4 ✓ `BEDC.Derived.RadonNikodymUp.RadonNikodymTasteGate single carrier alignment`

Theorem 2495.2 (RadonNikodym NameCert obligations)**.** *For every accepted* $\mathsf{RadonNikodym}^{\uparrow}$ *carrier* $R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *shared-event carrier admission for the two* $\mathsf{Measure}^{\uparrow}$ *rows* M_μ *and* M_ν*;*
2. *absolute-continuity exposure through* A*, requiring null-event requests to remain visible in the measure ledger;*
3. *density admission through* D*, requiring the candidate density to be a displayed nonnegative measurable integrand;*
4. *integral readback through* I*, requiring each event-facing density read to pass through the public* $\mathsf{Integral}^{\uparrow}$ *row;*
5. *regular-rational tolerance accounting through* Q*, requiring finite simple-function and event-window approximants before Real-facing comparison;*
6. *Real sealing through* S*, requiring each displayed event comparison to be named only after* M_ν, D, I, Q *have been read;*
7. *structural stability through* H, C, P, N*, requiring transport, replay, provenance, and local naming to preserve exactly the displayed route.*

No obligation may be discharged by ambient measure completion, quotient almost-everywhere equality, countable choice, or an imported Radon–Nikodym theorem.

Proof. Project the carrier of Definition 3448.1. The two measure rows, the absolute-continuity ledger, the candidate density, the integral readback, the tolerance ledger, the Real seal, and the structural rows are the entire local surface.

The source side is read first. The rows M_μ and M_ν expose the common measurable-event family, while A records the null-event implication that licenses density search without hiding a completed measure algebra.

The density side is finite. The row D admits the displayed nonnegative measurable integrand, I reads its eventwise integrals, and Q records the finite tolerance windows used before a Real comparison is available.

Finally S seals the eventwise comparison and H, C, P, N transport, replay, source, and name the same route. Since the excluded completion, quotient, choice, and host theorem coordinates are absent from the carrier, they cannot discharge any NameCert obligation. □

Theorem 2495.3 (RadonNikodym density route)**.** *Every* $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RadonNikodym}^{\uparrow}$ *carrier* $R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$ *must read the finite route*

$$M_\mu, M_\nu \longmapsto A \longmapsto D \longmapsto I \longmapsto Q \longmapsto S \longmapsto H, C, P, N.$$

Thus the target measure is consumed through displayed absolute-continuity data, a density row, Integral readback, RegSeqRat tolerance ledgers, and the Real seal before any density conclusion is available.

Proof. Begin with the two measure rows. A consumer first reads M_μ and M_ν, so every event request remains attached to the same displayed measurable-event family.

Next read the absolute-continuity and density rows. The ledger A keeps null-event transfer visible, and the row D supplies the candidate density as a displayed nonnegative measurable integrand rather than an almost-everywhere quotient class.

The integral and Real-facing route is downstream. The row I reads the density on each requested event, Q records the regular rational approximants and tolerances, and S seals the comparison in $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N preserve only this route. Therefore no consumer can bypass $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or the Fatou-style lower-bound boundary of chapter 3447 by importing measure completion, quotient almost-everywhere equality, countable choice, or the classical Radon–Nikodym theorem as hidden data. □

Falsifiable prediction: Every $\mathsf{RadonNikodym}^{\uparrow}$ density-facing or Real-facing read factors through displayed Measure source rows, an absolute-continuity ledger, a density integrand row, Integral readback, RegSeqRat tolerance windows, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: measure, integral, real, regseqrat, fatou-lemma: $\mathsf{RadonNikodym}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two measure rows, an absolute-continuity ledger, a density integrand, eventwise integral readback, tolerance accounting, Real comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}$.

Constructive story: A $\mathsf{RadonNikodym}^{\uparrow}$ packet is generated from two shared-event Measure rows, an absolute-continuity ledger, a displayed density integrand, Integral readback, RegSeqRat tolerance windows, Real comparison seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3448.1, Definition 3448.2, and Definition 3448.3 seed the finite density route from shared Measure rows to Integral, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RadonNikodymUp.RadonNikodymTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim ambient measure completion, quotient almost-everywhere equality, countable choice, imported Radon–Nikodym theorem use, Lean theorem checking, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-event measure exposure, absolute-continuity ledger exactness, density admission, integral readback, regular-rational tolerance accounting, Real sealing, transport stability, and replay stability for BEDC.Derived.RadonNikodymUp.

Chapter 2496

A Concrete Naming Certificate for $\mathsf{GaussBonnetBoundary}^{\uparrow}$

Following Chern, Gauss, Bonnet, and Stokes, the $\mathsf{GaussBonnetBoundary}^{\uparrow}$ packet records the finite boundary-sensitive route from an oriented compact surface window to an Euler-characteristic ledger. It consumes the chart and orientation discipline of chapter 96, the metric rows of chapter 133, the curvature rows of chapter 135, the differential-form and Stokes surface of chapter 136, the cohomology-class readback of chapter 137, and the characteristic-class handoff of chapter 272. The packet names only the displayed surface, boundary, curvature, form, cohomology, transport, replay, provenance, and local NameCert rows; it does not claim host smooth integration, unrestricted compactness, full topological classification, Lean verification, or bridge checking.

Definition 2496.1 (Gauss-Bonnet boundary carrier)**.** *A* $\mathsf{GaussBonnetBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, K, B, F, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the oriented compact surface window row, read through the local* $\mathsf{Manifold}^{\uparrow}$ *chart, atlas, boundary, and orientation data.*
2. *M is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row assigning the displayed metric data on the same surface window.*
3. *K is the* $\mathsf{Curvature}^{\uparrow}$ *row naming the Gaussian-curvature display produced from M on the interior chart windows.*
4. *B is the boundary geodesic-curvature row, carrying the finite boundary arcs, endpoint turns, and tangent-readback data used by the boundary term.*
5. *F is the* $\mathsf{DiffForm}^{\uparrow}$ *and Stokes row, recording the two-dimensional curvature form, the boundary one-form readback, and the Stokes-compatible orientation route.*
6. *D is the* $\mathsf{DeRham}^{\uparrow}$ *class row that reads the curvature form as a cohomology-class witness over the displayed surface window.*
7. *E is the Euler-characteristic ledger row, storing the finite cell, triangulation, or admissible decomposition count consumed by the comparison.*
8. *H records componentwise* hsame *transport for S, M, K, B, F, D, E and the structural rows below.*
9. *C records displayed* Cont *replay from the surface window through metric, curvature, boundary, form, cohomology, and Euler-ledger reads in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{ChernWeil}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, K, B, F, D, E, H, C, P, N$. It has no coordinate for host smooth integration, arbitrary compact surface selection, quotient topological classification, external triangulation equality, or a full Chern-Weil theory.

Theorem 2496.2 (Gauss-Bonnet boundary NameCert obligations)**.** *For every accepted* $\mathsf{GaussBonnetBoundary}^{\uparrow}$ *carrier $G = (S, M, K, B, F, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *surface consists exactly of oriented compact surface admission through S, Riemannian metric admission through M, curvature display through K, boundary geodesic-curvature exposure through B, differential-form and Stokes routing through F, DeRham class readback through D, Euler-characteristic ledger exposure through E, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N.*

No Gauss-Bonnet boundary obligation may be discharged by a host integration operator, an unrestricted compactness principle, a hidden smooth structure, a quotient classification theorem, or a Chern-Weil characteristic-class result not displayed in the packet provenance.

Proof. Project the carrier of Definition 3449.1. The rows $S, M, K, B, F, D, E, H, C, P, N$ are the only coordinates visible to the local classifier.

The geometric source is read first. The oriented surface window S fixes the chart, atlas, orientation, and boundary data; the metric row M and curvature row K then expose only the curvature display available over that same window.

The boundary and form route is separate. The row B supplies the boundary geodesic-curvature ledger, while F supplies the differential-form and Stokes readback tying the interior and boundary terms to the displayed orientation.

Finally D and E provide the cohomology and Euler-characteristic ledgers consumed by the comparison. Transport, replay, provenance, and naming preserve this row order through H, C, P, N, so no excluded host or classification coordinate can discharge a local obligation. □

Lean4 ✓ `BEDC.Derived.GaussBonnetBoundaryUp.GaussBonnetBoundaryTasteGate single carrier alignment`

Theorem 2496.3 (Gauss-Bonnet boundary Stokes-curvature handoff)**.** *Let $G = (S, M, K, B, F, D, E, H, C, P, N)$ be an accepted* $\mathsf{GaussBonnetBoundary}^{\uparrow}$ *packet. Every accepted boundary-sensitive curvature read factors through the finite route*

$$\mathsf{Manifold}^{\uparrow} \longmapsto \mathsf{RiemannianMetric}^{\uparrow} \longmapsto \mathsf{Curvature}^{\uparrow} \longmapsto \mathsf{DiffForm}^{\uparrow} \longmapsto \mathsf{DeRham}^{\uparrow} \longmapsto \mathsf{ChernWeil}^{\uparrow}.$$

Concretely, the consumer must read the oriented compact surface row S, the metric row M, the curvature row K, the boundary geodesic-curvature row B, the Stokes form row F, the DeRham class row D, and the Euler-characteristic ledger E before the local $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *row is visible.*

Proof. Begin with the surface window. The row S supplies the only accepted manifold, orientation, and boundary source for the packet.

Metric and curvature data are then read from M and K. These rows are tied to S by the classifier and cannot be replaced by an ambient smooth manifold or host integration surface.

The boundary term must pass through B, and the interior-plus-boundary form route must pass through F. The DeRham row D and Euler ledger E then supply the cohomology and characteristic-count surfaces consumed by the Chern-Weil-facing handoff.

The structural rows H, C, P, N transport, replay, source, and name exactly this finite route. Hence the packet exports a boundary Stokes-curvature handoff, not a full smooth integration theorem or topological classification theorem. □

Falsifiable prediction: Every accepted $\mathsf{GaussBonnetBoundary}^{\uparrow}$ read that exposes an Euler-characteristic ledger must also expose the oriented compact surface row, metric row, curvature row, boundary geodesic-curvature row, differential-form/Stokes row, and DeRham class row.

Independence witness: manifold, riemannianmetric, curvature, diffform, derham, chernweil: none is bijective to $\mathsf{GaussBonnetBoundary}^{\uparrow}$ because this packet jointly carries an oriented compact surface window, metric and curvature displays, boundary geodesic-curvature data, Stokes form routing, DeRham readback, Euler-characteristic ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GaussBonnetBoundary}^{\uparrow}$ packet is generated from an oriented compact surface window, a RiemannianMetric row, a Curvature row, a boundary geodesic-curvature row, a DiffForm/Stokes row, a DeRham class row, an Euler-characteristic ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3449.1, Definition 3449.2, and Definition 3449.3 seed the finite boundary-sensitive Gauss-Bonnet NameCert route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.GaussBonnetBoundaryUp.GaussBonnetBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host smooth integration, unrestricted compactness, full topological classification, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric and curvature row exactness, boundary geodesic-curvature exposure, Stokes form routing, DeRham readback, Euler-ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.GaussBonnetBoundaryUp.

Chapter 2497

A Concrete Naming Certificate for $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$

The $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ packet realizes the Force–MetaCIC relation discipline of section 9416.5 as finite NameCert data. It consumes the same-schema carrier of chapter 1090, the family-level audit rows of chapter 807, the confluence-audit packet of chapter 735, the quotient-refusal boundary of chapter 634, and the authorized-generator route of chapter 525. The packet names a two-chain relation schema only; it does not claim empirical force content, identity between the domains, object-level MetaCIC closure, or bridge-checked verification.

Definition 2497.1 (Force-MetaCIC same-schema carrier)**.** *A* $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, B, L, P, Q, R, V, H, C, K, N)$$

with the following displayed rows:

1. *A is the force-cognition source-chain row, recording the displayed primitive-token, obstruction, classifier, ledger, and boundary stages.*
2. *B is the MetaCIC self-recognition source-chain row, recording the displayed token, audit, carrier, classifier, ledger, and boundary stages.*
3. *L is the finite two-chain layer-map row from the force chain to the MetaCIC chain.*
4. *P is the preserved-structure row: primitive-token replacement by relation, classifier, invariant, ledger, certificate, and boundary.*
5. *Q is the not-preserved row: empirical physical content, numerical constants, machine-checked proofs, phenomenal essence, object-level* T*, and global universe structure.*
6. *R is the failure-boundary row, refusing any read that treats same-schema exactness as object identity, empirical transfer, quotient equality, or a checked proof.*
7. *V is the same-schema verdict row, assigning paper-certificate strength to the displayed relation.*
8. *H records componentwise* hsame *transport for the two source chains, layer map, preserved row, not-preserved row, boundary row, and verdict row.*
9. *C records displayed* Cont *replay through A, B, L, P, Q, R, V in that order.*
10. *K is the* Pkg *provenance over* $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, B, L, P, Q, R, V, H, C, K, N$. It has no coordinate for empirical-force semantics, object identity, quotient transport, hidden proof transfer, global universe structure, or unchecked promotion of paper-certificate strength.

Lean4 ✓ `BEDC.Derived.ForceMetaCICSameSchemaUp.ForceMetaCICSameSchemaTasteGate single carrier alignment`

Theorem 2497.2 (Force-MetaCIC same-schema NameCert obligations)**.** *For every accepted* ForceMetaCICSameSc*carrier $F = (A, B, L, P, Q, R, V, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$ *surface consists exactly of force-chain admission through A, MetaCIC-chain admission through B, finite layer-map exposure through L, preserved-structure exposure through P, explicit not-preserved exposure through Q, failure-boundary exposure through R, same-schema verdict exposure through V, componentwise transport through H, continuation replay through C, package provenance through K, and local naming through N.*

Any downstream use omitting the not-preserved row Q or the failure-boundary row R cannot consume the same-schema relation, because the packet would then lack the rows that prevent identity collapse, empirical transfer, quotient transport, and proof-strength promotion.

Proof. Project the carrier of Definition 3450.1. The two displayed chains are A and B, and the comparison surface is the layer map L together with the preserved, not-preserved, boundary, and verdict rows P, Q, R, V.

The same-schema read first consumes A, B, L. This fixes the two chains and the relation map without identifying their objects or importing empirical force content into MetaCIC.

The rows P and Q are read together. The first records what the relation preserves, and the second records what it explicitly does not preserve.

The row R then refuses reads that bypass Q or promote the verdict V beyond paper-certificate strength. Transport, replay, provenance, and local naming through H, C, K, N preserve this finite order. □

Theorem 2497.3 (Force-MetaCIC same-schema refusal boundary)**.** *Every accepted* $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ *read that exposes the same-schema verdict must also expose the not-preserved row and the failure-boundary row. Consequently the packet refuses object identity between force cognition and MetaCIC self-recognition, empirical transfer from force theory into BEDC, quotient equality, unchecked proof transfer, and any reading of paper-certificate strength as checked or bridged evidence.*

Proof. Begin with an accepted verdict read. By Definition 3450.1, the verdict coordinate is V and is classified only together with $A, B, L, P, Q, R, H, C, K, N$.

The not-preserved row Q is therefore present before the verdict can be consumed. It records the empirical, numerical, proof-theoretic, phenomenal, object-level, and global-structure content that the relation does not preserve.

The failure-boundary row R is also present. It blocks identity collapse, empirical transfer, quotient equality, hidden proof transfer, and strength promotion before the local name row N can export the packet.

Thus the same-schema relation is available only as the finite carrier named by this chapter. Later Lean work in `TasteGate.lean` must provide Nontrivial and FieldFaithful instances for the displayed carrier rows and classifier projections. □

Falsifiable prediction: Any downstream use of $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ that omits the not-preserved row or the failure-boundary row cannot consume the same-schema relation.

Independence witness: layeredrelationcert, auditmapfamily, metacicconfluenceauditpacket, quotientsoundnessboundary, authorizedgeneratorrecursor: none is bijective to $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ because this packet jointly carries two source chains, a layer map, preserved and not-preserved rows, a failure-boundary row, same-schema verdict, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$.

Constructive story: A $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ packet is generated from a force-cognition source-chain row, a MetaCIC self-recognition source-chain row, a finite two-chain layer map, preserved-structure rows, not-preserved rows, failure-boundary rows, a same-schema verdict row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3450.1, Definition 3450.2, and Definition 3450.3 seed the finite Force-MetaCIC same-schema relation certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim empirical force content, identity between the domains, object-level MetaCIC closure, checked Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-chain layer-map exactness, preserved-row exposure, not-preserved-row exposure, failure-boundary nonescape, same-schema verdict routing, transport, replay, provenance, and local naming for BEDC.Derived.ForceMetaCICSameSchemaUp.

Chapter 2498

A Concrete Naming Certificate for $\mathsf{RealBoundedSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealBoundedSequence}^{\uparrow}$ packet records the finite bounded-sequence neighbour needed by the L10 real route. It reads terminal real names from chapter 38, finite observation windows from chapter 126, regular rational readback from chapter 301, and dyadic tolerance ledgers from chapter 291; its downstream consumer is the subsequence compactness surface of chapter 1254. The packet names bounded real sequence inspection as displayed NameCert data, not as ambient sequential compactness, trichotomy, quotient equality of streams, a selected subsequence, or host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2498.1 (RealBoundedSequence carrier)**.** *A* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row, listing the admitted real-name positions whose values are inspected by the packet.*
2. *B is the bound witness row, recording the displayed lower and upper bounding data and the finite ledger that each inspected value stays inside those bounds.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite observation windows of the source sequence.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximants used by the inspected windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed value and for the bound endpoints consumed by the packet.*
6. *H records componentwise* hsame *transport for S, B, W, R, E and the structural rows below.*
7. *C records the displayed* Cont *replay from source sequence windows through regular rational readback to the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RealBoundedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$ *row.*

The carrier exposes only the source sequence, displayed bound witness, finite windows, regular rational readbacks, terminal Real seals, transport, replay, provenance, and local naming. It has no coordinate for arbitrary subsequence choice, host sequence equality, quotient stream equality, trichotomy on reals, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealBoundedSequenceUp.RealBoundedSequenceTasteGate single carrier alignment`

Theorem 2498.2 (RealBoundedSequence NameCert obligations)**.** *For every accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier* $K = (S, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed sequence source* S*;*
2. *bound-ledger exactness for the row* B*, including the displayed endpoint reads and the finite evidence that inspected values stay within the bound;*
3. *sequence-window stability for every* $\mathsf{StreamName}^{\uparrow}$ *window in* W *and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* R*;*
4. *Real-seal nonescape, requiring every value and bound endpoint read to pass through the terminal* $\mathsf{Real}^{\uparrow}$ *row* E*;*
5. *downstream handoff through* H, C, P, N *before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer may inspect the bounded source.*

No accepted obligation is discharged by host sequence convergence, quotient equality of streams, a selected infinite subsequence, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 3451.1. The visible rows are exactly $S, B, W, R, E, H, C, P, N$, so carrier admission, bound evidence, windows, readback, seal, transport, replay, provenance, and naming exhaust the local certificate surface.

The bound row is read only through displayed windows. A request for an inspected sequence value first enters the source row S, then the finite $\mathsf{StreamName}^{\uparrow}$ window W, and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback R before the bound witness B can compare the value with its endpoints.

The terminal Real row is downstream of that finite route. Componentwise transport in H and displayed replay in C preserve the same rows, while P, N package and name the packet. Since the excluded host and compactness coordinates are absent, they cannot discharge any local NameCert obligation. □

Theorem 2498.3 (RealBoundedSequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier* $K = (S, B, W, R, E, H, C, P, N)$ *factors through the finite route*

$$S, B \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus sequence values and bounding endpoints reach the terminal Real seal only after the displayed source row, bound witness, finite StreamName window, and RegSeqRat readback have been exposed.

Proof. Start with a Real-facing read of a sequence value or bound endpoint. The carrier has no Real coordinate outside E, so the read must use the terminal seal row of Definition 3451.1.

Before E is visible, the requested value is displayed through S and the requested inspection window is displayed through W. The regular rational readback R then supplies the rational approximants against which the bound witness B is checked.

Transport and replay do not introduce a second Real seal. The rows H and C preserve the same finite route, and P, N source and name it for consumers. Hence a Real-facing consumer cannot bypass the sequence source, bound ledger, StreamName window, or RegSeqRat readback. □

Theorem 2498.4 (RealBoundedSequence monotone-subsequence boundary)**.** *The* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer of chapter 1254 may read a bounded source from an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier only through the displayed route of Definition 3451.3. In particular, any monotone-subsequence or cluster-point boundary downstream receives the sequence source S, bound witness B, finite windows W, regular readback R, terminal Real seal E, transport H, replay C, provenance P, and local name N, but it receives no arbitrary subsequence selector, quotient stream equality, trichotomy principle, open-cover compactness proof, or ambient Real-completeness row.*

Proof. Read the bounded source demanded by the Bolzano–Weierstrass carrier. By Definition 3451.2, the only bounded-sequence surface available to that consumer is the finite package $S, B, W, R, E, H, C, P, N$.

Apply Definition 3451.3. The downstream route reaches $\mathsf{Real}^{\uparrow}$ only after the source sequence, bound witness, finite windows, and regular readback have been displayed, so the consumer cannot replace the packet with a host sequence or quotient stream.

The monotone-subsequence and cluster-point boundaries can therefore inherit only this bounded-source surface. Since the carrier contains no row for arbitrary subsequence selection, trichotomy, open-cover compactness, or ambient completeness, those principles remain outside the handoff. □

Theorem 2498.5 (RealBoundedSequence uniform-Cauchy subsequence boundary)**.** *Any uniform-Cauchy or subsequence-compactness consumer of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, E, H, C, P, N)$$

must read the finite bounded-source route through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows before it can export a subsequence or Cauchy-window conclusion. Thus the consumer receives $S, B, W, R, E, H, C, P, N$ as displayed packet data, but it receives no selected subsequence, quotient stream, trichotomy row, open-cover compactness proof, or ambient Real-completeness coordinate.*

Proof. Project the carrier of Definition 3451.1. The rows $S, B, W, R, E, H, C, P, N$ are the only displayed source, bound, window, readback, seal, transport, replay, provenance, and local naming coordinates.

Apply the NameCert obligations of Definition 3451.2. They force each bounded-sequence read to pass through the source row, bound witness, finite StreamName window, RegSeqRat readback, terminal Real seal, and structural support.

Use the Real-seal handoff of Definition 3451.3 and then the monotone-subsequence boundary of Definition 3451.4. The same route is the only input available to a uniform-Cauchy or subsequence-compactness consumer, so any exported subsequence or Cauchy-window conclusion is read through that finite packet.

Since no selected subsequence, quotient stream, trichotomy, open-cover compactness, or ambient Real-completeness row occurs in the carrier, none of those principles is exported by the boundary. □

Theorem 2498.6 (RealBoundedSequence compact-window handoff)**.** *Every compact-window, Bolzano-Weierstrass, or monotone-subsequence consumer of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, E, H, C, P, N)$$

must first read the displayed rows $S, B, W, R, E, H, C, P, N$ *through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Only after that finite packet is exposed may the consumer pass the bounded-source data to* $\mathsf{BolzanoWeierstrass}^{\uparrow}$. *The handoff refuses selected subsequences, quotient stream equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 3451.1. The rows $S, B, W, R, E, H, C, P, N$ are the only sequence source, bound witness, finite window, regular readback, terminal Real seal, transport, replay, provenance, and local naming coordinates.

Apply the NameCert obligations of Definition 3451.2. They require a downstream bounded-source read to expose the StreamName windows, RegSeqRat readback, dyadic tolerance ledger, and terminal Real seal before the packet is available to any consumer.

Use the Real-seal handoff of Definition 3451.3. It prevents a compact-window route from bypassing W, R, E or replacing them by host sequence data.

Finally apply the monotone-subsequence boundary of Definition 3451.4. The same displayed packet is the only input available to $\mathsf{BolzanoWeierstrass}^{\uparrow}$ and to monotone-subsequence consumers, while the refused subsequence, quotient, trichotomy, open-cover, and completeness principles have no carrier row. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RealBoundedSequence}^{\uparrow}$ packet is generated from a displayed sequence source, a bound witness row, finite StreamName windows, RegSeqRat readback, terminal Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3451.1, Definition 3451.2, Definition 3451.3, and Definition 3451.4 seed the finite bounded-sequence source consumed by Bolzano-Weierstrass and monotone-subsequence routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealBoundedSequenceUp.RealBoundedSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host sequence convergence, arbitrary subsequence choice, quotient stream equality, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bound-ledger exactness, sequence-window stability, Real-seal nonescape, and downstream handoff for BEDC.Derived.RealBoundedSequenceUp.

Chapter 2499

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ packet records the finite criterion by which a regular rational name is admitted to the completion-facing Real seal. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ rows of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus schedule of chapter 1438, the sequence-limit handoff of chapter 1413, the completion-admission boundary of chapter 1476, the functorial completion surface of chapter 3639, the recursor route of chapter 3628, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a displayed NameCert criterion for L10 completion consumers, not a quotient completion, selected representative, countable-choice schedule, host sequence theorem, trichotomy split, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Theorem 2499.1 (RegularCauchyCompletionCriterion carrier)**.** *A* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, M, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular rational readback inspected by the completion criterion.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the source name is observed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested dyadic radius, slack cells, and comparison budget used by the regular readback.*
4. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule row naming the displayed tail threshold.*
5. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row reached only after R, W, D, M have been displayed.*
6. *Q is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary row that admits the regular Cauchy route to the completion-facing* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, modulus schedule, sequence-limit handoff, completion boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from window requests through dyadic tolerance, regular readback, modulus schedule, sequence-limit handoff, and completion boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row naming the displayed criterion and its permitted completion-facing reads.*

The classifier compares accepted packets only through $R, W, D, M, L, Q, H, C, P, N$. It has no coordinate for quotient stream equality, a selected completion representative, countable choice over tail indices, host convergence, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier is the displayed L10 completion-neighbour packet. The first six rows record the analytic route from regular rational readback through finite StreamName windows, dyadic tolerance, modulus data, sequence-limit handoff, and completion admission.

The remaining rows are structural. They transport, replay, source, and name exactly that route, with the Bishop completion functor and completion recursor recorded only as provenance-facing consumers. No field remains for quotient completion, selected representatives, countable choice, host convergence, trichotomy, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionTasteGate`
`single carrier alignment`

Theorem 2499.2 (RegularCauchyCompletionCriterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier $K = (R, W, D, M, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary, transport, replay, provenance, and local naming row;*
2. *finite-window exactness, requiring every regular Cauchy observation to read W before R or M is available;*
3. *dyadic tolerance exactness, requiring the comparison budget D to be read before the modulus schedule M, sequence-limit handoff L, or completion boundary Q is exposed;*
4. *completion-boundary handoff, requiring Q to remain downstream of R, W, D, M, L and upstream only of the structural rows H, C, P, N;*
5. *ledger refusal, excluding quotient completion, selected representatives, countable choice over thresholds, host sequence convergence, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is tuple projection from Definition 3452.1. The accepted packet displays exactly the regular readback R, finite windows W, dyadic ledger D, modulus schedule M, sequence-limit handoff L, completion boundary Q, and the structural rows H, C, P, N.

The analytic route first reads the finite windows W and the tolerance ledger D. The regular rational readback R and the convergence-modulus schedule M are therefore exposed only as displayed finite data.

The handoff rows L and Q are downstream of that finite route. The $\mathsf{RealSequenceLimit}^{\uparrow}$ row receives the displayed sequence-limit comparison, and the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ row records the completion-admission boundary before any $\mathsf{Real}^{\uparrow}$-facing consumer can read the seal.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no row for quotient completion, selected representatives, countable choice over thresholds, host convergence, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionNameCertObligat
↪ variant BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionCarrier
namecert obligations

Theorem 2499.3 (RegularCauchyCompletionCriterion Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier* $K = (R, W, D, M, L, Q, H, C, P, N)$ *must first read the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{CauchyConvergenceModulus}^{\uparrow} \longrightarrow \mathsf{RealSequenceLimit}^{\uparrow} \longrightarrow \mathsf{C}$$

The handoff exposes only finite windows, dyadic tolerances, regular readback, modulus schedule, sequence-limit comparison, completion-admission boundary, componentwise transport, replay, provenance, and local naming.

Proof. Start with the carrier of Definition 3452.1. Its $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are the only source of inspected windows and tolerance data.

The regular readback and modulus rows consume those displayed data. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ row M cannot be replaced by a host sequence predicate or by an unbounded tail search.

Next the $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff L and $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ boundary Q are read in order. The $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ and $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ rows can consume this boundary only as displayed completion data, so the completion-facing consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only after the finite route has been displayed.

The structural rows H, C, P, N preserve and name the same handoff. Hence the exported surface is exactly the finite route in the statement. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionRealSealHandoff

Theorem 2499.4 (RegularCauchyCompletionCriterion ledger refusal)**.** *For every accepted* RegularCauchyCompletio *packet, the completion-facing ledger refuses quotient stream equality, quotient completion, selected representatives, countable choice over tail thresholds, host sequence convergence, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer requesting one of those objects must leave the displayed carrier route* $R, W, D, M, L, Q, H, C, P, N$*.*

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionLedgerRefusal

Proof. Inspect the carrier fields in Definition 3452.1. They record only the regular readback, finite windows, dyadic tolerance ledger, modulus schedule, sequence-limit handoff, completion boundary, completion functor and recursor provenance, transport, replay, provenance, and local naming rows.

The NameCert obligations of Definition 3452.2 make those rows exhaustive. No additional row stores a quotient, selected representative, choice schedule, host convergence predicate, trichotomy split, host equality proof, or ambient completeness witness.

Therefore any such request cannot be answered by the packet. It is refused at the ledger boundary before a $\mathsf{Real}^{\uparrow}$-facing consumer can treat the displayed completion criterion as a quotient completion theorem. □

Theorem 2499.5 (RegularCauchyCompletionCriterion transport stability)**.** *Let $K = (R, W, D, M, L, Q, H, C, P, N$ and $K' = (R', W', D', M', L', Q', H', C', P', N')$ be accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *packets. If componentwise* hsame *transport identifies the rows $R, W, D, M, L, Q, H, C, P, N$ with $R', W', D', M', L', Q', H$ then the finite-window route, dyadic tolerance ledger, modulus schedule,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary, replay, provenance, and local* NameCert *row are preserved. The transported packet still refuses quotient completion, selected representatives, countable choice over thresholds, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project both carriers through Definition 3452.1. The assumed componentwise hsame transport aligns exactly the displayed rows $R, W, D, M, L, Q, H, C, P, N$ and their primed counterparts.

Apply the NameCert obligations of Definition 3452.2. Finite-window exactness, dyadic tolerance exactness, and completion-boundary handoff are all phrased in those rows, so the transport cannot move the route outside the displayed packet.

Finally use the Real-seal handoff and ledger refusal theorems, Definition 3452.3 and Definition 3452.4. They identify the terminal read and the refused coordinates, so the transported packet preserves both the completion-facing route and the same boundary exclusions. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionTransportStabil`

Theorem 2499.6 (RegularCauchyCompletionCriterion obligation closure package)**.** *The carrier, NameCert obligations, Real-seal handoff, ledger refusal, and transport stability results for* RegularCauchyCompletionC *assemble the obligation-closure surface over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, CauchyConvergenceModulu $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The surface closes exactly the finite criterion rows $R, W, D, M, L, Q, H, C, P, N$ and does not claim quotient completion, selected representatives, countable choice, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionObligationClosu`

Proof. Start with the carrier and obligation rows in Definition 3452.1 and Definition 3452.2. They expose every analytic and structural coordinate of the packet and make the local NameCert surface exhaustive.

Next apply the Real-seal handoff and ledger refusal theorems. The handoff theorem routes any completion-facing read through the displayed StreamName, DyadicRatCore, RegSeqRat, modulus, sequence-limit, and completion-boundary rows, while the refusal theorem removes quotient, choice, host, and ambient-completeness exits from the package.

Finally apply transport stability from Definition 3452.5. Since componentwise hsame preserves the displayed rows and their replay and provenance data, the package is stable as an obligation-closure surface over the named BEDC primitives. □

Theorem 2499.7 (RegularCauchyCompletionCriterion scoped factorization)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *packet factors through the displayed rows of Definition 3452.1 before it may export a local* NameCert *surface. Concretely, the read must pass through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionC}}$ *row.*

The factorization is scoped to those rows. It does not supply quotient completion, selected representatives, countable choice over thresholds, host sequence convergence, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier and NameCert obligations of Definition 3452.1 and Definition 3452.2. They make the rows $R, W, D, M, L, Q, H, C, P, N$ exhaustive for any accepted packet.

Apply the Real-seal handoff of Definition 3452.3. It orders the completion-facing route through StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, convergence modulus, sequence-limit handoff, completion boundary, completion functor, completion recursor, and the terminal Real seal before a consumer can read completion data.

The ledger refusal of Definition 3452.4 removes quotient, choice, host, trichotomy, and ambient-completeness exits from that route. Transport stability in Definition 3452.5 then preserves the same finite rows under componentwise hsame and displayed Cont replay.

Finally use the obligation-closure package of Definition 3452.6. It repacks the displayed route with Pkg provenance and the local NameCert row, so every completion-facing read factors through exactly the named scope. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionScopeFactorizat

Theorem 2499.8 (RegularCauchyCompletionCriterion public export)**.** *Every accepted* RegularCauchyCompletionC *packet exports exactly the public completion-criterion surface*

$$R, W, D, M, L, Q, H, C, P, N$$

to Real-facing consumers: regular rational readback, finite StreamName windows, dyadic tolerance ledger, convergence-modulus schedule, RealSequenceLimit handoff, CauchyCompletionCriterion boundary, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row. It exports no quotient completion, selected representative, countable choice, trichotomy, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge certificate.*

Proof. Start with the carrier of Definition 3452.1. It exposes exactly the rows $R, W, D, M, L, Q, H, C, P, N$ and has no additional public coordinate for a quotient, representative selector, choice schedule, trichotomy split, host equality proof, ambient completeness witness, or bridge certificate.

Apply the NameCert obligations of Definition 3452.2. They force every accepted read to pass through the finite-window exactness row, dyadic tolerance exactness row, completion-boundary handoff, structural transport, replay, provenance, and local naming row.

The Real-seal handoff of Definition 3452.3 identifies the Real-facing order of the exported rows. The ledger refusal and transport stability theorems, Definition 3452.4 and Definition 3452.5, remove the forbidden exits and preserve the same public surface under componentwise hsame.

Finally use the obligation package and scoped factorization of Definition 3452.6 and Definition 3452.7. They repack the carrier, obligations, Real-seal handoff, ledger refusal, and transport stability into the displayed completion-criterion route, so the public export is exactly $R, W, D, M, L, Q, H, C, P, N$. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionPublicExport

Theorem 2499.9 (RegularCauchyCompletionCriterion public export checked route)**.** *The public export of Definition 3452.8 is exactly the already-scoped finite route through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BishopCauchyCom $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row. In particular, every public Real-facing read factors through the same rows* $R, W, D, M, L, Q, H, C, P, N$ *that support the carrier, NameCert obligations, Real-seal handoff, ledger refusal, transport stability, obligation package, and scoped factorization theorems.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionPublicExportChe`

Proof. Start with the carrier and NameCert obligations of Definition 3452.1 and Definition 3452.2. They identify the finite rows $R, W, D, M, L, Q, H, C, P, N$ and make the local certificate surface exhaustive.

Apply the Real-seal handoff, ledger refusal, and transport stability theorems, Definition 3452.3, Definition 3452.4, and Definition 3452.5. These results order the public read through the RegSeqRat, StreamName, DyadicRatCore, convergence-modulus, sequence-limit, completion-boundary, completion-functor, completion-recursor, and Real rows while excluding quotient, choice, host, trichotomy, and ambient-completeness exits.

Finally use the obligation package and scoped factorization of Definition 3452.6 and Definition 3452.7. They repack the same route with componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row, so the public export theorem is precisely the checked finite route already isolated by the scoped surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ packet is generated from a RegSeqRat readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a CauchyConvergenceModulus schedule, a RealSequenceLimit handoff, a CauchyCompletionCriterion boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 3452.1, Definition 3452.2, Definition 3452.3, Definition 3452.4, Definition 3452.5, Definition 3452.6, and Definition 3452.7 bind the finite regular-Cauchy completion criterion route through RegSeqRat, StreamName, DyadicRatCore, CauchyConvergenceModulus, RealSequenceLimit, CauchyCompletionCriterion, BishopCauchyCompletionFunctor, CauchyCompletionRecursor, Real rows, hsame transport, Cont replay, Pkg provenance, and local NameCert transport before any completion-facing consumer reads the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, host sequence convergence, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a checked export theorem for the same RegSeqRat, StreamName, DyadicRatCore, modulus, sequence-limit, completion-boundary, recursor, Real, transport, replay, provenance, and local NameCert primitives.

Chapter 2500

A Concrete Naming Certificate for $\mathsf{ConstructiveDiniModulus}^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ packet records the finite chain by which a monotone sequence of displayed real-valued functions on a compact metric source receives one uniform tolerance modulus. It sits after the compact finite-net route of chapter 129, the pointwise continuity surface of chapter 130, the compact uniform-modulus handoff of chapter 3444, the real-name seal of chapter 38, the finite-window readbacks of chapter 301 and chapter 126, and the dyadic tolerance ledger of chapter 291. The chapter names the finite BEDC route from pointwise monotone convergence data to a displayed Dini modulus without importing open-cover compactness, selected subcovers, choice, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2500.1 (Constructive Dini modulus carrier)**.** *A* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, L, N, Q, W, S, R, E, G, M, H, C, P, A)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the finite-net rows used by the modulus extraction.*
2. *F is the monotone real-function sequence row, listing the inspected function names, their pointwise order comparisons, and their target* $\mathsf{Real}^{\uparrow}$ *value rows.*
3. *L is the pointwise-limit row, naming the displayed limit function and the finite pointwise tail requests that compare $F_n(x)$ with $L(x)$.*
4. *N is the compact finite-net row selected from K for the current tolerance.*
5. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *continuity ledger for the finitely many functions and the limit row inspected at the net centers.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for function values, limit values, and net-center observations.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W.*
8. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested tolerance, the smaller comparison tolerances, and the finite dyadic descent used by the chain.*

9. *E is the finite chain comparison row. It lists the net-center tail index, monotonicity step, continuity transfer, and target estimate used for each displayed center.*
10. *G is the global modulus row obtained by taking the finite maximum of the center indices recorded in E.*
11. *M is the exported Dini-modulus witness for the monotone sequence and limit row.*
12. *H records componentwise* hsame *transport for $K, F, L, N, Q, W, S, R, E, G, M$ and the administrative rows.*
13. *C records displayed* Cont *replay from compact finite-net data through pointwise-limit windows, dyadic tolerances, finite chain comparisons, and the exported modulus.*
14. *P is the* Pkg *provenance over* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *A is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$ *row.*

The carrier exposes only monotone real-function sequence rows, pointwise-limit rows, compact finite-net rows, finite chain comparison rows, dyadic tolerance ledgers, regular rational and stream-name finite windows, componentwise transport, replay, provenance, and local naming. It has no coordinate for open-cover compactness, selected subcovers, choice, quotient equality, trichotomy, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusCarrier`
Lean4 ✓ `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusTasteGate single carrier alignment`

Theorem 2500.2 (Constructive Dini finite-chain modulus)**.** *For every accepted* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier*

$$D = (K, F, L, N, Q, W, S, R, E, G, M, H, C, P, A),$$

the exported row M is a finite Dini-modulus witness for the monotone sequence row F and the pointwise-limit row L at the tolerance recorded by R. Every consumer of M must factor through the displayed route

$$K, F, L, R \longmapsto N, Q, W, S, E \longmapsto G, M \longmapsto H, C, P, A.$$

Thus compact finite nets, pointwise limit windows, $\mathsf{ContinuousMap}^{\uparrow}$ *continuity ledgers,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance descent,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readbacks, finite chain comparisons, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *ledger suffice to export the constructive Dini modulus.*

Proof. Project the carrier of Definition 3454.1. The tolerance ledger R determines the comparison scale, while K supplies the finite net row N at the source scale consumed by the chain comparison.

At each center of N, read the pointwise-limit row L through the finite windows W and the regular rational readback S. The monotone function sequence row F supplies the ordered tail comparisons, and Q supplies the continuity rows needed to move from centers to neighboring source points.

The chain row E records the only comparison route used by the packet. It combines the center tail estimate, monotonicity, continuity transfer, and the dyadic tolerance descent of R into a displayed estimate at the requested tolerance.

Take the finite maximum of the center indices in E to obtain G, and read M as the exported modulus row. Componentwise transport in H preserves the compact source, function sequence, limit, window, tolerance, comparison, and modulus rows, while C replays the same route for downstream consumers.

Finally P and A record provenance and local naming. Since no carrier coordinate contains an open cover, selected subcover, choice functional, quotient equality, or ambient completeness principle, no consumer can extract any of those stronger readings from the modulus witness. □

Theorem 2500.3 (Constructive Dini modulus obligation lattice)**.** *Every exported modulus row of an accepted* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier is read in the dependency order*

$$K, N \longmapsto L, W \longmapsto S \longmapsto R \longmapsto E \longmapsto G \longmapsto A.$$

In this order, K, N *expose the compact finite-net source,* L, W *expose pointwise limit windows,* S *exposes the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* R *exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance descent,* E *records the finite comparison chain,* G *records the finite maximum row, and* A *supplies the local* $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$ *surface. No consumer may read the exported modulus* M *before these rows have been displayed.*

Proof. Project the carrier of Definition 3454.1. The rows K, N provide the compact finite-net source before any pointwise or tolerance comparison is available.

Apply Definition 3454.2. The pointwise limit row L is inspected through the finite windows W, and those windows are read back through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ row S.

The dyadic tolerance ledger R then fixes the descent used by the finite comparison chain E. Only after the centerwise indices in E have been displayed can the finite maximum row G be read.

Finally H, C, P, A transport, replay, source, and name the same route. The exported row M is therefore downstream of the compact finite net, pointwise windows, stream and regular-rational readback, dyadic descent, finite chain, finite maximum, and local NameCert rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, monotone real-function sequence rows, pointwise-limit rows, ContinuousMap continuity ledgers, StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, finite chain comparison rows, a finite maximum row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3454.1 and Definition 3454.2 seed the finite BEDC route from monotone pointwise convergence on compact finite-net data to a displayed Dini modulus.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, choice, quotient equality, host function extensionality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone sequence visibility, pointwise-limit windows, compact finite-net extraction, finite chain comparison, dyadic tolerance descent, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveDiniModulusUp.

Chapter 2501

A Concrete Naming Certificate for $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$

The $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ packet realizes the Poincare disk reading of chapter 9463 as finite BEDC data. It reads visible phase coordinates through chapter 39, disk coordinates and boundary rows through chapter 377, reversible transport through chapter 219, flow-style evolution through chapter 195, and boundary-circle consumers through chapter 58. The packet is a representation carrier over displayed histories, transports, flows, and provenance; it is not a host hyperbolic plane, not a global observer clock, not a physical cosmology, and not an internal point named T.

Definition 2501.1 (Poincare disk phase-transport carrier)**.** *A* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N)$$

with the following displayed rows:

1. *L is the focused finite multi-*Hist *ledger whose visible records are being represented.*
2. *Z is the* $\mathsf{Complex}^{\uparrow}$ *phase-coordinate row used for displayed amplitude, direction, or phase readback.*
3. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row, including the finite product-metric radius ledger and the optional* $\mathsf{S}^{1\uparrow}$ *boundary consumer row.*
4. *B is the boundary-far-end row, read only as a representation boundary over the disk packet and never as an interior object.*
5. *Γ is the* $\mathsf{UnitaryGroup}^{\uparrow}$ *reversible-transport row, recording classifier-preserving movement of the visible phase or disk surface.*
6. *V is the* $\mathsf{DynSystem}^{\uparrow}$ *flow row for finite evolution of the displayed disk-coordinate records.*
7. *F is the fiber and provenance row, retaining the hidden-side source distinctions behind each visible disk or phase readout.*
8. *O is the observer-focus row tying the disk surface back to the local ledger L rather than to a global observer or global clock.*
9. *A is the audit-refusal row excluding internal* T*, host hyperbolic geometry, physical light-speed derivation, global universe digest, and subject-substance readings.*

10. *H records componentwise* hsame *transport for the ledger, phase, disk, boundary, reversible-transport, flow, fiber, focus, and audit rows.*
11. *C records displayed* Cont *replay through finite disk coordinates, reversible transport, flow read-back, fiber provenance, and refusal rows.*
12. *R is the* Pkg *provenance over* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}^{\uparrow}}$ *row.*

The carrier compares packets only through these rows. It contains no coordinate for a completed Poincare disk, no host hyperbolic metric, no quotient boundary, no global physical evolution law, no subject-substance carrier, and no object-level T *inside the disk.*

Theorem 2501.2 (Poincare disk phase-transport ledger)**.** *For every accepted* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *carrier*

$$P = (L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N),$$

every disk-phase transport read factors through the finite route

$$L, Z, U \longmapsto B, \Gamma, V, F, O, A \longmapsto H, C, R, N.$$

The route exports visible disk-coordinate, reversible-transport, flow, fiber-provenance, observer-focus, and refusal data, while refusing host hyperbolic geometry, quotient boundary equality, global observer time, physical light-speed derivation, internal T*, and subject-substance readings.*

Lean4 ✓ `BEDC.Derived.PoincareDiskPhaseTransportUp.PoincareDiskPhaseTransportTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3455.1. The visible surface begins with the local ledger L, the complex phase row Z, and the disk-coordinate row U.

The representation rows are then read in order. The boundary row B records a far-end representation boundary, Γ supplies reversible classifier-preserving transport, V supplies finite flow-style evolution, F retains the fiber provenance behind the readout, O localizes the surface to the focused ledger, and A records the refused readings.

Transport and replay do not add new geometry. The rows H and C preserve and replay the same finite coordinates, while R and N record provenance and local naming.

Since the packet has no coordinate for a completed disk geometry, quotient boundary, global clock, physical constant, subject-substance, or internal T, no consumer of the ledger can export those stronger readings. □

Falsifiable prediction: Removing the fiber-provenance row or the audit-refusal row lets a disk-phase consumer identify visible disk-coordinate equality with source equality or with an internal T reading, so the packet is no longer accepted.

Independence witness: complex, unitdisk, unitarygroup, dynsystem, s1: none is bijective to $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ because the phase packet jointly carries visible complex coordinates, disk coordinates, optional boundary consumption, reversible transport, finite flow, fiber provenance, observer focus, refusal ledger, transport, replay, provenance, and local naming.

The Lean follow-up for this packet must provide explicit Nontrivial $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ and FieldFaithful $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ TasteGate instances. Those instances must keep

$L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}$.

Constructive story: A $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ packet is generated from a focused finite multi-Hist ledger, Complex phase coordinates, UnitDisk disk coordinates with optional SOne boundary consumption, UnitaryGroup reversible transport, DynSystem finite flow rows, fiber provenance, observer focus, refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3455.1 and Definition 3455.2 seed the finite disk-coordinate, reversible-transport, flow, fiber-provenance, and refusal route for the hyperbolic phase representation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Poincare disk geometry, host hyperbolic metric, quotient boundary equality, global observer time, physical light-speed derivation, internal T, subject-substance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-coordinate visibility, reversible transport, flow replay, fiber-provenance preservation, refusal exactness, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.PoincareDiskPhaseTransportUp.

Chapter 2502

A Concrete Naming Certificate for $\mathsf{MaschkeTheorem}^{\uparrow}$

Following Maschke's theorem for finite-group representations, the $\mathsf{MaschkeTheorem}^{\uparrow}$ packet records a finite averaging-projector and complement-decomposition ledger. It reads the group source of chapter 41, the vector-space source of chapter 47, the character-theory and representation-ring rows of chapter 245 and chapter 246, and the orthogonality surface of chapter 1603. The packet names only the displayed finite semisimplicity route; it is not a host representation category, not a quotient-subrepresentation construction, and not a global semisimplicity theorem.

Definition 2502.1 (Maschke theorem carrier)**.** *A* $\mathsf{MaschkeTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, V, R, X, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Group}^{\uparrow}$ *source row supplying the group operation, unit, inverse, and finite averaging support;*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row supplying the represented vector carrier and its addition, scalar, and zero rows;*
3. *R is the* $\mathsf{RepresentationRing}^{\uparrow}$ *row recording the representation input and the direct-sum handoff surface;*
4. *X is the invariant-subspace row, displayed as a carried subobject of the represented vector row rather than as a quotient representation;*
5. *A is the averaging-projector row, storing the finite group average used to make a raw complement map invariant;*
6. *Q is the complement row, recording the kernel-image decomposition exposed by the averaged projector;*
7. *H is the componentwise* hsame *transport for G, V, R, X, A, Q;*
8. *C is the* Cont *replay row for the averaging and complement ledgers;*
9. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a host representation category, a quotient subrepresentation carrier, a selected basis, or a global semisimplicity theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.MaschkeTheoremUp.MaschkeTheoremTasteGate single carrier alignment`

Theorem 2502.2 (Maschke theorem NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$ *surface for the carrier of Definition 3456.1 consists of the finite obligations that admit the group source G, the vector source V, the representation row R, the invariant-subspace row X, the averaging projector A, the complement row Q, and the transport, replay, provenance, and local naming rows H, C, P, N. These obligations export no host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem outside $G, V, R, X, A, Q, H, C, P, N$.*

Proof. Project the carrier rows of Definition 3456.1. The group row is read through chapter 41, while the vector row is read through chapter 47.

Next read the representation surface through chapter 246 and the character-theory support through chapter 245. These rows account for R and for the finite representation-theoretic evidence consumed by the averaging ledger.

Finally inspect the remaining rows. The invariant subspace X, averaging projector A, and complement row Q are preserved only by the displayed hsame, Cont, Pkg, and NameCert rows H, C, P, N. Since the carrier contains no further coordinate, the excluded host category, quotient carrier, basis selection, and global semisimplicity data cannot be exported. □

Theorem 2502.3 (Maschke averaging-projector route)**.** *Every complement-facing consumer of an accepted* $\mathsf{MaschkeTheorem}^{\uparrow}$ *packet must pass through*

$$G \longmapsto V \longmapsto R \longmapsto X \longmapsto A \longmapsto Q \longmapsto H, C, P, N.$$

The averaging row A is read as a finite $\mathsf{Group}^{\uparrow}$ *average over the represented* $\mathsf{VecSpace}^{\uparrow}$ *row and the representation row R, and the complement row Q is read only after that projector row has been displayed. The route exports no host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem.*

Proof. Begin with the group and vector sources. The finite averaging support is supplied by chapter 41, and the linear target of the average is supplied by chapter 47.

Then use the representation rows. The representation-ring certificate of chapter 246 identifies the action row R, while the character-theory and Schur-orthogonality surfaces of chapter 245 and chapter 1603 supply the finite representation-theoretic ledger read by the projector route.

The complement row is downstream of the averaging row. Once A is displayed, Q records only the finite complement decomposition and the rows H, C, P, N replay and name that route. No listed sibling chapter supplies a hidden host category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem, so the consumer cannot receive those objects from this packet. □

Theorem 2502.4 (Maschke representation-ring handoff)**.** *Every* $\mathsf{RepresentationRing}^{\uparrow}$*-facing read of an accepted* $\mathsf{MaschkeTheorem}^{\uparrow}$ *packet receives only the finite handoff from the group and vector rows to the representation row R, the invariant row X, the averaging row A, the complement row Q, and the structural rows H, C, P, N. The handoff may record a displayed direct-sum decomposition row, but it does not create a host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem outside the finite packet.*

Proof. First expose the representation-ring dependency. The handoff begins at chapter 246, whose rows name representation addition and direct-sum bookkeeping over the underlying group and vector surfaces.

Next attach the finite Maschke rows. The invariant row X and the averaging projector row A produce the complement row Q only through the displayed packet route, with chapter 1603 available as a finite orthogonality-side ledger rather than as a basis selector.

Finally repack the handoff through H, C, P, N. These rows transport, replay, source, and name the same finite route for $\mathsf{RepresentationRing}^{\uparrow}$ consumers. Since no quotient carrier, host category, selected basis, or global semisimplicity coordinate is present, the handoff remains confined to the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MaschkeTheorem}^{\uparrow}$ packet is generated from a finite $\mathsf{Group}^{\uparrow}$ source row, a $\mathsf{VecSpace}^{\uparrow}$ source row, a $\mathsf{RepresentationRing}^{\uparrow}$ row, an invariant-subspace row, an averaging-projector row, a complement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3456.1, Definition 3456.2, Definition 3456.3, and Definition 3456.4 seed the finite averaging-projector and complement-decomposition route for finite-group representation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MaschkeTheoremUp.MaschkeTheoremTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close a host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem.

Upgrade path: Obligation closure requires checked group, vector, representation-ring, invariant-subspace, averaging-projector, complement, transport, replay, provenance, and local naming rows for the displayed packet.

Chapter 2503

A Concrete Naming Certificate for $\mathsf{NestedCauchyRealSection}^{\uparrow}$

Bishop–Bridges constructive analysis treats nested Cauchy sections as finite approximation data before any completed real object is read. The $\mathsf{NestedCauchyRealSection}^{\uparrow}$ packet names that Real and Completion phase surface over the nested-interval carrier of chapter 320, the dyadic tolerance ledger of chapter 291, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It packages nested Cauchy section data for downstream completion consumers without asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2503.1 (Nested Cauchy real section carrier)**.** *A* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval row, containing the finite interval chain, endpoint order data, and the diameter bounds consumed from chapter 320.*
2. *T is the cofinal tail-window row, assigning each requested tolerance a finite interval index and the later windows through which the section may be inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing only finite stream windows before any regular rational readback is available.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reached from the stream windows and the nested interval endpoints.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic comparison budgets for endpoint width, tail inclusion, and section readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after I, T, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for nested intervals, tail windows, stream observations, regular readback, dyadic tolerance ledgers, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Real or Completion request through T, D, W, R, I and then into the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedCauchyRealSection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, W, R, D, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.NestedCauchyRealSectionUp.NestedCauchyRealSectionUp`
Lean4 ✓ `BEDC.Derived.NestedCauchyRealSectionUp.NestedCauchyRealSectionTasteGate single carrier alignment`

Theorem 2503.2 (Nested Cauchy real section NameCert obligations)**.** *For every accepted* NestedCauchyRealSectio *carrier $S = (I, T, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the nested interval row, cofinal tail windows, finite stream observations, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *tail-window exactness, requiring every Real or Completion tolerance request to expose T before reading W, R, D;*
3. *nested-interval compatibility, requiring the interval chain I and dyadic ledger D to control endpoint width, tail inclusion, and section readback;*
4. *Real-seal handoff, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row E to be read only after I, T, W, R, D have been displayed;*
5. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no quotient stream, selected limit, countable choice, host real equality, or ambient Real completeness.*

Proof. Carrier admission is projection from Definition 3458.1. The tuple displays the nested intervals, tail windows, stream windows, regular readback, dyadic ledger, Real seal, and structural rows.

Tail-window exactness follows from the placement of T. A tolerance request reaches finite observations and regular readback only through the displayed cofinal window, so no off-packet schedule can be used.

Nested-interval compatibility is the joint route through I and D. The interval chain supplies the nested endpoint data, while the dyadic ledger supplies the width and comparison budgets.

The Real seal is downstream of those rows. The replay row C reaches E only after I, T, W, R, D are visible, and H, P, N preserve and name that same route.

The excluded objects are absent from the carrier. Hence quotient streams, selected limits, countable choice, host real equality, and ambient Real completeness cannot appear as local NameCert obligations. □

Theorem 2503.3 (Nested Cauchy real section Real-completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *carrier factors through the* $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it exposes the local section certificate. The handoff exports finite interval, tolerance, observation, regular-readback, and Real-seal data, and exports no quotient stream, selected limit, countable choice, host real equality, or ambient Real completeness theorem.*

Proof. Start from an accepted packet $S = (I, T, W, R, D, E, H, C, P, N)$. Projection gives the nested interval row I, cofinal tail row T, finite stream observations W, regular rational readback R, dyadic tolerance ledger D, terminal Real seal E, and structural rows.

A completion-facing tolerance request is replayed through C. The replay fixes the tail window T, reads dyadic budgets from D, opens the stream window W, obtains regular rational readback from R, and checks the nested interval row I before exposing the Real seal E.

The structural rows add no external completion principle. They transport, replay, source, and name the same finite rows, so the read cannot export a quotient stream, a selected limit, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 2503.4 (Nested Cauchy real section choice-free Real seal)**.** *Every accepted* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *packet exposes its terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed nested interval, cofinal tail-window row, finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows. The exported Real-facing boundary contains no quotient stream, selected limit, countable-choice row, host real equality, or ambient Real-completeness coordinate.*

Proof. Project the carrier tuple $S = (I, T, W, R, D, E, H, C, P, N)$ of Definition 3458.1. The terminal Real seal is the row E, and the rows available before it are exactly the nested interval row I, cofinal tail-window row T, finite observation row W, regular rational readback row R, and dyadic tolerance ledger D.

Apply the Real-completion handoff theorem. It forces every completion-facing read to pass through I, T, W, R, D before exposing the local section certificate and the terminal Real seal.

The refused objects are absent from the carrier definition. Since no coordinate names a quotient stream, selected limit, countable-choice row, host real equality, or ambient Real-completeness theorem, the seal exported by E is choice-free in exactly the displayed packet sense. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedCauchyRealSection}^{\uparrow}$ packet is generated from nested interval rows, cofinal tail windows, finite $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3458.1, Definition 3458.2, and Definition 3458.3 seed the finite nested Cauchy section handoff surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exactness, nested-interval compatibility, dyadic tolerance stability, regular rational readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedCauchyRealSectionUp.

Chapter 2504

A Concrete Naming Certificate for $\mathsf{RealUniformCauchyName}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealUniformCauchyName}^{\uparrow}$ packet records a regular Cauchy real name together with the finite modulus route that makes its requested tolerances inspectable. It sits between the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RealCauchyModulus}^{\uparrow}$ schedule of chapter 1423, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives completion and rate-extraction consumers a named real-name surface without importing host limits, quotient streams, selected representatives, or ambient completeness.

Definition 2504.1 (RealUniformCauchyName carrier)**.** *A* $\mathsf{RealUniformCauchyName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, M, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected for the requested precision levels.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those windows, exposing the rational approximants and their regularity checks.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, recording the finite threshold chosen for each visible dyadic tolerance request.*
4. *Q is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *route that attaches the schedule M to the real-name readback rather than to a host modulus function.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, listing the inspected dyadic radii, comparison budgets, and refinement rows used by S, R, M, Q.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream windows, regular readbacks, modulus schedules, and dyadic ledger have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, modulus, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through $S, R, M, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealUniformCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by componentwise displayed transport through these rows. It has no coordinate for quotient stream equality, host sequence limits, countable-choice schedules, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealUniformCauchyNameUp.RealUniformCauchyNameUp`
Lean4 ✓ `BEDC.Derived.RealUniformCauchyNameUp.RealUniformCauchyNameTasteGate single carrier alignment`

Theorem 2504.2 (RealUniformCauchyName NameCert obligations)**.** *For every accepted* RealUniformCauchyName *carrier* $U = (S, R, M, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window access through* S*, regular rational readback through* R*,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule exposure through* M*,* $\mathsf{RealCauchyModulus}^{\uparrow}$ *schedule-to-real-name routing through* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through* D*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3459.1. The only rows available to the local naming surface are $S, R, M, Q, D, E, H, C, P, N$, so carrier admission supplies precisely the window, readback, modulus, tolerance, seal, transport, replay, provenance, and naming data.

The modulus route is split into M and Q. The row M records the finite threshold schedule for visible dyadic requests, while Q attaches that schedule to the real-name readback. Thus the route names a displayed Cauchy real name rather than a host modulus function.

Transport by H is componentwise, and replay by C follows the same displayed tolerance route. Since P, N source and name only this packet, no quotient stream, selected representative, countable-choice schedule, or ambient completeness datum is available to the NameCert surface. □

Theorem 2504.3 (RealUniformCauchyName seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{RealUniformCauchyName}^{\uparrow}$ *carrier factors through the ordered route*

$$S \longmapsto R \longmapsto M \longmapsto Q \longmapsto D \longmapsto E$$

together with H, C, P, N*. Consequently a completion or rate-extraction consumer may read the named real only through displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{RealCauchyModulus}^{\uparrow}$ *schedules, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the local structural rows.*

Proof. Begin with a requested terminal seal. By Definition 3459.1, the request first exposes the selected finite windows S and their regular rational readbacks R.

The tolerance request is then governed by the schedule row M and attached to the real-name route by Q. The dyadic ledger D records the inspected radii and comparison budgets before the terminal seal E is visible.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the same finite route, so the consumer receives no host limit, quotient stream equality, countable-choice selector, selected representative, trichotomy, independent real value, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Falsifiable prediction: Every $\mathsf{RealUniformCauchyName}^{\uparrow}$ seal read factors through StreamName finite windows, RegSeqRat readbacks, CauchyModulus and RealCauchyModulus schedules, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, realcauchymodulus, real: RealUniformCauchyNam is not bijective to any listed sibling because it jointly carries the finite window family, regular rational readback, explicit Cauchy schedule, Real-facing modulus route, dyadic tolerance ledger, terminal Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformCauchyName}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, CauchyModulus and RealCauchyModulus schedules, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$, seed)

Scope closed: Definition 3459.1, Definition 3459.2, and Definition 3459.3 bind the finite regular Cauchy real-name certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host sequence limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-route exactness, tolerance-ledger accounting, componentwise classifier transport, and Real seal nonescape for BEDC.Derived.RealUniformCauchyNameUp.

Chapter 2505

A Concrete Naming Certificate for $\mathsf{ProofObligationGap}^{\uparrow}$

The $\mathsf{ProofObligationGap}^{\uparrow}$ packet realizes the MetaCIC subject-reduction obstruction of chapter 9386 as finite NameCert data. Its carrier is narrower than a confluence, normalization, or conversion endpoint: it packages named obligation rows, obstruction rows, refusal rows, and downstream handoff rows so that a subject-reduction consumer must see the gap surface before it reads any obstruction as usable evidence.

Falsifiable prediction: Every accepted $\mathsf{ProofObligationGap}^{\uparrow}$ route exposes the named obligation row before any downstream subject-reduction or consistency consumer can read the obstruction.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, normalformconsistencyseal, bounded-normal-conversion-bridge: the gap packet is not bijective to these siblings because it jointly carries named obligations, obstruction/refusal rows, and downstream handoff routes rather than a positive confluence, normalization, or conversion endpoint.

The $\mathsf{ProofObligationGap}^{\uparrow}$ carrier is read after the MetaCIC critical-path packet of chapter 765 and before the bounded conversion boundary of chapter 3439 can consume an obstruction row.

Definition 2505.1 (Proof-obligation gap carrier)**.** *A* $\mathsf{ProofObligationGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (O, B, R, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the named obligation row, carrying one of the four MetaCIC subject-reduction obligations isolated in section 9386.2.*
2. *B is the obstruction row, recording that the named obligation cannot be replaced by a blanket closed subject-reduction theorem at this boundary.*
3. *R is the refusal row, rejecting any route that tries to read the obstruction before the obligation name has been exposed.*
4. *M is the MetaCIC handoff row, routing accepted packets to downstream subject-reduction, consistency, critical-path, or audit consumers only after O, B, R have been displayed together.*
5. *H records componentwise* hsame *transport for obligation, obstruction, refusal, handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a downstream MetaCIC request through O, B, R, M.*

7. *P is the* Pkg *provenance over* $\mathsf{ProofObligationGap}^{\uparrow}$*, the MetaCIC open-problem frontier,* BHist, hsame, Cont, Pkg, *and* NameCert.

8. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$ *row.*

The classifier accepts exactly packets where a named obligation row is paired with its obstruction and refusal rows before any MetaCIC handoff is available. It has no coordinate for positive confluence, normalization, bounded conversion, closed subject reduction, or a consistency seal.

Theorem 2505.2 (Proof-obligation gap NameCert obligations)**.** *For every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *carrier $G = (O, B, R, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the named obligation, obstruction, refusal, MetaCIC handoff, transport, replay, provenance, and local naming rows;*

2. *obligation-first exposure, requiring O to be visible before any downstream consumer reads B;*

3. *refusal exactness, requiring R to reject obstruction-only reads and three-row subject-reduction shortcuts;*

4. *handoff confinement, requiring M to route only the displayed O, B, R surface to MetaCIC consumers;*

5. *structural non-escape, requiring H, C, P, N to preserve and name the same packet without exporting confluence, normalization, conversion, closed subject reduction, or consistency-seal evidence.*

Lean `TasteGate.lean` *will include explicit* `Nontrivial ProofObligationGapUp` *and* `FieldFaithful ProofObligationGapUp` *instances so the accepted and refused surfaces remain distinct and the row order O, B, R, M, H, C, P, N is preserved by faithful reads.*

Lean4 ✓ `BEDC.Derived.ProofObligationGapUp.ProofObligationGapTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3460.1. The accepted packet displays one named obligation row, one obstruction row, one refusal row, one handoff row, and the structural rows.

Obligation-first exposure follows from the classifier. The obstruction row B is accepted only together with the named obligation row O, so a downstream subject-reduction or consistency consumer cannot read B as a standalone discharge.

Refusal exactness is the role of R. It rejects routes that omit O, routes that treat the obstruction as a positive theorem, and routes that offer fewer than the named obligation surface required by section 9386.2.

The handoff row M then confines downstream reads to the displayed surface O, B, R. The structural rows H, C, P, N transport, replay, source, and name that packet, and they add no coordinate for confluence, normalization, conversion, closed subject reduction, or a consistency seal. □

Theorem 2505.3 (Proof-obligation gap MetaCIC handoff)**.** *Every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *handoff to a MetaCIC subject-reduction, consistency, critical-path, or audit consumer first exposes the named obligation row O, then the obstruction row B, then the refusal row R, and only then the handoff row M. Consequently the packet records the obstruction surface as a finite named gap and exports no positive confluence theorem, normalization endpoint, bounded conversion bridge, closed subject-reduction theorem, or consistency seal.*

Proof. Project an accepted carrier $G = (O, B, R, M, H, C, P, N)$. The only consumer-facing handoff coordinate is M, and it is displayed after the named obligation, obstruction, and refusal rows.

Replay through C preserves that order. A downstream MetaCIC request must pass through O, B, R before it can read M, so the obstruction is always seen together with the obligation name and the refusal boundary.

The packet has no endpoint row for confluence, normalization, bounded conversion, closed subject reduction, or consistency sealing. Provenance and local naming therefore authenticate only the finite gap surface, not any positive theorem at those endpoints. □

Theorem 2505.4 (Proof-obligation gap MetaCIC sibling dependency)**.** *Every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *packet consumed by a MetaCIC critical-path or bounded-normal-conversion consumer must cite the named obligation row before it can hand off to chapter 765 or chapter 3439. This sibling dependency is substantive: those consumers can read the finite gap only after obligation-first exposure and refusal exactness, not as positive confluence evidence, normalization evidence, bounded-conversion evidence, or a closed subject-reduction endpoint.*

Proof. Project an accepted carrier $G = (O, B, R, M, H, C, P, N)$. The named obligation row O is displayed before the obstruction row B, the refusal row R, and the MetaCIC handoff row M.

A critical-path consumer may therefore cite chapter 765 only after reading O together with the obstruction and refusal boundary. A bounded-normal-conversion consumer may likewise cite chapter 3439 only after the same obligation-first exposure.

The dependency is not a positive endpoint. The rows B and R record obstruction and refusal exactness, while M, H, C, P, N preserve and name that finite gap surface. No row supplies confluence, normalization, bounded conversion, or closed subject reduction. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofObligationGap}^{\uparrow}$ packet is generated from named MetaCIC obligation rows, obstruction rows, refusal rows, downstream MetaCIC handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3460.1, Definition 3460.2, Definition 3460.3, and Definition 3460.4 seed the finite proof-obligation gap surface for MetaCIC subject-reduction obstruction consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close positive confluence, normalization, bounded conversion, closed subject reduction, consistency sealing, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, obligation-first exposure, refusal exactness, MetaCIC handoff confinement, transport, replay, provenance, local naming, Nontrivial ProofObligationGapUp, and Field-Faithful ProofObligationGapUp.

Chapter 2506

A Concrete Naming Certificate for $\mathsf{MeasurableFunction}^{\uparrow}$

Following Caratheodory and Bourbaki measurable maps, the $\mathsf{MeasurableFunction}^{\uparrow}$ packet records the finite preimage surface by which a source measurable-event family is routed into a target measurable-event family. It sits after the event rows of chapter 92, the generated-set boundary of chapter 3384, the $\mathsf{SetLike}^{\uparrow}$ set-membership carrier discipline, the terminal real-name seal of chapter 38, and the finite observation windows of chapter 126. The packet names measurable-map behavior as finite preimage ledgers, not as a host function space, quotient equality of functions, selected representatives, or ambient completed measure theory.

Definition 2506.1 (MeasurableFunction carrier)**.** *A* $\mathsf{MeasurableFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, A, B, G, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source measurable carrier row, read through* $\mathsf{Measure}^{\uparrow}$ *or through a* $\mathsf{BorelSet}^{\uparrow}$ *event source when the map is real-facing.*
2. *T is the target measurable carrier row, again read through* $\mathsf{Measure}^{\uparrow}$ *or the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set boundary.*
3. *A is the source event-family row, with membership and subset readback routed through* $\mathsf{SetLike}^{\uparrow}$*.*
4. *B is the target event-family row, with target membership readback routed through* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and finite* $\mathsf{StreamName}^{\uparrow}$ *observations when the target is a real or Borel source.*
5. *G is the graph-read row naming the finite source-to-target observation pairs exposed to the packet.*
6. *R is the preimage ledger assigning each displayed target event row in B to a displayed source event row in A and recording the finite membership readback for the graph row G.*
7. *H stores componentwise* hsame *transport for the source carrier, target carrier, event-family rows, graph row, preimage ledger, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay by which a consumer reads target events, preimage events, graph observations, and source memberships.*
9. *P is the* Pkg *provenance over* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, T, A, B, G, R, H, C, P, N$. It has no coordinate for a host function type, quotient equality of functions, almost-everywhere equality, completion of a measure, or a selected representative.

Lean4 (definition) `BEDC.Derived.MeasurableFunctionUp.MeasurableFunctionUp`

Theorem 2506.2 (MeasurableFunction NameCert obligations)**.** *For the carrier of Definition 3461.1, the local* $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, source-event admission, target-event admission, graph-read admission, preimage-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted measurable-map consumer must read a target event through B, its displayed preimage through R, and the source-event result through A before any* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *sibling row is exposed.*

Proof. Project the carrier $F = (S, T, A, B, G, R, H, C, P, N)$. The source and target carrier rows S, T supply the measurable contexts, while A and B supply the only event-family rows available to the packet.

Read a target event request through B. The preimage ledger R is the unique row that assigns that target event to a source event in A, and the graph row G supplies the finite membership readback tying source observations to target observations.

The structural rows add no host function object. The componentwise transport H, replay row C, provenance row P, and local naming row N preserve the same displayed route through the sibling certificates, so the listed rows exhaust the NameCert obligation surface. □

Theorem 2506.3 (MeasurableFunction simple preimage ledger)**.** *Every simple target-event read of an accepted* $\mathsf{MeasurableFunction}^{\uparrow}$ *packet factors through the finite ledger route*

$$B \longmapsto R \longmapsto A \longmapsto G \longmapsto H, C, P, N.$$

For real-facing Borel targets, the target membership observation reaches $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *only through the displayed* $\mathsf{BorelSet}^{\uparrow}$ *row in B and the graph-read row G. Thus preimage formation exports a named source event row, not a host inverse-image function, almost-everywhere quotient, completed measure, selected representative, or ambient function equality.*

Proof. Start with an accepted target-event request. The request is not a free subset of a host codomain; it is a displayed target event row in B, transported through the target carrier row T.

The preimage ledger R is then inspected. It names the corresponding source event row in A and records the graph-read membership comparison supplied by G. For real-facing Borel targets, this graph read is the only place where $\mathsf{Real}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ observations enter the route.

Finally replay the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite route over $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, and $\mathsf{SetLike}^{\uparrow}$. Since no further coordinate is present, the excluded host inverse-image, quotient, completion, representative-selection, and ambient equality objects cannot be exported. □

Falsifiable prediction: A $\mathsf{MeasurableFunction}^{\uparrow}$ consumer that obtains a host function equality, an almost-everywhere quotient class, a completed measure, or a selected representative without first reading the displayed preimage ledger R refutes the carrier surface.

Independence witness: measure, borelset, setlike, real, streamname: no listed sibling supplies the whole ten-row packet $F = (S, T, A, B, G, R, H, C, P, N)$, because each sibling contributes only

event-family, generated-set, membership, real-seal, or finite-window data while this chapter binds the measurable-map preimage ledger with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3461.1, Definition 3461.2, and Definition 3461.3 seed the finite measurable-map surface over source events, target events, graph reads, preimage ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close random variables, distributions, integrals, almost-everywhere equality, completed measures, quotient function spaces, selected representatives, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, preimage-ledger exactness, graph-read preservation, componentwise transport, and ledger-refusal proofs for BEDC.Derived.MeasurableFunctionUp.

Constructive story: A $\mathsf{MeasurableFunction}^{\uparrow}$ packet is generated from source and target measurable carrier rows, source and target event-family rows, a graph-read row, a preimage ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2507

A Concrete Naming Certificate for $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ packet records the finite comparison surface between directed Cauchy nets and Cauchy filter bases before any completion-facing consumer reads a Real seal. It sits between the Cauchy-net carrier of chapter 589, the Cauchy-filter carrier of chapter 333, the Moore–Smith completion packet of chapter 2190, the uniform completion route of chapter 1473, and the Real readback boundary of chapter 38. The packet is finite BEDC data: directed index rows, tail rows, filter-base rows, cofinal comparison rows, completion handoff rows, readback rows, transport, replay, provenance, and a local NameCert. It does not import quotient completions, selected limits, countable choice, or an ambient completed space.

Definition 2507.1 (Moore–Smith Cauchy-net/filter bridge carrier)**.** *A* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, T, F, K, U, M, S, R, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row naming the finite directed comparisons used by the net side;*
2. *T is the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row, recording eventual containment tests over the directed index surface D;*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *base row, recording finite base refinements and Cauchy diameter tests;*
4. *K is the cofinal net-filter comparison row, displaying the finite route from directed tails to filter bases and back to cofinal tail data;*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row, read only after D, T, F, K have been displayed;*
6. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row for metric-facing completion consumers;*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by completion-facing readback;*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same window and comparison data;*
9. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic comparison ledger used to bound the displayed tails, bases, and readbacks;*

10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the net, filter, cofinal, completion, and readback rows have been displayed;*
11. *H records componentwise* hsame *transport for all displayed rows;*
12. *C records displayed* Cont *replay from directed tails through filter bases, completion handoff, readback windows, and the Real seal;*
13. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$ *row.*

The public carrier exports the finite packet B and no quotient completion, selected subnet, chosen filter limit, countable-choice schedule, or ambient completeness datum.

Lean4 ✓ `BEDC.Derived.MooreSmithCauchyNetFilterBridgeUp.MooreSmithCauchyNetFilterBridgeTasteGate`
`single carrier alignment`

Theorem 2507.2 (Moore–Smith Cauchy-net/filter bridge NameCert obligations)**.** *For the carrier of Definition 3462.1, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (D, T, F, K, U, M, S, R, Y, E, H, C, P, N)$; directed-index admission through D; Cauchy-net tail admission through T; Cauchy-filter base admission through F; cofinal comparison through K; uniform and metric completion handoff through U and M; StreamName, RegSeqRat, DyadicRatCore, and Real readback through S, R, Y, E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The rows D, T, F, K expose the directed index surface, the Cauchy-net tails, the Cauchy-filter bases, and the cofinal comparison between them as visible coordinates of one finite packet.

The completion and readback rows are downstream. The uniform and metric completion handoffs U, M are read only after the net/filter comparison row is displayed, and the finite readback rows S, R, Y, E attach the StreamName, RegSeqRat, dyadic, and Real-facing reads to the same displayed route.

The structural rows exhaust the NameCert surface. Componentwise transport, displayed replay, provenance, and local naming are exactly H, C, P, N, so a consumer cannot obtain a quotient completion, selected subnet, chosen filter limit, countable-choice schedule, or ambient completion principle from the packet. □

Theorem 2507.3 (Moore–Smith Cauchy-net/filter bridge handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ *carrier factors through the displayed route*

$$D, T, F, K, U, M, S, R, Y, E, H, C, P, N$$

before reading any $\mathsf{Real}^{\uparrow}$ *seal. In particular, a consumer moving between chapter 589, chapter 333, chapter 2190, and chapter 1473 reads the cofinal net/filter rows before it can use the Real-facing readback of chapter 38.*

Proof. Begin with a completion-facing request. Carrier admission forces the directed index row D, Cauchy-net tail row T, Cauchy-filter base row F, and cofinal comparison row K to be visible before any completion handoff row appears.

The handoff rows U and M then carry only the displayed comparison route. They do not identify completion classes, choose a subnet, or select a filter limit; they pass the finite net/filter comparison to uniform and metric completion consumers.

Finally the readback rows S, R, Y, E and the structural rows H, C, P, N keep the Real seal downstream of the same displayed route. Since the packet has no coordinate for quotient completion, countable choice, selected limits, or ambient completeness, every completion-facing read factors through the named rows before it reaches $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ packet is generated from directed index data, Cauchy-net tail rows, Cauchy-filter base rows, a cofinal net-filter comparison, uniform and metric completion handoff rows, StreamName windows, RegSeqRat and DyadicRatCore readbacks, a RealUp seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3462.1, Definition 3462.2, and Definition 3462.3 bind the finite net/filter/cofinal route before uniform, metric, and Real readback consumers inspect the packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completions, selected limits, countable choice, or ambient completed-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal comparison, completion handoff, transport stability, replay, provenance, and local naming for BEDC.Derived.MooreSmithCauchyNetFilterBridgeUp.

Chapter 2508

A Concrete Naming Certificate for $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$

The $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ packet realizes the punctured WNA cube, CUN square, bridge-edge, and articulation-gate support graph of Definition 9462.1293 as finite NameCert data. Its carrier records the cube vertex rows, the deleted ACA row, the CUN square, the UAG and AGG bridge rows, adjacency ledgers, biconnected-component rows, transport, replay, provenance, and local naming. It sits beside the codon source-window carrier of chapter 2234, the DNA window boundary tile of chapter 3393, and the codon-window spectral inverse of chapter 2161. The packet is a finite combinatorial support certificate; it does not assert a biochemical mechanism, a selected evolutionary path, or a host-level graph quotient.

Definition 2508.1 (Punctured cube skeleton carrier)**.** *A* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (W, A, C, B, G, E, H, T, Q, R, N)$$

with the following displayed rows:

1. *W is the WNA cube row listing the eight codon vertices with first coordinate in* $\{\mathrm{U}, \mathrm{A}\}$*, middle coordinate in* $\{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}$*, and terminal coordinate* A*;*
2. *A is the puncture row deleting* ACA *and retaining the seven displayed WNA vertices;*
3. *C is the CUN square row over* $\{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\}$*;*
4. *B is the bridge-edge row carrying* $\{\mathrm{CUA}, \mathrm{UUA}\}$*,* $\{\mathrm{UAA}, \mathrm{UAG}\}$*, and* $\{\mathrm{AGA}, \mathrm{AGG}\}$*;*
5. *G is the articulation-gate row identifying the finite support graph induced by A, C, B;*
6. *E is the edge-decomposition ledger comparing the cube face, CUN square, bridge edges, and biconnected components;*
7. *H records componentwise* hsame *transport for the vertex, puncture, square, bridge, gate, and edge-decomposition rows;*
8. *T records displayed* Cont *replay from the support formula to the induced graph and component rows;*
9. *Q is the* Pkg *provenance over* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*
10. *R is the TasteGate requirement row for later* Nontrivial $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *and* FieldFaithful PuncturedCubeS *instances;*

11. N is the local $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$ *row.*

Theorem 2508.2 (Punctured cube skeleton NameCert obligations)**.** *For the carrier of Definition 3463.1, the local* $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$ *surface consists exactly of WNA cube admission, puncture admission, CUN square admission, bridge-edge admission, articulation-gate admission, edge-decomposition admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, TasteGate nontriviality and field-faithfulness requirements, and local naming. The future Lean-facing TasteGate surface must distinguish the puncture row A from the unpunctured cube row W and must preserve every field $W, A, C, B, G, E, H, T, Q, R, N$.*

Lean4 ✓ `BEDC.Derived.PuncturedCubeSkeletonUp.PuncturedCubeSkeletonTasteGate single carrier alignment`

Proof. Project the carrier. The rows W, A, C, B, G, E display the finite support formula, the puncture, the CUN square, the bridge edges, the articulation gate, and the induced edge-decomposition ledger.

Transport and replay are structural. The rows H, T, Q, N preserve and name the same finite graph support data without adding a host quotient or a biological mechanism.

The TasteGate requirement row R records the exact Lean obligations: nontriviality must separate the punctured packet from a collapsed graph, and field faithfulness must preserve every displayed vertex, puncture, bridge, gate, and decomposition row. These obligations are part of the carrier surface and do not assert that the Lean instances already exist. □

Theorem 2508.3 (Punctured cube skeleton boundary nonescape)**.** *Every consumer of an accepted* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier factors through*

$$W, A, C, B, G, E, H, T, Q, R, N$$

before reading the punctured cube skeleton as finite NameCert data. Consequently the consumer cannot erase the ACA *puncture, collapse the CUN square into the WNA cube, drop the bridge-edge rows, replace the articulation gate by a host graph quotient, or bypass the later* Nontrivial *and* FieldFaithful *TasteGate requirements.*

Proof. Start from the support rows. The WNA cube, puncture, CUN square, bridge edges, articulation gate, and edge-decomposition ledger are separate coordinates of the carrier.

Now use componentwise transport and replay. The rows H and T preserve those coordinates as finite BHist data, while Q and N record provenance and local naming for the same route.

Finally read the TasteGate row. Because R is displayed before a downstream carrier can be admitted, any later Lean realization must keep nontriviality and field faithfulness visible. Thus no consumer can erase the puncture, collapse the support graph, or bypass the stated gate requirements. □

Theorem 2508.4 (Punctured cube skeleton philosophy route)**.** *Every accepted* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier is the finite support-graph consumer of Definition 9462.1293: the rows $W, A, C, B, G, E, H, T, Q, R, N$ record the displayed punctured WNA cube, CUN square, bridge edges, articulation gate, decomposition ledger, transport, replay, provenance, TasteGate requirements, and local name. Its concrete sibling route is through the codon source-window packet of chapter 2234, the DNA window boundary tile of chapter 3393, and the codon-window spectral inverse packet of chapter 2161; those chapters supply finite window, boundary-tile, and spectral-inverse rows, not biochemical mechanism claims.*

Proof. The philosophy theorem supplies the displayed support formula. The carrier rows W, A, C, B, G, E are exactly the finite graph-support data used to read that formula as a punctured cube skeleton.

The sibling chapters named in the statement provide only finite BEDC routes: codon windows, DNA boundary tiles, and codon-window spectral inverse rows. Reading them alongside H, T, Q, R, N records a dependency route for the support graph, not an empirical biochemical assertion.

Thus the packet leaves kernel-only isolation by naming its concrete finite consumers while retaining the same boundary as Definition 3463.3. □

Falsifiable prediction: A packet that omits the ACA puncture row, the CUN square row, or one of the bridge-edge rows cannot be accepted as a $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ carrier.

Independence witness: codon-window-six-bridge, dna-window-six-dynamics-bridge, codon-window-spectral-inverse: none is bijective to $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ because this carrier binds the punctured WNA cube, CUN square, bridge edges, articulation gate, edge decomposition, TasteGate requirement row, transport, replay, provenance, and local naming as one finite support graph packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$.

Constructive story: A $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ packet is generated from WNA cube vertices, the deleted ACA puncture row, CUN square rows, bridge-edge rows, an articulation-gate support graph, an edge-decomposition ledger, hsame transports, Cont replay, Pkg provenance, a TasteGate requirement row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3463.1, Definition 3463.2, and Definition 3463.3 bind the finite punctured cube skeleton support graph and its TasteGate requirements.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biochemical mechanism, evolutionary pathway selection, host graph quotients, or completed TasteGate instances.

Upgrade path: Obligation closure requires checked carrier admission, boundary nonescape, transport stability, replay, provenance, local naming, and explicit Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.PuncturedCubeSkeletonUp.

Chapter 2509

A Constructive Closed Metric Ball Certificate for $\mathsf{ClosedMetricBall}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ClosedMetricBall}^{\uparrow}$ packet records the center-radius containment object of a constructive metric space as finite BEDC NameCert data. It sits after the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the separated comparison boundary of chapter 1411, the complete-metric limit surface of chapter 128, and the regular-Cauchy equality route of chapter 1412. The packet names non-strict metric containment for one displayed center and radius; it does not import host subsets, open-cover compactness, quotient point equality, or ambient completeness.

Definition 2509.1 (Closed metric ball carrier)**.** *A* $\mathsf{ClosedMetricBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, X, c, r, \rho, D, R, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the distance-read interface.*
2. *X is the inspected point row.*
3. *c is the displayed center row.*
4. *r is the* $\mathsf{Real}^{\uparrow}$ *or rational radius row.*
5. *ρ is the nonnegative-radius ledger, read before any containment classifier is consumed.*
6. *D is the metric-distance witness row for the finite comparison $d(c, X)$.*
7. *R is the ordered-real comparison row recording $d(c, X) \leq r$ through the* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *compatibility row, used only to transport zero-distance agreement through the same metric source.*
9. *Q is the optional* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *handoff row for consumers that arrive from completed or regular-Cauchy names.*
10. *H stores componentwise* hsame *transport for $M, X, c, r, \rho, D, R, S, Q$ and the structural rows.*
11. *C stores displayed* Cont *replay from the metric source and radius ledger to the distance witness, ordered comparison, separated route, and completion-facing handoff.*

12. *P is the* Pkg *provenance over* $\mathsf{ClosedMetricBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$ *row.*

The carrier compares accepted containment packets only through $M, X, c, r, \rho, D, R, S, Q, H, C, P, N$. It has no coordinate for arbitrary subsets, topology closure, selected limits, open-cover compactness, quotient point equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedMetricBallUp.ClosedMetricBallTasteGate single carrier alignment`

Theorem 2509.2 (Closed metric ball NameCert obligations)**.** *For an accepted* $\mathsf{ClosedMetricBall}^{\uparrow}$ *carrier $B = (M, X, c, r, \rho, D, R, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, center and point exposure, radius nonnegativity, distance-witness visibility, ordered-real containment, separated-metric compatibility, completion-facing handoff when Q is present, componentwise transport, continuation replay, provenance, and local naming. Every containment read therefore follows the displayed route*

$$M, X, c, r, \rho \longmapsto D \longmapsto R \longmapsto S, Q \longmapsto H, C, P, N$$

and no host subset or compactness principle supplies an extra membership coordinate.

Proof. Project the carrier rows. The metric source, inspected point, center, radius, nonnegative-radius ledger, distance witness, ordered-real comparison, separated route, completion-facing row, transport, replay, provenance, and local name are the only public coordinates.

The containment classifier reads the distance witness D only after M, X, c, r, ρ have been displayed. The row R then records the non-strict comparison through the $\mathsf{Real}^{\uparrow}$ seal rather than through an ambient subset predicate.

Separated and completed consumers are downstream. The row S can transport only zero-distance agreement already visible in the same metric source, and the row Q can hand a completed or regular-Cauchy read to the same comparison route only after the closed-ball rows are present.

Finally H, C, P, N preserve, replay, source, and name the route. Since the carrier contains no host subset, topological closure, open-cover, selected-limit, quotient, choice, trichotomy, or ambient-completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 2509.3 (Closed metric ball containment handoff)**.** *Every separated, bounded, complete, or compact metric consumer of an accepted* $\mathsf{ClosedMetricBall}^{\uparrow}$ *packet must read the finite containment route $M, X, c, r, \rho, D, R, S, Q, H, C, P, N$ before it may use the point as lying in the closed ball. The exported handoff is only the center-radius comparison $d(c, X) \leq r$ with its displayed transport, replay, provenance, and local name; it does not provide a compactness theorem, a host-level closed subset, quotient equality of points, a selected completion limit, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the containment request. The rows M, X, c, r, ρ fix the metric source, point, center, radius, and nonnegative-radius ledger before a distance comparison can be consumed.

The distance and ordered-real rows are next. The row D supplies the displayed metric-distance read, and R supplies the non-strict $\mathsf{Real}^{\uparrow}$ comparison. This is the only route by which the packet exposes membership.

Downstream metric consumers then read S or Q only as compatibility rows. They preserve the same center-radius comparison for separated or completed contexts, but they cannot replace it by topology closure, host set equality, compactness, or selected limits.

The structural rows H, C, P, N carry the route forward without adding mathematical content. Therefore every exported handoff remains the finite closed-ball containment certificate described above. □

Falsifiable prediction: A downstream metric consumer that omits the metric source row, the center-radius ledger, or the ordered-real containment row cannot read a $\mathsf{ClosedMetricBall}^{\uparrow}$ packet as a closed ball.

Independence witness: metric, real, separatedmetric, completemetric, cauchysequenceequivalence, metricclosedball: $\mathsf{ClosedMetricBall}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, inspected point, center, radius, nonnegative-radius ledger, distance witness, ordered-real containment, separated compatibility, completion-facing handoff, transport, replay, provenance, and local naming as one center-radius containment packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedMetricBall}^{\uparrow}$ packet is generated from a MetricUp source row, an inspected point row, a center row, a RealUp or rational radius row, a nonnegative-radius ledger, a metric-distance witness, an ordered-real containment comparison, separated-metric compatibility, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3464.1, Definition 3464.2, and Definition 3464.3 seed the finite center-radius containment packet for constructive metric spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, topology closure, open-cover compactness, quotient point equality, selected completion limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, radius nonnegativity, distance-witness visibility, ordered-real containment, separated compatibility, completion-facing handoff, transport and replay stability, provenance non-escape, and local naming.

Chapter 2510

A Concrete Naming Certificate for $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen, the $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ packet records finite-prefix dependence of Baire and choice-sequence functionals as NameCert data. It sits beside the Baire-space surface of chapter 334, the choice-sequence rows of chapter 1482, the spread-law route of chapter 1532, and the finite stream-window surface of chapter 126. The packet also routes rational and real readbacks through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, keeping the continuity claim as a finite-prefix certificate rather than as a host functional, countable-choice selector, quotient stream equality, or ambient real-completeness theorem.

Definition 2510.1 (Brouwer continuity principle carrier)**.** *A* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, Q, S, W, R, E, M, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *source row, exposing only finite inspected prefixes of the input sequence;*
2. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row, recording the displayed choice-sequence presentation used by the functional read;*
3. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row, naming the spread law that admits the finite prefixes consumed by the packet;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, storing the requested prefix length, inspected stream cells, and pullback route to the source rows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations made from the accepted finite prefix;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same finite-window and rational-readback data;*
7. *M is the finite modulus row, stating which prefix length is sufficient for the displayed output read;*
8. *H records componentwise* hsame *transports for B, Q, S, W, R, E, M and the local naming row;*
9. *C records the displayed* Cont *replay by which consumers read $B, Q, S, W, R, E, M, H, P, N$ in that order;*
10. *P is the* Pkg *provenance over* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through the displayed finite prefix, modulus, readback, seal, transport, continuation, package, and local NameCert rows.

Theorem 2510.2 (Brouwer continuity principle NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$ *surface for Definition 3465.1 consists exactly of carrier admission for $K = (B, Q, S, W, R, E, M, H, C, P, N)$, Baire-source exposure through B, choice-sequence exposure through Q, spread-law admission through S, finite stream-window inspection through W, rational readback through R, real-seal routing through E, finite-modulus display through M, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N. Every accepted continuity read is classified only by these displayed rows.*

Lean4 ✓ `BEDC.Derived.BrouwerContinuityPrincipleUp.BrouwerContinuityPrincipleTasteGate single carrier alignment`

Proof. Project the carrier tuple K. The Baire source, choice-sequence presentation, spread law, finite stream window, rational readback, real seal, finite modulus, transports, continuations, package row, and local name are all stored coordinates.

The source and admissibility obligations project B, Q, S, W. The output-facing obligations project R, E, M, so the real read is reached only after the finite prefix and rational readback have been displayed.

The structural obligations project H, C, P, N. Since no coordinate names a host functional, global continuity theorem, countable-choice selector, quotient stream equality, selected real representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the local certificate exports only the finite-prefix route. □

Theorem 2510.3 (Brouwer continuity principle finite-prefix modulus)**.** *For every accepted* $\mathsf{BrouwerContinuityPrinci}$ *carrier, each output read from a Baire or choice-sequence input factors through the finite route*

$$B, Q, S \longmapsto W \longmapsto M \longmapsto R \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Thus the consumer receives a displayed prefix length and the corresponding inspected $\mathsf{StreamName}^{\uparrow}$ *window before it reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The route exports no host functional continuity theorem, no choice-selected modulus, no quotient stream equality, no arbitrary bar induction principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Read B, Q, S first. These rows display the Baire-space source, the choice-sequence presentation, and the spread-law admission surface on which the functional read is allowed to depend.

Next read the finite-window and modulus rows W, M. They name the inspected prefix and the prefix length sufficient for the displayed output read, so the rational and real observations cannot be consumed before the finite prefix is exposed.

Finally route the output through R, E, H, C, P, N. Componentwise transport and continuation replay preserve the same finite-prefix packet for downstream consumers, while provenance records the sibling Baire-space, choice-sequence, spread-law, stream-name, regular-rational, and real rows. Since no refused object is a carrier coordinate, the output read remains a finite-prefix NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ packet is generated from $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite-modulus, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{BrouwerContinuityPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 3465.1, Definition 3465.2, and Definition 3465.3 seed the finite-prefix continuity route over the displayed Baire-space, choice-sequence, spread-law, stream-name, regular-rational, real-seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{BrouwerContinuityPrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, host functional continuity theorem, choice-selected modulus, quotient stream equality, arbitrary bar induction principle, selected real representative, ambient Real$^{\uparrow}$ completeness theorem, or bridge-checked standard model.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same BaireSpace$^{\uparrow}$, ChoiceSequence$^{\uparrow}$, SpreadLaw$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 2511

A Concrete Naming Certificate for $\mathsf{CauchyHadamard}^{\uparrow}$

Following Cauchy and Hadamard's power-series radius formula, the $\mathsf{CauchyHadamard}^{\uparrow}$ packet records the constructive radius route for a real power series as finite BEDC certificate data. It sits over the coefficient windows of chapter 1583, the radius seal of chapter 66, the regular root-bound route of chapter 1481, and the sibling ratio-test comparison route of chapter 1461. The chapter packages the traditional formula as a NameCert route through displayed coefficient windows, dyadic tolerance rows, and regular-Cauchy readbacks rather than through a host limsup, quotient stream, choice-selected radius, or ambient completeness theorem.

Definition 2511.1 (Cauchy-Hadamard carrier)**.** *A* $\mathsf{CauchyHadamard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, R, G, D, H, C, P, N)$$

with the following visible rows:

1. *A is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *source row, exposing only the displayed coefficient family and its finite inspection windows;*
2. *W is the coefficient-window ledger, recording the inspected unary indices, absolute-coefficient reads, and root-bound requests used by the packet;*
3. *R is the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound route, read from the same coefficient windows and tolerance requests;*
4. *G is the* $\mathsf{ConvRad}^{\uparrow}$ *seal row reached from the root-bound route and the power-series source;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic slack and comparison budgets used by the root-bound and radius rows;*
6. *H records componentwise* hsame *transports for the source row, coefficient windows, root-bound route, radius seal, tolerance ledger, and local naming row;*
7. *C records the displayed* Cont *replay by which consumers read A, W, R, G, D, H, P, N in that order;*
8. *P is the* Pkg *provenance over* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and their recorded transport, continuation, package, and local NameCert provenance.

Lean4 ✓ `BEDC.Derived.CauchyHadamardUp.CauchyHadamardTasteGate single carrier alignment`

Theorem 2511.2 (Cauchy-Hadamard NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$ *surface for Definition 3466.1 consists exactly of carrier admission for* $K = (A, W, R, G, D, H, C, P, N)$*, coefficient-window exposure through* A, W*, root-bound exposure through* R*, radius-seal exposure through* G*, dyadic tolerance exactness through* D*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*. Every accepted radius read is classified only by the displayed power-series row, finite coefficient windows, root-bound route, tolerance ledger, radius seal, and structural rows.*

Proof. Project the carrier tuple K. The source row, coefficient-window ledger, root-bound route, radius seal, dyadic tolerance ledger, transport rows, continuation routes, package row, and local name are all stored coordinates.

Carrier admission and coefficient exposure are therefore finite projections from A and W. The root-bound and radius obligations project R and G, with D recording the tolerance comparisons used by both rows.

The structural obligations project H, C, P, N. Since no coordinate names a host limsup, quotient stream, choice-selected radius, or ambient completeness theorem, the local certificate exports only the displayed Cauchy-Hadamard route. □

Theorem 2511.3 (Cauchy-Hadamard root-bound handoff)**.** *For every accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *carrier, each root-bound request consumed by the radius row factors through the finite route*

$$A \longmapsto W \longmapsto D \longmapsto R \longmapsto G \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The handoff uses the $\mathsf{RealPowerSeries}^{\uparrow}$ *coefficient source, the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{ConvRad}^{\uparrow}$ *seal. The* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *sibling is available only as a comparison provenance row in* P *and does not replace the displayed root-bound route.*

Proof. Read A, W, D, R, G from the carrier. A radius request first opens the power-series coefficient source and the finite coefficient-window ledger, so the root-bound row sees only displayed coefficient data.

Use the dyadic tolerance ledger D to align the slack and comparison budget consumed by R. The radius seal G is then read after the root-bound route, not from a host supremum or selected limsup witness.

The continuation and provenance rows preserve the same order for downstream consumers. The ratio-test sibling remains a recorded dependency anchor in P, while the accepted handoff itself is the root-bound route through $A, W, D, R, G, H, C, P, N$. □

Theorem 2511.4 (Cauchy-Hadamard root-bound NameCert route)**.** *Any accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *radius read whose* $\mathsf{ConvRad}^{\uparrow}$ *row consumes a Cauchy-Hadamard exactness witness factors through the displayed* $\mathsf{RealPowerSeries}^{\uparrow}$ *coefficient source, finite coefficient windows,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{ConvRad}^{\uparrow}$ *seal. The exactness row is a local witness inside the displayed packet, not a host limsup, quotient stream, selected radius representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier $K = (A, W, R, G, D, H, C, P, N)$ of Definition 3466.1. The coefficient source is A, the finite windows are W, the root-bound route is R, the tolerance ledger is D, and the radius seal is G.

Apply the root-bound handoff theorem. It orders every radius request through A, W, D, R, G before the structural rows H, C, P, N expose the local certificate to a consumer.

The Cauchy-Hadamard exactness witness is therefore read as part of the local $\mathsf{ConvRad}^{\uparrow}$ seal row. Since no carrier coordinate stores a host limsup, quotient coefficient stream, selected radius, or ambient completeness theorem, the NameCert route remains the displayed finite root-bound packet. □

Theorem 2511.5 (Cauchy-Hadamard radius nonescape)**.** *Every* $\mathsf{ConvRad}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RealPowerSeries}^{\uparrow}$ *source row, finite coefficient windows,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* BHist *histories,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *row. It exports no host limsup, real supremum row, quotient coefficient stream, countable-choice selector, selected radius representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the radius seal row G. By the carrier definition, G is reached only after the source row A, coefficient windows W, tolerance ledger D, and root-bound route R have been displayed.

Apply the NameCert obligations. They identify the structural support of the read as H, C, P, N and keep componentwise transport and continuation replay attached to the same finite packet.

No refused object has a row in the carrier. Therefore a downstream $\mathsf{ConvRad}^{\uparrow}$ consumer cannot escape the displayed Cauchy-Hadamard route by invoking a host limsup, real supremum, quotient stream, choice selector, selected radius, or ambient real-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyHadamard}^{\uparrow}$ packet is generated from a $\mathsf{RealPowerSeries}^{\uparrow}$ coefficient source row, finite coefficient windows, a $\mathsf{RegularCauchyRootTest}^{\uparrow}$ root-bound route, a $\mathsf{ConvRad}^{\uparrow}$ seal row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3466.1, Definition 3466.2, Definition 3466.3, and Definition 3466.5 bind the constructive power-series radius route over $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, host limsup, real supremum row, quotient coefficient stream, countable-choice selector, selected radius representative, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, or bridge-checked standard model.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 2512

A Concrete Naming Certificate for $\mathsf{DyadicSubsequence}^{\uparrow}$

Following Bishop and Bridges, constructive subsequence extraction is read as a finite choice of inspected windows and monotone index evidence before any compactness conclusion is invoked. The $\mathsf{DyadicSubsequence}^{\uparrow}$ packet records that L10 selection surface over the bounded real sequence rows of chapter 3451, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the compactness-facing route of chapter 1254. It packages dyadic tolerance windows, monotone unary reindexing, and Real-seal readback as displayed NameCert data rather than as an ambient subsequence selector or a completed compactness theorem.

Definition 2512.1 (Dyadic subsequence carrier)**.** *A* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, D, I, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the bounded real sequence source row, exposing only the displayed sequence name, bound certificate, and finite request interface.*
2. *W is the finite window ledger, recording the sequence windows inspected before each subsequence output is admitted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, storing dyadic comparison budgets for window selection, bound reuse, and later readback.*
4. *I is the monotone unary reindexing row, recording a finite increasing index map from output positions to inspected source positions.*
5. *O is the* $\mathsf{StreamName}^{\uparrow}$ *output-window row, exposing the selected subsequence only by finite windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from O, W, D, I.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each accepted readback request, exposed only after the source, window, tolerance, index, stream, and regular-readback rows are displayed.*
8. *H records componentwise* hsame *transport for the bounded sequence source, windows, dyadic ledger, monotone index row, stream windows, regular readback, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a subsequence or compactness-facing request through* $A, W, D, I, O, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{RealBoundedSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, D, I, O, R, E, H, C, P, N$. It has no coordinate for a global choice function, quotient stream, selected limit, ambient compactness theorem, or host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2512.2 (Dyadic subsequence classifier)**.** *Two accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carriers*

$$S = (A, W, D, I, O, R, E, H, C, P, N) \quad \text{and} \quad S' = (A', W', D', I', O', R', E', H', C', P', N')$$

are classified as the same local subsequence certificate exactly when the displayed bounded-source rows agree, the finite window ledgers present the same inspected source windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows agree on every recorded comparison budget, the monotone unary reindexing rows choose the same finite output-to-source positions, the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback rows agree after the recorded* hsame *transports, and the* Cont, Pkg, *and* NameCert *rows replay and name the same packet.*

Theorem 2512.3 (Dyadic subsequence NameCert obligations)**.** *For every accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carrier $S = (A, W, D, I, O, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded sequence source, finite windows, dyadic tolerance rows, monotone index row, finite stream output, regular rational readback, terminal Real seals, componentwise transport, continuation replay, package provenance, and local naming. Every compactness-facing read of the packet is classified only by these displayed rows and exports no global selector, quotient stream, selected limit, ambient compactness theorem, or host Real-completeness coordinate.*

Proof. Carrier admission is projection from Definition 3467.1. The tuple displays the bounded source, inspected windows, dyadic budgets, monotone reindexing row, output stream windows, regular readback, Real seals, and structural rows.

The classifier of Definition 3467.2 fixes equality of accepted packets by those same rows. Thus a NameCert consumer cannot compare subsequence data by an off-packet selector or by an ambient limit object.

The structural obligations project H, C, P, N. They transport, replay, source, and name only the displayed packet, so the excluded global choice, quotient-stream, selected-limit, compactness, and Real-completeness coordinates remain absent. □

Lean4 ✓ `BEDC.Derived.DyadicSubsequenceUp.TasteGate.DyadicSubsequenceTasteGate single carrier alignment`

Theorem 2512.4 (Dyadic subsequence window extraction)**.** *Every finite output window exported by an accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *packet factors through the route*

$$A \longmapsto W \longmapsto D \longmapsto I \longmapsto O \longmapsto R \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route first opens a bounded real sequence source, then records the inspected source windows, dyadic tolerances, and monotone unary reindexing before exposing the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback rows. No output window may be read directly from an ambient subsequence choice function or from a Bolzano-Weierstrass compactness conclusion.*

Proof. Start from an accepted carrier $S = (A, W, D, I, O, R, E, H, C, P, N)$. A finite output-window request is replayed through C, so it must expose the bounded source A, the source windows W, the dyadic tolerance ledger D, and the monotone index row I before it reads the output stream row O.

Regular rational readback and Real sealing occur after those rows. The packet reaches R and E only through the displayed stream window and tolerance route, with H preserving componentwise transport.

Finally P and N source and name the same finite route. Since no carrier coordinate stores an ambient selector or compactness theorem, every exported output window remains a finite window-extraction certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSubsequence}^{\uparrow}$ packet is generated from a bounded real sequence source, finite source windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, monotone unary reindexing, finite $\mathsf{StreamName}^{\uparrow}$ output windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3467.1, Definition 3467.2, Definition 3467.3, and Definition 3467.4 seed the finite L10 subsequence-selection surface between bounded real sequence windows and compactness-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean implementation, Bolzano-Weierstrass compactness, selected limits, quotient streams, countable choice, host Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance exactness, monotone index-window extraction, StreamName output, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicSubsequenceUp.

Chapter 2513

A Concrete Naming Certificate for $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ packet records when a finite tail shift preserves the named regular Cauchy object as displayed BEDC data. It sits over the finite schedule rows of chapter 126, the regular rational classifier of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-shift packet of chapter 1578, and the terminal real-name seal of chapter 38. The packet is a NameCert surface for tail-shift equivalence of regular Cauchy names, not a quotient of streams, not selected host equality, and not an ambient completeness theorem.

Definition 2513.1 (Regular Cauchy shift equivalence carrier)**.** *A* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, k, S, R, D, H, C, P, N)$$

with the following visible rows:

1. *Q is the source regular Cauchy name row, presented through the same regular rational readback used by the original* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier.*
2. *k is the finite tail-shift index, displayed as a unary prefix length and carrying no unbounded selector.*
3. *S is the shifted* $\mathsf{StreamName}^{\uparrow}$ *row whose window request at n reads the source name at the displayed index $n + k$.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row for the shifted readback, attached to Q through the displayed schedule S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic slack preserved by the shifted readback.*
6. *H records componentwise* hsame *transport for the source name, shift index, shifted stream row, classifier row, dyadic ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an equivalence request through k, S, R, D, and the source row Q, and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected representatives, host sequence equality, countable-choice tail selectors, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyShiftEquivalenceUp.RegularCauchyShiftEquivalenceTasteGate`
`single carrier alignment`

Theorem 2513.2 (Regular Cauchy shift equivalence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier* $E = (Q, k, S, R, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source regular Cauchy name, finite shift index, shifted* $\mathsf{StreamName}^{\uparrow}$ *row, shifted* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, transports, continuation, provenance, and local naming row;*
2. *source-name retention, requiring the shifted classifier to read the same source regular Cauchy name Q after the finite index shift;*
3. *shifted-window exposure, requiring every request to pass through the displayed shift index k before it can inspect S;*
4. *classifier stability, requiring the shifted* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to remain attached to the displayed windows in S and the tolerance ledger D;*
5. *real-seal compatibility, requiring every* $\mathsf{Real}^{\uparrow}$*-facing read to pass through Q, k, S, R, D before any seal-facing consumer can inspect the structural rows;*
6. *ledger refusal, excluding quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3468.1. The displayed coordinates are exactly the source regular Cauchy name, finite shift index, shifted stream row, shifted regular-rational classifier, dyadic tolerance ledger, and the structural rows H, C, P, N.

Replay the continuation row C. An equivalence request first consumes the finite shift index k, then reads the shifted schedule S and the shifted classifier row R attached to the source row Q.

The dyadic ledger D is read on the same route. It supplies the tolerance data needed before a $\mathsf{Real}^{\uparrow}$-facing consumer can inspect the shifted readback as a real-name seal.

The remaining rows only transport, replay, package, and locally name that finite route. Since no coordinate stores quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2513.3 (Regular Cauchy shift equivalence tail stability)**.** *Let $E = (Q, k, S, R, D, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier. Every tail-equivalence consumer and every* $\mathsf{Real}^{\uparrow}$*-facing consumer factors through the finite route*

$$Q \longmapsto k \longmapsto S \longmapsto R \longmapsto D \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Thus the shifted regular Cauchy name remains stable under the displayed finite tail shift before quotient streams, host sequence equality, selected representatives, countable-choice tail selectors, or ambient completion data are available.

Proof. Begin with a consumer request for the shifted regular Cauchy name. Carrier admission supplies only the source row Q, the finite shift index k, the shifted schedule S, the classifier row R, the dyadic ledger D, and the structural rows.

The shifted schedule is read after k. Hence every displayed window of the shifted name is still a window of the source name at the translated finite index, and the classifier row R remains attached to that same readback.

The tolerance ledger D travels with the shifted read. Transport by H and replay by C preserve the route through Q, k, S, R, D, while P and N only record provenance and local naming for that route.

No refused object has a carrier coordinate. A tail-equivalence or $\mathsf{Real}^{\uparrow}$-facing consumer therefore receives the displayed finite shift route and no quotient stream, host sequence equality row, selected representative, choice selector, or ambient completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ packet is generated from a source regular Cauchy name row, finite tail-shift index, shifted StreamName row, RegSeqRat classifier row, DyadicRatCore tolerance ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3468.1, Definition 3468.2, and Definition 3468.3 seed the finite tail-shift equivalence route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, ambient Real completeness, Lean verification, or a checked standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shifted-window exposure, classifier stability, dyadic tolerance preservation, real-seal compatibility, and ledger refusal for BEDC.Derived.RegularCauchyShiftEquivalenceUp.

Chapter 2514

A Concrete Naming Certificate for $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ packet records the constructive diagonal extraction of a regular Cauchy subsequence as finite BEDC NameCert data. It sits beside the stream-window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the subsequence and interleaving routes of chapter 1854 and chapter 3405. The packet names a diagonal finite-window selector for one displayed family of tails; it is not host subsequence equality, a chosen limit, quotient equality of real names, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2514.1 (Diagonal Cauchy subsequence carrier)**.** *A* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, T, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *schedules whose rows display the source sequence windows and the tail windows available to the diagonal construction;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger attached to the same source and tail schedules;*
3. *T is the Cauchy subsequence extraction row, recording the finite increasing-index and tail-request data consumed from the* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *and* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *sibling routes;*
4. *Q is the diagonal selector row, pairing each requested precision window with the subsequence tail whose regular rational readback is admitted for that window;*
5. *H records componentwise* hsame *transport for schedules, regular readbacks, extraction rows, diagonal selector rows, replay, provenance, and local naming;*
6. *C records displayed* Cont *replay from a diagonal request through S, R, T, Q, H before any* $\mathsf{Real}^{\uparrow}$ *seal is readable;*
7. *P is the* Pkg *provenance over* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySubsequenceExtra}$ $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$ *row.*

The carrier exports a finite diagonal extraction surface. It has no coordinate for host subsequence equality, selected limits, quotient equality of real names, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiagonalCauchySubsequenceUp.DiagonalCauchySubsequenceTasteGate single carrier alignment`

Theorem 2514.2 (Diagonal Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier* $D = (S, R, T, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$ *surface consists exactly of schedule admission for the* $\mathsf{StreamName}^{\uparrow}$ *rows in* S*; regular rational readback admission through* R*; subsequence-extraction admission through* T*; diagonal selector exactness through* Q*; preservation under the componentwise* hsame *rows of* H*; displayed* Cont *replay through* C*; provenance through* P*; and local naming through* N*. These obligations keep the diagonal selector inside the finite schedule, readback, extraction, transport, replay, provenance, and naming rows.*

Proof. Project the carrier. The schedules, regular readbacks, extraction row, selector row, transports, replay, provenance, and local name are the only visible coordinates of D.

Read the analytic rows in order. The $\mathsf{StreamName}^{\uparrow}$ schedules in S display the finite source and tail windows, R supplies the regular rational readbacks for those windows, and T records the subsequence extraction data supplied by the sibling subsequence and interleaving criteria.

The diagonal row Q is downstream of those reads. It may choose among displayed windows already admitted by S, R, T, but it has no socket for a host subsequence, a selected real limit, quotient equality, trichotomy, or ambient completeness.

Finally use H, C, P, N. Transport and replay preserve the same finite rows, and the package and local certificate name that route without adding any external diagonal principle. □

Theorem 2514.3 (Diagonal Cauchy subsequence regularity)**.** *For an accepted* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier* $D = (S, R, T, Q, H, C, P, N)$*, every diagonal read exported to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer preserves the regular Cauchy windows displayed by* S *and* R*. The consumer route factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchySubsequenceExtraction}^{\uparrow},\ \mathsf{CauchyInterleavingCriterion}^{\uparrow},\ Q, H, C, P, N.$$

Consequently the packet cannot expose host subsequence equality, selected limits, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted diagonal read. The route must first inspect the finite schedules S and the attached regular rational readbacks R, so the read is already tied to displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows.

Next inspect the subsequence and interleaving row T. It records only finite increasing-index and tail-window evidence from the sibling Cauchy routes, and the diagonal selector Q can select only from those displayed rows.

The structural rows preserve the same regular windows. Componentwise hsame transport and displayed Cont replay carry S, R, T, Q together, while P, N source and name the accepted packet. Since no excluded object is a coordinate of the carrier, no consumer can obtain host subsequence equality, a selected limit, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the diagonal packet. □

Falsifiable prediction: Every accepted $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ read factors through displayed StreamName schedules, RegSeqRat readbacks, a Cauchy subsequence extraction row, a diagonal selector row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any RealUp seal is consumed.

Independence witness: streamname, regseqrat, real, cauchysubsequencecriterion, cauchy-interleaving-criterion: $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source schedules, regular readbacks, extraction data, a diagonal selector, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readbacks, a Cauchy subsequence extraction row, a diagonal selector row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3469.1, Definition 3469.2, and Definition 3469.3 seed the finite diagonal subsequence route through stream schedules, regular readbacks, and RealUp seal boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsequence equality, selected limits, quotient equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal selector exactness, regular-window preservation, non-escape, and bridge-facing consumer rows for BEDC.Derived.DiagonalCauchySubsequenceUp.

Chapter 2515

A Concrete Naming Certificate for $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$

The $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ packet realizes the trigger-support and coverage-rank rows of subsection 9462.28.21 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the boundary-tile classifier of chapter 3393, the punctured support skeleton of chapter 3463, and the inverse spectral route of chapter 2161. The packet records coverage witnesses for boundary-trigger rows; it is not a biological code table, a stochastic evolutionary model, a quotient codon space, or a hidden global semantics.

Definition 2515.1 (Six boundary tile coverage carrier)**.** *A* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, T, G, R, A, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the six bit coordinates on which trigger support is read;*
2. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers;*
3. *G is the* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *support graph row listing the puncture, bridge, CUN-square, and articulation rows used by coverage reads;*
4. *R is the coverage-rank row, recording which displayed boundary triggers cover which support faces and which rows remain uncovered;*
5. *A is the* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *inverse-spectral handoff row, used only after W, T, G, R are displayed;*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, support graph rows, coverage-rank rows, inverse-spectral handoff, replay, provenance, and local naming;*
7. *C records displayed* Cont *replay from a trigger-coverage request through W, T, G, R, A, H;*
8. *P is the* Pkg *provenance over* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$, $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$, *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$ *row.*

The carrier accepts coverage only when every trigger read is exhausted by $W, T, G, R, A, H, C, P, N$. It has no coordinate for a hidden biological table, quotient codon equality, selected stochastic dynamics, or global semantic equivalence.

Theorem 2515.2 (Six boundary tile coverage NameCert obligations)**.** *For every accepted* SixBoundaryTileCoverage *carrier $K = (W, T, G, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$ *surface consists exactly of six-bit source-window admission through W, boundary-tile admission through T, punctured-skeleton support admission through G, coverage-rank exactness through R, inverse-spectral handoff through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.SixBoundaryTileCoverageUp.SixBoundaryTileCoverageTasteGate single carrier alignment`

Proof. Project the accepted carrier. The rows $W, T, G, R, A, H, C, P, N$ list all source-window, boundary-tile, support-graph, coverage-rank, inverse-spectral, transport, replay, provenance, and naming data available to the packet.

The trigger-support surface is finite. A coverage request first reads the six-coordinate source window W, then the boundary tile row T, and then the punctured support row G before the rank ledger R can be inspected.

The inverse-spectral handoff A is downstream of that support route. It may compare coverage rows with spectral inverse data, but it cannot replace them by a hidden code table or by stochastic semantics.

Finally H, C, P, N preserve and name the same finite route. Since every accepted coverage read is exhausted by those rows, no quotient codon space, selected dynamics, or global biological semantics is exported. □

Theorem 2515.3 (Six boundary tile trigger nonescape)**.** *Every accepted trigger-coverage read of a* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *packet factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{PuncturedCubeSkeleton}^{\uparrow},\ R,\ \mathsf{CodonWindowSpectralInverse}^{\uparrow},\ H$$

In particular, a trigger coverage claim cannot escape the displayed six-coordinate source, boundary-tile classifier, punctured support graph, coverage-rank row, inverse-spectral handoff, componentwise transport, replay, provenance, and local naming rows.

Proof. Start with the source rows. The six-bit window W exposes the coordinates on which a trigger is read, while T restricts the trigger to the displayed boundary-tile classifier.

Next read the support and rank rows. The punctured skeleton G supplies the support faces, and R records exactly which displayed triggers cover which faces. The spectral inverse row A is only a handoff from this route, not an alternate coverage source.

The structural rows close the route. Componentwise transport and replay preserve W, T, G, R, A together, while provenance and local naming source the accepted packet. Thus a consumer cannot obtain a hidden biological table, quotient codon equality, selected stochastic dynamics, or global semantic equivalence from a trigger-coverage read. □

Falsifiable prediction: A concrete trigger-coverage nonescape claim for $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ is that every accepted boundary-trigger read preserving $W, T, G, R, A, H, C, P, N$ remains inside the displayed six-coordinate source, boundary tile, punctured support graph, coverage-rank, and inverse-spectral route.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, punctured-cube-skeleton, codon-window-spectral-inverse: $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries trigger support, coverage rank, inverse-spectral handoff, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.SixBoundaryTileCoverageUp.TasteGate` must provide explicit Nontrivial $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ and FieldFaithful $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ instances preserving $W, T, G, R, A, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$.

Constructive story: A $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ packet is generated from codon six-bit windows, boundary-tile classifiers, punctured support skeleton rows, coverage-rank rows, inverse-spectral handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3470.1, Definition 3470.2, and Definition 3470.3 seed the finite trigger-coverage packet over six-bit windows, boundary tiles, support skeletons, and spectral inverse rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological table identity, stochastic evolutionary modeling, quotient codon equality, global semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-support exactness, coverage-rank nonescape, inverse-spectral handoff, Nontrivial SixBoundaryTileCoverageUp, and FieldFaithful SixBoundaryTileCoverageUp for BEDC.Derived.SixBoundaryTileCoverageUp.

Chapter 2516

A Concrete Naming Certificate for $\mathsf{HeineBorelFiniteNet}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of compact intervals by finite approximation data, the $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ packet records the interval-cover certificate obtained from a located closed interval, a displayed finite net, a positive Lebesgue-number ledger, and the endpoint real-name boundary. It consumes the finite-net interval surface of chapter 1808, the compact-cover radius route of chapter 2301, the compact uniform-modulus route of chapter 3444, the terminal real-name seal of chapter 38, and the dyadic tolerance rows of chapter 291. The packet is a finite-net interval-cover certificate; it is not a general open-cover compactness theorem, an unrestricted Heine-Borel principle, a selected subcover functional, or an ambient completeness assertion.

Definition 2516.1 (Heine-Borel finite-net carrier)**.** *A* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, F, L, C, R, D, E, T, Q, P, N)$$

with the following displayed rows:

1. *I is the located closed interval source row, exposing the lower endpoint, upper endpoint, and endpoint order data as* $\mathsf{Real}^{\uparrow}$ *histories.*
2. *F is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *finite-net row, listing the displayed mesh scale and finite interval net consumed by the cover request.*
3. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *positive-radius row attached to the displayed finite cover or pointwise radius family.*
4. *C is the* $\mathsf{HeineCantor}^{\uparrow}$ *compact-uniformity row that may be read only as a sibling route through the same finite-net and radius data.*
5. *R is the cover-read row, assigning each listed net point to its displayed cover witness and recording the finite neighborhood test used for nearby interval points.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording mesh width, endpoint containment, radius comparison, and finite cover slack checks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and cover-boundary reads.*
8. *T records componentwise* hsame *transport for the interval source, finite-net row, Lebesgue-number row, Heine-Cantor row, cover-read row, dyadic ledger, Real seal, replay row, provenance row, and local naming row.*

9. *Q records displayed* Cont *replay from an interval-cover request through I, F, L, R, D, E and, when needed by a uniform-continuity consumer, through the sibling row C.*

10. *P is the* Pkg *provenance over* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, C, R, D, E, T, Q, P, N$. It has no coordinate for arbitrary open covers, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineBorelFiniteNetUp.HeineBorelFiniteNetTasteGate single carrier alignment`

Theorem 2516.2 (Heine-Borel finite-net NameCert obligations)**.** *For every accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the located interval source, finite-net row, Lebesgue-number row, Heine-Cantor sibling row, cover-read row, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*

2. *interval finite-net exposure, requiring the cover request to display I and F before any cover witness is read;*

3. *radius compatibility, requiring the positive-radius row L and dyadic ledger D to control mesh width, endpoint containment, cover slack, and finite neighborhood tests;*

4. *sibling uniformity compatibility, requiring the* $\mathsf{HeineCantor}^{\uparrow}$ *row C to be read only as a sibling route over the same finite-net and radius rows;*

5. *non-escape, requiring T, Q, P, N to transport, replay, source, and name only the displayed finite packet and to export no arbitrary open-cover compactness, unrestricted Heine-Borel principle, selected subcover, countable-choice row, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is projection from Definition 3471.1. The tuple displays exactly the interval source, finite net, positive-radius row, sibling uniformity route, cover reads, dyadic tolerance ledger, Real seal, and structural rows.

The interval-cover computation first reads I and F. Thus every point-facing cover read is tied to the displayed located interval and to a listed finite mesh row before any neighborhood witness can be inspected.

The radius and cover checks are finite. The row L supplies the positive radius ledger, while D records the mesh-width and containment comparisons that let R transfer a cover witness from a listed net point to the interval point under inspection.

The sibling uniformity row C is only a compatibility route. It can be consumed by a Heine-Cantor-style downstream reader, but it does not replace the finite-net cover row or introduce an open-cover compactness principle.

Finally, the refused objects have no carrier coordinate. Since T, Q, P, N transport, replay, source, and name only the displayed packet, no accepted $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ certificate exports unrestricted Heine-Borel compactness, a selected subcover, countable choice, host equality, or ambient Real completeness. □

Theorem 2516.3 (Heine-Borel finite-net interval-cover route)**.** *Every interval-cover read from an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{HeineCantor}^{\uparrow}$ *row is available as the sibling compact-uniformity route over the same finite-net and radius data, but the exported certificate remains the displayed finite-net interval-cover route.*

Proof. Project the accepted carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$. The interval source and finite-net row are I and F, the positive-radius row is L, the dyadic comparison ledger is D, and the terminal endpoint seal is E.

Replay through Q keeps that order. A cover request opens the finite net, reads the positive-radius ledger, checks the dyadic slack against the mesh and endpoint rows, and only then exposes the Real-facing endpoint seal.

The row C records the compatible Heine-Cantor route for consumers that also need compact uniformity. It remains tied to the same finite rows, so it cannot introduce a selected subcover, an unrestricted open-cover theorem, or an ambient completeness coordinate. □

Definition 2516.4 (Heine-Borel finite-net grid refinement)**.** *Let* $H = (I, F, L, C, R, D, E, T, Q, P, N)$ *be an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier and let a displayed dyadic submesh refine the finite net row* F*. The grid refinement is the packet obtained by transporting the positive-radius row* L *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* D *while keeping the located interval source* I*, cover-read row* R*, endpoint seal* E*, transport row* T*, replay row* Q*, provenance row* P*, and local name row* N *unchanged. The refinement preserves the same finite mesh route and does not add any selected subcover coordinate.*

Theorem 2516.5 (Heine-Borel finite-net grid refinement stability)**.** *Refining the finite net row* F *of an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet by a displayed dyadic submesh and transporting the positive-radius row* L *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger yields another accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet with the same* $\mathsf{Real}^{\uparrow}$ *endpoint seal and the same nonescape boundary. The refined packet still factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

and it exports no selected subcover, quotient interval equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3471.1. The finite net row F and the positive-radius row L are the only rows changed by the displayed dyadic submesh.

Replay Definition 3471.3. It forces the refined mesh to be read through the located interval source, the positive-radius ledger, the dyadic comparison row, and the Real seal in the same order as the unrefined packet.

Now use the grid-refinement packet itself. Because the refinement keeps I, R, E, T, Q, P, N unchanged and only retunes the mesh through L and D, the consumer still sees the same finite-net interval-cover route and no selected subcover or ambient completeness coordinate. □

Theorem 2516.6 (Heine-Borel finite-net obligation lattice)**.** *Any accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet must be read in the obligation order*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{HeineCantor}^{\uparrow}$ *row is only a sibling uniformity consumer over the same finite interval, positive-radius, dyadic, and Real-seal rows. The packet does not export general open-cover compactness, unrestricted Heine-Borel, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$. The finite interval source and net rows are I and F, the positive-radius row is L, the dyadic tolerance ledger is D, and the terminal Real-facing seal is E.

The cover-read row R can be consumed only after the interval and finite-net rows have been displayed, and the replay row Q keeps the Lebesgue-number row ahead of every dyadic slack check. Thus the finite-net obligation surface is ordered by interval net, positive radius, dyadic comparison, and Real seal.

The row C is read differently. It is available to a $\mathsf{HeineCantor}^{\uparrow}$ consumer that needs compact uniformity over the same finite rows, but it does not precede the finite-net route and does not replace F, L, D, E.

Finally the structural rows T, P, N transport, source, and name only the displayed packet. Since the refused objects have no coordinate in H, the obligation lattice cannot escape to unrestricted compactness, selected subcovers, quotient representatives, countable choice, host equality, or ambient completeness. □

Theorem 2516.7 (Heine-Borel finite-net scope witness)**.** *Every accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet reads only the displayed located-closed-interval finite-net row, the Lebesgue-number positive-radius row, the Heine-Cantor sibling row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. Thus the chapter scope is closed at the finite-net interval-cover certificate boundary and does not extend to general open-cover compactness, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition Definition 3471.1 lists exactly the rows $I, F, L, C, R, D, E, T, Q, P, N$ and states that the classifier compares accepted packets only through those rows.

Apply Definition 3471.2 to identify the local NameCert obligations, Definition 3471.3 to force every interval-cover read through the located interval, finite net, positive radius, dyadic ledger, and Real seal, and Definition 3471.6 to keep the Heine-Cantor row as a sibling uniformity route over the same finite data.

The refused objects have no coordinate in the carrier and are excluded by all three displayed theorem blocks. Therefore the accepted packet reads exactly the stated finite-net scope. □

Theorem 2516.8 (Heine-Borel finite-net subcover extraction)**.** *The finite subcover row consumes the finite-net interval carrier of chapter 1808 and the positive-radius witness of chapter 2301 before the extracted witness list reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Let $H = (I, F, L, C, R, D, E, T, Q, P, N)$ *be an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet. If the cover-read row* R *reaches the displayed finite net* F*, then the packet extracts a finite subcover row only through the route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow},\quad \mathsf{LebesgueNumberLemma}^{\uparrow},\quad \mathsf{HeineCantor}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{hsame},$$

The extracted row is the finite list of cover witnesses already recorded by R *for the net points and transported to interval points through* L, D, E, T, Q, P, N*. It is not an arbitrary open-cover*

compactness theorem, a selected subcover functional, a quotient-interval construction, a countable-choice principle, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

The same extracted finite subcover row is the cover input consumed by $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *through chapter 1897 when interval-image connectedness or compact interval-image routes need finite cover data.*

Proof. Project the carrier of Definition 3471.1. The only cover-facing rows are the located interval source I, the finite net F, the positive-radius ledger L, the sibling uniformity row C, the cover-read row R, the dyadic ledger D, and the Real seal E.

Use Definition 3471.2 to keep every accepted cover read inside the local NameCert obligations. In particular, the cover request must display the interval and finite-net rows before a witness from R can be inspected, and the radius compatibility check must pass through L and D.

Apply Definition 3471.3 to order the extraction. The finite list of witnesses is read from R along the located finite net, then transported across the Lebesgue-number and dyadic slack rows to the endpoint Real seal.

Finally apply Definition 3471.7. It keeps the $\mathsf{HeineCantor}^{\uparrow}$ row as a sibling compact-uniformity route and confines transport, replay, provenance, and naming to $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. Since the refused objects have no carrier coordinate, the extracted finite subcover row cannot be read as arbitrary compactness, selected choice, quotient interval data, countable choice, or ambient Real completeness. □

Theorem 2516.9 (Heine-Borel finite-net obligation-closure package)**.** *The exact obligation-closure surface of a* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet is the package consisting of the carrier of Definition 3471.1, the NameCert obligations of Definition 3471.2, the interval-cover route of Definition 3471.3, the obligation lattice of Definition 3471.6, the scope witness of Definition 3471.7, and the subcover extraction row of Definition 3471.8. Any downstream interval compactness consumer must read those rows as one finite packet over* $I, F, L, C, R, D, E, T, Q, P, N$ *and cannot replace them by arbitrary open-cover compactness, a selected subcover functional, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First project the carrier. It fixes the displayed rows $I, F, L, C, R, D, E, T, Q, P, N$ and leaves no coordinate outside the located interval, finite-net, positive-radius, sibling uniformity, cover-read, dyadic, Real-seal, transport, replay, provenance, and local naming data.

Next apply the NameCert obligations, interval-cover route, and obligation lattice. These rows force the cover request to pass through $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with $\mathsf{HeineCantor}^{\uparrow}$ available only as a sibling uniformity route over the same finite data.

Finally apply the scope witness and subcover extraction theorem. They identify the exportable finite subcover row as the witnessed cover reads already recorded by R and transported through L, D, E, T, Q, P, N. Since no refused object appears in the carrier, the displayed package is exactly the obligation-closure surface. □

Theorem 2516.10 (Heine-Borel finite-subcover extraction scope)**.** *Any finite subcover row exported by an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet remains scoped to* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *It may be read only through the displayed rows* $I, F, L, C, R, D, E, T, Q, P, N$*, so it is not arbitrary open-cover compactness, a choice-selected subcover, quotient interval equality, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 3471.8. The extracted row is already the finite list of witnesses recorded by R for the displayed finite net and transported to interval points through the positive-radius and dyadic slack rows.

Use Definition 3471.9 to keep that row inside the whole packet. The admissible dependencies are exactly the located finite-net source, Lebesgue-number radius, Heine-Cantor sibling route, dyadic ledger, Real seal, and structural $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ rows.

The refused readings have no carrier coordinate and no replay route. Therefore the exported finite subcover row cannot be detached from the finite-net interval certificate or promoted into unrestricted compactness or choice-selected subcover data. □

Theorem 2516.11 (Heine-Borel finite-net primitive induction route)**.** *Let $H = (I, F, L, C, R, D, E, T, Q, P, N)$ be an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet. The finite-net cell list displayed by F supports primitive finite induction from the first accepted cell through each subsequent* Cont *replay row in Q. At every step the induction reads only the current net cell, the corresponding radius check in L, the dyadic slack row in D, the retained cover read in R, and the endpoint seal in E. The resulting induction route is stable under componentwise* hsame *transport in T and remains inside the provenance and local naming ledger P, N.*

Proof. Project the carrier of Definition 3471.1. The finite list is the displayed row F, and the replay row Q records how a cover request advances through the listed cells.

The base step reads the first accepted cell together with its positive-radius row from L, dyadic slack row from D, cover witness from R, and endpoint seal from E. The successor step repeats the same displayed read for the next cell and adjoins the resulting witness to the finite subcover row.

Componentwise transport through T preserves each cell, radius, slack, cover-read, and endpoint-seal row. Since P and N name only the same displayed packet, the induction route cannot leave the local NameCert ledger or import an arbitrary open-cover compactness principle. □

Theorem 2516.12 (Heine-Borel choice-free located net)**.** *For every accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier*

$$H = (I, F, L, C, R, D, E, T, Q, P, N),$$

the extracted located-net cover row is read only from the displayed located interval source I, finite-net row F, Lebesgue-number row L, Heine-Cantor sibling row C, DyadicRatCore ledger D, Real seal E, componentwise hsame *transport T, displayed* Cont *replay Q,* Pkg *provenance P, and local* $\mathsf{NameCert}$ *row N. The route packages a finite located-net handoff and does not claim general open-cover compactness, unrestricted Heine-Borel compactness, selected subcover functionals, quotient interval representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite subcover extraction theorem Definition 3471.8. It identifies the exported row as the finite list of cover witnesses already recorded for the displayed finite net and transported across the positive-radius, dyadic, endpoint, structural, provenance, and naming rows.

The finite-subcover extraction scope of Definition 3471.10 keeps that row inside the located interval, Lebesgue-number, Heine-Cantor sibling, DyadicRatCore, Real, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface. Thus no external subcover selector or open-cover compactness object is available.

Finally apply the primitive induction route of Definition 3471.11. The witness list is assembled by finite induction over the displayed cells of F, with each step reading only the current radius check, dyadic slack row, retained cover read, and endpoint seal. Therefore the located-net handoff is choice-free and remains confined to the accepted packet rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ packet is generated from a located closed interval source, a LocatedClosedIntervalFiniteNet row, a LebesgueNumberLemma positive-radius ledger, a HeineCantor sibling route, finite cover reads, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3471.1, Definition 3471.2, and Definition 3471.3 seed the constructive finite-net interval-cover certificate over located interval, positive-radius, dyadic tolerance, Heine-Cantor sibling, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net interval exposure, radius compatibility, sibling uniformity compatibility, cover-read exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelFiniteNetUp.

Chapter 2517

A Concrete Naming Certificate for $\mathsf{HilbertBasis}^{\uparrow}$

Following Hilbert's basis theorem and constructive commutative algebra, the $\mathsf{HilbertBasis}^{\uparrow}$ packet records the finite generating rows needed to read a polynomial ideal as a named basis surface. It sits after the polynomial constructors of chapter 50, the ring and commutative-ring ledgers of chapter 43 and chapter 44, the ideal carrier of chapter 81, and the finite reduction route of chapter 2197.

Definition 2517.1 (HilbertBasis carrier)**.** *A* $\mathsf{HilbertBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{HB}(P, R, C, I, G, B, H, K, Q, N)$$

with the following rows.

- P *is the* $\mathsf{Poly}^{\uparrow}$ *source row for the ambient polynomial syntax and monomial ledger.*
- R *is the* $\mathsf{Ring}^{\uparrow}$ *source row for coefficient addition, multiplication, zero, one, and additive inverse.*
- C *is the* $\mathsf{CommRing}^{\uparrow}$ *row that makes coefficient multiplication commute for the displayed polynomial operations.*
- I *is the* $\mathsf{Ideal}^{\uparrow}$ *row naming the finite ideal-membership and generator surface.*
- G *is the* $\mathsf{GroebnerBasis}^{\uparrow}$ *reduction row used only as a finite normal-form witness.*
- B *is the displayed Hilbert-basis list of polynomial generators, together with the bounded reduction traces that place each ideal row under that list.*
- H, K, Q, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$ *naming rows.*

The packet has no coordinate for quotient ideals, host noetherianity, excluded middle over ideals, selected infinite generator search, or choice-produced basis extraction.

Lean4 ✓ `BEDC.Derived.HilbertBasisUp.HilbertBasisTasteGate single carrier alignment`

Theorem 2517.2 (HilbertBasis NameCert obligations)**.** *For an accepted* $\mathsf{HilbertBasis}^{\uparrow}$ *carrier* $\mathsf{HB}(P, R, C, I, G, B, H, K, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, polynomial-source compatibility through* P*, coefficient-ring compatibility through* R *and* C*, ideal generator access through* I*, finite reduction replay through* G*, basis-list exactness through* B*, componentwise transport through* H*, continuation replay through* K*, package provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 3472.1. The only rows available to the local certificate are $P, R, C, I, G, B, H, K, Q, N$.

The polynomial, ring, commutative-ring, and ideal rows supply the algebraic source surface. The Groebner row supplies finite reduction traces only after those source rows have been displayed.

The basis-list row B records the finitely many generators and the reductions that witness their coverage. Transport, continuation, provenance, and local naming then repack the same finite rows, so quotient ideals, host noetherianity, and choice data cannot enter the certificate. □

Theorem 2517.3 (HilbertBasis polynomial-ideal route)**.** *Every downstream consumer of an accepted* $\mathsf{HilbertBasis}^{\uparrow}$ *packet that asks for a polynomial-ideal basis must read the route*

$$\mathsf{Poly}^{\uparrow} \longrightarrow \mathsf{Ring}^{\uparrow} \longrightarrow \mathsf{CommRing}^{\uparrow} \longrightarrow \mathsf{Ideal}^{\uparrow} \longrightarrow \mathsf{GroebnerBasis}^{\uparrow} \longrightarrow \mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}.$$

The exported object is the finite generator row B together with its displayed reduction traces and structural rows H, K, Q, N; it is not a quotient ideal, not an infinite ascending-chain argument, not a host noetherian theorem, and not a selected basis obtained by choice.

Proof. Start with the polynomial row P and coefficient rows R, C. These determine the syntax and coefficient operations that an ideal consumer is allowed to inspect.

Next read the ideal row I and the finite Groebner row G. The ideal surface provides the finite membership and generator coordinates, while the Groebner route provides terminating reduction traces for the displayed basis list.

Finally apply the NameCert obligations of Definition 3472.2. They force the exported basis to remain inside B, H, K, Q, N, so no quotient ideal, no ambient noetherian principle, and no choice-produced generator search is exported. □

Falsifiable prediction: Every $\mathsf{HilbertBasis}^{\uparrow}$ consumer that claims a polynomial-ideal basis must exhibit the polynomial row, ring row, commutative-ring row, ideal row, Groebner reduction row, basis-list row, transport, continuation, provenance, and local naming row.

Independence witness: polynomial, ring, commring, ideal, groebner-basis: $\mathsf{HilbertBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries polynomial syntax, coefficient algebra, ideal membership, finite reduction traces, and a named basis-list surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{HilbertBasis}^{\uparrow}$ packet is generated from PolynomialUp syntax rows, RingUp and CommRingUp coefficient rows, an IdealUp generator surface, a GroebnerBasisUp reduction row, a finite basis-list row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$, seed)

Scope closed: Definition 3472.1, Definition 3472.2, and Definition 3472.3 seed the finite polynomial-ideal basis interface over PolynomialUp, RingUp, CommRingUp, IdealUp, GroebnerBasisUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient ideals, host noetherianity, excluded middle over ideals, selected infinite generator search, choice-produced basis extraction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial and coefficient compatibility, ideal generator access, finite reduction replay, basis-list exactness, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.HilbertBasisUp.

Chapter 2518

A Concrete Naming Certificate for $\mathsf{CantorBernstein}^{\uparrow}$

Following Cantor, Bernstein, and Schroeder, the $\mathsf{CantorBernstein}^{\uparrow}$ packet records the finite BEDC route by which two displayed injection classifiers determine one mutual-injection equivalence boundary. It sits beside the equality and comparison surfaces of chapter 1404, the stream-window classifier surface of chapter 126, and the governance naming discipline of chapter 9467. The packet exposes classifier rows, finite zig-zag decomposition data, BHist inspection, hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming; it is not a host quotient equality theorem, a selected bijection from ambient set theory, or a hidden choice principle.

Definition 2518.1 (Cantor–Bernstein carrier)**.** *A* $\mathsf{CantorBernstein}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, I, J, X, Y, Z, H, C, P, N)$$

with the following displayed rows:

1. *A is the first classifier surface, naming the observed source-side rows and their equality test before any target-side identification is read.*
2. *B is the second classifier surface, naming the observed target-side rows and their equality test before any source-side identification is read.*
3. *I is the displayed injection row from A into B, including source classifier preservation and target classifier reflection for every inspected finite row.*
4. *J is the displayed injection row from B into A, including target classifier preservation and source classifier reflection for every inspected finite row.*
5. *X is the finite zig-zag decomposition ledger. It separates forward chains, backward chains, anchored components, and unresolved boundary components using only the two displayed injection rows.*
6. *Y is the classifier-equivalence route read from X. It records which side supplies the representative for each finite zig-zag component without identifying host quotient classes.*
7. *Z is the mutual-injection boundary row exported to later classifier consumers.*
8. *H records componentwise* hsame *transport for the two classifier surfaces, the two injection rows, the zig-zag decomposition ledger, the classifier-equivalence route, and the exported boundary row.*

9. *C records displayed* Cont *replay from classifier inspection through injection preservation, zig-zag decomposition, and boundary export.*
10. *P is the* Pkg *provenance over* $\mathsf{CantorBernstein}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the sibling classifier chapters cited above.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$ *row.*

The carrier compares the two classifier surfaces only through the rows listed here. It exports no host quotient of classifiers, no ambient set equality, no selected global bijection, no arbitrary chain choice, and no completed StreamName or Real equivalence theorem.

Lean4 ✓ `BEDC.Derived.CantorBernsteinUp.CantorBernsteinTasteGate single carrier alignment`

Theorem 2518.2 (Cantor–Bernstein NameCert obligations)**.** *For every accepted* $\mathsf{CantorBernstein}^{\uparrow}$ *carrier*

$$K = (A, B, I, J, X, Y, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$ *surface consists exactly of carrier admission, two displayed classifier surfaces, two injection rows, finite zig-zag decomposition, classifier-equivalence routing, exported mutual-injection boundary data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the boundary row Z must factor through*

$$A, B, I, J \longmapsto X, Y, Z \longmapsto H, C, P, N.$$

Thus the mutual-injection theorem surface is available as finite BEDC certificate data while refusing host quotient equality, ambient set extensionality, arbitrary chain choice, and a selected global bijection.

Proof. Project the carrier of Definition 3473.1. The two classifier surfaces A and B are visible before either injection row is consumed, so no side is treated as a hidden quotient of the other.

Read the injections next. The row I records classifier preservation and reflection from A to B, while J records the corresponding route from B to A. These two rows are the only admissible source of mutual reachability data.

Build the finite zig-zag ledger X from the displayed reachability rows. The ledger separates chain components by finite inspection and hands the representative-side decision to Y without invoking an ambient chain selector.

The exported row Z is then a boundary for downstream classifier consumers. Transport by H and replay by C preserve the same displayed rows, while P and N record provenance and local naming. No additional quotient, extensionality, or choice row is present in the packet. □

Falsifiable prediction: Removing either displayed injection row or the finite zig-zag decomposition ledger prevents a consumer from distinguishing mutual-injection classifier equivalence from a host quotient or selected bijection.

Independence witness: regularcauchyequivalence, streamname, real: none is bijective to $\mathsf{CantorBernstein}^{\uparrow}$ because the packet jointly carries two classifier surfaces, two displayed injection rows, a finite zig-zag decomposition ledger, classifier-equivalence routing, boundary export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorBernstein}^{\uparrow}$ packet is generated from two displayed classifier surfaces, two injection rows, finite zig-zag decomposition data, classifier-equivalence routing, a mutual-injection boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CantorBernstein}\uparrow}$, seed)

Scope closed: Definition 3473.1 and Definition 3473.2 seed the finite mutual-injection classifier-equivalence boundary over displayed BEDC rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CantorBernstein}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient equality, ambient set extensionality, arbitrary chain choice, a selected global bijection, StreamName equivalence, Real equivalence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two classifier surfaces, two injection rows, finite zig-zag decomposition, classifier-equivalence routing, boundary export, transport, replay, provenance, and local naming for BEDC.Derived.CantorBernsteinUp.

Chapter 2519

A Concrete Naming Certificate for $\mathsf{HyperbolicMetric}^{\uparrow}$

Following Gromov's four-point metric viewpoint and Kelley's topology-side use of metric balls, the $\mathsf{HyperbolicMetric}^{\uparrow}$ packet realizes hyperbolic metric geometry as finite BEDC certificate data. It sits after the metric readback of chapter 57, the center-radius surface of chapter 1450, the ordered-real comparison rows of chapter 38, the topology handoff of chapter 88, and the bounded-set ledger of chapter 1757. The chapter names a finite four-point metric surface without turning the host disk, smooth hyperbolic manifold, ideal boundary, or ambient metric completion into a primitive model.

Definition 2519.1 (Hyperbolic metric carrier)**.** *A* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, b, t, \beta, q, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite* $\mathsf{Met}^{\uparrow}$ *carrier row whose displayed distance comparisons are the only metric data read by the packet.*
2. *d is the* $\mathsf{Real}^{\uparrow}$ *valued distance row, naming the finite comparisons used by the four-point ledger.*
3. *b is the* $\mathsf{MetricBall}^{\uparrow}$ *neighborhood row, exposing center-radius packets that can be transported without an ambient powerset.*
4. *t is the* $\mathsf{Topology}^{\uparrow}$ *handoff row, reading the ball-neighborhood surface as an indexed-open request.*
5. *β is the* $\mathsf{BoundedSet}^{\uparrow}$ *boundary ledger, recording bounded containment data for the displayed ball rows.*
6. *q is the four-point inequality row, comparing the three paired sums determined by four displayed metric observations.*
7. *F is the refusal row excluding smooth hyperbolic manifolds, quotient ideal boundaries, host geodesic completions, ambient Real completeness, physical cosmology, and internal observer-substance readings.*
8. *H records componentwise* hsame *transport for the metric carrier, distance, ball, topology, bounded-set, four-point, and refusal rows.*
9. *C records displayed* Cont *replay through finite metric observations, ball-neighborhood transport, topology handoff, bounded containment, four-point comparison, provenance, and refusal.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$ *row.*

The carrier exports a hyperbolic metric-read packet only through these rows. It contains no primitive smooth hyperbolic space, no quotient boundary, no completed geodesic model, no host powerset of balls, no ambient Real completeness principle, and no object-level observer substance.

Theorem 2519.2 (Hyperbolic metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier*

$$M = (X, d, b, t, \beta, q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$ *surface consists exactly of metric-carrier admission, real-valued distance readback, metric-ball neighborhood transport, topology handoff, bounded-set boundary ledger, four-point inequality exposure, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every metric consumer must factor through*

$$X, d, b \longmapsto t, \beta, q, F \longmapsto H, C, P, N.$$

Thus hyperbolic geometry is read as a finite four-point metric certificate, not as a primitive ambient model.

Proof. Project the carrier of Definition 3474.1. The metric carrier row X is visible before any distance is read, and the distance row d is tied to displayed $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ observations.

Read the ball, topology, and bounded rows next. The row b records center-radius neighborhoods, t exposes the topology-side indexed-open request, and β records the bounded-set boundary ledger without creating a host subset.

The four-point row q is the only hyperbolic comparison row. The refusal row F keeps smooth hyperbolic geometry, quotient boundary equality, completed geodesic space, physical cosmology, and ambient Real completeness outside the packet.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier has no coordinate for a primitive smooth hyperbolic space, quotient boundary, completed metric model, physical cosmology, or observer substance, no consumer can extract those readings from the NameCert surface. □

Lean4 ✓ `BEDC.Derived.HyperbolicMetricUp.HyperbolicMetricCarrier namecert obligations`

Theorem 2519.3 (Hyperbolic metric four-point obligation surface)**.** *For every accepted* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier* $M = (X, d, b, t, \beta, q, F, H, C, P, N)$*, carrier admission, four-point inequality exposure, ball-neighborhood transport, bounded-set boundary reading, topology handoff, and non-escape from smooth hyperbolic geometry or host metric completion are available only through the displayed rows*

$$X, d, b, t, \beta, q, F, H, C, P, N.$$

The row q may compare four displayed points by the usual hyperbolic four-point ledger, but it may not import a smooth manifold, an ideal-boundary compactification, a selected geodesic model, a host powerset of balls, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the projection of Definition 3474.1. It exposes X as the metric carrier and d as the real-valued distance row before any hyperbolic comparison is inspected.

The four-point surface is row q. It reads only the finite distance comparisons supplied by d, so the hyperbolic inequality is a ledger entry over displayed observations rather than a theorem about an ambient smooth model.

Ball-neighborhood transport uses b together with H and C. The same transported center-radius rows are handed to t for the topology request and to β for bounded-set boundary accounting.

The refusal row F is then preserved by the structural rows H, C, P, N. Hence every consumer receives a finite NameCert surface over metric, ball, real, topology, bounded-set, transport, replay, provenance, and naming rows, with no coordinate from which smooth hyperbolic geometry or host metric completion can be recovered. □

Falsifiable prediction: Omitting the metric carrier row, distance row, metric-ball row, or four-point row prevents carrier acceptance because the packet can no longer distinguish finite hyperbolic metric reading from a primitive host geometry.

Independence witness: metric, metricball, real, topology, boundedset: none is bijective to $\mathsf{HyperbolicMetric}^{\uparrow}$ because the packet jointly carries metric carrier data, real-valued distance reads, ball-neighborhood rows, topology handoff, bounded-set boundary accounting, four-point comparison, refusal exactness, transport, replay, package sourcing, and local naming.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial\ HyperbolicMetric}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicMetric}^{\uparrow}$ TasteGate instances. Those instances must keep $X, d, b, t, \beta, q, F, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicMetric}^{\uparrow}$ packet is generated from Metric carrier observations, Real distance reads, MetricBall neighborhoods, Topology handoff, BoundedSet boundary accounting, a four-point comparison ledger, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3474.1, Definition 3474.2, and Definition 3474.3 seed the finite four-point metric surface through metric, ball, real, topology, bounded-set, refusal, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close smooth hyperbolic manifold geometry, quotient boundary equality, host metric completion, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric rows, ball rows, real comparison rows, topology handoff, bounded-set boundary accounting, four-point ledger exactness, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicMetricUp.

Chapter 2520

A Concrete Naming Certificate for $\mathsf{DenseIsometryExtension}^{\uparrow}$

Following Bishop and Bridges on constructive completions and isometric extension, the $\mathsf{DenseIsometryExtension}^{\uparrow}$ packet records the finite surface by which an isometry on a dense displayed source is extended through the completion interface. It sits after the entry surface of chapter 1512, the extension-and-uniqueness surface of chapter 397, the separated comparison boundary of chapter 1411, the regular rational readback of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only the displayed dense-source extension route; it does not construct quotient completions, select limit representatives, assert host function equality, import ambient completeness, or add bridge checking.

Definition 2520.1 (Dense isometry extension carrier)**.** *A* $\mathsf{DenseIsometryExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, J, U, S, F, W, T, R, H, C, P, N)$$

with the following visible rows:

1. *D is the dense-source row, exposing the displayed source observations and their finite density witnesses.*
2. *J is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *entry row that sends dense-source observations into the completion-facing surface.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-and-uniqueness row attached to the same dense source and target route.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row that makes zero-distance agreement a displayed classifier fact.*
5. *F is the source isometry row, recording finite distance preservation on the dense source through* $\mathsf{Real}^{\uparrow}$ *readback rows.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect dense-source names.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling approximation requests.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row whose observations feed the terminal real seal.*
9. *H records componentwise* hsame *transport for dense source, completion entry, universal-property, separatedness, isometry, finite-window, tolerance, regular-rational, and real-seal rows.*

10. *C records displayed* Cont *replay through the same rows.*

11. *P is the* Pkg *provenance over* DenseIsometryExtension$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}$ *row.*

The carrier exports an extension service only through these rows. It contains no quotient completion object, no selected limit, no host function equality, no unrecorded completeness theorem, and no hidden choice schedule.

Lean4 ✓ `BEDC.Derived.DenseIsometryExtensionUp.TasteGate.DenseIsometryExtensionTasteGate single carrier alignment`

Theorem 2520.2 (Dense isometry extension NameCert obligations)**.** *For every accepted* DenseIsometryExtension$^{\uparrow}$ *carrier*

$$E = (D, J, U, S, F, W, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}$ *surface consists exactly of dense-source admission, completion-entry exposure, universal-property extension, separated comparison, source isometry preservation, finite-window exposure, dyadic tolerance accounting, regular-rational handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every extension consumer must factor through*

$$D, J, F, W, T, R \longmapsto U, S \longmapsto H, C, P, N.$$

Thus the extension is read as finite NameCert data over the dense source and its completion route, not as a host-level function on a quotient completion.

Proof. Project the carrier of Definition 3475.1. The dense-source row D, completion-entry row J, source isometry row F, finite-window row W, dyadic ledger T, and regular-rational row R are the only rows from which a source observation can be read.

Next use the universal-property row U. It repacks those displayed observations as an extension route, while the separated comparison row S supplies the zero-distance classifier boundary needed for uniqueness of the extended isometry.

Finally transport and replay are confined to H and C, with P and N recording provenance and local naming. Since the carrier has no coordinate for a quotient completion, selected representative, host function equality, ambient completeness theorem, or choice schedule, every accepted extension consumer is forced through the displayed finite route. □

Theorem 2520.3 (Dense isometry extension completion handoff)**.** *Let $E = (D, J, U, S, F, W, T, R, H, C, P, N)$ be an accepted* DenseIsometryExtension$^{\uparrow}$ *carrier. Any completion-facing consumer of the extended isometry first reads the dense-source observations through D, W, T, R, then reads the completion-entry row J, then reads the universal-property and separatedness rows U, S, and only then receives transport, replay, provenance, and naming through H, C, P, N.*

The handoff is therefore a finite completion-extension service. It exports no quotient completion equality, selected host limit, host function equality, countable-choice schedule, ambient Real$^{\uparrow}$ *completeness theorem, or bridge-checking credential.*

Proof. Begin with the source side. The dense-source row D is inspected only through the finite windows W, dyadic tolerances T, and regular-rational handoff R, so a completion-facing request cannot skip directly to a completed target.

The completion-entry row J is the first completion-facing coordinate. After J has exposed the source observation as completion-entry data, U supplies the extension-and-uniqueness ledger and S supplies the separated classifier boundary.

The suffix H, C, P, N does not add mathematical content beyond transport, replay, provenance, and local naming of the displayed rows. Therefore the completion handoff is exactly the stated finite route, with no coordinate from which a quotient completion, selected limit, host function, choice schedule, ambient completeness theorem, or bridge credential could be read. □

Falsifiable prediction: Every accepted $\mathsf{DenseIsometryExtension}^{\uparrow}$ consumer reads the dense-source row, completion-entry row, universal-property row, separated comparison row, source isometry row, finite-window schedule, dyadic tolerance ledger, regular-rational handoff, transport, replay, provenance, and local naming before any completion-facing extension service is exposed.

Independence witness: completionembedding, completionuniversalproperty, separatedmetric, regseqrat, streamname, dyadicratcore, real: none is bijective to $\mathsf{DenseIsometryExtension}^{\uparrow}$ because the packet jointly carries dense-source admission, completion entry, universal-property extension, separated uniqueness, isometry preservation, finite windows, tolerance accounting, regular-rational readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseIsometryExtension}^{\uparrow}$ packet is generated from dense-source observations, a CompletionEmbedding entry row, a CompletionUniversalProperty extension-and-uniqueness row, a SeparatedMetric comparison row, source isometry preservation, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal access, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3475.1, Definition 3475.2, and Definition 3475.3 seed the finite dense-isometry extension route through dense source, completion entry, universal property, separated comparison, source isometry, finite observation, tolerance, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected limit extraction, host function equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exposure, completion-entry handoff, universal-property extension, separated uniqueness, source isometry preservation, finite-window and tolerance accounting, regular-rational readback, and local naming for BEDC.Derived.DenseIsometryExtensionUp.

Chapter 2521

A Concrete Naming Certificate for $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$

Following Bishop and Bridges' constructive completion theory and Mac Lane's finite-product interface, the $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ packet records the finite BEDC surface at which completion reads commute with binary product reads. It consumes the Cauchy completion monad route of chapter 378, the metric-completion surface of chapter 1506, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the product naming envelope of chapter 34. The packet names a finite product-preservation handoff; it does not assert quotient completion equality, selected limits, countable choice, or an ambient completed product.

Definition 2521.1 (Cauchy completion product-preservation carrier)**.** *A* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{CPP}(L, R, M, \Pi, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left source-completion route, read through the Cauchy completion monad and the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *source.*
2. *R is the right source-completion route, read through the same completion discipline for the second source.*
3. *M is the product metric row, recording the finite product carrier, projection rows, and coordinatewise tolerance ledger.*
4. *Π is the product projection row, exposing the two completed-coordinate projections and their recombination read.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the paired finite windows and coordinatewise regular rational data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the coordinate tolerances and the product distance read.*
7. *H stores componentwise* hsame *transport for the two completion routes, product row, projection row, readback, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a product-completion request through L, R, M, Π, S, E.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, R, M, \Pi, S, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, ambient product completion, or host equality of completed points.

Lean4 ✓ BEDC.Derived.CauchyCompletionProductPreservationUp.CauchyCompletionProductPreservationTasteGate
single carrier alignment

Theorem 2521.2 (Cauchy completion product-preservation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ *carrier* $\mathsf{CPP}(L, R, M, \Pi, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *surface consists exactly of carrier admission for the two source completion routes, product metric row, projection row, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows; coordinate compatibility through L, R, M, Π; readback compatibility through S, E; and non-escape from quotient completions, selected limits, countable choice, ambient completed products, and host equality.*

Proof. Carrier admission is projection from Definition 3477.1. The displayed tuple exposes exactly the two completion routes, product row, projection row, readback row, Real seal, and structural rows.

Coordinate compatibility is forced by L, R, M, Π. The left and right completion routes are read before the product projection row, and the product row records only the coordinatewise product carrier and finite tolerance ledger.

Readback compatibility is then carried by S and E. The regular rational row reads paired finite windows, and the terminal Real seal presents only the coordinate tolerances and the product distance read produced from those windows.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, ambient product completion, or host equality, the listed obligations exhaust the local certificate surface. □

Theorem 2521.3 (Cauchy completion finite-product handoff)**.** *Every downstream consumer that reads completion after a finite product, or a finite product after two completion reads, must factor through the same displayed route*

$$\mathsf{CauchyCompletionMonad}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow} \longrightarrow \mathsf{Prod}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The exported object is the finite product-preservation handoff carried by $L, R, M, \Pi, S, E, H, C, P, N$, not a quotient-level isomorphism of completions and not a selected-limit construction.

Proof. Start with the accepted carrier of Definition 3477.1. The only completion-facing rows are L and R, and both are routed through the Cauchy completion monad and the metric-completion source surface.

The product read is M, Π. These rows expose the finite product carrier, coordinate projections, recombination read, and coordinatewise tolerance ledger without adding an ambient completed product.

The numerical handoff is S, E. Paired finite windows are read through $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal records the coordinate tolerances and product distance read.

Apply Definition 3477.2. The transport, replay, provenance, and local naming rows preserve exactly that route, so every finite-product completion consumer factors through the displayed handoff and cannot export quotient equality or selected limits. □

Theorem 2521.4 (Cauchy completion product-preservation scoped ledger)**.** *Every accepted* CauchyCompletionProd *packet factors its visible product-preservation read through exactly the following scoped ledger:*

1. *the two completion routes L and R, each read through* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$*;*
2. *the product metric row M and projection row Π, including the displayed finite product carrier, coordinate projections, recombination read, and coordinatewise tolerance ledger;*
3. *the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row S and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E;*
4. *componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *row N.*

No accepted packet exports quotient completion equality, selected-limit data, countable-choice schedules, ambient completed products, or host equality of completed points outside this ledger.

Proof. Start from Definition 3477.3. The handoff route already forces every completion-product read through the displayed completion, metric, product, regular-rational, and Real rows.

The carrier of Definition 3477.1 names the ledger entries. The rows L and R are the two source completion routes, while M and Π are the only product metric and projection rows.

The numerical readback is confined to S and E. These rows pair the regular rational windows and then seal the coordinate tolerances and product distance read at the $\mathsf{Real}^{\uparrow}$ boundary.

Finally H, C, P, N preserve the same ledger under transport, replay, provenance, and local naming. Since no other carrier coordinate exists, the scoped ledger is exact and the excluded quotient, selection, choice, ambient-product, and host-equality data cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ packet is generated from two CauchyCompletionMonad and MetricCompletion source routes, a product metric row, projection rows, RegSeqRat paired-window readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$, seed)

Scope closed: Definition 3477.1, Definition 3477.2, Definition 3477.3, and Definition 3477.4 seed the finite completion-product handoff over CauchyCompletionMonadUp, MetricCompletionUp, ProdUp, RegSeqRatUp, RealUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, ambient completed products, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coordinate compatibility, paired RegSeqRat readback, Real seal exactness, projection non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionPro

Chapter 2522

A Concrete Naming Certificate for $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$

The $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ packet realizes the finite support-closure algebra of Definition 9462.1609 and Definition 9462.1618 as BEDC NameCert data. It consumes the boundary-tile row of chapter 3393, the coverage-rank row of chapter 3470, the finite observation interface of chapter 1007, the observer support lattice of chapter 1122, and the finite residue boundary of chapter 1167. The packet names closure of visible trigger supports under finite windows; it is not a hidden global semantics, stochastic evolution model, quotient codon space, or infinite coverage oracle.

Definition 2522.1 (Window coverage-trigger algebra carrier)**.** *A* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, C, F, L, R, A, H, Q, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *trigger-boundary row exposing the finite window classifiers whose supports are read.*
2. *C is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *coverage row, recording which visible triggers cover which displayed support faces.*
3. *F is the* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *row that bounds every trigger read by a selected finite window and an omitted-tail ledger.*
4. *L is the* $\mathsf{ObserverSupportLattice}^{\uparrow}$ *support-lattice row, used only as a finite inclusion and coverage-rank ledger.*
5. *R is the* $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$ *residue row, recording which support faces remain outside the current closure.*
6. *A is the trigger-algebra row, carrying the finite support set, closure set, blocker row, coverage threshold row, and handoff to the $(4, 2)$ codon instance when that instance is requested.*
7. *H records componentwise* hsame *transport for the trigger-boundary, coverage, finite-window, support-lattice, residue, algebra, replay, provenance, and naming rows.*
8. *Q records displayed* Cont *replay from a coverage-trigger request through T, C, F, L, R, A.*

9. *P is the* Pkg *provenance over* WindowCoverageTriggerAlgebra$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, FiniteObservationInterface$^{\uparrow}$, ObserverSupportLattice$^{\uparrow}$, FiniteObservationResidueBounda *the source vision of Definition 9462.1609,* BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{WindowCoverageTriggerAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, F, L, R, A, H, Q, P, N$. It has no coordinate for hidden biological semantics, stochastic transition laws, quotient codon equality, global support completion, infinite search over triggers, or a final coverage oracle.

Theorem 2522.2 (Window coverage-trigger algebra NameCert obligations)**.** *For every accepted* WindowCoverageTriggerAlgebra$^{\uparrow}$ *carrier $W = (T, C, F, L, R, A, H, Q, P, N)$, the local* NameCert$_{\text{WindowCoverageTriggerAlg}}$ *surface consists exactly of boundary-trigger admission through T, coverage-rank admission through C, finite-window exposure through F, support-lattice inclusion through L, residue-boundary accounting through R, algebra-row exactness through A, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.WindowCoverageTriggerAlgebraUp.WindowCoverageTriggerAlgebraTasteGate`
`single carrier alignment`

Proof. Carrier admission is projection from Definition 3478.1. The accepted packet exposes only the boundary trigger, coverage, finite-window, support-lattice, residue, trigger-algebra, transport, replay, provenance, and naming rows.

The trigger read starts with T, C, F. A consumer first sees the displayed boundary-tile classifiers, then the coverage-rank row, and then the finite observation interface that records the selected window and omitted-tail ledger.

The support and residue stages are L, R. The support lattice provides finite inclusion and coverage-rank comparison, while the residue row records which faces remain outside the displayed closure.

The algebra row A may then package the support set, closure set, blocker row, threshold row, and $(4, 2)$ handoff. The structural rows H, Q, P, N preserve only this displayed route, so hidden semantics, stochastic laws, quotient codon equality, global support completion, infinite trigger search, and final coverage oracles have no carrier coordinate. □

Theorem 2522.3 (Window coverage-trigger vision handoff)**.** *Every concrete read of the universal coverage-trigger system through* WindowCoverageTriggerAlgebra$^{\uparrow}$ *factors through the finite route*

DnaWindowSixBoundaryTile$^{\uparrow}$ $\longrightarrow$ SixBoundaryTileCoverage$^{\uparrow}$ $\longrightarrow$ FiniteObservationInterface$^{\uparrow}$ $\longrightarrow$ ObserverSupportLa

The exported object is the finite support-closure trigger algebra carried by $T, C, F, L, R, A, H, Q, P, N$, with the source vision used only as provenance for the displayed closure, blocker, threshold, and codon-instance rows.

Proof. Start with the coverage-trigger definitions in Definition 9462.1609. The concrete packet does not import the whole vision as an oracle; it reads only the finite trigger-boundary and coverage rows T, C.

Next bind finite access through F. The selected window and omitted-tail ledger determine which trigger supports are visible before any closure row can be exported.

Then read L and R. The support lattice records finite inclusion and coverage rank, while the residue boundary records the uncovered faces that remain outside the current closure.

Finally package the closure, blocker, threshold, and codon-instance handoff through A, H, Q, P, N. Since these rows preserve the displayed finite route, every concrete read of the universal coverage-trigger algebra factors through the NameCert packet rather than through hidden semantics, quotient codon equality, stochastic dynamics, or infinite trigger search. □

Theorem 2522.4 (Window coverage-trigger algebra Markov source route)**.** *Every Markov-trigger-facing consumer of* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *must read the boundary-trigger row of chapter 3393, the coverage-rank row of chapter 3470, the finite-observation window row of chapter 1007, the support-lattice row of chapter 1122, and the residue-boundary row of chapter 1167 before any Markov state transition row can be formed.*

Thus the rank-sensitive transition surface of Definition 9462.1541 is reached only after the finite support-closure route $T, C, F, L, R, A, H, Q, P, N$ *has exposed boundary triggers, coverage rank, finite windows, support inclusion, and residue accounting. The route exports no stochastic evolution law, hidden biological semantics, quotient codon equality, global support completion, or infinite trigger search.*

Proof. Start with a consumer that requests a Markov-trigger-facing transition row. The carrier of Definition 3478.1 exposes no transition coordinate before the boundary-trigger, coverage, finite-window, support-lattice, and residue rows have been read.

The source route is therefore T, C, F, L, R. These rows are supplied by the five sibling concrete chapters named in the statement and fix the displayed boundary triggers, rank data, finite observation window, support inclusion, and residue boundary before any transition-facing handoff is admissible.

Only after that route is displayed may A, H, Q, P, N package the trigger algebra for the rank-sensitive surface of Definition 9462.1541. Since the carrier contains no stochastic evolution row, hidden biological semantics, quotient codon equality, global support completion, or infinite trigger-search coordinate, none of those objects can be introduced by the Markov source route. □

Falsifiable prediction: Every accepted $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ consumer that claims coverage-trigger closure must exhibit the boundary-trigger row, coverage-rank row, finite-observation row, support-lattice row, residue-boundary row, algebra row, componentwise transport, replay, provenance, and local naming row. A consumer omitting any of probe-bundle, finite-observation, support-lattice, namecert, or trigger-boundary sibling slugs refutes the packet.

Independence witness: probe-bundle, finite-observation, support-lattice, namecert, trigger-boundary, six-boundary-tile-coverage: $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boundary triggers, coverage rank, finite-window access, support closure, residue accounting, algebraic blocker and threshold rows, transport, replay, provenance, and local naming.

The later Lean route for `BEDC.Derived.WindowCoverageTriggerAlgebraUp` must provide explicit $\mathsf{Nontrivial}\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ instances preserving $T, C, F, L, R, A, H, Q, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ packet is generated from boundary-trigger rows, coverage-rank rows, finite observation windows, support-lattice inclusion rows, residue-boundary rows, a trigger-algebra row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3478.1, Definition 3478.2, and Definition 3478.3 seed the finite support-closure trigger algebra over boundary tiles, coverage rows, finite observation windows, support lattices, residue boundaries, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{WindowCoverageTriggerAlgebra}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close hidden biological semantics, stochastic transition laws, quotient codon equality, global support completion, infinite trigger search, final coverage oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-trigger exactness, coverage-rank exactness, finite-window non-escape, support-lattice inclusion, residue accounting, Nontrivial WindowCoverageTriggerAlgebraUp, and FieldFaithful WindowCoverageTriggerAlgebraUp for BEDC.Derived.WindowCoverageTriggerAlgebraUp.

Chapter 2523

A Concrete Naming Certificate for $\mathsf{BorelFunction}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on Borel maps between topological and real-facing spaces, the $\mathsf{BorelFunction}^{\uparrow}$ packet records the finite pullback route by which a displayed source topology reads target Borel rows. It sits between the generated-set boundary of chapter 3384, the measurable-map preimage surface of chapter 3461, the terminal real-name seal of chapter 38, and the topology interface of chapter 88. The packet names Borel pullback stability as a local BEDC certificate, not as a host inverse-image operation, quotient equality of functions, countable-choice enumeration of Borel codes, or ambient measure completion.

Definition 2523.1 (BorelFunction carrier)**.** *A* $\mathsf{BorelFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, B, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the source topology row read through* $\mathsf{Topology}^{\uparrow}$*, including the finite open-neighbourhood data needed by the current source-side query.*
2. *B is the target* $\mathsf{BorelSet}^{\uparrow}$ *row whose generated-set boundary supplies the target event being pulled back.*
3. *R is the pullback ledger assigning the displayed target Borel row in B to a displayed source Borel row over X.*
4. *Q is the* $\mathsf{Real}^{\uparrow}$ *readback row used when the target Borel query is real-facing; it is consumed only through the displayed Borel row and the pullback ledger.*
5. *H stores componentwise* hsame *transport for the source topology, target Borel row, pullback ledger, real readback, continuation, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a source query through the target Borel row, the pullback ledger, and the real-facing readback row.*
7. *P is the* Pkg *provenance over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, R, Q, H, C, P, N. It has no coordinate for host set preimages, quotient equality of functions, countable-choice selection of Borel codes, completed measure spaces, almost-everywhere equality, or an ambient function space.

Lean4 ✓ `BEDC.Derived.BorelFunctionUp.BorelFunctionTasteGate single carrier alignment`

Theorem 2523.2 (BorelFunction NameCert obligations)**.** *For the carrier of Definition 3479.1, the local* $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$ *surface consists exactly of source-topology admission, target Borel-row admission, pullback-ledger exactness, real-facing readback through* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted Borel-function consumer must read the source row X, the target Borel row B, and the pullback ledger R before any* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{MeasurableFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or topology-facing sibling row is exposed.*

Proof. Project the carrier $F = (X, B, R, Q, H, C, P, N)$. The rows X and B are the only source-topology and target-Borel coordinates, and the row R is the only coordinate that relates them.

Read a real-facing target query through B and Q. The readback row Q is downstream of the displayed Borel row, while the source-side result is named only by the pullback ledger R over X.

The structural rows add no host inverse-image operation. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route, so the excluded quotient, choice, completion, almost-everywhere, and ambient function-space objects remain outside the packet. □

Theorem 2523.3 (BorelFunction pullback stability)**.** *Let $F = (X, B, R, Q, H, C, P, N)$ and*

$$F' = (X', B', R', Q', H', C', P', N')$$

be accepted $\mathsf{BorelFunction}^{\uparrow}$ *carriers. If the source topology rows, target* $\mathsf{BorelSet}^{\uparrow}$ *rows, pullback ledgers, and real-facing readback rows agree componentwise through the displayed* hsame *transports, then every accepted source-side Borel read produced from F is transported to the corresponding read produced from F'. The comparison uses only*

$$X, B, R, Q, H, C, P, N$$

and

$$X', B', R', Q', H', C', P', N'.$$

It exports pullback stability as a finite certificate, not as host set preimage equality, quotient equality of functions, countable-choice enumeration, completed measure semantics, almost-everywhere equality, or ambient function equality.

Proof. Apply Definition 3479.2 to both carriers. Any accepted target Borel read first passes through the displayed target rows B and B', then through the pullback ledgers R and R'.

Use the componentwise transports. The hypotheses identify the source topology rows, target Borel rows, pullback ledgers, and real readback rows, while H, C, P, N and H', C', P', N' preserve, replay, source, and name those same routes.

The transported consumer sees only the displayed coordinates. Since neither carrier contains a host inverse-image coordinate, quotient-function equality, choice-selected code sequence, completed measure, almost-everywhere class, or ambient function object, the stability claim is exactly the preservation of the finite pullback ledger. □

Falsifiable prediction: A $\mathsf{BorelFunction}^{\uparrow}$ consumer that obtains a host set preimage, a quotient equality of functions, a countable-choice enumeration of Borel codes, a completed measure, or an almost-everywhere representative without first reading the displayed pullback ledger R refutes the carrier surface.

Independence witness: borelset, measurablefunction, real, topology: no listed sibling supplies the whole eight-row packet $F = (X, B, R, Q, H, C, P, N)$, because each sibling contributes only generated-set, measurable-map, real-readback, or source-topology data while this chapter binds the Borel pullback ledger with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3479.1, Definition 3479.2, and Definition 3479.3 seed the finite Borel-function pullback boundary over source topology, target Borel rows, real readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, almost-everywhere equality, quotient function spaces, countable-choice Borel codes, host set preimages, ambient function equality, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, pullback-ledger exactness, real-readback preservation, componentwise transport, and ledger-refusal proofs for BEDC.Derived.BorelFunctionUp.

Constructive story: A $\mathsf{BorelFunction}^{\uparrow}$ packet is generated from a source topology row, a target BorelSet row, a pullback ledger, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2524

A Concrete Naming Certificate for $\mathsf{ChebyshevPolynomial}^{\uparrow}$

Following Chebyshev and the classical approximation-theory presentation of Chebyshev polynomials, the $\mathsf{ChebyshevPolynomial}^{\uparrow}$ packet names the finite recurrence surface for the displayed family T_n without importing an ambient real-function algebra. It sits after the terminal real-name seal of chapter 38, the polynomial carrier of chapter 50, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the stream-window surface of chapter 126. The packet supplies a traditional approximation-theory object as finite NameCert data for later minimax and constructive approximation routes.

Definition 2524.1 (Chebyshev polynomial carrier)**.** *A* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(d, T_0, T_1, R, W, Q, H, C, P, N).$$

Here d is the unary degree row, T_0 and T_1 are the displayed recurrence seeds, R is the recurrence-step ledger for $T_{n+1} = 2xT_n - T_{n-1}$, W is the dyadic evaluation-window schedule, Q is the $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row, H records componentwise* hsame *transport, C records* Cont *replay of the recurrence route, P records* Pkg *provenance from the polynomial and real-name sources, and N is the local* $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$ *row. A consumer of the packet may inspect a Chebyshev read only through these displayed rows.*

Lean4 ✓ `BEDC.Derived.ChebyshevPolynomialUp.ChebyshevPolynomialTasteGate single carrier alignment`

Theorem 2524.2 (Chebyshev polynomial NameCert obligations)**.** *For every accepted* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *carrier $(d, T_0, T_1, R, W, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 3480.1. The degree row chooses the recurrence depth, the seed rows display T_0 and T_1, the recurrence ledger displays each finite step, the dyadic window and regular readback rows mediate evaluation, and the transport, replay, provenance, and local naming rows export the resulting polynomial certificate. The packet imports no host polynomial equality, selected infinite family, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet rows in order. The degree row d bounds the visible recurrence segment, so the seed rows and recurrence ledger already determine every displayed polynomial row consumed by the packet.

The evaluation side factors through W and Q. The dyadic windows give the finite tolerance requests, while the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal keep coefficient and value reads inside the existing real-name route.

Finally use H, C, P, N. These rows transport equal displayed histories, replay the recurrence computation, remember the polynomial and real-name provenance, and attach the local certificate name. No additional host equality or infinite polynomial family is available to a consumer outside the listed rows. □

Theorem 2524.3 (Chebyshev recurrence handoff)**.** *Every accepted* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *recurrence read factors through*

$$d \longmapsto T_0, T_1 \longmapsto R \longmapsto W \longmapsto Q \longmapsto H, C, P, N$$

before any approximation-facing consumer receives a polynomial value or coefficient row. Thus the Chebyshev recurrence is handed to later constructive approximation packets as finite degree, seed, recurrence, dyadic-window, real-readback, transport, replay, provenance, and naming data, not as an external analytic theorem about an infinite sequence of polynomials.

Proof. Start with the carrier of Definition 3480.1. The only recurrence coordinates exposed by the packet are d, T_0, T_1, R, so any accepted recurrence read is bounded by the displayed unary degree and the finite seed-and-step ledger.

Next read the evaluation interface. The route through W and Q forces every value-facing request to pass through dyadic windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal before it can be consumed as approximation data.

The remaining rows provide structural export. Componentwise hsame transport, Cont replay, Pkg provenance, and the local NameCert row preserve the same finite recurrence route for downstream minimax, interpolation, and approximation packets. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChebyshevPolynomial}^{\uparrow}$ packet is generated from a unary degree row, recurrence seed rows, a finite recurrence-step ledger, dyadic evaluation windows, regular rational and Real readback, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$, obligation)

Scope closed: Definition 3480.1, Definition 3480.2, and Definition 3480.3 declare the finite recurrence certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, minimax optimality, Markov or Bernstein inequalities, Jackson approximation, host polynomial equality, selected infinite families, or ambient RealUp completeness.

Upgrade path: Checked closure requires Lean carrier admission, recurrence ledger stability, evaluation-window readback, and recurrence handoff for the same finite packet.

Chapter 2525

A Concrete Naming Certificate for $\mathsf{CellularWindowTrace}^{\uparrow}$

The $\mathsf{CellularWindowTrace}^{\uparrow}$ packet realizes the cellular-carrier alternative of chapter 9391 as finite orbit-window NameCert rows while leaving the BEDC kernel carrier as BHist. It consumes the cellular automaton and substrate readings of chapter 934 and chapter 942, the finite BHist encoding discipline of chapter 943, the stream-window surface of chapter 126, and the bounded universal trace boundary of chapter 944. The packet names a finite window of a cellular orbit, not a replacement foundation for BEDC histories.

Definition 2525.1 (Cellular window trace carrier)**.** *A* $\mathsf{CellularWindowTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(w, r, i, o, B, H, C, P, N).$$

Here w is the displayed width row, r is the local-rule row, i is the initial cell-row encoding, o is the finite orbit-window ledger, B is the BHist *encoding of the selected rows, H records componentwise* hsame *transport, C records* Cont *replay of the local update route, P records* Pkg *provenance from the cellular-substrate sources, and N is the local* $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$ *row.*

Theorem 2525.2 (Cellular window trace NameCert obligations)**.** *For every accepted* $\mathsf{CellularWindowTrace}^{\uparrow}$ *carrier $(w, r, i, o, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$ *surface consists exactly of width, local-rule, initial-row, finite orbit-window,* BHist *encoding, transport, replay, provenance, and local naming rows. A consumer may read the cellular window only by first consuming these rows; it receives no global orbit, no infinite cellular history, no substrate replacement of* BHist*, and no bijection between cellular traces and all BEDC histories.*

Lean4 ✓ `BEDC.Derived.CellularWindowTraceUp.CellularWindowTraceTasteGate single carrier alignment`

Proof. Project the displayed carrier rows. The width and local-rule rows determine which finite cell row can be updated, and the initial row gives the only starting configuration available to the packet.

The orbit-window ledger o records the finite segment that is actually visible. Its BHist encoding row B translates that segment into the BEDC kernel carrier without changing the carrier used by the surrounding certificate.

Finally H, C, P, N export the route. Transport preserves the displayed rows, replay records the local update computation, provenance ties the route to the cellular-substrate sources, and the local certificate names this finite window only. □

Theorem 2525.3 (Cellular window trace finite-read factorization)**.** *Every accepted read of a* $\mathsf{CellularWindowTrace}^{\uparrow}$ *packet factors through*

$$w \longmapsto r \longmapsto i \longmapsto o \longmapsto B \longmapsto H, C, P, N.$$

Thus the cellular-carrier alternative is exposed as finite orbit-window data before any downstream consumer can treat it as a cellular trace.

Proof. Read the packet in dependency order. Width and local rule are needed before the initial row can be updated, and the finite orbit window is the only displayed result of those updates.

The BHist encoding row then returns the trace to the BEDC carrier. Since transport, replay, provenance, and naming are all downstream of that encoding, the public consumer sees a finite cellular-window certificate rather than a direct cellular replacement for BHist. □

Falsifiable prediction: Every accepted cellular-window read of $\mathsf{CellularWindowTrace}^{\uparrow}$ factors through width, local-rule, initial-row, finite orbit-window, BHist encoding, hsame transport, Cont replay, Pkg provenance, and local naming rows before any downstream cellular-trace consumer can inspect it.

Independence witness: cellularsubstrate, cellularautomaton, streamname, bhist-arity16-eventflow, boundeduniversaltrace: the cellular-window packet is not bijective with the BEDC carrier. It records one finite orbit window with a displayed encoding, while BHist supplies the kernel carrier independently of any chosen cellular width, rule, initial row, or orbit segment.

On the Lean side, the formal carrier should supply ‘Nontrivial CellularWindowTraceUp‘ and ‘FieldFaithful CellularWindowTraceUp‘ instances in ‘lean4/BEDC/Derived/CellularWindowTraceUp/TasteGate.le

Closure status for $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CellularWindowTrace}^{\uparrow}$ packet is generated from a width row, local-rule row, initial-row encoding, finite orbit-window ledger, BHist encoding, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$, obligation)

Scope closed: Definition 3481.1, Definition 3481.2, and Definition 3481.3 declare the finite orbit-window certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CellularWindowTraceUp.CellularWindowTraceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim a global cellular orbit, an infinite cellular history, a substrate replacement for BHist, a bijection between cellular traces and all BEDC histories, or bridge checking.

Upgrade path: Checked closure requires Lean carrier admission, finite orbit-window factorization, BHist encoding transport, and TasteGate instance rows for the same finite packet.

Chapter 2526

A Concrete Naming Certificate for $\mathsf{WeakKonigBoundary}^{\uparrow}$

Following Kleene and Bishop–Bridges on finite binary trees, Cantor-space paths, and constructive real completion, the $\mathsf{WeakKonigBoundary}^{\uparrow}$ packet records the boundary at which finite tree inspection remains available while weak König path selection is refused as a BEDC primitive. It sits after the finite obstruction surface of chapter 1622, the Cantor and FanTheorem interfaces of chapter 1475 and chapter 400, the finite window and regular readback route of chapter 126 and chapter 301, and the completion-facing seal rows of chapter 38 and chapter 1714. The packet names the finite-tree/path-selection boundary used by Real and Completion consumers without turning that boundary into countable choice, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2526.1 (WeakKonigBoundary carrier)**.** *A* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, A, F, S, Q, R, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{KleeneTree}^{\uparrow}$ *finite-prefix tree packet, including the decidable node ledger and obstruction row of Definition 1622.1.*
2. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix observation row offered to consumers that inspect binary path fragments.*
3. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar test row available only as displayed finite cover data, not as open-cover compactness.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which every path-like request must pass.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from displayed finite windows before any Real-facing seal is inspected.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *and completion-facing seal row reached only after K, A, F, S, Q have been displayed.*
7. *O is the boundary-refusal row: no infinite branch selector, no countable-choice functional, no quotient stream equality, no open-cover compactness principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of the packet.*

8. *H records componentwise* hsame *transport for the tree packet, Cantor prefix row, fan row, stream windows, regular readback, Real seal, refusal row, continuation, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree or path request through $K, A, F, S, Q, R, O, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{WeakKonigBoundary}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite tree/path boundary packet. It has no coordinate for an infinite branch, a path-selection functional, countable choice, quotient stream equality, open-cover compactness, selected completion limits, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.WeakKonigBoundaryUp.TasteGate.WeakKonigBoundaryTasteGate single carrier alignment`

Theorem 2526.2 (WeakKonigBoundary NameCert obligations)**.** *For every accepted* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier $W = (K, A, F, S, Q, R, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{KleeneTree}^{\uparrow}$ *packet,* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, refusal row, transport, replay, provenance, and local naming;*
2. *tree-window exactness, requiring every path-like read to pass through the finite tree row K and finite window row S before any Cantor, FanTheorem, RegSeqRat, Real, or Completion-facing row is exposed;*
3. *regular-readback discipline, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q to be obtained from displayed finite windows rather than from quotient stream equality;*
4. *seal nonescape, requiring the Real and completion-facing row R to remain downstream of K, A, F, S, Q and the refusal row O;*
5. *boundary refusal, excluding infinite branch selection, countable choice, quotient stream equality, open-cover compactness, and ambient Real completeness from the packet.*

No WeakKonigBoundary obligation is stored outside these rows.

Proof. Project the carrier of Definition 3482.1. The projection exposes exactly the Kleene tree packet K, Cantor prefix row A, FanTheorem finite-bar row F, stream windows S, regular readback Q, Real seal R, refusal row O, and structural rows H, C, P, N.

Tree-window exactness is inherited from the finite route displayed by Definition 1622.3. A path-like request first reads the finite tree and the finite stream window; it cannot jump directly to an infinite branch or completed Cantor point.

The regular-readback and seal rows are downstream. The row Q is admitted only after the finite windows are fixed, and the row R is reached only after K, A, F, S, Q and the refusal row have been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier rows stores an infinite branch selector, countable-choice functional, quotient stream equality, open-cover compactness object, or ambient Real-completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 2526.3 (WeakKonigBoundary Real-completion refusal)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier factors through the finite route*

$$K, A, F, S, Q, R, O, H, C, P, N.$$

Consequently the consumer may reuse finite Kleene-tree prefixes, Cantor-space observations, Fan-Theorem finite-bar rows, StreamName windows, and RegSeqRat readbacks, but it receives no infinite branch selection, no countable-choice schedule, no quotient stream equality, no open-cover compactness principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with a Real or Completion-facing read of $W = (K, A, F, S, Q, R, O, H, C, P, N)$. Carrier admission gives the consumer the tree, Cantor, fan, stream-window, regular-readback, and seal rows only as the displayed finite packet.

Use the Kleene-tree boundary from Definition 1622.10. Any Real-facing interpretation of tree data must already pass through finite $\mathsf{StreamName}^{\uparrow}$ windows and the obstruction row before a real seal is inspected.

The $\mathsf{WeakKonigBoundary}^{\uparrow}$ carrier adds the refusal row O at the same boundary. Thus the completion-facing row R can be read only after the finite tree/path route and cannot supply a path selector, quotient stream equality, open-cover compactness, or ambient Real completeness.

Transport and replay through H and C, with provenance and naming through P and N, preserve the displayed route. Therefore every Real or Completion consumer receives the finite boundary package and none of the excluded principles. □

Falsifiable prediction: A $\mathsf{WeakKonigBoundary}^{\uparrow}$ consumer that obtains an infinite branch selector, countable-choice schedule, quotient stream equality, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem without first reading the displayed route $K, A, F, S, Q, R, O, H, C, P, N$ refutes the carrier surface.

Independence witness: kleene-tree, cantorspace, fantheorem, streamname, regseqrat, real, separable-completion: $\mathsf{WeakKonigBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite tree packet, Cantor prefix row, fan row, stream windows, regular readback, Real seal, refusal row, transport, replay, provenance, and local NameCert row in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3482.1, Definition 3482.2, and Definition 3482.3 seed the finite tree/path-selection boundary separating Real and Completion-facing reads from weak König path selection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak König's lemma, infinite branch selection, countable choice, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree-window exactness, regular-readback discipline, seal nonescape, and boundary-refusal proofs for BEDC.Derived.WeakKonigBoundaryUp.

Constructive story: A $\mathsf{WeakKonigBoundary}^{\uparrow}$ packet is generated from a KleeneTreeUp finite tree packet, CantorSpace finite-prefix row, FanTheorem finite-bar row, StreamName finite windows, RegSeqRat readback, RealUp seal, explicit refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2527

A Concrete Naming Certificate for $\mathsf{CauchyCutBridge}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCutBridge}^{\uparrow}$ packet records the finite handoff by which a regular Cauchy real name supplies located-cut observations without identifying either presentation by quotient equality. It sits between the regular-rational route of chapter 301, the located-cut surface of chapter 404, the dyadic arithmetic ledger of chapter 291, the finite stream windows of chapter 126, and the terminal real-name seal of chapter 38. The packet is a bridge for displayed approximation data, not a Dedekind-completeness theorem or an equality principle for completed reals.

Definition 2527.1 (Cauchy-cut bridge carrier)**.** *A* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, L, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the regular Cauchy rational source.*
2. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *row recording the lower and upper located observations obtained from the same regular name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare regular readbacks with located lower–upper brackets.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule listing the finite precision rows at which the regular name and located cut are jointly inspected.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after Q, L, D, W have been displayed.*
6. *H records componentwise* hsame *transport for the regular readback, located-cut row, dyadic ledger, stream windows, Real seal, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay of the bridge route*

$$Q, W, D \longmapsto L \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCutBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient reals, Dedekind completeness, host equality of real names, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCutBridgeUp.TasteGate.CauchyCutBridgeTasteGate single carrier alignment`

Theorem 2527.2 (Cauchy-cut bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier* $B = (Q, L, D, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed regular readback, located-cut row, dyadic tolerance ledger, stream-window schedule, Real seal, componentwise transport, continuation replay, package provenance, and local naming row; regular-to-located exposure through* Q, W, D, L*; Real-seal exposure through* E*; and refusal of quotient reals, Dedekind-completeness claims, host equality, trichotomy, choice, and ambient completeness.*

Proof. Project the carrier of Definition 3483.1. The visible rows are exactly $Q, L, D, W, E, H, C, P, N$, so carrier admission exposes the regular readback, located-cut observation, dyadic ledger, stream schedule, terminal seal, and structural rows.

The bridge exposure is supplied by the replay row C. It first reads the regular name through Q and the finite windows W, then checks the matching dyadic tolerances in D, and only then exposes the located-cut row L.

The Real seal and structural rows are downstream of that route. Transport through H, provenance through P, and local naming through N record the same finite bridge, while no refused object appears as a carrier coordinate. □

Theorem 2527.3 (Cauchy-cut bridge regular-located handoff)**.** *Let* $B = (Q, L, D, W, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier. Every located-cut observation exported by* B *factors through the finite handoff*

$$Q, W, D \longmapsto L \longmapsto E, H, C, P, N.$$

Thus the bridge reads the regular Cauchy name through $\mathsf{RegSeqRat}^{\uparrow}$*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the* $\mathsf{LocatedCut}^{\uparrow}$ *data or the terminal* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. It does not export quotient reals, Dedekind completeness, host equality, trichotomy, countable choice, or ambient completeness.*

Proof. Begin with a consumer of the located-cut row. The carrier definition gives L only as a displayed row of B, and the continuation row records that it is reached from Q, W, D.

Use the sibling routes cited in the chapter introduction. $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance comparison that turns the readback into lower–upper located observations.

Finally read the terminal seal row E and the structural rows H, C, P, N. They transport, replay, source, and name the same handoff; none of them stores quotient equality, a Dedekind-completeness witness, a host equality test, trichotomy, choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCutBridge}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{LocatedCut}^{\uparrow}$ observation row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3483.1, Definition 3483.2, and Definition 3483.3 seed the finite bridge from regular Cauchy readbacks to located-cut observations over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, Dedekind-completeness claims, host equality of real names, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, regular-to-located exposure, tolerance exactness, Real-seal nonescape, and ledger-refusal proofs for the same finite packet.

Chapter 2528

A Concrete Naming Certificate for $\mathsf{BishopCompletionTheorem}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionTheorem}^{\uparrow}$ packet records the theorem-level completion handoff as finite BEDC certificate data. It sits after the explicit modulus surface of chapter 1438, the finite-window naming surface of chapter 126, the regular rational readback of chapter 301, the terminal Real surface of chapter 38, and the universal regular-Cauchy handoff of chapter 1425. The packet names the constructive theorem route without quotient limits, selected representatives, countable choice, or ambient completeness.

Definition 2528.1 (Bishop completion theorem carrier)**.** *A* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row recording the explicit tolerance-to-tail schedule.*
2. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window source row inspected by the theorem route.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the displayed windows and modulus schedule.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after* M, S, R *have been displayed.*
5. U *is the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *handoff row for extension and uniqueness consumers.*
6. H *records componentwise* hsame *transport for the modulus, stream, readback, Real seal, universal handoff, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a theorem consumer through* M, S, R, E, U.
8. P *is the* Pkg *provenance over* $\mathsf{BishopCompletionTheorem}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$ *row.*

The carrier exports only this theorem route. It contains no quotient limit, selected representative, host convergence predicate, host real equality test, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BishopCompletionTheoremUp.BishopCompletionTheoremTasteGate single carrier alignment`

Theorem 2528.2 (Bishop completion theorem NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionTheore}$ *carrier* $B = (M, S, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission, explicit Cauchy-modulus exposure, finite StreamName source exposure, RegSeqRat readback, Real seal handoff, regular-Cauchy universal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every theorem consumer must factor through*

$$M \longmapsto S \longmapsto R \longmapsto E \longmapsto U \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3484.1. The accepted packet displays exactly $M, S, R, E, U, H, C, P, N$.

Read the analytic route in order. The explicit modulus row M licenses the finite StreamName windows S, the windows feed the regular rational readback R, and only then can the Real seal E be consumed.

The universal handoff U is downstream of the same displayed route. The structural rows H, C, P, N transport, replay, source, and name the route, while the refused quotient, selected, host, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 2528.3 (Bishop completion theorem regular Cauchy existence route)**.** *Every accepted* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *carrier* $B = (M, S, R, E, U, H, C, P, N)$ *exposes the regular Cauchy existence route as finite certificate composition:*

$$\mathsf{CauchyConvergenceModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$$

The route witnesses existence only as a displayed regular-Cauchy handoff through $M, S, R, E, U, H, C, P, N$*; it does not create a quotient limit, select a representative, invoke countable choice, compare host real values, or assert ambient completeness.*

Proof. Start with the modulus-controlled source. The row M supplies the displayed tolerance schedule, and S supplies the finite windows inspected at those requests.

Compose the readback and seal rows. The row R repacks the finite windows as regular rational data, and E records the terminal Real-facing seal reached by that regular route.

Finally consume the universal handoff U. By projection and finite route composition, any theorem consumer reaches the regular-Cauchy existence surface only after M, S, R, E and before H, C, P, N close transport, replay, provenance, and naming. The excluded quotient, selected, choice, host, and ambient-completeness objects are absent from the packet. □

Falsifiable prediction: A claimed $\mathsf{BishopCompletionTheorem}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ without the displayed CauchyConvergenceModulus, StreamName, and RegSeqRat rows refutes the carrier surface.

Independence witness: cauchyconvergencemodulus, streamname, regseqrat, real, regularcauchycompletionuniversal: no listed sibling is bijective to $\mathsf{BishopCompletionTheorem}^{\uparrow}$ because the theorem packet binds the modulus-controlled source, finite windows, regular readback, Real seal, universal handoff, transport, replay, provenance, and local NameCert rows in one theorem-level carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$.

Constructive story: A BishopCompletionTheorem$^{\uparrow}$ packet is generated from a CauchyConvergenceModulus row, StreamName finite windows, RegSeqRat readback, RealUp seal, RegularCauchyCompletionUniversal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopCompletionTheorem}\uparrow}$, seed)

Scope closed: Definition 3484.1, Definition 3484.2, and Definition 3484.3 seed the finite theorem-level route from modulus-controlled Cauchy data through regular rational readback into Real and regular-completion handoff rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopCompletionTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion, selected representatives, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, modulus exposure, finite StreamName windows, RegSeqRat readback, Real seal handoff, regular-Cauchy universal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionTheoremUp.

Chapter 2529

A Concrete Naming Certificate for $\mathsf{BaireUltrametric}^{\uparrow}$

Following Brouwer's spread semantics and the prefix-space practice of Troelstra–van Dalen and Bishop–Bridges, the $\mathsf{BaireUltrametric}^{\uparrow}$ packet names the common-prefix ultrametric surface on Baire streams. It sits between the finite prefix carrier of chapter 334, the prefix metric route of chapter 1602, the general ultrametric discipline of chapter 1535, the finite-window readback of chapter 126, and the completion-facing packet of chapter 1884. The chapter names the finite common-prefix distance interface; it is not host stream equality, quotient equality of streams, countable choice, a selected branch principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The finite prefix route consumes the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126 and the strong-triangle comparison surface of chapter 1535; the present packet only binds those sibling rows to the Baire prefix classifier.

Definition 2529.1 (Baire ultrametric carrier)**.** *A* $\mathsf{BaireUltrametric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, M, V, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row, naming the two inspected Baire prefixes and their common-prefix length.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *row, recording the dyadic distance value attached to the first displayed disagreement or to full agreement on the inspected window.*
3. *V is the* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison row, exposing the strong-triangle comparison only through finite common-prefix witnesses.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the stream coordinates read before the prefix distance is declared.*
5. *H records componentwise* hsame *transport for prefix rows, metric rows, ultrametric comparisons, finite windows, replay rows, provenance, and local naming.*
6. *C records displayed* Cont *replay of the same finite prefix windows and distance rows.*
7. *P is the* Pkg *provenance over* $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{UltrametricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, M, V, S, H, C, P, N. It has no coordinate for host stream equality, quotient stream equality, selected branches, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier`

Theorem 2529.2 (Baire ultrametric NameCert obligations)**.** *For every accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *carrier* $U = (B, M, V, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed Baire prefix row, BaireMetric distance row, UltrametricSpace comparison row, StreamName window row, transport, replay, provenance, and local naming;*
2. *common-prefix distance exactness, requiring each distance read to be the dyadic value named by the first displayed disagreement, or the zero-distance window when the inspected prefixes agree throughout the displayed window;*
3. *finite-cylinder classifier stability, requiring every cylinder comparison to transport only the displayed prefix rows through componentwise* hsame *and displayed* Cont *replay;*
4. *StreamName readback, requiring every coordinate read to pass through the finite window row S before the metric row M or ultrametric row V is consulted;*
5. *nonescape, excluding selected branches, quotient stream equality, countable choice, ambient Real completeness, host stream equality, Lean verification, and bridge checking.*

No BaireUltrametric obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 3485.1. The accepted packet displays exactly the Baire prefix row, the prefix metric row, the ultrametric comparison row, the finite StreamName window, and the structural rows.

Common-prefix distance exactness is read from B and M. The prefix row records the inspected coordinates and the first displayed disagreement if one exists; the metric row records the corresponding dyadic distance, with the zero-distance window available only for agreement across the displayed finite window.

Finite-cylinder classifier stability is supplied by H and C. Componentwise hsame transport identifies the displayed prefix rows, and displayed Cont replay rereads the same finite windows before any classifier output is compared.

StreamName readback is the row S. Since every coordinate request must pass through S, the ultrametric row V can only witness the strong-triangle comparison through finite common-prefix evidence already displayed by the packet.

The listed rows contain no selector for an infinite branch, no quotient equality of streams, no countable-choice operation, no host stream equality predicate, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. Thus the five rows exhaust the local naming surface. □

Theorem 2529.3 (Baire ultrametric prefix stability)**.** *Let* $U = (B, M, V, S, H, C, P, N)$ *and* $U' = (B', M', V', S', H', C', P', N')$ *be accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *carriers. If the displayed* $\mathsf{BaireSpace}^{\uparrow}$ *prefix rows,* $\mathsf{BaireMetric}^{\uparrow}$ *common-prefix distance rows,* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison rows, and* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows transport componentwise under* hsame *and replay through displayed* Cont *routes, then U and U' determine the same finite-cylinder classifier value and the same common-prefix ultrametric route.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier prefix stability`

Proof. Project both carriers through Definition 3485.1. The only analytic rows available for comparison are the prefix rows B, B', the metric rows M, M', the ultrametric rows V, V', and the window rows S, S'.

Use the assumed componentwise hsame transports to identify the displayed prefix and distance rows. The displayed Cont replays then force the same finite StreamName coordinates to be reread before either carrier can expose a classifier value.

Apply Definition 3485.2. Common-prefix distance exactness fixes the dyadic value from the first displayed disagreement or from the displayed agreement window, and finite-cylinder classifier stability keeps the classifier on those same rows.

The structural rows P, N and P', N' provide provenance and local naming only for the already transported route. Since no excluded branch, quotient, choice, host equality, or ambient Real-completeness coordinate is present, both carriers determine the same finite common-prefix ultrametric surface. □

Theorem 2529.4 (Baire ultrametric finite-cylinder separation)**.** *Let $U = (B, M, V, S, H, C, P, N)$ and $U' = (B', M', V', S', H', C', P', N')$ be accepted* BaireUltrametric$^{\uparrow}$ *packets. Suppose their displayed* BaireSpace$^{\uparrow}$ *prefix rows expose a first-disagreement window before length k, and suppose the* StreamName$^{\uparrow}$ *rows S, S' list every coordinate inspected up to that window. Then the* BaireMetric$^{\uparrow}$ *rows M, M' display a positive dyadic common-prefix distance for the two inspected names, and the packets determine disjoint finite cylinders before any* CompleteBaireSpace$^{\uparrow}$ *consumer reads a limit row.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier finite cylinder separation`

Proof. Project both carriers through Definition 3485.1. The only rows capable of naming the disagreement are the BaireSpace$^{\uparrow}$ prefix rows B, B', the BaireMetric$^{\uparrow}$ distance rows M, M', and the finite StreamName$^{\uparrow}$ windows S, S'.

Use the displayed first-disagreement window in B, B'. It identifies a least inspected coordinate at which the two finite prefixes differ, while S, S' certify that the coordinate is part of the finite window actually read by the packets.

The positive distance is then read from M, M'. By Definition 3485.2, common-prefix distance exactness assigns the dyadic value named by that first displayed disagreement, so the two packets cannot be assigned the zero-distance window for the inspected prefixes.

The finite cylinders are the two prefix cylinders ending at the displayed disagreement window. Any stream extending the first prefix must keep the first value at the disagreement coordinate, and any stream extending the second prefix must keep the second value there; the two finite-cylinder rows are therefore disjoint without consulting a completed limit.

Finally, V, V' are used only for the strong-triangle boundary. They preserve the finite-cylinder separation when a later consumer compares a third prefix, but they do not introduce an infinite branch, selected limit, quotient equality, or host stream equality. Thus a CompleteBaireSpace$^{\uparrow}$ consumer receives the positive dyadic separation before any completion-facing limit row is read. □

Theorem 2529.5 (Baire ultrametric cylinder refinement)**.** *Let $U = (B, M, V, S, H, C, P, N)$ and $U' = (B', M', V', S', H', C', P', N')$ be accepted* BaireUltrametric$^{\uparrow}$ *packets. If U displays a finite prefix cylinder and U' extends the same prefix through a longer* StreamName$^{\uparrow}$ *window without changing the already inspected coordinates, then the later cylinder refines the earlier one and the common-prefix ultrametric distance cannot increase on the displayed window. The refinement route*

uses only finite prefix rows, StreamName readback, componentwise hsame *comparison, and the finite strong-triangle boundary; it exports no selected branch, quotient stream equality, countable choice, host stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier cylinder refinement`

Proof. Project both packets through Definition 3485.1. The rows B, B' name the displayed finite prefixes, S, S' name the inspected StreamName windows, M, M' name the common-prefix distances, and V, V' supply only the finite ultrametric comparison boundary.

Use the StreamName readback and common-prefix exactness clauses of Definition 3485.2. The longer window in S' rereads every coordinate already present in S, and the hypothesis says that those displayed coordinates are unchanged.

Compare the two finite prefix rows componentwise through H and H'. The shared coordinates are identified by hsame, while the additional coordinates in the later packet can only shrink the cylinder by imposing more displayed prefix data; they cannot move a stream outside the earlier cylinder.

The distance comparison is then read from M, M' with V, V' used only as a finite strong-triangle boundary. Extending a common displayed prefix cannot create an earlier disagreement on the already inspected window, so the common-prefix ultrametric distance on that window cannot increase.

The remaining rows C, C', P, P', N, N' replay, source, and name this finite route. Since no row contains a selected infinite branch, quotient stream equality, countable choice, host stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the refinement stays inside finite prefix evidence before any $\mathsf{CompleteBaireSpace}^{\uparrow}$ completion row is read. □

Theorem 2529.6 (Baire ultrametric finite-cylinder completion handoff)**.** *Every* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *consumer reading a limit boundary from an accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *packet* $U = (B, M, V, S, H, C, P, N)$ *must first expose the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row* B*, the* $\mathsf{BaireMetric}^{\uparrow}$ *distance row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, the* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison row* V*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*. The completion handoff is therefore a finite-cylinder route, not a selected branch, quotient stream equality, countable choice, host stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier through Definition 3485.1. The only rows visible before a completion-facing boundary are the prefix row, distance row, ultrametric comparison row, finite StreamName window, and the structural support rows.

Apply Definition 3485.2. Carrier admission and StreamName readback order the read through B, M, V, S, H, C, P, N, while common-prefix distance exactness keeps the distance row attached to displayed finite prefix witnesses.

Use finite-cylinder separation and refinement, Definition 3485.4 and Definition 3485.5. Separation gives the positive finite-cylinder handoff before a completion row can read a limit boundary, and refinement keeps later windows inside the same displayed prefix route.

Finally repeat the nonescape clauses from the local NameCert surface. Since the packet has no coordinate for a selected branch, quotient stream equality, countable choice, host stream equality, or ambient completeness, a $\mathsf{CompleteBaireSpace}^{\uparrow}$ consumer receives only the finite-cylinder route before the limit boundary. □

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier finite cylinder completion handoff`

Falsifiable prediction: Every $\mathsf{BaireUltrametric}^{\uparrow}$ read factors through a displayed $\mathsf{BaireSpace}^{\uparrow}$ prefix row, a $\mathsf{BaireMetric}^{\uparrow}$ common-prefix distance row, a finite $\mathsf{StreamName}^{\uparrow}$ window row, an $\mathsf{UltrametricSpace}^{\uparrow}$ strong-triangle comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bairespace, bairemetric, ultrametricspace, streamname, complete-bairespace: $\mathsf{BaireUltrametric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite Baire prefixes, common-prefix dyadic distances, ultrametric comparisons, finite stream windows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3485.1, Definition 3485.2, Definition 3485.3, and Definition 3485.4 seed the common-prefix ultrametric route from Baire prefixes through prefix distances, finite StreamName windows, finite-cylinder separation, ultrametric comparison rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host stream equality, quotient stream equality, selected branches, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-prefix distance exactness, finite-cylinder classifier stability, finite-cylinder separation, StreamName readback, prefix stability, and nonescape for BEDC.Derived.BaireUltrametric

Constructive story: A $\mathsf{BaireUltrametric}^{\uparrow}$ packet is generated from finite Baire prefix rows, BaireMetric common-prefix distance rows, UltrametricSpace comparison rows, StreamName finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2530

A Concrete Naming Certificate for $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$

The $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ packet realizes the observer-expansion reading of section 9463.4 as finite BEDC certificate data. It sits after the disk-coordinate surface of chapter 377, the complex phase readout of chapter 39, the finite flow route of chapter 195, the Poincare disk transport packet of chapter 3455, and the metric-read packet of chapter 3474. The chapter names expansion as a records-side ledger reading, not as global time, physical cosmology, or object-level boundary fixedness.

Definition 2530.1 (Hyperbolic expansion ledger carrier)**.** *A* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (F, D, G, M, B, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the focused-ledger row recording the visible record family whose growth is being read.*
2. *D is the disk-readout row over* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*, and* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *coordinates.*
3. *G is the* Cont*-growth row comparing local continuation extension with successive ledger readouts.*
4. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row consumed only after F, D, G have been displayed.*
5. *B is the boundary-refusal row excluding global clock, physical cosmology, completed boundary object, and object-level boundary fixedness.*
6. *A is the provenance row tying F, D, G, M, B to the hyperbolic observer-expansion reading.*
7. *H records componentwise* hsame *transport for focused-ledger, disk-readout, growth, metric-read, boundary-refusal, provenance, replay, and local naming rows.*
8. *C records displayed* Cont *replay through the focused ledger, disk readout, growth route, metric read, and refusal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicExpansionLedger}^{\uparrow}}$ *row.*

The carrier exports expansion only as a finite focused-ledger and metric-read route. It contains no global clock, no empirical cosmology, no completed boundary model, and no object-level fixed boundary.

Theorem 2530.2 (Hyperbolic expansion ledger NameCert obligations)**.** *For every accepted* HyperbolicExpansionLe *carrier* $X = (F, D, G, M, B, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicExpansionLedger}^{\uparrow}}$ *surface consists exactly of focused-ledger admission, disk-readout admission,* Cont*-growth routing, metric-read admission, boundary-refusal exactness, provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted expansion read factors through*

$$F \longmapsto D \longmapsto G \longmapsto M \longmapsto B \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Lean4 ✓ `BEDC.Derived.HyperbolicExpansionLedgerUp.HyperbolicExpansionLedgerTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3486.1. The focused ledger F, disk readout D, growth row G, metric read M, and boundary-refusal row B are displayed before provenance and naming are consumed.

Read the route in the stated order. The disk metric is available only after the focused ledger and Cont-growth route have been displayed, so expansion is a records-side reading rather than a global clock.

The rows A, H, C, P, N then record provenance, transport, replay, package sourcing, and local naming. Since no global clock, physical cosmology, completed boundary model, or object-level fixed boundary has a carrier coordinate, none can be exported by the NameCert surface. □

Theorem 2530.3 (Hyperbolic expansion ledger Cont-growth route)**.** *Every accepted expansion read of a* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *packet factors through the focused ledger and* Cont*-growth route before disk metric consumption:*

$$F, D, G \longmapsto M.$$

The hyperbolic metric row M may compare displayed disk readouts only after the local continuation-growth row G has been opened, and the boundary-refusal row B prevents the comparison from becoming a global clock, physical cosmology, completed boundary, or object-level fixedness claim.

Proof. Start with a metric consumer of M. Carrier projection shows that M is downstream of the focused ledger F, disk readout D, and continuation-growth row G.

Use the displayed Cont replay row C. It can replay metric consumption only along the focused-ledger and growth route, so the disk metric cannot be read as an independent time parameter.

Finally apply boundary-refusal exactness through B. The packet refuses global clock, physical cosmology, completed boundary, and object-level fixedness readings, leaving only the finite records-side expansion route. □

Falsifiable prediction: Every accepted expansion read factors through the focused ledger and Cont-growth route before disk metric consumption; a consumer that reads the hyperbolic metric without F, D, G refutes the $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ carrier.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, unitdisk, dynsystem, complex: no listed sibling is bijective to $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ because the packet jointly binds focused-ledger growth, disk readout, Cont-growth routing, metric consumption, boundary refusal, provenance, transport, replay, package sourcing, and local naming.

The Lean follow-up for this packet must provide explicit Nontrivial HyperbolicExpansionLedger$^{\uparrow}$ and FieldFaithful HyperbolicExpansionLedger$^{\uparrow}$ TasteGate instances. Those instances must keep $F, D, G, M, B, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$.

Constructive story: A HyperbolicExpansionLedger$^{\uparrow}$ packet is generated from a focused-ledger row, disk-readout row, Cont-growth route, HyperbolicMetric read, boundary-refusal row, provenance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$, seed)

Scope closed: Definition 3486.1, Definition 3486.2, and Definition 3486.3 seed the finite observer-expansion route through focused-ledger, disk-readout, Cont-growth, metric-read, boundary-refusal, provenance, transport, replay, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HyperbolicExpansionLedgerUp.HyperbolicExpansionLedgerTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close global clock semantics, physical cosmology, completed boundary geometry, object-level boundary fixedness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, focused-ledger rows, disk-readout rows, Cont-growth routing, metric-read rows, boundary-refusal exactness, TasteGate nontriviality, field faithfulness, provenance, and local naming for BEDC.Derived.HyperbolicExpansionLedgerUp.

Chapter 2531

A Concrete Naming Certificate for $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$

The $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ packet realizes the reversible phase-transport conservation row of Definition 9463.4 as finite NameCert data. It sits between the complex-history surface of chapter 39, the phase-transport carrier of chapter 3455, the metric readback of chapter 3474, and the expansion ledger of chapter 3486. The packet records conservation only as auditable preservation of source, transport, classifier, and GAP provenance rows; it does not assert a completed symplectic manifold, a global Hamiltonian flow, a host Liouville measure, or an ambient unitary group.

Falsifiable prediction: Any accepted $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ route preserves source, transport, classifier, and GAP provenance rows before a conserved phase read is exported.

Independence witness: complex, unitarygroup, dynsystem, poincare-disk-phase-transport, hyperbolic-metric: the ledger-conservation packet is not bijective to these siblings because it records reversible preservation of finite source, route, classifier, and GAP provenance rows rather than a complex carrier, ambient unitary group, dynamical system, disk-transport chart, or metric tensor alone.

Definition 2531.1 (Liouville ledger-conservation carrier)**.** *A* $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, K, G, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source phase-surface row, carrying the finite visible histories whose distinctions must remain readable.*
2. *T is the reversible transport row, read through the local phase-transport surface and recording the forward and return continuations as displayed data.*
3. *K is the classifier-preservation row, requiring the transported surface to preserve the classifier readout attached to S.*
4. *G is the* GAP *provenance row, recording that residues, exits, and unresolved boundary data remain ledger-visible after transport.*
5. *Q is the conserved phase-read row. It is exposed only after S, T, K, G have all been displayed.*
6. *H records componentwise* hsame *transport for source, reversible transport, classifier preservation,* GAP *provenance, conserved read, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a conservation request through S, T, K, G, Q, H, P, N.*

8. *P is the* Pkg *provenance over* LiouvilleLedgerConservation$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicExpansionLedger$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose conserved phase read Q is downstream of source exposure, reversible transport, classifier preservation, and GAP *provenance. It has no coordinate for host measure theory, a symplectic manifold, an ambient Hilbert space, a hidden unitary operator, or a global dynamical-system solution.*

Theorem 2531.2 (Liouville ledger-conservation NameCert obligations)**.** *For every accepted* LiouvilleLedgerConserva *carrier*

$$L = (S, T, K, G, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the source phase surface, reversible transport, classifier preservation,* GAP *provenance, conserved phase read, transport, replay, package provenance, and local naming rows;*

2. *source preservation, requiring every conserved read to keep the displayed source distinctions of S visible after transport;*

3. *route reversibility, requiring T to expose both the forward and return transport rows before conservation is read;*

4. *classifier preservation, requiring K to certify the transported classifier readout before Q is exported;*

5. GAP *ledger preservation, requiring G to keep residue and boundary provenance visible rather than erasing it into the conserved read;*

6. *structural non-escape, requiring H, C, P, N to preserve and name the same packet without adding an ambient unitary group, host Hamiltonian flow, completed phase volume, or hidden global state.*

Future Lean `TasteGate.lean` *will provide explicit* `Nontrivial LiouvilleLedgerConservationUp` *and* `FieldFaithful LiouvilleLedgerConservationUp` *instances so the accepted conservation packet remains distinct from refused routes and faithful reads preserve all displayed rows.*

Proof. Carrier exposure is projection from Definition 3487.1. The displayed packet contains exactly the source, reversible transport, classifier, GAP provenance, conserved read, and structural rows.

Source preservation and route reversibility are enforced before any conserved read is available. The row Q is downstream of S and T, so a consumer cannot export conservation after forgetting either the source distinctions or the return route.

Classifier and GAP preservation are separate rows. The classifier row K records preservation of the readout, while G keeps residues and boundary provenance visible as provenance rather than absorbing them into the phase read.

The structural rows then confine the packet. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming preserve only S, T, K, G, Q and add no coordinate for the excluded host structures. □

Lean4 ✓ `BEDC.Derived.LiouvilleLedgerConservationUp.LiouvilleLedgerConservation namecert obligations`

Theorem 2531.3 (Liouville ledger-conservation export discipline)**.** *Every accepted* $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ *route exports a conserved phase read only along*

$$S, T, K, G \longmapsto Q \longmapsto H, C, P, N.$$

Consequently a downstream consumer receives source exposure, reversible transport, classifier preservation, GAP *provenance, the conserved read, transport, replay, package provenance, and local naming, and receives no host Liouville measure, ambient unitary operator, global dynamical system, completed symplectic phase space, or hidden state.*

Proof. Project an accepted carrier $L = (S, T, K, G, Q, H, C, P, N)$. The only conservation-facing row is Q, and it is displayed after the source, transport, classifier, and GAP rows.

Replay through C preserves that order. A downstream conservation request must pass through S, T, K, G before it can read Q, so the conserved phase read remains tied to visible source distinctions, route data, classifier preservation, and boundary provenance.

The structural rows authenticate only that finite route. Since the carrier has no host-measure, unitary-operator, global-dynamics, symplectic-manifold, or hidden-state coordinate, none of those structures can be exported from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3487.1, Definition 3487.2, and Definition 3487.3 seed the finite reversible ledger-conservation surface over source, transport, classifier, GAP provenance, conserved-read, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Liouville measure theory, ambient unitary dynamics, completed symplectic geometry, global Hamiltonian flow, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source preservation, reversible transport, classifier preservation, GAP ledger preservation, export discipline, Nontrivial LiouvilleLedgerConservationUp, and FieldFaithful LiouvilleLedgerConservationUp.

Constructive story: A $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ packet is generated from finite source phase-surface rows, reversible transport rows, classifier-preservation rows, GAP provenance rows, a conserved phase-read row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2532

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet records the located-real completion boundary as finite NameCert data. It consumes the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the located-real observation row of chapter 1933, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a human-origin completion route for displayed located Cauchy data; it is not a quotient-real construction, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2532.1 (Bishop located-real completion carrier)**.** *A* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the requested radii and comparison budgets.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family inspected at those dyadic tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the displayed finite windows.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locatedness row recording lower–upper separation evidence for the same readback windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R, L have been displayed.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, locatedness, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each completion-facing request through D, S, R, L, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected representatives, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedRealCompletionUp.BishopLocatedRealCompletionTasteGate single carrier alignment`

Theorem 2532.2 (Bishop located-real completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier* $K = (D, S, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, finite StreamName window exposure, RegSeqRat readback, LocatedReal locatedness exposure, Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every completion-facing consumer reads only the displayed route*

$$D \longmapsto S \longmapsto R \longmapsto L \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3488.1. Its analytic rows are the dyadic tolerance ledger, finite stream windows, regular rational readback, locatedness row, and Real seal.

Read the route in order. The tolerance row D fixes the finite request, S opens the corresponding window, R repacks that window as regular rational data, and L supplies the located lower–upper evidence attached to the same displayed readback.

The seal row E is downstream of those four rows. The structural rows H, C, P, N transport, replay, source, and name the route, while quotient streams, selected limits, host equality, choice, trichotomy, compactness, and ambient completeness have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.BishopLocatedRealCompletionUp.BishopLocatedRealCompletion namecert obligations`

Theorem 2532.3 (Bishop located-real completion tolerance-window exactness)**.** *For an accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier* $K = (D, S, R, L, E, H, C, P, N)$*, every located completion request fixes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before selecting the* $\mathsf{StreamName}^{\uparrow}$ *window* S*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and* $\mathsf{LocatedReal}^{\uparrow}$ *row* L *are then read over that same window before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed.*

Proof. Project the carrier of Definition 3488.1. The tolerance ledger D is the first analytic row of the packet, and the finite window row S is admitted only as the stream window inspected at those dyadic tolerances.

Apply the NameCert obligations of Definition 3488.2. They force the completion route to read D before S, then R, then L, and only then the terminal seal E.

The located Cauchy handoff of Definition 3488.4 keeps the regular readback and locatedness evidence on the same displayed window. Since quotient streams, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, no completion request can replace this tolerance-window route with a broader Real construction. □

Theorem 2532.4 (Bishop located-real completion located Cauchy handoff)**.** *Every* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *consumer that treats a displayed regular Cauchy name as a located real must pass through* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{LocatedReal}^{\uparrow}$ *row before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exposes located Cauchy evidence only as the finite packet* $D, S, R, L, E, H, C, P, N$*.*

Proof. Start with the NameCert obligations. They place the dyadic tolerance ledger before the window family and the regular rational readback.

The locatedness row is then forced to share the same finite windows. Since L is read after R, a consumer cannot substitute a host located real, a quotient stream class, or a selected representative for the displayed regular readback.

Finally the Real seal is reached through E and preserved by H, C, P, N. The handoff is therefore the finite BEDC route through chapter 291, chapter 126, chapter 301, chapter 1933, and chapter 38. □

Theorem 2532.5 (Located-real completion Cauchy representation)**.** *Every accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *packet exposes its regular Cauchy representation only through the shared rows* D, S, R, L *before the terminal* E *seal. Thus* $\mathsf{Real}^{\uparrow}$ *consumers can read DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, and LocatedReal locatedness evidence, but they cannot read quotient-stream representatives, selected limits, host equality, countable choice, trichotomy, or ambient Real completeness from the packet.*

Proof. Project the carrier of Definition 3488.1. The rows D, S, R, L are the only analytic coordinates available before the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Apply Definition 3488.2 and Definition 3488.3. These results force the representation route to pass through the displayed tolerance ledger, finite window, regular rational readback, and locatedness row on the same window.

Now apply Definition 3488.4 and Definition 3488.6. The handoff identifies the regular Cauchy representation consumed by located-real clients, and the seal nonescape theorem excludes quotient-stream representatives, selected limits, host equality, choice, trichotomy, and ambient completeness from the exported $\mathsf{Real}^{\uparrow}$ surface. □

Theorem 2532.6 (Bishop located-real completion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *packet factors through the displayed rows* $D, S, R, L, E, H, C, P, N$*. The seal exports no quotient-real equality, selected host limit, host real equality test, countable-choice selector, trichotomy split, open-cover compactness witness, or ambient completeness theorem.*

Proof. Read the carrier definition. The only route to E is through the tolerance ledger, finite windows, regular readback, and locatedness row.

Apply the located Cauchy handoff. It identifies D, S, R, L as the exact finite route consumed before the terminal seal can be inspected.

The structural rows do not add analytic content. They preserve and name the displayed route, while the refused quotient, selected, host, choice, order, compactness, and ambient-completeness objects are absent from the carrier and from the local NameCert provenance. □

Falsifiable prediction: A claimed $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the displayed DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, and LocatedReal locatedness row refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, locatedreal, real: no listed sibling supplies the whole $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet, because each sibling contributes only tolerance, window, readback, locatedness, or terminal seal data while this chapter binds all five rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, LocatedReal locatedness row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3488.1, Definition 3488.2, Definition 3488.4, and Definition 3488.6 seed the finite located-real completion route through dyadic tolerances, stream windows, regular rational readback, locatedness rows, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real construction, selected host limits, countable choice, trichotomy, open-cover compactness, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, regular readback, locatedness handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealCompletion

Chapter 2533

A Concrete Naming Certificate for $\mathsf{KirszbraunExtension}^{\uparrow}$

Following Kirszbraun and the Banach-space tradition around nonexpansive and Lipschitz extension theorems, the $\mathsf{KirszbraunExtension}^{\uparrow}$ packet records the finite certificate surface needed to read a Lipschitz map on a metric subspace as a convex inner-product-valued extension request. It sits beside the metric subspace packet of chapter 1519, the Lipschitz map packet of chapter 410, the reusable extension interface of chapter 3650, the convex target surface of chapter 208, the inner-product target source of chapter 86, and the metric source of chapter 57. The chapter names the finite Kirszbraun-facing route without claiming Hilbert-space completeness, a host extension operator, pointwise equality of functions, or a selected global formula.

Definition 2533.1 (Kirszbraun extension carrier)**.** *A* $\mathsf{KirszbraunExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, A, L, Y, V, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *row for the source observations.*
2. *A is the* $\mathsf{MetricSubspace}^{\uparrow}$ *row, including the membership, inclusion, and inherited-distance route from A into X.*
3. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row on A, carrying the finite graph window and displayed Lipschitz constant.*
4. *Y is the target* $\mathsf{InnerProduct}^{\uparrow}$ *row, read through its vector, scalar-form, norm-seed, and metric-seed observations.*
5. *V is the target* $\mathsf{ConvexSet}^{\uparrow}$ *row inside the inner-product carrier, recording the admitted convex codomain surface for extension values.*
6. *B is the Lipschitz-constant ledger, recording the rational bound, the source-distance reads through A, X, and the target-distance reads through Y, V.*
7. *F is the extension graph row over the ambient metric window X with values admitted by V.*
8. *R is the restriction and bound row, requiring source points admitted by A to replay the graph of L and requiring every displayed ambient pair of F to respect the same ledger B.*

9. *H records componentwise* hsame *transport for the ambient metric, subspace, source Lipschitz map, target inner-product, convex codomain, Lipschitz ledger, extension graph, restriction, replay, provenance, and local name rows.*
10. *C records displayed* Cont *replay from a Kirszbraun extension request through $X, A, L, Y, V, B, F, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{KirszbraunExtension}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers componentwise through these rows. It contains no coordinate for a host Hilbert space, a global extension operator, quotient equality of functions, countable choice, ambient compactness, or unbounded real completion.

Lean4 ✓ `BEDC.Derived.KirszbraunExtensionUp.KirszbraunExtensionTasteGate single carrier alignment`

Theorem 2533.2 (Kirszbraun extension NameCert obligations)**.** *For every accepted* $\mathsf{KirszbraunExtension}^{\uparrow}$ *carrier $K = (X, A, L, Y, V, B, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, metric-subspace inclusion through A, source Lipschitz-map admission through L, inner-product target admission through Y, convex codomain admission through V, Lipschitz-constant ledger readback through B, extension-graph admission through F, restriction and bound exactness through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The finite route is

$$X, A \longmapsto L \longmapsto Y, V \longmapsto B, F, R \longmapsto H, C, P, N.$$

No obligation is discharged by a host extension formula, a quotient of maps, or a hidden Hilbert-space theorem.

Proof. Carrier admission is projection from Definition 3490.1. The displayed rows are exactly $X, A, L, Y, V, B, F, R, H, C, P, N$.

The source side is read through X, A, L. The subspace row admits source points and inherited distances, and the Lipschitz-map row exposes the finite graph window and bound that the extension must preserve.

The target side is read through Y, V. The inner-product row supplies the target metric seed, while the convex-set row supplies the admitted value surface for extension graph reads.

The ledger and graph clauses are B, F, R. The ledger stores the single Lipschitz constant and both source and target distance reads, the graph row stores ambient observations, and the restriction row forces source points to replay the original Lipschitz map.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no host extension operator, quotient-function coordinate, choice row, compactness row, or completion theorem coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2533.3 (Kirszbraun extension non-escape)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, *or* $\mathsf{InnerProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{KirszbraunExtension}^{\uparrow}$ *packet factors through*

$$X, A, L, Y, V, B, F, R, H, C, P, N.$$

It may read the packet as a finite extension candidate whose graph restricts to the source Lipschitz map and whose target values remain in the displayed convex inner-product row, but it cannot export a host Kirszbraun theorem, a chosen extension operator, a quotient-map equality, countable choice, compactness, or ambient Hilbert completeness.

Proof. Start from the obligation surface of Definition 3490.2. Every accepted consumer route must read the ambient metric, subspace, source Lipschitz map, target inner-product row, convex codomain row, ledger, graph, restriction, transport, replay, provenance, and name rows.

A source-facing consumer reaches the extension only through A, L, R. Thus an admitted source point is compared with the original Lipschitz graph before any ambient graph read is treated as an extension value.

A target-facing consumer reaches values only through Y, V, B, F. The inner-product row supplies the metric seed, the convex row admits values, and the ledger checks the same Lipschitz constant against displayed distances.

The remaining rows H, C, P, N repack that route. They preserve and name the carrier but do not add a host theorem, operator selector, quotient equality, choice schedule, compactness witness, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3490.1, Definition 3490.2, and Definition 3490.3 seed the finite Kirszbraun extension carrier over metric subspace, Lipschitz map, convex target, inner-product target, extension graph, and Lipschitz-ledger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Kirszbraun theorem, global Hilbert completeness, quotient equality of functions, selected extension operators, countable choice, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subspace inclusion, source Lipschitz admission, convex inner-product target admission, Lipschitz-ledger readback, graph admission, restriction exactness, transport, replay, provenance, and non-escape for BEDC.Derived.KirszbraunExtensionUp.

Constructive story: A $\mathsf{KirszbraunExtension}^{\uparrow}$ packet is generated from an ambient Metric row, a MetricSubspace source row, a source LipschitzMap row, an InnerProduct target row, a ConvexSet target row, a Lipschitz-constant ledger, an extension graph, a restriction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2534

A Concrete Naming Certificate for HyperbolicGeodesicBoundary$^{\uparrow}$

The HyperbolicGeodesicBoundary$^{\uparrow}$ packet realizes the boundary-geodesic route of chapter 9463 as finite Poincare-disk, hyperbolic-metric, phase-transport, and provenance rows. It sits after the disk-coordinate surface of chapter 377, reversible transport in chapter 219, finite flow rows in chapter 195, the disk-phase carrier of chapter 3455, and the metric-read surface of chapter 3474. The packet names geodesic-boundary data without treating the boundary as a completed host circle, quotient ideal point, physical horizon, or observer substance.

Definition 2534.1 (HyperbolicGeodesicBoundary carrier)**.** *A* HyperbolicGeodesicBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, T, B, J, F, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-coordinate row, including the finite radius and boundary-approach observations used by the packet.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *T is the* PoincareDiskPhaseTransport$^{\uparrow}$ *transport row, retaining reversible phase movement, finite flow data, and fiber provenance.*
4. *B is the boundary-geodesic incidence row, recording which finite disk observations are read as approaching the same geodesic far-end.*
5. *J is the classifier-stability row, preserving disk-coordinate, hyperbolic-metric, phase-transport, and boundary-geodesic classifiers under the displayed route.*
6. *F is the refusal row excluding quotient boundary equality, completed geodesic compactification, host horocycle selection, physical cosmology, and observer-substance readings.*
7. *A is the audit row tying D, M, T, B, J, F to the hyperbolic phase vision and to the Poincare-disk metric and transport packets.*
8. *H records componentwise* hsame *transport for disk-coordinate, metric-read, phase-transport, boundary-geodesic, classifier-stability, refusal, and audit rows.*
9. *C records displayed* Cont *replay through finite disk observations, metric reads, phase transport, boundary-geodesic checks, classifier stability, provenance, and refusal.*

10. *P is the* Pkg *provenance over* HyperbolicGeodesicBoundary$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskPhaseTra UnitaryGroup$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicGeodesicBoundary}^{\uparrow}}$ *row.*

The carrier exports boundary-geodesic data only through these rows. It has no coordinate for a completed hyperbolic plane, quotient ideal boundary, selected horocycle, global physical evolution law, or object-level observer substance.

Theorem 2534.2 (HyperbolicGeodesicBoundary NameCert obligations)**.** *For every accepted* HyperbolicGeodesicBo *carrier*

$$G = (D, M, T, B, J, F, A, H, C, P, N),$$

the local NameCert$_{\text{HyperbolicGeodesicBoundary}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric read admission, Poincare-disk phase-transport admission, boundary-geodesic incidence, classifier stability, refusal exactness, audit provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every boundary-geodesic consumer must factor through

$$D, M, T \longmapsto B, J, F, A \longmapsto H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicBoundaryUp.HyperbolicGeodesicBoundaryTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3491.1. The visible surface begins with disk-coordinate row D, metric-read row M, and phase-transport row T.

Read the incidence and stability rows next. The row B records boundary-geodesic incidence as finite far-end evidence, while J preserves the disk, metric, transport, and boundary classifiers across the displayed data.

The refusal and audit rows keep the packet aligned with the hyperbolic phase reading without exporting host geometry. Finally H, C, P, N transport, replay, source, and name only those displayed rows, so no quotient boundary, selected horocycle, physical horizon, or observer-substance reading can be extracted. □

Theorem 2534.3 (HyperbolicGeodesicBoundary phase handoff)**.** *Every phase-transport consumer of an accepted* HyperbolicGeodesicBoundary$^{\uparrow}$ *carrier receives the geodesic far-end only after the route*

UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, UnitaryGroup$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, Pkg

has been displayed. The consumer may use finite disk coordinates, hyperbolic metric reads, reversible phase transport, finite flow, boundary incidence, classifier stability, provenance, and local naming. It may not read quotient boundary equality, completed geodesic compactification, selected horocycles, physical cosmology, or observer substance.

Proof. Start from the NameCert obligation surface of Definition 3491.2. It places the disk-coordinate row, hyperbolic metric read, and phase-transport route before boundary incidence is exposed.

Use the phase-transport ledger of Definition 3455.2. The reversible transport and finite flow rows keep the source ledger visible while the boundary-geodesic row is read.

The hyperbolic metric obligations of Definition 3474.2 supply the metric-read discipline. Together with H, C, P, N, these rows give a finite phase handoff and no stronger host-side boundary object. □

Falsifiable prediction: A $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ consumer that obtains quotient boundary equality, selected horocycle data, or a completed geodesic compactification without first reading the route $D, M, T, B, J, F, A, H, C, P, N$ refutes the carrier surface.

Independence witness: hyperbolic_metric, unitdisk, unitarygroup, dynsystem, poincare_disk_phase_transpor $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk-coordinate observations, hyperbolic metric reads, phase transport, boundary-geodesic incidence, classifier stability, refusal, audit provenance, transport, replay, package sourcing, and local NameCert rows.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial\ HyperbolicGeodesicBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicGeodesicBoundary}^{\uparrow}$ TasteGate instances. Those instances must keep $D, M, T, B, J, F, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ packet is generated from UnitDisk disk coordinates, HyperbolicMetric metric reads, PoincareDiskPhaseTransport rows, boundary-geodesic incidence, classifier stability, refusal exactness, audit provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3491.1, Definition 3491.2, and Definition 3491.3 seed the finite boundary-geodesic route over disk-coordinate, hyperbolic-metric, phase-transport, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HyperbolicGeodesicBoundaryUp.HyperbolicGeodesicBoundaryTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient boundary equality, completed geodesic compactification, selected horocycles, physical cosmology, observer-substance readings, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-geodesic incidence, metric-read stability, phase-transport handoff, refusal exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicGeodesicBoundaryUp.

Chapter 2535

A Concrete Naming Certificate for $\mathsf{UniformClosure}^{\uparrow}$

Following Bishop-style metric and uniform-space practice, the $\mathsf{UniformClosure}^{\uparrow}$ packet records the closure interface whose tests are uniform-neighbourhood rows rather than host open covers. It sits between the metric closure surface of chapter 3342, the completion handoff of chapter 1506, the separation boundary of chapter 1411, the terminal real seal of chapter 38, and the regular-rational readback route of chapter 301. The packet names closure by finite entourage, window, and seal data; it is not open-cover compactness, quotient closure, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2535.1 (UniformClosure carrier)**.** *A* $\mathsf{UniformClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricClosure}^{\uparrow}$ *row naming the metric closure surface whose point, source, and candidate-limit data are being read.*
2. *Q is the uniform-neighbourhood ledger, recording the finite entourage requests admitted for the same carrier.*
3. *S is the source subset or located-source row being closed under those uniform requests.*
4. *L is the candidate closure point, read only through the displayed uniform and metric rows.*
5. *E is the entourage-to-metric comparison row that routes uniform requests into the metric tests already visible in M.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface used to inspect the approximating observations and the candidate point.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows, tolerances, and comparison requests.*
8. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, Q, S, L, E, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the metric-closure, uniform-ledger, source, candidate, comparison, window, readback, and seal rows.*
10. *C records displayed* Cont *replay from an entourage request through the metric-closure test, finite windows, regular-rational readback, and real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{UniformClosure}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose uniform requests, metric closure route, candidate point, finite windows, regular readbacks, real seal, transport, replay, provenance, and local naming are displayed. It has no coordinate for open-cover compactness, quotient closure, selected limits, a quotient completion object, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformClosureUp.UniformClosureTasteGate single carrier alignment`

Theorem 2535.2 (UniformClosure NameCert obligations)**.** *For every accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier* $U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{MetricClosure}^{\uparrow}$ *row, uniform-neighbourhood ledger, source row, candidate closure point, entourage-to-metric comparison row, finite window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every public uniform-closure read must factor through

$$M, Q, S, L, E, W, R, Z, H, C, P, N$$

before the candidate closure point can be consumed.

Proof. Project the carrier of Definition 3492.1. The only admissible source data are the metric-closure row M, uniform ledger Q, source row S, and candidate closure point L.

The comparison row E then binds each uniform request to the metric tests already exposed by M. The finite window row W, regular-rational readback R, and real seal Z give the displayed numerical route for those tests.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores open-cover compactness, quotient closure, selected limits, a quotient completion object, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2535.3 (UniformClosure completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier reads the closure candidate only through the finite route*

$$\mathsf{MetricClosure}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The consumer receives the displayed metric-closure row, uniform-neighbourhood ledger, entourage-to-metric comparison, regular-rational readback, real seal, separation-facing identity boundary, transport, replay, provenance, and local naming. It receives no selected completion limit, quotient closure object, open-cover compactness principle, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the public route of Definition 3492.2. It places the uniform ledger and entourage-to-metric comparison before any completion-facing consumer can read the candidate point.

Use the metric-closure handoff of Definition 3342.3. The same finite windows and regular-rational readback rows feed the completion-facing side, so the consumer cannot detach the candidate from the displayed metric closure surface.

The separated-metric boundary of Definition 1411.48 supplies the identity discipline for zero-distance reads. Transport, replay, provenance, and naming through H, C, P, N preserve that route without adding selected limits, quotient closure, compactness, or ambient real completeness. □

Theorem 2535.4 (UniformClosure separated stability)**.** *Let $U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$ be an accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier. If a downstream separated-metric consumer compares two transported closure candidates from U, the comparison must pass through the displayed metric-closure row M, uniform ledger Q, comparison row E, finite windows W, regular-rational readback R, real seal Z, and the structural rows H, C, P, N before the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier can be read.*

Thus separated stability is a finite transport-and-seal obligation over the displayed rows, not a quotient closure equality or host identity theorem.

Proof. Project the two transported candidates through the componentwise hsame rows in H. The transport preserves the metric-closure row, uniform ledger, comparison row, windows, readback, and real seal as displayed rows of the same packet.

Replay the comparison through C. The route reaches the $\mathsf{Real}^{\uparrow}$ seal only after M, Q, E, W, R are fixed, and then uses the separated-metric identity boundary rather than host equality.

The provenance and local naming rows P, N expose this route as the public certificate. Since no quotient closure object, selected limit, open-cover compactness row, or ambient real-completeness coordinate occurs in U, the separated-stability conclusion remains confined to the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformClosure}^{\uparrow}$ packet is generated from a MetricClosureUp row, a uniform-neighbourhood ledger, source and candidate rows, an entourage-to-metric comparison row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$, seed)

Scope closed: Definition 3492.1, Definition 3492.2, Definition 3492.3, and Definition 3492.4 seed the finite uniform-closure interface for completion and separated-metric consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient closure, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-neighbourhood ledger stability, entourage-to-metric comparison, RegSeqRat readback, Real seal nonescape, completion handoff, separated stability, transport, replay, provenance, and local naming for BEDC.Derived.UniformClosureUp.

Chapter 2536

A Concrete Naming Certificate for CauchyFilterLocatedRealExtraction$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the CauchyFilterLocatedRealExtraction$^{\uparrow}$ packet records the finite route by which a Cauchy-filter surface is read as located-real window data. It consumes the Cauchy filter interface of chapter 3416, the located-real comparison surface of chapter 3395, the stream-name observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite extraction route only; it is not ambient Real completeness, a quotient filter construction, a selected limit, or a countable-choice principle.

Definition 2536.1 (Cauchy-filter located-real extraction carrier)**.** *A* CauchyFilterLocatedRealExtraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, L, S, Q, D, R, T, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyFilter$^{\uparrow}$ *base row, exposing the filter-facing Cauchy surface and its finite request windows.*
2. *L is the* LocatedReal$^{\uparrow}$ *window row, recording the lower and upper located cuts read from the accepted filter request.*
3. *S is the* StreamName$^{\uparrow}$ *observation row, listing the finite observation windows through which the filter data are sampled.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, translating the sampled filter windows into regular rational approximants.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, recording the dyadic mesh, modulus, and separation checks used by the located window.*
6. *R is the* Real$^{\uparrow}$ *seal row for the extracted endpoint and readback data.*
7. *T records componentwise* hsame *transport for $F, L, S, Q, D, R, C, P, N$.*
8. *C records displayed* Cont *replay from a filter request through the observation, rational readback, dyadic tolerance, located-window, and Real-seal rows.*
9. *P is the* Pkg *provenance over* CauchyFilterLocatedRealExtraction$^{\uparrow}$*,* CauchyFilter$^{\uparrow}$*,* LocatedReal$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, L, S, Q, D, R, T, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient filters, selected limits, countable choice, host real equality, or an unrestricted completion principle.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLocatedRealExtractionUp.CauchyFilterLocatedRealExtractionTasteGate`
`single carrier alignment`

Theorem 2536.2 (Cauchy-filter located-real extraction)**.** *Every accepted* $\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}$ *packet extracts a located-real finite-window route from the Cauchy-filter surface:*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported route is the displayed finite packet $E = (F, L, S, Q, D, R, T, C, P, N)$ and no stronger completeness, quotient-filter, selected-limit, or countable-choice object is produced.

Proof. Project the accepted carrier. The Cauchy-filter surface is F, the finite observation windows are S, the regular rational readback is Q, the dyadic tolerance ledger is D, the located-real window is L, and the terminal Real-facing seal is R.

Replay through C keeps these rows in extraction order. A filter request is sampled only through the displayed stream-name windows, those observations are read back as regular rational data, and the dyadic ledger checks the tolerance needed before the located window is exposed.

Transport and provenance do not add new mathematical objects. The rows T, P, N move and name only the displayed packet, so the extraction cannot escape to quotient filters, selected limits, countable choice, ambient Real completeness, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}$ packet is generated from a CauchyFilter base row, LocatedReal finite window, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 3493.1 and Definition 3493.2 seed the finite extraction route from Cauchy-filter windows to located-real readback data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient filters, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observation-window readback, dyadic tolerance exactness, located-window extraction, Real-seal compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterLocatedRealExtractionUp.

Chapter 2537

A Concrete Naming Certificate for $\mathsf{SignedDyadicExpansion}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers presented by finite approximation data, the $\mathsf{SignedDyadicExpansion}^{\uparrow}$ packet records the finite signed dyadic prefix route before a prefix may be consumed as a regular rational name or sealed as a $\mathsf{Real}^{\uparrow}$ object. It sits between the binary expansion surface of chapter 418, the signed-digit stream interface of chapter 316, the signed dyadic normal form row of $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite signed dyadic readback data; it is not a quotient stream, a selected infinite expansion, a uniqueness theorem for dual binary tails, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2537.1 (Signed dyadic expansion carrier)**.** *A* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, W, E, T, F, A, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the sign row, recording whether the displayed prefix is read with positive, negative, or zero orientation before any numeric readback is opened.*
2. *W is the finite dyadic digit-prefix window, exposing the inspected binary cells and their request index as finite data.*
3. *E is the exponent and tail-radius ledger, recording the dyadic scale of the prefix and the displayed dyadic radius bounding the unseen tail.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the tail-radius ledger and by every readback request.*
5. *F is the* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *row, normalizing the sign, exponent, and visible digit window without selecting an infinite expansion.*
6. *A is the* $\mathsf{StreamName}^{\uparrow}$ *window row that identifies which finite prefix cells are inspected at each requested tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, W, E, T, F, A.*
8. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the signed prefix, dyadic tolerance, normal-form, stream-window, and regular-rational rows have been displayed.*

9. *H records componentwise* hsame *transport for the sign, digit window, exponent ledger, tail-radius ledger, normal form, stream window, regular readback, real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$S, W, E, T, F, A \longmapsto R \longmapsto L.$$

11. *P is the* Pkg *provenance over* $\mathsf{SignedDyadicExpansion}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, E, T, F, A, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite signed dyadic expansion, uniqueness at dual binary tails, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SignedDyadicExpansionUp.SignedDyadicExpansionTasteGate single carrier alignment`

Theorem 2537.2 (Signed dyadic expansion NameCert obligations)**.** *For every accepted* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier*

$$X = (S, W, E, T, F, A, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the sign row, finite dyadic digit-prefix window, exponent and tail-radius ledger, dyadic tolerance row, signed dyadic normal form, stream window, regular rational handoff, real seal, transport, replay, provenance, and local name;*

2. *sign-prefix coherence, requiring S and W to expose the same finite signed prefix before any rational readback is admitted;*

3. *exponent and tail-radius visibility, requiring E and T to bound the unseen dyadic tail by a displayed dyadic tolerance;*

4. *normal-form locality, requiring F to normalize only the displayed sign, exponent, and digit-window rows;*

5. *window exactness, requiring A to select exactly the finite prefix cells consumed by the requested tolerance;*

6. *handoff order, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume S, W, E, T, F, A before the terminal* $\mathsf{Real}^{\uparrow}$ *seal L becomes available;*

7. *componentwise transport, displayed replay, package provenance, and local naming for the same rows through H, C, P, N;*

8. *refusal of quotient stream equality, selected infinite expansions, dual-tail uniqueness, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No signed dyadic expansion obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SignedDyadicExpansionUp.SignedDyadicExpansionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 3494.1. The accepted packet displays exactly the sign row, finite digit window, exponent and tail-radius ledger, dyadic tolerance row, signed normal form, stream window, regular readback, real seal, and structural rows.

The prefix information is read first. The sign row S and digit window W determine the visible signed dyadic prefix, while E and T expose the dyadic scale and the tail-radius budget that bounds all unseen cells relevant to the request.

The normal-form and window rows keep the readback local. The row F normalizes only the displayed finite prefix data, and A selects the finite cells consumed at the requested tolerance.

The handoff is ordered through R and L. A regular rational name may be read only after S, W, E, T, F, A are displayed, and the $\mathsf{Real}^{\uparrow}$ seal is terminal over that regular rational row.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores quotient stream equality, an infinite expansion selector, a dual-tail uniqueness decision, trichotomy, countable choice, or ambient completeness, none of those objects becomes a local obligation. □

Theorem 2537.3 (Signed dyadic expansion Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier factors through the finite route*

$$S, W, E, T, F, A \longmapsto R \longmapsto L \longmapsto H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the sign row, finite dyadic prefix window, exponent and tail-radius ledger,* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row before the regular rational handoff and terminal real seal are available. The route exports no quotient stream, selected infinite expansion, dual-tail uniqueness theorem, trichotomy, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a consumer request for the terminal real-facing row. The accepted carrier exposes L only after the signed prefix and tolerance route has been displayed.

The request first opens S, W, E, T, F, A. These rows determine the visible sign, dyadic digit prefix, exponent, tail radius, normal form, and inspected window as finite BEDC data.

The regular rational handoff R is then read from that same finite route. Because the tail-radius bound is already displayed in E and T, the handoff does not require a quotient stream or an infinite representative.

The seal L and structural rows H, C, P, N package the readback for downstream consumers. The excluded coordinates are absent from the carrier, so every exported real read remains a finite signed dyadic expansion certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3494.1, Definition 3494.2, and Definition 3494.3 seed the finite signed dyadic prefix route from sign and digit-window rows through DyadicRatCore tolerances, SignedDyadicNormalForm rows, StreamName windows, RegSeqRat handoff, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite expansions, dual-tail uniqueness, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, sign-prefix coherence, exponent and tail-radius visibility, normal-form locality, window exactness, handoff order, transport, replay, provenance, and local naming for BEDC.Derived.SignedDyadicExpansionUp.

Constructive story: A $\mathsf{SignedDyadicExpansion}^{\uparrow}$ packet is generated from a sign row, finite dyadic digit-prefix window, exponent and tail-radius ledger, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{FrechetSpace}^{\uparrow}$ hub routes complete metrizable locally convex structure through chapter 1506, chapter 3386, and chapter 205.

Chapter 2538

A Concrete Naming Certificate for $\mathsf{FrechetSpace}^{\uparrow}$

Following Frechet and Bourbaki on complete metrizable locally convex spaces, the $\mathsf{FrechetSpace}^{\uparrow}$ packet records the finite BEDC surface where a topological vector space, a locally convex neighbourhood ledger, a displayed compatible metric, and a completion-facing row are read as one carrier. It sits beside the metric-completion surface of chapter 1506, the locally convex source of chapter 3386, and the topological-vector-space source of chapter 205. The chapter names only the carrier rows needed for a Frechet-space read; it does not import a host complete locally convex space, a selected metric, quotient completion equality, countable choice, or an infinite seminorm enumeration.

Every complete-metrizable $\mathsf{FrechetSpace}^{\uparrow}$ read consumes the $\mathsf{TopVecSpace}^{\uparrow}$ operation-continuity surface of chapter 205, the $\mathsf{LocallyConvexSpace}^{\uparrow}$ seminorm and neighbourhood ledger of chapter 3386, and the $\mathsf{MetricCompletion}^{\uparrow}$ completion handoff of chapter 1506 through the displayed rows $V, L, M, C, Q, H, R, P, N$ before a functional-analysis consumer can replay completeness or local convexity.

Definition 2538.1 (Frechet space carrier)**.** *A* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, C, Q, H, R, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{TopVecSpace}^{\uparrow}$ *source row, exposing the vector carrier, topology, addition-continuity ledger, and scalar-continuity ledger.*
2. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *row, recording the finite seminorm and convex balanced neighbourhood evidence consumed by local convexity reads.*
3. *M is the compatible* $\mathsf{Met}^{\uparrow}$ *row, recording the finite distance witnesses and the topology-compatibility comparison used by metrizable consumers.*
4. *C is the completeness ledger, read through Cauchy windows, Cauchy-filter or Cauchy-net evidence, and displayed convergence rows without selecting a completed point.*
5. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, used only to expose the completion-facing boundary of the compatible metric.*
6. *H records componentwise* hsame *transport for V, L, M, C, Q and for the replay, provenance, and local naming rows.*

7. *R records displayed* Cont *replay through the topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion handoff, transport, provenance, and local naming rows.*

8. *P is the* Pkg *provenance over* $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, L, M, C, Q, H, R, P, N$. It has no coordinate for a host Frechet space, an external complete metric, a selected translation-invariant metric, quotient completion equality, countable choice, or an infinite family of seminorms.

Theorem 2538.2 (Frechet space NameCert obligations)**.** *For every accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier $F = (V, L, M, C, Q, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ *surface consists exactly of topological-vector source exposure through V; locally convex source exposure through L; compatible metric exposure through M; completeness evidence through C; metric-completion handoff through Q; componentwise transport through H; displayed replay through R; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 3495.1. Its mathematical rows are the topological-vector source V, the locally convex row L, the compatible metric row M, the completeness ledger C, and the metric-completion handoff Q.

The topological and locally convex rows are consumed before the metric row. Thus a Frechet-space consumer first reads vector operations, topology, continuity, seminorm probes, and convex balanced neighbourhood evidence through the already named $\mathsf{TopVecSpace}^{\uparrow}$ and $\mathsf{LocallyConvexSpace}^{\uparrow}$ surfaces.

The compatible metric and completeness rows then give the metrizable complete handoff. The row M supplies only displayed distance and topology-compatibility evidence, while C records Cauchy and convergence checks without adding a selected completion point.

Finally Q, H, R, P, N transport, replay, source, and name the same finite route. Since the carrier contains no host Frechet object, selected metric, quotient-completion equality, countable-choice schedule, or infinite seminorm enumeration, the listed rows exhaust the local NameCert surface. □

Theorem 2538.3 (Frechet space metrizable complete handoff)**.** *Let $F = (V, L, M, C, Q, H, R, P, N)$ be an accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier. Every downstream functional-analysis, completion, or locally convex consumer that asks for metrizability and completeness must factor through*

$$V, L, M, C, Q \longmapsto H, R, P, N.$$

The handoff exposes the topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion boundary, transport, replay, provenance, and local naming, and it exports no host complete locally convex space, selected compatible metric, quotient completion, countable-choice row, or infinite seminorm-family object.

Proof. Start with a downstream request for a Frechet-space read. The vector and topology data are not independent host structure; they are obtained from the $\mathsf{TopVecSpace}^{\uparrow}$ row V.

The locally convex evidence is read next through L. This forces finite seminorm probes and convex balanced neighbourhood rows to remain visible before any metric or completion claim is consumed.

The metric and completeness portion then passes through M, C, Q. The compatible metric row M supplies the displayed distance and topology comparison, C supplies the finite completeness ledger, and Q exposes only the $\mathsf{MetricCompletion}^{\uparrow}$ handoff boundary.

The structural rows H, R, P, N preserve that same route under transport, replay, provenance, and local naming. Therefore a consumer receives a metrizable complete locally convex carrier only as the displayed finite packet above, not as a host Frechet space, selected metric, quotient completion, choice principle, or infinite seminorm family. □

Falsifiable prediction: Any accepted $\mathsf{FrechetSpace}^{\uparrow}$ consumer that cites complete metrizability without first reading $V, L, M, C, Q, H, R, P, N$ refutes the carrier surface.

Independence witness: `metriccompletion`, `locally_convex_space`, `topvecspace`: no listed sibling is bijective with $\mathsf{FrechetSpace}^{\uparrow}$, because this carrier alone binds topological-vector data, locally convex seminorm and neighbourhood rows, a compatible metric, completeness evidence, metric-completion handoff, transport, replay, provenance, and local NameCert rows into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetSpace}^{\uparrow}$ packet is generated from a topological-vector source, a locally convex source, a compatible metric row, a completeness ledger, a metric-completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3495.1, Definition 3495.2, and Definition 3495.3 seed the finite Frechet-space carrier over $V, L, M, C, Q, H, R, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, barrelledness, Baire category, nuclearity, selected compatible metrics, quotient completions, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion handoff, transport, replay, provenance, and local naming rows for BEDC.Derived.FrechetSpaceUp.

After the complete metric row of Definition 3495.1 has been displayed, any $\mathsf{FrechetSpace}^{\uparrow}$ consumer that needs a Baire-category conclusion must leave the Frechet carrier through the theorem route of chapter 3570 and the inspected complete-metric surface of chapter 3535. The Frechet packet supplies local convexity, topological-vector continuity, compatible metric data, and completion-facing rows; the dense-open theorem route is a separate sibling surface.

Theorem 2538.4 (Frechet Baire dependency route)**.** *Every accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *read that asks for Baire-category evidence factors first through the complete metric row* C *and the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff* Q *of Definition 3495.1, and then through the* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *and* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *chapters cited above. The route adds no Hahn–Banach separation, selected generic point, countable-choice schedule, host intersection object, or ambient complete locally convex space to the Frechet carrier.*

Proof. Project the Frechet carrier as $F = (V, L, M, C, Q, H, R, P, N)$. The rows V, L, M, C, Q expose topological-vector structure, local convexity, compatible metric data, completeness evidence, and the metric-completion boundary before any theorem consumer reads the packet.

A Baire-category request is not one of those Frechet rows. Route it through chapter 3535 for the inspected complete-metric Baire surface and through chapter 3570 for the theorem-level dense-open package.

The remaining rows H, R, P, N only transport, replay, source, and name the displayed route. Since no row stores Hahn–Banach separation, a selected generic point, a countable-choice schedule,

a host intersection object, or an ambient complete locally convex space, none can be exported by the dependency route. □

Theorem 2538.5 (Frechet Baire-category completeness route)**.** *Every* $\mathsf{FrechetSpace}^{\uparrow}$ *consumer that needs dense-open completeness must expose the full carrier*

$$F = (V, L, M, C, Q, H, R, P, N)$$

and then pass through chapter 3570 and chapter 3535. The Frechet chapter contributes the displayed vector, local-convex, compatible-metric, complete-metric, completion-handoff, transport, replay, provenance, and local naming rows; the dense-open conclusion remains a theorem-level sibling route rather than a hidden field of F*.*

Proof. Project the accepted carrier as $F = (V, L, M, C, Q, H, R, P, N)$. The rows V, L, M, C, Q expose the topological-vector, local-convex, compatible-metric, complete-metric, and completion-handoff data needed before a dense-open consumer can even form its request.

Now apply the theorem route. The complete-metric surface is supplied by chapter 3535, while the dense-open theorem statement is supplied by chapter 3570. These two sibling chapters are therefore the only exits from the Frechet carrier toward Baire-category completeness.

The remaining rows H, R, P, N preserve and name the route. Since they contain only componentwise transport, replay, provenance, and the local $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ row, they cannot introduce a selected generic point, a countable-choice schedule, or an ambient complete locally convex space outside the displayed dependency route. □

Every $\mathsf{FrechetSpace}^{\uparrow}$ consumer reads the locally convex linear-topology surface of chapter 3386 before it treats the compatible metric as complete. The route then passes through the metric-completion handoff of chapter 1506, so Baire-category and metric-completion readers receive the displayed locally convex, complete-metric, transport, replay, provenance, and local naming rows rather than a host Frechet space or an unbounded seminorm schedule. Every $\mathsf{FrechetSpace}^{\uparrow}$ read first consumes the topological vector-space carrier of chapter 205 and the locally convex carrier of chapter 3386. It then reads the compatible metric row M, the completeness ledger C, and the $\mathsf{MetricCompletion}^{\uparrow}$ handoff row Q supplied by the metric-completion route of chapter 1506. The handoff is only the displayed carrier route needed to replay a complete locally convex packet through the local NameCert surface; it exports no selected compatible metric, quotient completion, countable-choice schedule, or infinite seminorm family.

Chapter 2539

A Concrete Naming Certificate for $\mathsf{FinitePartitionMesh}^{\uparrow}$

Following Bishop and Bridges on constructive partitions of real intervals, the $\mathsf{FinitePartitionMesh}^{\uparrow}$ packet records the finite dyadic mesh surface used before interval estimates may pass to Darboux or regular-completion consumers. It sits between the real endpoint seal of chapter 38, the dyadic tolerance rows of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the closed bounded interval Darboux partition route of chapter 3429, and the finite Darboux-sum boundary of chapter 1880. The packet names only displayed endpoint rows, dyadic breakpoints, mesh adjacency, refinement witnesses, and downstream handoff rows; it is not a selected tagged partition, an open-cover compactness theorem, a quotient equality of functions, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 2539.1 (Finite partition mesh carrier)**.** *A* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, Q, M, R, B, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *closed-interval endpoint row, exposing the lower endpoint, upper endpoint, endpoint order, and terminal real seal used by the mesh.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *breakpoint ledger naming the dyadic cut points and the dyadic mesh bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family in which endpoint and breakpoint reads are inspected.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed endpoints and dyadic breakpoints.*
5. *M is the finite mesh row listing the cells, endpoint adjacency, left-to-right order, and boundary inclusion witnesses.*
6. *R is the displayed common-refinement row comparing M with any accepted finite dyadic submesh over the same interval.*
7. *B is the* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *handoff row reached only after I, D, W, Q, M, R have been displayed.*
8. *S is the* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer row for finite lower and upper sum reads over the cells of M or its displayed refinement.*

9. *H records componentwise* hsame *transport of interval endpoints, dyadic breakpoints, windows, readbacks, mesh cells, refinement witnesses, Darboux handoff, sum rows, provenance, and local naming.*

10. *C records displayed* Cont *replay from endpoint rows to dyadic breakpoints, from breakpoints to mesh cells, and from mesh refinement to Darboux partition and sum handoffs.*

11. *P is the* Pkg *provenance over* $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite partition, a selected tag family, quotient function equality, measure completion, open-cover compactness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the displayed endpoint seal.*

Lean4 ✓ `BEDC.Derived.FinitePartitionMeshUp.FinitePartitionMeshTasteGate single carrier alignment decode encode`

Theorem 2539.2 (Finite partition mesh refinement obligations)**.** *For the carrier of Definition 3496.1, the local* $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, D, W, Q, M, R, B, S, H,$ interval endpoint exposure through I; dyadic breakpoint and mesh-bound exposure through D; finite observation windows through W; regular rational readback through Q; mesh-cell exposure through M; common-refinement evidence through R; closed bounded interval Darboux partition handoff through B; finite Darboux-sum handoff through S; componentwise transport through H; displayed continuation replay through C; provenance through P; and local naming through N.*

Every accepted refinement request must therefore route a $\mathsf{Real}^{\uparrow}$ *interval consumer, a* ClosedBoundedIntervalDarbo *consumer, or a* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer through the finite path*

$$I \longmapsto D \longmapsto W \longmapsto Q \longmapsto M \longmapsto R \longmapsto B \longmapsto S \longmapsto (H, C, P, N)$$

before any lower-sum, upper-sum, or interval estimate can be read. The route exports no selected tags, quotient stream, host partition predicate, countable-choice schedule, open-cover compactness row, measure-completion object, or ambient real-completeness theorem.

Proof. Project the displayed packet. The interval row I, dyadic ledger D, finite windows W, regular readback Q, and mesh row M are all finite certificate rows, so the carrier exposes a bounded list of cells before a refinement witness can be inspected.

Read the refinement row next. Since R compares mesh cells only through the already displayed endpoint, breakpoint, window, and readback rows, every common refinement remains a finite dyadic submesh over the same interval boundary.

Finally inspect the Darboux handoff rows. The rows B and S are downstream of the finite mesh and its refinement witness, while H, C, P, N only transport, replay, source, and name that route. The excluded tag, quotient, choice, compactness, measure, and ambient-completeness objects have no carrier coordinate, so the listed rows exhaust the refinement obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePartitionMesh}^{\uparrow}$ packet is generated from RealUp interval endpoint rows, DyadicRatCoreUp breakpoint rows, StreamNameUp finite windows, RegSeqRatUp readbacks, finite mesh cells, a displayed common-refinement row, ClosedBoundedIntervalDarbouxPartitionUp and DarbouxSumUp handoff rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$, seed)

Scope closed: Definition 3496.1 and Definition 3496.2 seed the finite dyadic partition-mesh route over $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, and $\mathsf{DarbouxSum}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected tags, quotient stream equality, host partition predicates, countable choice, open-cover compactness, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, breakpoint exactness, finite mesh refinement, Darboux partition handoff, Darboux-sum handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.FinitePartitionMeshU

Chapter 2540

A Concrete Naming Certificate for $\mathsf{ChoiceWindowSequence}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on choice sequences, the $\mathsf{ChoiceWindowSequence}^{\uparrow}$ packet isolates the finite inspected windows by which a choice sequence becomes available to an L10 $\mathsf{StreamName}^{\uparrow}$ consumer. It refines the finite-prefix surface of chapter 1482, keeps the lawless-sequence boundary of chapter 1495 visible, and reads Baire-style prefix data from chapter 334. The packet uses $\mathsf{Nat}^{\uparrow}$ positions and answer rows, $\mathsf{Bool}^{\uparrow}$ presence rows, and the $\mathsf{StreamName}^{\uparrow}$ hand-off only as displayed finite data; it does not import countable choice, a completed host sequence, quotient stream equality, or a global law selecting all future answers.

Definition 2540.1 (ChoiceWindowSequence finite-window carrier)**.** *A* $\mathsf{ChoiceWindowSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, L, B, S, I, A, O, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix packet whose currently available answers may be inspected;*
2. *L is the sibling* $\mathsf{LawlessSequence}^{\uparrow}$ *boundary row, recording that the packet reads only displayed answers and no governing law for future positions;*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix context in which the window is interpreted;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite positions requested by the consumer;*
5. *I is the* $\mathsf{Nat}^{\uparrow}$ *index ledger listing the positions of the inspected window;*
6. *A is the* $\mathsf{Nat}^{\uparrow}$ *answer ledger attached to the positions in I;*
7. *O is the* $\mathsf{Bool}^{\uparrow}$ *presence classifier for each requested position, distinguishing displayed answers from absent future data;*
8. *H records componentwise* hsame *transport for Q, L, B, S, I, A, O and the local naming row;*
9. *C records displayed* Cont *replay of the same finite-window read;*
10. *P is the* Pkg *provenance over* $\mathsf{ChoiceWindowSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LawlessSequence}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^\uparrow}$ *row.*
The carrier accepts only the finite window displayed by S, I, A, O. It exports no completed function $\mathsf{Nat}^\uparrow \to \mathsf{Nat}^\uparrow$*, no countable-choice selector, no quotient infinite stream, no hidden tail oracle, and no law determining positions outside the displayed window.*

Lean4 ✓ `BEDC.Derived.ChoiceWindowSequenceUp.ChoiceWindowSequenceTasteGate single carrier alignment`

Theorem 2540.2 (ChoiceWindowSequence carrier admission obligations)**.** *For every accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *carrier $W = (Q, L, B, S, I, A, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^\uparrow}$ *surface consists exactly of carrier admission for the source* $\mathsf{ChoiceSequence}^\uparrow$ *row, the lawless-sequence boundary row, the* $\mathsf{BaireSpace}^\uparrow$ *prefix context, the finite* $\mathsf{StreamName}^\uparrow$ *schedule, the* $\mathsf{Nat}^\uparrow$ *index and answer ledgers, the* $\mathsf{Bool}^\uparrow$ *presence classifier, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Project the carrier of Definition 3497.1. The visible coordinates are precisely $Q, L, B, S, I, A, O, H, C, P, N$, so carrier admission reads no object outside the displayed finite packet.

The source rows Q, L, B identify the traditional choice-sequence, lawless-sequence, and Baire-prefix context. The inspection rows S, I, A, O then name the finite schedule, the requested positions, the displayed answers, and the presence decisions.

The structural rows H, C, P, N transport, replay, source, and name the same finite window. Since the carrier has no coordinate for a total host sequence, countable-choice selector, quotient stream equality, hidden tail oracle, or global law, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2540.3 (ChoiceWindowSequence window classifier stability)**.** *Let $W = (Q, L, B, S, I, A, O, H, C, P, N)$ and $W' = (Q', L', B', S', I', A', O', H', C', P', N')$ be accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *carriers. If the displayed* $\mathsf{StreamName}^\uparrow$ *schedules,* $\mathsf{Nat}^\uparrow$ *index ledgers,* $\mathsf{Nat}^\uparrow$ *answer ledgers, and* $\mathsf{Bool}^\uparrow$ *presence classifiers agree componentwise under the available* hsame *rows, then every public finite-window read classifies W and W' the same way. The classifier is insensitive to tail values or laws outside the displayed window.*

Proof. Begin with the displayed schedules and ledgers. Agreement of S, I, A, O and S', I', A', O' gives the same requested positions, the same visible answer rows, and the same present-or-absent decisions for every finite request a consumer may replay.

The source context is read only through those windows. The $\mathsf{ChoiceSequence}^\uparrow$ and $\mathsf{BaireSpace}^\uparrow$ rows supply finite-prefix data, while the $\mathsf{LawlessSequence}^\uparrow$ row records that no law for undisplayed positions is part of the packet.

Finally transport the read through H, H' and replay it through C, C'. Since public reads factor through the displayed window and the carrier contains no tail oracle, completed function, quotient stream, or countable-choice selector, no classifier distinction can be obtained outside the agreed finite data. □

Theorem 2540.4 (ChoiceWindowSequence StreamName handoff)**.** *Every* $\mathsf{StreamName}^\uparrow$ *consumer of an accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *packet factors through the finite route*

$$\mathsf{ChoiceSequence}^\uparrow,\ \mathsf{LawlessSequence}^\uparrow,\ \mathsf{BaireSpace}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow,\ \mathsf{Nat}^\uparrow,\ \mathsf{Bool}^\uparrow \longmapsto \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Na}$$

The handoff supplies a finite inspected choice-sequence window to the L10 StreamName-facing interface and supplies no countable choice, total host sequence, quotient stream equality, selected future value, or law for undisplayed positions.

Proof. Read the source side first. The consumer receives a choice-sequence prefix row, a lawless-sequence boundary row, and a Baire-prefix context, all as finite displayed packets.

Next open the StreamName-facing window. The schedule row S selects the inspected positions, the $\mathsf{Nat}^{\uparrow}$ rows I, A list the requested indices and displayed answers, and the $\mathsf{Bool}^{\uparrow}$ row O records which requested positions are currently present.

Finally pass through H, C, P, N. Componentwise transport and continuation replay preserve the same finite window, while provenance and local naming attach the packet to the BEDC kernel rows. No excluded global choice object is introduced by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceWindowSequence}^{\uparrow}$ packet is generated from a ChoiceSequence prefix row, LawlessSequence boundary row, BaireSpace finite-prefix context, StreamName finite schedule, NatUp index and answer ledgers, BoolUp presence classifiers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3497.1, Definition 3497.2, Definition 3497.3, and Definition 3497.4 seed the finite inspected-window route for choice-sequence data consumed by StreamName-facing L10 packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, a total host sequence, quotient stream equality, selected future values, a global law for undisplayed positions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window classifier stability, StreamName handoff, transport, continuation, provenance, and non-escape proofs for BEDC.Derived.ChoiceWindowSequenceUp.

Chapter 2541

A Concrete Naming Certificate for $\mathsf{ProximinalSet}^{\uparrow}$

Following Bishop and Bridges on located sets in metric spaces, a proximinal set is read here as a located subset whose displayed distance data may be accompanied by finite distance-attainment witnesses. The $\mathsf{ProximinalSet}^{\uparrow}$ packet sits between the located-subset interface of chapter 1541, the located-metric handoff of chapter 1533, the real distance seal of chapter 38, and the Hilbert or convex projection specialization of chapter 2356. It records displayed distance-attainment witness rows and deliberately does not export a host nearest-point selector, quotient point equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2541.1 (Proximinal set carrier)**.** *A* $\mathsf{ProximinalSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, L, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the parent* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance witness, metric-law ledger, and finite observation histories.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row. It supplies the displayed subspace admission, located-distance ledger, rational comparison windows,* $\mathsf{Real}^{\uparrow}$ *distance seal, and* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff already exposed by the located-set packet.*
3. *D is the located-metric distance-read row. It records that the ambient point, admitted set row, and requested tolerance are read through the* $\mathsf{LocatedMetric}^{\uparrow}$ *and* $\mathsf{LocatedSet}^{\uparrow}$ *interfaces before any attainment evidence is inspected.*
4. *W is the distance-attainment witness row. For each displayed request it stores a finite admitted point row and a finite equality-of-distance certificate tying that point to the real distance name supplied by L and D.*
5. *E is the real attainment seal. It records the* $\mathsf{Real}^{\uparrow}$ *comparison between the displayed distance name and the displayed witness point distance, without adding trichotomy, quotient point equality, or ambient real completeness.*
6. *H stores componentwise* hsame *transport for the parent metric row, located-set row, located-metric distance read, attainment witness row, real seal, continuation replay, provenance, and local naming row.*

7. *C stores displayed* Cont *replay from ambient point observations into located-set distance data, from located-set distance data into D, from D into W, and from the witness row into E.*
8. *P is the* Pkg *provenance over* $\mathsf{ProximinalSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$ *row.*

The classifier compares packets only through $X, L, D, W, E, H, C, P, N$. It has no coordinate for a host nearest-point function, quotient equality of ambient points, trichotomy, open-cover compactness, a Hilbert projection theorem, or full ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ProximinalSetUp.ProximinalSetUp`

Theorem 2541.2 (Proximinal set NameCert obligations)**.** *For an accepted* $\mathsf{ProximinalSet}^{\uparrow}$ *carrier $Z = (X, L, D, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$ *obligations are exactly carrier admission, parent metric compatibility, located-set distance compatibility, located-metric read ordering, displayed distance-attainment witness stability, real attainment seal readback, and structural non-escape. These obligations are discharged by the finite route*

$$X \longmapsto L \longmapsto D \longmapsto W \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.ProximinalSetUp.ProximinalSetTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3498.1. The tuple lists the parent metric row, located-set row, located-metric distance read, attainment witness row, real seal, and structural rows.

Parent metric compatibility and located-set distance compatibility are read first. The consumer sees ambient points through X and sees set distance data through L, so the packet inherits the nearest-point refusal boundary of the located-set interface before any attainment row can be consumed.

The row D fixes the located-metric read order. It prevents an attainment witness from being read as a primitive selector by forcing the request through the located-set and located-metric distance route.

The witness row W then stores only displayed request-specific evidence: an admitted point row and a finite equality-of-distance certificate. The real seal E compares that displayed point distance with the real distance name already read from L and D.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for a host nearest-point function, quotient point equality, trichotomy, open-cover compactness, Hilbert projection, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 2541.3 (Proximinal set distance-witness route)**.** *Every accepted* $\mathsf{ProximinalSet}^{\uparrow}$ *packet exposes distance-attainment data only through the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{LocatedSet}^{\uparrow} \longmapsto \mathsf{LocatedMetric}^{\uparrow} \longmapsto W \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a $\mathsf{MetricProjection}^{\uparrow}$ *or compactness consumer may read a displayed witness row, but it must supply any host nearest-point selector, quotient point equality principle, trichotomy branch, open-cover compactness argument, or ambient real-completeness principle outside the packet.*

Proof. Start with Definition 3498.1. The only analytic rows available before the witness are the parent metric row X, the located-set row L, and the located-metric distance read D.

Apply Definition 1541.3 and Definition 1541.4. These force located-set distance data through parent metric observations, subspace admission, located-distance ledgers, rational windows, and a $\mathsf{Real}^{\uparrow}$ seal before a consumer can read the distance as real data.

The $\mathsf{ProximinalSet}^{\uparrow}$ carrier then adds W only as a displayed witness row for that already-read distance. It does not turn the row into a host function from ambient points to nearest points, because the witness is indexed by the finite request already present in the packet.

The terminal seal E records the equality-of-distance readback for the displayed witness. The structural rows H, C, P, N preserve the same route, so downstream projection or compactness consumers receive a finite distance-attainment witness boundary rather than quotient point equality, trichotomy, open-cover compactness, Hilbert projection, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{ProximinalSet}^{\uparrow}$ read from an accepted packet factors through the parent $\mathsf{Met}^{\uparrow}$ row, $\mathsf{LocatedSet}^{\uparrow}$ distance interface, $\mathsf{LocatedMetric}^{\uparrow}$ handoff, displayed distance-attainment witness row, $\mathsf{Real}^{\uparrow}$ attainment seal, componentwise transport, continuation replay, provenance, and local NameCert before any projection, compactness, or completion consumer reads the route.

Independence witness: locatedset, locatedmetric, metricprojection, real: $\mathsf{ProximinalSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries displayed located-set distance data and request-indexed distance-attainment witness rows while refusing host nearest-point selection, quotient point equality, trichotomy, open-cover compactness, and ambient real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$.

Constructive story: A $\mathsf{ProximinalSet}^{\uparrow}$ packet starts from a parent Metric row, reads a LocatedSet distance interface, routes the distance through a LocatedMetric handoff, records displayed request-indexed distance-attainment witnesses, seals the equality of distance in a RealUp row, and preserves the same route through componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$, seed)

Scope closed: Definition 3498.1, Definition 3498.2, and Definition 3498.3 seed the finite proximinal-set interface through located-set distance data, located-metric reads, displayed attainment witnesses, and real distance seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host nearest-point selection, quotient point equality, trichotomy, open-cover compactness, Hilbert projection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parent metric compatibility, located-set distance compatibility, located-metric read ordering, displayed distance-attainment witness stability, RealUp attainment seal readback, structural non-escape, Nontrivial ProximinalSetUp, and FieldFaithful ProximinalSetUp.

Chapter 2542

A Concrete Naming Certificate for $\mathsf{CompletionUnitComparison}^{\uparrow}$

Following Bourbaki and Bishop-style completion theory, the $\mathsf{CompletionUnitComparison}^{\uparrow}$ packet records the finite route by which the Cauchy-completion unit is compared with the naturality and triangle-identity certificates that consume it. It sits after the unit packet of chapter 1527, the triangle surface of chapter 2256, and the naturality comparison of chapter 2102. The chapter names a displayed comparison object for unit-facing completion reads; it does not import quotient completions, selected representatives, host equality of completed objects, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2542.1 (Completion unit comparison carrier)**.** *A* $\mathsf{CompletionUnitComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, N, T, S, R, H, C, P, L)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row exposing the source-to-completion unit used by the comparison.*
2. *N is the* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *row recording naturality compatibility for the same displayed unit.*
3. *T is the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *row recording triangle-identity compatibility for the same unit-facing route.*
4. *S is the source-and-target ledger over the finite stream, regular-rational, dyadic-tolerance, and terminal Real rows read by the unit comparison.*
5. *R is the refusal row excluding quotient completion equality, selected representatives, host equality of completed carriers, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
6. *H records componentwise* hsame *transport for unit, naturality, triangle, source-target, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the unit row through naturality compatibility and triangle compatibility before any consumer reads the comparison.*
8. *P is the* Pkg *provenance over* $\mathsf{CompletionUnitComparison}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}$ $\mathsf{CompletionUnitTriangle}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$ *row.*

The carrier compares only displayed completion-unit routes. It has no coordinate for host quotient completions, representative selection, equality of completed objects, choice principles, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionUnitComparisonUp.CompletionUnitComparisonTasteGate single carrier alignment`

Theorem 2542.2 (Completion unit comparison NameCert obligations)**.** *For every accepted* CompletionUnitCompar *carrier*

$$U = (Q, N, T, S, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *unit-comparison exposure, requiring every comparison consumer to cite the displayed* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row Q;*
2. *naturality compatibility, requiring the comparison to pass through the* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *row N for the same source and target data;*
3. *triangle compatibility, requiring the comparison to pass through the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *row T before any completion-facing return is exported;*
4. *transport and provenance stability, requiring H, C, P, L to transport, replay, source, and name only the rows Q, N, T, S, R;*
5. *refusal of host quotient completion, requiring the row R to exclude quotient equality, selected representatives, host equality of completed carriers, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No unit-comparison obligation is stored outside these rows.

Proof. Project the accepted carrier. The comparison-facing rows are $Q, N, T, S, R, H, C, P, L$.

The unit row Q supplies the only source-to-completion entry. Naturality compatibility must therefore read the same source and target data through N, and triangle compatibility must read the same unit-facing route through T.

The transport, replay, provenance, and local naming rows H, C, P, L preserve exactly this displayed route. Since the refusal row R contains the only host-side exclusions and no quotient-completion coordinate appears in the carrier, the five listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CompletionUnitComparison}^{\uparrow}$ consumer factors through displayed $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$, $\mathsf{CompletionUnitTriangle}^{\uparrow}$, source-target ledger, refusal, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: cauchycompletionunit, cauchy-completion-comparison-naturality, completion-unit-triangle: the unit packet records source admission without the comparison route, the naturality packet records functorial compatibility without the triangle return, and the triangle packet records return compatibility without the naturality comparison. None is bijective to $\mathsf{CompletionUnitComparison}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompletionUnitComparison}^{\uparrow}$ packet is generated from a CauchyCompletionUnit row, CauchyCompletionComparisonNaturality row, CompletionUnitTriangle row, source-target ledger, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3499.1 and Definition 3499.2 seed the finite completion-unit comparison route over unit, naturality, triangle, transport, provenance, and refusal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompletionUnitComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality of completed carriers, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked unit-comparison exposure, naturality compatibility, triangle compatibility, transport and provenance stability, and refusal exactness for BEDC.Derived.CompletionUnitComparisonUp.

Chapter 2543

A Concrete Naming Certificate for PoincareBoundaryFixedTransport$^{\uparrow}$

The PoincareBoundaryFixedTransport$^{\uparrow}$ packet realizes the boundary-fixed disk transport reading of Definition 9463.10 as finite disk-map, boundary, hyperbolic-metric, reversible-transport, provenance, refusal, and local naming rows. It sits after the disk-coordinate surface of chapter 377, the disk-phase transport packet of chapter 3455, the metric-read surface of chapter 3474, and the geodesic far-end boundary of chapter 3491. The chapter names a finite boundary-preserving transport object; it does not turn the Poincare disk into a host plane, identify ideal boundary points by quotient equality, choose a global geodesic representative, or assert physical observer motion.

Definition 2543.1 (Poincare boundary-fixed transport carrier)**.** *A* PoincareBoundaryFixedTransport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, M, T, P, R, H, C, A, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-map row, exposing the finite source and target disk-coordinate observations.*
2. *F is the boundary far-end row, recording the displayed* HyperbolicGeodesicBoundary$^{\uparrow}$ *far-end that must remain fixed by the transport.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same disk-map data.*
4. *T is the reversible* PoincareDiskPhaseTransport$^{\uparrow}$ *row carrying the phase-transport route between the displayed disk observations.*
5. *P is the provenance row tying D, F, M, T to the hyperbolic phase vision and to the sibling UnitDisk, HyperbolicMetric, PoincareDiskPhaseTransport, and HyperbolicGeodesicBoundary packets.*
6. *R is the refusal row excluding host disk automorphisms, quotient ideal-boundary equality, selected geodesic representatives, physical observer motion, and global cosmological dynamics.*
7. *H records componentwise* hsame *transport for disk-map, boundary far-end, metric-read, reversible-transport, provenance, refusal, replay, and local naming rows.*
8. *C records displayed* Cont *replay through disk-map admission, boundary far-end preservation, metric-read compatibility, reversible transport, provenance, and refusal.*

9. *A records the local transport audit row: every accepted replay must preserve the displayed boundary far-end while reading the same disk-map and metric rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$ *row.*

The carrier exports boundary-fixed disk transport only through $D, F, M, T, P, R, H, C, A, N$. It has no coordinate for a host-level disk automorphism, quotient boundary point, selected infinite geodesic, physical observer trajectory, or completed hyperbolic plane.

Lean4 (definition) `BEDC.Derived.PoincareBoundaryFixedTransportUp.PoincareBoundaryFixedTransportUp`
Lean4 ✓ `BEDC.Derived.PoincareBoundaryFixedTransportUp.PoincareBoundaryFixedTransportTasteGate`
`single carrier alignment`

Theorem 2543.2 (Poincare boundary-fixed transport NameCert obligations)**.** *For every accepted* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *carrier*

$$B = (D, F, M, T, P, R, H, C, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$ *surface consists exactly of disk-map admission, boundary far-end preservation, hyperbolic-metric read compatibility, reversible Poincare-disk transport, provenance, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay, local transport audit, and local naming.*

Every accepted transport preserving the boundary far-end must factor through

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, BHist, hsame, Cont, Pk

Proof. Project the accepted carrier. Disk-map admission is row D, the fixed far-end is row F, the metric read is row M, and the reversible disk transport is row T.

The provenance and refusal rows P, R bind these four rows to the sibling packets and exclude host-level automorphisms, quotient boundary equality, selected geodesics, physical observer motion, and global cosmology.

Transport through H and replay through C preserve the same displayed route. The audit row A checks that the far-end row F is the same before and after the transport, and N names the resulting packet. No further row remains, so the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ transport preserving the boundary far-end factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, provenance, refusal, hsame, Cont, and local NameCert rows.

Independence witness: unitdisk, poincare-disk-phase-transport, hyperbolic-metric, hyperbolic-geodesic-boundary: the UnitDisk packet records disk coordinates without reversible transport or fixed far-end data, the Poincare disk phase-transport packet records reversible transport without fixed-boundary audit, the HyperbolicMetric packet records metric reads without transport preservation, and the HyperbolicGeodesicBoundary packet records far-end incidence without a disk-map transport replay. None is bijective to $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ packet is generated from UnitDisk disk-map rows, HyperbolicGeodesicBoundary far-end rows, HyperbolicMetric metric reads, PoincareDiskPhaseTransport reversible transport, provenance, refusal exactness, componentwise hsame transport, displayed Cont replay, local transport audit, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$, seed)

Scope closed: Definition 3500.1 and Definition 3500.2 seed the finite boundary-fixed disk transport route over disk-map, boundary, metric, transport, provenance, refusal, replay, audit, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host disk automorphisms, quotient boundary equality, selected geodesic representatives, physical observer motion, global cosmological dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, boundary far-end preservation, metric-read compatibility, reversible transport, provenance, refusal exactness, transport, replay, audit, and local naming for BEDC.Derived.PoincareBoundaryFixedTransportUp.

Chapter 2544

A Concrete Naming Certificate for $\mathsf{MetricCompletionNaturality}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion and uniform continuity, the $\mathsf{MetricCompletionNaturality}^{\uparrow}$ packet records the finite L10 square by which a uniformly continuous source map is read by completion-facing consumers. It sits over the metric-completion surface of chapter 1506, the map carrier of chapter 130, the extension route of chapter 769, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the displayed naturality packet; it is not a quotient completion construction, not a selected representative principle, not host function equality, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2544.1 (Metric completion naturality carrier)**.** *A* $\mathsf{MetricCompletionNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L)$$

with the following displayed rows:

1. *M_X is the source* $\mathsf{MetricCompletion}^{\uparrow}$ *row, exposing the finite Cauchy-window requests inspected before the map is applied.*
2. *M_Y is the target* $\mathsf{MetricCompletion}^{\uparrow}$ *row, exposing the completion-facing consumer reached after the map and extension rows have been displayed.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the displayed source map and the finite continuity data it exposes to the packet.*
4. *E is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row that transports the same displayed map through the completion interface.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the source completion, target completion, and extension reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which numerical tolerance requests are answered.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream window and regular readback rows have been fixed.*
8. *H records componentwise* hsame *transport for the source completion, target completion, map, extension, stream-window, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the naturality route*

$$M_X \longmapsto F, E \longmapsto S, R, A \longmapsto M_Y.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionNaturality}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M_X, M_Y, F, E, S, R, A, H, C, P, L$. It has no coordinate for quotient completion equality, selected representatives, host function equality, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.MetricCompletionNaturalityUp.MetricCompletionNaturalityTasteGate single carrier alignment`

Theorem 2544.2 (Metric completion naturality NameCert obligations)**.** *For every accepted* MetricCompletionNatu *carrier*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source completion, target completion, continuous-map, uniform-extension, stream-window, regular-readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *source-before-extension exposure, requiring any completion-facing read to inspect M_X and F before the extension row E can be consumed;*
3. *target-consumer routing, requiring M_Y to be reached only after the displayed extension route has passed through S, R, A;*
4. *finite observation exactness, requiring the shared* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows to be the only numerical readback route between the extension row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *classifier stability, requiring componentwise* hsame *transport through H and displayed replay through C to preserve the same finite naturality route;*
6. *ledger non-escape, excluding quotient completions, selected representatives, host function equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking from the packet.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3501.1. The accepted packet displays exactly the two completion rows, the map row, the extension row, the finite observation rows, and the structural rows.

Source-before-extension exposure is read from M_X, F, E. The map row F is available only with the source completion observations in M_X, and the extension row E records the completion-facing transport of that same displayed map.

Target-consumer routing is read from M_Y together with S, R, A. The target completion row can be offered to a consumer only after the extension has exposed the finite stream windows, regular readback, and real seal used by the route.

Classifier stability is componentwise transport through H followed by replay through C. Since P and L only source and name the same displayed route, there is no separate row from which quotient completion equality, selected representatives, host function equality, choice, ambient completeness, Lean verification, or bridge credentials could be projected. □

Theorem 2544.3 (Metric completion naturality consumer nonescape)**.** *Every completion-facing or real-seal-facing consumer of an accepted* $\mathsf{MetricCompletionNaturality}^{\uparrow}$ *carrier*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L)$$

must factor through the finite route

$$M_X, F \longmapsto E \longmapsto S, R, A \longmapsto M_Y \longmapsto H, C, P, L.$$

Consequently the consumer receives only the displayed source completion row, continuous-map row, uniform-extension row, shared finite window, regular readback, terminal real seal, target completion row, transport, replay, provenance, and local name. It exports no quotient completion, no selected representative, no host function equality, no countable-choice schedule, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no bridge-checking credential.*

Proof. Start with a completion-facing or real-seal-facing read of the carrier. Projection first exposes the source completion row M_X and the continuous-map row F, so the consumer cannot begin from a hidden completed representative or a host function.

The extension row E is then the only row carrying the displayed map through the completion interface. It does not add a quotient completion object; it only names the finite extension route attached to the same source and map rows.

The read next passes through S, R, A. These rows fix the finite observation windows, the regular rational readback, and the terminal real seal before the target completion row M_Y is available.

Finally H, C, P, L transport, replay, source, and name the same displayed route. Because the carrier has no further coordinates, the stated excluded principles cannot be exported by any consumer of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionNaturality}^{\uparrow}$ packet is generated from source and target MetricCompletion rows, a ContinuousMap row, a UniformContinuousExtension row, shared StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$, seed)

Scope closed: Definition 3501.1, Definition 3501.2, and Definition 3501.3 seed the finite naturality route from source metric completion through displayed continuous-map extension to target completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-before-extension exposure, target-consumer routing, finite observation exactness, classifier stability, consumer nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.MetricCompletionNaturalityUp.

Chapter 2545

A Concrete Naming Certificate for $\mathsf{AsymptoticSequence}^{\uparrow}$

Following Hardy and Littlewood on asymptotic comparison of sequences, the $\mathsf{AsymptoticSequence}^{\uparrow}$ packet records finite tail-comparison data for displayed sequence names rather than importing host sequence equality or completed real analysis. It sits over the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, the limit-facing handoff of chapter 1413, the symmetric tail comparison of chapter 1738, and the cofinal route comparison of chapter 1255. The packet is a named consumer for Real and Completion tail comparisons: it exposes finite windows, thresholds, readbacks, transport, replay, provenance, and local naming, while refusing quotient streams, arbitrary subsequence choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2545.1 (AsymptoticSequence carrier)**.** *An* $\mathsf{AsymptoticSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, W, M, Q, L, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S and T are the two source sequence-name rows, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the inspected source cells and the corresponding comparison windows.*
3. *M is the tail-threshold row recording which displayed unary tail must be reached before the comparison classifier is read.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations attached to W and M.*
5. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row available only after the scheduled windows and rational readbacks are displayed.*
6. *E is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *provenance row for symmetric regular-Cauchy tail comparison.*
7. *F is the* $\mathsf{CofinalTailEquivalence}^{\uparrow}$ *provenance row for cofinal reindexing and tail-route comparison.*
8. *H records componentwise* hsame *transport for the source rows, windows, threshold, readback, limit handoff, tail-equivalence rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each asymptotic-comparison request through $S, T, W, M, Q, L, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{AsymptoticSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{CofinalTailEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^\uparrow}$ *row.*
The carrier exports no quotient equality of streams, no host asymptotic relation, no selected subsequence, no trichotomy row, and no ambient $\mathsf{Real}^\uparrow$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.AsymptoticSequenceUp.AsymptoticSequenceTasteGate single carrier alignment`

Theorem 2545.2 (AsymptoticSequence NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticSequence}^\uparrow$ *carrier*

$$A = (S, T, W, M, Q, L, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^\uparrow}$ *surface consists exactly of carrier admission for the two source sequence rows, paired* $\mathsf{StreamName}^\uparrow$ *windows, the displayed tail threshold,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{RealSequenceLimit}^\uparrow$ *handoff,* $\mathsf{CauchyTailEquivalence}^\uparrow$ *provenance,* $\mathsf{CofinalTailEquivalence}^\uparrow$ *provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted comparison reads W and M before Q, reads Q before L, E, F, and reaches H, C, P, N only as structural support for the same finite route.*

Proof. Project the carrier of Definition 3502.1. The source rows S, T expose only the finite $\mathsf{StreamName}^\uparrow$ windows stored in W, and the comparison cannot inspect a tail before the threshold row M has been displayed.

The readback row Q is then forced by the $\mathsf{RegSeqRat}^\uparrow$ surface of chapter 301. It consumes the same windows and threshold data rather than a host sequence value.

The analytic rows are downstream of that readback. The limit-facing handoff L is the finite route supplied by chapter 1413, while E and F record the symmetric and cofinal tail provenance of chapter 1738 and chapter 1255.

Finally H, C, P, N transport, replay, source, and name only the displayed route. Since no carrier coordinate contains quotient stream equality, host asymptotic comparison, a selected subsequence, trichotomy, or ambient completeness, none can appear as a local NameCert obligation. □

Theorem 2545.3 (AsymptoticSequence tail classifier)**.** *Let $A = (S, T, W, M, Q, L, E, F, H, C, P, N)$ and $A' = (S', T', W', M', Q', L', E', F', H', C', P', N')$ be accepted* $\mathsf{AsymptoticSequence}^\uparrow$ *carriers. Their accepted tail classifiers agree exactly when the paired source rows, displayed* $\mathsf{StreamName}^\uparrow$ *windows, tail-threshold rows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks,* $\mathsf{RealSequenceLimit}^\uparrow$ *handoff rows,* $\mathsf{CauchyTailEquivalence}^\uparrow$ *provenance rows,* $\mathsf{CofinalTailEquivalence}^\uparrow$ *provenance rows, componentwise* hsame *transports, displayed* Cont *replays, package provenance rows, and local naming rows agree under the displayed transport. The classifier has no further coordinate for host sequence equality, quotient tails, arbitrary subsequence choice, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. For the forward direction, read an accepted classifier agreement through the NameCert obligations of Definition 3502.2. The agreement must pass through $S, T, W, M, Q, L, E, F, H, C, P, N$, so each corresponding displayed row is preserved under the componentwise transport.

For the reverse direction, assume the listed rows agree under the displayed transport. Carrier admission reconstructs the same paired windows and tail threshold, the same $\mathsf{RegSeqRat}^\uparrow$ readback, and the same limit, Cauchy-tail, and cofinal-tail provenance rows.

Displayed replay through C and C' then gives the same consumer route, while P, N and P', N' record the same source and local naming surface. The absence of quotient, choice, host equality, and completeness coordinates shows that no hidden classifier field remains. □

Falsifiable prediction: Every $\mathsf{AsymptoticSequence}^\uparrow$ comparison read factors through paired StreamName windows, one displayed tail threshold, RegSeqRat readback, RealSequenceLimit

handoff, CauchyTailEquivalence provenance, CofinalTailEquivalence provenance, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, realsequencelimit, cauchytailequivalence, cofinaltailequivalence: $\mathsf{AsymptoticSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired source windows, a tail threshold, rational readback, limit handoff, symmetric tail comparison, cofinal route comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticSequence}^{\uparrow}$ packet is generated from paired source sequence rows, finite StreamName windows, a tail-threshold row, RegSeqRat readback, RealSequenceLimit handoff, CauchyTailEquivalence and CofinalTailEquivalence provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3502.1, Definition 3502.2, and Definition 3502.3 seed the finite tail-comparison certificate for sequence-limit and Cauchy-criterion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, host asymptotic comparison, arbitrary subsequence choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{CofinalTailEquivalence}^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 2546

A Concrete Naming Certificate for $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$

Following the Heine–Borel and Bishop finite-subcover discipline, the $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ packet records the finite-net criterion that lets a compactness consumer use displayed metric cover data without importing arbitrary open-cover compactness. It sits next to the compact metric carrier of chapter 129, the finite center family of chapter 1767, the located interval boundary of chapter 394, and the terminal real-name seal of chapter 38. The packet names a finite criterion over already displayed rows; it does not select subcovers from arbitrary covers, choose representatives, assert countable choice, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2546.1 (FiniteSubcoverCriterion carrier)**.** *A* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, E, L, R, Q, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposed only through its displayed finite-net and metric coverage fields.*
2. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *center-bundle row for the requested positive scale.*
3. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *or located source-window row when the compact source is an interval-facing carrier.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to endpoint, radius, and distance comparisons already displayed by the source.*
5. *Q is the cover-read ledger, recording for each bundled center the finite witness row that covers the corresponding metric ball.*
6. *D is the dyadic or rational comparison ledger certifying that the center radius and mesh slack fit the requested coverage scale.*
7. *H records componentwise* hsame *transport for the compact source, finite-net row, located window, Real seal, cover-read ledger, comparison ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a finite cover request through $K, E, L, R, Q, D, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, E, L, R, Q, D, H, C, P, N$. It has no coordinate for arbitrary open covers, countable choice, selected representatives, a host finite-subcover operation, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteSubcoverCriterionUp.FiniteSubcoverCriterionTasteGate single carrier alignment`

Theorem 2546.2 (FiniteSubcoverCriterion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *carrier $S = (K, E, L, R, Q, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the compact metric source, finite epsilon-net row, located source-window row, Real seal, cover-read ledger, rational comparison ledger, transport, replay, provenance, and local naming rows;*
2. *finite-net exposure, requiring every cover read to display the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *center bundle E before any covering witness is inspected;*
3. *source compatibility, requiring K, L, R to be read as the compact metric, located-window, and Real-seal rows already carried by sibling chapters;*
4. *radius exactness, requiring Q and D to record the finite center-to-cover and radius-slack checks that make the criterion usable by downstream compactness consumers;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no open-cover compactness, countable choice, selected representatives, host finite-subcover operation, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3503.1. Its coordinates are exactly the compact source, finite net, located source window, Real seal, cover-read ledger, comparison ledger, and structural rows.

Finite-net exposure is immediate from the order of the displayed packet. A cover consumer must first read E, so each later covering witness is tied to a named center bundle rather than to an arbitrary family of open sets.

Source compatibility is supplied by K, L, R. These rows identify the compact metric, located-window, and Real-seal surfaces already present in the sibling chapters, while Q and D record only finite coverage and comparison checks.

Finally, the refused objects have no coordinate in S. Transport, replay, provenance, and local naming preserve the same finite rows, so no accepted packet exports open-cover compactness, countable choice, selected representatives, host finite-subcover selection, quotient real equality, or ambient Real completeness. □

Theorem 2546.3 (FiniteSubcoverCriterion net extraction)**.** *Every compactness-facing read from an accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported object is the displayed finite-net subcover criterion over $K, E, L, R, Q, D, H, C, P, N$, not an open-cover compactness theorem.

Proof. Begin with an accepted carrier $S = (K, E, L, R, Q, D, H, C, P, N)$. The compactness-facing source is K, and the finite centers are exposed by E before the located and Real-facing rows L and R are read.

The cover ledger Q assigns the displayed cover witnesses only to the bundled centers, while D records the rational slack needed to pass from center coverage to the requested metric scale. Thus the extraction is a finite read from the listed packet rows.

Replay through C, with transport through H and provenance through P, preserves that route. The local naming row N therefore exports the finite-net criterion and cannot add open-cover compactness, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2546.4 (FiniteSubcoverCriterion compact-metric consumer)**.** *Let $S = (K, E, L, R, Q, D, H, C, P, N)$ be an accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *carrier. Any compact-metric consumer that reads the finite subcover criterion must first display the* $\mathsf{CompactMetric}^{\uparrow}$ *source data K, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *center family E, the* $\mathsf{LocatedInterval}^{\uparrow}$ *or located-radius boundary row L, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows carried by D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R before the cover-read ledger Q is consumed. The exported criterion is therefore a finite compact-metric route through $K, E, L, D, R, Q, H, C, P, N$, with no arbitrary open-cover compactness, selected representative, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3503.1. The only compact-metric source coordinate is K, and the only finite center-family coordinate is E.

By Definition 3503.2, source compatibility and finite-net exposure require K and E to be visible before any cover witness is inspected. The located boundary row L and the terminal seal row R are part of the same source-compatibility boundary.

By Definition 3503.3, every compactness-facing read factors through the finite route from compact metric data to finite epsilon nets, located boundary rows, and the Real seal. The dyadic tolerance ledger D records the finite radius slack that makes the cover-read ledger Q meaningful to that route.

The remaining coordinates H, C, P, N only transport, replay, source, and name the same finite packet. Since the carrier has no row for arbitrary open-cover compactness, selected representatives, countable choice, quotient real equality, or ambient Real completeness, no compact-metric consumer can obtain one from the finite subcover criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a FiniteEpsilonNetUp center bundle, a LocatedIntervalUp source-window row, a RealUp seal, finite cover reads, rational comparison ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3503.1, Definition 3503.2, and Definition 3503.3 seed the finite-net compactness criterion consumed by later compact-continuity and Heine–Borel routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected representatives, a host finite-subcover operation, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, source compatibility, radius exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubcoverCriterionUp.

Chapter 2547

A Concrete Naming Certificate for $\mathsf{RieszMarkov}^{\uparrow}$

Following Riesz and Markov, the $\mathsf{RieszMarkov}^{\uparrow}$ packet isolates the compact positive functional representation surface: a positive linear functional on continuous real-valued observables over a displayed compact source is read only through a representing Radon row and an integral-readback ledger. It consumes the real scalar boundary of chapter 38, the compact source of chapter 60, the continuous observable rows of chapter 130, the regular measure packet of chapter 2263, and the broader representation boundary of chapter 1643. The chapter names the finite BEDC obligation surface needed by compact-analysis consumers without choosing a representing measure from a host theorem.

2547.1 Compact positive functional carrier

Definition 2547.1 (RieszMarkov compact positive functional carrier)**.** *A* $\mathsf{RieszMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{RieszMarkov}(K, F, L, M, I, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through* $\mathsf{Compact}^{\uparrow}$ *over the inherited* $\mathsf{Real}^{\uparrow}$ *scalar boundary;*
2. *F is the continuous real-valued observable row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *with codomain* $\mathsf{Real}^{\uparrow}$*;*
3. *L is the positive functional ledger, recording linear scalar reads and positivity reads on the displayed observable rows;*
4. *M is the representing* $\mathsf{RadonMeasure}^{\uparrow}$ *row over the same compact source;*
5. *I is the integral-readback row comparing each displayed functional value with the corresponding integral against M;*
6. *H is the componentwise* hsame *transport ledger for K, F, L, M, I, C, P, N;*
7. *C is the displayed* Cont *replay ledger for compact source, observable, functional, representing measure, and integral-readback routes;*
8. *P is the* Pkg *provenance over* $\mathsf{RieszMarkov}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{RieszRepresentation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, M, I, H, C, P, N$. It does not identify host functions, host measures, arbitrary sigma-algebras, quotient measure classes, selected representing measures, or ambient functional extensionality.

2547.2 Certificate obligations

Theorem 2547.2 (RieszMarkov NameCert obligations)**.** *For every accepted* $\mathsf{RieszMarkov}^{\uparrow}$ *carrier*

$$R = \mathsf{RieszMarkov}(K, F, L, M, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, real-valued continuity admission through F, positive-functional ledger visibility through L, representing Radon row visibility through M, integral-readback exactness through I, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

The certificate refuses arbitrary choice of representing measures, host sigma-algebras beyond the displayed $\mathsf{RadonMeasure}^{\uparrow}$ *rows, quotient equality of measures, host equality of functionals, and ambient functional extensionality. A consumer may use the representation only by reading the finite rows of R.*

Lean4 ✓ `BEDC.Derived.RieszMarkovUp.RieszMarkovCarrier namecert obligations`

Proof. Project the carrier of Definition 3504.1. The compact source and continuous observable are fixed by K and F, so the functional ledger L is read on displayed observables rather than on a host function space.

The representing object is the displayed Radon row M. The readback row I is the only route from a functional value to an integral value, and it is attached to the same compact source and observable rows.

Transport and replay are componentwise through H and C, while P and N source and name the route. Since no carrier coordinate stores a selector for representing measures, a larger host sigma-algebra, a quotient measure equality, host functional equality, or functional extensionality, the stated rows exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$, seed)

Scope closed: Definition 3504.1 and Definition 3504.2 seed the compact positive functional representation packet over Real, Compact, ContinuousMap, RadonMeasure, RieszRepresentation, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary representing-measure choice, host sigma-algebras beyond displayed Radon rows, quotient measure equality, ambient functional extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-functional ledger visibility, representing Radon row visibility, integral-readback exactness, transport replay, and consumer refusal for BEDC.Derived.RieszMarkovUp.

Constructive story: A $\mathsf{RieszMarkov}^{\uparrow}$ packet is generated from compact source, continuous real-valued observable, positive functional ledger, representing Radon row, integral-readback row, componentwise transport, replay, provenance, and local NameCert rows.

Chapter 2548

A Constructive Apartness Certificate for $\mathsf{MetricApartness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{MetricApartness}^{\uparrow}$ packet records the positive-distance separation surface of a constructive metric space as finite BEDC NameCert data. It sits after the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the separated zero-distance classifier of chapter 1411, and the completion boundary of chapter 1506. The packet supplies a finite apartness interface for metric and completion consumers; it does not import host inequality, real trichotomy, quotient classes, selected representatives, or ambient completeness.

Definition 2548.1 (MetricApartness carrier)**.** *A* $\mathsf{MetricApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, x, y, D, R, Z, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the displayed distance-read interface.*
2. *x and y are the two inspected point rows over the same metric source.*
3. *D is the metric-distance witness row for the finite read $d(x, y)$.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *positive-distance row, recording that the displayed distance has a positive real witness.*
5. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance refusal row, recording that the accepted apartness read rejects the zero-distance branch by displayed real data.*
6. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier-comparison row, used only to compare the zero-distance boundary with the displayed point classifier.*
7. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *finite-window transport row for completion-facing consumers.*
8. *H records componentwise* hsame *transport for the metric source, point rows, distance witness, real positivity row, zero-distance refusal row, separated comparison, finite-window transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an apartness request through $M, x, y, D, R, Z, S, Q, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$ *row.*

The classifier compares accepted apartness packets only through these displayed rows. It contains no host inequality predicate, no trichotomy of real numbers, no quotient-completion class, no selected representative, and no ambient completeness witness.

Lean4 ✓ `BEDC.Derived.MetricApartnessUp.TasteGate.MetricApartnessTasteGate single carrier alignment`

Theorem 2548.2 (MetricApartness NameCert obligations)**.** *For every accepted* $\mathsf{MetricApartness}^{\uparrow}$ *carrier* $A = (M, x, y, D, R, Z, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source and point rows, distance-witness visibility through* D*, positive-distance readback through* R*, zero-distance refusal through* Z*, separated classifier comparison through* S*, completion-facing finite-window transport through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every apartness read therefore follows the route*

$$M, x, y \longmapsto D \longmapsto R, Z \longmapsto S, Q \longmapsto H, C, P, N.$$

No additional coordinate can decide apartness by host inequality, real trichotomy, quotient class comparison, selected representative equality, or ambient completeness.

Proof. Project the carrier of Definition 3505.1. The rows M, x, y expose the metric source and the two inspected points, and D is the only row from which their displayed distance can be read.

The real-facing branch is then forced through R and Z. The row R supplies positive-distance evidence, while Z records refusal of the zero-distance branch by the same $\mathsf{Real}^{\uparrow}$ readback surface.

Separated and completion-facing consumers are downstream. The row S compares the zero-distance boundary with the separated classifier, and Q transports finite-window completion reads through $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ without creating quotient classes or selected representatives.

Finally H, C, P, N preserve, replay, source, and name the displayed route. Since no row stores host inequality, real trichotomy, quotient comparison, selected representative equality, or ambient completeness, the listed coordinates exhaust the NameCert surface. □

Theorem 2548.3 (MetricApartness completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchySequenceEquivalence}$ *consumer of an accepted* $\mathsf{MetricApartness}^{\uparrow}$ *packet must read the finite route* $M, x, y, D, R, Z, S, Q, H, C, P, N$ *before treating the two displayed names as apart. The exported handoff is only the positive-distance certificate together with zero-distance refusal, separated-classifier comparison, finite-window transport, componentwise transport, replay, provenance, and local naming. It does not export host inequality, real trichotomy, quotient-completion equality, selected representatives, or an ambient completeness principle.*

Proof. Begin with a completion-facing apartness request. The metric source and inspected point rows M, x, y must be displayed before the distance witness D can be consumed.

The rows R and Z bind the constructive apartness decision to real readback: one row supplies positive-distance evidence, and the other refuses the zero-distance branch. This gives the separated row S a displayed classifier boundary to compare.

The finite-window row Q is then the only completion-facing handoff. It transports the same distance and refusal data through $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ consumers, while H, C, P, N record transport, replay, provenance, and local naming.

Thus the completion consumer receives exactly the finite apartness route described above, with no hidden route through host inequality, trichotomy, quotient classes, selected representatives, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{MetricApartness}^{\uparrow}$ read factors through a MetricUp source, two displayed point rows, a distance witness, RealUp positive-distance evidence, RealUp zero-distance refusal, a SeparatedMetric classifier comparison, MetricCompletion and CauchySequenceEquivalence finite-window transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, real, separatedmetric, metriccompletion, cauchysequenceequivalence: $\mathsf{MetricApartness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, two displayed point rows, positive-distance evidence, zero-distance refusal, separated classifier comparison, completion-facing finite-window transport, transport, replay, provenance, and local naming as one apartness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricApartness}^{\uparrow}$ packet is generated from a MetricUp source row, two displayed point rows, a metric-distance witness, RealUp positive-distance evidence, RealUp zero-distance refusal, a SeparatedMetric classifier-comparison row, MetricCompletion and CauchySequenceEquivalence finite-window transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3505.1, Definition 3505.2, and Definition 3505.3 seed the finite constructive apartness surface for metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inequality, real trichotomy, quotient-completion equality, selected representatives, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-witness visibility, positive-distance evidence, zero-distance refusal, separated classifier comparison, completion-facing finite-window transport, transport and replay stability, provenance non-escape, and local naming for BEDC.Derived.MetricApartnessUp.

Chapter 2549

A Concrete Naming Certificate for $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ packet records the finite orbit-to-Cauchy-completion handoff produced by a contraction before any fixed-point endpoint is sealed. It sits after the contraction-map surface of chapter 1418, the Picard-step ledger of chapter 395, the complete-metric service of chapter 128, the convergence-modulus boundary of chapter 1438, and the fixed-point endpoint discipline of chapter 396. The packet exposes the finite carrier and residual-modulus rows that a completion consumer must read before speaking about a completed orbit point.

Definition 2549.1 (ContractiveCauchyOrbit carrier)**.** *A* $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, T, I, W, R, M, K, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row over which distances between displayed orbit entries are read.*
2. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row, including the displayed contraction ratio and the self-map application route.*
3. *I is the* $\mathsf{PicardContraction}^{\uparrow}$ *finite iterate ledger recording the visible orbit entries $x_0, \ldots, x_n$ and the application rows $x_{k+1} = T(x_k)$.*
4. *W is the finite iterate-window exactness row, recording which adjacent orbit windows are available to every tolerance request.*
5. *R is the residual-distance row, storing the adjacent distance comparisons and ratio propagation read from X, T, I, W.*
6. *M is the extracted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row, obtained from the residual ledger without selecting a limit.*
7. *K is the completion handoff row, naming the* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer that may read the Cauchy orbit after M is displayed.*
8. *E is the endpoint-seal boundary available only after the completion handoff has been read by a downstream fixed-point or real-name consumer.*

9. *H records componentwise* hsame *transport for the metric, contraction, iterate, window, residual, modulus, completion handoff, endpoint boundary, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from contraction admission through finite orbit windows, residual comparisons, modulus extraction, completion handoff, and endpoint separation.*
11. *P is the* Pkg *provenance over* ContractiveCauchyOrbit$^{\uparrow}$, ContractionMapping$^{\uparrow}$, PicardContraction$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{ContractiveCauchyOrbit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected fixed point, host limit equality, quotient completion representative, function extensionality, countable choice, or ambient completeness outside the displayed CompleteMetric$^{\uparrow}$ *row.*

Lean4 (definition) `BEDC.Derived.ContractiveCauchyOrbitUp.ContractiveCauchyOrbitUp`
Lean4 ✓ `BEDC.Derived.ContractiveCauchyOrbitUp.ContractiveCauchyOrbitTasteGate single carrier alignment`

Theorem 2549.2 (ContractiveCauchyOrbit NameCert obligations)**.** *For every accepted* ContractiveCauchyOrbit$^{\uparrow}$ *carrier*

$$O = (X, T, I, W, R, M, K, E, H, C, P, N),$$

the local NameCert$_{\text{ContractiveCauchyOrbit}^{\uparrow}}$ *surface consists exactly of carrier admission, contraction-row visibility, finite iterate-window exactness, residual Cauchy-modulus extraction, completion handoff, endpoint-boundary separation, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger non-escape for the rows of Definition 3506.1.*

Proof. Carrier admission projects the displayed tuple. The rows X, T, I, W expose the complete metric, contraction map, finite Picard ledger, and exact finite windows before any convergence-facing row is read.

The residual row R is then forced through the same finite data. It records adjacent distances and contraction-ratio propagation, so the modulus row M is extracted from displayed orbit windows rather than from a host convergence theorem.

The completion handoff K and endpoint boundary E remain downstream. They may be inspected only after the residual-to-modulus route has been displayed, and they do not add a selected fixed point or quotient representative.

Finally H, C, P, N transport, replay, source, and name exactly these rows. Thus every NameCert obligation is visible in the carrier, and the ledger has no escape route outside the twelve displayed coordinates. □

Theorem 2549.3 (ContractiveCauchyOrbit completion handoff)**.** *Every completion-facing consumer of an accepted* ContractiveCauchyOrbit$^{\uparrow}$ *carrier must read the ordered route*

$$\text{ContractionMapping}^{\uparrow},\ \text{PicardContraction}^{\uparrow},\ \text{CompleteMetric}^{\uparrow} \longmapsto W, R, M \longmapsto K, E, H, C, P, N.$$

The consumer receives the finite iterate windows, residual-distance comparisons, and extracted Cauchy modulus before it may read the completion handoff. It receives no Banach fixed point, selected limit, host function equality, quotient completion representative, or ambient completeness beyond the named CompleteMetric$^{\uparrow}$ *row.*

Proof. Project the contraction and iteration rows. The $\mathsf{ContractionMapping}^{\uparrow}$ row supplies the self-map and ratio data, while the $\mathsf{PicardContraction}^{\uparrow}$ row supplies the finite orbit ledger.

Read the complete-metric row next. All residual comparisons in R are distance reads through X, and the exact windows W determine which finite orbit entries are visible to a tolerance request.

The row M is therefore a Cauchy-modulus extraction from finite residual data. The completion handoff K may consume this route, and the endpoint boundary E may be read only after K.

The structural rows preserve the same dependency order. Since no row stores a selected endpoint or host equality, every completion-facing consumer remains confined to the displayed handoff. □

Theorem 2549.4 (ContractiveCauchyOrbit ledger non-escape)**.** *Every public read of an accepted* $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ *packet factors through*

$$X, T, I, W, R, M, K, E, H, C, P, N.$$

In particular, the residual Cauchy-modulus extraction and completion handoff are visible as finite BHist *rows, and no consumer can replace them by an external convergence theorem, a chosen completion point, a quotient representative, countable choice, or host fixed-point equality.*

Proof. Read the carrier definition together with the NameCert obligation theorem. They expose exactly the twelve rows listed in the statement, with finite orbit windows and residual comparisons preceding the modulus row.

Apply Definition 3506.3. Any completion-facing read must pass through the displayed contraction, Picard, complete-metric, window, residual, and modulus route before K and E are available.

The transport, replay, provenance, and naming rows add no analytic coordinate. Therefore a public consumer can only inspect the displayed carrier route, and the excluded host, choice, quotient, and endpoint objects remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ completion read factors through one CompleteMetric row, one ContractionMapping row, one PicardContraction finite orbit ledger, one finite iterate-window exactness row, one residual-distance row, one CauchyConvergenceModulus extraction, one completion handoff, one endpoint boundary, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: contractionmapping, picardcontraction, completemetric, cauchy-convergence-modulus, banachfixedpoint: none is bijective with $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ because the packet jointly stores the contraction row, Picard orbit ledger, complete-metric distance reads, finite window exactness, residual modulus extraction, completion handoff, endpoint boundary, transport, replay, provenance, and local naming as one finite completion-facing orbit.

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3506.1, Definition 3506.2, Definition 3506.3, and Definition 3506.4 seed the finite contraction-orbit route from Picard rows to Cauchy completion handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Banach fixed points, selected limits, quotient completion representatives, host function equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-row visibility, finite iterate-window exactness, residual Cauchy-modulus extraction, completion handoff, endpoint-boundary separation, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.ContractiveCauchyOrbitUp.

Constructive story: A ContractiveCauchyOrbit$^{\uparrow}$ packet is generated from a complete-metric row, contraction-map row, PicardContraction finite iterate ledger, finite window exactness row, residual-distance row, extracted CauchyConvergenceModulus row, completion handoff, endpoint boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2550

A Concrete Naming Certificate for $\mathsf{CompletionSubsetClosure}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{CompletionSubsetClosure}^{\uparrow}$ packet records the traditional fact that a closed subset of a complete metric completion remains complete, but only as finite BEDC NameCert data. It sits over the complete separable metric surface of chapter 1735, the dense embedding route of chapter 3351, the dense Cauchy embedding packet of chapter 2250, the Polish-space readback of chapter 1490, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a closed-subset completion surface for downstream completion consumers; it is not open-cover compactness, a quotient equality of completed points, countable choice, selected limit extraction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2550.1 (Completion subset closure carrier)**.** *A* $\mathsf{CompletionSubsetClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, E, D, P, S, W, Q, R, L, H, C, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *source row exposing the complete separable metric packet from which the closed subset is read.*
2. *E is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *dense-embedding row naming the completion-facing source image.*
3. *D is the* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ *handoff row, recording the Cauchy-name route by which subset points are inspected.*
4. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *consumer row, used only after the complete separable and dense embedding rows have been displayed.*
5. *S is the closed-subset ledger, listing subset membership, closedness witnesses, and boundary refusal rows over the completion source.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used to inspect subset membership and Cauchy approximation rows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed Cauchy and distance-budget data.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite distance and radius comparisons read by the subset closure ledger.*

9. *L is the limit-handoff row, recording that every accepted Cauchy read remaining in S reaches a displayed subset limit row before any downstream completion consumer sees it.*
10. *H records componentwise* hsame *transport for the complete metric source, dense embeddings, closed-subset ledger, stream windows, readback rows, Real seals, limit handoff, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from subset Cauchy reads through $M, E, D, P, S, W, Q, R, L, H$.*
12. *G is the* Pkg *provenance over* $\mathsf{CompletionSubsetClosure}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, MetricCompletionDenseEmbe DenseCauchyEmbedding$^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, D, P, S, W, Q, R, L, H, C, G, N$. It has no coordinate for open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2550.2 (CompletionSubsetClosure NameCert obligations)**.** *For every accepted* CompletionSubsetClosure$^{\uparrow}$ *carrier $K = (M, E, D, P, S, W, Q, R, L, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete separable metric source, metric-completion dense embedding, dense Cauchy embedding, Polish-space consumer row, closed-subset ledger, finite stream windows, regular rational readback, Real seal, limit handoff, componentwise transport, displayed replay, provenance, and local naming.*

Every completion-facing read must therefore pass through

$$M \longmapsto E \longmapsto D \longmapsto P \longmapsto S \longmapsto W \longmapsto Q \longmapsto R \longmapsto L \longmapsto (H, C, G, N)$$

before it can treat a Cauchy read in the subset as closed under completion. The route exports no open-cover compactness theorem, no quotient equality of completion classes, no selected limit, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3507.1. The rows M, E, D, P expose the complete separable source, dense completion image, dense Cauchy handoff, and Polish consumer boundary before the subset ledger can be read.

Next inspect the closed-subset and finite observation rows. The ledger S is read only through finite windows W, regular rational data Q, and Real seals R, so a subset Cauchy request has a displayed membership and boundary route before a limit row is exposed.

Finally read the structural rows. The limit handoff L is transported by H, replayed by C, sourced by G, and named by N. Since the carrier has no row for open covers, quotient completion equality, selected limits, countable choice, trichotomy, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CompletionSubsetClosureUp.CompletionSubsetClosure namecert obligations`

Theorem 2550.3 (CompletionSubsetClosure dense embedding handoff)**.** *Every accepted* CompletionSubsetClosure$^{\uparrow}$ *packet hands a subset-completion consumer the displayed route*

$\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseCauchyEmbedding}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, StreamNa

The handoff is the finite closed-subset completion route carried by K; it is not a host theorem about all closed subsets of all complete spaces.

Proof. Begin with an accepted packet K. The complete separable source M and dense embedding row E identify the completion surface, while the dense Cauchy row D fixes how subset Cauchy names enter the packet.

The Polish row P is downstream of those completion rows, and the observation rows W, Q, R display the finite reads used by the closed-subset ledger S. Thus the subset-completion consumer sees only the named route through the listed sibling chapters.

Transport, replay, provenance, and local naming preserve that route. No host closed-subset theorem or ambient completion object is introduced by the handoff. □

Theorem 2550.4 (CompletionSubsetClosure nonescape)**.** *Every accepted* $\mathsf{CompletionSubsetClosure}^{\uparrow}$ *completion read remains inside the displayed rows* $M, E, D, P, S, W, Q, R, L, H, C, G, N$*. In particular, the packet cannot export open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet coordinates in order. The completion source, dense embedding, dense Cauchy handoff, and Polish consumer rows are exactly M, E, D, P; the subset boundary is exactly S.

The finite observation data is exactly W, Q, R, and the only completion-facing exit is the displayed limit handoff L. Componentwise transport, replay, provenance, and naming through H, C, G, N do not add new coordinates.

Thus every accepted read is exhausted by the displayed rows, and the refused principles have no carrier position from which they could be exported. □

Falsifiable prediction: Every accepted $\mathsf{CompletionSubsetClosure}^{\uparrow}$ completion read must display complete-separable source data, dense completion embedding data, dense Cauchy embedding data, Polish consumer data, a closed-subset ledger, StreamName windows, RegSeqRat readback, a RealUp seal, a limit handoff, transport, replay, provenance, and local naming before any closed-subset completion claim is public.

Independence witness: complete-separable-metric, metric-completion-dense-embedding, dense-cauchy-embedding, polishspace, streamname, regseqrat, real: $\mathsf{CompletionSubsetClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a closed-subset ledger, finite observation windows, regular readback, Real seals, a limit handoff, transport, replay, provenance, and local naming over the completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompletionSubsetClosure}^{\uparrow}$ packet is generated from CompleteSeparableMetricUp source rows, MetricCompletionDenseEmbeddingUp and DenseCauchyEmbeddingUp completion rows, a PolishSpaceUp consumer row, a closed-subset ledger, StreamName windows, RegSeqRat readback, RealUp seals, a limit handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3507.1, Definition 3507.2, Definition 3507.3, and Definition 3507.4 seed the closed-subset completion route over the L10 completion siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-subset ledger exactness, dense-embedding handoff, limit-handoff non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionSubsetClosureUp.

Chapter 2551

A Concrete Naming Certificate for DnaBoundaryTriggerClosure$^{\uparrow}$

The DnaBoundaryTriggerClosure$^{\uparrow}$ packet realizes the boundary-closure structure of Definition 9462.1306 as finite BEDC NameCert data. It sits over the six boundary-tile coverage packet of chapter 3470, the finite trigger algebra of chapter 3478, the cellular trace window of chapter 3481, and the weak Konig boundary refusal surface of chapter 3482. The packet names boundary-trigger closure for six-tile DNA window rows; it is not a biological code table, a stochastic evolutionary model, an infinite branch selector, a quotient codon semantics, or a host closure oracle.

Definition 2551.1 (DnaBoundaryTriggerClosure carrier)**.** *A* DnaBoundaryTriggerClosure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, A, T, W, F, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-tile coverage row, exposing the six-tile trigger and coverage-rank data.*
2. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *finite trigger-algebra row, recording closure of displayed support rows under boundary-trigger operations.*
3. *T is the* CellularWindowTrace$^{\uparrow}$ *trace row, used only to display finite local evolution windows attached to accepted boundary triggers.*
4. *W is the* WeakKonigBoundary$^{\uparrow}$ *refusal row, recording that the closure packet does not export an infinite branch selector or countable-choice schedule.*
5. *F is the finite frontier ledger listing the boundary codons, Q_2 completion faces, stage rows, and pressure rows read from the source vision.*
6. *L is the closure ledger, recording that each accepted boundary-trigger closure step is obtained from displayed six-tile rows, trigger algebra, trace windows, and weak-branch refusals.*
7. *H records componentwise* hsame *transport for boundary-tile coverage, trigger algebra, trace windows, weak-boundary refusals, frontier rows, closure rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a boundary-trigger closure request through B, A, T, W, F, L, H.*
9. *P is the* Pkg *provenance over* DnaBoundaryTriggerClosure$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAl CellularWindowTrace$^{\uparrow}$, WeakKonigBoundary$^{\uparrow}$, *the boundary-closure source vision,* BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, A, T, W, F, L, H, C, P, N$. It has no coordinate for hidden genetic-code semantics, stochastic dynamics, infinite branch choice, quotient codon equality, or a host closure oracle.

Lean4 ✓ `BEDC.Derived.DnaBoundaryTriggerClosureUp.DnaBoundaryTriggerClosureTasteGate single carrier alignment`

Theorem 2551.2 (DnaBoundaryTriggerClosure NameCert obligations)**.** *For every accepted* DnaBoundaryTriggerCl *carrier $K = (B, A, T, W, F, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for six-boundary-tile coverage, window-coverage trigger algebra, cellular window trace, weak Konig boundary refusal, finite frontier rows, closure ledger rows, componentwise transport, displayed replay, provenance, and local naming.*

Every boundary-trigger closure read must therefore route through

$$B \longmapsto A \longmapsto T \longmapsto W \longmapsto F \longmapsto L \longmapsto (H, C, P, N)$$

before it can be consumed as a finite closure packet for DNA window rows. The route exports no biological code table, stochastic dynamics, infinite branch selector, quotient codon equality, or host closure oracle.

Proof. Project the carrier of Definition 3508.1. The rows B, A, T, W expose the four sibling surfaces: six-boundary coverage, trigger algebra, cellular trace, and weak-branch refusal.

The frontier and closure ledgers are finite. The row F records only displayed boundary codons, Q_2 completion faces, stage rows, and pressure rows, while L records closure steps obtained from those rows and the sibling packets.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no hidden semantics, stochastic model, infinite branch selector, quotient equality, or host oracle is present in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 2551.3 (DnaBoundaryTriggerClosure nonescape)**.** *Every accepted* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *boundary-trigger closure read passes through displayed six-tile boundary rows, finite trigger algebra rows, cellular trace rows, weak-branch refusal rows, frontier rows, closure rows, transport, replay, provenance, and local naming. The packet cannot escape to a biological code table, stochastic dynamics, infinite branch choice, quotient codon equality, or host closure oracle.*

Proof. Read B first. It supplies the six-boundary tile coverage row, so the trigger source is a displayed finite boundary packet rather than an implicit biological table.

Read A, T, W next. The trigger algebra closes only finite support rows, the cellular trace row supplies only finite local windows, and the weak-boundary row records the refusal of infinite branch selection.

The rows F, L, H, C, P, N then bind the frontier, closure, transport, replay, provenance, and naming surface. No further coordinate remains, so an accepted read cannot export any of the refused objects. □

Falsifiable prediction: Every boundary-trigger closure read for $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ must pass through displayed six-tile boundary rows, finite trigger-algebra rows, cellular trace windows, weak-branch refusal rows, frontier rows, closure rows, transport, replay, provenance, and local naming; omitting any displayed six-tile boundary row refutes the packet.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, cellular-window-trace, weak-konig-boundary: $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six-tile coverage, finite trigger algebra, cellular trace windows, weak-branch refusal, frontier ledgers, closure ledgers, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.DnaBoundaryTriggerClosureUp.TasteGate` supplies explicit $\mathsf{Nontrivial\ DnaBoundaryTriggerClosure}^{\uparrow}$ and $\mathsf{FieldFaithful\ DnaBoundaryTriggerClosure}^{\uparrow}$ instances preserving $B, A, T, W, F, L, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ packet is generated from SixBoundaryTileCoverageUp boundary rows, WindowCoverageTriggerAlgebraUp finite trigger rows, CellularWindowTraceUp finite trace windows, WeakKonigBoundaryUp refusal rows, finite frontier ledgers, closure ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3508.1, Definition 3508.2, and Definition 3508.3 seed the finite boundary-trigger closure packet for six-tile DNA window rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological code-table identity, stochastic evolutionary dynamics, infinite branch choice, quotient codon equality, host closure oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, six-tile boundary coverage, trigger-algebra exactness, trace-window handoff, weak-branch refusal, Nontrivial DnaBoundaryTriggerClosureUp, and FieldFaithful DnaBoundaryTriggerClosureUp for BEDC.Derived.DnaBoundaryTriggerClosureUp.

Chapter 2552

A Concrete Naming Certificate for $\mathsf{Ceva}^{\uparrow}$

Following Ceva, the $\mathsf{Ceva}^{\uparrow}$ packet records the finite incidence data for three cevians in a triangle and the concurrence witness that makes the side-intersection rows compatible. It sits beside the affine carrier of chapter 206, the projective incidence boundary of chapter 207, and the bilinear pairing row of chapter 146. The chapter names only triangle vertices, side-incidence rows, cevian rows, concurrence, transport, replay, provenance, and local naming; it does not export a coordinate determinant proof, a host projective plane, or an analytic area computation.

Definition 2552.1 (Ceva carrier)**.** *A* $\mathsf{Ceva}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (T, S, L, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the triangle-vertex row, recording three carried point histories read through the* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{ProjectiveSpace}^{\uparrow}$ *source certificates;*
2. *S is the side-incidence row, recording the three side lines and the finite incidence witnesses that place each side between the corresponding vertex pair;*
3. *L is the cevian row, recording one displayed cevian from each triangle vertex to the opposite side-incidence row;*
4. *X is the concurrence witness, recording that the three cevian rows meet at a common projective point;*
5. *H records componentwise* hsame *transport for vertices, side incidences, cevian rows, concurrence, and the bilinear incidence tests used by the local classifier;*
6. *C records displayed* Cont *replay for vertex admission, side construction, cevian attachment, concurrence inspection, and incidence readback;*
7. *P is the* Pkg *provenance over* $\mathsf{Ceva}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{ProjectiveSpace}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Ceva}^{\uparrow}}$ *row.*

The carrier contains no coordinate chart, no determinant numerator, no host quotient of points and lines, and no external proof that bypasses the displayed incidence rows.

Lean4 ✓ `BEDC.Derived.CevaUp.TasteGate.CevaTasteGate single carrier alignment`

Theorem 2552.2 (Ceva NameCert obligations)**.** *For the carrier of Definition 3509.1, the local* $\mathsf{NameCert}_{\mathsf{Ceva}^\uparrow}$ *surface consists exactly of carrier admission for* $V = (T, S, L, X, H, C, P, N)$*, triangle-vertex coverage through* T*, side-incidence coverage through* S*, cevian coverage through* L*, concurrence exactness through* X*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. The* $\mathsf{AffineSpace}^\uparrow$*,* $\mathsf{ProjectiveSpace}^\uparrow$*, and* $\mathsf{BilinForm}^\uparrow$ *rows are consumed only as dependency certificates for the displayed incidence packet.*

Lean4 ✓ `BEDC.Derived.CevaUp.CevaCarrier namecert obligations`

Proof. Project the carrier of Definition 3509.1. The vertex row T supplies the three carried point histories, while S records the side incidences they determine inside the affine and projective dependency surfaces.

Read the cevian row L next. Each cevian is attached to one vertex and the opposite side, so the line data cannot be consumed without the side-incidence witnesses already present in S.

The concurrence row X then binds the three cevian rows to one displayed projective point. The bilinear row is used only as the incidence test named by the packet, and the transport and replay rows H and C preserve the same vertices, sides, cevians, and concurrence witness.

Finally P and N source and name the resulting packet. Since the carrier has no further coordinate, these rows exhaust the local $\mathsf{NameCert}_{\mathsf{Ceva}^\uparrow}$ obligations. □

Theorem 2552.3 (Ceva incidence non-escape)**.** *Let*

$$V = (T, S, L, X, H, C, P, N)$$

be an accepted $\mathsf{Ceva}^\uparrow$ *packet. Any consumer that reads a Ceva concurrence conclusion must pass through the triangle-vertex row* T*, the side-incidence row* S*, the cevian row* L*, the concurrence witness* X*, the transport row* H*, the replay row* C*, the provenance row* P*, and the local naming row* N*. No conclusion can be recovered from a coordinate determinant, an unnamed projective-plane quotient, or a bilinear incidence test detached from the displayed side and cevian rows.*

Lean4 ✓ `BEDC.Derived.CevaUp.CevaCarrier incidence nonescape`

Proof. Use the obligation surface of Definition 3509.2. It places triangle vertices, side incidences, cevian rows, and concurrence in one finite packet, with the affine, projective, and bilinear sources appearing only through displayed dependency rows.

If a route omits S, it no longer knows which opposite side each cevian meets. If it omits L, it no longer has the three cevians whose concurrence is being read. If it omits X, it has only side and line data, not a concurrence witness.

The rows H, C, P, N keep the same incidence data under transport, replay, provenance, and local naming. Thus any route bypassing the listed rows exits the $\mathsf{Ceva}^\uparrow$ carrier rather than exporting a BEDC Ceva incidence conclusion. □

Falsifiable prediction: If the concurrence witness row is absent, no accepted $\mathsf{Ceva}^\uparrow$ packet exports a three-cevian concurrence reading.

Independence witness: affinespace, projectivespace, bilinform: the affine-space packet names carried point and translation data without the three side and cevian rows, the projective-space packet names scaling-orbit representatives without the Ceva concurrence witness, and the bilinear-form packet names pairing data without the triangle incidence carrier. None is bijective to $\mathsf{Ceva}^\uparrow$.

Closure status for $\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}$**.**

Constructive story: A $\mathsf{Ceva}^{\uparrow}$ packet is generated from triangle-vertex, side-incidence, cevian, concurrence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}, \text{obligation})$

Scope closed: Definition 3509.1, Definition 3509.2, and Definition 3509.3 bind the finite incidence packet for triangle vertices, side rows, cevian rows, concurrence, transport, replay, provenance, local naming, and the displayed $\mathsf{Ceva}^{\uparrow}$ carrier exclusions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, coordinate determinant proof, a host projective plane, host projective-plane quotient, quotient equality of point-line incidences, analytic area computation, or bridge-checked standard model.

Upgrade path: Scoped closure requires a Lean carrier and obligation package for the same triangle-vertex, side-incidence, cevian, concurrence, hsame, Cont, Pkg, and NameCert route, including carrier admission, incidence-row stability, concurrence exactness, and nonescape for BEDC.Derived.CevaUp.

Chapter 2553

A Concrete Naming Certificate for $\mathsf{BishopCauchyApproximation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyApproximation}^{\uparrow}$ packet names the finite approximation surface by which a regular Cauchy rational source answers a Real-facing precision request. It sits between the regular rational source of chapter 301, the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the finite approximation service of chapter 601. The packet is the L10 human-origin approximation row used by Real and Completion consumers before any quotient stream, selected limit, countable choice, trichotomy, host equality, or ambient completeness principle is available.

Definition 2553.1 (Bishop Cauchy approximation source)**.** *A* $\mathsf{BishopCauchyApproximation}^{\uparrow}$ *source is a finite* BHist *packet*

$$A = (R, n, W, D, V, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational name supplies the Cauchy data.*
2. *n is the unary precision request row. It records the requested approximation depth before any finite window is opened.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at precision n.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same request and window.*
5. *V is the displayed approximation value row read from R through W and checked against D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It can be consumed only after R, n, W, D, V have all been displayed.*
7. *H stores componentwise* hsame *transports for the source, request, window, tolerance, value, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a Real-facing approximation read through R, n, W, D, V and into the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}$ *row.*

The local classifier compares two sources only through the displayed rows $R, n, W, D, V, E, H, C, P, N$. It has no coordinate for a quotient stream, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyApproximationUp.BishopCauchyApproximationTasteGate single carrier alignment`

Theorem 2553.2 (Bishop Cauchy approximation window handoff)**.** *Let*

$$A = (R, n, W, D, V, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyApproximation}^{\uparrow}$ *source. Every* $\mathsf{Real}^{\uparrow}$*-facing approximation read of A passes through, in order, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, the unary precision request n, the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the displayed approximation value V, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff E, with componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support supplied by H, C, P, N. The read exports no quotient stream, selected limit, host equality, countable-choice selector, trichotomy row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the source through Definition 3510.1. The only analytic rows visible to a Real-facing consumer are R, n, W, D, V, E, while H, C, P, N supply transport, replay, provenance, and local naming.

The regular source row is read first. It presents the $\mathsf{RegSeqRat}^{\uparrow}$ Cauchy data as finite packet data, so the consumer has not yet received a Real value, quotient stream, or selected limit.

The precision request and finite window then determine the inspected observation. The row n fixes the requested depth, W opens the corresponding finite $\mathsf{StreamName}^{\uparrow}$ segment, and D records the dyadic tolerance ledger attached to that same finite request.

The displayed value row V is therefore read only after the source, request, window, and tolerance rows are all present. The terminal row E seals the result as Real-facing data, and the support rows H, C, P, N replay exactly that route.

Finally compare with the sibling service of chapter 601. Both packets use the same regular-source, finite-window, dyadic-ledger, and Real-seal order. Since neither route contains a coordinate for quotient streams, selected limits, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no such datum can be exported by the Bishop approximation handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3510.1 and Definition 3510.2 seed the L10 Bishop approximation route from $\mathsf{RegSeqRat}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, host equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, precision-window handoff, dyadic tolerance exactness, RealUp seal nonescape, and finite approximation compatibility for BEDC.Derived.BishopCauchyApproximationUp.

Constructive story: A $\mathsf{BishopCauchyApproximation}^{\uparrow}$ packet is generated from a RegSeqRat source row, unary precision request, StreamName finite window, DyadicRatCore tolerance ledger, displayed approximation value, RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2554

A Concrete Naming Certificate for $\mathsf{KakutaniFixedPoint}^{\uparrow}$

Following Kakutani's fixed point theorem and constructive fixed-point readings, the $\mathsf{KakutaniFixedPoint}^{\uparrow}$ packet records only a finite certificate surface for compact convex domains, convex-valued set maps, approximate selector windows, and fixed-point witness ledgers. It sits beside $\mathsf{Compact}^{\uparrow}$ in chapter 60, $\mathsf{ConvexSet}^{\uparrow}$ in chapter 208, $\mathsf{SchauderFixedPoint}^{\uparrow}$ in chapter 1923, $\mathsf{FixedPointIteration}^{\uparrow}$ in chapter 2383, and $\mathsf{Met}^{\uparrow}$ in chapter 57. The packet is a NameCert surface for the fixed-point obligation; it does not import a host choice selector, a full classical compactness theorem, or ambient real completeness.

Definition 2554.1 (Kakutani fixed point carrier)**.** *A* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, G, V, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the named* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *surfaces.*
2. *G is the set-valued graph row, recording only finite graph windows and their source ledger.*
3. *V is the nonempty compact convex value row for the displayed graph windows.*
4. *S is the approximate selector-window row, listing finite selector approximants and their domain restrictions.*
5. *F is the fixed-point witness ledger, recording approximate self-contact rows and the accepted witness boundary.*
6. *H stores componentwise* hsame *transport for the domain, graph, value, selector, witness, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from domain reads to graph windows, from graph windows to selector windows, and from selector windows to the fixed-point witness ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{KakutaniFixedPoint}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{SchauderFixedPoint}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, G, V, S, F, H, C, P, N$. It has no coordinate for a host choice selector, a full classical compactness principle, an ambient real-completeness oracle, or an unrestricted multivalued map.

Lean4 ✓ `BEDC.Derived.KakutaniFixedPointUp.KakutaniFixedPointTasteGate single carrier alignment`

Theorem 2554.2 (Kakutani fixed point NameCert obligations)**.** *For every accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier $K = (D, G, V, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, compact-convex domain admission through D, finite set-valued graph access through G, nonempty compact convex value admission through V, approximate selector-window admission through S, fixed-point witness ledger admission through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3511.1. The domain, graph, value, selector, and witness coordinates are exactly D, G, V, S, F, so every fixed-point observation enters through those displayed rows.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2554.3 (Kakutani graph-selector nonescape)**.** *Every selector-facing read from an accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier factors through*

$$D, G, V \longmapsto S \longmapsto F$$

together with H, C, P, N. Consequently a consumer may use the displayed $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, *and* $\mathsf{Met}^{\uparrow}$ *interfaces named by the packet, but it receives no host selection function, no unbounded graph oracle, no full classical compactness theorem, and no ambient real-completeness principle.*

Proof. Begin with a selector-facing request. By Definition 3511.1, the request can inspect the compact convex domain row D, the finite set-valued graph row G, and the nonempty compact convex value row V before it reads a selector window.

The selector row S exposes only the displayed approximate selectors and their domain restrictions. The witness ledger F repacks those selector reads as approximate self-contact and fixed-point boundary data.

Finally read H, C, P, N. These rows preserve, replay, source, and name the same displayed data, so none of the excluded host principles can escape through the accepted classifier. □

Theorem 2554.4 (Kakutani convex compact value scope)**.** *The value coordinate V of an accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier is scoped only as a displayed nonempty compact convex value row relative to D and G. It may feed the selector-window route S and the witness ledger F, but it exports no ambient hyperspace compactness theorem, no quotient set-of-points object, no arbitrary selection principle, and no host real-completeness assumption.*

Proof. Use the carrier rows D, G, V from Definition 3511.1. The value row is admitted relative to the displayed domain and graph windows, so its compactness and convexity evidence remains a finite row in the packet.

The only downstream consumers are the selector windows S and the fixed-point witness ledger F. The structural rows H, C, P, N then keep the value evidence tied to the same finite NameCert route rather than an ambient hyperspace or choice construction. □

Theorem 2554.5 (Kakutani Brouwer Schauder handoff)**.** *Every accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *fixed-point read that is later consumed by* $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, *or* $\mathsf{NashEquilibrium}^{\uparrow}$ *factors first through the compact-convex domain row D, the finite set-valued graph row G, the nonempty compact convex value row V, approximate selector windows S, and the fixed-point witness ledger F of Definition 3511.1. The handoff reaches the Brouwer and Schauder fixed-point surfaces only through those displayed rows together with H, C, P, N; it exports no host choice selector, unrestricted hyperspace compactness theorem, selected fixed point, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or equilibrium/common-fixed-point shortcut.*

Proof. Begin with the carrier and NameCert obligations of Definition 3511.1 and Definition 3511.2. These fix the only admissible rows for the domain, graph, values, selectors, witness ledger, transport, replay, provenance, and local naming.

Next apply Definition 3511.3. A Brouwer or Schauder consumer can read selector data only after D, G, V have been displayed, and the selector window can be repacked into F only as finite approximate self-contact data.

Finally use Definition 3511.4. The value row remains a displayed nonempty compact convex row relative to the graph and domain. Therefore downstream equilibrium or common-fixed-point consumers receive the Brouwer/Schauder handoff through the finite Kakutani route, with no host selector, hyperspace compactness oracle, chosen fixed point, or ambient real-completeness principle. □

Falsifiable prediction: Every $\mathsf{KakutaniFixedPoint}^{\uparrow}$ fixed-point read factors through a compact convex domain row, finite set-valued graph row, nonempty compact convex value row, approximate selector windows, fixed-point witness ledger, transport, continuation, provenance, and local naming.

Independence witness: compact, convexset, schauder-fixed-point, fixed-point-iteration, metric-space: $\mathsf{KakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact convex domain, set-valued graph, nonempty compact convex values, approximate selector windows, fixed-point witness ledger, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{KakutaniFixedPoint}^{\uparrow}$ packet is generated from finite rows for a compact convex domain, set-valued graph, nonempty compact convex values, approximate selector windows, a fixed-point witness ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3511.1, Definition 3511.2, Definition 3511.3, Definition 3511.4, and Definition 3511.5 bind the finite Kakutani fixed-point certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, full classical compactness, ambient real completeness, unrestricted hyperspace compactness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, graph-selector nonescape, convex-compact value scope, and TasteGate field faithfulness for BEDC.Derived.KakutaniFixedPointUp.

Chapter 2555

A Concrete Naming Certificate for $\mathsf{RatCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of the real line as regular Cauchy data over the rationals, the $\mathsf{RatCauchyCompletion}^{\uparrow}$ packet records the finite BEDC interface by which rational rows are read as regular rational Cauchy names and then presented to the real-name seal. It consumes the rational source surface of chapter 37, the Cauchy-modulus tail budget of chapter 292, the finite schedule surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet gives Phase Real and Completion consumers a named rational-Cauchy handoff without quotienting Cauchy names, selecting limits, invoking countable choice, or asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2555.1 (RatCauchyCompletion carrier)**.** *A* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, M, S, Q, D, E, H, K, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Rat}^{\uparrow}$ *source row, recording the rational entries that seed the completion-facing stream.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row, recording the displayed budget by which two sufficiently late rational observations are forced inside the requested tolerance.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite windows through which later consumers request rational approximants.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence visible to the completion corridor.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic budgets, endpoint comparisons, and finite error rows used to check the regularity bounds.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording only the handoff boundary at which the regular rational name may be read as a real-facing name.*
7. *H records componentwise* hsame *transport for the rational source, tail budget, schedule, regular readback, dyadic ledger, seal, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *routes from rational source rows through tail budgets, schedules, regular readback, dyadic tolerance checks, and the real-seal boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RatCauchyCompletion}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, S, Q, D, E, H, K, P, N$. It has no coordinate for quotient-real equality, selected limits, arbitrary Cauchy filters, countable-choice schedules, host completeness, or a theorem that every $\mathsf{Real}^{\uparrow}$ *request is complete.*

Theorem 2555.2 (RatCauchyCompletion NameCert obligations)**.** *For every accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier*

$$C = (R, M, S, Q, D, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier exposure for the rational source row R, Cauchy-modulus tail row M, stream schedule row S, regular rational readback row Q, dyadic tolerance ledger D, real-seal boundary E, componentwise transport row H, continuation row K, package provenance row P, and local naming row N.*

Lean4 ✓ `BEDC.Derived.RatCauchyCompletionUp.TasteGate.RatCauchyCompletionCarrier namecert obligations`

Proof. Carrier exposure is projection from Definition 3512.1. Each row named in the statement is one coordinate of the finite packet, and the definition lists no hidden coordinate for completion beyond those rows.

The rational source and Cauchy-modulus obligations are R and M. The stream and regular-readback obligations are S and Q. The dyadic ledger and real-facing seal obligations are D and E.

Transport, replay, provenance, and local naming are supplied by H, K, P, N. Repacking these coordinates yields the displayed $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ surface, while any claim about quotient equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness remains outside the packet. □

Theorem 2555.3 (RatCauchyCompletion Real seal boundary)**.** *An accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier may hand its regular rational readback row Q and dyadic tolerance ledger D to the* $\mathsf{Real}^{\uparrow}$ *seal row E only through the displayed continuation row K. The handoff preserves the rational-Cauchy interface for Phase Real and Completion consumers, but it does not assert that* $\mathsf{Real}^{\uparrow}$ *is complete, does not choose a quotient representative, and does not manufacture a limit outside the rows $R, M, S, Q, D, E, H, K, P, N$.*

Proof. The only real-facing coordinate of the carrier is E, and the only route into that coordinate is the displayed continuation row K. Thus a consumer reaches the seal boundary by reading the rational source, tail budget, stream schedule, regular rational readback, and dyadic ledger through the packet's finite route.

The carrier definition also lists the negative boundary: no coordinate stores a selected limit, a quotient representative, a countable-choice schedule, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. Therefore the seal row is a handoff boundary for existing regular rational data, not a proof of completeness of the target interface. □

Theorem 2555.4 (RatCauchyCompletion seal nonescape)**.** *Every accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed rational source row, Cauchy-modulus tail row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise transport, displayed replay, provenance, and local NameCert rows. The packet exports no quotient Cauchy name, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3512.1. Its only Real-facing coordinate is E, and every coordinate before E is one of R, M, S, Q, D together with the structural rows H, K, P, N.

Apply the obligation theorem Definition 3512.2. It exposes the rational source, Cauchy modulus, stream schedule, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming as the whole local NameCert surface.

Now use the handoff theorem Definition 3512.3. The seal row is reachable only through the displayed continuation row after the rational-Cauchy and dyadic rows have been presented.

Therefore a seal read cannot escape to quotient Cauchy names, selected limits, countable-choice schedules, or ambient Real completeness. None of those objects is a carrier coordinate or a local NameCert obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RatCauchyCompletion}^{\uparrow}$ packet is generated from $\mathsf{Rat}^{\uparrow}$ source rows, $\mathsf{CauchyModulus}^{\uparrow}$ tail data, $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3512.1, Definition 3512.2, Definition 3512.3, and Definition 3512.4 bind the carrier, local NameCert obligations, Real-facing seal boundary, and seal nonescape theorem for the rational-Cauchy completion interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove $\mathsf{Real}^{\uparrow}$ completeness, quotient Cauchy equality, selected limits, arbitrary Cauchy-filter completion, countable choice, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the NameCert obligation surface, the Real seal boundary, and the rejection of completeness claims outside the displayed rows.

Chapter 2556

A Concrete Naming Certificate for DyadicDensityRealEmbeddingComparison$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicDensityRealEmbeddingComparison$^{\uparrow}$ packet compares the dyadic-density route of chapter 1484 with the constant dyadic real-name route of chapter 467. It keeps the comparison over the finite rows already exposed by DyadicRatCore$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, DyadicEmbedding$^{\uparrow}$, RealDyadicEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$. The packet is a naming certificate for route agreement at displayed windows, not a construction of host real equality, selected dense points, trichotomy, or completeness of Real$^{\uparrow}$.

Definition 2556.1 (Dyadic-density real embedding comparison carrier)**.** *A* DyadicDensityRealEmbeddingCompariso *carrier is a finite* BHist *packet*

$$K = (D, E, E', S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicDensityReal$^{\uparrow}$ *density-witness row whose dyadic source is read through the* DyadicRatCore$^{\uparrow}$ *tolerance ledger.*
2. *E is the* DyadicEmbedding$^{\uparrow}$ *constant-name route from the same dyadic source into a stream, a regular rational readback, and a real seal.*
3. *E' is the* RealDyadicEmbedding$^{\uparrow}$ *comparison row that records the Real-facing read of the embedded dyadic name at the displayed window.*
4. *S is the shared* StreamName$^{\uparrow}$ *finite-window row opened by both the density witness and the embedding route.*
5. *R is the shared* RegSeqRat$^{\uparrow}$ *readback row attached to that window.*
6. *Q is the* Real$^{\uparrow}$ *seal row consumed by the comparison.*
7. *H stores componentwise* hsame *transport for the density row, embedding rows, stream window, regular readback, real seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from the dyadic source to the density witness, from the same source through the embedding route, and from the common window-readback pair to the* Real$^{\uparrow}$ *seal.*
9. *P stores* Pkg *provenance for* DyadicRatCore$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, DyadicEmbedding$^{\uparrow}$, RealDyadicEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$ *row.*
The carrier contains no coordinate for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ BEDC.Derived.DyadicDensityRealEmbeddingComparisonUp.DyadicDensityRealEmbeddingComparisonTasteGa
single carrier alignment

Theorem 2556.2 (Dyadic-density real embedding comparison NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *carrier* $K = (D, E, E', S, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier admission for the density, embedding, comparison, window, readback, real-seal, transport, continuation, provenance, and local naming rows;*
2. *dyadic-source agreement, stating that the density row D and the embedding row E read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row;*
3. *window agreement, stating that the comparison opens the same* $\mathsf{StreamName}^{\uparrow}$ *row S for the density read and for the embedded constant name;*
4. *readback agreement, stating that the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *row R is the only rational readback passed to the* $\mathsf{Real}^{\uparrow}$ *seal Q;*
5. *Real-facing comparison, stating that the* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *row E' compares only the displayed density route with the displayed embedding route at the common finite window;*
6. *non-escape, stating that host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are absent from the carrier.*

Lean4 ✓ BEDC.Derived.DyadicDensityRealEmbeddingComparisonUp.DyadicDensityRealEmbeddingComparisonCarrier
namecert obligations

Proof. Project the accepted carrier. The tuple lists exactly the density row, embedding rows, shared stream window, shared regular readback, real seal, transport, continuation, provenance, and local naming rows available to the classifier.

Dyadic-source agreement is read from the common $\mathsf{DyadicRatCore}^{\uparrow}$ source named by the density and embedding routes. The carrier has one source ledger for the comparison, so the embedded constant name cannot use a separate dyadic point.

Window agreement and readback agreement follow from the shared rows S and R. Both routes must open the same finite $\mathsf{StreamName}^{\uparrow}$ window and pass the same $\mathsf{RegSeqRat}^{\uparrow}$ readback before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

The Real-facing comparison is therefore only the row E'. It records agreement of the two displayed routes at the common finite read, while the transport and continuation rows replay the same packet under componentwise hsame.

Finally inspect the carrier inventory. No row names host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, so the six rows exhaust the local NameCert surface. □

Theorem 2556.3 (Dyadic-density real embedding comparison route)**.** *Every accepted* DyadicDensityRealEmbedding
packet compares the density witness route and the constant dyadic embedding route only through the finite spine

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow (\mathsf{DyadicDensityReal}^{\uparrow}, \mathsf{DyadicEmbedding}^{\uparrow}, \mathsf{RealDyadicEmbedding}^{\uparrow}) \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSe}$$

The comparison is a route equality of displayed certificate rows at a common finite window; it does not import host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Start with an accepted packet $K = (D, E, E', S, R, Q, H, C, P, N)$. The density route begins with the $\mathsf{DyadicDensityReal}^{\uparrow}$ row D, and the constant-name route begins with the $\mathsf{DyadicEmbedding}^{\uparrow}$ row E over the same dyadic source.

The comparison row E' is downstream of those two rows. It does not add a new source; it only records the Real-facing comparison after the shared $\mathsf{StreamName}^{\uparrow}$ window S and the shared $\mathsf{RegSeqRat}^{\uparrow}$ readback R have been displayed.

The real seal Q is therefore reached through the same finite read on both sides. Componentwise transport H, continuation replay C, provenance P, and the local name N preserve the packet as a finite BEDC certificate.

Since the carrier provides no row for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, no consumer can replace this finite comparison by an external real-equality or completeness argument. □

Theorem 2556.4 (Dyadic-density real embedding comparison RegSeqRat modulus use)**.** *Every accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *packet reads the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *row through the same finite modulus window before comparing the density witness route with the constant dyadic embedding route. In particular, for* $K = (D, E, E', S, R, Q, H, C, P, N)$*, the window row* S *and readback row* R *are consumed before the comparison row* E' *can export agreement to the* $\mathsf{Real}^{\uparrow}$ *seal* Q*.*

Proof. Project the accepted carrier as $K = (D, E, E', S, R, Q, H, C, P, N)$. The carrier has one shared $\mathsf{StreamName}^{\uparrow}$ row S and one shared $\mathsf{RegSeqRat}^{\uparrow}$ readback row R for the density witness route and the constant dyadic embedding route.

Use the shared-window and readback agreement rows from Definition 3513.2. They force both sides of the comparison to open the same finite window and to pass the same regular rational readback before the Real-facing comparison row is consumed.

Apply Definition 3513.3 to identify the comparison spine ending in the $\mathsf{Real}^{\uparrow}$ seal Q. Since the carrier contains no row for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, the shared $\mathsf{RegSeqRat}^{\uparrow}$ window cannot be replaced by any external Real-side principle. □

Theorem 2556.5 (Dyadic-density real embedding comparison non-escape)**.** *The dependency ledger of an accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *carrier is exhausted by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicDensityReal}^{\uparrow}$*,* $\mathsf{DyadicEmbedding}^{\uparrow}$*,* $\mathsf{RealDyadicEmbedding}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No accepted comparison consumer may add host real equality, a selected dense point, trichotomy, quotient stream equality, arbitrary limit choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as an implicit dependency.*

Proof. Project the package row P and the local naming row N. They list exactly the finite dependency rows displayed in Definition 3513.1.

Apply Definition 3513.2 for carrier admission, source agreement, window agreement, readback agreement, Real-facing comparison, and non-escape. Then use Definition 3513.3 to identify the only comparison spine consumed by downstream Real-facing readers.

No coordinate remains that could carry an external equality test, a chosen dense point, a trichotomy decision, a quotient-stream equality proof, an arbitrary limit choice, or a completeness

principle. Each such object would therefore be an off-packet dependency, not a row of the accepted comparison certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ packet is generated from a DyadicDensityReal density row, a DyadicEmbedding constant-name route, a RealDyadicEmbedding comparison row, shared StreamName and RegSeqRat reads, a Real seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3513.1, Definition 3513.2, Definition 3513.3, and Definition 3513.5 seed the finite comparison packet over dyadic density and dyadic real embedding rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, shared-window agreement, shared-readback agreement, Real-facing comparison, and non-escape for the same finite comparison packet.

Chapter 2557

A Concrete Naming Certificate for $\mathsf{CauchyInterleavingComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ packet records the finite comparison surface for two interleaved regular-Cauchy names before a completion-facing consumer reads either name as a real-facing object. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the convergence-modulus route of chapter 1438, the sequence-limit handoff of chapter 1413, and the terminal real seal of chapter 38. The packet supplies a finite comparison certificate for completion consumers; it is not a quotient of Cauchy names, not a selected representative, and not an ambient completeness principle.

Definition 2557.1 (Cauchy interleaving comparison carrier)**.** *A* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, E, O, D, T, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the paired regular-Cauchy source row, storing the two regular rational names whose observations are compared.*
2. *E is the even interleaving window row, exposing the finite observations assigned to the first source.*
3. *O is the odd interleaving window row, exposing the finite observations assigned to the second source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic comparison budget and endpoint error rows.*
5. *T is the shared tail threshold, read before either interleaving window may be used as a comparison witness.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family that displays the finite window requests routed through T.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback comparison row, reconstructing the regular rational comparison from E, O, D, T, W.*

8. *H records componentwise* hsame *transport for the source, window, tolerance, threshold, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each completion-facing comparison request through $S, E, O, D, T, W, Q, H, P$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of Cauchy names, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterleavingComparisonUp.CauchyInterleavingComparisonTasteGate`
`single carrier alignment`

Theorem 2557.2 (Cauchy interleaving comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *carrier $K = (S, E, O, D, T, W, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComp}}$ *surface consists exactly of paired regular-Cauchy source admission, even-window exposure, odd-window exposure, dyadic tolerance exposure, shared tail-threshold exposure, StreamName finite-window exposure, RegSeqRat readback comparison, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 3514.1. Its analytic rows are the paired source S, the even and odd windows E, O, the dyadic ledger D, the shared threshold T, the finite window family W, and the readback comparison row Q.

The NameCert obligations read those rows in the displayed order. The even and odd windows are not independent representatives; they are the two finite faces of the same paired source, controlled by the same dyadic ledger and the same threshold row.

The structural rows H, C, P, N preserve, replay, source, and name the packet. Since the carrier lists no quotient equality, chosen representative, choice selector, trichotomy row, or ambient completeness row, the local certificate surface is exhausted by the displayed finite comparison data. □

Theorem 2557.3 (Cauchy interleaving comparison tail lock)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *packet must read the shared threshold row T before inspecting either the even window E or the odd window O. The comparison route is*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto E, O \longmapsto Q \longmapsto H, C, P, N,$$

so a downstream $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer receives a single finite tail lock rather than two selected tail representatives.*

Proof. Begin with the NameCert obligations. They place D and T before the window family and before the even–odd observations may be read as comparison evidence.

The $\mathsf{StreamName}^{\uparrow}$ family W then displays the finite requests controlled by the shared threshold. Since both interleaving windows are read through the same T, W pair, a consumer cannot replace the comparison by separate chosen tails for the two sources.

Finally Q repacks the displayed windows as a $\mathsf{RegSeqRat}^{\uparrow}$ readback comparison, and the structural rows preserve that route for $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. Quotient equality, countable choice, trichotomy, selected representatives, and ambient completeness are absent from the carrier and from the local provenance. □

Falsifiable prediction: A $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ consumer that reads an even or odd interleaving window without first reading the shared threshold T refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, cauchyconvergencemodulus, realsequencelimit, real: no listed sibling supplies the whole packet $K = (S, E, O, D, T, W, Q, H, C, P, N)$, because each sibling contributes only tolerance, finite-window, readback, convergence-modulus, limit-handoff, or seal data while this chapter binds the two interleaved windows to one shared comparison threshold.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ packet is generated from a paired regular-Cauchy source, even and odd interleaving windows, a DyadicRatCore tolerance ledger, a shared tail threshold, StreamName finite windows, RegSeqRat readback comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3514.1, Definition 3514.2, and Definition 3514.3 seed the finite comparison route for interleaved regular-Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, even–odd window exposure, shared threshold locking, RegSeqRat readback comparison, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterleavingComparisonUp.

Chapter 2558

A Concrete Naming Certificate for RegularCauchySubsequenceStability$^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the RegularCauchySubsequenceStability$^{\uparrow}$ packet records the finite certificate surface for the standard fact that a monotone cofinal subsequence of a regular Cauchy rational name remains regular and presents the same real-name boundary. It consumes the subsequence interface of chapter 381, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal seal surface of chapter 38. The packet is a finite BEDC handoff for Phase Real and Completion consumers; it does not choose a subsequence by countable choice, quotient streams, or invoke ambient completeness.

Definition 2558.1 (RegularCauchySubsequenceStability carrier)**.** *A* RegularCauchySubsequenceStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy packet, carrying the accepted* RegSeqRat$^{\uparrow}$ *readback rows and the source finite stream observations.*
2. *T is a monotone-cofinal schedule row, recording the unary reindexing route by which each selected subsequence window is reached from an inspected source window.*
3. *W records the finite* StreamName$^{\uparrow}$ *source and subsequence windows compared by the packet.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, carrying dyadic radii, endpoint comparisons, and the transported regularity budgets.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the subsequence window after transport along T.*
6. *E is the* Real$^{\uparrow}$ *seal boundary exposed only after the source, schedule, window, dyadic, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the source packet, schedule, windows, tolerance ledger, readback row, seal boundary, continuation routes, provenance, and local certificate.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records the* Pkg *provenance over* RegularCauchySubsequenceStability$^{\uparrow}$, RegularCauchySubsequence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by $S, T, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a choice-selected infinite subsequence, host sequence equality, host real equality, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityTasteGate`
`single carrier alignment`
Lean4 (definition) `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarri`

Theorem 2558.2 (RegularCauchySubsequenceStability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ *carrier*

$$G = (S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ *surface consists exactly of the source regular Cauchy rows S, the monotone-cofinal schedule T, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *subsequence readback R, the* $\mathsf{Real}^{\uparrow}$ *seal boundary E, the componentwise transport row H, the continuation replay row C, the package provenance row P, and the local naming row N.*

Proof. Carrier exposure is projection from Definition 3515.1. Each row in the statement is one displayed coordinate of the finite packet, and the definition lists no hidden coordinate for an externally selected subsequence or completion object.

The source and schedule obligations are S and T. The finite-window obligation is W, while the dyadic tolerance and regular rational readback obligations are D and R. These are the rows that express the usual cofinal-subsequence regularity argument inside the BEDC packet.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections yields the displayed $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ surface, with no quotient stream, host equality, or ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarrier`
`namecert obligations`

Theorem 2558.3 (RegularCauchySubsequenceStability Real seal boundary)**.** *An accepted* $\mathsf{RegularCauchySubseque}$ *carrier may expose its* $\mathsf{Real}^{\uparrow}$ *seal row E only after the source regular Cauchy data S, the monotone-cofinal schedule T, the finite stream windows W, the transported dyadic tolerance ledger D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R have been displayed through H and C. Thus cofinal subsequence transport preserves the real-name seal surface as finite BEDC data and does not export quotient streams, a choice-selected subsequence, host sequence equality, or ambient real completeness.*

Proof. Start with an accepted carrier. The seal row E is not a source coordinate; it is reached after the source, schedule, window, dyadic, readback, transport, continuation, provenance, and naming rows have been projected.

Apply the NameCert obligation surface of Definition 3515.2. It identifies the monotone-cofinal schedule, the inspected and selected finite windows, the transported dyadic tolerance ledger, and the regular rational readback as the only data available before the seal boundary.

The ordinary constructive subsequence argument is therefore recorded as carrier projection followed by cofinal schedule transport and finite-window replay. Since the classifier has no coordinate for quotient streams, choice-selected subsequences, host sequence equality, host real equality, or a completeness principle, none of those data can escape through the $\mathsf{Real}^{\uparrow}$ seal row. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarrier`
`real seal boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ packet is generated from source regular Cauchy rows, a monotone-cofinal schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, transported $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3515.1, Definition 3515.2, and Definition 3515.3 bind the carrier, local NameCert obligations, and Real-facing seal boundary for cofinal regular Cauchy subsequence stability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove a public standard bridge, quotient stream equality, host sequence equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the NameCert obligation surface, the Real seal boundary, and the nonescape boundary for data outside the displayed rows.

DNA hash: `aggregate=sha256:5f957223fdebb99cd0a4300c035bf3311aafd9888c8f99810e0304aa1e677f18;statement=sha256:a7`

Chapter 2559

A Concrete Naming Certificate for $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$

Following Specker's constructive counterexample tradition and Bishop–Bridges constructive analysis, the $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ packet isolates the reusable nonlocated-supremum obstruction behind monotone bounded rational streams. It sits after $\mathsf{SpeckerSequence}^{\uparrow}$ in chapter 1406 and reads only the finite windows supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{LocatedCut}^{\uparrow}$ in chapter 126, chapter 301, chapter 291, chapter 38, and chapter 404. The packet names the obstruction surface; it does not supply Real completeness, a located supremum, trichotomy, quotient stream equality, countable choice, or selected limits.

Definition 2559.1 (Specker sequence boundary carrier)**.** *A* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, M, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SpeckerSequence}^{\uparrow}$ *source row naming the monotone bounded rational stream and its nonlocated-supremum boundary.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite source windows inspected by the boundary packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound ledger read from the same selected windows.*
4. *M is the monotone and bounded obstruction ledger recording the displayed comparison and bound data that can be read before any real-facing seal.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows certified by S, W, D, M.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{LocatedCut}^{\uparrow}$ *refusal row; it records that the displayed windows may be sealed as finite real readback data, while a located-cut or supremum witness remains external.*
7. *H stores componentwise* hsame *transport for the source, schedule, dyadic ledger, obstruction ledger, regular readback, seal-or-refusal row, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from S, W into D, from D into M, from M into R, and from R into the seal-or-refusal row L.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, D, M, R, L, H, C, P, N$. It has no coordinate for a located supremum, a selected limiting real, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SpeckerSequenceBoundaryUp.SpeckerSequenceBoundaryTasteGate single carrier alignment`

Theorem 2559.2 (Specker sequence boundary NameCert obligations)**.** *For every accepted* SpeckerSequenceBounda *carrier $B = (S, W, D, M, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$ *surface consists exactly of source admission through S, finite schedule selection through W, dyadic tolerance and bound admission through D, monotone bounded obstruction admission through M, regular rational readback through R, the* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{LocatedCut}^{\uparrow}$ *refusal row L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3516.1. The analytic rows are exactly the Specker source, stream schedule, dyadic ledger, monotone bounded obstruction ledger, regular readback, and seal-or-refusal row.

The structural rows H, C, P, N transport, replay, source, and name those same coordinates. Since the carrier contains no further analytic field, the listed rows exhaust the local NameCert obligations.

The missing fields are also part of the boundary. A located supremum, quotient stream equality, countable-choice selector, trichotomy branch, or selected limit would require an additional coordinate, and no such coordinate is present. □

Theorem 2559.3 (Specker sequence boundary Cauchy readback)**.** *Every Cauchy-facing read from an accepted* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier factors through*

$$S, W \longmapsto D \longmapsto M \longmapsto R \longmapsto L$$

together with H, C, P, N. The row R exports only the $\mathsf{RegSeqRat}^{\uparrow}$ *window readback certified by the displayed schedule and dyadic ledger, and the row L exports only a finite* $\mathsf{Real}^{\uparrow}$ *seal or a* $\mathsf{LocatedCut}^{\uparrow}$ *refusal boundary.*

Proof. Start with the source route. The $\mathsf{SpeckerSequence}^{\uparrow}$ row S and the $\mathsf{StreamName}^{\uparrow}$ schedule W determine which finite rational windows are inspected.

Read the dyadic and obstruction ledgers next. The row D supplies the tolerance and bound data for those windows, while M records the monotone bounded obstruction information available at the same finite stages.

The regular readback row R is downstream of those data, and the seal-or-refusal row L is downstream of R. Transport, replay, provenance, and local naming through H, C, P, N preserve this route, so no located supremum or selected limiting real is produced by the Cauchy readback. □

Theorem 2559.4 (Specker sequence boundary nonescape)**.** *Every consumer of an accepted* SpeckerSequenceBounda *carrier receives only the finite rows $S, W, D, M, R, L, H, C, P, N$. Consequently the packet may reuse the* $\mathsf{SpeckerSequence}^{\uparrow}$ *nonlocated-supremum obstruction and its* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{LocatedCut}^{\uparrow}$ *interfaces, but it exports no* $\mathsf{Real}^{\uparrow}$ *completeness principle, located supremum, trichotomy, quotient stream equality, countable-choice selector, or selected limit.*

Proof. Use Definition 3516.2 to identify the complete public surface of the accepted packet. All analytic reads pass through the six displayed rows S, W, D, M, R, L.

Use Definition 3516.3 to order those rows. The regular readback and seal-or-refusal row appear only after finite windows, dyadic data, and monotone bounded obstruction data have been displayed.

Finally read the structural rows. Componentwise transport, displayed replay, provenance, and local naming do not add analytic coordinates, so none of the excluded completeness, choice, equality, order-decision, or selected-limit principles can escape from the packet. □

Theorem 2559.5 (Specker sequence boundary LPO separation)**.** *Every consumer reading an accepted* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier* $B = (S, W, D, M, R, L, H, C, P, N)$ *through the displayed rows* $S, W, D, M, R, L, H, C, P, N$ *receives only the Specker source, finite schedule, dyadic tolerance ledger, monotone bounded obstruction ledger, regular rational readback, seal-or-refusal boundary, transport, continuation, provenance, and local naming data. Such a consumer cannot extract* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$, $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, *countable choice, trichotomy, or a located supremum unless an external row supplies that principle.*

Proof. Project the carrier of Definition 3516.1. The only analytic rows available to the consumer are S, W, D, M, R, L, and the only structural rows are H, C, P, N.

Apply the Cauchy readback theorem Definition 3516.3. It orders the read through the Specker source, finite schedule, dyadic ledger, monotone bounded obstruction ledger, regular readback, and seal-or-refusal row before any Real-facing conclusion is public.

Now apply nonescape from Definition 3516.4. The carrier has no coordinate for located supremum construction, trichotomy, quotient stream equality, countable choice, selected limits, or ambient Real completeness.

The neighbouring omniscience packets of chapter 5945 and chapter 1977 are therefore only comparison siblings. Without an external row naming one of those principles, the Specker boundary exports the obstruction route and not LPO, LLPO, countable choice, trichotomy, or a located supremum. □

Falsifiable prediction: Every $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ Cauchy-facing read factors through a SpeckerSequence source row, StreamName schedule, DyadicRatCore tolerance ledger, monotone bounded obstruction ledger, RegSeqRat readback, Real seal or LocatedCut refusal row, transport, continuation, provenance, and local naming.

Independence witness: speckersequence, streamname, regseqrat, dyadicratcore, real, located-cut: $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nonlocated-supremum obstruction route, regular readback, seal-or-refusal boundary, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ packet is generated from a $\mathsf{SpeckerSequence}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and bound rows, a monotone bounded obstruction ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{LocatedCut}^{\uparrow}$ refusal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3516.1, Definition 3516.2, Definition 3516.3, Definition 3516.4, and Definition 3516.5 bind the finite nonlocated-supremum obstruction certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, located supremum construction, trichotomy, quotient stream equality, countable choice, selected limits, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Cauchy readback, nonescape, and TasteGate field faithfulness for BEDC.Derived.SpeckerSequenceBoundaryUp.

Chapter 2560

A Concrete Naming Certificate for $\mathsf{BishopOrderCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet records the order-completion face of the real-number construction as finite BEDC NameCert data. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the comparison boundary of chapter 1730, the lower and upper cut rows of chapter 1446 and chapter 2087, and the terminal real-name seal of chapter 38. The packet names a constructive order-completion interface; it is not a quotient-cut construction, selected supremum principle, trichotomy oracle, host real equality test, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2560.1 (Bishop order-completion carrier)**.** *A* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, E, B, L, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison-budget ledger used by all lower, upper, and real-seal requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at those dyadic requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R have been displayed.*
5. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison row, read through the located-real comparison surface and tied to the same windows and dyadic budgets.*
6. *L is the LowerReal-style located lower-cut ledger, recording only the displayed inhabited lower observations and downward-use rows available to the packet.*
7. *U is the UpperReal-style located upper-cut ledger, recording only the displayed upper observations and upward-use rows available to the packet.*
8. *H records componentwise* hsame *transport for the tolerance, window, readback, Real seal, comparison, lower-cut, upper-cut, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each order-completion request through $D, S, R, E, B, L, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopOrderCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $D, S, R, E, B, L, U, H, C, P, N$. It has no coordinate for quotient cuts, host supremum objects, selected representatives, trichotomy, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopOrderCompletionUp.TasteGate.BishopOrderCompletionTasteGate single carrier alignment`

Theorem 2560.2 (Bishop order-completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier $O = (D, S, R, E, B, L, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, finite StreamName window exposure, RegSeqRat readback, Real seal handoff, BishopRealComparison exposure, LowerReal-style lower-cut exposure, UpperReal-style upper-cut exposure, componentwise transport, displayed continuation replay, package provenance, and local naming. Every order-completion consumer factors through*

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto B \longmapsto (L, U) \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3517.1. Its real-facing rows are the dyadic tolerance ledger, finite stream windows, regular rational readback, and terminal Real seal.

Read the order rows after the same real-facing route. The comparison row B uses the displayed windows and dyadic budgets, while L and U record the lower and upper located-cut observations available to the packet.

The structural rows then preserve this ordered route. The hsame, Cont, Pkg, and local NameCert rows transport, replay, source, and name the route, and the refused quotient, host-supremum, selected, trichotomy, host-equality, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 2560.3 (Bishop order-completion located-cut handoff)**.** *Every located lower-or-upper cut read exported by an accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier is downstream of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison row. The handoff exposes the LowerReal-style and UpperReal-style ledgers only as the finite packet $D, S, R, E, B, L, U, H, C, P, N$.*

Proof. Start with the NameCert obligations. They place D, S, R, E before any order-facing row can be consumed.

Next read the comparison boundary. The row B ties lower and upper observations to the same finite windows and dyadic budgets, so the lower and upper ledgers cannot be inspected as independent host cuts.

Finally project L and U. These rows are named only inside the carrier and are preserved by H, C, P, N, so the handoff contains no quotient-cut equality, selected representative, trichotomy branch, host supremum, or ambient completeness coordinate. □

Theorem 2560.4 (Bishop order-completion Cauchy-cut comparison)**.** *For every accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier, the comparison between regular-Cauchy real data and located lower–upper cut data is a single finite route through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BishopRealComparison}^{\uparrow} \longmapsto (\mathsf{LowerReal}^{\uparrow}, \mathsf{UpperR}$$

Thus a Cauchy-facing consumer and a cut-facing consumer read the same packet-local rows $D, S, R, E, B, L, U, H, C$ rather than two unrelated completion principles.

Proof. Read the Cauchy-facing side first. The rows D, S, R, E display the tolerance, window, readback, and Real seal route used by regular-Cauchy consumers.

Read the cut-facing side through the comparison row. The row B is downstream of the same route and supplies the finite comparison request before L and U can expose lower and upper located observations.

The two faces are therefore synchronized by the structural rows. Transport, continuation, provenance, and local naming preserve one packet-local route, while quotient-cut completion, selected supremum, host real equality, trichotomy, countable choice, and ambient Real completeness remain outside the carrier. □

Falsifiable prediction: A claimed $\mathsf{BishopOrderCompletion}^{\uparrow}$ consumer that reaches a lower-cut or upper-cut row without the displayed DyadicRatCore, StreamName, RegSeqRat, Real, and BishopRealComparison rows refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishoprealcomparison, lowerreal, upperreal: no listed sibling supplies the whole $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet, because each sibling contributes only tolerance, window, readback, seal, comparison, or one-sided cut data while this chapter binds all of them with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, BishopRealComparison row, LowerReal-style and UpperReal-style located cut rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3517.1, Definition 3517.2, Definition 3517.3, and Definition 3517.4 seed the finite order-completion interface joining regular-Cauchy real data to located lower and upper cut rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient cut completion, selected suprema, trichotomy, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, regular readback, RealUp seal handoff, comparison-row stability, located cut handoff, Cauchy-cut comparison, transport, replay, provenance, and local naming for BEDC.Derived.BishopOrderCompletionUp.

Chapter 2561

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ packet records the finite precision-to-tail schedule that lets a Cauchy name serve L10 Real and Completion consumers without importing ambient completeness. It sits over the generic modulus schedule of chapter 292, the fast threshold discipline of chapter 1781, the request-to-tail realizer of chapter 1918, the finite-window interface of chapter 126, the regular rational route of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet supplies one displayed schedule row before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads a seal.

Definition 2561.1 (Effective Cauchy modulus carrier)**.** *An* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, M, T, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose observations are opened only through finite windows.*
2. *D is the dyadic precision ledger consumed by the tail estimates and regular rational readbacks.*
3. *M is the effective precision-to-tail modulus row, assigning each displayed precision request a finite tail index.*
4. *T is the tail-index comparison row showing that the selected index is stable under stronger displayed precision requests.*
5. *W is the finite tail-window row opened from S at the index selected by M.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W under the dyadic ledger D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row, exposed only after S, D, M, T, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for the stream, dyadic ledger, effective modulus, tail comparison, window, readback, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a precision request through $S, D, M, T, W, Q, E_H, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, M, T, W, Q, E, H, C, P, N$. It has no coordinate for quotient limits, countable choice, host convergence, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2561.2 (Effective Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *carrier*

$$X = (S, D, M, T, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the stream source, dyadic precision ledger, effective modulus row, tail comparison row, finite window, regular readback, Real handoff, transport, replay, provenance, and local naming rows;*
2. *schedule classification, requiring M to be the only precision-to-tail row read by a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer;*
3. *monotone tail-index stability, requiring T to certify that stronger displayed precision requests never require a weaker tail boundary than the row already exposed by M;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal compatibility, requiring W and Q to be read before the handoff row E exposes a terminal Real seal;*
5. *ledger refusal for quotient limits, countable choice, host convergence, trichotomy, selected representatives, and ambient completeness as local certificate data.*

No effective Cauchy modulus obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are exactly the stream source, dyadic ledger, effective modulus, tail comparison, finite window, regular readback, and Real handoff rows.

The schedule classifier is the row M. Since every consumer reaches a tail window only after reading M, no hidden convergence selector or external completeness principle can choose the boundary.

The monotonicity information is stored in T. It relates stronger displayed precision requests to tail indices already chosen by M, so transport through H and replay through C preserve the same schedule.

Finally, the readback and seal are downstream of the finite window. The rows Q and E can expose regular rational and Real-facing data only after S, D, M, T, W have been displayed, and the refused objects have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.EffectiveCauchyModulusUp.EffectiveCauchyModulusCarrier namecert obligations`

Theorem 2561.3 (Effective Cauchy modulus seal route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *packet factors through*

$$S, D, M, T \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The route exports a displayed finite tail schedule and does not export quotient limits, countable choice, host convergence, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier coordinates in order. A precision request first sees the stream and dyadic rows S, D, then the effective modulus and tail-stability rows M, T.

Only after those rows are displayed can the finite window W be opened. The regular rational readback Q and terminal handoff E_H are downstream of that finite window.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the refused principles, the exported object remains the finite schedule-to-window packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ packet is generated from a StreamName source row, a dyadic precision ledger, an effective precision-to-tail modulus row, a monotone tail-index comparison row, a finite tail-window row, a RegSeqRat readback, a RealUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3518.1, Definition 3518.2, and Definition 3518.3 seed the finite schedule object consumed by L10 StreamName, RegSeqRat, and Real routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient limits, countable choice, host convergence, trichotomy, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, schedule classification, monotone tail-index stability, RegSeqRat and RealUp seal compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchyModulusUp.

Chapter 2562

A Concrete Naming Certificate for $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$

The $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ packet realizes the MetaCIC audit-map frontier of section 9376.7 as finite BEDC rows. It sits next to the redex frontier, normalization frontier, confluence-audit packet, and decidable-boundary anchors already used by the MetaCIC vision, but it names the smaller object needed by a candidate-mediated reducer: a displayed finite choice surface for which redex rows may be inspected, which normalization and confluence-audit rows are available, and which boundaries remain undecided. The packet consumes the redex-row surface of chapter 1008, the candidate-normalization boundary of chapter 892, the local critical-pair fork surface of chapter 2382, and the beta critical-pair row of chapter 881. The normal-form endpoint row is consumed through chapter 740, while the redex-choice packet keeps subject reduction and global consistency outside the carrier. The packet does not assert global normalization, Church–Rosser, subject reduction for all terms, or a complete MetaCIC semantics.

Definition 2562.1 (MetaCIC redex-choice frontier carrier)**.** *A* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, N, F, D, B, H, K, L, C, P, Q)$$

with the following displayed rows:

1. *A is the MetaCIC redex row, listing the candidate redexs exposed by the local term-and-context window.*
2. *N is the normalization-frontier row, recording which candidates have a displayed bounded reduction trace.*
3. *F is the confluence-audit row, recording local fork and join checks for the same candidate set.*
4. *D is the decidable-boundary row, recording which redex choices are classified by finite inspection and which remain outside the packet.*
5. *B is the* BHist *replay ledger for the term window, candidate row, trace row, fork row, boundary row, and local naming row.*
6. *H records componentwise* hsame *transport for the redex, normalization, confluence-audit, decidability, replay, continuation, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a term-window request through A, N, F, D, B, H, P, Q.*

8. *L is the* Pkg *provenance over* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$*, MetaCIC redex rows, normalization-frontier rows, confluence-audit rows, decidable-boundary rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *C records the candidate-choice classifier: it exposes only the redex choices whose A, N, F, D rows are present in the packet.*
10. *P records the local package boundary for Nontrivial and FieldFaithful TasteGate instances to be checked by a later Lean development over the displayed carrier.*
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, N, F, D, B, H, K, L, C, P, Q$. It has no coordinate for global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, or a completed MetaCIC model.

Lean4 ✓ `BEDC.Derived.MetacicRedexChoiceFrontierUp.MetacicRedexChoiceFrontierTasteGate single carrier alignment`

Theorem 2562.2 (MetaCIC redex-choice frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier*

$$R = (A, N, F, D, B, H, K, L, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for MetaCIC redex rows, normalization-frontier traces, confluence-audit fork rows, decidable-boundary classifiers, replay, transport, continuation, provenance, candidate-choice, TasteGate boundary, and local naming rows;*
2. *redex-choice classification, requiring the candidate-choice row C to inspect only redexs displayed in A and only boundary outcomes displayed in D;*
3. *normalization-frontier compatibility, requiring every candidate exported by C to carry either the bounded trace row in N or the explicit boundary refusal in D;*
4. *confluence-audit compatibility, requiring every fork exported by C to use the same candidate set as A and the same local fork rows as F;*
5. *TasteGate preparation, requiring the packet to expose the Nontrivial and FieldFaithful instance boundary in P without claiming those Lean instances as paper-side evidence;*
6. *ledger refusal for global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, and a completed MetaCIC model.*

No redex-choice frontier obligation is stored outside these rows.

Proof. Project the accepted carrier. The term-facing data are the redex row A, the normalization-frontier row N, the confluence-audit row F, the decidable-boundary row D, and the candidate-choice row C.

The candidate classifier cannot inspect a redex outside A, because C is downstream of the displayed redex row. It cannot treat an undecided boundary as decided, because every exported choice must pass through D.

The normalization and confluence rows are tied to the same candidate set. A selected candidate therefore carries the bounded trace information from N and the local fork information from F exactly when those rows are displayed by the packet.

The TasteGate boundary row P records the intended Nontrivial and FieldFaithful instance surface for Lean without adding a marker or treating the instances as verified. The refused global principles have no carrier coordinate, so the NameCert surface remains finite. □

Theorem 2562.3 (MetaCIC redex-choice frontier nonescape)**.** *Every reducer-facing read from an accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *packet factors through*

$$A, N, F, D \longmapsto C \longmapsto B, H, K, L, P, Q.$$

The exported object is a finite candidate-mediated frontier for local redex choice, not global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, or a completed MetaCIC model.

Proof. Start with the displayed redex, normalization, confluence-audit, and decidable-boundary rows. These rows are the only inputs to the candidate-choice classifier.

The classifier C exports only the candidates that survive those displayed rows. Replay through B, transport through H, continuation through K, provenance through L, TasteGate boundary through P, and naming through Q preserve the same finite route.

Since no global normalization theorem, global confluence theorem, unrestricted redex selector, quotient syntax object, host evaluator soundness proof, or completed MetaCIC model appears in the carrier, the reducer-facing read cannot escape the finite frontier. □

Theorem 2562.4 (MetaCIC redex-choice frontier candidate-boundary exhaustion)**.** *For every accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier*

$$R = (A, N, F, D, B, H, K, L, C, P, Q),$$

each exported candidate choice is exhausted by the displayed A, N, F, D, C *rows before replay, provenance, transport, TasteGate preparation, and local naming are read through* B, H, K, L, P, Q*. No accepted route can import global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, or completed MetaCIC semantics.*

Proof. Begin with the candidate-choice classifier C. By Definition 3519.1, it exports only choices whose redex row A, normalization-frontier row N, confluence-audit row F, and decidable-boundary row D are already displayed.

Apply the nonescape theorem Definition 3519.3. After A, N, F, D and C have been consumed, replay and provenance can only preserve the same finite route through B, H, K, L, P, Q.

The refused global objects have no carrier coordinate. Thus the boundary is exhausted at the candidate-mediated finite rows and remains separate from global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, and completed MetaCIC semantics. □

Theorem 2562.5 (MetaCIC redex-choice candidate-normalization handoff)**.** *Every exported redex choice from an accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *packet consumed by a normalization-window or confluence-audit reader must pass through*

$$A, N, F, D, C \longmapsto B, H, K, L, P, Q.$$

The row N *supplies only displayed candidate-normalization evidence, the row* F *supplies only local fork information, and the row* D *preserves undecided boundary rows. The packet refuses global normalization, Church–Rosser, subject reduction for all terms, quotient syntax, host evaluator soundness, and a completed MetaCIC semantics.*

Proof. Start with the NameCert obligations of Definition 3519.2. Redex-choice classification forces C to inspect only redexs in A and boundary outcomes in D, while normalization-frontier and confluence-audit compatibility tie exported candidates to the displayed N and F rows.

The normalization-window reader therefore receives only candidate-normalization evidence from N. The confluence-audit reader receives only local fork information from F, and undecided rows remain recorded by D rather than being converted into global decisions.

Apply Definition 3519.4. Each exported candidate is already exhausted by A, N, F, D, C before replay, transport, continuation, provenance, TasteGate preparation, and local naming are read through B, H, K, L, P, Q.

Finally use Definition 3519.3. The refused global principles and host-side objects have no carrier coordinate, so the handoff cannot assert global normalization, Church–Rosser, subject reduction for all terms, quotient syntax, host evaluator soundness, or a completed MetaCIC semantics. □

Falsifiable prediction: A $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ reducer-facing read factors through displayed MetaCIC redex rows, normalization-frontier traces, confluence-audit fork rows, decidable-boundary classifiers, candidate-choice rows, BHist replay, hsame transport, Cont replay, Pkg provenance, TasteGate boundary rows, and local NameCert data.

Independence witness: metacic-redex, metacic-normalization-frontier, metacic-confluence-audit, metacic-decidable-boundary: $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ is not bijective to any listed anchor because it jointly carries candidate-choice classification, shared redex-boundary routing, Nontrivial and FieldFaithful TasteGate boundaries, replay, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ packet is generated from MetaCIC redex rows, normalization-frontier rows, confluence-audit rows, decidable-boundary rows, BHist replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, a candidate-choice classifier, Nontrivial and FieldFaithful TasteGate boundary rows, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3519.1, Definition 3519.2, Definition 3519.3, and Definition 3519.4 seed the finite candidate-mediated redex-choice frontier for MetaCIC audit maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, a completed MetaCIC model, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, redex-choice classification, normalization-frontier compatibility, confluence-audit compatibility, Nontrivial and FieldFaithful TasteGate instances, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetacicRedexChoiceFrontierUp.

Chapter 2563

A Concrete Naming Certificate for $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ packet records the finite route by which a displayed finite coproduct of metric sources is read before any completion-facing or real-facing consumer may inspect the completed object. It sits after the metric-completion surface of chapter 1506, the finite sum carrier of chapter 35, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite coproduct completion handoff, not a quotient completion equality, not a selected limit, not a countable-choice selector, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2563.1 (Metric completion finite-coproduct carrier)**.** *A* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{MCFC}(I, S, M, W, R, A, J, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the admitted summands.*
2. *S is the* $\mathsf{Sum}^{\uparrow}$ *coproduct row, recording injections, finite case analysis, and the branch classifier.*
3. *M is the family of* $\mathsf{MetricCompletion}^{\uparrow}$ *source rows, one for each admitted branch.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used when a branch request is inspected by a completion consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the branch-local tolerance and distance data.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W and R have fixed the requested branch data.*
7. *J is the coproduct-completion join row, stating that every accepted consumer first selects a displayed finite branch and then reads the corresponding completion route.*
8. *H records componentwise* hsame *transport for index, sum, completion, window, readback, seal, join, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a coproduct-completion request through I, S, M, W, R, A, J.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Sum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, S, M, W, R, A, J, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, infinite coproducts, countable choice, host case analysis outside S, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteCoproductUp.MetricCompletionFiniteCoproductTasteGate`
`single carrier alignment`

Theorem 2563.2 (Metric completion finite-coproduct NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *carrier* $\mathsf{MCFC}(I, S, M, W, R, A, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index row,* $\mathsf{Sum}^{\uparrow}$ *coproduct row, branchwise* $\mathsf{MetricCompletion}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, coproduct-completion join, componentwise transport, displayed replay, provenance, and local naming.*

Every obligation is branch-local after the finite case split in S, and every public read must pass through W, R, A before it may expose a completed coproduct branch. No obligation is stored in quotient completion equality, selected representatives, countable choice, infinite coproduct data, host case analysis, or ambient completeness.

Proof. Carrier admission is projection from Definition 3520.1. The tuple displays the finite index row, sum row, branchwise completion rows, numerical readback rows, join row, and structural support.

The finite case split is the row S. Once a branch is selected, the corresponding completion route in M is the only completion-facing source available to a consumer.

The readback rows W, R, A then bind finite observation windows, regular rational tolerance data, and the terminal Real seal to that same branch. The join row J records that this branch-local route is the public coproduct-completion handoff.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded quotient, choice, host, or ambient-completeness objects, the stated rows exhaust the NameCert obligations. □

Theorem 2563.3 (Metric completion finite-coproduct universal route)**.** *Every consumer that reads a finite coproduct after metric completion, or metric completion after a finite coproduct, factors through the displayed route*

$$\mathsf{Sum}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The exported object is the branch classifier named by $I, S, M, W, R, A, J, H, C, P, N$, not a quotient-level universal property and not a selected completed representative.

Proof. Start with the finite index and coproduct rows I, S. They are the only rows that select a branch, so any universal read begins with the displayed finite case split.

For the chosen branch, project the corresponding $\mathsf{MetricCompletion}^{\uparrow}$ row from M. The carrier offers no completed branch outside this family.

Now pass through W, R, A. These rows provide the finite stream window, regular rational readback, and real seal used by the branch consumer.

Apply Definition 3520.2. The join, transport, replay, provenance, and naming rows keep the same route visible, so the consumer receives only the finite universal route and cannot import a quotient completion isomorphism or selected representative. □

Theorem 2563.4 (Metric completion finite-coproduct nonescape)**.** *No accepted* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *consumer can escape from the finite branch route into quotient completion equality, selected limits, countable choice, an infinite coproduct, host-side case analysis, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exposed claim is read through the same finite packet rows*

$$I, S, M, W, R, A, J, H, C, P, N.$$

Proof. Use Definition 3520.3. It places every completion-coproduct consumer behind the finite branch classifier and the branch-local completion rows.

The numerical boundary is fixed by W, R, A. These rows provide only finite windows, regular rational readback, and a Real seal for the displayed branch.

The support rows J, H, C, P, N join, transport, replay, source, and name the same packet. Because no excluded object occurs as a coordinate of Definition 3520.1, no consumer can export it. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ packet is generated from a finite index row, a SumUp coproduct row, branchwise MetricCompletionUp rows, StreamName windows, RegSeqRat readback, a RealUp seal, a coproduct-completion join row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3520.1, Definition 3520.2, Definition 3520.3, and Definition 3520.4 seed the finite coproduct completion route over MetricCompletionUp, SumUp, StreamNameUp, RegSeqRatUp, RealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, infinite coproducts, host case analysis, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite branch classification, branchwise completion routing, StreamName and RegSeqRat readback, Real seal exactness, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionFiniteCoproductUp.

Chapter 2564

A Concrete Naming Certificate for $\mathsf{LatentCompletion}^{\uparrow}$

The $\mathsf{LatentCompletion}^{\uparrow}$ packet realizes the finite monotone-path completion witness of Definition 9462.1836 as BEDC NameCert data. It records observed module states, latent prefix states, target-monotone edges, exception rows, and a completion-choice ledger before any consumer may claim coverage of the module-state exception geometry. The observed module-state ledger is consumed through chapter 3470, chapter 3508, and chapter 3478 before the latent-prefix exception rows are read. The packet is a finite completion surface for the two missing monotone squares of Definition 9462.1837; it is not an empirical biological transition model, not a host graph-completion oracle, and not a global state-space completeness claim.

Definition 2564.1 (Latent completion carrier)**.** *A* $\mathsf{LatentCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{LatentCompletion}(O, E, L, D, M, K, H, C, P, N)$$

with the following displayed rows:

1. *O is the observed module-state ledger containing the sampled endpoints and their module-value rows.*
2. *E is the exception ledger for the target-monotone failures represented by table identifiers* 9*,* 14*, and* 28*.*
3. *L is the latent-state row containing one deep-branch prefix and one partial-boundary prefix from Definition 9462.1836.*
4. *D is the directed rewrite-edge row for target-monotone one-module changes.*
5. *M is the monotone-path coverage row verifying that every observed endpoint is reachable after the latent rows are admitted.*
6. *K is the exception-kill row recording that the two latent rows cover the two independent missing squares and no third exception branch is created.*
7. *H records componentwise* hsame *transport for observed states, latent states, rewrite edges, coverage rows, exception rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of a coverage request through O, E, L, D, M, K.*
9. *P is the* Pkg *provenance over* $\mathsf{LatentCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows, with the module-state source tied to Definition 9462.1836.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, E, L, D, M, K, H, C, P, N$. It has no coordinate for a continuous biological transition, a complete empirical state space, a graph oracle, unbounded latent search, or a module-state equality principle outside the displayed rows.

Theorem 2564.2 (Latent completion NameCert obligations)**.** *For every accepted* $\mathsf{LatentCompletion}^{\uparrow}$ *carrier* $\mathsf{LatentCompletion}(O, E, L, D, M, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for observed module states, exception rows, latent-prefix rows, directed rewrite edges, monotone-path coverage, exception-kill accounting, componentwise transport, displayed replay, provenance, and local naming.*

Every accepted coverage read must identify the two independent missing squares from E, consume the two latent rows in L, rebuild the directed rewrite graph through D, and then expose coverage through M, K, H, C, P, N. No obligation is stored in a host graph-completion oracle, a biological-transition claim, an unbounded latent search, or a complete empirical state space.

Lean4 ✓ `BEDC.Derived.LatentCompletionUp.LatentCompletionTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3521.1. The tuple displays the observed ledger, exception ledger, latent rows, rewrite edges, coverage row, exception-kill row, and structural support.

The exception analysis is read from E and L. The deep-branch exception rows require one deep prefix, and the partial-boundary exception row requires one partial-boundary prefix.

The coverage route is D, M, K. The directed rewrite row rebuilds target-monotone one-module steps, the coverage row records reachability of the observed endpoints, and the exception-kill row records that no third missing square remains.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no graph oracle, biological transition row, unbounded search row, or complete empirical state-space row, the stated surface exhausts the NameCert obligations. □

Theorem 2564.3 (Latent completion monotone-path coverage)**.** *Every accepted* $\mathsf{LatentCompletion}^{\uparrow}$ *packet covers the finite monotone-path exception geometry by routing each observed endpoint through the displayed rewrite graph after adjoining exactly one deep-branch latent prefix and exactly one partial-boundary latent prefix. The public coverage claim is the finite route*

$$O, E, L, D, M, K, H, C, P, N$$

and it exports no stronger temporal, mechanistic, or empirical completeness claim.

Proof. Start with the observed ledger O and exception ledger E. They identify the endpoints and the three target-monotone failures that must be covered.

Use the latent row L. It supplies one deep-branch prefix for the table-9 and table-14 square and one partial-boundary prefix for the table-28 square.

Now apply the rewrite-edge and coverage rows D, M. These rows rebuild the finite directed graph and record reachability of every observed endpoint from the standard state.

The exception-kill and support rows K, H, C, P, N preserve and name the same coverage route. Since no temporal mechanism or empirical-completeness coordinate is present, the exported conclusion is exactly monotone-path coverage of the finite module-state packet. □

Falsifiable prediction: If either the deep-branch latent prefix or the partial-boundary latent prefix is removed from an accepted $\mathsf{LatentCompletion}^{\uparrow}$ packet, at least one of the table-9, table-14, or table-28 monotone-path exceptions remains uncovered.

Independence witness: six-boundary-tile-coverage, dna-boundary-trigger-closure, window-coverage-trigger-algebra, choice-window-sequence, metacic-redex-choice-frontier: none is bijective to $\mathsf{LatentCompletion}^{\uparrow}$ because this packet binds observed module states, two latent prefix choices, directed rewrite edges, monotone-path coverage, and exception-kill accounting in one NameCert surface.

The Lean follow-up supplies Nontrivial and FieldFaithful TasteGate instances for the accepted latent-completion carrier, so the formal route can distinguish nonempty coverage from faithful preservation of the module-state fields without adding new paper-side rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LatentCompletion}^{\uparrow}$ packet is generated from observed module-state rows, exception rows, two latent-prefix rows, directed rewrite-edge rows, monotone-path coverage, exception-kill accounting, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3521.1, Definition 3521.2, and Definition 3521.3 seed the finite monotone-path completion witness over observed module states, latent prefixes, rewrite edges, coverage rows, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close empirical biological transition mechanisms, complete state-space sampling, host graph-completion oracles, unbounded latent search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exception splitting, latent-prefix coverage, directed rewrite reachability, exception-kill accounting, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.LatentCompletionUp.

Chapter 2565

A Concrete Naming Certificate for $\mathsf{BoundedFunctionFamily}^{\uparrow}$

Following the bounded-family usage standard in Kelley, *General Topology*, and Bourbaki-style analysis, the $\mathsf{BoundedFunctionFamily}^{\uparrow}$ packet records the finite BEDC surface for a family of continuous maps whose displayed values are controlled by pointwise real bounds and a sup-norm ledger. It sits between the compact-continuity route of chapter 129, the equicontinuity family surface of chapter 413, the Arzela–Ascoli packet of chapter 412, and the real-name boundary of chapter 38. The carrier is a finite BHist packet over $\mathsf{ContinuousMap}^{\uparrow}$ rows, a $\mathsf{CompactMetric}^{\uparrow}$ source row, a $\mathsf{CompleteMetric}^{\uparrow}$ target row, pointwise $\mathsf{Real}^{\uparrow}$ bound rows, a displayed sup-norm ledger, hsame, Cont, Pkg, and a local $\mathsf{NameCert}$ row.

Definition 2565.1 (Bounded function-family carrier)**.** *A* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, I, M, B, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *surface of chapter 129;*
2. *Y is the complete metric target row, supplying only the displayed target metric and completion-facing name data;*
3. *I is the finite family-index row naming the inspected family members;*
4. *M is the finite list of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, one for each displayed family member from X into Y;*
5. *B is the pointwise* $\mathsf{Real}^{\uparrow}$ *bound ledger, recording for each displayed index and inspected source window the real-name bound read from the value row;*
6. *U is the displayed sup-norm ledger, obtained by comparing the value-distance rows against the finite pointwise bound surface and not by selecting an ambient supremum;*
7. *H stores componentwise* hsame *transport for the source, target, index, map, pointwise-bound, sup-norm, provenance, and name rows;*
8. *C stores the displayed* Cont *replay from compact source inspection through continuous-map values, pointwise real bounds, and the sup-norm ledger;*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$ *row.*
The carrier has no row for an open-cover compactness principle, an ambient function-space topology, a selected supremum, quotient real equality, or a choice-selected subsequence.

Lean4 (definition) `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier`
Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyUpTasteGate single carrier alignment`

Theorem 2565.2 (Bounded function-family NameCert obligations)**.** *For the carrier of Definition 3522.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for* $F = (X, Y, I, M, B, U, H, C, P, N)$*, compact-source readback through* X*, complete-target readback through* Y*, finite index admission through* I*, continuous-map readback through* M*, pointwise* $\mathsf{Real}^{\uparrow}$ *bound readback through* B*, displayed sup-norm ledger readback through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and terminal naming through* N*.*

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier namecert obligations`

Proof. Project the carrier row by row. The source row X is read through the compact metric certificate, while Y supplies only the complete metric target surface needed by the displayed value distances.

The finite index row I then bounds the family members that can be inspected. Each inspected member is read through a corresponding $\mathsf{ContinuousMap}^{\uparrow}$ row in M, so no ambient function-space topology is introduced.

The pointwise bound row B and the sup-norm ledger U are downstream of the map rows. They compare displayed real-name value distances and record the bound ledger as finite data; they do not choose a supremum in the host real line and do not identify quotient real representatives.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no other coordinate occurs in F, these rows exhaust the local NameCert obligations. □

Theorem 2565.3 (Bounded function-family sup-norm route)**.** *Every accepted* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *packet exports its sup-norm ledger only through*

$$X, Y, I, M, B, U, H, C, P, N.$$

A consumer first reads the compact source row, complete target row, finite family-index row, and displayed continuous-map rows; only then may it read the pointwise $\mathsf{Real}^{\uparrow}$ *bound rows and the finite sup-norm ledger. The route exports no open-cover compactness theorem, ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence.*

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier sup norm route`

Proof. Start with the source and target rows. The $\mathsf{CompactMetric}^{\uparrow}$ row X supplies the inspected source windows, and the $\mathsf{CompleteMetric}^{\uparrow}$ row Y supplies the target metric surface without importing a function-space object.

Next read the finite family data. The index row I lists the family members available to the packet, and M attaches each listed member to a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row.

The pointwise real bounds in B are then read against those map values. The sup-norm ledger U is a finite comparison ledger over the displayed values and bounds, so it can be replayed by C and transported by H without selecting an ambient supremum.

The provenance and local name rows P, N close the packet. Because the carrier contains no open-cover, ambient topology, quotient equality, or subsequence-selector coordinate, the listed rows exhaust every sup-norm read available to downstream Arzela–Ascoli and equicontinuity consumers. □

Theorem 2565.4 (Bounded function-family sup-norm non-escape)**.** *Any Arzela–Ascoli, equicontinuity, or compact-continuity consumer using the* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *sup-norm ledger must read exactly*

$$X, Y, I, M, B, U, H, C, P, N.$$

Consequently it cannot replace U by an ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence. The only available sup-norm data are the displayed pointwise $\mathsf{Real}^{\uparrow}$ *bound comparisons, the finite family-index and continuous-map rows, structural transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3522.1. The rows available to a sup-norm consumer are precisely $X, Y, I, M, B, U, H, C, P, N$, with U positioned after the compact source, complete target, finite index, continuous-map, and pointwise-bound rows.

Apply Definition 3522.2 to identify the local certificate obligations and Definition 3522.3 to force every sup-norm read through the displayed route.

No additional carrier coordinate remains after those rows are exhausted. Thus an Arzela–Ascoli, equicontinuity, or compact-continuity consumer cannot import an ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence in place of U. □

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilySupNormNonescape`

Falsifiable prediction: An accepted $\mathsf{BoundedFunctionFamily}^{\uparrow}$ consumer that reads a sup-norm ledger without the displayed finite family-index row, continuous-map rows, pointwise $\mathsf{Real}^{\uparrow}$ bound rows, and structural hsame, Cont, Pkg, and NameCert rows refutes the carrier surface.

Independence witness: compactmetric, equicontinuityfamily, arzelaascoli, real: no listed sibling jointly carries the compact source, complete target, finite continuous-map family, pointwise real-bound ledger, displayed sup-norm route, transport, replay, provenance, and local name rows of $F = (X, Y, I, M, B, U, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedFunctionFamily}^{\uparrow}$ packet is generated from a compact metric source row, complete metric target row, finite family-index row, continuous-map rows, pointwise RealUp bound rows, displayed sup-norm ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$, seed)

Scope closed: Definition 3522.1, Definition 3522.2, Definition 3522.3, and Definition 3522.4 bind the finite bounded-family carrier and its displayed sup-norm ledger route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient function-space topology, selected suprema, quotient real equality, choice-selected subsequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-map readback, pointwise bound exactness, displayed sup-norm replay, structural transport, provenance, and local naming for BEDC.Derived.BoundedFunctionFamilyUp.

Chapter 2566

A Concrete Naming Certificate for CauchyCompletionApartness$^{\uparrow}$

Following Bishop and Bridges on constructive metric completion and apartness, the CauchyCompletionApartness$^{\uparrow}$ packet records the positive separation boundary for completed Cauchy names without passing through quotient representatives. It is a Real-and-Completion neighbour of the universal completion route in chapter 2319, the apartness completion surface of chapter 1567, the separated metric classifier in chapter 1411, the regular rational readback of chapter 301, and the finite stream window interface of chapter 126.

Definition 2566.1 (Cauchy completion apartness carrier)**.** *A* CauchyCompletionApartness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, R, S, W, E, G, H, C, P, N)$$

with the following displayed rows:

1. *U is the* CauchyCompletionUniversalProperty$^{\uparrow}$ *completion source row, exposing the completed Cauchy-name request before any apartness read is attempted.*
2. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the Cauchy name and to the requested tolerance budget.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row through which the compared Cauchy names are inspected.*
4. *W is the* Real$^{\uparrow}$ *witness-seal row reached only after the regular readback and stream-window rows have been displayed.*
5. *E is the* SeparatedMetric$^{\uparrow}$ *zero-distance classifier handoff that identifies when a completed comparison must collapse to the separated metric boundary.*
6. *G is the* RealApartnessCompletion$^{\uparrow}$ *positive-gap row, recording apartness by a displayed completed-real gap rather than by an external representative choice.*
7. *H records componentwise* hsame *transport for the completion source, readback, windows, real seal, separatedness handoff, apartness gap, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an apartness request through U, R, S, W, E, G.*
9. *P is the* Pkg *provenance over* CauchyCompletionApartness$^{\uparrow}$, CauchyCompletionUniversalProperty$^{\uparrow}$, RealApartnessCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, R, S, W, E, G, H, C, P, N$. It has no coordinate for quotient-stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Theorem 2566.2 (Cauchy completion apartness NameCert obligations)**.** *For every accepted* CauchyCompletionApa *carrier*

$$A = (U, R, S, W, E, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion source, regular readback, stream-window, real-seal, separatedness, apartness-gap, transport, replay, provenance, and local naming rows;*
2. *completion-source exposure, requiring an apartness request to pass through the displayed* CauchyCompletionUniver *row U before any completed comparison is read;*
3. *regular-window readback, requiring R and S to be the only numerical and stream observation route before the* $\mathsf{Real}^{\uparrow}$ *seal row W is reached;*
4. *separatedness handoff, requiring zero-distance collapse to be read only through the displayed* $\mathsf{SeparatedMetric}^{\uparrow}$ *row E;*
5. *apartness-gap exposure, requiring positive separation to be read only through the* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *row G after U, R, S, W, E have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport through H and displayed replay through C to preserve the same route;*
7. *ledger non-escape, excluding quotient representatives, selected limits, host equality, trichotomy, countable choice, ambient completeness, Lean verification, and bridge checking from the packet.*

No NameCert obligation is stored outside these rows.

```
Lean4✓  BEDC.Derived.CauchyCompletionApartnessUp.TasteGate.CauchyCompletionApartnessCarrier
namecert obligations
```

Proof. Carrier admission is projection from Definition 3523.1. The accepted packet exposes exactly the completion source, regular readback, finite windows, real seal, separatedness row, apartness-gap row, and structural rows.

The completion-facing route begins at U. The regular rational row R and stream-window row S then supply the only finite observation data available before the real seal W is read.

Separatedness and apartness are two downstream readings of the same displayed route. The row E records the separated metric handoff for zero-distance collapse, while G records the positive-gap apartness completion row.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since the carrier has no coordinate for quotient representatives, host equality, trichotomy, choice, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2566.3 (Cauchy completion apartness separatedness handoff)**.** *Every separatedness or apartness read of an accepted* $\mathsf{CauchyCompletionApartness}^{\uparrow}$ *carrier*

$$A = (U, R, S, W, E, G, H, C, P, N)$$

must factor through the finite route

$$U \longmapsto R, S, W \longmapsto E, G \longmapsto H, C, P, N.$$

Consequently a consumer receives only the displayed completion source, regular rational readback, finite stream windows, real seal, separated-metric handoff, apartness-completion gap, componentwise transport, continuation replay, provenance, and local name. It receives no quotient representative, selected limit, host real equality, trichotomy split, countable-choice schedule, or ambient completeness theorem.

Proof. Start with a separatedness or apartness read of the carrier. Projection exposes U first, so the read begins from the displayed universal completion row rather than from a quotient representative of the completed Cauchy name.

The numerical read then passes through R, S, W. The regular rational row and finite stream-window row fix the inspected data before the real seal is available.

The comparison boundary is exactly E, G. Zero-distance collapse is handled by the separated metric row E, and positive separation is handled by the apartness-completion gap row G.

The structural rows H, C, P, N preserve and name only that finite route. Because no excluded coordinate appears in the carrier, the handoff cannot export quotient representatives, selected limits, host real equality, trichotomy, choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionApartness}^{\uparrow}$ packet is generated from a CauchyCompletionUniversalProperty source row, RegSeqRat readback, StreamName windows, a RealUp seal row, a SeparatedMetric handoff, a RealApartnessCompletion positive-gap row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3523.1, Definition 3523.2, and Definition 3523.3 seed the finite apartness route for completed Cauchy names over displayed completion, readback, window, separatedness, gap, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host real equality, real trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-source exposure, regular-window readback, separatedness handoff, apartness-gap exposure, classifier stability, consumer nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionApartnessUp.

Chapter 2567

A Concrete Naming Certificate for $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ packet records the finite comparison certificate by which located Dedekind cut language and regular Cauchy real names meet inside the existing $\mathsf{LocatedReal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ routes. It is a NameCert reconstruction over the concrete rows of chapter 1933, chapter 436, chapter 301, chapter 291, and chapter 38.

The packet is deliberately finite. It records a displayed cut row, a located-real enclosure row, a regular rational readback, a dyadic bracket ledger, and a terminal real seal, but it does not add quotient reals, trichotomy, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 2567.1 (Bishop regular-cut equivalence carrier)**.** *A* $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, L, R, D, Q, H, T, P, N)$$

with the following visible rows:

1. *C is a displayed Dedekind or located cut row whose lower and upper sides are read as positive and negative rational evidence rather than as quotient-real data;*
2. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row that records the finite located interval information associated to the cut row C;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the located enclosures, giving the regular Cauchy name visible to completion consumers;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger that records the dyadic lower and upper rational brackets used to compare the cut and regular-name presentations;*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from R through the existing real-name route;*
6. *H stores the displayed* hsame *transports for C, L, R, D, Q;*
7. *T stores the displayed* Cont *routes that replay the cut-to-enclosure, enclosure-to-readback, bracket-compatibility, and real-seal handoffs;*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$ *row for the finite comparison packet.*

The carrier predicate is $\mathsf{BishopRegularCutEquivalencePacket}(E)$*. Its public rows are exactly the displayed cut, located enclosure, regular-rational readback, dyadic bracket ledger, real seal, transports, provenance, and local naming row.*

Theorem 2567.2 (Bishop regular-cut equivalence NameCert obligations)**.** *For the carrier*

$$E = (C, L, R, D, Q, H, T, P, N)$$

of Definition 3524.1, the local $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$ *surface has five obligations:*

1. *carrier admission for the displayed finite packet;*
2. *bracket and readback compatibility, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *brackets in* D *to support both the located cut row* C *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*;*
3. *located-real handoff, requiring the enclosure row* L *to be the only route from* C *to* R*;*
4. *classifier transport, requiring accepted carriers to compare only through the displayed* hsame *and* Cont *routes in* H, T *together with the package row* P*;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring every real-facing read through* Q *to factor through* C, L, R, D, H, T, P, N *and no outside quotient, trichotomy, choice, or completeness datum.*

Lean4 ✓ `BEDC.Derived.BishopRegularCutEquivalenceUp.TasteGate.BishopRegularCutEquivalenceCarrier namecert obligations`

Proof. Carrier admission is projection from the finite packet E. The rows list the cut display, located enclosure, regular readback, dyadic bracket ledger, real seal, transports, provenance, and local naming certificate as one visible history.

Bracket and readback compatibility is the joint routing condition on D and R. The same dyadic bracket ledger that locates the cut side also supplies the rational bounds used by the regular Cauchy readback, so the comparison has no hidden quotient-real coordinate.

The located-real handoff is the row L. Any passage from the cut row C to the regular name R must pass through the displayed located enclosure, and any transported comparison must replay through H, T with provenance P.

Finally the real-facing row Q is a terminal seal, not a new completeness principle. Since Q is reached only after the displayed cut, enclosure, readback, bracket, transport, package, and local certificate rows have been consumed, the packet exports the finite comparison surface and no stronger real-analysis principle. □

Theorem 2567.3 (Bishop regular-cut equivalence located-real handoff)**.** *For every accepted* $\mathsf{BishopRegularCutEquiv}$ *carrier*

$$E = (C, L, R, D, Q, H, T, P, N),$$

the public comparison between the located cut row C *and the regular Cauchy row* R *factors through the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row* L *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger* D*. The* $\mathsf{Real}^{\uparrow}$ *seal row* Q *may be consumed only after that finite route has been replayed through* H, T, P, N*.*

Consequently the packet gives a finite L10 comparison certificate between Bishop located cuts and regular Cauchy real names without asserting Dedekind completeness, quotient-real extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3524.1. The row C is the displayed cut surface, L is the located enclosure row attached to it, and R is the regular rational readback.

Apply Definition 3524.2. The bracket and readback obligation forces D to be the shared finite dyadic ledger for both sides of the comparison, while the located-real handoff obligation prevents a direct jump from C to R.

The classifier transport obligation then carries the same route through H, T, P, N. The real-seal nonescape obligation leaves Q as the terminal consumer of that route, so no unlisted quotient, trichotomy, choice, or completeness datum can be read from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ packet is generated from a displayed located cut row, a $\mathsf{LocatedReal}^{\uparrow}$ enclosure row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ bracket ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, displayed hsame and Cont transports, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3524.1, Definition 3524.2, and Definition 3524.3 bind the finite comparison surface between Bishop located cuts and regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient reals, trichotomy, choice principles, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The next paper grade is a checked finite carrier theorem for the cut, located-enclosure, regular-readback, dyadic-bracket, and real-seal rows.

Chapter 2568

A Concrete Naming Certificate for DyadicToleranceLadder$^{\uparrow}$

Following Bishop and Bridges' constructive use of rational tolerances, the DyadicToleranceLadder$^{\uparrow}$ packet records the finite ladder of dyadic requests used by Real and Completion consumers before any completed real object is exposed. It sits on the dyadic arithmetic core of chapter 291, the finite request-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the tail-budget packet of chapter 379. The packet gives downstream Real and Completion chapters one reusable source object for the tolerances 2^{-n}, their halving route, and their finite handoff to regular-rational windows.

Definition 2568.1 (DyadicToleranceLadder carrier)**.** *A* DyadicToleranceLadder$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, Q, E, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *endpoint row, recording dyadic numerators, denominators, comparison rows, and the visible tolerance term 2^{-n}.*
2. *S is the* StreamName$^{\uparrow}$ *request-window row, recording the finite list of precision indices at which a consumer asks for tolerances.*
3. *Q is the* RegSeqRat$^{\uparrow}$ *regular-rational window row, recording the rational readback surface that consumes the requested dyadic tolerances.*
4. *E is the* Real$^{\uparrow}$ *seal boundary row, recording only the terminal handoff at which the tolerance ladder can be read by a real-facing consumer.*
5. *T is the* DyadicTailBound$^{\uparrow}$ *row, recording the finite tail-budget comparison that treats successive ladder entries as shrinking tolerance requests.*
6. *H records componentwise* hsame *transport for the dyadic endpoint, window, regular-rational, real-seal, tail-bound, continuation, provenance, and naming rows.*
7. *C records displayed* Cont *routes from dyadic endpoints through finite windows, regular-rational readback, tail-bound comparisons, and the real-seal boundary.*
8. *P is the* Pkg *provenance over* DyadicToleranceLadder$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, DyadicTailBound$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, Q, E, T, H, C, P, N$. It has no coordinate for quotient stream equality, trichotomy on real names, selected limits, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DyadicToleranceLadderUp.DyadicToleranceLadderTasteGate single carrier alignment`

Theorem 2568.2 (DyadicToleranceLadder NameCert obligations)**.** *For every accepted* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *carrier*

$$L = (D, S, Q, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ *surface consists exactly of carrier exposure for the dyadic endpoint row D, finite request-window row S, regular-rational readback row Q, real-seal boundary E, tail-bound comparison row T, componentwise transport row H, continuation row C, package provenance row P, and local naming row N.*

Proof. Carrier exposure is projection from Definition 3525.1. Each row named in the statement is one coordinate of the finite packet, and the definition lists no hidden coordinate for a completed real object.

The dyadic endpoint and request-window obligations are D and S. The regular-rational readback and real-facing boundary obligations are Q and E, while the shrinking tolerance budget is the row T.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these rows gives the displayed $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ surface, with quotient stream equality, trichotomy, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the packet. □

Theorem 2568.3 (DyadicToleranceLadder monotone halving)**.** *Let $L = (D, S, Q, E, T, H, C, P, N)$ be an accepted* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *carrier. If two requested indices m, n are both displayed in the finite* $\mathsf{StreamName}^{\uparrow}$ *window S and m is the successor request after n, then the tolerance read from D at m is routed through the same dyadic endpoint row as one halving of the tolerance read at n, and the comparison is consumed by T before Q or E can export the request.*

Proof. The only dyadic endpoint row in the carrier is D, so both displayed requests are read from the same dyadic arithmetic surface. The successor relation is visible only as part of the finite window S, not as an external sequence choice.

The halving comparison is the role of the tail-bound row T. It consumes the two displayed endpoint reads before the regular-rational window Q is opened, so the comparison cannot be replaced by an unrecorded real-order argument.

After T records the comparison, the readback row Q and the seal boundary E may consume the tolerance request. The route therefore remains a finite ladder through $D, S, T, Q, E, H, C, P, N$, and it exports no quotient stream equality, trichotomy, selected limit, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicToleranceLadder}^{\uparrow}$ packet is generated from dyadic endpoint rows, finite StreamName request windows, RegSeqRat readback windows, Real seal boundaries, DyadicTailBound comparison rows, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$, seed)

Scope closed: Definition 3525.1, Definition 3525.2, and Definition 3525.3 bind the finite dyadic tolerance ladder, its local NameCert surface, and its monotone halving route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove Real$^{\uparrow}$ completeness, quotient stream equality, trichotomy, selected limits, countable choice, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the local NameCert obligation surface, and the monotone halving route over the displayed dyadic and regular-rational rows.

Chapter 2569

A Concrete Naming Certificate for $\mathsf{NestedClosedBall}^{\uparrow}$

Following Bishop and Bridges and the Cantor-style nested closed ball criterion, the $\mathsf{NestedClosedBall}^{\uparrow}$ packet records explicit nested closed ball completion data as finite NameCert rows. It sits beside the metric-completion carrier of chapter 1506, the complete-metric service of chapter 128, the Cauchy-filter evidence of chapter 333, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a displayed descending ball route rather than a host completeness theorem, a quotient completion, or a choice-selected center.

Definition 2569.1 (Nested closed ball carrier)**.** *A* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, K, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row naming the ambient completion request before any ball center is read.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row available only as a displayed limit-service dependency, not as an independent selector.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *filter row generated by the descending family of closed balls.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspecting the indexed balls and their center approximations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the inspected center and radius data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger, recording the dyadic shrink schedule, containment witnesses, and late-radius comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the completion source, complete-metric service, filter, windows, readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a nested-ball request through M, K, F, S, R, D, E.*
10. *P is the* Pkg *provenance over* $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected limits, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.TasteGate.NestedClosedBallTasteGate single carrier alignment`

Theorem 2569.2 (Nested closed ball NameCert obligations)**.** *For every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion source, displayed complete-metric service, Cauchy-filter row, finite stream windows, regular rational readback, dyadic diameter ledger, terminal real seal, componentwise transport, displayed continuation replay, package provenance, and local naming. Every nested-ball read is therefore routed through $M, K, F, S, R, D, E, H, C, P, N$, and no obligation is stored outside those rows.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallNameCertObligations`

Proof. Carrier admission is projection from Definition 3526.1. The packet displays the completion source, the complete-metric service, the Cauchy filter generated by the nested balls, the finite windows, the rational readback, and the dyadic diameter ledger before the real seal can be consumed.

The constructive nested-ball content is the route through F, S, R, D. The filter row records the descending family, the stream row exposes only finite indexed windows, the regular rational row reads center and radius data, and the dyadic row records the shrink and containment comparisons.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. Since the carrier contains no open-cover compactness row, selected limit, quotient completion, countable-choice schedule, ambient real-completeness row, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2569.3 (Nested closed ball Cauchy-center route)**.** *Every center read exported by an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

Equivalently, a consumer may inspect the descending closed-ball family only through the displayed Cauchy-filter row, finite stream windows, regular rational readback, dyadic shrink ledger, and terminal real seal; it receives no open-cover compactness theorem, selected limit, quotient completion, countable-choice schedule, independent real-completeness witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier and project the rows M, K, F, S, R, D, E. The metric-completion and complete-metric rows provide the surrounding completion service, while the Cauchy-filter row is the only entry point for the descending closed-ball family.

The center data are then inspected through finite windows. The stream row S exposes finite ball indices, the regular rational row R reads the center and radius approximants, and the dyadic ledger D records the shrink and containment comparisons needed for a Cauchy-center route.

Finally the real seal E and the structural rows H, C, P, N preserve the same route for public consumers. Because no excluded coordinate occurs in the carrier, the handoff cannot export open-cover compactness, selected limits, quotient completions, countable choice, or ambient real completeness. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallCauchyCenterRoute`

Theorem 2569.4 (Nested closed ball RegSeqRat center extraction)**.** *Every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *packet whose displayed diameters refine through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows yields a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing center readback through the* $\mathsf{StreamName}^{\uparrow}$ *finite windows before the* $\mathsf{Real}^{\uparrow}$ *seal is available. The extracted readback uses the nested closed-ball carrier row, the local NameCert obligations, and the Cauchy-center route to expose only the finite center schedule, dyadic shrink comparisons, and regular-rational handoff.*

Proof. Begin with the carrier of Definition 3526.1. Its rows S, R, D are the only finite-window, regular-rational, and dyadic tolerance coordinates available before the terminal real seal.

Apply the local obligations of Definition 3526.2. The NameCert surface requires the Cauchy-filter row, finite windows, regular rational readback, and dyadic diameter ledger to be admitted together, so a displayed diameter refinement through D cannot bypass S or R.

Use the Cauchy-center route of Definition 3526.3. The nested finite-ball chain is inspected through S, its center and radius approximants are read through R, and the dyadic shrink comparisons in D provide the tolerance control needed for a center readback.

The structural rows H, C, P, N preserve this route under transport and replay. Since E is downstream of S, R, D, the packet yields the $\mathsf{RegSeqRat}^{\uparrow}$-facing center readback before any $\mathsf{Real}^{\uparrow}$ seal is consumed. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallRegSeqRatCenterExtraction`

Theorem 2569.5 (Nested closed ball diameter-tail stability)**.** *For every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N),$$

the dyadic diameter ledger D is stable under later $\mathsf{StreamName}^{\uparrow}$ *windows S and under the componentwise* hsame *and* Cont *transport rows H, C. Every later ball read therefore keeps the displayed containment witness and shrink comparison before the* $\mathsf{Real}^{\uparrow}$ *seal E is consumed.*

The packet exports no selected center, quotient completion, choice witness, or ambient completeness object through this stability route.

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallCarrier diameter tail stability`

Proof. Project the diameter, stream, transport, and replay rows D, S, H, C from the accepted carrier. The dyadic ledger records the shrink comparisons and containment witnesses before the terminal seal is available.

Use the Cauchy-filter row F to keep tail membership in the descending closed-ball family. Later stream windows in S inspect only finite indexed balls already covered by this filter row, so no later read can replace the displayed containment ledger.

The regular rational row R supplies the scheduled center and radius reads attached to those windows. Its center readback is downstream of the dyadic ledger, so the same shrink comparison remains attached to the later window.

Finally, H transports the completion source, filter, stream windows, readback, dyadic ledger, and seal componentwise, while C replays the same nested-ball route. Thus every later ball read preserves containment and diameter control before E is consumed. Since the carrier has no selected-center, quotient-completion, choice, or ambient-completeness coordinate, the stability theorem exports none. □

Theorem 2569.6 (Nested closed ball obligation package)**.** *The carrier of Definition 3526.1, the local obligations of Definition 3526.2, and the Cauchy-center route of Definition 3526.3 jointly form the exact obligation package consumed by later metric-completion and complete-metric chapters. The package consists of the metric-completion source row, complete-metric service row, Cauchy-filter row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row.*

It supplies no open-cover compactness, selected limit, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallObligationPackage`

Proof. Start with the carrier of Definition 3526.1. Its displayed tuple is precisely the list of metric-completion, complete-metric, Cauchy-filter, stream-window, regular-rational, dyadic, real-seal, transport, replay, provenance, and naming rows.

Apply Definition 3526.2. This identifies the local NameCert surface with admission and replay through exactly those rows, with no obligation stored outside the packet.

Finally apply Definition 3526.3. Every center read factors through the same finite route, so later metric-completion and complete-metric consumers receive one obligation package and no open-cover compactness, selected limit, quotient completion, countable-choice schedule, or ambient real-completeness witness. □

Theorem 2569.7 (Nested closed ball consumer nonescape)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$*-,* $\mathsf{CompleteMetric}^{\uparrow}$*-, or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *packet must read exactly the displayed route*

$$M, K, F, S, R, D, E, H, C, P, N$$

before using any nested-ball endpoint. The consumer may use the selected center only through the finite windows, regular-rational readback, dyadic diameter ledger, Cauchy-filter route, terminal real seal, componentwise transport, displayed replay, provenance, and local NameCert rows of that packet.

Consequently selected centers, quotient completions, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness cannot escape the packet boundary.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallConsumerNonescape`

Proof. Start with the carrier of Definition 3526.1. Its displayed tuple is the only accepted source for a nested-ball endpoint, and the local obligations of Definition 3526.2 identify the NameCert surface with the rows $M, K, F, S, R, D, E, H, C, P, N$.

Apply the Cauchy-center route of Definition 3526.3. Any center read used by a metric-completion, complete-metric, or real-facing consumer must pass through the Cauchy-filter row F, the stream windows S, the regular-rational readback row R, and the dyadic ledger D before the terminal seal E is available.

Use the RegSeqRat extraction of Definition 3526.4 and the diameter-tail stability of Definition 3526.5. They keep later endpoint reads attached to the same finite windows, regular readback, dyadic shrink comparisons, and transport and replay rows.

Finally apply the obligation package of Definition 3526.6. The package is exactly the displayed route consumed by downstream completion chapters, so no consumer obtains a separate selected center, quotient completion, countable-choice schedule, open-cover compactness theorem, or ambient real-completeness witness outside the packet. □

Theorem 2569.8 (Nested closed ball kernel scope binding)**.** *Every scoped read of an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N)$$

binds M *to* $\mathsf{MetricCompletion}^{\uparrow}$*,* K *to* $\mathsf{CompleteMetric}^{\uparrow}$*,* F *to* $\mathsf{CauchyFilter}^{\uparrow}$*,* S *to* $\mathsf{StreamName}^{\uparrow}$*,* R *to* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *to* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *to* $\mathsf{Real}^{\uparrow}$*, and* H, C, P, N *respectively to* BHist *transport by* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row. No scoped consumer can reinterpret one of these rows as an open-cover compactness object, a selected endpoint, a quotient completion, a countable-choice schedule, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallKernelScopeBinding`

Proof. Project the carrier of Definition 3526.1. Its displayed tuple already assigns the completion, complete-metric, filter, stream, regular-rational, dyadic, real-seal, transport, replay, provenance, and naming roles to the rows $M, K, F, S, R, D, E, H, C, P, N$.

Apply the obligation package of Definition 3526.6 and the consumer nonescape theorem Definition 3526.7. These results force every downstream completion, complete-metric, or real-facing read to consume the same displayed route before any endpoint is used.

Use the RegSeqRat extraction of Definition 3526.4 together with the diameter-tail stability of Definition 3526.5. They keep finite windows, center readback, dyadic shrink comparisons, and later tail reads attached to their original rows under hsame transport and Cont replay. Thus the scoped read binds the rows to the named kernel objects and has no coordinate for the excluded host-level objects. □

Theorem 2569.9 (Nested closed ball public NameCert export)**.** *The carrier of Definition 3526.1, the obligation package of Definition 3526.6, the consumer nonescape theorem Definition 3526.7, the RegSeqRat extraction of Definition 3526.4, the diameter-tail stability theorem Definition 3526.5, and the kernel scope binding of Definition 3526.8 assemble the public* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *export for the nested-ball completion route.*

The exported surface is exactly the finite route through $M, K, F, S, R, D, E, H, C, P, N$*. It refuses open-cover compactness, selected limits, quotient completions, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallPublicNameCertExport`

Proof. Start with the carrier and the obligation package. They identify the displayed rows $M, K, F, S, R, D, E, H, C$ as the metric-completion source, complete-metric service, Cauchy-filter row, stream windows, regular rational readback, dyadic diameter ledger, real seal, transport, replay, provenance, and local naming route.

Use consumer nonescape together with RegSeqRat extraction. Any completion-facing or real-facing read must pass through the finite windows, regular readback, dyadic shrink ledger, Cauchy-filter route, terminal seal, transport, replay, provenance, and local NameCert rows before it can use an endpoint.

Apply diameter-tail stability and kernel scope binding. Later ball reads keep the same containment and shrink comparisons under hsame transport and Cont replay, and each row remains bound to its displayed kernel role.

Thus the public export is the finite nested-ball completion route named by the local certificate. Since the carrier has no coordinate for open-cover compactness, selected limits, quotient completions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects is exported. □

Theorem 2569.10 (Nested closed ball public completion surface)**.** *Every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier exposes the finite nested-ball completion route as a public consumer surface through the rows*

$$M, K, F, S, R, D, E, H, C, P, N.$$

The surface is public exactly as a displayed completion interface: M names the metric-completion source, K the complete-metric service, F the nested Cauchy-filter row, S finite ball windows, R regular rational center readback, D dyadic diameter control, E the terminal real seal, and H, C, P, N the transport, replay, provenance, and local naming rows. It exposes no selected limit, no ambient completeness theorem, no quotient completion, no countable-choice schedule, and no open-cover compactness object.

Proof. Start with the public NameCert export of Definition 3526.9. It already packages the carrier, obligation package, nonescape theorem, center extraction, diameter-tail stability, and kernel scope binding as one public route.

Read the exported rows objectwise. The completion source, complete-metric service, nested filter, finite windows, regular readback, dyadic ledger, real seal, transport, replay, provenance, and local naming entries are exactly $M, K, F, S, R, D, E, H, C, P, N$.

Finally apply consumer nonescape and kernel scope binding. They keep every downstream completion or real-facing read on the displayed route, so the public surface exports only the finite nested-ball completion interface and none of the excluded host-level objects. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallPublicCompletionSurface`

Theorem 2569.11 (Nested closed ball separated uniqueness handoff)**.** *Let $B = (M, K, F, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier whose public completion surface is read through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *service row K. Any two endpoint reads obtained from the same accepted packet can be compared only through a* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *separated comparison route as in chapter 5983. The handoff uses the same nested-filter row F, stream windows S, regular readback R, dyadic diameter ledger D, terminal seal E, transport H, replay C, provenance P, and local name N; it exports no selected center, quotient completion point, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallSeparatedUniquenessHandoff`

Proof. Start with the carrier and obligation package of Definition 3526.1 and Definition 3526.6. They identify $M, K, F, S, R, D, E, H, C, P, N$ as the only rows available before a nested-ball endpoint can be read.

Use consumer nonescape and the public NameCert export, Definition 3526.7 and Definition 3526.9. These keep both endpoint reads inside the same displayed completion route and prevent either read from becoming a selected host center or quotient representative.

Now apply the public completion surface of Definition 3526.10. Since the surface exposes K only as the displayed $\mathsf{CompleteMetric}^{\uparrow}$ service row, any comparison between two endpoint reads must be delegated to the separated uniqueness route of chapter 5983. The structural rows H, C, P, N preserve the same packet identity, so the comparison cannot import ambient $\mathsf{Real}^{\uparrow}$ completeness or host equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NestedClosedBall}^{\uparrow}$ packet is generated from a MetricCompletion source row, a CompleteMetric service row, a CauchyFilter nested-ball row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore diameter

ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{NestedClosedBall}^{\uparrow}}$, publicClosed)

Scope closed: Definition 3526.9 and Definition 3526.10 bind the public nested-ball completion route over the displayed rows $M, K, F, S, R, D, E, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\mathsf{NestedClosedBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient completions, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification and bridge checking require the public completion surface to be mirrored by a checked target and a standard bridge.

Chapter 2570

A Concrete Naming Certificate for $\mathsf{RealBoundedInterval}^{\uparrow}$

Bishop–Bridges constructive real analysis treats a bounded interval by endpoint evidence and finite approximation data rather than by an ambient compact set. The $\mathsf{RealBoundedInterval}^{\uparrow}$ packet records that endpoint interval as finite BEDC NameCert data over the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic endpoint-order ledger of chapter 291, the located endpoint route of chapter 1709, and the bracketing surface of chapter 1807. It names a closed finite endpoint packet for downstream Real and Completion consumers; it is not open-cover compactness, a Heine–Borel theorem, a selected point of the interval, a quotient interval, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2570.1 (Real bounded interval carrier)**.** *A* $\mathsf{RealBoundedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, S, T, R, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint* $\mathsf{Real}^{\uparrow}$ *seal rows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order ledger recording the displayed finite evidence that the lower endpoint precedes the upper endpoint at the requested tolerance.*
3. *S and T are finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the endpoint names are observed before any interval read is exported.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the paired endpoint windows.*
5. *Q is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row tying the endpoint seals, dyadic order ledger, and readback route to a located interval surface.*
6. *B is the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *row giving the finite bracket consumed by Real and Completion-facing readers.*
7. *H records componentwise* hsame *transport for endpoint seals, endpoint-order evidence, windows, readback, located interval, bracket, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded-interval request through $L, U, D, S, T, R, Q, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{FiniteBracketingInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, Heine–Borel compactness, selected points, quotient intervals, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealBoundedIntervalUp.RealBoundedIntervalTasteGate single carrier alignment`

Theorem 2570.2 (Real bounded interval NameCert obligations)**.** *For every accepted* $\mathsf{RealBoundedInterval}^{\uparrow}$ *carrier*

$$I = (L, U, D, S, T, R, Q, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the two endpoint* $\mathsf{Real}^{\uparrow}$ *seals,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order evidence, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{LocatedRealInterval}^{\uparrow}$ *located endpoint row,* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Every bounded-interval consumer must therefore read $L, U, D, S, T, R, Q, B, H, C, P, N$ in the displayed route, and no obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3527.1. The projection exposes exactly the two endpoint seals, the endpoint-order ledger, the two finite windows, readback, located interval, bracket, and structural rows.

The endpoint-order obligation is read from D. The finite-window obligation is read from S, T, and the regular rational readback obligation is read from R before either the located interval row Q or the bracket row B is available.

Componentwise transport is supplied by H, and continuation replay is supplied by C. The provenance and local naming rows P, N attach the same finite route to the packet, so there is no separate host interval, quotient interval, selected point, compactness, or completeness coordinate from which a consumer could read additional data. □

Theorem 2570.3 (Real bounded interval endpoint seal route)**.** *Every endpoint read of an accepted* $\mathsf{RealBoundedInterval}^{\uparrow}$ *packet factors through the paired* $\mathsf{Real}^{\uparrow}$ *seals L, U, the endpoint-order ledger D, the finite windows S, T, and the regular rational readback R before the located interval row Q and bracket row B are consumed. The route exports no host real equality, quotient interval equality, selected endpoint representative, trichotomy row, open-cover compactness proof, Heine–Borel theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an endpoint-facing consumer and project the route C. The only endpoint rows are L, U, and the only order witness attached to them is the dyadic ledger D.

The consumer then reads the finite windows S, T and the regular rational readback R. Those rows provide the displayed approximation surface needed by the located interval and bracketing rows.

Finally Q and B package the endpoint surface for interval readers. Since the carrier contains no host equality, quotient interval, selected representative, compactness theorem, or completeness coordinate, the endpoint seal route is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealBoundedInterval}^{\uparrow}$ packet is generated from two RealUp endpoint seals, a DyadicRatCore endpoint-order ledger, paired StreamName finite windows, RegSeqRat readback, LocatedRealInterval and FiniteBracketingInterval rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3527.1, Definition 3527.2, and Definition 3527.3 seed the finite endpoint interval route over real seals, dyadic order evidence, finite windows, regular readback, located interval rows, and bracketing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBoundedInterval}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Heine–Borel compactness, selected interval points, quotient intervals, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order evidence, finite-window readback, located interval handoff, bracketing handoff, endpoint seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealBoundedIntervalUp.

Chapter 2571

A Concrete Naming Certificate for $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$

Following Stieltjes, Jordan, and the Bishop–Bridges constructive reading of bounded variation, this packet records only the finite boundary by which a $\mathsf{BoundedVariation}^{\uparrow}$ integrator supplies interval-jump rows to $\mathsf{RiemannStieltjes}^{\uparrow}$ and $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ consumers. It sits after the partition-variation ledger of chapter 398, the tagged-sum handoff of chapter 1637, the regulated Cauchy readback of chapter 2351, the regular rational surface of chapter 301, the dyadic endpoint core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite interval-jump boundary; it is not a variation measure, a Lebesgue integration object, a quotient of host functions, a selected limit, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2571.1 (Stieltjes measure boundary carrier)**.** *A* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, \Pi, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row naming the integrator only through displayed finite partition-variation data.*
2. *Π is the displayed interval partition, including ordered endpoints, interval cells, and the refinement witnesses on which boundary reads are requested.*
3. *J is the endpoint-jump* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, recording the finite left and right endpoint differences attached to the cells of Π.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes those jump rows at the requested rational windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, Π, J, R have been displayed.*
6. *H records componentwise* hsame *transport for the bounded-variation source, partition, jump ledger, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from bounded-variation partition reads to interval jumps, from jumps to regular rational readback, and from readback to the* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *P is the* Pkg *provenance over* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, \Pi, J, R, E, H, C, P, N$. It has no coordinate for measure completion, Lebesgue integration, a host function quotient, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.StieltjesMeasureBoundaryUp.StieltjesMeasureBoundaryTasteGate single carrier alignment`

Theorem 2571.2 (Stieltjes measure boundary NameCert obligations)**.** *For every accepted* StieltjesMeasureBoundar *carrier*

$$S = (B, \Pi, J, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{BoundedVariation}^{\uparrow}$ *source, displayed interval-partition exposure through Π, endpoint-jump exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger J,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3528.1. The source row B is already the finite partition-variation object of chapter 398, so the boundary packet begins from displayed partition data rather than from a completed variation measure.

The interval and jump obligations are the rows Π and J. The partition cells provide the only endpoint pairs that may be inspected, and the dyadic ledger records the corresponding finite endpoint differences through chapter 291.

The readback and seal obligations are downstream. The regular rational route of chapter 301 consumes J through R, and the real-name route of chapter 38 consumes R through E. Transport, replay, provenance, and local naming are exactly H, C, P, N, leaving no hidden measure, quotient, choice, selected-limit, or completeness row. □

Theorem 2571.3 (Stieltjes boundary bounded-variation handoff)**.** *Every* $\mathsf{RiemannStieltjes}^{\uparrow}$ *or* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *packet receives the integrator boundary only through the displayed route*

$$\mathsf{BoundedVariation}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}.$$

Concretely, the consumer must read the bounded-variation source B, the interval partition Π, the endpoint-jump ledger J, and the regular rational readback R before any $\mathsf{Real}^{\uparrow}$ *seal E is visible.*

Proof. Start with the bounded-variation source. By chapter 398, that source exposes partition data and dyadic variation ledgers, not a variation measure or a host function quotient.

The boundary carrier refines the source to endpoint jumps on the displayed partition. The cells of Π determine the finite list of endpoint pairs, and the ledger J records their dyadic jump reads.

Now pass to consumers. The $\mathsf{RiemannStieltjes}^{\uparrow}$ route of chapter 1637 and the $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ route of chapter 2351 may consume only the displayed regular readback and seal rows R, E together with H, C, P, N. Thus the handoff is the listed finite route, with no channel for measure completion, Lebesgue integration, selected limits, countable choice, or ambient real completeness. □

Theorem 2571.4 (Stieltjes boundary Real seal non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* StieltjesMeasureBoundar *carrier is available only after the bounded-variation source row, displayed interval partition, endpoint-jump* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been read. The seal exposes the same*

finite interval-jump boundary and exports no measure completion, Lebesgue integration object, host function quotient, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the seal coordinate of Definition 3528.1. The row E is downstream from B, Π, J, R and is transported or replayed only by the structural rows H and C.

Apply Definition 3528.2. It identifies the bounded-variation source, partition exposure, dyadic jump ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows as the complete local certificate surface.

Apply Definition 3528.3. Every integration-facing read must pass through the same finite jump route before the seal is visible. Since none of the excluded objects is a carrier coordinate, the $\mathsf{Real}^{\uparrow}$ seal cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ packet is generated from a BoundedVariation source row, a displayed interval partition, endpoint-jump DyadicRatCore ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3528.1, Definition 3528.2, Definition 3528.3, and Definition 3528.4 seed the finite Stieltjes boundary by which bounded-variation interval jumps are read by Riemann–Stieltjes and regulated Cauchy integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, Lebesgue integration, host function quotients, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-jump exactness, regular readback, bounded-variation handoff, Real-seal non-escape, transport and replay stability, provenance non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.StieltjesMeasureBoundaryUp.

Chapter 2572

A Concrete Naming Certificate for BanachContraction$^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the BanachContraction$^{\uparrow}$ packet records the fixed-point route as finite displayed data from the contraction map and Picard orbit surfaces to the complete-metric endpoint boundary. It consumes ContractionMapping$^{\uparrow}$ from chapter 1418, PicardIteration$^{\uparrow}$ from chapter 2359, MetricCompletion$^{\uparrow}$ from chapter 1506, and SeparatedMetric$^{\uparrow}$ from chapter 1411. The packet also reads the complete-metric row of chapter 128, regular rational readback through chapter 301, and the terminal real-name boundary of chapter 38; it does not claim quotient completions, selected representatives, host function extensionality, or ambient Real$^{\uparrow}$ completeness.

Definition 2572.1 (BanachContraction carrier)**.** *A* BanachContraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, I, M, Q, S, E, H, C, P, N)$$

whose coordinates are exactly:

1. K*, the accepted* ContractionMapping$^{\uparrow}$ *row displaying the self-map graph, contraction bound, one-step Picard comparison, and fixed-point readiness surface;*
2. I*, the* PicardIteration$^{\uparrow}$ *row displaying the finite orbit, residual-distance ledger, Cauchy-modulus extraction, and regular-sequence handoff;*
3. M*, the* CompleteMetric$^{\uparrow}$ *row, read only as the endpoint-admission surface for the Picard Cauchy data;*
4. Q*, the regular-Cauchy and readback route through* RegSeqRat$^{\uparrow}$*,* MetricCompletion$^{\uparrow}$*, and the terminal* Real$^{\uparrow}$ *seal used by endpoint-facing consumers;*
5. S*, the* SeparatedMetric$^{\uparrow}$ *uniqueness row, consumed only after the two endpoint reads expose zero distance through the displayed metric row;*
6. E*, the endpoint seal row naming the fixed-point candidate as the result of the completion route rather than as a selected host point;*
7. H*, componentwise* hsame *transport for contraction data, Picard iterates, complete-metric endpoint rows, readback rows, separatedness rows, endpoint seals, replay rows, provenance, and local naming;*

8. *C, displayed* Cont *replay from contraction admission through Picard iteration, Cauchy readback, complete-metric endpoint admission, separated uniqueness, and endpoint sealing;*
9. *P, the* Pkg *provenance row over* $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N, the local* $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$ *row.*

The carrier names the constructive Banach contraction route. It has no coordinate for quotient completion equality, selected representatives, host fixed-point equality, host function extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BanachContractionUp.BanachContractionTasteGate single carrier alignment`

Theorem 2572.2 (BanachContraction NameCert obligations)**.** *For an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier $B = (K, I, M, Q, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$ *surface consists exactly of contraction-map admission, Picard-iteration admission, complete-metric endpoint visibility, regular-Cauchy and Real readback visibility, separated uniqueness visibility, endpoint-seal stability, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 3529.1.*

Proof. Project the carrier. The rows K, I, M, Q, S, E are the only mathematical coordinates, and the rows H, C, P, N are the only structural coordinates available to the local certificate.

The contraction and Picard obligations come from chapter 1418 and chapter 2359. They expose the self-map graph, contraction bound, finite orbit ledger, residual comparisons, and Cauchy-modulus handoff before any endpoint row is inspected.

The completion and readback obligations come next. The complete-metric row M admits the Cauchy data to an endpoint-facing surface, while Q records the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback path used by consumers of that endpoint.

Finally S, E, H, C, P, N bind uniqueness, endpoint sealing, transport, replay, provenance, and naming to the same displayed rows. Since no excluded object is a coordinate of B, the NameCert surface exports only the constructive Banach contraction packet. □

Theorem 2572.3 (BanachContraction Cauchy orbit route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier first consumes the contraction row K and the Picard row I, then the complete-metric row M, then the readback route Q through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and only afterward the endpoint seal E. Thus the Cauchy iteration route is the displayed finite chain*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto (\mathsf{RegSeqRat}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow}) \longmapsto E.$$

No read may replace this chain by quotient completion equality, a selected limit representative, countable choice, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with K. The contraction row supplies only the finite self-map graph, rational contraction bound, one-step Picard comparison, and fixed-point readiness boundary.

Next consume I. The Picard row generates the finite orbit ledger, residual-distance comparisons, Cauchy-modulus extraction, and regular handoff, so the read has Cauchy data before it asks for an endpoint.

The complete-metric row M is then the only endpoint-admission coordinate. The readback row Q records how the admitted endpoint is inspected through regular rational windows, the metric-completion surface, and the terminal real seal.

The endpoint seal E is downstream of those rows. Because the carrier includes no quotient completion, representative selector, choice schedule, host function equality, or ambient real-completeness coordinate, the displayed route is the whole Cauchy iteration route. □

Theorem 2572.4 (BanachContraction Picard tail modulus)**.** *Every accepted* $\mathsf{BanachContraction}^{\uparrow}$ *orbit read must pass through the* $\mathsf{PicardIteration}^{\uparrow}$ *finite orbit row, residual-distance row,* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{CompleteMetric}^{\uparrow}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the unique fixed-point seal can be read. Equivalently, the fixed-point consumer may inspect the separated uniqueness row only after the Picard tail modulus has been routed through the displayed regular rational completion boundary.*

Proof. Apply Definition 3529.3. It forces every endpoint-facing read through the contraction row, Picard row, complete-metric row, readback route, and endpoint seal before any fixed-point endpoint is consumed.

Now read the Picard row through Definition 2359.5. The finite orbit ledger and residual-distance row determine the $\mathsf{CauchyModulus}^{\uparrow}$ row, and that row is the source of the $\mathsf{RegSeqRat}^{\uparrow}$ handoff exposed to completion-facing consumers.

Finally apply Definition 3529.5. The separated uniqueness seal is downstream of the endpoint route, so it cannot be read before the Picard finite orbit, residual-distance ledger, modulus handoff, regular rational boundary, complete-metric row, and terminal real seal have all been displayed. □

Theorem 2572.5 (BanachContraction unique fixed-point seal)**.** *For an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier* $B = (K, I, M, Q, S, E, H, C, P, N)$*, uniqueness of the fixed-point endpoint is exported only through the separated zero-distance route* S *after both compared endpoint reads have passed through* K, I, M, Q, E *and the structural rows* H, C, P, N*. The exported uniqueness is classifier equality inside the displayed* $\mathsf{SeparatedMetric}^{\uparrow}$ *surface, not host equality of points, quotient equality of completion names, a selected representative, trichotomy, or ambient completeness.*

Proof. Take two endpoint-facing reads of the same carrier. By Definition 3529.3, each read must expose the contraction row, Picard row, complete-metric row, readback route, and endpoint seal before any uniqueness comparison is available.

Apply the separatedness row S. It compares the two endpoint reads only after the displayed zero-distance witness is routed through the $\mathsf{SeparatedMetric}^{\uparrow}$ surface; this is the same kind of classifier route isolated in chapter 1411.

Transport and replay are componentwise through H and C, while P and N record provenance and local naming for the same finite packet. These rows do not add a host equality, quotient equality, representative selector, trichotomy principle, or ambient completeness theorem.

Therefore the fixed-point seal is unique only in the displayed local classifier sense. The carrier records the constructive endpoint route and its separated uniqueness boundary, not a host-level fixed-point object. □

Falsifiable prediction: Every accepted $\mathsf{BanachContraction}^{\uparrow}$ read factors through one ContractionMapping row, one PicardIteration row, one CompleteMetric row, one RegSeqRat/MetricCompletion/Real readback route, one SeparatedMetric uniqueness row, one endpoint seal, hsame transport, Cont replay, Pkg provenance, and NameCert row before any fixed-point endpoint is consumed.

Independence witness: contractionmapping, picard-iteration, completemetric, metriccompletion, separatedmetric, regseqrat, real: none is bijective with $\mathsf{BanachContraction}^{\uparrow}$ because the packet jointly stores the contraction route, Picard Cauchy iteration, completion endpoint admission, readback route, separated uniqueness seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachContraction}^{\uparrow}$ packet is generated from a ContractionMapping row, a PicardIteration row, a CompleteMetric endpoint row, a RegSeqRat/MetricCompletion/Real readback route, a SeparatedMetric uniqueness row, an endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3529.1, Definition 3529.2, Definition 3529.3, and Definition 3529.5 seed the finite constructive Banach contraction fixed-point route over the existing contraction, Picard, completion, readback, and separatedness packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host fixed-point equality, host function extensionality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-map visibility, Picard-iteration visibility, complete-metric endpoint visibility, readback exactness, separated uniqueness routing, endpoint-seal stability, transport, replay, provenance, and local naming.

Chapter 2573

A Concrete Naming Certificate for $\mathsf{PeanoExistence}^{\uparrow}$

Following Peano and Bishop and Bridges, the $\mathsf{PeanoExistence}^{\uparrow}$ packet records constructive local ODE existence data without uniqueness as finite NameCert rows. It consumes the real endpoint discipline of chapter 38, the continuous vector-field surface of chapter 130, and the displayed ODE source of chapter 193. The packet names an interval-window, vector-field, Euler-polygon, and compact-containment route; it does not name a Lipschitz modulus, a unique flow, a maximal solution, host compactness, or a choice-selected path.

Definition 2573.1 (Peano existence carrier)**.** *A* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, V, W, E, Q, M, H, C, G, N)$$

with the following displayed rows:

1. *I is the interval source row, binding the initial time, initial state, local radius, and finite rational time window through the* $\mathsf{Ode}^{\uparrow}$ *source surface.*
2. *V is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row, read on the displayed time-state window and classified only through its continuous-map graph and modulus ledgers.*
3. *W is the closed* $\mathsf{Real}^{\uparrow}$ *interval and state-window row that bounds the Euler-polygon construction before any solution read is exported.*
4. *E is the finite Euler polygon approximation ledger, including step windows, endpoint approximants, and local slope reads from V.*
5. *Q is the compact-containment row, recording that every displayed polygon segment remains in the certified window W.*
6. *M is the modulus row used to compare nearby polygon windows and to pass from finite polygon data to a Cauchy-style local solution row.*
7. *H records componentwise* hsame *transport for interval sources, vector-field rows, real windows, polygon ledgers, containment rows, modulus rows, replay rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from the ODE request through I, V, W, E, Q, M.*
9. *G is the* Pkg *provenance over* $\mathsf{PeanoExistence}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for uniqueness, Lipschitz dependence, a selected maximal solution, ambient host compactness, or a choice principle.

Lean4 ✓ `BEDC.Derived.PeanoExistenceUp.TasteGate.PeanoExistenceTasteGate single carrier alignment`

Theorem 2573.2 (Peano existence NameCert obligations)**.** *For every accepted* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier*

$$P = (I, V, W, E, Q, M, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval source row, continuous vector-field row, real closed window row, Euler polygon approximation ledger, compact-containment row, modulus row, componentwise transport, displayed continuation replay, package provenance, and local naming. Every local existence read is therefore routed through* $I, V, W, E, Q, M, H, C, G, N$.

Proof. Project the accepted carrier of Definition 3530.1. The interval source I fixes the ODE request, V supplies the displayed continuous vector-field read, and W supplies the real closed window in which the construction is inspected.

The approximation content is carried by E, Q, M. Euler polygons in E provide finite stepwise approximants, Q keeps every inspected polygon segment inside the recorded window, and M gives the finite comparison ledger needed for a Cauchy-style local solution read.

The structural rows H, C, G, N transport, replay, package, and name exactly that finite route. Since the carrier has no coordinate for uniqueness, Lipschitz dependence, maximal-extension selection, host compactness, or choice, the NameCert surface cannot export any of those objects. □

Definition 2573.3 (Peano interval trap carrier)**.** *The Peano interval trap inside a* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier is the displayed subpacket*

$$(I, V, W, E, Q, M)$$

consisting of the closed $\mathsf{Real}^{\uparrow}$ *time-state window named by I and W, the continuous vector-field row V, the Euler polygon ledger E, the compact-containment row Q, and the modulus comparison row M. It records only the finite data used to keep polygon approximants inside the source packet before a local solution row is exported.*

Theorem 2573.4 (Peano interval trap obligations)**.** *Every accepted local-existence read of a* $\mathsf{PeanoExistence}^{\uparrow}$ *packet must pass through the interval source I, the continuous vector-field row V, the closed* $\mathsf{Real}^{\uparrow}$ *time-state window W, the Euler polygon ledger E, the compact-containment row Q, and the modulus handoff M before it can export a solution row. Thus the interval trap obligations are exactly the finite rows displayed in Definition 3530.3, together with the transport, replay, provenance, and local naming rows of Definition 3530.1.*

Proof. Begin with the carrier of Definition 3530.1. The interval source I and vector-field row V are the only rows that provide the ODE request and continuous slope reads, while W is the only closed Real window in which the finite construction is inspected.

The Euler polygon ledger E gives the finite approximants. The compact-containment row Q checks each displayed segment against the same source window, and the modulus row M supplies the comparison handoff needed before a local solution read can be exported.

Rows H, C, G, N do not add another analytic source. They transport, replay, record provenance, and name the same interval trap. Therefore every accepted local-existence read must expose I, V, W, E, Q, M before the solution row is public. □

Theorem 2573.5 (Peano existence scoped primitive route)**.** *Every accepted local-existence read of a* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier factors through the exact primitive route*

$\mathsf{Ode}^{\uparrow}$ *interval source* $\to$ $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row* $\to$ $\mathsf{Real}^{\uparrow}$ *closed window* $\to$ *Euler polygon ledger* $\to$ *compac*

Thus the scoped read exposes precisely the interval source, vector-field, closed-window, polygon, containment, modulus, transport, replay, provenance, and local naming rows of Definition 3530.1. It exports no Lipschitz uniqueness, selected maximal solution, host compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier $P = (I, V, W, E, Q, M, H, C, G, N)$ of Definition 3530.1. The only analytic entrance rows are the $\mathsf{Ode}^{\uparrow}$ interval source I, the $\mathsf{ContinuousMap}^{\uparrow}$ vector-field row V, and the closed $\mathsf{Real}^{\uparrow}$ window W.

Apply Definition 3530.2. It identifies the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ surface with the displayed carrier rows and records that every local-existence read is routed through $I, V, W, E, Q, M, H, C, G, N$.

Apply Definition 3530.4. The finite solution-producing part of the route is exactly the Euler polygon ledger E, compact-containment row Q, and modulus handoff M before transport, replay, provenance, and local naming are consumed.

Since the carrier definition has no coordinate for Lipschitz uniqueness, a selected maximal solution, host compactness, countable choice, or ambient Real completeness, none of those objects can be exported by the scoped primitive route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PeanoExistence}^{\uparrow}$ packet is generated from an $\mathsf{Ode}^{\uparrow}$ interval source row, a $\mathsf{ContinuousMap}^{\uparrow}$ vector-field row, a closed $\mathsf{Real}^{\uparrow}$ time-state window, Euler polygon approximation ledgers, compact-containment and modulus rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 3530.1, Definition 3530.2, Definition 3530.3, and Definition 3530.4 expose the constructive local existence obligations over displayed ODE, continuous-map, real-window, Euler-polygon, containment, modulus, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim uniqueness, Lipschitz dependence, selected maximal solutions, host compactness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding these obligations to their exact primitive routes for interval sources, continuous vector fields, closed Real windows, Euler polygon ledgers, compact-containment rows, modulus handoffs, transport, replay, provenance, and local naming.

Chapter 2574

A Concrete Naming Certificate for $\mathsf{LocatedCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges on located constructive metric completions and Kelley on functorial completion discipline, the $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet records the finite route by which a located metric-space presentation is sent to its completion-facing regular-name surface. It sits after the located real boundary of chapter 1933, the metric-completion carrier of chapter 1506, the universal-property witness of chapter 2319, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the displayed functorial handoff used by L10 completion consumers; it is not a quotient completion theorem, not a selected host limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2574.1 (Located completion functor carrier)**.** *A* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{LocatedCompletionFunctor}(L, M, U, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *or located metric-source row exposing the positive-apartness and approximation data that make Cauchy requests decidable as finite located reads.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row naming the completion-facing object surface, without quotienting Cauchy names.*
3. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row recording the displayed universal-property handoff for uniformly continuous maps already routed through finite data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the completion object is presented through regular rational names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located, completion, universal-property, and regular rational rows have been displayed.*
6. *F is the functorial action row, naming how source maps are transported to completion-facing maps through L, M, U, R, E.*
7. *H records componentwise* hsame *transport for the located source, completion carrier, universal-property handoff, regular rational readback, Real seal, functorial action, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a located source request through the completion, universal-property, regular rational, Real-seal, and functorial-action rows.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, M, U, R, E, F, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, countable-choice schedules, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge data.

Lean4 ✓ `BEDC.Derived.LocatedCompletionFunctorUp.LocatedCompletionFunctorTasteGate single carrier alignment`

Theorem 2574.2 (Located completion functor NameCert obligations)**.** *For every accepted* LocatedCompletionFunc *carrier* $\mathsf{LocatedCompletionFunctor}(L, M, U, R, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the located source, metric-completion carrier, Cauchy-completion universal-property handoff, regular rational readback, Real seal, functorial action, componentwise transport, displayed replay, provenance, and local naming rows.*

Every completion-facing consumer must read the route

$$L, M, U, R, E, F, H, C, P, N$$

before treating a located map as a completion map. The route exports no quotient completion theorem, selected host limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge datum.*

Proof. Carrier admission is projection from Definition 3531.1. The accepted tuple displays the located source row, completion carrier, universal-property row, regular rational readback, Real seal, functorial action, and structural support.

The completion-facing read begins with L and M. These rows expose located input data and the completion object surface before any map action is available.

Next read U, R, E, F. The universal-property row supplies the finite handoff discipline, the regular rational and Real rows name the target readback boundary, and the functorial action row records the transported map only after those rows are present.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since no quotient equality, selected limit, choice schedule, trichotomy row, host real equality, Real completeness theorem, Lean verification row, or bridge datum is present in the carrier, none can become a local NameCert obligation. □

Theorem 2574.3 (Located completion functor Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ *packet reaches the Real-facing handoff only through the displayed located source row, metric-completion row, Cauchy-completion universal-property row, regular rational readback row, Real seal row, and functorial-action row of Definition 3531.1. In particular, the handoff is a finite NameCert route for located completion data, not a quotient completion, selected host limit, countable-choice construction, trichotomy principle, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. The consumer-facing evidence is the displayed carrier, so the located source, completion object, universal-property handoff, readback, seal, and functorial action are the only rows it may inspect.

The regular rational and Real rows are downstream. The consumer reaches R and E only after L, M, U have fixed the located completion route and before F packages the transported map.

The support rows H, C, P, N carry the same route through transport, replay, provenance, and local naming. No coordinate supplies quotient completion equality, a selected limit, countable choice, trichotomy, host equality, ambient Real completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet factors a Real-facing completion read through one located source row, one MetricCompletion row, one CauchyCompletionUniversalProperty handoff, one RegSeqRat readback, one RealUp seal, one functorial-action row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real, metric-completion, cauchy-completion-universal-property, regseqrat, real: $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located source, completion carrier, universal-property handoff, regular rational readback, Real seal, and functorial action as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3531.1, Definition 3531.2, and Definition 3531.3 seed the finite located-completion functor route over located source rows, metric-completion carriers, universal-property handoffs, regular rational readback, Real seals, functorial action, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-source exactness, completion-carrier handoff, universal-property consumption, RegSeqRat and Real handoff, functorial-action exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionFunctorUp.

Constructive story: A $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet is generated from a located source row, a MetricCompletion row, a CauchyCompletionUniversalProperty handoff, a RegSeqRat readback row, a RealUp seal row, a functorial-action row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2575

A Concrete Naming Certificate for $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ packet records the finite surface at which a located regular rational sequence exposes the modulus windows consumed by Phase Real and Completion classifiers. It sits after the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the located Cauchy representation route of chapter 4070, and the terminal real-name seal of chapter 38. The packet is a carrier for displayed located-modulus data, not a quotient completion, a selected limit, a countable-choice schedule, a trichotomy oracle, or an ambient completeness principle.

Definition 2575.1 (RegSeqRat located-modulus carrier)**.** *A* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, L, \mu, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance query row.*
2. *L is the* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *located regular-sequence row.*
3. *μ is the displayed modulus row assigning finite stream windows to the tolerance queries admitted by Q.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed by μ.*
5. *D is the dyadic decision ledger binding the tolerance query to the displayed window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row at the window named by W.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached after the located regular-sequence readback.*
8. *H stores componentwise* hsame *transport for $Q, L, \mu, W, D, R, E, C, P, N$.*
9. *C stores displayed* Cont *replay from tolerance request to modulus window, regular readback, locatedness row, and real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $Q, L, \mu, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegSeqRatLocatedModulusUp.TasteGate.RegSeqRatLocatedModulusTasteGate single carrier alignment`

Theorem 2575.2 (RegSeqRat located-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegSeqRatLocatedMod}$ *carrier $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, located-modulus exposure, finite-window stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger refusal for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3532.1. The accepted packet first exposes the tolerance query Q, the located regular-sequence row L, and the displayed modulus row μ.

The modulus row opens only the finite $\mathsf{StreamName}^{\uparrow}$ window W and the dyadic decision ledger D attached to that request. The rational readback row R is then read at the displayed window, and the terminal real-name seal E is reached only after this finite route has been replayed.

Transport and continuation are exhausted by H and C, while P and N record provenance and local naming. Since the carrier has no further coordinate, no accepted packet can import quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2575.3 (RegSeqRat located-modulus finite-window stability)**.** *Every* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *packet factors through $Q, L, \mu, W, D, R, E, H, C, P, N$ before any consumer-visible located read is exported. The handoff therefore exposes the located modulus as a finite window discipline over regular rational readback and terminal real-name sealing, not as quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier as $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$. A $\mathsf{LocatedRegSeqRat}^{\uparrow}$ consumer must read the located row L together with the displayed modulus row μ before it can export a locatedness response.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches rational data only through the finite window W named by μ and the readback row R bound to that window. The dyadic ledger D records which tolerance request was served, so the read cannot be replaced by an unbounded stream equality or by an arbitrary limit choice.

A $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal E only after the same window and readback route has been replayed by C and transported by H. The remaining rows P and N certify provenance and local naming, so the handoff contains no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness. □

Theorem 2575.4 (RegSeqRat located-modulus located row exhaustion)**.** *For an accepted* $\mathsf{RegSeqRatLocatedModul}$ *carrier $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$, every located read exported to* $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *is exhausted by exactly the located row L, the displayed modulus row μ, the finite windows W, the dyadic decision ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal E, the componentwise transport row H, the displayed replay row C, the provenance row P, and the local naming row N. No exported located read factors through quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 3532.1. The located export has no coordinates other than $Q, L, \mu, W, D, R, E, H, C, P, N$, and its consumer-visible located row is L together with the displayed modulus row μ.

Use the finite-window stability theorem Definition 3532.3. A $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ consumer reads L and μ before exporting locatedness; a $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches data through W, D, R; and a $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal E only after transport and replay through H and C.

The remaining rows P and N source and name this same route. Since the carrier definition contains no quotient stream equality, selected-limit representative, countable-choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness witness, the listed rows exhaust every exported located read. □

Theorem 2575.5 (RegSeqRat located-modulus Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *carrier* $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$ *passes in order through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance query* Q*, the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *row* L*, the displayed modulus row* μ*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*, the dyadic decision ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* NameCert *row* N*. No* $\mathsf{Real}^{\uparrow}$*-facing read bypasses this route by quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness.*

Proof. Project the carrier from Definition 3532.1. The only Real-facing rows in the packet are the displayed tolerance, locatedness, modulus, window, ledger, readback, seal, transport, replay, provenance, and naming rows.

Apply the finite-window stability theorem Definition 3532.3 to force every $\mathsf{Real}^{\uparrow}$ consumer through $Q, L, \mu, W, D, R, E, H, C, P, N$ before a consumer-visible seal is exported.

Apply the located-row exhaustion theorem Definition 3532.4. It rules out additional coordinates for quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness, so the displayed order is the full Real route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance query, a LocatedRegSeqRat row, a displayed modulus row, StreamName finite windows, a dyadic decision ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3532.1, Definition 3532.2, Definition 3532.3, and Definition 3532.5 seed the finite located-modulus route from regular rational sequence data to Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-modulus exposure, finite-window stability, RegSeqRat readback, RealUp seal routing, and ledger refusal for BEDC.Derived.RegSeqRatLocatedModulusUp.

Chapter 2576

A Concrete Naming Certificate for $\mathsf{MorseTheory}^{\uparrow}$

Following Milnor on Morse theory, the $\mathsf{MorseTheory}^{\uparrow}$ packet records the finite certificate surface by which critical-point data, Hessian signatures, local chart rows, gradient-flow incidence, and filtration boundary rows are read as one displayed Morse-theory name. It sits after the charted carrier of chapter 96, the derivative observation rows of chapter 122, the scalar comparison seal of chapter 38, the tangent-fibre bookkeeping of chapter 210, and the fibrewise metric rows of chapter 133. The packet names a local finite Morse certificate; it is not a host smooth-function space, not a Sard or transversality theorem, not a handle-decomposition completeness theorem, and not a cobordism classification principle.

Definition 2576.1 (Morse theory carrier)**.** *A* $\mathsf{MorseTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, D, R, T, G, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *charted source row naming the local carrier on which the Morse data are read.*
2. *F is the displayed function row, exposing only the finite function-name observations used by the certificate.*
3. *D is the* $\mathsf{Derivative}^{\uparrow}$ *first- and second-derivative observation row attached to F over the charted source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar comparison row used for critical values, local height comparisons, and finite ordering tests.*
5. *T is the* $\mathsf{VectorBundle}^{\uparrow}$ *tangent-fibre row over X, carrying the finite source for Hessian and flow-incidence reads.*
6. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *metric row used to read gradient-flow incidence only after X, T, D have been displayed.*
7. *L is the local Morse ledger row: critical-point admission, nondegenerate Hessian signature, index tag, finite flow-incidence boundary, and filtration boundary are recorded here as finite observations.*

8. *H records componentwise* hsame *transport for the chart, function, derivative, scalar, tangent-fibre, metric, ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through chart admission, derivative readback, real comparison, tangent-fibre transport, metric readback, and Morse ledger inspection.*
10. *P is the* Pkg *provenance over* $\mathsf{MorseTheory}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$ *row.*

The carrier exports Morse-theory reads only through $X, F, D, R, T, G, L, H, C, P, N$. It has no coordinate for a host function space, global transversality, Sard regular-value selection, complete handle decomposition, cobordism classification, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.MorseTheoryUp.MorseTheoryTasteGate single carrier alignment`

Theorem 2576.2 (Morse theory NameCert obligations)**.** *For every accepted* $\mathsf{MorseTheory}^{\uparrow}$ *carrier*

$$M = (X, F, D, R, T, G, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$ *surface consists exactly of charted manifold admission, displayed function rows, derivative readback, real scalar comparison, tangent-fibre bookkeeping, Riemannian metric readback, local Morse ledger inspection, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every critical-point, index, or flow-incidence read exported by the packet must factor through

$\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Proof. Project the accepted carrier. The charted source is row X, the displayed function observations are row F, derivative readback is row D, real comparison is row R, tangent-fibre bookkeeping is row T, and metric readback is row G.

The Morse-specific content is confined to row L. It records critical-point admission, nondegenerate Hessian signature, index tag, finite flow-incidence boundary, and filtration boundary only after the chart, derivative, scalar, tangent, and metric rows are available.

The structural rows H, C, P, N transport, replay, source, and name exactly that displayed route. Since no row stores Sard regular values, transversality, complete handle decompositions, cobordism classification, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{MorseTheory}^{\uparrow}$ critical-point, index, or flow-incidence read factors through displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, provenance, transport, replay, and local NameCert rows.

Independence witness: manifold, derivative, real, vectorbundle, riemannianmetric: the Manifold packet records charted carriers without Morse functions, the Derivative packet records derivative observations without critical-index ledgers, the Real packet records scalar comparison without tangent flow incidence, the VectorBundle packet records fibres without a fibrewise metric read, and the RiemannianMetric packet records metric rows without Morse ledger data. None is bijective to $\mathsf{MorseTheory}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MorseTheory}^{\uparrow}$ packet is generated from Manifold chart rows, displayed function observations, Derivative first- and second-derivative reads, Real scalar comparisons, VectorBundle tangent-fibre rows, RiemannianMetric

metric reads, a local Morse ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MorseTheory}\uparrow}$, seed)

Scope closed: Definition 3533.1 and Definition 3533.2 seed the finite Morse-theory certificate over chart, function, derivative, scalar, tangent, metric, ledger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MorseTheory}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Sard theory, transversality, handle decomposition completeness, cobordism classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-point ledger exposure, Hessian signature exactness, index tagging, finite flow-incidence routing, filtration boundary routing, and a Lean carrier/obligation formalization for BEDC.Derived.MorseTheoryUp.

Chapter 2577

A Concrete Naming Certificate for $\mathsf{HyperbolicHorocycle}^{\uparrow}$

Following classical hyperbolic geometry in the Poincare disk, the $\mathsf{HyperbolicHorocycle}^{\uparrow}$ packet realizes the boundary-tangent horocycle rows requested by chapter 9463 as finite NameCert data. It sits after the disk-coordinate carrier of chapter 377, the metric-read surface of chapter 3474, the boundary-fixed transport route of chapter 3500, and the geodesic metric carrier of chapter 2213. The packet names a displayed horocycle certificate rather than a host circle model, a completed ideal-boundary quotient, a global geodesic foliation, or a physical observer trajectory.

Definition 2577.1 (Hyperbolic horocycle carrier)**.** *A* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, M, B, G, T, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source observations.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same disk data.*
3. *B is the boundary-anchor row, recording the displayed ideal boundary point only as a finite boundary-facing observation.*
4. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *incidence row used to compare tangent geodesic directions near the boundary anchor.*
5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *transport row preserving the boundary anchor during admissible disk transport.*
6. *L is the horocycle ledger row: boundary tangency, constant hyperbolic-distance readback relative to the anchored geodesic family, and finite incidence compatibility are recorded here.*
7. *H records componentwise* hsame *transport for disk, metric, boundary, geodesic, transport, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through disk admission, metric readback, boundary anchoring, geodesic incidence, boundary-fixed transport, and horocycle ledger inspection.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTranspo}$ $\mathsf{GeodesicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$ *row.*

The carrier exports horocycle reads only through $D, M, B, G, T, L, H, C, P, N$. It has no coordinate for a host Euclidean circle, quotient ideal boundary, global horocycle foliation, selected geodesic representative, physical observer motion, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicHorocycleUp.HyperbolicHorocycleTasteGate single carrier alignment`

Theorem 2577.2 (Hyperbolic horocycle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *carrier*

$$Q = (D, M, B, G, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric readback, boundary-anchor admission, geodesic-incidence readback, boundary-fixed transport compatibility, horocycle ledger inspection, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every boundary-tangent horocycle read exported by the packet must factor through

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{PoincareBoundaryFixedTransport}^{\uparrow},\ \mathsf{GeodesicMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

Proof. Project the accepted carrier. The disk surface is row D, the hyperbolic metric read is row M, the boundary anchor is row B, the geodesic incidence read is row G, and the boundary-fixed transport route is row T.

The horocycle-specific content is row L. It records boundary tangency, constant hyperbolic-distance readback relative to the anchored geodesic family, and finite incidence compatibility only after the disk, metric, boundary, geodesic, and transport rows have been displayed.

The structural rows H, C, P, N preserve and name that route. Since the carrier contains no host Euclidean circle, quotient ideal boundary, global horocycle foliation, selected geodesic representative, physical observer motion, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHorocycle}^{\uparrow}$ boundary-tangent read factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, provenance, transport, replay, and local NameCert rows.

Independence witness: unitdisk, hyperbolic-metric, poincare-boundary-fixed-transport, geodesic-metric: the UnitDisk packet records disk coordinates without hyperbolic tangency, the HyperbolicMetric packet records metric reads without a boundary anchor ledger, the PoincareBoundaryFixedTransport packet records fixed-boundary transport without horocycle incidence, and the GeodesicMetric packet records geodesic metric data without a boundary-tangent horocycle row. None is bijective to $\mathsf{HyperbolicHorocycle}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicHorocycle}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric metric reads, a boundary-anchor row, GeodesicMetric incidence, PoincareBoundaryFixedTransport compatibility, a horocycle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$, seed)

Scope closed: Definition 3534.1 and Definition 3534.2 seed the finite horocycle certificate over disk, metric, boundary, geodesic, transport, ledger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Euclidean circle models, ideal-boundary quotient equality, global horocycle foliations, selected geodesic representatives, physical observer motion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk admission, metric readback, boundary anchoring, geodesic incidence, boundary-fixed transport compatibility, horocycle ledger exactness, and a Lean carrier/obligation formalization for BEDC.Derived.HyperbolicHorocycleUp.

Chapter 2578

A Concrete Naming Certificate for $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ packet records the textbook complete-metric Baire argument as finite BEDC NameCert rows rather than as an ambient host theorem. It sits over the dense-open schedule of chapter 1542, the Cauchy-thread and limit-service row of chapter 128, the metric carrier rows of chapter 57, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the surface on which a Baire dense-open construction can be inspected before any countable-choice selector, host intersection, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim is available.

Definition 2578.1 (CompleteMetricBaireSurface carrier)**.** *A* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, K, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, listing the finite dense-open requests and the predecessor-ball refinements made visible to the packet.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, exposing only the displayed center sequence, radius shrinkage, and limit-service handoff used by the dense-open route.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, naming the distance, ball classifier, and metric comparison data consumed by B and K.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule, centers, radii, and requested tolerances are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational center and radius data visible before the terminal seal.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the schedule, Cauchy thread, metric carrier, finite windows, and rational readback have been displayed.*
7. *H records componentwise* hsame *transport for the dense-open schedule, complete-metric thread, metric carrier, finite windows, rational readback, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the same dense-open, Cauchy-thread, metric, window, readback, and seal route.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for countable choice, host intersections, open-cover compactness, selected limit points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CompleteMetricBaireSurfaceUp.CompleteMetricBaireSurfaceTasteGate single carrier alignment`

Theorem 2578.2 (Baire-category complete-metric NameCert surface)**.** *For every accepted* CompleteMetricBaireSur *carrier*

$$U = (B, K, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$ *surface exposes exactly the Baire dense-open schedule, complete-metric Cauchy thread, metric carrier rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every complete-metric Baire read is routed through $B, K, M, S, R, E, H, C, P, N$, and no obligation is stored outside those rows.*

Proof. Project the accepted carrier of Definition 3535.1. The rows B, K, M, S, R, E are the only analytic rows available to a consumer: dense-open schedule, Cauchy-thread service, metric carrier, finite windows, rational readback, and terminal real seal.

The Baire-category content is read first through B. Each dense-open request and predecessor-ball refinement is displayed as finite schedule data, then the complete-metric row K carries the Cauchy thread built from those refinements.

The numerical and metric observations are constrained by M, S, R, E. The metric carrier supplies distance and ball comparisons, the stream row supplies finite inspection windows, the regular rational row reads center and radius data, and the real seal is reached only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for countable choice, host intersections, open-cover compactness, selected limit points, ambient real completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2578.3 (Complete metric Baire surface nested-ball route)**.** *Let $U = (B, K, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier. A finite dense-open schedule displayed by B together with the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy service displayed by K determines a nested closed ball route whose radii are readable through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and whose limit request is sealed through* $\mathsf{Real}^{\uparrow}$ *only after the corresponding* $\mathsf{StreamName}^{\uparrow}$ *windows S have been displayed. The route is transported, replayed, sourced, and named only by H, C, P, N.*

Proof. Consume the dense-open schedule in B. Each finite request exposes the predecessor-ball refinement data that the Baire surface records before any host intersection or choice selector is available.

Choose the displayed ball rows from that finite schedule and read their metric comparisons through M. The complete-metric row K then supplies the Cauchy thread and limit-service handoff for the same displayed nested ball route.

The numerical observations pass through S and R. The stream windows display the center and radius observations, and the regular rational readback makes the radii available without adding a selected limit point.

Finally use the Real seal nonescape already built into Definition 3535.2. The seal row E is downstream from the dense-open schedule, complete-metric thread, metric carrier, stream windows, and rational readback, while H, C, P, N preserve that route without adding countable choice, host intersections, open-cover compactness, selected limit points, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2578.4 (CompleteMetricBaireSurface open-density thread)**.** *Let*

$$U = (B, K, M, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier. Every dense-open schedule read from B must expose the complete-metric Cauchy thread K, metric row M,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and* $\mathsf{Real}^{\uparrow}$ *seal E before a Baire consumer may inspect the terminal route. The exposed thread is transported, replayed, sourced, and named only by H, C, P, N, and it exports no selected generic point, countable-choice schedule, host intersection, open-cover compactness, selected limit point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3535.1. The dense-open schedule B and Cauchy thread K are coordinates of the same packet, so a consumer cannot read the schedule as an external countable-choice object.

Use the NameCert surface of Definition 3535.2. It makes $B, K, M, S, R, E, H, C, P, N$ the complete visible route for every complete-metric Baire read, with no obligation stored outside those rows.

Compose with the nested-ball route of Definition 3535.3. The metric row M, finite windows S, rational readback R, and seal row E must be exposed before the terminal route can be inspected.

Finally close by carrier exhaustion. The structural rows H, C, P, N preserve the same route, while the excluded generic point, choice schedule, host intersection, compactness, selected limit, and ambient completeness endpoints have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3535.1 and Definition 3535.2 seed the complete-metric Baire surface over dense-open schedules, Cauchy-thread rows, metric carriers, finite windows, rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, host intersections, open-cover compactness, selected limit points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric Cauchy-thread exactness, finite-window readback, rational center readback, real sealing, and nonescape for BEDC.Derived.CompleteMetricBair

Constructive story: A $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteMetric Cauchy thread, MetricSpace carrier rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2579

A Concrete Naming Certificate for $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$

Following Bishop and Bridges' constructive finite-subcover treatment for compact intervals, the $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ packet records the BEDC route that turns a located closed interval cover request into a displayed finite subcover through finite nets and positive-radius data. It sits over the located interval finite-net surface of chapter 1808, the compact-cover radius route of chapter 2301, the finite-net interval-cover packet of chapter 3471, the terminal real-name seal of chapter 38, and the dyadic tolerance rows of chapter 291. The packet names a located finite-subcover certificate; it does not claim unrestricted open-cover Heine–Borel compactness, selected subcovers from arbitrary covers, countable choice, quotient interval equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Its subcover read uses chapter 3471 and the located interval finite-net sibling as mandatory predecessors before any retained witness can reach the $\mathsf{Real}^{\uparrow}$ seal.

Definition 2579.1 (Located Heine–Borel subcover carrier)**.** *A* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, L, H, R, D, E, T, C, P, N)$$

with the following displayed rows:

1. *I is the located closed interval source row, exposing endpoint names, endpoint order, and locatedness only as finite* $\mathsf{Real}^{\uparrow}$ *histories.*
2. *F is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *finite-net row, listing the mesh scale and finite interval net used by the cover request.*
3. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *positive-radius row attached to the displayed finite cover or pointwise radius family.*
4. *H is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *sibling row that records the finite-net interval-cover route over the same interval, net, radius, dyadic, and real-seal data.*
5. *R is the retained-subcover row, listing exactly the finitely many cover witnesses selected by the displayed net and radius checks.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording mesh width, radius slack, endpoint containment, and finite neighborhood comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint, radius, and coverage reads.*

8. *T records componentwise* hsame *transport for the interval source, finite-net row, radius row, finite-net sibling row, retained-subcover row, dyadic ledger, real seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a located cover request through $I, F, L, H, R, D, E, T, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, LebesgueNumberLem HeineBorelFiniteNet$^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, H, R, D, E, T, C, P, N$. It has no coordinate for arbitrary open covers, unrestricted Heine–Borel compactness, selected subcover functionals, quotient interval representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedHeineBorelSubcoverUp.LocatedHeineBorelSubcoverTasteGate single carrier alignment`

Theorem 2579.2 (Located Heine–Borel subcover NameCert obligations)**.** *For every accepted* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ *carrier*

$$S = (I, F, L, H, R, D, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the located interval source, finite-net row, positive-radius row, finite-net sibling row, retained-subcover row, dyadic ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *located interval exposure, requiring every cover request to display I and the finite-net row F before any retained cover witness is inspected;*

3. *radius compatibility, requiring L and D to record the positive-radius and dyadic slack checks that allow listed net points to control nearby interval points;*

4. *finite-subcover exactness, requiring R to list only cover witnesses reached through the finite net, radius ledger, and* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *sibling route;*

5. *real-seal non-escape, requiring E to be downstream of the located interval, finite-net, radius, retained-subcover, and dyadic rows;*

6. *structural preservation through T, C, P, N and refusal of arbitrary open-cover compactness, unrestricted Heine–Borel, selected subcover functionals, quotient interval representatives, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No located finite-subcover obligation is stored outside these rows.

Proof. Project the carrier of Definition 3538.1. The analytic rows are exactly the located interval source I, finite-net row F, positive-radius row L, finite-net sibling row H, retained-subcover row R, dyadic ledger D, and terminal real seal E.

The cover request first exposes I and F. The packet therefore reads a located interval and a displayed finite net before it can inspect any cover witness, so the route cannot start from an arbitrary open-cover selector.

The radius step is finite. The row L supplies the positive radius ledger, while D records the mesh-width and endpoint-slack comparisons that transfer a witness at a listed net point to nearby interval points.

The retained row R is downstream of F, L, H, D. Hence every retained cover witness is tied to the same finite-net route already named by chapter 3471, and the terminal seal E is reached only after those finite checks have been displayed.

Transport, replay, provenance, and local naming preserve this finite surface. Since the refused objects have no carrier coordinate, the local certificate exports only the located finite-subcover route. □

Theorem 2579.3 (Located Heine–Borel finite-net route)**.** *Every located cover read from an accepted* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The exported object is the retained finite subcover row R over $I, F, L, H, D, E, T, C, P, N$, not an unrestricted open-cover Heine–Borel principle.

Proof. Begin with an accepted carrier $S = (I, F, L, H, R, D, E, T, C, P, N)$. The located interval and finite net are exposed by I and F, the positive-radius ledger is L, and the already displayed finite-net cover route is H.

The retained-subcover row R may be read only after those rows. Its witnesses are checked by the dyadic ledger D and then sealed by E, so the exported finite subcover is a displayed BEDC packet row rather than a host-level choice operation.

Replay through C, transport through T, provenance through P, and naming through N keep the same route. No arbitrary open cover, unrestricted compactness theorem, quotient interval representative, countable-choice row, or ambient completeness coordinate is available to bypass it. □

Theorem 2579.4 (Located Heine–Borel subcover sibling route)**.** *Any accepted read of a* $\mathsf{LocatedHeineBorelSubcover}$ *retained subcover row must pass through chapter 3471 and the located interval finite-net sibling before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Equivalently, in the carrier $S = (I, F, L, H, R, D, E, T, C, P, N)$, the row R is readable only after the displayed finite-net row F and the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *sibling row H have exposed the same interval, radius, dyadic, and cover-witness surface.*

Proof. Project the carrier of Definition 3538.1. The located interval finite-net row is F, the finite-net sibling row is H, and the retained-subcover row is R.

Apply Definition 3538.2. Its located interval exposure and finite-subcover exactness clauses require every retained witness to be downstream of F, L, H, D before the real-seal row E is available.

Finally apply Definition 3538.3. The displayed route places $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ between the positive-radius row and the dyadic/Real seal, so a located subcover consumer cannot read R by bypassing either sibling. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ packet is generated from a located closed interval source, a LocatedClosedIntervalFiniteNet row, a LebesgueNumberLemma positive-radius ledger, a HeineBorelFiniteNet sibling row, retained finite subcover witnesses, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3538.1, Definition 3538.2, and Definition 3538.3 seed the located finite-subcover route over interval nets, positive-radius ledgers, finite-net cover rows, dyadic tolerances, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted open-cover Heine–Borel compactness, arbitrary subcover selection, countable choice, quotient interval equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval exposure, radius compatibility, finite-subcover exactness, real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedHeineBorelSubcover

Chapter 2580

A Concrete Naming Certificate for MetaCICTypedNormalizationAudit$^{\uparrow}$

The MetaCICTypedNormalizationAudit$^{\uparrow}$ packet realizes the candidate-mediated typed-normalization route of chapter 9383 as finite BEDC rows. It sits beside the normalization audit packet of chapter 750, the candidate-evidence strong-normalization route of chapter 758, the Pi-application adequacy route of chapter 907, and the normalization frontier of chapter 892. The packet names the closed typed term, candidate evidence, Pi and application adequacy, no-infinite-reduction projection, frontier refusal, transport, replay, provenance, and local naming rows needed for an audit citation. It does not claim evidence-free strong normalization for every closed typed MetaCIC term, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Definition 2580.1 (MetaCIC typed-normalization audit carrier)**.** *A* MetaCICTypedNormalizationAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, K, S, P, R, Q, F, H, C, G, N)$$

with the following displayed rows:

1. *T is the closed typed term row, exposing the empty-context term, type, and closedness checks read by the normalization audit map.*
2. *K is the candidate-evidence row, carrying the reducibility candidate, closed-candidate, or candidate-set membership evidence consumed by the closed typed term projection.*
3. *S is the strong-normalization projection row, recording the candidate-mediated route from K to the displayed strong-normalization witness for T.*
4. *P is the Pi adequacy row, exposing Pi-candidate and neutral-normal evidence only when the inspected term route requires it.*
5. *R is the application adequacy row, exposing function-side Pi evidence, argument candidate evidence, and the application-candidate handoff.*
6. *Q is the no-infinite-reduction row, recording the projection from strong normalization to absence of infinite beta chains along the same candidate-mediated route.*
7. *F is the frontier row, retaining the boundary named by* MetaCICNormalizationFrontier$^{\uparrow}$ *when the requested read would erase candidate evidence or subject-reduction obligations.*

8. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, SN projection, Pi route, application route, no-infinite-reduction row, frontier row, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a typed-normalization audit request through $T, K, S, P, R, Q, F, H, G, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$*, chapter 9383, chapter 750, chapter 758, chapter 907, chapter 892,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, S, P, R, Q, F, H, C, G, N$. It has no coordinate for evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, erased candidate evidence, or completed closed consistency.

Theorem 2580.2 (MetaCIC typed-normalization audit NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ *carrier*

$$A = (T, K, S, P, R, Q, F, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the closed typed term row, candidate-evidence row, SN projection row, Pi adequacy row, application adequacy row, no-infinite-reduction row, frontier row, transport, replay, provenance, and local naming rows;*
2. *candidate gate exactness, requiring every SN projection in S to read candidate, closed-candidate, or candidate-set evidence from K before it can read the closed typed term row T as normalized;*
3. *Pi and application adequacy compatibility, requiring P and R to be consumed only as typed route data for the same candidate-mediated term window;*
4. *no-infinite-reduction compatibility, requiring Q to be downstream of S and to reuse the same candidate evidence rather than introducing a separate termination source;*
5. *frontier retention, requiring F to remain visible whenever a requested read would erase candidate evidence, subject-reduction obligations, confluence boundaries, or the normalization audit map's conditional interface;*
6. *structural preservation through H, C, G, N and refusal of evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, erased candidate evidence, completed closed consistency, Lean verification, and bridge checking.*

No typed-normalization audit obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.MetaCICTypedNormalizationAuditUp.MetaCICTypedNormalizationAuditNameCertObligations`

Proof. Project the carrier of Definition 3539.1. Its typed-normalization data are exactly the closed typed term row T, candidate-evidence row K, SN projection row S, Pi row P, application row R, no-infinite-reduction row Q, and frontier row F.

The candidate gate is first. A closed typed term can be read as strongly normalizing only after K supplies the candidate, closed-candidate, or candidate-set membership evidence consumed by the audit map.

The Pi and application rows refine that same route. The row P supplies Pi-candidate and neutral-normal data when the term route requires it, while R supplies function, argument, and application-candidate evidence for application cases. Neither row can replace the candidate gate.

The no-infinite-reduction row Q is downstream of S. It inherits the same candidate-mediated strong-normalization witness and therefore cannot introduce a separate global termination theorem.

The frontier row F remains visible for rejected readings. Transport, replay, provenance, and local naming preserve the listed rows, so the packet refuses evidence-free normalization, unrestricted subject reduction, global confluence, host evaluation soundness, quotient syntax equality, and closed consistency outside the displayed carrier. □

Theorem 2580.3 (MetaCIC typed-normalization substitution route)**.** *Every closed typed normalization read from an accepted* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ *packet factors through the finite route*

$$T \longmapsto K \longmapsto S \longmapsto (P, R) \longmapsto Q \longmapsto F.$$

The exported route is candidate-mediated typed-normalization audit data over $A = (T, K, S, P, R, Q, F, H, C, G, N)$*, not a theorem that every closed typed MetaCIC term normalizes without supplied candidate evidence.*

Proof. Start with a closed typed term read. The row T exposes the typed and closed term window, but it does not by itself carry a normalization theorem.

The next row is K. It supplies the candidate evidence used by S to project strong normalization for that same closed typed term. The Pi and application rows P, R are read only as adequacy data for this candidate-mediated route.

The row Q then projects no-infinite-reduction along the same strong-normalization path. If a consumer asks for a reading that drops the candidate evidence or erases subject-reduction and confluence boundaries, the frontier row F records that refusal.

Thus every accepted read remains inside the displayed finite carrier and cannot be reread as evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency. □

Falsifiable prediction: Every $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ closed typed normalization read factors through displayed closed typed term data, candidate evidence, candidate-mediated strong-normalization projection, Pi adequacy rows, application adequacy rows, no-infinite-reduction projection, frontier retention, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacicnormalizationaudit, candidate-evidence-sn-route, pi-application-adequacy-route, metacicnormalizationfrontier: $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed typed term row, candidate gate, SN projection, Pi route, application route, no-infinite-reduction route, frontier retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ packet is generated from a closed typed term row, candidate-evidence row, candidate-mediated strong-normalization projection, Pi adequacy row, application adequacy row, no-infinite-reduction row, normalization-frontier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3539.1, Definition 3539.2, and Definition 3539.3 seed the finite candidate-mediated typed-normalization audit route over closed typed term, candidate, Pi, application, no-infinite-reduction, and frontier rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICTypedNormalizationAudit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization for every closed typed MetaCIC term, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate gate exactness, Pi and application adequacy compatibility, no-infinite-reduction compatibility, frontier retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedNormalizationAuditUp.

Chapter 2581

A Concrete Naming Certificate for $\mathsf{LocatedCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ packet records the traditional located criterion by which completion-facing consumers recognize a displayed Cauchy name as admitted by finite tolerance windows. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric-completion carrier of chapter 1506, and the located complete-metric service of chapter 3349. The packet names the criterion route used by L10 Real and Completion consumers; it is not a quotient completion, selected limit, countable-choice schedule, trichotomy principle, host equality test, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2581.1 (Located completion criterion carrier)**.** *A* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, E, M, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint comparisons, and error-budget rows inspected by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the requested observations of the Cauchy name and the local locatedness tests attached to those observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence obtained from the displayed windows and dyadic budgets.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been exposed.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row that records the completion-facing object surface for the same displayed Cauchy name.*
6. *L is the* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row, recording the located complete-metric limit service available for accepted regular Cauchy point names.*
7. *H records componentwise* hsame *transport for the dyadic tolerance ledger, stream windows, regular readback, Real seal, metric-completion carrier, located complete-metric service, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a criterion request through D, S, R, E, M, L.*

9. *P is the* Pkg *provenance over* LocatedCompletionCriterion$^\uparrow$, DyadicRatCore$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, MetricCompletion$^\uparrow$, LocatedCompleteMetric$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LocatedCompletionCriterion}^\uparrow}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, M, L, H, C, P, N$. It has no coordinate for quotient Cauchy equality, selected representatives, countable choice, host real equality, trichotomy, or completeness outside the displayed located complete-metric service.

Lean4 ✓ `BEDC.Derived.LocatedCompletionCriterionUp.LocatedCompletionCriterionTasteGate single carrier alignment`

Theorem 2581.2 (Located completion criterion NameCert obligations)**.** *For every accepted* LocatedCompletionCrit *carrier $K = (D, S, R, E, M, L, H, C, P, N)$, the local* NameCert$_{\text{LocatedCompletionCriterion}^\uparrow}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, metric-completion carrier, located complete-metric service, componentwise transport, displayed continuation replay, package provenance, and local naming rows.*

Every completion-facing consumer must read the finite route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto M \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before treating the Cauchy name as admitted by the located completion criterion.

Proof. Project the carrier of Definition 3540.1. The accepted packet displays exactly the dyadic tolerance, stream-window, regular readback, Real seal, metric-completion, located complete-metric, transport, replay, provenance, and naming rows.

The analytic criterion begins with D and S. The dyadic ledger supplies the requested tolerance budget, and the stream row supplies the finite observations and locatedness tests made at that budget.

The readback and completion handoff are downstream. The row R repacks the inspected windows as regular rational data, the row E seals the Real-facing name, and the rows M and L expose only the displayed metric-completion and located complete-metric service attached to the same route.

Finally H, C, P, N preserve, replay, source, and name that route. Since the carrier contains no quotient equality, selected representative, countable-choice schedule, host equality test, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the local NameCert obligations. □

Theorem 2581.3 (Located completion criterion Real seal route)**.** *Every* Real$^\uparrow$, MetricCompletion$^\uparrow$, *or* LocatedCompleteMetric$^\uparrow$ *consumer of an accepted* LocatedCompletionCriterion$^\uparrow$ *packet reaches the terminal seal and completion-facing service only through the dyadic tolerance ledger D, finite* StreamName$^\uparrow$ *window S,* RegSeqRat$^\uparrow$ *readback R,* Real$^\uparrow$ *seal E,* MetricCompletion$^\uparrow$ *carrier M,* LocatedCompleteMetric$^\uparrow$ *service row L, componentwise transport H, displayed replay C, provenance P, and local name N.*

The route exports a located criterion for displayed completion data, not a quotient completion, selected host limit, countable-choice construction, host equality test, trichotomy principle, or ambient completeness theorem for all Real$^\uparrow$ *names.*

Proof. Start with a completion-facing or Real-facing consumer. The only terminal Real coordinate in the carrier is E, and the only completion-facing rows are M and L.

The consumer must first inspect D, S, R. The dyadic tolerances, finite windows, and regular rational readback are the displayed data that make the criterion located before the seal is exposed.

After the seal, M and L record the metric-completion surface and located complete-metric service attached to the same displayed Cauchy name. The support rows H, C, P, N can transport and replay this path, but they cannot add quotient completion equality, selected limits, countable choice, host equality, trichotomy, or ambient completeness because none of those coordinates is present in the accepted carrier. □

Theorem 2581.4 (Located completion criterion sequential closure)**.** *Let $K = (D, S, R, E, M, L, H, C, P, N)$ be an accepted* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier. Any Real-facing sequential-closure read of K must pass through the dyadic tolerance windows in D, the finite* $\mathsf{StreamName}^{\uparrow}$ *observations in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R, the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row M, the* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row L, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, with transport, replay, provenance, and local naming supplied only by H, C, P, N.*

Consequently the accepted carrier admits a Real-facing read only as a finite tolerance-window route. It does not export quotient completion equality, a selected limit, countable choice, trichotomy, host real equality, or an ambient completeness theorem for all displayed names.

Proof. Read the carrier through Definition 3540.1. The only rows available for sequential closure are $D, S, R, E, M, L, H, C, P, N$, and the finite observation route begins with the dyadic tolerance ledger and stream windows.

Apply the Real seal route of Definition 3540.3. That theorem forces every Real-facing or completion-facing consumer to reach E, M, L only after D, S, R have been exposed.

The closure read is therefore finite at every stage. The dyadic ledger supplies the tolerance windows, the stream row supplies the inspected observations, the regular rational row supplies readback, and the metric-completion and located complete-metric rows supply the completion-facing service before the terminal Real seal is consumed.

Finally H, C, P, N preserve the same route without adding coordinates. Since the carrier definition contains no quotient completion, selected limit, countable-choice schedule, trichotomy row, host equality test, or ambient completeness field, the sequential-closure read cannot export any of those non-claims. □

Theorem 2581.5 (Located completion criterion terminal seal exhaustion)**.** *Let $K = (D, S, R, E, M, L, H, C, P, N)$ be an accepted* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier. Any Real-facing completion read from K exhausts exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{StreamName}^{\uparrow}$ *observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E,* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier M,* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row L, componentwise transport H, displayed replay C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}$ *row N. It has no remaining coordinate for selected limits, quotient equality, host equality, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the NameCert obligations of Definition 3540.2. They identify the displayed rows $D, S, R, E, M, L, H, C, P, N$ as the exact local certificate surface for an accepted carrier.

Use the Real seal route of Definition 3540.3. It forces every Real-facing or completion-facing consumer to pass through the dyadic ledger, stream windows, regular readback, terminal Real seal, metric-completion carrier, located complete-metric service, transport, replay, provenance, and local naming rows.

Finally apply sequential closure, Definition 3540.4. The completion read remains finite at each stage and cannot acquire coordinates absent from the carrier. Therefore exhausting the listed rows exhausts the accepted terminal-seal route. □

Falsifiable prediction: Every accepted LocatedCompletionCriterion$^{\uparrow}$ consumer factors a completion-facing read through one DyadicRatCore$^{\uparrow}$ tolerance ledger, one StreamName$^{\uparrow}$ finite-window row, one RegSeqRat$^{\uparrow}$ readback row, one Real$^{\uparrow}$ seal, one MetricCompletion$^{\uparrow}$ carrier, one LocatedCompleteMetric$^{\uparrow}$ service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletion, located-complete-metric: LocatedCompletionCriterion$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries tolerance inspection, finite windows, regular readback, Real sealing, metric-completion admission, and located complete-metric service as one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}\uparrow}$, seed)

Scope closed: Definition 3540.1, Definition 3540.2, and Definition 3540.3 seed the finite located completion criterion route over dyadic tolerance ledgers, stream windows, regular rational readback, Real seals, metric-completion carriers, located complete-metric service, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window inspection, RegSeqRat readback, Real seal handoff, MetricCompletion carrier admission, LocatedCompleteMetric service, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionCriterionUp.

Constructive story: A LocatedCompletionCriterion$^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, a MetricCompletion carrier, a LocatedCompleteMetric service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2582

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$

Following Bishop–Bridges and Bourbaki-style constructive Cauchy-space grounding, the RegularCauchySequenceSpa packet records rate-controlled families of regular rational Cauchy names before any completion or equality quotient is read. It sits after the broader Cauchy-name surface of chapter 406, the finite schedule rows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite interface for regular-rate sequence-space consumers; it is not a proof that $\mathsf{Real}^{\uparrow}$ is complete, not a quotient of Cauchy streams, and not a selected limiting representative.

Definition 2582.1 (RegularCauchySequenceSpace carrier)**.** *A* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is a finite family of* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence rows.*
2. *ρ is the regular-rate ledger assigning dyadic Cauchy budgets to the rows of F.*
3. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that chooses the inspected sequence and finite window.*
4. *W records the finite common-window data used by the scheduled family inspection.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the requested regularity budget.*
6. *Q is the surrounding* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *row through which the same family can be read as a Cauchy-name-space object.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, ρ, σ, W, D, Q have been displayed.*
8. *H stores componentwise* hsame *transport for the family, rate ledger, schedule, window, tolerance, Cauchy-space, and real-seal rows.*
9. *C stores displayed* Cont *replay from finite rate requests to scheduled windows, regular rational readback, the CauchySequenceSpace handoff, and the terminal Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, \rho, \sigma, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceSpaceUp.RegularCauchySequenceSpaceTasteGate single carrier alignment`

Theorem 2582.2 (RegularCauchySequenceSpace NameCert obligations)**.** *For every accepted* RegularCauchySequen *carrier $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, regular-rate ledger exposure, scheduled finite-window stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of quotient streams, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Carrier admission is projection from Definition 3541.1. The accepted packet exposes the regular rational family F, the regular-rate ledger ρ, the schedule σ, the finite window row W, and the dyadic tolerance row D before any Cauchy-space or Real-facing consumer can read the packet.

The Cauchy-name-space handoff is then forced through Q. The surrounding $\mathsf{CauchySequenceSpace}^{\uparrow}$ row can consume the same family, schedule, and tolerance data, but it receives them only as displayed rows of this finite carrier.

Finally the terminal seal E is downstream of F, ρ, σ, W, D, Q and is transported only by H and replayed only by C. The provenance row P and local naming row N close the certificate surface, while the absence of any additional carrier coordinate excludes quotient streams, selected limits, countable choice, host real equality, ambient completeness, Lean verification, and bridge checking. □

Theorem 2582.3 (RegularCauchySequenceSpace real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

Thus the consumer first reads regular rational sequence data, then a regular-rate dyadic budget, then a scheduled finite window, then the $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff, and only then the terminal real-name seal. The route exports no quotient of Cauchy names, no selected limiting representative, no host real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceSpaceUp.RegularCauchySequenceSpaceRealHandoff`

Proof. Begin with an accepted carrier $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$. The only regular sequence data available to a consumer are the rows F and ρ, and the only finite observation channel is the schedule and window pair σ, W together with the dyadic tolerance row D.

Use Definition 3541.2. It binds regular-rate exposure, scheduled window stability, readback, and tolerance exactness to the same finite packet.

The Cauchy-space read reaches Q only after those rows have been displayed, and the Real-facing read reaches E only after Q has been exposed. Since the carrier definition lists no quotient, selected limit, host real equality, or ambient completeness coordinate, every $\mathsf{Real}^{\uparrow}$ consumer route is exhausted by the displayed rows. □

Theorem 2582.4 (RegularCauchySequenceSpace rate-ledger stability)**.** *Let $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier. Any classifier-preserving replacement of the family row F, regular-rate ledger ρ, schedule σ, and finite window row W preserves the same dyadic tolerance row D, the same* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff row Q, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row E whenever the replacement remains inside the local classifier determined by H, C, P, N.*

Thus a consumer may not replace the displayed rate ledger by quotient stream equality, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. The stable route remains the finite row sequence*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

Proof. Project the carrier rows from Definition 3541.1. The rows F, ρ, σ, W are the only family, rate, schedule, and finite-window data available to the local classifier.

Apply Definition 3541.2. It identifies regular-rate ledger exposure, scheduled finite-window stability, tolerance exactness, CauchySequenceSpace handoff, and Real seal routing as obligations over the same displayed carrier.

Now apply Definition 3541.3. Real-facing consumers see only the displayed route through D, Q, E after the replacement has preserved the classifier on F, ρ, σ, W. Since quotient stream equality, selected limits, host real equality, and ambient completeness are not carrier coordinates, the replacement preserves the same tolerance, handoff, and terminal seal rows rather than substituting any of those refused objects. □

Theorem 2582.5 (RegularCauchySequenceSpace tail-zero route)**.** *Let $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier. If the regular-rate ledger ρ carries an explicit dyadic tail-to-zero modulus, then every* $\mathsf{Real}^{\uparrow}$*-facing consumer still factors through*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

The tail-zero conclusion is a displayed rate-ledger read over ρ, D, Q, E; it is not a selected limit, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier rows from Definition 3541.1. The family F, rate ledger ρ, schedule σ, window W, dyadic row D, CauchySequenceSpace handoff Q, and Real seal E are the only data available before the structural rows transport and name the packet.

Apply Definition 3541.2. It binds regular-rate exposure, scheduled finite-window stability, regular rational readback, dyadic tolerance exactness, the CauchySequenceSpace handoff, and the Real seal to the same displayed packet.

Now apply Definition 3541.3 and Definition 3541.4. The Real-facing route remains the finite row sequence above, and the tail-to-zero modulus remains a rate-ledger entry rather than a selected limit or quotient-stream coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ packet is generated from a finite regular rational sequence family, a regular-rate dyadic ledger, scheduled finite stream windows, dyadic tolerance rows, a CauchySequenceSpace handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3541.1, Definition 3541.2, Definition 3541.3, and Definition 3541.4 seed the finite regular-rate sequence-space route from regular rational families to Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient Cauchy-name equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, regular-rate ledger exposure, scheduled finite-window stability, tolerance exactness, CauchySequenceSpace handoff, RealUp seal routing, and non-escape for BEDC.Derived.RegularCauchySe

Chapter 2583

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive completions and dense embeddings, the $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ packet records the finite dense-source face used by the Cauchy completion universal-property route. It sits between the universal-property surface of chapter 2319, the metric dense-image packet of chapter 3351, the regular-Cauchy dense route of chapter 1453, the Real admission route of chapter 3389, the regular rational source of chapter 301, the stream-window source of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet lets universal-property, completion-functor, and Real-completion consumers cite a dense source without importing quotient completion, selected limits, countable choice, or ambient real completeness.

Definition 2583.1 (CauchyCompletionDensitySource carrier)**.** *A* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, M, J, R, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *consumer row that reads the dense source before any extension or uniqueness row is available.*
2. *M is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row, exposing the source metric, completion carrier, dense image, separated comparison, uniform handoff, and regular rational readback used by metric completion consumers.*
3. *J is the* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *row, recording the regular Cauchy rational source-to-completion route.*
4. *R is the* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *row, recording the admission of the same regular Cauchy source into the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name and its displayed modulus discipline.*
6. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source only at requested stages.*
7. *S is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording precision budgets, slack rows, and finite comparison data before the* $\mathsf{Real}^{\uparrow}$ *seal is read.*

8. *H records componentwise* hsame *transport for $U, M, J, R, W, Q, S, C, P, N$.*
9. *C records displayed* Cont *replay from dense-source requests through the metric dense image, regular Cauchy route, Real admission route, source windows, dyadic tolerance ledger, transport, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient representatives, selected completion limits, selected real representatives, countable choice, trichotomy, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDensitySourceUp.CauchyCompletionDensitySourceTasteGate`
`single carrier alignment`

Theorem 2583.2 (CauchyCompletionDensitySource NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *carrier $D = (U, M, J, R, W, Q, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDen}}$ *surface consists exactly of carrier admission for the universal-property consumer row, metric dense-image row, regular Cauchy dense route, Real admission route, RegSeqRat source, StreamName finite windows, DyadicRatCore tolerance ledger, componentwise transport, displayed replay, provenance, and local naming. A completion universal-property consumer must read M, J, R, W, Q, S as dense-source evidence before it can consume the extension and uniqueness rows of U.*

Proof. Project the carrier of Definition 3542.1. The tuple displays the universal-property consumer row, metric dense-image row, regular Cauchy dense route, Real admission route, regular rational source, stream windows, dyadic tolerance ledger, transport, replay, provenance, and local naming row.

The dense-source evidence is finite and source-facing. The metric dense-image row supplies the completion-side dense embedding, the regular Cauchy route supplies the rational source discipline, the Real admission route records the terminal seal, and the RegSeqRat, StreamName, and DyadicRatCore rows expose only requested source observations and tolerance budgets.

The universal-property row U is downstream of those rows. Its extension and uniqueness reads therefore consume displayed dense-source evidence rather than quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient completeness. □

Theorem 2583.3 (CauchyCompletionDensitySource universal route)**.** *Every universal-property, completion-functor, or* $\mathsf{Real}^{\uparrow}$*-completion consumer of an accepted* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetricCompletionDenseEmbedding}^{\uparrow},\ \mathsf{RegularCauchyDenseEmbedding}^{\uparrow},\ \mathsf{RealCauchyDenseEmbedding}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},$$

The exported object is a dense-source packet. It is not a quotient completion, a selected limit operator, a countable-choice principle, a host equality comparison, or an ambient real-completeness theorem.

Proof. Begin with the local obligations of Definition 3542.2. Any consumer reaching the universal-property row must first read the metric dense image, regular Cauchy dense route, Real admission route, regular rational source, finite stream windows, and dyadic tolerance ledger.

The sibling packets determine the route. The metric dense-image row gives completion-facing density, the regular Cauchy dense route gives the source discipline, the Real admission route gives

the seal handoff, and the RegSeqRat, StreamName, and DyadicRatCore rows give finite observation and tolerance data.

Transport, replay, provenance, and local naming preserve the same displayed route. Since the carrier contains no row for quotient representatives, selected limits, countable choice, host equality, or ambient real completeness, the universal route exports only the dense-source packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ consumer that reaches a universal-property extension or uniqueness read while omitting the metric dense image, regular Cauchy dense route, Real admission route, RegSeqRat source, StreamName windows, DyadicRatCore tolerance ledger, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchy-completion-universal-property, metric-completion-dense-embedding, regular-cauchy-dense-embedding, real-cauchy-dense-embedding, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective with $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ because the packet jointly carries dense-source evidence, universal-property consumption, completion-facing dense image data, regular rational source discipline, finite windows, dyadic tolerance accounting, Real-seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ packet is generated from a universal-property consumer row, a metric dense-image row, a regular Cauchy dense route, a Real admission route, a RegSeqRat source row, finite StreamName windows, a DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3542.1, Definition 3542.2, and Definition 3542.3 seed the dense-source face consumed by constructive Cauchy completion universal-property routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, countable choice, trichotomy, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exactness, metric dense-image handoff, regular Cauchy source readback, Real-seal handoff, finite-window and tolerance accounting, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyCompletionDensitySourceUp.

Chapter 2584

A Concrete Naming Certificate for $\mathsf{SeparatedCompletionReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ packet records the finite reflection boundary where separated zero-distance evidence is allowed to feed completion uniqueness. It sits between the separated metric classifier of chapter 1411, the complete-limit rows of chapter 128, the metric distance surface of chapter 57, the terminal real-name seal of chapter 38, and the regular-Cauchy comparison route of chapter 1412. The packet names a reflection route for completion consumers without importing quotient completion, selected representatives, trichotomy, countable choice, or a claim that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 2584.1 (SeparatedCompletionReflection carrier)**.** *A* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, C, E, Z, H, K, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier and the metric-distance witness surface that completion reads may inspect.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier row, read only after the metric distance witness has been displayed.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *pair of completion-limit rows for a single regular Cauchy point name, including the displayed limit modulus and convergence-distance ledgers.*
4. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route binding the two completion-limit reads to the same finite regular-Cauchy window data.*
5. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance read of the metric-distance witness between the two displayed completion limits.*
6. *H records componentwise* hsame *transport for M, S, C, E, Z, K, P, N.*
7. *K records displayed* Cont *replay from finite Cauchy windows to completion limits, from those limits to a metric-distance witness, and from the real zero-distance row to the separated classifier.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$ *row.*

The carrier is a reflection surface: it records how already displayed separated zero-distance evidence is consumed by completion uniqueness. It has no quotient-completion object, no representative selector, no trichotomy row, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SeparatedCompletionReflectionUp.SeparatedCompletionReflectionTasteGate`
`single carrier alignment`

Theorem 2584.2 (SeparatedCompletionReflection NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *carrier* $R = (M, S, C, E, Z, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the* $\mathsf{Met}^{\uparrow}$ *source row, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier, the paired* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route, the* $\mathsf{Real}^{\uparrow}$ *zero-distance read, componentwise transport, continuation replay, provenance, and local naming;*
2. *completion-window exactness, requiring the two completion-limit reads in C to be compared only through the finite-window route E before any endpoint classifier is inspected;*
3. *zero-distance exactness, requiring the metric-distance witness from M to pass through the* $\mathsf{Real}^{\uparrow}$ *row Z before the separated classifier S is consumed;*
4. *transport and replay stability, requiring* hsame *transport through H and* Cont *replay through K to preserve the same order E, M, Z, S under local naming;*
5. *ledger visibility, requiring P and N to expose only the displayed rows and to refuse quotient completion, selected representatives, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3543.1. Its fields give exactly the metric source, separated classifier, complete-limit pair, Cauchy comparison route, real zero-distance read, and structural support rows named in the first obligation.

For completion-window exactness, read C through E. The two limit rows are thereby tied to the same regular-Cauchy window data before the metric-distance witness is formed, so no quotient completion or selected representative can supply an alternate comparison.

For zero-distance exactness, the metric row M supplies the distance witness, Z reads that witness at zero through the $\mathsf{Real}^{\uparrow}$ seal, and only then may S export the separated classifier. The transport and replay rows H, K preserve this same order componentwise.

The final obligation is the provenance inventory. Since P and N name only the rows listed in the carrier, the local certificate cannot expose trichotomy, countable choice, a quotient completion carrier, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2584.3 (SeparatedCompletionReflection route)**.** *Every completion-uniqueness consumer of an accepted* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, the consumer must first read the paired complete-limit rows C, then the finite-window comparison route E, then the metric-distance witness from M, then the $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, and only afterward the separated classifier row S with support from H, K, P, N. The exported conclusion is completion uniqueness as a local separated-classifier route, not quotient completion equality, a selected limit, host equality, trichotomy, countable choice, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the paired $\mathsf{CompleteMetric}^{\uparrow}$ limit rows in C. The comparison route E binds them to the same regular Cauchy point name through finite windows before any endpoint comparison is available.

Next form the displayed metric-distance witness using M. The real-valued zero-distance read Z is then the only row that turns that witness into data consumable by a separated classifier.

Finally apply the $\mathsf{SeparatedMetric}^{\uparrow}$ row S and carry the result through H, K, P, N. Transport, replay, provenance, and local naming do not add analytic coordinates; they only preserve and name the displayed route. Since the carrier contains no quotient-completion, selected-representative, trichotomy, countable-choice, or ambient-completeness field, the consumer route has no escape from the stated boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ packet is generated from a MetricUp source row, a SeparatedMetricUp zero-distance classifier, paired CompleteMetricUp limit rows, a CauchySequenceEquivalenceUp comparison route, a RealUp zero-distance read, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3543.1, Definition 3543.2, and Definition 3543.3 seed the completion-reflection boundary over separated zero-distance evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SeparatedCompletionReflectionUp.SeparatedCompletionReflectionTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient completion, selected representatives, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-window exactness, zero-distance exactness, transport replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionReflectionUp.

Chapter 2585

A Concrete Naming Certificate for $\mathsf{ConvergentNet}^{\uparrow}$

Following Moore–Smith and Kelley, the $\mathsf{ConvergentNet}^{\uparrow}$ packet records ordinary convergent-net evidence as finite BEDC naming data. It sits over the directed-index substrate of chapter 1607, beside the Cauchy-tail surface of chapter 589, and before the convergence boundary of chapter 1521 and chapter 1561. The same finite route is readable by the completion consumers in chapter 1473 and chapter 1506, without turning convergence into a quotient of nets, a selected host limit, countable choice, or an ambient completeness assertion.

Definition 2585.1 (ConvergentNet carrier)**.** *A* $\mathsf{ConvergentNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, V, T, L, M, C, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index-window row, naming the inspected indices, tail windows, and displayed upper-window witnesses.*
2. *V is the net-value row over the indices displayed by D.*
3. *T is the convergence-test row, recording the finite neighborhoods, entourages, or metric tolerances that the packet actually inspects.*
4. *L is the displayed boundary row toward which the packet tests convergence, read only as finite naming data.*
5. *M is the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *boundary row that packages the same directed-window, value, test, and boundary evidence.*
6. *C is the optional* $\mathsf{CauchyNet}^{\uparrow}$ *tail row consumed by completion-facing readers that need Cauchy evidence.*
7. *H records componentwise* hsame *transport for the directed-window, value, test, boundary, convergence, Cauchy-tail, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from directed windows through net values, tests, boundary reads, Moore–Smith convergence, optional Cauchy-tail evidence, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{ConvergentNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient nets, selected host limits, countable-choice selectors, cofinal choice functions, host equality of boundary points, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ConvergentNetUp.ConvergentNetTasteGate single carrier alignment`

Theorem 2585.2 (ConvergentNet NameCert obligations)**.** *For every accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *carrier* $G = (D, V, T, L, M, C, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed directed index windows, net values, convergence tests, boundary row, Moore–Smith row, Cauchy-tail row, transport, continuation, provenance, and local naming;*
2. *directed-window coverage, requiring every convergence test in T to be attached to a tail window displayed by D;*
3. *value exposure, requiring each accepted test to inspect only net values displayed by V;*
4. *boundary exposure, requiring L to be read only after D, V, T have been displayed;*
5. *Moore–Smith packaging, requiring M to preserve the same finite route before any completion-facing consumer reads it;*
6. *completion handoff readiness, requiring the optional Cauchy-tail row C before* $\mathsf{CauchyNetLimit}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumes the packet;*
7. *ledger refusal, excluding quotient nets, selected host limits, countable choice, cofinal choice, host equality of limits, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3544.1. The tuple exposes directed windows, net values, convergence tests, a boundary row, the Moore–Smith convergence row, the optional Cauchy-tail row, and the structural rows H, K, P, N.

Directed-window coverage and value exposure are forced by the continuation row K. A convergence test is replayed first through D, then through V, and only then reaches T and the boundary row L.

The Moore–Smith row M packages the same displayed route. If a completion-facing reader also needs Cauchy evidence, it must consume the row C before the route can be handed to $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, or $\mathsf{MetricCompletion}^{\uparrow}$.

Since the carrier has no coordinate for quotient nets, selected limits, choice operators, host equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2585.3 (ConvergentNet Cauchy handoff)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Any completion-facing consumer that reads G as Cauchy-directed evidence must first display the* $\mathsf{DirectedSet}^{\uparrow}$ *index substrate D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, and then the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row C. The handoff preserves the finite directed-net route toward chapter 1561, chapter 1473, and chapter 1506, and exports no quotient net, selected limit, countable-choice selector, cofinal-choice operator, host equality, or ambient completeness principle.*

Proof. Start from a completion-facing read of the packet. Carrier admission gives the directed-window substrate D and the values V, but no completion consumer is visible before the convergence tests and boundary row have been replayed.

Use the continuation row K. It routes the read through D, V, T, L and then into the Moore–Smith row M, so the convergence evidence remains a finite directed-net display.

For a Cauchy-directed completion consumer, the row C is the required tail-evidence handoff. After C is displayed, the same finite route can be consumed by the directed-net limit, uniform-completion, or metric-completion surface.

The omitted coordinates are exactly the forbidden exits: quotient nets, selected limits, countable choice, cofinal choice, host equality, and ambient completeness. Hence the handoff preserves only the displayed finite route. □

Theorem 2585.4 (ConvergentNet completion-consumer boundary)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of G must read the* $\mathsf{DirectedSet}^{\uparrow}$ *row D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, and the optional* $\mathsf{CauchyNet}^{\uparrow}$ *tail-evidence row C in that order before any completion-facing handoff is available. The consumer receives only the finite route named by $D, V, T, L, M, C, H, K, P, N$ and receives no selected host limit, quotient net, countable-choice selector, cofinal-choice operator, host equality of boundary points, or ambient completeness principle.*

Proof. Project the carrier of Definition 3544.1. The only rows visible to a completion consumer are the directed-window row D, the value row V, the test row T, the boundary row L, the Moore–Smith row M, the optional Cauchy-tail row C, and the structural rows H, K, P, N.

Use Definition 3544.2. Its directed-window coverage, value exposure, boundary exposure, and Moore–Smith packaging clauses force the read order D, V, T, L, M before any completion-facing consumer can see convergence evidence.

Then apply Definition 3544.3. A metric-completion or uniform-completion consumer that needs Cauchy-directed evidence must next read C, so the completion-facing route is the displayed finite route through D, V, T, L, M, C.

Since the carrier omits selected host limits, quotient nets, choice operators, host equality, and ambient completeness, no completion consumer can bypass this order or turn the convergence packet into a selected limit object. □

Theorem 2585.5 (ConvergentNet limit handoff)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Every accepted limit read of G must expose the* $\mathsf{DirectedSet}^{\uparrow}$ *row D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, the optional* $\mathsf{CauchyNet}^{\uparrow}$ *row C, componentwise transport H, continuation replay K, provenance P, and the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *row N before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer can read the packet as completion-facing data. The read exports the Moore–Smith boundary only through this finite row order and exports no quotient net, selected host limit, countable-choice selector, cofinal-choice operator, host equality of boundary points, or ambient completeness principle.*

Proof. Apply Definition 3544.2. Its carrier admission and exposure clauses display exactly $D, V, T, L, M, C, H, K, P,$ while its ledger refusal excludes quotient nets, selected limits, choice operators, host equality, and ambient completeness.

The limit-facing part is the Moore–Smith boundary row. Boundary exposure and Moore–Smith packaging force a read through D, V, T, L, M before the packet can be treated as a convergent-net boundary.

For completion consumers, apply Definition 3544.3. The optional Cauchy-tail row C is the only displayed Cauchy-directed handoff, and it is read after the Moore–Smith route rather than as a separate completion object.

Finally, use Definition 3544.4. It states that $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers receive only the finite route named by the carrier rows, so the accepted limit read is exactly the Moore–Smith boundary row plus the listed structural support before completion consumption. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvergentNet}^{\uparrow}$ packet is generated from a DirectedSet index-window row, net values, convergence tests, a displayed boundary row, Moore–Smith convergence data, optional CauchyNet tail evidence, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3544.1, Definition 3544.2, Definition 3544.3, Definition 3544.4, and Definition 3544.5 seed the finite convergent-net surface between directed index data, Moore–Smith convergence, limit handoff, and completion-consumer handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected host limits, countable choice, cofinal choice, host equality of boundary points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window coverage, value exposure, boundary exposure, Moore–Smith packaging, Cauchy handoff, provenance, and ledger refusal for the same finite packet.

Chapter 2586

A Concrete Naming Certificate for $\mathsf{MedianClosure}^{\uparrow}$

The $\mathsf{MedianClosure}^{\uparrow}$ packet realizes the coordinatewise-majority closure surface of Definition 9462.1319 and Definition 9462.1320 as finite NameCert data. It records the reassigned support, the majority-saturation row, inactive-corner readback, f-vector ledger, and maximal-cell ledger for the 20-codon WNR∪CUN carrier. It sits beside the sibling finite-structure packets in chapter 2309, chapter 349, and chapter 112, but is not bijective with them because it carries the coordinatewise-majority saturation row.

Definition 2586.1 (MedianClosure carrier)**.** *A* $\mathsf{MedianClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, M, I, F, X, H, C, P, N)$$

with the following displayed rows:

1. *S is the reassigned support row, naming the* 13 *codons of the source support $\mathcal{R}$.*
2. *M is the coordinatewise-majority saturation row, displaying exactly the* 20 *codons of* $\mathrm{WNR} \cup \mathrm{CUN}$ *as the median closure of S.*
3. *I is the inactive-corner readback row, displaying the seven codons in* $\mathrm{Med}(\mathcal{R}) \setminus \mathcal{R}$ *and the four codons in* $\mathrm{Med}(\mathcal{R}) \setminus \overline{\mathcal{R}}$.
4. *F is the f-vector ledger for $K(\mathrm{Med}(\mathcal{R}))$, displaying* $(20, 38, 26, 8, 1, 0, \ldots)$.
5. *X is the maximal-cell ledger, displaying the* WNR 4*-cube, the* CUN 2*-cube, and the listed bridge cells from the source vision.*
6. *H records componentwise* hsame *transport for support, median saturation, inactive-corner, f-vector, maximal-cell, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from support through coordinatewise-majority saturation, inactive-corner readback, f-vector ledger, maximal-cell ledger, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{MedianClosure}^{\uparrow}$, $\mathsf{HellyTheorem}^{\uparrow}$, $\mathsf{Trie}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$ *row.*

The carrier contains no coordinate for the full Q_6 convex hull, no Helly theorem, no cubical homology package, no trie dictionary order, and no abstract finite-set equality outside the displayed codon rows.

Theorem 2586.2 (MedianClosure NameCert obligations)**.** *For every accepted* $\mathsf{MedianClosure}^{\uparrow}$ *carrier* $Q = (S, M, I, F, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$ *surface consists exactly of support admission through* S*, coordinatewise-majority saturation through* M*, inactive-corner readback through* I*, f-vector exposure through* F*, maximal-cell exposure through* X*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MedianClosureUp.MedianClosureTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3545.1. The support row, median-saturation row, inactive-corner row, f-vector ledger, maximal-cell ledger, and structural rows are explicit coordinates of the same finite BHist packet.

The continuation row C fixes the order of reads. A consumer first sees the reassigned support S, then the coordinatewise-majority saturation row M, and only then the inactive-corner, f-vector, and maximal-cell rows.

Transport, provenance, and local naming preserve that same route. Since the carrier omits convex-hull completion, Helly theorem data, cubical homology, trie order, and abstract finite-set equality, the listed rows exhaust the local NameCert surface. □

Theorem 2586.3 (MedianClosure vision realization)**.** *Every accepted* $\mathsf{MedianClosure}^{\uparrow}$ *packet realizes the median-closure claims of Definition 9462.1319, Definition 9462.1320, Definition 9462.1321, Definition 9462.1323, and Definition 9462.1324 as one finite NameCert surface. In particular, the displayed closure-size row rejects any carrier whose median-saturation row is not the* 20*-codon set* $\mathrm{WNR} \cup \mathrm{CUN}$*.*

Proof. Read the support row S and the median-saturation row M. By carrier admission, M is the displayed coordinatewise-majority closure of the reassigned support and has the 20 codons $\mathrm{WNR} \cup \mathrm{CUN}$.

The inactive-corner row I and f-vector row F are downstream of M, so their readback is tied to the same displayed closure rather than to an independent finite set.

The maximal-cell row X uses that same closure to expose the WNR and CUN cells and their bridge cells. Therefore any carrier with a different median closure cannot satisfy the local NameCert route. □

Falsifiable prediction: The $\mathsf{MedianClosure}^{\uparrow}$ closure-size row rejects any carrier whose displayed median closure is not the 20-codon $\mathrm{WNR} \cup \mathrm{CUN}$ set.

Independence witness: helly theorem, cubical finite-support chapters, trie, finset: none is bijective with $\mathsf{MedianClosure}^{\uparrow}$ because this carrier alone binds the coordinatewise-majority saturation row to the inactive-corner readback, f-vector ledger, and maximal-cell ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MedianClosure}^{\uparrow}$ packet is generated from the reassigned support row, coordinatewise-majority saturation, inactive-corner readback, f-vector ledger, maximal-cell ledger, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$, seed)

Scope closed: Definition 3545.1, Definition 3545.2, and Definition 3545.3 seed the finite median-closure packet for the 20-codon $\mathrm{WNR} \cup \mathrm{CUN}$ carrier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the full Q_6 convex hull, Helly theorem content, cubical homology, trie ordering, abstract finite-set equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support admission, majority saturation, inactive-corner readback, f-vector exposure, maximal-cell exposure, provenance, and ledger refusal for the same finite packet.

Chapter 2587

A Concrete Naming Certificate for $\mathsf{RegularCauchyReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyReal}^{\uparrow}$ packet presents a real number by regular Cauchy rational data with an explicit finite tolerance route before any quotient, selected limit, or ambient completeness principle is read. It sits on the finite observation surface of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the Cauchy-convergence handoff of chapter 1509, and the terminal real-name seal of chapter 38. The packet names the non-quotient regular Cauchy route used by L10 Real and Completion consumers; it does not assert that every Cauchy object has a chosen limit.

Definition 2587.1 (RegularCauchyReal carrier)**.** *A* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule listing the rational observations opened for each displayed tolerance request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic error rows, comparison bounds, and late-window inequalities.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled stream windows and dyadic tolerance ledger.*
4. *K is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff row, checking that the displayed regular readback satisfies the finite convergence criterion exposed to completion consumers.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, Q, K have all been displayed.*
6. *H records componentwise* hsame *transport for the stream schedule, dyadic ledger, regular readback, convergence handoff, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a regular-Cauchy-real request through S, D, Q, K, and E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRealUp.RegularCauchyRealTasteGate single carrier alignment`

Theorem 2587.2 (RegularCauchyReal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier*

$$R = (S, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$ *surface consists exactly of carrier admission, finite stream-window exposure, dyadic tolerance exactness, regular rational readback, Cauchy-convergence handoff, Real-seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3546.1. The accepted packet exposes the stream schedule S, dyadic tolerance ledger D, regular readback Q, convergence handoff K, terminal Real seal E, and structural rows H, C, P, N.

The obligation rows are then read in order. The finite windows of chapter 126 supply the observations, the dyadic comparisons of chapter 291 supply the tolerance ledger, and the readback surface of chapter 301 packages them as regular rational data.

The convergence and seal rows close the route without adding an ambient limit. The handoff through chapter 1509 exposes only the displayed criterion row K, and the Real seal of chapter 38 is reached only through E.

Finally H and C transport and replay the same coordinates, while P and N record provenance and local naming. Since no additional coordinate is present, the certificate surface exports none of the excluded quotient, choice, completeness, host-equality, verification, or bridge data. □

Theorem 2587.3 (RegularCauchyReal Real seal route)**.** *Every Real-facing read exported by an accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, Q, K \longmapsto E$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Thus a consumer receives scheduled stream windows, dyadic tolerance rows, regular rational readback, and the Cauchy-convergence handoff before any Real seal is visible.*

Proof. Start with an accepted carrier $R = (S, D, Q, K, E, H, C, P, N)$. By Definition 3546.2, the only Real-facing row is E, and it is downstream of the displayed stream, dyadic, regular-readback, and convergence rows.

The route cannot skip the finite observation stage because S is the only coordinate containing opened stream windows. It cannot skip the tolerance stage because D is the only coordinate containing dyadic comparison and late-window information.

The regular readback and convergence handoff are also forced. The row Q is the sole $\mathsf{RegSeqRat}^{\uparrow}$ readback, while K is the sole $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff available to a completion consumer.

Transport, replay, provenance, and local naming preserve the same route. Since H, C, P, N have no quotient, selected-limit, countable-choice, or ambient-completeness coordinate, every Real-facing read reaches E only through the displayed route. □

Theorem 2587.4 (RegularCauchyReal seal factorization)**.** *Every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *Real seal factors through the L10 four-object route*

$$\mathsf{StreamName}^{\uparrow}(S) \longmapsto \mathsf{DyadicRatCore}^{\uparrow}(D) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(Q) \longmapsto \mathsf{CauchyConvergenceCriterion}^{\uparrow}(K) \longmapsto \mathsf{Real}^{\uparrow}(E)$$

before any consumer can expose the terminal $\mathsf{Real}^{\uparrow}$ *row. The factorization is carried by the same* H, C, P, N *support rows of the carrier, and it supplies no quotient equality of Cauchy names, no selected limit, no host real equality, and no ambient completeness principle.*

Proof. Project the accepted carrier $R = (S, D, Q, K, E, H, C, P, N)$ through Definition 3546.1. The rows S, D, Q, K, E are the only analytic rows available to a Real-facing consumer.

The sibling route is ordered by the carrier itself. The finite windows of chapter 126 are read from S, the tolerance comparisons of chapter 291 are read from D, and the regular rational readback of chapter 301 is read from Q.

Apply Definition 3546.3 and Definition 3546.5. The first theorem forces every Real-facing read through S, D, Q, K before E, while the second identifies K as the displayed Cauchy-convergence handoff rather than an ambient completeness theorem.

Finally use the support rows H, C, P, N. They transport, replay, source, and name exactly the displayed route, so the terminal $\mathsf{Real}^{\uparrow}(E)$ seal has no coordinate from which quotient equality, selected limits, host real equality, or ambient completeness could be recovered. □

Theorem 2587.5 (RegularCauchyReal Cauchy-criterion handoff)**.** *Every completion-facing request for an accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier reads the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row* K *only after the stream schedule* S*, dyadic tolerance ledger* D*, and regular rational readback* Q *have been displayed. The handoff therefore supplies a finite criterion witness, not an ambient proof that an arbitrary Cauchy object has a chosen Real limit.*

Proof. Project the carrier rows of Definition 3546.1. The row K is listed after S, D, Q and is defined as the convergence handoff for that displayed regular readback.

The sibling routes determine the inputs to K. The schedule row S gives the finite inspected windows, the dyadic ledger D supplies the tolerance comparisons, and the regular rational row Q packages those reads into the source expected by the convergence criterion.

Thus a completion-facing consumer cannot read K as a standalone completeness principle. The only accepted handoff is the displayed finite route through S, D, Q, K, transported by H, replayed by C, and named by P, N. □

Theorem 2587.6 (RegularCauchyReal Cauchy-modulus source lock)**.** *Every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *packet must read its displayed Cauchy modulus or convergence budget before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row can be sealed as* $\mathsf{Real}^{\uparrow}$*. Concretely, any Real-facing consumer of*

$$R = (S, D, Q, K, E, H, C, P, N)$$

must expose the $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *budget row* K*, with the same source discipline as the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *interface, before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed.*

Proof. Project the rows $S, D, Q, K, E, H, C, P, N$ from Definition 3546.1. The stream schedule S is the only row exposing finite observations, and D is the only row exposing dyadic tolerance and late-window comparisons.

The readback row Q is therefore source-locked to S and D. The convergence row K records the displayed Cauchy budget for that readback, matching the modulus-source role of chapter 292 while staying inside the carrier rows already present in the packet.

Only after S, D, Q, K have been read can the terminal row E serve as the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The seal is not an independent limit selector; it is downstream of the StreamName, DyadicRatCore, RegSeqRat, and Cauchy-budget route.

Finally H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no selected limit, quotient stream equality, countable-choice row, host equality, or ambient completeness coordinate, no such datum can enter the Real seal. □

Theorem 2587.7 (RegularCauchyReal non-escape)**.** *An accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier exports no quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, host real equality, Lean verification, or bridge credential. Every exported read is confined to* $S, D, Q, K, E, H, C, P, N$.

Proof. Use the carrier definition to enumerate the available coordinates. They are exactly the stream schedule, dyadic ledger, regular readback, convergence handoff, Real seal, transport, replay, provenance, and local naming rows.

The obligation theorem identifies the NameCert surface with those same rows. Any proposed exported read must therefore be one of the displayed coordinates or a structural transport of them through H and C.

None of the displayed coordinates stores quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, host real equality, Lean verification, or bridge data. Hence no such datum can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3546.1, Definition 3546.2, Definition 3546.3, Definition 3546.5, Definition 3546.6, and Definition 3546.7 seed the non-quotient regular Cauchy real-name packet over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyRealUp.RegularCauchyRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance exactness, regular rational readback, convergence handoff, Real-seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchyRealUp.

Constructive story: A $\mathsf{RegularCauchyReal}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2588

A Concrete Naming Certificate for $\mathsf{GDeltaSet}^{\uparrow}$

Following Kechris and Engelking on Polish spaces and descriptive set theory, the $\mathsf{GDeltaSet}^{\uparrow}$ packet records the traditional countable-intersection-of-open-sets interface as finite BEDC NameCert data. It sits over the complete separable metric source of chapter 1490, the generated-set boundary of chapter 3384, the countable-base discipline of chapter 1901, and the indexed density surface of chapter 1566. The packet names the route by which Polish and Borel consumers inspect a displayed open-row schedule without importing a host subset, selected enumeration, classical complement closure, ambient countable choice, or a standard descriptive-set theorem.

Definition 2588.1 (GDeltaSet carrier)**.** *A* $\mathsf{GDeltaSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, B, S, W, O, H, C, L, N)$$

with the following displayed rows:

1. *P is the Polish and second-countable source row. It exposes the* $\mathsf{PolishSpace}^{\uparrow}$ *complete-separable metric packet, the* $\mathsf{SecondCountable}^{\uparrow}$ *base row, and the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *indexed-density handoff visible to the current consumer.*
2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *row that receives the generated-set handoff after the displayed open schedule has been inspected.*
3. *S is the countable open-row schedule. Each finite prefix names the open rows whose intersection is being offered, together with the schedule position and the source row over which the open row is read.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records exactly the schedule positions, base-cell indices, density indices, and finite query windows visible before the consumer reads the Borel row.*
5. *O is the open-set readback row, carrying the displayed open classifier, local basis witness, and source compatibility proof for every open row inspected through S and W.*
6. *H records componentwise* hsame *transport for the Polish source, Borel row, open schedule, stream windows, open readback, continuation replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Polish or Borel query through P, S, W, O before the generated-set row B is consumed.*

8. *L is the* Pkg *provenance over* $\mathsf{GDeltaSet}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, B, S, W, O, H, C, L, N$. It has no coordinate for host subsets, chosen countable enumerations, ambient countable choice, quotient set equality, classical complement closure, or standard descriptive-set transfer theorems.

Lean4 ✓ `BEDC.Derived.GDeltaSetUp.GDeltaSetTasteGate single carrier alignment`

Theorem 2588.2 (GDeltaSet NameCert obligations)**.** *For every accepted* $\mathsf{GDeltaSet}^{\uparrow}$ *carrier* $G = (P, B, S, W, O, H, C, L, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Polish and second-countable source row, Borel row, open-row schedule, finite windows, open readback, transport, replay, provenance, and local naming;*
2. *countable schedule exposure, requiring every inspected intersection stage to appear as a displayed finite prefix of S before it reaches B;*
3. *finite-window stability, requiring each schedule position, base-cell index, density index, and query window to be read through W;*
4. *open-row readback, requiring each open component to pass through the classifier and local-basis data stored in O over the same source row P;*
5. *Borel and Polish handoff, requiring the generated-set row B to consume only the displayed route through P, S, W, O;*
6. *componentwise transport, requiring H to preserve the source, schedule, windows, open readback, Borel row, replay, provenance, and local name together;*
7. *continuation replay, requiring C to replay the same ordered route before any Polish or Borel consumer receives the packet;*
8. *provenance, requiring L to name exactly the displayed sibling packets and BEDC kernel rows used by the carrier;*
9. *local naming, requiring N to certify only the rows P, B, S, W, O, H, C, L;*
10. *ledger refusal, excluding host countable choice, quotient set equality, selected enumerations, host subsets, classical complement closure, and ambient descriptive-set theorems.*

No $\mathsf{GDeltaSet}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 3547.1. The tuple displays exactly the Polish and second-countable source row, Borel row, countable open-row schedule, finite stream windows, open readback row, transport, replay, provenance, and local naming.

The schedule exposure and finite-window obligations are read through S and W. A consumer sees only a finite prefix of the countable open schedule, and each schedule position, base-cell index, density index, and query window is inspected through the same displayed stream-window row.

Open-row readback is forced by O over the source row P. Thus each open component is tied to the Polish, second-countable, and indexed-density data before the generated-set row B can be consumed.

Finally H, C, L, N transport, replay, source, and name the same route. Since no listed row stores a host subset, chosen enumeration, countable-choice selector, quotient set equality, classical

complement closure, or standard descriptive-set theorem, the enumerated obligations exhaust the local NameCert surface. □

Theorem 2588.3 (GDeltaSet Polish-Borel handoff)**.** *Every Polish or Borel consumer of an accepted* $\mathsf{GDeltaSet}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{PolishSpace}^{\uparrow},\ \mathsf{SecondCountable}^{\uparrow},\ \mathsf{CountableDenseSubset}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BorelSet}^{\uparrow}.$$

More precisely, the consumer first reads the source row P, then the schedule and finite windows S, W, then the open-row readback O, and only then the generated-set row B. The exported object is a finite NameCert packet for the displayed countable open-row route; it is not a host subset, a chosen enumeration, a classical complement-closure proof, an ambient standard descriptive-set theorem, or a quotient set equality.

Proof. Begin with a Polish or Borel consumer of the accepted packet. Carrier admission provides the source row P, but no generated-set read is available until the open schedule and window rows have been displayed.

Replay the continuation row C. The read moves through S and W, so every countable-intersection stage is consumed only as a finite visible prefix with finite schedule, base-cell, density, and query observations.

The open readback row O then ties the visible open components back to the Polish and second-countable source. After that readback, the Borel row B receives the generated-set handoff.

The remaining rows H, L, N preserve, source, and name the same route. Therefore every consumer factors through the displayed schedule, window, and open-row route, with no host subset, chosen enumeration, classical complement closure, standard descriptive-set theorem, or quotient set equality exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3547.1, Definition 3547.2, and Definition 3547.3 seed the finite GDeltaSet route over Polish, second-countable, countable-dense, StreamName, open-readback, Borel, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, selected countable enumerations, countable choice, quotient set equality, classical complement closure, standard descriptive-set theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable schedule exposure, finite-window stability, open-row readback, Polish-Borel handoff, transport, replay, provenance, local naming, and ledger refusal for BEDC.Derived.GDeltaSetUp.

Constructive story: A $\mathsf{GDeltaSet}^{\uparrow}$ packet is generated from a Polish and second-countable source row, a BorelSet row, a countable open-row schedule, finite StreamName windows, open-set readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2589

A Concrete Naming Certificate for $\mathsf{ClosedBallCompactness}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ClosedBallCompactness}^{\uparrow}$ packet records the theorem-facing finite certificate that every displayed closed ball in a proper metric carrier is compact. It sits between the closed-ball membership surface of chapter 1451, the proper-metric witness of **??**, the nested-ball completion route of chapter 3526, the compact-metric certificate of chapter 129, and the total-bounded plus complete-metric route through chapter 127 and chapter 128. The packet names the finite closed-ball compactness handoff only; it is not an open-cover compactness theorem, not a host Heine–Borel proof, not an arbitrary properness oracle, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2589.1 (Closed-ball compactness carrier)**.** *A* $\mathsf{ClosedBallCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, P, N, K, T, M, R, H, D, Q, A)$$

with the following displayed rows:

1. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row naming the center, radius, metric source, and closed membership comparison.*
2. *P is the* $\mathsf{ProperMetric}^{\uparrow}$ *row whose displayed closed-ball family contains the same center-radius window.*
3. *N is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *route available for Cauchy-center and descending-ball handoff reads.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness row attached to the displayed closed ball.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row exposed by the compact metric witness.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row exposed by the compact metric witness and by the proper-metric completion handoff.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for center, radius, distance, and compactness-window comparisons.*
8. *H records componentwise* hsame *transport for the closed-ball, proper-metric, nested-ball, compact-metric, totally-bounded, complete-metric, real-readback, replay, provenance, and naming rows.*
9. *D records displayed* Cont *replay from a closed-ball compactness request through B, P, N, K, T, M, R.*

10. *Q is the* Pkg *provenance over* ClosedBallCompactness$^{\uparrow}$, MetricClosedBall$^{\uparrow}$, ProperMetric$^{\uparrow}$, NestedClosedBall$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *A is the local* NameCert$_{\text{ClosedBallCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their displayed transports, their continuation routes, provenance, and local naming. It has no coordinate for open covers, selected subcovers, quotient metric points, a host compactness theorem, arbitrary properness beyond the shown ball, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.ClosedBallCompactnessUp.ClosedBallCompactnessTasteGate single carrier alignment`

Theorem 2589.2 (Closed-ball compactness NameCert obligations)**.** *For every accepted* ClosedBallCompactness$^{\uparrow}$ *carrier*

$$C = (B, P, N, K, T, M, R, H, D, Q, A),$$

the local NameCert$_{\text{ClosedBallCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric closed-ball row, proper-metric row, nested-ball route, compact-metric witness, totally-bounded row, complete-metric row, real-readback row, componentwise transport, displayed continuation replay, package provenance, and local naming; center-radius compatibility between B and the closed-ball family displayed by P; compact witness exactness through K; finite-net and complete-limit exposure through T and M; and ledger refusal for open-cover compactness, selected subcovers, quotient metric points, host compactness, arbitrary properness outside the displayed ball, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3548.1. The accepted packet displays the closed-ball row, the proper-metric source, the nested-ball route, the compact witness, the total-bounded and complete-metric witnesses, the real readback, and the structural rows.

Center-radius compatibility is forced before compactness is read. The row B names the closed-ball membership comparison, and P exposes the same center-radius window in its displayed proper-metric family, so a consumer cannot switch to another ball.

Compact witness exactness is the route through K. The finite-net face T and the limit-service face M are read only after K has named the compact metric witness attached to B.

The structural rows H, D, Q, A transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, open-cover compactness, selected subcovers, quotient point equality, host compactness, arbitrary properness outside the displayed ball, and ambient real completeness cannot enter the local NameCert surface. □

Theorem 2589.3 (Closed-ball compactness nested-ball handoff)**.** *Every nested-ball or completion-facing consumer of an accepted* ClosedBallCompactness$^{\uparrow}$ *carrier reads the closed ball through*

MetricClosedBall$^{\uparrow}$, ProperMetric$^{\uparrow}$, NestedClosedBall$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CompleteMetric$^{\uparrow}$

The handoff therefore exposes the displayed closed-ball membership row, the proper-metric compact witness, the nested-ball Cauchy-center route, the finite-net face, and the complete-metric face, and it exports no open-cover compactness proof or choice-selected center.

Proof. Begin with the accepted carrier $C = (B, P, N, K, T, M, R, H, D, Q, A)$. The closed-ball request first reads B and the matching proper-metric row P.

The nested-ball route N is then available only as a displayed handoff. It supplies the descending-ball and Cauchy-center route used by completion consumers, while K supplies the compact metric witness attached to the same closed ball.

Finally T, M, R, H, D, Q, A expose the finite-net, limit-service, real-readback, transport, replay, provenance, and naming rows. Because the carrier has no selector coordinate, the consumer receives no open-cover compactness proof and no choice-selected center. □

Theorem 2589.4 (Closed-ball compactness nonescape)**.** *Every public read of an accepted* $\mathsf{ClosedBallCompactness}^{\uparrow}$ *packet factors through the carrier of Definition 3548.1 and the obligations of Definition 3548.2. In particular, the packet can export compactness for the displayed closed ball only as the route*

$$B, P, N, K, T, M, R, H, D, Q, A,$$

and cannot export open-cover compactness, a selected finite subcover, quotient equality of metric points, a host Heine–Borel theorem, arbitrary properness beyond the displayed center-radius window, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows B, P, N, K, T, M, R are the only mathematical rows available before structural replay, and the rows H, D, Q, A are the only transport, continuation, provenance, and naming rows.

Apply Definition 3548.2. The local NameCert surface requires the closed-ball row, proper-metric row, nested-ball route, compact witness, total-bounded face, complete-metric face, and real-readback face to be admitted together.

Any public compactness read must therefore pass through the same finite tuple. Since the tuple has no coordinate for the excluded objects, those objects cannot escape through the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBallCompactness}^{\uparrow}$ packet is generated from a MetricClosedBall row, ProperMetric closed-ball family row, NestedClosedBall route, CompactMetric witness, TotallyBounded finite-net row, CompleteMetric limit-service row, RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3548.1, Definition 3548.2, Definition 3548.3, and Definition 3548.4 seed the theorem-facing finite compactness certificate for displayed proper-metric closed balls.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient metric points, host Heine–Borel compactness, arbitrary properness beyond displayed closed balls, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center-radius compatibility, compact witness exactness, nested-ball handoff, finite-net and complete-limit exposure, nonescape, and TasteGate field faithfulness for BEDC.Derived.ClosedBallCompactnessUp.

Chapter 2590

A Concrete Naming Certificate for $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$

Following the finite-net proof of compact uniform continuity in chapter 9237, the $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ packet records the residual triangle-estimate surface that remains after a compact source, continuous-map pointwise modulus, and uniform-modulus candidate have already been displayed. It sits over the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the metric triangle estimate packet of chapter 1159, and the uniform-modulus consumer of chapter 369. The packet names the final source-and-target triangle ledger only; it is not a compactness theorem, not a pointwise-modulus selector, not a finite-net constructor, and not a host proof of Heine–Cantor.

Definition 2590.1 (Compact-source triangle estimate carrier)**.** *A* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, U, E, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the finite-net coverage and located-distance comparisons for the compact source metric.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus and graph evaluations are inspected on the same compact source.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *candidate row produced by the finite-net lower-bound route.*
4. *E is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *residual row, recording the source triangle estimate around a selected net center and the target triangle estimate for the two image points.*
5. *S is the finite center-radius ledger that ties the selected compact-source probe to the pointwise modulus radius and to the uniform-modulus precision.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger for the epsilon-third, half-radius, and lower-bound inequalities used by E.*
7. *L is the local nonescape ledger saying that the estimate consumes only the displayed compact source, continuous-map graph, uniform-modulus candidate, triangle-estimate row, and rational comparisons.*
8. *H records componentwise* hsame *transport for the compact source, continuous map, uniform-modulus candidate, triangle estimate, center-radius ledger, rational comparison ledger, nonescape ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-uniform-continuity request through* K, F, U, E, S, R, L.
10. *P is the* Pkg *provenance over* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open cover, an external finite subset, a selected point outside the displayed probe row, a host supremum norm, or an independent Heine–Cantor theorem.

Lean4 ✓ `BEDC.Derived.CompactSourceTriangleEstimateUp.CompactSourceTriangleEstimateTasteGate`
`single carrier alignment`

Theorem 2590.2 (Compact-source triangle estimate NameCert obligations)**.** *For every accepted* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *carrier*

$$T = (K, F, U, E, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-map row, uniform-modulus candidate, metric triangle estimate, finite center-radius ledger, rational comparison ledger, local nonescape ledger, componentwise transport, displayed continuation replay, package provenance, and local naming; compact-source compatibility between K and the source row of F; triangle-estimate exactness through E; rational-bound exactness through R; and refusal of open covers, external finite subsets, hidden point selectors, host norms, and independent Heine–Cantor reasoning.*

Proof. Carrier admission is projection from Definition 3549.1. The accepted packet displays the compact source, continuous map, uniform-modulus candidate, triangle estimate, center-radius ledger, rational comparisons, and structural rows.

Compact-source compatibility is read first. The compact row K and continuous-map row F name the same source metric before any center-radius estimate in S can be consumed.

The residual estimate is the row E. It combines the source triangle comparison around the displayed compact-net center with the target triangle comparison for the two image points, while R records the epsilon-third and radius inequalities used by that comparison.

The rows L, H, C, P, N record nonescape, transport, replay, provenance, and naming. Since no excluded coordinate appears in the carrier, the packet cannot export an open cover, external finite subset, hidden selector, host norm, or independent Heine–Cantor proof. □

Theorem 2590.3 (Compact-source triangle estimate UniformModulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *packet must read the compact source, continuous-map graph, uniform-modulus candidate, metric triangle estimate, finite center-radius ledger, rational comparison ledger, and nonescape ledger before it reads the structural rows*

$$H, C, P, N.$$

Consequently the consumer receives a final compact-source triangle residual over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *and* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, *rather than a new compactness or continuity principle.*

Proof. Start with the accepted carrier $T = (K, F, U, E, S, R, L, H, C, P, N)$. The uniform-modulus consumer can read U only after K and F have identified the compact source and graph surface.

The estimate then passes through S, R, E. The center-radius ledger ties the selected compact-source probe to the pointwise-modulus radius and uniform precision, the rational ledger records the required inequalities, and the metric triangle estimate row supplies the source and target triangle steps.

The nonescape ledger L and structural rows H, C, P, N preserve the same route. Thus the handoff is a residual triangle-estimate packet for an already displayed compact-uniform route, not an additional source of compactness or continuity data. □

Falsifiable prediction: Every accepted $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ read must expose a $\mathsf{CompactMetric}^{\uparrow}$ source row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph row over the same source, a $\mathsf{UniformModulus}^{\uparrow}$ candidate row, a $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ residual row, finite center-radius and rational comparison ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric-triangle-uniform-estimate: $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ is not bijective to any listed sibling because it binds the final compact-source triangle residual after the finite-net and pointwise-modulus data have already been displayed.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ packet is generated from a CompactMetric source row, ContinuousMap graph row, UniformModulus candidate row, MetricTriangleUniformEstimate residual row, finite center-radius ledger, RatUp comparison ledger, local nonescape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3549.1, Definition 3549.2, and Definition 3549.3 seed the final compact-source triangle residual consumed by the compact-uniform-continuity route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, pointwise continuity, finite-net construction, selected centers outside displayed probes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source compatibility, triangle-estimate exactness, rational-bound exactness, UniformModulus handoff, nonescape, and TasteGate field faithfulness for BEDC.Derived.CompactSourceT

Chapter 2591

A Concrete Naming Certificate for $\mathsf{RealLineConnected}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLineConnected}^{\uparrow}$ packet records the finite connectedness surface of the real line as interval-chain and intermediate-value data visible before any host topological connectedness theorem is invoked. It sits over the real-name boundary of chapter 38, the interval connectedness route of chapter 1462, the intermediate-value bisection route of chapter 1402, the graph and modulus rows of chapter 130, and the dyadic tolerance ledger of chapter 291. The chapter names a constructive real-line connectedness packet for later continuous-image and completion consumers; it does not claim open-cover compactness, real trichotomy, quotient stream equality, selected paths, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2591.1 (Real line connected carrier)**.** *A* $\mathsf{RealLineConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, J, V, F, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *line-source row, exposing only terminal real-name reads through displayed finite windows;*
2. *J is the* $\mathsf{ConnectedInterval}^{\uparrow}$ *interval-chain row, recording the finite chain of located interval segments used by a connectedness consumer;*
3. *V is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row attached to the interval-chain handoff;*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus row inspected by continuous-image consumers of the line route;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the interval-chain, bisection, and finite-window requests;*
6. *W is the finite observation-window row opened by D before any real-line connectedness read is exported;*
7. *Q is the regular rational readback row attached to the same windows;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after J, V, F, D, W, Q have been displayed;*
9. *H stores componentwise* hsame *transport for the real-source, interval-chain, intermediate-value, continuous-map, dyadic, window, readback, seal, replay, provenance, and local naming rows;*

10. *C stores displayed* Cont *replay through the same rows;*
11. *P is the* Pkg *provenance over* $\mathsf{RealLineConnected}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConnectedInterval}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open-cover connectedness, trichotomy, quotient stream equality, a selected path object, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealLineConnectedUp.RealLineConnectedTasteGate single carrier alignment`

Theorem 2591.2 (Real line connected NameCert obligations)**.** *For every accepted* $\mathsf{RealLineConnected}^{\uparrow}$ *carrier*

$$L = (R, J, V, F, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real-source, interval-chain, intermediate-value, continuous-map, dyadic, finite-window, readback, terminal-seal, transport, replay, provenance, and local naming rows;*
2. *interval-chain visibility, requiring every connectedness read to pass through J before a continuous-image or completion consumer can inspect the terminal seal;*
3. *intermediate-value routing, requiring each separation-refusal read along the chain to cite V and its finite bisection route;*
4. *continuous-map handoff, requiring downstream image consumers to use F only after the same interval-chain and intermediate-value rows have been displayed;*
5. *dyadic-window exactness, requiring the tolerance row D, window row W, and readback row Q to name the same finite request before E is exported;*
6. *nonescape from host open-cover connectedness, trichotomy, quotient stream equality, selected path objects, host real equality, and ambient completeness.*

No $\mathsf{RealLineConnected}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 3550.1. The tuple lists the real-source, interval-chain, intermediate-value, continuous-map, dyadic, finite-window, readback, terminal-seal, transport, replay, provenance, and local naming rows.

Interval-chain visibility is read from J. A connectedness consumer must inspect the finite $\mathsf{ConnectedInterval}^{\uparrow}$ chain before it can treat the real line as connected data.

Intermediate-value routing is read from V. The only separation-refusal information available to the packet is the finite intermediate-value sign route attached to the displayed interval chain.

Continuous-map handoff is read from F. A continuous-image consumer therefore receives graph and modulus data only after the line-source, interval-chain, and intermediate-value rows have already been displayed.

Dyadic-window exactness is read from D, W, Q, E. The dyadic tolerance opens the finite window, the readback row names the regular rational data in that window, and the terminal seal exports the same request as $\mathsf{Real}^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name only the displayed packet. Since the carrier contains no host open-cover connectedness, trichotomy, quotient stream equality, selected path object, host real equality, or ambient completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 2591.3 (Real line connected interval-chain handoff)**.** *Every continuous-image or completion-facing read exported by an accepted* $\mathsf{RealLineConnected}^{\uparrow}$ *packet factors through*

$$\mathsf{ConnectedInterval}^{\uparrow} \longmapsto \mathsf{IntermediateValue}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route uses the displayed rows $J, V, F, D, W, Q, E, H, C, P, N$ *of Definition 3550.1. It exports no host open-cover connectedness theorem, trichotomy principle, quotient stream equality, selected path object, host real equality, or ambient completeness claim.*

Proof. Start with an accepted carrier $L = (R, J, V, F, D, W, Q, E, H, C, P, N)$. The source connectedness row is J, so any real-line connectedness read begins with the displayed $\mathsf{ConnectedInterval}^{\uparrow}$ chain rather than with a host connected-subset assertion.

Read the separation-refusal surface through V. This intermediate-value row records the finite bisection and sign-route data used by the chain; no trichotomy or selected separator is available.

Read the downstream continuous-map surface through F. The graph and modulus rows are attached to the same chain and sign-route data, so continuous-image consumers cannot bypass the interval-chain handoff.

Finally read the dyadic and real-name boundary through D, W, Q, E. The dyadic tolerance opens the finite window, the readback row presents the regular rational data, and the terminal seal exports the Real-facing datum.

The remaining rows H, C, P, N preserve the same displayed route. Since no row names open-cover compactness, quotient stream equality, selected paths, host real equality, or ambient completeness, every continuous-image or completion-facing read remains inside the finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3550.1, Definition 3550.2, and Definition 3550.3 seed the finite real-line connectedness route through interval chains, intermediate-value rows, continuous-map rows, dyadic windows, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLineConnectedUp.RealLineConnectedTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host open-cover connectedness, trichotomy, quotient stream equality, selected path objects, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, interval-chain visibility, intermediate-value routing, continuous-map handoff, dyadic-window exactness, and Real seal nonescape for BEDC.Derived.RealLineConnectedUp.

Constructive story: A $\mathsf{RealLineConnected}^{\uparrow}$ packet is generated from a Real source row, ConnectedInterval chain row, IntermediateValue sign route, ContinuousMap graph and modulus row, DyadicRatCore tolerance ledger, finite observation windows, regular readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2592

A Concrete Naming Certificate for $\mathsf{CauchyProductMetric}^{\uparrow}$

Following Bishop–Bridges on metric products and regular Cauchy names, the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet records the finite product-metric surface used by two displayed CauchyProduct rows before any host product metric or Real completeness principle is available. It sits beside the product route of chapter 464, the metric carrier surface of chapter 57, the regular rational readback of chapter 301, the scheduled window layer of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is the regular-Cauchy specialization consumed by chapter 424 when the product-distance ledger is read as a metric packet, and by chapter 1506 when paired Cauchy windows are passed to completion-facing consumers. The packet gives downstream Real, metric, and Cauchy-completion consumers a non-quotient product-distance certificate over displayed rows only.

Definition 2592.1 (CauchyProductMetric carrier)**.** *A* $\mathsf{CauchyProductMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows. Each row supplies a finite product-name route rather than host product equality.*
2. *W_L and W_R are the paired* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected on the left and right product names.*
3. *Q_L and Q_R are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the two displayed window pairs.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger. It records coordinate tolerances, the product-distance radius, the comparison budget, and the displayed error allowance used by the packet.*
5. *Δ is the* $\mathsf{Met}^{\uparrow}$ *distance row, read only through the displayed product windows, regular rational readbacks, and dyadic ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product-distance read.*
7. *H records componentwise* hsame *transport for the CauchyProduct rows, stream windows, regular readbacks, dyadic ledger, metric row, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a product-metric request through $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductMetric}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N$. It has no coordinate for host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyProductMetricUp.CauchyProductMetricTasteGate single carrier alignment`

Theorem 2592.2 (CauchyProductMetric NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *carrier $M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{CauchyProduct}^{\uparrow}$ *rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger,* $\mathsf{Met}^{\uparrow}$ *distance row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 3551.1. The accepted packet exposes exactly the two CauchyProduct rows, paired stream windows, regular rational readbacks, dyadic product-distance ledger, metric row, Real seal, and structural rows.

The product-distance ledger is downstream of the displayed windows and readbacks. Thus D cannot be read before W_L, W_R, Q_L, Q_R have fixed the finite CauchyProduct observations.

The metric and Real rows are then forced through Δ and E. The distance row reads only the displayed dyadic ledger, and the seal row records the resulting product-distance value without adding a second carrier.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host product metrics, quotient stream equality, selected limits, trichotomy, detached real carriers, or ambient completeness, the listed obligations exhaust the local certificate surface. □

Theorem 2592.3 (CauchyProductMetric projection stability)**.** *Let $M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P,$ be an accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet. Any projection-facing metric consumer that reads the left or right coordinate distance must factor through the same* $\mathsf{CauchyProduct}^{\uparrow}$ *source row, the same* $\mathsf{StreamName}^{\uparrow}$ *window row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger, and the same structural rows H, C, P, N. Projection reads may forget the unused coordinate, but they cannot introduce a host product metric, a quotient stream, a selected limit, or an independent real carrier.*

Proof. Start with a projection-facing read of the accepted packet. The only coordinate data exposed by the carrier are L, R together with their displayed window and regular readback rows.

For the left projection, the read passes through L, W_L, Q_L and then through the shared dyadic ledger D. For the right projection, the same argument uses R, W_R, Q_R and the same ledger.

The metric row Δ and Real seal E are downstream of this ledger. Forgetting one coordinate therefore removes data from the displayed packet, but it does not create a new metric source or a new Real-facing endpoint.

The structural rows H, C, P, N preserve exactly this projected route. The omitted coordinates in Definition 3551.1 rule out host product metrics, quotient streams, selected limits, and detached real carriers. □

Lean4 ✓ `BEDC.Derived.CauchyProductMetricUp.CauchyProductMetricProjectionStability`

Theorem 2592.4 (CauchyProductMetric Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet must first display the finite route*

$$\mathsf{CauchyProduct}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

through the rows $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N$*. The Real seal is a product-distance seal for this displayed route only; it exports no host product metric, no quotient product of Cauchy streams, no selected product limit, no trichotomy row, no detached real carrier, and no ambient completeness witness.*

Proof. Begin with a Real-facing read. Carrier admission makes the two CauchyProduct rows visible, but no Real seal is available before the paired stream windows and regular readbacks have been displayed.

The dyadic ledger D records the product-distance budget attached to those windows. The metric row Δ can only read this ledger, so it cannot import a host product metric or a separate distance object.

The seal E is reached after Δ, and the continuation row C replays the same route for downstream Real, metric, or completion-facing consumers. The transport and provenance rows keep the endpoint tied to $L, R, W_L, W_R, Q_L, Q_R, D, \Delta$.

Thus the consumer receives exactly the finite product-distance route named by the local certificate. All quotient, selection, trichotomy, detached-carrier, host-metric, and ambient-completeness exits are outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductMetric}^{\uparrow}$ packet is generated from two CauchyProduct source rows, paired StreamName windows, RegSeqRat readbacks, a DyadicRatCore product-distance ledger, a MetricUp distance row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3551.1, Definition 3551.2, Definition 3551.3, and Definition 3551.4 seed the finite product-metric route from CauchyProduct rows through displayed windows, regular readbacks, dyadic product-distance ledgers, metric rows, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-distance ledger exactness, projection stability, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductMetricUp.

Chapter 2593

A Concrete Naming Certificate for $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$

Following Bishop and Bridges on uniformly continuous maps between Cauchy presentations, the $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ packet records the finite morphism surface between regular Cauchy sequence spaces before any completion equality or limiting representative is read. It sits after the source-space surface of chapter 3541, the finite window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the Cauchy-name-space packet of chapter 406, and the completion-functor route of chapter 423. The packet is the L10 finite morphism interface consumed by completion-functor and Real-facing handoff chapters; it is not a proof of ambient completeness, quotient equality of streams, selected limits, or host equality of real names.

Definition 2593.1 (RegularCauchySpaceMorphism carrier)**.** *A* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, A, W, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, including the source regular rational family, rate ledger, finite schedule, dyadic tolerance route, CauchySequenceSpace handoff, and Real seal.*
2. *T is the target* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, carrying the same visible sequence-space structure for the codomain presentation.*
3. *A is the finite-window action row assigning each accepted source window request to a displayed target window request.*
4. *W is the dyadic budget ledger recording the input precision, output precision, and finite modulus rows used by A.*
5. *D is the classifier-transport row, binding source acceptance to target acceptance through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling routes.*
6. *H records componentwise* hsame *transport for source space, target space, action, budget ledger, classifier transport, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from each morphism request through the source finite window, action row, dyadic budget ledger, target finite window, target CauchySequenceSpace handoff, and target Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$, $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, A, W, D, H, C, P, N$. It has no coordinate for quotient stream equality, selected source or target limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySpaceMorphismUp.RegularCauchySpaceMorphismTasteGate single carrier alignment`

Theorem 2593.2 (RegularCauchySpaceMorphism NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ *carrier $M = (S, T, A, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$ *surface consists exactly of carrier admission, source-target compatibility, finite-window exactness, rate-ledger preservation, classifier transport, non-escape, displayed replay, provenance, and local naming.*

Source-target compatibility requires both S and T to be accepted $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *packets. Finite-window exactness requires A to transform only displayed source windows into displayed target windows. Rate-ledger preservation requires W to expose the dyadic input and output budget rows before any target seal is read. Classifier transport requires D to preserve the visible source and target classifiers through the same sibling routes named in the carrier. Non-escape refuses quotient stream equality, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as certificate data.*

Proof. Project the carrier of Definition 3552.1. The rows S and T supply the source and target regular Cauchy sequence spaces, while A supplies only a displayed finite-window action between them.

Next read the dyadic budget and classifier transport rows. The budget ledger W records the precision rows consumed by the action, and D binds acceptance of the source and target windows through the named StreamName, RegSeqRat, DyadicRatCore, CauchySequenceSpace, and Real sibling routes.

Finally attach H, C, P, N. These rows preserve and replay the same finite morphism packet, so every local naming obligation is exhausted by the displayed tuple. Since the tuple has no quotient, selected-limit, countable-choice, host-equality, completeness, verification, or bridge coordinate, the refused objects cannot be exported by the NameCert surface. □

Theorem 2593.3 (RegularCauchySpaceMorphism finite-window exactness)**.** *Every accepted* RegularCauchySpaceM *read is exhausted by the displayed source and target regular Cauchy sequence-space rows, the finite-window action row, the dyadic budget ledger, classifier transport, componentwise transport, replay, provenance, and local naming:*

$$S, T, A, W, D, H, C, P, N.$$

Thus the morphism exports a finite L10 action between regular Cauchy sequence spaces. It exports no quotient stream equality, selected source limit, selected target limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted carrier $M = (S, T, A, W, D, H, C, P, N)$. The only source-space and target-space data available to the consumer are the displayed rows S and T, each read as a $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ packet.

Use Definition 3552.2. It places the action A behind finite windows and places the precision behavior behind the dyadic ledger W and classifier transport row D.

The structural rows H, C, P, N then transport, replay, source, and name only this route. No remaining coordinate can supply quotient stream equality, selected limits, countable choice, host real equality, or ambient real completeness, so every accepted morphism read is finite-window exact in the displayed sense. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ packet is generated from source and target RegularCauchySequenceSpaceUp rows, a finite-window action, a dyadic budget ledger, classifier transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3552.1, Definition 3552.2, and Definition 3552.3 seed the finite morphism surface between regular Cauchy sequence spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-target compatibility, finite-window exactness, rate-ledger preservation, classifier transport, non-escape, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.RegularCauchySpaceMorphismUp.

Chapter 2594

A Concrete Naming Certificate for $\mathsf{MetricCompletionComparison}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive metric completions, the $\mathsf{MetricCompletionComparison}^{\uparrow}$ packet records the finite comparison route by which two dense completions of the same metric source are read as the same completion surface. It sits after the universal-property surface of chapter 2319, the dense-source row of chapter 3542, the metric dense-image packet of chapter 3351, the regular Cauchy sequence-space packet of chapter 3541, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names constructive uniqueness of dense metric completions through displayed comparison rows; it is not a quotient representative theorem, not a selected-limit operator, not countable choice, and not an ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Definition 2594.1 (MetricCompletionComparison carrier)**.** *A* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row used twice, once to extend the first dense embedding into the second completion and once in the opposite direction.*
2. *D is the* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *row that fixes the common dense source before either comparison map is read.*
3. *E_0 and E_1 are* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *rows for the two displayed dense completions of the same metric source.*
4. *S is the* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row recording the common regular Cauchy sequence-space interface inspected by both completions.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite regular rational windows of S.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the source, the two dense embeddings, and the comparison maps.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the precision budgets used to compare the two completion reads.*

8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* D, E_0, E_1, S, R, W, Q *have been displayed.*
9. *H records componentwise* hsame *transport for the dense-source, dense-embedding, sequence-space, readback, window, tolerance, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the two comparison directions and of the round-trip reads through the same finite rows.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionComparison}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N$ and their displayed replay paths. It has no coordinate for quotient representatives, selected Cauchy limits, host equality of completion classes, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionComparisonUp.MetricCompletionComparisonTasteGate single carrier alignment`

Theorem 2594.2 (MetricCompletionComparison NameCert obligations)**.** *For every accepted* MetricCompletionCor *carrier $K = (U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the universal-property row, common dense-source row, paired dense embeddings, regular Cauchy sequence-space row, regular rational readback, stream-window schedule, dyadic tolerance ledger, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. The two comparison directions must read the same D, S, R, W, Q, A rows before either round-trip read is accepted.*

Proof. Project the carrier of Definition 3553.1. The rows $D, E_0, E_1, S, R, W, Q, A$ display the common source, the paired dense embeddings, the regular Cauchy sequence-space interface, the finite readback windows, the dyadic tolerances, and the terminal real-name seal.

Apply the universal-property row U in the two displayed directions. The first use extends E_0 across the dense source into the second completion surface, and the second use extends E_1 back across the same dense source. Both uses are forced to read the same dense-source, sequence-space, regular-readback, stream-window, tolerance, and seal rows.

The structural rows H, C, P, N then preserve, replay, source, and name that finite route. Since the carrier contains no row for quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient real completeness, no additional NameCert obligation can be projected from the packet. □

Theorem 2594.3 (MetricCompletionComparison universal route)**.** *Every constructive comparison between the two dense completions named by an accepted* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetricCompletionDenseEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionDensitySource}^{\uparrow},\ \mathsf{RegularCauchySequenceSpace}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

The exported result is the displayed pair of comparison maps with finite round-trip evidence on the common dense source. It is not a quotient-completion equality proof, a selected representative, a selected limiting operator, a countable-choice principle, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the obligation surface of Definition 3553.2. A completion-comparison consumer must first read the paired dense-embedding rows E_0, E_1 and the common dense-source row D.

The finite observation route is then fixed by S, R, W, Q, A. The regular Cauchy sequence-space row supplies the shared source discipline, the regular rational and stream-window rows supply the inspected finite windows, the dyadic row supplies the precision ledger, and the real seal is reached only after those rows are displayed.

Finally use the two universal-property reads stored in U. They produce comparison maps in both directions and check the round-trip evidence only on the displayed dense source. Transport and replay through H and C, with provenance and local naming through P, N, preserve the same route and export no excluded host-side principle. □

Theorem 2594.4 (MetricCompletionComparison triangle route)**.** *Let two accepted* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *packets compare a first dense completion surface to a middle dense completion surface and the same middle surface to a third dense completion surface over one displayed dense metric source. Then every completion-facing read from the first surface to the third surface factors through the two comparison packets in sequence: the common* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *row, the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row, the relevant* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *rows, the shared* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and the two local* $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *rows.*

The composite route exports only the finite comparison read obtained by passing through the middle surface. It exports no quotient representative, selected limit, host equality of completions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 3553.2 to the first-to-middle packet and again to the middle-to-third packet. The two projections expose the same obligation surface: dense source, universal-property row, dense embeddings, regular Cauchy sequence-space interface, regular rational readback, stream windows, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming.

Next apply Definition 3553.3 to each comparison packet. The first application gives the route from the first completion surface to the middle surface, while the second gives the route from the middle surface to the third surface. Both routes are constrained to read the displayed dense source and the same universal-property discipline before any completion-facing read is accepted.

Compose the displayed routes along the middle completion surface, using the shared dense-source and universal-property rows as the overlap. The componentwise hsame row transports the common finite rows across the overlap, and the Cont row replays the two comparison reads in sequence without adding a new observation coordinate.

The carrier of Definition 3553.1 has no field for quotient representatives, selected limits, host equality of completion classes, countable choice, or ambient real completeness. Therefore the composite route can export only the finite first-to-third comparison read assembled from the two displayed packets. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionComparison}^{\uparrow}$ consumer that reaches a comparison map or round-trip read while omitting the common dense-source row, either dense-embedding row, the regular Cauchy sequence-space row, the RegSeqRat readback, the StreamName windows, the DyadicRatCore tolerance ledger, the Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchy-completion-universal-property, cauchy-completion-density-source, metric-completion-dense-embedding, regular-cauchy-sequence-space, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective with $\mathsf{MetricCompletionComparison}^{\uparrow}$ because the packet

jointly carries paired dense embeddings, a common dense source, two universal-property comparison directions, shared finite sequence-space readback, tolerance accounting, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionComparison}^{\uparrow}$ packet is generated from a CauchyCompletionUniversalProperty row, a CauchyCompletionDensitySource row, two MetricCompletionDenseEmbedding rows, a RegularCauchySequenceSpace row, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3553.1, Definition 3553.2, Definition 3553.3, and Definition 3553.4 seed the finite comparison route for constructive uniqueness of dense metric completions over a common source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, selected limits, countable choice, trichotomy, host equality of completion classes, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired dense-embedding exposure, common dense-source exactness, two universal-property comparison directions, round-trip evidence on the displayed source, finite readback and tolerance accounting, Real-seal routing, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.MetricCompletionComparisonUp.

Chapter 2595

A Concrete Naming Certificate for $\mathsf{BaireTwoFunction}^{\uparrow}$

Following Baire and classical descriptive set theory, the $\mathsf{BaireTwoFunction}^{\uparrow}$ packet records the successor step from Baire-one real-valued maps to pointwise limits of Baire-one approximants as finite BEDC NameCert data. It sits over the $\mathsf{BaireOneFunction}^{\uparrow}$ surface of chapter 1604, the finite schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-value seal of chapter 38. The packet names only the displayed approximation ladder and its finite read route; it is not a host function-space quotient, a selected limiting function, a measurability theorem, or an ambient completeness assertion.

Definition 2595.1 (BaireTwoFunction carrier)**.** *A* $\mathsf{BaireTwoFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, A, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BaireOneFunction}^{\uparrow}$ *approximation-ladder row, listing the Baire-one packets whose pointwise observations are being inspected.*
2. *A is the finite ladder-admission row, naming the displayed source points, approximation indices, and precision requests visible to the successor packet.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite approximation windows opened for each displayed request.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows obtained from the selected Baire-one observations.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Baire-one ladder, displayed schedule, and regular rational readback have been replayed.*
6. *H records componentwise* hsame *transport for the Baire-one ladder, finite admission row, schedule windows, rational readback, real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a pointwise successor request through the Baire-one ladder, the stream schedule, the rational readback, the real seal, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{BaireTwoFunction}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^\uparrow}$ *row.*

The carrier has no coordinate for a host space of Baire-two functions, an arbitrary selected approximating ladder, quotient equality of real names, full Borel hierarchy closure, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireTwoFunctionUp.TasteGate.BaireTwoFunctionTasteGate single carrier alignment`

Theorem 2595.2 (BaireTwoFunction NameCert obligations)**.** *For every accepted* $\mathsf{BaireTwoFunction}^\uparrow$ *carrier* $T = (B, A, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Baire-one approximation ladder, ladder-admission row, schedule windows, rational readback, real seal, componentwise transport, displayed continuation, provenance, and local naming rows;*
2. *Baire-one ladder exposure, requiring every successor observation to pass first through a displayed* $\mathsf{BaireOneFunction}^\uparrow$ *row in* B*;*
3. *schedule exactness, requiring every approximation request to use only the finite* $\mathsf{StreamName}^\uparrow$ *windows named by* S *after the ladder-admission row* A *is displayed;*
4. *readback exactness, requiring the value read to pass through the* $\mathsf{RegSeqRat}^\uparrow$ *row* Q *before the* $\mathsf{Real}^\uparrow$ *seal* R *is available;*
5. *transport and replay stability, requiring* H *and* C *to preserve the same route* B, A, S, Q, R *componentwise;*
6. *ledger refusal, excluding host function-space quotients, selected limiting functions, arbitrary ladder choice, quotient equality of real names, full Borel hierarchy closure, and ambient* $\mathsf{Real}^\uparrow$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3554.1. The rows B, A, S, Q, R are the only analytic rows available to a consumer, and H, C, P, N are the only structural rows attached to them.

The successor step begins with the Baire-one ladder B. The admission row A fixes the displayed source points, approximation indices, and precision requests before any schedule window can be opened.

The schedule and readback rows then force the finite route. A request reads the $\mathsf{StreamName}^\uparrow$ windows in S, checks the selected observations through the $\mathsf{RegSeqRat}^\uparrow$ row Q, and reaches the $\mathsf{Real}^\uparrow$ seal R only after those rows have been replayed.

Transport, continuation, provenance, and local naming preserve exactly this route. Since no carrier coordinate stores a host function space, selected limiting function, arbitrary ladder choice, quotient real equality, full Borel hierarchy closure, or ambient real completeness, the listed rows exhaust the NameCert surface. □

Theorem 2595.3 (BaireTwoFunction Baire-one handoff)**.** *Every* $\mathsf{BaireOneFunction}^\uparrow$ *approximation consumed by an accepted* $\mathsf{BaireTwoFunction}^\uparrow$ *packet is read before the* $\mathsf{StreamName}^\uparrow$ *schedule,* $\mathsf{RegSeqRat}^\uparrow$ *readback, and* $\mathsf{Real}^\uparrow$ *seal. Thus every successor value exported by the packet factors through*

$$B, A, S, Q \longmapsto R.$$

The handoff exposes a finite pointwise successor route from Baire-one data to real seals and exports no host function-space quotient, selected limiting function, arbitrary ladder selector, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer of the value seal R. By carrier admission, R is downstream of the same packet that contains the Baire-one ladder B, the admission row A, the schedule S, and the readback row Q.

Replay the continuation row C. The consumer first reads a displayed $\mathsf{BaireOneFunction}^{\uparrow}$ approximation from B, then uses A to identify the pointwise request and the finite approximation window available to the packet.

The schedule row S selects only those displayed windows, and the regular rational row Q reads the selected observations before R is exposed. Componentwise transport by H preserves this order for every accepted carrier.

The provenance and local naming rows contain no host function-space quotient, selected limiting function, arbitrary ladder selector, quotient real equality, or ambient completeness coordinate. Hence the exported successor value is only the displayed finite Baire-one handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3554.1, Definition 3554.2, and Definition 3554.3 seed the finite Baire-two successor packet over Baire-one ladders, finite schedules, rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BaireTwoFunctionUp.TasteGate.BaireTwoFunctionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host function-space quotients, selected limiting functions, arbitrary ladder choice, quotient equality of real names, full Borel hierarchy closure, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-one ladder exposure, schedule exactness, rational readback exactness, real-seal nonescape, transport stability, provenance, and ledger refusal for BEDC.Derived.BaireTwoFunctionUp.

Constructive story: A $\mathsf{BaireTwoFunction}^{\uparrow}$ packet is generated from a BaireOneFunction approximation ladder, finite request admission, StreamName schedule windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2596

A Concrete Naming Certificate for $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$

The $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ packet realizes the Alexander-dual trigger and blocker algebra of subsection 9462.28.21 as finite BEDC NameCert data. It sits over the finite $\mathsf{Bool}^{\uparrow}$ tag discipline of chapter 32, the finite observation interface of chapter 1007, the DNA boundary-tile carrier of chapter 3393, the six-tile coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the finite coverage-trigger algebra of chapter 3478. The packet names the blocker-dual surface as finite rows; it is not a polynomial-ring oracle, stochastic biological model, quotient codon semantics, infinite trigger search, or hidden Alexander-duality principle.

Definition 2596.1 (AlexanderDualTriggerAlgebra carrier)**.** *An* $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, T, I, U, L, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-tile carrier row read through* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *and* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$.
2. *T is the trigger row recording the finite coordinate supports whose full-trigger behavior is visible.*
3. *I is the inactive-face row recording the complements that remain blocked by a missed coordinate.*
4. *U is the blocker-dual cover row, carrying the four displayed dual generator faces and their finite lcm-degree ledger.*
5. *L is the* $\mathsf{Bool}^{\uparrow}$ *and finite-observation ledger that tags present, absent, trigger, blocker, active, and inactive faces without importing host truth values.*
6. *R is the transport row from boundary tiles through* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *and* WindowCoverageTriggerAlgeb *to the displayed blocker-dual cover.*
7. *H records componentwise* hsame *transport for boundary tiles, trigger rows, inactive faces, blocker-dual cover rows, finite tags, algebra transport, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each dual-trigger read through B, T, I, U, L, R, H.*
9. *P is the* Pkg *provenance over* $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{FiniteObservationInterface}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, WindowCoverageTriggerA *the Alexander-dual trigger source vision,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, T, I, U, L, R, H, C, P, N$. It has no coordinate for host polynomial equality, ambient monomial ideals, stochastic evolutionary dynamics, quotient codon equality, infinite trigger search, or a hidden duality oracle.

Theorem 2596.2 (AlexanderDualTriggerAlgebra NameCert obligations)**.** *For every accepted* AlexanderDualTrigger *carrier $A = (B, T, I, U, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}}$ *surface consists exactly of boundary-tile admission, trigger-row exposure, inactive-face exposure, blocker-dual cover exactness, finite Bool and observation tagging, algebra transport, componentwise transport, displayed replay, provenance, and local naming.*

The blocker-dual cover row U is accepted only when its trigger and blocker faces are read through the displayed B, T, I, L, R route. Any consumer that attempts to read Alexander-dual data without those rows, or that replaces them by host polynomial equality, an ambient ideal, quotient codon semantics, stochastic dynamics, infinite search, or a hidden duality oracle, leaves the accepted packet surface.

Proof. Project the carrier of Definition 3555.1. The rows B, T, I, U, L, R expose the boundary-tile source, trigger supports, inactive faces, blocker-dual cover, finite tag ledger, and transport from the boundary-trigger algebra.

The cover row is finite. It records only the four displayed dual generator faces and their lcm-degree ledger, with $\mathsf{Bool}^{\uparrow}$ and finite-observation rows marking which faces are present, absent, active, inactive, triggering, or blocking.

Finally H, C, P, N preserve the same displayed route. Since no host polynomial equality, ambient ideal, stochastic model, quotient codon equality, infinite search, or hidden duality oracle is a carrier coordinate, the local NameCert obligations are exactly the listed finite rows. □

Theorem 2596.3 (AlexanderDualTriggerAlgebra blocker-duality nonescape)**.** *Every accepted* AlexanderDualTrigge *read factors through displayed trigger, blocker, inactive-face, dual-cover, transport, replay, provenance, and local* NameCert *rows:*

$$B, T, I, U, L, R, H, C, P, N.$$

In particular, a dual-trigger consumer cannot replace the blocker-duality row by the $\mathsf{Bool}^{\uparrow}$ *tag discipline alone, by finite observation alone, by boundary closure alone, by trigger-hypergraph reliability alone, by universal coverage-trigger algebra alone, or by any host polynomial-ring equality.*

Proof. Read the boundary source first. The row B supplies the boundary tiles and coverage rows, while T and I separate visible trigger supports from inactive faces.

Next read the blocker-dual data. The row U carries the dual-cover faces, and L records the finite Bool and observation tags needed to distinguish trigger, blocker, active, and inactive positions.

The algebra transport R then ties those rows to the boundary-trigger closure and coverage-trigger algebra siblings. The structural rows H, C, P, N preserve the same route, so no sibling alone can absorb the blocker-duality coordinate and no host polynomial equality can be exported. □

Falsifiable prediction: Every accepted $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ read factors through displayed trigger, blocker, inactive-face, dual-cover, transport, replay, provenance, and local NameCert rows; omitting any of those rows refutes the packet.

Independence witness: bool, finite-observation-budget-selector, dna-boundary-trigger-closure, trigger-hypergraph-reliability, universal-coverage-trigger-algebra: no listed sibling is bijective with $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ because the packet jointly carries trigger rows, blocker rows, inactive-face rows, dual-cover rows, finite tag ledgers, algebra transport, replay, provenance, and local

naming. A bijection preserving only finite tags, observation budgets, boundary closure, reliability, or coverage-trigger closure cannot preserve the blocker-duality row U together with the inactive-face row I.

The Lean follow-up `BEDC.Derived.AlexanderDualTriggerAlgebraUp.TasteGate` supplies explicit Nontrivial AlexanderDualTriggerAlgebra$^{\uparrow}$ and FieldFaithful AlexanderDualTriggerAlgebra$^{\uparrow}$ instances preserving $B, T, I, U, L, R, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$.

Constructive story: An AlexanderDualTriggerAlgebra$^{\uparrow}$ packet is generated from boundary-tile rows, trigger rows, inactive-face rows, blocker-dual cover rows, finite Bool and observation ledgers, algebra transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3555.1, Definition 3555.2, and Definition 3555.3 seed the finite Alexander-dual trigger and blocker NameCert packet.

Formal verification: FormalStatus(NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host polynomial equality, ambient monomial ideals, stochastic biological dynamics, quotient codon equality, infinite trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-row exposure, blocker-dual cover exactness, inactive-face accounting, finite tag and observation ledgers, algebra transport, Nontrivial AlexanderDualTriggerAlgebraUp, and FieldFaithful AlexanderDualTriggerAlgebraUp for BEDC.Derived.AlexanderDualTriggerAlgebraUp.

Chapter 2597

A Concrete Naming Certificate for $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$

Following Moore–Smith and Kelley, the $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ packet records the finite criterion by which a directed Cauchy net exposes a convergence-facing subnet route without adding a selected subnet object or a choice principle. It sits over the directed-index surface of chapter 1607, the Moore–Smith carrier of chapter 3403, the Cauchy-net surface of chapter 589, the convergence interface of chapter 1667, the regular-rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a NameCert boundary for the L10 completion route: it records directed tails, Cauchy windows, subnet witnesses, and Real-facing readback rows, not a quotient of nets or an ambient compactness theorem.

Definition 2597.1 (Moore–Smith Cauchy subnet criterion carrier)**.** *A* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, M, A, W, B, N, S, G, Y, R, H, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row naming finite vertices, preorder comparisons, and upper-window witnesses.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *row exposing directed net values and the tail ledger consumed by the criterion.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *row recording the common-window Cauchy evidence read on the same directed surface.*
4. *W is the cofinal subnet-window row, listing the finite tail requests and their displayed upper refinements.*
5. *B is the subnet bound row, recording how each requested late window in M is met by a later displayed witness in W.*
6. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *consumer row that reads the subnet route only after D, M, A, W, B have been displayed.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for finite observation windows along the accepted subnet route.*
8. *G is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the scheduled observations.*

9. *Y is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those stream windows and dyadic tolerances.*
10. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, G, Y are displayed.*
11. *H records componentwise* hsame *transport for all criterion rows.*
12. *C records displayed* Cont *replay through the same directed-tail and readback route.*
13. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
14. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a globally chosen subnet, quotient-net equality, countable choice, an ambient compactness theorem, or host-level real completeness.

Lean4 ✓ `BEDC.Derived.MooreSmithCauchySubnetCriterionUp.MooreSmithCauchySubnetCriterionTasteGate`
`single carrier alignment`

Theorem 2597.2 (Moore–Smith Cauchy subnet criterion NameCert obligations)**.** *For an accepted carrier $Q = (D, M, A, W, B, N, S, G, Y, R, H, C, P, L)$ of Definition 3556.1, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCr}}$ *surface consists exactly of carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-window exposure, cofinal subnet-window evidence, subnet-bound compatibility, NetConvergence handoff, StreamName scheduling, DyadicRatCore tolerance comparison, RegSeqRat readback, Real seal exposure, componentwise transport, displayed route replay, provenance, and local naming.*

Proof. Carrier admission projects the finite packet of Definition 3556.1. The directed evidence is stored in D, M, A, W, B, so the criterion first reads the index row, Moore–Smith net row, Cauchy-net row, cofinal subnet windows, and subnet bounds.

The convergence-facing handoff is then N. By chapter 1667, a convergence consumer reads only displayed directed-tail evidence before it reaches filter or Real-facing rows.

The Real-facing part is S, G, Y, R. The StreamName, DyadicRatCore, RegSeqRat, and Real chapters supply finite observation windows, dyadic tolerances, regular-rational readback, and the terminal seal, so the packet does not need a hidden completed limit.

Finally H, C, P, L transport, replay, name, and certify exactly those rows. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 2597.3 (Moore–Smith directed-tail handoff)**.** *Every* $\mathsf{NetConvergence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ \mathsf{MooreSmithNet}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow} \longmapsto W, B \longmapsto \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{R}$$

The route supplies directed tails, Cauchy common-window evidence, subnet-window refinements, and finite readback rows only through the displayed carrier. It does not export a selected subnet function, quotient net, countable-choice selector, ambient compactness statement, or host-level completeness witness.

Proof. Start with the directed rows D, M, A. The $\mathsf{DirectedSet}^{\uparrow}$ row names upper-window witnesses, the $\mathsf{MooreSmithNet}^{\uparrow}$ row names the net-value and tail ledger, and the $\mathsf{CauchyNet}^{\uparrow}$ row names common-window evidence on that same directed surface.

The subnet criterion is the displayed pair W, B. The row W records the finite cofinal windows, while B records that each requested late Moore–Smith window is met by a later displayed witness.

The handoff to $\mathsf{NetConvergence}^{\uparrow}$ reads only after these rows are present. The downstream Real-facing rows S, G, Y, R are then the finite StreamName, DyadicRatCore, RegSeqRat, and Real seal route.

Transport, replay, provenance, and naming through H, C, P, L preserve the same carrier. Thus every accepted consumer factors through the displayed directed-tail handoff, and the excluded subnet-selection, quotient, choice, compactness, and host-completeness routes have no export row. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ packet is generated from directed indices, a Moore–Smith net row, a CauchyNet row, cofinal subnet windows, subnet-bound compatibility, a NetConvergence handoff, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3556.1, Definition 3556.2, and Definition 3556.3 seed the finite directed-tail subnet criterion used by L10 completion and compactness-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subnets, quotient nets, countable choice, compactness, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-window exposure, cofinal-window compatibility, convergence handoff, Real-facing readback, transport preservation, and local naming for the displayed criterion packet.

Chapter 2598

A Concrete Naming Certificate for $\mathsf{ReassignmentSupportCompression}^{\uparrow}$

The $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ packet realizes the codon reassignment support compression branch of the DNA six-bit boundary tile analysis as finite NameCert data. The support row is read through the six-bit codon window of chapter 2234 and the boundary-tile coverage surface of chapter 3470; the compression packet therefore consumes those displayed coordinate and coverage rows before it names motif and prime-cover data. It packages the displayed 13-codon support, five-motif formula, seven prime implicants, and unique five-motif prime cover from Definition 9462.1274 through finite transport and replay rows, rather than treating the support as a host-side Boolean simplification oracle.

Falsifiable prediction: Every accepted $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ read factors through displayed codon support, motif-expansion, prime-implicant, minimal-cover, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, stopcodonzeckendorfsuffix, codon-transition-matrix, bitvector: none is bijective to $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ because this packet simultaneously binds the reassigned 13-codon support, five-motif expansion, seven prime implicants, unique minimum prime cover, transport, replay, provenance, and local naming rows. The six-bit window carrier supplies coordinates without the support theorem; the Zeckendorf suffix row names a stop-codon boundary; the transition matrix records table movement; and a raw bitvector lacks motif and cover obligations.

Definition 2598.1 (ReassignmentSupportCompression carrier)**.** *A* $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ *carrier is a finite packet*

$$K = (S, M, I, U, H, C, P, N)$$

with the following visible rows:

1. *S is the displayed reassigned support row containing exactly the* 13 *codons of Definition 9462.1275;*
2. *M is the five-motif expansion row recording* CUN, AGR, UNA, WWA, *and* UAR *as the support formula;*
3. *I is the prime-implicant row recording exactly the seven maximal exact subcubes of Definition 9462.1280;*
4. *U is the minimal-cover row recording the unique five-prime cover of Definition 9462.1281;*

5. *H stores componentwise* hsame *transport for the support, motif, prime-implicant, minimal-cover, provenance, and local naming rows;*
6. *C stores the displayed* Cont *replay route from codon support to motif expansion, from motif expansion to prime implicants, and from prime implicants to the minimum-cover row;*
7. *P is the* Pkg *provenance over the DNA six-bit boundary tile source rows,* BHist, hsame, Cont, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}^{\uparrow}}$ *row.*

The carrier contains no host Boolean minimizer, no unbounded codon-table oracle, no genetic-code semantics beyond the displayed reassignment support, and no quotient identifying support lists with motif certificates.

Theorem 2598.2 (ReassignmentSupportCompression NameCert obligations)**.** *Every accepted* ReassignmentSuppo *packet satisfies the following finite obligations:*

1. *carrier admission exposes the support, motif, prime-implicant, minimal-cover, transport, replay, provenance, and NameCert rows of Definition 3558.1;*
2. *support exactness identifies S with the displayed* 13*-codon reassigned support and rejects any added or omitted codon;*
3. *motif expansion identifies M with the five IUPAC motifs whose union is the same support;*
4. *prime-cover compression identifies I with the seven prime implicants and U with the unique five-prime cover;*
5. *nonescape prevents a consumer from replacing the displayed rows by a host minimizer, a raw bitvector equality, a transition-matrix equivalence, a stop-codon suffix rule, or an unbounded codon-table oracle.*

Lean4 ✓ `BEDC.Derived.ReassignmentSupportCompressionUp.ReassignmentSupportCompressionNameCertObligations`

Proof. Project the carrier rows of Definition 3558.1. The support row S is the only entry point for codon membership, and the continuation row C replays it first into the five-motif row M.

Next replay M through the finite prime-implicant enumeration row I. This preserves exactly the displayed maximal exact subcubes because every admitted implicant is checked against the already fixed support row.

Finally project the minimal-cover row U from I. The package, transport, replay, and local naming rows carry no additional semantic oracle, so a consumer cannot bypass the finite support, motif, prime-implicant, and cover rows. The Lean follow-up will provide ‘Nontrivial ReassignmentSupportCompressionUp‘ and ‘FieldFaithful ReassignmentSupportCompressionUp‘ in ‘lean4/BEDC/Derived/ReassignmentSupportCompressionUp/TasteGate.lean‘. □

Theorem 2598.3 (ReassignmentSupportCompression sibling lattice)**.** *Every accepted* ReassignmentSupportCompre *consumer must read the codon six-bit window of chapter 2234, the stop-codon Zeckendorf suffix surface of chapter 1374, the codon transition matrix of chapter 2245, and the bitvector carrier of chapter 346 before treating the five-motif row M and the prime-cover rows I, U as support-compression data.*

Thus the packet is read as a horizontal lattice over displayed codon coordinates, stop-boundary suffixes, transition rows, and finite bitvector surfaces. It cannot be consumed as an isolated Boolean minimization certificate, a raw support bitvector, a stop-codon suffix rule, or a transition-matrix equivalence.

Proof. Project the carrier of Definition 3558.1. The support row S is already tied to displayed codon membership, and the packet's provenance row names the DNA six-bit boundary tile source rows before the motif row is replayed.

Read the sibling surfaces in order. The codon six-bit window supplies the coordinate surface, the stop-codon Zeckendorf suffix supplies the stop-boundary comparison row, the codon transition matrix supplies table-movement visibility, and the bitvector carrier supplies the finite support encoding boundary.

Only after those sibling rows are visible can the consumer interpret M, I, U as support-compression rows. Since the carrier contains no host Boolean minimizer, raw bitvector equality, isolated suffix rule, or transition-matrix quotient, the five-motif and prime-cover rows remain downstream of the sibling lattice. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}$.

Constructive story: A $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ packet is generated from the reassigned codon support, five-motif expansion, seven-prime-implicant surface, unique minimum prime cover, componentwise transport, displayed replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3558.1 and Definition 3558.2 seed the finite support-compression route for the reassignment support branch.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a checked Lean certificate, genetic-code semantics outside the displayed support, host Boolean minimization, table-transition completeness, or biological causality.

Upgrade path: Obligation closure requires checked carrier admission, support exactness, motif expansion, prime-implicant enumeration, minimum-cover uniqueness, and nonescape from host minimizers.

Chapter 2599

A Concrete Naming Certificate for $\mathsf{TopologicalClosure}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{TopologicalClosure}^{\uparrow}$ packet records closure of a displayed subset through finite neighbourhood evidence. It sits after the metric closure certificate of chapter 3342, the metric source of chapter 57, the topology surface of chapter 88, the finite ball discipline of $\mathsf{MetricBall}^{\uparrow}$, and the terminal real-name seal of chapter 38. The packet names a topological closure interface for BEDC neighbourhood reads; it is not a host closure operator, not a powerset completion, not quotient equality, not open-cover compactness, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2599.1 (TopologicalClosure carrier)**.** *A* $\mathsf{TopologicalClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, x, U, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row naming the finite neighbourhood request surface.*
2. *S is the displayed subset row whose closure is being tested.*
3. *x is the candidate point row, readable only through the same neighbourhood requests.*
4. *U is the neighbourhood-hit ledger, pairing each displayed neighbourhood of x with a retained source witness from S.*
5. *B is the optional* $\mathsf{MetricBall}^{\uparrow}$ *refinement row when the topology is read through metric balls.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained neighbourhood and witness observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for metric radii and witness distances when a metric refinement is present.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the neighbourhood, ball, window, and readback rows have been displayed.*
9. *H records componentwise* hsame *transport for the topology, subset, candidate, neighbourhood ledger, metric-ball refinement, windows, readback, and Real seal rows.*
10. *C records displayed* Cont *replay from neighbourhood requests to source witnesses, through optional metric balls, finite windows, regular readback, and the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{TopologicalClosure}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose topology row, subset row, candidate point, neighbourhood-hit ledger, optional metric-ball refinement, finite windows, rational readbacks, Real seal, transport, replay, provenance, and local naming are all displayed. It has no coordinate for a host closure operator, arbitrary open-set quantification, powersets, quotient equality, countable choice, or ambient completion of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.TopologicalClosureUp.TopologicalClosureTasteGate single carrier alignment`

Theorem 2599.2 (TopologicalClosure NameCert obligations)**.** *For every accepted* $\mathsf{TopologicalClosure}^{\uparrow}$ *carrier* $K = (T, S, x, U, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient topology row, displayed subset row, candidate point row, neighbourhood-hit ledger, optional metric-ball refinement, StreamName windows, RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public closure read must pass through those rows before it can consume the candidate point as closure evidence.*

Proof. Project the accepted carrier to the displayed rows $T, S, x, U, B, W, R, E, H, C, P, N$. The topology, subset, and candidate rows give the only admissible source data for a closure read.

The ledger row U binds each neighbourhood request around x to a retained source witness from S. When the topology is read metrically, the row B refines that same request through finite $\mathsf{MetricBall}^{\uparrow}$ data before the window row W and readback row R are exposed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same displayed packet. Since no row stores a host closure operator, a powerset, quotient equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the NameCert obligations are exhausted by the listed rows. □

Theorem 2599.3 (TopologicalClosure metric neighbourhood route)**.** *Every* $\mathsf{MetricClosure}^{\uparrow}$ *or metric-topology consumer of an accepted* $\mathsf{TopologicalClosure}^{\uparrow}$ *packet reads a candidate closure point through the finite route*

$$T, S, x, U, B, W, R, E, H, C, P, N.$$

The consumer receives a topology row, displayed subset row, candidate point, neighbourhood-hit ledger, metric-ball refinement, finite StreamName windows, RegSeqRat readbacks, RealUp seal, transport, replay, provenance, and local naming. It receives no ambient closure operator, arbitrary open-set family, quotient-set equality, selected convergent net, or RealUp completeness theorem.

Proof. Start with a metric-neighbourhood consumer. The carrier exposes the candidate point x only together with the common topology row T and source subset row S, so the consumer cannot detach the candidate from the displayed neighbourhood interface.

Use the ledger U to align each neighbourhood read with a source witness. The metric-ball row B, finite window row W, regular rational row R, and Real seal E then give the same finite evidence surface used by the $\mathsf{MetricClosure}^{\uparrow}$ packet of chapter 3342.

The rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Thus the handoff is the finite topological closure surface itself, not a stronger host closure, selected net, powerset, quotient, or completeness principle. □

Falsifiable prediction: Every $\mathsf{TopologicalClosure}^{\uparrow}$ public read factors through a Topology row, a displayed subset row, a candidate point row, a neighbourhood-hit ledger, optional MetricBall refinement, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: topology, metric, metric-closure, streamname, regseqrat, real: $\mathsf{TopologicalClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a neighbourhood-hit ledger, optional metric-ball refinement, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming for closure evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{TopologicalClosure}^{\uparrow}$ packet is generated from a Topology row, displayed subset row, candidate point, neighbourhood-hit ledger, optional MetricBall refinement, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3559.1, Definition 3559.2, and Definition 3559.3 seed the finite topological closure interface for displayed neighbourhood evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.TopologicalClosureUp.TopologicalClosureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host topological closure operators, arbitrary open-set families, selected convergent nets, quotient equality, $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood-hit stability, metric-ball refinement, StreamName and RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TopologicalClosureUp.

Chapter 2600

A Concrete Naming Certificate for $\mathsf{UniformSpaceCompletion}^{\uparrow}$

Following Bourbaki–Kelley uniform-space completion practice, the $\mathsf{UniformSpaceCompletion}^{\uparrow}$ packet records the finite route by which a uniform entourage source admits Cauchy-filter or Cauchy-net data, passes through dense-source and separated-reflection rows, and reaches a completion seal without turning the seal into quotient classes. It sits after the uniform entourage surface of chapter 339, the Cauchy filter surface of chapter 333, the generic completion packet of chapter 95, the metric comparison route of chapter 3553, the finite stream-window discipline of chapter 126, the regular rational source of chapter 301, and the terminal real-name seal of chapter 38. The packet names the L10 completion handoff as displayed NameCert data; it is not a quotient completion theorem, not a selected-limit operator, not countable choice, not trichotomy, not host equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2600.1 (UniformSpaceCompletion carrier)**.** *A* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, F, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform entourage source row, read from the* $\mathsf{UniformSpace}^{\uparrow}$ *interface before any Cauchy or completion demand is accepted.*
2. *F is the Cauchy filter or Cauchy net row, exposing exactly the finite entourage tests used by the completion consumer.*
3. *D is the dense-source embedding row, recording the source-to-completion handoff that lets original points enter the completion surface.*
4. *S is the separated reflection row, forcing indistinguishable Cauchy data to pass through a displayed separated interface before the completion seal is read.*
5. *E is the completion seal row, recording the accepted completion name after U, F, D, S have all been displayed.*
6. *H records componentwise* hsame *transport for the uniform source, Cauchy row, dense-source row, separated-reflection row, completion seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from Cauchy demands through the uniform source, dense-source handoff, separated reflection, and completion seal.*

8. *P is the* Pkg *provenance over* $\mathsf{UniformSpaceCompletion}^{\uparrow}$*, the seven sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, D, S, E, H, C, P, N$ and their displayed replay paths. It has no coordinate for quotient representatives, selected limits, countable choice, trichotomy, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformSpaceCompletionUp.UniformSpaceCompletionTasteGate single carrier alignment`

Theorem 2600.2 (UniformSpaceCompletion NameCert obligations)**.** *For every accepted* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *carrier $K = (U, F, D, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$ *surface consists exactly of five completion-facing obligations: carrier admission for the displayed finite packet; uniform-Cauchy exactness of the F row against the entourage source U; dense-source handoff through D; separated-reflection non-escape through S before the seal is consumed; and completion-seal stability through E, H, C, P, N. Every completion consumer must read those five rows in that order before it can export a completion name.*

Proof. Project the carrier of Definition 3560.1. The packet first displays the uniform entourage source U and the Cauchy row F, so the Cauchy demand is tested only against named finite entourages rather than against an ambient host uniformity.

The dense-source and separated-reflection rows then control the handoff. The row D records how original source data enters the completion surface, and S records that indistinguishable Cauchy data cannot escape the displayed separated interface before the completion seal is read.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Because the carrier has no row for quotient completion classes, selected limits, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be projected as a NameCert obligation. □

Falsifiable prediction: Every accepted $\mathsf{UniformSpaceCompletion}^{\uparrow}$ consumer that reaches a completion seal while omitting the uniform entourage source, Cauchy filter or net row, dense-source embedding row, separated-reflection row, completion-seal row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: uniformspace, cauchyfilter, completion, metric-completion-comparison, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{UniformSpaceCompletion}^{\uparrow}$ because the packet jointly carries a uniform entourage source, Cauchy filter or net exactness, dense-source handoff, separated-reflection non-escape, completion seal stability, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformSpaceCompletion}^{\uparrow}$ packet is generated from a uniform entourage source row, a Cauchy filter or Cauchy net row, a dense-source embedding row, a separated-reflection row, a completion seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3560.1 and Definition 3560.2 seed the finite L10 uniform completion route over uniform source, Cauchy exactness, dense-source handoff, separated reflection, seal stability, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion classes, selected limits, countable choice, trichotomy, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-Cauchy exactness, dense-source handoff, separated-reflection non-escape, completion-seal stability, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.UniformSpaceCompletionUp.

Chapter 2601

A Concrete Naming Certificate for $\mathsf{RegularCauchySubspace}^{\uparrow}$

Following Bishop and Bridges on subspaces of constructive metric spaces, the $\mathsf{RegularCauchySubspace}^{\uparrow}$ packet names regular Cauchy rational data whose accepted reads remain inside a displayed subspace predicate before any quotient equality, selected limit, or ambient completeness service is exposed. It sits on the regular rational readback of chapter 301, the finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the classifier discipline of chapter 1820. The packet is a finite NameCert surface for subspace-restricted regular Cauchy names; it is not a quotient real, not a selected limit, and not an assertion that the ambient space is complete.

Definition 2601.1 (RegularCauchySubspace carrier)**.** *A* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, S, D, A, K, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational readback inspected by the subspace consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that opens only the windows needed to test the displayed subspace route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bounds, window radii, and comparison rows consumed by Q and S.*
4. *A is the displayed subspace predicate row; it records the finite acceptance evidence that the inspected regular Cauchy name is read inside the named subspace.*
5. *K is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row, used only to compare accepted displayed names through their regular readback, tolerance ledger, and subspace predicate evidence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, S, D, A, K have all been displayed.*
7. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, subspace predicate, classifier, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a subspace-restricted regular Cauchy request through S, D, Q, A, K and then to the Real seal E.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubspace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, D, A, K, E, H, C, P, N$. The carrier has no coordinate for quotient real equality, a selected limit, an ambient completeness theorem, a host real equality predicate, countable choice, or an external subspace oracle.

Lean4 ✓ `BEDC.Derived.RegularCauchySubspaceUp.RegularCauchySubspaceTasteGate single carrier alignment`

Theorem 2601.2 (RegularCauchySubspace NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier*

$$U = (Q, S, D, A, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$ *surface consists exactly of carrier admission, regular rational readback, finite StreamName window exposure, DyadicRatCore tolerance exactness, displayed subspace-predicate evidence, CauchyEquivalenceSetoid classifier transport, Real-seal routing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3561.1. The accepted packet exposes the regular readback Q, finite windows S, dyadic ledger D, subspace predicate evidence A, classifier row K, terminal Real seal E, and structural rows H, C, P, N.

The finite observation route is forced before the subspace predicate can be consumed. The $\mathsf{StreamName}^{\uparrow}$ schedule supplies the inspected windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies the tolerance comparisons, and the $\mathsf{RegSeqRat}^{\uparrow}$ row packages those observations as the regular rational name to which A applies.

Classifier transport is then read only through K and H. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row compares accepted displayed names through the same regular readback, tolerance ledger, and subspace predicate evidence; componentwise hsame preserves those coordinates without adding a quotient real equality row.

Finally, the Real seal E is downstream of Q, S, D, A, K, while C, P, N only replay, source, and name the same route. Since the carrier has no coordinate for selected limits, ambient completeness, host real equality, countable choice, or an external subspace oracle, the local NameCert exports none of those objects. □

Theorem 2601.3 (RegularCauchySubspace inherited modulus route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier inherits its regular Cauchy modulus only through the finite route*

$$S, D, Q, A, K, E, H, C, P, N.$$

Thus the consumer reads StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, displayed subspace evidence, the CauchyEquivalenceSetoid classifier, and the terminal Real seal in that order. It receives no quotient equality for Cauchy names, no selected representative, no ambient completeness service, and no detached subspace-membership oracle.

Proof. Project the accepted carrier $U = (Q, S, D, A, K, E, H, C, P, N)$. The only windows and tolerance data available to a consumer are S and D, and the only regular Cauchy readback attached to them is Q.

The subspace restriction is not a separate oracle. It is the displayed predicate row A applied to the same regular readback route, so every accepted subspace-facing read passes through the same finite windows and dyadic tolerance ledger.

The classifier row K transports accepted names only after Q, S, D, A have been displayed. The terminal seal E is therefore a Real-facing endpoint of this packet, not a selected ambient limit.

The structural rows H, C, P, N preserve and name the route. Because no quotient equality, selected representative, ambient completeness service, or detached membership oracle is present among the displayed coordinates, none can be exported by composing the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3561.1, Definition 3561.2, and Definition 3561.3 seed the finite subspace-restricted regular Cauchy NameCert packet over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular readback, subspace-predicate stability, classifier transport, Real-seal routing, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchySubspaceUp.

Constructive story: A $\mathsf{RegularCauchySubspace}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, displayed subspace-predicate evidence, a $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2602

A Concrete Naming Certificate for $\mathsf{MetricReflectionSandwich}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{MetricReflectionSandwich}^{\uparrow}$ packet records the standard constructive route that reads a separated metric surface through a metric embedding, lands it in a completion surface, and checks that the dense embedding row preserves the same finite distance evidence. It sits on the metric carrier of chapter 57, the separation witness of chapter 1411, the embedding rows of chapter 3339, the completion surface of chapter 1506, the dense completion discipline of chapter 3351, and the terminal scalar seal of chapter 38. The packet is a finite NameCert consumer for the metric-reflection route; it is not a quotient completion, not a selected-representative theorem, not a choice principle, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2602.1 (MetricReflectionSandwich carrier)**.** *A* $\mathsf{MetricReflectionSandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, M, Q, E, C, D, R, H, T, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose distances are inspected.*
2. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row, exposing the finite zero-distance separation evidence for the same source.*
3. *Q is the* $\mathsf{MetricEmbedding}^{\uparrow}$ *row, read as a distance-preserving route out of X rather than as a host function equality.*
4. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the completion-facing target surface.*
5. *C is the* $\mathsf{DenseEmbedding}^{\uparrow}$ *row showing that the embedded source rows are dense in the displayed completion target.*
6. *D is the dyadic distance ledger used by the metric, embedding, completion, and density rows.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal reached only after X, M, Q, E, C, D have all been displayed.*
8. *H records componentwise* hsame *transport for metric distances, separation evidence, embedding rows, completion rows, dense embedding rows, dyadic comparisons, Real sealing, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from a source-distance read through the embedding and completion rows to the dense target read.*

10. *P is the* Pkg *provenance over* $\mathsf{MetricReflectionSandwich}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$ *row.*

The classifier accepts only packets whose separated metric, metric embedding, completion, dense embedding, dyadic distance ledger, Real seal, transport, replay, provenance, and local naming rows are displayed together. It has no coordinate for quotient completion, selected representatives, countable choice, host metric equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.MetricReflectionSandwichUp.MetricReflectionSandwichTasteGate single carrier alignment`

Theorem 2602.2 (MetricReflectionSandwich NameCert obligations)**.** *For every accepted* MetricReflectionSandwich *carrier $S = (X, M, Q, E, C, D, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$ *surface consists exactly of carrier admission, separated-metric exposure, metric-embedding distance preservation, completion-row routing, dense-embedding readback, dyadic distance-ledger exactness, Real distance sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3562.1. The accepted packet exposes the source metric, separation row, embedding row, completion target, dense embedding row, dyadic ledger, Real seal, and structural rows as finite coordinates.

The route through the sandwich is forced by those coordinates. A source-distance read starts at X, is separated by M, passes through the distance-preserving row Q, lands in the completion row E, and is reread through the dense embedding row C.

The remaining rows keep the route finite. The ledger D supplies the distance comparisons, R seals only those displayed comparisons as Real-facing data, and H, T, P, N transport, replay, source, and name the same packet. Since no row stores quotient completion, selected representatives, countable choice, host equality, or ambient Real completeness, the NameCert surface exports none of those objects. □

Theorem 2602.3 (MetricReflectionSandwich completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricReflectionSandwich}^{\uparrow}$ *carrier reads the source separated metric through the finite route*

$\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist,

The consumer receives a displayed metric source, separation evidence, distance-preserving embedding, completion target, dense embedding readback, dyadic distance ledger, Real distance seal, transport, replay, provenance, and local naming. It receives no quotient completion, selected representative, countable-choice schedule, detached host metric equality, or ambient Real-completeness theorem.

Proof. Begin with the carrier and apply Definition 3562.2. The completion-facing reader can inspect only the displayed rows of the packet.

The source metric is not read directly by the completion target. It first passes through the separated metric row and the metric embedding row, so the completion consumer sees the same finite distance data under the embedding route.

The dense embedding row then rereads the embedded source inside the completion surface. Dyadic comparisons and the Real seal are downstream of the displayed metric and embedding

rows, while transport, replay, provenance, and local naming preserve that order. Therefore the completion route is exactly the finite sandwich route, with no quotient completion, selected representative, choice, host equality, or ambient Real completeness exported. □

Falsifiable prediction: An accepted $\mathsf{MetricReflectionSandwich}^{\uparrow}$ completion consumer that reaches $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{DenseEmbedding}^{\uparrow}$ without first reading the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: metric, separatedmetric, metricembedding, metriccompletion, denseembedding, dyadicratcore, real: $\mathsf{MetricReflectionSandwich}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the separated source metric, embedding, completion target, dense readback, dyadic distance ledger, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3562.1, Definition 3562.2, and Definition 3562.3 seed the finite metric-reflection route between separated metric, metric embedding, completion, dense embedding, dyadic distance, Real seal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host metric equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-metric exposure, metric-embedding preservation, completion routing, dense-embedding readback, dyadic distance exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricReflectionSandwichUp.

Constructive story: A $\mathsf{MetricReflectionSandwich}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ source row, a $\mathsf{SeparatedMetric}^{\uparrow}$ row, a $\mathsf{MetricEmbedding}^{\uparrow}$ route, a $\mathsf{MetricCompletion}^{\uparrow}$ target, a $\mathsf{DenseEmbedding}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2603

A Concrete Naming Certificate for $\mathsf{ConvexConjugate}^{\uparrow}$

Following Rockafellar and Boyd–Vandenberghe, the $\mathsf{ConvexConjugate}^{\uparrow}$ packet names the Fenchel–Legendre conjugate surface by finite primal rows, dual rows, pairing evaluations, epigraph rows, and displayed supremum budgets. It sits over the convex-set, vector-space, real-name, and real-order interfaces of chapter 208, chapter 47, chapter 38, and section 38.3, and it supplies the conjugacy handoff consumed by chapter 321. The packet records conjugate data as BEDC carrier rows; it does not import host convex analysis, selected suprema, quotient function spaces, choice selectors, or ambient completeness.

Definition 2603.1 (Convex conjugate carrier)**.** *A* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, V, P, D, E, S, H, C, Q, N)$$

with the following displayed rows:

1. *F is the primal convex functional row, read through a displayed* $\mathsf{ConvexSet}^{\uparrow}$ *epigraph boundary.*
2. *V is the dual-vector row over the visible* $\mathsf{VecSpace}^{\uparrow}$ *dependency surface.*
3. *P is the pairing-evaluation row that records the finite bilinear read used by the conjugate comparison.*
4. *D is the domain and epigraph admission row tying the primal functional to the visible convex-set carrier.*
5. *E is the extended-value comparison row, read through* $\mathsf{Real}^{\uparrow}$ *and the real-order interface.*
6. *S is the supremum-budget row listing the finite inspected witnesses and the budget that prevents escape to a host supremum.*
7. *H stores componentwise* hsame *transport for the primal row, dual vector, pairing row, epigraph row, comparison row, supremum budget, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from primal and dual reads to pairing evaluations, from pairing evaluations to epigraph comparisons, and from those comparisons to the budgeted conjugate row.*
9. *Q is the* Pkg *provenance over* $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, P, D, E, S, H, C, Q, N$. It has no coordinate for a host supremum operator, quotient function space, choice-selected maximizer, ambient dual-space completion, or external conjugacy theorem.

Lean4 ✓ `BEDC.Derived.ConvexConjugateUp.ConvexConjugateTasteGate single carrier alignment`

Theorem 2603.2 (Convex conjugate NameCert obligations)**.** *For every accepted* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier* $K = (F, V, P, D, E, S, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, classifier stability under componentwise* hsame *transport, epigraph transport through the visible* $\mathsf{ConvexSet}^{\uparrow}$ *dependency, supremum-budget nonescape through* S*, and ledger visibility through the replay and provenance rows* C, Q, N*. Every conjugate read must pass through the primal functional row, dual-vector row, pairing evaluation row, epigraph row, extended-value comparison row, and displayed budget before a Fenchel-facing consumer can inspect it.*

Proof. Project the carrier of Definition 3563.1. Its visible coordinates are exactly $F, V, P, D, E, S, H, C, Q, N$, so carrier admission and classifier comparison are componentwise on those rows.

The epigraph and comparison clauses route every primal functional read through $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the real-order interface before it can be paired with the dual row. Transport by H and replay by C preserve that displayed route.

The row S is the only supremum-facing coordinate. Since the carrier has no host supremum, quotient function space, or selected maximizer field, the budgeted ledger S, C, Q, N is the entire local NameCert surface available to downstream consumers. □

Theorem 2603.3 (Convex conjugate Fenchel handoff)**.** *Every* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer that reads conjugacy through an accepted* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier receives only the displayed packet route*

$$F, V, P, D, E, S \longmapsto H, C, Q, N.$$

The consumer reads primal functionals, dual vectors, pairings, epigraphs, extended-value comparisons, and supremum budgets through the $\mathsf{ConvexConjugate}^{\uparrow}$ *packet, and receives no host supremum, quotient function space, choice selector, ambient completeness theorem, or external Fenchel conjugacy principle.*

Proof. Use Definition 3563.2. It places every conjugate-facing read on the carrier rows F, V, P, D, E, S before the structural rows H, C, Q, N can replay or source the answer.

Now pass to the Fenchel-facing consumer. The consumer may cite the $\mathsf{FenchelDuality}^{\uparrow}$ packet only after the displayed conjugate rows have been read, so conjugacy enters as a finite BEDC row rather than as an analytic operation in the host language.

The excluded host objects are absent from Definition 3563.1. Therefore the handoff supplies exactly the packet route and no selected supremum, quotient-function value, choice witness, completeness theorem, or external duality theorem. □

Falsifiable prediction: Every $\mathsf{ConvexConjugate}^{\uparrow}$ consumer reads conjugacy through a primal functional row, dual-vector row, pairing-evaluation row, epigraph row, extended-value comparison row, supremum-budget row, transport, continuation replay, provenance, and local naming.

Independence witness: convexset, vecspace, real, realalgorder, fenchelduality: $\mathsf{ConvexConjugate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries primal functional data, dual-vector data, pairing evaluation, epigraph comparison, a displayed supremum budget, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvexConjugate}^{\uparrow}$ packet is generated from a primal convex functional row, a dual-vector row, a pairing-evaluation row, epigraph and extended-value comparison rows, a displayed supremum budget, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3563.1, Definition 3563.2, and Definition 3563.3 bind the finite Fenchel–Legendre conjugate certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConvexConjugateUp.ConvexConjugateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host convex analysis, selected suprema, quotient function spaces, choice selectors, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, supremum-budget nonescape, Fenchel handoff, and TasteGate field faithfulness for BEDC.Derived.ConvexConjugateUp.

$\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ is routed through chapter 38, chapter 301, and the apartness sibling surface before the L10 completion-facing dependency rows are consumed.

Chapter 2604

A Concrete Naming Certificate for LocatedApartnessRealBridge$^{\uparrow}$

Following Bishop and Bridges, the LocatedApartnessRealBridge$^{\uparrow}$ packet records the finite comparison route by which located-radius readback and positive apartness evidence can be consumed together without promoting the result to classical real order. It sits between the located-real surface of **??**, the Cauchy real apartness surface of chapter 1881, the real apartness order bridge of chapter 1905, and the cotransitivity route of chapter 1788. The packet is a constructive comparison bridge for Real and Completion consumers; it is not a trichotomy theorem, quotient-real equality principle, host real equality test, selected-limit operator, or completeness claim.

Definition 2604.1 (Located apartness real bridge carrier)**.** *A* LocatedApartnessRealBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$$

with the following displayed rows:

1. *L_0 and L_1 are accepted* LocatedReal$^{\uparrow}$ *endpoint packets whose finite located-radius windows are being compared.*
2. *R_0 and R_1 are the located-radius readback rows, each exposing a* StreamName$^{\uparrow}$ *window, a* RegSeqRat$^{\uparrow}$ *readback, and a* DyadicRatCore$^{\uparrow}$ *radius ledger before any terminal* Real$^{\uparrow}$ *seal is inspected.*
3. *A is the positive apartness witness row, read through the* CauchyRealApartness$^{\uparrow}$ *surface and tied to the same endpoint windows.*
4. *O is the order-facing bridge row supplied by* RealApartnessOrderBridge$^{\uparrow}$*, used only after A has exposed positive apartness.*
5. *T is the cotransitivity route row, pointing to the existing* ApartnessCotransitivity$^{\uparrow}$ *surface when a third displayed real name is inspected.*
6. *H records componentwise* hsame *transport for the endpoint packets, readbacks, apartness witness, order bridge, cotransitivity route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located-apartness bridge request through $L_0, L_1, R_0, R_1, A, O, T, H, P, N$.*
8. *P is the* Pkg *provenance over* LocatedApartnessRealBridge$^{\uparrow}$, LocatedReal$^{\uparrow}$, CauchyRealApartness$^{\uparrow}$, RealApartnessOrderBridge$^{\uparrow}$, ApartnessCotransitivity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$ *row.*
The classifier admits exactly the finite packet in which located-radius readback and apartness evidence share the displayed endpoint windows. It has no coordinate for real trichotomy, quotient equality, host real equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Theorem 2604.2 (Located real apartness cotransitive bridge)**.** *For every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N),$$

any comparison read that uses the located-radius readbacks R_0, R_1 together with the positive apartness witness A reaches a Real-facing consumer only through the existing apartness-cotransitivity surface. More precisely, the read first opens the two $\mathsf{LocatedReal}^{\uparrow}$ *endpoint packets, then the shared finite readback route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

then the $\mathsf{CauchyRealApartness}^{\uparrow}$ *witness, then the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *order-facing handoff, and finally the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row T when a third real name is present. The export is finite constructive comparison evidence only. It exports no trichotomy, quotient real equality, host real equality, selected limit, countable-choice branch selection, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or completion principle.*

Proof. Project the carrier of Definition 3564.1. The endpoint information is exactly the two located real packets L_0, L_1 and the two finite located-radius readbacks R_0, R_1.

The apartness row is opened only after those finite readbacks are visible. Thus A is tied to the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ route as the endpoint data.

The order-facing handoff is downstream of apartness. The row O may expose the order bridge supplied by $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, but it receives positive separation from A and cannot introduce an equality test or a trichotomy branch.

When a third real name is inspected, the route must pass through T. That row is precisely the cotransitivity surface of $\mathsf{ApartnessCotransitivity}^{\uparrow}$, so the consumer receives a displayed cotransitive apartness route rather than a classical comparison principle.

The remaining rows H, C, P, N transport, replay, package, and name the same finite route. Since the carrier lists no quotient equality, host equality, selected limit, choice, or completeness row, none can be exported by the local certificate. □

Theorem 2604.3 (Located apartness real bridge NameCert obligations)**.** *For every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *endpoint admission for the two* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint packets L_0, L_1;*
2. *located-radius readback for R_0, R_1, each displaying the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *route attached to its endpoint;*
3. *positive apartness witness admission for the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A;*

4. *order-facing handoff for the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *row* O*, read only after* A *has exposed positive apartness;*
5. *cotransitivity routing for the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row* T *when a third displayed real name is inspected;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*.*

These rows export only the displayed constructive comparison route. They do not assert real trichotomy, quotient real equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the carrier of Definition 3564.1. The endpoint and readback obligations are exactly the rows L_0, L_1, R_0, R_1, so the local certificate admits only displayed located endpoints and finite radius-readback data.

The apartness and order rows are A and O. The carrier orders them so that the $\mathsf{CauchyRealApartness}^{\uparrow}$ witness is exposed before the $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ handoff can be consumed.

Apply Definition 3564.2. It reads the same endpoint, readback, apartness, and order rows as the consumer-facing route, and it sends any third-name comparison through the cotransitivity row T.

The remaining rows H, C, P, N transport, replay, package, and name that route. Since the carrier has no coordinate for trichotomy, quotient equality, host equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 2604.4 (Located apartness real two-way separation)**.** *An accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier exposes* $\mathsf{Real}^{\uparrow}$ *separation reads only through the displayed located-apartness row* A *and its endpoint readbacks* L_0, L_1, R_0, R_1*. Conversely, it exposes located-apartness reads only through the terminal* $\mathsf{Real}^{\uparrow}$ *seal route carried by* R_0, R_1 *together with the bridge transport, replay, provenance, and naming rows* H, C, P, N*.*

Proof. Project the carrier $J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$ from Definition 3564.1. The Real-facing data are the two located endpoints and their radius-readback rows, while the positive separation data are confined to the apartness witness row A.

Apply the NameCert obligations of Definition 3564.3. They require endpoint admission, located-radius readback, positive apartness admission, order-facing handoff, cotransitivity routing, and structural replay to be read as the same finite packet.

In the Real-facing direction, a separation read reaches the endpoint seal only after the displayed apartness row has tied the endpoints to positive evidence. In the apartness-facing direction, a consumer of A receives only the endpoint $\mathsf{Real}^{\uparrow}$ seal route plus H, C, P, N; the carrier exports no trichotomy, quotient equality, host equality, selected limit, countable-choice branch, or ambient completeness coordinate. □

Theorem 2604.5 (Located apartness real nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *read that exports an order-facing handoff still factors through the positive apartness row* A *and the cotransitivity route* T*. Consequently the handoff cannot export a three-way comparison branch, quotient-real equality, host real equality, or a detached order comparison outside*

$$L_0, L_1, R_0, R_1, A, O, T, H, C, P, N.$$

Proof. Project the carrier $J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$ from Definition 3564.1. The order-facing row is O, and it is listed downstream of the positive apartness witness A rather than as an independent trichotomy or equality coordinate.

Apply Definition 3564.2. It forces any third-name comparison to pass through the cotransitivity row T after the endpoint readbacks and the positive apartness row have been displayed.

Apply Definition 3564.3 and Definition 3564.4. These obligations keep O downstream of A and keep all Real-facing separation reads attached to the endpoint readbacks R_0, R_1.

The remaining rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate records a three-way comparison branch, quotient-real equality, host real equality, or detached order comparison, none can be exported by an accepted read. □

Theorem 2604.6 (Located apartness real bridge completion forward route)**.** *Any L10 completion consumer, in particular the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *consumer of chapter 1639, that reads located apartness from an accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$$

must pass through the located-radius readbacks R_0, R_1*, the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *surface* A*, the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *compatibility row* O*, and the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row* T *before it consumes any completion-facing comparison. The route exports constructive comparison evidence only; it exports no trichotomy, quotient-real equality, selected limit, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or completion principle outside the displayed rows.*

Proof. Project the endpoint and readback rows from Definition 3564.1. The completion-facing consumer first sees the two located endpoints and their finite radius readbacks, so the comparison begins with visible $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ data.

Apply Definition 3564.2. It opens the Cauchy real apartness witness after the readbacks, then routes the positive evidence through the order-facing bridge and the cotransitivity surface whenever a third name is inspected.

Use Definition 3564.3 and Definition 3564.4. These keep located-apartness evidence and Real-facing separation tied to the same finite packet before any completion-facing comparison is consumed.

Since the carrier has no coordinate for trichotomy, quotient equality, selected limits, host equality, choice, completeness, or an independent completion principle, the forward route is exactly the displayed located-radius, apartness, order-bridge, and cotransitivity handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3564.1 and Definition 3564.2 seed the finite constructive comparison route over located-real endpoints, located-radius readbacks, Cauchy apartness, the real-apartness order bridge, and apartness cotransitivity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, quotient real equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-radius readback stability, Cauchy apartness handoff, order-bridge compatibility, cotransitivity routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedApartnessRealBridgeUp.

Constructive story: A $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ packet is generated from two located-real endpoints, two located-radius readbacks, a positive Cauchy real apartness witness, a real-apartness order bridge row, an apartness-cotransitivity route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lemma 2604.7 (Located apartness real bridge sibling dependency route)**.** *Every accepted* LocatedApartnessRealBri *read consumes the located endpoint rows of* **??**, *the positive Cauchy apartness evidence of chapter 1881, the order-facing handoff rows of chapter 1905, and the cotransitivity routing of chapter 1788 before the local bridge name is inspected.*

Proof. Project the carrier of Definition 3564.1. The endpoint rows L_0, L_1 are located-real packets, so their finite located-radius windows are read through the sibling $\mathsf{LocatedReal}^{\uparrow}$ surface.

The positive separation row A is then read through $\mathsf{CauchyRealApartness}^{\uparrow}$, and the order-facing row O is downstream of that apartness evidence. If a third displayed real name is present, the row T routes the comparison through $\mathsf{ApartnessCotransitivity}^{\uparrow}$.

The structural rows H, C, P, N transport, replay, package, and name this same route. Since the carrier has no additional dependency coordinate, every local bridge read factors through exactly the listed sibling chapters. □

Lemma 2604.8 (Located apartness real bridge forward lattice)**.** *Every downstream Real and Completion consumer of a* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *packet reads the local bridge only after the located-real, Cauchy-apartness, order-bridge, and cotransitivity sibling surfaces of Definition 3564.7 have been exposed. In particular, a handoff to the separated-completion route of chapter 1612 may consume the bridge as finite apartness-routing data, but not as quotient-real equality, trichotomy, countable choice, or a selected completion limit.*

Proof. Project the dependency route already recorded in Definition 3564.7. The local bridge has no independent downstream coordinate before the located endpoint rows, positive Cauchy apartness evidence, order-facing handoff rows, and cotransitivity routing have been read.

A separated-completion consumer may therefore receive only the finite bridge packet produced by those sibling surfaces. The consumer obtains a displayed Real and Completion handoff, while the excluded host principles remain outside the carrier. □

Lemma 2604.9 (Located apartness real bridge Cauchy setoid route)**.** *Every located-apartness read exported by an accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *packet to a completion or separated-setoid consumer must pass through* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier route before it can be used as apartness evidence. The exported evidence is the finite L10 readback chain through chapter 38, chapter 301, chapter 126, chapter 291, and chapter 1820; it is not quotient-real equality, a selected completion representative, host order data, or ambient completeness.*

Proof. Project the carrier of Definition 3564.1. The located-radius readbacks are R_0, R_1, and each exposes the stream-name window, regular-rational readback, dyadic-radius ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal before any apartness row is consumed.

Use Definition 3564.3. The positive apartness row A is tied to those same endpoint windows, so a completion or separated-setoid consumer cannot detach the apartness witness from the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Now route the completion-facing read through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier. The setoid classifier is the sibling surface that records equality-compatible Cauchy comparison before a completion consumer may reuse the apartness evidence.

The structural rows H, C, P, N only transport, replay, source, and name this route. Since no row stores quotient-real equality, selected completion representatives, host order data, or ambient completeness, the L10 chain is nonescaping. □

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ hub reads its bounded candidate row through chapter 719 and the MetaCIC normalization frontier, so the socket is a visible sibling dependency rather than an isolated kernel float.

Chapter 2605

A Concrete Naming Certificate for $\mathsf{NormalizationCandidateSocket}^{\uparrow}$

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet realizes the candidate-mediated normalization evidence of chapter 9383 as a finite socket object. It sits beside the critical-path packet of chapter 765, the normalization frontier of chapter 892, the confluence audit packet of chapter 735, and the normal-form consistency seal of chapter 740. The packet separates candidate rows from closed-candidate rows and candidate-set rows, records the Pi and App adequacy surfaces, and keeps no-infinite-reduction exports downstream of supplied candidate evidence. It is not an unconditional closed-term strong-normalization theorem.

The socket consumes the MetaCIC candidate-normalization handoff through the candidate-set surface of chapter 2320 and the closed candidate route of chapter 5351. Its subject-reduction boundary is the four-row discharge socket of chapter 719; the local packet may hand candidate-mediated normalization data toward that boundary, but it does not replace the beta, application, lambda, or Pi preservation rows.

Definition 2605.1 (Normalization candidate socket carrier)**.** *A* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the closed typed MetaCIC term window, exposing the empty-context term, type, and closedness checks available to the audit map.*
2. *K is the reducibility candidate row, carrying the candidate predicate and its strong-normalization consequence.*
3. *D is the closed-candidate row, carrying closedness together with candidate evidence.*
4. *E is the candidate-set membership row, carrying membership, soundness, and beta-closure data.*
5. *A is the application adequacy row, recording the function-side Pi evidence, argument candidate evidence, and application candidate handoff.*
6. *B is the Pi adequacy row, recording Pi-candidate and neutral-normal evidence consumed by typed application routes.*
7. *I is the no-infinite-reduction row, downstream of the candidate-mediated strong-normalization projection.*

8. *F is the frontier row, retaining the refusal of any read that asks for closed-term strong normalization after deleting K, D, or E.*
9. *H records componentwise* hsame *transport for the term window, candidate row, closed-candidate row, candidate-set row, App adequacy row, Pi adequacy row, no-infinite-reduction row, frontier row, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a normalization-candidate socket request through $M, K, D, E, A, B, I, F, H,$*
11. *P is the* Pkg *provenance over* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$*, chapter 9383, chapter 765, chapter 892, chapter 735, chapter 740,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}$ *row.*

The classifier compares accepted sockets only through $M, K, D, E, A, B, I, F, H, C, P, N$. It has no coordinate for global closed-term strong normalization, evidence-erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Theorem 2605.2 (Normalization candidate socket boundary)**.** *For every accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N),$$

the local socket exports closed typed strong-normalization and no-infinite-reduction evidence only through at least one of the candidate rows K, D, E. Application adequacy A and Pi adequacy B are admissible only as route data for that same candidate-mediated socket. Any consumer missing the candidate row, the closed-candidate row, or the candidate-set row reaches F and cannot export closed-term strong normalization from this packet.

Proof. Read the carrier of Definition 3566.1. The closed term window M is present, but the socket does not attach strong normalization directly to M.

The candidate rows are the gate. A read may use K as reducibility candidate evidence, D as closed-candidate evidence, or E as candidate-set membership evidence. The strong-normalization export is downstream of one of these rows.

The adequacy rows refine the same gate. The application row A and Pi row B supply route data for function, argument, Pi, and neutral cases, but neither row replaces candidate evidence.

The no-infinite-reduction row I is downstream of the candidate-mediated strong-normalization projection. If a consumer omits K, D, E or asks to erase them, the frontier row F records the refusal.

Transport, replay, provenance, and local naming preserve this row order through H, C, P, N. Because no global normalization, unrestricted subject-reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed consistency row occurs in the carrier, the socket cannot export one. □

Theorem 2605.3 (Normalization candidate socket candidate-erasure refusal)**.** *Let $S = (M, K, D, E, A, B, I, F, H,$ be an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier. Any consumer that keeps the closed typed term window M but deletes all candidate evidence rows K, D, E cannot export the no-infinite-reduction row I from this packet. Such a consumer must reach the frontier refusal row F. The App adequacy row A and Pi adequacy row B remain route data for candidate-mediated reads; they are not substitutes for reducibility candidate, closed-candidate, or candidate-set evidence.*

Proof. Project the carrier of Definition 3566.1. The closed typed term window M is a visible socket input, but the carrier does not attach no-infinite-reduction evidence directly to M.

Apply Definition 3566.2. That boundary places closed typed strong-normalization and no-infinite-reduction exports downstream of at least one of K, D, E.

Now delete all three candidate evidence rows. With K, D, E absent, the consumer has no displayed reducibility candidate, closed-candidate, or candidate-set membership route from which I can be replayed.

The adequacy rows do not close the gap. App adequacy A and Pi adequacy B organize function, argument, Pi, and neutral routes once candidate evidence is present, but neither row supplies candidate evidence on its own. Therefore the only available response is the frontier row F, and the packet refuses the evidence-erased no-infinite-reduction export. □

Theorem 2605.4 (Normalization candidate socket MetaCIC diamond route)**.** *Every diamond-residual consumer of an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

must first read the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, and the frontier row F before citing chapter 4061. The MetaCIC diamond residual budget may consume the socket only after those candidate-mediated rows are visible; it cannot read a closed typed term window M as a premise-free normalization endpoint.

Proof. Project the socket carrier of Definition 3566.1. The rows relevant to a diamond-residual consumer are displayed as K, D, E, A, B, I, F inside the same finite packet.

Use Definition 3566.2. It places closed typed strong-normalization and no-infinite-reduction exports downstream of at least one candidate evidence row among K, D, E.

Use Definition 3566.3. It shows that deleting K, D, E while retaining M forces the frontier refusal row F, so the diamond-residual consumer cannot treat the closed term window alone as normalization evidence.

The adequacy rows A and B remain route data for candidate-mediated App and Pi reads, and I remains downstream of the same candidate evidence. Therefore the sibling route to chapter 4061 is available only after K, D, E, A, B, I, F are visible in the socket. □

Theorem 2605.5 (Normalization candidate socket forward dependency)**.** *Every forward discharge consumer of an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

must expose the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, the frontier row F, and then the forward-socket ledger of chapter 3387 before exporting a downstream socket read. The forward ledger is therefore downstream of visible candidate evidence, not a replacement for reducibility, closed-candidate, candidate-set, App adequacy, Pi adequacy, or frontier rows.

Proof. Project the carrier of Definition 3566.1. The forward discharge consumer can inspect only the displayed rows $M, K, D, E, A, B, I, F, H, C, P, N$.

Use Definition 3566.2. It places any strong-normalization or no-infinite-reduction export downstream of the candidate rows K, D, E and keeps App and Pi adequacy in the rows A, B.

Use Definition 3566.3. If the consumer deletes K, D, E while keeping the closed term window M, the route reaches the frontier refusal F and cannot export I.

Only after K, D, E, A, B, I, F are visible may the consumer cite chapter 3387. The forward-socket ledger records the downstream socket read, while H, C, P, N transport, replay, source, and name the same candidate-mediated packet. □

The socket is a consumer-side sibling of chapter 2320, chapter 5351, and chapter 719. It may route candidate-mediated normalization toward residual or discharge consumers only while keeping the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, and the frontier row F visible.

Falsifiable prediction: Any consumer missing candidate, closed-candidate, or candidate-set rows in a $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet cannot export closed-term strong normalization from that packet.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacicconfluenceauditpacket, normalformconsistencyseal: $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, frontier, transport, replay, provenance, and local naming rows.

Lean TasteGate coverage for this packet must provide Nontrivial and FieldFaithful instances for the candidate row family, the closed-candidate row family, the candidate-set row family, and the socket classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3566.1 and Definition 3566.2 seed the finite candidate-mediated normalization socket over candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, and frontier rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global closed-term strong normalization, evidence-erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate gate exactness, App and Pi adequacy compatibility, no-infinite-reduction routing, frontier refusal, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.NormalizationCandidateSocketUp.

Constructive story: A $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet is generated from one closed typed MetaCIC term window, candidate evidence, closed-candidate evidence, candidate-set evidence, App adequacy, Pi adequacy, no-infinite-reduction, frontier refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 2605.6 (Normalization candidate socket fuel boundary)**.** *Every* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *consumer that asks for bounded normalization receives only the displayed finite socket rows: the candidate term window M, a fuel row U, a bounded trace row T, a closedness-preservation row Q, and the structural rows H, C, P, N that transport and name that finite route. The trace row records that the fuel-indexed computation reaches the displayed bounded result within U steps and that each recorded step preserves the closedness row Q. The socket does not export a strong-normalization certificate, a no-infinite-reduction theorem, an erased candidate projection, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency from this bounded trace.*

Proof. Project the socket carrier of Definition 3566.1. The closed typed term window is available as M, while candidate, closed-candidate, and candidate-set rows remain the only rows that can support any strong-normalization export.

For bounded normalization, insert the fuel row U and bounded trace row T between the term window and the replay rows. The trace is finite because each step is indexed by the displayed fuel ledger rather than by an unbounded reduction argument.

Closedness preservation is carried separately by Q. Each trace edge may be read only together with the source and target closedness rows named by Q, so the consumer receives a closed finite computation rather than a global normalization theorem.

Finally, H, C, P, N transport, replay, package, and name exactly these displayed rows. Since no strong-normalization certificate is a coordinate of the bounded route, the socket boundary refuses any consumer that tries to read strong normalization from fuel alone. □

The bounded trace row U, T, Q is the local socket-side consumer of the MetaCIC normalization fuel boundary in chapter 5936. The socket fuel boundary reuses the MetaCIC fuel ledger as bounded trace evidence while refusing to turn fuel alone into strong normalization or no-infinite-reduction.

Theorem 2605.7 (Normalization candidate socket fuel-erasure refusal)**.** *Every* NormalizationCandidateSocket$^{\uparrow}$ *no-infinite-reduction read or closed strong-normalization read that crosses the socket fuel boundary must preserve the candidate rows K, D, E and the candidate-set fuel route before reading the no-infinite-reduction row I or frontier refusal row F. Erasing K, D, E from such a read turns the socket into an unconditional normalization endpoint over the closed typed term window M and fuel row U, but the carrier has no coordinate for that endpoint. The erased route is therefore refused at F rather than exported as evidence-free normalization.*

Proof. Project the carrier rows and use the fuel boundary of Definition 3566.6. The fuel row U and trace row T expose only bounded computation data, while the candidate rows K, D, E remain the visible evidence rows for strong-normalization and no-infinite-reduction reads.

Now apply the candidate-set fuel route of Definition 3566.8. A read that reaches candidate-set evidence must pass through E, the App and Pi adequacy rows, the bounded trace, the closedness-preservation row, and the structural rows before any public normalization-facing consequence is named.

Use the fuel-boundary candidate gate, Definition 3566.9. If K, D, E are erased, the fuel boundary cannot be lowered into a strong-normalization or no-infinite-reduction export; the only admissible output is the frontier refusal row F.

Finally specialize the candidate-set readback theorem, Definition 3566.10. It places the no-infinite-reduction row I downstream of M, E, A, B and the structural route. Thus reading I or F after deleting K, D, E would assert an endpoint that the carrier does not store, so the socket refuses the erased route. □

Theorem 2605.8 (Normalization candidate socket candidate-set fuel route)**.** *If an accepted* NormalizationCandidate *packet exposes the candidate-set row E and bounded fuel row U, then every bounded normalization read factors through candidate evidence, the closedness-preservation trace, App and Pi adequacy rows, the bounded trace, and the structural rows H, C, P, N. The route uses the candidate-set readback row E, the App adequacy row A, the Pi adequacy row B, the bounded trace row T, the closedness-preservation row Q, and the fuel row U; it does not export global strong normalization or completed closed consistency.*

Proof. Project the socket rows from Definition 3566.1. The candidate-set row E supplies the candidate evidence available to the bounded read, while the adequacy rows A and B keep application and Pi cases attached to that evidence.

Use Definition 3566.6. The bounded fuel row U and trace row T display the finite computation, and the closedness-preservation row Q accompanies each recorded step.

The transport and replay rows preserve the order of the read. Thus H, C, P, N can move and name only the route that has already passed through candidate evidence, adequacy rows, bounded fuel, bounded trace, and closedness preservation.

Since neither the candidate-set route nor the bounded-fuel boundary contains a coordinate for global strong normalization or completed closed consistency, those conclusions are not exported by the packet. □

Theorem 2605.9 (Normalization candidate socket fuel-boundary candidate gate)**.** *A fuel-indexed normalization read of an accepted* NormalizationCandidateSocket$^{\uparrow}$ *packet may lower or inspect the finite fuel boundary only while at least one of the candidate rows K, D, E remains visible. If K, D, E are erased, the read is forced to the frontier refusal row F and cannot export no-infinite-reduction, evidence-erased strong normalization, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.*

Proof. Project the carrier of Definition 3566.1. The strong-normalization and no-infinite-reduction exports are mediated by the candidate rows K, D, E, while the fuel row is only route data for bounded inspection.

Apply the boundary theorem Definition 3566.2. It places normalization-facing reads downstream of visible candidate evidence rather than downstream of the closed term window alone.

Use candidate-erasure refusal, Definition 3566.3. When K, D, E are removed, the packet has no candidate evidence to support a no-infinite-reduction read, so the frontier refusal row F is the only admissible boundary.

Thus fuel inspection can refine a displayed bounded trace but cannot replace reducibility-candidate evidence. The excluded strong-normalization, confluence, quotient, evaluator, and consistency conclusions require coordinates absent from the fuel boundary. □

Theorem 2605.10 (Normalization candidate socket candidate-set readback)**.** *If an accepted* NormalizationCandidat *packet exposes the candidate-set row E, then closed typed strong-normalization and no-infinite-reduction reads factor through the finite route*

$$M, E, A, B, I, H, C, P, N.$$

Bounded fuel rows may refine the displayed trace, but they cannot replace E as the candidate-set readback row. The packet therefore refuses global normalization, erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, and completed consistency.

Proof. Project the carrier of Definition 3566.1. The closed typed term window M is available to the socket, but Definition 3566.2 places strong-normalization and no-infinite-reduction exports downstream of candidate evidence.

Now specialize that boundary to the candidate-set row E. The application adequacy row A and Pi adequacy row B can refine the route through function and dependent-function cases, while I records the no-infinite-reduction read downstream of the candidate-mediated strong-normalization projection.

Use Definition 3566.6 for bounded fuel traces. A fuel row may display a finite computation and preserve closedness along that trace, but it supplies no candidate-set membership, no candidate soundness, and no beta-closure data. It therefore cannot replace E.

Finally H, C, P, N transport, replay, source, and name the candidate-set route. Since neither the base carrier nor the bounded-fuel extension contains rows for global normalization, erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed consistency, those reads remain refused. □

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ dependency route reads the candidate rows K, D, E, App and Pi adequacy rows A, B, and frontier refusal row F beside the MetaCIC candidate normalization witness of chapter 6577. The socket therefore records witness-facing dependency as a horizontal route through displayed candidate evidence, not as a theorem-free shortcut from a closed term window to global normalization.

Theorem 2605.11 (Normalization candidate socket MetaCIC candidate dependency route)**.** *The* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *packet consumes candidate-mediated strong-normalization evidence and normalization-frontier rows through chapter 892 and chapter 765. Its accepted route therefore reads candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, and frontier-refusal rows before any kernel-facing consumer may use the socket. The socket is not a kernel-only packet and cannot export strong normalization or confluence without the displayed MetaCIC candidate dependency edge.*

Proof. Project the carrier of Definition 3566.1. The candidate rows K, D, E, adequacy rows A, B, no-infinite-reduction row I, and frontier refusal row F are all visible in the same finite socket.

Use the boundary theorem Definition 3566.2 and the diamond route Definition 3566.4. They keep normalization-facing reads downstream of the candidate evidence rows and attach the socket to the MetaCIC critical-path consumer.

Finally read the normalization frontier and critical path chapters as the candidate-mediated parents. The accepted socket route therefore depends on those MetaCIC rows rather than floating as a closed-term or kernel-only surface. □

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet reads the subject-reduction discharge surface through chapter 765 and the normalization-frontier sibling of chapter 892. A candidate-normalization consumer must therefore expose the critical-path subject-reduction budget, the displayed normalization frontier rows, and the socket-local candidate evidence before treating the socket as a closed proof. The route is a forward dependency edge for MetaCIC normalization discharge, not an independent strong-normalization theorem or a replacement for the subject-reduction proof surface. The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ route is consumed by chapter 892 and chapter 6577 only after the candidate evidence rows, endpoint obstruction row, and socket rows remain visible in the same finite packet. The frontier read therefore passes through displayed candidate evidence rather than treating the socket as a theorem-free normalization endpoint.

Theorem 2605.12 (NormalizationCandidateSocket frontier cross-reference)**.** *Every* MetaCICNormalizationFrontier *consumer of a* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *packet must first pass through the candidate evidence rows, endpoint obstruction rows, socket rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any normalization endpoint is read. The route reaches the frontier consumer of chapter 892 and the candidate witness surface of chapter 6577 only after those rows remain visible in the same finite packet.*

Proof. Project the accepted socket packet. Its public coordinates already contain the candidate evidence rows, endpoint obstruction rows, socket rows, transport, replay, provenance, and local naming.

The frontier route then consumes those displayed rows through the sibling dependency named in chapter 892. The witness route of chapter 6577 is available only as part of the same displayed candidate evidence surface.

Since no carrier coordinate stores an already-read endpoint or an external normalization result, any endpoint-facing read must traverse the listed rows before it reaches the frontier consumer. □

Theorem 2605.13 (NormalizationCandidateSocket frontier dependency)**.** *Every endpoint-facing* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *read must expose candidate evidence rows, frontier refusal rows,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*, and candidate-normalization witness rows before it exports a normalization endpoint. The conclusion is conditional on the displayed socket packet and claims no global strong normalization, subject reduction, confluence, evaluator equality, or kernel consistency.*

Proof. Start with the frontier cross-reference of Definition 3566.12. It already routes a frontier consumer through candidate evidence, endpoint obstruction, socket, transport, replay, provenance, and local naming rows before any normalization endpoint is read.

Use the socket boundary theorem from the parent chapter. The endpoint-facing surface of $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ is therefore not available until the finite packet has displayed the candidate evidence and refusal rows that keep the endpoint conditional.

The $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ row supplies the frontier side of that refusal, while the $\mathsf{MetaCICCriticalPath}^{\uparrow}$ row records the critical-path dependency that must remain visible to endpoint consumers. The candidate-normalization witness rows are read as part of the same finite packet, not as an external evaluator equality or kernel theorem.

Thus an endpoint-facing read can export only the conditional endpoint licensed by those displayed rows. It cannot turn the socket into global strong normalization, subject reduction, confluence, evaluator equality, or kernel consistency. □

Chapter 2606

A Concrete Naming Certificate for MetaCICSnCandidateInterface$^{\uparrow}$

The MetaCICSnCandidateInterface$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization frontier of chapter 9386 as finite MetaCIC NameCert data. It sits beside the normalization frontier of chapter 892, the closure trace of chapter 795, the normalization certificate router of chapter 1108, and the closed-term substitution boundary of chapter 527. The packet names candidate evidence, closed-candidate routing, bounded-normal endpoint data, and subject-reduction frontier retention without claiming evidence-free closed-term normalization.

Definition 2606.1 (MetaCIC SN candidate interface carrier)**.** *A* MetaCICSnCandidateInterface$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, K, L, B, E, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term window in scope for the normalization read.*
2. *K is the candidate evidence row, allowing candidate, closed-candidate, or candidate-set evidence to be displayed explicitly.*
3. *L is the closed-candidate routing row that binds the evidence in K to the closed-term boundary consumed by downstream readers.*
4. *B is the bounded normalizer evidence row, recording the finite endpoint data available at the inspected fuel or budget.*
5. *E is the endpoint readback row that exposes the bounded normal form only after T, K, L, B have been displayed.*
6. *F is the subject-reduction frontier retention row, keeping the required subject-reduction discharge boundary visible instead of erasing it into an unconditional theorem.*
7. *H stores componentwise* hsame *transport for closed typed terms, candidate evidence, closed-candidate routing, bounded endpoint data, frontier retention, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for candidate-mediated normalization requests.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICClosureTrace}^{\uparrow}$, $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$ *row.*

The classifier accepts only packets whose strong-normalization-facing read traverses $T, K, L, B, E, F, H, C, P, N$. It has no coordinate for unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Lean4 ✓ `BEDC.Derived.MetaCICSnCandidateInterfaceUp.MetaCICSnCandidateInterfaceTasteGate single carrier alignment`

Theorem 2606.2 (MetaCIC SN candidate interface NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ *carrier $S = (T, K, L, B, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$ *obligations are exactly carrier admission, closed typed term scoping, candidate evidence admission, closed-candidate routing, bounded normalizer endpoint readback, subject-reduction frontier retention, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3567.1. Carrier admission exposes exactly the closed term window, candidate evidence, closed-candidate route, bounded endpoint data, frontier retention, and structural rows.

Read the candidate side first. The rows T, K, L ensure that a closed typed term is paired with explicit candidate evidence and that the evidence is routed through the closed-term boundary before any endpoint readback is consumed.

The bounded endpoint row is then inspected through B and E. This read records only finite normalizer evidence at the displayed budget, not a global normalization theorem.

Finally, F, H, C, P, N keep the subject-reduction frontier, transport, replay, provenance, and local name visible. Since the carrier lacks evidence-free normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, and completed consistency rows, the listed obligations exhaust the packet. □

Theorem 2606.3 (MetaCIC SN candidate interface frontier refusal)**.** *Any closed-term strong-normalization consumer of an accepted* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ *packet is refused unless the route exposes candidate evidence, bounded normalizer evidence, subject-reduction frontier retention, transport, replay, provenance, and local naming rows. The packet cannot be read as arbitrary closed-term normalization or as a completed MetaCIC consistency theorem.*

Proof. Start from the closed typed term window T. The carrier makes T visible only together with the candidate evidence row K and closed-candidate routing row L.

Inspect the endpoint data. Rows B and E provide bounded normalizer evidence and endpoint readback at the displayed budget, so a consumer cannot skip directly from a closed term to a strong-normalization conclusion.

Now read the frontier row F. It retains the subject-reduction boundary as part of the packet rather than discharging it as an unconditional theorem.

The structural rows H, C, P, N preserve and name the same candidate-mediated route. Therefore any consumer missing one of these rows is refused, and the packet cannot export evidence-free closed-term normalization or completed consistency. □

Falsifiable prediction: If a MetaCIC strong-normalization consumer exports a closed-term SN row without candidate evidence, bounded normalizer evidence, subject-reduction frontier retention, transport, replay, provenance, and local naming rows, then the packet is refuted.

Independence witness: metacicnormalizationfrontier, metacicclosuretrace, metacicnormalizationcertificaterouter, closedtermsubstitutionboundary: $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate evidence, closed-candidate routing, bounded-normal endpoint data, and subject-reduction frontier retention.

The Lean TasteGate site for $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ provides explicit Nontrivial MetaCICSnCandidateInter and FieldFaithful $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ instances in `TasteGate.lean`. These instances preserve candidate evidence, closed-candidate routing, bounded endpoint readback, subject-reduction frontier retention, transport, replay, provenance, and local naming as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ packet is generated from a closed typed MetaCIC term window, candidate evidence, closed-candidate routing, bounded normalizer endpoint data, subject-reduction frontier retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3567.1, Definition 3567.2, and Definition 3567.3 seed the finite candidate-mediated strong-normalization interface for MetaCIC readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetaCICSnCandidateInterfaceUp.MetaCICSnCandidateInterfaceTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close arbitrary closed-term normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, completed closed consistency, full MetaCIC strong normalization, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence routing, bounded endpoint readback, subject-reduction frontier retention, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSnCandidateInterfaceUp.

$\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ reads its finite family-modulus packet as a consumer of chapter 388, chapter 369, and chapter 413, while the routed child files below keep the compact-net, equicontinuity, and completion handoffs separate.

Chapter 2607

A Concrete Naming Certificate for $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet records the finite-modulus subsequence route for an equicontinuous family on a compact metric source. It consumes the uniform-continuity witness pattern of chapter 3444, the compact source surface of chapter 129, the displayed map rows of chapter 130, the window schedules of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the dyadic tolerance arithmetic of chapter 291. The chapter names a finite extraction certificate; it does not assert general open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Concretely, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ consumes the equicontinuity-family rows of chapter 4076 and the compact metric finite-net rows of chapter 129 before exporting a finite modulus packet.

It also consumes the compact finite-net and uniform-modulus route of chapter 7330 together with the equicontinuity obligation surface of chapter 4076 before exposing the Arzela–Ascoli finite-modulus carrier.

The same finite-modulus route is consumed by the downstream selector lattice of chapter 4076, chapter 388, and chapter 914. Before selector rows are exported, the route also consumes the finite ProbeBundle realization route of chapter 3957, which supplies compact metric coverage, continuous pointwise moduli, RatUp lower-bound folds, and UniformModulusUp handoff rows for $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$. It also reads the finite compactness certificate of chapter 3586, whose compact-metric, finite-net-equicontinuity, compact-uniform-continuity-modulus, and finite-epsilon-net rows are available before the selector row is exported. The downstream equicontinuity and compact-uniform-continuity chapters consume the shared finite-modulus row M only through this displayed lattice, so their dependency on $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ is a row-level handoff rather than a hub-level citation. Those siblings may read the packet only after the compact metric, continuous-map, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are displayed. In row form, the selector lattice receives exactly $K, F, M, W, R, E, H, C, P, N$: K supplies the compact finite-net source, F the inspected continuous-map family, M the shared finite modulus, W the windows, R the rational readback, E the terminal real seal, and H, C, P, N the transport, replay, provenance, and local naming rows.

The finite-modulus extraction route also consumes the compact-uniform-continuity surface of chapter 388 and the uniform modulus ledger of chapter 369 before exporting any Arzela–Ascoli selection row. Compact finite nets provide the domain-side finite cover, while $\mathsf{UniformModulus}^{\uparrow}$ supplies the image-side modulus ledger that keeps the extracted windows tied to one displayed

tolerance schedule.

Before any selector-facing row is exported, the same finite-modulus extraction also consumes compact oscillation and compact uniform modulus rows, for example through chapter 2299. The dependency is row-level: compact finite-net data, oscillation or modulus ledgers, $\mathsf{UniformModulus}^{\uparrow}$ handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows are exposed before selection data, without selected subsequence choice or host function-space compactness.

Definition 2607.1 (Arzela–Ascoli finite-modulus carrier)**.** *An* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, M, W, R, E, H, C, P, N)$$

with the following rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source certificate, read only through finite net and dyadic-radius data;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family surface, inspected by finite* $\mathsf{StreamName}^{\uparrow}$ *windows rather than by a host function-space object;*
3. *M is the finite common modulus ledger, routed through* $\mathsf{HeineCantor}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;*
4. *W is the finite net-window extraction surface for the chosen inspected depths;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the extracted finite values;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed pointwise limit windows;*
7. *H records componentwise* hsame *transport of K, F, M, W, R, E;*
8. *C records displayed* Cont *replay of the same finite extraction route;*
9. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a choice-selected subsequence, function-space compactness, quotient equality of real names, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusCarrier`
Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusTasteGate single carrier alignment`

Theorem 2607.2 (Arzela–Ascoli finite-modulus NameCert obligations)**.** *For the carrier of Definition 3568.1, the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-metric source admission through K, continuous-map family admission through F, finite common modulus admission through M, finite net-window extraction through W, regular rational readback through R, pointwise* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusNamecertObligations`

Proof. Project the carrier. The compact source row K, continuous-map row F, and finite common modulus row M are visible before any limit seal is read.

Read the extraction rows next. The finite windows W are selected from displayed net data and inspected depths, while R reads those windows as regular rational schedules through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$.

Finally the seal row E and the structural rows H, C, P, N transport, replay, source, and name exactly that finite route. Since no other coordinate appears in the carrier, no open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness obligation is introduced. □

Theorem 2607.3 (Arzela–Ascoli equicontinuous finite-net extraction)**.** *Let $A = (K, F, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier. Every equicontinuous extraction row factors through the compact metric finite-net data in K, the common finite modulus in M, and the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows in W before the regular rational readback R is consumed. Thus the extraction is a dyadic finite-net construction over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and not a host compactness or function-space compactness principle.*

Proof. Use the compact source row K to obtain only the displayed finite-net and radius data. The row does not expose an arbitrary open-cover compactness theorem.

Apply the finite common modulus row M. It routes the inspected family values through the same Heine–Cantor-style modulus ledger at every selected depth.

The row W then records the finite net windows, and R reads them as $\mathsf{RegSeqRat}^{\uparrow}$ schedules. Transport and replay by H and C preserve the same dyadic route, so no selected subsequence or host function-space compactness object is available. □

Theorem 2607.4 (Arzela–Ascoli subsequence nonescape)**.** *Every subsequence or limit-facing consumer of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier must pass through*

$$K, F, M, W, R, E, H, C, P, N.$$

The consumer receives finite compact-source nets, continuous-map inspection rows, a shared modulus ledger, finite extraction windows, regular rational readback, and a $\mathsf{Real}^{\uparrow}$ *seal. It receives no selected subsequence choice, no host function-space compactness theorem, no quotient equality of real names, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with Definition 3568.3. It places the compact-source nets, common modulus, finite windows, and regular rational readback before the limit seal.

Then read the seal row E. It is downstream of K, F, M, W, R and therefore names only the displayed pointwise limit windows.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier inventory in Definition 3568.1 has no field for the excluded host principles, those principles cannot escape through the subsequence consumer. □

Theorem 2607.5 (Arzela–Ascoli finite-modulus forward lattice)**.** *Every downstream read of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet by chapter 4076, chapter 388, or chapter 914 factors through the displayed rows*

$$K, F, M, W, R, E, H, C, P, N.$$

The consuming selector receives compact metric finite-net data, continuous-map inspection data, a finite common modulus ledger, finite windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ *seal, and dyadic tolerance arithmetic before it can export any finite selector row. It receives no selected subsequence choice, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3568.1. The compact metric row K, continuous-map family row F, finite common modulus row M, and window row W are visible before any selector-facing consumer can read a limit row.

Apply Definition 3568.3. It places the finite-net and modulus route ahead of the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R and keeps the dyadic tolerance arithmetic in the displayed dependency surface.

Before the selector row is exported, read chapter 3586. Its compact-metric, finite-net-equicontinuity, compact-uniform-continuity-modulus, and finite-epsilon-net rows provide the finite compactness certificate that feeds the same selector route.

Finally apply Definition 3568.4. The real seal E and structural rows H, C, P, N preserve the same finite route for the equicontinuity, compact-uniform-continuity, and compact-net-modulus selector siblings. Since no excluded host principle is a carrier coordinate, no downstream selector can consume one through this packet. □

Theorem 2607.6 (Arzela–Ascoli Heine–Cantor compactness route)**.** *Every selector-facing finite-modulus read of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier*

$$A = (K, F, M, W, R, E, H, C, P, N)$$

first consumes the compact uniform modulus route supplied by $\mathsf{HeineCantor}^{\uparrow}$*, the finite-net source data of* $\mathsf{CompactMetric}^{\uparrow}$*, the pointwise rows of* $\mathsf{ContinuousMap}^{\uparrow}$*, the tolerance ledger of* $\mathsf{UniformModulus}^{\uparrow}$*, the finite windows of* $\mathsf{StreamName}^{\uparrow}$*, the regular readback of* $\mathsf{RegSeqRat}^{\uparrow}$*, the tolerance arithmetic of* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Only after these rows are visible may the selector export a finite-modulus handoff to chapter 4076, chapter 388, or chapter 914.*

Proof. Project the carrier. The displayed rows $K, F, M, W, R, E, H, C, P, N$ expose the compact source, inspected continuous-map family, finite modulus ledger, finite windows, rational readback, terminal real seal, transport, replay, provenance, and local naming.

Apply Definition 3568.5. It places the selector-facing read behind the compact metric, continuous-map, finite-modulus, window, readback, real-seal, and dyadic tolerance rows.

Now read Definition 3568.9. The compact-uniform finite-selector route forces the same consumer to pass through the $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ sibling rows before it reaches the compact-uniform-continuity surface.

Finally apply Definition 3568.4. Since the packet has no selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness coordinate, the Heine–Cantor compactness route remains a finite row-level dependency for every selector-facing export. □

Theorem 2607.7 (Arzela–Ascoli compact-open dependency)**.** *Every finite-modulus extraction read that compares a family of maps in an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first pass through the* $\mathsf{CompactOpenMetric}^{\uparrow}$ *finite source-net, graph-sampling, target-distance, and sup-distance rows of chapter 1903 before exporting the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector row. In the notation of Definition 3568.1, the dependency is read through*

$$K, F, M, W, R, E, H, C, P, N.$$

Proof. Project the accepted carrier. The compact source row K supplies the finite source-net surface, while the continuous-map family row F supplies the maps whose graph samples are compared.

Read the modulus and window rows M, W. They restrict the comparison to displayed finite source windows and keep the sampled values tied to one tolerance schedule.

Now apply the compact-open metric route of chapter 1903. Its target-distance and finite sup-distance rows are consumed before the regular rational readback R and terminal seal E are available to the selector.

Finally H, C, P, N transport, replay, source, and name the same route. Thus the Arzela–Ascoli finite-modulus selector receives a compact-open finite sup-distance dependency, not an unrecorded host function-space comparison. □

Theorem 2607.8 (Arzela–Ascoli finite modulus compactmetric link)**.** *Every selector-facing read through an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source,* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows,* $\mathsf{Equicontinuity}^{\uparrow}$ *family-modulus rows,* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, and* $\mathsf{Real}^{\uparrow}$ *seal before exporting selector data. In the notation of Definition 3568.1, this read is carried by the finite rows*

$$K, F, M, W, R, E, H, C, P, N.$$

Proof. Begin with the compactmetric sibling readback in Definition 3568.3. It places the compact metric finite-net source K and the common finite modulus M before the regular rational readback.

Apply Definition 3568.5. The selector-facing lattice then reads the continuous-map family rows, equicontinuity family-modulus rows, finite windows, dyadic tolerance arithmetic, and terminal $\mathsf{Real}^{\uparrow}$ seal in the same displayed route.

Use Definition 3568.7 for the compact-open comparison. It keeps graph-sampling and finite sup-distance data downstream of the compact metric and continuous-map rows.

Finally apply Definition 3568.4. The seal and structural rows E, H, C, P, N prevent selector data from escaping the finite row surface, so no selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness is exported. □

Theorem 2607.9 (Arzela–Ascoli compact-uniform finite-selector route)**.** *Any* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-selector consumer of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first read*

$$K, F, M, W, R, E, H, C, P, N$$

together with the compact metric, continuous map, Heine–Cantor, $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling rows. The route then lands in chapter 388 only through the finite selector surface also consumed by chapter 914.*

Proof. Project the carrier of Definition 3568.1. The rows $K, F, M, W, R, E, H, C, P, N$ expose the compact metric source, continuous-map family, finite common modulus, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming.

Apply Definition 3568.5. It already forces every downstream finite-selector read to pass through the compact metric, continuous-map, Heine–Cantor, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before any selector-facing handoff is available.

Therefore the compact-uniform-continuity consumer reaches chapter 388 by the same finite route that feeds chapter 914, with no selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle inserted. □

Falsifiable prediction: Every consumer of an $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet must read the compact metric finite nets, continuous-map family windows, finite common modulus, regular rational readback, and $\mathsf{Real}^{\uparrow}$ seal before consuming the extracted limit row.

Independence witness: compactmetric, continuousmap, heine-cantor, streamname, regseqrat, real, dyadicratcore: $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ is not bijective to any listed sibling because it

jointly carries compact-source nets, a continuous-map family surface, a finite common modulus ledger, finite extraction windows, regular rational readback, Real sealing, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet is generated from compact metric finite nets, continuous-map family windows, a finite common modulus ledger, finite extraction windows, regular rational readback, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3568.1, Definition 3568.2, Definition 3568.3, and Definition 3568.4 bind the finite-modulus extraction surface for one displayed equicontinuous family.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Scoped closure requires the finite-net extraction rows, Real seal row, and NameCert dependency surface to be formalized for BEDC.Derived.ArzelaAscoliFiniteModulusUp.

Lemma 2607.10 (Arzela–Ascoli compact-metric forward lattice)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net carrier of chapter 129 and the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *modulus route before a finite selector row can be read. The dependency remains the displayed finite lattice through compact metric nets, compact-uniform-continuity moduli, inspected stream windows, regular rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal; it does not expose open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2607.11 (Arzela–Ascoli compact metric forward route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows of chapter 129 and the* $\mathsf{Equicontinuity}^{\uparrow}$ *pointwise modulus rows of chapter 4076 before exporting its finite subfamily modulus route. In the carrier of chapter 3568, the compact metric source row is read before the equicontinuity family row, the shared finite modulus row, the finite window ledger, the regular rational readback, and the terminal real seal.*

Lean4 ✓ `BEDC.Derived.Arzela ascoli finite modulusUp.ArzelaAscoliFiniteModulusCompactmetricForwardRoute`

Proof. Project the accepted $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ carrier through the local NameCert chapter chapter 3568. The compact metric coordinate supplies only finite-net centers, radius rows, and source coverage windows.

Read the equicontinuity coordinate next. Its pointwise modulus rows are attached to the displayed family and to the compact metric source windows, so the shared finite modulus cannot be exported before those two dependency surfaces are visible.

The finite subfamily modulus route then reads the window ledger, regular rational readback, real seal, transport, replay, provenance, and local naming rows. Since neither dependency chapter exports a selected subsequence or host function-space compactness row, the Arzela–Ascoli packet remains a finite forward route from $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{Equicontinuity}^{\uparrow}$ into the local selector surface. □

Theorem 2607.12 (Arzela–Ascoli compact metric forward reference)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *extraction first reads the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row* K *of chapter 129 before any selector output, finite window, regular rational readback, or terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route then passes through the continuous-map row* F*, finite modulus row* M*, window row* W*, readback row*

R, seal row E, and structural rows H, C, P, N; it exports no open-cover compactness theorem, selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3568.1. The compact source row K is the only coordinate that names compact metric data, and it is read as finite-net and dyadic-radius information.

Apply the finite-net extraction theorem Definition 3568.3. It places K before the shared modulus row M, the inspected windows W, and the regular rational readback R.

Finally use the nonescape theorem Definition 3568.4. The terminal seal E and the structural rows H, C, P, N preserve the same finite route, so no selector-facing consumer can bypass the $\mathsf{CompactMetric}^{\uparrow}$ source row or replace it by an excluded host compactness principle. □

Theorem 2607.13 (Arzela–Ascoli finite modulus compact-metric sibling readback)**.** *Any finite-modulus selector read through an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first consume the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows of chapter 129 and the* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows before the selector row can be exported. The route passes through the compact-uniform-continuity finite-modulus handoff of chapter 7330; it therefore exposes finite covers, continuity witnesses, and the shared modulus ledger before a downstream consumer may read the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector.*

Proof. Read the compact-metric sibling first. The finite-net rows named by chapter 129 provide the bounded finite cover surface on which a modulus selector can be evaluated.

Next read the $\mathsf{ContinuousMap}^{\uparrow}$ family rows. These rows keep the maps in the family visible as continuous data rather than as an external collection of functions.

Finally pass through chapter 7330. Its finite-modulus handoff is the point where compactness and continuity are combined, so the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ selector row can be exported only after those sibling rows have been consumed. □

Theorem 2607.14 (Arzela–Ascoli compact-equicontinuity finite-modulus dependency)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector route consumes three sibling surfaces before the Arzela–Ascoli readback is public: the compact finite-net source of chapter 129, the shared family-modulus discipline of chapter 4076, and the compact-uniform finite-modulus handoff of chapter 7330. In carrier rows, the compact metric dependency supplies K, equicontinuity supplies the family-level constraint read by F and M, and compact-uniform continuity supplies the finite modulus handoff consumed before W, R, E, H, C, P, N are exported.*

Proof. Start with the compact metric sibling. Its finite-net source gives the only compact-domain row available to the selector route, so the readback cannot begin from an unrecorded host compactness theorem.

Next consume the equicontinuity sibling. The shared family-modulus rows constrain the continuous-map family before the finite-modulus row M can be read as a common selector budget.

Finally use the compact-uniform finite-modulus handoff. It combines compact finite nets with the displayed continuity rows before the stream windows, regular rational readback, and terminal real seal are exposed. Therefore the Arzela–Ascoli readback is a consumer of those three sibling surfaces, not a kernel-only packet detached from the dependency lattice. □

Theorem 2607.15 (Arzela–Ascoli compact-metric sibling route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector row consumes the compact-metric finite-net source of chapter 129, the equicontinuity family rows, the compact-uniform-continuity finite-modulus handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$

seal, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the local* $\mathsf{NameCert}$ *rows before the selector is public. In displayed carrier order the route is*

$$K,\ F,\ U,\ R,\ E,\ Q,\ H,\ C,\ P,\ N,$$

where K *is the compact-metric finite-net source,* F *is the family-level equicontinuity surface,* U *is the compact-uniform finite-modulus handoff,* R *is regular-rational readback,* E *is the terminal Real seal,* Q *is the dyadic tolerance ledger, and* H, C, P, N *are componentwise transport, replay, provenance, and local naming.*

Proof. Read the displayed row order. The compact-metric source K is first because the selector cannot choose a finite modulus over an unnamed compact domain; its substantive sibling source is chapter 129.

Next consume the equicontinuity family row F and the compact-uniform handoff U. These rows expose the family constraint and the finite-modulus transfer before any rational or Real-facing readback occurs.

Finally read R, E, Q, H, C, P, N in the stated order. The regular-rational row, Real seal, dyadic tolerance ledger, transport, replay, provenance, and local naming rows together make the selector public without adding an external compactness theorem, hidden choice of subsequence, or host-level equality. □

Theorem 2607.16 (Arzela–Ascoli compact-image bounded dependency route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *finite-selector read that exports selector-facing modulus data must first consume the compact source nets of chapter 129, the displayed map rows of chapter 130, the finite tolerance ledger of chapter 369, and the bounded image radius route of chapter 3924. In carrier notation, the read stays inside*

$$K, F, M, W, R, E, H, C, P, N.$$

Thus the selector-facing finite modulus is exported only after compact source nets, uniform continuity data, bounded image radius data, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal have all been displayed.*

Proof. Project the accepted carrier of Definition 3568.1. The compact source row K, continuous-map row F, finite modulus row M, window row W, regular readback row R, terminal seal row E, and structural rows H, C, P, N are the only rows available to a selector-facing consumer.

Apply the forward lattice theorem Definition 3568.5. It places compact metric finite-net data, continuous-map inspection, finite common modulus data, finite windows, dyadic tolerance arithmetic, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal before any downstream selector row is public.

Read the compact-image boundedness sibling next. The total-bounded image certificate of chapter 3924 supplies the bounded radius route consumed after the compact source nets, continuous-map rows, and uniform modulus ledger have been displayed.

Finally use subsequence nonescape from Definition 3568.4. Since the carrier has no coordinate for open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness, the selector-facing finite modulus cannot bypass the compact-image boundedness data while remaining inside the accepted packet. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet consumes the finite Cauchy modulus surface of chapter 1486 together with the equicontinuity schedule rows already routed through the compact finite-net lattice. A compactness consumer therefore reads the Cauchy-modulus threshold, the

equicontinuity schedule, and the compact finite-net rows before it may inspect the Arzela–Ascoli finite-modulus packet. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ selector-facing route also consumes the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet of chapter 5549. This child route records the finite compact-net, pointwise modulus, rational lower-bound fold, and $\mathsf{UniformModulus}^{\uparrow}$ handoff rows as a dependency before any finite extraction data is public.

Theorem 2607.17 (Arzela–Ascoli compact-uniform-continuity ledger route)**.** *Every accepted* ArzelaAscoliFiniteMo *selector read that exports finite-modulus data must first consume the* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *rows through finite compact-net data, pointwise modulus rows, rational lower-bound folds, and the terminal* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. In the notation of Definition 3568.1, the read remains inside*

$$K, F, M, W, R, E, H, C, P, N.$$

It therefore exports no open-cover compactness theorem, selected subcover, selected subsequence, host function-space compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The compact source row K, continuous-map family row F, finite modulus row M, finite windows W, regular readback R, terminal seal E, and structural rows H, C, P, N are the only rows available to a selector-facing consumer.

Apply Definition 3568.5. It places compact finite-net data, inspected pointwise rows, finite tolerance ledgers, dyadic arithmetic, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal before any selector row is exported.

Read the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ route through Definition 5549.3. Its lower-bound fold and $\mathsf{UniformModulus}^{\uparrow}$ handoff supply the compact-uniform-continuity dependency consumed by the finite modulus row M.

Finally apply Definition 3568.4. Since the carrier contains no field for open-cover compactness, selected subcovers, selected subsequences, quotient real equality, countable choice, or ambient completeness, the selector-facing export cannot bypass the compact-uniform-continuity ledger. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ finite modulus is consumed by compact-uniform-continuity and equicontinuity-family routes: chapter 388 uses the compact window budget as the continuity handoff, while chapter 413 reads the same finite family modulus before exporting an equicontinuity surface. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ hub consumes the compact-uniform-continuity route of chapter 388, the shared-radius discipline of chapter 4076, and the tolerance-only ledger of chapter 369 before its finite-modulus selector is public. Thus the hub records the actual sibling lattice used by the carrier: compact finite-net rows, equicontinuity rows, uniform-modulus rows, finite windows, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal stay visible as separate dependencies rather than collapsing into a host compactness or function-space compactness theorem. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ route consumes the equicontinuity surface of chapter 4076, the compact-uniform-continuity handoff of chapter 388, and the compact metric finite-net rows before any selector-facing modulus export is public. Thus a downstream selector reads compact source data, equicontinuity bounds, compact-uniform transfer, finite windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the structural H, C, P, N rows as a displayed dependency lattice rather than as host compactness, selected subsequences, quotient real equality, or ambient completeness. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ route consumes the compact-uniform-continuity finite modulus surface of chapter 388 together with the equicontinuity-family rows of chapter 4076 before it extracts the finite-modulus window. The same dependency route may also read the finite-net minimum fold of chapter 619 when the compact source has already exposed its finite inspection schedule, but it receives only the displayed compact source, equicontinuity, modulus-window, BHist,

hsame, Cont, Pkg, and local NameCert rows. The ArzelaAscoliFiniteModulus$^{\uparrow}$ route consumes the CompactMetric$^{\uparrow}$ finite-net data of chapter 129, the Equicontinuity$^{\uparrow}$ family-modulus rows of chapter 4076, and the CompactUniformContinuity$^{\uparrow}$ uniform-modulus handoff of chapter 388 before it exports an Arzela–Ascoli finite-modulus packet. Thus the selector-facing surface is a dependency lattice over compact source nets, shared equicontinuity bounds, compact-uniform continuity transfer, finite windows, regular rational readback, and terminal Real$^{\uparrow}$ sealing, not an isolated compactness principle, host function-space compactness theorem, selected subsequence choice, quotient real equality, or ambient completeness claim. The ArzelaAscoliFiniteModulus$^{\uparrow}$ finite-modulus rows consume the compact finite-net uniform-modulus ledger before exporting equicontinuous family bounds: chapter 388 supplies the compact-uniform-continuity handoff, while chapter 369 supplies the tolerance-to-window modulus ledger that keeps the selector-facing rows finite. The finite Ascoli modulus route consumes the compact uniform modulus row of chapter 2299 and the equicontinuous family row of chapter 413 before finite-subfamily extraction is visible. Thus an accepted ArzelaAscoliFiniteModulus$^{\uparrow}$ packet reads compact-source uniform control and shared family moduli first, then exposes the local selector, window, readback, and seal rows; it exports no host compact-open topology, selected subsequence, or detached function-space compactness principle.

Theorem 2607.18 (Arzela–Ascoli compact uniform modulus dependency route)**.** *Every accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *selector-facing read consumes the* CompactUniformModulus$^{\uparrow}$ *finite-net fold of chapter 2299 as a substantive dependency before the finite-modulus row is public. The same read passes through the compact-uniform-continuity ledger surface of chapter 5549, so compact source coverage, pointwise continuity data, rational lower-bound folding,* UniformModulus$^{\uparrow}$ *output, finite windows,* RegSeqRat$^{\uparrow}$ *readback, and the terminal* Real$^{\uparrow}$ *seal remain one displayed route rather than a local Arzela–Ascoli row alone.*

Proof. Project the accepted carrier of Definition 3568.1. The selector-facing route exposes compact source data, continuous-family data, the finite modulus row, finite windows, regular rational readback, the Real seal, and the structural transport, replay, provenance, and naming rows.

Now read chapter 2299. Its CompactUniformModulus$^{\uparrow}$ finite-net fold supplies the compact-domain coverage and tolerance-only modulus handoff consumed before the Arzela–Ascoli finite modulus can be exported.

Finally read the compact-uniform-continuity ledger route through chapter 5549. That ledger keeps the compact finite-net, pointwise modulus, rational lower-bound fold, and UniformModulus$^{\uparrow}$ handoff visible to the same selector-facing consumer. Since no carrier row supplies host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness, the dependency is a displayed finite-modulus route rather than a background compactness assumption. □

The ArzelaAscoliFiniteModulus$^{\uparrow}$ dependency lattice consumes the finite-net rows of CompactMetric$^{\uparrow}$ through chapter 129, the pointwise-to-uniform modulus rows of CompactUniformContinuity$^{\uparrow}$ through chapter 388, and the tolerance schedules of CauchyModulus$^{\uparrow}$ through chapter 292.

The route is a sibling ledger for compactness and uniform-modulus consumers; it does not add a theorem body to the hub, choose a subcover, select a modulus from an ambient proof, or assert compact-open completeness. The hub route for ArzelaAscoliFiniteModulus$^{\uparrow}$ records the selector-facing dependency already present in the carrier: a finite-modulus read must pass through the compact-open metric rows of chapter 1903 and the compact-uniform handoff rows of chapter 388 before the rows $K, F, M, W, R, E, H, C, P, N$ can be exported to a downstream selector. Thus the hub links the compact-open comparison surface and compact-uniform continuity surface through

the existing finite-modulus selector, not through a host function-space compactness theorem, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 2607.19 (Compact-open equicontinuity dependency route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *finite-modulus read that compares a family of maps consumes the compact-open metric surface of chapter 1903 and the equicontinuity family surface of chapter 4076 before exporting selector-facing rows. The exported selector data therefore remains downstream of compact-open source-window comparison, equicontinuity radius discipline, finite modulus rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural* H, C, P, N *rows, rather than a host function-space compactness theorem or a selected subsequence principle.*

Proof. Project the carrier of Definition 3568.1. A family-comparison read already exposes compact source data, continuous-map rows, finite modulus rows, window ledgers, regular readback, the terminal Real seal, and structural replay rows.

Use the compact-open dependency route of Definition 3568.7. It places the compact-open metric comparison surface before any selector-facing finite-modulus export can be read.

Use the equicontinuity anchors Definition 3568.23 and Definition 3568.25. They force the same family comparison to consume the equicontinuity family-window and shared-radius rows before finite subfamily modulus data is public.

Thus a selector read that compares maps must pass through both sibling surfaces. Since neither sibling supplies host function-space compactness, selected subsequences, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the Arzela–Ascoli export remains a finite displayed dependency route. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ sibling lattice is read through the compact source rows of chapter 129 and the compact-uniform modulus rows of chapter 388 before either the forward finite-modulus lattice of Definition 3568.5 or the compact-uniform finite-selector route of Definition 3568.9 can export selector-facing data. The equicontinuity children Definition 3568.10 and Definition 3568.25 then consume the same displayed rows $K, F, M, W, R, E, H, C, P, N$, so the hub records a genuine lattice dependency rather than an isolated kernel float.

Theorem 2607.20 (Arzela–Ascoli sibling lattice repair)**.** *Every selector-facing export from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the sibling rows of chapter 129, chapter 388, chapter 4076, chapter 126, chapter 301, chapter 38, and chapter 291, namely*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{CompactUniformContinuity}^{\uparrow},\quad \mathsf{Equicontinuity}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Dyadi}$$

before any finite-modulus selector row is public. The export therefore reads compact source data, compact-uniform continuity data, equicontinuity discipline, finite stream windows, regular-rational readback, the terminal real-name seal, and dyadic tolerance arithmetic as separate certificate rows.

Proof. Project the carrier of Definition 3568.1. Its selector-facing surface already exposes the compact source row, finite window row, regular rational readback, real seal, transport, replay, provenance, and local naming rows.

Apply the sibling lattice routes recorded in Definition 3568.5, Definition 3568.12, Definition 3568.9, and Definition 3568.26. These routes place $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, and $\mathsf{Equicontinuity}^{\uparrow}$ before the exported selector row.

The same carrier supplies $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic. Since the export is obtained only after those displayed rows are read, no selector-facing consumer can bypass the horizontal dependency lattice. □

Theorem 2607.21 (Arzela–Ascoli compact-metric source link)**.** *Every selector-facing* ArzelaAscoliFiniteModulus$^{\uparrow}$ *read first consumes the compact source row through chapter 129 and the compact-uniform route before exporting finite-modulus selector data. Thus the selector export is downstream of the* CompactMetric$^{\uparrow}$ *source,* CompactUniformContinuity$^{\uparrow}$ *transport, and the finite carrier rows of* ArzelaAscoliFiniteModulus$^{\uparrow}$*, not a host compactness principle, selected-subsequence principle, quotient-real equality, countable-choice principle, or ambient* Real$^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier of Definition 3568.1. Its selector surface has a compact source coordinate together with finite window, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Apply Definition 3568.20. The theorem places the compact source row, compact-uniform route, equicontinuity discipline, stream windows, regular-rational readback, real seal, and dyadic arithmetic before the selector-facing export.

The compact source is therefore a named certificate dependency rather than a host compactness assumption. Because the carrier and sibling route add no coordinate for selected subsequences, quotient real equality, countable choice, or ambient Real$^{\uparrow}$ completeness, every selector-facing export must pass through the displayed compact-metric source link first. □

Theorem 2607.22 (Arzela–Ascoli finite-modulus Equicontinuity anchor)**.** *Every* ArzelaAscoliFiniteModulus$^{\uparrow}$ *finite-modulus read first consumes the shared-family modulus and compact-source route of chapter 4076 before any finite subfamily, diagonal, compactness-facing, or selector-facing handoff is public. The anchor route exposes the compact source, continuous-family ledger, tolerance row, shared radius row, uniform-modulus handoff, transport, replay, provenance, and local naming rows supplied by* Equicontinuity$^{\uparrow}$*, and only then may the finite-modulus packet read the compact-uniform-continuity, finite-window, regular-rational, dyadic-tolerance, and terminal Real rows of the Arzela–Ascoli lattice.*

Proof. Read the Equicontinuity$^{\uparrow}$ carrier first through Definition 4076.1. Its rows expose the compact metric source, continuous-family ledger, tolerance, shared radius, uniform-modulus handoff, transport, replay, provenance, and local naming before any family selector is visible.

Apply Definition 4076.3 and Definition 4076.7. These force compact-source and shared-radius consumption before a compact-uniform or finite-net consumer can use the family.

Now pass to the finite-modulus lattice through Definition 3568.24 and Definition 3568.25. Those routes identify the downstream Arzela–Ascoli rows as consumers of the equicontinuity family ledger, so finite subfamilies, diagonal data, compactness-facing handoffs, and selector outputs remain downstream of the Equicontinuity$^{\uparrow}$ anchor. □

Theorem 2607.23 (Arzela–Ascoli finite-modulus equicontinuity forward anchor)**.** *Every* ArzelaAscoliFiniteModulus *finite-modulus read first consumes the* Equicontinuity$^{\uparrow}$ *family-window and shared-radius rows before it exports compact finite-subfamily data. In carrier terms, the read must expose the equicontinuity compact source, continuous-family ledger, tolerance row, shared radius row, uniform-modulus handoff, transport, replay, provenance, and local naming supplied by chapter 4076; only then may the Arzela–Ascoli finite-modulus packet read its compact finite-subfamily, diagonal, modulus, and selector-facing rows.*

Proof. Project the existing finite-modulus rows through the equicontinuity anchor of Definition 3568.22. That route already places the Equicontinuity$^{\uparrow}$ compact source, family ledger, tolerance, shared radius, uniform-modulus handoff, transport, replay, provenance, and local naming before any finite-modulus consumer is public.

Use chapter 4076 to identify the family-window and shared-radius rows as substantive sibling data rather than optional annotations. The finite-modulus packet can therefore export compact finite-subfamily data only after those rows have been consumed.

Finally apply the downstream dependency routes of this chapter, which read finite subfamilies, diagonal data, compactness-facing handoffs, and selector rows as Arzela–Ascoli consumers of the equicontinuity family surface. This proves the forward anchor without adding a second source for compactness or equicontinuity. □

Theorem 2607.24 (Arzela–Ascoli finite-modulus equicontinuity dependency route)**.** *Every finite-modulus extraction read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier factors through the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *ledger, the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-modulus handoff, the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any selector output is public. The route exports no selected subsequence, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $A = (K, F, M, W, R, E, H, C, P, N)$. The compact metric row K, continuous-map family row F, finite modulus row M, window row W, regular-rational readback row R, and seal row E are the whole analytic surface available before a selector can read output.

Apply Definition 3568.5. It places every downstream selector read through the compact metric, continuous-map, finite common modulus, window, readback, Real seal, and dyadic tolerance rows.

Use the sibling routes of chapter 4076 and chapter 388. The equicontinuity family supplies the shared finite family ledger, while compact uniform continuity supplies the finite-modulus handoff consumed by the selector surface.

Finally use Definition 3568.9. It identifies the compact-uniform-continuity consumer as the same finite selector route, so selector output cannot become public before the listed dependency rows are displayed. □

Theorem 2607.25 (Arzela–Ascoli EquicontinuityFamily dependency route)**.** *Every finite-modulus extraction read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier consumes the family-level ledger of chapter 413 before any selector row is exported. The route then passes through the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-modulus handoff, the compact metric source row, the continuous-map family row, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no selected subsequence, host function-space compactness, quotient equality of real names, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or selector row detached from the family ledger.*

Proof. Project the accepted carrier $A = (K, F, M, W, R, E, H, C, P, N)$. The continuous-map family row F is not a bare list of maps: it is read as the family surface whose shared modulus and common finite-window obligations are supplied by $\mathsf{EquicontinuityFamily}^{\uparrow}$.

Apply Definition 3568.24. It already forces the finite-modulus extraction through the equicontinuity ledger, compact-uniform-continuity handoff, compact source, continuous-map rows, windows, readback, and terminal Real seal before selector output becomes public.

It remains to name the family-level source. The sibling chapter 413 supplies the displayed family ledger consumed by F and M, while chapter 388 supplies the finite-modulus handoff consumed after that ledger is visible.

Therefore a selector-facing consumer cannot bypass $\mathsf{EquicontinuityFamily}^{\uparrow}$. The rows H, C, P, N preserve only the same displayed carrier, so no selected subsequence, host function-space compactness, quotient real equality, ambient completeness, or detached selector coordinate is exported. □

Theorem 2607.26 (Arzela–Ascoli finite-modulus sibling dependency route)**.** *Every accepted* ArzelaAscoliFiniteMod *read must pass through the* CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, CompactUniformContinuityLedger$^{\uparrow}$ *of chapter 5549,* UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *sibling rows before it exports the finite-modulus handoff. The route exposes compact metric source data, equicontinuity-family discipline, compact-uniform-continuity ledger data, uniform tolerance-to-precision data, finite stream windows, regular-rational readback, and the terminal Real seal as separate rows, and it does not replace them by host compactness, selected subsequences, quotient Real equality, or ambient completeness.*

Proof. Read the finite carrier of Definition 3568.1. Its selector-facing handoff is available only after the compact source, finite-window, regular readback, Real seal, transport, replay, provenance, and local naming rows have been displayed.

Now use the sibling lattice and forward routes, including Definition 3568.5, Definition 3568.25, and Definition 3568.30. These rows insert the CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, and UniformModulus$^{\uparrow}$ dependencies before the exported finite modulus.

The compact-uniform-continuity ledger supplies the finite compact-net row and the pointwise modulus lower-bound fold consumed before the exported finite modulus.

The remaining rows are the L10 source rows StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ already carried by the same finite packet. Since the handoff is obtained by exhausting those displayed rows, no host compactness principle, selected subsequence, quotient equality, or ambient completeness row can substitute for the sibling dependency route. □

Theorem 2607.27 (Arzela–Ascoli sibling dependency export)**.** *Every selector-facing read exported from an accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *packet factors through the displayed horizontal dependency lattice*

CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Dyadi

The export may consume compact finite-net data, equicontinuity radii, compact-uniform transfer, finite stream windows, regular rational schedules, terminal real seals, and dyadic tolerance arithmetic, but it has no coordinate for host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness.

Proof. Read the carrier rows of Definition 3568.1. The compact source row, finite-window row, regular-rational readback, terminal real seal, transport, replay, provenance, and local naming row are all displayed before the selector-facing finite-modulus handoff is public.

The sibling route theorems refine those rows. The compact-metric, equicontinuity, and compact-uniform-continuity routes place CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, and CompactUniformContinuity$^{\uparrow}$ before the exported selector row.

The L10 readback rows then supply StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ as the finite window, rational schedule, terminal seal, and tolerance-arithmetic part of the same displayed packet. Since the export is obtained only after these rows are visible, every selector-facing read factors through the stated lattice. □

Theorem 2607.28 (Arzela–Ascoli sibling route export)**.** *Every selector-facing export from an accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *packet factors through the displayed sibling route*

CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqR

The export may read compact finite nets, equicontinuity bounds, compact-uniform-continuity transfer, tolerance-to-precision ledgers, finite stream windows, regular rational readback, the terminal

real seal, and dyadic tolerance arithmetic; it does not obtain a selector row from host compactness, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the forward lattice of Definition 3568.5. It places compact metric finite-net data, continuous-map inspection, finite common modulus rows, stream windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and dyadic tolerance arithmetic before any selector-facing handoff is public.

The sibling routes refine that lattice. The compact-metric, equicontinuity, compact-uniform-continuity, and uniform-modulus dependency theorems force $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{CompactUniformCon}$ and $\mathsf{UniformModulus}^{\uparrow}$ rows to be displayed before the finite-modulus export can be consumed.

The L10 readback rows are then fixed by the same carrier. $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational schedules, $\mathsf{Real}^{\uparrow}$ seals the terminal values, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance arithmetic. Since the carrier has no coordinate for host compactness, selected subsequence choice, quotient real equality, or ambient completeness, every selector-facing export factors through the stated sibling route. □

Theorem 2607.29 (Arzela–Ascoli finite modulus proper-map consumer)**.** *Every* $\mathsf{ProperMap}^{\uparrow}$ *compact-preimage consumer that reads a finite-modulus selector from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first display the dependency route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{CompactUniformContinuity}^{\uparrow},\quad \mathsf{Equicontinuity}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Dyadi}$$

The consumer may then use the finite-modulus selector as compact-preimage evidence for the proper-map boundary; it does not obtain host compactness, selected subsequences, quotient real equality, ambient Real completeness, or a ProperMap compact-preimage row detached from the displayed selector route.

Proof. Project the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ carrier through the sibling route export. Definition 3568.28 places compact metric data, equicontinuity bounds, compact-uniform-continuity transfer, finite stream windows, regular rational readback, the terminal Real seal, and dyadic tolerance arithmetic before any selector-facing export.

Next use the existing sibling lattice repair. The lattice paragraph preceding Definition 3568.28 ties the compact source rows of chapter 129 and the compact-uniform rows of chapter 388 to the equicontinuity children, so the selector cannot be read before the displayed compact and modulus rows.

Finally attach the proper-map compact-preimage boundary. **??** requires every compact-preimage read to remain behind the source, target, continuous-map, compact-preimage, transport, replay, provenance, and local naming rows of $\mathsf{ProperMap}^{\uparrow}$. Hence a proper-map consumer may use the finite-modulus selector only after all rows in the displayed route have been shown. □

Theorem 2607.30 (Arzela–Ascoli UniformModulus forward route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector read must first consume the* $\mathsf{UniformModulus}^{\uparrow}$ *finite tolerance-to-precision ledger, together with the* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, before any selector-facing finite-modulus data is public.*

The read remains inside the displayed carrier route

$$K, F, M, W, R, E, H, C, P, N.$$

It does not export host function-space compactness, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The compact metric row K, continuous-map family row F, finite modulus row M, finite windows W, regular readback R, Real seal E, and structural rows H, C, P, N are the only visible selector coordinates.

Apply the existing forward lattice, Definition 3568.5. It already forces downstream selector reads to pass through the compact metric, continuous-map, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before the selector-facing handoff.

Now read the uniform-modulus chapter, chapter 369. Its finite ProbeBundle fold supplies the tolerance-to-precision ledger that keeps the image-side windows tied to one displayed compact-continuous modulus route.

Repack the route through H, C, P, N. Since no carrier coordinate supplies host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness, the UniformModulus handoff is a dependency edge inside the finite packet rather than an added analytic principle. □

The dependence on chapter 369 is substantive: $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ consumes $\mathsf{UniformModulus}^{\uparrow}$ through the finite ProbeBundle lower-bound fold and the continuous-map modulus rows before any selector-facing modulus data is visible. The route keeps compact-domain coverage, continuous-map modulus readback, and finite image-window selection in one displayed ledger instead of treating uniform continuity as background analysis.

Theorem 2607.31 (Arzela–Ascoli Cauchy-filter completeness forward route)**.** *Every completion-facing read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet may consume Cauchy-filter completion data only after the compact-metric source, equicontinuity family, uniform-modulus handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal have already been displayed. The dependency route is read through chapter 4076, chapter 369, and chapter 3885; it exports no selected subsequence, quotient filter, selected limit, countable-choice schedule, host function-space compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The rows $K, F, M, W, R, E, H, C, P, N$ expose the compact metric source, continuous-map family, finite common modulus, finite windows, regular readback, terminal real seal, transport, replay, provenance, and local naming before any downstream completion read is public.

Use the equicontinuity and uniform-modulus sibling routes, including Definition 3568.24 and Definition 3568.30. They force the equicontinuity family and $\mathsf{UniformModulus}^{\uparrow}$ tolerance ledger to be visible before selector-facing finite-modulus data is exported.

Now read the Cauchy-filter completion criterion and Cauchy-filter completeness surfaces. The completion consumer may inspect those rows only as a continuation of the displayed compact-source, family, modulus, window, readback, and Real-seal route, so the Cauchy-filter dependency is a forward row-level handoff rather than a hidden completeness theorem.

The refused endpoints are absent from the carrier and from the cited completion-facing route. Thus the Arzela–Ascoli finite-modulus packet can pass Cauchy-filter completeness data forward without exporting selected subsequences, quotient filters, selected limits, countable choice, host function-space compactness, or ambient Real completeness. □

Theorem 2607.32 (Arzela–Ascoli Real-and-completion dependency route)**.** *Every Real-and-completion-facing read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the route through finite modulus data, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The route therefore reads chapter 38, chapter 301, chapter 126, and chapter 291 as row-level dependencies, not as a selected limit, quotient*

stream equality, countable-choice schedule, host function-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier of Definition 3568.1. The finite modulus row M, window row W, regular readback row R, and terminal seal row E are part of the displayed packet.

Read the Real-and-completion route in order. The finite modulus controls which family values may be compared, the $\mathsf{StreamName}^{\uparrow}$ row bounds the inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational handoff, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells keep each read tied to the same finite precision ledger.

Only after those rows have been displayed may the consumer see the $\mathsf{Real}^{\uparrow}$ seal. Since none of the carrier rows contains a selected limit, quotient stream equality, countable-choice schedule, host function-space compactness, or ambient completeness coordinate, the dependency remains a finite row-level route. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ dependency lattice consumes the Cauchy sequential criterion of chapter 5139 only after its compact finite-modulus, equicontinuity, finite-window, regular readback, and terminal real-seal rows have been displayed. Thus a sequence-facing compactness consumer reads the same compact metric row K, continuous-family row F, finite modulus row M, window row W, readback row R, seal row E, transport row H, replay row C, provenance row P, and local naming row N before the Cauchy sequential criterion can be used as a completion-facing dependency.

Every $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ read that reaches the Cauchy sequential criterion first exposes the compact finite-modulus and equicontinuity rows named by Definition 3568.5 and Definition 3568.9. The dependency is a displayed row handoff to chapter 5139, not a host compactness theorem, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Chapter 2608

A Concrete Naming Certificate for $\mathsf{CompactTotallyBounded}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactTotallyBounded}^{\uparrow}$ packet records the constructive compact-metric surface obtained from a complete metric carrier with explicit finite nets at visible positive radii. It sits between the finite-net interface of chapter 127, the limit-service interface of chapter 128, and the compact-metric interface of chapter 129. The packet names the finite bridge used by compact and continuous arguments; it does not turn metric compactness into open-cover compactness, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2608.1 (Compact totally bounded carrier)**.** *A* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, C, T, N, E, H, K, P, L)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric row, including the point carrier, point classifier, distance witness relation, and rational comparison ledger used by every compactness read.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row. It records only displayed regular Cauchy streams, moduli, limit histories, and late-distance ledgers available inside the metric carrier.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row. For each visible positive radius it exposes a closed* $\mathsf{ProbeBundle}$ *of center histories, membership constructors, and coverage witnesses.*
4. *N is the net-radius schedule. It records the positive* $\mathsf{Rat}^{\uparrow}$ *radii at which the finite-net rows are read and the refinement relation between requested tolerances and displayed net radii.*
5. *E is the exactness row tying M, C, T, N together. It says that each distance comparison used by T and each convergence comparison used by C is a row of the same metric source M.*
6. *H records componentwise* hsame *transport for the metric, complete-metric, totally-bounded, radius, exactness, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a compactness request through M, C, T, N, E, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactTotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $M, C, T, N, E, H, K, P, L$. It has no coordinate for open covers, selected subcovers, quotient metric points, host equality of Cauchy streams, a choice-selected finite net, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.CompactTotallyBoundedUp.compactTotallyBoundedEncodeBHist`

Theorem 2608.2 (Compact totally bounded carrier admission)**.** *For every accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier $J = (M, C, T, N, E, H, K, P, L)$, the local carrier admits exactly the metric source, complete-metric limit service, totally-bounded finite-net family, radius schedule, exactness row, transport row, replay row, provenance row, and local naming row displayed in Definition 3569.1.*

Proof. Project the accepted packet. The mathematical rows are M, C, T, N, E, and the structural rows are H, K, P, L.

No other coordinate is present in the carrier. Hence any accepted packet is admitted through exactly those rows, with the metric, completion, finite-net, radius, exactness, transport, replay, provenance, and naming data listed in the definition. □

Theorem 2608.3 (Compact totally bounded classifier stability)**.** *The classifier of an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier is stable under componentwise* hsame *transport of the metric row, complete-metric row, totally-bounded finite-net row, radius schedule, and exactness row. The transported packet has the same displayed* Cont *replay,* Pkg *provenance, and local* NameCert *surface.*

Proof. Use the row H. It records componentwise transport for M, C, T, N, E and for the structural rows.

The replay row K reads only the transported coordinates, and P, L record only the same provenance and local naming after transport. Thus the classifier cannot change by moving to an unrecorded metric, net family, limit service, or host compactness principle. □

Theorem 2608.4 (Compact totally bounded finite-net exhaustion)**.** *Every compactness read from an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier factors its finite coverage demand through the displayed* $\mathsf{TotallyBounded}^{\uparrow}$ *row T and the radius schedule N. For each requested positive tolerance, the read obtains only the closed* ProbeBundle *centers, membership rows, metric coverage rows, and rational comparison ledgers exposed by those two rows.*

Proof. Begin with a compactness read and inspect the finite-net coordinate. The only available finite-net family is T.

The radius at which the finite net is consumed is not chosen externally. It is read from N, and the exactness row E ties the resulting coverage distances back to the metric row M.

Therefore finite coverage is exhausted by T, N, E over the metric source. A host enumeration, a selected finite subcover, or an ambient compactness theorem has no row in the packet. □

Theorem 2608.5 (Compact totally bounded completion handoff)**.** *Every Cauchy or completion-facing read from an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier reaches the complete-metric service only through the displayed row C, after the same metric row M and exactness row E have been fixed. The handoff exports no selected limit outside the carrier and no quotient equality of Cauchy names.*

Proof. The complete-metric coordinate is C. It names the displayed streams, moduli, limit histories, and late-distance ledgers available in the carrier.

The row E ties those convergence comparisons to M, so a completion-facing consumer cannot replace the metric source or import a host equality of Cauchy names.

Since the carrier contains no selector outside C and no quotient coordinate, the completion handoff remains the displayed finite BEDC route. □

Theorem 2608.6 (Compact totally bounded nonescape)**.** *No accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *packet exports open-cover compactness, selected subcovers, selected limits, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its compact-metric content is exactly the finite route*

$$M, C, T, N, E, H, K, P, L.$$

Proof. Apply the carrier admission theorem. Every public read of the packet factors through the displayed tuple $M, C, T, N, E, H, K, P, L$.

Finite coverage is confined to T, N, E, and completion service is confined to C, E. The transport, replay, provenance, and local naming rows preserve those routes.

The excluded principles do not occur among the carrier coordinates. They therefore cannot be exported by an accepted packet without leaving the local $\mathsf{CompactTotallyBounded}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactTotallyBounded}^{\uparrow}$ packet is generated from a MetricSpace row, CompleteMetric limit-service row, TotallyBounded finite-net row, net-radius schedule, exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3569.1, Definition 3569.2, Definition 3569.3, Definition 3569.4, Definition 3569.5, and Definition 3569.6 seed the constructive compact-metric bridge from complete metric service and finite total-bounded nets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected limits, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite-net exhaustion, completion handoff, nonescape, and local NameCert registration for BEDC.Derived.CompactTotallyBoundedUp.

Chapter 2609

A Concrete Naming Certificate for $\mathsf{BaireCategoryTheorem}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireCategoryTheorem}^{\uparrow}$ packet records the theorem-level complete-metric Baire route as finite NameCert data. It sits over the dense-open schedule of $\mathsf{BaireCategory}^{\uparrow}$ in chapter 1542, the inspected complete-metric route of $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ in chapter 3535, the Cauchy service of $\mathsf{CompleteMetric}^{\uparrow}$ in chapter 128, the carrier comparisons of $\mathsf{Met}^{\uparrow}$ in chapter 57, the finite observation windows of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular-rational readback of $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, and the terminal seal of $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet names the finite route by which a complete-metric dense-open construction is exposed to downstream theorem consumers; it does not assert a host intersection object, a countable-choice selector, an open-cover compactness theorem, or ambient completeness outside the displayed route.

Definition 2609.1 (Baire category theorem carrier)**.** *A* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, U, K, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, including the finite predecessor-ball and refinement data offered by the Baire-category packet.*
2. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection row, exposing the complete-metric Baire surface through dense-open, Cauchy-thread, metric, stream, regular-rational, and real-seal reads.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row naming the Cauchy-thread and limit-service data used by the theorem-level route.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row naming the metric carrier, ball classifier, distance comparisons, and shrinkage checks consumed by B, U, K.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the schedule, centers, radii, and requested tolerances.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular-rational center, radius, and tolerance data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after B, U, K, M, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the theorem route and all structural rows.*
9. *C records displayed* Cont *replay of the dense-open, complete-metric, metric, finite-window, readback, and real-seal route.*

10. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host countable intersection, countable choice, a selected point outside the displayed readback, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BaireCategoryTheoremUp.BaireCategoryTheoremTasteGate single carrier alignment`

Theorem 2609.2 (Baire category theorem NameCert obligations)**.** *For an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier*

$$T = (B, U, K, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed theorem route, dense-open schedule exposure through B, complete-metric Baire inspection through U, Cauchy-thread service through K, metric comparison exposure through M, finite-window and regular-rational readback through S, R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3570.1. Its analytic-facing rows are exactly B, U, K, M, S, R, E, followed by the structural rows H, C, P, N.

The dense-open schedule is exposed first through B and then inspected through U. The complete-metric Baire surface row binds that schedule to a displayed Cauchy thread, so the theorem-level route cannot begin from an unrecorded host intersection.

The rows K and M provide the complete-metric service and metric comparisons consumed by the route. The rows S and R then record the finite windows and regular-rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, these rows exhaust the local NameCert obligation surface. □

Theorem 2609.3 (Complete-metric Baire theorem nonescape)**.** *Every theorem-level consumer of an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

and through the structural rows BHist, hsame, Cont, Pkg, NameCert. *The consumer receives the finite complete-metric Baire route carried by $T = (B, U, K, M, S, R, E, H, C, P, N)$, not a host countable intersection, countable-choice selector, ambient complete-metric theorem, open-cover compactness theorem, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Start with the NameCert obligations of Definition 3570.2. They force the consumer to read the dense-open row B, the complete-metric Baire inspection row U, the complete-metric Cauchy service K, the metric comparison row M, the finite-window row S, the regular-rational readback R, and the terminal real seal E in the displayed order.

The surface row U imports only the inspected route already named by $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$. Thus the theorem-level packet packages that route for downstream use without creating a second intersection or selector coordinate.

The structural rows H, C, P, N preserve the same route under accepted transport, replay the same observations, source the same package, and name the same local certificate. Coordinate exhaustion then blocks every excluded host object in the statement. □

Definition 2609.4 (BaireCategoryTheorem finite certificate route)**.** *For an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier* $T = (B, U, K, M, S, R, E, H, C, P, N)$*, the finite certificate route is the ordered row*

$$\mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

The source coordinate B *supplies the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule,* U *supplies the* $\mathsf{CompleteMetricBaireSurface}$ *inspection row,* K *and* M *supply the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *service rows,* S *and* R *supply the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* E *supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The structural rows* H, C, P, N *transport, replay, source, and name only this route.*

Theorem 2609.5 (BaireCategoryTheorem schedule induction)**.** *Let* $T = (B, U, K, M, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier. For every finite dense-open prefix listed in the* $\mathsf{BaireCategory}^{\uparrow}$ *schedule row* B*, the prefix is matched by a predecessor ball, a successor refinement ball, a complete-metric Cauchy-thread row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal in the same order as the route of Definition 3570.4. The structural rows* H, C, P, N *preserve the constructed prefix route under componentwise transport, displayed replay, provenance, and local naming.*

Proof. Use primitive induction over the displayed finite schedule prefix in B. The base case reads the initial ball data from the carrier of Definition 3570.1 and the metric comparison row made available by M. The NameCert obligations of Definition 3570.2 attach that initial read to the complete-metric inspection route before any theorem-level consumer can inspect it.

For the successor step, assume the predecessor prefix already has its predecessor ball, Cauchy-thread row, finite windows, regular-rational readback, and real seal. Consume the next dense-open request from B, refine inside the predecessor ball through the $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ rows U, K, and record the successor refinement ball through the same metric row M.

Append the finite observation data through S and R. The $\mathsf{StreamName}^{\uparrow}$ windows record the schedule, center, radius, and tolerance reads for the successor ball, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback converts those finite windows into the regular-rational row used before the terminal seal E.

Finally transport the extended prefix through H, C, P, N. These rows do not introduce a second schedule or an unrecorded point; they preserve the route named by Definition 3570.4. Thus every finite dense-open prefix is matched by the displayed predecessor, successor, Cauchy-thread, finite-window, readback, and real-seal rows in order. □

Theorem 2609.6 (BaireCategoryTheorem finite-prefix nested ball route)**.** *Let* $T = (B, U, K, M, S, R, E, H, C, P, N$ *be an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier. Every finite dense-open prefix recorded in the schedule row* B *determines a nested predecessor/successor ball chain in which each successor ball is read through the same metric row* M*, its radius and requested tolerance are displayed through the finite-window and regular-rational rows* S, R*, containment in the predecessor ball is preserved by the transport and replay rows* H, C*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is reachable only after that finite nested chain has been displayed. The chain is part of the finite certificate route of Definition 3570.4 and carries no host intersection object, countable-choice selector, or ambient completeness assertion outside the accepted packet.*

Proof. Use induction over the finite prefix, as in Definition 3570.5. The base case reads the first predecessor ball from B and the metric comparison row from M; the NameCert obligations of Definition 3570.2 attach those reads to the displayed theorem route before any terminal seal can be inspected.

For the successor step, assume the previous prefix has already produced a nested chain whose last ball is the current predecessor. The next dense-open request in B is consumed through the finite certificate route of Definition 3570.4, and the successor ball is read through the same M row rather than through a new metric carrier.

The radius and tolerance attached to the successor ball are then exposed through S and R. The row S supplies the finite schedule window, center, radius, and requested tolerance reads, while R records the regular-rational readback used by the route before any $\mathsf{Real}^{\uparrow}$ seal is available.

It remains to preserve containment. The transport row H carries the predecessor/successor comparison componentwise, and the replay row C displays the same containment relation inside the accepted packet. Thus the terminal seal E can be reached only after the finite nested chain has been built, transported, and replayed.

No step introduces a coordinate for a host intersection, a countable-choice selector, or ambient completeness. Those objects are absent from the carrier of Definition 3570.1, so the induction isolates exactly the finite nested-ball invariant consumed by downstream complete-metric readers. □

Theorem 2609.7 (BaireCategoryTheorem route nonescape)**.** *Every theorem-level read admitted by an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier is exhausted by the finite certificate route of Definition 3570.4. In particular, any downstream complete-metric, compactness, or Baire-surface consumer receives the displayed chain through* $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support, and no additional host intersection, choice selector, open-cover compactness row, or ambient real-completeness row.*

Proof. Start with the carrier definition Definition 3570.1. It lists exactly the analytic rows B, U, K, M, S, R, E and the structural rows H, C, P, N.

Apply the local NameCert obligations in Definition 3570.2. Those obligations identify the dense-open schedule, complete-metric Baire inspection, Cauchy-thread service, metric comparison, finite-window readback, regular-rational readback, and terminal real seal as the only theorem-facing analytic rows.

Finally use Definition 3570.3. It already blocks host intersections, countable-choice selectors, ambient complete-metric theorems, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the displayed chain. Therefore every theorem-level read is exhausted by the route named in Definition 3570.4. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3570.1, Definition 3570.2, and Definition 3570.3 seed the theorem-level complete-metric Baire route over dense-open schedules, complete-metric Baire inspection, Cauchy-thread service, metric comparison, finite windows, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host intersections, countable choice, selected points outside the displayed readback, open-cover compactness, ambient complete-metric theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric Baire inspection, Cauchy-thread service, metric comparison exposure, finite-window readback, regular-rational readback, real sealing, and nonescape for BEDC.Derived.BaireCategoryTheoremUp.

Constructive story: A $\mathsf{BaireCategoryTheorem}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteMetricBaireSurface inspection row, a CompleteMetric Cauchy-thread service row, MetricSpace comparison rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2610

A Concrete Naming Certificate for RationalBracketRefinement$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RationalBracketRefinement$^{\uparrow}$ packet records the finite rational operation that tightens a displayed bracket after a dyadic tolerance request and before the terminal Real$^{\uparrow}$ seal is read. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the real-name seal of chapter 38, the finite bracketing surface of chapter 1807, and the Archimedean approximation boundary of chapter 1507. The packet names refinement as finite rational bracket data; it is not a host-order trichotomy principle, a quotient-real equality test, a selected limiting interval, or a completeness theorem.

Definition 2610.1 (RationalBracketRefinement carrier)**.** *A* RationalBracketRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N)$$

with the following rows:

- *B is the incoming* FiniteBracketingInterval$^{\uparrow}$ *bracket row.*
- *ϵ is the* DyadicRatCore$^{\uparrow}$ *tolerance row that controls the requested rational refinement scale.*
- *L, U are the incoming rational endpoint rows and L', U' are the displayed refined rational endpoint rows.*
- *W is the* StreamName$^{\uparrow}$ *finite observation window used to inspect the named object.*
- *G is the* RegSeqRat$^{\uparrow}$ *readback row extracted from W.*
- *A is the* ArchimedeanApproximation$^{\uparrow}$ *row that supplies the finite bound and scale comparison used by the refinement.*
- *E is the terminal* Real$^{\uparrow}$ *seal reached only after the refined bracket has been displayed.*
- *H records componentwise* hsame *transport for the bracket, tolerance, endpoint, window, readback, approximation, real seal, replay, provenance, and local name.*
- *C is the displayed* Cont *replay row for the same finite route.*
- *P is the* Pkg *provenance over* RationalBracketRefinement$^{\uparrow}$, FiniteBracketingInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, ArchimedeanApproximation$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

- *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$ *row.*

The carrier accepts only packets in which the refined endpoints are read after the tolerance row and before the real seal, using the same finite window, regular rational readback, and Archimedean approximation rows.

Lean4 ✓ `BEDC.Derived.RationalBracketRefinementUp.RationalBracketRefinementTasteGate single carrier alignment`

Theorem 2610.2 (RationalBracketRefinement NameCert obligations)**.** *For every accepted* RationalBracketRefinem *carrier*

$$R = (B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the incoming* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, incoming and refined rational endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *scale row,* $\mathsf{Real}^{\uparrow}$ *terminal seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may read host real order, quotient equality of real names, trichotomy, selected infinite refinement, or ambient completeness.*

Proof. Project the accepted carrier of Definition 3572.1. The rows $B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N$ are the entire inventory available to the local certificate.

The bracket row B supplies the incoming endpoint surface, and the dyadic tolerance row ϵ fixes the finite refinement scale. The refined endpoint rows L', U' are therefore checked against displayed rational and dyadic data, not against a host real-order oracle.

The window row W and the readback row G keep the inspected object on the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route. The Archimedean approximation row A supplies the finite scale handoff, and the real seal E is available only after those rows have been displayed.

Transport, replay, provenance, and local naming are the remaining structural rows H, C, P, N. Since the carrier has no coordinate for trichotomy, quotient equality, selected infinite refinement, or completeness, none of those principles can enter the local NameCert obligations. □

Theorem 2610.3 (RationalBracketRefinement L10 handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{RationalBracketRefinement}^{\uparrow}$ *packet reaches the refined rational bracket only through the finite route*

$$\mathsf{FiniteBracketingInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{ArchimedeanApproximation}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff displays the incoming bracket, the dyadic tolerance, the regular rational readback, and the Archimedean scale row before the terminal $\mathsf{Real}^{\uparrow}$ *seal is exported.*

Proof. Begin with a consumer requesting a refined bracket for a displayed real name. The incoming bracket row B and the dyadic tolerance row ϵ are read first, so the refinement request has a finite endpoint surface and a finite scale.

Next open the finite window W and its regular rational readback G. These rows keep the computation on the StreamName and RegSeqRat route before any Real-facing consumer can inspect the terminal seal.

The Archimedean approximation row A binds the bound and scale data to the same finite packet, and the refined endpoint rows L', U' are then available as rational bracket data. The real seal E is exported only after this displayed route has been traversed.

Because the route never introduces host order, quotient equality, selected limiting intervals, or completeness, the handoff exports only the finite rational bracket refinement surface named by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBracketRefinement}^{\uparrow}$ packet is generated from an incoming FiniteBracketingInterval bracket row, a DyadicRatCore tolerance row, incoming and refined rational endpoint rows, StreamName finite observation windows, RegSeqRat readback, ArchimedeanApproximation scale data, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3572.1, Definition 3572.2, and Definition 3572.3 seed the finite rational bracket refinement route through FiniteBracketingInterval brackets, DyadicRatCore tolerance, incoming and refined rational endpoint rows, StreamName windows, RegSeqRat readback, ArchimedeanApproximation scale data, terminal RealUp seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real order, quotient equality of real names, trichotomy, selected infinite refinement, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket endpoint visibility, dyadic tolerance routing and exactness, endpoint refinement exactness, StreamName window exposure, RegSeqRat readback, Archimedean scale handoff, terminal seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalBracketRefinementUp.

Chapter 2611

A Concrete Naming Certificate for LocatedCompactIntervalOscillationBound$^{\uparrow}$

Following Bishop and Bridges on constructive compact intervals and finite moduli, the LocatedCompactIntervalOscill packet records the finite route by which a continuous map on a located compact interval receives a displayed oscillation bound from interval nets and dyadic tolerance ledgers. It sits after the finite-net source of chapter 1808, the compact interval surface of chapter 3431, the continuous-map graph rows of chapter 130, and the tolerance-window route of chapter 2224. The packet names only the L10 finite boundedness handoff through displayed rows; it does not assert Heine–Borel compactness, choice-selected extrema, quotient-real equality, host supremum existence, or ambient Real$^{\uparrow}$ completeness.

Definition 2611.1 (Located compact interval oscillation-bound carrier)**.** *A* LocatedCompactIntervalOscillationBoun *carrier is a finite* BHist *packet*

$$B = (I, K, F, N, M, D, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the located compact interval source row, read through the located endpoint and interval-compactness interfaces before any bound request is accepted.*
2. *K is the* LocatedIntervalCompactness$^{\uparrow}$ *row recording the compact-interval admission used by the packet.*
3. *F is the* ContinuousMap$^{\uparrow}$ *graph row for the map whose interval oscillation is bounded.*
4. *N is the* LocatedClosedIntervalFiniteNet$^{\uparrow}$ *finite-net row, listing the inspected interval net and mesh scale.*
5. *M is the* OscillationModulus$^{\uparrow}$ *row converting the displayed oscillation request into a finite tolerance-window schedule.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger carrying requested radii, mesh slack, and comparison budgets for N and M.*
7. *W is the* StreamName$^{\uparrow}$ *finite-window row used to inspect interval endpoints, net points, graph reads, and oscillation witnesses.*
8. *R is the* RegSeqRat$^{\uparrow}$ *readback row exposing the rational observations consumed by the oscillation bound.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed bound, readable only after I, K, F, N, M, D, W, R have been displayed.*

10. *H records componentwise* hsame *transport for the interval, compactness, map, finite-net, modulus, tolerance, window, readback, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the located interval and map rows through the finite net and oscillation modulus into the Real-facing seal.*

12. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$, $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, K, F, N, M, D, W, R, E, H, C, P, L$. It has no coordinate for open-cover compactness, Heine–Borel subcovers, choice-selected extrema, quotient stream equality, host supremum data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompactIntervalOscillationBoundUp.LocatedCompactIntervalOscillationBoundTas`
`single carrier alignment`

Theorem 2611.2 (Located compact interval oscillation-bound NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ *carrier $B = (I, K, F, N, M, D, W, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located compact interval source, compactness row, continuous-map row, finite-net row, oscillation-modulus row, dyadic tolerance ledger, finite stream windows, rational readback, terminal Real seal, transport, replay, provenance, and local naming;*

2. *source compatibility, requiring K and F to read the same located interval source I before the finite net or modulus is consumed;*

3. *finite-net exactness, requiring N to expose the mesh and inspected net through D and W without importing an open-cover compactness theorem;*

4. *oscillation-modulus exactness, requiring M to consume the displayed continuous-map row F together with the finite net N and the same dyadic tolerance ledger D;*

5. *Real seal nonescape, requiring the bound seal E to be downstream of I, K, F, N, M, D, W, R and forbidding choice-selected extrema, quotient-real equality, host supremum data, Lean verification, and bridge checking.*

No located compact interval oscillation-bound obligation is stored outside these rows.

Proof. Project the accepted carrier. The source rows are I, K, F, so the interval, compactness admission, and map graph are fixed before any finite net or oscillation modulus can be read.

The finite-net row N then supplies the inspected mesh through the dyadic ledger D and stream windows W. Because the carrier has no open-cover row or subcover selector, the net is only the displayed finite row inherited from the located interval route.

The oscillation-modulus row M consumes the same map, net, tolerance, and window data. The readback row R exposes the rational observations used by that modulus route before the terminal seal E is available.

The structural rows H, C, P, L transport, replay, source, and name this route. Since no coordinate stores Heine–Borel compactness, selected extrema, quotient equality, host supremum data, Lean verification, or bridge evidence, those objects cannot become local NameCert obligations. □

Theorem 2611.3 (Located compact interval oscillation-bound route)**.** *Every accepted* LocatedCompactIntervalOscil *packet routes a Real-facing oscillation-bound read through*

$$\mathsf{LocatedIntervalCompactness}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longrightarrow \mathsf{OscillationModulus}^{\uparrow} \longrightarrow \mathsf{Dy}$$

The route exposes the located compact interval source, continuous-map graph row, finite-net row, oscillation-modulus row, dyadic tolerance ledger, finite observation windows, rational readback, and terminal Real seal before any bound is consumed. A consumer that omits one of those rows is outside the accepted carrier.

Proof. Read the carrier of Definition 3573.1. The first analytic coordinates bind the located interval source, compactness admission, and continuous-map graph.

The finite-net coordinate N supplies the inspected interval mesh. The oscillation-modulus coordinate M then converts the displayed continuity and net data into a finite tolerance-window schedule using D and W.

The rational readback R and terminal seal E are downstream of that schedule. Transport and replay preserve the same components, and provenance and local naming add no analytic row.

Thus every accepted bound follows the displayed route. If any row in the route is absent, the packet has no coordinate from which a Real-facing oscillation-bound read can be projected. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ consumer read factors through one located compact interval source, one ContinuousMap graph row, one LocatedClosedIntervalFiniteNet row, one OscillationModulus row, one DyadicRatCore tolerance ledger, one StreamName window, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert, and the terminal RealUp seal.

Independence witness: located-closed-interval-finite-net, located-interval-compactness, continuousmap, oscillation-modulus: no listed sibling is bijective with $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ because the packet jointly carries the compact interval source, continuous-map graph, finite-net mesh, oscillation-modulus schedule, dyadic tolerance ledger, finite windows, rational readback, Real-facing bound seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ packet is generated from a located compact interval source row, a ContinuousMap graph row, a LocatedClosedIntervalFiniteNet mesh row, an OscillationModulus tolerance-window row, DyadicRatCore budgets, StreamName windows, RegSeqRat readback, a terminal RealUp bound seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3573.1, Definition 3573.2, and Definition 3573.3 seed the finite L10 oscillation-bound packet over located compact interval, continuous-map, finite-net, modulus, dyadic, window, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, selected extrema, quotient-real equality, host supremum data, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite-net exactness, oscillation-modulus exactness, Real seal nonescape, transport replay, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocatedCompactIntervalOscillationBoundUp.

Chapter 2612

A Concrete Naming Certificate for $\mathsf{CharacterTable}^{\uparrow}$

Following Serre and Fulton–Harris on finite-group characters, the $\mathsf{CharacterTable}^{\uparrow}$ packet records a finite character table as displayed NameCert data rather than as a host matrix. It consumes the finite group source of chapter 41, the trace and class-function rows of chapter 245, the Grothendieck ledger of chapter 246, and the Schur row and column ledger of chapter 1603.

Definition 2612.1 (CharacterTable carrier sketch)**.** *A* $\mathsf{CharacterTable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, C, I, V, \Omega, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row.*
2. *C is the finite class-row ledger, listing the displayed conjugacy-class readbacks only through the group packet and not through a quotient carrier.*
3. *I is the irreducible-character row ledger, read through* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *and the representation-ring classifier rows.*
4. *V is the value-table row ledger, whose entries are displayed trace readbacks indexed by the accepted C and I rows.*
5. *Ω is the Schur-orthogonality ledger imported as finite row-normalization, column-normalization, and off-diagonal-zero rows.*
6. *H records componentwise* hsame *transport for the group, class, irreducible-character, value-table, orthogonality, provenance, and naming rows.*
7. *P is the* Pkg *provenance over the four sibling certificates named above,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}$ *row.*

The carrier contains no host character matrix, no quotient conjugacy-class object, no ambient complex inner product, and no selected list of representatives outside these rows.

Lean4 ✓ `BEDC.Derived.CharacterTableUp.CharacterTableTasteGate single carrier alignment`

Theorem 2612.2 (CharacterTable NameCert obligations)**.** *Every accepted* $\mathsf{CharacterTable}^{\uparrow}$ *packet satisfies the following five local NameCert obligations:*

1. *carrier admission exposes exactly the group, class-row, irreducible-character-row, value-table, Schur ledger, transport, provenance, and local naming rows of Definition 3574.1;*
2. *class-row exactness requires each class read to be sourced by the displayed finite group packet of chapter 41;*
3. *irreducible-row exactness requires each row name to pass through the trace and class-function surface of chapter 245 and the representation-ring classifier of chapter 246;*
4. *value-table stability requires* hsame *transport to preserve the displayed value rows together with the Schur row and column constraints of chapter 1603;*
5. *nonescape prevents a consumer from replacing the finite rows by a host character table, quotient class carrier, selected representative list, ambient matrix equality, or external representation category.*

Proof. Project the carrier rows of Definition 3574.1. The class ledger is available only through the consumed group row, and the irreducible-character ledger is available only through the character-theory and representation-ring source rows.

The value-table entries are then read as displayed trace values indexed by those two ledgers. The Schur ledger supplies the row and column normalization and off-diagonal rows that make the table stable under accepted transport.

Finally repack the displayed rows with H, P, and N. Since the carrier has no coordinate for a host matrix, quotient class carrier, representative selector, or external representation category, every accepted table read remains inside the finite NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3574.1 and Definition 3574.2 seed the finite character-table route over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, and $\mathsf{SchurOrthogonality}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close character-table computation, quotient conjugacy classes, host matrix equality, complete representation categories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, class-row exactness, irreducible-row exactness, value-table stability, and no-host-matrix boundary for the same finite packet.

Constructive story: A $\mathsf{CharacterTable}^{\uparrow}$ packet is generated from a finite group source, displayed class rows, displayed irreducible-character rows, value-table trace rows, Schur orthogonality rows, componentwise transport, package provenance, and a local NameCert row.

Chapter 2613

A Concrete Naming Certificate for $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$

The $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ packet realizes the Denjoy–Wolff-style fixed-end reading of section 9463.5 in chapter 9463 as finite NameCert data. It consumes the disk-coordinate surface of chapter 377, the hyperbolic metric-read rows of chapter 3474, the disk phase-transport route of chapter 3455, and the boundary-tangent comparison rows of chapter 3534. The packet names boundary dynamics without importing a completed ideal boundary, a classical holomorphic self-map theorem, or an observer substance.

Definition 2613.1 (DenjoyWolffBoundaryDynamics carrier)**.** *A* $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, M, T, B, R, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-map row, including the finite source observations and target observations accepted by the packet.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row used to compare the accepted disk observations.*
3. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *route row, recording how the disk observations move through displayed phase transport.*
4. *B is the boundary-read row, recording the finite boundary-facing observation used as the fixed far-end.*
5. *R is the nonexpansive-route row, recording that every accepted step is read through the metric and transport rows before the boundary far-end is inspected.*
6. *F is the fixed-boundary row, recording stability of the same boundary read under the displayed route.*
7. *H records componentwise* hsame *transport for disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, provenance, and naming rows.*
8. *C records displayed* Cont *replay from D, M, T into B, R, F and then into provenance and naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$*, the four sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host ideal-boundary quotient, a selected fixed point, a global contraction theorem, a physical horizon, or a completed disk compactification.

Lean4 (definition) `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsUp`
Lean4 ✓ `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsTasteGate single carrier alignment`

Theorem 2613.2 (DenjoyWolffBoundaryDynamics NameCert obligations)**.** *Every accepted* DenjoyWolffBoundaryD *packet satisfies the following finite obligations:*

1. *carrier admission exposes exactly the disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, transport, replay, provenance, and local naming rows of Definition 3575.1;*
2. *boundary-read exactness requires B to be sourced through the displayed* $\mathsf{UnitDisk}^{\uparrow}$ *and* $\mathsf{HyperbolicMetric}^{\uparrow}$ *rows;*
3. *nonexpansive-route exactness requires R to pass through the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *row before a boundary read is exported;*
4. *fixed-boundary stability requires F, H, C to preserve the same boundary-facing read under accepted route transport;*
5. *nonescape prevents any consumer from replacing these finite rows by a host fixed-point theorem, selected ideal point, quotient boundary equality, completed compactification, or physical observer trajectory.*

Proof. Project the carrier rows of Definition 3575.1. The disk-map, metric-read, and phase-transport rows are the only route by which the boundary read can be formed.

The nonexpansive-route row then records the displayed metric and transport discipline before the fixed-boundary row is inspected. Transport and replay preserve the same boundary-facing read because H and C have no endpoint outside the displayed rows.

Finally the provenance and naming rows repack the route as a local NameCert surface. Since the packet contains no host fixed-point theorem, selected ideal point, quotient boundary equality, completed compactification, or physical observer trajectory, those readings cannot be exported. □

Falsifiable prediction: Every accepted $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ read factors through displayed disk-map, boundary-read, nonexpansive-route, fixed-boundary, provenance, and local NameCert rows.

Independence witness: unitdisk, hyperbolic-metric, poincare-disk-phase-transport, hyperbolic-horocycle: none is bijective to $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ because this packet jointly carries disk-map observations, metric-read discipline, phase-transport routing, boundary-read data, nonexpansive-route rows, fixed-boundary stability, transport, replay, provenance, and local naming. The UnitDisk packet supplies coordinates without boundary dynamics; the HyperbolicMetric packet supplies metric reads without a fixed-boundary row; the Poincare disk phase-transport packet supplies movement without a Denjoy–Wolff far-end commitment; and the horocycle packet supplies boundary-tangent rows without nonexpansive disk-map dynamics.

The Lean follow-up must provide `Nontrivial DenjoyWolffBoundaryDynamicsUp` and `FieldFaithful DenjoyWolffBoundaryDynamicsUp` in `lean4/BEDC/Derived/DenjoyWolffBoundaryDynamicsUp/TasteGate.lean`

keeping $D, M, T, B, R, F, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3575.1 and Definition 3575.2 seed the finite boundary-dynamics route over displayed disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close the classical Denjoy–Wolff theorem, holomorphic self-map classification, ideal-boundary quotient equality, selected fixed points, physical observer trajectories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-read exactness, nonexpansive-route exactness, fixed-boundary stability, nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.

Constructive story: A $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ packet is generated from UnitDisk disk-map rows, HyperbolicMetric metric reads, PoincareDiskPhaseTransport rows, a boundary-read row, a nonexpansive-route row, a fixed-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2614

A Concrete Naming Certificate for $\mathsf{BishopCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyCriterion}^{\uparrow}$ packet records the finite criterion by which a real-facing sequence displays Cauchy control before any completion seal is consumed. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion criterion route of chapter 1476. The packet names the criterion as finite BEDC NameCert data; it is not a proof of ambient $\mathsf{Real}^{\uparrow}$ completeness, a quotient-stream equality theorem, a selected limit, or a host Cauchy completion object.

Definition 2614.1 (Bishop Cauchy criterion carrier)**.** *A* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii, comparison budgets, and slack entries used by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite source windows inspected at the displayed tolerance requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff row, assembled from the stream windows and dyadic tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *K is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *dependency row, recording that the criterion can be consumed by completion-facing packets without exporting completion itself.*
6. *H records componentwise* hsame *transport for the tolerance, window, regularity, seal, and dependency rows.*
7. *C records displayed* Cont *replay through the same finite criterion route.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $D, S, R, E, K, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2614.2 (Bishop Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier $B = (D, S, R, E, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, StreamName finite-window exposure, RegSeqRat regularity handoff, Real seal exposure, CauchyCompletionCriterion dependency exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3576.1. The analytic route is exactly the tolerance row D, the stream-window row S, the regularity handoff R, the terminal seal E, and the completion-criterion dependency K.

The dyadic and stream rows supply the finite Cauchy evidence. The regularity handoff then repacks those finite observations through $\mathsf{RegSeqRat}^{\uparrow}$ before a $\mathsf{Real}^{\uparrow}$ seal is available.

The dependency row K records the only completion-facing use of the criterion. It points to the displayed $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ surface without adding a completion object or an ambient completeness theorem.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCauchyCriterionUp.TasteGate.BishopCauchyCriterionTasteGate single carrier alignment`

Theorem 2614.3 (Bishop Cauchy criterion completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionCriterion}^{\uparrow}$$

together with H, C, P, N. Consequently the consumer receives dyadic tolerances, finite stream windows, regular rational readback, a terminal real seal, and the named completion criterion dependency, but it receives no selected limit, quotient stream, countable-choice schedule, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a completion-facing read of B. The only tolerance and observation rows are D and S, so the read must expose the dyadic budgets and stream windows before any regularity handoff can occur.

The row R supplies the regular rational readback, and E supplies the terminal real seal. Thus the real-facing part is downstream of the displayed finite Cauchy observations.

The row K is the sole completion criterion dependency. It permits the packet to be consumed by the Cauchy-completion criterion route, while the absence of quotient, selected-limit, and ambient-completeness coordinates prevents any stronger export. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyCriterion}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite windows, RegSeqRat regularity readback, a Real seal, a CauchyCompletionCriterion dependency row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 3576.1, Definition 3576.2, and Definition 3576.3 close the finite Bishop Cauchy criterion route for the L10 Real and Completion surface.

Formal verification: FormalStatus(NameCert$_{\text{BishopCauchyCriterion}\uparrow}$, theoremChecked)
Lean target: `BEDC.Derived.BishopCauchyCriterionUp.TasteGate.BishopCauchyCriterionTasteGate\_single\_carrier\_alig`
Bridge status: none
Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient Real completeness, or bridge checking.
Upgrade path: Public export or bridge work requires a standard interface for the checked criterion route.

Chapter 2615

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ packet records the finite equivalence route between a displayed Bishop Cauchy criterion and a displayed dyadic modulus schedule before any completion-facing consumer can read a real seal. It sits after the criterion packet of chapter 3576, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion criterion route of chapter 1476. Its downstream completion consumer is the witness route of chapter 3579; the packet does not export quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2615.1 (Bishop Cauchy modulus equivalence carrier)**.** *A* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, D, S, R, E, K, U, H, T, P, N)$$

with displayed rows Q for the $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *criterion, D for the dyadic modulus schedule, S for the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule, R for the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E for the terminal* $\mathsf{Real}^{\uparrow}$ *seal, K for the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *dependency, U for the criterion-to-modulus transport row, H for componentwise* hsame *transport, T for displayed* Cont *replay, P for* Pkg *provenance, and N for the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$ *row. Accepted packets compare criterion and modulus data only through these rows.*

Lean4 ✓ `BEDC.Derived.BishopCauchyModulusEquivalenceUp.BishopCauchyModulusEquivalenceTasteGate`
`single carrier alignment`

Theorem 2615.2 (Bishop Cauchy modulus equivalence NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier $M = (Q, D, S, R, E, K, U, H, T, P, N)$, the local* $\mathsf{NameCert}$ *surface consists exactly of criterion-row admission, dyadic modulus exposure, stream-window exposure, regular rational readback, Real seal exposure, completion-criterion dependency exposure, criterion-to-modulus transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3577.1. The analytic part is Q, D, S, R, E, K, U: the criterion row is read first, the dyadic modulus schedule and finite stream windows display the same Cauchy demand, and the regular rational readback reaches the Real seal only through that displayed route.

The completion dependency is K, so completion-facing use remains a dependency row rather than a selected completion object. The structural rows H, T, P, N transport, replay, source, and name exactly the same finite data, leaving no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 2615.3 (Bishop Cauchy modulus equivalence consumer route)**.** *Every consumer of an accepted* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier that claims a criterion/modulus handoff for L10 completion must factor through* $\mathsf{BishopCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, *and the downstream* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *route before the local* NameCert *row is exported.*

Proof. Begin with the criterion row Q and the dyadic modulus row D. The stream windows S and regular readback R are the only rows that connect those two presentations to the terminal seal E, so a completion consumer cannot jump from a criterion statement to a Real-facing witness.

The dependency row K points to the completion criterion, and the witness consumer reads that dependency through the displayed L10 route. The rows U, H, T, P, N then record the handoff, transport, replay, provenance, and name; no additional carrier coordinate supplies quotient equality, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ packet is generated from a BishopCauchyCriterion row, a dyadic modulus row, StreamName finite windows, RegSeqRat readback, a Real seal, a CauchyCompletionCriterion dependency, criterion-to-modulus transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3577.1, Definition 3577.2, and Definition 3577.3 seed the finite criterion/modulus equivalence route for the L10 Real and Completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exposure, dyadic modulus exposure, stream-window exposure, regular readback, Real seal non-escape, completion-criterion dependency preservation, transport stability, replay stability, provenance, and local naming.

Chapter 2616

A Concrete Naming Certificate for $\mathsf{BurnsideOrbitCount}^{\uparrow}$

Following Burnside and Pólya on finite group actions and orbit enumeration, the $\mathsf{BurnsideOrbitCount}^{\uparrow}$ packet records the finite ledger by which a displayed $\mathsf{Group}^{\uparrow}$ action on a displayed $\mathsf{FinSet}^{\uparrow}$ carrier yields an orbit-count read from fixed-point rows. It sits between the group certificate of chapter 41, the finite-set certificate of chapter 112, the character ledger of chapter 245, and the representation-ring surface of chapter 246. The packet names only the finite action and fixed-row accounting surface; it does not import quotient orbit sets, host cardinal arithmetic, complete character tables, or a representation category.

Definition 2616.1 (Burnside orbit-count carrier)**.** *A* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, X, A, F, O, M, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row for the finite group endpoint.*
2. *X is the consumed* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *row for the acted-on finite carrier.*
3. *A is the displayed action ledger, read as finite* Cont *rows from G and X to the endpoint rows of X.*
4. *F is the fixed-point table whose row for g lists exactly the displayed x-rows satisfying the action read $g \cdot x = x$ through the classifier of X.*
5. *O is the orbit ledger, listing carried orbit-name rows and the finite membership witnesses by which each displayed x-row is assigned to one orbit row.*
6. *M is the finite averaging ledger displaying the numerator rows $\sum_{g \in G} \#F_g$, the divisor row $\#G$, and the resulting orbit-count read.*
7. *H records componentwise* hsame *transport for the group, finite-set, action, fixed-point, orbit, averaging, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the group and finite-set rows through the action ledger into the fixed-point table, orbit ledger, and finite averaging row.*
9. *P is the* Pkg *provenance over* $\mathsf{BurnsideOrbitCount}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, X, A, F, O, M, H, C, P, N$. It has no coordinate for a quotient orbit carrier, a host set cardinal, a complex character table, an ambient representation ring, or a choice-selected orbit representative.

Lean4 ✓ `BEDC.Derived.BurnsideOrbitCountUp.BurnsideOrbitCountTasteGate single carrier alignment`

Theorem 2616.2 (Burnside orbit-count NameCert obligations)**.** *For every accepted* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier $B = (G, X, A, F, O, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the consumed group row, consumed finite-set row, action ledger, fixed-point table, orbit ledger, averaging row, transport, replay, provenance, and local naming;*
2. *action visibility, requiring every action read in A to expose its group source in G and its input and output rows in X;*
3. *fixed-point visibility, requiring every row of F to be witnessed by an action read in A and a displayed classifier comparison in X;*
4. *orbit-ledger stability, requiring O to transport along the same* hsame *rows that transport G, X, A, and F;*
5. *finite averaging exactness, requiring M to consume only the displayed fixed-point rows and the displayed group-size divisor before exposing the orbit-count read.*

No Burnside orbit-count obligation is stored outside these rows.

Proof. Project the accepted carrier. The rows G and X supply the finite group source and finite acted-on carrier, and A is the only row in which action endpoints are displayed.

The fixed-point table F is downstream of A and the classifier of X. Thus a fixed-point read must show both the action result and the displayed comparison with the original x-row.

The orbit ledger O is then read as finite assignment data over the same X rows. Its stability is exactly componentwise transport by H, because no separate quotient-orbit carrier is present.

The averaging row M consumes the displayed fixed-point rows and the displayed group-size divisor. The replay row C, provenance row P, and local naming row N repack this data as the local NameCert surface, so no host cardinal, character table, representation ring, or selected representative becomes an additional obligation. □

Theorem 2616.3 (Burnside fixed-row handoff)**.** *Every orbit-count read exported by an accepted* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Group}^{\uparrow},\ \mathsf{FinSet}^{\uparrow} \longrightarrow \mathsf{ActionLedger} \longrightarrow \mathsf{FixedRowTable} \longrightarrow \mathsf{OrbitLedger} \longrightarrow \mathsf{FiniteAverage}.$$

The handoff exposes the finite group row, finite-set row, action ledger, fixed-point table, orbit-ledger stability data, and finite averaging row before any orbit count is consumed. A consumer that omits one of those rows is outside the accepted carrier.

Proof. Read the carrier of Definition 3578.1. The first two coordinates bind the consumed $\mathsf{Group}^{\uparrow}$ and $\mathsf{FinSet}^{\uparrow}$ rows, and the action coordinate A records all displayed action reads.

The fixed-row table F can only be projected after those action reads are visible. Each fixed row is therefore a finite ledger row over G, X, and A, rather than a host fixed-point subset.

The orbit ledger O records the finite orbit-name assignments transported by H. The averaging ledger M then reads the fixed rows and group-size divisor to expose the orbit-count row.

Since C, P, and N only replay, package, and name the same displayed route, every exported orbit count factors through the stated handoff. If any row in the route is absent, the packet has no coordinate from which the orbit-count read can be projected. □

Falsifiable prediction: Every accepted $\mathsf{BurnsideOrbitCount}^{\uparrow}$ consumer read factors through one $\mathsf{Group}^{\uparrow}$ row, one $\mathsf{FinSet}^{\uparrow}$ row, one displayed action ledger, one fixed-point table, one orbit ledger, one finite averaging row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: group, finset, character-theory, representation-ring: no listed sibling is bijective with $\mathsf{BurnsideOrbitCount}^{\uparrow}$ because the packet jointly carries action reads, fixed-point rows, orbit assignment rows, finite averaging data, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$.

Constructive story: A $\mathsf{BurnsideOrbitCount}^{\uparrow}$ packet is generated from a consumed $\mathsf{Group}^{\uparrow}$ row, a consumed $\mathsf{FinSet}^{\uparrow}$ row, a displayed finite action ledger, a fixed-point table, an orbit assignment ledger, a finite averaging row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3578.1, Definition 3578.2, and Definition 3578.3 seed the finite Burnside orbit-count packet over displayed group, finite-set, action, fixed-row, orbit-ledger, averaging, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient orbit carriers, host cardinal arithmetic, complete character theory, representation-ring computation, selected orbit representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action visibility, fixed-point visibility, orbit-ledger stability, finite averaging exactness, transport replay, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.BurnsideOrbitCountUp.

Chapter 2617

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessWitness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ packet records the finite witness by which regular Cauchy data, finite stream windows, dyadic tolerance rows, a completeness modulus, and a terminal Real seal are read as one completion-facing surface. It sits after the regular rational sequence surface of chapter 301, the finite stream-name discipline of chapter 126, the dyadic tolerance core of chapter 291, the terminal Real-name seal of chapter 38, and the Cauchy completeness modulus packet of chapter 2051. The packet names the L10 completion witness between regular Cauchy data and the $\mathsf{Real}^{\uparrow}$ seal; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient Real completeness.

Definition 2617.1 (Cauchy completeness witness carrier)**.** *A* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, S, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row whose Cauchy data is being witnessed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the inspected stream observations used by the witness.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic budgets attached to those windows.*
4. *M is the* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *row that bounds the tail-coherence demands accepted by the witness.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D, M have all been displayed.*
6. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, completeness modulus, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-witness request through R, S, D, M, E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, M, E, H, C, P, N$ and their displayed replay path. It has no coordinate for quotient stream equality, selected representatives, host-limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessWitnessUp.CauchyCompletenessWitnessTasteGate single carrier alignment`

Theorem 2617.2 (Cauchy completeness witness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletenessW}$ *carrier $W = (R, S, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, classifier stability under componentwise* hsame*, witness-to-seal routing through the completeness modulus and terminal Real row, ledger nonescape for dyadic tolerance data, and sibling dependency routing through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Every completion-facing consumer must read the route

$$R \longmapsto S \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N$$

before treating the regular Cauchy data as sealed by a completion witness.

Proof. Project the carrier of Definition 3579.1. The accepted packet exposes only the regular readback row, finite stream windows, dyadic tolerance ledger, completeness modulus, terminal Real seal, transport, replay, provenance, and local naming rows.

Carrier admission and classifier stability are componentwise. The rows R, S, D, M, E are compared only by the displayed hsame transports in H, and the replay row C preserves the same order of inspection for any accepted completion-witness request.

The witness-to-seal route is forced by M and E. The completeness modulus bounds the tail-coherence demand before the Real seal is consumed, while the dyadic ledger D has already exhausted the tolerance data that could otherwise be mistaken for a detached limit choice.

Finally P and N record sibling dependency routing and the local certificate name. Since the carrier contains no quotient stream equality, selected representative, host-limit operator, countable-choice schedule, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the NameCert obligation surface. □

Theorem 2617.3 (Cauchy completeness witness Real seal route)**.** *Every completion-seal read of an accepted* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *packet factors through the displayed finite witness rows $R, S, D, M, E, H, C, P, N$. In particular, a* $\mathsf{Real}^{\uparrow}$ *consumer must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational data, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window evidence, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *tail-control row before it reaches the terminal seal E.*

The route exports a completion witness for displayed regular Cauchy data. It does not export quotient stream equality, a selected host limit, countable choice, trichotomy, a detached Real seal, or ambient Real completeness.

Proof. Start with a consumer that reads a completion seal. The only Real-facing terminal coordinate in the carrier is E, and E appears after the readback, finite-window, dyadic-ledger, and completeness-modulus rows.

The consumer must therefore pass through R, S, D, M. These rows provide the regular Cauchy data, the finite observations, the tolerance budget, and the tail-control witness needed before the Real seal can be named.

The support rows H, C, P, N transport, replay, source, and locally certify that same finite route. They can preserve the displayed witness, but they cannot introduce quotient equality, selected

limits, countable choice, trichotomy, detached Real seals, or ambient completeness because no such coordinate is present in the accepted packet. □

Theorem 2617.4 (Cauchy completeness witness obligation package)**.** *The carrier definition Definition 3579.1, the NameCert-obligations theorem Definition 3579.2, and the Real-seal-route theorem Definition 3579.3 together supply the obligation surface for* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*. This surface consists of carrier admission, classifier stability through componentwise* hsame*, witness-to-seal routing through the completeness modulus,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger nonescape, sibling routing through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and local* NameCert *provenance.*

Proof. Project the finite tuple $W = (R, S, D, M, E, H, C, P, N)$ from Definition 3579.1. Carrier admission is exactly the admission of these displayed rows, with no extra coordinate for quotient stream equality, selected limits, choice, trichotomy, detached Real seals, or ambient completeness.

Apply Definition 3579.2. It identifies classifier stability, componentwise hsame transport, dyadic ledger discipline, sibling dependency routing, provenance, and the local NameCert row as the NameCert surface of the packet.

Apply Definition 3579.3. It forces every completion-seal read through $R, S, D, M, E, H, C, P, N$, so the witness-to-seal obligation passes through the completeness modulus and terminal Real seal rather than through an unstated host limit. The three cited surfaces therefore exhaust the obligation package for the finite completion witness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting the regular rational readback, finite stream windows, dyadic tolerance ledger, completeness modulus, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: regseqrat, streamname, dyadicratcore, cauchy-completeness-modulus, real: no listed sibling supplies the whole witness packet $W = (R, S, D, M, E, H, C, P, N)$, because each sibling contributes only readback, windows, tolerances, tail control, or the terminal seal while this chapter binds all five analytic faces as one finite completion witness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ packet is generated from a RegSeqRat regular readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a CauchyCompletenessModulus tail-control row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$, seed)

Scope closed: Definition 3579.1, Definition 3579.2, and Definition 3579.3 seed the finite completion-witness route over regular rational readback, stream windows, dyadic tolerances, completeness modulus data, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, detached Real seals, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, witness-to-seal routing, ledger nonescape, sibling dependency routing, and Real-seal factorization for BEDC.Derived.CauchyCompletenessWitnessUp.

Chapter 2618

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ packet records the finite obligation package that a completion-facing witness must discharge before downstream Real-completion consumers may use the $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ route. It sits over the regular rational readback of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the witness carrier of chapter 3579. The chapter names only the obligation surface for that textbook completion witness; it does not assert quotient stream equality, selected host limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2618.1 (Cauchy completeness witness obligation carrier)**.** *A* $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, A, K, R, D, S, E, H, C, P, N)$$

whose rows record a displayed $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *carrier* W*, carrier-admission evidence* A*, classifier-stability evidence* K*, sibling readback rows* R, D, S*, a terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance over the witness and obligation surfaces, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}$ *row* N*. The classifier compares such carriers only by these rows and their displayed route through* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessWitnessObligationUp.CauchyCompletenessWitnessObligationTasteGate`
`single carrier alignment`

Theorem 2618.2 (Cauchy completeness witness obligation closure package)**.** *For every accepted* $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ *carrier* $O = (W, A, K, R, D, S, E, H, C, P, N)$*, carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling dependency routing, and Real-seal factorization exhaust the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *obligation surface.*

More precisely, A *admits the displayed witness carrier,* K *compares the accepted rows only through the componentwise* hsame *transports,* W *routes every completion-witness request to the terminal seal* E*,* D *accounts for the dyadic tolerance ledger before the seal is visible,* R, S, D *route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* P, N *attach the same finite route to the local package and naming certificate. No quotient stream, selected representative, host-limit operator, countable-choice schedule, trichotomy row, detached Real seal, or ambient completeness coordinate is exported.*

Proof. Project the carrier of Definition 3580.1. The displayed rows are exactly the witness carrier, admission row, classifier row, sibling readbacks, Real seal, transport, replay, provenance, and local naming certificate.

Carrier admission is the row A, while classifier stability is the row K transported through H. The replay row C can only read the same finite route, so accepted comparisons cannot introduce a second classifier surface.

The witness-to-seal route is the displayed $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ row W ending at E. The dyadic row D is already part of that route, so every tolerance ledger read is consumed before the terminal seal is available.

The sibling dependencies are the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows, together with the witness chapter of chapter 3579. The Real-seal factorization is therefore the finite route from those sibling rows through W to E, packaged by P and named by N.

Since the carrier has no coordinate for quotient streams, selected host limits, choice, trichotomy, detached Real seals, or ambient completeness, those objects cannot become obligations of the local certificate. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ consumer that reaches a $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ seal while omitting carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling dependency routing, Real-seal factorization, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cauchy-completeness-witness, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole obligation packet $O = (W, A, K, R, D, S, E, H, C, P, N)$, because each sibling contributes only witness structure, regular readback, finite windows, dyadic tolerances, or the terminal seal while this chapter binds the obligation surface itself.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ packet is generated from a CauchyCompletenessWitness carrier, carrier admission, classifier stability, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance data, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3580.1 and Definition 3580.2 close the finite obligation surface for CauchyCompletenessWitness carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling routing, Real-seal factorization, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, detached Real seals, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A checked theorem for the same $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert obligation route would raise the verification axis.

Chapter 2619

A Concrete Naming Certificate for $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$

Following Hardy–Littlewood maximal estimates and the Vitali-cover selection lemma, the $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ packet records the finite maximal-function surface for metric balls, positive radius windows, and local average rows. It sits beside the finite covering ledger of chapter 2166 and the terminal real-name boundary of chapter 38; the packet names one displayed maximal domination route without importing measure completion, countable choice, selected suprema, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2619.1 (Hardy-Littlewood maximal carrier)**.** *A* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, B, R, A, V, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric-source row. It records the carried points, the displayed distance reads, and the finite ball names inspected by the maximal request.*
2. *B is the metric-ball row. Each ball record stores a center, a positive radius witness, the membership probes read by the packet, and the carrier-local inclusion comparisons used downstream.*
3. *R is the positive-radius window row. It stores only finite radius candidates and rational comparison budgets; it has no coordinate for a selected optimizing radius.*
4. *A is the local-average row. For every displayed ball window it records the finite readback route for the corresponding average value and the rational tolerance by which that value is compared.*
5. *V is the finite Vitali-cover ledger, read through* $\mathsf{FiniteVitaliCover}^{\uparrow}$*. It records the selected disjoint subfamily, the overlap budget, and the cover cells used to dominate the requested window.*
6. *D is the domination witness row. It records the finite comparison that bounds each displayed local average by the exported maximal value read through the same radius and cover windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the finite real-name windows reached from A and D.*
8. *H records componentwise* hsame *transport for the metric source, ball windows, radius budgets, local averages, Vitali ledger, domination witness, real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay of the route*

$$\mathsf{FiniteVitaliCover}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist} \longmapsto \mathsf{HardyLittlewoodMaximal}^{\uparrow}.$$

10. *P is the* Pkg *provenance over* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $X, B, R, A, V, D, E, H, C, P, N$ and has no coordinate for a selected supremum, a countable ball family, a measure-completion object, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HardyLittlewoodMaximalUp.HardyLittlewoodMaximalTasteGate single carrier alignment`

Theorem 2619.2 (Hardy-Littlewood maximal NameCert obligations)**.** *For an accepted* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *carrier*

$$M = (X, B, R, A, V, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, metric-ball windows, positive-radius budgets, local average rows, finite Vitali-cover ledger, domination witness, real seal, transport, replay, provenance, and local naming row;*
2. *positive-radius visibility, requiring every local average request to read a displayed radius witness from R and the matching ball data from B before any average row in A is inspected;*
3. *finite Vitali-cover discipline, requiring domination to pass through the ledger V and its overlap budget before the witness row D is exported;*
4. *real-seal discipline, requiring every numerical output to pass through E after the local average and domination rows have been displayed;*
5. *ledger non-escape, requiring H, C, P, N to preserve exactly the rows X, B, R, A, V, D, E and to export no selected supremum, no countable-choice ball selection, no measure completion, no quotient function equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3581.1. The accepted packet exposes exactly the metric source, ball windows, positive-radius budgets, local average rows, finite Vitali ledger, domination witness, real seal, transport, replay, provenance, and local naming rows.

The radius and average obligations are ordered by the carrier itself. A local average read from A is meaningful only after the matching ball data from B and positive radius witness from R have been displayed, so the packet cannot replace finite windows by an optimizing host radius.

The covering step is the row V. Its $\mathsf{FiniteVitaliCover}^{\uparrow}$ ledger supplies the finite selected subfamily and overlap budget used by D, and the domination witness can only cite those displayed cells.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since there is no carrier coordinate for a selected supremum, countable-choice ball family, measure completion, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles appears in the local $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ packet is generated from a metric source row, metric-ball windows, positive-radius budgets, local average rows, a finite Vitali-cover ledger, a domination witness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3581.1 and Definition 3581.2 seed the finite maximal-function carrier and obligation surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{HardyLittlewoodMaximal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, measure completion, countable-choice ball selection, selected suprema, quotient function equality, ambient Real$^{\uparrow}$ completeness, Lebesgue differentiation, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius visibility, finite Vitali-cover discipline, real-seal discipline, and ledger non-escape for BEDC.Derived.HardyLittlewoodMaximalUp.

Chapter 2620

A Concrete Naming Certificate for $\mathsf{RationalRootInterval}^{\uparrow}$

Following Sturm and Bishop–Bridges constructive interval methods, the $\mathsf{RationalRootInterval}^{\uparrow}$ packet records a finite rational interval certificate for a displayed polynomial sign change and its bisection route. It sits after the dyadic rational carrier of chapter 291, the located interval carrier of chapter 1709, the bisection route of chapter 1711, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite interval-certificate route used by L10 Real and Completion consumers; it is not a host real equality test, quotient root set, global root selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 2620.1 (RationalRootInterval carrier)**.** *A* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite coefficient row for the inspected rational polynomial, including the displayed degree bound and the rational coefficient reads used by the certificate.*
2. *L and U are the left and right endpoint rows, carried through* $\mathsf{DyadicRatCore}^{\uparrow}$ *or rational endpoint data and then embedded into the located interval surface.*
3. *G is the sign-read ledger, recording the endpoint sign observations and every comparison row used to treat the interval as bracketing a rational root candidate.*
4. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *provenance row, recording the displayed bisection path from the original endpoint pair to the certified subinterval.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for coefficient reads, endpoint reads, sign reads, and bisection-stage observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the dyadic or rational endpoint ledgers, so every exported approximation is sourced by finite observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the coefficient row, endpoint rows, sign-read ledger, bisection provenance, stream windows, and regular rational readback have been replayed.*
8. *H records componentwise* hsame *transport for coefficient data, endpoints, sign reads, bisection provenance, stream windows, regular readback, terminal seal, replay, provenance, and local naming data.*

9. *C records displayed* Cont *replay for consumers that inspect the polynomial row, endpoint interval, sign ledger, bisection route, regular readback, or terminal seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RationalRootInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisecti}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, U, G, B, W, R, S, H, C, P, N$. It has no coordinate for host real equality, quotient root sets, global root selectors, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RationalRootIntervalUp.RationalRootIntervalTasteGate single carrier alignment`

Theorem 2620.2 (RationalRootInterval bisection seal)**.** *For every accepted* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$*-facing interval seal is exported only by the finite route*

$$A, L, U, G \longrightarrow B \longrightarrow W \longrightarrow R \longrightarrow S.$$

The coefficient row fixes the inspected rational polynomial, the dyadic or rational endpoint rows fix the displayed interval, the sign-read ledger records the endpoint comparisons, the bisection provenance supplies the certified narrowing route, the $\mathsf{StreamName}^{\uparrow}$ *windows expose finite observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row packages the regular rational readback, and* $\mathsf{Real}^{\uparrow}$ *consumes exactly that seal.*

Proof. Project the accepted carrier of Definition 3582.1. The rows A, L, U, G display the polynomial coefficients, the two endpoint carriers, and the sign-read ledger; there is no separate coordinate for a host-level polynomial root or for equality of ambient reals.

The bisection data are then forced through B. The route records the finite interval refinements and their provenance, so a consumer cannot replace the displayed path by a global root selector or by a quotient root set.

Regular readback starts only after the finite observations have been opened. The windows in W expose coefficient, endpoint, sign, and bisection-stage reads; the row R packages those reads as regular rational data; and the terminal row S is the only $\mathsf{Real}^{\uparrow}$ seal available to the packet.

Finally, H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no host real equality, quotient root set, global root selector, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, none of those principles is claimed by the bisection seal. □

Theorem 2620.3 (RationalRootInterval sign-read stability)**.** *For every accepted* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N),$$

endpoint sign observations are transported only through the displayed coefficient row A, endpoint rows L, U, sign-read ledger G, bisection provenance B, and componentwise hsame *row H. The rows C, P, N replay, source, and name that same route. No host root selector, quotient root set, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can alter the sign-read path.*

Proof. Project the carrier of Definition 3582.1. The polynomial data read by a sign observation are fixed by A, and the two endpoint observations are fixed by L and U.

Read the sign ledger through G and the finite bisection provenance through B. These rows record which endpoint comparisons are visible before the interval is narrowed, so the sign route is determined before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Transport is componentwise through H. The replay row C, provenance row P, and local naming row N preserve the same displayed observations. Since the carrier contains no host root selector, quotient root set, or ambient completeness coordinate, those objects cannot change the sign-read route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3582.1 and Definition 3582.2 seed the finite rational root interval route over coefficient rows, dyadic or rational endpoint rows, sign-read ledgers, bisection provenance, StreamName windows, RegSeqRat readback, Real seals, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient root sets, global root selectors, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-row exactness, endpoint routing, sign-read stability, bisection provenance, StreamName window exposure, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalRootIntervalUp.

Constructive story: A $\mathsf{RationalRootInterval}^{\uparrow}$ packet is generated from finite polynomial coefficient rows, dyadic or rational endpoint rows, a sign-read ledger, located-interval bisection provenance, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2621

A Concrete Naming Certificate for EnergyBiasedTriggerReliability$^{\uparrow}$

The EnergyBiasedTriggerReliability$^{\uparrow}$ packet realizes the energy-weighted trigger reliability formula of Definition 9462.1593 as finite BEDC NameCert rows. It consumes the six-coordinate codon window of chapter 2234, the boundary-tile row of chapter 3393, the coverage row of chapter 3470, and the trigger-algebra surface of chapter 3478. The packet records weight rows, partition-factor rows, blocker rows, and reliability rows; it is not a stochastic evolution model, an empirical biological likelihood claim, quotient codon equality, or an infinite trigger search.

Definition 2621.1 (Energy-biased trigger reliability carrier)**.** *An* EnergyBiasedTriggerReliability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, B, V, A, G, F, L, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the finite codon-window coordinates used by the trigger counts.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary row supplying the stop/Trp and Ile/Met trigger faces.*
3. *V is the* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row recording which boundary faces are covered by each displayed trigger family.*
4. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *trigger-algebra row carrying support closure, blocker, and threshold data.*
5. *G is the energy-weight row, recording the displayed powers of t and y assigned to finite trigger cardinality and trigger energy.*
6. *F is the partition-factor row, recording the finite $A_k(t, y)$ partition contributions that feed the reliability quotient.*
7. *L is the blocker row, recording the finite inclusion-exclusion blocker expression used in the numerator of the reliability formula.*
8. *R is the reliability row, recording the displayed value of $R(t, y)$ only after G, F, L and the trigger-algebra dependencies have been displayed.*

9. *H records componentwise* hsame *transport for source, boundary, coverage, algebra, weight, partition, blocker, and reliability rows.*
10. *C records displayed* Cont *replay through the same finite trigger-partition route.*
11. *P is the* Pkg *provenance over* EnergyBiasedTriggerReliability$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, *the source vision of Definition 9462.1593,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{EnergyBiasedTriggerReliability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, V, A, G, F, L, R, H, C, P, N$. It has no coordinate for hidden biological semantics, empirical likelihood calibration, stochastic dynamics, quotient codon equality, global support completion, or infinite trigger enumeration.

Lean4 (definition) `BEDC.Derived.EnergyBiasedTriggerReliabilityUp.EnergyBiasedTriggerReliabilityUp`
Lean4 ✓ `BEDC.Derived.EnergyBiasedTriggerReliabilityUp.EnergyBiasedTriggerReliabilityTasteGate`
`single carrier alignment`

Theorem 2621.2 (Energy-biased trigger reliability NameCert obligations)**.** *For every accepted* EnergyBiasedTriggerReliability$^{\uparrow}$ *carrier $E = (W, B, V, A, G, F, L, R, H, C, P, N)$, the local* NameCert$_{\text{EnergyBiasedTriggerRe}}$ *surface consists exactly of carrier admission, codon-window exposure, boundary-tile exposure, six-boundary coverage exposure, trigger-algebra exposure, energy-weight exposure, partition-factor exposure, blocker exposure, reliability-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3583.1. The trigger source rows are exactly W, B, V, A, so a reliability consumer first receives the finite codon window, boundary tile, coverage, and trigger-algebra surfaces.

The quantitative rows are G, F, L, R. The weight row records the powers assigned to trigger cardinality and energy, the partition row records the displayed $A_k(t, y)$ factors, the blocker row records the finite inclusion-exclusion expression, and the reliability row is available only after those rows have been exposed.

The structural rows H, C, P, N transport, replay, source, and name this same packet. Since no hidden semantics, empirical calibration, stochastic dynamics, quotient codon equality, global completion, or infinite enumeration coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 2621.3 (Energy-biased trigger partition route)**.** *Every consumer of an accepted* EnergyBiasedTriggerRelia *packet reads the reliability formula through the finite route*

$$\text{CodonSixBitWindow}^{\uparrow} \longmapsto \text{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \text{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \text{WindowCoverageTriggerA}$$

Thus the exported reliability value is a finite trigger-partition NameCert row over displayed weights, partition factors, blockers, and reliability data. The packet does not export stochastic evolution, empirical likelihood calibration, quotient codon equality, global support completion, or an infinite trigger search.

Proof. Start with the source route. The codon window row W supplies finite coordinates, the boundary row B supplies the boundary triggers, and the coverage row V records the displayed support faces.

The trigger algebra row A then binds those supports to blocker and threshold data. It is the only algebraic route into the energy-biased reliability packet.

Finally the rows G, F, L, R record weights, partition factors, blockers, and the reliability quotient. Transport, replay, provenance, and naming through H, C, P, N preserve this finite route and leave the refused external readings without export rows. □

Falsifiable prediction: Every accepted $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumer that reads $R(t, y)$ must exhibit CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WindowCoverageTriggerAlgebra, energy-weight, partition-factor, blocker, reliability, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra: $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite codon windows, boundary triggers, coverage rows, trigger algebra, energy weights, partition factors, blocker expression, reliability row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}$.

Constructive story: An $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ packet is generated from codon six-bit windows, DNA boundary tiles, six-boundary coverage rows, trigger-algebra rows, energy weights, partition factors, blockers, reliability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3583.1, Definition 3583.2, and Definition 3583.3 seed the finite trigger-partition route for energy-biased reliability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close stochastic evolution, empirical likelihood calibration, quotient codon equality, global support completion, infinite trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window exposure, boundary-tile exposure, coverage exposure, trigger-algebra exposure, energy-weight exactness, partition-factor exactness, blocker exactness, reliability non-escape, transport stability, replay stability, provenance, and local naming.

Chapter 2622

A Concrete Naming Certificate for $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ packet records the theorem-level product route saying that two displayed source series may be multiplied by the Cauchy product once the summand windows, convolution windows, absolute-convergence or majorant tail ledgers, and terminal real-name seal are all visible. It sits over the local criterion of chapter 1586, the convolution surface of chapter 409, the series-product packet of chapter 2214, the fast product route of chapter 368, the majorant discipline of chapter 399, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names the finite L10 Real seal route for Mertens' product theorem; it does not assert a quotient series product, a selected infinite sum, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2622.1 (Mertens Cauchy product theorem carrier)**.** *A* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, K, T, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the finite source-series rows exposing the inspected summand windows for the two input series.*
2. *W is the paired* $\mathsf{CauchyConvolution}^{\uparrow}$ *window row, recording the finite diagonal product cells used at each requested product index.*
3. *K is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *handoff row that binds the convolution windows to the Cauchy-product acceptance test.*
4. *T is either an* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *row for one source series or a finite majorant-tail ledger of the kind consumed by* $\mathsf{WeierstrassMTest}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchySeriesProduct}^{\uparrow}$ *or* $\mathsf{FastSeriesProduct}^{\uparrow}$ *product packet read from A, B, W, K, T.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the accepted product packet.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the product packet and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for the source rows, convolution windows, tail ledger, criterion handoff, product packet, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a product-theorem request through A, B, W, K, T, Q, R, E.*

10. *P is the* Pkg *provenance over* MertensCauchyProductTheorem$^{\uparrow}$, CauchyConvolution$^{\uparrow}$, CauchyProductCriterion$^{\uparrow}$, CauchySeriesProduct$^{\uparrow}$, FastSeriesProduct$^{\uparrow}$, AbsoluteConvergence$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{MertensCauchyProductTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, host multiplication of completed sums, selected limiting sums, countable-choice schedules, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MertensCauchyProductTheoremUp.MertensCauchyProductTheoremTasteGate single carrier alignment`

Theorem 2622.2 (Mertens Cauchy product theorem NameCert obligations)**.** *For every accepted* MertensCauchyProductTheorem$^{\uparrow}$ *carrier $M = (A, B, W, K, T, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{MertensCauchyPro}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the two finite source-series rows, paired convolution windows, tail ledger, criterion handoff, product packet, regular readback, Real seal, transport, replay, provenance, and local naming data;*

2. *convolution-window exactness, requiring W to be read through chapter 409 before any product packet Q is exposed;*

3. *tail-to-criterion handoff, requiring the absolute-convergence row or majorant-tail ledger T to feed the* CauchyProductCriterion$^{\uparrow}$ *row K through chapter 1586;*

4. *product readback routing, requiring Q to be consumed through the* CauchySeriesProduct$^{\uparrow}$ *and* FastSeriesProduct$^{\uparrow}$ *sibling surfaces before the* RegSeqRat$^{\uparrow}$ *row R is read;*

5. *L10 Real seal factorization, requiring every terminal seal read E to pass through $A, B, W, K, T, Q, R, H, C, P, N$.*

Thus a product-theorem consumer receives a finite Mertens certificate only after the source-series rows, CauchyConvolution windows, AbsoluteConvergence or majorant tail ledgers, CauchyProductCriterion handoff, RegSeqRat readback, Real seal, transport, provenance, and local NameCert rows have all been displayed.

Proof. Project the carrier of Definition 3584.1. The accepted packet exposes exactly the two source-series rows, the paired convolution window, the tail ledger, the criterion handoff, the product packet, the regular readback, the terminal Real seal, and the structural support rows.

For the convolution obligation, read W through the sibling convolution certificate. This fixes the finite diagonal product cells before any series-product packet is available.

For the tail-to-criterion obligation, read T either as an absolute-convergence witness or as a finite majorant ledger. In both cases the displayed tail control is consumed by the CauchyProductCriterion$^{\uparrow}$ handoff K, so the criterion route is not a hidden appeal to host completeness.

For the product readback obligation, route Q through the CauchySeriesProduct and FastSeriesProduct surfaces and then through the regular rational readback row R. The terminal seal E is reached only after this readback, and H, C, P, N transport, replay, source, and locally name that same route.

Since the carrier contains no quotient stream equality, host product of completed sums, selected limiting sum, countable-choice schedule, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the local NameCert surface. □

Theorem 2622.3 (Mertens Cauchy product theorem Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *packet factors through the displayed route*

$$A, B \longmapsto W \longmapsto T \longmapsto K \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The route exports the Mertens product theorem as a finite product certificate over the L10 Real seal corridor. It does not export quotient series equality, a selected product sum, countable choice, trichotomy, host real multiplication, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer that asks for the terminal product seal. The only Real-facing coordinate in the carrier is E, and E is stored after the source rows, convolution window, tail ledger, criterion handoff, product packet, and regular readback.

The consumer must therefore pass through A, B, W, T, K, Q, R. These rows display the finite source observations, the CauchyConvolution cells, the absolute-convergence or majorant tail control, the CauchyProductCriterion acceptance row, and the regular rational readback used to expose the product.

The remaining rows H, C, P, N preserve the same route under componentwise transport, replay it for accepted requests, record its package provenance, and name it locally. Because no excluded object is a carrier coordinate, the Real seal cannot bypass the finite product route. □

Theorem 2622.4 (Mertens absolute-convergence product route)**.** *Let* $M = (A, B, W, K, T, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *carrier. If the displayed tail row* T *is read as an* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *witness for one source series, then every downstream product-convergence consumer must read the route through*

$$\mathsf{AbsoluteConvergence}^{\uparrow} \longmapsto \mathsf{CauchyProductCriterion}^{\uparrow} \longmapsto (\mathsf{CauchySeriesProduct}^{\uparrow} \textit{ or } \mathsf{FastSeriesProduct}^{\uparrow}) \longmapsto \mathsf{RegSeqR}$$

before invoking $\mathsf{CauchyProductConvergence}^{\uparrow}$*. The branch therefore exports the Mertens theorem to product-convergence consumers only through the absolute-convergence handoff, the finite product criterion, the product packet, the regular rational readback, and the terminal Real seal.*

Proof. Project the carrier of Definition 3584.1. Under the stated branch hypothesis, the tail row T is not merely an unnamed tail ledger: it is displayed as an $\mathsf{AbsoluteConvergence}^{\uparrow}$ row and is therefore subject to the handoff and Real-seal nonescape routes of Definition 1700.3 and Definition 1700.4.

Feed this displayed tail row into the local route of Definition 3584.3. The route forces the consumer through W, T, K, Q, R, E; hence the absolute-convergence branch reaches the product packet only after the $\mathsf{CauchyProductCriterion}^{\uparrow}$ handoff and reaches the terminal seal only after the $\mathsf{CauchySeriesProduct}^{\uparrow}$ or $\mathsf{FastSeriesProduct}^{\uparrow}$ packet and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Finally compose this finite Mertens route with the downstream fusion theorem Definition 3608.4. A $\mathsf{CauchyProductConvergence}^{\uparrow}$ consumer that consumes the Mertens surface reads the source-window and tail-control stage before any regular rational, dyadic, stream-name, or Real-seal product data. Thus the absolute-convergence branch cannot enter product convergence except through the displayed criterion, product, readback, and Real-seal corridor. □

Falsifiable prediction: Every accepted $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source-series row, CauchyConvolution window, tail ledger,

CauchyProductCriterion handoff, product packet, RegSeqRat readback, transport row, replay row, provenance row, or local naming row refutes the packet.

Independence witness: cauchyconvolution, cauchy-product-criterion, cauchy-series-product, fast-seriesproduct, regseqrat, real: no listed sibling supplies the whole theorem packet $M = (A, B, W, K, T, Q, R, E, H, C$ because each sibling contributes only convolution cells, a criterion handoff, a product packet, readback, or terminal seal while this chapter binds them as one finite Mertens certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ packet is generated from two finite source-series rows, CauchyConvolution windows, AbsoluteConvergence or majorant tail ledgers, a CauchyProductCriterion handoff, a CauchySeriesProduct or FastSeriesProduct packet, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3584.1, Definition 3584.2, and Definition 3584.3 seed the finite product-theorem route over source-series rows, convolution windows, tail ledgers, criterion handoff, product readback, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient series equality, selected product sums, countable choice, trichotomy, host real multiplication, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convolution-window exactness, tail-to-criterion handoff, product readback routing, Real-seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.MertensCauchyP

Chapter 2623

A Concrete Naming Certificate for $\mathsf{BishopCompletionUniversalComposition}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of completions by regular Cauchy data, the $\mathsf{BishopCompletionUniversalComposition}^{\uparrow}$ packet records the finite composition surface for two Bishop completion universal-property extensions. It consumes the boundary route of chapter 1838 and the extension-plus-uniqueness ledger of chapter 397. The shared source rows are the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

The packet composes two already displayed extension ledgers. It does not select representatives of quotient streams, choose host limits, use countable choice, or appeal to ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2623.1 (Bishop completion universal composition carrier)**.** *A* $\mathsf{BishopCompletionUniversalComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (U, V, E, F, R, S, D, H, T, P, N)$$

with the following displayed rows:

1. *U is the first* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *ledger, restricted to the shared Bishop regular Cauchy completion source;*
2. *V is the second* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *ledger, whose dense-source read begins at the extension surface exported by U;*
3. *E is the first extension row exposed by U through the Bishop completion universal boundary;*
4. *F is the second extension row exposed by V after E has fixed the intermediate completion-facing map;*
5. *R is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by both extension ledgers;*
6. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source regular Cauchy name;*
7. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to align the two precision schedules;*
8. *H records the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the source route and preserved by both extensions;*
9. *T records componentwise* hsame *transport and displayed* Cont *replay for U, V, E, F, R, S, D, H;*

10. *P is the* Pkg *provenance over* BishopCompletionUniversalBoundary$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{BishopCompletionUniversalComposition}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed extension ledgers, shared source rows, transport, replay, provenance, and local naming data. It contains no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient Real$^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BishopCompletionUniversalCompositionUp.BishopCompletionUniversalCompositionTasteGa`
`single carrier alignment`

Theorem 2623.2 (Bishop completion universal composition obligations)**.** *For every accepted* BishopCompletionUniversalComposition$^{\uparrow}$ *carrier $C = (U, V, E, F, R, S, D, H, T, P, N)$, the local* NameCert$_{\text{BishopComplet}}$ *surface consists exactly of carrier admission, first-extension exactness through U, E, second-extension exactness through V, F, shared-source exactness through S, R, D, H, componentwise transport and replay through T, provenance through P, and local naming through N.*

The two extension reads must share the same StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows before the composed extension row is exported. No local obligation can be discharged by quotient representatives, selected limits, countable choice, host equality of extensions, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3585.1. The visible coordinates are exactly the two universal-property ledgers, their two extension rows, the shared source and seal rows, and the structural rows T, P, N.

Use the Bishop completion universal boundary route for the first extension. It requires the source windows, regular readback, dyadic tolerance ledger, and terminal Real seal to be displayed before the extension row E can be read.

Use the CompletionUniversalProperty obligations for both ledgers. The first ledger exports E from U over the shared source surface; the second ledger exports F from V only after the same source and seal rows have been transported and replayed through T.

The package and local naming rows then record the composed route as a single NameCert surface. Since the carrier has no quotient representative, selected-limit coordinate, choice schedule, host equality row, or ambient-completeness field, none of these can discharge an obligation. □

Theorem 2623.3 (Bishop completion universal composition route)**.** *Let $C = (U, V, E, F, R, S, D, H, T, P, N)$ be an accepted* BishopCompletionUniversalComposition$^{\uparrow}$ *carrier. Every composed completion-universal read factors through the finite route*

$$S, R, D, H \longmapsto U, E \longmapsto T \longmapsto V, F \longmapsto P, N.$$

The first leg is the Bishop completion universal boundary extension route, the second leg is the CompletionUniversalProperty extension-and-uniqueness route over the transported intermediate extension, and the handoff between the legs is only the displayed hsame *and* Cont *replay stored in T.*

Consequently the composed read exports a finite NameCert packet over the shared source rows. It does not export quotient representatives, selected limits, countable choice, host equality of maps, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the shared source rows. The StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ coordinates S, R, D, H are the only source and seal data carried by C.

Apply the Bishop completion universal boundary extension route to U and E. This gives the first extension only after the shared finite windows, regular rational readback, dyadic tolerance ledger, and terminal Real seal have been displayed.

Next apply the CompletionUniversalProperty extension-uniqueness scope to V and F. The intermediate extension is available to the second ledger only through the transported and replayed row T, so uniqueness is read as a displayed ledger comparison rather than as host function equality.

Finally package the composed route with P and N. The carrier inventory has no quotient stream, selected limit, countable-choice schedule, host equality row, or ambient completeness theorem, so the route cannot leave the finite NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3585.1, Definition 3585.2, and Definition 3585.3 seed the finite composition surface for two Bishop completion universal-property extension ledgers over shared StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected limits, countable choice, host equality of extensions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-source exactness, first-extension exactness, second-extension exactness, transport-replay, provenance, and local naming for BEDC.Derived.BishopCompletionUniversalCompositionUp

Constructive story: A $\mathsf{BishopCompletionUniversalComposition}^{\uparrow}$ packet is generated from two CompletionUniversalProperty ledgers, two extension rows, shared StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2624

A Concrete Naming Certificate for $\mathsf{FiniteModulusCompactness}^{\uparrow}$

Following Arzela–Ascoli finite-modulus compactness in constructive metric analysis, the $\mathsf{FiniteModulusCompactness}^{\uparrow}$ packet records the finite BEDC surface that turns compact metric inspection, finite-net equicontinuity, and compact uniform-continuity modulus rows into one displayed compactness certificate. It sits between the compact source of chapter 129, the finite-net family route of chapter 2362, the modulus extraction route of chapter 2292, the finite epsilon-net interface of chapter 1767, and the downstream extraction surface of chapter 3568. The packet names a finite-modulus compactness carrier; it does not assert open-cover compactness, host function-space compactness, selected subsequences, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2624.1 (Finite modulus compactness carrier)**.** *A* $\mathsf{FiniteModulusCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, E, U, A, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes only the finite compact-metric inspection data, finite-net coverage rows, metric-distance ledgers, and source* $\mathsf{NameCert}$ *provenance consumed by the certificate.*

2. *E is the* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *row. It records the finite family windows, shared center rows, radius ledgers, and equicontinuity comparison rows available at the inspected compact-net scale.*

3. *U is the* $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$ *row. It displays the pointwise modulus rows, compact-net fold, dyadic tolerance schedule, and public modulus handoff used by the compact source.*

4. *A is the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *handoff row. It records the downstream extraction interface that may consume the finite-modulus compactness certificate after K, E, U have been displayed.*

5. *W is the finite modulus-window schedule. It records the requested tolerances, the selected compact-net radii, the finite family indices inspected at those radii, and the dyadic support row that links E and U.*

6. *H records componentwise* hsame *transport for the compact source, equicontinuity, compact-uniform-modulus, Ascoli handoff, window schedule, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a finite compactness request through K, E, U, A, W, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$, $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, E, U, A, W, H, C, P, N$. It has no coordinate for open-cover compactness, a host compactness theorem, a selected subsequence, a quotient stream, a host function-space point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusCompactnessUp.FiniteModulusCompactnessTasteGate single carrier alignment`

Theorem 2624.2 (Finite modulus compactness NameCert obligations)**.** *For an accepted* FiniteModulusCompactnes *carrier $F = (K, E, U, A, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$ *surface consists exactly of compact source admission through K, finite-net equicontinuity admission through E, compact uniform-continuity modulus admission through U, Ascoli finite-modulus handoff through A, finite modulus-window scheduling through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every public finite-modulus compactness read must therefore pass through the displayed compact metric, finite-net equicontinuity, compact uniform-continuity modulus, and Ascoli handoff rows before it can export a downstream compactness certificate.

Proof. Project the accepted carrier. The mathematical rows are K, E, U, A, W, and the structural rows are H, C, P, N.

The row W is the only place where requested tolerances, compact-net radii, family indices, and dyadic support are tied together. Rows K, E, U, A supply the four named chapter interfaces read by that schedule, while H, C, P, N preserve transport, replay, provenance, and local naming.

No further coordinate appears in the carrier. Hence the local NameCert obligations are exhausted by exactly the displayed rows, and every public read must traverse those rows before it can be consumed by a compactness or Ascoli-facing packet. □

Theorem 2624.3 (Finite modulus compactness Ascoli handoff)**.** *Every* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteModulusCompactness}^{\uparrow}$ *carrier receives a finite handoff through*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{FiniteNetEquicontinuity}^{\uparrow},\quad \mathsf{CompactUniformContinuityModulus}^{\uparrow},\quad \mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}.$$

The handoff consists of the compact source row K, the finite-net equicontinuity row E, the compact uniform-continuity modulus row U, the downstream Ascoli row A, the window schedule W, and the structural rows H, C, P, N. It contains no open-cover compactness route, no selected subsequence, no host function-space compactness point, no quotient real equality, no countable-choice row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from the obligation theorem. It identifies $K, E, U, A, W, H, C, P, N$ as the complete public surface of the carrier.

The Ascoli-facing row is A, but A is not read alone. Its finite-modulus input is supplied by the compact source row K, the finite-net equicontinuity row E, the compact uniform-continuity modulus row U, and the shared window schedule W.

The transport and replay rows keep the handoff inside that finite packet. Since the excluded objects do not occur among $K, E, U, A, W, H, C, P, N$, an Ascoli consumer cannot obtain them from the $\mathsf{FiniteModulusCompactness}^{\uparrow}$ certificate. □

Theorem 2624.4 (Finite modulus compactness nonescape)**.** *No accepted* $\mathsf{FiniteModulusCompactness}^{\uparrow}$ *packet exports open-cover compactness, selected subcovers, selected subsequences, host function-space compactness, quotient stream equality, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its compactness content is exactly the finite row package*

$$K, E, U, A, W, H, C, P, N.$$

Proof. Apply Definition 3586.2. Every public read of the packet factors through the displayed tuple $K, E, U, A, W, H, C, P, N$.

The compact metric input is confined to K, the equicontinuity input to E, the compact uniform-continuity modulus input to U, the Ascoli-facing output to A, and the finite scheduling relation to W. Rows H, C, P, N only transport, replay, package, and name those displayed rows.

None of the excluded principles is a carrier coordinate or a structural row. Exporting any of them would leave the local $\mathsf{FiniteModulusCompactness}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteModulusCompactness}^{\uparrow}$ packet is generated from a CompactMetricUp source row, FiniteNetEquicontinuityUp row, CompactUniformContinuityModulusUp row, ArzelaAscoliFiniteModulusUp handoff row, finite modulus-window schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 3586.1, Definition 3586.2, Definition 3586.3, and Definition 3586.4 seed the finite BEDC compactness certificate consumed by downstream Ascoli and compact-open routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected subsequences, host function-space compactness, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window scheduling, Ascoli handoff, nonescape, and local NameCert registration for BEDC.Derived.FiniteModulusCompactnessUp.

Chapter 2625

A Concrete Naming Certificate for $\mathsf{LocatedCauchyNet}^{\uparrow}$

Following Moore–Smith, Kelley, and constructive net convergence, the $\mathsf{LocatedCauchyNet}^{\uparrow}$ packet records the located directed-tail Cauchy evidence needed by uniform-completion arguments. It sits between the Moore–Smith carrier of chapter 3403, the convergence interface of chapter 1667, the Cauchy-net surface of chapter 589, the filter-base handoff of chapter 1601, the uniform-space entourage rows of chapter 339, and the completion consumer of chapter 1473. The packet supplies finite located-Cauchy windows for the L10 directed-net completion corridor; it does not introduce quotient nets, selected limits, cofinal choice, or host completeness.

Definition 2625.1 (Located Cauchy net carrier)**.** *A* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, M, A, N, U, F, S, W, H, C, P, K)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *row naming finite directed vertices, preorder comparisons, and displayed upper-window witnesses.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *row exposing net-value windows and the finite tail ledger over D.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *row recording common-tail Cauchy evidence on the same directed surface.*
4. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *row naming the convergence-facing route that may inspect D, M, A before any Real or filter handoff is read.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer row that reads only the located Cauchy data displayed by D, M, A, N, F, S.*
6. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained from the displayed tail filters of M and the Cauchy windows of A.*
7. *S is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row used to compare Cauchy windows, located neighbourhood requests, and completion-facing refinements.*
8. *W is the located-window ledger, listing the finite neighbourhood requests, positive tolerance names, and displayed tail witnesses used to locate each Cauchy read.*
9. *H records componentwise* hsame *transport for D, M, A, N, U, F, S, W and for the structural rows.*

10. *C records displayed* Cont *replay through the same directed-tail, Cauchy, filter, uniform-space, convergence, and completion route.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *K is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $D, M, A, N, U, F, S, W, H, C, P, K$. It has no coordinate for quotienting nets, selecting a limit, selecting a cofinal subnet, invoking countable choice, or importing ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyNetUp.LocatedCauchyNetTasteGate single carrier alignment`

Theorem 2625.2 (Located Cauchy net NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *carrier*

$$L = (D, M, A, N, U, F, S, W, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$ *surface consists exactly of carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-net common-window exposure, NetConvergence handoff, UniformCompletion consumption, FilterBase handoff, UniformSpace entourage comparison, located-window evidence, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the accepted carrier of Definition 3587.1. The directed and Cauchy-facing rows are D, M, A, W, so every located read first fixes an index window, a Moore–Smith tail, common-tail Cauchy evidence, and a finite located-window witness.

The handoff rows are then N, F, S, U. The convergence route can inspect only the displayed tail evidence, the filter-base route is obtained from the same tails, the uniform-space row supplies the entourage comparisons, and the completion row consumes these data after they have been displayed.

Finally H, C, P, K transport, replay, source, and name the same packet. Since the carrier contains no additional coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2625.3 (Located Cauchy net uniform completion route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ \mathsf{MooreSmithNet}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow},\ \mathsf{UniformSpace}^{\uparrow},\ \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{UniformComplet}$$

The route supplies directed tails, located Cauchy windows, filter-base handoff, entourage comparison, and convergence-facing evidence only through the displayed carrier. It exports no quotient net, selected limit, selected cofinal subnet, countable-choice operator, or ambient completeness theorem.

Proof. Start with the directed rows D, M, A, W. The index row supplies finite upper-window witnesses, the Moore–Smith row supplies net values and tail ledgers, the Cauchy-net row supplies common-window evidence, and the located-window row binds each neighbourhood request to a displayed tail witness.

The filter-base and uniform-space rows F, S are read next. They translate the same displayed tails into filter and entourage language without creating a quotient of nets.

The convergence handoff N and completion consumer U then read only this finite route. Transport, replay, provenance, and naming through H, C, P, K preserve the same rows, so the excluded selection, quotient, choice, and ambient-completeness routes have no export coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyNet}^{\uparrow}$ packet is generated from DirectedSet, MooreSmithNet, CauchyNet, NetConvergence, UniformCompletion, FilterBase, UniformSpace, located-window, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3587.1, Definition 3587.2, and Definition 3587.3 close the located directed-tail route used by uniform-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedCauchyNetUp.LocatedCauchyNetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, selected cofinal subnets, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Public export or bridge work requires a standard interface for the checked located-net route.

Chapter 2626

A Concrete Naming Certificate for $\mathsf{AlternativeCodeReassignment}^{\uparrow}$

The $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ packet realizes the comparative genetic-code reassignment spectrum of Definition 9462.1208 as finite NameCert data. It packages the standard-code classifier, curated alternative table row, reassignment set, module ledger, sparse support-count ledger, transport, replay, provenance, and local naming rows; it does not treat the observed comparative spectrum as a kernel fact about biological mechanism or as an oracle for all translation systems.

The standard-code classifier row and codon carrier are read through chapter 3393, where the finite boundary-tile classifier rows, transport, replay, provenance, and NameCert rows are already displayed. The sparse support-count ledger and compression boundary are consumed through chapter 3558, so the reassignment packet depends only on finite support-count ledgers, transport, replay, provenance, and local NameCert rows. Its concrete sibling lattice also passes through the four-phase alphabet bridge of chapter 2260, the six-bit codon window of chapter 2234, and the support-compression surface of chapter 3558.

Definition 2626.1 (Alternative-code reassignment carrier)**.** *An* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, D, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the standard-code classifier row, naming the displayed Q_6 codon carrier and the standard assignment classifier used as the comparison base.*
2. *T is the alternative-table row, recording one curated non-standard table on the same codon carrier together with its displayed assignment labels.*
3. *D is the reassignment-set row, listing exactly those codons whose row in T differs from the standard classifier row S.*
4. *M is the module ledger over the Stop/Trp, Ile/Met, Phe/Leu, and AGA/AGG local modules of Definition 9462.1209.*
5. *Q is the support-count ledger, recording sparse codon support and multiplicity information as displayed in Definition 9462.1212.*
6. *H records componentwise* hsame *transport for the standard classifier, alternative table, reassignment set, module ledger, support-count ledger, provenance, replay, and local naming rows.*

7. *C records displayed* Cont *replay from the standard classifier and alternative-table comparison to the reassignment set, then to the module ledger and support-count ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$*, the DNA six-bit boundary tile source rows,* $\mathsf{ReassignmentSupportCompression}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$ *row.*

The carrier contains no biochemical mechanism coordinate, no exhaustive table-family oracle, no unbounded codon-table search, no quotient identifying biological labels with BEDC constructors, and no claim that reassignment frequency is a cause.

Lean4 ✓ `BEDC.Derived.AlternativeCodeReassignmentUp.AlternativeCodeReassignmentTasteGate single carrier alignment`

Theorem 2626.2 (Alternative-code reassignment NameCert obligations)**.** *For an accepted* AlternativeCodeReassigr *carrier $R = (S, T, D, M, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed comparison packet, standard-code classifier exposure through S, alternative-table exposure through T, reassignment-set exactness through D, module-ledger grouping through M, sparse support-count exposure through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3589.1. The comparison starts from the standard classifier row S and the alternative-table row T, both over the same displayed codon carrier.

The reassignment row D is then forced by finite row comparison. The module ledger M groups those displayed changes by the four local modules, and the support-count row Q records the sparse support and multiplicity data available to downstream consumers.

The structural rows H, C, P, N transport, replay, source, and name exactly this finite packet. No other coordinate exists, so the listed rows exhaust the local NameCert obligation surface. □

Theorem 2626.3 (Alternative-code reassignment spectrum nonescape)**.** *Every consumer of an accepted* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ *packet factors through the finite route*

$$S, T \longmapsto D \longmapsto M, Q \longmapsto H, C, P, N.$$

It receives a displayed standard-code comparison, alternative-table row, reassignment set, module ledger, sparse support-count ledger, transport, replay, provenance, and local NameCert surface. It does not receive biological causality, exhaustive coverage of possible genetic codes, a mechanism for translation dynamics, an unbounded codon-table oracle, or a quotient from external labels into BEDC kernel facts.

Proof. Use the obligation surface of Definition 3589.2. The only comparison inputs are S and T, and the only reassignment output is the finite row D obtained from them.

The module and support-count reads pass through M, Q. These rows preserve the comparative spectrum as displayed finite data and do not add a mechanistic or exhaustive biological premise.

Finally H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the excluded claims have no carrier coordinate, they cannot be exported by an accepted $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ read. □

Falsifiable prediction: An accepted $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ read is refuted if the standard-code comparison, curated alternative table, reassignment set, module ledger, sparse support-count

ledger, and local NameCert rows can no longer be routed through dna-four-phase-alphabet-bridge, codon-six-bit-window, reassignment-support-compression, and dna-window-six-boundary-tile sibling surfaces.

Independence witness: dna-four-phase-alphabet-bridge, codon-six-bit-window, reassignment-support-compression, dna-window-six-boundary-tile: none is bijective with $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ because the siblings respectively expose alphabet phase rows, codon-window coordinates, support-compression counts, or boundary-tile classifier rows, while the reassignment packet jointly binds standard and alternative table comparison, reassignment exactness, module grouping, sparse support accounting, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$.

Constructive story: An $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ packet is generated from a standard-code classifier, alternative table row, reassignment set, module ledger, sparse support-count ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3589.1, Definition 3589.2, and Definition 3589.3 seed the finite comparative-code reassignment spectrum surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological causality, exhaustive genetic-code coverage, translation mechanism, unbounded table search, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, standard-code comparison, alternative-table exposure, reassignment-set exactness, module-ledger grouping, sparse support-count exposure, transport preservation, and local naming for BEDC.Derived.AlternativeCodeReassignmentUp.

Chapter 2627

A Concrete Naming Certificate for $\mathsf{UniformReflectionKernel}^{\uparrow}$

Following Bourbaki's uniform structures and Kelley's separated uniform reflection, the $\mathsf{UniformReflectionKernel}^{\uparrow}$ packet records the finite indistinguishability kernel through which a uniform space becomes visible to separated and completion-facing consumers. It sits after the entourage rows of chapter 339, the displayed separated-uniform boundary of chapter 1876, the Hausdorff reflection route of **??**, and the completion-facing routes of chapter 2332 and chapter 2345. The packet names the kernel boundary itself; it is not a quotient representative set, not host equality, not a selected completion limit, not Classical choice, and not ambient Real completeness.

Definition 2627.1 (Uniform reflection kernel carrier)**.** *A* $\mathsf{UniformReflectionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, K_0, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, exposing finite entourage membership, refinement, inverse, and composition data before any separated consumer is reached.*
2. *K_0 is the indistinguishability-kernel row over the same entourage surface:*

$$x \equiv_{K_0} y \quad \Longleftrightarrow \quad \textit{every displayed entourage row of } U \textit{ admits } (x, y).$$

3. *S is the separated handoff row, read as either the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *boundary or the* $\mathsf{HausdorffReflection}^{\uparrow}$ *boundary according to the downstream consumer.*
4. *H records componentwise* hsame *transport for the uniform row, kernel row, separated or Hausdorff handoff, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an entourage request through U, then K_0, then the separated or Hausdorff handoff S.*
6. *P is the* Pkg *provenance over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient representatives, host equality, selected completion limits, Classical choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformReflectionKernelUp.UniformReflectionKernelTasteGate single carrier alignment`

Theorem 2627.2 (Uniform reflection separated handoff)**.** *Every separated or completion-facing consumer of an accepted* $\mathsf{UniformReflectionKernel}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto K_0 \longmapsto \mathsf{SeparatedUniformSpace}^{\uparrow}, \mathsf{HausdorffReflection}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}.$$

The consumer first reads the displayed entourage rows, then the indistinguishability-kernel row, then the separated-uniform or Hausdorff-reflection boundary, and only then any completion-facing row. It cannot bypass the kernel through quotient representatives, host equality, selected limits, Classical choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3590.1. The uniform source is row U, and the kernel row K_0 is attached to the same displayed entourage data.

The separated handoff is row S. By construction, S is available only after U and K_0 have been replayed, so a $\mathsf{SeparatedUniformSpace}^{\uparrow}$ or $\mathsf{HausdorffReflection}^{\uparrow}$ consumer receives the finite kernel boundary rather than a host quotient.

A completion-facing consumer can then read the $\mathsf{UniformCompletion}^{\uparrow}$ route only through that separated or Hausdorff boundary. The transport, replay, provenance, and local NameCert rows preserve this order, while the excluded quotient, representative, selector, choice, host-equality, and ambient-completeness objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3590.1 and Definition 3590.2 seed the finite route from displayed uniform entourages through the indistinguishability kernel into separated, Hausdorff-reflection, and uniform-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host equality, selected limits, Classical choice, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exposure, kernel exactness, separated handoff, Hausdorff reflection handoff, completion nonescape, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{UniformReflectionKernel}^{\uparrow}$ packet is generated from UniformSpace entourage rows, an indistinguishability-kernel row, SeparatedUniformSpace or HausdorffReflection handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2628

A Concrete Naming Certificate for $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$

The $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ packet realizes the focused phase-ledger and boundary-transport reading of chapter 9463 as finite NameCert rows. It consumes the horocycle carrier of chapter 3534, the metric-read surface of chapter 3474, the boundary-fixed transport route of chapter 3500, the disk-coordinate carrier of chapter 377, the flow-style route of chapter 195, and the Poincare disk representation principles of section 9463.3 and section 9463.4. The packet is a BEDC-native phase-ledger boundary row; it is not a host hyperbolic surface, not an unlogged observer trajectory, not a global clock, and not a physical cosmology.

Falsifiable prediction: Every accepted $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ read factors through displayed phase-ledger, disk-coordinate, hyperbolic-metric, horocycle, boundary-fixed transport, flow, provenance, replay, and local NameCert rows before it exports a boundary-transport reading.

Independence witness: hyperbolic-horocycle, hyperbolic-metric, poincare-boundary-fixed-transport, unitdisk, dynsystem: the horocycle packet records boundary-tangent incidence without focused phase-ledger provenance; the metric packet records distance readback without ledger phase; the boundary-fixed transport packet records a preserved boundary anchor without horocycle phase accounting; the UnitDisk packet supplies coordinates without phase-ledger replay; and the DynSystem packet records flow-style evolution without hyperbolic boundary tangency. None is bijective to $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$.

Definition 2628.1 (Phase-ledger horocycle carrier)**.** *A* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, D, M, Q, T, Y, H, C, P, N)$$

with the following displayed rows:

1. *F is the focused phase-ledger row, carrying the source* Hist *chain, visible phase readout, and provenance cells of the observer-expansion reading.*
2. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row through which the ledger readout is placed on a finite disk surface.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row over the same displayed disk observations.*
4. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *boundary-tangent row, read only after D and M have been displayed.*

5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row preserving the boundary far-end reading along admissible disk transport.*
6. *Y is the* $\mathsf{DynSystem}^{\uparrow}$ *flow-style ledger row that replays local record expansion without introducing a global clock.*
7. *H records componentwise* hsame *transport for the phase ledger, disk, metric, horocycle, boundary transport, flow, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the focused ledger through disk placement, metric readback, horocycle incidence, boundary-fixed transport, and flow-style expansion.*
9. *P is the* Pkg *provenance over* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$ *row.*

The carrier exports no host disk model, global universe digest, synchronized radial clock, physical observer path, selected boundary representative, quotient ideal-boundary equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PhaseLedgerHorocycleUp.PhaseLedgerHorocycleTasteGate single carrier alignment`

Theorem 2628.2 (Phase-ledger horocycle NameCert obligations)**.** *For every accepted* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *carrier $L = (F, D, M, Q, T, Y, H, C, P, N)$, the local NameCert surface consists exactly of focused phase-ledger admission, disk-coordinate placement, hyperbolic-metric readback, horocycle boundary-tangency exposure, boundary-fixed transport, flow-style replay, componentwise transport, displayed continuation, provenance, and local naming.*

Every boundary-transport read from the packet factors through

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicHorocycle}^{\uparrow} \longmapsto \mathsf{PoincareBoundaryFixedTransport}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow}$$

together with the focused phase-ledger row and the structural BHist, hsame, Cont, Pkg, *and* NameCert *rows. The Lean TasteGate follow-up must include 'Nontrivial PhaseLedgerHorocycleUp' and 'Field-Faithful PhaseLedgerHorocycleUp' instances for the same carrier surface.*

Proof. Project the accepted carrier. The focused phase-ledger row F supplies the source history chain and phase readout, while D places that readout on a displayed disk surface.

The metric and horocycle rows are then forced in order. Row M reads the hyperbolic metric over the displayed disk data, and row Q exposes the horocycle boundary-tangency surface only after D and M have been replayed.

Boundary transport and flow-style replay are the last public geometric rows. Row T preserves the boundary far-end reading, and row Y records local expansion as a finite flow-style ledger rather than as a global clock.

The structural rows H, C, P, N preserve and name that route. Since the carrier contains no host disk model, universe digest, synchronized radial clock, physical observer path, selected boundary representative, or quotient ideal-boundary equality, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$, seed)

Scope closed: Definition 3591.1 and Definition 3591.2 seed the focused phase-ledger route through disk coordinates, hyperbolic metric readback, horocycle incidence, boundary-fixed transport, flow-style replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperbolic geometry, global clocks, physical cosmology, quotient ideal-boundary equality, selected boundary representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, phase-ledger exactness, disk placement, metric readback, horocycle incidence, boundary-fixed transport, flow replay, TasteGate nontriviality, FieldFaithful coverage, and local naming.

Constructive story: A PhaseLedgerHorocycle$^{\uparrow}$ packet is generated from a focused phase-ledger row, UnitDisk coordinates, HyperbolicMetric readback, HyperbolicHorocycle incidence, PoincareBoundaryFixedTransport, DynSystem flow-style replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2629

A Concrete Naming Certificate for $\mathsf{MooreSpace}^{\uparrow}$

Following R. L. Moore and the standard developable-space formulation, the $\mathsf{MooreSpace}^{\uparrow}$ packet records a topology equipped with a countable development as finite BEDC carrier data. It sits beside the topology carrier of chapter 88, the metric-space distance surface of chapter 57, and the entourage-style neighbour rows of chapter 2334. The point of the packet is not to infer a metric or a cover-selection theorem; it is to expose the neighbourhood classifiers, development rows, star-refinement witnesses, transport, replay, provenance, and local naming rows that make the Moore-space name usable by downstream topology consumers.

Definition 2629.1 (Moore space carrier)**.** *A* $\mathsf{MooreSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, N, D, S, H, C, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *carrier row, exposing the point carrier, finite open-row classifier, finite-meet row, arbitrary-union interface as recorded BEDC cover data, and membership transport surface.*
2. *N is the neighbourhood classifier row. It records when a displayed open row surrounds a point and when one neighbourhood row refines another around the same point.*
3. *D is the countable development row, storing a finite name for each development level together with the pointwise neighbourhood request that the level is meant to answer.*
4. *S is the star-refinement witness row. For each accepted point-neighbourhood request, it records a development level whose displayed star around the point refines the requested neighbourhood row.*
5. *H records componentwise* hsame *transport for the topology, neighbourhood, development, star-refinement, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a point-neighbourhood request through T, then N, then the development row D, then the star-refinement row S.*
7. *P is the* Pkg *provenance over* $\mathsf{MooreSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a metric selector, paracompact refinement, choice-selected open cover, open-cover compactness theorem, or host topological equality.

Theorem 2629.2 (Moore space NameCert obligations)**.** *For every accepted* $\mathsf{MooreSpace}^{\uparrow}$ *carrier* $M = (T, N, D, S, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, topology exposure, neighbourhood classification, countable-development rows, star-refinement witnesses, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every Moore-space consumer reads neighbourhood information through the finite route

$$\mathsf{Topology}^{\uparrow} \longmapsto N \longmapsto D \longmapsto S \longmapsto \mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}.$$

The route may compare with metric and quasi-uniform neighbours only through the displayed $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *provenance rows; it exports no metrization theorem, no paracompactness theorem, no choice-selected cover, and no open-cover compactness conclusion.*

Proof. Project the accepted packet of Definition 3592.1. The topology row T supplies the open-row carrier and membership surface, while N is the only row that may classify neighbourhoods around a displayed point.

The development data is then forced through D and S. Row D lists the countable development levels, and row S records the local star-refinement witness that answers a point-neighbourhood request.

The structural rows H, C, P, L preserve, replay, cite, and name that route. Since the carrier contains no metric selector, paracompact refinement, choice-selected cover, open-cover compactness theorem, or host equality row, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.MooreSpaceUp.TasteGate.MooreSpaceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3592.1 and Definition 3592.2 seed the finite developable-topology packet over topology rows, neighbourhood classifiers, countable development rows, star-refinement witnesses, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MooreSpaceUp.TasteGate.MooreSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close metrization, paracompactness, choice-selected covers, open-cover compactness, host topological equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology exposure, neighbourhood classification, development-level indexing, star-refinement exactness, componentwise transport, displayed replay, provenance, and local naming.

Constructive story: A $\mathsf{MooreSpace}^{\uparrow}$ packet is generated from a topology carrier, neighbourhood classifier, countable development rows, star-refinement witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2630

A Concrete Naming Certificate for $\mathsf{VariationMeasure}^{\uparrow}$

Following the finite-partition side of Jordan bounded variation as treated by Rudin and Folland, the $\mathsf{VariationMeasure}^{\uparrow}$ packet records the ledger that a bounded-variation consumer may read before any passage to classical measure theory. It sits over the bounded-variation surface of chapter 398, the Jordan signed-variation split of chapter 2069, the real-name seal of chapter 38, the regular rational stream surface of chapter 301, and the dyadic observation rows of chapter 291. The packet names only displayed finite partition ledgers, finite edge sums, common refinements, and their visible transport rows.

Definition 2630.1 (VariationMeasure finite-partition carrier)**.** *A* $\mathsf{VariationMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, J, \Pi, E, X, D, V, A, R, H, C, P, N)$$

with the following visible coordinates:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet and its finite partition-variation ledger.*
2. *J is the optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed ledger read on the same displayed partition surface.*
3. *Π is a displayed ordered finite partition with endpoint and edge rows.*
4. *E is the finite edge ledger induced by adjacent endpoints of Π.*
5. *X records endpoint* $\mathsf{Real}^{\uparrow}$ *rows or their* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name representatives.*
6. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations used to compare adjacent endpoint values.*
7. *V is the finite dyadic variation-sum ledger on Π.*
8. *A records finite additivity rows for displayed concatenations and subpartition blocks.*
9. *R records common-refinement rows comparing transported partitions, edge ledgers, and finite sums.*
10. *H records componentwise* hsame *transport for the bounded-variation source, Jordan split, partition, endpoint, dyadic, finite-sum, additivity, refinement, package, and naming rows.*
11. *C records displayed* Cont *replay from a bounded-variation request to endpoints, edge observations, finite sums, refinement comparison, and consumer exposure.*

12. *P is the* Pkg *provenance over* $\mathsf{VariationMeasure}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, \Pi, E, X, D, V, A, R, H, C, P, N$. It has no coordinate for a sigma-additive measure, a host supremum over all partitions, quotient function equality, selected limits, ambient Real completeness, hidden partition families, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationMeasureUp.VariationMeasureTasteGate single carrier alignment`

Theorem 2630.2 (VariationMeasure finite partition additivity)**.** *Let $M = (B, J, \Pi, E, X, D, V, A, R, H, C, P, N)$ be an accepted* $\mathsf{VariationMeasure}^{\uparrow}$ *carrier. If Π is displayed as the concatenation of finite subpartition blocks $\Pi_0, \ldots, \Pi_k$ whose shared endpoints are transported by the visible* hsame *and* Cont *rows, then the public variation-measure read on Π is the finite sum of the block reads recorded in A. Each block read uses only the source* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger, the optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed ledger, endpoint rows from* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, and the local* Pkg *and* $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$ *provenance.*

The additivity statement is restricted to the displayed finite partition. It exports no sigma-additive measure, host supremum over all partitions, quotient function equality, selected limit, or ambient Real-completeness principle.

Proof. Project the carrier of Definition 3593.1. The rows available to a consumer are the source bounded-variation packet, the optional Jordan signed ledger, the displayed partition, endpoint names, dyadic edge observations, finite variation sum, additivity rows, refinement rows, and visible provenance.

For each displayed block Π_i, read its endpoint rows through $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ and its adjacent-edge comparisons through $\mathsf{DyadicRatCore}^{\uparrow}$. The source bounded-variation ledger supplies the finite edge-sum surface, while the Jordan ledger, when present, supplies the signed split on the same finite edge list.

The additivity row A concatenates those finite block sums inside the same partition ledger. Shared endpoints are identified only by the displayed hsame and Cont rows, so no unrecorded endpoint or selected limit is introduced.

Finally apply the no-escape clause built into the carrier. Since every summand is a displayed finite dyadic row and the packet has no coordinate for sigma-additivity, host suprema, quotient function equality, selected limits, or ambient Real completeness, the result is a finite-partition additivity certificate rather than a measure-theoretic extension. □

Theorem 2630.3 (VariationMeasure common-refinement classifier stability)**.** *Let M and M' be accepted* $\mathsf{VariationMeasure}^{\uparrow}$ *packets. Suppose their source* $\mathsf{BoundedVariation}^{\uparrow}$ *rows, optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *rows, endpoint* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *names,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, finite variation-sum ledgers, additivity rows,* hsame *rows,* Cont *rows,* Pkg *provenance, and local* NameCert *rows agree after transport to a displayed common finite refinement. Then the local* $\mathsf{VariationMeasure}^{\uparrow}$ *classifier identifies M and M' on that refinement, and every public finite-partition read factors through the same transported refinement ledger.*

This classifier stability uses no sigma-additive measure, host supremum over all partitions, quotient function equality, selected limit, ambient Real completeness, hidden partition family, or bridge row.

Proof. Start from the displayed common finite refinement. Its rows provide the only comparison surface for the two packets: transported source bounded-variation data, optional Jordan signed ledgers, endpoint names, dyadic observations, finite sums, additivity rows, and provenance.

Use Definition 3593.2 on each packet after transport to the common refinement. Both public reads are finite sums over the same displayed block and edge rows, and the shared endpoints are handled by the same hsame and Cont routes.

Repack these componentwise comparisons into the local classifier data. Because the carrier has no additional coordinate beyond $B, J, \Pi, E, X, D, V, A, R, H, C, P, N$, the common-refinement comparison exhausts the classifier.

The excluded objects cannot distinguish the packets. None is present among the carrier rows, and none is generated by the finite additivity or transport steps. □

Theorem 2630.4 (VariationMeasure bounded-variation dependency route)**.** *Every* $\mathsf{VariationMeasure}^{\uparrow}$ *public read factors through a displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *partition ledger and, when a signed split is requested, through the displayed* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *finite signed ledger. Endpoint values are read only through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, and adjacent-edge variation rows are read only through* $\mathsf{DyadicRatCore}^{\uparrow}$*. The packet exposes those sibling rows as finite additivity and common-refinement data with visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Project the source row B from Definition 3593.1. By the bounded-variation chapter, the available variation data are finite partition ledgers, endpoint rows, dyadic edge observations, finite sums, refinement rows, and visible provenance.

If the signed row J is present, read it through the Jordan variation decomposition chapter. That row only splits the same displayed edge ledger into positive and negative finite ledgers and reconstructs the displayed endpoint-difference row.

Endpoint rows remain on the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces, while edge comparisons remain on $\mathsf{DyadicRatCore}^{\uparrow}$. The $\mathsf{VariationMeasure}^{\uparrow}$ carrier packages these rows into finite additivity and common-refinement certificates without adding a host measure object. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationMeasure}^{\uparrow}$ packet is generated from a bounded-variation source row, an optional Jordan signed ledger, a displayed finite partition, endpoint real or regular-rational names, dyadic edge observations, finite variation sums, additivity rows, common-refinement transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3593.1, Definition 3593.2, Definition 3593.3, and Definition 3593.4 seed the finite-partition variation-measure packet over $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-additive measure, host supremum over all partitions, quotient function equality, selected limits, ambient Real completeness, hidden partition families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition additivity, common-refinement classifier stability, bounded-variation dependency routing, transport, provenance, and no-escape for BEDC.Derived.VariationMeasureUp.

Chapter 2631

A Concrete Naming Certificate for $\mathsf{CauchyContinuityCalibration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ packet records the finite comparison by which Cauchy-name continuity data is calibrated against the tolerance-only uniform-modulus interface. It sits between the finite transformer surface of chapter 1401, the pointwise Cauchy-modulus schedule of chapter 1486, the explicit uniform-modulus packet of chapter 2228, the stream-name and regular-rational readback layers of chapter 126 and chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite calibration surface, not a host function on reals, quotient stream equality, compactness, or an ambient completeness theorem.

Definition 2631.1 (CauchyContinuityCalibration carrier)**.** *A* $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer row that reads source Cauchy-name windows and produces the target Cauchy-name windows visible to the consumer.*
2. *M is the* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *row that sends each requested target tolerance to the finite source-window threshold used by S.*
3. *U is the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *comparison row, read only after the Cauchy tolerance route has displayed the same target tolerance and source radius as finite data.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger containing the source windows inspected by M and the target windows produced by S.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source and target regular rational names exposed on W.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested target tolerance, the source threshold supplied by M, and the comparison tolerance consumed by U.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the calibrated target readback after the stream, regular-rational, and dyadic rows have been replayed.*
8. *H records componentwise* hsame *transport for S, M, U, W, R, D, E and for the support rows.*

9. *C records displayed* Cont *replay from a target tolerance request through M, the source windows W, the transformer row S, the readbacks R, the dyadic ledger D, the uniform comparison U, and the seal row E.*

10. *P is the* Pkg *provenance over* CauchyContinuityCalibration$^{\uparrow}$, CauchyContinuousMap$^{\uparrow}$, CauchyModulusOfContinuity UniformContinuityModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyContinuityCalibration}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying these rows to a common Cauchy-name transformer, Cauchy-modulus schedule, uniform-modulus comparison, stream-window, regular-readback, dyadic-tolerance, and real-seal surface.

Lean4 ✓ `BEDC.Derived.CauchyContinuityCalibrationUp.CauchyContinuityCalibrationTasteGate single carrier alignment`

Theorem 2631.2 (CauchyContinuityCalibration NameCert obligations)**.** *For every accepted* CauchyContinuityCali *carrier $K = (S, M, U, W, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyContinuityCalibration}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the displayed Cauchy transformer, Cauchy-modulus schedule, uniform-modulus comparison, stream windows, regular-rational readbacks, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *Cauchy-route exactness, requiring each target tolerance to pass through M before any source window in W is consumed by S;*

3. *readback accountability, requiring the source-window and target-window reads to factor through W, R, D before the seal row E or the comparison row U is available;*

4. *uniform-comparison compatibility, requiring U to read only the tolerance and radius already displayed by M and D;*

5. *componentwise transport and replay stability under H and C for the same displayed rows;*

6. *ledger refusal, stating that host real functions, quotient streams, compactness selectors, selected limits, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 3594.1. The projection exposes the transformer row S, the Cauchy-modulus row M, the uniform comparison U, the stream windows W, the readbacks R, the dyadic ledger D, the seal E, and the support rows H, C, P, N.

The Cauchy route starts with a target tolerance. The schedule M turns that request into a source-window threshold, and only then may the transformer row S inspect the windows in W and expose the target readback through R and D.

The uniform comparison is downstream of the same route. Since U has no independent tolerance coordinate in the carrier, it can read only the tolerance and source radius already displayed by M and D.

Transport and replay preserve the row order. The hsame and Cont coordinates move and replay the same finite packet, while P and N only source and name it. The listed rows exhaust the carrier, so no host function, quotient stream, compactness selector, selected limit, or ambient completeness datum can be exported. □

Theorem 2631.3 (Cauchy continuity to uniform modulus bridge)**.** *Every accepted* $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ *carrier* $K = (S, M, U, W, R, D, E, H, C, P, N)$ *routes a target tolerance first through the Cauchy-modulus row* M*, reads the required source windows through* S, W, R, D*, and then compares the induced tolerance-only source-radius row with the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *row* U*. The resulting bridge exports a finite uniform-modulus consumer boundary while retaining the Cauchy-name transformer and real-seal rows as the only source of target readback evidence.*

Proof. Begin with the target tolerance stored in the dyadic ledger D. The row M supplies the source-window threshold demanded by the Cauchy-name route, so the source observations are read through W before S can produce target windows.

Readback then passes through R and returns to D for the tolerance comparison. This isolates the tolerance-only datum induced by the Cauchy route: a requested target tolerance together with the displayed source threshold that was sufficient for the transformer row.

Compare that induced row with U. The uniform-modulus consumer sees the same target tolerance and source radius, but it does not receive the Cauchy-name windows except through the already displayed provenance of M, S, W, R, D.

Finally replay C and transport through H. The seal row E remains downstream of the Cauchy readback, while P, N bind the bridge as a finite NameCert surface. Thus the bridge connects Cauchy-name continuity to tolerance-only uniform-modulus consumers without adding a host function, quotient stream, compactness selector, selected limit, or completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ packet is generated from a CauchyContinuousMap transformer row, a CauchyModulusOfContinuity schedule, a UniformContinuityModulus comparison row, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3594.1, Definition 3594.2, and Definition 3594.3 seed the finite calibration route from Cauchy-name continuity data to tolerance-only uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real functions, quotient stream equality, compactness, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-route exactness, readback accountability, uniform-comparison compatibility, transport and replay stability, and field faithfulness for BEDC.Derived.CauchyContinuityCalibrationUp.

Chapter 2632

A Concrete Naming Certificate for MetaCICLocalDiamondWitness$^{\uparrow}$

The MetaCICLocalDiamondWitness$^{\uparrow}$ packet realizes the local-peak witness surface of chapter 9378 as finite BEDC confluence data. It sits beside the confluence audit packet of chapter 735, the critical-path carrier of chapter 765, the subject-reduction discharge socket of chapter 719, and the typed reduction trace of chapter 2330. The packet names local peaks and candidate joins; it does not promote the conditional substitution boundary into a closed MetaCIC confluence theorem.

Falsifiable prediction: Every accepted local-diamond witness has peak, left-step, right-step, candidate-join, closedness, transport, continuation, provenance, and naming rows.

Independence witness: metacic-confluence-audit-packet, metacic-critical-path, subject-reduction-discharge-socket, metacic-subject-reduction-trace: the confluence audit packet records the global boundary without a per-peak witness carrier; the critical-path packet records priority and dependency pressure without candidate joins; the subject-reduction discharge socket records typing-side hypotheses without confluence joins; and the subject-reduction trace records reduction evidence without a local-diamond peak certificate. None is bijective to MetaCICLocalDiamondWitness$^{\uparrow}$.

Definition 2632.1 (MetaCIC local-diamond witness carrier)**.** *A* MetaCICLocalDiamondWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, L, R, J, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the peak row, naming the source term and the two outgoing MetaCIC reduction requests that form the local peak.*
2. *L is the left-step row, recording the displayed one-step or parallel-step evidence for the left branch.*
3. *R is the right-step row, recording the displayed one-step or parallel-step evidence for the right branch.*
4. *J is the candidate-join row, listing the proposed common reduct together with the two displayed route requests into it.*
5. *B is the closedness and boundary row, recording whether the witness is closed, normal-source, atom-source, or substitution-boundary conditional according to the confluence audit map.*

6. *H records componentwise* hsame *transport for the peak, branch steps, candidate join, closedness boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the peak through the two branch rows, then the candidate join, then the closedness boundary.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$*, the MetaCIC confluence audit map,* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{TypedExampleReductionTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$ *row.*

The carrier exports no arbitrary closed Church–Rosser theorem, no proof of BetaParallelSubstitutionCompatibility*, no proof of* BetaParallelDiamondClosed*, no subject-reduction theorem, and no host quotient equality of terms.*

Lean4 ✓ `BEDC.Derived.MetaCICLocalDiamondWitnessUp.MetaCICLocalDiamondWitnessTasteGate single carrier alignment`

Theorem 2632.2 (MetaCIC local-diamond witness NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ *carrier $W = (K, L, R, J, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$ *surface consists exactly of peak admission, left-step evidence, right-step evidence, candidate-join evidence, closedness and boundary classification, componentwise transport, displayed continuation replay, provenance, and local naming.*

The Lean TasteGate follow-up must provide Nontrivial *and* FieldFaithful *instances for the same carrier surface, so the witness cannot be collapsed to an empty marker and cannot hide a missing peak, branch, join, boundary, transport, replay, provenance, or naming coordinate.*

Proof. Project the carrier of Definition 3595.1. The peak row K names the source and the two outgoing reduction requests, while L and R record the displayed branch evidence.

The join and boundary rows are then read in order. Row J supplies the candidate common reduct and the route requests into it; row B classifies the witness as closed, normal-source, atom-source, or substitution-boundary conditional.

The structural rows H, C, P, N preserve and name that local peak route. Since the carrier contains no arbitrary closed Church–Rosser theorem, no substitution-compatibility proof, no closed parallel-diamond proof, no subject-reduction theorem, and no quotient equality of terms, the listed rows exhaust the NameCert obligations. □

Theorem 2632.3 (MetaCIC local-diamond candidate-join exactness)**.** *For every accepted* MetaCICLocalDiamondW *carrier $W = (K, L, R, J, B, H, C, P, N)$, every candidate-join row J is read only after the peak row K, left-step row L, right-step row R, and boundary row B have been displayed through the same local-diamond witness packet. The row J exports a candidate common reduct and the two route requests into it, but it exports no closed Church–Rosser theorem, no substitution-compatibility proof, no closed parallel-diamond proof, no subject-reduction theorem, and no quotient equality of terms.*

Proof. Project the carrier of Definition 3595.1. The NameCert obligations theorem Definition 3595.2 lists peak admission, left-step evidence, right-step evidence, candidate-join evidence, and closedness and boundary classification as the visible local-diamond rows.

Reading J therefore occurs inside that listed route. The peak row K fixes the source and outgoing requests; the branch rows L and R fix the displayed left and right steps; and the boundary row B records whether the witness is closed, normal-source, atom-source, or substitution-boundary conditional before the join row is consumed as local evidence.

The structural rows H, C, P, N only transport, replay, record provenance for, and locally name the same packet. Since the carrier has no coordinate for a closed Church–Rosser theorem, substitution compatibility, a closed parallel diamond, subject reduction, or quotient equality, the candidate join cannot export any of those objects. □

Theorem 2632.4 (MetaCIC local-diamond closedness-boundary classification)**.** *For every accepted* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ *carrier* $W = (K, L, R, J, B, H, C, P, N)$*, the boundary row* B *must be displayed before any consumer reads the candidate-join row* J*. The displayed row classifies the witness as closed, normal-source, atom-source, or substitution-boundary conditional, and the later join read may only be interpreted inside that classification.*

Consequently a local-diamond witness cannot turn a candidate join into arbitrary closed Church–Rosser, BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, subject reduction, or quotient equality of terms.*

Proof. Project the carrier of Definition 3595.1. The carrier lists B as the closedness and boundary row, while Definition 3595.3 already requires J to be read only after the peak, branch, and boundary rows have been displayed.

Read B before J. Its alternatives are closed, normal-source, atom-source, and substitution-boundary conditional, so the candidate common reduct in J is consumed only under the exposed boundary case.

The structural rows H, C, P, N transport, replay, source, and name the same classification. Since no row contains a closed Church–Rosser theorem, substitution compatibility, a closed parallel diamond, subject reduction, or quotient equality of terms, the candidate join remains a classified local witness rather than a global confluence claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3595.1 and Definition 3595.2 seed the finite local-diamond witness packet over peaks, branch steps, candidate joins, closedness boundaries, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary closed Church–Rosser, BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, subject reduction, quotient equality of terms, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, peak exposure, left-step and right-step evidence, candidate-join exactness, boundary classification, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ packet is generated from a local peak row, left-step row, right-step row, candidate-join row, closedness and boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2633

A Concrete Naming Certificate for $\mathsf{TangentCone}^{\uparrow}$

Following Bouligand and Rockafellar, the $\mathsf{TangentCone}^{\uparrow}$ packet names the finite feasible-direction surface of a convex set at a displayed base point. It sits after the convex carrier of chapter 208, the affine ambient row of chapter 206, the vector-space operations of chapter 47, the metric window route of chapter 57, and the terminal real-name interface of chapter 38. The packet records finite secant windows and scale ledgers for a proposed direction; it does not build a host limit, a quotient tangent space, a differentiability theorem, or a separation theorem.

Definition 2633.1 (TangentCone carrier)**.** *A* $\mathsf{TangentCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, L, V, H, C, P, N)$$

with the following displayed rows:

1. *B is the base-point row, naming the displayed point in the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier together with its* $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *ambient readbacks.*
2. *S is the feasible secant-window row. It lists finite windows in which the point obtained by moving from B along the candidate direction remains inside the convex feasible row at the displayed scale.*
3. *L is the positive scale and radius ledger. It records the inspected positive scale rows, metric radii, comparison budgets, and the refusal of zero-scale or unbounded-host reads.*
4. *V is the direction row, read through the* $\mathsf{VecSpace}^{\uparrow}$ *addition and scalar-action surface and compared only by the displayed metric windows.*
5. *H records componentwise* hsame *transport for the base point, secant windows, scale ledger, direction row, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a feasible-direction request through B, S, L, V before any structural row is consumed.*
7. *P is the* Pkg *provenance over* $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through B, S, L, V, H, C, P, N. It has no coordinate for an ambient host limit, quotient tangent space, differentiability theorem, normal cone, separation theorem, or closed convex analysis theorem outside the displayed finite rows.

Theorem 2633.2 (TangentCone secant-window exposure)**.** *Let $T = (B, S, L, V, H, C, P, N)$ be an accepted* $\mathsf{TangentCone}^{\uparrow}$ *carrier. Every feasible-direction read must first expose the base-point row B, then the secant-window row S, then the positive scale and radius ledger L, and only then the direction row V. Consequently a consumer sees a finite Bouligand-style secant surface, not a host limiting process or a selected tangent representative.*

Proof. Project the carrier of Definition 3596.1. The only rows available before the structural data are B, S, L, V, and the secant-window row is explicitly indexed by the base point exposed in B.

Replay the continuation row C. A feasible-direction request is routed through the displayed base-point read, then through the finite feasible secant windows, and then through the positive scale and radius ledger before the direction row can be inspected.

Transport by H preserves the same route, while P and N package only those rows. Thus no host limit, quotient tangent representative, differentiability theorem, normal cone, or separation theorem can be read from the packet. □

Theorem 2633.3 (TangentCone cone-scaling stability)**.** *For an accepted* $\mathsf{TangentCone}^{\uparrow}$ *carrier $T = (B, S, L, V, H, C, P, N)$, scaling an accepted displayed direction by a positive scale already present in L stays on the same finite tangent-cone surface. The resulting read uses the same base-point row B, the scaled secant windows recorded in S, the positive comparison ledger L, and the vector-space scalar-action row inside V, with transport and replay supplied by H and C.*

Proof. Start with a direction read accepted by the packet and a positive scale row displayed in L. The vector-space row V contains the scalar-action data, while the scale ledger records the positive comparison budget needed to reuse the feasible secant windows.

The secant-window row S is then read with the scaled direction and the same base point B. Convex feasibility is consumed only through the displayed $\mathsf{ConvexSet}^{\uparrow}$ row, and metric comparison is consumed only through the windows and radii already listed in L.

Finally replay C and transport through H. Provenance P and the local name row N certify that the scaled direction remains inside the same finite packet surface, with no appeal to a host closure theorem, quotient tangent space, differentiability result, or separation principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3596.1, Definition 3596.2, and Definition 3596.3 seed the finite convex feasible-direction packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host limit, quotient tangent space, differentiability theorem, separation theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, secant-window exposure, cone-scaling stability, transport, replay, provenance, and local naming for BEDC.Derived.TangentConeUp.

Chapter 2634

A Concrete Naming Certificate for $\mathsf{WeakTopology}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{WeakTopology}^{\uparrow}$ packet records a weak topology only through finite windows of scalar linear-functional observations. It sits over the source carrier of chapter 1950, the functional-analysis surface of chapter 94, the continuous-map rows of chapter 130, the open-neighbourhood classifier of chapter 88, and the scalar seal of chapter 38. The chapter names the finite observation-window certificate used by weak-convergence consumers; it does not choose dual elements, assert Banach–Alaoglu, compactness, quotient function spaces, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2634.1 (WeakTopology carrier)**.** *A* $\mathsf{WeakTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (V, F, U, T, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the source vector or normed-space carrier, read through the* $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *sibling rows.*
2. *F is the finite family of linear-functional observation rows. Each row names a scalar-valued functional already admitted by the functional-analysis surface, not a selected element of a completed dual space.*
3. *U is the weak-neighbourhood classifier row. It records finite scalar constraints around a displayed source point and sends them to the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood surface.*
4. *T is the finite test-window row. It lists the scalar probes, tolerances, source points, and comparison budgets inspected by a consumer.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for the observed functional values and their finite tolerance comparisons.*
6. *H records componentwise* hsame *transport for the source carrier, functional rows, neighbourhood classifier, test windows, scalar readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source reads through finite functional observations, from observations to weak-neighbourhood rows, and from those rows to scalar readbacks.*
8. *P is the* Pkg *provenance over* $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, F, U, T, R, H, C, P, N$. It has no coordinate for an unrestricted dual object, a selected family of all functionals, compactness, quotient function spaces, choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyTasteGate single carrier alignment`

Theorem 2634.2 (WeakTopology NameCert obligations)**.** *For every accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier $W = (V, F, U, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the source vector or normed-space row V, finite functional-observation family F, weak-neighbourhood classifier U, finite test windows T, scalar readback row R, componentwise transport H, displayed replay C, provenance P, and local naming N. Every weak-neighbourhood read must cite the sibling routes named above before it reaches a public topology row.*

Proof. Project the carrier of Definition 3597.1. The only source-facing coordinate is V, the only functional-facing coordinate is the finite row F, and the only topology-facing coordinate is the neighbourhood classifier U.

The test-window row T records which finitely many probes and tolerances are inspected. Scalar comparison then passes through R, so the public read receives finite $\mathsf{Real}^{\uparrow}$ observations rather than a selected dual object.

Finally transport, replay, provenance, and naming are supplied by H, C, P, N. Since the carrier contains no further coordinates, the listed rows exhaust the local certificate obligations. □

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyNamecertObligations`

Theorem 2634.3 (WeakTopology functional-window stability)**.** *Let $W = (V, F, U, T, R, H, C, P, N)$ be an accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier. Replacing a test request by a finite subwindow already recorded in T preserves the same source carrier V, the same displayed functional family F, the same weak-neighbourhood classifier U, and the same scalar readback route R. The transport and replay rows H, C carry the subwindow through the same provenance and local naming rows P, N.*

Proof. Read the finite test-window clause of Definition 3597.1. A subwindow of T still names only scalar probes from F, source-point reads from V, and finite tolerance comparisons.

Replay through C. The subwindow reaches the weak-neighbourhood classifier U and scalar row R along the same displayed path, so it cannot add a new functional, a hidden dual element, or a host function-space quotient.

Transport through H and repack through P, N. The packet therefore preserves the weak-topology observation surface while narrowing the inspected finite window. □

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyFunctionalWindowStability`

Theorem 2634.4 (WeakTopology scalar observation nonescape)**.** *Every scalar observation exported by an accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier factors through*

$$V, F, T, U, R, H, C, P, N.$$

Thus weak-convergence, neighbourhood, or continuity consumers may inspect only finite functional windows and their $\mathsf{Real}^{\uparrow}$ *scalar comparisons. They receive no Banach–Alaoglu compactness claim, no selected dual element, no quotient function space, no countable-choice selector, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with an exported scalar observation. By Definition 3597.2, the observation must first expose the source row V, the finite functional family F, the test windows T, and the weak-neighbourhood classifier U.

The scalar row R is then the only route to $\mathsf{Real}^{\uparrow}$ readback. It records finite observed values and tolerance comparisons, not a completed dual-space object or a host function-space equality.

The structural rows H, C, P, N preserve and name that same route. Since the excluded analytic principles have no carrier coordinate, they cannot be exported by the accepted packet. □

Falsifiable prediction: Every $\mathsf{WeakTopology}^{\uparrow}$ neighbourhood or convergence read factors through a source vector or normed-space carrier, a finite family of scalar functional-observation rows, a weak-neighbourhood classifier, finite test windows, scalar Real readbacks, transport, continuation, provenance, and local naming.

Independence witness: normedspace, functionalanalysis, continuousmap, topology, real: $\mathsf{WeakTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source carrier rows, finite linear-functional observations, weak-neighbourhood classifiers, finite scalar test windows, RealUp readbacks, transport, replay, provenance, and local naming without selecting a completed dual or importing compactness.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WeakTopology}^{\uparrow}$ packet is generated from a source vector or normed-space row, finite scalar linear-functional observation rows, a weak-neighbourhood classifier, finite test windows, scalar $\mathsf{Real}^{\uparrow}$ readbacks, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3597.1, Definition 3597.2, Definition 3597.3, and Definition 3597.4 seed the finite weak-topology observation-window certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Banach–Alaoglu, compactness, quotient function spaces, selected dual elements, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite functional-window stability, scalar observation nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeakTopologyUp.

Chapter 2635

A Concrete Naming Certificate for $\mathsf{CondensationTailSelector}^{\uparrow}$

Following Cauchy's condensation test and the Bishop–Bridges constructive series criterion, the $\mathsf{CondensationTailSelector}^{\uparrow}$ packet records the finite tail-selector route used by constructive series-test consumers before a regular rational handoff or real seal is read. It sits after the dyadic block comparison surface of chapter 1478, the alternating route of chapter 1483, the Dirichlet route of chapter 1497, the summable-tail modulus of chapter 1799, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a displayed selector for finite tail windows; it is not an infinite tail choice, a selected sum, a quotient series equality, a host real equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2635.1 (Condensation tail selector carrier)**.** *A* $\mathsf{CondensationTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, A, D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *dyadic-block row, exposing the monotone nonnegative source tail, block-index ledger, condensed block terms, and finite comparison budget.*
2. *A is the* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *face row, exposing the alternating sign ledger, monotone magnitude row, and finite tail-bound schedule when the downstream consumer uses the alternating route.*
3. *D is the* $\mathsf{DirichletTest}^{\uparrow}$ *face row, exposing bounded partial sums, coefficient windows, null-tail data, and the finite Dirichlet tail estimate schedule.*
4. *M is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row, naming the displayed modulus request that bounds the inspected tail window without choosing an infinite tail.*
5. *S is the selector row. It records which of the condensation, alternating, or Dirichlet faces supplies the next finite tail window and which modulus row certifies the requested tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row read only after K, A, D, M, S have displayed the selected finite tail route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff and the selected finite tail route have been replayed.*

8. *H records componentwise* hsame *transport for the condensation, alternating, Dirichlet, modulus, selector, handoff, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a series-test request through K, A, D, M, S, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{CondensationTailSelector}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay route. It has no coordinate for selected infinite sums, countable choice, quotient series equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CondensationTailSelectorUp.TasteGate.CondensationTailSelectorTasteGate single carrier alignment`

Theorem 2635.2 (Condensation tail selector NameCert obligations)**.** *For every accepted* $\mathsf{CondensationTailSelector}^{\uparrow}$ *carrier $T = (K, A, D, M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the condensation, alternating, Dirichlet, summable-tail modulus, selector, regular handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *selector coverage, requiring S to name one of the displayed* $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *faces before a tail window is exported;*
3. *modulus handoff, requiring the selected face to pass through the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row M before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is available;*
4. *non-escape, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to pass through $K, A, D, M, S, R, H, C, P, N$ and excluding selected infinite sums, countable choice, quotient series equality, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3598.1. The rows K, A, D are the only convergence-test faces, and S is the only row allowed to choose among those displayed finite faces for a requested tail window.

The selected face then passes through M. Thus the requested tail window is bounded by a displayed $\mathsf{SummableTailModulus}^{\uparrow}$ row before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R can be read.

Finally E is downstream of R, while H, C, P, N transport, replay, source, and locally name the same finite route. Since the carrier contains no coordinate for selected infinite sums, countable choice, quotient series equality, host real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2635.3 (Condensation tail selector test handoff)**.** *Every series-test consumer of an accepted* $\mathsf{CondensationTailSelector}^{\uparrow}$ *packet factors its tail read through the finite route*

$$K, A, D \longmapsto S \longmapsto M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer may cite the condensation, alternating, or Dirichlet test surface through one displayed selector row and one displayed modulus row, but it receives no selected infinite sum, countable-choice operator, quotient series equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer asking for a tail window usable by a Real-facing readback. The only available test data are the three displayed faces K, A, D, and the only row that records the chosen face is S.

The route then reads M before R. This forces the tail request to become a finite modulus-bound request rather than a hidden choice of an unbounded tail.

The seal row E is reached only after this handoff, and H, C, P, N preserve the same route under transport, replay it as displayed continuation data, record provenance, and name it locally. The excluded objects are not coordinates of the packet, so they cannot be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{CondensationTailSelector}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal while omitting the selected convergence-test face, the selector row, the summable-tail modulus row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cauchy-condensation-test, alternating-series-test, dirichlet-test, summable-tail-modulus, regseqrat, real: no listed sibling supplies the whole selector packet $T = (K, A, D, M, S, R, E, H, C, P, N$ because each sibling contributes a test face, a modulus row, a regular handoff, or a terminal seal while this chapter binds them as one finite tail-selector certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CondensationTailSelector}^{\uparrow}$ packet is generated from CauchyCondensationTest, AlternatingSeriesTest, DirichletTest, SummableTailModulus, RegSeqRat, and RealUp rows, together with componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3598.1, Definition 3598.2, and Definition 3598.3 seed the finite tail-selector route over condensation, alternating, Dirichlet, summable-tail modulus, regular handoff, Real seal, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, countable choice, quotient series equality, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector coverage, modulus handoff, test handoff, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CondensationTailSelectorUp.

Chapter 2636

A Concrete Naming Certificate for NestedIntervalCauchy$^{\uparrow}$

Following Bishop and Bridges on nested interval and Cauchy real constructions, the NestedIntervalCauchy$^{\uparrow}$ packet records the finite route by which a displayed nested interval chain becomes a regular rational Cauchy name before the terminal real seal is available. It consumes the dyadic endpoint and comparison rows of chapter 291, the finite observation schedules of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name boundary of chapter 38, and the dyadic finite-tail handoff of chapter 1811. The packet is a textbook-grounded L10 completion route for interval evidence; it is not a completed intersection point, quotient real, host real equality, countable-choice selector, or ambient completeness principle.

Definition 2636.1 (Nested interval Cauchy carrier)**.** *A* NestedIntervalCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested interval window. It records the inspected interval prefix, endpoint rows, containment rows, and nonempty-window evidence visible to the nested-interval consumer.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *endpoint and comparison ledger used to read endpoint order, diameter, and tolerance comparisons at the requested dyadic scales.*
3. *W is the shrinking-width witness row. It assigns each displayed precision request to a prefix index whose interval diameter is bounded by the requested tolerance.*
4. *S is the* StreamName$^{\uparrow}$ *schedule row. It names exactly the interval-prefix entries, endpoint observations, and diameter comparisons that may be opened before regular rational readback.*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row formed from the scheduled endpoint or midpoint observations and the shrinking-width witness.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after I, D, W, S, R have been displayed.*
7. *H records componentwise* hsame *transport for interval prefixes, dyadic ledgers, shrinking-width witnesses, stream schedules, regular rational readback, real seals, replay rows, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite nested-interval request through D, W, S, R, E and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalCauchy}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedDyadicCauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a completed intersection point, quotient real equality, host real equality, countable-choice selector, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.NestedIntervalCauchyUp.NestedIntervalCauchyTasteGate single carrier alignment`

Theorem 2636.2 (Nested interval Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *carrier*

$$K = (I, D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval prefix, dyadic endpoint ledger, shrinking-width witness, stream schedule, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*
2. *prefix exposure, requiring each interval read to be one of the displayed rows of I;*
3. *endpoint and diameter exactness, requiring every endpoint-order and diameter comparison to be read through D;*
4. *precision discharge, requiring every Cauchy tolerance request to be answered by a prefix index and comparison stored in W;*
5. *schedule compatibility, requiring S to expose only the interval-prefix, endpoint, and diameter observations used by W and R;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be formed only after the prefix, dyadic, width, and schedule rows have been projected;*
7. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to be consumed only after R and the structural rows have fixed the finite route.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3599.1. The projection exposes one finite nested interval prefix, one dyadic endpoint ledger, one shrinking-width witness, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Prefix exposure is forced by I. A consumer may inspect only interval rows already present in the finite prefix, and each such row carries the endpoint observations that the downstream dyadic ledger may compare.

Endpoint and diameter exactness are forced by D. Every order or diameter comparison used in the Cauchy route is a displayed dyadic comparison, so no external host interval relation is introduced.

Precision discharge is then forced by W. Each tolerance request names a displayed prefix index and a displayed comparison certifying that the selected interval is small enough for the requested regular rational read.

The schedule and handoff rows are downstream. The row S records the finite prefix, endpoint, and diameter observations that may be opened, and R repacks exactly those observations as regular rational Cauchy data.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection point, quotient real equality, host real equality, countable-choice selector, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 2636.3 (Nested interval Cauchy RegSeqRat handoff)**.** *Let $K = (I, D, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of K factors through the finite route*

$$I \longmapsto D, W \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite nested interval prefix, displayed dyadic endpoint and diameter comparisons, a shrinking-width witness, and scheduled endpoint observations before it exports the regular rational Cauchy row.

Proof. Start with the displayed interval prefix I. Its rows give the only endpoint data available to a regular rational consumer, and the containment rows keep later prefix rows inside earlier rows.

Use the dyadic ledger D and shrinking-width witness W to answer a displayed Cauchy tolerance request. The selected prefix index, endpoint observations, and diameter comparison are part of the same finite packet, so no external modulus or choice of tail is needed.

The schedule row S then fixes which interval endpoints or midpoint observations are opened. The handoff row R is obtained by reading those scheduled observations under the width discipline and repacking them as the regular rational Cauchy surface.

The rows H, C, P, N transport, replay, source, and name that handoff. Since the carrier has no additional tail selector, quotient-real coordinate, or completed intersection coordinate, every RegSeqRat read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalCauchy}^{\uparrow}$ packet is generated from finite nested interval windows, dyadic endpoint and diameter ledgers, shrinking-width witnesses, StreamName schedules, RegSeqRat handoffs, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3599.1, Definition 3599.2, and Definition 3599.3 seed the finite nested interval Cauchy route from interval-prefix evidence to RegSeqRat readback and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed intersections, quotient real equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix exposure, endpoint and diameter exactness, precision discharge, schedule compatibility, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.NestedIntervalCauchyUp.

Chapter 2637

A Concrete Naming Certificate for $\mathsf{TypeLike}^{\uparrow}$

Following Martin-Lof type theory, categorical families, and displayed fibrations, the $\mathsf{TypeLike}^{\uparrow}$ packet records the carrier-plus-classifier-plus-transport interface used when BEDC data are read as typed families rather than as an unindexed host universe. It sits beside the membership discipline of chapter 3668, the syntactic carrier of chapter 197, and the satisfaction carrier of chapter 198. The packet names only displayed classifier families, fiber rows, classifier transport, componentwise history transport, continuation replay, package provenance, and local naming; it does not introduce a universe hierarchy, quotient type equality, impredicative comprehension, host isomorphism of types, or an external category of families.

Definition 2637.1 (TypeLike carrier)**.** *A* $\mathsf{TypeLike}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the displayed base classifier family row, read as BEDC classifier data rather than as a host universe;*
2. *F is the fiber-carrier row assigning to each accepted base entry a finite displayed fiber packet;*
3. *Q is the classifier row for fiber membership, endpoint equality, and family-compatible reads;*
4. *E is the extensionality and endpoint-exactness ledger, recording which displayed fiber endpoints may be identified under the classifier;*
5. *H records componentwise* hsame *transport for B, F, Q, E together with replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay through which a consumer reads a base classifier, a fiber carrier, a classifier row, and an endpoint-exactness ledger;*
7. *P is the* Pkg *provenance over* $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient type universe, quotient identity, selected representative, host-level equivalence of categories, or a comprehension rule not already displayed by B, F, Q, E, H, C, P, N.

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeTasteGate single carrier alignment`

Theorem 2637.2 (TypeLike NameCert obligations)**.** *For every accepted* $\mathsf{TypeLike}^{\uparrow}$ *carrier*

$$T = (B, F, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *surface consists exactly of carrier admission for* T*, base-family classifier admission through* B*, fiber-carrier admission through* F*, classifier exposure through* Q*, endpoint-exactness exposure through* E*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 3600.1. Its base side is the displayed classifier family B, while its fiber side is the finite carrier row F indexed by accepted base entries.

The classifier row Q supplies the only family-compatible membership and endpoint reads. The exactness row E records when two displayed endpoints may be identified by that classifier, so endpoint equality is a carried ledger row rather than host type equality.

The structural rows H, C, P, N preserve and expose the same packet. Since no further coordinate is present, the NameCert obligations cannot ask for an ambient universe, quotient identity, selected representative, categorical equivalence, or external comprehension rule. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeNameCert obligations`
↪ variant `BEDC.Derived.TypeLikeUp.TypeLikeNameCertObligations`

Theorem 2637.3 (TypeLike classifier transport)**.** *Every downstream consumer of an accepted* $\mathsf{TypeLike}^{\uparrow}$ *packet that transports a fiber endpoint must factor through the finite route*

$$B, F, Q, E \longmapsto H, C, P, N.$$

Consequently the transported endpoint remains inside the displayed classifier family and cannot be replaced by host equality of types, an untracked equivalence of fibers, or a SetLike, FirstOrder, or ModelTheory read that bypasses the carried classifier row.

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeClassifierTransport`

Proof. Begin with an accepted carrier $T = (B, F, Q, E, H, C, P, N)$. A consumer can reach a fiber endpoint only after reading the base classifier B and the indexed fiber carrier F.

The classifier row Q fixes the admissible endpoint comparison, and E records the exact endpoint ledger for that comparison. Transport therefore has to preserve both the base-family row and the fiber classifier row.

The rows H, C, P, N then move, replay, source, and name the same finite route. Reads inherited from $\mathsf{SetLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, or $\mathsf{ModelTheory}^{\uparrow}$ may be cited only through the package provenance P and the displayed continuation C, so no sibling can supply a bypass around B, F, Q, E. □

Theorem 2637.4 (TypeLike displayed-family consumer exhaustion)**.** *Every downstream* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{TypeLike}^{\uparrow}$ *endpoint factors through the displayed base classifier* B*, fiber carrier* F*, classifier row* Q*, endpoint-exactness ledger* E*, componentwise transport* H*, displayed replay* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *row* N*. No such consumer may read the endpoint as unindexed membership, bare syntax, or satisfaction data before the displayed family route has been exposed.*

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeDisplayedFamilyConsumerExhaustion`

Proof. Begin with the accepted carrier $T = (B, F, Q, E, H, C, P, N)$ of Definition 3600.1. Its first four rows are the only displayed family data: base classifier, indexed fiber carrier, family-compatible classifier, and endpoint-exactness ledger.

Apply Definition 3600.3. Any endpoint transport used by a sibling consumer must preserve those four rows before the structural rows are available.

Finally inspect the sibling reads. $\mathsf{SetLike}^{\uparrow}$ can supply membership only through Q, $\mathsf{FirstOrder}^{\uparrow}$ can supply syntax only through the provenance row P and replay row C, and $\mathsf{ModelTheory}^{\uparrow}$ can supply satisfaction only after the same displayed classifier route has been named locally by N. Thus the three consumer families are exhausted by B, F, Q, E, H, C, P, N and cannot add an ambient universe, quotient type equality, or external category of families. □

Theorem 2637.5 (TypeLike SetLike FirstOrder ModelTheory carrier boundary)**.** *Every accepted* $\mathsf{TypeLike}^{\uparrow}$ *endpoint consumed by chapter 3668, chapter 197, or chapter 198 must expose the finite displayed-family route*

$$B, F, Q, E \longmapsto H, C, P, N$$

before the sibling read. The consumer may then interpret the exposed row as membership, syntax, or satisfaction-facing data, but it cannot replace the TypeLike carrier by unindexed SetLike membership, bare FirstOrder syntax, external ModelTheory satisfaction, quotient type equality, or an ambient host family.

Proof. Start with classifier transport, Definition 3600.3. It forces endpoint movement to preserve the displayed base classifier, fiber carrier, classifier row, and endpoint-exactness ledger.

Then apply consumer exhaustion, Definition 3600.4. The SetLike, FirstOrder, and ModelTheory consumers enter only after the displayed route has reached the transport, replay, provenance, and local naming rows.

Thus the sibling chapters can consume the endpoint only through B, F, Q, E, H, C, P, N. Since the carrier contains no unindexed membership source, bare syntax row, external satisfaction predicate, quotient equality, or ambient family coordinate, none of those can serve as a replacement boundary. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeSetLikeFirstOrderModelTheoryCarrierBoundary`

Theorem 2637.6 (TypeLike displayed-family inversion)**.** *Let* $T = (B, F, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{TypeLike}^{\uparrow}$ *carrier. Any endpoint read classified as* $\mathsf{TypeLike}^{\uparrow}$ *data determines the displayed base classifier row* B*, the fiber-carrier row* F*, the classifier row* Q*, and the endpoint-exactness ledger* E *before any structural row* H, C, P, N *is used. Thus endpoint readback is inverted through the concrete displayed-family spine*

$$B, F, Q, E \longmapsto H, C, P, N$$

and cannot be read first as host type equality, unindexed $\mathsf{SetLike}^{\uparrow}$ *membership, bare* $\mathsf{FirstOrder}^{\uparrow}$ *syntax, or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction.*

Proof. Project the carrier of Definition 3600.1. The endpoint-facing data appear in the first four rows: base classifier, indexed fiber carrier, family-compatible classifier, and endpoint-exactness ledger.

Use classifier transport, Definition 3600.3. It keeps any transported endpoint inside the displayed classifier family and prevents a downstream consumer from replacing the route by host type equality or an untracked fiber equivalence.

Finally apply consumer exhaustion, Definition 3600.4. The SetLike, FirstOrder, and ModelTheory consumers may enter only after B, F, Q, E have been exposed and carried through H, C, P, N. Hence an accepted endpoint read determines those four displayed rows before the structural rows are consumed. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeDisplayedFamilyInversion`

Falsifiable prediction: A consumer that claims a $\mathsf{TypeLike}^{\uparrow}$ endpoint transport while omitting the base classifier row, fiber-carrier row, classifier row, endpoint-exactness ledger, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 3600.1.

Independence witness: setlike, firstorder, modeltheory: $\mathsf{TypeLike}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a displayed base classifier family, indexed fiber carrier, classifier row, endpoint-exactness ledger, componentwise transport, continuation replay, provenance, and local NameCert row. SetLikeUp has membership without indexed fiber transport; FirstOrderUp has syntax and deduction rows without the carrier family; ModelTheoryUp has satisfaction rows over FirstOrderUp without a general classifier-family transport interface.

Closure status for $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeLike}^{\uparrow}$ packet is generated from a displayed base classifier family, indexed fiber carrier rows, a family-compatible classifier row, endpoint-exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3600.1, Definition 3600.2, Definition 3600.3, and Definition 3600.4 bind the displayed carrier-plus-classifier-plus-transport scope and exhaust the SetLikeUp, FirstOrderUp, and ModelTheoryUp sibling consumers over BHist, hsame, Cont, Pkg, NameCert, and the local family rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universe hierarchy, quotient type equality, impredicative comprehension, host isomorphism of types, Lean verification, bridge checking, or an external category of families.

Upgrade path: No theory-axis upgrade remains for this displayed classifier-family transport surface; Lean verification and bridge status are recorded separately.

Chapter 2638

A Concrete Naming Certificate for $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$

Following Bishop and Bridges on constructive apartness and completion, the $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ packet records a displayed apartness space whose Cauchy data are complete by finite tolerance inspection. It sits after the located completion criterion of chapter 3540, the apartness completion boundary of chapter 1787, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion object for L10 Real and Completion consumers; it is not a quotient completion, a selected limit, a trichotomy principle, a countable-choice schedule, or a host real equality test.

Definition 2638.1 (Cauchy-complete apartness space carrier)**.** *A* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, G, T, S, R, D, E, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed apartness-space carrier, recording the points, positive separation evidence, and local apartness classifier inspected by the packet.*
2. *G is the positive-gap row, carrying the finite apartness witnesses used to separate accepted Cauchy windows.*
3. *T is the Cauchy tolerance route, recording the requested dyadic budgets and the finite window discipline that makes the Cauchy request inspectable.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the Cauchy name.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including radius, endpoint, and comparison rows used by the Cauchy and apartness checks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, D have been exposed.*
8. *L is the* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *row, recording the finite located-completion admission route for the same Cauchy data.*
9. *Q is the* $\mathsf{ApartnessCompletion}^{\uparrow}$ *handoff row, recording the separated apartness-completion boundary read by completion-facing consumers.*

10. *H records componentwise* hsame *transport for the apartness source, positive gap, Cauchy tolerance route, stream window, regular readback, dyadic ledger, Real seal, located-completion criterion, apartness-completion handoff, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a Cauchy-complete apartness-space request through $X, G, T, S, R, D, E, L, Q$*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$, $\mathsf{LocatedCompletionCriterion}^{\uparrow}$, $\mathsf{ApartnessCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, G, T, S, R, D, E, L, Q, H, C, P, N$. It has no coordinate for quotient Cauchy classes, chosen representatives, selected limits, countable choice, trichotomy, host real equality, or a host-level complete space.

Lean4 ✓ `BEDC.Derived.CauchyCompleteApartnessSpaceUp.CauchyCompleteApartnessSpaceTasteGate`
`single carrier alignment`

Theorem 2638.2 (Cauchy-complete apartness space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *carrier $A = (X, G, T, S, R, D, E, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyComplete}}$ *surface consists exactly of carrier admission for the apartness source, positive-gap witnesses, Cauchy tolerance route, finite stream windows, regular rational readback, dyadic tolerance ledger, terminal Real seal, located-completion criterion, apartness-completion handoff, componentwise transport, displayed continuation replay, package provenance, and local naming rows.*

Every completion-facing consumer must read the finite route

$$X, G, T \longmapsto S, R, D \longmapsto E \longmapsto L, Q \longmapsto H, C, P, N$$

before treating the space as Cauchy complete for displayed apartness data.

Proof. Project the carrier of Definition 3601.1. The rows X, G, T expose the apartness source, positive separation witnesses, and the Cauchy tolerance route before any Real-facing or completion-facing row is available.

The finite readback then passes through S, R, D. Stream windows supply the observed name data, regular rational readback repacks those windows, and the dyadic ledger records the tolerance and comparison budgets used by both the Cauchy and apartness checks.

The terminal and completion rows are downstream. Row E seals the Real-facing read, row L records the located completion criterion route of chapter 3540, and row Q records the apartness completion boundary of chapter 1787.

Finally H, C, P, N transport, replay, source, and name only that displayed route. Since the carrier contains no quotient class, chosen representative, selected limit, countable-choice schedule, trichotomy row, host equality test, or host complete-space coordinate, those objects cannot become local NameCert obligations. □

Theorem 2638.3 (Cauchy-complete apartness space located-completion handoff)**.** *Every* LocatedCompletionCriteri $\mathsf{ApartnessCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *packet reaches the completion-facing surface only through the displayed apartness carrier X, positive-gap row G, Cauchy tolerance route T, finite* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, located-completion row L, apartness-completion handoff Q, componentwise transport H, displayed replay C, provenance P, and local name N.*

The handoff exports complete-apartness data only as finite NameCert rows. It does not export quotient completions, selected limits, countable choice, trichotomy, host real equality, or an ambient completeness theorem for all $\mathsf{Real}^{\uparrow}$ *names.*

Proof. Begin with a completion-facing or Real-facing consumer. Projection of the carrier gives the ordered route from the apartness source and positive gap rows through the Cauchy tolerance row, then through the finite stream and dyadic readback rows.

The located-completion handoff is forced through the existing siblings. The row L is read as the located criterion route named in Definition 3540.2, while Q is read as the apartness-completion boundary named in Definition 1787.2. Both are downstream of the same S, R, D, E readback route.

Transport and replay cannot add coordinates. Rows H, C, P, N preserve and name the same finite route, so a consumer receives complete-apartness evidence only through displayed BEDC rows and receives no quotient equality, selected representative, choice schedule, trichotomy decision, host equality test, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ consumer factors a completion-facing read through one apartness source row, one positive-gap row, one Cauchy tolerance route, one $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{Real}^{\uparrow}$ seal, one $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ row, one $\mathsf{ApartnessCompletion}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-completion-criterion, apartness-completion, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries apartness-source evidence, positive gaps, Cauchy tolerance inspection, finite readback, Real sealing, located-completion admission, apartness-completion handoff, transport, replay, provenance, and local naming as one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3601.1, Definition 3601.2, and Definition 3601.3 seed the finite Cauchy-complete apartness-space route over apartness sources, positive gaps, Cauchy tolerance windows, regular readback, Real seals, located-completion criteria, apartness-completion handoffs, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-source exactness, positive-gap visibility, Cauchy tolerance inspection, finite readback, Real seal handoff, located-completion criterion admission, apartness-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompleteApartnessSpaceUp.

Constructive story: A $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ packet is generated from an apartness source row, positive-gap evidence, a Cauchy tolerance route, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a LocatedCompletionCriterion row, an ApartnessCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2639

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ packet records the finite functorial surface of tail shift on regular Cauchy rational names. It sits over the finite window service of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the one-step tail-shift packet of chapter 1578, and the terminal real-name seal of chapter 38. The packet names the identity and composition behaviour of displayed tail shifts; it does not identify quotient streams, select host limits, import countable choice, or claim ambient completeness.

Definition 2639.1 (Regular Cauchy tail-functor carrier)**.** *A* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, W, D, I, K, J, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational name is inspected only through displayed finite windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the source windows, shifted windows, and comparison windows used by the functorial read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the precision requests and endpoint comparisons preserved by the shifts.*
4. *I is the identity-tail row, recording that the zero shift rereads the same displayed windows and tolerance rows.*
5. *K is the composable-tail row, recording two displayed tail shifts k and ℓ together with the single displayed shift $k + \ell$.*
6. *J is the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *handoff row that supplies each one-step or finite-tail shift certificate used by I and K.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after R, W, D, I, K, J have been displayed.*
8. *H records componentwise* hsame *transport for the source, window, tolerance, identity, composition, shift-handoff, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each tail-shift request through $R, W, D, I, K, J, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, D, I, K, J, E, H, C, P, N$. It has no coordinate for quotienting streams by tail agreement, selecting a limit, countable-choice tail selectors, host equality of sequences, or ambient completeness.

Theorem 2639.2 (Regular Cauchy tail-functor NameCert obligations)**.** *For every accepted* RegularCauchyTailFunct *carrier $F = (R, W, D, I, K, J, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the source, window, dyadic tolerance, identity-tail, composable-tail, shift-handoff, Real seal, transport, replay, provenance, and local naming rows; identity stability through I; composition stability through K and J; Real-seal non-escape through E; and refusal of quotient stream equality, selected limits, countable choice, host sequence equality, and ambient completeness as certificate data.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFunctorUp.RegularCauchyTailFunctorTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3602.1. The analytic rows are R, W, D, I, K, J, E, and the structural rows H, C, P, N transport, replay, source, and name the same finite data.

The identity obligation is read from I. It records the zero-tail request against the same displayed $\mathsf{StreamName}^{\uparrow}$ windows and the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, so the accepted read has no second source row.

The composition obligation is read from K and J. The two displayed shifts k and ℓ are admitted only through $\mathsf{RegularCauchyShift}^{\uparrow}$ handoff rows, and the same handoff surface records the single displayed shift $k + \ell$.

Finally E is downstream of the source, window, tolerance, identity, composition, and shift-handoff rows. Since the excluded quotient, selection, choice, host-equality, and completeness objects have no carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 2639.3 (Regular Cauchy tail-functor identity and composition)**.** *Let $F = (R, W, D, I, K, J, E, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of the identity tail read and every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a composed tail read factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegularCauchyShift}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The zero shift rereads the displayed R, W, D rows, and the composite of shifts k and ℓ is exported only through the displayed $k + \ell$ shift handoff. No consumer receives a quotient-stream equality, selected host limit, countable-choice selector, or ambient completeness theorem from the packet.

Proof. Use Definition 3602.2. The identity-tail row I is tied to the displayed source R, window schedule W, and tolerance ledger D, so an identity consumer cannot bypass those rows.

For composition, the row K records the two shifts and their displayed sum. The handoff row J supplies the required $\mathsf{RegularCauchyShift}^{\uparrow}$ certificates before any Real-facing seal is read.

The terminal row E seals only this displayed route. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming, and they do not add quotient, selection, choice, host-equality, or completeness coordinates. □

Theorem 2639.4 (Regular Cauchy tail-functor window composition)**.** *Let $F = (R, W, D, I, K, J, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *packet. For any two displayed tail shifts k and ℓ recorded by K, their composed tail read is the same finite-window route obtained by rereading the* $\mathsf{StreamName}^{\uparrow}$ *schedule W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *handoff row J for the displayed shift $k + \ell$. The* $\mathsf{Real}^{\uparrow}$ *seal E is reached only after that $k + \ell$ handoff, so the packet exports no quotient-stream equality, host sequence equality, selected limit, countable-choice selector, or ambient completeness row.*

Proof. Project the carrier in Definition 3602.1. The rows $R, W, D, K, J, E, H, C, P, N$ are the only rows visible to a composed-tail request, with K naming the two displayed shifts and their displayed sum.

Use Definition 3602.2 to read the local certificate obligations. The window data are reread from W, the tolerance checks from D, and the finite-tail handoff from J; the structural rows only transport and replay these displayed rows.

Apply Definition 3602.3 to route every Real-facing consumer through

$$\mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegularCauchyShift}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Because E is downstream of the displayed $k+\ell$ handoff, none of the refused quotient, host-equality, selection, choice, or completeness objects can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName schedule rows, a DyadicRatCore tolerance ledger, identity-tail and composable-tail rows, RegularCauchyShift handoff rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 3602.1, Definition 3602.2, Definition 3602.3, and Definition 3602.4 seed the finite tail-functor route over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RegularCauchyShiftUp, RealUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable-choice tail selectors, host sequence equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, identity-tail stability, composition stability, RegularCauchyShift handoff exactness, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailFunctorUp.

Chapter 2640

A Concrete Naming Certificate for $\mathsf{OperatorNorm}^{\uparrow}$

Following Bishop's constructive normed-space practice and the bounded-operator presentations in Conway and Kreyszig, the $\mathsf{OperatorNorm}^{\uparrow}$ packet records the public BEDC carrier by which a bounded linear operator exposes a nonnegative scalar bound, a norm-bound ledger, and the transport rows needed by downstream spectral, dual, and uniform-boundedness certificates. It is the named operator-norm surface over the bounded-linear-operator route of chapter 1644, the scalar endpoint discipline of chapter 38, and the functional-analysis stability obligation of Definition 94.3; it is not a host supremum construction, quotient operator equality, a Banach-completeness assertion, or a selected norm-attaining vector.

Definition 2640.1 (OperatorNorm carrier)**.** *An* $\mathsf{OperatorNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (T, K, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the carried source endpoint, target endpoint, linear map code, boundedness witness, and source-target norm reads.*
2. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint exposed as the operator-norm scalar boundary.*
3. *L is the norm-bound ledger, recording the input norm row, output norm row, scalar comparison row, and the local inequality route saying that the output norm is controlled by K on carried source inputs.*
4. *H records componentwise* hsame *transport for the bounded-linear-operator row, the* $\mathsf{Real}^{\uparrow}$ *bound endpoint, the norm-bound ledger, the replay row, provenance, and local naming.*
5. *C records displayed* Cont *replay from a bounded-operator request through T, K, L, H back to the same operator-norm surface.*
6. *P is the* Pkg *provenance over* $\mathsf{OperatorNorm}^{\uparrow}$*,* $\mathsf{BoundedLinearOperator}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, K, L, H, C, P, N. It has no coordinate for a host supremum over unit vectors, quotient equality of operators, norm-attainment choice, hidden Banach completeness, spectral-radius limits, or uniform-boundedness families.

Lean4 ✓ `BEDC.Derived.OperatorNormUp.OperatorNormTasteGate single carrier alignment`

Theorem 2640.2 (OperatorNorm NameCert obligations)**.** *For every accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *carrier* $O = (T, K, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-linear-operator source compatibility, nonnegative* $\mathsf{Real}^{\uparrow}$ *bound exposure, norm-bound ledger exactness, componentwise transport, continuation replay, provenance, local naming, and non-escape from the displayed scalar-bound surface. A downstream consumer may read the carried operator row, the displayed scalar bound, and the ledger showing where that bound is used, but it receives no host supremum, no quotient operator equality, no selected norm-attaining vector, no hidden completeness, and no spectral or uniform-boundedness conclusion.*

Proof. Carrier admission is projection from Definition 3603.1. The packet displays exactly the bounded-linear-operator row T, the nonnegative scalar endpoint K, the norm-bound ledger L, and the structural rows H, C, P, N.

The bounded-operator source compatibility is inherited from the public bounded-linear-operator row. Thus a consumer sees the carried source and target endpoints, the map code, and the boundedness witness before it can inspect the scalar bound.

The norm-bound surface is finite. The row L records the input norm, output norm, scalar comparison, and local inequality route, while K supplies the only Real-facing endpoint to which those comparisons may point.

Finally H, C, P, N transport, replay, source, and name only the displayed rows. Since the refused objects have no coordinate in O, the certificate cannot export a host supremum, quotient equality, selected norm-attainment data, hidden completeness, spectral-radius limits, or uniform-boundedness families. □

Theorem 2640.3 (OperatorNorm bound stability)**.** *If two accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *packets are related by the classifier, then they expose the same* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *source row, the same* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint, and the same norm-bound ledger up to the componentwise* hsame *transport row. Consequently every downstream use reads a transported scalar-bound certificate rather than a host-level equality of operators or norms.*

Proof. Unfold the classifier of Definition 3603.1. Its comparison coordinates are exactly the bounded-linear-operator row, Real bound endpoint, norm-bound ledger, replay row, provenance row, and local naming row.

The operator component transports through the inherited bounded-linear-operator classifier, while the scalar endpoint transports through the $\mathsf{Real}^{\uparrow}$ comparison component. The ledger component then moves the input-norm row, output-norm row, scalar comparison row, and inequality route together.

Thus any downstream consumer receives the same public scalar-bound surface after transport. No step introduces a host equality of operators, a supremum equality of norms, or a quotient representative. □

Theorem 2640.4 (OperatorNorm non-escape boundary)**.** *Every read of an accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto T, K, L, H, C, P, N \longmapsto \mathsf{OperatorNorm}^{\uparrow}.$$

In particular, the packet cannot be used to recover a host supremum over all unit vectors, a norm-attainment selector, quotient operator equality, a Banach-completeness assertion, a spectral-radius theorem, or a uniform-boundedness principle.

Proof. Start with an accepted packet $O = (T, K, L, H, C, P, N)$. Any read first exposes the bounded-linear-operator row T and the scalar endpoint K, then reads the norm-bound ledger L and structural rows.

Replay through C returns to the same finite route, and transport through H is componentwise. The provenance and naming rows record only this displayed surface.

The excluded analytic objects require coordinates not present in the carrier. Since no such coordinate can be manufactured by transport or replay, every accepted read remains inside the public operator-norm bound certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$.

Constructive story: An $\mathsf{OperatorNorm}^{\uparrow}$ packet is generated from a bounded-linear-operator row, a nonnegative $\mathsf{Real}^{\uparrow}$ bound endpoint, a norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3603.1, Definition 3603.2, Definition 3603.3, and Definition 3603.4 seed the public scalar-bound certificate over BoundedLinearOperatorUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OperatorNormUp.OperatorNormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host supremum construction, quotient operator equality, norm-attainment choice, hidden Banach completeness, spectral-radius limits, uniform-boundedness families, or bridge checking.

Upgrade path: Further checked bound-stability and non-escape theorems for BEDC.Derived.OperatorNormUp would extend the verification axis while preserving the displayed scalar-bound surface.

Chapter 2641

A Concrete Naming Certificate for $\mathsf{ContinuousImageCompact}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet records the finite-net reading of the image of a compact metric source under a displayed continuous map. It sits over the compact source of chapter 129, the graph and pointwise-continuity row of chapter 130, and the compact-uniform-continuity route of chapter 388. The packet names the finite image-net certificate needed by downstream compactness and extreme-value consumers; it is not an open-cover compactness theorem, not a host image-set construction, and not a quotient equality principle for target names.

Definition 2641.1 (Continuous image compact carrier)**.** *A* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, U, E, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes only the compact metric inspection data, finite source-net rows, source radii, and source* $\mathsf{NameCert}$ *provenance consumed by the image certificate.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from the source surface displayed by K into the target metric surface. It records the graph, pointwise continuity data, and target readback boundary available to the packet.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row for F over K. It turns each target tolerance request into a finite source-net radius before any image-net row is exported.*
4. *E is the finite source-net ledger. It records the displayed source centers, the radius attached to the requested target tolerance, and the finite coverage row supplied by K.*
5. *T is the finite target-image ledger. It records the target names $F(e)$ for each displayed source center e in E and the target tolerance justified through U.*
6. *H records componentwise* hsame *transport for the compact source, continuous-map row, compact-uniform-continuity row, source-net ledger, target-image ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an image-compactness request through K, F, U, E, T, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{ContinuousImageCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, F, U, E, T, H, C, P, N$. It has no coordinate for open-cover compactness, a host image subset, a selected inverse image, quotient equality of target names, or ambient completeness.

Theorem 2641.2 (Continuous image compact NameCert obligations)**.** *For every accepted* ContinuousImageCompac *carrier $I = (K, F, U, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the compact source, continuous-map row, compact-uniform-continuity row, source-net ledger, target-image ledger, transport, replay, provenance, and local naming rows;*
2. *source-net accountability, requiring every displayed source center and radius in E to be read through the compact metric row K;*
3. *continuity routing, requiring every target tolerance request to pass through the compact-uniform-continuity row U for the continuous-map row F before T is exposed;*
4. *image-net exactness, requiring every target center in T to be the displayed F-readback of a source center from E;*
5. *componentwise transport and replay stability under H and C for the same displayed rows;*
6. *ledger refusal, stating that open-cover compactness, host image sets, selected inverse images, quotient equality, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.ContinuousImageCompactUp.ContinuousImageCompactTasteGate single carrier alignment`

Proof. Project the accepted carrier. The mathematical rows are K, F, U, E, T, and the structural rows are H, C, P, N.

The source-net obligation is read first. Row E may display only centers and radii whose coverage data has already been supplied by the compact metric row K.

The continuity route is next. A target tolerance is sent through U, which supplies the source radius for the same displayed map row F over the same compact source K.

The target-image ledger is therefore forced. Each target center in T is obtained by reading F at one of the displayed source centers in E, and the radius attached to that target center is the tolerance justified by U.

Transport, replay, provenance, and local naming preserve the row order. Since the excluded objects do not occur among $K, F, U, E, T, H, C, P, N$, no local obligation can export them. □

Theorem 2641.3 (Finite-net image compactness)**.** *Every accepted* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *carrier $I = (K, F, U, E, T, H, C, P, N)$ exports compactness of the displayed image only as the finite target-net row T obtained from the source-net row E along the continuous-map row F and the compact-uniform-continuity row U. A downstream compactness consumer receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{CompactUniformContinuity}^{\uparrow}, \quad \mathsf{ContinuousImageCompact}^{\uparrow}$$

and receives no open-cover compactness datum, host image subset, selected preimage, quotient equality of target names, or ambient completeness coordinate.

Proof. Begin with an image-compactness request at a target tolerance. By the obligation theorem, the request must pass through U before the target-image ledger T can be read.

The row U supplies a source radius for the same map row F over the same compact source K. The compact source row then exposes a finite source-net ledger E at that radius.

Map the displayed source centers through F. The finite list of target readbacks is exactly the target-image ledger T, and the tolerance attached to it is the one already routed through U.

Rows H, C, P, N transport, replay, package, and name this finite route without adding another coordinate. Thus the exported compactness content is the finite image-net row T and not an open-cover, host-image, selected-preimage, quotient-equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}$.

Constructive story: A $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a ContinuousMapUp row, a CompactUniformContinuityUp row, a finite source-net ledger, a finite target-image ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3604.1, Definition 3604.2, and Definition 3604.3 seed the finite-net image-compactness route from compact metric sources to downstream compactness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host image-set construction, selected preimages, quotient equality of target names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-net accountability, continuity routing, image-net exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.ContinuousImageCompactUp.

Chapter 2642

A Concrete Naming Certificate for $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive compact metric analysis, the $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ packet records the finite BEDC surface by which a displayed finite cover over a compact metric source yields a positive-radius ledger. It sits between the compact metric source of chapter 129, the metric ball rows of chapter 57, the finite radius ledger of chapter 452, the theorem route of chapter 2301, and the uniform-modulus consumer of chapter 369. The packet names finite-cover-to-positive-radius data only; it is not an open-cover compactness theorem, a subcover selector, a host minimum principle, countable choice, or an ambient completeness claim.

Definition 2642.1 (Finite-cover Lebesgue-number carrier)**.** *A* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$F = (K, M, V, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row supplying the compact finite-net and coverage data admitted by the cover request.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row through which distances, rational balls, and containment checks are read.*
3. *V is the displayed finite cover row, listing the named cover cells and their metric ball witnesses.*
4. *R is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive radius ledger attached to the displayed finite cover.*
5. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *route row certifying that the compact metric source and finite cover have been routed to the positive ledger R.*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that may consume the positive radius after K, M, V, R, L have been exposed.*
7. *H records componentwise* hsame *transport for $K, M, V, R, L, U, C, P, N$.*
8. *C records displayed* Cont *replay from compact metric finite nets and metric ball containment rows to the finite cover, positive radius ledger, lemma route, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $K, M, V, R, L, U, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteCoverLebesgueNumberUp.TasteGate.FiniteCoverLebesgueNumberTasteGate single carrier alignment`

Theorem 2642.2 (Finite-cover Lebesgue-number NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier* $F = (K, M, V, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$ *surface consists exactly of five rows: carrier admission for the displayed tuple; cover-to-radius exactness binding the finite cover row V to the positive radius ledger R; lemma-row compatibility requiring the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *row L to read the same compact metric source, metric ball observations, finite cover, and positive ledger; uniform handoff through U; and non-escape from open-cover compactness, selected subcovers, host minima, countable choice, and ambient completeness.*

Proof. Project the accepted carrier of Definition 3606.1. The displayed tuple contains exactly the compact metric source, metric observation row, finite cover, positive radius ledger, lemma route, uniform handoff, and structural rows.

Cover-to-radius exactness is read through V and R. The finite cover is not replaced by an arbitrary open family, and the radius is not selected by a host minimum; it is the displayed positive ledger attached to the cover row.

Lemma-row compatibility is read through L. It records that the compact metric source K, metric ball row M, finite cover row V, and positive ledger R are the same rows consumed by the $\mathsf{LebesgueNumberLemma}^{\uparrow}$ route.

The uniform handoff U can be reached only after this finite route has been transported by H and replayed by C, with provenance and local naming stored in P and N. Since the carrier has no further coordinate, no accepted packet imports open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness. □

Theorem 2642.3 (Finite-cover Lebesgue-number uniform-radius handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier reaches the positive radius only through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{FiniteLebesgueNumber}^{\uparrow}, \quad \mathsf{LebesgueNumberLemma}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The consumer-facing route is the finite compact-cover-to-positive-radius handoff $K, M, V, R, L, U, H, C, P, N$, not open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness.

Proof. Read K, M, V first. These rows display the compact metric source, the metric ball observations, and the finite cover whose cover cells generate the positive-radius request.

Then read R and L. The radius ledger is the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ row for the displayed finite cover, and the lemma route records the compact-cover-to-ledger compatibility needed before a uniform-modulus consumer can inspect the radius.

Finally read U, H, C, P, N. These rows transport, replay, source, and name the same finite route. The carrier therefore exports a uniform-radius handoff without adding any coordinate for open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ packet is generated from a CompactMetric source row, Metric ball observations, a displayed finite cover, a FiniteLebesgueNumber positive radius ledger, a LebesgueNumberLemma route row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3606.1, Definition 3606.2, and Definition 3606.3 seed the finite compact-cover-to-positive-radius data route consumed by uniform-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteCoverLebesgueNumberUp.TasteGate.FiniteCoverLebesgueNumberTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, host minima, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-to-radius exactness, lemma-row compatibility, uniform handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.FiniteCoverLebesgueNumberUp.

Chapter 2643

A Concrete Naming Certificate for $\mathsf{ContractionPrinciple}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{ContractionPrinciple}^{\uparrow}$ packet records the theorem-level completion-neighbour object that connects the contraction-map surface of chapter 1418, the finite iteration ledger of chapter 2383, the complete-metric handoff of chapter 128, the metric source of chapter 57, and the endpoint-facing consumer discipline of chapter 396. The packet names the finite route through which contraction data becomes completion-facing evidence; it does not itself choose a fixed point or assert ambient completeness.

Definition 2643.1 (ContractionPrinciple carrier)**.** *A* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, d, T, q, I, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier row whose point histories and classifier are read by the contraction route.*
2. *d is the metric-distance row used for all finite source, target, and iterate-distance comparisons.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *graph row, exposing the self-map application surface without host function extensionality.*
4. *q is the rational contraction-bound ledger, including the displayed nonnegative row and strict unit-bound row consumed by finite estimates.*
5. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *route, displaying the initial row, finite iterate window, and residual-budget rows generated from T and q.*
6. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, naming the completion-facing service that may consume the finite Cauchy evidence without being stored as ambient completeness in the contraction packet.*
7. *H records componentwise* hsame *transport for the metric source, distance row, contraction graph, rational bound, finite iteration route, completion handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for metric reads, contraction graph inspection, bound comparison, finite iteration extension, completion handoff, and consumer readback.*
9. *P is the* Pkg *provenance row over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$ *row.*

The carrier is a theorem-level route packet. It contains no selected fixed point, no completed endpoint row, no host equality of maps, and no completeness assertion outside the displayed handoff row.

Lean4 ✓ `BEDC.Derived.ContractionPrincipleUp.ContractionPrincipleTasteGate single carrier alignment`

Theorem 2643.2 (ContractionPrinciple NameCert obligations)**.** *For an accepted* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier* $R = (X, d, T, q, I, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$ *surface consists of carrier admission, contraction-bound stability, iterate visibility, complete-metric handoff visibility, componentwise transport, continuation replay, package provenance stability, and local naming for the packet of Definition 3607.1. Every accepted read of the principle is routed through these rows.*

Proof. Carrier admission projects the displayed tuple $X, d, T, q, I, L, H, C, P, N$. The contraction graph row and rational bound ledger supply the map-level comparison surface, while the metric row is the only source of distance reads.

Iterate visibility is supplied by I. It records only finite Picard windows and residual budgets generated from T and q, so it cannot introduce a completed orbit or a selected endpoint.

The completion handoff is supplied by L, transported by H, and replayed by C. Provenance and local naming are then read from P and N, giving exactly the listed NameCert obligations and no additional analytic coordinate. □

Theorem 2643.3 (ContractionPrinciple completion handoff)**.** *Every* $\mathsf{BanachFixedPoint}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier must pass through the displayed route*

$$\mathsf{ContractionMapping}^{\uparrow}, \quad \mathsf{FixedPointIteration}^{\uparrow}, \quad \mathsf{CompleteMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}.$$

Equivalently, the consumer reads the rows $X, d, T, q, I, L, H, C, P, N$ before any endpoint-facing fixed-point seal can be consumed. The handoff exports contraction-map evidence, finite iterate evidence, metric-distance reads, complete-metric service visibility, transport, replay, provenance, and local naming; it does not export a selected fixed point, a completed orbit endpoint, host fixed-point equality, quotient equality of maps, or ambient completeness.

Proof. Start with the contraction surface. The rows T and q are read as the $\mathsf{ContractionMapping}^{\uparrow}$ graph and rational bound ledger over the $\mathsf{Met}^{\uparrow}$ source X, d.

Next read the finite iteration route I. It exposes the $\mathsf{FixedPointIteration}^{\uparrow}$ orbit window and residual budget that a completion-facing consumer may inspect before any endpoint row is available.

Finally pass through L, H, C, P, N. The row L is only a $\mathsf{CompleteMetric}^{\uparrow}$ handoff, while H and C preserve the same componentwise route under transport and replay. Therefore a $\mathsf{BanachFixedPoint}^{\uparrow}$ or $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives the displayed theorem-level route but must provide its own endpoint or completeness discipline downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{ContractionPrinciple}^{\uparrow}$ packet is generated from a MetricUp source row, a distance row, a ContractionMapping graph row, a rational contraction-bound ledger, a FixedPointIteration route, a CompleteMetric handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3607.1, Definition 3607.2, and Definition 3607.3 seed the theorem-level contraction principle route over metric, contraction-map, finite iteration, complete-metric, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, completed endpoints, host fixed-point equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-bound stability, iterate visibility, complete-metric handoff visibility, transport stability, replay stability, provenance stability, and local naming for BEDC.Derived.ContractionPrincipleUp.

Chapter 2644

A Concrete Naming Certificate for $\mathsf{CauchyProductConvergence}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{CauchyProductConvergence}^{\uparrow}$ packet records the finite Real-and-completion route by which two displayed regular Cauchy or summable rows expose a Cauchy product name. It sits over the regular rational readback of chapter 301, the tolerance ledger of chapter 291, the completion witness route of chapter 3579, and the product theorem surface of chapter 3584. The chapter names only the finite certificate route for product convergence; it does not assert quotient stream equality, selected infinite limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2644.1 (Cauchy product convergence carrier)**.** *A* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, P, T, R, D, W, E, H, C, Q, N)$$

with the following displayed rows:

1. *A and B are the two source rows, each presented either as a regular Cauchy row or as a summable row with the finite windows needed by the product route.*
2. *P is the Cauchy product window. It records the finite convolution band, the summand-pair reads, and the displayed partial product windows used by the convergence claim.*
3. *T is the tail-control row. It records the finite majorant, Mertens, or Bishop–Bridges tail ledger that bounds the product tail after the window P has been fixed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the product windows determined by A, B, P, T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the source reads, product-window reads, and tail-control inequalities.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window data obtained from the same product windows and tolerance requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback, stream windows, and dyadic tolerance ledger have been exposed.*
8. *H records componentwise* hsame *transport for the source rows, Cauchy product window, tail-control row, readback, tolerance ledger, stream windows, Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a product-convergence request through $A, B, P, T, R, D, W, E, H, Q, N$.*
10. *Q is the* Pkg *provenance over* $\mathsf{CauchyProductConvergence}^{\uparrow}$, $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite product-convergence history. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductConvergenceUp.CauchyProductConvergenceTasteGate single carrier alignment`

Theorem 2644.2 (Cauchy product convergence NameCert obligations)**.** *For every accepted* CauchyProductConverg *carrier $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two source rows, the product window, the tail-control row, the regular readback, the dyadic tolerance ledger, stream windows, the Real seal, transport, replay, provenance, and local naming;*
2. *product-window stability under componentwise* hsame *transport of A, B, P, T;*
3. *tail-control compatibility between the Cauchy product window P, the tail row T, and the dyadic tolerance ledger D;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback for the product windows exposed by P, T, D;*
5. $\mathsf{StreamName}^{\uparrow}$ *window exposure using the same finite product windows and tolerance rows;*
6. $\mathsf{Real}^{\uparrow}$ *seal nonescape through R, D, W, E;*
7. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
8. *ledger refusal for any product-convergence consumer that omits a source row, product window, tail-control row, tolerance ledger, readback row, stream window, Real seal, transport row, replay row, provenance row, or local naming row.*

Proof. Project the carrier of Definition 3608.1. The projection exposes exactly the two source rows, the Cauchy product window, tail-control ledger, regular readback, dyadic tolerance ledger, stream windows, Real seal, and structural support rows.

The product-window and tail obligations are bound by P, T, D. A consumer can read a product window only together with the corresponding source-window reads and dyadic tolerance request, and it can claim tail control only through the displayed row T.

The readback and seal obligations are bound by R, W, E. These rows repack the same finite product windows as regular rational and stream-name data before the terminal Real seal is available.

Finally H, C, Q, N transport, replay, package, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, or ambient completeness coordinate, the obligation list is exhaustive. □

Theorem 2644.3 (Cauchy product convergence tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *packet factors through the finite route*

$$A, B \longmapsto P, T, D \longmapsto R, W \longmapsto E, H, C, Q, N.$$

Thus the read first consumes the two source rows, then the displayed Cauchy product window, tail-control row, and dyadic tolerance ledger, then the regular rational and stream-name readbacks, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient stream semantics, selected product limit, countable-choice schedule, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$. The only source coordinates are A and B, so all source regularity or summability reads begin there.

The next stage is P, T, D. The product window P fixes the finite convolution band, the tail-control row T supplies the displayed convergence bound for the omitted band, and the dyadic ledger D records the tolerance requests used by both.

The readback stage is R, W. The regular rational row and stream-name row use the same product windows fixed by P, T, D, so there is no separate selected-limit coordinate.

Finally the seal stage is E, H, C, Q, N. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the readback rows, and the structural rows transport, replay, package, and name the same finite route. □

Theorem 2644.4 (Cauchy product convergence tail-budget fusion)**.** *Every accepted* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *tail-control row that consumes a* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *surface factors through chapter 3584 before the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows are read. In the carrier* $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$, *the theorem packet is consumed by the source-window and tail-control stage* A, B, P, T, *while* R, D, W, E *only read the resulting finite product route.*

Proof. Start with the carrier route of Definition 3608.3. Its first stage consumes A, B, P, T, and the provenance row Q already names $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ among the package sources.

The readback rows R, D, W, E occur only after this stage. Therefore a consumer cannot read regular rational, dyadic, stream-name, or Real-seal data for the product tail until the Mertens theorem packet has fixed the finite product-window surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductConvergence}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source row, product window, tail-control row, DyadicRatCore tolerance ledger, RegSeqRat readback, StreamName window, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductConvergence}^{\uparrow}$ packet is generated from two source regular Cauchy or summable rows, a Cauchy product window, a tail-control row, RegSeqRat readback, DyadicRatCore tolerance ledger, StreamName windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 3608.1, Definition 3608.2, and Definition 3608.3 seed the finite product-convergence route from source rows and Cauchy product windows to tail control, readback, tolerance, StreamName, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, product-window stability, tail-control compatibility, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductConvergenceUp.

Chapter 2645

A Concrete Naming Certificate for $\mathsf{CauchyProductMertens}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{CauchyProductMertens}^{\uparrow}$ packet records the finite theorem row that mediates between the displayed Mertens Cauchy product theorem of chapter 3584 and the product-convergence seal of chapter 3608. It sits over the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the stream-window route of chapter 126, the completion witness of chapter 3579, and the terminal Real naming surface of chapter 38.

Definition 2645.1 (Cauchy product Mertens carrier)**.** *A* $\mathsf{CauchyProductMertens}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, T, R, D, S, E, H, C, Q, N)$$

whose rows record two displayed regular Cauchy or summable source names A, B*, a finite convolution-window row* W*, a Mertens tail-budget row* T*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, a* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, a* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *handoff row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* Q*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$ *row* N*. The carrier predicate asserts that all rows belong to one finite product-theorem history from the Mertens theorem packet to the later convergence packet; it contains no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductMertensUp.CauchyProductMertensTasteGate single carrier alignment`

Theorem 2645.2 (Cauchy product Mertens NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductMertens}^{\uparrow}$ *carrier* $M = (A, B, W, T, R, D, S, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$ *surface consists exactly of carrier admission for those twelve rows, Mertens-window compatibility between* A, B, W, T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance routing through* D*,* $\mathsf{StreamName}^{\uparrow}$ *exposure through* S*, completion handoff through* E*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal of any consumer that omits one of these rows.*

Proof. Project the carrier in Definition 3609.1. The source, window, and tail rows A, B, W, T expose exactly the Mertens theorem surface needed before product convergence can be read.

The readback rows R, D, S, E then carry the same finite window through regular rational, dyadic, stream-name, and completion-witness routes. The structural rows H, C, Q, N transport, replay, source, and name that route.

Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, or ambient completeness coordinate, the displayed list exhausts the local NameCert obligations. □

Theorem 2645.3 (Cauchy product Mertens tail-budget handoff)**.** *Every accepted* $\mathsf{CauchyProductMertens}^{\uparrow}$ *tail-budget row that can be consumed by* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *first factors through the finite route*

$$A, B \longmapsto W, T \longmapsto R, D, S, E \longmapsto H, C, Q, N.$$

Thus the Mertens product theorem packet is read before the $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows of the convergence route are consumed.*

Proof. Start from an accepted carrier $M = (A, B, W, T, R, D, S, E, H, C, Q, N)$. The only rows that expose the product theorem surface are A, B, W, T, so a convergence-facing tail read begins with those coordinates.

The readback stage is then forced through R, D, S, E. These rows encode the same finite window as regular rational, dyadic tolerance, stream-window, and completion-witness data.

Finally H, C, Q, N provide transport, replay, provenance, and local naming. No other coordinate can supply a tail-budget handoff to the convergence packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductMertens}^{\uparrow}$ consumer that reaches a product-convergence tail handoff while omitting a source row, convolution window, Mertens tail row, RegSeqRat readback, DyadicRatCore tolerance ledger, StreamName window, CauchyCompletenessWitness handoff, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductMertens}^{\uparrow}$ packet is generated from two source regular Cauchy or summable rows, a finite convolution window, a Mertens tail-budget row, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, CauchyCompletenessWitness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$, seed)

Scope closed: Definition 3609.1, Definition 3609.2, and Definition 3609.3 seed the finite Mertens product-theorem route from source rows and tail budgets to readback, stream, completion, and convergence-facing rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyProductMertensUp.CauchyProductMertensTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Mertens-window compatibility, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, CauchyCompletenessWitness handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductMertensUp.

Chapter 2646

A Concrete Naming Certificate for $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$

Following Krein and Milman, the $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ packet records the finite extreme-point witness route that a convex-analysis consumer may inspect before any ambient locally convex compactness theorem is available. It sits over the compact metric carrier of chapter 129, the convex-set interface of chapter 208, the scalar $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the weak functional-window surface of chapter 3597, the operator-norm row of chapter 3603, and the finite weak-star window of chapter 3420. The chapter names only a finite exposed-face and candidate-extreme certificate; it does not assert full locally convex duality, selected supporting functionals, unrestricted compactness, quotient convex bodies, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2646.1 (Krein–Milman finite extreme-point carrier)**.** *A* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, V, X, F, E, S, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *row carrying the finite compact-window data through which the candidate convex body is inspected.*
2. *V is the ambient vector or functional-analysis row, read through the weak-topology and operator-norm sibling packets.*
3. *X is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the displayed convex subset, including the affine-combination and transport data available to the packet.*
4. *F is the finite exposed-face ledger, listing the exposed faces, scalar thresholds, comparison budgets, and source probes that the certificate actually opens.*
5. *E is the candidate extreme-point witness row, recording the displayed points and finite refusal rows for nontrivial convex decompositions inside the opened face ledger.*
6. *S is the separating-functional ledger. Each functional is read only through finite weak-topology, operator-norm, and scalar* $\mathsf{Real}^{\uparrow}$ *observation rows.*
7. *W records the scalar and metric windows used by F, E, S, including finite* $\mathsf{Real}^{\uparrow}$ *readbacks for values, gaps, and tolerance comparisons.*

8. *H records componentwise* hsame *transport for the compact metric row, ambient row, convex-set row, exposed-face ledger, candidate extreme witnesses, separating-functional ledger, scalar windows, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from compact and convex reads through exposed-face and separating-functional ledgers to the candidate extreme witness rows.*
10. *P is the* Pkg *provenance over* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, X, F, E, S, W, H, C, P, N$. It has no coordinate for a host choice of all extreme points, a completed dual space, full locally convex separation, unrestricted compactness, quotient convex-body equality, selected supporting functionals outside S, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.KreinMilmanFiniteExtremeUp.KreinMilmanFiniteExtremeUpTasteGate single carrier alignment`

Theorem 2646.2 (Krein–Milman finite extreme-point NameCert obligations)**.** *For every accepted* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *carrier*

$$K = (M, V, X, F, E, S, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$ *surface consists exactly of compact-window admission for M, ambient weak functional admission for V, convex-set admission for X, exposed-face ledger visibility for F, candidate extreme-point witness admission for E, separating-functional ledger visibility for S, scalar and metric window readback for W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every consumer of the finite extreme-point certificate must read the route

$$M, V, X \longmapsto F, S, W \longmapsto E \longmapsto H, C, P, N$$

before treating a candidate as an extreme witness. The route exports finite certificate rows only; it exports no selected set of all extreme points, no full separation theorem, no host compact convex hull equality, no quotient convex-body object, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3610.1. The compact and convex source rows are exactly M and X, while the ambient functional-analysis reads are confined to V.

The exposed-face and separating-functional data are then forced through F and S. Their scalar values, thresholds, and gap comparisons are opened through the finite window row W, so a consumer sees only displayed weak and normed scalar observations.

The candidate extreme witness row E is downstream from those ledgers. It records the displayed point and the finite refusal of nontrivial convex decompositions inside the opened face route, not a global selector for all extreme points.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row contains a completed dual space, unrestricted compactness object, quotient convex body, host separation theorem, or ambient completeness principle, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ packet is generated from compact metric rows, convex-set rows, weak functional-analysis windows, finite exposed-face ledgers, candidate extreme-point witnesses, separating-functional

ledgers, scalar Real readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{KreinMilmanFiniteExtreme}\uparrow}$, seed)

Scope closed: Definition 3610.1 and Definition 3610.2 seed the finite exposed-face route for a Krein–Milman extreme-point witness over compact metric, convex-set, real scalar, weak-topology, operator-norm, and Banach–Alaoglu window rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KreinMilmanFiniteExtreme}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full locally convex duality, unrestricted compactness, host separation theorems, selected supporting functionals, quotient convex-body equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window routing, convex-set admission, finite exposed-face exactness, candidate extreme witness admission, separating-functional ledger visibility, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.KreinMilmanFiniteExtremeUp.

Chapter 2647

A Concrete Naming Certificate for $\mathsf{CFunCarrier}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley continuous-function carriers, the $\mathsf{CFunCarrier}^{\uparrow}$ packet records a continuous function as finite source, map, target, modulus, and certificate rows rather than as an ambient function-space quotient. It sits between the terminal real-name seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the graph-and-modulus surface of chapter 130. The chapter isolates the non-quotient carrier object consumed by $\mathsf{RealPowerSeries}^{\uparrow}$ and later $\mathsf{ContinuousMap}^{\uparrow}$ readers: a consumer sees only displayed BEDC rows, not a host function, function extensionality principle, selected modulus function, or completed mapping space.

Definition 2647.1 (Continuous function carrier)**.** *A* $\mathsf{CFunCarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, T, M, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-domain row, including the finite input histories and any* $\mathsf{StreamName}^{\uparrow}$ *windows used to read source points.*
2. *F is the graph or map row, recording only the displayed relation between inspected source histories and target histories.*
3. *T is the target-domain row, including the terminal* $\mathsf{Real}^{\uparrow}$ *or sibling endpoint rows that the carrier may expose.*
4. *M is the modulus row, recording the finite tolerance conversion used by the continuity request.*
5. *Q is the certificate row that binds the source read, graph read, target read, and modulus read into one continuity witness.*
6. *W is the shared finite-window ledger over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and any real-facing tolerance readbacks needed by the source and target rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row used when the carrier is consumed by real-valued maps or power-series endpoints.*
8. *H records componentwise* hsame *transport for the source, map, target, modulus, certificate, window, readback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an input tolerance request through S, F, T, M, Q, W, R to the named endpoint rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CFunCarrier}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, M, Q, W, R, H, C, P, N$. It has no coordinate for host functions, quotient mapping spaces, function extensionality, selected moduli, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ContinuousFunctionCarrierUp.ContinuousFunctionCarrierTasteGate single carrier alignment`

Theorem 2647.2 (Continuous function carrier NameCert obligations)**.** *For every accepted* $\mathsf{CFunCarrier}^{\uparrow}$ *carrier $K = (S, F, T, M, Q, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, source access through S, graph or map access through F, target access through T, modulus access through M, continuity-certificate binding through Q, shared* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows through W, real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3611.1. The only rows available to a consumer are $S, F, T, M, Q, W, R, H, C,$ so source, graph, target, modulus, certificate, window, and readback obligations must be read there.

The continuity witness is not a host theorem. It is the finite row Q together with the shared windows W and the real-facing readback row R, so every endpoint read stays inside the displayed packet.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no other coordinate exists, the listed obligations exhaust the local NameCert surface. □

Theorem 2647.3 (Continuous function carrier L10 readback)**.** *Every L10 consumer read of an accepted* $\mathsf{CFunCarrier}^{\uparrow}$ *packet factors through*

$$S, F, T, M, Q \longmapsto W, R \longmapsto H, C, P, N.$$

Thus $\mathsf{RealPowerSeries}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *consumers may use the displayed source, graph, target, modulus, certificate, stream-window, regular-rational, real-readback, transport, replay, provenance, and local naming rows, but they receive no host function object, quotient mapping space, selected modulus function, function extensionality principle, countable-choice selector, or ambient real-completeness witness.*

Proof. Start with the displayed carrier. The source, graph, target, modulus, and certificate rows form the only continuity-facing entry point.

The finite windows and readback rows come next. They are the rows through which $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ observations are exposed to real-valued and continuous-map consumers.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. The field-exclusion clause of the carrier then blocks every host-function or quotient-function-space escape route. □

Falsifiable prediction: Every accepted $\mathsf{CFunCarrier}^{\uparrow}$ read must expose source, graph or map, target, modulus, certificate, finite StreamName windows, RegSeqRat readback, Real-facing readback when present, componentwise transport, displayed replay, provenance, and local naming rows.

Independence witness: continuousmap, real, streamname, regseqrat: $\mathsf{CFunCarrier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source, graph, target, modulus, certificate, window, readback, transport, replay, provenance, and local naming rows without quotienting maps or selecting host functions.

Closure status for $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$.

Constructive story: A $\mathsf{CFunCarrier}^{\uparrow}$ packet is generated from source-domain rows, graph or map rows, target-domain rows, a modulus row, a continuity-certificate row, shared finite StreamName and RegSeqRat windows, real-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3611.1, Definition 3611.2, and Definition 3611.3 seed the finite non-quotient continuous-function carrier route over source, graph, target, modulus, certificate, window, readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions, function extensionality, quotient mapping spaces, selected modulus functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source access, graph or map access, target access, modulus access, continuity-certificate binding, shared window routing and readback, real-facing readback, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousFunctionCarrierUp.

Chapter 2648

A Concrete Naming Certificate for $\mathsf{SturmRootCount}^{\uparrow}$

Following Sturm's theorem and Bishop–Bridges constructive analysis, the $\mathsf{SturmRootCount}^{\uparrow}$ packet records the finite route by which a displayed polynomial and two endpoint windows yield a root-count ledger without selecting any root in a host field. It consumes the sign-variation surface of chapter 2079 and exposes the retained count to the bracket consumers of chapter 2269. The packet names the count-and-handoff interface between a Sturm-chain variation ledger and a Real-facing isolating-window consumer; it does not name a chosen algebraic root.

Definition 2648.1 (Sturm root count carrier)**.** *A* $\mathsf{SturmRootCount}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, S, E, V, D, I, A, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, carrying the coefficient ledger, degree bound, derivative row, and polynomial identity whose interval counts are inspected.*
2. *S is the* $\mathsf{SturmSequence}^{\uparrow}$ *sign-variation window row, recording the accepted Sturm chain, endpoint sign table, zero-deletion witnesses, and the two visible variation counts.*
3. *E is the endpoint dyadic-window row. It stores the left and right dyadic endpoint cells, requested precisions, comparison rows, and endpoint containment checks.*
4. *V is the variation-count difference row. It records the displayed subtraction of the right endpoint variation count from the left endpoint variation count.*
5. *D is the* $\mathsf{RealRootIsolation}^{\uparrow}$ *bracket handoff row, exposing only the polynomial source, endpoint windows, and certified count accepted by the downstream isolating-window consumer.*
6. *I is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, retaining the same two endpoint windows and the same count at unary precision before a real seal is inspected.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for the two endpoint names and the bracket-facing interval read.*
8. *H records componentwise* hsame *transport for polynomial source, Sturm-sequence window, endpoint dyadic windows, variation-count difference, isolation handoff, regular-rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a root-count request through polynomial inspection, Sturm-chain variation, endpoint dyadic comparison, count subtraction, isolation handoff, regular-rational readback, and real-seal access.*

10. *K is the* Pkg *provenance over* $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, $\mathsf{RealRootIsolation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$ *registration row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected root, a host-field factorization, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SturmRootCountUp.TasteGate.SturmRootCountTasteGate single carrier alignment`

Theorem 2648.2 (Sturm root count NameCert obligations)**.** *For every accepted* $\mathsf{SturmRootCount}^{\uparrow}$ *carrier* $R = (P, S, E, V, D, I, A, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the rows listed in Definition 3612.1;*
2. *polynomial-source exactness through the coefficient, degree, derivative, and identity cells of P;*
3. *Sturm-window exactness through the finite chain, endpoint sign table, zero-deletion witnesses, and variation counts of S;*
4. *endpoint-window exactness through the dyadic cells, precision requests, comparison rows, and containment checks of E;*
5. *count-difference exactness through the displayed subtraction row V;*
6. *bracket-consumer discipline through the* $\mathsf{RealRootIsolation}^{\uparrow}$ *handoff row D;*
7. *regular-rational and Real-facing readback through I and A;*
8. *componentwise transport, continuation replay, provenance, and local naming through H, C, K, N.*

There is no additional NameCert obligation for selected roots, factorization over a host field, quotient real equality, real trichotomy, or ambient completeness.

Proof. Project the carrier of Definition 3612.1. The polynomial source P and the Sturm-window row S fix the displayed chain, endpoint sign table, zero-deletion witnesses, and variation counts, so a count read cannot inspect roots or factors outside the finite ledger.

The dyadic endpoint row E supplies the only endpoint cells admitted to the count. The difference row V then records the visible subtraction of endpoint variation counts, with no separate root-selection coordinate.

The handoff row D gives the downstream $\mathsf{RealRootIsolation}^{\uparrow}$ consumer exactly the source polynomial, endpoint windows, and accepted count. Rows I and A expose the same endpoint data through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal.

Finally, H, C, K, N transport, replay, source, and locally name the same route. Since none of those structural rows adds a selected root, factorization theorem, quotient equality proof, trichotomy decision, or completeness witness, the local certificate is exhausted by the displayed rows. □

Theorem 2648.3 (Sturm root count isolation handoff)**.** *Every accepted* $\mathsf{SturmRootCount}^{\uparrow}$ *read handed to a* $\mathsf{RealRootIsolation}^{\uparrow}$ *bracket consumer factors through*

$$P, S, E, V \longmapsto D \longmapsto I, A, H, C, K, N.$$

The consumer receives the polynomial source, Sturm-chain sign-variation window, endpoint dyadic windows, variation-count difference, regular-rational endpoint readback, Real-facing seal, componentwise transport, replay, provenance, and local naming. It receives no selected host root, no factorization over a host field, no quotient real equality principle, no trichotomy decision, and no ambient completeness principle.

Proof. Start with the downstream read. The source polynomial must pass through P, and the variation data must pass through the $\mathsf{SturmSequence}^{\uparrow}$ row S from chapter 2079.

The endpoint cells are then read from E, while V records the displayed count difference. Thus the bracket-facing row can see only a finite source, two endpoint windows, and one accepted integer count.

The handoff row D is the only row available to the $\mathsf{RealRootIsolation}^{\uparrow}$ consumer described in chapter 2269. The regular-rational readback I and real seal A preserve the same endpoint information for Real-facing consumers.

The remaining rows H, C, K, N compare, replay, source, and name the route componentwise. They add no hidden analytic principle, so the isolation handoff remains a finite Sturm-count certificate. □

Falsifiable prediction: Every $\mathsf{SturmRootCount}^{\uparrow}$ read handed to a $\mathsf{RealRootIsolation}^{\uparrow}$ bracket consumer factors through one $\mathsf{Poly}^{\uparrow}$ source row, one $\mathsf{SturmSequence}^{\uparrow}$ sign-variation window row, one dyadic endpoint-window row, one variation-count difference row, one bracket handoff row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, sturmsequence, realrootisolation, dyadicratcore, regseqrat, real: $\mathsf{SturmRootCount}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the polynomial source, Sturm-chain sign-variation windows, endpoint dyadic windows, count-difference row, bracket handoff, readback, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3612.1, Definition 3612.2, and Definition 3612.3 seed the finite bridge from Sturm-chain variation ledgers to Real-root-isolation bracket consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected roots, factorization over a host field, quotient real equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sturm-window exactness, endpoint-window exactness, count-difference exactness, isolation handoff, readback discipline, and transport-replay coverage for BEDC.Derived.SturmRootCountUp

Constructive story: A $\mathsf{SturmRootCount}^{\uparrow}$ packet is generated from a $\mathsf{Poly}^{\uparrow}$ source row, a $\mathsf{SturmSequence}^{\uparrow}$ sign-variation window row, dyadic endpoint windows, a variation-count difference row, a $\mathsf{RealRootIsolation}^{\uparrow}$ bracket handoff row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2649

A Concrete Naming Certificate for $\mathsf{UniformCauchyBicompletion}^{\uparrow}$

Following Bourbaki on uniform structures and Kelley on completions of uniform spaces, the $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ packet records the finite two-sided Cauchy-filter corridor used to approach a separated uniform-completion boundary. It sits over the uniform-space surface of chapter 339, the Cauchy-filter route of chapter 333, the separated comparison row of chapter 3365, the uniform-completion consumer of chapter 1473, and the real readback boundary of chapter 38. The chapter names only finite bicompletion data for a displayed left and right Cauchy route; it does not construct a quotient completion, select a limit representative, use choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2649.1 (Uniform Cauchy bicompletion carrier)**.** *A* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, L, R, J, S, E, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, recording finite entourage membership, diagonal, inverse, refinement, and composition data available to both Cauchy sides.*
2. *L is the left* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing the finite left filter-base observations and their entourage refinements.*
3. *R is the right* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing the finite right filter-base observations and their entourage refinements.*
4. *J is the joint Cauchy row, recording that requested left and right tails share a displayed common entourage refinement before any completion-facing read.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row, used only to identify two displayed boundary reads through separated zero-distance data.*
6. *E is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row, reached after U, L, R, J, S have been displayed.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for real-valued observers of the separated boundary, available only downstream of the uniform-completion handoff.*
8. *H records componentwise* hsame *transport for the uniform, left-filter, right-filter, joint-Cauchy, separated, completion, Real, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a bicompletion request through $U, L, R, J, S, E, A, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite two-sided Cauchy corridor. It has no coordinate for quotient filters, selected completion points, ambient host equality, choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyBicompletionUp.UniformCauchyBicompletionTasteGate single carrier alignment`

Theorem 2649.2 (Uniform Cauchy bicompletion NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *carrier $B = (U, L, R, J, S, E, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform-space row, two Cauchy-filter rows, joint-Cauchy row, separated comparison row, uniform-completion handoff, Real readback, transport, replay, provenance, and local naming;*
2. *left and right Cauchy-filter stability under componentwise* hsame *transport over the same* $\mathsf{UniformSpace}^{\uparrow}$ *row U;*
3. *joint-refinement exactness through J, requiring both Cauchy sides to reach every displayed entourage request through finite filter-base refinements;*
4. *separated-boundary exactness through S, requiring any identification of two displayed boundary reads to pass through the separated comparison row;*
5. *completion and Real nonescape through E and A, requiring every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to stay downstream of U, L, R, J, S;*
6. *displayed* Cont *replay and* Pkg *provenance for the whole two-sided corridor;*
7. *refusal of quotient filters, selected completion points, ambient host equality, choice, and ambient Real completeness as certificate data.*

Proof. Project the carrier of Definition 3613.1. The source rows are $U, L, R, J, S, E, A, H, C, P, N$, so every obligation must be read from those coordinates.

The two Cauchy sides are bound by L and R over the same uniform row U. The joint row J records the common finite entourage refinement requested by a bicompletion read, so neither side can be replaced by an independent host filter or a selected representative.

The separated and completion-facing rows are S, E, A. The row S is the only comparison route for displayed boundary reads, E is available only after the joint two-sided Cauchy data have been exposed, and A is downstream of that handoff.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same corridor. Since the carrier has no quotient-filter, selected-point, host-equality, choice, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 2649.3 (Uniform Cauchy bicompletion separated route)**.** *Every separated* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *packet factors through the finite route*

$$U \longmapsto L, R \longmapsto J \longmapsto S \longmapsto E, A, H, C, P, N.$$

Thus the consumer first reads the uniform entourage row, then both Cauchy-filter sides, then the joint-refinement row, then the separated comparison row, and only then the completion handoff and Real readback rows. The route exports no quotient completion, selected limit representative, choice schedule, ambient host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $B = (U, L, R, J, S, E, A, H, C, P, N)$. The only uniform source is U, so every completion-facing read begins with the displayed entourage data.

The two Cauchy inputs are L and R. Their common-refinement witness is J, which records the finite two-sided response to each requested entourage before a boundary read is available.

The separated boundary is S. Any comparison of two displayed completion reads must pass through this row, so host equality or a selected quotient representative cannot enter the route.

The endpoint rows are E, A, H, C, P, N. They hand off to the uniform-completion and Real-facing consumers, transport and replay the same data, and locally name the corridor without adding quotient, selection, choice, host-equality, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ consumer that reaches a separated $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ boundary while omitting the UniformSpace row, either CauchyFilter side, the joint-refinement row, the SeparatedMetric comparison row, transport, replay, provenance, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ packet is generated from a UniformSpace entourage row, two CauchyFilter rows, a joint-refinement row, a SeparatedMetric comparison row, a UniformCompletion handoff, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3613.1, Definition 3613.2, and Definition 3613.3 seed the finite two-sided Cauchy corridor from uniform entourage rows to separated completion and Real readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limit representatives, choice, ambient host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left and right Cauchy-filter stability, joint-refinement exactness, separated-boundary exactness, UniformCompletion handoff nonescape, Real readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyBicompletionUp.

Chapter 2650

A Concrete Naming Certificate for $\mathsf{DyadicCauchyApproximation}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ packet records the finite dyadic approximation surface used to pass from tolerance requests to regular rational and real-name consumers. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the completed real-name boundary of chapter 38, and the locator-facing route of chapter 3410. The chapter names only finite dyadic approximation data; it does not introduce quotient streams, selected limits, trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2650.1 (Dyadic Cauchy approximation carrier)**.** *A* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius, comparison, and refinement data requested by the approximation.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the inspected stream prefixes associated with the dyadic tolerance request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same dyadic tolerances and stream windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after D, W, Q have been displayed.*
5. *L is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *handoff row, recording the locator-facing use of the same finite approximation data.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, Real seal, locator handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an approximation request through D, W, Q, E, L to the named consumer rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite approximation history. It has no coordinate for quotient stream equality, selected limits, trichotomy, open-cover compactness, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.DyadicCauchyApproximationUp.DyadicCauchyApproximationTasteGate single carrier alignment`

Theorem 2650.2 (Dyadic Cauchy approximation NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *carrier* $K = (D, W, Q, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, dyadic tolerance admission through* D*, finite stream-window exposure through* W*, regular rational readback through* Q*, RealUp sealing through* E*, CauchyRealLocator handoff through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3614.1. The only rows available to an approximation consumer are $D, W, Q, E, L, H, C, P, N$, so tolerance, window, readback, seal, locator, transport, replay, provenance, and naming obligations must be read there.

The dyadic and stream obligations are bound by D and W. The dyadic radius, comparison, and refinement data come from the $\mathsf{DyadicRatCore}^{\uparrow}$ row, while the inspected prefixes are the finite $\mathsf{StreamName}^{\uparrow}$ windows attached to the same request.

The readback obligations are bound by Q, E, L. The regular rational row reads exactly the dyadic-window data, the Real seal is downstream of that row, and the locator handoff reuses the same finite approximation surface.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier has no quotient-stream, selected-limit, trichotomy, open-cover, choice, or ambient completeness coordinate, the listed obligations exhaust the certificate surface. □

Theorem 2650.3 (Dyadic Cauchy approximation RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *packet factors through*

$$D, W \longmapsto Q \longmapsto E, L, H, C, P, N.$$

Thus the consumer first receives the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row and finite* $\mathsf{StreamName}^{\uparrow}$ *windows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{CauchyRealLocator}^{\uparrow}$ *handoff. The route exports no quotient stream equality, selected limit, trichotomy, open-cover compactness, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $K = (D, W, Q, E, L, H, C, P, N)$. The only approximation-facing entry point is the pair D, W, so every tolerance request is read as a dyadic core row together with a finite stream-window row.

The regular rational stage is Q. It reads precisely the windows and dyadic tolerances already displayed by D and W, so no separate stream quotient or selected limit can enter the handoff.

The endpoint stage is E, L, H, C, P, N. The $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{CauchyRealLocator}^{\uparrow}$ handoff are downstream of the regular readback, while transport, replay, provenance, and local naming preserve the same finite route. □

Theorem 2650.4 (Dyadic Cauchy approximation window composition)**.** *Every accepted* DyadicCauchyApproximati *packet composes a tolerance request through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$*-facing seal* E *before any*

locator-facing or consumer-facing output is exported. The composed route is preserved by hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N, and it provides no quotient-stream equality, selected limit, trichotomy, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 3614.1. The tolerance request is recorded first by D, and the only stream data available to it is the finite window row W.

Apply the NameCert obligations theorem Definition 3614.2. It identifies the regular rational readback as Q, the Real-facing endpoint as E, and the locator handoff as L, all sourced by the same finite approximation history.

Apply the RegSeqRat handoff theorem Definition 3614.3. It forces every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through D, W, Q, E before transport, replay, provenance, and naming rows export the route.

Since the carrier has no quotient-stream, selected-limit, trichotomy, choice, or ambient-completeness coordinate, the displayed window composition is the full consumer-visible path. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{CauchyRealLocator}^{\uparrow}$ while omitting the DyadicRatCore tolerance row, StreamName window row, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, finite StreamName windows, RegSeqRat readback, a RealUp seal, a CauchyRealLocator handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3614.1, Definition 3614.2, and Definition 3614.3 seed the finite dyadic approximation route from tolerance rows and stream windows to regular rational readback, Real seal, and locator handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance routing, StreamName window exposure, RegSeqRat readback, Real seal nonescape, CauchyRealLocator handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCauchyApproximationUp.

Chapter 2651

A Concrete Naming Certificate for $\mathsf{SequentialCauchyRegularization}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ packet records the finite route by which ordinary Cauchy stream data is regularized into a displayed regular Cauchy name before the terminal Real seal is read. It sits over the displayed threshold discipline of chapter 376, the finite window service of chapter 126, the rational readback of chapter 301, the tolerance ledger of chapter 291, and the Real seal of chapter 38. The packet names a sequential regularization certificate only; it does not identify quotient streams, choose a limit, use countable choice, invoke trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2651.1 (Sequential Cauchy regularization carrier)**.** *A* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source Cauchy stream row. It records the ordinary sequential Cauchy data through finite requests, not through a quotient stream or selected limit.*
2. *T is the displayed regularization threshold row imported from the* $\mathsf{CauchyRegularization}^{\uparrow}$ *route. It records, for each requested precision, the finite stage after which the source windows are regular enough to serve the requested name.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite source windows and the regularized output windows read at the threshold supplied by T.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regularized rational name exposed by T and W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the source-window comparisons, threshold requests, regularized readbacks, and Real-facing output tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source row, regularization threshold, stream windows, readback, and dyadic tolerance ledger have been displayed.*
7. *H records componentwise* hsame *transport for the source stream, threshold row, window row, readback row, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a sequential regularization request through $X, T, W, R, D, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$, $\mathsf{CauchyRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite sequential regularization history. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host sequence equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.SequentialCauchyRegularizationUp.SequentialCauchyRegularizationTasteGate single carrier alignment`

Theorem 2651.2 (Sequential Cauchy regularization NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *carrier* $S = (X, T, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegulariza}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source Cauchy stream, displayed regularization threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming;*
2. *threshold stability under componentwise* hsame *transport of* X, T, W, D*;*
3. *regularized readback compatibility between the threshold row* T*, the window row* W*, the rational readback row* R*, and the tolerance ledger* D*;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape through* E*, excluding any read that bypasses* X, T, W, R, D*;*
5. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
6. *refusal of quotient streams, selected limits, countable choice, trichotomy, host sequence equality, and ambient Real completeness as certificate data.*

Proof. Project the carrier of Definition 3615.1. The analytic rows are the source stream X, the regularization threshold T, the window row W, the readback row R, the tolerance ledger D, and the terminal seal E.

The threshold obligation is read from T together with W and D. The row T names the finite stage supplied by the $\mathsf{CauchyRegularization}^{\uparrow}$ route, while W and D record the finite windows and tolerance comparisons checked at that stage.

The readback and seal obligations are read from R and E. The regularized rational name is exposed only through R, and the Real-facing seal is downstream of the same threshold, window, and tolerance data.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, host-equality, or completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 2651.3 (Sequential Cauchy regularization Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *packet factors through the finite route*

$$X \longmapsto T \longmapsto W, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the consumer first reads the source Cauchy stream, then the displayed regularization threshold from $\mathsf{CauchyRegularization}^{\uparrow}$*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient stream, selected limit, countable-choice selector, trichotomy row, host sequence equality, or ambient completeness theorem.*

Proof. Start with an accepted packet $S = (X, T, W, R, D, E, H, C, P, N)$. The only source coordinate is X, so any sequential Cauchy read begins with the displayed source stream row.

The regularization stage is T, W, D. The threshold row T supplies the finite stage used by the regularization route, and the window and tolerance rows record the exact finite observations made at that stage.

The readback stage is R. It packages the regularized windows as a regular rational name before any Real-facing seal is available.

The terminal stage is E, H, C, P, N. These rows seal, transport, replay, record provenance for, and locally name the same route, and none of them adds quotient, selection, choice, trichotomy, host-equality, or completeness data. □

Falsifiable prediction: Every accepted $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source Cauchy stream row, displayed CauchyRegularization threshold, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ packet is generated from a source Cauchy stream, a displayed CauchyRegularization threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3615.1, Definition 3615.2, and Definition 3615.3 seed the finite sequential regularization route from ordinary Cauchy stream data to regular readback and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold stability, regularized readback compatibility, DyadicRatCore tolerance routing, StreamName exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialCauchyRegularizationUp.

Chapter 2652

A Concrete Naming Certificate for MetaCICDecidableTypingFrontier$^{\uparrow}$

The MetaCICDecidableTypingFrontier$^{\uparrow}$ packet realizes the checker frontier of chapter 9379 as finite NameCert data. It sits beside the decidability witness of chapter 751, the decidability boundary packet of chapter 749, the local diamond witness of chapter 3595, and the self-compilation substrate of section 9235.3. The packet names the boundary between checked typing inference and bounded normal equality; it does not claim unrestricted beta-conversion, global normalization, subject reduction, confluence, or a closedness decider.

Falsifiable prediction: Every accepted MetaCICDecidableTypingFrontier$^{\uparrow}$ packet displays checker rows for `inferTypeCtx`, `checkSameTerm`, `Term.eq`, bounded normal equality, frontier refusal, transport, continuation, provenance, and local naming.

Independence witness: metacicdecidabilitywitness, metacicdecidableboundary, bounded-normal-equality-checker, metacic-local-diamond-witness, metacic-confluence-audit-packet: the decidability witness packages positive checker exactness, the decidability boundary records a broader refusal surface, the bounded-normal-equality checker names the normal-form comparison row, the local-diamond witness names confluence peaks, and the confluence audit packet records join obligations. None is bijective to MetaCICDecidableTypingFrontier$^{\uparrow}$.

Definition 2652.1 (MetaCIC decidable typing frontier carrier)**.** *A* MetaCICDecidableTypingFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the `inferTypeCtx` row. It records the context, represented term, optional inferred type, and the checked soundness and completeness surface available for that option-valued procedure.*
2. *S is the `checkSameTerm` row. It records successful same-term checks as structural equality evidence and refuses to treat failure as negative beta-conversion data.*
3. *Q is the `Term.eq` row. It records Boolean structural equality, reflexivity, true-to-equality, equality-to-true, and the exported correctness theorem for represented terms.*
4. *B is the bounded normal-equality row. It records the fuel, bounded normalizer outputs, and finished-normalization hypotheses required before `decideClosedBetaConv` can be read as a sound comparison.*

5. *R is the frontier refusal row. It records that unrestricted beta-conversion, global normalization, arbitrary closedness decidability, subject reduction, and confluence are not exported by this packet.*
6. *H records componentwise* hsame *transport for the typing checker, same-term checker, structural equality, bounded normal equality, frontier refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decidable-typing-frontier request through I, S, Q, B, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$*, the MetaCIC decidability audit map,* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unrestricted beta-conversion oracle, a global normalization theorem, a subject-reduction theorem, a confluence theorem, host quotient equality of terms, or an ambient completeness principle for all represented syntax.

Lean4 ✓ `BEDC.Derived.MetaCICDecidableTypingFrontierUp.MetaCICDecidableTypingFrontierTasteGate single carrier alignment`

Theorem 2652.2 (MetaCIC decidable typing frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$ *carrier $F = (I, S, Q, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypingFront}}$ *surface consists exactly of typing-checker admission through I, same-term checker admission through S, structural equality readback through Q, bounded normal-equality admission through B, frontier refusal through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Later Lean TasteGate work must provide Nontrivial *and* FieldFaithful *instances for this carrier surface. Those instances must witness that the packet cannot collapse to an empty frontier marker and cannot hide a missing typing, same-term, structural-equality, bounded-normal, refusal, transport, replay, provenance, or naming row.*

Proof. Project the carrier of Definition 3616.1. The positive checker rows are I, S, Q: typing inference, same-term checking, and structural equality.

The bounded comparison row is B. It is read only as bounded normal equality with the finished-normalization hypotheses named by the audit map, so it cannot become unrestricted beta-conversion data.

The refusal row R records the open boundary for global normalization, closedness decidability, subject reduction, confluence, and host quotient equality. This row is part of the certificate rather than prose outside the packet.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same frontier route. Since the carrier has no coordinate for the stronger theorems listed above, the displayed rows exhaust the NameCert obligations. □

Theorem 2652.3 (MetaCIC decidable typing frontier nonescape)**.** *Every consumer of an accepted* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$ *packet factors through the finite route*

$$\texttt{inferTypeCtx},\ \texttt{checkSameTerm},\ \texttt{Term.eq},\ \texttt{decideClosedBetaConv} \longmapsto R, H, C, P, N.$$

The route exports checked typing-inference and structural same-term data, plus bounded normal-equality data under its displayed hypotheses. It exports no unrestricted beta-conversion oracle, no global normalization theorem, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, and no host quotient equality of terms.

Proof. Start with an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$. Rows I, S, Q expose the typing, same-term, and structural equality procedures named by the audit map.

The bounded normal-equality request then passes through B. This row contains the fuel and finished-normalization boundary, so the consumer receives only the checked bounded comparison surface.

The refusal row R is read before the structural rows finish the packet. It prevents the consumer from replacing the bounded route by global beta-conversion, normalization, subject reduction, confluence, closedness decidability, or quotient equality.

The final rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. □

Theorem 2652.4 (MetaCIC decidable typing audit split)**.** *Every audit-map consumer of an accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet first reads* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq` *as the positive typing and structural syntax-equality side. It reads* `decideClosedBetaConv` *only through the bounded normal-equality row and its displayed finished-normalisation hypotheses. The refusal row then blocks any promotion to global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or host quotient equality of terms.*

Proof. Project an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$. The positive checker side is exactly I, S, Q: typing inference through `inferTypeCtx`, same-term checking through `checkSameTerm`, and structural equality through `Term.eq`.

The beta-conversion side is narrower. A consumer reaches `decideClosedBetaConv` only through B, whose stored data includes bounded normalizer outputs, fuel, and the finished-normalisation hypotheses required by the audit map.

The refusal row R is read before the packet is exported through H, C, P, N. It prevents the bounded comparison row from being reinterpreted as global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality.

Thus the packet preserves the audit-map split: positive typing and syntax equality are read directly from checker rows, while beta conversion remains bounded and hypothesis-indexed. □

Theorem 2652.5 (MetaCIC decidable typing coverage frontier)**.** *Every accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet exports exactly the positive checker rows* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq`*. It exports* `decideClosedBetaConv` *only through the bounded normal-equality row carrying displayed fuel, bounded normalizer outputs, and finished-normalization hypotheses. Therefore any consumer claiming global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality must leave this packet.*

Proof. Project an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$ from Definition 3616.1. The projected positive checker rows are exactly I, S, Q, corresponding to `inferTypeCtx`, `checkSameTerm`, and `Term.eq`.

Use Definition 3616.2. It identifies the local NameCert obligations with the displayed checker rows, bounded normal-equality row, refusal row, and structural rows, with no hidden coordinate for stronger metatheory.

Apply Definition 3616.4. The only route to `decideClosedBetaConv` passes through B, where the fuel, bounded outputs, and finished-normalization hypotheses are part of the exported row.

Finally apply Definition 3616.3. The refusal row R blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and quotient equality from being read as packet exports. □

Theorem 2652.6 (MetaCIC decidable typing frontier positive checker exactness)**.** *An accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet exports positive checker exactness only for* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq`*. Any read of* `decideClosedBetaConv` *must pass through the bounded normal-equality row with displayed fuel, bounded normalizer outputs, and finished-normalization hypotheses, and the refusal row blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and quotient equality.*

Proof. Project the carrier $F = (I, S, Q, B, R, H, C, P, N)$ of Definition 3616.1. The positive checker rows are exactly I, S, Q, so exactness for typing inference, same-term checking, and structural equality is confined to those rows.

Compose the NameCert obligations of Definition 3616.2 with the audit split of Definition 3616.4. They keep the bounded beta-conversion read in B, where fuel and finished-normalization hypotheses are displayed before any comparison result is exported.

Finally use the coverage frontier of Definition 3616.5. It separates the positive checker rows from the bounded normal-equality row and reads the refusal row R before transport, replay, provenance, and local naming finish the packet.

Because the carrier has no row for global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality, those claims cannot be exported as positive checker exactness. □

Closure status for NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$.

Constructive story: A MetaCICDecidableTypingFrontier$^{\uparrow}$ packet is generated from inferTypeCtx typing rows, checkSameTerm rows, Term.eq structural equality rows, bounded normal-equality rows, frontier refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 3616.1, Definition 3616.2, Definition 3616.3, and Definition 3616.4 seed the finite decidable-typing frontier over checked typing inference, structural equality, bounded normal equality, and explicit refusal rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted beta-conversion, global normalization, subject reduction, confluence, arbitrary closedness decidability, host quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typing-checker exactness, same-term exactness, structural equality readback, bounded-normal comparison exactness, frontier refusal, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDecidableTypingFrontierUp.

Chapter 2653

A Concrete Naming Certificate for $\mathsf{DenseEmbedding}^{\uparrow}$

Following Kelley and Bourbaki on dense embeddings, the $\mathsf{DenseEmbedding}^{\uparrow}$ packet records the finite BEDC surface by which a source metric carrier is read inside a target metric completion through a dense image-window and an approximation ledger. It sits before the specialized metric-completion embedding packet of chapter 3351, and it supplies a canonical source route for $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumers in chapter 38, chapter 301, chapter 126, chapter 291, and chapter 3579. The packet names dense-image approximation as finite certificate data; it does not import a host embedding theorem, quotient representatives, selected limits, countable choice, or ambient completion completeness.

Definition 2653.1 (Dense embedding carrier)**.** *A* $\mathsf{DenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, T, C, I, W, A, E, H, K, P, N)$$

with the following displayed rows:

1. *S is the source metric row, including the finite carrier, distance readings, and source-side comparison windows.*
2. *T is the target metric row, read only through displayed target-distance comparisons.*
3. *C is the target completion row, carrying the completion-facing names used by the embedding read.*
4. *I is the source-to-target image row. It records the embedded image of inspected source points without choosing quotient representatives or host limits.*
5. *W is the dense image-window row. Each density request opens a finite target window, a source witness window, and the target-distance tolerance being tested.*
6. *A is the approximation ledger. It records that the witness in S reaches the target request through I, W, T, C at the displayed tolerance.*
7. *E is the completion handoff row exposed to completion-facing consumers after S, T, C, I, W, A have already been read.*
8. *H stores componentwise* hsame *transport for the source metric, target metric, completion, image, dense window, approximation ledger, handoff, continuation, provenance, and local naming rows.*

9. *K stores displayed* Cont *replay from source observations to image observations, from image observations to dense windows, and from dense windows to the completion handoff row.*
10. *P is the* Pkg *provenance over* $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, C, I, W, A, E, H, K, P, N$. It has no coordinate for host topological density, quotient completion equality, a selected limit, a global inverse, countable choice, or ambient completeness of the target completion.

Lean4 ✓ `BEDC.Derived.DenseEmbeddingUp.DenseEmbeddingTasteGate single carrier alignment`

Theorem 2653.2 (Dense embedding NameCert obligations)**.** *For every accepted* $\mathsf{DenseEmbedding}^{\uparrow}$ *carrier $D = (S, T, C, I, W, A, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the source metric, target metric, target completion, source-to-target image, dense image-window, approximation ledger, completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every density read must first expose S, T, C, I, W, A before the handoff row E can be consumed.*

Proof. Project the carrier of Definition 3617.1. The source metric, target metric, completion row, image row, dense window, and approximation ledger are exactly S, T, C, I, W, A, so a density request has no other analytic coordinates to inspect.

The handoff row E is downstream of those six rows. Hence completion-facing consumers first see the displayed source witness, its image, the finite target window, and the approximation ledger before receiving any completion data.

The structural rows H, K, P, N only transport, replay, source, and name that same finite route. Since the carrier excludes host density, quotient representatives, selected limits, countable choice, global inverses, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2653.3 (Dense embedding completion handoff)**.** *Every* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenseEmbedding}^{\uparrow}$ *packet factors through the finite route*

$$S, T, C, I \longmapsto W \longmapsto A \longmapsto E$$

together with H, K, P, N. The route may be read through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows named by the packet, but it exports no selected completion limit, quotient representative, ambient density theorem, countable-choice selector, or host completeness principle.*

Proof. Start with the NameCert obligations of Definition 3617.2. They place the source metric row, target metric row, target completion row, and image row before any dense-window or approximation-ledger read is available.

The dense image-window W opens only the finite source and target observations requested by the consumer. The approximation ledger A then records that this displayed source witness reaches the requested target window through the image row I at the stated tolerance.

Only after that ledger is read does the completion handoff row E become available. The rows H, K, P, N preserve and name the same route, so specialized completion, real, and Cauchy-completeness consumers receive finite dense-embedding data rather than host completion structure. □

Falsifiable prediction: Every accepted $\mathsf{DenseEmbedding}^{\uparrow}$ completion-facing read factors through a source metric row, target metric row, target completion row, source-to-target image row, dense image-window, approximation ledger, completion handoff row, transport, continuation replay, provenance, and local naming.

Independence witness: metric-completion-dense-embedding, real, regseqrat, streamname, dyadicratcore, cauchy-completeness-witness: $\mathsf{DenseEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it carries the canonical dense image-window and approximation ledger before those siblings consume the completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseEmbedding}^{\uparrow}$ packet is generated from a source metric row, target metric row, target completion row, source-to-target image row, dense image-window, approximation ledger, completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3617.1, Definition 3617.2, and Definition 3617.3 seed the canonical dense-embedding carrier and completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topological density, quotient completion equality, selected limits, global inverses, countable choice, ambient completion completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, dense image-window exactness, approximation-ledger handoff, completion nonescape, and TasteGate field faithfulness for BEDC.Derived.DenseEmbeddingUp.

Chapter 2654

A Concrete Naming Certificate for $\mathsf{UniformCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completions, the $\mathsf{UniformCauchyCompletion}^{\uparrow}$ packet records the finite uniform-Cauchy handoff used by Phase Real and Completion consumers before any completed object is exposed. It sits over the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy-sequence space surface of chapter 406, the terminal Real seal of chapter 38, and the Bishop completion route of chapter 3484. The packet is a finite NameCert surface for uniform Cauchy families; it is not a quotient completion, selected limit, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2654.1 (Uniform Cauchy completion carrier)**.** *A* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the uniform Cauchy family row, recording the inspected finite family, common-window requests, and uniform Cauchy tolerance responses.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the windows used by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on those same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that discharges every displayed uniform Cauchy request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *and completion seal row reached only after F, S, R, D have been displayed, with the sibling handoff to* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *and* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *recorded as part of the same finite route.*
6. *H records componentwise* hsame *transport for the family, stream-window, regular-readback, dyadic-ledger, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through F, S, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{BishopCompletionTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite uniform-Cauchy completion handoff. It has no coordinate for quotient completions, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 2654.2 (Uniform Cauchy completion NameCert obligations)**.** *For every accepted* UniformCauchyComplet *carrier* $U = (F, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform Cauchy family, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real and completion seal, transport, replay, provenance, and local naming rows;*
2. *uniform Cauchy family stability, requiring every requested comparison to stay on the displayed family row* F *and the finite windows exposed by* S*;*
3. *regular rational readback exactness, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to read only the same windows and budgets already named by* F *and* S*;*
4. *dyadic tolerance exactness, requiring the row* D *to discharge each uniform Cauchy request before the seal row is visible;*
5. *Real and completion seal nonescape, requiring the row* E *to be reached only through* F, S, R, D *and the sibling* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *and* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *routes;*
6. *componentwise* hsame *transport and displayed* Cont *replay through* H *and* C*;*
7. Pkg *provenance and local naming through* P *and* N*;*
8. *refusal of quotient completions, selected limits, countable choice, host real equality, and ambient Real completeness as certificate data.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCompletionUp.UniformCauchyCompletionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3618.1. The visible coordinates are $F, S, R, D, E, H, C, P, N$, so every local obligation must be read from those rows.

The uniform Cauchy family is fixed by F and its finite observations are fixed by S. The regular rational readback R and tolerance ledger D are attached to those same windows, so a completion-facing request cannot exchange them for an external sequence, host equality, or selected limit representative.

The seal row is E. It is downstream of the family, StreamName, RegSeqRat, and DyadicRatCore rows, and its sibling route names the $\mathsf{CauchySequenceSpace}^{\uparrow}$ and $\mathsf{BishopCompletionTheorem}^{\uparrow}$ handoffs without adding quotient or choice data.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same handoff. Since no quotient completion, selected limit, countable choice, host real equality, or ambient Real completeness coordinate appears in the carrier, the listed rows exhaust the certificate surface. □

Theorem 2654.3 (Uniform Cauchy completion finite handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *packet factors through the finite route*

$$F \longmapsto S \longmapsto R \longmapsto D \longmapsto E, H, C, P, N.$$

Thus a consumer first reads the uniform Cauchy family, then the finite StreamName windows, then the RegSeqRat readback, then the DyadicRatCore tolerance ledger, and only then the Real and

completion seal. The route exports no quotient completion, selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $U = (F, S, R, D, E, H, C, P, N)$. The only family coordinate is F, and the only finite observation coordinate is S.

The readback and tolerance rows are R and D. They keep the completion-facing request on the same regular rational windows and dyadic tolerance ledger before any terminal seal can be read.

The endpoint row is E, with structural support from H, C, P, N. These rows hand the same finite packet to $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, and $\mathsf{BishopCompletionTheorem}^{\uparrow}$ consumers without adding quotient, selection, choice, host-equality, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing boundary while omitting the uniform Cauchy family row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, seal row, transport, replay, provenance, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCompletion}^{\uparrow}$ packet is generated from a uniform Cauchy family row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real and completion seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3618.1, Definition 3618.2, and Definition 3618.3 seed the finite uniform-Cauchy route from stream windows and regular rational readback to Real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform Cauchy family stability, finite-window readback, dyadic tolerance exactness, Real and completion seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyCompletionUp.

Chapter 2655

A Concrete Naming Certificate for $\mathsf{ArchimedeanCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of Archimedean ordered fields and regular Cauchy completions, the $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet records the finite naming surface that carries a dyadic ordered-field source into the Real and Completion route. It sits after the dyadic ordered core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the ordered cut comparison of chapter 3517, and the completion witness of chapter 3579. The packet names an Archimedean completion interface; it is not quotient equality of Cauchy streams, a selected host limit, a trichotomy oracle, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2655.1 (Archimedean completion carrier)**.** *An* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, K, R, D, O, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the dyadic ordered-field source row, including the finite addition, multiplication, order, unit, and sign-comparison data read from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
2. *K is the Archimedean cofinality row: every displayed field element is bounded by a named dyadic scale already present in the same packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row generated from the source and cofinality data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare finite approximants and dyadic scales.*
5. *O is the order-density comparison row linking the regular readback to the order-completion surface of chapter 3517.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, K, R, D, O have been displayed.*
7. *H records componentwise* hsame *transport for the ordered-field source, cofinality witness, regular readback, dyadic ledger, order comparison, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an Archimedean-completion request through F, K, R, D, O, E.*
9. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopOrderCompletion}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, K, R, D, O, E, H, C, P, N$ and the displayed replay path. It contains no coordinate for quotient stream equality, selected representatives, host-limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanCompletionUp.TasteGate.ArchimedeanCompletionTasteGate single carrier alignment`

Theorem 2655.2 (Archimedean completion NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier $A = (F, K, R, D, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the finite dyadic ordered-field packet F together with the Archimedean cofinality witness K;*
2. *regular-Cauchy readback through R and dyadic tolerance accounting through D;*
3. *order-density comparison through O, with no independent located-cut or trichotomy oracle;*
4. *terminal* $\mathsf{Real}^{\uparrow}$ *handoff through E, reached only after the source, cofinality, readback, ledger, and order-comparison rows have been displayed;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local obligation can be discharged by quotient Cauchy equality, selected host limits, countable choice, a detached order-completion theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3619.1. Its analytic rows are exactly the dyadic ordered-field source, the Archimedean cofinality witness, the regular-Cauchy readback, the dyadic tolerance ledger, the order-density comparison, and the terminal Real seal.

Carrier admission is the first row: F supplies the finite ordered-field operations and comparisons, while K supplies only the displayed cofinal dyadic scales. The readback row R and ledger row D then account for every regular-Cauchy and tolerance datum consumed by the packet.

The order-facing route is forced through O. It compares the same finite readback with the Bishop order-completion surface, but it cannot introduce a located-cut selector, trichotomy oracle, or host order equality because no such row belongs to the carrier.

Finally E, H, C, P, N route the terminal Real handoff and structural support. Since the carrier has no quotient stream equality, selected limit, countable-choice schedule, detached completion theorem, or ambient-completeness coordinate, none of those objects belongs to the NameCert obligation surface. □

Theorem 2655.3 (Archimedean completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier factors through the displayed dyadic, regular, and order-completion rows*

$$F, K \longmapsto R, D \longmapsto O \longmapsto E \longmapsto H, C, P, N.$$

The dyadic ordered-field source and Archimedean cofinality rows determine the regular-Cauchy readback; the dyadic ledger and order-density row compare that readback with the Bishop order-completion interface; the terminal Real seal is available only after those rows have been replayed.

Consequently a Real-facing consumer receives a finite NameCert route from dyadic ordered-field data to regular Cauchy and order-completion data. It receives no quotient stream equality, selected representative, host-limit operator, trichotomy oracle, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer that reaches the terminal Real row. In the carrier of Definition 3619.1, the only Real-facing coordinate is E, and E is placed after F, K, R, D, O.

The consumer must therefore read the dyadic ordered-field source and cofinality rows before any regular-Cauchy data is available. The readback row R and ledger row D expose the finite approximation route, while the order-density row O aligns that route with the Bishop order-completion packet.

The structural rows H, C, P, N then transport, replay, source, and locally certify the same route. They preserve the displayed handoff, but they cannot add quotient equality, selected limits, trichotomy, countable choice, detached completion, or ambient Real completeness because the accepted packet has no such coordinate. □

Falsifiable prediction: Every accepted $\mathsf{ArchimedeanCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ handoff while omitting the dyadic ordered-field source, Archimedean cofinality row, regular-Cauchy readback, dyadic tolerance ledger, order-density comparison, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: dyadicratcore, regseqrat, real, bishop-order-completion, cauchy-completeness-witness: no listed sibling supplies the whole $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet, because each sibling contributes only dyadic arithmetic, regular readback, Real sealing, order comparison, or completion witnessing while this chapter binds all of them with cofinality, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet is generated from a dyadic ordered-field source row, an Archimedean cofinality row, RegSeqRat readback, a DyadicRatCore tolerance ledger, a BishopOrderCompletion order-density comparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3619.1, Definition 3619.2, and Definition 3619.3 seed the finite Archimedean completion route from dyadic ordered-field source data through regular-Cauchy readback, order-density comparison, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ArchimedeanCompletionUp.TasteGate.ArchimedeanCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected host limits, trichotomy, countable choice, detached order completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinality routing, regular-Cauchy readback, dyadic ledger exactness, order-density comparison, Real handoff factorization, transport, replay, provenance, and local naming for BEDC.Derived.ArchimedeanCompletionUp.

Chapter 2656

A Concrete Naming Certificate for $\mathsf{RieszSpace}^{\uparrow}$

Following Riesz and Kantorovich, the $\mathsf{RieszSpace}^{\uparrow}$ packet records the finite BEDC naming surface for a real vector lattice with a displayed positive cone and norm-facing compatibility rows. It sits over the algebraic vector-space carrier of chapter 47, the join-meet order rows of chapter 55, the norm comparison surface of chapter 89, the positive scalar endpoint discipline of chapter 1587, and the real scalar seal of chapter 38. The packet names the BEDC-facing ordered-functional-analysis object; it is not a host order primitive, a supremum completion theorem, a Banach-lattice completeness claim, or a choice of lattice decompositions outside the displayed finite ledger.

Definition 2656.1 (RieszSpace carrier)**.** *A* $\mathsf{RieszSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (V, L, N, P, H, C, S)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming the carried vector endpoint, real scalar endpoint, zero vector, addition, additive inverse, scalar action, and finite vector-space law ledger.*
2. *L is the lattice row, recording join and meet operations on the same vector endpoint, the induced order-comparison rows, and the finite absorption, commutativity, associativity, and distributivity witnesses used by the local classifier.*
3. *N is the norm or seminorm compatibility row, read through* $\mathsf{Norm}^{\uparrow}$*, saying that the displayed norm endpoint respects the vector rows and the lattice decomposition rows already carried by V and L.*
4. *P is the positive-cone decomposition row, read through* $\mathsf{PositiveReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording positive and negative part formation, cone membership, scalar positivity, and the local identity route tying a vector to the difference of its displayed positive pieces.*
5. *H records componentwise* hsame *transport for the vector-space rows, lattice join and meet rows, norm or seminorm rows, positive-cone rows, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from a Riesz-space request through V, L, N, P, H back to the same finite ordered vector-lattice surface.*
7. *S is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$ *row over* $\mathsf{RieszSpace}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Lattice}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{PositiveReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

The classifier compares accepted packets only through V, L, N, P, H, C, S. It has no coordinate for an ambient order relation, a host supremum construction, a selected order completion, Banach-lattice completeness, pointwise function models, or a quotient by order ideals.

Lean4 ✓ `BEDC.Derived.RieszSpaceUp.RieszSpaceUpTasteGate single carrier alignment`

Theorem 2656.2 (RieszSpace NameCert obligations)**.** *For every accepted* $\mathsf{RieszSpace}^{\uparrow}$ *carrier* $R = (V, L, N, P, H, C, S)$*, the local* $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, vector-space row compatibility, lattice join and meet exactness, norm or seminorm compatibility, positive-cone decomposition, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A downstream consumer may read the common vector endpoint, the real scalar endpoint, the join and meet operations, the norm-facing comparison row, and the positive-cone ledger, but it receives no host order primitive, no chosen order completion, no Banach-lattice completeness theorem, and no pointwise function representation.*

Proof. Carrier admission is projection from Definition 3620.1. The packet displays exactly the vector-space row V, the lattice row L, the norm or seminorm compatibility row N, the positive-cone decomposition row P, and the structural rows H, C, S.

The vector-space obligations are inherited from the carried $\mathsf{VecSpace}^{\uparrow}$ row, while the lattice obligations are read only through the displayed join and meet ledger. Thus the same vector endpoint is the domain of addition, scalar multiplication, join, meet, and order comparison.

The norm-facing and positive-cone obligations are finite rows. The row N records compatibility with the vector and lattice decompositions, and P records positive and negative part formation using the $\mathsf{PositiveReal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar seals.

Finally H, C, S transport, replay, source, and locally certify only the displayed coordinates. Since the carrier has no ambient order coordinate, no completion selector, no Banach-lattice completeness row, and no pointwise function model, none of those objects can be exported by the local certificate. □

Theorem 2656.3 (RieszSpace lattice-norm compatibility)**.** *If two accepted* $\mathsf{RieszSpace}^{\uparrow}$ *packets are related by the classifier, then their vector-space rows, lattice join and meet rows, norm or seminorm compatibility rows, and positive-cone decomposition rows are related by the componentwise* hsame *transport recorded in H. Consequently every downstream norm-facing or order-facing read factors through the displayed vector-lattice ledger and never through a host order equality or a hidden normed-lattice quotient.*

Proof. Unfold the classifier of Definition 3620.1. Its comparison coordinates are exactly the vector-space row, lattice join and meet row, norm or seminorm compatibility row, positive-cone row, replay row, provenance row, and local naming row.

The vector component transports through the $\mathsf{VecSpace}^{\uparrow}$ comparison, and the lattice component transports through the displayed join and meet comparisons. The norm or seminorm row then moves along the same transported vector and lattice data.

The positive-cone row is carried componentwise as well: positive part, negative part, cone membership, scalar positivity, and reconstruction of the original vector all remain inside the visible ledger. Hence every norm-facing or order-facing consumer reads a transported finite certificate, not a host order equality or a hidden quotient construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RieszSpace}^{\uparrow}$ packet is generated from vector-space rows, lattice join and meet rows, norm or seminorm compatibility rows, positive-cone decomposition data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RieszSpace}\uparrow}$, seed)

Scope closed: Definition 3620.1, Definition 3620.2, and Definition 3620.3 seed the finite ordered vector-lattice certificate over VecSpaceUp, LatticeUp, NormUp, PositiveRealUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RieszSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host order primitives, supremum completion, Banach-lattice completeness, pointwise function representation, quotient by order ideals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vector-space compatibility, lattice join and meet exactness, norm or seminorm compatibility, positive-cone decomposition, transport, replay, provenance, and local naming for BEDC.Derived.RieszSpaceUp.

Chapter 2657

A Concrete Naming Certificate for ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$

Following Arzela and Ascoli, the ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ packet names the finite-window bridge between the diagonal extraction surface of chapter 2219 and the compact family interface of chapter 412. It consumes the equicontinuity ledger of chapter 413, the compact source rows of chapter 129, the target completion rows of chapter 128, the finite window names of chapter 126, the rational readback rows of chapter 301, and the pointwise seal of chapter 38. The chapter records a finite BEDC NameCert bridge; it does not assert an ambient compact-open topology, a host sequence-space compactness theorem, or a choice-selected subsequence.

Definition 2657.1 (Arzela-Ascoli finite-window diagonal bridge)**.** *An* ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, E, Y, D, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *K records the compact source net rows read from* CompactMetric$^{\uparrow}$*, including the finite probe windows and coverage witnesses consumed by the diagonal route;*
2. *E records the* EquicontinuityFamily$^{\uparrow}$ *ledger that binds every requested target tolerance to a displayed source-radius row uniformly over the family ledger;*
3. *Y records the* CompleteMetric$^{\uparrow}$ *target surface through which finite-window target reads are sealed;*
4. *D records the accepted* AscoliDiagonal$^{\uparrow}$ *carrier whose diagonal extraction rows are being exposed to the Arzela-Ascoli consumer;*
5. *W records the finite diagonal* StreamName$^{\uparrow}$ *windows, with each window naming the source-net points, family indices, and target reads currently inspected;*
6. *R records* RegSeqRat$^{\uparrow}$ *readback rows for the rational tolerances and target-distance witnesses visible on W;*
7. *S records the* Real$^{\uparrow}$ *seal that repacks each finite readback into the target-side pointwise boundary;*
8. *H records componentwise* hsame *transport for K, E, Y, D, W, R, S and the structural rows;*
9. *C records displayed* Cont *replay through AscoliDiagonal, EquicontinuityFamily, CompactMetric, CompleteMetric, StreamName, RegSeqRat, Real, and local NameCert reads;*

10. *P is the* Pkg *provenance over* ArzelaAscoliFiniteWindowDiagonalBridge$^\uparrow$, AscoliDiagonal$^\uparrow$, ArzelaAscoli$^\uparrow$, EquicontinuityFamily$^\uparrow$, CompactMetric$^\uparrow$, CompleteMetric$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^\uparrow}$ *row.*

The carrier exports only the displayed finite-window diagonal bridge data. It has no coordinate for an unbounded subsequence selector, an ambient compact-open topology, or an open-cover compactness principle.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteWindowDiagonalBridgeUp.ArzelaAscoliFiniteWindowDiagonalBridgeTas`
`single carrier alignment`

Theorem 2657.2 (Finite-window diagonal bridge route)**.** *Every bridge read from an accepted* ArzelaAscoliFiniteWindowDiagonalBridge$^\uparrow$ *carrier factors through the finite route*

$$\text{AscoliDiagonal}^\uparrow \longrightarrow (\text{EquicontinuityFamily}^\uparrow, \text{CompactMetric}^\uparrow) \longrightarrow \text{CompleteMetric}^\uparrow \longrightarrow \text{StreamName}^\uparrow \longrightarrow \text{RegSeqRa}$$

The route first reads the accepted AscoliDiagonal carrier, then the compact source nets and equicontinuity ledger, then the complete target surface, then the finite diagonal StreamName windows, then RegSeqRat readback and the Real seal. The exported surface is a finite NameCert bridge and no choice-selected subsequence or host function-space compactness object.

Proof. Let $B = (K, E, Y, D, W, R, S, H, C, P, N)$ be an accepted carrier. Project D first. Its accepted AscoliDiagonal$^\uparrow$ row fixes the diagonal extraction surface before any Arzela-Ascoli consumer can inspect a target read.

Next project K and E. The compact source row supplies finite net windows, and the equicontinuity-family row supplies the shared tolerance-to-radius ledger used by those windows.

Project Y to read the complete-metric target boundary. The bridge can then inspect only the finite StreamName$^\uparrow$ windows recorded in W, so each target read is tied to displayed source-net points and displayed family indices.

The readback row R records the corresponding RegSeqRat$^\uparrow$ tolerance and distance data, and the seal row S repacks the finite readback as the visible Real$^\uparrow$ boundary. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve exactly those rows.

Since the carrier contains no unbounded subsequence selector, no ambient compact-open topology, and no host function-space compactness coordinate, every bridge read factors through the displayed finite route and exports nothing beyond it. □

Closure status for NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^\uparrow}$ **·**

Constructive story: An ArzelaAscoliFiniteWindowDiagonalBridge$^\uparrow$ packet is generated from an accepted AscoliDiagonal carrier, compact source-net rows, equicontinuity-family rows, complete-metric target rows, finite StreamName diagonal windows, RegSeqRat readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^\uparrow}$, seed)

Scope closed: Definition 3621.1 and Definition 3621.2 seed the finite-window diagonal bridge from AscoliDiagonal data to the Arzela-Ascoli compact family interface.

Formal verification: FormalStatus(NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a choice-selected subsequence, ambient compact-open topology, open-cover compactness, host sequence-space compactness, or a standard bridge.

Upgrade path: Obligation closure requires Lean carrier admission, classifier transport, finite-window route extraction, readback sealing, and non-escape rows for BEDC.Derived.ArzelaAscoliFiniteWindowDiagonalBridgeUp.

Chapter 2658

A Concrete Naming Certificate for $\mathsf{BishopCauchySequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchySequence}^{\uparrow}$ packet records the textbook sequence-level Cauchy surface that mediates between regular rational names and the sealed Real object. It sits after the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational name packet of chapter 301, and the uniqueness route of chapter 3390, while feeding the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite carrier for Cauchy sequence evidence; it does not assert quotient streams, selected limits, countable choice, or ambient Real completeness.

Definition 2658.1 (Bishop Cauchy sequence carrier)**.** *A* $\mathsf{BishopCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, D, X, U, R, H, C, P, N)$$

with the following visible rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the requested rational sequence entries;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, comparison budget, and rational-distance rows used by the Cauchy test;*
3. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row that certifies the sequence entries respect the displayed tolerance discipline;*
4. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *comparison row used only to compare accepted regular names at the same displayed tolerance surface;*
5. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, D, X, U have been displayed;*
6. *H stores componentwise* hsame *transport for the stream window, dyadic ledger, regularity row, uniqueness comparison, seal row, continuation route, package row, and local naming row;*
7. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto X \longmapsto U \longmapsto R;$$

8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $S, D, X, U, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, ambient Real completeness, or a host equality oracle for Real values.

Lean4 ✓ `BEDC.Derived.BishopCauchySequenceUp.BishopCauchySequenceUpTasteGate single carrier alignment`

Theorem 2658.2 (Bishop Cauchy sequence NameCert obligations)**.** *For an accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *carrier*

$$B = (S, D, X, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row,* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *comparison row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under the same displayed rows; tolerance exactness through D; regularity handoff through X; uniqueness comparison through U; and Real-seal boundary control through R.*

Proof. Carrier admission is projection from Definition 3622.1. The projection exposes exactly the stream window, dyadic tolerance ledger, regular rational sequence evidence, uniqueness comparison, Real seal, and structural rows.

Classifier stability follows row by row because the classifier has no coordinate beyond $S, D, X, U, R, H, C, P, N$. Tolerance exactness is the projection of D, and regularity handoff is the projection of X with the same displayed tolerance row.

The uniqueness row U compares only accepted regular names at the displayed surface. The Real seal row R is downstream from S, D, X, U, so no consumer can read a sealed Real object without first passing through the finite Bishop Cauchy sequence packet. □

Theorem 2658.3 (Bishop Cauchy sequence regularity handoff)**.** *Every accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *packet hands its stream-window and tolerance data to a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row before the uniqueness comparison or Real seal is available. The handoff route is the finite chain*

$$S, D \longmapsto X \longmapsto U \longmapsto R,$$

preserved by H, C, P, N and not by quotient stream equality, a selected limiting object, countable choice, or an ambient completeness theorem.

Proof. Project the carrier and read the continuation route C. The route opens the finite stream window S and dyadic tolerance ledger D before the regularity row X.

The comparison row U is downstream from the regularity row. Since U is a $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ row, it compares accepted regular names through the displayed sequence and tolerance data rather than through a quotient stream.

The Real seal row R occurs at the end of the displayed route. The transport, provenance, and naming rows preserve that order, while no carrier coordinate supplies a selected limit, countable-choice principle, or ambient completeness argument. □

Theorem 2658.4 (Bishop Cauchy sequence Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *packet factors through the finite rows S, D, X, U before it reaches the seal row R. The packet therefore exposes a sequence-level route from rational windows and dyadic tolerances to the Real seal, while withholding quotient streams, selected limits, countable choice, and ambient Real completeness.*

Proof. Start from a Real-facing read and project the carrier. The only seal row is R, and the continuation route places it after the stream window, tolerance ledger, regularity row, and uniqueness comparison.

Use Definition 3622.3 to bind the regularity row to the same stream and dyadic tolerance data. Then use the uniqueness row U only as a displayed comparison between accepted regular names.

The coordinate list of Definition 3622.1 exhausts the packet. Since it contains no quotient stream, selected limit, countable choice, or ambient-completeness coordinate, no consumer can derive such data from the Real seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchySequence}^{\uparrow}$ packet is generated from finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat regularity evidence, a RegularCauchyUniqueness comparison row, a Real seal, transport data, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3622.1, Definition 3622.2, Definition 3622.3, and Definition 3622.4 name the L10 sequence-level carrier between regular rational streams and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, tolerance exactness, regularity handoff, uniqueness comparison, and Real-seal boundary rows for the displayed Bishop Cauchy sequence packet.

Chapter 2659

A Concrete Naming Certificate for $\mathsf{MeirKeelerContraction}^{\uparrow}$

Following Meir and Keeler's fixed point theorem and Bishop–Bridges constructive metric analysis, the $\mathsf{MeirKeelerContraction}^{\uparrow}$ packet records the finite epsilon-to-delta contraction ledger that sits between ordinary contraction data and completion-facing fixed-point consumers. It reads the metric carrier of chapter 57, the fixed-point route of chapter 3607, the iteration ledger of chapter 2383, the endpoint-admission surface of chapter 128, and the Banach fixed-point packet of chapter 396. It also sits beside the Lipschitz and nonexpansive surfaces of chapter 410, chapter 2328, and chapter 2105, while refusing to replace the Meir–Keeler ledger by a single host Lipschitz constant.

Definition 2659.1 (Meir–Keeler contraction carrier)**.** *A* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, T, E, I, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{Met}^{\uparrow}$ *carrier row for the source points.*
2. *d is the metric-distance readback row used by all comparison requests.*
3. *T is the self-map graph row for the candidate contraction.*
4. *E is the Meir–Keeler epsilon-to-delta contraction ledger: for each displayed positive tolerance request it stores a displayed response window forcing a strictly smaller image distance whenever the source distance lies in the named annulus.*
5. *I is the fixed-point iteration row, recording the finite Picard-style orbit segments and adjacent comparison reads used by downstream consumers.*
6. *L is the Cauchy handoff ledger, listing the finite tail requests that the Meir–Keeler row transfers to the complete-metric endpoint surface.*
7. *H records componentwise* hsame *transport for metric rows, distance reads, self-map graph rows, epsilon-to-delta ledger rows, iteration rows, Cauchy handoff rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a contraction request through $X, d, T, E, I, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MeirKeelerContraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContractionPrinciple}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, d, T, E, I, L, H, C, P, N$. It has no coordinate for a global Lipschitz constant, a selected endpoint, quotient completion equality, host function extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MeirKeelerContractionUp.MeirKeelerContractionTasteGate single carrier alignment`

Theorem 2659.2 (Meir–Keeler contraction NameCert obligations)**.** *For an accepted* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier $M = (X, d, T, E, I, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$ *surface consists exactly of metric-carrier admission, distance-readback visibility, self-map graph admission, epsilon-to-delta ledger exactness, fixed-point iteration visibility, Cauchy handoff visibility, componentwise transport, displayed replay, provenance, and local naming. These obligations bind the Meir–Keeler route without adding a global contraction constant, endpoint selection, quotient completion equality, host fixed-point equality, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 3623.1. The displayed rows $X, d, T, E, I, L, H, C, P, N$ are the whole local surface, so every obligation must be one of their admissions, transports, or replay checks.

The contraction content is carried by T and E. The self-map row fixes the map being iterated, while the epsilon-to-delta row supplies the strict Meir–Keeler comparison at each displayed tolerance window. No single Lipschitz bound is extracted from this row.

The iteration and completion-facing content is carried by I and L. These rows expose finite orbit segments and finite tail requests before any complete-metric endpoint can be consumed by another chapter.

Finally H, C, P, N keep the same rows stable under classifier transport, replay, provenance, and local naming. Since the excluded host objects are not carrier coordinates, the NameCert surface cannot export them. □

Theorem 2659.3 (Meir–Keeler contraction Cauchy handoff)**.** *Every fixed-point or completion-facing read of an accepted* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow},\ d,\ T,\ E,\ \mathsf{FixedPointIteration}^{\uparrow},\ L,\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The read uses the Meir–Keeler epsilon-to-delta ledger to feed finite iteration-tail requests into the complete-metric endpoint surface. It never exposes a selected limit, a quotient representative, a host global Lipschitz constant, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the metric and map rows. A consumer first reads X, d, T, so the source carrier, comparison relation, and self-map graph are fixed before the Meir–Keeler ledger is inspected.

Next consume E. The epsilon-to-delta ledger answers each displayed tolerance request by a finite annulus response, which is the contraction data available to the iteration route. This is weaker than a global Lipschitz constant but strong enough for the displayed tail comparisons.

Then read I and L. The iteration row supplies the finite orbit comparisons, and the Cauchy handoff row packages the resulting tail requests for the $\mathsf{CompleteMetric}^{\uparrow}$ endpoint surface.

The structural rows H, C, P, N transport and replay the same route. Because no endpoint selector, quotient equality, host Lipschitz constant, or ambient completeness row occurs in the carrier, every downstream read remains inside the finite handoff just described. □

Falsifiable prediction: Every accepted $\mathsf{MeirKeelerContraction}^{\uparrow}$ read factors through one MetricUp carrier row, one distance readback row, one self-map row, one Meir–Keeler epsilon-to-delta ledger, one fixed-point iteration row, one Cauchy handoff ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any completion-facing fixed-point consumer can read it.

Independence witness: metric, contraction-principle, fixed-point-iteration, completemetric, banach-fixed-point, lipschitzmap, regular-cauchy-nonexpansive-map: none is bijective with $\mathsf{MeirKeelerContraction}^{\uparrow}$ because the packet jointly stores the metric carrier, self-map graph, epsilon-to-delta contraction ledger, finite iteration route, Cauchy handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{MeirKeelerContraction}^{\uparrow}$ packet is generated from a MetricUp carrier, distance readback, self-map graph, Meir–Keeler epsilon-to-delta contraction ledger, fixed-point iteration row, Cauchy handoff ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3623.1, Definition 3623.2, and Definition 3623.3 seed the finite Meir–Keeler contraction packet and its completion-facing Cauchy handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global Lipschitz constant, selected endpoint, quotient completion equality, host fixed-point equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, epsilon-to-delta ledger exactness, fixed-point iteration visibility, Cauchy handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.MeirKeelerContractionUp.

Chapter 2660

A Concrete Naming Certificate for $\mathsf{IntervalHalvingModulus}^{\uparrow}$

Beginning from Bishop–Bridges constructive interval bisection, the $\mathsf{IntervalHalvingModulus}^{\uparrow}$ packet records the finite halving schedule used by L10 interval and Cauchy constructions before any completed point or ambient real-completeness principle is exposed. It consumes the located interval surface of chapter 1709, the dyadic tolerance ladder of chapter 3525, the finite observation windows of chapter 126, the regular located-rational modulus route of chapter 3532, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet gives bisection, nested-interval, and regular-Cauchy extraction chapters a shared finite modulus object rather than a compactness or choice principle.

Definition 2660.1 (Interval halving modulus carrier)**.** *An* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, L, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row. It records the inspected interval window, endpoint location data, inclusion rows, and nonempty finite evidence visible to a halving consumer.*
2. *L is the* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *row. It records the finite tolerance requests and the displayed halving relation between successive dyadic scales.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names the finite interval, endpoint, midpoint, and tolerance observations that may be opened by the route.*
4. *R is the* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *handoff row. It records the located-rational regular Cauchy modulus extracted from scheduled interval halves and tolerance requests.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for midpoint formation, endpoint comparison, diameter comparison, and dyadic tolerance readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after I, L, S, R, D have displayed the finite halving route.*
7. *H records componentwise* hsame *transport for interval windows, tolerance ladders, stream schedules, located-modulus handoffs, dyadic ledgers, real seals, replay rows, provenance, and local naming.*

8. *C records displayed* Cont *replay from an interval-halving request through I, L, S, R, D, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{IntervalHalvingModulus}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicToleranceLadder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit, countable choice, trichotomy, quotient interval equality, host interval equality, or bridge checking.*

Lean4 ✓ `BEDC.Derived.IntervalHalvingModulusUp.TasteGate.IntervalHalvingModulusTasteGate single carrier alignment`

Theorem 2660.2 (Interval halving modulus NameCert obligations)**.** *For every accepted* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier*

$$M = (I, L, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, dyadic tolerance ladder L, finite stream schedule S, regular located-rational modulus handoff R, dyadic arithmetic ledger D, real-seal boundary E, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3624.1. The projection exposes one finite interval row, one dyadic tolerance ladder, one stream schedule, one located-rational modulus handoff, one dyadic arithmetic ledger, one real-seal boundary, and the structural rows.

The interval and tolerance obligations are I and L. The finite schedule S states which interval endpoints, midpoints, and tolerance requests are opened, while D supplies the arithmetic comparisons needed by those observations.

The handoff row R is downstream from the displayed interval, ladder, schedule, and dyadic rows. It exports only the regular located-rational modulus read from those finite observations.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no extra coordinate for completeness, selected limits, countable choice, trichotomy, quotient interval equality, or host interval equality, the displayed rows exhaust the NameCert obligations. □

Theorem 2660.3 (Interval halving modulus ladder handoff)**.** *Let $M = (I, L, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier. Every bisection, nested-interval, or regular-Cauchy consumer of M factors through the finite route*

$$I, L \longmapsto S, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the consumer receives a displayed halving modulus and located-rational Cauchy handoff before any $\mathsf{Real}^{\uparrow}$ *seal can be read.*

Proof. Start with the located interval row I and the dyadic tolerance ladder L. They provide the only interval window and halving-scale data available to the consumer.

The schedule row S selects the finite observations to open, and the dyadic ledger D checks midpoint formation, endpoint order, diameter bounds, and tolerance comparisons for those observations.

The row R repacks exactly this scheduled data as the regular located-rational modulus. The real-facing row E is therefore downstream from the finite halving route rather than an external source of completeness.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Hence bisection, nested-interval, and regular-Cauchy consumers cannot leave the finite ladder handoff without leaving the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalHalvingModulus}^{\uparrow}$ packet is generated from located interval rows, dyadic tolerance ladders, StreamName schedules, regular located-rational modulus handoffs, dyadic arithmetic ledgers, Real seal boundaries, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3624.1, Definition 3624.2, and Definition 3624.3 seed the finite halving schedule used by interval bisection, nested interval, and regular-Cauchy extraction routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntervalHalvingModulusUp.TasteGate.IntervalHalvingModulusTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter refuses $\mathsf{Real}^{\uparrow}$ completeness, selected limits, countable choice, trichotomy, quotient interval equality, host interval equality, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exposure, tolerance-ladder exactness, schedule compatibility, dyadic midpoint and diameter exactness, regular located-rational modulus handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IntervalHalvingModulusUp.

Chapter 2661

A Concrete Naming Certificate for $\mathsf{PostMedianGateShell}^{\uparrow}$

The $\mathsf{PostMedianGateShell}^{\uparrow}$ packet realizes the post-median gate-shell object of Definition 9462.1490 as finite BEDC NameCert data. It reads the six-boundary-tile vision through codon-shell rows, Boolean-coordinate rows, spectral-optimum rows, first-exit rows, transport, replay, provenance, and local naming. The packet is a science-facing finite certificate: it does not treat the observed shell as a universal genetic-code law, an unbounded evolutionary optimizer, or an oracle for unseen codon tables.

Falsifiable prediction: Every accepted $\mathsf{PostMedianGateShell}^{\uparrow}$ read must display the six shell rows CAR, CCR, GUR, the median gate M, the first-exit target layer, and the retained CUR exit-closing row before any spectral-optimum claim is exported.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, punctured-cube-skeleton, wrr-state-phase-space, median-closure-structure: six-boundary-tile coverage names the codon-window partition, window-coverage trigger algebra names activation gates, the punctured-cube skeleton names cube geometry, WRR state phase space names replay states, and median closure structure names the median set. None is bijective to $\mathsf{PostMedianGateShell}^{\uparrow}$ because this packet binds post-median shell rows, Boolean coordinates, first-exit rows, spectral optimality, transport, replay, provenance, and local naming in one accepted surface.

The shell route consumes the finite boundary-tile coverage row of chapter 3470 and the trigger-algebra row of chapter 3478 before the median gate and first-exit rows are read. These sibling rows provide the boundary support and activation surface; the present packet adds the post-median shell ledger, spectral row, first-exit row, transport, replay, provenance, and local naming.

Definition 2661.1 (Post-median gate shell carrier)**.** *A* $\mathsf{PostMedianGateShell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, B, S, F, T, C, P, N)$$

with the following displayed rows:

1. *K is the codon-shell row. It records the median gate M, the shell rows* CAR*,* CCR*,* GUR*, the retained* CUR *exit-closing row, and the finite read request.*
2. *B is the Boolean-coordinate row. It records the six-boundary tile coordinates used to read the shell as a finite yes-or-no window rather than as a host-level biological assertion.*
3. *S is the spectral-optimum row. It records the killed-walk Perron-root comparison for the displayed shell rows and refuses to export an optimizer outside the finite comparison table.*

4. *F is the first-exit row. It records that* CAR, CCR, *and* GUR *are read as the top first-exit target layer after the median gate.*
5. *T records componentwise* hsame *transport for codon-shell rows, Boolean coordinates, spectral rows, first-exit rows, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from the post-median gate-shell request through K, B, S, F, T, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{PostMedianGateShell}^{\uparrow}$*, the six-boundary-tile coverage, the window-coverage trigger algebra, the punctured-cube skeleton, the WRR state phase space, the median closure structure,* BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eight rows. It has no coordinate for arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, host biochemical causality, or bridge checking.

Lean4 ✓ `BEDC.Derived.PostMedianGateShellUp.PostMedianGateShellTasteGate single carrier alignment`

Theorem 2661.2 (Post-median gate shell NameCert obligations)**.** *For every accepted* $\mathsf{PostMedianGateShell}^{\uparrow}$ *carrier*

$$G = (K, B, S, F, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$ *surface consists exactly of carrier admission for the codon-shell row K, Boolean-coordinate row B, spectral-optimum row S, first-exit row F, componentwise transport T, continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3625.1. The projection exposes one finite codon-shell row, one Boolean-coordinate row, one spectral row, one first-exit row, and the structural rows.

The codon and coordinate obligations are K and B. They force every read to display the six shell rows and the finite boundary-tile coordinates before the shell can be compared spectrally.

The spectral and first-exit obligations are S and F. The spectral row is tied to the displayed finite comparison table, while the first-exit row ties CAR, CCR, and GUR to the post-median target layer.

Finally T, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, or host biochemical causality, the displayed rows exhaust the NameCert obligations. □

Theorem 2661.3 (Post-median gate shell accepted-read frontier)**.** *Every accepted read of a* $\mathsf{PostMedianGateShell}^{\uparrow}$ *packet factors through*

$$K, B \longmapsto S, F \longmapsto T, C, P, N.$$

Thus a consumer may export the post-median shell only after the six shell rows, Boolean-coordinate window, finite spectral comparison, and first-exit target layer have all been displayed.

Proof. Begin with the shell row K and Boolean-coordinate row B. These rows display the median gate, CAR, CCR, GUR, CUR, and the finite six-boundary coordinates available to the read.

The spectral row S then records the finite Perron-root comparison for that displayed shell. The first-exit row F records the route from the median gate to the top target layer, so the consumer cannot replace the shell by an unrelated codon orbit.

The rows T, C, P, N preserve the same finite surface under transport, replay, provenance, and local naming. Any read that claims an unbounded optimizer, unseen-window extrapolation, global evolutionary selection, or host biochemical causality leaves the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$.

Constructive story: A $\mathsf{PostMedianGateShell}^{\uparrow}$ packet is generated from finite codon-shell rows, Boolean-coordinate rows, spectral-optimum rows, first-exit rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3625.1, Definition 3625.2, and Definition 3625.3 seed the finite post-median shell certificate over displayed codon, Boolean-coordinate, spectral, and first-exit rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, host biochemical causality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, six-row shell exposure, Boolean-coordinate exactness, spectral-optimum exactness, first-exit exactness, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.PostMedianGateShellUp.TasteGate.

Chapter 2662

A Concrete Naming Certificate for MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$

The MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ packet realizes the dependent-codomain barrier of subsection 9386.7.1 as finite NameCert data. It reads the inversion boundary of chapter 781, the closure-preservation packet of chapter 804, the subject-reduction boundary of chapter 721, and the closed-normal consistency boundary of chapter 733. The packet names the stability row required before a dependent-codomain application branch can be routed to subject reduction; it does not assert unconditional subject reduction.

Definition 2662.1 (MetaCIC dependent-codomain stability boundary carrier)**.** *A* MetaCICDependentCodomainSta *carrier is a finite* BHist *packet*

$$S = (A, D, R, I, V, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the application-argument reduction row whose codomain may observe the reducing argument.*
2. *D is the dependent-codomain readback row, displaying the pre-step and post-step codomain observations.*
3. *R is the subject-reduction discharge request row that asks for the application branch of the MetaCIC interface.*
4. *I is the dependent-codomain inversion-boundary row.*
5. *V is the closure-preservation transport row available before typing preservation is requested.*
6. *Q is the obstruction row recording that subject reduction is unavailable unless the dependent-codomain stability row is supplied.*
7. *L is the stability ledger: it records the exact row that a later bundle witness must fill to make the dependent codomain readback stable under the application-argument step.*
8. *H records componentwise* hsame *transport for application, codomain readback, discharge request, inversion, closure-preservation, obstruction, stability-ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the MetaCIC open-problem boundary through $A, D, R, I, V, Q, L, H, P, N$.*

10. *P is the* Pkg *provenance over* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$, DependentCodomainInversionBoun DependentCodomainClosurePreservation$^{\uparrow}$, MetaCICSubjectReductionBoundaryUp, ClosedNormalConsistencyBound SubjectReductionDischargeBundle, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, D, R, I, V, Q, L, H, C, P, N$. It has no coordinate for unconditional subject reduction, conversion completeness, closed consistency, ambient normalization, or a hidden discharge witness.

Theorem 2662.2 (MetaCIC dependent-codomain stability boundary obligations)**.** *For an accepted* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ *carrier $S = (A, D, R, I, V, Q, L, H, C, P, N)$, the local* NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$ *surface consists exactly of application-argument reduction admission, dependent-codomain readback visibility, subject-reduction discharge-request visibility, inversion-boundary routing, closure-preservation transport, obstruction recording, stability-ledger exactness, componentwise transport, displayed replay, provenance, and local naming. A downstream Lean follow-up must supply the corresponding 'Nontrivial' and 'FieldFaithful' instances before this boundary can be consumed as a checked discharge row.*

Lean4 ✓ `BEDC.Derived.MetaCICDependentCodomainStabilityBoundaryUp.TasteGate.MetaCICDependentCodomainStab`
`single carrier alignment`

Proof. Project the carrier. The rows $A, D, R, I, V, Q, L, H, C, P, N$ are the only displayed coordinates, so the local NameCert obligations are their admission, compatibility, and replay checks.

The codomain-sensitive part is A, D, R. These rows record that the reduction branch is an application-argument branch and that the codomain readback can observe the reducing argument before the subject-reduction request is made.

The boundary part is I, V, Q, L. The inversion row and closure-preservation row provide the available finite data, the obstruction row records why they are insufficient by themselves, and the stability ledger names the missing row rather than discharging it.

Finally H, C, P, N transport, replay, source, and name the same finite packet. The required Lean 'Nontrivial' and 'FieldFaithful' instances belong to the later bundle witness because no hidden instance coordinate occurs in this carrier. □

Theorem 2662.3 (MetaCIC dependent-codomain stability handoff)**.** *Every subject-reduction-facing read of an accepted* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ *carrier factors through the finite route*

DependentCodomainInversionBoundary$^{\uparrow}$, DependentCodomainClosurePreservation$^{\uparrow}$, MetaCICSubjectReductionBound

The handoff exposes the stability ledger that a later bundle witness must fill. It does not turn the dependent-codomain application case into unconditional subject reduction and does not promote closed-normal consistency to full closed consistency.

Proof. Begin with A, D, R. The read is subject-reduction-facing only because it has first displayed the application-argument step, the dependent-codomain readback, and the discharge request.

Next consume I, V, Q. The inversion boundary and closure-preservation packet give the available codomain and closedness data, while the obstruction row records that this data does not yet type the post-step codomain.

The ledger L is then the only exported positive handoff. It names the dependent-codomain stability row that a later bundle witness must supply, including the instance obligations recorded in the carrier theorem.

Finally H, C, P, N replay the route without adding theorem strength. Thus a consumer receives a precise boundary packet and a setup-facing row, not an unconditional subject-reduction theorem. □

Theorem 2662.4 (AppArgTypeStable boundary row)**.** *For an accepted* MetaCICDependentCodomainStabilityBoun *carrier* $S = (A, D, R, I, V, Q, L, H, C, P, N)$*, the ledger row* L *is the AppArgTypeStable-facing boundary: any application-argument reduction read must display the pre-step and post-step codomain observations in* D *before it can feed the subject-reduction discharge request* R*.*

Proof. Project the rows A, D, R, L from the accepted carrier. The row A identifies the read as an application-argument reduction, while D displays the dependent codomain before and after that step.

Apply Definition 3626.3. The only positive row exported toward the subject-reduction request is L, and the handoff reaches it only after A, D, R have been displayed.

Thus L names the AppArgTypeStable-facing boundary for the later discharge witness. It records the required codomain-stability row without claiming unconditional subject reduction. □

Falsifiable prediction: Any MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ packet lacking the dependent-codomain stability row L cannot discharge the application-argument branch of subject reduction through the displayed MetaCIC discharge interface.

Independence witness: dependentcodomaininversionboundary, dependentcodomainclosurepreservation, metacicsubjectreductionboundary, closed-normal-consistency-boundary, metacic-open-problems: none is bijective with MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ because this packet jointly stores application-argument reduction, dependent codomain readback, discharge request, inversion boundary, closure-preservation transport, obstruction row, stability ledger, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$.

Constructive story: A MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ packet is generated from application-argument reduction, dependent-codomain readback, subject-reduction discharge request, inversion-boundary, closure-preservation, obstruction, stability-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3626.1, Definition 3626.2, and Definition 3626.3 seed the finite dependent-codomain stability boundary packet for MetaCIC subject-reduction consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unconditional subject reduction, conversion completeness, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked application-argument admission, dependent-codomain readback visibility, inversion-boundary routing, closure-preservation transport, obstruction recording, stability-ledger exactness, and the required witness instances for BEDC.Derived.MetaCICDependentCodomainStabilityBoundaryUp.

Chapter 2663

A Concrete Naming Certificate for $\mathsf{DyadicSandwich}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicSandwich}^{\uparrow}$ packet records the finite squeeze surface by which lower and upper dyadic rows trap a regular rational tail before a Real-facing consumer can read the seal. It sits over the dyadic tolerance core of chapter 291, the finite schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tail ball boundary of chapter 1633, and the terminal Real packet of chapter 38. The packet names the squeeze route as finite certificate data; it is not a quotient-real equality theorem, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 2663.1 (DyadicSandwich carrier)**.** *A* $\mathsf{DyadicSandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic row, listing the inspected lower endpoints and their finite-window order witnesses.*
2. *U is the upper dyadic row, listing the inspected upper endpoints and the matching containment witnesses.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing only the windows on which L and U are compared.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic radii, order comparisons, and residual budgets for the sandwich.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-tail row whose displayed values remain between the lower and upper dyadic rows at the scheduled windows.*
6. *H records componentwise* hsame *transport for lower rows, upper rows, schedules, dyadic ledgers, regular tails, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from the dyadic lower and upper rows through the schedule, tolerance ledger, regular tail, and Real-facing handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicSandwich}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicTailBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $L, U, S, D, R, H, C, P, N$. It has no coordinate for quotient equality of regular names, selected limits, trichotomy, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 2663.2 (DyadicSandwich NameCert obligations)**.** *For every accepted* $\mathsf{DyadicSandwich}^{\uparrow}$ *carrier* $Q = (L, U, S, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the lower dyadic row, upper dyadic row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat regular-tail row, transport, replay, provenance, and local naming;*
2. *lower/upper order, requiring each displayed lower row to lie below the matching upper row through the dyadic comparison data in D;*
3. *tail containment, requiring the regular-tail values in R to remain inside the displayed dyadic interval at every window named by S;*
4. *classifier stability, requiring any accepted comparison of packets to transport $L, U, S, D, R, H, C, P, N$ componentwise and to preserve the displayed containment route;*
5. *squeeze handoff, requiring every* $\mathsf{DyadicTailBall}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read the schedule, dyadic ledger, and regular-tail row before the terminal seal is available;*
6. *provenance exhaustion, requiring all accepted reads to be sourced by P and locally named by N, with no hidden quotient, selected-limit, trichotomy, countable-choice, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicSandwichUp.DyadicSandwichTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3627.1. The visible coordinates are exactly $L, U, S, D, R, H, C, P, N$, so carrier admission and provenance exhaustion are read from those rows.

The order and containment obligations are the two analytic faces of the same displayed dyadic ledger. The lower and upper rows are compared through D, while the regular-tail row R is inspected only at windows named by S.

Classifier stability follows componentwise through H and by replay through C. Since the replay route names the same lower row, upper row, schedule, dyadic ledger, and regular tail after transport, a classifier comparison cannot replace the sandwich with a quotient class or a selected limiting object.

Finally the consumer handoff is downstream of S, D, R. A $\mathsf{DyadicTailBall}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read therefore reaches the terminal boundary only after the finite squeeze rows have been displayed, and the carrier list supplies no trichotomy, countable-choice, or ambient-completeness datum. □

Theorem 2663.3 (DyadicSandwich regular Cauchy squeeze route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{DyadicSandwich}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicTailBall}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The route first reads the dyadic lower and upper endpoints with their tolerance ledger, then the finite schedule, then the regular-tail row, then the dyadic tail-ball boundary, and only then the Real seal. It exports no quotient equality, selected limit, trichotomy, countable-choice principle, or ambient completeness theorem.

Proof. Start from an accepted packet $Q = (L, U, S, D, R, H, C, P, N)$. The dyadic information is exhausted by L, U, D, so every lower/upper comparison and tolerance request is first reduced to the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

The schedule row S fixes the finite windows on which the squeeze is inspected. The regular-tail row R is attached to those windows, so a consumer cannot read a tail value outside the displayed schedule.

The dyadic tail-ball boundary is then obtained by replaying the same lower, upper, schedule, dyadic-ledger, and regular-tail rows through C and transporting them through H. The provenance row P and local naming row N package that route for the terminal $\mathsf{Real}^{\uparrow}$ read.

No remaining carrier coordinate can supply quotient equality, a selected limit, trichotomy, countable choice, or ambient completeness. Hence every Real-facing read factors through the displayed constructive squeeze route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSandwich}^{\uparrow}$ packet is generated from a lower dyadic row, an upper dyadic row, a finite StreamName schedule, a DyadicRatCore tolerance ledger, a RegSeqRat regular-tail row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$, seed)

Scope closed: Definition 3627.1, Definition 3627.2, and Definition 3627.3 seed the constructive finite squeeze route from dyadic tail windows to Real-facing reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower/upper order, tail containment, classifier stability, squeeze handoff, and provenance exhaustion for BEDC.Derived.DyadicSandwichUp.

Chapter 2664

A Concrete Naming Certificate for $\mathsf{CauchyCompletionRecursor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and uniformly continuous extension, the $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ packet records the elimination principle by which a completion-facing consumer reads a completed object only through a displayed dense source, a modulus-controlled map, an extension handoff, and a uniqueness row. It sits after the regular completion carrier of chapter 1639, the source admission row of chapter 1527, the uniformly continuous extension packet of chapter 769, and the terminal Real seal of chapter 38. The recursor is an elimination route for displayed completion data; it is not a selected limit, quotient stream, countable-choice principle, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2664.1 (CauchyCompletionRecursor carrier)**.** *A* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, f, m, E, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the dense source row, read through the Bishop regular Cauchy completion carrier and the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-admission route.*
2. *T is the completion target row, exposing only the completion-facing boundary consumed by the recursor.*
3. *f is the source map row, visible only on the displayed dense source S.*
4. *m is the uniform modulus row controlling the source map before any completion-facing read is admitted.*
5. *E is the extension handoff row supplied by the uniformly continuous extension route.*
6. *U is the uniqueness row, requiring any two admitted extension reads with the same source restriction and modulus ledger to agree through the displayed completion and Real seal rows.*
7. *H records componentwise* hsame *transport for the dense source, target, source map, modulus, extension handoff, uniqueness row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing request through S, f, m, E, U and then to the terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for selected limits, quotient stream representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionRecursorUp.CauchyCompletionRecursorTasteGate single carrier alignment`

Theorem 2664.2 (CauchyCompletionRecursor NameCert obligations)**.** *For every accepted* CauchyCompletionRecu *carrier $R = (S, T, f, m, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dense source, completion target, source map, modulus ledger, extension handoff, uniqueness row, transport, replay, provenance, and local naming rows;*
2. *source restriction, requiring f to be read only on the displayed dense source S admitted by the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route;*
3. *modulus preservation, requiring every extension read to preserve the same uniform modulus row m before the target row T is consumed;*
4. *extension handoff, requiring E to be supplied through the uniformly continuous extension route and not by a selected limit or host completion theorem;*
5. *uniqueness, requiring U to identify any two admitted extension reads with the same source restriction, modulus ledger, and completion target route;*
6. *ledger non-escape, requiring H, C, P, N to replay and name only the displayed rows and to export no selected limit, quotient stream, countable-choice selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3628.1. The visible packet consists exactly of the dense source row S, target row T, source map row f, modulus row m, extension handoff E, uniqueness row U, and structural rows H, C, P, N.

Source restriction follows from the placement of S before f. The source map can be inspected only after the dense source has been displayed through the Bishop regular completion and $\mathsf{CauchyCompletionUnit}^{\uparrow}$ routes.

Modulus preservation and extension handoff are read through m and E. The uniformly continuous extension route consumes the same modulus ledger before it exposes an extension row, so no selected limit or ambient completion theorem can replace the displayed handoff.

Uniqueness is exactly the role of U. Once the source restriction, modulus ledger, target row, and extension handoff agree, the local uniqueness row identifies the two recursor reads through the displayed completion-facing route.

Finally, H, C, P, N exhaust the ledger. They transport, replay, source, and name the displayed rows only, so there is no carrier coordinate from which quotient streams, selected limits, countable choice, host real equality, or ambient Real completeness can be exported. □

Theorem 2664.3 (CauchyCompletionRecursor universal route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionUnit}^{\uparrow} \longmapsto \mathsf{UniformContinuousExtension}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer first reads the displayed dense source S, then the source map f and modulus ledger m, then the extension handoff E, then the uniqueness row U, and only then the terminal Real seal. The route contains no selected limit, countable choice, quotient stream, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an accepted packet $R = (S, T, f, m, E, U, H, C, P, N)$. The dense source row S is the only source coordinate available to the consumer, and it is admitted through the Bishop regular completion and $\mathsf{CauchyCompletionUnit}^{\uparrow}$ routes.

Replay the continuation row C. A completion-facing request reaches f only after S is displayed, and it reaches E only after the modulus ledger m has been read. Thus the extension handoff is modulus-controlled rather than limit-selected.

The uniqueness row U is read before the terminal Real seal. It forces any two admitted extension reads with the same source restriction, modulus ledger, and target route to share the same completion-facing output row.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and naming. Since the carrier has no coordinate for selected limits, countable choice, quotient streams, host real equality, or ambient Real completeness, no completion-facing consumer can export any of those data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ packet is generated from a dense source row, completion target row, source map, uniform modulus ledger, extension handoff, uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3628.1, Definition 3628.2, and Definition 3628.3 seed the finite elimination route from displayed dense source data through modulus-controlled extension and uniqueness into the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source restriction, modulus preservation, extension handoff, uniqueness, replay stability, provenance exhaustion, and Real seal nonescape for BEDC.Derived.CauchyCompletionRecursorUp.

Chapter 2665

A Concrete Naming Certificate for $\mathsf{DedekindReal}^{\uparrow}$

Following Dedekind and Bishop–Bridges constructive analysis, the $\mathsf{DedekindReal}^{\uparrow}$ packet records a real name by a displayed located cut together with the finite Cauchy readback surface that lets completion-facing consumers compare cut data with regular rational Cauchy names. It sits beside the located-cut carrier of chapter 404, the tight cut boundary of **??**, the regular rational sequence route of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the cut-versus-Cauchy comparison surface; it is not a quotient of cuts, a selected supremum, trichotomy on rationals, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 2665.1 (DedekindReal carrier)**.** *A* $\mathsf{DedekindReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, C, T, R, S, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *row exposing the lower and upper rational probes and the locatedness witness available to the cut.*
2. *C is the* $\mathsf{CauchyCut}^{\uparrow}$ *row tying finite Cauchy windows to the same cut boundary.*
3. *T is the* $\mathsf{TightLocatedCut}^{\uparrow}$ *row recording the no-gap compatibility between lower and upper probes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row used as the Cauchy name readback.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the inspected rational approximants.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare cut probes and Cauchy windows at one displayed budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, C, T, R, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for cut, Cauchy-cut, tightness, sequence, stream, tolerance, Real-seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a cut-facing or Cauchy-facing request through L, C, T, R, S, Q and then to E.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{TightLocatedCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient cut equality, a selected supremum, a detached Cauchy limit, rational trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DedekindRealUp.DedekindRealTasteGate single carrier alignment`

Theorem 2665.2 (DedekindReal NameCert obligations)**.** *For every accepted* $\mathsf{DedekindReal}^{\uparrow}$ *carrier* $D = (L, C, T, R, S, Q, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, located-cut exposure through L, Cauchy-cut compatibility through C, tight no-gap checking through T, regular rational readback through R, finite stream-window exposure through S, dyadic tolerance comparison through Q, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Every Real and Completion consumer that treats a Dedekind-style cut as a real name must read the finite route

$$\mathsf{LocatedCut}^{\uparrow} \longrightarrow \mathsf{CauchyCut}^{\uparrow} \longrightarrow \mathsf{TightLocatedCut}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Re}$$

before exporting any real-facing row.

Proof. Project the carrier of Definition 3629.1. The accepted packet exposes only the located cut, Cauchy-cut, tightness, regular rational sequence, stream window, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

The cut side is forced by L, C, T. Locatedness supplies rational probes on both sides of the boundary, the Cauchy-cut row ties those probes to finite Cauchy windows, and the tight row records that no displayed gap remains between the lower and upper faces.

The Cauchy side is forced by R, S, Q, E. The regular rational sequence supplies the finite approximants, the stream row selects the inspected windows, the dyadic ledger fixes the comparison budget, and the Real seal can be reached only after those rows have been consumed.

Finally H, K, P, N exhaust the structural surface. They transport, replay, source, and name the displayed rows only, so quotient cut equality, selected suprema, detached limits, rational trichotomy, countable choice, and ambient Real completeness have no export coordinate in the local certificate. □

Theorem 2665.3 (DedekindReal cut–Cauchy comparison)**.** *Let $D = (L, C, T, R, S, Q, E, H, K, P, N)$ be an accepted* $\mathsf{DedekindReal}^{\uparrow}$ *carrier. A cut-facing read and a Cauchy-facing read compare only when both are routed through the same tolerance ledger Q and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal E. In particular, every admitted comparison between the Dedekind cut surface and the regular Cauchy real-name surface factors through*

$$(L, C, T) \longmapsto Q \longleftarrow (R, S)$$

and then through E, H, K, P, N.

The comparison contains no quotient of cuts, no selected supremum, no selected Cauchy limit, no trichotomy branch, no countable-choice selector, and no appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted packet. A cut-facing read can inspect only L, C, T, while a Cauchy-facing read can inspect only R, S before the common dyadic tolerance row is reached.

The row Q is the only displayed comparison budget. Therefore a lower-or-upper rational probe from L and a regular rational approximant from R, S can be compared only after both sides have entered the same dyadic ledger.

The terminal seal E is downstream of that shared ledger. Transport and replay through H and K preserve the same order, and P, N package only the resulting finite route.

Since no carrier coordinate stores quotient cut equality, a supremum selector, a detached limit, trichotomy, countable choice, or ambient completeness, the comparison is exactly the finite cut-versus-Cauchy route displayed above. □

Falsifiable prediction: Every accepted $\mathsf{DedekindReal}^{\uparrow}$ consumer that claims a cut-versus-Cauchy real-name comparison must exhibit locatedcut, cauchycut, tightlocatedcut, regseqrat, streamname, dyadicratcore, real, transport, replay, provenance, and local namecert rows. A consumer omitting any one of those rows refutes the packet.

Independence witness: locatedcut, cauchycut, tightlocatedcut, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole packet $D = (L, C, T, R, S, Q, E, H, K, P, N)$, because each sibling contributes only cut probes, Cauchy compatibility, tightness, rational sequence data, finite windows, tolerance budgets, or terminal real sealing while this chapter binds them as one finite comparison certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindReal}^{\uparrow}$ packet is generated from a LocatedCut row, CauchyCut compatibility row, TightLocatedCut no-gap row, RegSeqRat readback, StreamName finite windows, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3629.1, Definition 3629.2, and Definition 3629.3 seed the finite cut-versus-Cauchy real-name route from located cut probes through Cauchy compatibility, tightness, regular rational readback, finite stream windows, dyadic tolerance, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cut equality, selected suprema, selected Cauchy limits, rational trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cut exposure, Cauchy-cut compatibility, tight no-gap checking, regular rational readback, finite-window exposure, dyadic tolerance comparison, terminal Real seal routing, and ledger nonescape for BEDC.Derived.DedekindRealUp.

Chapter 2666

A Concrete Naming Certificate for $\mathsf{ObservableDiameterDecay}^{\uparrow}$

The $\mathsf{ObservableDiameterDecay}^{\uparrow}$ packet realizes the compact finite-net and uniform-modulus route of chapter 9237 as a finite diameter ledger. It consumes the compact source of chapter 129, the pointwise graph and radius data of chapter 130, the finite radius ledger of chapter 452, the uniform-modulus consumer of chapter 369, and the source-to-target triangle estimate of chapter 1159. The packet exposes observable diameter decay as NameCert data rather than as a narrative compactness bridge.

Definition 2666.1 (Observable diameter decay carrier)**.** *An* $\mathsf{ObservableDiameterDecay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, L, T, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the inspected centers and source-radius windows.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise graph and local-modulus row for the displayed map on those centers.*
3. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius row binding the finite compact cover to one positive radius ledger.*
4. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *triangle row that turns source-radius bounds and pointwise target estimates into a target-diameter bound.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, read only after K, F, L, T have been displayed.*
6. *H records componentwise* hsame *transport for compact-net, continuous-map, radius-ledger, triangle-estimate, uniform-modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from finite-net source windows through pointwise estimates, the finite Lebesgue-number row, the triangle estimate, and the uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{ObservableDiameterDecay}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, T, M, H, C, P, N$. It has no coordinate for open-cover compactness, host semantic continuity, selected centers outside the displayed finite net, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayTasteGate` single carrier alignment

Theorem 2666.2 (Observable diameter decay NameCert obligations)**.** *For every accepted* ObservableDiameterDeca *carrier $O = (K, F, L, T, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$ *surface consists exactly of carrier admission; compact finite-net coverage through K; pointwise-modulus compatibility through F; finite Lebesgue-number radius exactness through L; triangle-estimate diameter control through T; uniform-modulus handoff through M; componentwise transport and displayed replay through H and C; and provenance exhaustion through P and N.*

Proof. Project the carrier of Definition 3630.1. The visible coordinates are exactly the compact finite-net row, continuous-map row, finite radius ledger, triangle estimate, uniform-modulus handoff, and structural rows.

Coverage is read from K, and pointwise-modulus compatibility is read from F. The finite Lebesgue-number row L binds those local source-radius windows to a single positive radius ledger.

The triangle row T converts the shared source-radius bound and pointwise target estimates into the displayed target-diameter control. The uniform-modulus row M is downstream from K, F, L, T, so the handoff cannot be inspected before the finite diameter-decay route is exposed.

Transport and replay through H and C preserve exactly those rows, while P and N source and name them. Since the carrier contains no open-cover compactness, host semantic continuity, external center selector, or ambient completeness coordinate, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayNamecertObligations`

Theorem 2666.3 (Observable diameter decay compact-uniform route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read from an accepted* $\mathsf{ObservableDiameterDecay}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{FiniteLebesgueNumber}^{\uparrow} \longrightarrow \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longrightarrow \mathsf{Uniform}$$

The route measures diameter decay on displayed compact-net cells, transports pointwise target estimates along the finite Lebesgue-number radius row, and seals the result as a uniform-modulus handoff. It exports no open-cover compactness, host semantic continuity, selected center outside the packet, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayCompactUniformRoute`

Proof. Begin with an accepted packet $O = (K, F, L, T, M, H, C, P, N)$. The compact source data are exactly the finite-net centers and source-radius windows recorded in K.

Read the continuous-map row F at those displayed centers. These pointwise target estimates are tied to the same finite source windows that the compact-net row exposes.

The radius row L supplies the finite Lebesgue-number ledger needed to make the source-radius request uniform over the inspected compact-net cells. The triangle row T then turns the source bounds and pointwise target estimates into a target-diameter bound.

Only after that route is replayed through C and transported through H can the uniform-modulus handoff M be read. The provenance and naming rows P, N package the same route, and no carrier coordinate supplies open-cover compactness, host semantic continuity, external center selection, or ambient completeness. □

Falsifiable prediction: Removing the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ radius row from an accepted ObservableDiameterDec carrier breaks the $\mathsf{UniformModulus}^{\uparrow}$ handoff, because the compact finite-net source row and pointwise $\mathsf{ContinuousMap}^{\uparrow}$ estimates no longer share one displayed positive-radius ledger.

Independence witness: compactuniformcontinuity, finitelebesguenumber, metrictriangleuniformestimate, uniformmodulus, compactmetric: none is bijective to $\mathsf{ObservableDiameterDecay}^{\uparrow}$ because this packet jointly carries compact-net coverage, pointwise continuous-map estimates, the finite Lebesgue-number radius ledger, triangle-estimate diameter control, uniform-modulus handoff, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservableDiameterDecay}^{\uparrow}$ packet is generated from a CompactMetric finite-net row, a ContinuousMap pointwise-modulus row, a FiniteLebesgueNumber radius ledger, a MetricTriangleUniformEstimate diameter-control row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3630.1, Definition 3630.2, and Definition 3630.3 seed the finite compact-net diameter-decay route consumed by uniform-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host semantic continuity, selected centers outside the displayed finite net, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net coverage, pointwise-modulus compatibility, finite radius exactness, triangle-estimate diameter control, uniform-modulus handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.ObservableDiameterDecayUp.

Chapter 2667

A Concrete Naming Certificate for $\mathsf{FastCauchyNormalForm}^{\uparrow}$

Bishop–Bridges constructive Cauchy completion treats a real number as regular approximation data rather than as a quotient object selected by an ambient limit operation. The $\mathsf{FastCauchyNormalForm}^{\uparrow}$ packet records the finite normal-form surface by which dyadic rational cores, finite stream names, regular rational sequences, terminal Real rows, and Cauchy-rate comparisons are read as one L10 completion-facing NameCert object. It is not quotient real equality, selected representative choice, countable choice, trichotomy, or an ambient completeness oracle.

The sibling dependency is part of the carrier: D imports the dyadic tolerance carrier of chapter 291, S imports the finite-window stream surface of chapter 126, R imports the regular rational route of chapter 301, E imports the terminal seal of chapter 38, and C imports the rate comparison surface of chapter 1671. The normal-form packet binds those sibling rows before any completion consumer can treat a regular name as fast normalized.

Definition 2667.1 (Real fast Cauchy normal-form carrier)**.** *A* $\mathsf{FastCauchyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, C, H, Q, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row, carrying the finite endpoint, radius, distance, and budget data used by the normal form.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected stream observations and the omitted-tail ledger.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row whose modulus is being normalized.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *C is the* $\mathsf{CauchyRateComparison}^{\uparrow}$ *row comparing the displayed regular sequence against the fast dyadic rate demanded by the packet.*
6. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular-sequence, Real-seal, Cauchy-rate, replay, provenance, and local naming rows.*
7. *Q records displayed* Cont *replay from a fast-normal-form request through D, S, R, E, C.*
8. *P is the* Pkg *provenance over* $\mathsf{FastCauchyNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRateComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, E, C, H, Q, P, N$ and their displayed replay route. It has no coordinate for quotient real equality, selected representatives, detached limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchyNormalFormUp.FastCauchyNormalFormTasteGate single carrier alignment`

Theorem 2667.2 (Real fast Cauchy normal-form NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyNormalForm}^{\uparrow}$ *carrier $F = (D, S, R, E, C, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, classifier stability under componentwise* hsame*, dyadic tolerance nonescape through D, finite-window exposure through S, regular-sequence admission through R, terminal seal routing through E, fast-rate comparison through C, displayed replay through Q, provenance through P, and local naming through N.*

Every completion-facing consumer that treats a regular rational sequence as fast-normalized must read the finite route

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{CauchyRateComparison}^{\uparrow}$$

before exporting any normal-form row.

Proof. Project the carrier of Definition 3631.1. The accepted packet exposes only the dyadic tolerance row, finite stream window, regular rational sequence, terminal Real seal, Cauchy-rate comparison, transport, replay, provenance, and local naming rows.

The analytic route is forced by D, S, R, E, C. The dyadic ledger supplies the displayed tolerance budgets, the stream row records the finite observations, the regular sequence row supplies the Cauchy data, the Real row seals only that displayed route, and the comparison row checks the fast rate against those same finite rows.

The structural rows H, Q, P, N preserve this route under componentwise transport and replay. Since no accepted coordinate stores quotient real equality, a selected representative, detached limit choice, countable choice, trichotomy, or ambient completeness, none of those objects belongs to the NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{FastCauchyNormalForm}^{\uparrow}$ consumer that claims fast Cauchy normal form must exhibit the dyadic tolerance row, finite stream window, regular rational sequence row, terminal Real seal row, Cauchy-rate comparison row, componentwise transport, displayed replay, provenance, and local naming row. A consumer omitting dyadicratcore, streamname, regseqrat, real, cauchyratecomparison, or namecert sibling slugs refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchyratecomparison: no listed sibling supplies the whole packet $F = (D, S, R, E, C, H, Q, P, N)$, because each sibling contributes only tolerances, finite observations, regular sequence data, terminal sealing, or rate comparison while this chapter binds all five analytic faces as one finite normal-form certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FastCauchyNormalForm}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular-sequence data, a terminal RealUp seal, a CauchyRateComparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 3631.1 and Definition 3631.2 seed the finite fast Cauchy normal-form route over dyadic tolerance data, finite stream names, regular rational sequences, terminal Real sealing, Cauchy-rate comparison, transport, replay, provenance, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, detached limit choice, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance nonescape, finite-window exposure, regular-sequence admission, terminal-seal routing, fast-rate comparison, and FieldFaithful FastCauchyNormalFormUp for BEDC.Derived.FastCauchyNormalFormUp.

The Calculus$^{\uparrow}$ dossier consumes the finite Limit$^{\uparrow}$ route of chapter 1678 and the regular-Cauchy diagonal meet of chapter 909 before derivative and integral rows are read.

Chapter 2668

A Concrete Naming Certificate for $\mathsf{Calculus}^{\uparrow}$

Following Cartan's local differential calculus and the Bishop–Bridges constructive treatment of limits, derivatives, and integrals, the $\mathsf{Calculus}^{\uparrow}$ packet records the finite analysis root used by later derivative and integration chapters. Its derivative/integral local-linearization carrier consumes the $\mathsf{ContinuousMap}^{\uparrow}$ modulus surface of chapter 130, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ error ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It sits beside the real-name boundary of chapter 38, the generic limit route of chapter 1678, the continuous-map surface used throughout the analysis region, the derivative carrier of chapter 122, the integral carrier, and the regular rational readback of chapter 301. The packet names the common calculus substrate without importing a host function space, a global limit operator, quotient equality of functions, or an ambient completeness theorem.

Definition 2668.1 (Calculus local-linearization carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier is a finite* BHist *packet*

$$\Lambda = (E, R, D, L, J, H, C, P, N)$$

whose rows are the terminal $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Derivative}^{\uparrow}$ *finite difference-quotient row* L*, the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite tagged-sum row* J*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. The classifier compares accepted packets only through these rows. It has no coordinate for a host differentiability theorem, selected antiderivative, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusLocalLinearizationCarrier`
Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocalLinearizationCarrier admission`

Theorem 2668.2 (Calculus derivative continuation obligation)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes a derivative-facing continuation only after the* $\mathsf{Real}^{\uparrow}$ *seal* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient row* L*,* hsame *transport* H*,* Cont *replay* C*, provenance* P*, and local NameCert row* N *have been displayed. The read cannot export a host differentiability theorem, quotient real equality, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeContinuationObligation`

Proof. Project the carrier of Definition 3632.1. The derivative-facing row is L, and its tolerance and value data are visible only together with the real seal E, regular-rational readback R, and dyadic ledger D.

The structural rows H, C, P, N transport, replay, source, and name that same finite packet. Since none of those rows contains a host differentiability theorem or quotient real equality, the continuation remains a displayed finite difference-quotient route.

The Riemann row J may coexist in the packet, but it does not supply derivative data. Thus every derivative consumer must read through E, R, D, L, H, C, P, N before any derivative-facing conclusion is public. □

Theorem 2668.3 (Calculus integral finite-sum obligation)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes an integral-facing read only through the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite tagged-sum row* J *together with the* $\mathsf{Real}^{\uparrow}$ *seal* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, componentwise transport, replay, provenance, and local naming. The read does not select an antiderivative and does not import quotient real equality or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusIntegralFiniteSumObligation`

Proof. Read J from the carrier. It records the partition, tags, finite sum, and endpoint handoff as displayed calculus data, so an integral consumer cannot bypass the finite-sum route.

The rows E, R, D provide the real-facing seal, regular-rational comparison, and dyadic tolerance ledger used to interpret that sum. The structural rows H, C, P, N preserve the same finite route under accepted transport and replay.

The derivative row L remains separate. Because no coordinate of Λ contains a selected antiderivative, host integration functional, quotient real equality, or ambient completeness theorem, the integral obligation is exactly the finite-sum route through J and the displayed support rows. □

Theorem 2668.4 (Calculus limit real-seal obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing limit read exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, derivative row* L *or Riemann finite-sum row* J*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*,* hsame *transport,* Cont *replay, provenance, and local NameCert rows. No such read may use a host limit theorem, quotient real equality, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local data.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealSealObligation`

Proof. Start from a Real-facing calculus read. The carrier exposes E as the only terminal real seal, so every such read must arrive at E through the displayed regular-rational and dyadic rows R, D.

There are two calculus sources for that seal. A derivative read uses L and the finite difference-quotient route; an integral read uses J and the finite tagged-sum route. Both sources are transported and replayed only by H and C, with provenance and local naming recorded by P and N.

Since the carrier has no coordinate for host limits, quotient real equality, selected antiderivatives, or ambient completeness, the real-seal obligation is confined to the displayed BEDC rows. □

Definition 2668.5 (Calculus root public package)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root public package of an accepted carrier is the finite packet*

$$\Pi_{\mathsf{Calculus}^{\uparrow}} = (E, R, D, L, J, H, C, P, N; C_{\mathrm{root}}, D_{\mathrm{root}}, I_{\mathrm{root}}, L_{\mathrm{root}}, R_{\mathrm{root}}, Q_{\mathrm{root}}, Y_{\mathrm{root}}, H_{\mathrm{root}}, T_{\mathrm{root}}, P_{\mathrm{root}}, N_{\mathrm{root}})$$

obtained by reading the local-linearization rows $E, R, D, L, J, H, C, P, N$ *together with the root rows* $C_{\text{root}}, D_{\text{root}}, I_{\text{root}}, L_{\text{root}}, R_{\text{root}}, Q_{\text{root}}, Y_{\text{root}}, H_{\text{root}}, T_{\text{root}}, P_{\text{root}}, N_{\text{root}}$ *of Definition 3632.43. Its public surface is the derivative row, Riemann finite-sum row,* $\mathsf{Limit}^{\uparrow}$ *row,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming.*

The package compares accepted calculus reads only through those rows. It has no coordinate for a host function space, selected antiderivative, selected derivative operator, quotient equality of functions, host limit operator, host integral functional, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootPublicPackage`

Definition 2668.6 (Calculus root obligation surface)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root obligation surface is the ordered read of* $\Pi_{\mathsf{Calculus}^{\uparrow}}$ *through the derivative row, integral finite-sum row,* $\mathsf{Limit}^{\uparrow}$ *row,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, and the transport, replay, provenance, and local naming rows. It is the same finite packet as Definition 3632.5, read as downstream obligations rather than as a public package.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootObligationSurface`

Theorem 2668.7 (Calculus root limit derivative integral route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *root read for limits, derivatives, integrals, continuous-map replay,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{Real}^{\uparrow}$ *sealing passes through the obligation surface of Definition 3632.6. The consumer may choose the relevant operation row, but the shared route remains* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Proof. Read the obligation surface as the public package with its rows ordered by consumer. Derivative, integral, and limit consumers select D_{root}, I_{root}, or L_{root}, respectively, while continuous-map replay, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming remain common rows.

Apply the existing root package readiness theorem. It already forces each listed consumer through $\Pi_{\mathsf{Calculus}^{\uparrow}}$, so reading the same packet as an obligation surface gives the stated L10 route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitDerivativeIntegralRoute`

Theorem 2668.8 (Calculus root L10 nonescape)**.** *The obligation surface of Definition 3632.6 exports no host function space, quotient function equality, selected antiderivative, host limit operator, host integral functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every L10 calculus consumer stays inside the displayed limit, derivative, integral, continuous-map, dyadic-rational, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootL10Nonescape`

Proof. Compare the refused objects with the row inventory of Definition 3632.6. None is a row of the displayed obligation surface.

Use Definition 3632.7 for the positive route and Definition 3632.10 for the boundary. Together they exhaust the public calculus read through $\Pi_{\mathsf{Calculus}^{\uparrow}}$ and leave no coordinate for the refused host objects. □

Theorem 2668.9 (Calculus root public package readiness)**.** *Every downstream* $\mathsf{Calculus}^{\uparrow}$ *consumer of derivative, integral, limit, continuous-map, or Real-facing data factors through the root public*

package of Definition 3632.5. A derivative consumer reads the derivative rows and the shared graph, dyadic, regular-rational, Real, transport, replay, provenance, and naming rows; an integral consumer reads the Riemann finite-sum rows and the same support; a limit consumer reads the $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows and the same support. No downstream calculus consumer obtains a public conclusion from a row outside* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootPublicPackageReadiness`

Proof. Start with the local-linearization obligations. The derivative, integral, and limit real-seal statements above force every local read through $E, R, D, L, J, H, C, P, N$ before a public conclusion is visible.

Now project the root carrier of Definition 3632.43. Its public operation rows are $C_{\mathrm{root}}, D_{\mathrm{root}}, I_{\mathrm{root}}, L_{\mathrm{root}}, R_{\mathrm{root}}, Q$

Use Definition 3632.44, Definition 3632.45, and Definition 3632.48. They place derivative, integral, and limit consumers behind the graph row, the relevant operation row, and the common regular-rational, dyadic, Real, transport, replay, provenance, and naming support.

The package $\Pi_{\mathsf{Calculus}^{\uparrow}}$ is exactly the union of those local and root rows. Since the cited obligations exhaust the operation ledger, a downstream consumer has no remaining coordinate from which to read calculus data outside the package. □

Theorem 2668.10 (Calculus root package nonescape)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root public package exports no host function space, selected antiderivative, selected derivative operator, quotient real equality of functions, host limit operator, host integral functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public derivative, integral, limit, continuous-map, or Real-facing read remains inside the derivative, Riemann finite-sum,* $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local naming rows of* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootPackageNonescape`

Proof. Read the excluded objects against Definition 3632.5. None is a row of $\Pi_{\mathsf{Calculus}^{\uparrow}}$, and the package comparison only inspects the displayed local and root rows.

For derivative and integral reads, apply Definition 3632.13 and Definition 3632.45. They keep derivative and Riemann finite-sum data in separate source rows and refuse a host fundamental theorem or selected antiderivative.

For limit, continuous-map, and Real-facing reads, apply Definition 3632.4, Definition 3632.42, and Definition 3632.46. These place the read behind the continuous-map, limit, regular-rational, dyadic, and Real-seal rows before anything is public.

Thus every exported calculus conclusion is obtained through the package rows. The excluded host structures remain absent, so the package cannot supply them by projection. □

Theorem 2668.11 (Calculus RegSeqRat dyadic error route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *Real-facing error estimate exposed by an accepted local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. The route is finite packet readback and not a host analytic estimate, quotient real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegSeqRatDyadicErrorRoute`

Proof. Project the local-linearization carrier. The rows R and D are the only coordinates that record rational readback and dyadic tolerance data for a Real-facing calculus estimate.

The derivative source row L and the Riemann finite-sum source row J both sit before the terminal seal E. Their estimates can be transported and replayed only by the displayed structural rows H, C, P, N.

Thus any accepted Real-facing error estimate must read R, D before E. Since the carrier has no coordinate for host analysis, quotient real equality, or ambient completeness, no other route can supply the estimate. □

Theorem 2668.12 (Calculus ContinuousMap limit route)**.** *Every continuous-map limit read exposed by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier must display the graph and pointwise replay data of* $\mathsf{ContinuousMap}^{\uparrow}$ *before any* $\mathsf{Limit}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is consumed. In the local-linearization packet this means the derivative or Riemann source row is read only after its graph data has been routed through* R, D, H, C, P, N *toward* E*.*

Proof. Read the carrier rows in order. The local-linearization packet has no separate host function-space coordinate; graph and pointwise replay data can appear only through the displayed calculus source rows and their replay row C.

The real-seal obligation already confines limit-facing reads to the regular-rational row R, dyadic tolerance row D, one of the calculus source rows L, J, and the seal E. Therefore a continuous-map consumer cannot reach the Limit or Real seal without first displaying the graph data attached to that source row.

The transport, provenance, and local naming rows preserve the same dependency order. Hence the route is a finite ContinuousMap-to-Limit-to-Real read, not an ambient continuity theorem or a hidden limit operator. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuousMapLimitRoute`

Theorem 2668.13 (Calculus Riemann derivative separation)**.** *In every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$*, the* $\mathsf{Derivative}^{\uparrow}$ *row* L *and the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite-sum row* J *remain separate consumers of the common displayed support* E, R, D, H, C, P, N*. The shared support does not identify derivative and integral conclusions and does not supply a host fundamental theorem between them.*

Proof. Project the carrier. The derivative data is stored in L, and the finite tagged-sum data is stored in J; these are distinct coordinates of the finite packet.

Both rows may use the same real seal, rational readback, dyadic tolerance, transport, replay, provenance, and local naming support. Sharing those support rows only records common BEDC infrastructure; it does not turn one source row into the other.

Since the carrier contains no coordinate for a host fundamental theorem, selected antiderivative, quotient function equality, or ambient real completeness, derivative and Riemann finite-sum consumers remain separated after projection. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannDerivativeSeparation`

Theorem 2668.14 (Calculus Real-seal error budget)**.** *Every derivative, integral, or limit read exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes a source graph or finite-sum row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any calculus conclusion is public. The exposed route is one of*

$$C, L, R, D, E, H, P, N \qquad \text{or} \qquad C, J, R, D, E, H, P, N,$$

according to whether the consumer asks for a derivative/limit read or a Riemann finite-sum read, and it has no coordinate for host limits, quotient function equality, selected antiderivatives, or ambient real completeness.

Proof. Project the local-linearization carrier. Its calculus-facing source rows are the derivative row L, the Riemann finite-sum row J, the continuous-map graph row implicit in C, the regular-rational readback row R, the dyadic tolerance row D, and the real seal E.

Apply Definition 3632.2 to derivative-shaped reads and Definition 3632.3 to integral-shaped reads. Apply Definition 3632.30 and Definition 3632.12 for limit and continuous-map reads.

The real-seal route is then forced by Definition 3632.4. The separating theorem Definition 3632.13 prevents the finite-sum row from supplying derivative data and prevents the derivative row from supplying integral data.

The structural rows H, C, P, N transport, replay, source, and name the same exposed inventory. Since the carrier lacks the excluded host coordinates, the listed rows form the full public error budget before any calculus conclusion is read. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealSealErrorBudget`

Theorem 2668.15 (Calculus source-window obligation)**.** *For an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier, every derivative, integral, and limit consumer must display the finite source window on which the read depends before it reaches a public conclusion. A derivative read displays a difference-quotient window, an integral read displays a partition and tagged finite-sum window, and a limit read displays the* $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *window rows, all with* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming.*

Proof. Start from the carrier rows of Definition 3632.21 and the local-linearization rows of Definition 3632.1. They display the finite source rows D, I, L, J together with the regular-rational, dyadic, Real, transport, replay, provenance, and naming support.

For derivative reads, use Definition 3632.2 and Definition 3632.33. The conclusion remains behind the displayed difference-quotient source, the continuous-map route, the regular-rational readback, and the dyadic tolerance data.

For integral reads, use Definition 3632.3 and Definition 3632.41. The finite partition, tagged-sum, endpoint, and readback rows are visible before the Real boundary is consumed.

For limit reads, use Definition 3632.30, Definition 3632.12, and Definition 3632.42. These force the continuous-map and limit windows to remain inside the displayed BEDC route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusSourceWindowObligation`

Theorem 2668.16 (Calculus root-unblock limit window scope)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *limit consumer that exports a* $\mathsf{Real}^{\uparrow}$*-facing conclusion must expose the* $\mathsf{Limit}^{\uparrow}$ *window row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming before the conclusion is public. The consumer cannot replace those rows by a host limit operator, quotient real equality, selected function-space theorem, or ambient completeness principle.*

Proof. Project the carrier rows of Definition 3632.21. The limit route is L, the continuous-map route is C, the regular-rational readback is Q, the terminal real boundary is R, and the structural rows are H, T, P, N.

Use Definition 3632.15 to require the finite source window before any public read. For a limit-shaped consumer, Definition 3632.30 and Definition 3632.12 identify the required $\mathsf{Limit}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows.

Finally apply Definition 3632.42 and Definition 3632.46. They keep the read behind the regular-rational, dyadic, transport, replay, provenance, naming, and Real-seal rows. Since the excluded host objects are absent from the carrier, the listed rows form the full limit-window scope. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockLimitWindowScope`

Theorem 2668.17 (Calculus root-unblock finite-sum handoff)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *integral consumer that exports a* $\mathsf{Real}^{\uparrow}$*-facing conclusion must pass through the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite-sum row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the conclusion is public. The consumer cannot bypass the finite-sum row by selecting an antiderivative, a host integration functional, quotient equality of functions, or ambient real completeness.*

Proof. Project Definition 3632.21. The integral source is I, the common regular-rational readback is Q, the terminal Real boundary is R, and the transport, replay, provenance, and naming rows are H, T, P, N.

Apply Definition 3632.15 and Definition 3632.3. Together they force the partition, tagged finite sum, endpoint handoff, regular-rational readback, dyadic tolerance data, and structural rows to be displayed before any Real-facing integral read is exported.

The handoff is separated from derivative data by Definition 3632.45. The Real seal is then controlled by Definition 3632.41 and Definition 3632.46, so the public conclusion remains a finite-sum-to-regular-rational-to-Real route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockFiniteSumHandoff`

Theorem 2668.18 (Calculus root-unblock continuous derivative window)**.** *Every derivative-facing* $\mathsf{Calculus}^{\uparrow}$ *root read that reaches a public* $\mathsf{Real}^{\uparrow}$*-facing conclusion must expose the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{Derivative}^{\uparrow}$ *difference row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming before the derivative conclusion is public. It cannot replace those rows by a host function space, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3632.21. The continuous-map route is C, the derivative source is D, the regular-rational readback is Q, the terminal Real boundary is R, and the structural rows are H, T, P, N.

Use Definition 3632.15 to force the finite derivative source window before any public read. Then apply Definition 3632.2, Definition 3632.33, and Definition 3632.12 to keep the derivative read behind the displayed graph and difference rows.

The dyadic and Real boundary is supplied by Definition 3632.11, Definition 3632.4, and Definition 3632.46. Since Definition 3632.52 excludes the host coordinates, the listed rows form the full continuous derivative window. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockContinuousDerivativeWindow`

Theorem 2668.19 (Calculus root-unblock Riemann endpoint readback)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *endpoint read consumed by a* $\mathsf{Calculus}^{\uparrow}$ *root integral conclusion must expose the finite tagged-sum row, the endpoint* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any integral conclusion is public. The endpoint read is a finite sum and readback route, not a host integration functional or selected antiderivative.*

Proof. Project the integral route of Definition 3632.21. The integral source is I, the shared regular-rational readback is Q, the Real boundary is R, and the transport, replay, provenance, and naming rows are H, T, P, N.

Apply Definition 3632.3, Definition 3632.41, and Definition 3632.17. These theorems place partition, tag, endpoint, finite-sum, regular-rational, dyadic, and Real-seal data before the public integral read.

Finally use Definition 3632.13 and Definition 3632.10. They prevent the endpoint read from being supplied by derivative data, a host integral functional, a selected antiderivative, quotient equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRiemannEndpointReadback`

Theorem 2668.20 (Calculus root-unblock RegSeqRat Real-seal factorization)**.** *Every derivative, integral, and limit consumer exported by the* $\mathsf{Calculus}^{\uparrow}$ *root package factors first through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No such consumer may use a host function space, quotient equality of functions, selected antiderivative, selected derivative operator, host limit operator, host integration functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness in place of the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *factorization.*

Proof. Read the common carrier rows from Definition 3632.21 and the root package rows from Definition 3632.5. In both inventories, derivative, integral, and limit data reach a Real-facing conclusion only through the regular-rational row and the Real boundary row.

For derivative reads, apply Definition 3632.18. For integral reads, apply Definition 3632.17 and Definition 3632.19. For limit reads, apply Definition 3632.16.

The common factorization is then exactly the one recorded by Definition 3632.11, Definition 3632.4, and Definition 3632.46. The nonescape statements Definition 3632.52 and Definition 3632.10 exclude every listed host replacement. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRegSeqRatRealSealFactorization`

Definition 2668.21 (Calculus carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, L, C, D, I, Q, H, T, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar boundary used by all endpoint values, tolerances, finite sums, and error estimates.*
2. *L is the* $\mathsf{Limit}^{\uparrow}$ *route for displayed convergence windows and downstream real seals.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the finite graph, pointwise continuity, and replay windows available to calculus consumers.*
4. *D is the* $\mathsf{Derivative}^{\uparrow}$ *local difference-quotient row, including the source point, target read, and remainder ledger.*

5. *I is the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row, exposing the partition, tagged-read, and endpoint handoff data used by integration consumers.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row shared by the limit, derivative, and integral ledgers.*
7. *H records componentwise* hsame *transport for real, limit, continuous-map, derivative, integral, regular-rational, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from real reads to limits, from limits to continuous maps, from finite difference quotients to derivative reads, and from finite sums to integral handoffs.*
9. *P is the* Pkg *provenance over* $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row.*

The classifier reads only $R, L, C, D, I, Q, H, T, P, N$. It has no coordinate for a host function space, a selected global derivative operator, an unbounded integration functional, quotient equality of functions, an open-cover compactness principle, or ambient real completeness.

Theorem 2668.22 (Calculus NameCert obligations)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *surface consists exactly of carrier admission, real-boundary exposure, limit-window readback, continuous-map replay, local-linear approximation routing, finite difference quotient readback, integral finite-sum handoff, regular-rational readback, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3632.21. The rows $R, L, C, D, I, Q, H, T, P, N$ are the full finite inventory available to a calculus consumer.

The local-linear approximation route is carried by D and is read only after the real boundary R, limit route L, continuous-map row C, and regular-rational readback Q have supplied the displayed point, tolerance, and remainder data. Thus a derivative-facing consumer sees a finite difference quotient route rather than a hidden global derivative operator.

The integral handoff is carried by I. Its partition, finite-sum, and endpoint rows are exposed before any real-facing value is consumed, and the same Q row supplies the rational readback needed to compare finite sums with limit and derivative windows.

Finally H, T, P, N transport, replay, source, and name exactly the displayed rows. Since no coordinate contains host function-space equality, ambient completeness, a selected derivative operator, or an unbounded integral functional, every accepted read remains inside the finite calculus packet. □

Definition 2668.23 (Calculus completion-source envelope)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *completion-source envelope is the finite row inventory read jointly from Definition 3632.21 and Definition 3632.1. It consists of the displayed BEDC rows*

$$\mathsf{Real}^{\uparrow},\ \mathsf{Limit}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Derivative}^{\uparrow},\ \mathsf{Integral}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The $\mathsf{Real}^{\uparrow}$ *row is the terminal seal for endpoint values and tolerances. The* $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows expose convergence windows and graph replay. The* $\mathsf{Derivative}^{\uparrow}$ *and* $\mathsf{Integral}^{\uparrow}$ *rows expose finite difference-quotient, remainder, partition, tagged-sum, and endpoint handoff data. The* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows carry rational readback and dyadic tolerance data before any Real-facing seal is consumed. The remaining* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows transport, replay, source, and name exactly this finite packet.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusCompletionSourceEnvelope`

Theorem 2668.24 (Calculus completion-source envelope obligations)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier and every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier, any derivative, integral, or limit consumer must expose the completion-source envelope rows of Definition 3632.23 before it reaches a public calculus conclusion. Thus a calculus conclusion may read only through displayed Real, Limit, ContinuousMap, Derivative, Integral, RegSeqRat, DyadicRatCore,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and not through a host limit operator, quotient function equality, selected global derivative operator, unbounded integration functional, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCompletionSourceEnvelopeObligations`

Proof. Project the finite rows of Definition 3632.21 and Definition 3632.1. Together they expose the Real boundary, limit route, continuous-map replay, derivative source, integral source, regular-rational readback, dyadic tolerance data, transport, replay, provenance, and local naming listed in Definition 3632.23.

Apply Definition 3632.22. It identifies carrier admission, real-boundary exposure, limit-window readback, continuous-map replay, local-linear approximation routing, finite difference-quotient readback, integral finite-sum handoff, regular-rational readback, componentwise transport, displayed replay, provenance, and naming as the local NameCert surface.

Now apply Definition 3632.15. Derivative, integral, and limit consumers must display their finite source windows with the regular-rational, dyadic, transport, replay, provenance, and naming rows before any public conclusion is read. The excluded host objects have no row in either carrier, so the envelope is the full public route. □

Theorem 2668.25 (Calculus completion-source real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing calculus read from an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier or accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows of Definition 3632.23 before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. In particular, derivative, integral, and limit reads may hand a value to the Real boundary only after the displayed regular-rational readback and dyadic tolerance route have been exposed inside the finite BEDC packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCompletionSourceRealReadback`

Proof. Use Definition 3632.11. It forces every Real-facing error estimate in the local-linearization carrier through the regular-rational readback row and dyadic tolerance ledger before the terminal Real seal is read.

Use Definition 3632.14. The derivative, integral, and limit routes expose a source graph or finite-sum row together with regular-rational readback, dyadic tolerance data, and the Real seal before any calculus conclusion is public.

Finally apply Definition 3632.52. Public local operation reads cannot replace the displayed packet by a host function space, host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient Real completeness. Therefore every Real-facing calculus read factors through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ before reaching $\mathsf{Real}^{\uparrow}$. □

Theorem 2668.26 (Calculus continuous-map graph replay obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a continuous-map read is public only after the consumer has displayed the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, the* $\mathsf{Limit}^{\uparrow}$ *window route* L*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*, provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. A derivative or integral consumer using that continuous-map read must therefore factor through* R, L, C, Q, H, T, P, N *before reading* D *or* I*.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuousMapGraphReplayObligation`

Proof. Project the carrier of Definition 3632.21. The only row that names a continuous-map graph is C, and its endpoint values are typed by the real boundary R and compared through the regular-rational readback Q.

The limit route L supplies the finite convergence windows through which the graph row is replayed. The rows H, T, P, N transport, replay, source, and name the same displayed graph data.

Now apply Definition 3632.22. Since the derivative row D and integral row I are later consumers of the continuous-map graph, neither can use C until the route through R, L, C, Q, H, T, P, N is visible. □

Theorem 2668.27 (Calculus derivative-integral separation obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, the derivative row* D *and the integral row* I *are separate finite rows. The row* D *may expose local difference-quotient and remainder data only through* R, L, C, Q, H, T, P, N*, while the row* I *may expose finite-sum and endpoint handoff data only through those same support rows. Neither row supplies the other's conclusion, and the carrier contains no host fundamental theorem connecting them.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeIntegralSeparationObligation`

Proof. Project Definition 3632.21. The derivative data is stored in D, and the integral data is stored in I; they are distinct coordinates of the finite packet.

Use Definition 3632.2 for the derivative-facing route and Definition 3632.3 for the integral-facing route. Each theorem reads its own calculus row together with the real, limit, continuous-map, regular-rational, transport, replay, provenance, and naming support.

Definition 3632.22 lists the full local surface and includes no coordinate for a host fundamental theorem. Hence a consumer cannot derive an integral conclusion from D, cannot derive a derivative conclusion from I, and cannot import a global theorem to identify the two reads. □

Theorem 2668.28 (Calculus RegSeqRat readback root obligation)**.** *Every real-facing value exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must display the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *together with the* $\mathsf{Real}^{\uparrow}$ *boundary* R *and the dyadic tolerance route inherited by the regular-rational readback. This readback is required before any limit, derivative, or integral read is exported from* L*,* D*, or* I*.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegSeqRatReadbackRootObligation`

Proof. Begin with the carrier of Definition 3632.21. The terminal real boundary is R, and the common rational readback row is Q; no other coordinate converts limit, derivative, or integral data into a real-facing value.

The earlier real-seal theorem Definition 3632.4 already forces Real-facing limit reads through regular-rational and dyadic tolerance data. The derivative and integral obligations similarly keep their finite quotient and finite-sum reads attached to the displayed real and rational support rows.

Finally use Definition 3632.22. The local NameCert surface lists regular-rational readback as a root obligation, so every export from L, D, I must first pass through Q with R and the underlying dyadic tolerance route visible. □

Theorem 2668.29 (Calculus root regular readback obligation)**.** *Every derivative, integral, or limit export from an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the inherited* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route before the terminal*

$\mathsf{Real}^{\uparrow}$ *seal R is consumed. The export is a displayed calculus packet read and not a host analytic estimate, quotient real equality, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRegularReadbackObligation`

Proof. Project the carrier in Definition 3632.21. The limit row is L, the derivative row is D, the integral row is I, the common regular-rational readback row is Q, and the terminal Real boundary is R.

Apply Definition 3632.28. It already requires every real-facing value exported from L, D, I to display Q together with R and the dyadic tolerance route inherited by the regular-rational readback.

Apply Definition 3632.30. It forces a continuous-map graph read and a limit-window read to remain visible before any Real seal is exported, with Q supplying the readback used by R.

Thus no derivative, integral, or limit consumer can replace the displayed readback by a host analytic estimate, quotient real equality, or ambient Real completeness. The root obligation is exactly the route through Q and its dyadic tolerance data before R is consumed. □

Theorem 2668.30 (Calculus continuous-limit carrier budget)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, a continuous-map graph read and a limit-window read must both be displayed before any* $\mathsf{Real}^{\uparrow}$ *seal is exported. The graph row C supplies the named finite graph and pointwise replay windows, the limit row L supplies the convergence windows, and the regular-rational row Q supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the terminal real boundary R. Thus a real-facing calculus export cannot bypass C, L, Q or treat Real completeness as an ambient carrier.*

Proof. Project the carrier of Definition 3632.21. The only continuous-map graph coordinate is C, the only limit-window coordinate is L, and the only regular-rational readback coordinate is Q.

Use Definition 3632.26. It requires R, L, C, Q, H, T, P, N before a derivative or integral consumer can read the continuous-map graph.

Use Definition 3632.28. Every real-facing export from L, D, I must expose Q together with R and the dyadic tolerance route. Hence the Real seal is downstream of the displayed graph, limit, and readback rows rather than an ambient completeness principle. □

Theorem 2668.31 (Calculus derivative-integral source separation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, the derivative row D and integral row I remain separated even when they share the source rows R, L, C, Q, H, T, P, N. A derivative consumer must read local difference-quotient and remainder data from D, while an integral consumer must read partition, tagged-sum, and endpoint handoff data from I; no shared support row identifies the two sources or exports a host fundamental theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeIntegralSourceSeparation`

Proof. Project Definition 3632.21. The derivative source is the coordinate D, and the integral source is the coordinate I.

Apply Definition 3632.27. That theorem already records that the two rows use the same finite support but do not supply each other's conclusions.

The carrier list contains no coordinate for quotient function equality, a selected antiderivative, or a host fundamental theorem. Therefore shared support through R, L, C, Q, H, T, P, N preserves source separation rather than collapsing D and I. □

Theorem 2668.32 (Calculus real RegSeqRat error handoff)**.** *Every real-facing error estimate exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must first display the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the dyadic rational tolerance route inherited by that readback. Limit, derivative, and integral consumers may then hand the estimate to the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, but only after the regular-rational and dyadic data have been read inside the finite packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealRegseqratErrorHandoff`

Proof. Begin with Definition 3632.28. It makes Q mandatory before any real-facing value is exported from L, D, I.

The derivative and integral routes use the same readback discipline. Their finite quotient, remainder, partition, tagged-sum, and endpoint rows are compared through regular-rational data before a Real-facing estimate is named.

Finally apply Definition 3632.22. The local NameCert surface contains $R, L, C, D, I, Q, H, T, P, N$ and no ambient Real-completeness or host error functional. Thus the handoff to R is downstream of $\mathsf{RegSeqRat}^{\uparrow}$ and dyadic readback data in Q. □

Theorem 2668.33 (Calculus derivative-limit root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a derivative-limit read is public only after the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*, the* $\mathsf{Derivative}^{\uparrow}$ *quotient row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the dyadic tolerance ledger inherited by* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *have all been displayed. The read exports a finite BEDC derivative-limit route and not a host function space, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeLimitObligation`

Proof. Project the carrier of Definition 3632.21. The derivative source is the row D, while the limit-facing route is L and the graph replay available to a derivative consumer is C.

Use Definition 3632.26. It places every derivative consumer behind R, L, C, Q, H, T, P, N, so the graph and pointwise replay are visible before the derivative row is consumed.

Use Definition 3632.28. The regular-rational readback Q carries the dyadic tolerance route required before any Real-facing value reaches R.

The remaining rows H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host function space, quotient function equality, selected derivative operator, selected antiderivative, or ambient completeness, the derivative-limit obligation is exactly this displayed route. □

Theorem 2668.34 (Calculus integral-sum root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, an integral read may consume the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row* I *only after the* $\mathsf{Real}^{\uparrow}$ *boundary* R*,* $\mathsf{Limit}^{\uparrow}$ *route* L*,* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, dyadic tolerance ledger, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *are visible. The finite-sum row records partition, tag, sum, and endpoint handoff data; it does not export a host integration functional, quotient equality of functions, a selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusIntegralSumObligation`

Proof. Read the carrier K and isolate the integral source row I. The statement of Definition 3632.22 lists integral finite-sum handoff as one local NameCert obligation, so I is a displayed row rather than a host integral object.

Apply Definition 3632.3. It keeps the partition, tag, finite-sum, and endpoint data attached to the real boundary, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows.

Apply Definition 3632.26. If the integral consumer reads graph data, it reaches that graph through R, L, C, Q, H, T, P, N before the finite-sum handoff is public.

The derivative row D remains separate by Definition 3632.31. Thus the integral-sum obligation is the finite route through I and the displayed support rows, with no selected antiderivative, host function space, quotient equality, or ambient Real-completeness coordinate. □

Theorem 2668.35 (Calculus continuity real-readback root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a continuity-shaped Real read must display the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the dyadic tolerance ledger inherited by* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *before derivative or integral data is consumed. The structural rows* H, T, P, N *preserve this same route through* hsame*,* Cont*,* Pkg*, and* NameCert*, and no host continuity theorem, function-space equality, quotient real equality, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is exported.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuityRealReadbackObligation`

Proof. Start with Definition 3632.30. It identifies C, L, Q as mandatory displayed rows before any $\mathsf{Real}^{\uparrow}$ seal is exported from the calculus carrier.

Use Definition 3632.32. The real-facing estimate must pass through Q and the dyadic rational tolerance route before it reaches R.

Derivative and integral reads remain downstream consumers. The row D supplies only finite difference-quotient data, and the row I supplies only finite-sum data, so neither can replace the continuity-to-limit-to-readback route.

Finally read H, T, P, N. These rows give componentwise transport, displayed continuation replay, package provenance, and local naming for the same packet. The carrier therefore exports a finite continuity real-readback route and no host continuity theorem, function-space equality, quotient real equality, selected antiderivative, or ambient completeness principle. □

Theorem 2668.36 (Calculus root chain-rule window obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a chain-rule-shaped read for composed local maps must display the outer and inner* $\mathsf{ContinuousMap}^{\uparrow}$ *graph windows in* C*, the two* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient windows in* D*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The read is a finite window-composition obligation, not a host chain-rule theorem over an ambient function space.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootChainRuleWindowObligation`

Proof. Project the root carrier of Definition 3632.43. The graph data for a composed read can only be displayed through C, while derivative data can only be read through D.

Use Definition 3632.44 to place every derivative, integral, or limit consumer behind the $\mathsf{ContinuousMap}^{\uparrow}$ graph row. Use Definition 3632.33 to keep the derivative-limit read behind $R, L, C, D, Q, H, T, P, N$.

The regular-rational and dyadic rows Q, Y supply the comparison window before R is consumed. The structural rows H, T, P, N transport, replay, source, and name the same finite packet.

Thus the chain-rule-shaped obligation is a displayed composition of two local derivative windows. Since the carrier has no coordinate for host function-space equality or a global derivative operator, no ambient chain-rule theorem is exported. □

Theorem 2668.37 (Calculus root fundamental-theorem refusal boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a consumer that reads both the* $\mathsf{Derivative}^{\uparrow}$ *row* D *and the* $\mathsf{Integral}^{\uparrow}$ *row* I *must keep them as separate boundary rows over the common* $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *support. The packet may expose compatible finite source windows, but it does not identify* D *with* I*, select an antiderivative, or claim the full fundamental theorem of calculus.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootFtcBoundaryRefusal`

Proof. Project D and I from Definition 3632.43. The derivative source and integral source are distinct coordinates of the same finite BHist packet.

Apply Definition 3632.45. It states that D and I remain separate over $C, L, Q, Y, R, H, T, P, N$ and that neither row supplies the other's conclusion.

The real-facing support is controlled by Definition 3632.46. Any value read from either source must pass through Q, Y before reaching R.

Therefore the carrier records a refusal boundary: derivative and integral rows can share finite support without becoming a fundamental-theorem certificate. Selected antiderivatives, quotient equality of functions, and a host theorem relating all derivatives and integrals are absent from the packet. □

Theorem 2668.38 (Calculus root local-linearity composition)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a local-linearity composition read must route the source* $\mathsf{ContinuousMap}^{\uparrow}$ *graph windows through* C*, the two local* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient rows through* D*, the comparison windows through* $\mathsf{Limit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows* L, Q, Y*, and the terminal value through the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, with componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*. The conclusion is only that the finite local-linear windows compose inside the displayed BEDC packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLocalLinearityComposition`

Proof. Begin with the graph route. By Definition 3632.44, every local operation consumer must display C before reading derivative, integral, or limit data.

For the local-linear source, use Definition 3632.36. It gives the two derivative windows and their limit/readback support through D, L, Q, Y before R is consumed.

The rows H, T, P, N then transport, replay, source, and name this composed finite window. No row of G supplies a host function space, quotient equality of functions, selected global derivative operator, or ambient Real completeness theorem.

Thus local linearity composes only as a finite BEDC dependency route: graph windows, derivative windows, regular-rational and dyadic comparison windows, and the terminal Real boundary remain visible in the same packet. □

Definition 2668.39 (Calculus root derivative-integral carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier is the carrier*

$$K = (R, L, C, D, I, Q, H, T, P, N)$$

of Definition 3632.21 read with the following root dependency rows singled out:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary for endpoint, value, error, derivative, integral, and limit readback.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the rational comparison and error schedule used before R is consumed.*
3. *the dyadic tolerance data inherited by Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dependency row for denominator, radius, and comparison reads;*
4. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise replay row used before derivative, integral, or limit consumers read function data;*
5. *D, I, L, H, T, P, N are respectively the derivative source row, integral finite-sum row, limit route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row.*

The root classifier compares accepted packets only through these displayed BHist, hsame, Cont, $\mathsf{NameCert}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows. It has no coordinate for a host function space, quotient equality of functions, a selected derivative operator, a selected antiderivative, or ambient real completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootDerivativeIntegralCarrier`

Theorem 2668.40 (Calculus root derivative real handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, a derivative-facing real read must expose the* $\mathsf{ContinuousMap}^{\uparrow}$ *row C, the* $\mathsf{Derivative}^{\uparrow}$ *row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data inherited by Q, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row N before the* $\mathsf{Real}^{\uparrow}$ *boundary R is consumed.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootDerivativeRealHandoff`

Proof. Project the root carrier of Definition 3632.39. The derivative source is D, but D is public only as part of the same BHist packet as the graph row C, readback row Q, transport row H, replay row T, provenance row P, and local naming row N.

Use Definition 3632.33. It already places every derivative-limit read behind the $\mathsf{ContinuousMap}^{\uparrow}$ graph row, the regular-rational readback, the dyadic tolerance route, and the structural rows.

The terminal real boundary is R. Since no coordinate of the root carrier contains quotient function equality, a selected derivative operator, or ambient real completeness, the derivative-facing handoff reaches R only through the displayed finite route. □

Theorem 2668.41 (Calculus root integral RegSeqRat handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, an integral-facing read through the* $\mathsf{Integral}^{\uparrow}$ *row I must first display the* $\mathsf{ContinuousMap}^{\uparrow}$ *row C, the finite-sum and endpoint handoff data in I, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route inherited by Q, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row N. Only after this readback may the consumer reach the* $\mathsf{Real}^{\uparrow}$ *boundary R.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootIntegralRegSeqRatHandoff`

Proof. Read the integral source row I from Definition 3632.39. Its partition, tag, finite-sum, and endpoint data are displayed inside the same finite BHist packet as C, Q, H, T, P, N.

Apply Definition 3632.34. That obligation forces the integral row through the real boundary, limit route, continuous-map replay, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is therefore not optional. The consumer cannot replace Q by a host integration functional, selected antiderivative, quotient function equality, or ambient completeness theorem before reading R. □

Theorem 2668.42 (Calculus root limit nonescape)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, each limit-shaped read exported by* L *must remain inside the route through the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route inherited by* Q*, the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. It cannot escape to a host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitNonescape`

Proof. Project the limit row L from the root carrier. By construction, L is not a standalone host limit object; it is stored together with C, Q, R, H, T, P, N in the displayed calculus packet.

Use Definition 3632.30. A continuous-map graph read and a limit-window read must both be displayed before any real-facing seal is exported, and the regular-rational row Q supplies the readback used by R.

The structural rows preserve the same route through hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since the carrier has no coordinate for the excluded host principles, every limit-shaped read remains confined to the listed BEDC dependency rows. □

Definition 2668.43 (Calculus root carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *root carrier is a finite* BHist *packet*

$$G = (C, D, I, L, R, Q, Y, H, T, P, N)$$

whose rows are the $\mathsf{ContinuousMap}^{\uparrow}$ *graph and replay surface* C*, the* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient row* D*, the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row* I*, the* $\mathsf{Limit}^{\uparrow}$ *window row* L*, the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and denominator ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. The root classifier compares accepted packets only through these finite rows. It has no coordinate for a host limit operator, quotient equality of functions, a selected derivative operator, a selected antiderivative, or ambient Real completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootCarrier`
Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootCarrier admission`

Theorem 2668.44 (Calculus root ContinuousMap handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, any derivative, integral, or limit consumer must display the* $\mathsf{ContinuousMap}^{\uparrow}$ *row* C *before it reads* D*,* I*, or* L *and before it consumes the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *rows. The handoff is a finite graph-and-replay read through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, not an ambient function space or host continuity theorem.*

Proof. Project the root carrier of Definition 3632.43. The only coordinate that names graph and pointwise replay data is the row C, so calculus consumers cannot read derivative, integral, or limit data as graph data unless C is visible.

Use Definition 3632.26. It already places derivative and integral consumers behind the continuous-map graph route of the calculus packet.

The rows H, T, P, N preserve that same graph route under transport, replay, provenance, and local naming. Since G has no coordinate for a host function space or host continuity theorem, the handoff remains the displayed $\mathsf{ContinuousMap}^{\uparrow}$ row. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootContinuousMapHandoff`

Theorem 2668.45 (Calculus root derivative-integral boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, the* $\mathsf{Derivative}^{\uparrow}$ *row* D *and the* $\mathsf{Integral}^{\uparrow}$ *row* I *are separate boundary rows over the common support* $C, L, Q, Y, R, H, T, P, N$*. A derivative read may use finite difference-quotient data from* D*, and an integral read may use finite-sum data from* I*, but neither row supplies the other's conclusion or a host fundamental theorem.*

Proof. Read the two source rows from Definition 3632.43. The derivative source is D, and the integral source is I; they are distinct coordinates of the finite packet.

Apply Definition 3632.31. It states that derivative and integral sources remain separated even when they share the real, limit, continuous-map, regular-rational, transport, replay, provenance, and local naming support.

The dyadic tolerance row Y and readback row Q only provide comparison data before the terminal Real boundary R is consumed. They do not identify D with I and do not contain a selected antiderivative, quotient function equality, or host fundamental theorem. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootDerivativeIntegralBoundary`

Theorem 2668.46 (Calculus root Real-seal nonescape)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, every Real-facing export from derivative, integral, or limit data must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The export remains inside the displayed finite packet and cannot escape to host limit operators, quotient function equality, or ambient Real completeness.*

Proof. Start with the root carrier. The terminal Real boundary is R, while Q and Y are the only regular-rational and dyadic tolerance rows available before that boundary is read.

Use Definition 3632.28. It requires every real-facing value exported from the calculus limit, derivative, and integral rows to display regular-rational readback before the Real boundary is consumed.

Use Definition 3632.42 for limit-shaped reads and Definition 3632.45 for derivative and integral source separation. Together they keep the route inside $C, D, I, L, Q, Y, R, H, T, P, N$.

Because the root carrier has no coordinate for host limit operators, quotient function equality, selected derivative or antiderivative operators, or ambient Real completeness, no Real-facing export can bypass Q, Y on the way to R. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRealSealNonescape`

Definition 2668.47 (Calculus root operation ledger)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root operation ledger attached to an accepted carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$ *is the displayed row inventory*

$$\mathsf{OpLedger}_{\mathsf{Calculus}^{\uparrow}}(G) = (C, D, I, L, R, Q, Y, H, T, P, N).$$

Here C *is the continuous-map graph and replay row,* D *is the derivative difference-quotient row,* I *is the integral finite-sum row,* L *is the limit-window row,* R *is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary,* Q *is*

the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* Y *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* H *is componentwise* hsame *transport,* T *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row. The ledger has no row for a host function space, quotient equality of functions, a selected antiderivative, a selected derivative operator, or ambient completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootOperationLedger`

Theorem 2668.48 (Calculus root operation ledger exhaustion)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, every derivative, integral, limit, or Real-facing operation read is exhausted by the operation ledger of Definition 3632.47. A consumer may read derivative data only through* $C, D, Q, Y, R, H, T, P, N$*, integral data only through* $C, I, Q, Y, R, H, T, P, N$*, and limit data only through* $C, L, Q, Y, R, H, T, P, N$*. No operation read may replace those rows by a host function space, quotient equality of functions, selected antiderivatives, selected derivative operators, or ambient completeness.*

Proof. Begin with the root carrier of Definition 3632.43. It displays exactly the rows $C, D, I, L, R, Q, Y, H, T, P, N$, and Definition 3632.47 only repackages that same finite inventory as the operation ledger.

Apply Definition 3632.44. Every derivative, integral, and limit consumer must display C before reading D, I, or L.

For derivative and integral reads, apply Definition 3632.45. It keeps D and I as separate source rows over the common support and refuses a host fundamental theorem or selected antiderivative.

For limit and Real-facing reads, apply Definition 3632.42 and Definition 3632.46. They force the read through Q, Y before R is consumed and leave no coordinate for host limits, quotient function equality, selected derivative or antiderivative operators, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationLedgerExhaustion`

Theorem 2668.49 (Calculus root unblock Real operation ledger)**.** *The root* $\mathsf{Calculus}^{\uparrow}$ *operation ledger unblocks downstream derivative, integral, limit, and completion consumers only as the finite BEDC route*

$$C, D, I, L, R, Q, Y, H, T, P, N.$$

Any downstream consumer that asks for a Real-facing calculus value must display the relevant operation row D*,* I*, or* L*, the graph row* C*, the regular-rational readback* Q*, the dyadic tolerance ledger* Y*, the terminal Real boundary* R*, and the structural rows* H, T, P, N*. The ledger does not export ambient function spaces, quotient equality, selected antiderivatives, selected derivative operators, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the operation ledger from Definition 3632.47. The only public rows available to a downstream consumer are the displayed finite rows $C, D, I, L, R, Q, Y, H, T, P, N$.

Use the derivative-integral boundary of Definition 3632.45 for operation separation. It prevents the derivative row from supplying integral conclusions and prevents the integral row from supplying derivative conclusions.

Use the Real-seal nonescape theorem Definition 3632.46. It requires every Real-facing export from derivative, integral, or limit data to pass through Q, Y before R is consumed.

Finally apply Definition 3632.48. The exhaustion statement identifies the ledger as the full consumer-facing route, so the root unblock surface is a finite operation ledger and not a host analytic structure or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRealOperationLedger`

Theorem 2668.50 (Calculus root Real-completion operation handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, any consumer that combines derivative, integral, or limit rows with a Real-facing value must route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *before reading the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*. The handoff is the Phase Real and Completion route of the finite root carrier, not an ambient completion or host analytic-value principle.*

Proof. Apply Definition 3632.45 to keep the derivative and integral rows separate over the common support.

Use Definition 3632.46 and Definition 3632.48. These force every derivative, integral, limit, or Real-facing read through Q, Y before R is public.

Finally apply Definition 3632.51 for the completion-facing derivative window. The structural rows H, T, P, N preserve the same finite route, so the consumer reaches Phase Real and Completion only through the displayed root ledger. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRealCompletionOperationHandoff`

Theorem 2668.51 (Calculus root local linearization Cauchy handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a local-linearization consumer that asks for a Cauchy-facing derivative window must display the* BHist *derivative row* D*, the continuous graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The handoff is a finite derivative-to-regular-readback route and not a host differentiability theorem, quotient equality of functions, or ambient Cauchy completeness claim.*

Proof. Project the root operation ledger. The derivative source for local linearization is the displayed row D, and any graph data used by that derivative source is routed through C.

Apply Definition 3632.40. It places every derivative-facing real read behind C, D, Q, H, T, P, N before R is consumed. Apply Definition 3632.46 to keep the same read behind the dyadic tolerance ledger Y and the regular-rational row Q.

The structural rows H, T, P, N preserve the same finite packet under hsame, Cont, Pkg, and local naming. Because the carrier has no coordinate for host differentiability, quotient function equality, or ambient completion, the Cauchy-facing handoff is exactly the displayed derivative, readback, tolerance, and Real-seal route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLocalLinearizationCauchyHandoff`

Theorem 2668.52 (Calculus local operation nonescape)**.** *Every public* $\mathsf{Calculus}^{\uparrow}$ *local operation read is confined to the displayed* BHist *rows for source graph data or finite sums,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming. No derivative, integral, or limit conclusion may replace those rows by a host function space, host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocalOperationNonescape`

Proof. Read the carrier inventory from Definition 3632.21. The local operations are stored only in the derivative, integral, and limit rows, and all Real-facing reads pass through the regular-rational and dyadic rows before the terminal seal is consumed.

Apply Definition 3632.44 to keep every operation behind its graph-and-replay row. Apply Definition 3632.45 and Definition 3632.13 to keep derivative and integral sources distinct.

The Real-facing boundary is controlled by Definition 3632.46 together with Definition 3632.14. The source-window requirement is exactly Definition 3632.15.

Thus each operation is exported only through the displayed BEDC packet. The excluded host and quotient objects are absent from the carrier, so there is no accepted escape route outside the finite local operation surface. □

Theorem 2668.53 (Calculus root continuity composition ledger)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a continuity-composition read for derivative, integral, or limit consumers must first display the* BHist *continuous-map row* C*, the operation row among* D, I, L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, the terminal* $\mathsf{Real}^{\uparrow}$ *row* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*. The composition ledger exports no host continuity theorem, function-space equality, selected operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the continuous-map row C from the root carrier. By Definition 3632.44, every derivative, integral, or limit consumer must expose C before it reads D, I, or L.

For the operation side, apply Definition 3632.48. It separates derivative reads through D, integral reads through I, and limit reads through L, while keeping all three behind the common regular-rational, dyadic, Real, transport, replay, provenance, and naming rows.

The Real-facing portion is controlled by Definition 3632.46. Hence the continuity-composition ledger is a finite route through $C, D, I, L, Q, Y, R, H, T, P, N$, with no extra coordinate from which a host continuity theorem or ambient function-space equality could be read. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootContinuityCompositionLedger`

Theorem 2668.54 (Calculus root Riemann-sum Real seal)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a Riemann-sum-facing integral read must display the* BHist *finite-sum row* I*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*, and the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *before any integral value is public. The seal is a finite Riemann row over L10 readback data and not a selected antiderivative, host integral functional, quotient equality of functions, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRiemannSumRealSeal`

Proof. Project the integral source row I from the root carrier. It is the only row carrying finite-sum and endpoint handoff data for an integral-shaped read.

Apply Definition 3632.41. It forces the integral row through C, I, Q, H, T, P, N and then through the terminal Real boundary R. Apply Definition 3632.46 to keep the same route behind the dyadic tolerance ledger Y before R is consumed.

The derivative row D and limit row L remain separate by Definition 3632.45 and Definition 3632.48. Thus the Riemann-sum Real seal is exactly the finite route through $I, C, Q, Y, H, T, P, N, R$, with no selected antiderivative, host integration functional, quotient equality, or ambient completeness row. □

Theorem 2668.55 (Calculus root limit-derivative-integral triad)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, the derivative row* D*, integral row* I*, and limit row* L *form one finite root triad over the continuous-map source* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R*, and the structural rows* H, T, P, N*. A consumer of any one of* D, I, L *must expose the whole route* $C, D, I, L, Q, Y, R, H, T, P, N$ *before exporting a Real-facing calculus value.*

Proof. Apply Definition 3632.44. It places every derivative, integral, and limit read behind the shared continuous-map row C.

Apply Definition 3632.45 to keep the derivative and integral coordinates distinct, and use Definition 3632.48 for the limit coordinate and the common ledger inventory.

Finally apply Definition 3632.54 and Definition 3632.46. The integral and Real-seal routes force the same Q, Y, R, H, T, P, N readback boundary, so the triad is the displayed finite carrier surface and not an ambient analytic operator family. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitDerivativeIntegralTriad`

Theorem 2668.56 (Calculus root common Real readback)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, derivative, integral, and limit consumers share the same Real readback route: the chosen operation row among* D, I, L *is read only together with* C, Q, Y, R, H, T, P, N*. No operation row exports a Real value through a separate quotient representative, selected limiting object, or host completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootCommonRealReadback`

Proof. Start from Definition 3632.46. It requires every Real-facing calculus export to pass through Q, Y before the terminal row R is consumed.

Use Definition 3632.44 to attach that readback to C, and use Definition 3632.48 to identify the only operation rows as D, I, L.

The Riemann-sum boundary of Definition 3632.54 supplies the integral case, while Definition 3632.45 prevents that case from being reused as derivative data. The shared readback is therefore $C, D, I, L, Q, Y, R, H, T, P, N$ with the selected operation coordinate named by the consumer. □

This common readback consumes the Real and completion chain through chapter 38, chapter 301, and chapter 291. Those sibling rows provide the terminal seal, regular rational readback, and tolerance ledger for the calculus root route, while importing no ambient Real completeness theorem or host function space.

Theorem 2668.57 (Calculus root operation separation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, the derivative, integral, and limit obligations are separated by the operation rows* D, I, L *while remaining tied to the same continuous-map, regular-rational, dyadic, Real, transport, replay, provenance, and naming rows* C, Q, Y, R, H, T, P, N*. Thus no accepted calculus consumer may infer a derivative claim from* I *or* L*, an integral claim from* D *or* L*, or a limit claim from* D *or* I *without reopening the displayed finite route.*

Proof. Apply Definition 3632.45. It already separates derivative reads through D from integral reads through I.

Apply Definition 3632.48. The exhaustion ledger adds the limit row L to the same separated inventory and gives no hidden coordinate from which one operation can supply another.

Use Definition 3632.44 and Definition 3632.46 to keep all three operation rows attached to C, Q, Y, R, H, T, P, N. The carrier route is therefore common at the readback boundary and separated at the operation boundary. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationSeparation`

Theorem 2668.58 (Calculus root chain-rule ContinuousMap RegSeqRat scope)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, a composable pair of local-linearization ledgers whose graph replay is accepted by* $\mathsf{ContinuousMap}^{\uparrow}$ *through* C *and whose dyadic error budgets descend through* $\mathsf{RegSeqRat}^{\uparrow}$ *via* Q, Y *produces a* $\mathsf{Real}^{\uparrow}$ *readback scope for the chain-rule window. The consumer may read only the finite route* $C, D, Q, Y, R, H, T, P, N$ *together with the two composable local-linearization rows; it may not infer the full fundamental theorem of calculus, a global derivative operator, a selected antiderivative, quotient equality of functions, or ambient Real completeness.*

Proof. Begin with the calculus root ContinuousMap handoff of Definition 3632.44. It forces the graph replay for each local-linearization row through the displayed continuous-map coordinate C before the derivative row D is consumed.

Use the derivative-integral boundary of Definition 3632.45. The chain-rule window remains on the derivative side of the root carrier and does not borrow an integral row or a host antiderivative claim.

Next apply the regular-rational and dyadic readback route supplied by Definition 3632.53 and Definition 3632.56. These keep the error budgets in Q, Y before any terminal $\mathsf{Real}^{\uparrow}$ readback row R is exposed.

Finally use Real-seal nonescape, Definition 3632.46. The resulting scope is the displayed finite route through $C, D, Q, Y, R, H, T, P, N$ for the composable local-linearization pair, not the full fundamental theorem of calculus or any ambient analytic completeness principle. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootChainRuleContinuousMapRegSeqRatScope`

Theorem 2668.59 (Calculus root operation readback tail chain-rule boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, any chain-rule readback that uses the derivative row* D *must retain the common route* $C, D, Q, Y, R, H, T, P, N$*. The derivative evidence cannot be supplied by the integral row* I *or the limit row* L*; those rows remain available only through their own integral and limit obligations.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationReadbackTailChainRuleBoundary`

Proof. Apply Definition 3632.58. It fixes the chain-rule window on the displayed derivative route through $C, D, Q, Y, R, H, T, P, N$ together with the composable local-linearization rows.

Next use Definition 3632.57. The separated operation ledger prevents an integral row I or a limit row L from being consumed as derivative evidence without reopening the corresponding operation obligation.

Finally apply Definition 3632.56. The Real-facing readback is the common finite BEDC route through the continuous-map, derivative, regular-rational, dyadic, Real, transport, replay, provenance, and naming rows. By Real-seal nonescape this route is not a host analytic chain-rule operator, selected integral, selected limit, quotient function equality, or ambient completeness principle. □

Lemma 2668.60 (Calculus derivative and Riemann-integral dependency route)**.** *Every substantive* $\mathsf{Calculus}^{\uparrow}$ *consumer of a derivative or integral conclusion reads the derivative branch through chapter 122, the Riemann finite-sum branch through chapter 1526, and the shared endpoint readback through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The route is therefore a displayed BEDC dependency through derivative windows, tagged Riemann sums, regular rational readback, and real-name sealing; it is not a host function space, a host measure-space integral, a selected antiderivative, or quotient equality of functions.*

Proof. Project the $\mathsf{Calculus}^{\uparrow}$ local-linearization and root carrier rows already named in Definition 3632.1 and Definition 3632.5. Their derivative-facing coordinates are the $\mathsf{Derivative}^{\uparrow}$ rows, and their integral-facing coordinates are the $\mathsf{RiemannIntegral}^{\uparrow}$ or finite-sum rows.

The shared numerical readback is not supplied by a host analytic object. It passes through the regular rational and real-name rows before a public endpoint value is exposed.

The existing nonescape theorems in the chapter exclude host function spaces, host integration functionals, selected antiderivatives, quotient equality of functions, and ambient completeness. Hence the dependency route for calculus consumers is the finite $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route stated above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannIntegralDependencyRoute`

Theorem 2668.61 (Calculus derivative-integral dependency factorization)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer that combines a derivative conclusion with a Riemann-integral conclusion factors its public read through the* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before exposing endpoint equality or an estimate. In the notation of Definition 3632.1 and Definition 3632.5, the derivative branch consumes the displayed derivative rows, the integral branch consumes the displayed finite-sum rows, and the common endpoint readback consumes the regular rational and real-name rows before any equality or estimate is public.*

Proof. Start with the dependency route of Definition 3632.60. It already separates derivative-facing reads through chapter 122 from Riemann finite-sum reads through chapter 1526.

The local-linearization carrier Definition 3632.1 supplies the derivative-side window, while the root public package Definition 3632.5 supplies the shared public rows. Thus a consumer combining the two conclusions must first read the derivative and integral branches as displayed BEDC rows.

The endpoint or estimate then passes through the regular rational readback and terminal Real seal. This is the same nonescape boundary used by Definition 3632.52, Definition 3632.10, and Definition 3632.46.

Those nonescape statements exclude host function spaces, selected antiderivatives, quotient function equality, host integration functionals, and ambient completeness. Hence no combined derivative-integral conclusion can expose an endpoint equality or estimate before the finite $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ factorization has been consumed. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannIntegralDependencyFactorization`

Theorem 2668.62 (Calculus finite-limit seal dependency)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *consumer that turns derivative or integral endpoint data into a limit-facing read must first pass through the finite* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *route*

$$M, W, R, D, E$$

of Definition 5722.5. The calculus consumer may read derivative windows through $\mathsf{Derivative}^{\uparrow}$ *and finite tagged sums through* $\mathsf{RiemannIntegral}^{\uparrow}$*, but its public endpoint handoff is limit-facing only*

after the displayed modulus, window, readback, tolerance, and terminal real-seal rows have been consumed.

Proof. Begin with Definition 3632.60. It separates the derivative branch, the Riemann finite-sum branch, and the shared regular-rational and real-name readback before any calculus endpoint is public.

Use Definition 3632.61 to combine the derivative and integral branches. The factorization leaves the endpoint read behind the same $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ support, not behind a host function space or selected antiderivative.

Finally apply Definition 5722.5. Its finite route M, W, R, D, E is the Real-facing limit seal. Therefore a calculus consumer that turns the endpoint into a limit-facing read must expose that seal before the endpoint can be consumed as public limit data. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealDependency`

Theorem 2668.63 (Calculus finite-limit seal public handoff)**.** *Any public* $\mathsf{Calculus}^{\uparrow}$ *endpoint equality or estimate that combines derivative and Riemann-integral branches may be exported only after the* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *route*

$$M, W, R, D, E$$

of Definition 5722.5, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal have all been exposed. The exported handoff is therefore the displayed finite route through* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and not a host function space, selected antiderivative, host integration functional, quotient function equality, or ambient completeness assertion.*

Proof. Start from the dependency route of Definition 3632.60. It already places derivative reads, Riemann finite-sum reads, regular-rational readback, and real-name sealing on displayed BEDC rows.

Apply Definition 3632.61. Any combined endpoint equality or estimate must first factor through the derivative branch, the Riemann-integral branch, and the shared $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ readback before it is public.

Now use Definition 3632.62. When that endpoint read becomes limit-facing, the finite route M, W, R, D, E of the $\mathsf{FiniteLimitSeal}^{\uparrow}$ seal is exposed before export.

The non-escape boundary is unchanged: the handoff supplies no host function space, selected antiderivative, host integration functional, quotient function equality, or ambient completeness principle. Hence later calculus statements can consume only the public finite-limit-seal handoff displayed above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealPublicHandoff`

The $\mathsf{Calculus}^{\uparrow}$ plane-boundary branch reads vector-calculus integration through the Green theorem certificate before treating the result as a calculus-facing Riemann-integral dependency. This keeps the planar region, boundary curve, differential-form row, scalar curl or divergence row, and tagged integration handoff visible as a finite route.

Thus any $\mathsf{Calculus}^{\uparrow}$ consumer of plane boundary integration passes through chapter 7971 and chapter 5720 before reading the $\mathsf{RiemannIntegral}^{\uparrow}$ packet of chapter 1526.

Lemma 2668.64 (Calculus finite-limit seal route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *endpoint consumer that turns derivative, integral, or root limit data into a limit-facing read must pass through chapter 5722 and the finite* M, W, R, D, E *route of Definition 5722.5 before the public* $\mathsf{Real}^{\uparrow}$ *endpoint is exposed.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealDependencyRoute`

Proof. Start with the derivative and integral dependency route of Definition 3632.60. It already forces calculus endpoint reads through the displayed $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Now read the finite-limit seal route. Theorem Definition 5722.5 supplies the M, W, R, D, E handoff for real-facing limit reads, so the calculus endpoint consumer must expose that seal before treating the endpoint as public limit-facing data. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealRoute`

The $\mathsf{Calculus}^{\uparrow}$ packet supplies the local difference-quotient and derivative-window route consumed by chapter 5135, chapter 6674, and chapter 6763. These sibling certificates read the same finite calculus window before exporting finite-difference tables, Taylor-polynomial coefficient rows, or mean-value comparison data. The $\mathsf{Calculus}^{\uparrow}$ derivative-facing route reads finite computations through chapter 5135 before using the derivative carrier of chapter 122. Its Real-facing rows remain tied to the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Consequently the finite-difference derivative route is a downstream consumer of the L10 real-completion spine before any derivative endpoint is read: the dyadic tolerance rows of chapter 291 feed the regular rational sequence rows of chapter 301, and those rows feed the real-completion seal of chapter 38. The derivative carrier may inspect finite differences only after that spine has supplied the tolerance-visible $\mathsf{Real}^{\uparrow}$ endpoint. The $\mathsf{Calculus}^{\uparrow}$ packet supplies derivative and limit local-linearization rows to the located-compact boundary only after the finite-net and real-readback routes are visible. A $\mathsf{LocatedCompact}^{\uparrow}$ consumer of a calculus window must first read the local-linearization carrier of Definition 3632.1 and the Cauchy-facing derivative handoff of Definition 3632.51; it then receives the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{ContinuousMap}^{\uparrow}$ graph replay row, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any compactness-facing finite-net row may be consumed.

Theorem 2668.65 (Calculus located-compact dependency route)**.** *Every* $\mathsf{LocatedCompact}^{\uparrow}$ *consumer of a* $\mathsf{Calculus}^{\uparrow}$ *derivative, limit, or local-linearization row factors through the finite calculus route displayed by Definition 3632.1, Definition 3632.51,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it reaches the located-compact packet of chapter 1488. The handoff exports a finite graph-and-readback window for compactness use, not open-cover compactness, a selected finite subcover, quotient function equality, or ambient Real completeness.*

Proof. Project the calculus local-linearization carrier. Its derivative and limit rows expose graph data through $\mathsf{ContinuousMap}^{\uparrow}$, regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, and endpoint values through the $\mathsf{Real}^{\uparrow}$ boundary.

Use Definition 3632.51 for the Cauchy-facing derivative window. That route displays the derivative row, graph row, readback row, dyadic tolerance ledger, transport, replay, provenance, and local naming before the Real boundary is consumed.

Only after those rows are visible may the consumer enter chapter 1488. The located-compact packet therefore receives finite calculus data suitable for its finite-net and locatedness rows, with no host compactness theorem or selected subcover coordinate added by the calculus route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocatedCompactDependencyRoute`

The $\mathsf{Calculus}^{\uparrow}$ packet consumes the Cauchy and real-completion route before derivative, integral, or limit rows can expose a real-valued endpoint. Its root carrier therefore sits downstream of the regular rational readback of chapter 301, the terminal real boundary of chapter 38, and the generic limit packet of chapter 1678.

Theorem 2668.66 (Calculus Cauchy dependency route)**.** *Every* Calculus$^{\uparrow}$ *derivative, integral, or limit read that exposes a real-valued limit factors through the finite Cauchy route displayed by the local-linearization carrier of Definition 3632.1 and the root local-linearization Cauchy handoff of Definition 3632.51. In particular, the read must display the* RegSeqRat$^{\uparrow}$ *readback row and the terminal* Real$^{\uparrow}$ *seal before any calculus operation can export a real-valued endpoint. The route supplies finite Cauchy-window data for the* Limit$^{\uparrow}$ *consumer, not an ambient completeness theorem, a host limit operator, quotient equality of functions, or a selected real representative.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCauchyDependencyRoute`

Proof. Project the calculus local-linearization carrier. Its derivative, integral, and limit rows are interpreted only together with the regular rational readback row and the terminal real seal.

Apply the root local-linearization Cauchy handoff. That theorem keeps the derivative-facing calculus window behind the displayed Cauchy rows, dyadic tolerance ledger, transport, replay, provenance, and local naming before any Real$^{\uparrow}$ endpoint is public.

Finally read the sibling route through chapter 301, chapter 38, and chapter 1678. These chapters supply the finite readback, real boundary, and limit consumer rows consumed by calculus; none supplies an ambient host completeness or quotient-function equality coordinate. □

Theorem 2668.67 (Calculus Cauchy dependency real-completion readback)**.** *Every accepted* Calculus$^{\uparrow}$ *real-valued derivative, integral, or limit endpoint read exposes the* RegSeqRat$^{\uparrow}$ *readback row, the terminal* Real$^{\uparrow}$ *seal, and the* Limit$^{\uparrow}$ *consumer row in that order. The route exports only finite Cauchy-window data from the local-linearization carrier and never exports an ambient Real-completion theorem, a selected limit, quotient equality of functions, or a host endpoint object.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCauchyDependencyRealCompletionReadback`

Proof. Project the local-linearization carrier of Definition 3632.1. Its derivative, integral, and limit endpoints are visible only through the displayed Cauchy-window rows, so no endpoint can bypass the finite route.

Apply Definition 3632.66. It already places every real-valued calculus read behind the regular rational readback and the terminal Real seal before any endpoint is public.

Now read the sibling routes in order. The RegSeqRat$^{\uparrow}$ chapter supplies the regular rational readback row, the Real$^{\uparrow}$ chapter supplies the terminal seal, and the Limit$^{\uparrow}$ chapter consumes the resulting finite Cauchy-window data as a limit-facing row. Since those are the only exposed rows, the readback cannot import host completeness, selected representatives, quotient-function equality, or an ambient endpoint. □

The Calculus$^{\uparrow}$ packet consumes the shared finite limit route whenever a derivative or integral read exposes a limiting value. This route is the same public route recorded by chapter 1678, not a host limit operator or a selected real representative.

Theorem 2668.68 (Calculus Limit dependency route)**.** *Every* Calculus$^{\uparrow}$ *finite derivative or integral read that consumes a limiting value must pass through the shared* Limit$^{\uparrow}$ *route of chapter 1678, together with the* RegSeqRat$^{\uparrow}$ *readback rows of chapter 301, the* StreamName$^{\uparrow}$ *schedule rows of chapter 126, the* DyadicRatCore$^{\uparrow}$ *tolerance ledgers of chapter 291, and the downstream* Real$^{\uparrow}$ *seal rows of chapter 38. The calculus carrier cannot replace that route by host limit operators, quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitDependencyRoute`

Proof. Project a calculus read that asks for a derivative or integral endpoint. The endpoint is not read as a host real value; it is routed through the finite limit consumer surface.

Apply the public $\mathsf{Limit}^{\uparrow}$ handoff of Definition 1678.3, Definition 1678.4, and Definition 1678.8. These theorems force the same StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming route before a limiting value can be consumed.

The surrounding calculus rows can then read derivative and integral data through that shared route. Since the route contains no host limit operator, quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient-completeness coordinate, no calculus consumer can export one by bypassing $\mathsf{Limit}^{\uparrow}$. □

Theorem 2668.69 (Calculus Limit dependency refusal boundary)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative or integral endpoint read that omits one of the* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *seal rows is outside the BEDC calculus limit-consumer surface. Such a read cannot be retyped as an accepted calculus limit consumer by replacing the missing row with a host limit operator, quotient stream equality, a selected representative, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitDependencyRefusalBoundary`

Proof. Apply Definition 3632.68. It requires every finite derivative or integral endpoint read that consumes a limiting value to pass through the public $\mathsf{Limit}^{\uparrow}$ route and its RegSeqRat, StreamName, DyadicRatCore, and Real handoffs.

If one of those seal rows is absent, the route supplied by Definition 1678.3, Definition 1678.4, and Definition 1678.8 is not available. The attempted endpoint read therefore lacks the displayed finite limit source that the calculus carrier consumes.

The omitted row cannot be supplied by the host side. The cited handoffs exclude host limit operators, quotient stream equality, selected representatives, countable-choice selectors, and ambient completeness rows from the accepted $\mathsf{Limit}^{\uparrow}$ route, so the calculus consumer has no alternate BEDC row through which to retype the missing dependency. □

The $\mathsf{Calculus}^{\uparrow}$ limit route is read only after the real-completion display has fixed its readback rows. The child route below records the sibling order used by derivative, integral, and endpoint consumers: $\mathsf{Limit}^{\uparrow}$ supplies the limit-facing consumer surface, while $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ supply the terminal seal, regular rational readback, and schedule rows that make the limit read finite.

Theorem 2668.70 (Calculus Limit real handoff)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or endpoint consumer that cites the* $\mathsf{Limit}^{\uparrow}$ *route must first display the* $\mathsf{Real}^{\uparrow}$ *terminal seal, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{StreamName}^{\uparrow}$ *schedule row used by that limit route. The handoff is the finite dependency lattice through chapter 1678, chapter 38, chapter 301, and chapter 126; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealHandoff`

Proof. Start with the existing calculus limit dependency route of Definition 3632.68. It already places every derivative or integral endpoint read behind the shared $\mathsf{Limit}^{\uparrow}$ consumer surface.

Now read the sibling rows that make this consumer surface finite. The $\mathsf{StreamName}^{\uparrow}$ schedule names the displayed stream positions, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal consumed by the limit-facing row.

The calculus packet can therefore cite $\mathsf{Limit}^{\uparrow}$ only after those readback rows are visible. Since the route contains no coordinate for a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the handoff remains a finite BEDC dependency route. □

This terminal-export theorem is the canonical public handoff consumed by the $\mathsf{Calculus}^{\uparrow}$ closure ledger, so the ledger points to this route without changing the independent verification grade recorded for the chapter.

Theorem 2668.71 (Calculus Limit real derivative readback)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative endpoint consumer that exports a Real-facing derivative value must pass through* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *before the endpoint may be read as a* $\mathsf{Real}^{\uparrow}$ *value. The derivative readback is the finite BEDC route through limit consumption, stream scheduling, regular rational readback, dyadic rational coordinates, and terminal Real sealing; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealDerivativeReadback`

Proof. Start with the real handoff of Definition 3632.70. It already places derivative endpoint reads behind the $\mathsf{Limit}^{\uparrow}$ consumer surface together with $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Now include the existing calculus limit dependency route of Definition 3632.68. Its endpoint schedule is read through rational coordinates, so the $\mathsf{DyadicRatCore}^{\uparrow}$ row must be visible before the regular readback can be sealed as a Real-facing derivative value.

The route therefore exposes only the displayed limit, stream, dyadic rational, regular readback, and Real seal rows. Since no coordinate supplies a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the derivative endpoint remains a finite BEDC readback. □

Theorem 2668.72 (Calculus Limit real terminal export)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or endpoint theorem that exports a Real-facing value through the* $\mathsf{Limit}^{\uparrow}$ *route must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any downstream calculus consumer may read that value. The export composes the real handoff of Definition 3632.70, the derivative readback of Definition 3632.71, and the limit dependency route of Definition 3632.68; it is not a host limit, quotient stream equality, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealTerminalExport`

Proof. Begin with Definition 3632.70. It places every derivative, integral, or endpoint consumer behind the $\mathsf{Limit}^{\uparrow}$ surface together with the StreamName schedule, RegSeqRat readback, and terminal Real seal.

Apply Definition 3632.71 to the Real-facing endpoint case. That theorem inserts the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row before the regular readback is sealed as a Real-facing value.

Finally use Definition 3632.68 to identify the common calculus limit route consumed by derivative, integral, and endpoint theorems. The same displayed limit surface is the only route by which those consumers can reach a Real-facing value.

The terminal export is therefore the finite row order $\mathsf{Limit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Because none of the composed routes contains a host limit operator, quotient stream equality, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, no downstream calculus consumer can bypass the displayed rows. □

The $\mathsf{Calculus}^{\uparrow}$ product-distance route is read through the Cauchy product metric packet before derivative-product or endpoint consumers expose any paired Real data. This child route records that the calculus hub uses the finite product-metric surface of chapter 3551 rather than a host product metric.

Theorem 2668.73 (Calculus product metric forward link)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative-product, endpoint, or paired-distance consumer that uses product-distance data must first display the* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet through* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before reading either component as a* $\mathsf{Real}^{\uparrow}$ *value. The route supplies the finite product-distance ledger, paired stream windows, regular rational readbacks, dyadic tolerance coordinates, metric distance row, and componentwise terminal Real seals. It supplies no host product metric, quotient stream equality, selected product limit, detached Real carrier, detached Real pair, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardLink`

Proof. Begin with the real handoff of Definition 3632.70. It already forces endpoint reads through $\mathsf{Limit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a calculus consumer may expose a Real-facing value.

Now insert the product-distance sibling. The $\mathsf{CauchyProductMetric}^{\uparrow}$ carrier of chapter 3551 reads two $\mathsf{CauchyProduct}^{\uparrow}$ rows, the $\mathsf{Met}^{\uparrow}$ distance row, paired stream windows, regular rational readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The terminal Real seal is componentwise. A paired-distance read may expose the left component and the right component only after the product-metric sibling has displayed the two carried streams, their dyadic comparison ledger, and the Real-facing seals for those same coordinates. It cannot replace that route by a detached host product metric or a selected product limit whose components are later read as independent Real values.

Therefore a derivative-product, endpoint, or paired-distance consumer can use product-distance data only after that finite route is visible. Since no row in the calculus packet or the product-metric sibling carries a host product metric, quotient stream equality, selected product limit, detached Real carrier, detached Real pair, or ambient completeness theorem, the forward link remains a BEDC dependency route. □

Theorem 2668.74 (Calculus product metric seal pairing)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative-product, endpoint, or paired-distance consumer that exposes two component* $\mathsf{Real}^{\uparrow}$ *seals must read both seals through the same* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet and the shared route*

$$\mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The paired seals are therefore componentwise but not detached: the consumer may inspect the left and right $\mathsf{Real}^{\uparrow}$ *rows only as terminal rows of one displayed product-metric packet. It has no route through detached Real pairs, host product metrics, selected product limits, quotient stream equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricSealPairing`

Proof. Start with the forward link of Definition 3632.73. It places every derivative-product, endpoint, or paired-distance consumer behind the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet before either component Real value is read.

Next apply the finite modulus route of Definition 3632.75. This identifies the common route through the two $\mathsf{CauchyProduct}^{\uparrow}$ source rows, the $\mathsf{Met}^{\uparrow}$ distance row, paired $\mathsf{StreamName}^{\uparrow}$ windows,

paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Finally use the handoff theorem Definition 3632.76. The handoff keeps both component seals downstream of the same product-metric packet, so a calculus consumer cannot split the pair into detached Real data and later attach a host product metric, selected product limit, quotient stream equality, or ambient completeness witness. □

Theorem 2668.75 (Calculus product metric finite modulus route)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *paired-distance read that reaches component* $\mathsf{Real}^{\uparrow}$ *seals must expose the finite product-metric modulus rows before those component seals are read. Concretely, the read factors through the* $\mathsf{CauchyProductMetric}^{\uparrow}$ *route of chapter 3551, the two* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger, the* $\mathsf{Met}^{\uparrow}$ *distance row, and only then the terminal component* $\mathsf{Real}^{\uparrow}$ *seals. It does not factor through a host product metric, quotient stream equality, ambient completeness, or a detached product Real.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardModulusRoute`

Proof. Start with the forward link of Definition 3632.73. It already makes the product-distance sibling visible before any calculus derivative-product, endpoint, or paired-distance consumer can inspect component Real data.

Next read the product-metric sibling through Definition 3551.4. That boundary orders the route by the two product sources, paired stream windows, regular rational readbacks, dyadic product-distance ledger, metric row, and terminal Real seal. Thus the finite modulus rows are not optional decoration: they are the rows through which the paired distance is displayed before the component seals are available.

The excluded coordinates are the same on both sides of the handoff. Neither the calculus route nor the $\mathsf{CauchyProductMetric}^{\uparrow}$ route contains a host product metric, quotient stream equality, selected product limit, detached product Real, or ambient completeness witness. Therefore every accepted paired-distance read keeps the finite modulus route visible before it reads the component $\mathsf{Real}^{\uparrow}$ seals. □

Theorem 2668.76 (Calculus product metric modulus handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer that reads a derivative-product, endpoint, or paired-distance conclusion must read the product-metric modulus handoff through*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

before either component $\mathsf{Real}^{\uparrow}$ *seal is exposed. The handoff supplies only the displayed product-distance packet, paired source rows, metric row, finite stream windows, regular rational readbacks, dyadic tolerance ledger, and terminal component seals. It supplies no host product metric, quotient stream equality, detached Real pair, selected product limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricModulusHandoff`

Proof. Start with Definition 3632.73. It places every derivative-product, endpoint, or paired-distance consumer after the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet and before any component Real-facing value can be inspected.

Refine the route by Definition 3632.75. That theorem identifies the finite modulus rows: the $\mathsf{CauchyProduct}^{\uparrow}$ sources, $\mathsf{Met}^{\uparrow}$ distance row, paired $\mathsf{StreamName}^{\uparrow}$ windows, paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Finally use the $\mathsf{CauchyProductMetric}^{\uparrow}$ real-seal boundary of Definition 3551.4. The boundary keeps the component seals downstream of the product-metric route, so a calculus consumer cannot detach the two Real rows and later pretend that their product distance came from a host product metric.

The excluded principles have no row in the composed display. Quotient stream equality, detached Real pairs, selected product limits, host product metrics, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not generated by the calculus packet or by the product-metric sibling, so the modulus handoff is the only accepted route to the paired-distance read. □

Theorem 2668.77 (Calculus product metric regulated-integral handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *paired-distance or derivative-product consumer that feeds a* $\mathsf{RegulatedIntegral}^{\uparrow}$ *endpoint must first expose the shared route*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and then the $\mathsf{RegulatedIntegral}^{\uparrow}$ *finite ledger*

$$I,\ G,\ A,\ S,\ D,\ R,\ E,\ H,\ C,\ P,\ N$$

before any integral endpoint is public. The route supplies product-metric Real endpoints and the regulated finite-sum handoff through the same L10 Real-completion path. It supplies no host product metric, selected product limit, measure completion, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRegulatedIntegralHandoff`

Proof. Start with Definition 3632.76. It forces any derivative-product, endpoint, or paired-distance conclusion to expose the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet, the $\mathsf{CauchyProduct}^{\uparrow}$ source rows, the $\mathsf{Met}^{\uparrow}$ distance row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Next feed the terminal Real-facing data into the regulated-integral route of Definition 1557.7. That route keeps the integral endpoint behind the same L10 Real-completion interface rather than behind a host integral or a detached completed Real object.

Finally apply the public interface theorem Definition 1557.10. The public regulated-integral endpoint is readable only through the finite ledger $I, G, A, S, D, R, E, H, C, P, N$, so the calculus product-metric route and the regulated-integral finite-sum handoff meet at the displayed Real-completion boundary.

The excluded objects have no coordinate in the composed route. Host product metrics, selected product limits, measure completion, quotient function equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are therefore not available to the consumer before the regulated endpoint is exposed. □

Theorem 2668.78 (Calculus product metric regulated-endpoint route)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer exporting regulated integral endpoints from paired-distance or derivative-product data must factor through* $\mathsf{CauchyProductMetric}^{\uparrow}$*,* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and then the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *endpoint ledger. The route first exposes the shared product-metric rows*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and then reads the regulated endpoint ledger

$$I,\ G,\ A,\ S,\ D,\ R,\ E,\ H,\ C,\ P,\ N.$$

It supplies no host product metric, measure completion, quotient function equality, selected product limit, detached endpoint pair, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the product metric handoff of Definition 3632.76. It places every derivative-product, endpoint, or paired-distance conclusion behind the CauchyProductMetric$^{\uparrow}$ packet, the CauchyProduct$^{\uparrow}$ source rows, the Met$^{\uparrow}$ distance row, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, the DyadicRatCore$^{\uparrow}$ tolerance ledger, and the terminal Real$^{\uparrow}$ seals.

Refine the route by the finite modulus theorem Definition 3632.75. That theorem keeps the paired-distance data attached to the product-metric packet before either component Real seal can be read, so the endpoint source is not a detached product limit.

Now apply the regulated-integral handoff of Definition 3632.77. It feeds the terminal product-metric Real data into the regulated endpoint ledger $I, G, A, S, D, R, E, H, C, P, N$ rather than into a host integral or a measure-completion object.

Finally use the public interface theorem Definition 1557.10. The public endpoint is readable only through the displayed regulated-integral rows, and the preceding product-metric rows supply its Real-facing inputs. Since neither route contains a host product metric, measure completion, quotient function equality, selected product limit, detached endpoint pair, or ambient completeness principle, every exported endpoint factors through the finite route stated above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRegulatedEndpointRoute`

Theorem 2668.79 (Calculus product metric Real route)**.** *Any* Calculus$^{\uparrow}$ *paired-distance consumer that exports a* Real$^{\uparrow}$*-facing derivative or endpoint read factors through the same finite route: the* CauchyProductMetric$^{\uparrow}$ *packet, two* CauchyProduct$^{\uparrow}$ *rows, paired* StreamName$^{\uparrow}$ *windows, paired* RegSeqRat$^{\uparrow}$ *readbacks, the* DyadicRatCore$^{\uparrow}$ *product-distance ledger, the* Met$^{\uparrow}$ *distance row, the modulus handoff row, and then the terminal* Real$^{\uparrow}$ *seals. The route exports no host product metric, quotient stream equality, detached Real pair, selected product limit, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRealRoute`

Proof. Start with the forward link of Definition 3632.73. It forces every derivative-product, endpoint, or paired-distance consumer that uses product-distance data to display the CauchyProductMetric$^{\uparrow}$ packet before reading component Real values.

Use the seal pairing theorem of Definition 3632.74. It keeps the two terminal Real$^{\uparrow}$ seals componentwise but attached to one shared product-metric packet rather than to a detached Real pair.

Refine the shared packet by Definition 3632.75 and Definition 3632.76. These rows expose the two CauchyProduct$^{\uparrow}$ sources, paired StreamName$^{\uparrow}$ windows, paired RegSeqRat$^{\uparrow}$ readbacks, the DyadicRatCore$^{\uparrow}$ product-distance ledger, the Met$^{\uparrow}$ distance row, and the handoff before the terminal seals are read.

Since each cited route excludes host product metrics, quotient stream equality, selected product limits, detached Real pairs, and ambient completeness, their composition gives exactly the finite Real-facing route stated above. □

Theorem 2668.80 (Calculus product metric forward-link export)**.** *Any public* Calculus$^{\uparrow}$ *paired-distance or derivative-product export factors through the product-metric route before it may feed regulated-integral or endpoint consumers. The export must display the existing* CauchyProductMetric$^{\uparrow}$ *route,* CauchyProduct$^{\uparrow}$ *rows, the* Met$^{\uparrow}$ *distance row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and terminal* Real$^{\uparrow}$ *seals as one finite Real-completion path.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardLinkExport`

Proof. Start with Definition 3632.73. It places every paired-distance or derivative-product consumer behind the product-metric route before component Real-facing data can be read.

Use Definition 3632.74, Definition 3632.75, and Definition 3632.76. These rows keep the paired seals attached to one $\mathsf{CauchyProductMetric}^{\uparrow}$ packet, expose the product rows, metric row, stream windows, regular readbacks, and dyadic tolerance ledger, and prevent a detached Real pair or host product metric from entering the export.

Finally apply Definition 3632.79. The Real route is already terminal for the component seals, so the public export can feed regulated-integral or endpoint consumers only through that same displayed Real-completion path. □

Theorem 2668.81 (Calculus Peano dependency route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *read that is used as an ODE-facing derivative row or local-flow row must consume the Peano and initial-value surfaces after the Real and continuous-map rows have already been displayed. Concretely, such a read is routed through the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus rows, chapter 3530, and chapter 3979 before it may be exported as an ODE existence consumer.*

Lean4 ✓ `BEDC.Derived.CalculusUp.PeanoDependencyRoute.CalculusPeanoDependencyRoute`

Proof. The hub spine fixes $\mathsf{Calculus}^{\uparrow}$ as a packet over Real-facing limits, local operations, and continuous-map readback. The existing Real handoff and finite-difference routes make any derivative read visible only after the terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows have been named.

An ODE-facing consumer needs more than that derivative row. It must also expose the local existence packet of chapter 3530 and the initial-data packet of chapter 3979, since those chapters bind the interval source, vector-field row, local window, and solution-read surface used by the existence request.

Thus the calculus derivative or local-flow read is not a detached host derivative. It is an additive child route from the already displayed Real and continuous-map rows through Peano existence and initial-value data before any ODE existence consumer is public. □

The $\mathsf{Calculus}^{\uparrow}$ real-completion route is read as a child of the existing limit handoff. The endpoint surface of Definition 3632.70 and Definition 3632.72 already fixes the finite readback order through $\mathsf{Limit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any calculus consumer can export a Real-facing value.

Theorem 2668.82 (Calculus Real-completion forward link)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *Real-facing endpoint export that consumes a* $\mathsf{Limit}^{\uparrow}$ *row must pass through the* $\mathsf{Real}^{\uparrow}$ *terminal seal, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row displayed by Definition 3632.70 and Definition 3632.72. The forward link is the finite L10 route from calculus limits into the Real-completion spine of chapter 38; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionForwardLink`

Proof. Read the endpoint export through Definition 3632.70. That theorem already places every derivative, integral, or endpoint consumer behind the $\mathsf{Limit}^{\uparrow}$ surface together with the $\mathsf{Real}^{\uparrow}$ terminal seal, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{StreamName}^{\uparrow}$ schedule rows.

Now apply Definition 3632.72. It inserts the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row before the regular readback is sealed as a Real-facing value, so the downstream calculus export cannot skip the displayed real-completion readback order.

Since the route has no coordinate for a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the child link records only the finite L10 dependency lattice already visible in the sibling handoff theorems. □

Theorem 2668.83 (Calculus real-completion sibling route)**.** *Every accepted* Calculus$^{\uparrow}$ *derivative, integral, or limit consumer that reaches Real-facing completion data must route through the* Real$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *completion stack, with the* StreamName$^{\uparrow}$ *naming surface available when the consumer reads a displayed sequence. The route is witnessed by chapter 38, chapter 301, and chapter 126. It is not an appeal to ambient Real completeness, host quotient completion, or selected limits outside the accepted completion stack.*

Proof. Read the accepted Calculus$^{\uparrow}$ consumer through its derivative, integral, or limit interface. A Real-facing completion request first has to expose the Real$^{\uparrow}$ certificate surface rather than a host completeness object.

The completion data then factors through the regular rational naming route supplied by RegSeqRat$^{\uparrow}$. When the consumer carries a sequential name, the visible stream component is the StreamName$^{\uparrow}$ naming surface, so the read remains inside the displayed completion stack.

Thus the horizontal dependency is exactly the sibling route through Real, regular rational sequences, and StreamName names. Since no carrier row in the local route names ambient Real completeness, host quotient completion, or selected limits, those endpoints are not exported by the calculus consumer. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionSiblingRoute`

Theorem 2668.84 (Calculus regulated-integral forward link)**.** *Every* Calculus$^{\uparrow}$ *integration-facing consumer that reads a regulated integral first exposes the regulated-integral finite step route of chapter 1557 before it may use* Real$^{\uparrow}$ *endpoint seals. In the root package of Definition 3632.5, the integral branch reads the calculus finite-sum row, then the* RegulatedIntegral$^{\uparrow}$ *interval, regulated-function, step approximation, finite step-sum, Darboux/Riemann compatibility, error, transport, replay, provenance, and local NameCert rows, and only then consumes the Real-facing endpoint.*

The forward link is therefore a sibling dependency from the accepted Calculus$^{\uparrow}$ *root package to the* RegulatedIntegral$^{\uparrow}$ *route. It does not supply a host integral functional, selected antiderivative, quotient equality of functions, measure completion, or ambient* Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegulatedIntegralForwardLink`

Proof. Start from the Calculus$^{\uparrow}$ root public package of Definition 3632.5. Its integration-facing branch contains the finite-sum row and the structural rows that transport, replay, source, and locally name the accepted calculus read.

Route that branch through chapter 1557. The regulated-integral packet exposes its interval, regulated-function, dyadic step approximation, finite step-sum, Darboux/Riemann compatibility, error, transport, replay, provenance, and local NameCert rows before a Real-facing handoff is public.

The terminal endpoint seal is then read through Real$^{\uparrow}$ and the existing calculus Real-completion sibling route. Since neither the calculus root package nor the regulated-integral sibling carries a coordinate for host integration, selected antiderivatives, quotient function equality, measure completion, or ambient completeness, the forward link remains the displayed finite route. □

Theorem 2668.85 (Calculus real and continuous-map dependency route)**.** *Every* Calculus$^{\uparrow}$ *derivative, integral, or limit-facing consumer that reads a scalar endpoint or function-value row must*

expose the Real$^{\uparrow}$ *seal of chapter 38 and the* ContinuousMap$^{\uparrow}$ *modulus-and-evaluation surface of chapter 130 before the calculus packet can replay the read through its local* NameCert *row. Thus the calculus hub records a substantive L10-to-analysis edge through Real names and continuous-map data rather than a detached kernel-only route.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealContinuousDependencyRoute`

Theorem 2668.86 (Calculus Real-completion orphan lattice route)**.** *Every* Calculus$^{\uparrow}$ *regulated-integral or limit-facing consumer that reads an endpoint seal must first pass through the L10 Real-completion source lattice of chapter 38, chapter 301, and chapter 126. In particular, the consumer may use the* RegulatedIntegral$^{\uparrow}$ *handoff of Definition 3632.84 only after the source stream window, regular rational readback, and terminal Real seal are visible as the displayed completion route. It exports no host completion theorem, selected representative of a Cauchy stream, quotient real equality, detached endpoint seal, or ambient integration functional.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionOrphanLatticeRoute`

Proof. Start with the Calculus$^{\uparrow}$ root package. Its regulated-integral and limit-facing rows are accepted only as finite BEDC reads through the calculus carrier, transport, replay, provenance, and local naming rows.

Use the regulated-integral forward link. It routes endpoint reads through the displayed RegulatedIntegral$^{\uparrow}$ ledger rather than through a host integral or detached endpoint object.

Finally compose this handoff with the Real-completion source lattice. The StreamName$^{\uparrow}$ window supplies the finite source, RegSeqRat$^{\uparrow}$ supplies regular rational readback, and Real$^{\uparrow}$ supplies the terminal seal, so every such endpoint read remains inside the L10 completion lattice. □

Theorem 2668.87 (Calculus real-completion CauchySpace dependency link)**.** *Every accepted* Calculus$^{\uparrow}$ *derivative, integral, or limit consumer that reaches Real-facing completion data must pass through the* CauchySpace$^{\uparrow}$ *filter-uniform and regular-sequence boundary before the terminal* Real$^{\uparrow}$ *seal is read. Concretely, the existing route through Definition 3632.83 factors through the Cauchy-space consumer surface of chapter 1847: the* CauchyFilter$^{\uparrow}$ *and* UniformSpace$^{\uparrow}$ *rows expose the completion request, the* RegSeqRat$^{\uparrow}$ *row supplies the regular rational readback, the* StreamName$^{\uparrow}$ *row supplies any displayed sequential name, and only then may the* Real$^{\uparrow}$ *seal be consumed.*

Proof. Start with a Real-facing Calculus$^{\uparrow}$ consumer and read it through Definition 3632.83. That route already forces the endpoint through the Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and StreamName$^{\uparrow}$ completion stack rather than through ambient completeness.

Now insert the Cauchy-space boundary supplied by chapter 1847. Its Real-consumer rows require the displayed filter, uniform, regular-name, precompletion, transport, replay, provenance, and local naming data before a Real-facing consumer can inspect completion data.

Thus the calculus endpoint cannot jump directly from a local derivative, integral, or limit row to a Real seal. It first traverses the Cauchy-space filter-uniform and regular-sequence interface, and the route exports no host quotient completion, selected limit, countable-choice selector, or ambient Real$^{\uparrow}$ completeness theorem. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionDependencyLink`

Theorem 2668.88 (Calculus pointwise stream metric dependency route)**.** *Every accepted* Calculus$^{\uparrow}$ *derivative, integral, or local-error consumer that compares real-valued stream data must first read the pointwise metric-stream packet of chapter 4139. The comparison route consumes the displayed*

StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *seal, and metric row before derivative or integral estimates may treat the local error as a real-valued bound. Interval-local derivative or integral consumers also read the compact interval endpoint and finite-net route of chapter 5553 before treating a local error bound as a calculus estimate on a compact real interval.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusPointwiseStreamMetricDependencyRoute`

Proof. Project the calculus consumer through its derivative, integral, or local limit row. A local error comparison needs a pointwise distance before it can be read as a Real-facing estimate.

Use chapter 4139. Its carrier displays the stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, terminal Real seal, and metric row before any downstream completion or calculus consumer receives the distance packet.

Use chapter 5553 for interval-local estimates. The CompactRealInterval$^{\uparrow}$ packet supplies endpoint Real seals and finite interval-net rows, so the pointwise metric stream comparison is anchored to a displayed compact interval packet rather than to an ambient host interval.

Thus the calculus route reads pointwise stream metric control before derivative and integral consumers compare real-valued local errors. The dependency exports no host function equality, quotient stream equality, selected representative, countable-choice schedule, or ambient Real$^{\uparrow}$ completeness principle. □

Theorem 2668.89 (Calculus L10 Real and Completion dependency)**.** *Every* Calculus$^{\uparrow}$ *certificate that exports derivative, integral, or limit-facing clauses to the Real and Completion route must first read the L10 substrate through chapter 38, chapter 301, chapter 126, and chapter 130. The calculus row therefore consumes terminal real seals, regular rational readbacks, finite stream windows, and displayed continuous-map evidence before derivative or integral clauses may be used as public BEDC obligations.*

Proof. Read the finite observation schedule through the StreamName certificate. This fixes the inspected windows before any regular rational or real-facing calculus row is exported.

Use the RegSeqRat and Real certificates next. They attach regular rational readback and terminal real sealing to the same windows.

Finally read the ContinuousMap certificate for displayed function evidence. With those four sibling certificates present, the derivative, integral, and limit-facing calculus clauses have a visible L10 dependency route rather than a kernel-only carrier island. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusL10RealCompletionDependency`

Theorem 2668.90 (Calculus L10 dependency route)**.** *Every accepted* Calculus$^{\uparrow}$ *consumer reads derivative and integral data only after the displayed* Real$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* Limit$^{\uparrow}$*, and* ContinuousMap$^{\uparrow}$ *style rows have been exposed. The route is therefore horizontal through chapter 38, chapter 301, chapter 126, chapter 1678, and chapter 130; it is not a host differentiability oracle, a selected antiderivative, an ambient completeness principle, or a quotient of function germs.*

Proof. Project an accepted Calculus$^{\uparrow}$ packet through the existing dependency routes in this hub. The Real-facing row supplies the scalar seal, the regular rational and stream-name rows supply finite approximation schedules, the limit row supplies the local convergence boundary, and the continuous-map row supplies the function readback surface.

Derivative and integral reads occur only after those rows are displayed. The packet may replay them through its structural BHist, hsame, Cont, Pkg, and NameCert rows, but those rows preserve the displayed dependencies rather than adding a host-level analytic oracle.

Thus an L10 consumer sees calculus data as a finite dependency route through the sibling chapters named above. No row exports selected antiderivatives, ambient completeness, quotient germs, or an untyped host computation. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusL10DependencyRoute`

The $\mathsf{Calculus}^{\uparrow}$ limit and endpoint consumers that reuse a Real-completion diagonal estimate read the shared-threshold packet of chapter 909 before exporting a Real-facing calculus row. That sibling packet exposes the route through the shared modulus, regular-Cauchy tail meet, witness envelope, diagonal-window coverage, and comparison row; calculus therefore receives a displayed finite diagonal-meet boundary rather than a detached host limit, quotient stream equality, or selected completion witness. The $\mathsf{Calculus}^{\uparrow}$ hub consumes its Real-facing dependency route through the existing BEDC name certificates for chapter 38, chapter 301, chapter 126, and chapter 291. Its paired-distance consumers enter the product-metric child route of Definition 3632.73 before exposing any derivative-product, endpoint, or component $\mathsf{Real}^{\uparrow}$ seal, so the hub records a horizontal dependency on those finite route packets rather than a bare list of local children.

The $\mathsf{Calculus}^{\uparrow}$ endpoint and derivative-product reads also pass through the finite Cauchy real completion interface of chapter 1844 before any completion-facing consumer may inspect a terminal $\mathsf{Real}^{\uparrow}$ seal.

Lemma 2668.91 (Calculus lower-envelope Real-facing route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or variation consumer that reads a lower-envelope regularization as a Real-facing value must route first through chapter 1536, then through chapter 6224, then through the regular rational readback of chapter 301, and finally through the terminal seal of chapter 38. Thus the calculus packet sees the lower-envelope value only after the finite lower-semicontinuity, epigraph, regular-rational, and Real rows are displayed, with no route through host suprema, quotient functions, selected minimizers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLowerEnvelopeRealFacingRoute`

Proof. Read the lower-envelope request as a calculus-facing value. The lower-semicontinuity row supplies the admissible lower-envelope boundary, and the epigraph row supplies the displayed value/lower-bound comparison used by that boundary.

Next pass to regular rational data. The $\mathsf{RegSeqRat}^{\uparrow}$ route records the finite readback through visible windows, so the calculus consumer has a concrete value surface before it can inspect any Real seal.

Finally read the $\mathsf{Real}^{\uparrow}$ seal. Since the route has already passed through the displayed lower-semicontinuity, epigraph, and regular-rational rows, no derivative or integral consumer can replace it by a host supremum, quotient-function equality, selected minimizer, or ambient completeness principle. □

Theorem 2668.92 (Calculus lower-envelope regularization handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or variation consumer that imports lower-envelope regularization must preserve the finite route*

$$\mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{Epigraph}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before any calculus operation reads the regularized value. The consumer first displays the lower-semicontinuity obligation and epigraph comparison rows, then reads the regular rational approximation together with the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no host supremum, quotient function, selected minimizer, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLowerEnvelopeRegularizationHandoff`

Proof. Begin with the route lemma above. It already forces a calculus-facing lower-envelope value through $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the value is visible to a derivative, integral, or variation consumer.

Next use the lower-envelope regularization packet of chapter 7792. Its NameCert obligations keep the finite lower-envelope ledger between the epigraph row and the regular-rational readback, while its L10 handoff keeps the regularized value inside the displayed envelope route.

Finally insert the shared dyadic tolerance discipline. The regularization carrier records the finite comparison ledger read by the regular-rational row, and downstream calculus consumers must retain that $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance surface before they read the $\mathsf{Real}^{\uparrow}$ seal.

The resulting handoff is the listed finite route. Since the cited rows contain no coordinate for host suprema, quotient functions, selected minimizers, or ambient completeness, no calculus operation can recover those objects from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Calculus}^{\uparrow}$ packet is generated from Real, Limit, ContinuousMap, Derivative, Integral, RegSeqRat, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge-facing export surface is the finite source-window calculus route over $\mathsf{Real}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CalculusUp.CalculusLimitRealTerminalExport`

Bridge status: none

Not claimed: This chapter does not close bridge checking, a host function space, global differentiability, an ambient integration operator, quotient function equality, or real completeness beyond the displayed Real row.

Upgrade path: Further work is standard-bridge verification over the checked limit-real terminal export without adding host function spaces, global differentiability, ambient integration, quotient function equality, or extra real completeness.

Chapter 2669

A Concrete Naming Certificate for $\mathsf{FrechetDerivative}^{\uparrow}$

Following Cartan-style differential calculus on normed spaces, the $\mathsf{FrechetDerivative}^{\uparrow}$ packet records the local linear approximation surface as finite NameCert data. It sits beside the norm carrier of chapter 89, the linear-map obligations of chapter 48, the calculus route named by $\mathsf{Calculus}^{\uparrow}$, and the terminal real-name seal of chapter 38. The packet names a displayed derivative read through norm, linearity, and dyadic remainder control; it is not a host function-space derivative, not a selected tangent map outside the carrier, and not an ambient theorem about differentiability.

Definition 2669.1 (Frechet derivative carrier)**.** *A* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the normed source row, carrying the domain and codomain norm data through which difference vectors and error sizes are read.*
2. *A is the point-and-window row, recording the base point, admissible neighbourhood window, and finite comparison requests visible to the derivative packet.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the linear approximation admitted at the displayed point and over the same source and target norm rows.*
4. *R is the dyadic remainder-modulus row. It records a finite tolerance schedule that bounds the residual after the linear read L has been subtracted from the local step read.*
5. *H records componentwise* hsame *transport for X, A, L, R and the structural rows.*
6. *C records displayed* Cont *replay from a derivative request through the normed source, point window, linear map, and dyadic remainder-modulus rows.*
7. *P is the* Pkg *provenance over* $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, a global function-space topology, an unrecorded derivative operator, quotient tangent germs, or classical differentiability primitives outside the displayed finite route.

Lean4 ✓ `BEDC.Derived.FrechetDerivativeUp.TasteGate.FrechetDerivativeTasteGate single carrier alignment`

Theorem 2669.2 (Frechet derivative NameCert obligations)**.** *For every accepted* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier* $D = (X, A, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the normed source row, point-and-window row, linear-map row, dyadic remainder-modulus row, componentwise transport, displayed replay, provenance, and local naming; linear approximation of the local step through* L*; remainder-modulus control through* R *over the same window* A *and norm row* X*; classifier stability under* hsame *transport of all displayed rows; and ledger non-escape from host function equality, global function-space topology, unrecorded derivative operators, quotient tangent germs, and classical differentiability primitives outside the packet.*

Proof. Project the carrier of Definition 3634.1. The tuple X, A, L, R, H, C, P, N is the whole accepted packet, so carrier admission for the named rows is a direct read from the finite history.

The linear approximation clause is the read of L over the source row X and the point window A. Because L is a $\mathsf{LinMap}^{\uparrow}$ row in the same packet, the approximation is not detached into a host tangent map.

The remainder-modulus clause is the read of R. Its dyadic tolerance schedule is attached to the same X and A, so the residual bound is compared through the displayed norm and window rows rather than through an external limit predicate.

Classifier stability is the componentwise transport row H, replay is the continuation row C, and source accounting is the package row P. Since no coordinate names any excluded host object, the local certificate exports only the displayed Frechet-derivative route. □

Theorem 2669.3 (Frechet derivative linear-remainder handoff)**.** *Every derivative read of an accepted* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier factors through the ordered finite route*

$$\mathsf{Norm}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{Calculus}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The linear approximation row and dyadic remainder-modulus row are exposed before any consumer may read the derivative as public BEDC data. Thus the packet supplies a traditional analysis neighbour for later derivative and implicit-function certificates without importing a host derivative operator.

Proof. Read a derivative request through the continuation row C. The replay first exposes the normed source row X and point window A, so the request is already tied to displayed source, target, and local neighbourhood data.

Next read the $\mathsf{LinMap}^{\uparrow}$ row L and the dyadic remainder-modulus row R. The first row supplies the linear approximation, while the second supplies the finite tolerance ledger that keeps the residual under the displayed norm comparison.

The package row P records the dependency route through $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The local naming row N exports only that route, with no host function-space derivative, selected tangent germ, quotient equality, or ambient differentiability theorem. □

Falsifiable prediction: Every accepted $\mathsf{FrechetDerivative}^{\uparrow}$ packet must expose $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before any derivative read is public.

Independence witness: norm, linearmap, calculus, real: $\mathsf{FrechetDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a normed source row, point-and-window row, linear approximation row, dyadic remainder-modulus row, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3634.1, Definition 3634.2, and Definition 3634.3 seed the finite Frechet derivative route over Norm, LinearMap, Calculus, Real, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, global function-space topology, quotient tangent germs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, linear approximation, remainder-modulus control, classifier stability, derivative handoff stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.FrechetDerivative

Constructive story: A $\mathsf{FrechetDerivative}^{\uparrow}$ packet is generated from a normed source row, a point-and-window row, a LinearMap approximation row, a dyadic remainder-modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2670

A Concrete Naming Certificate for $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Riemann–Darboux treatment of bounded intervals, the $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ packet records a finite subdivision surface for a displayed bounded $\mathsf{Real}^{\uparrow}$ interval. It sits after the dyadic rational core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the located partition surface of chapter 2248. The packet supplies a reusable finite subdivision carrier for $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$ consumers; it does not supply open-cover compactness, selected limits, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2670.1 (Bounded real interval partition carrier)**.** *A* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, M, T, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the bounded interval endpoint row, carrying lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seals together with the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used to observe them.*
2. *S is the ordered subdivision row, listing finitely many subdivision nodes between the displayed endpoints and the adjacent cell ledger they generate.*
3. *M is the dyadic mesh-bound row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording a rational precision budget, finite cell-width bounds, and the endpoint-to-cell comparison rows.*
4. *T is the tag-window row, assigning each cell a finite tag window and recording only the displayed containment evidence for that tag.*
5. *Q is the regular rational readback row for the subdivision nodes, tag windows, mesh bounds, and endpoint comparisons.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and subdivision observations, reached only after E, S, M, T, Q have been displayed.*
7. *L is the integration-consumer ledger, recording that* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* $\mathsf{RiemannStieltjes}^{\uparrow}$*, and* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumers may read the finite partition only through the rows E, S, M, T, Q, R.*

8. *H records componentwise* hsame *transport for endpoints, subdivision nodes, dyadic mesh bounds, tag windows, regular readbacks, real seals, consumer ledgers, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from endpoint observations to ordered cells, from cells to tag windows, and from mesh bounds to the consumer ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed endpoint, subdivision, mesh-bound, tag-window, readback, seal, consumer-ledger, transport, replay, provenance, and local naming rows. It has no coordinate for an open-cover compactness proof, selected limiting partition, quotient real equality, host interval partition, countable choice, or ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.BoundedRealIntervalPartitionUp.TasteGate.BoundedRealIntervalPartitionTasteGate single carrier alignment`

Theorem 2670.2 (Bounded real interval partition NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier* $B = (E, S, M, T, Q, R, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartitio}}$ *surface consists exactly of carrier admission; bounded endpoint exposure through E; ordered subdivision exposure through S; dyadic mesh-bound exposure through M; tag-window containment through T;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through R; integration-consumer routing through L; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Thus a Riemann, Darboux, Riemann–Stieltjes, or regulated-integral consumer receives a finite subdivision surface only after the endpoint, cell, mesh-bound, tag-window, readback, and seal rows have all been displayed.

Proof. Project the carrier of Definition 3635.1. The tuple lists the only available rows: endpoints, ordered subdivision nodes, dyadic mesh bounds, tag windows, regular readbacks, real seals, consumer routing, transport, replay, provenance, and local naming.

The endpoint stage is E. Its $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{Real}^{\uparrow}$ seals supply bounded endpoint observations before a cell can be inspected.

The finite subdivision stage is S, M, T, Q. The ordered row S supplies the cells, the dyadic row M records the mesh budget and cell-width bounds, the tag row T confines each tag to a displayed window, and the regular readback row Q reads those finite observations.

The consumer stage is R, L, H, C, P, N. The terminal seal R is reached only through the displayed rows, while L records the permitted $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$ reads. The structural rows transport, replay, package, and name exactly this route, so the listed obligations exhaust the local NameCert surface. □

Theorem 2670.3 (Bounded real interval partition mesh consumer readiness)**.** *Let* $B = (E, S, M, T, Q, R, L, H, C, P$ *be an accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier. The ordered subdivision row S and dyadic mesh row M are read as a bounded refinement of the located partition surface of chapter 2248 before any* $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer may inspect the integration-consumer ledger L. The consumer therefore receives a located finite partition with bounded dyadic mesh evidence, not an isolated mesh code or host interval partition.*

Proof. Project the carrier of Definition 3635.1. The endpoint row E supplies the bounded Real interval, and the subdivision row S lists ordered cells over that same endpoint surface.

Apply Definition 3635.2. The dyadic mesh-bound row M is not a separate object; it is read after S and compares adjacent cells by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ budget.

The located partition dependency is therefore substantive. The rows S and M refine the located interval partition of chapter 2248 by adding bounded endpoint access and dyadic mesh bounds before the regular readback Q and real seal R are exposed.

Only then may the ledger L route the packet to Riemann, Darboux, Riemann–Stieltjes, or regulated-integral consumers. Since the carrier contains no host partition coordinate, every such consumer reads the bounded refinement route rather than a free-standing subdivision code. □

Theorem 2670.4 (Bounded real interval partition transport stability)**.** *Let $B = (E, S, M, T, Q, R, L, H, C, P, N)$ and $B' = (E', S', M', T', Q', R', L', H', C', P', N')$ be accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carriers classified together by componentwise* hsame *transport. If S' is read as the transported ordered subdivision over the transported bounded endpoint row E', then the mesh-bound row, tag-window row, regular readback row, real-seal row, and integration-consumer ledger are all read through the same finite transported subdivision route.*

The transported carrier therefore preserves the finite partition surface without adding open-cover compactness, selected limits, quotient real equality, a host interval partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint row. Componentwise transport carries E to E', including the finite StreamName windows, RegSeqRat readbacks, and Real seals attached to the lower and upper endpoints.

Read the transported subdivision through S'. Since its cells are displayed over E', every later mesh or tag read remains attached to the same bounded interval route.

The rows M', T', Q', R' are downstream of that route. The dyadic mesh bounds compare transported adjacent cells, the tag windows remain cell-local, the regular readbacks observe only the displayed finite windows, and the Real seal is reached after those observations.

Finally L', H', C', P', N' carry the same route to the integration consumers and name it. No additional coordinate exists in the carrier, so the transported packet cannot export any of the excluded compactness, selection, quotient, host-partition, choice, or completeness data. □

Theorem 2670.5 (Bounded real interval partition consumer nonescape)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* $\mathsf{RiemannStieltjes}^{\uparrow}$*, or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BoundedRealIntervalPartition}^{\uparrow}.$$

The route exposes endpoint rows, ordered subdivision rows, dyadic mesh bounds, tag windows, readbacks, terminal seals, componentwise transport, displayed replay, provenance, and local naming. It exports no open-cover compactness theorem, selected limiting partition, quotient real equality, host interval partition, measure-completion object, or ambient real-completeness principle.

Proof. Apply Definition 3635.2. It makes $E, S, M, T, Q, R, L, H, C, P, N$ the entire public surface of the packet.

Apply Definition 3635.4. Any transported or refined consumer read must preserve the same endpoint-to-subdivision-to-mesh route before the consumer ledger L is available.

Now inspect the consumers named in L. A Riemann or Darboux read may use the displayed cells and mesh bounds; a Riemann–Stieltjes read may additionally pair those cells with its own

integrator row; and a regulated-integral read may use the same finite subdivision as a step surface. In each case the consumer receives only the listed finite rows and their structural provenance.

Since the carrier contains no row for compactness, selected limits, quotient equality, host partitions, measure completion, or ambient real completeness, none of those objects can escape through the consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ packet is generated from bounded Real endpoint seals, ordered subdivision rows, dyadic mesh bounds, tag windows, RegSeqRat readbacks, integration-consumer routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3635.1, Definition 3635.2, Definition 3635.4, and Definition 3635.5 seed the bounded real interval partition boundary over $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient real equality, host interval partitions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, ordered subdivision exactness, dyadic mesh-bound stability, tag-window containment, RegSeqRat readback, Real seal nonescape, integration-consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRealIntervalPartitionUp.

Chapter 2671

A Concrete Naming Certificate for $\mathsf{CauchySequenceOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceOrder}^{\uparrow}$ packet records the finite constructive-order route by which two regular rational Cauchy source rows are compared before any $\mathsf{Real}^{\uparrow}$ consumer reads an ordered boundary. It sits over the binary algebra packet of chapter 3708, the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic comparison ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives completion and ordered-Real consumers one finite comparison surface for Cauchy sequence algebra; it does not import trichotomy, quotient Cauchy classes, selected limits, countable choice, host real order, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2671.1 (Cauchy sequence order carrier)**.** *A* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations supply the left and right comparison inputs.*
2. *A is the upstream* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row that fixes the shared binary Cauchy-algebra source from which the comparison is read.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite pairs of observations where X and Y are inspected at the same requested tolerance scale.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row. It stores the displayed dyadic radius, rational endpoint differences, slack conversions, strict-gap tests, and non-strict comparison checks used by the accepted order request.*
5. *L is the apartness row. It records only displayed positive-gap evidence between the two regular names, with the gap budget read through D and the windows read through W.*
6. *Q is the non-strict order row. It records only displayed tolerance domination evidence, again through the same W and D rows.*
7. *R is the comparison replay row, exposing whether a downstream consumer requested apartness, non-strict order, or the boundary case where no displayed comparison row is accepted.*

8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the requested comparison row has displayed its regular-source and dyadic-budget dependencies.*

9. *H records componentwise* hsame *transport for the two source rows, algebra row, synchronized windows, dyadic budget, apartness row, non-strict order row, replay row, Real seal, continuation route, provenance, and local naming row.*

10. *C records displayed* Cont *replay from a constructive order request through $X, Y, A, W, D, L, Q, R, S, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceOrder}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, A, W, D, L, Q, R, S, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient-stream equality, a selected limit, host real order, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceOrderUp.CauchySequenceOrderTasteGate single carrier alignment`

Theorem 2671.2 (Cauchy sequence order NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier*

$$O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceOrder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, upstream* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budget, apartness row, non-strict order row, comparison replay row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local naming;*

2. *source-window synchronization, requiring every comparison request to inspect X and Y through the same displayed window row W before any dyadic comparison is read;*

3. *dyadic-budget exactness, requiring every apartness or non-strict order row to consume only the displayed radius, difference, slack, strict-gap, and domination entries of D;*

4. *comparison-row separation, requiring L to carry only positive-gap apartness evidence and Q to carry only non-strict domination evidence, while both rows consume the same source, algebra, window, and dyadic-budget rows;*

5. *Real seal handoff, requiring S to be exposed only after the requested comparison row has displayed its regular-source and dyadic-budget dependencies;*

6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, A, W, D, L, Q, R, S, P, N$ without adding trichotomy, quotient equality, selected limits, host real order, countable choice, or ambient completeness.*

No Cauchy sequence order NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3636.1. The displayed coordinates are exactly the two source rows X, Y, algebra row A, synchronized windows W, dyadic budget D, comparison rows L, Q, R, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the comparison budget is available, so the carrier has no second observation service for either input.

Dyadic-budget exactness and comparison-row separation are read from D, L, Q. The row D supplies the finite arithmetic comparisons, the row L consumes those comparisons for positive-gap apartness, and the row Q consumes them for non-strict domination. Since L and Q are distinct rows with the same upstream inputs, neither row can import a hidden order oracle.

Finally, the replay, seal, and structural rows close the finite route. The row R records which comparison request is being replayed, the seal S is downstream of the displayed comparison dependency, and H, C, P, N only transport, replay, package, and name the listed rows. The excluded trichotomy, quotient, selected-limit, host-order, choice, and ambient-completeness data have no carrier coordinate. □

Theorem 2671.3 (Cauchy sequence order dyadic budget exactness)**.** *Every apartness or non-strict order fact exported from an accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier reads its comparison budget through the displayed route*

$$X, Y, A \longmapsto W \longmapsto D \longmapsto L, Q, R \longmapsto S \longmapsto H, C, P, N.$$

Consequently the exported fact obtains synchronized regular-source observations, dyadic radius and slack data, strict-gap or domination evidence, the terminal Real seal handoff, transport, continuation, provenance, and local naming, and obtains no trichotomy datum, quotient Cauchy class, selected limit, host real order, countable-choice datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Begin with a downstream comparison request. By Definition 3636.1, the only source coordinates are X and Y, and the upstream algebra row is A.

The finite observation surface is then forced through W. The dyadic comparison budget is forced through D, whose rows record the displayed radius, endpoint differences, slack conversions, strict-gap tests, and domination checks.

Apartness reads the route row L, non-strict order reads the route row Q, and replay reads the route row R. Each of these rows is downstream of the same sources, algebra row, windows, and dyadic budget named in Definition 3636.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ consumer can read S only after the requested comparison row has displayed its dependencies, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which trichotomy, quotient equality, selected limits, host order, choice, or ambient completeness could be exported. □

Theorem 2671.4 (Cauchy sequence order non-escape)**.** *Let* $O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier. Any order-facing fact exported from* O *is a rowwise consequence of the displayed regular source rows, algebra row, synchronized windows, dyadic comparison budget, apartness row, non-strict order row, replay row, Real seal, transport, continuation, provenance, and local naming rows. It is not a trichotomy principle, quotient Cauchy class, selected limit, host real-order fact, countable-choice datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly $X, Y, A, W, D, L, Q, R, S$, and the structural coordinates are exactly H, C, P, N.

An exported apartness fact must use L after the source, algebra, window, and dyadic-budget rows have been displayed. An exported non-strict order fact must use Q after the same upstream rows have been displayed. A replayed comparison must pass through R, and any Real-facing boundary then passes through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create trichotomy, quotient equality, selected limits, host order, choice, or

ambient completeness, while provenance and local naming record only the finite packet. Thus every exported order-facing fact is confined to the listed rows, and each excluded principle has no entry point. □

Closure status for NameCert$_{CauchySequenceOrderUp}$.

Theory closure: TheoryGate(NameCert$_{CauchySequenceOrderUp}$, seed)

Scope closed: Definition 3636.1, Definition 3636.2, Definition 3636.3, and Definition 3636.4 seed the finite constructive-order route over CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{CauchySequenceOrderUp}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient Cauchy classes, selected limits, host real order, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, comparison-row separation, Real seal handoff, transport, continuation, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceOrderUp.

Constructive story: A CauchySequenceOrder$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, an upstream CauchySequenceAlgebra$^{\uparrow}$ row, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ comparison-budget ledger, apartness and non-strict order rows, a comparison replay row, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2672

A Concrete Naming Certificate for $\mathsf{NoetherianRing}^{\uparrow}$

Following Emmy Noether's ascending-chain formulation, the $\mathsf{NoetherianRing}^{\uparrow}$ packet records finite ideal-chain stabilization data over a commutative-ring source and an ideal source. In BEDC form it sits after $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44) and $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81), supplies a reusable finitary algebra row for the reduction surface of chapter 2197, and exposes the Noetherian component later consumed by chapter 172.

2672.1 Carrier definition

Definition 2672.1 (Noetherian ring carrier)**.** *A* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{NoethRing}(C, I, A, G, H, Q, P, N).$$

The row C is the $\mathsf{CommRing}^{\uparrow}$ *source surface. The row I is the* $\mathsf{Ideal}^{\uparrow}$ *source surface over the same classifier. The row A is the ascending-chain stabilization witness: every displayed chain of ideal rows over C and I carries a finite stage after which all later ideal endpoints classify with that stage. The row G is the finite-generator witness used by algebraic consumers that read Noetherianity through finitely generated ideal data. The row H is componentwise* hsame *transport for the commutative-ring, ideal, stabilization, and generator rows. The row Q is displayed* Cont *replay for the chain and generator certificates. The row P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ *naming row.*

Lean4 ✓ `BEDC.Derived.NoetherianRingUp.NoetherianRingTasteGate single carrier alignment`

Theorem 2672.2 (Noetherian ring NameCert obligations)**.** *For an accepted* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier* $\mathsf{NoethRing}(C, I, A, G, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ *surface consists exactly of carrier admission for the eight displayed rows, source compatibility with* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$*, finite stabilization access through A, finite-generator access through G, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoetherianRingUp.NoetherianRingCarrier namecert obligations`

Proof. Project the eight rows of Definition 3637.1. The source rows C and I fix the shared commutative-ring and ideal classifier before any chain is inspected.

The stabilization row A and generator row G are then read over that classifier. Componentwise transport H carries each displayed row through hsame, while Q replays the corresponding continuations without replacing the source surface.

Finally, P and N repack the same finite inventory as the local naming certificate. No quotient ideal space, maximal ideal selection, or Hilbert basis proof is introduced by this packet. □

Theorem 2672.3 (Noetherian ascending-chain stability)**.** *Let* $\mathsf{NoethRing}(C, I, A, G, H, Q, P, N)$ *be an accepted* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier. For every displayed ascending chain of ideal rows over the shared* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$ *classifier, the witness A supplies a finite stabilization stage, and H transports that stabilization along the same classifier. Thus any downstream consumer may use* $\mathsf{NoetherianRing}^{\uparrow}$ *as a finite ideal-chain certificate without re-proving stabilization inside its own packet.*

Proof. Begin with the source compatibility rows from Definition 3637.2. They force the chain to be read over the commutative-ring and ideal classifier already present in C and I.

Apply the stabilization witness A to obtain a finite stage for the displayed chain. If the accepted packet is transported through a classified replacement, use the componentwise transport row H to move both the chain entries and the stabilization witness through the shared classifier.

The transported stage remains a stage for the same finite chain surface because Q replays only the displayed continuation data and P prevents provenance escape. The conclusion is therefore a reusable finite ideal-chain certificate for later algebra chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$.

Constructive story: A $\mathsf{NoetherianRing}^{\uparrow}$ packet is generated from a $\mathsf{CommRing}^{\uparrow}$ source, an $\mathsf{Ideal}^{\uparrow}$ source over the same classifier, an ascending-chain stabilization witness, a finite-generator witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3637.1, Definition 3637.2, and Definition 3637.3 seed the finite ideal-chain stabilization interface over CommRingUp, IdealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal spaces, maximal ideal choice, Hilbert basis proof, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite stabilization access, finite-generator access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.NoetherianRingUp.

Chapter 2673

A Concrete Naming Certificate for $\mathsf{LeviCivitaConnection}^{\uparrow}$

Following Kobayashi and Nomizu, the $\mathsf{LeviCivitaConnection}^{\uparrow}$ packet records the finite NameCert surface that ties a Riemannian metric to the unique torsion-free metric-compatible connection visible to curvature consumers. It is grounded in the source-fibre metric packet of chapter 133, the connection packet of chapter 134, and the curvature boundary surface of chapter 135. The packet exports only displayed rows and certificate ledgers; it does not import an ambient smooth connection operator, coordinate Christoffel symbols, or a host-level uniqueness theorem.

Definition 2673.1 (Levi-Civita connection carrier)**.** *A* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, C, T, M, R, H, K, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row whose source-fibre surface supplies the metric data on the displayed tangent fibres.*
2. *C is the* $\mathsf{Connection}^{\uparrow}$ *row carrying the base, fibre, section, tangent-direction, derivative endpoint, and connection provenance visible to the packet.*
3. *T is the torsion-free boundary row, recording that the two displayed tangent-direction orders have no residual antisymmetric endpoint outside the connection ledger.*
4. *M is the metric-compatibility row, recording that the connection derivative preserves the displayed* $\mathsf{RiemannianMetric}^{\uparrow}$ *inner-product row along the inspected tangent window.*
5. *R is the* $\mathsf{Curvature}^{\uparrow}$ *compatibility row, exposing the curvature boundary obtained from the connection row after the torsion-free and metric-compatibility rows have been displayed.*
6. *H records componentwise* hsame *transport for G, C, T, M, R and their dependency rows.*
7. *K records the displayed* Cont *replay from the metric row and connection row through torsion-free verification, metric compatibility, and curvature compatibility.*
8. *P is the* Pkg *provenance over* $\mathsf{LeviCivitaConnection}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$ *row.*

The carrier therefore packages a displayed Riemannian metric together with a displayed connection whose torsion-free, metric-compatible, and curvature-facing rows are all visible in the same finite certificate. It does not contain a coordinate formula for Christoffel symbols, a selected smooth atlas, a global geodesic theorem, or a host-level equality between possible connections.

Lean4 ✓ `BEDC.Derived.LeviCivitaConnectionUp.LeviCivitaConnectionTasteGate single carrier alignment decode encode`

Theorem 2673.2 (Levi-Civita connection NameCert obligations)**.** *For an accepted* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier* $L = (G, C, T, M, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *carrier admission for the displayed metric and connection rows* G *and* C*;*
2. *torsion-free exposure for the boundary row* T*;*
3. *metric-compatibility exposure for the row* M*;*
4. *curvature-facing compatibility for the row* R*;*
5. *componentwise transport for* G, C, T, M, R *through* H*;*
6. *replay exactness for the route recorded by* K*;*
7. *provenance and local naming through* P *and* N*.*

No obligation asks for an unlogged covariant derivative, a coordinate chart calculation, a selected geodesic flow, or a host-level uniqueness principle.

Proof. Project the carrier of Definition 3638.1. The rows G and C are the only metric and connection sources available to the local certificate, and their dependency surfaces are already the public rows of $\mathsf{RiemannianMetric}^{\uparrow}$ and $\mathsf{Connection}^{\uparrow}$.

The torsion-free and metric-compatibility clauses are the two displayed boundary rows T and M. They are read after the connection row has exposed tangent-direction and derivative-endpoint data and after the metric row has exposed the fibrewise inner-product surface.

The curvature-facing clause is exactly R. It can be consumed only after the connection and compatibility rows have been replayed by K, so the downstream $\mathsf{Curvature}^{\uparrow}$ read receives a finite boundary route rather than a detached tensor object.

Transport and replay exactness are then componentwise projections from H and K. The package row P and local name row N repack these rows as the whole local NameCert surface, leaving no coordinate for the excluded ambient objects. □

Theorem 2673.3 (Levi-Civita connection curvature compatibility)**.** *Every* $\mathsf{Curvature}^{\uparrow}$ *consumer of an accepted* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier reads curvature through the finite route*

$$G \longmapsto C \longmapsto T \longmapsto M \longmapsto R.$$

Thus the curvature row is downstream of the displayed $\mathsf{RiemannianMetric}^{\uparrow}$ *metric packet, the displayed* $\mathsf{Connection}^{\uparrow}$ *derivative packet, the torsion-free boundary, and the metric-compatibility row. It cannot be read as an independent coordinate curvature tensor, a host connection form, or a curvature object detached from the metric-compatible connection certificate.*

Proof. Start with a curvature-facing request. The carrier has only one curvature coordinate, namely R, and R is declared as a compatibility row over the displayed connection row C.

The connection row C is not detached from the metric. The torsion-free row T and metric-compatibility row M sit between C and R in the replay row K, while G supplies the metric source that M preserves.

Apply the connection curvature-boundary surface from Definition 134.5 and the curvature source obligations of chapter 135. These existing rows explain how a connection boundary becomes consumable by $\mathsf{Curvature}^{\uparrow}$.

Repacking the projected rows gives the displayed route $G \mapsto C \mapsto T \mapsto M \mapsto R$. Since no coordinate curvature tensor, host connection form, or detached curvature object appears in the carrier, no such object is exported to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3638.1, Definition 3638.2, and Definition 3638.3 seed the displayed metric-compatible torsion-free connection route from $\mathsf{RiemannianMetric}^{\uparrow}$ and $\mathsf{Connection}^{\uparrow}$ rows through torsion-free, metric-compatible, curvature-facing, transport, replay, provenance, and local naming boundaries over $\mathsf{Curvature}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close coordinate Christoffel symbols or formulae, selected smooth atlases, global geodesic theorems or flows, host-level connection uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, torsion-free exposure, metric-compatibility exposure, curvature-facing compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LeviCivitaConnectionUp.

Constructive story: A $\mathsf{LeviCivitaConnection}^{\uparrow}$ packet is generated from displayed Riemannian metric and connection rows, torsion-free and metric-compatibility boundary rows, a curvature-facing compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2674

A Concrete Naming Certificate for $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ packet records completion as a finite transformation of metric names rather than as an ambient quotient or selected limit. It sits on the dyadic endpoint discipline of chapter 291, the finite observation route of chapter 126, the regular rational sequence row of chapter 301, the terminal real-name seal of chapter 38, the regular-Cauchy completion packet of chapter 1639, the functor rows of chapter 423, and the extension-uniqueness surface of chapter 397.

Definition 2674.1 (Bishop Cauchy completion functor carrier)**.** *A* $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, R, E, F, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, containing the named points, displayed distance observations, and finite tolerance ledger read by the source classifier.*
2. *T is the target metric row, containing the metric surface into which transported Cauchy data are read.*
3. *R is the regular-Cauchy family row, exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules and stream windows used by the completion handoff.*
4. *E is the completion seal row, recording the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary reached from S and R.*
5. *F is the transported map row, storing the displayed map action on source rows and the finite replay by which regular-Cauchy families are sent to target completion rows.*
6. *U is the extension-uniqueness row, read through the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *surface and tied to the same source, target, regular-Cauchy, and seal rows.*
7. *H records componentwise* hsame *transport for source, target, regular-Cauchy, completion seal, transported map, extension, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for a completion-facing consumer from S, T, R through E, F, U and back to the local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BishopRegularCauchyCompletio CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$ *evidence.*

The carrier contains no quotient completion, no selected limit, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyCompletionFunctorUp.BishopCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 2674.2 (Bishop Cauchy completion functor NameCert obligations)**.** *For every accepted carrier $Q = (S, T, R, E, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$ *surface consists of six finite obligation rows: carrier admission for $S, T, R, E, F, U, H, C, P, N$; classifier stability by componentwise* hsame *transport; functorial map transport through the displayed source, target, regular-Cauchy, and seal rows; unit handoff from source metric points into the completion seal; extension uniqueness through the row U; and non-escape from quotient completions, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is coordinate projection from Definition 3639.1. The packet exposes the source and target metric rows, regular-Cauchy family row, completion seal, transported map row, extension row, and the structural rows H, C, P, N as one finite history.

Classifier stability is componentwise. The row H supplies the only equality-like transport, and C replays the same completion-facing route through S, T, R, E, F, U whenever a consumer replaces one accepted packet by a classified mate.

Functorial map transport and unit handoff are sibling-row compositions. The source row S and regular-Cauchy row R feed the completion seal E; the transported map row F carries that route to the target row T; and the unit read is the special case in which a displayed source point enters the same seal route without naming a host limit.

Extension uniqueness is read only from U after E and F have been fixed. Since no carrier coordinate stores a quotient completion, selected representative, choice schedule, or ambient Real-completeness theorem, coordinate exhaustion gives the non-escape clauses. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$.

Constructive story: A BishopCauchyCompletionFunctor$^{\uparrow}$ packet is generated from source and target metric rows, a regular-Cauchy family row, a completion seal row, a transported map row, an extension-uniqueness row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 3639.1 and Definition 3639.2 bind the seed carrier and its finite obligation rows over the displayed completion-functor surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not assert quotient completion construction, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, functorial map transport, unit handoff, extension uniqueness, and non-escape rows for the displayed Bishop completion-functor packet.

Chapter 2675

A Concrete Naming Certificate for $\mathsf{RealLocatedOrder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLocatedOrder}^{\uparrow}$ packet names located comparison for two displayed real names by reading only regular Cauchy data, finite stream windows, dyadic tolerance budgets, a located comparison row, and an apartness handoff before any completion-facing consumer sees an order claim. It consumes the terminal $\mathsf{Real}^{\uparrow}$ seals of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the positive-separation boundary of chapter 304, and the finite located-order test of chapter 2112. The packet is a public constructive located-order certificate over $\mathsf{Real}^{\uparrow}$ names; it is not host trichotomy, host real equality, quotient stream equality, selected limits, countable choice, or ambient real completeness.

Definition 2675.1 (Real located-order carrier)**.** *A* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, R, S, D, O, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Real}^{\uparrow}$ *seal rows whose order relation is being read.*
2. *R stores the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks for the two real names.*
3. *S stores the shared* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to inspect the paired approximants.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that fixes the visible precision budget.*
5. *O is the located comparison row imported from the* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ *surface.*
6. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *handoff row, present only when the comparison exposes positive separation.*
7. *H stores componentwise* hsame *transport for the two real seals, readbacks, windows, dyadic tolerances, located comparison, apartness handoff, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from Real-name reads through regular Cauchy readbacks, stream windows, dyadic tolerances, located comparison, and apartness handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RealLocatedOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, R, S, D, O, A, H, C, P, N$. It has no coordinate for host trichotomy, host real equality, quotient stream equality, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderTasteGate single carrier alignment`

Theorem 2675.2 (Real located-order NameCert obligations)**.** *For every accepted* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier* $L = (X, Y, R, S, D, O, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{Real}^{\uparrow}$ *seals, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, located comparison row, optional* $\mathsf{ApartnessReal}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderCarrier namecert obligations`

Proof. Project the carrier of Definition 3640.1. The two real names are visible only as the seal rows X, Y, and their numerical reads pass through the paired regular rational row R.

The finite windows S and dyadic tolerance ledger D are then the only inspection surface available to the located comparison row O. Thus the local NameCert reads order evidence as the displayed CauchyRealLocatedOrderTest route rather than as a host order theorem.

Finally read A, H, C, P, N. The apartness row is a handoff when positive separation is displayed, while the structural rows transport, replay, source, and name the same finite route. Since no remaining coordinate stores trichotomy, host equality, quotient stream equality, selected limits, choice, or ambient completeness, the listed rows exhaust the obligation surface. □

Theorem 2675.3 (Real located-order apartness handoff)**.** *For every accepted* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier* $L = (X, Y, R, S, D, O, A, H, C, P, N)$*, an apartness-facing consumer may read positive separation only through the route*

$$X, Y \longmapsto R \longmapsto S \longmapsto D \longmapsto O \longmapsto A$$

together with H, C, P, N*. The handoff exports an* $\mathsf{ApartnessReal}^{\uparrow}$ *row when the located comparison displays a positive gap, but it exports no host real equality test, quotient-stream equality, trichotomy, selected limit, countable-choice selector, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderCarrier apartness handoff`

Proof. Start with the located comparison row in Definition 3640.1. The row O is downstream of the two Real seals, paired regular readbacks, finite windows, and dyadic tolerance ledger.

Use the apartness clause of the carrier. The row A is not an independent equality or order oracle; it is the ApartnessReal handoff available only when the displayed located comparison has exposed positive separation.

The structural rows H, C, P, N preserve this same packet route. Therefore a consumer can export the apartness handoff, but cannot bypass the displayed rows to obtain host equality, quotient stream equality, trichotomy, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealLocatedOrder}^{\uparrow}$ consumer must exhibit two RealUp seals, paired RegSeqRat readbacks, StreamName finite windows, a DyadicRatCore tolerance budget, a CauchyRealLocatedOrderTest row, an ApartnessReal handoff when positive separation is claimed, componentwise transport, displayed replay, provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore, apartnessreal, cauchy-real-located-order-test: no listed sibling carries the whole packet $L = (X, Y, R, S, D, O, A, H, C, P, N)$

because each sibling supplies only real sealing, rational readback, finite windows, tolerance budgets, apartness handoff, or located comparison while this chapter binds all rows as one Real-facing located-order certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}$.

Constructive story: A $\mathsf{RealLocatedOrder}^{\uparrow}$ packet is generated from two RealUp seals, paired RegSeqRat readbacks, StreamName finite windows, a DyadicRatCore tolerance budget, a CauchyRealLocatedOrderTest located comparison row, an ApartnessReal handoff when positive separation is displayed, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3640.1, Definition 3640.2, and Definition 3640.3 seed the finite constructive located-order certificate over two displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, host real equality, quotient stream equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, apartness handoff, and TasteGate field faithfulness for BEDC.Derived.RealLocatedOrderUp.

Chapter 2676

A Concrete Naming Certificate for $\mathsf{RealSequenceSpace}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of sequences of real numbers, the $\mathsf{RealSequenceSpace}^{\uparrow}$ packet records a real-valued sequence carrier as finite BEDC route data rather than as a quotient of streams or a completed function space. It sits between the finite stream windows of chapter 126, the regular rational names of chapter 301, the dyadic tolerance core of chapter 291, the finite real-name boundary of chapter 38, and the Cauchy convergence handoff of chapter 1509.

Definition 2676.1 (RealSequenceSpace carrier)**.** *A* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, E, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite sequence windows visible to the carrier.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row supplying regular rational names for the inspected real-valued coordinates.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those inspected coordinates.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the scheduled regular rows and dyadic tolerances have been displayed.*
5. *K is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff row for consumers that ask for Cauchy behavior of the displayed real-valued sequence.*
6. *H records componentwise* hsame *transport for the stream schedule, regular names, dyadic ledger, Real seal, Cauchy handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each finite window request through S, R, D, E, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealSequenceSpaceUp.RealSequenceSpaceTasteGate single carrier alignment`

Theorem 2676.2 (RealSequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier* $Q = (S, R, D, E, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$ *surface consists exactly of finite* BHist *packet rows over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$*:*

1. *carrier admission for the displayed stream schedule, regular rational coordinate names, dyadic tolerance ledger, Real seal, Cauchy-criterion handoff, transport, replay, provenance, and local name;*
2. *finite-window classification, requiring every coordinate read to pass through the* $\mathsf{StreamName}^{\uparrow}$ *row* S*;*
3. *regular-coordinate admission, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to be read only at windows named by* S*;*
4. *tolerance admission, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *to accompany every displayed regular coordinate request;*
5. *Real-seal ordering, requiring the* $\mathsf{Real}^{\uparrow}$ *row* E *to be downstream of* S, R, D*;*
6. *Cauchy-handoff ordering, requiring the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row* K *to consume only the same displayed* S, R, D, E *route;*
7. *transport and replay stability through* H, C, P, N*.*

No obligation supplies quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.

Proof. Project the carrier of Definition 3641.1. The packet exposes only the stream schedule, regular rational coordinate rows, dyadic tolerance ledger, Real seal, Cauchy-criterion handoff, and structural rows.

The finite-window obligations are then forced. A coordinate read has no source except S, the regular name row R is read only at those windows, and the dyadic ledger D records the tolerance data attached to the same request.

The Real and Cauchy rows are downstream. The seal E is reached only after the displayed S, R, D route, and the handoff K reuses that same route when a Cauchy-facing consumer reads the sequence.

Finally the rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier definition contains none of the excluded host-side data, the listed rows exhaust the local NameCert surface. □

Theorem 2676.3 (RealSequenceSpace Cauchy criterion handoff)**.** *Every* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier* $Q = (S, R, D, E, K, H, C, P, N)$ *reads a finite* BHist *packet route over*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyConvergenceCriterion}^{\uparrow}$$

in the ordered form

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K.$$

Thus a convergence-facing read may inspect only scheduled finite windows, regular rational coordinate rows, dyadic tolerance rows, the finite Real seal, and the displayed Cauchy handoff. It cannot extract quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness from the sequence-space packet.

Proof. Start from an accepted convergence-facing read. By Definition 3641.2, the read first enters the finite $\mathsf{StreamName}^{\uparrow}$ schedule and then the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ coordinate rows.

The tolerance and seal rows are fixed by the same packet. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D records the precision budget for the inspected coordinates, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of S, R, D.

The $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ row K consumes only that displayed route. The structural rows H, C, P, N preserve and name the same finite packet, so no quotient-stream representative, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient completeness theorem is available to the consumer. □

Theorem 2676.4 (RealSequenceSpace finite-window classifier determinacy)**.** *Let*

$$Q = (S, R, D, E, K, H, C, P, N) \qquad \textit{and} \qquad Q' = (S', R', D', E', K', H', C', P', N')$$

be accepted $\mathsf{RealSequenceSpace}^{\uparrow}$ *carriers. If Q and Q' display the same* $\mathsf{StreamName}^{\uparrow}$ *schedule, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{Real}^{\uparrow}$ *seal, the same* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff, and componentwise matching* hsame *and* Cont *rows, then their public classifier reads are the same. Conversely, any public classifier disagreement between Q and Q' must occur in one of those displayed finite rows; it cannot be hidden in quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.*

Proof. Project both carriers through Definition 3641.1. The only classifier coordinates exposed by either packet are the stream schedule, regular rational coordinate rows, dyadic tolerance ledger, Real seal, Cauchy handoff, transport, replay, provenance, and local naming rows.

Apply Definition 3641.2 to both packets. The theorem fixes the ordering of finite-window admission, regular-coordinate admission, tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability, so matching displayed rows force matching public reads at every permitted coordinate.

It remains to exclude hidden disagreement. By Definition 3641.3, every convergence-facing read factors through the same ordered route

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K.$$

Thus no quotient stream, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient completeness theorem can supply an additional classifier coordinate. Any disagreement must therefore be visible in one of the displayed finite rows. □

Theorem 2676.5 (RealSequenceSpace window-seal obligation closure)**.** *For every accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier $Q = (S, R, D, E, K, H, C, P, N)$, the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff,* hsame *transport,* Cont *replay, provenance, and local* $\mathsf{NameCert}$ *jointly close the local obligation surface. Every sequence-space consumer must read*

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$$

before any public sequence-space classifier is exported, and any classifier disagreement must appear in one displayed finite row rather than in a quotient stream, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient Real completeness theorem.

Proof. Project the carrier of Definition 3641.1. The projection exposes exactly the window schedule S, coordinate row R, tolerance ledger D, Real seal E, Cauchy handoff K, and structural rows H, C, P, N.

Apply Definition 3641.2. This fixes the row ordering from finite-window admission through regular-coordinate admission, dyadic tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability.

Apply Definition 3641.3 for convergence-facing reads. Such a consumer can inspect only the ordered route $S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$, so the Cauchy side receives no additional host equality or completeness coordinate.

Finally apply Definition 3641.4. Matching displayed rows force matching public reads, and any remaining classifier disagreement is therefore visible in one displayed finite row rather than hidden in a quotient-stream or host-equality layer. □

Theorem 2676.6 (RealSequenceSpace tail-zero subspace route)**.** *Let $Q = (S, R, D, E, K, H, C, P, N)$ be an accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier equipped with a displayed dyadic tail modulus T whose entries are read through the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger as S, R, D. Then every tail-zero subspace consumer factors through the route*

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$$

before it exports the tail-zero surface. The extra tail claim is visible only as the displayed D, K-facing ledger T; it does not add quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3641.1. The sequence source, regular coordinate readback, dyadic tolerance rows, Real seal, and Cauchy handoff are already the rows S, R, D, E, K, and the tail modulus T is displayed over the same finite windows.

Apply Definition 3641.2 and Definition 3641.3. These results force every Real-facing or Cauchy-facing read to reuse the ordered route through S, R, D, E, K before any public sequence-space conclusion is exported.

The tail-zero subspace surface therefore adds no new equality coordinate. It records that the dyadic tail requests in T are ledger entries facing D and K, while the structural rows H, C, P, N transport and name the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSequenceSpace}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ coordinate rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, a $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3641.1, Definition 3641.2, and Definition 3641.3 seed the real-valued sequence-space carrier over finite StreamName, RegSeqRat, DyadicRatCore, Real, and Cauchy convergence rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window classification, regular-coordinate admission, dyadic tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability for the same displayed packet.

Chapter 2677

A Concrete Naming Certificate for $\mathsf{FrechetUrysohn}^{\uparrow}$

Following Frechet and Urysohn's sequential analysis of closure in topology, the $\mathsf{FrechetUrysohn}^{\uparrow}$ packet records the finite data by which a point is certified as sequentially reachable from a displayed subset rather than assuming an ambient topological closure operator. It sits between the finite real-valued sequence surface of chapter 3641, the sequential compactness packet of chapter 2372, and the finite stream windows of chapter 126.

Definition 2677.1 (Frechet-Urysohn carrier)**.** *A* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, S, W, R, A, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology carrier row, containing the named point, the displayed subset, and the finite neighborhood probes used by the closure read.*
2. *S is the sequence-space row over* $\mathsf{RealSequenceSpace}^{\uparrow}$*, recording the accepted real-valued or chart-valued sequence route used to approach the named point.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the finite initial segments and tail windows visible to the classifier.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the coordinate reads that witness convergence through S and W.*
5. *A is the closure-witness row, displaying that every inspected tail window lies in the named subset and that the limit read is the named point.*
6. *X is the transport row for the topology probes, sequence-space rows, StreamName windows, Real seal, closure witness, replay, provenance, and local naming data.*
7. *H records componentwise* hsame *transport along the same displayed rows.*
8. *C records displayed* Cont *replay from a closure request through T, S, W, R, A, X.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohn}^{\uparrow}$, $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$ *row.*

The classifier has no coordinate for arbitrary nets, choice-selected sequences, host closure equality, excluded middle over neighborhoods, or an ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.FrechetUrysohnUp.FrechetUrysohnTasteGate single carrier alignment`

Theorem 2677.2 (Frechet-Urysohn NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier* $Q = (T, S, W, R, A, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$ *surface consists exactly of finite* BHist *rows for topology admission, sequence-space admission, StreamName windowing, Real-seal ordering, closure-witness display, transport, replay, provenance, and local naming. Every sequential-closure classifier read must pass through* $T, S, W, R, A, X, H, C, P, N$ *in that order, and any disagreement between two accepted packets must be visible in one of those displayed rows.*

Proof. Project the carrier of Definition 3642.1. The packet exposes the topology row, the real sequence-space row, the finite StreamName windows, the Real seal, and the closure-witness row before any structural replay is available.

The NameCert obligations are therefore row-local. Topology admission is read from T, the sequence route from S, the finite windows from W, the coordinate seal from R, and the sequential closure witness from A.

The structural rows X, H, C, P, N then transport, compare, replay, source, and name the same finite packet. Since the carrier contains none of the excluded host-side coordinates, a classifier disagreement cannot be hidden outside the displayed rows. □

Theorem 2677.3 (Frechet-Urysohn sequential closure handoff)**.** *Every consumer that reads a point as belonging to the sequential closure of a displayed subset through an accepted* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSequenceSpace}^{\uparrow}, \qquad \mathsf{SequentialCompact}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

and then through the closure-witness row A*. The handoff exports only the finite sequence windows, the accepted sequence-space route, the Real seal, and the displayed tail-in-subset witness; it does not export arbitrary nets, countable choice, host closure equality, or ambient topological compactness.*

Proof. Start with an accepted sequential-closure read. By Definition 3642.2, the read first enters the topology row T and then the sequence-space row S with its displayed StreamName window schedule W.

The Real seal R and the closure-witness row A are downstream of the same sequence route. The witness can therefore certify only the inspected tail windows and the displayed limit read named by the packet.

The sequential compactness reference is consumed as a finite comparison boundary rather than as an ambient compactness theorem. The structural rows X, H, C, P, N preserve this same route, so the consumer receives a sequential handoff but no hidden net, selected sequence, host equality, or external closure principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetUrysohn}^{\uparrow}$ packet is generated from a topology carrier row, a real sequence-space row, StreamName windows, a Real seal, a closure-witness row, transport, componentwise hsame rows, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3642.1, Definition 3642.2, and Definition 3642.3 seed the sequential-closure carrier over finite topology, sequence-space, window, Real, and witness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FrechetUrysohnUp.FrechetUrysohnTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim arbitrary nets, countable choice, host closure equality, excluded middle over neighborhoods, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-space admission, finite-window ordering, Real-seal ordering, closure-witness display, and transport stability for the same packet.

Chapter 2678

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ packet realizes the boundary-phase audit face of chapter 9463 as finite NameCert data. It sits after the observer-expansion ledger of chapter 3486, the boundary-geodesic route of chapter 3491, the horocycle packet of chapter 3534, the phase-ledger horocycle packet of chapter 3591, and the boundary-fixed disk transport route of chapter 3500. The packet names an audit row for checking that a visible boundary-phase read has passed through the displayed hyperbolic ledger, boundary, horocycle, and transport surfaces; it is not a second horocycle carrier, not a host ideal-boundary quotient, and not a physical observer trajectory.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ read factors through displayed $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows before it exports a boundary-phase audit verdict.

Independence witness: hyperbolic-expansion-ledger, hyperbolic-geodesic-boundary, hyperbolic-horocycle, phase-ledger-horocycle, poincare-boundary-fixed-transport: the expansion ledger records records-side growth without a boundary-phase audit verdict; the geodesic-boundary packet records far-end incidence without horocycle phase accounting; the horocycle packet records boundary tangency without focused phase-ledger provenance; the phase-ledger horocycle packet records the boundary-transport route without a separate audit verdict; and the boundary-fixed transport packet records preserved disk transport without ledger-phase coverage. None is bijective to $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$.

Definition 2678.1 (Hyperbolic boundary-phase audit carrier)**.** *A* $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, G, Q, L, T, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *observer-expansion ledger row.*
2. *G is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary-geodesic incidence row.*
3. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *boundary-tangent row.*
4. *L is the* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *focused phase-ledger route.*

5. *T is the* PoincareBoundaryFixedTransport$^{\uparrow}$ *boundary-preserving disk-transport row.*
6. *V is the audit verdict row, recording that the visible boundary-phase read has consumed X, G, Q, L, T before any public verdict is exported.*
7. *H records componentwise* hsame *transport for the expansion ledger, boundary-geodesic, horocycle, phase-ledger, transport, audit, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the same ordered audit route.*
9. *P is the* Pkg *provenance over* HyperbolicBoundaryPhaseAudit$^{\uparrow}$, HyperbolicExpansionLedger$^{\uparrow}$, HyperbolicGeodesicB HyperbolicHorocycle$^{\uparrow}$, PhaseLedgerHorocycle$^{\uparrow}$, PoincareBoundaryFixedTransport$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$ *row.*

The carrier exports no host hyperbolic plane, quotient ideal-boundary equality, selected boundary representative, global clock, physical observer path, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryPhaseAuditUp.HyperbolicBoundaryPhaseAuditTasteGate`
`single carrier alignment`

Theorem 2678.2 (Hyperbolic boundary-phase audit NameCert obligations)**.** *For every accepted* HyperbolicBoundaryPhaseAudit$^{\uparrow}$ *carrier $A = (X, G, Q, L, T, V, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicBoundaryPhaseAu}}$ *surface consists exactly of observer-expansion ledger admission, boundary-geodesic incidence, horocycle boundary tangency, focused phase-ledger route admission, boundary-fixed transport, audit verdict exposure, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the accepted carrier. The analytic route begins with the expansion ledger X, then reads the boundary-geodesic row G, the horocycle row Q, the phase-ledger horocycle row L, and the boundary-fixed transport row T.

The audit verdict V is downstream of those five rows. It therefore cannot certify a boundary-phase read unless the expansion, boundary, horocycle, phase-ledger, and transport surfaces have already been displayed.

The structural rows H, C, P, N preserve and name that route. Since the carrier has no coordinate for a host hyperbolic plane, quotient boundary equality, selected boundary representative, global clock, physical observer path, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 2678.3 (Hyperbolic boundary-phase audit route nonescape)**.** *Every boundary-phase audit verdict exported by an accepted* HyperbolicBoundaryPhaseAudit$^{\uparrow}$ *packet factors through*

$$\text{HyperbolicExpansionLedger}^{\uparrow} \longmapsto \text{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \text{HyperbolicHorocycle}^{\uparrow} \longmapsto \text{PhaseLedgerHorocycle}$$

before the audit verdict row is visible. The verdict exports only the finite audit route carried by $A = (X, G, Q, L, T, V, H, C, P, N)$, not a host ideal-boundary quotient, a selected boundary representative, a physical observer trajectory, a global clock, or a completed hyperbolic surface.

Proof. Start with the NameCert obligations of Definition 3643.2. They place the expansion ledger, boundary-geodesic row, horocycle row, phase-ledger horocycle route, and boundary-fixed transport before the audit verdict.

Read the sibling packets in order. Definition 3486.3 supplies the records-side expansion route, Definition 3491.3 supplies the boundary-geodesic handoff, Definition 3534.2 supplies the horocycle boundary-tangency route, and Definition 3591.2 supplies the focused phase-ledger route.

Finally consume the boundary-fixed transport row and the structural rows H, C, P, N. Because no other carrier coordinate exists, the audit verdict cannot escape to host boundary equality, selected representatives, physical observer motion, global clocks, or completed hyperbolic geometry. □

The Lean follow-up for this packet must provide explicit Nontrivial HyperbolicBoundaryPhaseAudit$^{\uparrow}$ and FieldFaithful HyperbolicBoundaryPhaseAudit$^{\uparrow}$ TasteGate instances. Those instances must keep $X, G, Q, L, T, V, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$, seed)

Scope closed: Definition 3643.1, Definition 3643.2, and Definition 3643.3 seed the finite boundary-phase audit route over expansion-ledger, boundary-geodesic, horocycle, phase-ledger, boundary-fixed transport, audit, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperbolic geometry, quotient ideal-boundary equality, selected boundary representatives, physical observer trajectories, global clocks, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, expansion-ledger routing, boundary-geodesic handoff, horocycle incidence, phase-ledger coverage, boundary-fixed transport, audit verdict exactness, TasteGate nontriviality, FieldFaithful coverage, and local naming for BEDC.Derived.HyperbolicBoundaryPhaseAuditUp.

Constructive story: A HyperbolicBoundaryPhaseAudit$^{\uparrow}$ packet is generated from a HyperbolicExpansionLedger row, HyperbolicGeodesicBoundary row, HyperbolicHorocycle row, PhaseLedgerHorocycle route, PoincareBoundaryFixedTransport row, audit verdict row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2679

A Concrete Naming Certificate for $\mathsf{SardTheoremFiniteJet}^{\uparrow}$

Following Sard and the finite-jet presentation of differential topology, the $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet records the displayed local data by which a smooth map is inspected through finitely many derivative, chart, rank, and critical-value rows. It sits after the Frechet derivative surface of chapter 3634, the local inverse-function route of chapter 2055, the comparison discipline of chapter 1912, and the charted carrier of chapter 96. The packet names a finite jet certificate for critical-value exclusion windows; it is not the classical full measure-zero Sard theorem, not a transversality theorem, not a choice-selected regular-value principle, and not a standard bridge.

Definition 2679.1 (Sard theorem finite-jet carrier)**.** *A* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, F, D, R, V, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row naming the source and target chart windows in which the finite jet is read.*
2. *F is the displayed map row, containing only the finite map observations inspected by the packet.*
3. *D is the* $\mathsf{FrechetDerivative}^{\uparrow}$ *jet row, recording the derivative orders and finite remainder windows consumed by the critical-value test.*
4. *R is the rank-defect row, displaying which derivative minors are inspected and where the finite packet records criticality.*
5. *V is the candidate value-window row, naming the finite output windows whose regular-value status is being tested.*
6. *L is the local inverse or mean-value ledger supplied through* $\mathsf{InverseFunction}^{\uparrow}$ *and* $\mathsf{MeanValueTheorem}^{\uparrow}$ *when a displayed noncritical branch is certified.*
7. *H records componentwise* hsame *transport for the chart, map, derivative, rank, value-window, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite value-window request through the chart, map, derivative, rank, and local ledger rows.*
9. *P is the* Pkg *provenance over* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{InverseFunction}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for ambient measure zero, global transversality, choice-selected regular values, host smooth-function spaces, or a standard bridge.

Lean4 ✓ `BEDC.Derived.SardTheoremFiniteJetUp.TasteGate.SardTheoremFiniteJetTasteGate single carrier alignment`

Theorem 2679.2 (Sard theorem finite-jet NameCert obligations)**.** *For every accepted* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *carrier* $J = (M, F, D, R, V, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *charted carrier admission through* $\mathsf{Manifold}^{\uparrow}$ *for the source and target windows;*
2. *finite derivative admission through* $\mathsf{FrechetDerivative}^{\uparrow}$ *for the displayed jet orders and remainders;*
3. *rank-defect and value-window inspection through R and V;*
4. *noncritical local handoff through* $\mathsf{InverseFunction}^{\uparrow}$ *and* $\mathsf{MeanValueTheorem}^{\uparrow}$ *when the packet records a regular finite window;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Every accepted finite-jet regularity read factors through these five rows, and any disagreement between two accepted packets is visible in one of them.

Proof. Project the carrier of Definition 3644.1. The rows $M, F, D, R, V, L, H, C, P, N$ are the whole accepted packet, so the chart, map, derivative, rank, value-window, and local ledger reads are finite BHist data.

The critical-value boundary is read only from D, R, V. The derivative row supplies the inspected jet windows, the rank-defect row records the displayed minors, and the value row records the finite output windows under consideration.

When a noncritical branch is exposed, the local ledger L records the inverse-function and mean-value handoff. That handoff is local to the same chart and derivative rows, so it does not choose a global regular value or import a host transversality principle.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores ambient measure-zero data, global transversality, selected regular values, host smooth-function spaces, or bridge data, the five obligation rows exhaust the NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet must expose $\mathsf{Manifold}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{InverseFunction}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows before any finite-jet regularity read is public.

Independence witness: frechet-derivative, inverse-function, mean-value-theorem, manifold: no listed sibling is bijective to $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ because the Sard finite-jet packet jointly binds chart windows, map observations, derivative jets, rank-defect rows, value windows, local regularity ledger, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet is generated from Manifold chart rows, displayed map observations, Frechet derivative jet rows, rank-defect rows, finite value-window rows, an inverse-function and mean-value local ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SardTheoremFiniteJet}\uparrow}$, seed)

Scope closed: Definition 3644.1 and Definition 3644.2 seed the finite Sard jet certificate over chart, derivative, rank, value-window, local regularity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SardTheoremFiniteJet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the classical full measure-zero Sard theorem, transversality, choice-selected regular values, host smooth-function spaces, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite derivative admission, rank-defect inspection, value-window inspection, local inverse-function and mean-value handoff, and transport stability for BEDC.Derived.SardTheoremFiniteJetU

Chapter 2680

A Concrete Naming Certificate for $\mathsf{BusemannBoundary}^{\uparrow}$

Following Busemann and Bridson–Haefliger, the $\mathsf{BusemannBoundary}^{\uparrow}$ packet records boundary reads by finite distance-difference rows rather than by a completed host ideal-boundary quotient. It sits after the metric-read surface of chapter 3474, the geodesic far-end carrier of chapter 3491, the boundary-fixed transport route of chapter 3500, and the disk-coordinate surface of chapter 377.

Definition 2680.1 (Busemann boundary carrier)**.** *A* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, o, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *source row, exposing the metric carrier from which all distance-difference reads are taken.*
2. *o is the basepoint row, recording the named origin point and its admission through the same metric source.*
3. *R is the geodesic-ray window row, displaying the finite ray segments and endpoint probes supplied by* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*.*
4. *D is the dyadic distance-difference ledger for the inspected values $d(x, \gamma(t)) - d(o, \gamma(t))$ over the displayed windows.*
5. *L is the boundary-read row, classifying the finite Busemann read against the visible ideal-boundary data of* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*, and* $\mathsf{UnitDisk}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for the metric source, basepoint, ray windows, dyadic ledger, boundary-read row, replay, provenance, and local naming data.*
7. *C records displayed* Cont *replay from a boundary request through X, o, R, D, L.*
8. *P is the* Pkg *provenance over* $\mathsf{BusemannBoundary}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$ *row.*

The classifier has no coordinate for an arbitrary compactification, a host quotient of ideal points, a globally chosen geodesic representative, classical sequential compactness at infinity, or an ambient visual boundary not displayed by the packet.

Lean4 ✓ `BEDC.Derived.BusemannBoundaryUp.BusemannBoundaryTasteGate single carrier alignment`

Theorem 2680.2 (Busemann boundary NameCert obligations)**.** *For every accepted* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier* $B = (X, o, R, D, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$ *surface consists exactly of finite* BHist *rows for carrier admission, ray-window exactness, metric-read stability, boundary classifier laws, and nonescape from the host ideal-boundary quotients. Every boundary consumer must read* X, o, R, D, L *before transport, replay, provenance, and local naming are available, and any disagreement between accepted Busemann reads must occur in one of those displayed rows.*

Proof. Project the carrier of Definition 3645.1. The packet exposes a hyperbolic metric source, a basepoint, geodesic-ray windows, a dyadic distance-difference ledger, and a boundary-read row before any structural replay is present.

Carrier admission is therefore row-local. Ray-window exactness is read from R, metric-read stability from the shared X, o source, and boundary classifier laws from L against the displayed geodesic, transport, and disk rows.

The structural rows H, C, P, N then preserve, replay, source, and name the same finite surface. Since the carrier contains no coordinate for a completed host quotient or an unrecorded compactification, the read cannot escape the displayed ideal-boundary data. □

Theorem 2680.3 (Busemann boundary nonescape)**.** *Every accepted boundary read exported by a* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \quad \mathsf{HyperbolicGeodesicBoundary}^{\uparrow}, \quad \mathsf{PoincareBoundaryFixedTransport}^{\uparrow}, \quad \mathsf{UnitDisk}^{\uparrow}$$

and then through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the same packet. The exported read is therefore a finite distance-difference boundary certificate; it is not a host compactification point, a quotient class of rays, or an ambient visual-boundary object.*

Proof. Begin with an accepted boundary read. By Definition 3645.2, the read first enters the $\mathsf{HyperbolicMetric}^{\uparrow}$ row X and the basepoint row o, so each distance-difference entry is tied to the same displayed metric source.

The ray-window and boundary rows then force the read through $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, and $\mathsf{UnitDisk}^{\uparrow}$. The ledger D records only the finite dyadic distance-difference values attached to those windows, and L classifies only against the displayed boundary data.

Finally H, C, P, N transport, replay, source, and name this same route. No additional coordinate remains for a host ideal-boundary quotient, a global ray representative, or an ambient compactification, so every exported Busemann boundary read stays inside the named packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BusemannBoundary}^{\uparrow}$ packet is generated from a HyperbolicMetric source row, a basepoint row, geodesic-ray windows, a dyadic distance-difference ledger, a boundary-read row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3645.1, Definition 3645.2, and Definition 3645.3 seed the Busemann boundary carrier over finite metric, basepoint, ray-window, distance-difference, and boundary-read rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary compactifications, host ideal-boundary quotients, global geodesic representatives, classical compactness at infinity, ambient visual boundaries, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ray-window exactness, metric-read stability, boundary classifier laws, nonescape from host ideal-boundary quotients, transport, replay, provenance, and local naming for the same displayed packet.

Chapter 2681

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ packet records the unit route from displayed regular Cauchy source data into the completion-facing seal. It sits between the regular-name source of chapter 371, the unit map of chapter 1527, the monadic completion surface of chapter 378, and the real completion boundary of chapter 360. Its finite reads pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as exposed in chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the source-to-completion unit as finite BEDC data; it is not quotient completion equality, a selected Cauchy representative, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2681.1 (RegularCauchyCompletionUnit carrier)**.** *A* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (G, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the source row, read either as a* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet or as the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy row already visible to a completion consumer.*
2. *R is the unit-admission row stating that the displayed source row is admitted as the unit image before any completion-facing seal is consumed.*
3. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows through which the source row is inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those source windows and precision requests.*
5. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after G, R, S, D have been displayed.*
6. *H stores componentwise* hsame *transport for the source, unit-admission row, stream windows, dyadic tolerance ledger, seal handoff, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from source inspection through unit admission, finite windows, tolerance readback, and seal handoff.*
8. *P records* Pkg *provenance over* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, a selected regular-Cauchy representative, countable-choice schedules, host sequence equality, an ambient completion theorem, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUnitUp.RegularCauchyCompletionUnitTasteGate single carrier alignment`

Theorem 2681.2 (RegularCauchyCompletionUnit NameCert obligations)**.** *For an accepted* RegularCauchyComplet *carrier $U = (G, R, S, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$ *surface consists exactly of source admission through G, unit-route admission through R, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback through D,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3646.1. Its displayed source row is G, and the unit-admission row is R.

The finite observation surface is S, D. The stream windows expose only scheduled source reads, and the dyadic ledger records only the tolerance data needed before completion-facing consumption.

The seal handoff E is downstream of those rows. The structural rows H, C, P, N transport, replay, source, and name the same route, so the local certificate is exhausted by the listed fields. □

Theorem 2681.3 (RegularCauchyCompletionUnit seal handoff)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* RegularCauchyCompletionUnit *packet reaches the seal row E only through the route*

$$G \longmapsto R \longmapsto S \longmapsto D \longmapsto E$$

together with H, C, P, N. Thus the unit image is consumed as displayed regular Cauchy source data with finite windows and dyadic tolerance readback, not as a quotient class or selected completion representative.

Proof. Begin with a completion-facing consumer. By the carrier definition, the only source row available to the consumer is G, and the only unit-admission row is R.

The consumer must next pass through the finite windows S and the dyadic tolerance ledger D. These rows are the displayed readback route inherited from the regular-name and regular-rational source surfaces.

Only after that route is available can E be read. The rows H, C, P, N attach transport, replay, provenance, and local naming to the same route, so the seal handoff is the displayed unit route and not a hidden completion object. □

Theorem 2681.4 (RegularCauchyCompletionUnit non-escape)**.** *No accepted* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *packet exports data outside $G, R, S, D, E, H, C, P, N$. In particular, it cannot be used to recover quotient completion equality, a selected regular-Cauchy representative, countable-choice schedules, host sequence equality, an ambient completion theorem, analytic closure of the monad, or completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Inspect the carrier. The source, unit-admission, window, tolerance, seal, transport, replay, provenance, and local-name rows are the only fields.

Apply Definition 3646.2. Every accepted obligation is a projection of those rows or a structural transport, replay, provenance, or naming row for them.

Finally use Definition 3646.3. All completion-facing consumers pass through $G, R, S, D, E, H, C, P, N$, so no excluded quotient, choice, host, monadic closure, or completeness object can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ packet is generated from a RegularCauchyName or RegSeqRat source row, unit admission, finite StreamName windows, a DyadicRatCore tolerance ledger, a RealCauchyCompletion or RealUp seal handoff, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3646.1, Definition 3646.2, Definition 3646.3, and Definition 3646.4 seed the finite source-to-completion unit route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, selected representatives, countable choice, host sequence equality, monadic analytic closure, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, unit-route admission, finite window exposure, dyadic tolerance readback, seal handoff, transport stability, provenance, and non-escape for the displayed packet.

Chapter 2682

A Concrete Naming Certificate for $\mathsf{SobolevTrace}^{\uparrow}$

Following Lions–Magenes and Adams–Fournier, the $\mathsf{SobolevTrace}^{\uparrow}$ packet records finite BEDC boundary trace data for Sobolev carriers. It sits next to the weak-derivative packet of chapter 440, consumes the boundary-control route of chapter 2329, uses the compact-window handoff of chapter 2218, and reads scalar and metric data through chapter 38, chapter 57, and chapter 128. The packet names only the displayed boundary restriction surface; it does not claim a host trace operator on a completed Sobolev quotient.

Definition 2682.1 (Sobolev trace carrier)**.** *A* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, D, B, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, exposing the weak-derivative, integrability, codomain, transport, replay, provenance, and local naming rows already admitted by the Sobolev packet.*
2. *D is the boundary-domain row. It records the finite boundary chart, collar, normal-window, or boundary-read ledger on which the trace is inspected.*
3. *B is the Poincare-facing boundary-control row from* $\mathsf{PoincareInequality}^{\uparrow}$*, used only to keep constant or zero-boundary escape from entering the trace read.*
4. *R is the Rellich–Kondrachov compact-window row, recording the finite compactness or subsequence-window handoff consumed by boundary convergence tests.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *row that supplies the finite distance and Cauchy-readback surface for the boundary values.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for boundary magnitudes, radii, and norm estimates.*
7. *H stores componentwise* hsame *transport for S, D, B, R, M, E and the structural rows.*
8. *C stores the displayed* Cont *replay from a trace request through the Sobolev source, boundary row, Poincare row, compact-window row, metric row, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SobolevTrace}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*,* $\mathsf{PoincareInequality}^{\uparrow}$*,* $\mathsf{RellichKondrachov}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$ *row.*

The classifier has no coordinate for an ambient trace operator, quotient equality of boundary functions, selected representative, distributional boundary value, open-cover compactness theorem, or global completeness theorem for the real line.

Lean4 ✓ `BEDC.Derived.SobolevTraceUp.sobolevTraceChapterTasteGate`

Theorem 2682.2 (Sobolev trace NameCert obligations)**.** *For every accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier* $T = (S, D, B, R, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Sobolev source, boundary-domain row, Poincare control row, Rellich–Kondrachov compact-window row, metric or complete-metric row, real seal, transport, replay, provenance, and local naming data;*
2. *source compatibility, requiring every trace request to enter through the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative ledger before any boundary row is read;*
3. *boundary-control exactness, requiring* D *and* B *to supply the boundary restriction and normalization data consumed by the trace;*
4. *compact and complete-metric handoff, requiring* R *and* M *to expose only finite convergence, compact-window, distance, and Cauchy-readback rows;*
5. *scalar readback and transport, requiring* E, H, C, P, N *to preserve, replay, source, and name the same finite trace route.*

Proof. Project the carrier. The accepted packet exposes only the rows $S, D, B, R, M, E, H, C, P, N$, so carrier admission is row-local.

The source compatibility row is forced by S. A trace request first reads the $\mathsf{Sobolev}^{\uparrow}$ weak-derivative and magnitude ledger, then reaches the boundary-domain row D and the Poincare control row B.

The compact and complete-metric handoff is similarly row-local. The Rellich–Kondrachov row R supplies only finite compact-window data, and M supplies only the displayed metric or complete-metric readback surface.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier contains no ambient trace operator, host quotient, selected representative, or distributional boundary coordinate, the local NameCert obligations are exhausted by the five displayed rows. □

Lean4 ✓ `BEDC.Derived.SobolevTraceUp.SobolevTraceNamecertObligationSurface`

Theorem 2682.3 (Sobolev trace boundary readback)**.** *Every boundary value read exported by an accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *packet factors through*

$$\mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{PoincareInequality}^{\uparrow} \longmapsto \mathsf{RellichKondrachov}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

and then through the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the same packet. The exported object is a finite boundary readback certificate; it is not a host boundary function, quotient trace class, distributional trace, or selected representative.*

Proof. Start with an accepted trace read. By Definition 3647.2, the read first enters the Sobolev source row S, so it inherits only the weak-derivative and magnitude data visible in the source packet.

The boundary rows D and B then restrict the request to the displayed boundary ledger and Poincare-facing control surface. These rows determine which boundary observations are available before any compactness or completion-facing datum is consumed.

The remaining analytic read passes through R, M, E. Thus compact-window information, metric or complete-metric readback, and scalar magnitude are all finite rows of the same packet.

The structural rows H, C, P, N merely preserve, replay, source, and name this route. No carrier coordinate remains from which a host boundary function, quotient class, distributional trace, or selected representative could be read. □

Theorem 2682.4 (Sobolev trace complete-metric handoff)**.** *Any* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *PDE, variational, or boundary-analysis consumer citing an accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier receives only the complete-metric handoff displayed by* M*, the real scalar seal displayed by* E*, and the upstream rows* S, D, B, R*. In particular, a complete-metric consumer may request Cauchy-readback data for the boundary trace only after the Sobolev source, boundary-control, Poincare, and Rellich–Kondrachov rows have been read.*

Proof. Project a downstream consumer request. The carrier places M after S, D, B, R, so the complete-metric surface cannot be reached before the Sobolev source, boundary row, boundary-control row, and compact-window row are fixed.

The real scalar seal is read through E. It records the displayed boundary magnitudes and radii used by M, not a host completeness theorem for all real-valued boundary functions.

The transport, replay, provenance, and local naming rows keep this handoff inside the same BHist packet. Hence the complete-metric consumer receives only the finite Cauchy-readback route displayed by the trace carrier. □

Falsifiable prediction: Every $\mathsf{SobolevTrace}^{\uparrow}$ boundary read factors through displayed SobolevUp, PoincareInequalityUp, RellichKondrachovUp, MetricUp, CompleteMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any downstream trace consumer receives it.

Independence witness: sobolev, poincare-inequality, rellich-kondrachov, metric, completemetric, real: $\mathsf{SobolevTrace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Sobolev source, boundary-domain row, boundary-control row, compact-window row, complete-metric boundary readback, scalar seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevTrace}^{\uparrow}$ packet is generated from a Sobolev source row, boundary-domain row, Poincare control row, Rellich–Kondrachov compact-window row, metric or complete-metric readback row, real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$, seed)

Scope closed: Definition 3647.1, Definition 3647.2, Definition 3647.3, and Definition 3647.4 seed the finite boundary trace route over Sobolev, Poincare, Rellich–Kondrachov, metric, complete-metric, and real neighbours.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host trace operator, quotient equality of boundary functions, selected representatives, distributional boundary values, open-cover compactness, real-line completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, boundary-control exactness, compact and complete-metric handoff, scalar readback, componentwise transport, replay, provenance, and local naming.

Chapter 2683

A Concrete Naming Certificate for $\mathsf{NoetherianModule}^{\uparrow}$

Following Emmy Noether and standard module theory, the $\mathsf{NoetherianModule}^{\uparrow}$ packet records finite submodule-chain stabilization data over a displayed module source. In BEDC form it sits after $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ (chapter 46), the commutative scalar source of $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44), the ideal-side Noetherian surface of $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ (chapter 3637), and the ideal carrier of $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81). The packet supplies the module-side finite-chain row consumed by algebraic reduction and kernel-image arguments; it does not assert quotient-module classification, selected bases, maximal-submodule choice, or a host Noetherian theorem.

2683.1 Carrier definition

Definition 2683.1 (Noetherian module carrier)**.** *A* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N).$$

The row M is the $\mathsf{Module}^{\uparrow}$ *source surface. The row R is the scalar* $\mathsf{CommRing}^{\uparrow}$ *source read from the module certificate. The row I is the* $\mathsf{Ideal}^{\uparrow}$ *comparison surface used when scalar-generated submodule rows are inspected. The row A is the ascending-chain stabilization witness: every displayed chain of submodule rows over M carries a finite stage after which all later submodule endpoints classify with that stage. The row G is the finite-generator witness used by algebraic consumers that read the module through finitely generated submodule data. The row H is componentwise* hsame *transport for the module, scalar, ideal-comparison, stabilization, and generator rows. The row Q is displayed* Cont *replay for the submodule-chain and generator certificates. The row P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$ *naming row.*

Lean4 ✓ `BEDC.Derived.NoetherianModuleUp.TasteGate.NoetherianModuleTasteGate single carrier alignment`

Theorem 2683.2 (Noetherian module NameCert obligations)**.** *For an accepted* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier* $\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, source compatibility with* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{CommRing}^{\uparrow}$*,* $\mathsf{NoetherianRing}^{\uparrow}$*, and* $\mathsf{Ideal}^{\uparrow}$*, finite submodule-chain stabilization access through A, finite-generator access through G, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoetherianModuleUp.NoetherianModuleCarrier namecert obligations`

Proof. Project the nine rows of Definition 3648.1. The module row M fixes the additive carrier, scalar action, and module classifier before any submodule chain is inspected.

The scalar and ideal comparison rows R and I bind the chain to the same commutative-ring and ideal surfaces used by chapter 3637. Thus the $\mathsf{NoetherianRing}^{\uparrow}$ dependency supplies only the scalar-side finite-chain context; it does not replace the module chain by an ideal chain.

The stabilization row A and generator row G are then read over the displayed module classifier. Componentwise transport H carries the same rows through hsame, while Q replays only the displayed continuations.

Finally, P and N repack this inventory as the local naming certificate. Since no quotient-module classifier, basis selector, maximal-submodule choice, or host Noetherian principle occurs in the carrier, none can occur among the local obligations. □

Theorem 2683.3 (Noetherian submodule-chain stability)**.** *Let* $\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N)$ *be an accepted* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier. For every displayed ascending chain of submodule rows over the* $\mathsf{Module}^{\uparrow}$ *classifier of* M*, the witness* A *supplies a finite stabilization stage, and* H *transports that stabilization along the same classifier. Thus any downstream algebraic consumer may use* $\mathsf{NoetherianModule}^{\uparrow}$ *as a finite submodule-chain certificate without importing quotient-module classification, selected bases, maximal-submodule choice, or a host Noetherian theorem.*

Proof. Begin with the source compatibility rows from Definition 3648.2. They force the chain to be read over the module classifier already present in M, while R and I keep the scalar and ideal comparison surfaces visible.

Apply the stabilization witness A to the displayed submodule chain. The result is a finite stage inside the same module classifier, not an external quotient or selected basis.

If the accepted packet is transported through a classified replacement, use the componentwise transport row H to move the chain entries and the stabilization witness together. The replay row Q preserves only the displayed continuation data, and the provenance row P prevents an ambient Noetherian principle from entering the route.

The transported stage is therefore a reusable finite submodule-chain certificate for later module consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$.

Constructive story: A $\mathsf{NoetherianModule}^{\uparrow}$ packet is generated from a $\mathsf{Module}^{\uparrow}$ source, a scalar $\mathsf{CommRing}^{\uparrow}$ source, a $\mathsf{NoetherianRing}^{\uparrow}$ scalar-side chain context, an $\mathsf{Ideal}^{\uparrow}$ comparison surface, a submodule-chain stabilization witness, a finite-generator witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3648.1, Definition 3648.2, and Definition 3648.3 seed the finite submodule-chain stabilization interface over ModuleUp, NoetherianRingUp, CommRingUp, IdealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-module classification, selected bases, maximal-submodule choice, host Noetherian module theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite submodule-chain stabilization access, finite-generator access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.NoetherianModuleUp.

Chapter 2684

A Concrete Naming Certificate for $\mathsf{LaplaceTransform}^{\uparrow}$

Following classical Laplace transform theory in analysis, the $\mathsf{LaplaceTransform}^{\uparrow}$ packet records only the finite constructive transform interface visible to BEDC consumers. It sits after the coefficient and convergence rows of chapter 1583, the integral carrier of chapter 123, the graph discipline of chapter 130, the real seal of chapter 38, the regular rational readback of chapter 301, the finite observation discipline of chapter 126, and the dyadic tolerance core of chapter 291. The chapter records finite exponential-kernel, integral-window, transform-output, transport, replay, provenance, and local NameCert rows; it does not assert host analytic continuation, a global function-space quotient, selected improper integrals, or ambient real completeness.

Definition 2684.1 (Laplace transform carrier)**.** *A* $\mathsf{LaplaceTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, I, F, W, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the exponential-kernel row, read through* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *data for the finitely inspected values of e^{-st}.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *window row naming the displayed interval, mesh or tagged integration data, and tolerance budget used by the transform read.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *input-function row restricted to the finite source observations consumed by I.*
4. *W is the finite product-read window combining the kernel row K with the input row F before the integral row I is consumed.*
5. *O is the transform-output row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff reached only through K, F, W, I.*
6. *H records componentwise* hsame *transport for the kernel, integral window, input function, product reads, output handoff, provenance, and local name.*
7. *C records displayed* Cont *replay from kernel and function reads through the integral window into the output row.*
8. *P is the* Pkg *provenance over* $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $K, I, F, W, O, H, C, P, N$. It has no coordinate for host analytic continuation, quotient equality of functions, selected improper limits, unbounded integration outside the displayed window, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LaplaceTransformUp.LaplaceTransformTasteGate single carrier alignment`

Theorem 2684.2 (Laplace transform NameCert obligations)**.** *For an accepted* $\mathsf{LaplaceTransform}^{\uparrow}$ *carrier $L = (K, I, F, W, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}$ *surface consists exactly of carrier admission, exponential-kernel admission, integral-window exactness, input-function read compatibility, product-window replay, transform-output handoff, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged along the finite route*

$$K, F \longmapsto W \longmapsto I \longmapsto O$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 3649.1. The kernel row K, input-function row F, product-read window W, integral row I, and output row O are stored coordinates before any downstream consumer reads the transform.

The exponential-kernel obligation is confined to the displayed $\mathsf{RealPowerSeries}^{\uparrow}$ and real-read rows inside K. The input-function obligation is confined to the finite $\mathsf{ContinuousMap}^{\uparrow}$ observations in F, and the product-window obligation records exactly which paired reads enter the integration window.

Integral-window exactness is the row I. It consumes only the paired reads in W and the displayed tolerance and mesh data carried by the integral interface, so the calculation is a finite BHist route rather than an improper-integral assertion.

The rows H and C transport and replay the same route componentwise. The provenance row P and local name N source the packet, while O is downstream of K, F, W, I. Since none of the excluded host objects appears as a carrier coordinate, the listed obligations exhaust the certificate surface. □

Theorem 2684.3 (Laplace transform exponential-kernel window)**.** *Every accepted* $\mathsf{LaplaceTransform}^{\uparrow}$ *read of the exponential factor passes through the finite kernel row K before it can enter the product window W and the integral row I. The kernel read is therefore a displayed* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *window over the inspected s, t values, not a host exponential-function oracle and not an analytic-continuation principle.*

Proof. Project the kernel row K from the carrier of Definition 3649.1. The row records only the finitely inspected real and regular-rational reads needed for the exponential kernel.

The product window W consumes K together with the input-function row F. Thus an integral read cannot reach the kernel through any route except the displayed finite kernel row and its stored transport and replay data.

Finally the integral row I consumes W rather than a host exponential function. The transform output O is downstream of this route, so no consumer can replace the kernel window by analytic continuation or an unrecorded exponential oracle. □

Theorem 2684.4 (Laplace transform real seal nonescape)**.** *The output row O of an accepted* $\mathsf{LaplaceTransform}^{\uparrow}$ *packet exports to* $\mathsf{Real}^{\uparrow}$ *only after the displayed route*

$$K, F \longmapsto W \longmapsto I$$

has been transported and replayed through H and C. The real seal records the finite transform-output handoff already present in the packet, and it does not export host analytic continuation, global function-space equality, selected improper limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The output row O is a carrier coordinate, but Definition 3649.1 places it downstream of the kernel row, input-function row, product-read window, and integral row. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only the output handoff obtained from the displayed finite route.

Componentwise transport through H preserves the same coordinates, and continuation replay through C rebuilds the same dependency route. The package row P and local name N record provenance without adding analytic structure.

The excluded objects are absent from the carrier. There is no host analytic-continuation row, no quotient function-space coordinate, no selected improper-integral limit, no countable-choice selector, and no ambient completeness witness. Hence the real seal is nonescaping. □

Closure status for $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3649.1, Definition 3649.2, Definition 3649.3, and Definition 3649.4 seed the finite exponential-kernel and integral-window transform boundary over $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, global function-space equality, selected improper limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponential-kernel admission, integral-window exactness, product-window replay, transform-output handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.LaplaceTransformUp.

Constructive story: A $\mathsf{LaplaceTransform}^{\uparrow}$ packet is generated from a finite exponential-kernel row, an input-function row, a product-read window, an Integral window, a RegSeqRat and Real output handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2685

A Concrete Naming Certificate for $\mathsf{LipschitzExtension}^{\uparrow}$

Following McShane, Whitney, and Bishop–Bridges constructive metric analysis, the $\mathsf{LipschitzExtension}^{\uparrow}$ packet records the reusable finite certificate by which a displayed Lipschitz map on a displayed subcarrier is read as an extension candidate on an ambient metric carrier. It sits above the map-level surface of chapter 410, the metric carrier of chapter 57, the continuity handoff of chapter 130, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the formula-specific McShane–Whitney certificate of chapter 1941. The chapter isolates the extension interface itself: source inclusion, bound ledger, candidate graph, restriction agreement, continuity handoff, transport, replay, provenance, and local naming.

Definition 2685.1 (Lipschitz extension carrier)**.** *A* $\mathsf{LipschitzExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, A, L, B, F, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the source metric observations available to the extension request.*
2. *A is the displayed subcarrier row, including membership evidence and inclusion routes into X.*
3. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row on A, carrying the finite graph window and displayed Lipschitz constant.*
4. *B is the* $\mathsf{RegSeqRat}^{\uparrow}$ *bound row that records the rational tolerance and constant reads used by the extension certificate.*
5. *F is the candidate extension graph row, exposed only as finite ambient graph observations.*
6. *R is the restriction-agreement row, requiring each candidate read over an admitted source point to replay the source* $\mathsf{LipschitzMap}^{\uparrow}$ *read through A, L, B.*
7. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row for the candidate extension on the displayed ambient window.*
8. *H records componentwise* hsame *transport for the metric, subcarrier, source map, bound, candidate graph, restriction, continuity handoff, provenance, and local name rows.*
9. *C records displayed* Cont *replay from an ambient extension request through $X, A, L, B, F, R, Q, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{McShaneExtension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function equality, quotient function spaces, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LipschitzExtensionUp.LipschitzExtensionTasteGate single carrier alignment`

Theorem 2685.2 (Lipschitz extension NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzExtension}^{\uparrow}$ *carrier* $E = (X, A, L, B, F, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$ *surface consists exactly of carrier admission, subcarrier inclusion, source Lipschitz-map admission, rational bound readback, candidate graph admission, restriction-agreement exactness, continuity handoff, componentwise transport, displayed replay, provenance, and ledger non-escape. The finite route is*

$$X, A \longmapsto L \longmapsto B \longmapsto F \longmapsto R, Q \longmapsto H, C, P, N.$$

No obligation is discharged by a host extension formula or by equality of ambient functions.

Proof. Carrier admission is projection from Definition 3650.1. The rows $X, A, L, B, F, R, Q, H, C, P, N$ are the complete visible inventory of an accepted packet.

The subcarrier and source-map clauses read A and L. A source value can enter the extension certificate only after the membership and inclusion row A has been displayed and the $\mathsf{LipschitzMap}^{\uparrow}$ row L has exposed the finite graph and bound data.

The rational bound and candidate graph clauses read B and F. The bound is a regular-rational ledger, and the candidate graph is a finite ambient observation row, so neither row selects a global function object or an unrecorded real limit.

Restriction exactness is the row R, while continuity handoff is the row Q. The transport and replay rows H and C preserve the same route componentwise, and P, N source and name it. Since no excluded host object occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2685.3 (Lipschitz extension restriction handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{McShaneExtension}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzExtension}^{\uparrow}$ *packet must read the restriction route*

$$A, L, B \longmapsto R$$

before it can treat the candidate graph row F as an extension of the source Lipschitz row. The handoff exports the displayed agreement between source and ambient rows, not a quotient function equality, not a selected representative, and not an ambient completeness theorem.

Proof. Start with a consumer asking whether the candidate extends the source map. The subcarrier row A is read first, because the question is meaningful only for points admitted by the displayed inclusion route.

Next read the source Lipschitz row L and the rational bound row B. These rows supply the finite source graph and tolerance data against which the candidate graph will be checked.

The restriction-agreement row R then compares the candidate graph row F with the source route over the admitted point. The continuity row Q and structural rows H, C, P, N replay and source the same comparison without adding a host function-space coordinate.

Thus a McShane–Whitney certificate may instantiate the interface, but the interface itself exports only the finite restriction handoff displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3650.1, Definition 3650.2, and Definition 3650.3 seed the finite reusable Lipschitz extension surface over displayed metric, Lipschitz-map, continuity, regular-rational, real, and McShane-extension rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subcarrier inclusion, source Lipschitz admission, rational bound readback, candidate graph admission, restriction exactness, continuity handoff, transport, replay, provenance, and non-escape for BEDC.Derived.LipschitzExtensionUp.

Constructive story: A $\mathsf{LipschitzExtension}^{\uparrow}$ packet is generated from an ambient Metric row, a displayed subcarrier row, a source LipschitzMap row, a regular rational bound ledger, a candidate extension graph, a restriction-agreement row, a ContinuousMap handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2686

A Concrete Naming Certificate for $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$

The $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ packet realizes the anchor-subcube trigger theorem of Definition 9462.1519 as finite NameCert data. It records the displayed anchor, boundary-tile span, coverage-rank, trigger-algebra, blocker, transport, replay, provenance, and local naming rows that let a DNA-window consumer read the median closure route without importing a global codon geometry oracle or a host powerset calculation. The packet consumes the boundary-tile carrier of chapter 3393 and the trigger-algebra route of chapter 3478 before exposing the median-closure read.

Definition 2686.1 (Anchor subcube trigger carrier)**.** *An* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, S, R, G, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the anchor row naming the* CUY *boundary anchor and the admitted coordinate window.*
2. *B is the boundary-tile row listing the outside Y-halfspace observations inspected by the packet.*
3. *S is the subcube-span row recording the axis-aligned span generated by the anchor and the projected outside observations.*
4. *R is the coverage-rank row, including the displayed value of the span dimension used by the trigger theorem.*
5. *G is the trigger-algebra row recording the finite median-closure operations allowed by the packet.*
6. *K is the blocker row excluding closure reads outside the generated anchor subcube.*
7. *H records componentwise* hsame *transport for anchor, boundary-tile, span, rank, trigger-algebra, blocker, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a trigger request through $A, B, S, R, G, K, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, anchored in Definition 9462.1519.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host codon powerset, a global median algebra, an unbounded closure engine, or equality of biological interpretations outside the displayed trigger window.

Lean4 ✓ `BEDC.Derived.AnchorSubcubeTriggerUp.AnchorSubcubeTriggerTasteGate single carrier alignment`

Theorem 2686.2 (Anchor subcube trigger NameCert obligations)**.** *For every accepted* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *carrier* $T = (A, B, S, R, G, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$ *surface consists exactly of anchor admission, boundary-tile admission, subcube-span calculation, coverage-rank readback, trigger-algebra replay, blocker non-escape, componentwise transport, displayed continuation replay, provenance, and local naming. The finite route is*

$$A, B \longmapsto S \longmapsto R \longmapsto G, K \longmapsto H, C, P, N.$$

No obligation is discharged by a host median-closure oracle or by an unrecorded codon-space equality.

Proof. Project the accepted carrier. The rows $A, B, S, R, G, K, H, C, P, N$ are the complete packet, so anchor admission and boundary-tile admission are direct finite reads.

The span row S records the axis-aligned subcube generated by the anchor and the outside boundary observations. The rank row R then reads the displayed span dimension, so the trigger threshold is a certificate row rather than a host calculation over all codon subsets.

The trigger-algebra row G replays the finite median operations that generate the subcube, while the blocker row K prevents reads outside the displayed span. Transport through H and replay through C preserve this same route componentwise.

The provenance and local naming rows P, N source the packet in the anchor-subcube trigger theorem. Since no carrier coordinate names a global codon powerset or an ambient closure engine, the listed rows exhaust the local NameCert obligations. □

Theorem 2686.3 (Anchor subcube trigger replay boundary)**.** *Every consumer of an accepted* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *packet that reads a trigger conclusion must first traverse the displayed route*

$$A, B, S, R, G, K.$$

The replay exposes the anchor-relative subcube span and its coverage rank before the trigger is public. It exports no closure claim for boundary tiles outside B*, no radial trigger not carried by* S*, and no codon equality outside the finite rows named by the packet.*

Proof. Begin with a trigger read. The anchor row A and boundary-tile row B fix the finite observation window before any span or rank claim is available.

Read the span and rank rows S and R. These rows are the only source of the generated subcube and its dimension, so a consumer cannot substitute an external radial or global closure calculation.

The trigger-algebra row G then replays the median operations licensed by the span, and the blocker row K records the boundary beyond which the packet refuses closure reads. The structural rows H, C, P, N transport, replay, source, and name the same route without adding further codon geometry. □

Closure status for $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3651.1, Definition 3651.2, and Definition 3651.3 seed the finite anchor, boundary-tile, span, rank, trigger-algebra, blocker, transport, replay, provenance, and NameCert surface for anchor-subcube triggers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host codon powersets, global median algebra, radial trigger claims outside the displayed span, biological equality outside the packet, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked anchor admission, boundary-tile admission, span calculation, rank readback, trigger-algebra replay, blocker non-escape, transport, replay, provenance, and local naming for BEDC.Derived.AnchorSubcubeTrigger

Constructive story: An AnchorSubcubeTrigger$^{\uparrow}$ packet is generated from a displayed anchor row, boundary-tile row, anchor-relative span row, coverage-rank row, trigger-algebra row, blocker row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 2687

A Concrete Naming Certificate for $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$

Following Fourier's heat-equation analysis and the Courant–Friedrichs–Lewy finite-difference tradition, the $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ packet records only the finite constructive stencil interface visible to BEDC consumers. It sits after the partial-differential-equation row of chapter 194, the derivative row of chapter 122, the integral row of chapter 123, the transform handoff of chapter 3649, the real seal of chapter 38, the regular rational readback of chapter 301, the stream-window discipline of chapter 126, and the dyadic tolerance core of chapter 291. The chapter records heat-equation finite-stencil data as a local NameCert surface; it does not assert analytic solution existence, global PDE well-posedness, host function spaces, measure theory, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2687.1 (Heat-equation finite-stencil carrier)**.** *A* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, T, X, B, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite source-function window row, read through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observations of the finitely inspected source values.*
2. *T is the time-step row containing the displayed dyadic time width, time-index schedule, and local tolerance budget.*
3. *X is the spatial stencil row containing the displayed dyadic mesh width, nearest-neighbour or finite-radius stencil weights, and inspected spatial index schedule.*
4. *B is the boundary row containing only the boundary values and boundary classifier reads requested by the finite stencil.*
5. *Q is the finite replay row that records how source, time-step, spatial-stencil, and boundary reads produce the next displayed heat row.*
6. *R is the real-seal row reached through* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks for the finitely inspected stencil values.*
7. *H records componentwise* hsame *transport for F, T, X, B, Q, R, P, N and for the local certificate rows.*

8. *C records displayed* Cont *replay from the source-function windows through time-step, spatial-stencil, boundary, and real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, T, X, B, Q, R, H, C, P, N$. It has no coordinate for analytic solution existence, global parabolic well-posedness, host function-space equality, measure-theory integration, infinite mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeatEquationFiniteStencilUp.HeatEquationFiniteStencilTasteGate single carrier alignment`

Theorem 2687.2 (Heat-equation finite-stencil NameCert obligations)**.** *For an accepted* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *carrier $E = (F, T, X, B, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window admission, time-step exactness, spatial-stencil exactness, boundary-row exactness, finite replay, real-seal handoff, componentwise transport, displayed continuation replay, provenance, and local naming. These obligations are discharged along the finite route*

$$F, T, X, B \longmapsto Q \longmapsto R$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 3652.1. The projection exposes the source-function windows, time-step row, spatial-stencil row, boundary row, finite replay row, real-seal row, transport rows, replay rows, provenance row, and local name.

The source-window, time-step, spatial-stencil, and boundary obligations are the rows F, T, X, B. They record only finite stream positions, regular rational readbacks, dyadic mesh and time budgets, and boundary reads already displayed in the packet.

The finite replay obligation is Q. It consumes F, T, X, B before the real seal R is read, so the heat row is a displayed stencil calculation rather than a host PDE solution.

Finally H transports the same rows componentwise, C replays the same route, and P, N source and name the packet. Since the carrier contains no row for analytic existence, global well-posedness, host function spaces, measure theory, infinite mesh limits, or ambient real completeness, the listed rows exhaust the local certificate surface. □

Theorem 2687.3 (Heat-equation finite-stencil Laplace-PDE handoff)**.** *Every accepted* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *handoff to a* $\mathsf{Pde}^{\uparrow}$ *or* $\mathsf{LaplaceTransform}^{\uparrow}$ *consumer passes first through the displayed finite stencil route*

$$F, T, X, B \longmapsto Q \longmapsto R.$$

The $\mathsf{Pde}^{\uparrow}$ *read receives only the derivative-relation and boundary rows visible through the stencil packet, while the* $\mathsf{LaplaceTransform}^{\uparrow}$ *read receives only the finite transform-window handoff after R has been displayed. Neither consumer obtains analytic solution existence, global heat-kernel semantics, selected improper limits, host function-space equality, measure theory, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $E = (F, T, X, B, Q, R, H, C, P, N)$. The PDE-facing data are not independent host equations; they are the derivative-relation, time-step, spatial-stencil, and boundary reads already present in T, X, B, Q.

For a $\mathsf{Pde}^{\uparrow}$ consumer, replay C from the source windows through the time and spatial rows into Q. The consumer can inspect only the finite relation and boundary rows that the replay exposes, transported by H and sourced by P, N.

For a $\mathsf{LaplaceTransform}^{\uparrow}$ consumer, the transform-window handoff is available only after the real-seal row R has been reached through the same route. The read therefore consumes finite stream, regular rational, and dyadic rows before it reaches any transform output.

The excluded analytic objects are not fields of the carrier. The handoff is therefore a finite NameCert route from stencil data to PDE and Laplace interfaces, not a claim of global heat-equation solvability or host analytic semantics. □

Falsifiable prediction: An accepted $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ consumer that reaches a $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed $\mathsf{StreamName}^{\uparrow}$ source windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ mesh and tolerance rows, finite stencil replay, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: pde, derivative, integral, laplacetransform, real, regseqrat, streamname, dyadicratcore: no listed sibling supplies the whole heat finite-stencil packet $E = (F, T, X, B, Q, R, H, C, P, N)$, because each sibling contributes only a PDE relation, derivative read, integral window, transform handoff, real seal, regular rational readback, stream schedule, or dyadic tolerance ledger while this chapter binds them into one finite replay and local naming route.

Closure status for $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$.

Constructive story: A $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ packet is generated from finite source-function windows, a displayed time-step row, a displayed spatial-stencil row, boundary rows, finite replay rows, a Real-seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3652.1, Definition 3652.2, and Definition 3652.3 seed the finite constructive heat-equation stencil boundary over $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim analytic solution existence, global PDE well-posedness, host function spaces, measure theory, infinite mesh convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and Laplace-PDE handoff.

Chapter 2688

A Concrete Naming Certificate for $\mathsf{HausdorffDistance}^{\uparrow}$

Following the Hausdorff–Kuratowski compact-set metric reading, the $\mathsf{HausdorffDistance}^{\uparrow}$ packet records the scalar finite-distance certificate between two displayed nonempty compact subsets of one compact metric source. It sits next to the host hyperspace route of chapter 2086, consumes the compact-metric finite-net surface of chapter 129, and seals the resulting value through the real-name interface of chapter 38. The packet names the distance value as a finite BEDC carrier rather than treating $\mathsf{HausdorffMetric}^{\uparrow}$ itself as the only available consumer object.

Definition 2688.1 (Hausdorff distance finite carrier)**.** *A* $\mathsf{HausdorffDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, H, N_X, N_Y, D, E, R, T, C, P, L)$$

with the following displayed rows:

1. *A and B are the two nonempty compact-subset rows over the same compact metric source.*
2. *H is the* $\mathsf{HausdorffMetric}^{\uparrow}$ *ledger row that records the finite directed comparisons already admitted by the neighbouring Hausdorff-metric certificate.*
3. *N_X and N_Y are finite net rows for the two compact-subset names, read from the* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *sources.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the displayed mesh, directed bound, and symmetric slack.*
5. *E is the symmetric bound-exactness row, recording that the two directed finite-net estimates determine the same scalar tolerance window.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal reached only after the ledger, nets, and dyadic tolerance rows have been displayed.*
7. *T records componentwise* hsame *transport for $A, B, H, N_X, N_Y, D, E, R, C, P, L$.*
8. *C records displayed* Cont *replay of the route from compact-subset rows through finite nets, symmetric bounds, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, H, N_X, N_Y, D, E, R, T, C, P, L$. It has no coordinate for a host hyperspace point, quotient subset equality, selected nearest point, open-cover compactness, or ambient supremum formation.

Lean4 ✓ `BEDC.Derived.HausdorffDistanceUp.HausdorffDistanceTasteGate single carrier alignment`

Theorem 2688.2 (Hausdorff distance NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffDistance}^{\uparrow}$ *carrier*

$$Q = (A, B, H, N_X, N_Y, D, E, R, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the two compact-subset rows, HausdorffMetric ledger, finite nets, dyadic tolerance row, symmetric bound row, Real seal, transport, replay, provenance, and local naming row;*
2. *symmetric bound exactness, requiring the directed finite-net estimates in H and E to share the dyadic tolerance budget recorded by D;*
3. *finite-net witness stability, requiring N_X and N_Y to remain attached to the same* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *sources under componentwise transport;*
4. *Real seal nonescape, requiring every scalar distance read to pass through R after H, N_X, N_Y, D, E have been displayed;*
5. *ledger nonescape, requiring C, P, L to replay, source, and name only the displayed finite route and to export no host hyperspace point, quotient subset equality, selected nearest point, open-cover compactness, or ambient supremum row.*

No scalar Hausdorff distance obligation is stored outside these rows.

Theorem 2688.3 (Hausdorff distance scalar handoff)**.** *Every Gromov–Hausdorff-facing, Vietoris-facing, or compact-subset consumer of an accepted* $\mathsf{HausdorffDistance}^{\uparrow}$ *packet reads the scalar value only through the finite route*

$$A, B, H, N_X, N_Y, D, E \longmapsto R \longmapsto T, C, P, L.$$

Thus the consumer receives a $\mathsf{Real}^{\uparrow}$ *distance seal for the two displayed compact-subset rows without receiving a host hyperspace metric object, quotient subset equality, selected nearest point, open-cover compactness theorem, or ambient supremum principle.*

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffDistance}^{\uparrow}$ packet is generated from two nonempty compact-subset rows, a HausdorffMetricUp ledger, finite CompactMetricUp and TotallyBoundedUp net rows, a DyadicRatCore tolerance row, symmetric bound exactness, a RealUp distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$, seed)

Scope closed: Definition 3653.1, Definition 3653.2, and Definition 3653.3 seed the finite scalar Hausdorff distance certificate consumed by compact-metric downstream routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperspaces, quotient subset equality, selected nearest points, open-cover compactness, ambient supremum formation, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, symmetric bound exactness, finite-net witness stability, RealUp seal nonescape, and ledger nonescape for BEDC.Derived.HausdorffDistanceUp.

Chapter 2689

A Concrete Naming Certificate for BoundedVariationRealFunction$^{\uparrow}$

Following Jordan's bounded variation theorem and Bishop–Bridges constructive analysis, the BoundedVariationRealFunction$^{\uparrow}$ packet records a real-valued function on a displayed real interval through finite partition windows and variation budgets. It sits between the real-valued sequence interface of chapter 3641, the finite partition carrier of chapter 1863, the stream-name observation route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the finite BEDC surface needed by later regulated-integration and Jordan-decomposition consumers without selecting a total variation supremum or importing ambient real completeness.

Definition 2689.1 (BoundedVariationRealFunction carrier)**.** *A* BoundedVariationRealFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, F, W, R, D, B, J, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval and partition window row supplied by* DyadicPartition$^{\uparrow}$*, listing the finite dyadic subdivision on which variation is inspected.*
2. *F is the displayed real-valued function row, exposed only through* RealSequenceSpace$^{\uparrow}$ *coordinate requests over the partition endpoints.*
3. *W is the* StreamName$^{\uparrow}$ *observation row scheduling the finite endpoint windows and adjacent-pair probes.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the values of F at the endpoints named by W.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *variation-budget ledger, recording absolute-difference budgets, partial variation sums, and the displayed bound attached to the current partition window.*
6. *B is the bounded-variation acceptance row, asserting that the inspected partition window is admitted only when its dyadic variation sum is witnessed below the displayed budget in D.*
7. *J is the Jordan-route row, naming the finite positive and negative variation ledgers available to a later Jordan-decomposition consumer without selecting a global decomposition.*
8. *H records componentwise* hsame *transport for the partition, function, observation, readback, variation-budget, bounded-variation, Jordan-route, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a bounded-variation request through I, F, W, R, D, B, J.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedVariationRealFunctionUp.TasteGate.BoundedVariationRealFunctionTasteGate single carrier alignment`

Theorem 2689.2 (BoundedVariationRealFunction NameCert obligations)**.** *For every accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier* $V = (I, F, W, R, D, B, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFun}}$ *surface consists exactly of finite* BHist *rows for carrier admission, partition-window exposure, real-value readback, variation-budget exactness, bounded-variation acceptance, Jordan-route availability, componentwise transport, replay, provenance, and local naming:*

1. *carrier admission requires the displayed* $\mathsf{DyadicPartition}^{\uparrow}$ *row,* $\mathsf{RealSequenceSpace}^{\uparrow}$ *function row,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets, bounded-variation row, Jordan-route row, transport, replay, provenance, and local name;*

2. *partition exposure requires every variation request to pass through the finite dyadic row I before any endpoint value is read;*

3. *value readback requires the endpoint values of F to be inspected only through W and R, and then sealed as* $\mathsf{Real}^{\uparrow}$ *data through the provenance row P;*

4. *variation-budget exactness requires the partial sums and displayed bound in D to mediate every acceptance read by B;*

5. *Jordan-route availability requires J to expose only finite positive and negative variation ledgers downstream of the same I, F, W, R, D, B route;*

6. *transport and replay stability require H, C, P, N to preserve and name the same finite surface.*

No obligation supplies total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3654.1. The packet exposes a dyadic partition window, a displayed real-valued function row, finite observation windows, regular rational readbacks, a dyadic variation ledger, a bounded-variation acceptance row, and a Jordan-route row before any structural replay is available.

The local obligations are therefore row-local. A variation request first enters the $\mathsf{DyadicPartition}^{\uparrow}$ row I; endpoint values of F are then read only through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows W and R; and every absolute-difference or partial-sum comparison is mediated by the $\mathsf{DyadicRatCore}^{\uparrow}$ row D.

The bounded-variation and Jordan-facing rows are downstream of the same route. The acceptance row B can certify only the displayed partition window and budget, while J can expose only finite positive and negative variation ledgers built from that accepted data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host supremum, quotient function equality, selected decomposition, measure integration, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2689.3 (BoundedVariationRealFunction Jordan route)**.** *Every Jordan-facing consumer of an accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier* $V = (I, F, W, R, D, B, J, H, C, P, N)$ *reads the ordered finite route*

$$\mathsf{DyadicPartition}^{\uparrow} \longrightarrow \mathsf{RealSequenceSpace}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through $I, F, W, R, D, B, J, H, C, P, N$*. The consumer may use the finite positive and negative variation ledgers exposed by* J*, but it cannot extract total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Start with a Jordan-facing read of an accepted carrier. By Definition 3654.2, the read first passes through the finite $\mathsf{DyadicPartition}^{\uparrow}$ row and the displayed $\mathsf{RealSequenceSpace}^{\uparrow}$ function row, so every inspected endpoint value is tied to the same partition window.

The observation and readback rows then fix the value interface. The $\mathsf{StreamName}^{\uparrow}$ row schedules the endpoint probes, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readbacks, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the variation-budget comparisons used by the bounded-variation acceptance row.

The Jordan route J is downstream of that accepted data. It may expose finite positive and negative variation ledgers for the displayed window, and the structural rows H, C, P, N transport, replay, source, and name the route.

No additional carrier coordinate remains for a host supremum of all partitions, quotient equality of real-valued functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every Jordan-facing read factors through the displayed finite route. □

Theorem 2689.4 (BoundedVariationRealFunction partition-refinement induction)**.** *Let* $V = (I, F, W, R, D, B, J, H$ *be an accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier. Suppose that* I *displays a finite dyadic partition window and that a finite refinement spine is obtained by repeatedly inserting one dyadic endpoint into an already displayed adjacent interval. Then primitive induction over that spine preserves the variation-budget row: at every stage each inserted endpoint is read through the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *route* W, R*, the two adjacent absolute-difference budgets replacing the old interval budget are recorded in the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*, and the refined partial variation ledger is accepted by the same bounded-variation row* B*.*

Proof. Use induction on the finite refinement spine displayed over I. In the base case the spine has no inserted endpoint, so the current ledger is exactly the partition row already accepted by B through the budget comparisons recorded in D.

For the successor step, assume that the current displayed ledger is accepted by B and insert one dyadic endpoint inside a single adjacent interval. The endpoint observation is scheduled by the $\mathsf{StreamName}^{\uparrow}$ row W and read back by the $\mathsf{RegSeqRat}^{\uparrow}$ row R, so no value of F is inspected outside the carrier route.

The $\mathsf{DyadicRatCore}^{\uparrow}$ row D then replaces the old interval budget by the two adjacent absolute-difference budgets and updates the displayed partial variation sum. Since the induction hypothesis already identifies every unchanged interval with its accepted budget row, this replacement is the only local comparison required for the refined window.

Finally H, C, P, N transport, replay, source, and name the refined packet without changing the carrier rows. The acceptance row B therefore reads the refined ledger through the same finite packet, and the induction preserves bounded-variation acceptance along the whole refinement spine. □

Theorem 2689.5 (BoundedVariationRealFunction ledger handoff)**.** *Every accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier whose partition window is read as a* $\mathsf{BoundedVariation}^{\uparrow}$ *packet must expose*

$$I, F, W, R, D, B, J, H, C, P, N$$

first. The finite partition, endpoint, and dyadic-variation ledger recorded by I, W, R, D, B *is precisely the input consumed by chapter 398; the Jordan row* J *only routes the same accepted ledger to later positive and negative variation consumers.*

Proof. Start with the NameCert obligations of Definition 3654.2. They require carrier admission, partition-window exposure, value readback, variation-budget exactness, bounded-variation acceptance, and Jordan-route availability to be read through the same tuple $I, F, W, R, D, B, J, H, C, P, N$.

The Jordan route theorem Definition 3654.3 keeps J downstream of the accepted partition and variation-budget rows. Thus the positive and negative variation ledgers do not replace the bounded-variation input; they are read only after I, F, W, R, D, B have been displayed.

Finally apply the refinement induction of Definition 3654.4. It shows that endpoint insertions preserve the dyadic variation ledger through the same W, R, D, B route, so the reusable $\mathsf{BoundedVariation}^{\uparrow}$ chapter consumes exactly this finite ledger and no host supremum, quotient function equality, selected Jordan decomposition, or ambient completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ packet is generated from finite $\mathsf{DyadicPartition}^{\uparrow}$ windows, a displayed $\mathsf{RealSequenceSpace}^{\uparrow}$ function row, $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ variation-budget ledger, a bounded-variation acceptance row, a finite Jordan-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3654.1, Definition 3654.2, Definition 3654.3, Definition 3654.4, and Definition 3654.5 seed the finite bounded-variation real-function surface over partition windows, real-valued sequence rows, finite observations, regular readbacks, dyadic variation budgets, refinement ledgers, bounded-variation handoff, and local Jordan-route ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-window exposure, real-value readback, variation-budget exactness, bounded-variation acceptance, finite Jordan-route availability, componentwise transport, replay, provenance, and local naming for the same displayed packet.

Chapter 2690

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ packet records the finite modulus evidence that turns regular Cauchy rational windows into located real-name reads. It sits between the dyadic arithmetic substrate of chapter 291, the stream observation surface of chapter 126, the regular rational sequence packet of chapter 301, the located comparison rows of chapter 1730, and the sealed real-name handoff of chapter 38.

2690.1 Located modulus carrier

Definition 2690.1 (Regular Cauchy located modulus carrier)**.** *A* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *carrier is a finite packet* $(D, S, Q, L, U, M, H, C, P, N)$ *with the following displayed rows:*

1. D*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger whose positive rows name the dyadic error budgets used by the modulus;*
2. S*, a* $\mathsf{StreamName}^{\uparrow}$ *observation window for the rational name under inspection;*
3. Q*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular Cauchy rational values visible on* S*;*
4. L *and* U*, lower and upper located comparison rows tied to the same finite window and the same dyadic precision ledger;*
5. M*, a finite modulus row assigning to each displayed dyadic precision a window depth after which the* Q*-values stay inside the* L, U *located bracket;*
6. H*, the componentwise* hsame *transport rows for the dyadic ledger, stream window, regular readback, located comparison rows, and modulus row;*
7. C*, the displayed* Cont *replay routes preserving the finite window and modulus reads under transport;*
8. P*, the* Pkg *provenance rows for the dyadic ledger, stream window, regular readback, located comparison evidence, and modulus ledger;*
9. N*, the local* $\mathsf{NameCert}$ *row binding these nine rows as one located modulus certificate.*

The carrier contains no quotient equality of Cauchy names, no selected limit point, no host real order predicate, and no ambient completeness principle.

Lean4 ✓ BEDC.Derived.RegularCauchyLocatedModulusUp.RegularCauchyLocatedModulusTasteGate single carrier alignment

Theorem 2690.2 (Regular Cauchy located modulus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedM}}$ *surface for Definition 3655.1 consists of six finite obligations:*

1. *carrier admission: the rows* $(D, S, Q, L, U, M, H, C, P, N)$ *occur in the same finite* BHist *packet;*
2. *window alignment:* S *is the only observation window read by* Q*, by the located comparison rows* L, U*, and by the modulus row* M*;*
3. *dyadic precision exactness: every modulus query in* M *consumes a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget displayed in* D*;*
4. *regular readback confinement: all rational values inspected by* M *come through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and not through an external sequence selector;*
5. *located bracket handoff: the lower and upper comparison rows* L, U *are the only located evidence exported toward* $\mathsf{LocatedRealComparison}^{\uparrow}$*;*
6. *transport stability:* hsame, Cont, Pkg*, and* NameCert *move the dyadic ledger, window, readback, located brackets, and modulus row componentwise.*

These obligations certify a finite located-modulus route for regular Cauchy names. They do not certify quotient equality of names, chosen limits, total order, or real completeness.

Proof. Project the carrier of Definition 3655.1. The projection exposes the dyadic precision ledger D, the stream window S, the regular readback Q, the located comparison rows L, U, and the modulus row M.

Window alignment and dyadic precision exactness are read directly from those rows. Since M has no coordinate except the displayed D, S, Q, L, U data, each precision query is evaluated against the same finite observation window and the same regular Cauchy readback.

The located bracket handoff is componentwise. The L, U rows are already the displayed $\mathsf{LocatedRealComparison}^{\uparrow}$ evidence, and the transport rows H, C replay them together with D, S, Q, M.

Finally P and N record the provenance and local certificate for exactly these rows. No extra coordinate can supply quotient stream equality, a selected limit, a host real order predicate, or ambient completeness. □

Theorem 2690.3 (Regular Cauchy located modulus Real-seal handoff)**.** *Every accepted* RegularCauchyLocatedMod *packet exports toward* $\mathsf{Real}^{\uparrow}$ *only the sealed read obtained from its* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* S*, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger* D*, located comparison rows* L, U*, and modulus row* M*. The handoff preserves the same* $D, S, Q, L, U, M, H, C, P, N$ *carrier rows and supplies no quotient equality of Cauchy names, selected limit point, trichotomy, host real order predicate, or ambient completeness principle.*

Proof. Start with an accepted carrier and read the regular Cauchy name through Q on the displayed window S. The dyadic budgets used to test locatedness are the positive rows of D, and the lower and upper comparison evidence is precisely L, U.

The modulus row M turns each displayed precision into a finite window-depth claim over those same rows. Thus the Real-facing read has the data needed for a sealed $\mathsf{Real}^{\uparrow}$ handoff without selecting a representative limit.

Transport is confined to H and replayed by C. These rows preserve the dyadic ledger, stream window, regular readback, located brackets, and modulus ledger componentwise, while P, N record provenance and local naming.

The carrier has no quotient, trichotomy, host order, selected limit, or completeness coordinate. Hence the Real-seal consumer receives exactly the finite located-modulus evidence stated above. □

Theorem 2690.4 (Regular Cauchy located modulus finite-window nonescape)**.** *Every* $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *packet must pass through the displayed dyadic ledger* D, $\mathsf{StreamName}^{\uparrow}$ *window* S, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q, *lower and upper located comparison rows* L, U, *modulus row* M, *transport rows* H, *replay rows* C, *provenance rows* P, *and local* $\mathsf{NameCert}$ *row* N. *A strengthened finite-window read can refine these rows only by reading the same carrier coordinates; it cannot bypass them by using quotient Cauchy equality, selected limits, trichotomy, host order, or ambient completeness.*

Proof. Project the carrier of Definition 3655.1. The projection exposes exactly $D, S, Q, L, U, M, H, C, P, N$ as the finite rows available to a consumer.

Apply Definition 3655.2. Window alignment, dyadic precision exactness, regular readback confinement, located bracket handoff, and transport stability force each accepted read through the displayed ledger, window, readback, brackets, modulus, transport, replay, provenance, and local naming rows.

Apply Definition 3655.3 for the Real-facing case. The same $D, S, Q, L, U, M, H, C, P, N$ route is preserved at the seal, so a strengthened finite-window read remains a refinement of the displayed packet route.

The refused objects are absent carrier coordinates. Therefore quotient Cauchy equality, selected limits, trichotomy, host order, and ambient completeness cannot be used to bypass the finite-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ packet is generated from a dyadic precision ledger, a StreamName observation window, a RegSeqRat readback, located comparison brackets, a finite modulus row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3655.1, Definition 3655.2, and Definition 3655.3 name the seed carrier, finite NameCert obligation surface, and Real-seal handoff for located moduli of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy names, selected limits, trichotomy, host real order, ambient completeness, or a standard bridge.

Upgrade path: To reach obligationClosure, the carrier, finite modulus obligations, and Real-seal handoff must be checked as BEDC packet obligations.

Chapter 2691

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ packet records the L10 completion reflector as one finite NameCert surface. It reads the reflector rows of chapter 1585, the functor rows of chapter 423, the extension-and-uniqueness rows of chapter 397, and the reflective-inclusion rows of chapter 1793. The chapter names the reflector boundary consumed by completion arguments; it does not assert quotient completions, choose limit representatives, identify host completions, import countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2691.1 (CauchyCompletionUniversalReflector carrier)**.** *A* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, F, E, J, W, G, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and name-data row, including the displayed metric classifier, finite source observations,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *reads, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger available before completion is consumed.*
2. *R is the reflector row read through* $\mathsf{CauchyCompletionReflector}^{\uparrow}$*, naming the source-to-completion admission and completed-object surface without exposing a quotient carrier or selected host limit.*
3. *F is the functor row read through* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$*, recording the finite action on displayed completion-facing source maps and continuation schedules.*
4. *E is the universal-extension row read through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*, carrying extension existence, extension uniqueness, and finite-window consumer factorization after R and F have been displayed.*
5. *J is the reflective-inclusion row read through* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$*, exposing completed objects only through the public reflective boundary.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the source and completed-name reads.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row used by the source and completion-facing sequences.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row used by S, W, G, R, F, E, J.*

9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *name row reached only after the stream, regular-rational, dyadic, reflector, functor, universal-extension, and reflective-inclusion rows have been displayed.*

10. *H records componentwise* hsame *transport for the source metric/name data, reflector row, functor row, universal-extension row, reflective-inclusion row, StreamName row, RegSeqRat row, DyadicRatCore row, Real row, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from source observations through $W, G, D, R, F, E, J, Q, H$.*

12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$ *row.*

The carrier exposes only this finite reflector surface. It has no coordinate for a quotient completion, chosen limit, host-level completion equality, countable-choice schedule, ambient completeness theorem, or full completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.CauchyCompletionUniversalReflectorUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorTasteGate`
`single carrier alignment`

Theorem 2691.2 (CauchyCompletionUniversalReflector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier $U = (S, R, F, E, J, W, G, D, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, reflector, functor, universal-extension, reflective-inclusion, stream-window, regular-rational, dyadic-tolerance, real-name, transport, replay, provenance, and local naming rows;*

2. *reflector admission, requiring each completion-facing object read to pass through R as in chapter 1585;*

3. *functorial action, requiring each source-map continuation to pass through F as in chapter 423;*

4. *universal-extension factorization, requiring each finite-window consumer to pass through E as in chapter 397;*

5. *reflective-inclusion exposure, requiring completed-object access to pass through J as in chapter 1793;*

6. *local StreamName, RegSeqRat, DyadicRatCore, and Real discipline, requiring W, G, D, Q to be read as the only name and tolerance rows;*

7. *ledger nonescape, requiring P and N to export no quotient completion, selected limit, host completion equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3656.1. The packet displays the source row, reflector row, functor row, universal-extension row, reflective-inclusion row, local StreamName-RegSeqRat-DyadicRatCore-Real rows, transport, replay, provenance, and local name.

The reflector and functor obligations are the first completion-facing projections. The row R is read through chapter 1585, and the row F is read through chapter 423; no consumer reaches completion data before those rows are visible.

The universal-extension and reflective-inclusion obligations follow the same displayed route. The row E cites chapter 397, while J cites chapter 1793; both are projected from the same finite packet.

Finally W, G, D, Q, H, C, P, N transport, replay, package, and name only the displayed rows. Since quotient completions, selected limits, host completion equality, countable choice, ambient completeness, and full $\mathsf{Real}^{\uparrow}$ completeness are absent as coordinates, no NameCert obligation can export them. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorNamecertOb`

Theorem 2691.3 (CauchyCompletionUniversalReflector extension factorization)**.** *Let* $U = (S, R, F, E, J, W, G, D,$ *be an accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier. Every accepted completion-extension read factors through the finite route*

$$S, W, G, D \longmapsto R \longmapsto F \longmapsto E \longmapsto J \longmapsto Q \longmapsto H, C, P, N.$$

Thus a finite-window source consumer can reach an extension row only after the source metric/name data, reflector row, functor row, universal-property row, reflective-inclusion row, and local StreamName-RegSeqRat-DyadicRatCore-Real rows have all been projected from the same carrier.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorCarrier extension factorization`

Proof. Begin with an accepted completion-extension read and project the carrier of Definition 3656.1. The source metric/name data and the local W, G, D rows supply the only finite observations and tolerance accounting available to the read.

Apply the reflector and functor projections. The reflector row R is the public entry into completion-facing data, and the functor row F is the only displayed action row for source-map continuations.

Apply the universal-property projection from E. By chapter 397, the extension and uniqueness read is downstream of the displayed completion rows rather than an independent host function.

The reflective-inclusion row J and real-name row Q close the route. The structural rows H, C, P, N preserve exactly the same route, so the extension factorization has no coordinate from which to read quotient completions, selected limits, host completion equality, choice, or ambient completeness. □

Theorem 2691.4 (CauchyCompletionUniversalReflector nonescape)**.** *Every public L10 completion read exported by an accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *packet is exhausted by the rows* $S, R, F, E, J, W, G, D, Q, H, C, P, N$*. In particular, the read cannot replace the reflector by* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *alone, the functor by* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *alone, the extension by* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *alone, the public completed-object boundary by* CauchyCompletionReflectiveSubcateg *alone, or the local name route by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *alone.*

Proof. Project P first. It names the available sibling packets: $\mathsf{CauchyCompletionReflector}^{\uparrow}$, CauchyCompletionFunctc $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 3656.2. Each sibling is assigned to one row of the carrier, so a consumer that keeps only one sibling loses the displayed relation between source data, reflector entry, functor action, universal extension, reflective inclusion, and local name data.

Apply Definition 3656.3. The extension route is finite and ordered, and the structural rows H, C, P, N preserve that order. Therefore no public L10 completion read can leave the packet surface to obtain a quotient completion, selected limit, host equality, choice schedule, ambient completeness theorem, or full $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ packet is generated from source metric/name data, reflector rows, functor rows, universal-extension rows, reflective-inclusion rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Real name rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3656.1, Definition 3656.2, Definition 3656.3, and Definition 3656.4 seed the finite universal-reflector boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host completion equality, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector admission, functorial action, universal-extension factorization, reflective-inclusion exposure, local name-route discipline, and ledger nonescape for BEDC.Derived.CauchyCompletionUn

Chapter 2692

A Concrete Naming Certificate for $\mathsf{TriggerHypergraphReliability}^{\uparrow}$

The $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ packet realizes the trigger-hypergraph reliability surface of Definition 9462.1556 as finite BEDC NameCert data. It sits over the six-boundary-tile coverage route of chapter 3470, the window-coverage trigger algebra of chapter 3478, the cellular window trace of chapter 3481, the dyadic tolerance ladder of chapter 3525, and the trigger-hypergraph reliability theorem of Definition 9462.1564. The packet names accepted reliability reads as finite trigger-edge and budget rows; it is not a stochastic oracle, polynomial-ring equality, quotient codon semantics, infinite trigger search, or hidden probability model.

Definition 2692.1 (TriggerHypergraphReliability carrier)**.** *A* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, E, R, L, W, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the six-boundary-tile coverage row naming the finite boundary positions whose trigger behavior is read.*
2. *E is the finite trigger-edge list row, recording the accepted hyperedges and their coordinate supports.*
3. *R is the reliability-budget row, carrying the finite acceptance budget against which trigger-edge reads are evaluated.*
4. *L is the local tag ledger for present, absent, covered, uncovered, triggering, nontriggering, accepted, and rejected edge reads.*
5. *W is the cellular window-trace row assigning trigger-edge reads to the visible window route.*
6. *D is the dyadic-tolerance row used to compare reliability budgets without importing real-valued probability or host polynomial equality.*
7. *A is the algebra transport row through the window-coverage trigger algebra and the trigger-hypergraph source vision.*
8. *H records componentwise* hsame *transport for boundary tiles, trigger-edge lists, reliability budgets, local tags, window traces, dyadic tolerances, algebra transport, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from each accepted trigger-edge read through B, E, R, L, W, D, A, H.*

10. *P is the* Pkg *provenance over* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$*, the six-boundary-tile coverage route, the window-coverage trigger algebra, the cellular window trace, the dyadic tolerance ladder, the trigger-hypergraph source vision,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, E, R, L, W, D, A, H, C, P, N$. It has no coordinate for host probability spaces, ambient polynomial equality, quotient codon equality, infinite trigger search, stochastic evolutionary dynamics, or a hidden reliability oracle.

Lean4 (statement) `BEDC.Derived.TriggerHypergraphReliabilityUp`

Theorem 2692.2 (TriggerHypergraphReliability NameCert obligations)**.** *For every accepted* TriggerHypergraphRel *carrier $T = (B, E, R, L, W, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$ *surface consists exactly of boundary-coverage admission, trigger-edge list exposure, reliability-budget exposure, finite tag accounting, window-trace assignment, dyadic-tolerance comparison, algebra transport, componentwise transport, displayed replay, provenance, and local naming.*

The reliability row R is accepted only when its read is paired with one finite trigger-edge list E and one finite budget row over the displayed B, L, W, D, A route. Any consumer that reads reliability without those rows, or replaces them by host probability spaces, ambient polynomial equality, quotient codon semantics, stochastic dynamics, infinite search, or a hidden reliability oracle, leaves the accepted packet surface.

Proof. Project the carrier of Definition 3657.1. The rows B, E, R, L, W, D, A expose the boundary coverage, finite edge list, reliability budget, finite tag ledger, window trace, dyadic tolerance row, and algebra transport.

The reliability-budget row is finite. It is read only together with the trigger-edge list E, while L, W, D record finite tags, cellular placement, and dyadic comparison data before any reliability decision is accepted.

Finally H, C, P, N preserve the same displayed route. Since no host probability space, polynomial equality, quotient codon equality, stochastic model, infinite search, or hidden reliability oracle is a carrier coordinate, the local NameCert obligations are exactly the listed finite rows. □

Theorem 2692.3 (TriggerHypergraphReliability factorization)**.** *Every accepted reliability read for* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *factors through one finite trigger-edge list and one reliability-budget row:*

$$B \longmapsto E \longmapsto R \longmapsto L, W, D, A \longmapsto H, C, P, N.$$

Thus accepted reliability is a finite NameCert route through boundary coverage, trigger-edge exposure, budget accounting, window tracing, dyadic tolerance comparison, algebra transport, replay, provenance, and local naming.

Proof. Start from an accepted reliability read and project the carrier rows. The boundary row B supplies the only visible coverage data, and the trigger-edge list E supplies the only finite hypergraph edge surface.

Read the budget row R next. The row is paired with E, so reliability is not evaluated against an ambient probability space or an unbounded family of triggers.

The rows L, W, D, A then assign finite tags, cellular window placement, dyadic budget comparison, and algebra transport. The structural rows H, C, P, N preserve exactly this route, so the factorization has no coordinate outside the finite packet. □

Lean4 ✓ `BEDC.Derived.TriggerHypergraphReliabilityUp.TriggerHypergraphReliabilityFactorization`

Theorem 2692.4 (TriggerHypergraphReliability nonescape)**.** *Every public reliability read exported by an accepted* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *packet is exhausted by* $B, E, R, L, W, D, A, H, C, P, N$*. In particular, a consumer cannot replace the trigger-edge list by the six-boundary-tile coverage route alone, the reliability budget by the cellular window trace alone, the dyadic comparison by a host real-probability model, or the algebra transport by polynomial-ring equality.*

Proof. Project the provenance row P. It names the six-boundary-tile coverage route, the window-coverage trigger algebra, the cellular window trace, the dyadic tolerance ladder, and the source trigger-hypergraph vision.

Apply Definition 3657.2. Each dependency is assigned to one displayed carrier row, so no sibling route alone contains the trigger-edge list, reliability budget, tag ledger, window trace, dyadic comparison, algebra transport, replay, provenance, and local name.

Apply Definition 3657.3. The reliability read is ordered through the finite edge list and budget row before any downstream tag, trace, tolerance, or algebra transport is consumed. Therefore the packet cannot export host probability spaces, polynomial equality, quotient codon equality, stochastic dynamics, infinite trigger search, or a hidden reliability oracle. □

Falsifiable prediction: Every accepted reliability read factors through one finite trigger-edge list and one reliability-budget row.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, cellular-window-trace, dyadic-tolerance-ladder, trigger-hypergraph vision source: none is bijective with $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ because the packet jointly carries boundary coverage, trigger-edge lists, reliability budgets, finite tag accounting, cellular windows, dyadic tolerance comparison, algebra transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.TriggerHypergraphReliabilityUp.TasteGate` supplies explicit Nontrivial $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ and FieldFaithful $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ instances preserving $B, E, R, L, W, D, A, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ packet is generated from boundary-coverage rows, trigger-edge list rows, reliability-budget rows, finite tag ledgers, cellular window traces, dyadic tolerance rows, algebra transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3657.1, Definition 3657.2, Definition 3657.3, and Definition 3657.4 seed the finite trigger-edge reliability NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host probability spaces, ambient polynomial equality, quotient codon equality, stochastic biological dynamics, infinite trigger search, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-edge exposure, reliability-budget exposure, finite tag accounting, window-trace assignment, dyadic-tolerance comparison, algebra transport, Nontrivial TriggerHypergraphReliabilityUp, and FieldFaithful TriggerHypergraphReliabilityUp for BEDC.Derived.TriggerHypergraphReliabilityUp.

Chapter 2693

A Concrete Naming Certificate for $\mathsf{LocatedCompleteInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompleteInterval}^{\uparrow}$ packet records a located complete closed interval as finite BEDC NameCert data. It refines the closed bounded interval carrier of chapter 1591, the located real interval surface of chapter 1709, and the nested interval Cauchy route of chapter 3599. Its completion-facing reads pass through the regular rational sequence packet of chapter 301, the terminal real seal of chapter 38, and the dyadic tolerance substrate of chapter 291. The packet names finite interval completeness evidence; it is not a host completeness axiom, quotient construction, trichotomy oracle, selected limit, or choice principle.

2693.1 Located Complete Interval Carrier

Definition 2693.1 (Located complete interval carrier)**.** *A* $\mathsf{LocatedCompleteInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, L, N, Q, D, S, R, E, K, H, C, P, M)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row that answers finite membership, containment, separation, and refinement requests through endpoint-radius locatedness data.*
3. *N is the* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *row carrying a descending bracket family, diameter budget, and finite Cauchy extraction route.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational centers and endpoint witnesses inspected by the bracket family.*
5. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to measure bracket diameter, endpoint slack, and refinement depth.*
6. *S is the finite observation schedule for bracket requests and Cauchy reads.*
7. *R is the completion-read row that sends each nested bracket request to a sealed* $\mathsf{Real}^{\uparrow}$ *output through B, L, N, Q, D, S.*

8. *E records the interval-membership evidence for the sealed output, again read only through the closed bounded and located interval rows.*
9. *K is the Cauchy-bracket induction ledger, recording the base bracket, successor refinement, diameter decrease, and endpoint containment rows for the displayed finite family.*
10. *H records componentwise* hsame *transport for all rows above.*
11. *C records displayed* Cont *replay from bracket refinement requests to the same completion-read and membership rows.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedCompleteInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{NestedIntervalCauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *M is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$ *row binding the packet.*

The carrier contains no host closed interval object, no quotient equality of real names, no selected representative stream, no trichotomy row, no countable choice schedule, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCompleteIntervalUp.LocatedCompleteIntervalTasteGate single carrier alignment`

Theorem 2693.2 (Located complete interval NameCert obligations)**.** *For the carrier of Definition 3658.1, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for $I = (B, L, N, Q, D, S, R, E, K, H, C, P, M)$ as one* BHist *packet;*
2. *source-interval exactness, requiring endpoint, containment, and Real-seal reads to pass through the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row B;*
3. *located-boundary exactness, requiring membership and separation reads to pass through the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row L;*
4. *nested-bracket exactness, requiring each Cauchy bracket read to pass through the* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *row N and the induction ledger K;*
5. *rational and dyadic budget exactness, requiring all centers, endpoint witnesses, diameters, and tolerances to be read through Q and D;*
6. *completion-read confinement, requiring the sealed* $\mathsf{Real}^{\uparrow}$ *output and its interval-membership evidence to be exported only through R and E;*
7. *transport and replay stability through H, C, P, M.*

These obligations name located interval completeness as a finite certificate surface. They do not assert host completeness, quotient construction, trichotomy, selected limits, countable choice, compactness beyond the displayed bracket route, or a public standard bridge.

Proof. Project the carrier of Definition 3658.1. The projection exposes exactly the closed bounded source row, located interval row, nested Cauchy bracket row, regular rational readback, dyadic tolerance ledger, schedule, completion read, membership evidence, induction ledger, and structural rows.

Source-interval and located-boundary exactness are immediate from the rows B and L. Every endpoint, containment, membership, separation, and refinement read is already routed through those two interval surfaces before a completion-facing consumer can inspect R or E.

Nested-bracket exactness is supplied by N and K. The bracket family, diameter decrease, and endpoint containment facts are displayed as finite rows, and the rational and dyadic budgets used by those rows are exactly Q and D.

Finally the structural rows close the obligation surface. The hsame transports, Cont replay routes, provenance, and local naming row preserve the listed packet without adding a host completeness coordinate, quotient, trichotomy decision, selected limit, or choice schedule. □

Theorem 2693.3 (Located complete interval Cauchy bracket induction)**.** *Let $I = (B, L, N, Q, D, S, R, E, K, H, C,$ be an accepted* $\mathsf{LocatedCompleteInterval}^{\uparrow}$ *carrier. Every nested-bracket Cauchy read exported by I is obtained by induction over the displayed bracket ledger K: the base step reads the source interval through B and L; the successor step refines the current bracket through N, Q, D, and S; and the terminal read exports a sealed* $\mathsf{Real}^{\uparrow}$ *row through R together with interval-membership evidence through E. The induction is finite-window BEDC data, not an appeal to ambient completeness or choice.*

Proof. Begin with the base bracket. The closed bounded source row B supplies the endpoint and containment packet, while the located interval row L supplies the finite membership and separation service for the same endpoints.

For the successor step, read the nested interval row N. Each refinement consumes a rational center or endpoint witness from Q, a positive dyadic tolerance from D, and an observation window from S. The ledger K records that the next bracket remains inside the previous bracket and has the displayed smaller diameter budget.

Iterating this finite ledger yields the Cauchy route named by the packet. The completion-read row R seals the resulting read as a $\mathsf{Real}^{\uparrow}$ output only after the bracket data have passed through B, L, N, Q, D, S, K.

The membership row E then reads the sealed output back into the same interval carrier. Since the construction has used only the displayed bracket induction rows and the structural transports H, C, P, M, no host completeness principle, quotient equality, selected representative, trichotomy, or choice schedule enters the export. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompleteInterval}^{\uparrow}$ packet is generated from a closed bounded interval source row, a located real interval row, a nested interval Cauchy row, regular rational readback, dyadic tolerance data, finite schedules, completion-read and membership rows, a bracket induction ledger, componentwise transport, replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3658.1, Definition 3658.2, and Definition 3658.3 name the seed carrier, finite NameCert obligation surface, and nested-bracket induction route for located complete closed intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, quotient construction, trichotomy, selected limits, countable choice, compactness beyond the displayed nested-bracket route, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, located-boundary exactness, nested-bracket exactness, rational and dyadic budget exactness, completion-read confinement, and bracket induction for the same finite packet.

Chapter 2694

A Concrete Naming Certificate for $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ packet records the closed-interval Cauchy completeness route as finite BEDC NameCert data. It reads the endpoint and containment rows of chapter 1591, the regular rational sequence route of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the located interval boundary of chapter 3658. The packet names the interval-contained Cauchy readback surface; it does not assert open-cover compactness, selected subsequences, quotient equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

2694.1 Interval Cauchy Completeness Carrier

Definition 2694.1 (Interval Cauchy completeness carrier)**.** *An* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *interval row, exposing lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, and the local closed-interval NameCert surface.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy row for the inspected rational approximants, including the finite modulus windows used by the interval read.*
3. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare successive approximants, endpoint slack, and interval containment at the displayed scale.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row selecting exactly the stream observations consumed by the regular-Cauchy and interval-containment checks.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, Q, D, W have been replayed.*
6. *E records that the sealed read remains inside the same closed interval source row B through the located complete interval boundary.*
7. *H records componentwise* hsame *transport for endpoint rows, regular-Cauchy windows, dyadic tolerances, StreamName windows, the Real seal, interval-membership evidence, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite Cauchy-read request through W, Q, D, B and only then to R and E.*
9. *P is the* Pkg *provenance over* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompleteInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$ *row.*

The accepted carrier contains no open-cover compactness coordinate, no selected subsequence, no quotient equality of real names, no trichotomy row, no countable-choice schedule, and no ambient completeness field for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntervalCauchyCompletenessUp.IntervalCauchyCompletenessTasteGate single carrier alignment`

Theorem 2694.2 (Interval Cauchy completeness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *carrier $I = (B, Q, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint-row exactness, regular-Cauchy window consumption, dyadic tolerance exactness, StreamName window confinement, Real seal handoff, interval-membership readback, componentwise transport, continuation replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 3659.1. The projection exposes the closed interval row B, the regular rational row Q, the dyadic ledger D, the finite stream window row W, the Real seal R, the interval-membership evidence E, and the structural rows H, C, P, N.

Endpoint-row exactness is supplied by B. Every endpoint, order, rational containment, and dyadic containment read is already a row of the closed bounded interval packet before the Cauchy consumer can inspect Q, D, W, R, or E.

Regular-Cauchy and tolerance exactness are supplied by Q, D, W. The finite stream window names the observations, the regular rational row names the Cauchy approximants and modulus windows, and the dyadic ledger names the positive tolerances used in the endpoint and successor comparisons.

Finally R and E close the seal route, while H, C, P, N preserve and name only the displayed packet. Since no row stores open-cover compactness, a selected subsequence, quotient equality, trichotomy, countable choice, or ambient Real completeness, none of those objects can enter the local NameCert surface. □

Theorem 2694.3 (Interval Cauchy completeness regular seal)**.** *Let $I = (B, Q, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by I first consumes the finite route*

$$B, W, Q, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the sealed real output is visible only after the closed-interval endpoint rows, StreamName observation windows, regular-Cauchy readback, and dyadic tolerance ledger have all been displayed.

Proof. Begin with a Real-facing consumer of I. Carrier admission exposes R, but only as the seal row downstream of B, W, Q, D.

Replay the continuation row C. The request first reads the closed interval row B, then the StreamName window W, the regular rational Cauchy row Q, and the dyadic tolerance ledger D.

Only after this finite replay can R be read as a $\mathsf{Real}^{\uparrow}$ seal. The interval-membership row E then reads the same sealed output back through the source interval, and the structural rows H, P, N transport, package, and name that route without adding any separate completeness principle. □

Theorem 2694.4 (Interval Cauchy completeness no open-cover route)**.** *Every consumer of an* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *packet receives a finite regular-Cauchy interval route through* $B, Q, D, W, R, E, H, C, P$ *The packet has no row from which an open-cover compactness proof, a selected subsequence, quotient equality for real names, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness can be recovered.*

Proof. Inspect the carrier of Definition 3659.1. Its non-structural rows are exactly the closed interval source, regular rational Cauchy readback, dyadic tolerance ledger, StreamName window, Real seal, and interval-membership readback.

Open-cover compactness and selected subsequences would require separate rows for cover indices, finite subcover choice, or subsequence selection. Quotient equality, trichotomy, countable choice, and ambient Real completeness would likewise require decision, selection, or host-completeness coordinates not present in the packet.

The only exported route is therefore the displayed finite Cauchy interval route. Componentwise transport, continuation replay, provenance, and local NameCert evidence preserve that route; they do not create any of the excluded coordinates. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ packet is generated from a closed bounded interval row, a RegSeqRat regular-Cauchy row, a DyadicRatCore tolerance ledger, StreamName finite windows, a RealUp seal, interval-membership readback through the located complete interval boundary, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3659.1, Definition 3659.2, Definition 3659.3, and Definition 3659.4 name the seed carrier, finite NameCert obligation surface, regular-Cauchy seal route, and excluded compactness route for closed-interval Cauchy completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subsequences, quotient equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-row exactness, regular-Cauchy window consumption, dyadic tolerance exactness, StreamName window confinement, Real seal handoff, interval-membership readback, and structural stability for the same finite packet.

Chapter 2695

A Concrete Naming Certificate for $\mathsf{LimitPointCompact}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LimitPointCompact}^{\uparrow}$ packet records the sequence-facing compactness boundary: an infinite sequence in a displayed compactly bounded carrier has a named cluster-point route through a subsequence and a real-valued convergence witness. The packet sits beside the $\mathsf{SequentialCompact}^{\uparrow}$ chapter of chapter 2372, the $\mathsf{BolzanoWeierstrass}^{\uparrow}$ chapter of chapter 1254, the $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ chapter of chapter 1580, and the $\mathsf{Real}^{\uparrow}$ chapter of chapter 38. It deliberately names a limit-point route for sequences, not open-cover compactness, arbitrary choice, or ambient real-completeness.

Definition 2695.1 (LimitPointCompact carrier)**.** *A* $\mathsf{LimitPointCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, S, B, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compactly bounded carrier row whose points are read by the sequence packet;*
2. *S is the input sequence row, displaying a finite code for the sequence-facing object under consideration;*
3. *B is the boundedness and membership ledger, keeping every visible sequence value inside the carrier row K;*
4. *W is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *witness row that extracts a subsequence candidate from the bounded sequence data;*
5. *Q is the* $\mathsf{SequentialCompact}^{\uparrow}$ *subsequence route row, naming the selected subsequence only through displayed finite evidence;*
6. *E is the* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *endpoint row, reading the subsequence as converging to a real cluster point through* $\mathsf{Real}^{\uparrow}$ *data;*
7. *H records componentwise* hsame *transport for K, S, B, W, Q, E and the structural rows;*
8. *C records displayed* Cont *replay from the sequence row through boundedness, subsequence extraction, endpoint reading, and local naming;*
9. *P is the* Pkg *provenance over the displayed compactness, subsequence, cluster-point, real, transport, replay, and naming rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^\uparrow}$ *row.*

The classifier accepts exactly packets whose visible coordinates are $K, S, B, W, Q, E, H, C, P, N$. It contains no open-cover subcover coordinate, no countable-choice selector, no ambient $\mathsf{Real}^\uparrow$ *completeness theorem, and no host-level selection of cluster points.*

Lean4 ✓ `BEDC.Derived.LimitPointCompactUp.LimitPointCompactTasteGate single carrier alignment`

Theorem 2695.2 (LimitPointCompact NameCert obligations)**.** *For the carrier of Definition 3660.1, the local* $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^\uparrow}$ *surface consists exactly of five finite obligations:*

1. *carrier admission: every accepted object exposes the displayed packet $L = (K, S, B, W, Q, E, H, C, P, N)$;*
2. *bounded sequence exposure: every sequence value consumed by the packet is read through S and certified inside K by the ledger row B;*
3. *subsequence extraction: every cluster-point route passes through the* $\mathsf{BolzanoWeierstrass}^\uparrow$ *witness row W and the* $\mathsf{SequentialCompact}^\uparrow$ *subsequence row Q;*
4. *endpoint exactness: the cluster endpoint is read only through the* $\mathsf{RealSequenceClusterPoint}^\uparrow$ *row E and its* $\mathsf{Real}^\uparrow$ *coordinate data;*
5. *ledger exhaustion: H, C, P, N preserve, replay, source, and name the displayed route without adding open-cover compactness, countable choice, a selected representative, or ambient real completeness.*

Consequently the certificate names the sequence-facing limit-point compactness route and no stronger compactness principle.

Proof. Carrier admission is projection from the displayed tuple $L = (K, S, B, W, Q, E, H, C, P, N)$. The only point carrier is K, the only input sequence row is S, and the only boundedness ledger is B.

The subsequence part is forced by the witness inventory. The row W is the Bolzano–Weierstrass extraction witness and Q is the sequential-compactness subsequence route; a consumer reaches endpoint data only after those rows have been read.

The endpoint part is then supplied by E. It is the cluster-point row and reads the relevant real-valued convergence data through the $\mathsf{Real}^\uparrow$ packet, so no separate ambient completeness theorem is imported.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no open-cover, choice, selected-representative, or ambient-completeness coordinate, those principles cannot be exported by the local certificate. □

Theorem 2695.3 (LimitPointCompact Bolzano–Weierstrass factorization)**.** *Every accepted* $\mathsf{LimitPointCompact}^\uparrow$ *limit-point consumer factors through*

$$K, S \longmapsto B \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Here W is read from the $\mathsf{BolzanoWeierstrass}^\uparrow$ *surface of chapter 1254, Q is read from the* $\mathsf{SequentialCompact}^\uparrow$ *surface of chapter 2372, and E is read from the* $\mathsf{RealSequenceClusterPoint}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *surfaces of chapter 1580 and chapter 38. The factorization exports no open-cover compactness, no countable choice, no selected representative, and no ambient real-completeness claim.*

Proof. Start with Definition 3660.2. Its bounded sequence exposure clause places every sequence-facing read behind K, S, B.

Next read the extraction rows. The Bolzano–Weierstrass chapter supplies the finite witness row W, and the sequential compactness chapter supplies the subsequence route Q; neither row carries an open-cover subcover object or an unrestricted choice function.

The endpoint is then read through the real sequence cluster-point packet and the real packet. This is exactly the row E, so the conclusion is a displayed sequence endpoint rather than ambient completeness for all real Cauchy data.

The structural rows H, C, P, N preserve and name the same route. Because the carrier inventory contains no stronger compactness coordinate, every consumer remains inside the stated factorization. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3660.1, Definition 3660.2, and Definition 3660.3 seed the sequence-facing compactness route through bounded sequence data, Bolzano–Weierstrass extraction, sequential compactness, real cluster endpoints, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary choice, selected cluster-point representatives, quotient completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded sequence exposure, Bolzano–Weierstrass factorization, sequential-subsequence exactness, real cluster-point endpoint exactness, nonescape ledgers, classifier transport, and provenance/name exhaustion for BEDC.Derived.LimitPointCompactUp.

Constructive story: A $\mathsf{LimitPointCompact}^{\uparrow}$ packet is generated from a compactly bounded carrier row, an input sequence row, a boundedness ledger, a Bolzano–Weierstrass witness row, a sequential-compactness subsequence row, a real sequence cluster-point endpoint row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2696

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$

The $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ packet follows the proof-theoretic normal-form consistency tradition for typed calculi: a closed term displayed at the internal false encoding cannot simultaneously be a beta-strong normal endpoint. In BEDC form, this is not a full closed-consistency claim. It is a finite witness package for the exact closed-normal boundary of chapter 9376, sitting beside the closed-normal canonicity packet of chapter 799, the endpoint ledger of chapter 837, and the Z-normal predicate packet of chapter 936.

Definition 2696.1 (ClosedNormalConsistencyWitness carrier)**.** *A* $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, F, L, B, E, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC term row, displaying the closed proof candidate under audit;*
2. *F is the internal false-encoding row, naming the target type at which the term is displayed;*
3. *L is the empty-context typing ledger for the judgement that T has type F;*
4. *B is the beta-strong endpoint row, recording that the displayed term is already at the normal-form boundary;*
5. *E is the closedness row, recording that the term has no external context dependency;*
6. *R is the false-exclusion witness row read from the closed-normal consistency theorem;*
7. *H records componentwise* hsame *transport for T, F, L, B, E, R and the structural rows;*
8. *C records displayed* Cont *replay from typing and closedness through beta-strong endpoint evidence to false exclusion;*
9. *P is the* Pkg *provenance over MetaCIC typing, closedness, beta-strong endpoint, false encoding, transport, replay, and naming rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are $T, F, L, B, E, R, H, C, P, N$. It contains no subject-reduction transport, no full closed-consistency theorem, no Church–Rosser quotient, no normalization theorem for all closed terms, and no host proof of arbitrary false exclusion.

Lean4 ✓ BEDC.Derived.ClosedNormalConsistencyWitnessUp.ClosedNormalConsistencyWitnessTasteGate
single carrier alignment

Theorem 2696.2 (ClosedNormalConsistencyWitness NameCert obligations)**.** *For the carrier of Definition 3661.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission: every accepted object exposes the displayed packet* $W = (T, F, L, B, E, R, H, C, P, N)$*;*
2. *false typing exposure: the typing ledger* L *displays the empty-context judgement that the term row* T *has the false-encoding type row* F*;*
3. *normal endpoint exposure: the rows* B *and* E *display beta-strong endpoint evidence and closedness before false exclusion is read;*
4. *false-exclusion exactness: the row* R *is available only after* T, F, L, B, E *have been displayed together;*
5. *ledger exhaustion:* H, C, P, N *preserve, replay, source, and name that route without adding full closed consistency, subject reduction, Church–Rosser transport, or global normalization.*

Consequently the certificate names the closed-normal false-exclusion witness and no stronger consistency principle.

Proof. Carrier admission is projection from the displayed tuple. The only term row is T, the only false-encoding row is F, and the only typing ledger is L.

The normal endpoint is part of the input, not a theorem extracted from arbitrary closed typing. Rows B and E display beta-strong normality and closedness before the false-exclusion witness can be read.

The false-exclusion row R is therefore exactly the closed-normal consistency route. It consumes the typing, closedness, and beta-strong evidence together, matching the boundary stated in chapter 9376.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the packet has no subject-reduction, Church–Rosser, global-normalization, or full-consistency coordinate, none of those stronger principles can be exported by the certificate. □

Theorem 2696.3 (ClosedNormalConsistencyWitness false-exclusion factorization)**.** *Every accepted* $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ *consumer of false exclusion factors through*

$$T, F \longmapsto L \longmapsto E \longmapsto B \longmapsto R \longmapsto H, C, P, N.$$

The route reads the term and false-encoding rows, exposes empty-context typing, checks closedness, checks beta-strong endpoint evidence, and only then exports the false-exclusion witness. It does not infer full closed consistency, arbitrary subject-reduction transport, Church–Rosser conversion, quotient equality of proofs, or normalization for all closed terms.

Proof. Use Definition 3661.2. The false typing exposure clause places the term and false encoding behind T, F, L.

Next the normal endpoint exposure clause places the closedness and beta-strong rows before any contradiction is read. This mirrors the main-result boundary: closed and typed alone are not enough for the present packet.

The row R is then the only false-exclusion coordinate. It is reached after the typing, closedness, and beta-strong rows have all been displayed.

The structural rows H, C, P, N preserve and name this route. The missing stronger principles have no carrier coordinate, so the exported witness remains closed-normal rather than full closed consistency. □

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyWitnessUp.ClosedNormalConsistencyWitnessFalseExclusionfactorization`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3661.1, Definition 3661.2, and Definition 3661.3 seed the closed-normal false-exclusion route through MetaCIC term display, false typing, closedness, beta-strong endpoint evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full closed consistency, subject reduction, Church–Rosser conversion, quotient equality of proofs, global normalization for all closed terms, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, false typing exposure, closedness and beta-strong endpoint exactness, false-exclusion factorization, nonescape ledgers, classifier transport, and provenance/name exhaustion for BEDC.Derived.ClosedNormalConsistencyWitnessUp.

Constructive story: A $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ packet is generated from a MetaCIC term row, an internal false-encoding row, an empty-context typing ledger, a closedness row, a beta-strong endpoint row, a false-exclusion witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2697

A Concrete Naming Certificate for $\mathsf{FiniteGroupRepresentation}^{\uparrow}$

Following Frobenius on finite-group characters and the representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ packet records a finite group action on a finite displayed vector-space row as NameCert data. It consumes the group source of chapter 41, the vector-space source of chapter 47, the linear-map and matrix rows of chapter 48 and chapter 49, and the downstream trace and ledger consumers of chapter 245, chapter 246, and chapter 3574. The packet names only the displayed finite action, transport, and consumer boundary rows; it does not assert Maschke semisimplicity, Schur orthogonality, complete reducibility, irreducible decomposition, or a host category of representations.

Definition 2697.1 (FiniteGroupRepresentation carrier)**.** *A* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, V, L, M, A, U, C, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, including the multiplication, unit, inverse, and finite support rows needed to index the action.*
2. *V is the consumed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row for the represented vector-space endpoint.*
3. *L is the consumed* $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ *row naming the linear endomorphism surface over V.*
4. *M is the consumed* $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ *row used only when the finite action is displayed by matrix rows over a chosen presentation already recorded in the packet.*
5. *A is the finite action ledger assigning to each displayed group row a displayed linear endomorphism row and recording the multiplication and unit compatibility rows.*
6. *U is the finite action-unit row, recording that the unit row of G acts by the identity linear-map row on V.*
7. *C is the downstream consumer ledger, exposing only the rows read by* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *and* $\mathsf{CharacterTable}^{\uparrow}$ *consumers.*
8. *H is the componentwise* hsame *transport for G, V, L, M, A, U, C.*
9. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *row.*

No carrier coordinate stores a host representation category, a quotient representation, an irreducible decomposition, a character table, a Maschke complement, or a Schur orthogonality relation.

Lean4 (definition) `BEDC.Derived.FiniteGroupRepresentationUp.FiniteGroupRepresentationUp`

Theorem 2697.2 (FiniteGroupRepresentation NameCert bridge)**.** *For every accepted* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *carrier*

$$R = (G, V, L, M, A, U, C, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission exposes the displayed group row G, vector-space row V, linear-map row L, optional matrix presentation row M, action ledger A, unit row U, consumer ledger C, transport row H, provenance row P, and local naming row N;*
2. *action exactness requires each action entry in A to be sourced by a group row of G and to land in a linear endomorphism row read through L over the represented vector-space row V;*
3. *unit and multiplication stability require the unit row U and the product rows of G to replay through finite* Cont *rows inside A and remain preserved by the componentwise* hsame *transport H;*
4. *matrix presentation compatibility allows M to display action entries only as rows already tied to L and V, so matrix reads cannot create an additional carrier outside the packet;*
5. *consumer boundary requires every* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTable}^{\uparrow}$ *read to pass through C after the action ledger has been displayed, and it excludes Maschke semisimplicity, Schur orthogonality, irreducible decomposition, quotient representation carriers, and host representation categories.*

These five obligations give the finite NameCert bridge from the consumed group and vector-space certificates to the downstream character and representation-ring consumers.

Proof. Project the carrier rows of Definition 3662.1. Carrier admission is exactly the displayed tuple $G, V, L, M, A, U, C, H, P, N$, so no additional source row is available.

For action exactness, read each entry of A as a finite row from the group source G into the endomorphism surface supplied by L over V. The unit row U and the product rows of G are then replayed through the same finite continuation ledger.

Matrix presentation compatibility is only a display route. Any matrix row in M must already be tied to the linear-map and vector-space rows L and V, so it cannot supply a second representation carrier.

Finally expose the consumer ledger C. Character-theory, representation-ring, and character-table consumers read only after the displayed action rows have been named, while Maschke semisimplicity and Schur orthogonality remain in their own chapters. Repacking the projected rows with H, P, and N gives the stated local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteGroupRepresentationUp.FiniteGroupRepresentationNameCertBridge`

Theorem 2697.3 (FiniteGroupRepresentation Schur consumer boundary)**.** *Any* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, *or* $\mathsf{SchurOrthogonality}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *packet*

$$R = (G, V, L, M, A, U, C, H, P, N)$$

must first expose the action ledger A, linear-map row L, vector-space row V, optional matrix row M, and consumer ledger C. A $\mathsf{SchurOrthogonality}^{\uparrow}$ *read may inspect finite character-pairing rows only through those displayed rows and cannot import a host representation category, quotient carrier, irreducible decomposition, or complete character table.*

Proof. Project the accepted carrier tuple of Definition 3662.1. The action ledger A, vector-space row V, linear-map row L, optional matrix row M, and consumer ledger C are the only rows through which downstream character-facing consumers can see the finite representation packet.

Route a Schur-facing request through C. The request may read character-pairing information only after the displayed action entries in A have landed in the endomorphism surface supplied by L over V, with any matrix presentation in M already tied to the same rows.

The exclusions follow from the carrier boundary. Since R contains no host representation category, quotient carrier, irreducible decomposition, or complete character-table coordinate, a $\mathsf{SchurOrthogonality}^{\uparrow}$ consumer cannot obtain any of those endpoints from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3662.1, Definition 3662.2, and Definition 3662.3 seed the finite action carrier, its NameCert obligation bridge, and the Schur-facing consumer boundary over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, and $\mathsf{SchurOrthogonality}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Maschke semisimplicity, Schur orthogonality, irreducible decomposition, complete reducibility, character-table computation, quotient representation carriers, host representation categories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action exactness, unit and multiplication stability, matrix presentation compatibility, and downstream consumer boundary for the same finite packet.

Constructive story: A $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ packet is generated from a displayed finite group source, represented vector-space source, linear-map and matrix presentation rows, a finite action ledger, unit and multiplication replay rows, componentwise transport, package provenance, and local NameCert evidence.

Chapter 2698

A Concrete Naming Certificate for $\mathsf{KSpace}^{\uparrow}$

Following Kelley's compactly generated spaces, the $\mathsf{KSpace}^{\uparrow}$ packet records the topology whose public open rows are tested by compact probes. It sits after the topology carrier of chapter 88, the compact-space certificate surface, and the compact-open function-space consumer of chapter 2221. The packet names compact-test topology rows for downstream continuity and mapping-space consumers; it does not import arbitrary open-cover choice, a host category of spaces, quotient equality of opens, or an ambient set-theoretic coreflection.

Definition 2698.1 (K-space carrier)**.** *A* $\mathsf{KSpace}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$K = (X, T, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-name carrier whose observations are read only through* BHist *rows.*
2. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row listing the candidate open-name classifier.*
3. *Q is the compact-probe ledger: each accepted row records a compact source certificate, a probe map into X, and its compact-domain provenance.*
4. *A is the compact-test topology ledger, saying that an open-name row U is admitted precisely when every compact probe in Q pulls U back to an open row on its compact source.*
5. *E records exactness between T and A for the displayed open names, so the packet exports the compactly generated topology rather than a second unrelated topology.*
6. *H records componentwise* hsame *transport for X, T, Q, A, E, C, P, N.*
7. *C records displayed* Cont *replay from topology requests and compact-probe requests through the same rows.*
8. *P is the* Pkg *provenance over* $\mathsf{KSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, *compact probes,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$ *row.*

The carrier exposes compact-test topology data only through $X, T, Q, A, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.KSpaceUp.KSpaceTasteGate single carrier alignment`

Definition 2698.2 (K-space classifier)**.** *The* $\mathsf{KSpace}^{\uparrow}$ *classifier accepts two carrier rows as the same public packet exactly when their point carrier, ambient topology row, compact-probe ledger, compact-test topology ledger, exactness row, transport row, replay row, provenance row, and local naming row are componentwise related by the displayed* hsame *evidence. It has no coordinate for selected subcovers, host open families, quotient-open representatives, choice of compact probes outside* Q*, or an external coreflection functor.*

Theorem 2698.3 (K-space NameCert obligations)**.** *For every accepted* $\mathsf{KSpace}^{\uparrow}$ *carrier* $K = (X, T, Q, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed point carrier, topology row, compact-probe ledger, compact-test topology ledger, exactness row, transport, replay, provenance, and local naming rows;*
2. *compact-probe accountability, requiring every probe inspected by* A *to be named by the compact-probe ledger* Q*;*
3. *compact-test exactness, requiring the admitted open rows to be exactly those whose pullbacks along all displayed compact probes are open on their compact sources;*
4. *componentwise transport and replay stability under* H *and* C *for the same displayed rows;*
5. *ledger refusal, stating that arbitrary open-cover choice, host compact-object selection, quotient equality of opens, and ambient set-theoretic coreflection are not carrier coordinates.*

No local obligation is stored outside these five rows.

Proof. Project the accepted carrier. The only mathematical rows are X, T, Q, A, E, and the structural rows are H, C, P, N.

The compact-probe accountability row is read through Q. Each pullback test used by A must name its compact source and probe map before the topology request can inspect it.

The exactness row E binds the topology row T to the compact-test ledger A. Hence the packet exports the compactly generated topology determined by the displayed probes, not a freely chosen open-family classifier.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since the excluded objects are absent from $X, T, Q, A, E, H, C, P, N$, no accepted packet can export them. □

Theorem 2698.4 (Compact-test continuity handoff)**.** *A downstream continuity or compact-open function-space consumer of an accepted* $\mathsf{KSpace}^{\uparrow}$ *carrier can inspect openness only through*

$$\mathsf{Topology}^{\uparrow}, \quad \mathsf{KSpace}^{\uparrow}, \quad Q, \quad A.$$

Thus a continuity check against a $\mathsf{KSpace}^{\uparrow}$ *source is routed through displayed compact probes and compact-test topology rows, while open-cover choice, external compact-object selection, and host coreflection data remain outside the packet.*

Proof. Read the topology request through T and the compact-test row through A. The row A can certify an open-name request only after all displayed compact probes in Q have supplied the corresponding pullback-open rows.

The exactness row E then identifies the exported open-name classifier with that compact-test ledger. A compact-open function-space consumer therefore receives the same probe route rather than a host-space construction.

Rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. No extra coordinate is available for open-cover choice, selected compact probes outside Q, or an ambient coreflection principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{KSpace}^{\uparrow}$ packet is generated from a point-name carrier, a TopologyUp row, compact-probe rows, compact-test topology rows, exactness between displayed opens and compact tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3663.1, Definition 3663.2, Definition 3663.3, and Definition 3663.4 seed the compactly generated topology packet for topology and compact-open consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.KSpaceUp.KSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected compact probes outside the displayed ledger, quotient equality of opens, host coreflections, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-probe accountability, compact-test exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.KSpaceUp.

Chapter 2699

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$

Following Heine–Cantor and Lebesgue finite-cover analysis, the $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ packet records the finite selector route used by compact-uniform-continuity proofs. It consumes the compact finite-net source of chapter 3606, the pointwise modulus rows of the continuous-function carrier, the rational lower-bound fold, and the uniform-modulus handoff. The packet names the finite cover selector as BEDC data; it is not an appeal to countable choice, a global modulus oracle, an open-cover compactness theorem, or an ambient minimum over a host set of radii.

Definition 2699.1 (Compact-uniform finite-cover selector carrier)**.** *A* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$S = (K, F, E, M, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric or compact finite-net source row that supplies a finite* ProbeBundle *cover for the requested tolerance.*
2. *F is the continuous-function row whose graph and pointwise modulus are being uniformized.*
3. *E is the finite cover selector ledger, listing the selected compact centers, half-radius coverage rows, and probe provenance.*
4. *M is the pointwise modulus ledger at the selected centers.*
5. *R is the rational lower-bound fold over the displayed positive radius rows.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported after E, M, R have been read in order.*
7. *H records componentwise* hsame *transport for $K, F, E, M, R, U, C, P, N$.*
8. *C records displayed* Cont *replay from compact-uniform requests through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$*, compact finite covers, continuous-function pointwise moduli, rational lower-bound folds,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}}$ *row.*

The carrier exposes only the finite cover selector, pointwise modulus, rational fold, and uniform-modulus handoff rows displayed above.

Lean4 ✓ `BEDC.Derived.CompactUniformFiniteCoverSelectorUp.CompactUniformFiniteCoverSelectorTasteGate`
`single carrier alignment`

Definition 2699.2 (Compact-uniform selector classifier)**.** *The* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *classifier compares accepted packets componentwise through* $K, F, E, M, R, U, H, C, P, N$*. Two packets are classifier-equal exactly when the same compact source, continuous graph, finite cover selector, pointwise modulus rows, rational lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local naming rows are related by the displayed* hsame *evidence. It has no coordinate for an external selector, a host finite minimum, countable choice, a global modulus oracle, or open-cover compactness outside the displayed compact row.*

Theorem 2699.3 (Compact-uniform finite-cover selector obligations)**.** *For every accepted* CompactUniformFiniteCo *carrier* $S = (K, F, E, M, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the compact source, continuous-function row, finite cover selector, pointwise modulus ledger, rational lower-bound fold, uniform handoff, transport, replay, provenance, and local naming rows;*
2. *finite-cover accountability, requiring the selected centers and half-radius coverage rows in* E *to be read from the displayed compact source* K*;*
3. *pointwise-modulus compatibility, requiring every selected center in* E *to carry the matching pointwise modulus row in* M *for the same continuous-function row* F*;*
4. *rational-fold exactness and uniform handoff, requiring* R *to be a certified lower bound for the displayed radii before* U *exports the tolerance-only precision row;*
5. *ledger refusal, stating that countable choice, external finite minima, global modulus oracles, host open-cover compactness, and ambient completeness are not carrier coordinates.*

No accepted selector packet carries any additional NameCert obligation.

Proof. Project the accepted carrier. The compact source, continuous-function row, finite cover selector, pointwise modulus ledger, rational fold, and uniform handoff are precisely K, F, E, M, R, U.

The finite-cover accountability row is read first. Row E may list only centers and half-radius coverage witnesses supplied by the compact source row K.

The pointwise modulus row is then checked at those centers. Compatibility requires the ledger M to use the same continuous-function row F and the same selected centers exposed by E.

The rational fold R is read before the uniform handoff U. It certifies one positive precision history below the displayed half-radii, so U exports a tolerance-only uniform-modulus row without selecting a radius outside the finite ledger.

Rows H, C, P, N transport, replay, source, and name this route. Since no further coordinate occurs in S, the refused external principles cannot be exported by the local NameCert surface. □

Theorem 2699.4 (Finite selector to uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *carrier reaches its precision history only through the finite route*

$$\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}, \quad E, \quad M, \quad R, \quad U.$$

The consumer sees a displayed compact finite-net selector, pointwise modulus rows at its centers, a certified rational lower-bound fold, and the uniform-modulus handoff; it receives no countable-choice selector, host minimum, or open-cover compactness datum.

Proof. Start with the compact finite-net selector E. It exposes a finite ProbeBundle of centers and half-radius coverage rows already tied to the compact source K.

At each displayed center, read the pointwise modulus row in M for the same continuous-function row F. The row order prevents a modulus from being imported for a different graph or an unlisted center.

Fold the positive radius rows through R. The fold certificate supplies one lower-bound precision history, and only then can U export the uniform-modulus handoff.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Therefore the consumer-facing datum is the finite BEDC selector route, not an external choice or host minimum. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}$.

Constructive story: A $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ packet is generated from a compact source row, a continuous-function row, compact finite-net selector rows, pointwise modulus rows, rational lower-bound fold rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3664.1, Definition 3664.2, Definition 3664.3, and Definition 3664.4 seed the finite cover selector packet for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, external finite minima, global modulus oracles, host open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover accountability, pointwise-modulus compatibility, rational-fold exactness, uniform handoff, transport and replay stability, and local NameCert registration for BEDC.Derived.CompactUniformFiniteCoverSelectorUp.

Chapter 2700

A Concrete Naming Certificate for $\mathsf{BishopRealCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealCompletion}^{\uparrow}$ packet records real completion by regular rational names with located finite observation data. It sits on the dyadic arithmetic substrate of chapter 291, the finite stream-window surface of chapter 126, the regular rational sequence packet of chapter 301, the terminal real-name seal of chapter 38, and the located completion route of chapter 3488.

Definition 2700.1 (Bishop real completion carrier)**.** *A* $\mathsf{BishopRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger whose positive dyadic rows name the tolerances inspected by the completion route;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the finite rational observations read from the candidate Cauchy name;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular rational values visible on the windows in S;*
4. *L is the* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *located-completion row, recording the located brackets available for the same dyadic precision ledger and stream windows;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached from D, S, Q, L;*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readbacks, located-completion row, Real-seal row, replay row, provenance row, and local naming row;*
7. *C records displayed* Cont *replay from dyadic precision queries through stream readback, located completion, and Real sealing;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *evidence.*

The carrier contains no quotient real object, no selected limiting representative, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRealCompletionUp.TasteGate.BishopRealCompletionTasteGate single carrier alignment`

Theorem 2700.2 (Bishop real completion NameCert obligations)**.** *For every accepted carrier* $R = (D, S, Q, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; dyadic precision exactness through* D*; finite stream-window readback through* S*; regular rational Cauchy readback through* Q*; located completion handoff through* L*; Real-seal exposure through* E*; componentwise* hsame *stability through* H*; displayed* Cont *replay through* C*;* Pkg *provenance through* P*; and local naming through* N*.*

Proof. Carrier admission is coordinate projection from Definition 3665.1. The packet displays the dyadic precision ledger, finite stream windows, regular rational readbacks, located-completion row, Real-seal row, transport, replay, provenance, and local name as one BHist surface.

The dyadic and stream obligations are read before the located-completion row. Thus every precision request uses a positive row of D, every rational observation enters through the finite window ledger S, and every rational value is exposed by the regular readback row Q.

The located-completion and Real-seal obligations are then forced through L and E. Since H and C are the only transport and replay rows, any classified mate of the packet preserves the same finite route.

Finally P and N record provenance and local naming for exactly these rows. The carrier inventory has no quotient real, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, so the listed obligations exhaust the certificate surface. □

Theorem 2700.3 (Bishop real completion Real-seal route)**.** *Every Real-facing consumer of an accepted* $\mathsf{BishopRealCompletion}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopLocatedRealCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $D, S, Q, L, E, H, C, P, N$*. The route supplies finite precision, window, readback, located-completion, and seal rows, and it supplies no quotient real equality, no selected limit, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $R = (D, S, Q, L, E, H, C, P, N)$. The only Real-facing row is E, and the only rows feeding it are the dyadic ledger D, stream windows S, regular rational readbacks Q, and located-completion row L.

Use the NameCert obligations of Definition 3665.2. They force precision queries, observations, regular readbacks, located completion, and Real sealing to pass through the same displayed carrier rows.

Transport and replay do not add a second route. The rows H and C preserve and replay the displayed path componentwise, while P and N record its provenance and local name.

The carrier excludes quotient real equality, selected limits, countable-choice schedules, and ambient completeness. Hence every Real-facing consumer receives exactly the finite Real-seal route stated above. □

Theorem 2700.4 (Bishop real completion local NameCert exhaustion)**.** *For any accepted* $\mathsf{BishopRealCompletion}^{\uparrow}$ *carrier*

$$R = (D, S, Q, L, E, H, C, P, N),$$

every local $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *consumer factors through exactly the five dependency rows*

$$D, S, Q, L, E$$

followed by the structural rows H, C, P, N*. The first five rows are respectively the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger,* $\mathsf{StreamName}^{\uparrow}$ *window ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *located-completion row, and* $\mathsf{Real}^{\uparrow}$ *seal; no consumer-visible branch is supplied by quotient real equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier using Definition 3665.1. The displayed dependency coordinates are exactly D, S, Q, L, E, and the only structural coordinates are H, C, P, N.

Split over the finite row inventory. Precision requests land in D, finite observations in S, regular rational values in Q, located completion data in L, and Real-facing seals in E.

Apply Definition 3665.2. Its obligation list identifies the same dependency rows and then the same transport, replay, provenance, and local naming rows as the full local certificate surface.

Finally use Definition 3665.3 for consumer-visible reads that reach $\mathsf{Real}^{\uparrow}$. That route ends at E after D, S, Q, L, and the remaining rows H, C, P, N preserve and name it without adding a second branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealCompletion}^{\uparrow}$ packet is generated from a dyadic precision ledger, finite StreamName windows, RegSeqRat readbacks, a BishopLocatedRealCompletion row, a Real seal row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3665.1, Definition 3665.2, and Definition 3665.3 name the seed carrier, finite NameCert obligation surface, and Real-seal route for Bishop real completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert quotient reals, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic precision exactness, finite stream-window readback, regular rational readback, located-completion handoff, Real-seal exposure, transport stability, replay, provenance, and local naming for the displayed Bishop real completion packet.

Chapter 2701

A Concrete Naming Certificate for $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley uniform-space terminology, the $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ packet records the uniform analogue of the L10 completion universal property. It sits after the uniform-completion carrier of chapter 1473, the generic completion surface of chapter 397, and the regular-rational readbacks of chapter 126, chapter 301, and chapter 38. The packet names extension and uniqueness only through finite uniform rows, not through quotient completions, selected limits, ambient complete spaces, or choice.

Definition 2701.1 (Uniform completion universal-property carrier)**.** *A* $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, D, T, E, L, M, W, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform source row, exposing only the displayed* $\mathsf{UniformSpace}^{\uparrow}$ *entourages and their finite source windows.*
2. *D is the dense unit row from the source into the completion carrier, aligned with the* $\mathsf{UniformCompletion}^{\uparrow}$ *unit surface.*
3. *T is the uniformly continuous target row, naming the complete separated target interface through finite entourage/modulus data.*
4. *E is the extension row for uniformly continuous source consumers, read only after U, D, T have been displayed.*
5. *L is the uniqueness row, comparing two displayed extensions by* hsame *over the dense unit and the same target row.*
6. *M is the entourage/modulus ledger for source entourages, target entourages, and the finite modulus witnesses used by E and L.*
7. *W is the readback row for* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *observations downstream of the same uniform route.*
8. *H stores componentwise* hsame *transport for the source, dense unit, target, extension, uniqueness, modulus, readback, continuation, provenance, and naming rows.*

9. *C stores displayed* Cont *replay from source windows through the dense unit, target modulus, extension row, uniqueness row, and readback row.*

10. *P is the* Pkg *provenance over* UniformCompletionUniversalProperty$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CompletionUniversalPro UniformSpace$^{\uparrow}$, UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* NameCert$_{\text{UniformCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, T, E, L, M, W, H, C, P, N$. It has no coordinate for quotient completion points, selected representatives, ambient target completeness, host function extensionality, or countable choice.

Theorem 2701.2 (Uniform completion universal-property NameCert obligations)**.** *For every accepted* UniformCompletionUniversalProperty$^{\uparrow}$ *carrier $K = (U, D, T, E, L, M, W, H, C, P, N)$, the local* NameCert$_{\text{UniformCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, uniform-source exposure through U, dense-unit exposure through D, complete separated target exposure through T, extension existence through E, uniqueness through L, entourage/modulus accounting through M,* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *readbacks through W, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.UniformCompletionUniversalPropertyUp.UniformCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 3666.1. The source, dense unit, target, extension, uniqueness, modulus ledger, readback row, transport, replay, provenance, and local name are exactly the displayed coordinates of one finite BHist packet.

The extension row E is available only after the source row U, dense unit row D, target row T, and entourage/modulus ledger M have been displayed. The uniqueness row L compares extensions over the same rows by the recorded hsame ledger.

The readback row W is downstream of that uniform route. Since the classifier has no remaining coordinate for selected limits, quotient completions, host function equality, countable choice, or ambient target completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2701.3 (Uniform completion universal-property extension uniqueness)**.** *Let $K = (U, D, T, E, L, M, W, H$ be an accepted* UniformCompletionUniversalProperty$^{\uparrow}$ *carrier. Every extension-and-uniqueness consumer factors through*

$$U, D, T, M \longmapsto E \longmapsto L \longmapsto W$$

together with H, C, P, N. Thus extension existence, uniqueness over the dense unit, and any StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *readback are all read from the same finite uniform packet, and no consumer obtains a quotient completion, selected extension, ambient complete target theorem, or choice principle.*

Proof. Read the carrier definition. The extension coordinate is E, and its stated inputs are the uniform source U, dense unit D, uniformly continuous target T, and entourage/modulus ledger M.

The uniqueness row L is not an independent equality principle. It compares displayed extensions only along the same dense unit and target row, using the componentwise hsame route recorded by H and replayed by C.

Finally inspect the downstream rows. The readback coordinate W is attached to the same route, while P and N record provenance and local naming. Row exhaustion excludes every omitted host object, so the extension-and-uniqueness surface is exactly the finite route displayed above. □

Falsifiable prediction: Every accepted $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence or uniqueness while omitting the uniform source, dense unit, uniformly continuous target, extension row, uniqueness row, entourage/modulus ledger, readback row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: uniformcompletion, completionuniversalproperty, uniformspace, uniformmodulus, streamname, regseqrat, real: $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries uniform source, dense unit, target continuity, extension, uniqueness, modulus ledger, readbacks, transport, replay, provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ packet is generated from a uniform source row, dense unit row, uniformly continuous target row, extension row, uniqueness row, entourage/modulus ledger, StreamName/RegSeqRat/Real readbacks, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3666.1, Definition 3666.2, and Definition 3666.3 seed the finite uniform universal-property certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected extensions, ambient complete target theorems, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, extension existence, uniqueness, readback nonescape, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.UniformCompletionUniversalPropertyUp.

Chapter 2702

A Concrete Naming Certificate for CauchyCompletionStableReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completion, the CauchyCompletionStableReflection$^{\uparrow}$ packet records the stable reflection surface by which a displayed completion functor, its universal property, separatedness, regular rational sequence data, and terminal real-name rows remain attached to one finite NameCert route. It sits after the completion functor of chapter 423, the universal handoff of chapter 397, the separated limit classifier of chapter 1411, the regular rational sequence rows of chapter 301, and the Real seal of chapter 38. The packet names a stable reflection certificate for the displayed route; it does not assert quotient completion construction, selected limits, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 2702.1 (Cauchy completion stable reflection carrier)**.** *A* CauchyCompletionStableReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyCompletionFunctor$^{\uparrow}$ *row, containing the named source metric, target completion surface, and functorial map route.*
2. *U is the* CompletionUniversalProperty$^{\uparrow}$ *row, storing the extension and uniqueness handoff for maps out of the displayed dense source.*
3. *S is the* SeparatedMetric$^{\uparrow}$ *row, recording the zero-distance classifier used for completion-limit uniqueness.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *row, carrying the regular rational sequence schedules and finite tolerance windows read by the completion route.*
5. *E is the* Real$^{\uparrow}$ *seal row reached from the regular sequence and zero-distance data.*
6. *H records componentwise* hsame *transport for functor, universal-property, separatedness, regular-sequence, Real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from F, Q through U, S, E and back to the local naming row.*
8. *P is the* Pkg *provenance over* CauchyCompletionStableReflection$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}$ *row.*
The carrier contains no quotient completion object, no chosen representative of a limit, no external completeness theorem for $\mathsf{Real}^{\uparrow}$*, and no route outside the displayed BEDC rows.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionStableReflectionUp.CauchyCompletionStableReflectionTasteGate`
`single carrier alignment`

Theorem 2702.2 (Cauchy completion stable reflection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ *carrier* $R = (F, U, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableRefl}}$ *surface consists exactly of carrier admission for* $F, U, S, Q, E, H, C, P, N$*; functor stability through* F*; extension uniqueness through* U*; separated zero-distance classification through* S*; regular rational window control through* Q*; Real seal confinement through* E*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Carrier admission is projection from Definition 3667.1. The accepted packet displays exactly the completion functor row, the universal-property row, the separatedness row, the regular rational sequence row, the Real seal, and the structural rows.

Functor stability and extension uniqueness are read in that order. The row F supplies the completion-facing map route, while U allows an extension consumer to compare two displayed routes only after they share the same source, completion surface, and regular sequence data.

Separatedness and Real sealing are likewise internal to the packet. The row S turns zero-distance evidence into classifier equivalence, the row Q supplies the finite regular rational windows that feed the completion comparison, and E seals the resulting value inside the displayed $\mathsf{Real}^{\uparrow}$ boundary.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since the carrier has no quotient object, selected limit, countable-choice schedule, trichotomy row, or ambient Real-completeness coordinate, every local NameCert obligation is confined to the listed rows. □

Theorem 2702.3 (Cauchy completion stable reflection extension route)**.** *Every extension-facing consumer of an accepted* $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow} \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the consumer receives the rows $F, U, S, Q, E, H, C, P, N$ *and no quotient completion carrier, selected representative, hidden completion theorem, countable-choice schedule, trichotomy row, or off-packet certificate.*

Proof. Begin with the functor row F. It is the only coordinate that exposes the completion-facing map route and the source-to-completion handoff.

Next use the universal-property row U. It compares extension candidates only after the route through F is fixed, so extension uniqueness cannot be read as a host equality or as a selected representative of a completed object.

The separatedness and regular-sequence rows then bind the uniqueness read to finite BEDC data. The row S supplies the zero-distance classifier, Q supplies the regular rational windows that make the completion comparison visible, and E confines the resulting value to the Real seal.

Finally H, C, P, N transport, replay, source, and name that same route. Coordinate exhaustion rules out any hidden quotient completion, ambient Real-completeness theorem, choice schedule, trichotomy row, or off-packet certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a CompletionUniversalProperty row, a SeparatedMetric row, a RegSeqRat row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3667.1, Definition 3667.2, and Definition 3667.3 seed the finite stable-reflection certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, functor stability, extension uniqueness, separated zero-distance classification, regular rational window control, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionStableReflectionUp.

Chapter 2703

A Concrete Naming Certificate for $\mathsf{SetLike}^{\uparrow}$

Following Russell and Bourbaki-style set-membership grounding, the $\mathsf{SetLike}^{\uparrow}$ packet records the finite BEDC carrier by which membership, subset implication, restricted comprehension, and extensional boundary reads are exposed without importing an ambient powerset, host quotient, or impredicative universe. It sits before the displayed family discipline of chapter 3600, the syntactic carrier of chapter 197, and the satisfaction carrier of chapter 198. The packet names only finite membership source, classifier, implication, bounded comprehension, extensionality boundary, transport, replay, provenance, and local naming rows.

Definition 2703.1 (SetLike carrier)**.** *A* $\mathsf{SetLike}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, Q, I, R, E, H, C, P, N)$$

with the following rows:

1. *M is the membership-source row, naming the finite source from which element and set-read markers are admitted;*
2. *Q is the classifier row for displayed membership decisions and membership-compatible reads;*
3. *I is the subset-implication row, recording finite implications from one membership classifier to another;*
4. *R is the restricted-comprehension row, admitting only classifiers already routed through M, Q, I;*
5. *E is the extensionality-boundary ledger, recording when two displayed set rows may be compared by the carried membership classifier;*
6. *H records componentwise* hsame *transport for M, Q, I, R, E together with the structural rows;*
7. *C records* Cont *replay through which a consumer reads membership source, classifier, subset implication, restricted comprehension, and extensionality boundary data;*
8. *P is the* Pkg *provenance over* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{ModelTheory}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, global choice of elements, or a host universe of all sets.

Lean4 ✓ BEDC.Derived.SetlikeUp.SetlikeTasteGate single carrier alignment

Definition 2703.2 (SetLike membership carrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the membership carrier is the finite BHist *surface*

$$S_{\in} = (M, Q, H, C, P, N).$$

Here M is the membership-source row, Q is the classifier-based membership row, H is the componentwise hsame *transport for the displayed rows, C is the* Cont *replay route, P is the* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The membership carrier has no coordinate for host elementhood, powersets, quotient equality of sets, unrestricted comprehension, or an external universe of sets.*

Lean4 (definition) BEDC.Derived.SetlikeUp.SetlikeMembershipCarrier

Theorem 2703.3 (SetLike membership stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P,$ every classifier-based membership read through the membership carrier $S_{\in} = (M, Q, H, C, P, N)$ remains the same finite* BHist *read after componentwise* hsame *transport and* Cont *replay. In particular, transport may expose only the membership source M, classifier row Q, structural route H, C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row N.*

Lean4 ✓ BEDC.Derived.SetlikeUp.SetlikeMembershipStability

Proof. Project the membership carrier of Definition 3668.2. Its only membership data are the source row M and the classifier row Q.

Transport is componentwise through H, and replay is displayed through C. These rows preserve and expose the same classifier-based membership surface rather than creating a second element relation.

The rows P and N source and name the same finite packet. Hence a transported membership read remains a BHist read through M, Q, H, C, P, N and cannot become host elementhood, a powerset coordinate, quotient equality, or unrestricted comprehension. □

Theorem 2703.4 (SetLike membership classifier public boundary)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, every* $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *or root consumer must read membership through M, Q, H, C, P, N, subset and restricted-comprehension data through I, R, and the extensional boundary through E. The public boundary exports no host membership relation, powerset, unrestricted separation rule, quotient set equality, selected representative, or external satisfaction predicate.*

Proof. Project Definition 3668.1 and Definition 3668.2. The membership carrier exposes only the source row M, classifier row Q, transport row H, replay row C, provenance row P, and local naming row N.

Apply Definition 3668.3. A membership read transported through the public boundary remains the same finite classifier read, so a downstream root or typed consumer cannot replace it by host elementhood or an external satisfaction predicate.

For subset and comprehension consumers, use Definition 3668.5 and Definition 3668.6. These place subset implication and restricted comprehension behind I, R over the displayed membership classifier.

Finally read the extensional boundary through E and close by the existing consumer-boundary rows. Since no coordinate of S stores a powerset, unrestricted separation rule, quotient set equality, selected representative, or external satisfaction predicate, the public boundary exports exactly the displayed rows. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierPublicBoundary`

Theorem 2703.5 (SetLike subset ledger exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the subset ledger is exact: every accepted subset read is the implication row* I *over the classifier-based membership carrier* $S_{\in}$*, together with the restricted-comprehension row* R *only as a bounded consumer of* M, Q, I*. No subset consumer receives host-set extensionality, a global inclusion predicate, quotient equality of sets, or a powerset row outside* M, Q, I, R, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetLedger exactness`

Proof. Start with the membership carrier $S_{\in}$ and the subset row I. The classifier Q is the only membership classifier available to the implication row, and M is the only membership source.

The restricted-comprehension row R is bounded by the same displayed route. It can consume M, Q, I but does not provide an ambient subset object.

Transport and replay through H, C preserve the same ledger, while P, N record provenance and local naming. Since no extra coordinate exists, the subset ledger is exactly the finite route M, Q, I, R, H, C, P, N. □

Theorem 2703.6 (SetLike comprehension scope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P,$ *restricted comprehension through* R *is scoped by classifier-based membership over* M, Q *and subset implication through* I*. The resulting read remains a finite* BHist *row transported by* H*, replayed by* C*, sourced by* P*, and named by* N*; it does not assert unrestricted separation, host-set extensionality, quotient equality, or a universe of all subsets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeComprehensionScope`

Proof. Read R from the carrier of Definition 3668.1. The row is restricted comprehension, not a powerset constructor.

The scope is supplied by the membership carrier of Definition 3668.2 and the implication row I. Thus a comprehension read is accepted only when its classifier is already routed through M, Q, I.

Finally apply the structural rows. Componentwise hsame transport, displayed Cont replay, provenance, and local naming keep the same finite row visible, so the comprehension scope does not become unrestricted separation, host-set extensionality, quotient equality, or a universe of all subsets. □

Theorem 2703.7 (SetLike bounded-comprehension induction)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every accepted restricted-comprehension read through* R *is generated by finite induction over the displayed membership-source row* M *and subset-implication row* I*. The generated row is transported through* H *and replayed through* C *before provenance* P *and local naming* N *are consumed, and the route cannot introduce unrestricted separation, powerset data, quotient set equality, selected representatives, or an external universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionInduction`

Proof. Project the carrier of Definition 3668.1. The only rows available before restricted comprehension is read are the membership-source row M, the classifier row Q, and the subset-implication row I.

Apply Definition 3668.6. It fixes R as a bounded consumer of M, Q, I, so the induction has no case for a host powerset, unrestricted separator, quotient equality witness, or external universe.

Induct over the displayed BHist packet. The base case reads the empty or initial membership-source row and carries no unrestricted subset data. In the finite step case, a new displayed membership row from M may be admitted only when the implication row I keeps it inside the already displayed classifier route.

Transport through H and replay through C preserve the same generated route, while P and N only source and name the finite packet. Thus every accepted restricted-comprehension read is produced by the finite row induction and cannot export any refused comprehension principle. □

Theorem 2703.8 (SetLike finite membership induction)**.** *Every accepted finite* $\mathsf{SetLike}^{\uparrow}$ *membership ledger whose rows are generated from displayed* BHist *membership observations and transported by* hsame *can be consumed by induction over the displayed ledger rows. The induction exports only the carrier row, membership row, classifier row, transport row,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row; it exports no host powerset membership, quotient-set equality, unrestricted comprehension, selected representatives, or ambient extensional equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFiniteMembershipInduction`

Proof. Start from the membership carrier of Definition 3668.2. Its displayed ledger exposes only M, Q, H, C, P, N, so the base case is the empty or initial membership observation together with the same carrier, replay, provenance, and naming rows.

For the finite step case, append one displayed BHist membership observation. The row is admitted only through the classifier row Q and then transported componentwise by hsame through H.

Finally read the structural rows. The Cont replay C, provenance P, and local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ row N repack the same finite ledger for downstream consumers, while Definition 3668.3 blocks replacement by host powerset membership or ambient extensional equality. □

Theorem 2703.9 (SetLike membership-family BHist induction)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any finite membership-family consumer generated from* $M, Q, I, R, E, H, C, P, N$ *is obtained by induction on the displayed* BHist *membership source of Definition 3668.2. The step case first routes a displayed membership request through the classifier row* Q*, then through the subset-implication row* I*, and only then may the restricted-comprehension row* R *consume it. The induction has no branch for powersets, unrestricted separation, quotient set equality, host universe rows, selected representatives, or ambient elementhood.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipFamilyBHistInduction`

Proof. Start with Definition 3668.1 and Definition 3668.2. The displayed membership-family source is $S_{\in} = (M, Q, H, C, P, N)$, so the base case exposes only the membership source, classifier, transport, replay, provenance, and local naming rows.

For the finite step case, read the classifier row Q before any subset or comprehension consumer is allowed. The implication row I records the displayed subset route over that classifier, and Definition 3668.7 then permits R to consume only the already routed M, Q, I data.

The extensional boundary remains separate. By Definition 3668.82, E is a displayed comparison boundary and not a quotient set equality row. Hence the induction never opens powerset data,

unrestricted separation, quotient equality, host universe rows, selected representatives, or ambient elementhood. □

Theorem 2703.10 (SetLike NameCert obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *surface consists exactly of carrier admission for* S*, membership-source admission through* M*, classifier exposure through* Q*, subset-implication exposure through* I*, restricted-comprehension exposure through* R*, extensionality-boundary exposure through* E*, componentwise* hsame *stability through* H*,* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeNameCertObligations`

Proof. Project the accepted carrier of Definition 3668.1. The membership source M and classifier Q provide the only entry points for element-in-set reads.

The subset row I records implication between displayed classifiers, while the restricted-comprehension row R may bind only predicates already exposed through the same membership source and classifier route. The extensionality row E records the comparison boundary for displayed set rows; it does not identify host sets outside the packet.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert obligations cannot require unrestricted separation, ambient powersets, quotient equality of sets, selected representatives, global element choice, or a universe of all sets. □

Theorem 2703.11 (SetLike membership source admission)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every membership read must expose the membership-source row* M*, displayed classifier row* Q*, componentwise transport row* H*, continuation replay row* C*, package provenance row* P*, and local naming row* N *before the read is accepted. In particular, a membership consumer cannot replace* M, Q, H, C, P, N *by an ambient element relation, a host powerset, selected representatives, or an external universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipSourceAdmission`

Proof. Project the carrier of Definition 3668.1. The only rows that open an element-in-set assertion are the membership source M and displayed classifier Q.

Use the obligation surface of Definition 3668.10. It requires membership-source admission, classifier exposure, componentwise hsame stability, Cont replay, provenance, and local naming as visible NameCert rows.

The existing route theorem Definition 3668.27 then blocks every bypass. Since no carrier coordinate stores an ambient element relation, powerset, representative selector, or host universe, every accepted membership read must expose M, Q, H, C, P, N. □

Theorem 2703.12 (SetLike classifier transport route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, classifier transport for subset implication, restricted comprehension, and extensional boundary rows stays inside the finite route*

$$Q, I, R, E \longmapsto H, C, P, N.$$

The transported classifier remains the displayed membership classifier over M *and cannot be reinterpreted as proof-irrelevant host truth, unrestricted separation, quotient equality of sets, or an external inclusion relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierTransportRoute`

Proof. Start from the classifier row Q in Definition 3668.1. The subset implication row I, restricted-comprehension row R, and extensionality-boundary ledger E are all defined as displayed rows that consume the same membership classifier.

Apply Definition 3668.58, Definition 3668.59, and Definition 3668.60. These obligations keep subset, comprehension, and extensional comparison reads attached to M, Q and to the structural rows H, C, P, N.

Finally use Definition 3668.68. Transport and replay preserve the displayed classifier route rather than replacing it by host truth, unrestricted separation, quotient equality, or external inclusion. □

Theorem 2703.13 (SetLike membership carrier admission)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any membership-facing consumer must first admit the membership carrier*

$$S_{\in} = (M, Q, H, C, P, N)$$

from Definition 3668.2. The admissible read exposes M, Q, H, C, P, N *as source, classifier, transport, replay, provenance, and local naming rows, and it cannot inspect host elementhood, a powerset coordinate, unrestricted comprehension, quotient equality, selected representatives, or an ambient universe before those rows are visible.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierAdmission`

Proof. Project the carrier of Definition 3668.1 and the membership carrier of Definition 3668.2. The only membership rows available at this stage are the source row M and classifier row Q.

Use Definition 3668.10. The NameCert surface requires componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N for the same displayed rows.

Finally apply Definition 3668.12. Classifier transport keeps the read inside the displayed route, so no consumer can replace M, Q, H, C, P, N by host elementhood, powerset data, unrestricted comprehension, quotient equality, selected representatives, or an ambient universe. □

Theorem 2703.14 (SetLike membership kernel obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every consumer that reads membership or subset data from* S *must first expose the membership source* M*, displayed classifier* Q*, subset implication* I*, restricted comprehension* R*, extensional boundary* E*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. The admissible read is therefore the concrete kernel route*

$$M, Q, I, R, E \longmapsto H, C, P, N,$$

and it cannot be replaced by an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a host universe.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipKernel obligations`

Proof. Project the carrier of Definition 3668.1. Its membership-facing coordinates are exactly the source row M and the classifier row Q, while subset and bounded family reads are carried only by I and R.

Use Definition 3668.13. A membership-facing consumer must admit $S_{\in} = (M, Q, H, C, P, N)$ before it can inspect element-in-set data, so the source and classifier are visible before the structural rows are replayed.

Next apply Definition 3668.12. The subset implication, restricted comprehension, and extensional boundary rows stay attached to the displayed classifier route Q, I, R, E under H, C, P, N.

It remains to exclude host-level replacements. The membership route theorem Definition 3668.27 and the bounded-comprehension and extensional-boundary obligations keep every membership, subset, comprehension, and comparison read inside $M, Q, I, R, E, H, C, P, N$. Since Definition 3668.1 has no powerset, unrestricted-separation, quotient-equality, representative-selection, or host-universe coordinate, every accepted consumer must expose the displayed kernel route before reading membership or subset data. □

Theorem 2703.15 (SetLike subset classifier exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, subset implication and restricted comprehension are exact over the classifier route*

$$Q, I, R, E \longmapsto H, C, P, N.$$

The implication row I *and restricted-comprehension row* R *may use only the displayed classifier* Q *and extensional-boundary ledger* E *before transport, replay, provenance, and local naming are read. They export no powerset row, host-universe coordinate, unrestricted separation rule, quotient set equality, or external inclusion predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetClassifierExactness`

Proof. Start from Definition 3668.1. The rows I and R are displayed subset-implication and restricted-comprehension rows over the classifier row Q, while E records the comparison boundary.

Apply Definition 3668.10. Its obligation surface keeps subset, comprehension, and boundary reads attached to the accepted packet before structural replay.

Use Definition 3668.12. Since classifier transport stays inside Q, I, R, E, H, C, P, N, exactness is a property of the displayed rows rather than an ambient powerset, host universe, unrestricted separation rule, quotient equality, or external inclusion relation. □

Theorem 2703.16 (SetLike TypeLike and model-theory handoff)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer that reads family, satisfaction, or model data from an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must first pass through the membership carrier* M, Q, H, C, P, N *and the classifier route* Q, I, R, E, H, C, P, N*. Satisfaction-facing and family-facing reads therefore remain downstream of the SetLike membership and classifier surface, not of an ambient set-theoretic assumption.*

Proof. Begin with the accepted carrier of Definition 3668.1. The membership source, classifier, subset implication, restricted comprehension, and extensional boundary rows are M, Q, I, R, E.

Use Definition 3668.2 and Definition 3668.10. These fix the membership carrier and require the structural rows H, C, P, N before a sibling consumer receives any membership data.

Apply Definition 3668.12. The classifier route remains displayed through Q, I, R, E, H, C, P, N, so $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ consumers may build family, syntax, and satisfaction reads only after that SetLike surface is visible.

Thus no downstream consumer obtains satisfaction, family membership, or model-theory data from an ambient powerset, host universe, external satisfaction relation, quotient set equality, or unrestricted comprehension principle. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypelikeModeltheoryHandoff`

Theorem 2703.17 (SetLike membership classifier transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, membership transport is accepted only as the displayed classifier route*

$$M, Q \longmapsto H, C, P, N.$$

Here M *is the finite* BHist *membership source,* Q *is the membership classifier,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The transported read cannot become host elementhood, an ambient powerset, quotient equality of sets, unrestricted comprehension, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierTransport`

Proof. Project the accepted carrier and its membership subcarrier. The only element-facing rows before the structural tail are M and Q.

Apply the SetLike membership route and classifier transport route. These force every transported membership read to remain a classifier-bounded BHist read before H and C replay it.

Finally use the local NameCert obligations. The rows P and N source and name the same displayed route, so no host elementhood, powerset, quotient equality, unrestricted comprehension, or external satisfaction coordinate is available. □

Theorem 2703.18 (SetLike NameCert ledger nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *ledger records membership, subset implication, restricted comprehension, and extensional-boundary obligations only through*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The ledger is a finite BHist *route stable under* hsame *and* Cont*; it cannot export an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a host universe outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeNameCertLedgerNonescape`

Proof. Read the NameCert obligations of Definition 3668.10. They expose exactly the carrier rows $M, Q, I, R, E, H, C, P, N$.

Use subset classifier exactness, restricted-comprehension membership factorization, and extensional-subset transport. These keep subset, comprehension, and extensional comparison on the same membership classifier before structural replay.

The structural tail is exhausted by hsame transport, Cont replay, provenance, and local naming. Therefore the ledger cannot escape into powersets, unrestricted separation, quotient equality, selected representatives, or a host universe. □

Theorem 2703.19 (SetLike type-model satisfaction boundary)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must first expose* $M, Q, I, R, E, H, C, P, N$ *as a finite* BHist *route with componentwise* hsame*, displayed* Cont*, provenance, and local* NameCert *rows. Type, syntax, and satisfaction reads remain downstream of this displayed SetLike route and cannot replace it by host membership, quotient equality of sets, unrestricted comprehension, an ambient universe, or an external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeModelSatisfactionBoundary`

Proof. Start with the SetLike TypeLike and model-theory handoff. It already requires family, syntax, and satisfaction consumers to pass through the membership carrier and classifier route.

Apply membership classifier transport and NameCert ledger nonescape. Together they expose $M, Q, I, R, E, H, C, P, N$ and block any replacement by host membership or a powerset-style carrier.

Finally use the FirstOrder signature and ModelTheory satisfaction root-unblock theorems. A typed or model-theoretic consumer that stays on the displayed route receives only the finite BEDC packet, while a consumer that leaves it lacks the required SetLike source. □

Theorem 2703.20 (SetLike family handoff obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{TypeLike}^{\uparrow}$ *family consumer that reads indexed fibers from* S *must expose the membership source* M*, classifier row* Q*, subset implication row* I*, restricted-comprehension row* R*, extensional boundary row* E*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*. The family handoff cannot replace these rows by an ambient powerset, unrestricted separation principle, host element relation, quotient equality of sets, or external universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyHandoffObligation`

Proof. Project the accepted carrier of Definition 3668.1. The family consumer sees only the displayed rows $M, Q, I, R, E, H, C, P, N$ before it can build a TypeLike fiber read.

Use Definition 3668.16. It forces sibling family reads through the membership carrier and the classifier route, so M, Q and Q, I, R, E remain visible before the consumer proceeds.

Finally apply Definition 3668.26 and Definition 3668.52. These keep the restricted family row bounded and the comparison row tied to the displayed classifier; hence no ambient powerset, unrestricted separation, quotient set equality, host element relation, or external universe is exported. □

Theorem 2703.21 (SetLike satisfaction handoff obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer must obtain its membership and comprehension base through the concrete route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The satisfaction handoff exports no host satisfaction predicate, ambient powerset, unrestricted separation rule, quotient equality of sets, selected representative, or external universe outside those nine rows.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSatisfactionHandoffObligation`

Proof. Start with Definition 3668.16. Satisfaction-facing reads are downstream of the SetLike membership and classifier surface, not independent model-theoretic coordinates.

Apply Definition 3668.54. First-order syntax may consume membership, subset implication, restricted comprehension, and extensional comparison only after the carrier route has been displayed.

Then use Definition 3668.89. The ModelTheory satisfaction consumer inherits the same classifier-bounded envelope through FirstOrder syntax, so host satisfaction, ambient powersets, unrestricted separation, quotient equality, selected representatives, and external universes remain outside the route. □

Theorem 2703.22 (SetLike family fiber obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer accepted from* S *must expose the membership source* M*, classifier row* Q*, subset implication row* I*, restricted-comprehension row* R*, extensional boundary row* E*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *before any* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer reads satisfaction or family membership. The fiber route cannot replace those rows by a host powerset, unrestricted separation principle, quotient equality of sets, selected representative, external satisfaction predicate, or ambient universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberObligations`

Proof. Project the accepted carrier of Definition 3668.1. The family-fiber consumer has access only to $M, Q, I, R, E, H, C, P, N$ before it can form a typed fiber read.

Use the SetLike family handoff obligation. It forces indexed fibers through the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows.

Finally compose with the satisfaction handoff obligation and the type-model satisfaction boundary. FirstOrder and ModelTheory consumers inherit the same visible rows from the TypeLike fiber route, so satisfaction and family membership cannot be read from an external powerset, host element relation, quotient set equality, selected representative, unrestricted separation rule, or ambient universe. □

Theorem 2703.23 (SetLike family-fiber model-satisfaction route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer that feeds a* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction read must expose the rows* $M, Q, I, R, E, H, C, P, N$ *before satisfaction or family membership is inspected. The route cannot be replaced by host membership, quotient equality of sets, unrestricted comprehension, selected representatives, or an external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberModelSatisfactionRoute`

Proof. Start with the family handoff obligation. It opens the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows before a TypeLike family fiber is available.

Next compose with the satisfaction handoff obligation and the type-model satisfaction boundary. These force FirstOrder and ModelTheory satisfaction reads to inherit the same displayed SetLike route rather than introducing a separate model-theoretic carrier.

Finally apply the family-fiber obligations above. Since the fiber route already blocks host powersets, unrestricted separation, quotient equality, selected representatives, and external universes, the satisfaction-facing consumer has no coordinate from which host membership or an external satisfaction predicate can be read. □

Theorem 2703.24 (SetLike family-fiber FirstOrder ModelTheory lock)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer that feeds a* $\mathsf{FirstOrder}^{\uparrow}$ *syntax read or a* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction read must expose exactly the rows* $M, Q, I, R, E, H, C, P, N$ *from the accepted packet. The lock cannot replace those rows by host membership, unrestricted comprehension, quotient set equality, selected representatives, external satisfaction, or ambient universe data.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberFirstorderModeltheoryLock`

Proof. Start with Definition 3668.23. It already routes every family-fiber satisfaction read through the displayed SetLike packet before any FirstOrder or ModelTheory consumer inspects membership or satisfaction.

Apply Definition 3668.20. A TypeLike family-fiber consumer receives the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows from S, not from a host membership relation or ambient family carrier.

Next apply Definition 3668.21. FirstOrder syntax and ModelTheory satisfaction inherit the same route, so no external satisfaction predicate, unrestricted comprehension rule, quotient set equality, selected representative, or universe datum is available as a replacement.

Finally use Definition 3668.18. The local ledger names only $M, Q, I, R, E, H, C, P, N$ and exports no coordinate outside the accepted packet, so the cross-consumer route is locked to those nine rows. □

Theorem 2703.25 (SetLike extensionality refusal obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensional row* E *is a classifier-boundary row over the displayed membership route* M, Q, I, R *and the structural route* H, C, P, N*. A downstream TypeLike, FirstOrder, or ModelTheory consumer may cite* E *only as that boundary row, not as host set extensionality, quotient equality, a powerset equality principle, unrestricted separation, or an external universe equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalityRefusalObligation`

Proof. Project the row E from Definition 3668.1. It is the carrier's extensionality-boundary coordinate, not a host equality object.

Use Definition 3668.62 and Definition 3668.71. Together they route every extensional read through $M, Q, I, R, E, H, C, P, N$ and rule out quotient equality, powersets, unrestricted separation, host set equality, and external inclusion predicates.

Finally apply Definition 3668.16. Since TypeLike, FirstOrder, and ModelTheory consumers must receive SetLike data through the same membership and classifier surface, they cannot reinterpret E as ambient extensionality or as an equality principle outside the concrete carrier rows. □

Theorem 2703.26 (SetLike bounded comprehension nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a restricted-comprehension read through* R *cannot export a powerset, unrestricted separation principle, host subset object, or universe of all subsets. It remains a bounded finite* BHist *row over* M, Q, I *and is accepted only together with* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionNonescape`

Proof. Project the restricted-comprehension coordinate R from Definition 3668.1. The carrier describes R as a row admitting only classifiers already routed through M, Q, I.

Use Definition 3668.59 and Definition 3668.66. Both results keep the comprehension read inside the shared membership classifier and the subset implication route.

The structural rows H, C, P, N preserve and name that finite row. Since the packet has no coordinate for a powerset, unrestricted separation rule, host subset object, or universe of all subsets, restricted comprehension cannot export any of them. □

Theorem 2703.27 (SetLike membership route)**.** *Every downstream consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that reads membership, subset implication, restricted comprehension, or extensional comparison must factor through the finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Consequently sibling reads by $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *may use* $\mathsf{SetLike}^{\uparrow}$ *only through the carried membership route, and cannot replace it by an ambient powerset, unrestricted comprehension principle, host quotient of sets, or external satisfaction relation.*

Proof. Begin with an accepted carrier $S = (M, Q, I, R, E, H, C, P, N)$. A membership consumer can reach an element-in-set assertion only by reading the source row M and the classifier row Q.

Subset and comprehension reads then remain inside the same boundary. The row I supplies finite implication between displayed membership classifiers, and R admits restricted comprehension only when the classifier has already passed through M, Q, I.

Extensional reads are likewise bounded. The row E permits comparison only through the displayed membership classifier, so it does not introduce quotient equality or an ambient host set identity.

Finally replay the structural rows. The rows H, C, P, N preserve, replay, source, and name the same finite route, while $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ can cite the packet only through P and C. No sibling read therefore bypasses M, Q, I, R, E. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipRoute`

Theorem 2703.28 (SetLike kernel membership obligation)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *that reads membership must expose* M *as the* BHist *membership source and* Q *as the displayed membership classifier before any TypeLike, FirstOrder, or ModelTheory row is consumed. The read remains valid only through componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeKernelMembershipObligation`

Proof. Project the carrier and the membership carrier. Their membership-facing coordinates are exactly M, Q, H, C, P, N.

Apply the membership route theorem. It forces every downstream membership read through the displayed BHist source and classifier before sibling consumers can use the packet.

The structural tail is then read by projection. The rows H, C, P, N transport, replay, source, and locally name the same membership surface, so no host element relation, powerset row, or unrecorded classifier is introduced. □

Definition 2703.29 (SetLike root membership carrier)**.** *The root membership carrier of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet* $S = (M, Q, I, R, E, H, C, P, N)$ *is the finite* BHist *route*

$$M \longrightarrow Q \longrightarrow I \longrightarrow R \longrightarrow E,$$
$$H \longrightarrow C \longrightarrow P \longrightarrow N.$$

Here M *is the displayed membership source,* Q *is the classifier-based membership row,* I *records subset as implication of membership over* Q*,* R *records restricted comprehension only after* M, Q, I *are visible,* E *records the extensional boundary for the restricted result,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeRootMembershipCarrier`
Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipCarrier route readback`

Theorem 2703.30 (SetLike root membership transport)**.** *Every root membership read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet transports only the classifier-based membership surface of Definition 3668.29. A consumer that moves the read along* hsame *must replay* M, Q, I, R, E *through the displayed* Cont *row, preserve the* Pkg *provenance, and expose the same local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows before any TypeLike, FirstOrder, ModelTheory, or root-unblock consumer may use the membership judgment.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipTransport`

Proof. Project the carrier of Definition 3668.29. The only membership row is Q over the displayed BHist source M, while subset and comprehension rows are downstream of that classifier.

Use the existing membership, subset, and comprehension obligations in this surface. They keep membership reads on M, Q, subset reads on the implication row I, and restricted-comprehension reads on R, E.

The structural rows then provide the transport discipline. Componentwise hsame transport, Cont replay, Pkg provenance, and the local NameCert row preserve the same route, so no host powerset, quotient set equality, unrestricted separation principle, or carrier-equivalence replacement is introduced. □

Theorem 2703.31 (SetLike root subset ledger non-escape)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet, every subset ledger exported to a root consumer is the implication-of-membership row* I *from Definition 3668.29. The export must display classifier-based membership* M, Q*, restricted-comprehension boundary rows* R, E *when comprehension is read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows; it cannot escape to host inclusion, powerset equality, quotient set equality, or an unrestricted subset object.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootSubsetLedgerNonescape`

Proof. Read the subset coordinate I. Its source and target are classifier memberships over M, Q, so the ledger records implication of membership rather than an ambient inclusion relation.

When a comprehension consumer is present, read R only through the same M, Q, I route and then attach E as the displayed extensional boundary. This keeps restricted comprehension subordinate to the classifier ledger.

Finally project H, C, P, N. These rows transport, replay, source, and locally name the finite subset route, leaving no carrier coordinate for host inclusion, powerset equality, quotient equality, or unrestricted separation. □

Theorem 2703.32 (SetLike root comprehension scope)**.** *Every root comprehension consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must treat comprehension as the restricted row* R *scoped by classifier-based membership* M, Q *and subset implication* I*. The consumer may read the extensional boundary* E*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*, but it receives no unbounded predicate former, no host powerset, no quotient equality of sets, and no carrier-equivalence transport outside the displayed root route.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootComprehensionScope`

Proof. Start with the restricted-comprehension obligation already present in the child surface. It places R after the membership source, classifier, and subset implication rows.

Attach the extensional boundary E. The boundary describes the displayed restricted result only, and it remains tied to the same classifier-based membership route.

The structural tail H, C, P, N is then read without adding a new classifier or carrier. Thus the comprehension scope is exactly the finite SetLike root route and not an unrestricted separation, host powerset, quotient-set equality, or external subset object. □

Theorem 2703.33 (SetLike subset classifier obligation)**.** *Every accepted subset-classifier read of a* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must expose the* BHist *membership source* M*, membership classifier* Q*, subset implication row* I*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. It cannot replace the implication row by host inclusion, carrier-equivalence transport, quotient equality of sets, or an ambient powerset predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetClassifierObligation`

Proof. Start with the subset row I in the carrier. It is an implication over the displayed classifier Q and membership source M, not an external inclusion relation.

Use subset ledger exactness and classifier transport. These keep subset reads on the same M, Q, I route while H and C replay the displayed classifier rather than transporting across a different carrier.

Finally read P and N. Provenance and local naming expose the same finite packet to downstream consumers, so the subset classifier has no coordinate for powersets, quotient equality, host inclusion, or carrier-equivalence replacement. □

Theorem 2703.34 (SetLike comprehension ledger obligation)**.** *Every restricted-comprehension ledger read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must expose* $M, Q, I, R, E, H, C, P, N$ *as one finite* BHist *route. The row* R *is admitted only after the membership source, classifier, and subset implication rows are visible; the row* E *records the resulting extensional boundary;* H, C, P, N *preserve, replay, source, and locally name that ledger.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeComprehensionLedgerObligation`

Proof. Project the carrier to the comprehension row R. By the restricted-comprehension factorization theorem, R is bounded by M, Q, I before the structural rows are read.

Attach the extensional boundary row E. The extensional-boundary route keeps comparison of the displayed comprehension result inside the same classifier ledger.

The final projection reads H, C, P, N. Componentwise transport, continuation replay, provenance, and local NameCert expose the ledger as one finite packet, with no powerset, unrestricted separation principle, host subset object, quotient set equality, or carrier-equivalence transport. □

Theorem 2703.35 (SetLike root extensional boundary)**.** *Every extensionality read exported by the* $\mathsf{SetLike}^{\uparrow}$ *root route is bounded by the displayed membership source* M*, classifier* Q*, subset row* I*, restricted-comprehension row* R*, extensional boundary* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. The export gives no host powerset equality, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootExtensionalBoundary`

Proof. Project the root membership carrier of Definition 3668.29. Its extensional-facing row is E, and E is available only after the membership source, classifier, subset implication, and restricted-comprehension rows have been displayed.

Use the subset and comprehension obligations. The subset ledger keeps comparison on I over M, Q, while the comprehension ledger places R behind the same membership route and then attaches E as the boundary for the displayed restricted result.

Finally read H, C, P, N. These rows transport, replay, source, and locally name the same finite route, so the exported extensionality is the bounded root boundary rather than host powerset equality, quotient equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of carriers. □

Theorem 2703.36 (SetLike root finite route package)**.** *Every FirstOrder, ModelTheory, TypeLike, or root logic consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must read*

$$M, Q, I, R, E, H, C, P, N$$

as one finite BHist *route. The package exposes classifier-based membership, subset-as-implication, restricted comprehension, and the extensional boundary only through the displayed carrier of Definition 3668.29; it gives no host powerset, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootFiniteRoutePackage`

Proof. Start with Definition 3668.30. It places every root membership read on M, Q and keeps downstream consumers inside the displayed transport, replay, provenance, and local naming rows.

Next apply Definition 3668.31 and Definition 3668.32. The subset ledger is the implication row I over the classifier membership surface, and the comprehension row R is available only after M, Q, I have been displayed.

Finally apply Definition 3668.35. The extensional row E remains tied to the same restricted-comprehension route, while H, C, P, N preserve the finite packet for logic consumers. Since the existing root obligations already exclude host powersets, quotient set equality, unrestricted separation, carrier-equivalence replacement, and ambient carrier equality, the packaged consumer boundary excludes them as well. □

Theorem 2703.37 (SetLike root type-model membership factorization)**.** *Every TypeLike, FirstOrder, and ModelTheory consumer of the* $\mathsf{SetLike}^{\uparrow}$ *root route factors membership through* M, Q*, subset through* I*, restricted comprehension through* R, E*, and structural preservation through* H, C, P, N *before any logic-facing judgment is exported. The factorization gives no host powerset, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootTypeModelMembershipFactorization`

Proof. Start with the membership transport theorem. It forces each membership read to expose the displayed source M and classifier Q before a TypeLike, FirstOrder, or ModelTheory consumer can use the judgment.

Next apply subset non-escape and comprehension scope. The subset read is the implication row I over classifier membership, while restricted comprehension is admitted only through R together with the extensional boundary E.

Use the root extensional boundary and finite-route package. They keep the resulting logic-facing judgment inside M, Q, I, R, E and preserve it only through componentwise hsame transport, displayed Cont replay, provenance, and local naming rows H, C, P, N.

The excluded host powerset, quotient equality, unrestricted separation, carrier-equivalence replacement, and ambient carrier equality are not coordinates of this route. Hence no logic or model-theory consumer can export them from the root packet. □

Theorem 2703.38 (SetLike membership classifier non-escape)**.** *Every TypeLike, FirstOrder, and ModelTheory consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *root packet must keep membership on the displayed source and classifier rows* M, Q *and must preserve the structural rows* H, C, P, N *before exporting any logic-facing judgment. The consumer cannot replace classifier membership by host elementhood, powerset membership, quotient equality, carrier-equivalence transport, or an ambient set equality outside the root route.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierNonescape`

Proof. First apply the root membership transport theorem. It keeps every membership read on the displayed BHist source M and classifier row Q before any logic consumer may use the judgment.

Next use subset ledger non-escape and comprehension scope. The subset row remains the implication ledger I over classifier membership, while restricted comprehension is admitted only through R, E after the same M, Q, I route has been displayed.

Finally apply the type-model membership factorization theorem. It packages TypeLike, FirstOrder, and ModelTheory reads through $M, Q, I, R, E, H, C, P, N$, so the structural rows preserve the classifier route rather than transporting it to host elementhood, powerset membership, quotient equality, carrier-equivalence replacement, or ambient set equality. □

Theorem 2703.39 (SetLike membership classifier transport boundary)**.** *Every TypeLike, FirstOrder, or ModelTheory consumer that transports membership from an accepted* $\mathsf{SetLike}^{\uparrow}$ *root packet* $S = (M, Q, I, R, E, H, C, P, N)$ *must preserve the displayed route*

$$M, Q, I, R, E, H, C, P, N.$$

The transported read cannot replace Q *by host elementhood, powerset membership, quotient set equality, carrier-equivalence transport, or ambient set equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierTransportBoundary`

Proof. Start with the membership transport and membership classifier non-escape theorems. They keep the membership source on the displayed BHist row M and the classifier on Q before any TypeLike, FirstOrder, or ModelTheory consumer can read the judgment.

Next apply subset non-escape and restricted comprehension scope. The subset row is still the displayed implication row I, and the comprehension route remains confined to R, E over the same M, Q, I prefix.

Finally use the root extensional boundary and type-model membership factorization theorem. These force transport to proceed through H, C, P, N after the displayed classifier route has been exposed, so none of the excluded host elementhood, powerset, quotient, carrier-equivalence, or ambient set-equality routes is available as a replacement for Q. □

Theorem 2703.40 (SetLike root membership obligation)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that attempts to read membership must display the route*

$$M, Q, H, C, P, N$$

before the membership judgment is available. The source row M *supplies the carried membership surface,* Q *supplies the displayed membership classifier, and* H, C, P, N *transport, replay, source, and name the same route. The read exports no ambient powerset, unrestricted subset object, host equality of sets, quotient set equality, or universe-level membership relation.*

Proof. Project the accepted carrier of Definition 3668.1. The only membership-facing source row is M, and the only classifier row attached to that source is Q.

Use Definition 3668.27. Any downstream membership read must factor through the displayed finite route before sibling consumers may cite the packet.

Finally read the structural rows. The rows H, C, P, N preserve, replay, source, and locally name the same membership surface, so the root consumer cannot replace it by a host powerset, quotient set equality, or universe-level membership relation. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipObligation`

Theorem 2703.41 (SetLike bounded comprehension family handoff)**.** *Every restricted-comprehension consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet reads a bounded family only through*

$$M, Q, I, R.$$

The membership source M*, displayed classifier* Q*, implication ledger* I*, and restricted-comprehension row* R *together provide the bounded family handoff. The handoff does not supply unrestricted separation, a powerset carrier, quotient equality of subsets, selected representatives, or an external comprehension principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionFamilyHandoff`

Proof. Start from the restricted-comprehension row R in Definition 3668.1. It is a displayed packet coordinate and not a host comprehension operator.

Apply Definition 3668.53. The bounded family consumed by R is visible only after the membership source, classifier, and implication rows M, Q, I have been read.

The structural preservation clauses then keep the same bounded route under transport and replay. Since no row stores unrestricted separation, powersets, quotient subset equality, selected representatives, or an external comprehension principle, the consumer receives only the bounded family handoff. □

Theorem 2703.42 (SetLike TypeLike family nonescape)**.** *A* $\mathsf{TypeLike}^{\uparrow}$ *family consumer that cites a* $\mathsf{SetLike}^{\uparrow}$ *packet may use only the displayed rows*

$$M, Q, I, R, E, H, C, P, N$$

and cannot reinterpret the SetLike membership or family rows as a host universe, an ambient type of all subsets, quotient set equality, or external equality of family fibers.

Proof. Begin with the sibling read through the package row P. The $\mathsf{TypeLike}^{\uparrow}$ consumer receives a cited $\mathsf{SetLike}^{\uparrow}$ packet, not a replacement carrier for the host universe.

Apply Definition 3668.27 and Definition 3668.53. Membership, implication, bounded comprehension, and extensional comparison all remain inside M, Q, I, R, E before the structural rows are replayed.

Finally use H, C, P, N to transport, replay, source, and name that same route. These rows do not add a universe classifier, a powerset object, quotient set equality, or external equality of family fibers, so the TypeLike family read cannot escape the SetLike certificate surface. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeFamilyNonescape`

Theorem 2703.43 (SetLike root membership family exhaustion)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that combines membership reads, bounded-comprehension family reads, and* $\mathsf{TypeLike}^{\uparrow}$ *family reads is exhausted by the displayed rows*

$$M, Q, I, R, E, H, C, P, N.$$

The combined route supplies membership, the bounded implication and comprehension handoff, extensional comparison, componentwise transport, continuation replay, package provenance, and local naming. It supplies no powerset, unrestricted separation, host universe, quotient set equality, selected representatives, or external family equality.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipFamilyExhaustion`

Proof. First apply Definition 3668.40. A root membership consumer must expose M, Q, H, C, P, N before it can read membership, and that route has no ambient powerset or host equality field.

Next apply Definition 3668.41. A restricted-comprehension consumer may add only the implication and bounded-comprehension rows I, R to the same displayed membership route.

Finally apply Definition 3668.42. A $\mathsf{TypeLike}^{\uparrow}$ family read may cite the SetLike packet only through $M, Q, I, R, E, H, C, P, N$ and cannot reinterpret those rows as a host universe, quotient set equality, selected representatives, or external equality of family fibers.

The three consumer routes therefore share the same finite row inventory. Their union is exactly $M, Q, I, R, E, H, C, P, N$, so no further family or set-forming coordinate is available to the root consumer. □

Definition 2703.44 (SetLike root carrier membership)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet* $S = (M, Q, I, R, E, H, C, P, N)$*, the root carrier membership read is the certified family route*

$$\begin{gathered} M \longrightarrow Q \longrightarrow I \longrightarrow R \longrightarrow E, \\ H \longrightarrow C \longrightarrow P \longrightarrow N. \end{gathered}$$

The row M is the finite BHist *membership source, Q is the classifier row over that source, I is subset implication between certified membership rows, R is bounded comprehension after the membership and subset rows are visible, E is the displayed extensional boundary, and H, C, P, N transport, replay, source, and locally name the same certified family packet.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeRootCarrierMembership`
Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootCarrierMembership route closed`

Theorem 2703.45 (SetLike root classifier transport)**.** *Every* hsame *or* Cont *transport of the root carrier membership read of Definition 3668.44 preserves the displayed membership classifier. A TypeLike, FirstOrder, ModelTheory, or root consumer that reads the transported packet receives $M, Q, I, R, E, H, C, P, N$ as one certified family route, and receives no host membership relation, ambient powerset, quotient equality of sets, selected representative, or external classifier.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootClassifierTransport`

Proof. Project the route of Definition 3668.44. The membership read is based on M and Q, while I, R, E are downstream ledger rows over the same classifier.

Use componentwise hsame transport and displayed Cont replay. These rows move the finite packet without replacing Q by a host relation or by a classifier over a different source.

Finally read P and N. Provenance and local naming expose the same certified family packet to downstream consumers, so TypeLike, FirstOrder, ModelTheory, and root consumers see only the transported SetLike route. □

Theorem 2703.46 (SetLike root NameCert obligations)**.** *The root* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *obligations for membership consumers are exactly carrier admission for Definition 3668.44, classifier transport for M, Q, bounded-comprehension ledger exposure for I, R, E, componentwise* hsame *and* Cont *preservation through H, C, provenance and local naming through P, N, and the boundary excluding host extensionality. Any root consumer that imports host set equality, ambient powersets, unrestricted separation, selected representatives, or an external membership classifier is outside the accepted packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootNameCertObligations`

Proof. Start with carrier admission. The definition presents the only membership carrier available to a root consumer: the finite route $M, Q, I, R, E, H, C, P, N$.

Apply classifier transport. It keeps membership reads on M, Q and makes H, C preserve the displayed classifier rather than a host membership relation.

Use the bounded-comprehension ledger rows. The subset and comprehension coordinates I, R, E are read only after the membership classifier is visible, so comprehension remains a finite BEDC ledger.

Finally attach provenance and local naming. Since P, N name only the displayed route, the accepted NameCert surface has no coordinate for host extensionality, powersets, unrestricted separation, selected representatives, or external classifiers. □

Theorem 2703.47 (SetLike membership classifier root unblock)**.** *Every downstream membership read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet exposes the membership source M, the classifier row Q, the transport row H, the replay row C, the provenance row P, and the local naming row N before the read may be consumed by a root dependency. The read therefore reaches TypeLike, FirstOrder, ModelTheory, or another root consumer only as the displayed finite membership classifier route, not as host membership, ambient powerset inclusion, unrestricted separation, quotient equality, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierRootUnblock`

Proof. Project Definition 3668.44. The membership source and classifier are exactly M and Q, and no other row supplies a membership predicate.

Apply Definition 3668.27 and Definition 3668.45. They force every membership-facing read through the displayed carrier route and preserve the classifier under hsame transport and Cont replay.

Finally read the structural rows H, C, P, N. Provenance and local naming expose the same finite route to downstream consumers, so the root dependency cannot replace it by host membership, powersets, unrestricted separation, quotient equality, or external satisfaction. □

Theorem 2703.48 (SetLike subset comprehension root unblock)**.** *Every root read of subset implication or restricted comprehension for an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must pass through the finite route* M, Q, I, R, H, C, P, N*:* M, Q *expose the displayed membership classifier,* I *exposes subset implication between those classified rows,* R *exposes restricted comprehension only after* M, Q, I *are visible, and* H, C, P, N *transport, replay, source, and locally name the same packet. The route exports no ambient powerset, unrestricted comprehension principle, host subset relation, selected representative, or quotient equality of sets.*

Proof. Start with the membership route of Definition 3668.27. Any subset or comprehension read is downstream of the displayed membership classifier M, Q.

Use the rows I and R from Definition 3668.44. The implication row compares certified membership rows, and the restricted-comprehension row is available only after that comparison has been displayed.

Finally apply Definition 3668.46. The structural rows H, C, P, N keep the same finite route available to root consumers and leave no coordinate for ambient powersets, unrestricted comprehension, host subset relations, selected representatives, or quotient equality. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetComprehensionRootUnblock`

Theorem 2703.49 (SetLike extensionality root unblock)**.** *Every extensional comparison exported by an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet is only the displayed classifier boundary* E *over the membership rows* M, Q *with support from* H, C, P, N*. A downstream root consumer may read that boundary as the packet's extensional comparison route, but it may not identify it with host set equality, quotient equality of sets, selected representatives, an ambient inclusion predicate, or an external extensionality principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalityRootUnblock`

Proof. Project the extensional boundary of Definition 3668.44. The row E is the only comparison coordinate, and it is tied to the same displayed membership source M and classifier Q.

Use the SetLike membership route and the root NameCert obligations. They make E available only after the carrier rows have exposed the membership classifier and the structural preservation rows.

Thus a root consumer receives an extensional boundary for the accepted packet, not host set equality. The rows H, C, P, N preserve and name that boundary without adding quotient equality, selected representatives, ambient inclusion, or external extensionality. □

Theorem 2703.50 (SetLike root membership provenance exhaustion)**.** *Every TypeLike, FirstOrder, ModelTheory, or root consumer of accepted* $\mathsf{SetLike}^{\uparrow}$ *membership must exhaust the finite route* $M, Q, I, R, E, H, C, P, N$ *before exporting membership, subset, comprehension, or extensionality rows. In particular, the provenance row* P *and the local naming row* N *are required before any of the displayed membership classifier, subset implication, restricted-comprehension, or extensional boundary rows may leave the root packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipProvenanceExhaustion`

Proof. Project Definition 3668.44. The only accepted membership carrier is the displayed route $M, Q, I, R, E, H, C, P, N$, with P and N attached to the same packet as the membership and comparison rows.

Apply Definition 3668.45 and Definition 3668.46. These results keep the membership classifier on M, Q under hsame transport and Cont replay, while requiring provenance and local naming before downstream reads can treat the route as a NameCert surface.

Use Definition 3668.47, Definition 3668.48, and Definition 3668.49. The membership, subset, comprehension, and extensionality exports are exactly the rows M, Q, I, R, E supported by H, C, P, N.

Thus any consumer that omits P or N has not exhausted the root route. The omitted coordinate would be precisely where host membership, powersets, unrestricted separation, selected representatives, quotient set equality, or external satisfaction could enter, so the accepted packet refuses that export. □

Theorem 2703.51 (SetLike root membership nonescape ledger)**.** *Every TypeLike, FirstOrder, ModelTheory, or root consumer that has exhausted the accepted* $\mathsf{SetLike}^{\uparrow}$ *route* $M, Q, I, R, E, H, C, P, N$ *can export only the membership, subset, restricted-comprehension, and extensional-boundary rows already displayed in the SetLike packet. Such a consumer cannot introduce host membership, ambient powersets, unrestricted separation, quotient equality, selected representatives, or external satisfaction.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipNonescapeLedger`

Proof. Start from Definition 3668.44. The carrier route exposes exactly the membership source M, classifier Q, subset row I, restricted-comprehension row R, extensional boundary E, transport H, replay C, provenance P, and local naming N.

Apply the root classifier and NameCert rows, Definition 3668.45 and Definition 3668.46. They keep the membership classifier on M, Q and confine the accepted NameCert surface to the displayed finite route.

Use the three root-unblock exports: Definition 3668.47, Definition 3668.48, and Definition 3668.49. These are exactly the membership, subset, restricted-comprehension, and extensional-boundary rows available to downstream root consumers.

Finally use provenance exhaustion from Definition 3668.50. Once P, N have been read, no remaining carrier coordinate can supply host membership, ambient powersets, unrestricted separation, quotient equality, selected representatives, or external satisfaction. Hence the ledger is exhausted by the displayed SetLike packet. □

Theorem 2703.52 (SetLike extensional boundary classifier lock)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an extensional comparison read must expose the membership source* M*, the displayed classifier* Q*, the extensionality-boundary ledger* E*, and the structural rows* H, C, P, N *before any downstream consumer treats the comparison as available. The read is locked to the displayed classifier and cannot be replaced by host set equality, quotient equality, selected representatives, or an external inclusion predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryClassifierLock`

Proof. Project the carrier of Definition 3668.1. The extensional read is the row E, and its only membership data are the source row M and classifier row Q.

Use Definition 3668.27. The route forces any membership-facing consumer to expose M, Q, I, R, E before the structural rows are replayed.

Finally apply the local carrier rows H, C, P, N. Transport, replay, provenance, and naming keep E attached to the same displayed classifier, so the comparison is not host set equality, quotient equality, selected representatives, or external inclusion. □

Theorem 2703.53 (SetLike bounded comprehension transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, restricted-comprehension transport remains inside the finite route*

$$M, Q, I, R, H, C, P, N.$$

It may read a bounded family only after the membership source, classifier, and subset implication rows are visible, and componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming preserve that bounded route without importing powersets, unrestricted separation, quotient equality, or an external comprehension principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionTransport`

Proof. Project the restricted-comprehension row R from Definition 3668.1. The row is part of the accepted finite packet and is not an ambient comprehension operator.

Apply Definition 3668.27. The bounded family consumed by R must already pass through M, Q, I before comprehension data are visible.

Transport is then componentwise. The rows H, C, P, N preserve, replay, source, and name the same bounded route, so the transported comprehension remains inside M, Q, I, R, H, C, P, N and cannot introduce the excluded host-level surfaces. □

Theorem 2703.54 (SetLike first-order membership handoff)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *consumer that receives membership data from an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet receives it only through the displayed finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Thus first-order syntax may consume SetLike membership, subset implication, restricted comprehension, and extensional comparison only after the carrier route has been displayed, and it cannot import an ambient powerset, unrestricted comprehension principle, quotient equality of sets, selected representatives, or external satisfaction relation.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFirstOrderMembershipHandoff`

Proof. Start with Definition 3668.27. It states the exact finite route available to downstream $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ consumers.

Project the carrier of Definition 3668.1. The rows M, Q, I, R, E are the only membership, subset, comprehension, and extensional comparison rows available before structural replay.

Finally replay H, C, P, N. These rows transport, replay, source, and name the same finite membership route for the $\mathsf{FirstOrder}^{\uparrow}$ consumer, so first-order syntax receives no independent powerset, unrestricted comprehension, quotient equality, selected representative, or external satisfaction surface. □

Theorem 2703.55 (SetLike finite-family membership route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a finite family of membership reads accepted by* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumers is routed componentwise through the same displayed packet*

$$M, Q, I, R, E, H, C, P, N.$$

Each family member first reads the BHist *membership source M and classifier Q, then consumes subset implication I, bounded comprehension R, and the extensional boundary E before the shared transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows may be used. The family route exports no ambient powerset, unrestricted separation, quotient equality of sets, selected representative, host elementhood relation, or external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFiniteFamilyMembershipRoute`

Proof. Project the accepted carrier and read the family one displayed member at a time. Carrier admission supplies the same finite rows $M, Q, I, R, E, H, C, P, N$ for every member, with no additional row indexed by the family.

Apply the membership route and the bounded-comprehension transport theorem. They force every member read to enter through M, Q and to use I, R only as bounded rows over that same classifier.

Finally use the extensional boundary classifier lock. The row E compares only displayed SetLike data, while H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no powerset, unrestricted separation, quotient-set equality, selector, host elementhood, or satisfaction row, a finite family cannot export one. □

Theorem 2703.56 (SetLike TypeLike consumer handoff)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, a* $\mathsf{TypeLike}^{\uparrow}$ *consumer receives SetLike membership as the classifier-bounded route M, Q together with the bounded-comprehension rows I, R, E and the structural tail H, C, P, N. The handoff preserves both the displayed membership classifier and the bounded-comprehension ledger; it cannot replace them by a host type predicate, an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a global elementhood oracle.*

Proof. Start with the membership route for downstream TypeLike, FirstOrder, and ModelTheory consumers. It gives the TypeLike consumer access to SetLike data only after the membership source, classifier, subset, comprehension, extensional, transport, replay, provenance, and local naming rows have been displayed.

Use the extensional boundary classifier lock to keep Q attached to the same source row M. Then use bounded-comprehension transport to keep I, R, E attached to that classifier route rather than an ambient type predicate or powerset.

The structural rows H, C, P, N complete the handoff by transporting, replaying, sourcing, and naming the same finite packet. Because no host type predicate, powerset, unrestricted separation, quotient equality, selector, or global elementhood coordinate occurs in the accepted carrier, the TypeLike consumer receives only the preserved SetLike route. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeConsumerHandoff`

Theorem 2703.57 (SetLike ModelTheory satisfaction boundary)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, a* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer may read SetLike data only through the FirstOrder-facing finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The boundary supplies displayed membership, classifier-bounded subset and comprehension rows, extensional comparison, componentwise transport, continuation replay, provenance, and local naming. It refuses external satisfaction predicates, powerset semantics, unrestricted separation, quotient equality of sets, selected representatives, host universes, and completed model-theoretic semantics.

Proof. First route the consumer through the FirstOrder membership handoff. The syntax-facing read exposes $M, Q, I, R, E, H, C, P, N$ before satisfaction-facing data can be consumed.

Next apply the root unblock for ModelTheory satisfaction. It keeps satisfaction downstream of the FirstOrder syntactic carrier and therefore downstream of the displayed SetLike membership classifier and bounded-comprehension ledger.

Finally use the finite-family route for any displayed family of satisfaction reads. Each member still reads the same rows, and the carrier has no external satisfaction predicate, powerset semantics, unrestricted separation, quotient equality, selector, host universe, or completed semantic closure coordinate. The ModelTheory boundary is therefore the finite BEDC packet itself. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeModelTheorySatisfactionBoundary`

Theorem 2703.58 (SetLike subset implication obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an accepted subset read is an implication row* I *over the displayed membership classifier* Q *and the membership-source row* M*. The read is admitted as finite* BHist *data only when the componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N *all keep the same implication surface visible; it cannot import host subset equality or a global inclusion relation outside those rows.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetImplicationObligation`

Proof. Project the carrier and isolate the subset row I. The only membership facts available to I are the source row M and the displayed classifier row Q.

Route the implication through the existing membership classifier. A consumer may read that an accepted membership assertion under one displayed row implies an accepted membership assertion under another, but the route is still the finite M, Q, I ledger.

The structural rows close the boundary. Transport by H, replay by C, provenance by P, and local naming by N preserve the same finite implication row, so no host subset equality or external inclusion predicate enters the packet. □

Theorem 2703.59 (SetLike bounded comprehension obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a comprehension read is admitted only through the restricted-comprehension row* R *bounded by a displayed certified family over* M, Q, I*. The resulting* BHist *row remains tied to componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming* N*, and therefore does not create a powerset or unrestricted separation principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionObligation`

Proof. Begin with the restricted-comprehension coordinate R. By Definition 3668.1, R is not an ambient class former; it is one row of the accepted finite packet.

The bound is supplied by the existing membership route. The source row M, classifier row Q, and implication row I determine which displayed family may be read before R is exposed.

Finally replay the structural rows. The rows H, C, P, N transport, replay, source, and name the same bounded family row, so the comprehension read remains finite BEDC data and never becomes a powerset, unrestricted separation rule, or host universe of subsets. □

Theorem 2703.60 (SetLike extensional boundary obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, extensional comparison is a boundary ledger* E *over displayed membership rows rather than quotient equality of sets. The comparison is accepted as* BHist *data*

only when its membership evidence is routed through M, Q, I, R and its stability is preserved by componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryObligation`

Proof. Project the extensionality row E. It is a boundary ledger inside the carrier, not a quotient constructor and not a host equality of sets.

Use the existing membership classifier to supply the comparison evidence. A displayed extensional read compares membership rows already exposed through M, Q and routed through the subset and bounded-comprehension rows I, R.

The structural rows keep the boundary exact. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed membership ledger, so no quotient equality, representative selection, or ambient set identity is exported. □

Definition 2703.61 (SetLike membership boundary subcarrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the membership-boundary subcarrier is the finite prefix

$$S_{\partial} = (M, Q, I, R, E).$$

It cuts out the membership source, classifier, subset implication, restricted-comprehension row, and extensionality-boundary ledger before transport, replay, provenance, and local naming are consumed. The subcarrier has no coordinate for powersets, unrestricted separation, quotient equality of sets, selected representatives, or host universe membership.

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeMembershipBoundarySubcarrier`

Theorem 2703.62 (SetLike extensional boundary route)**.** *Every extensional read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$ must first factor through the membership-boundary subcarrier $S_{\partial} = (M, Q, I, R, E)$ and then through H, C, P, N. Thus an extensional consumer may compare displayed set rows only after the membership source, classifier, subset implication, restricted-comprehension boundary, extensionality ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row have all remained visible.*

Proof. Project the membership-boundary subcarrier of Definition 3668.61. It exposes exactly M, Q, I, R, E before any structural replay is used.

The extensional comparison is the row E. By Definition 3668.60, the comparison evidence must already be routed through the membership source, classifier, subset implication, and restricted-comprehension rows.

Finally replay the structural rows H, C, P, N. They transport, replay, source, and name the same finite boundary, so an extensional consumer cannot bypass M, Q, I, R, E or replace the displayed boundary by host set equality. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryRoute`

Theorem 2703.63 (SetLike subset implication witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, the implication row I is the only subset witness available over the displayed membership classifier Q and membership source M. Any accepted subset read must keep I attached to M, Q, H, C, P, N and cannot replace it by powersets, unrestricted separation, quotient equality of sets, host subset equality, selected representatives, or an external inclusion predicate.*

Proof. Project the carrier of Definition 3668.1. The subset coordinate is exactly the row I, and the only membership data available to it are the source row M and displayed classifier row Q.

Use the membership route and subset obligation, Definition 3668.27 and Definition 3668.58. These force every subset consumer to read implication as finite carrier data over M, Q, I before the structural rows are replayed.

Finally replay through H, C, P, N. Since the carrier has no powerset, unrestricted separation, quotient-equality, selected-representative, host-subset, or external-inclusion coordinate, I is the only subset witness exposed by the packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetImplicationWitness`

Theorem 2703.64 (SetLike bounded comprehension witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the restricted-comprehension row* R *is bounded by the displayed membership source* M*, classifier* Q*, and subset-implication row* I*. The row* R *witnesses only bounded comprehension inside that carrier route and cannot export powersets, unrestricted separation, quotient equality of sets, selected representatives, host subset objects, or an external comprehension principle.*

Proof. Project the carrier of Definition 3668.1. The comprehension coordinate is the row R, and its definition admits only classifiers already routed through M, Q, I.

Apply bounded-comprehension obligation and restricted-comprehension factorization, Definition 3668.59 and Definition 3668.75. They identify M, Q, I as the required bound before R may be read.

The structural rows H, C, P, N preserve and name that same bounded route. Because no other carrier row stores powersets, unrestricted separation, quotient equality, selected representatives, host subset objects, or external comprehension, R remains only the displayed bounded-comprehension witness. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionWitness`

Theorem 2703.65 (SetLike extensional boundary witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensionality row* E *is only a classifier boundary over the displayed membership route* M, Q, I, R*. It may witness that two displayed set rows have the same carried classifier behavior, but it does not export host set equality, powersets, unrestricted separation, quotient equality of sets, selected representatives, or an equality oracle outside* $M, Q, I, R, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 3668.1. The row E is an extensionality-boundary ledger, not a quotient constructor or host equality coordinate.

Use the extensional boundary obligation and route, Definition 3668.60 and Definition 3668.62. They force every extensional read through the membership source, classifier, subset implication, and restricted-comprehension rows before H, C, P, N are replayed.

Finally cite classifier extensional nonescape, Definition 3668.71. The displayed classifier boundary remains inside $M, Q, I, R, E, H, C, P, N$, so the refused powerset, unrestricted-separation, quotient-equality, selected-representative, host-equality, and equality-oracle readings cannot be exported. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryWitness`

Theorem 2703.66 (SetLike subset-comprehension stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, subset implication through* I *and restricted comprehension through* R *share the same membership classifier* Q *over the membership source* M*. Any consumer that reads both rows must keep the common classifier visible through* H, C, P, N*, and the read cannot introduce a powerset row, unrestricted separation row, host subset equality, or an external comprehension principle.*

Proof. Start from the accepted carrier and its subcarrier $S_{\partial} = (M, Q, I, R, E)$. The rows I and R both sit inside the same finite boundary and have access only to the membership source M and classifier Q.

The subset implication row I reads implication between displayed membership classifiers. The restricted-comprehension row R may bind only a classifier already routed through M, Q, I; it therefore cannot choose a second classifier outside the packet.

The structural rows close the stability claim. Componentwise transport H, replay C, provenance P, and local naming N preserve the same classifier route for both I and R, so the combined read remains finite BHist data and does not become a powerset or unrestricted separation surface. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetComprehensionStability`

Theorem 2703.67 (SetLike membership carrier stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a membership read remains on the same finite carrier when transported by the componentwise* hsame *row* H *and replayed by* Cont *row* C*. More precisely, if a consumer reads membership through* M, Q *before transport, then every transported read is again a membership-source and classifier read through* M, Q, H, C, P, N*, not a new host element relation, powerset coordinate, or external equality predicate.*

Proof. Project the carrier of Definition 3668.1. The membership source M and classifier Q are the only rows from which an element-in-set read can be opened.

Apply the componentwise transport row H. Since H is recorded for M, Q, I, R, E together with the structural rows, it preserves the displayed membership source and classifier rather than replacing them by an ambient relation.

Finally replay through C and name the route through P, N. The continuation can expose only the transported M, Q rows, so the transported consumer remains inside the same finite BHist carrier and no host powerset or external equality predicate enters. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierStability`

Theorem 2703.68 (SetLike classifier transport exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, classifier transport is exact on the displayed membership surface: subset implication through* I*, restricted comprehension through* R*, and extensional comparison through* E *may use only the classifier* Q *transported by* H *and replayed by* C*. The transported classifier cannot be weakened to a proof-irrelevant host proposition or strengthened to an unrestricted comprehension classifier.*

Proof. Start with the classifier row Q. The subset, comprehension, and extensionality rows I, R, E are all described in Definition 3668.1 as rows over the same membership source and classifier.

Transport by H is componentwise. It therefore moves Q together with I, R, E and leaves no independent classifier coordinate from which a consumer could read a stronger or weaker predicate.

Replay by C exposes the same classifier route to downstream packets. Because P, N source and name only this finite packet, the transported classifier is exact: it remains the displayed membership classifier and does not become proof irrelevance, unrestricted separation, or a host proposition. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierTransportExactness`

Theorem 2703.69 (SetLike ledger scope closure)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the visible ledger scope for membership, subset implication, restricted comprehension, and extensional boundary reads is exactly the finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Every $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *consumer that leaves this route lacks the carrier rows needed for an accepted read, and every consumer that stays on the route has the full ledger scope required by the chapter.*

Proof. The forward direction is carrier projection. The rows M, Q, I, R, E are the complete membership-boundary prefix, and the rows H, C, P, N are the complete transport, replay, provenance, and local naming tail.

For nonescape, suppose a consumer leaves this route. It then omits either the membership source, the classifier, one of the subset/comprehension/extensional boundary rows, or one of the structural rows that preserves and names them. By Definition 3668.10, such a consumer lacks a required local obligation.

For closure inside the route, keep all nine rows visible. The earlier membership route, subset implication, bounded comprehension, extensional boundary, and subset ledger theorems each consume only these rows, so the displayed $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ ledger scope is exhausted by the route and no additional host coordinate is required. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeLedgerScopeClosure`

Falsifiable prediction: A consumer that claims a $\mathsf{SetLike}^{\uparrow}$ membership, subset, restricted-comprehension, or extensionality read while omitting the membership source, classifier, subset implication, restricted comprehension, extensionality boundary, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 3668.1.

Independence witness: typelike, firstorder, modeltheory: $\mathsf{SetLike}^{\uparrow}$ is not bijective to any listed sibling because it carries membership source, classifier, subset implication, restricted comprehension, extensionality boundary, componentwise transport, continuation replay, provenance, and local NameCert rows. TypeLikeUp adds displayed indexed fibers beyond membership; FirstOrderUp adds syntax and deduction rows without membership-source extensionality; ModelTheoryUp adds satisfaction over FirstOrderUp without a general finite membership carrier.

Theorem 2703.70 (SetLike subset ledger obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the subset ledger exposed to downstream consumers consists exactly of membership-carrier rows* M, Q*, subset-as-membership-implication rows* I*, restricted-comprehension provenance rows* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows named through* P, N*. The ledger exports no host set extensionality, unrestricted comprehension, powerset row, quotient equality of sets, or external inclusion predicate.*

Proof. Project the accepted carrier in Definition 3668.1. The membership source M and displayed classifier Q are the only rows from which membership facts can be read.

Use the subset obligation of Definition 3668.58. The subset row I is an implication between displayed membership classifiers over M, Q, not a host inclusion relation.

Add the restricted-comprehension and extensional boundary controls from Definition 3668.59 and Definition 3668.60. The row R records only bounded comprehension provenance, while E remains a boundary ledger and does not export host set extensionality.

Finally replay Definition 3668.27. The structural rows H, C, P, N preserve and name the same finite route, so the subset ledger seen by downstream TypeLike, FirstOrder, and ModelTheory consumers contains no powerset, unrestricted comprehension, quotient equality, or external inclusion coordinate. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetLedgerObligation`

Theorem 2703.71 (SetLike classifier extensional nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensional row* E *can compare only displayed membership rows over* M, Q, I, R*. It cannot export quotient equality of sets, powersets, unrestricted separation, host set equality, or an external inclusion predicate; any accepted extensional read must keep* E *inside the structural route* H, C, P, N*.*

Proof. Project E from the carrier of Definition 3668.1. The carrier assigns E to the extensionality-boundary ledger, not to a quotient or host equality coordinate.

Use the extensional boundary route of Definition 3668.62. That theorem forces every extensional read through the membership-boundary subcarrier $S_{\partial} = (M, Q, I, R, E)$ before the structural rows are replayed.

Finally apply Definition 3668.68 and Definition 3668.70. Classifier transport stays on the displayed Q surface, and the downstream ledger exposes no powerset, unrestricted comprehension, quotient equality, or host inclusion coordinate. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierExtensionalNonescape`

Definition 2703.72 (SetLike extensional-subset carrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the extensional-subset subcarrier is the finite displayed comparison packet

$$S_{\subseteq\partial} = (M, Q, I, E, H, C, P, N).$$

Here M, Q *expose the membership source and classifier,* I *records subset-as-classifier implication,* E *records the extensional comparison boundary, and* H, C, P, N *transport, replay, source, and locally name the same displayed comparison. The subcarrier has no coordinate for host set equality, powersets, unrestricted separation, quotient equality of sets, or an external inclusion predicate.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetCarrier`

Theorem 2703.73 (SetLike extensional-subset NameCert obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an accepted extensional-subset read through* $S_{\subseteq\partial}$ *requires exactly membership-source admission through* M*, classifier exposure through* Q*, subset implication through* I*, extensional-boundary exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming through* N*. It cannot require or export host set equality, powersets, unrestricted separation, quotient equality of sets, or an external inclusion predicate.*

Proof. Project the subcarrier of Definition 3668.72. The only comparison rows available before the structural tail are the membership source, classifier, subset implication, and extensional boundary.

Use the membership route and subset ledger obligation, Definition 3668.27 and Definition 3668.70. These force subset reads to remain implications over the displayed classifier rather than external inclusion predicates.

Apply classifier extensional nonescape and ledger scope closure, Definition 3668.71 and Definition 3668.69. The row E stays a boundary ledger over displayed membership rows, while H, C, P, N preserve and name the same route. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetNameCertObligations`

Theorem 2703.74 (SetLike extensional-subset transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, componentwise* hsame *transport and displayed* Cont *replay preserve the extensional-subset route*

$$M, Q, I, E \longmapsto H, C, P, N.$$

Thus a transported extensional-subset read remains the same classifier route and cannot become host set equality, a powerset read, unrestricted separation, quotient equality of sets, or an external inclusion predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetTransport`

Proof. Start with the extensional-subset subcarrier of Definition 3668.72. It keeps only M, Q, I, E before structural replay, so the displayed classifier and implication rows determine the comparison.

Apply classifier transport exactness. The row H transports the displayed classifier surface componentwise, and C replays the same subset and extensional-boundary route rather than introducing a second inclusion relation.

Finally use the extensional-subset obligations of Definition 3668.73. Provenance and local naming through P, N keep the transported read inside the finite BHist packet, excluding host equality, powersets, unrestricted separation, quotient equality, and external inclusion. □

Theorem 2703.75 (SetLike restricted comprehension membership factorization)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every restricted-comprehension read through* R *first factors through the membership source* M*, displayed classifier* Q*, and subset implication row* I*. Only after this factorization may the read use* H, C, P, N *for transport, replay, provenance, and local naming; it cannot create a powerset row, unrestricted separation principle, host subset object, or external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRestrictedComprehensionMembershipFactorization`

Proof. Start with Definition 3668.59. It admits R only as a restricted-comprehension row bounded by a displayed certified family over M, Q, I.

The common classifier is fixed by Definition 3668.66. Subset implication through I and restricted comprehension through R share the membership classifier Q over the source M, so a comprehension read cannot skip the membership route.

Apply bounded comprehension nonescape and ledger scope closure, Definition 3668.26 and Definition 3668.69. The structural rows H, C, P, N preserve the displayed factorization and add no powerset, unrestricted separation, host subset object, or external comprehension coordinate. □

Theorem 2703.76 (SetLike TypeLike FirstOrder ModelTheory consumer boundary)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, *or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must cite the SetLike route through*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Such a consumer cannot replace the route by external satisfaction, host membership, quotient equality of sets, a powerset row, unrestricted comprehension, or a host universe of sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeFirstOrderModelTheoryConsumerBoundary`

Proof. Use the membership route theorem Definition 3668.27. It already states that TypeLike, FirstOrder, and ModelTheory sibling reads may use $\mathsf{SetLike}^{\uparrow}$ only through the carried membership route.

Next apply the subset ledger obligation and classifier extensional nonescape, Definition 3668.70 and Definition 3668.71. These results keep subset, comprehension, and extensionality reads on M, Q, I, R, E before H, C, P, N are used.

Finally invoke Definition 3668.69. A consumer leaving the displayed route lacks the required carrier rows, while a consumer staying on it receives the complete local ledger. Therefore external satisfaction, host membership, quotient equality, powersets, unrestricted comprehension, and host universes are outside the accepted SetLike consumer boundary. □

Theorem 2703.77 (SetLike root unblock for FirstOrder signatures)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *signature consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must factor membership, subset, restricted-comprehension, and extensional-boundary reads through the displayed route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The signature consumer may use this route as the SetLike membership base for syntactic symbols, arities, relation slots, and bounded classifier reads, but it cannot replace the route by host set membership, an ambient universe of sets, a powerset row, unrestricted comprehension, or quotient equality of sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockFirstOrderSignature`

Proof. Project the accepted carrier of Definition 3668.1. The only SetLike data available to a signature consumer are the membership source, classifier, subset implication, restricted comprehension, extensional boundary, transport, replay, provenance, and local naming rows.

Use the membership route theorem Definition 3668.27. It forces FirstOrder sibling reads through the carried membership route before syntax can consume SetLike membership, subset, comprehension, or extensional comparison data.

Apply classifier transport and extensional boundary routing, Definition 3668.12 and Definition 3668.62. The syntactic consumer receives transported displayed classifiers and displayed boundary ledgers, not host elementhood, ambient universes, powersets, unrestricted comprehension, or quotient equality. □

Theorem 2703.78 (SetLike root unblock for ModelTheory satisfaction)**.** *Every* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *may read SetLike data only through the classifier-based membership route and the* $\mathsf{FirstOrder}^{\uparrow}$ *syntactic carrier that consumes that route. The satisfaction-facing read must keep* $M, Q, I, R, E, H, C, P, N$ *visible, and cannot import an external satisfaction relation, quotient equality of sets, host powerset, unrestricted comprehension, or ambient host universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockModelTheorySatisfaction`

Proof. Start from Definition 3668.1. The carrier has no satisfaction row of its own; it exposes only the SetLike membership source, classifier, subset, restricted-comprehension, extensional-boundary, transport, replay, provenance, and local-name rows.

Use Definition 3668.27. A ModelTheory consumer can cite SetLike only through the same classifier-based route already available to TypeLike and FirstOrder siblings, so satisfaction cannot read an independent element relation.

Now apply Definition 3668.12 and Definition 3668.62. Transport preserves the displayed classifier route, and extensional reads remain boundary ledgers over M, Q, I, R, E before H, C, P, N are replayed. The satisfaction consumer therefore obtains SetLike data only through the FirstOrder syntactic carrier built on that route, with no external satisfaction relation, quotient equality, host powerset, unrestricted comprehension, or ambient universe. □

Theorem 2703.79 (SetLike kernel membership source totality)**.** *Every accepted* $\mathsf{SetLike}^{\uparrow}$ *membership consumer must expose the full kernel membership source before any TypeLike, FirstOrder, or ModelTheory read is available. Concretely, the consumer must display M as the finite* BHist *membership source, Q as the membership classifier row, H as the componentwise* hsame *transport ledger, C as the displayed* Cont *replay, P as the* Pkg *provenance row, and N as the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

Proof. Start with the carrier of Definition 3668.1. The carrier lists $M, Q, I, R, E, H, C, P, N$, so any accepted membership read has M, Q, H, C, P, N available as displayed rows.

Use the membership route theorem Definition 3668.27. It forces TypeLike, FirstOrder, and ModelTheory consumers through the carried membership route before they may read typed, syntactic, or satisfaction-facing data.

Finally apply the consumer boundary of Definition 3668.76. A consumer lacking M, Q, H, C, P, N has left the displayed route, while a consumer staying inside the route exposes exactly the kernel membership source and structural rows. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeKernelMembershipSourceTotality`

Theorem 2703.80 (SetLike restricted comprehension kernel scope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, the restricted-comprehension row R is admitted only over the displayed membership source M, classifier Q, and subset implication row I. The row R cannot export a powerset, unrestricted separation, a host subset family, or an external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRestrictedComprehensionKernelScope`

Proof. Apply the restricted-comprehension membership factorization theorem Definition 3668.75. It already factors every R read through M, Q, I before structural replay.

Use bounded comprehension nonescape from Definition 3668.26. This blocks powerset, unrestricted separation, host subset objects, and external comprehension predicates from the comprehension row.

The ledger scope closure theorem Definition 3668.69 then repacks the result through H, C, P, N. Thus R remains a kernel-scoped row over M, Q, I and not an escaping separation principle. □

Theorem 2703.81 (SetLike extensional ledger kernel nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, extensional comparison through E is a visible ledger over*

the membership, classifier, subset, and restricted-comprehension rows. It is not quotient equality of sets, host extensionality, selected representatives, or an equality oracle outside $M, Q, I, R, E, H, C, P, N$.

Proof. Begin with the extensional boundary route Definition 3668.62. It routes extensional reads through the membership-boundary subcarrier before structural replay.

Apply classifier extensional nonescape from Definition 3668.71. The extensional comparison remains attached to the displayed classifier and cannot become quotient equality or host extensionality.

Finally use the TypeLike FirstOrder ModelTheory consumer boundary Definition 3668.76. Downstream consumers receive the visible ledger $M, Q, I, R, E, H, C, P, N$ and no equality oracle outside it. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalLedgerKernelNonescape`

Theorem 2703.82 (SetLike extensional-boundary consumer exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of a SetLike extensional comparison must read exactly*

$$M, Q, I, R, E, H, C, P, N.$$

The row E *remains the displayed extensional-boundary ledger over the membership, classifier, subset, and restricted-comprehension route. It cannot be replaced by host set equality, quotient equality, powerset equality, unrestricted separation, or any equality oracle outside the accepted packet.*

Proof. Begin with the extensional boundary route Definition 3668.62. It makes E a boundary row over M, Q, I, R before the structural tail H, C, P, N is replayed.

Use the classifier transport route Definition 3668.12. Transport by H preserves the displayed classifier route and does not substitute a host equality predicate, quotient equality, powerset equality, or unrestricted separation row for E.

Apply ledger scope closure Definition 3668.69. The membership, classifier, subset, comprehension, and extensional rows are repacked with H, C, P, N as one finite BEDC ledger.

Finally use the TypeLike FirstOrder ModelTheory consumer boundary Definition 3668.76. A downstream consumer that reads the extensional comparison therefore receives exactly $M, Q, I, R, E, H, C, P, N$, while any attempted replacement for E leaves the accepted $\mathsf{SetLike}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryConsumerExactness`

Theorem 2703.83 (SetLike membership carrier root surface)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the membership-facing root surface seen by* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, and* $\mathsf{ModelTheory}^{\uparrow}$ *is exactly the finite* BHist *carrier*

$$S_{\in} = (M, Q, H, C, P, N).$$

Here M *is the displayed membership source,* Q *is the membership classifier,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The surface exports no powerset, host elementhood, quotient equality of sets, unrestricted comprehension, selected representatives, or external satisfaction predicate.*

Proof. Project the accepted carrier and the membership carrier of Definition 3668.2. The only rows retained in $S_\in$ are M, Q, H, C, P, N.

Use the membership route theorem Definition 3668.27. It forces TypeLike, FirstOrder, and ModelTheory consumers to enter through the carried membership source and classifier before any sibling read is available.

Finally apply membership carrier stability and the consumer boundary, Definition 3668.67 and Definition 3668.76. Componentwise transport and continuation replay keep the read inside the same BHist route, while provenance and local naming certify that no host element relation, powerset, quotient equality, unrestricted comprehension, selected representative, or external satisfaction row has been inserted. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierRootSurface`

Theorem 2703.84 (SetLike membership classifier stability)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the membership classifier* Q *remains stable under the complete displayed route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Subset implication through I*, bounded comprehension through* R*, and extensional comparison through* E *may consume* Q *only as the same classifier over the same* BHist *source* M*. A* $\mathsf{FirstOrder}^\uparrow$*,* $\mathsf{TypeLike}^\uparrow$*, or* $\mathsf{ModelTheory}^\uparrow$ *consumer therefore receives a stable SetLike classifier, not a proof-irrelevant host predicate, ambient powerset classifier, unrestricted separation rule, or external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierStability`

Proof. Start with classifier transport exactness, Definition 3668.68. It states that I, R, E consume the displayed membership classifier Q and that transport by H does not replace it with a second predicate.

Next use subset-comprehension stability and extensional ledger nonescape, Definition 3668.66 and Definition 3668.81. These keep the subset, comprehension, and comparison rows attached to the same M, Q route.

The structural tail is supplied by C, P, N. By the TypeLike, FirstOrder, and ModelTheory consumer boundary, any downstream read that omits this tail is not an accepted SetLike read, and any downstream read that keeps it receives only the stable displayed classifier. □

Theorem 2703.85 (SetLike bounded comprehension ledger)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the bounded-comprehension ledger is the finite route*

$$M, Q, I, R \longmapsto E, H, C, P, N.$$

The row R *records only comprehension over classifiers already exposed by the membership source* M*, membership classifier* Q*, and subset-implication row* I*; the extensional ledger* E *can compare only the displayed result. Thus FirstOrder syntax, TypeLike family data, and ModelTheory satisfaction may consume the ledger only through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, with no powerset, unrestricted separation, host subset object, quotient set equality, or external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionLedger`

Proof. Apply restricted-comprehension membership factorization, Definition 3668.75. It factors every R read through M, Q, I before the structural rows are available.

Use bounded comprehension nonescape and the extensional boundary route, Definition 3668.26 and Definition 3668.62. The first keeps R from becoming a powerset or unrestricted separation principle; the second keeps any comparison of the displayed result inside E over the same classifier route.

Finally replay ledger scope closure and the sibling consumer boundary, Definition 3668.69 and Definition 3668.76. The FirstOrder, TypeLike, and ModelTheory consumers therefore receive only the finite BEDC ledger $M, Q, I, R, E, H, C, P, N$ and no host comprehension or quotient-set coordinate. □

Theorem 2703.86 (SetLike TypeLike signature scope)**.** *Let $S = (M, Q, I, R, E, H, C, P, N)$ be an accepted $\mathsf{SetLike}^{\uparrow}$ packet. Any TypeLike, FirstOrder, or ModelTheory consumer that reads S as a set-like signature must first read the membership source row M, membership classifier row Q, subset implication row I, restricted-comprehension row R, extensional boundary row E, componentwise transport row H, continuation row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ row N. The signature read exports no ambient powerset, unrestricted separation, quotient equality of sets, selected representative, or host equality oracle outside $M, Q, I, R, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypelikeSignatureScope`

Proof. Start with the membership route of Definition 3668.27. It makes M and Q the visible source of every membership read before a typed, syntactic, or satisfaction-facing consumer can inspect the packet.

Next apply restricted comprehension factorization from Definition 3668.75 and the kernel scope theorem Definition 3668.80. These force R to remain a row over M, Q, I rather than a powerset or unrestricted separation principle.

Finally use the consumer boundary of Definition 3668.76 together with the extensional ledger nonescape theorem Definition 3668.81. The downstream TypeLike, FirstOrder, and ModelTheory reads receive exactly $M, Q, I, R, E, H, C, P, N$, and no external set-equality or comprehension coordinate is available. □

Theorem 2703.87 (SetLike typed-comprehension kernel envelope)**.** *For every accepted $\mathsf{SetLike}^{\uparrow}$ carrier $S = (M, Q, I, R, E, H, C, P, N)$, a typed-comprehension read is accepted only as the finite BEDC envelope*

$$M, Q, I, R \longmapsto H, C, P, N.$$

Here M is the BHist membership source, Q is the classifier-bounded membership row, I is subset implication over that classifier, and R is restricted comprehension over the same displayed rows. The envelope cannot be read as an ambient powerset, a host universe, unrestricted separation, quotient equality of sets, or an external comprehension predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypedComprehensionKernelEnvelope`

Proof. Project the carrier. The rows M, Q, I, R are the only membership, classifier, subset, and restricted-comprehension data before the structural tail.

Use restricted-comprehension membership factorization and restricted-comprehension kernel scope. These results force R to remain bounded by M, Q, I rather than becoming an unrestricted separation or powerset row.

Finally replay the structural route through H, C, P, N. Since no carrier coordinate stores an ambient powerset, host universe, quotient equality, or external comprehension predicate, the accepted typed-comprehension read is exactly the finite BEDC envelope. □

Theorem 2703.88 (SetLike FirstOrder membership substitution boundary)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{FirstOrder}^{\uparrow}$ *substitution consumer may substitute membership only through the displayed classifier route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The substitution boundary is classifier-bounded over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it cannot import host membership, an ambient universe of sets, unrestricted comprehension, quotient equality, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFirstOrderMembershipSubstitutionBoundary`

Proof. Start with SetLike TypeLike signature scope. It exposes $M, Q, I, R, E, H, C, P, N$ before any syntactic consumer may read the packet as a signature.

Apply the typed-comprehension kernel envelope. The comprehension side of the substitution remains bounded by M, Q, I, R, and the extensional row E stays a boundary ledger over the same classifier route.

Use the FirstOrder signature root unblock theorem. A substitution consumer that leaves $M, Q, I, R, E, H, C, P, N$ lacks the displayed SetLike base, so host membership, ambient universes, unrestricted comprehension, quotient equality, and external satisfaction remain outside the accepted boundary. □

Theorem 2703.89 (SetLike ModelTheory satisfaction envelope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer may read SetLike data only after the* $\mathsf{FirstOrder}^{\uparrow}$ *syntax has consumed the classifier-bounded membership and typed-comprehension route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Thus satisfaction sees a BEDC envelope over displayed membership, restricted comprehension, extensional boundary, transport, replay, provenance, and local naming rows, not an ambient powerset, host universe, quotient set equality, unrestricted separation, or external satisfaction predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeModelTheorySatisfactionEnvelope`

Proof. Read the ModelTheory satisfaction root unblock theorem. It already routes satisfaction-facing use through the classifier-based membership route and the FirstOrder syntactic carrier.

Next use FirstOrder membership substitution boundary. The syntactic carrier receives only $M, Q, I, R, E, H, C, P, N$, so the satisfaction consumer inherits the same classifier-bounded envelope.

Finally apply typed-comprehension kernel envelope and extensional ledger kernel nonescape. The comprehension and extensional rows remain finite BEDC ledgers, and no ambient powerset, host universe, quotient set equality, unrestricted separation, or external satisfaction predicate is available. □

Theorem 2703.90 (SetLike root-unblock kernel membership envelope)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *or* $\mathsf{TypeLike}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *reads membership only through the finite kernel envelope*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Here M *is the displayed* BHist *membership carrier,* Q *is the classifier-bounded membership row,* I *is the subset-implication ledger,* R *is the bounded-comprehension row,* E *is the extensional-boundary row, and* H, C, P, N *provide componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. The envelope does not expose host set extensionality, quotient membership, unbounded comprehension, or an ambient universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockKernelMembershipEnvelope`

Proof. Start with the SetLike root-unblock theorem and the TypeLike signature scope. These routes expose the membership carrier M, classifier row Q, subset implication ledger I, bounded-comprehension row R, extensional-boundary row E, and the structural tail H, C, P, N before any typed consumer can read the packet.

For FirstOrder consumers, apply the FirstOrder membership substitution boundary. For ModelTheory consumers, apply the ModelTheory satisfaction envelope. Both routes pass through the same classifier-bounded membership and typed-comprehension rows before syntax or satisfaction can use SetLike data.

Finally use typed-comprehension kernel envelope and extensional ledger kernel nonescape. The bounded-comprehension and extensional-boundary rows remain displayed BEDC rows, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host set extensionality, quotient membership, unbounded comprehension, or an ambient universe of sets, every listed consumer receives exactly the finite kernel membership envelope. □

Theorem 2703.91 (SetLike root membership triple unblock)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *and* $\mathsf{TypeLike}^{\uparrow}$ *consumer that reads membership from an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must enter through the shared BEDC membership source* M *and classifier row* Q *before it can consume syntax, satisfaction, or carrier-transport data. The common read is the finite route*

$$M, Q \longmapsto I, R, E, H, C, P, N.$$

It cannot be replaced by host elementhood, an ambient powerset, unrestricted separation, quotient equality, an external satisfaction predicate, or a selected element.

Proof. Start with the membership carrier root surface and classifier stability. They make M, Q the only membership source and classifier available to the three downstream consumers.

For syntactic use, apply the FirstOrder membership substitution boundary. For satisfaction use, apply the ModelTheory satisfaction envelope. For carrier transport, apply the TypeLike signature scope.

Finally use the root-unblock kernel membership envelope. All three reads are repacked through the same $M, Q, I, R, E, H, C, P, N$ ledger, so none of them has a coordinate for host elementhood, powersets, unrestricted separation, quotient equality, external satisfaction, or selected elements. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipTripleUnblock`

Theorem 2703.92 (SetLike root membership source triple)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the row* M *is the single source-admission row for* $\mathsf{FirstOrder}^{\uparrow}$ *formulas,* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction rows, and* $\mathsf{TypeLike}^{\uparrow}$ *carrier transport. Each consumer may read* M *only together with the classifier* Q *and the structural rows* H, C, P, N*, and no consumer may import a second membership source, host universe, quotient-set source, unrestricted separation domain, or selected representative.*

Proof. Project the membership carrier $S_{\in} = (M, Q, H, C, P, N)$ from Definition 3668.83. It exposes one source row M and one classifier row Q.

Use classifier stability and the TypeLike signature scope. They force typed transport and syntactic reads to keep the same source while H, C, P, N replay and name the packet.

Apply the ModelTheory satisfaction envelope for satisfaction-facing reads. Since satisfaction is downstream of the FirstOrder syntax carrier, it inherits the same M, Q, H, C, P, N source admission and cannot add a host universe, quotient source, unrestricted domain, or selected representative. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipSourceTriple`

Theorem 2703.93 (SetLike root comprehension ledger triple)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, bounded comprehension remains a ledger-filtered membership route for* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{ModelTheory}^{\uparrow}$*, and* $\mathsf{TypeLike}^{\uparrow}$ *consumers:*

$$M, Q, I, R \longmapsto E, H, C, P, N.$$

The row R *may be read only after* M, Q, I *are visible, and its result may be compared only through* E *before transport, replay, provenance, and local naming. The route exports no ambient powerset, unrestricted separation, quotient equality, host subset object, external comprehension predicate, or choice of elements.*

Proof. Begin with the bounded-comprehension ledger and the typed-comprehension kernel envelope. They place R after the membership source, membership classifier, and subset-implication rows.

Next apply the FirstOrder membership substitution boundary and ModelTheory satisfaction envelope. Syntax and satisfaction can consume comprehension only through the same classifier-bounded ledger.

Finally use TypeLike signature scope and the root-unblock kernel membership envelope. Typed carrier transport receives the same $M, Q, I, R, E, H, C, P, N$ route, while powersets, unrestricted separation, quotient equality, host subset objects, external comprehension predicates, and element choices remain outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootComprehensionLedgerTriple`

Theorem 2703.94 (SetLike type-model membership envelope)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, and* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must read membership, bounded comprehension, extensional boundary, transport, replay, provenance, and local naming as the single finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

No such consumer may replace this route by host membership, an ambient powerset, unrestricted separation, quotient equality, or external satisfaction.

Proof. Begin with the root membership triple and the root comprehension ledger triple, Definition 3668.91 and Definition 3668.93. They expose M, Q, I, R, E before any typed, syntactic, or satisfaction-facing consumer can read the packet.

For syntactic consumers, apply the FirstOrder membership substitution boundary, Definition 3668.88. For satisfaction consumers, apply the ModelTheory satisfaction envelope, Definition 3668.89. For typed-family consumers, use TypeLike signature scope, Definition 3668.86.

The three routes share the same structural tail H, C, P, N. Since the accepted carrier stores no coordinate for host membership, powersets, unrestricted separation, quotient equality, or external satisfaction, the displayed route is the only membership envelope available to all three consumers. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeModelMembershipEnvelope`

Theorem 2703.95 (SetLike public NameCert export)**.** *The public* $\mathsf{SetLike}^{\uparrow}$ *export is exactly the finite route*

$$M, Q, I, R, E, H, C, P, N.$$

Membership reads expose M, Q, subset reads expose I, restricted-comprehension reads expose R, extensional-boundary reads expose E, and all public consumers pass through the structural rows H, C, P, N. The export contains no powerset coordinate, unrestricted separation principle, host quotient equality of sets, selected representative row, global choice coordinate, or external universe of all sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikePublicNameCertExport`

Proof. Start from the carrier of Definition 3668.1. Its displayed rows are precisely $M, Q, I, R, E, H, C, P, N$, so every public read must be assembled from those rows before any downstream consumer can name a $\mathsf{SetLike}^{\uparrow}$ packet.

The membership route theorem Definition 3668.27 supplies the public read for M, Q, H, C, P, N. The subset ledger obligation Definition 3668.70 and the bounded comprehension obligation Definition 3668.59 add I and R without adding a powerset or unrestricted separation coordinate.

Finally, the extensional-boundary consumer exactness theorem Definition 3668.82 fixes E as a displayed boundary row rather than host quotient equality. The TypeLike, FirstOrder, and ModelTheory consumer boundary keeps sibling consumers on the same nine-row route, so the public export is exactly $M, Q, I, R, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SetLike}^{\uparrow}$ packet is generated from a finite membership source, classifier row, subset implication row, restricted-comprehension row, extensionality-boundary ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3668.1, Definition 3668.10, the membership, subset, bounded-comprehension, extensional-boundary, TypeLike, FirstOrder, and ModelTheory consumer-boundary rows, and Definition 3668.95 bind the mature paper surface from the finite SetLike membership carrier to its public consumer export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not claim an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, global choice of elements, Lean verification, bridge checking, or a host universe of all sets.

Upgrade path: Bridge checking requires Lean verification of the same carrier, consumer-boundary rows, and public export for BEDC.Derived.SetLikeUp.

Chapter 2704

A Concrete Naming Certificate for $\mathsf{SchauderBasis}^{\uparrow}$

Following Schauder and the Banach-space basis tradition, the $\mathsf{SchauderBasis}^{\uparrow}$ packet records a sequence of named basis vectors together with finite coordinate projections and partial-sum controls. It sits over the linear carrier of chapter 47, the norm row of chapter 89, the Banach-space completion surface of chapter 90, the finite-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives basis expansions a finite NameCert surface before any coordinate functional or projection budget is consumed.

Definition 2704.1 (Schauder basis carrier)**.** *A* $\mathsf{SchauderBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, Nrm, B, C, S, E, R, H, T, P, L)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming the vector-space carrier, addition, scalar action, zero vector, and finite linear-combination syntax.*
2. *Nrm is the* $\mathsf{Norm}^{\uparrow}$ *row naming the norm, triangle budget, scalar-compatibility budget, and zero-detection surface.*
3. *B is the basis-vector stream row, a* $\mathsf{StreamName}^{\uparrow}$ *finite-window record whose inspected entries name the vectors $b_0, b_1, \ldots$ available to the packet.*
4. *C is the coordinate-readback row, recording finite coordinate requests and their* $\mathsf{RegSeqRat}^{\uparrow}$ *rational names.*
5. *S is the partial-sum row, recording finite sums $\sum_{i<n} c_i b_i$ together with the requested truncation index.*
6. *E is the error-control row, a* $\mathsf{Real}^{\uparrow}$*-facing norm estimate for the tail between the named vector and the displayed partial sum.*
7. *R is the reconstruction row, recording that each accepted vector read is recovered through the displayed partial sums and tail estimates.*
8. *H records componentwise* hsame *transport for the vector, norm, basis stream, coordinate, partial-sum, error, reconstruction, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a coordinate request through B, C, S, E, R.*

10. *P records* Pkg *provenance over* $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, Nrm, B, C, S, E, R, H, T, P, L$. It has no coordinate for a Hamel basis choice, an arbitrary linear functional, a detached coefficient oracle, ambient compactness, countable choice, or a global unconditional-basis assertion.

Lean4 ✓ `BEDC.Derived.SchauderBasisUp.SchauderBasisTasteGate single carrier alignment`

Theorem 2704.2 (Schauder basis NameCert obligations)**.** *For every accepted* $\mathsf{SchauderBasis}^{\uparrow}$ *carrier $K = (V, Nrm, B, C, S, E, R, H, T, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, vector-space exposure through V, norm exposure through Nrm, basis-stream exposure through B, coordinate readback through C, partial-sum exposure through S, tail-error control through E, reconstruction through R, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through L.*

Proof. Carrier admission is coordinate projection from Definition 3669.1. The accepted packet exposes the vector-space carrier, norm row, basis-vector stream, coordinate readback, partial sums, tail-error estimates, reconstruction row, transport, replay, provenance, and local naming.

The analytic route is fixed by V, Nrm, B, C, S, E, R. The vector row supplies the finite linear-combination syntax, the norm row supplies the tail metric, the basis stream supplies inspected vectors, the coordinate row supplies rational names, the partial-sum row supplies the finite approximants, the error row supplies the Real-facing tail estimate, and the reconstruction row binds those finite approximants to the named vector.

The structural rows H, T, P, L preserve that same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for a Hamel basis choice, detached coefficient oracle, arbitrary functional, compactness substitute, countable-choice schedule, or unconditional-basis assertion, those objects cannot belong to the local NameCert surface. □

Theorem 2704.3 (Schauder basis coordinate-projection handoff)**.** *Every coordinate-projection consumer of an accepted* $\mathsf{SchauderBasis}^{\uparrow}$ *packet must read the projection through the basis stream B, coordinate row C, partial-sum row S, tail-error row E, reconstruction row R, replay row T, provenance row P, and local name L. The handoff exports finite coordinate projections and norm-controlled partial sums, but it exports no standalone bounded linear functional, no choice of representatives outside the displayed stream, and no unconditional convergence claim.*

Proof. Start with a coordinate-projection request. The carrier definition places the visible projection data in B, C, S, E, R: the basis vectors are inspected through the stream row, coordinates are read through regular rational names, partial sums are formed in the vector-space row, and the tail is measured by the norm and Real-facing estimate.

Next use the replay and provenance rows. The replay row T requires the consumer to traverse the displayed coordinate and partial-sum route, while P records that the route belongs to the $\mathsf{SchauderBasis}^{\uparrow}$ packet and its Banach-space dependencies.

Finally the local name L certifies only that displayed route. Because the packet has no coordinate for arbitrary functionals, detached coefficient selection, ambient compactness, or unconditional convergence, a coordinate-projection consumer receives only the finite projection handoff governed by B, C, S, E, R, T, P, L. □

Falsifiable prediction: Every accepted $\mathsf{SchauderBasis}^{\uparrow}$ coordinate-projection export factors through a VecSpace carrier row, Norm row, StreamName basis-vector window, RegSeqRat coordinate readback, finite partial-sum row, Real tail estimate, reconstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: banach, vecspace, norm, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (V, Nrm, B, C, S, E, R, H, T, P, L)$, because the Schauder basis carrier binds linear syntax, norm control, basis windows, rational coordinate readback, finite partial sums, tail estimates, reconstruction, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderBasis}^{\uparrow}$ packet is generated from a vector-space row, norm row, basis-vector StreamName window, RegSeqRat coordinate readback, finite partial-sum row, Real tail estimate, reconstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3669.1, Definition 3669.2, and Definition 3669.3 seed the Schauder basis route from Banach-space carrier data to finite coordinate projections and norm-controlled partial sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Hamel bases, arbitrary linear functionals, detached coefficient oracles, ambient compactness, countable choice, unconditional convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, basis-window exposure, coordinate-readback exactness, partial-sum control, tail-error control, reconstruction, projection handoff, and non-escape rows for the displayed Schauder basis packet.

Chapter 2705

A Concrete Naming Certificate for $\mathsf{RationalCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalCauchyCompletion}^{\uparrow}$ packet records the rational Cauchy completion route as finite regular-name data rather than as a quotient of rational sequences or a selected ambient limit. It sits between the rational source of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 completion route a concrete carrier before later formalization reads it as a derived Lean surface.

Definition 2705.1 (Rational Cauchy completion carrier)**.** *A* $\mathsf{RationalCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, S, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row, recording the rational endpoints and rational difference witnesses supplied to the completion route.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording finite dyadic radii, slack comparisons, and tolerance budgets.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording inspected requests and omitted-tail boundaries for the source rational stream.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational name produced by the displayed windows and tolerances.*
5. *M is the explicit Cauchy-modulus row, a finite schedule assigning every requested tolerance a displayed tail threshold.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, D, S, R, M have been displayed.*
7. *H records componentwise* hsame *transport for the rational source, dyadic tolerance, stream window, regular readback, modulus, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through Q, D, S, R, M, E.*
9. *P records* Pkg *provenance over* $\mathsf{RationalCauchyCompletion}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, S, R, M, E, H, C, P, N$. It has no coordinate for quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalCauchyCompletionUp.RationalCauchyCompletionTasteGate single carrier alignment`

Theorem 2705.2 (Rational Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{RationalCauchyComplet}$ *carrier $K = (Q, D, S, R, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, rational-source exposure through Q, dyadic tolerance exposure through D, finite-window exposure through S, regular readback through R, explicit Cauchy-modulus control through M, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is coordinate projection from Definition 3670.1. The accepted packet exposes the rational source, dyadic tolerance ledger, finite stream window, regular rational readback, explicit modulus, terminal Real seal, transport, replay, provenance, and local naming rows.

The analytic route is fixed by Q, D, S, R, M, E. The source row Q supplies rational endpoints, D supplies dyadic tolerance comparisons, S exposes only finite observations, R records the regular rational readback, M binds each requested tolerance to a displayed tail threshold, and E seals the resulting Real-facing read.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 2705.3 (Rational Cauchy completion diagonal seal handoff)**.** *Every diagonal completion read from a* $\mathsf{RationalCauchyCompletion}^{\uparrow}$ *packet to a* $\mathsf{Real}^{\uparrow}$ *consumer factors through the same finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The explicit modulus row M is the only row that may choose tail thresholds for the diagonal, and the terminal seal E exports only the regular rational readback named by R.

Proof. Start with a Real-facing diagonal read. The carrier definition places the terminal seal in coordinate E, and E is downstream of the source, tolerance, stream, readback, and modulus rows.

Trace the read backward. A diagonal tolerance request is served by M, interpreted through the dyadic row D, inspected through the finite-window row S, and returned as the regular rational readback R from the rational source Q.

Finally H, C, P, N transport, replay, package, and name the same displayed route. There is no separate coordinate for a selected limit, quotient stream equality, countable choice schedule, or host real equality, so every accepted diagonal handoff reaches $\mathsf{Real}^{\uparrow}$ only through Q, D, S, R, M, E. □

Falsifiable prediction: Every accepted $\mathsf{RationalCauchyCompletion}^{\uparrow}$ Real-facing read factors through a Rat source row, DyadicRatCore tolerance row, StreamName finite-window row, RegSeqRat readback row, explicit Cauchy-modulus row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (Q, D, S, R, M, E, H, C, P, N)$, because the completion carrier binds

source rationals, dyadic tolerances, finite windows, regular readback, explicit modulus, terminal sealing, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}$.

Constructive story: A $\mathsf{RationalCauchyCompletion}^{\uparrow}$ packet is generated from a Rat source row, DyadicRatCore tolerance row, StreamName finite-window row, RegSeqRat readback row, explicit Cauchy modulus, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3670.1, Definition 3670.2, and Definition 3670.3 seed the rational Cauchy completion route from rational source data to the Real seal through finite tolerance, window, readback, modulus, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance exposure, finite-window exposure, regular readback, explicit modulus control, Real-seal handoff, and non-escape rows for the displayed rational Cauchy completion packet.

Chapter 2706

A Concrete Naming Certificate for $\mathsf{CompactMetricClosedBall}^{\uparrow}$

Following Bishop and Bridges constructive compact metric practice, the $\mathsf{CompactMetricClosedBall}^{\uparrow}$ packet consumes chapter 3464, chapter 129, chapter 57, chapter 127, chapter 128, and chapter 38. It names the finite-net surface of a closed ball inside a displayed compact metric carrier; it does not turn the ball into a host subset, choose a finite subcover, quotient points, or import ambient real completeness.

Definition 2706.1 (Compact metric closed ball carrier)**.** *A* $\mathsf{CompactMetricClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, c, r, M, F, T, C, H, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the shared metric carrier, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net family, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *complete-limit family over one point classifier.*
2. *c is the displayed center row carried by the same metric source.*
3. *r is the* $\mathsf{Real}^{\uparrow}$ *radius row together with its nonnegative-radius ledger.*
4. *M is the* $\mathsf{ClosedMetricBall}^{\uparrow}$ *membership row, recording the comparison $d(c, x) \leq r$ for each displayed point row x that enters the ball surface.*
5. *F is the finite-net row obtained by restricting a* $\mathsf{TotallyBounded}^{\uparrow}$ *ProbeBundle-net request to centers whose coverage route remains inside the closed-ball membership row M.*
6. *T is the total-bounded handoff row that records the tolerance request, bundled centers, membership constructors, coverage distances, and rational comparison ledgers inherited from the compact metric source.*
7. *C is the complete-limit inheritance row: every carried Cauchy name whose observations remain inside M is routed through the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit witness of X and returns a limit row still admitted by M.*
8. *H records componentwise* hsame *transport for the compact metric source, center, radius, closed-ball membership, finite-net row, total-bounded handoff, complete-limit inheritance, and structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricClosedBall}^{\uparrow}$, $\mathsf{ClosedMetricBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected finite subcover, quotient balls, host subsets, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactMetricClosedBallUp.CompactMetricClosedBallTasteGate single carrier alignment`

Theorem 2706.2 (Compact metric closed ball finite-net obligations)**.** *For an accepted* $\mathsf{CompactMetricClosedBall}^{\uparrow}$ *carrier* $K = (X, c, r, M, F, T, C, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the compact metric source X, center c, radius r, closed-ball membership row M, finite-net row F, total-bounded handoff T, complete-limit inheritance row C, transport row H, provenance row P, and local naming row N;*
2. *closed-ball membership stability, requiring the comparison $d(c, x) \leq r$ to transport through the metric classifier, the* $\mathsf{Real}^{\uparrow}$ *radius ledger, and the* $\mathsf{ClosedMetricBall}^{\uparrow}$ *row M;*
3. *finite-net extraction inside the closed ball, requiring each displayed tolerance request to read bundled centers, membership constructors, coverage distances, and rational comparison rows through F and T without leaving M;*
4. *completion inheritance for carried Cauchy names, requiring every Cauchy thread whose observations are admitted by M to use the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness family of X and return a limit row still admitted by the same closed-ball comparison;*
5. *ledger nonescape, stating that no open-cover compactness theorem, selected subcover, quotient ball, host subset, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row is exported by the packet.*

Proof. Project the carrier of Definition 3671.1. The tuple lists exactly the compact metric source, center, radius, closed-ball membership row, finite-net row, total-bounded handoff, complete-limit inheritance, transport, provenance, and local naming row.

Membership stability is inherited from the displayed metric and closed-ball surfaces. The row M supplies the center-radius comparison, X supplies the metric classifier and distance rows, and the radius row r supplies the $\mathsf{Real}^{\uparrow}$ comparison ledger. Transport through H preserves the same comparison rather than replacing it by a host subset predicate.

For a positive tolerance request, read the $\mathsf{TotallyBounded}^{\uparrow}$ family inside X and then restrict the resulting ProbeBundle route through F. The handoff row T records the bundled centers, membership constructors, coverage distances, and rational comparison ledgers that remain inside the closed-ball membership row M.

For a carried Cauchy name, use only the complete-limit family already present in X. The inheritance row C states that observations admitted by M are handed to the complete-metric limit witness and that the returned limit row is checked again by the same center-radius comparison.

The remaining structural rows H, P, N transport, source, and name this finite route. Since the carrier inventory contains no open-cover object, subcover selector, quotient-ball constructor, host subset, countable-choice row, or ambient real-completeness row, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactMetricClosedBall}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source row, center row, $\mathsf{Real}^{\uparrow}$ radius row, $\mathsf{ClosedMetricBall}^{\uparrow}$ membership row, restricted finite-net row, total-bounded handoff, complete-limit inheritance, componentwise hsame transport, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$, seed)

Scope closed: Definition 3671.1 and Definition 3671.2 seed the finite closed-ball compact-metric NameCert surface for displayed center-radius packets.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice of subcover, quotient balls, host subsets, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, membership stability, finite-net extraction inside the closed ball, completion inheritance for carried Cauchy names, and ledger nonescape.

Chapter 2707

A Concrete Naming Certificate for RegularCauchyProductCriterion$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the RegularCauchyProductCriterion$^{\uparrow}$ packet isolates the finite sufficiency criterion by which two regular rational Cauchy source rows produce a product read suitable for the CauchyProductCriterion$^{\uparrow}$ surface. It sits between the pointwise regular product carrier of chapter 461 and the synchronized product criterion of chapter 1586. The chapter records the L10 Real and Completion handoff as finite NameCert data, not as host multiplication of completed reals, quotient equality of streams, a selected product limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 2707.1 (Regular Cauchy product criterion carrier)**.** *A* RegularCauchyProductCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A and B are the two* RegSeqRat$^{\uparrow}$ *source rows whose scheduled rational observations supply the regular Cauchy inputs.*
2. *W_A and W_B are synchronized* StreamName$^{\uparrow}$ *window rows for the same precision request and product observation index.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *product-budget row, recording input tolerances, visible endpoint magnitude bounds, cross-error terms, and the output tolerance assigned to the product read.*
4. *P is the* RegularCauchyProduct$^{\uparrow}$ *route row, read only after A, B, W_A, W_B, D have been displayed.*
5. *K is the* CauchyProductCriterion$^{\uparrow}$ *sufficiency row, certifying that the displayed product route is adequate for the downstream criterion.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced by the route P under the budget D and criterion row K.*
7. *E is the* Real$^{\uparrow}$ *seal row, consumed only after the regular-rational readback R has been displayed.*
8. *H records componentwise* hsame *transport for $A, B, W_A, W_B, D, P, K, R, E, C, L, N$ and the local packet shape.*
9. *C records displayed* Cont *replay from the two source rows through the synchronized windows, dyadic product budget, regular product route, criterion row, readback row, and real seal.*

10. *L is the* Pkg *provenance over* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W_A, W_B, D, P, K, R, E, H, C, L, N$. It has no coordinate for host multiplication of completed reals, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductCriterionUp.RegularCauchyProductCriterionTasteGate single carrier alignment`

Theorem 2707.2 (Regular Cauchy product criterion NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier $Q = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *obligations are exactly carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic product-budget exactness, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, component-wise transport through H, displayed continuation replay through C, provenance through L, and local naming through N.*

Proof. Carrier admission is projection from Definition 3672.1. The displayed packet lists the two source rows, synchronized windows, dyadic budget, product route, criterion row, readback row, real seal, and structural rows.

The source and window obligations are read from A, B, W_A, W_B. These rows display the regular rational inputs and the shared product window before any product observation reaches the criterion.

The dyadic product-budget obligation is the row D. It records finite tolerance and magnitude data, so the route P is a budgeted regular Cauchy product route rather than host multiplication of completed real names.

The rows K, R, E are the downstream sufficiency, regular-rational, and real-seal boundaries. Transport through H, replay through C, provenance through L, and local naming through N preserve the same route and introduce no additional coordinate.

Thus the listed rows exhaust the local certificate surface. □

Theorem 2707.3 (Regular Cauchy product criterion window-budget exactness)**.** *For an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier, the synchronized source windows W_A, W_B and the dyadic row D determine the product-budget data consumed by the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route P and the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *row K. Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the product factors through*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto P \longmapsto K \longmapsto R.$$

No additional budget rule is read from a quotient stream, a selected product limit, countable choice, trichotomy, host real multiplication, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductCriterionUp.RegularCauchyProductCriterionCarrier window budget exactness`

Proof. Read the two regular rational source rows through the synchronized windows. The rows W_A and W_B expose the finite observations used by the product request at the same displayed precision.

The row D is the only dyadic product-budget row in the packet. It supplies the tolerance and endpoint-bound data consumed by the regular product route P.

The sufficiency row K then certifies that the displayed route is acceptable for the Cauchy product criterion. Since the readback row R occurs only after P and K, every regular-rational product read factors through the displayed chain.

The carrier has no coordinate for any excluded budget source, so the budget exactness is internal to the finite packet. □

Theorem 2707.4 (Regular Cauchy product criterion Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the displayed* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *row. Hence the terminal seal is a finite product-sufficiency object, not host multiplication, quotient stream equality, a selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The real-facing consumer can read only E as the terminal $\mathsf{Real}^{\uparrow}$ seal row. In the carrier, E is downstream of R, and R is downstream of the criterion route through P and K.

Apply the window-budget exactness theorem. It places the readback route on the finite chain from the two source rows through the synchronized windows, dyadic product budget, product route, criterion row, and regular-rational readback.

The structural rows H, C, L, N preserve that same chain under transport, replay, provenance, and local naming. They do not add a second source for the real seal.

Therefore every terminal read remains inside the displayed finite criterion, and the excluded host, quotient, selector, choice, order, and completeness routes cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore product budget, a RegularCauchyProduct route, a CauchyProductCriterion sufficiency row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3672.1, Definition 3672.2, Definition 3672.3, and Definition 3672.4 seed the finite L10 product-sufficiency object between RegularCauchyProduct and CauchyProductCriterion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic product-budget exactness, RegularCauchyProduct routing, CauchyProductCriterion sufficiency, RegSeqRat readback, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.RegularCauchyProductCriterionUp.

Chapter 2708

A Concrete Naming Certificate for $\mathsf{BishopMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopMetricSpace}^{\uparrow}$ packet names a metric presentation whose distances are read through displayed regular rational approximations, finite stream windows, and real-name seals. It sits between the abstract metric certificate of chapter 57, the positive-separation interface of chapter 3505, the finite observation surface of chapter 126, the regular rational route of chapter 301, and the sealed real-name consumer of chapter 38. The packet is a constructive metric object for Real and completion-facing consumers; it is not a host metric-space structure, quotient completion, choice-selected limit, or ambient classical distance oracle.

Definition 2708.1 (Bishop metric-space carrier)**.** *A* $\mathsf{BishopMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, D, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed point-name row and local carrier admission surface.*
2. *D is the metric-distance row, read as a* $\mathsf{Real}^{\uparrow}$ *name rather than as a host real-valued function.*
3. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to observe the distance name.*
4. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of each displayed distance.*
5. *A is the* $\mathsf{MetricApartness}^{\uparrow}$ *handoff row, present when a positive distance window exports apartness.*
6. *H stores componentwise* hsame *transport for point names, distance rows, stream windows, regular rational readbacks, apartness handoff, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from point-name reads through the distance, stream, regular rational, apartness, and Real-seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, D, S, R, A, H, C, P, N$. It has no coordinate for host equality of points, host-valued distance functions, quotient completion, choice-selected limits, total boundedness, compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BishopMetricSpaceUp.BishopMetricSpaceTasteGate single carrier alignment`

Theorem 2708.2 (Bishop metric-space NameCert obligations)**.** *For every accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *carrier*

$$M = (X, D, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed point names, Real-valued distance rows, finite stream windows, regular rational readbacks, optional positive-distance apartness handoff, componentwise transport, displayed replay, package provenance, and local naming row.*

Proof. Project the carrier of Definition 3673.1. The projection exposes point-name rows X, distance rows D, stream windows S, regular readbacks R, and the apartness handoff A.

The distance row is Real-facing because every inspected distance is routed through S and R before it reaches the $\mathsf{Real}^{\uparrow}$ seal. Thus no consumer can replace the displayed constructive distance with a host-valued function.

The apartness handoff is confined to A. It can be exported only when the displayed distance window supplies positive separation, and the transport rows H, C replay that same finite route.

Finally P and N record provenance and local naming for exactly these rows. Since the carrier has no coordinate for host point equality, quotient completion, selected limits, compactness, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 2708.3 (Bishop metric-space located Cauchy route)**.** *Every completion-facing read of an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet factors through*

$$X, D \longmapsto S, R \longmapsto A, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may inspect a Cauchy or locatedness claim only through displayed distance windows, regular rational readbacks, optional apartness handoff, transport, replay, provenance, and local naming.*

Proof. Begin with the point-name and distance rows X, D. They name the metric input but do not by themselves expose a Real-valued oracle.

The stream and regular rational rows S, R provide the finite observation route for the distance name. Any located Cauchy read must therefore pass through those rows before it reaches a Real-facing consumer.

The rows A, H, C, P, N preserve the same route under apartness handoff, componentwise transport, replay, provenance, and local naming. A claim using quotient completion, selected limits, host equality of points, compactness, total boundedness, or ambient completeness has no carrier coordinate and leaves the packet. □

Theorem 2708.4 (Bishop metric-space distance-window exactness)**.** *Let* $M = (X, D, S, R, A, H, C, P, N)$ *be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Every accepted read of a distance row factors through the displayed route*

$$D \longmapsto S \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the distance consumed by a $\mathsf{Real}^{\uparrow}$*-facing or completion-facing reader is the finite StreamName window and RegSeqRat readback of* D*, not a host-valued metric function or an ambient distance oracle.*

Proof. Project the carrier of Definition 3673.1. The row D is present only as the displayed metric-distance row, while S and R are the StreamName and RegSeqRat rows attached to that distance.

Apply Definition 3673.2. Its local certificate surface lists distance rows, finite stream windows, regular rational readbacks, transport, replay, provenance, and local naming as the exact obligations for accepted packets.

Finally use Definition 3673.3. Any Real-facing or completion-facing read must pass through the same S, R windows before it reaches a Real consumer, so no host-valued metric function can replace the displayed route. □

Theorem 2708.5 (Bishop metric-space apartness handoff)**.** *Let $M = (X, D, S, R, A, H, C, P, N)$ be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Apartness can be exported from M only through the displayed positive-distance row A, and only after the same distance-window route*

$$D \longmapsto S \longmapsto R$$

has supplied the positive separation read. The handoff cannot be replaced by host inequality of points or by an apartness predicate detached from the displayed distance data.

Proof. Read the carrier in Definition 3673.1. The row A is the only MetricApartness handoff coordinate, and the carrier contains no separate host point-equality or host inequality field.

Apply Definition 3673.2. The certificate surface makes the apartness row optional and ties it to positive-distance evidence transported with the same componentwise rows H and C.

Use Definition 3673.3 to identify the upstream read. Since every completion-facing distance claim goes through D, S, R before the structural rows, the exported apartness handoff is exactly the displayed row A downstream of those windows. □

Theorem 2708.6 (Bishop metric-space Real-seal nonescape)**.** *Let $M = (X, D, S, R, A, H, C, P, N)$ be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Every Real-facing consumer of a distance row receives only the finite route*

$$D \longmapsto S \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}$$

together with the structural rows A, H, C, P, N when they are displayed. The packet exports no quotient completion, selected limit, host point equality, total boundedness, compactness, or ambient completeness principle.

Proof. Start from Definition 3673.1. The accepted packet exposes point names, a distance row, finite windows, regular readbacks, optional apartness, transport, replay, provenance, and local naming, and it has no coordinate for the excluded objects.

Apply Definition 3673.2. The local NameCert admits exactly those displayed rows and refuses host-valued distance functions, quotient completion data, selected limits, compactness, total boundedness, and ambient completeness.

Apply Definition 3673.3. The Real-facing read stays on the finite StreamName and RegSeqRat path before it reaches the seal, so the consumer cannot escape into any object outside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMetricSpace}^{\uparrow}$ packet is generated from displayed point names, Real-valued distance rows, StreamName finite windows, RegSeqRat regular rational readbacks, optional MetricApartness handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3673.1, Definition 3673.2, and Definition 3673.3 seed the constructive metric carrier, finite NameCert obligations, and located Cauchy route for Real and completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{BishopMetricSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, compactness, total boundedness, host equality of points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-window exactness, regular rational readback, apartness handoff, located Cauchy routing, and TasteGate field faithfulness for BEDC.Derived.BishopMetricSpaceUp.

Chapter 2709

A Concrete Naming Certificate for $\mathsf{RealClosure}^{\uparrow}$

Following Kelley and Bourbaki topology, the $\mathsf{RealClosure}^{\uparrow}$ packet names a closure operation by displayed finite sequential data rather than by ambient completeness of a host real line. It sits over the terminal real-name seal of chapter 38, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the topological carrier of chapter 88, and the sequential closure packet of chapter 1571. The chapter uses these rows to expose an L10 closure object for Real-facing and completion-facing consumers without asserting quotient reals, host equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2709.1 (Real closure carrier)**.** *A* $\mathsf{RealClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, T, S, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *point-name row whose accepted entries are inspected only through terminal real-name seals.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row for the displayed Real-facing carrier.*
3. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to observe candidate closure witnesses.*
4. *Q stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of the same windows.*
5. *A is the* $\mathsf{SequentialClosure}^{\uparrow}$ *approach row, displaying a sequence or net-free sequential route from the named source rows toward the closure point.*
6. *E is the exactness and refusal row stating that the closure verdict is read from X, T, S, Q, A and not from quotient representatives, host real equality, choice-selected subsequences, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
7. *H stores componentwise* hsame *transport for Real names, topological neighbourhood rows, stream windows, regular rational readbacks, sequential approach rows, exactness rows, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay from finite Real-name observations through topology, stream, regular rational, sequential closure, and refusal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RealClosure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealClosure}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, T, S, Q, A, E, H, C, P, N$. It has no coordinate for completed subsets of a host real line, quotient real equality, selected limits, countable-choice schedules, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 2709.2 (Real closure NameCert obligations)**.** *For every accepted* $\mathsf{RealClosure}^\uparrow$ *carrier*

$$R = (X, T, S, Q, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealClosure}^\uparrow}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Real}^\uparrow$ *point names,* $\mathsf{Topology}^\uparrow$ *neighbourhood rows,* $\mathsf{StreamName}^\uparrow$ *windows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks,* $\mathsf{SequentialClosure}^\uparrow$ *approach row, exactness and refusal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RealClosureUp.TasteGate.RealClosureTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3675.1. The projection exposes Real names X, neighbourhood rows T, stream windows S, regular rational readbacks Q, and the sequential approach row A.

The closure verdict is then forced through E. That row records exactness of the displayed sequential route and refuses any replacement by quotient real representatives, host equality, countable-choice selection, trichotomy, open-cover compactness, or an ambient completeness theorem.

Finally transport and replay are only the componentwise rows H and C, while P and N record provenance and local naming for the same finite surface. Since no other coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 2709.3 (Real closure stream-regular handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{RealClosure}^\uparrow$ *packet reaches the closure verdict only through the ordered handoff*

$$\mathsf{Real}^\uparrow \longrightarrow \mathsf{StreamName}^\uparrow \longrightarrow \mathsf{RegSeqRat}^\uparrow \longrightarrow \mathsf{Topology}^\uparrow \longrightarrow \mathsf{SequentialClosure}^\uparrow$$

displayed by $X, S, Q, T, A, E, H, C, P, N$. The handoff supplies finite observation windows, regular rational readback, neighbourhood membership, sequential approach evidence, transport, replay, provenance, and local naming; it supplies no quotient real, host equality, countable-choice schedule, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. Begin with an accepted carrier $R = (X, T, S, Q, A, E, H, C, P, N)$. A consumer may inspect a Real name only through X, and every finite observation of that name is recorded by the stream-window and regular-rational rows S, Q.

The topological and sequential closure rows then consume the same readback. The neighbourhood row T records the local closure test, while A records the displayed sequential route that meets the test.

Use the exactness row E and the transport-replay rows H, C. They preserve the ordered handoff under classifier comparison and prevent a consumer from replacing the finite route by quotient equality, selected representatives, countable choice, trichotomy, compactness, or ambient completeness.

The provenance and local name rows P, N package exactly this route. Hence every downstream closure read receives the stated L10 handoff and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{RealClosure}^{\uparrow}$ packet is generated from Real point names, topology neighbourhood rows, finite StreamName windows, RegSeqRat readbacks, a SequentialClosure approach row, exactness refusal, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3675.1, Definition 3675.2, and Definition 3675.3 seed the constructive L10 closure carrier, finite NameCert obligation surface, and stream-to-regular-rational handoff for Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient reals, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-neighbourhood exactness, finite stream-window readback, regular rational readback, sequential closure handoff, refusal, transport stability, replay, provenance, and local naming for BEDC.Derived.RealClosureUp.

Chapter 2710

A Concrete Naming Certificate for $\mathsf{IntermediateValueTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{IntermediateValueTheorem}^{\uparrow}$ packet records the constructive intermediate-value theorem as a finite theorem-object certificate rather than as host real analysis. It reads the real-name seal of chapter 38 for $\mathsf{Real}^{\uparrow}$, the regular rational readback route of chapter 301 for $\mathsf{RegSeqRat}^{\uparrow}$, the finite observation windows of chapter 126 for $\mathsf{StreamName}^{\uparrow}$, the endpoint interval packet of chapter 394 for $\mathsf{LocatedInterval}^{\uparrow}$, the graph-and-modulus surface of chapter 130 for $\mathsf{ContinuousMap}^{\uparrow}$, and the closed interval carrier of chapter 1591 for $\mathsf{ClosedBoundedInterval}^{\uparrow}$. The chapter packages the theorem route as BEDC rows: it does not decide classical trichotomy, use Heine–Borel or open-cover compactness, select a zero, identify quotient real names, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2710.1 (Intermediate value theorem carrier)**.** *An* $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, J, F, A, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source interval row, exposing lower and upper endpoint reads, endpoint order, containment ledgers, and interval windows.*
2. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-location row used by the sign route, carrying the same endpoint surface as displayed interval data.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row for the displayed map on the interval.*
4. *A and B are the endpoint sign rows, recording the inspected negative-side and positive-side evidence as finite real-name reads rather than as a trichotomy decision.*
5. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *windows used to inspect endpoint, midpoint, and branch values along the theorem route.*
6. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readbacks attached to those windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the located sign-transition output, reached only after I, J, F, A, B, S, R have been displayed.*
8. *H stores componentwise* hsame *transport for the interval, location, map, sign, window, readback, seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from an intermediate-value request through I, J, F, A, B, S, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{IntermediateValueTheorem}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, J, F, A, B, S, R, E, H, C, P, N$. It has no coordinate for classical trichotomy, Heine–Borel compactness, open-cover compactness, a selected zero, quotient real equality, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremTasteGate single carrier alignment`

Theorem 2710.2 (Intermediate value theorem NameCert obligations)**.** *For every accepted* IntermediateValueTheore *carrier*

$$T = (I, J, F, A, B, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval and endpoint-location rows, continuous-map admission through F, endpoint sign exposure through A and B, finite-window exposure through S, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremCarrier namecert obligations`

Proof. Project the carrier of Definition 3676.1. The accepted packet exposes the closed bounded interval row I, the located endpoint row J, the continuous-map row F, the sign rows A, B, the finite windows S, the regular readbacks R, and the terminal seal row E.

The theorem route is constructive because every sign and value read is a displayed window-readback pair. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback, and the $\mathsf{Real}^{\uparrow}$ row receives only the sealed sign-transition output.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for classical trichotomy, Heine–Borel or open-cover compactness, a selected zero, quotient real equality, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 2710.3 (Intermediate value theorem finite sign-route handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *packet factors through*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longrightarrow \mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only the displayed interval row, endpoint-location row, continuous-map row, endpoint sign rows, finite windows, regular readbacks, Real seal, componentwise transport, replay, provenance, and local naming.

Proof. Start with a Real-facing theorem read. The carrier definition places the terminal seal in row E, and E is downstream of the interval, location, map, sign, window, and readback rows.

Trace the read backward through the finite route. The closed interval row I and located endpoint row J supply the domain surface, the continuous-map row F supplies the displayed modulus and graph data, and the endpoint sign rows A, B are inspected only through S and R before the seal is reached.

Finally H, C, P, N transport, replay, package, and name the same displayed route. There is no second coordinate for a selected zero, quotient real equality, classical trichotomy, Heine–Borel compactness, open-cover compactness, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so every accepted handoff reaches $\mathsf{Real}^{\uparrow}$ only through the displayed rows. □

Falsifiable prediction: Every accepted $\mathsf{IntermediateValueTheorem}^{\uparrow}$ Real-facing read factors through a ClosedBoundedInterval source row, a LocatedInterval endpoint-location row, a ContinuousMap graph-and-modulus row, endpoint sign rows, StreamName finite windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: closedboundedinterval, locatedinterval, continuousmap, streamname, regseqrat, real: $\mathsf{IntermediateValueTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the theorem-level interval source, endpoint-location surface, continuous-map row, endpoint sign ledger, finite window route, regular rational readback, terminal seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{IntermediateValueTheorem}^{\uparrow}$ packet is generated from a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source row, a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint-location row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph-and-modulus row, endpoint sign rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3676.1, Definition 3676.2, and Definition 3676.3 seed the constructive intermediate-value theorem route as a finite NameCert packet over interval, map, sign, window, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close classical trichotomy, Heine–Borel compactness, open-cover compactness, selected zeros, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign visibility, continuous-map routing, finite-window readback, Real-seal handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.IntermediateValueTheor

Chapter 2711

A Concrete Naming Certificate for $\mathsf{FirstExitChannel}^{\uparrow}$

The $\mathsf{FirstExitChannel}^{\uparrow}$ packet realizes the first-exit channel geometry of Definition 9462.1480 as finite BEDC NameCert data. It reads a killed bit-flip walk state through origin state, flipped bit, outside target, target class, transition ledger, transport, replay, provenance, and local naming before any first-exit geometry claim is exported. The packet is a finite science-facing certificate; it is not a bijection with the six-boundary-tile coverage, not a host biological law, not an optimizer for unseen codon tables, and not an unrestricted stochastic-process oracle.

The transition ledger is consumed beside chapter 3470, chapter 3478, chapter 3463, and chapter 3625 because those chapters supply the codon-window partition, trigger activation, cube adjacency, and post-median shell consumer rows.

Falsifiable prediction: Every accepted $\mathsf{FirstExitChannel}^{\uparrow}$ read must display origin state, flipped bit, outside target, target class, transition ledger, transport, replay, provenance, and local naming before any first-exit geometry claim is exported.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, post-median-gate-shell, punctured-cube-skeleton: six-boundary-tile coverage names the codon-window partition, window-coverage trigger algebra names gate activation, post-median gate shell names the downstream shell consumer, and the punctured-cube skeleton names cube adjacency. None is bijective to $\mathsf{FirstExitChannel}^{\uparrow}$ because this packet binds origin row, flipped-bit row, outside-target row, target-class row, transition ledger, transport, replay, provenance, and local naming as one accepted first-exit surface.

Definition 2711.1 (First-exit channel carrier)**.** *A* $\mathsf{FirstExitChannel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, B, T, K, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the origin-state row inside the killed median-closure domain.*
2. *B is the flipped-bit row naming the bit coordinate through which the walk exits.*
3. *T is the outside-target row naming the codon reached after the bit flip.*
4. *K is the target-class row recording the standard genetic-code class of T.*
5. *L is the transition ledger binding O, B, T, and K to the displayed killed-walk boundary half-edge.*

6. *H stores componentwise* hsame *transport for origin, bit, outside target, target class, transition ledger, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the first-exit request through O, B, T, K, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FirstExitChannel}^{\uparrow}$*, the six-boundary-tile coverage, the window-coverage trigger algebra, the post-median gate shell, the punctured-cube skeleton,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, B, T, K, L, H, C, P, N$. It has no coordinate for global codon-table optimality, unseen-window extrapolation, host biochemical causality, an unrestricted Markov-chain oracle, or bridge checking.

Lean4 (definition) `BEDC.Derived.FirstExitChannelUp.FirstExitChannelUp`
Lean4 ✓ `BEDC.Derived.FirstExitChannelUp.FirstExitChannelTasteGate single carrier alignment`

Theorem 2711.2 (First-exit channel NameCert obligations)**.** *For every accepted* $\mathsf{FirstExitChannel}^{\uparrow}$ *carrier*

$$F = (O, B, T, K, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$ *surface consists exactly of carrier admission for the origin-state row O, flipped-bit row B, outside-target row T, target-class row K, transition ledger L, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

Lean4 ✓ `BEDC.Derived.FirstExitChannelUp.FirstExitChannelNamecertObligations`

Proof. Project the carrier of Definition 3677.1. The projection exposes origin, bit, outside target, target class, and transition rows before it exposes the structural rows.

The transition ledger L is the only row that binds the four observational rows into a first-exit boundary half-edge. Thus a consumer cannot replace the displayed exit by an unrelated codon adjacency or by an unrestricted stochastic-process read.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for global codon-table optimality, unseen-window extrapolation, or host biochemical causality, the displayed rows exhaust the NameCert obligations. □

Theorem 2711.3 (First-exit channel geometry readback)**.** *Every accepted first-exit geometry read through* $\mathsf{FirstExitChannel}^{\uparrow}$ *factors through*

$$O, B, T, K \longmapsto L \longmapsto H, C, P, N.$$

Consequently the read may export origin-state distribution rows, flipped-bit rows, outside-target rows, and target-class rows only after the transition ledger and structural NameCert rows are displayed.

Proof. Start with the four observational rows O, B, T, K. They name the origin state, flipped bit, outside target, and target class, but they do not by themselves assert first-exit geometry.

The transition ledger L binds those rows to the killed-walk boundary half-edge. This is the point where the packet distinguishes a first exit from an arbitrary bit flip or a free codon-class lookup.

Finally H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Any geometry claim that bypasses one of these rows, or that asserts an unseen-window optimizer or host biochemical cause, is outside the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$.

Constructive story: A $\mathsf{FirstExitChannel}^{\uparrow}$ packet is generated from origin-state rows, flipped-bit rows, outside-target rows, target-class rows, a transition ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3677.1, Definition 3677.2, and Definition 3677.3 seed the finite first-exit channel carrier, NameCert obligations, and geometry readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global codon-table optimality, unseen-window extrapolation, host biochemical causality, unrestricted Markov-chain semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, origin-bit-target-class ledger exactness, transition replay, provenance, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.FirstExitChannelUp.TasteGate.

Chapter 2712

A Concrete Naming Certificate for $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$

Following Whitney's finite-jet extension theorem and the constructive finite-row presentation of differentiability data, the $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ packet records a finite jet family over a closed carrier together with the compatibility estimates needed by extension consumers. It sits over the real and order-facing rows of chapter 38 and chapter 766, the metric and ball rows of chapter 57 and chapter 1450, the derivative-facing surface of chapter 122, and the Sobolev trace boundary of chapter 3647. The packet is a finite NameCert carrier for jet compatibility and consumer routing; it is not a global smooth extension operator, a selected extension function, host compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2712.1 (Whitney finite-jet extension carrier)**.** *A* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, F, J, A, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *or Euclidean metric carrier whose distance rows supply the only spatial comparisons used by the packet.*
2. *F is the closed-set classifier row naming the finite observation domain on which jets are read.*
3. *J is the finite jet table, containing the displayed value and derivative rows indexed by the admitted multiindex budget.*
4. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison ledger for the Whitney difference quotients, radius bounds, and finite error budgets.*
5. *E is the local extension-admissibility ledger, recording that every requested jet comparison is routed through F, J, A and the displayed metric-ball scales.*
6. *R is the readback row for derivative-facing and Sobolev-trace-facing consumers.*
7. *H records componentwise* hsame *transport for the carrier, closed-set classifier, jet table, order ledger, extension-admissibility ledger, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite jet request through $X, F, J, A, E, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, F, J, A, E, R, H, C, P, N$. It has no coordinate for a host smooth function, a selected extension, compactness, arbitrary choice, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2712.2 (Whitney finite-jet NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier $W = (X, F, J, A, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$ *surface consists exactly of the finite rows named in Definition 3679.1: carrier admission for $X, F, J, A, E, R, H, C, P, N$, closed-set classifier stability, finite jet-table stability, order-ledger comparison for the Whitney estimates, extension-admissibility replay through metric-ball scales, derivative and Sobolev trace readback, transport, provenance, and local naming. No obligation row stores a host extension function, a global smoothness theorem, a selected representative, compactness, choice, quotient equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.WhitneyExtensionFiniteJetUp.TasteGate.WhitneyExtensionFiniteJetTasteGate`
`single carrier alignment`

Proof. Project the accepted carrier of Definition 3679.1. The only geometric data are X and F, and the only analytic data are the finite jet table J, the comparison ledger A, the extension-admissibility row E, and the readback row R.

The NameCert obligations are therefore rowwise. A finite jet request first names the metric carrier and closed-set classifier, then opens the jet table and the order-facing comparison ledger before any admissibility claim is available.

The admissibility row E is the only coordinate that relates two jet reads at a displayed scale. It forces every difference-quotient and remainder comparison through F, J, A and the relevant metric-ball radius rather than through a host extension function.

Finally, H, C, P, N transport, replay, package, and name exactly the same route. Since the carrier contains no coordinate for a selected smooth extension, compactness, arbitrary choice, quotient equality, trichotomy, or ambient Real completeness, none of those objects can become a local obligation. □

Theorem 2712.3 (Whitney finite-jet consumer boundary)**.** *Every* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *consumer reading an accepted* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier must factor through the displayed route*

$$X, F \longmapsto J, A \longmapsto E, R \longmapsto H, C, P, N.$$

The consumer receives finite jet compatibility, order-facing estimates, and derivative or trace readback only along this route. It receives no global extension operator, no selected smooth function, no host compactness theorem, no arbitrary choice principle, no quotient equality, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer read. By Definition 3679.1, the read can inspect only the metric carrier X, the closed-set classifier F, the jet table J, the comparison ledger A, and the admissibility row E before it reaches derivative-facing or trace-facing output.

The row R supplies the public readback boundary. It exposes the finite derivative and trace-facing records already justified by E, but it does not introduce a host smooth function or select an extension outside the displayed rows.

The structural rows preserve the same route. Componentwise transport and continuation replay move the packet through H and C, while P and N record provenance and local naming. Thus every downstream consumer remains inside the finite Whitney jet surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3679.1, Definition 3679.2, and Definition 3679.3 seed the finite Whitney jet packet over metric carriers, closed-set classifier rows, jet tables, order-facing estimates, and derivative or Sobolev trace readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the classical global Whitney theorem, a selected smooth extension operator, compactness, arbitrary choice, quotient equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, jet-table stability, order-ledger comparison, extension-admissibility replay, derivative or trace readback, transport, provenance, and local naming for BEDC.Derived.WhitneyExtensionFinite.

Constructive story: A $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ packet is generated from a MetricUp carrier, a closed-set classifier row, a finite jet table, a RealAlgOrderUp comparison ledger, a local extension-admissibility row, derivative or Sobolev trace readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2713

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionKernel}^{\uparrow}$

Following Hausdorff and Bourbaki-style completion language, the $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ packet records the separated Cauchy-completion boundary as finite BEDC data. It sits between the separated metric classifier of chapter 1411, the complete metric service of chapter 128, the Cauchy-filter completion route of chapter 1569, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Hausdorff completion boundary of chapter 402. The packet names the zero-distance kernel that separates completion reads before any quotient representative, host equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem may be consumed.

Definition 2713.1 (Hausdorff completion kernel carrier)**.** *A* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, Z, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the Cauchy-completion thread row, recording the inspected Cauchy filter, Cauchy sequence, or regular Cauchy name requests that enter the completion boundary.*
2. *Z is the separated zero-distance row supplied by* $\mathsf{SeparatedMetric}^{\uparrow}$*, recording when two displayed completion reads have zero carried distance.*
3. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row, keeping filter-completion requests tied to the same finite Cauchy thread.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing only regular rational windows read from Q and F.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, Z, F, R have been displayed.*
6. *H records componentwise* hsame *transport for the Cauchy thread, zero-distance row, filter-completion handoff, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a separated completion request through Q, Z, F, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{HausdorffCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, Z, F, R, E, H, C, P, N$. It has no coordinate for quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionKernelUp.HausdorffCompletionKernelTasteGate single carrier alignment`

Theorem 2713.2 (Hausdorff completion kernel NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionKe}$ *carrier $K = (Q, Z, F, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, Cauchy-completion thread exposure through Q, separated zero-distance exposure through Z, Cauchy-filter completion handoff through F, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionKernelUp.TasteGate.HausdorffCompletionKernelTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3680.1. The accepted packet exposes the Cauchy thread, separated zero-distance boundary, filter-completion handoff, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows.

The separated completion route is fixed by Q, Z, F, R, E. The Cauchy thread Q supplies the inspected completion source, Z supplies the zero-distance classifier, F binds Cauchy-filter completion reads to that source, R gives the regular rational readback, and E seals the resulting Real-facing boundary.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 2713.3 (Hausdorff completion kernel nonescape)**.** *Every separated completion consumer of an accepted* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ *packet factors through*

$$\mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HausdorffCompletion}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

In particular, equality of completed reads is exposed only by the displayed zero-distance row Z and its transported replay through H, C, P, N.

Proof. Start with a consumer that requests equality or reuse of a completed read. The carrier exposes no equality coordinate before the Cauchy thread Q and the separated zero-distance row Z have been read.

Next read the completion and real-facing rows. The handoff F ties the request to the Cauchy-filter completion route, R gives the regular rational readback used by the real-name boundary, and E is downstream of those rows.

Finally replay the structural data. The rows H, C, P, N preserve and name the same finite route, while the $\mathsf{HausdorffCompletion}^{\uparrow}$ provenance identifies the separated completion boundary. No quotient representative, host equality, selected limit, countable-choice schedule, detached Real equality, or ambient completeness coordinate is available to the consumer. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ separated completion read factors through a Cauchy-completion thread row, a separated zero-distance row, a CauchyFilterCompletion handoff, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: separatedmetric, completemetric, cauchyfiltercompletion, regseqrat, real, hausdorffcompletion: $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cauchy thread, zero-distance kernel, filter-completion handoff, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ packet is generated from a Cauchy-completion thread row, a separated zero-distance row, a CauchyFilterCompletion handoff, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3680.1, Definition 3680.2, and Definition 3680.3 seed the separated Cauchy-completion kernel over displayed completion, zero-distance, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated zero-distance exactness, Cauchy-filter handoff, regular readback, Real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.HausdorffCompletionKernelUp.

Chapter 2714

A Concrete Naming Certificate for RadialWobbleSpectrum$^{\uparrow}$

The RadialWobbleSpectrum$^{\uparrow}$ packet realizes the radial spectral surface of Definition 9462.1448 as finite BEDC NameCert data. It consumes the boundary-tile coverage packet of chapter 3470, the support-closure algebra of chapter 3478, the hyperbolic metric reading of chapter 3474, and the substrate trace invariant of chapter 2360. The packet names radial-shell, spectral-hinge, wobble-edge, and trigger-ledger rows; it is not a biological code table, stochastic evolutionary model, quotient codon semantics, or hidden spectral oracle.

Definition 2714.1 (Radial wobble spectrum carrier)**.** *A* RadialWobbleSpectrum$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, S, H, W, L, M, A, Q, P, N)$$

with the following displayed rows:

1. *T is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-tile coverage row, exposing the finite trigger support read by the radial packet.*
2. *S is the radial-shell row, recording the six displayed states $X_3, X_2, X_1, X_0, Y_R, Y_Y$ and their finite adjacency counts.*
3. *H is the spectral-hinge row, recording the cubic hinge $r^3 - 12r^2 + 26r - 9$ and the Perron read attached to the radial matrix.*
4. *W is the wobble-edge row, recording the f_3 self-loop contribution and its normalized contribution to the killed bit-flip spectrum.*
5. *L is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *trigger ledger, binding support closure, residue accounting, blocker rows, and coverage thresholds to the same radial read.*
6. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row used only as a boundary geometry witness for the radial shells.*
7. *A is the* SubstrateTraceInvariant$^{\uparrow}$ *substrate trace row, preserving the source trace across shell, hinge, wobble, and trigger reads.*
8. *Q records componentwise* hsame *transport and displayed* Cont *replay for T, S, H, W, L, M, A.*
9. *P records* Pkg *provenance over* RadialWobbleSpectrum$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgeb HyperbolicMetric$^{\uparrow}$, SubstrateTraceInvariant$^{\uparrow}$, *the source vision of Definition 9462.1448,* BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, S, H, W, L, M, A, Q, P, N$. It has no coordinate for hidden biological semantics, stochastic transition laws, quotient codon equality, infinite spectral computation, or a detached Perron oracle.

Lean4 ✓ `BEDC.Derived.RadialWobbleSpectrumUp.RadialWobbleSpectrumTasteGate single carrier alignment`

Theorem 2714.2 (Radial wobble spectrum NameCert obligations)**.** *For every accepted* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *carrier $R = (T, S, H, W, L, M, A, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}$ *surface consists exactly of boundary-tile coverage admission through T, radial-shell exposure through S, spectral-hinge exactness through H, wobble-edge accounting through W, trigger-ledger exposure through L, hyperbolic metric boundary reading through M, substrate trace preservation through A, componentwise transport and replay through Q, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3681.1. The accepted packet exposes only the boundary-tile coverage, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, metric, substrate-trace, transport, replay, provenance, and naming rows.

The radial spectrum is carried by S, H, W. The shell row records the six states and adjacency counts, the hinge row records the cubic spectral constraint, and the wobble row records the f_3 contribution before any spectral read may be exported.

The support and geometry rows are T, L, M, A. Boundary-tile coverage and trigger algebra bind the finite support surface, while the hyperbolic metric and substrate trace rows preserve the geometric and trace witnesses without adding a hidden spectral oracle.

Finally Q, P, N transport, replay, source, and name the same finite route. Since no coordinate records biological semantics, stochastic transition laws, quotient codon equality, infinite spectral computation, or detached Perron data, none of those objects belongs to the local NameCert surface. □

Theorem 2714.3 (Radial wobble spectrum trigger nonescape)**.** *Every concrete radial wobble stability read of an accepted* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *packet factors through*

$$\mathsf{SixBoundaryTileCoverage}^{\uparrow} \longrightarrow \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow} \longrightarrow \mathsf{HyperbolicMetric}^{\uparrow} \longrightarrow \mathsf{SubstrateTraceInvariant}^{\uparrow}$$

The radial-shell row S, spectral-hinge row H, wobble-edge row W, and trigger ledger L must be displayed before the packet may claim radial wobble stability.

Proof. Start with the boundary-tile support. The row T supplies finite trigger coverage, and L records the support closure and residue ledger from the trigger algebra.

Next read the radial spectral data. The shell row S gives the six-state adjacency surface, the hinge row H gives the spectral cubic, and W records the wobble self-loop contribution. Removing any one of these rows leaves no carrier coordinate for the claimed radial wobble stability.

Finally read the boundary geometry and trace rows. The hyperbolic metric row M and substrate trace row A preserve the geometric and source-trace witnesses, while Q, P, N transport, replay, source, and name the same route. Thus the packet cannot export a hidden biological table, stochastic transition law, quotient codon equality, infinite spectral computation, or detached Perron oracle. □

Falsifiable prediction: The packet fails if the spectral-hinge row is removed while the same carrier still claims radial wobble stability.

Independence witness: `six_boundary_tile_coverage`, `window_coverage_trigger_algebra`, `hyperbolic_metric`, `substrate_trace_invariant`: the carrier has a radial spectral row and trigger ledger not bijective to any single sibling packet.

The later Lean route for `BEDC.Derived.RadialWobbleSpectrumUp` must provide explicit Nontrivial RadialWobbl and FieldFaithful RadialWobbleSpectrum$^{\uparrow}$ instances preserving $T, S, H, W, L, M, A, Q, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{RadialWobbleSpectrum}^{\uparrow}}$.

Constructive story: A RadialWobbleSpectrum$^{\uparrow}$ packet is generated from boundary-tile coverage, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, hyperbolic metric, substrate trace, componentwise transport, displayed replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{RadialWobbleSpectrum}^{\uparrow}}$, seed)

Scope closed: Definition 3681.1, Definition 3681.2, and Definition 3681.3 seed the finite radial wobble spectrum packet over boundary-tile, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, geometry, and trace rows.

Formal verification: FormalStatus(NameCert$_{\text{RadialWobbleSpectrum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological table identity, stochastic evolutionary modeling, quotient codon equality, infinite spectral computation, detached Perron oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radial-shell exactness, spectral-hinge preservation, wobble-edge accounting, trigger-ledger nonescape, Nontrivial RadialWobbleSpectrumUp, and FieldFaithful RadialWobbleSpectrumUp for BEDC.Derived.RadialWobbleSpectrumUp.

Chapter 2715

A Concrete Naming Certificate for $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$

Following Schwarz–Pick and Ahlfors' hyperbolic metric reading of holomorphic disk maps, the $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ packet names disk-map contraction as finite BEDC data. It is grounded in the unit disk carrier of chapter 377, the Poincare disk transport route of chapter 3455, the hyperbolic metric certificate of chapter 3474, the boundary-dynamics route of chapter 3575, and the observer-expansion vision of chapter 9463. The packet certifies finite disk-map, metric comparison, boundary refusal, transport, provenance, and NameCert rows; it is not an ambient theorem about all analytic self-maps of a completed complex disk.

Definition 2715.1 (Schwarz-Pick disk-contraction carrier)**.** *A* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, F, M, K, B, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for displayed source and target disk points.*
2. *F is the finite disk-map row, recording how the displayed points are transported by the named map.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *readout row for source and target point pairs.*
4. *K is the contraction-comparison row stating that the target hyperbolic readout is no larger than the source readout on the displayed pair.*
5. *B is the boundary-refusal row, separating interior disk-map comparison from Denjoy–Wolff or far-end boundary dynamics.*
6. *R is the route row linking finite disk coordinates, map transport, metric readout, comparison, and boundary refusal.*
7. *H stores componentwise* hsame *transport for disk coordinates, map rows, metric readouts, comparison rows, boundary refusals, route rows, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay of the same disk-map and metric-comparison route.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, F, M, K, B, R, H, C, P, N$. It has no coordinate for a host holomorphic function space, completed complex disk equality, boundary fixed-point selection, open-cover compactness, or unlogged analytic continuation.

Lean4 ✓ `BEDC.Derived.SchwarzPickDiskContractionUp.SchwarzPickDiskContractionTasteGate single carrier alignment`

Theorem 2715.2 (Schwarz-Pick disk-contraction NameCert obligations)**.** *For every accepted* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *carrier*

$$S = (D, F, M, K, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$ *surface consists exactly of carrier admission for displayed disk coordinates, finite disk-map rows, hyperbolic metric readouts, contraction-comparison rows, boundary-refusal rows, route rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3682.1. The projection exposes disk coordinates D, the finite map row F, hyperbolic metric readouts M, the contraction comparison K, and the boundary-refusal row B.

The comparison is read only through R. Thus a consumer first displays the source and target disk coordinates, applies the named finite disk-map row, reads the hyperbolic metric on both displayed pairs, and then consumes the no-expansion comparison.

Transport and replay are confined to H and C. The provenance and local naming rows P, N record exactly the same route, and the carrier has no coordinate for host analytic function spaces, completed disk equality, boundary fixed-point selection, compactness, or unlogged continuation. These rows therefore exhaust the NameCert obligations. □

Theorem 2715.3 (Schwarz-Pick hyperbolic-metric handoff)**.** *Every hyperbolic-metric consumer of an accepted* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *packet reaches the contraction verdict only through the ordered handoff*

$$\mathsf{UnitDisk}^{\uparrow} \longrightarrow \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longrightarrow \mathsf{HyperbolicMetric}^{\uparrow} \longrightarrow \mathsf{SchwarzPickDiskContraction}^{\uparrow}$$

displayed by $D, F, M, K, B, R, H, C, P, N$. The handoff supplies finite disk-map transport, source and target metric readouts, contraction comparison, boundary refusal, provenance, and NameCert data, and it supplies no ambient analytic self-map theorem or boundary fixed-point conclusion.

Proof. Start with the disk-coordinate row D and the finite map row F. These rows display the only source and target points available to the contraction comparison.

The hyperbolic-metric row M then reads the source and target distances on those displayed pairs. The comparison row K consumes those two readouts, while B keeps Denjoy–Wolff boundary dynamics and far-end fixedness outside the interior disk-contraction verdict.

Use the route, transport, and replay rows R, H, C to preserve the same handoff under classifier comparison. The provenance and local naming rows P, N package the finite route, so no host analytic function space, completed disk equality, selected boundary point, compactness argument, or unlogged continuation can enter the metric handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ packet is generated from UnitDisk coordinates, a finite disk-map row, HyperbolicMetric readouts, a contraction-comparison row, boundary refusal, a route row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3682.1, Definition 3682.2, and Definition 3682.3 seed the finite disk-contraction carrier, NameCert obligations, and hyperbolic-metric handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert a host holomorphic-function theorem, completed disk equality, selected boundary fixed point, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-map exactness, hyperbolic metric readback, contraction comparison, boundary refusal, route preservation, transport stability, replay, provenance, and local naming for BEDC.Derived.SchwarzPickDiskContractionUp.

Chapter 2716

A Concrete Naming Certificate for $\mathsf{PoincareTriangleDefect}^{\uparrow}$

The $\mathsf{PoincareTriangleDefect}^{\uparrow}$ packet realizes the Poincare disk representation of chapter 9463 as finite NameCert data. It reads disk-point rows, geodesic-side rows, hyperbolic metric rows, cross-ratio comparison rows, and terminal $\mathsf{Real}^{\uparrow}$ seals through the existing Poincare disk phase-transport, hyperbolic metric, geodesic boundary, hyperbolic cross-ratio, and real-name certificates. The packet names a finite triangle-defect surface, not a host model of the hyperbolic plane, an ambient angle measure, a selected ideal boundary representative, a global field instance, or a completed real-geometric quotient.

Definition 2716.1 (Poincare triangle defect carrier)**.** *A* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, G, M, X, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-point row, displaying three accepted Poincare disk readouts and the local incidence data needed to form a triangle.*
2. *G is the geodesic-side row, read through the boundary-geodesic route and recording the three displayed side carriers.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row used to read side and angle-facing metric comparisons.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the finite cross-ratio comparison data used to orient the defect reading.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed defect value or defect-bound read.*
6. *H records componentwise* hsame *transport for D, G, M, X, R, C, P, N and the local triangle-defect packet shape.*
7. *C records displayed* Cont *replay from disk-point admission through geodesic-side admission, metric reading, cross-ratio reading, and the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{PoincareTriangleDefect}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, M, X, R, H, C, P, N$. It has no coordinate for a host hyperbolic plane, global angle measure, selected boundary representative, field-completion quotient, Lean TasteGate instance, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PoincareTriangleDefectUp.PoincareTriangleDefectTasteGate single carrier alignment`

Theorem 2716.2 (Poincare triangle defect NameCert obligations)**.** *For an accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier* $T = (D, G, M, X, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$ *obligations are exactly disk-point carrier admission, geodesic-side admission, hyperbolic-metric readout, cross-ratio comparison admission,* $\mathsf{Real}^{\uparrow}$ *defect sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3683.1. The tuple displays disk points, geodesic sides, metric rows, cross-ratio rows, a real seal, and the structural rows.

The disk and geodesic rows determine the finite triangle surface. The metric and cross-ratio rows provide the only comparison data from which the defect read can be formed.

The real seal R is downstream of those displayed geometric rows. The transport, replay, provenance, and local name preserve the same packet and introduce no ambient model.

Thus the listed rows exhaust the local certificate obligations. □

Theorem 2716.3 (Poincare triangle defect transport stability)**.** *Let* $T = (D, G, M, X, R, H, C, P, N)$ *be an accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier. Replacing disk-point rows, geodesic-side rows, hyperbolic-metric rows, cross-ratio rows, and the* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible triangle-defect route*

$$D \longmapsto G \longmapsto M \longmapsto X \longmapsto R.$$

The transported packet exports no host hyperbolic plane, ambient angle measure, selected boundary representative, global field quotient, or unrecorded geometric equality.

Proof. The accepted carrier has a single real-facing route: disk points are displayed first, geodesic sides are read from those points, the metric and cross-ratio rows compare the displayed sides, and the real seal records the defect read.

The row H transports each displayed coordinate componentwise. The row C then replays the same continuation from the transported disk and side rows through the transported metric, cross-ratio, and real-seal rows.

The provenance row P and local name N only source and name this route. Since the carrier contains no coordinate for an ambient model or selected boundary representative, the transported defect remains the same finite BEDC packet. □

Theorem 2716.4 (Poincare triangle defect sibling route)**.** *Every accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier reads disk rows through the Poincare disk phase-transport surface of chapter 3455, side rows through the hyperbolic geodesic boundary surface of chapter 3491, comparison rows through the hyperbolic metric surface of chapter 3474 and the hyperbolic cross-ratio surface of chapter 3948, and terminal value rows through the real-name seal of chapter 38. The local packet exports the resulting route only through* $D, G, M, X, R, H, C, P, N$ *and exports no host hyperbolic plane, ambient angle measure, selected boundary representative, or unrecorded quotient geometry.*

Proof. Project the carrier of Definition 3683.1. The disk row D is the only row from which point data may be read, so it is routed through the Poincare disk phase-transport sibling before any side or comparison consumer sees the packet.

Next read the side and comparison rows. The side row G is routed through the hyperbolic geodesic boundary sibling, while M and X are routed through the hyperbolic metric and hyperbolic cross-ratio siblings. These four sibling surfaces provide the displayed geometric input rows; none adds a host hyperbolic plane or ambient angle measure.

Finally read R, H, C, P, N. The terminal value is sealed through the RealUp sibling, and the transport, replay, provenance, and local name rows keep the same sibling route attached to the finite triangle-defect packet. Since no accepted coordinate names a selected boundary representative or quotient geometry, those objects cannot be exported by the sibling route. □

Falsifiable prediction: A candidate implementation of $\mathsf{PoincareTriangleDefect}^{\uparrow}$ that omits any one of the disk-point, geodesic-side, hyperbolic-metric, cross-ratio, or $\mathsf{Real}^{\uparrow}$ seal rows will fail to export a stable defect read under componentwise hsame transport.

Independence witness: The packet depends on the visible rows named poincare-disk-phase-transport, hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, and real; it does not depend on a host hyperbolic plane, ambient angle measure, selected boundary representative, or global quotient geometry. Lean TasteGate support will require explicit Nontrivial and FieldFaithful instances before the geometric rows can be checked as a typed finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareTriangleDefect}^{\uparrow}$ packet is generated from disk-point rows, geodesic-side rows, HyperbolicMetric rows, HyperbolicCrossRatio rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3683.1, Definition 3683.2, Definition 3683.3, and Definition 3683.4 seed the finite Poincare disk triangle-defect packet and its sibling dependency route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host hyperbolic plane, ambient angle measure, selected ideal boundary representative, global field quotient, Lean TasteGate instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-point admission, geodesic-side admission, hyperbolic-metric readout, cross-ratio comparison, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.PoincareTriangleDefectUp.

Chapter 2717

A Concrete Naming Certificate for $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed regular Cauchy source carries a boundedness ledger before it is sealed as a Real-completion object. It sits beside the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the bounded regular Cauchy route of chapter 1697, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet supplies an L10 completion-facing boundedness criterion; it is not a quotient stream, not a selected limit, not countable choice, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2717.1 (Cauchy boundedness criterion carrier)**.** *A* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the Cauchy source row, naming the regular rational sequence whose finite observations are admitted by the packet.*
2. *B is the boundedness ledger, recording the displayed initial bound, tail bound, and comparison rows that make the source a* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *object.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source at the requested precision and tail threshold.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite window and boundedness ledger.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the precision budget, bound comparison, and tail estimate consumed by the readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, B, W, R, D have been displayed.*
7. *H stores componentwise* hsame *transport for the Cauchy source, boundedness ledger, window row, readback row, dyadic tolerance row, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a bounded Cauchy request through $S, B, W, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$, $\mathsf{BoundedRegularCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, B, W, R, D, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected Real limit, countable choice, trichotomy, a host supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyBoundednessCriterionUp.CauchyBoundednessCriterionTasteGate single carrier alignment`

Theorem 2717.2 (Cauchy boundedness criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row S, boundedness ledger B,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3684.1. The analytic rows are the source S, boundedness ledger B, window row W, readback row R, dyadic tolerance row D, and Real seal E.

The boundedness obligation is read only from B together with the dyadic row D. The ledger records the displayed bound comparisons and tail estimates, while D records the precision budget consumed by those comparisons.

The regular readback is downstream of the same finite window. Thus R is a $\mathsf{RegSeqRat}^{\uparrow}$ row obtained through W, B, D, not a host sequence, quotient stream, selected limit, or completeness argument.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for countable choice, trichotomy, host suprema, quotient-stream equality, selected Real limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 2717.3 (Cauchy boundedness criterion Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing completion read exported by an accepted* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *packet factors through*

$$S \longmapsto B \longmapsto W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently a completion consumer may use the displayed Cauchy source, boundedness ledger, finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local structural rows, but it receives no quotient-stream equality, selected limit, countable choice, trichotomy, host supremum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the displayed source S. The source is admissible for the packet only as a Cauchy row whose boundedness information is supplied by the ledger B.

The finite observation then passes through W and D. The row W exposes the inspected stream window, while D supplies the dyadic tolerance and bound-comparison budget for the same request.

Only after those rows have been displayed does the packet expose R and then E. The row R is the regular rational readback, and E is the terminal Real seal served by that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Since none of the excluded objects occurs in the carrier, no completion-facing read can bypass the bounded finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ packet is generated from a Cauchy source row, a boundedness ledger, a StreamName finite-window row, a RegSeqRat readback row, a DyadicRatCore tolerance row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3684.1, Definition 3684.2, and Definition 3684.3 seed the finite boundedness criterion for regular Cauchy sources and Real-completion handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected Real limits, countable choice, trichotomy, host suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundedness-ledger exactness, StreamName window admission, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.CauchyBoundednessCriterionUp.

Chapter 2718

A Concrete Naming Certificate for $\mathsf{LocatedMetricSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet records a constructive metric space whose distance reads are located before they are consumed by the L10 Phase Real and Completion route. It refines the metric source of chapter 57 by forcing every distance observation through the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet supplies a finite NameCert surface for completion-facing consumers without importing host equality of points, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2718.1 (Located metric-space carrier)**.** *A* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric-source carrier row, including the local point classifier and the finite point-observation windows on which distance requests are opened.*
2. *M is the metric source row inherited from chapter 57; it displays the distance witness inspected by each accepted pair of point observations.*
3. *D is the located-distance witness and dyadic tolerance ledger. For each displayed distance request and positive tolerance, it records the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison data that locates the requested distance at that scale.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It packages the dyadic comparison windows in D as regular rational distance names before any real-facing consumer can read them.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *readback and seal boundary reached only after the metric source, located-distance witness, dyadic tolerance ledger, and regular rational readback have all been displayed.*
6. *H stores componentwise* hsame *transport for point observations, metric source rows, located-distance witnesses, dyadic tolerance ledgers, regular rational readbacks, real seals, replay rows, provenance rows, and local naming rows.*
7. *C stores displayed* Cont *replay from point observations to metric witnesses, from metric witnesses to located dyadic comparisons, from those comparisons to* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and from the readbacks to the* $\mathsf{Real}^{\uparrow}$ *seal.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted located metric-space packets only through $X, M, D, R, E, H, C, P, N$. It has no coordinate for host equality of points, quotient real equality, quotient completion, selected limits, total trichotomy of distances, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedMetricSpaceUp.LocatedMetricSpaceTasteGate single carrier alignment`

Theorem 2718.2 (Located metric-space NameCert obligations)**.** *For every accepted* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *carrier*

$$L = (X, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, distance-witness readback through M, located-distance and dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3685.1. The point observations and their local classifier are read from X, and every distance request is forced through the metric source row M before locatedness data can be inspected.

The located-distance obligation is the role of D. A positive tolerance request reads the dyadic comparison window stored in the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, so the packet gives a finite locatedness witness rather than a semantic real-order decision.

The readback obligation then passes through R and E. Row R packages the same dyadic windows as regular rational distance names, and row E exposes the real seal only after that regular readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name exactly the same rows. Since the carrier contains no coordinate for host point equality, quotient equality, quotient completion, selected limits, trichotomy, choice, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2718.3 (L10 located metric-space completion route)**.** *Every L10 Phase Real and Completion consumer that reads an accepted* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *packet factors through*

$$X \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N. Thus completion-facing distance reads pass through the metric source, located-distance witness, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before any completion data is consumed.*

Proof. Begin with a completion-facing distance request. The source row X supplies only displayed point observations, so the request must use the metric source row M to obtain the finite distance witness certified by the metric sibling.

Locatedness is supplied by D. The dyadic tolerance ledger binds the requested radius to the same metric witness, and it exposes only finite $\mathsf{DyadicRatCore}^{\uparrow}$ comparison data at the requested scale.

The L10 real-facing route is then determined by R and E. The regular rational readback packages the dyadic windows as distance-name data, and the real seal is reached only after that package is visible.

Rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because no excluded quotient, selection, host equality, choice, trichotomy, or ambient-completeness coordinate appears in the carrier, every completion-facing read remains on the displayed L10 route. □

Falsifiable prediction: Every L10 completion-facing read from a $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet factors through the displayed metric source, located-distance witness, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local naming row before completion data is consumed.

Independence witness: metric, dyadicratcore, regseqrat, real: $\mathsf{LocatedMetricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, located-distance witness, dyadic tolerance ledger, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet is generated from a metric source, located-distance witness, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3685.1, Definition 3685.2, and Definition 3685.3 seed the located metric-space carrier and its L10 real-and-completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host equality of points, quotient real equality, quotient completion, selected limits, total trichotomy of distances, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-distance witness exactness, dyadic tolerance ledger accounting, RegSeqRat readback, RealUp seal handoff, transport replay, and ledger refusal for BEDC.Derived.LocatedMetricSpaceU

Chapter 2719

A Concrete Naming Certificate for $\mathsf{ProperMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{ProperMetricSpace}^{\uparrow}$ packet records the constructive metric-space interface in which displayed closed metric balls carry compactness-style readback for completion consumers. It sits after the metric carrier of chapter 57, the open-ball boundary of chapter 1450, the closed-ball containment surface of chapter 3464, the compact-metric certificate of chapter 129, the bounded-family completion route of chapter 1896, and the terminal real-name seal of chapter 38. The packet is a finite closed-ball compactness ledger, not an open-cover compactness theorem, a selected finite subcover, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2719.1 (Proper metric space carrier)**.** *A* $\mathsf{ProperMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P_{\mathrm{pms}} = (X, B, R, K, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *row carrying the point classifier, distance witnesses, and metric transport surface.*
2. *B is the displayed family of* $\mathsf{ClosedMetricBall}^{\uparrow}$ *rows indexed by the accepted center-radius windows of X.*
3. *R is the radius ledger, recording nonnegative radius rows, strict-boundary handoffs to* $\mathsf{MetricBall}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *comparison readback.*
4. *K assigns compactness-style* $\mathsf{CompactMetric}^{\uparrow}$ *readback to each displayed closed ball in B.*
5. *O records compatibility between the closed-ball family and the* $\mathsf{CauchyBornology}^{\uparrow}$ *bounded-family route consumed by completion-facing readers.*
6. *H records componentwise* hsame *transport for the metric row, closed-ball family, radius ledger, compactness readback, bornology compatibility, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a proper-metric request through X, B, R, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ProperMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{ClosedMetricBall}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CauchyBornology}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and exposes no host subset of points, open-cover compactness object, selected compactness witness outside the displayed closed ball, quotient metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.ProperMetricSpaceUp.TasteGate.ProperMetricSpaceTasteGate single carrier alignment`

Theorem 2719.2 (Proper metric space NameCert obligations)**.** *For every accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *carrier*

$$P_{\mathrm{pms}} = (X, B, R, K, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric row* X*, closed-ball family row* B*, radius ledger* R*, compactness readback* K*, bornology compatibility row* O*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*; center-radius compatibility between* X, B, R*; compact readback exactness for each displayed ball through* K*; completion-facing bounded-family compatibility through* O*; and non-escape for host subsets, open-cover compactness, selected subcovers, quotient point equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3686.1. The metric row, closed-ball family, radius ledger, compact readback, bornology compatibility, and structural rows are all displayed before a consumer reads the local naming row.

Center-radius compatibility is forced by the order of the rows. The row X supplies the metric source, B supplies only closed balls over that source, and R supplies the radius comparisons and strict-boundary handoffs used by $\mathsf{MetricBall}^{\uparrow}$ consumers.

Compactness readback is attached ball by ball. The row K is indexed by the displayed closed-ball family B, so a consumer cannot replace it by an open-cover compactness proof or by a compactness witness for an undisplayed ball.

Finally O, H, C, P, N preserve the same route for completion-facing and bounded-family consumers. Since the carrier has no coordinate for the excluded host principles, the listed rows exhaust the NameCert obligations. □

Theorem 2719.3 (Proper metric space closed-ball compactness readback)**.** *Every compactness-style read of a displayed closed ball in an accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{ClosedMetricBall}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The $\mathsf{ClosedMetricBall}^{\uparrow}$ *row supplies the center-radius containment, the radius ledger supplies the ordered-real comparison, and the* $\mathsf{CompactMetric}^{\uparrow}$ *row supplies the compactness readback attached to that same displayed ball. The readback exports no open-cover compactness theorem, selected finite subcover, quotient point equality, host closed subset, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the closed-ball family row B of Definition 3686.1. Each accepted ball read is indexed by a center-radius window over the metric row X and by a radius comparison in R.

Read the compactness row K only after that ball is fixed. Its index is the displayed closed-ball row, so compactness-style data are attached to the same metric source, center, radius, and containment comparison.

The structural rows H, C, P, N transport, replay, source, and name this finite route. No row names an open cover, a chosen subcover, a quotient point equality, or an ambient completeness theorem. □

Theorem 2719.4 (Proper metric space ball-bornology boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *packet reads bounded-family data through the displayed boundary*

$$\mathsf{MetricBall}^{\uparrow},\quad \mathsf{ClosedMetricBall}^{\uparrow},\quad \mathsf{CauchyBornology}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Open-ball handoffs use the strict-radius side of R, closed-ball handoffs use B, bounded-family requests use O, and compactness-style requests use K. Thus the consumer receives a finite ball and bornology boundary rather than a host bounded set, selected bound, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the radius ledger R. Its strict-radius entries are the only route to $\mathsf{MetricBall}^{\uparrow}$ consumers, while its non-strict entries are the radius comparisons used by the $\mathsf{ClosedMetricBall}^{\uparrow}$ family B.

The bornology compatibility row O is downstream of those ball rows. A completion-facing consumer therefore reads bounded-family data only after the metric row, radius ledger, and ball family have been displayed.

Compactness-style information remains attached through K. The same transport, replay, provenance, and local naming rows preserve the boundary, so the route cannot turn into a host bounded set, selected bound, open-cover compactness proof, or ambient Real-completeness datum. □

Theorem 2719.5 (Proper metric space compact-ball route)**.** *Every completion-facing proper-metric consumer must expose the metric row X, the displayed* $\mathsf{ClosedMetricBall}^{\uparrow}$ *family B, the* $\mathsf{Real}^{\uparrow}$ *radius ledger R, the* $\mathsf{CompactMetric}^{\uparrow}$ *readback row K, the* $\mathsf{CauchyBornology}^{\uparrow}$ *bounded-family route O, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N before reading compact-ball data.*

The compact-ball route is therefore the displayed finite packet

$$X, B, R, K, O, H, C, P, N$$

read in the order forced by Definition 3686.1. It refuses open-cover compactness, selected finite subcovers, quotient point equality, host bounded sets, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3686.1. The only available completion-facing rows are the metric source, closed-ball family, radius ledger, compact readback, bornology route, transport, replay, provenance, and local naming rows.

Use the closed-ball compactness readback theorem, Definition 3686.3, to attach compactness-style information to a displayed closed ball over X, B, R. Then compose the ball-bornology boundary of Definition 3686.4, which routes bounded-family consumers through O while keeping the compact readback attached through K.

The structural rows H, C, P, N preserve the same finite route. Since no carrier coordinate names open covers, selected finite subcovers, quotient point equality, host bounded sets, or ambient Real completeness, a compact-ball consumer cannot read any of those principles from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProperMetricSpace}^{\uparrow}$ packet is generated from a MetricUp row, a displayed ClosedMetricBallUp family, a RealUp radius ledger, CompactMetricUp readback for each displayed closed ball, CauchyBornologyUp compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3686.1, Definition 3686.2, Definition 3686.3, Definition 3686.4, and Definition 3686.5 seed the constructive closed-ball compactness ledger for completion-facing proper metric space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, selected subcovers, quotient metric equality, host bounded sets, arbitrary compactness outside displayed closed balls, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, center-radius compatibility, compact readback exactness, ball-bornology boundary routing, transport, replay, provenance, and local NameCert exactness for BEDC.Derived.ProperMetricSpace

Chapter 2720

A Concrete Naming Certificate for $\mathsf{CompleteMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive metric completion, the $\mathsf{CompleteMetricSpace}^{\uparrow}$ packet records a metric carrier together with finite Cauchy-limit service data visible to BEDC consumers. It sits beside the constructive metric surface of chapter 3673, the separated metric route of chapter 2355, the regular Cauchy completion unit of chapter 3646, and the Real-facing finite-window route of chapter 38. The packet supplies an L10 completion-facing neighbour for separated metric, Baire, and completion chapters; it is not a quotient completion, not a selected-limit principle, not countable choice, not host equality, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2720.1 (Complete metric-space carrier)**.** *A* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, C, E, L, U, H, T, P, N)$$

with the following displayed rows:

1. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *row, exposing only the point-name and distance-read surfaces consumed by the packet.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording that accepted Cauchy requests are answered through displayed limit rows.*
3. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *boundary row, used only to identify regular Cauchy names that have the same displayed tail behaviour.*
4. *L is the limit-row ledger. It names the proposed limit point, the convergence-distance rows, and the precision requests consumed by the limit service.*
5. *U is the uniform Cauchy-service route through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; it exposes finite windows, regular rational readbacks, and the terminal Real seal in that order.*
6. *H stores componentwise* hsame *transport for the metric row, completeness row, Cauchy-equivalence boundary, limit ledger, uniform service route, replay, provenance, and local naming rows.*
7. *T stores displayed* Cont *replay from Cauchy-request inspection through M, C, E, L, U, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, C, E, L, U, H, T, P, N$. It has no coordinate for quotient completions, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteMetricSpaceUp.CompleteMetricSpaceTasteGate single carrier alignment`

Theorem 2720.2 (Complete metric-space NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *carrier*

$$K = (M, C, E, L, U, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric row M, the complete-metric service row C, the Cauchy-sequence equivalence boundary E, the limit-row ledger L, the uniform Cauchy-service route U, componentwise transport H, displayed replay T, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3687.1. The projection exposes exactly the metric surface M, completeness service C, Cauchy-equivalence boundary E, limit ledger L, uniform service route U, and structural rows H, T, P, N.

The completeness service is read only through C and L. A consumer receives a displayed limit row and convergence-distance ledger, not a quotient completion object or an externally selected representative.

The Cauchy boundary is confined to E, while the finite observation route is confined to U. Thus tail comparison passes through $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any limit read is exported.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, host equality, trichotomy, selected limits, quotient completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 2720.3 (Complete metric-space Cauchy-limit route)**.** *Every Cauchy-limit read exported by an accepted* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *packet factors through*

$$M \longmapsto E \longmapsto U \longmapsto C, L \longmapsto H, T, P, N.$$

Consequently separated metric, Baire, and completion-facing consumers may use only the displayed metric row, Cauchy-equivalence boundary, finite-window regular readback route, complete-metric service row, limit ledger, transport, replay, provenance, and local naming.

Proof. Begin with the metric row M. It supplies the point-name and distance-read surface, but it does not by itself export a limit object.

The Cauchy request then passes through E and U. The row E fixes the tail-equivalence boundary for the displayed Cauchy name, while U exposes the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and terminal $\mathsf{Real}^{\uparrow}$ seal needed for the convergence-distance route.

Only after those rows are displayed may the packet read C and L. The row C supplies the complete-metric service, and L records the proposed limit point and convergence ledger for the same request.

The rows H, T, P, N preserve that route under transport, replay, provenance, and local naming. Any route that appeals to quotient completions, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness has no displayed coordinate in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricSpace}^{\uparrow}$ packet is generated from a Metric row, a CompleteMetric service row, a CauchySequenceEquivalence boundary, a limit ledger, a RegSeqRat–StreamName–Real uniform service route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3687.1, Definition 3687.2, and Definition 3687.3 seed the finite complete metric-space carrier, NameCert obligation surface, and Cauchy-limit route for separated metric, Baire, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-equivalence exactness, finite-window regular readback, complete-metric service confinement, limit-ledger exactness, and non-escape for BEDC.Derived.CompleteMetricSpaceUp.

The $\mathsf{Cauchy}^{\uparrow}$ hub routes finite completion reads through the sibling lattice of chapter 126, chapter 301, chapter 38, chapter 3794, and chapter 1569 before any completed-object consumer is reached.

Chapter 2721

A Concrete Naming Certificate for $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packet records the constructive complete ordered-field interface by finite regular-Cauchy and order-algebra rows rather than by an ambient completed real line. It sits beside the terminal Real naming seal of chapter 38, the regular rational sequence packet of chapter 301, the finite stream-window surface of chapter 126, the dyadic arithmetic substrate of chapter 291, the constructive algebraic order packet of chapter 766, and the Bishop real completion route of chapter 3665. The chapter exposes the common interface downstream completion and metric chapters can cite when they need Real scalars, regular Cauchy limit rows, and ordered-field compatibility in one NameCert packet.

Definition 2721.1 (Cauchy-complete ordered-field carrier)**.** *A* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar row, read only through terminal Real name seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-precision ledger used by the regular Cauchy and order rows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row for rational windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row.*
5. *B is the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *completion route from D, S, Q into the Real seal R.*
6. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *algebra-order row, recording addition, multiplication, zero, one, positivity, and order compatibility only as displayed finite ledger entries.*
7. *L is the regular-Cauchy limit ledger naming accepted Cauchy requests, precision windows, candidate limit rows, and convergence readbacks through D, S, Q, B, R.*
8. *O is the order-algebra compatibility ledger, recording that limit rows preserve the displayed ordered-field operations through A and not through host equality.*
9. *H records componentwise* hsame *transport for scalar, dyadic, stream, regular-Cauchy, completion, algebra-order, limit, compatibility, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from precision requests through regular Cauchy readback, Bishop completion, Real sealing, algebra-order compatibility, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, S, Q, B, A, L, O, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, trichotomy, open-cover compactness, selected representatives, or choice.*

Theorem 2721.2 (Cauchy-complete ordered-field NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Real scalar row R, dyadic precision ledger D, stream-window row S, regular rational sequence row Q, Bishop completion route B, algebra-order row A, regular-Cauchy limit ledger L, order-algebra compatibility ledger O, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3688.1. The projection exposes exactly the Real scalar, dyadic precision, stream-window, regular rational, Bishop completion, algebra-order, limit, compatibility, transport, replay, provenance, and naming rows.

The completion-facing rows are read only through D, S, Q, B, R, L. A consumer receives finite precision requests, finite rational windows, regular readbacks, the Bishop completion route, the terminal Real seal, and a displayed limit ledger.

The ordered-field rows are read only through A and O. They record operation and order compatibility for the same displayed limit rows, without adding host equality, quotient representatives, trichotomy, open-cover compactness, selected limits, or choice.

Finally H, C, P, N transport, replay, source, and name that finite surface. Since the carrier has no further coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 2721.3 (Regular-Cauchy limit route)**.** *Every limit read exported by an accepted* CauchyCompleteOrderedF *packet factors through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopRealCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

and then through the displayed limit ledger L and structural rows H, C, P, N. Thus downstream completion and metric consumers may cite one finite interface for regular-Cauchy limit service, but they receive no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient equality, selected representative, trichotomy, open-cover compactness, or choice principle.*

Proof. Begin with the precision ledger D. It supplies only the positive dyadic rows inspected by the regular-Cauchy request.

The request then passes through S and Q. The row S names the finite rational windows, and Q supplies the regular rational readback for those windows.

Only after that readback may the packet use B and R. The Bishop completion row B routes the same data to the terminal Real seal R, while L records the candidate limit and convergence readbacks for the accepted request.

The rows H, C, P, N preserve, replay, source, and name the route. No excluded ambient completeness or choice coordinate appears in the carrier, so every exported limit read is confined to the stated finite path. □

Theorem 2721.4 (Order-algebra compatibility route)**.** *Every algebraic or order read exported by an accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *packet factors through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row A together with the regular-Cauchy limit ledger L and the compatibility ledger O. Addition, multiplication, zero, one, positivity, and order comparison are available only as displayed rows compatible with the route through D, S, Q, B, R, and not as ambient host-field operations.*

Proof. Read the algebra-order row A. It is the only coordinate naming field operations and order comparison, so a consumer cannot obtain algebraic structure from the host Real carrier.

Next read the limit ledger L. The ledger ties the operation and order requests to the same regular-Cauchy route through D, S, Q, B, R used for limit service.

Use the compatibility ledger O to bind the two surfaces. It records preservation of addition, multiplication, zero, one, positivity, and order comparison for the displayed limit rows.

Transport and replay are again confined to H and C, with P and N supplying provenance and local naming. Thus the compatibility surface remains finite and supplies no quotient equality, trichotomy, open-cover compactness, selected representative, ambient completeness, or choice. □

Theorem 2721.5 (Cauchy-complete ordered-field regular-Cauchy existence)**.** *For every accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

every regular-Cauchy limit request admitted by K is presented only by a finite route through

$$D \longrightarrow S \longrightarrow Q \longrightarrow B \longrightarrow R$$

and by the limit ledger L. The existence surface therefore names limit service as displayed precision, stream-window, regular-sequence, Bishop-completion, Real-seal, and ledger rows, with no quotient equality, selected representative, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with an admitted limit request. The carrier projection gives the dyadic precision row D, stream-window row S, regular rational sequence row Q, Bishop completion route B, Real seal R, and limit ledger L as the only completion-facing coordinates.

The request is first read against D, then against the finite observation row S, then against the regular-sequence row Q. The route can enter the Real naming surface only through B and then R.

The ledger L records the accepted request, the precision windows, the candidate limit row, and the convergence readbacks for that same route. Since the carrier has no quotient, representative-selection, choice, trichotomy, or ambient-completeness coordinate, the displayed rows are the whole existence surface. □

Theorem 2721.6 (Cauchy-complete ordered-field order-limit transport)**.** *For every accepted* $\mathsf{CauchyCompleteOrder}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

every algebraic and order read on a limit row of L factors through the algebra-order row A and the compatibility ledger O. The transport rows H, replay rows C, provenance row P, and local naming row N preserve that same finite route through D, S, Q, B, R, L, A, O and add no host equality, selected representative, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read a limit row from L. Its completion data has already been confined to the route through D, S, Q, B, R.

Algebraic and order observations can then use only A and O. The row A names addition, multiplication, zero, one, positivity, and comparison, while O records that those reads are compatible with the same displayed limit rows.

The structural rows do not create a second route. The transport row H preserves the componentwise names, C replays the displayed dependencies, P records provenance, and N supplies the local certificate name. Hence order-limit transport stays on the finite rows and does not expose host-field equality, representatives, choice, trichotomy, or ambient completeness. □

Theorem 2721.7 (CauchyCompleteOrderedField scoped kernel route)**.** *For every accepted* CauchyCompleteOrdere *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

the consumer-visible kernel route consists exactly of Real scalar exposure through R, dyadic precision through D, finite $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, Bishop completion handoff through B,* $\mathsf{RealAlgOrder}^{\uparrow}$ *operations through A, regular-Cauchy limit ledger reads through L, order-algebra compatibility through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *row N. No accepted carrier exports ambient Real completeness, quotient streams, host field operations, trichotomy, or selected limits beyond those displayed rows.*

Proof. Project the carrier of Definition 3688.1. The projection exposes precisely the rows $R, D, S, Q, B, A, L, O, H, C$ so any consumer route must begin inside the displayed finite packet.

Apply the NameCert obligations of Definition 3688.2. They identify the Real scalar, dyadic precision, stream-window, regular-rational, Bishop completion, algebra-order, limit, compatibility, transport, replay, provenance, and local naming rows as the full obligation surface.

The completion side is then fixed by Definition 3688.3. The route to a Real-facing limit read passes through D, S, Q, B, R, L and the structural rows, with no quotient stream or selected-limit coordinate.

The algebra and order side is fixed by Definition 3688.4. Operations and order comparisons are read only through A and O against the same displayed limit ledger.

Since no carrier coordinate remains after H, C, P, N, the route refuses ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, host field operations, trichotomy, and selected limits. The scoped dependency surface is exactly the finite row list stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packet is generated from Real scalar seals, dyadic precision rows, StreamName windows, RegSeqRat readbacks, a BishopRealCompletion route, RealAlgOrder operation and order rows, regular-Cauchy limit ledgers, compatibility ledgers, componentwise transport, replay, provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 3688.1, Definition 3688.2, Definition 3688.3, Definition 3688.4, Definition 3688.5, Definition 3688.6, and Definition 3688.7 close the explicit finite scoped surface for Real, regular-Cauchy existence, order-limit transport, completion, and metric consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, trichotomy, open-cover compactness, selected representatives, choice, Lean verification, or bridge checking.

Upgrade path: Public closure requires checked consumer exports and downstream bridge rows for BEDC.Derived.CauchyCompleteOrdered

The $\mathsf{Cauchy}^{\uparrow}$ hub exposes a displayed Real and Completion route only after finite stream windows and regular rational readback rows have been made visible. This child route records the dependency on chapter 126, chapter 301, and chapter 38 without adding substantive prose to the hub.

Theorem 2721.8 (Cauchy displayed RegSeqRat route)**.** *Any* $\mathsf{Cauchy}^{\uparrow}$ *route used by the L10* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *surface must pass through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows before a* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route carries only displayed* BHist*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows, and it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyDisplayedRegseqratRoute`

Proof. Begin with a Real- or Completion-facing Cauchy read. The finite window surface of chapter 126 is the first visible observation boundary, so the consumer cannot treat the Cauchy source as an uninspected host sequence.

Read those windows through chapter 301. The regular rational readback row supplies the displayed tail data required before any terminal seal can be used.

Finally consume the terminal row of chapter 38. The structural rows transport, replay, source, and name the same finite route, so the Cauchy handoff reaches the L10 Real and Completion surface only after StreamName windows and RegSeqRat readback have been displayed. □

Theorem 2721.9 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 2721.10 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 2721.11 (Cauchy regular-source completion dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *read consumed by an L10 completion-facing packet is public only through the displayed dependency route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first exposes the regular-rational tail budget of chapter 301, then the metric-distance comparison used by the $\mathsf{MetricCompletion}^{\uparrow}$ *boundary of chapter 1506, and only then the terminal real seal of chapter 38. It exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child carriers. The hub contributes only displayed finite rows, so a completion-facing consumer must first inspect the regular-rational tail budget that names the tail data and tolerance requests.

Next compare distances at the metric-completion boundary. The $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives the same displayed regular-source route and metric-distance comparison; it does not receive a quotient representative or a selected completed point.

Finally consume the terminal $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the regular tail budget and the metric-completion comparison have been displayed, the public route is a dependency edge through sibling chapters rather than an ambient completeness principle. □

Theorem 2721.12 (Cauchy filter-base net-convergence route)**.** *Every completion-facing consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet that reads Cauchy finite-window evidence through filter or net interfaces must first pass through the displayed routes of chapter 1601, chapter 1915, and chapter 1667. The route exposes the same finite windows, tolerance ledger, tail comparisons, componentwise transport, replay, provenance, and local naming rows before any filter-base or net-convergence*

completion consumer inspects the packet. It exports no selected limit, quotient completion equality, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or detached net convergence theorem.*

Proof. Project the accepted $\mathsf{Cauchy}^{\uparrow}$ packet to its finite-window and tolerance rows. These are the only rows available to a downstream filter or net consumer.

The filter-base read is routed through the finite basis surface named by chapter 1601 and the Cauchy-specific basis of chapter 1915. The net read then passes through chapter 1667, which consumes the same displayed rows rather than a separate completion theorem.

Since the route never adds a coordinate for selected limits, quotient completion equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or detached convergence, those objects remain outside the exported Cauchy packet. □

Theorem 2721.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Every completion-facing $\mathsf{Cauchy}^{\uparrow}$ read first exposes the regular rational row of chapter 301, then passes the same finite-window and tolerance data to the $\mathsf{MetricCompletion}^{\uparrow}$ boundary of chapter 1506, and only afterward reaches the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Thus the visible dependency route is $\mathsf{Cauchy}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$, not a self-contained hub read, quotient stream, selected limit, or ambient completeness principle.

Theorem 2721.14 (Cauchy regular-fusion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that uses a fused regular-Cauchy tail passes through the sibling* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *packet before the Cauchy hub exports the route. The displayed route consumes the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *evidence,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any completion consumer may inspect the fused tail.*

Proof. Read the fused tail as a sibling packet rather than as a hidden coordinate of the Cauchy hub. The $\mathsf{RegularCauchyFusion}^{\uparrow}$ row supplies the displayed fusion boundary, while $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ expose the finite observation windows, regular-rational readback, and dyadic tolerance rows used by the tail.

The completion-facing consumer then reaches the $\mathsf{Real}^{\uparrow}$ seal only after the finite route has been replayed through BHist, hsame, Cont, Pkg, and NameCert. Thus the Cauchy hub exports a sibling dependency route for fused regular-Cauchy tails, not a quotient stream, selected limit, choice schedule, trichotomy oracle, or ambient completeness row. □

The $\mathsf{Cauchy}^{\uparrow}$ carrier reads regular rational tails through chapter 301 and finite observation schedules through chapter 126. The tolerance rows supplied by chapter 291 and the terminal sealing of chapter 38 remain downstream of that lattice, so the hub exposes the RegSeqRat and StreamName dependencies before completion-facing consumers inspect the Real boundary.

Theorem 2721.15 (Cauchy RegSeqRat and Real sibling lattice repair)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *route used by this hub depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling for regular rational windows and on the* $\mathsf{Real}^{\uparrow}$ *sibling for the terminal seal boundary. Thus every completion-facing consumer first reads the regular rational dependency of chapter 301 and then the Real seal of chapter 38; the hub exports no quotient stream, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row of its own.*

Proof. Use the displayed RegSeqRat dependency files already routed through the hub. They force the Cauchy read through regular rational windows before any completion-facing row can be interpreted.

Then apply the terminal Real sibling route named by chapter 38. The seal boundary is therefore read as a sibling obligation after RegSeqRat evidence, not as hidden content of the top-level hub. □

The $\mathsf{Cauchy}^{\uparrow}$ hub also exposes the forward lattice used by regular-real consumers: the displayed Cauchy source is first read as $\mathsf{RegSeqRat}^{\uparrow}$ data and then handed to the $\mathsf{RegularCauchyReal}^{\uparrow}$ packet before any terminal real-facing consumer treats it as a completed object.

Theorem 2721.16 (Cauchy RegSeqRat to RegularCauchyReal sibling route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *route consumed by a regular-real boundary must pass through the regular rational readback of chapter 301 before it reaches the* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier of chapter 3546. The route exports a displayed L10 handoff through finite windows, regular readback, transport, replay, provenance, and local naming; it exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the hub through its existing regular-source children. Those children expose the Cauchy data as finite observation windows and regular rational rows before any real-facing seal can be consumed.

The $\mathsf{RegSeqRat}^{\uparrow}$ row is then the source read by the $\mathsf{RegularCauchyReal}^{\uparrow}$ carrier. Its sibling chapter adds the StreamName, dyadic tolerance, convergence handoff, Real seal, transport, replay, provenance, and local naming rows around the same regular readback.

Since neither route contains a detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient completeness coordinate, the handoff is exactly the displayed finite L10 lattice. □

The public $\mathsf{Cauchy}^{\uparrow}$ route consumed by completion and real-readback arguments is routed through the same finite lattice: $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ rational

readback from chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the net-facing completion surface of chapter 3794, and the filter-facing completion surface of chapter 1569. The hub exports this dependency route only as finite BHist, componentwise hsame, displayed Cont, Pkg, and local NameCert data; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Theorem 2721.17 (Cauchy RegSeqRat Real dependency top link)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *consumer must expose the regular-source row of chapter 301, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before it can read a completion endpoint. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Read the top-level $\mathsf{Cauchy}^{\uparrow}$ hub through its displayed children. The hub exports only ordered sibling routes, so a completion-facing consumer first encounters finite observation windows and dyadic tolerance data rather than a hidden completion endpoint.

The regular-source handoff is then forced through the $\mathsf{RegSeqRat}^{\uparrow}$ row named by chapter 301. Only after that handoff is visible can the consumer read the terminal $\mathsf{Real}^{\uparrow}$ seal named by chapter 38.

The structural rows preserve the same path. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row keep the completion read inside the finite packet, with no quotient stream, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 2721.18 (Cauchy forward RegSeqRat and Real route)**.** *Every completion-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301 and then the terminal* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38 before any completion consumer may use it. The route carries only the finite stream-window, dyadic tolerance, regular-rational readback, real-seal,* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *rows exposed by the Cauchy sibling files; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with the carrier of Definition 3413.1. Its public readback is organized by finite stream windows, dyadic tolerances, regular-rational readback, terminal real-name sealing, componentwise transport, replay, provenance, and local naming.

Apply Definition 3688.8 and Definition 3413.4. These routes place every Real-facing or completion-facing Cauchy read behind displayed finite windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row before any terminal seal is consumed.

Finally apply Definition 3413.9 together with Definition 5888.13. The Real seal is reached only after the regular-rational row has been displayed, and the completion-facing consumer receives the same finite route. Since the cited sibling routes provide no coordinate for quotient equality, selected limits, choice, host equality, trichotomy, or ambient completeness, none can enter the forward route. □

Theorem 2721.19 (Cauchy StreamName–RegSeqRat forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that is later consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *must pass through* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* BHist*-indexed observations, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. The route is the displayed L10 handoff through chapter 126 and chapter 301; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.*

Proof. Begin with the displayed RegSeqRat route of Definition 3688.8. It already places every Real- or Completion-facing Cauchy read behind finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the terminal seal can be consumed.

Next use the dependency lattice of the Cauchy hub. The lattice records the sibling order in which the $\mathsf{RegSeqRat}^{\uparrow}$ tail rows and $\mathsf{StreamName}^{\uparrow}$ observation schedules are read, so the consumer cannot bypass the finite windows by appealing to a host Cauchy sequence.

Finally apply the regular-fusion route of Definition 3688.14. Its fused-tail handoff preserves the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the Cauchy hub reaches any later $\mathsf{RegSeqRat}^{\uparrow}$ consumer only through the finite StreamName and regular rational surface. □

The $\mathsf{Cauchy}^{\uparrow}$ hub routes completion-facing consumers through the regular rational source of chapter 301, the complete-metric boundary of chapter 128, and the Cauchy-completion functor surface of chapter 423. Thus a completion read remains a displayed finite dependency route before any regular-name, metric-completion, or functorial completion sibling is consumed.

Chapter 2722

A Concrete Naming Certificate for $\mathsf{CauchyTailDominance}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailDominance}^{\uparrow}$ packet records the finite tail-dominance surface for Cauchy names at a displayed tolerance. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the index cofinality discipline of chapter 1594. The packet gives completion-facing Real arguments a named route from tolerance requests to tail windows; it does not select a limit, choose a subsequence, identify quotient real names, or import ambient completeness.

Definition 2722.1 (Cauchy tail-dominance carrier)**.** *A* $\mathsf{CauchyTailDominance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, I, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected Cauchy-name segment.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from that window.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the requested radius, slack budget, and comparison cell used for the tail test.*
4. *I is the* $\mathsf{CofinalSequence}^{\uparrow}$ *index row witnessing that the displayed tail index dominates the requested tolerance window.*
5. *W is the tail-window ledger, pairing the tolerance row D with the cofinal index row I and the inspected stream segment S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, I, W have been displayed.*
7. *H records componentwise* hsame *transport for the stream window, regular readback, tolerance row, cofinal index row, tail ledger, Real seal, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a tolerance request through $D, I, S, R, W, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailDominance}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a chosen limit, quotient stream equality, selected subsequence, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2722.2 (Cauchy tail-dominance NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *carrier*

$$T = (S, R, D, I, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream window* S*, regular rational readback* R*, dyadic tolerance row* D*, cofinal index row* I*, tail-window ledger* W*, terminal Real seal* E*, componentwise transport* H*, displayed continuation replay* C*, package provenance* P*, and local naming row* N*.*

Proof. Project the carrier of Definition 3689.1. The projection exposes exactly $S, R, D, I, W, E, H, C, P, N$.

The tolerance request is first read through D. The cofinal row I then supplies the displayed index dominance needed to open the tail window, while S and R supply only the finite stream segment and regular rational readback attached to that request.

The ledger W binds D, I, S, R before the terminal seal E is visible. Thus the Real-facing consumer receives a tolerance-directed tail route rather than a chosen limit or a quotient equality of completed names.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no row for selected subsequences, countable choice, trichotomy, quotient equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2722.3 (Cauchy tail-dominance RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{CofinalSequence}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The consumer therefore reads tail dominance only as a displayed tolerance request, cofinal index witness, finite stream window, regular rational readback, and terminal Real seal. It receives no chosen Cauchy limit, selected subsequence, quotient stream equality, countable choice, trichotomy, or ambient completeness row.

Proof. Start with an admitted consumer read. By Definition 3689.1, the only analytic rows available before the Real seal are the tolerance row D, cofinal index row I, stream window S, regular readback R, and tail-window ledger W.

Read the requested precision through D. The row I then certifies the displayed dominance of the chosen tail index over that request, and W records that the same request controls the window opened in S.

The regular rational handoff is exactly R. It is downstream of D, I, S, W and upstream of the terminal seal E, so the Real-facing boundary cannot bypass the finite tail route.

The structural rows H, C, P, N replay and name this route without adding analytic coordinates. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read factors through the displayed tail-dominance packet and exports none of the excluded limit, subsequence, quotient, choice, trichotomy, or completeness data. □

Theorem 2722.4 (Cauchy tail-dominance Real seal consumption)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer of an accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *packet must read the tolerance row, cofinal index*

witness, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and tail-window ledger before the Real seal is visible. Thus the completion boundary consumes the finite tail route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CofinalSequence}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ W \longrightarrow \mathsf{Real}^{\uparrow}$$

and cannot introduce selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 3689.1. The terminal Real row E is downstream of the tolerance row D, cofinal index row I, stream window S, regular rational readback R, and tail-window ledger W.

Apply Definition 3689.2. It makes those rows part of the local certificate surface, so a completion-facing consumer cannot read E before it has exposed the same displayed tolerance request, dominance witness, finite window, readback, and ledger.

Now use the handoff of Definition 3689.3. It identifies the route from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ to the terminal Real seal.

The structural rows H, C, P, N preserve and name that route without adding analytic coordinates. Since the accepted carrier has no selected-limit row, no quotient stream equality row, no countable-choice row, no trichotomy row, and no ambient completeness row, the Real seal can only be consumed through the displayed finite tail boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3689.1, Definition 3689.2, Definition 3689.3, and Definition 3689.4 seed the finite tolerance-directed Cauchy tail-dominance route through StreamName, RegSeqRat, DyadicRatCore, CofinalSequence, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, selected subsequences, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-row exactness, cofinal-index dominance, tail-window confinement, RegSeqRat handoff, and RealUp seal non-escape for BEDC.Derived.CauchyTailDominanceUp.

Constructive story: A $\mathsf{CauchyTailDominance}^{\uparrow}$ packet is generated from a StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance row, CofinalSequence index row, tail-window ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{Cauchy}^{\uparrow}$ completion-facing tail arguments consume chapter 3689 as the finite route that opens the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{CofinalSequence}^{\uparrow}$ index witness, and terminal $\mathsf{Real}^{\uparrow}$ seal before any Cauchy boundary is visible. This sibling route records that the Cauchy lattice reaches tail dominance through displayed L10 rows rather than through selected limits, quotient stream equality, countable choice, trichotomy, or ambient completeness.

Theorem 2722.5 (Cauchy metastability-filter sibling route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that uses finite tail control factors through* $\mathsf{CauchyMetastability}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *before reaching* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. Concretely, the consumer first reads the finite-tail stability packet of chapter 2093 and the finite-filterbase carrier of chapter 333; only then may it inspect stream windows, regular rational readback, dyadic tolerance ledgers, terminal real seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Proof. Start with the Cauchy hub dependency surface. Its completion-facing children already route through finite stream windows and regular rational readback before the terminal real seal is visible.

For finite tail control, use chapter 2093. The metastability packet exposes the precision request, control row, finite tail-window answer, dyadic ledger, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local name.

For filter-facing Cauchy data, use chapter 333. The CauchyFilter packet exposes the finite filterbase carrier, directed refinement row, Cauchy modulus row, stream schedules, regular observations, dyadic neighbourhood witnesses, transport, replay, provenance, and local name.

Combining the two sibling packets gives the stated route. No completion-facing $\mathsf{Cauchy}^{\uparrow}$ read can reach the StreamName, RegSeqRat, DyadicRatCore, or RealUp rows by bypassing both finite-tail stability and finite-filterbase data. □

Theorem 2722.6 (Cauchy top-level sibling route)**.** *The top-level* $\mathsf{Cauchy}^{\uparrow}$ *hub routes ordinary Cauchy evidence,* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *sequence-surface data, completion algebra, and tail-dominance readback through its sibling chapters before any Real or Completion consumer reads a terminal seal. The route consumes the finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Bishop regular completion consumer of chapter 1639. It thereby keeps Cauchy, sequence-space, completion-algebra, and tail-dominance surfaces on sibling NameCert routes rather than on a hidden hub coordinate.*

Proof. Read the hub as an ordered sibling route. Its ordinary Cauchy and tail-dominance files supply local Cauchy evidence, while the displayed RegSeqRat and metric-completion dependency files point outward to the Real and Completion lattice.

The finite-window side is supplied by chapter 126 and chapter 291. Those siblings provide observation windows and tolerance rows before a regular rational or real-seal consumer can inspect the Cauchy surface.

The completion-facing side then passes through chapter 301, chapter 38, and chapter 1639. Thus the Cauchy hub exports only a horizontal dependency route among displayed sibling chapters, with no quotient stream, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 2722.7 (Cauchy Real-completion forward route)**.** *Every Real-completion consumer of the top-level* $\mathsf{Cauchy}^{\uparrow}$ *hub first passes through the sibling route displayed in Definition 3689.6: finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the Bishop regular completion handoff. No hub coordinate supplies quotient streams, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness for such a consumer.*

Proof. Apply Definition 3689.6 to read the hub as an ordered dependency route rather than as a terminal completion object. Its first two displayed siblings provide the finite-window and dyadic-tolerance data that a completion-facing consumer must expose before any rational or real readback can be inspected.

The remaining siblings then force the read through RegSeqRat, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the Bishop regular completion handoff. Since the route already accounts for every completion-facing coordinate named by the hub, there is no remaining coordinate from which a quotient stream, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle could be exported. □

Theorem 2722.8 (Cauchy top-level completion lattice route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of the top-level hub factors through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can use any completion*

boundary. The route is read through chapter 126, chapter 291, chapter 301, chapter 38, and the Bishop regular completion surface of chapter 1639; it exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a completion-facing read of the hub. The hub is only an ordered list of child routes, so the consumer first meets the finite StreamName window rows and the DyadicRatCore tolerance rows named by the displayed siblings, not a hidden completion object.

The regular rational readback is then routed through $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Structural replay through hsame, Cont, Pkg, and NameCert preserves the same ordered rows rather than adding a quotient stream or selected endpoint.

Finally the Bishop regular completion surface may consume the already displayed route. Since every completion-facing coordinate has passed through the sibling lattice, no remaining hub coordinate can supply choice, host equality, or an ambient completeness theorem. □

Chapter 2723

A Concrete Naming Certificate for MonotoneCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the MonotoneCauchySubsequence$^{\uparrow}$ packet records the finite surface by which a monotone source sequence exposes a Cauchy subsequence route without exporting a selected limit or quotient stream. It sits after the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real$^{\uparrow}$ seal of chapter 38, the monotone bounded source packet of chapter 3407, and the subsequence-stability handoff of chapter 3515. The packet names the extraction surface available to Real and Completion consumers; it has no coordinate for a quotient stream, selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 2723.1 (MonotoneCauchySubsequence carrier)**.** *A* MonotoneCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, I, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source monotone sequence row, read through the* MonotoneBoundedSequence$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *surfaces before any subsequence request is consumed.*
2. *I is the monotone index row. It records the finite increasing index schedule used to inspect subsequence windows.*
3. *M is the modulus and tail row. It records the finite tail threshold requested by a Cauchy read and the matching source windows opened by I.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *budget row carrying the dyadic tolerance and refinement data for the source and subsequence windows.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the selected windows after the index and dyadic budget rows have been displayed.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after S, I, M, D, R have been read.*
7. *H records componentwise* hsame *transport for the source sequence, index schedule, modulus and tail row, dyadic budget, readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a subsequence Cauchy request through $S, I, M, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* MonotoneCauchySubsequence$^{\uparrow}$*,* MonotoneBoundedSequence$^{\uparrow}$*,* RegularCauchySubseq StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$ *row.*
The carrier exports only the finite subsequence extraction packet. It exports no quotient stream, selected limit, countable choice, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchySubsequenceUp.MonotoneCauchySubsequenceTasteGate single carrier alignment`

Definition 2723.2 (MonotoneCauchySubsequence classifier)**.** *Two accepted* $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ *carriers*

$$Q = (S, I, M, D, R, E, H, C, P, N)$$

and

$$Q' = (S', I', M', D', R', E', H', C', P', N')$$

are classified together exactly when the displayed hsame *rows transport S to S', I to I', M to M', D to D', R to R', and E to E', and the displayed* Cont *rows replay the same route from the monotone source through index schedule, tail modulus, dyadic budget, regular readback, and Real seal. The classifier has no coordinate for host sequence equality, quotient stream equality, a selected infinite subsequence, a selected limit, countable choice, trichotomy, or ambient completeness.*

Theorem 2723.3 (MonotoneCauchySubsequence NameCert obligations)**.** *For every accepted* MonotoneCauchySub *carrier*

$$Q = (S, I, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the source monotone sequence row, monotone index row, modulus and tail row, dyadic budget row, regular rational readback, Real seal, componentwise transport, replay, provenance, and local naming. Every subsequence Cauchy read must display S, I, M, D, R before E is available, and H, C, P, N only preserve that route.*

Proof. Project the carrier of Definition 3690.1. The displayed analytic rows are S, I, M, D, R, E, and the structural rows are H, C, P, N.

The source row S is read through the monotone bounded sequence surface, while I records the increasing index schedule. A Cauchy request can therefore inspect a subsequence window only after the source schedule and the selected index row have both been displayed.

The modulus and budget rows come next. The row M supplies the finite tail threshold, and D supplies the dyadic tolerance and refinement budget for the same source and subsequence windows.

The readback and seal rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the scheduled windows after the index and dyadic budget have been displayed, and the $\mathsf{Real}^{\uparrow}$ row E seals only that finite route. Transport, replay, provenance, and local naming preserve the same coordinates, so the listed rows exhaust the local NameCert obligations. □

Theorem 2723.4 (MonotoneCauchySubsequence RegSeqRat handoff)**.** *Every accepted* MonotoneCauchySubsequen *carrier $Q = (S, I, M, D, R, E, H, C, P, N)$ hands a subsequence Cauchy read to* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the finite route*

$$S, I \longmapsto M, D \longmapsto R \longmapsto E, H, C, P, N.$$

The handoff composes the source monotone bounded sequence route with the stability surface of chapter 3515; it does not export a quotient stream, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and classifier. They force every subsequence read to preserve the source sequence row, monotone index schedule, modulus and tail row, dyadic budget, readback, and seal by componentwise transport and displayed replay.

Apply Definition 3690.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ row R after S, I, M, D, so the regular rational readback cannot be detached from the selected source windows or the dyadic tolerance ledger.

Finally use the sibling stability route of chapter 3515. That route preserves regular Cauchy readback under monotone cofinal subsequence schedules. The present packet supplies the corresponding $S, I, M, D, R, E, H, C, P, N$ rows, and no excluded quotient, selector, choice, trichotomy, or completeness coordinate is available. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ must display the monotone source row, increasing index row, modulus and tail row, dyadic budget, regular readback, Real seal, transport, replay, provenance, and local NameCert row first.

Independence witness: monotoneboundedsequence, regularcauchysubsequencestability, regseqrat, streamname, dyadicratcore, real: $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source monotone sequence, increasing index schedule, modulus and tail row, dyadic budget, readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3690.1, Definition 3690.2, Definition 3690.3, and Definition 3690.4 seed the finite extraction surface for monotone Cauchy subsequence reads over source sequence, index, modulus, dyadic budget, readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, classifier preservation, monotone index exactness, dyadic budget transport, RegSeqRat handoff, Real seal nonescape, provenance, and local naming for BEDC.Derived.MonotoneCauchySubseque

Constructive story: A $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ packet is generated from a monotone source sequence row, increasing index row, modulus and tail row, dyadic budget, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2724

A Concrete Naming Certificate for $\mathsf{CauchySequenceCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchySequenceCompletion}^{\uparrow}$ packet records Cauchy completion data by finite regular-sequence rows rather than by ambient quotient language. It sits after the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the completion handoff of chapter 1506 and chapter 128, the terminal real-name seal of chapter 38, and the separated uniqueness boundary of chapter 1411. The packet gives Real and Completion consumers one displayed route from finite Cauchy sequence observations to a named completion point.

Definition 2724.1 (CauchySequenceCompletion carrier)**.** *A* $\mathsf{CauchySequenceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, S, R, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric or regular-Cauchy source row whose distance observations are the only metric data available to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected indices and the request schedule used by the Cauchy read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on exactly those windows, including the rational approximants and their displayed tail budgets.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row exposing the completion-facing carrier and entry discipline.*
5. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row reached only after S, R, M have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the completed readback, paired with the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness route when two completed names are compared.*
7. *H records componentwise* hsame *transport for the source, window, readback, completion handoff, limit-service, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through $X, S, R, M, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, S, R, M, L, E, H, C, P, N$. It has no coordinate for quotient completion classes, selected representatives, countable choice, host equality of sequences, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCompletionUp.CauchySequenceCompletionTasteGate single carrier alignment`

Theorem 2724.2 (CauchySequenceCompletion NameCert obligations)**.** *For every accepted* CauchySequenceCompl*carrier $K = (X, S, R, M, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the source metric row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff,* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness route, componentwise transport, displayed replay, provenance, and local naming.*

Every completion-facing consumer must read these rows in the order

$$X, S \longmapsto R \longmapsto M, L \longmapsto E \longmapsto H, C, P, N.$$

No obligation is discharged by a quotient-completion equality, a selected limit, countable choice, host sequence equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3691.1. The only source observations are the metric row X, the finite windows S, and the regular rational readback R.

The completion data is downstream of those observations. The row M exposes the $\mathsf{MetricCompletion}^{\uparrow}$ handoff, while L records the $\mathsf{CompleteMetric}^{\uparrow}$ limit service and cannot be read before the displayed windows and readback have been accepted.

The terminal row E is the Real-facing seal. When two completed names are compared, the separated uniqueness boundary is reached through the same displayed rows before any classifier identification is exported.

Finally H, C, P, N transport, replay, source, and name the route. Since the carrier contains no quotient, selected representative, countable-choice schedule, host equality, trichotomy, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2724.3 (CauchySequenceCompletion finite-window route)**.** *Every public read of an accepted* $\mathsf{CauchySequenceCompletion}^{\uparrow}$ *packet that reaches* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the Cauchy sequence. If a uniqueness consumer compares two completed reads, it must then pass through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance route before reading equality of the exposed completion names.*

Thus the route exports completion data only as a finite window-and-readback handoff:

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow}.$$

It exports no quotient completion class, selected representative, unrestricted limit selector, host sequence equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a completion-facing request. By Definition 3691.1, the request must open S and R before it can reach either completion row.

The $\mathsf{MetricCompletion}^{\uparrow}$ handoff M and $\mathsf{CompleteMetric}^{\uparrow}$ row L then carry the displayed Cauchy read into the completion surface. The Real seal E is downstream of the same handoff, so a Real-facing read cannot bypass finite windows or regular rational readback.

For uniqueness reads, apply the separated boundary recorded in the carrier. The comparison passes through the zero-distance route of $\mathsf{SeparatedMetric}^{\uparrow}$ before any completed-name identification is public.

The structural rows H, C, P, N preserve this dependency order under transport and replay. No excluded completion, choice, host-equality, trichotomy, or Real-completeness principle is present as a carrier row, so none can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3691.1, Definition 3691.2, and Definition 3691.3 seed the finite Cauchy sequence completion route over StreamName windows, RegSeqRat readback, MetricCompletion and CompleteMetric handoffs, Real seals, separated uniqueness, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host sequence equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, completion handoff, limit-service exposure, Real seal nonescape, separated uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCompletionUp.

Constructive story: A $\mathsf{CauchySequenceCompletion}^{\uparrow}$ packet is generated from a source metric row, finite StreamName windows, RegSeqRat readback, MetricCompletion handoff, CompleteMetric limit service, Real seal, SeparatedMetric uniqueness route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2725

A Concrete Naming Certificate for $\mathsf{BoundedIntervalProjection}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{BoundedIntervalProjection}^{\uparrow}$ packet records the finite data needed to project a displayed real-name input into a displayed closed bounded interval. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream window surface of chapter 126, the dyadic tolerance core of chapter 291, the endpoint interval packet of chapter 3527, the clamp route of Definition 1810.4, and the located interval surface of chapter 1709. The packet gives bounded-subsequence, interval-trapping, and completion-facing consumers a named projection object; it is not host trichotomy, not quotient equality, not a selected limit, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2725.1 (Bounded interval projection carrier)**.** *A* $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *row whose projected value is being named.*
2. *L and U are the lower and upper endpoint* $\mathsf{Real}^{\uparrow}$ *rows of the closed bounded interval.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order and tolerance ledger used to compare finite reads of X, L, U.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface for the source and endpoint names.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
6. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row tying the endpoint seals, dyadic order ledger, and locatedness data to the interval surface.*
7. *C is the* $\mathsf{RealClamping}^{\uparrow}$ *row naming the constructive clamp of X against the displayed interval.*
8. *H stores componentwise* hsame *transport for the source, endpoints, tolerance ledger, windows, readback, interval, clamp, replay, provenance, and naming rows.*
9. *T stores displayed* Cont *replay of the projection route through $X, L, U, D, W, R, I, C, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedIntervalProjection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, L, U, D, W, R, I, C, H, T, P, N$. It has no coordinate for host trichotomy, quotient equality of real names, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2725.2 (Bounded interval projection NameCert obligations)**.** *For every accepted* BoundedIntervalProjecti *carrier*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *row X, endpoint rows L, U, dyadic endpoint-order and tolerance ledger D, finite stream windows W, regular rational readback R, located interval row I, constructive clamping row C, componentwise transport H, displayed replay T, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3692.1. The projection exposes exactly the source real row, endpoint rows, dyadic tolerance ledger, finite windows, regular readback, located interval row, clamp row, and structural rows.

The interval boundary is read through the $\mathsf{RealBoundedInterval}^{\uparrow}$ and $\mathsf{LocatedRealInterval}^{\uparrow}$ surfaces. Thus endpoint order, locatedness, and finite bracket information are visible before any projection output is named.

The projection value is read through C. By the $\mathsf{RealClamping}^{\uparrow}$ route cited above, the clamp consumes the same source, endpoints, dyadic tolerance ledger, stream windows, regular readback, and located interval row, so it introduces no host comparison oracle or hidden limit selector.

Finally H, T, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for host trichotomy, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.BoundedIntervalProjectionUp.BoundedIntervalProjectionTasteGate single carrier alignment`

Theorem 2725.3 (Bounded interval projection clamping stability)**.** *Let*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N)$$

be an accepted $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *carrier. Any componentwise* hsame *transport of the source row, endpoint rows, dyadic ledger, stream windows, regular readback, and located interval row transports the clamp row C to the same* $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *output surface. Consequently bounded-subsequence and interval-trapping consumers read a stable closed-interval projection through the displayed rows, not through quotient equality, selected limits, host trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier and its finite interval boundary. The rows L, U, D, W, R, I display the endpoint seals, dyadic tolerance comparisons, finite windows, regular rational readback, and located interval data used by the clamp.

Apply the clamp stability route of Definition 1810.4. It states that the clamped value is stable under the displayed finite Cauchy readback and transport rows, so the projection output is determined by the same rows already present in B.

The transport row H and replay row T preserve that route for the local packet. Provenance and naming through P, N then repack the result as a $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ surface.

There is no further coordinate in Definition 3692.1 from which a host comparison, quotient real equality, selected limit, or ambient completeness principle could be read. Thus clamping stability is exactly stability of the displayed finite projection packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedIntervalProjection}^{\uparrow}$ packet is generated from a source RealUp row, two endpoint RealUp rows, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a LocatedRealInterval row, a RealClamping row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3692.1, Definition 3692.2, and Definition 3692.3 seed the finite closed-interval projection route over $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, and $\mathsf{LocatedRealInterval}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host trichotomy, quotient equality of real names, selected limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order evidence, finite-window readback, located interval handoff, clamping stability, transport, replay, provenance, and local naming for BEDC.Derived.BoundedIntervalProjection

Chapter 2726

A Concrete Naming Certificate for $\mathsf{MontelTheorem}^{\uparrow}$

Following Montel's theorem in the normal-family tradition, the $\mathsf{MontelTheorem}^{\uparrow}$ packet records only the BEDC finite-window content of constructive normal-family compactness. It binds compact-window rows, local boundedness and equicontinuity ledgers, uniform Cauchy extraction, and holomorphic readback rows over displayed finite charts. The chapter does not claim classical open-cover compactness, countable choice, quotient real or complex equality, selected subsequences, or full analytic compactness. Its sibling route passes through the finite-modulus Arzela–Ascoli surface of chapter 412, the compact metric and complete metric rows already available in the concrete-instance family, the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, and the holomorphic and complex surfaces when those rows are present.

Definition 2726.1 (Montel theorem carrier)**.** *A* $\mathsf{MontelTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-window row, naming the finite subwindow, metric inspection data, and compact-source certificate used by the family read.*
2. *L is the local boundedness ledger for the displayed family on the same compact window.*
3. *E is the equicontinuity ledger, expressed as finite modulus rows over the compact-window inspection schedule.*
4. *U is the uniform Cauchy extraction row, recording a finite tail window, a subsequence or subnet schedule when already displayed by the source packet, and a dyadic tolerance budget.*
5. *Q is the uniform-limit readback row over the same compact window.*
6. *G is the holomorphic or complex-read row, restricted to the displayed chart and consumed only after K, L, E, U, Q have been exposed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback family for the finite observation windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the local boundedness, equicontinuity, and uniform Cauchy rows.*
9. *Z is the terminal complex or real seal row read by the holomorphic surface when such a row is part of the source family.*

10. *H records componentwise* hsame *transport for all displayed rows.*
11. *C records displayed* Cont *replay from compact-window request through local boundedness, equicontinuity, uniform Cauchy extraction, uniform-limit readback, and holomorphic readback.*
12. *P is the* Pkg *provenance over* $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, *compact-window and complete-metric rows,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *complex or holomorphic surfaces,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their structural transport, replay, provenance, and local naming data.

Lean4 (definition) `BEDC.Derived.MontelTheoremUp.TasteGate.MontelTheoremUp`
Lean4 ✓ `BEDC.Derived.MontelTheoremUp.TasteGate.MontelTheoremTasteGate single carrier alignment`

Theorem 2726.2 (Montel theorem NameCert obligations)**.** *For every accepted* $\mathsf{MontelTheorem}^{\uparrow}$ *carrier* $M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *compact-window admission for K and agreement of every later row with the same displayed window;*
2. *local boundedness admission for L over the compact-window schedule;*
3. *equicontinuity admission for E with dyadic tolerances read from D;*
4. *uniform Cauchy extraction through U, with every late-tail comparison served by R and D;*
5. *uniform-limit readback through Q before any holomorphic or complex read in G is exposed;*
6. *holomorphic readback compatibility, requiring G and Z to consume only the displayed uniform Cauchy and uniform-limit rows;*
7. *transport, replay, provenance, and naming stability through H, C, P, N.*

No obligation introduces open-cover compactness, countable choice, quotient real equality, quotient complex equality, a selected analytic limit, or ambient analytic compactness.

Proof. Project the carrier in Definition 3693.1. The compact-window row K is the only source of domain inspection data, so every later read is scoped to the same finite compact window.

Local boundedness and equicontinuity are the finite normal-family inputs. The rows L and E expose pointwise bound and finite-modulus data over K, while D supplies the dyadic tolerance budget used by both ledgers.

The extraction step is the finite Arzela–Ascoli route. Apply the sibling surface chapter 412 through the displayed rows K, L, E, U, R, D to obtain a uniform Cauchy extraction over the compact window rather than a choice-selected analytic subsequence.

The readback step consumes Q and then G. The uniform-limit row is displayed before the holomorphic or complex-read row, so holomorphic readback can use only the finite uniform Cauchy and tolerance surface already named by the carrier.

Finally, H, C, P, N preserve the same row inventory under transport, continuation, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, countable choice, quotient equality, selected analytic limits, or ambient analytic compactness, the NameCert surface cannot export any such datum. □

Theorem 2726.3 (Montel theorem finite normal-family criterion)**.** *Let $M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$ be an accepted* $\mathsf{MontelTheorem}^{\uparrow}$ *packet. Any consumer claiming a normal-family compactness read from M factors through the finite route*

$$\textit{compact window} \longmapsto \textit{local boundedness and equicontinuity} \longmapsto \textit{uniform Cauchy extraction} \longmapsto \textit{uniform-limit readl}$$

Thus the consumer receives compact-window, boundedness, equicontinuity, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *uniform Cauchy, uniform-limit, and holomorphic rows as finite BEDC data. It receives no classical open-cover compactness theorem, countable-choice selector, quotient real or complex equality, selected analytic limit, or full analytic compactness assertion.*

Proof. Start from the accepted carrier. The only admissible source route begins with K, so every compactness-facing consumer first reads the displayed compact-window packet.

Next consume L and E. These rows give local boundedness and equicontinuity over that window, with the tolerance budget synchronized by D and the regular readbacks synchronized by R.

Now replay C through U and Q. The route supplies uniform Cauchy extraction and uniform-limit readback on the same finite window, matching the ContinuousMap and UniformLimit style chapters without selecting an external analytic limit.

Finally consume G and Z only after the uniform-limit row is present. Componentwise transport through H and provenance through P, N preserve the same route, so no forbidden open-cover, choice, quotient-equality, selected-limit, or ambient-compactness coordinate can enter the public read. □

Closure status for $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3693.1, Definition 3693.2, and Definition 3693.3 seed the finite normal-family route through compact windows, local boundedness, equicontinuity, uniform Cauchy extraction, uniform-limit readback, and holomorphic readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical open-cover compactness, countable choice, quotient real equality, quotient complex equality, selected analytic limits, full analytic compactness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exactness, local-boundedness and equicontinuity ledgers, uniform Cauchy extraction, holomorphic readback compatibility, and provenance exhaustion for BEDC.Derived.MontelTheoremUp.

Constructive story: A $\mathsf{MontelTheorem}^{\uparrow}$ packet is generated from compact-window rows, local boundedness and equicontinuity ledgers, uniform Cauchy extraction, uniform-limit readback, holomorphic or complex readback, RegSeqRat and DyadicRatCore tolerance rows, transport, replay, provenance, and local NameCert rows.

Chapter 2727

A Concrete Naming Certificate for $\mathsf{AntipodalTriggerLedger}^{\uparrow}$

The $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ packet realizes the antipodal AGY trigger of Definition 9462.1506 as finite NameCert data. It records the median-closure source, the R-halfspace with its CUY anchor, the AGY antipode row, the trigger ledger, transport, replay, provenance, and local naming surface. The packet is adjacent to the six-boundary and trigger-algebra family of chapter 3470, chapter 3478, chapter 3651, and chapter 3681, but it is not bijective to any of them because it binds the one-pass antipodal median trigger as its own carrier row.

Definition 2727.1 (Antipodal trigger ledger carrier)**.** *An* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, Y, G, T, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row for $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$, including the saturated R halfspace and the* CUY *anchor row.*
2. *Y is the third-position Y halfspace shell ledger, displaying the shell decomposition H_0, H_1, H_2, H_3, H_4.*
3. *G is the antipodal* AGY *row, displaying either* AGU *or* AGC *as the single trigger.*
4. *T is the trigger ledger that pairs the* CUY *anchor, the* AGY *antipode, and the already present R-halfspace witness t_b for each target Y-side codon b.*
5. *M is the median-closure replay row, recording that the coordinatewise median of those three displayed witnesses reaches b and hence fills the whole codon cube in one pass.*
6. *H records componentwise* hsame *transport for the source, shell, antipode, trigger, median-closure, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source presentation through the antipode trigger ledger into the median-closure row.*
8. *P is the* Pkg *provenance over* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^{\uparrow}}$ *row.*

The classifier admits only packets carrying these finite rows and future implementations must provide the corresponding Nontrivial *and* FieldFaithful *instances before a Lean-side field interpretation can be claimed.*

Lean4 (definition) `BEDC.Derived.AntipodalTriggerLedgerUp.AntipodalTriggerLedgerUp`

Falsifiable prediction: Removing the displayed AGY antipode row prevents an accepted $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ packet from claiming one-pass full-cube median closure.

Independence witness: `six_boundary_tile_coverage`, `window_coverage_trigger_algebra`, `anchor_subcube_trigger`, and `radial_wobble_spectrum`: none is bijective to $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ because the packet jointly stores the $M^{\star}$ source, the CUY anchor, the AGY antipode, the one-pass target witness ledger, median replay, transport, continuation, provenance, and local naming rows.

Theorem 2727.2 (Antipodal trigger ledger NameCert obligations)**.** *For every accepted* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *carrier* $A = (S, Y, G, T, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *source admission for* $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$ *and its median-closed halfspace anchor;*
2. *shell admission for the displayed* Y*-halfspace ledger and its antipodal* $H_4 = \mathrm{AGY}$ *row;*
3. *trigger exactness, requiring the same* CUY *anchor,* AGY *antipode, and* R*-halfspace witness row to feed the median calculation;*
4. *one-pass median replay, requiring the row* M *to expose every* Y*-side target before the full-cube claim is exported;*
5. *transport, replay, provenance, and naming stability through* H, C, P, N*;*
6. *field-read discipline, requiring future* Nontrivial *and* FieldFaithful *instances before any field-facing interpretation of the trigger ledger is exposed.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier in Definition 3694.1. The source row S supplies the R halfspace and the CUY anchor, while Y supplies the finite shell ledger.

The trigger row G is indispensable. Without a displayed AGY antipode, the packet contains no finite row pairing the anchor with the antipodal point required by Definition 9462.1506.

The ledger T then names the witness triple for each target Y-side codon b: the CUY anchor, the AGY antipode, and the already present R-halfspace point t_b. The replay row M records the coordinatewise median calculation and exposes the one-pass full-cube result.

Finally H, C, P, N preserve this route under transport, continuation, provenance, and local naming. The field-read clause is only a discipline on later interpretation: until Nontrivial and FieldFaithful instances are displayed, the packet exports a finite median-trigger ledger rather than a field theorem. □

Theorem 2727.3 (Antipodal trigger ledger median nonescape)**.** *Every accepted* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *full-cube read factors through the finite route*

$$M^{\star} \longmapsto \mathrm{CUY}\ \textit{anchor} \longmapsto \mathrm{AGY}\ \textit{antipode} \longmapsto \textit{target witness ledger} \longmapsto \textit{one-pass median replay.}$$

Thus the one-pass median-closure claim is available only when $S, Y, G, T, M, H, C, P, N$ *are all displayed. A packet missing the* AGY *row, the target witness ledger, or the median replay row cannot export full-cube closure.*

Proof. Begin with an accepted carrier. The public full-cube read must first expose S, so the median source is exactly $M^\star$ with the CUY anchor and the already present R halfspace.

Next the read consumes Y and G. These rows place the AGY point in the antipodal shell H_4 and identify it as the single trigger rather than as an undifferentiated radial-shell element.

Now apply the trigger ledger T and median replay M. For each target Y-side codon, the route supplies the anchor, antipode, and matching R-halfspace witness, so the coordinatewise median calculation remains inside displayed finite data.

The structural rows H, C, P, N do not add another closure principle; they preserve the same finite route. Hence a consumer that omits the antipode, witness ledger, or median replay row cannot claim one-pass full-cube median closure from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 3694.1, Definition 3694.2, and Definition 3694.3 seed the finite median-trigger ledger from the $M^\star$ source through the AGY antipode into one-pass full-cube median replay.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, field-facing interpretation without Nontrivial and FieldFaithful instances, radial optimality, spectral optimality, or a public standard bridge.

Upgrade path: Obligation closure requires checked source admission, shell admission, antipode exactness, target-witness ledger replay, median nonescape, field-read discipline, and provenance exhaustion for BEDC.Derived.AntipodalTriggerLedgerUp.

Constructive story: An $\mathsf{AntipodalTriggerLedger}^\uparrow$ packet is generated from the $M^\star$ source row, Y-halfspace shell ledger, AGY antipode row, target witness ledger, one-pass median replay, transport, continuation, provenance, and local NameCert rows.

Chapter 2728

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ packet records the displayed criterion by which a Cauchy source, an effective precision schedule, and a finite tail-window readback become one completion-facing BEDC object. It sits after the criterion surface of chapter 392, the modulus schedule of chapter 292, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the effective modulus packet of chapter 3518, and the terminal real-name seal of chapter 38. The packet supplies a finite L10 criterion/modulus handoff; it does not construct quotient streams, select a host limit, import countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2728.1 (Effective Cauchy criterion carrier)**.** *An* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, M, S, T, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *source row naming the finite Cauchy request and its accepted comparison surface.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, refined through the* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *packet to a displayed precision-to-tail request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite source window opened by the criterion request.*
4. *T is the finite tail-window row selected by the effective modulus schedule for the same request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested radius, tail comparison, and endpoint budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after Q, M, S, T, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only to consumers that have already read the criterion, schedule, windows, tolerance ledger, and regular-rational readback.*
8. *H stores componentwise* hsame *transport for the criterion, modulus, windows, tolerance ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from an effective Cauchy request through $Q, M, S, T, D, R, L, H, P, N$.*

10. *P is the* Pkg *provenance over* EffectiveCauchyCriterion$^\uparrow$, CauchyCriterion$^\uparrow$, CauchyModulus$^\uparrow$, EffectiveCauchyModulus$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{EffectiveCauchyCriterion}^\uparrow}$ *row.*

The classifier compares accepted packets only through $Q, M, S, T, D, R, L, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host sequence equality, countable choice, trichotomy, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchyCriterionUp.TasteGate.EffectiveCauchyCriterionTasteGate single carrier alignment`

Theorem 2728.2 (Effective Cauchy criterion NameCert obligations)**.** *For every accepted* EffectiveCauchyCriterion$^\uparrow$ *carrier*

$$E = (Q, M, S, T, D, R, L, H, C, P, N),$$

the local NameCert$_{\text{EffectiveCauchyCriterion}^\uparrow}$ *surface consists exactly of carrier admission for the criterion row Q, effective modulus schedule M, source window S, tail window T, dyadic tolerance ledger D,* RegSeqRat$^\uparrow$ *readback R,* Real$^\uparrow$ *seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3695.1. The analytic rows are exactly the criterion request Q, the effective modulus schedule M, the source and tail windows S, T, the dyadic tolerance ledger D, the regular-rational readback R, and the terminal real seal L.

The schedule obligation is read through M and T. The modulus row turns a displayed precision request into a finite tail threshold, and the tail-window row records the finite observations permitted at that threshold.

The readback obligation is downstream of Q, M, S, T, D. The regular-rational row R is therefore a finite readback from the same criterion and tolerance ledger, not a quotient stream, selected limit, or ambient completeness argument.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded objects, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2728.3 (Effective Cauchy criterion RegSeqRat handoff)**.** *Every* RegSeqRat$^\uparrow$ *or* Real$^\uparrow$ *consumer of an accepted* EffectiveCauchyCriterion$^\uparrow$ *packet factors through*

$$Q, M \longmapsto S, T, D \longmapsto R \longmapsto L \longmapsto H, C, P, N.$$

Consequently the consumer receives the displayed criterion request, effective modulus schedule, finite source and tail windows, dyadic tolerance ledger, regular-rational readback, terminal real seal, and local structural rows; it receives no quotient-stream equality, selected host limit, countable choice, host sequence equality, trichotomy, or ambient Real$^\uparrow$ *completeness.*

Proof. Begin with the criterion and modulus rows. The row Q states the Cauchy comparison request, while M supplies the effective precision-to-tail schedule consumed by that request.

Next open the finite windows and tolerance ledger. The source window S and tail window T provide the only observations exposed to the request, and D records the dyadic budget against which those observations are compared.

Only after those rows are displayed does the packet expose R and then L. The row R is the RegSeqRat$^\uparrow$ readback produced from the finite windows and tolerance ledger, and L is the terminal Real$^\uparrow$ seal available to completion-facing consumers.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Thus every consumer route factors through the displayed finite handoff and cannot bypass it through any excluded host object. □

Closure status for NameCert$_{EffectiveCauchyCriterionUp}$.

Constructive story: An EffectiveCauchyCriterion$^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus effective schedule, StreamName source and tail windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{EffectiveCauchyCriterionUp}$, seed)

Scope closed: Definition 3695.1, Definition 3695.2, and Definition 3695.3 seed the finite effective Cauchy criterion handoff for RegSeqRat and Real-completion consumers.

Formal verification: FormalStatus(NameCert$_{EffectiveCauchyCriterionUp}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-stream equality, selected limits, host sequence equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective schedule exactness, StreamName source and tail window admission, DyadicRatCore tolerance readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.EffectiveCauchyCriterionUp.

Chapter 2729

A Concrete Naming Certificate for $\mathsf{UniformlyCauchySequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the packet records a uniformly Cauchy sequence as displayed finite tail schedules rather than host convergence. It sits beside the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the displayed modulus interface of chapter 1438, and the terminal seal of chapter 38. The packet supplies an L10 completion-facing finite schedule object; it is not a quotient stream, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2729.1 (Uniformly Cauchy sequence carrier)**.** *A* $\mathsf{UniformlyCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row, naming the displayed stream whose finite observations are admitted by the packet.*
2. *T is the uniform tail-threshold schedule. For each requested dyadic tolerance it records the single tail index after which all later source windows are compared.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source at the threshold and at later comparison windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the precision budget and dyadic comparison rows consumed by the uniform tail schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same source, window, tolerance, and tail-threshold data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, T, W, D, R have been displayed.*
7. *H stores componentwise* hsame *transport for the source, threshold schedule, window row, tolerance row, readback row, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a uniform Cauchy request through $S, T, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, W, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, hidden cofinal subsequences, countable choice, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformlyCauchySequenceUp.UniformlyCauchySequenceTasteGate single carrier alignment`

Theorem 2729.2 (Uniformly Cauchy sequence NameCert obligations)**.** *For every accepted* UniformlyCauchySequen *carrier*

$$K = (S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the sequence source S, uniform tail-threshold schedule T,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3696.1. The analytic rows are the source S, threshold schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback R, and Real seal E.

The uniformity obligation is read only from T, W, D. For a requested tolerance, T gives one displayed threshold and W supplies the inspected windows beyond it, while D records the dyadic comparison budget used by those windows.

The regular readback is downstream of the same schedule. Thus R is a $\mathsf{RegSeqRat}^{\uparrow}$ row obtained through S, T, W, D, not a host sequence, quotient stream, selected limit, hidden cofinal family, or completeness argument.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for countable choice, host equality, quotient streams, selected limits, hidden cofinal subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 2729.3 (Uniformly Cauchy sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{UniformlyCauchySequence}^{\uparrow}$ *packet factors through*

$$S \longmapsto T \longmapsto W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently completion and filter consumers may use the displayed source, uniform tail threshold, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local structural rows, but they receive no quotient stream, selected limit, hidden cofinal tail, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the displayed source S. The source is admissible for the packet only as a sequence row whose uniform Cauchy information is supplied by the threshold schedule T.

The finite observation then passes through W and D. The row W exposes the inspected stream windows at and beyond the chosen tail threshold, while D supplies the dyadic tolerance and comparison budget for the same request.

Only after those rows have been displayed does the packet expose R and then E. The row R is the regular rational readback, and E is the terminal Real seal served by that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Since none of the excluded objects occurs in the carrier, no completion-facing or filter-facing consumer can bypass the finite uniform schedule. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyCauchySequence}^{\uparrow}$ packet is generated from a sequence source row, a uniform tail-threshold schedule, a StreamName finite-window row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3696.1, Definition 3696.2, and Definition 3696.3 seed the finite uniform Cauchy schedule object and its regular-rational handoff for completion and filter consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, hidden cofinal tails, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-threshold exactness, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.UniformlyCauchySequenceUp.

Chapter 2730

A Concrete Naming Certificate for DyadicMeshScaleNormalization$^{\uparrow}$

Following Bishop and Bridges constructive analysis and the interval-normalization practice of Moore–Kearfott–Cloud interval computation, the DyadicMeshScaleNormalization$^{\uparrow}$ packet records the common-scale conversion needed before finite nets, dyadic approximations, and real-name seal windows compare mesh cells. It sits after the finite mesh carrier of chapter 358, the dyadic arithmetic core of chapter 291, the precision-scale ledger of chapter 1768, and the real embedding boundary of chapter 1942. The packet exposes a displayed common scale and its readback route; it is not host real equality, quotient stream equality, countable choice, or an appeal to ambient completeness.

Definition 2730.1 (Dyadic mesh scale-normalization carrier)**.** *A* DyadicMeshScaleNormalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, A, B, S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* DyadicMesh$^{\uparrow}$ *mesh row whose cells are to be compared or consumed by a finite-net route.*
2. *A and B are the source mesh-scale rows exposed before normalization.*
3. *S is the common-scale row, recording one dyadic scale that refines both A and B for the requested comparison window.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *arithmetic ledger for rescaling endpoints, widths, and cell representatives from A and B into S.*
5. *T is the* DyadicToleranceScale$^{\uparrow}$ *precision row that binds the normalization scale to the requested tolerance.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window row opened only after S, D, T have been displayed.*
7. *R is the* RegSeqRat$^{\uparrow}$ *handoff row obtained from the normalized dyadic windows.*
8. *E is the* RealDyadicEmbedding$^{\uparrow}$ *seal row reached after the normalized dyadic readback.*
9. *H records componentwise* hsame *transport for the mesh row, source scales, common scale, arithmetic ledger, tolerance row, window row, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a mesh comparison request through* $G, A, B, S, D, T, W, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDyadicEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, A, B, S, D, T, W, R, E, H, C, P, N$. *It has no coordinate for host real equality, quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicMeshScaleNormalizationUp.DyadicMeshScaleNormalizationTasteGate`
`single carrier alignment`

Theorem 2730.2 (Dyadic mesh scale-normalization NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ *carrier*

$$M = (G, A, B, S, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic mesh row G, source scale rows A, B, common-scale row S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *rescaling ledger D,* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *precision row T,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *seal row E, componentwise transport H, displayed continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3697.1. The analytic rows are exactly the mesh row G, the two source scales A, B, the common scale S, the dyadic arithmetic ledger D, the tolerance-scale row T, the finite-window row W, the regular rational readback R, and the real-dyadic seal E.

The normalization obligation is read only from S, D, T. The row S names the common dyadic scale, D records the arithmetic that rewrites both source scales into that scale, and T binds the scale choice to the requested tolerance.

The stream and real-facing rows are downstream of the same normalization. Thus W opens only normalized finite windows, R reads the corresponding regular rational data, and E seals the route through $\mathsf{RealDyadicEmbedding}^{\uparrow}$ rather than through host equality of real numbers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2730.3 (Dyadic mesh common-scale exhaustion)**.** *Every finite-net, dyadic-approximation, or* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{DyadicMesh}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{DyadicToleranceScale}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealDyadicEm}$$

Consequently such a consumer may read the original mesh, the two source scales, the common scale, the dyadic rescaling ledger, the tolerance row, normalized stream windows, regular rational readback, and real-dyadic seal, but it receives no host real equality, quotient stream, selected representative, hidden cofinal refinement, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an admitted consumer read. By Definition 3697.1, the only mesh-facing data before readback are G, A, B, S, D, T, W.

The common scale is forced through S. The rows A and B expose the two source scales, while D supplies the dyadic rescaling ledger that rewrites both into S under the tolerance request recorded by T.

Only after that scale has been displayed may the packet expose W, R, and E. The finite windows are therefore normalized before regular-rational readback, and the real-facing seal is the $\mathsf{RealDyadicEmbedding}^{\uparrow}$ row downstream of that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Hence every exported finite-net, approximation, or real-dyadic consumer factors through the displayed common-scale route and cannot bypass it by using any excluded host, quotient, choice, or completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ packet is generated from a DyadicMesh row, two source scale rows, one displayed common scale, a DyadicRatCore rescaling ledger, a DyadicToleranceScale precision row, StreamName finite windows, a RegSeqRat readback row, a RealDyadicEmbedding seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3697.1, Definition 3697.2, and Definition 3697.3 seed the common-scale normalization packet used by finite nets, dyadic approximation, and real-dyadic seal windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-scale exactness, dyadic rescaling confinement, tolerance-scale binding, normalized StreamName coverage, RegSeqRat handoff, RealDyadicEmbedding seal non-escape, transport stability, continuation replay, and provenance confinement for BEDC.Derived.DyadicMeshScaleNormalizationUp.

Chapter 2731

A Concrete Naming Certificate for $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$

Following the Heine–Cantor and Bishop finite-net route, the $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ packet records the compact-domain uniform modulus data that completion-extension consumers need before a completed map can be read. It refines the compact-uniform-continuity vision of chapter 9237 into concrete rows over the compact metric packet of chapter 129, the continuous-map surface of chapter 130, the uniform modulus consumer of chapter 369, the Cauchy lift route of chapter 2037, and the regular completion readback rows of chapter 301, chapter 126, chapter 291, and chapter 38. The packet names the finite modulus consumer; it does not choose a pointwise selector, quotient a completion, or import ambient compactness or completeness.

Definition 2731.1 (Compact completion extension modulus carrier)**.** *A* $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, G, B, M, L, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the compact metric finite-net source row, carrying the finite net bundle, located radius rows, and coverage witnesses used by the uniform-modulus fold.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus and graph row, read at bundled centers of the compact net.*
3. *B is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, recording the finite rational radius comparison that produces a tolerance-only precision history.*
4. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row opened by Q, G, B.*
5. *L is the* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *route row, exposing the modulus handoff before any completion-extension consumer reads lifted map data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular Cauchy approximants opened by the completion consumer.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to compare the compact-net source data with the completion readback.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying dyadic radii, error budgets, and transported comparison rows.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, visible only after Q, G, B, M, L, R, S, D have been displayed.*
10. *H records componentwise* hsame *transport for the compact source, graph, lower-bound fold, uniform modulus, lift route, readback, window, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of the route*

$$Q, G, B \longmapsto M, L \longmapsto R, S, D \longmapsto E.$$

12. *P is the* Pkg *provenance over* $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, G, B, M, L, R, S, D, E, H, C, P, N$. It has no coordinate for a global point selector, quotient completion equality, selected completed representatives, host function equality, open-cover compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2731.2 (Compact modulus preservation for completion-extension consumers)**.** *Let*

$$K = (Q, G, B, M, L, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ *carrier. Any completion-extension consumer that reads the compact-domain map must preserve the uniform modulus row M produced from the finite-net source Q, pointwise graph row G, and rational lower-bound fold B before it can inspect the* $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.CompactCompletionExtensionModulusUp.TasteGate.CompactCompletionExtensionModulus modulus preservation route`

Proof. Project the carrier. The compact finite-net source row Q, continuous graph row G, and rational lower-bound fold row B are displayed before the uniform modulus row M.

The completion-extension consumer then enters through L. Since L is downstream of M, the completion-facing map is read as data carrying the compactly produced uniform modulus rather than as a host function on completed representatives.

The readback rows R, S, D, E preserve this dependency order. Regular approximants, finite stream windows, dyadic tolerance budgets, and the terminal Real seal are all opened after the modulus handoff, so none can replace it with quotient completion equality or a selected limit.

The structural rows H, C, P, N only transport, replay, source, and name the same route. Therefore point selectors, host equality, open-cover compactness, choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the accepted consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3698.1 and Definition 3698.2 seed the compact finite-net uniform-modulus route consumed by completion-extension packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close point selector choice, quotient completion equality, selected completed representatives, host function equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net visibility, continuous-map pointwise modulus readback, rational lower-bound fold stability, uniform-modulus preservation, completion-lift routing, regular readback, finite-window stability, dyadic tolerance stability, Real seal non-escape, and ledger exhaustion for BEDC.Derived.CompactCompletionExtensionModulusUp.

Constructive story: A CompactCompletionExtensionModulus$^{\uparrow}$ packet is generated from CompactMetric finite-net source rows, a ContinuousMap pointwise graph row, a RatUp lower-bound fold, a UniformModulus consumer row, a CauchyCompletionLift route, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2732

A Concrete Naming Certificate for UniformlyContinuousCompletionExtension$^{\uparrow}$

Following Bishop and Bridges on uniformly continuous maps between completions, the UniformlyContinuousComplet packet records the finite NameCert surface by which a uniformly continuous source map extends across displayed BEDC completion data. It sits after the Cauchy lift route of chapter 2037, the uniform completion consumer of chapter 1892, the regular readback of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38. The packet names the extension route through finite rows; it does not introduce quotient completion classes, selected representatives, host function extensionality, choice, or ambient Real$^{\uparrow}$ completeness.

Definition 2732.1 (Uniformly continuous completion extension carrier)**.** *A* UniformlyContinuousCompletionExtensi *carrier is a finite* BHist *packet*

$$X = (D, T, M, L, U, R, S, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source row, recording the displayed source observations, density witnesses, Cauchy windows, and tolerance requests made before any completed target is inspected.*
2. *T is the target completion-consumer row, exposing the target-facing completion object only through an accepted consumer interface.*
3. *M is the uniform modulus and graph row for the source map, carrying finite graph observations and the tolerance-to-precision ledger consumed by the extension.*
4. *L is the* CauchyCompletionLift$^{\uparrow}$ *handoff row, requiring the source map and its modulus to be read before a lifted completion-facing map is named.*
5. *U is the* UniformCompletionFunctor$^{\uparrow}$ *route row, binding the same source, target, graph, and modulus data to the uniform completion surface.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row, recording the regular rational approximants opened by the completion consumer.*
7. *S is the* StreamName$^{\uparrow}$ *finite-window row, recording the finite stream observations used to compare source Cauchy data with target readback data.*

8. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic radii and comparison budgets through the modulus handoff.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, visible only after the dense source, target consumer, modulus, lift, uniform completion, regular readback, stream-window, and dyadic tolerance rows have been displayed.*
10. *H records componentwise* hsame *transport for the dense source, target consumer, modulus graph, lift, uniform completion route, readback, stream windows, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of the route*

$$D, T, M \longmapsto L, U \longmapsto R, S, Y \longmapsto E.$$

12. *P is the* Pkg *provenance over* $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$, $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, M, L, U, R, S, Y, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host function extensionality, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2732.2 (Uniformly continuous completion extension modulus handoff)**.** *Let*

$$X = (D, T, M, L, U, R, S, Y, E, H, C, P, N)$$

be an accepted $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$ *carrier. Every extension consumer of the completed map factors through the uniform modulus and graph row M, the* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *handoff row L, the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *route row U, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Y, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E before the structural rows H, C, P, N name the result.*

Lean4 ✓ `BEDC.Derived.UniformlyContinuousCompletionExtensionUp.TasteGate.UniformlyContinuousCompletionEx`
`modulus handoff route`

Proof. Carrier projection gives the dense source row D, the target completion-consumer row T, and the modulus-bearing graph row M. The carrier orders these rows before any lift, uniform completion, readback, finite-window, dyadic, or real-seal row.

The completion-facing handoff is then read through L and U. The row L records the same uniform-modulus handoff required by the Cauchy completion lift route, while U records the uniform completion consumer surface; neither row supplies a selected completed representative.

Regular readback is downstream of the handoff. The rows R, S, Y expose regular rational approximants, finite stream windows, and dyadic tolerance budgets before E may seal a real-facing read.

Finally H, C, P, N transport, replay, source, and name only the displayed route. Since the packet has no coordinate for quotient completion, selected representatives, host function extensionality, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no extension consumer can factor through those alternate surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3699.1 and Definition 3699.2 seed the finite modulus-handoff route for uniformly continuous maps extending across BEDC completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host function extensionality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source visibility, uniform-modulus handoff, completion-lift routing, uniform-completion routing, regular readback, finite-window stability, dyadic tolerance stability, Real seal non-escape, and ledger exhaustion for BEDC.Derived.UniformlyContinuousCompletionExtensionUp.

Constructive story: A $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$ packet is generated from dense source rows, a target completion-consumer row, a uniform modulus and graph row, CauchyCompletionLift and UniformCompletionFunctor rows, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2733

A Concrete Naming Certificate for BishopMonotoneRealConvergence$^{\uparrow}$

Following Bishop and Bridges' constructive monotone convergence argument, the BishopMonotoneRealConvergence$^{\uparrow}$ packet records the finite route from a bounded monotone real sequence to a located-supremum handoff and a terminal Real$^{\uparrow}$ seal. It sits beside the located monotone convergence route of chapter 3373, the Bishop monotone limit packet of chapter 1914, the monotone sequence convergence surface of chapter 1435, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the bounded monotone case only; it does not assert arbitrary Real completeness, trichotomy, selected suprema outside the displayed located row, countable choice, or quotient-stream equality.

Definition 2733.1 (BishopMonotoneRealConvergence carrier)**.** *A* BishopMonotoneRealConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, B, O, L, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded monotone* Real$^{\uparrow}$ *sequence row, recording the source sequence, monotonicity direction, and the displayed upper or lower bound.*
2. *B is the boundedness ledger that exposes the finite bound witnesses used by the monotone route.*
3. *O is the order-comparison ledger that records the finite monotonicity reads consumed before any limit row is available.*
4. *L is the* LocatedSupremum$^{\uparrow}$ *handoff row attached to the bounded monotone sequence data.*
5. *W is the* StreamName$^{\uparrow}$ *window family through which source entries, bound witnesses, and located-supremum approximants are inspected.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the finite windows in W.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row that records the requested precision and the dyadic comparison budget.*
8. *E is the* Real$^{\uparrow}$ *seal row reached after L, W, Q, D have been displayed.*
9. *H records componentwise* hsame *transport for the sequence, bound, order, located-supremum, window, readback, tolerance, and real-seal rows.*

10. *C records displayed* Cont *replay from bounded monotone data through the located-supremum handoff and the regular rational readback to the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$, $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$, $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$, $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, O, L, W, Q, D, E, H, C, P, N$. It has no coordinate for arbitrary Cauchy completeness, open-cover compactness, trichotomy of real order, quotient stream equality, selected host suprema, or countable choice.

Lean4 ✓ `BEDC.Derived.BishopMonotoneRealConvergenceUp.BishopMonotoneRealConvergenceTasteGate`
`single carrier alignment`

Theorem 2733.2 (BishopMonotoneRealConvergence NameCert obligations)**.** *For every accepted* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ *carrier $M = (S, B, O, L, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopMonotone}}$ *surface consists exactly of bounded monotone sequence exposure, bound-ledger exposure, monotonicity-read exposure, located-supremum handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

A consumer may read the convergence row only after S, B, O, L, W, Q, D have been displayed. The exported real seal is therefore the finite bounded-monotone route through located supremum and regular rational readback, not an ambient completeness theorem for arbitrary $\mathsf{Real}^{\uparrow}$ *data.*

Proof. Project the carrier of Definition 3700.1. The rows S, B, O expose the bounded monotone source data before any limit-facing row is available.

The row L supplies the located-supremum handoff. The finite windows W, regular readback Q, and dyadic tolerance ledger D then turn the located handoff into the displayed sequence of rational observations consumed by the terminal seal.

Finally E, H, C, P, N preserve the same route. Since the carrier has no coordinate for arbitrary Cauchy completeness, order trichotomy, quotient streams, selected host suprema, or countable choice, the listed rows exhaust the local NameCert surface. □

Theorem 2733.3 (BishopMonotoneRealConvergence located supremum route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *limit read exported by an accepted* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ *packet factors through the displayed located-supremum row L and the finite readback route*

$$S, B, O \longmapsto L \longmapsto W, Q, D \longmapsto E, H, C, P, N.$$

Thus the bounded monotone convergence claim is a finite NameCert route from monotone source data to a located supremum and then to a $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Read the route rows in order. The source sequence S, boundedness row B, and order ledger O provide the only monotone data available to the packet.

The located-supremum handoff L is therefore the unique route by which a limit candidate enters the certificate. The window, readback, and tolerance rows W, Q, D inspect that handoff at finite precision before the real seal E can be consumed.

Transport, replay, provenance, and local naming are supplied by H, C, P, N over the same displayed rows. Hence an exported limit read cannot bypass L or import an ambient completeness principle outside the finite packet. □

Falsifiable prediction: Every accepted bounded monotone convergence read factors through a displayed located-supremum row before any RealUp seal is exported.

Independence witness: located-monotone-convergence, bishop-monotone-real-limit, monotone-sequence-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded monotone source data, bound and order ledgers, a located-supremum handoff, finite windows, regular rational readback, dyadic tolerance accounting, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ packet is generated from bounded monotone RealUp sequence data, a boundedness ledger, monotonicity reads, a LocatedSupremum handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3700.1, Definition 3700.2, and Definition 3700.3 seed the bounded monotone Real convergence NameCert route through located supremum and finite regular-rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary RealUp completeness, order trichotomy, open-cover compactness, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone source exposure, located-supremum handoff exactness, finite-window readback, dyadic tolerance accounting, RealUp seal routing, and TasteGate nontriviality with field faithfulness.

Chapter 2734

A Concrete Naming Certificate for $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy completion, the $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ packet records the finite classifier lock by which a diagonal limit candidate and its uniqueness witness are read through one completion-seal route. It sits after the Cauchy limit seal of chapter 386, the completion-seal classifier of chapter 900, the finite limit-seal envelope of chapter 1276, the regular Cauchy comparison surface of chapter 411, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the uniqueness classifier for the diagonal regular Cauchy limit, not an arbitrary choice of representatives or an ambient Real completeness principle.

Definition 2734.1 (CauchyLimitClassifierLock carrier)**.** *A* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, \mu, L, U, S, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the regular Cauchy input row over* $\mathsf{RegSeqRat}^{\uparrow}$*.*
2. *μ is the finite modulus-budget row used to select diagonal and comparison windows.*
3. *L is the diagonal limit-candidate row.*
4. *U is the uniqueness classifier row comparing any second candidate with L through the displayed RealUp classifier.*
5. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row carrying the common seal interface for L and U.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the input, diagonal candidate, and comparison candidate.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the diagonal and uniqueness comparisons.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, Q, D are displayed.*
10. *H records componentwise* hsame *transport for the input, modulus, diagonal candidate, uniqueness classifier, seal, windows, readback, tolerance, and real seal.*

11. *C records displayed* Cont *replay from regular Cauchy data through the diagonal candidate and classifier lock to the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, RealCompletionSealClassifierUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \mu, L, U, S, W, Q, D, E, H, C, P, N$. It has no coordinate for selected quotient representatives, host real equality, countable choice, arbitrary Cauchy completeness, or a second seal route outside S.

Lean4 ✓ `BEDC.Derived.CauchyLimitClassifierLockUp.CauchyLimitClassifierLockTasteGate single carrier alignment`

Theorem 2734.2 (CauchyLimitClassifierLock NameCert obligations)**.** *For every accepted* CauchyLimitClassifierLoc *carrier $K = (X, \mu, L, U, S, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$ *surface consists exactly of regular Cauchy input exposure, modulus-budget exposure, diagonal candidate exposure, uniqueness-classifier exposure,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The uniqueness row U is accepted only when it is replayed through the same S, W, Q, D, E route as the diagonal candidate L. A consumer that compares candidates through a distinct host equality, quotient representative, or hidden completion seal leaves the packet.

Proof. Project the carrier of Definition 3701.1. The rows X, μ, L, U expose the regular input, modulus budget, diagonal candidate, and uniqueness classifier.

The seal row S is shared by the candidate and the uniqueness comparison. The finite windows W, regular readback Q, and dyadic tolerance ledger D are the only rows through which the comparison may reach the terminal real seal E.

Finally H, C, P, N preserve and replay this same route. Since no host equality, quotient representative, countable choice, arbitrary completeness row, or second seal route is a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2734.3 (CauchyLimitClassifierLock uniqueness route)**.** *Every uniqueness read exported by an accepted* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ *packet factors through the common completion-seal route*

$$X, \mu, L, U \longmapsto S \longmapsto W, Q, D \longmapsto E, H, C, P, N.$$

Thus the diagonal limit candidate and the RealUp uniqueness classifier are locked to the same finite CauchyLimitSeal, RegSeqRat, StreamName, DyadicRatCore, and RealUp NameCert packet.

Proof. Start from an accepted uniqueness read. The carrier exposes only one diagonal candidate row L and one uniqueness classifier row U, both attached to the same regular input X and modulus budget μ.

The common seal row S is then read before any window, readback, tolerance, or real-seal row is exported. Consequently W, Q, D, E inspect the same completion-seal route for the candidate and for the uniqueness comparison.

Transport, replay, provenance, and local naming are supplied by H, C, P, N over this single route. A second classifier route would require an additional carrier coordinate, which Definition 3701.1 excludes. □

Falsifiable prediction: Every accepted Cauchy limit uniqueness read uses the same CauchyLimitSeal route as the displayed diagonal candidate before reaching RealUp.

Independence witness: cauchy-limit-seal, real-completion-seal-classifier, finite-limit-seal-envelope, regular-cauchy-comparison, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regular Cauchy input, modulus budget, diagonal candidate, uniqueness classifier, common seal, finite windows, regular readback, dyadic tolerance ledger, RealUp seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ packet is generated from a regular Cauchy input, modulus budget, diagonal limit candidate, uniqueness classifier, CauchyLimitSeal handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3701.1, Definition 3701.2, and Definition 3701.3 seed the finite classifier-lock route for the Real completeness limit theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary RealUp completeness, host equality of completed reals, selected quotient representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal candidate exposure, uniqueness classifier exactness, common CauchyLimitSeal routing, finite-window readback, dyadic tolerance accounting, RealUp seal routing, and TasteGate nontriviality with field faithfulness.

Chapter 2735

A Concrete Naming Certificate for $\mathsf{DarbouxTheorem}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxTheorem}^{\uparrow}$ packet records the theorem route saying that a displayed continuous real map on a displayed interval has the Darboux property. It sits over the graph-and-modulus surface of chapter 130, the constructive finite bisection route of chapter 1422, the target-value handoff of chapter 3348, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The chapter packages Darboux's theorem as finite NameCert data rather than as host continuity semantics, a selected intermediate point, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2735.1 (Darboux theorem carrier)**.** *A* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the displayed graph, pointwise modulus, and continuity budget for the real-valued map.*
2. *I is the interval endpoint row, exposing the lower and upper endpoint reads and their orientation data.*
3. *A and B are the endpoint value rows obtained from F at the two endpoint rows of I.*
4. *Y is the target-value row displayed between the endpoint value rows.*
5. *S is the endpoint-side exposure row, recording which finite residual comparisons place Y between A and B.*
6. *D is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *schedule that retains branches whose residual comparisons still bracket Y.*
7. *R is the* $\mathsf{DarbouxProperty}^{\uparrow}$ *handoff row reached after the bisection schedule has produced the retained target-value route.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained windows and requested dyadic tolerances.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal available to theorem consumers only after D, R, Q have been read.*

10. *H stores componentwise* hsame *transport for the map, interval, endpoint values, target value, endpoint-side exposure, bisection schedule, property handoff, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a Darboux-theorem request through $F, I, A, B, Y, S, D, R, Q, E$.*
12. *P is the* Pkg *provenance over* $\mathsf{DarbouxTheorem}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted theorem packets only through $F, I, A, B, Y, S, D, R, Q, E, H, C, P, N$. It has no coordinate for host real equality, quotient real equality, a selected intermediate point, real trichotomy, open-cover compactness, semantic continuity outside F, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DarbouxTheoremUp.DarbouxTheoremTasteGate single carrier alignment`

Theorem 2735.2 (Darboux theorem NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier $T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed packet of Definition 3702.1, endpoint-side exposure through I, A, B, Y, S, bisection exactness through D, Darboux-property handoff through R, Real-seal readback through Q, E, componentwise transport through H, displayed continuation replay through C, provenance through P, local naming through N, and non-escape from host real equality, quotient real equality, selected intermediate points, trichotomy, open-cover compactness, semantic continuity outside F, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3702.1. The continuous-map row F and interval row I expose the only theorem source data, while A, B, Y, S expose the finite endpoint-side comparisons for the requested target value.

The bisection schedule is the row D. It is the only branch-retention route in the packet, so every accepted theorem read must pass through the finite $\mathsf{IntermediateValueBisection}^{\uparrow}$ schedule before a target-value handoff is visible.

The property handoff is then the row R, and the Real-facing readback is Q, E. The row Q supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback against the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data named by provenance, and E is the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier has no coordinate for host equality, quotient equality, a selected intermediate point, trichotomy, open-cover compactness, semantic continuity outside F, or ambient Real completeness, none of those objects can appear in the local NameCert surface. □

Theorem 2735.3 (Darboux theorem bisection route)**.** *Every accepted theorem read of a* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier factors through the finite route*

$\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsam

In particular, the read first consumes the displayed map, endpoint rows, target-value row, and endpoint-side exposure; then the finite bisection schedule; then the Darboux-property handoff; then the regular rational and dyadic readback; and only then the $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start from an accepted carrier $T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$. The continuation row C exposes theorem reads only through the map, interval, endpoint value, target value, and endpoint-side rows before any Real-facing seal is available.

The finite bisection route is forced by D. The carrier contains no alternate theorem coordinate between the endpoint-side exposure S and the Darboux-property handoff R, so a consumer cannot bypass $\mathsf{IntermediateValueBisection}^{\uparrow}$ by appealing to host connectedness or semantic continuity.

After the bisection schedule, the theorem read reaches R as the $\mathsf{DarbouxProperty}^{\uparrow}$ handoff. The retained windows are then read through Q using the dyadic tolerance surface named in P, and the terminal read reaches E as the $\mathsf{Real}^{\uparrow}$ seal.

The rows H, C, P, N keep this route stable under transport, replay, provenance, and local naming. Thus every accepted theorem read factors through the displayed finite bisection schedule before the Real seal, with the excluded host and completeness objects remaining outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxTheorem}^{\uparrow}$ packet is generated from a ContinuousMap row, interval endpoint rows, endpoint value rows, a target-value row, endpoint-side exposure, an IntermediateValueBisection schedule, a Darboux-Property handoff, RegSeqRat readback, a DyadicRatCore tolerance surface, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3702.1, Definition 3702.2, and Definition 3702.3 seed Darboux's theorem as a finite bisection route from displayed endpoint-side exposure to DarbouxProperty handoff and Real-seal readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient real equality, selected intermediate points, trichotomy, open-cover compactness, semantic continuity outside the displayed ContinuousMap row, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-side exposure, bisection exactness, DarbouxProperty handoff, Real-seal readback, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.DarbouxTheoremUp.

Chapter 2736

A Concrete Naming Certificate for LocatedCompletionUniversalProperty$^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the LocatedCompletionUniversalProperty$^{\uparrow}$ packet records the extension-and-uniqueness surface for located completion as finite BEDC NameCert data. It sits after the located completion seal of chapter 1805, the generic completion universal-property packet of chapter 397, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names located extension and uniqueness only through displayed source, route, modulus, transport, replay, provenance, and local naming rows; it is not a quotient completion, a selected host limit, countable choice, ambient real completeness, or function extensionality.

Definition 2736.1 (Located completion universal-property carrier)**.** *A* LocatedCompletionUniversalProperty$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, E, Q, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the located completion source row, exposing the source metric, locatedness request surface, dense unit, and completion seal read from chapter 1805.*
2. *U is the universal extension row, read against the target row and the generic completion-universal interface of chapter 397.*
3. *E is the extension uniqueness row, comparing two displayed extensions only over the same located source, dense unit, and target route.*
4. *Q is the request route through* RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$*, carrying finite windows, regular rational readback, and the observations used by located requests.*
5. *M is the uniform-modulus compatibility row, tying source moduli, target moduli, and located request tolerances to the same extension route.*
6. *H records componentwise* hsame *transport for the located source, universal extension, uniqueness row, request route, modulus row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from located source requests through finite stream windows, regular rational readback, uniform-modulus compatibility, extension, and uniqueness.*

8. *P is the* Pkg *provenance over* LocatedCompletionUniversalProperty$^{\uparrow}$, LocatedCompletionSeal$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. *N is the local* NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, E, Q, M, H, C, P, N$. It has no coordinate for quotient completion representatives, selected limits, ambient complete-space theorems, function extensionality, countable choice, or unlocated real equality.

Lean4 ✓ `BEDC.Derived.LocatedCompletionUniversalPropertyUp.LocatedCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Theorem 2736.2 (Located completion universal-property NameCert obligations)**.** *For every accepted* LocatedCompletionUniversalProperty$^{\uparrow}$ *carrier $K = (L, U, E, Q, M, H, C, P, N)$, the local* NameCert$_{\text{LocatedComple}}$ *surface consists exactly of located-source admission through L, universal extension exposure through U, extension uniqueness through E,* RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *request routing through Q, uniform-modulus compatibility through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3703.1. The located completion source row L is visible before any extension row is consumed, and it reads the dense unit and completion seal through chapter 1805.

Read the universal-property rows next. The row U supplies the displayed extension route, while E compares two such routes only along the same located source and dense unit. The generic obligation surface is therefore the one already named by chapter 397, refined by located request data rather than by a new host completion object.

The request and modulus rows keep the route finite. The row Q exposes the StreamName$^{\uparrow}$ windows and RegSeqRat$^{\uparrow}$ readback consumed by located requests, and M ties those requests to the same uniform-modulus compatibility ledger before the Real$^{\uparrow}$ seal is read.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier has no quotient representative, selected host limit, ambient completeness theorem, function-extensionality row, countable-choice row, or unlocated real-equality coordinate, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted LocatedCompletionUniversalProperty$^{\uparrow}$ extension or uniqueness read factors through the displayed located source row, finite request route, uniform-modulus compatibility row, transport, replay, provenance, and local naming before reaching a Real$^{\uparrow}$ consumer.

Independence witness: located-completion-seal, completionuniversalproperty, streamname, regseqrat, real: LocatedCompletionUniversalProperty$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a located completion source, universal extension row, extension uniqueness row, finite request route, uniform-modulus compatibility, transport, replay, provenance, and local NameCert row.

Closure status for NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$ **·**

Constructive story: A LocatedCompletionUniversalProperty$^{\uparrow}$ packet is generated from a located completion source row, universal extension row, extension uniqueness row, RegSeqRat and StreamName request route, uniform-modulus compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$, seed)

Scope closed: Definition 3703.1 and Definition 3703.2 seed the finite located-completion universal-property certificate surface.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCompletionUniversalProperty}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, ambient complete-space theorems, unlocated real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located request routing, extension existence, extension uniqueness, uniform-modulus compatibility, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.LocatedCompletionUniversalPropertyUp.

Chapter 2737

A Concrete Naming Certificate for $\mathsf{BooleanTriggerLattice}^{\uparrow}$

The $\mathsf{BooleanTriggerLattice}^{\uparrow}$ packet realizes the Boolean trigger closure lattice of Definition 9462.1520 and Definition 9462.1532 as finite BEDC NameCert data. It sits over the finite truth-tag discipline of chapter 32, the six-boundary-tile coverage route of chapter 3470, and the trigger-algebra sibling surface of chapter 3478. The packet names Boolean support, rank, meet, join, spectral table, transport, replay, provenance, and local naming rows; it is not a biological code oracle, stochastic evolution model, host powerset lattice, quotient codon semantics, or infinite trigger search.

Definition 2737.1 (BooleanTriggerLattice carrier)**.** *A* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, R, J, K, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite* $\mathsf{Bool}^{\uparrow}$ *tag row for present, absent, covered, uncovered, full-trigger, and non-trigger reads.*
2. *S is the six-boundary support row, read from* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *and carrying the displayed coordinates $\{s_1, f_1, s_2, f_2\}$.*
3. *R is the rank row, naming the finite support union $J(B)$ and the rank $r(B)$ of the source vision.*
4. *J is the join-trigger row, recording median closure of a union as support union.*
5. *K is the meet-trigger row, recording intersection of trigger closures as support intersection.*
6. *O is the observed table row, carrying the finite size, cubical, spectral, threshold, and singleton-trigger reads displayed in the source vision.*
7. *H records componentwise* hsame *transport for Boolean tags, support rows, rank rows, join rows, meet rows, observed table rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each accepted trigger read through B, S, R, J, K, O, H.*
9. *P is the* Pkg *provenance over* $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, WindowCoverageTriggerA *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, R, J, K, O, H, C, P, N$. It has no coordinate for host powersets, biological-code truth, stochastic dynamics, quotient codon equality, hidden spectral computation, or unbounded trigger enumeration.

Lean4 ✓ `BEDC.Derived.BooleanTriggerLatticeUp.BooleanTriggerLatticeTasteGate single carrier alignment`

Theorem 2737.2 (BooleanTriggerLattice NameCert obligations)**.** *For every accepted* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *carrier* $T = (B, S, R, J, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$ *surface consists exactly of Boolean tag admission, six-boundary support exposure, finite rank exposure, join-trigger exposure, meet-trigger exposure, observed table exposure, componentwise transport, displayed replay, provenance, and local naming.*

The future Lean TasteGate plan for this packet requires $\mathsf{Nontrivial\ BooleanTriggerLattice}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ BooleanTriggerLattice}^{\uparrow}$ *so that* $B, S, R, J, K, O, H, C, P, N$ *remain separate displayed fields rather than a collapsed tag.*

Proof. Project the carrier of Definition 3704.1. The rows B, S, R, J, K, O expose the Boolean tags, boundary support, rank, join, meet, and observed table before any consumer can read a trigger-lattice conclusion.

The join and meet rows are finite. The row J reads support union from the source vision, while K reads support intersection. Both rows depend on the same six-boundary support surface and therefore cannot be replaced by an ambient powerset lattice.

The observed table row is also displayed. It records the finite rank-size, cubical, spectral, threshold, and singleton-trigger data already present in the vision file, but it does not add a stochastic model or a hidden spectral oracle.

Finally H, C, P, N preserve the same displayed route. Since no host powerset, biological truth oracle, quotient codon equality, stochastic dynamics, or unbounded trigger enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Falsifiable prediction: An accepted $\mathsf{BooleanTriggerLattice}^{\uparrow}$ read is refuted if its full-trigger, rank, meet, or join decision can be produced without the displayed Boolean tags, six-boundary support row, finite rank row, observed table row, transport, replay, provenance, and local naming.

Independence witness: bool, six-boundary-tile-coverage, window-coverage-trigger-algebra, boolean-trigger source vision: none is bijective with $\mathsf{BooleanTriggerLattice}^{\uparrow}$ because the packet jointly carries Boolean tags, six-boundary support, trigger rank, join rows, meet rows, observed rank-spectral table data, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BooleanTriggerLattice}^{\uparrow}$ packet is generated from BoolUp tag rows, six-boundary support rows, finite rank rows, join-trigger rows, meet-trigger rows, observed rank-spectral table rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3704.1 and Definition 3704.2 seed the finite Boolean trigger-lattice NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, host powerset lattices, quotient codon equality, hidden spectral computation, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Boolean tag exposure, support-rank exposure, meet and join rows, observed table exposure, Nontrivial BooleanTriggerLatticeUp, and FieldFaithful BooleanTriggerLatticeUp for BEDC.Derived.BooleanTriggerLatticeUp.

Chapter 2738

A Concrete Naming Certificate for $\mathsf{BishopCauchyComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyComparison}^{\uparrow}$ packet records the finite comparison route between Cauchy-criterion reads and convergence-modulus reads before a $\mathsf{Real}^{\uparrow}$ or completion consumer can inspect a terminal seal. It sits beside the criterion surface of chapter 392, the bidirectional criterion/modulus surface of chapter 3343, the convergence-modulus boundary of chapter 2074, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the finite comparison only; it does not assert quotient-stream equality, selected limits, countable choice over thresholds, host convergence, or ambient completeness.

Definition 2738.1 (Cauchy criterion comparison chain)**.** *A* $\mathsf{BishopCauchyComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, displaying the finite sequence windows read by both presentations.*
2. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *late-pair request row, recording the criterion-facing tolerance read.*
3. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *schedule row, recording the modulus-facing threshold hand-off.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by the criterion request and the modulus schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which both presentations inspect the same rational windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source windows, criterion request, modulus schedule, tolerance, readback, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from either criterion-facing or modulus-facing entry through the same finite route.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyComparison}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, M, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable-choice threshold selection, detached completion, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyComparisonUp.BishopCauchyComparisonTasteGate single carrier alignment`

Theorem 2738.2 (Cauchy criterion comparison NameCert obligations)**.** *For every accepted* BishopCauchyComparis *carrier $B = (S, Q, M, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$ *surface consists exactly of carrier admission, criterion-to-modulus handoff, modulus-to-criterion handoff, shared tolerance-readback exactness,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance exhaustion, and local naming.*

A criterion-facing consumer may read the modulus schedule M only through the same S, D, R, E, H, C, P, N rows that support the criterion request Q. Conversely, a modulus-facing consumer may read the criterion request Q only through those same rows. Thus the packet supplies a finite comparison chain rather than a host equality between presentations.

Proof. Project the carrier of Definition 3705.1. Carrier admission exposes exactly the tuple $S, Q, M, D, R, E, H, C, P, N$.

For the criterion-to-modulus direction, the late-pair request Q is read together with the source windows S, the shared dyadic tolerance row D, and the regular rational readback row R. The modulus schedule M is visible only as the threshold row attached to that same finite data.

For the modulus-to-criterion direction, the schedule M is read with the same source, tolerance, and readback rows. The criterion request Q is therefore reconstructed as a finite late-pair obligation over the displayed windows, not as an independent convergence predicate.

The real seal E is downstream of both reads. Transport, replay, provenance, and naming are supplied by H, C, P, N over the same rows, so no quotient stream, selected limit, countable-choice selector, host convergence theorem, or ambient completeness row can enter the local certificate. □

Theorem 2738.3 (Cauchy criterion comparison real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyComparison}^{\uparrow}$ *packet reaches the terminal seal only through the finite route*

$$S, Q, M, D, R \longmapsto E \longmapsto H, C, P, N.$$

In particular, neither the criterion presentation nor the modulus presentation can bypass the shared tolerance-readback route or replace it by quotient-stream equality, selected limits, host convergence, countable choice, or ambient completion.

Proof. Use Definition 3705.2. Both handoff directions force the criterion row Q and modulus row M to share the source windows S, tolerance row D, and readback row R.

The terminal row E is listed only after those shared rows. A $\mathsf{Real}^{\uparrow}$ or completion-facing consumer therefore cannot inspect the seal before the finite comparison route has been displayed.

Finally H, C, P, N preserve and name that same route. Since the carrier has no row for quotient streams, selected limits, countable-choice schedules, host convergence, or detached completion, the seal cannot escape through such data. □

Falsifiable prediction: Every accepted criterion/modulus comparison read exposes the shared StreamName, RegSeqRat, and DyadicRatCore rows before any RealUp or completion-facing seal is exported.

Independence witness: cauchycriterion, cauchy-criterion-equivalence, real-modulus-of-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopCauchyComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries criterion request, modulus schedule, shared tolerance-readback, RealUp seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3705.1, Definition 3705.2, and Definition 3705.3 seed the finite comparison route between Cauchy-criterion reads and convergence-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected limits, countable-choice threshold selection, host convergence, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bidirectional handoff exactness, shared tolerance-readback exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyComparisonUp.

Constructive story: A $\mathsf{BishopCauchyComparison}^{\uparrow}$ packet is generated from source StreamName windows, a Cauchy-Criterion request, a RealModulusOfConvergence schedule, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2739

A Concrete Naming Certificate for $\mathsf{ClosedObservationTotalHost}^{\uparrow}$

The $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ packet realizes the total-host discipline of chapter 9370 as finite NameCert data. It sits beside the candidate-mediated normalization socket of chapter 3566, the MetaCIC normalization frontier of chapter 892, the total-host fuel handoff of chapter 962, and the reduction fuel boundary of chapter 970. The packet names a discipline on accepted host-observation records: a finite termination or observation row must be visible before substrate-step readback. It does not claim global normalization of arbitrary host calculi, substrate halting, or evaluator soundness outside the displayed rows.

Definition 2739.1 (ClosedObservationTotalHost carrier)**.** *A* $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, T, S, F, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the accepted host-observation row, displaying the closed observation record and its host-side typing boundary.*
2. *T is the totality row, recording the finite termination or normal-form readback required before the record can be consumed.*
3. *S is the substrate-step row, displaying the finite step-indexed substrate observation that the host record describes.*
4. *F is the refusal row for any consumer that asks to read S before A, T have been displayed.*
5. *R is the readback row binding the host observation to the substrate-step record without asserting substrate halting.*
6. *H records componentwise* hsame *transport for accepted observation, totality, substrate-step, refusal, readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a closed-observation request through A, T, S, F, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedObservationTotalHost}^{\uparrow}$*, chapter 9370,* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$*,* $\mathsf{ReductionFuelBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, T, S, F, R, H, C, P, N$. It has no coordinate for unbounded substrate halting, non-total host terms, erased termination evidence, host evaluator soundness, completed consistency, global confluence, or quotient syntax equality.

Theorem 2739.2 (ClosedObservationTotalHost NameCert obligations)**.** *For every accepted* ClosedObservationTota *carrier $O = (A, T, S, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$ *surface consists exactly of accepted observation exposure, finite termination or normal-form exposure, substrate-step readback, pre-read refusal, readback binding, componentwise transport, displayed replay, provenance exhaustion, and local naming.*

Every accepted host-observation record exposes A and T before any consumer can read S. The substrate row is therefore a finite step-indexed observation named by a total host record, not a claim that the substrate computation halts or that arbitrary host computations are accepted.

Lean4 ✓ `BEDC.Derived.ClosedObservationTotalHostUp.ClosedObservationTotalHostNameCertObligations`

Proof. Project the carrier of Definition 3706.1. The accepted record A and totality row T are distinct rows, so accepting the host observation does not erase the termination or normal-form evidence.

The substrate row S is downstream of A, T. If a consumer asks for substrate readback first, the refusal row F is the only available route, so the packet records the total-host discipline as a finite ordering of rows.

The row R then binds the displayed host observation to the displayed substrate step. This binding is step-indexed and finite; it is not a substrate-halting theorem and does not import an unbounded evaluator.

Finally H, C, P, N transport, replay, source, and name exactly these rows. Since no coordinate carries non-total host terms, erased termination evidence, host evaluator soundness, global confluence, quotient syntax equality, or completed consistency, none of those principles can be exported from the packet. □

Falsifiable prediction: Every accepted host-observation record exposes a finite termination or observation row before any substrate-read row is available to a consumer.

Independence witness: normalization-candidate-socket, metacicnormalizationfrontier, continuationtrace, certificate-machine: $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries accepted observation, finite totality, substrate-step, refusal, readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3706.1 and Definition 3706.2 seed the finite total-host discipline for closed observation records before substrate-step readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ClosedObservationTotalHostUp.ClosedObservationTotalHostNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close global host normalization, substrate halting, host evaluator soundness, global confluence, quotient syntax equality, completed consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, accepted-observation exposure, totality-row precedence, substrate-read refusal, readback binding, transport, replay, provenance, and local naming for BEDC.Derived.ClosedObservationTota

Constructive story: A $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ packet is generated from an accepted host-observation row, a finite totality row, a substrate-step row, a pre-read refusal row, a readback binding row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2740

A Concrete Naming Certificate for $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ packet records the finite BEDC route by which regular rational Cauchy addition and product share one source surface, one window discipline, one dyadic arithmetic ledger, and one terminal real-name seal. It refines the binary route of chapter 3708 by isolating the coherence rows that make the addition and product certificates descend through the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ interfaces. The packet supplies later Real algebra consumers with a finite $\mathsf{NameCert}$ surface for shared operation descent, without importing quotient Cauchy classes, host real arithmetic, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2740.1 (RegularCauchyAlgebraCoherence carrier)**.** *A* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, D, A, M, S, T, C, P, N)$$

with the following rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect both sources at the requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the displayed slack conversions, endpoint arithmetic, comparison checks, addition budgets, product budgets, and magnitude bounds.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *descent row. It consumes only L, R, W, D and returns the regular addition readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent row. It consumes only L, R, W, D and returns the regular product readback.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *operation seal. It exposes an addition or product real-name handoff only after the corresponding regular readback row has been displayed.*
7. *T records componentwise* hsame *transport for the sources, window, dyadic ledger, addition row, product row, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an operation request through $L, R, W, D, A, M, S, T, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyAlgebraCoherence$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegularCauchyAddition$^\uparrow$, RegularCauchyProduct$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{RegularCauchyAlgebraCoherence}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $L, R, W, D, A, M, S, T, C, P, N$. It has no coordinate for quotient-stream equality, a host real operation, a selected limit, countable choice, trichotomy, or an ambient Real$^\uparrow$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAlgebraCoherenceUp.RegularCauchyAlgebraCoherenceTasteGate`
`single carrier alignment`

Theorem 2740.2 (RegularCauchyAlgebraCoherence NameCert obligations)**.** *For every accepted* RegularCauchyAlgebraCoherence$^\uparrow$ *carrier*

$$K = (L, R, W, D, A, M, S, T, C, P, N),$$

the local NameCert$_{\text{RegularCauchyAlgebraCoherence}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^\uparrow$ *sources, the shared* StreamName$^\uparrow$ *window, the* DyadicRatCore$^\uparrow$ *ledger, the* RegularCauchyAddition$^\uparrow$ *row, the* RegularCauchyProduct$^\uparrow$ *row, the* Real$^\uparrow$ *seal, transport, replay, provenance, and local naming;*
2. *source coherence, requiring addition and product to read the same ordered source pair L, R;*
3. *window coherence, requiring both operation rows to inspect the source pair only through W;*
4. *dyadic ledger coherence, requiring every addition and product estimate to consume only the displayed arithmetic and comparison entries of D;*
5. *descent separation, requiring A to carry the addition descent and M to carry the product descent while sharing the same upstream rows;*
6. *operation sealing, requiring S to expose a* Real$^\uparrow$ *handoff only after the selected operation row has produced its regular readback;*
7. *stability and replay, requiring T and C to preserve the full route $L, R, W, D, A, M, S, P, N$ under* hsame *and* Cont *without adding any excluded analytic or quotient principle.*

No RegularCauchyAlgebraCoherence NameCert obligation is stored outside these rows.

Proof. Carrier exposure is the projection list in Definition 3707.1. The displayed data rows are exactly L, R, W, D, A, M, S, and the structural rows are exactly T, C, P, N.

Source and window coherence follow from the single source pair and the single window row. Since both A and M cite L, R, W as their upstream surface, there is no second source pair or hidden window service from which either operation can draw.

The dyadic ledger row gives the common arithmetic budget. Addition consumes it through A, product consumes it through M, and the theorem asserts only those two separated descent rows over the same ledger.

Finally, S, T, C, P, N close the finite certificate surface. The seal exposes only regular readback data as a Real$^\uparrow$ handoff, while transport, replay, provenance, and local naming preserve the displayed route. None of the excluded principles has a carrier coordinate. □

Theorem 2740.3 (RegularCauchyAlgebraCoherence operation seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier reads regular Cauchy addition or product data through the finite route*

$$L, R \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto T, C, P, N.$$

Consequently the consumer obtains the two regular source rows, the shared stream window, the dyadic ledger, the addition descent row, the product descent row, the operation seal, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ operation request. By Definition 3707.1, the request has access to the source rows only as L, R, and it has access to their finite observations only through W.

The arithmetic step is then forced through D. Addition reads A, product reads M, and both descent rows are downstream of the same source, window, and dyadic ledger rows recorded in Definition 3707.2.

The seal S is the only Real-facing coordinate. It can expose an addition or product handoff only after the corresponding regular readback is displayed, and the rows T, C, P, N preserve that same finite route. Therefore a $\mathsf{Real}^{\uparrow}$ consumer has no coordinate from which to extract any excluded quotient, host, choice, order, or completeness datum. □

Theorem 2740.4 (RegularCauchyAlgebraCoherence non-escape)**.** *Let* $K = (L, R, W, D, A, M, S, T, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier. Any algebraic fact exported from* K *is a rowwise consequence of the displayed regular source rows, shared stream window, dyadic ledger, operation descent rows, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly L, R, W, D, A, M, S, and the structural coordinates are exactly T, C, P, N.

An exported addition fact must pass through A after the common source, window, and ledger rows are displayed. An exported product fact must pass through M after the same upstream rows are displayed. A Real-facing fact must then pass through S.

The remaining rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot introduce a quotient relation, host operation, selected limit, choice principle, trichotomy principle, or completeness principle, while Pkg and NameCert record only the finite packet. Hence every exported algebraic fact is confined to the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3707.1, Definition 3707.2, Definition 3707.3, and Definition 3707.4 seed the finite coherence route for addition and product descent over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source coherence, window coherence, dyadic ledger coherence, descent separation, operation sealing, stability, replay, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchyAlgebraCoherenceUp.

Constructive story: A $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, one shared $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegularCauchyAddition}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$ descent rows, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2741

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2741.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through* $X, Y, W, D, A, M, S, H, P, N$.
9. *P is the* Pkg *provenance over* CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 2741.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* CauchySequenceAlgebra$^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *operation budgets,* RegularCauchyAddition$^{\uparrow}$ *route,* RegularCauchyProduct$^{\uparrow}$ *route,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* RegSeqRat$^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 2741.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 2741.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 2741.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 2741.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2742

A Concrete Naming Certificate for CauchySequenceProductStability$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchySequenceProductStability$^{\uparrow}$ packet records the finite stability route by which two regular rational Cauchy source rows keep their product inside the Real and Completion interface. It gathers the regular product route of chapter 461, the product criterion of chapter 1586, the synchronized window discipline of chapter 126, the dyadic budget rows of chapter 291, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet is a finite NameCert carrier for product stability, not host multiplication of completed reals, quotient equality of streams, selected limits, countable choice, trichotomy, or ambient completeness.

Definition 2742.1 (Cauchy sequence product stability carrier)**.** *A* CauchySequenceProductStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A and B are the two* RegSeqRat$^{\uparrow}$ *source rows whose regular Cauchy schedules supply the two input sequences.*
2. *W_A and W_B are synchronized* StreamName$^{\uparrow}$ *window rows for the same product request, precision demand, and observation index.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *product-budget row, recording input tolerances, finite magnitude bounds, cross-error terms, and the output tolerance assigned to the product window.*
4. *P is the* RegularCauchyProduct$^{\uparrow}$ *route row, read only after A, B, W_A, W_B, D have been displayed.*
5. *K is the* CauchyProductCriterion$^{\uparrow}$ *acceptance row for the displayed product route.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced by the product route under the dyadic budget and criterion row.*
7. *E is the* Real$^{\uparrow}$ *seal row, consumed only after the regular-rational product readback R has been displayed.*
8. *H records componentwise* hsame *transport for $A, B, W_A, W_B, D, P, K, R, E, C, L, N$ and the local product-stability packet shape.*

9. *C records displayed* Cont *replay from the two source rows through the synchronized windows, dyadic budget, product route, criterion row, readback row, and real seal.*

10. *L is the* Pkg *provenance over* $\mathsf{CauchySequenceProductStability}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, W_A, W_B, D, P, K, R, E, H, C, L, N$. It has no coordinate for host multiplication of completed reals, quotient stream equality, a selected product limit, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchySequenceProductStabilityUp.CauchySequenceProductStabilityTasteGate`
`single carrier alignment`

Theorem 2742.2 (Cauchy sequence product stability NameCert obligations)**.** *For an accepted* $\mathsf{CauchySequenceProductStability}^{\uparrow}$ *carrier $S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$ *obligations are exactly carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic product-budget exactness, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *acceptance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport through H, displayed continuation replay through C, provenance through L, and local naming through N.*

Proof. Carrier admission is projection from Definition 3709.1. The tuple displays the two source rows, synchronized windows, dyadic product budget, product route, criterion row, regular-rational readback, real seal, and structural rows.

The source and window obligations are read from A, B, W_A, W_B. These rows expose the two regular schedules and align them before the product route is available.

The budget and route obligations are read from D, P, K. The dyadic row records the finite tolerance and magnitude data, the regular product route consumes that data, and the criterion row accepts the displayed product route for downstream readback.

The rows R, E, H, C, L, N then provide the regular-rational boundary, real-seal boundary, transport, replay, provenance, and local name. Since no other coordinate occurs in the carrier, the listed rows exhaust the certificate surface. □

Theorem 2742.3 (Cauchy sequence product budget stability)**.** *Let $S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$ be an accepted* $\mathsf{CauchySequenceProductStability}^{\uparrow}$ *carrier. Replacing the two source rows, synchronized window rows, dyadic product budget, product route, criterion row, readback row, and real seal by displayed componentwise* hsame *rows, and replaying the replacement through C, preserves the admissible product-stability route*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto P \longmapsto K \longmapsto R \longmapsto E.$$

Thus every Real and Completion consumer of the product reads the same finite stability packet and cannot introduce host multiplication, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only route from the two source sequences to the terminal real seal is the displayed chain through the synchronized windows, dyadic budget, regular product route, criterion row, regular-rational readback, and real seal.

The replacement hypothesis is internal to the structural rows. The row H gives componentwise hsame transport for every displayed product-stability coordinate, and C replays the continuation through the transported coordinates in the same order.

Therefore the transported packet still reaches D, then P, then K, then R, then E. The provenance row L and local name N preserve the same finite packet and do not add a second product operation.

Since the carrier has no coordinate for host multiplication, quotient streams, selected limits, choice, trichotomy, or ambient completeness, the budget-stable route is exactly the displayed BEDC product-stability route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceProductStability}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore product budget, a RegularCauchyProduct route, a CauchyProductCriterion acceptance row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3709.1, Definition 3709.2, and Definition 3709.3 seed the finite product-stability packet for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic product-budget exactness, RegularCauchyProduct routing, CauchyProductCriterion acceptance, RegSeqRat readback, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.CauchySequenceProductStabilityUp.

Chapter 2743

A Concrete Naming Certificate for $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$

Following Oresme and Bishop–Bridges constructive analysis, the $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ packet names the dyadic-block lower-bound route for the harmonic series as finite BEDC data. It sits beside the divergence refusal surface of chapter 2047, the positive condensation route of chapter 3345, the rational comparison source of chapter 37, the dyadic arithmetic ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback row of chapter 301, and the terminal real-name seal of chapter 38. The carrier records only displayed finite partial sums, dyadic block schedules, rational comparisons, tolerance ledgers, readback windows, transport, replay, provenance, and local naming.

Definition 2743.1 (OresmeHarmonicDivergence carrier)**.** *An* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, B, Q, D, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *S is the harmonic partial-sum row, exposing only finite sums of the displayed terms $1, 1/2, 1/3, \ldots$ through inspected windows.*
2. *B is the dyadic block schedule, listing the finite intervals $[2^k, 2^{k+1})$ requested by the packet and the endpoints used by each partial-sum comparison.*
3. *Q is the $\mathsf{Rat}^{\uparrow}$ comparison row, carrying the rational lower-bound entries used inside each displayed block.*
4. *D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, recording powers of two, reciprocal denominators, and the finite comparison budget used to show that each inspected block contributes at least $1/2$.*
5. *W is the $\mathsf{StreamName}^{\uparrow}$ window row for the harmonic terms, block endpoints, rational comparisons, and partial-sum reads.*
6. *R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback row for the displayed partial sums and block lower bounds.*
7. *E is the $\mathsf{Real}^{\uparrow}$ seal row reached only after S, B, Q, D, W, R have been displayed.*
8. *T records componentwise* hsame *transport for the partial-sum row, block schedule, rational comparison row, dyadic ledger, stream windows, readback row, Real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a block request through the harmonic partial sums, dyadic schedule, rational comparison row, dyadic ledger, stream windows, readback row, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$, $\mathsf{SeriesDivergenceTest}^{\uparrow}$, $\mathsf{CauchyCondensationRealSeries}$ $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, Q, D, W, R, E, T, C, P, N$. It has no coordinate for a host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OresmeHarmonicDivergenceUp.OresmeHarmonicDivergenceTasteGate single carrier alignment`

Theorem 2743.2 (OresmeHarmonicDivergence NameCert obligations)**.** *For every accepted* OresmeHarmonicDiverg *carrier $H = (S, B, Q, D, W, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed harmonic partial-sum row, dyadic block schedule, rational comparison row, dyadic tolerance ledger, stream-window row, readback row, Real seal, transport, replay, provenance, and local naming rows;*

2. *block schedule exactness, requiring B to expose precisely the inspected finite intervals $[2^k, 2^{k+1})$ and their partial-sum endpoints before any lower-bound comparison is read;*

3. *rational lower-bound visibility, requiring Q, D, W, R to display the finite comparison that every term in the inspected block is at least $1/2^{k+1}$ and that the whole block contributes at least $1/2$;*

4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to remain downstream of the displayed finite block route and to export no host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, or ambient completeness principle;*

5. *ledger exhaustion, requiring T, C, P, N to transport, replay, source, and name the same finite route without adding unlisted analytic or choice data.*

No Oresme harmonic-divergence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.OresmeHarmonicDivergenceUp.OresmeHarmonicDivergenceNameCertObligations`

Proof. Project the carrier of Definition 3711.1. The accepted packet exposes exactly the harmonic partial-sum row, the dyadic block schedule, the rational comparison row, the dyadic tolerance ledger, stream windows, readback, Real seal, and structural rows.

The block schedule is finite. For each inspected k, the row B gives the interval $[2^k, 2^{k+1})$ and the partial-sum endpoints that the consumer may read from S.

The lower-bound comparison is carried by Q, D, W, R. The schedule W selects the finitely many terms in the displayed block, Q supplies the rational comparison against $1/2^{k+1}$, D records the dyadic denominator and tolerance budget, and R reads back the block contribution.

The Real-facing route is downstream of those rows. The seal E can only be reached after the finite block comparison has passed through the displayed readback, while T, C, P, N preserve and name the same route.

Since a host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, and ambient completeness are not coordinates of H, the listed rows exhaust the local NameCert surface. □

Theorem 2743.3 (Oresme dyadic block lower bound)**.** *Every accepted* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ *packet exports the finite lower-bound route for each displayed block* $[2^k, 2^{k+1})$ *through*

$$S, B \longmapsto Q, D, W, R \longmapsto E, T, C, P, N.$$

Thus the packet records the Oresme block fact that each inspected dyadic block contributes at least $1/2$ *to the harmonic partial-sum row, while exporting no host infinite sum, quotient series equality, countable choice, selected endpoint, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted packet $H = (S, B, Q, D, W, R, E, T, C, P, N)$. A consumer requesting the block lower bound must first read the partial-sum row S together with the dyadic block schedule B.

For a displayed block $[2^k, 2^{k+1})$, the window row W exposes exactly the finitely many harmonic terms in that interval. The comparison row Q and dyadic ledger D show that every such term is bounded below by $1/2^{k+1}$.

There are 2^k displayed terms in the block, so the same finite ledger records the rational lower bound $2^k/2^{k+1} = 1/2$. The readback row R attaches this lower bound to the corresponding block contribution of the partial-sum row.

The seal row E and the structural rows T, C, P, N only carry this finite route forward. Because the carrier has no field for host infinite sums, quotient series equality, countable choice, selected endpoints, or ambient completeness, the exported statement is exactly the finite dyadic-block lower-bound route. □

Falsifiable prediction: Every accepted $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ block read factors through the harmonic partial-sum row, dyadic block schedule, RatUp comparison row, DyadicRatCore tolerance ledger, StreamName window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: series-divergence-test, cauchy-condensation-real-series, rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the harmonic partial-sum row, dyadic block schedule, RatUp comparison row, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3711.1, Definition 3711.2, and Definition 3711.3 seed the finite Oresme dyadic-block lower-bound carrier for harmonic partial sums, rational comparisons, dyadic ledgers, stream windows, readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient series equality, countable choice, selected endpoints, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, block schedule exactness, rational lower-bound visibility, RealUp seal nonescape, ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.OresmeHarmonicDiverge

Constructive story: An $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ packet is generated from a harmonic partial-sum row, a dyadic block schedule, RatUp comparison rows, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2744

A Concrete Naming Certificate for $\mathsf{Subdifferential}^{\uparrow}$

Following Rockafellar and Boyd–Vandenberghe, the $\mathsf{Subdifferential}^{\uparrow}$ packet records the finite supporting-hyperplane data that makes a convex subgradient usable inside BEDC without importing a separation theorem. It sits beside the convex-set carrier of chapter 208, the conjugacy rows of chapter 3563, the duality handoff of chapter 321, the order boundary of chapter 2297, and the stationarity surface of chapter 300. The chapter names the local subgradient certificate only; it does not assert Hahn–Banach separation, quotient function-space duality, selected suprema, or choice-selected subgradients.

2744.1 Subdifferential Carrier

Definition 2744.1 (Subdifferential carrier)**.** *A* $\mathsf{Subdifferential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, x, \xi, E, L, G, K, H, C, P, N)$$

with the following visible rows:

1. *F is the displayed convex functional row, including its finite convex-domain, convex-combination, value-comparison, and lower-support ledgers;*
2. *x is the displayed base point row inside the carrier domain of F;*
3. *ξ is the displayed dual covector row, read only through the finite pairing tests carried by the packet;*
4. *E is the pairing-evaluation ledger recording the finite values $\langle \xi, y-x\rangle$ that are actually inspected;*
5. *L is the supporting-inequality ledger asserting, for each displayed comparison point y, the row $F(y) \geq F(x) + \langle \xi, y - x\rangle$;*
6. *G is the epigraph row that reads the same support inequality as a displayed lower supporting face of the epigraph carrier;*
7. *K is the cone-boundary row connecting the finite support inequality to the* $\mathsf{PositiveCone}^{\uparrow}$ *order boundary and to the stationarity rows later consumed by* $\mathsf{KKT}^{\uparrow}$*;*
8. *H stores componentwise* hsame *transport for the functional, base point, covector, pairing ledger, support ledger, epigraph row, cone boundary, continuation row, provenance row, and local naming row;*

9. *C stores displayed* Cont *replay from a subgradient request through the convex functional, pairing, support, epigraph, cone-boundary, and downstream handoff rows;*
10. *P is the* Pkg *provenance over* $\mathsf{Subdifferential}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{Subdifferential}^{\uparrow}}$ *row.*

The carrier contains no ambient locally convex dual, no quotient function space, no selected separating hyperplane, no selected supremum in a conjugate formula, and no rule choosing a subgradient when the displayed support ledger is absent.

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialTasteGate single carrier alignment`

2744.2 Subdifferential Certificate

Theorem 2744.2 (Subdifferential NameCert obligations)**.** *For every accepted* $\mathsf{Subdifferential}^{\uparrow}$ *carrier* $S = (F, x, \xi, E, L, G, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Subdifferential}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the convex functional row, base point row, dual covector row, finite pairing-evaluation ledger, supporting-inequality ledger, epigraph row, cone boundary, transport, replay, provenance, and local naming row;*
2. *support-inequality exactness, requiring every exported subgradient read to pass through the displayed row $F(y) \geq F(x) + \langle \xi, y - x \rangle$ for the listed comparison point;*
3. *epigraph compatibility, requiring the support ledger and epigraph row to name the same lower supporting face and the same base point;*
4. *Fenchel handoff, requiring any conjugacy or duality consumer to pass through the displayed* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *rows before reading a dual certificate;*
5. *KKT and normal-cone consumer boundary, requiring stationarity consumers to read the cone-boundary row and refusing any normal-cone claim not carried by the finite support ledger;*
6. *non-escape from Hahn–Banach, quotient function spaces, selected suprema, and choice-selected subgradients.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier namecert obligations`

Proof. Project the accepted carrier. The rows F, x, ξ, E, L, G, K are the entire analytic surface, and the rows H, C, P, N are the entire structural surface.

Support exactness is a ledger condition. A consumer can call ξ a subgradient only at a displayed comparison point whose support inequality is present in L and whose pairing value is present in E.

Epigraph compatibility and the Fenchel handoff are then read through G and the provenance row P. The same lower support row is available to epigraph, conjugacy, and duality consumers, but no hidden supremum or separating functional is added by that handoff.

Finally the stationarity boundary passes through K. Since no coordinate of S names Hahn–Banach separation, a quotient dual space, a selected supremum, or a choice functional for subgradients, those principles cannot be exported by the local certificate. □

Theorem 2744.3 (Subdifferential support inequality exactness)**.** *For every accepted* Subdifferential$^{\uparrow}$ *carrier* $S = (F, x, \xi, E, L, G, K, H, C, P, N)$*, a public subgradient read is exact only when its displayed comparison point is named in the finite support ledger* L *and its pairing value is named in the finite pairing ledger* E*. The row* L *must expose the inequality* $F(y) \geq F(x) + \langle \xi, y - x \rangle$ *for that same* y*, and no consumer may replace this row by an ambient separation principle or by a chosen supporting functional.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier support inequality exactness`

Proof. Project the carrier coordinates E and L. The pairing ledger E is the only row that evaluates $\langle \xi, y - x \rangle$, and the support ledger L is the only row that compares this value with $F(y)$ and $F(x)$.

Since the carrier contains no Hahn–Banach coordinate and no selection rule for subgradients, a consumer can export only the support inequalities that are already displayed in L. Exactness is therefore the rowwise agreement of the finite comparison point, the pairing value, and the listed inequality. □

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialSupportInequalityExactness`

Theorem 2744.4 (Subdifferential epigraph and Fenchel handoff)**.** *Every* ConvexConjugate$^{\uparrow}$ *or* FenchelDuality$^{\uparrow}$ *read of an accepted* Subdifferential$^{\uparrow}$ *carrier factors through the displayed epigraph row* G*, the support ledger* L*, the pairing ledger* E*, the componentwise transport row* H*, and the provenance row* P*. The handoff may expose a conjugacy or duality consumer, but it does not add selected suprema, quotient dual spaces, or an unlisted separating hyperplane.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier epigraph fenchel handoff`

Proof. Read the epigraph coordinate G together with the ledgers E and L. These rows identify the lower supporting face that the conjugacy and duality consumers may inspect.

The provenance row P names the ConvexConjugate$^{\uparrow}$ and FenchelDuality$^{\uparrow}$ dependencies, while H carries the same finite rows across permitted comparisons. Because no coordinate stores a selected supremum or quotient dual object, the Fenchel-facing handoff is exactly the displayed epigraph support route. □

Theorem 2744.5 (Subdifferential KKT boundary)**.** *Every* PositiveCone$^{\uparrow}$ *or* KKT$^{\uparrow}$ *stationarity read of an accepted* Subdifferential$^{\uparrow}$ *carrier must pass through the cone-boundary row* K*, the support ledger* L*, the pairing ledger* E*, transport* H*, replay* C*, provenance* P*, and the local naming row* N*. The boundary records only finite stationarity data and refuses Hahn–Banach separation, normal-cone completion, and choice-selected subgradients not present in the support ledger.*

Proof. Start from the cone-boundary coordinate K. It is the only carrier row that connects the support inequality to the positive-cone and KKT stationarity surface.

The ledgers E and L supply the finite pairing and support data used by that boundary, while H, C, P, N transport, replay, source, and name the same packet. With no coordinate for Hahn–Banach separation or an ambient normal-cone completion, the stationarity consumer receives exactly the finite boundary listed above. □

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier kkt boundary`

Closure status for NameCert$_{\text{Subdifferential}^{\uparrow}}$**.**

Constructive story: A Subdifferential$^{\uparrow}$ packet is generated from a convex functional row, base point row, dual covector row, finite pairing-evaluation ledger, supporting-inequality ledger, epigraph row, cone-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{Subdifferential}\uparrow}$, scopedClosed)

Scope closed: Definition 3712.1, Definition 3712.2, Definition 3712.3, Definition 3712.4, and Definition 3712.5 bind the finite subgradient obligation surface between convex sets, convex conjugacy, Fenchel duality, positive cones, and KKT stationarity.

Formal verification: FormalStatus(NameCert$_{\mathsf{Subdifferential}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, quotient function-space duality, selected suprema, choice-selected subgradients, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public subgradient consumer theorem and later Lean verification for BEDC.Derived.SubdifferentialUp.

Chapter 2745

A Concrete Naming Certificate for $\mathsf{ProximalMap}^{\uparrow}$

Following Rockafellar's convex-analysis treatment of subdifferentials and resolvents, and Boyd–Vandenberghe's finite optimization presentation of proximal operators, the $\mathsf{ProximalMap}^{\uparrow}$ packet records the displayed proximal minimization surface as BEDC NameCert data. It sits after the convex-set carrier of chapter 208, the conjugacy rows of chapter 3563, the Fenchel-duality handoff of chapter 321, the positive-cone order boundary of chapter 2297, the stationarity ledger of chapter 300, the real seal of chapter 38, and the vector-space source of chapter 47. The packet gives subdifferential and monotone-operator consumers a finite resolvent-facing route without asserting a selected global minimizer, Hilbert projection theorem, host lower semicontinuity, or ambient completeness.

2745.1 ProximalMap Carrier

Definition 2745.1 (ProximalMap carrier)**.** *A* $\mathsf{ProximalMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, F, u, z, R, D, K, G, O, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed vector-space source row, read through* $\mathsf{VecSpace}^{\uparrow}$ *and the real scalar seal of* $\mathsf{Real}^{\uparrow}$*;*
2. *F is the convex functional and domain row, including the visible* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{ConvexConjugate}^{\uparrow}$ *surfaces used by the request;*
3. *u is the input point row in the carried vector-space source;*
4. *z is the displayed candidate output row;*
5. *R is the real penalty and distance-evaluation row used for the displayed expression $F(y)+\frac{1}{2\lambda}\|y-u\|^2$ at the finitely inspected comparison points;*
6. *D is the finite dominance ledger recording that the candidate row is no worse than each displayed comparison row;*
7. *K is the KKT stationarity row tying the candidate, penalty, cone boundary, and finite residual rows to the* $\mathsf{KKT}^{\uparrow}$ *packet;*

8. *G is the Fenchel and convex-conjugate handoff row, read only after F, R, D, K have been displayed;*
9. *O is the convex-resolvent consumer row for later subdifferential or monotone-operator packets;*
10. *H records componentwise* hsame *transport for the vector-space source, convex functional, input, candidate, real penalty, dominance ledger, KKT stationarity, Fenchel handoff, resolvent consumer, replay, provenance, and local naming rows;*
11. *C records displayed* Cont *replay from a proximal request through convex admission, real penalty evaluation, stationarity, duality, dominance, and resolvent-facing readback;*
12. *P is the* Pkg *provenance over* $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient Hilbert projection theorem, a selected minimizer outside the displayed candidate row, host lower semicontinuity, global coercivity, quotient vector equality, or an existence theorem for every input.

Lean4 (definition) `BEDC.Derived.ProximalMapUp.ProximalMapUp`
Lean4 ✓ `BEDC.Derived.ProximalMapUp.ProximalMapTasteGate single carrier alignment`

2745.2 ProximalMap Certificate

Theorem 2745.2 (ProximalMap NameCert obligations)**.** *For every accepted* $\mathsf{ProximalMap}^{\uparrow}$ *carrier* $Q = (X, F, u, z, R, D, K, G, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the vector-space source, convex functional, input point, candidate output, real penalty row, dominance ledger, KKT row, Fenchel handoff, resolvent consumer row, transport, replay, provenance, and local naming;*
2. *penalty evaluation exactness, requiring every proximal comparison to read the same* $\mathsf{Real}^{\uparrow}$ *penalty row R and the same* $\mathsf{VecSpace}^{\uparrow}$ *source row X before any dominance claim is exported;*
3. *convex-domain compatibility, requiring the candidate and comparison points to be admitted through the* $\mathsf{ConvexSet}^{\uparrow}$ *row carried by F;*
4. *stationarity compatibility, requiring the displayed residual, cone boundary, and multiplier data to pass through the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{PositiveCone}^{\uparrow}$ *rows before any optimality-facing read is public;*
5. *Fenchel and conjugacy handoff, requiring dual or subgradient consumers to pass through* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *only after the dominance and stationarity rows have been displayed;*
6. *non-escape from global minimizer selection, host projection theorems, quotient vector equality, lower-semicontinuity assumptions, and ambient completeness.*

Proof. Project the accepted carrier. The rows $X, F, u, z, R, D, K, G, O$ are the complete proximal surface, while H, C, P, N are the structural route that transports, replays, sources, and names the same packet.

Penalty exactness is a finite ledger condition. A comparison with a point y can be read only when the relevant vector-space rows in X, convex-domain rows in F, and real penalty value in R are all displayed, and the dominance row in D names that comparison.

Stationarity and order compatibility are then read through K and the provenance row P. The KKT and positive-cone rows may inspect only the displayed residual, cone, multiplier, real, and vector entries already present in the packet.

The Fenchel and conjugacy handoff is downstream of the same displayed data. Since no coordinate of Q stores a selected global minimizer, a Hilbert projection theorem, quotient vector equality, lower-semicontinuity evidence, or ambient completeness, those principles cannot be exported by the local certificate. □

Theorem 2745.3 (ProximalMap convex-resolvent handoff)**.** *Every subdifferential, monotone-operator, or splitting-method consumer of an accepted* $\mathsf{ProximalMap}^{\uparrow}$ *packet must factor through the finite route*

$$X, F, u, z, R, D, K, G, O \longmapsto H, C, P, N.$$

The consumer may read the candidate z as a resolvent-facing output only after the convex-domain row, real penalty row, dominance ledger, KKT stationarity row, positive-cone boundary, and Fenchel-duality handoff have all been displayed. It receives no selected minimizer for unlisted inputs, no maximal-monotone theorem, no quotient vector equality, no host projection semantics, and no ambient completeness principle.

Proof. Begin with the candidate row. By Definition 3713.1, the row z is only one displayed output candidate in the finite packet, and it is tied to the input row u and vector-space source row X.

The dominance ledger D and penalty row R supply the proximal comparison surface. They are read through the same convex-domain row F and the same real scalar seal, so a consumer cannot replace the displayed comparisons by an external minimization predicate.

Next read the stationarity and duality boundary. The row K routes optimality-facing material through the KKT and positive-cone packets, while G routes conjugacy-facing material through the convex-conjugate and Fenchel-duality packets. The resolvent consumer row O is exposed only after those rows are present.

Transport, continuation replay, package provenance, and local naming preserve exactly this route through H, C, P, N. Since the carrier lacks coordinates for unlisted input selection, maximal-monotone closure, quotient vector equality, host projection semantics, or completeness, the handoff stops at the finite convex-resolvent boundary. □

Falsifiable prediction: Every accepted $\mathsf{ProximalMap}^{\uparrow}$ consumer read factors through a VecSpaceUp source, ConvexSetUp and ConvexConjugateUp functional rows, one displayed input, one displayed candidate output, a RealUp penalty row, a finite dominance ledger, KKTUp stationarity rows, PositiveConeUp order rows, FenchelDualityUp handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, convex-conjugate, fenchelduality, positive-cone, kkt, real, vecspace: $\mathsf{ProximalMap}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an input row, candidate output row, real penalty ledger, dominance comparisons, stationarity boundary, Fenchel handoff, and resolvent-facing consumer row in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProximalMap}^{\uparrow}$ packet is generated from a vector-space source, convex functional and domain rows, one input point, one candidate output, RealUp penalty evaluations, finite dominance comparisons, KKT stationarity, PositiveCone order, FenchelDuality and ConvexConjugate handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$, seed)

Scope closed: Definition 3713.1, Definition 3713.2, and Definition 3713.3 seed the finite proximal-resolvent certificate surface over convex, real, vector-space, KKT, cone, and Fenchel rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProximalMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a global proximal existence theorem, selected minimizers for all inputs, Hilbert projection semantics, maximal-monotone closure, quotient vector equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, penalty evaluation exactness, convex-domain compatibility, stationarity compatibility, Fenchel handoff, resolvent consumer boundary, transport, replay, provenance, and local naming for BEDC.Derived.ProximalMapUp.

Chapter 2746

A Concrete Naming Certificate for TypedNormalFormDecider$^{\uparrow}$

The TypedNormalFormDecider$^{\uparrow}$ packet realizes the normal-equality section of chapter 9379 as a finite BEDC certificate tying `inferTypeCtx` to bounded normal-form comparison. It sits beside the bounded normal-equality checker of chapter 752, the decidability witness of chapter 751, the closure-preservation audit trace of chapter 795, and the normalization audit map of chapter 9383. The packet names the typed checker and bounded comparison route; it refuses to turn that route into unrestricted beta-conversion decidability.

Falsifiable prediction: Every accepted TypedNormalFormDecider$^{\uparrow}$ packet displays `inferTypeCtx` rows, typed input rows, bounded normalizer output rows, finished-normalization witnesses when supplied, bounded normal-form comparison rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: bounded-normal-equality-checker, metacicdecidabilitywitness, closure-preservation audit, normalization audit: TypedNormalFormDecider$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries typed-checker admission, bounded normal-form comparison, finished-normalization side conditions, refusal, transport, replay, provenance, and local naming rows.

Definition 2746.1 (Typed normal-form decider carrier)**.** *A* TypedNormalFormDecider$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, L, R, T_L, T_R, F, N_L, N_R, E, W, Q, H, C, P, A)$$

with the following displayed rows:

1. *I is the `inferTypeCtx` checker row for the shared context and the represented input terms;*
2. *L and R are the two closed represented MetaCIC input term rows;*
3. *T_L and T_R are the optional inferred type rows produced by the checker, together with the displayed success or refusal status of each check;*
4. *F is the bounded fuel row used by the normalizer;*
5. *N_L and N_R are the bounded normal-form output rows obtained from L and R at fuel F;*
6. *E is the structural normal-form comparison row, read through `Term.eq` on N_L and N_R;*
7. *W stores the finished-normalization witnesses required before the comparison can be consumed as a typed normal-form equality result;*

8. *Q is the refusal row blocking unrestricted beta-conversion decidability, global normalization, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, and fuel selection;*
9. *H records componentwise* hsame *transport for checker, term, type, fuel, normal-form output, comparison, finished-witness, refusal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from a typed normal-form decision request through $I, L, R, T_L, T_R, F, N_L, N_R, E,$*
11. *P is the* Pkg *provenance over* $\mathsf{TypedNormalFormDecider}^{\uparrow}$*, the MetaCIC decidability audit map,* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$*, closure-preservation audit rows, normalization audit rows,* BHist, hsame, Cont, *and* NameCert*;*
12. *A is the local* $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a fuel-erased beta-conversion oracle, no proof that all closed typed terms normalize, no hidden subject-reduction discharge, no confluence theorem, and no host quotient over terms.

Lean4 ✓ `BEDC.Derived.TypedNormalFormDeciderUp.TypedNormalFormDeciderTasteGate single carrier alignment`

Theorem 2746.2 (Typed normal-form decider NameCert obligations)**.** *For the carrier of Definition 3714.1, the local* $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$ *surface consists exactly of typing-checker admission through I, T_L, T_R, bounded-normal comparison through F, N_L, N_R, E, finished-normalization side-condition exposure through W, explicit refusal through Q, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through A.*

Proof. Project the carrier coordinates. The checker side is exhausted by I, L, R, T_L, T_R, so a consumer obtains typed input only through the displayed `inferTypeCtx` rows.

The normal-form side is exhausted by F, N_L, N_R, E, W. The comparison row E can be consumed only with the bounded fuel and finished-normalization witnesses displayed in the same packet.

The refusal row Q is part of the certificate surface. It prevents the bounded route from being reread as global normalization, unrestricted beta-conversion decidability, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, or fuel selection.

Finally H, C, P, A preserve and name the same finite route. Future Nontrivial and FieldFaithful TasteGate instances must witness that none of these rows can be collapsed away or replaced by an unlisted oracle. □

Theorem 2746.3 (Typed normal-form decider bounded comparison boundary)**.** *Every consumer of an accepted* $\mathsf{TypedNormalFormDecider}^{\uparrow}$ *packet reaches a typed normal-form comparison only through the finite route*

$$\mathit{inferTypeCtx},\ L, R, T_L, T_R, F, N_L, N_R, E, W, Q, H, C, P, A.$$

The route exports typing-checker data and bounded finished-normal comparison data; it exports no unrestricted closed beta-conversion decider, no global normalizer, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, and no host quotient equality for MetaCIC terms.

Proof. Read the checker rows first. A typed comparison request must pass through I and the displayed optional type rows T_L, T_R before it can inspect normal forms.

Next read the bounded normal-form rows. The fuel F, the outputs N_L, N_R, the structural comparison E, and the finished-normalization witnesses W form the complete comparison surface.

The refusal row Q is then read before structural replay. It records that the packet has no coordinate for a stronger conversion, normalization, subject-reduction, confluence, closedness, quotient, or fuel-selection theorem.

Thus transport, replay, provenance, and local naming in H, C, P, A preserve only the displayed bounded route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$.

Constructive story: A $\mathsf{TypedNormalFormDecider}^{\uparrow}$ packet is generated from inferTypeCtx checker rows, closed input term rows, optional inferred type rows, bounded fuel, bounded normalizer outputs, structural normal-form comparison, finished-normalization witnesses, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3714.1, Definition 3714.2, and Definition 3714.3 seed the finite typed normal-form decider surface over typing-checker and bounded normal-form comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted beta-conversion decidability, global normalization, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typing-checker exactness, bounded-normal comparison exactness, finished-normalization side-condition exposure, refusal, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.TypedNormalFormDeciderUp.

Chapter 2747

A Concrete Naming Certificate for $\mathsf{ResolventIdentity}^{\uparrow}$

Following Dunford–Schwartz and Conway on the resolvent calculus for bounded operators, the $\mathsf{ResolventIdentity}^{\uparrow}$ packet records the finite BEDC surface on which the algebraic resolvent identity is read before any completed spectrum or host operator quotient is available. It sits at the operator-theory frontier beside the bounded-operator carrier of chapter 1644, the scalar operator-norm surface of chapter 3603, the resolvent-facing spectral-radius route of chapter 2352, and the compact-operator schedule boundary of chapter 1952.

Definition 2747.1 (Resolvent identity carrier)**.** *A* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the Banach source surface on which the operator rows are read.*
2. *T is the accepted* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the displayed operator action, linearity ledger, norm-bound endpoint, and transport data.*
3. *λ, μ are the two scalar-window rows whose separation witness is recorded before any inverse row is consumed.*
4. *A_λ, A_μ are the shifted-operator rows for $\lambda I - T$ and $\mu I - T$, each read through the same bounded-operator and operator-norm surface.*
5. *U_λ, U_μ are the displayed two-sided inverse-check rows for the two shifts.*
6. *I is the identity and composition ledger needed to compare $U_\lambda - U_\mu$ with $(\mu - \lambda)U_\lambda U_\mu$.*
7. *D is the domain and denominator ledger recording the scalar separation, multiplication order, and refusal of zero-denominator or detached inverse reads.*
8. *H records componentwise* hsame *transport for the operator, scalar, shifted-operator, inverse-check, identity, and denominator rows.*
9. *C records displayed* Cont *replay of the route from bounded operator and operator norm rows to the two resolvent checks and the identity ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed spectrum, a spectral measure, quotient operator equality, selected inverse outside the displayed two-sided checks, compact-operator eigenvalue lists, or countable choice.

Lean4 ✓ `BEDC.Derived.ResolventIdentityUp.ResolventIdentityTasteGate single carrier alignment`

Theorem 2747.2 (Resolvent identity NameCert obligations)**.** *For an accepted* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier* $R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-operator source admission through* T*, operator-norm scalar-bound admission for the two shifts, scalar-separation admission for* λ, μ*, two-sided inverse-check admission for* U_λ, U_μ*, identity and composition ledger exactness through* I*, denominator and refusal ledger exactness through* D*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Every downstream resolvent consumer must read the same rows before using the identity. It receives no completed spectrum, no spectral-radius limit theorem beyond the finite resolvent-facing boundary of chapter 2352, no compact-operator spectral decomposition, and no quotient equality of operators.

Proof. Project the carrier of Definition 3715.1. The rows $X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N$ are the whole accepted surface.

The bounded-operator and operator-norm rows expose the two shifted operators. The scalar rows record λ, μ and their separation before any inverse-check row is read, so no inverse can be selected outside the displayed denominator ledger.

The rows U_λ, U_μ supply only two-sided inverse checks for the displayed shifts. The identity ledger I then compares the two compositions and the scalar multiple on the same bounded-operator route.

Finally, H, C, P, N transport, replay, source, and name that finite route. Since the carrier has no coordinate for a completed spectrum, compact spectral decomposition, quotient equality, or choice, those objects cannot be part of the NameCert surface. □

Theorem 2747.3 (Resolvent identity two-sided law)**.** *Let* $R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$ *be an accepted* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier. If the two displayed inverse-check rows certify*

$$U_\lambda A_\lambda = A_\lambda U_\lambda = I_X \quad \textit{and} \quad U_\mu A_\mu = A_\mu U_\mu = I_X,$$

then every public resolvent read factors through the finite identity ledger

$$U_\lambda - U_\mu = (\mu - \lambda) U_\lambda U_\mu$$

with the same scalar-separation row, bounded-operator source row, operator-norm bound rows, and structural rows H, C, P, N*.*

The law exports only this two-sided displayed algebraic comparison. It does not export a completed resolvent set, a spectral theorem, quotient operator equality, selected approximate eigenvectors, compact-operator eigenvalue data, or countable choice.

Lean4 ✓ `BEDC.Derived.ResolventIdentityUp.ResolventIdentityCarrier two sided law`

Proof. Start with the two shifted-operator rows A_λ and A_μ. They are read from the same bounded operator T and scalar-window rows λ, μ, so their comparison is already inside the accepted packet.

Use the inverse-check rows U_λ and U_μ. The hypotheses certify both left and right inverse equations for the displayed shifts, and the denominator ledger D records the scalar separation needed to form $\mu - \lambda$.

Apply the identity and composition ledger I. It records the finite algebraic rearrangement of the two inverse equations and the composition $U_\lambda U_\mu$, producing the displayed equality without invoking a host quotient of operators.

The structural rows H, C, P, N preserve and name the same route for every public consumer. Since no carrier coordinate stores a completed spectrum, selected inverse, compact eigenvalue list, or choice schedule, the public law is exactly the finite two-sided resolvent identity. □

Falsifiable prediction: Every accepted $\mathsf{ResolventIdentity}^{\uparrow}$ consumer exposes the bounded operator row, both scalar-window rows, both shifted-operator rows, both two-sided inverse-check rows, the identity ledger, the denominator ledger, componentwise transport, replay, provenance, and local naming before reading the resolvent identity.

Independence witness: boundedlinearoperator, operatornorm, spectralradius, compactoperator: BoundedLinearOperatorUp supplies the action and norm-bound carrier but not the two scalar shifted-inverse ledger; OperatorNormUp supplies scalar norm bounds without the two-sided inverse checks; SpectralRadiusUp supplies a resolvent-facing boundary but not the algebraic two-shift identity packet; CompactOperatorUp supplies image-net schedules without the bounded-operator resolvent algebra.

Closure status for $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$.

Constructive story: A $\mathsf{ResolventIdentity}^{\uparrow}$ packet is generated from a Banach source surface, a bounded-operator row, two scalar-window rows, two shifted-operator rows, two two-sided inverse-check rows, identity and denominator ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3715.1, Definition 3715.2, Definition 3715.3, the falsifiable prediction, and the independence witness seed the finite two-shift resolvent identity route over BoundedLinearOperatorUp, OperatorNormUp, SpectralRadiusUp, CompactOperatorUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a completed resolvent set, a spectral theorem, quotient operator equality, selected approximate eigenvectors, compact-operator eigenvalue data, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, shifted-operator stability, inverse-check exactness, denominator-ledger exactness, two-sided identity replay, classifier transport, and consumer non-escape for BEDC.Derived.ResolventIdentit

Chapter 2748

A Concrete Naming Certificate for BoundedIntervalCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the BoundedIntervalCauchySubsequence$^{\uparrow}$ packet records a finite-window substitute for the bounded-interval subsequence argument used by completion-facing real analysis. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the interval projection route of chapter 3692, and the cofinal regular-Cauchy subsequence route of chapter 3515. The packet names the data needed by Real$^{\uparrow}$ and completion consumers to inspect a bounded interval source through explicit finite windows; it does not assert full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, or a choice-selected infinite subsequence.

Definition 2748.1 (Bounded interval Cauchy subsequence carrier)**.** *A* BoundedIntervalCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row, including endpoint* Real$^{\uparrow}$ *seals and the finite interval projection surface.*
2. *S is the source bounded sequence row, read only through displayed* StreamName$^{\uparrow}$ *observation windows.*
3. *W is the finite window ledger listing the inspected source positions, endpoint windows, and selected comparison windows.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger containing the mesh radius, endpoint comparisons, pairwise distance bounds, and late-window budgets used by the extraction route.*
5. *T is the monotone cofinal schedule row naming how each selected window is reached from an inspected finite source window.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *handoff row obtained from the displayed windows, dyadic ledger, and schedule.*
7. *E is the* RegularCauchySubsequenceStability$^{\uparrow}$ *row recording that the selected finite windows remain a regular Cauchy readback surface.*

8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the interval, window, dyadic, schedule, regular-readback, and subsequence-stability rows have been displayed.*
9. *H stores componentwise* hsame *transport for $I, S, W, D, T, Q, E, R, C, P, N$.*
10. *C stores displayed* Cont *replay of the extraction route*

$$I, S \longmapsto W, D, T \longmapsto Q, E \longmapsto R.$$

11. *P is the* Pkg *provenance over* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedIntervalProjection}^{\uparrow}$, $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, T, Q, E, R, H, C, P, N$. It has no coordinate for an ambient compactness theorem, open-cover subcover, host trichotomy, quotient stream equality, selected representatives, countable choice, or an unrecorded $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BoundedIntervalCauchySubsequenceUp.TasteGate.BoundedIntervalCauchySubsequenceTaste`
`single carrier alignment`

Theorem 2748.2 (Bounded interval Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *carrier*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded interval row I, source sequence row S, finite stream-window ledger W, dyadic tolerance ledger D, monotone cofinal schedule T, regular rational handoff Q, subsequence-stability row E, terminal real seal R, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

The obligation surface also includes classifier stability under the same displayed rows, dyadic tolerance exactness through D, finite-window visibility through W, cofinal schedule replay through T, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through Q,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through R, and provenance visibility through P, N.*

Proof. Carrier admission is projection from Definition 3716.1. The projection exposes exactly the interval, source, window, dyadic, schedule, readback, subsequence-stability, seal, transport, replay, provenance, and naming rows listed in the statement.

The classifier is componentwise on those rows. Hence any accepted comparison must preserve the same bounded interval row, the same finite source-window surface, the same dyadic ledger, and the same schedule route before it reaches the regular-readback and real-seal rows.

The nonescape clauses follow from the absence of further coordinates in the carrier. There is no slot for open-cover compactness, host trichotomy, quotient stream equality, countable choice, selected representatives, or an ambient completeness principle, so no local NameCert consumer can read such data from the packet. □

Theorem 2748.3 (Bounded interval dyadic window extraction)**.** *Let*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N)$$

be an accepted $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *carrier. Every finite extraction read from* K *factors through the displayed dyadic ledger* D *and the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W *before it may consume the monotone cofinal schedule* T *or the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* Q*.*

Consequently each selected comparison window is certified by endpoint interval data from I*, inspected source windows from* S*, dyadic radii and pairwise bounds from* D*, and replay data from* C*, rather than by a hidden infinite subsequence, trichotomy decision, open cover, quotient stream, or choice principle.*

Proof. Start with an accepted extraction read and project the carrier. The only rows that mention finite observations are W and D: W lists the inspected source and endpoint windows, while D records the mesh radius, endpoint comparisons, pairwise distance bounds, and late-window budgets attached to those observations.

The schedule row T is downstream from those finite ledgers. It records which displayed windows are reused along a monotone cofinal route; it does not add a new source of observations or a selection functional outside the packet.

The regular handoff row Q consumes W, D, T together with the interval source I, S. Replaying the route through C therefore forces every extraction read to pass through the finite dyadic-window data before any subsequence or regular Cauchy surface is available.

The forbidden principles have no carrier coordinate. Since the read has already exhausted the displayed rows, no infinite choice-selected subsequence, host trichotomy, open-cover compactness, quotient stream equality, or ambient completeness proof can be exported by the extraction route. □

Theorem 2748.4 (Bounded interval Cauchy subsequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *packet must read the route*

$$I, S \longmapsto W, D, T \longmapsto Q, E \longmapsto R$$

before consuming the terminal seal row R*. Thus the handoff to* $\mathsf{Real}^{\uparrow}$ *and completion consumers is a finite BEDC route through bounded-interval endpoint data, stream windows, dyadic tolerances, a cofinal schedule, regular rational readback, subsequence stability,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The handoff does not close full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, quotient stream equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The terminal seal R is downstream from the interval and source rows I, S, the finite ledgers W, D, the schedule row T, the regular handoff Q, and the subsequence-stability row E.

Use Definition 3716.3 to bind the extraction stage to W and D. This ensures that the cofinal schedule and regular handoff are read only after the finite window and dyadic tolerance rows have been displayed.

The transport and replay rows H, C preserve the same ordered route, while P, N record provenance and the local naming surface. A $\mathsf{Real}^{\uparrow}$ consumer can therefore consume R only as the final row of that route.

The carrier contains no coordinate for the excluded compactness or choice principles. The theorem is therefore a finite-window handoff to Real and completion work, not a claim of unrestricted subsequence compactness or open-cover compactness for closed intervals. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ packet is generated from a bounded interval row, a source sequence row, finite StreamName windows, a DyadicRatCore tolerance ledger, a monotone cofinal schedule, RegSeqRat readback, RegularCauchySubsequenceStability handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3716.1, Definition 3716.2, Definition 3716.3, and Definition 3716.4 seed the finite-window bounded-interval subsequence packet over displayed Real, RegSeqRat, StreamName, DyadicRatCore, BoundedIntervalProjection, and RegularCauchySubsequenceStability rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, quotient stream equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic window extraction, cofinal schedule replay, RegSeqRat handoff, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BoundedIntervalCauchySubsequenceUp.

Chapter 2749

A Concrete Naming Certificate for $\mathsf{BishopCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCompletion}^{\uparrow}$ packet names the finite boundary at which regular Cauchy rational names, stream windows, dyadic tolerance ledgers, filter-completion handoffs, and the terminal Real seal can be read as one completion-facing NameCert object. It sits on the L10 route supplied by the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic arithmetic ledger of chapter 291, the terminal real-name seal of chapter 38, the filter completion handoff of chapter 1569, and the universal extension boundary of chapter 397. The packet records a Bishop-style completion boundary for displayed names; it is not a quotient of streams, not a selected limit, not countable choice, not host real equality, and not an ambient completeness principle.

Definition 2749.1 (Bishop completion carrier)**.** *A* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, S, D, E, F, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row, carrying the rational readback and regularity budget visible to completion-facing consumers.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row from which the displayed rational observations in R are read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and precision requests.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D have been displayed.*
5. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row generated from the same finite tail windows and dyadic budgets.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row available only after the regular-name route and filter-completion handoff have both been exposed.*
7. *H records componentwise* hsame *transport for the regular name, stream window, dyadic ledger, Real seal, filter-completion handoff, universal-property row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through S, R, D, E, F, U.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows and their displayed replay paths. It has no coordinate for quotient stream equality, a selected representative, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionTasteGate single carrier alignment`

Theorem 2749.2 (Bishop completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier $B = (R, S, D, E, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed regular rational name, stream window, dyadic tolerance ledger, Real seal, filter-completion handoff, universal-property row, transport, replay, provenance, and local naming row;*
2. *finite-window exposure, requiring every completion-facing observation to pass through S before R or D can be read;*
3. *regular rational boundary exposure, requiring R and D to be displayed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E becomes available;*
4. *filter-compatibility exposure, requiring the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row F to be generated from the same S, R, D route;*
5. *universal-property exposure, requiring U to read only the already displayed regular-name, Real-seal, and filter-completion rows;*
6. *ledger non-escape, requiring H, C, P, N to export no quotient stream equality, selected representative, countable choice, host real equality, or ambient Real completeness coordinate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the accepted carrier through Definition 3717.1. The visible packet consists exactly of $R, S, D, E, F, U, H, C, P, N$.

The finite-window and regular-name obligations are ordered by the carrier. A completion-facing observation first exposes the window row S, then reads the regular rational row R and the dyadic ledger D before reaching the terminal Real seal E.

The filter and universal-property rows are downstream of the same route. The row F uses the displayed windows and dyadic budgets as its filter-completion handoff, while U is read only after the regular-name, Real-seal, and filter-completion rows have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since quotient stream equality, selected representatives, countable choice, host real equality, and ambient completeness have no carrier row, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionCarrier namecert obligations`

Theorem 2749.3 (Bishop completion regular real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier reads the completion boundary only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In the notation of Definition 3717.1, the consumer reads S, then R and D, and only then the seal row E together with the structural rows H, C, P, N. The route exports no quotient stream, selected host limit, countable-choice schedule, host real equality, or ambient Real completeness datum.

Proof. Begin with a Real-facing read of $B = (R, S, D, E, F, U, H, C, P, N)$. The carrier has a single Real-seal coordinate, namely E, and E is placed after the window, regular-readback, and dyadic-ledger rows.

The window row S supplies the finite observations available to the read. The row R records the regular rational name extracted from those observations, and D records the dyadic tolerance ledger for the same precision requests.

The structural rows preserve this order. The hsame row H transports the displayed coordinates componentwise, the Cont row C replays the same route, and P, N record provenance and local naming without adding analytic data. Thus every Real-facing read remains inside the listed finite route. □

Theorem 2749.4 (Bishop completion filter compatibility)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier must read the same displayed regular-name boundary S, R, D, E before it can read the filter handoff F or the universal-property row U. Consequently sequence-style and filter-style completion reads agree on the finite BEDC support*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The compatibility exports no quotient filter, selected limit, countable choice, host equality proof, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionCarrier filter compatibility`

Proof. Project the filter and universal rows from Definition 3717.1. The row F is downstream of the same stream, regular-readback, dyadic-ledger, and Real-seal route used by the regular-name boundary; the row U is downstream of both that boundary and F.

A filter-style read therefore cannot start from an independent filter representative. It first sees the displayed finite windows S, the regular rational readback R, and the dyadic tolerance ledger D, then reaches the Real seal E and only afterward the filter-completion handoff F.

The universal-property read is even later in the route. It is exposed through U after S, R, D, E, F have been displayed and after H, C, P, N have fixed the componentwise transport, replay, provenance, and local naming rows.

Since the carrier includes no quotient filter, selected limit, countable-choice selector, host equality proof, or ambient completeness coordinate, compatibility between the sequence-style and filter-style reads is exactly the shared finite support listed in the statement. □

Falsifiable prediction: Every $\mathsf{BishopCompletion}^{\uparrow}$ Real-facing or filter-facing read factors through displayed StreamName, RegSeqRat, DyadicRatCore, RealUp, CauchyFilterCompletion, CompletionUniversalProperty, transport, replay, provenance, and local naming rows.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchyfiltercompletion, completionuniversalproperty: $\mathsf{BishopCompletion}^{\uparrow}$ is not bijective to any listed sibling because it carries the regular-name route, Real seal, filter-completion handoff, and universal-property row in one finite completion boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopCompletion}\uparrow}$, seed)

Scope closed: Definition 3717.1, Definition 3717.2, Definition 3717.3, and Definition 3717.4 seed the finite Bishop completion boundary over regular Cauchy names, stream windows, dyadic ledgers, Real seals, filter completion, and universal-property rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, quotient filters, selected limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular Real boundary exposure, filter compatibility, universal-property exposure, and ledger non-escape for BEDC.Derived.BishopCompletionUp.

Constructive story: A BishopCompletion$^{\uparrow}$ packet is generated from a StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, CauchyFilterCompletion handoff, CompletionUniversalProperty row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2750

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational stream exposes a modulus schedule before a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the name. It sits over the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ tail-bound route of chapter 1659, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate makes the modulus extraction visible as finite NameCert data rather than as a prose appeal to a completed Cauchy object.

Definition 2750.1 (Regular Cauchy modulus extraction carrier)**.** *A* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the regular Cauchy source and by each downstream precision request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding displayed stream windows to rational observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii, comparison cells, half-radius splits, and endpoint observations.*
4. *T is the* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *row, displaying the tail bound that makes the requested dyadic scale stable after a finite stage.*
5. *E is the extracted modulus schedule, assigning each displayed tolerance request to the finite stage justified by S, R, D, T.*
6. *H records componentwise* hsame *transport for the window, readback, tolerance, tail-envelope, schedule, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, R, D, T \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, T, E, H, C, P, N$. It has no coordinate for selected limits, quotient equality of streams, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a modulus schedule chosen outside the displayed tail-envelope route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusExtractionUp.RegularCauchyModulusExtractionTasteGate single carrier alignment`

Theorem 2750.2 (Regular Cauchy modulus extraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (S, R, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, DyadicCauchyTailEnvelope bound, extracted schedule, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *readback exactness, requiring R to read only rational observations displayed in S and tolerance data displayed in D;*
3. *tail-bound exposure, requiring T to be the only row that certifies a stable tail after a finite stage for the requested dyadic scale;*
4. *modulus extraction exactness, requiring E to be computed from the displayed route S, R, D, T and from no detached selector;*
5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the same displayed route.*

No $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 3718.1. The accepted packet displays exactly the window, readback, tolerance, tail-envelope, extracted schedule, transport, replay, provenance, and local naming rows.

Readback exactness is controlled by S, R, D. The regular rational observations are available only through displayed stream windows and displayed dyadic tolerance cells.

Tail-bound exposure is the role of T. It records the finite tail envelope attached to the same dyadic scale, so the packet has no independent convergence oracle.

The extracted schedule E is downstream of S, R, D, T. A consumer can therefore read a modulus only as the displayed finite schedule justified by the tail envelope.

Finally, H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no remaining coordinate stores a selected limit, quotient equality, outside schedule, trichotomy, or ambient Real-completeness principle, the listed rows exhaust the local certificate surface. □

Theorem 2750.3 (Regular Cauchy modulus extraction tail-bound route)**.** *Let*

$$M = (S, R, D, T, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer that reads a modulus from M must factor through the finite tail-bound route*

$$S, R, D \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed StreamName window row, RegSeqRat rational readback, DyadicRatCore tolerance ledger, DyadicCauchyTailEnvelope bound, extracted schedule, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data, and receives no quotient stream equality, selected limit, trichotomy, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

Proof. Begin with an accepted carrier and project the rows $S, R, D, T, E, H, C, P, N$. The only observation surface before a modulus is read is the displayed window, readback, and dyadic tolerance route.

The tail-bound row T is then read at the same dyadic scale. It supplies the finite stage after which the displayed rational tail is stable for the request.

The extracted schedule E is downstream of that tail-bound row. Thus any modulus read by a Real or completion consumer is the schedule justified by S, R, D, T.

The structural rows H, C, P, N transport, replay, source, and name the same route. Because the carrier has no coordinate for the excluded equality, selection, order, choice, completeness, or bridge objects, every such consumer factors through the displayed tail-bound route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ packet is generated from a StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance row, DyadicCauchyTailEnvelope bound row, extracted modulus schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3718.1, Definition 3718.2, and Definition 3718.3 seed the finite modulus-extraction route from regular Cauchy readbacks through dyadic tail envelopes and displayed schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, readback exactness, tail-bound exposure, modulus extraction exactness, and structural stability for BEDC.Derived.RegularCauchyModulusExtractionUp.

Chapter 2751

A Concrete Naming Certificate for $\mathsf{CauchySelectionFunction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySelectionFunction}^{\uparrow}$ packet records the finite selector interface by which a Real and Completion consumer asks for a displayed regular Cauchy window without importing countable choice. It sits after the Cauchy modulus surface of chapter 292, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the displayed modulus-selection route of chapter 1986, and the selector-budget packet of chapter 678. It provides the finite selector row used by the Real completeness agenda of chapter 9238; it is not a countable-choice principle, quotient stream object, selected global limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2751.1 (Cauchy selection-function carrier)**.** *A* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the request row, naming the precision request, source regular Cauchy name, and completion-facing consumer boundary.*
2. *M is the modulus schedule row, read through the* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *surfaces for the displayed request.*
3. *W is the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window opened by M for this request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row assembled from the selected window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the dyadic radius, slack, comparison, and scale data used by the readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, M, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the request, schedule, selected window, regular readback, dyadic tolerance ledger, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the selector route*

$$Q, M \longmapsto W \longmapsto R, D \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchySelectionFunction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusSelection}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, M, W, R, D, E, H, C, P, N$. It has no coordinate for countable choice, quotient streams, selected global limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySelectionFunctionUp.CauchySelectionFunctionTasteGate single carrier alignment`

Theorem 2751.2 (Cauchy selection-function NameCert obligations)**.** *For every accepted* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier*

$$S = (Q, M, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the request row Q, modulus schedule M, selected finite window W, regular readback R, dyadic tolerance ledger D, terminal real seal E, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

The obligation surface also includes request-schedule compatibility, finite-window visibility, regular-readback exactness, dyadic tolerance compatibility, real-seal nonescape, and stability under the displayed hsame *and* Cont *rows. No obligation is stored outside $Q, M, W, R, D, E, H, C, P, N$.*

Proof. Carrier admission is projection from Definition 3719.1. The projection exposes exactly the request, schedule, selected-window, readback, dyadic, seal, transport, replay, provenance, and naming rows listed in the statement.

Request-schedule compatibility is the role of Q and M. The schedule is read as a displayed modulus-selection row for the request, and the selected finite window W is downstream of that schedule rather than a separate choice function.

Regular readback and dyadic tolerance exactness are the roles of R and D. The carrier places both rows after W, so the readback is assembled only from displayed observations and the displayed dyadic ledger.

The terminal real seal E is downstream of Q, M, W, R, D, while H, C, P, N transport, replay, source, and name the same finite route. Since no further coordinates occur in the packet, countable choice, quotient streams, selected global limits, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be exported by the local certificate. □

Theorem 2751.3 (Cauchy selection-function regular readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier*

$$S = (Q, M, W, R, D, E, H, C, P, N)$$

must first read the displayed request row Q, the modulus schedule M, the selected finite window W, and the dyadic tolerance ledger D. The regular readback row R and real seal row E then expose only the finite selector route

$$Q, M \longmapsto W, D \longmapsto R \longmapsto E$$

together with the structural rows H, C, P, N.

Thus the selector interface supplies a finite BEDC service for Real and Completion consumers without providing countable choice, quotient stream equality, a selected global limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read. By the carrier definition, the readback rows available to that consumer are R and E, and both are downstream of the request row, schedule row, selected window, and dyadic ledger.

The selected window is finite because it is the displayed $\mathsf{StreamName}^{\uparrow}$ row W. The schedule M only names the request-facing modulus surface that opens W; it does not add a second stream, a quotient stream, or an infinite family of choices.

The dyadic ledger D records the tolerance data used by the same finite window, and R assembles the regular rational readback from those displayed rows. The real seal E can therefore be consumed only as the terminal row of the route through Q, M, W, D, R.

Finally, H, C, P, N preserve and name the route under componentwise transport and continuation replay. The excluded objects have no carrier coordinate, so a consumer cannot read countable choice, quotient stream equality, selected global limits, host real equality, or ambient completeness from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySelectionFunction}^{\uparrow}$ packet is generated from a request row, a displayed modulus schedule, a selected finite StreamName window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3719.1, Definition 3719.2, and Definition 3719.3 seed the finite selector interface for Real and Completion consumers over displayed CauchyModulus, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient streams, selected global limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-schedule compatibility, selected-window visibility, regular readback, dyadic tolerance compatibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySelectionFunctionUp.

Chapter 2752

A Concrete Naming Certificate for $\mathsf{CauchySumLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySumLimit}^{\uparrow}$ packet records the finite route by which two displayed regular Cauchy summands produce a regular Cauchy sum and then a terminal real-name seal. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name boundary of chapter 38. The chapter packages the familiar algebraic limit route for sums as NameCert data rather than as quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2752.1 (CauchySumLimit carrier)**.** *A* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy summand rows, each exposing only its displayed finite regularity surface.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite read windows opened for both summands and for the sum-read request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the rational readbacks of the two summands and the pointwise sum readback obtained from the same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, splitting every requested sum tolerance into displayed summand tolerances and recording the dyadic addition comparisons used by the readback row.*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the sum, available only after S, Q, D have been displayed.*
6. *H records componentwise* hsame *transport for the summand rows, stream windows, regular readbacks, dyadic tolerance ledger, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from any Cauchy-sum-limit request through X, Y, S, Q, D, A.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySumLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, Q, D, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchySumLimitUp.CauchySumLimitTasteGate single carrier alignment toEventFlow`

Theorem 2752.2 (CauchySumLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier*

$$L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy summands, finite stream-window display, regular rational sum readback, dyadic tolerance splitting, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from quotient streams, selected limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3720.1. The only summand data are X and Y, and the only observation windows available to consumers are the finite windows recorded by S.

The readback stage is finite. The row Q records the rational observations of both summands and the pointwise sum readback obtained from those same windows, while D records the dyadic tolerance split and the addition comparisons that justify the requested sum tolerance.

The seal stage is terminal. The row A can be consumed only after the summand windows, readbacks, and tolerance ledger have been displayed, so a Real-facing consumer cannot bypass S, Q, D.

Finally H, C, P, N transport, replay, source, and name exactly the displayed route. Since the packet has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.CauchySumLimitUp.CauchySumLimitNamecertObligations`

Theorem 2752.3 (CauchySumLimit Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y \longmapsto S \longmapsto Q, D \longmapsto A, H, C, P, N.$$

Thus the sum limit is presented as a displayed regular-Cauchy algebra route to a Real seal, not as a quotient stream, selected limit, countable-choice schedule, trichotomy principle, or ambient completeness theorem.

Proof. Begin with an accepted packet $L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N)$. A consumer first reads the displayed summand rows X and Y; no host sequence equality or quotient representative is available at this stage.

Next the stream windows in S determine the finite observations used for the request. The row Q reads both summands on those windows and records the pointwise rational sum, while D supplies the dyadic tolerance split and the corresponding addition comparisons.

The terminal Real seal is A. Since the continuation row C replays only the route through X, Y, S, Q, D, A, every Real-facing read reaches A after the finite regular readback and tolerance ledger have been displayed.

The transport and provenance rows preserve the same route under componentwise hsame. The local naming row N therefore exports the sum-limit seal without adding quotient streams, selected limits, countable choice, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3720.1, Definition 3720.2, and Definition 3720.3 seed the finite route from two regular Cauchy summands to StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, and a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, summand-window display, regular rational sum readback, dyadic tolerance splitting, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchySumLimitUp.

Constructive story: A $\mathsf{CauchySumLimit}^{\uparrow}$ packet is generated from two regular Cauchy summand rows, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2753

A Concrete Naming Certificate for $\mathsf{CauchyProductLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyProductLimit}^{\uparrow}$ packet records the finite route by which two displayed regular Cauchy factor rows produce a regular Cauchy product and then a terminal real-name seal. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic product-tolerance ledger of chapter 291, the product descent surface of chapter 461, and the real-name boundary of chapter 38. The chapter is the product analogue of chapter 3720: it packages a finite regular-Cauchy algebra route to a Real seal rather than quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2753.1 (CauchyProductLimit carrier)**.** *A* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy factor rows, each exposing only its displayed finite regularity surface.*
2. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite read windows opened for both factors and for the product-read request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the rational readbacks of both factors and the pointwise product readback obtained from the same shared window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product tolerance ledger, recording factor bounds, product-error splitting, dyadic multiplication comparisons, and the tolerance entries consumed by the product readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent row. It can be read only after X, Y, S, Q, D have been displayed.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product, available only after the regular product descent has reached the displayed readback boundary.*
7. *H records componentwise* hsame *transport for the factor rows, shared windows, regular readbacks, dyadic product ledger, product descent row, Real seal, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from any product-limit request through X, Y, S, Q, D, M, A.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, Q, D, M, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductLimitUp.CauchyProductLimitTasteGate single carrier alignment`

Theorem 2753.2 (CauchyProductLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier*

$$L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy factors, shared finite stream-window display, regular rational product readback, dyadic product-tolerance accounting,* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3721.1. The only factor data are X and Y, and the only observation windows available to consumers are the finite shared windows recorded by S.

The readback stage is finite. The row Q records the rational observations of both factors and the pointwise product readback obtained from those same windows, while D records the dyadic factor bounds, product-error split, and multiplication comparisons that justify the requested product tolerance.

The descent stage is forced by row order. The row M is downstream of X, Y, S, Q, D, so a product-limit consumer cannot read $\mathsf{RegularCauchyProduct}^{\uparrow}$ evidence before the factor windows, product readback, and dyadic ledger have been displayed.

Finally A, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since the packet has no coordinate for quotient stream equality, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Theorem 2753.3 (CauchyProductLimit Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y \longmapsto S \longmapsto Q, D \longmapsto M \longmapsto A, H, C, P, N.$$

Thus the product limit is presented as a displayed regular-Cauchy algebra route to a Real seal, not as a quotient stream, selected product limit, countable-choice schedule, trichotomy principle, host multiplication of completed reals, or ambient completeness theorem.

Proof. Begin with an accepted packet $L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N)$. A consumer first reads the displayed factor rows X and Y; no host sequence equality, quotient representative, or completed-real multiplication primitive is available at this stage.

Next the shared stream windows in S determine the finite observations used for the request. The row Q reads both factors on those windows and records the pointwise rational product, while D supplies the dyadic factor bounds and product-tolerance ledger.

The regular product descent is M. Since M is downstream of the factor windows, readbacks, and tolerance ledger, the terminal Real seal A is reached only after the finite product route has already been displayed.

The transport and provenance rows preserve the same route under componentwise hsame. The local naming row N therefore exports the product-limit seal without adding quotient streams, selected limits, countable choice, trichotomy, host multiplication of completed reals, or ambient Real completeness. □

Theorem 2753.4 (CauchyProductLimit shared-window stability)**.** *Let*

$$L_\Pi = (X, Y, S, Q, D, M, A, H, C, P, N)$$

and

$$L'_\Pi = (X', Y', S', Q', D', M', A', H', C', P', N')$$

be accepted $\mathsf{CauchyProductLimit}^\uparrow$ *carriers. If the two packets preserve the two factor rows, the shared* $\mathsf{StreamName}^\uparrow$ *window, the* $\mathsf{RegSeqRat}^\uparrow$ *readback, the* $\mathsf{DyadicRatCore}^\uparrow$ *product ledger, the* $\mathsf{RegularCauchyProduct}^\uparrow$ *descent row, and the terminal* $\mathsf{Real}^\uparrow$ *seal componentwise under* hsame*, then every product-limit consumer reads the same finite route through* $S, Q, D, M, A, H, C, P, N$*. In particular the read is stable under the displayed product-limit route and cannot be replaced by quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the two carrier displays from Definition 3721.1. The admissible coordinates of a product-limit read are exactly the factor rows, the shared window row, the regular rational readback, the dyadic product ledger, the product descent row, the Real seal, the componentwise transport, replay, provenance, and local naming rows.

Apply Definition 3721.2 to both packets. Carrier admission binds every accepted read to the displayed rows, while the shared-window, readback, dyadic-ledger, descent, and seal clauses rule out any extra route not present in L_Π or L'_Π.

Use the componentwise hsame hypotheses next. They identify the two factor rows, S with S', Q with Q', D with D', M with M', and A with A' at the exact sites where a consumer can inspect the product-limit packet.

Finally apply the Real-seal route theorem Definition 3721.3. Since the terminal read must pass through the finite route from X, Y to $S, Q, D, M, A, H, C, P, N$, the preserved components force the same product-limit route and keep quotient streams, selected product limits, countable choice, trichotomy, host multiplication, and ambient Real completeness outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^\uparrow}$, seed)

Scope closed: Definition 3721.1, Definition 3721.2, and Definition 3721.3 seed the finite route from two regular Cauchy factor rows to shared StreamName windows, RegSeqRat readbacks, a DyadicRatCore product ledger, RegularCauchyProduct descent, and a RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window display, regular rational product readback, dyadic product-tolerance accounting, RegularCauchyProduct descent, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyProductLimitUp.

Constructive story: A $\mathsf{CauchyProductLimit}^\uparrow$ packet is generated from two regular Cauchy factor rows, shared finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore product-tolerance ledger, a RegularCauchyProduct descent row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2754

A Concrete Naming Certificate for $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ packet records the finite diagonal selector route used by the real-completeness corridor of chapter 9238. It concretizes the transported stationary diagonal theorem of Definition 9238.17 as a NameCert packet over the no-choice selector boundary of chapter 695, the cofinal tail route of chapter 697, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The selector is a displayed Bishop-style regular-Cauchy diagonal construction, not a host choice function, quotient stream, selected completed limit, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness primitive.

Definition 2754.1 (RealCompletenessDiagonalSelector carrier)**.** *A* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, E, T, S, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular Cauchy source-family row whose finite observations are available through the displayed real-completeness route.*
2. *E is the* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *row, recording the explicit precision request, selected finite window, and refusal of a host choice function.*
3. *T is the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *row, recording the cofinal tail index and the finite tail-window handoff consumed by the diagonal route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row shared by the selector, tail, and readback requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected diagonal windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the precision budget, selected-window comparisons, and tail-cofinality tolerance entries.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the selected diagonal representative, available only after E, T, S, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for the source-family, selector, tail, window, readback, dyadic ledger, Real seal, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from any diagonal-selector request through R, E, T, S, Q, D, A.*
10. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$, $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, E, T, S, Q, D, A, H, C, P, N$. It has no coordinate for host choice, quotient stream equality, selected completed limits, hidden tail oracles, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCompletenessDiagonalSelectorUp.RealCompletenessDiagonalSelectorTasteGate`
`single carrier alignment`

Theorem 2754.2 (RealCompletenessDiagonalSelector NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier*

$$G = (R, E, T, S, Q, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular Cauchy source family, choice-free finite selector admission, cofinal diagonal-tail routing, shared StreamName window display, RegSeqRat selected-window readback, DyadicRatCore precision and tail-cofinality ledgers, terminal RealUp seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3722.1. The source-family row R is only readable through the finite selector and tail rows E and T, so no host family choice or completed-limit selector is available before the displayed rows have been opened.

The finite selector stage is explicit. The row E records the precision request and selected window without a host choice function, while T records the cofinal tail handoff used by the same request.

The observation stage is also finite. The shared window row S, readback row Q, and dyadic ledger D name the StreamName, RegSeqRat, and DyadicRatCore data consumed by the diagonal representative before the Real seal A can be read.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no coordinate for host choice, quotient stream equality, selected completed limits, hidden tail oracles, trichotomy, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Theorem 2754.3 (RealCompletenessDiagonalSelector non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or real-completeness-facing consumer of an accepted* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto E, T \longmapsto S, Q, D \longmapsto A, H, C, P, N.$$

The transported stationary diagonal row of Definition 9238.17 is therefore consumed as finite selector, tail, window, readback, dyadic, seal, transport, replay, provenance, and local naming data, not as host choice, quotient stream equality, selected completed limit, hidden tail oracle, trichotomy, or ambient completeness.

Proof. Start with an accepted packet $G = (R, E, T, S, Q, D, A, H, C, P, N)$. The source-family row R is not a completed real carrier; it becomes visible to a consumer only through the displayed selector and tail rows.

The selector rows E and T fix the finite observation request. The former records the no-choice selection boundary, and the latter records the cofinal tail route that serves the same diagonal window.

The rows S, Q, D then supply the finite StreamName window, RegSeqRat readback, and DyadicRatCore tolerance ledger. The Real seal A is downstream of these rows, so a consumer cannot bypass them to obtain an independent RealUp limit.

Transport and replay preserve the row order under componentwise hsame. The local naming row N therefore exports only the displayed diagonal-selector route and cannot export host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealCompletenessDiagonalSelectorUp.RealCompletenessDiagonalSelectorNonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3722.1, Definition 3722.2, and Definition 3722.3 seed the finite Bishop-style diagonal selector route from regular Cauchy source rows through ChoiceFreeDiagonalSelector and DiagonalTailSelector rows to StreamName windows, RegSeqRat readbacks, a DyadicRatCore ledger, and a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, choice-free selector admission, cofinal tail routing, selected-window readback, dyadic precision and tail-cofinality ledgers, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.RealCompletenessDiagonalSelectorUp.

Constructive story: A $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ packet is generated from a regular Cauchy source-family row, a ChoiceFreeDiagonalSelector row, a DiagonalTailSelector row, shared StreamName windows, RegSeqRat selected-window readbacks, a DyadicRatCore precision ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2755

A Concrete Naming Certificate for $\mathsf{LeibnizSeriesTest}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of alternating series, the $\mathsf{LeibnizSeriesTest}^{\uparrow}$ packet records the classical alternating-series tail-bound route as finite BEDC data before any completion-facing consumer may read a regular rational handoff or a real seal. It refines the convergence-test surface of chapter 1483, uses the regular rational readback of chapter 301, opens only the finite observation windows of chapter 126, measures tail tolerances through the dyadic core of chapter 291, and terminates at the real-name boundary of chapter 38. The packet names the finite Leibniz tail-bound route; it does not import selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2755.1 (LeibnizSeriesTest carrier)**.** *A* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, listing the finite partial-sum and summand windows opened by the request.*
2. *Σ is the finite alternating-sign row, recording the parity and sign entries visible on the displayed windows.*
3. *M is the monotone magnitude row, recording nonnegative dyadic magnitude reads and their finite monotonicity comparisons on the inspected tail.*
4. *T is the tail-bound schedule, assigning each requested dyadic tolerance to a displayed tail index whose next visible magnitude is within that tolerance.*
5. *Q is the* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *handoff row, accepting the source, alternating-sign, monotone-magnitude, and tail-bound rows as the local convergence-test face.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed partial-sum stream after the handoff row has been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, Σ, M, T, Q, R have been read.*
8. *H records componentwise* hsame *transport for the source, sign, magnitude, tail-bound, handoff, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any Leibniz-series-test request through* S, Σ, M, T, Q, R, E.
10. *P is the* Pkg *provenance over* $\mathsf{LeibnizSeriesTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, \Sigma, M, T, Q, R, E, H, C, P, N$. It has no coordinate for selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LeibnizSeriesTestUp.LeibnizSeriesTestTasteGate single carrier alignment`

Theorem 2755.2 (LeibnizSeriesTest NameCert obligations)**.** *For every accepted* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier*

$$L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$ *surface consists exactly of finite source-window admission, alternating-sign display, monotone magnitude display, dyadic tail-bound scheduling,* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from selected sums, quotient series equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LeibnizSeriesTestUp.LeibnizSeriesTestNameCertObligations`

Proof. Project the carrier of Definition 3723.1. The only source observations are the finite windows in S, and the only sign information is the displayed parity ledger Σ.

The tail estimate is finite. The row M gives the visible monotone magnitude comparisons, while T records the tail index chosen for each requested dyadic tolerance. The handoff row Q then accepts exactly those source, sign, magnitude, and tail-bound rows through the $\mathsf{AlternatingSeriesTest}^{\uparrow}$ surface.

The completion-facing route is also finite. The regular readback R may be read only after Q, and the terminal seal E receives only R together with the displayed source, sign, magnitude, and tail-bound route.

Finally H, C, P, N transport, replay, source, and name that same route. Since no carrier coordinate stores a selected sum, quotient series equality, countable-choice schedule, trichotomy premise, or ambient completeness theorem, none of those objects can be exported by the local certificate. □

Theorem 2755.3 (LeibnizSeriesTest Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by an accepted* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier factors through the finite route*

$$S, \Sigma, M, T \longmapsto Q \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the Leibniz tail-bound route reaches the terminal Real seal only through displayed alternating signs, monotone magnitudes, dyadic tail bounds, the alternating-series-test handoff, and regular rational readback; it does not expose selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted packet $L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N)$. A Real-facing consumer must first open the displayed source windows S, because no partial-sum observation is available outside those windows.

The sign and magnitude checks are then forced by Σ and M. The tail-bound schedule T is the only row that converts a requested dyadic tolerance into a displayed tail index, and the handoff row Q records that those finite rows enter the alternating-series-test surface.

The regular rational handoff is R, and the Real seal is E. Since E is available only after R and the finite handoff route have been displayed, any seal-facing read stays inside the named route.

The transport, replay, provenance, and local naming rows preserve that route componentwise. Hence the exported boundary cannot contain selected sums, quotient series equality, countable choice, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3723.1, Definition 3723.2, and Definition 3723.3 seed the finite Leibniz tail-bound route from displayed alternating signs and monotone magnitudes through RegSeqRat readback to a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign and magnitude display, tail-bound scheduling, alternating-series-test handoff, regular rational readback, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.LeibnizSeriesTestUp.

Constructive story: A $\mathsf{LeibnizSeriesTest}^{\uparrow}$ packet is generated from finite StreamName windows, an alternating-sign row, monotone magnitude row, dyadic tail-bound schedule, AlternatingSeriesTestUp handoff, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2756

A Concrete Naming Certificate for $\mathsf{LipschitzModulus}^{\uparrow}$

Following Bishop–Bridges constructive metric analysis and Kelley-style uniform-space bookkeeping, the $\mathsf{LipschitzModulus}^{\uparrow}$ packet records the finite tolerance object carried by a displayed Lipschitz bound. It sits over the source and target metric rows of chapter 57, the real-bound seal of chapter 38, the Lipschitz map and constant surfaces of chapter 410 and chapter 2328, the division and rounding ledger of chapter 291, and the exported modulus and continuity consumers of chapter 369 and chapter 391. The packet names a finite tolerance-only route; it is not a host proof of continuity, a quotient of maps, a selected inverse for a real bound, or an ambient completeness theorem.

Definition 2756.1 (LipschitzModulus carrier)**.** *A* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, F, K, E, D, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *F is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row whose graph and source-distance reads use X and Y.*
3. *K is the* $\mathsf{LipschitzConstant}^{\uparrow}$ *row for the same source, target, and graph data.*
4. *E is the positive target-tolerance row, read through* $\mathsf{Real}^{\uparrow}$ *rather than through a host scalar.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division and rounding ledger that turns E and the displayed bound in K into a source-tolerance request.*
6. *U is the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row consumed by* $\mathsf{UniformContinuity}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *routes.*
7. *H records componentwise* hsame *transport for metric rows, map rows, constant rows, tolerance rows, dyadic division and rounding rows, the exported modulus, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a target-tolerance request through E, K, D to the source-tolerance row exported by U.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for an unbounded host Lipschitz proof, a quotient-function equality, a hidden choice of reciprocal, a completed function space, or an ambient real-completeness argument.

Lean4 ✓ `BEDC.Derived.LipschitzModulusUp.LipschitzModulusTasteGate single carrier alignment`

Theorem 2756.2 (LipschitzModulus NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier* $M = (X, Y, F, K, E, D, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the source metric, target metric, Lipschitz map, Lipschitz constant, positive tolerance, dyadic division and rounding, exported modulus, transport, replay, provenance, and local naming rows;*
2. *bound alignment, requiring F and K to read the same source, target, and graph rows before a tolerance conversion is exported;*
3. *tolerance positivity, requiring E to be displayed before the dyadic ledger is read;*
4. *modulus export, requiring every* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer to pass through E, K, D, U in that order;*
5. *ledger non-escape, excluding host Lipschitz proofs, quotient map equality, hidden reciprocal choice, completed function spaces, selected limits, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 3724.1. The displayed packet exposes exactly the metric, Lipschitz-map, Lipschitz-constant, positive-tolerance, dyadic-ledger, modulus, transport, replay, provenance, and naming rows.

The bound-alignment and positivity obligations are row-order checks. The map row F and constant row K share the source and target metric rows, and the tolerance row E is the only positive target request available to the dyadic ledger D.

The export obligation follows because U is the only $\mathsf{UniformModulus}^{\uparrow}$ row in the carrier. A continuity consumer therefore reaches U only after reading the displayed tolerance, constant, and dyadic division and rounding route.

The refusal clause is an inventory check on the carrier. None of the displayed rows supplies host continuity, quotient map equality, hidden reciprocal selection, completed function-space data, selected limits, choice, or ambient completeness. □

Theorem 2756.3 (LipschitzModulus uniform-continuity handoff)**.** *Let* $M = (X, Y, F, K, E, D, U, H, C, P, N)$ *be an accepted* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier. Every* $\mathsf{UniformContinuity}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer that requests a target tolerance receives only the source-tolerance route exported by U after the packet has displayed the target tolerance E, the aligned Lipschitz constant K, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division and rounding ledger D. Thus the handoff is a finite modulus route, not a rereading of a host Lipschitz proof.*

Proof. Start from the carrier of Definition 3724.1. The source and target metrics are X and Y, the graph and bound rows are F and K, and the only target-tolerance request is E.

The dyadic ledger D performs the finite division and rounding step from the displayed tolerance and bound data. Since D is the only arithmetic route in the packet, no consumer can replace it by a host reciprocal or by a selected real approximation.

The exported row U is then read by the $\mathsf{UniformModulus}^{\uparrow}$ surface and by downstream $\mathsf{UniformContinuity}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ consumers. Transport through H and replay through C preserve this order, so the consumer receives only the displayed finite modulus route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3724.1, Definition 3724.2, and Definition 3724.3 seed the finite Lipschitz tolerance route from displayed metric, Lipschitz-map, Lipschitz-constant, positive-tolerance, dyadic division and rounding, and uniform-modulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient function spaces, host continuity proofs, reciprocal choice, completed Sobolev or function-space carriers, selected limits, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, bound alignment, tolerance positivity, dyadic division and rounding exactness, uniform-modulus export, transport stability, and replay stability for BEDC.Derived.LipschitzModulusUp.

Constructive story: A $\mathsf{LipschitzModulus}^{\uparrow}$ packet is generated from source and target metric rows, Lipschitz map and constant rows, a positive tolerance row, a DyadicRatCore division and rounding ledger, an exported UniformModulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 2757

A Concrete Naming Certificate for $\mathsf{ModulusCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ packet records the finite completion endpoint surface obtained when a displayed Cauchy modulus is bound to regular rational readback and a terminal real seal. It sits over the modulus interface of chapter 292, the regular rational surface of chapter 301, the finite stream-window route of chapter 126, the dyadic tolerance ledger of chapter 291, the Bishop regular Cauchy completion packet of chapter 1639, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a modulus-driven endpoint handoff; it does not choose a quotient representative, assert ambient completeness, or select limits by countable choice.

Definition 2757.1 (Modulus Cauchy completion carrier)**.** *A* $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, carrying the finite precision-to-window schedule used by the completion endpoint request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the finite source windows admitted by the modulus schedule.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the displayed stream windows as regular rational Cauchy data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic thresholds, comparison rows, and endpoint error budgets used by M, S, R.*
5. *E is the endpoint seal row, read through the Bishop regular completion boundary and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
6. *H stores componentwise* hsame *transport for the modulus, stream, readback, dyadic ledger, endpoint seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the modulus schedule to source windows, from source windows to regular readback, from the dyadic ledger to endpoint tolerances, and from the regular completion handoff to the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *Bishop regular Cauchy completion,* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, S, R, D, E, H, C, P, N$. It contains no coordinate for quotient completion equality, selected host limits, countable choice, ambient real completeness, or a bridge-checked standard endpoint.

Lean4 ✓ `BEDC.Derived.ModulusCauchyCompletionUp.ModulusCauchyCompletionTasteGate single carrier alignment`

Theorem 2757.2 (Modulus Cauchy completion NameCert obligations)**.** *For every accepted* ModulusCauchyComple *carrier $Q = (M, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed modulus, stream-window, regular readback, dyadic tolerance, endpoint seal, transport, replay, provenance, and local naming rows;*
2. *modulus compatibility, requiring every endpoint request to read its threshold and observation window through the same* $\mathsf{CauchyModulus}^{\uparrow}$ *row M and* $\mathsf{StreamName}^{\uparrow}$ *row S;*
3. *regular-readback stability, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be transported componentwise from the displayed stream windows and dyadic tolerance rows;*
4. *endpoint handoff, requiring the endpoint row E to be exposed only after the Bishop regular completion boundary and terminal* $\mathsf{Real}^{\uparrow}$ *seal have read M, S, R, D;*
5. *ledger nonescape, excluding quotient completion equality, host real equality, selected limits, countable choice, ambient completeness, and bridge checking from the local carrier.*

Proof. Project the carrier of Definition 3725.1. The rows $M, S, R, D, E, H, C, P, N$ are the full accepted inventory, so carrier admission and classifier comparison are finite.

The modulus and stream rows are read together. A precision request reaches an observation window only through M and S, and the dyadic row D records the tolerance ledger attached to that same read.

The regular readback and endpoint handoff are downstream. The row R is obtained from the displayed windows and tolerance data, while E is exposed only after the Bishop regular completion boundary and the $\mathsf{Real}^{\uparrow}$ seal consume the same rows.

Finally inspect the missing coordinates. Since the carrier has no quotient equality, host real equality, selected limit, countable-choice, ambient-completeness, or bridge-checking row, the local NameCert surface cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ packet is generated from a CauchyModulus row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an endpoint Real/completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3725.1 and Definition 3725.2 seed the finite modulus-to-endpoint packet over completion-facing regular rational Cauchy data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient completion equality, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus compatibility, regular-readback stability, endpoint handoff, and ledger nonescape for the same finite packet.

Chapter 2758

A Concrete Naming Certificate for $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ packet records the finite-cover substrate used by located compact-uniform-continuity arguments on closed bounded interval data. It refines the closed interval source of chapter 1591, the finite ball-cover carrier of chapter 2377, the positive-radius ledger of chapter 3606, the compact uniform-continuity route of chapter 2292, and the metric-ball observation surface of chapter 1450. The packet names only finite located cover rows, local membership witnesses, Lebesgue-number handoff data, transport, replay, provenance, and local naming evidence.

Definition 2758.1 (Uniformly located finite cover carrier)**.** *A* $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including endpoint order, rational and dyadic containment ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *F is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row whose ball cells, centers, dyadic radii, coverage witnesses, and refinement ledger cover the inspected interval windows.*
3. *L is the* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *ledger row that records the positive-radius handoff for the displayed finite cover.*
4. *M is the metric-ball and local locatedness membership row: each inspected interval window is paired with an admitted* $\mathsf{MetricBall}^{\uparrow}$ *cell and a finite locatedness witness for membership in that cell.*
5. *H records componentwise* hsame *transport for I, F, L, M, C, P, N and for the endpoint, cover, radius, ball, membership, and local naming coordinates exposed by those rows.*
6. *C records displayed* Cont *replay from interval containment rows to finite cover coverage checks, from coverage checks to local locatedness membership, and from the cover ledger to the Lebesgue-number handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, a choice-selected subcover, a host-set membership oracle, quotient real equality, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformlyLocatedFiniteCoverUp.UniformlyLocatedFiniteCoverTasteGate single carrier alignment`

Theorem 2758.2 (Uniformly located finite cover NameCert obligations)**.** *For an accepted* UniformlyLocatedFiniteC *carrier $U = (I, F, L, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple; finite cover coverage through F over the interval source I; local locatedness preservation through the metric-ball membership row M; Lebesgue-number handoff through L; componentwise transport stability through H; displayed replay through C; provenance through P; local naming through N; and ledger nonescape from open-cover compactness, selected subcovers, host membership oracles, quotient real equality, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 3726.1. The displayed tuple exposes the closed bounded interval source, finite cover row, positive-radius ledger, local metric-ball membership row, and the structural rows H, C, P, N.

Coverage is read only through F after I has supplied its endpoint, containment, window, readback, and Real-seal data. Locatedness preservation is read through M, where each finite cell membership remains attached to an admitted $\mathsf{MetricBall}^{\uparrow}$ row and to the same interval window.

The Lebesgue-number handoff is downstream of the displayed finite cover: L may be consumed only after the cover row and local membership ledger have been replayed. The transport and replay rows then preserve the same interval-cover-membership route componentwise.

No excluded object is a coordinate of U. The local certificate therefore exports only the displayed finite located cover route, with P and N recording provenance and naming for that route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ packet is generated from a closed bounded interval source, a finite real cover row, a finite-cover Lebesgue-number ledger, metric-ball located membership rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$, seed)

Scope closed: Definition 3726.1 and Definition 3726.2 seed the finite located cover substrate consumed by compact uniform-continuity routes over closed bounded interval data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, choice-selected subcovers, host-set membership oracles, quotient real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coverage exactness, locatedness preservation, Lebesgue-number handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.UniformlyLocatedFiniteCoverUp.

Chapter 2759

Cauchy Condensation Theorem NameCert Construction

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ packet records the theorem-level NameCert surface that turns a monotone nonnegative $\mathsf{RealSeries}^{\uparrow}$ tail into the dyadic block series and back through Cauchy criteria. It sits after the dyadic-block packet of chapter 1464, the comparison-test surface of chapter 1478, the Real-series carrier of chapter 3345, and the finite dyadic partition ledger of chapter 1863. The packet is a theorem certificate over displayed tails, block ledgers, and Cauchy handoff rows; it is not a host summability principle, a quotient of infinite series, a selected sum, countable choice, arbitrary trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2759.1 (Cauchy condensation theorem carrier)**.** *A* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the monotone nonnegative* $\mathsf{RealSeries}^{\uparrow}$ *source row, read only through finite term windows and partial-sum windows.*
2. *M is the monotone-tail ledger, recording the nonnegativity row, decreasing transport, and the finite tail cut from which the theorem is applied.*
3. *D is the* $\mathsf{DyadicPartition}^{\uparrow}$ *block schedule, listing the powers of two, block endpoints, and finite block intervals used by the theorem.*
4. *B is the original block-sum ledger, storing the finite sums of A over the dyadic intervals named by D.*
5. *G is the condensed-series row, displaying the dyadic terms $2^n a_{2^n}$ on the same block schedule.*
6. *Q is the blockwise comparison ledger, recording the lower and upper estimates that monotonicity supplies between B and G.*
7. *U is the original-series Cauchy criterion row, expressed through displayed partial-sum windows of A.*
8. *V is the condensed-series Cauchy criterion row, expressed through displayed partial-sum windows of G.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the original and condensed partial-sum windows.*
10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after the Cauchy criterion handoff has passed through U, V, R.*
11. *H records componentwise* hsame *transport for the source, monotone ledger, dyadic schedule, block sums, condensed terms, comparison ledger, Cauchy rows, readback row, seal row, continuation, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from each finite tail request through the dyadic blocks, comparison ledger, Cauchy criteria, regular readback, and seal-facing row.*
13. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, M, D, B, G, Q, U, V, R, E, H, C, P, N$. It has no coordinate for a host infinite sum, quotient equality of series, selected endpoint, countable choice, an arbitrary trichotomy split, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyCondensationTheoremUp.CauchyCondensationTheoremTasteGate single carrier alignment`

Definition 2759.2 (Blockwise Cauchy comparison classifier)**.** *Two accepted* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carriers are blockwise Cauchy equivalent when their displayed* $\mathsf{RealSeries}^{\uparrow}$ *source rows, monotone-tail ledgers, dyadic schedules, original block sums, condensed terms, comparison rows, original and condensed Cauchy criterion rows, regular readbacks, seal-facing rows, transports, continuations, provenance, and local names are pairwise* hsame *through the rows named in Definition 3727.1. Equivalence is not allowed to inspect an unlisted host limit, a quotient of streams, or a chosen infinite sum.*

Theorem 2759.3 (Cauchy condensation theorem NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carrier $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyConden}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed fourteen-row packet;*
2. *monotone nonnegative source exposure through A and M;*
3. *dyadic block scheduling through D and original block-sum formation through B;*
4. *condensed dyadic-term formation through G;*
5. *blockwise tail comparison through Q, using only monotone transport on displayed finite windows;*
6. *original-to-condensed Cauchy transport through U, V, R;*
7. $\mathsf{Real}^{\uparrow}$ *seal exposure through E only after the Cauchy handoff has reached the regular readback row;*
8. *transport stability, continuation replay, provenance, and local naming through H, C, P, N.*

No obligation row asserts host summability, a selected infinite sum, quotient equality of real streams, countable choice, arbitrary trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$. The carrier row exposes exactly the monotone Real-series source, monotone-tail ledger, dyadic block schedule, original block sums, condensed terms, comparison ledger, two Cauchy criterion rows, regular readback, seal-facing row, and structural rows.

The source and classifier obligations are finite. The rows A and M provide nonnegativity and decreasing behavior only on displayed term windows, while Definition 3727.2 permits comparison only through the carrier coordinates named in Definition 3727.1.

The dyadic block ledger is then read. The schedule D fixes the inspected powers of two and finite intervals, B records the original block sums on those intervals, and G records the condensed terms $2^n a_{2^n}$ on the same indices.

Monotone transport enters only through Q. Each block comparison is a finite estimate between rows already displayed by A, M, D, B, G, so the theorem-level packet does not appeal to an infinite selector of blocks or to host equality of series.

Finally the Cauchy-convergence handoff is read through U, V, R, E. The original and condensed Cauchy criterion rows route the two partial-sum tails into the same regular readback boundary before the $\mathsf{Real}^{\uparrow}$ seal row is visible. The rows H, C, P, N preserve, replay, source, and name that route, so the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ consumer that claims theorem-level condensation without reading the monotone source row, dyadic block schedule, original block sums, condensed terms, blockwise comparison ledger, Cauchy criterion rows, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: cauchycondensation, cauchy-condensation-test, cauchy-condensation-real-series, dyadic-partition, realseries, regseqrat, real: no listed sibling supplies the whole fourteen-row theorem packet $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$, because each sibling contributes only a dyadic block, comparison-test, Real-series, partition, readback, or seal face while this chapter binds the theorem-level equivalence through blockwise comparison and Cauchy handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ packet is generated from a monotone nonnegative RealSeries source row, a monotone-tail ledger, a DyadicPartition block schedule, original block sums, condensed dyadic terms, a blockwise comparison ledger, original and condensed Cauchy criterion rows, RegSeqRat readback, a RealUp seal-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3727.1, Definition 3727.2, and Definition 3727.3 seed the theorem-level dyadic block carrier, classifier, monotone transport ledger, and Cauchy-convergence handoff for monotone nonnegative real series.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient equality of series, selected sums, countable choice, arbitrary trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode BEDC.Derived.CauchyCondensationTheoremUp and prove carrier admission, blockwise classifier stability, monotone dyadic transport, the block ledger, and the Cauchy criterion handoff.

Chapter 2760

A Concrete Naming Certificate for $\mathsf{Sylow}^{\uparrow}$

Following Sylow's finite group theorem, the $\mathsf{Sylow}^{\uparrow}$ packet records a prime-power subgroup witness inside a displayed finite $\mathsf{Group}^{\uparrow}$ carrier. It consumes the group and subgroup rows of chapter 41 and chapter 80, the prime and unary counting rows of chapter 64 and chapter 28, and the finite action surface of chapter 3662. The packet names only a displayed p-subgroup carrier, maximal prime-power coverage, conjugacy transport, and representation-facing consumer boundary; it does not assert a host subgroup lattice, quotient group classification, character table, arbitrary normalizer computation, or complete finite-group structure theorem.

Definition 2760.1 (Sylow carrier)**.** *A* $\mathsf{Sylow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (G, S, p, n, W, A, T, C, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, including multiplication, unit, inverse, finite support, and componentwise group transport.*
2. *S is the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *row naming the candidate subgroup inside G.*
3. *p is the* $\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$ *row naming the prime whose power controls the subgroup order.*
4. *n is the* $\mathsf{Nat}^{\uparrow}$ *unary exponent row recording the displayed prime-power size p^n.*
5. *W is the membership and order witness row saying that the subgroup carrier has exactly the displayed p^n support and that no larger displayed p-subgroup row is present in the packet.*
6. *A is the finite conjugation-action row, read through the group operation of G and through the action discipline of* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *when a downstream representation consumer asks for transport of a Sylow row.*
7. *T is the conjugacy-transport row sending any displayed Sylow witness with the same prime-power coverage to a conjugate subgroup row inside G.*
8. *C is the consumer ledger for representation, character, and subgroup-normalizer reads; it exposes only the rows G, S, p, n, W, A, T before a consumer may name a Sylow conclusion.*
9. *H records componentwise* hsame *transport for group, subgroup, prime, exponent, coverage, action, conjugacy, consumer, replay, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{Sylow}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, S, p, n, W, A, T, C, H, P, N$. It has no coordinate for an ambient subgroup lattice, quotient classification, selected normalizer, character-table computation, or host finite-group theorem.

Lean4 (definition) `BEDC.Derived.SylowUp.SylowCarrier`
Lean4 ✓ `BEDC.Derived.SylowUp.SylowTasteGate single carrier alignment`

Theorem 2760.2 (Sylow NameCert obligations)**.** *For every accepted* $\mathsf{Sylow}^{\uparrow}$ *carrier* $Y = (G, S, p, n, W, A, T, C, H,$ *the local* $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$ *surface consists exactly of finite group admission through G, subgroup admission through S, prime admission through p, unary exponent exposure through n, prime-power coverage through W, conjugation-action exposure through A, conjugacy transport through T, consumer-boundary exposure through C, componentwise transport through H, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3728.1. The rows G, S, p, n, W expose the finite group, candidate subgroup, prime, unary exponent, and prime-power coverage before any Sylow read can be consumed.

The conjugation row A is finite because it is read from the displayed group operation and, when representation-facing consumers are present, through the finite action surface of chapter 3662. The transport row T is therefore a row of the packet, not a host orbit theorem.

Finally C, H, P, N preserve, source, and name the same route. Since no ambient subgroup lattice, quotient classification, selected normalizer, character-table computation, or complete finite-group structure theorem is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier namecert obligations`

Theorem 2760.3 (Sylow conjugacy transport)**.** *Let $Y = (G, S, p, n, W, A, T, C, H, P, N)$ and $Y' = (G, S', p, n, W', A', T', C', H', P', N')$ be accepted* $\mathsf{Sylow}^{\uparrow}$ *packets over the same displayed finite* $\mathsf{Group}^{\uparrow}$ *row and the same* $\mathsf{Prime}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *rows. If both coverage ledgers display maximal p^n support inside G, then every consumer that identifies S with S' must pass through the conjugation-action rows A, A' and the transport rows T, T'. The exported comparison is a displayed conjugacy transport inside G, not equality of host subgroup sets or a quotient of all subgroups.*

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier conjugacy transport`

Proof. Read the two carriers and their shared rows G, p, n. The only rows that connect subgroup witnesses with the ambient group operation are the coverage rows W, W' and the conjugation-action rows A, A'.

The transport rows T, T' record the allowed movement between displayed maximal prime-power subgroup rows. Thus a consumer comparing S and S' must use those rows before it may name a conjugacy result.

The consumer and structural rows $C, C', H, H', P, P', N, N'$ preserve the same displayed route. Because neither carrier contains a host subgroup lattice or quotient over all subgroup rows, the comparison remains a finite conjugacy-transport packet inside G. □

Theorem 2760.4 (Sylow prime-power coverage)**.** *For an accepted* $\mathsf{Sylow}^{\uparrow}$ *carrier* $Y = (G, S, p, n, W, A, T, C, H, P, N$ *every prime-power coverage read of* S *in* G *factors through the prime row* p*, the unary exponent row* n*, and the coverage ledger* W*. The packet may expose that the displayed subgroup has maximal* p^n *support among the subgroup rows named by the packet, but it does not compute the full host order of* G*, enumerate all subgroups of* G*, or choose a canonical Sylow subgroup.*

Proof. Project p, n, W from the carrier. The prime row supplies the arithmetic base, the unary exponent row supplies the finite counting index, and the coverage ledger records the displayed p^n support for the subgroup row S.

Apply Definition 3728.2. No coverage conclusion is available before G, S, p, n, W have been read, and no later row can replace W with an ambient order computation.

The action, transport, consumer, and structural rows A, T, C, H, P, N only replay the same coverage route. Therefore prime-power coverage is a packet-level finite witness, with full subgroup enumeration and canonical selection outside the chapter. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier prime power coverage`

Theorem 2760.5 (Sylow obligation closure surface)**.** *The* $\mathsf{Sylow}^{\uparrow}$ *obligation surface is exactly the carrier admission, prime-power coverage, conjugacy transport, and consumer-boundary route exposed by Definition 3728.2, Definition 3728.3, and Definition 3728.4.*

Proof. Apply Definition 3728.2 for the carrier-admission and consumer-boundary rows. Apply Definition 3728.4 for the prime-power ledger, and apply Definition 3728.3 for the displayed conjugacy route.

Together these three blocks expose the finite group, subgroup, prime, exponent, coverage, action, transport, consumer, structural, provenance, and local naming rows before any Sylow consumer can read the packet. No further host subgroup lattice, canonical selector, quotient classification, or character-table computation is a coordinate of the carrier. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier obligation closure surface`

Theorem 2760.6 (Sylow scope dependency lock)**.** *Every* $\mathsf{Sylow}^{\uparrow}$ *consumer must first read the* $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 3728.1. Only after those rows are exposed may the consumer name any prime-power coverage, conjugacy transport, or representation-facing consumer-boundary conclusion.*

Proof. Project the carrier and apply Definition 3728.2. The rows $G, S, p, n, W, A, T, C, H, P, N$ are the only admission, arithmetic, transport, replay, provenance, and local naming coordinates available to a consumer.

Apply Definition 3728.4 for the coverage read and Definition 3728.3 for the transport read. Both conclusions depend on the displayed rows already named by the carrier.

The consumer-boundary row C and the structural rows H, P, N then prevent any conclusion from escaping to an ambient subgroup lattice, quotient classification, character table, normalizer computation, or canonical Sylow selector. Thus the dependency scope is locked before the chapter exposes any Sylow conclusion. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier scope dependency lock`

Closure status for $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3728.1, Definition 3728.2, Definition 3728.3, and Definition 3728.4 seed the finite prime-power subgroup packet over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{Sylow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subgroup lattices, quotient group classification, character-table computation, arbitrary normalizer calculation, canonical Sylow selection, Lean verification, or bridge checking.

Upgrade path: Carrier admission, prime-power coverage, and conjugacy transport already have checked targets; the remaining next step is checked representation-facing consumer-boundary and public-scope coverage, followed by bridge discipline for BEDC.Derived.SylowUp.

Constructive story: A Sylow$^{\uparrow}$ packet is generated from a finite GroupUp row, a SubgroupUp row, a PrimeUp row, a NatUp exponent row, a prime-power coverage ledger, conjugation-action and transport rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2761

A Concrete Naming Certificate for $\mathsf{BernsteinPolynomial}^{\uparrow}$

Following Bernstein and Bishop–Bridges constructive analysis, the $\mathsf{BernsteinPolynomial}^{\uparrow}$ packet records constructive uniform approximation over displayed $\mathsf{Real}^{\uparrow}$ interval data as finite BEDC rows. It sits after the polynomial carrier of chapter 50, the real-name surface of chapter 38, the compact-uniform continuity packet of chapter 388, the uniform-modulus carrier of chapter 369, and the rational observation rows of chapter 37. The chapter names the finite Bernstein basis, coefficient, mesh, and error-ledger surface; it does not import a host Weierstrass theorem, a selected infinite approximation sequence, quotient equality of real functions, or ambient completeness beyond the cited rows.

Definition 2761.1 (Bernstein polynomial carrier)**.** *A* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, U, M, S, K, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *closed interval row, exposing the endpoint names and the interval membership boundary used by the approximation request.*
2. *F is the sampled* $\mathsf{Real}^{\uparrow}$ *function-read row over I, read only through the compact-uniform continuity surface.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}/\mathsf{UniformModulus}^{\uparrow}$ *row that supplies a finite modulus request for the interval.*
4. *M is the dyadic mesh row, recording the chosen mesh scale, mesh points, and interval subdivision ledger.*
5. *S is the* $\mathsf{Rat}^{\uparrow}$ *coefficient row, listing the rational sample and binomial-weight reads used by the displayed Bernstein polynomial.*
6. *K is the* $\mathsf{Poly}^{\uparrow}$ *Bernstein-basis row, recording the finite basis terms and the polynomial assembly route.*
7. *E is the uniform-error ledger, comparing each mesh and interval cell read against the requested tolerance by rational bounds.*
8. *H records componentwise* hsame *transport for interval, function-read, modulus, mesh, coefficient, basis, error, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from interval admission through compact-uniform modulus selection, dyadic mesh construction, rational coefficient readback, polynomial assembly, and uniform-error certification.*

10. *P records* Pkg *provenance over* $\mathsf{BernsteinPolynomial}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$ *row.*

The classifier compares accepted Bernstein packets only through $I, F, U, M, S, K, E, H, C, P, N$. It has no coordinate for host function equality, a nonconstructive supremum, a choice-selected polynomial sequence, or an external real-analysis theorem.

Lean4 ✓ `BEDC.Derived.BernsteinPolynomialUp.BernsteinPolynomialTasteGate single carrier alignment`

Theorem 2761.2 (Bernstein polynomial NameCert obligations)**.** *For every accepted* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *carrier*

$$B = (I, F, U, M, S, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$ *surface consists exactly of interval admission, compact-uniform modulus exposure, dyadic mesh construction, rational coefficient readback, Bernstein-basis polynomial assembly, uniform-error ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every approximation-facing consumer must read

$\mathsf{Real}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert

through these rows before it receives a Bernstein approximation certificate.

Theorem 2761.3 (Bernstein uniform approximation boundary)**.** *Given an accepted* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *packet and a displayed tolerance request, the exported approximation boundary is the finite route*

$$I, F, U \longmapsto M \longmapsto S, K \longmapsto E \longmapsto H, C, P, N.$$

The packet may certify only that the displayed Bernstein polynomial row meets the displayed uniform-error ledger over the displayed interval. It does not claim a free-standing Weierstrass theorem, an ambient complete function space, a quotient equality class of continuous functions, or a selected infinite chain of approximants.

Closure status for $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BernsteinPolynomial}^{\uparrow}$ packet is generated from Real interval rows, compact-uniform modulus rows, dyadic mesh rows, Rat coefficient rows, Polynomial Bernstein-basis rows, a uniform-error ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 3730.1, Definition 3730.2, and Definition 3730.3 seed the finite constructive approximation certificate over interval, modulus, mesh, coefficient, basis, error, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the Weierstrass approximation theorem, host function equality, selected approximation sequences, ambient complete function spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, modulus exposure, dyadic mesh construction, rational coefficient readback, polynomial assembly, uniform-error exactness, transport, replay, provenance, and local naming for BEDC.Derived.BernsteinPolynomialUp.

Chapter 2762

A Concrete Naming Certificate for $\mathsf{SequentialCauchySelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentialCauchySelector}^{\uparrow}$ packet records the finite selector rows needed when a displayed finite family of Cauchy requests is answered one request at a time, without turning that finite ledger into countable choice or ambient real completeness. It sits after the single-request selection surface of chapter 3719, the choice-sequence boundary of chapter 1482, the located choice-sequence boundary of chapter 2088, the stream-window carrier of chapter 126, the regular rational readback row of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite sequential handoff; it does not choose an infinite sequence of witnesses, quotient-identify completed reals, or assert a general completeness principle.

Definition 2762.1 (Sequential Cauchy selector carrier)**.** *A* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, I, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request ledger, listing the Cauchy-selection queries whose answers are demanded by the packet.*
2. *M is the modulus schedule, assigning each listed request its displayed precision and finite window budget.*
3. *I is the finite index ledger, ordering the listed requests and recording which rows have already been answered.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite stream windows admitted by Q, M, I.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and their regular rational comparison data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, recording dyadic thresholds, comparison rows, and error budgets for every listed request.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing a terminal real-facing name only after the selected stream window, regular readback, and dyadic tolerance rows have been read.*

8. *H records componentwise* hsame *transport for the request, modulus, index, window, readback, dyadic, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each finite request through its modulus schedule, index row, stream window, regular readback, dyadic ledger, and real seal.*

10. *P records* Pkg *provenance over* $\mathsf{SequentialCauchySelector}^{\uparrow}$, $\mathsf{CauchySelectionFunction}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LocatedChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$ *row.*

The classifier compares accepted selector packets only through $Q, M, I, W, R, D, E, H, C, P, N$. It has no coordinate for countable choice, an infinite choice sequence, quotient real equality, ambient real completeness, or a choice-selected family beyond the listed requests.

Lean4 ✓ `BEDC.Derived.SequentialCauchySelectorUp.SequentialCauchySelectorTasteGate single carrier alignment`

Theorem 2762.2 (Sequential Cauchy selector NameCert obligations)**.** *For every accepted* SequentialCauchySelector *carrier $S = (Q, M, I, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the request ledger, modulus schedule, index ledger, selected stream windows, regular rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *request-index exactness, requiring every selected row to be indexed by I and sourced from a displayed query in Q;*

3. *modulus-window compatibility, requiring every precision demand to pass through M before it reads a finite* $\mathsf{StreamName}^{\uparrow}$ *window in W;*

4. *readback and seal stability, requiring R, D, E to be transported componentwise from the same request, modulus, index, and window rows;*

5. *boundary refusal, excluding countable choice, infinite choice sequences, quotient real equality, ambient real completeness, and unlisted selected witnesses from the local carrier.*

Proof. Project the carrier of Definition 3731.1. The tuple $Q, M, I, W, R, D, E, H, C, P, N$ is the full accepted inventory, so each NameCert obligation must be read from one of these finite rows.

The request-index and modulus-window clauses are internal to Q, M, I, W. A selected window is exposed only after the query appears in Q, its finite index appears in I, and its precision demand passes through M.

The readback and seal clauses are downstream. The rows R and D read the displayed windows as regular rational and dyadic data, while E records the terminal real-facing seal obtained from the same finite route.

Finally H and C preserve the route under transport and replay, and P, N record provenance and local naming. Since no row stores countable choice, an infinite sequence, quotient real equality, ambient completeness, or an unlisted witness, the NameCert surface cannot export any of those objects. □

Theorem 2762.3 (Sequential Cauchy selector finite-window determinacy)**.** *Every consumer of an accepted* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *packet reads each selected Cauchy answer through the finite*

route

$$Q, I \longmapsto M \longmapsto W \longmapsto R, D \longmapsto E \longmapsto H, C, P, N.$$

Consequently two accepted reads agreeing on $Q, M, I, W, R, D, E, H, C, P, N$ determine the same finite selected windows, the same regular rational readback rows, the same dyadic tolerance ledgers, and the same terminal real-seal rows for the listed requests.

Proof. Fix an accepted carrier and a listed request. The index ledger I identifies the request row in Q, and the modulus schedule M fixes the precision and finite window budget attached to that row.

The selected window row W is read only after those rows have been fixed. Thus any second accepted read agreeing on Q, M, I, W reaches the same finite stream windows.

The rows R and D are readbacks from those windows, and the seal row E is exposed only after the same readback and tolerance data. Agreement on R, D, E therefore fixes the regular rational, dyadic, and real-facing coordinates for the listed request.

Transport, replay, provenance, and naming do not add fresh selector data. The rows H, C, P, N preserve and certify the same finite route, so determinacy is exactly the equality of the displayed packet rows. □

Theorem 2762.4 (Sequential Cauchy selector choice-boundary nonescape)**.** *Every read from an accepted* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *packet into* $\mathsf{CauchySelectionFunction}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{LocatedChoiceSequence}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *factors through the finite rows $Q, M, I, W, R, D, E, H, C, P, N$. The read may reuse the listed finite selections, but it cannot produce a countable-choice functional, an infinite choice sequence, quotient real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or selector rows for requests absent from Q.*

Proof. Start with the factorization route in Definition 3731.3. Every consumer sees the request, index, modulus, stream-window, readback, tolerance, and seal rows before it can inspect the structural rows.

The single-request surface of $\mathsf{CauchySelectionFunction}^{\uparrow}$ is therefore available only for the finitely listed requests. The $\mathsf{ChoiceSequence}^{\uparrow}$ and $\mathsf{LocatedChoiceSequence}^{\uparrow}$ chapters can read those displayed rows as finite sequence data, but no row supplies a law assigning witnesses to all future requests.

The regular rational, dyadic, and real-facing consumers likewise receive only R, D, E as finite downstream rows. These rows certify the listed readbacks and seals, not quotient equality of completed reals or ambient completeness.

Thus the packet remains inside its displayed BHist carrier. Any claimed countable-choice functional, infinite choice sequence, unlisted selector row, quotient real equality, or real-completeness theorem would require a coordinate absent from $Q, M, I, W, R, D, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCauchySelector}^{\uparrow}$ packet is generated from a finite request ledger, modulus schedule, index ledger, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3731.1, Definition 3731.2, Definition 3731.3, and Definition 3731.4 seed the finite sequential selector boundary over request, modulus, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, an infinite choice sequence, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, selector rows outside the finite request ledger, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-index exactness, modulus-window compatibility, readback and seal stability, and boundary-refusal proofs for BEDC.Derived.SequentialCauchySelectorUp.

Chapter 2763

A Concrete Naming Certificate for HyperbolicBusemannBoundary$^{\uparrow}$

The HyperbolicBusemannBoundary$^{\uparrow}$ packet realizes the boundary-fixed disk transport and horocycle reading of Definition 9463.10 as finite Busemann-level boundary data. It sits after the metric-read surface of chapter 3474, the horocycle packet of chapter 3534, the boundary-fixed transport route of chapter 3500, the disk phase-transport packet of chapter 3455, and the Denjoy-Wolff boundary-dynamics packet of chapter 3575. The chapter names a finite boundary metric and horocycle ledger; it does not turn the ideal boundary into a quotient compactification, select a global geodesic representative, or identify boundary dynamics with a host analytic fixed-point theorem.

Definition 2763.1 (Hyperbolic Busemann boundary carrier)**.** *A* HyperbolicBusemannBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, O, T, R, B, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* PoincareDiskPhaseTransport$^{\uparrow}$ *disk-phase row exposing the finite source, target, and reversible disk-coordinate route.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk observations.*
3. *O is the* HyperbolicHorocycle$^{\uparrow}$ *horocycle row, recording the boundary-tangent horocycle read used by the packet.*
4. *T is the* PoincareBoundaryFixedTransport$^{\uparrow}$ *row, requiring the displayed boundary end to remain fixed during the disk transport.*
5. *R is the* DenjoyWolffBoundaryDynamics$^{\uparrow}$ *boundary-dynamics row consumed only as displayed finite boundary-dynamics data.*
6. *B is the Busemann-level row, recording the finite distance-difference read against the fixed boundary end.*
7. *L is the boundary-horocycle ledger, pairing the Busemann-level row with the horocycle row and the boundary-fixed transport audit.*
8. *H records componentwise* hsame *transport for disk phase, metric, horocycle, boundary-fixed transport, boundary dynamics, Busemann-level, ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through disk phase, hyperbolic metric, horocycle read, boundary-fixed transport, boundary dynamics, Busemann-level comparison, and boundary-horocycle ledger exposure.*

10. *P records* Pkg *provenance over* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted boundary packets only through $D, M, O, T, R, B, L, H, C, P, N$. It has no coordinate for quotient ideal-boundary equality, a selected geodesic representative, a host compactification, or an analytic fixed-point theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.HyperbolicBusemannBoundaryUp.HyperbolicBusemannBoundaryTasteGate single carrier alignment`

Theorem 2763.2 (Hyperbolic Busemann boundary NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *carrier*

$$G = (D, M, O, T, R, B, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$ *surface consists exactly of disk-phase admission, hyperbolic-metric read compatibility, horocycle read exposure, boundary-fixed transport, boundary-dynamics provenance, Busemann-level distance-difference reading, boundary-horocycle ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

The Lean follow-up for this packet must provide `Nontrivial` *and* `FieldFaithful` *instances in* `lean4/BEDC/Derived/HyperbolicBusemannBoundaryUp/TasteGate.lean`*; the paper packet itself records only the finite NameCert obligation surface.*

Theorem 2763.3 (Hyperbolic Busemann horocycle route)**.** *Every accepted* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary read factors through*

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, DenjoyW

before the Busemann-level row and boundary-horocycle ledger are exposed. A consumer that bypasses the metric read, horocycle read, boundary-fixed transport, provenance, or local NameCert *row is outside the accepted packet.*

Falsifiable prediction: Any accepted boundary consumer omitting $\mathsf{HyperbolicMetric}^{\uparrow}$, horocycle, boundary-fixed transport, provenance, or local NameCert rows refutes the $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ packet.

Independence witness: `hyperbolic_metric`, `hyperbolic_horocycle`, `poincare_boundary_fixed_transport` `poincare_disk_phase_transport`, `denjoy_wolff_boundary_dynamics`: none is bijective to HyperbolicBusemannB because the packet adds the Busemann-level row and the boundary-horocycle ledger tying metric, horocycle, fixed-boundary transport, boundary dynamics, provenance, replay, and local naming into one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ packet is generated from disk-phase transport rows, HyperbolicMetric reads, HyperbolicHorocycle rows, boundary-fixed transport, Denjoy-Wolff boundary-dynamics rows, a Busemann-level distance-difference row, a boundary-horocycle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3732.1, Definition 3732.2, and Definition 3732.3 seed the finite boundary metric, horocycle, fixed-transport, Busemann-level, ledger, provenance, replay, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient ideal-boundary equality, selected geodesic representatives, host compactifications, analytic fixed-point theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-phase admission, metric-read compatibility, horocycle exposure, boundary-fixed transport, Busemann-level reading, boundary-horocycle ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBusemannBoundaryUp.

The BooleanAlgebra$^{\uparrow}$ hub routes its finite join, meet, complement, endpoint, order, transport, replay, provenance, and local naming rows to the lattice and Stone-duality consumers of chapter 55 and chapter 261.

Chapter 2764

A Concrete Naming Certificate for $\mathsf{BooleanAlgebra}^{\uparrow}$

Following Stone and Birkhoff, the $\mathsf{BooleanAlgebra}^{\uparrow}$ packet records the finite algebraic endpoint where complemented distributive lattice operations meet two-element truth rows. It sits beside the order carrier of chapter 53, the meet–join interface of chapter 55, and the endpoint discriminator of chapter 32; it supplies a finite order-algebra neighbour for later Stone-duality and topology bridges without assuming a host Boolean algebra. The packet consumes the finite meet/join order rows of chapter 55 and the endpoint discriminator of chapter 32 before the Stone-duality handoff of chapter 261. Its complemented distributive-lattice rows require the meet and join witnesses, the order comparison, and the complement-to-zero and complement-to-one endpoint checks before they can serve as the algebraic input exported to chapter 261. Before any local Boolean-algebra NameCert row is accepted, the packet must display the $\mathsf{Bool}^{\uparrow}$ truth-value rows of chapter 32 and the $\mathsf{Lattice}^{\uparrow}$ meet, join, and complement-stability rows of chapter 55; the Boolean classifier then reads endpoints and operations only through those sibling certificates. The finite complemented distributive-lattice rows are the algebraic input later consumed by chapter 261, while this chapter itself does not construct the Stone space, ultrafilters, or topology. The same rows also serve topology-facing consumers: when a displayed open-set regularization route is read as Boolean meet, join, and complement, the consumer must first expose the finite complemented distributive-lattice packet here, then pass through the indexed-open surface of chapter 88 and the clopen ledger of chapter 261. Thus regular-open and clopen uses receive the same algebraic rows, not a second host Boolean algebra or a powerset representation.

Definition 2764.1 (BooleanAlgebra BHist carrier)**.** *A* $\mathsf{BooleanAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (J, M, K, Z, O, R, H, C, P, N)$$

with the following displayed rows:

1. *J is the binary join row, read as the finite least-upper-bound operation inherited from the lattice-facing surface.*
2. *M is the binary meet row, read as the finite greatest-lower-bound operation inherited from the lattice-facing surface.*
3. *K is the complement row, assigning to each displayed carrier element its finite complement witness.*
4. *Z is the zero row, the finite bottom endpoint of the order surface.*

5. *O is the one row, the finite top endpoint of the order surface.*
6. *R is the partial-order row tying J, M, Z, O to the displayed* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *obligations.*
7. *H records componentwise* hsame *transport for join, meet, complement, endpoint, order, provenance, and local naming rows.*
8. *C records displayed* Cont *replay routes by which a consumer reads the operation, endpoint, order, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BooleanAlgebra}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host powerset algebra, excluded-middle oracle, ultrafilter space, locale of opens, quotient lattice, or ambient topological representation.

Lean4 (definition) `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier`

Theorem 2764.2 (BooleanAlgebra NameCert obligation surface)**.** *For the carrier of Definition 3733.1, the local* $\mathsf{NameCert}_{\mathsf{BooleanAlgebra}^{\uparrow}}$ *surface consists of exactly five finite obligation rows: carrier exposure for $J, M, K, Z, O, R, H, C, P, N$; lattice-order compatibility with the* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *rows; complement endpoint exactness, requiring $x \wedge Kx$ to read through Z and $x \vee Kx$ to read through O inside the displayed route; endpoint transport under componentwise* hsame *and displayed* Cont *replay; and non-escape from host Boolean algebras, powersets, ultrafilter spaces, locale carriers, quotient lattices, and topological representation theorems.*

Proof. Project the accepted carrier. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete visible inventory, with R carrying the order comparison and P naming the Bool, Poset, and Lattice sources.

Apply the sibling order surfaces. The poset row supplies the comparison route, while the lattice row supplies the meet and join readings; the Bool row supplies only finite endpoint discrimination, not an external truth oracle.

The complement row is checked against the displayed endpoints. The meet-with-complement route lands at Z, the join-with-complement route lands at O, and both reads are transported only by H and replayed only by C.

Since no row names a powerset algebra, ultrafilter space, locale carrier, quotient lattice, or ambient topology, coordinate exhaustion gives the stated non-escape boundary for the local naming certificate. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier namecert obligation surface`

Theorem 2764.3 (BooleanAlgebra Stone-duality forward route)**.** *Every* $\mathsf{StoneDuality}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet must read the finite join row J, meet row M, complement row K, zero row Z, one row O, order row R, componentwise* hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N before any Stone-space, ultrafilter, clopen-ledger, or topology bridge is public. Thus the Stone-duality source of chapter 261 starts from the displayed Boolean-algebra rows rather than from a host Boolean algebra or ambient Stone representation theorem.*

Proof. Project the accepted Boolean-algebra carrier. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete operation, endpoint, order, transport, replay, provenance, and naming inventory.

Apply the NameCert obligation surface. It requires the join, meet, complement, zero, one, and order rows to be displayed and transported before any downstream consumer can treat the packet as a Boolean-algebra source.

The Stone-duality chapter consumes exactly such a source row before it constructs clopen or topology-facing ledgers. Since the Boolean-algebra carrier contains no powerset algebra, ultrafilter space, locale carrier, quotient lattice, or ambient topology coordinate, no Stone-space or topology bridge can be public before these displayed rows are read. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone duality forward route`

Theorem 2764.4 (BooleanAlgebra Stone-source lattice determinacy)**.** *Any* StoneDuality$^{\uparrow}$ *consumer reading an accepted* BooleanAlgebra$^{\uparrow}$ *source receives the same finite join row J, meet row M, complement row K, zero row Z, one row O, order row R, componentwise* hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N from the carrier of Definition 3733.1. The endpoint discriminator is the* Bool$^{\uparrow}$ *row named by P, while the meet/join order is read through the* Poset$^{\uparrow}$ *and* Lattice$^{\uparrow}$ *rows also named by P.*

Thus two Stone-duality consumers that agree on the accepted Boolean-algebra packet cannot diverge by choosing different host Boolean algebras, powerset representations, ultrafilter spaces, locale carriers, quotient lattices, or topology bridges before those displayed rows have been read.

Proof. Project the accepted carrier. The inventory $J, M, K, Z, O, R, H, C, P, N$ is exactly the finite operation, endpoint, order, transport, replay, provenance, and naming surface exposed by Definition 3733.1.

Apply Definition 3733.2. It forces the complement endpoint reads to pass through Z and O, and it forces the order comparison to be the displayed Poset$^{\uparrow}$/Lattice$^{\uparrow}$ route recorded by R and named by P.

Now use Definition 3733.3. Every StoneDuality$^{\uparrow}$-facing read starts from those Boolean-algebra rows before any Stone-space or topology-facing ledger is public. Since the carrier has no coordinate for a host Boolean algebra or alternate representation, the source boundary is deterministic. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone source lattice determinacy`

Closure status for NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$.

Constructive story: A BooleanAlgebra$^{\uparrow}$ packet is generated from finite join, meet, complement, zero, one, order, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert rows, with Bool, Poset, and Lattice sources read only through their displayed NameCert surfaces.

Theory closure: TheoryGate(NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3733.2, Definition 3733.3, Definition 3733.4, Definition 3733.7, Definition 3733.8, and Definition 3733.10 bind the mature finite Bool, Poset, Lattice, Stone-source, and RegularOpenAlgebra route for Boolean algebra packets through the displayed finite order and algebra rows.

Formal verification: FormalStatus(NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Stone duality, Heyting algebra, locale theory, frame theory, ultrafilter construction, quotient Boolean algebra equality, host powerset semantics, or a Lean theorem.

Upgrade path: Lean verification requires checked carrier habitation, order-lattice compatibility, complement endpoint exactness, transport replay, and non-escape for the same finite packet.

Theorem 2764.5 (BooleanAlgebra dossier lattice route)**.** *Every top-level* BooleanAlgebra$^{\uparrow}$ *dossier read first consumes the* Bool$^{\uparrow}$ *endpoint rows and the* Lattice$^{\uparrow}$ *meet, join, and complement rows of the local carrier before it is available as a Boolean source for* StoneDuality$^{\uparrow}$*. The route adds no Stone space, ultrafilter, powerset representation, quotient lattice, or host Boolean-algebra equality row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraDossierLatticeRoute`

Proof. Project the carrier surface of chapter 3733. The endpoint coordinates are the displayed zero and one rows inherited from $\mathsf{Bool}^{\uparrow}$, while the algebraic coordinates are the meet, join, complement, and order rows inherited through the $\mathsf{Lattice}^{\uparrow}$ dependency route.

The Stone-duality handoff can read a Boolean source only after those local Boolean-algebra rows have been exposed. Since the dossier has no coordinate for Stone spaces, ultrafilters, powerset representatives, quotient lattices, or host equality, the outgoing route remains the existing Bool–Lattice–Stone dependency surface. □

The $\mathsf{BooleanAlgebra}^{\uparrow}$ dependency route consumes the $\mathsf{Bool}^{\uparrow}$ endpoint rows of chapter 32 together with the $\mathsf{Lattice}^{\uparrow}$ meet, join, and complement-facing lattice rows of chapter 55 before any $\mathsf{StoneDuality}^{\uparrow}$ handoff in chapter 261 is public.

Theorem 2764.6 (Boolean algebra lattice dependency route)**.** *Every accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *read exposes the finite meet, join, complement, endpoint, order, transport, replay, provenance, and local naming rows before any* $\mathsf{Lattice}^{\uparrow}$ *or* $\mathsf{StoneDuality}^{\uparrow}$ *consumer can use the packet. The lattice consumer of chapter 55 reads the meet and join rows together with the complement and endpoint rows as displayed Boolean data, while the Stone-duality consumer of chapter 261 receives the same rows only after that finite lattice surface has been made visible.*

Proof. Project the $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier rows used by the existing lattice and Stone routes. The meet and join rows are already displayed before the complement and endpoint rows can be consumed, and the structural rows only transport and replay that finite packet.

Thus a $\mathsf{Lattice}^{\uparrow}$ consumer reads the lattice part directly from the Boolean carrier. A $\mathsf{StoneDuality}^{\uparrow}$ consumer then uses the same Boolean-source rows as its input side, but no host Boolean algebra, powerset representation, or ambient topology is introduced between the two reads. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraLatticeDependencyRoute`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier consumes the finite meet, join, order, complement, and endpoint rows of chapter 55 before the Boolean source is read by chapter 261. Thus a $\mathsf{StoneDuality}^{\uparrow}$ handoff sees the same displayed $\mathsf{Lattice}^{\uparrow}$ dependency surface already present in the Boolean packet, rather than a host powerset algebra, quotient lattice, ultrafilter selection, or ambient topology.

Theorem 2764.7 (BooleanAlgebra obligation closure package)**.** *The accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet has an obligation-closure package over the displayed rows*

$$J, M, K, Z, O, R, H, C, P, N.$$

Carrier admission is the finite BHist *surface of Definition 3733.1; complement stability is the pair of endpoint reads* $x \wedge Kx$ *through* Z *and* $x \vee Kx$ *through* O *from Definition 3733.2; finite meet and join laws are consumed through the* $\mathsf{Lattice}^{\uparrow}$ *route of Definition 3733.6; and the Stone-source dependency is exported through Definition 3733.3 and Definition 3733.4. Thus the obligation package contains exactly carrier admission, complement stability, finite meet/join laws, Stone-source route, lattice dependency rows, transport, replay, provenance, and local naming, and no host powerset algebra, ultrafilter selector, quotient lattice, locale carrier, or ambient topology row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier obligation closure package`

Proof. Project the carrier from Definition 3733.1. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete finite inventory available to a Boolean-algebra consumer.

Use Definition 3733.2 for carrier admission and complement stability. It forces the complement row K to meet the zero endpoint Z and join the one endpoint O inside the displayed route, with transport and replay still attached.

Use Definition 3733.6 for the finite meet and join laws. The lattice dependency reads J and M together with complement, endpoint, order, transport, replay, provenance, and local naming rows before any lattice or Stone-duality consumer can use the packet.

Finally apply Definition 3733.3 and Definition 3733.4. They identify the Stone-source route with the same Boolean rows and prevent divergence through host Boolean algebras, powerset representations, ultrafilter spaces, quotient lattices, locale carriers, or topology bridges. These cited rows are exactly the obligation-closure package. □

Theorem 2764.8 (BooleanAlgebra public NameCert export)**.** *The finite* $\mathsf{BooleanAlgebra}^{\uparrow}$ *NameCert surface exported from Definition 3733.1 is public for* $\mathsf{StoneDuality}^{\uparrow}$ *and* $\mathsf{RegularOpenAlgebra}^{\uparrow}$ *consumers. A public read keeps the* $\mathsf{Bool}^{\uparrow}$ *endpoint rows,* $\mathsf{Poset}^{\uparrow}$ *order row,* $\mathsf{Lattice}^{\uparrow}$ *join and meet rows, complement row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row visible before either downstream consumer may treat the packet as a Boolean source.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier public namecert export`

Proof. Project the accepted carrier $B = (J, M, K, Z, O, R, H, C, P, N)$. The rows Z and O are the visible $\mathsf{Bool}^{\uparrow}$ endpoints, R is the $\mathsf{Poset}^{\uparrow}$ order route, and J, M, K are the $\mathsf{Lattice}^{\uparrow}$ join, meet, and complement-facing rows.

Use the local obligation surface and the obligation package. They bind complement endpoint exactness, order-lattice compatibility, transport, replay, provenance, and local naming to the same finite packet, so the public export cannot drop H, C, P, N while retaining a Boolean source.

The outgoing consumers read this same surface in two directions. $\mathsf{StoneDuality}^{\uparrow}$ consumes it as the Boolean-source side of Stone duality, while $\mathsf{RegularOpenAlgebra}^{\uparrow}$ consumes it as the algebraic surface behind regular-open meet, join, and complement. No host Boolean algebra, powerset representation, selected ultrafilter, quotient lattice, locale carrier, or ambient topology row is exported. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraPublicNameCertExport`

The $\mathsf{RegularOpenAlgebra}^{\uparrow}$ consumer of chapter 6835 reads the $\mathsf{BooleanAlgebra}^{\uparrow}$ join, meet, complement, endpoint, order, transport, replay, provenance, and local NameCert rows through topology-facing regularization. The read starts from the finite Boolean-algebra carrier of Definition 3733.1, passes through the regular-open closure and interior rows, and exports no second host Boolean algebra, powerset representation, quotient lattice, or detached topology carrier.

Theorem 2764.9 (BooleanAlgebra Stone-duality handoff)**.** *Every* $\mathsf{StoneDuality}^{\uparrow}$*-facing read of an accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet first consumes the finite join row* J*, meet row* M*, complement row* K*, endpoint rows* Z, O*, order row* R*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local* NameCert *row* N *of chapter 3733. Only after those rows are exposed may the consumer reach the Boolean-source side of chapter 261.*

The handoff is additive to the local Boolean-algebra carrier: it exports a concrete outgoing dependency edge to $\mathsf{StoneDuality}^{\uparrow}$ *while adding no Stone space, ultrafilter, clopen topology, powerset representation, quotient lattice, or host Boolean-algebra equality row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneDualityHandoff`

Proof. Project the accepted Boolean-algebra carrier from Definition 3733.1. Its visible operation and endpoint surface is exactly J, M, K, Z, O, R, with H, C, P, N carrying transport, replay, provenance, and local naming.

Apply Definition 3733.2. The local certificate requires the operation rows, endpoint exactness rows, and order compatibility rows to be displayed before any downstream packet can treat the carrier as a Boolean source.

The Stone-duality source of chapter 261 consumes a Boolean-algebra source before clopen or topology-facing rows appear. Therefore the Stone-duality-facing read must pass through the BooleanAlgebra rows first, and no absent Stone-space, ultrafilter, clopen-topology, powerset, quotient-lattice, or host-equality coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone duality handoff`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier exposes the join, meet, complement, endpoint, order, transport, replay, provenance, and local $\mathsf{NameCert}$ rows that a $\mathsf{StoneDuality}^{\uparrow}$ consumer later reads as its Boolean-source side. Thus the dependency into chapter 261 and the lattice-source handoff of chapter 55 is a finite classifier route through $J, M, K, Z, O, R, H, C, P, N$, not a hidden import of a host Boolean algebra, ultrafilter space, powerset representation, or ambient topology.

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneSourceLatticeForwardConsumer`

The same outgoing route also records the handoff row: after the Stone-source row is read from order and local naming, the provenance row transports it to the concrete downstream handoff row while the forward Stone-source certificate remains available to the same consumer.

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneHandoffForwardConsumer`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier supplies the complemented distributive-lattice source read by chapter 261 and the Boolean endpoint discipline read by chapter 5153. The horizontal route is the same finite join, meet, complement, endpoint, order, transport, replay, provenance, and local $\mathsf{NameCert}$ surface; it does not add a host powerset algebra, selected ultrafilter, ambient locale frame, quotient lattice equality, or topology object outside the displayed Boolean rows.

Theorem 2764.10 (Boolean algebra mature export)**.** *The finite* $\mathsf{BooleanAlgebra}^{\uparrow}$ *rows routed through this hub provide the mature export consumed by the* $\mathsf{Lattice}^{\uparrow}$ *and* $\mathsf{StoneDuality}^{\uparrow}$ *sibling routes. The export consists exactly of the join row* J*, meet row* M*, complement row* K*, endpoint rows* Z, O*, order row* R*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *of Definition 3733.1. It is the same packet exposed by the lattice dependency route, the Stone-duality handoff, the regular-open forward lattice route, the obligation closure package, and the public NameCert export.*

Proof. Project the carrier of Definition 3733.1. The rows $J, M, K, Z, O, R, H, C, P, N$ are the full finite Boolean-algebra inventory, so every export to a lattice or Stone-facing consumer must use those rows.

Apply Definition 3733.6. It makes the meet and join rows visible together with complement, endpoint, order, transport, replay, provenance, and local naming before any $\mathsf{Lattice}^{\uparrow}$ consumer uses the packet.

Apply Definition 3733.9 and Definition 3733.8. They send the same finite packet to $\mathsf{StoneDuality}^{\uparrow}$ and regular-open consumers after the public obligation surface has been exposed.

Finally use Definition 3733.7. It identifies carrier admission, complement endpoint exactness, finite meet and join laws, Stone-source routing, transport, replay, provenance, and naming as one package. Thus the mature export has no coordinate for a second host Boolean algebra, powerset

representation, selected ultrafilter, quotient lattice equality, detached locale frame, or ambient topology. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraMatureExport`

Chapter 2765

A Concrete Naming Certificate for $\mathsf{CoverageRankAlgebra}^{\uparrow}$

The $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet realizes the coverage-rank algebra of Definition 9462.1576 as finite BEDC NameCert data. It consumes the six-boundary tile coverage route of chapter 3470, the Boolean trigger lattice of chapter 3704, the window coverage-trigger algebra of chapter 3478, and the finite observation fold of chapter 2236. The packet binds boundary-tile coverage, rank, trigger support, classifier stability, replay, provenance, and local naming rows; it is not a biological code oracle, stochastic transition model, quotient codon space, hidden global support lattice, or infinite trigger search.

Definition 2765.1 (Coverage-rank algebra carrier)**.** *A* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, R, T, L, F, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the consumed* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row exposing the six boundary-tile coverage coordinates and their visible support faces.*
2. *R is the rank row, recording the finite support-union rank read from the coverage-rank algebra vision.*
3. *T is the boundary-trigger support row, carrying the visible trigger support whose equality or inequality is tested by the classifier.*
4. *L is the consumed* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *row naming Boolean meet, join, and support-rank comparisons over the same visible boundary supports.*
5. *F is the consumed* $\mathsf{FiniteObservationFold}^{\uparrow}$ *row that bounds every algebra read by a displayed finite observation fold and its omitted-tail ledger.*
6. *Q is the consumed* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *row, carrying the finite closure, blocker, threshold, and residue-facing algebra rows used by downstream trigger consumers.*
7. *H records componentwise* hsame *transport for coverage, rank, trigger support, Boolean lattice, finite fold, trigger algebra, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-trigger coverage request through B, R, T, L, F, Q, H.*

9. *P is the* Pkg *provenance over* $\mathsf{CoverageRankAlgebra}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{FiniteObservationFold}^{\uparrow}$, *the source vision of Definition 9462.1576,* BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, T, L, F, Q, H, C, P, N$. It has no coordinate for hidden biological semantics, stochastic dynamics, quotient codon equality, global support completion, host rank lattices, or infinite trigger enumeration.

Theorem 2765.2 (Coverage-rank algebra NameCert obligations)**.** *For every accepted* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *carrier $K = (B, R, T, L, F, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$ *surface consists exactly of boundary-tile coverage through B, rank exposure through R, boundary-trigger support through T, Boolean trigger-lattice comparison through L, finite observation folding through F, coverage-trigger algebra handoff through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The proposed Lean follow-up for `BEDC.Derived.CoverageRankAlgebraUp` *must provide explicit* Nontrivial $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *and* FieldFaithful $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *instances preserving $B, R, T, L, F, Q, H, C, P, N$ as separate displayed fields.*

Proof. Project the carrier of Definition 3734.1. The rows B, R, T, L, F, Q expose boundary-tile coverage, rank, trigger support, Boolean comparison, finite observation folding, and trigger-algebra handoff before any algebra consumer can read a coverage-rank conclusion.

The classifier is rank-sensitive but support-visible. Equal-rank reads are admitted only through the displayed trigger support row T and the Boolean comparison row L, so a consumer cannot collapse unequal boundary-trigger supports into one stable classifier row.

The finite observation fold F bounds every read, and Q supplies only the already displayed coverage-trigger algebra route. The rows H, C, P, N preserve, replay, source, and name this finite route, leaving hidden biological semantics, stochastic dynamics, quotient codon equality, global support completion, host rank lattices, and infinite trigger enumeration outside the carrier. □

Theorem 2765.3 (Coverage-rank boundary-trigger separation)**.** *Let $K = (B, R, T, L, F, Q, H, C, P, N)$ and $K' = (B', R', T', L', F', Q', H', C', P', N')$ be accepted* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *packets. If the rank rows R and R' display the same finite rank but the boundary-trigger support rows T and T' expose different support, then a classifier-stable identification of K and K' is refused unless it also supplies a displayed transport row through L, L', F, F', Q, Q' preserving the unequal support evidence. Thus equal rank alone never determines equality of the coverage-rank algebra packet.*

Lean4 ✓ `BEDC.Derived.CoverageRankAlgebraUp.CoverageRankAlgebraBoundaryTriggerSeparation`

Proof. Read the two carriers. The rank rows R, R' and trigger-support rows T, T' are separate coordinates, so equality of rank does not erase the displayed support data.

The Boolean lattice and finite-fold rows L, L', F, F' are the only rows that compare support and bound the observation. A classifier-stable identification must therefore preserve these rows before the trigger-algebra handoff can be read.

Finally $Q, Q', H, H', C, C', P, P', N, N'$ replay and name the same finite route. Since no host rank lattice or hidden global support completion is a carrier coordinate, equal rank with unequal boundary-trigger support remains separated at the packet boundary. □

Falsifiable prediction: Any accepted $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet that identifies equal rank with unequal boundary-trigger support without displaying the Boolean trigger-lattice, finite observation fold, coverage-trigger algebra, transport, replay, provenance, and local naming rows violates classifier stability.

Independence witness: six-boundary-tile-coverage, boolean-trigger-lattice, window-coverage-trigger-algebra, finite-observation-fold: $\mathsf{CoverageRankAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boundary-tile coverage, rank, boundary-trigger support, Boolean comparison, finite observation fold, coverage-trigger algebra handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet is generated from six-boundary tile coverage rows, rank rows, boundary-trigger support rows, Boolean trigger-lattice comparison rows, finite observation fold rows, coverage-trigger algebra rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3734.1, Definition 3734.2, and Definition 3734.3 seed the finite coverage-rank algebra packet over boundary-tile coverage, rank, trigger support, classifier, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CoverageRankAlgebraUp.CoverageRankAlgebraBoundaryTriggerSeparation`

Bridge status: none

Not claimed: This chapter does not close biological-code truth, stochastic transition laws, quotient codon equality, hidden global support lattices, host rank lattices, infinite trigger search, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rank exposure, boundary-trigger separation, classifier stability, Nontrivial CoverageRankAlgebraUp, and FieldFaithful CoverageRankAlgebraUp for BEDC.Derived.CoverageRankAlgebraUp.

Chapter 2766

A Concrete Naming Certificate for $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ packet records the finite boundary where an ordered Cauchy sequence can be read as a completion-facing real-order object without quotient equality, selected limits, or an ambient completeness primitive. It sits over the ordered sequence route of chapter 3636, the algebra route of chapter 3708, the regular rational handoff of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the complete-metric boundary of chapter 128. The packet names the ordered Cauchy completion boundary as displayed BEDC data; it is not a quotient completion, not a selected representative, and not a host real-order completeness theorem.

Definition 2766.1 (Cauchy sequence order completeness carrier)**.** *A* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, A, R, S, D, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the* ***??*** *row, carrying the finite order-comparison requests and monotone-tail witnesses available to the completion boundary.*
2. *A is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, carrying the finite algebraic closure operations used by the same tail windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the ordered algebraic Cauchy sequence data.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the inspected sequence entries and comparison windows are read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, split tolerances, and comparison budgets used by O, A, R, S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback and dyadic ledger have been displayed.*
7. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row, recording the finite completion-facing metric handoff visible to consumers after O, A, R, S, D, E.*

8. *H stores componentwise* hsame *transport for the order, algebra, regular readback, stream window, dyadic ledger, real seal, complete-metric boundary, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from ordered Cauchy requests through algebraic closure, stream windows, regular rational readback, dyadic comparison, real sealing, and the complete-metric handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$, *??*, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $O, A, R, S, D, E, M, H, C, P, N$. It contains no coordinate for quotient completion equality, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge-checked standard completion object.*

Theorem 2766.2 (Cauchy sequence order completeness NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier $K = (O, A, R, S, D, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed order, algebra, regular readback, stream-window, dyadic tolerance, real-seal, complete-metric boundary, transport, replay, provenance, and local naming rows;*
2. *order-algebra compatibility, requiring every completion-facing order comparison to read the same ?? row O and* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row A before any regular rational readback is exposed;*
3. *regular-name handoff, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be generated from the displayed stream windows S and dyadic tolerance ledger D;*
4. *real-seal and metric-boundary exposure, requiring the* $\mathsf{Real}^{\uparrow}$ *row E and* $\mathsf{CompleteMetric}^{\uparrow}$ *row M to appear only after O, A, R, S, D have been transported through H and replayed through C;*
5. *ledger nonescape, excluding quotient completion equality, selected representatives, countable choice, host real equality, ambient real completeness, and bridge checking from the local carrier.*

Proof. Project the carrier of Definition 3735.1. The accepted inventory is exactly $O, A, R, S, D, E, M, H, C, P, N$, so carrier admission and classifier comparison are finite.

The order and algebra rows are read together. A completion-facing order request cannot inspect the regular rational row until the displayed order row O, algebra row A, stream windows S, and dyadic comparison ledger D have been exposed.

The real seal and complete-metric boundary are downstream. The row E seals the regular rational readback through $\mathsf{Real}^{\uparrow}$, while M records only the finite completion-facing metric handoff after the same route has been transported and replayed.

Finally inspect the excluded coordinates. Since the carrier has no quotient equality, selected representative, countable-choice, host equality, ambient-completeness, or bridge-checking row, the local NameCert cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.CauchySequenceOrderCompletenessUp.CauchySequenceOrderCompletenessNamecertObligatio`

Theorem 2766.3 (Cauchy sequence order completeness nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier must traverse the displayed route*

??, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame,

before it can expose an ordered completion boundary. The route exports no quotient completion, no selected limit representative, no countable-choice schedule, no host real equality, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no bridge claim.*

Proof. Begin with an accepted carrier $K = (O, A, R, S, D, E, M, H, C, P, N)$. The only order source is O, and the only algebraic closure source is A; both are upstream of the finite stream windows and regular rational readback.

Read S, R, D next. These rows expose finite observations, regular rational data, and dyadic tolerance budgets before any real or metric boundary can be inspected.

The terminal rows E and M are reached only through the componentwise transports H and displayed replay C. The provenance row P and local naming row N source exactly that route.

Since the carrier inventory has no quotient, selector, choice, host equality, ambient completeness, or bridge coordinate, every ordered completion read remains inside the displayed finite BEDC route. □

Falsifiable prediction: Every $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ completion-facing read factors through displayed CauchySequenceOrder, CauchySequenceAlgebra, StreamName, RegSeqRat, DyadicRatCore, RealUp, CompleteMetric, transport, replay, provenance, and local naming rows before exposing an ordered completion boundary.

Independence witness: cauchy-sequence-order, cauchy-sequence-algebra, regseqrat, streamname, dyadicratcore, real, completemetric: $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ordered tail comparisons, algebraic Cauchy closure, regular rational readback, finite stream windows, dyadic tolerance rows, a Real seal, and a complete-metric handoff in one finite completion-boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ packet is generated from CauchySequenceOrder and CauchySequenceAlgebra rows, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a CompleteMetric boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3735.1, Definition 3735.2, and Definition 3735.3 seed the finite ordered Cauchy completion boundary over order, algebra, regular rational, stream, dyadic, real, and complete-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order-algebra compatibility, regular-name handoff, real-seal exposure, complete-metric boundary exposure, and ledger nonescape for BEDC.Derived.CauchySequenceOrderCompletenessUp.

Chapter 2767

A Concrete Naming Certificate for $\mathsf{HilleYosida}^{\uparrow}$

Following Hille and Yosida, the $\mathsf{HilleYosida}^{\uparrow}$ packet records the finite BEDC surface by which a densely read linear generator graph is accepted as strongly continuous contraction-semigroup generator data. It consumes the functional-analytic source of chapter 94, the Banach carrier of chapter 90, the linear graph rows of chapter 48, the norm ledgers of chapter 89, and the complete-metric convergence surface of chapter 128. The packet names the generator and resolvent route used by later evolution equations; it is not a spectral theorem, a PDE solver, a selected semigroup object outside the displayed rows, or a global regularity principle.

Definition 2767.1 (Hille–Yosida carrier)**.** *A* $\mathsf{HilleYosida}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (B, A, R, S, N, H, C, P, L)$$

with the following visible rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *source row, recording the ambient vector space, normed structure, and complete state classifier in which generator reads are interpreted.*
2. *A is the linear generator graph row, a* $\mathsf{LinMap}^{\uparrow}$*-facing finite graph with its displayed domain row and graph-compatibility observations.*
3. *R is the positive-parameter resolvent ledger, recording the displayed $\lambda > 0$ requests, resolvent graph rows, domain-return rows, and resolvent identity comparisons used by the generator test.*
4. *S is the contraction semigroup row, recording the finite time-parameter observations, identity-at-zero row, composition replay row, and contraction comparisons visible to a strongly continuous semigroup consumer.*
5. *N is the norm-bound ledger, joining the* $\mathsf{Norm}^{\uparrow}$ *contraction bounds for S and the positive-parameter resolvent bounds for R.*
6. *H records componentwise* hsame *transport for the Banach source, generator graph, resolvent ledger, semigroup row, norm bounds, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from finite positive-parameter and finite time requests through*

$$B, A, R, N \longmapsto S, H, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{HilleYosida}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a chosen global semigroup outside S, a spectral calculus, differentiability beyond the displayed graph, maximal PDE regularity, excluded middle, quotient equality, or choice-selected limits.

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaTasteGate single carrier alignment`

Theorem 2767.2 (Hille–Yosida NameCert obligations)**.** *For an accepted* $\mathsf{HilleYosida}^{\uparrow}$ *carrier*

$$Y = (B, A, R, S, N, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Banach and functional-analysis source compatibility for B, linear generator graph exposure for A, positive-parameter resolvent accounting for R, contraction semigroup visibility for S, norm-bound accounting for N, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through L.*

Proof. Carrier admission is projection from Definition 3736.1. The accepted packet displays the Banach source, generator graph, resolvent ledger, semigroup row, norm ledger, transport row, replay row, provenance row, and local name.

The analytic obligations are read in the displayed order. The source row B fixes the complete normed state space, the generator row A supplies the linear graph, and the resolvent row R records the positive-parameter reads whose estimates are accounted for by N.

The semigroup-facing obligation is then restricted to S. A consumer may read finite time observations, the identity-at-zero row, composition replay, and contraction comparisons, but it cannot import an additional selected semigroup object.

Finally H, C, P, L transport, replay, source, and name exactly the same route. Since no row stores spectral calculus, maximal PDE regularity, quotient equality, excluded middle, or choice-selected limits, those principles are outside the local certificate surface. □

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaCarrier namecert obligations`

Theorem 2767.3 (Hille–Yosida semigroup generator handoff)**.** *Every strongly continuous semigroup generator read accepted by a* $\mathsf{HilleYosida}^{\uparrow}$ *carrier factors through*

$\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Equivalently, an evolution-equation consumer may inspect the Banach source B, generator graph A, positive-parameter resolvent ledger R, contraction semigroup row S, norm-bound ledger N, and structural rows H, C, P, L, but it cannot bypass them by selecting a global semigroup, invoking spectral calculus, or reading a host completion principle not displayed by the packet.

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaSemigroupGeneratorHandoff`

Proof. Use Definition 3736.2. It exposes a generator-facing read only after B, A, R, S, N have been projected from the finite carrier.

The resolvent side is controlled by R and N. Positive-parameter requests, domain-return rows, identity comparisons, and norm bounds are visible only as finite ledger entries in those coordinates.

The semigroup side is controlled by S. Strong continuity and contraction information are read as the displayed time-row observations and replay rows, not as an external choice of a completed semigroup.

The structural rows H, C, P, L then preserve the same data under transport, replay, provenance, and local naming. Thus a later ODE or PDE evolution packet receives precisely the finite generator handoff and no stronger global regularity or spectral conclusion. □

Falsifiable prediction: Every $\mathsf{HilleYosida}^{\uparrow}$ generator handoff read factors through one Banach and FunctionalAnalysis source row, one linear generator graph row, one positive-parameter resolvent ledger, one contraction semigroup row, one norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: functionalanalysis, banach, linearmap, norm, completemetric: $\mathsf{HilleYosida}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a generator graph, positive-parameter resolvent ledger, contraction semigroup row, norm-bound ledger, and the transport/replay/provenance rows tying them into a semigroup-generator certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3736.1, Definition 3736.2, and Definition 3736.3 seed the finite generator certificate from Banach and functional-analysis source rows through linear graph, resolvent, semigroup, norm-bound, transport, replay, provenance, and local naming data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral calculus, maximal PDE regularity, a selected global semigroup outside the displayed row, quotient equality, choice principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator graph exposure, resolvent positive-parameter accounting, contraction semigroup visibility, norm-bound stability, complete-metric convergence visibility, transport stability, replay stability, and ledger non-escape for BEDC.Derived.HilleYosidaUp.

Constructive story: A $\mathsf{HilleYosida}^{\uparrow}$ packet is generated from one Banach and FunctionalAnalysis source row, one LinearMap generator graph, one positive-parameter resolvent ledger, one contraction semigroup row, one Norm ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2768

A Concrete Naming Certificate for $\mathsf{CompactUniformOscillation}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor finite-net route, the CompactUniformOscillatio packet records the finite data by which compact metric source rows and uniform-modulus rows produce bounded target-distance oscillation witnesses. It consumes the compact metric source of chapter 129, the continuous-map graph of chapter 130, the uniform-modulus ledger of chapter 369, and the target distance surface of chapter 57. The packet is a finite oscillation-bound consumer for compact-uniform-continuity routes; it is not an open-cover compactness principle, a selected supremum, host extensionality of functions, or a global image-compactness theorem.

Definition 2768.1 (Compact uniform oscillation carrier)**.** *A* $\mathsf{CompactUniformOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, M, D, W, H, C, P, L)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, recording the compact finite-net observations and metric comparison rows on the source space.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the finite graph evaluations and local continuity reads used by the compact-uniform route.*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger, recording tolerance requests, source-radius responses, and the modulus rows obtained from compact finite nets.*
4. *D is the target-distance row, a* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{RealMetric}^{\uparrow}$ *readback ledger for the distances between displayed output observations.*
5. *W is the finite* ProbeBundle *oscillation-witness row, recording source net points, paired source observations within the modulus radius, and the resulting target-distance bounds.*
6. *H records componentwise* hsame *transport for the compact source, continuous graph, uniform modulus, target-distance ledger, oscillation witnesses, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from each finite target-oscillation request through*

$$K, F, M, D, W \longmapsto H, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformOscillation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected supremum of image diameters, open-cover compactness, quotient equality of maps, choice-selected net refinements, trichotomy, or host function extensionality.

Lean4 ✓ `BEDC.Derived.CompactUniformOscillationUp.CompactUniformOscillationTasteGate single carrier alignment`

Theorem 2768.2 (Compact uniform oscillation NameCert obligations)**.** *For an accepted* CompactUniformOscillatio *carrier*

$$O = (K, F, M, D, W, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, compact finite-net exposure for K, continuous graph exposure for F, uniform-modulus accounting for M, target-distance accounting for D, finite oscillation-witness exposure for W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Carrier admission is projection from Definition 3737.1. The accepted packet displays the compact metric source, continuous-map graph, uniform-modulus ledger, target-distance ledger, finite witness row, transport row, replay row, provenance row, and local name.

The compact-uniform route is read from K, F, M. A consumer first receives finite compact-net observations, then the displayed continuous graph rows, and then the tolerance-only modulus ledger built over those observations.

The oscillation estimate is read from D and W. Target-distance comparisons are visible only through D, while W stores the finite witness pairs whose source separation is small enough for the modulus row.

Finally H, C, P, L transport, replay, source, and name the same route. Because no coordinate stores a selected supremum, open-cover compactness, quotient equality, choice-selected refinements, trichotomy, or host function extensionality, those principles are outside the local certificate surface. □

Theorem 2768.3 (Compact uniform oscillation modulus handoff)**.** *Every bounded image-diameter or equicontinuity-facing read accepted by a* $\mathsf{CompactUniformOscillation}^{\uparrow}$ *carrier factors through*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, NameCert.

Equivalently, a downstream consumer may inspect compact finite-net data K, continuous graph data F, the uniform-modulus ledger M, target-distance rows D, finite oscillation witnesses W, and structural rows H, C, P, L, but it cannot bypass them by selecting a supremum, importing open-cover compactness, or comparing maps by host extensionality.

Proof. Apply Definition 3737.2. It forces every oscillation-facing request to open K, F, M, D, W before the structural rows are available.

The finite-net part is controlled by K and M. Compactness is consumed as displayed net data and a tolerance-only modulus ledger, not as an external open-cover principle.

The target estimate is controlled by D and W. The witness row stores the paired observations whose target distances are bounded, and the distance ledger records the corresponding metric readback.

Transport, replay, provenance, and naming through H, C, P, L preserve exactly that route. Hence later bounded-image or equicontinuity packets receive a finite oscillation-bound handoff and no selected supremum or host-level function equality. □

Falsifiable prediction: Every $\mathsf{CompactUniformOscillation}^{\uparrow}$ oscillation read factors through one CompactMetric finite-net source row, one ContinuousMap graph row, one UniformModulus ledger, one target Metric or RealMetric distance row, one finite ProbeBundle oscillation-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric, realmetric: $\mathsf{CompactUniformOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net data, continuous graph data, a uniform-modulus ledger, target-distance readback, finite oscillation witnesses, and the transport/replay/provenance rows tying them into an oscillation-bound packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3737.1, Definition 3737.2, and Definition 3737.3 seed the finite compact-uniform oscillation packet over compact metric source rows, continuous graph rows, uniform-modulus ledgers, target-distance readbacks, finite witnesses, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected suprema, open-cover compactness, image compactness, host function extensionality, choice-selected net refinements, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous graph exposure, uniform-modulus accounting, target-distance accounting, oscillation-witness stability, transport stability, replay stability, and ledger non-escape for BEDC.Derived.CompactUniformOscillationUp.

Constructive story: A $\mathsf{CompactUniformOscillation}^{\uparrow}$ packet is generated from one CompactMetric finite-net source row, one ContinuousMap graph row, one UniformModulus ledger, one target-distance row, one finite ProbeBundle oscillation-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2769

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$

Following Bishop and Bridges on diagonal arguments in constructive analysis, the $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ packet names the finite window data that a Real or Completion route may inspect before it reads a diagonal regular Cauchy seal. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the downstream diagonal extraction route of chapter 1773. The packet records the named diagonal-window boundary; it is not a quotient stream, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2769.1 (Regular Cauchy diagonal window carrier)**.** *A* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row containing the requested dyadic radius, split-tolerance ledger, and comparison budget for the diagonal window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite stream-window row that displays the source entries available to the diagonal read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the same window and tolerance budget.*
4. *I is the finite diagonal index schedule, recording only the named indices whose windows are inspected by the consumer.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after D, S, R, I have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, stream window, regular readback, diagonal index schedule, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal-window request through the tolerance row D, stream window S, regular readback R, index schedule I, and Real seal E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, I, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalWindowUp.RegularCauchyDiagonalWindowTasteGate single carrier alignment`

Theorem 2769.2 (Regular Cauchy diagonal window obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalWindo}$ *carrier $W = (D, S, R, I, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance row, stream window, regular readback, diagonal index schedule, Real seal, transport, replay, provenance, and local naming row;*
2. *diagonal-window visibility, requiring each consumer-facing diagonal request to read a displayed finite window before any Real seal is exposed;*
3. *dyadic tolerance exactness, requiring the requested tolerance and every split tolerance to be read through D;*
4. *regular readback compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume the same finite window S and diagonal schedule I named by the carrier;*
5. *Real-seal nonescape, requiring E to appear only after D, S, R, I have been transported through H and replayed through C;*
6. *transport, replay, provenance, and local naming for exactly the displayed rows $D, S, R, I, E, H, C, P, N$.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3738.1. The accepted inventory is exactly the dyadic tolerance row D, stream window S, regular readback R, diagonal index schedule I, Real seal E, and the structural rows H, C, P, N.

The diagonal-window obligation is therefore visible before the seal. A consumer request first reads the dyadic tolerance and split budget in D, then the finite stream window S, and then the regular rational readback R attached to the displayed schedule I.

The Real seal is downstream of that finite route. Componentwise hsame transport in H preserves the displayed rows, and Cont replay in C repeats the same ordered route before E can be inspected.

Finally, provenance and naming are exhausted by P and N. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness, no such obligation can be exported by the local certificate. □

Theorem 2769.3 (Regular Cauchy diagonal window nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

together with the finite diagonal index schedule. The packet exports no quotient stream, selected limit, countable-choice schedule, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted carrier $W = (D, S, R, I, E, H, C, P, N)$. The only tolerance source is D, the only finite source-window row is S, and the only regular rational readback row is R.

The diagonal schedule I is finite and displayed. It selects only the named window entries already present in S and the tolerance budgets already present in D before the readback reaches R.

The terminal Real seal E is reached only through the componentwise transport H and displayed replay C. The provenance row P and local naming row N source exactly that route.

Since the carrier inventory has no quotient-stream, selected-limit, countable-choice, trichotomy, host-equality, or ambient-completeness coordinate, every Real or Completion read remains inside the displayed finite diagonal-window packet. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ Real or Completion read factors through displayed DyadicRatCore, StreamName, RegSeqRat, RealUp, transport, replay, provenance, local naming, and finite diagonal-index rows before exposing a diagonal seal.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-diagonal-extraction: $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance exactness, finite stream-window exposure, regular rational readback, a diagonal index schedule, and a Real seal in one finite window packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite window, RegSeqRat readback, a finite diagonal index schedule, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3738.1, Definition 3738.2, and Definition 3738.3 seed the finite diagonal-window packet used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyDiagonalWindowUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-window visibility, dyadic tolerance exactness, regular readback compatibility, Real-seal exposure, and ledger nonescape for BEDC.Derived.RegularCauchyDiagonalWindowUp.

Chapter 2770

A Concrete Naming Certificate for DunfordFunctionalCalculus$^{\uparrow}$

Following Dunford and Schwartz on contour functional calculus for bounded operators, the DunfordFunctionalCalculu packet records the finite NameCert route by which a holomorphic scalar window is read through resolvent rows before any spectral theorem or projection-valued measure is public. It sits beside the Banach-algebra source of chapter 2189, the bounded-operator carrier of chapter 1644, the resolvent identity packet of chapter 3715, the spectral-radius boundary of chapter 2352, and the compact-operator schedule of chapter 1952.

Definition 2770.1 (Dunford functional calculus carrier)**.** *A* DunfordFunctionalCalculus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the* BanachAlgebra$^{\uparrow}$ *source row in which products, scalar multiplication, and norm estimates are read.*
2. *T is the* BoundedLinearOperator$^{\uparrow}$ *row naming the operator whose calculus is being inspected.*
3. *Γ is the finite contour-window row. It records the oriented contour pieces, mesh endpoints, denominator separation witnesses, and enclosure data used by the accepted request.*
4. *Ω is the holomorphic scalar-window row. It records the finite function values and local compatibility checks visible on the contour windows.*
5. *R is the* ResolventIdentity$^{\uparrow}$ *row. It supplies the displayed shifted-operator inverse checks and two-shift identity ledger used along Γ.*
6. *I is the contour-integral ledger. It records only finite Riemann-sum windows, dyadic tolerance budgets, and composition order through A, T, Γ, Ω, R.*
7. *Q is the spectral-radius and compact-operator boundary row. It allows later spectral consumers to read enclosure and compact-image schedules, but only as rows already displayed by chapter 2352 and chapter 1952.*
8. *S is the terminal calculus handoff row naming the value of the accepted contour read as a bounded-operator-facing packet.*

9. *H records componentwise* hsame *transport for the Banach-algebra, operator, contour, scalar-window, resolvent, integral-ledger, spectral-boundary, handoff, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from contour requests through $A, T, \Gamma, \Omega, R, I, Q, S, H, P, N$.*
11. *P is the* Pkg *provenance over* DunfordFunctionalCalculus$^{\uparrow}$, BanachAlgebra$^{\uparrow}$, BoundedLinearOperator$^{\uparrow}$, ResolventIdentity$^{\uparrow}$, SpectralRadius$^{\uparrow}$, CompactOperator$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{DunfordFunctionalCalculus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a spectral projection, a projection-valued measure, quotient operator equality, a selected spectrum, countable choice, or a host holomorphic functional calculus outside the displayed contour packet.

Lean4 ✓ `BEDC.Derived.DunfordFunctionalCalculusUp.DunfordFunctionalCalculusTasteGate single carrier alignment`

Theorem 2770.2 (Dunford functional calculus NameCert obligations)**.** *For every accepted* DunfordFunctionalCalcu *carrier*

$$F = (A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N),$$

the local NameCert$_{\text{DunfordFunctionalCalculus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Banach-algebra source compatibility through A, bounded-operator source compatibility through T, contour-window admission through Γ, holomorphic scalar-window admission through Ω, resolvent identity exactness through R, finite contour-ledger exactness through I, spectral-radius and compact-operator boundary visibility through Q, terminal calculus handoff through S, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every calculus-facing consumer must read the same rows before using the contour value. It receives no spectral projection, no spectral measure, no selected spectrum, no quotient equality of operators, no countable-choice schedule, and no ambient host functional calculus.

Proof. Carrier admission is projection from Definition 3740.1. The accepted packet displays exactly $A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N$.

The source rows A and T fix the Banach-algebra and bounded-operator surfaces. The contour and scalar-window rows Γ, Ω then provide the finite requests on which the calculus read is allowed to inspect function values.

The resolvent row R supplies only the displayed inverse checks and two-shift identity ledger. The contour ledger I reads those rows through finite Riemann-sum and dyadic-budget windows, so it cannot import a host integral or an unrecorded inverse.

The spectral-boundary row Q and handoff row S are downstream of the same route. Transport, replay, provenance, and local naming in H, C, P, N preserve that finite packet. Since the carrier has no coordinate for spectral projections, spectral measures, selected spectra, quotient equality, countable choice, or a host calculus, the listed rows exhaust the local obligations. □

Theorem 2770.3 (Dunford functional calculus resolvent-contour handoff)**.** *Every bounded-operator or spectral-analysis consumer of an accepted* DunfordFunctionalCalculus$^{\uparrow}$ *carrier reaches the calculus value only through the displayed route*

$$A, T \longmapsto \Gamma, \Omega \longmapsto R \longmapsto I \longmapsto Q, S \longmapsto H, C, P, N.$$

Thus the consumer may read Banach-algebra operations, the bounded operator, contour windows, holomorphic scalar windows, resolvent identity rows, finite contour-integration budgets, spectral-radius and compact-operator boundary rows, the terminal handoff, transport, replay, provenance, and local naming, and it may not read spectral projections, spectral measures, quotient operator equality, a selected spectrum, countable choice, or a host functional calculus.

Proof. Begin with a calculus-facing consumer. The only operator sources in the carrier are A and T, and the only contour-facing source rows are Γ and Ω.

The resolvent dependency is forced through R. It supplies the two-sided inverse checks and identity ledger already isolated by the $\mathsf{ResolventIdentity}^{\uparrow}$ packet, so no separate selected inverse or quotient operator comparison is available.

The finite contour read is then the row I. It records the Riemann-sum windows, dyadic tolerance budgets, and composition order needed to pass from the contour and resolvent rows to the handoff.

Finally Q, S, H, C, P, N expose the spectral boundary, terminal bounded-operator-facing result, transport, replay, provenance, and local naming. The excluded projection, spectral-measure, selected-spectrum, choice, quotient, and host-calculus objects have no carrier coordinate and therefore cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3740.1, Definition 3740.2, and Definition 3740.3 seed the contour functional-calculus route over Banach-algebra, bounded-operator, contour-window, holomorphic scalar-window, resolvent, spectral-boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral projections, spectral measures, selected spectra, quotient operator equality, countable choice, host holomorphic functional calculus, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contour-window admission, scalar-window admission, resolvent exactness, contour-ledger exactness, spectral-boundary visibility, handoff non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DunfordFunctionalCalculusUp.

Constructive story: A $\mathsf{DunfordFunctionalCalculus}^{\uparrow}$ packet is generated from a BanachAlgebra source row, a BoundedLinearOperator row, finite contour windows, holomorphic scalar windows, a ResolventIdentity route, a finite contour ledger, SpectralRadius and CompactOperator boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2771

A Concrete Naming Certificate for $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$

Following Kelley on uniform spaces and Bishop–Bridges on completion by explicit regular data, the $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ packet records the finite common-window route by which uniform-completion requests share one observation window before any $\mathsf{Real}^{\uparrow}$ seal is read. It concretizes the Real and Completion vision of chapter 9238 by tying the $\mathsf{UniformCompletionFunctor}^{\uparrow}$ surface to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows without quotient completion equality. Its concrete dependency route reads the functor source of chapter 1892, the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic budget of chapter 291, and the terminal real-name seal of chapter 38.

Definition 2771.1 (Uniform completion common-window carrier)**.** *A* $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (U, E, S, R, D, L, F, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *source row naming the completion-facing request, entourage scale, and functorial handoff boundary.*
2. *E is the finite entourage-window row. It records the common uniform scale used by all source observations in the accepted request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *common-window row. It synchronizes the finite observation windows before any regular rational readback is exposed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source-readback row generated from the synchronized window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, recording the dyadic tolerance, slack, comparison, and endpoint-conversion entries used by the readback.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is exposed only after S, R, D have displayed the shared window and regular rational readback.*
7. *F is the common-window functoriality row. It states that completion-facing consumers of U and Real-facing consumers of L read the same E, S, R, D route.*
8. *H records componentwise* hsame *transport for the uniform-completion source, common entourage window, StreamName window, RegSeqRat readback, DyadicRatCore budget, Real seal, functoriality, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through $U, E, S, R, D, L, F, H, P, N$.*
10. *P is the* Pkg *provenance over* UniformCompletionCommonWindow$^{\uparrow}$, UniformCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformCompletionCommonWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected completion points, host metric completeness, countable choice, or a second private observation window.

Lean4 ✓ `BEDC.Derived.UniformCompletionCommonWindowUp.UniformCompletionCommonWindowTasteGate`
`single carrier alignment`

Theorem 2771.2 (Uniform completion common-window NameCert obligations)**.** *For every accepted* UniformCompletionCommonWindow$^{\uparrow}$ *carrier*

$$W = (U, E, S, R, D, L, F, H, C, P, N),$$

the local NameCert$_{\text{UniformCompletionCommonWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-completion source compatibility through U, common entourage-window admission through E, synchronized* StreamName$^{\uparrow}$ *admission through S,* RegSeqRat$^{\uparrow}$ *readback exactness through R,* DyadicRatCore$^{\uparrow}$ *budget exactness through D,* Real$^{\uparrow}$ *seal handoff through L, common-window functoriality through F, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every completion-facing or Real-facing consumer must use the same E, S, R, D route. It receives no quotient completion equality, no selected completion point, no host metric-completeness principle, no countable-choice schedule, and no private window not displayed by the packet.

Lean4 ✓ `BEDC.Derived.UniformCompletionCommonWindowUp.UniformCompletionCommonWindowNameCertObligations`

Proof. Carrier admission is projection from Definition 3741.1. The displayed rows are exactly $U, E, S, R, D, L, F, H, C, P, N$.

The uniform-completion source row U is read first, and the common entourage window E fixes the finite scale for the request. The synchronized StreamName row S then supplies the only observation window available to the packet.

The regular rational and dyadic rows R, D are downstream of that shared window. The Real seal L can be exposed only after those rows display the readback and budget entries, so no consumer can substitute a quotient completion class or selected limit.

The row F ties the completion-facing and Real-facing reads to the same route. Transport, replay, provenance, and local naming in H, C, P, N preserve the route without adding host completeness, choice, or a second private window. □

Theorem 2771.3 (Uniform completion common-window Real seal handoff)**.** *Every* UniformCompletionFunctor$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* UniformCompletionCommonWindow$^{\uparrow}$ *carrier reaches the terminal Real seal only through*

$$U \longmapsto E \longmapsto S \longmapsto R, D \longmapsto L \longmapsto F, H, C, P, N.$$

Thus the completion request, common entourage window, synchronized StreamName window, RegSeqRat readback, DyadicRatCore budget, RealUp seal, common-window functoriality, transport, replay, provenance, and local naming are jointly visible, and quotient completion equality, selected completion points, host metric completeness, countable choice, and private observation windows are not exported.

Proof. Start with a completion-facing or Real-facing read. The source request is U, and the only uniform scale that may be consumed is the common row E.

The observation side is forced through S. Once that StreamName window is displayed, the regular rational readback R and dyadic budget D provide the finite data needed by the Real seal.

The seal row L is therefore downstream of the shared completion window, not of a quotient completion object. The functoriality row F records that both the completion consumer and the Real consumer read the same route.

Finally H, C, P, N transport, replay, source, and name the packet. Since no coordinate stores selected completion points, host completeness, choice, or private windows, the handoff exports only the displayed common-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3741.1, Definition 3741.2, and Definition 3741.3 seed the common-window route from UniformCompletionFunctor through StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completion points, host metric completeness, countable choice, private observation windows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common entourage-window admission, synchronized StreamName admission, RegSeqRat readback exactness, DyadicRatCore budget exactness, Real seal handoff, common-window functoriality, transport, replay, provenance, and local naming for BEDC.Derived.UniformCompletionCommonWindowUp.

Constructive story: A $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ packet is generated from one UniformCompletionFunctor request row, one common entourage-window row, one StreamName synchronized window, one RegSeqRat readback, one DyadicRatCore budget, one RealUp seal, a common-window functoriality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2772

A Concrete Naming Certificate for $\mathsf{FiniteMetricSpace}^{\uparrow}$

Following Burago–Burago–Ivanov and Gromov on finite metric spaces as completely displayed distance data, the $\mathsf{FiniteMetricSpace}^{\uparrow}$ packet records a metric carrier whose public observations are a finite point row and a finite distance table. It sits over the $\mathsf{Met}^{\uparrow}$ surface of chapter 57, the finite-net and total-bounded handoff of chapter 1767 and chapter 127, and the compact-metric consumer boundary of chapter 129. The packet names finite metric data without importing host finite sets, quotient point equality, selected centers, open-cover compactness, or ambient completion.

Definition 2772.1 (Finite metric space carrier)**.** *A* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, D, T, B, H, C, P, N)$$

with the following displayed rows:

1. *X is a finite point row, recorded as a closed* ProbeBundle *with point histories and membership witnesses for every public point read.*
2. *D is the finite distance-table row, assigning each displayed ordered pair of point histories a* $\mathsf{Real}^{\uparrow}$ *distance witness.*
3. *T is the metric-law ledger for the displayed table: zero diagonal, symmetry, triangle-budget rows, and positive-separation rows are read only as finite table entries.*
4. *B is the finite-net and boundedness handoff row. For each displayed positive tolerance it reuses the point row X as the center bundle and exposes the coverage route consumed by* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, and* $\mathsf{CompactMetric}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the point row, distance table, metric-law ledger, finite-net handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from point membership reads to distance-table reads, from distance-table reads to metric-law checks, and from the finite point row to finite-net and compact-metric consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host finite set, powerset membership, quotient point equality, selected nearest-point search, open-cover compactness, or ambient completion.

Lean4 ✓ `BEDC.Derived.FiniteMetricSpaceUp.FiniteMetricSpaceTasteGate single carrier alignment`

Theorem 2772.2 (Finite metric space NameCert obligation surface)**.** *For every accepted* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *carrier* $F = (X, D, T, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite point row, distance table, metric-law ledger, finite-net handoff, componentwise transport, displayed replay, provenance, and local naming rows; distance readback through* D*; metric compatibility through* T*; finite-net and compact-metric handoff through* B*; and nonescape from host finite sets, powerset membership, quotient point equality, selected nearest-point search, open-cover compactness, and ambient completion.*

Proof. Project the carrier of Definition 3742.1. The displayed tuple exposes the point bundle, distance table, metric-law ledger, finite-net handoff, and the structural rows H, C, P, N.

The distance readback is table-local. Every distance observation names two point histories from X and then reads the corresponding entry of D, while T supplies the finite metric-law rows attached to those entries.

The finite-net handoff is the row B. It reuses the same point row as the center bundle and sends the resulting coverage data to finite-net, total-bounded, and compact-metric consumers through the displayed replay route.

No excluded object is a coordinate of the carrier. Thus the local certificate exports only the displayed finite metric route and its structural transport, replay, provenance, and naming rows. □

Theorem 2772.3 (Finite metric space distance-table boundary)**.** *Every* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *packet reads metric information only through*

$$X \longmapsto D \longmapsto T \longmapsto B.$$

The consumer receives a finite point bundle, a displayed distance table, the metric-law ledger for that table, and the finite-net handoff generated from the same point bundle. It receives no host finite-set equality, ambient enumeration principle, selected nearest point, external compactness theorem, quotient completion, or hidden completion carrier.

Proof. Begin with a consumer request for metric or compact-metric data. The only public points are the histories displayed in X, and every distance read is routed through the table row D.

The metric-space laws are then read from T. Since T is attached to the same table entries, a consumer cannot replace the displayed table by a host metric or by a quotient comparison on points.

For finite-net and compact-metric use, the handoff row B supplies the center bundle and coverage route from X. The replay row C preserves the order from point membership to distance-table read to metric-law check to finite-net export.

The remaining rows transport, source, and name this same route. Because no carrier coordinate stores a host finite set, selected nearest point, external compactness theorem, quotient completion, or hidden completion carrier, every consumer boundary is exhausted by X, D, T, B together with H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteMetricSpace}^{\uparrow}$ packet is generated from a finite point row, a displayed Real-valued distance table, finite metric-law rows, a finite-net and compact-metric handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteMetricSpace}\uparrow}$, seed)

Scope closed: Definition 3742.1, Definition 3742.2, and Definition 3742.3 seed the finite distance-table packet and its finite-net, total-bounded, and compact-metric consumer boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteMetricSpace}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteMetricSpaceUp.FiniteMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host finite sets, quotient point equality, nearest-point selection, open-cover compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, table-local metric-law exactness, finite-net handoff stability, compact-metric boundary nonescape, and provenance exhaustion for BEDC.Derived.FiniteMetricSpaceUp.

Chapter 2773

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailBound}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailBound}^{\uparrow}$ packet records the finite tail estimate used when a regular rational Cauchy name is read as a completion-facing real datum. It consumes the observation windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic radius ledger of chapter 291, the bounded Cauchy sequence route of chapter 1522, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite tail-bound witness for completion and Real sealing; it is not a proof of ambient Real completeness, quotient equality, countable choice, or trichotomy.

Definition 2773.1 (Regular Cauchy tail-bound carrier)**.** *A* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, N, T_0, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, displaying the finite stream windows from which the tail request is read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking those windows as regular rational Cauchy data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording the dyadic tolerance and comparison rows used by the tail estimate.*
4. *N is the unary tail-index row, naming the finite stage after which the displayed radius is to be enforced.*
5. *T_0 is the later-index comparison row, recording the finite comparisons that put every displayed later observation beyond N and inside the dyadic radius from D.*
6. *H records componentwise* hsame *transport for the stream, regular readback, dyadic radius, tail index, later-index comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through*

$$S, R, D, N, T_0 \longmapsto H, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, R, D, N, T_0, H, C, P, L$. It contains no coordinate for a quotient completion equality, selected limit, countable-choice sequence, trichotomy decision, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailBoundUp.RegularCauchyTailBoundTasteGate single carrier alignment`

Definition 2773.2 (Regular Cauchy tail-bound classifier)**.** *Two accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *carriers*

$$T = (S, R, D, N, T_0, H, C, P, L)$$

and

$$T' = (S', R', D', N', T_0', H', C', P', L')$$

are classified together exactly when the displayed hsame *rows transport S to S', R to R', D to D', N to N', and T_0 to T_0', and when the displayed* Cont *rows replay the same stream-to-readback-to-radius-to-tail-index route on both packets. The classifier has no coordinate for host real equality, completion quotient equality, selected representatives, or order trichotomy.*

Theorem 2773.3 (Regular Cauchy tail-bound NameCert obligations)**.** *For every accepted* RegularCauchyTailBound *carrier*

$$T = (S, R, D, N, T_0, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, StreamName observation exposure through S, RegSeqRat readback exposure through R, dyadic radius accounting through D, unary tail-index exposure through N, later-index comparison through T_0, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 3743.1. The accepted packet displays the stream observation row, regular rational readback row, dyadic radius ledger, unary tail index, later-index comparison row, transport row, replay row, provenance row, and local name.

The tail estimate is read in order. A consumer first opens S and R, then reads the dyadic tolerance in D, and only then consumes the finite index row N and the later-index comparisons in T_0.

The structural rows preserve that route. The hsame and Cont rows transport and replay the same finite observations, while P and L source and name the packet. Since no coordinate stores ambient Real completeness, quotient equality, countable choice, selected limits, or trichotomy, those principles are outside the local certificate surface. □

Theorem 2773.4 (Regular Cauchy tail-bound completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, regular-completion, or bounded-Cauchy consumer of an accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CauchySequenceBounded}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff supplies a displayed tail-bound packet to completion-facing chapters, but it does not export ambient completeness, countable choice, quotient equality of Real names, selected representatives, or trichotomy.

Proof. Apply Definition 3743.3. It forces every tail-bound read through S, R, D, N, T_0 before the structural rows can be consumed.

The completion-facing route is finite. Stream windows are repacked as regular rational readback, the dyadic row supplies the radius, and the unary index plus later-index comparisons form the displayed tail witness.

Finally inspect the missing coordinates of the carrier and classifier. They preserve the finite route under transport and replay, but they contain no selected limit, quotient equality, countable-choice selector, trichotomy row, or ambient completeness principle. Thus downstream Real and completion packets receive only the finite tail-bound handoff. □

Theorem 2773.5 (Regular Cauchy tail-bound RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *reader of an accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *packet must read the* $\mathsf{StreamName}^{\uparrow}$ *observation row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger D, the unary tail-index row N, and the later-index comparison row T_0 before any* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal is consumed. The handoff keeps the Real seal downstream and exports no quotient streams, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Specialize Definition 3743.3 to a RegSeqRat-facing reader. The obligations expose S, R, D, N, T_0 before the transport, replay, provenance, and local naming rows can be used.

Use the completion handoff of Definition 3743.4 only at the boundary where a Real or completion-facing consumer appears. Before that boundary, the reader sees the stream observation, regular rational readback, radius ledger, tail index, and later-index comparisons as finite rows.

Thus the narrower handoff is the RegSeqRat portion of the same packet. The downstream Real seal cannot be used to supply quotient streams, selected limits, countable choice, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3743.1, Definition 3743.2, Definition 3743.3, Definition 3743.4, and Definition 3743.5 seed the finite regular-Cauchy tail-bound packet over stream observations, regular rational readback, dyadic radii, unary tail indices, later-index comparisons, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, countable choice, quotient equality, trichotomy, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, tail-index exposure, later-index comparison, dyadic-radius accounting, replay stability, and ledger nonescape for BEDC.Derived.RegularCauchyTailBoundUp.

Constructive story: A $\mathsf{RegularCauchyTailBound}^{\uparrow}$ packet is generated from one StreamName observation row, one RegSeqRat readback row, one DyadicRatCore radius ledger, one unary tail-index row, one later-index comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2774

A Concrete Naming Certificate for HochsterNerveBettiReduction$^{\uparrow}$

The HochsterNerveBettiReduction$^{\uparrow}$ packet realizes the Hochster–nerve reduction of subsection 9462.28.21 as a finite NameCert surface. It turns the vision theorem Definition 9462.1639 into displayed support rows, nerve-face rows, homology-range rows, Betti-strand rows, and ledger rows. The packet is a sibling of the homology carrier of chapter 98, the finite simplicial-chain carrier of chapter 3414, and the simplicial approximation surface of chapter 3402. It names a finite algebraic-topology reduction; it does not assert field faithfulness, nontriviality of coefficients, unrestricted minimal-resolution construction, or a Lean-checked homology computation.

Definition 2774.1 (Hochster nerve Betti reduction carrier)**.** *A* HochsterNerveBettiReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (V, F, N, H_0, B_0, G, T, P, L)$$

with the following displayed rows:

1. *V is the finite support row, recording the displayed vertex subset and the local support window read by Hochster's formula.*
2. *F is the nerve-face row, recording the finite face and nonface tests that produce the nerve complex used by the reduction.*
3. *N is the homology-range row, recording the displayed dimensions in which reduced homology can be read.*
4. *H_0 is the finite homology row, read through the* Homology$^{\uparrow}$ *and* FiniteSimplicialChain$^{\uparrow}$ *carriers rather than through an ambient chain complex.*
5. *B_0 is the Betti-strand row, recording the diagonal and top-strand entries exposed by the support and homology rows.*
6. *G records componentwise* hsame *transport for the support, nerve-face, homology-range, finite homology, Betti-strand, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from finite support and nerve faces to homology ranges and then to Betti-strand ledgers.*
8. *P is the* Pkg *provenance over* HochsterNerveBettiReduction$^{\uparrow}$, Homology$^{\uparrow}$, FiniteSimplicialChain$^{\uparrow}$, SimplicialApproximation$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. L is the local $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $V, F, N, H_0, B_0, G, T, P, L$. It has no coordinate for a field-faithfulness theorem, coefficient nontriviality instance, unrestricted free resolution, or host-level equality of Betti tables.

Theorem 2774.2 (Hochster nerve Betti reduction obligations)**.** *For every accepted* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *carrier*

$$B = (V, F, N, H_0, B_0, G, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$ *surface consists exactly of finite support exposure through V, nerve-face exposure through F, homology-range exposure through N, finite homology readback through H_0, Betti-strand ledger exposure through B_0, componentwise transport through G, displayed replay through T, package provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.HochsterNerveBettiReductionUp.HochsterNerveBettiReductionNameCertObligations`

Proof. Project the carrier of Definition 3744.1. The finite support, nerve-face, homology-range, finite homology, Betti-strand, transport, replay, provenance, and local name rows exhaust the accepted packet.

The reduction is read along the finite route. The support row V fixes the displayed vertex window, the face row F builds the nerve complex, and the range row N restricts the homology degrees that can be queried.

The Betti ledger is downstream of finite homology. The row H_0 is read through homology and finite simplicial-chain carriers, and B_0 records only the strand entries obtained from that range. The structural rows G, T, P, L transport, replay, source, and name the same finite route, so no field-faithfulness theorem, nontrivial coefficient instance, unrestricted resolution, or host Betti-table equality is exported. □

Theorem 2774.3 (Hochster nerve Betti reduction TasteGate boundary)**.** *A downstream consumer may promote a* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *packet to a checked Betti computation only after future* Nontrivial *and* FieldFaithful *TasteGate instances are displayed for the coefficient surface. Before those instances are present, the packet exports only the finite support, nerve-face, homology-range, Betti-strand, transport, replay, provenance, and local naming rows of Definition 3744.2.*

Proof. Read the NameCert obligations first. They expose $V, F, N, H_0, B_0, G, T, P, L$ and no coefficient-faithfulness or nontriviality coordinate.

The TasteGate instances are therefore separate inputs. A consumer that wants a field-faithful Betti computation must supply those coefficient gates before interpreting the finite Betti ledger as a checked algebraic-topology calculation.

Without those gates, the accepted reading remains the finite reduction surface. It may cite the Hochster–nerve vision route and the sibling homology carriers, but it cannot claim unrestricted free resolutions, host-level Betti-table equality, or Lean-checked homology arithmetic. □

Falsifiable prediction: Every $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ read factors through one finite support row, one nerve-face row, one homology-range row, one finite homology row, one Betti-strand ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: homology, finite_simplicial_chain, simplicial_approximation, dna_window_six_bound alexander_dual_trigger_algebra: $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries support windows, nerve-face tests, homology-range restriction, Betti-strand ledgers, and the coefficient TasteGate boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3744.1, Definition 3744.2, and Definition 3744.3 seed the finite Hochster–nerve Betti-reduction packet over support rows, nerve-face rows, homology-range rows, finite homology rows, Betti-strand ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HochsterNerveBettiReductionUp.HochsterNerveBettiReductionNameCertObligations`

Bridge status: none

Not claimed: This chapter does not claim unrestricted free resolutions, host Betti-table equality, or bridge checking.

Upgrade path: Obligation closure requires checked finite support exposure, nerve-face exposure, homology-range exposure, finite homology readback, Betti-strand ledger stability, coefficient TasteGate instances, replay stability, and ledger nonescape for BEDC.Derived.HochsterNerveBettiReductionUp.

Constructive story: A $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ packet is generated from a finite support row, a nerve-face row, a homology-range row, a finite homology row, a Betti-strand row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2775

A Concrete Naming Certificate for $\mathsf{BrouwerDegree}^{\uparrow}$

Following Brouwer and standard constructive degree-theory presentations, the $\mathsf{BrouwerDegree}^{\uparrow}$ packet records the finite boundary ledger by which a displayed continuous map on a compact domain is read as an integer degree certificate. It sits between the circle boundary surface of chapter 58, the compact-domain ledger of chapter 60, the graph and modulus rows of chapter 130, the chain-boundary route of chapter 98, and the terminal real-name seal of chapter 38. The packet names degree data only as finite BHist rows; it does not claim the classical Brouwer fixed point theorem, host topological degree, selected preimages, quotient homology classes, or ambient compactness.

Definition 2775.1 (Brouwer degree carrier)**.** *A* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, B, F, O, I, H, T, C, P, N)$$

with the following displayed rows:

1. *K is the compact-domain row, read through the finite compact or compact-metric certificate surface.*
2. *B is the boundary row, with its $\mathsf{S}^{1\uparrow}$ orientation circle ledger and the displayed boundary inclusion into K.*
3. *F is the $\mathsf{ContinuousMap}^{\uparrow}$ row whose graph, modulus, and boundary restriction are visible to the packet.*
4. *O is the orientation ledger, recording the finite sign, face-order, or winding-window data used by the boundary read.*
5. *I is the integer-degree readback row, storing the displayed $\mathbb{Z}$-valued degree output and its finite comparison classifier.*
6. *H is the $\mathsf{Homology}^{\uparrow}$ boundary row, exposing the finite chain-boundary or cycle-class route consumed by the degree readback.*
7. *T is the componentwise hsame transport row for the compact-domain, boundary, continuous-map, orientation, integer-readback, and homology-boundary rows.*
8. *C is the displayed Cont replay from K, B, F, O, H to the integer readback row I and then to the local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, F, O, I, H, T, C, P, N$. It has no coordinate for a host degree function, selected zeros or preimages, quotient homology classes, open-cover compactness, or a Brouwer fixed-point conclusion.

Lean4 ✓ `BEDC.Derived.BrouwerDegreeUp.BrouwerDegreeTasteGate single carrier alignment`

Theorem 2775.2 (Brouwer degree NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier*

$$D = (K, B, F, O, I, H, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-domain row K, boundary and* $\mathsf{S}^{1\uparrow}$ *row B, continuous-map row F, orientation ledger O, integer-degree readback I, homology boundary row H, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N. Every degree consumer must read the boundary, orientation, and homology rows before the integer row is exported.*

Proof. Project the accepted carrier. The compact-domain row K, boundary row B, and continuous-map row F are the only geometric inputs, and each is already a displayed finite BEDC row.

The degree-specific data are O, I, H. The orientation ledger O supplies the finite boundary orientation, the homology boundary row H supplies the finite chain-boundary route, and the integer row I is read only after those two rows and the continuous-map boundary restriction have been displayed.

The structural rows T, C, P, N transport, replay, source, and name that same route. Since no carrier coordinate stores host topological degree, selected preimages, quotient homology classes, open-cover compactness, or fixed-point existence, the listed rows exhaust the local NameCert obligations. □

Theorem 2775.3 (Brouwer degree boundary handoff)**.** *Every boundary-facing degree read from an accepted* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier factors through*

$$K, B, F, O, H \longmapsto I \longmapsto T, C, P, N.$$

Thus the compact domain, boundary $\mathsf{S}^{1\uparrow}$ *row, continuous-map restriction, orientation ledger, and homology boundary row are displayed before the integer-degree readback can be transported or named.*

The handoff exports only this finite boundary route. It does not assert the Brouwer fixed point theorem, a host topological-degree functor, selected zeros, selected preimages, quotient homology classes, or ambient compactness.

Proof. Start with the NameCert obligations of Definition 3745.2. They place the compact-domain, boundary, continuous-map, orientation, and homology rows before the integer readback row.

The boundary handoff is therefore forced through K, B, F, O, H. The row B supplies the $\mathsf{S}^{1\uparrow}$ boundary ledger, F supplies the displayed map restriction, O records the orientation data, and H records the homology boundary route consumed by the integer row I.

Finally replay the structural rows T, C, P, N. These rows preserve and name the same finite route, so the exported degree surface has no additional coordinate for host degree theory, fixed-point existence, selected preimages, quotient homology, or ambient compactness. □

Falsifiable prediction: A $\mathsf{BrouwerDegree}^{\uparrow}$ consumer that exports an integer degree without reading the boundary row B, orientation ledger O, and homology boundary row H refutes the carrier surface.

Independence witness: s1, compact, continuousmap, homology, real: the sibling chapters supply boundary circles, compact rows, graph and modulus rows, chain-boundary routes, and real seals separately, while this chapter binds those rows into the finite degree readback packet $D = (K, B, F, O, I, H, T, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerDegree}^{\uparrow}$ packet is generated from a compact-domain row, boundary and SOne row, ContinuousMap row, orientation ledger, integer-degree readback, Homology boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$, seed)

Scope closed: Definition 3745.1, Definition 3745.2, and Definition 3745.3 seed the finite compact-domain, boundary, continuous-map, orientation, homology-boundary, integer-readback, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the Brouwer fixed point theorem, host topological degree, selected zeros, selected preimages, quotient homology classes, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary and orientation exactness, continuous-map restriction stability, homology boundary handoff, integer readback stability, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerDegreeUp.

Chapter 2776

A Concrete Naming Certificate for $\mathsf{Metacompact}^{\uparrow}$

Following Morita and Kelley general topology, the $\mathsf{Metacompact}^{\uparrow}$ packet records the point-finite refinement surface between paracompact locally finite refinement, Moore-space developments, and star-refinement cover ledgers. It sits after the paracompact cover surface of chapter 1963, beside the developable-neighbourhood rows of chapter 3592, and before the uniform star-refinement handoff of chapter 2167. The packet names only a displayed open-cover request, a point-finite open refinement, and the ledgers that certify coverage and point incidence; it does not choose arbitrary subcovers, infer local finiteness, or import compactness.

Definition 2776.1 (Metacompact open-cover carrier)**.** *A* $\mathsf{Metacompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, O, R, Q, I, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row containing the open predicates, point observations, and neighbourhood classifier used by the packet.*
2. *O is the open-cover request row, listing the displayed cover cells and the pointwise coverage ledger for T.*
3. *R is the open-refinement row, listing the displayed refinement cells and their containment witnesses back into cells of O.*
4. *Q is the point-finite incidence row, recording for each displayed point observation the finite list of refinement cells that contain it.*
5. *I is the coverage-and-incidence exactness row, requiring every point covered by O to be covered by a refinement cell and requiring every incidence read to be one of the finite entries in Q.*
6. *S is the sibling handoff row to the* $\mathsf{Paracompact}^{\uparrow}$*,* $\mathsf{MooreSpace}^{\uparrow}$*, and* $\mathsf{StarRefinementUniformCover}^{\uparrow}$ *cover ledgers.*
7. *H records componentwise* hsame *transport for $T, O, R, Q, I, S, C, P, N$.*
8. *C records displayed* Cont *replay from an open-cover request through refinement, point-finite incidence, sibling handoff, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{Metacompact}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Paracompact}^{\uparrow}$*,* $\mathsf{MooreSpace}^{\uparrow}$*,* $\mathsf{StarRefinementUniformCov}$ BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}$ *row.*
The carrier contains no coordinate for selected subcovers, locally finite partitions, metric development extraction, uniform-star generation, or ambient compactness.

Lean4 ✓ `BEDC.Derived.MetacompactUp.MetacompactTasteGate single carrier alignment`

Definition 2776.2 (Metacompact classifier)**.** *The* $\mathsf{Metacompact}^{\uparrow}$ *classifier accepts exactly those displayed packets $M = (T, O, R, Q, I, S, H, C, P, N)$ whose refinement row R is open in T, whose containment ledger maps each refinement cell into a displayed cell of O, whose coverage ledger in I covers the same point observations as O, and whose incidence row Q displays a finite list of refinement cells at each point observation. A classifier read is rejected if it asks for a subcover selection, a locally finite refinement, a Moore development, a uniform star refinement, or a compactness principle not present among $T, O, R, Q, I, S, H, C, P, N$.*

Theorem 2776.3 (Metacompact carrier admission)**.** *For every accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet $M = (T, O, R, Q, I, S, H, C, P, N)$, carrier admission exposes exactly the topology source, open-cover request, open-refinement row, point-finite incidence row, coverage-and-incidence exactness row, sibling handoff row, transport, replay, provenance, and local naming rows of Definition 3746.1.*

Proof. Project the carrier definition. The tuple lists $T, O, R, Q, I, S, H, C, P, N$ and no further coordinate.

The classifier of Definition 3746.2 accepts only packets that read refinement, containment, coverage, point-finite incidence, sibling handoff, transport, replay, provenance, and local naming through those coordinates. Therefore carrier admission is the displayed finite packet itself. □

Theorem 2776.4 (Metacompact point-finite refinement exactness)**.** *For every accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet $M = (T, O, R, Q, I, S, H, C, P, N)$, the row R is an open refinement of the request row O, and the row Q is a point-finite incidence ledger for R. Every displayed point covered by O is covered by at least one row of R, and every refinement cell containing that point appears in the finite list recorded by Q.*

Proof. Use the refinement part of the classifier. It requires the cells of R to be open in T and to carry containment witnesses into the displayed cover cells of O.

Use the exactness row I next. It links coverage by O to coverage by R and restricts incidence reads to the finite lists displayed in Q.

Thus the point-finite refinement is not inferred from a choice principle. It is stored as the finite incidence ledger attached to the accepted packet. □

Theorem 2776.5 (Metacompact sibling cover handoff)**.** *Every cover-refinement consumer of an accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet factors through the sibling handoff row S before citing* $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{MooreSpace}^{\uparrow}$, *or* $\mathsf{StarRefinementUniformCover}^{\uparrow}$. *The handoff may compare locally finite, development, and star-refinement ledgers with the point-finite row Q, but it cannot promote the metacompact packet to any of those stronger cover principles.*

Proof. Read the sibling coordinate S in the carrier. It is the only row that mentions the paracompact, Moore-space, and star-refinement neighbours.

The refinement and incidence rows remain R, Q, I. Comparing them with sibling cover ledgers does not add locally finite partitions, countable developments, or uniform-star generation to the packet.

Therefore any downstream citation of the neighbours passes through S as a handoff and returns to the same point-finite refinement surface. □

Theorem 2776.6 (Metacompact transport stability)**.** *Transporting an accepted* $\mathsf{Metacompact}^{\uparrow}$ *read along H preserves the topology source T, open-cover request O, refinement row R, point-finite incidence row Q, exactness row I, sibling handoff S, replay row C, provenance P, and local naming row N. The transported classifier still reads only the displayed point-finite refinement packet.*

Proof. The row H is componentwise by definition of the carrier. It transports the same open-cover request, refinement cells, point-finite incidence lists, exactness row, and sibling handoff.

The classifier has no alternate coordinate for cover selection or compactness. After transport, the accepted read is still checked by the same rows $T, O, R, Q, I, S, C, P, N$ and by no hidden cover operation. □

Theorem 2776.7 (Metacompact nonescape)**.** *No consumer of an accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet can read selected subcovers, locally finite partitions of unity, Moore developments, uniform star-refinement generation, ambient compactness, or standard bridge data from the packet. All consumer reads are confined to $T, O, R, Q, I, S, H, C, P, N$.*

Proof. Inspect the carrier coordinates. The only cover-theoretic data are the open-cover request O, open refinement R, point-finite incidence row Q, exactness row I, and sibling handoff S.

Transport, replay, provenance, and naming do not add cover choices. They preserve and expose the same finite rows already present in the carrier.

The excluded stronger objects have no projection from $T, O, R, Q, I, S, H, C, P, N$, so a consumer cannot extract them from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3746.1, Definition 3746.2, and Definition 3746.4 seed the finite point-finite open-refinement packet over topology source, open-cover request, refinement, incidence, exactness, sibling handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, local finiteness, partitions of unity, Moore-space development extraction, uniform star-refinement generation, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, point-finite refinement exactness, sibling cover handoff, transport stability, and nonescape for BEDC.Derived.MetacompactUp.

Constructive story: A $\mathsf{Metacompact}^{\uparrow}$ packet is generated from a topology source row, open-cover request row, open-refinement row, point-finite incidence ledger, coverage-and-incidence exactness row, sibling cover handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2777

A Concrete Naming Certificate for FiniteRecognizerCarrierObstruction$^{\uparrow}$

The FiniteRecognizerCarrierObstruction$^{\uparrow}$ packet realizes the recognizer-to-carrier boundary in chapter 9379 as a finite seven-row NameCert obstruction. It sits beside the recognizer-flow packet of chapter 1048, the finite reflection tuple of chapter 993, the normalization certificate router of chapter 1108, and the inductive-acceptance packet of chapter 915. The packet names the finite landing surface that must be present before a recognizer-derived carrier can even be compared with a standard-bridge question.

Definition 2777.1 (FiniteRecognizerCarrierObstruction carrier)**.** *A* FiniteRecognizerCarrierObstruction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, G, Q, F, R, T, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the recognizer-acceptance row consumed from* GroundCompilerRecognizerFlow$^{\uparrow}$*.*
2. *G is the generated-recognizer mediation row, recording that the accepted recognizer is the generated object read by the compiler-facing flow.*
3. *Q is the certificate-gate row, recording that the recognizer output has passed the displayed gate before any carrier comparison is attempted.*
4. *F is the finite tuple extraction row consumed through* FiniteReflectionTuple$^{\uparrow}$*.*
5. *R is the component readback row, exposing the* BHist *components recovered from the finite tuple image.*
6. *T is the componentwise* hsame *and* NameCert *transport row for the recovered components.*
7. *K is the* ChapterTasteGate *row for the displayed carrier-facing surface.*
8. *H records componentwise* hsame *transport for $A, G, Q, F, R, T, K, C, P, N$.*
9. *C records displayed* Cont *replay from recognizer acceptance through generated-recognizer mediation, certificate gate, tuple extraction, readback, componentwise transport,* ChapterTasteGate*, provenance, and local naming.*
10. *P is the* Pkg *provenance over* FiniteRecognizerCarrierObstruction$^{\uparrow}$*,* GroundCompilerRecognizerFlow$^{\uparrow}$*,* FiniteReflectionTuple$^{\uparrow}$*,* MetaCICNormalizationCertificateRouter$^{\uparrow}$*,* KernelInductiveAcceptance$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and* ChapterTasteGate *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}}$ *row.*
The carrier does not construct a recognizer-to-carrier map, does not assert unrestricted beta-conversion, and does not supply a StandardBridge *theorem. It only records the finite landing rows that a later construction must expose before those questions are reached.*

Lean4 (definition) `BEDC.Derived.FiniteRecognizerCarrierObstructionUp.FiniteRecognizerCarrierObstructionUp`

Theorem 2777.2 (FiniteRecognizerCarrierObstruction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *carrier* $O = (A, G, Q, F, R, T, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}}$ *surface consists exactly of recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport,* ChapterTasteGate *exposure, componentwise transport, replay, provenance, and local naming.*

Proof. Project the carrier. The first seven rows A, G, Q, F, R, T, K are the recognizer-to-carrier landing surface, while H, C, P, N are the structural rows that preserve and expose that surface.

Each obligation is read from its displayed row. Acceptance comes from A, generated-recognizer mediation from G, gate admission from Q, tuple extraction from F, readback from R, componentwise transport from T, and chapter-facing gate exposure from K.

No obligation row supplies a recognizer-to-carrier map or a standard bridge. The packet therefore isolates the obstruction boundary rather than resolving it. □

Theorem 2777.3 (FiniteRecognizerCarrierObstruction missing-row refusal)**.** *An attempted recognizer-to-carrier read that omits any one of A, G, Q, F, R, T, K fails inside the* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *packet before the* StandardBridge *question is reached. The failure is a missing landing row, not a failed bridge theorem.*

Proof. Use the obligation surface of Definition 3747.2. It lists the seven landing rows as jointly required before the carrier-facing surface is displayed.

If one row is absent, the replay route C cannot traverse the full path from recognizer acceptance to ChapterTasteGate. The packet then records the absence as an obstruction at the landing surface.

Since the packet has no StandardBridge theorem coordinate, such a failure occurs before bridge comparison and cannot be read as a bridge counterexample. □

Theorem 2777.4 (FiniteRecognizerCarrierObstruction seven-row landing)**.** *If a packet displays recognizer acceptance A, generated-recognizer mediation G, certificate-gate admission Q, finite tuple extraction F, component readback R, componentwise* hsame *and* NameCert *transport T, and* ChapterTasteGate *exposure K, together with the structural transport, replay, provenance, and local naming rows of Definition 3747.1, then the packet lands on the* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *surface before any* StandardBridge *question is evaluated.*

Proof. Project the carrier of Definition 3747.1. Its first seven displayed coordinates are exactly A, G, Q, F, R, T, K, and the remaining rows H, C, P, N preserve, replay, source, and name that same landing surface.

Apply Definition 3747.2. The obligation inventory recognizes the seven rows as the complete recognizer-to-carrier landing face, so a packet displaying all of them has reached the obstruction surface rather than a later bridge surface.

This is the positive side of Definition 3747.3. Omission of one landing row refuses the read, while display of all seven rows admits the finite obstruction surface. Neither direction constructs a recognizer-to-carrier map or closes a StandardBridge theorem. □

Theorem 2777.5 (FiniteRecognizerCarrierObstruction sibling nonbijection)**.** *The carrier of Definition 3747.1 is not bijective, row for row, with* GroundCompilerRecognizerFlow$^{\uparrow}$, FiniteReflectionTuple$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, *or* KernelInductiveAcceptance$^{\uparrow}$. *Any row-preserving bijection would have to preserve recognizer acceptance, generated-recognizer mediation, certificate-gate passage, tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *exposure at once, and no listed sibling carries that whole surface.*

Proof. Compare first with GroundCompilerRecognizerFlow$^{\uparrow}$. It supplies recognizer acceptance, generated-recognizer mediation, and certificate-gate rows, but it does not carry finite tuple extraction, tuple readback, componentwise NameCert transport, and ChapterTasteGate exposure as one carrier.

Compare next with FiniteReflectionTuple$^{\uparrow}$. It supplies tuple image and readback rows, but it does not carry recognizer acceptance, generated-recognizer mediation, or certificate-gate passage.

The normalization router and inductive-acceptance packet have different endpoints. They record normalization evidence routing or inductive declaration acceptance, not the simultaneous seven-row recognizer-to-carrier landing surface. Hence no row-preserving bijection can identify them with this obstruction packet. □

Theorem 2777.6 (FiniteRecognizerCarrierObstruction TasteGate follow-up surface)**.** *Any Lean-side* Nontrivial *and* FieldFaithful *TasteGate instance for* FiniteRecognizerCarrierObstruction$^{\uparrow}$ *must preserve the displayed carrier rows* $A, G, Q, F, R, T, K, H, C, P, N$. *Nontriviality separates a complete seven-row landing from a missing-row obstruction, and field faithfulness requires every accepted read to retain each of the seven landing rows componentwise.*

Proof. The nontrivial boundary is visible in the carrier. A complete landing displays A, G, Q, F, R, T, K, while a failed landing omits at least one of those rows and is refused by Definition 3747.3.

Field faithfulness is componentwise. Preserving the recognizer-to-carrier obstruction requires preserving acceptance, generated recognizer, gate, tuple extraction, readback, componentwise transport, and ChapterTasteGate rows together.

Thus the later Lean instances must consume the same rows as the paper carrier and may not replace them by a vacuous or report-only encoding. □

Falsifiable prediction: Any attempted recognizer-to-carrier map missing recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, or ChapterTasteGate exposure fails before the StandardBridge question is reached.

Independence witness: groundcompilerrecognizerflow, finitereflectiontuple, metacicnormalizationcertificaterouter, kernelinductiveacceptance: GroundCompilerRecognizerFlow$^{\uparrow}$ records recognizer acceptance, generated-recognizer mediation, and certificate-gate passage without tuple extraction and readback; FiniteReflectionTuple$^{\uparrow}$ records tuple image and readback without recognizer-flow acceptance; MetaCICNormalizationCertificateRouter$^{\uparrow}$ routes normalization evidence without the seven-row recognizer landing surface; KernelInductiveAcceptance$^{\uparrow}$ records inductive declaration acceptance without recognizer-to-carrier landing. None is bijective to FiniteRecognizerCarrierObstruction$^{\uparrow}$.

Closure status for NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$, seed)

Scope closed: Definition 3747.1, Definition 3747.2, and Definition 3747.3 seed the seven-row recognizer-to-carrier obstruction packet over recognizer acceptance, generated recognizer, certificate gate, finite tuple extraction, component readback, componentwise transport, ChapterTasteGate, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a recognizer-to-carrier construction, unrestricted beta-conversion, standard bridge closure, Lean verification, or global MetaCIC normalization.

Upgrade path: Obligation closure requires checked carrier admission, seven-row landing exactness, missing-row refusal, sibling nonbijection, TasteGate nontriviality, and field faithfulness for BEDC.Derived.FiniteRecognizerCarrierObstructionUp.

Constructive story: A FiniteRecognizerCarrierObstruction$^{\uparrow}$ packet is generated from recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, ChapterTasteGate exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 2778

A Concrete Naming Certificate for $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy reals, the $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ packet records the finite naming surface for the canonical modulus read from a regular rational Cauchy presentation. It sits after the dyadic tolerance rows of chapter 291, the finite window rows of chapter 126, the regular rational sequence carrier of chapter 301, and the real seal of chapter 38. The chapter names the displayed modulus object needed by the L10 RegSeqRat-to-Real route without importing an ambient Cauchy-search principle.

Definition 2778.1 (Regular Cauchy canonical modulus carrier)**.** *A* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic radii, denominator windows, and scale comparison data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the observed rational stream windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row recording the Cauchy bound read from the same windows.*
4. *Q is the canonical modulus row, assigning to each displayed tolerance scale the finite window at which the regularity row certifies stability.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached by the regular rational presentation.*
6. *H records componentwise* hsame *transport for the dyadic, stream, regularity, modulus, endpoint, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a modulus request through D, S, R, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded search over Cauchy witnesses, quotient equality of real presentations, countable choice, a selected limit outside E, or a host completeness theorem.

Lean4 (definition) `BEDC.Derived.RegularCauchyCanonicalModulusUp.RegularCauchyCanonicalModulusUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyCanonicalModulusUp.RegularCauchyCanonicalModulusTasteGate`
`single carrier alignment`

Theorem 2778.2 (Regular Cauchy canonical modulus NameCert obligations)**.** *For the carrier of Definition 3748.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance row D, stream-window row S, regular rational row R, canonical modulus row Q, real endpoint row E, transport row H, replay row C, provenance row P, and local naming row N;*
2. *canonical-modulus exactness, requiring each tolerance scale read from D to be assigned by Q to a finite stream window already present in S and certified by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
3. *classifier stability, requiring two accepted packets to compare only through componentwise* hsame *transport of D, S, R, Q, E, P, N;*
4. *endpoint nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to reach E only after the displayed dyadic, stream, regularity, and modulus rows have been read;*
5. *provenance and naming exactness, requiring P and N to record the same finite RegSeqRat-to-Real route.*

Proof. Carrier admission is projection from the displayed packet $K = (D, S, R, Q, E, H, C, P, N)$. Each row is finite BHist data, and no row names an ambient search operator.

Canonical-modulus exactness reads Q against the tolerance row D and stream row S. The regularity row R supplies the bounded Cauchy witness for the displayed finite window, so the modulus is canonical only relative to those already named rows.

For classifier stability, compare accepted packets componentwise through H. The dyadic, stream, regularity, modulus, endpoint, provenance, and local naming rows are transported together, and replay through C preserves the same route.

Finally inspect a $\mathsf{Real}^{\uparrow}$ consumer. It can reach the endpoint row E only through D, S, R, Q and the structural rows H, C, P, N, so quotient equality, countable choice, unbounded search, and host completeness are outside the NameCert surface. □

Theorem 2778.3 (Regular Cauchy canonical modulus RegSeqRat handoff)**.** *Every accepted* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *read that cites a* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyCanonicalModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff first reads dyadic tolerance scales, then finite stream windows, then the regular rational Cauchy row, then the canonical modulus row, and only then the real endpoint seal. It supplies no independent limit selector, no quotient equality of presentations, no unbounded Cauchy-search oracle, and no ambient real-completeness argument.

Proof. Start with a RegSeqRat-to-Real consumer. The carrier forces the request to read D and S before the regularity row R can be interpreted, because the Cauchy bound is attached to dyadic scales and finite stream windows.

Next read the canonical modulus row Q. Its entries are finite assignments from displayed tolerance scales to displayed windows, with R certifying regularity on those windows.

The endpoint row E is then reached only through the same finite route. The structural rows H, C, P, N transport, replay, source, and name that route, so no absent search, quotient, selector, or completion principle can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ read factors through displayed DyadicRatCoreUp tolerance rows, StreamName finite windows, RegSeqRat regularity rows, a canonical modulus row, a RealUp endpoint seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance scales, finite stream windows, regular rational Cauchy bounds, a canonical modulus assignment, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ packet is generated from dyadic tolerance scales, finite StreamName windows, RegSeqRat regularity rows, a canonical modulus assignment, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3748.1, Definition 3748.2, and Definition 3748.3 seed the finite canonical-modulus route from RegSeqRat data to the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of real presentations, countable choice, unbounded Cauchy search, selected limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, canonical-modulus exactness, classifier stability, endpoint nonescape, and RegSeqRat handoff for the displayed packet.

Chapter 2779

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$

Following the Bishop–Bridges constructive-completion treatment, the $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ packet records the finite stability surface by which a regular rational Cauchy name, its finite stream windows, dyadic tolerance ledger, terminal real seal, and completion-witness row remain attached to one completion route. It sits after the regular rational sequence interface of chapter 301, the stream-window naming surface of chapter 126, the dyadic rational core of chapter 291, the real-name seal of chapter 38, and the completion witness boundary of chapter 3579. The packet names finite stability and idempotence-boundary data for the displayed route; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of streams, selected limits, countable choice, or an external completion theorem.

Definition 2779.1 (Regular Cauchy completion stability carrier)**.** *A* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational Cauchy name and its finite tolerance schedule.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that selects the finite source observations inspected by the completion route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison rows used to certify regularity on the selected windows.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular sequence, stream windows, and dyadic tolerance ledger.*
5. *E is the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *row, exposing only the displayed completion-witness surface attached to Q, W, D, R.*
6. *H records componentwise* hsame *transport for the regular sequence, stream windows, dyadic ledger, real seal, completion witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from Q through W, D, R, E and back to the local naming row.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a chosen limit representative, ambient completeness of $\mathsf{Real}^{\uparrow}$*, countable choice, trichotomy, or a second completion route.*

Theorem 2779.2 (Regular Cauchy completion stability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier $S = (Q, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCom}}$ *surface consists exactly of carrier admission for the displayed rows; regular-source exposure through Q; finite window selection through W; dyadic tolerance accounting through D; real-seal confinement through R; completion-witness attachment through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionStabilityUp.RegularCauchyCompletionStabilityNameCertObligat`

Proof. Carrier admission is projection from Definition 3749.1. The accepted packet displays exactly the regular rational source, the finite window schedule, the dyadic tolerance ledger, the terminal real seal, and the completion-witness row.

The stability route is read in source order. The row Q supplies the regular Cauchy name, W selects the finite observations that a consumer may inspect, and D records the dyadic tolerance comparisons visible on those observations.

The rows R and E then bind the completion-facing read. The real seal R is reached only after Q, W, D have been displayed, and the completion witness E attaches to that same route rather than to an external completeness principle.

Finally H, C, P, N transport, replay, source, and name the route. Since no coordinate stores quotient stream equality, a chosen limit representative, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, trichotomy, or a second completion route, the listed rows exhaust the local NameCert obligations. □

Theorem 2779.3 (Regular Cauchy completion idempotence boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletenessWitness}^{\uparrow}.$$

If the consumer asks to complete a route already sealed by R and witnessed by E, the packet can only replay the same displayed rows $Q, W, D, R, E, H, C, P, N$. It cannot introduce a quotient of streams, a selected representative of the limit, a second real seal, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionStabilityUp.RegularCauchyCompletionStabilityCarrier idempotence boundary`

Proof. Start with a completion-facing read of the carrier. By Definition 3749.1, the only source of regular Cauchy data is Q, and every finite observation passes through the stream window row W and dyadic tolerance ledger D.

The completion read reaches R and E only along this route. Thus the real seal and the completion witness are already attached to the finite regular-source path before any consumer receives the packet.

For a second completion request, replay C and transport along H. The replay has no coordinate that could manufacture a quotient stream, chosen limit, second real seal, or ambient completeness proof, so the request remains on $Q, W, D, R, E, H, C, P, N$. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ consumer that exports a completion witness without first displaying the RegSeqRat source, StreamName windows, DyadicRatCore tolerance ledger, and RealUp seal refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchy-completeness-witness: these sibling chapters supply regular schedules, finite windows, dyadic tolerance rows, real seals, and completion witnesses separately, while this packet binds them into the finite stability carrier $S = (Q, W, D, R, E, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite windows, a DyadicRatCore tolerance ledger, a RealUp seal, a CauchyCompletenessWitness row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3749.1, Definition 3749.2, and Definition 3749.3 seed the finite regular-Cauchy completion stability surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-source exposure, finite-window selection, dyadic tolerance accounting, real-seal confinement, completion-witness attachment, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionStabilityUp.

Chapter 2780

A Concrete Naming Certificate for $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ packet records the finite subsequence bridge from a bounded real sequence display toward regular-Cauchy extraction. It sits between the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational handoff of chapter 301, the finite stream schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the bounded regular Cauchy route of chapter 1698, the bounded interval route of chapter 3716, and the diagonal extraction route of chapter 3469. The packet names the monotone-window surface used by Bolzano-Weierstrass-style arguments; it does not select an infinite subsequence or assert classical compactness.

Definition 2780.1 (Bounded monotone subsequence carrier)**.** *A* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule exposing the finite sequence windows available to the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the dyadic tolerances and rational comparisons used inside the windows.*
3. *I is the bounded interval row naming the finite lower and upper barriers consumed by the monotone-window read.*
4. *W is the monotone-window ledger, consisting of a finite subsequence index list, a direction tag, and the displayed comparisons proving monotonicity inside the named window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from S, D, I, W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational handoff has been displayed.*
7. *H records componentwise* hsame *transport for the schedule, dyadic ledger, interval bounds, monotone window, regular handoff, Real seal, continuation, provenance, and local name rows.*
8. *C records displayed* Cont *replay from finite window requests through the dyadic ledger, interval bounds, monotone window, regular-rational handoff, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, I, W, R, E, H, C, P, N$. It has no coordinate for a chosen infinite subsequence, quotient real equality, trichotomy, countable choice, full Bolzano-Weierstrass compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneSubsequenceUp.BoundedMonotoneSubsequenceTasteGate single carrier alignment`

Theorem 2780.2 (Bounded monotone subsequence monotone-window preservation)**.** *For an accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier $X = (S, D, I, W, R, E, H, C, P, N)$, every subsequence read exposed by the packet preserves the displayed monotone-window ledger W under the componentwise transport H and replay row C. The read first consumes the finite stream schedule S, the dyadic comparison ledger D, and the interval bound row I; only then may it read the monotone direction and finite subsequence list stored in W.*

Proof. Project the carrier rows. The stream schedule S is the only source of sequence observations, while D and I are the only rows that record the comparison scale and bounded interval data used by the window.

The monotone-window row W is therefore downstream of S, D, I. A consumer that asks for an increasing or decreasing window must read the finite index list and direction tag from W, and the displayed comparisons in D certify the order relation inside the named interval row.

Transport and replay do not add another source. The row H preserves the same displayed coordinates componentwise, and C replays the request path through the same windows and comparisons. Since the carrier has no coordinate for a chosen infinite subsequence, trichotomy, countable choice, or ambient completeness, the monotone-window surface is exactly the finite one stored in W. □

Theorem 2780.3 (Bounded monotone subsequence regular-Cauchy handoff)**.** *Every regular-Cauchy consumer of an accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier factors through the displayed route*

$$S \longmapsto D \longmapsto I \longmapsto W \longmapsto R \longmapsto E$$

together with H, C, P, N. The handoff can be consumed by the bounded regular Cauchy, bounded interval, and diagonal Cauchy subsequence packets, but it does not identify quotient real representatives, choose an infinite subsequence, or assert $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a regular-Cauchy read. Carrier admission gives the finite schedule S, the dyadic ledger D, the bounded interval row I, and the monotone-window row W as the only pre-handoff data.

Apply the monotone-window preservation theorem. The consumer cannot read R until the finite monotone comparisons in W have been displayed over the same S, D, I rows.

The regular-rational row R is then the unique handoff available to the downstream Cauchy packets, and the Real seal row E is downstream of that handoff. The structural rows H, C, P, N preserve, replay, source, and name the same route. Hence bounded regular Cauchy, bounded interval, and diagonal Cauchy consumers may reuse the finite handoff, but no quotient equality, choice-selected infinite subsequence, or ambient completeness principle is exported. □

Theorem 2780.4 (Bounded monotone subsequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier $X = (S, D, I, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubseque}}$

surface consists exactly of carrier admission, finite-window classification, dyadic comparison coverage, bounded-interval nonescape, monotone-window preservation, regular-Cauchy handoff, Real-seal nonescape, and provenance replay through H, C, P, N.

No obligation row is stored outside the displayed carrier, and no exported read depends on full Bolzano-Weierstrass, chosen infinite subsequences, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission projects the tuple $S, D, I, W, R, E, H, C, P, N$. The finite-window classifier reads S, the dyadic comparison coverage reads D, and bounded-interval nonescape reads I before any subsequence-facing row is available.

Monotone-window preservation is supplied by Definition 3750.2. It makes W the finite source of the direction tag, subsequence index list, and displayed monotone comparisons.

Regular-Cauchy and Real-facing obligations are supplied by Definition 3750.3. The regular-rational row R and Real seal row E remain downstream of the same finite window data.

The structural rows close the surface. The transport H, replay C, provenance P, and local name N preserve and source the displayed route, and the carrier contains no coordinate for the excluded classical or ambient-completeness principles. Therefore the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ packet is generated from a StreamName schedule, a DyadicRatCore comparison ledger, bounded interval rows, a finite monotone-window ledger, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3750.1, Definition 3750.2, Definition 3750.3, and Definition 3750.4 name the finite monotone-window bridge from bounded real sequence data to regular-Cauchy extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BoundedMonotoneSubsequenceUp.BoundedMonotoneSubsequenceTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close full Bolzano-Weierstrass, chosen infinite subsequences, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier, monotone-window, regular-Cauchy handoff, Real-seal, transport, replay, provenance, and local NameCert rows for the same finite packet.

Chapter 2781

A Concrete Naming Certificate for $\mathsf{DistributionDerivative}^{\uparrow}$

Following Schwartz distribution theory and Evans' weak-derivative presentation for PDE, the $\mathsf{DistributionDerivative}^{\uparrow}$ packet names the finite action route by which a distribution is differentiated through test probes. It sits after the distribution endpoint interface of chapter 186, the weak-derivative handoff of chapter 2210, the Sobolev ledger of chapter 440, the real seal of chapter 38, the regular rational readback surface of chapter 301, and the finite stream windows of chapter 126. The chapter records only the finite carrier, its naming obligations, and the weak-action route needed by downstream PDE and variational consumers.

Definition 2781.1 (Distribution derivative carrier)**.** *A* $\mathsf{DistributionDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, D, I, E, Q, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the test-action row, listing the finite test probes, support windows, sign convention, and coordinate direction.*
2. *D is the input* $\mathsf{Distribution}^{\uparrow}$ *row, read only by evaluating the tests named in A.*
3. *I is the integration-by-parts sign row, recording the displayed minus sign and coordinate handoff for the derivative action.*
4. *E is the candidate derivative action row on the same finite tests.*
5. *Q is the shared* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for input and candidate action values.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite tests, tolerances, and readback prefixes.*
7. *H records componentwise* hsame *transport for $A, D, I, E, Q, W, C, P, N$.*
8. *C records displayed* Cont *replay from test selection to input action, candidate action, readback, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DistributionDerivative}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for a host distribution quotient, selected representative, completed test-function space, full PDE regularity theorem, or ambient Real completeness.

Theorem 2781.2 (Distribution derivative NameCert obligations)**.** *For every accepted* $\mathsf{DistributionDerivative}^{\uparrow}$ *carrier* $K = (A, D, I, E, Q, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, test-action coverage, input distribution access, integration-by-parts sign exposure, candidate derivative action exposure, shared* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite* $\mathsf{StreamName}^{\uparrow}$ *window selection, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DistributionDerivativeUp.DistributionDerivativeTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3751.1. The analytic rows visible to a consumer are A, D, I, E, Q, W: tests, input distribution action, sign handoff, candidate derivative action, readback, and finite windows.

The structural rows H, C, P, N preserve and name only that route. Componentwise transport may compare two accepted packets, and replay may re-read the same displayed action chain, but neither operation creates a host derivative object.

Since the carrier has no coordinate for distribution quotients, selected representatives, completed test-function spaces, full PDE regularity, or ambient Real completeness, the NameCert obligations are exactly the displayed finite rows. □

Theorem 2781.3 (Distribution derivative weak-action route)**.** *Every* $\mathsf{WeakDerivative}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*, PDE, trace, Rellich, or variational consumer of an accepted* $\mathsf{DistributionDerivative}^{\uparrow}$ *packet must read the derivative through the finite route*

$$A \longmapsto D, I, E \longmapsto Q \longmapsto W \longmapsto H, C, P, N.$$

The route supplies a distribution action and a candidate derivative action on the same displayed tests, with shared readback and stream windows. It does not supply quotient equality of distributions, a chosen representative, Sobolev completion, density of smooth tests, elliptic regularity, trace compactness, or a completed Real carrier.

Proof. Begin with a derivative-facing consumer. The test row A fixes the probes and support windows before any action value can be inspected.

Read D, I, E next. The input distribution action and candidate derivative action are tied by the displayed sign handoff, and both actions are evaluated only on the tests already selected by A.

Finally read Q, W, H, C, P, N. These rows provide real and regular-rational readback, finite windows, transport, replay, provenance, and local naming for the same action route. No absent analytic completion or quotient surface can be recovered from that finite packet. □

Falsifiable prediction: Every accepted $\mathsf{DistributionDerivative}^{\uparrow}$ weak-action read factors through a displayed test-action row, input DistributionUp row, integration-by-parts sign row, candidate derivative action row, shared RealUp and RegSeqRatUp readback, StreamName windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: distribution, weak-derivative, sobolev, real, regseqrat, streamname: $\mathsf{DistributionDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries input distribution action, candidate derivative action, sign handoff, shared readback, finite windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{DistributionDerivative}^{\uparrow}$ packet is generated from finite test-action rows, an input DistributionUp row, an integration-by-parts sign row, a candidate derivative action row, shared RealUp and RegSeqRatUp readback, StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3751.1, Definition 3751.2, and Definition 3751.3 seed the finite distribution-action derivative packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution quotients, selected representatives, completed test-function spaces, full PDE regularity, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-action exactness, sign-handoff stability, readback stability, weak-action nonescape, and TasteGate field faithfulness for BEDC.Derived.DistributionDerivativeUp.

Chapter 2782

A Concrete Naming Certificate for $\mathsf{NetClusterPoint}^{\uparrow}$

Following Moore–Smith and Kelley cluster points, the $\mathsf{NetClusterPoint}^{\uparrow}$ packet records cluster-point evidence for a directed net through displayed cofinal tails and neighbourhood hits rather than through a selected subnet or an ambient compactness theorem. It sits over the Moore–Smith net carrier of chapter 3403, the directed-index surface of chapter 1607, the cofinal subnet route of chapter 1600, the convergence interface of chapter 1667, and the compactness handoff of chapter 3660. The chapter names the finite BEDC landing for cluster-point evidence consumed by later net and compactness arguments, not a quotient of nets, a choice-selected subnet, or a host-level limit principle.

Definition 2782.1 (Net cluster-point carrier)**.** *A* $\mathsf{NetClusterPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, D, S, T, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *source row, naming the directed net, its displayed value windows, and the finite tail ledger consumed by every cluster-point read.*
2. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row, exposing the upper-window witnesses needed to compare tail requests.*
3. *S is the* $\mathsf{DirectedSubnet}^{\uparrow}$ *cofinal-tail row. It records the cofinal reindexing evidence through which a cluster-point consumer may follow a subnet-shaped tail without selecting an external subnet representative.*
4. *T is the* $\mathsf{NetConvergence}^{\uparrow}$ *neighbourhood-hit row. It records that each displayed neighbourhood request is met along the cofinal tail route before a convergence-facing consumer is exposed.*
5. *L is the* $\mathsf{LimitPointCompact}^{\uparrow}$ *handoff row, available only after the Moore–Smith source, directed index, cofinal-tail, and neighbourhood-hit rows have been displayed.*
6. *H records componentwise* hsame *transport for the Moore–Smith net row, directed-index row, cofinal-tail row, neighbourhood-hit row, compactness handoff, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a cluster-point request through M, D, S, T, L, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{NetClusterPoint}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{DirectedSubnet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{LimitPointCompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$ *row.*

The carrier contains no quotient-net coordinate, no selected cluster point, no countable-choice subnet selector, no host neighbourhood filter, and no ambient compactness or completeness theorem.

Lean4 ✓ `BEDC.Derived.NetClusterPointUp.NetClusterPointTasteGate single carrier alignment`

Theorem 2782.2 (Net cluster-point NameCert obligations)**.** *For every accepted* $\mathsf{NetClusterPoint}^{\uparrow}$ *carrier $K = (M, D, S, T, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Moore–Smith source, directed-index row, cofinal-tail row, neighbourhood-hit row, compactness handoff, transport, replay, provenance, and local naming rows;*
2. *source-tail exposure, requiring each cluster-point request to read the* $\mathsf{MooreSmithNet}^{\uparrow}$ *value window and* $\mathsf{DirectedSet}^{\uparrow}$ *upper-window witnesses before any neighbourhood-hit row is consumed;*
3. *cofinal-tail exactness, requiring the* $\mathsf{DirectedSubnet}^{\uparrow}$ *row S to expose only displayed cofinal reindexing data from the same directed source;*
4. *neighbourhood-hit routing, requiring T to route every cluster-point neighbourhood request through the* $\mathsf{NetConvergence}^{\uparrow}$ *interface before the compactness handoff is visible;*
5. *compactness handoff, requiring L to consume only the displayed cluster route supplied by M, D, S, T before any* $\mathsf{LimitPointCompact}^{\uparrow}$ *consumer receives the packet;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Consequently the packet supplies finite Moore–Smith cluster-point evidence and exports no quotient of nets, selected subnet, host neighbourhood filter, countable choice, or ambient compactness theorem.

Proof. Project the accepted carrier of Definition 3752.1. Its analytic rows are M, D, S, T, L, and its structural rows are H, C, P, N.

The source-tail exposure is read from M and D. A cluster-point consumer first names the displayed Moore–Smith net window, then reads the directed upper-window witness that places the request inside the finite tail ledger.

The cofinal route is fixed by S. Since S is a $\mathsf{DirectedSubnet}^{\uparrow}$ row over the same directed source, it contributes only displayed cofinal-tail evidence and cannot introduce a selected subnet representative or a countable-choice selector.

The neighbourhood-hit row T is downstream of that cofinal route. It uses the $\mathsf{NetConvergence}^{\uparrow}$ interface to expose only the finite neighbourhood hits visible along the displayed tail, so no host neighbourhood filter or quotient-net equality is available.

Lean4 ✓ `BEDC.Derived.NetClusterPointUp.NetClusterPoint neighbourhood hit netconvergence dependency`

Finally L hands the route to $\mathsf{LimitPointCompact}^{\uparrow}$, while H, C, P, N transport, replay, source, and name the same finite packet. The carrier inventory has no stronger compactness or completeness coordinate, so the local certificate is exhausted by the stated obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3752.1 and Definition 3752.2 seed the finite cluster-point route through MooreSmithNet, DirectedSet, DirectedSubnet, NetConvergence, LimitPointCompact, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{NetClusterPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected subnets, countable-choice selectors, host neighbourhood filters, ambient compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-tail exposure, cofinal-tail exactness, neighbourhood-hit routing, compactness handoff, transport, replay, provenance, and local naming for BEDC.Derived.NetClusterPointUp.

Constructive story: A NetClusterPoint$^{\uparrow}$ packet is generated from a MooreSmithNet source row, a DirectedSet index row, a DirectedSubnet cofinal-tail row, a NetConvergence neighbourhood-hit row, a LimitPointCompact handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2783

A Concrete Naming Certificate for $\mathsf{GreedySpectralClosure}^{\uparrow}$

The $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet realizes the block-optimal spectral closure route of Definition 9462.1419 as finite BEDC NameCert data. It reads the active codon support, fixed-budget spectral frontiers, anchor and median layers, and symmetric inactive completion from the DNA-window spectral closure vision, then records only the finite trigger-spectrum rows needed by downstream finite-window and trigger-algebra consumers.

The trigger-spectrum route consumes the finite observation discipline of chapter 689, the boolean trigger rows of chapter 3704, the window-coverage trigger algebra of chapter 3478, and the stabilization surface of chapter 2270. Thus the packet is not a kernel-only spectral table: its support, trigger, and layered-closure rows are read as sibling finite-window and trigger-algebra data before the local NameCert row can be consumed.

Definition 2783.1 (Greedy spectral closure carrier)**.** *A* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, A, M, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the active support row for $\mathcal{R}$ and the finite complement inside Q_6.*
2. *A is the anchor-block row naming the unique three-point spectral maximizer* $\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$.
3. *M is the median-block row naming the next three-point spectral maximizer* $\{\mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG}\}$.
4. *T is the terminal singleton row naming* ACG *as the next single-point spectral maximizer.*
5. *R is the radius table row carrying the displayed killed-walk spectral values and multiplicity counts.*
6. *L is the layered-closure row saying that the packet is block-optimal rather than a nested fixed-budget optimum from $\mathcal{R}$.*
7. *H records componentwise* hsame *transport for support, anchor, median, terminal, radius, layered-closure, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a trigger-spectrum request through S, A, M, T, R, L, H.*
9. *P is the* Pkg *provenance over* $\mathsf{GreedySpectralClosure}^{\uparrow}$, $\mathsf{FiniteWindowObserver}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{SpectralStabilization}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, M, T, R, L, H, C, P, N$. It has no coordinate for a host Perron-root oracle, biological-code truth, stochastic evolution model, quotient codon semantics, or unbounded spectral search.

Lean4 (definition) `BEDC.Derived.GreedySpectralClosureUp.GreedySpectralClosureUp`

Theorem 2783.2 (Greedy spectral closure NameCert obligations)**.** *For every accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier* $G = (S, A, M, T, R, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$ *surface consists exactly of finite support admission, anchor-block exposure, median-block exposure, terminal singleton exposure, spectral table exposure, layered-closure exposure, componentwise transport, displayed replay, provenance, and local naming.*

The Lean follow-up for this packet should provide Nontrivial and FieldFaithful instances for BEDC.Derived.GreedySpectralClosureUp so that $S, A, M, T, R, L, H, C, P, N$ remain separated carrier fields.

Proof. Project the carrier of Definition 3753.1. The rows S, A, M, T, R, L expose the finite support, the three chosen blocks, the spectral table, and the layered-closure reading before any consumer can cite the packet.

The spectral data are finite table rows. They record the displayed optima and multiplicities from the vision file, but they do not introduce a Perron-root oracle or an unbounded search procedure.

The structural rows H, C, P, N preserve that route. Since no host biological truth predicate, stochastic model, quotient codon equality, hidden spectral computation, or unbounded enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Theorem 2783.3 (Greedy spectral closure trigger nonescape)**.** *Every trigger-spectrum consumer of an accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *packet factors through*

$$S \longmapsto A, M, T \longmapsto R, L \longmapsto H, C, P, N.$$

The route may cite the finite anchor block, median block, terminal singleton, displayed spectral values, and layered-closure row. It cannot export a biological-code oracle, a stochastic evolutionary law, an ambient powerset closure, quotient codon equality, or an unbounded spectral optimizer.

Proof. Start with the finite support row S. It fixes the active support and complement inside the displayed codon window, so no later read can range over an unbounded search space.

Read the rows A, M, T next. They name the anchor block, median block, and terminal singleton as the finite closure path consumed by trigger-spectrum clients.

The table and layered rows R, L identify the route as block-optimal and budget-frustrated. They do not change the packet into a global optimizer, because the carrier stores only the displayed values and multiplicities.

Finally H, C, P, N transport, replay, source, and name the same route. The absent host objects listed in the statement are not coordinates of the carrier and cannot be exported by a consumer read. □

Theorem 2783.4 (Greedy spectral closure sibling trigger route)**.** *Let* $G = (S, A, M, T, R, L, H, C, P, N)$ *be an accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier. Every trigger-spectrum consumer of G first reads the finite-window observer, boolean trigger lattice, window-coverage trigger algebra, and spectral-stabilization sibling rows recorded in P before it consumes the displayed rows*

$$S, A, M, T, R, L, H, C, P, N.$$

Consequently the packet cannot be consumed as an isolated spectral kernel detached from finite observation, trigger algebra, coverage, or stabilization surfaces.

Proof. Project the provenance row P in Definition 3753.1. It names $\mathsf{FiniteWindowObserver}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, and $\mathsf{SpectralStabilization}^{\uparrow}$ as part of the same accepted packet source.

The finite-window observer fixes the public observation row from which the active support S is read. The boolean trigger lattice and window-coverage trigger algebra then supply the trigger rows against which the anchor, median, and terminal rows A, M, T are interpreted.

Only after those sibling rows are in place can the displayed spectral table and layered-closure rows R, L be consumed. The structural rows H, C, P, N transport, replay, source, and name that route, so no consumer can bypass the sibling finite-window and trigger-algebra surfaces while still reading G as an accepted $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet. □

Falsifiable prediction: An accepted $\mathsf{GreedySpectralClosure}^{\uparrow}$ trigger-spectrum read is refuted if the anchor block, median block, terminal singleton, displayed spectral table, layered-closure row, transport, replay, provenance, or local naming can be removed while preserving the same packet claim.

Independence witness: finite-window-observer, boolean-trigger-lattice, window-coverage-trigger-algebra, spectral-stabilization, dna-window spectral closure vision: none is bijective with $\mathsf{GreedySpectralClosure}^{\uparrow}$ because the packet jointly carries support, anchor, median, singleton, spectral table, layered closure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet is generated from active support rows, anchor-block rows, median-block rows, terminal singleton rows, displayed spectral table rows, layered-closure rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3753.1, Definition 3753.2, and Definition 3753.3 seed the finite trigger-spectrum closure packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, ambient powerset closure, quotient codon equality, hidden spectral computation, unbounded optimization, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-spectrum exactness, displayed-table stability, Nontrivial and FieldFaithful instances, and trigger nonescape for BEDC.Derived.GreedySpectralClosureUp.

Chapter 2784

A Concrete Naming Certificate for CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$ packet names the finite comparison surface between two displayed Cauchy-complete ordered-field presentations of a Real-style object. It reads the terminal Real$^{\uparrow}$ seal, regular rational sequence rows, dyadic arithmetic ledger, algebra-order rows, the CauchyCompleteOrderedField$^{\uparrow}$ interface, and the CompleteArchimedeanField$^{\uparrow}$ capstone as input packets, then records only the BEDC route that compares their NameCert-visible rows. The chapter exports uniqueness as rowwise route agreement, not as quotient field equality, trichotomy, a selected limit, or a host complete ordered field.

Definition 2784.1 (Cauchy-complete ordered-field uniqueness carrier)**.** *A* CauchyCompleteOrderedFieldUniqueness *carrier is a finite* BHist *packet*

$$U = (K_0, K_1, R_0, R_1, Q_0, Q_1, D, A_0, A_1, F_0, F_1, G, H, C, P, N)$$

with the following displayed rows:

1. *K_0 and K_1 are two accepted* CauchyCompleteOrderedField$^{\uparrow}$ *packets whose Real-facing reads are to be compared.*
2. *R_0 and R_1 are the terminal* Real$^{\uparrow}$ *seal rows exposed by K_0 and K_1.*
3. *Q_0 and Q_1 are the* RegSeqRat$^{\uparrow}$ *readback rows used before the terminal Real seals are consumed.*
4. *D is the shared* DyadicRatCore$^{\uparrow}$ *comparison ledger for precision scales, rational windows, and tolerance transfer.*
5. *A_0 and A_1 are the* RealAlgOrder$^{\uparrow}$ *operation-and-order rows displayed by the two packets.*
6. *F_0 and F_1 are the* CompleteArchimedeanField$^{\uparrow}$ *capstone rows reached from the two Cauchy-complete ordered-field presentations.*
7. *G is the BEDC comparison route, recording componentwise agreement of the regular-rational windows, dyadic tolerances, Real seals, algebra-order rows, Cauchy-complete ordered-field rows, and complete Archimedean capstone rows.*
8. *H records componentwise* hsame *transport for all displayed rows of U.*
9. *C records displayed* Cont *replay from a comparison request through $Q_0, Q_1, D, R_0, R_1, A_0, A_1, K_0, K_1, F_0, F_1, G, H$*

10. *P is the* Pkg *provenance over* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchimedeanField$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted uniqueness packets only through these rows. It has no coordinate for quotient field equality, trichotomy, a selected limit, a chosen isomorphism outside G, or an ambient host complete ordered field.

Lean4 ✓ `BEDC.Derived.CauchyCompleteOrderedFieldUniquenessUp.CauchyCompleteOrderedFieldUniquenessTasteGa`
`single carrier alignment`

Theorem 2784.2 (Cauchy-complete ordered-field uniqueness sibling dependency route)**.** *Every accepted* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$ *carrier is a dependency consumer of the Real and Completion lattice. The two terminal seal rows R_0, R_1 are read through chapter 38; the readback rows Q_0, Q_1 are read through chapter 301; the shared tolerance ledger D is read through chapter 291; the operation-and-order rows A_0, A_1 are read through chapter 766; the comparison packets K_0, K_1 are read through chapter 3688; and the capstone rows F_0, F_1 are read through chapter 2025. Hence the uniqueness packet is not self-standing: its rowwise comparison route is grounded in those sibling chapters before any consumer may cite the uniqueness surface.*

Proof. Project the carrier U of Definition 3754.1. Its visible comparison data begins with the two regular rational readbacks Q_0, Q_1 and the shared dyadic ledger D, so a consumer must first pass through the RegSeqRat and DyadicRatCore sibling rows that make the tolerance comparison meaningful.

The Real seals R_0, R_1 are then consumed as terminal name rows, not as quotient-field equality. The algebra-order rows A_0, A_1 supply the operation-order surface used by the two Cauchy-complete ordered-field packets K_0, K_1.

Finally the complete Archimedean capstone rows F_0, F_1 and the comparison route G record agreement along the displayed BEDC path. The structural rows H, C, P, N preserve and name that path, so no host complete ordered field, selected isomorphism, trichotomy, or quotient equality can bypass the listed sibling dependencies. □

Theorem 2784.3 (Cauchy-complete ordered-field uniqueness NameCert obligations)**.** *For every accepted* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$ *carrier*

$$U = (K_0, K_1, R_0, R_1, Q_0, Q_1, D, A_0, A_1, F_0, F_1, G, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$ *surface consists exactly of the two displayed* CauchyCompleteOrderedField$^{\uparrow}$ *packets, their* Real$^{\uparrow}$ *seal rows, their* RegSeqRat$^{\uparrow}$ *readback rows, the shared* DyadicRatCore$^{\uparrow}$ *comparison ledger, their* RealAlgOrder$^{\uparrow}$ *operation-and-order rows, their* CompleteArchimedean *capstone rows, the BEDC comparison route G, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3754.1. The projection exposes exactly the two Cauchy-complete ordered-field packets, the Real seals, regular-rational readbacks, dyadic comparison ledger, algebra-order rows, complete Archimedean capstone rows, comparison route, transport, replay, provenance, and local naming.

The comparison route first reads Q_0, Q_1 against the shared dyadic ledger D. This places every tolerance and rational-window comparison before the terminal Real seals R_0, R_1 are consumed.

It then compares the ordered-field surfaces through A_0, A_1 and the completion-facing surfaces through K_0, K_1, F_0, F_1. The row G records exactly this BEDC route and not an external field equality or selected isomorphism.

Finally H, C, P, N preserve, replay, source, and name the same displayed rows. Since quotient field equality, trichotomy, selected limits, and host complete ordered fields have no coordinate in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2784.4 (Cauchy-complete ordered-field uniqueness route)**.** *Every accepted* CauchyCompleteOrderedFieldU *comparison between two displayed Cauchy-complete ordered-field packets factors through*

$$Q_0, Q_1, D \longmapsto R_0, R_1 \longmapsto A_0, A_1, K_0, K_1 \longmapsto F_0, F_1 \longmapsto G, H, C, P, N.$$

Thus a consumer may cite a finite BEDC route from regular rational readback and dyadic tolerance comparison through Real seals, algebra-order rows, Cauchy-complete ordered-field rows, and complete Archimedean capstone rows. The consumer receives no quotient field equality, trichotomy principle, selected limit, chosen external isomorphism, or host complete ordered field.

Proof. Start with the regular-rational side. The rows Q_0, Q_1 are the only readbacks available before Real sealing, and the shared ledger D supplies the dyadic precision and comparison data for both packets.

The route next reaches R_0, R_1. Only after these Real seals are displayed may the comparison inspect the algebra-order rows A_0, A_1 and the two $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packets K_0, K_1.

The capstone comparison is then made through F_0, F_1, the $\mathsf{CompleteArchimedeanField}^{\uparrow}$ rows attached to the two presentations. The row G records that the two presentations agree along this displayed BEDC path.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Because no excluded equality, trichotomy, selected-limit, external-isomorphism, or host-field coordinate appears anywhere in the carrier, every uniqueness read factors through the finite route stated above. □

Theorem 2784.5 (Cauchy-complete ordered-field uniqueness regular-Cauchy dependency)**.** *Every accepted* $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ *packet must route its completion witness through the regular-Cauchy completion surface of chapter 1513 and the terminal Real seal of chapter 38 before it compares ordered-field endpoints. The ordered-field comparison therefore reads the displayed completion and real-name rows first; it does not introduce a host complete ordered field, quotient equality, or a selected external isomorphism.*

Proof. Begin with an accepted carrier U. Its only completion-facing source rows are the regular-rational readbacks Q_0, Q_1, the dyadic comparison ledger D, the Real seals R_0, R_1, and the Cauchy-complete ordered-field rows K_0, K_1.

The separated regular-Cauchy completion sibling supplies the finite completion route consumed by those rows, while the Real chapter supplies the terminal seal that the ordered-field comparison is allowed to inspect. Thus the comparison cannot jump directly from K_0, K_1 to endpoint equality.

Only after that route is displayed may the algebra-order rows A_0, A_1 and capstone rows F_0, F_1 be read. The comparison row G records agreement along this finite path, so no host complete ordered field, quotient equality, or external isomorphism is available. □

Falsifiable prediction: Every $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ comparison read factors through two CauchyCompleteOrderedField packets, two Real seals, two RegSeqRat readbacks, one DyadicRatCore comparison ledger, two RealAlgOrder rows, two CompleteArchimedeanField rows, one BEDC comparison route, transport, continuation replay, provenance, and local naming.

Independence witness: real, regseqrat, dyadicratcore, real-alg-order, cauchy-complete-ordered-field, complete-archimedean-field: $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two displayed complete ordered-field presentations and a rowwise comparison route between them.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ packet is generated from two CauchyCompleteOrderedField packets, two Real seals, two RegSeqRat readbacks, a DyadicRatCore comparison ledger, two RealAlgOrder rows, two CompleteArchimedeanField rows, one BEDC comparison route, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3754.1, Definition 3754.3, and Definition 3754.4 seed the finite comparison surface for two displayed Cauchy-complete ordered-field presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert quotient field equality, trichotomy, selected limits, a chosen external isomorphism, a host complete ordered field, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, and the rowwise BEDC comparison route for BEDC.Derived.CauchyCompleteOrderedFieldUniquenessUp.

Chapter 2785

A Concrete Naming Certificate for $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$

Following Bishop and Bridges constructive metric completion theory, the $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ packet records the finite route by which a uniformly controlled metric map induces a map between displayed completion packets. It sits after the metric completion carrier of chapter 1506, the functor-facing Cauchy completion surface of chapter 423, the universal extension ledger of chapter 397, the dense-extension route of chapter 3475, the regular rational readback of chapter 301, and the finite stream-window interface of chapter 126. The packet names functoriality only as displayed source, target, map, Cauchy-control, completion-entry, universal-property, comparison, transport, replay, provenance, and local NameCert rows; it is not a quotient-completion equality, a selected limit map, function extensionality, countable choice, or ambient completeness.

Definition 2785.1 (MetricCompletionFunctoriality carrier)**.** *A* $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, G, A, E, U, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose completion-facing names are inspected.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose completion packet receives the induced read.*
3. *G is the finite graph row of the underlying map from source names to target names.*
4. *A is the Cauchy-control ledger, recording the modulus or window transfer by which* $\mathsf{StreamName}^{\uparrow}$ *observations of X yield* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks over Y.*
5. *E is the completion-entry row, naming the* $\mathsf{MetricCompletion}^{\uparrow}$ *source and target entries used by the induced map.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row through which the extension and uniqueness read is consumed.*
7. *S is the separated comparison row, requiring induced target reads to compare only through the displayed separated metric or dense-extension surface.*
8. *H records componentwise* hsame *transport for the source metric, target metric, graph, Cauchy-control, completion-entry, universal-property, separated-comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a source completion request through G, A, E, U, S, H, P, N.*
10. *P is the* Pkg *provenance over* MetricCompletion$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, DenseIsometryExtension$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetricCompletionFunctoriality}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completions, selected representatives of Cauchy classes, host function equality, countable-choice selection, an ambient completed metric space, or a second extension principle outside U.

```
Lean4 ✓  BEDC.Derived.MetricCompletionFunctorialityUp.MetricCompletionFunctorialityTasteGate
single carrier alignment
```

Theorem 2785.2 (MetricCompletionFunctoriality NameCert obligations)**.** *For every accepted* MetricCompletionFunctoriality$^{\uparrow}$ *carrier $F = (X, Y, G, A, E, U, S, H, C, P, N)$, the local* NameCert$_{\text{MetricCompletionFunctoria}}$ *surface consists exactly of carrier admission for the source metric, target metric, finite map graph, Cauchy-control ledger, completion-entry row, universal-property row, separated comparison row, componentwise transport, continuation replay, provenance, and local naming. Every induced completion read must pass through the* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *control recorded in A, the* MetricCompletion$^{\uparrow}$ *entry row E, the* CompletionUniversalProperty$^{\uparrow}$ *ledger U, and the separated comparison row S before the local name N is exposed.*

Proof. Project the carrier of Definition 3755.1. The rows $X, Y, G, A, E, U, S, H, C, P, N$ are the whole public packet, so source and target metric data, the map graph, the Cauchy-control ledger, completion entries, universal-property data, and separated comparison are all displayed before the induced map is consumed.

The Cauchy-control row supplies the constructive handoff. A source completion request first reads finite StreamName$^{\uparrow}$ windows, then the RegSeqRat$^{\uparrow}$ readback governed by A, and only then reaches the completion-entry row E.

The universal-property and comparison rows bind the extension. The row U records the extension-and-uniqueness ledger, while S keeps target comparison inside the displayed separated surface rather than in host equality or a quotient-completion principle.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for selected representatives, choice, ambient completeness, host function equality, or a detached extension principle, the listed rows exhaust the local NameCert obligations. □

Theorem 2785.3 (MetricCompletionFunctoriality composition)**.** *Let*

$$F = (X, Y, G, A, E, U, S, H, C, P, N)$$

and

$$F' = (Y, Z, G', A', E', U', S', H', C', P', N')$$

be accepted MetricCompletionFunctoriality$^{\uparrow}$ *carriers whose middle metric row Y and completion-entry row compare by the displayed* hsame *transport. The composed completion read is the finite replay obtained by first consuming G, A, E, U, S and then consuming G', A', E', U', S'. It is sourced by the same* CauchyCompletionFunctoriality$^{\uparrow}$ *and* MetricCompletion$^{\uparrow}$ *rows named in P and P', and it exports no host-level function composition, quotient-completion equality, selected limit representative, countable choice, or ambient completeness theorem.*

Proof. Start with the two carriers. The first packet exposes the route from X to Y through its finite graph, Cauchy-control ledger, completion-entry row, universal-property row, and separated comparison row; the second exposes the same kind of route from Y to Z.

Use the displayed hsame transport on the shared middle rows. This identifies the target metric and completion-entry read of the first packet with the source metric and completion-entry read of the second packet only through the finite transport rows already present in H and H'.

Replay the two continuation rows in order. The composed route consumes C and then C', so every source completion read still passes through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, and separated comparison rows before the final local name is available.

The nonescape boundary is inherited from the carrier inventory. Neither packet contains host function equality, quotient-completion equality, selected representatives, countable choice, or ambient completeness, and sequential replay of displayed rows cannot create such a coordinate. □

Falsifiable prediction: Any accepted $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ consumer that maps a source completion name to a target completion name without displaying the finite graph row, Cauchy-control ledger, completion-entry row, universal-property row, and separated comparison row refutes the carrier surface.

Independence witness: metriccompletion, cauchycompletionfunctor, completionuniversalproperty, dense-isometry-extension, regseqrat, streamname: these sibling chapters supply completion entries, functorial Cauchy routes, universal extension ledgers, dense separated comparison, regular readbacks, and finite windows separately, while this packet binds them into one functorial carrier $F = (X, Y, G, A, E, U, S, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ packet is generated from source and target metric rows, a finite map graph, a Cauchy-control ledger, a completion-entry row, a universal-property row, a separated comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3755.1, Definition 3755.2, and Definition 3755.3 seed the finite functoriality surface for metric completion maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives of Cauchy classes, host function extensionality, countable-choice selection, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-control routing, completion-entry exactness, universal-property replay, separated comparison, transport, provenance, and local naming for BEDC.Derived.MetricCompletionFunctoriali

Chapter 2786

A Concrete Naming Certificate for NaturalDeduction$^{\uparrow}$

Following Gentzen and Prawitz, the NaturalDeduction$^{\uparrow}$ packet records a finite proof-tree calculus whose rows expose assumptions, rule applications, discharge, conclusion readback, and proof-term boundary data without importing host proof equality. It sits beside the syntax-and-deduction surface of chapter 197, the proof-term carrier of chapter 200, the bounded computation packet of chapter 199, and the run-trace surface of chapter 280. The packet is a finite NameCert neighbour: it names proof-tree evidence, but it does not identify proofs by normalization, beta equality, semantic validity, or machine execution.

Definition 2786.1 (NaturalDeduction carrier)**.** *A* NaturalDeduction$^{\uparrow}$ *carrier is a finite proof-tree* BHist *packet*

$$D = (A, R, S, Q, B, T, H, K, P, N)$$

with the following displayed rows:

1. *A is the assumption row, listing the open formula endpoints admitted at the leaves;*
2. *R is the rule row, recording introduction, elimination, weakening, and structural rule instances as finite local steps;*
3. *S is the discharge schedule, pairing discharged assumption occurrences with the rule rows that close them;*
4. *Q is the conclusion row, exposing the formula endpoint read at the root of the tree;*
5. *B is the proof-term boundary row, recording when a proof tree may be compared with a* LambdaCalc$^{\uparrow}$ *term without quotienting proof identity;*
6. *T is the transport row, comparing proof-tree nodes only by displayed* hsame *routes over formula, assumption, rule, discharge, and conclusion rows;*
7. *H records componentwise* hsame *stability for the assumption, rule, discharge, conclusion, proof-term boundary, transport, replay, provenance, and local name rows;*
8. *K records* Cont *replay from assumptions through rule rows and discharge rows to the conclusion and proof-term boundary;*
9. *P is the* Pkg *provenance over* FirstOrder$^{\uparrow}$, LambdaCalc$^{\uparrow}$, Computable$^{\uparrow}$, TuringMachine$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$ *row.*
The carrier has no coordinate for host proof equality, global proof normalization, semantic validity, executable machine trace, selected proof representative, or quotient proof identity.

Lean4 ✓ `BEDC.Derived.NaturalDeductionUp.TasteGate.NaturalDeductionTasteGate single carrier alignment`

Theorem 2786.2 (NaturalDeduction NameCert obligations)**.** *For the carrier of Definition 3756.1, the local* $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$ *surface consists exactly of carrier admission for* $D = (A, R, S, Q, B, T, H, K, P, N)$ *assumption exposure through* A*, rule-step exposure through* R*, discharge exposure through* S*, conclusion readback through* Q*, proof-term boundary exposure through* B*, transport through* T*, componentwise* hsame *stability through* H*,* Cont *replay through* K*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier of Definition 3756.1. The rows $A, R, S, Q, B, T, H, K, P, N$ are the only coordinates from which the local certificate may read proof-tree data.

The proof-tree content is exposed by the assumption, rule, discharge, and conclusion rows. A consumer can inspect open hypotheses, local rule instances, discharged occurrences, and the root conclusion, but those reads remain inside the finite tree packet.

The proof-term boundary row is separate from proof identity. It records the visible point at which a $\mathsf{LambdaCalc}^{\uparrow}$ term may be compared with a tree boundary, while forbidding any quotient that identifies proof trees by normalization or beta equality.

The structural rows then provide transport, replay, provenance, and naming. Since the carrier contains no further field, the listed rows exhaust the NameCert obligations. □

Theorem 2786.3 (NaturalDeduction FirstOrder consumer route)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *consumer of an accepted* $\mathsf{NaturalDeduction}^{\uparrow}$ *packet reads only the displayed formula endpoints, assumption rows, rule rows, discharge schedule, conclusion row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row of Definition 3756.1. It cannot replace the proof-tree route by host syntax equality, semantic validity, model-theoretic completeness, or an external proof checker.*

Proof. Start with a $\mathsf{FirstOrder}^{\uparrow}$ read of the accepted carrier D. The package row P names $\mathsf{FirstOrder}^{\uparrow}$ as the syntax and formula source, so the formula endpoints available to the consumer are exactly the endpoints carried by the assumption and conclusion rows.

Rule and discharge reads pass through R and S. These rows record finite local proof steps and the assumptions closed by those steps; they do not add a semantic model, a completeness theorem, or a host proof oracle.

Finally replay T, H, K, P, N. Transport and continuation preserve the same displayed proof-tree rows, and the local name row registers that route as the $\mathsf{NaturalDeduction}^{\uparrow}$ certificate. No external proof checker has a coordinate from which it could be projected. □

Theorem 2786.4 (NaturalDeduction LambdaCalc proof-term boundary)**.** *The* $\mathsf{LambdaCalc}^{\uparrow}$ *comparison surface of an accepted* $\mathsf{NaturalDeduction}^{\uparrow}$ *packet is exactly the proof-term boundary row* B *together with the assumption, rule, discharge, conclusion, transport, replay, provenance, and local naming rows of Definition 3756.1. It supplies a boundary for Curry–Howard-style comparison, but it does not identify natural-deduction proof trees with lambda terms, alpha classes, beta-normal forms, or machine traces.*

Proof. Read the boundary row B from the carrier. Its purpose is to name the visible point where a proof tree may be compared with the $\mathsf{LambdaCalc}^{\uparrow}$ term surface.

The comparison still depends on the proof-tree rows. The assumption row A, rule row R, discharge row S, and conclusion row Q determine which proof boundary is being compared, while T, H, K preserve and replay those rows componentwise.

The provenance row P may cite $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, and $\mathsf{TuringMachine}^{\uparrow}$ as nearby surfaces. This citation does not add alpha quotienting, beta-normal equality, host program equality, or a machine run trace to the natural-deduction packet.

Thus the chapter supplies a boundary for proof-term comparison only. Any stronger identification would require a separate carrier coordinate, and none is present in Definition 3756.1. □

Falsifiable prediction: If a proof-tree packet claims $\mathsf{NaturalDeduction}^{\uparrow}$ acceptance while omitting assumption rows, rule rows, discharge schedule, conclusion readback, proof-term boundary, transport, replay, provenance, or local NameCert rows, it refutes the carrier of Definition 3756.1.

Independence witness: firstorder, lambdacalc, computable, turingmachine: $\mathsf{NaturalDeduction}^{\uparrow}$ is not bijective to any listed sibling because it carries assumption rows, rule rows, discharge schedule, conclusion readback, and a proof-term boundary as a single proof-tree certificate. FirstOrderUp carries syntax and deduction endpoints without the separate proof-term boundary; LambdaCalcUp carries terms and beta-substitution ledgers without discharged proof assumptions; ComputableUp and TuringMachineUp carry bounded computation and run traces rather than finite proof-tree discharge.

Closure status for $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$.

Constructive story: A $\mathsf{NaturalDeduction}^{\uparrow}$ packet is generated from assumption rows, local rule rows, a discharge schedule, a conclusion row, a proof-term boundary row, transport, componentwise hsame stability, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3756.1, Definition 3756.2, Definition 3756.3, and Definition 3756.4 seed the finite proof-tree carrier and its consumer boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host proof equality, global proof normalization, semantic validity, Curry–Howard equivalence, alpha quotienting, beta-normal equality, Lean verification, bridge checking, or executable machine-trace identity for $\mathsf{NaturalDeduction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, proof-tree rule routing, discharge exactness, FirstOrder consumer nonescape, and LambdaCalc proof-term boundary coverage for BEDC.Derived.NaturalDeductionUp.

Chapter 2787

A Concrete Naming Certificate for $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real arithmetic, the $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ packet records finite sums of displayed regular Cauchy rational names as folded addition routes. It sits after the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, the binary addition packet of chapter 1695, the terminal real-name seal of chapter 38, and the finite-product sibling of chapter 2009. The carrier records displayed folded addition data, not host finite sums, quotient streams, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2787.1 (Regular Cauchy finite sum carrier)**.** *A* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, W, D, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite unary index row. It lists exactly the summand slots opened by the request and the displayed fold order used by the addition route.*
2. *S is the component* $\mathsf{RegSeqRat}^{\uparrow}$ *source family, one regular rational Cauchy name for each slot of I.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for component reads, intermediate addition reads, and the final sum-facing read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *sum-budget ledger, storing component observations, intermediate sum observations, requested radii, and slack cells.*
5. *A is the folded* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It records the adjacent binary additions named by I, W, D and exposes no host finite-sum operation.*
6. *R is the final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the source, window, dyadic, and folded-addition rows have been displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback of the folded sum has been displayed.*
8. *H records componentwise* hsame *transport for the index row, summand sources, synchronized windows, dyadic ledger, folded-addition route, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any finite-sum request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, A, R, E, H, C, P, N$. It has no coordinate for a host finite sum, quotient stream equality, selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFiniteSumUp.RegularCauchyFiniteSumTasteGate single carrier alignment`

Theorem 2787.2 (Regular Cauchy finite sum NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier*

$$K = (I, S, W, D, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the finite unary index row, component* $\mathsf{RegSeqRat}^{\uparrow}$ *summands, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger, folded* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route, final readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local name;*

2. *finite tuple-window coverage, requiring every summand slot in I to open the matching source row and all synchronized windows needed by the fold;*

3. *dyadic sum-budget exactness, requiring every intermediate addition observation to consume its assigned radius and slack cells from D;*

4. *RegSeqRat readback stability, requiring the final readback row R to depend only on the displayed sources, windows, dyadic ledger, and folded-addition route;*

5. *Real seal non-escape, requiring E, H, C, P, N to expose no host finite sum, quotient stream equality, selected limit, trichotomy, ambient completeness theorem, Lean verification, or bridge credential.*

No Regular Cauchy finite sum obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3757.1. The displayed tuple $I, S, W, D, A, R, E, H, C, P, N$ is the whole public surface, so carrier admission is exactly the admission of those rows.

Tuple-window coverage is forced by the index row. Each slot named by I has one component $\mathsf{RegSeqRat}^{\uparrow}$ source in S, and the synchronized window row W is the only place where component, intermediate, and final sum reads are opened.

The dyadic budget is read through D. Each intermediate binary addition request consumes a displayed radius and slack cell from that ledger before the corresponding folded $\mathsf{RegularCauchyAddition}^{\uparrow}$ row A can be inspected.

The final regular readback row R is downstream of S, W, D, A. Therefore readback stability follows from the carrier order: no sum-facing $\mathsf{RegSeqRat}^{\uparrow}$ data can be inspected until the synchronized windows, dyadic budget, and folded addition route have already been displayed.

Finally, the seal, transport, replay, provenance, and local name rows preserve only this route. Since none of the excluded objects is a coordinate of the carrier, the listed five rows exhaust the local NameCert obligations. □

Theorem 2787.3 (Regular Cauchy finite sum budget route)**.** *Let*

$$K = (I, S, W, D, A, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier. Every finite sum read from* K *factors through*

$$I, S \longmapsto W \longmapsto D, A \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, regrouping displayed adjacent additions may change only the fold order named by I *and* A*; it preserves the same synchronized windows* W*, dyadic sum-budget ledger* D*, final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *when those rows are fixed by the accepted carrier.*

Proof. Begin with a finite sum request. The request first names the finite unary index row I and the component source family S, because these are the only carrier rows that identify which regular Cauchy names participate in the fold.

The requested component and intermediate observations then pass through W. This synchronized window row aligns component reads with the intermediate addition reads used by the displayed fold.

The dyadic part of the request is read next through D, and the adjacent binary additions are read through A. The proof uses the displayed $\mathsf{RegularCauchyAddition}^{\uparrow}$ rows rather than host associativity of sums.

Only after those rows are displayed can the final regular rational readback R be consumed. The downstream rows E, H, C, P, N seal, transport, replay, source, and name the same route, so the finite sum read cannot bypass the synchronized windows, the dyadic ledger, or the folded-addition route. □

Falsifiable prediction: Every $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ sum-facing read factors through a finite unary index row, component RegSeqRat sources, synchronized StreamName windows, DyadicRatCore sum-budget observations, a folded RegularCauchyAddition route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-addition, real, regular-cauchy-finite-product: $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite unary summand index, synchronized multi-source windows, dyadic sum-budget accounting, folded addition rows, final RegSeqRat readback, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3757.1, Definition 3757.2, and Definition 3757.3 seed the finite sum route from displayed regular Cauchy sources through synchronized windows, dyadic sum budgets, folded addition rows, final RegSeqRat readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host finite sums, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tuple-window coverage, dyadic budget exactness, folded-addition routing, readback stability, seal non-escape, and local naming for BEDC.Derived.RegularCauchyFiniteSumUp.

Constructive story: A $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ packet is generated from a finite unary summand index row, component RegSeqRat source rows, synchronized StreamName windows, DyadicRatCore sum-budget observations, a folded RegularCauchyAddition route, final RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2788

A Concrete Naming Certificate for $\mathsf{EpsilonDeltaLimit}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet records the traditional epsilon-delta limit interface as finite BEDC data. It sits between the source and target metric carriers of chapter 57, the dyadic tolerance surface of chapter 291, the finite-window names of chapter 126, the regular rational readback rows of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a displayed point, a proposed target limit, positive dyadic epsilon requests, matching positive dyadic delta witnesses, finite window reads, regular-rational readback, transport, replay, provenance, and local naming; it does not identify host reals, select limits, form quotient streams, invoke trichotomy, use choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2788.1 (Epsilon-delta limit carrier)**.** *An* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, a, L, E, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including its accepted point classifier and distance-witness surface.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including its accepted point classifier and* $\mathsf{Real}^{\uparrow}$*-valued distance-witness surface.*
3. *a is the source point row at which the limiting read is taken.*
4. *L is the proposed target-limit row, read as a displayed point of Y before any Real-facing equality question is asked.*
5. *E is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *epsilon request row for the target distance to L.*
6. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *delta witness row for source distance to a.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source point, the candidate values, the proposed limit, and the two dyadic tolerance rows.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and for the* $\mathsf{Real}^{\uparrow}$ *distance rows reached through X and Y.*
9. *H records componentwise* hsame *transport for the metric rows, point row, proposed limit, epsilon request, delta witness, windows, readback, continuation, provenance, and local name.*

10. *C records displayed* Cont *replay from epsilon requests through delta witnesses, metric-distance reads, finite windows, regular-rational readback, and the target* $\mathsf{Real}^{\uparrow}$ *seal.*
11. *P is the* Pkg *provenance over* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$ *row.*

Lean4✓ `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitTasteGate single carrier alignment`

The classifier accepts the packet only when all rows name the same finite epsilon-delta route: every accepted epsilon request in E is met by the displayed delta row D, and every Real-facing target-distance read passes through $X, Y, a, L, E, D, W, R, H, C, P, N$.

Theorem 2788.2 (Epsilon-delta limit NameCert obligations)**.** *For the carrier of Definition 3759.1, the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (X, Y, a, L, E, D, W, R, H, C,$ source and target metric-space admission through X and Y, point admission through a, proposed-limit admission through L, positive dyadic epsilon and delta admission through E and D, finite-window admission through W, regular-rational and Real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The metric rows X and Y display the source and target point classifiers and the distance witnesses before the point row a or the proposed target limit L can be consumed by a limit read.

The tolerance part is finite. The epsilon request E and delta witness D are $\mathsf{DyadicRatCore}^{\uparrow}$ rows, so the smallness comparison is a dyadic ledger rather than a host real inequality or selected limit object.

The analytic readback is downstream of those rows. The $\mathsf{StreamName}^{\uparrow}$ window row W records the finite observations used by the source point, target values, proposed limit, and dyadic tolerances, while the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the displayed windows before any $\mathsf{Real}^{\uparrow}$ distance seal is exposed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host real equality, quotient streams, selected limits, trichotomy, choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2788.3 (Epsilon-delta limit Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ *packet reads the limit assertion through the finite route*

$$X, Y, a, L, E, D, W, R, H, C, P, N.$$

The consumer first sees the source and target $\mathsf{Met}^{\uparrow}$ *rows, the point row, the proposed limit row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *epsilon request and delta witness, the* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. Only then may it read the target* $\mathsf{Real}^{\uparrow}$ *distance seal carried by the target metric row, and it receives no host real equality, quotient stream, selected limit, trichotomy, choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4✓ `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitRealSealNonescape`

Proof. Use Definition 3759.2. It orders every Real-facing read after X, Y, a, L, E, D, W, R, so the target distance to the proposed limit is visible only as a readback routed through the displayed metric, dyadic, window, and regular-rational rows.

The target metric row Y is the only source of the $\mathsf{Real}^{\uparrow}$-valued distance seal. Because E and D are displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, the consumer cannot replace the certificate by a host epsilon quantifier, a hidden delta chooser, or a quotient representative of a target real.

The structural rows preserve the same route. Componentwise transport H, replay C, provenance P, and local naming N carry the finite packet forward without adding coordinates for host equality, selected limits, trichotomy, choice, quotient streams, or ambient completeness. □

Falsifiable prediction: Every epsilon-delta limit read of an $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet factors through displayed metric rows, dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, and the terminal RealUp distance seal.

Independence witness: metric, dyadicratcore, streamname, regseqrat, real: EpsilonDeltaLimitUp is not bijective to any listed sibling because it jointly carries source and target MetricSpace rows, a point row, a proposed limit row, dyadic epsilon and delta rows, finite windows, regular-rational readback, Real-facing distance seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$.

Constructive story: An $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet is generated from source and target MetricSpace rows, a point row, a proposed limit row, dyadic epsilon requests, dyadic delta witnesses, finite StreamName windows, RegSeqRat readback, Real-facing distance seals, hsame transport, Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3759.1, Definition 3759.2, and Definition 3759.3 seed the finite epsilon-delta limit interface for Phase Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected limits, trichotomy, choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance handling, finite-window readback, target Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EpsilonDeltaLimitUp.

Chapter 2789

A Concrete Naming Certificate for $\mathsf{PackingNumber}^{\uparrow}$

Following Bishop and Bridges and Kelley metric-space practice, the $\mathsf{PackingNumber}^{\uparrow}$ packet records the finite separated-center count dual to a displayed metric covering tolerance. It sits beside the finite net route of chapter 1767, the upper-count packet of chapter 2187, the scale-control surface of chapter 2035, and the metric source of chapter 57. The packet names a lower-bound quantitative ledger as finite BEDC data, not as a maximal subset, a choice-selected packing, a compactness proof, or an ambient cardinality comparison.

Definition 2789.1 (Metric packing-number carrier)**.** *A* $\mathsf{PackingNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, \varepsilon, U, D, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier, distance classifier, and metric-law ledger used by all separation reads.*
2. *ε is the positive tolerance row, read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with the displayed positivity evidence needed by separated-center consumers.*
3. *U is the finite closed* $\mathsf{ProbeBundle}$ *of candidate centers admitted for the tolerance ε.*
4. *D is the pairwise distance ledger. For every two distinct displayed centers of U, it records the metric-distance row in X and the comparison witnessing ε-separation.*
5. *B is the unary packing-budget row naming the number of center entries exposed by U and certified by the pairwise ledger D.*
6. *H records componentwise* hsame *transport for the metric source, tolerance row, separated-center bundle, pairwise distance ledger, budget row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from any packing-number request through the same rows $X, \varepsilon, U, D, B, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{PackingNumber}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, \varepsilon, U, D, B, H, C, P, N$. It has no coordinate for an ambient finite subset, host maximality proof, selected saturated packing, powerset cardinality, compactness, total boundedness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PackingNumberUp.TasteGate.PackingNumberTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberTasteGate single carrier alignment`

Theorem 2789.2 (Metric packing-number NameCert obligations)**.** *For every accepted* $\mathsf{PackingNumber}^{\uparrow}$ *carrier* $K = (X, \varepsilon, U, D, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, positive tolerance, finite separated-center bundle, pairwise distance ledger, unary packing budget, transport, replay, provenance, and local naming rows;*
2. *center admission, requiring every entry counted by* B *to be visible in the same closed* $\mathsf{ProbeBundle}$ *row* U *and admitted by the metric source row* X*;*
3. *pairwise separation exactness, requiring every two distinct displayed centers to be compared by* D *through the same metric distance and the same tolerance* ε*;*
4. *budget exactness, requiring the exported packing number to count exactly the visible center entries of* U *certified by the pairwise ledger;*
5. *tolerance transport, requiring the* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row to remain attached to the separation comparisons under* H *before any consumer reads the count;*
6. *ledger nonescape, excluding maximal subset selection, host cardinality comparison, powerset enumeration, open-cover compactness, total-bounded completion, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No metric packing-number obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberNamecertObligations`

Proof. Project the accepted carrier of Definition 3761.1. The analytic rows are exactly the metric source X, positive tolerance ε, separated-center bundle U, pairwise ledger D, and unary budget B.

Center admission is rowwise. A center can be counted only after it is visible in U and read as a point of the metric source X.

Pairwise separation is also rowwise. The ledger D supplies each displayed distance comparison between distinct entries of U, and every comparison uses the same positive tolerance row ε.

The budget row then names the quantitative boundary. Because B counts the same visible entries certified by U and D, a consumer cannot replace the packet by a host maximal packing, an external enumeration, or a powerset cardinality argument.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier contains none of the excluded compactness, choice, completeness, or maximality coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 2789.3 (Metric packing covering-dual handoff)**.** *Every* $\mathsf{MetricCoveringNumber}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{DoublingMetric}^{\uparrow}$*, or compact-metric consumer reading an accepted* $\mathsf{PackingNumber}^{\uparrow}$ *carrier factors through the finite route*

$$X, \varepsilon \longmapsto U, D, B \longmapsto H, C, P, N.$$

The consumer receives a separated-center lower-bound ledger before comparing it with a covering-number or finite-net upper-count ledger. The handoff exports no maximal packing, no selected cover, no open-cover compactness proof, no powerset cardinality comparison, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberCoveringDualHandoff`

Proof. Begin with a covering, finite-net, doubling, or compact-metric consumer of K. By Definition 3761.1, the consumer can enter the packet only through the displayed metric source X and the displayed positive tolerance row ε.

The separated data are opened next. The bundle U names the candidate centers, the ledger D certifies pairwise separation through the metric distance row, and B records the resulting unary packing budget.

The covering-dual comparison is external to the packing packet. A covering-number or finite-net consumer may compare its own center budget with B, but only after both packets expose their metric source, tolerance, finite bundle, and distance ledger rows.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Applying Definition 3761.2, the consumer cannot replace the finite separated ledger by a selected maximal packing, compactness argument, powerset cardinality comparison, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3761.1, Definition 3761.2, and Definition 3761.3 seed the quantitative separated-center packet over displayed metric sources, positive rational or dyadic tolerances, finite ProbeBundle centers, pairwise distance ledgers, unary budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close maximal packing existence, minimal covering duality, total boundedness, compact metric completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center admission, pairwise separation exactness, budget exactness, tolerance transport, ledger nonescape, and covering-dual handoff for BEDC.Derived.PackingNumberUp.

Constructive story: A $\mathsf{PackingNumber}^{\uparrow}$ packet is generated from a MetricUp source row, a positive RatUp or DyadicRatCoreUp tolerance, a finite ProbeBundle of separated centers, pairwise metric-distance comparisons, a unary packing budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2790

A Concrete Naming Certificate for $\mathsf{LipschitzFunction}^{\uparrow}$

Following Kelley metric-space practice and Bishop–Bridges constructive analysis, the $\mathsf{LipschitzFunction}^{\uparrow}$ packet records a function only through displayed metric rows, a graph ledger, and a carried bound. It sits between the source and target metric surface of chapter 57, the terminal real-bound seal of chapter 38, the graph discipline of chapter 130, and the tolerance export of chapter 369. The packet is a finite NameCert connector for uniform-continuity and completion-facing consumers; it does not import host function extensionality, quotient map equality, choice of representatives, or ambient completeness.

Definition 2790.1 (Lipschitz function carrier)**.** *A* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, G, K, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row carrying the function-read relation from X to Y.*
3. *K is the nonnegative* $\mathsf{Real}^{\uparrow}$ *Lipschitz-constant row.*
4. *D is the distance-inequality ledger recording every displayed comparison between a source-distance request and the corresponding target-distance response.*
5. *H records componentwise* hsame *transport for source points, target points, graph rows, bound rows, distance ledgers, continuation rows, provenance, and local naming.*
6. *C records displayed* Cont *replay from source metric requests through G, K, D to target metric responses and uniform-modulus reads.*
7. *P is the* Pkg *provenance over* $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$ *row.*

The classifier accepts only packets whose target-distance reads are sourced by G and bounded through the same K, D ledger. It has no coordinate for a host function object, quotient equality of maps, selected representatives, differentiability, compactness, or completeness of either metric space.

Lean4 ✓ `BEDC.Derived.LipschitzFunctionUp.LipschitzFunctionTasteGate single carrier alignment`

Theorem 2790.2 (Lipschitz function NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier* $F = (X, Y, G, K, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, componentwise classifier comparison, Lipschitz-bound exactness, graph-to-distance replay, uniform-modulus export, transport stability, provenance sourcing, and local naming through the displayed rows* $X, Y, G, K, D, H, C, P, N$*.*

No obligation row is stored outside the carrier, and no exported read depends on host function extensionality, quotient map equality, selected representatives, countable choice, differentiability, compactness, or ambient metric completeness.

Proof. Project the carrier. The only analytic rows are the source metric X, target metric Y, graph row G, Lipschitz constant K, and distance-inequality ledger D.

The classifier comparison is componentwise over those rows and over H, C, P, N. Bound exactness follows because a target-distance read has to pass through the graph row and then through the displayed comparison stored in D against the same constant K.

The uniform-modulus export reads no further object. Transport H and replay C preserve the same graph, constant, and inequality rows, while P and N only source and name the route. Since the carrier contains none of the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 2790.3 (Lipschitz function uniform-continuity handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier receives its source-tolerance route only after the packet has displayed the source metric* X*, target metric* Y*, graph row* G*, Lipschitz constant* K*, distance-inequality ledger* D*, and replay rows* H, C, P, N*. The handoff is a finite BEDC modulus route, not a rereading of a host Lipschitz proof.*

Proof. Start from the carrier and its NameCert obligations. The graph row G supplies the only function-read surface, while K and D supply the only bound and distance-comparison surface.

A uniform-modulus request therefore cannot bypass the displayed inequality ledger. It first fixes the target-distance request in Y, reads the bound row K, and then uses D to return the corresponding source-distance request in X.

The continuation and transport rows preserve this order. Hence uniform-continuity and continuous-map consumers receive a finite modulus handoff through $X, Y, G, K, D, H, C, P, N$, with no host function proof or selected representative added. □

Theorem 2790.4 (Lipschitz function completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier remains confined to the displayed metric rows, graph row, bound row, distance-inequality ledger, transport, replay, provenance, and local name rows. Reading a complete source or target metric in another packet does not turn* F *into a completed function-space object, a quotient of uniformly continuous maps, a selected extension, or a statement about limits outside* $X, Y, G, K, D, H, C, P, N$*.*

Proof. The packet carries only two metric rows and a graph-bound ledger between them. Completion-facing reads may reuse those metric rows, but no completed function-space coordinate is present in the carrier.

The distance ledger D is also local. It records comparisons for the displayed graph row and constant, and replay through C keeps those comparisons attached to the same source and target metric requests.

Thus a completion consumer can transport the finite Lipschitz route but cannot manufacture quotient maps, selected extensions, host limits, or ambient completeness claims. The exported surface remains exactly $X, Y, G, K, D, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3762.1, Definition 3762.2, Definition 3762.3, and Definition 3762.4 seed the finite Lipschitz-function route from metric, graph, real-bound, distance-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function extensionality, quotient map equality, selected extensions, differentiability, compactness, completed function spaces, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier comparison, Lipschitz-bound exactness, graph-to-distance replay, uniform-modulus export, transport stability, provenance sourcing, and local NameCert rows for BEDC.Derived.LipschitzFunctionUp.

Constructive story: A $\mathsf{LipschitzFunction}^{\uparrow}$ packet is generated from source and target metric rows, a ContinuousMap graph row, a RealUp Lipschitz constant, a distance-inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 2791

A Concrete Naming Certificate for $\mathsf{CantorTernarySet}^{\uparrow}$

Following Cantor's middle-third construction and Bishop–Bridges constructive analysis, the $\mathsf{CantorTernarySet}^{\uparrow}$ packet records the traditional ternary description of the Cantor set as finite BEDC data. It refines the retained-prefix and deleted-gap surface of chapter 1530 by making the inspected base-three windows explicit, and it routes every real-facing read through chapter 38, chapter 301, chapter 126, and chapter 291. The packet names only finite ternary windows, triadic endpoint ledgers, gap checks, regular-rational readback, and the terminal Real seal; it does not assert host powersets, infinite branch choice, quotient real equality, measure, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2791.1 (Cantor ternary set carrier)**.** *A* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, D, V, G, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite base-three digit positions currently inspected.*
2. *D is the ternary digit ledger for W, and every accepted digit row is tagged as a retained digit* 0 *or* 2*.*
3. *V is the triadic place-value row, evaluating the inspected prefix as a rational with denominator a displayed power of* 3*.*
4. *G is the deleted-middle gap ledger, listing the middle-third intervals excluded by every inspected non-retained alternative at the displayed depths.*
5. *I is the retained interval row inherited from* $\mathsf{CantorSet}^{\uparrow}$*, with endpoints computed from the same triadic prefix and checked against the gap ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from W, D, V, G, I.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular-rational readback has been displayed.*
8. *H records componentwise* hsame *transport for the window, digit, triadic value, gap, interval, readback, Real seal, continuation, provenance, and local name rows.*
9. *C records displayed* Cont *replay from window requests through the digit ledger, triadic value ledger, gap ledger, retained interval, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorTernarySet}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$ *row.*

The classifier accepts a packet only when $W, D, V, G, I, R, E, H, C, P, N$ are all displayed for the same finite triadic window, the digit ledger uses only retained digits, the gap ledger refuses the corresponding middle thirds, and every $\mathsf{Real}^{\uparrow}$ *consumer enters through R and E. It has no coordinate for a host subset of* $\mathsf{Real}^{\uparrow}$*, selected infinite ternary stream, quotient real equality, endpoint convention selector, measure, open-cover compactness, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CantorTernarySetUp.CantorTernarySetTasteGate single carrier alignment`

Theorem 2791.2 (Cantor ternary set NameCert obligations)**.** *For an accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier*

$$T = (W, D, V, G, I, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$ *surface consists exactly of carrier admission, finite ternary-window classification, retained-digit checking, triadic place-value evaluation, middle-third gap exclusion, retained-interval compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transport, continuation replay, provenance, and local naming through the displayed rows $W, D, V, G, I, R, E, H, C, P,$*

Proof. Carrier admission projects the finite tuple of Definition 3763.1. The window row W fixes the inspected positions, and the digit ledger D is the only source of ternary digit information for those positions.

The retained-digit and triadic-value checks are then internal to the same finite prefix. The row V evaluates exactly the inspected prefix, while the gap row G records the deleted middle thirds rejected by replacing a retained digit with the middle alternative.

The retained interval row I is inherited from the $\mathsf{CantorSet}^{\uparrow}$ surface and is therefore tied to the same prefix and gap ledger. The regular-rational row R and Real seal row E are downstream of that finite data.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host subsets, infinite streams, quotient equality, endpoint selectors, measure, compactness, choice, or ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 2791.3 (Cantor ternary set triadic nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier remains on the finite triadic route*

$$W, D, V, G, I \longmapsto R \longmapsto E$$

together with H, C, P, N. A consumer may inspect retained ternary windows, deleted middle thirds, triadic endpoint values, regular-rational readback, and the terminal Real seal, but it cannot extract a selected infinite ternary branch, a host real subset, quotient equality of real representatives, a measure row, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier. Any real-facing consumer must first enter through the displayed window W, because no digit row outside that finite window is part of the packet.

The digit ledger D, triadic value row V, and gap ledger G determine the retained middle-third decision at the inspected depths. The interval row I packages the same decision as a $\mathsf{CantorSet}^{\uparrow}$ retained-prefix row, so there is no separate branch or endpoint source.

The handoff to real data is forced through R and E. The regular-rational row R is reached only after the finite triadic rows have been displayed, and the Real seal E is downstream of R.

The structural rows H, C, P, N preserve and name this same route. Thus every exported read remains on the finite triadic packet, and the excluded host subset, infinite branch, quotient equality, measure, choice, compactness, and completeness data are not available to the consumer. □

Theorem 2791.4 (Cantor ternary set CantorSet dependency)**.** *Every accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *packet factors through the retained-prefix and deleted-gap surface of chapter 1530 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is read. After that CantorSet row is displayed, the remaining route uses* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows to reach the terminal real-name boundary, without asserting compactness, measure, host powersets, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3763.1. The row I is the retained interval row inherited from the CantorSet sibling, and it is computed from the same finite triadic prefix and deleted-gap ledger as W, D, V, G.

By Definition 3763.3, every real-facing read remains on the route $W, D, V, G, I \longmapsto R \longmapsto E$. Thus the CantorSet retained-prefix and deleted-gap row is consumed before the regular-rational readback R and Real seal E are available.

The remaining rows R, E, H, C, P, N use the RegSeqRat, StreamName, and DyadicRatCore surfaces named in the carrier provenance. None of those rows supplies compactness, measure, host powersets, or ambient real completeness, so the dependency is exactly the finite sibling consumption stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorTernarySet}^{\uparrow}$ packet is generated from finite StreamName windows, retained ternary digit rows, triadic place-value rows, deleted-middle gap rows, a retained CantorSet interval row, RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3763.1, Definition 3763.2, and Definition 3763.3 seed the explicit finite ternary-window carrier for the middle-third Cantor set through retained digits, triadic values, deleted gaps, regular-rational readback, and the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, selected infinite branches, quotient real equality, endpoint uniqueness, measure, compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained-digit classification, triadic place-value evaluation, gap exclusion, retained-interval compatibility, RegSeqRat readback, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CantorTernarySetUp.

Chapter 2792

A Concrete Naming Certificate for $\mathsf{CyclicTagSystem}^{\uparrow}$

Following Cook 2004 and Neary–Woods 2007, the $\mathsf{CyclicTagSystem}^{\uparrow}$ packet records cyclic tag systems as finite step-indexed rewriting data between the machine surface of chapter 280, the finite Rule 110 glider-catalog surface of chapter 1045, and the total-language boundary of chapter 9370. The packet names the alphabet, cyclic production list, deletion number, inspected tape window, bounded fuel evaluator, and Rule 110 handoff as displayed BHist rows. It does not import a partial halting function, an unbounded convergence theorem, a host evaluator, or Cook universality as a hidden coordinate.

Definition 2792.1 (Cyclic tag system carrier)**.** *A* $\mathsf{CyclicTagSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, P, D, W, F, R, H, C, Q, N)$$

with the following displayed rows:

1. *A is the finite alphabet row, including the distinguished blank or empty-word convention used by the inspected packet.*
2. *P is the cyclic production-list row, exposing the finite list of appendants and the current production index as BEDC data.*
3. *D is the deletion-number row, recording the positive deletion budget used by each accepted step.*
4. *W is the tape-window state row, containing only the displayed prefix, suffix guard, current word window, and step-index ledger.*
5. *F is the bounded fuel evaluator row, naming the finite replay of at most the displayed fuel many cyclic-tag steps.*
6. *R is the Rule 110 handoff row, read through the glider-catalog and cellular rows when a bounded cyclic-tag trace is presented to a Rule 110 consumer.*
7. *H records componentwise* hsame *transport for alphabet, production-list, deletion, tape-window, fuel, handoff, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from alphabet and production reads to one-step tape updates, from tape updates to bounded fuel evaluation, and from the bounded trace to the Rule 110 handoff row.*

9. *Q is the* Pkg *provenance over* CyclicTagSystem$^{\uparrow}$, TuringMachine$^{\uparrow}$, RuleOneTenGliderCatalog$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the rows $A, P, D, W, F, R, H, C, Q, N$. It has no coordinate for a host partial function, an unbounded halting predicate, an infinite tape quotient, a complete Rule 110 orbit, or a detached universality theorem.

Lean4 ✓ `BEDC.Derived.CyclicTagSystemUp.CyclicTagSystemTasteGate single carrier alignment`

Theorem 2792.2 (Cyclic tag system step-indexed Rule 110 handoff)**.** *For every accepted* CyclicTagSystem$^{\uparrow}$ *carrier $T = (A, P, D, W, F, R, H, C, Q, N)$, every finite fuel read factors through the cyclic-tag evaluator rows A, P, D, W, F before any Rule 110 orbit claim is exported through R. The handoff may cite the* TuringMachine$^{\uparrow}$ *source surface, the* RuleOneTenGliderCatalog$^{\uparrow}$ *finite catalog surface, and the* CellularAutomaton$^{\uparrow}$ *closed-observation route only after the bounded cyclic-tag replay has been displayed. Thus a Rule 110-facing consumer receives a finite step-indexed handoff and receives no halting decision, unbounded convergence theorem, host partial evaluator, or full universality proof.*

Proof. Project the accepted carrier. The alphabet row A, production-list row P, deletion-number row D, and tape-window row W are the only inputs from which a cyclic-tag step can be read.

The bounded evaluator row F is downstream of those rows. A fuel-indexed request must therefore replay the displayed one-step cyclic-tag updates before it can ask for any substrate-facing handoff.

The Rule 110 handoff row R is read only after F. At that point the consumer may cite the Turing-machine source chapter, the finite Rule 110 glider-catalog chapter, and the cellular closed-observation route, but only as sibling certificate surfaces attached to the displayed bounded replay.

Finally inspect H, C, Q, N. These rows transport, replay, source, and name the same bounded route. Since the carrier has no coordinate for halting, unbounded convergence, host partiality, or a complete universality proof, none of those objects can be exported with the Rule 110-facing handoff. □

Falsifiable prediction: Every CyclicTagSystem$^{\uparrow}$ Rule 110-facing read first exposes a finite alphabet row, cyclic production-list row, deletion-number row, tape-window row, bounded fuel evaluator row, transport row, continuation route, provenance row, and local naming row.

Independence witness: turingmachine, rule110glidercatalog, cellularautomaton: CyclicTagSystem$^{\uparrow}$ is not bijective to any listed sibling because it carries cyclic production-list and deletion-number rows together with a bounded tape-window evaluator before the Rule 110 handoff.

Closure status for NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$**.**

Constructive story: A CyclicTagSystem$^{\uparrow}$ packet is generated from a finite alphabet row, cyclic production-list row, deletion-number row, tape-window state row, bounded fuel evaluator row, Rule 110 handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$, seed)

Scope closed: Definition 3764.1 and Definition 3764.2 seed the finite cyclic-tag route between total BEDC step-indexing, TuringMachine$^{\uparrow}$, and the Rule 110 glider-catalog surface.

Formal verification: FormalStatus(NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close halting, unbounded convergence, host partiality, full universality, global Rule 110 orbit classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cyclic production-list replay, deletion-step exactness, bounded fuel evaluation, Rule 110 handoff nonescape, provenance, and local NameCert rows for the same packet.

Chapter 2793

A Concrete Naming Certificate for $\mathsf{HausdorffContent}^{\uparrow}$

Following Hausdorff and later constructive metric-measure treatments, the $\mathsf{HausdorffContent}^{\uparrow}$ packet records finite cover content data for metric subsets without importing a completed measure theory. It sits over the metric carrier of chapter 57, the finite measure-facing ledger of chapter 92, the retained-prefix surfaces of chapter 1530 and chapter 3763, and the terminal real-name seal of chapter 38. The packet names only displayed cover rows, rational radius and exponent budgets, refinement transport, content-bound replay, package provenance, and local NameCert evidence.

Definition 2793.1 (Hausdorff content carrier)**.** *A* $\mathsf{HausdorffContent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, A, R, S, U, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *row whose distance observations are sealed through* $\mathsf{Real}^{\uparrow}$*.*
2. *A is the inspected subset or source row, presented only by finite membership probes inherited from metric, Cantor, or ternary Cantor packets.*
3. *R is the rational radius-budget row, listing finitely many positive rational or dyadic radius bounds.*
4. *S is the rational exponent-budget row, recording the displayed content exponent and the arithmetic route used for each radius power.*
5. *U is the finite metric-cover row, pairing each listed cover center and radius with the source probes it covers.*
6. *B is the content-bound replay row, summing the displayed radius-power bounds into a* $\mathsf{Real}^{\uparrow}$*-sealed upper content value.*
7. *H is the componentwise* hsame *transport row for the metric source, subset probes, radius budgets, exponent budgets, cover rows, content bounds, provenance, and local name.*
8. *C is the displayed* Cont *replay route for cover refinement, radius-budget reuse, exponent-budget reuse, and content-bound replay.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffContent}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{CantorSet}^{\uparrow}$*,* $\mathsf{CantorTernarySet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$ *row.*

The classifier accepts a packet only when $M, A, R, S, U, B, H, C, P, N$ are displayed for the same finite cover request, every cover row uses a listed radius budget, every content-bound replay uses the listed exponent budget, and every metric, measure-facing, Cantor-facing, or real-facing consumer enters through the displayed rows. It has no coordinate for a host infimum over all covers, a choice-selected cover family, quotient real equality, σ-additive Hausdorff measure, or completed measure theory.

Lean4 ✓ `BEDC.Derived.HausdorffContentUp.HausdorffContentTasteGate single carrier alignment`

Theorem 2793.2 (Hausdorff content NameCert obligations)**.** *For an accepted* $\mathsf{HausdorffContent}^{\uparrow}$ *carrier*

$$Q = (M, A, R, S, U, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source admission through M, finite source-probe admission through A, rational radius-budget admission through R, rational exponent-budget admission through S, finite cover admission through U, content-bound replay through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission projects the packet in Definition 3765.1. The only metric data are in M, and the only source observations are the finite probes in A.

The radius and exponent budgets are displayed before any content value is read. Thus R and S determine the finite arithmetic route for the cover row U and the bound row B.

Transport and replay add no analytic coordinate. The rows H and C preserve the same metric, source, budget, cover, and bound rows, while P and N source and name the packet.

Since no carrier coordinate stores a host infimum, selected cover family, quotient real equality, σ-additive measure, or completed measure theory, the listed rows exhaust the local NameCert obligations. □

Theorem 2793.3 (Hausdorff content finite-cover refinement)**.** *Every refinement-facing read of an accepted* $\mathsf{HausdorffContent}^{\uparrow}$ *packet factors through the displayed route*

$$M, A, R, S, U, B, H, C, P, N.$$

If a finer cover is cited, it must reuse or refine the listed radius-budget row R, exponent-budget row S, finite cover row U, and content-bound replay row B through hsame *transport and displayed continuation. The refinement read cannot cite an ambient cover lattice, a host infimum over covers, a choice-selected minimizing cover, or quotient equality of real content values.*

Proof. Start with the finite cover coordinate. By Definition 3765.1, the row U is the only place where cover centers, radii, and covered source probes are displayed.

The budget rows R and S are upstream of U and B. A refinement can therefore change a cover only by displaying the reused or refined radius and exponent ledgers before the content-bound replay is read.

Use H and C to transport and replay the refinement route. The provenance row P and local name N keep the refined packet attached to the same metric, measure-facing, Cantor-facing, and real-facing source chapters.

The excluded host cover lattice, infimum, selected minimizing cover, and quotient real equality are not packet coordinates, so no refinement-facing consumer can obtain them from $\mathsf{HausdorffContent}^{\uparrow}$. □

Theorem 2793.4 (Hausdorff content Cantor cover handoff)**.** *Every* $\mathsf{CantorSet}^{\uparrow}$ *or* $\mathsf{CantorTernarySet}^{\uparrow}$ *consumer citing* $\mathsf{HausdorffContent}^{\uparrow}$ *receives only a finite cover-content handoff through*

$$A \longmapsto R, S \longmapsto U \longmapsto B$$

together with M, H, C, P, N*. The handoff can record finite retained-prefix probes, ternary-window probes, rational radius and exponent budgets, finite cover rows, and replayed content bounds. It does not supply* σ*-additive Hausdorff measure, a host infimum, a choice-selected cover, quotient real equality, or completed measure theory.*

Proof. Read the Cantor-facing source through A. The source row admits only finite retained-prefix or ternary-window probes inherited from chapter 1530 and chapter 3763.

Next read the budget and cover rows. The rational radius and exponent budgets R, S are displayed before the finite cover row U, and the content-bound replay row B is downstream of that cover.

Finally attach the metric and structural rows. The metric source M supplies the distance observations, while H, C, P, N transport, replay, source, and name the same finite handoff.

The measure-facing provenance does not create a completed measure. Because σ-additivity, host infima, choice-selected covers, quotient real equality, and completed measure theory are absent from the carrier, the Cantor consumer receives only the finite cover-content route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffContent}^{\uparrow}$ packet is generated from metric source rows, finite source probes, rational radius and exponent budgets, finite cover rows, content-bound replay, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3765.1, Definition 3765.2, Definition 3765.3, and Definition 3765.4 seed the finite cover-content carrier for metric, measure-facing, Cantor-facing, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close σ-additive Hausdorff measure, host infimum over all covers, choice-selected covers, quotient real equality, completed measure theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget admission, cover admission, refinement transport, content-bound replay, Cantor cover handoff, provenance, and local naming for BEDC.Derived.HausdorffContentUp.

Chapter 2794

A Concrete Naming Certificate for $\mathsf{CauchyTailFilterBornology}^{\uparrow}$

Following Bourbaki and Weil on uniform spaces and bounded structures, the $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ packet records bounded Cauchy-tail filters as finite BEDC rows. It sits between the finite stream schedule of chapter 126, the regular rational source discipline of chapter 301, the bounded-set carrier of chapter 1757, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names tail-bound surfaces for regular Cauchy data; it is not a quotient of streams, a choice-selected bound, trichotomy, or an ambient completeness theorem.

Definition 2794.1 (Cauchy tail filter bornology carrier)**.** *A* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, B, W, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are the only tail observations visible to the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row read through the same windows.*
3. *B is the* $\mathsf{BoundedSet}^{\uparrow}$ *tail-bound row, recording the displayed bound cell for the admitted tail window.*
4. *W is the tail-window row specifying the finite index after which the bound row is read.*
5. *M is the Cauchy-modulus compatibility row, matching the tail window to the tolerance requests carried by the regular sequence.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, B, W, M have been displayed.*
7. *H records componentwise* hsame *transport for schedule, regular source, bounded tail, window, modulus, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail-bound request through S, R, B, W, M, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$ *row.*

The carrier exports no quotient stream, selected bound, selected tail representative, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyTailFilterBornologyUp.CauchyTailFilterBornologyTasteGate single carrier alignment`

Definition 2794.2 (Cauchy tail filter bornology classifier)**.** *Two accepted* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ *carriers*

$$T = (S, R, B, W, M, E, H, C, P, N)$$

and

$$T' = (S', R', B', W', M', E', H', C', P', N')$$

are classified together exactly when the displayed hsame *rows transport* S *to* S'*,* R *to* R'*,* B *to* B'*,* W *to* W'*,* M *to* M'*, and* E *to* E'*, and the displayed* Cont *rows replay the same tail-window-to-bound-to-seal route on both packets. The classifier has no coordinate for equality of quotient streams, a least bound, a selected eventual index, trichotomy, or ambient completeness.*

Theorem 2794.3 (Cauchy tail filter bornology NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailFilterBornolog}$ *carrier* $T = (S, R, B, W, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *tail-window carrier admission, requiring* S, R, W *to display the finite observation window before any tail-bound row is consumed;*
2. *eventual boundedness classifier, requiring* B *to be preserved under the componentwise* hsame *transport of the accepted carrier and replayed by the same* Cont *route;*
3. *Cauchy-modulus compatibility, requiring* M *to attach each bound read to the same regular Cauchy source and finite window;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal handoff, requiring* R *and* E *to remain downstream of the displayed schedule, tail window, bound row, and modulus row;*
5. *ledger non-escape, requiring* H, C, P, N *to expose no quotient stream, selected bound, selected tail representative, trichotomy row, or ambient completeness theorem.*

No Cauchy tail filter bornology obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3766.1. The displayed tuple $S, R, B, W, M, E, H, C, P, N$ is the whole public surface, so tail-window carrier admission is exactly admission of those rows.

The eventual boundedness classifier is forced by the bound row. A tail-bound consumer can read B only through the finite window W over the stream schedule S and the regular source R, and classifier preservation transports that same row componentwise.

The Cauchy-modulus row supplies the compatibility check. It attaches the displayed bound to the tolerance and tail-window requests already visible in S, R, W, so the read cannot replace the packet by a host-level eventual bound or by a chosen index.

Finally, the Real-facing rows are downstream. The seal row E is reached only after S, R, B, W, M are displayed, and H, C, P, N preserve and name that route. Since the carrier has no coordinate for quotient streams, selected bounds, trichotomy, or ambient completeness, the five listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyTailFilterBornologyUp.CauchyTailFilterBornologyNameCertObligations`

Falsifiable prediction: Every $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ read factors through a finite StreamName schedule, a RegSeqRat source row, a BoundedSet tail-bound row, a finite tail-window row, a Cauchy-modulus compatibility row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, boundedset, real: $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite tail-window schedule, regular Cauchy source, bounded tail row, modulus compatibility, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3766.1, Definition 3766.2, and Definition 3766.3 seed the bounded tail-filter surface from finite StreamName windows through RegSeqRat readback, BoundedSet tail bounds, Cauchy-modulus compatibility, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected bounds, least bounds, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, eventual boundedness classification, Cauchy-modulus compatibility, RegSeqRat and Real seal handoff, ledger non-escape, and local naming for BEDC.Derived.CauchyTailFilterBornologyUp.

Constructive story: A $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ packet is generated from finite StreamName windows, a RegSeqRat source row, a BoundedSet tail-bound row, a finite tail-window row, Cauchy-modulus compatibility, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2795

A Concrete Naming Certificate for SequentialCompletenessCriterion$^{\uparrow}$

Following Bishop–Bridges and Kelley, the SequentialCompletenessCriterion$^{\uparrow}$ packet records the finite criterion by which sequence completion evidence is read as real sequential completeness data. It sits after the sequence-completion surfaces of chapter 1683 and chapter 3691, reads the real-facing seal of chapter 1544, and keeps the metric criterion aligned with chapter 1515.

Definition 2795.1 (SequentialCompletenessCriterion carrier)**.** *A* SequentialCompletenessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, C, R, M, W, G, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the* SequentialCompletion$^{\uparrow}$ *row naming the displayed sequence-completion interface consumed by the criterion.*
2. *C is the* CauchySequenceCompletion$^{\uparrow}$ *row naming the regular Cauchy sequence completion route available to the same finite packet.*
3. *R is the* RealSequentialCompleteness$^{\uparrow}$ *row, exposed only after the sequence-completion and Cauchy-sequence completion rows have been displayed.*
4. *M is the* SequentiallyCompleteMetric$^{\uparrow}$ *row, recording the metric criterion surface through which later metric consumers read sequential completeness.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row for the sequence observations inspected by the criterion.*
6. *G is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed stream windows and Cauchy schedules.*
7. *E is the* Real$^{\uparrow}$ *seal row reached from the regular readback and real sequential completeness rows.*
8. *H records componentwise* hsame *transport for S, C, R, M, W, G, E.*
9. *T records displayed* Cont *replay from sequence windows through Cauchy completion, metric criterion data, regular readback, and the real seal.*
10. *P is the* Pkg *provenance over* SequentialCompletenessCriterion$^{\uparrow}$, SequentialCompletion$^{\uparrow}$, CauchySequenceCompleti RealSequentialCompleteness$^{\uparrow}$, SequentiallyCompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite criterion packet. It has no coordinate for quotient completion, countable choice, host equality of sequences or completion points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompletenessCriterionUp.SequentialCompletenessCriterionTasteGate single carrier alignment`

Theorem 2795.2 (SequentialCompletenessCriterion NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompletenessCriterion}^{\uparrow}$ *carrier* $Q = (S, C, R, M, W, G, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletenes}}$ *surface consists exactly of carrier admission for the sequence-completion row, Cauchy-sequence completion row, real sequential-completeness row, sequentially complete metric row, stream-window row, regular readback row, real seal row, componentwise transport, displayed replay, provenance, and local naming. Any real sequential-completeness consumer must read*

$$S, C, W, G, R, M, E, H, T, P, N$$

before it can export a criterion result, and the export contains no quotient completion, countable choice, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3767.1. The rows $S, C, R, M, W, G, E, H, T, P, N$ are the only rows available to a consumer of the local name.

The sequence-facing part of the criterion is supplied by S and C. The finite observation schedule is supplied by W, and the regular rational readback is supplied by G. Thus the criterion never begins from a host sequence or from a quotient completion object.

The real-facing part is then forced through R, M, E. The real sequential-completeness row and the sequentially complete metric row name the criterion surface, while the $\mathsf{Real}^{\uparrow}$ seal can be read only after the displayed regular readback route has been consumed.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the packet contains no row for countable choice, host equality, open-cover compactness, quotient completion, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.SequentialCompletenessCriterionUp.SequentialCompletenessCriterionCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompletenessCriterion}^{\uparrow}$ packet is generated from a $\mathsf{SequentialCompletion}^{\uparrow}$ row, a $\mathsf{CauchySequenceCompletion}^{\uparrow}$ row, a $\mathsf{RealSequentialCompleteness}^{\uparrow}$ row, a $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3767.1 and Definition 3767.2 seed the finite criterion surface connecting sequence completion, Cauchy sequence completion, metric sequential completeness, regular readback, and real sequential completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completion, countable choice, host equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-completion handoff, Cauchy-sequence completion readback, metric criterion exposure, real seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 2796

A Concrete Naming Certificate for $\mathsf{RegularModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularModulus}^{\uparrow}$ packet records the explicit modulus row carried by a regular Cauchy rational name before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the terminal seal. It sits between the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the regular Cauchy naming surface of chapter 371, and the real-name seal of chapter 38. The packet is the finite modulus object consumed by Real and Completion routes; it does not choose limits, quotient streams, or host real equality.

Definition 2796.1 (Regular modulus carrier)**.** *A* $\mathsf{RegularModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, \mu, D, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row whose rational windows are already admitted through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*.*
2. *μ is the unary modulus row assigning, to each displayed precision request, a finite stream-window bound for the source row.*
3. *D is the dyadic tolerance ledger translating precision requests into* $\mathsf{DyadicRatCore}^{\uparrow}$ *radii and comparison budgets.*
4. *W is the finite window route showing that the window selected by μ is sufficient for the requested tolerance.*
5. *H records componentwise* hsame *transport for the source row, modulus row, dyadic ledger, finite window route, continuation replay, package provenance, and local name.*
6. *C records displayed* Cont *replay from a precision request through S, μ, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularModulus}^{\uparrow}$*,* $\mathsf{RegularCauchyName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the modulus row, dyadic tolerance ledger, and finite window route are displayed for the same source regular Cauchy row. It has no coordinate for

a selected limit, quotient stream equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularModulusUp.RegularModulusTasteGate single carrier alignment`

Theorem 2796.2 (Regular modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularModulus}^{\uparrow}$ *carrier* $M = (S, \mu, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$ *surface consists exactly of source regular Cauchy admission, unary modulus display, dyadic tolerance translation, finite window sufficiency, componentwise transport, continuation replay, package provenance, and local naming through the rows* S, μ, D, W, H, C, P, N*.*

Proof. Project the carrier of Definition 3768.1. The source row S is the only regular Cauchy data exposed by the packet, and the row μ is the only modulus coordinate attached to that source.

The dyadic ledger D translates each precision request into the tolerance language used by chapter 291. The window route W then records that the bound supplied by μ is sufficient for the inspected $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Finally H, C, P, N preserve, replay, source, and name the same displayed rows. Since selected limits, quotient streams, host real equality, countable choice, and ambient completeness have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2796.3 (Regular modulus RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularModulus}^{\uparrow}$ *packet must read the source regular Cauchy row through*

$$S, \mu, D, W \longmapsto H, C, P, N$$

before the regular-rational readback or terminal real-name seal is consumed. Thus the modulus is a displayed finite BEDC row on the Real and Completion route, not a hidden choice of limit, quotient representative, or host equality witness.

Lean4 ✓ `BEDC.Derived.RegularModulusUp.RegularModulusRegSeqRatHandoff`

Proof. Start with a consumer request for regular-rational or real-name data. The packet exposes the source only as S, so the consumer must read the unary modulus row μ and the dyadic tolerance ledger D before it can identify a sufficient finite window.

The row W supplies that finite window route. It is already tied to the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces named in the provenance row, and it reaches the $\mathsf{Real}^{\uparrow}$ seal only after the displayed tolerance check has been replayed.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. No excluded limit, quotient, host-equality, choice, or ambient-completeness coordinate is available, so every handoff remains on the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularModulus}^{\uparrow}$ packet is generated from a source regular Cauchy row, a unary modulus row, a dyadic tolerance ledger, a finite window route, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3768.1, Definition 3768.2, and Definition 3768.3 seed the finite modulus surface for regular Cauchy rational names on the Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert completeness, choice of limits, quotient streams, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus sufficiency, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularModulusUp.

Chapter 2797

A Concrete Naming Certificate for $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ packet records the constructive finite-subcover surface obtained when a located real interval is paired with displayed finite-net and local-coverage data. It consumes the located endpoint and radius discipline of chapter 394, the finite-net replacement surface of chapter 3471, and the terminal real-name boundary of chapter 38. The packet names explicit extraction of a finite subcover from a finite mesh route; it is not general open-cover compactness, sequential compactness, countable choice, or a host Heine-Borel theorem.

Definition 2797.1 (Compact located interval subcover carrier)**.** *A* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, N, C, M, E, H, R, P, L)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row, carrying the displayed endpoint names, radius budget, and locatedness reads.*
2. *N is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *row, carrying a finite mesh, mesh-scale ledger, and finite-net coverage route for the same interval.*
3. *C is the local coverage row: each finite mesh cell is paired with an admitted cover token and the finite witness that the cell is covered at the displayed tolerance.*
4. *M is the extraction row listing the selected finite cover tokens in the order read from the finite net.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located interval, finite net, and local coverage rows have been displayed.*
6. *H records componentwise* hsame *transport for interval, net, local coverage, extracted token list, seal, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from an interval-subcover request through I, N, C, M, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, sequential compactness without choice-like data, selected quotient representatives, trichotomy, countable choice, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactLocatedIntervalSubcoverUp.CompactLocatedIntervalSubcoverTasteGate single carrier alignment`

Theorem 2797.2 (Compact located interval subcover obligations)**.** *For every accepted* $\mathsf{CompactLocatedIntervalSub}$ *carrier* $S = (I, N, C, M, E, H, R, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$ *surface consists exactly of located interval admission through* I*, finite-net admission through* N*, local mesh-cell coverage through* C*, explicit finite token extraction through* M*, terminal real-name sealing through* E*, componentwise transport through* H*, displayed replay through* R*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 3769.1. The rows I, N, C, M, E expose the located interval, finite mesh, local coverage, extracted token list, and real-name seal before any consumer can read a subcover claim.

The obligation surface is finite. The only coverage evidence available to the packet is the row C, whose entries are indexed by the finite mesh row N and whose selected tokens are exactly the entries displayed in M.

Transport and replay do not add compactness principles. The rows H and R preserve the same finite route, while P and L record provenance and local naming for that route.

Because no arbitrary open-cover, sequential compactness, choice, quotient-representative, trichotomy, host-equality, or ambient-completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2797.3 (Compact located interval finite-net extraction)**.** *Every accepted* $\mathsf{CompactLocatedIntervalSubcove}$ *carrier exports a finite subcover list only by the route*

$$\mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{HeineBorelFiniteNet}^{\uparrow} \longrightarrow C \longrightarrow M \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the finite list M *is extracted from the displayed mesh cells and their local coverage witnesses; it is not chosen from an arbitrary open cover and is not obtained from sequential compactness.*

Proof. Start with the located interval row I. It fixes the endpoint and radius data whose finite-net request is read by N.

The row N supplies only a finite mesh route. The local coverage row C attaches cover tokens to that finite mesh, so the extraction row M can list exactly those displayed tokens.

The terminal seal E is downstream of the finite extraction. It records the $\mathsf{Real}^{\uparrow}$ boundary of the already displayed route and does not supply a separate compactness theorem.

Thus the exported subcover is the finite list read from M, transported by H and replayed by R. The excluded open-cover and sequential compactness principles have no coordinate from which a consumer could read them. □

Theorem 2797.4 (Compact located interval L10 scope)**.** *The* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ *packet is an L10 real/completion consumer of located interval and finite-net data. It may be used where a downstream real or completion argument needs a displayed finite subcover extracted from a located interval mesh. It does not close general Heine-Borel compactness, arbitrary open-cover compactness, sequential compactness without choice-like data, or any bridge to host compact spaces.*

Proof. The dependency route is already displayed in the carrier. It starts at $\mathsf{LocatedInterval}^{\uparrow}$, passes through $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, and reaches $\mathsf{Real}^{\uparrow}$ only after the local coverage and extraction rows are present.

Any L10 consumer can therefore inspect the finite mesh, the selected coverage tokens, and the real-name seal. No consumer receives an open-cover family, a sequence-to-subsequence selector, or a host compact-space object.

The scope claim follows by ledger exhaustion: P records the listed dependencies, L names the local packet, and all excluded compactness principles are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ packet is generated from a located interval row, a finite-net row, local mesh-cell coverage witnesses, an extracted finite token list, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3769.1, Definition 3769.2, Definition 3769.3, and Definition 3769.4 seed explicit finite-subcover extraction for located real intervals from finite-net data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Heine-Borel compactness, arbitrary open-cover compactness, sequential compactness without choice-like data, countable choice, selected quotient representatives, host interval equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net extraction, local coverage exactness, L10 nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactLocatedIntervalSubcoverUp.

Chapter 2798

A Concrete Naming Certificate for $\mathsf{RealAbsoluteValue}^{\uparrow}$

Following Bishop and Bridges on constructive absolute value for real names, the $\mathsf{RealAbsoluteValue}^{\uparrow}$ packet records absolute value as a finite NameCert read over the current L10 $\mathsf{Real}^{\uparrow}$ construction. It seals the dyadic endpoint of chapter 293 into regular real-name data without importing host real equality, trichotomy, or an ambient ordered-field object.

Definition 2798.1 (Real absolute-value carrier)**.** *A* $\mathsf{RealAbsoluteValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, \delta, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite read windows used to inspect the real name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name row for the input real, including the rational approximants and regularity ledger.*
3. *δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to choose dyadic approximants at each displayed window.*
4. *D is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint readback row, applying dyadic absolute value to the inspected dyadic cores.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability budget that bounds the replay of the endpoint readback across later windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that packages the resulting regular absolute-value name.*
7. *H records componentwise* hsame *transport for the schedule, regular-name data, dyadic tolerance, dyadic endpoint readback, modulus budget, and real seal.*
8. *C records the* Cont *replay routes from schedule inspection through dyadic readback to the sealed regular real name.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$ *row.*

The accepted carrier contains no host real equality test, no trichotomy branch, no order-completion shortcut, no RealAlgOrder completion claim, and no absolute-value operation detached from the displayed regular-name and dyadic endpoint rows.

Lean4 ✓ `BEDC.Derived.RealAbsoluteValueUp.RealAbsoluteValueTasteGate single carrier alignment`

Theorem 2798.2 (Real absolute-value NameCert obligations)**.** *For an accepted packet* $A = (S, R, \delta, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, dyadic absolute-value stability, regular-sequence seal compatibility, triangle nonescape, and ledger exactness for the displayed rows* $S, R, \delta, D, M, E, H, C, P, N$*.*

Proof. Carrier admission reads the schedule, regular-name data, dyadic tolerance, dyadic endpoint readback, stability budget, and real seal from the finite packet. The support rows H, C, P, N transport, replay, package, and name exactly those displayed rows.

Dyadic absolute-value stability is local to D and δ. The endpoint readback uses the dyadic absolute-value packet, while the tolerance row fixes the finite accuracy at which the dyadic core is inspected.

Regular-sequence seal compatibility is supplied by M and E. The modulus budget records enough stability to replay the endpoint readback across later windows, and the $\mathsf{Real}^{\uparrow}$ seal packages the resulting regular name without adding a host real comparison.

Ledger exactness follows by projection from the carrier. Every exported read is one of $S, R, \delta, D, M, E, H, C, P, N$ so no trichotomy branch, order-completion shortcut, or RealAlgOrder completion field can be recovered from the packet. □

Theorem 2798.3 (Real absolute-value triangle nonescape)**.** *Every triangle-facing consumer of an accepted* $\mathsf{RealAbsoluteValue}^{\uparrow}$ *packet reads only the regular-name seal* E *together with its schedule, dyadic tolerance, dyadic endpoint readback, modulus budget, componentwise transport, continuation replay, package provenance, and local naming rows. The packet therefore exposes the triangle surface as a stability obligation over regular real names and dyadic absolute-value readback, not as host real order, trichotomy, or a completed* $\mathsf{RealAlgOrder}^{\uparrow}$ *structure.*

Proof. Start with the carrier rows of Definition 3770.1. The only endpoint operation is the dyadic readback D, and the only real-facing result is the $\mathsf{Real}^{\uparrow}$ seal E.

The triangle-facing route must use the schedule and tolerance rows to compare finite read windows, then use M to replay the resulting regular-name stability. Transport by H and replay by C keep the route tied to those same displayed rows.

Package provenance P and local naming N identify the output as the $\mathsf{RealAbsoluteValue}^{\uparrow}$ surface. Since the carrier has no host real equality test, no trichotomy coordinate, and no RealAlgOrder completion field, a triangle consumer cannot escape the regular-name and dyadic-readback ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealAbsoluteValue}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat regular-name data, DyadicRatCore tolerance, DyadicAbsoluteValue endpoint readback, CauchyModulus stability budget, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$, seed)

Scope closed: Definition 3770.1, Definition 3770.2, and Definition 3770.3 bind the seed carrier, obligation surface, and triangle-facing nonescape boundary for constructive real absolute value.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealAbsoluteValueUp.RealAbsoluteValueTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close RealAlgOrderUp completion, host real equality, trichotomy, or bridge checking.

Upgrade path: Bridge checking requires a paper bridge from the checked carrier alignment to dyadic absolute-value stability, regular-sequence seal compatibility, triangle nonescape, and ledger exactness.

Chapter 2799

A Concrete Naming Certificate for $\mathsf{CUNSpectralAntenna}^{\uparrow}$

The $\mathsf{CUNSpectralAntenna}^{\uparrow}$ packet realizes the CUN tail spectral antenna theorem of Definition 9462.1457 as finite BEDC NameCert data. It reads the retention inequality, Perron response, positive self-energy row, and small-mass row from the six-boundary-tile spectral window, then exposes only the finite antenna rows needed by downstream trigger-spectrum consumers. The packet is not a biological-code oracle, stochastic evolution law, hidden Perron solver, quotient codon semantics, or unbounded spectral search.

The antenna packet consumes the spectral-closure and trigger-algebra sibling surfaces through the same separation row used below: chapter 3753 supplies the greedy spectral closure boundary, chapter 2270 supplies the stabilization read, and chapter 3478 supplies the finite trigger-algebra coverage surface. These sibling routes are the concrete sources for the S row and for the Pkg provenance row Q.

Definition 2799.1 (CUN spectral antenna carrier)**.** *A* $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, R, P, E, M, S, H, C, Q, N)$$

with the following displayed rows:

1. *T is the CUN tail row naming the finite tail cells and their attachment to the same six-boundary window as Definition 9462.1457.*
2. *R is the retention row displaying $\rho(M) = 19/30$ and its comparison with the WNR baseline.*
3. *P is the Perron-response row displaying $\lambda(M) = 0.675247931\ldots$ and the comparison with the same WNR baseline.*
4. *E is the self-energy row recording that the tail contributes a positive perturbation to the radial Perron mode.*
5. *M is the small-mass row recording the quasi-stationary tail mass boundary used by the source theorem.*
6. *S is the sibling separation row over* $\mathsf{GreedySpectralClosure}^{\uparrow}$, $\mathsf{SpectralStabilization}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, *and* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$.
7. *H records componentwise* hsame *transport for tail, retention, Perron-response, self-energy, small-mass, sibling-separation, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an antenna read through T, R, P, E, M, S, H.*
9. *Q is the* Pkg *provenance over* $\mathsf{CUNSpectralAntenna}^{\uparrow}$, $\mathsf{GreedySpectralClosure}^{\uparrow}$, $\mathsf{SpectralStabilization}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem Definition 9462.1457,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, P, E, M, S, H, C, Q, N$. It has no coordinate for hidden spectral computation, biological-code truth, stochastic dynamics, quotient codon equality, ambient eigenvector choice, or unbounded enumeration.

Theorem 2799.2 (CUN spectral antenna NameCert obligations)**.** *For every accepted* $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *carrier $A = (T, R, P, E, M, S, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$ *surface consists exactly of CUN tail admission, retention-row exposure, Perron-response exposure, self-energy exposure, small-mass exposure, sibling-separation exposure, componentwise transport, displayed replay, provenance, and local naming.*

The Lean follow-up `lean4/BEDC/Derived/CUNSpectralAntennaUp/TasteGate.lean` *must provide* Nontrivial $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *and* FieldFaithful $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *instances so that T, R, P, E, M, S, H, C remain separate displayed fields.*

Lean4 ✓ `BEDC.Derived.CUNSpectralAntennaUp.CUNSpectralAntennaTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3771.1. The rows T, R, P, E, M, S expose the finite tail, retention comparison, Perron response, self-energy mechanism, small-mass boundary, and sibling separation before any consumer can read an antenna conclusion.

The spectral rows are displayed table rows from the source theorem. They record the stated numerical comparisons and the self-energy interpretation, but they do not introduce a hidden Perron solver or a stochastic model.

The structural rows H, C, Q, N preserve that route. Since no host biological truth predicate, quotient codon equality, eigenvector choice, hidden spectral computation, or unbounded enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Theorem 2799.3 (CUN spectral antenna downstream separation)**.** *Every downstream antenna read of an accepted* $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *packet factors through*

$$T \longmapsto R, P, E, M \longmapsto S, H, C, Q, N.$$

The read may cite the finite CUN tail, retention row, Perron-response row, self-energy row, small-mass row, and sibling separation row. It cannot export a biological-code oracle, stochastic evolutionary law, hidden Perron solver, quotient codon equality, or unbounded spectral optimizer.

Proof. Start with the tail row T. It fixes the finite CUN tail cells read by the source theorem, so later consumers cannot range over an unbounded codon space.

Read the four spectral rows next. The retention row R, Perron-response row P, self-energy row E, and small-mass row M are displayed finite data tied to the same tail.

The sibling row S separates the packet from trigger closure, stabilization, radial wobble, Boolean trigger, and coverage-trigger algebra packets. The remaining rows H, C, Q, N transport, replay, source, and name the same antenna route.

The absent host objects listed in the statement are not coordinates of the carrier and cannot be exported by a consumer read. □

Falsifiable prediction: An accepted $\mathsf{CUNSpectralAntenna}^{\uparrow}$ antenna read is refuted if removing the CUN tail self-energy row preserves the same retention, Perron-response, small-mass, sibling-separation, transport, replay, provenance, and local naming claim.

Independence witness: greedy-spectral-closure, spectral-stabilization, radial-wobble-spectrum, boolean-trigger-lattice, window-coverage-trigger-algebra: none is bijective with $\mathsf{CUNSpectralAntenna}^{\uparrow}$ because the packet jointly carries CUN tail, retention, Perron response, self-energy, small-mass, sibling-separation, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$.

Constructive story: A $\mathsf{CUNSpectralAntenna}^{\uparrow}$ packet is generated from a finite CUN tail row, retention row, Perron-response row, self-energy row, small-mass row, sibling-separation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3771.1, Definition 3771.2, and Definition 3771.3 seed the finite CUN tail spectral antenna packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, quotient codon equality, hidden spectral computation, ambient eigenvector choice, unbounded optimization, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retention exactness, Perron-response exactness, self-energy preservation, small-mass exposure, Nontrivial CUNSpectralAntennaUp, and FieldFaithful CUNSpectralAntennaUp for BEDC.Derived.CUNSpectralAntennaUp.

Chapter 2800

A Concrete Naming Certificate for $\mathsf{FrechetCauchyCriterion}^{\uparrow}$

Following Frechet and Bishop–Bridges constructive analysis, the $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ packet records the finite criterion surface identifying convergence of a displayed real sequence with the Cauchy tail condition. It sits after the Real seal of chapter 38, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Real Cauchy criterion surface of chapter 1590, and the sequence-limit interface of chapter 1413. The packet exposes a choice-free equivalence route between a displayed limit row and a displayed Cauchy-tail row; it does not import quotient streams, countable choice, host real equality, or an ambient completeness theorem.

Definition 2800.1 (Frechet Cauchy criterion carrier)**.** *A* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, S, Q, D, K, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row naming the real surface on which the sequence and candidate limit are displayed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for source entries, candidate-limit reads, and late-tail comparisons.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radii and the dyadic budgets used for both convergence and tail comparisons.*
5. *K is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *row naming the accepted Cauchy-tail certificate for the displayed sequence.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row naming the accepted sequence-to-limit certificate over the same real source, windows, readbacks, and tolerances.*
7. *H records componentwise* hsame *transport for R, S, Q, D, K, L and the structural rows.*
8. *C records displayed* Cont *replay from a tolerance request through sequence windows, regular readback, dyadic budgets, Cauchy-tail admission, and sequence-limit admission.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, Q, D, K, L, H, C, P, N$. It has no coordinate for quotient stream equality, a selected subsequence, countable choice, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FrechetCauchyCriterionUp.FrechetCauchyCriterionTasteGate single carrier alignment`

Theorem 2800.2 (Frechet Cauchy criterion NameCert obligations)**.** *For an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier $F = (R, S, Q, D, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Real source, finite windows, regular rational readback, dyadic tolerance ledger, Cauchy-tail row, sequence-limit row, transport, replay, provenance, and local naming;*
2. *tail-to-limit compatibility, requiring the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *row K and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row L to use the same R, S, Q, D route before either certificate is consumed;*
3. *eventual-tail exactness, requiring each tolerance request to be represented by a displayed late window and dyadic budget in S, Q, D;*
4. *transport and replay stability, requiring H and C to preserve the same route under componentwise* hsame *comparison;*
5. *ledger non-escape, refusing quotient streams, selected representatives, countable-choice selectors, host real equality, and ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3772.1. The tuple exposes exactly the Real source, finite windows, readbacks, dyadic tolerances, Cauchy-tail certificate, sequence-limit certificate, and structural rows.

The compatibility row reads K and L only after R, S, Q, D have been displayed. Thus the Cauchy-tail side and the convergence side compare the same sequence observations and the same dyadic tolerance ledger.

Eventual-tail exactness is finite. Each accepted tolerance request is represented by a late window in S, regular rational data in Q, and a dyadic budget in D, so no hidden selector over tail positions is introduced.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded objects are not carrier coordinates, the local NameCert surface is exhausted by the displayed rows. □

Theorem 2800.3 (Frechet Cauchy criterion eventual-tail exactness)**.** *Let $F = (R, S, Q, D, K, L, H, C, P, N)$ be an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier. A convergence-facing read through L and a Cauchy-tail-facing read through K consume the same finite tail route*

$$R, S, Q, D \longmapsto K, L \longmapsto H, C, P, N.$$

For every displayed tolerance request, the route supplies a late finite window, a regular rational readback, and a dyadic comparison budget before either the limit certificate or the Cauchy-tail certificate can be exported.

Proof. Start with the carrier projection. The source and observation rows R, S, Q, D precede both certificate rows K and L.

Read the Cauchy-tail certificate K through chapter 1590. Its accepted route already requires finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance data before the Cauchy criterion is exposed.

Read the sequence-limit certificate L through chapter 1413. Its accepted route uses the same kind of finite tail-window schedule and dyadic ledger before any Real seal is consumed.

The Frechet packet identifies those routes at the displayed rows R, S, Q, D and then preserves them with H, C, P, N. Hence the eventual-tail data are exact inside the packet and cannot be replaced by a detached sequence observation. □

Theorem 2800.4 (Frechet Cauchy criterion Real seal nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier* $F = (R, S, Q, D, K, L, H, C, P, N)$ *is confined to the displayed route from the Real source through the finite windows, regular readback, dyadic tolerance ledger, Cauchy-tail row, sequence-limit row, and structural rows. The consumer cannot escape into quotient streams, a selected limit representative, countable choice, host real equality, or an ambient Real completeness principle.*

Proof. Project the carrier. The only Real-facing rows are the source row R and the sequence-limit row L, and both are attached to the finite observation and tolerance rows S, Q, D.

Use Definition 3772.2. The ledger non-escape obligation refuses quotient streams, selected representatives, choice, host equality, and ambient completeness as exported coordinates.

Use Definition 3772.3 to fix the terminal route. Since K and L are consumed only through the same displayed tail data and the structural rows preserve that route, no Real-facing consumer can introduce an additional seal or completion principle outside $R, S, Q, D, K, L, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ packet is generated from a RealUp source row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealCauchyCriterion row, a RealSequenceLimit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3772.1, Definition 3772.2, Definition 3772.3, and Definition 3772.4 seed the finite equivalence surface between convergence and Cauchy-tail data for real sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FrechetCauchyCriterionUp.FrechetCauchyCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, countable choice, host real equality, ambient Real completeness, or bridge checking.

Upgrade path: Bridge checking requires a paper bridge from the checked carrier alignment to tail-to-limit compatibility, eventual-tail exactness, transport stability, replay stability, provenance exhaustion, and Real seal nonescape.

Chapter 2801

A Concrete Naming Certificate for $\mathsf{LocatedCutReal}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedCutReal}^{\uparrow}$ packet records the constructive comparison surface where located lower and upper cut data are read as finite real-name evidence before any endpoint comparison is exported. It sits between the regular Cauchy real-name route of $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 38, chapter 301, chapter 126, and chapter 291, and the cut-facing siblings $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, and $\mathsf{DyadicLocatedCut}^{\uparrow}$ in **??**, chapter 436, and chapter 2089. The packet exposes located comparison as a finite NameCert route; it does not claim Dedekind completeness, quotient-real equality, trichotomy, countable choice, selected suprema, or an ambient host real order.

Definition 2801.1 (LocatedCutReal carrier)**.** *A* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, G, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the located lower-and-upper cut ledger. It records finite lower cells, upper cells, gap requests, and comparison witnesses in the style of the located cut surface.*
2. *G is the dyadic gap row over* $\mathsf{DyadicRatCore}^{\uparrow}$*. It supplies the displayed dyadic endpoint and radius cells by which a finite gap request is read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite observation windows opened by the comparison request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the endpoint-normalized windows as regular rational data before any real seal is consumed.*
5. *Q is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row. It records the finite locatedness evidence used by order, metric, and comparison consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It can be read only after D, G, W, R, Q have been displayed.*
7. *H records componentwise* hsame *transport for cut cells, dyadic endpoints, schedules, regular-rational handoff, located comparison, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for every consumer read through the same ordered route $D, G, W, R, Q, E, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCutReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicLocatedCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, W, R, Q, E, H, C, P, N$. It has no coordinate for quotient reals, selected cuts, Dedekind completeness, trichotomy, countable choice, or host analytic order.

Lean4 ✓ `BEDC.Derived.LocatedCutRealUp.LocatedCutRealTasteGate single carrier alignment`

Theorem 2801.2 (LocatedCutReal NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier*

$$L = (D, G, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the located cut ledger, dyadic gap row, finite stream schedule, regular-rational handoff, located real comparison row, terminal real seal, componentwise transport, displayed replay, provenance, and local naming.*

The obligation surface may cite the $\mathsf{DedekindCut}^{\uparrow}$ *lower-and-upper cut discipline, the* $\mathsf{DyadicLocatedCut}^{\uparrow}$ *endpoint route, the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, and the* $\mathsf{Real}^{\uparrow}$ *seal route. It cannot add a selected supremum, quotient-real representative, ambient completeness proof, trichotomy branch, or external comparison oracle.*

Proof. Project the carrier of Definition 3773.1. The rows D, G, W, R, Q, E expose the located cut cells, dyadic endpoint budgets, finite schedules, regular-rational readback, located real comparison, and terminal real seal in order.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier stores no quotient real, selected cut, completeness witness, trichotomy branch, choice selector, or external order oracle, the local NameCert surface is exactly the displayed finite route. □

Theorem 2801.3 (LocatedCutReal Cauchy comparison)**.** *Every comparison consumer of an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *packet factors through the finite Cauchy-facing route*

$$D, G, W, R, Q, E, H, C, P, N.$$

The read may use $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff data, the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It may not bypass those rows by invoking a host Dedekind-completeness theorem, quotient-real equality, selected supremum, trichotomy, countable choice, or an ambient real-order decision procedure.*

Proof. Begin with the located cut ledger D and dyadic gap row G. They provide only finite lower, upper, endpoint, radius, and gap cells, so the comparison request starts as a displayed packet route rather than as an ambient real-order statement.

Next read the schedule W and handoff row R. These rows force endpoint data to pass through finite stream windows and regular-rational budgets before any real-facing comparison is available.

The located comparison row Q and terminal seal E are downstream of that Cauchy-facing route. The rows H, C, P, N preserve the same carrier under transport and replay, so no consumer can export the excluded host objects while still consuming the accepted $\mathsf{LocatedCutReal}^{\uparrow}$ packet. □

Theorem 2801.4 (LocatedCutReal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier exports only the displayed located cut, dyadic endpoint, stream schedule, regular-rational handoff, and located comparison rows. In particular, it exports no Dedekind completeness principle, quotient real, selected cut representative, trichotomy branch, countable-choice selector, or host analytic order.*

Proof. Read the seal row E through Definition 3773.3. The seal is available only after D, G, W, R, Q have been displayed, and every transported or replayed use is still sourced and named by H, C, P, N.

The absent objects listed in the statement are not fields of Definition 3773.1. Therefore a terminal seal read can only expose the displayed finite comparison route and cannot be reinterpreted as an external completeness, quotient, choice, trichotomy, or host-order witness. □

Falsifiable prediction: An accepted $\mathsf{LocatedCutReal}^{\uparrow}$ comparison read is refuted if the located cut ledger, dyadic endpoint row, StreamName schedule, RegSeqRat handoff, LocatedReal comparison row, Real seal, transport, replay, provenance, or local naming can be removed while preserving the same packet claim.

Independence witness: real, regseqrat, streamname, dyadicratcore, locatedreal, dedekindcut, dyadic-located-cut: $\mathsf{LocatedCutReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located cut cells, dyadic endpoint budgets, finite schedules, regular-rational handoff, located comparison, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCutReal}^{\uparrow}$ packet is generated from located lower-and-upper cut ledgers, dyadic endpoint and gap budgets, StreamName finite windows, RegSeqRat handoff rows, LocatedReal comparison evidence, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3773.1, Definition 3773.2, Definition 3773.3, and Definition 3773.4 seed the finite constructive located-cut comparison surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedCutRealUp.LocatedCutRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient reals, selected suprema, trichotomy, countable choice, ambient host real order, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located cut exactness, dyadic endpoint transport, regular-Cauchy handoff, located comparison preservation, and Real seal nonescape for BEDC.Derived.LocatedCutRealUp.

Chapter 2802

A Concrete Naming Certificate for $\mathsf{DyadicStepApproximation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicStepApproximation}^{\uparrow}$ packet records the finite step approximation of a located real interval by dyadic cells, displayed sample values, and an oscillation ledger. It sits after the dyadic rational carrier of chapter 291, the located interval source of chapter 1709, the tagged finite-sum boundary of chapter 1615, the upper/lower sum surface of chapter 1880, and the regulated integration consumer of chapter 1557. The packet gives L10 Real and Completion routes one reusable finite approximation carrier before any quotient of functions, selected mesh limit, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 2802.1 (Dyadic step approximation carrier)**.** *A* $\mathsf{DyadicStepApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, M, L, V, O, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row, read through the endpoint-plus-radius surface of chapter 1709.*
2. *M is the dyadic mesh row from* $\mathsf{DyadicRatCore}^{\uparrow}$*. It records the common exponent scale, ordered dyadic breakpoints, adjacent cell widths, endpoint containment, and common-scale distance windows for the interval cells.*
3. *L is the finite cell ledger. It lists the carried cells of M, their inclusion in I, the boundary orientation rows, and the adjacency data consumed by Riemann and Darboux readers.*
4. *V is the step value row. For each carried cell it displays one value row and the finite readback route by which that value is available to* $\mathsf{RiemannSum}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *consumers.*
5. *O is the oscillation bound row. It records the displayed upper-minus-lower tolerance on each carried cell, using dyadic endpoint and radius comparisons from M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located source, dyadic mesh, cell ledger, step values, and oscillation bounds have been displayed.*
7. *H records componentwise* hsame *transport for I, M, L, V, O, E and for the replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an interval request through the dyadic mesh, finite cells, step values, oscillation bounds, terminal seal, transports, provenance, and local name.*
9. *P is the* Pkg *provenance over* DyadicStepApproximation$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RegulatedIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite step-approximation surface. It contains no quotient function equality, selected tag family outside L, limiting mesh process, open-cover compactness witness, countable-choice schedule, ambient Real$^{\uparrow}$ *completeness theorem, Lean verification, or bridge credential.*

Lean4 ✓ `BEDC.Derived.DyadicStepApproximationUp.DyadicStepApproximationTasteGate single carrier alignment`

Theorem 2802.2 (Dyadic step approximation obligation surface)**.** *For every accepted* DyadicStepApproximation$^{\uparrow}$ *carrier $A = (I, M, L, V, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, dyadic mesh, finite cell ledger, step value rows, oscillation bound rows, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *mesh exactness, requiring every carried breakpoint, width, and endpoint comparison to pass through the common-scale* DyadicRatCore$^{\uparrow}$ *row M;*
3. *interval coverage, requiring every cell in L to lie in the located interval source I through displayed endpoint and radius rows;*
4. *step-value readback, requiring every value row in V to be read through the same finite cells before a* RiemannSum$^{\uparrow}$ *or* DarbouxSum$^{\uparrow}$ *consumer can use it;*
5. *oscillation control, requiring O to compare the upper and lower cell values through the displayed dyadic distance-window ledger;*
6. *real-seal handoff, requiring the terminal* Real$^{\uparrow}$ *row E to be exposed only after I, M, L, V, O are displayed;*
7. *transport and replay stability, requiring H and C to preserve the same finite rows componentwise;*
8. *ledger non-escape, excluding quotient function equality, selected mesh limits, arbitrary tag choice, open-cover compactness, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No step-approximation obligation is stored outside these rows.

Proof. Project the carrier of Definition 3774.1. The analytic rows are precisely the located interval source I, dyadic mesh row M, finite cell ledger L, step value row V, oscillation bound row O, and terminal real seal E.

Mesh exactness is read from M. The dyadic source chapter 291 supplies the common exponent scale, breakpoint comparisons, distance-window rows, and radius ledgers used by every cell of the packet.

Interval coverage is read from I and L. The located interval chapter supplies the endpoint-plus-radius boundary, while the ledger lists exactly the cells whose endpoints are certified to lie inside that boundary.

Step-value readback and oscillation control are then finite. The row V gives one displayed value route for each carried cell, and O compares the upper and lower values on that same cell through the dyadic distance-window ledger. These are the rows consumed by the Riemann and Darboux sibling chapters.

Finally, the Real-facing route exposes E only after I, M, L, V, O are visible. The transport row H, continuation row C, provenance row P, and local name N repack the same finite surface. Since the carrier has no quotient-function coordinate, limiting mesh coordinate, arbitrary tag selector, compactness row, choice row, ambient completeness theorem, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 2802.3 (Dyadic step approximation integration handoff)**.** *Every* $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicStepApproximation}^{\uparrow}$ *carrier receives the finite route*

$$I, M, L, V, O \longmapsto E \longmapsto H, C, P, N.$$

The Riemann consumer reads the displayed cells and values as a tagged finite-sum boundary, the Darboux consumer reads the same cells and oscillation rows as upper/lower finite sums, and the regulated-integral consumer reads the packet as a step approximation ledger before any terminal $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start from the obligation surface of Definition 3774.2. It confines every accepted packet to the located source, dyadic mesh, finite cell ledger, value rows, oscillation rows, terminal seal, and structural rows.

For a Riemann consumer, apply the sibling surface of chapter 1615. The cell ledger L supplies the carried mesh cells and the row V supplies the finite value reads, so the consumer obtains a tagged finite-sum boundary rather than a hidden tag family.

For a Darboux consumer, apply chapter 1880. The same cells and value rows are read through O as upper/lower oscillation data, so the upper-minus-lower comparison remains a displayed finite ledger.

For a regulated-integral consumer, apply chapter 1557. The packet is consumed as a step approximation ledger before the $\mathsf{Real}^{\uparrow}$ seal E is public. Thus all three consumers receive the same finite route, and the excluded quotient, limit, compactness, choice, and ambient-completeness objects have no row through which to enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicStepApproximation}^{\uparrow}$ packet is generated from a located interval source, a $\mathsf{DyadicRatCore}^{\uparrow}$ mesh row, a finite cell ledger, step value rows, oscillation bound rows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3774.1, Definition 3774.2, and Definition 3774.3 seed the finite step-approximation carrier consumed by RiemannSum, DarbouxSum, and RegulatedIntegral routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient function equality, selected mesh limits, arbitrary tag choice, open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh exactness, interval coverage, step-value readback, oscillation control, real-seal handoff, transport stability, replay stability, provenance exhaustion, and integration handoff for BEDC.Derived.DyadicStepApproximationUp.

Chapter 2803

A Concrete Naming Certificate for $\mathsf{HausdorffUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley uniform spaces, the $\mathsf{HausdorffUniformSpace}^{\uparrow}$ packet names the separated uniform interface that lies between the entourage carrier of chapter 339 and the separated boundary already exposed by chapter 1876. It also gives the Hausdorff-reflection route of **??** a finite NameCert surface before any completion-facing consumer reads quotient points, host equality, or selected representatives.

Definition 2803.1 (Hausdorff uniform space carrier)**.** *A* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (U, S, R, A, T, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, including the finite diagonal, inverse, refinement, and composition data exposed by the source uniformity.*
2. *S is the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *classifier row, recording when two displayed points are separated by the available entourage observations.*
3. *R is the* $\mathsf{HausdorffReflection}^{\uparrow}$ *route row, naming the quotient-free reflection boundary as a consumer of U and S.*
4. *A is the antisymmetry ledger row, requiring indistinguishable point-pair reads to be identified only through the displayed separated classifier.*
5. *T records componentwise* hsame *transport for U, S, R, A, C, P, N.*
6. *C records displayed* Cont *replay from uniform, separated, and Hausdorff-reflection consumers through the same finite route.*
7. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one separated uniform-space packet. It contains no coordinate for quotient points, host equality of points, a completion object, selected representatives, countable choice, or ambient set-theoretic separation.

Lean4 ✓ `BEDC.Derived.HausdorffUniformSpaceUp.TasteGate.HausdorffUniformSpaceTasteGate single carrier alignment`

Theorem 2803.2 (Hausdorff uniform space NameCert obligations)**.** *For an accepted* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *carrier* $H = (U, S, R, A, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed uniform entourage row, separated classifier row, Hausdorff-reflection route row, antisymmetry ledger row, transport row, replay row, provenance row, and local naming row;*
2. *uniform-source exactness, requiring every separated or Hausdorff-reflection read to begin with the finite entourage data in* U*;*
3. *separated-classifier stability, requiring* S *and* A *to compare point-pair reads only through displayed uniform observations;*
4. *reflection non-escape, requiring* R *to expose the Hausdorff boundary without adding quotient points, host equality, selected representatives, or completion coordinates;*
5. *transport, replay, provenance, and local-naming stability for the same displayed row order.*

No Hausdorff uniform-space obligation is stored outside U, S, R, A, T, C, P, N*.*

Proof. Carrier admission is projection from Definition 3775.1. The projected coordinates are exactly the uniform row U, separated classifier row S, Hausdorff-reflection route R, antisymmetry ledger A, and structural rows T, C, P, N.

Uniform-source exactness follows by reading every separated and reflection-facing consumer through the same finite entourage surface used by chapter 339. The separated row S and antisymmetry ledger A add only classifier evidence on top of those entourage observations.

Reflection non-escape is the role of R. It may cite the separated packet of chapter 1876 and the reflection boundary of **??**, but it receives no quotient point, host equality, selected representative, completion object, or ambient separation principle.

The transport and replay rows T and C preserve that same route. Since P and N package and name only the displayed carrier rows, every excluded object remains outside the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffUniformSpace}^{\uparrow}$ consumer reads a $\mathsf{UniformSpace}^{\uparrow}$ entourage row, a $\mathsf{SeparatedUniformSpace}^{\uparrow}$ classifier row, and a $\mathsf{HausdorffReflection}^{\uparrow}$ route row before citing Hausdorff uniform separation.

Independence witness: uniformspace, separateduniformspace, hausdorffreflection: $\mathsf{HausdorffUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the entourage source, separated classifier, Hausdorff-reflection route, antisymmetry ledger, transport, replay, provenance, and local NameCert rows as one finite interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3775.1 and Definition 3775.2 seed the separated uniform-space packet over displayed entourage, separated classifier, Hausdorff-reflection route, antisymmetry, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffUniformSpaceUp.TasteGate.HausdorffUniformSpaceTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient point construction, host equality, selected representatives, completion existence, countable choice, or ambient set-theoretic separation.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exactness, separated-classifier stability, reflection non-escape, transport and replay stability, and ledger non-escape for BEDC.Derived.HausdorffUniformSpaceUp.

Constructive story: A HausdorffUniformSpace$^{\uparrow}$ packet is generated from UniformSpace entourage data, SeparatedUniformSpace classifier data, a HausdorffReflection route, an antisymmetry ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2804

A Concrete Naming Certificate for $\mathsf{SecantMethod}^{\uparrow}$

Following Traub and Ortega–Rheinboldt on iterative root-finding, the $\mathsf{SecantMethod}^{\uparrow}$ packet records the finite two-point root-finding surface used by the L10 completion-facing numerical-analysis route. It sits beside the Newton–Kantorovich packet of chapter 1785, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, and the terminal real-name seal of chapter 38. The chapter names only the displayed bracketing windows, two previous iterates, finite secant denominator checks, residual reads, next-iterate rows, Cauchy handoff, transport, replay, provenance, and local naming; it does not assert global convergence, a derivative bound, quotient equality of reals, a selected host limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2804.1 (Secant method carrier)**.** *A* $\mathsf{SecantMethod}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, X, Y, D, R, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-evaluation row, naming the finite* $\mathsf{Real}^{\uparrow}$ *input and residual observations used by the root-finding request.*
2. *X and Y are the two previous iterate rows, each exposed through* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the same* $\mathsf{StreamName}^{\uparrow}$ *window schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator-separation row, recording the finite nonzero secant-denominator comparison needed before the next iterate can be read.*
4. *R is the residual-control row, binding the two displayed residuals and the candidate next residual to finite dyadic tolerances.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the iterate, residual, denominator, and tolerance observations.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads every iterate and residual window as regular rational data.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, X, Y, D, R, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for function-evaluation, iterate, denominator, residual, window, readback, endpoint, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a secant request through $F, X, Y, D, R, W, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SecantMethod}^{\uparrow}$, $\mathsf{NewtonKantorovich}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$ *row.*

The classifier accepts the packet only through these finite rows. There is no coordinate for derivative availability, a completed limit point, quotient equality of real presentations, an unbounded convergence search, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SecantMethodUp.TasteGate.SecantMethodTasteGate single carrier alignment`

Theorem 2804.2 (Secant method NameCert obligations)**.** *For the carrier of Definition 3776.1, the local* $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (F, X, Y, D, R, W, Q, E, H, C, P,$ function-evaluation admission through F, two-iterate admission through X, Y, dyadic denominator-separation admission through D, residual-control admission through R, finite-window admission through W, regular-rational readback through Q, downstream* $\mathsf{Real}^{\uparrow}$ *sealing through E, component-wise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The function row F and iterate rows X, Y expose the only data from which the secant step may be read, and both iterate rows are tied to the same finite window surface.

The denominator and residual checks are displayed before the next-iterate read. The row D supplies a dyadic nonzero comparison for the secant denominator, while R records residual tolerances using the same finite observations.

Readback then passes through W and Q. The $\mathsf{StreamName}^{\uparrow}$ row supplies the requested windows, and $\mathsf{RegSeqRat}^{\uparrow}$ turns those windows into regular rational data before the downstream $\mathsf{Real}^{\uparrow}$ seal E is visible.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier contains no derivative row, limit selector, quotient equality, unbounded search, choice principle, or ambient completeness coordinate, the listed rows exhaust the local obligations. □

Theorem 2804.3 (Secant method Cauchy handoff)**.** *Every completion-facing read of an accepted* $\mathsf{SecantMethod}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{SecantMethod}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff first exposes finite iterate and residual windows, then regular rational readback, then dyadic denominator and residual budgets, and only then the terminal real-name seal. It supplies no independent convergence theorem, no derivative-based Newton certificate, no selected limit, and no ambient completeness principle.

Proof. Start with a completion-facing consumer. The carrier forces the request to read W and Q before a secant step can be interpreted, so the iterates and residuals are finite regular rational observations rather than host real representatives.

Next inspect the dyadic rows. The denominator-separation row D and residual-control row R provide the only permitted comparison data for the next iterate, and both are attached to the same displayed windows.

The $\mathsf{Real}^{\uparrow}$ seal E is reached after those rows have been replayed through C and transported through H. Thus the exported handoff is a finite Cauchy-facing route, not an assertion of global

convergence, derivative control, selected limits, quotient equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{SecantMethod}^{\uparrow}$ completion-facing read factors through finite StreamName windows, RegSeqRat readback, DyadicRatCore denominator and residual budgets, displayed secant-step rows, a downstream RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: newtonkantorovich, regseqrat, dyadicratcore, streamname, real: $\mathsf{SecantMethod}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two previous iterate rows, a finite secant denominator check, residual-control rows, regular rational readback, dyadic tolerance data, a downstream Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$.

Constructive story: A $\mathsf{SecantMethod}^{\uparrow}$ packet is generated from function-evaluation rows, two previous iterate rows, dyadic denominator-separation and residual-control rows, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3776.1, Definition 3776.2, and Definition 3776.3 seed the finite secant root-finding route from regular rational windows to the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global convergence, derivative-control hypotheses, quotient equality of real presentations, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, two-iterate admission, denominator-separation exactness, residual-control routing, Cauchy handoff, transport, replay, provenance, and local naming for BEDC.Derived.SecantMethodUp.

Chapter 2805

A Concrete Naming Certificate for $\mathsf{CauchyCompletionBind}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{CauchyCompletionBind}^{\uparrow}$ packet records the finite Kleisli-composition surface for Bishop-style completion: a completion-facing read may bind a source completion row to a displayed completion-valued route only through the already named $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ certificates. It is a local ledger for bind composition, not a quotient completion, not a selected representative, not countable choice, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

The bind packet consumes the completion monad of chapter 378, the unit admission of chapter 1527, the Kleisli consumer route of chapter 1845, the Bishop-regular completion boundary of chapter 1639, and the terminal Real seal of chapter 38 as concrete sibling rows rather than as background terminology.

Definition 2805.1 (CauchyCompletionBind carrier)**.** *A* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, U, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-completion row over* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, exposing the regular Cauchy input windows before any bind consumer is read.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row recording the unit admission for source points and terminal seal handoff.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row recording the single-bind consumer route and the reassociation schedule inherited from* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*.*
4. *E is the completion seal row that records which displayed* $\mathsf{Real}^{\uparrow}$ *or Bishop-regular completion boundary the bind route may consume.*
5. *H is the componentwise* hsame *transport row for the source, unit, Kleisli, seal, replay, provenance, and local naming data.*
6. *C is the displayed* Cont *replay row for the finite bind route.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionBind}^{\uparrow}$*,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$*,* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$ *row naming the bind packet.*

The carrier has no coordinate for quotient Cauchy classes, selected representatives, countable-choice schedules, host equality of completed objects, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionBindUp.CauchyCompletionBindTasteGate single carrier alignment`

Theorem 2805.2 (CauchyCompletionBind NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *carrier* $B = (S, U, K, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$ *surface consists exactly of source-completion exposure, unit admission, Kleisli bind exposure, completion-seal compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every bind consumer must first read the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$*,* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows recorded in* P*.*

Proof. Project the accepted carrier. The rows S, U, K, E are the only analytic rows available to a bind consumer, and each is already sourced from the named completion, unit, Kleisli, Bishop-regular, and real-seal siblings.

Transport and replay are structural. The row H preserves the same source, unit, Kleisli, and seal coordinates under displayed hsame routes, while C replays the same finite bind path through the consumer surface.

Finally P and N source and name the local certificate. Since the carrier contains no quotient class, representative selector, choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 2805.3 (CauchyCompletionBind associativity ledger)**.** *Let* $B = (S, U, K, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *packet. Any two public reads of a two-step completion bind that use the same displayed source-completion row* S*, unit row* U*, Kleisli row* K*, completion seal* E*, transport row* H*, replay row* C*, provenance row* P*, and local name* N *have the same finite associativity ledger. The ledger is a displayed route equality of BEDC rows; it is not quotient completion equality, not equality of selected representatives, not countable choice, and not a proof that* $\mathsf{Real}^{\uparrow}$ *is complete in the ambient host.*

Proof. Read the two-step bind only through the carrier rows. The first route and the reassociated route both project the same S, U, K, E coordinates, so their source completion, unit admission, Kleisli schedule, and seal handoff are the displayed data of one packet.

Use H to transport the coordinates componentwise and C to replay the same finite consumer path. These structural rows compare the bind schedules inside the packet rather than by host equality of completed objects.

The provenance and naming rows attach the comparison to the same $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources. No excluded quotient, representative, choice, or ambient-completeness object is available to alter the ledger. □

Theorem 2805.4 (CauchyCompletionBind real-completion forward link)**.** *Every accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *consumer that exposes a completion seal must first route through chapter 378, chapter 1527, chapter 1845, chapter 1639, and chapter 38. The exposed seal is therefore read only after the source-completion row* S*, unit row* U*, Kleisli row* K*, completion seal row* E*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *have been displayed. It exports no quotient completion class, selected representative, countable-choice schedule, host equality of completed objects, or ambient completeness theorem.*

Proof. Project the accepted carrier $B = (S, U, K, E, H, C, P, N)$. The source row S and seal row E are completion-facing only because the provenance row P names the Cauchy monad, Bishop-regular completion, and Real seal siblings.

Read the bind route through U and K. The unit row supplies the source admission boundary, and the Kleisli row supplies the displayed consumer route before any completion seal can be exposed.

The structural rows H and C only transport and replay the same finite route. Together with P and N, they keep the consumer inside the cited sibling certificates, so the packet has no coordinate from which to recover quotient completion equality, selected representatives, countable choice, host completed-object equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{CauchyCompletionBind}^{\uparrow}$ read is refuted if a completion consumer obtains a two-step bind without first exposing the source completion row, unit row, Kleisli row, completion seal, transport, replay, provenance, and local naming rows.

Independence witness: cauchycompletionmonad, cauchycompletionunit, cauchycompletionkleisli, bishopregularcauchycompletion, real: none is bijective with $\mathsf{CauchyCompletionBind}^{\uparrow}$ because the bind packet jointly carries the source-completion row, unit admission, Kleisli consumer route, completion seal, componentwise transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionBind}^{\uparrow}$ packet is generated from source completion rows, unit rows, Kleisli rows, a completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3777.1, Definition 3777.2, and Definition 3777.3 bind completion-facing Kleisli composition to finite source, unit, seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not assert quotient completions, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CauchyCompletionBind}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, source-completion exposure, unit compatibility, Kleisli bind exposure, associativity-ledger exactness, and non-escape for BEDC.Derived.CauchyCompletionBindUp.

Chapter 2806

A Concrete Naming Certificate for $\mathsf{GlobalFiberSpectra}^{\uparrow}$

The $\mathsf{GlobalFiberSpectra}^{\uparrow}$ packet realizes the global-fiber spectra branch of Definition 9462.1167 and Definition 9462.1169 as finite BEDC NameCert data. It sits beside the six-boundary tile coverage packet of chapter 3470, the window-coverage trigger algebra of chapter 3478, and the greedy spectral closure packet of chapter 3753; those siblings expose coverage, trigger, and closure faces, while this packet records the global tripleton, singleton, edge-spectrum, tile, trigger, transport, replay, provenance, and naming rows together.

Definition 2806.1 (GlobalFiberSpectra carrier)**.** *A* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, T, E, L, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the global codon-cube fiber row recording tripleton, singleton, doubleton, quartet, and sextupleton multiplicities for the standard-code classifier.*
2. *T is the complete global fiber tile row recording the displayed Ile/Met and Stop/Trp $3+1$ tiles.*
3. *E is the edge-spectrum row recording same-class and cross-class Hamming edge counts by bit direction.*
4. *L is the sibling tile-coverage row over* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *and finite standard-tile enumeration data.*
5. *R is the trigger handoff row over* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *and* $\mathsf{GreedySpectralClosure}^{\uparrow}$*.*
6. *H is the componentwise* hsame *transport row for the fiber, tile, edge, coverage, trigger, replay, provenance, and local naming data.*
7. *C is the displayed* Cont *replay row for standard-code reads.*
8. *P is the* Pkg *provenance over* $\mathsf{GlobalFiberSpectra}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{GreedySpectralClosure}^{\uparrow}$*, finite codon-cube tiles,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden genetic-code oracle, biochemical causation claim, infinite codon family, non-displayed trigger semantics, or host probability model.

Lean4 ✓ `BEDC.Derived.GlobalFiberSpectraUp.GlobalFiberSpectraCarrier event flow rows`

Theorem 2806.2 (GlobalFiberSpectra NameCert obligations)**.** *For every accepted* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *carrier* $G = (F, T, E, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$ *surface consists exactly of global fiber multiplicity, complete global fiber tile exposure, edge-spectrum exposure, six-boundary tile coverage, trigger handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A standard-code read cannot expose trigger interpretation before the global tripleton, singleton, edge-spectrum, tile, and sibling coverage rows have been displayed.*

Proof. Project the carrier rows. The fiber row F, tile row T, and edge row E are the only global-spectrum data available to the packet.

Route sibling dependence through L and R. The coverage row keeps the packet tied to six-boundary and standard-tile data, while the trigger handoff row records the window-coverage and greedy-closure interfaces without exposing trigger interpretation early.

Use H, C, P, N to transport, replay, source, and name the same finite route. Since no hidden oracle, biochemical causation coordinate, infinite family, non-displayed trigger semantics, or host probability model is present, the listed rows exhaust the NameCert surface. □

Theorem 2806.3 (GlobalFiberSpectra sibling separation)**.** *An accepted* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *packet is not determined by any one of* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*,* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *or finite standard-tile enumeration alone. The packet jointly carries global fiber multiplicity* F*, complete* $3 + 1$ *tile rows* T*, Hamming edge-spectrum rows* E*, sibling coverage* L*, trigger handoff* R*, transport* H*, replay* C*, provenance* P*, and local naming* N*.*

Proof. Read each sibling as a proper face. Six-boundary coverage supplies tile coverage, window-coverage trigger algebra supplies trigger handoff, greedy spectral closure supplies closure rows, and standard-tile enumeration supplies finite local tiles.

The global fiber and edge rows are additional coordinates. They record global tripleton, singleton, and direction-sensitive edge spectra before trigger interpretation is exposed.

Therefore no sibling alone is bijective to the packet. The structural rows H, C, P, N preserve the joint route and prevent any one face from replacing the whole $\mathsf{GlobalFiberSpectra}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{GlobalFiberSpectra}^{\uparrow}$ standard-code read factors through displayed global tripleton, singleton, edge-spectrum, and tile rows before any trigger interpretation is exposed.

Independence witness: sixboundarytilecoverage, windowcoveragetriggeralgebra, greedyspectralclosure, standardtileenumeration: $\mathsf{GlobalFiberSpectra}^{\uparrow}$ is not bijective to these siblings because it jointly carries global fiber multiplicity, local tile spectrum, edge counts, trigger handoff, transport, replay, provenance, and local naming.

The Lean follow-up in `lean4/BEDC/Derived/GlobalFiberSpectraUp.lean` must provide explicit Nontrivial $\mathsf{GlobalFiberSpectra}^{\uparrow}$ and FieldFaithful $\mathsf{GlobalFiberSpectra}^{\uparrow}$ TasteGate instances preserving $F, T, E, L, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GlobalFiberSpectra}^{\uparrow}$ packet is generated from finite codon-cube fiber rows, complete tile rows, edge-spectrum rows, sibling coverage rows, trigger handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$, seed)

Scope closed: Definition 3778.1, Definition 3778.2, and Definition 3778.3 bind the global-fiber spectra vision to finite codon-cube fiber, tile, edge, trigger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert biochemical causation, hidden genetic-code optimality, host probability semantics, non-displayed trigger interpretation, Lean verification, or bridge checking for GlobalFiberSpectra$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, global-fiber exposure, edge-spectrum exactness, sibling-separation exactness, Nontrivial and FieldFaithful instances, and trigger nonescape for BEDC.Derived.GlobalFiberSpectraUp.

Chapter 2807

A Concrete Naming Certificate for MetaCICNormalizationAuditPacket$^{\uparrow}$

The MetaCICNormalizationAuditPacket$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization surface of chapter 9383 as finite NameCert data. It sits beside the normalization audit packet of chapter 750, the critical-path packet of chapter 765, the normalization frontier of chapter 892, the candidate-evidence strong-normalization route of chapter 758, and the typed-normalization audit of chapter 3539. The packet names the closed typed term row, candidate or candidate-set evidence, candidate-to-SN projection, no-infinite-reduction projection, application and Pi adequacy rows, refusal frontier, transport, replay, provenance, and local naming rows needed before a public SN read may be cited. It does not assert unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Definition 2807.1 (MetaCIC normalization audit packet carrier)**.** *A* MetaCICNormalizationAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, K, A, S, I, P, F, H, C, G, N)$$

with the following displayed rows:

1. *T is the closed typed term row, exposing the term, type, empty-context read, and closedness data.*
2. *K is the candidate-evidence row, carrying* Candidate*, closed-candidate, or candidate-set membership evidence for the displayed term.*
3. *A is the application adequacy row, recording the function-side Pi evidence, argument evidence, and application-candidate handoff used when the term route contains an application.*
4. *S is the strong-normalization projection row, reading SN only through the candidate or candidate-set evidence displayed in K.*
5. *I is the no-infinite-reduction row, projecting the same candidate-mediated SN evidence to the absence of infinite reduction chains.*
6. *P is the Pi-candidate adequacy row, recording the Pi route used by dependent function cases without replacing K.*
7. *F is the normalization-frontier refusal row, kept visible whenever a requested citation would erase candidate evidence, subject-reduction boundaries, or confluence boundaries.*

8. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, application adequacy, SN projection, no-infinite-reduction projection, Pi adequacy, frontier, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a normalization-audit request through $T, K, A, S, I, P, F, H, G, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$*, chapter 9383,* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$*,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, A, S, I, P, F, H, C, G, N$. No row stores a theorem that every closed typed MetaCIC term normalizes without candidate or candidate-set evidence.

Theorem 2807.2 (MetaCIC normalization audit packet NameCert obligations)**.** *For the carrier of Definition 3779.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for $M = (T, K, A, S, I, P, F, H, C, G, N)$, closed typed term admission through T, candidate or candidate-set evidence admission through K, application adequacy through A, candidate-mediated SN projection through S, no-infinite-reduction projection through I, Pi-candidate adequacy through P, frontier retention through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through G, and local naming through N.*

```
Lean4 ✓  BEDC.Derived.MetaCICNormalizationAuditPacketUp.MetaCICNormalizationAuditPacketCarrier
namecert obligations
```

Proof. Project the carrier. The only term source is T, and the only evidence source for a public SN read is the candidate row K.

Read A and P as adequacy rows for application and Pi cases. They refine the same candidate-mediated route and do not create an evidence-free normalization source.

The SN row S is downstream of K. The no-infinite-reduction row I is downstream of S, so absence of infinite reduction chains is cited only through the same candidate or candidate-set evidence.

The frontier row F records requests that would erase the evidence boundary. Transport, replay, provenance, and local naming through H, C, G, N preserve these rows without adding subject reduction, confluence, quotient syntax equality, host evaluator soundness, or closed consistency. □

Theorem 2807.3 (MetaCIC normalization audit packet candidate handoff)**.** *Every public strong-normalization or no-infinite-reduction read from an accepted* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ *carrier factors through the finite route*

$$T \longmapsto K \longmapsto (A, P) \longmapsto S \longmapsto I \longmapsto F.$$

In particular, a public SN read must pass through candidate or candidate-set evidence before the closed typed term can be read as strongly normalizing, and the exported packet is not an unconditional normalization theorem for all closed typed MetaCIC terms.

Proof. Start with a public SN request. The term row T displays the closed typed term, but the carrier does not allow T alone to be read as normalized.

The request must next pass through K. Candidate, closed-candidate, or candidate-set evidence is the only input accepted by the projection row S, while A and P supply only the application and Pi adequacy data required by that same route.

The no-infinite-reduction row I reuses the output of S. If a consumer asks for evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or closed consistency, the frontier row F keeps that request outside the packet.

Thus the handoff exports candidate-mediated SN and no-infinite-reduction evidence, not a theorem that every closed typed MetaCIC term normalizes without the displayed candidate or candidate-set rows. □

Theorem 2807.4 (MetaCIC normalization candidate interface composition)**.** *An accepted* MetaCICNormalizationA *carrier whose candidate row K is accepted exports the three-step route from candidate evidence to strong normalization and then to no-infinite-reduction evidence:*

$$K \longmapsto S \longmapsto I.$$

The route preserves the closed typed term row T, application and Pi adequacy rows A, P, frontier row F, and structural rows H, C, G, N. It does not erase candidate evidence into an unconditional closed-term strong-normalization claim.

Proof. Apply Definition 3779.2. The local NameCert surface exposes the candidate row K, the strong-normalization projection S, and the no-infinite-reduction projection I as distinct displayed rows.

Apply Definition 3779.3. Public SN and no-infinite-reduction reads must pass through the route $T \longmapsto K \longmapsto (A, P) \longmapsto S \longmapsto I \longmapsto F$, so the candidate evidence remains visible at the interface.

The Lean-facing normalization API follows the same discipline: `candidate_implies_sn` reads candidate evidence before SN, `closed_typed_term_sn_from_candidate` exports closed typed SN from that evidence, and `closed_typed_term_no_infinite_reduction_from_candidate` projects the resulting SN evidence to absence of infinite reductions. These names record the conditional route; they do not license an evidence-free theorem for every closed typed MetaCIC term. □

Theorem 2807.5 (MetaCIC normalization audit packet residual-candidate route)**.** *For an accepted* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ *carrier $M = (T, K, A, S, I, P, F, H, C, G, N)$, every residual normalization or no-infinite-reduction read must expose candidate or candidate-set evidence in row K before application adequacy A, Pi adequacy P, strong-normalization projection S, and no-infinite-reduction projection I can be consumed. The structural rows H, C, G, N preserve the displayed route, and the frontier row F prevents evidence-free residual normalization from being exported.*

Proof. Project the carrier of Definition 3779.1. The rows $T, K, A, S, I, P, F, H, C, G, N$ are the full classifier-visible surface for residual normalization and no-infinite-reduction reads.

Use Definition 3779.2. It identifies K, A, S, I, P, F as the candidate-evidence row, application adequacy row, SN projection row, no-infinite-reduction projection row, Pi adequacy row, and frontier row.

Apply Definition 3779.3 and Definition 3779.4. Together they order candidate evidence before S and I, while A and P remain adequacy rows attached to the same candidate-mediated route.

Finally use row exhaustion for F, H, C, G, N. The frontier row records requests that would erase candidate evidence, and the structural rows transport, replay, source, and name the displayed route without creating evidence-free normalization. □

Falsifiable prediction: Every public strong-normalization or no-infinite-reduction read of a $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ carrier factors through closed typed term data, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated SN projection, no-infinite-reduction projection, frontier retention, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: metacicconfluenceauditpacket, metaciccriticalpath, metacicnormalizationfrontier, closedtermsubstitutionboundary: $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries closed typed term data, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated SN projection, no-infinite-reduction projection, frontier retention, transport, replay, provenance, and local naming as one audit carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ packet is generated from a closed typed term row, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated strong-normalization projection, no-infinite-reduction projection, frontier retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3779.1, Definition 3779.2, and Definition 3779.3 seed the finite candidate-mediated normalization audit packet for public SN and no-infinite-reduction reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetaCICNormalizationAuditPacketUp.MetaCICNormalizationAuditPacketCarrier\_namecert\_obl`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Upgrade path: Obligation closure requires checked carrier admission, candidate-evidence exactness, application and Pi adequacy routing, SN projection, no-infinite-reduction projection, frontier retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalizationAuditPacketUp.

Chapter 2808

A Concrete Naming Certificate for $\mathsf{StepFunction}^{\uparrow}$

Following Bourbaki and Bishop–Bridges constructive analysis, the $\mathsf{StepFunction}^{\uparrow}$ packet records a finite partition-to-real approximation object as BEDC NameCert data. It sits after the partition surface of chapter 201, the dyadic step-function specialization of chapter 428, the tagged finite-sum route of chapter 1615, the integration boundary of chapter 123, and the regular rational readback of chapter 301. The packet supplies a textbook finite step-function interface without forcing every integration-facing consumer through dyadic endpoints alone.

Definition 2808.1 (Step function carrier)**.** *A* $\mathsf{StepFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, V, A, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is a* $\mathsf{Partition}^{\uparrow}$ *row listing finitely many ordered cells and their boundary incidence data.*
2. *V assigns a displayed* $\mathsf{Real}^{\uparrow}$ *value row to each cell of P.*
3. *A records the constant-on-cell admission relation, so every accepted domain read first chooses a cell of P and then reads the corresponding value in V.*
4. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *comparison row, present when the partition cells and values are supplied through dyadic subrows.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *finite-sum row obtained from the displayed cells, tags, and values.*
6. *I is the* $\mathsf{Integral}^{\uparrow}$ *consumer boundary that may read S only after P, V, A, D have been displayed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation ledger used when a later real-facing route samples the finite row by regular rational names.*
8. *H records componentwise* hsame *transport for partition, value, cell-admission, dyadic comparison, finite-sum, integral-boundary, regular-readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a step-function read through P, V, A, D, S, I, R, H.*
10. *Q is the* Pkg *provenance over* $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StepFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these rows. It has no coordinate for a measurable-function quotient, an ambient function space, a selected refinement limit, open-cover compactness, or an unbounded integration procedure.

Theorem 2808.2 (Step function NameCert obligations)**.** *For every accepted* $\mathsf{StepFunction}^{\uparrow}$ *carrier* $F = (P, V, A, D, S, I, R, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StepFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure, real-value exposure, constant-on-cell admission, dyadic comparison when present, finite-sum handoff, integral-boundary handoff, regular rational readback, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.StepFunctionUp.StepFunctionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3780.1. The rows P, V, A display the finite partition, the value row on each cell, and the constant-on-cell admission relation before any consumer can read a function value.

The analytic consumers are downstream of those rows. The dyadic comparison row D is only a displayed specialization, the finite-sum row S is computed from the visible cells and values, and the integral boundary I reads the same finite row before any integration claim is exported.

The regular readback and structural rows preserve that route. Since R, H, C, Q, N name only regular sampling, transport, replay, provenance, and the local certificate, and since the carrier has no coordinate for a host function quotient, selected refinement limit, compactness proof, or unbounded integration procedure, the listed rows exhaust the NameCert surface. □

Theorem 2808.3 (Step function partition-integral route)**.** *Every integration-facing read of an accepted* $\mathsf{StepFunction}^{\uparrow}$ *carrier factors through*

$$\mathsf{Partition}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

together with the $\mathsf{RegSeqRat}^{\uparrow}$ *approximation ledger and the structural rows* H, C, Q, N*. The route exports a finite partition-to-real approximation surface, not a quotient function object, a selected limiting mesh, measure-theoretic integration, open-cover compactness, or ambient completeness.*

Proof. Start with the displayed partition row. A read has no domain observation except the cells listed by P, and it has no value observation except the corresponding $\mathsf{Real}^{\uparrow}$ rows in V.

The step-function comparison and finite-sum handoff are then forced. The dyadic row D records only the dyadic specialization of the same cells and values, while S is the finite $\mathsf{RiemannSum}^{\uparrow}$ row computed from those displayed data.

Finally read the integral and approximation rows. The $\mathsf{Integral}^{\uparrow}$ boundary I is available only after P, V, A, D, S are displayed, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies regular rational sampling without changing the carrier. The structural rows H, C, Q, N transport and name the same route, so no external function quotient, mesh-limit choice, compactness principle, measure-theoretic integral, or ambient completeness coordinate can be exported. □

Falsifiable prediction: An accepted $\mathsf{StepFunction}^{\uparrow}$ integration-facing read is refuted if its $\mathsf{Integral}^{\uparrow}$ boundary is preserved while the displayed $\mathsf{Partition}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ value row, $\mathsf{RiemannSum}^{\uparrow}$ finite-sum row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback are removed.

Independence witness: partition, real-value, dyadic-step-function, riemann-sum, integral, regseqrat: none is bijective with $\mathsf{StepFunction}^{\uparrow}$ because the packet jointly carries finite partition cells, displayed real values, constant-on-cell admission, dyadic comparison, finite-sum handoff, integral

boundary, regular rational readback, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}$.

Constructive story: A $\mathsf{StepFunction}^{\uparrow}$ packet is generated from a finite PartitionUp row, displayed RealUp values, constant-on-cell admission, optional DyadicStepFunctionUp comparison, finite RiemannSumUp handoff, IntegralUp boundary, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3780.1, Definition 3780.2, and Definition 3780.3 seed the finite partition-to-real step-function approximation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.StepFunctionUp.StepFunctionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim measure-theoretic integration, host function quotients, selected refinement limits, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, real-value exposure, constant-on-cell admission, finite-sum handoff, IntegralUp boundary routing, RegSeqRatUp readback, and structural transport for BEDC.Derived.StepFunctionUp.

Chapter 2809

A Concrete Naming Certificate for $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$

Following Khinchin's continued fractions and Hardy–Wright's Stern–Brocot/Farey treatment, the $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ packet records the finite bridge by which Stern–Brocot interval refinement and continued-fraction convergent readback expose the same rational-approximation route before any $\mathsf{Real}^{\uparrow}$ completion-facing consumer reads it. It sits over the finite tree surface of chapter 1704, the bracketing approximation route of chapter 1431, the continued-fraction carrier of chapter 342, the convergent row surface of chapter 1777, the Farey-neighbor sequence route of chapter 1759, the rational source of chapter 37, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite NameCert handoff for the L10 rational-approximation lattice; it does not use quotient rational equality, selected representatives, host enumeration, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2809.1 (Stern–Brocot continued-fraction bridge carrier)**.** *A* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, A, F, Q, V, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *tree row, including the finite address, boundary, mediant, and Farey-neighbor ledger read by the bridge.*
2. *A is the* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *bracket row, exposing the same inspected rational interval as a finite approximation surface.*
3. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *handoff row, recording the adjacent determinant-one boundary sequence used to compare tree intervals with continued-fraction cylinders.*
4. *Q is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *partial-quotient spine row, restricted to the finite prefix inspected by the bridge.*
5. *V is the* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *convergent row, including the numerator, denominator, normalized rational, dyadic error, and Real-facing handoff rows of the inspected prefix.*
6. *I is the rational-interval comparison ledger, read through* $\mathsf{Rat}^{\uparrow}$ *rows and the displayed determinant-one adjacency data rather than through host rational equality.*
7. *W is the finite window row scheduling the Stern–Brocot branch, Farey sequence, partial-quotient prefix, and convergent observations.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row built from the shared rational interval and convergent data.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, A, F, Q, V, I, W, R have been displayed.*
10. *H records componentwise* hsame *transport for the tree, approximation, Farey, continued-fraction, convergent, rational-comparison, window, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a bridge request through Stern–Brocot boundary inspection, Farey adjacency, continued-fraction prefix inspection, convergent readback, regular rational handoff, and the terminal Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{ContinuedFractionConvergents}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$ *row.*

The classifier compares accepted bridge packets only through $S, A, F, Q, V, I, W, R, E, H, C, P, N$. The carrier has no coordinate for quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.SternBrocotContinuedFractionBridgeUp`
Lean4 ✓ `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.SternBrocotContinuedFractionBridgeTasteGate`
`single carrier alignment`
Lean4 ✓ `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.TasteGate.SternBrocotContinuedFractionBridgeT`
`single carrier alignment`

Theorem 2809.2 (Stern–Brocot continued-fraction NameCert obligations)**.** *For every accepted* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *carrier*

$$B = (S, A, F, Q, V, I, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Stern–Brocot tree row, bracketing approximation row, Farey-sequence handoff, continued-fraction prefix row, convergent row, rational-interval ledger, finite window row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-prefix compatibility, requiring the Stern–Brocot branch in S and the continued-fraction prefix in Q to inspect the same displayed rational interval through F and I;*
3. *convergent handoff, requiring every rational convergent row in V to be read through the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *route before the terminal seal E is consumed;*
4. *approximation bridge discipline, requiring the bracketing row A and the Farey handoff F to mediate between Stern–Brocot intervals and continued-fraction convergents as finite* BHist *rows;*
5. *componentwise transport, displayed replay, package provenance, and local naming, requiring H, C, P, N to preserve the same bridge packet under* hsame *transport and* Cont *replay;*
6. *refusal of quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as bridge data.*

No local bridge obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3781.1. The accepted tuple exposes exactly the tree, approximation, Farey, continued-fraction, convergent, rational-comparison, window, readback, seal, transport, replay, provenance, and local naming rows listed in the statement.

Interval-prefix compatibility is read through S, F, Q, I. The Stern–Brocot side supplies a finite branch and boundary pair, the Farey side supplies determinant-one adjacency, the continued-fraction side supplies the inspected partial-quotient prefix, and the rational ledger I records that these rows refer to the same displayed rational interval.

The convergent handoff is downstream of V, W, R, E. A consumer first reads the scheduled finite convergent window, then the regular rational readback R, and only then reaches the Real seal E.

The approximation bridge is finite because A and F are displayed rows. They mediate from rational bracketing and Farey adjacency to the continued-fraction convergent route without importing a host enumeration of rationals or a selected representative.

Finally H, C, P, N transport, replay, package, and name the same packet. Since the carrier inventory contains no quotient rational equality, selected representative, host enumeration, choice, trichotomy, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2809.3 (Stern–Brocot continued-fraction Farey handoff)**.** *Let $B = (S, A, F, Q, V, I, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *packet. Every bridge-facing read from a Stern–Brocot interval to a continued-fraction convergent factors through the finite route*

$$\mathsf{SternBrocot}^{\uparrow} \longmapsto \mathsf{SternBrocotApproximation}^{\uparrow} \longmapsto \mathsf{FareySequence}^{\uparrow} \longmapsto \mathsf{ContinuedFraction}^{\uparrow} \longmapsto \mathsf{ContinuedFractionCo}$$

The Farey handoff F is the only adjacency surface in the route, and the rational comparison ledger I is read as displayed $\mathsf{Rat}^{\uparrow}$ *data. Thus the bridge exports a Real-facing regular-rational handoff while refusing quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Stern–Brocot side. The row S exposes only a finite tree address, boundary rows, a mediant row, and a Farey-neighbor ledger, while A records the corresponding bracketing approximation surface.

Next read the Farey handoff F. It supplies determinant-one adjacency for the displayed interval boundary, so the bridge compares the tree interval with the continued-fraction cylinder through finite Farey data rather than through a host enumeration of rational numbers.

The continued-fraction side then reads Q and V. The prefix row Q names the finite partial quotients under inspection, and the convergent row V names the numerator, denominator, normalized rational, dyadic error, and regular handoff data for that prefix.

Finally read W, R, E, H, C, P, N. The finite window row schedules the common observation, R carries the $\mathsf{RegSeqRat}^{\uparrow}$ readback, E is terminal for the Real-facing route, and the structural rows preserve and name the same packet. The carrier has no coordinate for quotient rational equality, selected representatives, host enumeration, choice, trichotomy, or ambient completeness, so none is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ packet is generated from a Stern–Brocot tree row, a bracketing approximation row, a Farey adjacency handoff, a continued-fraction prefix row, a convergent row, a displayed rational-interval ledger, finite observation windows, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3781.1, Definition 3781.2, and Definition 3781.3 seed the finite bridge from Stern–Brocot interval refinement to continued-fraction convergent readback and Real-facing regular rational handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{SternBrocotContinuedFractionBridge}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-prefix compatibility, convergent handoff, Farey mediation, transport, replay, provenance, and local naming for the same bridge packet.

Chapter 2810

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ packet records the finite route by which a regular Cauchy family of regular rational names is read along a unary diagonal selector before any Real or Completion consumer treats the diagonal read as a completed real name. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the sibling diagonal extraction packet of chapter 1773. The packet gives the L10 Real and Completion route one displayed diagonalization carrier; it does not assert quotient stream equality, choose a host limit, invoke countable choice, use trichotomy, claim ambient $\mathsf{Real}^{\uparrow}$ completeness, provide Lean verification, or record a checked bridge.

Definition 2810.1 (Regular Cauchy diagonalization carrier)**.** *A* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, \delta, S, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the regular Cauchy family row. It displays the source family of rational names together with the finite precision requests inspected by the diagonal read.*
2. *δ is the unary diagonal selector row. It names the selected source index for each inspected precision and records no independent choice schedule beyond that displayed unary selector.*
3. *S is the aligned* $\mathsf{StreamName}^{\uparrow}$ *window row. It displays the finite windows of the selected source names and aligns them with the requests carried by F and δ.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested dyadic radius, split-tolerance budget, comparison windows, and common-scale data used by the diagonal read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row. It reads the selected stream windows through the tolerance ledger as one regular rational sequence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, δ, S, T, R have been displayed.*
7. *H records componentwise* hsame *transport for the family, selector, stream windows, tolerance ledger, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a diagonalization request through the family row, unary selector, aligned windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, transports, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, \delta, S, T, R, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected host limits, countable-choice schedules, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Theorem 2810.2 (Regular Cauchy diagonalization NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier*

$$D = (F, \delta, S, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular Cauchy family, unary diagonal selector, aligned StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat diagonal readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *selector exactness, requiring every diagonal read to pass through the displayed unary selector δ and the family row F;*
3. *window alignment, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row S to expose exactly the selected finite windows named by F and δ;*
4. *tolerance exactness, requiring the requested radius, split budget, comparison windows, and common-scale data to be read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row T;*
5. *regular readback exactness, requiring R to consume the same selected windows and tolerance ledger before any Real-facing seal is visible;*
6. *Real-seal nonescape, requiring E to appear only after F, δ, S, T, R are transported through H and replayed through C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly the rows $F, \delta, S, T, R, E, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable-choice schedules, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No local diagonalization obligation is stored outside these rows.

Proof. Project the carrier of Definition 3782.1. The accepted inventory is exactly the family row F, unary selector δ, aligned StreamName windows S, dyadic tolerance ledger T, regular rational readback R, Real seal E, and structural rows H, C, P, N.

Selector exactness is read from F and δ. A consumer may inspect only the named source family entries selected by the displayed unary diagonal row, so no hidden choice schedule or host enumeration is available.

Window alignment and tolerance exactness are read from S and T. The stream row exposes the finite windows selected by the diagonal read, while the dyadic row supplies the requested radius, split budget, comparison windows, and common-scale ledger used to compare those windows.

Regular readback and the Real seal are downstream of the same route. The row R consumes S and T as one $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback, and E is visible only after that readback has been transported and replayed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the carrier inventory contains no quotient-stream equality, selected host limit, countable-choice schedule, trichotomy, ambient completeness principle, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalizationUp.RegularCauchyDiagonalizationTasteGate single carrier alignment`

Theorem 2810.3 (Regular Cauchy diagonalization Real seal handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier factors through the finite route*

$$F, \delta \longmapsto S \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Equivalently, the consumer receives a regular Cauchy family row, one unary diagonal selector, aligned $\mathsf{StreamName}^{\uparrow}$ *windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, and one terminal* $\mathsf{Real}^{\uparrow}$ *seal before any Real-facing output is named.*

Proof. Start with the obligation surface of Definition 3782.2. It confines every accepted packet to the displayed family, selector, stream-window, tolerance-ledger, readback, Real-seal, and structural rows.

The consumer first reads F and δ. These rows name the source family and the unary diagonal selector, so the diagonal route has a displayed source for every inspected precision request.

The consumer then reads S, T, R. The aligned stream windows expose the selected entries, the dyadic ledger supplies the tolerance and comparison budget, and the regular rational readback repacks those rows as one $\mathsf{RegSeqRat}^{\uparrow}$ diagonal sequence.

Only after that route is visible may the consumer read the terminal $\mathsf{Real}^{\uparrow}$ seal E and the structural rows H, C, P, N. Because the carrier has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking, none of those objects can enter the Real-facing handoff. □

Theorem 2810.4 (Regular Cauchy diagonalization selector-window exactness)**.** *For an accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier*

$$D = (F, \delta, S, T, R, E, H, C, P, N),$$

the unary diagonal selector row δ determines the aligned $\mathsf{StreamName}^{\uparrow}$ *window row S before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R are consumed. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the diagonal seal must therefore use the same selected window with the tolerance ledger T before reading R or the terminal seal E.*

Proof. Apply Definition 3782.2. It states selector exactness through F, δ, window alignment through S, tolerance exactness through T, regular readback exactness through R, and Real-seal nonescape through E.

Then apply Definition 3782.3. It orders every Real or Completion consumer through

$$F, \delta \longmapsto S \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the selector fixes the window before the tolerance ledger and readback are visible, and the same selected window is preserved until the diagonal seal is read. □

Theorem 2810.5 (Regular Cauchy diagonalization common-tolerance lock)**.** *If two accepted* RegularCauchyDiagona *packets share the same regular Cauchy family row F, unary selector δ, aligned* $\mathsf{StreamName}^{\uparrow}$ *window row S, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal handoff in both packets is locked to that common tolerance ledger. More explicitly, for*

$$D = (F, \delta, S, T, R, E, H, C, P, N)$$

and

$$D' = (F, \delta, S, T, R', E', H', C', P', N'),$$

every diagonal readback route through R or R' and every Real seal handoff through E or E' must consume the displayed ledger T after the same F, δ, S route.

Proof. Use the obligation theorem, Definition 3782.2, to project both carriers. Their shared rows F, δ, S, T are the only family, selector, window, and tolerance rows available before any readback or Real seal is exposed.

Apply selector-window exactness, Definition 3782.4. It fixes the route through F, δ, S before the tolerance ledger and readback rows are consumed.

Now apply the Real seal handoff theorem, Definition 3782.3. It keeps R, E and R', E' downstream of the same tolerance ledger T, with the structural rows only transporting, replaying, sourcing, and naming the displayed evidence.

Therefore both packets lock every RegSeqRat readback and Real seal handoff to the common ledger T. Since neither carrier contains quotient stream equality, selected host limits, countable choice, trichotomy, or ambient Real completeness, none of those excluded objects can loosen the common tolerance budget. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ Real or Completion read factors through a regular Cauchy family row, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-diagonal-extraction: $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source regular Cauchy family, a unary diagonal selector, aligned finite windows, a dyadic tolerance ledger, a regular rational diagonal readback, and a terminal RealUp seal in one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ packet is generated from a regular Cauchy family row, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3782.1, Definition 3782.2, Definition 3782.3, Definition 3782.4, and Definition 3782.5 seed the finite diagonalization packet used by L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector exactness, window alignment, dyadic tolerance exactness, regular readback exactness, Real-seal exposure, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyDiagonalizationUp.

Chapter 2811

A Concrete Naming Certificate for $\mathsf{TotallyBoundedClosure}^{\uparrow}$

Following Bishop and Bridges on total boundedness and Kelley and Bourbaki on metric closure, the $\mathsf{TotallyBoundedClosure}^{\uparrow}$ packet records the finite BEDC route by which a displayed closure window of a totally bounded metric carrier remains available to finite-net consumers. It sits after the finite-net source of chapter 127, the metric-distance surface of chapter 57, the compact metric consumer of chapter 129, the closed-subspace completion route of chapter 1949, and the precompact bridge of chapter 1531. The packet names only displayed closure-window data and finite rational readbacks; it does not import host closure, selected limits, quotient completion, open-cover compactness, or ambient real completeness.

Definition 2811.1 (TotallyBoundedClosure carrier)**.** *A* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, M, F, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net source row, including the closed* $\mathsf{ProbeBundle}$ *center families,* $\mathsf{InBundle}$ *membership rows, center-carrier rows, distance witnesses, and rational bound ledgers of Definition 127.3;*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row for the same source, exposing endpoint classifiers, distance-witness reads, triangle ledgers, and the exact comparison rows used by closure requests;*
3. *F is the finite closure-window ledger, naming the displayed approximating source points, their tolerance windows, and the inclusion route from each window point into the closure carrier;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records only the finite observation prefixes needed by a closure consumer before a net center is inspected;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tails attached to the closure windows and the finite-net tolerances;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint distance bounds and closure radii already exposed by M, F, W, R;*
7. *H records componentwise* hsame *transport for all rows T, M, F, W, R, E and for every center, window point, rational tail, real seal, and metric-distance history read from them;*

8. *C records the displayed* Cont *replay route from closure-window reads to the source finite-net row, then back through the metric distance ledger to the closure carrier;*
9. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedClosure}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$, $\mathsf{PrecompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$ *row.*

The classifier on such packets is componentwise transport through H together with replay of the finite routes in C. The carrier has no coordinate for a host closure operator, a choice-selected convergent sequence, quotient completion equality, an open-cover compactness principle, or an ambient completeness theorem.

Theorem 2811.2 (TotallyBoundedClosure NameCert obligations)**.** *For an accepted* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *carrier $K = (T, M, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$ *surface consists exactly of six obligations:*

1. *carrier admission: every row exposed to a consumer is one of the ten displayed rows or a finite subrow of the totally bounded source, metric distance ledger, closure-window ledger, stream window, rational readback, or real seal;*
2. *classifier transport: packet equality is componentwise* hsame *transport of centers, source points, closure-window points, stream prefixes, rational tails, real seals, metric distances, provenance rows, and local naming rows;*
3. *finite-net inheritance: each positive tolerance read reaches the closed* ProbeBundle*, membership constructors, center histories, distance witnesses, and rational comparison ledgers already named by T;*
4. *closure-window exactness: each closure point read first passes through a finite window in F, W, R, E before the metric row M compares it with a displayed source point;*
5. *replay discipline: every handoff from a closure-window point to a finite net factors through the displayed* Cont *route in C and the same source metric rows in M;*
6. *provenance and non-escape: P and N name only the listed source certificates, and the packet exports no host closure, selected limit, quotient completion, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

These obligations make total boundedness under displayed closure a finite-row interface rather than a semantic closure theorem in an ambient metric space.

Proof. Carrier admission is projection from Definition 3783.1. The ten rows list the finite-net source, metric distance row, closure-window ledger, stream windows, regular rational readbacks, real seals, transport ledger, replay ledger, provenance row, and local naming row.

Classifier transport is the role of H. It moves the source centers, closure-window points, rational tails, real seals, and metric-distance histories by hsame, while the point and distance classifiers in M keep the source and closure endpoint reads aligned.

Finite-net inheritance is read from T. A positive tolerance request reaches exactly the $\mathsf{TotallyBounded}^{\uparrow}$ ProbeBundle spine, its InBundle membership constructors, center-carrier rows, coverage-distance witnesses, and rational comparison ledgers.

Closure-window exactness is supplied by F, W, R, E. The packet never asks for a host closure membership proof; it displays the finite windows, the stream prefixes, the rational tail reads, and the real endpoint seals that justify each closure read before any finite net is consumed.

Finally, C, P, N package the handoff. The replay route sends a closure-window point to a displayed source approximation, reads the totally bounded center through T, compares the closure point through M, R, E, and returns only the named finite rows to compact, completion, or precompact consumers. Since no other coordinate is present, host closure, selected limits, quotient completion, open-cover compactness, countable choice, and ambient real completeness cannot be exported. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedClosureUp.TotallyBoundedClosureTasteGate single carrier alignment`

Theorem 2811.3 (TotallyBoundedClosure finite-net closure handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$, *or* $\mathsf{PrecompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *packet* $K = (T, M, F, W, R, E, H, C, P, N)$ *reads total boundedness of the closure only through the finite route*

$$\mathsf{TotallyBounded}^{\uparrow}, \mathsf{Met}^{\uparrow}, F, W, R, E, H, C, P, N.$$

For each positive tolerance history, the consumer receives a displayed source net from T*, a closure-window approximation from* F, W, R, E*, and the metric comparison row from* M *that transfers the source-net bound to the closure point. The route exports no host closure operator, no chosen dense sequence, no quotient completion, no selected subcover, and no ambient completeness principle.*

Proof. Fix a positive tolerance request made by one of the listed consumers. The request is first routed by C to the finite-net source row T, so the available centers and coverage witnesses are exactly the closed ProbeBundle rows of chapter 127.

For a closure point, the packet then reads a finite window in F together with the prefix row in W, the regular rational tail row in R, and the real seal in E. These rows display a source approximation close enough for the requested tolerance; no closure predicate outside the packet is inspected.

The metric row M compares the closure point, the displayed source approximation, and the selected net center through the same endpoint classifiers and distance ledgers. Transport by H and replay by C make the comparison stable under the packet classifier.

Thus any compact-metric, closed-subspace-completion, or precompact consumer sees only the finite-net source, closure windows, rational readbacks, real seals, metric comparisons, transport, replay, provenance, and local NameCert rows. The excluded host closure, selected dense sequence, quotient completion, selected subcover, and ambient completeness data have no carrier coordinate from which they could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3783.1, Definition 3783.2, and Definition 3783.3 seed the finite L10 closure interface over totally bounded source nets, metric distance rows, finite closure windows, stream windows, rational readbacks, real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host closure operators, selected dense sequences, quotient completions, selected subcovers, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite-net inheritance, closure-window exactness, replay discipline, and provenance non-escape for the same packet.

Constructive story: A closure consumer reads a finite window around the requested closure point, routes that window back to the totally bounded source net, compares the displayed source approximation by the metric ledger, and exports only the resulting finite rows.

Chapter 2812

A Concrete Naming Certificate for $\mathsf{UniformModulusComposition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformModulusComposition}^{\uparrow}$ packet records the finite bookkeeping needed to compose displayed uniform moduli for map-level continuity before a continuous-map or completion-facing consumer reads the result. It sits beside the map carrier of chapter 130, the modulus surface of chapter 2228, the Cauchy budget composition of chapter 1617, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet names only the displayed composition route for uniform-modulus rows; it is not a quotient stream, not a selected limit, not host real equality, not trichotomy, not ambient Real completeness, and not a bridge assertion.

Definition 2812.1 (Uniform modulus composition carrier)**.** *A* $\mathsf{UniformModulusComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *F and G are composable* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, with G read after the codomain row exported by F.*
2. *M_F and M_G are the displayed* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *rows for F and G.*
3. *B is the composed uniform-budget row. It first reads the precision request through M_G, then uses the resulting domain tolerance as the request for M_F, and stores the single source tolerance supplied to the composite map.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window route through which the composed budget may inspect only displayed finite observations.*
5. *D is the dyadic or rational tolerance ledger naming every radius comparison used by the two modulus lookups.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that receives the composed source tolerance after W and D have been replayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *route row that may be read only after the regular-rational handoff has consumed the same composed budget.*

8. *H records componentwise* hsame *transport for $F, G, M_F, M_G, B, W, D, R, E$, the continuation row, the provenance row, and the local naming row.*
9. *C records displayed* Cont *replay from a requested output tolerance through M_G, M_F, the composed budget B, stream windows W, dyadic or rational tolerances D, the RegSeqRat handoff R, and the Real route E.*
10. *P is the* Pkg *provenance over* ContinuousMap$^\uparrow$, UniformContinuityModulus$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformModulusComposition}^\uparrow}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, ambient completeness of Real$^\uparrow$, *or standard bridge data.*

Lean4 ✓ BEDC.Derived.UniformModulusCompositionUp.TasteGate.UniformModulusCompositionTasteGate single carrier alignment

Theorem 2812.2 (Uniform modulus composition NameCert obligations)**.** *For every accepted* UniformModulusComposition$^\uparrow$ *carrier $K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{UniformModu}}$ *surface consists exactly of carrier admission for the two composable map rows, admission of their displayed uniform-modulus rows, budget composition through M_G then M_F, window and tolerance exposure through W and D, RegSeqRat and Real handoff through R and E, componentwise transport stability through H, continuation replay through C, package provenance through P, and local naming through N. No obligation row stores a quotient stream, selected limit, host equality of reals or maps, trichotomy, ambient completeness principle, or standard bridge claim.*

Proof. Project the carrier of Definition 3784.1. The tuple displays exactly the two continuous-map rows, their uniform-modulus rows, the composed budget, finite-window route, tolerance ledger, RegSeqRat handoff, Real route, transport, continuation, provenance, and naming row.

The composition order is fixed by B. An output tolerance for the composite is first read through M_G because G controls the second map. The tolerance exposed by that read is then passed to M_F, and the resulting source tolerance is stored as one composed budget row.

The consumer route is ordered by C. Before any regular-rational or Real-facing read can consume the composed budget, it must replay the finite windows W and the tolerance ledger D.

Transport stability is componentwise. The row H replaces only displayed rows by hsame-classified rows and replays the same continuation order, so it cannot introduce a new equality principle.

The carrier contains no quotient-stream, selected-limit, host-equality, trichotomy, completeness, or bridge coordinate. The listed rows therefore exhaust the local NameCert obligation surface. □

Theorem 2812.3 (Uniform modulus composition Real route)**.** *Let $K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$ be an accepted* UniformModulusComposition$^\uparrow$ *carrier. Every Real-facing read of the composite map reaches the* Real$^\uparrow$ *route only after the composed uniform budget has passed through the two map rows, the two displayed uniform-modulus rows, the StreamName windows, the tolerance ledger, and the RegSeqRat handoff. The exported surface contains no quotient stream, selected limit, host real equality, trichotomy, ambient Real completeness assertion, or standard bridge claim.*

Proof. The Real route row E is displayed in the carrier, but its accessibility is ordered by C. A Real-facing read must therefore replay $F, G, M_G, M_F, B, W, D, R$ before E is available.

The replay keeps the composed modulus finite. The two source map rows supply the composability context, M_G and M_F supply the only tolerance lookups, W supplies the only stream-window route, D supplies the only tolerance ledger, and R supplies the only regular rational handoff.

The support rows H, P, N name and transport that same finite packet. They cannot introduce quotient streams, selected limits, host equality, trichotomy, ambient completeness, or bridge data, because none of those data appear in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformModulusComposition}^{\uparrow}$ packet is generated from two composable ContinuousMap rows, two UniformContinuityModulus rows, a composed uniform-budget row, a StreamName window route, a tolerance ledger, a RegSeqRat handoff, a RealUp route, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3784.1, Definition 3784.2, and Definition 3784.3 seed the finite map-level route that composes displayed uniform moduli before RegSeqRat and Real consumers read them.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget composition, window and tolerance exposure, RegSeqRat handoff, Real route nonescape, and transport stability for BEDC.Derived.UniformModulusCompositionUp.

Chapter 2813

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet names a closed constructive interval by regular Cauchy data rather than by quotient real equality. It sits between the closed bounded interval rows of chapter 1591 and the completion-facing consumers of chapter 301, chapter 126, chapter 291, and chapter 38.

Definition 2813.1 (Constructive Cauchy interval carrier)**.** *A* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, O, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are lower and upper located endpoint rows inherited from a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source.*
2. *O is the endpoint-order row asserting that the lower endpoint is not above the upper endpoint in the located order used by the source interval.*
3. *D is the dyadic containment row over* $\mathsf{DyadicRatCore}^{\uparrow}$*; it names the finite dyadic windows that contain the interval at each displayed precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite stream of containment windows read by Cauchy consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row; it turns each accepted containment window into regular rational sequence data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row recording the real-completion consumer reached from R without quotient real equality, trichotomy, choice, or an ambient completeness axiom.*
7. *H records componentwise* hsame *transport for endpoint, order, dyadic, stream, regular-sequence, real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from any interval-completion read through L, U, O, D, S, R, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$*,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, O, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected interval points, quotient real equality, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ConstructiveCauchyIntervalUp.ConstructiveCauchyIntervalTasteGate single carrier alignment`

Theorem 2813.2 (Constructive Cauchy interval NameCert obligations)**.** *For every accepted* ConstructiveCauchyInt *carrier $I = (L, U, O, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$ *surface consists exactly of endpoint admission, endpoint-order preservation, dyadic containment exposure, StreamName window exposure, RegSeqRat handoff, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3785.1. The rows L, U, O are inherited from the closed bounded interval source and therefore supply the only endpoint data visible to the packet.

Next read the completion route. The dyadic row D exposes finite containment windows, S names their stream surface, R supplies the regular rational sequence handoff, and E seals the route as a Real-facing completion consumer.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the excluded host objects are not carrier coordinates, the local NameCert obligations are exactly the listed rows. □

Theorem 2813.3 (Constructive Cauchy interval RegSeqRat handoff)**.** *Every completion-facing read of an accepted* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ *packet factors through*

$$L, U, O \longmapsto D \longmapsto S \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, any RegSeqRat or Real consumer sees only the located endpoints, endpoint order, dyadic containment ledger, StreamName windows, RegSeqRat handoff row, Real seal, transport, replay, provenance, and local NameCert row.

Proof. Start with the endpoint rows. The closed bounded interval source fixes L, U, O, so a completion-facing consumer cannot introduce a new endpoint classifier or a selected point of the interval.

The containment route is finite. Dyadic containment D supplies the displayed windows, the stream row S names their readback order, and the handoff row R exposes exactly the regular rational sequence surface consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

Finally the Real seal E and the structural rows H, C, P, N preserve the same route for downstream readers. Thus every RegSeqRat or Real-facing read factors through the displayed rows and does not claim open-cover compactness, quotient real equality, trichotomy, countable choice, or ambient completeness. □

Falsifiable prediction: Every completion-facing read of a $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet factors through located endpoints, endpoint order, dyadic containment, StreamName windows, RegSeqRat handoff, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: closedboundedinterval, regseqrat, streamname, dyadicratcore, real: ConstructiveCauchyI is not bijective to any listed sibling because it jointly carries interval endpoints, endpoint order, dyadic containment, stream windows, regular rational handoff, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet is generated from located endpoint rows, an endpoint-order row, dyadic containment data, StreamName windows, RegSeqRat handoff data, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3785.1, Definition 3785.2, and Definition 3785.3 seed the constructive Cauchy interval completion packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConstructiveCauchyIntervalUp.ConstructiveCauchyIntervalTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient real equality, trichotomy, selected interval points, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order preservation, dyadic containment exposure, StreamName window exposure, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.ConstructiveCauchyIntervalUp.

Chapter 2814

A Concrete Naming Certificate for CauchyCompletionDenseReflector$^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley on dense embeddings, the CauchyCompletionDenseRef packet records the dense-reflector face of Cauchy completion as finite NameCert data. It reads the completion transformer of chapter 378, the dense image surface of chapter 3351, and the universal mapping row of chapter 1456. Its local name route is the finite sequence of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the classical dense-reflection argument without importing quotient completions, selected limits, countable choice, host equality, or ambient completeness.

Definition 2814.1 (CauchyCompletionDenseReflector carrier)**.** *A* CauchyCompletionDenseReflector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, I, A, W, T, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source completion schedule, read through the* CauchyCompletionMonad$^{\uparrow}$ *unit, bind, and public* Real$^{\uparrow}$ *seal route rather than through a quotient completion.*
2. *I is the dense-image row inherited from* MetricCompletionDenseEmbedding$^{\uparrow}$*, exposing the source metric image inside the completion seal as finite approximation data.*
3. *A is the adjunction row inherited from* CauchyCompletionAdjunction$^{\uparrow}$*, comparing completion-facing consumers with source-side finite-window consumers over the same dense image.*
4. *W is the finite* StreamName$^{\uparrow}$ *window row through which the source and completion observations are inspected.*
5. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used by the source schedule, dense-image row, adjunction row, and real-seal readback.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed stream windows and dyadic tolerance ledger.*
7. *R is the terminal* Real$^{\uparrow}$ *seal row reached only after S, I, A, W, T, Q have been displayed.*
8. *H records componentwise* hsame *transport for the schedule, dense-image, adjunction, window, tolerance, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the source completion schedule through the dense image, adjunction row, finite windows, dyadic tolerance, regular readback, and real seal.*

10. *P is the* Pkg *provenance over* CauchyCompletionDenseReflector$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, MetricCompletionDe CauchyCompletionAdjunction$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCompletionDenseReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion representatives, selected limits, countable-choice schedules, host equality of completed objects or real names, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseReflectorUp.CauchyCompletionDenseReflectorTasteGate`
`single carrier alignment`

Theorem 2814.2 (CauchyCompletionDenseReflector NameCert obligations)**.** *For every accepted* CauchyCompletionDenseReflector$^{\uparrow}$ *carrier $D = (S, I, A, W, T, Q, R, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionDen}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source completion schedule, dense-image row, adjunction row, finite windows, dyadic tolerance ledger, regular readback, real seal, transport, replay, provenance, and local naming;*

2. *dense-source exposure, requiring every completion-facing read to expose S and then I before the real-seal row R can be consumed;*

3. *reflector exactness, requiring the dense image I and source schedule S to be replayed through the same* BHist, hsame, *and* Cont *rows before any completion consumer is served;*

4. *universal-row handoff, requiring every extension or uniqueness consumer to pass through the adjunction row A after the dense image and before the terminal real seal;*

5. *classifier transport, requiring all accepted comparisons to use only the displayed componentwise transport H, replay C, provenance P, and local name N;*

6. *ledger refusal, excluding quotient completion representatives, selected limits, countable choice, host equality of completed objects or real names, and ambient completeness.*

No dense-reflector obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3786.1. The accepted packet exposes exactly the rows $S, I, A, W, T, Q, R, H, C, P, N$ and no detached completion object.

Dense-source exposure is read in order. The source completion schedule S supplies the finite completion transformer route, and the dense image row I places the displayed source inside the completion seal before any real-facing read reaches R.

Reflector exactness and universal-row handoff then use the same route. The rows H and C replay the schedule and dense-image coordinates componentwise, while A is the only adjunction row available to extension and uniqueness consumers.

Finally inspect the provenance and local naming rows. The row P names only the completion monad, dense embedding, adjunction, StreamName, RegSeqRat, DyadicRatCore, Real, and structural BEDC rows, and N packages exactly that surface. Hence the listed refusals cannot be exported by an accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ packet is generated from a CauchyCompletionMonad source schedule, a MetricCompletionDenseEmbedding dense-image row, a CauchyCompletionAdjunction universal row, finite StreamName windows, DyadicRatCore tolerance accounting, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3786.1 and Definition 3786.2 seed the finite dense-reflector surface connecting completion monad, dense embedding, adjunction, and the terminal StreamName-RegSeqRat-DyadicRatCore-Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of completed objects or real names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exposure, reflector exactness, universal-row handoff, classifier transport, and ledger refusal for BEDC.Derived.CauchyCompletionDenseReflectorUp.

Chapter 2815

A Concrete Naming Certificate for $\mathsf{LocallyCompactMetric}^{\uparrow}$

Following Bourbaki and Kelley metric topology together with Bishop-style constructive metric practice, the $\mathsf{LocallyCompactMetric}^{\uparrow}$ packet records local compact neighbourhood data as finite BEDC rows around a displayed metric point. It sits over the $\mathsf{Met}^{\uparrow}$ distance surface of chapter 57, the $\mathsf{CompactMetric}^{\uparrow}$ compact-neighbourhood witness of chapter 129, the $\mathsf{CompleteMetric}^{\uparrow}$ limit rows of chapter 128, the $\mathsf{TotallyBounded}^{\uparrow}$ finite-net rows of chapter 127, and the positive $\mathsf{Real}^{\uparrow}$ radius seal of chapter 38. The chapter names a local compactness certificate for later proper-metric and compactification consumers; it does not assert arbitrary open-cover compactness, choice of neighbourhood bases, quotient completion equality, or ambient local compactness outside the displayed packet.

Definition 2815.1 (Locally compact metric carrier)**.** *A* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, x, U, K, T, C_m, r, H, D, P, N)$$

with the following displayed rows:

1. *X is a* $\mathsf{Met}^{\uparrow}$ *carrier row whose point histories and distance witnesses are read from chapter 57.*
2. *x is the inspected point history carried by X.*
3. *U is a local neighbourhood row around x, displayed only by metric-ball membership and finite route data.*
4. *K is a* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the neighbourhood closure packet.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net component read by K.*
6. *C_m is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness component read by K.*
7. *r is a positive* $\mathsf{Real}^{\uparrow}$ *radius seal for the local ball.*
8. *H records componentwise* hsame *transport for X, x, U, K, T, C_m, r.*
9. *D records displayed* Cont *replay from a local query through the metric carrier, neighbourhood row, compact witness, finite-net row, completion row, radius seal, and transport row.*
10. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$ *row.*
The carrier has no coordinate for arbitrary open covers, a selected neighbourhood basis, global compactness of X, quotient completion equality, countable choice, or a host topological space.

Lean4 (definition) `BEDC.Derived.LocallyCompactMetricUp.LocallyCompactMetricUp`
Lean4 ✓ `BEDC.Derived.LocallyCompactMetricUp.LocallyCompactMetricTasteGate single carrier alignment`

Definition 2815.2 (Locally compact metric classifier)**.** *Two accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carriers*

$$L = (X, x, U, K, T, C_m, r, H, D, P, N) \quad \textit{and} \quad L' = (X', x', U', K', T', C'_m, r', H', D', P', N')$$

classify together exactly when the metric carrier rows, inspected point rows, local neighbourhood rows, compact-metric witnesses, totally-bounded rows, complete-metric rows, Real radius seals, transport rows, continuation replays, provenance rows, and local naming rows classify componentwise by the sibling classifiers and the displayed hsame *transports.*

Theorem 2815.3 (Locally compact metric NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier $L = (X, x, U, K, T, C_m, r, H, D, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$ *surface consists exactly of the metric carrier row X, the inspected point row x, the neighbourhood row U, the compact-metric witness K, the totally-bounded finite-net row T, the complete-metric limit row C_m, the positive Real radius seal r, componentwise transport H, displayed replay D, provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3787.1. The projection exposes the metric row, inspected point, local neighbourhood, compact witness, finite-net witness, completion witness, radius seal, transport, replay, provenance, and naming row.

The compact witness K is read only through the sibling $\mathsf{CompactMetric}^{\uparrow}$ surface, whose own carrier consumes $\mathsf{TotallyBounded}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ rows. Thus the local compactness packet records compact neighbourhood evidence without adding an open-cover coordinate.

The radius row r is read through the Real seal surface and the route D replays all displayed dependencies. Since the carrier has no coordinate for selected bases, global compactness, quotient completion equality, countable choice, or an ambient host topology, the listed rows exhaust the NameCert obligations. □

Theorem 2815.4 (Locally compact metric neighbourhood non-escape)**.** *Every consumer of an accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *packet that preserves the classifier of Definition 3787.2 must preserve the same metric carrier, inspected point, neighbourhood row, compact-metric witness, total-boundedness row, completeness row, Real radius seal, transport, replay, provenance, and local naming row. In particular, it cannot replace the displayed local neighbourhood by an unrecorded open cover, a choice-selected compact set, or a host topology.*

Proof. Use the classifier of Definition 3787.2. It compares the accepted packets componentwise and has no residual coordinate where a different neighbourhood, compact witness, finite-net row, completion row, or radius seal could be hidden.

The continuation row preserves the same route through the metric, compact, total-boundedness, completeness, and Real-radius siblings. Therefore a consumer that keeps the classifier fixed must read the same finite packet and cannot escape to an unrecorded open cover, selected compact subset, or host topology. □

Theorem 2815.5 (Locally compact metric sibling dependency route)**.** *Every accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier routes its local compactness certificate through chapter 57, chapter 129, chapter 128, chapter 127, and chapter 38. The* $\mathsf{Met}^{\uparrow}$ *row supplies the point and distance surface; the* $\mathsf{CompactMetric}^{\uparrow}$ *row supplies the compact-neighbourhood witness; the* $\mathsf{TotallyBounded}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows supply the finite-net and limit components read by compact metric data; and the* $\mathsf{Real}^{\uparrow}$ *row supplies the positive radius seal.*

Proof. Read the rows of L in the order displayed by Definition 3787.1. The metric carrier X and point x must be available before a neighbourhood row U can be read as a metric-local object.

The compact witness K then exposes its two constructive compact-metric dependencies: total boundedness through T and complete-metric convergence through C_m. The radius seal r is a Real-facing row, so it is consumed through the terminal Real chapter rather than through a host real number.

Finally H, D, P, N transport, replay, source, and name the same dependency chain. Thus the local compactness packet is a finite L10 neighbour of the metric, compact metric, complete metric, totally bounded, and Real chapters, not a stand-alone topological assertion. □

Falsifiable prediction: Every $\mathsf{LocallyCompactMetric}^{\uparrow}$ read factors through one Metric carrier, one inspected point, one neighbourhood row, one CompactMetric witness, one TotallyBounded finite-net row, one CompleteMetric limit row, one Real radius seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: metric, compactmetric, completemetric, totallybounded, real: $\mathsf{LocallyCompactMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a point-local neighbourhood row, a compact-metric witness for that neighbourhood, the compact witness's total-boundedness and completeness components, and a positive Real radius seal in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactMetric}^{\uparrow}$ packet is generated from a Metric carrier, inspected point, local neighbourhood row, CompactMetric witness, TotallyBounded finite-net row, CompleteMetric limit row, positive Real radius seal, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3787.1, Definition 3787.2, Definition 3787.3, Definition 3787.4, and Definition 3787.5 seed the finite local-compactness surface over metric, compact-metric, complete-metric, total-boundedness, and Real-radius rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not assert arbitrary open-cover compactness, a selected neighbourhood basis, global compactness of the carrier, quotient completion equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, NameCert obligations, neighbourhood non-escape, and sibling route coverage for BEDC.Derived.LocallyCompactMetricUp.

Chapter 2816

A Concrete Naming Certificate for $\mathsf{LocallyCompactPolish}^{\uparrow}$

Following Bourbaki, Kelley, and Alexandrov–Urysohn, the $\mathsf{LocallyCompactPolish}^{\uparrow}$ packet records the finite BEDC surface where local compact neighbourhood data is read together with a Polish complete-separable metric presentation. It sits at the joint frontier of the local compact metric packet of chapter 3787, the Polish readback of chapter 1490, the proper closed-ball route of **??**, the sigma-compact exhaustion of chapter 2369, and the complete-separable metric presentation of chapter 1735. The certificate is finite data: neighbourhood rows, complete-separable readback, properness rows, sigma-compact exhaustion windows, StreamName and RegSeqRat observations, transport, replay, provenance, and local naming.

Definition 2816.1 (Locally compact Polish carrier)**.** *A* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, K, C, S, E, W, T, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the underlying metric-object row shared by the locally compact, Polish, proper, sigma-compact, and complete-separable presentations.*
2. *M is the* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *neighbourhood row. It records, for every displayed point-window, a compact neighbourhood candidate, an interior witness, and the metric inspection data by which the candidate is consumed.*
3. *K is the local compactness compact-row ledger. It binds compact neighbourhood reads to the compact witnesses supplied through M and refuses any neighbourhood not named in the packet.*
4. *C is the* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *readback row. It records the complete metric presentation, the countable dense probe family, and the finite comparison windows through which completeness and separability are inspected.*
5. *S is the* $\mathsf{PolishSpace}^{\uparrow}$ *row obtained from the same complete-separable readback; it carries no independent host topology coordinate.*
6. *E is the* $\mathsf{ProperMetric}^{\uparrow}$ *closed-ball compactness row. It is read only after M, K, C, S have fixed the metric and neighbourhood surface.*
7. *W is the* $\mathsf{SigmaCompact}^{\uparrow}$ *exhaustion row. Its compact exhaustion windows are generated from the displayed local compactness and properness rows, not from a selected external exhaustion.*

8. *T is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row for points, compact neighbourhood candidates, complete-separable probes, closed balls, exhaustion windows, and endpoint-facing reads.*
9. *R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row for the same metric probes and exhaustion windows.*
10. *H records componentwise* hsame *transport for $X, M, K, C, S, E, W, T, R, Q, P, N$.*
11. *Q records displayed* Cont *replay of the route*

$$M, K \longmapsto C, S \longmapsto E, W \longmapsto T, R.$$

12. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactPolish}^{\uparrow}$, $\mathsf{LocallyCompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{SigmaCompact}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, K, C, S, E, W, T, R, H, Q, P, N$. It has no coordinate for arbitrary open-cover compactness, a selected exhaustion, host topological equality, quotient metric equality, countable choice, or a Polish presentation detached from the displayed complete-separable row.

Theorem 2816.2 (Locally compact Polish NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier*

$$L = (X, M, K, C, S, E, W, T, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the thirteen displayed rows;*
2. *local compact neighbourhood exactness, requiring M and K to bind every compact neighbourhood read to a displayed point-window and compact witness;*
3. *complete-separable readback compatibility, requiring C to carry the complete metric presentation and countable dense probe family used by the packet;*
4. *Polish readback compatibility, requiring S to be consumed through C and not through an independent host topology;*
5. *proper closed-ball compactness compatibility, requiring E to inspect the metric and compact neighbourhood rows already displayed;*
6. *sigma-compact exhaustion compatibility, requiring W to use only the compact windows generated from the local compactness and properness route;*
7. *StreamName and RegSeqRat finite-window compatibility, requiring T and R to observe the same metric probes, compact neighbourhoods, closed balls, and exhaustion windows;*
8. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, Q, P, N.*

No locally compact Polish obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocallyCompactPolishUp.LocallyCompactPolishNameCertObligations`

Proof. Project the accepted packet $L = (X, M, K, C, S, E, W, T, R, H, Q, P, N)$. The metric object, local compactness, compact-neighbourhood ledger, complete-separable readback, Polish row,

properness row, sigma-compact exhaustion, finite observations, and structural rows are precisely the displayed carrier coordinates.

The neighbourhood route is fixed first. Rows M and K name the compact neighbourhood candidates and bind every accepted local compactness read to the corresponding point-window and compact witness.

Next the complete-separable route is read through C, and the Polish row S is downstream of that readback. Since S has no separate host-topology coordinate, every Polish consumer receives the same complete metric and dense-probe data.

The rows E and W expose proper closed-ball compactness and sigma-compact exhaustion only after the metric, neighbourhood, and complete-separable rows have been displayed. The observation rows T and R then inspect the same probes and windows through finite StreamName and RegSeqRat data.

Finally H, Q, P, N transport, replay, source, and name the route without adding new coordinates. The excluded objects do not occur in the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 2816.3 (Locally compact Polish proper-sigma route)**.** *Every properness, sigma-compactness, completion, or compact-neighbourhood consumer of an accepted* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{LocallyCompactMetric}^{\uparrow}, \mathsf{CompleteSeparableMetric}^{\uparrow}, \mathsf{PolishSpace}^{\uparrow} \longmapsto M, K, C, S \longmapsto \mathsf{ProperMetric}^{\uparrow}, \mathsf{SigmaCompact}^{\uparrow}$$

The route exposes local compact neighbourhood rows and Polish complete-separable readback before proper closed-ball compactness or sigma-compact exhaustion is public. It exports no arbitrary open-cover compactness, selected exhaustion, host topological equality, quotient metric equality, countable-choice selector, or Polish presentation detached from the displayed complete-separable packet.

Proof. Start with the rows M, K, C, S. The first pair carries the local compact neighbourhood data, while the second pair carries the complete-separable presentation and the Polish readback consumed by the packet.

The properness and sigma-compactness rows are then downstream. Row E reads closed-ball compactness only through the displayed metric and local compactness surface, and row W records exhaustion windows generated from the same source.

The observation and structural rows T, R, H, Q, P, N preserve that route for consumers. Because the carrier has no coordinate for open-cover compactness, selected exhaustion, host equality, quotient equality, countable choice, or detached Polish data, every consumer remains inside the displayed proper-sigma route. □

Falsifiable prediction: An accepted $\mathsf{LocallyCompactPolish}^{\uparrow}$ read is refuted if a properness, sigma-compactness, completion, or compact-neighbourhood consumer exports a closed-ball compactness row or exhaustion row without displaying LocallyCompactMetric neighbourhood data, Polish complete-separable readback, StreamName and RegSeqRat finite windows, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: locally-compact-metric, polishspace, propermetric, sigma-compact, complete-separable-metric, streamname, regseqrat: none alone is bijective with $\mathsf{LocallyCompactPolish}^{\uparrow}$ because the packet jointly carries local compact neighbourhood rows, complete-separable Polish readback, proper closed-ball compactness, sigma-compact exhaustion, finite observations, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactPolish}^{\uparrow}$ packet is generated from locally compact metric neighbourhood rows, compact-neighbourhood ledgers, complete-separable metric readback, PolishSpaceUp readback, ProperMetricUp closed-ball compactness, SigmaCompactUp exhaustion windows, StreamNameUp and RegSeqRatUp finite observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3788.1, Definition 3788.2, and Definition 3788.3 seed the locally compact Polish certificate surface from compact neighbourhood rows and complete-separable readback through proper closed-ball compactness and sigma-compact exhaustion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected exhaustions, host topological equality, quotient metric equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked local compact neighbourhood exactness, complete-separable and Polish readback compatibility, proper closed-ball compatibility, sigma-compact exhaustion compatibility, StreamName and RegSeqRat finite-window compatibility, transport, replay, provenance, and local naming for the displayed packet.

Chapter 2817

A Concrete Naming Certificate for $\mathsf{SequentialRealCompactness}^{\uparrow}$

Following Bolzano and Weierstrass, the $\mathsf{SequentialRealCompactness}^{\uparrow}$ packet records the finite route by which a bounded real sequence exposes a regular cluster subsequence through displayed interval-tree data. It sits after the subsequence extraction certificate of chapter 1254, the bounded source surface of chapter 1415, the interval-net carrier of chapter 1250, the regular rational readback of chapter 301, the finite window interface of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a sequential compactness interface for bounded real sequences, not an open-cover compactness theorem, a host completeness principle, quotient equality of real names, or a choice-selected arbitrary subsequence.

Definition 2817.1 (Sequential real compactness carrier)**.** *A* $\mathsf{SequentialRealCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, I, E, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence window schedule exposed through* $\mathsf{StreamName}^{\uparrow}$*.*
2. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row naming the bounded real sequence interface consumed by the extraction route.*
3. *I is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *interval-tree row carrying dyadic retained cells and finite mesh comparisons.*
4. *E is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *extraction row, including the monotone-cofinal subsequence schedule and retained finite windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the extracted windows and dyadic tolerances.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal reached only after S, B, I, E, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source schedule, bounded source row, interval tree, extraction row, regular readback, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from bounded sequence inspection through interval-tree retention, Bolzano–Weierstrass extraction, RegSeqRat readback, and RealUp sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentialRealCompactness}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$ *row.*

The carrier stores no open-cover compactness object, no arbitrary subsequence selector, no quotient equality of real names, no host supremum, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no branch outside the displayed interval-tree extraction route.*

Lean4 ✓ `BEDC.Derived.SequentialRealCompactnessUp.SequentialRealCompactnessTasteGate single carrier alignment`

Theorem 2817.2 (Sequential real compactness NameCert obligations)**.** *For every accepted* SequentialRealCompact *carrier $K = (S, B, I, E, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, bounded-source admission through B, finite interval-tree retention through I, Bolzano–Weierstrass extraction through E, finite stream-window and dyadic-tolerance readback through S and Q, terminal real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3789.1. The bounded sequence source is available only through S and B, while the interval-tree row I supplies the retained dyadic cells before extraction is read.

Use the bounded-source handoff of chapter 1415 and the interval-net surface of chapter 1250. These rows expose finite sequence windows, regular readback, real-name seals, and interval containment without adding an open-cover compactness object or a host completeness theorem.

Read the extraction row through chapter 1254. The Bolzano–Weierstrass packet supplies the monotone-cofinal subsequence schedule and finite retained windows, which are then read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

The structural rows H, C, P, N transport, replay, source, and name only this finite route. Since the carrier has no coordinate for arbitrary subsequence choice, quotient real equality, host supremum, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2817.3 (Sequential real compactness Bolzano–Weierstrass handoff)**.** *For every accepted* $\mathsf{SequentialRealCompactness}^{\uparrow}$ *carrier $K = (S, B, I, E, Q, R, H, C, P, N)$, the terminal compactness-facing read factors through the finite route*

$$\mathsf{BoundedRealSequence}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatC}$$

The handoff composes the bounded-source row with Bolzano–Weierstrass finite interval-tree extraction and the RegSeqRat/Real seal readback. It exports a displayed sequential cluster route, not open-cover compactness, choice-like subsequence selection, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3789.2. It places the bounded source row B, interval-tree row I, extraction row E, readback row Q, and terminal seal R in a single finite carrier route.

The bounded source is read through chapter 1415. Its finite windows and interval bound supply the source surface consumed by the Bolzano–Weierstrass certificate, while chapter 1250 supplies the dyadic retained-cell discipline.

Now compose with chapter 1254. The extraction row reads only the finite interval-tree schedule and then hands the selected windows to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ cluster seal can be consumed.

Transport and replay through H and C, then source and name the same route through P and N. The packet therefore records a finite sequential compactness handoff and refuses the excluded open-cover, choice, quotient, and ambient-completeness exits. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialRealCompactness}^{\uparrow}$ packet is generated from finite StreamName windows, a bounded real sequence source row, a closed bounded interval-net tree, a Bolzano–Weierstrass extraction row, a RegSeqRat readback row, a RealUp cluster seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3789.1, Definition 3789.2, and Definition 3789.3 seed the finite bounded-sequence compactness route through interval-tree extraction and regular real readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary subsequence choice, quotient real equality, host supremum, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source handoff, interval-tree extraction, RegSeqRat readback, Real seal routing, and NameCert field faithfulness for BEDC.Derived.SequentialRealCompactnessUp.

Chapter 2818

A Concrete Naming Certificate for $\mathsf{LawlikeSequence}^{\uparrow}$

Following Brouwer and Bishop-style constructive sequence analysis, the $\mathsf{LawlikeSequence}^{\uparrow}$ packet records a sequence source whose future values are governed by a displayed finite rule rather than by a host infinite function. It sits between the finite-window interface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the unfolding prefix surface of chapter 1482, and the finite observation boundary of chapter 1495. The packet supplies a lawful finite-rule stream source for the L10 Real and Completion route while keeping quotient streams, countable choice, host equality, and selected Real limits outside the exported surface.

Definition 2818.1 (Lawlike sequence carrier)**.** *A* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, W, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite rule row: a program, recurrence, primitive recursion table, or other finite calculation ledger whose admitted executions produce the inspected sequence values.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request-window row listing the finite prefixes and positions currently asked of the rule.*
3. *O is the observation row produced by executing R on the requests in W; it records only the displayed output values and their finite calculation traces.*
4. *H records componentwise* hsame *transport for R, W, O and for the structural rows that preserve the same displayed rule executions.*
5. *C records displayed* Cont *replay from a prefix request in W through the finite rule row R to the observation row O, and then to any downstream regular-rational or Real-facing read.*
6. *P is the* Pkg *provenance over* $\mathsf{LawlikeSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LawlessSequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through R, W, O, H, C, P, N. It has no coordinate for a host total function $\mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$*, quotient stream equality, a countable-choice selector, host sequence equality, a selected Real limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LawlikeSequenceUp.TasteGate.LawlikeSequenceTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.LawlikeSequenceUp.LawlikeSequenceUp`

Definition 2818.2 (Lawlike sequence classifier)**.** *The local* $\mathsf{LawlikeSequence}^{\uparrow}$ *classifier accepts two packets*

$$L = (R, W, O, H, C, P, N) \quad \textit{and} \quad L' = (R', W', O', H', C', P', N')$$

as the same lawlike sequence read precisely when the displayed rule rows are related by the local hsame *transport, the request windows list the same finite positions, the observation rows agree on every requested position by replaying the displayed rule executions, and the continuation, provenance, and local NameCert rows preserve that same finite route. Agreement is therefore a finite-rule replay judgment over the current window, not equality of completed streams or equality of host functions.*

Theorem 2818.3 (Lawlike sequence NameCert obligations)**.** *For every accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier* $L = (R, W, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite rule row, request-window exposure through* W*, observation replay through* O*, classifier stability under componentwise* hsame *transport, displayed* Cont *replay of the same finite rule executions, package provenance through* P*, and local naming through* N*. No obligation row may be discharged by a host infinite function, a countable-choice selector, quotient stream equality, host sequence equality, a selected Real limit, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LawlikeSequenceUp.LawlikeSequenceNameCertObligations`

Proof. Project the carrier of Definition 3790.1. The only sequence source displayed to a consumer is the finite rule row R, and the only inspected arguments are the finite requests listed in W.

For each request in W, replay the calculation trace stored in O. The replay is finite because O stores only the values produced by executing the displayed rule on the displayed window, not an ambient total function.

Transport stability is componentwise. The row H compares rule rows, request windows, and observation rows by hsame, while C replays the same dependency order from requests through rule execution to downstream reads.

Finally repack the route with P and N. Since none of the carrier coordinates stores a completed stream, a choice selector, quotient equality, host sequence equality, selected limit, or ambient completeness principle, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2818.4 (Lawlike sequence StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier receives the finite route*

$$R \longmapsto W \longmapsto O$$

before any regular-rational readback or terminal Real seal can be consumed. The route presents lawlike sequence data as a displayed finite-rule stream source: $\mathsf{StreamName}^{\uparrow}$ *reads the request window,* $\mathsf{RegSeqRat}^{\uparrow}$ *may read only the observations already produced on that window, and* $\mathsf{Real}^{\uparrow}$ *may be reached only after that regular-rational handoff has been replayed through* H, C, P, N*. The handoff exports no host infinite function, countable-choice selector, quotient stream equality, host equality, selected Real limit, or ambient Real completeness.*

Proof. Read the accepted carrier through Definition 3790.1. The first available sequence-facing row is the rule row R, and every downstream request must pass through the displayed finite window W.

The observation row O is then obtained by replaying the finite executions named by R on the requests in W. A $\mathsf{StreamName}^{\uparrow}$ consumer receives exactly that windowed read; it receives no tail values beyond the displayed requests.

If a regular-rational or Real-facing consumer is present, its read is downstream of the same observation row. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff can consume only the displayed observations, and the $\mathsf{Real}^{\uparrow}$ seal can be reached only after that handoff is replayed through H, C, P, N.

Thus the lawful source is the finite-rule packet itself. The absent coordinates are exactly the data needed for a host infinite function, countable-choice selector, quotient stream equality, host equality, selected Real limit, or ambient Real completeness claim, so none of those objects can be exported by the handoff. □

Theorem 2818.5 (Lawlike sequence regular-window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier* $L = (R, W, O, H, C, P, N)$ *must read the finite rule row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *request-window row* W*, the observation replay row* O*, and the componentwise transport and replay rows* H, C *before treating the displayed observations as a regular rational sequence. The consumer obtains no host infinite function, quotient stream equality, countable-choice selector, selected* $\mathsf{Real}^{\uparrow}$ *limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

If the read continues to a $\mathsf{Real}^{\uparrow}$ *endpoint, it does so only after the same regular-window route has passed through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the terminal* $\mathsf{Real}^{\uparrow}$ *seal recorded by* P, N*.*

Proof. Project the carrier through Definition 3790.1. The rule row R and the request-window row W determine exactly which finite calculations may be inspected.

Apply Definition 3790.4. It forces every stream-facing read to pass through

$$R \longmapsto W \longmapsto O$$

before any regular-rational consumer can see observations.

Now route the regularity readback through $\mathsf{RegSeqRat}^{\uparrow}$. Since the observations are the replayed outputs stored in O, regularity can be checked only against the displayed window and the componentwise H, C rows.

Finally, if a Real-facing endpoint is requested, the route is sealed through the terminal $\mathsf{Real}^{\uparrow}$ boundary named by P, N. The absent carrier coordinates are precisely the data needed for a host infinite function, quotient stream equality, countable choice, selected Real limit, or ambient completeness, so none of those objects is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LawlikeSequence}^{\uparrow}$ packet is generated from a finite rule row, StreamName request windows, observation rows produced by the rule on those windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3790.1, Definition 3790.2, Definition 3790.3, and Definition 3790.4 seed the finite-rule stream source feeding StreamName, RegSeqRat, and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, countable choice, host sequence equality, host function equality, selected Real limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-rule classifier stability, StreamName request-window exposure, RegSeqRat handoff exactness, Real seal nonescape, and refusal of host equality and choice for BEDC.Derived.LawlikeSequenceUp.

Chapter 2819

A Concrete Naming Certificate for MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$

The MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ packet realizes a finite conditional-boundary row from the MetaCIC confluence audit map of chapter 9378. It sits beside the residual-substitution boundary of chapter 2181, the critical-pair boundary of chapter 2212, the frontier critical-pair packet of chapter 2382, the local diamond witness of chapter 3595, and the confluence audit witness of chapter 745. The carrier isolates the substitution-stability gate that must remain visible before any conditional parallel-diamond route is read as a closed confluence theorem.

Definition 2819.1 (MetaCIC confluence substitution-boundary carrier)**.** *A* MetaCICConfluenceSubstitutionBounda *carrier is a finite* BHist *packet*

$$S = (T, R, U, D, J, O, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite typed-term window containing the source term, the two one-step parallel reducts, and the candidate common reduct.*
2. *R is the redex-frontier row, exposing the local redex shapes consumed by the confluence audit map.*
3. *U is the substitution-boundary row, recording the closed substitution support, binder-depth ledger, and free-variable non-escape checks needed by the branch.*
4. *D is the conditional parallel-diamond handoff row. It may be consumed only after T, R, U have been displayed.*
5. *J is the local join witness row, recording the candidate common reduct as a finite replay path rather than as a global Church–Rosser theorem.*
6. *O is the obstruction row naming the absent endpoints: unrestricted substitution compatibility, arbitrary subject reduction, and global closed confluence.*
7. *G is the* TasteGate *readiness row, reserving the later* Nontrivial *and* FieldFaithful *instance obligations for this finite carrier.*
8. *H records componentwise* hsame *transport for the typed-term window, redex frontier, substitution boundary, conditional-diamond handoff, local join, obstruction, TasteGate readiness, replay, provenance, and local name rows.*

9. *C records displayed* Cont *replay from a confluence-boundary query through $T, R, U, D, J, O, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$*, the confluence audit map, residual-substitution boundary rows, critical-pair rows, local-diamond rows,* BHist*,* hsame*,* Cont*,* Pkg*,* TasteGate*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for global confluence, unrestricted substitution compatibility, subject reduction, kernel consistency, evaluator equality, or hidden conversion.

Theorem 2819.2 (MetaCIC confluence substitution-boundary obligations)**.** *For every accepted* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$ *carrier $S = (T, R, U, D, J, O, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, redex-frontier exposure through R, substitution-boundary exposure through U, conditional parallel-diamond handoff through D, local join replay through J, obstruction retention through O, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The confluence-facing route is

$$T, R, U \longmapsto D, J, O \longmapsto G, H, C, P, N.$$

It is conditional at U and D: the packet records where substitution stability is available, not a closed confluence theorem.

Proof. Project the carrier of Definition 3791.1. The term-facing rows are T, R, U, D, J, O, and the structural rows are G, H, C, P, N.

The conditional branch begins with T, R, U. The typed-term window supplies the finite peak, the redex-frontier row supplies the local critical shape, and the substitution-boundary row supplies exactly the closed substitution support that must be checked before the diamond handoff is read.

The rows D, J, O then separate positive and negative content. The handoff and join rows record a local candidate route, while the obstruction row keeps unrestricted substitution compatibility, subject reduction, and global confluence outside the packet.

Finally G, H, C, P, N prepare the later TasteGate instance surface, transport the same rows, replay the same query, source it, and name it. Since no hidden conversion, evaluator equality, global confluence theorem, or kernel-consistency coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2819.3 (MetaCIC confluence substitution-boundary non-escape)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$ *carrier must factor through*

$$T, R, U, D, J, O, G, H, C, P, N.$$

If the substitution-boundary row U or the conditional handoff row D is absent, the packet no longer exports a confluence-boundary read. The remaining rows may still record typed terms, redex shapes, obstruction names, local join candidates, transport, replay, provenance, TasteGate readiness, and local naming, but they cannot supply global Church–Rosser, unrestricted substitution compatibility, arbitrary subject reduction, or kernel consistency.

Proof. Apply the obligation surface of Definition 3791.2. A confluence-audit read reaches the local join only through the typed window, redex frontier, substitution boundary, and conditional handoff.

If U is absent, the substitution-stability premise is not displayed. If D is absent, the conditional parallel-diamond handoff is not displayed. In either case the finite packet no longer contains the route needed to treat J as a confluence-boundary read.

The rows O, G, H, C, P, N do not repair the missing premise. They retain obstruction names, reserve the later TasteGate obligations, and preserve the visible packet, but none is a coordinate for global Church–Rosser, unrestricted substitution compatibility, subject reduction, or consistency. □

Theorem 2819.4 (MetaCIC confluence substitution-boundary TasteGate readiness)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ *at the displayed finite carrier surface*

$$T, R, U, D, J, O, G, H, C, P, N.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the typed-term window, redex-frontier row, substitution-boundary row, conditional handoff, local join row, obstruction row, transport, replay, provenance, and local NameCert data.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceSubstitutionBoundaryUp.MetaCICConfluenceSubstitutionBoundaryTaste`
`single carrier alignment`

Proof. The readiness row G is part of the carrier in Definition 3791.1. It is deliberately tied to the whole tuple rather than to the obstruction row alone.

Nontriviality must therefore exhibit an accepted packet with all finite rows present. FieldFaithfulness must show that deleting any of $T, R, U, D, J, O, H, C, P, N$ changes the readable carrier surface. The theorem records those Lean obligations at paper level while leaving their proofs to the later formalization namespace. □

Falsifiable prediction: A MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ confluence-audit read factors through displayed typed-term windows, redex-frontier rows, substitution-boundary rows, conditional parallel-diamond handoffs, local join witnesses, obstruction rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, metacic-frontier-critical-pair, metacic-local-diamond-witness, metacic-confluence-audit-witness: MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a typed peak, redex frontier, substitution-boundary row, conditional handoff, local join candidate, obstruction ledger, TasteGate readiness, transport, replay, provenance, and local naming as one boundary packet.

Closure status for NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3791.1, Definition 3791.2, Definition 3791.3, and Definition 3791.4 seed the finite substitution-sensitive confluence-boundary packet for MetaCIC audit consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global Church–Rosser, unrestricted substitution compatibility, arbitrary subject reduction, kernel consistency, evaluator equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, redex-frontier exposure, substitution-boundary coverage, conditional handoff, local join replay, obstruction retention, TasteGate readiness, transport, replay, provenance, and non-escape for BEDC.Derived.MetaCICConfluenceSubstitutionBoundaryUp.

Constructive story: A MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ packet is generated from a typed-term peak window, redex-frontier rows, a substitution-boundary row, a conditional parallel-diamond handoff, a local join witness, an obstruction ledger, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2820

A Concrete Naming Certificate for $\mathsf{CauchyCompletionComonad}^{\uparrow}$

Following Mac Lane on comonads and Bishop–Bridges on constructive Cauchy completion, the $\mathsf{CauchyCompletionComonad}^{\uparrow}$ packet records the finite completion-read surface obtained by reading already displayed completion data back through counit and comparison rows. It is paired with the monad packet of chapter 378, the Kleisli consumer boundary of chapter 1845, the counit packet of chapter 1539, the comparison packet of chapter 1954, and the triangle-identity packet of chapter 1795. The analytic route is the same finite L10 exit through chapter 126, chapter 301, chapter 291, and chapter 38; the chapter names this dual read as finite NameCert data rather than as a quotient completion object.

Definition 2820.1 (CauchyCompletionComonad carrier)**.** *A* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *route whose source, unit, bind, finite-window, readback, and terminal-seal rows are being read in the opposite direction.*
2. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *consumer row, displaying how a completion-facing request reaches the same finite source and seal rows.*
3. *E is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *return row from displayed completion data to the real-facing boundary.*
4. *Q is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row comparing the monad-facing and counit-facing reads over the same finite completion route.*
5. *T is the* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *row identifying the two displayed unit-counit composites with the direct completion read.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected before any rational readback or terminal seal is exposed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row consumed by the completion read.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation ledger attached to the same finite windows.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D have been displayed.*

10. *H records componentwise* hsame *transport for $M, K, E, Q, T, S, R, D, A$ and the structural rows.*
11. *C records displayed* Cont *replay for consumers that inspect the comultiplication side, the counit side, the comparison side, or the terminal real-facing route.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionComonad}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected Cauchy representatives, host equality of completed objects, countable choice, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-axis evidence.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionComonadUp.CauchyCompletionComonadTasteGate single carrier alignment`

Theorem 2820.2 (CauchyCompletionComonad NameCert obligations)**.** *For every accepted* CauchyCompletionCom *carrier $G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the monad, Kleisli, counit, comparison, triangle, finite-window, regular-readback, dyadic-ledger, terminal-seal, transport, replay, provenance, and local naming rows;*
2. *comultiplication exposure, requiring every doubled completion-read request to pass through M, K, Q, T before a terminal seal can be inspected;*
3. *counit return, requiring every already displayed completion read to pass through E before it reaches S, R, D, A;*
4. *finite triangle compatibility, requiring the comultiplication-counit composites to reach the same displayed S, R, D, A route as the direct completion read;*
5. *classifier stability, requiring transported packets with the same displayed rows to be compared only by the carried* hsame *and* Cont *data;*
6. *non-escape, requiring P, N to source and name no quotient completion, selected representative, countable-choice schedule, host equality, ambient completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3792.1. The tuple lists the monad, Kleisli, counit, comparison, triangle, finite-window, regular-readback, dyadic-ledger, terminal-seal, transport, replay, provenance, and local naming rows.

Comultiplication exposure is read from M, K, Q, T. The monad and Kleisli rows give the doubled completion consumer surface, the comparison row aligns it with the already displayed completion read, and the triangle row identifies the visible composites with the direct route.

Counit return is read from E. Once completion data have already been displayed, the packet forces their return through S, R, D, A before any real-facing consumer sees a terminal seal.

The structural obligations are componentwise. The rows H and C preserve and replay only the displayed packet, while P and N source and name that packet. Since no excluded coordinate occurs in the carrier, no stronger completion object can be exported. □

Theorem 2820.3 (CauchyCompletionComonad comultiplication and counit route)**.** *Let* $G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier. Every consumer of the finite comultiplication-counit surface factors through the displayed route*

$$M, K \longmapsto Q, T \longmapsto E \longmapsto S, R, D \longmapsto A \longmapsto H, C, P, N.$$

The route exposes a completion read only after the monad-facing and Kleisli-facing rows have been compared with the counit-facing return and then passed through the finite $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. It exports no quotient completion, selected representative, host equality of completed objects, countable-choice selector, ambient completeness theorem, or bridge credential.*

Proof. Project the accepted carrier. The only doubled completion-read rows are M and K, so a comultiplication-facing request first reads the monad route and the Kleisli consumer boundary.

The middle comparison is forced by Q and T. The comparison row aligns the monad-facing read with the counit-facing return, and the triangle row identifies the two visible composites with the same direct completion route.

The counit row E then returns the displayed completion read to the terminal finite route. That route must pass through the finite window S, the regular readback R, the dyadic ledger D, and the real seal A.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no coordinate for quotient completions, selected representatives, host equality, countable choice, ambient completeness, or bridge evidence, the displayed route is the whole comonad surface. □

Theorem 2820.4 (CauchyCompletionComonad finite completion-read boundary)**.** *Every completion-facing,* $\mathsf{StreamName}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing,* $\mathsf{DyadicRatCore}^{\uparrow}$*-facing, or* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier is generated by the rows of Definition 3792.1. In particular, the packet gives later completion algebra and reflector consumers a finite non-quotient read surface, not an ambient completed object.*

Proof. Read the completion-facing side first. The only completion control rows are M, K, E, Q, T, so every completion-facing consumer must factor through the monad route, Kleisli route, counit return, comparison row, and triangle row.

Read the analytic side next. The only finite observation and terminal seal rows are S, R, D, A, so every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal read through those rows.

The structural rows H, C, P, N preserve the same surface and add no new carrier coordinate. Therefore algebra and reflector consumers can reuse the finite read boundary, but they cannot obtain a quotient completion, selected limit, host completed object, ambient completeness theorem, or bridge credential from it. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionComonad}^{\uparrow}$ packet is generated from CauchyCompletionMonad, CauchyCompletionKleisli, CauchyCompletionCounit, CauchyCompletionComparison, CauchyCompletionTriangleIdentity, StreamName, RegSeqRat, DyadicRatCore, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$, seed)

Scope closed: Definition 3792.1, Definition 3792.2, Definition 3792.3, and Definition 3792.4 seed the finite dual completion-read surface over the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected Cauchy representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comultiplication exposure, counit return, triangle compatibility, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionComonadUp.

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ lattice is routed as a Cantor-to-Real diagonal boundary through chapter 1530, chapter 5879, chapter 301, and chapter 38.

Chapter 2821

A Concrete Naming Certificate for $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$

Following Cantor's diagonal argument, the $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet records the finite BEDC boundary that prevents a displayed countable $\mathsf{StreamName}^{\uparrow}$ enumeration from being read as exhaustive $\mathsf{Real}^{\uparrow}$ data. It sits over the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic bit ledger of chapter 291, the terminal real-name seal of chapter 38, and the comparison setoid boundary of chapter 1820. The packet names only the inspected enumeration rows, the diagonal bit-flip row, the regular readback route, transport, replay, provenance, and local naming data; it is not a host real-uncountability theorem, not a quotient-stream principle, and not a completed powerset of reals. The downstream Cantor diagonal real-uncountability packet of **??** consumes this finite non-surjection boundary as its Real-facing diagonal obstruction before adding CantorSpace and CantorSet rows.

The dependency direction is one-way at this lattice node: $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ consumes $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ to compare the diagonal witness with inspected rows, while CantorDiagonalRealUncountab consumes this chapter as the Real-facing obstruction before adding CantorSpace and CantorSet rows.

The diagonal-window side is read through chapter 4080. That sibling supplies the finite Cauchy diagonal-window selector used before the bit-flip row F and the terminal Real seal row S are consumed, so this chapter records the non-surjection witness as a finite selector-to-flip route rather than as only a kernel-to-Real route.

Definition 2821.1 (Real diagonal non-surjection carrier)**.** *A* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, W, B, F, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the countable-enumeration row, displaying the finite slice of the proposed* $\mathsf{StreamName}^{\uparrow}$ *enumeration inspected by the packet.*
2. *W is the row-index schedule, listing the inspected diagonal rows and the finite windows used for each row.*
3. *B is the dyadic-bit observation row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any real-name seal is exposed.*
4. *F is the diagonal bit-flip row, recording for each inspected row the opposite dyadic bit at the matching diagonal window.*

5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed bit-flip data into regular rational real-name observations.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the diagonal witness, exposed only after E, W, B, F, R have been displayed.*
7. *H records componentwise* hsame *transport for E, W, B, F, R, S and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an enumeration request through*

$$E, W, B, F, R \longmapsto S \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *row.*

The carrier contains no host real equality test, quotient stream equality, completed powerset of real names, countable-choice schedule, selected global enumeration inverse, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge credential.*

Theorem 2821.2 (Real diagonal non-surjection NameCert obligations)**.** *For every accepted* RealDiagonalNonSurjec *carrier $D = (E, W, B, F, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, enumeration-slice exposure through E, finite diagonal-window accounting through W, dyadic-bit readback through B, bit-flip exactness through F, regular rational readback through R, terminal real-name sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealDiagonalNonSurjectionUp.RealDiagonalNonSurjectionNameCertObligations`

Proof. Project the carrier of Definition 3793.1. The analytic rows are the enumeration slice E, the finite diagonal schedule W, the dyadic observation row B, the flipped-bit row F, the regular rational readback R, and the terminal seal S.

The enumeration obligations are finite. A consumer sees only the rows named by W, and for each such row it sees only the dyadic window displayed in B before the corresponding flipped bit is recorded in F.

The real-name handoff is routed through R and then S. Thus the diagonal witness is read as regular rational data before it is sealed as $\mathsf{Real}^{\uparrow}$ data, and the Cauchy-equivalence boundary remains a displayed comparison surface rather than a quotient of streams.

Transport and replay preserve the same rows componentwise. Since the carrier has no coordinate for host real equality, quotient streams, completed powersets, countable choice, selected enumeration inverses, ambient real completeness, Lean verification, or bridge checking, the listed rows exhaust the local certificate. □

Theorem 2821.3 (Real diagonal non-surjection diagonal witness)**.** *Every accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet exports a displayed diagonal witness route*

$$\begin{gathered} \textit{inspected row } n, \quad \textit{dyadic bit } b_n, \\ b_n \longmapsto 1 - b_n \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}. \end{gathered}$$

For each inspected enumeration row, the witness differs from that row at the matching displayed dyadic window before the terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed. Therefore a countable* $\mathsf{StreamName}^{\uparrow}$ *enumeration cannot be used as an exhaustive Real-facing packet unless it supplies an additional displayed row that defeats the bit-flip witness through the same* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route.*

Proof. Start with an accepted carrier $D = (E, W, B, F, R, S, H, C, P, N)$. For a row listed in W, the enumeration slice E exposes the row and B exposes the dyadic bit observed at the matching diagonal window.

The row F records the opposite bit at that window. The displayed step is

$$b_n \longmapsto 1 - b_n.$$

This is a finite dyadic row, not an appeal to host real equality or to a completed set of all real names.

The readback row R repacks the flipped bits as regular rational observations, and S is the only terminal $\mathsf{Real}^{\uparrow}$ seal available to the witness. Any comparison with the inspected row must therefore pass through the displayed Cauchy-equivalence setoid route, not through quotient streams or ambient completeness.

Consequently every inspected row is separated from the diagonal witness at its own displayed dyadic window. The packet refuses host real equality, quotient streams, completed powersets, countable choice, and ambient Real completeness as sources of surjectivity data. □

Theorem 2821.4 (Real diagonal non-surjection diagonal sequence route)**.** *Every accepted* RealDiagonalNonSurjecti *diagonal witness route of Definition 3793.3 factors through the reusable* $\mathsf{DiagonalSequence}^{\uparrow}$ *schedule of chapter 3940 before the dyadic bit flip and terminal* $\mathsf{Real}^{\uparrow}$ *seal are read. The* $\mathsf{DiagonalSequence}^{\uparrow}$ *row supplies only the finite row-index and window schedule used by* $E, W, B, F, R, S, H, C, P, N$*; it supplies no host countability theorem, no completed enumeration, and no Real surjectivity principle.*

Proof. Project the accepted carrier $D = (E, W, B, F, R, S, H, C, P, N)$. The rows E and W expose the inspected enumeration slice and the finite row-index schedule before any dyadic bit is flipped.

Read this schedule through chapter 3940. The sibling packet contributes the reusable diagonal row and finite-window discipline, so the present carrier does not need a separate kernel-level scheduling surface.

Only after that schedule is displayed may B and F read the inspected dyadic bit and its opposite. The readback row R and seal row S then turn the flipped finite row into the displayed $\mathsf{Real}^{\uparrow}$ witness.

The structural rows H, C, P, N preserve the same route. Since $\mathsf{DiagonalSequence}^{\uparrow}$ supplies only finite schedule data, the factorization imports no host countability theorem, completed enumeration, quotient stream equality, or Real surjectivity principle. □

Theorem 2821.5 (Real diagonal non-surjection StreamName dependency)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *read that exposes a diagonal non-surjection witness must first expose the* $\mathsf{StreamName}^{\uparrow}$ *enumeration slice and its finite diagonal windows from chapter 126. Only after those rows are displayed can the carrier read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bit flip, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the diagonal witness route depends on the finite StreamName window surface before any dyadic flip or Real-facing seal is visible.*

Proof. Project the carrier of Definition 3793.1. The rows E and W are the enumeration slice and finite diagonal-window schedule, and both are read before the dyadic observation row B and flipped-bit row F.

The diagonal-sequence route of Definition 3793.4 preserves this order. It first exposes the finite schedule, then routes through the dyadic bit read, the regular rational readback R, and the Real seal S.

Therefore every downstream Cantor or Real consumer sees a StreamName dependency theorem: the diagonal obstruction is not a direct host uncountability claim, but a finite path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Theorem 2821.6 (Real diagonal non-surjection Cantor-RegSeqRat dependency)**.** *Every Cantor-style consumer of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet must read the diagonal obstruction in the order*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{CantorSpace}^{\uparrow}/\mathsf{ConstructiveC}$$

Thus the packet consumed by the Cantor diagonal real-uncountability route is a finite Real-facing diagonal obstruction: it exposes the StreamName enumeration slice, the dyadic bit flip, the regular rational readback, the Cauchy-equivalence comparison boundary, the CantorSpace and constructive Cantor theorem route, and the terminal Real seal before any downstream Real consumer may cite non-surjection.

Proof. Start with the carrier of Definition 3793.1. It lists the StreamName enumeration slice E, W, the dyadic observation and flip rows B, F, the RegSeqRat readback R, the Real seal S, and the Cauchy-equivalence comparison route as displayed packet data.

Apply the diagonal witness theorem Definition 3793.3. The witness is separated from each inspected row at a displayed dyadic window before it is read back through $\mathsf{RegSeqRat}^{\uparrow}$ and sealed as $\mathsf{Real}^{\uparrow}$.

Apply Definition 3793.5 and the Cantor sibling route of Definition 3793.7. They force Cantor consumers to read the StreamName slice first and to pass through the CantorSpace and constructive Cantor theorem route only after the dyadic, regular-rational, and Cauchy-equivalence rows are visible.

Consequently the consumer receives a finite Real-facing diagonal obstruction. It does not receive a host uncountability theorem, a completed powerset of reals, quotient-stream equality, countable choice, or an ambient Real completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet is generated from a finite StreamName enumeration slice, a diagonal row-index schedule, a DyadicRatCore bit-observation row, a flipped-bit row, a RegSeqRat readback route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3793.1, Definition 3793.2, Definition 3793.3, Definition 3793.4, and Definition 3793.5 seed the finite diagonal boundary and reusable StreamName-to-Real schedule for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealDiagonalNonSurjectionUp`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, completed powersets of real names, countable choice, selected enumeration inverses, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, enumeration-slice exposure, diagonal-window accounting, dyadic-bit readback, bit-flip exactness, RegSeqRat readback, RealUp sealing, Cauchy-equivalence comparison routing, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealDiagonalNonSurjectionUp.

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet is a Real-facing diagonal boundary rather than a kernel-only island. Its diagonal witness consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 for the displayed regular-rational readback, and the Cantor diagonal real-uncountability route of **??** consumes this chapter as its Real-facing non-surjection input before adding CantorSpace and CantorSet rows.

The non-surjection carrier first reads the reusable diagonal schedule of chapter 3940: finite stream windows select the inspected enumeration row, the diagonal bit disagreement is recorded at the selected coordinate, the $\mathsf{RegSeqRat}^{\uparrow}$ readback exposes the rational window that realizes the disagreement, and the terminal $\mathsf{Real}^{\uparrow}$ seal packages that readback as the Real-facing obstruction. Only after this route is visible may **??** and chapter 5879 consume the packet as a Cantor-facing diagonal obstruction.

Theorem 2821.7 (RealDiagonalNonSurjection Cantor sibling route)**.** *Every accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier consumed by a Cantor-style diagonal argument remains a Real-facing sibling route through* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The consumer first reads the finite enumeration slice, diagonal row-index schedule, dyadic bit-observation row, flipped-bit row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison boundary, transport, replay, provenance, and local naming rows of Definition 3793.1; only then may the Cantor route of chapter 5879 interpret the same disagreement ledger as a Cantor-space non-enumeration witness.*

Proof. Start with the carrier and diagonal witness rows of Definition 3793.1 and Definition 3793.3. They expose the inspected enumeration row, finite diagonal window, dyadic bit read, flipped-bit row, regular-rational readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before any non-surjection conclusion is read.

Next route the displayed disagreement through the Cantor-space handoff of chapter 5879. That sibling consumes the same finite diagonal classifier over $\mathsf{CantorSpace}^{\uparrow}$ and carries it toward the Real-facing non-surjection boundary; it does not introduce a completed binary stream, host powerset, or ambient surjectivity principle.

Finally use the sibling dependency route already recorded for this chapter. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ comparison and $\mathsf{Real}^{\uparrow}$ seal remain visible on the Real side, so the packet is a Cantor/Real diagonal sibling route rather than a kernel-only packet. □

Theorem 2821.8 (Real diagonal non-surjection RegSeqRat dependency route)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *read that constructs the anti-diagonal* $\mathsf{Real}^{\uparrow}$ *row first exposes the finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and* $\mathsf{Real}^{\uparrow}$ *seal used by the source sequence. The consumer cannot bypass those rows with host stream equality, selected limits, quotient reals, ambient completeness, countable choice, or a hidden real-surjectivity principle.*

Proof. Start with the carrier of Definition 3793.1. The rows E and W expose only the inspected finite enumeration slice and diagonal windows, so the anti-diagonal read begins as finite $\mathsf{StreamName}^{\uparrow}$ data.

Next apply the diagonal witness route of Definition 3793.3. The dyadic bit read and bit flip pass through B and F, then the regular-rational readback R repacks the displayed windows before the terminal seal S is exposed.

Use the existing sibling dependency route of this chapter to keep the same boundary visible to downstream consumers. The sibling route already records that the diagonal packet is Real-facing, and chapter 301 and chapter 38 supply the named readback and seal surfaces.

Since the carrier has no coordinate for host stream equality, selected limits, quotient reals, ambient completeness, countable choice, or hidden surjectivity data, every anti-diagonal Real row must follow the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. □

Theorem 2821.9 (Real diagonal non-surjection Cantor–RegSeqRat lock)**.** *Every accepted* RealDiagonalNonSurjecti *packet that claims a non-surjectivity read must factor through the Cantor diagonal row of chapter 5879, the regular rational readback row of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38, and the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *ledger before the conclusion is exposed. In*

carrier coordinates this means that the enumeration slice E, diagonal schedule W, dyadic bit row B, flipped-bit row F, RegSeqRat row R, Real seal S, transport row H, replay row C, provenance row P, and naming row N of Definition 3793.1 remain visible in that order; no classical choice principle, host powerset, quotient stream equality, selected inverse, or ambient real-completeness row is introduced by the lock.

Proof. Start with the carrier and diagonal witness rows of Definition 3793.1 and Definition 3793.3. They expose $E, W, B, F, R, S, H, C, P, N$ before any non-surjectivity read is available.

Next use the Cantor sibling route of Definition 3793.7. The Cantor-facing consumer receives the same displayed disagreement row and not a host powerset, completed binary stream, or selected inverse.

Then apply the RegSeqRat dependency route of Definition 3793.8 and the Real source route of Definition 3793.11. The flipped bit row is read through R as regular rational data before S exposes the terminal Real-facing seal.

Finally the structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for classical choice, quotient stream equality, selected enumeration inverse, host powerset, or ambient real completeness, every accepted non-surjection claim remains locked to the Cantor, RegSeqRat, RealUp, and local NameCert rows. □

Theorem 2821.10 (Real diagonal non-surjection RegSeqRat–Real route)**.** *Every accepted* RealDiagonalNonSurjecti *read that exposes a diagonal non-surjection packet must consume the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row of chapter 301, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of chapter 38, and the Cantor-style diagonal sibling rows of Definition 3793.7 before the non-surjection surface is available.*

In carrier coordinates this route reads $E, W, B, F, R, S, H, C, P, N$ from Definition 3793.1: the enumeration slice and window schedule are exposed first, the flipped dyadic bit is repacked through R as regular rational data, the terminal seal S presents the Real-facing row, and the Cantor sibling consumes the same displayed disagreement ledger. No host stream equality, selected inverse, quotient real equality, completed powerset, or ambient Real-completeness row is introduced by the route.

Proof. Project the carrier of Definition 3793.1. The rows E and W expose only the inspected finite enumeration slice and diagonal windows, so the non-surjection read begins as $\mathsf{StreamName}^{\uparrow}$ window data.

Apply Definition 3793.8. The bit observation and flipped bit are read through B and F, then the row R repacks the disagreement as regular rational data before a Real-facing endpoint is visible.

Next apply Definition 3793.12. The seal row S is the only $\mathsf{Real}^{\uparrow}$ endpoint available to the diagonal witness, and it is exposed only after the regular readback has been displayed.

Finally use Definition 3793.7. The Cantor-style sibling consumes the same finite disagreement ledger after the StreamName, RegSeqRat, and Real seal rows are visible. The structural rows H, C, P, N transport, replay, source, and name that route, so no excluded host object can enter the non-surjection packet. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ completion-facing dependency route consumes the regular-rational readback of chapter 301 and the terminal real-name seal of chapter 38 before a diagonal non-surjection packet may refuse a surjective enumeration. This anchor keeps the L10 readback lattice visible: the inspected enumeration slice and finite diagonal windows are read first, the disagreement is repacked through $\mathsf{RegSeqRat}^{\uparrow}$, the resulting row is sealed by $\mathsf{Real}^{\uparrow}$, and only then can the packet deny that the displayed enumeration is exhaustive.

Theorem 2821.11 (Real diagonal non-surjection Real source route)**.** *Every diagonal non-surjection read of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet consumes the* $\mathsf{Real}^{\uparrow}$ *source row of chapter 38, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row of chapter 126, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301 before exposing a diagonal packet. The route exposes only the finite enumeration slice, diagonal window schedule, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows; it does not claim host uncountability, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 3793.1. The enumeration slice E and finite diagonal schedule W are read as $\mathsf{StreamName}^{\uparrow}$ source data before the bit observation row B or flipped-bit row F can be interpreted.

Apply the RegSeqRat dependency route of Definition 3793.8. The anti-diagonal readback passes through R as regular rational data and cannot bypass that row with host stream equality, selected limits, quotient reals, countable choice, or hidden surjectivity data.

Finally use the Real seal dependency of Definition 3793.12. The seal row S is the only $\mathsf{Real}^{\uparrow}$ endpoint available to the diagonal witness, and it is exposed after the stream and regular-rational source rows have already been read. The diagonal packet is therefore a downstream consumer of the $\mathsf{Real}^{\uparrow}$ terminal seal, the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback, so its non-surjection surface is unavailable unless those sibling NameCert chapters have supplied their finite rows.

Thus every diagonal packet is source-routed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the non-surjection surface is visible. The carrier has no coordinate for host uncountability, quotient stream equality, countable choice, or ambient Real completeness. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet consumes $\mathsf{RealBaireCylinder}^{\uparrow}$ as the finite prefix-cylinder readback surface for Real-facing diagonal rows. The sibling route through chapter 1685 first exposes the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, then the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal; the diagonal row therefore reads the Baire cylinder as displayed Real-facing data rather than as a host stream, quotient real equality, selected branch, or ambient completeness principle.

Theorem 2821.12 (Real diagonal non-surjection Real seal dependency)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *diagonal read reaches its contradiction surface only after the* $\mathsf{Real}^{\uparrow}$ *seal route of chapter 38 and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window rows are visible. The packet is not a host cardinality theorem, not a quotient-surjection argument, and not an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3793.1. The only analytic rows available before the terminal real-name handoff are the enumeration slice E, the finite diagonal schedule W, the dyadic bit row B, the bit-flip row F, and the regular rational readback R.

Apply Definition 3793.3. The contradiction surface is reached by comparing an inspected enumeration row with the flipped bit at the displayed diagonal window. That comparison is read through $\mathsf{DyadicRatCore}^{\uparrow}$ and then through $\mathsf{RegSeqRat}^{\uparrow}$ before the seal row S can expose the witness as $\mathsf{Real}^{\uparrow}$ data.

Now use the $\mathsf{Real}^{\uparrow}$ seal route of chapter 38. The seal does not add a separate cardinality principle; it only marks that the already displayed regular-rational finite windows have been admitted as the real-name row used by the diagonal witness.

Since the carrier has no coordinate for host cardinality, quotient surjection, selected enumeration inverse, completed powerset, countable choice, or ambient real completeness, every diagonal

contradiction remains inside the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal dependency route. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ dependency lattice consumes the Cantor-style diagonal split, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the terminal $\mathsf{Real}^{\uparrow}$ seal through chapter 1530, chapter 301, and chapter 38. This route exposes only the finite diagonal disagreement, regular rational readback, and Real sealing rows; it does not claim cardinality, a powerset object, quotient equality, or host diagonal completeness.

Theorem 2821.13 (Real diagonal non-surjection metric dependency route)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer that compares outputs of a* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet must first read the diagonal witness through the* $\mathsf{RegSeqRat}^{\uparrow}$ *route of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. Only after those rows are exposed may the consumer use the metric comparison surface of chapter 57; the diagonal packet itself exports no direct metric distance row, quotient real equality, selected limit, countable-choice schedule, or ambient completeness witness.*

Proof. Start with the carrier of Definition 3793.1. The rows B and F produce the displayed dyadic bit disagreement, but this data is still a finite diagonal observation rather than a metric comparison.

The readback row R converts the displayed bit-flip data into the regular rational route named by Definition 3793.8. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ endpoint visible to a downstream consumer.

A $\mathsf{Met}^{\uparrow}$ consumer can compare diagonal outputs only after it has both the regular rational readback and the terminal real seal. Thus the dependency order is

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}.$$

Transport, replay, provenance, and local naming preserve this order componentwise. Since no carrier row names a host metric distance directly, the metric consumer cannot bypass the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ sibling routes. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ dependency route combines the Cantor-style diagonal cylinder, the $\mathsf{RegSeqRat}^{\uparrow}$ finite readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the $\mathsf{Met}^{\uparrow}$ separation surface in one visible order. A consumer first reads the finite binary disagreement through the Cantor route of chapter 5879 and the regular-rational readback of chapter 301; only then does the route expose the Real seal of chapter 38 and the metric comparison boundary of chapter 57. The packet therefore supplies a concrete dependency edge from diagonal cylinders to metric separation without adding host cardinality, quotient real equality, selected branches, countable choice, or ambient completeness.

Theorem 2821.14 (RealDiagonalNonSurjection forward Cantor consumer route)**.** *Every* $\mathsf{CantorDiagonalRealUncou}$ *consumer of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier must first read the finite diagonal obstruction through the displayed* $\mathsf{StreamName}^{\uparrow}$ *enumeration slice,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *bit-observation and bit-flip rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the sibling Cantor route of Definition 3793.7. The consumer may then pass to* **??***, but only with the Real-facing seal and comparison boundary already constrained by chapter 38.*

Proof. Project the accepted carrier of Definition 3793.1. Its enumeration slice, row-index schedule, dyadic bit-observation row, flipped-bit row, regular rational readback, and Real seal are all displayed before the local diagonal witness is available.

Apply the StreamName dependency theorem and the Cantor sibling route. Theorems 3793.5 and 3793.7 force the consumer through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the disagreement ledger can be interpreted as Cantor-facing non-enumeration data.

The downstream chapter **??** therefore receives the diagonal obstruction as a finite Real-facing route, not as a free-standing host uncountability principle. The Real seal remains governed by chapter 38, so the consumer imports no quotient stream equality, completed powerset, countable choice, selected enumeration inverse, or ambient completeness. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ route is the Real-facing obstruction consumed by **??** and by chapter 5879. The downstream Cantor read occurs only after $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ have already exposed the finite name, rational readback, dyadic tolerance, real seal, and equivalence-setoid rows. The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ sibling lattice consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular readback route of chapter 301, the CantorSpace chapter of chapter 1475, and the Cantor-style diagonal handoff of Definition 3793.7. These dependencies keep the non-surjection packet on the displayed Real, RegSeqRat, CantorSpace, and Cantor diagonal rows, with no host enumeration theorem, quotient stream equality, or ambient completeness coordinate. The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ completion-facing diagonal route is read through the $\mathsf{CantorSpace}^{\uparrow}$ finite-prefix surface of chapter 1475, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion-facing siblings chapter 360 and chapter 1506. The Cantor row supplies only the displayed binary-prefix disagreement, the regular readback row repacks that disagreement as rational observations, and the Real seal is exposed before any completion consumer may inspect the packet. The route does not claim host cardinality, quotient real equality, selected infinite branches, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 2822

A Concrete Naming Certificate for $\mathsf{CauchyNetCompletion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet records the directed Cauchy-net completion carrier that sits between the raw $\mathsf{CauchyNet}^{\uparrow}$ directed-window surface and the completion consumers already named by $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$. It consumes the same separated comparison discipline as chapter 1411 and the same finite numerical readback route as chapter 301 and chapter 38. The chapter names the completion-facing carrier without selecting a limit, quotienting nets, invoking countable choice, asserting host equality of completed objects, importing an ambient completion, or claiming completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2822.1 (CauchyNetCompletion carrier)**.** *A* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, M, U, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index substrate read by the accepted* $\mathsf{CauchyNet}^{\uparrow}$ *row.*
2. *W is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window and common-upper-witness row over D.*
3. *Q is the completion-boundary request attached to the displayed Cauchy-net row.*
4. *M is the* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *handoff row, exposing the same directed-net completion boundary to Moore–Smith consumers.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row when the consumer reads entourages rather than metric balls.*
6. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row for comparing two displayed completion-boundary reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback row attached to the retained directed windows.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after D, W, Q, M, U, S, R have been displayed.*
9. *H records componentwise* hsame *transport for the directed substrate, Cauchy-net row, completion-boundary request, Moore–Smith handoff, uniform handoff, separated comparison, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from directed Cauchy-net evidence through completion-boundary reads, separated comparison, regular-rational readback, and the real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient nets, countable choice, host equality of completed objects, an ambient completion object, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionTasteGate single carrier alignment`

Theorem 2822.2 (CauchyNetCompletion NameCert obligations)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the directed substrate, Cauchy-net row, completion-boundary request, Moore–Smith handoff, uniform handoff, separated comparison, regular readback, real seal, transport, replay, provenance, and local naming row;*
2. *directed-net coverage, requiring every completion-facing request to consume D and W before Q, M, U, S, R, A can be inspected;*
3. *Moore–Smith handoff, requiring M to expose the same directed-net boundary that chapter 2190 consumes;*
4. *metric and uniform handoff, requiring* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumers to read the boundary through Q and U rather than through an ambient completed object;*
5. *separated comparison discipline, requiring any comparison of two displayed boundary reads to pass through S;*
6. *numerical readback discipline, requiring R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal A is inspected;*
7. *non-escape for selected limits, quotient nets, countable choice, host equality, ambient completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3794.1. The tuple displays exactly the directed substrate, the Cauchy-net window row, the completion-boundary request, the Moore–Smith and uniform handoffs, separated comparison, readback, seal, and structural rows.

Directed-net coverage is the first dependency. A completion-facing request cannot reach Q, M, U, S, R, A until the accepted $\mathsf{CauchyNet}^{\uparrow}$ row W over D has been displayed.

The handoff obligations use the sibling completion chapters. The row M feeds the Moore–Smith completion route, while Q and U feed metric and uniform completion consumers over the same displayed directed windows. The separated comparison row S supplies comparison data without host equality.

Finally R and A provide the finite numerical readback and terminal seal, and H, C, P, N preserve, replay, source, and name the same route. Since the excluded completion principles have no carrier coordinate, they cannot become NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier namecert obligations`

Theorem 2822.3 (CauchyNetCompletion Moore–Smith handoff)**.** *Every* $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *packet reads the directed-net completion boundary through*

$$D, W, Q \longmapsto M, U, S \longmapsto R, A \longmapsto H, C, P, N.$$

The route starts with the $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, passes through the Moore–Smith, metric, uniform, and separated comparison rows when those consumers are present, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the* $\mathsf{Real}^{\uparrow}$ *seal only afterward. No quotient net, selected limit, countable-choice schedule, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness row bypasses this route.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier moore smith handoff`

Proof. Project the accepted carrier of Definition 3794.1. The rows D and W are the only directed-net source rows, and Q is the first completion-boundary request attached to them.

Read the sibling handoff next. The Moore–Smith consumer receives M, metric completion consumers read the same boundary through Q, and uniform completion consumers receive U only as a handoff from the displayed Cauchy-net boundary. Comparisons of displayed boundary reads pass through S.

The numerical exit is downstream. The regular rational row R and real seal row A are available only after the directed-net and completion-boundary rows have been replayed, and the structural rows H, C, P, N preserve and name that route.

The carrier inventory contains no selected limit, quotient-net equality, countable-choice selector, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness theorem. Thus every accepted completion consumer factors through the displayed route. □

Theorem 2822.4 (CauchyNetCompletion choice-free limit boundary)**.** *Any accepted completion-limit read from a* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *packet factors through the displayed rows*

$$D, W, Q, M, U, S, R, A, H, C, P, N.$$

The separated comparison row S *mediates comparison of displayed boundary reads, and the rows* R, A *provide the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal only after the directed windows and completion handoff rows have been exposed. The read exports no selected limit, quotient-net representative, countable-choice schedule, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier choice free limit boundary`

Proof. Project the carrier of Definition 3794.1. Its coordinates are exactly $D, W, Q, M, U, S, R, A, H, C, P, N$, with D, W supplying the directed substrate and Cauchy-net windows before any completion-boundary request is inspected.

Apply the Moore–Smith handoff theorem. Theorem 3794.3 forces every Moore–Smith, metric, or uniform completion consumer through the displayed route from D, W, Q to M, U, S, then to R, A, and finally to the transport, replay, provenance, and naming rows.

The comparison and numerical exits are therefore internal to the same packet. The row S is the only separated comparison coordinate, while R, A are the only regular-rational readback and real-seal coordinates reachable by the consumer.

It remains to exclude escape data. The carrier definition has no coordinate for a selected limit, quotient net, countable-choice selector, host equality, ambient completion object, or $\mathsf{Real}^{\uparrow}$ completeness theorem, so no accepted completion-limit read can export any of those objects. □

Theorem 2822.5 (CauchyNetCompletion directed-window terminality)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier*

$$K = (D, W, Q, M, U, S, R, A, H, C, P, N),$$

every completion-facing read of $Q, M, U, S, R, A, H, C, P, N$ *has terminal directed-window source* D, W*. In particular, no selected limit, quotient net, countable-choice selector, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness row can supply the first directed witness for the packet.*

Proof. Project the carrier in Definition 3794.1. The only directed substrate row is D, and the only Cauchy-net directed-window row over it is W.

Apply the choice-free boundary theorem, Definition 3794.4. It places all completion-limit reads on the displayed route through $D, W, Q, M, U, S, R, A, H, C, P, N$.

The remaining rows are downstream witnesses. The request row Q, the Moore–Smith and uniform handoff rows M, U, the separated comparison row S, and the readback rows R, A can be inspected only after D, W have been displayed.

Since the excluded objects have no carrier coordinate, none of them can replace D, W as the directed source of an accepted read. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowTerminality`

Theorem 2822.6 (CauchyNetCompletion directed-window cofinal restriction)**.** *Let* $K = (D, W, Q, M, U, S, R, A, H$ *be an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier. Replace the directed substrate row* D *by a displayed cofinal restricted substrate*

$$D_{\mathrm{cof}} \hookrightarrow D$$

and replace W *by its restricted* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row* W_{cof}*, provided* W_{cof} *still carries the common-upper witnesses used by* W *for completion-boundary requests. Then the completion-facing route*

$$D, W, Q, M, U, S, R, A, H, C, P, N$$

is preserved as the restricted route

$$D_{\mathrm{cof}}, W_{\mathrm{cof}}, Q, M, U, S, R, A, H, C, P, N.$$

In particular, the restricted packet exports no selected limit, quotient net, countable-choice selector, host equality of completion points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3794.1. The directed substrate row D and directed-window row W are the only source rows before the completion-boundary request Q is read.

Use directed-window terminality, Definition 3794.5. It makes D, W terminal for every completion-facing read. A cofinal restricted substrate with the same common-upper witnesses supplies the same terminal source position through $D_{\mathrm{cof}}, W_{\mathrm{cof}}$.

Now apply the Moore–Smith handoff theorem, Definition 3794.3. The completion route still passes from the directed source through Q to the Moore–Smith, metric, uniform, and separated rows M, U, S, then to the regular readback and Real seal rows R, A.

The structural rows are unchanged. Componentwise transport H, displayed replay C, provenance P, and local naming N preserve the same restricted route, while the carrier has no coordinate for selected limits, quotient nets, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowCofinalRestriction`

Theorem 2822.7 (CauchyNetCompletion directed-window refinement functoriality)**.** *Let* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier. Suppose a first displayed cofinal restriction replaces* D, W *by* $D_{\mathrm{cof}}, W_{\mathrm{cof}}$*, and a second displayed cofinal restriction replaces those rows by*

$$D_{\mathrm{ref}}, W_{\mathrm{ref}}.$$

If each restricted window retains the common-upper witnesses needed for the same completion-boundary request Q*, then the composite restricted route preserves the same CauchyNetCompletion surface:*

$$D, W, Q, M, U, S, R, A, H, C, P, N \quad \rightsquigarrow \quad D_{\mathrm{ref}}, W_{\mathrm{ref}}, Q, M, U, S, R, A, H, C, P, N.$$

The composite route exports no selected limit, quotient net, countable-choice schedule, host equality of completion points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with directed-window terminality, Definition 3794.5. It makes the displayed directed-window rows the source of every completion-facing read, so the first cofinal restriction can replace D, W only by another displayed terminal source row for the same request.

Apply cofinal restriction, Definition 3794.6, to the first restriction and then to the second. The retained common-upper witnesses keep the completion-boundary request Q and the handoff rows $M, U, S, R, A, H, C, P, N$ on the same route.

Use the Moore–Smith handoff theorem to read the composite as one completion-facing package. The sibling Moore–Smith, metric, uniform, separated, regular-rational, and Real rows are not recomputed; they are transported and replayed through the final restricted window.

The carrier has no coordinate for selected limits, quotient nets, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness. Thus composing two displayed cofinal restrictions is functorial only for the visible directed-window route, and it cannot export any excluded completion principle. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowRefinementFunctoriality`

Theorem 2822.8 (CauchyNetCompletion separated-comparison exhaustion)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$*, every comparison between two displayed completion-boundary reads is exhausted by the separated row* S*. The comparison may use* H, C, P, N *to preserve, replay, source, and name the row, but it exports no quotient-net equality, host equality, ambient completion equality, selected limit, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionSeparatedComparisonExhaustion`

Proof. Start from the NameCert obligations of Definition 3794.2. They identify S as the separated comparison row attached to the displayed completion-boundary request.

Use Definition 3794.4. Any completion-limit read must first pass through D, W, Q, M, U and reaches comparison only at S.

The structural rows do not add a new comparison source. They transport, replay, remember provenance, and name the same separated row, so the comparison inventory is exhausted by S.

The carrier has no quotient-net, host-equality, ambient-completion, selected-limit, countable-choice, or $\mathsf{Real}^{\uparrow}$-completeness coordinate. Those excluded objects therefore cannot appear as comparison evidence. □

Theorem 2822.9 (CauchyNetCompletion uniform handoff nonescape)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer of an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *must read its completion boundary through* U *after the directed-window source* D, W *and the request row* Q*. The handoff cannot escape through a selected limit, quotient net, countable-choice selector, host equality, ambient completion object, or* Real$^{\uparrow}$ *completeness row.*

Proof. Project the carrier. The uniform handoff row is U, and it is listed only after the directed substrate D, the Cauchy-net window W, and the completion-boundary request Q.

Apply Definition 3794.3. It sends every Moore–Smith, metric, or uniform completion consumer through the displayed route from D, W, Q to M, U, S.

The choice-free boundary theorem, Definition 3794.4, keeps the later readback rows R, A, H, C, P, N downstream of the same route.

Thus a uniform consumer that bypasses U or replaces D, W, Q by any excluded completion object is outside the accepted CauchyNetCompletion$^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionUniformHandoffNonescape`

Theorem 2822.10 (CauchyNetCompletion root filter admission)**.** *Every root completion consumer of an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *must admit the directed substrate* D*, retained directed-window row* W*, and completion-boundary request* Q *before Moore–Smith, metric, uniform, separated, readback, or Real-seal rows are inspected. No root consumer may replace* D, W, Q *by a selected limit, quotient net, countable-choice selector, host completion equality, ambient completion object, or* Real$^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootFilterAdmission`

Proof. Project the carrier of Definition 3794.1. The first three root rows are exactly D, W, Q, and the Moore–Smith, uniform, separated, readback, and Real-seal rows occur only downstream.

Apply the NameCert obligations of Definition 3794.2. Directed-net coverage requires a completion-facing request to consume D and W before Q, M, U, S, R, A can be inspected.

The Moore–Smith handoff, Definition 3794.3, keeps metric and uniform consumers on the same route. Thus root admission starts with D, W, Q for every completion consumer.

The excluded completion objects have no carrier coordinate, so none can serve as the admitted root filter row. □

Theorem 2822.11 (CauchyNetCompletion root eventual stability)**.** *Suppose an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier retains a directed-window row* W *over* D*, and a later common-upper-window witness replaces* W *by a retained row* W' *over the same directed substrate and completion-boundary request* Q*. Then the accepted packet is preserved through the separated comparison row* S*, regular readback row* R*, Real seal* A*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming row* N*.*

Proof. Start with the directed-window terminality theorem, Definition 3794.5. It makes D, W the terminal directed-window source for every completion-facing read.

The later common-upper-window witness is still a retained directed-window row over the same D and Q. Replacing W by W' therefore changes only the displayed window coordinate and not the completion-boundary request or the sibling handoff route.

Apply the choice-free boundary theorem, Definition 3794.4. The rows S, R, A are downstream of the directed-window and completion-boundary route, so their accepted reads are preserved by the later witness.

Finally the structural rows H, C, P, N transport, replay, source, and name the same retained packet. Since no selected limit, quotient net, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness row participates, eventual stability is a finite retained-window preservation claim. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootEventualStability`

Theorem 2822.12 (CauchyNetCompletion root seal nonescape)**.** *In every accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$*, the Real seal* A *cannot be read before the directed substrate* D*, directed-window row* W*, completion-boundary request* Q*, Moore–Smith handoff* M*, uniform handoff* U*, separated comparison row* S*, and regular readback row* R *have been displayed. Reading* A *exports no selected limit, quotient net, countable choice, host completion equality, ambient completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The row A is listed after D, W, Q, M, U, S, R, and the carrier gives no alternate Real-seal source.

Apply Definition 3794.3. It routes Moore–Smith, metric, and uniform completion consumers from D, W, Q through M, U, S before the numerical readback rows are reached.

Use the choice-free limit boundary of Definition 3794.4. That theorem places R, A, H, C, P, N downstream of the displayed directed-net completion route and excludes hidden limit data.

The carrier contains no selected-limit, quotient-net, countable-choice, host-equality, ambient-completion, or $\mathsf{Real}^{\uparrow}$-completeness coordinate. Therefore the root Real seal cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootSealNonescape`

Falsifiable prediction: If a directed-net completion consumer reads a boundary without first displaying the $\mathsf{CauchyNet}^{\uparrow}$ directed-window row and the Moore–Smith, metric, or uniform handoff row it consumes, it is not a $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet.

Independence witness: cauchynet, moore-smith-completion, metriccompletion, uniformcompletion, separatedmetric, regseqrat, real: the packet is not bijective to these siblings because it binds the directed Cauchy-net carrier, completion-boundary request, sibling handoffs, separated comparison, finite readback, and terminal seal in one local NameCert route.

Theorem 2822.13 (CauchyNetCompletion public handoff)**.** *The public completion surface of an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *is the reusable handoff package formed by the directed substrate* D*, the retained* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row* W*, the completion-boundary request* Q*, the Moore–Smith, metric, and uniform handoff rows* M, U*, the separated comparison row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A*, and the structural rows* H, C, P, N*. Any downstream completion consumer may reuse this package only as the displayed directed-net completion route; it receives no selected limit, quotient net, countable-choice schedule, ambient completion object, host equality of completed points, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the carrier and the NameCert obligations. They display the directed substrate, the Cauchy-net window, the completion-boundary request, the sibling completion handoffs, separated comparison, numerical readback, Real seal, transport, replay, provenance, and local naming rows.

Use the Moore–Smith handoff and choice-free limit boundary. These theorems force every Moore–Smith, metric, and uniform completion read through $D, W, Q, M, U, S, R, A, H, C, P, N$ and

exclude selected limits, quotient nets, countable choice, host equality, ambient completion, and Real completeness from the route.

Apply directed-window terminality, cofinal restriction, root eventual stability, and root seal nonescape. They make the directed-window source reusable under retained windows and cofinal restriction while keeping the terminal Real seal behind the displayed completion route.

Thus the named rows assemble a public handoff surface: a downstream consumer can cite the package as completion-facing data, but cannot replace it by selected limits, quotient nets, or an ambient Real-completeness principle. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionPublicHandoff`

Theorem 2822.14 (CauchyNetCompletion regular-sequence public bridge)**.** *Suppose an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *factors its directed Cauchy-net packet through a cofinal unary* $\mathsf{StreamName}^{\uparrow}$ *schedule. If the reads along that schedule satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity predicate and the separated comparison exhausted by* S *is the displayed* $\mathsf{Real}^{\uparrow}$ *seal* A*, then the* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *public handoff exports only the same regular-sequence completion data*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ S,\ A,\ H,\ C,\ P,\ N.$$

It exports no additional directed-net limit, quotient net, selected representative, countable-choice schedule, host equality of completed points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRegularSequencePublicBridge`

Proof. Start from the public handoff theorem. The public surface is the displayed route through $D, W, Q, M, U, S, R, A, H, C, P, N$, with no selected limit, quotient net, countable choice, host equality, ambient completion, or Real-completeness row.

Use directed-window terminality and cofinal restriction. The cofinal unary $\mathsf{StreamName}^{\uparrow}$ schedule may replace the displayed directed-window source only as a retained terminal source for the same completion-boundary request Q; it does not add a new limit coordinate.

Read the regularity assumption through the existing readback row R. Since the schedule reads satisfy the $\mathsf{RegSeqRat}^{\uparrow}$ regularity predicate, the numerical exit of the restricted route is the regular-sequence readback already named by R.

Apply separated-comparison exhaustion. The row S exhausts comparisons between displayed boundary reads, and by hypothesis its separated comparison is precisely the displayed $\mathsf{Real}^{\uparrow}$ seal A.

Finally use uniform-handoff nonescape and the Moore–Smith handoff. They keep every completion consumer on the same displayed route, so the public packet can export only the regular-sequence completion data carried by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, S, A, H, C, P, N and none of the excluded completion principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet is generated from a directed substrate, a CauchyNet directed-window row, a completion-boundary request, MooreSmithCompletion, MetricCompletion, UniformCompletion, SeparatedMetric, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 3794.1, Definition 3794.2, Definition 3794.5, Definition 3794.6, Definition 3794.11, Definition 3794.12, Definition 3794.13, and Definition 3794.14 expose the finite directed-net completion route as a public handoff and paper bridge through directed-window data, cofinal restriction, root eventual stability, root seal, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependencies.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyNetCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient nets, countable choice, host equality of completed objects, ambient completion, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a Lean standard-bridge target for the public directed-net completion handoff and regular-sequence public bridge.

Chapter 2823

A Concrete Naming Certificate for $\mathsf{RealTightness}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RealTightness}^{\uparrow}$ packet records located dyadic trap data over a displayed $\mathsf{Real}^{\uparrow}$ source as finite NameCert evidence. It consumes the real-name seal of chapter 38, the dyadic rational grid of chapter 291, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, the located interval surface of chapter 1709, the bounded located-set row of chapter 2176, and the regular Cauchy located-order discipline of chapter 1681. The packet gives later Real and Completion consumers a finite located-tightness route; it is not Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2823.1 (Real tightness carrier)**.** *A* $\mathsf{RealTightness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$$

with the following visible rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose name is being trapped.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid row supplying dyadic endpoints, midpoint refinements, and tolerance comparisons.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the requested real-name observations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
5. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row recording the current located interval trap.*
6. *B is the* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *bounded-source row that bounds the admitted trap requests.*
7. *O is the* $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ *row recording located-order comparisons between the real-name readback and the dyadic trap endpoints.*
8. *T is the located dyadic trap ledger: a finite list of intervals, dyadic tolerances, inside/outside locator reads, and retained-window witnesses.*
9. *H records componentwise* hsame *transport for R, D, S, Q, I, B, O, T and the structural rows.*
10. *C records displayed* Cont *replay of the real-name read, dyadic trap refinement, located-interval admission, bounded-set read, located-order comparison, and terminal trap ledger.*

11. *P is the* Pkg *provenance over* $\mathsf{RealTightness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedBoundedRealSet}^{\uparrow}$, $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$ *row.*

The carrier has no coordinate for Heine–Borel, open-cover compactness, trichotomy of arbitrary reals, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealTightnessUp.RealTightnessTasteGate single carrier alignment`

Theorem 2823.2 (Real tightness NameCert obligations)**.** *For an accepted* $\mathsf{RealTightness}^{\uparrow}$ *carrier* $Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real source, dyadic grid, stream windows, regular rational readback, located interval, bounded located set, located-order row, trap ledger, transport, replay, provenance, and local naming row;*
2. *dyadic trap exposure, requiring every retained trap in T to display dyadic endpoints from D and the matching finite observation window from S;*
3. *located interval admission, requiring each trap read to pass through I and the bounded-source row B before it can be consumed as a located real interval;*
4. *regular Cauchy located-order comparison, requiring the readback row Q and the comparison row O to certify the displayed relation between R and the dyadic trap endpoints;*
5. *replay stability under componentwise* hsame *transport and displayed* Cont *routes for R, D, S, Q, I, B, O, T;*
6. *nonescape for Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3795.1. The tuple lists exactly the real source, dyadic grid, finite window, regular readback, located interval, bounded-set, located-order, trap, transport, replay, provenance, and naming rows.

Trap exposure is finite. Each trap row in T reads dyadic endpoints and tolerances from D, then matches them to the finite observation window named by S and the regular rational readback named by Q.

Located admission is downstream of those displayed rows. The located interval row I records the interval trap, while B records that the trap request stays inside the bounded located-set surface available to the consumer.

The comparison step is the located-order row O. It compares the regular Cauchy readback of Q with the dyadic endpoints retained in T, and it is transported and replayed only by H and C.

The carrier inventory contains no coordinate for Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient real completeness. Hence the six rows exhaust the local NameCert surface. □

Theorem 2823.3 (Real tightness located trap route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing located trap read of an accepted* $\mathsf{RealTightness}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto S, Q \longmapsto D, I, B, O \longmapsto T, H, C, P, N.$$

The route starts with the real-name source, reads finite stream windows and regular rational approximants, opens the dyadic grid and located interval rows, checks boundedness and located order, and only then exports the trap ledger as the named $\mathsf{RealTightness}^{\uparrow}$ *surface. It exports no Heine–Borel theorem, open-cover compactness principle, trichotomy classifier, countable-choice selector, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an accepted carrier $Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$. The source real-name row is R, and the first observable data are the finite windows S and the regular rational readback Q attached to that source.

The dyadic and located stages follow. The row D supplies endpoints and tolerances, I admits the current located interval, B bounds the trap request, and O compares the readback with the displayed endpoints through the regular Cauchy located-order surface.

The trap ledger T is therefore downstream of all numeric and located rows. The structural rows H, C, P, N transport, replay, record provenance, and name that same route without adding a host-level compactness or completeness object.

Since no carrier coordinate stores Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient real completeness, every accepted located trap read remains inside the displayed finite route. □

Falsifiable prediction: Every accepted RealTightnessUp consumer read of a located dyadic trap factors through RealUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, LocatedRealIntervalUp, LocatedBoundedRealSetUp, RegularCauchyLocatedOrderUp, and the local trap ledger before it can be exported.

Independence witness: real, dyadicratcore, regseqrat, streamname, locatedrealinterval, locatedboundedrealset, regularcauchylocatedorder: the packet is not bijective to these siblings because it binds the real source, finite windows, regular readback, dyadic trap grid, located interval, bounded located-set, located-order comparison, and local NameCert row as one located-tightness route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTightness}^{\uparrow}$ packet is generated from a Real source row, DyadicRatCore grid row, StreamName finite-window row, RegSeqRat readback row, LocatedRealInterval row, LocatedBoundedRealSet row, RegularCauchyLocatedOrder row, located dyadic trap ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$, seed)

Scope closed: Definition 3795.1, Definition 3795.2, and Definition 3795.3 seed the finite located-tightness carrier and trap route for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic trap exposure, located interval admission, bounded-source discipline, regular Cauchy located-order comparison, replay stability, and nonescape for BEDC.Derived.RealTightnessUp.

Chapter 2824

A Concrete Naming Certificate for $\mathsf{SchwarzLemma}^{\uparrow}$

Following Schwarz and Ahlfors, the $\mathsf{SchwarzLemma}^{\uparrow}$ packet names the unit-disk Schwarz lemma as finite BEDC certificate data. It consumes the holomorphic-disk source interface of chapter 68, the coordinate and boundary rows of chapter 377, the complex carrier of chapter 39, and the contraction handoff of chapter 3682. The packet records the pointed disk map, the zero-fixing row, the unit-bound row, the modulus comparison ledger, transport, provenance, and local NameCert rows; it is not a host theorem about arbitrary analytic continuation or a completed complex function space.

Definition 2824.1 (Schwarz lemma carrier)**.** *A* $\mathsf{SchwarzLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H, D, Z, B, M, R, T, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Holomorphic}^{\uparrow}$ *row naming the disk-local complex map over the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
2. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *source and target disk-coordinate row for the displayed input and output points.*
3. *Z is the zero-fixing row recording that the displayed map sends the named disk origin to the named disk origin.*
4. *B is the unit-bound row recording that displayed disk inputs are routed to displayed disk outputs.*
5. *M is the modulus-comparison row recording the Schwarz-lemma read $|f(z)| \leq |z|$ on the displayed disk point and the boundary equality refusal attached to the same row.*
6. *R is the route row through* $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, *and the local modulus ledger.*
7. *T stores componentwise* hsame *and displayed* Cont *transport for the holomorphic map row, disk-coordinate row, zero-fixing row, unit-bound row, modulus-comparison row, and route row.*
8. *P is the* Pkg *provenance over* $\mathsf{SchwarzLemma}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, D, Z, B, M, R, T, P, N$. It has no coordinate for a host identity theorem, Riemann mapping theorem, unlogged analytic continuation, completed function-space equality, or selected boundary extremal map.

Lean4 (definition) `BEDC.Derived.SchwarzLemmaUp.SchwarzLemmaUp`
Lean4 ✓ `BEDC.Derived.SchwarzLemmaUp.TasteGate.SchwarzLemmaTasteGate single carrier alignment`

Theorem 2824.2 (Schwarz lemma NameCert obligations)**.** *For every accepted* $\mathsf{SchwarzLemma}^{\uparrow}$ *carrier*

$$L = (H, D, Z, B, M, R, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the displayed holomorphic disk map, disk coordinates, zero-fixing row, unit-bound row, modulus-comparison row, route row, transport rows, provenance, and local name;*
2. *classifier stability under componentwise* hsame *and displayed* Cont *transport of exactly those rows;*
3. *zero-fixing exactness, requiring the Schwarz-lemma read to pass through Z before any modulus comparison is exported;*
4. *disk-bound exactness, requiring every input-output read to pass through the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate and unit-bound rows before the local name is consumed;*
5. *nonescape for the identity theorem, Riemann mapping, host analytic continuation, completed complex function-space equality, and selected boundary extremal maps.*

Proof. Carrier admission is projection from Definition 3796.1. The tuple exposes the holomorphic source row, disk-coordinate row, zero-fixing row, unit-bound row, modulus-comparison ledger, route, transport, provenance, and local name as one finite packet.

Classifier stability is componentwise. A transported packet may compare the same displayed rows after hsame and Cont replay, but it cannot keep the modulus comparison while dropping the zero-fixing or unit-bound rows.

Zero-fixing and disk-bound exactness follow from the route row R. The modulus comparison row M is read only after H, D, Z, B have been displayed, so a consumer cannot use it as an arbitrary disk-map contraction theorem.

The nonescape clauses are read from the carrier inventory. Since no row stores a host identity theorem, Riemann mapping theorem, analytic-continuation principle, completed function-space equality, or selected boundary extremal map, the local NameCert cannot export one. □

Theorem 2824.3 (Schwarz lemma disk-contraction handoff)**.** *Every disk-contraction consumer of an accepted* $\mathsf{SchwarzLemma}^{\uparrow}$ *packet factors through the ordered route*

$$\mathsf{Holomorphic}^{\uparrow} \longrightarrow \mathsf{Complex}^{\uparrow} \longrightarrow \mathsf{UnitDisk}^{\uparrow} \longrightarrow \mathsf{SchwarzLemma}^{\uparrow} \longrightarrow \mathsf{SchwarzPickDiskContraction}^{\uparrow}.$$

The handoff supplies only the displayed holomorphic disk-map row, zero-fixing row, unit-bound row, modulus-comparison row, transport, provenance, and local NameCert data. It supplies no full complex analytic identity theorem, Riemann mapping theorem, host analytic continuation principle, or ambient completed function-space equality.

Proof. Start with the holomorphic source row H over the complex carrier and the disk-coordinate row D. These rows are the only source of the displayed map and input-output points available to the handoff.

The zero-fixing row Z and unit-bound row B then restrict the read to the pointed disk situation. The modulus-comparison row M records the Schwarz-lemma inequality only for that displayed route.

Finally pass the same rows through R, T, P, N and into the $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ consumer. The downstream contraction packet can read the displayed disk contraction, but no coordinate in L adds identity-theorem reasoning, Riemann mapping, host analytic continuation, or ambient function-space equality. □

Falsifiable prediction: If a purported SchwarzLemmaUp packet exports a modulus comparison without displaying the zero-fixing row and the UnitDisk route, it is not a $\mathsf{SchwarzLemma}^{\uparrow}$ carrier.

Independence witness: holomorphic, complex, unitdisk, schwarz-pick-disk-contraction: the packet is not bijective to any one sibling because it binds the zero-fixing and unit-bound rows to a local modulus-comparison NameCert before the disk-contraction handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwarzLemma}^{\uparrow}$ packet is generated from a holomorphic disk-map row, complex carrier row, UnitDisk coordinate row, zero-fixing row, unit-bound row, modulus-comparison row, route row, transport, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3796.1, Definition 3796.2, and Definition 3796.3 seed the pointed disk-map carrier, NameCert obligations, and disk-contraction handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a full complex analytic identity theorem, Riemann mapping theorem, host analytic continuation principle, Lean verification, or bridge checking for $\mathsf{SchwarzLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, zero-fixing exactness, unit-bound exactness, modulus comparison, route preservation, transport stability, provenance, and local naming for BEDC.Derived.SchwarzLemmaUp.

Chapter 2825

A Concrete Naming Certificate for $\mathsf{CauchyProductCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyProductCompletion}^{\uparrow}$ packet records the constructive product of two completed real carriers by finite Cauchy windows, regular rational readback, and an explicit completion seal. It keeps the product-completion surface separate from ambient Cartesian powersets: each exported product read must still name the stream windows, dyadic observations, product budget, regular-sequence classifier, and real-completion seal that make the read finite.

Definition 2825.1 (CauchyProductCompletion carrier)**.** *A* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are paired* $\mathsf{StreamName}^{\uparrow}$ *windows for the two completion-facing coordinates.*
2. D_0, D_1 *are* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows for the finite rational approximants read from the two streams.*
3. B *is the* $\mathsf{CauchyProduct}^{\uparrow}$ *budget row, recording the common tolerance window used to multiply or pair the displayed approximants.*
4. R_0, R_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, certifying that each coordinate read is regular enough to enter the real-completion route.*
5. L_0, L_1 *are* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility rows, witnessing that the coordinate windows are read only through the completion seal used by the L10 route.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *export seal for the product-completion read.*
7. H *records the* hsame *transports for stream names, dyadic observations, regular classifiers, product budget, limit seals, and real export.*
8. C *records the* Cont *replays that move from stream windows through dyadic observations, regular-sequence readback,* $\mathsf{CauchyProduct}^{\uparrow}$ *budget,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
9. P *is the* Pkg *provenance over* $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}$ *row.*

The accepted carrier contains no ambient product set, no selected representative of a completion quotient, no unrestricted sequence equality, and no completion principle beyond the displayed finite-window route.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionUp.CauchyProductCompletionTasteGate single carrier alignment`

Theorem 2825.2 (CauchyProductCompletion NameCert obligations)**.** *For an accepted packet* $Q = (S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^\uparrow}$ *obligation surface consists exactly of carrier admission, classifier stability, budget factorization, completion-seal nonescape, and ledger exactness for the displayed rows* $S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N$*.*

Proof. Carrier admission reads the two $\mathsf{StreamName}^\uparrow$ windows first, then admits the two $\mathsf{DyadicRatCore}^\uparrow$ observations only as finite approximants attached to those windows. The $\mathsf{RegSeqRat}^\uparrow$ rows certify the regular readback needed by the real-completion route.

Classifier stability is carried by H. Transporting stream names, dyadic observations, regular classifiers, product budget, limit seals, and the real export moves the whole packet together, so no coordinate can be reclassified by an external quotient equality.

Budget factorization is supplied by B and replayed by C. The product-facing read reaches $\mathsf{CauchyProduct}^\uparrow$ only after both coordinate windows and dyadic observations are displayed, and it reaches $\mathsf{Real}^\uparrow$ only after $\mathsf{CauchyLimitSeal}^\uparrow$ compatibility has been exposed.

Completion-seal nonescape follows from the L_0, L_1, E rows. They bind the coordinate reads to the L10 sibling routes of chapter 126, chapter 291, chapter 301, chapter 464, chapter 386, and chapter 38.

Ledger exactness is an inspection of the packet. Every exported read is one of the displayed rows, so the local NameCert has no coordinate from which to recover an ambient product object, a representative-choice operation, unrestricted sequence equality, or a completion principle outside the finite route. □

Theorem 2825.3 (CauchyProductCompletion budget factorization)**.** *Every completion-facing product read exported by* $\mathsf{CauchyProductCompletion}^\uparrow$ *factors through* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{CauchyProduct}^\uparrow$*,* $\mathsf{CauchyLimitSeal}^\uparrow$*, and* $\mathsf{Real}^\uparrow$ *before export. In particular, the product budget is not a free-standing multiplication oracle: it is a finite tolerance-window route whose coordinate streams, dyadic observations, regular classifiers, completion seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows remain visible at the export boundary.*

Proof. Start with a completion-facing product read. By Definition 3797.1, the only carrier rows available before export are $S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N$.

The first stage is the paired stream window. The rows S_0, S_1 force the read to pass through the $\mathsf{StreamName}^\uparrow$ route before any dyadic observation is available, and D_0, D_1 then expose the $\mathsf{DyadicRatCore}^\uparrow$ approximants.

The second stage is regularization and budget. The rows R_0, R_1 record the $\mathsf{RegSeqRat}^\uparrow$ classifiers, while B records the $\mathsf{CauchyProduct}^\uparrow$ tolerance budget that combines the coordinate approximants.

The final stage is sealing and export. The rows L_0, L_1 expose $\mathsf{CauchyLimitSeal}^\uparrow$ compatibility before E exports a $\mathsf{Real}^\uparrow$ product-completion read. Since H, C, P, N are the only structural rows, the export cannot bypass this route through an ambient product, selected completion representative, unrestricted sequence equality, or hidden real-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletion}^{\uparrow}$ packet is generated from paired StreamName windows, DyadicRatCore observations, RegSeqRat classifiers, a CauchyProduct budget row, CauchyLimitSeal compatibility rows, a Real export seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3797.1, Definition 3797.2, and Definition 3797.3 bind the finite product-completion seed used by the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed product universal property, quotient equality of completed reals, arbitrary sequence multiplication, compactness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, classifier stability, budget factorization, completion-seal nonescape, and ledger exactness for BEDC.Derived.CauchyProductCompletionUp.

Chapter 2826

A Concrete Naming Certificate for $\mathsf{CantorSpaceCompactness}^{\uparrow}$

Following Brouwer and Bishop, the $\mathsf{CantorSpaceCompactness}^{\uparrow}$ packet records the constructive compactness route for Boolean streams as a finite NameCert surface adjacent to $\mathsf{CantorSpace}^{\uparrow}$ in chapter 1475. It reads finite prefix data through $\mathsf{FanTheorem}^{\uparrow}$, obstruction rows through $\mathsf{KleeneTree}^{\uparrow}$, and observation windows through $\mathsf{StreamName}^{\uparrow}$; it does not import open-cover compactness, countable choice, quotient equality of infinite streams, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2826.1 (CantorSpaceCompactness carrier)**.** *A* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, F, T, S, B, H, R, Q, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix source row, including the Boolean prefix observations and cylinder reads named in Definition 1475.1;*
2. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *row, recording the finite decidable bar, its inspected prefix depth, and the finite branch-window witness;*
3. *T is the* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row, used only to record that a displayed infinite-branch escape is absent from the finite fan route;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row that schedules the finite prefix requests consumed by C and F;*
5. *B is the prefix-cover row, listing the finitely many Boolean cylinders whose union covers the inspected prefix surface;*
6. *H records componentwise* hsame *transport for the Cantor-space, fan, obstruction, stream, prefix-cover, continuation, provenance, and local naming rows;*
7. *R records displayed* Cont *replay from a finite prefix request through the fan-depth row and the prefix-cover row;*
8. *Q is the refusal row excluding open-cover compactness, countable choice, quotient equality of infinite streams, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
9. *P is the* Pkg *provenance over* $\mathsf{CantorSpaceCompactness}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *prefix covers,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$ *row naming the displayed finite compactness packet and its permitted consumer reads.*

The classifier compares accepted packets only through the displayed Cantor prefix, fan-depth, obstruction, stream-window, and prefix-cover rows, transported by H and replayed by R. No classifier coordinate compares ambient infinite streams or completed function-space objects.

Lean4 ✓ `BEDC.Derived.CantorSpaceCompactnessUp.CantorSpaceCompactnessTasteGate single carrier alignment`

Theorem 2826.2 (CantorSpaceCompactness NameCert obligations)**.** *For every accepted* $\mathsf{CantorSpaceCompactness}$ *carrier $K = (C, F, T, S, B, H, R, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{CantorSpace}^{\uparrow}$ *source,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row,* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row,* $\mathsf{StreamName}^{\uparrow}$ *schedule, prefix-cover ledger, transport, replay, refusal, provenance, and local naming row;*
2. *prefix-cover exactness, requiring every accepted compactness read to use the same finite Boolean cylinders listed in B and the same inspected prefix schedule in S;*
3. *fan-depth stability, requiring the finite bar depth in F to bound the cylinders consumed by a downstream compactness read;*
4. *obstruction nonescape, requiring T to remain a finite refusal row rather than a source of an infinite branch;*
5. *ledger nonescape, excluding open-cover compactness, countable choice, quotient equality of infinite streams, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection of Definition 3798.1. The packet exposes exactly $C, F, T, S, B, H, R, Q, P, N$ before any compactness-facing read is permitted.

Prefix-cover exactness reads B together with the $\mathsf{StreamName}^{\uparrow}$ schedule S. Fan-depth stability reads F and forces the listed cylinders to sit below the inspected finite bar depth.

The obstruction row T is then read only as a finite nonescape boundary. Transport by H and replay by R preserve the same displayed route, while Q, P, N record refusal, provenance, and local naming. Since none of these rows contains open-cover compactness, countable choice, quotient equality, or ambient real completeness, the listed obligations exhaust the local surface. □

Theorem 2826.3 (CantorSpaceCompactness finite prefix subcover)**.** *Let $K = (C, F, T, S, B, H, R, Q, P, N)$ be an accepted* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier. Every compactness-facing read that asks for a cover of a displayed* $\mathsf{CantorSpace}^{\uparrow}$ *prefix is answered by the finite prefix-cover row B after the fan-depth row F has fixed the inspected depth and the* $\mathsf{StreamName}^{\uparrow}$ *schedule S has fixed the visible positions. The subcover is therefore a finite list of Boolean cylinders over the displayed prefix surface, not an open-cover compactness principle and not a choice-selected family of cylinders.*

Proof. Start with the $\mathsf{CantorSpace}^{\uparrow}$ row C. By Definition 1475.5, every cylinder read of C factors through finite $\mathsf{StreamName}^{\uparrow}$ windows, finite Boolean observations, and local NameCert rows.

Read the fan row F. It supplies a finite decidable bar depth and the inspected branch-window witness, so the compactness-facing request is reduced to the cylinders visible below that depth.

Now use the prefix-cover ledger B. The row lists the finitely many Boolean cylinders that cover the inspected prefix surface, and the replay row R routes the consumer through exactly that

ledger. The refusal row Q excludes open-cover compactness, countable choice, quotient stream equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the exported subcover. □

Theorem 2826.4 (CantorSpaceCompactness nonescape)**.** *For every accepted* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier* $K = (C, F, T, S, B, H, R, Q, P, N)$*, any downstream consumer of compactness data remains inside the route*

$$C, S \longmapsto F \longmapsto B \longmapsto T \longmapsto H, R, Q, P, N.$$

The route exports only finite Boolean prefix covers for the displayed Cantor-space packet. It exports no open-cover compactness theorem, no countable choice principle, no quotient equality of streams, no selected infinite branch through a $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 3798.3. A compactness-facing read first consumes the displayed Cantor prefix row C and the stream schedule S, then reads the fan-depth row F and the finite prefix-cover ledger B.

The obstruction row T is downstream of that finite route. It records that the displayed route has no accepted infinite-branch escape; it does not provide a selected branch or a host-level tree path.

Finally inspect H, R, Q, P, N. These rows transport, replay, refuse, source, and name only the displayed finite packet. Therefore a consumer has no row from which to recover open-cover compactness, countable choice, quotient stream equality, an infinite branch selector, or ambient real completeness. □

Falsifiable prediction: Every $\mathsf{CantorSpaceCompactness}^{\uparrow}$ compactness read factors through a displayed $\mathsf{CantorSpace}^{\uparrow}$ prefix, a finite $\mathsf{FanTheorem}^{\uparrow}$ bar-depth row, and a finite prefix-cover ledger before any consumer sees a compactness-shaped output.

Independence witness: cantorspace, fantheorem, kleenetree, streamname: $\mathsf{CantorSpaceCompactness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Cantor prefix data, fan-depth coverage, obstruction nonescape, stream scheduling, prefix-cover rows, transport, continuation replay, package provenance, refusal, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3798.1, Definition 3798.2, Definition 3798.3, and Definition 3798.4 seed the constructive finite-prefix compactness packet over $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, prefix covers, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, countable choice, quotient equality of streams, selected infinite branches, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix subcover exactness, fan-depth stability, obstruction nonescape, and local NameCert replay for BEDC.Derived.CantorSpaceCompactnessUp.

Constructive story: A $\mathsf{CantorSpaceCompactness}^{\uparrow}$ packet is generated from CantorSpace finite-prefix rows, FanTheorem finite-bar rows, KleeneTree obstruction rows, StreamName schedules, prefix-cover ledgers, componentwise hsame transport, displayed Cont replay, refusal, Pkg provenance, and a local NameCert row.

Chapter 2827

A Concrete Naming Certificate for CauchySchlomilchCondensation$^{\uparrow}$

Following Knopp and Hardy on condensation tests for positive series, the CauchySchlomilchCondensation$^{\uparrow}$ packet records the finite comparison route between a monotone nonnegative series and its Schlomilch-block condensation. It sits beside the dyadic block discipline of chapter 3598, the integral comparison surface of chapter 2259, and the real series endpoint route of chapter 1581. The packet names block maps, block masses, monotonicity windows, tail comparisons, and endpoint seals as finite NameCert data; it does not assert an unrestricted host convergence theorem or choose an infinite sum.

Definition 2827.1 (CauchySchlomilchCondensation carrier)**.** *A* CauchySchlomilchCondensation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, M, B, Q, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the nonnegative real-series row, including the inspected finite coefficient window and the displayed monotonicity direction.*
2. *M is the monotone tail row certifying that later blocks may be bounded by their boundary terms.*
3. *B is the Schlomilch block-index row, listing the increasing block cut points and their finite gap sizes.*
4. *Q is the block-mass comparison row, recording the finite inequalities between original tails and condensed block terms.*
5. *T is the tail-transfer row that routes convergence or divergence reads through the displayed block comparison.*
6. *E is the terminal* Real$^{\uparrow}$ *or* RealSeries$^{\uparrow}$ *endpoint seal reached only after A, M, B, Q, T have been displayed.*
7. *H records componentwise* hsame *transport for series rows, monotonicity rows, block ledgers, tail transfers, endpoint seals, provenance, and local naming.*
8. *C records displayed* Cont *replay from a tail request through the monotone series row, Schlomilch block ledger, comparison row, tail transfer, and endpoint seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{CauchyIntegralTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchlomilchCondensation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a nonmonotone rearrangement, an unstated analytic integral, a selected infinite sum, quotient real equality, or a convergence claim outside the displayed Schlomilch blocks.

Lean4 ✓ `BEDC.Derived.CauchySchlomilchCondensationUp.CauchySchlomilchCondensationTasteGate`
`single carrier alignment`

Theorem 2827.2 (CauchySchlomilchCondensation NameCert obligations)**.** *For every accepted* $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ *carrier* $S = (A, M, B, Q, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySchlomilchCondensa}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nonnegative series row, monotone tail row, block-index ledger, block-mass comparison, tail-transfer row, endpoint seal, transport, replay, provenance, and local naming;*
2. *monotone tail discipline, requiring M before any block comparison row can be consumed;*
3. *Schlomilch block exactness, requiring B and Q to name the same finite cut points and gap sizes;*
4. *condensation handoff, requiring every convergence-facing or divergence-facing tail read to pass through T before reaching E;*
5. *sibling compatibility, allowing dyadic Cauchy condensation and integral-test rows to be read only as named comparison siblings, not as hidden analytic premises;*
6. *ledger non-escape, excluding nonmonotone rearrangement, selected infinite sums, detached integral estimates, quotient real equality, and ambient completeness.*

No condensation obligation is stored outside these rows.

Proof. Project the carrier of Definition 3799.1. The rows $A, M, B, Q, T, E, H, C, P, N$ are the only displayed coordinates of the accepted packet.

Read the tail discipline first. The monotone row M governs the series row A before the Schlomilch cut points B and block comparisons Q are exposed, so the block ledger cannot be detached from the source tail.

Next apply the tail-transfer row T. It is the unique route from the displayed block comparison to the endpoint seal E, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic or quotient coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: A candidate packet with Schlomilch cut points but no monotone tail row is rejected by the $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ carrier.

Independence witness: condensationtailselector, cauchy-integral-test, realseries, regseqrat, streamname: none is bijective to $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ because this packet jointly carries monotone source-series rows, Schlomilch block cut points, block-mass comparisons, tail-transfer rows, endpoint seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchlomilchCondensation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ packet is generated from a monotone nonnegative series row, Schlomilch block-index ledger, block-mass comparison, tail-transfer row, endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchySchlomilchCondensation}\uparrow}$, seed)

Scope closed: Definition 3799.1 and Definition 3799.2 seed the finite Schlomilch condensation packet over monotone tails, block ledgers, comparison rows, and endpoint seals.

Formal verification: FormalStatus(NameCert$_{\text{CauchySchlomilchCondensation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, nonmonotone rearrangement, selected infinite sums, host integral estimates, quotient real equality, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, monotone tail discipline, Schlomilch block exactness, condensation handoff, sibling compatibility, and ledger non-escape for BEDC.Derived.CauchySchlomilchCondensationUp.

Chapter 2828

A Concrete Naming Certificate for $\mathsf{FiniteOscillationPartition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteOscillationPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed oscillation window for each cell. It sits after the closed bounded interval surface of chapter 1591, the located interval source of chapter 1709, the oscillation-modulus ledger of chapter 2224, the real-name seal of chapter 38, the regular rational sequence readback of chapter 301, and the stream source of chapter 126. The packet gives later uniform-continuity, Darboux, Riemann, and regulated-integral certificates a finite-means carrier for oscillation-controlled partitions without invoking open-cover compactness, quotient equality of real functions, selected suprema, or ambient real completeness.

Definition 2828.1 (Finite oscillation partition carrier)**.** *A* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, M, A, B, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded and located* $\mathsf{Real}^{\uparrow}$ *interval row, read through the endpoint, inclusion, and apartness surfaces of* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *and* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$*.*

2. *M is the finite mesh row, listing ordered partition points, adjacent cell rows, endpoint containment, and a common finite index ledger.*

3. *A is the oscillation-modulus row, recording the tolerance request and the finite cellwise radius or mesh constraint supplied by* $\mathsf{OscillationModulus}^{\uparrow}$*.*

4. *B is the cell-bound row, assigning to each cell a displayed upper and lower real-name readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is used.*

5. *R is the refinement-readback row, recording how a later common refinement or integration consumer reads I, M, A, B without changing the accepted classifier.*

6. *H records componentwise* hsame *transport for I, M, A, B, R and for the continuation, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from an oscillation-window request through*

$$I, M, A, B, R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row.*

The carrier contains no open-cover compactness witness, no quotient equality of real-valued functions, no selected supremum or infimum, no unbounded partition search, and no ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.FiniteOscillationPartitionUp.FiniteOscillationPartitionTasteGate single carrier alignment`

Theorem 2828.2 (Finite oscillation partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *carrier $O = (I, M, A, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, interval-source exposure through I, mesh exactness through M, oscillation-window accounting through A, cell-bound readback through B, refinement-readback stability through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3800.1. The interval, mesh, oscillation, cell-bound, and refinement rows are the analytic part of the packet, while H, C, P, N are the structural rows that transport, replay, source, and name it.

The interval and mesh obligations are finite. The row I supplies the closed bounded located interval boundary, and M lists the partition cells and endpoint containments over that same boundary.

The oscillation and bound obligations are also finite. The row A records the tolerance and the cellwise oscillation constraint, while B exposes upper and lower readbacks through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Refinement stability is read from R and transported by H. A downstream consumer may refine the displayed cells only by replaying the same interval, mesh, oscillation, and bound rows through C with provenance P and local name N.

Since the carrier has no coordinate for open-cover compactness, quotient function equality, selected extrema, unbounded search, or ambient completeness, no accepted NameCert obligation can project one of those objects. □

Lean4 ✓ `BEDC.Derived.FiniteOscillationPartitionUp.FiniteOscillationPartition namecert obligations`

Theorem 2828.3 (Finite oscillation partition uniform window)**.** *Every uniform-continuity or integration-facing consumer of an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{LocatedRealIntervalPartition}^{\uparrow},\ \mathsf{OscillationModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto$$

In particular, the consumer reads a displayed interval source, finite mesh, oscillation tolerance ledger, cell-bound readback, and refinement route before any terminal real seal is used.

Proof. Apply the NameCert obligations of Definition 3800.2. They require every accepted consumer to open I, M, A, B, R before the structural rows are available.

The uniform window is determined by A over the cells of M. Thus a uniform-continuity consumer obtains a tolerance-indexed cell ledger rather than an open-cover argument.

An integration-facing consumer reads the same route through B and R. The cell-bound readbacks expose finite upper and lower rows through stream and regular rational names, while the refinement row records how common refinements preserve the displayed classifier.

Therefore the accepted packet exports a finite oscillation-controlled partition window, and no hidden supremum, quotient-function comparison, open-cover compactness witness, or ambient completeness theorem is available to the consumer. □

Theorem 2828.4 (Finite oscillation partition uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet before a* $\mathsf{UniformModulus}^{\uparrow}$ *handoff must pass through exactly the displayed interval source, finite mesh, oscillation-window row, cell-bound readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row. The handoff exports a finite oscillation route to a uniform-modulus consumer, not open-cover compactness, selected extrema, quotient function equality, unbounded partition search, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3800.1. Its analytic rows are the interval source I, finite mesh M, oscillation-window row A, cell-bound readback B, and refinement-readback row R; its structural rows are H, C, P, N.

Apply Definition 3800.2. The local certificate forces any compact-uniform-continuity consumer to expose carrier admission, interval-source exposure, mesh exactness, oscillation accounting, cell-bound readback, refinement stability, transport, replay, provenance, and local naming before treating the packet as usable modulus data.

Apply Definition 3800.3. The uniform window is read only after $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal have been displayed over the same finite partition rows.

Thus the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the displayed finite oscillation handoff. Since no carrier coordinate stores open-cover compactness, selected extrema, quotient function equality, unbounded partition search, or ambient completeness, none of those principles can be exported by the handoff. □

Theorem 2828.5 (Compact-uniform finite oscillation modulus)**.** *Every compact-uniform-continuity or integration-facing consumer that receives an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet obtains a finite oscillation modulus only through the displayed interval source, finite mesh, oscillation-window row, cell-bound readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row. The received modulus is a finite packet read, not an open-cover compactness theorem, selected-extrema argument, quotient function equality, unbounded partition search, or ambient completeness principle.*

Proof. Project the carrier of Definition 3800.1. The only analytic rows available to a compact-uniform or integration-facing consumer are the interval source I, finite mesh M, oscillation-window row A, cell-bound readback B, and refinement route R.

Apply Definition 3800.2. It forces the consumer to expose interval-source data, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, componentwise transport, replay, provenance, and local naming before the packet can be consumed as a modulus.

Use Definition 3800.3 to identify the uniform window read. The window factors through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal, so the compact-uniform reader receives a finite tolerance-indexed route rather than a host-level compactness principle.

Finally inspect the absent coordinates of the carrier. It contains no open-cover compactness witness, selected extrema, quotient comparison of functions, unbounded partition search, or ambient completeness row; hence none of those principles can be projected from the finite oscillation modulus. □

Theorem 2828.6 (Finite oscillation partition step-function handoff)**.** *Let $O = (I, M, A, B, R, H, C, P, N)$ be an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet whose cell-bound row B chooses displayed lower and upper real-name reads for every cell of the finite mesh M. Then every* $\mathsf{StepFunction}^{\uparrow}$ *or* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer that reads the packet receives a finite approximation route through*

$$I, M, A, B, R,\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ H, C, P, N.$$

The route first exposes the interval source, finite mesh, oscillation modulus, stream windows, regular rational readbacks, dyadic tolerance rows, and terminal real seal; it then repacks the displayed cellwise constant lower and upper reads as $\mathsf{StepFunction}^{\uparrow}$ *rows and as the finite schedule visible to* $\mathsf{RegulatedFunction}^{\uparrow}$*. No open-cover compactness, selected extrema, quotient functions, unbounded partition search, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle is introduced by this handoff.*

Proof. Project the carrier of Definition 3800.1. The interval source I, mesh M, oscillation row A, cell-bound row B, and refinement row R are already finite rows of the accepted packet, while H, C, P, N provide componentwise transport, displayed replay, provenance, and the local naming certificate.

Apply Definition 3800.2. The NameCert obligations force every consumer to expose the interval source, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, transport, replay, provenance, and local name before the packet can be read as approximation data.

Use Definition 3800.5 to expose the uniform cell window. For each cell of M, the selected lower and upper reads in B pass through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, are calibrated by the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and reach the terminal $\mathsf{Real}^{\uparrow}$ seal only along that displayed finite route.

Now repack each displayed cellwise constant read as a $\mathsf{StepFunction}^{\uparrow}$ row over the same interval source and finite mesh. Reading the finitely many rows in the order recorded by R gives the regulated-function finite schedule: the $\mathsf{RegulatedFunction}^{\uparrow}$ consumer sees only the interval, mesh, step rows, oscillation window, readbacks, transport, replay, provenance, and local name.

Finally inspect the absent coordinates of the carrier. Since the packet has no coordinate for open-cover compactness, selected extrema, quotient functions, unbounded partition search, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the step-function and regulated-function consumers cannot acquire any of those principles through this handoff. □

Falsifiable prediction: Any accepted $\mathsf{FiniteOscillationPartition}^{\uparrow}$ consumer that claims a uniform oscillation window while omitting one of the interval source, finite mesh, oscillation-modulus, cell-bound, refinement, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: closedboundedinterval, located-real-interval-partition, oscillationmodulus, real, regseqrat, streamname: none is bijective with $\mathsf{FiniteOscillationPartition}^{\uparrow}$ because this packet jointly carries interval source, finite mesh, oscillation tolerance, cell-bound readback, refinement stability, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOscillationPartition}^{\uparrow}$ packet is generated from a closed bounded located interval row, a finite mesh row, an oscillation-modulus row, cell-bound readbacks through StreamName and RegSeqRat, a refinement-readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3800.1, Definition 3800.2, Definition 3800.3, and Definition 3800.6 seed the finite oscillation-controlled partition route for uniform-continuity, step-function, regulated-function, and integration-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality of real-valued functions, selected suprema, unbounded partition search, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.FiniteOscillationPartitionUp.

Chapter 2829

A Concrete Naming Certificate for $\mathsf{TriggerBlockerDuality}^{\uparrow}$

The $\mathsf{TriggerBlockerDuality}^{\uparrow}$ packet realizes the blocker-complex reading of Definition 9462.1547, Definition 9462.1548, Definition 9462.1550, and Definition 9462.1551 as finite BEDC NameCert data. It consumes the six-boundary coverage packet of chapter 3470, the finite support algebra of chapter 3478, the Boolean trigger lattice of chapter 3704, the Alexander-dual trigger algebra of chapter 3555, and the DNA boundary-tile source of chapter 3393. The packet names trigger-blocker duality by finite blocker-complex, maximal-blocker, intersection, nerve, transport, provenance, and local NameCert rows; it is not a biological stochastic model, a hidden codon semantics, an unbounded trigger search, or an external homology oracle.

Definition 2829.1 (Trigger-blocker duality carrier)**.** *A* $\mathsf{TriggerBlockerDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, L, B, I, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row exposing the finite boundary-trigger coordinates and coverage ranks.*
2. *A is the* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *row carrying the finite trigger-support closure and blocker threshold data.*
3. *L is the* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *row recording meet, join, rank, and observed trigger-table reads.*
4. *B is the blocker-complex row, displaying the four maximal blockers obtained by omitting one trigger coordinate and the non-trigger face predicate of Definition 9462.1547.*
5. *I is the intersection row, recording the sizes $14, 6, 2, 0$ for one-, two-, three-, and four-fold omitted-coordinate intersections.*
6. *V is the nerve row, recording the four vertices, six edges, four triangular faces, and absent three-simplex of the blocker nerve.*
7. *H records componentwise* hsame *transport for S, A, L, B, I, V and for the continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from trigger coverage through blocker readback and nerve readback.*

9. *P is the* Pkg *provenance over* TriggerBlockerDuality$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$ BooleanTriggerLattice$^{\uparrow}$, AlexanderDualTriggerAlgebra$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, *the source blocker-complex vision,* BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{TriggerBlockerDuality}^{\uparrow}}$ *row.*

The carrier contains no hidden biological semantics, no stochastic evolutionary model, no quotient codon space, no unbounded trigger enumeration, and no external homology oracle.

Lean4 ✓ `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityTasteGate single carrier alignment`

Theorem 2829.2 (Trigger-blocker duality NameCert obligations)**.** *For every accepted* TriggerBlockerDuality$^{\uparrow}$ *carrier $D = (S, A, L, B, I, V, H, C, P, N)$, the local* NameCert$_{\text{TriggerBlockerDuality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, coverage exposure through S, trigger-algebra exposure through A, Boolean lattice exposure through L, blocker-complex exactness through B, maximal-blocker intersection accounting through I, blocker-nerve readback through V, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityNameCertObligations`

Proof. Project the carrier of Definition 3801.1. The rows S, A, L expose the trigger side, the rows B, I, V expose the blocker and nerve side, and H, C, P, N transport, replay, source, and name the packet.

The trigger side is finite because S displays the boundary-tile coverage coordinates, A displays the support-closure algebra, and L displays Boolean meet, join, and rank rows over the same finite support surface.

The blocker side is finite because B lists the four maximal omitted-coordinate blockers, I records the displayed intersection sizes, and V records the nerve readback. These are exactly the rows needed by Definition 9462.1551.

Transport and replay preserve the same rows componentwise. Since the carrier has no coordinate for hidden biological meaning, stochastic evolution, quotient codon semantics, unbounded search, or an external homology oracle, no accepted NameCert obligation can project one of those objects. □

Theorem 2829.3 (Trigger-blocker duality finite nerve readback)**.** *Every accepted* TriggerBlockerDuality$^{\uparrow}$ *read of the blocker nerve factors through the finite route*

SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, BooleanTriggerLattice$^{\uparrow}$, AlexanderDualTriggerAlgebra$^{\uparrow}$,

Removing one maximal-blocker row changes the blocker nerve readback because V is computed from the four displayed blocker rows and their intersection ledger I.

Proof. Apply the obligation surface of Definition 3801.2. It forces every accepted nerve read to expose the trigger-side coverage, algebra, and Boolean lattice rows before the blocker rows are available.

The nerve row V is tied to B and I. The four vertices are the four maximal-blocker rows, the six edges and four triangular faces are read from the nonempty two- and three-fold intersections, and the absent three-simplex is read from the empty four-fold intersection.

If one maximal-blocker row is removed, the vertex set of V changes and the intersection ledger no longer has the same four omitted-coordinate coordinates. Thus the accepted blocker nerve readback changes before any downstream trigger-spectrum consumer can use the packet. □

Falsifiable prediction: Removing one displayed maximal-blocker row from an accepted $\mathsf{TriggerBlockerDuality}^{\uparrow}$ carrier changes the blocker nerve readback through V before any downstream trigger-spectrum consumer can preserve the certificate.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, boolean-trigger-lattice, alexander-dual-trigger-algebra, dna-window-six-boundary-tile: none is bijective with $\mathsf{TriggerBlockerDuality}^{\uparrow}$ because this packet jointly carries blocker-complex rows, maximal-blocker rows, intersection rows, nerve rows, transport, replay, provenance, and local naming.

The Lean follow-up for this chapter must provide `Nontrivial TriggerBlockerDualityUp` and `FieldFaithful TriggerBlockerDualityUp` instances in `BEDC.Derived.TriggerBlockerDualityUp` so that $S, A, L, B, I, V, H, C, P, N$ remain separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}$.

Constructive story: A $\mathsf{TriggerBlockerDuality}^{\uparrow}$ packet is generated from six-boundary coverage rows, trigger-algebra rows, Boolean trigger-lattice rows, blocker-complex rows, maximal-blocker intersection rows, blocker-nerve rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3801.1, Definition 3801.2, and Definition 3801.3 seed the finite blocker-complex, maximal-blocker, intersection, and nerve packet consumed by downstream trigger-spectrum chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close biological stochastic semantics, quotient codon semantics, unrestricted trigger search, external homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coverage exposure, trigger-algebra exposure, Boolean lattice exposure, blocker-complex exactness, intersection accounting, nerve readback, Nontrivial TriggerBlockerDualityUp, and FieldFaithful TriggerBlockerDualityUp for BEDC.Derived.TriggerBlockerDualityUp.

Chapter 2830

A Concrete Naming Certificate for $\mathsf{FiniteRelation}^{\uparrow}$

Following Tarski-style relation algebra and finite relational semantics, the $\mathsf{FiniteRelation}^{\uparrow}$ packet records a finite binary relation as displayed BEDC pair rows rather than as a host predicate, quotient graph, or finite map with discarded duplicate rows.

2830.1 Finite-relation source

Definition 2830.1 (FiniteRelationUp carrier)**.** *A* $\mathsf{FiniteRelation}^{\uparrow}$ *carrier from a displayed left surface to a displayed right surface is a finite* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *pair rows. Each row has first component a left* BHist *witness and second component a right* BHist *witness, with both components read through their local classifiers and package provenance.*

Membership in the relation is exact display membership in the spine: a pair (a, b) *is accepted precisely when some product-coded row in the spine has left component classified with* a *and right component classified with* b*. The domain support is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration obtained by projecting the left components of all rows, and the range support is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration obtained by projecting the right components. Repeated rows and repeated component classes remain visible ledger entries; unlike* $\mathsf{FiniteMap}^{\uparrow}$ *lookup in chapter 352, no first-hit rule suppresses later rows and no functionality condition is imposed.*

Lean4 ✓ `BEDC.Derived.FiniteRelationUp.FiniteRelationTasteGate single carrier alignment`

Theorem 2830.2 (FiniteRelationUp NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$ *obligation surface consists of seven rows:*

1. *the finite* $\mathsf{List}^{\uparrow}$ *carrier of* $\mathsf{Prod}^{\uparrow}$ *pair rows from Definition 3802.1;*
2. *membership exactness for displayed pair membership;*
3. *domain and range support projections through* $\mathsf{FinSet}^{\uparrow}$*;*
4. *classifier transport for both components and for the pair-row membership classifier;*
5. *replay through the displayed* Cont *route;*
6. *source provenance through the recorded* Pkg *and* BHist *rows;*
7. *local naming of the finite relation packet as a* NameCert *row.*

Together these rows expose a non-functional finite relation surface whose consumers may cite chapter 36, chapter 34, chapter 32, chapter 33, chapter 112, and chapter 352 without importing host relation equality.

Proof. Project the displayed relation spine. The carrier row is the finite $\mathsf{List}^{\uparrow}$ object, while each member row is a $\mathsf{Prod}^{\uparrow}$ pair whose two components retain their component classifiers and provenance.

Membership exactness is immediate from spine membership: a consumer can accept a pair only by pointing to a displayed row and transporting both components along their classifiers. Domain and range support are obtained by component projection followed by the finite-set enumeration interface, so repeated component classes remain ledger-visible.

Replay and provenance are inherited rowwise. The Cont route replays the same inspected product row, the Pkg row names the source packet, and the BHist entries identify the displayed component histories. These rows give the local naming certificate without requiring functionality or quotienting duplicate rows. □

Theorem 2830.3 (FiniteRelationUp image and preimage route)**.** *Every public image or preimage read of an accepted* $\mathsf{FiniteRelation}^{\uparrow}$ *packet factors only through the displayed finite-data rows named in Definition 3802.2. An image read starts from a displayed left witness, scans the* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *rows, tests component membership through the left classifier, and exports the corresponding right witnesses through a* $\mathsf{FinSet}^{\uparrow}$ *range-support row. A preimage read is the symmetric route through the right classifier and left support row.*

The route may use $\mathsf{Bool}^{\uparrow}$ *as a finite discriminator for visible classifier outcomes and* $\mathsf{Option}^{\uparrow}$ *as an optional read witness for single-row consumers, but it never turns the relation into a* $\mathsf{FiniteMap}^{\uparrow}$ *first-hit lookup. All exported facts pass through* $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$ *rows.*

Proof. For an image read, inspect the relation spine in order. Each accepted row is a displayed product row, so the left component can be compared with the queried witness through the component classifier and transported along hsame when the classifier accepts.

The accepted right components are collected as a finite support enumeration. The same construction with left and right interchanged gives the preimage route. The boolean and optional witnesses are only visible finite-data control rows for the scan; they do not choose a hidden branch or collapse the relation to a function.

Thus every consumer-visible output is a projection of displayed pair rows, replayed by Cont and named by Pkg provenance under the relation certificate. No image or preimage read can bypass the finite spine or import host relational semantics. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$, seed)

Scope closed: The chapter fixes the finite relation carrier, membership exactness surface, and image/preimage dependency route as paper-side obligations.

Constructive story: A finite relation is built by displaying a finite list spine of product-coded rows and projecting membership, domain support, and range support from that spine.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: No Lean target, relational algebra operations, transitive closure, graph reachability, or category hom-relation bridge is claimed here.

Upgrade path: Formalize the finite spine carrier and membership classifier as *BEDC.Derived.FiniteRelationUp*, then discharge the NameCert obligation surface.

Chapter 2831

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy sequences, the $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ packet names the finite tail-envelope data needed when a regular rational sequence is handed to the $\mathsf{Real}^{\uparrow}$ completion route. It sits between the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the explicit modulus surface of chapter 3768, and the terminal real-name seal of chapter 38. The packet is finite tail-control data for the L10 regular-Cauchy route; it is not a selected real limit, quotient sequence, countable-choice schedule, host equality comparison, or ambient completeness theorem.

Definition 2831.1 (Regular Cauchy tail envelope carrier)**.** *A* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, U, D, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row whose rational entries are read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *route.*
2. *U is the unary tail-bound row. For each precision request it records a finite tail index beyond which the source row is inspected.*
3. *D is the dyadic tolerance ledger, translating the precision request and the chosen tail bound into* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budgets.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected by the tail bound.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the inspected window as regular rational sequence data.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after S, U, D, W, R have been replayed.*
7. *H records componentwise* hsame *transport for the source row, tail bound, dyadic ledger, window, readback, seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a precision request through $S, U, D, W, R, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{RegularModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the source row, unary tail bound, dyadic ledger, StreamName window, RegSeqRat readback, and Real seal all describe the same displayed regular Cauchy route. No carrier coordinate stores a quotient representative, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEnvelopeUp.RegularCauchyTailEnvelopeTasteGate single carrier alignment`

Theorem 2831.2 (Regular Cauchy tail envelope NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier* $E = (S, U, D, W, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$ *surface consists exactly of source regular Cauchy admission, unary tail-bound display, dyadic tolerance accounting, finite StreamName window selection, RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows* $S, U, D, W, R, Q, H, C, P, N$*.*

Proof. Project the carrier of Definition 3803.1. The source row S is the only regular Cauchy input, and U is the only tail-bound coordinate attached to it.

The ledger D translates the requested precision and the displayed tail bound into dyadic comparison data. The window row W and readback row R then expose exactly the finite observations consumed by the StreamName and RegSeqRat surfaces.

The seal row Q is downstream of those rows. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, and the excluded quotient, selected-limit, choice, host-equality, and completeness data have no carrier coordinate. □

Theorem 2831.3 (Regular Cauchy tail envelope handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier must read the finite route*

$$S, U, D, W \longmapsto R \longmapsto Q \longmapsto H, C, P, N$$

before consuming the regular rational row or terminal real-name seal. Thus the tail envelope supplies named finite tail control for the L10 route rather than an informal appeal to a hidden limit, quotient stream, countable-choice schedule, host equality, or ambient completeness coordinate.

Proof. Start with a consumer request for regular rational or real-name data. The packet exposes the source row only as S, so the request must first read the tail bound U and the dyadic ledger D.

The finite window W is selected from that bound and ledger. The readback row R repacks the same inspected data as $\mathsf{RegSeqRat}^{\uparrow}$ information, so the consumer reaches the regular rational surface only through displayed finite observations.

The terminal seal Q is reached after this readback. The rows H, C, P, N then transport, replay, source, and name the same route; no excluded analytic coordinate is present, so every handoff remains finite BEDC data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ packet is generated from a source regular Cauchy row, a unary tail-bound row, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 3803.1, Definition 3803.2, and Definition 3803.3 seed the finite tail-envelope surface for the L10 RegSeqRat-to-Real route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient sequences, countable choice, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-bound exactness, dyadic tolerance accounting, StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailEnvelopeUp.

Chapter 2832

A Concrete Naming Certificate for $\mathsf{VariationNorm}^{\uparrow}$

Following Jordan and the bounded-variation treatment in Rudin, the $\mathsf{VariationNorm}^{\uparrow}$ packet names the finite norm row obtained from displayed partition variation data. It consumes the $\mathsf{BoundedVariation}^{\uparrow}$ partition ledger of chapter 398, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the regular completion route through chapter 301, chapter 126, and chapter 291. The packet records a finite variation-norm observation over displayed partitions; it does not introduce a completed Banach space of bounded-variation functions, a host supremum over all partitions, or quotient equality of functions.

Definition 2832.1 (VariationNorm carrier)**.** *A* $\mathsf{VariationNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, R, Q, S, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed interval row, finite partition ledger, endpoint-name rows, dyadic edge observations, finite variation sum, and common-refinement rows.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and value seal read from the same partition ledger.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name row used when the endpoint values are consumed through rational readback.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row from which the regular-name reads are inspected.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-difference and finite-sum ledger.*
6. *A is the displayed variation-norm row, recording the finite base-value read together with the finite partition-variation sum read from B.*
7. *H records componentwise* hsame *transport for the bounded-variation source, Real seal, regular-name row, stream window, dyadic ledger, norm row, continuation, provenance, and local name.*
8. *C records the displayed* Cont *replay route from a variation-norm request through $B, R, Q, S, D, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed bounded-variation packet, Real or regular endpoint reads, stream windows, dyadic variation ledger, variation-norm row, and structural transport and provenance. It has no coordinate for a completed function space, host function equality, variation measure, hidden partition family, or supremum over all partitions.

Lean4 ✓ `BEDC.Derived.VariationNormUp.VariationNormTasteGate single carrier alignment`

Theorem 2832.2 (VariationNorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationNorm}^{\uparrow}$ *carrier* $V = (B, R, Q, S, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$ *obligation surface consists exactly of bounded-variation source admission, Real endpoint sealing, RegSeqRat readback, finite StreamName window selection, DyadicRatCore edge accounting, displayed variation-norm row formation, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier. The only source of variation data is the $\mathsf{BoundedVariation}^{\uparrow}$ row B, and the only real-valued reads attached to it are the Real seal R, the regular-name row Q, the stream window S, and the dyadic ledger D.

The variation-norm row A is then formed from those displayed rows. It reads a finite base-value component and the displayed partition-variation sum, both already present in the bounded-variation ledger and its Real/Completion dependency route.

Transport and replay are handled by H and C. These rows move the same finite source, endpoint, stream, dyadic, and norm data componentwise, while P and N record provenance and local naming. Since the excluded analytic objects have no carrier coordinate, they cannot be part of the local obligation surface. □

Theorem 2832.3 (VariationNorm refinement stability)**.** *Let* $V = (B, R, Q, S, D, A, H, C, P, N)$ *and* $V' = (B', R', Q', S', D', A', H', C', P', N')$ *be accepted* $\mathsf{VariationNorm}^{\uparrow}$ *carriers whose bounded-variation source packets compare by a displayed common-refinement classifier. Then the variation-norm rows* A *and* A' *compare componentwise after transporting the Real endpoint seals, RegSeqRat readbacks, StreamName windows, DyadicRatCore edge ledgers,* Pkg *provenance, and local* NameCert *rows through the stored* hsame *and* Cont *routes.*

Proof. Begin with the common-refinement classifier for the source $\mathsf{BoundedVariation}^{\uparrow}$ packets. By chapter 398, the comparison is a displayed finite refinement with transported endpoint rows, dyadic edge observations, finite variation sums, and package and naming provenance.

Read the Real and completion-facing coordinates along that same refinement. The rows R, Q, S, D and R', Q', S', D' expose the Real seal, regular-name readback, stream windows, and dyadic edge ledgers without adding a hidden partition or completed function carrier.

This variation-norm row is a carrier and classifier consumer of the $\mathsf{BoundedVariation}^{\uparrow}$ partition ledger and of the L10 Real/Completion readback route through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$; it is not a decorative citation detached from the finite source rows.

Finally replay the norm rows through C and C' and transport them through H and H'. The row A is built from the same finite base-value and partition-variation data as A', after the displayed common refinement, so the variation-norm comparison is stable under the classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VariationNorm}^{\uparrow}$ packet is generated from a BoundedVariation source, Real endpoint seal, RegSeqRat readback, StreamName window, DyadicRatCore edge ledger, displayed variation-norm row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$, obligation)

Scope closed: Definition 3804.1, Definition 3804.2, and Definition 3804.3 bind the finite variation-norm carrier, obligation surface, and common-refinement stability route to the displayed BoundedVariation and Real/Completion dependency rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{VariationNorm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed bounded-variation function spaces, host function equality, a variation measure, hidden partition families, a supremum over all partitions, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked finite partition-norm consumer theorem over the same bounded-variation source, Real seal, regular-name, stream-window, dyadic-ledger, transport, replay, provenance, and local naming rows.

Chapter 2833

A Concrete Naming Certificate for $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges on regular Cauchy names, the $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ packet records the finite basis rows that sit between regular-Cauchy closeness and the terminal real seal. It refines the dyadic-entourage surface of chapter 1628 by naming basis generators before they are consumed by the $\mathsf{Real}^{\uparrow}$ route. The packet reads the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the real endpoint seal of chapter 38; it is not an ambient uniformity parameter, a quotient equality of Cauchy presentations, a selected limit, countable choice, or a completeness theorem.

Definition 2833.1 (Regular Cauchy entourage-basis carrier)**.** *A* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, B, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational Cauchy presentation whose finite windows are being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the displayed observation windows for the same presentation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic radii, denominator scales, and comparison data available to those windows.*
4. *B is the basis-entourage row, assigning each displayed dyadic tolerance scale to the finite entourage generator read from R, W, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source, window, tolerance, and basis-entourage rows have been displayed.*
6. *H records componentwise* hsame *transport for the regular source, window, tolerance, basis, real seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a basis-entourage request through R, W, D, B, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient uniform-space object, an arbitrary entourage filter, quotient equality of real presentations, a selected limit, countable choice, or real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyEntourageBasisUp.RegularCauchyEntourageBasisTasteGate single carrier alignment`

Theorem 2833.2 (Regular Cauchy entourage-basis NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier* $K = (R, W, D, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, regular-source exposure through* R*, finite-window exposure through* W*, dyadic-tolerance exposure through* D*, basis-entourage exactness through* B*, real-seal nonescape through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3805.1. The source, window, and tolerance rows are the only inputs from which the basis-entourage row can be read.

Basis-entourage exactness is finite because B assigns displayed dyadic tolerance scales from D to displayed windows from W for the same regular source row R. This gives the basis-level row used by chapter 1628 without importing an ambient uniformity object.

The real seal is nonescaping because E is reached only after R, W, D, B have been displayed and transported through H. Replay through C preserves the same route, while P and N record provenance and local naming for that finite packet. Thus no quotient equality, selected limit, countable-choice schedule, or completeness theorem can be projected from the NameCert surface. □

Theorem 2833.3 (Regular Cauchy entourage-basis Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyEntourageBasis}^{\uparrow} \longmapsto \mathsf{RegularCauchyUniformit}$$

The route exposes a basis-entourage generator before the uniformity and real-seal rows are read, and exports no ambient uniformity parameter, no selected completion limit, and no host equality of real presentations.

Proof. Start with the accepted carrier $K = (R, W, D, B, E, H, C, P, N)$. A real-seal request first reads R as the regular rational source, W as its finite observation window, and D as the dyadic tolerance ledger for that window.

The basis row B is then read as the finite entourage generator for the displayed tolerance scale. The regular-Cauchy uniformity consumer can use this row as its basis-level input, but it receives no coordinate for a larger ambient uniformity filter.

Finally the route reaches E through componentwise transport, replay, provenance, and local naming. Since the only visible path to E runs through R, W, D, B, H, C, P, N, the handoff cannot manufacture quotient equality, countable choice, selected limits, or completeness outside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ read exposes RegSeqRat source rows, StreamName finite windows, DyadicRatCore tolerance rows, a basis-entourage row, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any Real-facing consumer can use the packet.

Independence witness: regular-cauchy-uniformity, regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source row, finite windows, tolerance ledger, basis-entourage generator, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ packet is generated from a RegSeqRat source row, a StreamName finite-window row, a DyadicRatCore tolerance row, a basis-entourage row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3805.1, Definition 3805.2, and Definition 3805.3 seed the finite basis-level route between regular-Cauchy windows and the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of real presentations, selected limits, countable choice, ambient uniformity completeness, or Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, basis-entourage exactness, classifier stability, real-seal nonescape, and Real handoff for the displayed packet.

Chapter 2834

A Concrete Naming Certificate for $\mathsf{TriggerEnergyPartition}^{\uparrow}$

The $\mathsf{TriggerEnergyPartition}^{\uparrow}$ packet realizes the trigger-energy theorem of Definition 9462.1584 as finite BEDC NameCert data. It records support rows, an energy ledger, a partition row, codon-lift accounting, transport, replay, provenance, and local naming for the energy-minimal trigger surface. It is a sibling of the reliability packet in chapter 3583: $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumes energy-weighted reliability rows, while $\mathsf{TriggerEnergyPartition}^{\uparrow}$ isolates the partition certificate for equality in the energy lower bound.

Definition 2834.1 (Trigger-energy partition carrier)**.** *A* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, L, E, Q, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite support row over the four trigger coordinates $\Omega = \{s_1, f_1, s_2, f_2\}$.*
2. *L is the trigger-selection row naming the displayed quotient-level trigger and its codon-level lift choices.*
3. *E is the energy-ledger row recording the sum of the support sizes of the selected external codons.*
4. *Q is the partition row certifying that the selected supports are pairwise disjoint and cover Ω exactly in the energy-minimal case.*
5. *K is the codon-lift row recording the finite f_3-lift multiplicity for each selected support block.*
6. *H records componentwise* hsame *transport for support, selection, energy, partition, codon-lift, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an energy-minimal trigger request through S, L, E, Q, K, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{TriggerEnergyPartition}^{\uparrow}$*,* $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*, the source theorem Definition 9462.1584,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$ *row.*

The carrier contains no biological stochastic model, empirical likelihood claim, hidden codon semantics, unbounded trigger enumeration, or external partition oracle.

Lean4 ✓ `BEDC.Derived.TriggerEnergyPartitionUp.TriggerEnergyPartitionTasteGate single carrier alignment`

Theorem 2834.2 (Trigger-energy partition NameCert obligations)**.** *For every accepted* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *carrier* $T = (S, L, E, Q, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, finite support exposure through* S*, trigger-selection exposure through* L*, energy-ledger exactness through* E*, partition exactness through* Q*, codon-lift accounting through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3806.1. The support row S and selection row L expose the finite trigger family, while E records the total support-size ledger.

Partition exactness reads the equality case of Definition 9462.1584. When the energy ledger has value 4, the partition row Q records that the selected supports cover the four coordinates exactly once; conversely, the displayed partition row forces the energy ledger to count four coordinate appearances.

The codon-lift row K is finite because it records only the displayed f_3 choices over the selected support blocks. Transport and replay preserve S, L, E, Q, K componentwise, and provenance plus local naming keep the certificate tied to the source theorem and the energy-biased reliability sibling. No stochastic, empirical, hidden-semantic, or unbounded-search row is exported. □

Theorem 2834.3 (Trigger-energy partition reliability handoff)**.** *Every energy-minimal trigger read consumed by* $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ *factors through the finite* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *route*

$$S, L, E, Q, K, H, C, P, N.$$

The reliability consumer receives the support family, energy ledger, partition certificate, and codon-lift count as finite rows, not as an external enumeration or biological probability model.

Proof. Start with an accepted trigger-energy partition carrier. The finite support row S names the coordinate supports, and L names the selected quotient trigger together with its codon-level lift surface.

The energy ledger E and partition row Q then expose the equality case in the lower bound: every coordinate of Ω is covered exactly once. The codon-lift row K records the finite lift multiplicities attached to those blocks.

An $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumer may read these rows as inputs for its weighted reliability formula, but the transport, replay, provenance, and local naming rows keep the handoff inside the displayed finite certificate. Thus the consumer obtains no external trigger enumeration, empirical likelihood, or hidden codon semantics. □

Falsifiable prediction: Every accepted $\mathsf{TriggerEnergyPartition}^{\uparrow}$ read exposes finite support, trigger-selection, energy-ledger, partition, codon-lift, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any energy-biased reliability consumer can use an energy-minimal trigger partition.

Independence witness: energy-biased-trigger-reliability, six-boundary-tile-coverage, window-coverage-trigger-algebra: $\mathsf{TriggerEnergyPartition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the energy equality partition row and codon-lift accounting rather than only coverage, algebra, or reliability rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$**.**

Constructive story: A TriggerEnergyPartition$^{\uparrow}$ packet is generated from finite support rows, trigger-selection rows, an energy ledger, a partition certificate, codon-lift accounting, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{TriggerEnergyPartition}^{\uparrow}}$, seed)

Scope closed: Definition 3806.1, Definition 3806.2, and Definition 3806.3 seed the finite partition surface for equality in the trigger-energy lower bound.

Formal verification: FormalStatus(NameCert$_{\text{TriggerEnergyPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biological stochastic semantics, empirical likelihood, hidden codon semantics, or unrestricted trigger enumeration.

Upgrade path: Obligation closure requires checked carrier admission, support exposure, energy-ledger exactness, partition exactness, codon-lift accounting, reliability handoff, Nontrivial TriggerEnergyPartitionUp, and FieldFaithful TriggerEnergyPartitionUp for BEDC.Derived.TriggerEnergyPartitionUp.

Chapter 2835

A Concrete Naming Certificate for $\mathsf{ContinuousSupNorm}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ContinuousSupNorm}^{\uparrow}$ packet records the compact-source uniform metric for bounded continuous real-valued functions as finite BEDC data. It reads the compact metric source of chapter 129, the continuous-map row of chapter 130, the real-valued uniform Cauchy route of chapter 1830, the uniform-limit metric handoff of chapter 1764, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite sup-norm surface needed by compact continuous-function consumers without importing a completed function space, host function equality, or a global supremum operator outside the displayed ledger.

Definition 2835.1 (Continuous sup-norm carrier)**.** *A* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, R, O, M, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for a real-valued function on K.*
3. *R is the pointwise* $\mathsf{Real}^{\uparrow}$ *difference and absolute-value seal for inspected source windows.*
4. *O is the finite oscillation ledger, recording the inspected cover cells, source windows, and value-oscillation bounds.*
5. *M is the displayed modulus row that transports continuous-map control from source-window refinement to real-value tolerance.*
6. *B is the supremum-bound witness, recording a finite bound row for the displayed oscillation and pointwise-difference ledger.*
7. *H records componentwise* hsame *transport for the compact source, continuous-map row, Real difference row, oscillation ledger, modulus row, bound witness, replay row, provenance, and local name.*
8. *C records the displayed* Cont *replay route from a sup-norm request through $K, F, R, O, M, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{UniformLimitMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed compact source, continuous-map row, pointwise Real difference row, finite oscillation and modulus ledgers, supremum-bound witness, structural transport, replay, provenance, and local naming. It has no coordinate for a completed Banach space, quotient equality of functions, an ambient supremum over all host points, a choice-selected extremal point, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContinuousSupNormUp.ContinuousSupNormTasteGate single carrier alignment`

Theorem 2835.2 (ContinuousSupNorm NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *surface consists exactly of compact metric source admission through* K*, continuous-map graph and pointwise-continuity admission through* F*, pointwise Real difference sealing through* R*, finite oscillation exposure through* O*, modulus transport through* M*, finite sup-bound exposure through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3807.1. The tuple displays the compact source row, continuous-map row, Real difference row, finite oscillation ledger, modulus row, finite bound witness, transport, replay, provenance, and local naming.

The compact source and continuous-map rows determine the only function values that may be inspected. The Real difference row and finite oscillation ledger compare those displayed values, while the modulus row keeps the comparison tied to the same compact source windows.

The finite bound witness records the sup-norm-facing estimate without selecting an ambient supremum. Since the remaining structural rows only transport, replay, source, and name the same displayed data, the listed rows exhaust the local NameCert obligations. □

Theorem 2835.3 (ContinuousSupNorm compact-uniform handoff)**.** *Every compact-continuous norm consumer of an accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *packet reads the norm route through compact metric source rows, continuous-map rows, finite oscillation rows, modulus rows, Real difference seals, finite sup-bound witnesses,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data. The route may be consumed by compact-uniform and uniform-modulus surfaces such as chapter 388, chapter 369, and chapter 130, but it exports no host function equality, open-cover compactness, quotient real equality, selected supremum, or ambient completeness.*

Proof. Use Definition 3807.2. It orders the compact source row, continuous-map row, Real difference row, oscillation ledger, modulus row, and finite bound witness inside one accepted packet.

The compact and continuous rows expose only finite inspection windows and displayed graph reads. The modulus and oscillation rows transport those reads to a common Real tolerance, and the bound witness records the finite norm-facing estimate.

Thus every compact-uniform consumer receives the displayed BEDC route. The carrier has no coordinate for a host function object, open-cover compactness object, quotient Real equality, selected supremum, or ambient completeness theorem. □

Theorem 2835.4 (ContinuousSupNorm Real seal nonescape)**.** *Every Real-valued bound exported by an accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier passes through the pointwise* $\mathsf{Real}^{\uparrow}$ *difference seal* R *and the finite sup-bound witness* B*. Consequently the exported sup-norm data can depend only on compact source windows, continuous-map rows, finite oscillation and modulus ledgers, displayed Real comparison rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Begin with the accepted carrier. The Real-facing coordinate is R, and the norm-facing bound coordinate is B.

Any bound read first uses the compact source K and continuous-map row F, then the oscillation and modulus rows O, M, and then the Real difference seal R. Only after those displayed rows are fixed can the finite bound witness B export a sup-norm-facing value.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since no other Real-valued coordinate appears in the carrier, the exported bound cannot leave the displayed Real-seal surface. □

Theorem 2835.5 (Continuous sup-norm uniform-limit metric export)**.** *Every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$ *handed to* $\mathsf{UniformLimitMetric}^{\uparrow}$ *or* $\mathsf{UniformSequentialLimit}^{\uparrow}$ *exports only compact metric source rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, finite oscillation and modulus ledgers, pointwise* $\mathsf{Real}^{\uparrow}$ *difference seals, finite sup-bound witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *row. The export route has no coordinate for host function equality, a selected supremum point, a completed function-space quotient, a choice-selected limiting function, or ambient compactness outside the displayed compact metric source.*

Proof. Project the carrier and apply Definition 3807.3. The compact source, continuous-map row, oscillation ledger, modulus row, Real difference seal, and finite bound witness already form the only compact-uniform route available to a consumer.

Use Definition 3807.4 for the Real-facing boundary. It forces every exported bound through the displayed pointwise difference seal and finite sup-bound witness.

Finally compose this boundary with the Cauchy-control handoff of Definition 3807.6. The resulting uniform-limit metric export reads exactly $K, F, O, M, R, B, H, C, P, N$ and nothing else.

Since the carrier has no coordinate for host function equality, selected supremum data, completed function-space quotients, selected limiting functions, or ambient compactness beyond K, both $\mathsf{UniformLimitMetric}^{\uparrow}$ and $\mathsf{UniformSequentialLimit}^{\uparrow}$ receive only the finite BEDC route. □

Theorem 2835.6 (Continuous sup-norm Cauchy control)**.** *Every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$ *that is read as a sup-norm Cauchy row supplies uniform finite-window Cauchy control over the compact source. For each displayed tolerance request, the accepted read factors through the finite route*

$$K, F \longmapsto O, M \longmapsto R, B \longmapsto H, C, P, N.$$

Thus every inspected pair of function rows is controlled by the same compact-source window ledger and the same RealUp difference-and-bound witness.

Proof. Project the carrier of Definition 3807.1. The compact source K and continuous-map row F determine which source windows and pointwise Real reads can be inspected.

The finite oscillation row O and modulus row M then give a single finite window ledger for the tolerance request. The Real difference row R and supremum-bound witness B read all inspected pointwise differences through that ledger.

Finally, H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores host function equality, a completed function space, or an ambient supremum operator, the Cauchy control available to a consumer is exactly the displayed uniform finite-window control. □

Theorem 2835.7 (Continuous sup-norm uniform-limit handoff)**.** *Every* ContinuousSupNorm$^{\uparrow}$ *Cauchy-control read accepted by* $S = (K, F, R, O, M, B, H, C, P, N)$ *may be handed to the* UniformLimitMetric$^{\uparrow}$ *packet only through the displayed compact source, continuous-map row, finite oscillation and modulus ledgers, RealUp pointwise-difference seal, supremum-bound witness, componentwise transport, replay, provenance, and local naming rows. The same sup-norm comparison row is consumed by* UniformSequentialLimit$^{\uparrow}$ *through chapter 3874 when a uniform tail schedule is read as one tolerance-controlled Real-limit route. The handoff exports no additional completeness, quotient-function equality, or selected limit object.*

Proof. Begin with the Cauchy-control route of Definition 3807.6. It exposes the compact source and continuous-map row before the oscillation, modulus, Real difference, and bound rows are consumed.

The UniformLimitMetric$^{\uparrow}$ consumer can therefore read only the finite route already present in $K, F, O, M, R, B, H, C, P, N$. The package provenance P records the connection with RealFunctionUniformCauchy$^{\uparrow}$ and UniformLimitMetric$^{\uparrow}$, while N names only the local sup-norm packet.

There is no remaining coordinate for a host completion theorem, a quotient of functions, or a choice-selected limiting function. Hence the handoff is exactly the finite BEDC route displayed by the carrier. □

Closure status for NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$**.**

Constructive story: A ContinuousSupNorm$^{\uparrow}$ packet is generated from a compact metric source, continuous-map row, pointwise RealUp difference seal, finite oscillation and modulus ledgers, supremum-bound witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$, seed)

Scope closed: Definition 3807.1, Definition 3807.6, Definition 3807.5, and Definition 3807.7 seed the compact-source sup-norm carrier, Cauchy-control route, uniform-limit metric export, and uniform-limit handoff for bounded continuous real-valued functions.

Formal verification: FormalStatus(NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach-space completeness, quotient equality of functions, host supremum existence, selected limit functions, Lean verification, or bridge checking for ContinuousSupNorm$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite oscillation coverage, modulus transport, RealUp difference sealing, supremum-bound exactness, Cauchy-control routing, uniform-limit handoff, replay stability, and ledger non-escape for BEDC.Derived.ContinuousSupNormUp.

Chapter 2836

A Concrete Naming Certificate for $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$

The $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ packet realizes the bounded redex-frontier route needed by the MetaCIC confluence open problem of section 9386.3. It sits beside the residual-substitution boundary of chapter 2181, the closed parallel-diamond premise ledger of chapter 2325, the local diamond witness of chapter 3595, and the critical-pair boundary of chapter 2212. The packet records a finite bounded route for residual substitution without converting that route into full closed confluence.

Falsifiable prediction: Any claimed full parallel diamond for MetaCIC closed terms that bypasses the displayed $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ route through finite redex-frontier rows, residual-route rows, bounded checker rows, local joins, obstruction retention, transport, replay, provenance, and local NameCert data fails the packet boundary.

Independence witness: metacic-residual-substitution-boundary, closed-parallel-diamond-premise-ledger, metacic-local-diamond-witness, metacic-confluence-critical-pair-boundary: none is bijective to $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ because this packet jointly carries bounded redex-frontier data, residual-route accounting, bounded checking, local joins, obstruction retention, transport, replay, provenance, and local naming as one finite frontier.

Definition 2836.1 (Bounded residual substitution frontier carrier)**.** *A* $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, S, B, J, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite redex-frontier row, listing the inspected beta-redex positions and their bounded residual descendants.*
2. *S is the residual-route row, recording the displayed substitution route and binder-depth ledger for the bounded window.*
3. *B is the bounded checker row, reading only finite closedness, variable-escape, and branch-compatibility checks.*
4. *J is the local join row, storing the common-reduct candidate produced inside the bounded frontier.*

5. *O is the obstruction row, retaining the absent endpoints: full* BetaParallelSubstitutionCompatibility*, premise-free closed parallel diamond, arbitrary closed Church–Rosser, and unchecked subject-reduction discharge.*
6. *H records componentwise* hsame *transport for the redex frontier, residual route, bounded checker, local join, obstruction, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay from a confluence-frontier request through R, S, B, J, O, H, P, N.*
8. *P is the* Pkg *provenance over* BoundedResidualSubstitutionFrontier$^{\uparrow}$*,* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*,* ClosedParallelDiamondPremiseLedger$^{\uparrow}$*,* MetaCICLocalDiamondWitness$^{\uparrow}$*,* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* NameCert$_{\text{BoundedResidualSubstitutionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for full closed confluence, unrestricted residual-substitution compatibility, a hidden substitution theorem, evaluator equality, or kernel consistency.

Lean4 ✓ `BEDC.Derived.BoundedResidualSubstitutionFrontierUp.BoundedResidualSubstitutionFrontierTasteGate`
`single carrier alignment`

Theorem 2836.2 (Bounded residual substitution frontier obligations)**.** *For every accepted* BoundedResidualSubstitu *carrier $F = (R, S, B, J, O, H, C, P, N)$, the local* NameCert$_{\text{BoundedResidualSubstitutionFrontier}^{\uparrow}}$ *surface consists exactly of finite redex-frontier exposure through R, residual-route exposure through S, bounded checking through B, local join exposure through J, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The confluence-facing read is finite:

$$R, S, B \longmapsto J, O \longmapsto H, C, P, N.$$

It does not escape to full closed confluence; any consumer still sees the residual-substitution and closed parallel-diamond premise boundaries as displayed inputs.

Proof. Project the carrier of Definition 3808.1. The rows $R, S, B, J, O, H, C, P, N$ are the whole packet, so every local obligation is read from one displayed coordinate.

The bounded route begins with R, S, B. The redex-frontier row lists the inspected positions, the residual-route row records the substitution accounting, and the bounded checker row verifies only the finite window available to the packet.

The local join row J is therefore downstream of bounded evidence. The obstruction row O records that full substitution compatibility, a premise-free closed parallel diamond, arbitrary closed Church–Rosser, and unchecked subject-reduction discharge are not exported.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for a global confluence theorem or hidden substitution theorem, the displayed obligations cannot escape their bounded frontier. □

Lean4 ✓ `BEDC.Derived.BoundedResidualSubstitutionFrontierUp.BoundedResidualSubstitutionFrontierNamecert(`

Theorem 2836.3 (MetaCIC residual-substitution candidate support)**.** *Any candidate-mediated strong-normalization or confluence consumer using an accepted* BoundedResidualSubstitutionFrontier$^{\uparrow}$ *carrier $F = (R, S, B, J, O, H, C, P, N)$ must expose $R, S, B, J, O, H, C, P, N$ before projecting a common-reduct candidate. In particular, it must cite the residual-substitution boundary, the closed*

parallel-diamond premise ledger, the local diamond witness, and the critical-pair boundary before treating J as a confluence-facing candidate.

The read presents only bounded candidate support. The obstruction row O remains visible as the refusal row for full confluence, unrestricted substitution compatibility, premise-free closed parallel diamond, and unchecked subject-reduction discharge.

Proof. Project the carrier of Definition 3808.1. The rows $R, S, B, J, O, H, C, P, N$ are the complete support surface available to the consumer.

Apply Definition 3808.2. It orders the finite redex-frontier row R, residual-route row S, and bounded checker row B before the local join row J can be read.

The candidate projection therefore remains mediated by the sibling boundaries named in the provenance row: residual substitution supplies the boundary of the route, the closed parallel-diamond premise ledger supplies the premise account, the local diamond witness supplies the bounded join, and the critical-pair boundary supplies the finite obstruction interface.

Keep O in the exported packet. Since O records exactly the missing full-confluence and unrestricted-substitution data, the common-reduct candidate is a bounded local candidate rather than a proof of the unrestricted MetaCIC confluence claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ packet is generated from finite redex-frontier rows, residual-route accounting, bounded checker rows, local joins, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3808.1 and Definition 3808.2 seed the finite bounded residual-substitution frontier for MetaCIC confluence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full BetaParallelSubstitutionCompatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, redex-frontier coverage, residual-route exactness, bounded-checker soundness, local-join exposure, obstruction retention, transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.BoundedResidualSubstitutionFrontierUp.

Chapter 2837

A Concrete Naming Certificate for $\mathsf{LocatedRealContinuity}^{\uparrow}$

Following Bishop and Bridges, located continuity is recorded as finite precision-response data over located real names, not as a host function on completed reals. The $\mathsf{LocatedRealContinuity}^{\uparrow}$ packet sits over the located-real carrier of **??**, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located modulus surface of chapter 3406. It gives the L10 Real and Completion route a named continuity consumer whose public data are precision requests and finite responses rather than quotient-real functions, trichotomy, countable choice, or ambient completeness.

Definition 2837.1 (Located real continuity carrier)**.** *A* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, D, W, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the located input row, read through the* $\mathsf{LocatedReal}^{\uparrow}$ *surface.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for requested input and output tolerances.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window used to inspect the located input at the requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the inspected window as regular rational data.*
5. *M is the modulus-response row, returning a finite precision response for the displayed input tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the output name, visible only after X, D, W, R, M have been displayed.*
7. *H records componentwise* hsame *transport for the located input, dyadic ledger, stream window, regular readback, modulus response, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located precision request through $X, D, W, R, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealContinuity}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host real function, quotient-real equality test, trichotomy oracle, countable-choice schedule, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedRealContinuityUp.LocatedRealContinuityTasteGate single carrier alignment`

Theorem 2837.2 (Located real continuity modulus stability)**.** *Let* $K = (X, D, W, R, M, E, H, C, P, N)$ *and* $K' = (X', D', W', R', M', E', H', C', P', N')$ *be accepted* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carriers. If the located input rows, dyadic tolerance ledgers, finite windows, and regular-rational readbacks agree componentwise under* hsame*, then the modulus-response rows* M *and* M' *answer the same displayed dyadic tolerance requests, and every* $\mathsf{Real}^{\uparrow}$ *seal consumer reaches the same response route*

$$X, D, W, R \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

Thus modulus-response stability is a finite packet property: it is preserved by hsame *transport and by the displayed dyadic tolerance rows, without reading quotient-real equality, trichotomy, countable choice, selected representatives, host function extensionality, or ambient completeness.*

Proof. Project the two accepted carriers from Definition 3809.1. The only precision source rows are X, D, W, R and their transported counterparts.

Use the assumed componentwise hsame agreement on those rows. The dyadic ledger D fixes the tolerance request, the window W fixes the finite observations, and R fixes the regular-rational readback. Therefore the accepted modulus rows M and M' respond to the same displayed precision request.

The seal row is downstream of the response row. Since E and E' can be consumed only after the displayed route through the input, tolerance, window, readback, and response rows, any $\mathsf{Real}^{\uparrow}$ consumer reads the same finite route before seeing a real-name seal.

The excluded analytic principles have no carrier coordinate in either packet. They cannot alter the transported modulus response or add a second continuity channel. □

Theorem 2837.3 (Located real continuity NameCert obligations)**.** *For an accepted* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carrier* $K = (X, D, W, R, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability for the located input and precision-response rows, componentwise transport, the continuity-modulus ledger, terminal real-seal routing, package provenance, and local naming. The obligation surface does not include quotient reals, trichotomy, countable choice, selected representatives, host function extensionality, or ambient completeness.*

Proof. Carrier admission is projection from Definition 3809.1. The accepted packet displays exactly the located input, dyadic tolerance ledger, finite window, regular readback, modulus response, real seal, transport, replay, provenance, and local naming rows.

Classifier stability is supplied by the same inventory. An accepted continuity read must keep the rows X, D, W, R, M attached to one displayed precision route, so changing any of these rows changes the carrier rather than producing a hidden function on completed reals.

Transport and replay are the rows H and C. They preserve the displayed route through the dyadic tolerance and modulus-response ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally P and N source and name the same finite packet. Since quotient reals, trichotomy, countable choice, selected representatives, host function extensionality, and ambient completeness have no carrier rows, they are not local NameCert obligations for $\mathsf{LocatedRealContinuity}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealContinuity}^{\uparrow}$ packet is generated from a located input row, a dyadic precision ledger, finite StreamName windows, RegSeqRat readback, a modulus-response row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3809.1, Definition 3809.2, and Definition 3809.3 seed the finite located-continuity route from precision requests to RealUp seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions on completed reals, quotient-real equality, trichotomy, countable choice, selected representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, transport, continuity-modulus ledger rows, real-seal routing, provenance, and local naming for BEDC.Derived.LocatedRealContinuityUp.

Chapter 2838

A Concrete Naming Certificate for CellularSubstrateClassifierBoundary$^{\uparrow}$

The CellularSubstrateClassifierBoundary$^{\uparrow}$ packet realizes the cellular-automaton substrate reading of chapter 9390 as a finite classifier-boundary carrier. It binds the local transition table, orbit replay, pattern-classifier tag row, closed-observation manifest, and refusal boundary into one NameCert surface. The packet is designed to accept cellular substrate observations only when the classifier that names the observed pattern is displayed as data, so no untagged orbit replay can masquerade as a closed observation.

Before its manifest and closed-observation rows are exported, the packet consumes the rule-table and finite-orbit rows of chapter 934, the substrate carrier of chapter 942, and the classifier comparison surface of chapter 945. These sibling routes determine which local transition, orbit replay, and pattern tag rows may enter the boundary carrier.

The classifier-boundary packet also consumes the observation-window discipline of chapter 2371 and the classifier-row surface of chapter 945 before its Rule 110 handoff row is exposed. Thus the closed-observation boundary and the classifier lattice edge are both visible before a universal-substrate read can use the packet.

Definition 2838.1 (CellularSubstrateClassifierBoundary carrier)**.** *A* CellularSubstrateClassifierBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, O, G, M, A, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the local cellular transition row, including the finite alphabet, neighborhood window, and rule table.*
2. *O is the orbit replay row generated by iterating L over the stated initial row and finite time window.*
3. *G is the pattern-classifier tag row that assigns each inspected local pattern to a finite tag family.*
4. *M is the manifest row specifying the accepted observation windows, expected replay checks, and public audit fields.*
5. *A is the closed-observation admission row, requiring the replayed orbit and the classifier tags to agree on every observed window.*

6. *R is the Rule 110 substrate handoff row when the local rule is read through the cellular universal-substrate exhibit.*
7. *H records componentwise* hsame *transport for the local rule, orbit replay, classifier tags, manifest, admission row, Rule 110 handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from local transition data through orbit replay, pattern classification, closed-observation admission, and any Rule 110 handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$*, cellular-automaton substrate reading, Rule 110 substrate, classifier tags, closed observation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unclassified orbit, a detached semantic interpretation of a pattern, an induction principle over all time steps, or an observation that bypasses the manifest and classifier rows.

```
Lean4 ✓  BEDC.Derived.CellularSubstrateClassifierBoundaryUp.CellularSubstrateClassifierBoundaryTasteGate
single carrier alignment
```

Theorem 2838.2 (CellularSubstrateClassifierBoundary NameCert obligations)**.** *For every accepted* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *carrier* $B = (L, O, G, M, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the local transition table, orbit replay, classifier tag row, manifest, closed-observation admission row, optional Rule 110 handoff, transport, replay, provenance, and local naming;*
2. *local-rule replay, requiring every accepted observation to display L and O before any classifier-boundary claim is consumed;*
3. *classifier-boundary exposure, requiring the tag row G to be present for each observed pattern window;*
4. *manifest closure, requiring the row M and admission row A to bind replayed orbits and classifier tags into one finite observation record;*
5. *Rule 110 handoff discipline, requiring R when the packet is read through the universal cellular substrate route;*
6. *classifier transport, requiring all accepted comparisons to use only H, C, P, and N;*
7. *ledger refusal, excluding untagged orbit replay, detached semantic interpretation, all-time induction, and observations outside the manifest and classifier rows.*

No classifier-boundary obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3810.1. The accepted packet displays exactly the local rule, orbit replay, classifier tag row, manifest, admission row, Rule 110 handoff, and structural rows.

Local-rule replay comes first. The orbit row O is generated from the transition row L, so no accepted observation can be read without displaying the finite rule table and the replay window.

Classifier-boundary exposure and manifest closure then bind the orbit to a tag discipline. The tag row G names the inspected patterns, while M and A require those tags to agree with the manifest's closed-observation checks.

Finally the Rule 110 handoff and structural rows preserve the same route. The optional row R names the universal cellular substrate path when present, and H, C, P, N transport, replay, source, and name the local carrier. Since untagged or detached observations have no row in the packet, they cannot become NameCert obligations. □

Falsifiable prediction: A packet with a local transition row and orbit replay but no pattern-classifier tag row is rejected by the classifier-boundary carrier.

Independence witness: cellular-automaton-substrate-reading, metacicredexfrontier, rule110substrate, binaire, apophaticname: the packet is not bijective to these siblings because it binds local cellular transition, orbit replay, classifier tags, closed-observation admission, Rule 110 handoff, and classifier-boundary refusal in one finite NameCert route.

Theorem 2838.3 (CellularSubstrateClassifierBoundary closed-observation handoff)**.** *Every closed cellular observation accepted by a* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *packet reads the route*

$$L, O \longmapsto G \longmapsto M, A \longmapsto R \longmapsto H, C, P, N.$$

The route starts with a finite local transition rule and orbit replay, passes through the pattern-classifier tag row, binds the result to a manifest and closed-observation admission row, and reaches the Rule 110 handoff only as displayed substrate data. A Lean follow-up for this carrier must supply Nontrivial *and* FieldFaithful *instances in* `lean4/BEDC/Derived/CellularSubstrateClassifierBoundaryUp/Taste`

Proof. Project the accepted carrier of Definition 3810.1. The rows L and O are the only local-rule and orbit-replay rows, and G is the only classifier row attached to the inspected patterns.

Read manifest admission next. The rows M and A bind the displayed classifier tags to the closed-observation record, so an orbit replay without G cannot pass the admission surface.

The remaining rows preserve the same finite dependency chain. The Rule 110 handoff R is downstream of rule, orbit, classifier, and manifest data, while H, C, P, N transport, replay, source, and name the accepted route. Therefore every accepted observation factors through the classifier boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ packet is generated from a local cellular transition table, orbit replay, pattern-classifier tags, a closed-observation manifest, an admission row, an optional Rule110Substrate handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3810.1, Definition 3810.2, and Definition 3810.3 seed the finite classifier-boundary surface for cellular substrate observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close all-time induction, unclassified orbit semantics, a complete Rule 110 bridge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-rule replay, classifier-boundary exposure, manifest closure, Rule 110 handoff discipline, classifier transport, and ledger refusal for BEDC.Derived.CellularSubstrateClassifierBoundaryUp.

Chapter 2839

A Concrete Naming Certificate for HadamardProductSeries$^{\uparrow}$

Following Hadamard's coefficientwise product of power series and Hardy's series estimates, the HadamardProductSeries$^{\uparrow}$ packet records the finite route by which two displayed real coefficient streams are multiplied coordinate by coordinate before they are handed to the ordinary CauchyProduct$^{\uparrow}$ and RealSeries$^{\uparrow}$ convergence surfaces. It sits beside the real-series carrier of chapter 1581, the Cauchy-product certificate of chapter 464, the majorant route of chapter 399, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic coefficient ledger of chapter 291, and the terminal real seal of chapter 38. The packet names coefficientwise multiplication as finite BEDC data; it is not a host analytic-function identity, not a quotient of series, not a selected infinite sum, and not an ambient completeness theorem.

Definition 2839.1 (Hadamard product series carrier)**.** *A* HadamardProductSeries$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, R, S, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source* RealSeries$^{\uparrow}$ *coefficient-stream rows.*
2. *W is the synchronized* StreamName$^{\uparrow}$ *window row selecting matching coefficient positions from A and B.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *coefficient-product ledger; for every displayed position in W it records the dyadic product of the two read coefficients and its tolerance budget.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row repacking the coefficientwise products as regular rational finite-window data.*
5. *S is the derived* RealSeries$^{\uparrow}$ *coefficient stream assembled from R and the synchronized windows.*
6. *M is the convergence-control row, either a displayed* WeierstrassMTest$^{\uparrow}$ *majorant for the product coefficients or a* CauchyProduct$^{\uparrow}$ *handoff consuming the same finite product ledger.*
7. *E is the terminal* Real$^{\uparrow}$ *seal exposed only after A, B, W, D, R, S, M have been replayed.*
8. *H records componentwise* hsame *transport for the two source series, the synchronized windows, dyadic products, regular readback, derived series, convergence-control row, endpoint seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from paired coefficient requests through synchronized windows, dyadic products, regular readback, the derived series, convergence control, and the endpoint seal.*
10. *P is the* Pkg *provenance over* HadamardProductSeries$^{\uparrow}$, RealSeries$^{\uparrow}$, CauchyProduct$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HadamardProductSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, D, R, S, M, E, H, C, P, N$. It has no coordinate for host analytic continuation, quotient series equality, selected infinite sums, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 2839.2 (Hadamard product series NameCert obligations)**.** *For every accepted* HadamardProductSeries$^{\uparrow}$ *carrier $K = (A, B, W, D, R, S, M, E, H, C, P, N)$, the local* NameCert$_{\text{HadamardProductSeries}^{\uparrow}}$ *surface consists exactly of the two source* RealSeries$^{\uparrow}$ *rows, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *coefficient-product ledger,* RegSeqRat$^{\uparrow}$ *readback, derived* RealSeries$^{\uparrow}$ *row, convergence-control row, terminal* Real$^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming through $A, B, W, D, R, S, M, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.HadamardProductSeriesUp.HadamardProductSeriesTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3811.1. The only source series coordinates are A and B, and coefficient requests can reach them only through the synchronized finite window row W.

The coefficient-product data are then forced through D. Row D records the dyadic product and tolerance budget for each displayed pair of coefficients, and row R repacks exactly those reads as RegSeqRat$^{\uparrow}$ data.

The derived series row S and convergence-control row M are downstream of that readback. The structural rows H, C, P, N transport, replay, source, and name the same route, while the carrier excludes host analytic identities, quotient series equality, selected infinite sums, countable choice, and ambient real completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 2839.3 (Hadamard product Cauchy-product handoff)**.** *Every* CauchyProduct$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RealSeries$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer of an accepted* HadamardProductSeries$^{\uparrow}$ *carrier must read the finite route*

$$A, B \longmapsto W \longmapsto D \longmapsto R \longmapsto S \longmapsto M \longmapsto E$$

before consuming the derived coefficient stream or endpoint seal. The handoff therefore supplies coefficientwise-product data as synchronized finite windows and dyadic products, not as a host multiplication theorem for completed series.

Proof. Begin with a consumer request for product-series data. The two source streams are readable only as A and B, and the matching of coefficient positions is stored only in W.

The product coefficient cannot be read before the dyadic ledger D is displayed. Its regular-rational readback is R, and the derived real-series row S is assembled only from those finite product windows.

The convergence-control row M is the unique handoff to the named CauchyProduct$^{\uparrow}$ or WeierstrassMTest$^{\uparrow}$ surface, and E is downstream of that handoff. Since H, C, P, N preserve the same route and no additional analytic coordinate is present, every public consumer factors through the displayed finite coefficientwise-product path. □

Falsifiable prediction: Every accepted $\mathsf{HadamardProductSeries}^{\uparrow}$ endpoint read factors through paired $\mathsf{RealSeries}^{\uparrow}$ coefficient rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ product ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyProduct}^{\uparrow}$ or $\mathsf{WeierstrassMTest}^{\uparrow}$ convergence control, provenance, replay, and local NameCert rows.

Independence witness: realseries, cauchyproduct, weierstrassmtest, regseqrat, streamname, dyadicratcore, real: RealSeriesUp carries single-stream series data without paired coefficient products; CauchyProductUp records convolution-style handoff without the synchronized coefficientwise ledger; WeierstrassMTestUp records majorants without paired coefficient multiplication; RegSeqRatUp, StreamNameUp, and DyadicRatCoreUp supply readback, windows, and dyadic products separately; RealUp supplies the endpoint seal without the product-series route. None is bijective to $\mathsf{HadamardProductSeries}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardProductSeries}^{\uparrow}$ packet is generated from two RealSeries coefficient streams, synchronized StreamName windows, DyadicRatCore coefficient-product ledgers, RegSeqRat readback, a derived RealSeries row, CauchyProduct or WeierstrassMTest convergence control, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3811.1, Definition 3811.2, and Definition 3811.3 seed the finite coefficientwise-product route from paired real-series coefficients to convergence control and terminal real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, quotient series equality, selected infinite sums, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window exactness, dyadic coefficient-product accounting, RegSeqRat readback, RealSeries assembly, CauchyProduct or WeierstrassMTest handoff, RealUp seal routing, and local naming for BEDC.Derived.HadamardProductSeriesUp.

Chapter 2840

A Concrete Naming Certificate for $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley metric-space terminology, the $\mathsf{LipschitzContinuityHandoff}$ packet records the textbook handoff from a displayed Lipschitz bound to a uniform continuity modulus before regular-Cauchy or Real-facing consumers read the result. It is grounded in the bound-carrying $\mathsf{LipschitzMap}^{\uparrow}$ surface of chapter 410, the metric rows of chapter 57, the graph row of chapter 130, the induced modulus row of chapter 369, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the continuity handoff consumed by L10 Real and Completion routes; it does not duplicate the upstream Lipschitz map carrier or turn that carrier into a complete downstream theorem.

Definition 2840.1 (Lipschitz continuity handoff carrier)**.** *A* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, X, Y, G, B, U, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, read through the carrier package of chapter 410.*
2. *X and Y are the source and target* $\mathsf{Met}^{\uparrow}$ *rows used by the displayed distance comparison.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row carried by the same Lipschitz packet.*
4. *B is the displayed rational bound ledger used to compare source and target distances.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row induced by the bound ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row available to regular-Cauchy consumers of the mapped readings.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after L, X, Y, G, B, U, Q have fixed the handoff.*
8. *H records componentwise* hsame *transport for the Lipschitz row, metric rows, graph, bound ledger, modulus, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a continuity request through $L, X, Y, G, B, U, Q, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted handoff packets only through $L, X, Y, G, B, U, Q, R, H, C, P, N$. It has no coordinate for host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LipschitzContinuityHandoffUp.LipschitzContinuityHandoffTasteGate single carrier alignment`

Theorem 2840.2 (Lipschitz continuity handoff NameCert obligations)**.** *For every accepted* LipschitzContinuityHand *carrier*

$$K = (L, X, Y, G, B, U, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$ *surface consists exactly of carrier exposure for the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, source and target* $\mathsf{Met}^{\uparrow}$ *rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, rational Lipschitz-bound ledger, induced* $\mathsf{UniformModulus}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3812.1. The displayed tuple contains one upstream Lipschitz row, two metric rows, one graph row, one bound ledger, one induced modulus row, one regular-rational handoff row, one Real seal row, and the structural rows H, C, P, N.

The analytic route is inherited from the upstream $\mathsf{LipschitzMap}^{\uparrow}$ carrier rather than duplicated. The source and target metric rows, graph row, and bound ledger are read from L, and the modulus row U is the displayed uniform-modulus handoff induced by that same ledger.

The downstream rows Q and R are sealed only after the finite handoff through L, X, Y, G, B, U is visible. Since the carrier has no coordinate for host function extensionality, quotient map equality, representative choice, selected limits, countable choice, differentiability, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2840.3 (Lipschitz bound to uniform modulus handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ *carrier must first read the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, source and target* $\mathsf{Met}^{\uparrow}$ *rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, rational bound ledger, and induced* $\mathsf{UniformModulus}^{\uparrow}$ *row. Only then may it consume the regular-rational handoff row or terminal Real seal row.*

Proof. Start with an accepted handoff packet and project the route in Definition 3812.1. The only entrance to the continuity handoff is the upstream Lipschitz row L, and its metric, graph, and bound coordinates are displayed as X, Y, G, B.

The uniform modulus row U is downstream of that same bound ledger. Thus a regular-Cauchy consumer receives the modulus as a finite handoff from the Lipschitz bound rather than as a selected host-function modulus.

The rows Q and R are downstream seals. A consumer that reads either row has already exposed L, X, Y, G, B, U and the structural transport and replay rows, so it cannot bypass the route by host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ packet is generated from an upstream LipschitzMap row, source and target MetricSpace rows, a ContinuousMap graph row, a rational Lipschitz-bound ledger, an induced UniformModulus row, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3812.1, Definition 3812.2, and Definition 3812.3 seed the finite continuity handoff from displayed Lipschitz bounds to uniform moduli and regular-Cauchy or Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{LipschitzContinuityHandoff}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.LipschitzContinuityHandoffUp.LipschitzContinuityHandoffTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upstream Lipschitz row projection, metric and graph exposure, bound-ledger to uniform-modulus handoff, regular-rational handoff, Real-seal discipline, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.LipschitzContinuityHandoffUp.

Chapter 2841

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ packet realizes the boundary-fixed disk transport reading of chapter 9463 and Definition 9463.10 as finite NameCert rows. It consumes the disk-coordinate carrier of chapter 377, the metric-read surface of chapter 3474, the fixed-boundary transport route of chapter 3500, the boundary-tangent horocycle row of chapter 3534, and the flow-style replay surface of chapter 195. The packet names a boundary shadow of transport on a displayed disk surface; it is not a host ideal-boundary quotient, not a physical cosmology, not a global flow, and not a completed hyperbolic surface.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ read factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, audit/refusal, provenance, replay, and local NameCert rows before it exports a boundary-shadow reading.

Independence witness: unitdisk, hyperbolic-metric, poincare-boundary-fixed-transport, hyperbolic-horocycle, dynsystem: the UnitDisk packet records disk coordinates without metric shadowing; the HyperbolicMetric packet records metric reads without a fixed-boundary transport ledger; the PoincareBoundaryFixedTransport packet preserves a boundary anchor without a horocycle shadow; the HyperbolicHorocycle packet records boundary tangency without flow-style replay; and the DynSystem packet records routed evolution without a hyperbolic boundary anchor. None is bijective to $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$.

Definition 2841.1 (Hyperbolic boundary shadow carrier)**.** *A* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, M, B, T, Q, Y, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source and transported observations.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the boundary row naming the preserved far-end boundary reading as finite certificate data.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row that transports disk points while preserving the boundary row B.*
5. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *shadow row recording the boundary-tangent trace left by the transported disk points.*

6. *Y is the* $\mathsf{DynSystem}^{\uparrow}$ *flow-style replay row, used only to order the finite transport observations.*
7. *F is the audit/refusal row: it records that boundary-shadow reads pass through D, M, B, T, Q, Y and do not expose a host ideal-boundary quotient, physical trajectory, or global hyperbolic surface.*
8. *H records componentwise* hsame *transport for disk, metric, boundary, transport, horocycle shadow, flow replay, audit/refusal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from disk observations through metric readback, boundary preservation, boundary-fixed transport, horocycle shadowing, flow-style ordering, and audit/refusal.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, PoincareBoundaryFixed
$\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B, T, Q, Y, F, H, C, P, N$. It has no coordinate for host ideal-boundary equality, unlogged observer motion, a global flow parameter, physical spacetime, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryShadowUp.HyperbolicBoundaryShadowTasteGate single carrier alignment`

Theorem 2841.2 (Hyperbolic boundary shadow NameCert obligations)**.** *For every accepted* HyperbolicBoundarySh
carrier $S = (D, M, B, T, Q, Y, F, H, C, P, N)$, the local $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric readback, boundary preservation, boundary-fixed transport, horocycle-shadow exposure, flow-style replay, audit/refusal exactness, componentwise transport, displayed continuation, package provenance, and local naming through $D, M, B, T, Q, Y, F, H, C, P, N$.*

Every boundary-shadow read from the packet factors through

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{PoincareBoundaryFixedTransport}^{\uparrow} \longmapsto \mathsf{HyperbolicHorocycle}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow}$$

together with the boundary, audit/refusal, and structural BHist, hsame, Cont, Pkg, NameCert *rows.*

Proof. Project the accepted carrier. The disk surface is row D, and the hyperbolic metric readback over that surface is row M.

The boundary information is then forced through B and T. Row B names the preserved far-end boundary reading, while row T records transport that keeps that boundary row fixed in the sense of Definition 9463.10.

The shadow-specific content is row Q, read only after the disk, metric, boundary, and fixed-transport rows have been displayed. Row Y orders the finite observations as a flow-style replay without adding a global clock.

The audit/refusal row F and the structural rows H, C, P, N exhaust the public surface. Since no carrier coordinate stores host ideal-boundary equality, unlogged observer motion, a global flow parameter, or physical spacetime, the listed rows are exactly the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ packet is generated from UnitDisk coordinates, HyperbolicMetric readback, a preserved boundary row, PoincareBoundaryFixedTransport, a HyperbolicHorocycle shadow row, DynSystem flow-style replay, audit/refusal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$, seed)

Scope closed: Definition 3813.1 and Definition 3813.2 seed the finite boundary-shadow route through disk coordinates, metric readback, preserved boundary transport, horocycle shadowing, flow-style replay, audit/refusal, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicBoundaryShadow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host ideal-boundary quotient equality, physical spacetime, unlogged observer motion, global hyperbolic flows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk placement, metric readback, boundary preservation, boundary-fixed transport, horocycle-shadow exactness, flow replay, audit/refusal coverage, and local naming for BEDC.Derived.HyperbolicBoundaryShadowUp.

Chapter 2842

A Concrete Naming Certificate for $\mathsf{LocatedUniformSpace}^{\uparrow}$

Following Nachbin, Kelley, and Bishop-style located uniform structures, the $\mathsf{LocatedUniformSpace}^{\uparrow}$ packet records directed entourage data together with a finite locatedness ledger for deciding when a displayed point, filter, or net read has entered a named uniform neighbourhood. It sits between the directed-uniform carrier of chapter 2334, the symmetric uniform interface of chapter 339, the filter route of chapter 333, and the net route of chapter 589. Its constructive readback uses the finite window and rational ledgers of chapter 126, chapter 291, chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 2842.1 (LocatedUniformSpace carrier)**.** *A* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (U, D, A, W, T, F, N, H, C, P, Q)$$

with the following displayed rows:

1. *U is the directed entourage row, read through the* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *naming surfaces.*
2. *D is the finite directedness ledger, recording refinement, composition, and symmetry requests only as displayed entourage rows.*
3. *A is the located apartness row, recording when a point or finite window is apart from the complement of a displayed entourage.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row for finite point, net, and filter reads.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to measure the displayed locatedness request.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *handoff row.*
7. *N is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rationally named tolerance witnesses.*
8. *H is the componentwise* hsame *transport row.*
9. *C is the displayed* Cont *replay row.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformSpace}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$ *row.*

No carrier coordinate stores a completed uniform space, a quotient filter, a selected limit, or a host locatedness decision.

Lean4 ✓ `BEDC.Derived.LocatedUniformSpaceUp.LocatedUniformSpaceTasteGate single carrier alignment`

Theorem 2842.2 (LocatedUniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier* $L = (U, D, A, W, T, F, N, H, C, P, Q)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, directed entourage access through* U, D*, located apartness access through* A*, finite observation windows through* W*, dyadic tolerance access through* T*, Cauchy filter and Cauchy net handoff through* F*, regular rational readback through* N*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* Q*.*

Proof. Project the carrier of Definition 3814.1. The entourage, directedness, locatedness, window, tolerance, Cauchy, and rational-readback rows are exactly U, D, A, W, T, F, N.

The structural rows H, C, P, Q transport, replay, source, and name the same finite route. Since the carrier has no further coordinate, a consumer cannot replace this route by a host uniform space, a quotient filter, a selected limit, or an undecorated locatedness predicate. □

Theorem 2842.3 (LocatedUniformSpace Cauchy-filter route)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *consumer read accepted by a* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier factors through*

$$U, D, A \longmapsto W, T \longmapsto F \longmapsto N, H, C, P, Q.$$

The Cauchy handoff may inspect only the displayed directed entourage ledger, located apartness row, finite observation windows, dyadic tolerance ledger, and regular rational readback.

Proof. Start with a Cauchy-facing consumer. The carrier exposes Cauchy data only at row F, so the consumer must first read the entourage and locatedness rows that determine which finite windows are admissible.

The rows W and T then display the finite observations and dyadic tolerances used by the Cauchy request. The regular readback row N records the rational witness visible to the consumer.

Transport, replay, provenance, and local naming are supplied only by H, C, P, Q. Thus the route is finite and excludes quotient filters, selected subnet limits, and ambient completeness. □

Theorem 2842.4 (LocatedUniformSpace entourage-location obligation)**.** *For every accepted* $\mathsf{LocatedUniformSpace}$ *carrier*

$$L = (U, D, A, W, T, F, N, H, C, P, Q),$$

each located entourage request must expose the directed entourage row U*, directedness ledger* D*, located apartness row* A*, finite observation-window row* W*, and dyadic tolerance row* T *before any* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *handoff is available. The request may not enter* F *from an unnamed entourage, a host locatedness predicate, a quotient-filter class, a selected limit, or an ambient completeness principle.*

Proof. Project the carrier rows of Definition 3814.1. The only coordinates that can answer an entourage-location request before Cauchy handoff are U, D, A, W, T.

The directedness ledger D refines requests only as displayed entourage rows over U, while A records located apartness only for the displayed point or window. The rows W and T then provide the finite observation and dyadic tolerance data needed to state the request.

Since F is downstream of these five rows, a Cauchy consumer can be admitted only after the located entourage request has passed through them. No excluded host object is a coordinate of L, so none can supply a second route into F. □

Theorem 2842.5 (LocatedUniformSpace filter-net handoff exactness)**.** *For every accepted* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier*

$$L = (U, D, A, W, T, F, N, H, C, P, Q),$$

every $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *handoff through* F *is exact only along the finite route*

$$U, D, A, W, T \longmapsto F \longmapsto N, H, C, P, Q.$$

The packet therefore exports Cauchy-facing data as the displayed located-uniform handoff, not as quotient-filter equality, a selected subnet, a completed uniform-space theorem, separated reflection, or host locatedness.

Proof. Begin with the handoff row F. By Definition 3814.3, a Cauchy-facing read reaches F only after the directed entourage, locatedness, finite-window, and tolerance rows have been read.

Apply the entourage-location obligation. It prevents F from being entered by an unnamed entourage or external locatedness decision, so the handoff is tied to the displayed U, D, A, W, T rows.

Finally read N, H, C, P, Q. These rows supply rational readback, componentwise transport, displayed replay, provenance, and local naming for the same route. Because the carrier has no coordinate for quotient filters, selected subnet limits, completion, separated reflection, or host locatedness, the displayed route is the exact handoff surface. □

Theorem 2842.6 (LocatedUniformSpace real-completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *readback exported by a* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *packet is downstream of the displayed route through* U, D, A, W, T, F, N *and the structural rows* H, C, P, Q*. The packet can feed completion-facing chapters only by exposing the finite located uniform data; it does not assert that the surrounding uniform space is complete, separated, precompact, or represented by chosen Cauchy names.*

Proof. Read the completion-facing clause of the carrier. A real-facing consumer reaches rational witnesses only through N, and N is downstream of the displayed window and tolerance route.

The package row P records the relationship with $\mathsf{Real}^{\uparrow}$ and the uniform siblings, while Q names only this local packet. No row remains for a host completion theorem, a selected Cauchy representative, a quotient construction, or a global precompactness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformSpace}^{\uparrow}$ packet is generated from directed entourage data, located apartness, finite observation windows, dyadic tolerance rows, Cauchy filter and net handoff, regular rational readback, componentwise transport, displayed continuation, package provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$, obligation)

Scope closed: Definition 3814.1, Definition 3814.2, Definition 3814.3, Definition 3814.4, Definition 3814.5, and Definition 3814.6 bind the finite located directed-uniform certificate surface and its Cauchy filter/net obligation route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close completion, precompactness, quotient-filter equality, selected limits, separated reflection, or bridge checking for $\mathsf{LocatedUniformSpace}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, directed-entourage routing, locatedness exactness, Cauchy handoff, Real readback boundary, transport stability, replay stability, and ledger non-escape for BEDC.Derived.LocatedUniformSpa

Chapter 2843

A Concrete Naming Certificate for $\mathsf{ClosedIntervalBisection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedIntervalBisection}^{\uparrow}$ packet records constructive interval bisection over a located closed bounded interval by finite endpoint, midpoint, branch, and readback rows. It refines the closed bounded interval carrier of chapter 1591, the located bisection route of chapter 1711, the explicit interval split of chapter 1465, and the nested-route consumer of chapter 1713. Its completion-facing rows pass through chapter 126, chapter 301, and chapter 38. The packet supplies a closed-interval bisection surface for L10 Real and Completion arguments without importing open-cover compactness, real trichotomy, quotient real equality, selected infinite branches, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2843.1 (Closed interval bisection carrier)**.** *A* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal real-name seal of the source interval.*
2. *L is the located endpoint row. It records the endpoint-location evidence needed to decide which child interval is retained for the current finite request, without adding a trichotomy or host real comparison coordinate.*
3. *M is the midpoint row. It is computed from the displayed endpoint cells of I and is read before the branch row is inspected.*
4. *D is the dyadic branch row. It records the left or right retained child, the parent-to-child containment witness, and the dyadic diameter budget for the current bisection request.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the parent endpoints, midpoint read, retained child endpoints, and branch ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and dyadic branch data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, M, D, S, R have been displayed.*

8. *H records componentwise* hsame *transport for the closed interval, located endpoint, midpoint, dyadic branch, finite windows, regular rational readback, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a closed-interval bisection request through I, L, M, D, S, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, M, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, hidden endpoint choice, selected infinite branches, real trichotomy, quotient stream equality, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2843.2 (Closed interval bisection NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier $B = (I, L, M, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, closed-bounded interval exposure through I, located endpoint exposure through L, midpoint formation through M, dyadic branch and child-containment exposure through D, finite-window exposure through S, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3815.1. The row I supplies the only closed bounded interval source, while L supplies the only located endpoint evidence available to the branch request.

The midpoint and branch obligations are consecutive. The row M is read from the displayed endpoint cells of I, and the dyadic branch row D records the retained child interval only after that midpoint row is visible.

The completion-facing obligations are downstream of the branch. The finite windows S and regular rational readback R consume the parent endpoints, midpoint, and retained child rows fixed by I, L, M, D, and the real seal E is reached only after that readback route is displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded host objects occurs as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2843.3 (Closed interval bisection branch nonescape)**.** *Every retained-branch read of an accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto L, M, D \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the retained child interval is read from displayed closed-interval endpoints, located endpoint evidence, the midpoint row, and the dyadic branch ledger; it is not supplied by a hidden branch selector, a real trichotomy oracle, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalBisectionUp.ClosedIntervalBisection branch nonescape route`

Proof. Start with the source row I. It fixes the closed bounded interval whose lower and upper endpoint data can be inspected by the branch request.

Next read L and M. The located endpoint row supplies only the displayed endpoint-location evidence for the current finite request, and the midpoint row supplies the child boundary computed from the same endpoint cells.

The dyadic branch row D is the only retained-child coordinate. It records the left or right child, the containment relation to the parent interval, and the dyadic diameter budget before any StreamName window is exposed.

Finally S, R, E, H, C, P, N read the retained child through finite windows, regular rational readback, real sealing, transport, replay, provenance, and local naming. No hidden selector, trichotomy oracle, compactness principle, quotient equality, choice schedule, or ambient completeness row is available in the carrier. □

Theorem 2843.4 (Closed interval bisection completion handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *nested-interval, or compactness-facing consumer of an accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *packet must first read the closed source interval, located endpoint row, midpoint row, and dyadic branch row. The packet hands downstream consumers one displayed retained child route with transport, replay, provenance, and local naming; it does not hand them open-cover compactness, a completed branch, a selected real point, quotient equality of real names, or an ambient completeness theorem.*

Proof. Project the carrier $B = (I, L, M, D, S, R, E, H, C, P, N)$. The first four rows I, L, M, D are the complete interval-bisection route: source interval, located endpoint evidence, midpoint, and retained dyadic branch.

The readback rows are ordered after that route. A $\mathsf{StreamName}^{\uparrow}$ consumer reads only the finite windows in S, a $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads only the regular rational row R, and a $\mathsf{Real}^{\uparrow}$ consumer reaches the seal E only after the same branch data has been displayed.

Nested-interval and compactness-facing consumers receive the retained child through H, C, P, N. These structural rows preserve and replay the same packet; they do not introduce completed branches, selected points, quotient equality, open-cover compactness, or ambient completeness. □

Theorem 2843.5 (Closed interval bisection interval-modulus termination)**.** *An accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier with a displayed unary history depth row n exports a finite dyadic interval window at that depth. Its endpoint route is inherited from the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row I, its branch row is inherited from the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *and* $\mathsf{IntervalBisection}^{\uparrow}$ *rows L, M, D, and its* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal handoff remains inside the displayed rows S, R, E, H, C, P, N. This is a finite modulus and termination row for the displayed depth, not open-cover compactness, trichotomy, a selected infinite branch, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed unary depth row. At depth zero, read the original closed bounded interval from I; its endpoint order, containment, finite window, regular readback, and real-name seal are exactly the source data already exposed by Definition 3815.1.

For the successor depth, apply the midpoint and retained-branch rows M, D. The located endpoint data in L chooses only the displayed finite child, while D transports endpoint order, containment, and the dyadic diameter budget from the parent to that child.

Componentwise hsame transport in H and displayed replay in C keep the induction inside the same carrier. The completion handoff of Definition 3815.4 then composes S, R, E after the current branch row, so the depth-n window terminates as a finite row rather than as a hidden infinite branch or completeness principle. □

Theorem 2843.6 (Closed interval bisection RegSeqRat route)**.** *Every accepted retained finite bisection window reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row only after reading the closed source interval, located*

endpoint evidence, midpoint row, dyadic retained-branch ledger, and $\mathsf{StreamName}^{\uparrow}$ *finite windows. For* $B = (I, L, M, D, S, R, E, H, C, P, N)$*, the regular rational readback factors through*

$$I, L, M, D, S \longmapsto R \longmapsto E, H, C, P, N.$$

It does not use open-cover compactness, a selected infinite branch, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3815.1. The row I supplies the closed source interval, L supplies the located endpoint evidence, M supplies the midpoint row, and D supplies the dyadic retained-branch ledger.

Apply Definition 3815.2 to identify S as the finite-window exposure and R as the regular rational readback row. The retained-branch route of Definition 3815.3 orders the source, located endpoint, midpoint, branch, stream, regular rational, and real rows in that sequence.

The depth window of Definition 3815.5 keeps this read finite at the displayed unary depth. Therefore a $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives only the stated finite route.

No carrier coordinate names open-cover compactness, a selected infinite branch, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of those principles can mediate the readback. □

Falsifiable prediction: Every $\mathsf{ClosedIntervalBisection}^{\uparrow}$ consumer that omits the dyadic branch row loses the retained child interval before any RegSeqRat or Real seal can be read.

Independence witness: closedboundedinterval, located-interval-bisection, intervalbisection, located-real-nested-intervals, streamname, regseqrat, real: $\mathsf{ClosedIntervalBisection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a closed bounded interval source, located endpoint evidence, midpoint formation, a dyadic retained-branch ledger, finite windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalBisection}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, located endpoint evidence, a midpoint row, a dyadic retained-branch row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3815.1, Definition 3815.2, Definition 3815.3, and Definition 3815.4 seed the constructive closed-interval bisection route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, real trichotomy, quotient real equality, selected infinite branches, selected real points, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, closed-interval exposure, located endpoint exposure, midpoint formation, dyadic branch exactness, StreamName window exposure, RegSeqRat readback, and Real seal nonescape for BEDC.Derived.ClosedIntervalBisectionUp.

Chapter 2844

A Concrete Naming Certificate for $\mathsf{RealSpectrum}^{\uparrow}$

Following the classical Banach-algebra spectrum tradition of Gelfand theory, the $\mathsf{RealSpectrum}^{\uparrow}$ packet records the finite BEDC surface by which a real Banach-algebra element is tested against displayed real scalar windows, resolvent-boundary rows, and spectral-radius estimates without importing a host spectrum. It is a spectral neighbour of the L10 real and completion frontier: it reads the complete normed algebra surface of chapter 2189, the power-growth boundary of chapter 2352, the bounded-operator route of chapter 1644, the real seal of chapter 38, the regular-rational classifier of chapter 301, and the finite stream windows of chapter 126.

Definition 2844.1 (Real spectrum carrier)**.** *A* $\mathsf{RealSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, a, U, R, B, W, Q, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row carrying multiplication, norm, completion, product control, transport, replay, provenance, and local naming.*
2. *a is the carried algebra element whose real spectral behaviour is inspected.*
3. *U is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *or algebra-action row used when the element is read through a bounded operator surface.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row. It admits only displayed real scalars, not a completed host scalar field.*
5. *B is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for real scalar windows. It records the regular rational classifier data used to compare a scalar probe with the element.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *spectral-window observation row. It names the finite list of scalar probes, operator-power requests, and resolvent-boundary tests that are actually displayed.*
7. *Q is the resolvent-boundary test row. For each displayed scalar window it records the local multiplication and inverse-check surface, together with a refusal row when the inverse evidence is not displayed.*
8. *E is the* $\mathsf{SpectralRadius}^{\uparrow}$ *estimate row. It reads only the power-window and norm-growth estimates supplied by the spectral-radius packet.*

9. *G is the real-spectrum readback row. It classifies a displayed scalar as admitted, refused, or boundary-facing only through R, B, W, Q, E.*

10. *H records componentwise* hsame *transport for the Banach-algebra source, carried element, bounded-operator row, real seal, regular-rational readback, stream windows, resolvent tests, spectral-radius estimates, readback, and structural rows.*

11. *C records displayed* Cont *replay of the route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SpectralRadius}^{\uparrow} \longmapsto \mathsf{RealSpectrum}^{\uparrow}.$$

12. *P is the* Pkg *provenance over* $\mathsf{RealSpectrum}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, a, U, R, B, W, Q, E, G, H, C, P, N$ and has no coordinate for a host spectrum, maximal ideals, Gelfand transform, spectral theorem, quotient completion, selected approximate eigenvectors, trichotomy over real scalars, or countable choice.

Lean4 ✓ `BEDC.Derived.RealSpectrumUp.RealSpectrumTasteGate single carrier alignment`

Theorem 2844.2 (Real spectrum NameCert obligations)**.** *For an accepted* $\mathsf{RealSpectrum}^{\uparrow}$ *carrier*

$$S = (A, a, U, R, B, W, Q, E, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Banach-algebra source, carried element, bounded-operator read, real seal, regular-rational readback, stream-window observations, resolvent-boundary tests, spectral-radius estimates, readback, transport, replay, provenance, and local naming row;*

2. *scalar-window compatibility, requiring every scalar probe in W to read its* $\mathsf{Real}^{\uparrow}$ *seal through R and its* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier through B before the probe can enter a resolvent test;*

3. *resolvent-boundary locality, requiring Q to display the multiplication and inverse-check rows used by each accepted scalar test and to record refusal when those rows are absent;*

4. *spectral-radius estimate discipline, requiring E to use only the power-window and norm-growth rows already certified by* $\mathsf{SpectralRadius}^{\uparrow}$*;*

5. *classifier transport, requiring H and C to preserve the scalar windows, resolvent tests, radius estimates, real seal, regular-rational readback, bounded-operator row, provenance, and local naming componentwise.*

No NameCert obligation is stored outside these rows. In particular, the packet exports no completed host spectrum, no maximal ideal space, no Gelfand-transform equality, no spectral theorem, no quotient completion, no selected approximate eigenvector, no trichotomy row, and no countable-choice schedule.

Proof. Carrier admission is projection from Definition 3816.1. The accepted packet exposes exactly the Banach-algebra source, carried element, bounded-operator read, real seal, regular-rational classifier, stream-window observations, resolvent tests, spectral-radius estimates, readback, transport, replay, provenance, and local naming rows.

Scalar-window compatibility uses R, B, W. Each scalar probe is first a displayed stream-window request, then a regular-rational readback, and only then a real-seal read. Thus a scalar cannot enter the spectrum packet as an ambient real number detached from the displayed L10 rows.

Resolvent locality is the row Q. It names the local multiplication and inverse-check evidence used by each displayed test, and it records refusal when the evidence is absent. The row therefore supplies a boundary test rather than a host inverse or a completed spectrum.

The radius estimate discipline is the row E. It imports only the finite power-window and norm-growth surface already exposed by $\mathsf{SpectralRadius}^{\uparrow}$, so no limit over all powers or selected approximate eigenvector is introduced.

Finally H, C, P, N transport, replay, source, and name only the listed coordinates. Since the excluded spectral and choice-theoretic objects have no coordinate in the carrier, they cannot appear as local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealSpectrumUp.RealSpectrumCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSpectrum}^{\uparrow}$ packet is generated from a BanachAlgebraUp source, a carried element, a BoundedLinearOperatorUp read, RealUp scalar seals, RegSeqRatUp readback rows, StreamNameUp spectral windows, resolvent-boundary tests, SpectralRadiusUp estimates, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3816.1 and Definition 3816.2 seed the finite real-spectrum route over BanachAlgebraUp, SpectralRadiusUp, BoundedLinearOperatorUp, RealUp, RegSeqRatUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host spectrum, maximal ideal space, Gelfand representation, spectral theorem, quotient completion, selected approximate eigenvectors, real trichotomy, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-window compatibility, resolvent-boundary locality, spectral-radius estimate discipline, classifier transport, and consumer non-escape for BEDC.Derived.RealSpectrumUp.

Chapter 2845

A Concrete Naming Certificate for $\mathsf{BanachLattice}^{\uparrow}$

Following Kantorovich and Schaefer on Banach lattices, the $\mathsf{BanachLattice}^{\uparrow}$ packet records the finite BEDC naming surface for a vector lattice whose normed carrier is Cauchy-complete and whose positive cone remains visible to norm-facing consumers. It strengthens the ordered vector-lattice surface of chapter 3620 by adding the Banach-space completeness rows of chapter 90, while still reading the norm row of chapter 89, the join-meet lattice row of chapter 55, and the vector-space carrier of chapter 47. The packet supplies finite vector-lattice, norm, positive-cone, and Cauchy-completeness rows; it does not supply Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, or a host order-completion principle.

Definition 2845.1 (BanachLattice carrier)**.** *A* $\mathsf{BanachLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, N, P, M, H, C, S)$$

with the following displayed rows:

1. *R is the* $\mathsf{RieszSpace}^{\uparrow}$ *row, carrying the common vector endpoint, real scalar endpoint, lattice join and meet operations, positive and negative part ledgers, and vector-lattice classifier.*
2. *A is the* $\mathsf{Banach}^{\uparrow}$ *row over the same vector endpoint and normed metric surface, exposing the Cauchy-completeness witness through finite regular-Cauchy and limit-seal rows.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *compatibility row tying the norm used by A to the lattice decomposition and positive-cone rows visible in R.*
4. *P is the positive-cone monotonicity row. It records that positive part, negative part, cone membership, and order comparison are read before norm or completeness consumers inspect the packet.*
5. *M is the Cauchy-completeness ledger. It records the finite regular-Cauchy windows, modulus or metric-control rows, and terminal Banach seal used by downstream complete normed-lattice consumers.*
6. *H records componentwise* hsame *transport for the Riesz-space row, Banach row, norm compatibility row, positive-cone row, completeness ledger, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Banach-lattice request through R, A, N, P, M, H and back to the same finite ordered complete-normed surface.*

8. *S is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{BanachLattice}^\uparrow}$ *row over* $\mathsf{BanachLattice}^\uparrow$, $\mathsf{RieszSpace}^\uparrow$, $\mathsf{Banach}^\uparrow$, $\mathsf{Norm}^\uparrow$, $\mathsf{Lattice}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, BHist, hsame, Cont, *and* NameCert.

The classifier compares accepted packets only through these displayed rows. It has no coordinate for Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, pointwise function models, countable choice, or ambient host order equality.

Lean4 ✓ `BEDC.Derived.BanachLatticeUp.BanachLatticeTasteGate single carrier alignment`

Theorem 2845.2 (BanachLattice NameCert obligations)**.** *For every accepted* $\mathsf{BanachLattice}^\uparrow$ *carrier* $B = (R, A, N, P, M, H, C, S)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachLattice}^\uparrow}$ *surface consists exactly of carrier admission, vector-lattice row compatibility, norm compatibility, positive-cone exposure, Cauchy-completeness ledger exposure, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A downstream consumer may read the common vector endpoint, the lattice join and meet rows, the norm row, the positive-cone ledger, and the finite Banach completeness route, but it receives no Hahn-Banach functional, no Dedekind-complete lattice envelope, no selected supremum, and no quotient function-space representation.*

Proof. Carrier admission is projection from Definition 3817.1. The tuple displays exactly the Riesz-space row R, Banach row A, norm row N, positive-cone row P, completeness ledger M, and structural rows H, C, S.

The vector-lattice obligations are read through R. This fixes the vector endpoint, scalar endpoint, join and meet operations, order comparisons, and positive and negative part ledgers before any Banach-facing row is consumed.

The norm and completeness obligations are finite rows. The norm row N ties the Banach metric used by A to the lattice decomposition data in R, and the ledger M records the regular-Cauchy windows and terminal completeness seal exposed by the Banach row.

Finally H, C, S transport, replay, source, and name only those displayed coordinates. Since the carrier contains no coordinate for Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, or host order equality, none of those objects can be exported by the local certificate. □

Theorem 2845.3 (BanachLattice classifier stability)**.** *If two accepted* $\mathsf{BanachLattice}^\uparrow$ *packets are related by the classifier, then their Riesz-space rows, Banach completeness rows, norm compatibility rows, positive-cone rows, and Cauchy-completeness ledgers are related by the componentwise* hsame *transport recorded in H. Consequently every norm-facing, order-facing, or completeness-facing read factors through the displayed finite ledger and never through a host order equality, selected supremum, or hidden quotient carrier.*

Proof. Unfold the classifier of Definition 3817.1. Its comparison coordinates are exactly R, A, N, P, M, H, C, S.

The Riesz-space component transports the vector, lattice, and positive-part rows. The Banach component transports the normed metric and Cauchy-completeness rows over the same carried endpoint.

The norm compatibility and positive-cone rows then move along those transported coordinates. The completeness ledger M remains attached to the transported Banach row, so no accepted read can replace it by a selected supremum, quotient function-space equality, or host order construction. □

Theorem 2845.4 (BanachLattice Riesz-Banach handoff)**.** *Every accepted* $\mathsf{BanachLattice}^\uparrow$ *packet supplies a finite handoff*

$$\mathsf{RieszSpace}^\uparrow, \quad \mathsf{Banach}^\uparrow, \quad \mathsf{Norm}^\uparrow, \quad \mathsf{Lattice}^\uparrow, \quad \mathsf{VecSpace}^\uparrow$$

in which vector-lattice and positive-cone data are read before the Banach completeness ledger is consumed. The handoff exports the ordered complete normed carrier as NameCert data only through R, A, N, P, M, H, C, S.

Proof. Read the handoff from the carrier. The row R supplies the $\mathsf{RieszSpace}^{\uparrow}$ surface over the vector-space and lattice rows, while N repeats the norm compatibility needed by the Banach-facing route.

Next read A and M. These rows expose the Cauchy-complete normed carrier and the finite regular-Cauchy ledger without adding a supremum selector, Hahn-Banach functional, or quotient function model.

The positive-cone row P binds the order-facing part of the route to the same vector endpoint before H, C, S transport, replay, source, and name it. Thus the Riesz-space and Banach-space surfaces meet only through the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachLattice}^{\uparrow}$ packet is generated from a Riesz-space row, a Banach completeness row, norm compatibility data, positive-cone data, a Cauchy-completeness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3817.1, Definition 3817.2, Definition 3817.3, and Definition 3817.4 seed the finite ordered complete normed-lattice certificate over RieszSpaceUp, BanachUp, NormUp, LatticeUp, VecSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vector-lattice compatibility, norm compatibility, positive-cone exposure, Cauchy-completeness ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.BanachLatticeUp.

The $\mathsf{HeronSquareRoot}^{\uparrow}$ hub routes Newton, dyadic bisection, reciprocal, and regular-Cauchy square-root rows through chapter 1756, chapter 291, and chapter 301 before its child files expose the Heron packet.

Chapter 2846

A Concrete Naming Certificate for $\mathsf{HeronSquareRoot}^{\uparrow}$

From Heron's classical square-root iteration and Bishop-style constructive convergence, the $\mathsf{HeronSquareRoot}^{\uparrow}$ packet records the finite dyadic evidence needed to run the Babylonian recurrence inside the L10 Real and Completion route. It refines the square-root service of chapter 1756 by exposing seed positivity, reciprocal accounting, contraction estimates, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal as named rows. The dependency route passes through chapter 1756, chapter 291, chapter 223, chapter 301, and chapter 38; it is not a host square-root function, quotient-real equality, a selected completed limit, or an ambient completeness theorem.

The carrier consumes $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows from chapter 291 before every reciprocal, halving, and squared-error comparison, and it consumes $\mathsf{RegSeqRat}^{\uparrow}$ readbacks from chapter 301 before exposing the terminal $\mathsf{Real}^{\uparrow}$ square-root seal. The Heron recurrence also consumes the guarded dyadic Newton square-root packet of chapter 7196 before exporting the RegSeqRat and Real seal rows. Thus the Heron route is a dependency-bearing finite lattice: dyadic tolerance rows govern the visible recurrence windows, regular rational readbacks certify the Cauchy handoff, and only then may a Real-facing consumer inspect the seal row.

Definition 2846.1 (Heron square-root carrier)**.** *A* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *A is the positive dyadic radicand row, including lower-bound and nonzero-denominator witnesses inside* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *S is the finite positive dyadic seed row from which the Heron recurrence starts.*
3. *R is the reciprocal and division ledger row, recording the displayed divisions A/x_i and every denominator witness used by the recurrence.*
4. *L is the finite Heron ledger whose rows have the form $x_{i+1} = (x_i + A/x_i)/2$ and whose arithmetic is read only through A, S, R and* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *W is the finite iteration-window row, exposing exactly the initial segment of L requested by a precision consumer.*
6. *D is the dyadic contraction-estimate row, recording the squared-error, halving, and tolerance comparisons that bound the visible window.*

7. *K is the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row showing that the displayed recurrence is the specialized Newton step for $x^2 - A$ without importing a host root.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from A, S, R, L, W, D, K.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular readback row Q has been displayed.*
10. *T records componentwise* hsame *transport for the radicand, seed, reciprocal ledger, Heron ledger, window, contraction estimate, Newton provenance, readback, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of each precision request through $A, S, R, L, W, D, K, Q, E, T, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{HeronSquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host square-root operation, quotient-real equality, selected limits, arbitrary real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2846.2 (Heron square-root regular-Cauchy handoff)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite dyadic error ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal Real seal displayed in*

$$A, S \longmapsto R, L, W, D, K \longmapsto Q \longmapsto E.$$

Thus a Real or Completion consumer first sees the positive radicand and seed rows, then the reciprocal and division ledger, Heron recurrence window, dyadic contraction estimates, and $\mathsf{NewtonIteration}^{\uparrow}$ *provenance, and only then may read the regular rational name and* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no host square-root function, quotient-real equality, selected completed limit, arbitrary real trichotomy, or ambient completeness object.*

Proof. Start with an accepted carrier $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$. The carrier inventory exposes the positive dyadic radicand A and seed S before any recurrence row can be inspected.

The numerical ledger is finite. The reciprocal row R stores the displayed divisions A/x_i, the Heron ledger L stores the visible recurrence rows, the window row W selects the requested finite segment, and the contraction row D stores the dyadic error comparisons for that segment.

The route then records provenance through K. This row identifies the same displayed recurrence as the Newton step for $x^2 - A$ through $\mathsf{NewtonIteration}^{\uparrow}$, but it contributes only finite ledger provenance and no external square-root value.

The consumer-facing read is downstream of those rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q is produced by replaying the finite dyadic ledger, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after Q has been displayed.

The remaining rows T, C, P, N transport, replay, record provenance, and name the same packet. Since the carrier contains no host square-root coordinate, quotient-real equality, selected limit, trichotomy classifier, or ambient completeness row, every accepted Real-facing read factors through the displayed route. □

Theorem 2846.3 (Heron square-root finite-window contraction induction)**.** *Let $H = (A, S, R, L, W, D, K, Q, E, T,$ be an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier. Every finite window W of the Heron ledger L preserves the dyadic contraction estimate recorded by D by induction on the displayed window length. In the successor step the proof reads the reciprocal row R, the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row K,*

and the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *are inspected. The induction exports no host square-root function, selected limit, trichotomy, quotient-real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The finite rows A, S, R, L, W, D, K are available before any regular rational readback or Real seal, and the handoff theorem places Q and E downstream of those rows.

Induct on the displayed length of the window row W. The empty window has no Heron step to check, so the contraction estimate is exactly the empty restriction of D.

For the successor window, read the previous visible window from L and use the induction hypothesis for its restriction of D. The new row of L is the displayed recurrence $x_{i+1} = (x_i + A/x_i)/2$; its division data is supplied by R, its recurrence provenance is supplied by K, and each tolerance comparison is read from the $\mathsf{DyadicRatCore}^{\uparrow}$ rows already recorded in D.

After the successor estimate is recorded, apply Definition 3818.2. The $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and $\mathsf{Real}^{\uparrow}$ seal E are therefore downstream of the finite induction and cannot change the displayed contraction window.

Rows T, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate contains a host square-root function, selected completed limit, trichotomy, quotient-real equality, or ambient completeness theorem, the finite-window induction remains an internal Heron-ledger statement. □

Theorem 2846.4 (Heron square-root regular-Cauchy RegSeqRat lattice)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing square-root consumer of an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

must first read the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *sibling route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radicand and arithmetic rows, the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then the* $\mathsf{Real}^{\uparrow}$ *seal. In carrier notation, the lattice route projects* $A, S, R, L, W, D, K, Q, E, T, C, P, N$ *in that order and exposes no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the accepted carrier. The rows A and S are the positive dyadic radicand and seed rows, while R, L, W, D are the reciprocal ledger, Heron recurrence ledger, finite window, and dyadic contraction estimate.

Apply Definition 3818.2. It places every Real or Completion read downstream of the finite dyadic ledger and before the regular rational readback Q and terminal seal E.

Read the sibling lattice through provenance. The row K records the $\mathsf{NewtonIteration}^{\uparrow}$ specialization, P records the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependency route, and Q is the regular rational handoff generated from the displayed rows.

Finally transport and name the route through T, C, P, N. Since all consumer-facing data is already exhausted by $A, S, R, L, W, D, K, Q, E, T, C, P, N$, no Real-facing consumer can replace the lattice by a host square-root operation, quotient equality, selected limit, trichotomy, or ambient completeness. □

Theorem 2846.5 (Heron square-root NameCert obligations)**.** *For every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *surface consists exactly of the displayed carrier admission row, positive dyadic radicand and seed exactness, reciprocal and division ledger exactness, finite Heron-window stability, dyadic contraction accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The coordinates A and S give the positive dyadic radicand and seed rows, while R stores the reciprocal and division witnesses used by the recurrence.

The rows L, W, D, K form the finite numerical spine. The Heron ledger L records the displayed recurrence, W selects the finite window, D records the dyadic contraction estimates on that window, and K identifies the same row as the specialized $\mathsf{NewtonIteration}^{\uparrow}$ step for $x^2 - A$.

Apply Definition 3818.2 to the consumer-facing side. The regular readback Q is available only after the finite dyadic ledger has been replayed, and the $\mathsf{Real}^{\uparrow}$ seal E is read only after Q.

The rows T, C, P, N then transport, replay, source, and locally name the same packet. Since the carrier has no row for a host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 2846.6 (Heron square-root contraction-window stability)**.** *Let* $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N$ *be an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier. Any componentwise* hsame *transport recorded by* T *preserves the finite Heron window* W *and the same dyadic contraction estimate recorded by* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is inspected.*

Proof. Read the source window first. The window row W is a finite initial segment of the displayed Heron ledger L, and the contraction row D records the dyadic comparisons for exactly that segment.

Transport acts componentwise through T. Hence a transported carrier must carry corresponding A, S, R, L, W, D, K rows before any consumer can inspect Q or E.

The handoff theorem separates the finite dyadic ledger from the regular-Cauchy readback. Therefore the contraction estimate seen on the transported window is the same estimate certified by D, and no later $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row can alter the finite Heron-window bound. □

Theorem 2846.7 (Heron square-root regular-Cauchy sibling route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read factors through the sibling chain from chapter 1756 to chapter 291, chapter 223, chapter 301, and chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing square-root consumer receives the result. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the route first projects the positive dyadic radicand and seed rows A, S*, then the reciprocal ledger, Heron recurrence, window, contraction estimate, and Newton provenance rows* R, L, W, D, K*, then the regular readback row* Q*, and finally the Real seal* E *together with transport, replay, provenance, and local naming through* T, C, P, N*.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The first seven rows are exactly the finite dyadic and Newton-style data named by the sibling chapters: radicand and seed in $\mathsf{DyadicRatCore}^{\uparrow}$, recurrence provenance in $\mathsf{NewtonIteration}^{\uparrow}$, and a visible Heron window with contraction accounting.

Apply Definition 3818.2. It forces every Real or Completion consumer to pass from A, S through R, L, W, D, K before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and the terminal $\mathsf{Real}^{\uparrow}$ seal E are visible.

Use Definition 3818.6. The finite window and contraction estimate survive componentwise transport, so the sibling route cannot be replaced by a host square-root operation, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.

Thus the accepted read is a regular-Cauchy square-root route: it moves through the displayed dyadic, Newton, regular rational, and Real siblings before any Real-facing consumer can use the square-root result. □

Theorem 2846.8 (Heron square-root dyadic-Real forward link)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer whose output is read as a* $\mathsf{Real}^{\uparrow}$ *result must first consume the dyadic radicand and contraction rows from chapter 291, the Newton provenance row from chapter 223, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal from chapter 38. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the forward dependency order is A, S, R, L, W, D *before* K*,* K *before* Q*, and* Q *before* E*; the transport, replay, provenance, and local naming rows* T, C, P, N *preserve that order for every accepted Real-facing read.*

Proof. Project the carrier. The dyadic radicand and seed rows A, S, reciprocal ledger R, Heron ledger L, finite window W, and contraction row D are visible before any row with a Real-facing type is inspected.

Use the sibling route of Definition 3818.7. It places the Newton provenance row K downstream of the dyadic Heron rows, then places $\mathsf{RegSeqRat}^{\uparrow}$ readback Q downstream of K, and finally exposes the $\mathsf{Real}^{\uparrow}$ seal E only after Q.

The structural rows T, C, P, N cannot permute this order: they transport componentwise, replay the displayed route, record provenance, and name the same carrier. Hence a Real-facing consumer must follow the DyadicRatCore, NewtonIteration, RegSeqRat, and Real sibling route rather than reading a host square-root function, quotient-real equality, selected limit, or ambient completeness witness. □

Theorem 2846.9 (Heron square-root dyadic refinement route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

first refines the positive dyadic radicand and visible recurrence window through the $\mathsf{DyadicRatCore}^{\uparrow}$ *and interval-refinement tolerance rows before Newton provenance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are visible. The route reads the dyadic interval-refinement surface of chapter 1831 as part of the* A, S, R, L, W, D *window discipline, then exposes* K, Q, E, T, C, P, N *only as downstream provenance, readback, seal, transport, replay, provenance, and local naming rows.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The positive radicand row A, seed row S, reciprocal ledger R, Heron ledger L, finite window W, and contraction estimate D are the complete dyadic surface visible before Newton provenance or Real-facing readback.

Apply the dyadic-Real forward link of Definition 3818.8. It orders the DyadicRatCore radicand and contraction rows before K, places K before the regular rational readback Q, and places Q before the terminal Real seal E.

Use finite-window contraction induction from Definition 3818.3. Each successor window reads the reciprocal row R, the displayed recurrence row L, and the dyadic tolerance comparisons in D before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row is inspected.

The interval-refinement dependency is therefore internal to the dyadic window. The refinement surface of chapter 1831 is consumed while A, S, R, L, W, D are being read, and the remaining rows K, Q, E, T, C, P, N only preserve, replay, source, name, or seal that already-refined route. □

Theorem 2846.10 (Heron square-root Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet first exposes the positive dyadic radicand, seed, reciprocal ledger, Heron ledger, finite window, contraction estimate, Newton provenance, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows*

$$A, S, R, L, W, D, K, Q$$

and only then exposes the $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows*

$$E, T, C, P, N.$$

In particular, the read cannot bypass $\mathsf{RegSeqRat}^{\uparrow}$ *with a host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness.*

Proof. Project the accepted carrier $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$. The first eight rows are the dyadic Heron and regular-rational route, while E, T, C, P, N are the terminal seal and structural rows.

Invoke the regular-Cauchy handoff of Definition 3818.2. It places Q downstream of A, S, R, L, W, D, K and places E downstream of Q.

Use contraction-window stability from Definition 3818.6. Transport preserves the finite window W and contraction estimate D before any regular readback or Real seal can be inspected.

Apply the sibling route of Definition 3818.7. The terminal E row is therefore downstream of $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, so no host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness coordinate can replace the displayed L10 route. □

Falsifiable prediction: Every accepted HeronSquareRootUp Real-facing read factors through positive DyadicRatCoreUp radicand and seed rows, a reciprocal and division ledger, a finite Heron window, dyadic contraction estimates, NewtonIterationUp provenance, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, newtoniteration, regseqrat, real, regularcauchysquare-root: $\mathsf{HeronSquareRoot}^{\uparrow}$ is not bijective to any listed sibling because it binds positive dyadic seed data, reciprocal/division rows, Heron recurrence windows, contraction estimates, Newton provenance, regular readback, Real seal handoff, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{HeronSquareRoot}^{\uparrow}$ packet is generated from positive DyadicRatCore radicand and seed rows, reciprocal and division ledger rows, finite Heron recurrence windows, dyadic contraction estimates, NewtonIteration provenance, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3818.1 and Definition 3818.2 seed the finite Heron square-root carrier and its regular-Cauchy handoff route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host square-root function, quotient-real equality, selected completed limits, arbitrary real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic seed exactness, reciprocal and division ledger exactness, Heron-window stability, dyadic contraction accounting, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HeronSquareRootUp.

The $\mathsf{HeronSquareRoot}^{\uparrow}$ iteration route consumes $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ sealing through the same finite tail-filter discipline used by chapter 1605 and chapter 38. Its positive radicand,

reciprocal ledger, Heron window, contraction estimate, and Newton provenance are therefore visible before the regular rational readback and Real seal can be used, and no host square-root operation, quotient real equality, selected limit, countable choice, or ambient completeness row replaces that tail-filter handoff.

Theorem 2846.11 (Heron square-root DyadicRatCore anchor)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet that exposes a dyadic square-root iteration must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, denominator, reciprocal, halving, and squared-error rows of chapter 291 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 is consumed. In carrier notation the Heron recurrence route remains*

$$A, S, R, L, W, D, K, Q, E, T, C, P, N,$$

where A, S, R, L, W, D *are the dyadic radicand, seed, reciprocal, recurrence, finite-window, and contraction rows;* Q *is the regular rational readback; and* E *is the Real seal. The route exports no host square-root function, quotient real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the Heron carrier from the chapter spine. The radicand and seed rows are dyadic rows, and the reciprocal, halving, and squared-error comparisons in the visible recurrence are also read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next use the regular-Cauchy handoff and finite-window contraction induction already present in this chapter. They place the $\mathsf{RegSeqRat}^{\uparrow}$ readback after the dyadic iteration window, so a Real-facing consumer cannot reach the seal before the dyadic tolerance ledger is displayed.

Finally read the terminal $\mathsf{Real}^{\uparrow}$ route. Since the seal is downstream of the dyadic and regular-rational rows, the Heron packet contributes a dependency edge through chapter 291 and chapter 38, not a host square-root operator or an ambient completeness principle. □

Every accepted $\mathsf{HeronSquareRoot}^{\uparrow}$ consumer that reads a square-root endpoint through a bisection or interval-shrink route first consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, the dyadic interval cells of chapter 306, and the located bisection or nested-interval rows of chapter 1711 and chapter 1713. Only after those interval rows have been displayed may the route pass through the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38; no host square-root endpoint, selected limit, quotient-real equality, trichotomy, or ambient completeness row is exported.

Theorem 2846.12 (Heron square-root Newton dependency route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that seals a square-root approximation consumes the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row of chapter 223, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic rows before the square-root read is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route projects A, S, R, L, W, D *as the finite dyadic Heron ledger, reads* K *as the specialized Newton step, then exposes* Q *and* E *before transport, replay, provenance, and local naming through* T, C, P, N*. It exports no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D contain the positive radicand, seed, reciprocal ledger, recurrence window, and dyadic contraction data needed before any square-root approximation can be consumed.

Apply Definition 3818.2. It orders every Real-facing read through the finite dyadic ledger and the $\mathsf{NewtonIteration}^{\uparrow}$ provenance row before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal seal E are visible.

Use the sibling route in Definition 3818.7. It identifies the same route through chapter 223, chapter 301, and the Real-facing endpoint without adding a host square-root value.

Finally read the regular-Cauchy and Real dependency children already present in this chapter. They keep Q and E attached to the displayed Heron ledger, so the square-root read is sealed only after the Newton, regular rational, Real, and dyadic rows have all been displayed. □

Theorem 2846.13 (Heron square-root Newton interval dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that exports a* $\mathsf{Real}^{\uparrow}$*-facing square-root approximation consumes either the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row or the nested-interval shrink route before the regular-rational and Real rows are public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the route first exposes the positive dyadic radicand and seed rows A, S, the reciprocal ledger R, the Heron recurrence window L, W, and the contraction estimate D. It then reads the Newton provenance row K through chapter 223, or the interval-shrink evidence through the nested route of chapter 1713, before reaching the $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D are the finite radicand, seed, reciprocal, recurrence-window, and contraction rows; none of them is a completed square-root value.

The iteration witness is consumed next. If the packet follows the Newton route, the row K identifies the Heron recurrence as the displayed $\mathsf{NewtonIteration}^{\uparrow}$ step for $x^2 - A$. If it follows the interval route, the same contraction ledger is read as retained nested interval shrink data through the located nested-interval chapter.

Only after one of those finite iteration routes is visible may the Real-facing rows be inspected. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q records the regular rational readback of the displayed approximation window, and the $\mathsf{Real}^{\uparrow}$ row E seals the terminal read. Transport, replay, provenance, and local naming through T, C, P, N preserve that dependency order and add no host square-root operation, selected limit, quotient-real equality, trichotomy, or ambient completeness theorem. □

Theorem 2846.14 (Heron square-root Newton regular-Cauchy bridge)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read factors its Newton update route through chapter 223 and its convergence and seal route through chapter 1756 before reaching the Heron square-root* $\mathsf{NameCert}$ *surface. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the rows A, S, R, L, W, D, K expose the displayed Heron recurrence as a finite $\mathsf{NewtonIteration}^{\uparrow}$ *update, while Q, E, T, C, P, N expose the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. No accepted consumer may read the Heron square-root packet from the Newton row alone or from the regular-Cauchy seal alone.*

Proof. Project the accepted carrier. The rows A, S, R, L, W, D give the positive radicand, seed, reciprocal ledger, recurrence ledger, visible window, and contraction estimate, while K records that this finite recurrence is the displayed $\mathsf{NewtonIteration}^{\uparrow}$ route.

Apply Definition 3818.12. It places every square-root-facing read downstream of the finite Newton update ledger and before the regular rational readback and terminal seal are exposed.

Apply Definition 3818.18. It routes the same packet through the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ convergence service, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the $\mathsf{Real}^{\uparrow}$ seal row E before export.

Finally use the NameCert obligations of Definition 3818.5. The rows T, C, P, N preserve transport, replay, provenance, and local naming for the joint route, so neither the Newton update nor the regular-Cauchy seal can be consumed as an isolated square-root source. □

Theorem 2846.15 (Heron square-root Newton sibling lattice)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that exposes a square-root-facing value first consumes the* $\mathsf{NewtonIteration}^{\uparrow}$ *route of chapter 223, the Newton interval route already attached to the Heron window, the regular Cauchy dependency through chapter 301, and the reciprocal Real route of chapter 6251. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the read projects the positive radicand and seed rows, the reciprocal ledger, the finite Heron window, the contraction estimate, the Newton provenance row K, the regular-rational readback Q, and the terminal $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any square-root read is public.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The finite Heron ledger is visible before any consumer-facing read, so the Newton provenance row K is read while the positive radicand, reciprocal ledger, window, and contraction estimate are still part of the displayed packet.

Apply Definition 3818.12 and Definition 3818.13. These sibling routes attach the Newton iteration surface and the interval-shrink evidence to the same Heron window rather than to a separate square-root object.

Next apply Definition 3818.17 and Definition 3818.19. The regular Cauchy route supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, while the reciprocal route forces the reciprocal ledger through the existing Real reciprocal packet before Q and E are public.

Finally apply Definition 3818.10. The terminal $\mathsf{Real}^{\uparrow}$ row E is available only after the Newton, regular Cauchy, and reciprocal rows have all been displayed, so no host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness coordinate can bypass the sibling lattice. □

Theorem 2846.16 (Heron square-root nonnegative regular-Cauchy reciprocal-window route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed by either chapter 6251 or chapter 1756 must display the same nonnegative* $\mathsf{Real}^{\uparrow}$ *radicand row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and reciprocal-window budget before the square-root-facing seal is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route first exposes the positive radicand and seed rows A, S, then the reciprocal ledger R through the reciprocal-window rows of chapter 703, then the finite Heron window L, W, D, K, and only then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and terminal* $\mathsf{Real}^{\uparrow}$ *seal E used by the regular-Cauchy square-root route.*

Proof. Project the Heron carrier. The rows A, S display the positive radicand and seed, while R is the reciprocal ledger consumed by the Heron recurrence.

Read the reciprocal-window dependency through Definition 3818.19. The Real reciprocal chapter can consume the Heron packet only after the reciprocal ledger has passed through the finite inverse-budget and guarded-division rows named there.

Next read the regular-Cauchy square-root dependency through Definition 3818.18. The same nonnegative radicand route must expose L, W, D, K before the regular-rational readback Q and terminal Real seal E are visible.

The structural rows T, C, P, N preserve the shared route. Thus the reciprocal and regular-Cauchy square-root consumers meet at the displayed nonnegative Real row, RegSeqRat readback, and reciprocal-window rows, with no host square-root function, quotient real equality, selected completed limit, trichotomy, or ambient completeness coordinate. □

Theorem 2846.17 (Heron square-root regular-Cauchy dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read first consumes the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *route of chapter 1756. The visible dependency then proceeds through the positive radicand and reciprocal rows of* $\mathsf{DyadicRatCore}^{\uparrow}$*, the recurrence provenance row of* $\mathsf{NewtonIteration}^{\uparrow}$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any square-root-facing consumer may read the packet. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route is exactly

$$\mathsf{RegularCauchySquareRoot}^{\uparrow} \longmapsto A, S, R, L, W, D, K \longmapsto Q \longmapsto E \longmapsto T, C, P, N.$$

It has no coordinate for a host square-root function, quotient-real equality, selected completed limit, arbitrary real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The positive radicand, seed, reciprocal, division, Heron-window, and contraction rows are the finite data consumed by the regular-Cauchy square-root route.

Apply Definition 3818.2. It orders every Real-facing read through A, S, R, L, W, D, K before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and the $\mathsf{Real}^{\uparrow}$ seal E are visible.

Apply Definition 3818.7. This identifies the same route with the sibling chain through chapter 1756, chapter 291, chapter 223, chapter 301, and chapter 38.

The structural rows T, C, P, N then transport, replay, source, and name the dependency without adding any excluded host square-root, quotient equality, selected limit, trichotomy, or completeness row. □

Theorem 2846.18 (Heron square-root regular-Cauchy square-root forward route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed as a* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *route must display the finite path*

$$\mathsf{NewtonIteration}^{\uparrow}, \mathsf{NestedInterval}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

before the square-root packet is exported. For a carrier

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the consumer first reads the positive radicand and seed rows A, S*, the reciprocal ledger* R*, the Heron recurrence and window rows* L, W*, and the contraction estimate* D*; then it reads the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row* K*, the nested-interval shrink route attached to the same window data, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and finally the structural rows* T, C, P, N*. The route exports no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D are the finite numerical ledger for the radicand, seed, reciprocal accounting, recurrence window, and dyadic contraction estimate.

Read the iteration and interval support next. The row K records the Heron recurrence as the displayed $\mathsf{NewtonIteration}^{\uparrow}$ step, while the nested-interval dependency route of Definition 3818.13 attaches the same contraction ledger to a finite interval-shrink path.

Apply Definition 3818.17. It forces every $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ consumer through the sibling route before the square-root-facing read is public.

Apply Definition 3818.4. The regular rational readback row Q and terminal $\mathsf{Real}^{\uparrow}$ seal E are available only after the displayed Heron ledger and Newton route have been opened.

Finally transport, replay, source, and name the packet through T, C, P, N. These are exactly the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows needed before export, and they add no host square-root operation, quotient equality, selected limit, trichotomy, or ambient completeness coordinate. □

Theorem 2846.19 (Heron square-root real reciprocal forward link)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet reads its reciprocal row through chapter 703, chapter 1943, and chapter 6251 before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal are public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the reciprocal ledger R is the displayed nonzero-apart route feeding the Heron recurrence, and the visible consumer path is

$$R \longmapsto \mathsf{CauchyInverseBudget}^{\uparrow} \longmapsto \mathsf{RegularCauchyDivision}^{\uparrow} \longmapsto \mathsf{RealReciprocal}^{\uparrow} \longmapsto Q \longmapsto E.$$

No Heron read exports a host reciprocal function, total division, quotient real equality, arbitrary trichotomy, selected completed limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted Heron carrier. The reciprocal ledger row R is already a coordinate of the finite packet, and it appears before the Heron recurrence window, contraction estimate, regular readback, and Real seal.

Apply Definition 3818.2. It forces every Real-facing Heron read to pass through the reciprocal and division ledger before Q and E can be inspected.

Use the sibling route from Definition 3818.7. The row R is therefore not a local host inverse; it is a displayed reciprocal ledger over the dyadic and regular-Cauchy route.

Finally read the reciprocal lattice through the named chapters. chapter 703 supplies the finite inverse budget, chapter 1943 supplies the guarded division handoff, and chapter 6251 supplies the nonzero-apart Real reciprocal packet. Only after those rows are visible may the Heron readback Q and terminal seal E be public. □

Theorem 2846.20 (HeronSquareRoot real and RegSeqRat forward lattice)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *square-root approximation read consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint first passes through the finite lattice*

$$\mathsf{NewtonIteration}^{\uparrow},\ \mathsf{RegularCauchySquareRoot}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

already displayed by the sibling routes of chapter 223, chapter 1756, chapter 301, chapter 291, and chapter 38. For an accepted carrier

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the read exposes the Newton provenance row K, the regular-Cauchy square-root route over A, S, R, L, W, D, the regular-rational readback Q, the dyadic arithmetic rows A, S, R, D, and the terminal Real seal E before T, C, P, N may transport, replay, source, or name the result. It exports no host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows A, S, R, L, W, D display the positive dyadic radicand, seed, reciprocal ledger, Heron recurrence, finite window, and contraction estimate, while K records the same finite recurrence as a $\mathsf{NewtonIteration}^{\uparrow}$ route.

Apply Definition 3818.18. The square-root-facing consumer must read the regular-Cauchy route before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ endpoint is public.

Next apply Definition 3818.4 and Definition 3818.10. These place Q before E and keep both rows downstream of the finite dyadic and Newton lattice.

Finally attach T, C, P, N. Transport, replay, provenance, and local naming preserve the displayed route and add no coordinate for a host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient Real completeness. □

Theorem 2846.21 (Heron square-root Real route)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must first display the Newton–Heron recurrence route, the dyadic bisection or Newton-interval dependency route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the reciprocal and nonnegative regular-Cauchy rows, componentwise transport, replay, provenance, and local naming. The route exports no host square-root function, selected limit, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the routed siblings for the carrier. The Newton and Heron recurrence rows expose the iterative rational window, while the dyadic bisection or Newton-interval rows identify the finite interval dependency used by that window.

The completion-facing read then passes through the $\mathsf{RegSeqRat}^{\uparrow}$ row and the reciprocal and nonnegative regular-Cauchy rows before the $\mathsf{Real}^{\uparrow}$ seal is available. Componentwise transport, displayed replay, provenance, and the local $\mathsf{NameCert}$ row preserve the same route without changing its endpoints.

Since none of these routed rows contains a host square-root function, a selected limit, quotient real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$, every Real-facing Heron square-root consumer remains inside the displayed finite dependency path. □

Every $\mathsf{HeronSquareRoot}^{\uparrow}$ read consumes the dyadic-bisection dependency rows of chapter 361 and the Newton interval rows of Definition 3818.13 before the regular Cauchy square-root row is handed to chapter 301 and the terminal seal is read through chapter 38. This route keeps square-root dependency data inside the displayed L10 lattice and does not export selected limits or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 2846.22 (Heron square-root RegSeqRat Real dependency route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *iteration budget consumed by the reciprocal forward link first reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular windows and the* $\mathsf{Real}^{\uparrow}$ *seal rows already displayed by the Heron carrier. For*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the reciprocal route of Definition 3818.19 exposes the square-root approximation only after the regular readback Q and terminal seal E are attached to the same positive radicand, seed, reciprocal ledger, Heron recurrence window, dyadic contraction estimate, and Newton provenance rows. Thus the visible dependency passes through chapter 6251, chapter 301, and chapter 38 without adding a host square-root function, quotient real equality, selected completed limit, arbitrary trichotomy, or ambient completeness coordinate.

Proof. Project the Heron carrier. The rows A, S, R, L, W, D, K are the finite radicand, seed, reciprocal, recurrence, window, contraction, and Newton-provenance rows consumed before any square-root-facing read is public.

Apply Definition 3818.19. Its reciprocal forward path reads the existing reciprocal chapter through the same row R and then returns to the Heron readback boundary rather than fabricating a separate inverse or square-root object.

Now read the regular-rational and real rows. The row Q is the $\mathsf{RegSeqRat}^{\uparrow}$ regular-window readback produced from the displayed Heron iteration budget, and E is the terminal $\mathsf{Real}^{\uparrow}$ seal visible only after that readback.

Finally transport and replay through T, C, P, N. These rows keep the reciprocal, RegSeqRat, and Real dependencies on the same finite Heron route, so the square-root approximation is exposed as a displayed NameCert dependency rather than a host operation, quotient equality, selected limit, trichotomy, or completeness principle. □

Theorem 2846.23 (Heron square-root RegSeqRat forward dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint must first pass through chapter 1756. For*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the square-root-facing dependency displays the regular-Cauchy source route before exposing the Heron rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$*. Thus the RegSeqRat or Real consumer reads the positive radicand, seed, reciprocal ledger, recurrence window, contraction estimate, Newton provenance, regular readback, Real seal, transport, replay, provenance, and local naming only after the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *handoff is visible.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D, K display the radicand, seed, reciprocal ledger, recurrence window, contraction estimate, and Newton provenance before the regular readback row Q or terminal Real seal row E can be consumed.

Invoke the regular-Cauchy handoff through Definition 3818.18 and Definition 3818.17. These routes force the square-root-facing read through chapter 1756 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ endpoint receives the Heron packet.

Now use the RegSeqRat Real dependency route of Definition 3818.22. It keeps Q and E downstream of the same A, S, R, L, W, D, K rows and therefore prevents a detached square-root approximation from entering the regular-rational or Real boundary.

Finally close through T, C, P, N. Transport, replay, provenance, and local naming preserve the same dependency order, so the read exports a finite Heron route rather than a host square-root function, quotient Real equality, selected completed limit, arbitrary trichotomy, or ambient completeness coordinate. □

Theorem 2846.24 (Heron square-root sibling consumer lattice)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer route used by chapter 6251, chapter 6793, and chapter 7196 first reads the Heron square-root packet's dyadic approximation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *rows.*

Proof. Project the accepted Heron carrier from the chapter spine. The rows A, S, R, L, W, D expose the dyadic radicand, seed, reciprocal ledger, Heron recurrence window, and contraction estimate before any downstream consumer can inspect the regular-rational readback Q or terminal $\mathsf{Real}^{\uparrow}$ seal E.

Apply the Heron reciprocal, continuity, and Newton sibling routes already recorded in this chapter. The real reciprocal forward link routes the reciprocal row through chapter 6251; the nonnegative regular-Cauchy reciprocal-window route keeps the same Heron window visible for square-root continuity consumers; and the Newton sibling lattice keeps the dyadic Newton route attached to the displayed approximation rows.

Now read the downstream chapters. chapter 6251, chapter 6793, and chapter 7196 therefore receive the Heron packet only after Q, E, T, C, P, N have preserved readback, Real sealing, transport, replay, provenance, and local naming. No consumer route obtains a host square-root value, a hidden reciprocal, quotient-real equality, selected completion, or ambient completeness row. □

Theorem 2846.25 (Heron square-root sibling dependency lattice)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that is exposed to a Real or Completion consumer must pass through the displayed Newton, regular-Cauchy, RegSeqRat, and Real siblings before the terminal square-root value is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the lattice order first reads the positive radicand and seed rows, then the Heron recurrence and Newton provenance rows of chapter 223, then the regular-Cauchy square-root route of chapter 1756, then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, and finally the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route exports no host square-root operation, quotient Real equality, selected completed limit, arbitrary trichotomy, countable choice, or ambient completeness theorem.*

Proof. Project the accepted Heron carrier. The rows A, S, R, L, W, D, K expose the positive source data, reciprocal ledger, visible Heron window, dyadic contraction estimate, and Newton provenance before any Real-facing row is available.

Apply the existing Heron handoff and sibling-route theorems. They place the $\mathsf{NewtonIteration}^{\uparrow}$ row on the same finite recurrence window, route the convergence read through $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, and force the $\mathsf{RegSeqRat}^{\uparrow}$ row Q to be inspected before the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The square-root boundary is therefore a consumer of the same displayed seal, not an independent oracle. Transport, replay, provenance, and local naming through T, C, P, N preserve the order of the sibling rows, so the public Heron read remains inside the horizontal dependency lattice and cannot export any excluded host or quotient object. □

Theorem 2846.26 (Heron square-root sibling dependency route)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer that exports a square-root approximation must first read the initial positive* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *rows, the* $\mathsf{NewtonIteration}^{\uparrow}$ *update route, the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *convergence readback, and the* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *interval discipline. These sibling rows are a dependency route for the Heron packet, not a new square-root existence theorem, not a host square-root function, and not a selected completed limit.*

Proof. Project the Heron carrier through the already displayed radicand, seed, reciprocal ledger, recurrence window, dyadic contraction, readback, terminal seal, transport, replay, provenance, and local naming rows.

Read the initial numerical rows first. The positive $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ data supply the radicand and rational seed interface before any iterative square-root approximation can be consumed.

Next follow the sibling routes. The $\mathsf{NewtonIteration}^{\uparrow}$ row records the Heron update, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ records the constructive convergence readback, and $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ records the interval discipline that keeps the approximation inside finite dyadic windows.

The recurrence also reads the reciprocal-accounting route before the regular readback and Real seal become public. Its reciprocal ledger passes through chapter 703, chapter 1943, and chapter 6251, matching the forward link of Definition 3818.19.

Thus the public Heron read is a horizontal dependency route through the existing Newton, regular-Cauchy, Real, rational, and interval siblings. It exports only the displayed dependency order and adds no square-root theorem, selected limit, quotient-real equality, trichotomy, or ambient completeness coordinate. □

Theorem 2846.27 (Heron square-root sibling route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet must read* $\mathsf{DyadicRatCore}^{\uparrow}$ *radicand and tolerance data from chapter 291,* $\mathsf{NewtonIteration}^{\uparrow}$ *recurrence rows from chapter 223, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 together with the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows used by chapter 1756 before it exports the square-root route. The Heron carrier therefore sits on the displayed sibling lattice rather than on a host square-root function, selected completed limit, quotient-real equality, trichotomy row, or ambient completeness coordinate.*

Chapter 2847

A Concrete Naming Certificate for $\mathsf{ContractiveMap}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{ContractiveMap}^{\uparrow}$ packet records a displayed contraction between metric carriers as finite NameCert data. It sits over the source and target metric rows of chapter 57, the complete-metric consumer boundary of chapter 128, the tolerance export surface of chapter 369, the endpoint discipline of chapter 396, and the finite orbit route of chapter 2359. The packet names the map-level shrink surface that Banach and Picard consumers can inspect before any fixed point, completed limit, quotient equality, or host function extensionality is available.

Definition 2847.1 (Contractive map carrier)**.** *A* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, f, k, D, G, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *f is the finite map-code row from carried source histories to carried target histories.*
3. *k is the contraction-constant row, with displayed nonnegativity and strict unit-bound evidence.*
4. *D is the metric-distance witness row recording source-distance and target-distance requests for the accepted map reads.*
5. *G is the graph row witnessing that every accepted application of f is a finite BEDC graph read.*
6. *S is the shrink ledger recording each displayed comparison*

   $$d_Y(f(x), f(y)) \leq k\, d_X(x, y)$$

   for source histories carried by X and graph reads carried by G.
7. *H records componentwise* hsame *transport for the source and target metric rows, map-code row, contraction constant, distance witnesses, graph row, shrink ledger, continuation rows, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from source metric requests through f, k, D, G, S to target metric responses, uniform-modulus handoff, Picard iteration reads, and fixed-point consumers.*

9. *P is the* Pkg *provenance over* ContractiveMap$^{\uparrow}$, Met$^{\uparrow}$, CompleteMetric$^{\uparrow}$, UniformModulus$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{ContractiveMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected fixed point, a completed Picard limit, quotient equality of maps, host function extensionality, ambient completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContractiveMapUp.ContractiveMapTasteGate single carrier alignment`

Theorem 2847.2 (Contractive map NameCert obligations)**.** *For every accepted* ContractiveMap$^{\uparrow}$ *carrier $K = (X, Y, f, k, D, G, S, H, C, P, N)$, the local* NameCert$_{\text{ContractiveMap}^{\uparrow}}$ *surface consists exactly of source and target metric exposure, map-code admission, contraction-constant exactness, distance-witness exposure, graph replay, shrink-ledger replay, componentwise transport, displayed continuation, package provenance, and local naming through the rows $X, Y, f, k, D, G, S, H, C, P, N$.*

Proof. Project the carrier. The analytic rows are precisely the source metric X, target metric Y, map code f, contraction constant k, distance witnesses D, graph row G, and shrink ledger S.

The contraction obligation is replayed from the same rows. A target-distance read must pass through G, use the distance witnesses in D, and cite the comparison stored in S against the same constant row k.

The structural rows preserve that packet and add no analytic coordinate. Transport through H, replay through C, provenance through P, and local naming through N keep the displayed route intact, so the listed rows exhaust the NameCert surface. □

Theorem 2847.3 (Contractive map Picard consumer handoff)**.** *Every* PicardIteration$^{\uparrow}$ *or* BanachFixedPoint$^{\uparrow}$ *consumer of an accepted* ContractiveMap$^{\uparrow}$ *carrier receives contraction evidence only through*

$$X, Y, f, k, D, G, S, H, C, P, N.$$

The handoff exports metric-distance shrink replay and the strict unit-bound row for k; it does not export fixed-point existence, complete-metric convergence, quotient equality of maps, a selected endpoint, or host function extensionality.

Proof. Start with the NameCert obligations of Definition 3819.2. They make the shrink ledger S the only row that can justify a Picard or Banach contraction read.

A Picard consumer therefore reads the graph row G, the distance witnesses D, and the contraction constant k before it can replay a one-step or finite-orbit shrink comparison. The continuation row C transports this order to uniform-modulus, iteration, and fixed-point consumers without adding a limit point.

Completeness and endpoint selection remain outside the packet. A Banach consumer must attach its own complete-metric and endpoint rows, while the ContractiveMap$^{\uparrow}$ handoff supplies only the finite map-level shrink data. □

Theorem 2847.4 (Contractive map constant-row exactness)**.** *For every accepted* ContractiveMap$^{\uparrow}$ *carrier $K = (X, Y, f, k, D, G, S, H, C, P, N)$, every contraction read exposed by* NameCert$_{\text{ContractiveMap}^{\uparrow}}$ *uses the displayed constant row k together with its nonnegativity and strict unit-bound evidence. No Picard, Banach, or uniform-modulus consumer may replace k by an unstored scalar or by a scalar recovered from ambient host data.*

Proof. Project the carrier and apply the NameCert obligations of Definition 3819.2. The obligation list has a single contraction-constant coordinate, namely the row k, and it records the strict unit-bound evidence as part of that row.

The Picard handoff of Definition 3819.3 exports the same row to downstream consumers. Since the handoff excludes host function extensionality, quotient equality of maps, completed limits, and selected endpoints, there is no second route from which a different contraction scalar can be read. □

Theorem 2847.5 (Contractive map distance-witness stability)**.** *For every accepted* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier* $K = (X, Y, f, k, D, G, S, H, C, P, N)$*, each source-distance or target-distance request used in a contraction comparison factors through the distance-witness row* D*. If the request involves a displayed application of* f*, the same read is tied to the graph row* G *before it can be used by the shrink ledger* S*.*

Proof. Project the carrier. The obligations in Definition 3819.2 list distance-witness exposure and graph replay as separate rows, and the proof of that theorem identifies D as the row carrying source-distance and target-distance requests.

A contraction comparison in S is therefore not a free metric assertion. It first reads the relevant source and target distance witnesses through D, and when the comparison evaluates f it reads the corresponding finite graph entry through G. The continuation row C can replay this order, but it cannot bypass either displayed row. □

Theorem 2847.6 (Contractive map shrink-ledger replay)**.** *For every accepted* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier* $K = (X, Y, f, k, D, G, S, H, C, P, N)$*, every shrink comparison available to* $\mathsf{PicardIteration}^{\uparrow}$*,* $\mathsf{BanachFixedPoint}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *consumers is one of the comparisons stored in* S *and transported, replayed, provenanced, and locally named through* H, C, P, N*. The carrier exposes no additional contraction inequality outside those rows.*

Proof. Project the carrier and read the shrink-ledger clause of Definition 3819.2. That clause makes S the row recording the displayed comparisons

$$d_Y(f(x), f(y)) \leq k\, d_X(x, y)$$

for carried source histories and graph reads.

The consumer handoff in Definition 3819.3 sends the same finite shrink data to Picard and Banach consumers, and the carrier definition routes uniform-modulus handoff through the same continuation surface. Componentwise transport through H, displayed replay through C, provenance through P, and local naming through N preserve the stored ledger entries rather than generating fresh analytic inequalities. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3819.1, Definition 3819.2, Definition 3819.3, Definition 3819.4, Definition 3819.5, and Definition 3819.6 seed the finite contraction-map surface for metric, graph, constant, shrink-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert fixed-point existence, complete-metric convergence, quotient equality of maps, selected endpoints, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{ContractiveMap}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, contraction-constant exactness, graph replay, distance-witness exposure, shrink-ledger replay, Picard handoff, and field faithfulness for $X, Y, f, k, D, G, S, H, C, P, N$.

Constructive story: A ContractiveMap$^{\uparrow}$ packet is generated from source and target metric rows, a finite map-code row, a contraction constant below one, metric-distance witnesses, a graph row, a shrink ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2848

A Concrete Naming Certificate for $\mathsf{ClosedBoundedSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet records a bounded real sequence as finite BEDC source rows rather than as a completed sequence space or a compactness theorem. It is the bounded-sequence neighbour of the closed interval carrier in chapter 1591 and the Bolzano–Weierstrass route of chapter 1254. The carrier exposes only the finite bound interval row, the sequence windows read through chapter 126, the regular rational readbacks of chapter 301, the dyadic bound ledger of chapter 291, the terminal real seal of chapter 38, transport, replay, provenance, and local naming.

Definition 2848.1 (Closed bounded sequence carrier)**.** *A* $\mathsf{ClosedBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the bound interval row, read through the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *certificate of chapter 1591;*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows inspecting the requested sequence coordinates;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same coordinate windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording dyadic lower and upper bound comparisons against the interval row I;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for each inspected coordinate, reached only after I, W, R, D have been displayed;*
6. *H records componentwise* hsame *transport of the interval, window, readback, dyadic, and seal rows;*
7. *C records displayed* Cont *replay from interval bounds through finite sequence windows and coordinate seals;*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedSequence}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$ *row.*

The carrier exports no general sequential compactness principle, no countable choice selector, no selected convergent subsequence, and no ambient space of all bounded sequences. Its public object is the finite bounded-coordinate packet B.

Lean4 ✓ `BEDC.Derived.ClosedBoundedSequenceUp.TasteGate.ClosedBoundedSequenceTasteGate single carrier alignment`

Theorem 2848.2 (Closed bounded sequence NameCert obligations)**.** *For the carrier of Definition 3820.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for* $B = (I, W, R, D, E, H, C, P, N)$*; bound-interval admission through* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*; coordinate-window admission through* $\mathsf{StreamName}^{\uparrow}$*; regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*; dyadic containment of every inspected coordinate through* $\mathsf{DyadicRatCore}^{\uparrow}$*; terminal coordinate sealing through* $\mathsf{Real}^{\uparrow}$*; classifier preservation under* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier. Its interval row, coordinate-window rows, readback rows, dyadic bound ledger, Real seals, transport, replay, provenance, and local name are visible coordinates of one finite packet.

Read the bound before the coordinate seals. The $\mathsf{ClosedBoundedInterval}^{\uparrow}$ row supplies the lower endpoint, upper endpoint, endpoint order, and containment ledgers, while W, R, D inspect only finitely many sequence coordinates against that same interval.

The structural rows close the naming surface. Componentwise hsame transport and displayed Cont replay preserve exactly the listed rows, and the provenance and local naming rows source that finite route without adding sequential compactness, a choice-selected subsequence, or an ambient sequence-space object. □

Theorem 2848.3 (Closed bounded sequence Cauchy-subsequence window)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or Cauchy-subsequence consumer of an accepted* $\mathsf{ClosedBoundedSequence}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows $I, W, R, D, E, H, C, P, N$*. The route supplies bounded coordinate windows and dyadic containment readbacks, but it does not supply general sequential compactness, countable choice, a selected convergent subsequence, or an ambient completeness theorem.*

Proof. Start from the carrier inventory. The only sequence-facing rows are the finite windows W, regular rational readbacks R, dyadic bound ledger D, and coordinate seals E, all attached to the bound interval row I.

Next route the downstream consumer through the same rows. A $\mathsf{BolzanoWeierstrass}^{\uparrow}$ or Cauchy-subsequence consumer can request a bounded-coordinate window, but the request is replayed by C only after I, W, R, D, E have been transported by H and sourced by P, N.

The excluded principles are not carrier coordinates. Since the packet contains no selector for an infinite subsequence, no global compactness row, no countable choice row, and no ambient complete sequence space, the consumer receives only the displayed finite bounded-window route. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ or Cauchy-subsequence read of a $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet factors through a bound interval row, finite StreamName coordinate windows, RegSeqRat readbacks, a DyadicRatCore bound ledger, RealUp coordinate seals, transport, replay, provenance, and local naming.

Independence witness: closedboundedinterval, bolzanoweierstrass, streamname, regseqrat, dyadicratcore, real: $\mathsf{ClosedBoundedSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly

carries a bounded sequence source, finite coordinate windows, dyadic bound readbacks, coordinate seals, and a consumer-facing bounded-window route.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet is generated from a closed bounded interval row, finite sequence windows, regular rational readbacks, dyadic bound comparisons, RealUp coordinate seals, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3820.1, Definition 3820.2, and Definition 3820.3 bind the finite bounded-coordinate carrier before Bolzano–Weierstrass and Cauchy-subsequence consumers add their own extraction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, general sequential compactness, countable choice, selected convergent subsequences, or ambient sequence-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, bound-readback, and Cauchy-subsequence handoff for BEDC.Derived.ClosedBoundedSequenceUp.

Chapter 2849

A Concrete Naming Certificate for $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet records the finite radius-choice ledger used in the Heine–Cantor route. It reads compact finite-net data from chapter 129, pointwise continuity rows from chapter 130, finite lower-bound folds from chapter 619, dyadic tolerance bookkeeping from chapter 291, and the uniform-modulus consumer surface of chapter 369. The chapter is a finite NameCert carrier for the radius handoff; it is not an open-cover compactness proof, a semantic continuity theorem, or a choice function over all points.

Definition 2849.1 (Compact continuity radius-choice carrier)**.** *A* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, M, Q, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric finite-net row, read through* $\mathsf{CompactMetric}^{\uparrow}$*;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row at the displayed finite centers;*
3. *M is the finite center-to-radius matching ledger;*
4. *Q is the positive rational radius ledger attached to the selected center rows;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger;*
6. *L is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound fold over the displayed radius rows;*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row reached from the finite lower-bound ledger;*
8. *H records componentwise* hsame *transport for compact-net, continuous-map, radius, dyadic, fold, and handoff rows;*
9. *C records displayed* Cont *replay from finite-net centers and pointwise radii to the uniform-modulus handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$ *row.*

The carrier exposes the finite radius ledger and its consumer handoff. It has no coordinate for an external radius-selection function, open-cover compactness, semantic function equality, or a point chosen outside the finite compact-net bundle.

Lean4 (definition) `BEDC.Derived.CompactContinuityRadiusChoiceUp.CompactContinuityRadiusChoiceUp`

Theorem 2849.2 (Compact continuity radius-choice NameCert obligations)**.** *For the carrier of Definition 3821.1, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (K, F, M, Q, D, L, U, H, C, P, N)$*; compact finite-net admission through* $\mathsf{CompactMetric}^{\uparrow}$*; pointwise-radius admission through* $\mathsf{ContinuousMap}^{\uparrow}$*; radius matching through* M, Q, D*; finite lower-bound folding through* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*; the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through* U*; classifier preservation under* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier. The compact finite-net row, continuous-map pointwise rows, radius ledgers, dyadic comparisons, lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local name are all visible coordinates.

The radius choice is finite. Each radius row is attached to a displayed compact-net center and a displayed pointwise-modulus row; the lower-bound fold consumes only those finitely many positive rational and dyadic comparisons.

The consumer handoff is therefore confined to the displayed route. Componentwise transport and replay preserve the same rows, while provenance and local naming record the route without adding a global selector, open-cover compactness object, semantic continuity theorem, or point outside the finite bundle. □

Theorem 2849.3 (Compact continuity radius-choice uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{FiniteNetMinimumFold}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

Equivalently, the handoff receives the compact finite-net centers, pointwise radius rows, dyadic tolerance comparisons, the finite lower-bound ledger, and the displayed transport, replay, provenance, and local naming rows. It receives no open-cover compactness object, no external radius-selection function, no semantic continuity theorem, and no point selected outside the finite net.

Proof. Start with the compact-net and pointwise-modulus rows. The carrier exposes K and F before any radius row can be folded, so every radius is tied to a displayed finite center and a displayed continuous-map modulus row.

Next read the finite lower-bound ledger. The rows M, Q, D, L package the radius matching, positive rational radius data, dyadic comparisons, and finite minimum fold. This is the only source of a tolerance-only handoff to $\mathsf{UniformModulus}^{\uparrow}$.

Finally apply the structural rows. The route H, C, P, N transports, replays, sources, and names exactly the finite radius-choice ledger, so a uniform-modulus consumer cannot add open-cover compactness, a global radius-selection function, semantic continuity, or an off-net selected point. □

Falsifiable prediction: Every $\mathsf{UniformModulus}^{\uparrow}$ read of a $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet factors through CompactMetric finite-net centers, ContinuousMap pointwise radii, DyadicRatCore tolerance comparisons, a FiniteNetMinimumFold lower-bound ledger, transport, replay, provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, finitenetminimumfold, dyadicratcore, heinecantor: $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite compact-net radius ledger and the UniformModulus-facing lower-bound handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet is generated from compact finite-net centers, ContinuousMap pointwise radius rows, radius matching ledgers, dyadic tolerance comparisons, a finite lower-bound fold, a UniformModulus handoff row, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3821.1, Definition 3821.2, and Definition 3821.3 bind the finite radius-choice carrier feeding the Heine–Cantor uniform-modulus route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, semantic continuity, global radius selection, or off-net point choice.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, finite radius fold, and UniformModulus handoff for BEDC.Derived.CompactContinuityRadiusChoiceUp.

Chapter 2850

A Concrete Naming Certificate for $\mathsf{DuhamelPrinciple}^{\uparrow}$

Following Pazy's semigroup treatment of evolution equations and Evans' variation-of-constants discussion for forced linear problems, the $\mathsf{DuhamelPrinciple}^{\uparrow}$ packet records the finite BEDC route by which a homogeneous evolution row, a forcing row, and an integral convolution row are read as one solution handoff. It sits after the real-time surface of chapter 38, the integration interface of chapter 123, the ODE local-flow source of chapter 193, the dynamical-system flow packet of chapter 195, and the semigroup associativity surface of chapter 79. The carrier names the forcing route; it does not assert a host solution operator, classical existence theorem, quotient trajectory equality, or choice-selected mild solution.

Definition 2850.1 (DuhamelPrinciple carrier)**.** *A* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, E, F, G, I, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Real}^{\uparrow}$ *time-window row, including the initial time, endpoint time, and inspected finite subintervals.*
2. *E is the homogeneous evolution row, read as an* $\mathsf{Ode}^{\uparrow}$ *local-flow or* $\mathsf{DynSystem}^{\uparrow}$ *flow packet and constrained by the* $\mathsf{Semigroup}^{\uparrow}$ *composition ledger.*
3. *F is the forcing-data row, recording the displayed inhomogeneous source samples and their* $\mathsf{Real}^{\uparrow}$ *readbacks.*
4. *G is the propagator action row, pairing each displayed forcing sample with the elapsed-time evolution row that carries it toward the endpoint.*
5. *I is the* $\mathsf{Integral}^{\uparrow}$ *convolution row over the finite time window, with its partition, approximation, and endpoint ledgers visible as integral data.*
6. *S is the solution readback row, exposing the homogeneous contribution and the integral forcing contribution as the endpoint history inspected by downstream consumers.*
7. *H records componentwise* hsame *transport for time windows, evolution rows, forcing rows, propagator rows, integral rows, solution readbacks, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Duhamel request through $T, E, F, G, I, S, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{DuhamelPrinciple}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, F, G, I, S, H, C, P, N$. It has no coordinate for a host semigroup on a completed function space, an unobserved Bochner integral, a selected solution operator, classical choice, quotient trajectory equality, or global well-posedness.

Lean4 ✓ `BEDC.Derived.DuhamelPrincipleUp.DuhamelPrincipleTasteGate single carrier alignment`

Theorem 2850.2 (DuhamelPrinciple NameCert obligations)**.** *For every accepted* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *carrier $D = (T, E, F, G, I, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}$ *obligation surface consists exactly of real-time window admission, homogeneous evolution readback, forcing-data admission, propagator action exposure,* $\mathsf{Integral}^{\uparrow}$ *convolution readback, endpoint solution readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier. The time row T fixes the inspected interval before the evolution, forcing, or convolution rows are read. The homogeneous row E is already a named ODE or dynamical-system flow route, and its semigroup composition behavior is visible only through the displayed $\mathsf{Semigroup}^{\uparrow}$ ledger.

Read the forcing and propagator rows next. The row F supplies the inhomogeneous samples and real readbacks, while G records how the displayed evolution row carries each sample toward the endpoint. No host trajectory space is introduced at this stage.

The integral row I is then the only source of convolution data. Its partition, approximation, and endpoint ledgers are inherited from $\mathsf{Integral}^{\uparrow}$, and the solution row S may inspect only the homogeneous contribution together with that displayed convolution endpoint.

Finally use H and C to transport and replay the same finite rows, and use P and N for provenance and local naming. Since all excluded analytic objects lack carrier coordinates, they cannot enter the local NameCert obligation surface. □

Theorem 2850.3 (DuhamelPrinciple forcing route)**.** *Every downstream* $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, *stability, or PDE consumer of an accepted* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *packet reads the inhomogeneous contribution through the finite route*

$$T, E, F, G \longmapsto I \longmapsto S \longmapsto H, C, P, N.$$

Thus the forced endpoint is available only as homogeneous evolution plus the displayed $\mathsf{Integral}^{\uparrow}$ *convolution of forcing data against the propagator row. The route exports no host solution operator, unbounded semigroup theorem, chosen mild solution, quotient trajectory equality, or ambient completeness principle.*

Proof. Begin with a forcing-facing consumer. The real-time row T and the homogeneous evolution row E determine the displayed elapsed-time cells and the semigroup-style composition ledger available to the packet.

Read F and G after that boundary is fixed. The forcing samples are paired with visible propagator action rows, so the consumer sees only finitely displayed source observations and their transported endpoint contributions.

The convolution is then read from I. This invokes the $\mathsf{Integral}^{\uparrow}$ carrier and ledger discipline over the same time window, so partitions and approximations remain integral rows rather than hidden choices.

The endpoint row S repacks the homogeneous part and the forcing convolution as the public readback. Transport, replay, provenance, and naming are supplied by H, C, P, N, which preserve the same route and leave every absent analytic object outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{DuhamelPrinciple}^{\uparrow}$ packet is generated from a RealUp time window, homogeneous ODE or DynSystem evolution row, Semigroup composition ledger, forcing-data row, propagator action row, IntegralUp convolution row, solution readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3822.1, Definition 3822.2, and Definition 3822.3 seed the finite variation-of-constants carrier, obligation surface, and forcing route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host solution operators, unbounded semigroup theorems, chosen mild solutions, quotient trajectory equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, forcing-route exactness, IntegralUp convolution readback, evolution transport stability, solution nonescape, and TasteGate field faithfulness for BEDC.Derived.DuhamelPrincipleUp.

Chapter 2851

A Concrete Naming Certificate for $\mathsf{LocatedSequenceLimit}^{\uparrow}$

Following Bishop and Bridges on located real limits of Cauchy sequences, the $\mathsf{LocatedSequenceLimit}^{\uparrow}$ packet records the finite evidence that a displayed stream name, regular rational sequence, and real-name seal describe the same located limit route. It sits between the stream observation surface of chapter 126, the regular rational sequence surface of chapter 301, the terminal real-name seal of chapter 38, the Cauchy criterion route of chapter 1509, and the located Cauchy surface of chapter 1595. The packet is finite L10 NameCert data for a located sequence limit; it is not a selected quotient representative, an ambient completeness theorem, a trichotomy decision, or a countable-choice schedule.

Definition 2851.1 (Located sequence limit carrier)**.** *A* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing only finite windows of the displayed sequence name.*
2. *W is the located-window ledger. For each displayed precision request it records the finite inspected window, the located side of the comparison, and the tolerance budget used by the read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the inspected windows as regular rational sequence data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the stream window and regular rational readback agree on the same located limit route.*
5. *H records componentwise* hsame *transport for the stream source, located window, regular readback, real seal, replay, provenance, and local name.*
6. *C records displayed* Cont *replay from a precision request through*

$$S, W \longmapsto R \longmapsto E.$$

7. *P is the* Pkg *provenance over* $\mathsf{LocatedSequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCrite}$ $\mathsf{LocatedCauchySequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the finite stream windows, located comparison ledger, regular rational readback, and real-name seal describe one displayed limit route. No carrier coordinate stores a quotient sequence, selected representative of the limit, host equality on reals, trichotomy oracle, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedSequenceLimitUp.TasteGate.LocatedSequenceLimitTasteGate single carrier alignment`

Theorem 2851.2 (Located sequence limit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier* $L = (S, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}^{\uparrow}}$ *surface consists exactly of stream-window admission, located-window tolerance accounting, regular rational readback, real-name seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows* S, W, R, E, H, C, P, N*. The* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *and* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *siblings may consume the packet only through those displayed rows.*

Proof. Project the carrier of Definition 3823.1. The sequence is visible only through the finite stream row S, and the located comparison evidence is visible only through the window ledger W.

The readback row R repacks the same finite observations as $\mathsf{RegSeqRat}^{\uparrow}$ data. The real-name row E is downstream of S, W, R, so a real-facing consumer reaches the limit seal only after replaying the displayed located route.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier has no coordinate for quotient sequences, selected representatives, host equality, trichotomy, or ambient completeness, the listed NameCert rows are exhaustive. □

Theorem 2851.3 (Located sequence limit Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, *or* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier must read*

$$S, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the terminal real-name seal cannot be replaced by host real equality, a quotient completion object, a selected subsequence, trichotomy on signs, or an ambient completeness proof.

Proof. Begin with a consumer request for real-limit or Cauchy-facing data. The packet exposes sequence observations only through S, and every located precision request is accounted for by W.

The regular rational readback row R is the only bridge from the inspected windows to a rational sequence surface. Thus the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ seal E only after the finite route has been replayed.

The rows H, C, P, N transport, replay, source, and name the same route. No excluded analytic object is present in the tuple, so no downstream consumer can extract one from the packet. □

Theorem 2851.4 (Located sequence limit window uniqueness)**.** *Let*

$$L = (S, W, R, E, H, C, P, N) \quad \textit{and} \quad L' = (S', W', R', E', H', C', P', N')$$

be accepted $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *packets. If they expose the same source* $\mathsf{StreamName}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, located-window ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, then they determine the same finite located-limit route. In particular, there are not two different accepted finite L10 windows for the same located request.*

Proof. Project both packets through the carrier definition Definition 3823.1. The hypotheses identify the only visible source row S, the only located-window ledger W, and the only regular readback row R available to either packet.

Apply the NameCert obligation theorem Definition 3823.2. It states that all accepted located-limit data are exhausted by the displayed rows S, W, R, E, H, C, P, N, so the two packets have no additional coordinate on which their finite routes could diverge.

Finally use the Real seal nonescape theorem Definition 3823.3. The terminal seal is reached only through the same stream, located-window, and readback route, and the carrier contains no selected quotient representative, ambient completeness proof, trichotomy oracle, or countable-choice schedule. Thus the accepted packets determine one finite located-limit route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}$.

Constructive story: A $\mathsf{LocatedSequenceLimit}^{\uparrow}$ packet is generated from StreamName observations, located finite-window tolerance data, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3823.1, Definition 3823.2, Definition 3823.3, and Definition 3823.4 seed the finite located sequence-limit surface for the L10 StreamName-to-RegSeqRat-to-Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window accounting, RegSeqRat readback, Real seal routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSequenceLimitUp.

Chapter 2852

A Concrete Naming Certificate for $\mathsf{DedekindGapLocatedCut}^{\uparrow}$

Following Bishop and Bridges constructive located cuts, the $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ packet records the finite lower-upper separation data that a completion or real-comparison consumer may inspect before any construction of $\mathsf{Real}^{\uparrow}$ is claimed. It is a located-cut boundary over the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational sequence route of chapter 301, the terminal real-name seal of chapter 38, and the located-supremum consumer surface of chapter 1427. The packet supplies a gap-located comparison boundary consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers; it is not a Dedekind construction of $\mathsf{Real}^{\uparrow}$, not quotient equality of cuts, and not an ambient completeness principle.

Definition 2852.1 (Dedekind gap located cut carrier)**.** *A* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, D, W, R, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-cut ledger: finite rational cells certified to lie below the boundary.*
2. *U is the upper-cut ledger: finite rational cells certified to lie above the boundary.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row. For every displayed tolerance request it names a dyadic bracket or a separating dyadic cell between the currently exposed lower and upper ledgers.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the finite lower, upper, and gap requests actually read by a consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the windowed lower-upper data as regular rational observations.*
6. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *boundary handoff row available to consumers that need an upper-bound or lower-approximation surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is exposed only after L, U, D, W, R, S have been displayed.*
8. *H records componentwise* hsame *transport for lower cells, upper cells, dyadic gaps, stream windows, regular readback, located-supremum handoff, real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through the ordered route $L, U, D, W, R, S, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, W, R, S, E, H, C, P, N$. It has no coordinate for quotient equality of cuts, a selected real representative, countable choice, trichotomy, a host order-completeness theorem, or a Dedekind construction of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DedekindGapLocatedCutUp.DedekindGapLocatedCutTasteGate single carrier alignment`

Theorem 2852.2 (Dedekind gap located cut carrier boundary)**.** *Every accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier is read through the displayed route*

$$L, U, D, W, R, S, E, H, C, P, N.$$

Thus a consumer receives lower-cut cells, upper-cut cells, dyadic gap witnesses, finite stream windows, regular rational readback, a located-supremum boundary handoff, a terminal Real seal, transport, replay, provenance, and local naming. No accepted read exposes quotient equality of lower-upper cuts, a selected real representative, trichotomy, countable choice, ambient completeness, or a construction of $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts.*

Proof. Project the carrier of Definition 3824.1. The rows L and U are the only lower and upper cut data, while D supplies the finite dyadic gap evidence inspected by the request.

The rows W and R place those gap observations inside a finite observation schedule and regular rational readback. The located-supremum row S is downstream of the same windowed data, and the real seal E is reached only after that route is displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not fields of the carrier, no consumer can recover them from an accepted $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ read. □

Theorem 2852.3 (Dedekind gap located cut obligations)**.** *For an accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier $G = (L, U, D, W, R, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper ledgers, dyadic gap exposure, StreamName window exposure, RegSeqRat readback, LocatedSupremum boundary handoff, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming. The obligations do not include a quotient of cuts, a host real equality test, selected supremum data outside S, or any completeness assertion for* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission lists the rows $L, U, D, W, R, S, E, H, C, P, N$, so the local NameCert surface begins with a finite inventory rather than an external real construction.

The computational obligations are read in order. Dyadic gap exposure comes from D, window discipline from W, regular rational readback from R, and the upper-bound or lower-approximation consumer handoff from S before the terminal seal E can be consumed.

Transport and replay do not enlarge the inventory. The rows H and C preserve the same displayed data, while P and N record provenance and local naming, so the listed obligations exhaust the accepted packet. □

Theorem 2852.4 (Dedekind gap RegSeqRat boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *packet factors through the finite route*

$$L, U \longmapsto D \longmapsto W \longmapsto R \longmapsto S \longmapsto E.$$

The route exposes a located lower-upper gap boundary that $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers may use for comparison work. It does not identify cuts by quotient equality, select a representing real, assert a Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, or import ambient order completeness.*

Proof. Start with the lower and upper ledgers L, U. A consumer sees only the displayed finite cells, so the request is a boundary read rather than an equality of cuts.

The dyadic row D supplies the gap witness requested at the displayed precision, and W records the finite observation window in which that witness is inspected. The readback row R then repacks the same window as regular rational sequence data.

The located-supremum handoff S and terminal seal E remain downstream of those rows. Consequently every RegSeqRat or Real-facing read is a routed boundary read, with no coordinate for the excluded quotient, selection, construction, or completeness claims. □

Falsifiable prediction: An accepted $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ read is refuted if a RegSeqRat or Real-facing consumer can remove the lower ledger, upper ledger, dyadic gap row, StreamName window, RegSeqRat readback, LocatedSupremum handoff, Real seal, transport, replay, provenance, or local naming while preserving the same packet claim.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedsupremum: $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower and upper cut ledgers, dyadic gap witnesses, finite observation windows, regular rational readback, located-supremum handoff, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ packet is generated from lower-cut ledgers, upper-cut ledgers, dyadic gap witnesses, StreamName windows, RegSeqRat readback, a LocatedSupremum boundary handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3824.1, Definition 3824.2, Definition 3824.3, and Definition 3824.4 seed the L10 located-cut boundary consumed by RegSeqRat and Real-facing comparison packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Dedekind construction of RealUp, quotient equality of cuts, selected real representatives, trichotomy, countable choice, ambient order completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper gap exposure, dyadic tolerance transport, StreamName window discipline, RegSeqRat readback, LocatedSupremum handoff, and Real seal nonescape for BEDC.Derived.DedekindGapLocatedCutUp.

Chapter 2853

A Concrete Naming Certificate for $\mathsf{FastCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FastCauchyCriterion}^{\uparrow}$ packet records the finite criterion data by which displayed fast Cauchy windows, monotone fast-modulus control, regular rational readback, and the terminal real-name seal may be inspected together before a Real or Completion consumer reads the result. It sits between the fast sequence surface of chapter 2185, the fast modulus discipline of chapter 1781, the regular rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is criterion data for a displayed regular-Cauchy route; it is not quotient real equality, a selected representative, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2853.1 (FastCauchyCriterion carrier)**.** *A* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row recording the inspected sequence values used by the criterion.*
2. *D is the dyadic tolerance scale row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, against which the fast-window comparisons are measured.*
3. *M is the monotone* $\mathsf{FastCauchyModulus}^{\uparrow}$ *row, displaying the threshold discipline that sends a requested tolerance to a later inspected window and preserves monotonicity of the request ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row, recording that the displayed windows and dyadic bounds form the regular rational readback consumed by Real and Completion routes.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, available only after S, D, M, R have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerance scale, monotone modulus, regular rational readback, and Real seal handoff.*
7. *C records displayed* Cont *replay for window inspection, tolerance comparison, monotone threshold use, RegSeqRat compatibility, and Real seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{FastCauchyCriterion}^{\uparrow}$*,* $\mathsf{FastCauchySequence}^{\uparrow}$*,* $\mathsf{FastCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$ *row.*

The carrier exports no quotient equality of real names, no selected representative of a Cauchy class, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FastCauchyCriterionUp.FastCauchyCriterionTasteGate single carrier alignment`

Theorem 2853.2 (FastCauchyCriterion NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier* $K = (S, D, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$ *surface consists exactly of finite StreamName window admission, dyadic tolerance-scale admission, monotone fast-modulus threshold exposure, RegSeqRat compatibility, RealUp seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The rows S, D, M, R, E are the only analytic rows available to a criterion consumer, and they appear in the order needed by the fast Cauchy route: finite windows first, dyadic tolerances next, the monotone threshold ledger next, regular rational readback next, and the Real seal last.

Read the sibling routes through provenance. The row M is the visible threshold discipline of chapter 1781, while S, D, R are the finite window, dyadic tolerance, and regular rational surfaces already used by chapter 2185.

The structural rows do not add analytic content. The transport row H moves only the displayed coordinates componentwise, the replay row C repeats the same threshold and comparison route, and P, N source and name that finite surface. Thus no NameCert obligation remains outside $S, D, M, R, E, H, C, P, N$. □

Theorem 2853.3 (FastCauchyCriterion modulus readback)**.** *Let* $K = (S, D, M, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier. Every requested precision read obtains its fast threshold from* M*, opens only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows recorded in* S*, checks the dyadic bound through* D*, and hands the resulting regular schedule to* R *before any* $\mathsf{Real}^{\uparrow}$ *seal in* E *is consumed. The readback is therefore a finite criterion route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a selected representative or an ambient completeness argument.*

Proof. Start with the monotone modulus row. The request is sent to the displayed threshold in M, and monotonicity keeps later precision requests inside the same finite threshold ledger rather than invoking a choice sequence.

Open the finite windows through S and compare them through D. These rows provide exactly the inspected values and dyadic tolerance account needed for a fast Cauchy criterion, so the route remains inside the displayed BHist packet.

Finally read R and then E. The $\mathsf{RegSeqRat}^{\uparrow}$ compatibility row consumes the same windows and tolerances, and the $\mathsf{Real}^{\uparrow}$ handoff is downstream of that compatibility row. Componentwise transport, replay, provenance, and local naming preserve the same route and add no quotient equality, selected representative, countable choice, or ambient completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{FastCauchyCriterion}^{\uparrow}$ precision read exposes one finite StreamName window row, one dyadic tolerance-scale row, one monotone FastCauchyModulus row, one RegSeqRat compatibility row, one RealUp seal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer may read the criterion.

Independence witness: fastcauchysequence, fastcauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{FastCauchyCriterion}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries windows, tolerance scale, monotone threshold discipline, regular readback, Real seal handoff, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyCriterion}^{\uparrow}$ packet is generated from finite StreamName windows, a dyadic tolerance scale, a monotone FastCauchyModulus row, RegSeqRat compatibility, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3825.1, Definition 3825.2, and Definition 3825.3 seed the finite criterion surface connecting fast Cauchy sequence windows, fast modulus thresholds, RegSeqRat readback, and Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{FastCauchyCriterion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, monotone modulus exposure, dyadic tolerance readback, RegSeqRat compatibility, Real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.FastCauchyCriterionUp

Chapter 2854

A Concrete Naming Certificate for $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$

Following Mac Lane's Kleisli-category formulation, the $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ packet records the finite category-level surface of the L10 completion monad. It sits after the completion monad of chapter 378, the Kleisli-arrow packet of chapter 1845, the universal-property surface of chapter 397, and the finite real-name route through chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names a category of completion consumers through displayed finite rows; it is not a quotient completion, not a selected limit, not host equality of completed objects, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2854.1 (CauchyCompletionKleisliCategory carrier)**.** *A* $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, U, A, B, E, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the object row, reading each completion-facing object through* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *and the regular rational name surface.*
2. *U is the unit-arrow row, recording the monad unit route from displayed source rows into the completion packet.*
3. *A is the Kleisli-arrow row, whose arrows are completion-valued maps exposed by* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$*.*
4. *B is the bind-composition row, recording composition only through the monad bind schedule and the displayed Kleisli route.*
5. *E is the identity and associativity ledger, comparing unit and bind composites as finite BEDC routes rather than as host function equalities.*
6. *R is the real-completion handoff row, routing completion consumers through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seals.*
7. *H records componentwise* hsame *transport for object, unit, arrow, bind, ledger, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for unit-arrow reads, bind composition, completion consumers, and terminal* $\mathsf{Real}^{\uparrow}$ *seal requests.*

9. *P is the* Pkg *provenance over* CauchyCompletionKleisliCategory$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionKl CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyCompletionKleisliCategory}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, selected limits, host equality of arrows, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyCompletionKleisliCategoryUp.CauchyCompletionKleisliCategoryUp`

Theorem 2854.2 (CauchyCompletionKleisliCategory NameCert obligations)**.** *For every accepted* CauchyCompletionKleisliCategory$^{\uparrow}$ *carrier $K = (O, U, A, B, E, R, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionKleisli}}$ *surface consists exactly of object exposure, unit-arrow exposure, Kleisli-arrow exposure, bind-composition exposure, identity and associativity ledger exposure, real-completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3826.1. The object, unit, arrow, bind, ledger, and real-completion handoff rows are precisely O, U, A, B, E, R.

The unit-arrow row is read through CauchyCompletionMonad$^{\uparrow}$, while the arrow and bind rows are read through CauchyCompletionKleisli$^{\uparrow}$. The ledger E compares the category identities and associative composites only after those rows have been displayed.

The real-facing handoff R routes every completion consumer through CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ seals. Thus a consumer cannot read a terminal real-completion claim before the finite Cauchy-name and dyadic rows are visible.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since no quotient completion, selected representative, selected limit, host equality, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 2854.3 (CauchyCompletionKleisliCategory real-completion handoff)**.** *Every* Real$^{\uparrow}$ *or completion-facing consumer of an accepted* CauchyCompletionKleisliCategory$^{\uparrow}$ *packet factors through the ordered route*

CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionKleisli$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, D

The route may use unit arrows and bind composition inside the Kleisli category, but it exports no quotient completion, selected limit, host equality of arrows, selected representative, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. First read the monad and Kleisli rows. The unit arrows come from U, and the composition row B uses the bind schedule attached to the displayed Kleisli-arrow row A.

Next read the category ledger. The row E records the identity and associativity comparisons as finite route comparisons over O, U, A, B, so the categorical equations are consumed as BEDC ledger data rather than as host equality of functions.

Finally pass to real-completion consumers through R. The handoff row requires the universal-property surface and the finite name route through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ before any terminal seal is available.

The structural rows H, C, P, N keep this route inside the displayed packet. There is no carrier coordinate that could supply a quotient completion, selected limit, host arrow equality, selected representative, or ambient completeness theorem. □

Falsifiable prediction: A $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ completion consumer that reads a $\mathsf{Real}^{\uparrow}$ seal without first exposing the monad, Kleisli-arrow, universal-property, stream-window, regular-rational, dyadic-tolerance, transport, replay, provenance, and local NameCert rows refutes the packet.

Independence witness: cauchycompletionmonad, cauchycompletionkleisli, completionuniversal-property, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ because the category packet jointly carries objects, unit arrows, Kleisli arrows, bind composition, categorical ledgers, completion handoff, transport, replay, provenance, and local naming in one finite carrier.

Lean follow-up requires a finite carrier namespace at `BEDC.Derived.CauchyCompletionKleisliCategoryUp` with the displayed object, unit, Kleisli-arrow, bind-composition, ledger, handoff, transport, replay, provenance, and naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ packet is generated from completion-object rows, unit-arrow rows, Kleisli-arrow rows, bind-composition rows, identity and associativity ledgers, real-completion handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3826.1, Definition 3826.2, and Definition 3826.3 seed the finite Kleisli-category completion packet for L10 consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality of arrows, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-arrow exposure, Kleisli-arrow exposure, bind-composition exposure, categorical ledger exactness, real-completion handoff, transport stability, replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyCompletionKleisliCategoryUp.

Chapter 2855

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$

The $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ packet realizes the substitution-sensitive confluence frontier of chapter 9378 as finite checker data. It sits beside the substitution-boundary packet of chapter 3791, the bounded residual frontier of chapter 3808, the critical-pair boundary of chapter 2212, and the local diamond witness of chapter 3595. The packet records which typed windows, redex frontiers, local joins, obstruction rows, and replay routes have actually been checked before a confluence consumer may cite the conditional handoff.

Falsifiable prediction: A MetaCIC confluence read that treats a residual-substitution peak as checked while omitting the finite typed-window, redex-frontier, bounded-checker, local-join, obstruction, transport, replay, provenance, and local NameCert rows refutes $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$.

Independence witness: metacic-confluence-substitution-boundary, bounded-residual-substitution-frontier, metacic-confluence-critical-pair-boundary, metacic-local-diamond-witness: none is bijective to $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ because the checker packet jointly carries typed windows, redex frontiers, bounded checker outcomes, local joins, obstruction retention, transport, replay, provenance, and local naming as one finite confluence frontier.

Definition 2855.1 (MetaCIC confluence residual-checker carrier)**.** *A* $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, R, B, J, O, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row containing the source term, the two parallel reducts, the residual substitution window, and the candidate common reduct.*
2. *R is the redex-frontier row, listing the inspected beta positions, binder depths, residual descendants, and free-variable exposure checks.*
3. *B is the bounded-checker row, recording the finite closedness, substitution-support, variable-escape, and branch-compatibility checks.*
4. *J is the local-join row, storing the finite replay path to the candidate common reduct when the checker accepts.*

5. *O is the obstruction row, naming the still-visible endpoints: unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, and kernel consistency.*
6. *S is the checker-status row, recording whether the inspected residual window is accepted, rejected, or left at the named obstruction boundary.*
7. *H records componentwise* hsame *transport for typed windows, redex frontiers, bounded checks, local joins, obstruction rows, checker status, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence-frontier query through $T, R, B, J, O, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$, $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for full confluence, unrestricted residual-substitution compatibility, hidden conversion, evaluator equality, subject reduction, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceResidualCheckerUp.MetaCICConfluenceResidualCheckerTasteGate`
`single carrier alignment`

Theorem 2855.2 (MetaCIC confluence residual-checker obligations)**.** *For every accepted* MetaCICConfluenceResidu *carrier $Q = (T, R, B, J, O, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, redex-frontier exposure through R, bounded checking through B, local-join exposure through J, obstruction retention through O, checker-status exposure through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3827.1. The rows T, R, B, J, O, S are the checker-facing data, and the rows H, C, P, N are the structural certificate data.

The checker begins at T and R. These rows display the finite typed peak, the residual substitution window, the inspected redex positions, binder-depth ledger, and free-variable exposure checks.

The row B records the bounded decision surface, while J records only the local candidate join made available by that bounded surface. The checker-status row S separates accepted windows, rejected windows, and windows that remain at the named substitution obstruction.

The obstruction row O prevents the bounded read from being cited as global confluence. Transport, replay, provenance, and local naming preserve the same finite packet, and no hidden conversion, evaluator equality, subject reduction, or kernel-consistency coordinate is present. □

Theorem 2855.3 (MetaCIC confluence residual-checker handoff)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ *carrier factors through the finite route*

$$T, R, B \longmapsto J, S, O \longmapsto H, C, P, N.$$

The route may hand off a checked residual window to the substitution-boundary, bounded-frontier, critical-pair, and local-diamond packets named above, but it does not supply unrestricted substitution compatibility, a premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, or kernel consistency.

Proof. First read the finite window. The rows T and R expose the typed peak, residual substitution window, redex frontier, binder-depth ledger, and free-variable checks.

Next apply the bounded checker row B. If the finite checks accept, the local join row J and checker-status row S expose the handoff; if they reject or stop at the boundary, S and O keep the obstruction visible.

Finally route the handoff through H, C, P, N. These rows transport, replay, source, and name the same finite packet, and the provenance row points only to the substitution-boundary, bounded-frontier, critical-pair, and local-diamond packets.

The carrier therefore has no path to a global confluence theorem or hidden substitution theorem. Lean follow-up requires Nontrivial and FieldFaithful instances for `BEDC.Derived.MetaCICConfluenceResidualChec` at exactly this displayed carrier surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ packet is generated from typed-window rows, redex-frontier rows, bounded-checker rows, local joins, checker-status rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3827.1, Definition 3827.2, and Definition 3827.3 seed the finite residual-substitution checker packet for MetaCIC confluence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, redex-frontier coverage, bounded-checker exactness, local-join exposure, checker-status determinacy, obstruction retention, transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MetaCICConfluenceResidualChec

Chapter 2856

A Concrete Naming Certificate for $\mathsf{DarbouxContinuity}^{\uparrow}$

Darboux/Bishop–Bridges constructive analysis treats interval continuity as a finite route from a closed bounded source interval, through an explicitly continuous graph, into bracketed intermediate-value data before any completed real image is read. The $\mathsf{DarbouxContinuity}^{\uparrow}$ packet records that route over chapter 1591, chapter 130, chapter 1402, chapter 291, chapter 301, and chapter 38. It is a local continuity certificate for L10 interval-image consumers, not an open-cover compactness principle, not quotient real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2856.1 (DarbouxContinuity carrier)**.** *A* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, G, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval source row, carrying endpoint order, containment, finite-net source, and source-obligation data from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*.*
2. *G is the continuous-map graph row over the interval source, recording the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus, graph, and classifier-compatible point route.*
3. *V is the intermediate-value crossing row, recording the admitted sign or order crossing data read through* $\mathsf{IntermediateValue}^{\uparrow}$*.*
4. *D is the dyadic bracket ledger from* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording finite rational brackets for endpoints, graph samples, crossing windows, and tolerance budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the bracketed endpoint, sample, and crossing observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after I, G, V, D, R have been displayed.*
7. *H is the componentwise* hsame *row for the interval, graph, crossing, bracket, readback, seal, replay, provenance, and local naming data.*
8. *C records displayed* Cont *replay from any interval-continuity consumer through I, G, V, D, R, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DarbouxContinuity}^{\uparrow}$*,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{IntermediateValue}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open covers, selected subcovers, quotient real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DarbouxContinuityUp.DarbouxContinuityTasteGate single carrier alignment`

Theorem 2856.2 (DarbouxContinuity NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$ *surface consists exactly of closed-interval source admission, continuous-map graph exposure, intermediate-value crossing exposure, dyadic bracket accounting, RegSeqRat readback, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier rows. The interval source I, graph row G, crossing row V, dyadic ledger D, readback row R, and seal row E are the only analytic rows available to a consumer.

Read the structural rows separately. The transport row H preserves those analytic rows componentwise, while C replays the same displayed route for interval-continuity consumers.

The provenance and naming rows P, N attach the packet to the cited closed-interval, continuous-map, intermediate-value, dyadic, regular-readback, and Real certificates. No excluded cover, quotient, trichotomy, choice, or completeness coordinate is present. □

Theorem 2856.3 (DarbouxContinuity source route)**.** *Every* $\mathsf{DarbouxContinuity}^{\uparrow}$ *consumer that reads an interval-image or crossing value must first enter through the closed bounded interval source of chapter 1591, then through the continuous graph of chapter 130, then through the crossing row of chapter 1402, and only then through the dyadic, RegSeqRat, and Real rows of chapter 291, chapter 301, and chapter 38.*

Proof. Start with the source interval row I. Its endpoint, containment, finite-net, and readback obligations supply the finite interval boundary before any graph or image row is available.

Next read the graph and crossing rows. The continuous-map row G supplies the displayed modulus and graph route, and V supplies the intermediate-value crossing data relative to that graph.

Finally expose D, R, E through the same route. The dyadic bracket ledger, regular readback, and Real seal are downstream of I, G, V, so a consumer cannot start from an ambient interval image or a completed real-value selection. □

Theorem 2856.4 (DarbouxContinuity graph exposure)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* G *is exposed only after the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row* I *has been displayed. Any crossing read through* V *depends on that graph exposure and therefore cannot be read as an interval-image fact detached from the source interval and continuous-map graph.*

Proof. Project the carrier of Definition 3828.1. The row I is the source interval coordinate, and G is explicitly described as the continuous-map graph row over that source.

Now inspect the crossing row. The row V records intermediate-value crossing data read through the graph row G, so a consumer of V must have already displayed both the source interval row and the continuous graph row.

The structural rows H, C, P, N only transport, replay, source, and name this route. They add no detached image coordinate, so every accepted crossing read remains dependent on the I then G exposure order. □

Theorem 2856.5 (DarbouxContinuity Real-seal handoff)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *row* E *is terminal. It may be read only after the dyadic bracket ledger* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *expose the bracketed endpoint, sample, and crossing observations used by the intermediate-value route.*

Proof. Project the analytic rows. The dyadic ledger D stores the finite rational brackets, and the row R is the regular rational readback for those bracketed observations.

The seal row is downstream. In Definition 3828.1, E is exposed only after I, G, V, D, R have been displayed, so the completed real-name read cannot precede the bracket ledger or its regular readback.

Transport and replay preserve this terminal position. The rows H, C, P, N can reread the same sealed route for a consumer, but they cannot supply an ambient Real value without the displayed dyadic and regular-rational handoff. □

Theorem 2856.6 (DarbouxContinuity non-escape)**.** *Every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *packet exports only the finite rows*

$$I, G, V, D, R, E, H, C, P, N.$$

It exports no host interval image, open-cover compactness object, selected subcover, selected point, quotient real equality, trichotomy decision, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the packet and inspect the carrier inventory. The analytic rows are the closed interval source, continuous graph, intermediate-value crossing, dyadic bracket ledger, regular readback, and terminal Real seal.

The structural rows do not add analytic content. They only transport, replay, source, and name the same finite route through H, C, P, N.

Since no carrier coordinate stores host compactness, quotient equality, a selected point, trichotomy, choice, or ambient completeness, no accepted consumer can extract such data from the Darboux continuity certificate. □

Falsifiable prediction: An accepted $\mathsf{DarbouxContinuity}^{\uparrow}$ read is refuted if it exposes an interval-image or crossing value without first displaying the closed interval source, continuous graph, intermediate-value crossing, dyadic bracket, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: closedboundedinterval, continuousmap, intermediatevalue, dyadicratcore, regseqrat, real: none alone is bijective with $\mathsf{DarbouxContinuity}^{\uparrow}$ because the packet jointly records an interval source, continuous graph, crossing ledger, dyadic brackets, regular readback, Real seal, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DarbouxContinuity}^{\uparrow}$ packet is generated from a closed bounded interval source, a continuous-map graph, intermediate-value crossing data, a dyadic bracket ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 3828.1, Definition 3828.2, Definition 3828.3, and Definition 3828.6 bind the L10 interval-continuity carrier to closed-interval, continuous-map, intermediate-value, dyadic, regular-readback, and Real-seal source rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient real equality, trichotomy, selected interval-image points, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, graph exposure, crossing-route exactness, dyadic bracket readback, Real-seal handoff, and non-escape for BEDC.Derived.DarbouxContinuityUp.

Chapter 2857

A Concrete Naming Certificate for $\mathsf{ConcaveModulus}^{\uparrow}$

Following Bishop and Bridges on constructive moduli of continuity, the $\mathsf{ConcaveModulus}^{\uparrow}$ packet records the finite normalization surface at which a positive tolerance schedule is replaced by a monotone midpoint-concave modulus before Real or Completion consumers inspect it. It sits on the rational and dyadic tolerance rows of chapter 37 and chapter 291, the explicit uniform-modulus surface of chapter 2228, the located-uniform-continuity handoff of chapter 1859, and the terminal Real seal of chapter 38. The packet names a constructive normalization object; it does not choose a modulus from a host theorem, assert real completeness, or import quotient equality of functions.

Definition 2857.1 (Concave modulus carrier)**.** *A* $\mathsf{ConcaveModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, M, J, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row. It names the rational tolerance requests and records the finite order in which a consumer may inspect them.*
2. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same requests, giving the dyadic scale at which Real and Completion routes read the modulus.*
3. *M is the monotone modulus row. It records that weaker tolerance requests are served by no stronger source radius than the displayed stronger requests.*
4. *J is the midpoint-concavity row. It records the finite Jensen midpoint comparisons used by the normalization route before any subadditivity consumer may read the packet.*
5. *S is the subadditive handoff row. It exposes the normalized modulus as the public row consumed by* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *and* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *certificates.*
6. *H stores componentwise* hsame *transport for the rational tolerance, dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a modulus-normalization request through Q, D, M, J, S, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ConcaveModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformContinuityModulus}^{\uparrow}$, $\mathsf{LocatedUniformContinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected modulus, host continuity theorem, quotient function equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusTasteGate single carrier alignment`

Theorem 2857.2 (Concave modulus NameCert obligations)**.** *For every accepted* $\mathsf{ConcaveModulus}^{\uparrow}$ *carrier*

$$K = (Q, D, M, J, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the positive rational tolerance, positive dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, transport, replay, provenance, and local naming rows;*
2. *tolerance exposure, requiring every normalization request to read* Q *and* D *before it reads any modulus row;*
3. *monotonicity discipline, requiring* M *to preserve the displayed order of tolerance requests without choosing a new host-scale modulus;*
4. *midpoint-concavity discipline, requiring* J *to supply the finite midpoint comparisons used by the subadditive handoff;*
5. *non-escape, requiring* S, H, C, P, N *to export only the displayed normalized modulus route and no choice-selected modulus, quotient function equality, trichotomy, ambient completeness, Lean verification, or bridge checking.*

No $\mathsf{ConcaveModulus}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusCarrier namecert obligations`

Proof. Project the carrier of Definition 3829.1. The tuple lists exactly the positive rational tolerance, dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, and structural rows.

Tolerance exposure is forced by the order of Q and D. A consumer cannot inspect the normalized modulus before the positive rational request and dyadic scale have been displayed as finite packet data.

The monotone and midpoint-concavity obligations are the rows M and J. They are read before the public handoff row S, so the handoff is a displayed normalization route rather than a host theorem about all moduli.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since the carrier contains no coordinate for a selected modulus, quotient function equality, trichotomy, ambient completeness, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 2857.3 (Concave modulus subadditive handoff)**.** *Every* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *or* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{ConcaveModulus}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{ConcaveModulus}^{\uparrow} \longrightarrow \mathsf{UniformContinuityModulus}^{\uparrow} \longrightarrow \mathsf{LocatedUniformContinuity}^{\uparrow}$$

through the displayed rows $Q, D, M, J, S, H, C, P, N$*. The subadditive handoff row* S *is available only after the positive tolerance rows, monotone modulus row, and midpoint-concavity row have been displayed. The route exports no choice-selected modulus, host continuity theorem, quotient equality of functions, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with an accepted carrier $K = (Q, D, M, J, S, H, C, P, N)$. Projection exposes the tolerance rows Q, D before any modulus-normalization row is consumed.

Read the monotone row M and midpoint-concavity row J. These rows are finite packet data, so the consumer receives the normalization discipline as displayed comparisons rather than as a global host theorem.

The handoff row S is downstream of those rows. A $\mathsf{UniformContinuityModulus}^{\uparrow}$ consumer reads S as its public tolerance-to-radius surface, and a $\mathsf{LocatedUniformContinuity}^{\uparrow}$ consumer reads the same handoff only after its located source and modulus routes have been displayed.

The remaining rows H, C, P, N preserve componentwise transport, replay, provenance, and naming for this route. Since no listed row contains a selected modulus, quotient function equality, trichotomy, or ambient completeness coordinate, the subadditive handoff exports none of them. □

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusCarrier subadditive handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3829.1, Definition 3829.2, and Definition 3829.3 seed the finite constructive normalization route from positive rational and dyadic tolerances through monotone midpoint-concave modulus rows into uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a choice-selected modulus, host continuity theorem, quotient function equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive tolerance exposure, monotone modulus discipline, midpoint-concavity rows, subadditive handoff, transport, replay, provenance, and local naming for BEDC.Derived.ConcaveModulusU

Constructive story: A $\mathsf{ConcaveModulus}^{\uparrow}$ packet is generated from positive RatUp tolerance rows, positive DyadicRatCoreUp tolerance rows, monotone modulus rows, midpoint-concavity rows, a subadditive handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2858

A Concrete Naming Certificate for $\mathsf{SignedDigitIntervalReal}^{\uparrow}$

Following Avizienis and Bishop–Bridges, the $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ packet records a closed real interval whose endpoints are supplied by signed-digit finite-window streams, normalized dyadic endpoint rows, StreamName schedules, RegSeqRat readbacks, and a terminal RealUp seal. It consumes the signed-digit stream surface of chapter 316, the signed-digit real endpoint route of chapter 363, the dyadic normalization ledger of chapter 291, the finite schedule interface of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the closed bounded interval carrier of chapter 1591. The packet gives Real and Completion consumers a finite-window interval source without exporting a quotient stream, a selected infinite expansion, or ambient real completeness.

Definition 2858.1 (Signed digit interval real carrier)**.** *A* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower signed-digit endpoint row, carrying the finite signed-digit windows admitted by the signed-digit stream and signed-digit real routes.*
2. *U is the upper signed-digit endpoint row, carrying the corresponding finite signed-digit windows for the upper endpoint.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *normalization ledger, recording the dyadic endpoint rows obtained from the inspected signed-digit windows.*
4. *O is the endpoint-order row, recording that the normalized lower endpoint is contained below the normalized upper endpoint through the displayed dyadic comparisons.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, listing exactly the endpoint windows and precision requests inspected by the packet.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the normalized lower and upper endpoint observations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the signed-digit endpoint rows, dyadic ledger, order row, schedule, and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint rows, dyadic ledger, order row, schedule, readback, seal, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay from signed-digit windows through dyadic normalization, endpoint ordering, regular readback, and the RealUp seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDigitReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, O, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite signed-digit expansion, real trichotomy, a selected point of the interval, open-cover compactness, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SignedDigitIntervalRealUp.SignedDigitIntervalRealTasteGate single carrier alignment`

Theorem 2858.2 (Signed digit interval real NameCert obligations)**.** *For every accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier*

$$I = (L, U, D, O, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$ *obligations are exhausted by carrier admission for the eleven displayed rows; signed-digit endpoint normalization through L, U, D; dyadic containment through the normalized endpoint ledger D; endpoint-order preservation through O; regular readback through R along the finite schedule S; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

No obligation may call on quotient stream equality, a selected infinite expansion, trichotomy, a selected point of the interval, open-cover compactness, countable choice, or ambient RealUp completeness.

Proof. Project the carrier of Definition 3830.1. The endpoint rows L and U are the only signed-digit sources available to the packet, and the schedule row S records the finite windows through which those sources are inspected.

The dyadic and order obligations are then read through D and O. The ledger D normalizes the inspected endpoint windows into dyadic rows, while O records the displayed lower-before-upper comparison needed by the closed interval carrier.

The regular readback and seal obligations pass through R and E. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local certificate and leave the excluded principles outside the packet. □

Theorem 2858.3 (Signed digit interval real seal nonescape)**.** *An accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier exports its interval-facing RealUp seal only through*

$$L, U, D, O, S, R, E, H, C, P, N.$$

Consequently every consumer of the packet receives a finite signed-digit endpoint route, dyadic containment ledger, endpoint-order row, regular readback, and closed-interval seal; it receives no quotient stream equality, selected infinite expansion, trichotomy, selected point of the interval, open-cover compactness, countable choice, or ambient RealUp completeness.

Proof. Start with the endpoint rows. The lower and upper endpoints are visible only as the finite signed-digit windows displayed by L and U along the schedule S.

Normalize those endpoint windows through D and compare them through O. This is the only place where the carrier records containment of the lower endpoint below the upper endpoint, and it is expressed as a finite dyadic ledger rather than as trichotomy or quotient real equality.

The RealUp-facing read then passes through R, E, H, C, P, N. Since the carrier has no further public coordinates, any interval consumer is confined to the displayed finite route and none of the excluded selection, compactness, choice, or completeness principles can escape through the seal. □

Theorem 2858.4 (Signed digit interval real endpoint-order nonescape)**.** *Let*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier. Any consumer read that uses interval containment or the lower-before-upper comparison must expose the finite signed-digit endpoint rows L, U, the finite schedule S, the dyadic normalization ledger D, and the endpoint-order row O before it can pass to the readback and seal rows R, E.*

Consequently the order-facing read exports only finite dyadic endpoint comparison data. It exports no real trichotomy, quotient real equality, selected endpoint, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier displayed in Definition 3830.1. The rows L and U are the only endpoint-window sources, and S is the only row that records which finite windows and precision requests the packet has inspected.

The order-facing data then factors through D and O. The ledger D is the unique dyadic normalization row for the inspected endpoint windows, while O is the only coordinate that records the lower-before-upper comparison.

The rows R, E, H, C, P, N are downstream readback, seal, transport, replay, provenance, and naming rows. They add no further order coordinate, so every public order-facing read remains confined to the finite endpoint windows, their dyadic normalization ledger, and the displayed endpoint-order row. □

Theorem 2858.5 (Signed digit interval real refinement stability)**.** *Let*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

and

$$I' = (L', U', D', O', S', R', E', H', C', P', N')$$

be accepted $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carriers whose displayed rows are related componentwise by the transport row* hsame *on L, U, D, O, S. For any smaller signed-digit endpoint window and any dyadic tolerance refinement admitted by the finite schedule, the refined read factors again through the rows*

$$L, U, D, O, S, R, E, H, C, P, N.$$

The factorization preserves the endpoint-order row and the terminal seal without using trichotomy, selected signed-digit expansions, quotient stream equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the two accepted carriers. The componentwise hsame data identifies the endpoint rows L, U, the normalization ledger D, the endpoint-order row O, and the schedule row S as the only coordinates available for comparing the original and refined reads.

Refine the schedule row S to the smaller endpoint windows and requested dyadic tolerance. The refined observations are still signed-digit windows from L and U, so their normalization is replayed through the same dyadic ledger shape D rather than through a quotient stream or a selected expansion.

The order row O is transported along the displayed hsame data. Thus the refined dyadic endpoint comparison is the same finite lower-before-upper certificate, only read at the smaller window and tolerance.

Finally repack the refined route through R, E, H, C, P, N. The readback, seal, transport, continuation, provenance, and naming rows carry the refined finite route and add no coordinate from trichotomy, choice, compactness, quotient equality, or ambient completeness. □

Theorem 2858.6 (Signed digit interval real RegSeqRat seal route)**.** *Every Real-facing endpoint or interval read of an accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *normalization ledger D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E. The route exports no selected infinite signed-digit expansion and no completeness principle.*

Proof. Use the carrier of Definition 3830.1. Its only Real-facing coordinates are arranged after the endpoint rows, the schedule, the dyadic normalization ledger, and the regular rational readback row.

Apply Definition 3830.2. The local certificate lists regular readback along S and terminal RealUp sealing through E as obligations downstream of the signed-digit endpoint and dyadic normalization rows.

Apply Definition 3830.3. Since the seal exports only the displayed route $L, U, D, O, S, R, E, H, C, P, N$, any endpoint or interval read must pass through S, D, R before E and cannot expose an infinite expansion, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ packet is generated from lower and upper signed-digit endpoint rows, a DyadicRatCore normalization ledger, an endpoint-order row, StreamName finite schedules, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3830.1, Definition 3830.2, and Definition 3830.3 seed the finite signed-digit endpoint route for closed RealUp intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite signed-digit expansions, trichotomy, selected interval points, open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint normalization, dyadic containment, endpoint-order preservation, regular readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SignedDigitIntervalRealUp.

Chapter 2859

A Concrete Naming Certificate for $\mathsf{CayleyTheorem}^{\uparrow}$

Following Cayley, the $\mathsf{CayleyTheorem}^{\uparrow}$ packet records the finite left-regular action of a displayed group row as a permutation-action row. It consumes the finite $\mathsf{Group}^{\uparrow}$ source of chapter 41 and the finite-position bijection surface of chapter 116. The packet names the carrier, action, image, faithfulness, hsame, Cont, Pkg, and NameCert rows needed to read the theorem as BEDC data; it does not assert a classification of finite groups, a presentation by generators and relations, or a representation-theoretic decomposition.

Definition 2859.1 (Cayley theorem carrier)**.** *A* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (G, L, I, F, H, T, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, including the carrier, multiplication, unit, inverse, and finite support surfaces;*
2. *L is the left-regular action row sending a displayed group element g and a displayed group endpoint x to the product endpoint gx;*
3. *I is the permutation-image row, reading each left-translation row as a certified self-bijection of the finite carrier of G through the* $\mathsf{Permutation}^{\uparrow}$ *interface;*
4. *F is the faithfulness row, recording that if two left-translation rows have the same action on every displayed endpoint, then their source group rows classify together by the* $\mathsf{Group}^{\uparrow}$ *classifier;*
5. *H is the componentwise* hsame *transport for G, L, I, F;*
6. *T stores the displayed* Cont *routes through which consumers read the left action, permutation image, and faithfulness rows;*
7. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host embedding into an ambient symmetric group, no quotient by an external equality of functions, and no representation category.

Lean4 ✓ `BEDC.Derived.CayleyTheoremUp.TasteGate.CayleyTheoremTasteGate single carrier alignment`

Theorem 2859.2 (Cayley theorem NameCert obligations)**.** *For every accepted* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier*

$$C = (G, L, I, F, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite group row, left-regular action row, permutation-image row, faithfulness row, componentwise transport row, continuation routes, provenance, and local naming row. The obligation route is finite group row, left-regular action row, permutation image row, faithfulness row, and then the* hsame, Cont, Pkg, *and* NameCert *packaging rows.*

Proof. Project the carrier of Definition 3831.1. Carrier admission is the displayed tuple G, L, I, F, H, T, P, N, so no host symmetric group, host function equality, or representation category is available as an additional coordinate.

Read L from the multiplication row of G. For each displayed source row g, left multiplication by g sends every displayed endpoint x to the endpoint gx and sends the inverse endpoint through the inverse row of G, so the action has the inverse-witness data needed by the permutation surface.

Read I through chapter 116. The finite support row of G supplies the finite spine, while the left-action and inverse rows supply the forward graph, inverse graph, inverse witnesses, and action ledger of the carried $\mathsf{Permutation}^{\uparrow}$ row.

Finally read F at the unit endpoint of G. If two left-translation rows act the same on every displayed endpoint, their actions on the unit endpoint classify the two source group rows together. Repacking this faithfulness row with H, T, P, N gives the stated local NameCert surface. □

Theorem 2859.3 (Cayley theorem permutation-action handoff)**.** *Every downstream consumer of a* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier that asks for a permutation action of a finite group reads exactly the rows*

$$G, L, I, F, H, T, P, N$$

of Definition 3831.1. The handoff exports the consumed $\mathsf{Group}^{\uparrow}$ *row, the left-regular action row, the* $\mathsf{Permutation}^{\uparrow}$ *image row, and the faithfulness row, and then transports them through the displayed* hsame, Cont, Pkg, *and* NameCert *rows. It does not export a bare host function, a separate symmetric-group carrier, or any representation-theoretic decomposition.*

Proof. First apply Definition 3831.2. It identifies the public packet with the finite group row, left-regular action row, permutation-image row, faithfulness row, and the transport and provenance rows.

Next project the permutation-image coordinate through chapter 116. Each left-translation row is read as a certified finite-position self-bijection; the action row is the corresponding endpoint action, and inverse coverage is supplied by the inverse row of the consumed group packet.

Finally use the faithfulness row. The action on the unit endpoint recovers the source group classifier, so the consumer can read the group-to-permutation map as faithful without importing host function equality. Since every public field has already been listed in G, L, I, F, H, T, P, N, the handoff has exactly the displayed BEDC surface and no additional symmetric-group or representation coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3831.1, Definition 3831.2, and Definition 3831.3 seed the finite left-regular action carrier, permutation-image handoff, faithfulness row, and hsame/Cont/Pkg/NameCert packaging over displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Permutation}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a classification of finite groups, generator-and-relation presentations, symmetric-group closure, representation decomposition, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left-action exactness, permutation-image construction, faithfulness at the unit endpoint, and downstream permutation-action consumer boundary for the same finite packet.

Constructive story: A CayleyTheorem$^{\uparrow}$ packet is generated from a displayed finite group source, its left-regular action on its own finite carrier, the induced certified permutation row for each source element, a faithfulness read at the unit endpoint, componentwise transport, continuation routes, provenance, and local NameCert evidence.

Chapter 2860

A Concrete Naming Certificate for $\mathsf{CauchyMeshRefinement}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy data and regular rational approximations, the $\mathsf{CauchyMeshRefinement}^{\uparrow}$ packet records a finite common-refinement mesh for two displayed Cauchy requests before a terminal real-name seal is read. It sits between the finite-window gateway of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal seal of chapter 38, and the tail-budget surface of chapter 2071. The packet supplies later diagonal, tail-budget, and regular-Cauchy arguments with finite mesh data; it does not introduce quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2860.1 (Cauchy mesh refinement carrier)**.** *A* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *Q_0 and Q_1 are the two source Cauchy request rows. Each records the source name, requested precision, finite tail index, and the local tolerance budget visible to the request.*
2. *D_0 and D_1 are the two dyadic mesh rows. They display the finite dyadic partitions*

$$D_i = \{a_{i,0} < a_{i,1} < \cdots < a_{i,k_i}\}$$

   for $i = 0, 1$, together with the endpoint, radius, and comparison cells supplied by $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *J is the common refinement row. It lists the finite dyadic mesh containing every point of D_0 and D_1, the inclusion maps from both meshes, and the comparison ledger showing that the inherited tolerances are below the two source requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source schedules inspected along J.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same displayed windows and the common dyadic refinement.*
6. *T is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row recording that the inherited tail budget is read from J, not from an unlisted completion object.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $Q_0, Q_1, D_0, D_1, J, S, R, T$ have been displayed.*

8. *H records componentwise* hsame *transport for the source requests, dyadic meshes, common refinement, StreamName windows, RegSeqRat readback, tail-bound row, real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from a precision request through*

$$Q_0, Q_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto S \longmapsto R \longmapsto T \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyMeshRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$ *row.*

The classifier accepts only packets in which both displayed meshes factor through the same finite row J and every downstream read of S, R, T, E, H, C, P, N preserves that common refinement. No coordinate stores a quotient stream, selected limit, countable-choice schedule, trichotomy oracle, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyMeshRefinementUp.CauchyMeshRefinementTasteGate single carrier alignment`

Theorem 2860.2 (Cauchy mesh refinement NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier*

$$M = (Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$ *surface consists exactly of source Cauchy request admission, two displayed dyadic mesh ledgers, common-refinement inclusion,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyTailBound}^{\uparrow}$ *budget routing, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows $Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N$.*

Proof. Project the carrier of Definition 3832.1. The two source requests are visible only as Q_0, Q_1, and their dyadic data are visible only as the finite mesh rows D_0, D_1.

The common row J is the only place where the two meshes are compared. Since the inclusion maps and inherited tolerance checks are stored there, every later StreamName, RegSeqRat, and tail-budget consumer must read the same finite refinement before it can inspect a Real-facing seal.

The rows H, C, P, N transport, replay, source, and name this route. The carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, so the listed rows exhaust the certificate surface. □

Theorem 2860.3 (Cauchy mesh refinement RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier must factor through the displayed route*

$$Q_0, Q_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto S \longmapsto R \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently a regular rational readback cannot be replaced by a quotient stream, selected limit, countable-choice schedule, trichotomy decision, host real equality, detached tail witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Proof. Begin with a consumer request for regular rational, tail-budget, or real-seal data. The source Cauchy requests first expose Q_0, Q_1, and the dyadic rows D_0, D_1 display the two finite meshes answering those requests.

The common refinement row J is then forced by carrier admission. It is the unique finite row through which both dyadic meshes pass, so the StreamName windows S and the RegSeqRat readback R are read from the same mesh surface.

The tail-bound row T and Real seal row E are downstream of that readback. Transport, replay, provenance, and local naming preserve the displayed route, and the excluded analytic objects are absent from the tuple. Thus every consumer handoff is the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyMeshRefinement}^{\uparrow}$ packet is generated from two source Cauchy requests, two displayed dyadic meshes, a common refinement row, StreamName windows, RegSeqRat readback, a CauchyTailBound budget row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3832.1, Definition 3832.2, and Definition 3832.3 seed the finite common-refinement mesh surface for L10 Cauchy requests over StreamName, RegSeqRat, DyadicRatCore, CauchyTailBound, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-request admission, dyadic mesh ledgers, common-refinement inclusion, StreamName windows, RegSeqRat readback, CauchyTailBound routing, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMeshRefinementUp.

Chapter 2861

A Concrete Naming Certificate for $\mathsf{MonotoneConvergenceReal}^{\uparrow}$

Following Bishop and Bridges on bounded monotone real sequences, the $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ packet records the finite NameCert surface by which monotone inspection rows, bounded sequence rows, and located-supremum rows meet before any ambient completeness theorem for $\mathsf{Real}^{\uparrow}$ is available. It consumes the bounded sequence carrier of chapter 1415, the located upper-bound surface of chapter 1427, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a located-supremum route for bounded monotone RealUp sequences; it is not a proof of full RealUp completeness, a host supremum construction, a trichotomy decision, quotient real equality, countable choice, or a selected limit operator.

Definition 2861.1 (Monotone convergence real carrier)**.** *A* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, O, U, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row, exposing only the finite sequence schedule, stream windows, regular readback, RealUp seal, and closed bounded interval-net bound already named in Definition 1415.1.*
2. *O is the monotonicity ledger, recording for the inspected index pairs that the displayed sequence windows are nondecreasing through the same comparison route used by the bounded sequence carrier.*
3. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *upper-bound row attached to the bounded range displayed by B and to the comparison rows recorded in O.*
4. *A is the lower-approximation ledger, saying that each requested tolerance window receives a displayed sequence index whose regular-rational readback approaches the located-supremum row from below.*
5. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *precision-window row used to inspect the source sequence, monotonicity ledger, upper-bound row, and approximation ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected windows in W.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, O, U, A, W, R have been displayed.*

8. *H records componentwise* hsame *transport for the bounded source, monotonicity ledger, located-supremum row, approximation ledger, precision windows, regular readback, real seal, replay, provenance, and local naming row.*
9. *C records displayed* Cont *replay from bounded sequence inspection through monotonicity, located-supremum access, lower approximation, regular readback, and the RealUp seal.*
10. *P is the* Pkg *provenance over* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $B, O, U, A, W, R, E, H, C, P, N$. It has no coordinate for full $\mathsf{Real}^{\uparrow}$ *completeness, a host-level supremum, trichotomy, quotient real equality, countable choice, or a selected-limit functional.*

Lean4 ✓ `BEDC.Derived.MonotoneConvergenceRealUp.MonotoneConvergenceRealTasteGate single carrier alignment`

Theorem 2861.2 (Monotone convergence real NameCert obligations)**.** *For every accepted* MonotoneConvergenceRe *carrier $M = (B, O, U, A, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}$ *obligations are exhausted by the following finite rows: carrier admission for the eleven displayed coordinates; bounded-source exposure through B; monotonicity inspection through O; located upper-bound access through U; lower-approximation cofinality through A; shared finite-window accounting through W; regular-rational readback through R; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

No obligation may call on a full completeness principle for $\mathsf{Real}^{\uparrow}$*, a host supremum object, trichotomy, quotient real equality, countable choice, or a selected limit.*

Proof. Project the carrier of Definition 3833.1. The source sequence is visible only through the bounded sequence row B, and the monotonicity information is visible only through the displayed comparison ledger O.

The located-supremum side is then read through U and A. The row U supplies the upper-bound surface, while A supplies the lower approximants requested by the finite window row W.

Finally the readback and seal rows R and E repack the same inspected data as regular-rational and RealUp observations. The structural rows H, C, P, N transport, replay, source, and name this route, so the listed rows exhaust the local certificate. □

Theorem 2861.3 (Monotone convergence real located-supremum route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ *carrier reaches the candidate monotone limit only through*

$$B, O, U, A, W, R, E, H, C, P, N.$$

The route first reads the bounded real sequence row, then the monotonicity ledger, then the located-supremum upper-bound and lower-approximation rows, then the shared finite windows and regular rational readback, and only then the terminal RealUp seal. Consequently the consumer receives a located-supremum route for the displayed bounded monotone sequence rather than full RealUp completeness, a host supremum, trichotomy, quotient real equality, countable choice, or a selected limit.

Proof. Start with the bounded sequence surface. By Definition 1415.1, a downstream consumer can inspect only the finite sequence schedule, stream windows, regular readback, real seal, closed bounded interval-net bound, and structural naming rows carried by B.

Add the monotonicity ledger O. It compares only the inspected windows of the same bounded source, so it cannot introduce an unbounded stream, a quotient sequence, or a host order principle.

Now apply the located-supremum surface of chapter 1427. The rows U and A provide upper-bound access and lower-approximation cofinality for the same windowed route.

The final handoff passes through W, R, E, H, C, P, N. Since these rows are the only remaining public coordinates, the RealUp consumer obtains only the stated located-supremum route and none of the excluded completeness or choice principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a monotonicity ledger, a LocatedSupremumUp upper-bound row, a lower-approximation ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3833.1, Definition 3833.2, and Definition 3833.3 seed the finite located-supremum route for bounded monotone RealUp sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close full RealUp completeness, a host supremum construction, trichotomy, quotient real equality, countable choice, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exposure, monotonicity ledger stability, located-supremum access, lower-approximation cofinality, readback sealing, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneConvergenceRealUp.

$\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ reads unit and multiplication rows through chapter 423, chapter 3836, and chapter 397.

Chapter 2862

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$

Following Mac Lane's monad-law ledger and Bishop–Bridges constructive completion, the CauchyCompletionMonadL packet records the finite BEDC rows by which the Cauchy-completion transformer exposes its unit and associativity laws. It reads the completion monad of chapter 378, the unit route of chapter 1527, the bind ledger of chapter 3777, the idempotence packet of chapter 2381, and the real-and-completion readbacks of chapter 126, chapter 291, chapter 301, and chapter 38. The unit and multiplication law rows consume the functorial completion surface of chapter 423 and the universal-property witness of chapter 397 before associativity is read as law data. The chapter names a law surface for finite completion packets only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

As a dependency relation, the monad-law packet consumes the CauchyCompletionAssociativity route in chapter 1635 and the regular completion route in chapter 1639 before exporting unit and bind compatibility.

Definition 2862.1 (CauchyCompletionMonadLaw carrier)**.** *A* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, U, I, B, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *finite completion transformer row.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-admission row for constant and displayed regular-rational inputs.*
3. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *row recording right-unit reuse of already completed names.*
4. *B is the bind-association ledger data supplied by the* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *schedule route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window readback shared by the unit, bind, and idempotence rows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation ledger read by those windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
8. *H is the componentwise* hsame *transport row for the monad, unit, idempotence, bind, stream, dyadic, and regular-rational rows.*

9. *C is the displayed* Cont *replay row forcing every law-facing consumer through M, U, I, B, S, D, R, H in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and never through quotient completions, selected representatives, host equality, or an ambient limit object.

Lean4 (definition) `BEDC.Derived.Cauchy completion monad lawUp.CauchyCompletionMonadLawCarrier`

Theorem 2862.2 (CauchyCompletionMonadLaw NameCert obligations)**.** *For every accepted* CauchyCompletionMo *carrier $L = (M, U, I, B, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *surface consists exactly of these rows:*

1. *carrier admission for the monad transformer, unit admission, idempotence reuse, bind-association ledger, stream windows, dyadic observations, regular-rational readback, componentwise transport, replay, provenance, and local name;*
2. *left-unit exactness, requiring a unit-fed bind read to return through the same displayed S, D, R route before the* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
3. *right-unit and idempotence exactness, requiring a completed read followed by the unit/idempotence route to reuse the same M, I, S, D, R rows;*
4. *associativity exactness, requiring the two two-step bind bracketings to expose the same B, S, D, R, H, C, P, N ledger;*
5. *classifier transport for all law rows by componentwise* hsame *and displayed* Cont *replay;*
6. *non-escape of quotient completion classes, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No monad-law obligation is stored outside this finite carrier.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadLawUp.CauchyCompletionMonadLawCarrier namecert obligations`
↪variant `BEDC.Derived.Cauchy completion monad lawUp.CauchyCompletionMonadLawNamecertObligations`

Proof. Project the carrier of Definition 3834.1. Carrier admission and classifier transport are exactly the displayed rows $M, U, I, B, S, D, R, H, C, P, N$.

For the unit rows, read U before any bind or terminal seal. The left-unit route passes from U through the shared stream, dyadic, and regular-rational readback rows S, D, R, while the right-unit route reads I as the idempotence reuse of the same completion-facing data.

For associativity, use the bind ledger B together with H and C. Both bracketings must be replayed through the same displayed schedule, finite windows, dyadic observations, and regular-rational readback before the provenance and local naming rows are consumed.

The non-escape clause follows by exhaustion. The carrier has no coordinate for quotient completion classes, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient completeness, so none of those objects can be exported by the local certificate. □

Theorem 2862.3 (CauchyCompletionMonadLaw unit-associativity boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the unit rows and the associativity rows through the finite carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$*. In particular, left-unit, right-unit, idempotence, and associativity claims reach* $\mathsf{Real}^{\uparrow}$ *only after the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CompletionIdempotence}^{\uparrow}$*,* $\mathsf{CauchyCompletionBind}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have been displayed. No quotient completion, selected limit, host equality, hidden stream representative, or ambient completeness datum is available as a second route.*

Proof. Start from a $\mathsf{Real}^{\uparrow}$-facing read. By Definition 3834.2, the read must first pass through carrier admission and then through the law row relevant to the requested unit or associativity claim.

For left-unit and right-unit reads, the route is forced through U and I respectively, with S, D, R supplying the finite stream, dyadic, and regular-rational readbacks before any terminal real seal can be inspected.

For associativity reads, the route is forced through the bind-association ledger B. The replay row C and transport row H keep the two bracketings on the same displayed schedule, window, observation, and readback surface.

Thus every real-facing law consumer uses the finite packet rows $M, U, I, B, S, D, R, H, C, P, N$. Since the packet contains no second coordinate for quotient completion, selected limits, host equality, hidden representatives, or ambient completeness, those objects cannot serve as alternate witnesses for the monad laws. □

Theorem 2862.4 (CauchyCompletionMonadLaw completion-reflection route)**.** *Every unit, multiplication, idempotence, or associativity read exported by an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet first consumes the completion-reflection route through chapter 423, chapter 397, and chapter 1506. Only after those sibling rows have supplied functorial completion, universal-property reflection, and metric-completion admission may the finite law rows* $M, U, I, B, S, D, R, H, C, P, N$ *be read as monad-law data. The theorem is a finite dependency route; it does not identify quotient completions, select representatives, or assert equality of completed objects.*

Proof. Begin with the carrier of Definition 3834.1. Its law rows expose the monad transformer, unit, idempotence, bind ledger, stream windows, dyadic observations, regular-rational readback, transport, replay, provenance, and local name.

Apply the NameCert obligations of Definition 3834.2. A law-facing consumer can read left-unit, right-unit, idempotence, or associativity only through the displayed carrier and through the finite readback rows that precede the terminal $\mathsf{Real}^{\uparrow}$ seal.

The completion-reflection route supplies the required sibling boundary before those law rows are exported. The completion functor gives the functorial completion surface, the universal-property chapter gives the reflection witness, and the metric-completion chapter gives the completion admission surface consumed by both.

Apply Definition 3834.3. It rules out quotient completion, selected limits, host equality, hidden representatives, and ambient completeness as alternate witnesses, so the completion-reflection route remains a dependency theorem about finite rows only. □

Theorem 2862.5 (CauchyCompletionMonadLaw L10 forward consumer)**.** *Any* $\mathsf{BoundedCauchyInterval}^{\uparrow}$*,* $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$*, or* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *consumer of the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must read the functorial completion, unit, bind, idempotence,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before using unit or associativity law data. The downstream chapters*

chapter 6065, chapter 7362, and chapter 7527 therefore consume $L = (M, U, I, B, S, D, R, H, C, P, N)$ *as a finite L10 completion-law surface.*

This theorem is a consumer-facing dependency route. It does not import quotient completions, selected representatives, selected limits, host equality of completed objects, countable choice, hidden stream witnesses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the monad transformer, unit, idempotence, bind ledger, finite windows, dyadic observations, regular-rational readback, transport, replay, provenance, and local name.

Apply Definition 3834.2. It supplies the carrier admission, unit exactness, idempotence exactness, associativity exactness, classifier transport, and nonescape clauses for the law surface.

Apply Definition 3834.3 and Definition 3834.4. Together they force every law-facing consumer through the completion reflection siblings and through the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback rows before unit or associativity data are consumed.

The bounded interval, Kleisli extension, and completion-intersection chapters already read this chapter as a downstream completion-law dependency. Since the local carrier contains no quotient, selector, choice, hidden-stream, or ambient-completeness coordinate, those consumers receive only the finite L10 route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3834.1, Definition 3834.2, Definition 3834.3, and Definition 3834.4 seed the finite completion-monad law carrier over monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, local naming, and completion-reflection rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, quotient completion equality, selected representatives, selected limits, host equality of completed objects, countable choice, hidden stream witnesses, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, left-unit exactness, right-unit and idempotence exactness, associativity exactness, classifier transport, and non-escape for BEDC.Derived.CauchyCompletionMonadLawUp.

Constructive story: A $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet is generated from a CauchyCompletionMonad row, CauchyCompletionUnit row, CompletionIdempotence row, CauchyCompletionBind ledger, StreamName windows, DyadicRatCore observations, RegSeqRat readback, componentwise transport, displayed replay, Pkg provenance, and a local NameCert row.

Theorem 2862.6 (CauchyCompletionMonadLaw functor dependency route)**.** *Every accepted* CauchyCompletionMo *packet consumes the functorial completion surface of chapter 423 before its multiplication, unit, idempotence, or associativity rows can be read as law data. Concretely, the monad-law carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *first reads the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local NameCert rows, and only then replays* $M, U, I, B, S, D, R, H, C, P, N$ *as the law surface. Thus the law packet records coherence over the displayed functorial completion route rather than a kernel-only island or a host-level completion equality.*

Proof. Project the carrier of Definition 3834.1. Its monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming rows are law-facing rows, but they do not by themselves identify the completion transformer as a functorial public route.

Read chapter 423 together with the sibling universal-property certificate of chapter 397. The functor packet supplies the source-metric row, regular-Cauchy family row, completion seal, unit-and-bind schedule, universal-property row, classifier ledger, transport row, provenance row, and

local NameCert row consumed by completion-facing clients; the unit-and-bind law rows consume the universal-property witness before associativity is read as monad law data.

Now apply Definition 3834.2 and Definition 3834.3. They replay the unit, idempotence, bind, and associativity rows through the displayed stream, dyadic, regular-rational, transport, and provenance data. Since the functor route is the public completion surface being coordinated, the monad-law packet depends substantively on CauchyCompletionFunctor$^{\uparrow}$ before the law rows are read. □

The CauchyCompletionMonadLaw$^{\uparrow}$ packet also reads the completion universal-property and metric-completion siblings before its law rows are exposed to completion consumers. The extension-uniqueness side is supplied by chapter 397, while the metric-completion handoff is supplied by chapter 1506; hence the unit, multiplication, idempotence, and associativity rows remain attached to displayed completion data rather than to an isolated law ledger. The CauchyCompletionMonadLaw$^{\uparrow}$ packet reads the completion functor, universal-property, metric-completion, and Real-completion sibling rows before the monad multiplication and unit laws are available as law data. The completion functor route of chapter 423 supplies the displayed functorial transformer, the universal-property route of chapter 397 supplies the extension and uniqueness rows, the metric-completion packet of chapter 1506 supplies the completion carrier, and the Real-completion handoff of chapter 360 supplies the terminal completion-facing seal. Only after those rows are visible may the law packet expose multiplication, unit, idempotence, and associativity through its local carrier. The CauchyCompletionMonadLaw$^{\uparrow}$ packet reads the completion functor route of chapter 423, the completion universal-property surface of chapter 397, and the metric-completion dependency lattice of chapter 1506 before any Real$^{\uparrow}$ or Completion$^{\uparrow}$ consumer treats the monad boundary as available law data. Thus unit, multiplication, idempotence, and associativity rows are exposed only after the displayed completion-facing dependencies are visible, not as isolated quotient-completion equality, selected limits, or ambient real completeness. The CauchyCompletionMonadLaw$^{\uparrow}$ rows read the completion functor sibling of chapter 423 and the Cauchy-completion universal-property sibling of chapter 2319 before their unit, multiplication, and associativity coordinates are accepted as monad-law data. This dependency route is a displayed finite lattice over completion-facing NameCert rows; it does not add quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient Real$^{\uparrow}$ completeness. The CauchyCompletionMonadLaw$^{\uparrow}$ packet exports its multiplication-law rows toward chapter 1829 as a concrete completion-monad sibling route. The multiplication consumer reads the displayed unit, bind, idempotence, associativity, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming rows of the law packet before it flattens a completion of completions; the edge is finite NameCert data, not quotient-completion equality, selected representatives, countable choice, or ambient Real$^{\uparrow}$ completeness.

Theorem 2862.7 (CauchyCompletionMonadLaw Bishop completion forward route)**.** *Every accepted* CauchyCompletionMonadLaw$^{\uparrow}$ *read that supplies the completion monad multiplication and unit route must expose the* BishopRegularCauchyCompletion$^{\uparrow}$ *carrier of chapter 1639 before the monad law is consumed. The required forward route displays the* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* StreamName$^{\uparrow}$ *finite observation windows,* RegSeqRat$^{\uparrow}$ *regular rational readback, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows as the Bishop completion boundary for the monad multiplication and unit.*

Proof. Read the monad-law carrier through the existing completion route. Its multiplication and unit rows are completion-facing, so they cannot be consumed as a free algebraic law before the completion boundary is displayed.

Apply the Bishop regular Cauchy completion packet of chapter 1639. That packet exposes the dyadic tolerance ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows that a completion-facing consumer must inspect.

The monad law then receives its forward route through those visible rows. Because the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows remain on the path, the read is a finite completion route rather than quotient completion, selected limit, countable choice, host real equality, or detached ambient completeness. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads chapter 397 and chapter 423 before associativity law rows are public. This is a finite completion-law dependency through the universal-property and functor rows, not quotient completion equality, selected limits, or host equality.

Theorem 2862.8 (CauchyCompletionMonadLaw functor forward anchor)**.** *Every public monad-law read from an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet factors through the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *dependency route before unit, bind, idempotence, or associativity rows are exposed. The consumer must first read the functor dependency route of Definition 3834.6 and the local NameCert construction of Definition 3834.2; only then may it inspect* $M, U, I, B, S, D, R, H, C, P, N$ *as law rows. The route is a forward anchor into the public functorial completion surface, not a quotient-completion equality, selected representative, selected limit, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Read the functor dependency route first. The theorem Definition 3834.6 places the CauchyCompletionFunctor source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local naming rows before any monad-law consumer reads law data.

Then read the local construction through Definition 3834.2. It exposes $M, U, I, B, S, D, R, H, C, P, N$ as the finite monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming surface.

With both reads in place, the public law route may expose unit, bind, idempotence, and associativity as rows attached to the functorial completion dependency. Since the carrier has no quotient-completion, selected-representative, selected-limit, countable-choice, or ambient-completeness coordinate, the anchor is only the displayed forward dependency. □

Lemma 2862.9 (CauchyCompletionMonadLaw functorial law lattice route)**.** *Every accepted* CauchyCompletionMo *law read first consumes the functorial completion route of chapter 423 and the universal-property route of chapter 397. Only after those rows are displayed may the unit, bind, idempotence, and associativity rows of* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *be read as monad-law data. The packet therefore records a law lattice over the public functorial completion dependency, not a host-level completion equality or a quotient-completion shortcut.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ are the monad-law surface, while the functorial completion packet supplies the public source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local naming route.

Apply Definition 3834.6. It forces the completion functor and universal-property rows to be read before the law rows are consumed by a completion-facing client.

Finally apply Definition 3834.2 and Definition 3834.3. These replay unit, bind, idempotence, and associativity through the same finite stream, dyadic, regular-rational, transport, replay, provenance, and local naming rows, so the law lattice remains attached to the displayed functorial dependency. □

Lemma 2862.10 (CauchyCompletionMonadLaw completion dependency route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law read consumes the completion-route carrier of chapter 1556, the criterion surface of chapter 1476, and the finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *route before its unit and multiplication laws can be read as monad-law data. The packet therefore binds the unit, idempotence, bind, and associativity rows to the displayed constructive completion route rather than to quotient completion equality, selected limits, countable choice, trichotomy, host equality, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3834.1. Its law rows $M, U, I, B, S, D, R, H, C, P, N$ are completion-facing rows, so the source completion route must be visible before a client can read them as unit or multiplication data.

Read the constructive metric completion carrier and the Cauchy-completion criterion surface through the cited sibling chapters. They supply the completion admissibility and criterion rows consumed by the monad transformer before the law schedule is replayed.

Finally apply Definition 3834.2 and Definition 3834.3. These force the law read through the finite StreamName, RegSeqRat, and DyadicRatCore rows, leaving no coordinate for quotient completion equality, selected limits, countable choice, trichotomy, host equality, or ambient real completeness. □

Lemma 2862.11 (CauchyCompletionMonadLaw idempotent metric-monad route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *consumer that reads the unit, bind, multiplication, or associativity rows must first expose the completion idempotent route of* **??** *and the metric completion monad route of chapter 2156. The idempotent sibling supplies the completion-after-completion stabilization row, while the metric completion monad sibling supplies the unit and bind law surface; the combined route exports no quotient-completion equality, selected representative, countable choice, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the displayed law carrier from Definition 3834.1. Its unit, bind, multiplication, and associativity rows are read as completion-law data only after the carrier, stream, dyadic, regular rational, transport, replay, provenance, and local naming rows have been displayed.

Now read the completion idempotent sibling. The chapter **??** supplies the stabilization surface for applying completion after completion, so a monad-law consumer cannot treat the multiplication row as an unrecorded host equality.

Finally read the metric completion monad sibling. The chapter chapter 2156 supplies the unit and bind law surface consumed by the same route, and Definition 3834.2 keeps those rows inside the finite packet. Hence the sibling dependency lattice is substantive and finite, with none of the excluded quotient, selection, choice, host-equality, or ambient-completeness endpoints. □

Theorem 2862.12 (CauchyCompletionMonadLaw Kleisli consumer route)**.** *Every* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must first read the monad row, unit row, idempotence row, bind row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before exporting a Kleisli-facing* $\mathsf{Real}^{\uparrow}$ *seal. The route is a consumer of chapter 1845; it does not export quotient completion equality, selected representatives, countable-choice data, host equality of completed names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the local NameCert construction, Definition 3834.2. It exposes $M, U, I, B, S, D, R, H, C, P, N$ as the finite monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming surface.

Now use the unit-associativity boundary, Definition 3834.3. It forces every law-facing $\mathsf{Real}^{\uparrow}$ read through the displayed stream, dyadic, and regular-rational rows before any terminal seal can be inspected.

Finally attach the Kleisli consumer chapter chapter 1845. A $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ consumer may read the monad-law packet only after the same finite rows have been displayed, so its Kleisli-facing seal remains a finite BEDC consumer route rather than a quotient, selection, choice, host equality, or ambient-completeness claim. □

$\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ consumes the completion functor unit and multiplication route of chapter 423 together with the finite readback rows of the Cauchy carrier in chapter 3413. The dependency is only the displayed unit, multiplication, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming surface; it is not a quotient completion or a host universal property.

Theorem 2862.13 (CauchyCompletionMonadLaw completion-adjunction route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *unit or multiplication read consumes the same completion-adjunction route used by chapter 3836. The unit row, bind row, idempotence row, stream window, dyadic tolerance row, regular-rational readback, transport, replay, provenance, and local naming rows of* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *therefore sit over the displayed completion-adjunction surface rather than over an isolated kernel-only packet.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the law surface, but a completion-facing consumer still needs the adjunction route that identifies how completion reads are admitted and returned.

Read chapter 3836. Its source, dense embedding, completion seal, extension handoff, uniqueness, transport, replay, provenance, and local naming rows are the completion-adjunction surface through which the monad-law unit and multiplication rows are consumed.

Now apply Definition 3834.2 and Definition 3834.3. They force unit, idempotence, bind, and associativity reads through the same finite stream, dyadic, regular-rational, transport, replay, provenance, and naming rows, so the monad-law packet is routed through the completion-adjunction sibling before it can serve a Real-facing law read. □

Theorem 2862.14 (CauchyCompletionMonadLaw dense embedding dependency)**.** *Every unit, multiplication, idempotence, bind, and associativity read for an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must pass through the sibling dense embedding and completion route supplied by chapter 1651 and chapter 1512 before the law rows can be consumed as completion-facing data. The rows* $M, U, I, B, S, D, R, H, C, P, N$ *are therefore read as dense-image, completion-entry, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming data, not as a local hub-only law island.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the monad-law surface, but the completion-facing consumer still needs the dense image and completion embedding routes that make those law rows readable by a completion packet.

Apply the monad-law obligations and the unit-associativity boundary, Definition 3834.2 and Definition 3834.3. They force the unit, multiplication, idempotence, bind, and associativity rows to remain attached to the displayed stream, dyadic, regular-rational, transport, replay, provenance, and naming rows.

Finally route those rows through the dense uniform embedding and completion embedding siblings. The dense image witness supplies the source-to-completion coverage, and the completion

embedding handoff supplies the consumer-facing entry, so the law rows are consumed only after the sibling route has been read. □

Theorem 2862.15 (CauchyCompletionMonadLaw completion-reflection consumer link)**.** *Every* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *unit, multiplication, idempotence, or associativity read factors through the completion-adjunction surface of chapter 3836, the completion-functor route of chapter 423, and the completion universal-property sibling of chapter 397 before it is consumed as* $\mathsf{Real}^{\uparrow}$*-facing law data. The route is finite completion reflection through displayed carrier rows, not quotient completion equality, selected representatives, selected limits, host equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 3834.1. The monad-law rows $M, U, I, B, S, D, R, H, C, P, N$ expose unit, multiplication, idempotence, bind, stream, dyadic, regular-readback, transport, replay, provenance, and local naming data.

Read the sibling dependency route of Definition 3834.13. It already places the unit and multiplication rows over the metric completion-adjunction surface rather than over an isolated kernel packet.

Finally pass the same finite rows through the completion functor and universal-property siblings. These siblings supply the completion-reflection consumer route needed before a $\mathsf{Real}^{\uparrow}$-facing law read can occur, while the refused quotient, representative, limit, host-equality, and completeness objects remain absent from the carrier. □

Theorem 2862.16 (CauchyCompletionMonadLaw downstream completion consumer)**.** *Any downstream completion consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the law surface through chapter 423, chapter 397, and the totally bounded completion consumer of chapter 1573 before the rows* $M, U, I, B, S, D, R, H, C, P, N$ *are exported as completion-law data. The route is a displayed consumer edge through finite carrier rows; it is not quotient completion equality, selected representatives, selected limits, open-cover compactness, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming data, but none of those rows by itself supplies a downstream completion consumer.

Use the completion-functor route of Definition 3834.13. It places the unit and multiplication reads over the same functorial completion surface that a completion-facing client must inspect before law rows are available.

Next apply Definition 3834.14 and Definition 3834.15. These route the same finite rows through the displayed completion embedding, completion functor, and completion universal-property siblings before any $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer reads them.

Finally read chapter 1573. The totally bounded completion packet is an existing downstream consumer of completion rows, so it may consume the monad-law surface only after the functorial and universal-property routes have already kept $M, U, I, B, S, D, R, H, C, P, N$ attached to finite carrier data. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet also sits horizontally over the completion-functor sibling of chapter 423. Its unit, multiplication, idempotence, and associativity rows consume the functorial completion route before they are read as monad-law data, so the completion monad/functor law lattice is visible as a sibling dependency rather than a kernel-only float. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ route consumes the Real and Completion siblings before any unit or associativity row is exported: $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ provide the L10 real-name readback

through chapter 38, while CauchyCompletionFunctor$^{\uparrow}$ and CompletionUniversalProperty$^{\uparrow}$ provide the completion-facing handoff through chapter 397. Thus a monad-law consumer sees the finite real-name route and the completion universal-property route as displayed dependencies, not as hidden quotient completion data.

Theorem 2862.17 (CauchyCompletionMonadLaw real completion unit route)**.** *Every accepted* CauchyCompletionMonadLaw$^{\uparrow}$ *unit, multiplication, or associativity read that is consumed as completion-facing law data first passes through the* RealCompletionUnit$^{\uparrow}$ *route of chapter 7084. The read exposes the* StreamName$^{\uparrow}$ *window row,* DyadicRatCore$^{\uparrow}$ *tolerance row,* RegSeqRat$^{\uparrow}$ *readback row, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before the law packet* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *is exported.*

Thus the monad-law consumer receives the real-completion unit as a displayed finite route. It does not receive quotient-completion equality, a selected Cauchy representative, countable-choice schedule, host equality of completions, hidden real completeness, or a detached Real$^{\uparrow}$ *endpoint.*

Proof. Project the carrier of Definition 3834.1. The law rows M, U, I, B can be consumed only together with the finite stream, dyadic, regular-rational, transport, replay, provenance, and naming rows S, D, R, H, C, P, N.

Apply the real completion unit surface of chapter 7084. Its carrier and nonescape theorems place every real-completion unit read behind the dyadic source, stream-window, regular readback, terminal real seal, transport, replay, provenance, and local naming rows.

Finally use Definition 3834.2 and Definition 3834.3. They bind unit, multiplication, idempotence, and associativity reads to the same finite carrier before any completion-facing consumer sees law data, so the exported route cannot import quotient completions, selected representatives, countable choice, host equality, hidden completeness, or a detached real endpoint. □

The CauchyCompletionMonadLaw$^{\uparrow}$ packet consumes its sibling completion lattice through displayed finite rows. The unit, multiplication, and composition reads are routed through chapter 3413, chapter 1506, chapter 397, and chapter 301: Cauchy rows name the input streams, MetricCompletion rows expose the completion carrier, the universal-property rows supply the comparison route, and RegSeqRat rows give the regular rational windows used by the readbacks. No step appeals to a host completion monad outside the displayed Cauchy completion and regular-rational certificate rows.

Theorem 2862.18 (Cauchy completion monad law completion dependency lattice)**.** *Every accepted* CauchyCompletionMonadLaw$^{\uparrow}$ *packet consumes its completion-facing law rows through the displayed sibling lattice:* MetricCompletion$^{\uparrow}$ *supplies the completion carrier of chapter 1506,* Completion$^{\uparrow}$ *and the* Real$^{\uparrow}$ *completion seal rows supply the real-name boundary of chapter 38, and* CauchyCompletionFunctor$^{\uparrow}$ *supplies the functorial route before unit, bind, multiplication, or associativity data are exported. The route is a finite dependency projection; it is not a quotient-completion equality, selected representative, selected limit, host equality of completed objects, countable-choice principle, or ambient* Real$^{\uparrow}$ *completeness claim.*

Proof. Project the monad-law carrier and the existing sibling routes. The law rows are already carried through the Cauchy completion monad, unit, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming coordinates before any Real-facing read is available.

Read the metric-completion boundary next. The MetricCompletion$^{\uparrow}$ sibling fixes the completion carrier as a displayed route through chapter 1506, so the monad-law packet cannot obtain a hidden host completion object.

Finally compose with the Real-facing route and the functor dependency route. The $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is reached only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ route is consumed before unit, bind, multiplication, or associativity rows are exported. Thus the dependency lattice is horizontal and finite, with none of the refused quotient, selector, choice, equality, or completeness endpoints. □

Lemma 2862.19 (CauchyCompletionMonadLaw regular Cauchy uniform-modulus route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *consumer that compares completion unit and bind windows must first read the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *budget of chapter 4158. The modulus row fixes the common regular-Cauchy tolerance budget before the monad-law consumer may identify two completion routes, and the idempotent metric-monad route of Definition 3834.11 keeps the resulting comparison inside the displayed unit, bind, multiplication, and associativity packet. The route exports no quotient-completion equality, selected representative, countable-choice schedule, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The regular-modulus sibling is not a decorative cross-reference: it supplies the uniform tolerance row that lets the local carrier of Definition 3834.1 compare unit and bind reads over one shared regular-Cauchy budget. Without that row, the monad-law packet would still display law coordinates, but it would not expose the finite modulus through which completion consumers synchronize their Cauchy windows.

Theorem 2862.20 (CauchyCompletionMonadLaw regular-modulus unit route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *that exports a unit read to a completion-facing consumer first consumes the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *sibling of chapter 4158. The sibling supplies the common regular-Cauchy modulus for the stream row* S*, dyadic tolerance row* D*, and regular-rational row* R *before the unit row* U *is replayed through* Cont *and recorded by the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *row. Hence the unit route is a finite modulus-controlled read of the local carrier, not a quotient-completion equality, selected representative, countable-choice schedule, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier through Definition 3834.1. Its rows S, D, R are the finite stream, dyadic tolerance, and regular-rational readback data used by unit and bind comparisons.

Apply Definition 3834.19. It places the $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ budget before the monad-law consumer identifies unit and bind windows.

Finally apply Definition 3834.2. The unit row U, replay row C, provenance row P, and local naming row N are consumed only after the displayed carrier rows are available, so the route remains inside the finite monad-law packet and cannot import any excluded completion principle. □

Theorem 2862.21 (CauchyCompletionMonadLaw forward completion-category route)**.** *Every* $\mathsf{CauchyCompletionCategory}^{\uparrow}$ *or metric-completion Kleisli consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the monad-law surface through the displayed rows of chapter 378, the functorial completion rows of chapter 423, and the metric-completion admission rows of chapter 1506 before any categorical composition law is used. The visible route is*

$$\mathsf{CauchyCompletionMonad}^{\uparrow},\ \mathsf{CauchyCompletionUnit}^{\uparrow},\ \mathsf{CompletionIdempotence}^{\uparrow},\ \mathsf{CauchyCompletionBind}^{\uparrow},\ \mathsf{StreamNam}$$

It exports no quotient completion equality, selected representative, selected limit, host equality of completed objects, countable-choice schedule, hidden stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. A law-facing consumer receives only the rows $M, U, I, B, S, D, R, H, C, P, N$, so the monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming data are the complete visible surface.

For a completion-category or Kleisli consumer, first read the monad and functorial completion siblings through chapter 378 and chapter 423. The unit row U, idempotence row I, and bind row B supply the law data only as displayed carrier rows.

Next pass through the metric-completion admission of chapter 1506. The stream, dyadic, regular-rational, and Real rows S, D, R keep the consumer on the finite readback route before any categorical composition law is visible.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for quotient completion equality, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient completeness, the stated route is exhaustive. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads the multiplication sibling of chapter 1829 together with the monad sibling of chapter 378 before its associativity and unit-law rows are exposed. The multiplication row supplies the displayed bind-reassociation surface, while the monad row supplies the unit-and-bind schedule consumed by the law carrier; the route therefore remains a finite NameCert dependency through those sibling packets rather than a host-level equality of completed objects.

Theorem 2862.22 (CauchyCompletionMonadLaw finite-approximation consumer route)**.** *Every finite-approximation or totally bounded completion consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the monad-law surface through the displayed completion rows before using it as approximation data. The route passes through the finite window, dyadic tolerance, regular-rational readback, unit, multiplication, idempotence, bind, transport, replay, provenance, and local naming rows of Definition 3834.1, then through the finite approximation surface of chapter 7574, and finally through the totally bounded completion consumer of chapter 1573. It exports a finite completion-law route for approximation consumers, not quotient completion equality, selected representatives, selected limits, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadLawUp.CauchyCompletionMonadLawFiniteApproximationConsumerRout`

Proof. Project the carrier of Definition 3834.1. The law rows $M, U, I, B, S, D, R, H, C, P, N$ expose unit, multiplication, idempotence, bind, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming data.

Use the local NameCert obligations and the unit-associativity boundary, Definition 3834.2 and Definition 3834.3. They keep every law read attached to the same finite completion-facing rows before a consumer can treat the law as available approximation data.

Now route the read through the approximation and total-boundedness consumers. The finite approximation surface of chapter 7574 supplies the windowed approximation interface, while chapter 1573 supplies the totally bounded completion consumer edge.

Since the carrier and both consumer routes display only finite rows, the resulting read cannot introduce quotient completion equality, selected representatives, selected limits, open-cover compactness, countable choice, or ambient Real completeness. □

Theorem 2862.23 (CauchyCompletionMonadLaw forward consumer lattice)**.** *Finite-approximation and totally bounded completion consumers read the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *monad-law rows*

only through the displayed sibling routes

CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Reg

Thus a downstream approximation read may consume unit, multiplication, bind, idempotence, transport, replay, provenance, and local naming rows from the monad-law packet only after the functor, universal-property, metric-completion, finite stream-window, dyadic tolerance, regular-rational readback, and terminal Real rows have remained visible. The lattice exports no selected completion representative, quotient completion equality, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

The CauchyCompletionMonadLaw$^{\uparrow}$ route supplies a downstream completion consumer edge for finite-net packets: chapter 1573 reads monadic completion composition only after the Cauchy-filter and embedding rows have remained attached to the finite-net ledger, while chapter 1844 keeps the dyadic, stream, regular-rational, and Real-seal faces visible before any completion-facing read is consumed. Thus the monad-law chapter is a sibling lattice route for finite-net completion consumers, not a source of selected limits, quotient representatives, open-cover compactness, or ambient Real completeness.

Chapter 2863

A Concrete Naming Certificate for CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and filter convergence, and Bishop–Bridges on constructive completion, the CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$ packet records the finite filterbase-completion refinement of the complete uniform-space surface. It sits after the uniform entourage data of chapter 339, the saturated Cauchy-filter route of chapter 333, the finite basis generator of chapter 1501, the limit-facing boundary of chapter 1558, the window and readback surfaces of chapter 126 and chapter 301, the terminal seal of chapter 38, and the completion handoffs of chapter 1473 and chapter 1569. The packet names a displayed filterbase route to completion consumers; it is not a quotient filter, selected completion point, countable-choice selector, host equality, ambient completeness theorem, or public standard bridge.

Definition 2863.1 (Complete uniform-space filterbase-completion carrier)**.** *A* CompleteUniformSpaceFilterbaseCom *carrier is a finite* BHist *packet*

$$Q = (U, F, B, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformSpace$^{\uparrow}$ *entourage row, displaying the finite entourage filterbase, diagonal, refinement, symmetry, and composition data consumed by the completion request.*
2. *F is the* CauchyFilter$^{\uparrow}$ *row, recording the saturated Cauchy-filter request over the uniform surface of U.*
3. *B is the* CauchyFilterBasis$^{\uparrow}$ *generator row, exposing the finite basis elements and common-refinement data through which F is read.*
4. *L is the* CauchyFilterLimit$^{\uparrow}$ *boundary row, reached only after the basis and saturated filter rows have been replayed.*
5. *S is the* StreamName$^{\uparrow}$ *window row attached to the filterbase convergence request.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from the same windows.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row read only after U, F, B, L, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the entourage, filter, basis, limit boundary, window, readback, seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from the uniform entourage request through the filterbase generator, Cauchy saturation, limit boundary, finite windows, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, B, L, S, R, E, H, C, P, N$. The carrier has no coordinate for quotient filters, selected completion points, countable-choice schedules, host equality, ambient completeness, or a detached completion object.

Theorem 2863.2 (Filterbase convergence obligations)**.** *For every accepted* CompleteUniformSpaceFilterbaseComple *carrier $Q = (U, F, B, L, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed uniform-space entourage data, Cauchy-filter row, CauchyFilterBasis generator, CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, Real seal, transport, replay, provenance, and local naming row;*

2. *filterbase convergence exposure, requiring every completion-facing request to pass through U, B, and F before the limit boundary L is inspected;*

3. *readback exactness, requiring every accepted boundary observation to pass from L through the matching S and R rows before the Real seal E is available;*

4. *completion-handoff stability, requiring* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumers named by P to read the same displayed route U, B, F, L, S, R, E;*

5. *ledger nonescape, requiring H, C, P, N to export no quotient filter, selected completion point, countable-choice selector, host equality, ambient completeness theorem, or detached completion object.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3835.1. Its public rows are exactly $U, F, B, L, S, R, E, H, C, P, N$, so carrier admission exposes only the displayed uniform, filterbase, convergence, seal, and structural data.

The filterbase convergence exposure is forced by the row order. A completion-facing request first reads the finite entourage data U, then the basis generator B, and only then the saturated Cauchy-filter row F before the limit boundary L is available.

Readback exactness is downstream of the boundary row. The row L can expose observations only through the StreamName windows S and the RegSeqRat readback R, and the terminal Real seal E is available only after those rows have been replayed.

The completion-handoff and ledger clauses are structural. Uniform-completion and Cauchy-filter-completion consumers are named in P as readers of the same displayed route, while H, C, N transport, replay, and name only that route. Hence the excluded quotient, selector, host-equality, choice, completeness, and detached-completion objects have no coordinate in the packet. □

Lean4 ✓ `BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.CompleteUniformSpaceFilterbaseCompletio`
`single carrier alignment`

Theorem 2863.3 (Cauchy-filter completion handoff)**.** *Let $Q = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ *packet. Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer that reads Q receives the finite handoff*

$$U \longmapsto B \longmapsto F \longmapsto L \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion handoff is filterbase-first: the consumer cannot inspect a completion-facing boundary before the uniform entourage row, the CauchyFilterBasis generator, and the saturated CauchyFilter row have been displayed.

Proof. Begin with a completion-facing consumer. The provenance row P may mention $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{CauchyFilterCompletion}^{\uparrow}$, but it does so only as a row of the accepted carrier Q.

The source of the handoff is therefore the displayed route. The uniform-space row U supplies the entourage filterbase, B supplies the finite basis generator, and F supplies the saturated Cauchy-filter request over the same data.

The boundary and readback rows are downstream. The consumer reaches L only after U, B, F have been replayed, then reads S and R before the terminal seal E is available.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier has no separate completion object, selected point, quotient filter, choice schedule, host equality, or ambient completeness row, every accepted completion handoff is confined to the displayed path. □

Theorem 2863.4 (Filterbase Real-seal nonescape)**.** *Let $Q = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ *carrier. Every consumer that reaches the displayed* $\mathsf{Real}^{\uparrow}$ *seal row E must first pass through the uniform entourage row U, the CauchyFilterBasis row B, the saturated CauchyFilter row F, the CauchyFilterLimit boundary row L, the StreamName windows S, and the RegSeqRat readback row R. The rows H, C, P, N then preserve and name the same route, so no accepted read can escape from E into a selected completion point, quotient filter, host equality, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or detached* $\mathsf{UniformCompletion}^{\uparrow}$ *object.*

Proof. Project the carrier definition. The terminal row E appears only as the Real seal of the tuple, and the preceding public coordinates are precisely U, F, B, L, S, R.

Apply the obligation surface of Definition 3835.2. It requires the filterbase convergence request to consume U, B, F, then the limit boundary L, then the window and readback rows S, R before E can be read.

The structural rows cannot introduce a second route. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed packet, while the excluded selector, quotient, host-equality, choice, ambient-completeness, and detached-completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ packet is generated from UniformSpace entourage data, a CauchyFilter row, a CauchyFilterBasis generator, a CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, a RealUp seal, UniformCompletionUp and CauchyFilterCompletionUp provenance, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3835.1, Definition 3835.2, Definition 3835.3, and Definition 3835.4 seed the finite filterbase-completion route for complete uniform-space consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.CompleteUniformSpaceFilterbaseCompletionTast`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, countable-choice selectors, host equality, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, filterbase convergence exposure, readback exactness, completion-handoff stability, and Real-seal nonescape for BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.

Chapter 2864

A Concrete Naming Certificate for $\mathsf{MetricCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ packet records the universal mapping boundary for a displayed metric source and its completion as finite BEDC data. It reads the metric-completion carrier of chapter 1506, the dense embedding route of chapter 3351, the Cauchy adjunction surface of chapter 1456, and the unit-triangle ledger of chapter 2256. The chapter names the metric-specific adjunction packet without quotient completion, selected representatives, countable choice, host equality of completed objects, or ambient completeness.

Definition 2864.1 (MetricCompletionAdjunction carrier)**.** *A* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, M, D, U, K, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, carrying the displayed carrier, distance classifier, finite balls, and the* $\mathsf{Met}^{\uparrow}$ *provenance needed before completion is read.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of chapter 1506, including the chosen Cauchy-filter or Cauchy-net branch, readback row, separated comparison, and uniform-completion handoff.*
3. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row of chapter 3351, displaying the source-to-completion dense image used by the universal mapping read.*
4. *U is the unit row sending each displayed source point into the completion packet through D and M.*
5. *K is the counit row for completion-facing consumers whose target is already read through the same metric completion surface.*
6. *T is the triangle ledger. It records that the unit followed by the counit, and the counit followed by the unit-facing extension, replay the same finite rows rather than invoking host equality of completed carriers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal row used by the readback side of the completion packet.*
8. *E is the* $\mathsf{UniformCompletion}^{\uparrow}$ *and extension-consumer row through which uniformly continuous finite-window maps read the same adjunction packet.*

9. *H records componentwise* hsame *transport for the source metric, completion row, dense embedding, unit, counit, triangle ledger, readback seal, extension row, replay, provenance, and local name.*

10. *C records displayed* Cont *replay from the source metric through dense embedding, unit, completion readback, extension, counit, and triangle comparison.*

11. *P is the* Pkg *provenance over* MetricCompletionAdjunction$^{\uparrow}$, MetricCompletion$^{\uparrow}$, MetricCompletionDenseEmbeddin CauchyCompletionAdjunction$^{\uparrow}$, CompletionUnitTriangle$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, UniformCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$ *row.*

The classifier accepts only packets whose unit, counit, and triangle rows are read through the same displayed source metric, completion packet, dense embedding, regular readback, Real seal, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows.*

Theorem 2864.2 (MetricCompletionAdjunction NameCert obligations)**.** *For every accepted* MetricCompletionAdju *carrier*

$$A = (X, M, D, U, K, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of source metric admission, metric-completion admission, dense embedding exposure, unit visibility, counit visibility, triangle-ledger comparison, RegSeqRat/Real seal readback, uniform-extension handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of A. No obligation may be discharged by a quotient completion, selected Cauchy limit, countable-choice schedule, host equality of completion classes, or ambient complete metric space.*

Proof. Project the carrier of Definition 3836.1. The analytic rows are precisely the source metric X, completion packet M, dense embedding D, unit U, counit K, triangle ledger T, readback seal R, and extension row E.

The dense embedding and completion rows force the order of every accepted read. A consumer first opens the source metric and metric-completion packet, then reads the dense image route before any unit, counit, triangle, Real-seal, or uniform-extension coordinate is available.

Transport and replay are structural. The row H compares only the displayed components by hsame, while C replays the same route through source, dense image, completion, extension, counit, and triangle rows.

Finally P and N source and name the same finite packet. Since none of the listed rows stores a quotient completion, selected limit, countable-choice schedule, host equality, or ambient complete metric space, the displayed obligations exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetricCompletionAdjunctionUp.MetricCompletionAdjunctionTasteGate single carrier alignment`

Theorem 2864.3 (MetricCompletionAdjunction unit-counit triangle)**.** *Every unit-counit triangle read of an accepted* MetricCompletionAdjunction$^{\uparrow}$ *carrier factors through the finite route*

$$X \longmapsto D \longmapsto U \longmapsto M \longmapsto E \longmapsto K \longmapsto T \longmapsto R, H, C, P, N.$$

The route identifies the two triangle faces only by the displayed hsame *and* Cont *rows carried in H and C, and it consumes the RegSeqRat/Real seal row R only after the metric completion, dense embedding, unit, counit, and uniform-extension rows have been displayed.*

Proof. Start from the carrier inventory. The source metric row X and dense embedding row D are the only rows that introduce source points into the completion packet, so the unit face must pass through them before it can reach M.

The extension and counit faces are then downstream of the same completion packet. The row E records the uniform-extension consumer route, K records the counit-facing return, and T records the triangle comparison over those already displayed rows.

The terminal seal and structural rows cannot change the route. The row R is the RegSeqRat/Real readback attached to the completion packet, while H, C, P, N only transport, replay, source, and name the same finite data. Therefore the triangle read is exhausted by the displayed route and imports no quotient, selector, choice, host equality, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ packet is generated from a source metric row, metric-completion row, dense embedding row, unit row, counit row, triangle ledger, RegSeqRat/Real seal readback, uniform-extension handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3836.1, Definition 3836.2, and Definition 3836.3 seed the metric-specific completion adjunction boundary over dense embedding, unit, counit, triangle, readback, and uniform-extension rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricCompletionAdjunctionUp.MetricCompletionAdjunctionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of completed objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense embedding exposure, unit and counit visibility, triangle comparison, RegSeqRat/Real seal handoff, uniform-extension handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionAdjunctionUp.

The $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ hub routes first through chapter 486, chapter 782, and chapter 444 before the NameCertMorphism dependency child is read.

Chapter 2865

A Concrete Naming Certificate for $\mathsf{GeneratorClosureRobustness}^{\uparrow}$

The $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ packet records a finite hypercube and median-closure certificate for the zero-marker generator-closure robustness surface of Definition 9462.1779. It reads active median generators, median-essential codons, deletion-loss rows, Q_2 completion closure, and minimal percolating seeds as finite combinatorial data over Q_6. The finite table read is routed through the finite observation budget selector of chapter 782 before the active-generator, deletion-loss, and Q_2 completion rows are consumed. The collision-window consumer of chapter 444 reads the same finite moment-index and Hankel-shadow style rows, so the robustness packet's trigger and Q_2-completion ledgers feed a sibling finite-kernel consumer. The chapter is a NameCert target for finite closure robustness only; it does not claim biological necessity, probabilistic selection, host-level optimality, or an unbounded search principle.

The active-generator and Q_2-completion rows refine the generator-closure carrier of chapter 486. The robustness packet adds deletion-loss, minimal-seed, and trigger-comparison ledgers before any robustness read is exported, so every robustness consumer first sees the same finite generator-closure boundary and then the additional robustness tables.

The same dependency also controls replay: the robustness packet consumes the generator-closure carrier and replay rows of chapter 486 before it can state robustness under perturbation or audit transport.

Thus the active-generator and Q_2 completion rows consume three sibling surfaces before any trigger read is available: chapter 486 supplies the finite generator boundary, chapter 782 supplies the finite table-inspection budget, and chapter 444 supplies the downstream finite-kernel consumer that reads the same collision-facing row family.

Definition 2865.1 (GeneratorClosureRobustness carrier)**.** *A* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, E, L, Q, P, T, H, C, K, N)$$

with the following displayed rows:

1. *A is the active median-generator row, listing the two size-7 active generators and their common rigid skeleton.*
2. *E is the median-essential row, listing the five codons whose deletion changes the median closure.*
3. *L is the deletion-loss row, recording the finite lost-codon table for each median-essential deletion.*

4. *Q is the Q_2 completion row, recording the rule that a three-of-four occupied Q_2 face adjoins the fourth vertex.*
5. *P is the percolating-seed row, recording the 484 size-7 Q_2-completion seeds and the absence of smaller percolating seeds.*
6. *T is the trigger-closure row comparing rigid median generation with broad Q_2 percolation capacity.*
7. *H is the componentwise* hsame *transport row for active-generator, deletion-loss, completion, seed, and trigger rows.*
8. *C is the displayed* Cont *replay row forcing every robustness read through A, E, L, Q, P, T, H.*
9. *K is the* Pkg *provenance over* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$*, finite codon-window rows, Q_6, Q_2 faces,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by this finite table data and its displayed transports.

Theorem 2865.2 (GeneratorClosureRobustness NameCert obligations)**.** *For every accepted* GeneratorClosureRobu *carrier $G = (A, E, L, Q, P, T, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$ *surface consists exactly of these rows:*

1. *carrier admission for the active median generators, median-essential list, deletion-loss table, Q_2 completion rule, percolating-seed table, trigger comparison, transport, replay, provenance, and local name;*
2. *active-generator exactness, requiring the two displayed size-7 generators and their common six-codon skeleton to be read before a rigidity claim is accepted;*
3. *deletion-loss exactness, requiring every median-essential deletion to expose its finite lost-codon row;*
4. *Q_2 completion exactness, requiring every completion step to be witnessed by a displayed axis-aligned Q_2 face with at least three current vertices;*
5. *seed minimality, requiring the no-size-6 row and the 484 size-7 seed row before a percolation claim is accepted;*
6. *non-escape of biological necessity, probabilistic selection, unbounded search, host optimality, or hidden codon classifiers.*

No closure-robustness obligation is stored outside this finite packet.

Proof. Project the carrier of Definition 3837.1. The only rows available to the certificate are $A, E, L, Q, P, T, H, C, K, N$.

The active-generator and deletion-loss obligations read A, E, L as finite tables. The median-generator claim is accepted only after the two size-7 generators and their common skeleton are displayed, and each essential deletion is accepted only with its listed lost-codon row.

The Q_2 completion and seed obligations read Q and P. Each completion step is justified by a finite face row, and the percolation claim is accepted only after the no-smaller-seed row and the 484 minimal-seed row are present.

The trigger and non-escape clauses follow through T, H, C, K, N. These rows package the finite comparison between rigid median generation and broad Q_2 percolation; they contain no

biological-necessity coordinate, probability model, unbounded search oracle, host optimum, or hidden classifier. □

Theorem 2865.3 (GeneratorClosureRobustness Q2 completion exactness)**.** *Every accepted Q_2 completion read in a* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *packet must expose the displayed axis-aligned Q_2 face row Q, the active-generator row A, the deletion-loss row L, and the percolating-seed row P before a trigger-closure conclusion is read.*

Proof. Project the carrier of Definition 3837.1. The completion coordinate is the row Q, while the rows A, L, and P record the finite active-generator, deletion-loss, and percolating-seed data that determine whether the completion is part of the accepted robustness packet.

Apply Definition 3837.2. Its Q_2 completion exactness clause requires every completion step to be witnessed by a displayed axis-aligned Q_2 face with at least three current vertices, and its active-generator, deletion-loss, and seed clauses require the rows A, L, P before any trigger-facing read is accepted.

The rows T, H, C, K, N can only compare, transport, replay, source, and name those finite rows. Therefore the completion read exports no biological necessity, probability model, host optimum, hidden classifier, or unbounded search principle. □

Theorem 2865.4 (GeneratorClosureRobustness trigger-closure boundary)**.** *Every trigger-closure read of an accepted* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *packet factors through the displayed active-generator row, median-essential row, deletion-loss table, Q_2 completion rule, percolating-seed table, trigger comparison, componentwise transport, replay, provenance, and local NameCert row. A missing active-generator row, deletion-loss row, Q_2 face row, or minimal-seed row is a failure to land on the finite robustness surface, not a robustness counterexample.*

Proof. Begin with an accepted trigger-closure read. Carrier admission exposes the rows $A, E, L, Q, P, T, H, C, K, N$ before the trigger comparison can be consumed.

If the read concerns median rigidity, it must inspect A, E, L: the active generators, essential codons, and deletion losses are the displayed finite evidence for the rigid skeleton.

If the read concerns completion percolation, it must inspect Q and P: the Q_2 completion rule supplies the step witnesses, and the seed table supplies both the minimality lower bound and the 484 size-7 witnesses.

The replay row C and transport row H keep these two closure surfaces in one packet. Therefore a route missing one of the listed finite rows has not reached the accepted trigger-closure boundary, while a route that reaches the boundary exports no hidden classifier, biological-necessity assertion, probabilistic model, host optimum, or unbounded search principle. □

Falsifiable prediction: An accepted $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ read is refuted if a trigger-closure consumer reaches a rigidity or percolation conclusion without first displaying the active-generator row, deletion-loss table, Q_2 completion row, minimal-seed row, transport row, replay row, provenance row, and local naming row.

Independence witness: active median generation, median-essential deletion, Q_2 completion, and minimal percolating seeds: none is bijective with $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ because the packet jointly carries the rigid median skeleton, deletion losses, percolation rule, seed count, trigger comparison, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3837.1, Definition 3837.2, and Definition 3837.4 seed the finite generator-closure robustness carrier over active-generator, essential-codon, deletion-loss, Q_2 completion, seed, trigger, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{GeneratorClosureRobustness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, biological necessity, probabilistic selection, unbounded search, host optimality, or hidden codon classifiers.

Upgrade path: Obligation closure requires checked carrier admission, active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, trigger-closure boundary, and non-escape for BEDC.Derived.GeneratorClosureRobustnessUp.

Constructive story: A GeneratorClosureRobustness$^{\uparrow}$ packet is generated from active median generators, median-essential deletion rows, deletion-loss tables, Q2 completion faces, minimal percolating seed rows, trigger comparison, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theorem 2865.5 (GeneratorClosureRobustness forward-kernel consumer)**.** *Let $G = (A, E, L, Q, P, T, H, C, K, N)$ be an accepted* GeneratorClosureRobustness$^{\uparrow}$ *carrier. Any forward socket or kernel-audit consumer citing the robustness packet must read the active-generator row A, deletion-loss rows L, Q_2 completion row Q, minimal-seed row P, trigger row T, componentwise transport row H, replay row C, provenance row K, and local naming row N before invoking the* ForwardSocketLedger$^{\uparrow}$ *route of chapter 3387. The forward-kernel route is a displayed finite dependency route; it does not export biological necessity, probabilistic selection, host optimality, hidden codon classifiers, or an unbounded search principle.*

Proof. Project the carrier of Definition 3837.1. The rows $A, E, L, Q, P, T, H, C, K, N$ are the only coordinates a robustness consumer can inspect.

Use the NameCert obligations of Definition 3837.2. They require active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, and the local non-escape clauses before any public robustness read is accepted.

Apply Definition 3837.3 and Definition 3837.4. These force the completion and trigger reads to pass through $A, L, Q, P, T, H, C, K, N$ as one transported packet.

Only after those rows are visible may the consumer cite chapter 3387. The ForwardSocketLedger$^{\uparrow}$ chapter records the downstream socket ledger, while the robustness rows remain the finite kernel evidence that the consumer is auditing. □

Theorem 2865.6 (GeneratorClosureRobustness NameCertMorphism dependency route)**.** *Let $G = (A, E, L, Q, P, T, H, C, K, N)$ be an accepted* GeneratorClosureRobustness$^{\uparrow}$ *carrier. Every morphism-facing consumer of this packet must preserve the active-generator row A, the deletion-loss rows L, the Q_2 completion row Q, the minimal-seed row P, the trigger row T, the componentwise transport row H, the replay row C, the provenance row K, and the local naming row N before the read is transported through chapter 481. The route is a horizontal NameCert dependency route; it exports no biological necessity, probabilistic selection principle, host optimum, hidden codon classifier, unbounded search oracle, or morphism detached from the displayed robustness packet.*

Proof. Project the carrier of Definition 3837.1. The finite rows $A, E, L, Q, P, T, H, C, K, N$ are the only data available to a morphism-facing robustness consumer.

Apply the NameCert obligations of Definition 3837.2. They force carrier admission, active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, trigger comparison, transport, replay, provenance, and local naming to remain visible before any robustness read is accepted.

Use Definition 3837.3 and Definition 3837.4. These keep the completion and trigger rows inside the same transported packet, so a consumer cannot replace the robustness evidence by a hidden classifier or an unbounded search principle.

Only after those rows are preserved may the consumer cite chapter 481. The NameCertMorphism route transports the displayed naming dependency, while $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ still supplies the finite active-generator, deletion-loss, completion, seed, trigger, transport, replay, provenance, and local naming rows being transported. □

Chapter 2866

A Concrete Naming Certificate for $\mathsf{ComputableUniformContinuity}^{\uparrow}$

Following Bishop–Bridges and Weihrauch computable analysis, the $\mathsf{ComputableUniformContinuity}^{\uparrow}$ packet records the finite BEDC surface by which a represented-space map exposes one computable modulus route before any real-valued completion consumer reads it as an ambient host function. It sits over the CauchyRepresentation decoding of chapter 1680, the source and target represented-space rows of chapter 1494, the finite name-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, the modulus row of chapter 369, and the continuous-map route of chapter 130. The packet names a computable-analysis uniform-continuity boundary; it is not a host function, not a choice-selected modulus, not a quotient representation, not full $\mathsf{Real}^{\uparrow}$ completeness, and not a checked public bridge.

Definition 2866.1 (Computable uniform continuity carrier)**.** *A* $\mathsf{ComputableUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, K, R, M, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the represented-space source row, exposing only the source carrier, admissible names, and finite observation cells used by the packet.*
2. *Y is the represented-space target row, exposing the matching target carrier, target names, and target observation cells.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoding row, relating source and target names to regular Cauchy information before any real-facing seal is inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the decoded Cauchy data.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *realizer row, giving the finite rule that turns a requested target tolerance into a source-name window.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger for the source and target name segments inspected by M and K.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic budgets used by the source window, target window, and regular rational readback.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, Y, K, R, M, W, T have been displayed.*
9. *H records componentwise* hsame *transport for represented-space rows, decoding, readback, modulus, windows, tolerance ledger, seal, replay, provenance, and naming.*
10. *C records displayed* Cont *replay from a represented-space consumer through the decoding, modulus, finite-window, tolerance, readback, and seal rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ComputableUniformContinuity}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K, R, M, W, T, E, H, C, P, N$. The carrier has no coordinate for an external host function, a selected modulus outside M, quotient equality of representations, full real completeness, countable choice over names, or a detached continuity proof.

Lean4 ✓ `BEDC.Derived.ComputableUniformContinuityUp.ComputableUniformContinuityTasteGate single carrier alignment`

Theorem 2866.2 (Computable uniform continuity NameCert obligations)**.** *For every accepted* $\mathsf{ComputableUniformContinuity}^{\uparrow}$ *carrier $Q = (X, Y, K, R, M, W, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ComputableUniformC}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $X, Y, K, R, M, W, T, E, H, C, P, N$;*
2. *modulus visibility: every exported continuity read passes through the single* $\mathsf{UniformModulus}^{\uparrow}$ *row M before a target tolerance is accepted;*
3. *represented-space compatibility: the source row X, target row Y, and decoding row K are read together, so the modulus never acts on a detached host carrier;*
4. *window transport: the* $\mathsf{StreamName}^{\uparrow}$ *row W and* hsame *row H carry the same finite source and target name windows through every accepted replay;*
5. *tolerance stability: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E use the same dyadic budgets exposed by M and W;*
6. *ledger nonescape: H, C, P, N export no host function, choice-selected modulus, quotient representation, full* $\mathsf{Real}^{\uparrow}$ *completeness theorem, countable choice over names, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier in Definition 3838.1. Its accepted rows are exactly $X, Y, K, R, M, W, T, E, H, C, P, N$, so carrier admission exposes represented-space data, decoding, readback, modulus, windows, tolerance ledger, real seal, and structural rows only.

The modulus visibility and represented-space compatibility clauses are forced by the row order. A continuity consumer must first read the source and target represented-space rows X, Y, then the decoding row K, and then the displayed modulus realizer M before it can request a target tolerance.

The window and tolerance clauses read the finite ledgers. The row W lists the inspected source and target name windows, while T records the dyadic budgets that make the $\mathsf{RegSeqRat}^{\uparrow}$ readback R and terminal $\mathsf{Real}^{\uparrow}$ seal E available.

Finally inspect the structural rows. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed route and contain no coordinate for the excluded host function, selected modulus, quotient representation, full real completeness theorem, countable choice, or bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{ComputableUniformContinuity}^{\uparrow}$ read is refuted if it exports a uniform-continuity consumer without displaying represented-space source and target rows, CauchyRepresentation decoding, RegSeqRat readback, UniformModulus realizer, StreamName windows, DyadicRatCore tolerance ledger, RealUp seal, transport, replay, provenance, and local naming.

Independence witness: cauchy-representation, representedspace, streamname, regseqrat, dyadicratcore, real, uniformmodulus, continuousmap: none alone is bijective with $\mathsf{ComputableUniformContinuity}^{\uparrow}$ because the packet jointly records represented source and target rows, Cauchy decoding, a computable modulus realizer, finite name windows, rational readback, dyadic tolerance stability, terminal Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableUniformContinuity}^{\uparrow}$ packet is generated from represented-space source and target rows, CauchyRepresentation decoding, RegSeqRat readback, a UniformModulus realizer, StreamName finite windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3838.1 and Definition 3838.2 seed the finite computable-analysis route from represented-space names to a uniform-modulus Real consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions, choice-selected moduli, quotient representations, full RealUp completeness, countable choice over names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus visibility, represented-space compatibility, window transport, tolerance stability, ledger nonescape, and local naming for BEDC.Derived.ComputableUniformContinuityUp.

Chapter 2867

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy sequences, the $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ packet records the quotient-free separation certificate used when two completion names are compared before a terminal real seal is consumed. It sits on the finite observation route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the extension-and-uniqueness surface of chapter 2319. The packet names separation between displayed completion names; it is not quotient completion equality, selected representative choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2867.1 (Regular Cauchy completion separation carrier)**.** *A* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, D, W, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two completion-name source rows, each opened only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational route.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two names at a displayed precision.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window exposing the observations of A and B requested by D.*
4. *R is the regular-rational comparison row recording the inspected rational entries, common-tail data, and separation witness before any Real-facing read.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, D, W, R have been displayed.*
6. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *uniqueness row, read only as a finite extension-and-uniqueness boundary for the same displayed completion names.*
7. *H records componentwise* hsame *transport for $A, B, D, W, R, E, U, C, P, N$.*
8. *C records displayed* Cont *replay from a separation request through the two source rows, dyadic tolerance, stream window, regular rational comparison, terminal seal, and universal-property uniqueness row.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient representatives, host sequence equality, selected limits, countable choice over tails, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionSeparationUp.RegularCauchyCompletionSeparationTasteGate`
`single carrier alignment`

Theorem 2867.2 (Regular Cauchy completion separation NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *carrier* $S = (A, B, D, W, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$ *surface consists exactly of two-name source admission, shared dyadic tolerance exposure, finite StreamName window routing, regular rational comparison, terminal Real seal handoff, universal-property uniqueness routing, componentwise transport, continuation replay, package provenance, and local naming through the displayed rows* $A, B, D, W, R, E, U, H, C, P, N$.

Proof. Project the carrier of Definition 3839.1. The two completion names are visible only through A and B, and the comparison has a single displayed dyadic tolerance ledger D.

The finite window row W exposes the observations requested by D. The row R records the inspected regular-rational entries, common-tail data, and separation witness before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be read.

The universal-property row U supplies only the displayed uniqueness boundary already named by chapter 2319. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, and the carrier lacks any quotient representative, host equality, selected limit, countable-choice, trichotomy, or ambient completeness coordinate. □

Theorem 2867.3 (Regular Cauchy completion separation handoff)**.** *Every completion-facing consumer that compares two accepted* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *names must read the finite route*

$$A, B, D, W \longmapsto R \longmapsto E, U \longmapsto H, C, P, N$$

before treating the names as separated or identified by the completion boundary. The handoff routes through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *as displayed NameCert data, and it exports no quotient equality, selected representative, host real equality, countable-choice selector, trichotomy split, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier $S = (A, B, D, W, R, E, U, H, C, P, N)$. A comparison request first opens the two regular Cauchy source rows and the shared dyadic tolerance ledger.

The finite StreamName window W determines which observations of the two names are inspected. The regular-rational comparison row R then records the common-tail and separation data, so the request remains inside finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ evidence.

Only after that comparison can the terminal Real seal E and universal-property uniqueness row U be consumed. The structural rows H, C, P, N transport, replay, source, and name the same route, leaving no carrier coordinate for quotient equality, selected representatives, host equality, countable choice, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$.

Constructive story: A RegularCauchyCompletionSeparation$^{\uparrow}$ packet is generated from two regular Cauchy completion-name rows, a shared DyadicRatCore tolerance ledger, finite StreamName windows, a RegSeqRat comparison row, a terminal RealUp seal, a CauchyCompletionUniversalProperty uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 3839.1, Definition 3839.2, and Definition 3839.3 seed the quotient-free separation surface for displayed regular Cauchy completion names.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionSeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, host sequence equality, countable choice over tails, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-tolerance exactness, finite-window routing, regular-rational comparison, terminal seal handoff, universal-property uniqueness routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionSeparationUp.

Chapter 2868

A Concrete Naming Certificate for WRRHotspotCube$^{\uparrow}$

The WRRHotspotCube$^{\uparrow}$ packet realizes the purine-suffix hotspot cube of Definition 9462.1780, the reassignment count of Definition 9462.1781, and the highest-scoring cube certificate of Definition 9462.1783 as finite BEDC NameCert data. It consumes the DNA boundary-tile source of chapter 3393, the six-boundary coverage surface of chapter 3470, the finite trigger algebra of chapter 3478, the trigger-energy partition packet of chapter 3806, and the trigger-blocker duality packet of chapter 3801. The packet names C_{WRR} as finite cube, layer, transfer-square, count, and refusal data; it is not a biological causal model, stochastic evolutionary law, exhaustive code table, or hidden codon semantics.

Definition 2868.1 (WRR hotspot cube carrier)**.** *A* WRRHotspotCube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (V, A, L, T, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *V is the eight-vertex row for $C_{\mathrm{WRR}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}$.*
2. *A is the six-bit atlas row recording the free coordinates b_1^s, b_2^f, b_3^f and fixed coordinates $b_1^f = 0$, $b_2^s = 1$, and $b_3^s = 1$.*
3. *L is the layer-classifier row recording the $\mathrm{U} \times R \times R$ and $\mathrm{A} \times R \times R$ layers and the standard $3 + 2 + 2 + 1$ pattern.*
4. *T is the transfer-square row for $\{\mathrm{UGA}, \mathrm{UGG}, \mathrm{AGA}, \mathrm{AGG}\}$ and its Stop/Trp/Arg transition surface.*
5. *M is the reassignment-count ledger recording the codon counts $8, 10, 15, 0, 3, 0, 8, 8$ and total 52.*
6. *Q is the Q_3 score row recording that the displayed cube is the unique highest-scoring axis-aligned cube among the 160 displayed candidates.*
7. *H records componentwise* hsame *transport for vertex, atlas, layer, transfer-square, count, score, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source hotspot-cube vision through $V, A, L, T, M, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* WRRHotspotCube$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, TriggerEnergyPartition$^{\uparrow}$, TriggerBlockerDuality$^{\uparrow}$, *the source hotspot-cube vision,* BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$ *row.*
The carrier has no coordinate for biochemical mechanism, evolutionary ancestry, exhaustive biological coverage, stochastic causality, host codon semantics, or an unlisted reassignment table.

Lean4 ✓ `BEDC.Derived.WRRHotspotCubeUp.WRRHotspotCubeTasteGate single carrier alignment`

Theorem 2868.2 (WRR hotspot cube NameCert obligations)**.** *For every accepted* $\mathsf{WRRHotspotCube}^{\uparrow}$ *carrier* $W = (V, A, L, T, M, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$ *surface consists exactly of vertex exposure, six-bit atlas exposure, layer-classifier exposure, transfer-square exposure, reassignment-count ledger exactness,* Q_3 *score-row exposure, componentwise transport, continuation replay, package provenance, and local naming through the displayed rows* $V, A, L, T, M, Q, H, C, P, N$*.*

Proof. Project the carrier of Definition 3840.1. The vertex row V and atlas row A name the finite cube, while L and T expose the two layer and transfer-square classifiers already displayed in the source vision.

The count and score rows are finite ledgers. The row M records the eight codon-level reassignment counts and their total, and Q records the comparison against the displayed family of axis-aligned Q_3 cubes.

Transport, replay, provenance, and local naming preserve exactly those rows. Since the carrier has no coordinate for biochemical mechanism, evolutionary ancestry, stochastic causality, exhaustive biological coverage, hidden codon semantics, or an unlisted table, the listed rows exhaust the NameCert obligations. □

Theorem 2868.3 (WRR hotspot cube dependency handoff)**.** *Every downstream trigger-spectrum consumer of an accepted* $\mathsf{WRRHotspotCube}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{SixBoundaryTileCoverage}^{\uparrow},\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow},\ \mathsf{TriggerEnergyPartition}^{\uparrow},$$

The handoff exposes the cube vertices, atlas coordinates, layer classifier, transfer square, reassignment count ledger, and highest-score row before any consumer treats the hotspot cube as a finite NameCert packet.

Proof. The provenance row P lists the sibling dependency surface. The DNA boundary-tile packet supplies the source codon-window rows, the six-boundary coverage packet supplies the finite coverage coordinates, and the trigger algebra supplies the finite support and coverage-trigger readback.

The trigger-energy partition and trigger-blocker duality packets then supply the partition and blocker rows used by trigger-spectrum consumers. Only after those sibling rows are visible can V, A, L, T, M, Q be read as the hotspot-cube NameCert surface.

The structural rows H, C, P, N preserve and name the same route. Therefore the handoff is a finite dependency chain, not a biological mechanism, stochastic law, or hidden codon table. □

Falsifiable prediction: Any accepted $\mathsf{WRRHotspotCube}^{\uparrow}$ consumer that changes one of the eight displayed vertex rows or one of the count entries $8, 10, 15, 0, 3, 0, 8, 8$ must change the total-count row M or reject the packet.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra, trigger-energy-partition, trigger-blocker-duality: none is bijective with $\mathsf{WRRHotspotCube}^{\uparrow}$ because this packet jointly carries the WRR vertex cube, six-bit atlas row, layer classifier, transfer square, reassignment-count ledger, highest-score row, transport, replay, provenance, and local naming.

The Lean follow-up for this chapter must provide `Nontrivial WRRHotspotCubeUp` and `FieldFaithful WRRHotspotCubeUp` TasteGate instances under `BEDC.Derived.WRRHotspotCubeUp`, preserving $V, A, L, T, M, Q, H,$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$.

Constructive story: A $\mathsf{WRRHotspotCube}^{\uparrow}$ packet is generated from eight WRR cube vertices, six-bit atlas coordinates, layer-classifier rows, transfer-square rows, reassignment-count ledgers, highest-score rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3840.1, Definition 3840.2, Definition 3840.3, the falsifiable prediction, and the independence witness seed the finite WRR hotspot cube packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close biochemical causality, evolutionary ancestry, exhaustive biological coverage, stochastic semantics, hidden codon semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vertex exposure, atlas exposure, layer and transfer-square classifier exactness, count-ledger exactness, highest-score row exposure, Nontrivial WRRHotspotCubeUp, and FieldFaithful WRRHotspotCubeUp for BEDC.Derived.WRRHotspotCubeUp.

Chapter 2869

A Concrete Naming Certificate for $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$

Bishop–Bridges nested interval/regular Cauchy construction treats a decreasing located interval chain with a displayed diameter schedule as finite approximation data before any real point is read. The $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ packet records that handoff as a NameCert route from the nested-interval theorem surface of chapter 2308 and the located endpoint surface of chapter 394 through the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It names the L10 route from nested interval data to regular Cauchy real names; it is not an interval-intersection oracle, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2869.1 (Nested Cauchy interval-selection carrier)**.** *A* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *source row, exposing the finite descending interval windows, containment rows, and diameter controls.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, recording located lower and upper endpoint reads for the inspected intervals.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare interval diameter bounds with the requested regular Cauchy precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which endpoint and midpoint observations are requested.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy handoff row extracted from the nested interval windows and tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for interval, located endpoint, dyadic tolerance, stream-window, regular readback, real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a nested interval request through located endpoint reads, dyadic tolerances, stream windows, regular readback, and real seal handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$, $\mathsf{NestedIntervalTheorem}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $I, L, D, W, R, E, H, C, P, N$. It has no coordinate for a completed host intersection point, a quotient real representative, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedCauchyIntervalSelectionUp.NestedCauchyIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 2869.2 (Nested Cauchy interval-selection NameCert obligations)**.** *For every accepted* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *carrier $S = (I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}}$ *surface consists exactly of carrier admission for the displayed rows, nested-interval exposure through I, located endpoint exposure through L, dyadic tolerance exactness through D, finite-window exposure through W, regular Cauchy handoff through R, terminal real seal confinement through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3841.1. The rows $I, L, D, W, R, E, H, C, P, N$ are exactly the data visible to the local classifier.

The interval source and located endpoint rows are separate. The source row I carries the descending windows and diameter controls, while L carries the endpoint locatedness evidence needed to read those windows as rational approximation data.

The regular Cauchy handoff is downstream. The tolerance ledger D converts displayed diameter bounds into requested precision rows, the window row W supplies finite observations, and R packages the resulting observations as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The real seal and structural rows do not add a hidden selector. The row E is reached only after I, L, D, W, R have been replayed, and H, C, P, N preserve and name the same route. Hence the excluded host intersection, quotient representative, choice, trichotomy, compactness, and ambient-completeness coordinates are absent from the NameCert surface. □

Theorem 2869.3 (Nested Cauchy interval-selection regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{NestedIntervalTheorem}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports a regular Cauchy name read from displayed interval windows and diameter controls, not a completed intersection point or a host equality of real presentations.

Proof. Begin with an accepted carrier $S = (I, L, D, W, R, E, H, C, P, N)$. A regular Cauchy request first reads the nested interval theorem row I and the located endpoint row L.

Next the request consumes the dyadic tolerance ledger D and finite-window row W. These rows identify which displayed interval prefix supplies the requested Cauchy precision.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is then read as the regular Cauchy handoff. A $\mathsf{Real}^{\uparrow}$ consumer may proceed to E, but only through the same route and the structural rows H, C, P, N.

Since no carrier coordinate names a completed host intersection point, quotient real representative, countable-choice schedule, trichotomy split, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the handoff remains choice-free and finite. □

Falsifiable prediction: Every accepted $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ read exposes NestedIntervalTheorem rows, LocatedInterval endpoint rows, DyadicRatCore tolerance rows, StreamName finite windows, a RegSeqRat handoff, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any Real-facing consumer can use the packet.

Independence witness: nested-interval-theorem, locatedinterval, regseqrat, dyadicratcore, streamname, real: $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nested interval source, located endpoint evidence, dyadic tolerance ledger, finite-window route, regular Cauchy handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ packet is generated from a NestedIntervalTheorem source row, a LocatedInterval endpoint row, a DyadicRatCore tolerance ledger, StreamName finite windows, a RegSeqRat regular Cauchy handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3841.1, Definition 3841.2, and Definition 3841.3 seed the finite route from nested interval data to regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, host interval intersection, quotient real equality, selected limit points, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, nested interval exposure, located endpoint exposure, dyadic tolerance exactness, regular Cauchy handoff, Real seal confinement, and field-faithful TasteGate evidence for BEDC.Derived.NestedCauchyIntervalSelectionUp.

Chapter 2870

A Concrete Naming Certificate for $\mathsf{BoundaryPressureLedger}^{\uparrow}$

The $\mathsf{BoundaryPressureLedger}^{\uparrow}$ packet realizes the outer-boundary degree and pressure rows of Definition 9462.1298, Definition 9462.1300, and Definition 9462.1305 as finite BEDC NameCert data. It sits after the median-closure carrier of chapter 3545, the six-boundary trigger coverage route of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the coverage algebra of chapter 3478. The packet records pressure rows for inactive boundary codons before any trigger or class-pressure consumer may use them; it is not a biological stochastic model, an empirical fitness claim, a hidden codon table, or an unrestricted boundary enumerator.

Definition 2870.1 (Boundary-pressure ledger carrier)**.** *A* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, O, D, P, A, K, H, C, Q, N)$$

with the following displayed rows:

1. *S is the reassigned support row naming the displayed* 13*-codon support $\mathcal{R}$.*
2. *O is the outer-boundary row, listing exactly the inactive codons at Hamming distance one from $\mathcal{R}$.*
3. *D is the boundary-degree ledger, recording $d_{\partial}(z)$ for each displayed boundary codon z.*
4. *P is the boundary-pressure ledger, recording $P_{\partial}(z)$ from the displayed support-neighbor weights.*
5. *A is the aggregate class-pressure row, grouping boundary codons by standard-code class while preserving degree and pressure totals.*
6. *K is the high-pressure corner row naming the inactive envelope corners and the first completion frontier consumed by boundary-closure readers.*
7. *H records componentwise* hsame *transport for support, outer-boundary, degree, pressure, aggregate, corner, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-pressure request through $S, O, D, P, A, K, H, Q, N$.*
9. *Q is the* Pkg *provenance over* $\mathsf{BoundaryPressureLedger}^{\uparrow}$*,* $\mathsf{MedianClosure}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$ *row.*

The carrier accepts pressure reads only through $S, O, D, P, A, K, H, C, Q, N$. It has no coordinate for a pressure-row bypass, hidden biological semantics, stochastic selection dynamics, empirical likelihood, or a boundary row outside the displayed finite ledger.

Lean4 ✓ `BEDC.Derived.BoundaryPressureLedgerUp.BoundaryPressureLedgerTasteGate single carrier alignment`

Theorem 2870.2 (Boundary-pressure ledger NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *carrier* $B = (S, O, D, P, A, K, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$ *surface consists exactly of support admission through* S*, outer-boundary exposure through* O*, degree-ledger exactness through* D*, pressure-ledger exactness through* P*, class-pressure aggregation through* A*, inactive-corner and completion-frontier readback through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 3842.1. Its visible pressure data are exactly the rows $S, O, D, P, A, K, H, C, Q, N$.

The source support and boundary rows are finite. The row S names the reassigned support, and O lists the inactive codons at Hamming distance one from that support.

The degree and pressure rows are downstream of the same source. The row D counts displayed support neighbors for each boundary codon, while P sums the displayed support-neighbor weights. The aggregate row A groups those same entries by class without changing their provenance.

The corner, transport, replay, provenance, and naming rows preserve the ledger. Because no coordinate bypasses D and P, a consumer cannot obtain pressure data from hidden biological semantics, stochastic selection dynamics, empirical likelihood, or an unrestricted boundary enumerator. □

Lean4 ✓ `BEDC.Derived.BoundaryPressureLedgerUp.BoundaryPressureLedgerNamecertObligations`

Theorem 2870.3 (Boundary-pressure trigger handoff)**.** *Every boundary-trigger or class-pressure consumer of an accepted* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *packet factors through*

$$S \longmapsto O \longmapsto D \longmapsto P \longmapsto A \longmapsto K \longmapsto H, C, Q, N.$$

In particular, any consumer preserving $\mathsf{MedianClosure}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$*, and* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *provenance must read the degree and pressure ledgers before it can use high-pressure inactive boundary codons.*

Proof. Begin with a boundary-trigger request. The provenance row Q names the median-closure and boundary-trigger siblings, but the request cannot read those siblings as pressure rows unless the local support row S and outer-boundary row O have first been displayed.

The pressure data then pass through D and P. Degree is counted from displayed support neighbors, and pressure is the displayed weighted sum over the same neighbor relation.

The aggregate and corner rows are read only after the degree and pressure ledger. The row A groups class totals, while K names the inactive envelope corners and completion-frontier entries consumed by boundary-closure readers.

Finally H, C, Q, N transport, replay, source, and name the same finite route. Thus a pressure-row bypass would refute the packet: it would use high-pressure boundary data without passing through the displayed degree and pressure ledgers. □

Falsifiable prediction: A pressure-row bypass refutes $\mathsf{BoundaryPressureLedger}^{\uparrow}$: any accepted boundary-trigger consumer that reaches high-pressure inactive codons without first reading $S, O, D, P, A, K, H, C, Q$ is outside the packet.

Independence witness: median-closure, six-boundary-tile-coverage, dna-boundary-trigger-closure, window-coverage-trigger-algebra: $\mathsf{BoundaryPressureLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the outer-boundary row, degree ledger, pressure ledger, class aggregation, inactive-corner frontier, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.BoundaryPressureLedgerUp.TasteGate` must provide explicit $\mathsf{Nontrivial\ BoundaryPressureLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundaryPressureLedger}^{\uparrow}$ instances preserving $S, O, D, P, A, K, H, C, Q, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryPressureLedger}^{\uparrow}$ packet is generated from a reassigned support row, an outer-boundary row, boundary-degree and boundary-pressure ledgers, class-pressure aggregation, inactive-corner frontier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3842.1, Definition 3842.2, and Definition 3842.3 seed the finite pressure-ledger route for DNA boundary-trigger consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biological stochastic semantics, empirical likelihood, hidden codon semantics, unrestricted boundary enumeration, or pressure-row bypasses.

Upgrade path: Obligation closure requires checked support admission, outer-boundary exposure, degree-ledger exactness, pressure-ledger exactness, class aggregation, trigger handoff, Nontrivial BoundaryPressureLedgerUp, and FieldFaithful BoundaryPressureLedgerUp for BEDC.Derived.BoundaryPressureLedgerUp.

Chapter 2871

A Concrete Naming Certificate for $\mathsf{CompletionAdjoint}^{\uparrow}$

Following Mac Lane and Bishop–Bridges, the $\mathsf{CompletionAdjoint}^{\uparrow}$ packet records the Cauchy completion reflector as a finite adjunction certificate. It sits after the reflector surface of chapter 1585, the functorial completion surface of chapter 423, the universal extension ledger of chapter 397, the reflective-subcategory boundary of chapter 1793, the regular rational and stream-window interfaces of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed unit, counit, and universal-property rows; it is not a quotient completion, a selected limit object, an ambient reflective subcategory theorem, or an appeal to countable choice.

Definition 2871.1 (Completion adjoint carrier)**.** *A* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, F, U, V, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row from chapter 1585, recording the source-to-complete handoff through the Cauchy completion reflector surface.*
2. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row from chapter 423, carrying the functorial source, regular-Cauchy, seal, and classifier data.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row from chapter 397, recording extension existence, extension uniqueness, and unit compatibility for the same displayed source.*
4. *V is the* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$ *visibility row from chapter 1793: it records that complete objects are read through the reflected completion surface, not through a host subcategory object.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule for completion requests, approximating maps, unit reads, and counit reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular Cauchy names inspected by F and by the universal-property row U.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for all approximation and triangle reads used by the unit-counit ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after R, F, U, V, S, Q, D have been displayed.*

9. *H records componentwise* hsame *transport for the reflector, functor, universal-property, reflective-subcategory, stream, readback, dyadic, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from any completion-adjunction consumer through the rows R, F, U, V, S, Q, D, E and then through the structural rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CompletionAdjoint}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, host reflective subcategories, functor extensionality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionAdjointUp.CompletionAdjointTasteGate single carrier alignment`

Theorem 2871.2 (Completion adjoint NameCert obligations)**.** *For every accepted* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier $A = (R, F, U, V, S, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows; reflector exposure through R; functorial completion exposure through F; universal-property exposure through U; reflective-subcategory visibility through V; finite request scheduling through S; regular rational readback through Q; dyadic tolerance accounting through D; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier rows. The adjunction-facing data are precisely the reflector row R, the functor row F, the universal-property row U, the reflective-subcategory visibility row V, and the finite analytic rows S, Q, D, E.

Read the structural rows separately. The transport row H preserves those displayed rows componentwise, while C replays the same route for completion-adjunction consumers.

The provenance and naming rows P, N attach the packet to the cited completion, stream, regular-readback, dyadic, and Real certificates. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 2871.3 (Completion adjoint unit-counit ledger)**.** *Every unit or counit read exported by an accepted* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier factors through the finite route*

$$R, F, U, V \longrightarrow S, Q, D \longrightarrow E, H, C, P, N.$$

The unit side reads the source-to-completion handoff through the reflector and functor rows, while the counit side reads the complete-object return through the universal-property and reflective-subcategory visibility rows. Both sides use the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the structural ledger is inspected.*

Proof. Start with the rows R, F, U, V. They are the only coordinates in the packet that can name the reflector, functor, extension, uniqueness, and reflected-object handoff needed by an adjunction consumer.

Next read S, Q, D, E. These rows supply finite request schedules, regular rational readback, dyadic tolerance accounting, and the terminal Real seal for both the unit-facing and counit-facing directions.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same displayed ledger, so the unit and counit routes cannot escape to a quotient completion, selected limit, host reflective subcategory, functor-extensional equality, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionAdjoint}^{\uparrow}$ packet is generated from CauchyCompletionReflector, CauchyCompletionFunctor, CompletionUniversalProperty, CauchyCompletionReflectiveSubcategory, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3843.1, Definition 3843.2, and Definition 3843.3 seed the finite adjunction certificate for Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected limits, host reflective subcategories, functor extensionality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, reflector exposure, functorial completion exposure, universal-property compatibility, unit-counit ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompletionAdjointUp.

Chapter 2872

A Concrete Naming Certificate for UniformEquicontinuityModulus$^{\uparrow}$

Following Arzela–Ascoli and Bishop–Bridges equicontinuity arguments, the UniformEquicontinuityModulus$^{\uparrow}$ packet records the finite-net route by which a compact source and a finite family of continuous maps expose one shared modulus before any open-cover compactness or choice principle is read. It consumes compact-metric data from chapter 129, total-bounded finite nets from chapter 127, continuous-map graph and pointwise modulus rows from chapter 130, compact uniform-continuity handoff data from chapter 3444, and the downstream modulus surface of chapter 369. The certificate is a finite uniform-equicontinuity route for compact-window consumers, not a selected subcover, not a family-choice functional, and not ambient completeness of the codomain.

Definition 2872.1 (UniformEquicontinuityModulus carrier)**.** *A* UniformEquicontinuityModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, T, F, W, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing only the compact metric carrier, distance ledger, and compact finite-net surface used by the family.*
2. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row over the same source, listing inspected centers and dyadic source-radius budgets.*
3. *F is the finite family of* ContinuousMap$^{\uparrow}$ *rows whose pointwise modulus and graph rows are inspected at the centers named by T.*
4. *W is the finite target-window row, recording the common tolerance window and the family indices admitted by the packet.*
5. *M is the exported* UniformModulus$^{\uparrow}$ *row, obtained by a finite fold over the centerwise and familywise modulus witnesses displayed by F against T and W.*
6. *H records componentwise* hsame *transport for the compact source, finite-net, continuous-family, window, folded-modulus, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a compact-window consumer through K, T, F, W, M and then through the structural rows.*

8. *P is the* Pkg *provenance over* UniformEquicontinuityModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, ContinuousMap$^{\uparrow}$, HeineCantor$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{UniformEquicontinuityModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected finite subcovers, countable choice over the family, quotient equality of maps, host real trichotomy, or ambient codomain completeness.

Lean4 ✓ `BEDC.Derived.UniformEquicontinuityModulusUp.UniformEquicontinuityModulusTasteGate`
`single carrier alignment`

Theorem 2872.2 (UniformEquicontinuityModulus NameCert obligations)**.** *For every accepted* UniformEquicontinuityModulus$^{\uparrow}$ *carrier $U = (K, T, F, W, M, H, C, P, N)$, the local* NameCert$_{\text{UniformEquicontinuityModulus}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the finite packet $K, T, F, W, M, H, C, P, N$;*
2. *finite-window classifier: every family and target-tolerance read is confined to the displayed window row W and the listed* ContinuousMap$^{\uparrow}$ *rows in F;*
3. *compact finite-net exactness: the only compactness data read by the classifier are the* CompactMetric$^{\uparrow}$ *source row K and the* TotallyBounded$^{\uparrow}$ *finite-net row T;*
4. *uniform-modulus export: the shared modulus row M is obtained by the finite fold over the centerwise pointwise-modulus rows in F and the compact finite-net route already exposed by K, T;*
5. *non-escape: no accepted consumer receives an open-cover compactness object, selected subcover, family-choice functional, quotient map equality, host real trichotomy, or ambient completeness theorem.*

Proof. Project the carrier row by row. The analytic data available before the exported modulus are exactly the compact source K, the finite-net inventory T, the finite continuous-family rows F, and the common target window W.

The compact finite-net exactness clause follows from the route through K and T. Since T lists the centers and radius budgets as finite data, the classifier never needs an open-cover subcover or a choice over covers.

The uniform-modulus export is then the finite fold stored in M. Each fold input is a pointwise modulus row from F at a center named by T and a tolerance named by W; the Heine–Cantor handoff supplies the compact-window reading of those rows, while the final row is presented to the UniformModulus$^{\uparrow}$ surface.

Finally read the structural rows. The transport row H, replay row C, provenance row P, and local naming row N preserve the same finite route and contain no coordinate for the excluded cover, choice, equality, trichotomy, or completeness data. □

Falsifiable prediction: An accepted UniformEquicontinuityModulus$^{\uparrow}$ read is refuted if it exports a shared family modulus without displaying the compact source row, total-bounded finite-net row, finite continuous-family rows, target-window row, folded modulus row, transport, replay, provenance, and local naming rows.

Independence witness: compactmetric, totallybounded, continuousmap, heine-cantor, uniformmodulus: none alone is bijective with UniformEquicontinuityModulus$^{\uparrow}$ because the packet jointly records a compact source, finite net, finite continuous family, common target window, folded uniform modulus, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ packet is generated from CompactMetric source data, TotallyBounded finite nets, ContinuousMap family rows, a common finite target window, a finite folded UniformModulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3844.1 and Definition 3844.2 seed the compact finite-net route from a finite continuous family to a shared UniformModulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable family choice, quotient equality of maps, host real trichotomy, ambient codomain completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window classification, compact finite-net exactness, uniform-modulus export, transport, replay, provenance, and local naming for BEDC.Derived.UniformEquicontinuityModulusUp.

Chapter 2873

A Concrete Naming Certificate for $\mathsf{LimitSpace}^{\uparrow}$

Following Kelley and Bishop–Bridges convergence-space and limit-space terminology, the $\mathsf{LimitSpace}^{\uparrow}$ packet records the finite interface by which a displayed Cauchy-filter or filter-base row is read against an observation schedule, a dyadic tolerance ledger, a regular rational readback, and a terminal real limit seal. It consumes the Cauchy-filter surface of chapter 333, the finite observation schedule of chapter 126, the dyadic tolerance core of chapter 291, the regular rational route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a convergence-space interface, not a quotient filter, selected limit, host equality of convergent objects, or ambient completeness theorem.

Definition 2873.1 (LimitSpace carrier)**.** *A* $\mathsf{LimitSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, D, R, L_0, H, C, P, N)$$

with the following displayed rows:

1. *F is a* $\mathsf{CauchyFilter}^{\uparrow}$ *or filter-base row, carrying the finite filter requests and Cauchy-stability witnesses permitted at this limit-space site;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule that records which finite windows of F are inspected;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the scheduled and tolerated observations into regular rational data;*
5. *L_0 is the* $\mathsf{Real}^{\uparrow}$ *limit-seal row reached only after F, S, D, R have been displayed;*
6. *H records componentwise* hsame *transport for the filter, schedule, tolerance, readback, limit seal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a convergence request through F, S, D, R, L_0, H, P, N;*
8. *P is the* Pkg *provenance over* $\mathsf{LimitSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, host equality of convergent filters, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge evidence.*

Lean4 ✓ `BEDC.Derived.LimitSpaceUp.LimitSpaceTasteGate single carrier alignment`

Theorem 2873.2 (LimitSpace NameCert obligations)**.** *For every accepted* $\mathsf{LimitSpace}^{\uparrow}$ *carrier*

$$L = (F, S, D, R, L_0, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Cauchy-filter or filter-base row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, Real limit seal, transport, replay, provenance, and local naming row;*
2. *filter-source exposure, requiring every limit-space consumer to read F before any scheduled observation, tolerance comparison, readback, or limit seal is inspected;*
3. *schedule and tolerance discipline, requiring S and D to determine the finite windows and dyadic budgets used by the readback row;*
4. *regular readback discipline, requiring R to be downstream of F, S, D and upstream of the terminal seal;*
5. *limit-seal non-escape, requiring the* $\mathsf{Real}^{\uparrow}$ *row L_0 to be read only as a seal for the displayed readback route;*
6. *structural preservation through H, C, P, N;*
7. *refusal of quotient filters, selected limits, countable choice, host equality of convergent filters, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge evidence.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3845.1. The tuple displays exactly the Cauchy-filter or filter-base source, finite schedule, dyadic tolerance ledger, regular rational readback, real limit seal, and structural rows.

The source obligation is first. A convergence consumer cannot reach a scheduled observation or terminal seal without reading F, because no other row carries filter-source data.

The schedule and tolerance obligations are then forced by S and D. They determine the finite windows and dyadic budgets that the regular readback row may inspect, so R is downstream of the displayed observation discipline rather than a separate selected representative.

Finally L_0 is exposed only as the real seal for that readback route, while H, C, P, N preserve, replay, source, and name the same finite packet. Since the excluded principles have no carrier coordinate, they cannot become NameCert obligations. □

Theorem 2873.3 (LimitSpace Cauchy-filter consumer route)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LimitSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$F \longmapsto S, D \longmapsto R \longmapsto L_0 \longmapsto H, C, P, N.$$

The consumer first reads the Cauchy-filter or filter-base row, then the StreamName schedule and DyadicRatCore tolerance ledger, then the RegSeqRat readback, and only then the Real limit seal. The route exports no quotient filter, no selected limit, no host equality of convergent filters, no countable-choice schedule, and no ambient completeness theorem.

Proof. Project the accepted carrier. The first row F is the only source for Cauchy-filter or filter-base data, so every filter-facing read starts there.

The next step is the finite observation discipline. The schedule row S names the inspected windows and the tolerance row D names the dyadic budgets attached to them; without those two rows the regular readback row has no displayed finite request to answer.

The readback and seal rows are downstream. The row R converts the scheduled observations into regular rational data, and L_0 seals the resulting route as a $\mathsf{Real}^{\uparrow}$ limit row without selecting an external limit object.

The structural rows H, C, P, N preserve and name the same route. Because the carrier contains no quotient-filter coordinate, selected-limit coordinate, host equality row, choice schedule, or ambient completeness row, every accepted consumer factors through the displayed path. □

Falsifiable prediction: An accepted $\mathsf{LimitSpace}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ limit seal without first displaying the $\mathsf{CauchyFilter}^{\uparrow}$ or filter-base row, the $\mathsf{StreamName}^{\uparrow}$ schedule, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger refutes the carrier surface.

Independence witness: cauchyfilter, streamname, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{LimitSpace}^{\uparrow}$ because this packet jointly carries a filter-source row, observation schedule, tolerance ledger, regular readback, terminal limit seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LimitSpace}^{\uparrow}$ packet is generated from a CauchyFilter or filter-base row, StreamName observation schedule, DyadicRatCore tolerance ledger, RegSeqRat readback row, Real limit seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3845.1, Definition 3845.2, and Definition 3845.3 seed the finite convergence-space interface from CauchyFilter rows through StreamName, DyadicRatCore, RegSeqRat, and Real limit-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality of convergent filters, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-source exposure, schedule and tolerance discipline, regular readback discipline, Real limit-seal non-escape, and structural preservation for BEDC.Derived.LimitSpaceUp.

$\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ reads compactness-facing finite-window criteria through chapter 129, chapter 127, and chapter 3844 before its Arzela–Ascoli and completion consumers inspect the local rows.

Chapter 2874

A Concrete Naming Certificate for $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$

Following Kolmogorov–Riesz and Frechet compactness in constructive finite-window form, the $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet records the BEDC route by which compact metric control, total boundedness, uniform equicontinuity, completion readiness, and terminal Real readbacks are named as one compactness-facing criterion. It sits after the compact-metric surface of chapter 129, the total-bounded source of chapter 127, the uniform-equicontinuity modulus of chapter 3844, the complete-metric handoff of chapter 128, and the terminal real-name seal of chapter 38. Its compactness route also consumes bounded-set rows from chapter 1757, equicontinuous-sequence rows from chapter 4356, and finite-net compact-metric product rows from chapter 1822. The carrier names finite net and completion rows only; it does not assert open-cover compactness, choose subsequences, select subcovers, or import ambient completeness outside the displayed packet.

As a carrier dependency, the packet consumes the compactness route through chapter 129, the equicontinuity-family route through chapter 413, and the real and metric boundary routes through chapter 38 and chapter 57. These rows are part of the $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ admission surface: a compactness consumer must see the compact metric source, family-equicontinuity ledger, Real readback seal, and metric carrier before the Frechet–Kolmogorov criterion row is exported.

Definition 2874.1 (Frechet-Kolmogorov compactness carrier)**.** *A* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, E, Q, C, S, R, H, A, P, N)$$

with the following rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the displayed metric and compactness-facing finite-net interface.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row recording the finite ε-net budget consumed by the criterion.*
3. *E is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row binding a shared modulus to the same finite windows.*
4. *Q is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion boundary row, used only after the finite net and modulus rows are visible.*
5. *C is the compactness-criterion row relating the metric, total-bounded, equicontinuity, and completion rows as one finite BEDC route.*

6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the inspected net cells, modulus requests, and Cauchy-readback windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance comparisons.*
8. *H records componentwise* hsame *transport for all displayed source, net, modulus, completion, readback, and naming rows.*
9. *A records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, T, E, Q, C, S, R, H, A, P, N$. It has no coordinate for an open-cover compactness theorem, selected subsequence, selected subcover, quotient completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2874.2 (Frechet-Kolmogorov compactness NameCert obligations)**.** *For every accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier $K = (M, T, E, Q, C, S, R, H, A, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCor}}$ *surface consists exactly of compact-metric source exposure through M; total-bounded finite-net exposure through T; uniform-modulus exposure through E; completion-boundary exposure through Q; criterion compatibility through C; finite-window and rational readback exposure through S, R; componentwise transport through H; displayed replay through A; provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The coordinates are precisely the compact-metric source, total-bounded net, uniform-modulus, completion-boundary, criterion, finite-window, readback, transport, replay, provenance, and naming rows.

Read the source rows in order. The compact-metric row gives the metric-facing carrier; the total-bounded row supplies finite nets over that metric; the uniform-equicontinuity row uses the same windows; and the completion row is reached only through those displayed finite controls.

The structural rows H, A, P, N transport, replay, source, and name that same route. Since no carrier coordinate names a selected subsequence, selected subcover, open-cover compactness proof, quotient equality, or ambient completeness theorem, the listed rows exhaust the local certificate surface. □

Theorem 2874.3 (Frechet-Kolmogorov compactness total-bounded handoff)**.** *Every compactness-facing consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier must read the total-bounded row T through the compact-metric source M and the uniform-modulus row E before it inspects the StreamName and RegSeqRat readback rows S, R. Thus finite-net evidence is exported as a displayed BEDC handoff, not as a choice-selected dense set or an unrecorded cover refinement.*

Proof. Begin with the compact-metric source row. It fixes the metric and the carrier surface on which the total-bounded net is meaningful.

The total-bounded row then exposes the finite net budget, while the uniform-modulus row forces all later compactness requests to use the same finite tolerance windows. Only after those rows are visible can the StreamName and RegSeqRat rows read the inspected windows and rational tolerances.

Therefore the consumer receives a finite-net handoff through M, T, E, S, R and not a selected dense subset, a selected subcover, or an unrecorded refinement outside the packet. □

Theorem 2874.4 (Frechet-Kolmogorov compactness sibling route)**.** *Every compactness-facing export of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier consumes the compactness, total-boundedness, and equicontinuity sibling surfaces before it exports compactness of the function family. The route enters through the* $\mathsf{CompactMetric}^{\uparrow}$ *source of chapter 129, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row of chapter 127, and the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row of chapter 3844; it may then hand the assembled finite-window criterion to the Arzela–Ascoli finite-net surface of chapter 7576. The packet exports no open-cover compactness proof, selected subsequence, selected subcover, quotient completion equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier and NameCert obligations. The rows M, T, E are respectively the compact-metric source, total-bounded finite-net budget, and uniform-equicontinuity modulus, so no compactness-facing read can begin outside those sibling surfaces.

Use the total-bounded handoff. It forces finite-net evidence through the compact-metric source and the shared modulus before StreamName and RegSeqRat readbacks are inspected.

Finally use the completion boundary theorem. The row Q can be consumed only after the compactness, total-boundedness, and equicontinuity rows are visible, and the structural rows H, A, P, N preserve the same finite route for any Arzela–Ascoli or completion-facing consumer. □

Theorem 2874.5 (Frechet-Kolmogorov compactness Arzela-Ascoli dependency)**.** *Every Arzela–Ascoli-facing consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier first reads the* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already exposed by the carrier. The Arzela–Ascoli finite-net surface may consume this route, but it does not bypass the compact-metric source, total-bounded handoff, uniform-modulus row, completion boundary, finite windows, rational readback, or Real seal.*

Proof. Apply the sibling route of Definition 3846.4. It already directs compactness-facing exports through the CompactMetric, TotallyBounded, and UniformEquicontinuityModulus sibling surfaces before any Arzela–Ascoli finite-net consumer receives the assembled criterion.

Use the total-bounded handoff of Definition 3846.3. The finite-net budget is read through the compact-metric source and shared modulus before the StreamName and RegSeqRat rows expose inspected windows and rational tolerance comparisons.

Finally apply the completion boundary of Definition 3846.6. The CompleteMetric row and Real-facing seal remain downstream boundaries over the same finite data, so an Arzela–Ascoli-facing consumer cannot obtain open-cover compactness, a selected subsequence, a selected subcover, quotient completion equality, or ambient Real completeness outside the displayed route. □

Theorem 2874.6 (Frechet-Kolmogorov compactness completion boundary)**.** *The completion-facing part of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier is confined to the displayed row* Q *after the compact-metric source, total-bounded net, and uniform-modulus rows have been read. It may hand a downstream completion consumer a finite Cauchy-window boundary, but it does not claim quotient completion equality, a selected limit representative, sequential compactness, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the route through M, T, E first. These rows supply the metric source, finite-net budget, and shared modulus required before any completion boundary can be inspected.

The completion row Q is therefore a boundary row over already displayed finite data. The readback rows S, R and the structural rows H, A, P, N keep the same route visible under windowing, rational comparison, transport, replay, provenance, and naming.

Because the carrier contains no quotient-equality, selected-limit, sequential-compactness, open-cover, or ambient-completeness coordinate, a completion consumer cannot obtain any of those objects from the local packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet is generated from a CompactMetric source row, a TotallyBounded finite-net row, a UniformEquicontinuityModulus row, a CompleteMetric boundary row, StreamName finite windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3846.1, Definition 3846.2, Definition 3846.3, Definition 3846.4, Definition 3846.5, and Definition 3846.6 seed the finite compactness criterion over compact-metric, total-bounded, uniform-modulus, completion-boundary, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, selected subsequences, selected subcovers, quotient completion equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, uniform-modulus compatibility, completion-boundary nonescape, readback stability, transport, replay, provenance, and local naming for BEDC.Derived.FrechetKolmogoro

Theorem 2874.7 (Frechet-Kolmogorov finite-window compactness handoff)**.** *Every accepted* FrechetKolmogorovCo *compactness read that is exported to a finite-net or compact-metric consumer must route through the displayed boundedness row, finite translation-modulus row, tail-cutoff row, finite-net extraction row,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Met}^{\uparrow}$ *distance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ *row before compactness is public. In carrier notation the handoff refines the local route*

$$M, T, E, Q, C, S, R, H, A, P, N$$

by requiring boundedness, translation control, tail cutoff, finite nets, Real readback, rational readback, and metric-distance rows to be visible before the consumer can use the compactness criterion. The handoff exports no ambient compactness theorem, open-cover compactness, countable choice, selected subsequence, selected subcover, quotient completion equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the local NameCert obligations of Definition 3846.2. They force the compact-metric source M, total-bounded finite-net row T, uniform-modulus row E, completion boundary Q, finite-window row S, readback row R, and structural rows H, A, P, N to be displayed before a compactness-facing consumer receives the packet.

Read the additional finite-window controls through the same route. The boundedness row is supplied by the bounded-set dependency already named in the chapter introduction, the finite translation-modulus and tail-cutoff rows are consumed as uniform-window controls, and the finite-net extraction row is read through the total-bounded handoff of Definition 3846.3. The $\mathsf{Met}^{\uparrow}$ distance rows and the $\mathsf{Real}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}$ readbacks attach the tolerance comparisons to the displayed windows.

Finally apply the completion boundary of Definition 3846.6. The structural rows only transport, replay, source, and locally name the same finite handoff, so no coordinate remains for ambient compactness, open-cover compactness, countable choice, selected subsequences, selected subcovers, quotient completion equality, or ambient Real completeness. □

The finite compactness route of $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ feeds the Arzela–Ascoli finite-modulus criterion of chapter 3568 only after the compact metric source of chapter 129, the finite-net

discipline of chapter 127, and the uniform-equicontinuity modulus of chapter 3844 have been displayed. A compact-uniform-continuity consumer therefore receives compact metric, total-bounded, uniform-modulus, completion, transport, replay, provenance, and local NameCert rows, not an open-cover compactness theorem or a selected subsequence.

Theorem 2874.8 (Frechet–Kolmogorov Arzela–Ascoli dependency route)**.** *Every accepted* FrechetKolmogorovComp *compactness read consumes the equicontinuity and finite-subfamily surface of chapter 412 before it exports compactness for function families. The Frechet–Kolmogorov packet therefore routes compact metric data, total-bounded finite nets, bounded-family rows, Arzela–Ascoli equicontinuity rows, transport, replay, provenance, and local* NameCert *naming as one displayed dependency chain.*

Proof. Project the carrier through the local Frechet–Kolmogorov route. The compact source and finite-net rows are visible before any family compactness conclusion is offered to a downstream consumer.

Now read the Arzela–Ascoli sibling. Its equicontinuity and finite-subfamily surface supplies the function-family control that the Frechet–Kolmogorov packet consumes before exporting compactness.

The structural rows only transport, replay, source, and name this chain. Hence the dependency is horizontal and finite, not an ambient compactness theorem or a selected subsequence principle. □

Theorem 2874.9 (Frechet–Kolmogorov Arzela–Ascoli compact handoff)**.** *Every accepted* FrechetKolmogorovComp *handoff to an Arzela–Ascoli consumer must first display the compact-metric finite nets of chapter 129, the total-bounded finite-net rows of chapter 127, and the equicontinuity surface of chapter 412. The exported compactness packet is therefore a finite* BHist *route through compact metric, total-boundedness, Arzela–Ascoli finite-window selection, transport, replay, provenance, and local* NameCert *rows.*

Proof. Read the accepted Frechet–Kolmogorov carrier through its compact metric source. The compact-metric row fixes the metric-facing source, while the total-bounded row supplies the finite nets that any compactness-facing consumer can inspect.

Next read the Arzela–Ascoli dependency. Its equicontinuity and finite-window rows are consumed after those compact metric and total-bounded rows, so the function-family compactness conclusion is not detached from the finite net budget.

The remaining rows only preserve the displayed route. Componentwise hsame transport, Cont replay, provenance, and local naming keep the handoff inside the same finite packet, excluding selected subsequences, open-cover compactness, and ambient completeness. □

Theorem 2874.10 (Frechet–Kolmogorov downstream Arzela–Ascoli link)**.** *Every Arzela–Ascoli finite-net consumer of an accepted* FrechetKolmogorovCompactness$^{\uparrow}$ *packet must first read the* CompactMetric$^{\uparrow}$ *source row, the* TotallyBounded$^{\uparrow}$ *finite-net row, the* UniformEquicontinuityModulus$^{\uparrow}$ *row, the* CompleteMetric$^{\uparrow}$ *completion boundary, the* StreamName$^{\uparrow}$ *observation windows, the* RegSeqRat$^{\uparrow}$ *readback rows, and the* Real$^{\uparrow}$ *seal rows carried by the displayed dependency route. The handoff exports no selected subsequence, selected subcover, open-cover compactness theorem, quotient completion equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the existing Frechet–Kolmogorov carrier through the compactness dependency route. The compact metric source and total-bounded finite-net rows are visible before any function-family compactness read is offered downstream.

Read the Arzela–Ascoli-facing rows next. The dependency route of Definition 3846.8 and the compact handoff of Definition 3846.9 require the uniform-equicontinuity modulus before the finite-net consumer receives the family surface.

Finally inspect the completion and real-readback side. The compact metric total-bounded dependency of Definition 3846.11 keeps the $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows attached to the same finite route. Since no route theorem supplies a selected subsequence, selected subcover, open-cover compactness theorem, quotient completion equality, or ambient completeness coordinate, the downstream Arzela–Ascoli link exports only the displayed finite lattice. □

Theorem 2874.11 (Compact metric total-bounded dependency)**.** *The* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *criterion consumes the compact-metric source of chapter 129 and the total-bounded finite-net budget of chapter 127 before it reads the uniform-equicontinuity row of chapter 3844 and the completion boundary of chapter 128. Thus the later StreamName, RegSeqRat, and Real readbacks remain attached to displayed finite metric and net data, rather than to an unrecorded compactness principle.*

The $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet exposes a real-sequence compactness read only after its compact-metric finite-net rows, bounded-function-family rows, and Arzela–Ascoli finite-window rows have all been displayed. Thus every downstream compactness consumer first passes through the compact source of chapter 129, the bounded family surface of chapter 3522, and the Arzela–Ascoli route of chapter 412; before real-sequence compactness readback, it also consumes the finite epsilon-net chain of chapter 8049. The Frechet–Kolmogorov carrier then contributes only the finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows that assemble those dependencies into the real-sequence compactness read.

Theorem 2874.12 (Frechet-Kolmogorov locally convex boundedness route)**.** *Every* $\mathsf{FrechetKolmogorovCompactnes}$ *consumer that reads compactness through locally convex bounded sets must factor through the finite route*

$$\mathsf{LocallyConvexSpace}^{\uparrow}, \mathsf{FrechetSpace}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer must expose the locally convex seminorm row of chapter 3386, the Frechet metric completion row of chapter 3495, the compact metric row of chapter 129, the total bounded row of chapter 127, and the terminal Real seal row of chapter 38 before it may treat a bounded family as Frechet-Kolmogorov compactness data.

Proof. Start with the locally convex seminorm row. It supplies the bounded-set requests that a functional-analysis consumer is allowed to ask of the packet.

The Frechet metric completion row then turns those seminorm requests into a displayed metric-completion boundary. The compact metric row and total bounded row refine that boundary into finite compactness data, so the consumer sees nets and metric windows rather than an ambient bounded-set compactness theorem.

Finally the Real seal row records the scalar readback used by the seminorm and metric estimates. Since each step is an exposed row in the displayed route, no consumer can bypass the seminorm, completion, compact metric, total bounded, or Real-seal surfaces by appealing to host boundedness or selected compactness data. □

Theorem 2874.13 (Frechet–Kolmogorov metric-completion functor route)**.** *Every* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier must first read the compact-metric source of chapter 129, the total-bounded finite-net surface of chapter 127, the uniform-equicontinuity modulus of chapter 3844, the complete-metric boundary of chapter 128, the StreamName window rows of chapter 126, the RegSeqRat readback of chapter 301, and the Real seal of*

chapter 38. The route exports a displayed compactness-to-completion handoff only; it exports no quotient completion equality, selected compact limit, selected subsequence, selected subcover, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier of Definition 3846.1. Its compactness-facing source rows are the compact metric row, the total-bounded finite-net row, the uniform-modulus row, and the completion boundary row.

Apply the sibling route of Definition 3846.16. It keeps the compact metric, total-bounded, uniform-equicontinuity, RegSeqRat, and Real-seal rows visible as one dependency lattice before any compactness conclusion is exported.

Finally apply the completion boundary of Definition 3846.6. A $\mathsf{MetricCompletionFunctor}^{\uparrow}$ consumer can only repackage that displayed boundary through StreamName and RegSeqRat readbacks, so quotient completion equality, selected compact limits, and ambient completeness are outside the exported surface. □

The $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet is read through a substantive compactness sibling lattice rather than as a detached compactness kernel: its equicontinuity branch consumes the finite-modulus Arzela–Ascoli route of chapter 412, its compact source consumes the $\mathsf{CompactMetric}^{\uparrow}$ total-bounded finite-net rows of chapter 129, and its completion handoff is routed through the MetricCompletion completion surface of chapter 1506. These references keep compact metric, total-bounded finite nets, completion, transport, replay, provenance, and local NameCert rows visible before any compactness-facing consumer reads the Frechet–Kolmogorov packet.

Theorem 2874.14 (Frechet-Kolmogorov Sobolev compact-embedding forward link)**.** *Every Sobolev-family consumer that reaches the* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *packet through a compact-embedding route must first display the* $\mathsf{Sobolev}^{\uparrow}$ *source rows, the* $\mathsf{CompactEmbedding}^{\uparrow}$ *handoff of chapter 7166, and the compact-metric and total-bounded rows of chapter 129 and chapter 127. The Frechet–Kolmogorov packet may then consume the finite StreamName, RegSeqRat, completion-boundary, transport, replay, provenance, and local NameCert rows exposed by those siblings, but it does not replace the compact-embedding handoff with selected subsequences, open-cover compactness, quotient function equality, or ambient Real completeness.*

Proof. Start from the compact-embedding side. The $\mathsf{CompactEmbedding}^{\uparrow}$ chapter exposes a finite handoff from Sobolev source windows into compact and complete target rows, so a Sobolev-family read arrives with displayed source, target, readback, transport, replay, provenance, and naming data.

Use the Frechet–Kolmogorov compactness carrier and its sibling route. The rows $M, T, E, Q, S, R, H, A, P, N$ already require compact-metric source exposure, total-bounded finite nets, completion boundary, finite-window readback, rational comparison, transport, replay, provenance, and local naming before compactness-facing data are exported.

The forward link is therefore a dependency route, not a new compactness principle. The Frechet–Kolmogorov consumer reads the Sobolev and compact-embedding siblings as displayed rows and then reuses the compact-metric and total-bounded surfaces, leaving selected subsequences, open-cover compactness, quotient function equality, and ambient Real completeness outside the packet. □

Theorem 2874.15 (Frechet–Kolmogorov hub dependency index)**.** *The hub route for* FrechetKolmogorovCompactne *reads the* $\mathsf{CompactMetric}^{\uparrow}$ *source rows, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net rows, the Arzela–Ascoli finite-modulus rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback rows, the* $\mathsf{Real}^{\uparrow}$ *terminal boundary rows, and the*

MetricCompletion$^{\uparrow}$ *sibling rows before any Frechet–Kolmogorov compactness consumer can inspect the local compactness packet. Thus the top-level input boundary records the same dependency lattice that the child certificates expose: compact metric source, total-bounded budget, finite-modulus equicontinuity, regular rational comparison, Real boundary, completion handoff, transport, replay, provenance, and local* NameCert *data.*

Proof. Read the hub as an ordered certificate boundary. Its child files first expose the name certificate spine and then the compact-metric, total-bounded, Arzela–Ascoli, locally convex, and metric-completion routes.

The rational and Real rows are read through the same sibling lattice. The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ surfaces supply readback and terminal boundary data before the compactness consumer opens the Frechet–Kolmogorov packet.

Consequently a consumer cannot bypass the sibling rows at the hub boundary. The hub records the dependency lattice directly while leaving all theorem content in child files. □

The FrechetKolmogorovCompactness$^{\uparrow}$ dependency lattice reads compactness only after its sibling surfaces are visible: the compact metric source of chapter 129, the finite-family control supplied by chapter 412, the completion-facing transport of chapter 1506, the regular rational tolerance readback of chapter 301, the terminal real seal of chapter 38, and the tightness ledger of chapter 3795. This makes compactness a displayed route through compact metric, equicontinuity, completion, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and tightness rows, rather than an ambient subsequence selector, open-cover oracle, or quotient completion equality.

Theorem 2874.16 (Frechet–Kolmogorov compactness sibling route)**.** *The* FrechetKolmogorovCompactness$^{\uparrow}$ *packet consumes the Arzela–Ascoli finite-modulus route of chapter 3568 as its equicontinuity-facing compactness control, and it consumes the local seminorm-to-completion route recorded in Definition 3846.12 as its locally convex boundedness control. Together with the compact metric surface of chapter 129, the total-bounded surface of chapter 127, the equicontinuity radius source of chapter 3844, the compact metric total-bounded source of chapter 4055, the regular rational readback of chapter 301, and the Real seal of chapter 38, these sibling reads keep the compact metric, total-bounded, uniform-equicontinuity, locally convex seminorm, Frechet metric, RegSeqRat, and Real-seal rows visible as one finite dependency lattice rather than as an ambient function-space compactness theorem.*

Proof. Read the compact metric and total-bounded rows first. The compactness-facing consumer must pass through the displayed compact metric packet and its total-bounded control before any finite-cover or subsequence conclusion is available.

Next read the equicontinuity and local boundedness rows. The Arzela–Ascoli finite-modulus route supplies the uniform-equicontinuity control, while the locally convex boundedness route supplies the seminorm-to-completion control.

Finally read the completion, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ seal rows. Since each consumer opens these sibling rows before the compactness conclusion, the packet exports a finite dependency lattice rather than an ambient function-space compactness principle. □

Theorem 2874.17 (Frechet–Kolmogorov bounded-tail compactness route)**.** *An accepted* FrechetKolmogorovCompa *consumer may read total-bounded compactness only after it has exposed both the Arzela–Ascoli finite-modulus route and the locally convex boundedness route. The read passes through* CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and the displayed completion boundary of Definition 3846.6; it does not use open-cover compactness, selected subsequences, or an ambient choice of compact limit points.*

Proof. Start with the sibling route of Definition 3846.16. It orders the compact metric, total-bounded, Arzela–Ascoli finite-modulus, locally convex seminorm, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as one dependency lattice.

Next apply the locally convex boundedness route of Definition 3846.12. It supplies the bounded finite-tail control needed before the Frechet metric can be read through the completion surface.

Finally combine the total-bounded handoff of Definition 3846.3 with the completion boundary of Definition 3846.6. The handoff gives finite total-bounded data, and the boundary keeps the completion read in $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ form, so no open-cover compactness principle or selected subsequence is introduced. □

Theorem 2874.18 (Frechet–Kolmogorov sibling dependency witness)**.** *Every accepted* FrechetKolmogorovCompact *carrier witnesses a substantive dependency on the compact metric, total-bounded, and Arzela–Ascoli finite-modulus sibling chapters. The compactness criterion may be consumed only after the compact source of chapter 129, the finite-net surface of chapter 127, and the Arzela–Ascoli finite-modulus route of chapter 3568 have all been displayed together with the local carrier rows of Definition 3846.1.*

Proof. Read the compact source first. The row M in Definition 3846.1 is a $\mathsf{CompactMetric}^{\uparrow}$ source row, so the criterion has no compactness-facing read until the displayed compact metric packet is available.

Read the total-bounded surface next. The row T supplies the finite-net budget consumed by the criterion, and the theorem Definition 3846.3 forces that budget to pass through the same compact metric source before any readback rows are inspected.

Read the Arzela–Ascoli finite-modulus route as the equicontinuity-facing control. The sibling route of Definition 3846.16 binds the Arzela–Ascoli finite-modulus chapter to the same finite compactness route, so the equicontinuity row cannot be replaced by an ambient function-space compactness theorem.

Finally the local transport, replay, provenance, and naming rows H, A, P, N keep those sibling dependencies on the same packet. Thus the Frechet–Kolmogorov chapter records a dependency lattice over compact metric, total boundedness, and Arzela–Ascoli finite modulus rather than a kernel-only compactness assertion. □

Theorem 2874.19 (Frechet–Kolmogorov orphan compact metric lattice)**.** *The compactness-facing export of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *packet is not a kernel-only assertion. Before it exposes the local compactness criterion, the consumer must read the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *control row, the Arzela–Ascoli finite-modulus route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row as sibling dependencies of the same carrier.*

Proof. Start with the dependency witness of Definition 3846.18. It already forces the compact metric, total-bounded, and Arzela–Ascoli finite-modulus rows to be displayed with the local carrier rows.

Now read the sibling route of Definition 3846.16. That route places the uniform-equicontinuity, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows in the same finite lattice rather than letting the criterion stand as an isolated kernel compactness claim.

Finally apply the total-bounded handoff of Definition 3846.3. The finite-net budget must pass through the compact metric source before readback rows are inspected, so the local compactness criterion is exposed only after the compact-metric dependency lattice is visible. □

The FrechetKolmogorovCompactness$^{\uparrow}$ packet consumes the compact-metric total-bounded route from chapter 129 and the finite extraction route of chapter 412 before it can export a compactness criterion. Thus the compact metric source, finite extraction surface, transport rows, replay rows, and local NameCert data are visible as horizontal dependencies rather than as an isolated compactness kernel. The FrechetKolmogorovCompactness$^{\uparrow}$ packet consumes the finite total-bounded and completion rows of chapter 129 before it exports compactness data to the equicontinuity and finite-subfamily surface of chapter 412. This route keeps compact metric source data, finite nets, completion readiness, Arzela–Ascoli family control, transport, replay, provenance, and local NameCert rows visible together, rather than treating Frechet–Kolmogorov compactness as an isolated compactness oracle.

Theorem 2874.20 (Frechet–Kolmogorov top sibling pointer)**.** *Every finite-window criterion exported by* FrechetKolmogorovCompactness$^{\uparrow}$ *must first display the compact metric source of chapter 129, the total-bounded finite-net surface of chapter 127, and the Arzela–Ascoli finite-modulus route of chapter 3568. The top-level route therefore exposes the sibling lattice before any Frechet–Kolmogorov compactness consumer reads the local carrier.*

Theorem 2874.21 (Frechet–Kolmogorov tightness dependency route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read must expose the compact metric source of chapter 129, the total-bounded finite-net surface of chapter 127, the Arzela–Ascoli finite-modulus route of chapter 3568, and the locally-convex boundedness and tightness rows before the local carrier is consumed. The dependency route therefore records Frechet–Kolmogorov compactness as a finite lattice read over compact metric, total boundedness, equicontinuity-modulus, and tightness data, not as an ambient compactness oracle.*

Theorem 2874.22 (Frechet–Kolmogorov tightness route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read must display the* CompactMetric$^{\uparrow}$ *source row, the* TotallyBounded$^{\uparrow}$ *finite-net row, the* UniformEquicontinuityModulus$^{\uparrow}$ *control row, the* CompleteMetric$^{\uparrow}$ *boundary row, the* RegSeqRat$^{\uparrow}$ *rational readback row, the* Real$^{\uparrow}$ *seal, and the local tightness and boundedness rows before it exports compactness of the family. The route is a finite* BHist/Cont/Pkg *ledger over compact metric, total-bounded, equicontinuity, completion, regular-rational, real, tightness, boundedness, transport, replay, provenance, and local* NameCert *data; it is not an ambient compactness oracle, a selected subsequence, a selected subcover, or quotient completion equality.*

Proof. Read the carrier of Definition 3846.1. Its source rows already expose CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, UniformEquicontinuityModulus$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ as the visible compactness route.

Now combine the total-bounded handoff of Definition 3846.3 with the sibling route of Definition 3846.4. Those results force the finite-net, equicontinuity, and completion-facing rows to remain visible before any compactness export is read.

The tightness and boundedness rows are therefore local dependency rows in the same ledger. They do not introduce a hidden subsequence choice, a hidden subcover, or an equality in a quotient completion, because the structural rows transport and replay only the displayed finite packet. □

The compactness export of FrechetKolmogorovCompactness$^{\uparrow}$ is read through a compact-metric finite-net source together with the Arzela–Ascoli equicontinuity and tightness rows. The compact metric source fixes the finite-net budget; the Arzela–Ascoli sibling supplies the function-family tightness surface; the local replay rows then expose compactness only after both pieces have been displayed.

Theorem 2874.23 (Frechet–Kolmogorov compact-metric tightness route)**.** *Every accepted* FrechetKolmogorovCom *consumer that exports compactness for function-family windows must first read the compact-metric finite-net data of chapter 129 and the equicontinuity/tightness rows of chapter 412. The export is therefore a finite dependency route through compact metric source rows, finite-net coverage rows, Arzela–Ascoli tightness rows, transport, replay, provenance, and local* NameCert *data, not a selected subsequence, open-cover compactness theorem, quotient function-family equality, or ambient completeness principle.*

Proof. Project the Frechet–Kolmogorov carrier through its compactness-facing rows. The compact metric source supplies the finite-net data that any downstream compactness read must inspect before it can speak about a family window.

Next read the Arzela–Ascoli sibling route. Its equicontinuity and tightness rows are consumed after the compact-metric finite nets and before the local compactness export, so the function-family conclusion remains tied to displayed finite data.

Finally project the structural rows. Componentwise hsame transport, displayed Cont replay, provenance, and local naming preserve exactly the compact-metric and tightness route, leaving no coordinate for a hidden subsequence, open-cover theorem, quotient equality of families, or ambient completeness principle. □

Theorem 2874.24 (Frechet–Kolmogorov Lp-tail dependency route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *consumer that reads compactness through Lp tails must first pass through the displayed sibling route*

CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, MetricCompletion$^{\uparrow}$, LpSpace/LpTail, BHist, hsame, Cont, Pkg, N

The exported compactness row is therefore a finite compact-metric, total-bounded, completion-facing, and Lp-tail dependency route, not an ambient compactness oracle, a selected subsequence, a selected subcover, or quotient completion equality.

Proof. Project the accepted carrier of Definition 3846.1. Its compactness-facing read already begins with the compact metric source and the finite-net budget, so any Lp-tail consumer must inherit those rows before reading a tail estimate.

Apply the total-bounded handoff of Definition 3846.3. It keeps the finite-net row visible before the StreamName, RegSeqRat, and Real readbacks are inspected.

Now use the completion boundary and the existing sibling route of Definition 3846.4. The Lp-space and Lp-tail rows can only be attached after the displayed MetricCompletion$^{\uparrow}$ handoff has been routed through the same finite BHist/Cont/Pkg ledger.

Route exhaustion gives the stated dependency chain. No row in the carrier supplies an unrecorded compactness principle, selected subsequence, selected subcover, or quotient-completion equality, so the Lp-tail compactness export is exactly the displayed sibling route. □

Theorem 2874.25 (Frechet–Kolmogorov Lp-tail localization)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read that localizes through an Lp-tail estimate consumes that estimate only after the compact-metric total-bounded route, tightness route, and Lp-tail dependency route have been displayed. Equivalently, the localized read factors through Definition 3846.11, Definition 3846.22, and Definition 3846.24; it does not introduce an ambient compactness oracle, selected subsequence, selected subcover, quotient completion equality, or hidden Lp-tail selector.*

Proof. Start with the compact-metric total-bounded route. It keeps the compact metric source and finite-net surface visible before a compactness-facing consumer may inspect downstream rows.

Next use the tightness route. Tightness and boundedness remain local dependency rows in the same finite ledger, so a tail estimate cannot bypass the displayed $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, completion, readback, transport, replay, provenance, and local naming route.

Finally apply the Lp-tail dependency route. It attaches the $\mathsf{LpSpace/LpTail}$ row only after the compact-metric, total-bounded, and completion-facing rows have been routed through the same finite $\mathsf{BHist/Cont/Pkg}$ ledger. Thus Lp-tail localization is a local factorization through the existing sibling routes, not a new compactness principle or choice-selected tail object. □

The operator-tail compactness route for $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ is read only after a bounded-linear-operator source from chapter 1644, an equicontinuous-sequence surface from chapter 4356, and the existing compact-metric, total-bounded, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route have been displayed. A consumer that starts from operator-tail compactness therefore reaches the Frechet–Kolmogorov packet through bounded operator rows, equicontinuity rows, compact-metric finite nets, total-bounded coverage, regular-rational readback, and the terminal Real seal, not through an ambient compactness theorem or a selected subsequence.

Theorem 2874.26 (Frechet–Kolmogorov upstream dependency lattice)**.** *Every* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *compactness read must pass through the compact metric, totally bounded,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, Arzela–Ascoli finite-modulus, and Lp-tail localization rows before any compactness conclusion is exported. The exported route is therefore a finite dependency lattice over the displayed carrier rows, not an ambient compactness oracle, selected subsequence, selected subcover, quotient completion equality, or host completeness principle.*

Proof. Start with the local carrier and the compact metric total-bounded dependency of Definition 3846.11. It forces the compact metric source and total-bounded finite-net budget to appear before later StreamName, RegSeqRat, and Real readbacks are inspected.

Next read the Arzela–Ascoli side through Definition 3846.8 and Definition 3846.9. Those routes keep finite-modulus and finite-window function-family data downstream of the compact metric and total-bounded rows.

Now apply the completion and rational-readback routes, Definition 3846.13 and Definition 3846.16. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows attached to the same displayed finite lattice.

Finally use the Lp-tail routes, Definition 3846.24 and Definition 3846.25. These place tail localization after compact-metric, total-bounded, completion-facing, and rational-readback rows. The stated lattice follows because no cited carrier row supplies a hidden compactness principle or selected limiting object. □

Chapter 2875

A Concrete Naming Certificate for $\mathsf{RationalBracketing}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalBracketing}^{\uparrow}$ packet records the finite rational lower-and-upper bracket surface used before a real-name consumer reads an approximation. It sits after the rational density surface of chapter 1409, the Archimedean bound surface of chapter 3372, the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed bracket endpoints, tolerance rows, finite observation windows, nesting ledger, and downstream regular-Cauchy handoff; it is not a trichotomy principle, quotient real equality theorem, limit selector, or ambient completeness assertion.

Definition 2875.1 (Rational bracketing carrier)**.** *A* $\mathsf{RationalBracketing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, D, S, R, E, N, H, C, P, A)$$

with the following displayed rows:

1. *L is the rational lower-endpoint row supplied through the* $\mathsf{RationalDensityReal}^{\uparrow}$ *surface.*
2. *U is the rational upper-endpoint row supplied through the same rational-density surface.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the displayed bracket width with the requested precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observations used to inspect L, U, and D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row exported by the bracket window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, D, S, R have been displayed.*
7. *N is the nesting ledger recording that later bracket windows refine earlier ones by moving the lower row upward, moving the upper row downward, or preserving an endpoint already inside the requested tolerance.*
8. *H records componentwise* hsame *transport for lower endpoint, upper endpoint, dyadic tolerance, stream window, regular readback, real seal, nesting, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bracketing request through $L, U, D, S, R, E, N, H, P, A$.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalBracketing}^{\uparrow}$, $\mathsf{RationalDensityReal}^{\uparrow}$, $\mathsf{RealArchimedeanProperty}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *A is the local* $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $L, U, D, S, R, E, N, H, C, P, A$. It has no coordinate for trichotomy, quotient real equality, selected limits, hidden real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalBracketingUp.RationalBracketingTasteGate single carrier alignment`

Theorem 2875.2 (Rational bracketing NameCert obligations)**.** *For every accepted* $\mathsf{RationalBracketing}^{\uparrow}$ *carrier $B = (L, U, D, S, R, E, N, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$ *surface consists exactly of lower-endpoint exposure through L, upper-endpoint exposure through U, dyadic width and tolerance exactness through D, finite-window exposure through S, regular rational readback through R, terminal real-name sealing through E, nesting-ledger visibility through N, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through A.*

Proof. Project the carrier of Definition 3847.1. Its visible approximation data are exactly the endpoint rows L, U, the tolerance row D, the finite-window row S, the readback row R, and the real seal row E.

The endpoint and tolerance rows are separate. The rational-density surface supplies L and U, while the Archimedean and dyadic rows supply the bounded comparison budget used to certify that the bracket width is below the requested tolerance.

The nesting ledger is downstream of those rows. It records refinement between displayed windows, but it does not add a new real representative, a trichotomy split, or a selected limiting endpoint.

Finally H, C, P, A transport, replay, source, and name the same finite surface. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2875.3 (Rational bracketing window nesting)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalBracketing}^{\uparrow}$ *packet factors through a finite nested-bracket route*

$$\mathsf{RationalDensityReal}^{\uparrow} \longmapsto \mathsf{RealArchimedeanProperty}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Re}$$

For any later accepted window, the consumer must read the same lower endpoint, upper endpoint, dyadic tolerance, stream observation, regular readback, real seal, nesting, transport, replay, provenance, and local naming rows before using the narrowed bracket. The route exports nested rational brackets, not trichotomy, quotient real equality, selected limits, or ambient completeness.

Proof. Begin with an accepted carrier $B = (L, U, D, S, R, E, N, H, C, P, A)$. A bracketing request first reads the rational lower and upper endpoint rows L, U together with the Archimedean bound information named in the provenance row.

Next the request reads D and S. The dyadic ledger specifies the precision budget, while the stream row records the finite observations used to inspect the current bracket.

The regular rational readback R and real seal E are available only after those rows have been displayed. A later window can be consumed only through the nesting ledger N, which compares the later endpoints with the earlier displayed endpoints under the same dyadic budget discipline.

Thus every downstream read remains inside the finite route named in the theorem. No step introduces trichotomy, quotient real equality, a selected limiting endpoint, countable choice, hidden real representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{RationalBracketing}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without first reading rational lower and upper endpoint rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{StreamName}^{\uparrow}$ finite window, nesting evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence is outside the packet.

Independence witness: rational-density-real, real-archimedean-property, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalBracketing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rational lower and upper endpoint rows, dyadic width budgets, finite observation windows, regular readback, terminal Real sealing, nesting, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBracketing}^{\uparrow}$ packet is generated from rational lower and upper endpoint rows, an Archimedean boundedness context, a DyadicRatCore tolerance ledger, StreamName finite windows, a RegSeqRat readback row, a RealUp seal, a nesting ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3847.1, Definition 3847.2, and Definition 3847.3 seed the finite rational-bracket route from density and Archimedean rows to regular rational readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, trichotomy, quotient real equality, selected limits, hidden real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked lower and upper endpoint exposure, dyadic width exactness, finite-window exposure, nesting stability, regular readback, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.RationalBracketingUp.

Chapter 2876

A Concrete Naming Certificate for $\mathsf{RealModulusExtraction}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy data, the $\mathsf{RealModulusExtraction}^{\uparrow}$ packet records the finite BEDC surface by which an explicit regular Cauchy modulus is exposed before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits on the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names modulus extraction as displayed finite NameCert data; it does not import quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2876.1 (RealModulusExtraction carrier)**.** *A* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius, denominator budget, and comparison ledger for each requested precision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite windows inspected before a modulus answer is emitted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row, recording regular rational Cauchy data exposed by the scheduled windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *T records componentwise* hsame *transport for the dyadic tolerance, schedule, regularity, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a precision request through D, then through S, R, and finally to the seal row E.*
7. *P is the* Pkg *provenance over* $\mathsf{RealModulusExtraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$ *row.*

The carrier has no coordinate for an implicit analytic modulus, quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealModulusExtractionUp.RealModulusExtractionTasteGate single carrier alignment`

Theorem 2876.2 (RealModulusExtraction NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier* $M = (D, S, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$ *surface consists exactly of dyadic tolerance exposure through* D*, finite schedule exposure through* S*, regular Cauchy witness exposure through* R*, terminal Real sealing through* E*, componentwise transport through* T*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier coordinates. The dyadic tolerance row D, schedule row S, regular witness row R, and seal row E are the analytic-facing rows of the packet.

The structural rows do not add analytic assumptions. Transport through T is componentwise, replay through C follows the displayed precision route, provenance through P records the sibling chapters named above, and N supplies the local NameCert row. Since no further coordinate is present, the listed rows exhaust the obligation surface. □

Theorem 2876.3 (RealModulusExtraction regular Cauchy handoff)**.** *Every regular Cauchy handoff exported by an accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The extracted modulus is therefore read as dyadic tolerance data, finite schedule data, and regular rational witness data before the Real seal is available.

Proof. Begin with a precision request. The carrier first exposes the dyadic tolerance row D, so the requested radius and denominator budget are displayed before any stream window is inspected.

The schedule row S then supplies the finite windows, and the regular witness row R reads the rational Cauchy data inside those windows. Only after this displayed route has been replayed by C does the seal row E become available to a $\mathsf{Real}^{\uparrow}$ consumer.

Thus the handoff is the finite sibling route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not an implicit analytic assumption. □

Theorem 2876.4 (RealModulusExtraction Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier is exhausted by the rows* D, S, R, E, T, C, P, N*. In particular, the read exports no quotient stream equality, selected representative, countable-choice schedule, host real equality, implicit analytic modulus, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Use the obligation surface of Definition 3848.2. A Real-facing consumer can inspect only the dyadic tolerance, finite schedule, regular witness, seal, transport, replay, provenance, and local naming rows.

The nonescape claim follows from the carrier boundary. None of the excluded objects appears as a coordinate of M, and the replay row C only preserves the displayed precision route through D, S, R, E. Hence every Real seal read remains inside the finite NameCert packet. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusExtraction}^{\uparrow}$ read exposes dyadic tolerance, finite schedule, regular Cauchy witness, Real seal, transport, replay, provenance, and local NameCert rows before any modulus answer is exported.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RealModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it carries the dyadic tolerance row, schedule row, regular

witness row, terminal seal row, componentwise transport, displayed replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusExtraction}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName schedule row, RegSeqRat regular witness row, RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3848.1, Definition 3848.2, Definition 3848.3, and Definition 3848.4 seed the finite modulus-extraction route from regular Cauchy data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic schedule coverage, regular Cauchy handoff, Real seal nonescape, and a Lean carrier formalization for BEDC.Derived.RealModulusExtractionUp.

Chapter 2877

A Concrete Naming Certificate for $\mathsf{CauchyStability}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy data, the $\mathsf{CauchyStability}^{\uparrow}$ packet names the finite stability route used when displayed transport changes a Cauchy presentation but not the Real-facing read. It sits on the StreamName windows of chapter 126, the dyadic tolerance core of chapter 291, the regular rational sequence route of chapter 301, the Cauchy equivalence boundary of chapter 1820, and the terminal real-name seal of chapter 38. The packet records stability of displayed Cauchy data; it is not quotient equality, selected representatives, countable choice over tails, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2877.1 (CauchyStability carrier)**.** *A* $\mathsf{CauchyStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are two regular Cauchy source rows whose rational observations are opened only through* $\mathsf{RegSeqRat}^{\uparrow}$ *and finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *M is the shared tail-modulus comparison row relating the two source rows at each displayed precision request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the transported source observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row recording the regular rational readback after A, B, M, D have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached through the displayed* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary.*
6. *H records componentwise* hsame *transport for the source, modulus, dyadic, regular-readback, equivalence, seal, replay, package, and naming rows.*
7. *C records displayed* Cont *replay from a Cauchy-stability request through $A, B, M, D, R, E, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyStability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient sequence equality, selected limits, hidden representatives, countable-choice tail selection, host real equality, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.CauchyStabilityUp.CauchyStabilityUp`
Lean4 ✓ `BEDC.Derived.CauchyStabilityUp.CauchyStabilityTasteGate single carrier alignment`

Theorem 2877.2 (CauchyStability NameCert obligations)**.** *For every accepted* $\mathsf{CauchyStability}^{\uparrow}$ *carrier* $S = (A, B, M, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy source rows, shared tail-modulus comparison, dyadic tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *stability of the transported readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming through the rows* $A, B, M, D, R, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 3849.1. The only source data are the rows A and B, and the only comparison data between them are the shared tail-modulus row M and dyadic tolerance ledger D.

The regular-rational readback R is downstream from those rows, so stability is read as a finite $\mathsf{RegSeqRat}^{\uparrow}$ handoff rather than as equality of host sequences. The displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ boundary is consumed before the terminal seal E can be read.

Transport and replay use H and C componentwise. The package and naming rows P, N source the same finite route, and the excluded quotient, selected-limit, hidden-representative, choice, host-equality, and completeness data have no carrier coordinate. □

Theorem 2877.3 (CauchyStability Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyStability}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the terminal seal E *is consumed. Thus displayed transport may change the finite Cauchy presentation only through the stored* $A, B, M, D, R, H, C, P, N$ *rows, and the Real-facing boundary receives no quotient equality, selected representative, hidden tail family, countable-choice selector, host real equality, or ambient completeness coordinate.*

Proof. Start with a Real-facing request. The carrier first exposes the finite StreamName windows attached to A and B, together with the shared modulus comparison M.

The dyadic ledger D translates those windows into displayed tolerance observations. The regular-rational handoff R then repacks the inspected observations as $\mathsf{RegSeqRat}^{\uparrow}$ data.

Only after this readback does the packet enter the displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ boundary and then the seal E. The rows H, C, P, N transport, replay, source, and name that same route, leaving no hidden Real-facing coordinate beyond the listed sibling surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyStability}^{\uparrow}$ packet is generated from two regular Cauchy source rows, a shared tail-modulus comparison row, dyadic tolerance rows, a RegSeqRat handoff, a CauchyEquivalenceSetoid boundary, a RealUp seal, displayed hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$, seed)

Scope closed: Definition 3849.1, Definition 3849.2, and Definition 3849.3 seed the displayed stability route for Cauchy data under transport before the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of Cauchy presentations, selected representatives, hidden tails, countable choice over moduli, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-modulus comparison, RegSeqRat stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyStabilityUp.

Chapter 2878

A Concrete Naming Certificate for $\mathsf{CauchySequenceTailFilter}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki–Kelley filterbase terminology, the $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ packet records the finite route by which a Bishop Cauchy sequence supplies tail-filter data for completion-facing consumers. It sits between the sequence-side evidence of $\mathsf{BishopCauchySequence}^{\uparrow}$, the filter carrier of chapter 333, the finite stream schedule of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet names tail neighborhoods generated by displayed sequence windows; it does not select a limit, quotient streams, choose subsequences, decide trichotomy, or assert ambient real completeness.

Definition 2878.1 (Cauchy sequence tail-filter carrier)**.** *A* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, B, F, C, D, Q, R, E, H, K, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are the only sequence observations visible to the carrier.*
2. *B is the* $\mathsf{BishopCauchySequence}^{\uparrow}$ *row certifying that the displayed sequence windows satisfy the Bishop Cauchy modulus requests read by the packet.*
3. *F is the Frechet-tail filterbase row generated by tails of S: a basic tail is named by a displayed index threshold together with the windows beyond that threshold.*
4. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row saying that every admitted tail-basic row is a Cauchy-filter neighborhood request over the same displayed windows and thresholds.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording dyadic scales, rational comparisons, and exhaustion data used to pass from sequence tails to filterbase requests.*
6. *Q is the finite tail-query ledger pairing each tolerance request with the stream windows and tail threshold used by B, F, C, D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for consumers that read the same tail evidence as regular rational names.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, B, F, C, D, Q, R have been displayed.*

9. *H stores componentwise* hsame *transport for the stream windows, Bishop Cauchy row, tail-filterbase row, Cauchy-filter compatibility row, dyadic tolerance ledger, tail-query ledger, regular handoff, Real seal, continuation replay, provenance, and local name.*
10. *K stores displayed* Cont *replay from each tolerance request through the stream window, tail threshold, filterbase row, Cauchy-filter compatibility row, regular handoff, and seal row.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BishopCauchySequence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}$ *row.*

The classifier accepts a packet only when every tail-basic row in F is generated from the displayed stream schedule S, every Cauchy-filter compatibility read in C uses the Bishop Cauchy evidence B and dyadic ledger D, and every completion-facing read factors through Q, R, E, H, K, P, N. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterUp.CauchySequenceTailFilterTasteGate single carrier alignment`

Theorem 2878.2 (Cauchy sequence tail-filter NameCert obligations)**.** *For every accepted* CauchySequenceTailFilter *carrier $T = (S, B, F, C, D, Q, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}$ *obligations are exactly carrier admission, finite stream-window classification, Bishop Cauchy tail coverage, Frechet-tail filterbase generation,* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility, dyadic tolerance-ledger exhaustion,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal nonescape,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local naming row. These obligations are discharged by the displayed route*

$$S \longmapsto B \longmapsto F \longmapsto C \longmapsto D \longmapsto Q \longmapsto R \longmapsto E$$

together with H, K, P, N.

The obligation surface exports no quotient stream, selected limit, choice principle, trichotomy row, open-cover compactness object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterUp.CauchySequenceTailFilterCarrier namecert obligations`

Proof. Project the carrier of Definition 3850.1. The only sequence observations are the finite $\mathsf{StreamName}^{\uparrow}$ windows in S, and the only Cauchy evidence for those windows is the Bishop row B.

Read the tail-filter rows next. The Frechet-tail row F names basic tails by displayed thresholds, while the compatibility row C checks those basic tails against the $\mathsf{CauchyFilter}^{\uparrow}$ surface. The dyadic ledger D and tail-query ledger Q account for the tolerance requests used in that check.

The regular and real-facing rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and $\mathsf{Real}^{\uparrow}$ seal E are available only after the stream, Bishop, tail-filterbase, Cauchy-filter, dyadic, and query rows have been displayed.

Finally apply the structural rows. Componentwise transport H, continuation replay K, provenance P, and the local name N preserve the same finite route, and the carrier definition supplies no coordinate for quotient streams, selected limits, choice, trichotomy, open-cover compactness, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ completion-facing read factors through the displayed $\mathsf{StreamName}^{\uparrow}$ windows, Bishop Cauchy row, Frechet-tail filterbase row,

CauchyFilter$^{\uparrow}$ compatibility row, dyadic tolerance ledger, RegSeqRat$^{\uparrow}$ handoff, and Real$^{\uparrow}$ seal before any tail consumer is allowed to read the packet.

Independence witness: cauchyfilter, streamname, regseqrat, dyadicratcore, real: CauchySequenceTailFilter$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries sequence-window evidence, Bishop Cauchy tail coverage, tail-filterbase generation, filter compatibility, dyadic tolerance accounting, regular-rational handoff, real-seal routing, transport, continuation replay, provenance, and local naming.

Closure status for NameCert$_{\text{CauchySequenceTailFilter}^{\uparrow}}$.

Constructive story: A CauchySequenceTailFilter$^{\uparrow}$ packet is generated from finite stream windows, a Bishop Cauchy sequence row, a Frechet-tail filterbase row, a CauchyFilter compatibility row, a dyadic tolerance ledger, a tail-query ledger, a RegSeqRat handoff, a Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceTailFilter}^{\uparrow}}$, seed)

Scope closed: Definition 3850.1 and Definition 3850.2 seed the finite sequence-to-tail-filter carrier from StreamName windows through Bishop Cauchy evidence, tail filterbase rows, CauchyFilter compatibility, dyadic tolerance accounting, RegSeqRat handoff, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceTailFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, choice principles, trichotomy, open-cover compactness, ambient Real$^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, tail-filterbase generation, CauchyFilter compatibility, dyadic ledger exhaustion, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local NameCert rows for BEDC.Derived.CauchySequenceTailFilterUp.

Chapter 2879

A Concrete Naming Certificate for CauchySequenceTailFilterCompletion$^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki–Kelley filter terminology, the CauchySequenceTailFilte packet records the finite bridge by which sequence-tail filters are read as Cauchy-filter completion data before any quotient completion point or selected limit is available. It sits after the sequence-tail route of chapter 3850, the completion-facing carrier of chapter 1569, and the saturated filter surface of chapter 333. The chapter names only the displayed finite handoff from tail-filter evidence to completion readback; it does not assert ambient real completeness, quotient-filter equality, selected representatives, countable choice, trichotomy, or host equality of completed points.

Definition 2879.1 (Cauchy sequence tail-filter completion carrier)**.** *A* CauchySequenceTailFilterCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, F, C, D, W, R, E, H, K, P, N)$$

with the following visible rows:

1. *T is the* CauchySequenceTailFilter$^{\uparrow}$ *row naming the sequence-generated tail-filter data from chapter 3850.*
2. *F is the* CauchyFilter$^{\uparrow}$ *row that reads the same tail-filter neighborhoods as saturated Cauchy-filter requests.*
3. *C is the* CauchyFilterCompletion$^{\uparrow}$ *row whose completion-facing consumer is allowed to inspect only the displayed filter route.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to compare the tail thresholds with completion precision requests.*
5. *W is the finite family of* StreamName$^{\uparrow}$ *windows exposed by the tail-filter row before the completion consumer reads R or E.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from the same windows, dyadic tolerances, and filter requests.*
7. *E is the terminal* Real$^{\uparrow}$ *seal reached only after T, F, C, D, W, R have been displayed.*
8. *H stores componentwise* hsame *transport for the tail-filter, Cauchy-filter, completion, dyadic, window, readback, seal, replay, provenance, and local naming rows.*

9. *K stores displayed* Cont *replay from a sequence-tail request through Cauchy-filter compatibility, completion handoff, finite windows, dyadic tolerance, regular readback, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$ *row.*

The classifier accepts X only when every completion-facing read first opens the sequence-tail row T, then the Cauchy-filter row F, then the completion row C, and only then the dyadic, window, readback, and Real-seal rows D, W, R, E. It has no coordinate for quotient filters, selected limits, countable-choice selectors, host real equality, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterCompletionUp.CauchySequenceTailFilterCompletionTasteGate`
`single carrier alignment`

Theorem 2879.2 (Cauchy sequence tail-filter completion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *carrier $X = (T, F, C, D, W, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$ *obligations are exactly carrier admission, sequence-tail source admission,* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff, dyadic tolerance exactness, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local naming row. These obligations are discharged by the displayed route*

$$T \longmapsto F \longmapsto C \longmapsto D, W \longmapsto R \longmapsto E$$

together with H, K, P, N.

The obligation surface exports no quotient filter, selected limit, host equality, countable-choice selector, trichotomy row, open-cover compactness object, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or detached completion coordinate.*

Proof. Project the carrier of Definition 3851.1. The public rows are exactly $T, F, C, D, W, R, E, H, K, P, N$, so carrier admission exposes a sequence-tail source, a Cauchy-filter compatibility row, a completion handoff, a finite tolerance and window route, a regular readback, a Real seal, and the structural rows.

Read the source and filter rows first. The row T supplies the finite tail-filter data generated from sequence windows, while F reads that data as the Cauchy-filter surface used by completion-facing consumers. The row C is downstream of those two rows and does not replace them by a quotient filter or selected limit.

Next inspect the completion readback route. The dyadic ledger D and window row W are the only precision and observation rows available before the regular readback R and the terminal seal E are consumed.

Finally apply the structural rows. Transport H, replay K, provenance P, and the local name N preserve the same route, and the carrier definition supplies no coordinate for quotient filters, selected limits, choice, trichotomy, open-cover compactness, host equality, or ambient completeness. □

Theorem 2879.3 (Cauchy sequence tail-filter completion Real handoff)**.** *Let $X = (T, F, C, D, W, R, E, H, K, P, N)$ be an accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *packet. Every Real-facing or completion-facing consumer that reads the terminal seal factors through*

$$T, F, C, D, W, R \longmapsto E \longmapsto H, K, P, N.$$

Thus the $\mathsf{Real}^{\uparrow}$ *seal is available only after the sequence-tail filter, Cauchy-filter compatibility, Cauchy-filter completion row, dyadic tolerance ledger, finite stream windows, and regular rational readback have been displayed.*

Proof. Start with a consumer of the terminal seal. Carrier projection gives access to E only as the seal row of the accepted packet X.

Replay the continuation row K. The request consumes the sequence-tail source T, the Cauchy-filter row F, the completion row C, the dyadic tolerance ledger D, the finite windows W, and the regular readback R before E is available.

The structural rows preserve this order. Transport through H, provenance through P, and naming through N record the same route, so the consumer has no available coordinate for a selected completion point, quotient-filter equality, host real equality, choice, or ambient completeness. □

Theorem 2879.4 (Cauchy sequence tail-filter completion filterbase nonescape)**.** *Let* $X = (T, F, C, D, W, R, E, H, K$ *be an accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *packet. No accepted completion read can bypass the displayed sequence-tail filter row* T *or the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *when it reaches the completion row* C*, the regular readback* R*, or the seal row* E*.*

Consequently every filterbase-dependent completion read is confined to the finite route

$$T_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}, F_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto C_{\mathsf{CauchyFilterCompletion}^{\uparrow}} \longmapsto D, W, R, E, H, K, P, N.$$

Detached filterbases, quotient filters, selected limits, free tolerance parameters, countable-choice selectors, and ambient completeness principles are outside the accepted packet.

Proof. Inspect the carrier of Definition 3851.1. The only source for filterbase-dependent data is T, and the only Cauchy-filter compatibility row is F.

Apply the obligation surface of Definition 3851.2. It requires the completion handoff C to read after T and F, then requires D, W, R, E to remain downstream of that handoff.

The Real handoff of Definition 3851.3 keeps terminal consumers on the same route. Since H, K, P, N only transport, replay, source, and name the displayed rows, no detached filterbase or excluded completion object can be introduced later. □

Falsifiable prediction: Every accepted $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ completion read opens the displayed $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ row, $\mathsf{CauchyFilter}^{\uparrow}$ row, $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row, dyadic tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal before any consumer may inspect the packet as completion data.

Independence witness: cauchy-sequence-tail-filter, cauchyfilter, cauchyfiltercompletion, dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries sequence-tail source data, Cauchy-filter compatibility, completion handoff, dyadic tolerance accounting, finite stream windows, regular readback, Real-seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ packet is generated from a CauchySequenceTailFilter row, a CauchyFilter row, a CauchyFilterCompletion row, dyadic tolerance data, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3851.1, Definition 3851.2, Definition 3851.3, and Definition 3851.4 seed the finite sequence-tail-filter bridge into CauchyFilterCompletion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchySequenceTailFilterCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, selected completion representatives, countable choice, trichotomy, open-cover compactness, host equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, sequence-tail source admission, CauchyFilter compatibility, CauchyFilterCompletion handoff, dyadic tolerance exactness, StreamName window exposure, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local NameCert rows for BEDC.Derived.CauchySequenceTailFilterCompletion

Chapter 2880

A Concrete Naming Certificate for $\mathsf{LocalizedCauchyRate}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocalizedCauchyRate}^{\uparrow}$ packet records localized convergence-rate evidence as finite BEDC rows before a Real or Completion consumer reads a regular Cauchy name. It sits over the tail-bound route of chapter 2071, the regular tail-bound route of chapter 3743, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the oscillation ledger of chapter 448. The packet is a localized rate ledger; it does not choose quotient stream representatives, selected limits, trichotomy decisions, ambient completeness, or global convergence-speed functions.

Definition 2880.1 (Localized Cauchy rate carrier)**.** *A* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, D, T, R, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected rational stream segment.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the local dyadic budgets used by the packet.*
3. *T is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row displaying the tail-bound witness for the requested local tolerance window.*
4. *R is the* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *row recording the regular tail-bound strengthening consumed by regular Cauchy readers.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, tying the window and tail-bound rows to regular rational sequence data.*
6. *Q is the* $\mathsf{CauchyOscillation}^{\uparrow}$ *row, recording the local oscillation bound checked against the same window and dyadic budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after W, D, T, R, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for windows, tolerances, tail bounds, regular readback, oscillation, real seal, replay, provenance, and naming.*
9. *C records displayed* Cont *replay from a localized rate request through $W, D, T, R, S, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocalizedCauchyRate}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyOscillation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, D, T, R, S, Q, E, H, C, P, N$. The carrier has no coordinate for a selected rate outside T and R, quotient stream equality, an external limit selector, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocalizedCauchyRateUp.LocalizedCauchyRateTasteGate single carrier alignment`

Theorem 2880.2 (Localized Cauchy rate NameCert obligations)**.** *For every accepted* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *carrier $L = (W, D, T, R, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $W, D, T, R, S, Q, E, H, C, P, N$;*
2. *window localization: every rate read starts with the finite* $\mathsf{StreamName}^{\uparrow}$ *row W and the dyadic tolerance ledger D;*
3. *tail-bound visibility: the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row T and* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *row R are displayed before any regular rational readback is consumed;*
4. *regular readback: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row S reads the same window, dyadic budget, and tail-bound rows;*
5. *oscillation compatibility: the* $\mathsf{CauchyOscillation}^{\uparrow}$ *row Q checks the local oscillation bound against the same budget and tail window;*
6. *real handoff: the* $\mathsf{Real}^{\uparrow}$ *seal E is reachable only after W, D, T, R, S, Q have been displayed;*
7. *ledger nonescape: H, C, P, N export no selected rate, quotient representative, host limit, trichotomy decision, countable-choice schedule, ambient completeness theorem, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3852.1. The accepted rows are exactly $W, D, T, R, S, Q, E, H, C, P, N$, so carrier admission exposes only finite windows, dyadic budgets, tail bounds, regular readback, oscillation, a real seal, and structural rows.

The localized rate route begins with W and D. These rows name the finite stream segment and the tolerance budget, so the tail-bound rows cannot be read as global convergence-speed functions.

Next read T and R. The ordinary Cauchy tail-bound row supplies the requested local tail witness, while the regular tail-bound row records the strengthening needed by a regular Cauchy consumer. The readback row S consumes those same displayed rows, and Q checks the matching oscillation bound.

Finally E, H, C, P, N seal and transport the route. Since none of these rows adds a selector, quotient representative, trichotomy decision, countable-choice schedule, ambient completeness theorem, or bridge certificate, the listed obligations exhaust the local surface. □

Theorem 2880.3 (Localized Cauchy rate Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *packet factors through*

$$W \longmapsto D \longmapsto T \longmapsto R \longmapsto S \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

In particular, a consumer preserving the $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *routes must read the localized tail-bound ledger before it can treat the stream as a real-facing regular Cauchy name.*

Proof. Begin with a Real-facing request. The finite stream row W and dyadic ledger D determine the local tolerance window that the request may inspect.

The tail-bound handoff then passes through T and R. The first row gives the Cauchy tail witness at the displayed tolerance, and the second row exposes the regular tail-bound strengthening required by regular rational readback.

The readback and oscillation rows are downstream of the same data. The row S reads the regular rational sequence through the displayed window and tail-bound rows, while Q confirms that the local oscillation bound uses the same budget.

Only after those rows have been displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so any Real or Completion consumer bypassing T and R lies outside the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{LocalizedCauchyRate}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ seal without displaying StreamName windows, DyadicRatCore budgets, CauchyTailBound and RegularCauchyTailBound rows, RegSeqRat readback, CauchyOscillation compatibility, transport, replay, provenance, and local naming.

Independence witness: cauchy-tail-bound, regular-cauchy-tail-bound, dyadicratcore, streamname, regseqrat, real, cauchy-oscillation: none alone is bijective with $\mathsf{LocalizedCauchyRate}^{\uparrow}$ because the packet jointly records localized windows, dyadic budgets, ordinary and regular tail-bound rows, regular rational readback, oscillation compatibility, terminal Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalizedCauchyRate}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore budgets, CauchyTailBound and RegularCauchyTailBound rows, RegSeqRat readback, CauchyOscillation compatibility, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3852.1, Definition 3852.2, and Definition 3852.3 seed the finite localized convergence-rate ledger consumed by Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected rates, quotient stream representatives, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked window localization, tail-bound visibility, regular readback, oscillation compatibility, Real handoff, ledger nonescape, and local naming for BEDC.Derived.LocalizedCauchyRateUp.

Chapter 2881

A Concrete Naming Certificate for $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$

Following Bishop–Bridges compact metric continuity, the $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ packet records the finite route from compact source data and pointwise continuity rows through finite oscillation partitions to a displayed uniform-modulus output. It sits over the compact-metric source of chapter 129, the metric carrier of chapter 57, the continuous-map surface of chapter 130, the finite oscillation partition row of chapter 3800, the oscillation modulus boundary of chapter 2224, and the uniform-modulus consumer of chapter 369. The packet names a compact-uniform handoff; it does not select an external modulus, identify host functions, choose points from a cover, assert quotient equality, or import ambient compactness.

Definition 2881.1 (Finite oscillation uniform-modulus carrier)**.** *A* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, A, B, O, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, carrying the compact-metric carrier, finite cover data, and metric distance rows inspected by the packet.*
2. *M is the metric compatibility row tying the compact source to the distance comparisons used by oscillation estimates.*
3. *A is the continuous-map pointwise row, exposing source, target, graph, and local continuity evidence for the displayed map.*
4. *B is the* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, listing the finite partition cells and their source-side cover witnesses.*
5. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row, recording the oscillation bound attached to each partition cell and requested tolerance.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row obtained by folding the finite oscillation bounds into one tolerance-only precision ledger.*
7. *H records componentwise* hsame *transport for compact, metric, continuous-map, partition, oscillation, uniform-modulus, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform-continuity request through $K, M, A, B, O, U, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, A, B, O, U, H, C, P, N$. The carrier has no coordinate for an external modulus selector, host function equality, point choice from a cover, quotient representation, ambient compactness theorem, or bridge certificate.

Lean4 (definition) `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusUp`
Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusTasteGate`
`single carrier alignment`

Theorem 2881.2 (Finite oscillation uniform-modulus NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ *carrier $F = (K, M, A, B, O, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformM}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $K, M, A, B, O, U, H, C, P, N$;*
2. *compact source exposure: K and M display the compact-metric source and distance comparisons used by the packet;*
3. *continuity visibility: the pointwise continuous-map row A is read before any oscillation cell is accepted;*
4. *finite partition exactness: B lists the finite oscillation partition cells and their cover witnesses;*
5. *oscillation-bound compatibility: O attaches each partition cell to the requested tolerance through the same metric and continuity rows;*
6. *uniform-modulus output: U is the only exported tolerance-only precision ledger and is obtained after B and O are displayed;*
7. *ledger nonescape: H, C, P, N export no external modulus selector, host function equality, point choice from a cover, quotient representation, ambient compactness theorem, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusNameCertObligatio`

Proof. Project the carrier in Definition 3853.1. Its accepted rows are exactly $K, M, A, B, O, U, H, C, P, N$, so carrier admission exposes compact source data, metric comparisons, continuous-map evidence, finite partition rows, oscillation bounds, a uniform-modulus output, and structural rows only.

The compact and continuity clauses are read first. The compact source row K and metric row M supply the finite cover and distance comparisons, while A supplies the pointwise continuity evidence used on each displayed cell.

The partition route then reads B and O. The finite oscillation partition row lists the cells, and the oscillation-modulus row binds each cell to the requested tolerance through the same compact, metric, and continuity data.

Finally the output row U records the uniform-modulus ledger obtained from the finite oscillation bounds. The transport, replay, provenance, and naming rows preserve that displayed route and contain no coordinate for the excluded selector, host equality, point choice, quotient representation, ambient compactness theorem, or bridge certificate. □

Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusCarrier`
`namecert obligations`

Theorem 2881.3 (Finite oscillation compact-uniform handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ *packet factors through*

$$K \longmapsto M \longmapsto A \longmapsto B \longmapsto O \longmapsto U \longmapsto H, C, P, N.$$

In particular, a consumer preserving compact-metric, continuous-map, finite oscillation partition, oscillation modulus, and uniform-modulus provenance must read the finite oscillation partition and its oscillation bounds before exporting the tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ *row.*

Proof. Begin with a compact-uniform request. The rows K and M display the compact source and metric comparisons, so the request is tied to finite source-side data rather than to an ambient compactness theorem.

The continuous-map row A supplies the pointwise continuity evidence used on partition cells. After that, B displays the finite oscillation partition, including the cells and cover witnesses read by the source-side request.

The oscillation row O binds those cells to the requested tolerance. Since O is downstream of K, M, A, B, it cannot be read as a detached modulus or host-level continuity theorem.

The uniform-modulus row U is then the only exported tolerance-only precision ledger. The structural rows H, C, P, N transport and name the same finite route, so any compact-uniform consumer reaching U without B and O bypasses the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying compact source rows, metric comparisons, continuous-map evidence, finite oscillation partitions, oscillation-modulus bounds, transport, replay, provenance, and local naming.

Independence witness: compactmetric, metricspace, continuousmap, finite-oscillation-partition, oscillation-modulus, uniformmodulus: none alone is bijective with $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ because the packet jointly carries compact source data, metric comparisons, pointwise continuity rows, finite partition cells, oscillation bounds, uniform-modulus output, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ packet is generated from compact-metric source rows, metric comparison rows, ContinuousMap pointwise rows, finite oscillation partition rows, oscillation-modulus bounds, a UniformModulus output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3853.1, Definition 3853.2, and Definition 3853.3 seed the finite oscillation route from compact local control to a uniform-modulus output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusCarrier\_namecert\_obl`

Bridge status: none

Not claimed: This chapter does not close external modulus selectors, host function equality, point choice from covers, quotient representations, ambient compactness, or bridge checking.

Upgrade path: Obligation closure requires checked compact source exposure, continuity visibility, finite partition exactness, oscillation-bound compatibility, uniform-modulus output, ledger nonescape, and local naming for BEDC.Derived.FiniteOscillationUnifo

Chapter 2882

A Concrete Naming Certificate for $\mathsf{FubiniFiniteRectangle}^{\uparrow}$

Following Fubini and Bishop-style finite integration, the $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ packet records a bounded finite-product rectangle computation as BEDC data before either iterated-integral route may be read. It sits over the measure row of chapter 92, the integral row of chapter 123, the product rows of chapter 2000, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the Real seal of chapter 38. The chapter names the finite rectangle surface only; it does not assert sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, or ambient measure completeness.

Definition 2882.1 (Fubini finite rectangle carrier)**.** *A* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, M, I, P, A, B, D, S, E, H, C, Q, N)$$

with the following displayed rows:

1. *R is the finite product-rectangle row, listing the source rectangles and their cell tags.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *row assigning the displayed finite measure data used by the cells of R.*
3. *I is the* $\mathsf{Integral}^{\uparrow}$ *row for cellwise finite integration over the same rectangle table.*
4. *P is the* ***??*** *row tying the two coordinate directions to the same finite rectangle cells.*
5. *A is the first-then-second iterated-integral route, reading the first coordinate cells before folding the second coordinate.*
6. *B is the second-then-first iterated-integral route, reading the second coordinate cells before folding the first coordinate.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the common finite error budget used by both iterated routes.*
8. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the displayed finite sums.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, M, I, P, A, B, D, S have been displayed.*
10. *H records componentwise* hsame *transport for rectangle cells, measure rows, integral rows, product rows, both iterated routes, tolerance ledgers, rational readback, Real seal, replay, provenance, and naming.*

11. *C records displayed* Cont *replay from a finite-rectangle integration request through* $R, M, I, P, A, B, D, S, E, H, Q,$
12. *Q is the* Pkg *provenance over* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, *??*, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, I, P, A, B, D, S, E, H, C, Q, N$. It has no coordinate for quotient function equality, selected representatives, countable choice, ambient completion of a measure space, or an infinite summability principle.

Lean4 ✓ `BEDC.Derived.FubiniFiniteRectangleUp.FubiniFiniteRectangleTasteGate single carrier alignment`

Theorem 2882.2 (Fubini finite rectangle NameCert obligations)**.** *For every accepted* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *carrier $F = (R, M, I, P, A, B, D, S, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $R, M, I, P, A, B, D, S, E, H, C, Q, N$;*
2. *rectangle visibility: R and P display the finite product cells before either iterated route is read;*
3. *measure and integral compatibility: M and I attach measure and integral rows to the same finite cell table;*
4. *route symmetry: A and B read the same cells in opposite coordinate order and share the tolerance row D;*
5. *rational readback: S reads the finite sums produced by both routes through regular rational data;*
6. *Real sealing: E is reached only after both iterated routes, the dyadic ledger, and regular rational readback have been displayed;*
7. *ledger nonescape: H, C, Q, N export no sigma-finite theorem, Tonelli theorem, quotient function equality, selected representative, countable-choice schedule, ambient measure completeness, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FubiniFiniteRectangleUp.FubiniFiniteRectangleNamecertObligations`

Proof. Project the carrier of Definition 3854.1. The accepted rows are exactly $R, M, I, P, A, B, D, S, E, H, C, Q, N$, so carrier admission exposes a finite rectangle table, its measure and integral rows, two displayed iterated routes, a dyadic ledger, rational readback, a Real seal, and structural rows.

The finite rectangle route starts with R and P. These rows fix the cell table and the coordinate-product reading, so neither iterated route may introduce a new domain or a hidden product object.

Next read M, I, A, B, D, S. The measure and integral rows are attached to the same finite cells. The two iterated routes fold those cells in opposite coordinate order, while the dyadic ledger and regular rational readback force both routes to use the same finite tolerance and sum data.

Finally E, H, C, Q, N seal and transport the same displayed computation. Since none of these rows adds sigma-finite measure structure, Tonelli data, quotient function equality, selected representatives, countable choice, ambient measure completeness, or a bridge certificate, the listed obligations exhaust the local surface. □

Theorem 2882.3 (Fubini finite rectangle iterated-integral nonescape)**.** *Every iterated-integral consumer of an accepted* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *packet factors through*

$$R \longmapsto M \longmapsto I \longmapsto P \longmapsto A, B \longmapsto D \longmapsto S \longmapsto E \longmapsto H, C, Q, N.$$

Thus a consumer may cite equality of the two finite rectangle reads only through the displayed cells, measure rows, integral rows, product row, both iterated routes, dyadic tolerance ledger, regular rational readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It obtains no sigma-finite Fubini theorem, Tonelli theorem, quotient function equality, selected representative, countable-choice schedule, or ambient measure-completeness principle.*

Proof. Begin with an iterated-integral request. The finite rectangle row R and product row P identify the only cells and coordinate directions available to the request.

The measure and integral rows M and I then bind the computation to that same table. The first-then-second route A and second-then-first route B are therefore two readings of the displayed finite cell data rather than appeals to an ambient product-measure theorem.

The dyadic ledger D and regular rational readback S force both route totals through the same finite tolerance and sum surface. Only after those rows have been displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed.

The structural rows H, C, Q, N transport and name the same finite route. Since the packet contains no row for sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, or ambient measure completeness, no iterated-integral consumer can obtain such data from the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$.

Constructive story: A $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ packet is generated from finite product rectangle cells, displayed Measure and Integral rows, a Product row, first-then-second and second-then-first iterated routes, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3854.1, Definition 3854.2, and Definition 3854.3 seed the finite rectangle iterated-integral packet over displayed product cells and common tolerance readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, ambient measure completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rectangle visibility, measure-integral compatibility, route symmetry, regular rational readback, Real sealing, ledger nonescape, and local naming for BEDC.Derived.FubiniFiniteRectangleUp.

Chapter 2883

A Concrete Naming Certificate for $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$

The $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ packet realizes the disk-dynamics fixed-end reading of section 9463.5 in chapter 9463 as finite disk metric, boundary, flow, and provenance rows. It sits after the metric-read surface of chapter 3474, the boundary-geodesic packet of chapter 3491, the finite routed-flow carrier of chapter 195, the disk-coordinate surface of chapter 377, and the disk phase-transport route of chapter 3455. The packet names a finite geodesic-flow surface; it does not complete the hyperbolic disk, quotient ideal endpoints, choose a global geodesic representative, or identify observer expansion with a physical spacetime law.

Definition 2883.1 (Hyperbolic geodesic-flow carrier)**.** *A* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (U, M, B, T, D, F, J, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source and target observations inspected by the flow.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary row, recording the finite far-end incidence used by the route.*
4. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *transport row carrying reversible disk-surface movement and fiber provenance.*
5. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *routed-flow row, displaying the finite step relation read as geodesic-flow evolution.*
6. *F is the fixed-end row stating that the boundary far-end commitment is preserved along the displayed route.*
7. *J is the classifier-stability row for disk coordinates, metric reads, boundary incidence, phase transport, routed flow, and fixed-end data.*
8. *A is the audit row tying U, M, B, T, D, F, J to the hyperbolic observer-expansion vision and to its sibling concrete packets.*
9. *H records componentwise* hsame *transport for disk-coordinate, metric, boundary, phase-transport, routed-flow, fixed-end, classifier-stability, audit, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a geodesic-flow request through* $U, M, B, T, D, F, J, A, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, B, T, D, F, J, A, H, C, P, N$. *The carrier has no coordinate for a completed hyperbolic disk, quotient ideal-boundary equality, a selected global geodesic, a host analytic flow, physical cosmology, or observer substance.*

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicFlowUp.HyperbolicGeodesicFlowTasteGate single carrier alignment`

Theorem 2883.2 (Hyperbolic geodesic-flow NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *carrier* $G = (U, M, B, T, D, F, J, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet* $U, M, B, T, D, F, J, A, H, C, P, N$;
2. *disk and metric visibility:* U *and* M *display disk coordinates and hyperbolic metric reads before any flow row is exported;*
3. *boundary incidence:* B *supplies the finite far-end data consumed by the geodesic route;*
4. *phase and flow routing:* T *and* D *display the reversible phase-transport route and finite routed-flow relation;*
5. *fixed-end stability:* F *and* J *preserve boundary far-end and classifier readings along the displayed route;*
6. *audit provenance:* A, H, C, P, N *source, transport, replay, package, and name the same finite rows;*
7. *ledger nonescape: no row exports a completed disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, observer substance, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicFlowUp.HyperbolicGeodesicFlowNamecertObligations`

Proof. Project the carrier of Definition 3856.1. The accepted rows are exactly $U, M, B, T, D, F, J, A, H, C, P, N$, so carrier admission exposes finite disk observations, metric reads, boundary incidence, transport, routed flow, fixed-end data, classifier stability, audit provenance, and structural rows only.

The route begins with U and M. These rows bind the computation to displayed disk and metric data, so the flow cannot import a completed hyperbolic plane or an ambient analytic model.

Next read B, T, D, F, J. Boundary incidence supplies the finite far-end, phase transport and routed flow display the movement, and fixed-end plus classifier-stability rows keep the same boundary commitment through the route.

Finally A, H, C, P, N audit, transport, replay, source, and name the finite packet. Since none of these rows adds quotient ideal-boundary equality, selected global geodesics, host analytic flow, physical cosmology, observer substance, or a bridge certificate, the listed obligations exhaust the local surface. □

Theorem 2883.3 (Hyperbolic geodesic-flow fixed-end nonescape)**.** *Every geodesic-flow consumer of an accepted* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto \mathsf{DynSyster}$$

Thus a consumer may read disk coordinates, hyperbolic metric data, finite boundary incidence, phase transport, routed flow, fixed-end stability, classifier preservation, audit provenance, transport, replay, package sourcing, and local naming. It obtains no completed hyperbolic disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance.

Proof. Begin with a geodesic-flow request. The disk-coordinate row U and metric-read row M give the only geometric data available to the request.

The boundary row B supplies finite far-end incidence. The transport row T and routed-flow row D then display movement through certified phase transport and finite flow data, rather than through a host geodesic-flow theorem.

The fixed-end and classifier rows F, J preserve the boundary commitment and the sibling classifiers along the displayed route. The audit and structural rows A, H, C, P, N preserve that same finite packet.

No carrier coordinate names a completed hyperbolic disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance. Therefore a geodesic-flow consumer cannot escape the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ read is refuted if a geodesic-flow consumer obtains a completed disk, quotient ideal-boundary equality, a selected global geodesic, or a host analytic flow without displaying UnitDisk coordinates, HyperbolicMetric reads, HyperbolicGeodesicBoundary incidence, PoincareDiskPhaseTransport routing, DynSystem flow rows, fixed-end stability, transport, replay, provenance, and local naming.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-boundary, dynsystem, unitdisk, poincare-disk-phase-transport: none is bijective to $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ because the packet jointly carries disk observations, metric reads, boundary incidence, reversible phase transport, finite routed flow, fixed-end stability, classifier preservation, audit provenance, transport, replay, package sourcing, and local NameCert rows.

The Lean TasteGate for this packet will require explicit $\mathsf{Nontrivial}\ \mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ instances. Those instances must keep $U, M, B, T, D, F, J, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric reads, HyperbolicGeodesicBoundary incidence, PoincareDiskPhaseTransport rows, DynSystem routed-flow rows, fixed-end stability, classifier preservation, audit provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3856.1, Definition 3856.2, and Definition 3856.3 seed the finite geodesic-flow packet over disk metric, boundary, flow, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic disks, quotient ideal-boundary equality, selected global geodesics, host analytic flows, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk visibility, metric-read visibility, boundary incidence, phase-flow routing, fixed-end stability, TasteGate nontriviality, field faithfulness, ledger nonescape, and local naming for BEDC.Derived.HyperbolicGeodesic

Chapter 2884

A Concrete Naming Certificate for $\mathsf{LocatedSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedSequence}^{\uparrow}$ packet records the finite surface by which a sequence is read through located windows before it is consumed as regular rational, located real, or real-facing data. It sits after the finite stream observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located-real boundary of **??**, and the terminal real-name seal of chapter 38. The chapter isolates the generic located-sequence carrier used by later completion and compact-uniform-continuity routes; it does not choose a quotient stream, a selected limit, a trichotomy decision, countable choice, or ambient real completeness.

Definition 2884.1 (Located sequence carrier)**.** *A* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, D, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the finite sequence windows that may be inspected by a consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the inspected stream windows as regular rational sequence data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used when located windows are compared, tightened, or handed to a modulus consumer.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *window row, recording the located interval or ball evidence available at the displayed sequence indices.*
5. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular rational, dyadic, and located-real rows have been projected.*
6. *H records componentwise* hsame *transport for stream windows, regular rational readback, dyadic tolerances, located-real windows, real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite sequence request through*

$$S, D, L \longmapsto R \longmapsto Q.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocatedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the displayed stream windows, dyadic tolerance budgets, located-real windows, regular rational readback, and real seal describe one finite route. It has no coordinate for quotient streams, selected representatives, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedSequenceUp.TasteGate.LocatedSequenceTasteGate single carrier alignment`

Theorem 2884.2 (Located sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier* $A = (S, R, D, L, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$ *surface consists exactly of stream-window admission, regular rational readback, dyadic tolerance accounting, located-real window coverage, real-seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows* $S, R, D, L, Q, H, C, P, N$*. Consumers from* $\mathsf{LocatedSequenceModulus}^{\uparrow}$, $\mathsf{LocatedSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *may read the packet only through those displayed rows.*

Proof. Project the carrier of Definition 3857.1. The sequence is visible only through S, and every tolerance-facing comparison is visible only through the dyadic row D and the located-real row L.

The readback row R repacks the same inspected windows as $\mathsf{RegSeqRat}^{\uparrow}$ data. The real seal Q is downstream of S, D, L, R, so a real-facing consumer reaches it only after replaying the displayed located route.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier has no coordinate for quotient streams, selected representatives, host equality, trichotomy, countable choice, or ambient completeness, the listed NameCert rows are exhaustive. □

Theorem 2884.3 (Located sequence regular rational readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier must read*

$$S, D, L \longmapsto R \longmapsto Q \longmapsto H, C, P, N.$$

Thus regular rational readback is a displayed finite route from stream windows and located dyadic tolerances to the real seal, not a selected quotient stream, not host real equality, not a trichotomy principle, and not ambient real completeness.

Proof. Begin with a consumer request for regular rational or real-facing data. The packet exposes sequence observations only through S, and each located precision request is accounted for by D and L.

The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback surface. Therefore a located-real or real-facing consumer reaches the terminal seal Q only after the finite route through S, D, L, R has been displayed.

The structural rows H, C, P, N preserve that route by transport, replay, source, and local naming. No excluded analytic object is present in the tuple, so no downstream consumer can extract one from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedSequence}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, LocatedReal windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocatedSequence}\uparrow}$, seed)

Scope closed: Definition 3857.1, Definition 3857.2, and Definition 3857.3 seed the finite located-sequence surface for the L10 StreamName-to-RegSeqRat-to-Real route.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedSequence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window coverage, dyadic tolerance accounting, located-real window stability, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSequenceUp.

Chapter 2885

A Concrete Naming Certificate for $\mathsf{CauchyModulusExtractor}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusExtractor}^{\uparrow}$ packet records the finite operation that reads an accepted effective Cauchy sequence and exposes the explicit modulus row required by regular-Cauchy and real-completion consumers. It sits over the source certificate of chapter 1705, the fast-threshold selector of chapter 1781, the ordinary modulus schedule of chapter 292, the finite-window service of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet is a displayed extraction route from explicit Cauchy evidence to a Real-facing handoff; it is not quotient stream equality, a host-selected limit, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 2885.1 (Cauchy modulus extractor carrier)**.** *A* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, F, M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row whose explicit-modulus coordinate is to be read.*
2. *F is the* $\mathsf{FastCauchyModulus}^{\uparrow}$ *threshold row used when the consumer requests a fast or doubled-window presentation of the same precision query.*
3. *M is the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row, displayed before any finite stream window or regular rational readback is consumed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the stage selected from M and, when needed, refined by F.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the precision query, slack rows, comparison budgets, and tolerance conversions attached to the extracted stage.*
6. *Q is the precision-query row whose requested tolerance is routed through E, F, M, S, D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after the source, extractor, finite-window, and dyadic rows have been displayed.*
8. *H records componentwise* hsame *transport for the source, fast selector, extracted modulus, finite window, dyadic ledger, query, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a regular-Cauchy or Real-facing request through $E, F, M, S, D, Q, R, H, P, N$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusExtractor}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite explicit-modulus extraction packet. It has no coordinate for quotient stream equality, host-selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyModulusExtractorUp.CauchyModulusExtractorTasteGate single carrier alignment`

Theorem 2885.2 (Cauchy modulus extractor NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier $X = (E, F, M, S, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed effective Cauchy source, fast threshold selector, extracted modulus, finite stream window, dyadic tolerance ledger, precision query, regular and Real handoff, transport, replay, provenance, and local naming rows;*
2. *source exposure, requiring the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *row E to be read before M is exported;*
3. *modulus extraction, requiring M to be the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row exposed from E for the same precision query Q;*
4. *threshold compatibility, requiring F to refine only the same selected threshold row, either directly or through the doubled-window route;*
5. *finite-window and dyadic accounting, requiring S and D to be downstream of M and F before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read is consumed;*
6. *handoff nonescape, requiring R to expose only the regular rational readback or Real seal certified by E, F, M, S, D, Q;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No extractor obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3858.1. The accepted packet displays exactly the source effective Cauchy row, the fast selector, the extracted modulus, finite windows, the dyadic ledger, the precision query, the regular and Real handoff, and the structural rows.

Source exposure uses Definition 1705.2. The explicit modulus row is part of the accepted $\mathsf{EffectiveCauchySequence}^{\uparrow}$ carrier surface, so an extractor cannot export M before it has read the source row E.

Threshold compatibility is supplied by Definition 1781.4 and Definition 1781.5. The row F selects the direct fast threshold or the doubled-window threshold for the same query Q, and it creates no third selector coordinate.

The finite-window and dyadic rows then follow chapter 126 and chapter 291. The handoff row R is downstream of that route, while H, C, P, N transport, replay, package, and name the same rows. Hence the listed obligations exhaust the local certificate surface. □

Theorem 2885.3 (Cauchy modulus extractor stage determinacy)**.** *Let $X = (E, F, M, S, D, Q, R, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier. For the displayed precision query Q, every inspected* $\mathsf{StreamName}^{\uparrow}$ *window row S is opened at the stage determined by the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row M together with the fast-threshold row F. Thus two accepted reads with the same transported E, F, M, Q, H, C, P, N rows cannot diverge on the stage used by S or on the dyadic tolerance ledger D.*

Proof. Project the accepted carrier. The only coordinates that mention stage selection are E, F, M, Q together with the downstream window and ledger rows S, D.

The source theorem Definition 1705.3 places every regular rational read behind the explicit modulus row carried by E. The extractor reads that row as M for the same query Q.

The fast-modulus theorems Definition 1781.4 and Definition 1781.5 then determine whether the direct threshold or doubled threshold is used, and identify the corresponding S and D rows.

Since the carrier has no additional selector coordinate, transported agreement of E, F, M, Q, H, C, P, N forces the same selected stage, the same finite window, and the same dyadic ledger. □

Theorem 2885.4 (Cauchy modulus extractor Real handoff nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier $X = (E, F, M, S, D, Q, R, H, C, P, N)$ must first read the finite route*

$$E \longmapsto M, F, Q \longmapsto S, D \longmapsto R \longmapsto H, C, P, N.$$

Consequently the handoff exports no quotient stream equality, host-selected limit, countable-choice schedule, trichotomy principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, hidden stream read, Lean verification row, or bridge datum.*

Proof. Start from a regular-Cauchy or Real-facing consumer. The carrier definition exposes the handoff only as the row R, downstream of E, F, M, S, D, Q.

Apply Definition 1705.5. The effective source row E permits a Real-facing read only after its explicit modulus, finite windows, dyadic ledger, and regular rational handoff have been displayed.

Apply Definition 3858.3. The extractor route fixes the stage row used by S and the tolerance account stored in D before the handoff row is consumed.

The structural rows H, C, P, N preserve that route and contain no excluded analytic or verification coordinate. Therefore the regular-rational or Real handoff cannot escape the displayed extraction packet. □

Falsifiable prediction: Every $\mathsf{CauchyModulusExtractor}^{\uparrow}$ regular or Real-facing read factors through an EffectiveCauchySequence source row, an extracted CauchyModulus row, a FastCauchyModulus threshold row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: effective-cauchy-sequence, fast-cauchy-modulus, cauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyModulusExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source effective-Cauchy row, extracted modulus, fast threshold selector, finite window, dyadic ledger, regular and Real handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3858.1, Definition 3858.2, Definition 3858.3, and Definition 3858.4 seed the finite explicit-modulus extraction route from EffectiveCauchySequence$^{\uparrow}$ through FastCauchyModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusExtractor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host-selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, modulus extraction, threshold compatibility, finite-window and dyadic accounting, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusExtractorUp.

Constructive story: A CauchyModulusExtractor$^{\uparrow}$ packet is generated from an EffectiveCauchySequence source row, a FastCauchyModulus threshold row, an extracted CauchyModulus row, finite StreamName windows, a DyadicRatCore tolerance ledger, a precision query, a RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2886

A Concrete Naming Certificate for $\mathsf{RealArithmeticProgression}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealArithmeticProgression}^{\uparrow}$ packet records the textbook arithmetic progression route $a, a+d, a+2d, \ldots$ as finite BEDC data before any completed real-analysis theorem is read. It sits over the terminal real-name seal of chapter 38, the regular rational readback surface of chapter 301, the finite stream-window discipline of chapter 126, and the finite dyadic arithmetic ledgers of chapter 291. The packet supplies a constructive L10 sequence object for later finite-window sequence and series arguments; it is not a host function, not an infinite completed sum, not a quotient-real equality, not trichotomy, and not an ambient completeness principle.

Definition 2886.1 (Real arithmetic progression carrier)**.** *A* $\mathsf{RealArithmeticProgression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (a, d, S, R, D, H, C, P, N)$$

with the following displayed rows:

1. *a is the initial* $\mathsf{Real}^{\uparrow}$ *row, read through its displayed regular-rational name rather than through a host real value;*
2. *d is the common-difference* $\mathsf{Real}^{\uparrow}$ *row, read through the same real-name boundary;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule listing the indices currently requested from the progression;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose nth window is obtained by the displayed finite arithmetic route for $a + nd$;*
5. *D is the finite dyadic arithmetic ledger recording the repeated addition budget, tolerance scale, and endpoint readbacks used by the requested windows;*
6. *H records componentwise* hsame *transport for the initial real row, common-difference row, schedule, regular-rational readback, dyadic ledger, continuation, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a finite index request through S, then through the dyadic ledger D, the regular-rational readback R, and the endpoint real rows a, d;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host function $\mathbb{N} \to \mathbb{R}$, quotient equality of real representatives, selected limits, infinite sums, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealArithmeticProgressionUp.RealArithmeticProgressionTasteGate single carrier alignment`

Theorem 2886.2 (Real arithmetic progression NameCert obligations)**.** *For every accepted* RealArithmeticProgressio *carrier $A = (a, d, S, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the initial real row a, common-difference row d, finite schedule S, regular-rational readback R, dyadic arithmetic ledger D, transport, continuation, provenance, and local naming row;*
2. *finite-window exactness, requiring each inspected index to be replayed through S before the arithmetic readback R is exposed;*
3. *arithmetic-ledger exactness, requiring the displayed row D to account for the finite repeated-addition route from a and d to every requested readback;*
4. *real-name routing, requiring every downstream* $\mathsf{Real}^{\uparrow}$ *or sequence-facing consumer to reach the progression through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *rows;*
5. *transport stability, requiring componentwise* hsame *transport to preserve the same initial row, common-difference row, schedule, readback, and dyadic ledger;*
6. *ledger refusal, excluding host functions, completed infinite sums, quotient-real equality, trichotomy, selected limits, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3859.1. This exposes exactly the rows $a, d, S, R, D, H, C, P, N$ and no detached operation row.

Replay the continuation C at a finite index request. The request first passes through the $\mathsf{StreamName}^{\uparrow}$ schedule S, then reads the repeated-addition ledger D, and only then exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback R tied to the endpoint rows a and d.

Transport by H is componentwise, so changing representative histories preserves the same initial real row, common-difference row, finite schedule, arithmetic ledger, readback, replay route, provenance, and local naming row. The provenance row P cites only the displayed Real, regular-rational, stream, dyadic, and kernel surfaces.

Since the tuple has no coordinate for a host sequence function, quotient-real equality, infinite sum, trichotomy decision, selected limit, or ambient completeness theorem, those exits cannot be read as local $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ obligations. □

Lean4 ✓ `BEDC.Derived.RealArithmeticProgressionUp.RealArithmeticProgressionNameCertObligations`

Closure status for $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$.

Constructive story: A $\mathsf{RealArithmeticProgression}^{\uparrow}$ packet is generated from an initial Real row, a common-difference Real row, a finite StreamName schedule, a RegSeqRat readback, a DyadicRatCore repeated-addition ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$, seed)

Scope closed: Definition 3859.1 and Definition 3859.2 seed the finite arithmetic-progression route from $\mathsf{Real}^{\uparrow}$ endpoints through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completeness of $\mathsf{Real}^{\uparrow}$, trichotomy, quotient-real equality, host functions $\mathbb{N} \to \mathbb{R}$, selected limits, infinite sums, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, arithmetic-ledger exactness, real-name routing, transport stability, and ledger refusal for the same packet.

Chapter 2887

A Concrete Naming Certificate for $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ packet records the finite readback ledger through which a regular Cauchy presentation is reflected into completion-facing consumers. It sits after the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the reflector surface of chapter 1585, and the extension boundary of chapter 397. The ledger exposes a BEDC-native reflection route for completion consumers without quotient streams, selected limits, host real equality, countable choice, or ambient completeness.

Definition 2887.1 (Regular Cauchy reflection ledger carrier)**.** *A* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, W, D, R, K, U, J, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational Cauchy readback.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the source row is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the regular Cauchy read.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from Q, W, D.*
5. *K is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row recording the reflector-facing handoff from the displayed regular Cauchy route.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row consumed by completion-facing maps.*
7. *J is the reflection ledger row: it records that the same Q, W, D, R source route feeds both K and U.*
8. *H records componentwise* hsame *transport for Q, W, D, R, K, U, J and the local naming row.*
9. *C records displayed* Cont *replay of the route*
$$Q, W, D \longmapsto R \longmapsto K, U \longmapsto J, H, P, N.$$
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, W, D, R, K, U, J, H, C, P, N$. It has no coordinate for quotient stream classes, selected completion limits, host equality of real names, countable-choice schedules, hidden cofinal tails, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyReflectionLedgerUp.RegularCauchyReflectionLedgerTasteGate`
`single carrier alignment`

Theorem 2887.2 (Regular Cauchy reflection ledger NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *carrier $L = (Q, W, D, R, K, U, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflection}}$ *surface consists exactly of carrier admission; regular rational source exposure through Q; finite stream exposure through W; dyadic tolerance exposure through D; terminal real sealing through R; reflector handoff through K; universal-property extension through U; reflection ledger binding through J; componentwise transport through H; displayed replay through C; provenance through P; local naming through N; and non-escape from quotient streams, selected limits, host real equality, countable choice, hidden cofinal tails, or ambient completeness.*

Proof. Carrier admission is projection from Definition 3860.1. The accepted packet exposes exactly the regular rational source, finite stream schedule, dyadic tolerance ledger, real seal, reflector handoff, universal-property extension row, reflection ledger, transport, replay, provenance, and local name.

The source route is finite. Rows Q, W, D determine the regular Cauchy read before R can be consumed, so the packet never begins from a quotient stream class or a selected limit representative.

The completion-facing route is read only after the real seal is displayed. Rows K and U expose the reflector and universal-property surfaces, while J records that both surfaces use the same Q, W, D, R source route.

Rows H, C, P, N transport, replay, source, and name the same ledger. Since no excluded host-side coordinate exists in the carrier, the listed rows exhaust the local $\mathsf{NameCert}$ surface. □

Theorem 2887.3 (Regular Cauchy reflection ledger universal handoff)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionReflector}^{\uparrow}, \mathsf{CompletionUniversalProperty}$$

The handoff keeps the regular Cauchy source, finite observation schedule, dyadic tolerance ledger, terminal real seal, reflector row, universal-property row, transport, replay, provenance, and local naming inside one finite packet. It exports no quotient stream, selected limit, host real equality, countable-choice schedule, hidden cofinal tail, or ambient completeness witness.

Proof. Start with the displayed source rows. The regular rational readback Q is inspected through the finite windows W and the dyadic tolerance ledger D, and only then reaches the real seal R.

The reflector and universal-property rows are downstream of that same source route. The row K records the reflector-facing handoff, while U records the extension-facing handoff for completion consumers.

The ledger row J binds the two completion reads to the same Q, W, D, R route. The structural rows H, C, P, N transport, replay, source, and name that binding, so every completion-facing consumer remains inside the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ consumer that reaches a completion-facing reflector or universal-property row while omitting any of the regular rational source, finite stream schedule, dyadic tolerance ledger, terminal real seal, reflection ledger, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchycompletionreflector, completionuniversalproperty: none is bijective because $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ jointly carries the source readback, finite observation schedule, tolerance ledger, real seal, reflector handoff, universal-property handoff, reflection binding, transport, replay, provenance, and local NameCert rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3860.1, Definition 3860.2, and Definition 3860.3 seed the finite regular-Cauchy reflection ledger over RegSeqRat, StreamName, DyadicRatCore, Real, CauchyCompletionReflector, CompletionUniversalProperty, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyReflectionLedgerUp.RegularCauchyReflectionLedgerTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, hidden cofinal tails, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular source exposure, tolerance exactness, reflector handoff, universal-property handoff, ledger binding, structural transport, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyReflectionLedgerUp.

Constructive story: A $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ packet is generated from a regular rational source row, finite stream windows, dyadic tolerance data, a terminal Real seal, a CauchyCompletionReflector row, a CompletionUniversalProperty row, a reflection ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 2888

A Concrete Naming Certificate for $\mathsf{TubeLemma}^{\uparrow}$

Following Munkres and Kelley on product topology and compact neighbourhoods, the $\mathsf{TubeLemma}^{\uparrow}$ packet records the finite BEDC surface for the tube lemma: when a slice around a compact factor lies inside an open product-neighbourhood row, a displayed neighbourhood in the other factor suffices for the whole compact slice. It sits over the product topology certificate of chapter 2000, the generated-open topology rows of chapter 88, the compact finite-cover surface of chapter 60, and the finite-cover siblings such as chapter 1596 and chapter 3430. The chapter names only the finite tube-neighbourhood carrier and its local obligations; it does not assert product compactness, quotient equality of product points, selected neighbourhood bases, or ambient open-cover compactness beyond the displayed compact row.

Definition 2888.1 (Tube lemma carrier)**.** *A* $\mathsf{TubeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, K, O, U, V, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Topology}^{\uparrow}$ *row naming the first factor.*
2. *Y is the source* $\mathsf{Topology}^{\uparrow}$ *row naming the second factor.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *row for the compact slice inside the second factor.*
4. *O is the* ***??*** *open-neighbourhood row containing the displayed slice.*
5. *U is the finite family of product-open rows read from O along the compact slice.*
6. *V is the resulting neighbourhood row in the first factor.*
7. *F is the finite-cover row over K that justifies replacing the family U by one displayed tube neighbourhood.*
8. *H records componentwise* hsame *transport for the two topology rows, compact row, product-open row, finite family, tube neighbourhood, finite-cover row, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a slice-neighbourhood request through $X, Y, K, O, U, V, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{TubeLemma}^{\uparrow}$*,* ***??****,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K, O, U, V, F, H, C, P, N$. It has no coordinate for global product compactness, a host topological theorem, quotient equality of product points, an arbitrary open-cover selector, or a detached neighbourhood basis.

Lean4 ✓ `BEDC.Derived.TubeLemmaUp.TubeLemmaTasteGate single carrier alignment`

Theorem 2888.2 (Tube lemma NameCert obligations)**.** *For every accepted* $\mathsf{TubeLemma}^{\uparrow}$ *carrier $T = (X, Y, K, O, U, V, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$ *surface is exhausted by carrier admission, topology-row exposure for both factors, compact-slice exposure through K, product-open exposure through O, finite product-neighbourhood family exposure through U, tube-neighbourhood exposure through V, finite-cover justification through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3861.1. The accepted tuple exposes the two topology rows, the compact slice row, the product-open row, the finite product-neighbourhood family, the resulting tube neighbourhood, and the finite-cover row.

The compactness use is local to K and F. The finite family U is read from the displayed product-open neighbourhood O along the compact slice, and F records the finite-cover handoff that permits one neighbourhood row V in the first factor to cover the whole slice.

Finally transport and replay preserve the same finite route. Since H, C, P, N contain only componentwise transport, continuation replay, provenance, and naming, they cannot add product compactness, quotient product equality, selected bases, or an arbitrary open-cover selector. The listed rows therefore exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{TubeLemma}^{\uparrow}$ packet is generated from two topology rows, a compact slice row, a product-open neighbourhood row, a finite product-neighbourhood family, a tube neighbourhood row, a finite-cover row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$, seed)

Scope closed: Definition 3861.1 and Definition 3861.2 seed the finite tube-neighbourhood packet over product topology, compact slice, generated-open, and finite-cover rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close product compactness, arbitrary open-cover compactness, quotient equality of product points, selected neighbourhood bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topology-row exposure, compact-slice exposure, product-open exposure, finite-cover justification, tube-neighbourhood exactness, transport replay, provenance exhaustion, and local naming for BEDC.Derived.TubeLemmaUp.

Chapter 2889

A Concrete Naming Certificate for $\mathsf{FilterLimitBasis}^{\uparrow}$

Following Kelley and Bourbaki on filters, convergence, and completion by directed neighbourhood data, the $\mathsf{FilterLimitBasis}^{\uparrow}$ packet records the finite basis interface used when a completion consumer asks for limit evidence before reading a terminal real-name seal. It sits after the Cauchy-filter completion boundary of chapter 1569, the Cauchy-filter surface of chapter 333, the limit boundary of chapter 1558, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a traditional finite filter-basis route for limit reads; it is not a quotient filter, selected limit, host real equality, ultrafilter argument, countable-choice selector, or ambient completeness theorem.

Definition 2889.1 (FilterLimitBasis carrier)**.** *A* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, F, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-boundary row whose accepted reads are still tied to finite Cauchy-filter data.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row exposing the saturated finite-filter surface consumed by the basis request.*
3. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *limit-boundary row, read only after the same filter and basis data have been displayed.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the basis request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback at the requested precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after Q, F, L, W, R, D have been replayed.*
8. *H records componentwise* hsame *transport for completion, filter, limit, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a limit-basis request through $Q, F, L, W, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, L, W, R, D, E, H, C, P, N$. There is no coordinate for quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier field projection round trip`

Theorem 2889.2 (FilterLimitBasis NameCert obligations)**.** *For every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed Cauchy-filter completion row, Cauchy-filter row, limit-boundary row, finite stream window, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *completion-basis exposure, requiring every completion-facing read to consume Q and F before it consults L, W, R, D, or E;*
3. *limit compatibility, requiring the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row L to use the same displayed filter, window, readback, and dyadic tolerance route;*
4. *Real handoff, requiring the terminal seal E to remain downstream of Q, F, L, W, R, D and the structural rows H, C, P, N;*
5. *ledger refusal, excluding quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, and bridge certificates.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisNameCertObligations`

Proof. Project the carrier of Definition 3862.1. The displayed rows are exactly $Q, F, L, W, R, D, E, H, C, P, N$, so carrier admission exposes only finite completion, filter, limit, window, readback, tolerance, seal, transport, replay, provenance, and naming data.

The completion-basis route starts with Q and F. These rows force the request to pass through the accepted Cauchy-filter completion boundary and the saturated Cauchy-filter surface before any limit-facing row can be inspected.

Next read L, W, R, D. The limit boundary consumes the same finite stream windows, regular rational readback, and dyadic tolerance ledger, so the packet has one displayed basis route rather than two unrelated filter presentations.

Finally the terminal seal E and the structural rows H, C, P, N preserve the same route. Since quotient filters, host equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, and bridge certificates are absent from the carrier, the listed obligations exhaust the local surface. □

Theorem 2889.3 (FilterLimitBasis CauchyFilterCompletion handoff)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier factors through the finite route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus a completion-facing read receives the displayed Cauchy-filter completion boundary, the saturated filter surface, the limit boundary, finite stream windows, regular rational readback, dyadic tolerance accounting, the terminal real-name seal, componentwise transport, replay, provenance, and local naming. It receives no quotient filter, ultrafilter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or bridge certificate.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCauchyFilterCompletionHandoff`

Proof. Begin with a completion-facing consumer of B. The carrier admits the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row Q only together with the finite filter and limit rows F, L.

Replay the displayed continuation row C. The request consumes Q and F, then passes through the limit row L, finite windows W, regular readback R, and dyadic tolerance ledger D before the terminal seal E is available.

The structural rows do not create an additional completion object. Transport through H, provenance through P, and local naming through N record the same finite route, so every $\mathsf{CauchyFilterCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives only that route. □

Theorem 2889.4 (FilterLimitBasis limit compatibility)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier. The limit-boundary row L, finite-window row W, regular readback row R, and dyadic tolerance row D are read from the same displayed filter-basis data exposed by Q and F. In particular, a limit-facing consumer cannot replace the filter of F with a second host filter, detach W from the displayed basis request, or read R, D as tolerance evidence for a different filter presentation.*

↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasis limit compatibility`
↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier limit compatibility`

Proof. Project the carrier of Definition 3862.1. Its only completion and filter coordinates are Q and F, and its only limit-facing coordinates are L, W, R, D.

Apply the NameCert obligations of Definition 3862.2. The limit-compatibility row requires L to use the same displayed filter, window, readback, and dyadic tolerance route rather than a detached filter presentation.

Finally read the handoff theorem Definition 3862.3. Every completion-facing consumer follows the finite route Q, F, L, W, R, D before the terminal seal appears, so L, W, R, D remain tied to the filter-basis data already exposed by Q and F. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisLimitCompatibility`

Theorem 2889.5 (FilterLimitBasis ledger refusal)**.** *For every accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the local ledger has no coordinate for quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, or bridge certificates. Any request for one of those objects is therefore refused by the absence of a carrier row and cannot be recovered from $Q, F, L, W, R, D, E, H, C, P, N$.*

Proof. Inspect the carrier definition. The displayed tuple contains completion, filter, limit, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows, and no row with the requested quotient, ultrafilter, selected-limit, choice, completeness, or bridge role.

Apply Definition 3862.2. Its ledger-refusal clause lists exactly those excluded objects and states that the local NameCert surface stores no obligation outside the displayed rows.

The handoff theorem Definition 3862.3 then identifies every completion-facing and Real-facing read with the route through $Q, F, L, W, R, D, E, H, C, P, N$. Since the route has no additional ledger coordinate, the excluded objects remain refused for every accepted packet. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier ledger refusal`

Theorem 2889.6 (FilterLimitBasis Real handoff stability)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Componentwise transport of the displayed rows $Q, F, L, W, R, D, E, H, C, P,$ preserves the terminal* $\mathsf{Real}^{\uparrow}$ *handoff of Definition 3862.3. In particular, any transported completion-facing or Real-facing read still factors through the same finite route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N$$

and cannot create quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.

Proof. Start with the carrier definition Definition 3862.1. Its visible rows are exactly $Q, F, L, W, R, D, E, H, C, P, N,$ and the transport row H is componentwise over those rows rather than over an ambient filter quotient or a selected limit.

Apply the NameCert obligations of Definition 3862.2. They require completion-basis exposure through Q, F, limit compatibility through L, W, R, D, Real handoff through E, and structural replay through H, C, P, N.

Finally use the handoff theorem Definition 3862.3. Since the transported packet has the same displayed coordinates and no additional carrier row, the terminal $\mathsf{Real}^{\uparrow}$ handoff remains downstream of Q, F, L, W, R, D and cannot acquire quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisRealHandoffStability`

Theorem 2889.7 (FilterLimitBasis scoped route)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet exposes completion-facing limit evidence only through the finite route*

$$\mathsf{CauchyFilterCompletion}^{\uparrow} \longrightarrow \mathsf{CauchyFilter}^{\uparrow} \longrightarrow \mathsf{CauchyFilterLimit}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRa}$$

followed by the structural hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. Equivalently, in the carrier notation of Definition 3862.1, every completion-facing, limit-facing, or Real-facing read must pass through $Q, F, L, W, R, D, E, H, C, P, N$ in that order before it is exported as limit-basis evidence.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedRoute`

Proof. Start with the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the only local obligation rows.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. It forces every completion-facing and Real-facing consumer through Q, F, L, W, R, D before the terminal seal E and the structural rows are available.

Next use the limit compatibility theorem Definition 3862.4. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ row L, finite window row W, regular readback row R, and dyadic tolerance row D remain attached to the same displayed filter-basis data exposed by Q and F.

Finally apply Real handoff stability from Definition 3862.6. Componentwise transport preserves the same route through H, C, P, N, so the scoped route is exactly the displayed completion, filter, limit, stream, regular-rational, dyadic, Real, and structural sequence. □

Theorem 2889.8 (FilterLimitBasis window-basis exhaustion)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *request must expose the completion row* Q*, the Cauchy-filter row* F*, the limit-boundary row* L*, the finite-window row* W*, the regular readback row* R*, and the dyadic tolerance row* D *before it may read the terminal seal* E*. Only after that finite basis has been displayed may the request continue through* H, C, P, N*. Consequently the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*, and* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *rows are all visible before any* $\mathsf{Real}^{\uparrow}$ *consumer can inspect the seal.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisWindowBasisExhaustion`

Proof. Project the carrier of Definition 3862.1. Its visible order places Q, F, L, W, R, D before the terminal seal row E, and then records transport, replay, provenance, and local naming as H, C, P, N.

Apply the NameCert obligations of Definition 3862.2. The completion-basis row forces Q, F to be read first, the limit-compatibility row ties L, W, R, D to that same request, and the Real handoff row keeps E downstream of those finite rows.

Finally use Definition 3862.4 and Definition 3862.6. They preserve the same displayed basis under limit-facing and transported Real-facing reads, so no $\mathsf{Real}^{\uparrow}$ consumer can reach the seal before the StreamName, RegSeqRat, DyadicRatCore, CauchyFilter, and CauchyFilterLimit rows are visible. □

Theorem 2889.9 (FilterLimitBasis window induction)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier, and let* Λ *be a finite replay ledger whose entries are finite window reads or structural transports of the displayed rows. Induction on* Λ *preserves the following invariant: every* $\mathsf{Real}^{\uparrow}$*-facing consumer exposes* Q, F, L, W, R, D *before* E*, and every later replay through* H, C, P, N *preserves that same basis route. Thus no induction step can introduce a quotient filter, ultrafilter, selected limit, countable-choice schedule, host filter equality, host real equality, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisWindowInduction`

Proof. For the empty ledger, apply Definition 3862.8. The terminal seal is available only after Q, F, L, W, R, D have been displayed, and the remaining rows are exactly the structural route H, C, P, N.

For the induction step, assume the finite prefix of Λ preserves the displayed basis route. If the next entry is a finite window read, use Definition 3862.4: the rows L, W, R, D remain tied to the same filter-basis data exposed by Q and F.

If the next entry is a structural transport or replay, use Definition 3862.6. Componentwise transport of $Q, F, L, W, R, D, E, H, C, P, N$ keeps the $\mathsf{Real}^{\uparrow}$ handoff downstream of the same finite basis and does not create an additional carrier coordinate.

The carrier of Definition 3862.1 has no row for quotient filters, ultrafilters, selected limits, choice schedules, host equality, or ambient completeness. Therefore the invariant holds for every finite ledger Λ. □

Theorem 2889.10 (FilterLimitBasis finite-prefix StreamName consumption)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C,$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Every accepted request that reaches the terminal row* E *first consumes the finite prefix StreamName windows* W*, the RegSeqRat readback row* R*, and the DyadicRatCore tolerance row* D *through the same* Q, F, L *basis route. Consequently a Real or completion-facing read cannot replace the displayed finite-prefix route by a quotient filter, a selected limit, an ambient completeness row, or a detached stream representative.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisFinitePrefixStreamConsumption`

Proof. Project the carrier of Definition 3862.1. The only route to E is the displayed chain through Q, F, L, W, R, D followed by the structural rows H, C, P, N.

Apply Definition 3862.8. It forces every Real-facing or completion-facing request to expose Q, F, L, W, R, D before the terminal seal is read.

Then apply Definition 3862.9. Finite replay of the request preserves the same basis route through every prefix of the ledger, so W, R, D remain tied to the same Cauchy-filter completion, Cauchy-filter, and limit-boundary rows.

Since Definition 3862.1 has no coordinate for quotient filters, selected limits, detached stream representatives, or ambient completeness, none of those objects can be introduced before the E handoff. □

Theorem 2889.11 (FilterLimitBasis real-window exhaustion)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier exposing the completion-basis row, limit-compatibility row,* $\mathsf{Real}^{\uparrow}$ *handoff-stability row, and finite-window induction row. Then every limit-basis consumer must expose a finite window tuple through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it may read terminal* $\mathsf{Real}^{\uparrow}$ *evidence. In particular, the consumer has no route through quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisRealWindowExhaustion`

Proof. First apply Definition 3862.3. It sends every completion-facing or Real-facing request through the displayed route $Q, F, L, W, R, D, E, H, C, P, N$, so the terminal seal is not available before the completion and filter rows have been read.

Next apply Definition 3862.4. The limit boundary L remains tied to the same finite window row W, regular readback row R, and dyadic tolerance row D exposed by Q and F.

Then use Definition 3862.6 and Definition 3862.9. Transport and finite replay preserve the same route for every ledger prefix, so the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows exhaust the visible window before terminal Real evidence is read.

The carrier definition has no row for quotient filters, selected limits, ultrafilters, countable-choice schedules, host equalities, or ambient completeness. Therefore any consumer using one of those objects has left the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ packet rather than reading its real-window exhaustion surface. □

Theorem 2889.12 (FilterLimitBasis obligation-closure readiness)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. The carrier admission of Definition 3862.1, the local NameCert obligations of Definition 3862.2, the CauchyFilterCompletion handoff of Definition 3862.3, the limit-compatibility row of Definition 3862.4, the ledger refusal of Definition 3862.5, the Real handoff stability of Definition 3862.6, the window-basis exhaustion of Definition 3862.8, the window induction of Definition 3862.9, and the real-window exhaustion of Definition 3862.11 jointly provide the paper-side obligation surface needed to move the chapter from seed closure toward obligation closure.*

The readiness surface is still finite BEDC data: it records only the displayed completion row, Cauchy-filter row, limit-boundary row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise transport, replay, provenance, and local NameCert rows. It introduces no quotient filter, selected limit, ultrafilter, countable-choice schedule, host equality, ambient Real completeness, or bridge certificate.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisObligationClosureReadiness`

Proof. Project the carrier row of Definition 3862.1. The only available coordinates are $Q, F, L, W, R, D, E, H, C, P, N$ so any obligation-closure package must be assembled from those finite rows.

Apply Definition 3862.2, Definition 3862.3, and Definition 3862.4. These rows supply carrier admission, the completion-facing route, and the condition that the limit boundary uses the same finite filter-basis data.

Next apply Definition 3862.5, Definition 3862.6, and Definition 3862.8. They keep the terminal Real seal downstream of the displayed basis and exclude quotient, choice, ultrafilter, host-equality, ambient-completeness, and bridge coordinates.

Finally use Definition 3862.9 and Definition 3862.11. Finite replay preserves the same window route through every ledger prefix, so the chapter has the paper-side obligation surface required for the next closure step without adding any object outside the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier. □

Theorem 2889.13 (FilterLimitBasis scope exactness)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Every consumer read exported by the local limit-basis surface factors through the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row Q, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row L, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *row N before any limit-basis export is read. The scope contains no selected limit, quotient filter, ultrafilter, countable-choice schedule, host filter equality, host real equality, or ambient real-completeness premise.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopeExactness`

Proof. Read the carrier rows in order. The completion row Q and Cauchy-filter row F determine the finite filter-basis request, and the limit-boundary row L attaches the requested limit evidence to that same basis.

The finite-prefix material then fixes the observation route. By Definition 3862.10, the windows W, regular readback R, and dyadic tolerance ledger D are consumed before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be inspected.

The structural rows preserve exactness rather than adding new objects. Componentwise hsame transport in H and displayed Cont replay in C keep every consumer read on the same route, while P and N record only provenance and local naming.

Therefore any exported limit-basis read has passed through $Q, F, L, W, R, D, E, H, C, P, N$. The refused objects have no carrier coordinate and no replay edge, so selected limits, quotient filters, countable choice, and ambient real completeness cannot be smuggled into the scope. □

Theorem 2889.14 (FilterLimitBasis scoped obligation)**.** *For an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the obligation surface is scoped exactly by the displayed route through* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. Every completion-facing, limit-facing, or Real-facing read must pass through those rows, and no quotient filter, selected limit, ultrafilter, countable-choice schedule, host filter equality, host real equality, ambient completeness, or bridge certificate is part of the scope.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedObligation`

Proof. Start from the carrier of Definition 3862.1. It displays precisely the rows $Q, F, L, W, R, D, E, H, C, P, N$, so the candidate scope is already bounded by the completion boundary, Cauchy-filter surface, limit

boundary, window, readback, tolerance, Real seal, transport, replay, provenance, and local naming data.

Apply the NameCert obligations and the CauchyFilterCompletion handoff, Definition 3862.2 and Definition 3862.3. They force every completion-facing consumer through Q, F, L, W, R, D before E and keep H, C, P, N attached to the same finite route.

Use limit compatibility, window induction, and real-window exhaustion from Definition 3862.4, Definition 3862.9, and Definition 3862.11. These theorems preserve the same route for limit-facing and Real-facing reads, so the scoped obligation is exactly the displayed L10 completion route and contains no additional host or bridge coordinate. □

Theorem 2889.15 (FilterLimitBasis kernel scope upgrade)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet routes completion-facing limit evidence through the kernel sequence*

$$\mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyFilterLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

and only then through BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. In carrier notation, every accepted read is scoped by* $Q, F, L, W, R, D, E, H, C, P, N$ *before it is exported as limit-basis evidence.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisKernelScopeUpgrade`

Proof. Use the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the local obligation rows.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. A completion-facing consumer must pass through Q, F, L, W, R, D before the terminal seal E and the structural rows can be read.

The limit compatibility theorem and window-basis exhaustion theorem, Definition 3862.4 and Definition 3862.8, keep the $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows attached to the same displayed basis.

Finally use finite-prefix StreamName consumption from Definition 3862.10. The structural rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ replay and name that route without adding quotient filters, selected limits, countable choice, host equality, ambient completeness, or bridge data. □

Theorem 2889.16 (FilterLimitBasis cofinal subbasis scope)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier whose filter row* F *is generated from a displayed* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *route. Before* B *exports limit-basis evidence, the consumer must expose the subbasis finite intersections, the generated* $\mathsf{FilterBase}^{\uparrow}$ *row, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The export is therefore scoped by the cofinal subbasis route together with* $Q, F, L, W, R, D, E, H, C, P, N$ *and it contains no quotient filter, selected limit, ultrafilter, countable-choice schedule, host filter equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge certificate.*

Proof. Start with the generated-base cofinality and finite-intersection route of Definition 4232.4 and Definition 4232.5. They force a subbasis-generated completion read through finite subbasis tokens, finite intersections, generated events, generation ledger, and a generated $\mathsf{FilterBase}^{\uparrow}$ row before any downstream Cauchy-filter row can be consumed.

Compose this with the Cauchy filter subbasis completion handoff of Definition 4232.7. The handoff keeps the generated base before the $\mathsf{CauchyFilter}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Now apply the local scoped route of Definition 3862.15. Once the generated $\mathsf{CauchyFilter}^{\uparrow}$ row is the filter row F of the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier, the limit-basis read must continue through $Q, F, L, W, R, D, E, H, C, P, N$ in the kernel order displayed there.

The structural rows transport and name the same finite route. Since neither the subbasis carrier nor the limit-basis carrier has a coordinate for quotient filters, selected limits, ultrafilters, countable choice, host filter equality, ambient completeness, or bridge certificates, the composed scope is exactly the cofinal subbasis surface stated above. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCofinalSubbasisScope`

Theorem 2889.17 (FilterLimitBasis cofinal Cauchy-entourage induction)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet whose finite basis rows refine the Cauchy-filter route of* $\mathsf{CauchyFilter}^{\uparrow}$*, the completion boundary of* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*, the Real-induced uniformity of* $\mathsf{RealUniformStructure}^{\uparrow}$*, the regular rational readback of* $\mathsf{RegSeqRat}^{\uparrow}$*, and the source-window surface of* $\mathsf{StreamName}^{\uparrow}$*. Induction over the finite basis-window ledger produces a cofinal entourage route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N$$

such that every Real-facing limit read must first pass through the displayed Cauchy-filter basis, the completion boundary, the Real-uniform entourage row, the StreamName windows, the RegSeqRat readback, and the dyadic tolerance ledger before any terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed. The route exports no selected limit, quotient filter, host filter equality, ultrafilter, countable-choice schedule, ambient Real completeness, or bridge certificate.*

Proof. Start with the finite ledger induction of Definition 3862.9. Its base case reads the displayed rows Q, F, L, W, R, D before E, and its step case preserves the same route under every finite window read or structural replay.

Use the real-window exhaustion theorem Definition 3862.11. It identifies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as the visible Real-facing window route, so no terminal Real evidence is read before the basis windows and regular rational handoff are exposed.

Finally apply the kernel scope upgrade of Definition 3862.15. The upgraded scope places $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in one displayed kernel route. Since the accepted carrier has no coordinate for selected limits, quotient filters, host filter equality, ultrafilters, countable-choice schedules, ambient Real completeness, or bridge certificates, the induced entourage route is cofinal only inside that finite BEDC surface. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCofinalCauchyEntourageInduction`

Theorem 2889.18 (FilterLimitBasis scoped completion dependency)**.** *Every completion-facing* $\mathsf{FilterLimitBasis}^{\uparrow}$ *read consumes* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before a filter-limit basis can be sealed. In carrier notation, the read must expose the completion row Q, finite windows W, regular readback R, dyadic tolerance ledger D, terminal real seal E, componentwise transport H, displayed replay C, provenance P, and local naming N before any* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *consumer treats the basis as sealed. The route does not claim selected limits, ultrafilters, countable choice, quotient filters, host filter equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedCompletionDependency`

Proof. Project the carrier of Definition 3862.1. The completion-facing rows include $Q, W, R, D, E, H, C, P, N$, with the Cauchy-filter and limit rows attached by the same finite packet.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. It forces the completion row and the finite-prefix StreamName consumption to appear before the terminal Real seal is inspected.

Use Definition 3862.10 and Definition 3862.11. These results route the windows through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the structural rows replay and name the basis.

The accepted carrier has no coordinate for selected limits, ultrafilters, countable choice, quotient filters, host filter equality, or ambient completeness. Hence the listed rows are the complete scoped dependency for the L10 completion handoff. □

Theorem 2889.19 (FilterLimitBasis public export)**.** *Every public completion-facing read of* $\mathsf{FilterLimitBasis}^{\uparrow}$ *factors through the already scoped route from* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *In carrier notation, the public export must expose* $Q, F, L, W, R, D, E, H, C, P, N$ *before a limit-basis consumer may read the seal. The export contains no quotient filters, ultrafilters, selected limits, countable-choice schedules, host equality, ambient Real completeness, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisPublicExport`
↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier public export`

Proof. Use the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the only local public rows.

Apply the handoff and limit-compatibility theorems, Definition 3862.3 and Definition 3862.4. These force every completion-facing read through Q, F, L, W, R, D before the terminal seal and keep the limit boundary attached to the same displayed filter-basis data.

Use Real handoff stability from Definition 3862.6 together with the scoped route of Definition 3862.15. Componentwise transport, displayed replay, provenance, and local naming preserve the same route through E, H, C, P, N and do not add a second public object.

Finally apply the scoped completion dependency of Definition 3862.18. Since that dependency already excludes selected limits, ultrafilters, countable choice, quotient filters, host equality, and ambient completeness, the public export is exactly the displayed finite route and carries no bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ read is refuted if a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal without displaying the Cauchy-filter completion row, Cauchy-filter row, Cauchy-filter limit row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming.

Independence witness: cauchyfiltercompletion, cauchyfilter, cauchyfilterlimit, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{FilterLimitBasis}^{\uparrow}$ because the packet jointly records completion-boundary, saturated-filter, limit-boundary, finite-window, readback, tolerance, seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterLimitBasis}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, a CauchyFilter row, a CauchyFilterLimit row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$, publicClosed)

Scope closed: Definition 3862.19, Definition 3862.17, and Definition 3862.18 close the public completion-facing export through CauchyFilterCompletion, CauchyFilter, CauchyFilterLimit, StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FilterLimitBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host filter equality, ultrafilter arguments, countable-choice schedules, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Bridge closure requires a standard bridge surface for the same finite public export route.

Chapter 2890

A Concrete Naming Certificate for $\mathsf{PerfectSet}^{\uparrow}$

Following Cantor and Bendixson, the $\mathsf{PerfectSet}^{\uparrow}$ packet names the finite certificate surface for a closed set whose every displayed point remains a derived point. It sits over the open-neighborhood discipline of chapter 88, the compact metric inspection rows of chapter 129, the terminal real-name seal of chapter 38, and the derivative-stage route of chapter 2024. The chapter isolates the reusable perfect-kernel face consumed by Cantor-Bendixson analysis; it is not a transfinite decomposition, a selected point enumeration, quotient real equality, or an ambient completeness theorem.

Definition 2890.1 (PerfectSet carrier)**.** *A* $\mathsf{PerfectSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, K, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the open-neighborhood tests and closed-set complement reads through chapter 88.*
2. *K is the compact-metric source row, carrying the finite metric inspection and cover data supplied by chapter 129.*
3. *D is the derived-point ledger. For every displayed point read from the set and every finite neighborhood request, it records a second compatible point read through the same topology and metric rows.*
4. *W is the finite window schedule for neighborhood, metric, and derived-point requests.*
5. *R is the regular rational readback row used by the metric comparisons and the non-isolation witnesses.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the topology, compact-metric, derived-point, window, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for T, K, D, W, R, E and for the structural replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of every perfect-set consumer request through*

$$T, K, D, W, R \longmapsto E \longmapsto H, C, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{PerfectSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CantorBendixson}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PerfectSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected countable dense subset, transfinite rank, isolated-point enumeration, quotient equality of real names, host closed-set equality, countable choice, or ambient compactness beyond the displayed rows.

Theorem 2890.2 (PerfectSet NameCert obligations)**.** *For every accepted* $\mathsf{PerfectSet}^{\uparrow}$ *carrier* $S = (T, K, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PerfectSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology, compact-metric, derived-point, finite-window, regular readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *closed-set exposure, requiring complement and neighborhood requests to pass through the topology row T and compact-metric row K;*
3. *derived-point non-isolation, requiring each displayed point-and-neighborhood request to receive its second compatible point through D at a window listed in W;*
4. *real-seal compatibility, requiring every metric comparison and non-isolation witness to pass through R before the terminal seal E is consumed;*
5. *ledger nonescape, requiring H, C, P, N to export no isolated-point selector, transfinite derivative rank, quotient real equality, host closed-set equality, countable-choice schedule, or ambient completeness theorem.*

No perfect-set obligation is stored outside these rows.

Proof. Project the accepted carrier $S = (T, K, D, W, R, E, H, C, P, N)$. Carrier admission exposes exactly the topology row, compact-metric source, derived-point ledger, finite window schedule, regular readback row, Real seal, and the structural rows.

Closed-set exposure is read through T and K. The topology row supplies neighborhood and complement tests, while the compact-metric row supplies the finite metric inspection data used by the same requests.

The derived-point clause is the row D. For each displayed point-and-neighborhood request, the finite window row W names the inspected neighborhood and D records the compatible second point before any Real-facing seal is available.

Finally, R and E bind the metric and Real-facing parts of the route, while H, C, P, N transport, replay, source, and name the same packet. Since the excluded classical data are not carrier coordinates, the five listed rows exhaust the local certificate. □

Theorem 2890.3 (PerfectSet derived-point nonescape)**.** *Every* $\mathsf{CantorBendixson}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{PerfectSet}^{\uparrow}$ *packet must read the topology row, compact-metric row, derived-point ledger, finite window schedule, regular readback route, and terminal Real seal before treating the set as a perfect kernel. In particular, the consumer receives the finite route*

$$T, K, D, W, R \longmapsto E \longmapsto H, C, P, N$$

and receives no isolated-point selector, transfinite derivative rank, quotient real equality, host closed-set equality, countable-choice schedule, or ambient compactness theorem outside the displayed certificate.

Proof. Start with a consumer request for the perfect-kernel face. By Definition 3863.2, the request first passes through the topology and compact-metric rows T, K and then through the derived-point ledger D at a finite window listed in W.

The regular readback row R and terminal seal E are the only Real-facing route in the carrier. Thus a Cantor-Bendixson consumer can consume this packet only as the retained perfect-set face displayed by chapter 2024.

The structural rows H, C, P, N preserve and replay the same route. Because the carrier lacks coordinates for isolated-point selection, transfinite ranks, quotient equality, host closed-set equality, countable choice, or extra compactness principles, no such data can escape through the derived-point ledger. □

Lean4 ✓ `BEDC.Derived.PerfectSetUp.PerfectSetCarrier derived point nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$.

Constructive story: A $\mathsf{PerfectSet}^{\uparrow}$ packet is generated from topology rows, compact-metric inspection rows, a derived-point ledger, finite request windows, regular rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$, seed)

Scope closed: Definition 3863.1, Definition 3863.2, and Definition 3863.3 seed the finite perfect-kernel certificate consumed by Cantor-Bendixson and Real-facing routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close transfinite Cantor-Bendixson decomposition, selected isolated-point enumeration, quotient real equality, host closed-set equality, countable choice, ambient compactness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-set exposure, derived-point non-isolation, Real-seal compatibility, ledger nonescape, and Cantor-Bendixson consumer routing for BEDC.Derived.PerfectSetUp.

Chapter 2891

A Concrete Naming Certificate for $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ packet records the finite even–odd interleaving route by which two regular Cauchy rational names are read as one binary stream before any Real or Completion consumer treats the result as a sealed real-facing name. It sits over the finite stream surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 1569. The packet names a pairing-and-projection ledger; it does not identify host streams, select quotient representatives, or import ambient completeness.

Definition 2891.1 (Regular Cauchy binary interleaving carrier)**.** *A* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two finite* $\mathsf{StreamName}^{\uparrow}$ *source rows whose regular rational observations are to be paired.*
2. E *is the even* $\mathsf{StreamName}^{\uparrow}$ *projection row reading the* S_0 *observations from the interleaved stream.*
3. O *is the odd* $\mathsf{StreamName}^{\uparrow}$ *projection row reading the* S_1 *observations from the interleaved stream.*
4. T *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, assigning the same dyadic budget to the two projections and to the interleaved read.*
5. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the even and odd projections after the dyadic budget is fixed.*
6. L *is the* $\mathsf{Real}^{\uparrow}$ *seal compatibility row, requiring the two projected regular names and the interleaved name to expose the same terminal seal discipline to Real and Completion consumers.*
7. H *records componentwise* hsame *transport for the two source rows, projection rows, tolerance ledger, regular readbacks, Real seal row, replay, provenance, and naming rows.*
8. K *records displayed* Cont *replay from an interleaving request through* $S_0, S_1, E, O, T, R_0, R_1, L, H, P, N$.

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N$. It has no coordinate for host stream equality, quotient sequence equality, a selected completion representative, countable-choice data, or an ambient real-completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyBinaryInterleavingUp.RegularCauchyBinaryInterleavingTasteGate`
`single carrier alignment`

Theorem 2891.2 (Regular Cauchy binary interleaving NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *carrier $B = (S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$ *surface is exhausted by five rows:*

1. *carrier admission: the accepted object is exactly the displayed packet $S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N$;*
2. *projection stability: the even row E and odd row O read the two source* $\mathsf{StreamName}^{\uparrow}$ *rows without changing their finite windows;*
3. *dyadic tolerance preservation: the ledger T assigns one dyadic tolerance discipline to the two projections and to the interleaved stream;*
4. *Real seal compatibility: the row L is reachable only after E, O, T, R_0, R_1 have been displayed;*
5. *ledger refusal: H, K, P, N export no quotient sequences, choice selectors, host stream equality, hidden completion selector, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyBinaryInterleavingUp.RegularCauchyBinaryInterleavingNameCertObligatio`

Proof. Project the carrier of Definition 3864.1. Carrier admission exposes exactly the two sources, two projection rows, one dyadic tolerance ledger, two regular rational readbacks, one Real seal compatibility row, and the structural rows.

Read the projection rows next. The even row E is tied to S_0, and the odd row O is tied to S_1, so a consumer sees projection stability as a finite StreamName route rather than as host equality of streams.

The tolerance and readback rows then fix the regular Cauchy surface. The same dyadic ledger T is used before R_0 and R_1 are consumed, and the Real seal compatibility row L is available only after those rows have been displayed.

Finally inspect H, K, P, N. These rows transport, replay, source, and name the same finite packet, and none adds quotient sequences, choice selectors, host stream equality, hidden completion selectors, or ambient completeness data. Thus the five obligations exhaust the local NameCert surface. □

Theorem 2891.3 (Regular Cauchy binary interleaving seal stability)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *packet factors through*

$$S_0, S_1 \longmapsto E, O \longmapsto T \longmapsto R_0, R_1 \longmapsto L \longmapsto H, K, P, N.$$

In particular, the consumer cannot read the terminal $\mathsf{Real}^{\uparrow}$ *seal compatibility row L without first displaying the even–odd projection rows, the shared dyadic tolerance ledger, and the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows.*

Proof. Begin with a Real or Completion request. The only source rows in the carrier are S_0 and S_1, and the only ways to read them through the interleaved stream are the even and odd projection rows E and O.

Next apply the tolerance ledger. The row T fixes the dyadic budget shared by the two projected reads, so the regular rational rows R_0 and R_1 cannot be consumed as independent selected sequences.

The seal row L is downstream of those displayed rows. Since the structural rows H, K, P, N only transport and name the same route, any consumer reaching a Real or Completion conclusion without the projection, tolerance, and readback rows would ask for a coordinate outside the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches the terminal seal compatibility row without displaying both StreamName source rows, even and odd projection rows, the shared DyadicRatCore tolerance ledger, both RegSeqRat readbacks, transport, replay, provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchyfiltercompletion: none alone is bijective with $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ because the packet jointly records two source streams, even and odd projection rows, one dyadic tolerance ledger, two regular rational readbacks, a terminal Real seal compatibility row, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ packet is generated from two StreamName source rows, even and odd projection rows, a DyadicRatCore tolerance ledger, two RegSeqRat readback rows, a Real seal compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3864.1, Definition 3864.2, and Definition 3864.3 seed the finite binary interleaving route for regular Cauchy rational names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequences, host stream equality, choice selectors, selected completion representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection stability, dyadic tolerance preservation, Real seal compatibility, ledger refusal, seal stability, and local naming for BEDC.Derived.RegularCauchyBinaryInterleavingUp.

Chapter 2892

A Concrete Naming Certificate for ??

Following Bishop and Bridges on nested closed intervals, the **??** packet records the constructive point-name extracted from a displayed shrinking chain of closed bounded intervals. It sits after the endpoint carrier of chapter 1591, the retained-chain surface of chapter 320, the intersection route of chapter 389, and the Cantor-style singleton route of chapter 1421. The packet names the point by finite BHist rows over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it does not choose a host real, quotient representative, open-cover compactness proof, or countable-choice schedule.

Definition 2892.1 (Nested interval point-name carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$P = (I, J, K, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint-and-containment source row for each retained interval window;*
2. *J is the* $\mathsf{NestedInterval}^{\uparrow}$ *chain row, recording inclusion of the next retained interval inside the preceding one;*
3. *K is the* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *and* $\mathsf{CantorIntersection}^{\uparrow}$ *singleton-intersection route row;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule used to read endpoints and diameter bounds at requested precision;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from the retained endpoint windows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the named point, reached only after I, J, K, S, R have been displayed;*
7. *H records componentwise* hsame *transport for the interval source, nested chain, intersection row, StreamName windows, RegSeqRat readback, and Real seal;*
8. *C records displayed* Cont *replay from closed bounded intervals through nested containment, intersection, finite windows, regular readback, and the Real seal;*
9. *Q records* Pkg *provenance over **??**,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier has no coordinate for an ambient intersection set, a selected quotient representative, countable choice, open-cover compactness, or an unbounded completeness theorem.

Lean4 ✓ BEDC.Derived.NestedIntervalPointNameUp.NestedIntervalPointNameTasteGate single carrier alignment

Theorem 2892.2 (Nested interval point-name regular Cauchy seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted ?? carrier factors through the displayed route*

$\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}$

In particular, the regular rational readback row R and the terminal seal row E are downstream of the interval source I, nested-chain row J, singleton-intersection row K, and finite schedule row S.

Proof. Project the carrier of Definition 3865.1. The rows I, J, K are the source, nesting, and intersection rows; the rows S, R, E are the finite schedule, regular readback, and terminal Real seal.

Use the closed bounded interval source route from chapter 1591. That route exposes endpoint, containment, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as finite BHist data before any point-facing consumer can read them.

Now apply the nested and Cantor intersection siblings. The $\mathsf{NestedInterval}^{\uparrow}$ and $\mathsf{NestedIntervalIntersection}^{\uparrow}$ rows retain the chain and intersection route, while $\mathsf{CantorIntersection}^{\uparrow}$ supplies the singleton-facing seal. Since H, C, Q, N only transport, replay, source, and name these displayed rows, the exported $\mathsf{Real}^{\uparrow}$ read factors through the stated route. □

Theorem 2892.3 (Nested interval point-name endpoint containment)**.** *For every accepted ?? carrier $P = (I, J, K, S, R, E, H, C, Q, N)$, every finite read of the named point remains contained in each displayed retained* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *window before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The containment read is transported componentwise by H and replayed by C through the same* $\mathsf{NestedInterval}^{\uparrow}$ *and* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *chain.*

Proof. Start with the interval source row I. By the carrier, I is a closed bounded interval packet with lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName, RegSeqRat, and Real-seal rows.

Next use the nested-chain row J. It records that each retained window is included in the preceding one, so a later finite read cannot leave an earlier displayed closed bounded interval without breaking the hsame transport recorded in H.

Finally use the intersection row K. The $\mathsf{NestedIntervalIntersection}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ route supplies the point-facing intersection witness, while C replays the same containment route into S, R, E. Hence endpoint containment is preserved before the named point reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 2892.4 (Nested interval point-name uniqueness route)**.** *If two accepted ?? carriers are classified together by their displayed* hsame *transport rows, then their* $\mathsf{Real}^{\uparrow}$ *point-name seals agree only through the same closed-interval source, nested-chain, intersection, StreamName, and RegSeqRat rows. The classifier exposes no separate host equality of points and no quotient-real representative selection.*

Proof. Compare the two carriers componentwise. The classifier can transport the interval source, nested chain, intersection row, finite schedule, regular readback, and Real seal only along displayed hsame rows.

Apply the endpoint-containment theorem. Each finite read remains inside the same transported retained intervals before the Real seal is consumed, so the uniqueness route is pinned to the displayed closed bounded interval and nested-intersection data.

The terminal comparison is therefore a route comparison, not a host equality principle. Since the carrier has no coordinate for quotient real equality or representative choice, the only available uniqueness evidence is the transported $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route named by the packet. □

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from displayed ClosedBoundedIntervalUp interval rows, a NestedIntervalUp retained-chain row, NestedIntervalIntersectionUp and CantorIntersectionUp intersection rows, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{??}$, seed)

Scope closed: Definition 3865.1, Definition 3865.2, Definition 3865.3, and Definition 3865.4 seed the finite point-name route extracted from nested closed bounded intervals.

Formal verification: FormalStatus($\mathsf{NameCert}_{??}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host interval intersection, selected quotient representatives, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular Cauchy seal routing, endpoint containment, uniqueness routing, transport, replay, provenance, and local naming for the displayed point-name packet.

Chapter 2893

A Concrete Naming Certificate for $\mathsf{HyperbolicVisualMetric}^{\uparrow}$

The $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ packet realizes the disk readout branch of chapter 9463 as finite NameCert data. It sits after the Poincare disk metric surface of chapter 3474, the boundary-geodesic route of chapter 3491, the boundary-shadow packet of chapter 3813, and the boundary-fixed transport surface of chapter 3500. The packet names a simultaneous visual-metric read over displayed disk, metric, boundary, shadow, transport, and provenance rows; it does not turn the observer boundary into an object-level T or a completed global geometry.

Definition 2893.1 (Hyperbolic visual metric carrier)**.** *A* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (D, M, B, S, T, F, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk source row, read through the Poincare disk and unit-disk coordinate surface;*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same disk source;*
3. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary row for geodesic far-end incidence;*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *shadow row attached to the displayed boundary read;*
5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *classifier-preserving boundary transport row;*
6. *F is the global-geometry refusal row, rejecting reads that replace displayed disk, metric, boundary, shadow, and transport rows by an object-level geometry;*
7. *A is the vision anchor row tying the packet to the observer-expansion reading of chapter 9463;*
8. *H records componentwise* hsame *transport for disk source, metric read, boundary row, shadow row, transport row, refusal row, and anchor row;*
9. *C records displayed* Cont *replay from disk source through metric read, boundary read, shadow, transport, refusal, and provenance;*
10. *P records* Pkg *provenance over* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed boundary compactification, object-level T*, physical horizon, global metric oracle, or quotient boundary equality.*

Lean4 (definition) `BEDC.Derived.HyperbolicVisualMetricUp.HyperbolicVisualMetricUp`
Lean4 ✓ `BEDC.Derived.HyperbolicVisualMetricUp.HyperbolicVisualMetricTasteGate single carrier alignment`

Theorem 2893.2 (Hyperbolic visual metric obligation surface)**.** *For the carrier of Definition 3866.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$ *surface consists exactly of disk source admission, hyperbolic metric read admission, geodesic-boundary read admission, boundary-shadow admission, classifier-preserving boundary transport, global-geometry refusal, vision-anchor provenance, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier rows. The disk source D, metric read M, boundary row B, shadow row S, and transport row T are the only geometric data available to the packet.

Use the sibling surfaces in order. $\mathsf{HyperbolicMetric}^{\uparrow}$ supplies the metric read, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ supplies the boundary incidence route, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ supplies the shadow row, and $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ supplies classifier-preserving transport.

The refusal and structural rows finish the surface. The row F refuses any read that bypasses the displayed siblings, while A, H, C, P, N only anchor, transport, replay, source, and name the same finite route. Thus the listed rows exhaust the local certificate. □

Theorem 2893.3 (Hyperbolic visual metric global-geometry refusal)**.** *A visual metric read of an accepted* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *packet is refused unless the displayed route contains disk source, metric read, boundary row, boundary shadow, classifier-preserving transport, provenance, and the global-geometry refusal row. The packet cannot replace those rows by an object-level* T*, completed boundary compactification, physical horizon, or global metric oracle.*

Proof. Read the geometric rows first. The carrier exposes disk source D, hyperbolic metric row M, geodesic-boundary row B, boundary-shadow row S, and boundary-fixed transport row T as separate finite fields.

Apply the obligation surface. By Definition 3866.2, a local NameCert read can inspect only those rows plus refusal, anchor, transport, replay, provenance, and local naming.

The excluded objects have no carrier coordinate. Since F explicitly refuses global-geometry substitution and H, C, P, N only preserve the displayed route, no accepted read can produce object-level T, completed compactification, physical horizon, or a metric oracle outside the packet. □

Theorem 2893.4 (Hyperbolic visual metric sibling separation)**.** *The* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *carrier is not bijective, row for row, with any one of* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*, or* $\mathsf{DynSystem}^{\uparrow}$*. A bijection preserving local NameCert obligations would have to preserve simultaneous disk readout, metric read, boundary incidence, shadow, boundary-fixed transport, refusal, provenance, transport, replay, and local naming, and no listed sibling carries that whole surface.*

Proof. Compare first with $\mathsf{HyperbolicMetric}^{\uparrow}$. The metric sibling supplies a disk metric read, but it does not carry boundary-geodesic incidence, boundary shadow, boundary-fixed transport, and global-geometry refusal as simultaneous rows.

Compare next with $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ and $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$. These siblings supply boundary incidence and shadow reads, while neither also owns the metric read, boundary-fixed transport, and visual refusal row.

Finally compare with $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ and $\mathsf{DynSystem}^{\uparrow}$. Transport and flow rows can move or evolve displayed data, but they are not the full visual metric packet. Since each sibling omits at least one of D, M, B, S, T, F, A, no row-preserving bijection can preserve the $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ obligations. □

Falsifiable prediction: A visual metric read is refused whenever the packet lacks disk source, metric read, boundary shadow, transport, and provenance rows.

Independence witness: hyperbolic_metric, hyperbolic_geodesic_boundary, hyperbolic_boundary_shadow, poincare_boundary_fixed_transport, dynsystem: the visual metric packet carries the simultaneous readout route not bijective with any one sibling.

The Lean TasteGate site for $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ HyperbolicVisualMetric}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicVisualMetric}^{\uparrow}$ instances. These instances keep disk source, metric read, boundary row, shadow row, boundary-fixed transport, refusal, provenance, transport, replay, and local naming as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ packet is generated from a disk source row, HyperbolicMetricUp metric read, HyperbolicGeodesicBoundaryUp boundary row, HyperbolicBoundaryShadowUp shadow row, PoincareBoundaryFixedTransportUp transport, global-geometry refusal, vision-anchor provenance, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3866.1, Definition 3866.2, Definition 3866.3, and Definition 3866.4 seed the finite visual-metric route over disk, metric, boundary, shadow, transport, refusal, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close object-level T, completed boundary compactification, physical horizons, global metric oracles, quotient boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual metric obligation routing, global-geometry refusal, sibling separation, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming.

Chapter 2894

A Concrete Naming Certificate for $\mathsf{MetricCauchyNetCluster}^{\uparrow}$

Following Moore–Smith and Kelley net convergence, and the Bishop–Bridges treatment of metric completion by displayed Cauchy information, the $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ packet records the finite route by which a metric Cauchy net exposes cluster evidence before any completion or real-seal consumer reads it. It sits between the subnet route of chapter 1592, the Cauchy-net completion boundary of chapter 3794, the metric completion packet of chapter 1506, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a displayed cluster route for completion-facing reads; it is not quotient nets, selected cluster points, countable choice, host limit equality, or ambient completeness.

Definition 2894.1 (Metric Cauchy-net cluster carrier)**.** *A* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, M, J, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-net row, recording the inspected finite directed indices and the local comparison order used by the Moore–Smith route.*
2. *W is the Cauchy-window row, listing the finite cofinal windows and the tolerance requests that must be met before any cluster read is exposed.*
3. *M is the metric completion boundary row, connecting the displayed net windows to the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer surfaces.*
4. *J is the cluster witness row, recording only finite witness windows and not a selected cluster point.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that opens the finite observation windows used by W and J.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational distance and tolerance comparisons inspected along the route.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and readbacks.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the directed-net, Cauchy-window, cluster-witness, stream, regular-readback, dyadic, and metric-completion rows have been displayed.*

9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a completion-facing request through $D, W, M, J, S, R, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient nets, chosen cluster points, choice-selected subnets, host metric completion equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricCauchyNetClusterUp.MetricCauchyNetClusterTasteGate single carrier alignment`

Theorem 2894.2 (Metric Cauchy-net cluster NameCert obligations)**.** *For every accepted* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier $K = (D, W, M, J, S, R, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the directed-net row, Cauchy-window row, metric-completion boundary, cluster witness, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-window stability, requiring every inspected directed index to be read through D before W can answer a Cauchy request;*
3. *cluster-witness locality, requiring J to expose only finite witness windows tied to the same D, W, S, T rows;*
4. *completion-boundary handoff, requiring M to mediate every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer before E is read;*
5. *regular and dyadic accounting, requiring R and T to record the rational distance and tolerance ledger used by the same finite windows;*
6. *structural exactness through H, C, P, N, with no quotient-net equality, selected cluster point, countable choice, host real equality, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 3867.1. The packet exposes the directed-net row, Cauchy-window row, metric boundary, cluster witness, finite windows, regular readback, dyadic ledger, terminal seal, and the structural rows as one finite object.

The directed and Cauchy-window obligations are read first. The row D fixes the finite directed comparisons, and W answers tolerance requests only through those displayed comparisons and the finite stream windows S.

The cluster witness row J is then local to the same windows. It records witness windows rather than a selected point, while R and T record the rational distance readback and dyadic tolerance ledger used by the window.

The completion boundary M mediates the completion-facing read before the Real seal row E is available. Since H, C, P, N only transport, replay, source, and name the same rows, the listed obligations exhaust the NameCert surface. □

Theorem 2894.3 (Metric Cauchy-net cluster completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier $K = (D, W, M, J, S, R, T, E, H, C, P, N)$ factors*

through the displayed finite Cauchy-net cluster route

$$D \longmapsto W \longmapsto S, J \longmapsto R, T \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

In particular, no $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *seal can be consumed before the directed-net row, Cauchy-window row, finite cluster witness,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been displayed.*

Proof. Begin with a completion-facing consumer. By Definition 3867.1, the only completion boundary in the packet is the row M, and the only terminal real-facing seal is E.

Apply the obligation surface of Definition 3867.2. It places directed-net admission and Cauchy-window stability before any cluster witness is read, and it ties the witness row to the same finite stream windows.

The regular rational and dyadic rows then supply the metric tolerance accounting used by the finite windows. The completion boundary M is downstream of that accounting, and the seal row E is downstream of M.

Finally H, C, P, N preserve and name only the displayed route. Quotient nets, selected cluster points, countable choice, host real equality, detached completion objects, and ambient completeness have no carrier coordinate, so they cannot replace the finite route. □

Falsifiable prediction: Every accepted $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ completion-facing read factors through a directed-net row, Cauchy-window row, finite cluster witness, StreamName windows, RegSeqRat readback, DyadicRatCore ledger, MetricCompletion and CauchyNetCompletion boundary, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchysubnet, cauchynetcompletion, metriccompletion, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ because this packet jointly carries the directed-net row, Cauchy-window row, finite cluster witness, metric completion boundary, finite stream windows, rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ packet is generated from a directed-net row, Cauchy-window row, metric completion boundary, finite cluster witness, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3867.1, Definition 3867.2, and Definition 3867.3 seed the finite Cauchy-net cluster route from directed windows through completion and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected cluster points, countable choice, host real equality, detached completion objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window stability, cluster-witness locality, completion-boundary handoff, regular and dyadic accounting, structural exactness, and local naming for BEDC.Derived.MetricCauchyNetCl

Chapter 2895

A Concrete Naming Certificate for $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ packet records the constructive interval route from closed bounded interval data to a compact-uniform modulus extraction. It joins the endpoint surface of chapter 1591, the finite interval-net route of chapter 1250, the positive radius ledger of chapter 452, the compact-continuity rows of chapter 388, and the output consumer of chapter 369. The packet names the finite bridge data only; it does not assert open-cover compactness, countable choice, quotient stream equality, ambient real completeness, Lean verification, or bridge checking.

Definition 2895.1 (Compact interval modulus bridge carrier)**.** *A* $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, N, R, U, M, S, Q, H, C, P, L)$$

with the following displayed rows:

1. *E is the closed bounded interval endpoint row from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*, including the lower endpoint, upper endpoint, and displayed containment boundary.*
2. *N is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-net row, carrying dyadic cells, rational cell readback, displayed centers, finite coverage,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows.*
3. *R is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius row, read only as a finite positive-radius ledger attached to the displayed net and cover route.*
4. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *compact-continuity row whose pointwise continuity data are already routed through the interval net and radius ledger.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row consumed after E, N, R, U have been displayed.*
6. *S records the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows used by interval endpoints, net centers, and the final modulus handoff.*
7. *Q is the tolerance-query row whose requested precision is routed through the net, radius, compact-continuity, and output-modulus rows.*
8. *H records componentwise* hsame *transport for $E, N, R, U, M, S, Q, C, P, L$.*

9. *C records displayed* Cont *replay from the closed interval source through finite nets, finite-radius data, compact-continuity rows, and the uniform-modulus output.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNe}$ $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, countable choice, quotient real equality, hidden host continuity, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CompactIntervalModulusBridgeUp.CompactIntervalModulusBridgeTasteGate`
`single carrier alignment`

Theorem 2895.2 (Compact interval modulus bridge NameCert obligations)**.** *For the carrier of Definition 3868.1, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$ *surface consists exactly of endpoint exposure through E, finite interval-net exposure through N, finite positive-radius exposure through R, compact-continuity exposure through U, uniform-modulus output through M,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal exposure through S, tolerance-query routing through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the accepted carrier $B = (E, N, R, U, M, S, Q, H, C, P, L)$. The endpoint row E supplies the closed bounded interval source, while N supplies the finite interval-net route over the same displayed endpoint boundary.

Read the finite-radius and compact-continuity rows next. The row R is the positive-radius ledger supplied by the finite Lebesgue-number surface, and U is the compact-continuity row that consumes the interval net and radius data before any modulus output is exposed.

The modulus row M is therefore downstream of E, N, R, U, Q. The seal row S records the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readbacks used by endpoints, centers, and the final handoff.

Finally H, C, P, L transport, replay, package, and name exactly these rows. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ uniform-modulus read must pass through displayed closed bounded interval endpoints, a finite closed-bounded interval net, a finite Lebesgue-number radius row, compact-uniform-continuity rows, StreamName, RegSeqRat, and RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closedboundedinterval, closedboundedintervalnet, finitelebesguenumber, compactuniformcontinuity, uniformmodulus: $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, finite-net rows, finite positive-radius rows, compact-continuity rows, the UniformModulus output, StreamName/RegSeqRat/RealUp seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ packet is generated from closed bounded interval endpoints, finite interval-net rows, a finite Lebesgue-number radius ledger, compact-continuity rows, a UniformModulus output row, StreamName, RegSeqRat, and RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3868.1 and Definition 3868.2 seed the finite bridge from closed bounded interval data to compact-uniform modulus extraction.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, finite-net exposure, radius exposure, compact-continuity exposure, uniform-modulus output, seal rows, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalMo

Chapter 2896

A Concrete Naming Certificate for $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ packet records the finite search surface obtained by iterating bisection over a located real interval while keeping every branch, sign read, and retained window visible. It sits after the real-name surface of chapter 38, the located interval carrier of chapter 1709, the located bisection route of chapter 1711, the finite bracketing interval packet of chapter 1807, and the dyadic intermediate-value packet of chapter 2203. The packet names a bounded bisection search, not a host interval equality, not trichotomy on arbitrary reals, not choice of a hidden root, and not ambient completeness.

Definition 2896.1 (Located real bisection-search carrier)**.** *A* $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, B, F, D, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, exposing lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, a locatedness modulus, and the inspected finite interval budget.*
2. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *branch route, recording the requested finite bisection depth and the retained child at each stage.*
3. *F is the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket row, preserving the displayed sign or locator bracket for the retained interval.*
4. *D is the* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *dyadic sign-read row used by the bounded search.*
5. *Q is the query row, recording the finite search request, target tolerance, and admissible stopping condition.*
6. *W is the retained-window ledger, listing the lower and upper windows produced by the bounded branch sequence.*
7. *R is the* $\mathsf{Real}^{\uparrow}$*-facing seal row reached only after I, B, F, D, Q, W have been exposed.*
8. *H records componentwise* hsame *transport for the source, branch, bracket, dyadic read, query, window, and seal rows.*
9. *C records displayed* Cont *replay from a search request through finite bisection, bracket preservation, dyadic sign readback, retained-window exposure, and Real-facing seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{FiniteBracketingInterval}^{\uparrow}$, $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealBisectionSearch}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $I, B, F, D, Q, W, R, H, C, P, N$. It has no coordinate for host interval equality, trichotomy, countable choice, selected quotient representatives, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden root witness.*

Lean4 ✓ `BEDC.Derived.LocatedRealBisectionSearchUp.LocatedRealBisectionSearchTasteGate single carrier alignment`

Theorem 2896.2 (Located real bisection-search NameCert obligations)**.** *For every accepted* LocatedRealBisectionS *carrier $S = (I, B, F, D, Q, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealBisectionSearch}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located-source exposure through I, finite bisection exposure through B, bracket preservation through F, dyadic sign readback through D, bounded query admission through Q, retained-window exposure through W, Real-facing seal routing through R, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3869.1. The accepted packet contains only the displayed rows, so carrier admission fixes the source interval, branch route, bracket row, dyadic sign readback, query, retained windows, seal, and structural data before any consumer reads the search.

The analytic part of the route is finite. The located source is I, the bisection route is B, the bracket row is F, and the dyadic sign-read row is D. The query row Q fixes the bounded stopping condition, while W records only the retained windows produced by those rows.

Finally read R, H, C, P, N. These rows seal the route toward $\mathsf{Real}^{\uparrow}$, transport the displayed data componentwise, replay the same finite request, record provenance, and name the packet. Since no carrier coordinate stores host equality, trichotomy, choice, ambient completeness, or a hidden root witness, the listed rows exhaust the local NameCert surface. □

Theorem 2896.3 (Located real bisection-search finite route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or interval-search consumer of an accepted* $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{LocatedIntervalBisection}^{\uparrow} \longmapsto \mathsf{FiniteBracketingInterval}^{\uparrow} \longmapsto \mathsf{DyadicIntermediateValue}^{\uparrow} \longmapsto$$

The consumer receives only the bounded branch ledger, preserved bracket, dyadic sign reads, retained windows, Real-facing seal, transport, replay, provenance, and local naming.

Proof. Begin with the source row. A search request first reads I, so the packet cannot replace the located real interval by an ambient host interval.

Next read the finite search rows B, F, D, Q, W. They record the branch sequence, bracket preservation, dyadic sign readback, bounded stopping request, and retained windows. No row in this stage selects an infinite limit or an unbounded root.

The final route through R, H, C, P, N preserves the same finite ledger for downstream real and interval-search consumers. Because the carrier has no coordinate for host interval equality, trichotomy, quotient-real representatives, countable choice, ambient completeness, or hidden root selection, every such consumer remains on the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealBisectionSearch}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ packet is generated from a located real interval source row, located bisection branch route, finite bracketing row, dyadic sign readback, bounded query row, retained-window ledger, Real-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedRealBisectionSearch}\uparrow}$, seed)

Scope closed: Definition 3869.1, Definition 3869.2, and Definition 3869.3 seed the finite bisection-search packet over located real interval, finite bracket, dyadic sign, retained-window, and Real-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedRealBisectionSearch}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, real trichotomy, countable choice, selected quotient representatives, ambient RealUp completeness, hidden root selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, branch-route exactness, bracket preservation, dyadic sign readback, retained-window routing, RealUp seal nonescape, and local naming for BEDC.Derived.LocatedRealBisectionSearchUp.

Chapter 2897

A Concrete Naming Certificate for $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$

The $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ packet realizes the conditional trigger ensemble of Definition 9462.1594 and Definition 9462.1601 as finite weighted trigger, blocker, partition, expectation, transport, replay, provenance, and local NameCert rows. It sits beside the residue boundary of chapter 1167, the trigger-blocker duality packet of chapter 3801, the trigger-energy partition packet of chapter 3806, and the weighted deformation spine adjacent to the DNA window-six boundary-tile vision. The packet names a finite conditional ensemble surface; it is not a hidden biological semantics, not an external probability space, not an unbounded trigger search, and not an oracle for conditional means.

Definition 2897.1 (Conditional trigger ensemble carrier)**.** *A* $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, B, Q, Z, M, L, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the weighted trigger row, recording the finite trigger support and activity weights used by the conditional ensemble.*
2. *B is the blocker row, recording the blocker constraints inherited from* $\mathsf{TriggerBlockerDuality}^{\uparrow}$*.*
3. *Q is the partition row, recording the energy partition inherited from* $\mathsf{TriggerEnergyPartition}^{\uparrow}$*.*
4. *Z is the conditional normalization ledger, storing the finite partition function and the visible event boundary.*
5. *M is the mean row, storing the finite cardinality and energy expectations computed from Z.*
6. *L is the low-energy limit row, recording the partition-minimal trigger polynomial exposed by the source theorem.*
7. *A is the adjacency row tying the ensemble to finiteobservationresidueboundary, triggerblockerduality, triggerenergypartition, and weighted-deformation-spine-adjacent siblings.*
8. *H records componentwise* hsame *transport for trigger, blocker, partition, normalization, mean, limit, adjacency, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a conditional trigger query through weighted support, blocker filtering, energy partition, expectation readback, and low-energy limiting readback.*

10. *P is the* Pkg *provenance over* ConditionalTriggerEnsemble$^{\uparrow}$, FiniteObservationResidueBoundary$^{\uparrow}$, TriggerBlockerDuality$^{\uparrow}$, TriggerEnergyPartition$^{\uparrow}$, *the conditional trigger ensemble vision,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ConditionalTriggerEnsemble}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $T, B, Q, Z, M, L, A, H, C, P, N$. It has no coordinate for hidden codon semantics, an external probability space, an unbounded trigger search, a selected biological interpretation, or a host analytic probability theorem.

Lean4 (definition) `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleUp`

Theorem 2897.2 (Conditional trigger ensemble NameCert obligations)**.** *For every accepted* ConditionalTriggerEnse *carrier $E = (T, B, Q, Z, M, L, A, H, C, P, N)$, the local* NameCert$_{\text{ConditionalTriggerEnsemble}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, weighted trigger exposure through T, blocker exposure through B, energy-partition exposure through Q, conditional normalization through Z, expectation readback through M, low-energy limit readback through L, sibling adjacency through A, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3870.1. The accepted packet exposes only weighted triggers, blockers, energy partitions, normalization, means, low-energy limits, adjacency, and the structural rows.

The ensemble calculation is finite. The trigger row T, blocker row B, partition row Q, normalization row Z, mean row M, and low-energy row L are all displayed before any consumer receives an ensemble readback.

The adjacency and structural rows A, H, C, P, N preserve that same finite packet for downstream readers. Since no coordinate stores hidden codon semantics, an external probability space, unbounded search, or a selected biological interpretation, the listed rows exhaust the local NameCert surface. □

Theorem 2897.3 (Conditional trigger ensemble sibling route)**.** *Every conditional trigger, blocker, expectation, or low-energy consumer of an accepted* ConditionalTriggerEnsemble$^{\uparrow}$ *packet factors through the displayed route*

$$\text{FiniteObservationResidueBoundary}^{\uparrow},\ \text{TriggerBlockerDuality}^{\uparrow},\ \text{TriggerEnergyPartition}^{\uparrow} \longmapsto T, B, Q, Z, M, L, A, H, C, P$$

The route preserves finite weighted support, blocker constraints, energy partition, conditional normalization, expectation readback, low-energy readback, sibling adjacency, transport, replay, provenance, and local naming.

Proof. Start with the sibling rows. The finite observation residue boundary keeps the observed support and residue boundary visible, while trigger-blocker duality and trigger-energy partition supply the blocker and partition rows.

Next read the ensemble rows T, B, Q, Z, M, L. They expose weighted triggers, blockers, partitions, normalization, finite means, and low-energy limiting data without importing an external probability object.

Finally use A, H, C, P, N. These rows record sibling adjacency, transport the finite packet, replay the conditional query, source it to the vision theorem, and name the local certificate. No excluded semantic or probabilistic oracle is present in the carrier. □

Falsifiable prediction: An accepted $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ read is refuted if a consumer obtains a conditional mean, energy mean, or low-energy trigger law without displaying finite weighted triggers, blocker rows, partition rows, normalization, transport, replay, provenance, and local naming.

Independence witness: finiteobservationresidueboundary, triggerblockerduality, triggerenergypartition, weighted-deformation-spine-adjacent: none is bijective to $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ because this packet jointly carries weighted trigger support, blocker filtering, energy partition, conditional normalization, expectation readback, low-energy limiting readback, sibling adjacency, transport, replay, provenance, and local naming.

The Lean follow-up for this packet must provide `Nontrivial ConditionalTriggerEnsembleUp` and `FieldFaithful ConditionalTriggerEnsembleUp` instances in `BEDC.Derived.ConditionalTriggerEnsembl` so that $T, B, Q, Z, M, L, A, H, C, P, N$ remain separated as displayed fields.

Lean4 ✓ `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}$.

Constructive story: A $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ packet is generated from finite weighted trigger rows, blocker rows, energy-partition rows, conditional normalization, expectation readback, low-energy limiting readback, sibling adjacency, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3870.1, Definition 3870.2, and Definition 3870.3 seed the finite conditional trigger ensemble over weighted trigger, blocker, partition, expectation, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close hidden biological semantics, external probability spaces, unbounded trigger search, selected biological interpretation, host analytic probability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weighted-trigger exposure, blocker exposure, partition exposure, normalization exactness, expectation readback, Nontrivial ConditionalTriggerEnsembleUp, FieldFaithful ConditionalTriggerEnsembleUp, and local naming for BEDC.Derived.ConditionalTriggerEnsembleUp.

Chapter 2898

A Concrete Naming Certificate for NormalizationEvidenceBudget$^{\uparrow}$

The NormalizationEvidenceBudget$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization boundary of chapter 9383 as finite budget data inspected before any SN read. It sits beside the frontier of chapter 892, the router of chapter 1108, the closed-substitution boundary of chapter 527, the authorized generator route of chapter 525, and the compiler-recognizer flow of chapter 1048. The packet names bounded candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy, closed-term projection, no-infinite-reduction projection, transport, replay, provenance, and local naming rows; it does not claim evidence-free strong normalization, global closed consistency, erased subject reduction, host evaluation soundness, quotient syntax equality, Lean verification, or bridge checking.

Definition 2898.1 (Normalization evidence budget carrier)**.** *A* NormalizationEvidenceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$$

with the following displayed rows:

1. *A is the audit-map anchor row for chapter 9383.*
2. *C is the candidate-evidence budget row, listing the finite candidate witnesses available for the read.*
3. *K is the closed-candidate budget row, listing the finite closed-candidate wrappers and their closed typed term boundaries.*
4. *S is the candidate-set budget row, listing finite membership, soundness, and beta-closure evidence.*
5. *P is the Pi-adequacy row for function-side candidate use.*
6. *R is the application-adequacy row for argument and application candidate use.*
7. *J is the closed-term projection row consumed through* ClosedTermSubstitutionBoundary$^{\uparrow}$*.*
8. *I is the no-infinite-reduction projection row read only after candidate, closed-candidate, or candidate-set evidence is present.*
9. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *retention row keeping candidate evidence, finished-normal evidence, and subject-reduction obstruction visible.*

10. *G is the* AuthorizedGeneratorRecursor$^{\uparrow}$ *generator handoff row.*
11. *D is the* GroundCompilerRecognizerFlow$^{\uparrow}$ *compiler-recognizer row.*
12. *H records componentwise* hsame *transport for $A, C, K, S, P, R, J, I, F, G, D, T, Q, N$.*
13. *T records displayed* Cont *replay from evidence budgets through closed-term SN and no-infinite-reduction reads.*
14. *Q is the* Pkg *provenance over* NormalizationEvidenceBudget$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, GroundCompilerRecognizerFlow$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* NameCert$_{\text{NormalizationEvidenceBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a closed-term SN read outside C, K, S, no application-normalization read outside P, R, and no no-infinite-reduction read outside I.

Lean4 ✓ `BEDC.Derived.NormalizationEvidenceBudgetUp.NormalizationEvidenceBudgetTasteGate single carrier alignment`

Theorem 2898.2 (Normalization evidence budget NameCert obligations)**.** *For the carrier of Definition 3871.1, the local* NameCert$_{\text{NormalizationEvidenceBudget}^{\uparrow}}$ *surface consists exactly of audit-map anchoring through A, candidate evidence through C, closed-candidate evidence through K, candidate-set evidence through S, Pi and application adequacy through P, R, closed-term projection through J, no-infinite-reduction projection through I, frontier retention through F, authorized-generator handoff through G, compiler-recognizer handoff through D, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through Q, and local naming through N.*

Proof. Project the accepted carrier $E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$. The audit anchor A fixes the route to the candidate-mediated map, and the evidence rows C, K, S list exactly the candidate, closed-candidate, and candidate-set data available before any closed-term SN read.

The adequacy rows P, R keep function, argument, and application cases inside the displayed candidate route. The projection rows J, I are downstream: J records closed-term projection, while I records the no-infinite-reduction consequence only after evidence is present.

The sibling rows F, G, D retain the normalization frontier, authorized generator handoff, and compiler-recognizer flow as visible budget provenance. They route evidence; they do not create evidence-free normalization.

Finally H, T, Q, N transport, replay, package, and name the same displayed rows. Since the carrier exposes no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2898.3 (Normalization evidence-budget refusal route)**.** *Let*

$$E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$$

be an accepted NormalizationEvidenceBudget$^{\uparrow}$ *packet. If the candidate-evidence row C, closed-candidate row K, and candidate-set row S are absent from the accepted budget surface, then no closed-term strong-normalization read can pass through J and no no-infinite-reduction read can pass through I. The refusal is a local budget refusal along the MetaCIC normalization audit map boundary of chapter 9383; it does not assert evidence-free strong normalization.*

Proof. Project the carrier of Definition 3871.1. The only rows that can support a normalization read before the projections are the displayed candidate budget C, closed-candidate budget K, and candidate-set budget S.

Apply the NameCert obligations of Definition 3871.2. They place the closed-term projection J and the no-infinite-reduction projection I downstream of the evidence rows and route the remaining frontier, generator, compiler, transport, replay, provenance, and naming rows through the same finite budget.

If C, K, S are absent, the projected route has no evidence source from which J or I can be admitted. The audit-map boundary of chapter 9383 therefore records a refusal of the read, not a proof of evidence-free strong normalization. □

Falsifiable prediction: A closed-term SN read through $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ is refused when the displayed budget contains no candidate evidence, no closed-candidate evidence, and no candidate-set evidence.

Independence witness: metacicnormalizationfrontier, metacicnormalizationcertificaterouter, closedtermsubstitutionboundary, authorizedgeneratorrecursor, groundcompilerrecognizerflow: $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate, closed-candidate, candidate-set, Pi/app adequacy, closed-term projection, no-infinite-reduction, frontier, generator, compiler-recognizer, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ must provide explicit Nontrivial NormalizationEvidenc and FieldFaithful $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ instances. These instances preserve the finite evidence-budget rows as distinguishable fields before SN and no-infinite-reduction reads are admitted.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ packet is generated from an audit-map anchor, candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy, closed-term projection, no-infinite-reduction projection, frontier retention, authorized-generator handoff, compiler-recognizer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3871.1 and Definition 3871.2 seed the finite evidence-budget carrier for candidate-mediated normalization reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close evidence-free strong normalization, global closed consistency, erased subject reduction, host evaluation soundness, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-budget refusal, projection exactness, frontier retention, generator and compiler handoff, TasteGate nontriviality, and field-faithfulness for BEDC.Derived.NormalizationEvidenceBudg

Chapter 2899

A Concrete Naming Certificate for $\mathsf{RationalIntervalSubdivision}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ packet records the finite subdivision of a rational interval into ordered rational subintervals while keeping the endpoint, mesh, and handoff rows visible. It sits after the rational interval carrier of chapter 354, the dyadic tolerance core of chapter 291, and the closed bounded interval surface of chapter 1591. The packet names a finite rational partition and its local certificate rows; it is not an open-cover compactness principle, a choice of refinement from an infinite family, quotient interval equality, or ambient completeness.

Definition 2899.1 (Rational interval subdivision carrier)**.** *A* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, E, D, M, W, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{RationalInterval}^{\uparrow}$ *row, exposing rational endpoints $a \leq b$ and the interval-order classifier.*
2. *E is the finite ordered endpoint list $a = x_0 \leq x_1 \leq \cdots \leq x_m = b$.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh ledger, recording the dyadic radii, width comparisons, and refinement rows used to bound each adjacent interval.*
4. *M is the mesh row assigning to each adjacent pair (x_i, x_{i+1}) its dyadic width certificate.*
5. *W is the componentwise window row exposing the subinterval sequence $([x_i, x_{i+1}])_{i<m}$ through the same finite endpoint list.*
6. *K is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *handoff row, reached only after I, E, D, M, W have been displayed.*
7. *H records componentwise* hsame *transport for the source interval, endpoints, dyadic mesh, subinterval windows, closed-interval handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a subdivision request through endpoint inspection, dyadic mesh checking, subinterval-window exposure, and closed-interval handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $I, E, D, M, W, K, H, C, P, N$. It has no coordinate for open-cover compactness, infinite refinement choice, quotient interval equality, selected real representatives, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RationalIntervalSubdivisionUp.RationalIntervalSubdivisionTasteGate single carrier alignment round trip`

Theorem 2899.2 (Rational interval subdivision NameCert obligations)**.** *For every accepted* RationalIntervalSubdivi *carrier $S = (I, E, D, M, W, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$ *surface consists exactly of source-interval exposure through I, finite ordered endpoint admission through E, dyadic mesh exactness through D, adjacent-width certification through M, subinterval-window exposure through W, closed bounded interval handoff through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3872.1. The accepted packet exposes one rational source interval, one finite ordered endpoint list, one dyadic mesh ledger, one adjacent-width row, one subinterval-window row, one closed-interval handoff, and the structural rows.

The subdivision data are finite. The row I supplies the enclosing rational interval, while E lists every endpoint used by the subdivision. The dyadic ledger D and mesh row M check only adjacent pairs from this list, so no open cover, infinite refinement, or selected limiting partition enters the packet.

The consumer-facing handoff is then forced through W and K. The row W exposes the subinterval windows componentwise, and K packages the same rows for closed bounded interval consumers. Finally H, C, P, N transport, replay, source, and name exactly this displayed surface. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2899.3 (Rational interval subdivision closed-interval handoff)**.** *Every* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *or finite-net consumer of an accepted* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{RationalInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{ClosedBoundedInterval}^{\uparrow}.$$

Concretely, the consumer receives the source rational interval row, the ordered rational endpoint list, the dyadic mesh ledger, the adjacent-width row, the componentwise subinterval windows, the closed-interval handoff, transport, replay, provenance, and local naming. It receives no open-cover compactness witness, choice-selected refinement, quotient interval equality, selected real representative, or ambient completeness coordinate.

Proof. Begin with an accepted carrier $S = (I, E, D, M, W, K, H, C, P, N)$. Any subdivision read first inspects I and E, so the enclosing rational interval and every subdivision endpoint are part of the displayed packet before a consumer can read a subinterval.

Next the read passes through D and M. These rows certify the dyadic mesh and the adjacent rational widths componentwise; they do not select a refinement from an infinite family and they do not invoke open-cover compactness.

The finite-net or closed-interval consumer can then read W and K. The row W lists the component subinterval windows, and K is exactly the closed bounded interval handoff built from those finite rows. The structural rows H, C, P, N preserve this same route under hsame transport and Cont replay, so the consumer has no route to quotient interval equality, selected real representatives, choice, or ambient completeness beyond the displayed packet. □

Falsifiable prediction: An accepted $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ consumer that reaches a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ handoff or finite-net read without first inspecting the source $\mathsf{RationalInterval}^{\uparrow}$ row, ordered rational endpoint list, $\mathsf{DyadicRatCore}^{\uparrow}$ mesh ledger, componentwise subinterval windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row is outside the packet.

Independence witness: rationalinterval, dyadicratcore, closedboundedinterval: $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source rational interval, a finite ordered endpoint list, dyadic mesh certificates, componentwise subinterval windows, a closed-interval handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ packet is generated from a RationalInterval source row, finite ordered rational endpoints, a DyadicRatCore mesh ledger, adjacent-width rows, componentwise subinterval windows, a ClosedBoundedInterval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3872.1, Definition 3872.2, and Definition 3872.3 seed the finite rational subdivision route from rational interval rows through dyadic mesh data to closed bounded interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, infinite refinement choice, quotient interval equality, selected real representatives, finite-net compactness, or ambient completeness.

Upgrade path: Obligation closure requires checked source interval exposure, endpoint ordering, dyadic mesh exactness, subinterval-window routing, closed bounded interval handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalIntervalSubdivisionUp.

Chapter 2900

A Concrete Naming Certificate for $\mathsf{UniformSequentialLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformSequentialLimit}^{\uparrow}$ packet records the standard passage from uniform convergence of a sequence of real-valued observations to sequential real-limit consumers as a finite BEDC route. It sits beside the uniform-tail certificate of chapter 3417, the sequence-source surface of chapter 3857, the sup-norm comparison boundary of chapter 3807, the finite stream and regular rational readback rows of chapter 126 and chapter 301, and the terminal real seal of chapter 38. The chapter names the finite handoff by which a uniform tail schedule may be consumed by Real-limit readers without selecting a limit function, quotienting functions, invoking countable choice, asserting ambient completeness, or importing open-cover compactness.

Definition 2900.1 (Uniform sequential limit carrier)**.** *A* $\mathsf{UniformSequentialLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row. It records the accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *observations and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows from which the displayed sequence entries are read.*
2. *M is the uniform tail schedule row. It is read through the* $\mathsf{UniformConvergence}^{\uparrow}$ *packet and assigns one displayed tail bound to each accepted tolerance request.*
3. *V is the sup-norm comparison row. It consumes the* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *boundary used to compare all displayed source windows under the same tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations exposed by S, M, V.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the sequence source, uniform tail schedule, sup-norm comparison, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for S, M, V, R, E, C, P, N.*
7. *C records displayed* Cont *replay from a sequential Real-limit request through the source, tail, comparison, readback, seal, transport, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformSequentialLimit}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{LocatedSequence}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, M, V, R, E, H, C, P, N$. It has no coordinate for a selected limit, quotient equality of functions, countable choice, ambient completeness of $\mathsf{Real}^\uparrow$*, open-cover compactness, Lean verification, or a bridge credential.*

Lean4 ✓ `BEDC.Derived.UniformSequentialLimitUp.UniformSequentialLimitTasteGate single carrier alignment`

Theorem 2900.2 (Uniform sequential limit NameCert obligations)**.** *For every accepted* $\mathsf{UniformSequentialLimit}^\uparrow$ *carrier*

$$U = (S, M, V, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$ *surface consists exactly of carrier admission for the sequence source, uniform tail schedule, sup-norm comparison row, regular rational readback, Real seal, componentwise transport, displayed replay, package provenance, and local name. Every sequential* $\mathsf{Real}^\uparrow$*-limit consumer of U factors through*

$$\mathsf{LocatedSequence}^\uparrow \longmapsto \mathsf{UniformConvergence}^\uparrow \longmapsto \mathsf{ContinuousSupNorm}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as displayed by $S, M, V, R, E, H, C, P, N$.

Consequently the packet exports a finite uniform-limit handoff for Real-limit readers and exports no selected limit, quotient equality of functions, countable-choice schedule, ambient $\mathsf{Real}^\uparrow$ *completeness, or open-cover compactness principle.*

Proof. Project the accepted carrier through Definition 3874.1. The source row S supplies only displayed $\mathsf{LocatedSequence}^\uparrow$ observations and finite $\mathsf{StreamName}^\uparrow$ windows, while M records the uniform tail schedule supplied by the $\mathsf{UniformConvergence}^\uparrow$ boundary.

Read the comparison row V next. Its $\mathsf{ContinuousSupNorm}^\uparrow$ surface applies the same displayed tolerance to all source windows admitted by S and M, so the consumer receives a uniform comparison ledger rather than a pointwise host-limit selector.

Now read R and E. The regular rational observations pass through $\mathsf{RegSeqRat}^\uparrow$ before the $\mathsf{Real}^\uparrow$ seal is exposed, and the structural rows H, C, P, N transport, replay, source, and name exactly that route.

The excluded principles are absent from the carrier inventory. Therefore a sequential Real-limit consumer can use only the displayed finite uniform route and cannot extract a selected limit, quotient function equality, countable choice, ambient completeness, or open-cover compactness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 3874.1 and Definition 3874.2 seed the finite route from uniform convergence data and located sequence windows to sequential Real-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of functions, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform tail schedule stability, sup-norm comparison routing, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformSequentialLimitUp.

Constructive story: A $\mathsf{UniformSequentialLimit}^\uparrow$ packet is generated from a sequence source row, uniform tail schedule, sup-norm comparison row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2901

A Concrete Naming Certificate for $\mathsf{CauchyFilterConvergence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki–Kelley language of filters, the $\mathsf{CauchyFilterConvergence}^{\uparrow}$ packet records the finite route by which displayed Cauchy-filter data reaches a named limit boundary. It sits after the Cauchy-filter surface of chapter 333, the regular-filter handoff of chapter 1445, the uniform-window discipline of chapter 1694, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names convergence as a displayed BEDC route; it does not form a quotient of filters, choose a host limit, use countable choice, identify host real equality, or appeal to ambient completeness.

Definition 2901.1 (CauchyFilterConvergence carrier)**.** *A* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, B, M, S, D, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including displayed base packets, late-set witnesses, refinement routes, and Cauchy evidence.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *factorization row exposed by the same source filter data.*
3. *M is the directed meet row that names common refinements for requested filter tails.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read from the filter tails.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows and refinements.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the stream windows and dyadic rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, B, M, S, D, R have been read.*
8. *W is the convergence witness row, recording that every accepted neighbourhood read of E pulls back to a displayed tail of F through B, M, S, D, R.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records the displayed* Cont *replay from filter requests to basis tails, from tails to stream windows, from dyadic tolerances to regular rational readback, and from readback to the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterConvergence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$ *row.*

The carrier exposes convergence only through these rows. It has no coordinate for quotient filters, selected representatives, host equality of limits, countable-choice selectors, hidden cofinal tails, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterConvergenceUp.CauchyFilterConvergenceTasteGate single carrier alignment`

Theorem 2901.2 (CauchyFilterConvergence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier* $L = (F, B, M, S, D, R, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, CauchyFilter source exposure through F, basis factorization through B, directed tail refinement through M, StreamName window exposure through S, DyadicRatCore tolerance exactness through D, RegSeqRat readback through R, Real seal handoff through E, convergence pullback through W, and structural transport, replay, provenance, and local naming through H, C, P, N.*

No obligation may be discharged by quotient-filter equality, a selected host limit, countable choice, host real equality, a hidden cofinal tail, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3875.1. The rows $F, B, M, S, D, R, E, W, H, C, P, N$ are the only coordinates visible to the local certificate surface.

The source and basis side is finite. The Cauchy-filter row F supplies the displayed base packets and Cauchy evidence, while B and M expose the basis factorization and common directed refinements before any stream or real-facing read is available.

The readback side is ordered by the displayed route. The schedule S lists the finite windows, D attaches dyadic tolerances to the same windows, R reads regular rational data from them, and E is downstream from that readback.

Finally, the convergence witness W records only the pullback of Real-neighbourhood requests along the displayed route, and H, C, P, N transport, replay, source, and name the same data. Since the carrier contains none of the excluded coordinates, the listed rows exhaust the NameCert obligations. □

Theorem 2901.3 (CauchyFilterConvergence limit route)**.** *Let $L = (F, B, M, S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier. Every Real-facing convergence read factors through*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the read starts at the displayed Cauchy-filter row F, passes through the basis and directed-meet rows B, M, reads finite windows and tolerances through S, D, obtains the regular rational row R, reaches the Real seal E, and verifies convergence by pulling Real-neighbourhood requests back through W. The route exports no quotient filter, selected limit, countable-choice selector, host real equality, hidden cofinal tail, or ambient completeness coordinate.

Proof. Begin with a Real-facing convergence read of L. The Real seal coordinate is E, and by the carrier definition E is downstream from the source Cauchy-filter row, basis row, directed meet row, window schedule, dyadic ledger, and regular rational readback.

Use the basis and meet rows first. The rows B and M expose the finite CauchyFilterBasis route over F, so every requested convergence tail is a displayed row rather than an external cofinal family.

Next read the sequential data. The stream schedule S and dyadic ledger D determine the regular rational readback R, which is the only route from the filter-side packet to the Real seal.

The convergence witness W closes the route by pulling each accepted neighbourhood request of E back to the displayed filter tail. The structural rows H, C, P, N preserve that same route, and the absent quotient, choice, host-equality, hidden-tail, and completeness coordinates cannot be exported by any consumer. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterConvergence}^{\uparrow}$ convergence read factors through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterConvergence}^{\uparrow}$ packet is generated from a CauchyFilterUp row, a CauchyFilterBasisUp factorization row, a directed meet row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, a convergence pullback witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3875.1, Definition 3875.2, and Definition 3875.3 seed the constructive filter-to-limit route over CauchyFilterUp, CauchyFilterBasisUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected host limits, countable choice, host real equality, hidden cofinal tails, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, basis factorization, directed meet exposure, RegSeqRat readback, Real seal nonescape, convergence pullback, and TasteGate field faithfulness for BEDC.Derived.CauchyFilterConvergenceU

Chapter 2902

A Concrete Naming Certificate for UniformConvergenceCauchyCriterion$^{\uparrow}$

Following Bishop and Bridges on constructive uniform convergence and Cauchy criteria, the UniformConvergenceCauchyCriterion$^{\uparrow}$ packet records the finite means by which a displayed family of real-valued observations is certified uniformly Cauchy before any completion or uniform-limit consumer reads it. It sits between the uniform-limit handoff of chapter 3874, the common refinement mesh of chapter 3832, and the fast criterion route of chapter 3825. The packet supplies later uniform-limit and completion-extension certificates with a finite Cauchy surface; it does not import host function limits, quotient equality of functions, countable choice, ambient completeness, or a selected limiting map.

Definition 2902.1 (Uniform convergence Cauchy criterion carrier)**.** *A* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, M, F, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform observation row. It records the accepted* UniformConvergence$^{\uparrow}$ *tail requests and the finite source windows through which the family is inspected.*
2. *M is the common mesh row supplied by* CauchyMeshRefinement$^{\uparrow}$*. It aligns all displayed source windows under one finite tolerance ledger before any readback occurs.*
3. *F is the fast criterion row supplied by* FastCauchyCriterion$^{\uparrow}$*. It exposes the monotone threshold discipline used to answer each requested precision.*
4. *Q is the Cauchy comparison row. It records that every pair of displayed source entries beyond the threshold named by F is compared through the common mesh named by M.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same windows, mesh, and fast threshold.*
6. *E is the terminal completion or* Real$^{\uparrow}$ *seal row reached only after U, M, F, Q, R have been displayed.*
7. *H records componentwise* hsame *transport for $U, M, F, Q, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a uniform-Cauchy request through the observation, mesh, fast-threshold, comparison, readback, seal, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* UniformConvergenceCauchyCriterion$^{\uparrow}$, UniformConvergence$^{\uparrow}$, UniformSequentialLimi CauchyMeshRefinement$^{\uparrow}$, FastCauchyCriterion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{UniformConvergenceCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, F, Q, R, E, H, C, P, N$. It has no coordinate for a host-space function limit, quotient equality of functions, countable-choice schedule, ambient completion theorem, or selected limit object.

Lean4 ✓ `BEDC.Derived.UniformConvergenceCauchyCriterionUp.UniformConvergenceCauchyCriterionTasteGate` `single carrier alignment`

Theorem 2902.2 (Uniform convergence Cauchy criterion NameCert obligations)**.** *For every accepted* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier*

$$K = (U, M, F, Q, R, E, H, C, P, N),$$

the local NameCert$_{\text{UniformConvergenceCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following five obligation families:*

1. *uniform source-window admission through U;*
2. *common finite mesh exposure through M;*
3. *fast threshold and pairwise Cauchy comparison through F, Q;*
4. *regular rational readback and terminal seal routing through R, E;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently every accepted packet exports a finite uniform-Cauchy criterion over displayed BEDC rows, not a host function limit, quotient equality of functions, countable-choice schedule, ambient completeness claim, or selected limiting map.

Proof. Project the carrier of Definition 3876.1. The row U exposes only the displayed uniform source windows, while M supplies the common finite mesh used by all later comparisons.

Read F and Q next. The fast threshold row gives the precision discipline, and the comparison row forces each post-threshold pair to be checked against the same finite mesh rather than against an unlisted limiting object.

Finally read R, E, H, C, P, N. The regular rational readback and terminal seal are downstream of the source, mesh, threshold, and comparison rows, and the structural rows transport and name exactly that route. Since the excluded analytic objects are absent from the carrier inventory, the five listed obligation families exhaust the certificate surface. □

Theorem 2902.3 (Uniform convergence Cauchy criterion completion handoff)**.** *Every uniform-limit or completion-extension consumer of an accepted* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier factors through the displayed route*

$$\text{UniformConvergence}^{\uparrow} \longmapsto \text{CauchyMeshRefinement}^{\uparrow} \longmapsto \text{FastCauchyCriterion}^{\uparrow} \longmapsto \text{RegSeqRat}^{\uparrow} \longmapsto \text{Real}^{\uparrow}.$$

The handoff may be consumed by UniformSequentialLimit$^{\uparrow}$ *and later completion certificates only as finite Cauchy data over $U, M, F, Q, R, E, H, C, P, N$. It does not supply host function limits, quotient equality of functions, countable choice, ambient completeness, or a selected limit map.*

Proof. Begin with a consumer request for a uniform-limit or completion-extension read. Carrier admission exposes U first, so the request sees only displayed uniform source windows.

The request must then pass through M, F, Q. The common mesh row aligns the source windows, the fast row gives the precision threshold, and the comparison row records the Cauchy checks performed after that threshold.

The only downstream readback is R, E, H, C, P, N. These rows package the regular rational observations, terminal seal, transport, replay, provenance, and local naming. Hence $\mathsf{UniformSequentialLimit}^{\uparrow}$ and completion-facing consumers receive finite criterion data and no extra analytic principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3876.1, Definition 3876.2, and Definition 3876.3 seed the finite uniform-Cauchy criterion route over UniformConvergence, CauchyMeshRefinement, FastCauchyCriterion, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformConvergenceCauchyCriterionUp.UniformConvergenceCauchyCriterionTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close host function limits, quotient equality of functions, countable choice, ambient completeness, selected limit maps, or bridge checking.

Upgrade path: Obligation closure requires checked source-window admission, common mesh exposure, fast threshold stability, Cauchy comparison routing, readback, terminal seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformConvergenceCauchyCriterionUp.

Constructive story: A $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ packet is generated from uniform source windows, a common CauchyMeshRefinement row, a FastCauchyCriterion threshold row, a pairwise Cauchy comparison row, RegSeqRat readback, terminal seal routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2903

A Concrete Naming Certificate for $\mathsf{BoxDimension}^{\uparrow}$

Following Falconer's finite-cover formulation of box dimension, the $\mathsf{BoxDimension}^{\uparrow}$ packet records the BEDC surface at which metric scale, finite covers, and counting ledgers become an inspectable dimension invariant. It is downstream of the metric carrier of chapter 57, the compact metric finite-net boundary of chapter 129, the refinement language of chapter 1853, and the rational dense route of chapter 1809. The chapter names the finite scale-cover data used by lower and upper box-dimension reads; it does not add Hausdorff measure, choice over optimal covers, host logarithms, quotient compacta, or a detached completion object.

Definition 2903.1 (Box dimension finite-cover carrier)**.** *A* $\mathsf{BoxDimension}^{\uparrow}$ *carrier over a compact metric source* X *is a finite* BHist *packet*

$$B = (M, K, S, C, R, D, N, H, P, L)$$

with the following displayed rows:

1. M *is the inherited* $\mathsf{Met}^{\uparrow}$ *distance row on* X*, including the point classifier and the visible distance-witness ledger.*
2. K *is the compact metric source row. It exposes the finite-net and complete-limit data from chapter 129 before any scale cover is read.*
3. S *is the dyadic scale row. Each accepted scale is a positive dyadic tolerance history, ordered by refinement toward smaller radii.*
4. C *is the finite cover row. For each scale in* S*, it displays a finite family of balls whose centers are carried by* X *and whose radii are bounded by the current dyadic scale.*
5. R *is the refinement ledger. It records how a cover at a smaller dyadic scale refines the previous visible cover without selecting an optimal cover in the host theory.*
6. D *is the counting and comparison ledger. It records the finite cardinality of each displayed cover and the rational bounds used by the lower and upper box-dimension comparison read.*
7. N *is the separability net route. It allows the centers of the displayed finite covers to be read through the* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window surface when the compact metric source is a real metric carrier.*

8. *H stores componentwise* hsame *transport for metric points, dyadic scales, cover centers, cover membership, refinement rows, counting comparisons, and dense-route data.*
9. *P is the* Pkg *provenance over* $\mathsf{BoxDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected minimal cover, ambient Euclidean embedding, host logarithm, Hausdorff-measure comparison, quotient compactum, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoxDimensionUp.BoxDimensionTasteGate single carrier alignment`

Theorem 2903.2 (Box dimension finite-cover obligations)**.** *For every accepted* $\mathsf{BoxDimension}^{\uparrow}$ *carrier*

$$B = (M, K, S, C, R, D, N, H, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$ *surface consists exactly of finite scale-cover obligations: carrier admission for the metric and compact rows, positive dyadic scale exposure, finite cover membership, radius comparison, refinement transport, cover-count comparison, separability-net routing when the source is Real-facing, provenance, and local naming. No obligation is stored outside* $M, K, S, C, R, D, N, H, P, L$.

Proof. Project the carrier of Definition 3877.1. The rows M and K provide the metric source and compact metric finite-net boundary, while S provides the positive dyadic scales at which covers are inspected.

For a fixed scale, C displays the finite cover as packet data: center histories, ball radii, membership rows, and radius bounds. The row D counts that finite family and stores the rational comparisons used by lower and upper box-dimension reads.

When the scale is refined, R is the only refinement witness. It transports a visible cover to a smaller visible cover through the recorded refinement ledger rather than by selecting a host-minimal cover.

The dense-source route N, the transport row H, the provenance row P, and the naming row L preserve the same finite data. Since the carrier contains no coordinate for optimal cover selection, host logarithms, Hausdorff measure, quotient compacta, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 2903.3 (Box dimension covering-dimension anchor)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoxDimension}^{\uparrow}$ *packet may read only the displayed finite cover and refinement rows C and R after the compact metric source K and dyadic scale row S have been exposed. The anchor sends scale-cover covers to covering-dimension refinement ledgers as finite packet data; it does not export a minimal cover, a Lebesgue-number principle, a Hausdorff-measure estimate, or an ambient Euclidean coordinate system.*

Proof. Start with the accepted carrier $B = (M, K, S, C, R, D, N, H, P, L)$. The compact metric row K is read before scale-cover data, so the consumer begins with the same finite-net and complete-limit boundary used by the compact metric certificate.

Expose a dyadic scale through S and the corresponding finite cover through C. These rows provide exactly the cover families, centers, radius bounds, and membership witnesses that a covering-dimension refinement argument may inspect.

Route the handoff through R. The refinement ledger records how smaller-scale covers refine the displayed cover family, matching the finite-cover language of chapter 1853 without introducing a host-selected optimal cover.

The counting row D and structural rows H, P, L remain bookkeeping for the same displayed covers. Because no row stores a Lebesgue-number theorem, Hausdorff-measure estimate, Euclidean embedding, or minimal-cover selector, the covering-dimension anchor is limited to the finite refinement surface. □

Theorem 2903.4 (Box dimension real-separability net route)**.** *When the compact metric source of an accepted* $\mathsf{BoxDimension}^{\uparrow}$ *packet is a Real-facing metric carrier, every scale-cover center read that reaches a* $\mathsf{RealSeparability}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RealSeparability}^{\uparrow}$$

before the center may be used as a dense-net witness. The route uses only the dyadic scale row S, finite cover row C, counting ledger D, separability route N, transport row H, provenance row P, and local name row L.

Proof. Project the box-dimension carrier. The row S supplies the dyadic scale, and the row C supplies the finite cover centers inspected at that scale.

Read those centers through the Real-facing dense route N. By chapter 1809, a separability consumer reaches Real data through dyadic tolerance, finite stream windows, regular rational readback, and the terminal Real seal before it may consume the local density surface.

The counting row D travels with the same finite cover family, so the number of displayed centers is the finite count used by the scale-cover comparison. The transport and provenance rows H, P, L preserve the route without adding a selected nearest point, host real equality, quotient stream, ambient completeness, or choice over balls.

Thus every Real-facing center read factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealSeparability}^{\uparrow}$ before it is available as dense-net evidence for the box-dimension packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3877.1, Definition 3877.2, Definition 3877.3, and Definition 3877.4 seed the finite scale-cover route for compact metric carriers, covering-dimension refinement ledgers, and RealSeparability dense-net consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff dimension, minimal-cover selection, host logarithm analysis, ambient Euclidean embeddings, quotient compacta, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale positivity, finite cover membership, refinement ledgers, cover-count comparison, and RealSeparability net routing for BEDC.Derived.BoxDimensionUp.

Constructive story: A $\mathsf{BoxDimension}^{\uparrow}$ packet is generated from a metric row, compact metric source row, dyadic scale row, finite cover row, refinement ledger, counting comparison ledger, separability net route, componentwise transport, provenance, and a local NameCert row.

Chapter 2904

A Concrete Naming Certificate for $\mathsf{CompleteUltrametric}^{\uparrow}$

Following Bourbaki and Bishop–Bridges metric-space practice, the $\mathsf{CompleteUltrametric}^{\uparrow}$ packet records the finite certificate surface for a complete ultrametric rather than a choice-based completion. It sits after the strong-triangle surface of chapter 1535, the limit-witness surface of chapter 128, the Cauchy-filter admission row of chapter 333, the finite-window discipline of chapter 126, the regular-rational tolerance route of chapter 301, and the terminal seal of chapter 38. The packet names one constructive non-Archimedean completeness certificate; it does not import spherical completeness, a selected quotient completion, a host completion functor, or a p-adic field structure.

Definition 2904.1 (CompleteUltrametric carrier)**.** *A* $\mathsf{CompleteUltrametric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, U, C, L, H, R, P, N)$$

with the following displayed rows:

1. *M is the metric-carrier row, including the underlying* $\mathsf{Met}^{\uparrow}$ *distance observations and ball classifier.*
2. *U is the ultrametric strong-triangle row consumed through* $\mathsf{UltrametricSpace}^{\uparrow}$*.*
3. *C is the Cauchy-filter and Cauchy-thread row, including the finite* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads by which tolerance requests are displayed.*
4. *L is the complete-limit witness row consumed through* $\mathsf{CompleteMetric}^{\uparrow}$ *and sealed only at the displayed* $\mathsf{Real}^{\uparrow}$ *endpoint row.*
5. *H records componentwise* hsame *transport for the metric, strong-triangle, Cauchy-thread, limit, provenance, and local naming rows.*
6. *R records displayed* Cont *replay from finite Cauchy-thread reads to the complete-limit handoff and from ultrametric ball comparisons to downstream consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{CompleteUltrametric}^{\uparrow}$*,* $\mathsf{UltrametricSpace}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through M, U, C, L, H, R, P, N. It has no coordinate for spherical completeness, choice-based completion, quotient completion, selected limits outside the displayed thread, p-adic field operations, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CompleteUltrametricUp.CompleteUltrametricTasteGate single carrier alignment`

Theorem 2904.2 (CompleteUltrametric NameCert obligations)**.** *For an accepted* $\mathsf{CompleteUltrametric}^{\uparrow}$ *carrier* $U = (M, U, C, L, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed metric row M and local naming row N;*
2. *ultrametric triangle stability through the strong-triangle row U under componentwise* hsame *transport H;*
3. *Cauchy-thread admission through C, with every tolerance read passing through the* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows;*
4. *complete-limit handoff from C to L through the* $\mathsf{CompleteMetric}^{\uparrow}$ *witness and terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger non-escape for H, R, P, N, preventing transport, replay, provenance, or naming from adding spherical completeness, choice-based completion, quotient completion, selected external limits, or p-adic field structure.*

Proof. Project the accepted carrier of Definition 3879.1. The rows available to a consumer are precisely M, U, C, L, H, R, P, N.

Read the metric and ultrametric data first. The row M supplies the carrier and ball classifier, while U supplies the strong non-Archimedean triangle comparison. Transport by H preserves those displayed rows rather than replacing them by an ambient theorem.

Next read the Cauchy-thread data. The row C exposes only finite filter, stream, and regular-rational tolerance windows, so a limit request has to pass through the same displayed Cauchy evidence.

Finally consume the complete-limit handoff. The row L is reached from C through the $\mathsf{CompleteMetric}^{\uparrow}$ witness and the terminal $\mathsf{Real}^{\uparrow}$ seal. The replay, provenance, and naming rows R, P, N retain that route, so no unlisted completion principle or p-adic structure enters the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteUltrametric}^{\uparrow}$ packet is generated from a metric carrier row, an ultrametric strong-triangle row, Cauchy-filter and thread rows, a complete-limit witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3879.1 and Definition 3879.2 seed the constructive complete non-Archimedean metric certificate over displayed metric, ultrametric, Cauchy-thread, limit, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close spherical completeness, choice-based completion, quotient completion, selected limits outside the displayed thread, p-adic field structure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ultrametric triangle stability, Cauchy-thread admission, complete-limit handoff, and ledger non-escape for BEDC.Derived.CompleteUltrametricUp.

Chapter 2905

A Concrete Naming Certificate for PainleveKuratowskiConvergence$^{\uparrow}$

Following Painleve and Kuratowski on set convergence, the PainleveKuratowskiConvergence$^{\uparrow}$ packet records finite-window closed-set convergence by separating lower-hit witnesses from upper-miss finite-cover witnesses. It sits over the hyperspace carrier of chapter 1660, the finite Hausdorff comparison boundary of chapter 2086, the metric source of chapter 57, the compact metric source of chapter 129, the hit/miss topology of chapter 2375, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed hit/miss windows and their readback rows; it is not a host powerset convergence theorem or a quotient subset equality principle.

Definition 2905.1 (Painleve–Kuratowski convergence carrier)**.** *A* PainleveKuratowskiConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, K, H, V, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *L is the lower-hit witness ledger, listing the finite inspected hits by which points or probes of a displayed closed set are met by later closed-set rows.*
2. *U is the upper-miss finite-cover witness ledger, listing finite open or ball-cover exclusions for the inspected windows.*
3. *K is the inspected compact or hyperspace source row, read through* Hyperspace$^{\uparrow}$ *and the relevant* CompactMetric$^{\uparrow}$ *carrier.*
4. *H is the* HausdorffMetric$^{\uparrow}$ *comparison boundary available only for the displayed compact windows and finite tolerances.*
5. *V is the* VietorisTopology$^{\uparrow}$ *hit/miss row that records the lower-hit and upper-miss subbasic tests used by the packet.*
6. *W is the finite* StreamName$^{\uparrow}$ *window row that schedules the inspected convergence stages and tolerance requests.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational schedules and metric tolerances seen in W and H.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular-rational, Hausdorff, and hit/miss rows have been displayed.*
9. *T records componentwise* hsame *transport for the lower-hit ledger, upper-miss ledger, hyperspace source, Hausdorff boundary, Vietoris row, stream window, regular readback, real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite hit/miss request through $K, H, V, W, R, E, T, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, H, V, W, R, E, T, C, P, N$. It has no coordinate for a host powerset, quotient subset equality, arbitrary supremum or infimum, selected nearest point, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PainleveKuratowskiConvergenceUp.PainleveKuratowskiConvergenceTasteGate`
`single carrier alignment`

Theorem 2905.2 (Painleve–Kuratowski convergence NameCert obligations)**.** *For every accepted* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ *carrier*

$$Q = (L, U, K, H, V, W, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower-hit ledger, upper-miss ledger, hyperspace or compact source, Hausdorff comparison boundary, Vietoris row, stream window, regular readback, real seal, transport, replay, provenance, and local naming row;*
2. *lower-hit coverage, requiring every asserted hit to be witnessed in the displayed ledger L before a convergence read is available;*
3. *upper-miss coverage, requiring every asserted miss to be witnessed by the finite-cover ledger U before exclusion data is exported;*
4. *finite-window compatibility, requiring W and R to schedule exactly the inspected stages and tolerances used by L, U, H, V;*
5. *metric and hyperspace handoff, requiring K, H, V to expose the compact-source, Hausdorff-boundary, and Vietoris hit/miss rows before the real seal is read;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same L, U, K, H, V, W, R, E route.*

No Painleve–Kuratowski convergence obligation is stored outside these rows.

Proof. Project the carrier of Definition 3880.1. The lower-hit coordinate is L, the upper-miss coordinate is U, and the inspected source and topology data are K, H, V.

Lower-hit coverage is read from L. A consumer may cite a hit only after the displayed witness appears in the finite ledger, so the packet does not infer pointwise contact from an ambient subset relation.

Upper-miss coverage is read from U. Each miss is paired with a finite cover or ball-exclusion window before any convergence-facing exclusion is exported, so no arbitrary complement or open-cover compactness theorem is used.

The finite-window route is then checked through W and R. These rows schedule the inspected stages and regular rational tolerances, while H, V attach the Hausdorff and Vietoris readings to the same windows.

Finally, E, T, C, P, N seal, transport, replay, source, and name only that route. Since the carrier has no coordinate for host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, countable choice, or ambient Real completeness, the listed obligations exhaust the certificate. □

Theorem 2905.3 (Painleve–Kuratowski convergence finite-window boundary)**.** *Every convergence-facing consumer of an accepted* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ *carrier receives only the finite displayed route*

$$L, U \longmapsto K, H, V \longmapsto W, R \longmapsto E \longmapsto T, C, P, N.$$

Thus the chapter names finite hit/miss windows for closed-set convergence; it does not export host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a convergence-facing read. By Definition 3880.1, the only lower and upper convergence data are the finite ledgers L and U.

Use Definition 3880.2. The consumer must pass through the hyperspace or compact source row K, the Hausdorff comparison boundary H, and the Vietoris hit/miss row V before it may read the finite schedules W, R and the terminal seal E.

The structural rows T, C, P, N preserve the same displayed dependency order. Because the excluded host-level principles have no carrier coordinate and no replay edge, the packet can export only the named finite-window boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3880.1, Definition 3880.2, and Definition 3880.3 seed the finite hit/miss window route for closed-set convergence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked lower-hit coverage, upper-miss coverage, finite-window compatibility, metric and hyperspace handoff, transport, replay, provenance, and local naming for BEDC.Derived.PainleveKuratowskiConvergenceUp.

Constructive story: A $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ packet is generated from lower-hit witnesses, upper-miss finite-cover witnesses, a compact or hyperspace source row, HausdorffMetric comparison, Vietoris hit/miss topology, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2906

A Concrete Naming Certificate for $\mathsf{HaarWavelet}^{\uparrow}$

Starting from the classical Haar 1910 orthonormal system and its constructive dyadic approximation surface, the $\mathsf{HaarWavelet}^{\uparrow}$ packet records finite dyadic-window wavelet data as BEDC NameCert rows. It sits over the Schauder-basis coordinate route of chapter 3669, the dyadic partition surface of chapter 1863, the finite step-function interface of chapter 3780, and the terminal real-name seal of chapter 38. The packet supplies a finite wavelet coefficient window without importing Haar measure, quotient function spaces, ambient Hilbert-space completeness, or a selected infinite expansion.

Definition 2906.1 (Haar wavelet carrier)**.** *A* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, I, L, V, A, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the dyadic scale row, naming the inspected level and the finite family of cells at that level.*
2. *I is the parent-interval row, read through the* $\mathsf{DyadicPartition}^{\uparrow}$ *packet of chapter 1863.*
3. *L records the left and right child split of each inspected parent interval together with the sign assignment $+1, -1$.*
4. *V is the* $\mathsf{StepFunction}^{\uparrow}$ *value row, so each inspected wavelet is a signed dyadic step function over the displayed children.*
5. *A is the finite coefficient-window row, listing exactly the inspected scale-position coefficients and their local dyadic tolerances.*
6. *B is the* $\mathsf{SchauderBasis}^{\uparrow}$ *coordinate handoff row, consuming the finite coefficient window as a basis-coordinate projection.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficient values and sampled step-function values.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every coefficient and sampled value exported by the packet.*
9. *H records componentwise* hsame *transport for the scale, interval, split, value, coefficient, Schauder handoff, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a wavelet-window read through $K, I, L, V, A, B, R, E, H$.*

11. *P is the* Pkg *provenance over* $\mathsf{HaarWavelet}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for Haar measure, an L^2 quotient class, a host orthogonality theorem, a selected infinite coefficient stream, or ambient Hilbert-space completeness.

Lean4 ✓ `BEDC.Derived.HaarWaveletUp.HaarWaveletTasteGate single carrier alignment`

Theorem 2906.2 (Haar wavelet NameCert obligations)**.** *For every accepted* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier $W = (K, I, L, V, A, B, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic scale exposure, parent-interval exposure, left/right sign-split exposure, step-function value exposure, finite coefficient-window exposure, Schauder-basis coordinate handoff, regular rational readback, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3881.1. The rows K, I, L, V display the finite dyadic scale, the parent interval, the signed child split, and the induced step-function value row before any coefficient consumer can read a wavelet.

The coefficient route is finite. The row A lists only the inspected scale-position coefficients, B hands that finite list to the $\mathsf{SchauderBasis}^{\uparrow}$ coordinate route, and R, E provide the regular rational readback and terminal Real seal for the same displayed values.

The structural rows preserve that route. Since H, C, P, N add only transport, replay, provenance, and local naming, and since the carrier has no coordinate for Haar measure, quotient function classes, host orthogonality, selected infinite streams, or Hilbert-space completeness, the listed rows exhaust the local NameCert surface. □

Theorem 2906.3 (Haar wavelet dyadic-step handoff)**.** *Every* $\mathsf{StepFunction}^{\uparrow}$ *or* $\mathsf{SchauderBasis}^{\uparrow}$ *consumer of an accepted* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier reads the wavelet only through*

$$\mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{StepFunction}^{\uparrow} \longmapsto \mathsf{SchauderBasis}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed scale, split, transport, replay, provenance, and local naming rows. The handoff exports finite dyadic step functions and finite coefficient windows, not Haar integration, quotient function equality, an infinite expansion selector, or ambient completeness.

Proof. Start with the dyadic data. The scale row K and parent-interval row I determine the inspected dyadic cells, and the split row L records the left/right signed child cells over that finite partition.

Next use the step-function value row. The row V turns the signed child data into a $\mathsf{StepFunction}^{\uparrow}$ read, so a step-function consumer receives only the displayed finite cell values and not an ambient function object.

Finally read the coefficient and seal rows. The finite coefficient window A feeds the $\mathsf{SchauderBasis}^{\uparrow}$ coordinate handoff B, and the rows R, E provide regular rational readback and terminal Real seals for the same displayed coefficients. The structural rows H, C, P, N transport and name this route, so no Haar-measure row, quotient equality, infinite expansion selector, or ambient completeness coordinate is exported. □

Falsifiable prediction: An accepted $\mathsf{HaarWavelet}^{\uparrow}$ coefficient read is refuted if its $\mathsf{SchauderBasis}^{\uparrow}$ coordinate handoff is preserved while the displayed $\mathsf{DyadicPartition}^{\uparrow}$ parent interval, left/right sign split, $\mathsf{StepFunction}^{\uparrow}$ value row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal are removed.

Independence witness: dyadic-partition, step-function, schauder-basis, regseqrat, real: none is bijective with $\mathsf{HaarWavelet}^{\uparrow}$ because the packet jointly carries scale rows, parent intervals, signed child splits, step-function values, finite coefficient windows, coordinate handoff, regular readback, terminal seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$.

Constructive story: A $\mathsf{HaarWavelet}^{\uparrow}$ packet is generated from dyadic scale rows, parent interval rows, left/right sign splits, StepFunctionUp values, finite coefficient windows, SchauderBasisUp coordinate handoff, RegSeqRatUp readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3881.1, Definition 3881.2, and Definition 3881.3 seed the finite dyadic-window wavelet packet over partition, step-function, coordinate, readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Haar measure, host orthogonality, quotient function spaces, selected infinite expansions, Hilbert-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic scale exposure, parent-interval exposure, sign-split exactness, StepFunctionUp handoff, SchauderBasisUp coordinate routing, RegSeqRatUp readback, RealUp seal exposure, and structural transport for BEDC.Derived.HaarWaveletUp.

Chapter 2907

A Concrete Naming Certificate for $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ packet records the finite adjunction-facing surface by which regular Cauchy names present the completion route without passing through a host quotient. It sits after the functorial completion packet of chapter 423, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the extension-and-uniqueness surface of chapter 397. The chapter names the adjunction as displayed BEDC data over finite rows; it does not construct a quotient completion, select limits, assert host completed metric spaces, or import ambient completeness.

Definition 2907.1 (Cauchy regular completion adjunction carrier)**.** *A* $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, R, W, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row naming the functorial completion surface over the same source metric and regular-Cauchy data.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which regular rational Cauchy names are exposed.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by the regular Cauchy route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed stream windows and regular-rational reads.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D have been consumed.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row recording extension existence and uniqueness for the same displayed completion route.*
7. *H records componentwise* hsame *transport for F, R, W, D, L, U.*
8. *C records displayed* Cont *replay from functorial completion requests to stream windows, from windows and dyadic tolerances to regular rational readback, from readback to the Real seal, and from the Real seal into the universal-property handoff.*

9. *P is the* Pkg *provenance over* CauchyRegularCompletionAdjunction$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$ *row.*

The carrier exposes the regular completion adjunction only through these finite rows. It has no coordinate for quotient completion equality, selected representatives, host completed metric spaces, hidden completeness witnesses, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRegularCompletionAdjunctionUp.CauchyRegularCompletionAdjunctionTasteGate`
`single carrier alignment`

Theorem 2907.2 (Cauchy regular completion adjunction NameCert obligations)**.** *For every accepted* CauchyRegularCompletionAdjunction$^{\uparrow}$ *carrier $A = (F, R, W, D, L, U, H, C, P, N)$, the local* NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of carrier admission for A, regular-readback compatibility through R, finite-window budget compatibility through W, D, terminal Real seal determinacy through L, universal-property handoff through U, functorial completion source exposure through F, and structural transport, replay, provenance, and local naming through H, C, P, N.*

The ledger refuses every route that would discharge an obligation by quotient completion equality, a selected limit, a host completed metric space, an ambient completeness principle, a hidden cofinal stream family, or a host-level equality of real names.

Proof. Project the carrier of Definition 3882.1. The visible rows are exactly $F, R, W, D, L, U, H, C, P, N$, and each listed NameCert obligation is a projection of one of those rows.

The regular-readback side is finite. The stream schedule W names the exposed windows, the tolerance ledger D attaches dyadic bounds to those same windows, and the regular rational row R is the only readback route before the terminal Real seal L can be consumed.

The adjunction-facing side is downstream from that route. The functor row F records the completion-facing source packet, while the universal-property row U records extension and uniqueness only after the same R, W, D, L rows have been read.

Finally, H, C, P, N preserve componentwise comparison, replay, provenance, and local naming for the displayed rows. Since the carrier contains no quotient-completion, selection, host-completion, hidden-completeness, cofinal-family, or host-equality coordinate, the ledger refusal clauses follow from exhaustion of the tuple. □

Falsifiable prediction: Every accepted CauchyRegularCompletionAdjunction$^{\uparrow}$ consumer that exports a completion adjunction while omitting the CauchyCompletionFunctor row, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, CompletionUniversalProperty row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchycompletionfunctor, regseqrat, streamname, dyadicratcore, real, and completionuniversalproperty: none is bijective to the regular completion adjunction packet because the packet jointly carries functorial source exposure, regular readback, finite-window budget, tolerance ledger, terminal seal, universal-property handoff, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$.

Constructive story: A CauchyRegularCompletionAdjunction$^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a RegSeqRat readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, a CompletionUniversalProperty handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRegularCompletionAdjunction}\uparrow}$, seed)

Scope closed: Definition 3882.1 and Definition 3882.2 seed the finite regular completion adjunction surface over CauchyCompletionFunctorUp, RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, RealUp, CompletionUniversalPropertyUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRegularCompletionAdjunction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, RealUp completeness, quotient completion equality, selected host limits, host completed metric spaces, hidden cofinal streams, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, regular-readback compatibility, finite-window budget compatibility, terminal Real seal determinacy, universal-property handoff, ledger refusal, and TasteGate field faithfulness for BEDC.Derived.CauchyRegularCompletionAdjunctionUp.

Chapter 2908

A Concrete Naming Certificate for $\mathsf{Quasiisometry}^{\uparrow}$

Following Gromov and Bridson–Haefliger, the $\mathsf{Quasiisometry}^{\uparrow}$ packet records coarse metric equivalence as finite BEDC coarse metric data rather than as an ambient set-theoretic map. It reads source and target metric presentations from chapter 57, reuses the displayed host-map discipline of chapter 410, keeps properness visible through **??**, sends completion-facing rows through chapter 1937, and uses the readback surfaces of chapter 301 and chapter 38. The packet names finite coarse equivalence evidence; it is not a selected inverse function, not quotient equality of completions, not a host-space bijection, and not an oracle for external large-scale geometry.

Definition 2908.1 (Quasi-isometry carrier)**.** *A* $\mathsf{Quasiisometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, F, A, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, recording the named* $\mathsf{Met}^{\uparrow}$ *presentation and the source distance queries visible to the packet.*
2. *Y is the target metric row, recording the named* $\mathsf{Met}^{\uparrow}$ *presentation and the target distance queries visible to the packet.*
3. *F is the coarse map graph row, recording the finite graph of the displayed map route without adding a host-level function outside the BEDC carrier.*
4. *A is the affine distortion row, storing the multiplicative and additive control constants that bound target distances from source distances and source distances from target witnesses.*
5. *S is the coarse-surjectivity witness row, recording the finite target-cover ledger and its bounded return witnesses.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *distance readback row, storing rational windows and real-distance readings used by the coarse estimates.*
7. *E is the completion-facing row, recording the handoff to* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *consumers and the stability data exposed to completion routes.*
8. *H records componentwise* hsame *transport for source metric, target metric, graph, distortion, coarse-surjectivity, readback, completion-facing, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a source or target coarse query through graph readback, distortion control, surjectivity witness, distance readback, and completion-facing handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{QuasiIsometry}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, CauchyCompletionFunctoriality, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiIsometry}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $X, Y, F, A, S, R, E, H, C, P, N$. It has no coordinate for a selected inverse map, quotient equality of completions, a host-space bijection, an external large-scale geometry theorem, or a choice principle for target witnesses.

Lean4 ✓ `BEDC.Derived.QuasiIsometryUp.QuasiIsometryTasteGate single carrier alignment`

Theorem 2908.2 (Quasi-isometry NameCert obligations)**.** *For every accepted* $\mathsf{QuasiIsometry}^{\uparrow}$ *carrier $Q = (X, Y, F, A, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuasiIsometry}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *Carrier admission for the displayed source metric, target metric, graph, distortion, surjectivity, readback, completion-facing, transport, replay, provenance, and local naming rows.*
2. *Coarse Lipschitz control, read from F, A, R, H, C, for source-distance windows and target-distance windows displayed by the packet.*
3. *Coarse-surjectivity ledger exactness, read from S, R, H, C, for every target-side witness exported by the packet.*
4. *Completion-facing stability, read from E, H, C, P, for every consumer that routes coarse metric evidence into completion functoriality.*
5. *Non-escape from host maps, selected witnesses, choice principles, quotient equality, and ambient bijections, witnessed by the absence of such coordinates from $X, Y, F, A, S, R, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.QuasiIsometryUp.QuasiIsometryNamecertObligations`

Proof. Project the carrier of Definition 3883.1. The accepted packet exposes only the two metric rows, the coarse graph, affine distortion constants, coarse-surjectivity witnesses, distance readback, completion-facing handoff, transport, replay, provenance, and local naming.

The coarse Lipschitz and coarse-surjectivity obligations are finite route inspections. The rows F, A, S, R, H, C display the graph, bounds, witnesses, readback windows, transport equalities, and replay path before any consumer receives a quasi-isometry reading.

The completion-facing obligation is another projection. A completion consumer must pass through E together with the same transport, replay, and provenance rows H, C, P, so it cannot obtain a host completion isomorphism or quotient equality from the packet.

The non-escape row follows from the carrier boundary itself. Since no coordinate stores a selected inverse map, a choice functional, an ambient bijection, or quotient equality of completions, the five listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{QuasiIsometry}^{\uparrow}$ read is refuted if a consumer obtains coarse equivalence, completion-facing stability, or target-side coverage without displaying source and target metric rows, graph readback, affine distortion constants, coarse-surjectivity witnesses, RegSeqRat/Real readback, transport, replay, provenance, and local naming.

Independence witness: metric, lipschitzmap, propermetric, cauchy-completion-functoriality, regseqrat, real: none is bijective to $\mathsf{QuasiIsometry}^{\uparrow}$ because this packet jointly carries source and

target metric rows, coarse graph data, affine distortion constants, coarse-surjectivity witnesses, rational and real distance readback, completion-facing stability, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}$.

Constructive story: A $\mathsf{QuasiIsometry}^{\uparrow}$ packet is generated from source and target metric rows, a finite coarse map graph, affine distortion constants, coarse-surjectivity witnesses, RegSeqRat/Real distance readback, completion-facing stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3883.1 and Definition 3883.2 seed the finite coarse metric equivalence certificate over displayed metric, graph, distortion, surjectivity, readback, completion-facing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.QuasiIsometryUp`

Bridge status: none

Not claimed: This chapter does not close a selected inverse map, quotient equality of completions, host-space bijection, external large-scale geometry theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coarse Lipschitz control, coarse-surjectivity exactness, completion-facing stability, non-escape from host maps, and local naming for BEDC.Derived.QuasiIsometryUp.

Chapter 2909

A Concrete Naming Certificate for DyadicClosedIntervalEndpointGrid$^{\uparrow}$

Following Bishop and Bridges constructive interval analysis, the DyadicClosedIntervalEndpointGrid$^{\uparrow}$ packet records the finite dyadic endpoint grid read before compact-uniform-continuity interval consumers inspect nets, moduli, or coverage rows. It refines the closed bounded interval source of chapter 1591 through the dyadic interval surface of chapter 306, the StreamName and RegSeqRat readback rows, and the terminal Real seal. The route serves the compact-uniform-continuity program of chapter 9237 without invoking open-cover compactness, selected subcovers, quotient real equality, trichotomy, or ambient completeness.

Definition 2909.1 (Dyadic closed interval endpoint-grid carrier)**.** *A* DyadicClosedIntervalEndpointGrid$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* ClosedBoundedInterval$^{\uparrow}$ *row, exposing lower endpoint, upper endpoint, endpoint order, rational containment, dyadic containment, StreamName windows, RegSeqRat readbacks, and the Real seal through the root package.*
2. *D is the finite* DyadicInterval$^{\uparrow}$ *endpoint-grid row, listing dyadic cells whose endpoints are admitted by the interval source and ordered by the same endpoint-order route.*
3. *W is the finite window row, exposing the StreamName windows attached to the grid cells before any net, modulus, or coverage consumer reads them.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed dyadic endpoints and cell windows.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row, reached only after I, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport across the closed interval source, dyadic grid, windows, RegSeqRat readbacks, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from compact-uniform-continuity interval requests through the endpoint grid before finite-net or modulus rows are inspected.*
8. *P is the* Pkg *provenance over* ClosedBoundedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *compact-uniform-continuity interval consumers,* BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $I, D, W, R, E, H, C, P, N$. It has no coordinate for host closed intervals, open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicClosedIntervalEndpointGridUp.DyadicClosedIntervalEndpointGridTasteGate`
`single carrier alignment`

Theorem 2909.2 (Dyadic closed interval endpoint-grid obligations)**.** *For every accepted* DyadicClosedIntervalEndpo *carrier $G = (I, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$ *surface consists of carrier admission, dyadic-cell exposure, endpoint-order preservation, window handoff, Real-seal nonescape, and local naming. The carrier admits only source rows from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*, exposes dyadic cells only through D, preserves endpoint order from I to every displayed cell endpoint, hands StreamName windows and RegSeqRat readbacks through W, R, reaches the Real seal only through E, and names the entire route by H, C, P, N.*

Proof. Project the carrier of Definition 3884.1. The source row I is the only closed bounded interval input, and the finite grid row D is the only coordinate listing dyadic endpoint cells.

Endpoint order is inherited from the source row before any grid cell is read. Since D is attached to I, every displayed dyadic endpoint is admitted through the same lower-endpoint, upper-endpoint, order, rational-containment, and dyadic-containment route used by chapter 1591.

The consumer-facing rows are then forced through W, R, E. StreamName windows are exposed by W, regular rational endpoint readbacks by R, and the terminal Real seal by E. None of those rows supplies host compactness data or a selected subcover.

Finally H, C, P, N transport, replay, source, and name exactly this displayed route. Because the carrier has no other coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2909.3 (Dyadic closed interval endpoint-grid window handoff)**.** *Every compact-uniform-continuity interval read that consumes an accepted* $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ *carrier must pass through the displayed endpoint grid*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it reads any finite-net row, coverage row, pointwise-modulus row, uniform-modulus row, or compactness consumer row.

Proof. Start with the accepted carrier $G = (I, D, W, R, E, H, C, P, N)$. A compact-uniform-continuity interval consumer first receives I, so the closed bounded interval endpoint and containment package is displayed before any grid row is available.

The read then passes through the finite dyadic endpoint grid D. This row exposes the dyadic cells and their endpoint order, but it does not yet expose a finite net, a cover, or a modulus.

Next the consumer receives the window and readback rows W, R, E. These rows hand off StreamName windows, RegSeqRat endpoint readbacks, and the Real seal as finite endpoint-grid data.

Only after this route has been replayed by C and sourced by P may a downstream compact-uniform-continuity consumer add net or modulus rows. Thus the interval read reaches the compact-uniform-continuity surface through the endpoint grid rather than through an open-cover compactness principle or a host real interval. □

Falsifiable prediction: An accepted compact-uniform-continuity interval consumer that reads finite-net, coverage, pointwise-modulus, uniform-modulus, or compactness rows before inspecting $I, D, W, R, E, H, C, P, N$ is outside the $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ packet.

Independence witness: closedboundedinterval, dyadicinterval, streamname, regseqrat, real: $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed bounded interval root package, a finite dyadic endpoint grid, finite windows, regular rational readbacks, a terminal Real seal, transport, replay, provenance, and local naming for compact-uniform-continuity interval consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source row, DyadicInterval endpoint-grid row, StreamName window row, RegSeqRat readback row, Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3884.1, Definition 3884.2, and Definition 3884.3 seed the finite endpoint-grid route for compact-uniform-continuity interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, finite-net compactness, open-cover compactness, pointwise-modulus extraction, uniform-modulus construction, quotient real equality, trichotomy, selected points, selected subcovers, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-cell exposure, endpoint-order preservation, window handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicClosedIntervalEndpointGr

Chapter 2910

A Concrete Naming Certificate for CauchyFilterCompletionCriterion$^{\uparrow}$

Following Kelley and Bishop–Bridges, the CauchyFilterCompletionCriterion$^{\uparrow}$ packet records the textbook completion criterion that a displayed Cauchy-filter route may reach a completion boundary only through finite basis, window, tolerance, readback, and real-seal data. It sits after the Cauchy-filter completion boundary of chapter 1569, the filter-limit basis route of chapter 3862, the Cauchy-filter surface of chapter 333, the uniform-completion surface of chapter 1473, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the criterion as finite BEDC data; it does not quotient filters, select a limit, use host equality, invoke choice, or import ambient completeness.

Definition 2910.1 (CauchyFilterCompletionCriterion carrier)**.** *A* CauchyFilterCompletionCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, B, F, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* CauchyFilterCompletion$^{\uparrow}$ *completion-boundary row whose accepted reads remain tied to displayed Cauchy-filter data.*
2. *B is the* FilterLimitBasis$^{\uparrow}$ *row exposing the finite basis route used by limit and completion consumers.*
3. *F is the* CauchyFilter$^{\uparrow}$ *source row, including the displayed base, refinement, and Cauchy evidence read by Q and B.*
4. *U is the* UniformCompletion$^{\uparrow}$ *handoff row, available only after the same filter, basis, window, tolerance, and readback route has been displayed.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window schedule used to inspect requested tails.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to the same windows and completion requests.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the finite windows and dyadic ledger.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after Q, B, F, U, S, D, R have been replayed.*
9. *H records componentwise* hsame *transport for the completion, basis, filter, uniform, window, tolerance, readback, real-seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a completion-criterion request through* $Q, B, F, U, S, D, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, F, U, S, D, R, E, H, C, P, N$. There is no coordinate for quotient filters, selected representatives, host filter equality, selected completion points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionTasteGate`
`single carrier alignment`

Theorem 2910.2 (CauchyFilterCompletionCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterComple}}$ *surface consists exactly of carrier admission, CauchyFilterCompletion boundary exposure through Q, FilterLimitBasis route exposure through B, CauchyFilter source exposure through F, UniformCompletion handoff exposure through U, finite StreamName window exposure through S, DyadicRatCore tolerance exactness through D, RegSeqRat readback through R, Real seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by quotient-filter equality, selected representatives, selected completion points, host filter equality, countable choice, hidden cofinal tails, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 3885.1. The displayed rows are exactly $Q, B, F, U, S, D, R, E, H, C, P, N$, so carrier admission exposes only completion-boundary, basis, Cauchy-filter, uniform-handoff, window, tolerance, readback, seal, transport, replay, provenance, and naming data.

The completion side starts with Q, B, F. The Cauchy-filter completion row Q and the limit-basis row B both read the same Cauchy-filter source row F, so the criterion has one finite filter route rather than a quotient or detached basis route.

Next read the uniform and numerical rows. The handoff U is downstream from the displayed filter route, while S, D, R give the finite windows, dyadic tolerances, and regular rational readback used before the terminal seal E is available.

Finally the structural rows H, C, P, N preserve the same route. Since quotient-filter equality, selected representatives, selected completion points, host filter equality, choice, hidden cofinal tails, and ambient completeness are absent from the carrier, the listed rows exhaust the local surface. □

Theorem 2910.3 (CauchyFilterCompletionCriterion basis route)**.** *Let $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier. Every completion-facing or Real-facing read factors through*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{FilterLimitBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeq}$$

The $\mathsf{UniformCompletion}^{\uparrow}$ *handoff reads the same route through U and cannot bypass $F, B, Q, S, D, R, E, H, C, P, N$. Thus the criterion exports finite basis evidence for the L10 completion route without quotient filters, selected limits, host equality, choice, hidden cofinal tails, or ambient completeness.*

Lean4 ✓ BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionBasisRoute

Proof. Begin with a completion-facing or Real-facing read of K. The carrier exposes the source Cauchy-filter row F, the basis row B, and the completion-boundary row Q before any terminal Real seal can be inspected.

Replay the displayed continuation row C. The read passes from F to B and Q, then through the finite window schedule S, dyadic tolerance ledger D, and regular rational readback R before it reaches E.

The uniform handoff does not open a second completion object. The row U is one of the displayed coordinates and is preserved by H, C, P, N, so a $\mathsf{UniformCompletion}^{\uparrow}$ consumer receives the same basis route rather than a hidden quotient, chosen representative, or ambient completion.

The excluded objects are not carrier rows. Consequently every accepted read factors through the route displayed in the statement. □

Theorem 2910.4 (CauchyFilterCompletionCriterion uniform handoff nonescape)**.** *For an accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff read is confined to the displayed row* U *of Definition 3885.1. The handoff may be read only after the basis route of Definition 3885.3 has exposed* F, B, Q, S, D, R, E*, and it cannot replace that route by a quotient filter, selected completion point, host equality, or countable choice.*

Proof. Inspect the carrier rows. The uniform-facing coordinate is U, and the basis-route theorem requires the Cauchy-filter source, basis row, completion-boundary row, window schedule, tolerance ledger, readback row, and Real seal to be displayed in the same packet.

The handoff is therefore a downstream read of the existing route. It cannot open a second completion object or select a limit, because no quotient, selection, host-equality, or choice coordinate appears in K.

The structural rows H, C, P, N transport, replay, source, and name the same handoff. Thus every uniform read remains inside the finite criterion packet. □

Theorem 2910.5 (CauchyFilterCompletionCriterion readback exactness)**.** *For an accepted* CauchyFilterCompletion*carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every numerical readback consumed by the criterion is exactly the finite route through the window row* S*, the dyadic tolerance row* D*, and the regular rational readback row* R *named in Definition 3885.1. The basis route of Definition 3885.3 admits no hidden cofinal tail, host filter equality, or ambient real-completeness read.*

Proof. The basis route displays the Cauchy-filter, basis, and completion-boundary rows before any numerical read is available. After those rows, the only numerical coordinates in the carrier are S, D, R.

Exactness means that the requested tail is inspected through the finite window schedule S, the tolerance is read from D, and the regular rational value is read from R. A consumer has no coordinate from which to infer an unseen cofinal tail or compare filters by host equality.

The replay and transport rows C and H preserve this same finite route, with provenance P and local naming N. Hence every readback is precisely the displayed S, D, R ledger. □

Theorem 2910.6 (CauchyFilterCompletionCriterion Real seal nonescape)**.** *For an accepted* CauchyFilterCompletio*carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing seal read is confined to the terminal row* E *of Definition 3885.1. The row* E *may be consumed only after the basis route of Definition 3885.3 has displayed the Cauchy-filter source, basis, completion boundary, uniform handoff, windows, dyadic tolerances, and regular rational readback. It exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The terminal seal is a row of the carrier, not a theorem about all real names. The basis-route theorem places it after the filter, basis, completion-boundary, window, tolerance, and regular rational readback rows.

A Real-facing consumer therefore reaches E only by replaying the displayed finite path. The consumer cannot bypass the route through a hidden completion object, a selected representative, host equality, or countable choice.

Since E is transported and replayed only by H and C, with provenance and local naming in P and N, it seals the displayed readback without exporting ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ read factors through $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, FilterLimitBasis row, CauchyFilter source row, UniformCompletion handoff row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3885.1, Definition 3885.2, and Definition 3885.3 seed the L10 completion criterion route over CauchyFilterCompletionUp, FilterLimitBasisUp, CauchyFilterUp, UniformCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected limits, host equality, countable choice, hidden cofinal tails, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, CauchyFilterCompletion boundary exposure, FilterLimitBasis route exposure, UniformCompletion handoff nonescape, readback exactness, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyFilterCompletionCriterionUp.

Chapter 2911

A Concrete Naming Certificate for $\mathsf{ConstructiveUniformLimit}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ packet records the theorem route saying that a uniform limit of continuous real-valued maps is continuous when every ingredient is read through finite BEDC certificate rows. It refines the continuous-map surface of chapter 130, the uniform-limit endpoint route of chapter 463, the window ledger of chapter 126, the regular-rational readback of chapter 301, and the real seal of chapter 38.

The chapter is a human-origin L10 reconstruction of the classical uniform-limit theorem as a finite NameCert route. It exposes continuous approximants, a uniform convergence ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming. It does not introduce a quotient function space, a selected host limit, pointwise-only convergence, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 2911.1 (Constructive uniform-limit carrier)**.** *A* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, L, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *A is the finite family of accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant rows, each exposing a common source certificate, a target* $\mathsf{Real}^{\uparrow}$ *seal, and its local continuity modulus ledger;*
2. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *row that records the uniform convergence threshold ledger and endpoint-facing route for the same family;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule on which all approximants and all convergence thresholds are read;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row tying the window data to rational endpoint observations;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the uniform-limit and regular-rational readback rows have both been consumed;*
6. *H records componentwise* hsame *transport for approximants, window schedules, threshold ledgers, readbacks, endpoint seals, and local modulus rows;*
7. *C records displayed* Cont *replay of the approximant continuity ledgers and of the uniform-limit handoff;*

8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$ *row.*

The carrier classifier compares two packets only by replaying the displayed rows $A, L, W, R, E, H, C, P, N$. No coordinate stores pointwise-only convergence, a selected host limit, quotient equality between functions, open-cover compactness, or a free completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConstructiveUniformLimitUp.ConstructiveUniformLimitTasteGate single carrier alignment`

Theorem 2911.2 (Constructive uniform-limit NameCert obligations)**.** *For every accepted* ConstructiveUniformLimi *carrier $U = (A, L, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed packet, continuous-approximant admission through A, uniform-limit admission through L, finite window admission through W, regular-rational readback through R, terminal real-seal admission through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3886.1. The rows $A, L, W, R, E, H, C, P, N$ are the complete visible coordinates of one finite packet.

Read the analytic rows in dependency order. The continuous approximants provide local modulus ledgers, the uniform-limit row provides the common tail threshold, the window row supplies the shared finite schedule, and the regular-rational readback row exposes the endpoint observations used by both routes.

The seal and replay rows then close the local obligation surface. The $\mathsf{Real}^{\uparrow}$ endpoint E is available only after the shared windows, uniform threshold ledger, and regular readback row have been consumed; H and C preserve exactly those displayed rows.

Finally, P and N source the same finite packet as provenance and local naming. Since no carrier coordinate stores a selected host limit, quotient function equality, pointwise-only convergence, open-cover compactness, or ambient real completeness, no further obligation is exported. □

Theorem 2911.3 (Constructive uniform-limit continuity route)**.** *Every continuous-map consumer of an accepted* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed hsame, Cont, Pkg, *and* NameCert *rows. The limiting map is therefore consumed as a continuity-preserving finite handoff from continuous approximants and a uniform convergence ledger, not as a host-level function limit.*

Proof. Start with a continuous-map consumer of the limiting endpoint. By Definition 3886.1, such a consumer must first read the approximant row A and the shared window schedule W.

Next use the uniform-limit row L. It supplies the common threshold ledger that lets a single window and tolerance surface control the family uniformly, so the consumer cannot replace it with a pointwise convergence witness.

Read the regular-rational handoff R and the real seal E. The endpoint is visible only along the same window and threshold route that generated the uniform-limit packet.

Finally replay transport and continuation through H and C, then source the result through P and N. The displayed route is finite and certificate-local, so quotient function spaces, selected host limits, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ packet is generated from continuous approximant rows, a $\mathsf{UniformLimit}^{\uparrow}$ uniform convergence ledger, shared $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3886.1, Definition 3886.2, and Definition 3886.3 seed the finite route from continuous approximants and a uniform convergence ledger to a continuity-preserving real endpoint consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConstructiveUniformLimitUp.ConstructiveUniformLimitTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient function spaces, selected host limits, pointwise-only convergence, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or a full function-space completeness theorem.

Upgrade path: Obligation closure requires binding the approximant, uniform threshold, window, regular readback, real seal, transport, replay, provenance, and local naming rows to exact BEDC primitives and a Lean theorem surface.

Chapter 2912

A Concrete Naming Certificate for CompactnessUniformContinuityDiameterDecay$^{\uparrow}$

Following Bishop and Bridges, the CompactnessUniformContinuityDiameterDecay$^{\uparrow}$ packet realizes the compact finite-net route from continuous pointwise control to a diameter-decay uniform-modulus certificate. It sits over the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-modulus consumer of chapter 369, the triangle estimate of chapter 1159, and the real endpoint seal of chapter 38.

The packet records diameter decay as finite certificate data: compact-net windows, local continuous-map oscillation bounds, a common uniform-modulus handoff, source and target metric triangle rows, and a terminal Real$^{\uparrow}$ seal. It does not appeal to open-cover compactness, a selected subsequence, a choice-selected center outside the bundle, semantic continuity, or ambient real completeness.

Definition 2912.1 (Compact uniform-continuity diameter-decay carrier)**.** *A* CompactnessUniformContinuityDiamet *carrier is a finite* BHist *packet*

$$D = (K, F, M, T, E, H, C, P, N)$$

with the following visible rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row carrying finite-net centers, located source-distance reads, and compact bundle coverage;*
2. *F is the* ContinuousMap$^{\uparrow}$ *row exposing the graph and pointwise modulus data for the displayed map into a* Real$^{\uparrow}$ *target;*
3. *M is the* UniformModulus$^{\uparrow}$ *handoff row obtained from the compact finite-net fold and consumed by downstream uniform-continuity readers;*
4. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row joining the selected compact-net center, two source-distance bounds, two target estimates, and the final target triangle estimate;*
5. *E is the* Real$^{\uparrow}$ *endpoint seal through which the target-diameter and oscillation reads must pass;*
6. *H records componentwise* hsame *transport for the compact source, continuous-map row, uniform-modulus row, triangle estimate, and real seal;*
7. *C records displayed* Cont *replay of the finite-net coverage, pointwise-modulus read, uniform-modulus handoff, and triangle-estimate route;*

8. *P is the* Pkg *provenance over* $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}$ *row.*

The classifier compares two packets only through the displayed rows $K, F, M, T, E, H, C, P, N$. It has no coordinate for host compactness, selected subsequences, choice outside the finite bundle, semantic continuity, quotient equality, or real completeness.

Lean4 ✓ BEDC.Derived.CompactnessUniformContinuityDiameterDecayUp.CompactnessUniformContinuityDiameterDe
single carrier alignment

Theorem 2912.2 (Diameter-decay NameCert obligations)**.** *For every accepted* CompactnessUniformContinuityDiam *carrier $D = (K, F, M, T, E, H, C, P, N)$, the local* NameCert *obligation surface consists exactly of compact-metric admission through K, continuous-map admission through F, uniform-modulus handoff through M, metric-triangle uniform estimate through T, real-seal admission through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3887.1. The finite packet exposes exactly $K, F, M, T, E, H, C, P, N$.

Read the compact-continuity rows first. The compact metric row supplies the finite-net centers and located source-distance reads, while the continuous-map row supplies the pointwise target estimates at those centers.

Next read the diameter-decay handoff. The uniform-modulus row receives a precision chosen from the finite compact-net fold, and the metric-triangle row converts the selected center route into the displayed target-diameter estimate.

The real seal, transport, replay, provenance, and local naming rows close the obligation surface. Since host compactness, selected subsequences, choice outside the finite bundle, semantic continuity, quotient equality, and real completeness are absent from the carrier, they cannot be exported by the local certificate. □

Theorem 2912.3 (Compact diameter-decay uniform-modulus route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$ *packet receives the compact finite-net row, the continuous-map pointwise rows, the metric-triangle uniform estimate, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as one finite route:*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the diameter-decay certificate is a finite compact-continuous image packet, not a semantic compactness theorem or an external modulus selector.

Proof. Start from the compact metric row K. It supplies the finite centers and source-distance reads used to measure the diameter of each displayed image window.

Read the continuous-map row F at the selected centers. The pointwise modulus data are tied to the same source rows that K exposes, so a consumer cannot choose a detached center or a detached target estimate.

Apply the triangle-estimate row T and then the uniform-modulus handoff M. The source and target triangle rows convert the local estimates into a single diameter-decay precision consumed by $\mathsf{UniformModulus}^{\uparrow}$.

Finally consume the real seal E and replay the route through H, C, P, N. The consumer obtains only the finite packet data, so open-cover compactness, selected subsequences, choice outside the bundle, semantic continuity, and real completeness remain unavailable. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$ packet is generated from $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows, $\mathsf{ContinuousMap}^{\uparrow}$ pointwise modulus data, a $\mathsf{UniformModulus}^{\uparrow}$ handoff, a $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3887.1, Definition 3887.2, and Definition 3887.3 seed the finite diameter-decay route from compact metric finite nets and continuous-map pointwise rows to a uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactnessUniformContinuityDiameterDecayUp.CompactnessUniformContinuityDiameterDecayTa`

Bridge status: none

Not claimed: This chapter does not close semantic compactness, selected subsequences, choice outside the finite bundle, quotient equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires binding the compact finite-net, continuous-map, uniform-modulus, triangle-estimate, real-seal, transport, replay, provenance, and local naming rows to exact BEDC primitives and a Lean theorem surface.

Chapter 2913

A Concrete Naming Certificate for $\mathsf{MetricCompletionIdempotence}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ packet records the fixed-point surface of constructive metric completion: once a displayed metric-completion route is already separated and complete at the finite BEDC interface, a second completion read must return to the same named rows. It sits after the metric-completion carrier of chapter 1506, the Cauchy-filter completion route of chapter 1569, the separated-completion boundary of chapter 1612, the complete separated metric surface of chapter 2355, and the uniform-completion handoff of chapter 1473. The packet gives downstream completion-functor, real-completion, and regular-rational routes a fixed-point target without quotient completion equality, selected representatives, host equality, countable choice, or an ambient completeness axiom.

Definition 2913.1 (MetricCompletionIdempotence carrier)**.** *A* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, S, C, U, R, H, K, P, N)$$

with the following displayed rows:

1. *M is the first* $\mathsf{MetricCompletion}^{\uparrow}$ *row, including the source metric, Cauchy-filter or Cauchy-net branch, finite readback, and separated comparison route.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row requiring the first completion read to be separated before the fixed-point claim is consumed.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *limit-service row, exposing only the displayed Cauchy-name endpoint data available inside the completed surface.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *compatibility row, tying the metric-completion read to the same uniform completion route when a uniform consumer is present.*
5. *R is the second-completion readback row. It records that any repeated completion request replays M, S, C, U rather than introducing a second quotient, selector, host equality, or ambient complete metric space.*
6. *H records componentwise* hsame *transport for the first completion, separated boundary, complete-metric service, uniform handoff, repeated readback, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from the first completion request through the separated and complete rows to the repeated-completion readback.*

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompleteSeparatedMe}$ $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected Cauchy representatives, host equality of completed points, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a second completion object outside the replayed rows.*

Lean4 ✓ `BEDC.Derived.MetricCompletionIdempotenceUp.MetricCompletionIdempotenceTasteGate single carrier alignment`

Theorem 2913.2 (MetricCompletionIdempotence NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier* $I = (M, S, C, U, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *first-completion admission, requiring the initial read to pass through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *row M;*

2. *separated fixed-point boundary, requiring the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row S to be read before a repeated completion request is accepted;*

3. *complete-service compatibility, requiring the limit-service row C to expose only the completed finite endpoint data already available from the same source;*

4. *uniform-route compatibility, requiring every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer to share the route M, S, C before it reaches the repeated readback row R;*

5. *idempotence nonescape, requiring R, H, K, P, N to replay, transport, package, and name exactly the same route, with no quotient completion equality, selected representative, host equality, countable choice, ambient completeness, or detached second completion coordinate.*

Proof. Project the carrier of Definition 3888.1. The first completion row is M, and the only rows that can certify fixed-point behavior before the repeated readback are S, C, U.

Read S through the separated-completion boundary of chapter 1612. This forces any equality or comparison of completed names to use displayed separated classifier data rather than host equality or quotient classes.

Read C through the complete-metric service. The row supplies endpoint service only for the already displayed completion data, so it cannot select a new representative or create a second ambient complete space.

For uniform-facing consumers, read U through chapter 1473. The uniform route is required to share the same completion rows before the repeated readback R is available.

Finally H, K, P, N transport, replay, package, and name the same finite route. Since the carrier has no excluded coordinate, the five listed rows exhaust the local NameCert surface. □

Theorem 2913.3 (MetricCompletionIdempotence repeated completion fixed point)**.** *Let $I = (M, S, C, U, R, H, K, P, N)$ be an accepted* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier. Every second* $\mathsf{MetricCompletion}^{\uparrow}$ *read consumed through I factors through the finite route*

$$M \longmapsto S, C, U \longmapsto R \longmapsto H, K, P, N.$$

The second read therefore lands on the displayed completion surface already named by M, S, C, U and cannot be replaced by quotient completion equality, selected representatives, host equality, countable choice, ambient completeness, or a detached completion object.

Proof. Start with an accepted repeated completion request. Carrier admission gives R only as a row downstream of the first completion row M and the boundary rows S, C, U.

Use Definition 1506.3. The first completion read is already confined to the displayed metric source, cofinal branch, finite readback, and separated comparison route.

Use the separated boundary supplied by S and the complete-service row C. They ensure that the repeated request reads a completed separated metric surface rather than a quotient completion or a selected representative.

The uniform compatibility row U keeps uniform consumers on the same route. Transport and replay through H, K then carry that route to R, while P, N record provenance and local naming. □

Theorem 2913.4 (MetricCompletionIdempotence obligation scope)**.** *For an accepted* MetricCompletionIdempotenc *carrier $I = (M, S, C, U, R, H, K, P, N)$, the obligation surface is exactly first* MetricCompletion$^{\uparrow}$ *admission through M,* SeparatedCompletion$^{\uparrow}$ *boundary through S,* CompleteMetric$^{\uparrow}$/CompleteSeparatedMetric$^{\uparrow}$ *service through C,* UniformCompletion$^{\uparrow}$ *compatibility through U, repeated-readback determinacy through R,* hsame *transport through H,* Cont *replay through K,* Pkg *provenance through P, and local* NameCert *through N. It exports no quotient completion, selected representative, host equality, countable choice, or* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier. The first completion admission is the row M, and the fixed-point boundary can inspect only S, C, U before it reaches the repeated readback row R.

The separated and complete rows determine the admissible service surface. The row S blocks quotient completion equality, while C exposes only the displayed complete-metric or complete-separated-metric service already present in the packet.

Uniform compatibility is the role of U. It keeps uniform consumers on the same completion route before the repeated readback is public, so no separate completion object or host equality row is introduced.

Finally R, H, K, P, N determine repeated-readback determinacy, transport, replay, provenance, and local naming. Since the carrier has no coordinate for selected representatives, countable choice, or Real$^{\uparrow}$ completeness, the listed rows are exactly the obligation surface. □

Theorem 2913.5 (MetricCompletionIdempotence downstream scope)**.** *Every completion-functor, regular-rational, or real-completion consumer of an accepted* MetricCompletionIdempotence$^{\uparrow}$ *carrier receives only the fixed-point route displayed by $M, S, C, U, R, H, K, P, N$. The packet supplies a non-quotient idempotence target for downstream completion functoriality and Real or* RegSeqRat$^{\uparrow}$ *readback routes, but it does not close arbitrary completion monad laws, global reflection theorems, selected-limit principles, host equality of completed points, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier rows and read the repeated-completion fixed point theorem Definition 3888.3. Any downstream consumer entering through the packet is already on the route $M, S, C, U, R, H, K, P, N$.

For functorial consumers, the route starts with the same MetricCompletion$^{\uparrow}$ row M and reaches a repeated completion only through R. No additional functorial law is stored in the packet.

For regular-rational and real-completion consumers, the readback data are inherited from the finite completion rows named by M and the complete-service row C. The packet names the fixed-point surface but does not add a new RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ completeness theorem.

The nonescape clause of Definition 3888.2 then excludes quotient equality, selected limits, host equality, choice, ambient completeness, and detached second completion data from all such consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ packet is generated from a MetricCompletionUp row, a SeparatedCompletionUp boundary row, a CompleteMetricUp or CompleteSeparatedMetricUp service row, a UniformCompletionUp compatibility row, a repeated-completion readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3888.1, Definition 3888.2, Definition 3888.3, and Definition 3888.4 close the obligation surface for repeated metric completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricCompletionIdempotenceUp.MetricCompletionIdempotenceTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close arbitrary completion monad laws, global reflection theorems, quotient completion equality, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicit downstream dependency scope for completion-functor, regular-rational, and real-completion consumers of BEDC.Derived.MetricCompletionIdempotenceUp.

Chapter 2914

A Concrete Naming Certificate for MetaCICFuelTrace$^{\uparrow}$

The MetaCICFuelTrace$^{\uparrow}$ packet realizes the bounded-normalizer side of chapter 9379 as finite NameCert data. It sits beside the decidable typing frontier of chapter 3616, the bounded normal-equality checker of chapter 752, the MetaCIC decidability witness of chapter 751, and the local diamond witness of chapter 3595. The packet names bounded normalizer fuel, finished-normalization evidence, equality-check rows, refusal rows, transport, replay, provenance, and local NameCert data without promoting bounded conversion into global normalization, subject reduction, confluence, or unrestricted beta-conversion decidability.

Falsifiable prediction: Every accepted MetaCICFuelTrace$^{\uparrow}$ packet displays a finite fuel row, bounded normalizer input and output rows, finished-normalization evidence when available, a `Term.eq` or `checkSameTerm` equality-check row, a `decideClosedBetaConv` readback row, a refusal row for global conversion claims, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: metacic-decidable-typing-frontier, bounded-normal-equality-checker, typed-normal-form-decider, metacicdecidableboundary, metacic-local-diamond-witness: the typing frontier records checker admission, the bounded-normal-equality checker records comparison exactness, the typed normal-form decider records typed endpoint conditions, the decidability boundary records broader refusal data, and the local diamond witness records join obligations. None supplies the finite fuel trace packet with both explicit fuel rows and replayable finished-normalization evidence.

Definition 2914.1 (MetaCIC fuel trace carrier)**.** *A* MetaCICFuelTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, I, O, E, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded fuel row used by* `normalizeBounded` *and* `decideClosedBetaConv`*.*
2. *I records the represented closed-term inputs and the typing or closedness hypotheses that the decidability audit map allows this packet to consume.*
3. *O records the bounded normalizer outputs produced at fuel F.*
4. *E records finished-normalization evidence when the audit map supplies it, and records an explicit absent row when such evidence is not available.*

5. *Q is the structural equality-check row, read through* `Term.eq` *or* `checkSameTerm` *on the bounded outputs.*
6. *B is the* `decideClosedBetaConv` *readback row, interpreted only as bounded normal equality under the displayed fuel and endpoint evidence.*
7. *R is the refusal row excluding global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and host quotient equality of terms.*
8. *H records componentwise* hsame *transport for the fuel, input, output, endpoint, equality-check, bounded-conversion, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a decidability request through $F, I, O, E, Q, B, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICFuelTrace}^{\uparrow}$*, the MetaCIC decidability audit map,* MetaCICDecidableTypingF BoundedNormalEqualityChecker$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a fuel-erased global normalizer, a subject-reduction theorem, a confluence theorem, an arbitrary closedness decider, unrestricted beta-conversion decidability, or host quotient equality of represented terms.

Theorem 2914.2 (MetaCIC fuel trace NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *carrier $T = (F, I, O, E, Q, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$ *surface consists exactly of fuel admission through F, represented-input admission through I, bounded normalizer output through O, finished-normalization evidence through E, structural equality checking through Q, bounded beta-conversion readback through B, refusal through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Later Lean TasteGate work in BEDC.Derived.MetaCICFuelTraceUp.TasteGate *must provide* Nontrivial *and* FieldFaithful *instances for this carrier surface. These instances must show that the packet cannot collapse to an empty fuel marker and cannot hide a missing fuel, input, output, finished-normalization, equality-check, bounded-conversion, refusal, transport, replay, provenance, or naming row.*

Proof. Project the carrier of Definition 3889.1. The computational side is F, I, O: a displayed fuel value, represented inputs, and the bounded normalizer outputs obtained at that fuel.

Read E before the conversion result. If finished-normalization evidence is available, it is displayed here; if not, the absent row keeps the bounded comparison from becoming a global normalization theorem.

The equality rows Q, B then compare only the bounded outputs. They are interpreted through the structural equality and `decideClosedBetaConv` rows named by the decidability audit map.

The refusal row R blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and host quotient equality. Finally H, C, P, N transport, replay, record provenance for, and locally name the same fuel trace. □

Lean4 ✓ `BEDC.Derived.MetaCICFuelTraceUp.MetaCICFuelTraceTasteGate single carrier alignment`

Theorem 2914.3 (MetaCIC fuel trace bounded-conversion nonescape)**.** *Every consumer of an accepted* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *packet factors through the finite route*

$$F, I, O, E, Q, B \longmapsto R, H, C, P, N.$$

The route exports bounded normalizer outputs, optional finished-normalization evidence, and bounded equality-check readback at the displayed fuel. It exports no fuel-erased global normalization theorem, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, no unrestricted beta-conversion oracle, and no host quotient equality of represented terms.

Proof. Start with an accepted carrier $T = (F, I, O, E, Q, B, R, H, C, P, N)$. Rows F, I, O expose the finite computation trace rather than a theorem about all possible fuels.

The endpoint row E is read before the equality-check rows. This preserves the distinction made in chapter 9379: bounded normal equality becomes a finished-normal-form statement only when finished evidence is part of the packet.

Rows Q and B compare the displayed bounded outputs. The refusal row R is then read before transport, replay, provenance, and naming finish the packet.

Because the carrier has no coordinate for stronger metatheory, no consumer can reinterpret the bounded fuel trace as global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality. □

Theorem 2914.4 (MetaCIC fuel trace audit scope)**.** *The* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *packet concretizes the normal-equality edge of the decidability audit map at finite fuel. It may be consumed by typing-frontier and bounded-normal-equality chapters as a replayable trace of bounded normalizer computation and finished endpoint evidence, but it does not close the global MetaCIC normalization, confluence, subject-reduction, or closedness-decidability problems.*

Proof. Use Definition 3889.2 to identify the local rows. The packet contains exactly the finite fuel, input, output, endpoint, equality-check, bounded-conversion, refusal, and structural rows.

For typing-frontier consumers, the input row I and equality rows Q, B align with the checked typing and structural equality surfaces, but the trace still carries a fixed fuel row F.

For bounded-normal-equality consumers, the output and endpoint rows O, E provide the replayable evidence required by finished-normalization consequences. They do not supply a fuel-erased endpoint theorem for all closed terms.

The nonescape theorem Definition 3889.3 then blocks the stronger global readings. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICFuelTrace}^{\uparrow}$ packet is generated from bounded fuel, represented inputs, bounded normalizer outputs, finished-normalization evidence or its explicit absence, structural equality checks, bounded beta-conversion readback, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3889.1, Definition 3889.2, and Definition 3889.3 seed the finite fuel-trace boundary for MetaCIC bounded normal equality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, host quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, fuel-output exactness, finished-normalization evidence handling, equality-check readback, bounded-conversion nonescape, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICFuelTraceUp.

Chapter 2915

A Concrete Naming Certificate for $\mathsf{CauchyStructure}^{\uparrow}$

Following Bourbaki and Kelley on Cauchy structures and Bishop–Bridges on constructive completion, the $\mathsf{CauchyStructure}^{\uparrow}$ packet names the finite criterion surface where Cauchy filters and Cauchy nets are read through stream windows, regular rational readback, dyadic tolerances, and a terminal real seal. It is grounded in the StreamName route of chapter 126, the regular sequence route of chapter 301, the real seal of chapter 38, the Cauchy-filter surface of chapter 333, the Cauchy-net surface of chapter 589, and the dyadic tolerance ledger of chapter 291.

Definition 2915.1 (Cauchy structure carrier)**.** *A* $\mathsf{CauchyStructure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the filter-base request row naming the finite family of Cauchy-filter requests accepted from chapter 333;*
2. *G is the net-directed request row naming the finite family of directed Cauchy-net requests accepted from chapter 589;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by both filter-base and net-directed requests;*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window inspected by every request before regular readback;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream window and dyadic ledger;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the displayed readback route;*
7. *H is componentwise* hsame *transport across F, G, D, S, R, E;*
8. *C is displayed* Cont *replay from each filter-base or net-directed request through D, S, R, E;*
9. *P is* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyStructureUp.CauchyStructureTasteGate single carrier alignment`

Theorem 2915.2 (Cauchy structure NameCert obligations)**.** *For the carrier of Definition 3890.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (F, G, D, S, R, E, H, C, P, N)$*, filter-base request exposure through* F*, net-directed request exposure through* G*, dyadic tolerance exposure through* D*, finite StreamName window exposure through* S*, RegSeqRat readback through* R*, Real seal handoff through* E*, transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier row by row. The filter-base and net-directed requests are the two input faces, and both are read only through the same dyadic tolerance ledger, StreamName window, regular rational readback, and Real seal route.

Transport and replay are structural rows. The transport row compares the displayed request, tolerance, window, readback, and seal coordinates componentwise; the replay row orders every request through D, S, R, E before a consumer may read the result.

The package provenance and local NameCert row name exactly this finite packet. Since the carrier contains no other coordinate, the listed rows are precisely the NameCert obligations. □

Theorem 2915.3 (Cauchy structure filter-net criterion)**.** *Every accepted* $\mathsf{CauchyStructure}^{\uparrow}$ *route orders filter-base and net-directed Cauchy reads through the same finite criterion sequence*

$$F, G \longmapsto D \longmapsto S \longmapsto R \longmapsto E.$$

A filter-facing consumer and a net-facing consumer therefore meet at the displayed dyadic tolerance ledger, StreamName window, RegSeqRat readback, and Real seal; neither route may replace that finite sequence by quotient filters, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the filter direction, project the F row and follow its replay row. The replay first opens the dyadic tolerance ledger D, then the finite StreamName window S, then the RegSeqRat readback R, and only then the Real seal E.

For the net direction, project the G row and use the same replay ordering. Directed requests may change the source face, but they do not introduce a new tolerance, window, readback, or seal coordinate outside D, S, R, E.

The componentwise transport row H identifies admissible changes only along the displayed coordinates, while P and N record provenance and local naming. Thus the common filter-net criterion is the finite route above, with no quotient, choice, trichotomy, selected-limit, or ambient-completeness bypass. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyStructure}^{\uparrow}$ packet is generated from filter-base requests, net-directed requests, dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$, seed)

Scope closed: Definition 3890.1, Definition 3890.2, and Definition 3890.3 bind the finite L10 criterion surface connecting Cauchy filters and Cauchy nets to StreamName, RegSeqRat, DyadicRatCore, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyStructureUp.CauchyStructureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite route ordering, and consumer nonescape for BEDC.Derived.CauchyStructureUp.

The $\mathsf{UniformContinuityComposition}^{\uparrow}$ hub routes composition through chapter 130, chapter 369, and the downstream compact-uniform-continuity consumer while the certificate body remains in the sibling files.

Chapter 2916

A Concrete Naming Certificate for $\mathsf{UniformContinuityComposition}^{\uparrow}$

Following Bishop and Bridges on constructive uniformly continuous maps, the $\mathsf{UniformContinuityComposition}^{\uparrow}$ packet records the finite route by which the composite of two uniformly continuous maps receives a displayed modulus. It sits after the continuous-map carrier of chapter 130, the uniform-modulus surface of chapter 369, the metric-space route of chapter 57, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet supplies downstream compact-uniform-continuity and completion-extension certificates with a reusable finite composition certificate; it does not import a host function space, selected moduli, quotient equality of maps, countable choice, or ambient completeness.

The composed-modulus row is a direct sibling input for the compact-uniform-continuity certificate of chapter 388, where finite compact-source moduli may consume composition without re-reading a host function-space theorem.

The $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumer link is available only after the two $\mathsf{ContinuousMap}^{\uparrow}$ rows F, G and the two $\mathsf{UniformModulus}^{\uparrow}$ rows M_F, M_G have been displayed on the same composed-modulus route.

The composed-modulus row therefore consumes exactly the two $\mathsf{ContinuousMap}^{\uparrow}$ rows of chapter 130 and the two $\mathsf{UniformModulus}^{\uparrow}$ rows of chapter 369 before chapter 388 may read it.

Before the compact route of chapter 388 may reuse the composite certificate, the row M must consume the finite tolerance-to-precision ledger of chapter 369 through M_G and then M_F. Thus the sibling edge records an ordered ledger read M_G, M_F, M, not a detached compactness theorem or a hidden modulus selector.

Definition 2916.1 (Uniform continuity composition carrier)**.** *A* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, F, G, M_F, M_G, M, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are the source, middle, and target* $\mathsf{Met}^{\uparrow}$ *rows. They provide the only metric contexts in which the two maps and the composite are read.*
2. *F is the first* $\mathsf{ContinuousMap}^{\uparrow}$ *row, displayed from X to Y.*
3. *G is the second* $\mathsf{ContinuousMap}^{\uparrow}$ *row, displayed from Y to Z.*

4. *M_F is the* $\mathsf{UniformModulus}^{\uparrow}$ *row for F, reading target tolerances in Y and returning source tolerances in X.*
5. *M_G is the* $\mathsf{UniformModulus}^{\uparrow}$ *row for G, reading target tolerances in Z and returning middle tolerances in Y.*
6. *M is the composed modulus row. A requested tolerance in Z is first sent to M_G, and the resulting tolerance in Y is then sent to M_F.*
7. *H records componentwise* hsame *transport for the metric, map, modulus, composite-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a composite-map request through G, M_G, F, M_F, and the composed modulus row M.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformContinuityComposition}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, Z, F, G, M_F, M_G, M, H, C, P, N$. It has no coordinate for host-level function equality, selected moduli, a quotient function space, countable choice, an ambient completion theorem, or hidden real equality.

Theorem 2916.2 (Uniform continuity composition NameCert obligations)**.** *For every accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier*

$$K = (X, Y, Z, F, G, M_F, M_G, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-context admission for X, Y, Z;*
2. *continuous-map admission for $F : X \to Y$ and $G : Y \to Z$ through the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *rows;*
3. *modulus admission for M_F and M_G through the displayed* $\mathsf{UniformModulus}^{\uparrow}$ *rows;*
4. *composed-modulus exactness for M, requiring each target tolerance in Z to be routed first through M_G and then through M_F;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently the accepted packet exports the constructive composition law for uniformly continuous maps only as finite BEDC data over the displayed rows. It does not export host-level function equality, selected moduli, quotient function spaces, countable choice, ambient completion, or hidden real equality.

Proof. Project the carrier of Definition 3891.1. The visible rows are exactly the three metric contexts, the two continuous maps, the two input moduli, the composed modulus, and the structural rows H, C, P, N.

The map side is read first. The rows F and G are continuous-map packets over the displayed metric contexts X, Y, Z, so a composite request has no map coordinate except the ordered route $X \xrightarrow{F} Y \xrightarrow{G} Z$.

The modulus side is downstream from the requested target tolerance. The row M_G converts a Z-tolerance into a Y-tolerance, and M_F converts that Y-tolerance into an X-tolerance. The row M records exactly this two-stage route.

Finally, H, C, P, N preserve componentwise comparison, replay the displayed request route, record provenance, and name the local certificate. Since the excluded analytic objects are not coordinates of K, the five listed obligation families exhaust the NameCert surface. □

Theorem 2916.3 (Uniform continuity composition modulus factorization)**.** *Every composed-modulus read accepted by a* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Met}^{\uparrow}(X, Y, Z), \quad \mathsf{ContinuousMap}^{\uparrow}(F), \quad \mathsf{UniformModulus}^{\uparrow}(M_F), \quad \mathsf{ContinuousMap}^{\uparrow}(G), \quad \mathsf{UniformModulus}^{\uparrow}(M_G),$$

Equivalently, a consumer that asks for a modulus of $G \circ F$ must first read the target tolerance through the G-modulus, then read the resulting middle tolerance through the F-modulus, and only then may expose the composite row M through H, C, P, N. No separate host composition theorem, selected modulus, quotient map equality, countable-choice selector, or ambient completeness coordinate is exported by the packet.

Proof. Begin with a consumer request for a modulus of the composite. Carrier admission supplies the metric contexts X, Y, Z and the two displayed map rows F, G, so the request is already confined to the finite route through the two continuous-map packets.

Read the second modulus before the first. A target tolerance for the composite is a Z-side request, so M_G is the first modulus row that can answer it. The answer is a displayed Y-side tolerance, not a selected global modulus or a host theorem.

Now read M_F. The Y-side tolerance returned by M_G becomes the input request for the first map, and M_F returns the source tolerance in X. The composite row M is exact because it stores no further tolerance coordinate beyond this two-step read.

The stream, regular-rational, and real rows appear only as observation and seal rows for the same finite route. The structural rows H, C, P, N transport, replay, source, and name that route, so there is no insertion point for quotient map equality, a choice-selected modulus, or an ambient completeness principle. □

Theorem 2916.4 (Uniform continuity composition L10 handoff)**.** *For accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *packets over the same displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, composition factors only through the stored modulus rows M_F, M_G, M and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *routes. The handoff exports no host function equality, selected modulus, quotient function space, hidden completion, countable-choice selector, or ambient real equality.*

Proof. Use the carrier definition of Definition 3891.1. The displayed rows fix the metric contexts, continuous-map rows, stored input moduli, composed modulus, transport, replay, provenance, and local naming.

The NameCert obligations of Definition 3891.2 force each composite request through the two continuous-map rows and the modulus rows. No external function-space equality is part of the obligation surface.

The modulus factorization theorem Definition 3891.3 gives the L10 order: a target tolerance is sent through M_G, the returned middle tolerance is sent through M_F, and the composed row M records exactly that finite route.

Finally H, C, P, N preserve the same route over the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. Since the excluded host objects have no coordinate in the carrier, the handoff is finite and classifier-based. □

Theorem 2916.5 (Uniform continuity composition sibling route)**.** *Every* CompactUniformContinuity$^{\uparrow}$ *consumer of chapter 388, and every completion-extension consumer using the same* UniformContinuityComposition$^{\uparrow}$ *packet, must first expose the two* ContinuousMap$^{\uparrow}$ *rows* F, G*, the two* UniformModulus$^{\uparrow}$ *rows* M_F, M_G*, the composed modulus row* M*, the source, middle, and target* Met$^{\uparrow}$ *rows* X, Y, Z*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. The sibling route exports no host function-space theorem, selected modulus, quotient map equality, countable-choice selector, or ambient completion principle.*

Proof. Project the carrier of Definition 3891.1. Its visible analytic rows are precisely $X, Y, Z, F, G, M_F, M_G, M$, with H, C, P, N carrying transport, replay, provenance, and naming.

Apply Definition 3891.3. A compact-uniform-continuity or completion-extension request cannot read the composed modulus until the target tolerance has passed through M_G and the returned middle tolerance has passed through M_F.

Use the L10 handoff of Definition 3891.4. The handoff keeps the same rows inside the displayed Met$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ route, so the consumer edge into chapter 388 is a numbered sibling route rather than a detached host theorem. Since the excluded objects are absent from the carrier, they cannot be exported by the route. □

Theorem 2916.6 (Uniform continuity composition compact-modulus consumer)**.** *Every* CompactUniformContinuit *or compact-modulus consumer of an accepted* UniformContinuityComposition$^{\uparrow}$ *packet enters through chapter 388 only after reading the source, middle, and target* Met$^{\uparrow}$ *rows, the two* ContinuousMap$^{\uparrow}$ *rows* F, G*, the two* UniformModulus$^{\uparrow}$ *rows* M_F, M_G*, the composed modulus row* M*, and then the* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* NameCert *routes. The consumer receives no host function-space theorem, no selected modulus, no quotient map equality, no hidden compactness theorem, and no ambient completion principle.*

Proof. Start with the carrier of Definition 3891.1. Its visible analytic coordinates are exactly $X, Y, Z, F, G, M_F, M_G, M$, followed by the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 3891.2. They force the compact-modulus request to pass through the two displayed maps F, G and the two displayed input moduli M_F, M_G before the composed modulus row is available.

Use the modulus factorization theorem Definition 3891.3. The compact consumer first sends the requested target tolerance through M_G, then sends the resulting middle tolerance through M_F, and only then reads the composed modulus row M.

Finally apply the L10 handoff of Definition 3891.4. The observation and seal rows StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ are downstream of the same finite route, and H, C, P, N preserve it. Thus the consumer edge into chapter 388 is a displayed lattice edge from the two map rows and two modulus rows, not a hidden compactness theorem. Since the excluded objects have no carrier coordinate, the compact-modulus consumer cannot obtain them from the packet. □

Theorem 2916.7 (Uniform continuity composition ContinuousMap route)**.** *Every compact-modulus or completion-extension consumer of an accepted* UniformContinuityComposition$^{\uparrow}$ *packet must first consume the two displayed* ContinuousMap$^{\uparrow}$ *rows* F, G *of chapter 130 together with their* UniformModulus$^{\uparrow}$ *rows* M_F, M_G *before it can read the composed modulus row used by chapter 388. The route exposes no host function-space theorem, quotient map equality, selected modulus, hidden compactness theorem, countable-choice selector, or ambient completion principle.*

Proof. Start from the NameCert obligations. They list continuous-map admission for $F : X \to Y$ and $G : Y \to Z$ and modulus admission for M_F and M_G as separate displayed rows before any composed modulus can be consumed.

Apply modulus factorization and the L10 handoff. A target tolerance is routed through M_G and then M_F, and the resulting composed row remains inside the finite $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff.

Finally use the compact-modulus consumer theorem. Since that consumer already has to read $X, Y, Z, F, G, M_F, M_G, M, H, C, P, N$, the dependency on ContinuousMap and CompactUniformContinuity is recorded as the displayed sibling route rather than as a kernel-only packet or host theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformContinuityComposition}^{\uparrow}$ packet is generated from source, middle, and target MetricSpace rows, two ContinuousMap rows, two UniformModulus rows, a composed modulus row, StreamName and RegSeqRat observation rows, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3891.1, Definition 3891.2, and Definition 3891.3 seed the finite constructive composition route over ContinuousMap, UniformModulus, MetricSpace, StreamName, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host-level function equality, selected moduli, quotient function spaces, countable choice, ambient completion, or hidden real equality.

Upgrade path: Obligation closure requires checked metric-context admission, continuous-map admission, modulus admission, composed-modulus exactness, observation routing, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformContinuityCompositionUp.

Theorem 2916.8 (Uniform continuity composition sibling dependency route)**.** *Every accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *read that exposes the composed modulus row must first consume the two* $\mathsf{ContinuousMap}^{\uparrow}$ *routes of chapter 130 and the two* $\mathsf{UniformModulus}^{\uparrow}$ *routes of chapter 369. The handoff reaches the composed row only through the displayed* F, G, M_F, M_G *coordinates and the structural* H, C, P, N *route; it exports no kernel-only composition fact, host function-space theorem, quotient map equality, selected modulus, countable-choice selector, or ambient completion principle.*

Proof. Start with the carrier of Definition 3891.1. Its analytic input rows are exactly the three metric contexts, the two continuous-map rows F, G, the two input modulus rows M_F, M_G, and the composed modulus row M.

Apply the NameCert obligations of Definition 3891.2. They make continuous-map admission and modulus admission separate displayed obligations before the composed row can be read.

Finally use the modulus factorization route of Definition 3891.3. A target tolerance enters through the G-modulus and then the F-modulus, so the sibling chapters supply the exact dependency route rather than a hidden host theorem. □

Chapter 2917

A Concrete Naming Certificate for $\mathsf{HadamardThreeCircle}^{\uparrow}$

Following Hadamard's three-circle theorem and the classical complex-analysis reading of maximum-modulus bounds on annuli, the $\mathsf{HadamardThreeCircle}^{\uparrow}$ packet records a finite annulus ledger whose radius reads, boundary maxima, and logarithmic-convexity row are all displayed before a holomorphic-boundary consumer may read a bound. It sits over the complex carrier of chapter 39, the circle boundary surface of chapter 58, the Real seal of chapter 38, the holomorphic carrier of chapter 68, the finite integral row of chapter 123, and the metric annulus packet of chapter 3371. The chapter names the finite three-radius certificate only; it does not assert an ambient analytic continuation theorem, a selected maximizer, quotient function equality, or a bridge certificate.

Definition 2917.1 (Hadamard three-circle carrier)**.** *A* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N)$$

with the following displayed rows:

1. *A is the displayed annulus row, read through* $\mathsf{MetricAnnulus}^{\uparrow}$ *and carrying the two boundary circles and the inspected middle radius.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row over the annulus, with values in the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *R_0, R_1, R_2 are the three positive* $\mathsf{Real}^{\uparrow}$ *radius reads, ordered by the annulus row and sealed before any maximum row is consumed.*
4. *B is the boundary-circle readback row, using* $\mathsf{S}^{1\uparrow}$ *to expose the inner, middle, and outer circle probes.*
5. *M is the maximum-modulus row for the displayed boundary reads, recording only finite upper-bound data on the inspected circles.*
6. *L is the logarithmic-convexity ledger tying the three radius reads and the corresponding maximum-modulus rows to a single finite inequality certificate.*
7. *Hs records componentwise* hsame *transport for the annulus, holomorphic map, radius reads, boundary reads, maximum rows, logarithmic-convexity ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an annulus-bound request through $A, F, R_0, R_1, R_2, B, M, L, Hs$.*

9. *P is the* Pkg *provenance over* $\mathsf{HadamardThreeCircle}^{\uparrow}$, $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host supremum theorem, an uninspected boundary point, a selected maximizer, quotient function equality, analytic continuation outside the annulus, or countable choice.

Lean4 ✓ `BEDC.Derived.HadamardThreeCircleUp.HadamardThreeCircleTasteGate single carrier alignment`

Theorem 2917.2 (Hadamard three-circle NameCert obligations)**.** *For every accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier* $H = (A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$ *surface is exhausted by annulus admission, holomorphic-map exposure, three sealed radius reads, boundary-circle readback, maximum-modulus rows, logarithmic-convexity ledger compatibility, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet may not infer an ambient maximum principle, a selected extremal point, analytic continuation, quotient function equality, or any infinite covering principle from the local certificate.*

Proof. Project the carrier of Definition 3892.1. The rows A, F, R_0, R_1, R_2 display the annulus, the holomorphic map, and the three radius reads before a bound can be consumed.

The boundary computation is then finite. The row B exposes the inspected $\mathsf{S}^{1\uparrow}$ circle probes, M records the corresponding maximum-modulus upper-bound rows, and L binds those rows to the logarithmic-convexity ledger for the same three radii.

Finally the structural rows Hs, C, P, N transport, replay, cite provenance, and name that same finite route. Since none of these rows adds a host supremum theorem, a selected maximizer, an uninspected boundary point, analytic continuation, quotient equality, or countable choice, the listed obligations exhaust the local NameCert surface. □

Theorem 2917.3 (Hadamard annulus maximum handoff)**.** *Every holomorphic-boundary consumer of an accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier factors through*

$$\mathsf{MetricAnnulus}^{\uparrow} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{Complex}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

together with the displayed rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$. The handoff exports a finite three-radius annulus ledger and its logarithmic-convexity certificate, not a global maximum-modulus theorem or an ambient analytic object outside the displayed annulus.

Proof. Start from the annulus row. The packet reads the displayed $\mathsf{MetricAnnulus}^{\uparrow}$ data A and the $\mathsf{Holomorphic}^{\uparrow}$ map row F before any boundary maximum is available.

Next seal the radius and boundary rows. The three $\mathsf{Real}^{\uparrow}$ reads R_0, R_1, R_2 determine the inspected circles, and B turns those circles into finite $\mathsf{S}^{1\uparrow}$ boundary probes over the same annulus.

The maximum handoff is then local. The row M records the finite boundary upper bounds, while L ties them to the logarithmic-convexity inequality for the same radii. The $\mathsf{Integral}^{\uparrow}$ dependency is consumed only as a displayed finite contour or averaging row, not as an ambient selection principle.

The transport and naming rows Hs, C, P, N preserve this route. Therefore a consumer receives exactly the finite annulus ledger and no global maximum principle, selected extremal point, analytic continuation coordinate, quotient equality, or infinite covering data. □

Theorem 2917.4 (Hadamard three-circle maximum-modulus boundary)**.** *Let $H = (A, F, R_0, R_1, R_2, B, M, L, Hs,$ be an accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier. If the inner and outer boundary maximum ledgers in M display the same* $\mathsf{Real}^{\uparrow}$ *modulus row, then the middle-radius bound classified by L is read through that same displayed boundary modulus row. The conclusion uses only the finite annulus ledger, the* $\mathsf{Real}^{\uparrow}$ *modulus reads, and the* $\mathsf{Holomorphic}^{\uparrow}$ *source rows; it does not select a maximizer, add ambient analytic continuation, or import a host maximum-modulus theorem.*

Proof. Start with Definition 3892.3. It forces every holomorphic-boundary read to pass through the annulus row A, holomorphic row F, radius reads R_0, R_1, R_2, boundary probes B, maximum rows M, and logarithmic-convexity ledger L.

Read the hypothesis inside M. The inner and outer boundary maximum ledgers display the same $\mathsf{Real}^{\uparrow}$ modulus row, so the two endpoint values supplied to the logarithmic-convexity ledger are the same boundary datum.

The ledger L then classifies the middle-radius bound using that displayed endpoint row. Since the route is still the finite annulus route named by $A, F, R_0, R_1, R_2, B, M, L$, the conclusion has no coordinate from which to select an extremal point or extend the holomorphic source.

Finally Hs, C, P, N transport, replay, source, and name the same certificate. They preserve the finite boundary-modulus read and do not add ambient analytic continuation, quotient function equality, or a host maximum-modulus principle. □

Theorem 2917.5 (Hadamard three-circle annulus route)**.** *Every accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *annulus-bound consumer factors first through*

$$\mathsf{MetricAnnulus}^{\uparrow} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{Complex}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

and then through the displayed rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$. This route exports only the finite annulus ledger, holomorphic source row, complex value carrier, circle boundary probes, real radius and modulus reads, integral averaging surface, maximum-modulus boundary rows, logarithmic-convexity ledger, transport, replay, provenance, and local name. It excludes selected maximizers, global maximum principles, analytic continuation beyond the annulus, quotient function equality, infinite covers, and bridge certificates.

Proof. Apply the annulus maximum handoff of Definition 3892.3. It already orders the cross-chapter surfaces as $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Integral}^{\uparrow}$, and it exposes the same rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$ before any annulus-bound read is consumed.

Apply the boundary theorem of Definition 3892.4 to the maximum-modulus part of the route. The boundary theorem keeps the middle-radius conclusion inside the displayed boundary-modulus row and the finite logarithmic-convexity ledger, so the composed route has no coordinate for a selected maximizer, an ambient maximum principle, analytic continuation outside the annulus, quotient equality, an infinite cover, or a bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{HadamardThreeCircle}^{\uparrow}$ read is refuted if the logarithmic-convexity row L is preserved while the displayed annulus row A, holomorphic map row F, sealed radii R_0, R_1, R_2, boundary-circle row B, and maximum-modulus row M are removed.

Independence witness: complex, sone, real, holomorphic, integral, metric-annulus: none is bijective with $\mathsf{HadamardThreeCircle}^{\uparrow}$ because the packet jointly carries annulus rows, holomorphic map rows, three sealed radius reads, boundary circle probes, maximum-modulus rows, a logarithmic-convexity ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardThreeCircle}^{\uparrow}$ packet is generated from a displayed annulus, a holomorphic map row, three Real radius reads, boundary circle probes, maximum-modulus rows, a logarithmic-convexity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3892.1, Definition 3892.2, and Definition 3892.3 seed the finite annulus ledger for three-circle maximum-modulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global maximum-modulus theorem, selected extremal points, analytic continuation beyond the annulus, quotient function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked annulus admission, holomorphic-map exposure, three-radius ordering, boundary-circle readback, maximum-modulus rows, logarithmic-convexity compatibility, and structural transport for BEDC.Derived.HadamardThreeCircleUp.

Chapter 2918

A Concrete Naming Certificate for $\mathsf{HadamardThreeLines}^{\uparrow}$

Following Hadamard's three-lines theorem and the Riesz-Thorin style reading of holomorphic strip estimates, the $\mathsf{HadamardThreeLines}^{\uparrow}$ packet records a finite vertical-strip ledger whose boundary-line reads, interior height read, and logarithmic-convexity bound are all displayed before a strip-bound consumer may read an estimate. It sits beside the annular packet of chapter 3892, over the holomorphic carrier of chapter 68, the complex carrier of chapter 39, the Real seal of chapter 38, the finite integral row of chapter 123, and the metric annulus packet of chapter 3371. The chapter names the finite strip-bound certificate only; it does not collapse strip geometry into an annulus certificate, assert host supremum selection, or add a bridge certificate.

Definition 2918.1 (Hadamard three-lines carrier)**.** *A* $\mathsf{HadamardThreeLines}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N)$$

with the following displayed rows:

1. *S is the displayed vertical-strip row, exposing the two boundary lines and the inspected interior line.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row on the strip, with values in the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *Y_0, Y_1, Y are the two boundary-height* $\mathsf{Real}^{\uparrow}$ *reads and the inspected interior-height* $\mathsf{Real}^{\uparrow}$ *read, ordered by the strip row before any bound is consumed.*
4. *B is the boundary-line readback row, exposing the finite probes on the two boundary lines and the requested interior line.*
5. *M is the finite boundary-majorant row, recording only displayed upper-bound data for the boundary reads.*
6. *E is the logarithmic-convexity estimate row tying Y_0, Y_1, Y and M to a single finite strip-bound certificate.*
7. *Hs records componentwise* hsame *transport for the strip row, holomorphic map row, height reads, boundary reads, majorant row, estimate row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a strip-bound request through $S, F, Y_0, Y_1, Y, B, M, E, Hs$.*
9. *P is the* Pkg *provenance over* $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{HadamardThreeCircle}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{MetricAnnulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^\uparrow}$ *row.*
The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient function equality, a selected boundary supremum, an unbounded strip limit, countable choice, ambient Real completeness, or an annulus replacement of the strip route.

Lean4 ✓ `BEDC.Derived.HadamardThreeLinesUp.HadamardThreeLinesTasteGate single carrier alignment`

Theorem 2918.2 (Hadamard three-lines NameCert obligations)**.** *For every accepted* $\mathsf{HadamardThreeLines}^\uparrow$ *carrier* $L = (S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^\uparrow}$ *surface is exhausted by strip admission, holomorphic-map exposure, two boundary-height reads, one interior-height read, boundary-line readback, finite boundary-majorant rows, logarithmic-convexity estimate compatibility, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet may not infer quotient function equality, host function extensionality, selected boundary suprema, unbounded strip limiting, countable choice, ambient* $\mathsf{Real}^\uparrow$ *completeness, or a collapse into* $\mathsf{HadamardThreeCircle}^\uparrow$*.*

Proof. Project the carrier of Definition 3893.1. The rows S, F, Y_0, Y_1, Y display the strip, the holomorphic map, and the three height reads before an estimate can be consumed.

The boundary computation is finite. The row B exposes the inspected boundary-line probes, M records their finite majorant data, and E binds those rows to the logarithmic-convexity estimate for the same boundary and interior heights.

Finally the structural rows Hs, C, P, N transport, replay, cite provenance, and name that same finite route. Since none of these rows adds quotient function equality, host extensionality, selected suprema, an unbounded limiting principle, countable choice, ambient Real completeness, or an annulus replacement coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 2918.3 (Hadamard three-lines strip-bound route)**.** *Every strip-bound consumer of an accepted* $\mathsf{HadamardThreeLines}^\uparrow$ *carrier factors through*

$$\mathsf{Holomorphic}^\uparrow \longmapsto \mathsf{Complex}^\uparrow \longmapsto \mathsf{Real}^\uparrow \longmapsto \mathsf{Integral}^\uparrow$$

together with the displayed rows $S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N$*. The route exports a finite vertical-strip ledger and its logarithmic-convexity estimate, not the annulus ledger of* $\mathsf{HadamardThreeCircle}^\uparrow$*, not a* $\mathsf{MetricAnnulus}^\uparrow$ *maximum handoff, and not an ambient analytic object outside the displayed strip.*

Proof. Start from the strip row. The packet reads the displayed strip data S and the $\mathsf{Holomorphic}^\uparrow$ map row F before any boundary-line estimate is available.

Next seal the height and boundary rows. The $\mathsf{Real}^\uparrow$ reads Y_0, Y_1, Y determine the inspected boundary and interior lines, while B turns those lines into finite boundary probes.

The strip-bound handoff is then local. The row M records the finite boundary majorants, and E ties them to the logarithmic-convexity estimate for the same heights. The $\mathsf{Integral}^\uparrow$ dependency is consumed only as a displayed finite contour, averaging, or boundary-read row, not as an ambient selection principle.

The provenance row may cite $\mathsf{HadamardThreeCircle}^\uparrow$ and $\mathsf{MetricAnnulus}^\uparrow$ as neighboring complex-analysis certificates, but the carrier keeps its strip row S and height ledger Y_0, Y_1, Y. Therefore a consumer receives exactly the finite strip ledger and no annulus collapse, selected extremal point, quotient equality, or infinite limiting data. □

Theorem 2918.4 (Hadamard three-lines convexity window)**.** *Any strip-bound consumer that reads the logarithmic-convexity estimate row E of an accepted* $\mathsf{HadamardThreeLines}^{\uparrow}$ *carrier must also display the strip row S, holomorphic map row F, boundary and interior height rows Y_0, Y_1, Y, boundary-line readback row B, and finite majorant row M. Thus the convexity estimate cannot be detached from the finite vertical-strip ledger that supplies its boundary and interior witnesses.*

Proof. Project the carrier in Definition 3893.1. The row E is introduced only as the logarithmic-convexity estimate tying Y_0, Y_1, Y and M to the same displayed strip-bound certificate, while M itself records finite boundary-majorant data read from B.

Apply the strip-bound route of Definition 3893.3. That theorem sends every accepted consumer through $S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N$ before exporting a bound. Hence a read of E without S, F, Y_0, Y_1, Y, B, M would no longer be a read of the accepted $\mathsf{HadamardThreeLines}^{\uparrow}$ packet, but an external estimate outside the displayed finite ledger. □

Falsifiable prediction: An accepted $\mathsf{HadamardThreeLines}^{\uparrow}$ read is refuted if the logarithmic-convexity estimate row E is preserved while the displayed strip row S, holomorphic map row F, boundary and interior height reads Y_0, Y_1, Y, boundary-line row B, and finite majorant row M are removed.

Independence witness: hadamard-three-circle, holomorphic, complex, real, integral, metric-annulus: none is bijective with $\mathsf{HadamardThreeLines}^{\uparrow}$ because the packet jointly carries a vertical-strip row, holomorphic map row, boundary and interior height reads, boundary-line probes, finite majorant rows, a logarithmic-convexity estimate ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardThreeLines}^{\uparrow}$ packet is generated from a displayed vertical strip, a holomorphic map row, two boundary-height Real reads, one interior-height Real read, boundary-line probes, finite majorant rows, a logarithmic-convexity estimate ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3893.1, Definition 3893.2, and Definition 3893.3 seed the finite vertical-strip ledger for three-lines holomorphic strip estimates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient function equality, host function extensionality, selected boundary suprema, unbounded strip limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked strip admission, holomorphic-map exposure, boundary and interior height ordering, boundary-line readback, finite majorant rows, logarithmic-convexity estimate compatibility, and structural transport for BEDC.Derived.HadamardThreeLinesUp.

Chapter 2919

A Concrete Naming Certificate for $\mathsf{CauchyLocatedApartness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyLocatedApartness}^{\uparrow}$ packet records the constructive boundary where finite locator requests for a Cauchy real become positive located-apartness evidence. It consumes the Cauchy locator route of chapter 3410, the located-apartness surface of chapter 1380, the apartness-real interface of chapter 304, and the real-name seal of chapter 38. Its finite naming spine is read through chapter 126, chapter 301, and chapter 291.

The packet is a local handoff surface. It exports locator rows, endpoint-gap rows, finite stream windows, regular-rational readbacks, dyadic tolerance ledgers, real seals, transport rows, continuation replay, provenance, and a local NameCert row. It exports no trichotomy, no quotient real equality, no selected limit, no countable choice, no open-cover compactness, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 2919.1 (Cauchy located-apartness carrier)**.** *A* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, A, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row answering finite rational locator requests for the Cauchy real being inspected.*
2. *A and B are endpoint-gap rows read through* $\mathsf{LocatedApartness}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$*, recording positive separation windows on the two sides of the located comparison.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger used to schedule the locator observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed stream windows into regular rational observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the radius and endpoint arithmetic used by the locator and apartness rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, A, B, S, R, D have been displayed.*
7. *H stores componentwise* hsame *transport for locator rows, endpoint-gap rows, finite windows, readbacks, tolerance ledgers, real seals, continuation replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from locator requests through stream windows, regular-rational readbacks, dyadic tolerance checks, endpoint-gap rows, and the final real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyLocatedApartness}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, quotient real equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessTasteGate single carrier alignment`

Theorem 2919.2 (Cauchy located-apartness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier* $L = (Q, A, B, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, locator-window compatibility, endpoint-gap exactness, dyadic-tolerance coverage, real-seal handoff, transport replay, and non-escape from the finite route*

$$Q, S, R, D \longmapsto A, B \longmapsto E$$

together with H, C, P, N*.*

Proof. Carrier admission is projection from Definition 3895.1. The accepted tuple exposes the Cauchy locator row, two endpoint-gap rows, stream windows, regular-rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming.

Locator-window compatibility reads Q only through S, R, D. A finite comparison request must display the scheduled stream window, its regular rational readback, and the tolerance ledger before any located-apartness row can be consumed.

Endpoint-gap exactness is supplied by A and B. These rows carry the positive located-apartness evidence, while the real seal E is downstream of the same locator and tolerance route.

Finally, H, C, P, N preserve and name the route. Since the carrier contains no trichotomy, quotient equality, selected limit, countable-choice schedule, open-cover compactness theorem, or ambient real-completeness coordinate, the displayed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessCarrier namecert obligations`

Theorem 2919.3 (Cauchy locator to apartness handoff)**.** *Every accepted* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier converts a finite* $\mathsf{CauchyRealLocator}^{\uparrow}$ *request into positive* $\mathsf{LocatedApartness}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$ *endpoint-gap evidence only through the displayed route*

$$Q \longmapsto S \longmapsto R \longmapsto D \longmapsto A, B \longmapsto E.$$

Thus a downstream Real-facing comparison consumer receives a finite locator-apartness handoff, not trichotomy, quotient real equality, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessLocatorApartnessHandoff`

Proof. Start with a finite locator request in Q. The carrier forces the request to replay through the scheduled $\mathsf{StreamName}^{\uparrow}$ window S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D.

Only after that route is displayed may the endpoint-gap rows A and B be read as located-apartness evidence. The $\mathsf{Real}^{\uparrow}$ seal E records that the same finite route reaches the real-facing surface.

The structural rows H, C, P, N transport, replay, package, and name exactly these coordinates. No excluded principle is a field of the carrier, so the handoff is exhausted by the finite locator, window, readback, tolerance, apartness, and seal rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3895.1, Definition 3895.2, and Definition 3895.3 seed the finite Cauchy locator to located-apartness boundary over stream windows, regular-rational readbacks, dyadic tolerance ledgers, endpoint-gap rows, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locator-window compatibility, endpoint-gap exactness, dyadic-tolerance coverage, real-seal handoff, transport replay, and non-escape for BEDC.Derived.CauchyLocatedApartnessUp.

Constructive story: A $\mathsf{CauchyLocatedApartness}^{\uparrow}$ packet is generated from a CauchyRealLocator request row, LocatedApartness and ApartnessReal endpoint-gap rows, finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2920

A Concrete Naming Certificate for $\mathsf{RiemannLebesgueLemma}^{\uparrow}$

Following Zygmund, Katznelson, and Folland on Fourier analysis, the $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet records the finite oscillatory-decay surface of the Riemann–Lebesgue lemma as BEDC NameCert data. It sits over the Fourier coefficient interface of chapter 124, the circle phase sampler of chapter 58, the integral normalization route of chapter 123, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the averaging comparison supplied by chapter 1939. The packet exposes finite coefficient windows and oscillatory handoff data; it does not import Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2920.1 (Riemann–Lebesgue lemma carrier)**.** *A* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, S, I, W, T, A, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row, naming exactly the inspected frequency indices and coefficient requests.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-sample row, recording the finite circle phases used to test the requested oscillations.*
3. *I is the* $\mathsf{Integral}^{\uparrow}$ *normalization and readback row, exposing only the finite integral observations used to form each coefficient.*
4. *W is the window ledger pairing frequency indices, phase samples, and integral cells before any decay read is made.*
5. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, listing the rational error budgets attached to the coefficient windows.*
6. *A is the finite averaging comparison row, allowed to cite the* $\mathsf{FejerKernel}^{\uparrow}$ *sibling only as a displayed smoothing comparison.*
7. *Q is the oscillatory-decay handoff row, recording which finite tails have coefficient magnitude below the displayed tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for coefficient values, tolerance comparisons, and accepted decay reads.*

9. *H records componentwise* hsame *transport for the coefficient, phase, integral, window, tolerance, averaging, decay, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Fourier-coefficient request through $F, S, I, W, T, A, Q, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, I, W, T, A, Q, E, H, C, P, N$. It has no coordinate for Haar measure, host L^1 spaces, quotient equality of functions, a theorem of general Fourier convergence, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RiemannLebesgueLemmaUp.RiemannLebesgueLemmaTasteGate single carrier alignment`

Theorem 2920.2 (Riemann–Lebesgue lemma NameCert obligations)**.** *For every accepted* RiemannLebesgueLemma *carrier*

$$R = (F, S, I, W, T, A, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$ *surface consists exactly of Fourier coefficient-window admission through F, circle phase-sample admission through S, integral normalization and readback through I, window and tolerance exposure through W, T, finite averaging comparison through A, oscillatory-decay handoff through Q, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3896.1. The rows F, S, I expose the coefficient request, the finite phase samples, and the integral normalization before the packet can state any decay handoff.

The rows W, T, A, Q supply the finite analytic body of the certificate. The window row pairs frequencies with inspected cells, the tolerance row records the rational budgets, the averaging row gives the displayed Fejer-kernel comparison, and the handoff row records exactly the tails accepted below those budgets.

The structural rows E, H, C, P, N seal, transport, replay, source, and name the same data. Since the carrier has no coordinate for Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, selected representatives, or ambient real completeness, the listed rows exhaust the NameCert surface. □

Theorem 2920.3 (Riemann–Lebesgue oscillatory handoff)**.** *Every Fourier-decay consumer of an accepted* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Fourier}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The optional $\mathsf{FejerKernel}^{\uparrow}$ *comparison can be read only through the finite averaging row A, and the final decay read can inspect only $F, S, I, W, T, A, Q, E, H, C, P, N$. Thus the packet supplies a finite oscillatory-decay dependency for later Fourier approximation chapters without exporting Haar measure, host L^1 integration, quotient function equality, general convergence of Fourier series, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Begin with a Fourier-decay request. Carrier projection exposes F, so the consumer receives a finite coefficient window rather than an infinite Fourier transform.

The request next reads S, I, W, T. These rows bind each requested frequency to displayed circle phases, integral cells, window ledgers, and regular rational tolerances.

Only then can the consumer inspect A, Q, E, H, C, P, N. The averaging comparison remains a finite sibling comparison, the decay handoff is recorded at the displayed tolerances, and the real seal closes the numeric read. No excluded analytic object has a coordinate in the packet, so the handoff is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RiemannLebesgueLemmaUp.OscillatoryHandoff.RiemannLebesgueLemmaOscillatoryHandoff`

Falsifiable prediction: An accepted $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet is refuted if a Fourier-decay consumer preserves the coefficient window while bypassing the displayed $\mathsf{S}^{1\uparrow}$ phase samples, $\mathsf{Integral}^{\uparrow}$ normalization row, $\mathsf{RegSeqRat}^{\uparrow}$ tolerance row, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: fourier, sone, integral, regseqrat, real, fejer-kernel: none is bijective to $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ because the packet jointly carries coefficient windows, phase samples, integral normalization, tolerance ledgers, averaging comparison, oscillatory-decay handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet is generated from finite Fourier coefficient windows, SOne phase samples, Integral normalization and readback rows, RegSeqRat tolerance rows, a finite FejerKernel comparison, an oscillatory handoff row, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3896.1, Definition 3896.2, and Definition 3896.3 seed the finite oscillatory-decay route over Fourier, circle phase, integral, regular rational, real-seal, and averaging rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RiemannLebesgueLemmaUp.OscillatoryHandoff.RiemannLebesgueLemmaOscillatoryHandoff`

Bridge status: none

Not claimed: This chapter does not claim Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coefficient-window admission, phase-sample exposure, integral normalization, tolerance exactness, finite averaging comparison, oscillatory handoff, real seal exposure, transport, replay, provenance, and local naming for BEDC.Derived.RiemannLebesgueLemmaUp.

Chapter 2921

A Concrete Naming Certificate for $\mathsf{MedianClosureSpectrum}^{\uparrow}$

The $\mathsf{MedianClosureSpectrum}^{\uparrow}$ packet realizes the Walsh-spectrum reassignment rows of Definition 9462.1263, Definition 9462.1264, and Definition 9462.1273 as a finite NameCert spectrum. It reads the finite codon-window and module-boundary surfaces of **??** and **??**, then records the dominant Walsh coordinates as median-closure support data rather than as host spectral analysis. The packet does not depend on Fourier analysis, Fejer kernels, Parseval identities, finite phase sampling, or Dirichlet kernels.

Definition 2921.1 (Median closure spectrum carrier)**.** *A* $\mathsf{MedianClosureSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, W, D, S, L, E, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row whose coordinates are read by the reassignment spectrum.*
2. *W is the Walsh coefficient ledger, listing the displayed coefficients of Definition 9462.1264.*
3. *D is the dominant-coordinate row for b_3^s, $b_1^f b_3^s$, b_1^f, $b_1^f b_2^s$, $b_1^f b_2^s b_3^s$, $b_2^s b_3^s$, and b_2^s.*
4. *S is the median-closure support row, recording which dominant coordinates support the reassignment median boundary.*
5. *L is the finite low-order energy ledger read from the compression statement, with no appeal to an analytic Fourier transform.*
6. *E is the evidence row tying the Walsh ledger to the purine-core and module-classification surfaces.*
7. *R is the refusal row for Fourier, Fejer-kernel, Parseval, finite-phase-sampling, and Dirichlet-kernel dependencies.*
8. *H records componentwise* hsame *transport for the codon window, Walsh ledger, dominant-coordinate row, support row, energy ledger, evidence row, refusal row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a dominant-coordinate read through Q, W, D, S, L, E, R, H.*
10. *P is the* Pkg *provenance over* $\mathsf{MedianClosureSpectrum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *together with the cited vision rows and sibling codon-boundary packets.*

11. N is the local $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, W, D, S, L, E, R, H, C, P, N$. It has no coordinate for host Fourier transform data, trigonometric phase sampling, Parseval energy identities, Dirichlet kernels, Fejer kernels, or a hidden analytic spectrum outside the displayed Walsh ledger.

Lean4 ✓ `BEDC.Derived.MedianClosureSpectrumUp.MedianClosureSpectrumTasteGate single carrier alignment`

Theorem 2921.2 (Median closure spectrum NameCert obligations)**.** *For every accepted* $\mathsf{MedianClosureSpectrum}^{\uparrow}$ *carrier*

$$M = (Q, W, D, S, L, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$ *surface consists exactly of codon-window admission through Q, Walsh coefficient exposure through W, dominant-coordinate readback through D, median-closure support exposure through S, low-order energy accounting through L, purine-core and module-classification evidence through E, analytic-dependency refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3897.1. The rows Q, W, D expose the finite codon window, Walsh coefficient ledger, and dominant coordinates before any median-closure support read is allowed.

The rows S, L, E supply the support body of the packet. They identify the median-closure support row, record the low-order energy ledger, and tie those rows to the purine-core and module-classification siblings cited above.

The rows R, H, C, P, N close the certificate. The refusal row blocks analytic substitutions, while transport, replay, provenance, and local naming preserve the displayed finite Walsh spectrum as the whole accepted surface. □

Falsifiable prediction: Every dominant-coordinate read accepted by $\mathsf{MedianClosureSpectrum}^{\uparrow}$ factors through the displayed Walsh coefficient ledger W; replacing W by Fourier, phase-sampling, Parseval, Fejer-kernel, or Dirichlet-kernel data refutes the packet.

Independence witness: fourier, fejer-kernel, parseval, finite-phase-sampling, dirichlet-kernel: the median-closure spectrum is not bijective to any of these routes because its accepted reads are finite Walsh coefficient ledger reads over codon-window and support rows, not analytic phase, convolution, energy, or kernel reads.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{MedianClosureSpectrum}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MedianClosureSpectrum}^{\uparrow}$ instances in `TasteGate.lean`. The field-faithfulness witness must keep $Q, W, D, S, L, E, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out analytic-spectrum replacements of the displayed Walsh ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MedianClosureSpectrum}^{\uparrow}$ packet is generated from finite codon-window rows, Walsh coefficient ledgers, dominant-coordinate rows, median-closure support rows, low-order energy rows, purine-core and module-classification evidence, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3897.1 and Definition 3897.2 seed the finite Walsh-ledger spectrum packet over codon-window, dominant-coordinate, median-support, energy, evidence, refusal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MedianClosureSpectrum}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Fourier analysis, Fejer kernels, Parseval identities, finite phase sampling, Dirichlet kernels, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, Walsh ledger exposure, dominant-coordinate readback, median-support exactness, energy ledger accounting, evidence routing, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MedianClosureSpectrumUp.

Chapter 2922

A Concrete Naming Certificate for $\mathsf{DirectedFilter}^{\uparrow}$

Following Kelley and Bourbaki on directed sets, filters, and completion by tails, the $\mathsf{DirectedFilter}^{\uparrow}$ packet records the finite carrier that translates directed tail requests into a filter-base handoff before any completion consumer reads a limit-facing surface. It sits between the directed-set substrate of chapter 1607, the filter-base certificate of chapter 1601, the Moore–Smith net route of chapter 3403, and the filter-limit basis route of chapter 3862. The packet supplies the L10 completion corridor with a shared directed-filter carrier; it is not a quotient filter, selected limit, countable-choice selector, host equality of filters, or ambient completeness theorem.

Definition 2922.1 (DirectedFilter carrier)**.** *A* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, B, M, U, W, R, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *tail-request row, recording finite requests, preorder comparisons, and common-upper witnesses.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row assembled only from the displayed tails in D.*
3. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *and* $\mathsf{NetConvergence}^{\uparrow}$ *route row, exposing the same directed tails as net-index evidence before any convergence read.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row, available only after D, B, M have been replayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by completion-facing numerical consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W.*
7. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback rows before a seal is consumed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, downstream of directed tails, filter-base handoff, net convergence, completion consumers, windows, readback, and tolerance rows.*
9. *H records componentwise* hsame *transport for all analytic rows and the local structural rows.*
10. *C records displayed* Cont *replay from a directed-tail request through $D, B, M, U, W, R, Y, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{DirectedFilter}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for quotient filters, ultrafilters, selected limits, countable choice, host filter equality, or ambient completion.

Lean4 ✓ `BEDC.Derived.DirectedFilterUp.DirectedFilterTasteGate single carrier alignment`

Theorem 2922.2 (DirectedFilter NameCert obligations)**.** *For every accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier* $F = (D, B, M, U, W, R, Y, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$ *surface consists exactly of the finite obligations for carrier admission, directed-tail coverage, filter-base handoff, net-convergence routing, completion nonescape, numerical readback, componentwise transport, displayed replay, provenance, and local naming. Every obligation is read from* $D, B, M, U, W, R, Y, E, H, C, P, N$*, and no obligation is stored in a quotient filter, selected limit, countable-choice schedule, host equality relation, or ambient completeness witness.*

Proof. Project the carrier of Definition 3898.1. The displayed analytic rows are D, B, M, U, W, R, Y, E, and the structural rows are H, C, P, N.

Directed-tail coverage is supplied by D. Any request first reads the finite tail window and its common-upper witness before a filter or net consumer can inspect the packet.

The handoff and consumer rows are downstream. The row B exposes the filter-base surface named in chapter 1601, while M routes the same tails to the Moore–Smith and net-convergence surface of chapter 3403. The row U then presents only metric or uniform completion consumers of that same route.

Finally W, R, Y, E provide the numerical window, readback, dyadic tolerance, and real seal rows. Transport, replay, provenance, and naming preserve the same finite carrier. Since none of the excluded principles has a carrier coordinate, the listed obligations exhaust the local surface. □

Theorem 2922.3 (DirectedFilter tail FilterBase handoff)**.** *Every* $\mathsf{FilterBase}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier must first read the* $\mathsf{DirectedSet}^{\uparrow}$ *tail-request row D and only then the handoff row B. The handoff presents finite tails, common-upper witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it does not present a quotient filter, ultrafilter, selected limit, countable-choice selector, or host filter equality.*

Proof. Begin with the NameCert obligations of Definition 3898.2. They place every filter-facing read after the displayed directed-tail row D.

Read B next. Because B is assembled only from the finite tails and upper-window witnesses in D, the filter-base consumer receives the same displayed family rather than an external filter object.

The structural rows H, C, P, N transport, replay, source, and name that handoff. No omitted quotient, ultrafilter, selection, or host-equality row can be recovered from the carrier. □

Theorem 2922.4 (DirectedFilter net-convergence route)**.** *Every* $\mathsf{MooreSmithNet}^{\uparrow}$ *or* $\mathsf{NetConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *packet factors through*

$$\mathsf{DirectedSet}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{MooreSmithNet}^{\uparrow}, \mathsf{NetConvergence}^{\uparrow} \longmapsto H, C, P, N.$$

The route shares the directed-tail ledger with chapter 3403 and the filter-base handoff with chapter 1601. It exports no global confluence of filters and nets, no selected subnet, no selected limit, and no ambient convergence theorem.

Proof. Project F to D, B, M, H, C, P, N. The row D is the only directed-index source, and B is the only filter-base handoff.

The net row M is reached after those two rows. It therefore reads the same directed tails as the filter-base consumer, matching the Moore–Smith carrier rather than a separately chosen subnet or a detached convergence assertion.

Transport and replay keep the route componentwise. Since the packet contains no selected subnet, selected limit, or ambient convergence coordinate, every net-convergence read is exhausted by the displayed rows. □

Theorem 2922.5 (DirectedFilter completion nonescape)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *packet reads the completion corridor through*

$$D, B, M, U \longmapsto W, R, Y, E \longmapsto H, C, P, N.$$

The route passes through the filter-limit basis corridor of chapter 3862 without adding quotient filters, selected limits, countable choice, host real equality, or ambient completeness.

Proof. Use Definition 3898.3 to force the directed-tail and filter-base prefix. Then use Definition 3898.4 to place the net-convergence row before any completion-facing consumer.

The completion row U is followed by the numerical rows W, R, Y, E. These rows expose finite stream windows, regular rational readback, dyadic tolerance accounting, and the terminal real seal in the order displayed by the carrier.

The structural rows H, C, P, N preserve and name the same route. Because the carrier has no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, the completion consumer cannot escape the stated corridor. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{DirectedFilter}^{\uparrow}$ packet is generated from a DirectedSet tail-request row, a FilterBase handoff row, MooreSmithNet and NetConvergence route rows, UniformCompletion and MetricCompletion consumer rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3898.1, Definition 3898.2, Definition 3898.3, Definition 3898.4, and Definition 3898.5 seed the finite directed-filter carrier for the L10 completion corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, ultrafilter arguments, selected limits, selected subnets, countable choice, host filter equality, ambient convergence, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail coverage, filter-base handoff, net-convergence routing, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DirectedFilterUp.

Chapter 2923

A Concrete Naming Certificate for $\mathsf{OscillationFunction}^{\uparrow}$

Following Baire's function hierarchy and Bishop–Bridges constructive analysis, the $\mathsf{OscillationFunction}^{\uparrow}$ packet records the finite oscillation certificate associated to a pointwise real-valued limit. It sits after the pointwise-limit route of chapter 2314, the Baire-one consumer surface of chapter 1604, the Dini-continuity consumer surface of chapter 2196, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed oscillation windows and value-readback rows; it does not import a host function space, quotient real equality, a selected supremum, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2923.1 (OscillationFunction carrier)**.** *An* $\mathsf{OscillationFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (P, B, D, S, R, V, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *source row whose displayed point-index and approximant-window route is being inspected.*
2. *B is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *consumer row receiving the oscillation certificate as Baire-one boundary data.*
3. *D is the* $\mathsf{DiniContinuity}^{\uparrow}$ *consumer row that may read the same finite oscillation windows before any uniform-modulus handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finitely many source neighbourhoods and approximant windows used by the oscillation request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected value differences and tolerance comparisons.*
6. *V is the terminal* $\mathsf{Real}^{\uparrow}$ *oscillation-value seal reached only after P, B, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the pointwise-limit, Baire-one, Dini-continuity, schedule, readback, value-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an oscillation request through the pointwise-limit windows, the Baire-one and Dini consumer rows, the stream schedule, the regular rational readback, and the RealUp seal.*
9. *K is the* Pkg *provenance over* $\mathsf{OscillationFunction}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationFunction}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host oscillation function, quotient real equality, a selected supremum over all neighbourhoods, an ambient Baire-one theorem, or completeness of $\mathsf{Real}^\uparrow$*.*

Lean4 ✓ `BEDC.Derived.OscillationFunctionUp.OscillationFunctionTasteGate single carrier alignment`

Theorem 2923.2 (OscillationFunction NameCert obligations)**.** *For every accepted* $\mathsf{OscillationFunction}^\uparrow$ *carrier $O = (P, B, D, S, R, V, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{OscillationFunction}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the pointwise-limit source, Baire-one consumer, Dini-continuity consumer, stream schedule, regular rational readback, RealUp oscillation seal, transport, replay, provenance, and local naming row;*
2. *pointwise-window exposure, requiring every oscillation request to read the* $\mathsf{PointwiseLimit}^\uparrow$ *route of chapter 2314 before any value-difference comparison is inspected;*
3. *consumer compatibility, requiring the* $\mathsf{BaireOneFunction}^\uparrow$ *and* $\mathsf{DiniContinuity}^\uparrow$ *rows to consume the same displayed windows and not a second host function-space object;*
4. *Real-seal exactness, requiring the oscillation value row V to be downstream of P, B, D, S, R and transported only by H and C;*
5. *nonescape from host oscillation functions, quotient real equality, selected suprema, ambient Baire-one representation theorems, unrestricted Dini compactness, and completeness of* $\mathsf{Real}^\uparrow$*.*

No NameCert obligation is stored outside $P, B, D, S, R, V, H, C, K, N$.

Proof. Project the carrier of Definition 3899.1. The accepted packet exposes exactly the pointwise-limit source row, the Baire-one and Dini consumer rows, the finite stream schedule, the regular rational readback, the RealUp value seal, transport, replay, provenance, and local naming.

The pointwise-window exposure follows from the source row P. The route cited in chapter 2314 already displays the point-index, approximant windows, regular rational readback, and RealUp value seal before a pointwise value can be consumed.

Consumer compatibility is then read through B and D. Both rows are coordinates of the same packet, so the Baire-one and Dini-facing consumers receive the finite schedule S and readback R already attached to P rather than an external function-space witness.

Finally the seal row V is downstream of this route. Componentwise transport by H and replay by C preserve only the displayed rows, while K and N carry provenance and local naming. Since the excluded objects are not coordinates of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2923.3 (OscillationFunction pointwise readback)**.** *Every* $\mathsf{Real}^\uparrow$ *oscillation-value read exported by an accepted* $\mathsf{OscillationFunction}^\uparrow$ *carrier factors through the displayed route*

$$\mathsf{PointwiseLimit}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as recorded by P, S, R, V. The readback may be consumed by the $\mathsf{BaireOneFunction}^\uparrow$ *and* $\mathsf{DiniContinuity}^\uparrow$ *rows B, D, but it cannot bypass the pointwise-limit route by reading a host oscillation function, quotient real equality, selected neighbourhood supremum, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Begin with a consumer request for V. By the carrier definition, V is reached only after the pointwise-limit source P, the finite schedule S, and the regular rational readback R have been displayed.

Apply the pointwise-limit Real-seal handoff from chapter 2314. It supplies the route from displayed source requests and approximant windows to a terminal RealUp value seal, and the oscillation packet reuses that route for finite value-difference comparisons.

The Baire-one and Dini rows are downstream consumers of the same readback. Transport through H and replay through C do not add a host function coordinate or a selected supremum row, so every oscillation-value read remains confined to P, S, R, V and the visible package rows. □

Theorem 2923.4 (OscillationFunction Baire-one handoff)**.** *Every* $\mathsf{BaireOneFunction}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OscillationFunction}^{\uparrow}$ *packet must read the pointwise-limit source P, the Baire-one row B, the Dini-continuity row D, the stream schedule S, the regular rational readback R, the RealUp oscillation seal V, and the structural rows H, C, K, N before any Baire-one oscillation boundary is public. Thus the handoff is a finite NameCert route over the rows displayed in Definition 3899.1, not a representation theorem about an ambient space of functions.*

Proof. Read the obligation surface of Definition 3899.2. It places the Baire-one consumer row behind the same pointwise-window exposure and Real-seal exactness rows used by the oscillation value.

Use the Baire-one packet of chapter 1604. Its public pointwise-limit ledger accepts displayed continuous approximant windows and regular rational readbacks, not a host function-space quotient. The oscillation carrier supplies those rows through P, S, R, V.

The Dini row D records the companion compact-modulus consumer boundary, but it is still downstream of the same displayed schedule. Since H, C, K, N transport, replay, source, and name only that finite route, the Baire-one handoff exports no ambient representation theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}$.

Constructive story: An $\mathsf{OscillationFunction}^{\uparrow}$ packet is generated from a PointwiseLimitUp source row, BaireOneFunctionUp and DiniContinuityUp consumer rows, StreamName schedules, RegSeqRat readbacks, a RealUp oscillation-value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3899.1, Definition 3899.2, Definition 3899.3, and Definition 3899.4 seed the finite oscillation certificate over pointwise-limit windows and Baire-one consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host oscillation functions, quotient real equality, selected suprema, ambient Baire-one representation, unrestricted Dini compactness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise-window exposure, consumer compatibility, Real-seal exactness, and nonescape for BEDC.Derived.OscillationFunctionUp.

Chapter 2924

A Concrete Naming Certificate for $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ packet records the finite universal-property surface for the L10 filter-completion route. It sits after the completion packet of chapter 1569, the convergence boundary of chapter 1558, the universal extension surface of chapter 397, and the uniform handoff of chapter 1473. The packet names the extension and uniqueness rows that downstream completion consumers may cite without importing quotient filters, selected limits, host equality, countable choice, or ambient completeness.

Definition 2924.1 (Filter completion universal-property carrier)**.** *A* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, L, E, Q, F, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion route, already tying the accepted read to a displayed Cauchy filter, finite windows, dyadic tolerances, regular readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *convergence row consumed only after the completion route C has exposed its source-filter and readback rows.*
3. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-and-uniqueness row attached to the same filter-completion route.*
4. *Q is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row read as provenance for uniform-facing consumers, not as a detached completion object.*
5. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row supplied by the same finite windows and dyadic tolerance ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached through C, L, F and used as the terminal filter-completion target.*
7. *A is the extension-uniqueness ledger, comparing accepted extension rows over the same C, L, E, Q, F, R surface by displayed* hsame *transport.*
8. *H records componentwise* hsame *transport for the completion route, convergence row, universal-property row, uniform handoff, regular readback, Real seal, extension ledger, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay of the route*

$$C, L, F, R \longmapsto E, Q, A, H, P, N.$$

10. *P is the* Pkg *provenance over* FilterCompletionUniversalProperty$^{\uparrow}$, CauchyFilterCompletion$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, UniformCompletion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{FilterCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, L, E, Q, F, R, A, H, K, P, N$. It has no coordinate for quotient filters, selected filter limits, host equality of filters or real names, countable-choice selectors, ambient completeness, or a completion object detached from the displayed filter route.

Lean4 ✓ `BEDC.Derived.FilterCompletionUniversalPropertyUp.FilterCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Theorem 2924.2 (Filter completion universal-property NameCert obligations)**.** *For every accepted* FilterCompletionUniversalProperty$^{\uparrow}$ *carrier $U = (C, L, E, Q, F, R, A, H, K, P, N)$, the local* NameCert$_{\text{FilterCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple; filter-completion exposure through C; convergence exposure through L after the source-filter route has been read; regular readback through F and the terminal* Real$^{\uparrow}$ *seal R; completion-universal extension through E; uniform provenance through Q; uniqueness comparison through A over the same rows; componentwise transport through H; displayed replay through K; provenance through P; local naming through N; and ledger refusal for quotient filters, selected limits, host equality, countable choice, ambient completeness, and detached completion objects.*

Proof. Carrier admission is projection from Definition 3900.1. The accepted packet exposes exactly the Cauchy-filter completion route, convergence row, universal-property row, uniform handoff, regular readback, Real seal, uniqueness ledger, transport, replay, provenance, and local name.

The filter route is finite. The completion row C is read before the convergence row L, the regular readback F, and the seal R, so no consumer can begin from a selected filter limit or a quotient-filter representative.

The universal-property route is also finite. Rows E, Q, A are read only after the same filter-completion and Real-seal data have been displayed, while H, K, P, N transport, replay, source, and name that route. Since no excluded host-side coordinate appears in the carrier, these rows exhaust the local certificate surface. □

Theorem 2924.3 (Filter completion universal-property extension uniqueness)**.** *Let $U = (C, L, E, Q, F, R, A, H, K,$ be an accepted* FilterCompletionUniversalProperty$^{\uparrow}$ *packet. Any two extension rows consumed through U are compared only by the displayed uniqueness ledger A over the common route*

$$C, L, F, R \longmapsto E, Q \longmapsto A, H, K, P, N.$$

Thus extension uniqueness is available to completion-facing consumers only after the same Cauchy-filter completion row, convergence row, regular readback, Real seal, universal-property row, and uniform handoff have been displayed. It cannot be replaced by host function equality, quotient-filter equality, a selected limit, countable choice, or ambient completeness.

Proof. Start with two extension reads from the accepted packet. The carrier has a single universal-property row E and a single uniqueness ledger A attached to the displayed filter-completion route.

Replay the continuation row K. Both extension reads must pass through C, L, F, R before E and Q are available, so the comparison has the same source-filter, readback, convergence, and Real-seal data on both sides.

The ledger A then compares the two rows by displayed hsame transport, and H, P, N preserve and name the same comparison. The refused host equalities, quotient objects, selected limits, choice selectors, and completeness principles have no coordinate inside U. □

Theorem 2924.4 (Filter completion universal-property filter handoff)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *packet reads the same finite handoff*

$$C, L, F, R, E, Q, A, H, K, P, N.$$

The handoff starts from the displayed Cauchy-filter completion row, reads the convergence boundary and regular readback before the terminal $\mathsf{Real}^{\uparrow}$ *seal, and then exposes extension, uniqueness, uniform provenance, transport, replay, provenance, and local naming. No consumer may replace that handoff by a quotient filter, selected filter limit, host equality, countable-choice selector, ambient completeness theorem, or detached completion carrier.*

Proof. Project the accepted carrier. Its only completion source is C, its only convergence row is L, its only regular readback and terminal seal rows are F and R, and its only universal-property and uniform-facing rows are E and Q.

Use the NameCert obligations of Definition 3900.2. They order the read from filter-completion exposure through convergence, readback, seal, extension, uniqueness, uniform provenance, transport, replay, package provenance, and local naming.

The sibling surfaces enforce the same boundary: chapter 1569 supplies the completion source, chapter 1558 supplies the convergence boundary, and chapter 1473 supplies only the uniform-facing provenance handoff. Since the carrier has no extra coordinate, the listed finite handoff is the whole exported route. □

Falsifiable prediction: Every accepted $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence, extension uniqueness, filter convergence, or uniform-completion handoff while omitting one of the displayed Cauchy-filter completion, Cauchy-filter limit, regular readback, Real seal, completion-universal, uniform provenance, transport, replay, package provenance, or local naming rows refutes the packet.

Independence witness: cauchyfiltercompletion, cauchyfilterlimit, completionuniversalproperty, uniformcompletion, regseqrat, real: none is bijective because $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ jointly carries filter-completion source, convergence boundary, regular readback, terminal Real seal, extension, uniqueness, uniform provenance, transport, replay, package provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3900.1, Definition 3900.2, Definition 3900.3, and Definition 3900.4 seed the finite L10 filter-completion universal-property route over Cauchy-filter completion, filter convergence, regular readback, Real seal, universal extension, uniform provenance, transport, replay, package provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected filter limits, host equality, countable choice, ambient completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convergence exposure, extension uniqueness, filter handoff, uniform provenance, and non-escape proofs for BEDC.Derived.FilterCompletionUniversalPropertyUp.

Constructive story: A FilterCompletionUniversalProperty$^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, a CauchyFilterLimit convergence row, a CompletionUniversalProperty row, a UniformCompletion provenance row, a RegSeqRat readback, a RealUp seal, an extension-uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2925

A Concrete Naming Certificate for $\mathsf{FiniteChoiceWindow}^{\uparrow}$

Following Bishop–Bridges finite choice discipline, the $\mathsf{FiniteChoiceWindow}^{\uparrow}$ packet records only those selections whose request set is already displayed as a finite unary-indexed spine. It sits between the finite observation gateway of chapter 126, the regular rational readback route of chapter 301, and the dyadic tolerance ledger of chapter 291; it supplies the L10 StreamName and RegSeqRat route with a finite section, not a countable-choice principle or a selected limiting object.

Definition 2925.1 (Finite choice window carrier)**.** *A* $\mathsf{FiniteChoiceWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, A, S, D, R, H, C, P, N)$$

with the following rows.

1. *I is a displayed unary index spine naming the finite request positions.*
2. *A is the finite candidate-row ledger, with one nonempty candidate list attached to each position of I.*
3. *S is the selected finite-section row, containing exactly one selected candidate for each displayed position of I.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row consumed at the selected finite section.*
6. *H, C, P, N are respectively the componentwise* hsame *transport row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$ *row.*

The classifier reads only $I, A, S, D, R, H, C, P, N$. It has no coordinate for countable choice, dependent choice, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteChoiceWindowUp.FiniteChoiceWindowTasteGate single carrier alignment`

Theorem 2925.2 (Finite choice window NameCert surface)**.** *For every accepted* $\mathsf{FiniteChoiceWindow}^{\uparrow}$ *carrier $W = (I, A, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier admission for the unary index spine I and the candidate-row ledger A;*
2. *finite-section exactness, saying that S selects one displayed candidate for each displayed position and no undisplayed position;*

3. *tolerance compatibility with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *and the* $\mathsf{StreamName}^{\uparrow}$ *finite window that consumes the section;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback routing through* R *after the selected finite section is fixed;*
5. *structural support through* H, C, P, N*, together with refusal of countable choice, dependent choice, selected limits, quotient stream equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

These obligations are precisely the finite window surface needed by the L10 route through $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*.*

Proof. Induct on the displayed unary index spine I. In the empty case, the candidate ledger and selected section are empty, and the carrier is admitted by the displayed transport, replay, provenance, and local naming rows.

For the successor step, assume the obligation surface has been built for the preceding finite spine. The next displayed position contributes one finite nonempty candidate list and one selected row. Adjoin that selected row to the finite section, transport it by H, replay the local route by C, and record its provenance in P and N.

The dyadic tolerance row D is stable under this finite adjoining step because the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger is read at the displayed request positions. The $\mathsf{StreamName}^{\uparrow}$ consumer sees the resulting finite section as a finite observation window, and the $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads R only after that window has been fixed.

No limiting step is used in the induction. Since every constructor step consumes one displayed list row and one displayed selected row, the proof never forms a countable-choice schedule, a dependent-choice chain, a selected limit, a quotient stream equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3901.1 and Definition 3901.2 seed the finite unary-indexed candidate carrier and its five NameCert obligation families for the L10 StreamName and RegSeqRat route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, dependent choice, selected limits, quotient stream equality, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-section exactness, dyadic tolerance compatibility, StreamName window routing, RegSeqRat readback routing, and support-row refusal for the same finite packet.

Constructive story: A $\mathsf{FiniteChoiceWindow}^{\uparrow}$ packet is generated from a displayed unary index spine, finite candidate rows, selected finite-section rows, a DyadicRatCore tolerance ledger, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2926

A Concrete Naming Certificate for $\mathsf{CauchySequenceField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchySequenceField}^{\uparrow}$ packet records the finite regular-Cauchy route by which field-facing real operations are read from rational names. It sits after the algebraic operation rows of $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, the order and apartness rows of **??**, the finite-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, and before the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a non-quotient field-facing route; it does not import host real arithmetic, quotient representatives, trichotomy, countable choice, or ambient completeness.

Definition 2926.1 (CauchySequenceField carrier)**.** *A* $\mathsf{CauchySequenceField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, W, D, A, O, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name whose field-facing operations are being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *synchronized-window row used to read the source sequence, companion operation inputs, and comparison requests at the same finite precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger for radii, slack, denominator guards, and tolerance comparisons opened by the synchronized windows.*
4. *A is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *operation row, displaying addition, negation, multiplication, zero, one, and operation replay over the same regular-Cauchy source rows.*
5. *O is the* **??** *order and apartness row, displaying positivity, nonzero guards, reciprocal-side admission, and comparison data read through W and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, A, O have been displayed and replayed.*
7. *H records componentwise* hsame *transport for the regular-rational source, synchronized windows, dyadic budgets, algebra rows, order rows, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a field-operation request through $R, W, D, A, O, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceField}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, **??**, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, W, D, A, O, E, H, C, P, N$. It has no coordinate for a quotient real, selected sequence representative, host real field operation, trichotomy branch, countable-choice schedule, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchySequenceFieldUp.CauchySequenceFieldTasteGate single carrier alignment`

Theorem 2926.2 (CauchySequenceField NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceField}^{\uparrow}$ *carrier $F = (R, W, D, A, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular-rational source row R, synchronized-window exposure through W, dyadic budget exactness through D, algebraic operation exposure through A, order and apartness exposure through O, real-seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No field-facing obligation may replace these rows by host real arithmetic or quotient equality.*

Proof. Project the carrier of Definition 3902.1. The visible rows are precisely the regular-rational source, synchronized windows, dyadic budget ledger, algebra row, order row, real seal, transport, replay, provenance, and local naming.

The algebra row A supplies the operation side only after the source row R has been read through W and D. The order row O supplies positivity, apartness, and reciprocal-side guards through the same finite windows and budgets.

The seal row E is terminal. It is reached only after the regular-Cauchy source, window, budget, algebra, and order rows have been displayed, so it cannot be used as a detached host real.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient representative, host operation, trichotomy, countable-choice, or completeness coordinate, those principles are not local NameCert obligations. □

Theorem 2926.3 (CauchySequenceField operation route)**.** *Every field-facing read exported by an accepted* $\mathsf{CauchySequenceField}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchySequenceAlgebra}^{\uparrow} \longmapsto ?? \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes algebraic operations and order-side guards as regular-Cauchy rows before any terminal real-name seal is inspected, and exports no host real arithmetic, quotient equality, selected representative, trichotomy branch, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a field-operation request for the accepted carrier $F = (R, W, D, A, O, E, H, C, P, N)$. Carrier projection exposes the $\mathsf{RegSeqRat}^{\uparrow}$ source row R and the synchronized $\mathsf{StreamName}^{\uparrow}$ windows W before any operation row can be consumed.

The dyadic ledger D then fixes the finite budgets used by the request. Algebraic operation reads pass through A, while positivity, apartness, nonzero, and reciprocal-side reads pass through O over the same windows and budgets.

Only after those rows have been displayed does the request reach the terminal $\mathsf{Real}^{\uparrow}$ seal E. The replay row C and transport row H preserve this order componentwise, and P, N source and name the same route.

Because the packet has no alternate coordinate for host real arithmetic, quotient equality, selected representatives, trichotomy, countable choice, or ambient completeness, every exported field-facing read is forced through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3902.1, Definition 3902.2, and Definition 3902.3 seed the finite regular-Cauchy field route from RegSeqRat, StreamName, DyadicRatCore, CauchySequenceAlgebra, and CauchySequenceOrder rows into the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host real arithmetic, quotient equality, selected sequence representatives, trichotomy, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic-budget exactness, algebra and order handoff, real-seal non-escape, transport, and replay proofs for BEDC.Derived.CauchySequenceFieldUp.

Constructive story: A $\mathsf{CauchySequenceField}^{\uparrow}$ packet is generated from a regular rational source row, synchronized StreamName windows, a DyadicRatCore tolerance ledger, CauchySequenceAlgebra operation rows, CauchySequenceOrder order and apartness rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2927

A Concrete Naming Certificate for BishopLocatedCompletionReflector$^{\uparrow}$

Following Bishop and Bridges on constructive completion and located real analysis, the BishopLocatedCompletionRef packet records the reflector row that binds the located-real completion route of chapter 3488 to the public completion-reflector surface of chapter 1585. It consumes the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the located-real row of chapter 1933, and the terminal real seal of chapter 38. The packet is a human-origin reflector for displayed located completion data; it is not quotient completion equality, a selected limit, countable choice, trichotomy, or ambient completeness.

Definition 2927.1 (Bishop located completion reflector carrier)**.** *A* BishopLocatedCompletionReflector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, R, D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BishopLocatedRealCompletion$^{\uparrow}$ *row exposing the located-real completion carrier through dyadic tolerances, finite stream windows, regular rational readback, locatedness, and a terminal real seal.*
2. *R is the* CauchyCompletionReflector$^{\uparrow}$ *row exposing unit, counit, idempotence, extension, reflection, and local real-name transport for the same completion-facing read.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used by both the located completion route and the reflector's local real-name transport.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected at those displayed tolerances.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row obtained from the same finite windows.*
6. *L is the* LocatedReal$^{\uparrow}$ *locatedness row attached to Q.*
7. *E is the* Real$^{\uparrow}$ *seal reached only after B, R, D, S, Q, L have been displayed.*
8. *H records componentwise* hsame *transport for the located completion row, reflector row, dyadic tolerance, stream-window, regular readback, locatedness, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion-facing request through $B, R, D, S, Q, L, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$, $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$ *row binding the displayed completion-reflector packet.*

The classifier on accepted packets compares only these rows under the displayed transport and replay data. It has no coordinate for quotient streams, selected representatives, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionReflectorUp.BishopLocatedCompletionReflectorTasteGate`
`single carrier alignment`

Theorem 2927.2 (Bishop located completion reflector NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ *carrier* $K = (B, R, D, S, Q, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$ *surface consists exactly of carrier admission, Bishop-located completion exposure, CauchyCompletionReflector exposure, DyadicRatCore tolerance exposure, StreamName finite-window exposure, RegSeqRat readback, LocatedReal locatedness exposure, Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every reflector-facing consumer reads only the displayed route*

$$B, R \longmapsto D \longmapsto S \longmapsto Q \longmapsto L \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3903.1. The only analytic rows are the located completion row, the public reflector row, the dyadic tolerance ledger, finite stream windows, regular rational readback, locatedness row, and Real seal.

Read the route in order. The row B supplies the located completion side and the row R supplies the reflector side. Their common finite route is then displayed through D, S, Q, L, E, so the reflector consumer cannot choose a second tolerance ledger, a host stream, or a selected real representative.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since quotient streams, selected limits, host equality, countable choice, trichotomy, compactness, and ambient completeness have no coordinate in the carrier, they cannot become local NameCert obligations. □

Theorem 2927.3 (Bishop located completion reflector route)**.** *Every* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ *consumer that treats displayed located Cauchy data as a reflector object must pass through the sibling route of chapter 3488 and the public reflector route of chapter 1585. The accepted packet exposes the reflector only as the finite carrier* $B, R, D, S, Q, L, E, H, C, P, N$ *and exports no quotient completion equality, selected host limit, countable-choice selector, trichotomy split, detached* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or host real equality test.*

Proof. Start with the NameCert obligations. They force any reflector-facing read to expose both the Bishop-located completion row B and the CauchyCompletionReflector row R before the terminal real seal is readable.

The shared analytic route is D, S, Q, L, E. The dyadic tolerance row fixes the finite request, the StreamName row opens the corresponding windows, the RegSeqRat row repacks those windows as regular rational data, the LocatedReal row supplies the lower–upper separation evidence, and the Real row seals the route.

The sibling chapters supply adjacent but not interchangeable packets. $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ gives the located completion side, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ gives the public reflector side. The present carrier binds both with the common dyadic, stream, regular-rational, located-real, and real-seal rows, so no omitted sibling or host-level object can replace it. □

Falsifiable prediction: A claimed $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ read that reaches the $\mathsf{Real}^{\uparrow}$ seal without first displaying both the $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ row and the $\mathsf{CauchyCompletionReflector}^{\uparrow}$ row over the shared dyadic, stream, regular-rational, and located-real route refutes the carrier surface.

Independence witness: bishop-located-real-completion, cauchycompletionreflector, dyadicratcore, streamname, regseqrat, locatedreal, real: no listed sibling is bijective to $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ because this packet binds the located completion row and public reflector row to one shared finite tolerance, window, readback, locatedness, real-seal, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ packet is generated from a BishopLocatedRealCompletion row, a CauchyCompletionReflector row, DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, LocatedReal locatedness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3903.1, Definition 3903.2, and Definition 3903.3 seed the finite reflector route binding located completion and public completion reflection over the shared dyadic, stream, regular-rational, located-real, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected limits, countable choice, trichotomy, host real equality, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-completion exposure, reflector exposure, tolerance-window exactness, regular readback, locatedness handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompletionReflectorUp.

Chapter 2928

A Concrete Naming Certificate for $\mathsf{CellularOrbitSynchronization}^{\uparrow}$

The $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ packet realizes the cellular carrier alternative of chapter 9391 as a finite paired-orbit synchronization surface. It consumes the observation-boundary route of chapter 2371, the classifier-boundary route of chapter 3810, the paired audit coupling of chapter 847, and the twin-substrate bridge of chapter 850. The packet records two local orbit replays together with the classifier tag row that authorizes them as one closed substrate read.

Definition 2928.1 (Cellular orbit synchronization carrier)**.** *A* $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O_0, O_1, G, B, Q, A, T, H, C, P, N)$$

with the following displayed rows:

1. *O_0 and O_1 are paired local orbit replay rows, each giving a finite transition table, initial row, time window, spatial window, and replay ledger.*
2. *G is the classifier tag row that assigns both replayed windows to the same substrate observation family.*
3. *B is the* $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ *boundary row recording the manifest and accepted observation windows used by the paired replay.*
4. *Q is the* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *row requiring the classifier tags to be displayed before either orbit read is exported.*
5. *A is the* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *row that records paired audit use without collapsing the two replay rows into a shared host orbit.*
6. *T is the* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *row through which the paired read may be inspected by a bridge-facing consumer.*
7. *H records componentwise* hsame *transport for the two orbit replays, classifier tag, observation boundary, classifier boundary, audit coupling, twin-substrate bridge, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the paired local orbits through G, B, Q, A, T, H, P, N.*

9. *P is the* Pkg *provenance over* CellularOrbitSynchronization$^{\uparrow}$, CellularSubstrateObservationBoundary$^{\uparrow}$, CellularSubstrateClassifierBoundary$^{\uparrow}$, TwinSubstrateAuditCoupling$^{\uparrow}$, TwinSubstrateBridge$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{CellularOrbitSynchronization}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unclassified orbit read, a detached host evaluator, an all-time induction principle, quotient stream equality, or an identification of the two substrate roles.

Lean4 ✓ `BEDC.Derived.CellularOrbitSynchronizationUp.CellularOrbitSynchronizationTasteGate`
`single carrier alignment`

Theorem 2928.2 (CellularOrbitSynchronization NameCert obligations)**.** *For every accepted* CellularOrbitSynchron *carrier $S = (O_0, O_1, G, B, Q, A, T, H, C, P, N)$, the local* NameCert$_{\text{CellularOrbitSynchronization}^{\uparrow}}$ *surface consists exactly of carrier admission for the paired orbit replays, classifier tag row, observation boundary, classifier boundary, audit coupling, twin-substrate bridge, componentwise transport, displayed continuation replay, package provenance, and local naming. Every synchronized substrate read factors through*

$$O_0, O_1 \longmapsto G \longmapsto B, Q \longmapsto A, T \longmapsto H, C, P, N.$$

No synchronized orbit obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3904.1. The accepted packet displays exactly the paired local orbit rows, classifier tag row, observation boundary, classifier boundary, audit coupling, bridge row, and structural rows.

The classifier row is read before the boundary rows. Thus a pair of replay ledgers O_0, O_1 becomes a synchronized substrate read only when G names the inspected pattern family and the rows B, Q bind that tag to the observation and classifier boundaries.

The audit and bridge rows preserve the paired reading without identifying the two replays. The row A supplies paired audit coupling, T supplies bridge-facing inspection, and H, C, P, N transport, replay, source, and name only that finite packet. Therefore unclassified orbit reads, host evaluators, all-time induction, quotient streams, and substrate-role identifications cannot become NameCert obligations. □

Theorem 2928.3 (CellularOrbitSynchronization classifier-locked handoff)**.** *Every synchronized orbit read accepted by a* CellularOrbitSynchronization$^{\uparrow}$ *packet displays the classifier tag row G before either paired orbit replay is exported through the observation boundary, classifier boundary, audit coupling, or twin-substrate bridge. A Lean follow-up for this carrier must supply* Nontrivial *and* FieldFaithful *instances in* `lean4/BEDC/Derived/CellularOrbitSynchronizationUp/TasteGate.lean`.

Proof. Project the accepted carrier. The paired orbit rows O_0, O_1 are present, but they are not exported directly; they are routed through the classifier tag row G.

Use the observation and classifier boundaries. The observation-boundary row B fixes the manifest and local windows, while the classifier-boundary row Q requires the pattern tag to be displayed for each observed window. Hence an orbit replay without G has no accepted route to a synchronized read.

The audit coupling and bridge rows are downstream. They consume the already tagged boundary packet, and the structural rows H, C, P, N preserve the same finite route. Thus the classifier-locked handoff is the only exported synchronization surface. □

Falsifiable prediction: Any synchronized orbit read that displays the paired local orbit replays but omits the classifier tag row G is rejected by the CellularOrbitSynchronization$^{\uparrow}$ carrier.

Independence witness: `cellular_substrate_observation_boundary`, `cellular_substrate_classifier_bo`
`twinsubstrateauditcoupling`, `twinsubstratebridge`: none is bijective to $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ because the synchronization packet uniquely binds paired local orbit replays, classifier-locked observation rows, paired audit coupling, bridge-facing inspection, transport, replay, provenance, and local naming in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ packet is generated from paired local orbit replay rows, a classifier tag row, CellularSubstrateObservationBoundary and CellularSubstrateClassifierBoundary rows, TwinSubstrateAuditCoupling and TwinSubstrateBridge rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3904.1, Definition 3904.2, and Definition 3904.3 seed the finite classifier-locked synchronization surface for paired local cellular orbit reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, all-time cellular induction, host evaluator correctness, quotient stream equality, or identification of the two substrate roles.

Upgrade path: Obligation closure requires checked carrier admission, paired orbit replay exactness, classifier-tag exposure, observation-boundary and classifier-boundary handoff, audit coupling, bridge-facing inspection, transport, replay, provenance, and local naming for BEDC.Derived.CellularOrbitSynchronizationUp.

Chapter 2929

A Concrete Naming Certificate for FiniteDimensionalHahnBanachSeparation$^{\uparrow}$

Following Bishop–Bridges constructive analysis and finite-dimensional separation theory, the FiniteDimensionalHahn packet records the finite certificate surface for a separating affine functional between a located point and a displayed convex set in a finite-dimensional normed vector space. It consumes the vector-space row of chapter 47, the linear-map row of chapter 48, the finite norm row used by chapter 1644, the real-name row of chapter 38, the located comparison row of chapter 1730, and the convex-set row of chapter 208. The packet is deliberately finite-dimensional: it names coefficient rows and located comparisons only for the displayed finite basis window, and it has no coordinate for full Hahn–Banach extension, choice-selected duals, or ambient weak-star compactness.

Definition 2929.1 (Finite-dimensional Hahn–Banach separation carrier)**.** *A* FiniteDimensionalHahnBanachSeparati *carrier is a finite* BHist *packet*

$$S = (V, B, L, N, R, Q, C, E, H, T, P, M)$$

with the following displayed rows:

1. *V is the finite* VecSpace$^{\uparrow}$ *source row, including the finite coordinate window on which every vector read is performed;*
2. *B is the finite basis and coefficient-admission ledger, recording that each inspected vector is read by finitely many scalar coordinates;*
3. *L is the* LinMap$^{\uparrow}$ *row for the separating functional, displayed by its finite coefficient list on B;*
4. *N is the* Norm$^{\uparrow}$ *row used only to bound the coefficient list and the finite vector reads;*
5. *R is the* Real$^{\uparrow}$ *endpoint ledger for scalar evaluations and thresholds;*
6. *Q is the* LocatedRealComparison$^{\uparrow}$ *row deciding the displayed scalar inequality with located slack;*
7. *C is the* ConvexSet$^{\uparrow}$ *row for the accepted convex-set membership and convex-combination reads;*
8. *E is the exact separation row, containing the point-side and set-side evaluation endpoints and their located comparison;*
9. *H records componentwise* hsame *transport for $V, B, L, N, R, Q, C, E, P, M$;*
10. *T records displayed* Cont *replay through the finite coordinate, functional, comparison, convex-set, and exactness rows;*

11. *P is the* Pkg *provenance over* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$, VecSpace$^{\uparrow}$, LinMap$^{\uparrow}$, Norm$^{\uparrow}$, Real$^{\uparrow}$, LocatedRealComparison$^{\uparrow}$, ConvexSet$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *M is the local* NameCert$_{\text{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed finite rows. It has no coordinate for an arbitrary extension of a functional, a selected quotient dual, a Zorn-style maximal sublinear extension, weak-star compactness, or an infinite family of separating hyperplanes.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalHahnBanachSeparationUp.FiniteDimensionalHahnBanachSeparationTaste single carrier alignment`

Theorem 2929.2 (Finite coefficient admission)**.** *For every accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier $S = (V, B, L, N, R, Q, C, E, H, T, P, M)$, every read of the separating functional factors through the finite coefficient ledger B, the* LinMap$^{\uparrow}$ *row L, the norm row N, and the scalar endpoint ledger R. A consumer may inspect only the finitely displayed coefficient list on the finite* VecSpace$^{\uparrow}$ *window; it may not replace that list by a host dual-space element or by a choice-selected functional.*

Proof. Project the accepted carrier. The vector data are read through V and the coefficient window B, while the functional row is L and the scalar endpoints are R.

The norm row N bounds only those displayed coefficient and vector reads. Transport and replay through H, T, P, M preserve the same finite list, so no additional coordinate can introduce a host dual element.

Thus coefficient admission is finite-row admission. The accepted packet exports the displayed list and its NameCert routes, not an unrestricted dual object or a choice-selected functional. □

Theorem 2929.3 (Separating inequality exactness)**.** *For every accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier $S = (V, B, L, N, R, Q, C, E, H, T, P, M)$, the separating inequality is exactly the row E read through the* Real$^{\uparrow}$ *endpoints in R, the* LocatedRealComparison$^{\uparrow}$ *comparison row Q, and the* ConvexSet$^{\uparrow}$ *membership row C. Consequently any downstream convex-analysis consumer receives a located scalar comparison between the point-side evaluation and the set-side bound, not an ambient separation theorem.*

Proof. Read the exactness row E. Its scalar endpoints are supplied by R, and the comparison between them is supplied by Q.

The convex side is constrained by C: membership and convex-combination reads of the displayed set pass through the finite convex-set row before the set-side bound can be cited. The functional values used by both sides still pass through B, L, N from Definition 3905.2.

Therefore the public inequality is the located comparison stored in E over the displayed finite rows. No consumer can cite a stronger ambient separating-hyperplane principle from this packet. □

Theorem 2929.4 (No escape to full Hahn–Banach choice)**.** *An accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier exports no full Hahn–Banach extension principle, no Zorn-style maximal extension, no selected quotient dual, no weak-star compactness object, and no infinite separating family. Every exported read factors through the finite vector-space window, coefficient ledger, linear-map row, norm row, real endpoint ledger, located comparison row, convex-set row, exactness row, componentwise transport, displayed replay, provenance, and local NameCert rows.*

Proof. Inspect the carrier coordinates. They are exactly $V, B, L, N, R, Q, C, E, H, T, P, M$, so every accepted read is a projection of one of those rows or a replay through the structural rows.

The rows B, L, N, R, Q, C, E are finite-dimensional and located: coefficients are listed on the finite basis window, scalar endpoints are displayed as $\mathsf{Real}^{\uparrow}$ rows, and the separation statement is a $\mathsf{LocatedRealComparison}^{\uparrow}$ row.

The excluded principles require coordinates not present in the carrier. Since there is no row for arbitrary extension, maximality, quotient dual selection, weak-star compactness, or an infinite family, those principles cannot be exported by the local NameCert surface. □

Falsifiable prediction: Dropping the coefficient ledger B makes Definition 3905.2 false: a consumer could no longer distinguish a displayed finite functional from an ambient host dual element.

Independence witness: vecspace, linearmap, norm, real, locatedrealcomparison, convexset: none is bijective with $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ because this carrier alone binds the finite vector-space window, coefficient ledger, linear functional, norm bound, scalar endpoints, located comparison, convex-set membership, and exact separation row into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ packet is generated from a finite VecSpaceUp coordinate window, a finite coefficient ledger, a LinearMapUp separating functional row, a NormUp bound row, RealUp scalar endpoints, a LocatedRealComparisonUp inequality row, a ConvexSetUp membership row, separating exactness data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3905.1, Definition 3905.2, Definition 3905.3, and Definition 3905.4 seed the finite-dimensional separation packet over VecSpaceUp, LinearMapUp, NormUp, RealUp, LocatedRealComparisonUp, ConvexSetUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Hahn–Banach extension, Zorn maximality, selected quotient duals, weak-star compactness, infinite separating families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coefficient transport, separating inequality exactness, convex-set readback, classifier stability, and non-escape for BEDC.Derived.FiniteDimensionalHahnBanachSeparationUp.

Chapter 2930

A Concrete Naming Certificate for PiecewiseLinearApproximation$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the PiecewiseLinearApproximation$^{\uparrow}$ packet records polygonal approximation over a closed bounded interval as finite NameCert rows. It starts from the ClosedBoundedInterval$^{\uparrow}$ source of chapter 1591, reads a UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$ mesh through chapter 2033 and chapter 1863, compares its cell ledger with the DyadicStepApproximation$^{\uparrow}$ route of chapter 3774, and hands terminal endpoint and value reads to ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ consumers through chapter 130, chapter 38, and chapter 291. The certificate names the finite affine-cell surface only; it does not introduce a host function space, selected uniform limit, quotient equality of functions, open-cover compactness, or ambient real completeness.

Definition 2930.1 (PiecewiseLinearApproximation carrier)**.** *A* PiecewiseLinearApproximation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, M, E, A, D, F, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the* ClosedBoundedInterval$^{\uparrow}$ *source row, displaying the two endpoint* Real$^{\uparrow}$ *names and the bounded interval window being approximated.*
2. *M is a* UniformPartition$^{\uparrow}$ *or* DyadicPartition$^{\uparrow}$ *mesh row, including the ordered cell boundary ledger, the positive mesh-width row, and the endpoint-coverage route.*
3. *E is the finite endpoint-value ledger, recording one* Real$^{\uparrow}$ *value row at every displayed mesh boundary.*
4. *A is the affine cell ledger: for each adjacent mesh cell it records the two endpoint-value rows, the affine interpolation route between them, and the cell-local evaluation window.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *slope and error ledger, recording dyadic slope brackets, cell-width witnesses, tolerance requests, and the displayed error budget used by downstream approximation consumers.*
6. *F is the* ContinuousMap$^{\uparrow}$ *handoff row, exposing the polygonal route only after I, M, E, A, D have been displayed.*
7. *T is the comparison row with* DyadicStepApproximation$^{\uparrow}$*, reading the same mesh cells and tolerance requests as a step-surface neighbour without identifying the affine ledger with a step ledger.*

8. *H records componentwise* hsame *transport for the interval source, mesh, endpoint values, affine cell rows, dyadic slope and error ledger, continuous-map handoff, step-neighbour row, provenance, and local naming.*

9. *C records displayed* Cont *replay of the route*

$$I, M, E, A, D \longmapsto F, T \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* PiecewiseLinearApproximation$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, UniformPartition$^{\uparrow}$, DyadicPartition$^{\uparrow}$, DyadicStepApproximation$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{PiecewiseLinearApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host function graph, quotient equality of maps, selected limiting function, compactness theorem, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PiecewiseLinearApproximationUp.PiecewiseLinearApproximationTasteGate`
`single carrier alignment`

Theorem 2930.2 (PiecewiseLinearApproximation NameCert obligations)**.** *For every accepted* PiecewiseLinearApproximation$^{\uparrow}$ *carrier*

$$L = (I, M, E, A, D, F, T, H, C, P, N),$$

the local NameCert$_{\text{PiecewiseLinearApproximation}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, mesh coverage through M, endpoint-value display through E, affine cell exactness through A, dyadic slope and error accounting through D, continuous-map handoff through F, step-surface comparison through T, componentwise transport through H, replay through C, provenance through P, and local naming through N. No obligation may read a host function space, selected uniform limit, quotient map equality, open-cover compactness, countable choice, or ambient real completeness.*

Proof. Carrier admission is projection from Definition 3906.1. The accepted packet displays the closed bounded interval, mesh, endpoint values, affine cell rows, dyadic slope and error ledger, and structural rows before any continuous-map or step-neighbour consumer sees the approximation.

The mesh rows are finite. The UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$ row supplies the ordered cells, endpoint coverage, and positive width witnesses, while the ClosedBoundedInterval$^{\uparrow}$ row fixes the source window. Thus every affine cell row is attached to a displayed source cell rather than to an ambient interval object.

The affine rows and dyadic ledger exhaust the approximation data. Each cell read factors through its two endpoint Real$^{\uparrow}$ values and the DyadicRatCore$^{\uparrow}$ slope and error ledger before the ContinuousMap$^{\uparrow}$ handoff row is available.

Finally H, C, P, N transport, replay, source, and name only this finite packet. Since no further carrier coordinate remains, the excluded host-function, quotient, limit, compactness, choice, and completeness objects cannot become local NameCert obligations. □

Theorem 2930.3 (PiecewiseLinearApproximation uniform handoff)**.** *Every* ContinuousMap$^{\uparrow}$, DyadicStepApproxim
Real$^{\uparrow}$, *or* DyadicRatCore$^{\uparrow}$ *consumer of an accepted* PiecewiseLinearApproximation$^{\uparrow}$ *packet receives only the finite route*

ClosedBoundedInterval$^{\uparrow}$, UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$, endpoint Real$^{\uparrow}$ rows, affine cells, DyadicRatCore$^{\uparrow}$ le

The consumer may inspect the polygonal approximation as a finite uniform or dyadic mesh surface, but it cannot bypass that route to obtain a host continuous function, selected uniform limit, quotient function equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The terminal consumers see F or T only after I, M, E, A, D have been displayed.

For the continuous-map handoff, the row F reads the same endpoint values, affine cell routes, and dyadic error ledger. It therefore exports a polygonal route over the visible mesh, not a host function graph.

For the step-surface neighbour, the row T shares the mesh and tolerance requests with $\mathsf{DyadicStepApproximation}^{\uparrow}$ while preserving the affine cell ledger as a separate displayed row. Hence the handoff can compare approximation ledgers without collapsing them into quotient equality of functions.

The remaining rows H, C, P, N only transport and replay the same finite data. No hidden limit, compactness, choice, completeness, or host-map coordinate is present for a consumer to read. □

Closure status for $\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{PiecewiseLinearApproximation}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source row, a UniformPartition or DyadicPartition mesh row, endpoint RealUp value rows, affine cell rows, a DyadicRatCore slope and error ledger, a ContinuousMap handoff row, a DyadicStepApproximation neighbour row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3906.1, Definition 3906.2, and Definition 3906.3 seed the finite polygonal approximation surface over closed bounded interval, mesh, endpoint-value, affine-cell, dyadic-ledger, and continuous-map handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected uniform limits, quotient function equality, open-cover compactness, countable choice, host function spaces, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, mesh coverage, endpoint-value display, affine cell exactness, dyadic slope and error accounting, continuous-map handoff, step-neighbour comparison, transport, replay, provenance, and local naming for BEDC.Derived.PiecewiseLinearApproximationUp.

Chapter 2931

A Concrete Naming Certificate for $\mathsf{CompactIntervalMaximum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalMaximum}^{\uparrow}$ packet records the finite witness surface for a continuous real-valued map on a closed bounded interval to expose a maximum candidate. It consumes the endpoint and containment route of chapter 1591, the finite interval-net surface of $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, the graph and continuity rows of chapter 130, and the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback route. The packet names only finite BEDC evidence for a maximum witness; it does not import host extrema, open-cover compactness, selected subcovers, Real completeness, quotient equality, Lean verification, or bridge checking.

Definition 2931.1 (Compact interval maximum carrier)**.** *A* $\mathsf{CompactIntervalMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, F, G, N, S, R, E, W, H, C, P, L)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *sealing.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the displayed interval source, including finite graph reads and pointwise continuity data.*
3. *G is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite-cell row, listing the interval cells, centers, radius ledgers, and coverage reads used by the witness search.*
4. *N is the finite value-comparison row selecting a displayed cell and center whose readback is no smaller than the other displayed net values at the admitted precision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for endpoint, graph, and value observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected value and comparison witnesses.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the selected maximum-witness value after I, F, G, N, S, R have been displayed.*
8. *W is the witness-locality row saying that the selected center is read through a cell of G whose containment is certified by I.*
9. *H records componentwise* hsame *transport for I, F, G, N, S, R, E, W and the local packet shape.*

10. *C records displayed* Cont *replay from interval source and graph reads through finite cells, value comparison, witness locality, readback, Real sealing, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalMaximum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$ *row.*

The carrier exports no host maximum operator, open-cover compactness theorem, selected subcover, ambient Real completeness principle, quotient equality of real names, or global choice row.

Theorem 2931.2 (Compact interval maximum finite witness obligations)**.** *For the carrier of Definition 3907.1, every accepted maximum-witness read factors through the closed bounded interval row I, the continuous-map graph row F, the finite-net cell row G, the finite value-comparison row N, the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback route S, R, E, the witness-locality row W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming L. These rows are the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.CompactIntervalMaximumUp.CompactIntervalMaximumTasteGate single carrier alignment`

Proof. Project the carrier. The interval source, graph, finite-cell, comparison, schedule, readback, seal, witness-locality, transport, replay, provenance, and local naming rows are the only coordinates of the accepted packet.

The closed bounded interval row I supplies endpoint order and containment ledgers before any cell is inspected. The finite-net row G supplies the displayed cells and centers, while F supplies the graph reads whose values are compared by N.

The readback route is downstream of those finite rows. The schedule S, regular rational readback R, and Real seal E expose the selected witness value only through the displayed observations, and W keeps the selected center inside a certified cell of the interval net.

Finally H, C, P, L transport, replay, package, and name the same finite surface. Since the packet has no coordinate for host extrema, open covers, selected subcovers, ambient completeness, quotient equality, or global choice, every accepted maximum-witness read remains on the displayed NameCert rows. □

Falsifiable prediction: A compact-interval maximum consumer that omits the finite cell row, graph row, or value-comparison row cannot claim a $\mathsf{CompactIntervalMaximum}^{\uparrow}$ witness.

Independence witness: closedboundedinterval, continuousmap, heineborel-finite-net, regseqrat, real: $\mathsf{CompactIntervalMaximum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval containment, graph reads, finite cells, value comparison, selected witness locality, readback, sealing, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{CompactIntervalMaximum}^{\uparrow}$ is expected at `BEDC.Derived.CompactIntervalMaximu` It should preserve the finite maximum-witness surface as carrier data before any downstream extreme-value consumer can inspect it.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalMaximum}^{\uparrow}$ packet is generated from a closed bounded interval row, a continuous-map graph row, a finite interval-net row, a finite value-comparison witness, StreamName schedules, RegSeqRat readbacks, a Real seal, witness locality, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3907.1 and Definition 3907.2 seed the finite maximum-witness carrier for continuous maps on closed bounded intervals.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalMaximum}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host extrema, open-cover compactness, selected subcovers, Real completeness, quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell coverage, graph readback, value comparison, witness locality, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalMaximumUp.

Chapter 2932

A Concrete Naming Certificate for $\mathsf{FilterCompletionMonad}^{\uparrow}$

Following Kelley and Bourbaki on filters and completion, the $\mathsf{FilterCompletionMonad}^{\uparrow}$ packet records the finite monad-shaped surface by which a displayed Cauchy-filter completion route admits a unit, a flattening read, and a terminal Real handoff without quotienting filters or selecting limits. It sits after the Cauchy-filter completion boundary of chapter 1569, the finite basis route of chapter 3862, the Cauchy-filter cluster route of chapter 1734, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet exposes the completion monad only as finite BHist and $\mathsf{NameCert}$ rows.

Definition 2932.1 (Cauchy completion filter monad carrier)**.** *A* $\mathsf{FilterCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, B, Z, S, R, D, E, U, F, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-boundary row whose accepted reads remain tied to displayed finite Cauchy-filter evidence.*
2. *B is the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *basis row used by the completion request before any Real-facing seal is inspected.*
3. *Z is the* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *cluster-handoff row, exposing the same finite support as a windowed cluster route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by the basis and cluster rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite windows and readback rows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only downstream from Q, B, Z, S, R, D.*
8. *U is the unit row, admitting a displayed source filter as a one-layer completion packet without adding a quotient filter or chosen limit.*
9. *F is the multiplication row, flattening a displayed two-layer completion packet by replaying its inner and outer Q, B, Z, S, R, D, E routes.*

10. *A is the associativity replay row, recording that the two displayed flattening orders read the same finite rows before E.*

11. *H records componentwise* hsame *transport for the completion, basis, cluster, window, readback, tolerance, seal, unit, flattening, associativity, replay, provenance, and local naming rows.*

12. *C records displayed* Cont *replay for unit reads, flattening reads, associativity reads, and Real handoff reads through $Q, B, Z, S, R, D, E, U, F, A, H, P, N$.*

13. *P is the* Pkg *provenance over* $\mathsf{FilterCompletionMonad}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, Z, S, R, D, E, U, F, A, H, C, P, N$. There is no coordinate for quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FilterCompletionMonadUp.FilterCompletionMonadTasteGate single carrier alignment`

Theorem 2932.2 (Cauchy completion filter monad laws)**.** *For every accepted* $\mathsf{FilterCompletionMonad}^{\uparrow}$ *carrier $M = (Q, B, Z, S, R, D, E, U, F, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *unit, multiplication, associativity, transport, replay, provenance, and local naming rows;*

2. *unit admission through U, requiring every source-filter read to enter the completion route through the same Q, B, Z, S, R, D, E rows;*

3. *multiplication admission through F, requiring every two-layer completion read to flatten by replaying the inner and outer displayed completion routes before the terminal seal;*

4. *associativity of displayed replay through A, requiring the two flattening orders to expose the same $Q, B, Z, S, R, D, E, U, F, H, C, P, N$ rows before any Real-facing consumer reads E;*

5. *Real handoff through E, requiring completion, basis, cluster, window, readback, tolerance, transport, replay, provenance, and local naming to remain visible to every* $\mathsf{Real}^{\uparrow}$ *consumer.*

No listed obligation may be discharged by quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, detached completion points, or ambient completeness.

Proof. Project the carrier of Definition 3908.1. The visible rows are exactly $Q, B, Z, S, R, D, E, U, F, A, H, C, P, N$, so carrier admission has no hidden quotient-filter, selected-limit, choice, host-equality, or ambient-completeness coordinate.

The unit row is read as a displayed admission into the existing completion boundary. It points the source filter through the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row Q, the $\mathsf{FilterLimitBasis}^{\uparrow}$ row B, the $\mathsf{CauchyFilterCluster}^{\uparrow}$ row Z, and the finite observation rows S, R, D before E is available.

The multiplication row flattens only displayed completion data. An accepted two-layer packet first replays the inner and outer Q, B, Z, S, R, D, E routes, and then records the flattened route through F, H, C, P, N.

Associativity is the equality of the displayed replay surface, not an ambient quotient principle. The row A states that the two orders of flattening expose the same finite rows before the terminal seal, so no selected representative or ultrafilter argument can enter between the two replays.

Finally apply the Real handoff. Since E is downstream from the completion, basis, cluster, window, readback, tolerance, unit, flattening, associativity, transport, replay, provenance, and local naming rows, every $\mathsf{Real}^{\uparrow}$ consumer receives the displayed monad route and no external completion principle. □

Falsifiable prediction: An accepted $\mathsf{FilterCompletionMonad}^{\uparrow}$ read is refuted if unit, multiplication, associativity, or Real handoff reaches a terminal seal without displaying the CauchyFilterCompletion, FilterLimitBasis, CauchyFilterCluster, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Independence witness: cauchyfiltercompletion, filterlimitbasis, cauchyfiltercluster, streamname, regseqrat, dyadicratcore, real: $\mathsf{FilterCompletionMonad}^{\uparrow}$ is not bijective to any listed sibling because it records unit, flattening, associativity replay, and Real handoff as one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterCompletionMonad}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, FilterLimitBasis row, CauchyFilterCluster row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, unit admission, multiplication flattening, associativity replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3908.1 and Definition 3908.2 seed the finite filter-completion monad route over completion, basis, cluster, window, readback, tolerance, Real seal, unit, flattening, associativity, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, detached completion points, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit admission, multiplication flattening, associativity replay, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.FilterCompletionMonadUp.

Chapter 2933

A Concrete Naming Certificate for MetaCICDecidableNeutralSpine$^{\uparrow}$

The MetaCICDecidableNeutralSpine$^{\uparrow}$ packet realizes the neutral-spine branch of the MetaCIC decidability audit map as finite BEDC data. It reads the bounded normal-equality checker of chapter 752, the decidability boundary of chapter 749, the witness surface of chapter 751, the typing frontier of chapter 3616, and the fuel trace of chapter 3889. Its role is to expose a neutral-spine comparison route without promoting bounded computation to global conversion decidability.

Definition 2933.1 (MetaCIC decidable neutral-spine carrier)**.** *A* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier is a finite packet*

$$S = (T, E, Q, B, R, H, C, P, N)$$

with the following rows:

1. *T is the typing-checker frontier row, carrying the closed context, neutral head, spine arguments, and result type read by* MetaCICDecidableTypingFrontier$^{\uparrow}$*.*
2. *E is the same-term row for structurally identical neutral heads and argument lists.*
3. *Q is the bounded normal comparison row supplied by* BoundedNormalEqualityChecker$^{\uparrow}$*.*
4. *B is the boundary row stating which finished-normal-form evidence would be needed to pass from bounded comparison to public conversion decidability.*
5. *R is the refusal row blocking any consumer that asks for unbounded beta-conversion, subject reduction, or strong-normalization evidence not displayed in S.*
6. *H is componentwise* hsame *transport for the typing, same-term, bounded-comparison, boundary, refusal, replay, provenance, and local-name rows.*
7. *C is the displayed* Cont *replay from a neutral-spine decidability request through T, E, Q, B, R, H, P, N.*
8. *P is* Pkg *provenance over the MetaCIC typing checker,* BoundedNormalEqualityChecker$^{\uparrow}$*,* MetaCICDecidableBoun MetaCICDecidabilityWitness$^{\uparrow}$*,* MetaCICDecidableTypingFrontier$^{\uparrow}$*,* MetaCICFuelTrace$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* NameCert$_{\text{MetaCICDecidableNeutralSpine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for global conversion decidability, unrestricted normalization, subject reduction, confluence, or an external semantic equality oracle.

Theorem 2933.2 (Neutral-spine NameCert obligations)**.** *Every accepted* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier exposes the finite obligation surface consisting of typing-frontier exactness, same-term structural readback, bounded normal comparison, decidability-boundary refusal, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet receives no additional global conversion theorem.*

Proof. Project the carrier of Definition 3909.1. The typing row T fixes the closed neutral-spine boundary before any comparison row is read, and the same-term row E records only structural agreement of the displayed neutral head and argument list.

The bounded comparison row Q is then interpreted through the boundary row B and refusal row R. These rows expose exactly the finite checker result and the missing endpoint evidence; they do not add strong normalization, confluence, subject reduction, or unbounded conversion.

Finally, H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Hence every accepted read of the packet is confined to the displayed NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.MetaCICDecidableNeutralSpineUp.MetaCICDecidableNeutralSpineTasteGate`
`single carrier alignment`

Theorem 2933.3 (Neutral-spine bounded-comparison boundary)**.** *For any accepted* MetaCICDecidableNeutralSpine *carrier, the passage from a bounded normal-equality answer to a public neutral-spine conversion answer is admissible only when the finished-normal-form boundary named by B is supplied inside the carrier route. Without that row, the packet exports a checker-local answer and a refusal, not global decidability.*

Proof. Read the bounded comparison row Q together with the boundary row B. The comparison row supplies a finite result about the inspected normal forms, while B records the exact endpoint evidence needed for public conversion.

If B is absent from the consumer route, the refusal row R is the only admissible export beyond the bounded checker answer. The transport and replay rows keep this refusal attached to the same typed neutral spine, so no external normalizer or semantic equality proof can enter through provenance. □

Theorem 2933.4 (Neutral-spine fuel monotonicity)**.** *Let $S = (T, E, Q, B, R, H, C, P, N)$ be an accepted* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier whose bounded normal comparison row Q records a checker-local answer at fuel k. If a displayed extension of the same fuel trace reaches fuel k' while preserving the typing frontier, same-term row, endpoint-boundary row, refusal row, componentwise transport, replay, provenance, and local naming rows T, E, B, R, H, C, P, N, then the extended comparison consumer remains inside the same checker-local answer/refusal boundary. It does not export public conversion decidability, strong normalization, confluence, subject reduction, or a semantic equality oracle.*

Proof. Project the carrier and the displayed fuel trace from the bounded comparison row Q. The extension to k' is admissible only because it preserves the already displayed typing frontier and same-term rows T, E together with the same endpoint-boundary row B.

The refusal row R remains attached to the extended trace. Thus increasing the inspected fuel may refine the checker-local comparison row, but it still leaves any request for public conversion outside the displayed endpoint evidence.

Finally, replay the extended row through C and transport the unchanged coordinates componentwise through H. The provenance and local naming rows P, N identify the result as the

same finite neutral-spine packet, so no global conversion theorem or off-packet normalizer is made available. □

Falsifiable prediction: An accepted $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ consumer is refuted if it exports neutral-spine conversion while omitting the displayed typing-frontier row, same-term row, bounded normal-equality row, boundary row, refusal row, transport, replay, provenance, or local NameCert row.

Independence witness: boundednormalequalitychecker, metacicdecidableboundary, metacicdecidabilitywitness, metacic-decidable-typing-frontier: no listed sibling is bijective to $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ because this packet jointly carries the typed neutral-spine frontier, structural same-term row, bounded normal comparison, endpoint-boundary refusal, transport, replay, provenance, and local naming in one carrier. The bounded checker lacks the typed frontier and refusal packet; the boundary packet lacks the same-term spine; the witness packet records typing and conversion evidence without owning the neutral-spine route; the typing frontier lacks the bounded normal comparison boundary.

Lean follow-up must provide Nontrivial and FieldFaithful TasteGate instances for the neutral-spine carrier before any bridge can upgrade the verification axis.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ packet is generated from typing-checker frontier data, same-term neutral-spine readback, bounded normal comparison, endpoint-boundary refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3909.1, Definition 3909.2, and Definition 3909.3 seed the neutral-spine decidability carrier, obligation surface, and bounded-comparison boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global conversion decidability, strong normalization, subject reduction, confluence, semantic equality, Lean verification, or bridge checking for $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, typing-frontier exactness, same-term readback, bounded normal-comparison soundness, endpoint-boundary refusal, replay stability, provenance non-escape, and TasteGate instances for BEDC.Derived.MetaCICDecidableNeutralSpineUp.

Chapter 2934

A Concrete Naming Certificate for $\mathsf{YoungIntegral}^{\uparrow}$

Following Young and Lyons, the $\mathsf{YoungIntegral}^{\uparrow}$ packet records controlled Riemann-sum integration from finite p-variation data. It sits beside the bounded-variation carrier of chapter 398, the regulated-function surface of chapter 1906, the Riemann-sum packet of chapter 1615, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The chapter names only the finite controlled-sum route; it does not import measure completion, quotient equality of functions, selected refinements, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2934.1 (Young integral carrier)**.** *A* $\mathsf{YoungIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, V, C, S, D, R, E, H, K, P, N)$$

with the following displayed rows:

1. *F is the regulated integrand row, read through* $\mathsf{RegulatedFunction}^{\uparrow}$ *and its finite left/right window data.*
2. *G is the regulated integrator row, carrying the same interval ledger and endpoint schedule as F.*
3. *V is the p-variation and control ledger, recording bounded-variation comparison rows, variation exponents, control steps, and the finite superadditivity checks used by the route.*
4. *C is the controlled-window row, matching integrand oscillation windows with integrator variation windows before any sum is formed.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row listing the tagged partitions, dyadic mesh requests, controlled step values, and finite sum values.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for mesh radii, variation bounds, and sum-difference margins.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Cauchy trace of controlled sums.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *integral seal reached only after F, G, V, C, S, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the integrand, integrator, variation ledger, controlled windows, Riemann-sum row, dyadic ledger, readback, seal, provenance, and local name.*

10. *K records displayed* Cont *replay from a Young-integral request through controlled windows, finite sums, dyadic tolerances, regular readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a measure-completion object, quotient equality of functions, an unrecorded refinement selector, a host rough-path lift, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.YoungIntegralUp.YoungIntegralTasteGate single carrier alignment`

Theorem 2934.2 (Young integral NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegral}^{\uparrow}$ *carrier* $Y = (F, G, V, C, S, D, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$ *surface is exhausted by integrand admission, integrator admission, p-variation control, controlled-window matching, Riemann-sum formation, dyadic tolerance exactness, regular rational readback, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming. No accepted consumer may replace those rows by measure completion, quotient function equality, a selected refinement sequence, a host rough-path lift, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 3910.1. The rows F and G expose the regulated integrand and integrator data, while V records the finite variation and control ledger used by the integration route.

The controlled windows are then finite. The row C matches integrand oscillation windows to integrator variation windows, and S forms only the tagged sums displayed by the $\mathsf{RiemannSum}^{\uparrow}$ packet.

The convergence handoff is read through D, R, E. Dyadic tolerances in D bound the mesh and sum-difference rows, R records the regular rational trace, and E seals the resulting Real value only after the previous rows have been consumed.

Finally H, K, P, N transport, replay, source, and name that same route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 2934.3 (Young integral Riemann-sum handoff)**.** *Every accepted* $\mathsf{YoungIntegral}^{\uparrow}$ *integral read factors through*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as recorded by F, G, V, C, S, D, R, E. The handoff exports a finite controlled Riemann-sum route and no measure-completion integral, quotient function equality, unrecorded refinement selector, host rough-path lift, countable-choice principle, or ambient Real-completeness theorem.

Proof. Start with the integrand and integrator rows. The regulated-function siblings expose the finite endpoint and window data for F and G, and the bounded-variation sibling supplies the visible variation ledger read by V.

Next form the controlled Riemann sums. The window row C records which integrand and integrator windows are paired, and the $\mathsf{RiemannSum}^{\uparrow}$ row S records the tagged finite sums over those windows.

The tolerance and readback rows then force the terminal route. The dyadic ledger D supplies mesh and sum-difference margins, R records the regular rational trace, and E is the only Real seal available to the consumer.

Transport and replay through H, K, P, N preserve this finite route. Therefore the public integral read cannot escape to measure completion, quotient function equality, a hidden refinement selector, rough-path enhancement, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegral}^{\uparrow}$ packet is refuted if a consumer reaches the Real integral seal while omitting the controlled-window row C, the Riemann-sum row S, the dyadic tolerance row D, or the regular readback row R.

Independence witness: BoundedVariation, RegulatedFunction, RiemannSum, DyadicRatCore, RegSeqRat, Real: none is bijective with $\mathsf{YoungIntegral}^{\uparrow}$ because the packet jointly carries integrand and integrator rows, a variation-control ledger, controlled windows, finite sums, dyadic tolerances, regular readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegral}^{\uparrow}$ packet is generated from regulated integrand and integrator rows, a p-variation control ledger, controlled Riemann-sum windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3910.1, Definition 3910.2, and Definition 3910.3 seed the finite controlled-sum route for Young integration.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected refinement sequences, host rough-path lifts, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked integrand admission, integrator admission, variation-control exactness, controlled-window matching, Riemann-sum handoff, dyadic tolerance readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegralUp.

Chapter 2935

A Concrete Naming Certificate for $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$

The $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet realizes the type-checking-as-classifier-membership vision of chapter 9367 as a finite NameCert carrier. It reads the classifier-membership packet of chapter 476, the typing classifier of chapter 489, the subject-reduction surface of chapter 483, the derivation trace of chapter 598, and the proof-erasure route of chapter 642. The packet records membership, derivation, preservation, and erasure rows together; it does not export a host type checker, semantic conversion oracle, quotient equality of terms, hidden proof object, or ambient normalization theorem.

Definition 2935.1 (TypeCheckingClassifierMembership carrier)**.** *A* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, J, E, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the term-history row whose membership is being checked.*
2. *J is the finite typing-judgment ledger, read through* $\mathsf{ClassifierTyping}^{\uparrow}$*.*
3. *E is the* Ext *classifier-membership row, read through* $\mathsf{TypeClassifierMembership}^{\uparrow}$*.*
4. *D is the derivation trace row, read through* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$*.*
5. *S is the subject-reduction row, recording the displayed preservation handoff into* $\mathsf{SubjectReduction}^{\uparrow}$*.*
6. *Q is the proof-erasure row, read through* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *and separating public erased terms from proof evidence.*
7. *H records componentwise* hsame *transport for the term, judgment, membership, derivation, preservation, erasure, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a type-checking request through classifier membership, derivation trace, subject reduction, and erasure readback.*
9. *P is the* Pkg *provenance over* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$*,* $\mathsf{TypeClassifierMembership}^{\uparrow}$*,* $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$*,* $\mathsf{ProofTermErasureTrace}^{\uparrow}$*,* BHist*,* Ext*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host type checker, hidden conversion relation, semantic equality of terms, quotient term representative, proof object outside the erasure row, or global normalization theorem.

Lean4 (definition) `BEDC.Derived.TypeCheckingClassifierMembershipUp.taste gate`

Theorem 2935.2 (TypeCheckingClassifierMembership NameCert obligations)**.** *For every accepted* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *carrier* $M = (T, J, E, D, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingC}}$ *surface is exhausted by term-row admission, typing-ledger admission,* Ext *membership exposure, derivation-trace exposure, subject-reduction handoff, proof-erasure separation, componentwise transport, displayed replay, provenance, and local naming. No accepted consumer may replace those rows by host type checking, hidden conversion, semantic equality of terms, quotient representatives, proof objects outside* Q*, or ambient normalization.*

Proof. Project the carrier of Definition 3911.1. The rows T, J, E expose the term, finite typing ledger, and Ext membership surface used by the classifier-membership read.

The derivation and preservation rows are also displayed. The row D records the finite type-checking derivation trace, and S records the subject-reduction handoff that may be consumed only through the visible preservation boundary.

The erasure row Q separates proof evidence from public erased terms. It supplies the proof-term erasure trace without adding a proof object outside the packet.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the excluded host checker, conversion oracle, semantic equality, quotient representative, hidden proof object, and normalization theorem have no coordinate in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 2935.3 (TypeCheckingClassifierMembership subject-reduction handoff)**.** *Every subject-reduction read exported by an accepted* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{TypeClassifierMembership}^{\uparrow} \longmapsto \mathsf{ClassifierTyping}^{\uparrow} \longmapsto \mathsf{TypeCheckingDerivationTrace}^{\uparrow} \longmapsto \mathsf{SubjectReduction}^{\uparrow} \longmapsto \mathsf{Proc}$$

as recorded by E, J, D, S, Q*. The handoff preserves classifier membership and public erasure through* H, C, P, N*, and it exports no host typing judgment, hidden conversion relation, semantic equality of terms, quotient term representative, proof object outside* Q*, or ambient normalization theorem.*

Proof. Begin with the membership row E and the typing ledger J. The type-classifier membership and classifier-typing siblings already route type membership through finite Ext, Cont, SigRel, and NameCert data.

Next read the derivation trace D. It records the rule ledger and continuation route used by the type-checking run, so the preservation request reaches $\mathsf{SubjectReduction}^{\uparrow}$ only through the displayed trace.

The subject-reduction row S is therefore a handoff, not a hidden theorem. The proof-erasure row Q records which public erased data may be read after preservation while keeping proof evidence inside the trace boundary.

Transport and replay through H, C, P, N preserve this route. Therefore a consumer receives exactly classifier membership, derivation trace, preservation handoff, and erasure separation, with no host typing judgment or normalization oracle. □

Falsifiable prediction: An accepted $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet is refuted if a subject-reduction consumer reaches S without the membership row E, derivation trace D, proof-erasure row Q, and structural route H, C, P, N.

Independence witness: TypeClassifierMembership, ClassifierTyping, SubjectReduction, TypeCheckingDerivationTrace, ProofTermErasureTrace: none is bijective with $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ because the packet jointly carries term rows, typing ledgers, Ext membership, derivation traces, subject-reduction handoff, proof-erasure separation, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet is generated from a term-history row, finite typing ledger, Ext membership row, derivation trace, subject-reduction handoff, proof-erasure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3911.1, Definition 3911.2, and Definition 3911.3 seed the finite type-checking classifier-membership packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host type checking, hidden conversion, semantic equality of terms, quotient term representatives, proof objects outside erasure rows, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked term-row admission, typing-ledger admission, Ext membership exposure, derivation-trace exposure, subject-reduction handoff, proof-erasure separation, and structural transport for BEDC.Derived.TypeCheckingClassifierMembershipUp.

Chapter 2936

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$

Following Bishop and Bridges constructive completion and Bourbaki's universal-mapping language, the $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$ packet records one finite BEDC certificate comparing the reflector, comparison, adjunction, and monad presentations of Cauchy completion. It reads the reflector route of chapter 1585, the comparison surface of chapter 1954, the adjunction handoff of chapter 1456, and the monad schedule of chapter 378 through the same finite real-seal route. The packet names the common certificate surface only; it does not construct quotient completions, choose representatives, identify host objects, or appeal to ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 2936.1 (CauchyCompletionReflectorComparison carrier)**.** *A* $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\rho, K_0, A, M, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. ρ *is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row exposing unit, counit, idempotent, extension, reflection, and local real-transport data.*
2. K_0 *is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row exposing the comparison between unit-and-universal-property and functorial metric-completion presentations.*
3. A *is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row exposing the finite unit-counit handoff surface.*
4. M *is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row exposing the finite schedule, diagonal readback, and monad composition boundary.*
5. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window route inspected by all four completion presentations.*
6. Q *is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route reached after the displayed windows.*
7. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window route.*
8. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* W, D, Q *have been displayed.*
9. H *records componentwise* hsame *transport for the four presentation rows, the shared real-seal rows, replay, provenance, and naming.*
10. C *records displayed* Cont *replay of the route*

$$\rho, K_0, A, M \longmapsto W, D, Q \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* CauchyCompletionReflectorComparison$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, CauchyCompletionComparison$^{\uparrow}$, CauchyCompletionAdjunction$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows and their displayed transports. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, open-cover compactness, or completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorComparisonUp.CauchyCompletionReflectorComparisonTasteGate`
`single carrier alignment`

Theorem 2936.2 (CauchyCompletionReflectorComparison NameCert obligations)**.** *For every accepted* CauchyCompletionReflectorComparison$^{\uparrow}$ *carrier*

$$K = (\rho, K_0, A, M, W, Q, D, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$ *surface consists exactly of reflector-route admission through ρ; comparison-route exposure through K_0; adjunction handoff through A; monad schedule compatibility through M; shared finite real-seal routing through W, D, Q, E; componentwise transport through H; replay through C; provenance through P; and local naming through N. It exports no quotient completion, selected representative, countable choice, host equality, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is tuple projection from Definition 3912.1. The accepted packet displays exactly the four presentation rows, the shared window-tolerance-readback-seal route, and the structural rows.

The reflector, comparison, adjunction, and monad rows are then read as sibling routes. The row ρ contributes the reflector route, K_0 contributes the comparison route, A contributes the adjunction handoff, and M contributes the schedule and diagonal-readback compatibility. None of these rows contains an independent terminal seal.

The common real-facing route is exhausted by W, D, Q, E. A consumer must inspect the finite StreamName$^{\uparrow}$ windows, the DyadicRatCore$^{\uparrow}$ tolerance ledger, and the RegSeqRat$^{\uparrow}$ readback before reaching the terminal Real$^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name only the displayed packet. Since no further coordinate remains in the carrier, quotient completions, selected representatives, countable choice, host equality, open-cover compactness, and ambient Real$^{\uparrow}$ completeness cannot be exported. □

Closure status for NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$.

Constructive story: A CauchyCompletionReflectorComparison$^{\uparrow}$ packet is generated from reflector, comparison, adjunction, and monad presentation rows, shared StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3912.1 and Definition 3912.2 seed the finite certificate comparing reflector, comparison, adjunction, and monad completion presentations over one shared real-seal route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, open-cover compactness, Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector-route admission, comparison-route exposure, adjunction handoff, monad schedule compatibility, shared real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionReflectorComparisonUp.

Chapter 2937

A Concrete Naming Certificate for CompactRegularCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges on compact metric sequential arguments and regular Cauchy real names, the CompactRegularCauchySubsequence$^{\uparrow}$ packet records the finite route by which compact metric inspection supplies a regular Cauchy subsequence window before any Real or Completion consumer reads the result. It sits over the compact metric source of chapter 129, the mature regular-subsequence route of chapter 381, the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive compact-subsequence certificate for L10 completion consumers; it is not open-cover compactness, countable choice, quotient stream equality, host sequence equality, or ambient Real$^{\uparrow}$ completeness.

Definition 2937.1 (Compact regular Cauchy subsequence carrier)**.** *A* CompactRegularCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row carrying finite-net, located-distance, and compact-bundle inspection data.*
2. *S is the source* StreamName$^{\uparrow}$ *row for the sequence windows inspected by the compact metric consumer.*
3. *T is the monotone-cofinal unary reindexing ledger consumed by the* RegularCauchySubsequence$^{\uparrow}$ *surface.*
4. *W is the selected finite subsequence-window row, pairing each retained compact window with the inspected source window that produced it.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to compact mesh radii, selected-window distances, and regular Cauchy tail budgets.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the selected windows and their transported regularity evidence.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row, exposed only after the compact source, subsequence ledger, finite windows, dyadic tolerances, and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for K, S, T, W, D, R, E.*
9. *C records displayed* Cont *replay from compact metric inspection through finite windows, subsequence selection, dyadic tolerances, regular readback, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, T, W, D, R, E, H, C, P, N$. It has no coordinate for an externally selected infinite subsequence, open-cover compactness theorem, quotient stream representative, host sequence equality, host real equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactRegularCauchySubsequenceUp.CompactRegularCauchySubsequenceTasteGate`
`single carrier alignment`

Theorem 2937.2 (Compact regular Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ *carrier*

$$Q = (K, S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through K, source-window exposure through S, monotone-cofinal subsequence exposure through T, selected-window exposure through W, dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3913.1. The finite packet exposes exactly the compact metric source, source stream windows, reindexing ledger, selected windows, dyadic ledger, regular rational readback, Real seal, and the structural provenance rows.

Read the compact and subsequence rows first. The row K supplies only displayed compact metric inspection data, while S, T, W record the source windows, monotone-cofinal route, and selected finite windows consumed by the regular-subsequence surface.

The analytic handoff is then forced through D, R, E. The dyadic ledger records the mesh and tail budgets, the regular rational row reads the selected windows as regular data, and the Real seal row is downstream of those rows.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 2937.3 (Compact regular Cauchy subsequence route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The compact source row supplies finite inspection windows, the regular-subsequence row supplies the monotone-cofinal selected-window ledger, and the dyadic and regular rational rows supply the Cauchy budgets before the terminal Real seal is exposed.

Proof. Begin with an accepted carrier $Q = (K, S, T, W, D, R, E, H, C, P, N)$. The compact source row K is the only compact metric coordinate, and the source windows are read through S.

Apply the regular-subsequence public export of Definition 381.9. The selected windows are named by the monotone-cofinal ledger T and the finite window row W, so the consumer cannot replace them by an unrecorded subsequence selector.

Next read the dyadic and regular rational rows. The ledger D carries the compact mesh and tail budgets, and R repacks the selected windows as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The Real seal E is therefore reached only after the compact source, finite windows, subsequence ledger, dyadic tolerance rows, and regular readback have been displayed. Transport and replay through H, C, P, N preserve that same route, so open-cover compactness, countable choice, quotient streams, host equality, and ambient completeness remain outside the exported packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ packet is generated from a CompactMetric source row, StreamName finite windows, a RegularCauchySubsequence monotone-cofinal ledger, selected finite windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3913.1, Definition 3913.2, and Definition 3913.3 seed the compact-source regular Cauchy subsequence route over CompactMetric, StreamName, RegularCauchySubsequence, DyadicRatCore, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient stream equality, host sequence equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, monotone-cofinal selected-window routing, dyadic tolerance accounting, RegSeqRat readback exactness, RealUp seal nonescape, and selector-refusal proofs for BEDC.Derived.CompactRegularCauchySubsequenceUp.

Chapter 2938

A Concrete Naming Certificate for $\mathsf{LindelofSpace}^{\uparrow}$

Following Lindelof, Bourbaki, and Kelley on second-countable topology, the $\mathsf{LindelofSpace}^{\uparrow}$ packet records the countable subcover route that is visible from a displayed countable-base presentation. It sits over the countable-base carrier of chapter 1901, the local countable-neighbourhood route of chapter 2053, the generated-open classifier of chapter 88, and the finite inspection windows of chapter 126. The packet names only displayed cover enumeration windows, extracted subcover ledgers, base-refinement routes, transport, replay, provenance, and local naming; it does not select arbitrary subcovers by choice, assert compactness, or add a host open-family oracle.

Definition 2938.1 (Lindelof space carrier)**.** *A* $\mathsf{LindelofSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, T, B, U, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source topology carrier row whose finite histories supply the only points and opens inspected by the packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open classifier row used by every open-cover and subcover read.*
3. *B is the* $\mathsf{SecondCountable}^{\uparrow}$ *base-enumeration row, recording the finite basic-open windows through which cover membership is tested.*
4. *U is the countable cover enumeration row, displaying the open-cover entries visible to the packet through T and the finite windows of B.*
5. *S is the extracted countable subcover ledger, recording the selected cover indices and the basic cells whose refinements witness coverage.*
6. *R is the base-refinement route from displayed basic cells in B into cover entries in U, with every refinement read tied to the same topology row T.*
7. *H stores componentwise* hsame *transport for the source topology, topology classifier, base enumeration, cover enumeration, extracted subcover ledger, refinement route, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a point or basic-cell read through the topology classifier, countable-base window, cover enumeration, extracted subcover ledger, refinement route, transport, provenance, and local name.*

9. *P is the* Pkg *provenance row over* $\mathsf{LindelofSpace}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{FirstCountableSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}$ *row.*

The carrier exports a countable subcover only as the displayed ledger S over the displayed cover enumeration U. It has no coordinate for arbitrary open-cover compactness, host choice of subcovers, quotient equality of opens, or a fuel-erased enumeration of all covers.

Lean4 ✓ `BEDC.Derived.LindelofSpaceUp.LindelofSpaceTasteGate single carrier alignment`

Theorem 2938.2 (Lindelof countable-cover route)**.** *For an accepted* $\mathsf{LindelofSpace}^{\uparrow}$ *carrier*

$$Q = (X, T, B, U, S, R, H, C, P, N),$$

every countable-subcover read factors through the finite route

$$X, T, B, U, S, R, H, C, P, N.$$

The reader first opens the source topology and generated-open classifier, then inspects the displayed second-countable base window, then reads the cover enumeration, extracted subcover ledger, base-refinement route, componentwise transport, replay, provenance, and local naming rows. No consumer receives a host family of all open covers, an unrestricted subcover selector, an open-cover compactness theorem, or equality of opens outside the displayed classifier.

Proof. Project the accepted carrier through Definition 3914.1. The rows $X, T, B, U, S, R, H, C, P, N$ are the complete surface available to a consumer.

The cover read starts in X and T. A displayed open-cover entry is not read as a host open set; it is read through the generated-open classifier and the finite base window B.

The extracted subcover is then read through S and R. The subcover ledger stores the visible cover indices, while the refinement row records how displayed basic cells refine into those entries.

Finally H, C, P, N transport, replay, source, and name the same route. Since no other coordinate is present, the packet cannot export arbitrary open-cover compactness, a choice-selected subcover, quotient equality of opens, or a host oracle for all covers. □

Theorem 2938.3 (Lindelof second-countable handoff)**.** *Every accepted* $\mathsf{SecondCountable}^{\uparrow}$ *packet with a displayed countable open-cover enumeration induces a* $\mathsf{LindelofSpace}^{\uparrow}$ *read by replaying the cover through finite base windows and recording the extracted countable subcover ledger. The induced read keeps the same source topology, generated-open classifier, countable-base enumeration, stream-window discipline, transport, replay, provenance, and local naming rows; only the cover enumeration U, extracted ledger S, and refinement route R are added as Lindelof rows.*

Proof. Start with the accepted $\mathsf{SecondCountable}^{\uparrow}$ packet. Its source row, topology classifier, displayed base enumeration, stream-window discipline, transport, replay, provenance, and local name already have the BEDC roles required by Definition 3914.1.

Read the displayed countable cover through the same finite base windows. For each accepted basic cell, the refinement route records the cover entry that contains it, and the extracted ledger records the visible cover index.

Replay the finite base-window enumeration to form the countable subcover ledger. This is a ledger read over B, U, S, R, not a choice principle over a host family of opens.

The remaining rows are unchanged. The handoff therefore adds only the Lindelof cover, ledger, and refinement rows, while preserving the second-countable topology source and excluding compactness, arbitrary subcover selection, and host open equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3914.1, Definition 3914.2, and Definition 3914.3 seed the finite countable-subcover interface over displayed topology rows, second-countable base windows, cover enumerations, extracted subcover ledgers, refinement routes, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary subcover choice, quotient equality of opens, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable-cover routing, second-countable handoff, base-refinement replay, transport, provenance, and local naming for BEDC.Derived.LindelofSpaceUp.

Constructive story: A $\mathsf{LindelofSpace}^{\uparrow}$ packet is generated from one source topology row, one Topology classifier row, one SecondCountable base-enumeration row, one countable cover enumeration row, one extracted countable subcover ledger, one base-refinement route, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2939

A Concrete Naming Certificate for $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ packet records a finite dyadic refinement-tree certificate for closed bounded interval cover work. It sits between the dyadic cover rows of chapter 1428, the finite comparison surface of chapter 3430, and the endpoint-and-containment packet of chapter 1591. The packet names the tree of displayed subdivisions and retained cover leaves; it is not open-cover compactness, a host subcover selector, a selected Lebesgue number, quotient equality of real names, or countable choice.

Definition 2939.1 (Interval cover refinement tree carrier)**.** *An* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, D, R, M, L, E, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, naming the lower endpoint, upper endpoint, endpoint-order, containment, StreamName, RegSeqRat, and Real-seal data being covered.*
2. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *cover row whose finite dyadic endpoints and membership rows are inspected by the tree.*
3. *R is the* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *comparison row that records how a retained child cover refines its parent cover entry.*
4. *M is the finite mesh-depth ledger, listing the dyadic scale used at each visited tree node.*
5. *L is the finite leaf row. Each leaf records a dyadic subinterval, its parent node, and the displayed cover member that contains it at the tolerance named by M.*
6. *E is the edge row, recording the binary split relation between a node and its two dyadic children, together with the inherited endpoint-containment witnesses.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the interval endpoints and cover endpoints at the mesh scales named by M.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint and cover windows.*
9. *A is the visible* $\mathsf{Real}^{\uparrow}$ *seal row reached only through I, W, Q and the displayed dyadic cover data.*

10. *H stores componentwise* hsame *transport for interval, cover, refinement, mesh, leaf, edge, window, readback, seal, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from closed interval containment to dyadic subdivision, from subdivision to cover membership, from membership to refinement comparison, and from windows to the visible Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$ *row.*

The accepted carrier is only the finite tree packet displayed above. It contains no arbitrary open cover, no host finite-subcover selector, no selected point of the interval, no selected Lebesgue number, no quotient equality of real names, no countable-choice functional, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.IntervalCoverRefinementTreeUp.TasteGate.IntervalCoverRefinementTreeTasteGate single carrier alignment`

Theorem 2939.2 (Interval cover refinement tree NameCert obligations)**.** *For an accepted packet $T = (I, D, R, M, L, E, W, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$ *obligations are exhausted by closed bounded interval admission through I, dyadic cover admission through D, finite refinement admission through R, mesh-depth admission through M, leaf and edge admission through L and E, finite-window readback through W and Q, Real sealing through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier row by row. The interval source is exactly the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ packet I, the cover source is exactly the $\mathsf{DyadicIntervalCover}^{\uparrow}$ packet D, and the comparison source is exactly the $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet R.

The tree rows add only finite subdivision data. The mesh ledger M, leaves L, and edges E list the visited nodes, their dyadic children, and the retained cover members, so every local admission check is a finite projection from the displayed BHist packet.

The endpoint and cover reads remain on the displayed route. The $\mathsf{StreamName}^{\uparrow}$ row W, $\mathsf{RegSeqRat}^{\uparrow}$ row Q, and $\mathsf{Real}^{\uparrow}$ seal A are replayed by C and transported by H. Since the packet has no row for an arbitrary open cover, selected subcover, selected point, Lebesgue number choice, quotient equality, or completeness principle, no such datum can enter the local NameCert obligations. □

Theorem 2939.3 (Dyadic subcover route)**.** *Every retained subcover read exported by an accepted* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *packet factors through the finite route*

$$I \longmapsto D \longmapsto R \longmapsto M \longmapsto L \longmapsto E \longmapsto (W, Q, A, H, C, P, N).$$

In particular, a consumer may inspect only the dyadic child intervals, the finite leaf-cover assignments, the displayed refinement comparison rows, and the finite endpoint windows named in the carrier.

Proof. Start from the closed interval row I. It supplies the endpoint and containment surface before any cover or tree row is inspected.

Next read the dyadic cover and refinement rows. The cover packet D lists the available dyadic cover members, and the refinement packet R records how a child cover assignment is compared with a parent cover assignment. The mesh, leaf, and edge rows M, L, E then expose the finite tree spine on which these comparisons are made.

Finally pass through the structural rows. Transport H, replay C, provenance P, and local name N preserve the same finite route, while W, Q, A keep all endpoint reads behind finite StreamName, RegSeqRat, and Real-seal rows. No additional subcover or compactness object is available. □

Theorem 2939.4 (Choice-free nonescape)**.** *An* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *consumer cannot escape the finite refinement tree by invoking open-cover compactness, a host finite-subcover selector, a selected Lebesgue number, quotient equality of real names, a chosen point in the interval, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every accepted read is determined by the finite rows* $I, D, R, M, L, E, W, Q, A, H, C, P, N$ *and their displayed* hsame *and* Cont *routes.*

Proof. The carrier definition lists every coordinate of the packet. The analytic coordinates are $I, D, R, M, L, E, W, Q, A$, and the structural coordinates are H, C, P, N.

Apply the NameCert obligation theorem. It makes those rows the whole local naming surface and gives no replay path to an arbitrary open cover, host subcover selector, selected Lebesgue number, quotient equality, or completeness coordinate.

The dyadic subcover route supplies the consumer-facing consequence. Any accepted read must pass through the same finite interval, cover, refinement, tree, window, readback, and seal rows, so choice-based or host-level data cannot affect the result. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a DyadicIntervalCoverUp cover row, a FiniteCoverRefinementUp comparison row, finite mesh-depth, leaf, and edge rows, StreamName windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$, seed)

Scope closed: Definition 3916.1, Definition 3916.2, Definition 3916.3, and Definition 3916.4 bind the finite dyadic refinement-tree seed over closed bounded interval covers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.IntervalCoverRefinementTreeUp.TasteGate.IntervalCoverRefinementTreeTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel compactness, selected subcovers, selected Lebesgue numbers, quotient real equality, countable choice, Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic leaf coverage, refinement transport, Real-seal nonescape, and tree-ledger exactness for the displayed packet.

Chapter 2940

A Concrete Naming Certificate for $\mathsf{NeumannSeries}^{\uparrow}$

Following Banach algebra treatments of Neumann series, the $\mathsf{NeumannSeries}^{\uparrow}$ packet records the finite inverse-series certificate by which a norm-small operator or algebra element exposes partial sums, residual checks, and inverse-readback rows before any host inverse selector is available. It sits beside the Banach-algebra source of chapter 2189, the bounded-operator route of chapter 1644, the scalar norm-bound surface of chapter 3603, the resolvent identity packet of chapter 3715, and the spectral-radius boundary of chapter 2352. The packet names a finite inverse route: it does not select a completed inverse from the host, assert a completed spectrum, or import a quotient equality of operators.

Definition 2940.1 (Neumann inverse-series carrier)**.** *A* $\mathsf{NeumannSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (A, T, K, W, S, Q, R, E, H, C, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row, carrying multiplication, unit, norm, completeness, product-control, and local naming data.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *or algebra-action row whose smallness is being inspected.*
3. *K is the norm-smallness ledger, read through* $\mathsf{OperatorNorm}^{\uparrow}$ *or the Banach-algebra norm, recording the displayed bound below the unit row.*
4. *W is the finite partial-sum window for the inspected powers $1, T, T^2, \ldots, T^m$.*
5. *S is the finite sum row naming the displayed Neumann partial sums and their multiplication order.*
6. *Q is the residual row, recording the left and right error checks for $(1-T)S_m$ and $S_m(1-T)$ inside the same algebra surface.*
7. *R is the inverse-check row saying which residual bounds are strong enough for an inverse-facing consumer to read the packet.*
8. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for rational norm bounds, tail estimates, and terminal real seals.*
9. *H records componentwise* hsame *transport for A, T, K, W, S, Q, R, E.*

10. *C records displayed* Cont *replay from the smallness request through the finite partial-sum and residual rows.*
11. *P is the* Pkg *provenance over* $\mathsf{NeumannSeries}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-chosen inverse, completed resolvent set, completed spectrum, spectral theorem, quotient operator equality, selected approximate eigenvectors, or countable choice.

Lean4 (definition) `BEDC.Derived.NeumannSeriesUp.NeumannSeriesUp`

Theorem 2940.2 (Neumann series NameCert obligations)**.** *For every accepted* $\mathsf{NeumannSeries}^{\uparrow}$ *carrier* $N = (A, T, K, W, S, Q, R, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach-algebra source, bounded-operator or algebra-action row, norm-smallness ledger, finite power window, finite partial sums, residual checks, inverse-check row, rational and real readback, transport, replay, provenance, and local naming. Every inverse-facing consumer must read those rows before using the inverse-series conclusion.*

Proof. Project the carrier of Definition 3917.1. The algebra source A and operator row T are the only endpoints on which multiplication and powers may be read.

The smallness and finite-sum rows are K, W, S. The norm-smallness ledger is displayed before any partial sum is public, and the finite power window fixes the product order used by the sum row.

The residual and readback rows are Q, R, E. They expose the left and right inverse checks and the rational-to-real tail estimates before a consumer may treat the packet as an inverse route.

Finally, H, C, P, L transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a host inverse, completed resolvent set, completed spectrum, quotient equality, or choice schedule, the listed rows exhaust the local NameCert obligation surface. □

Theorem 2940.3 (Neumann inverse route)**.** *Every resolvent, spectral-radius, or inverse-operator consumer of an accepted* $\mathsf{NeumannSeries}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto K, W, S, Q, R, E \longmapsto H, C, P, L.$$

Thus a downstream read receives a norm-smallness ledger, finite partial-sum windows, residual and inverse-check rows, rational and real readback, transport, replay, provenance, and local naming before it may cite an inverse-series conclusion.

Proof. Start with an accepted packet $N = (A, T, K, W, S, Q, R, E, H, C, P, L)$. The Banach-algebra and bounded-operator rows fix the multiplication, unit, action, and norm-bound surface.

The route then passes through K, W, S. These rows expose the smallness bound, the inspected powers, and the finite partial sums; no completed infinite sum is available at this stage.

The inverse-facing part is Q, R, E. The residual row records the two-sided error checks, the inverse-check row states which displayed residual bound is consumed, and the readback row seals the rational tail estimates as real data.

The structural rows H, C, P, L preserve this route for chapter 3715 and chapter 2352 consumers. Because the packet lacks host inverse, completed spectrum, quotient equality, and choice coordinates, every such consumer factors through the displayed inverse-series route. □

Falsifiable prediction: Every accepted $\mathsf{NeumannSeries}^{\uparrow}$ inverse-facing read exposes BanachAlgebra, BoundedLinearOperator, norm-smallness, finite partial-sum, residual, inverse-check, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows before citing the inverse-series conclusion.

Independence witness: banachalgebra, boundedlinearoperator, operatornorm, resolvent-identity, spectral-radius: $\mathsf{NeumannSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries norm-smallness, finite power windows, partial sums, residual checks, inverse-readback rows, rational and real tail seals, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NeumannSeries}^{\uparrow}$ packet is generated from a BanachAlgebraUp source row, a BoundedLinearOperatorUp or algebra-action row, a norm-smallness ledger, finite partial-sum windows, residual and inverse-check rows, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3917.1, Definition 3917.2, and Definition 3917.3 seed the finite Neumann inverse-series route consumed by resolvent, spectral, and inverse-operator packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a completed inverse operator, a completed resolvent set, a completed spectrum, quotient operator equality, selected approximate eigenvectors, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, norm-smallness exactness, partial-sum window stability, residual readback, inverse-check exactness, rational and real tail sealing, transport and replay stability, and consumer nonescape for BEDC.Derived.NeumannSeriesUp.

Chapter 2941

A Concrete Naming Certificate for CauchyWindowStabilityBridge$^{\uparrow}$

Following Bishop and Bridges, the CauchyWindowStabilityBridge$^{\uparrow}$ packet records the finite tail-window stability bridge used by the regular-Cauchy route of chapter 9238. It sits after the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The bridge names shared finite tail-window transport; it is not a hidden limit selector, quotient stream equality, host equality of real names, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 2941.1 (Cauchy window stability bridge carrier)**.** *A* CauchyWindowStabilityBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S_0, S_1, T, D, R_0, R_1, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are* StreamName$^{\uparrow}$ *finite-window rows for the two regular-Cauchy presentations being compared.*
2. T *is the common finite tail-window row, listing the precision requests and indices beyond which the two presentations are read through the same displayed window schedule.*
3. D *is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to the shared tail window and to every rational comparison used by the bridge.*
4. R_0 *and* R_1 *are* RegSeqRat$^{\uparrow}$ *readback rows extracted from* S_0 *and* S_1 *at the tolerances named by* D*.*
5. E_0 *and* E_1 *are the visible* Real$^{\uparrow}$ *seal rows reached only after the shared tail window, dyadic tolerance ledger, and regular rational readbacks have been displayed.*
6. H *stores componentwise* hsame *transport for stream windows, the common tail row, dyadic tolerances, regular readbacks, Real seals, provenance, and local naming rows.*
7. C *stores displayed* Cont *replay from precision requests to common tail windows, from windows to regular readbacks, from dyadic tolerances to comparison rows, and from readbacks to the two Real seals.*
8. P *is the* Pkg *provenance over* CauchyWindowStabilityBridge$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$ *row.*
The accepted carrier contains no selected limit, no quotient representative for streams, no host equality of real names, no countable-choice sequence, and no ambient completeness coordinate. It names only the finite common-tail bridge displayed by $T, D, R_0, R_1, E_0, E_1, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CauchyWindowStabilityBridgeUp.CauchyWindowStabilityBridgeTasteGate single carrier alignment`

Theorem 2941.2 (Cauchy window stability bridge NameCert obligations)**.** *For an accepted carrier $B = (S_0, S_1, T, D, R_0, R_1, E_0, E_1, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$ *obligations are exactly carrier admission; finite window exposure through S_0 and S_1; common tail-window admission through T; dyadic tolerance exactness through D; regular rational readback through R_0 and R_1; Real-seal admission through E_0 and E_1; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier. The two stream rows S_0, S_1, the common tail row T, the dyadic ledger D, the readbacks R_0, R_1, and the seals E_0, E_1 are the entire analytic surface of the bridge.

The tail-window row is the only stability interface. It records which finite windows are shared for the precision request under consideration, while the dyadic ledger records the rational tolerance used to compare the two readbacks.

The structural rows add no hidden equality or limit object. Transport H, replay C, provenance P, and the local name N preserve the displayed rows and expose no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 2941.3 (Tail-window transport)**.** *Every stability read exported by an accepted* $\mathsf{CauchyWindowStabilityBrid}$ *packet factors through the finite route*

$$S_0, S_1 \longmapsto T \longmapsto D \longmapsto R_0, R_1 \longmapsto E_0, E_1 \longmapsto H, C, P, N.$$

Thus shared regular-Cauchy tail transport is determined by displayed $\mathsf{StreamName}^{\uparrow}$ *windows, one common finite tail row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and componentwise* hsame *replay.*

Proof. Begin with the two finite stream rows. A stability consumer may inspect only the windows exposed by S_0 and S_1.

Read the common tail row next. The row T names the shared finite window schedule for the current precision request, and the dyadic ledger D attaches the tolerance used to compare the two regular rational readbacks.

The regular readbacks and seals are downstream. Rows R_0, R_1 are consumed before E_0, E_1, and the structural rows H, C, P, N replay the same finite route. No selected tail, quotient stream, or countable-choice family is available outside this packet. □

Theorem 2941.4 (Real-seal stability handoff)**.** *For every accepted* $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ *packet, the two* $\mathsf{Real}^{\uparrow}$ *seal rows E_0 and E_1 are accessible only after the shared tail-window row T, dyadic tolerance row D, and regular rational readbacks R_0, R_1 have been displayed. Consequently a Real-completion consumer that cites this bridge receives a finite seal-stability handoff, not a proof of ambient* $\mathsf{Real}^{\uparrow}$ *completeness or quotient equality of real names.*

Proof. Project the carrier and isolate the seal coordinates. The rows E_0, E_1 occur only as terminal rows of the displayed bridge packet.

Apply the tail-window transport theorem. It forces every stability read through S_0, S_1, T, D, R_0, R_1 before the seals are available.

The refusal clause follows from carrier exhaustion. Since the packet has no coordinate for ambient completeness, quotient equality, host equality, selected limits, or countable choice, the Real-facing handoff can export only the displayed finite stability route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ packet is generated from paired StreamName finite windows, a common finite tail-window row, a DyadicRatCore tolerance ledger, paired RegSeqRat readbacks, paired RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3918.1, Definition 3918.2, Definition 3918.3, and Definition 3918.4 bind the finite regular-Cauchy tail-window stability bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, host equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, common-tail exactness, dyadic tolerance transport, paired readback stability, Real-seal handoff, and nonescape for the displayed packet.

Chapter 2942

A Concrete Naming Certificate for MetaCICPiAdequacyBudget$^{\uparrow}$

The MetaCICPiAdequacyBudget$^{\uparrow}$ packet realizes the Pi-adequacy route of chapter 9383 as finite BEDC data. It sits beside the critical-path packet of chapter 765, the normalization frontier of chapter 892, the candidate socket of chapter 3566, the normalization-confluence bridge of chapter 1190, and the normalization audit packet of chapter 3779. The packet records the candidate-mediated budget needed by Pi adequacy; it does not claim unconditional global normalization or erase the candidate evidence consumed by the Lean route.

Definition 2942.1 (MetaCIC Pi adequacy budget carrier)**.** *A* MetaCICPiAdequacyBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, \Pi, N, A, S, F, O, H, C, P, L)$$

with the following displayed rows:

1. *G is the closed context and typing-frontier row for the Pi candidate being inspected.*
2. *Π is the Pi-candidate evidence row consumed by the route named by* `PiAdequacy.lean`*.*
3. *N is the neutral-normal evidence row used by the neutral branch of the adequacy argument.*
4. *A is the application-side adequacy handoff row, recording the candidate argument and the application result row.*
5. *S is the strong-normalization handoff row, naming the projections corresponding to* `piCandidate_implies_sn`*,* `neutralNormal_implies_sn`*, and* `sn_pi`*.*
6. *F is the frontier-refusal row, keeping candidate evidence, subject-reduction boundaries, and confluence boundaries visible whenever a consumer requests a stronger normalization claim.*
7. *O is the obligation-budget row for the remaining endpoint evidence needed by closed-consistency consumers.*
8. *H records componentwise* hsame *transport for the typing, Pi-candidate, neutral-normal, application, strong-normalization, frontier, and obligation rows.*
9. *C records displayed* Cont *replay from a Pi-adequacy request through G, Π, N, A, S, F, O.*
10. *P is* Pkg *provenance over* MetaCICPiAdequacyBudget$^{\uparrow}$*,* MetaCICNormalizationAuditPacket$^{\uparrow}$*,* MetaCICDecidableNe MetaCICFuelTrace$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, global confluence, semantic equality, quotient syntax equality, or host evaluation soundness.

Lean4 (definition) `BEDC.Derived.MetaCICPiAdequacyBudgetUp.taste gate`

Theorem 2942.2 (MetaCIC Pi adequacy budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ *carrier* $B = (G, \Pi, N, A, S, F, O, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$ *surface consists exactly of typing-frontier admission, Pi-candidate evidence admission, neutral-normal evidence admission, application adequacy handoff, strong-normalization projection handoff, frontier refusal, obligation-budget exposure, componentwise transport, displayed replay, provenance, and local naming. A consumer receives no stronger normalization endpoint unless that endpoint is supplied as a displayed row of the same packet.*

Proof. Project the carrier of Definition 3919.1. The typing row G, Pi-candidate row Π, and neutral-normal row N are the only source evidence available to the adequacy budget.

The handoff rows are A and S. They expose the application-side candidate route and the named strong-normalization projections without replacing candidate evidence by an evidence-free theorem.

The boundary rows are F and O. They keep the normalization frontier, subject-reduction boundary, confluence boundary, and endpoint obligation budget visible to any closed-consistency consumer.

Finally, H, C, P, L transport, replay, source, and name the same finite route. Because the carrier lacks coordinates for global normalization, unrestricted subject reduction, confluence, semantic equality, quotient syntax equality, and host evaluation soundness, the listed rows exhaust the local NameCert surface. □

Theorem 2942.3 (MetaCIC Pi adequacy strong-normalization handoff)**.** *Every normalization-facing consumer of an accepted* $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ *packet factors through the route*

$$G, \Pi, N, A \longmapsto S \longmapsto F, O, H, C, P, L.$$

The consumer therefore reads closed context data, Pi-candidate evidence, neutral-normal evidence, application adequacy data, the strong-normalization handoff, frontier refusal, obligation budget, transport, replay, provenance, and local naming before citing the Pi adequacy contribution to candidate-mediated strong normalization.

Proof. Start with an accepted packet $B = (G, \Pi, N, A, S, F, O, H, C, P, L)$. The source rows G, Π, N, A provide the only candidate and neutral-normal evidence made public by the packet.

The strong-normalization handoff is S. It packages the displayed Pi-candidate, neutral-normal, and Pi-shaped SN projections as rows of the same finite carrier, so a consumer cannot cite them after dropping their evidence inputs.

The frontier and obligation rows F, O remain attached to the handoff. They record that subject-reduction, confluence, and global closed-normalization endpoints are not exported by the Pi adequacy budget alone.

The rows H, C, P, L preserve and name this route. Hence any normalization-facing consumer factors through the displayed candidate-mediated budget before it may use the Pi adequacy contribution. □

Falsifiable prediction: An accepted $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ normalization-facing read is refuted if it exports Pi-shaped strong-normalization data while omitting the displayed typing-frontier, Pi-candidate, neutral-normal, application-handoff, SN-handoff, frontier-refusal, obligation-budget, transport, replay, provenance, or local NameCert rows.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, `normalization_candidate_socket`, metacicnormalizationconfluencebridge: $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Pi-candidate evidence, neutral-normal evidence, application adequacy handoff, SN projection rows, frontier refusal, obligation-budget exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ packet is generated from closed context and typing-frontier data, Pi-candidate evidence, neutral-normal evidence, application adequacy rows, strong-normalization handoff rows, frontier refusal, obligation budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3919.1, Definition 3919.2, and Definition 3919.3 seed the Pi adequacy budget consumed by candidate-mediated strong-normalization routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, evidence-free closed-term strong normalization, unrestricted subject reduction, global confluence, semantic equality, quotient syntax equality, or host evaluation soundness.

Upgrade path: Obligation closure requires checked carrier admission, Pi-candidate evidence exactness, neutral-normal readback, application adequacy handoff, strong-normalization projection handoff, frontier refusal stability, obligation-budget exposure, and consumer nonescape for BEDC.Derived.MetaCICPiAdequacyBudgetUp.

$\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ consumes the Cauchy finite-window surface of chapter 3413 together with the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route displayed by its child chapter.

Chapter 2943

Finite Cauchy Diagonal NameCert Construction

Following Bishop and Bridges, the $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ packet records the finite diagonal selector needed when a displayed Cauchy family is read through regular rational windows and then sealed as a real name. It sits between the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational route of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet supplies only a finite selector surface for completion and Real consumers; it does not choose an infinite subsequence, a quotient representative, or a host limit.

Definition 2943.1 (Finite Cauchy diagonal carrier)**.** *A* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, W, K, S, R, Y, E, H, C, P, N)$$

with the following rows:

1. *F is the finite family row, naming the displayed Cauchy sources whose diagonal window is requested.*
2. *W is the selected-window classifier row, recording the finite window chosen for each displayed precision request and each displayed source row.*
3. *K is the classifier-stability row, comparing selected windows only by the same* hsame *and* Cont *routes used by the source family.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule exposed by the selected windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the diagonal comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, available only after F, W, K, S, R, Y have been displayed.*
8. *H records componentwise* hsame *transport for the family, window, stability, stream, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for diagonal-selection requests through $F, W, K, S, R, Y, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$ *row.*

The classifier reads only $F, W, K, S, R, Y, E, H, C, P, N$. There is no coordinate for countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2943.2 (Finite Cauchy diagonal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ *carrier $D = (F, W, K, S, R, Y, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *finite diagonal carrier admission through the displayed family row F;*
2. *selected-window classifier admission through W, with one displayed finite window for each displayed precision request and source row;*
3. *classifier stability through K, H, C, preserving the selected windows under componentwise* hsame *transport and displayed* Cont *replay;*
4. *diagonal extraction ledger admission through S, R, Y, requiring every extracted read to pass through the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;*
5. $\mathsf{Real}^{\uparrow}$ *seal handoff through E, requiring the finite family, selected-window classifier, stability row, stream schedule, readback row, dyadic ledger, transport, replay, provenance, and local naming to remain visible to the terminal real-name consumer.*

No obligation row certifies countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, or ambient completeness.

Proof. Project the carrier of Definition 3920.1. The visible rows are exactly $F, W, K, S, R, Y, E, H, C, P, N$, so the finite family and selected-window classifier are the only routes by which a diagonal request enters the packet.

The stability row is read before any terminal seal. Transport by H and replay by C preserve the same family row, selected-window row, and classifier-stability row, so a replayed request cannot replace the finite selector by an infinite choice schedule or by host sequence equality.

The extraction ledger is then read through S, R, Y. The stream schedule supplies finite observations, the regular rational row repacks those observations, and the dyadic row records the tolerance budget for the same displayed windows.

Finally the Real handoff is downstream from the ledger. Since E is available only after $F, W, K, S, R, Y, H, C, P, N$ have been displayed, every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives a finite diagonal selector surface rather than a hidden limit or quotient representative. □

Falsifiable prediction: An accepted $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ read is refuted if the Real-facing diagonal handoff reaches $\mathsf{Real}^{\uparrow}$ without displaying the finite family row, selected-window classifier, classifier-stability row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite family admission, selected-window classification, classifier stability, diagonal extraction, Real seal handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ packet is generated from a displayed finite Cauchy family, selected finite windows, classifier-stability rows, StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3920.1 and Definition 3920.2 seed the finite diagonal selector route over family, selected-window, stability, StreamName, RegSeqRat, DyadicRatCore, RealUp seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteCauchyDiagonal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, selected-window classifier admission, classifier stability, diagonal extraction, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyDiagonalUp.

The residual-envelope packet reads its conditional confluence boundary through chapter 3791, chapter 3827, and chapter 765 before the child file spells out the carrier.

Chapter 2944

MetaCIC Confluence Residual Envelope NameCert Construction

The MetaCICConfluenceResidualEnvelope$^{\uparrow}$ packet realizes the conditional residual-substitution boundary of chapter 9378 as finite MetaCIC NameCert data. It sits beside the substitution-boundary packet of chapter 3791, the residual checker of chapter 3827, the finite fuel trace of chapter 3889, the decidable neutral spine of chapter 3909, and the normalization evidence budget of chapter 3871. The envelope records which residual windows, substitution-boundary rows, checker rows, fuel traces, neutral-spine reads, and replay ledgers are visible before a consumer cites a conditional confluence handoff.

The residual envelope also consumes the closedness-preservation ledger of chapter 795 before any residual checker can be read as a confluence boundary.

Falsifiable prediction: A certification route to BetaParallelDiamondClosed that omits the residual-envelope row refutes MetaCICConfluenceResidualEnvelope$^{\uparrow}$, because the conditional substitution boundary has then been consumed without displaying the finite residual window, substitution-boundary, checker, fuel, neutral-spine, replay, provenance, and local NameCert rows.

Independence witness: metacic_fuel_trace, metacic_decidable_neutral_spine, metacic_confluence_substitut normalization_evidence_budget: none is bijective to MetaCICConfluenceResidualEnvelope$^{\uparrow}$ because the envelope jointly records residual windows, substitution-boundary exposure, checker status, bounded fuel, neutral-spine reads, normalization-budget non-escape, transport, replay, provenance, and local naming as one confluence-boundary packet.

Definition 2944.1 (MetaCIC confluence residual envelope carrier)**.** *A* MetaCICConfluenceResidualEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$$

with the following rows:

1. *T is the typed-window row naming the MetaCIC source and target windows under inspection.*
2. *R is the residual-frontier row, recording the beta-redex residues whose join is conditional on substitution compatibility.*
3. *S is the* MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ *row exposed before any closed diamond handoff.*
4. *B is the bounded residual-checker row inherited from* MetaCICConfluenceResidualChecker$^{\uparrow}$*.*

5. *J is the local-join ledger for those residual windows.*
6. *O is the obstruction row retaining the named substitution-compatibility boundary.*
7. *F is the* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *bounded replay row.*
8. *U is the* $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ *neutral-spine read used to prevent atom and neutral cases from being promoted to beta branches.*
9. *Z is the* $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ *row, recording that normalization evidence may be inspected only as bounded finite evidence and not as global normalization.*
10. *H records componentwise* hsame *transport for typed windows, residual frontiers, substitution boundary, checker status, local joins, obstruction, fuel, neutral-spine, budget, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay for conditional confluence requests through $T, R, S, B, J, O, F, U, Z, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$, NormalizationEvidenc BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *row.*

The classifier reads only $T, R, S, B, J, O, F, U, Z, H, C, P, N$. It has no coordinate for an unconditional BetaParallelSubstitutionCompatibility *proof, an unconditional* BetaParallelDiamondClosed *proof, arbitrary closed Church–Rosser, erased substitution transport, or host syntax equality.*

Theorem 2944.2 (Residual envelope NameCert surface)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission, typed-window exposure, residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, obstruction retention, fuel-trace replay, neutral-spine separation, normalization-budget non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3921.1. The only visible route to a conditional confluence read is the displayed tuple $T, R, S, B, J, O, F, U, Z, H, C, P, N$.

The residual and substitution-sensitive rows R, S, B, J, O are read before any closed-diamond handoff. Thus a consumer can see the residual frontier, the named substitution boundary, the bounded checker status, the local join ledger, and the obstruction row as finite data rather than as a hidden global proof.

The supporting rows F, U, Z then separate bounded computation, neutral-spine cases, and normalization evidence from the residual-substitution boundary. They keep fuel traces, neutral reads, and normalization budgets visible without converting them into global normalization or confluence.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no remaining coordinate contains an unconditional substitution-compatibility or closed-diamond proof, the packet certifies only the residual-envelope boundary and not the endpoint theorem. □

Theorem 2944.3 (MetaCIC confluence residual critical-pair audit surface)**.** *For an accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet with two displayed one-step residual branches, the finite critical-pair audit surface consists only of the source term row, the two residual branch rows, the shared candidate mediator row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *row. The audit surface does not*

assert global confluence, strong normalization, decidability of all residual joins, or host quotient conversion.

Proof. Project the carrier of Definition 3921.1. The typed-window row T supplies the source term row, while R, B, J, O display the residual frontier, bounded checker status, local join ledger, and obstruction boundary from which the two one-step branches and the candidate mediator are read.

Apply the residual envelope NameCert surface theorem. It requires residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, obstruction retention, bounded fuel replay, neutral-spine separation, normalization-budget non-escape, transport, replay, provenance, and local naming before any confluence-facing consumer reads the envelope.

Compare the two displayed branch rows only through the candidate mediator row retained in J and the obstruction boundary retained in O. The transport row H and replay row C preserve those displayed rows componentwise; they do not add a new conversion relation or quotient syntax equality.

Carrier-field exhaustion gives the non-escape claim. Since the accepted packet has no coordinate for global confluence, strong normalization, full join decidability, or host quotient conversion, the critical-pair read is only the finite audit surface stated above. □

Theorem 2944.4 (MetaCIC confluence residual envelope checker-status determinacy)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier* $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$*, any confluence consumer that reads the bounded checker status* B *must read exactly one of the displayed accepted, rejected, or obstruction alternatives supplied by the residual-checker route. The later rows* $J, O, F, U, Z, H, C, P, N$ *cannot change that status and cannot turn it into unconditional* BetaParallelDiamondClosed*.*

Proof. Project the carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$ of Definition 3921.1. The checker status is the single row B, inherited from the residual-checker route of chapter 3827.

The residual-checker route separates accepted, rejected, and obstruction status as displayed finite alternatives. Reading B therefore exposes one of those alternatives before the envelope rows $J, O, F, U, Z, H, C, P, N$ are inspected.

The obstruction row O retains the named residual-substitution obstruction rather than erasing it. The rows F, U, Z separate bounded computation, neutral-spine reads, and normalization budgets from the checker status, so they cannot promote a bounded status read into an endpoint confluence theorem.

Finally H, C, P, N transport, replay, source, and name only the displayed carrier rows. Because no later row contains an unconditional BetaParallelDiamondClosed coordinate, the checker status remains the one supplied by B. □

Theorem 2944.5 (MetaCIC residual envelope obstruction monotonicity)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier* $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$*, if the bounded checker row* B *exposes obstruction status and the row* O *retains the residual-substitution obstruction, then every read of* J, F, U, Z, H, C, P, N *preserves that obstruction boundary. In particular, those later rows cannot erase the obstruction status and cannot export an unconditional* BetaParallelDiamondClosed *theorem.*

Proof. Project B and O from the carrier of Definition 3921.1. The row B supplies the displayed checker status, and O is the only row that records the residual-substitution obstruction retained by the envelope.

Apply checker-status determinacy, Definition 3921.4. Once B exposes obstruction status, the rows inspected after B cannot replace that status by an accepted or rejected alternative.

Now inspect the remaining carrier inventory. The rows J, F, U, Z contain local joins, bounded fuel, neutral-spine reads, and normalization-budget evidence, while H, C, P, N contain transport, replay, provenance, and local naming data. None of these rows contains a closed-diamond coordinate.

Therefore the obstruction retained by O remains visible through every later read. Since the accepted carrier has no unconditional BetaParallelDiamondClosed coordinate, no consumer can obtain that endpoint theorem from J, F, U, Z, H, C, P, N. □

Theorem 2944.6 (MetaCIC confluence residual envelope neutral-spine separation)**.** *Any conditional confluence consumer that attempts to promote a neutral-spine read U into a beta residual branch must also display the residual-frontier row R, substitution-boundary row S, checker row B, local join row J, fuel trace F, and normalization budget Z. Without those rows, the accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet exports only a neutral case and no beta branch.*

Proof. Project the carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$ of Definition 3921.1. The neutral-spine row U is a displayed case-separation row, not a residual-frontier row.

Apply the NameCert surface theorem Definition 3921.2. It lists residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, fuel-trace replay, neutral-spine separation, and normalization-budget non-escape as distinct obligations of the accepted packet.

Therefore a consumer that reads only U has only the neutral case. To promote the read into a beta residual branch, the consumer must also display R, S, B, J, F, Z through the same transport, replay, provenance, and local naming rows. If any of these rows is absent, the carrier has no coordinate from which a beta residual branch can be exported. □

Theorem 2944.7 (MetaCIC residual envelope closed-diamond consumption boundary)**.** *Any consumer attempting to cite* BetaParallelDiamondClosed *through an accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet must first display the full row inventory*

$$T, R, S, B, J, O, F, U, Z, H, C, P, N.$$

If the substitution-boundary row S, checker row B, obstruction row O, fuel row F, or normalization-budget row Z is absent, the packet exports only a conditional residual envelope and not closed confluence.

Proof. Start with the NameCert surface theorem, Definition 3921.2. It identifies the accepted packet with the displayed rows $T, R, S, B, J, O, F, U, Z, H, C, P, N$ and says that no unconditional closed-diamond coordinate is part of the carrier.

Next read the checker and obstruction rows. The checker-status determinacy theorem fixes the status exposed by B, while Definition 3921.5 preserves any obstruction retained in O through the later local-join, fuel, neutral-spine, budget, transport, replay, provenance, and naming rows.

Finally apply neutral-spine separation. Theorem 3921.6 refuses to promote a neutral-spine read into a beta residual branch unless the residual frontier, substitution boundary, checker, local join, fuel trace, and normalization budget are all displayed.

Thus a consumer that omits S, B, O, F, or Z has not consumed the finite residual-substitution boundary required for the endpoint. It can read only the conditional residual envelope carried by the displayed rows, and no route to BetaParallelDiamondClosed is exported. □

Theorem 2944.8 (MetaCIC residual envelope diamond route)**.** *Every* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *read that exposes a closed-diamond consumer row factors first through the residual-substitution boundary of chapter 2181, the bounded residual-substitution frontier of chapter 3808, and the MetaCIC open-frontier ledger of chapter 762. The residual envelope may then expose the closed-diamond consumer only through the displayed rows* $T, R, S, B, J, O, F, U, Z, H, C, P, N$*, and it exports no direct route to* BetaParallelDiamondClosed*.*

Proof. Start from the closed-diamond consumption boundary, Definition 3921.7. It requires a consumer to display the full row inventory $T, R, S, B, J, O, F, U, Z, H, C, P, N$ before the closed-diamond endpoint can be cited.

The substitution-sensitive part of that inventory is not local to the envelope alone. The residual-substitution boundary keeps the substitution premise visible, the bounded residual-substitution frontier bounds the frontier read, and the open-frontier ledger records the confluence endpoint as a frontier entry rather than as a discharged theorem.

Reading S, B, J, O through those siblings fixes the route before the consumer can inspect F, U, Z, H, C, P, N. The fuel, neutral-spine, budget, transport, replay, provenance, and naming rows preserve the same route and do not add a closed-diamond coordinate.

Therefore the closed-diamond consumer row, when present, is exposed only after the three sibling packets and the envelope inventory have been read. No direct route to BetaParallelDiamondClosed is available from the residual envelope alone. □

Theorem 2944.9 (MetaCIC residual envelope closed-join boundary)**.** *Every closed-join or closed-diamond consumer of an accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet* $E = (T, R, S, B, J, O, F, U, Z, H, C, P$ *must read the envelope rows in the following boundary order: typed window, residual frontier, substitution boundary, checker status, local join, obstruction, fuel, neutral spine, normalization budget, transport, replay, provenance, and local naming. Equivalently, a consumer may cite the closed-join row only after the displayed route*

$$T, R, S, B, J, O, F, U, Z, H, C, P, N$$

has been consumed. The boundary exports no unrestricted BetaParallelDiamondClosed *theorem, no unrestricted substitution compatibility, no subject reduction theorem, and no Church–Rosser theorem.*

Proof. Begin with the NameCert surface theorem, Definition 3921.2. It fixes the visible surface of an accepted envelope as precisely $T, R, S, B, J, O, F, U, Z, H, C, P, N$.

Next read the residual-sensitive part of the route. The checker-status determinacy theorem fixes B before the local join is inspected, while obstruction monotonicity keeps O visible through the later rows. Thus J can be read only as a local join under the already displayed residual frontier, substitution boundary, checker status, and obstruction boundary.

Apply the closed-diamond consumption boundary and the diamond route. Together they force any closed-diamond or closed-join consumer through the same full inventory, and the sibling residual-substitution boundary, bounded frontier, and open-frontier ledger prevent the envelope from presenting the endpoint as a discharged global theorem.

Finally inspect the excluded coordinates. Since the carrier has no row for unrestricted closed diamond, substitution compatibility, subject reduction, or Church–Rosser, the closed-join read is exactly the ordered finite boundary stated above. □

The Lean TasteGate site for MetaCICConfluenceResidualEnvelope$^{\uparrow}$ must provide explicit Nontrivial and FieldFaithful instances for this carrier surface. These instances must keep the residual windows, substitution boundary, checker status, local joins, obstruction row, fuel trace, neutral spine, normalization budget, transport, replay, provenance, and local naming distinguishable.

Closure status for NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ **·**

Constructive story: A MetaCICConfluenceResidualEnvelope$^{\uparrow}$ packet is generated from typed windows, residual frontiers, a substitution-boundary row, bounded checker status, local joins, obstruction retention, bounded fuel traces, neutral-spine reads, normalization-budget rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 3921.1 and Definition 3921.2 seed the residual-substitution confluence envelope over typed-window, residual-frontier, substitution-boundary, checker, local-join, obstruction, fuel, neutral-spine, budget, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, arbitrary closed Church–Rosser, global normalization, Lean verification, bridge checking, erased substitution transport, or host syntax equality.

Upgrade path: Obligation closure requires checked carrier admission, residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join exposure, obstruction retention, fuel replay, neutral-spine separation, normalization-budget non-escape, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConfluenceResidualEnvelopeUp.

Chapter 2945

A Concrete Naming Certificate for $\mathsf{HahnBanach}^{\uparrow}$

Following Hahn and Banach, Bourbaki, and Bishop-style constructive functional analysis, the $\mathsf{HahnBanach}^{\uparrow}$ packet records the finite BEDC surface for bounded scalar functional extension rows. It sits beside the normed-space and Banach rows of chapter 1950 and chapter 90, the dual carrier of chapter 2265, the bounded-operator row of chapter 1644, the functional-analysis boundary of chapter 94, and the real-name seal of chapter 38. The chapter names a displayed finite extension ledger only; it is not the unrestricted classical Hahn–Banach theorem.

Definition 2945.1 (Hahn–Banach extension carrier)**.** *A* $\mathsf{HahnBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, B, D, F, E, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{NormedSpace}^{\uparrow}$ *or* $\mathsf{Banach}^{\uparrow}$ *row on which the extension is read.*
2. *B is the displayed subspace row, including the finite incidence data saying which inspected vectors lie in the subspace.*
3. *D is the* $\mathsf{BanachDual}^{\uparrow}$ *and* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for the bounded scalar functional on B.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *context row that records the local functional-analysis consumer boundary.*
5. *E is the extension row, displaying the scalar functional on the inspected ambient vectors and its agreement with D on the subspace row.*
6. *L is the norm-bound ledger, comparing the displayed bound for D with the displayed bound for E through* $\mathsf{Real}^{\uparrow}$ *endpoints.*
7. *H records componentwise* hsame *transport for $X, B, D, F, E, L, C, P, N$.*
8. *C records displayed* Cont *replay from ambient-space reads through the subspace, bounded-functional, extension, and norm-bound rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HahnBanach}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, Zorn maximality, or unrestricted ambient extension.

Lean4 ✓ `BEDC.Derived.HahnBanachUp.HahnBanachTasteGate single carrier alignment`

Theorem 2945.2 (Hahn–Banach NameCert obligations)**.** *For every accepted* $\mathsf{HahnBanach}^{\uparrow}$ *carrier* $K = (X, B, D, F, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$ *surface consists exactly of ambient normed or Banach admission, displayed subspace admission, bounded scalar functional admission, local functional-analysis context, extension-row agreement, norm-bound ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No extension obligation is stored outside those rows.*

Proof. Project the accepted carrier of Definition 3922.1. The rows $X, B, D, F, E, L, H, C, P, N$ are the full public inventory.

The bounded-functional read starts at the subspace row B and the scalar-functional row D. The extension row E can be inspected only with the same ambient row X and the same subspace incidence data.

The norm-bound ledger is L. It compares the displayed bound for D with the displayed bound for E through $\mathsf{Real}^{\uparrow}$ endpoints, while H, C, P, N only transport, replay, source, and name the same route.

Thus the NameCert obligations are exactly the listed finite rows. The unrestricted theorem, quotient duals, selected extensions, ultrafilter arguments, and choice principles have no carrier coordinate. □

Theorem 2945.3 (Hahn–Banach bounded functional extension route)**.** *Every bounded scalar functional extension read of an accepted* $\mathsf{HahnBanach}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{NormedSpace}^{\uparrow},\ \mathsf{Banach}^{\uparrow},\ \mathsf{BanachDual}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route first fixes the ambient and subspace rows, then reads the bounded scalar functional, then reads the displayed extension row and the norm-bound ledger, and only then exposes transport, replay, provenance, and local naming. It exports no unrestricted classical Hahn–Banach theorem, quotient dual space, selected extension, ultrafilter argument, countable choice, or detached extension principle.

Proof. Let $K = (X, B, D, F, E, L, H, C, P, N)$ be accepted. The carrier projection gives the ambient row X, the subspace row B, the bounded scalar functional row D, the functional-analysis context row F, and the extension row E.

The public extension route reads X and B before D. Only then can the consumer inspect E, because agreement with the original functional is meaningful only over the displayed subspace incidence row.

The norm-bound ledger L is read after the two functional rows. It records the displayed $\mathsf{Real}^{\uparrow}$ comparison between the original bound and the extension bound, so the consumer receives a finite bound ledger rather than an ambient domination theorem.

Finally H, C, P, N preserve this finite route. Since no row names quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, or unrestricted Hahn–Banach extension, none of those objects can be exported by the accepted packet. □

Falsifiable prediction: Dropping the norm-bound ledger L makes Definition 3922.3 false: a consumer could no longer distinguish a bounded extension row from an unbounded displayed scalar assignment.

Independence witness: normedspace, banach, banachdual, boundedlinearoperator, functional-analysis, real: $\mathsf{HahnBanach}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ambient space, subspace incidence, bounded scalar functional, extension agreement, norm-bound ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{HahnBanach}^{\uparrow}$ packet is generated from an ambient NormedSpaceUp or BanachUp row, a displayed subspace row, a BanachDualUp and BoundedLinearOperatorUp bounded scalar functional row, a FunctionalAnalysisUp context row, an extension row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$, seed)

Scope closed: Definition 3922.1, Definition 3922.2, and Definition 3922.3 seed the finite bounded scalar functional extension packet over normed-space, Banach, dual, bounded-operator, functional-analysis, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HahnBanachUp.HahnBanachTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim the unrestricted classical Hahn–Banach theorem, quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subspace incidence, bounded scalar functional transport, extension agreement, norm-bound exactness, non-escape, replay, provenance, and local naming for BEDC.Derived.HahnBanachUp.

Chapter 2946

A Concrete Naming Certificate for $\mathsf{RealIntervalCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealIntervalCompletion}^{\uparrow}$ packet records the finite completion-facing boundary at which located interval data, nested refinements, regular rational readbacks, and Real seals are read as one interval-completion carrier. It sits beside the located endpoint surface of chapter 394, the nested interval carrier of chapter 320, and the public completion reflector route of chapter 1585. The packet names the L10 Real and Completion handoff without importing Dedekind cuts, quotient completions, selected points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2946.1 (RealIntervalCompletion carrier)**.** *A* $\mathsf{RealIntervalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, N, D, W, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, carrying lower and upper located endpoint windows and their visible interval-order classifier.*
2. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row, recording the displayed nesting discipline and the dependency of each later interval on the earlier endpoint rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, carrying the dyadic shrinkage scale used by the refinement windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes only finite interval windows before any regular rational readback is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, reading the displayed interval windows as regular rational approximants.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the completed interval-facing read.*
7. *H records componentwise* hsame *transport for L, N, D, W, R, E and the structural rows.*
8. *C records displayed* Cont *replay from located endpoints through nested refinements, dyadic shrinkage, stream windows, regular rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalCompletion}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *Q is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^\uparrow}$ *row.*
The carrier has no coordinate for Dedekind cuts, quotient interval completions, selected interval points, open-cover compactness, countable choice, host real equality, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalCompletionUp.RealIntervalCompletionTasteGate single carrier alignment`

Theorem 2946.2 (RealIntervalCompletion NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalCompletion}^\uparrow$ *carrier* $I = (L, N, D, W, R, E, H, C, P, Q)$*, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located endpoint, nested refinement, dyadic radius, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-refinement classification, requiring every refinement read to pass through the* $\mathsf{LocatedInterval}^\uparrow$ *endpoint row and the* $\mathsf{NestedInterval}^\uparrow$ *nesting row before any completion-facing seal is visible;*
3. *diameter-shrinkage stability, requiring the* $\mathsf{DyadicRatCore}^\uparrow$ *radii to decrease along the displayed nested interval route and to remain stable under the classifier;*
4. *Real seal compatibility, requiring the* $\mathsf{StreamName}^\uparrow$ *windows and* $\mathsf{RegSeqRat}^\uparrow$ *readbacks to reach the* $\mathsf{Real}^\uparrow$ *seal only through the same finite interval route;*
5. *refusal of quotient intervals, selected limit points, countable-choice schedules, open-cover compactness, and ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Carrier admission is the projection of Definition 3923.1. The displayed tuple lists the located endpoints, nested refinements, dyadic radii, finite windows, regular rational readbacks, Real seal, transport, replay, provenance, and local naming row.

The interval-refinement classifier is the route L, N. A consumer must read located endpoint data before it can inspect the nested refinement row, so the packet cannot replace the route by a host interval completion or by selected endpoint representatives.

Diameter-shrinkage stability is carried by D together with the nesting row N. The dyadic radii are read as finite $\mathsf{DyadicRatCore}^\uparrow$ data, and componentwise hsame transport in H preserves the same shrinkage classifier.

Real seal compatibility is the route W, R, E. The stream windows feed the regular rational readback before the Real seal is exposed, and Cont replay in C preserves this order for downstream completion consumers.

Finally, P and Q name exactly the listed rows. Since the carrier contains no Dedekind-cut, quotient-interval, selected-point, choice, open-cover compactness, or ambient-completeness coordinate, those objects cannot be exported from an accepted packet. □

Theorem 2946.3 (RealIntervalCompletion completion handoff)**.** *Every* $\mathsf{CauchyCompletionReflector}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *consumer of an accepted* $\mathsf{RealIntervalCompletion}^\uparrow$ *carrier receives the finite route*

$$L, N, D, W, R, E, H, C, P, Q$$

and no independent completion object. The completion-facing read factors through located interval endpoints, nested interval refinement, dyadic diameter shrinkage, finite stream windows, regular rational readback, and the Real seal.

Proof. Start with an accepted consumer read. Carrier admission exposes only the rows $L, N, D, W, R, E, H, C, P, Q$.

The interval part is L, N, D. It fixes located endpoints, the displayed refinement order, and the dyadic shrinkage scale before any Real-facing data are available.

The completion-facing part is W, R, E. It reads finite windows, regular rational approximants, and the Real seal along the same route used by the public $\mathsf{CauchyCompletionReflector}^{\uparrow}$ surface.

Transport, replay, provenance, and local naming are the structural rows H, C, P, Q. They preserve the route but add no quotient completion, selected point, countable-choice schedule, open-cover compactness, or ambient Real completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3923.1, Definition 3923.2, and Definition 3923.3 seed the interval-completion carrier over located endpoint rows, nested refinements, dyadic shrinkage, finite windows, regular rational readback, Real seals, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind cuts, quotient completions, selected points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement classification, diameter-shrinkage stability, Real seal compatibility, and non-escape for BEDC.Derived.RealIntervalCompletionUp.

Constructive story: A $\mathsf{RealIntervalCompletion}^{\uparrow}$ packet is generated from LocatedInterval endpoint rows, NestedInterval refinement rows, DyadicRatCore radii, StreamName windows, RegSeqRat readbacks, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2947

A Concrete Naming Certificate for $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$

Following Bishop and Bridges constructive compact metric analysis, the $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ packet records the finite theorem-facing route by which a continuous image of compact metric data becomes totally bounded. It consumes the compact finite-net surface of chapter 60, the continuous-map graph and modulus rows of chapter 130, the source and target metric-distance rows, the uniform-modulus ledger, and the rational lower-bound fold. The packet is a concrete image-total-bounded carrier, not a semantic compact-image theorem in an ambient topology.

Definition 2947.1 (Compact continuous image total-bounded carrier)**.** *A* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, U, M, R, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric finite-net packet, exposing source centers and coverage rows through* $\mathsf{CompactMetric}^{\uparrow}$ *and the compact carrier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus and graph packet from the source metric carrier to the target metric carrier.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger read from the compact finite-net and continuous-map route.*
4. *M is the source and target* $\mathsf{Met}^{\uparrow}$ *distance row pair, carrying only displayed distance witnesses.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row used to select finite image radii from the displayed modulus and net data.*
6. *B is the* $\mathsf{ProbeBundle}$ *image-center row obtained by applying the displayed graph rows of F to the finite source-center bundle of X.*
7. *H records componentwise* hsame *transport for X, F, U, M, R, B and the local structural rows.*
8. *C records displayed* Cont *replay from compact source nets through continuous-map modulus rows, target distance rows, rational folds, and image-center bundle rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}}$ *row.*

The carrier contains no open-cover compactness object, no selected subcover from an arbitrary cover, no host image set, no quotient function equality, no choice-selected center, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CompactContinuousImageTotalBoundedUp.TasteGate.CompactContinuousImageTotalBounded`
`single carrier alignment`

Theorem 2947.2 (CompactContinuousImageTotalBounded NameCert obligations)**.** *For every accepted* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier* $K = (X, F, U, M, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for compact metric finite-net data, continuous-map rows, uniform-modulus ledger, metric-distance rows, rational lower-bound fold, image-center bundle, transport, replay, provenance, and local naming;*
2. *finite-net image classification, requiring every image center to arise from a bundled compact source center and a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row;*
3. *modulus compatibility, requiring the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger to use the same compact source, continuous-map, metric-distance, and* $\mathsf{Rat}^{\uparrow}$ *fold rows;*
4. *total-bounded export, requiring every target-image coverage read to factor through the finite* $\mathsf{ProbeBundle}$ *image-center row* B*;*
5. *refusal of open-cover compactness, arbitrary selected subcovers, host image sets, quotient function equality, choice-selected centers, and ambient completeness.*

Proof. Carrier admission is projection from Definition 3924.1. The tuple displays the compact source net, continuous graph and modulus data, uniform-modulus ledger, metric rows, rational fold, image centers, transport, replay, provenance, and local naming.

Finite-net image classification is the route X, F, B. Each image center is read by applying the displayed continuous-map graph row to a center already present in the compact source bundle, so no host image set or arbitrary center selector is introduced.

Modulus compatibility is the route X, F, U, M, R. The uniform-modulus ledger is obtained from the compact finite-net data, the continuous-map pointwise rows, the source and target metric distances, and the rational lower-bound fold over $\mathsf{Rat}^{\uparrow}$ histories.

The total-bounded export is B with its support rows. A target-image coverage read must pass through the finite $\mathsf{ProbeBundle}$ image-center row and the displayed target metric row before it reaches any consumer.

Finally, H, C, P, N preserve and name exactly this route. Since the carrier has no open-cover compactness object, arbitrary subcover selector, host image-set coordinate, quotient function equality, choice-selected center, or completeness coordinate, those objects cannot be exported. □

Theorem 2947.3 (Compact continuous image total-bounded handoff)**.** *Every total-boundedness consumer of an accepted* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier receives only the compact finite-net source row, the continuous-map graph and modulus rows, the uniform-modulus ledger, source and target metric-distance rows, the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold, the finite* $\mathsf{ProbeBundle}$ *image centers, and the visible structural rows*

$$X, F, U, M, R, B, H, C, P, N.$$

The handoff is therefore a finite BEDC certificate for image total-boundedness, not an open-cover compactness theorem and not a host image construction.

Proof. Begin with the accepted carrier and project its rows. The only source compactness data available are the finite-net rows in X.

Read the continuous map through F and the uniform modulus through U. Both rows are tied to the same source and target metric-distance rows M, so the image read cannot change its metric carrier mid-route.

The rational lower-bound fold R and the image-center bundle B produce the target finite-net witness. The fold is a finite $\mathsf{Rat}^{\uparrow}$ history calculation, and the image centers are a finite ProbeBundle list, so no arbitrary subcover or choice-selected image point is exposed.

The structural rows H, C, P, N transport, replay, prove provenance, and name that finite route. The excluded open-cover, host-image, quotient-function, choice, and completeness objects are absent from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$, seed)

Scope closed: Definition 3924.1, Definition 3924.2, and Definition 3924.3 seed the compact-continuous-image total-bounded carrier over compact finite nets, continuous-map modulus rows, uniform-modulus ledgers, metric distances, RatUp folds, ProbeBundle image centers, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary subcovers, host image sets, quotient function equality, choice-selected centers, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net image classification, modulus compatibility, total-bounded export, and non-escape for BEDC.Derived.CompactContinuousImageTotalBoundedUp.

Constructive story: A $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap pointwise-modulus rows, UniformModulus ledgers, source and target MetricSpace distance rows, RatUp lower-bound folds, ProbeBundle image centers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2948

A Concrete Naming Certificate for $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$

The $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ packet realizes the boundary-weighted disk-read branch of chapter 9463 as finite NameCert data. It sits beside the hyperbolic metric surface of chapter 3474, the boundary-shadow route of chapter 3813, the visual metric packet of chapter 3866, the unit-disk carrier of chapter 377, the circle carrier of chapter 58, the integral row of chapter 123, and the real-name seal of chapter 38. The packet names a finite boundary-kernel read, not a completed harmonic boundary theory.

Definition 2948.1 (Hyperbolic Poisson-kernel carrier)**.** *A* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-source row for the interior point being read.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row attached to the same disk source.*
3. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *boundary-shadow row that exposes the visible boundary window.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual-boundary comparison row.*
5. *U is the* $\mathsf{S}^{1\uparrow}$ *boundary-circle row over which angular samples are displayed.*
6. *S is the* $\mathsf{Integral}^{\uparrow}$ *boundary-integration row for the finite sample or displayed quadrature route.*
7. *I is the* $\mathsf{Real}^{\uparrow}$ *scalar endpoint row for kernel weights and normalization comparisons.*
8. *R is the radial denominator and numerator ledger tying D, M, B, U, I to the displayed hyperbolic Poisson weight.*
9. *W is the weighted boundary-integral readback row, exposing only the value determined by S, R, I over the displayed disk and boundary rows.*
10. *H records componentwise* hsame *transport for $D, M, B, V, U, S, I, R, W, C, P, N$.*
11. *C records displayed* Cont *replay from disk source through metric, boundary shadow, visual boundary, circle, integral, weight, and readback rows.*

12. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for completed boundary functions, unrestricted harmonic extension, selected radial limits, quotient boundary equality, ultrafilter convergence, or global boundary compactification.

Lean4 ✓ `BEDC.Derived.HyperbolicPoissonKernelUp.HyperbolicPoissonKernelTasteGate single carrier alignment`

Theorem 2948.2 (Hyperbolic Poisson-kernel NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier* $K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$ *surface consists exactly of disk-source admission, hyperbolic metric admission, boundary-shadow admission, visual-boundary comparison, circle-boundary admission, boundary-integral admission, Real endpoint admission, radial weight-ledger exactness, weighted readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3925.1. The disk, metric, boundary-shadow, visual-boundary, circle, integral, Real, weight, readback, transport, replay, provenance, and local naming rows are the full public inventory.

The geometric rows are read first. The disk row D fixes the interior source, M fixes the hyperbolic metric comparison, B fixes the visible boundary shadow, and V fixes the visual-boundary comparison.

The boundary-kernel rows are then read through U, S, I, R, W. The circle row supplies boundary samples, the integral row supplies the displayed boundary aggregation, the Real row supplies scalar endpoints, the weight ledger supplies the hyperbolic Poisson factor, and the readback row exposes only the value determined by those rows.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further row is present, completed boundary functions, unrestricted harmonic extension, selected limits, quotient boundary equality, ultrafilter convergence, and global compactification are outside the packet. □

Theorem 2948.3 (Hyperbolic Poisson-kernel boundary integral nonescape)**.** *Every accepted boundary-integral read of a* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicBoundaryShadow}^{\uparrow},\ \mathsf{HyperbolicVisualMetric}^{\uparrow},\ \mathsf{S}^{1\uparrow},\ \mathsf{Integral}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route first fixes the disk, metric, boundary-shadow, and visual-boundary rows, then reads the circle and integral rows, then reads the Real endpoint and weight ledger rows, and only then exposes weighted boundary readback. It exports no completed boundary function, unrestricted harmonic extension, selected radial limit, quotient boundary equality, ultrafilter convergence, or global compactification.

Proof. Let $K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$ be accepted. A boundary-integral consumer first receives the geometric route D, M, B, V, so the boundary window is the one displayed by the disk, metric, shadow, and visual rows.

The integration route is U, S. The circle row supplies the boundary parameter window, and the integral row supplies only the displayed finite or certified aggregation over that window.

The scalar kernel route is I, R, W. The Real row supplies the endpoints, the weight ledger ties those endpoints to the disk and boundary rows, and the readback row exposes the weighted value only after the integral and weight ledgers are fixed.

Transport, replay, provenance, and local naming through H, C, P, N preserve this route. The excluded analytic and choice-theoretic objects have no carrier coordinate, so the boundary-integral read cannot escape the finite packet. □

Falsifiable prediction: Removing either the $\mathsf{HyperbolicMetric}^{\uparrow}$ row M or the $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ row B makes Definition 3925.3 false: a consumer could no longer verify that the boundary weight belongs to the displayed hyperbolic disk route.

Independence witness: hyperbolicmetric, hyperbolic-boundary-shadow, unitdisk, sone, integral, real: $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk source, hyperbolic metric, boundary shadow, visual comparison, circle boundary, integral aggregation, Real endpoints, weight ledger, readback, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ HyperbolicPoissonKernel}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicPoissonKernel}^{\uparrow}$ instances. These instances keep the disk, metric, boundary, circle, integral, Real, weight, readback, transport, replay, provenance, and naming rows distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ packet is generated from UnitDiskUp disk data, HyperbolicMetricUp metric rows, HyperbolicBoundaryShadowUp and HyperbolicVisualMetricUp boundary rows, SOneUp circle rows, IntegralUp boundary aggregation, RealUp endpoints, a hyperbolic weight ledger, weighted readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3925.1, Definition 3925.2, and Definition 3925.3 seed the finite hyperbolic boundary-kernel route over disk, metric, boundary, circle, integral, Real, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary functions, unrestricted harmonic extension, selected radial limits, quotient boundary equality, ultrafilter convergence, global boundary compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, geometric boundary routing, circle and integral admission, weight-ledger exactness, readback nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPoissonKernelUp.

Chapter 2949

A Concrete Naming Certificate for $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$

Following Ekeland's variational principle and the Bishop-style use of complete-metric descent, the $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ packet names the finite descent ledger consumed by variational and fixed-point arguments. It sits downstream of the metric surface of chapter 57, the complete-metric carrier of chapter 128, the lower-semicontinuity packet of chapter 1536, the regular-rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet records a finite variational descent certificate; it is not a host minimizer, not a selected choice function, and not an ambient completeness theorem.

Definition 2949.1 (Ekeland variational-principle carrier)**.** *An* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, d, M, L, R, A, T, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite metric-domain row read through* $\mathsf{Met}^{\uparrow}$ *addresses.*
2. *d is the metric-comparison row naming the inspected distance witnesses on X.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *descent-readiness row, exposing only the finite Cauchy-tail obligations needed by the packet.*
4. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *value-stability row for the inspected functional.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row used to compare displayed values and descent budgets.*
6. *A is the approximation row naming the initial value, positive tolerance, and admissible descent budget.*
7. *T is the finite transfinite-free descent trace, ordered by the displayed metric and value-readback rows.*
8. *W is the winner-window row naming the final inspected point and the finite perturbation inequality visible at that point.*
9. *H records componentwise* hsame *transport for metric, completeness, lower-semicontinuity, rational readback, approximation, trace, and winner-window rows.*
10. *C records displayed* Cont *replay from a variational request through X, d, M, L, R, A, T, W.*

11. *P is the* Pkg *provenance over* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$ *row.*

The carrier exports exactly this finite descent packet. It contains no host-level minimizer, no global choice selector, no quotient of Cauchy approximants, and no independent assertion that the ambient metric space is complete.

Lean4 ✓ `BEDC.Derived.EkelandVariationalPrincipleUp.EkelandVariationalPrincipleTasteGate single carrier alignment`

Theorem 2949.2 (Ekeland variational-principle NameCert obligations)**.** *For every accepted* EkelandVariationalPrin *carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed metric-domain, distance, completeness, lower-semicontinuity, rational-readback, approximation, descent-trace, winner-window, transport, continuation, provenance, and naming rows;*
2. *metric readability, requiring every distance comparison used by the descent trace to be one of the displayed d rows on addresses in X;*
3. *complete-metric descent readiness, requiring every Cauchy-tail request made by the trace to pass through M;*
4. *lower-semicontinuity readability, requiring every value-stability read to pass through L and its regular-rational row R;*
5. *descent-budget coverage, requiring the initial value, tolerance, and perturbation budget to be named in A before the trace row T is consumed;*
6. *winner-window exposure, requiring the terminal inspected point and its finite perturbation inequality to be named in W;*
7. *transport, replay, provenance, and naming stability through H, C, P, N;*
8. *ledger non-escape, excluding host minimizers, global choice selectors, quotient-completion objects, and ambient completeness theorems.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier row by row. The metric-domain and distance rows expose the inspected addresses and comparisons, while M, L, R, A, T, W expose the complete-metric descent readiness, lower-semicontinuity reads, rational comparisons, approximation budget, finite trace, and terminal winner window.

The obligation order follows the displayed route. A descent comparison cannot be read before d and R are visible, a Cauchy-tail request cannot be read before M is visible, and the perturbation inequality cannot be read before the approximation row and trace row have been consumed.

Transport and replay are structural rows. They preserve the same metric, value, budget, trace, and winner-window data under hsame comparison and displayed Cont replay.

Finally, P and N source and name only the displayed packet. Since no coordinate stores a host minimizer, a global choice selector, a quotient-completion object, or an independent completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 2949.3 (Ekeland metric-descent handoff)**.** *Every variational, complete-metric, or fixed-point consumer reading an accepted* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *packet factors through the finite descent route*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto A, T, W, H, C, P, N.$$

The consumer therefore receives a certificate-local descent ledger and terminal perturbation window, not a host minimizer, selected choice function, quotient index, or ambient completeness theorem.

Proof. Start from an accepted carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$. The metric rows X, d supply the only addresses and distance comparisons available to the consumer.

The complete-metric and lower-semicontinuity rows then bind the analytic reads. The row M supplies the finite Cauchy-tail obligations used by descent, while L and R supply the value-stability and rational-readback route.

The approximation and trace rows A, T order the descent budget before the winner-window row W is consumed. Thus the terminal perturbation inequality is a displayed row of the packet, not a selected minimizer outside the packet.

The structural rows H, C, P, N transport, replay, source, and name this same route. Because no row carries a host choice selector, quotient completion, or independent ambient-completeness assertion, downstream consumers cannot obtain such data from the handoff. □

Theorem 2949.4 (Ekeland variational-principle descent window)**.** *Any accepted* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *packet exposes the approximation row A, finite descent trace T, complete-metric readiness row M, lower-semicontinuity row L, rational readback row R, and winner window W in that order before a downstream fixed-point or minimization consumer may read the perturbation inequality. The read is therefore a finite descent-window obligation over the carrier rows of Definition 3926.1, not a host minimizer, selected choice function, quotient completion, or ambient completeness principle.*

Proof. Project the accepted carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$ and read the obligation surface of Definition 3926.2. The approximation row A supplies the initial value, tolerance, and perturbation budget before any descent comparison can be inspected.

Next use the metric-descent handoff of Definition 3926.3. It keeps the finite trace T downstream of A and keeps every Cauchy-tail request routed through M.

The value comparisons then pass through L and R. Thus lower-semicontinuity and rational readback remain visible before the terminal perturbation inequality is exposed in W.

Finally, H, C, P, N transport, replay, source, and name the same ordered window. Since the carrier contains no coordinate for host minimizers, selected choice functions, quotient completions, or ambient completeness, the consumer receives only the displayed descent window. □

Closure status for $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$.

Constructive story: An $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ packet is generated from a finite metric-domain row, displayed distance witnesses, a complete-metric descent-readiness row, a lower-semicontinuity row, regular-rational readback, approximation budgets, a finite descent trace, a winner-window row, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3926.1, Definition 3926.2, and Definition 3926.3 seed the complete-metric variational descent ledger consumed by later fixed-point and minimization packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host minimizer, global choice selector, quotient completion, selected cofinal descent path, ambient completeness theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric descent readiness, lower-semicontinuity readback, descent-budget coverage, winner-window exposure, and consumer nonescape for BEDC.Derived.EkelandVariationalPr

Chapter 2950

A Concrete Naming Certificate for $\mathsf{KanExtension}^{\uparrow}$

Following Mac Lane's formulation of Kan extensions as universal arrows along a functor, the $\mathsf{KanExtension}^{\uparrow}$ packet reconstructs the usual left and right extension interface as finite BEDC certificate data. It depends on the certified category, functor, and natural-transformation surfaces of chapter 61, chapter 62, and chapter 63; it also reads the triangle-comparison discipline of chapter 107 and the cocone comparison rows of chapter 106. The chapter does not import an external functor category. It names the finite rows required for a displayed extension object, its comparison natural transformation, and its universal factorization route.

Definition 2950.1 (Kan extension carrier)**.** *A* $\mathsf{KanExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, D, E, J, F, L, \eta, U, V, H, R, P, N)$$

with the following displayed rows:

1. C, D, E *are* $\mathsf{Cat}^{\uparrow}$ *certificate rows for the source, target, and codomain categories;*
2. $J : C \to D$, $F : C \to E$, *and* $L : D \to E$ *are* $\mathsf{Functor}^{\uparrow}$ *rows whose object and morphism maps are already certified by the corresponding category classifiers;*
3. $\eta : F \Rightarrow L \circ J$ *is the displayed* $\mathsf{NatTrans}^{\uparrow}$ *comparison row;*
4. U *is the universal comparison row: for every displayed* $\mathsf{Functor}^{\uparrow}$ *row* $M : D \to E$ *and* $\mathsf{NatTrans}^{\uparrow}$ *row* $\alpha : F \Rightarrow M \circ J$, *it supplies the candidate mediating* $\mathsf{NatTrans}^{\uparrow}$ *row* $\overline{\alpha} : L \Rightarrow M$*;*
5. V *is the uniqueness row: two mediating rows with the same composite comparison are identified only by the displayed natural-transformation classifier;*
6. H *records componentwise* hsame *transport for categories, functors, comparison transformations, mediating transformations, and uniqueness witnesses;*
7. R *records displayed* Cont *replay from* α *to* $\overline{\alpha}$ *and from the composite* $\overline{\alpha} \circ J$ *back to* α*;*
8. P *records* Pkg *provenance over the* $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{Adjunction}^{\uparrow}$, $\mathsf{CatColimit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$ *row.*

The carrier has no coordinate for host functor equality, quotient natural-transformation equality, an external functor category, or an uncategorified universal property outside these displayed rows.

Lean4 ✓ `BEDC.Derived.KanExtensionUp.TasteGate.KanExtensionTasteGate single carrier alignment`

Theorem 2950.2 (Kan extension NameCert obligations)**.** *For the carrier of Definition 3927.1, the local* $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$ *surface consists exactly of the category rows* C, D, E*, functor rows* J, F, L*, comparison row* η*, universal comparison row* U*, uniqueness row* V*, componentwise transport row* H*, continuation replay row* R*, provenance row* P*, and local naming row* N*. Any accepted* $\mathsf{KanExtension}^{\uparrow}$ *read must expose those rows before it may be used as a left or right Kan-extension certificate.*

Lean4 ✓ `BEDC.Derived.KanExtensionUp.KanExtensionCarrier namecert obligations`

Proof. Project the carrier coordinates. The rows C, D, E license the three category classifiers, while J, F, L license the functor data and η supplies the displayed natural-transformation comparison.

Read the universal data next. The row U provides the mediating transformation for each displayed comparison α, and V constrains any two mediating rows by the $\mathsf{NatTrans}^{\uparrow}$ classifier rather than by host equality.

Finally attach the structural rows. The row H transports all displayed components by hsame, R replays the comparison route through Cont, P records the sibling certificate provenance, and N names the packet. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 2950.3 (Kan extension universal route)**.** *Every accepted* $\mathsf{KanExtension}^{\uparrow}$ *universal comparison factors through the displayed route*

$$(C, D, E, J, F, L, \eta), \alpha \longmapsto \overline{\alpha}, \overline{\alpha} \circ J \sim \alpha$$

using only U, V, H, R, P, N*. The* $\mathsf{Adjunction}^{\uparrow}$ *and* $\mathsf{CatColimit}^{\uparrow}$ *siblings may supply triangle-shaped and cocone-shaped comparison patterns, but neither sibling replaces the displayed* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *rows by an external universal-property assertion.*

Proof. Begin with a displayed comparison $\alpha : F \Rightarrow M \circ J$. By Definition 3927.1, the only row that can produce a mediator is U, and the mediator it returns is a $\mathsf{NatTrans}^{\uparrow}$ row $\overline{\alpha} : L \Rightarrow M$ over the already certified functors.

Use the uniqueness row to keep the route internal. The row V compares mediating transformations through the natural-transformation classifier, while H transports the category, functor, and transformation rows componentwise. This prevents the factorization from becoming a host-level equality claim.

The remaining rows preserve the same route. The continuation row R replays the composite comparison, P records its category-theory provenance through the cited siblings, and N names the packet. Hence the universal route is read as finite NameCert data, not as an untracked functor-category theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{KanExtension}^{\uparrow}$ packet is generated from category rows, functor rows, a comparison natural transformation, universal and uniqueness rows, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3927.1, Definition 3927.2, and Definition 3927.3 bind the displayed Kan-extension carrier, obligation surface, and universal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close external functor-category existence, quotient natural-transformation equality, arbitrary pointwise Kan-extension computation, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a Lean carrier theorem that binds the universal comparison rows to an explicit kernel scope.

Chapter 2951

A Concrete Naming Certificate for $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ packet presents bounded monotone sequence convergence as explicit dyadic-window and regular-Cauchy evidence, not as general $\mathsf{Real}^{\uparrow}$ completeness or choice. It sits beside the finite stream-name surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the bounded monotone seal boundary of chapter 918. The packet gives downstream Bolzano–Weierstrass, Cauchy completion, and sequence-limit consumers a finite L10 route for monotone convergence without importing ambient completeness.

Definition 2951.1 (MonotoneConvergenceSequence carrier)**.** *A* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, L, U, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the monotone sequence source row, carrying the displayed sequence observations and the finite monotonicity classifier read by the packet.*
2. *L and U are the lower and upper dyadic bound rows, giving the bounded interval in which every inspected source window is certified to land.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the initial and tail windows requested by the monotone-convergence consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic mesh widths, lower-upper bracket gaps, and tail budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the monotone windows and their dyadic tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source, bounds, finite windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for S, L, U, W, D, R, E and the structural rows.*
8. *C records displayed* Cont *replay from monotone source inspection through dyadic bounds, finite windows, tolerance shrinkage, regular readback, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$, $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, U, W, D, R, E, H, C, P, N$. It has no coordinate for an unrestricted supremum principle, quotient stream equality, host sequence equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneConvergenceSequenceUp.MonotoneConvergenceSequenceTasteGate single carrier alignment`

Theorem 2951.2 (MonotoneConvergenceSequence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier*

$$M = (S, L, U, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$ *surface consists exactly of monotone source exposure through S, lower and upper dyadic bound exposure through L, U, finite window exposure through W, dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3928.1. The tuple displays the monotone source row, dyadic bound rows, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, and the structural transport, replay, provenance, and naming rows.

The bounded monotone evidence is confined to S, L, U, W, D. The source row gives the monotonicity classifier, the bound rows give the visible interval enclosure, and the window and tolerance rows record the finite bracket shrinkage that a consumer may inspect.

The analytic handoff is then R, E. The regular rational readback row is obtained only from the displayed windows and dyadic ledger, and the Real seal row is exposed only after that readback has been displayed.

Finally H, C, P, N preserve and name the same finite route. Since the carrier contains no supremum, quotient stream, host sequence equality, host real equality, choice, or ambient completeness coordinate, those objects are outside the local NameCert surface. □

Theorem 2951.3 (MonotoneConvergenceSequence regular Cauchy route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The bounded monotone seal supplies the source and interval rows, the finite stream windows expose the inspected observations, the dyadic ledger supplies the tail tolerance budget, and the regular rational readback precedes the terminal Real seal.

Proof. Start with an accepted carrier $M = (S, L, U, W, D, R, E, H, C, P, N)$. The monotone source and bound rows S, L, U are the only coordinates through which bounded monotone evidence can enter the packet.

Read the bounded seal boundary through chapter 918. It supplies the public source-and-bound discipline consumed by this carrier, but it does not provide an unrestricted completeness principle.

Next read the finite observation route. The windows W are $\mathsf{StreamName}^{\uparrow}$ data, the tolerance ledger D is $\mathsf{DyadicRatCore}^{\uparrow}$ data, and the regular rational row R repacks exactly those displayed windows and tolerances as $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The Real seal E is therefore downstream of the source, dyadic bounds, finite windows, dyadic tolerance ledger, and regular rational readback. Transport and replay through H, C, P, N preserve that same route, so no consumer receives a hidden supremum, quotient stream equality, host sequence equality, host real equality, countable-choice schedule, or ambient Real-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ packet is generated from a monotone source row, lower and upper dyadic bound rows, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3928.1, Definition 3928.2, and Definition 3928.3 seed the bounded monotone regular-Cauchy route over StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a general supremum principle, quotient stream equality, host sequence equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone seal handoff, dyadic bracket shrinkage, RegSeqRat readback exactness, RealUp seal nonescape, and completeness-refusal proofs for BEDC.Derived.MonotoneConvergenceS

Chapter 2952

A Concrete Naming Certificate for $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ packet records the finite route by which a displayed Picard orbit is read as a regular rational Cauchy name and then sealed as a $\mathsf{Real}^{\uparrow}$ object. It sits over the orbit rows of chapter 2359, the contraction ledger of chapter 396, the precision schedule of chapter 292, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the bounded ambient interface of chapter 128. The packet is deliberately finite: it certifies the route from named Picard rows to a regular Cauchy seal, not an unrestricted fixed-point selection theorem.

Definition 2952.1 (Regular Cauchy Picard seal carrier)**.** *A* $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, B, M, Q, E, H, C, P, N)$$

whose rows have the following roles.

1. *I is a finite* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row displaying the initial point, the iterated contraction rows, and the finite prefix consumed by the certificate.*
2. *B is a* $\mathsf{BanachFixedPoint}^{\uparrow}$ *contraction-budget row exposing only the local Lipschitz decrease data needed to compare successive Picard rows.*
3. *M is a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule extracted from the displayed contraction budget and attached to finite precision requests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite Picard orbit and the modulus schedule.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached through the displayed regular rational readback.*
6. *H, C, P, N are respectively the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a selected fixed point, host function extensionality, quotient stream equality, ambient completeness, or an unrestricted Banach fixed-point existence principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyPicardSealUp.RegularCauchyPicardSealTasteGate single carrier alignment`

Theorem 2952.2 (Regular Cauchy Picard seal obligations)**.** *For every accepted carrier $R = (I, B, M, Q, E, H, C, P, N)$ of Definition 3929.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$ *surface*

consists exactly of the Picard orbit row I, the contraction-budget row B, the extracted modulus row M, the regular rational readback row Q, the terminal $\mathsf{Real}^{\uparrow}$ *seal row E, and the structural rows H, C, P, N. A downstream consumer must read the rows in the displayed order*

$$I \to B \to M \to Q \to E \to H, C, P, N.$$

No accepted obligation row can be replaced by a selected fixed point, a host-level proof that two functions are extensionally equal, quotient equality between stream representatives, an ambient completeness theorem, or an unrestricted Banach fixed-point existence row.

Proof. Project the carrier of Definition 3929.1. The analytic rows are exactly the finite Picard orbit, contraction budget, extracted modulus, regular rational readback, and terminal Real seal; the remaining rows are the standard BEDC transport, replay, provenance, and naming surface.

The Picard row is read first because the modulus schedule is extracted from finite comparisons between displayed iterates. The contraction-budget row supplies those comparisons, and the $\mathsf{CauchyModulus}^{\uparrow}$ row records the precision threshold as data rather than as a host completeness argument.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback can then consume only the displayed orbit prefix and the extracted modulus. Once that readback is present, the $\mathsf{Real}^{\uparrow}$ seal is reached through the same regular Cauchy route used by the surrounding Real-and-Completion chapters.

Finally the rows H, C, P, N transport and replay this finite route as the local certificate. Since none of the carrier coordinates stores a selected fixed point, function extensionality principle, quotient stream equality, ambient completeness theorem, or unrestricted fixed-point existence theorem, no consumer of the NameCert surface can obtain those claims from this packet. □

Theorem 2952.3 (Regular Cauchy Picard seal route)**.** *Let $R = (I, B, M, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ *carrier. Every finite precision request for the Real-facing Picard limit factors through the route*

$$\mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{BanachFixedPoint}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first reads a finite Picard orbit prefix, then extracts a Cauchy modulus from the contraction budget, then performs $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It does not claim a chosen fixed point, host function extensionality, quotient stream equality, ambient completeness, or unrestricted Banach fixed-point existence.*

Proof. Begin with the accepted carrier and the obligation theorem above. They force every precision request to start with the displayed finite Picard orbit I and the contraction budget B.

Next extract the modulus row M. This step is finite: a requested precision is matched with a displayed threshold from the contraction ledger, so the certificate does not invoke a global completeness principle or a choice schedule over all requests.

Use the extracted threshold to read the regular rational row Q. The readback is the ordinary $\mathsf{RegSeqRat}^{\uparrow}$ passage from finite rational data and a modulus to a regular Cauchy name; it has no quotient-stream coordinate and no host equality coordinate.

Finally pass from Q to the terminal Real seal E and replay the route through H, C, P, N. The resulting seal records the finite Picard-to-regular-Cauchy route, while the selected fixed point and unrestricted Banach existence claims remain outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$.

Constructive story: A RegularCauchyPicardSeal$^{\uparrow}$ packet is generated from a finite PicardIteration$^{\uparrow}$ orbit prefix, a BanachFixedPoint$^{\uparrow}$ contraction-budget row, an extracted CauchyModulus$^{\uparrow}$ precision schedule, a RegSeqRat$^{\uparrow}$ readback row, a terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyPicardSeal}^{\uparrow}}$, seed)

Scope closed: Definition 3929.1, Definition 3929.2, and Definition 3929.3 seed the finite Picard-orbit route from contraction data through modulus extraction, regular rational readback, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyPicardSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host function extensionality, quotient stream equality, ambient completeness, or unrestricted Banach fixed-point existence.

Upgrade path: A formal route can encode the finite carrier rows and prove the Picard-orbit modulus extraction, regular rational readback, and terminal Real-seal route.

Chapter 2953

A Concrete Naming Certificate for ArzelaAscoliSelection$^{\uparrow}$

Following Arzela–Ascoli, Kelley, and Bishop–Bridges constructive analysis, the ArzelaAscoliSelection$^{\uparrow}$ packet records the finite selection surface by which an equicontinuous bounded family on a compact metric source exposes a regular convergent subsequence. It sits after the family ledger of chapter 413, the classical compactness packet of chapter 412, the bounded-family surface of chapter 3522, and the diagonal extraction route of chapter 2219. The packet names the selection route through displayed finite rows only; it does not export a host compact-open topology, an arbitrary sequence choice, quotient equality of functions, or ambient completeness outside the named certificates.

Definition 2953.1 (ArzelaAscoliSelection carrier)**.** *An* ArzelaAscoliSelection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, E, B, D, M, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* CompactMetric$^{\uparrow}$ *source row supplying finite net and compact inspection data.*
2. *E is the* EquicontinuityFamily$^{\uparrow}$ *ledger, including shared-center modulus rows for the displayed family.*
3. *B is the* BoundedFunctionFamily$^{\uparrow}$ *row, recording pointwise and sup-norm boundedness for the same family.*
4. *D is the* AscoliDiagonal$^{\uparrow}$ *row selecting compatible finite windows across the source net and target readback.*
5. *M is the finite modulus row read through* ArzelaAscoliFiniteModulus$^{\uparrow}$ *and the compact uniform-continuity route.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the selected finite windows.*
7. *R is the* Real$^{\uparrow}$ *seal row attached to the regular readback of the selected subsequence values.*
8. *H records componentwise* hsame *transport for X, E, B, D, M, Q, R.*
9. *C records displayed* Cont *replay from compact-source inspection through equicontinuity, boundedness, diagonal selection, regular readback, and real sealing.*

10. *P is the* Pkg *provenance over* ArzelaAscoliSelection$^{\uparrow}$, CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, BoundedFunctionFamily$^{\uparrow}$, AscoliDiagonal$^{\uparrow}$, ArzelaAscoliFiniteModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ArzelaAscoliSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-selected subsequence, a compact-open completion, quotient equality of functions, countable choice, or an unstated completeness principle.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliSelectionUp.TasteGate.ArzelaAscoliSelectionTasteGate single carrier alignment`

Theorem 2953.2 (ArzelaAscoliSelection NameCert obligations)**.** *For every accepted* ArzelaAscoliSelection$^{\uparrow}$ *carrier $A = (X, E, B, D, M, Q, R, H, C, P, N)$, the local* NameCert$_{\text{ArzelaAscoliSelection}^{\uparrow}}$ *surface consists exactly of compact-source admission through* CompactMetric$^{\uparrow}$, *equicontinuity-family admission through* EquicontinuityFamily$^{\uparrow}$, *bounded-family admission through* BoundedFunctionFamily$^{\uparrow}$, *diagonal-window admission through* AscoliDiagonal$^{\uparrow}$, *finite modulus readback through* ArzelaAscoliFiniteModulus$^{\uparrow}$, *regular rational readback through* RegSeqRat$^{\uparrow}$, *terminal real sealing through* Real$^{\uparrow}$, *carrier storage in* BHist, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* NameCert.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliSelectionUp.ArzelaAscoliSelectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3930.1. The rows $X, E, B, D, M, Q, R, H, C, P, N$ are the full displayed inventory.

Read the source and family rows first. The compact metric row X supplies finite source nets, the equicontinuity row E supplies shared modulus ledgers, and the bounded-family row B supplies the bound surface needed before any diagonal selection is public.

Read the selection rows next. The diagonal row D consumes the finite net and family ledgers, while M records the finite modulus compatibility that keeps the selected windows inside the Arzela–Ascoli route. The row Q then exposes the selected windows as regular rational readback, and R seals the resulting values as Real$^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier has no coordinate for host subsequence choice, compact-open completion, quotient equality of functions, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2953.3 (ArzelaAscoliSelection selected regular subsequence route)**.** *Every accepted* ArzelaAscoliSelection$^{\uparrow}$ *packet exposes a selected regular subsequence only through the finite route*

CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, BoundedFunctionFamily$^{\uparrow}$, AscoliDiagonal$^{\uparrow}$, ArzelaAscoliFiniteModulus$^{\uparrow}$, Reg

The route first displays compact source nets, family equicontinuity, boundedness, and diagonal finite windows; only then does it expose regular rational readback and the real seal. It exports no arbitrary subsequence selector, compact-open host topology, quotient function equality, countable choice, or ambient completeness theorem.

Proof. Use Definition 3930.2. It orders the compact source, equicontinuity, bounded-family, diagonal, finite-modulus, regular-readback, and Real-seal rows inside one accepted carrier.

The diagonal row can be consumed only after X, E, B are displayed. The finite modulus row M ties the selected windows back to the shared equicontinuity ledger and the bounded-family surface, so the selection does not detach from the family certificate.

The readback rows Q, R are downstream. They convert the displayed windows into $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$ data, while H, C, P, N preserve the same route.

Thus the packet supplies a finite BEDC selection route. None of the refused host-level objects is present in the carrier inventory. □

Falsifiable prediction: An accepted $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ consumer that reads a selected regular subsequence without first exposing CompactMetric, EquicontinuityFamily, BoundedFunctionFamily, AscoliDiagonal, finite modulus, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows refutes the carrier surface.

Independence witness: compactmetric, equicontinuityfamily, bounded-function-family, ascoli-diagonal, arzelaascoli: $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-source nets, equicontinuity ledgers, bounded-family rows, diagonal selection windows, finite modulus readback, regular rational data, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ packet is generated from CompactMetricUp source nets, EquicontinuityFamilyUp modulus ledgers, BoundedFunctionFamilyUp boundedness rows, AscoliDiagonalUp finite windows, ArzelaAscoliFiniteModulusUp compatibility rows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3930.1, Definition 3930.2, and Definition 3930.3 seed the finite constructive selection route for equicontinuous bounded families over compact metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, compact-open topology, arbitrary subsequence choice, quotient equality of functions, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, family ledger exactness, boundedness readback, diagonal-window compatibility, finite modulus transport, regular readback, Real sealing, provenance, and local naming for BEDC.Derived.ArzelaAscoliSelectionUp.

Chapter 2954

A Concrete Naming Certificate for $\mathsf{ContractionMappingTheorem}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive metric analysis, the $\mathsf{ContractionMappingTheorem}^{\uparrow}$ packet records the fixed-point theorem as a finite complete-metric and Picard-orbit certificate. It sits next to the complete-metric carrier of chapter 128, the metric-distance surface of chapter 57, the contractive-map row of chapter 3819, the completion boundary of chapter 95, and the terminal real seal of chapter 38. The packet names the theorem-facing route from a displayed contraction to a unique fixed-point seal; it is not a host fixed-point theorem, a quotient completion principle, a selected representative, or an ambient completeness axiom.

Definition 2954.1 (Contraction mapping theorem carrier)**.** *A* $\mathsf{ContractionMappingTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, M, D, A, O, L, U, H, T, P, N)$$

with the following visible rows:

1. *C is the complete-metric source row, carrying the displayed Cauchy admission and endpoint-read interface from* $\mathsf{CompleteMetric}^{\uparrow}$*.*
2. *M is the metric-distance row, exposing only the finite distance comparisons admitted by the* $\mathsf{Met}^{\uparrow}$ *surface.*
3. *D is the* $\mathsf{ContractiveMap}^{\uparrow}$ *row, recording the self-map graph, contraction constant ledger, and one-step distance decrease over the same metric source.*
4. *A is the Picard/Cauchy orbit schedule, listing the finite initial point, iterate windows, residual-distance bounds, and Cauchy-modulus handoff generated from D.*
5. *O is the* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row that admits the orbit endpoint only after C, M, D, A have been displayed.*
6. *L is the fixed-point uniqueness ledger, comparing two endpoint reads through the same metric-distance row and the contraction decrease route before local equality is exported.*
7. *U records the fixed-point equation read, requiring the endpoint seal, the contractive-map graph, and the Picard orbit replay to agree componentwise.*
8. *H stores componentwise* hsame *transport for the complete-metric source, metric-distance row, contractive-map row, Picard/Cauchy schedule, completion or real seal, fixed-point uniqueness ledger, fixed-point equation row, replay, provenance, and local naming.*

9. *T stores displayed* Cont *replay from contraction admission through Picard iteration, Cauchy endpoint admission, fixed-point equation read, and uniqueness comparison.*
10. *P is the* Pkg *provenance over* ContractionMappingTheorem$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Met$^{\uparrow}$, ContractiveMap$^{\uparrow}$, Completion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ *row.*

The classifier accepts packets only when the complete-metric row, metric-distance row, contractive-map row, Picard/Cauchy schedule, endpoint seal, fixed-point equation row, uniqueness ledger, transport, replay, provenance, and local naming row are all read on the same displayed route.

Theorem 2954.2 (Contraction mapping theorem NameCert obligations)**.** *For every accepted* ContractionMappingTheorem$^{\uparrow}$ *carrier*

$$F = (C, M, D, A, O, L, U, H, T, P, N),$$

the local NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ *obligations are exactly complete-metric source admission, metric-distance visibility, contractive-map row admission, Picard/Cauchy orbit scheduling,* Completion$^{\uparrow}$ *or* Real$^{\uparrow}$ *endpoint sealing, fixed-point equation replay, fixed-point uniqueness comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ContractionMappingTheoremUp.ContractionMappingTheoremTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 3931.1. The source rows C and M are the only complete-metric and metric-distance data available to a theorem-facing consumer, and D is the only row that carries contraction information.

The Picard/Cauchy schedule A is generated from the displayed contraction row. Its residual bounds and Cauchy-modulus handoff must be read before the endpoint seal O is visible, so the completion or real seal cannot be consumed as an ambient completeness principle.

The fixed-point equation row U and uniqueness ledger L are downstream rows. They compare only endpoint reads that have traversed C, M, D, A, O and then preserve that route through the structural rows H, T, P, N.

Thus the NameCert obligations are precisely the displayed finite route. No carrier coordinate supplies a host fixed-point theorem, quotient completion equality, selected representative, unrestricted choice, trichotomy, or open-cover compactness. □

Falsifiable prediction: Every accepted ContractionMappingTheorem$^{\uparrow}$ consumer reads a complete-metric source, a metric-distance row, a contractive-map row, a Picard/Cauchy orbit schedule, a completion or real endpoint seal, a fixed-point equation row, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before it treats the packet as a fixed-point theorem.

Independence witness: completemetric, metric, contractive-map, completion, real: ContractionMappingTheorem binds the theorem-facing fixed-point route in one NameCert packet; the listed siblings expose source, distance, contraction, endpoint, or seal rows separately but do not jointly carry Picard/Cauchy scheduling, fixed-point equation replay, uniqueness, transport, replay, provenance, and local naming for the theorem surface.

Closure status for NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ **·**

Constructive story: A ContractionMappingTheorem$^{\uparrow}$ packet is generated from a CompleteMetric source, a Metric distance row, a ContractiveMap row, a Picard/Cauchy orbit schedule, a CompletionUp or RealUp seal, a fixed-point

equation row, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ContractionMappingTheorem}\uparrow}$, seed)

Scope closed: Definition 3931.1 and Definition 3931.2 seed the finite fixed-point theorem certificate over complete-metric, metric-distance, contractive-map, Picard/Cauchy, completion or real seal, uniqueness, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ContractionMappingTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host fixed-point equality, quotient completion equality, selected representatives, unrestricted choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contractive-map visibility, Picard/Cauchy scheduling, endpoint sealing, fixed-point equation replay, uniqueness routing, transport, replay, provenance, and local naming for BEDC.Derived.ContractionMappingTheoremUp.

Chapter 2955

A Concrete Naming Certificate for HilbertAlexanderBlocker$^{\uparrow}$

The HilbertAlexanderBlocker$^{\uparrow}$ packet realizes the Hilbert-series blocker complex of Definition 9462.1622 as finite BEDC NameCert data. It sits after the simplicial-complex carrier of chapter 238, the polynomial carrier of chapter 50, the finite-trace evaluator surface of chapter 818, the DNA boundary-tile packet of chapter 3393, and the arity-ten BHist tuple capstone of chapter 1364. The packet names blocker-cover, face-polynomial, Hilbert-row, Alexander-dual, and non-Cohen–Macaulay rows as displayed finite data; it is not a host Stanley–Reisner algebra oracle, a biological stochastic model, quotient codon semantics, or an unrestricted homology computation.

Definition 2955.1 (HilbertAlexanderBlocker carrier)**.** *A* HilbertAlexanderBlocker$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, P, T, B, F, L, A, R, C, Q, N)$$

with the following displayed rows:

1. *S is the* SimplicialComplex$^{\uparrow}$ *row for the blocker complex and its four omitted-coordinate cover faces.*
2. *P is the* Poly$^{\uparrow}$ *row recording the face polynomial and the Hilbert numerator polynomial.*
3. *T is the* FiniteTraceEvaluator$^{\uparrow}$ *row that evaluates the finite inclusion-exclusion table for cover intersections.*
4. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *row carrying the six-window boundary source and trigger coordinates.*
5. *F is the blocker-cover row, recording the four facets and the intersection multiplicities $4, 6, 4, 1$.*
6. *L is the Hilbert-row ledger, recording the rational-function denominator powers $14, 6, 2, 0$ and the signed inclusion-exclusion coefficients.*
7. *A is the Alexander-dual row, recording the four generator complements and their lcm-degree ledger.*
8. *R records componentwise* hsame *transport for S, P, T, B, F, L, A.*
9. *C records displayed* Cont *replay from boundary-tile reads through blocker cover, face polynomial, Hilbert rows, and Alexander-dual rows.*

10. *Q is the* Pkg *provenance over* HilbertAlexanderBlocker$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, Poly$^{\uparrow}$, FiniteTraceEvaluator$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, BHistDecaTupleNameCert$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient polynomial-ring equality oracle, an unrestricted minimal resolution, stochastic biological semantics, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision procedure.

Theorem 2955.2 (HilbertAlexanderBlocker NameCert obligations)**.** *For every accepted* HilbertAlexanderBlocker$^{\uparrow}$ *carrier $H = (S, P, T, B, F, L, A, R, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$ *surface consists exactly of simplicial-complex admission through S, polynomial-row admission through P, finite trace evaluation through T, DNA boundary-tile source admission through B, blocker-cover exactness through F, Hilbert-row readback through L, Alexander-dual ledger exposure through A, componentwise* hsame *transport through R, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.HilbertAlexanderBlockerUp.HilbertAlexanderBlockerNameCertObligations`

Proof. Project the carrier of Definition 3932.1. The rows $S, P, T, B, F, L, A, R, C, Q, N$ are the full accepted inventory.

The blocker-cover side is finite. The row B supplies the boundary-tile source, S supplies the simplicial-complex carrier, and F records the four cover facets together with their intersection multiplicities.

The Hilbert side is finite as well. The row T evaluates the inclusion-exclusion table, P stores the face and numerator polynomials, and L stores the denominator powers and signs of the Hilbert-series row.

The Alexander-dual row A records the generator-complement ledger attached to the same blocker cover. The structural rows R, C, Q, N transport, replay, source, and name exactly this packet, so no accepted NameCert obligation can project an ambient polynomial-ring oracle, unrestricted resolution, stochastic model, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision. □

Theorem 2955.3 (HilbertAlexanderBlocker finite Hilbert row)**.** *Every Hilbert-series read from an accepted* HilbertAlexanderBlocker$^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{SimplicialComplex}^{\uparrow},\ \mathsf{Poly}^{\uparrow},\ \mathsf{FiniteTraceEvaluator}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{BHistDecaTupleNameCert}^{\uparrow} \longmapsto F,$$

The read first displays the blocker cover and intersection table, then evaluates the face-polynomial and Hilbert-row ledger, and only then exposes Alexander-dual and local naming rows. It exports no ambient Stanley–Reisner algebra oracle, unrestricted minimal resolution, quotient codon equality, hidden homology computation, or biological stochastic model.

Lean4 ✓ `BEDC.Derived.HilbertAlexanderBlockerUp.HilbertAlexanderBlockerFiniteHilbertRow`

Proof. Apply Definition 3932.2. It places the simplicial-complex, polynomial, trace-evaluation, and boundary-tile sources before the blocker-cover and Hilbert-row ledgers.

The finite trace row T evaluates only the displayed intersection table F. The polynomial row P and Hilbert ledger L therefore read the same finite cover data rather than a host polynomial-ring quotient.

The Alexander-dual row A is downstream of the same cover. Transport, replay, provenance, and local naming through R, C, Q, N preserve that route, so the Hilbert-series read remains a finite NameCert packet. □

Falsifiable prediction: Changing the blocker-cover multiplicity row F changes the Hilbert denominator-power ledger L before any accepted $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ consumer can preserve the displayed Hilbert series.

Independence witness: simplicialcomplex, polynomial, finite-trace-evaluator, dna-window-six-boundary-tile, b-hist-deca-tuple-name-cert: $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries blocker-cover rows, face-polynomial rows, Hilbert-series rows, Alexander-dual rows, finite trace evaluation, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$.

Constructive story: A $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ packet is generated from a SimplicialComplexUp blocker complex, PolynomialUp face and numerator rows, FiniteTraceEvaluatorUp inclusion-exclusion evaluation, DnaWindowSixBoundaryTileUp source rows, blocker-cover data, Hilbert-row ledgers, Alexander-dual ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3932.1, Definition 3932.2, and Definition 3932.3 seed the finite Hilbert-series blocker packet over cover, polynomial, trace-evaluation, boundary-tile, and tuple-name rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Stanley–Reisner algebra equality, unrestricted minimal resolutions, stochastic biological semantics, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision.

Upgrade path: Obligation closure requires checked carrier admission, blocker-cover exactness, face-polynomial readback, Hilbert-row ledger exactness, Alexander-dual ledger exposure, trace-evaluation stability, transport, replay, provenance, and local naming for BEDC.Derived.HilbertAlexanderBlockerUp.

Chapter 2956

A Concrete Naming Certificate for $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface used when a compact interval argument must choose a displayed located subinterval before completion-facing consumers read a real name. It sits over the located interval carrier of chapter 394, the closed bounded interval source of chapter 1591, the nested interval route of chapter 320, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet is a NameCert surface for a displayed compact-interval selection route; it is not open-cover compactness, selected points, trichotomy, quotient $\mathsf{Real}^{\uparrow}$ equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2956.1 (Located compact interval selection carrier)**.** *A* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, B, N, D, S, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing the lower and upper located endpoint windows and the rational containment ledger.*
2. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *compact-bound row, exposing the closed bounded source used by the selection route.*
3. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row, recording the finite nested subintervals retained by the selection procedure.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for cell width, endpoint separation, and retained-refinement budgets.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint and retained-interval observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected located interval windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after I, B, N, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for I, B, N, D, S, R, E and the local packet shape.*
9. *C records displayed* Cont *replay from located interval source and compact-bound rows through nested refinements, dyadic tolerances, windows, readback, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$ *row.*

The carrier exports no open-cover compactness theorem, selected point of the interval, trichotomy principle, quotient equality of real names, or ambient real-completeness row.

Lean4 ✓ `BEDC.Derived.LocatedCompactIntervalSelectionUp.LocatedCompactIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 2956.2 (Located compact interval selection NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier*

$$L = (I, B, N, D, S, R, E, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$ *surface consists exactly of located interval exposure through I, closed bounded interval exposure through B, nested interval refinement through N, dyadic tolerance accounting through D, stream-window exposure through S, regular rational readback through R, real-name sealing through E, componentwise transport through H, replay through C, provenance through P, and local naming through Q.*

Proof. Project the carrier of Definition 3933.1. The finite packet exposes exactly the located interval source, compact-bound source, nested refinement ledger, dyadic tolerance row, stream windows, regular rational readback, Real seal, and structural rows.

Read the interval rows first. The located row I supplies endpoint windows and rational containment, while the closed bounded row B supplies the compact interval source used by the selection route.

The refinement and readback rows are downstream of those sources. The nested ledger N records retained subintervals, D records the dyadic budgets attached to those refinements, and S, R, E expose the selected interval windows only through stream, regular rational, and Real rows.

Finally H, C, P, Q transport, replay, package, and name the same route. Since no other coordinate exists, the listed rows exhaust the local NameCert obligations and cannot import open-cover compactness, selected points, trichotomy, quotient real equality, or ambient Real completeness. □

Theorem 2956.3 (Located compact interval selection real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The selected located interval is therefore handed off as displayed endpoint windows, nested subinterval rows, dyadic tolerance budgets, stream windows, regular rational readbacks, and a terminal Real seal, not as a selected point, quotient real representative, trichotomy comparison, open-cover compactness argument, or ambient completeness principle.

Proof. Begin with an accepted carrier $L = (I, B, N, D, S, R, E, H, C, P, Q)$. The source rows I and B are the only located and compact interval coordinates available to the packet.

Next inspect the nested and dyadic rows. The row N records the retained finite subintervals and D records the tolerances that make those retained intervals readable without a point selector.

The real-facing readback then passes through S, R, E. Stream windows expose the finite endpoint observations, regular rational rows read them as regular data, and the Real seal is downstream of those displayed rows.

Transport and replay through H, C, P, Q preserve that same route for completion-facing consumers. The packet has no coordinate for open-cover compactness, selected points, trichotomy, quotient Real equality, or ambient Real completeness, so none of those principles can be exported by the handoff. □

Falsifiable prediction: A compact-interval consumer that omits the located interval row, the closed bounded interval row, or the nested refinement row cannot claim a $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ selection handoff.

Independence witness: locatedinterval, closedboundedinterval, nestedinterval, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source interval rows, compact-bound rows, nested refinements, dyadic budgets, stream windows, regular readbacks, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ packet is generated from a LocatedInterval source row, a ClosedBoundedInterval compact-bound row, a NestedInterval refinement row, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3933.1, Definition 3933.2, and Definition 3933.3 seed the compact interval selection route over located interval, closed bounded interval, nested interval, dyadic, stream, regular rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected points, trichotomy, quotient $\mathsf{Real}^{\uparrow}$ equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval source exposure, closed-bounded source exposure, nested refinement routing, dyadic tolerance accounting, stream readback, RegSeqRat exactness, RealUp seal nonescape, and selection-refusal proofs for BEDC.Derived.LocatedCompactIntervalSelectionUp.

Chapter 2957

A Concrete Naming Certificate for CompactUniformRadiusLedger$^{\uparrow}$

Following Lebesgue and Bishop–Bridges, the CompactUniformRadiusLedger$^{\uparrow}$ packet records the finite positive-radius ledger used in the Heine–Cantor route of chapter 9237. It sits between the compact finite-net surface of chapter 129, the pointwise continuity surface of chapter 130, the positive Lebesgue-radius packet of chapter 452, and the downstream UniformModulus$^{\uparrow}$ consumer of chapter 369. The packet names the radius choices and their finite lower-bound fold as displayed BEDC rows; it does not export an open-cover compactness theorem, a selected subsequence, a host minimum over radii, quotient equality of reals, or an external analytic modulus.

Definition 2957.1 (Compact uniform radius ledger carrier)**.** *A* CompactUniformRadiusLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, G, F, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, carrying the finite* ProbeBundle *of centers and the source-distance coverage rows.*
2. *M is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus ledger for the same source and target metrics.*
3. *G is the* FiniteLebesgueNumber$^{\uparrow}$ *positive-radius row, assigning a positive radius below the cover radius attached to each displayed center.*
4. *F is the* Rat$^{\uparrow}$ *lower-bound fold row, produced by primitive recursion over the finite radius bundle.*
5. *R records the radius-selection compatibility row saying that the folded lower bound is below every positive radius used by G.*
6. *U is the* UniformModulus$^{\uparrow}$ *output row that may be read only after K, M, G, F, R have been displayed.*
7. *H records componentwise* hsame *transport for the compact-net, pointwise-modulus, Lebesgue-radius, rational-fold, compatibility, and output rows.*
8. *C records displayed* Cont *replay from finite-net inspection through pointwise-modulus reads, positive-radius extraction, rational folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* CompactUniformRadiusLedger$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, Rat$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through these rows. No coordinate records open-cover compactness, selected subsequences, external choice of a minimum radius, host real equality, or a semantic modulus function outside the displayed $\mathsf{UniformModulus}^{\uparrow}$ *output.*

Lean4 ✓ `BEDC.Derived.CompactUniformRadiusLedgerUp.CompactUniformRadiusLedgerTasteGate single carrier alignment`

Theorem 2957.2 (Compact uniform radius ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ *carrier* $L = (K, M, G, F, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedge}}$ *surface consists exactly of compact-net admission through* K*, pointwise-modulus admission through* M*, positive-radius exposure through* G*, finite rational lower-bound folding through* F*, compatibility exposure through* R*,* $\mathsf{UniformModulus}^{\uparrow}$ *output through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3934.1. The rows $K, M, G, F, R, U, H, C, P, N$ are the complete displayed inventory.

Read the analytic input rows first. The compact-metric row K supplies the finite net and its coverage witnesses, while M supplies the pointwise-modulus rows at the displayed centers. The positive-radius row G ties these local moduli to a finite Lebesgue-radius ledger.

Next read the finite fold. The $\mathsf{Rat}^{\uparrow}$ row F compares the positive radii by primitive recursion over the finite bundle, and R records that the resulting lower bound is below each radius used by G. These rows are BEDC histories, not a host minimum.

Finally U, H, C, P, N export, transport, replay, source, and name exactly this ledger. Since the carrier has no coordinate for open-cover compactness, selected subsequences, external minimum selection, host real equality, or a semantic modulus function, the listed rows exhaust the local NameCert obligations. □

Theorem 2957.3 (Compact uniform radius ledger Heine–Cantor handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ *packet reads the Heine–Cantor radius choice through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The handoff first displays the finite net, then the pointwise-modulus and positive-radius rows, then the rational lower-bound fold, and only then exports the uniform-modulus row. It supplies the finite-net lower-bound route described in chapter 9237, without importing open-cover compactness, sequential compactness, choice of a minimum radius, quotient real equality, or an ambient continuity theorem.

Proof. Use Definition 3934.2. It orders the compact-net row, pointwise-modulus row, positive-radius row, rational-fold row, compatibility row, and uniform-modulus output inside one accepted carrier.

The finite net in K supplies the centers that M may inspect. The row G records a positive radius below the local cover radius at each center, so the local moduli are tied to the same finite bundle that compactness exposed.

The fold row F and compatibility row R then produce the common precision history consumed by U. Because F is a $\mathsf{Rat}^{\uparrow}$ fold over the displayed bundle, the lower bound is stable under the classifier and does not depend on an external minimum operator.

Thus the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives a finite BEDC radius ledger. The refused analytic objects are absent from the carrier and cannot be reconstructed from H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, FiniteLebesgueNumberUp positive-radius rows, RatUp lower-bound folds, UniformModulusUp output rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3934.1, Definition 3934.2, and Definition 3934.3 seed the finite Lebesgue-radius ledger used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, sequential compactness, selected-subsequence extraction, host minimum selection, quotient real equality, or external analytic modulus choice.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, pointwise-modulus admission, positive-radius exposure, rational lower-bound folding, uniform-modulus output exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformRadiusLedgerUp.

Chapter 2958

A Concrete Naming Certificate for $\mathsf{LowerSemicontinuityReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, with the point-set vocabulary of Kelley and Bourbaki read only as finite BEDC routing data, the $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ packet specializes lower semicontinuity to the Real-and-Completion corridor. It is not a synonym of chapter 1536: that sibling names the general lower-real epigraph surface, while the present chapter records the real-name, regular-rational, topology, and completion-facing threshold rows needed by Real/Completion consumers. The packet sits beside the real carrier of chapter 38, the regular rational source of chapter 301, the stream schedule of chapter 126, the lower-real ledger of chapter 1446, the continuous-map surface of chapter 130, and the completion-facing real routes of chapter 3944.

Definition 2958.1 (LowerSemicontinuityReal carrier)**.** *A* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, T, F, W, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *source carrier row, read through accepted real-name histories rather than host real points;*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *threshold row whose basic opens are represented by finite rational cuts and stream windows;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *or value-row handoff used when the lower-semicontinuous function is inspected through a displayed map surface;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite source and value windows used by each threshold query;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for real-name and value approximants;*
6. *E is the* $\mathsf{LowerReal}^{\uparrow}$ *epigraph ledger, recording finite rows (x, q) that witness the rational lower threshold q at the inspected real name x;*
7. *O is the open-threshold route, recording that a query of the form $q < f(x)$ is answered by a finite lower-real row and then exposed as a basic* $\mathsf{Topology}^{\uparrow}$ *neighbourhood of the accepted real name;*

8. *H records componentwise* hsame *transport for real-name rows, topology thresholds, map rows, stream schedules, regular-rational readbacks, lower-real epigraph rows, threshold routes, provenance rows, and local naming rows;*
9. *C records displayed* Cont *replay from a threshold request through $X, T, F, W, R, E, O, H, P, N$;*
10. *P is the* Pkg *provenance over* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real completeness, quotient real equality, trichotomy, open-cover compactness, semantic liminf, or a selected limit point in a completed host space.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuityRealUp.LowerSemicontinuityRealTasteGate single carrier alignment`

Theorem 2958.2 (LowerSemicontinuityReal NameCert obligations)**.** *For an accepted* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *carrier $L = (X, T, F, W, R, E, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real-name, topology-threshold, map, stream, regular-rational, lower-real, open-threshold, transport, replay, provenance, and local-name rows; threshold exactness for the finite lower-real rows in E; regular-rational readback through R; stream scheduling through W; componentwise* hsame *transport through H; displayed* Cont *replay through C; and provenance through P.*

Proof. Project the carrier of Definition 3935.1. The rows $X, T, F, W, R, E, O, H, C, P, N$ are the whole Real-and-Completion packet, so carrier admission is rowwise.

Read threshold exactness through E and O. A rational query $q < f(x)$ is accepted only after the source real name has been scheduled by W, read back through R, and matched to a finite lower-real epigraph row in E.

Transport and replay are then forced by H and C. They move the real-name, threshold, map, schedule, regular-rational, lower-real, and open-neighbourhood rows together, so the local NameCert surface cannot replace them by host topology or quotient equality.

Finally P and N package the same finite rows. Since the excluded completeness, trichotomy, compactness, semantic-liminf, and selected-limit principles are absent from the carrier, they are not NameCert obligations. □

Theorem 2958.3 (LowerSemicontinuityReal open-threshold route)**.** *Every completion-facing consumer of an accepted* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *packet must read an open lower-threshold query through the finite route*

$$X, T, F, W, R, E, O \longmapsto H, C, P, N.$$

The route may expose Real-name windows, regular-rational approximants, stream schedules, lower-real epigraph rows, topology threshold rows, map handoffs, transport, replay, provenance, and local naming. It exports no host real completeness, quotient equality, trichotomy, open-cover compactness, semantic liminf, or completed-space selection principle.

Proof. Start with a threshold query. The source is the $\mathsf{Real}^{\uparrow}$ row X, and the neighbourhood vocabulary is the displayed $\mathsf{Topology}^{\uparrow}$ row T, so the query does not begin from a host open-set lattice.

Next use the scheduling and readback rows. The stream schedule W selects the finite windows, and the regular-rational row R supplies the approximants used by the lower-real ledger E.

The open-threshold row O turns the lower-real epigraph witness into the basic threshold route exposed to completion-facing consumers. The map row F is consumed only when the query is read through a displayed $\mathsf{ContinuousMap}^{\uparrow}$ surface.

Finally replay through H, C, P, N. These rows transport, continue, source, and name the same finite threshold packet. Since no excluded host principle is a row of the carrier, the route remains a Real-and-Completion specialization of lower semicontinuity rather than a completeness or compactness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3935.1, Definition 3935.2, and Definition 3935.3 seed the finite Real-and-Completion lower-semicontinuity packet over real-name windows, regular-rational readbacks, stream schedules, lower-real epigraph rows, topology thresholds, map handoffs, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real completeness, quotient real equality, trichotomy, open-cover compactness, semantic liminf, completed-space selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, regular-rational readback, stream scheduling, topology exposure, transport replay, provenance, and non-escape for BEDC.Derived.LowerSemicontinuityRealUp.

Constructive story: A $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ packet is generated from RealUp source histories, TopologyUp threshold rows, ContinuousMapUp handoffs, StreamNameUp schedules, RegSeqRatUp readbacks, LowerRealUp epigraph witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2959

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$

Following Bishop–Bridges regular Cauchy constructions, the $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ packet records the finite diagonal selector surface used by the Real and Completion route before a terminal limit representative is sealed. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the diagonal-limit representative boundary of chapter 1205. The packet names only a finite selector schedule, readback windows, tolerance rows, transport, replay, provenance, and local naming data; it does not export countable choice, host limits, quotient-stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2959.1 (Regular Cauchy limit selector carrier)**.** *A* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the selector schedule row, listing the finite diagonal stages and the requested precision levels.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the inspected source entries at each stage of A.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the dyadic error budget attached to the selected stages.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the selected stream windows and dyadic tolerances into regular rational observations.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the selector schedule, stream windows, dyadic tolerances, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for A, W, D, R, E and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite limit request through*

$$A, W, D, R \longmapsto E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, W, D, R, E, H, C, P, N$. The carrier has no coordinate for countable choice, selected host limits, quotient equality of stream names, hidden tail schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitSelectorUp.RegularCauchyLimitSelectorTasteGate single carrier alignment`

Theorem 2959.2 (Regular Cauchy limit selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ *carrier $S = (A, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the selector schedule,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every Real or Completion consumer of the selector reads the diagonal-limit handoff through these rows before using the representative surface of chapter 1205.*

Proof. Project the carrier of Definition 3936.1. The analytic rows are exactly the selector schedule A, finite stream window W, dyadic tolerance row D, regular rational readback R, and terminal seal E.

The selector route is finite. A consumer first reads the stages in A, then inspects only the windows named by W, then checks the dyadic tolerances recorded in D before any regular rational readback is available.

The readback row R is the only route from selected finite windows to the terminal seal E. Thus the Real-facing endpoint is reached through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ data before it can be consumed by the diagonal-limit representative boundary.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no coordinate for countable choice, selected host limits, quotient-stream equality, hidden tail schedules, ambient Real completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ packet is generated from a finite selector schedule, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$, seed)

Scope closed: Definition 3936.1 and Definition 3936.2 seed the finite diagonal selector packet consumed by Real and Completion routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, host limits, quotient-stream equality, hidden tail schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector schedule exactness, StreamName window exposure, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLimitSelectorUp.

Chapter 2960

A Concrete Naming Certificate for SubordinateFiniteModulusCover$^{\uparrow}$

Following Lebesgue-number and Bishop–Bridges compact-uniform-continuity arguments, the SubordinateFiniteMod packet records the finite subordinate cover that feeds the uniform-modulus fold in chapter 9237. It sits over the compact finite-net carrier of chapter 129, the pointwise map ledger of chapter 130, the positive-radius lower-bound surface of chapter 452, and the downstream uniform-modulus consumer of chapter 369. The packet names only the finite cover rows, pointwise modulus rows, rational fold ledger, transport, replay, provenance, and local naming data; it does not export open-cover compactness, a selected infinite subcover, host minima, quotient real equality, or an external semantic modulus.

Definition 2960.1 (Subordinate finite modulus cover carrier)**.** *A* SubordinateFiniteModulusCover$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (K, F, L, Q, U, H, R, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, naming the centers and the source-distance coverage windows inspected by the packet.*
2. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row, assigning a local modulus window to each displayed center.*
3. *L is the* FiniteLebesgueNumber$^{\uparrow}$ *subordinate-radius row, recording a positive radius below the cover radius for each displayed local modulus.*
4. *Q is the* Rat$^{\uparrow}$ *fold row that computes the finite lower-bound tolerance from the displayed subordinate radii.*
5. *U is the* UniformModulus$^{\uparrow}$ *handoff row, exposed only after K, F, L, Q have been displayed.*
6. *H records componentwise* hsame *transport for K, F, L, Q, U and for the replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from compact-net inspection through pointwise modulus rows, subordinate radii, rational folding, and uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* SubordinateFiniteModulusCover$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, Rat$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, F, L, Q, U, H, R, P, N$. The carrier has no coordinate for open-cover compactness, infinite subcover selection, host minimum extraction, quotient equality of real radii, ambient completeness, Lean verification, or bridge credentials.

Lean4 ✓ `BEDC.Derived.SubordinateFiniteModulusCoverUp.SubordinateFiniteModulusCoverTasteGate`
`single carrier alignment`

Theorem 2960.2 (Subordinate finite modulus cover NameCert obligations)**.** *For every accepted* $\mathsf{SubordinateFiniteModulusCover}^{\uparrow}$ *carrier $C = (K, F, L, Q, U, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}}$ *surface consists exactly of compact-net admission through K, pointwise-modulus admission through F, subordinate positive-radius exposure through L, finite rational lower-bound folding through Q, uniform-modulus handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3937.1. The analytic rows are the compact-net row K, the pointwise-modulus row F, the subordinate-radius row L, the rational fold Q, and the uniform-modulus handoff U.

The compact row K supplies only a finite bundle of centers and source-distance windows. The continuous-map row F is read at those displayed centers, and L records the positive subordinate radius attached to each local modulus row.

The fold row Q is a finite $\mathsf{Rat}^{\uparrow}$ computation over the displayed subordinate radii. It is not a host minimum over an open cover and cannot select an infinite subcover or appeal to quotient equality of real radii.

Finally U, H, R, P, N export, transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for open-cover compactness, infinite subcover choice, host minima, quotient real equality, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$.

Constructive story: A $\mathsf{SubordinateFiniteModulusCover}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, FiniteLebesgueNumber subordinate-radius rows, RatUp lower-bound folds, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$, seed)

Scope closed: Definition 3937.1 and Definition 3937.2 seed the subordinate finite-cover carrier consumed by compact-uniform-continuity routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite subcover selection, host minimum extraction, quotient equality of real radii, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, pointwise-modulus admission, subordinate-radius exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.SubordinateFiniteModulusCoverUp.

Chapter 2961

A Concrete Naming Certificate for $\mathsf{LocatedIntermediateValue}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedIntermediateValue}^{\uparrow}$ packet records the finite bisection route used to locate a sign change on a real interval without selecting a root as host data. It sits over the real-name surface of chapter 38, the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance core of chapter 291, the located interval carrier of chapter 394, and the nested interval route of chapter 2249. The packet names only endpoint rows, located sign-apartness observations, finite bisection data, regular Cauchy readback, transport, replay, provenance, and local naming obligations; it does not export trichotomy, open-cover compactness, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2961.1 (Located intermediate value carrier)**.** *A* $\mathsf{LocatedIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, S, B, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the located interval endpoint row naming the two* $\mathsf{Real}^{\uparrow}$ *endpoint seals and their* $\mathsf{LocatedInterval}^{\uparrow}$ *order witness.*
2. *S is the located sign-apartness row recording finite negative and positive separation observations for the endpoint values.*
3. *B is the bisection ledger row listing the inspected dyadic midpoint stages, the retained subinterval side, and the sign row used at each stage.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the bisection ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows and dyadic tolerances into regular rational observations.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after E, S, B, W, R have been displayed.*
7. *H records componentwise* hsame *transport for E, S, B, W, R, Q and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located sign-change request through finite bisection and regular Cauchy readback.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntermediateValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedIntervalSelection}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $E, S, B, W, R, Q, H, C, P, N$. The carrier has no coordinate for real trichotomy, open-cover compactness, quotient equality of real names, an externally selected zero, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.LocatedIntermediateValueUp.LocatedIntermediateValueTasteGate single carrier alignment`

Theorem 2961.2 (Located intermediate value NameCert obligations)**.** *For every accepted* LocatedIntermediateValu *carrier $I = (E, S, B, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$ *surface consists exactly of endpoint admission, located sign-apartness rows, bisection ledger exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3938.1. The analytic rows are the endpoint row E, the sign-apartness row S, the bisection ledger B, the finite window row W, the regular rational readback R, and the real seal Q.

The endpoint signs are read as located apartness data. They are not converted into a trichotomy principle, a global sign decision for every point, or a quotient equality test for real names.

The bisection ledger is finite. Each step records the dyadic midpoint, the inspected windows, and the retained side before the next row is read. The regular Cauchy readback R is the only route from these finite windows to the terminal $\mathsf{Real}^{\uparrow}$ seal Q.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the packet contains no coordinate for open-cover compactness, selected roots, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 2961.3 (Located intermediate value nested interval route)**.** *Every accepted* $\mathsf{LocatedIntermediateValue}^{\uparrow}$ *carrier exposes its intermediate-value read through the finite bisection ledger B, the* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *dependency named in P, the finite window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance discipline, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q. A downstream consumer must read those rows before it may cite the packet as a located intermediate-value route.*

Proof. Read P to identify the nested interval dependency and the dyadic tolerance source. The only displayed interval refinement data in the carrier is the ledger B, so a consumer has no alternate route to a candidate interval.

Read W and R next. The windows are finite, and the regular rational readback is the handoff that turns the bisection ledger into a Cauchy-facing observation.

The endpoint of the route is the seal row Q, transported by H and replayed by C. The proof therefore uses finite bisection and regular Cauchy handoff only; it does not use trichotomy, open-cover compactness, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntermediateValue}^{\uparrow}$ packet is generated from located interval endpoint rows, located sign-apartness rows, a finite bisection ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3938.1, Definition 3938.2, and Definition 3938.3 seed the finite bisection packet consumed by constructive intermediate-value routes over located real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, open-cover compactness, quotient real equality, selected roots, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-apartness exposure, bisection ledger exactness, finite StreamName windows, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntermediateValueUp.

Chapter 2962

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$

Following Bishop and Bridges on the fan theorem and constructive compactness of intervals, the $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ packet records the finite fan/bar data used to inspect a closed bounded interval through finite binary and dyadic prefixes. It consumes the closed bounded interval source of chapter 1591, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-coordinate seal of chapter 38. The packet names finite bar coverage and prefix stability only as displayed BEDC rows; it does not assert open-cover compactness, selected subcovers, countable choice, quotient equality of real names, trichotomy, or ambient Real completeness.

Definition 2962.1 (Closed bounded interval fan carrier)**.** *A* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, B, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *sealing.*
2. *T is the finite binary-prefix tree row used to enumerate dyadic subinterval decisions.*
3. *B is the finite bar row, recording that every inspected branch meets a displayed prefix in T.*
4. *W is the prefix-stability row saying that extension of a branch past a bar prefix preserves the associated interval containment decision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint names, branch prefixes, and dyadic containment tests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the endpoint and prefix windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, T, B, W, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, tree, bar, stability, window, readback, and seal rows.*
9. *C records displayed* Cont *replay from interval-source inspection through finite prefix reads, bar coverage, regular rational readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$ *row.*

The carrier exports a finite fan/bar interface over the interval source. It has no coordinate for an open cover, a selected subcover, a choice function on branches, quotient equality of real names, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalFanUp.ClosedBoundedIntervalFanTasteGate single carrier alignment`

Theorem 2962.2 (Closed bounded interval fan NameCert obligations)**.** *For every accepted* ClosedBoundedIntervalF *carrier $F = (I, T, B, W, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, finite prefix-tree admission through T, finite bar exposure through B, prefix-stability exposure through W, finite observation through S, regular rational readback through R, Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3939.1. The interval source I is already the closed bounded interval packet of chapter 1591; the fan-specific rows are the finite tree T, finite bar B, and prefix-stability row W.

Read the observation and seal rows next. The $\mathsf{StreamName}^{\uparrow}$ window S and $\mathsf{RegSeqRat}^{\uparrow}$ readback R inspect endpoint and prefix data at finite precision, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of those finite rows.

The structural rows close the local certificate. Transport H, replay C, provenance P, and naming N preserve exactly this finite fan/bar route, so the obligations introduce no open-cover compactness, selected subcover, choice over branches, quotient equality, trichotomy, or ambient completeness. □

Theorem 2962.3 (Closed bounded interval fan finite-bar route)**.** *Every compactness-facing or uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto T, B, W$$

before it uses fan/bar data. The route first displays the closed bounded interval source, then finite stream windows and regular rational readbacks, then the Real seal, and only then exposes the finite prefix tree, finite bar, and prefix-stability rows. It supplies the finite bar interface needed by interval-net, compactness, and uniform-modulus consumers without claiming open-cover compactness, choice of a subcover, quotient real equality, trichotomy, or ambient completeness.

Proof. Start with the interval source row. By Definition 1591.3, the source I already displays endpoint order, containment ledgers, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ sealing, transport, replay, provenance, and naming.

Use Definition 3939.2. It places the finite prefix tree T, finite bar B, and prefix-stability row W inside the same accepted carrier, downstream of the interval source and the finite readback route.

Thus a consumer receives fan data only as a finite BEDC route over an accepted closed bounded interval. Since the carrier has no open-cover coordinate, no branch selector, no quotient-real equality row, no trichotomy row, and no ambient completeness row, the finite-bar route cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, finite binary-prefix tree rows, finite bar rows, prefix-stability rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3939.1, Definition 3939.2, and Definition 3939.3 seed the finite fan/bar surface for closed bounded interval compactness and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, selected subcovers, countable choice, quotient real equality, trichotomy, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix-tree admission, finite bar exposure, prefix-stability exposure, StreamName readback, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalFanUp.

Chapter 2963

A Concrete Naming Certificate for $\mathsf{DiagonalSequence}^{\uparrow}$

Following Cantor and Bishop–Bridges constructive analysis, the $\mathsf{DiagonalSequence}^{\uparrow}$ packet records the finite diagonal-sequence surface used by Real-facing diagonal arguments. It sits over the stream-row discipline of chapter 126, the dyadic bit ledger of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. It is consumed downstream by chapter 3793 and **??**. The packet names the reusable diagonal schedule and readback route; it is not a host countability theorem, a quotient-stream principle, or a completed powerset construction.

Definition 2963.1 (Diagonal sequence carrier)**.** *A* $\mathsf{DiagonalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, W, B, F, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the inspected enumeration surface, exposing only finite* $\mathsf{StreamName}^{\uparrow}$ *candidate windows.*
2. *W is the diagonal schedule row, pairing each inspected enumeration row with the coordinate at which the diagonal value is read.*
3. *B is the dyadic observation row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any regular rational or Real-facing seal is exposed.*
4. *F is the finite flip row, assigning the displayed opposite dyadic value at each scheduled coordinate.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the same schedule and flip ledger.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, downstream of E, W, B, F, Q.*
7. *H records componentwise* hsame *transport for enumeration, schedule, dyadic observation, flip, regular readback, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an enumeration request through the scheduled dyadic read, flip, regular readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DiagonalSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the diagonal schedule is finite, the dyadic observation and flip rows share the same coordinate ledger, the regular readback reuses that ledger, and the Real seal is downstream of the displayed readback route.

Lean4 (definition) `BEDC.Derived.DiagonalSequenceUp.DiagonalSequenceUp`

Theorem 2963.2 (Diagonal sequence NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalSequence}^{\uparrow}$ *carrier* $D = (E, W, B, F, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$ *surface consists exactly of enumeration-window admission through* E*, finite diagonal scheduling through* W*, dyadic observation through* B*, flip exactness through* F*, regular rational readback through* Q*, Real sealing through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3940.1. The displayed inventory is exactly $E, W, B, F, Q, R, H, C, P, N$.

Read the construction rows in order. The enumeration surface E is inspected only along the finite schedule W, the dyadic observation B is made at those scheduled coordinates, and the flip row F assigns the opposite displayed dyadic value on the same ledger.

The readback rows are downstream. The row Q records regular rational readback from the scheduled flip ledger, while R seals the resulting sequence as $\mathsf{Real}^{\uparrow}$ data. Componentwise transport, displayed replay, provenance, and naming preserve those rows and introduce no extra coordinate. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 2963.3 (Diagonal sequence nonescape)**.** *Every accepted* $\mathsf{DiagonalSequence}^{\uparrow}$ *consumer must read the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed schedule W *and flip ledger* F *before it can expose a diagonal Real-facing value. The packet exports no host enumeration principle, quotient stream equality, unbounded choice of coordinates, powerset of reals, or completed uncountability theorem.*

Proof. Use Definition 3940.2. It orders the inspected stream windows, dyadic observations, flip ledger, regular readback, and Real seal inside one finite carrier.

The nonescape clauses follow from the absence of any other carrier coordinate. The rows H, C, P, N can transport, replay, source, and name the displayed route, but they cannot add a host enumeration theorem or a quotient equality principle. Thus every downstream diagonal packet must consume the same finite sequence route rather than bypass it. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiagonalSequence}^{\uparrow}$ packet is generated from finite StreamName windows, a diagonal schedule, DyadicRatCore observations, a flip ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3940.1, Definition 3940.2, and Definition 3940.3 seed the reusable finite diagonal-sequence packet consumed by Real-facing diagonal arguments.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host countability, quotient stream equality, unbounded coordinate choice, powerset constructions, Real uncountability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exactness, dyadic flip compatibility, regular readback, Real sealing, provenance exhaustion, and local naming for BEDC.Derived.DiagonalSequenceUp.

Chapter 2964

A Concrete Naming Certificate for $\mathsf{WobbleSaturationSeal}^{\uparrow}$

The $\mathsf{WobbleSaturationSeal}^{\uparrow}$ packet realizes the f_3 directional saturation route of Definition 9462.1423 as finite NameCert data. It consumes the six-coordinate codon window of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage packet of chapter 3470, the wobble quotient spectral trigger of chapter 2317, and the radial wobble spectrum of chapter 3681. The packet names the unique non-leaking f_3 saturation seal; it is not a biological stochastic model, quotient codon semantics, or an unrestricted support-closure oracle.

Definition 2964.1 (Wobble saturation seal carrier)**.** *A* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, C, Q, R, D, A, H, K, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row exposing the six Boolean coordinates of the inspected codon window.*
2. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *row naming the boundary tile through which the wobble read is admitted.*
3. *C is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row recording finite coverage of the relevant boundary support.*
4. *Q is the* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *row exposing the quotient-facing spectral trigger.*
5. *R is the* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *row exposing the radial spectrum and wobble-edge accounting.*
6. *D is the directional saturation row, recording that the only non-leaking active-support saturation direction is f_3.*
7. *A is the* ACG *seal row, recording the median, symmetry, spectral, and final f_3 sealing corner.*
8. *H records componentwise* hsame *transport for window, boundary-tile, coverage, quotient-trigger, radial-spectrum, saturation, seal, replay, provenance, and naming rows.*
9. *K records displayed* Cont *replay from the six-bit window through boundary coverage, quotient trigger, radial spectrum, and f_3 seal.*
10. *P is the* Pkg *provenance over* $\mathsf{WobbleSaturationSeal}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the f_3 saturation row is read after the six-bit window, boundary-tile, coverage, quotient-trigger, and radial-spectrum rows, and when the ACG *seal is the displayed terminal corner for that same route.*

Lean4 ✓ `BEDC.Derived.WobbleSaturationSealUp.WobbleSaturationSealTasteGate single carrier alignment`

Theorem 2964.2 (Wobble saturation seal NameCert obligations)**.** *For every accepted* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *carrier $S = (W, T, C, Q, R, D, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}$ *surface consists exactly of six-bit window admission through W, boundary-tile admission through T, six-boundary coverage through C, quotient spectral triggering through Q, radial wobble-spectrum exposure through R, unique f_3 directional saturation through D,* ACG *seal exposure through A, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3941.1. The rows $W, T, C, Q, R, D, A, H, K, P, N$ are the complete displayed inventory.

Read the support rows first. The six-bit window, boundary-tile, and coverage rows identify the finite codon support surface before any wobble saturation is visible.

Read the wobble rows next. The quotient spectral trigger Q and radial spectrum R supply the two sibling wobble surfaces, and D records the unique non-leaking f_3 direction over that shared surface. The row A then exposes ACG as the terminal seal for the same route.

Finally H, K, P, N transport, replay, source, and name exactly those rows. Since the carrier has no stochastic, quotient-semantic, or unrestricted support-closure coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2964.3 (Wobble saturation seal nonescape)**.** *Every accepted* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *consumer must read* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$*, and* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *before the unique f_3 saturation seal can be exposed. The seal exports no biological stochastic model, quotient codon semantics, unrestricted support closure, or route that treats a leaking direction as equivalent to the displayed f_3 row.*

Proof. Apply the obligation theorem. It places the six-bit window, boundary tile, coverage, quotient trigger, and radial spectrum before the directional saturation row.

The nonescape claim is then a carrier-inventory statement. The rows D and A name the unique f_3 saturation route and the terminal ACG seal, while H, K, P, N preserve the same route. No row remains from which to read a stochastic model, quotient codon semantics, or an unrestricted closure oracle. □

Falsifiable prediction: An accepted $\mathsf{WobbleSaturationSeal}^{\uparrow}$ consumer that exposes the f_3 saturation seal without first displaying CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WobbleQuotientSpectralTrigger, RadialWobbleSpectrum, hsame, Cont, Pkg, and local NameCert rows refutes the carrier surface.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, wobble-quotient-spectral-trigger, radial-wobble-spectrum: $\mathsf{WobbleSaturationSeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six-bit window, boundary-tile, coverage, quotient-trigger, radial-spectrum, unique f_3 saturation, terminal ACG seal, transport, replay, provenance, and local naming rows.

The Lean TasteGate route for WobbleSaturationSeal$^{\uparrow}$ must provide explicit Nontrivial WobbleSaturationSeal$^{\uparrow}$ and FieldFaithful WobbleSaturationSeal$^{\uparrow}$ instances preserving $W, T, C, Q, R, D, A, H, K, P, N$ as distinguishable fields.

Closure status for NameCert$_{\text{WobbleSaturationSeal}^{\uparrow}}$.

Constructive story: A WobbleSaturationSeal$^{\uparrow}$ packet is generated from six-bit codon windows, boundary-tile rows, six-boundary coverage, quotient spectral trigger rows, radial wobble-spectrum rows, a unique f3 saturation direction, the ACG seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{WobbleSaturationSeal}^{\uparrow}}$, seed)

Scope closed: Definition 3941.1, Definition 3941.2, and Definition 3941.3 seed the finite NameCert seal for the unique f_3 directional saturation route.

Formal verification: FormalStatus(NameCert$_{\text{WobbleSaturationSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological stochastic dynamics, quotient codon semantics, unrestricted support closure, Lean verification, bridge checking, or saturation beyond the displayed f3 route.

Upgrade path: Obligation closure requires checked carrier admission, six-bit window alignment, boundary-tile coverage, quotient-trigger exactness, radial-spectrum compatibility, unique f3 saturation, ACG sealing, Nontrivial WobbleSaturationSealUp, and FieldFaithful WobbleSaturationSealUp for BEDC.Derived.WobbleSaturationSealUp.

Chapter 2965

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ packet records the finite witness surface by which two regular Cauchy reads with the same displayed tail data are forced to have the same terminal Real-facing seal. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38; the packet stores only paired tail windows, dyadic witnesses, regular readbacks, seal comparisons, transport, replay, provenance, and local naming.

Definition 2965.1 (Regular Cauchy limit uniqueness carrier)**.** *A* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two regular Cauchy source rows whose finite tails are being compared.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger witnessing the common scale at which the two source rows are inspected.*
3. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window naming the finite tail entries used in the comparison.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two inspected tails.*
5. *E is the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the dyadic, stream-window, and regular-rational rows have been displayed.*
6. *H records componentwise* hsame *transport for source rows, dyadic tolerance, finite windows, regular readbacks, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay for any finite uniqueness request through L, R, D, W, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, D, W, Q, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.RegularCauchyLimitUniquenessUp.RegularCauchyLimitUniquenessTasteGate
single carrier alignment

Theorem 2965.2 (Regular Cauchy limit uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniquen}}$ *surface consists exactly of carrier admission for the paired regular Cauchy source rows, dyadic witness ledger, finite stream windows, regular rational readbacks, terminal Real seal comparison, componentwise transport, displayed replay, provenance, and local naming. No uniqueness obligation is stored outside these rows.*

Proof. Project the accepted carrier through Definition 3942.1. The analytic rows are exactly the two source rows L, R, the dyadic witness D, the finite stream window W, the regular rational readback Q, and the terminal seal comparison E.

The structural rows are H, C, P, N. They transport, replay, source, and name the same finite packet, so the NameCert surface cannot add a quotient equality, selected limit, host real equality, choice selector, trichotomy, or ambient completeness coordinate. □

Theorem 2965.3 (Regular Cauchy limit uniqueness dyadic witness determinacy)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, the dyadic witness row* D *determines the displayed comparison scale used by the paired stream window* W *and regular rational readback* Q*. Any accepted uniqueness read must therefore inspect the same finite dyadic tolerance before the terminal seal row* E *is available.*

Proof. Begin with the dyadic tolerance ledger. The row D is the only coordinate whose classifier names the comparison scale, and the window row W is admitted only as a finite inspection at that displayed scale.

The regular rational readback Q consumes the same finite window. Hence the terminal seal row E is downstream from the dyadic witness and cannot be read from an independent scale, selected subsequence, quotient stream, or host equality witness. □

Theorem 2965.4 (Regular Cauchy limit uniqueness Real seal nonescape)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, every terminal* $\mathsf{Real}^{\uparrow}$ *seal comparison factors through the route*

$$L, R \to D \to W \to Q \to E \to H, C, P, N.$$

The exported uniqueness surface contains no quotient stream equality, no host real equality, no selected limit, no countable-choice schedule, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from a terminal seal comparison. The carrier exposes the two source rows only through L, R, and the comparison scale is fixed by D before any finite stream entries can be inspected.

The window and readback rows W, Q then force the comparison through displayed finite tails and regular rational data. The Real-facing endpoint E is therefore a seal row for that already displayed finite route, not an independent completed real.

Finally H, C, P, N preserve and name the same packet. Since the carrier has no coordinate for quotient stream equality, host equality, selected limits, choice, trichotomy, or completeness, the terminal seal comparison cannot escape the listed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ packet is generated from paired regular Cauchy source rows, a shared dyadic tolerance ledger, finite StreamName windows, RegSeqRat readbacks, a paired RealUp seal row, componentwise transport, displayed continuation replay, package provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3942.1, Definition 3942.2, Definition 3942.3, and Definition 3942.4 seed the finite-window uniqueness surface for regular Cauchy limit reads over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyLimitUniquenessUp.RegularCauchyLimitUniquenessTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, trichotomy, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic witness determinacy, Real-seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 2966

A Concrete Naming Certificate for $\mathsf{FormalBallBasis}^{\uparrow}$

Bishop-style constructive metric-basis language presents completion data through located balls, positive rational radii, and finite cover-refinement witnesses before any quotient of Cauchy names is read. The $\mathsf{FormalBallBasis}^{\uparrow}$ packet records that basis-level substrate between the metric surface of chapter 57, the dyadic arithmetic core of chapter 291, the real-name seal of chapter 38, the formal-ball packet of chapter 1510, and the metric-completion route of chapter 1506. It is a finite source for L10 completion and Cauchy-filter consumers, not an open-cover compactness theorem, not a quotient-ball construction, and not an ambient completeness principle.

Definition 2966.1 (Formal-ball basis carrier)**.** *A* $\mathsf{FormalBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, d, c, r, I, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row that supplies the inspected points and distance requests.*
2. *d is the metric-distance ledger read through the same* $\mathsf{Met}^{\uparrow}$ *surface.*
3. *c is the finite centre schedule, listing the displayed centres used by the basis rows.*
4. *r is the positive radius ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording dyadic tolerances and their terminal real-name seals.*
5. *I is the incidence row recording which displayed point requests enter which formal balls.*
6. *R is the refinement row, recording finite ball-containment and radius-shrink requests over* $\mathsf{FormalBall}^{\uparrow}$ *before any completion consumer is reached.*
7. *H records componentwise* hsame *transport for the metric, centre, radius, incidence, refinement, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric requests through centres, radii, formal-ball incidence, refinement, and completion-facing handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FormalBall}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient balls, selected centres beyond c, selected Cauchy limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallBasisUp.FormalBallBasisTasteGate single carrier alignment`

Theorem 2966.2 (Formal-ball basis NameCert obligations)**.** *For an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *carrier*

$$B = (X, d, c, r, I, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-row exposure through X, d, centre-schedule exposure through c, positive-radius exposure through r, formal-ball incidence through I, finite refinement through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. Every basis read is exhausted by these rows.*

Proof. Project the carrier of Definition 3943.1. The displayed rows are exactly the metric carrier, metric ledger, centre schedule, radius ledger, incidence row, refinement row, transport row, replay row, provenance row, and local name.

The metric and radius rows supply the source data. The row X provides the inspected metric points and d provides the finite distance requests, while r records only dyadic and real-name radius data.

The formal-ball rows are then I and R. Incidence and refinement are read after the centres and radii have been displayed, so no consumer can substitute quotient balls or a selected completion point for the basis rows.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 2966.3 (Formal-ball basis metric-completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *packet must read the displayed route*

$$X, d \longmapsto c, r \longmapsto I, R \longmapsto H, C, P, N$$

before completion data is available. Thus the completion handoff sees a finite basis of metric centres, dyadic and real radii, formal-ball incidence, and formal-ball refinement, and it receives no quotient ball carrier, selected Cauchy limit, open-cover compactness witness, selected centre outside c, or ambient completeness row.

Proof. Start with the NameCert obligations of Definition 3943.2. They force every consumer to read the metric rows X, d, the centre and radius rows c, r, and the formal-ball rows I, R before any structural row is used.

The provenance row names $\mathsf{MetricCompletion}^{\uparrow}$ only as a downstream consumer of the same route. Therefore completion-facing reads inherit the displayed formal-ball basis rather than a quotient of Cauchy names or a host completion object.

Transport and replay through H and C preserve the route, while P and N only source and name it. The excluded compactness, quotient-ball, selected-centre, selected-limit, and ambient-completeness objects are absent from the carrier. □

Theorem 2966.4 (Formal-ball basis Cauchy-filter refinement)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *packet must read the metric row X, d, the centre schedule c, the dyadic and real radius ledger r, the incidence row I, and the refinement row R before completion-facing data is available. The consumer route factors through* $\mathsf{FormalBall}^{\uparrow}$ *and*

$\mathsf{MetricCompletion}^{\uparrow}$ *using only H, C, P, N for transport, replay, provenance, and local naming, and it receives no quotient filter, selected limit, countable-choice schedule, or ambient completeness row.*

Proof. Project the accepted carrier $B = (X, d, c, r, I, R, H, C, P, N)$ of Definition 3943.1. The NameCert obligations of Definition 3943.2 expose the metric row, centre schedule, radius ledger, incidence row, and finite refinement row as the whole basis surface.

The Cauchy-filter consumer therefore starts from the same displayed ball data as the metric-completion consumer. The formal-ball incidence row I names the inspected balls, and the refinement row R supplies the finite containment and radius-shrink reads before any completion-facing handoff can be consumed.

Now route through the existing handoff of Definition 3943.3. The rows H, C, P, N preserve and name the route from metric data through $\mathsf{FormalBall}^{\uparrow}$ into $\mathsf{MetricCompletion}^{\uparrow}$, so a $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ reader receives the finite formal-ball refinement corridor rather than a quotient filter or a chosen limit.

Since the carrier has no coordinate for quotient filters, selected Cauchy limits, countable choice, or ambient completeness, those objects cannot be exported by the refinement read. □

Falsifiable prediction: If a completion-facing read uses a formal-ball basis without displaying the metric row, centre schedule, dyadic or real radius ledger, incidence row, and refinement row, it is not a $\mathsf{FormalBallBasis}^{\uparrow}$ packet.

Independence witness: Nearest siblings: formalball, metriccompletion, bishopregularcauchy-completion, cauchyfiltercompletioncriterion, dyadicratcore; FormalBallBasisUp is not bijective to them because it binds a finite centre-radius basis and refinement row between metric balls and completion consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalBallBasis}^{\uparrow}$ packet is generated from a displayed MetricSpace row, metric-distance ledger, finite centre schedule, DyadicRatCore and Real radius rows, FormalBall incidence, finite refinement, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3943.1, Definition 3943.2, and Definition 3943.3 seed the finite basis-level substrate between formal balls and metric completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FormalBallBasisUp.FormalBallBasisTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient balls, selected centres beyond the displayed rows, selected Cauchy limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, centre and radius visibility, formal-ball incidence, refinement transport, replay stability, and metric-completion handoff non-escape for BEDC.Derived.FormalBallBasisUp.

Chapter 2967

A Concrete Naming Certificate for $\mathsf{RealConvergentSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealConvergentSequence}^{\uparrow}$ packet records displayed convergence of a real sequence to a named $\mathsf{Real}^{\uparrow}$ limit as finite NameCert data. It adjoins the L10 Real and completion route by sitting over the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the real-name seal of chapter 38, the real sequence limit laws of chapter 1480, and the Cauchy convergence criterion surface of chapter 1509. The packet separates convergence evidence from Cauchy completeness: it names a displayed source sequence, a displayed limit row, and the finite tolerance route witnessing convergence, but it does not assert that every Cauchy stream converges.

Definition 2967.1 (Real convergent sequence carrier)**.** *A* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, L, D, M, W, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the sequence-source row, exposing the real-sequence index discipline, the finite observation requests, and the source-side* $\mathsf{StreamName}^{\uparrow}$ *handles.*
2. *L is the candidate* $\mathsf{Real}^{\uparrow}$ *limit row, carrying the named real limit, its terminal real-name seal, and the limit-side* $\mathsf{StreamName}^{\uparrow}$ *handles.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested tolerance, the comparison tolerance, and the finite split budget used by each convergence read.*
4. *M is the regular-Cauchy modulus row, storing the tail threshold assigned to each dyadic tolerance request before any consumer reads the convergence classifier.*
5. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window row for source observations after the threshold and for the candidate-limit windows used in the same request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source windows, limit windows, and tolerance comparisons opened by W and D.*
7. *H records componentwise* hsame *transport for the source row, candidate limit, dyadic ledger, regular-Cauchy modulus, windows, readback, replay row, provenance row, and local naming row.*

8. *K records displayed* Cont *replay from a convergence query through the dyadic tolerance request, the regular-Cauchy threshold, the selected windows, the rational readback, and the candidate* $\mathsf{Real}^{\uparrow}$ *limit row.*

9. *P is the* Pkg *provenance over* $\mathsf{RealConvergentSequence}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RealSequenceLimitLaws}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, D, M, W, R, H, K, P, N$. It has no coordinate for Cauchy completeness, selected limits for arbitrary Cauchy streams, quotient-stream equality, countable choice, trichotomy, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RealConvergentSequenceUp.RealConvergentSequenceTasteGate single carrier alignment`

Theorem 2967.2 (Real convergent sequence NameCert obligations)**.** *For every accepted* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier*

$$C = (S, L, D, M, W, R, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source sequence row, candidate* $\mathsf{Real}^{\uparrow}$ *limit, dyadic tolerance ledger, regular-Cauchy modulus, finite windows, regular rational readback, transport, replay, provenance, and local naming rows;*

2. *source-limit exposure, requiring S and L to be the only source of the displayed sequence and the displayed limit used by the convergence classifier;*

3. *tolerance and modulus exactness, requiring every convergence request to read its dyadic tolerance from D and its tail threshold from M before any source observation is compared with the limit;*

4. *window and readback alignment, requiring W and R to expose only the source windows and limit windows selected by the same tolerance request;*

5. *limit-law compatibility, requiring uses of real sequence limit laws to consume only the displayed rows S, L, D, M, W, R and never a second limit surface;*

6. *Cauchy-criterion boundary, requiring any later Cauchy consumer to receive convergence evidence as input rather than a completeness principle;*

7. *componentwise transport, displayed replay, provenance, and local naming through H, K, P, N;*

8. *refusal of Cauchy completeness, arbitrary limit selection, quotient-stream equality, countable choice, trichotomy, Lean verification, and bridge checking as local certificate data.*

No real-convergent-sequence obligation is stored outside these rows.

Proof. Project the carrier of Definition 3944.1. Its visible coordinates are the source sequence row S, candidate limit row L, dyadic tolerance ledger D, regular-Cauchy modulus M, window row W, readback row R, and structural rows H, K, P, N.

The convergence request first reads D and M. The dyadic row fixes the requested tolerance and comparison budget, while the modulus row fixes the tail threshold for source observations.

The selected observations are then opened through W and R. The same tolerance request chooses both the source windows and the candidate-limit windows, and the regular rational row reads back only those scheduled observations.

Limit-law compatibility follows because every law-facing read has access only to S, L, D, M, W, R before the structural rows. A second limit surface or host equality proof is not a carrier coordinate.

Finally, H, K, P, N transport, replay, record, and name the same finite route. Since the excluded principles do not occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 2967.3 (Real convergent sequence L10 route)**.** *Every accepted* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier* $C = (S, L, D, M, W, R, H, K, P, N)$ *presents convergence evidence through the finite route*

$$S, L \longmapsto D \longmapsto M \longmapsto W \longmapsto R \longmapsto H, K, P, N.$$

Any Cauchy convergence criterion consumer downstream of C *receives only the displayed source sequence, candidate* $\mathsf{Real}^{\uparrow}$ *limit, dyadic tolerance ledger, regular-Cauchy threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Start with a downstream consumer. The carrier has no direct completeness coordinate, so the consumer must read the displayed convergence data from S and L.

The tolerance request then passes through D and M. These rows determine the comparison budget and the tail threshold before any observation window is opened.

The windows and rational observations are supplied by W and R. They expose only the scheduled source and candidate-limit data selected by the same request.

The structural rows H, K, P, N preserve the route for transport, replay, provenance, and naming. Thus every downstream criterion-facing read factors through the displayed finite route and cannot export Cauchy completeness, selected arbitrary limits, quotient-stream equality, countable choice, trichotomy, Lean evidence, or bridge data. □

Falsifiable prediction: Every $\mathsf{RealConvergentSequence}^{\uparrow}$ consumer read factors through a displayed source sequence, candidate $\mathsf{Real}^{\uparrow}$ limit, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, regular-Cauchy modulus, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, real-sequence-limit-laws, cauchy-convergence-criterion: $\mathsf{RealConvergentSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source sequence, candidate Real limit, dyadic tolerance ledger, regular-Cauchy modulus, scheduled source-and-limit windows, regular rational readback, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RealConvergentSequence}^{\uparrow}$ packet is generated from a displayed source sequence, a candidate RealUp limit, a DyadicRatCore tolerance ledger, a regular-Cauchy modulus, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3944.1, Definition 3944.2, and Definition 3944.3 seed the finite route from displayed real-sequence convergence evidence to later Cauchy-criterion consumption without asserting completeness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Cauchy completeness, selected limits for arbitrary Cauchy streams, quotient-stream equality, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-limit exposure, tolerance and modulus exactness, window and readback alignment, limit-law compatibility, Cauchy-criterion boundary, transport, replay, provenance, and local naming for BEDC.Derived.RealConvergentSequenceUp.

Chapter 2968

A Concrete Naming Certificate for $\mathsf{LocatedCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ packet records the finite route by which located regular rational windows expose a Cauchy subsequence handoff to the Real and Completion corridor. It consumes the located tail-window discipline of chapter 2376, the finite extraction surface of chapter 1854, the cofinal regular-subsequence stability row of chapter 3515, the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a displayed subsequence extraction surface for located regular names; it is not a selected infinite subsequence, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2968.1 (LocatedCauchySubsequence carrier)**.** *A* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, R, D, Q, I, S, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger, listing the source windows and selected subsequence windows inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the located regular rational source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by Cauchy and locatedness requests.*
4. *Q is the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison row, exposing located tail windows and finite cut comparisons.*
5. *I is the subsequence-index row, recording the monotone selected indices and the source windows from which they are read.*
6. *S is the* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *stability row, binding the extracted windows to the Cauchy tolerance schedule.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, R, D, Q, I, S have been displayed.*
8. *H records componentwise* hsame *transport for the window, readback, tolerance, located-comparison, index, extraction, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, Q, I, S, E, H, C, P, N$. It has no coordinate for a choice-selected infinite subsequence, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchySubsequenceUp.LocatedCauchySubsequenceTasteGate single carrier alignment`

Theorem 2968.2 (LocatedCauchySubsequence NameCert obligations)**.** *For every accepted* LocatedCauchySubsequ *carrier*

$$L = (W, R, D, Q, I, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$ *surface consists exactly of five obligation families: carrier admission for $W, R, D, Q, I, S, E, H, C, P, N$; located-window exposure through the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *row Q; subsequence extraction stability through the index row I and extraction row S;* $\mathsf{Real}^{\uparrow}$ *handoff through the seal row E; and refusal of selected subsequences, quotient stream equality, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3945.1. The packet displays exactly the stream windows, regular rational readback, dyadic tolerance ledger, located comparison row, subsequence-index row, extraction row, seal row, transport, replay, provenance, and local naming.

Located-window exposure is inherited from chapter 2376. A location request must pass through Q after reading the finite windows W, the regular rational row R, and the dyadic tolerance ledger D.

Subsequence extraction stability is the pair of rows I, S. The index row states which source windows are selected, and the extraction row states that the selected windows meet the displayed Cauchy tolerance schedule. Componentwise transport H and replay C preserve that route without adding a new selector.

The real-facing handoff is terminal at E. Since E is only reachable after $W, R, D, Q, I, S, H, C, P, N$ have been displayed, it exports a finite located subsequence seal and not a quotient completion or ambient completeness principle.

The classifier has no remaining coordinate beyond the listed rows. Thus selected subsequences, quotient stream equality, countable choice, trichotomy, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are refused by the carrier surface itself. □

Theorem 2968.3 (LocatedCauchySubsequence located-window exposure)**.** *Every accepted* LocatedCauchySubsequ *carrier exposes a located subsequence window only through the finite route*

$$W, R, D, Q, I, S, H, C, P, N.$$

In particular, the subsequence-index row cannot be consumed before the source $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison row have been displayed.*

Proof. Project the rows of Definition 3945.1. A located subsequence request first names a source observation window in W, reads the rational source through R, and consults the dyadic tolerance ledger D.

The located comparison row Q then decides the displayed tail-window comparison. Only after that row is present may the index row I and extraction row S expose the selected subsequence window.

Transport, replay, provenance, and local naming preserve the same route. No row in the packet allows I to be read independently of W, R, D, Q. □

Theorem 2968.4 (LocatedCauchySubsequence RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ *carrier factors through the located regular source, subsequence-index row, and extraction-stability row before reading the seal row. The handoff exposes only finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison rows, subsequence-index rows,* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *stability rows, transport, replay, provenance, and local naming.*

Proof. Begin with a Real-facing read. The seal row E is downstream of the source windows W, readback R, dyadic ledger D, located comparison row Q, index row I, and extraction row S.

Apply Definition 3945.2. The obligation surface forces every consumer-visible read to pass through those finite rows before the seal can be exposed.

The remaining structural rows only transport and replay the same displayed data. Therefore the handoff is a finite BEDC route from located regular rational windows to a terminal $\mathsf{Real}^{\uparrow}$ seal, not an ambient completion theorem. □

Falsifiable prediction: Every $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ read factors through displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, $\mathsf{LocatedRegSeqRat}^{\uparrow}$ comparison rows, subsequence-index rows, $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ stability rows, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locatedregseqrat, cauchysubsequenceextraction, regularcauchysubsequencestability, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located tail-window comparisons, monotone selected indices, extraction stability, finite stream windows, regular rational readback, dyadic tolerances, Real seal routing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ packet is generated from StreamName source and subsequence windows, RegSeqRat readback, DyadicRatCore tolerance rows, LocatedRegSeqRat comparison rows, subsequence-index rows, CauchySubsequenceExtraction stability rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$, seed)

Scope closed: Definition 3945.1, Definition 3945.2, Definition 3945.3, and Definition 3945.4 seed the finite located Cauchy subsequence route from displayed regular rational windows to Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected infinite subsequences, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-window exposure, subsequence extraction stability, RealUp handoff, and nonescape for BEDC.Derived.LocatedCauchySubsequenceUp.

Chapter 2969

A Concrete Naming Certificate for $\mathsf{CompletionReflectorTriangle}^{\uparrow}$

Following Mac Lane's adjunction triangle identities and the Bishop–Bridges construction of Cauchy completion, the $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ packet records the finite BEDC rows needed to read the unit, counit, and idempotent replay of the L10 completion reflector as one public triangle. It depends on the reflector packet of chapter 1585, the unit packet of chapter 1527, the counit packet of chapter 1539, the idempotent packet of **??**, the universal-property packet of chapter 397, and the reflection packet of chapter 403. Its real-facing route is read through chapter 360, chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a finite naming certificate for the triangle surface; it is not quotient completion equality, not a selected limit, not host equality, not countable choice, not ambient completeness, and not detached $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2969.1 (CompletionReflectorTriangle carrier)**.** *A* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, U, M, Q, I, E, R, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row for the displayed metric, regular-Cauchy, or real-name data entering the completion reflector.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row admitting the source into the completion-facing surface.*
3. *M is the completion-object row, read through the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and reflector surfaces rather than through a host quotient carrier.*
4. *Q is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning already completed or real-sealed reads through the displayed route.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *replay row for a second completion pass over the same completion-object row.*
6. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row consumed after the unit face is visible.*
7. *R is the* $\mathsf{CompletionReflection}^{\uparrow}$ *reflection row consumed after the unit, counit, idempotent, and extension faces are visible.*
8. *T is the local transport row through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *windows.*

9. *H records componentwise* hsame *transport for the source, unit, completion object, counit, idempotent replay, extension, reflection, transport, continuation, provenance, and naming rows.*
10. *C records displayed* Cont *replay along the unit-counit-idempotent triangle and through the local real-name transport row.*
11. *P is the* Pkg *provenance over* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$ $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *row.*

The classifier on accepted packets compares only these twelve rows under the displayed transport and replay data.

Lean4 ✓ `BEDC.Derived.CompletionReflectorTriangleUp.CompletionReflectorTriangleTasteGate single carrier alignment`

Theorem 2969.2 (CompletionReflectorTriangle NameCert obligations)**.** *For every accepted* CompletionReflectorTr *carrier $K = (S, U, M, Q, I, E, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source, unit, completion-object, counit, idempotent-replay, extension, reflection, transport, continuation, provenance, and naming rows; unit-first readability for completion-facing source reads; counit return for already completed or real-sealed reads; idempotent replay over the same completion-object row; extension and reflection consumption over the displayed triangle route; and non-escape from quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3946.1. The packet displays $S, U, M, Q, I, E, R, T, H, C, P, N$ and no other row.

The unit face reads S through U before the completion object M is consumed. The counit face reads Q only for data already on the completed or real-sealed side, and the idempotent face reads I over the same M row.

The extension and reflection faces are downstream of that route. The row E is consumed as the finite universal-property extension, and R is consumed as the reflection row over the same S, U, M, Q, I, T surface.

Finally H, C, P, N transport, replay, source, and name exactly the displayed route. Since no coordinate stores quotient completions, selected limits, host equality, countable choice, ambient completeness, or detached real completeness, the stated non-escape boundary follows. □

Theorem 2969.3 (CompletionReflectorTriangle unit-counit coherence)**.** *Let $K = (S, U, M, Q, I, E, R, T, H, C, P, N$ be an accepted* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ *carrier. The two triangle readings*

$$S \longmapsto U \longmapsto M \longmapsto I \longmapsto E, R \longmapsto H, C, P, N$$

and

$$S \longmapsto T \longmapsto Q \longmapsto I \longmapsto E, R \longmapsto H, C, P, N$$

meet at the same displayed idempotent, extension, reflection, transport-replay, provenance, and naming rows. Thus a consumer that first enters completion by the unit face and a consumer that first reads the real-name transport before counit return receive the same finite $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *boundary, with no quotient completion equality, selected limit, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness exported.*

Proof. Expose the carrier rows from Definition 3946.1. The only completion-facing rows are $S, U, M, Q, I, E, R, T, H, C, P, N$.

The first displayed reading is the unit leg: the source is admitted by U, the completion object M is read, and repeated completion is replayed by I before E and R are consumed.

The second displayed reading is the transport-counit leg: the source is read through T, returned by Q when already completed or real-sealed, and then replayed by the same I before E and R are consumed.

Both readings end in the same H, C, P, N rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row therefore identify the two readings as one finite triangle boundary. The excluded completion, selection, host-equality, choice, ambient-completeness, and detached-real-completeness data have no row in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ packet is generated from a displayed source row, unit admission, completion-object access, counit return, idempotent replay, universal-property extension, reflection consumption, local StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3946.1, Definition 3946.2, and Definition 3946.3 seed the finite unit-counit-idempotent triangle surface for the L10 completion reflector.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompletionReflectorTriangleUp.CompletionReflectorTriangleTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality, countable choice, ambient completeness, detached $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-counit coherence, idempotent replay, extension compatibility, reflection compatibility, local real-name transport, and non-escape for the same carrier.

The $\mathsf{LawvereMetricCompletion}^{\uparrow}$ hub exposes its sibling lattice against the dense metric-completion surface of chapter 3351 through the child route below.

Chapter 2970

A Concrete Naming Certificate for $\mathsf{LawvereMetricCompletion}^{\uparrow}$

Following Lawvere on generalized metric spaces and enriched-category distance, the $\mathsf{LawvereMetricCompletion}^{\uparrow}$ packet records the finite directed-distance profile surface through which an enriched metric completion can be consumed inside BEDC. It sits beside the Yoneda profile route of chapter 2148, the ordinary metric observation rows of chapter 57, the formal-ball approximation surface of chapter 1510, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a one-sided enriched completion boundary; it does not select host limits, quotient directed Cauchy objects, symmetric metric completions, global colimits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The sibling dependency route is substantive: $\mathsf{LawvereMetricCompletion}^{\uparrow}$ reads the $\mathsf{YonedaCompletion}^{\uparrow}$ profile route of chapter 2148, the $\mathsf{FormalBall}^{\uparrow}$ approximation cuts of chapter 1510, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any completion-facing directed profile can be consumed. The directed source row M is supplied by the $\mathsf{QuasiMetric}^{\uparrow}$ carrier route of chapter 2158 and is consumed before Yoneda profiles or FormalBall approximations.

Definition 2970.1 (Lawvere metric completion carrier)**.** *A* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, Y, F, D, A, T, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the directed* $\mathsf{QuasiMetric}^{\uparrow}$ *source row, carrying the visible point classifier and the finite one-sided distance comparisons accepted by the source packet.*
2. *Y is the* $\mathsf{YonedaCompletion}^{\uparrow}$ *handoff row, exposing the covariant and contravariant distance profiles retained for the displayed source points.*
3. *F is the finite probe-comparison row, listing the source probes against which two directed profiles are compared.*
4. *D is the directed-distance profile row, recording the left and right distance readings that make the completion candidate visible as a Lawvere-enriched profile.*
5. *A is the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row, displaying finite ball inclusions and radius cuts used before any completion-facing consumer reads D.*

6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row for directed profile errors, ball radii, and probe budgets.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained regular rational windows.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row for directed distance values and zero-distance comparison readback.*
9. *H records componentwise* hsame *transport for source rows, profile rows, probe comparisons, formal-ball approximations, dyadic tolerances, regular readbacks, Real seals, provenance, and local naming.*
10. *C records displayed* Cont *replay from source distances to Yoneda profiles, from profiles to formal-ball approximations, and from tolerance windows to Real readback.*
11. *P is the* Pkg *provenance over* $\mathsf{LawvereMetricCompletion}^{\uparrow}$, $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Y, F, D, A, T, R, S, H, C, P, N$. It has no coordinate for selected Cauchy weights, quotient profile equality, symmetric completion collapse, host enriched colimits, countable choice, or ambient real completeness.

Theorem 2970.2 (Lawvere metric completion NameCert obligations)**.** *For every accepted* LawvereMetricCompletio *carrier $L = (M, Y, F, D, A, T, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$ *surface consists exactly of directed source admission, YonedaCompletion handoff, finite probe comparisons, directed distance-profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every enriched-metric completion consumer must inspect those rows before treating the packet as a completion neighbour.*

Proof. Project the carrier of Definition 3947.1. The source and Yoneda rows M, Y are the only rows that introduce directed distance profiles from the displayed metric source.

The finite comparison route is F, D, A, T. Probe comparisons fix which source probes are being tested, the directed profile row records the left and right readings, the formal-ball row supplies finite approximation cuts, and the dyadic row records the tolerance accounting for those cuts.

Readback and transport are then forced through R, S, H, C, P, N. The regular rational windows and Real seal expose only the retained directed distance values, while componentwise transport, replay, provenance, and local naming preserve the same finite route.

Since the classifier has no additional coordinate, selected Cauchy weights, quotient profile equality, symmetric completion collapse, host enriched colimits, countable choice, and ambient real completeness cannot be read from the packet. □

Theorem 2970.3 (Lawvere metric completion finite profile nonescape)**.** *Every completion-facing read exported by an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet factors through*

$$\mathsf{QuasiMetric}^{\uparrow},\ \mathsf{YonedaCompletion}^{\uparrow},\ \mathsf{FormalBall}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The read first fixes the directed source and Yoneda profile rows, then the finite probe comparisons and formal-ball approximations, then dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and local naming. It exports no quotient directed Cauchy object, selected profile limit, symmetric metric completion collapse, host enriched colimit, or ambient completeness principle.

Proof. Let $L = (M, Y, F, D, A, T, R, S, H, C, P, N)$ be accepted. A completion-facing consumer first receives M, Y, F, D, so the directed profile is tied to displayed source distances, Yoneda handoff, and finite probe comparisons.

The approximation stage is A, T. Formal-ball inclusions and dyadic tolerance rows determine which finite directed-distance errors are visible before readback.

Finally R, S, H, C, P, N expose only regular rational windows, Real seals, transport, replay, provenance, and local naming for that same route. The excluded quotient, choice, colimit, symmetric-collapse, and ambient-completeness objects are not carrier fields, so no completion consumer can bypass the finite profile surface. □

Theorem 2970.4 (Lawvere metric completion directed-profile composition)**.** *Every completion-facing read from an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet composes through the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile handoff,* $\mathsf{QuasiMetric}^{\uparrow}$ *directed source data,* $\mathsf{FormalBall}^{\uparrow}$ *approximation rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows before it is visible as a Lawvere-enriched completion profile. The composed read exports directed source distances, finite probe comparisons, formal-ball approximation cuts, dyadic error budgets, regular rational windows, Real sealing, transport, replay, provenance, and local naming, but it exports no symmetric metric completion, quotient directed Cauchy object, selected profile limit, global colimit, countable-choice schedule, or ambient completeness claim.*

Proof. Project the carrier of Definition 3947.1. The directed source and profile handoff are exactly the rows M, Y, F, D.

Apply Definition 3947.2. It forces any completion-facing consumer to inspect directed source admission, YonedaCompletion handoff, finite probe comparison, directed profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Apply Definition 3947.3. The nonescape theorem identifies the public route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The composed read is therefore the displayed directed-profile route itself. Since the carrier has no coordinate for a symmetric completion, quotient directed Cauchy object, selected profile limit, host colimit, choice schedule, or ambient completeness witness, none of those objects can be exported by the composition. □

Theorem 2970.5 (Lawvere metric completion QuasiMetric source handoff)**.** *Any* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *read whose source distance is one-sided must first pass through the* $\mathsf{QuasiMetric}^{\uparrow}$ *carrier route of chapter 2158 before the* $\mathsf{YonedaCompletion}^{\uparrow}$ *and* $\mathsf{FormalBall}^{\uparrow}$ *rows are consumed. The completion-facing consumer may then read directed source distances, Yoneda profiles, finite probe comparisons, formal-ball approximation cuts, dyadic tolerance accounting, rational readback, Real sealing, transport, replay, provenance, and local naming, but it receives no symmetric metric source, quotient directed Cauchy object, selected profile limit, choice schedule, or ambient completeness claim.*

Proof. Apply Definition 2158.3. A one-sided source distance reaches completion-facing consumers only through the $\mathsf{QuasiMetric}^{\uparrow}$ carrier rows for source admission, ordered point names, directed distance, zero diagonal, forward triangle, directed balls, Cauchy rows, uniform reflection, transport, replay, provenance, and local naming.

Apply Definition 3947.4. Once that directed source route is displayed, the Lawvere completion read composes it with $\mathsf{YonedaCompletion}^{\uparrow}$ profiles, finite probe comparisons, $\mathsf{FormalBall}^{\uparrow}$ approximations, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal.

Apply Definition 3947.3. The resulting completion-facing route remains finite and displayed, so it cannot export a quotient directed Cauchy object, selected profile limit, symmetric completion collapse, host colimit, choice schedule, or ambient completeness witness. □

Theorem 2970.6 (Lawvere metric completion FormalBall universality route)**.** *Any completion-facing read of an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *profile must factor through the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row A after the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row Y and before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and* $\mathsf{Real}^{\uparrow}$ *seal row S. Thus the approximation gate cannot be replaced by quotient profile equality, symmetric completion collapse, selected profile limits, host enriched colimits, or ambient completeness.*

Proof. Project the carrier of Definition 3947.1. The profile route exposes $M, Y, F, D, A, T, R, S, H, C, P, N$, with A placed between the Yoneda-directed profile rows and the tolerance, readback, and seal rows.

Apply Definition 3947.2. It forces completion-facing consumers to inspect FormalBall approximation as one of the local NameCert obligations before the packet is treated as a completion neighbour.

Apply Definition 3947.3 and Definition 3947.4. Together they identify the public route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Since quotient profile equality, symmetric completion collapse, selected profile limits, host enriched colimits, and ambient completeness are absent carrier coordinates, no completion-facing read can substitute them for the FormalBall approximation gate. □

Theorem 2970.7 (Lawvere metric completion FormalBall profile initiality)**.** *Let $L = (M, Y, F, D, A, T, R, S, H, C,$ be an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet. Any completion-facing directed-profile read that reaches the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, or the* $\mathsf{Real}^{\uparrow}$ *sealing row S first factors through the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row A after the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row Y. Thus A is the initial completion-facing approximation gate after Yoneda profile data; no later tolerance, readback, or seal consumer may bypass it through quotient profile equality, selected profile limits, symmetric completion collapse, host enriched colimits, choice, or ambient completeness.*

Proof. Project the carrier of Definition 3947.1. The displayed order places Y before the finite probe and directed-profile rows, then places the FormalBall row A before the tolerance, readback, and seal rows T, R, S.

Apply Definition 3947.2. The NameCert obligations force every completion-facing consumer to inspect the directed source admission, YonedaCompletion handoff, finite probe comparison, directed profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming rows in the displayed carrier surface.

Apply Definition 3947.3 and Definition 3947.4. These theorems identify the finite route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the directed profile cannot leave the packet before the FormalBall row is read.

Finally apply Definition 3947.6. It names A as the mandatory approximation row between Y and the later tolerance, readback, and seal rows. The carrier has no coordinates for quotient profile equality, selected profile limits, symmetric completion collapse, host enriched colimits, choice, or ambient completeness, so none of those objects can serve as an earlier completion-facing gate. □

Falsifiable prediction: Removing any one of the $\mathsf{YonedaCompletion}^{\uparrow}$ handoff, finite probe-comparison row, $\mathsf{FormalBall}^{\uparrow}$ approximation row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, or local $\mathsf{NameCert}$ row makes

Definition 3947.3 false: a completion consumer could no longer verify the displayed directed-profile route.

Independence witness: yonedacompletion, metriccompletion, formalball, dyadicratcore, regseqrat, real: $\mathsf{LawvereMetricCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries directed source distances, Yoneda profile handoff, finite probe comparisons, formal-ball approximations, dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LawvereMetricCompletion}^{\uparrow}$ packet is generated from directed QuasiMetricUp source data, YonedaCompletionUp profile handoff, finite probe comparisons, directed distance-profile rows, FormalBallUp approximations, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback windows, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3947.1, Definition 3947.2, and Definition 3947.3 seed the finite Lawvere-enriched completion route over directed profile, approximation, tolerance, readback, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected Cauchy weights, quotient profile equality, symmetric metric completion collapse, host enriched colimits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed profile admission, probe comparison, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LawvereMetricCompletionUp.

The $\mathsf{LawvereMetricCompletion}^{\uparrow}$ directed-profile route consumes symmetric completion data only after the one-sided profile has already passed through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, and $\mathsf{FormalBall}^{\uparrow}$. The metric-completion dense embedding of chapter 3351 and the dense-reflector route of chapter 3786 are therefore horizontal consumers of the displayed profile surface, not replacements for the directed source row.

Proposition 2970.8 (Lawvere metric completion dense-reflector handoff)**.** *Any completion-facing read that compares an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet with* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *or* $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ *must first display the* $\mathsf{QuasiMetric}^{\uparrow}$ *source row,* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row,* $\mathsf{FormalBall}^{\uparrow}$ *approximation row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row from Definition 3947.4. Only after that directed profile is visible may the read consume the symmetric dense-embedding or Cauchy dense-reflector rows. The handoff exports no selected profile limit, quotient directed Cauchy object, symmetric-collapse equality, countable-choice schedule, or ambient completeness theorem.*

Proof. Start with the directed-profile composition theorem. It fixes the visible route through the QuasiMetric source, Yoneda profile, FormalBall approximation, dyadic tolerance ledger, regular rational readback, and Real seal before any completion-facing consumer can inspect the packet.

Next read the sibling dense-completion surfaces. $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ supplies the symmetric dense-image row for ordinary metric completion, while $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ supplies the Cauchy dense-reflector route through that dense image. Both are consumed after the Lawvere profile has exposed its directed rows.

Thus the comparison is a handoff from the directed profile into the existing completion infrastructure. Since the Lawvere carrier has no coordinate for selected profile limits, quotient directed Cauchy objects, symmetric-collapse equality, countable choice, or ambient completeness, none of those objects is exported by the handoff. □

Theorem 2970.9 (Lawvere metric completion Cauchy route)**.** *A symmetric dense-reflector comparison for an accepted* LawvereMetricCompletion$^{\uparrow}$ *packet reads* MetricCompletionDenseEmbedding$^{\uparrow}$ *and* CauchyCompletionDenseReflector$^{\uparrow}$ *only after the Lawvere one-sided route has displayed* QuasiMetric$^{\uparrow}$*,* YonedaCompletion$^{\uparrow}$*,* FormalBall$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. The comparison with chapter 3351 and chapter 3786 is therefore a horizontal lattice edge out of the directed profile route, not a replacement for any source, tolerance, regular-rational, or Real-seal row.*

Proof. Start from Definition 3947.4. It displays the one-sided Lawvere route through the quasi-metric source, Yoneda profile, formal-ball approximation, dyadic tolerance ledger, regular rational readback, and Real seal.

Apply the dense-reflector handoff proposition above. It allows the symmetric dense-embedding and Cauchy dense-reflector rows to be consumed only after that directed profile is already visible.

The two sibling chapters then provide the horizontal consumers: MetricCompletionDenseEmbedding$^{\uparrow}$ reads the symmetric dense image, and CauchyCompletionDenseReflector$^{\uparrow}$ reads the Cauchy reflector through that dense image. Since the carrier lacks selected profile limits, quotient directed Cauchy objects, symmetric-collapse equality, choice, and ambient completeness, the route remains the displayed lattice edge. □

Theorem 2970.10 (Lawvere metric-completion sibling lattice route)**.** *Any symmetric metric-completion consumer of an accepted* LawvereMetricCompletion$^{\uparrow}$ *packet must first pass through the directed route displayed after* QuasiMetric$^{\uparrow}$*,* YonedaCompletion$^{\uparrow}$*,* FormalBall$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. It may then compare that route with* MetricCompletionDenseEmbedding$^{\uparrow}$ *and* CauchyCompletionDenseRefl *but the comparison does not replace the directed source row or the one-sided profile classifier.*

Proof. Use the namecert construction route of chapter 3947. The packet names completion through the directed QuasiMetric$^{\uparrow}$, YonedaCompletion$^{\uparrow}$, and FormalBall$^{\uparrow}$ rows before any rational readback or terminal Real seal is available.

The symmetric siblings are therefore downstream comparison surfaces. The dense embedding of chapter 3351 and the dense-reflector route of chapter 3786 may inspect the same completed-window data, but neither sibling supplies a new directed source row.

Thus the horizontal lattice edge is substantive only after the Lawvere route has been read, and it preserves the one-sided profile classifier of the original packet. □

HyperbolicCrossRatio$^{\uparrow}$ routes four-boundary comparison through chapter 5110, chapter 6237, and the visual-metric boundary route before exposing cross-ratio comparisons.

Chapter 2971

A Concrete Naming Certificate for HyperbolicCrossRatio$^{\uparrow}$

The HyperbolicCrossRatio$^{\uparrow}$ packet realizes the four-boundary-point comparison branch of chapter 9463 as finite NameCert data. It sits beside the Poincare disk phase route of chapter 3455, the hyperbolic metric surface of chapter 3474, the boundary-fixed transport surface of chapter 3500, the Busemann boundary packet of chapter 3732, and the Schwarz–Pick disk-contraction row of chapter 3682. The packet names a local four-point invariant over displayed disk-boundary rows; it does not import a host ideal-boundary compactification, a global hyperbolic plane, selected geodesics, or quotient boundary equality.

Definition 2971.1 (Hyperbolic cross-ratio carrier)**.** *A* HyperbolicCrossRatio$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* PoincareDiskPhaseTransport$^{\uparrow}$ *disk-source and phase-transport row for the visible disk coordinate system.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *comparison row attached to that disk source.*
3. *B_0, B_1, B_2, B_3 are four boundary-point rows, each read through the displayed Poincare boundary interface rather than through a host compactification.*
4. *T is the* PoincareBoundaryFixedTransport$^{\uparrow}$ *row proving that the four boundary rows remain attached to the same boundary-fixed disk transport.*
5. *Q is the finite cross-ratio comparison row, exposing only the retained pairwise boundary-distance and oriented separation comparisons.*
6. *R is the refusal row excluding host global geometry, selected geodesics, quotient boundary equality, and object-level ideal-boundary compactification.*
7. *F is the* SchwarzPickDiskContraction$^{\uparrow}$ *provenance row that records nonexpansive disk-map compatibility for consumers comparing the cross-ratio under disk transport.*
8. *H records componentwise* hsame *transport for disk, metric, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, Schwarz–Pick provenance, package provenance, and local naming.*

9. *C records displayed* Cont *replay from disk phase transport through metric comparison, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, and provenance.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N$. It has no coordinate for fewer than four boundary rows, a host global metric oracle, selected ideal points outside the displayed boundary interface, quotient boundary equality, or unrecorded disk transport.

Theorem 2971.2 (Hyperbolic cross-ratio NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *carrier $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}$ *surface consists exactly of Poincare disk phase transport, hyperbolic-metric comparison, four displayed boundary rows, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, Schwarz–Pick provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer must inspect those rows before treating the packet as a hyperbolic four-point invariant.*

Proof. Project the carrier of Definition 3948.1. The disk and metric rows D, M fix the Poincare disk reading and the hyperbolic comparison surface.

The four boundary rows B_0, B_1, B_2, B_3 are then read together with the boundary-fixed transport row T. This is the only point at which the carrier exposes boundary points, and all four rows must belong to the same displayed disk-transport route.

The comparison and refusal route is Q, R, F. The finite cross-ratio row records the retained pairwise boundary comparisons, the refusal row prevents replacement by host global geometry, and the Schwarz–Pick row records the nonexpansive disk-map provenance consumed by downstream transport comparisons.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no other row is present, no consumer can omit a boundary row, replace hyperbolic comparison by a global oracle, or bypass the local naming route. □

Theorem 2971.3 (Hyperbolic cross-ratio nonescape)**.** *Every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{PoincareDiskPhaseTransport}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{PoincareBoundaryFixedTransport}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow}$$

The route fixes the disk phase surface, the hyperbolic metric comparison, four boundary rows, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, Schwarz–Pick provenance, transport, replay, package provenance, and local naming. It exports no host global geometry, quotient boundary equality, selected ideal-boundary compactification, or unrecorded disk-map invariant.

Proof. Let $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$ be accepted. A consumer first reads D, M, so all later comparisons are attached to the displayed disk phase transport and hyperbolic metric rows.

Next the consumer reads B_0, B_1, B_2, B_3, T. The four boundary rows are simultaneous carrier fields, and the boundary-fixed transport row keeps them in the same disk-transport context.

The comparison row Q is accepted only together with the refusal row R and Schwarz–Pick provenance F. Thus cross-ratio comparison remains finite and local to the displayed disk route.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. The excluded host and quotient objects have no carrier coordinate, so the consumer cannot escape the finite packet. □

Theorem 2971.4 (Hyperbolic cross-ratio Mobius consumer route)**.** *Any* $\mathsf{MobiusTransform}^{\uparrow}$ *consumer of an accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *packet must read the disk row D, metric row M, four boundary rows B_0, B_1, B_2, B_3, boundary-fixed transport row T, cross-ratio row Q, refusal row R, Schwarz–Pick provenance row F, and structural rows H, C, P, N before it may claim cross-ratio preservation. The consumer route of chapter 5110 has no accepted preservation coordinate outside this four-boundary packet.*

Proof. Apply Definition 3948.3. Every accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer already factors through the disk, metric, boundary, transport, comparison, refusal, provenance, replay, package, and naming rows of the finite carrier.

A $\mathsf{MobiusTransform}^{\uparrow}$ preservation read is a downstream consumer of the same packet, not a separate host-level geometric claim. Since the carrier has no coordinate for cross-ratio preservation unless the four displayed boundary rows and the boundary-fixed transport route are present, the Mobius consumer must enter through $D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N$. □

Theorem 2971.5 (Hyperbolic cross-ratio horofunction consumer route)**.** *Any* $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ *consumer of an accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *packet must first read the disk row D, hyperbolic metric row M, four boundary rows B_0, B_1, B_2, B_3, boundary-fixed transport row T, cross-ratio row Q, refusal row R, Schwarz–Pick provenance row F, and structural rows H, C, P, N before treating the packet as a boundary metric comparison. The consumer route reads boundary metric data through Busemann and visual-boundary rows, not through a host compactification.*

Proof. Apply Definition 3948.3. Every accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer already enters through the disk, metric, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, provenance, replay, package, and naming rows.

A $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ read is boundary-metric facing, so it cannot treat Q as an isolated scalar comparison. The four boundary rows and the metric row must first be routed through the displayed Busemann and visual-boundary surfaces that the horofunction-boundary packet consumes.

The refusal row R blocks host compactification and quotient ideal-boundary equality. The structural rows H, C, P, N then preserve the same finite route, so the horofunction consumer receives only the cross-ratio data that has passed through the displayed boundary metric rows. □

Falsifiable prediction: Any accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer omitting one of the four boundary rows, the $\mathsf{HyperbolicMetric}^{\uparrow}$ comparison, the $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ row, Pkg provenance, or the local $\mathsf{NameCert}$ row refutes Definition 3948.3.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, poincare-boundary-fixed-transport, hyperbolic-busemann-boundary, schwarz-pick-disk-contraction: $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk phase transport, hyperbolic metric comparison, four boundary rows, boundary-fixed transport, finite cross-ratio comparison, refusal of host global geometry, Schwarz–Pick provenance, transport, replay, package provenance, and local naming.

The Lean TasteGate file for HyperbolicCrossRatio$^{\uparrow}$ will provide explicit Nontrivial HyperbolicCrossRatio$^{\uparrow}$ and FieldFaithful HyperbolicCrossRatio$^{\uparrow}$ instances. These instances keep the disk, metric, boundary, transport, comparison, refusal, provenance, replay, package, and local naming rows distinguishable.

Closure status for NameCert$_{\text{HyperbolicCrossRatio}^{\uparrow}}$.

Constructive story: A HyperbolicCrossRatio$^{\uparrow}$ packet is generated from PoincareDiskPhaseTransportUp disk data, HyperbolicMetricUp comparison, four displayed boundary rows, PoincareBoundaryFixedTransportUp transport, finite cross-ratio comparison, global-geometry refusal, SchwarzPickDiskContractionUp provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicCrossRatio}^{\uparrow}}$, seed)

Scope closed: Definition 3948.1, Definition 3948.2, Definition 3948.3, and Definition 3948.4 seed the finite four-boundary-point hyperbolic cross-ratio route over disk, metric, boundary, transport, refusal, provenance, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicCrossRatio}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host global geometry, quotient boundary equality, selected ideal-boundary compactification, unrecorded disk-map invariants, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four boundary rows, hyperbolic metric comparison, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, TasteGate nontriviality, field faithfulness, provenance, transport, replay, and local naming for BEDC.Derived.HyperbolicCrossRatioUp.

Theorem 2971.6 (Hyperbolic cross-ratio Mobius-invariant sibling route)**.** *Every accepted* HyperbolicCrossRatio$^{\uparrow}$ *four-boundary-point carrier consumes the* MobiusTransform$^{\uparrow}$ *route of chapter 5110 and the* HyperbolicMobiusInvarian *route of chapter 6237 before any cross-ratio comparison is exported. In carrier notation* $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$*, the accepted comparison row* Q *is public only after the disk, metric, four boundary, boundary-fixed transport, refusal, provenance, replay, package, and naming rows have passed the sibling Mobius and invariant routes.*

Proof. Apply Definition 3948.3. Every public read of Q already factors through the displayed disk, metric, four boundary rows, boundary-fixed transport, refusal, Schwarz–Pick provenance, transport, replay, provenance, and local naming rows.

Apply Definition 3948.4. A MobiusTransform$^{\uparrow}$ consumer of that packet may claim preservation only through the same four-boundary carrier route, so the Mobius sibling is a prerequisite route rather than an external host symmetry.

The HyperbolicMobiusInvariant$^{\uparrow}$ sibling then reads the same Mobius-facing route as invariant data. Since X has no coordinate for an isolated cross-ratio scalar, quotient boundary equality, or unrecorded transformation invariant, any accepted cross-ratio comparison exported from Q has already consumed both sibling routes. □

Theorem 2971.7 (Hyperbolic cross-ratio logarithm route)**.** *Every accepted* HyperbolicCrossRatio$^{\uparrow}$ *read that exports a logarithmic distance or invariant must first display the four endpoint rows* B_0, B_1, B_2, B_3*, the cross-ratio comparison ledger* Q*, positive-domain admission, the* RealLogarithm$^{\uparrow}$ *inverse route of chapter 2315, the* RealExponential$^{\uparrow}$ *comparison row,* RealAbs$^{\uparrow}$ *error control, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The logarithm is a sibling dependency read through the displayed RealLogarithm certificate; it is not a host logarithm primitive and it exports no quotient boundary equality, selected ideal endpoint, host global geometry, or unrecorded disk-map invariant.*

Proof. Project the carrier of Definition 3948.1. The endpoint rows B_0, B_1, B_2, B_3 and the cross-ratio comparison row Q are already displayed before any downstream invariant can be read.

The logarithmic route is then admitted only as a positive-domain read. Invoke the RealLogarithm$^{\uparrow}$ certificate of chapter 2315: its inverse route passes through the RealExponential$^{\uparrow}$ comparison row and the RealAbs$^{\uparrow}$ error-control row before a logarithm endpoint is consumed.

Finally transport and replay the same carrier route through H, C, P, N. The sibling logarithm certificate supplies the inverse-route dependency, while the $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ carrier supplies the four endpoint rows and comparison ledger. Neither surface contains a host logarithm, quotient boundary equality, selected ideal endpoint, global geometry oracle, or unrecorded disk-map invariant. □

$\mathsf{HyperbolicCrossRatio}^{\uparrow}$ reads the boundary visual metric surface of chapter 3866 before any cross-ratio comparison is consumed as a boundary-distance statement. This route keeps Mobius-invariant comparison, logarithmic readback, visual boundary distance, transport, replay, provenance, and local naming in the displayed hyperbolic boundary ledger.

Chapter 2972

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, Kelley on filters, and Mac Lane on functorial transport, the $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ packet records the finite route that transports regular Cauchy filter data into stream-indexed real seals. It reads the regular filter surface of chapter 1445, the Cauchy filter basis and saturation surfaces of chapter 1501 and chapter 333, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-functor route of chapter 423. The packet names the transport surface only; it does not quotient filters, select limits, identify host real objects, or use ambient completeness.

Definition 2972.1 (RegularCauchyFilterFunctor carrier)**.** *A* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, B, W, D, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the regular Cauchy filter index row, reading the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *source packet and its displayed filter-indexed tail data.*
2. *B is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, including the finite basis and saturation route used before any stream window is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the requested windows obtained from the filter-indexed source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after W and D are displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the filter, window, tolerance, and readback rows have been replayed.*
7. *A is the functorial action ledger, recording source-map action on displayed filter indices and target-map action on the induced stream windows, readbacks, tolerances, and seals.*
8. *H records componentwise* hsame *transport for I, B, W, D, R, E, A and for the structural rows below.*

9. *C records displayed* Cont *replay for the route from regular Cauchy filter indices through CauchyFilter basis data, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, and action ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, W, D, R, E, A, H, C, P, N$. There is no coordinate for quotient filters, selected representatives, host equality of filters, host equality of reals, countable choice, or detached completion points.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterFunctorUp.RegularCauchyFilterFunctorTasteGate single carrier alignment`

Theorem 2972.2 (RegularCauchyFilterFunctor NameCert obligations)**.** *For every accepted* RegularCauchyFilterFu *carrier*

$$F = (I, B, W, D, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; regular Cauchy filter source exposure through I; CauchyFilter basis and saturation exposure through B; finite-window exposure through W; dyadic tolerance exactness through D; regular-rational readback through R; Real seal handoff through E; functorial action bookkeeping through A; and structural transport, replay, provenance, and local naming through H, C, P, N. No obligation is discharged by quotienting filters, choosing representatives, identifying host real objects, or importing ambient completeness.*

Proof. Project the carrier of Definition 3949.1. The accepted packet displays exactly the source index row, basis row, window row, tolerance row, readback row, seal row, action ledger, and structural rows.

The source side is exhausted by I and B. The regular Cauchy filter row supplies the filter-indexed tail data, while the CauchyFilter basis row supplies the finite basis and saturation surface before any stream-indexed consumer can inspect the packet.

The target side is exhausted by W, D, R, E. The finite windows are read first, the dyadic tolerance ledger is attached to those windows, the RegSeqRat row reads back the regular rational data, and the Real seal remains downstream from the displayed route.

The action and structural rows do not add hidden objects. The ledger A records how displayed source-map action is replayed as target-map action on W, D, R, E, while H, C, P, N transport, replay, package, and name the same finite surface. Hence the listed rows are the whole local NameCert obligation surface. □

Theorem 2972.3 (RegularCauchyFilterFunctor filter-to-stream route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads I, then B, then W and D, then R, then E, with the functorial action ledger A and the structural rows H, C, P, N preserving this order. No second real-facing route is available inside the carrier.

Proof. Start with the Real-facing coordinate. The row E is not detached from the packet; it is stored only after the source index, filter basis, finite windows, dyadic tolerances, and regular-rational readback have been displayed.

Read the route in order. The source packet contributes I, the CauchyFilter basis and saturation data contribute B, the stream-indexed finite observations contribute W, the tolerance checks contribute D, and the regular-rational readback contributes R.

Now inspect the action ledger. The row A records that source-map action on filter indices is replayed on the displayed window, tolerance, readback, and seal rows, so the functorial transport does not introduce a selected limit or a host equality principle.

Finally the transport, replay, provenance, and local naming rows certify the same route. Since the carrier has no other Real seal coordinate and no quotient-filter coordinate, every Real-facing consumer factors through the listed finite chain. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ read is refuted if a RealUp seal is reached without displaying the RegularCauchyFilter, CauchyFilter, StreamName, DyadicRatCore, RegSeqRat, functorial action, transport, replay, provenance, and local naming rows.

Independence witness: regularcauchyfilter, cauchyfilter, streamname, regseqrat, dyadicratcore, real, and cauchycompletionfunctor: none is bijective to the functor packet because only $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ records filter-index source action and stream-index target action together with one Real seal route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ packet is generated from a regular Cauchy filter index row, CauchyFilter basis row, StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, RealUp seal row, functorial action ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3949.1, Definition 3949.2, and Definition 3949.3 seed the finite filter-indexed route from RegularCauchyFilter and CauchyFilter data through StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected representatives, host equality of filters, host equality of reals, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-index source exposure, CauchyFilter basis exposure, window and tolerance routing, RegSeqRat readback, Real seal handoff, functorial action, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterFunctorUp.

Chapter 2973

A Concrete Naming Certificate for ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$

Following Arzela, Ascoli, and Bishop–Bridges constructive compactness, the ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$ packet records the finite extraction surface where equicontinuity and compact finite nets are read before a sequence-window consumer reaches a real seal. It sits downstream of the finite interval-net route of chapter 3471, the modulus composition route of chapter 3784, the compact uniform modulus extractor of chapter 2202, the convergent-sequence boundary of chapter 3944, the located Cauchy subsequence route of chapter 3945, the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the real seal of chapter 38. The chapter names only the finite equicontinuous extraction packet; it does not assert open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real$^{\uparrow}$ completeness.

Definition 2973.1 (Arzela–Ascoli finite equicontinuous carrier)**.** *An* ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, E, U, M, D, W, Q, L, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net source row, read through the* HeineBorelFiniteNet$^{\uparrow}$ *and compact-metric finite-net data already visible to the packet.*
2. *E is the finite equicontinuity-modulus row, assigning each displayed target tolerance a displayed source radius uniformly over the inspected family.*
3. *U is the* UniformModulusComposition$^{\uparrow}$ *and* CompactUniformModulusExtractor$^{\uparrow}$ *compatibility row, recording that the compact finite-net and equicontinuity ledgers use the same tolerance requests.*
4. *M is the diagonal or finite-window subsequence row, exposed only as the finite window data consumed by* RealConvergentSequence$^{\uparrow}$ *and* LocatedCauchySubsequence$^{\uparrow}$ *routes.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for source radii, target tolerances, mesh slack, and comparison budgets.*
6. *W is the finite* StreamName$^{\uparrow}$ *window row naming the inspected source points, function-family indices, and target observations.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the rational observations on W.*

8. *L is the located convergence-window row used by the* LocatedCauchySubsequence$^{\uparrow}$ *consumer.*
9. *S is the terminal* Real$^{\uparrow}$ *seal row, available only after K, E, U, M, D, W, Q, L have been displayed.*
10. *H records componentwise* hsame *transport for all source, modulus, window, readback, located, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a finite extraction request through $K, E, U, M, D, W, Q, L, S$.*
12. *P is the* Pkg *provenance over* ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$, HeineBorelFiniteNet$^{\uparrow}$, UniformModulusCompositi CompactUniformModulusExtractor$^{\uparrow}$, RealConvergentSequence$^{\uparrow}$, LocatedCauchySubsequence$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, U, M, D, W, Q, L, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteEquicontinuousUp.ArzelaAscoliFiniteEquicontinuousTasteGate`
`single carrier alignment`

Theorem 2973.2 (Arzela–Ascoli finite equicontinuous obligations)**.** *For every accepted* ArzelaAscoliFiniteEquiconti *carrier*

$$A = (K, E, U, M, D, W, Q, L, S, H, C, P, N),$$

the local NameCert$_{\text{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite-net row, equicontinuity modulus row, compact-modulus compatibility row, finite diagonal window, dyadic tolerance ledger, stream windows, rational readback, located convergence window, real seal, transport, replay, provenance, and local naming rows;*
2. *compact-net exposure, requiring every source point used by the extraction to be read through the displayed* HeineBorelFiniteNet$^{\uparrow}$ *finite-net data before any target observation is consumed;*
3. *equicontinuity visibility, requiring the shared source-radius row E and compatibility row U to be displayed before the diagonal window M can be read;*
4. *L10 readback order, requiring the dyadic tolerance ledger, stream windows, regular rational readback, located Cauchy window, and terminal Real seal to appear in the order D, W, Q, L, S;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only these finite rows and to export no open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3950.1. The packet exposes the compact finite-net source, equicontinuity modulus, compact-modulus compatibility row, finite diagonal window, tolerance ledger, stream windows, rational readback, located convergence row, real seal, and structural rows as one finite BHist object.

Compact-net exposure is the row K. A consumer must first read the finite net and coverage witnesses supplied by the Heine–Borel finite-net route, so the source compactness input is a displayed finite object rather than an arbitrary open cover.

Equicontinuity visibility is supplied by E and U. The requested target tolerance is converted into a displayed source radius, and the compact uniform modulus row records that the same tolerance budget is used by the finite-net and diagonal-window reads.

The L10 readback order is then forced by the packet coordinates. The diagonal window M can be inspected only through D, W, Q, L, and the terminal row S is downstream of those dyadic, stream, rational, and located Cauchy rows.

Finally, H, C, P, N preserve that row inventory. Since the carrier has no coordinate for open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real completeness, none of those objects can be exported by the local naming surface. □

Falsifiable prediction: Every $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ consumer read that reaches $\mathsf{Real}^{\uparrow}$ first displays compact finite-net rows, equicontinuity finite modulus rows, compact-modulus compatibility rows, a finite diagonal window, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, and located Cauchy window rows.

Independence witness: heineborelfinet, uniformmoduluscomposition, compactuniformmodulusextractor, realconvergentsequence, locatedcauchysubsequence, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ because this packet jointly carries compact finite-net exposure, equicontinuity-radius visibility, compact-modulus compatibility, diagonal-window data, dyadic tolerances, stream windows, rational readback, located convergence rows, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ packet is generated from compact finite-net rows, equicontinuity finite modulus rows, compact-modulus compatibility, a finite diagonal or subsequence window, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, LocatedCauchySubsequence rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3950.1 and Definition 3950.2 seed the finite equicontinuous extraction packet consumed by L10 compact-continuous and Real-completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, equicontinuity visibility, L10 readback order, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ArzelaAscoliFiniteEquicontinuousUp.

Chapter 2974

A Concrete Naming Certificate for $\mathsf{PoissonSummation}^{\uparrow}$

Following Poisson and Fourier analysis, the $\mathsf{PoissonSummation}^{\uparrow}$ packet is a finite BEDC route from lattice-indexed source windows through Fourier finite-observation rows and distribution-facing summation rows. It sits over the finite coefficient interface of chapter 124, the lattice source of chapter 55, the pushforward and test-row boundary of chapter 186, and the terminal real readback of chapter 38. The packet names a finite summation certificate; it does not import analytic convergence, host function equality, Schwartz-space quotients, Haar measure, or an infinite series oracle.

Definition 2974.1 (Poisson summation finite carrier)**.** *A* $\mathsf{PoissonSummation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, F, D, S, R, H, C, P, N)$$

with the following displayed coordinates.

1. *L is the* $\mathsf{Lattice}^{\uparrow}$ *row naming the finite lattice index window, dual-window tags, and shift ledger used by the summation read.*
2. *F is the* $\mathsf{Fourier}^{\uparrow}$ *row naming the finite coefficient and frequency observations made over the displayed lattice window.*
3. *D is the* $\mathsf{Distribution}^{\uparrow}$ *and test-function row through which the finite Fourier observations are read as distribution-facing data.*
4. *S is the finite summation-window row, recording the source-side and dual-side partial sums inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback surface for the displayed summation comparison.*
6. *H records componentwise* hsame *transport for lattice, Fourier, distribution, summation-window, and real-readback rows.*
7. *C records displayed* Cont *replay from lattice windows through Fourier observations, distribution rows, finite summation, and readback.*
8. *P is the* Pkg *provenance over* $\mathsf{PoissonSummation}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for analytic convergence, equality of host functions, an infinite summation theorem, Schwartz-space quotient equality, Haar measure, or a selected distribution representative.

Lean4 ✓ `BEDC.Derived.PoissonSummationUp.PoissonSummationTasteGate single carrier alignment`

Theorem 2974.2 (Poisson summation NameCert obligations)**.** *For every accepted* $\mathsf{PoissonSummation}^{\uparrow}$ *carrier* $Q = (L, F, D, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$ *surface consists exactly of carrier admission for* Q*, lattice-window admission through* L*, finite Fourier observation through* F*, distribution-facing test-row admission through* D*, finite source and dual summation-window comparison through* S*,* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.PoissonSummationUp.PoissonSummationNamecertObligations`

Proof. Project the carrier of Definition 3951.1. The analytic surface is exhausted by the lattice row, Fourier row, distribution-facing row, summation-window row, and readback row.

Read L, F, D, S in order. The lattice window determines the finite source and dual indices, the Fourier row records only the inspected frequency observations, the distribution row supplies the test-facing interpretation, and the summation row compares the two displayed finite windows.

The rows R, H, C, P, N then seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for convergence, host function equality, quotient spaces, Haar measure, or selected representatives, none of those objects is part of the local NameCert surface. □

Theorem 2974.3 (Poisson summation Fourier-lattice route)**.** *Every accepted* $\mathsf{PoissonSummation}^{\uparrow}$ *read factors through the displayed route*

$$\mathsf{Lattice}^{\uparrow} \longmapsto \mathsf{Fourier}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto S \longmapsto \mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}.$$

Thus a downstream consumer receives only finite lattice windows, finite Fourier observations, distribution-facing test rows, finite summation windows, real readback, transport, replay, provenance, and local naming.

Proof. Begin with the lattice coordinate L of Definition 3951.1. It supplies the only source of summation indices and dual-window tags.

Apply the Fourier and distribution coordinates F and D. They expose finite coefficient observations and test-facing rows before the summation window S can be compared.

Finally pass through R, H, C, P, N. The regular and real readback rows seal the finite comparison, while transport, replay, provenance, and naming preserve the route without adding analytic convergence or host equality. □

Falsifiable prediction: Every accepted $\mathsf{PoissonSummation}^{\uparrow}$ read factors through lattice windows, Fourier finite observations, distribution-facing test rows, finite summation windows, $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Independence witness: lattice, fourier, distribution, real: $\mathsf{PoissonSummation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lattice windows, Fourier observations, distribution-facing test rows, finite source and dual summation windows, readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$**.**

Constructive story: A PoissonSummation$^{\uparrow}$ packet is generated from a LatticeUp window row, a FourierUp finite-observation row, a DistributionUp test-facing row, a finite summation-window comparison, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PoissonSummation}^{\uparrow}}$, seed)

Scope closed: Definition 3951.1, Definition 3951.2, and Definition 3951.3 seed the finite Fourier-lattice summation packet over lattice, Fourier, distribution, and real-readback rows.

Formal verification: FormalStatus(NameCert$_{\text{PoissonSummation}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.PoissonSummationUp.PoissonSummationTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim analytic convergence, host function equality, Schwartz-space quotients, Haar measure, infinite summation theorems, or bridge checking.

Upgrade path: A formal route can prove the NameCert obligation surface and Fourier-lattice route theorem.

Chapter 2975

A Concrete Naming Certificate for $\mathsf{RealDecimalNormalForm}^{\uparrow}$

Following Bishop and Bridges on constructive real approximants and the classical decimal expansion normalization convention, the $\mathsf{RealDecimalNormalForm}^{\uparrow}$ packet records the finite readback surface by which a displayed $\mathsf{Real}^{\uparrow}$ name is presented through decimal prefixes. It sits after the decimal-prefix route of chapter 1417, the integer bracket route of chapter 1479, the rational naming surface of chapter 37, the dyadic tolerance surface of chapter 291, the regular-rational handoff of chapter 301, and the terminal seal discipline of chapter 38. The packet names a canonical decimal readback route only; it does not choose a quotient representative, compare host real streams, or import ambient real completeness.

Definition 2975.1 (RealDecimalNormalForm carrier)**.** *A* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, I, Q, Y, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *prefix ledger, including finite digit rows and decimal-place windows.*
2. *I is the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row that fixes the integer-part surface used by the decimal readback.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *place-value row that evaluates each inspected decimal prefix as a rational expression.*
4. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same prefix and bracket data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained after the decimal, bracket, rational, and dyadic rows are displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row whose name is read only after D, I, Q, Y, R have been exposed.*
7. *H records componentwise* hsame *transport for decimal-prefix, integer-part, rational, dyadic, regular-rational, and real-seal rows.*
8. *C records displayed* Cont *replay for the route from decimal prefix windows through integer brackets, place values, tolerances, regular readback, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealDecimalNormalForm}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$ *row.*

The classifier for the packet compares accepted decimal normal-form reads only through $D, I, Q, Y, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, trichotomy over arbitrary real names, host decimal reals, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealDecimalNormalFormUp.RealDecimalNormalFormTasteGate single carrier alignment`

Theorem 2975.2 (RealDecimalNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *carrier*

$$M = (D, I, Q, Y, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed decimal-prefix, integer-bracket, rational place-value, dyadic tolerance, regular-rational handoff, real-seal, transport, replay, provenance, and local naming rows;*
2. *digit legality and place-value exactness, requiring every accepted digit row in D to be a decimal digit and every inspected prefix to be evaluated through the rational row Q;*
3. *endpoint normalization, requiring the ledger to exclude the eventual-nine route before a decimal read is treated as normal-form data;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be reached only after the decimal, integer-bracket, rational, and dyadic rows have been displayed;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring every real-facing consumer to read E through the same $D, I, Q, Y, R, H, C, P, N$ route.*

No obligation is discharged by quotienting streams, selecting endpoint representatives, deciding trichotomy for all real names, or using ambient real completeness.

Lean4 ✓ `BEDC.Derived.RealDecimalNormalFormUp.RealDecimalNormalFormCarrier namecert obligation surface`

Proof. Project the carrier of Definition 3952.1. The displayed rows $D, I, Q, Y, R, E, H, C, P, N$ are the whole packet, so carrier admission is rowwise.

Digit legality and place-value exactness are read from D and Q. The decimal-prefix ledger supplies only finite digit rows and windows, while the rational row Q evaluates those windows before any dyadic or real-facing consumer can inspect them.

Endpoint normalization is local to the same finite ledger. The non-eventual-nine condition is recorded before the regular-rational handoff, so the packet does not identify two host decimal streams or select an endpoint representative.

The handoff and seal clauses follow the order of the remaining coordinates. The dyadic row Y is attached to the same prefix and bracket data, the regular-rational row R is downstream of D, I, Q, Y, and the terminal real seal E is downstream of R with H, C, P, N preserving the route. □

Theorem 2975.3 (RealDecimalNormalForm Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{ArchimedeanIntegerPart}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads decimal prefix rows, then the integer-part bracket rows, then the rational place-value rows, then dyadic tolerance rows, then the regular-rational handoff, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The packet supplies no quotient stream equality, selected representative, trichotomy row, host decimal real, or ambient completeness.*

Proof. Start with an accepted packet $M = (D, I, Q, Y, R, E, H, C, P, N)$. The decimal rows D are already finite prefix rows, and I supplies the integer bracket that fixes the readback surface before place-value evaluation is exposed.

Now read the arithmetic rows. The rational row Q evaluates the inspected prefixes, and the dyadic row Y translates the same values into the tolerance discipline used by regular-rational and real consumers.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R. It is listed only after the decimal, integer-bracket, rational, and dyadic rows, so no consumer can reach the regular-rational readback without displaying that route.

Finally read the terminal seal E and the structural rows H, C, P, N. These rows transport, replay, source, and name the same finite route, and none of them is a quotient equality, selected representative, trichotomy decision, host decimal real, or completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDecimalNormalForm}^{\uparrow}$ packet is generated from DecimalExpansion prefix rows, ArchimedeanIntegerPart bracket rows, Rat place-value rows, DyadicRatCore tolerance rows, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 3952.1, Definition 3952.2, and Definition 3952.3 seed the finite canonical decimal readback route from DecimalExpansion and ArchimedeanIntegerPart data through Rat, DyadicRatCore, RegSeqRat, and RealUp seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close uniqueness of all decimal expansions, quotient stream equality, selected representatives, trichotomy over arbitrary real names, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, digit legality, place-value exactness, endpoint-normalization ledger exhaustion, RegSeqRat handoff, and RealUp seal nonescape for BEDC.Derived.RealDecimalNormalFormUp.

Chapter 2976

A Concrete Naming Certificate for $\mathsf{CauchyModulusNormalizer}^{\uparrow}$

Following Bishop–Bridges constructive analysis with explicit Cauchy moduli, the $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ packet records the finite route by which a displayed Cauchy threshold schedule is normalized before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal consumers may read it. It sits next to the fast Cauchy criterion of chapter 3825, the monotone normalization surface of chapter 1744, the StreamName carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the Real seal of chapter 38. The packet is a finite normalizer certificate for Real and Completion routes; it is not Real completeness, quotient completion, quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or a standard bridge.

Definition 2976.1 (Cauchy modulus normalizer carrier)**.** *A* $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, M, F, D, W, Q, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source row, read through a* $\mathsf{FastCauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, or* $\mathsf{RegSeqRat}^{\uparrow}$ *Cauchy-window surface.*
2. *M is the input Cauchy modulus row, assigning displayed tail thresholds to requested dyadic tolerances.*
3. *F is the finite normalization table, recording prefix maxima or equivalent displayed threshold joins for the finitely many tighter requests visible to the packet.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for precision requests, slack conversions, and positivity checks consumed by M and F.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row available after the normalized threshold has been displayed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations exposed on W.*
7. *S is the normalized Cauchy-window seal row saying that the readback admitted after F is already admitted by the input modulus row M at the requested tolerance.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, M, F, D, W, Q, S have been displayed.*

9. *H records componentwise* hsame *transport for the source, input modulus, normalization table, dyadic ledger, windows, readback, Cauchy-window seal, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a precision request through X, M, D, F, W, Q, S, R.*

11. *P is the* Pkg *provenance over* CauchyModulusNormalizer$^{\uparrow}$, FastCauchyCriterion$^{\uparrow}$, CauchyModulusMonotoneNormal StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CauchyModulusNormalizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, F, D, W, Q, S, R, H, C, P, N$. It has no coordinate for Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or bridge checking.

Theorem 2976.2 (Cauchy modulus normalizer NameCert obligations)**.** *For every accepted* CauchyModulusNormali *carrier*

$$Z = (X, M, F, D, W, Q, S, R, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusNormalizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source, input modulus, finite normalization table, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Cauchy-window seal, Real seal, transport, replay, provenance, and local naming rows;*

2. *modulus-source exposure, requiring the source row X and input modulus row M to be displayed before the normalizer table F is read;*

3. *finite normalization exactness, requiring F to read only finitely many tolerance requests from D and threshold rows from M;*

4. *window-readback order, requiring StreamName windows, RegSeqRat readback, and the Real seal to appear downstream of the normalized threshold route as W, Q, S, R;*

5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed normalizer rows and to export no Real completeness, quotient completion, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or bridge datum.*

No Cauchy-modulus-normalizer obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyModulusNormalizerUp.CauchyModulusNormalizerTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3953.1. The accepted packet exposes the source row, input modulus, finite normalization table, dyadic tolerance ledger, stream windows, rational readback, Cauchy-window seal, Real seal, and structural rows as one finite carrier.

Modulus-source exposure is the order of the displayed packet. A consumer cannot inspect F until the source row X, input threshold schedule M, and dyadic tolerance ledger D have fixed the finite requests being normalized.

Finite normalization exactness is finite because F reads only the displayed tolerance requests in D and their corresponding thresholds in M. This is the same constructive modulus-normalization discipline used by the monotone normalization packet, but here it is packaged as the normalizer boundary immediately before downstream window reads.

The window-readback order follows from the coordinates W, Q, S, R. StreamName observations are read after the normalized threshold, RegSeqRat readback records the rational observations, the

Cauchy-window seal S checks preservation against M, and the terminal Real seal R is downstream of those rows.

Finally, H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or bridge data, none of those objects can become a local NameCert obligation. □

Falsifiable prediction: Every $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ consumer read factors through a source row, an input Cauchy modulus row, a finite normalization table, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a Cauchy-window seal, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fastcauchycriterion, cauchymodulusmonotonenormalization, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ because this packet jointly carries the source, input modulus, finite normalization table, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Cauchy-window preservation seal, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ packet is generated from a Cauchy-window source row, an input Cauchy modulus row, a finite normalization table, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a preservation seal, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3953.1 and Definition 3953.2 seed the finite modulus-normalizer packet used before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-source exposure, finite normalization exactness, window-readback order, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusNormalizerU

Chapter 2977

A Concrete Naming Certificate for $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$

The $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet realizes the total-host and universal-substrate separation of chapter 9370 as finite NameCert data. It sits between the neutral-spine comparison surface of chapter 3909, the bounded fuel trace of chapter 3889, the normal-form readback of chapter 936, the substrate trace boundary of chapter 944, and the total-host fuel discipline of chapter 964. The packet records the boundary at which host normalization evidence remains total while substrate universality remains fuel-indexed and external to the host evaluator.

Definition 2977.1 (MetaCIC strong-normalization boundary carrier)**.** *A* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, F, Z, U, T, R, H, C, P, N)$$

with the following displayed rows.

1. *S is the* $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ *row carrying the typed neutral-spine comparison route.*
2. *F is the* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *row carrying bounded normalizer fuel, endpoints, equality-check readback, and refusal data.*
3. *Z is the* $\mathsf{ZNormal}^{\uparrow}$ *row carrying normal-form and total-host readback evidence.*
4. *U is the* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *row carrying the fuel-indexed substrate trace.*
5. *T is the* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *row carrying total-host fuel discipline and step-indexed readback.*
6. *R is the refusal row stating that substrate universality is not re-exported as host nontermination or global conversion decidability.*
7. *H records componentwise* hsame *transport for spine, fuel, normal-form, substrate-trace, total-host, and refusal rows.*
8. *C records displayed* Cont *replay from total-host fuel through bounded trace and normal-form readback.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$, $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{ZNormal}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{StepIndexedTotalHost}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host nontermination, global conversion decidability, an unbounded substrate run, a selected normalization proof, semantic equality, or a collapse of substrate universality into the total host.

Lean4 ✓ `BEDC.Derived.MetaCICStrongNormalizationBoundaryUp.MetaCICStrongNormalizationBoundaryTasteGate single carrier alignment`

Theorem 2977.2 (MetaCIC strong-normalization boundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *carrier* $B = (S, F, Z, U, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, neutral-spine admission through* S*, fuel-trace admission through* F*, normal-form readback through* Z*, bounded substrate-trace admission through* U*, total-host fuel discipline through* T*, Turing-refusal exposure through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3954.1. The host-facing rows are S, F, Z, T, the substrate-facing row is U, and the boundary refusal is R.

Read the host rows first. The neutral-spine, fuel-trace, normal-form, and step-indexed total-host rows give only finite fuel, endpoint, and readback evidence.

Read the substrate row through U and then the refusal row R. Bounded substrate universality remains fuel-indexed, and the refusal row prevents that universality from becoming host nontermination or global conversion decidability.

The rows H, C, P, N transport, replay, source, and name the same boundary. Since the carrier contains no host nontermination, unbounded trace, selected normalization proof, semantic equality, or global conversion theorem, none of those objects is exported. □

Theorem 2977.3 (MetaCIC strong-normalization boundary Turing-refusal route)**.** *Every substrate-universality read of an accepted* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}, \mathsf{MetaCICFuelTrace}^{\uparrow}, \mathsf{ZNormal}^{\uparrow}, \mathsf{StepIndexedTotalHost}^{\uparrow} \longmapsto \mathsf{BoundedUniversalTrace}^{\uparrow}$$

Thus the packet preserves total-host normalization evidence while refusing to turn the bounded universal substrate trace into host nontermination, unbounded evaluation, or global conversion decidability.

Proof. Begin with the host-facing coordinates S, F, Z, T of Definition 3954.1. They expose only typed neutral-spine comparison, bounded normalizer fuel, normal-form readback, and step-indexed total-host rows.

Move to the substrate coordinate U. It supplies a bounded universal trace as a finite row, not an unbounded run and not a host evaluator.

Finally apply the refusal coordinate R and preserve it through H, C, P, N. The refusal row is part of the same carrier, so any read that keeps substrate universality must also keep the boundary that blocks host nontermination and global conversion claims. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet displays neutral-spine, finite fuel, normal-form, bounded universal trace, step-indexed total-host, refusal, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows before any consumer reads the total-host and universal-substrate boundary.

Independence witness: metacic-decidable-neutral-spine, metacic-fuel-trace, znormal, bounded-universal-trace, step-indexed-total-host: no listed sibling is bijective to $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ because this packet jointly carries neutral-spine comparison, finite fuel trace, normal-form read-back, bounded substrate trace, total-host fuel discipline, refusal, transport, replay, provenance, and local naming in one boundary carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet is generated from MetaCICDecidableNeutralSpineUp, MetaCICFuelTraceUp, ZNormalUp, BoundedUniversalTraceUp, and StepIndexedTotalHostUp rows, together with a Turing-refusal row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3954.1, Definition 3954.2, and Definition 3954.3 seed the finite MetaCIC boundary between total-host normalization evidence and bounded substrate universality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host nontermination, unbounded substrate evaluation, global conversion decidability, semantic equality, Lean verification, bridge checking, or a collapse of substrate universality into total-host computation.

Upgrade path: A formal route can encode the finite carrier, prove the NameCert obligation surface and Turing-refusal route, and add TasteGate Nontrivial and FieldFaithful instances.

Chapter 2978

A Concrete Naming Certificate for $\mathsf{CofinalFilter}^{\uparrow}$

Following Kelley and Bourbaki on cofinal filters, the $\mathsf{CofinalFilter}^{\uparrow}$ packet records the finite corridor by which a downstream filter or completion consumer reads a target filter through a displayed cofinal source route. It sits between the finite base surface of chapter 1601, the Cauchy-filter route of chapter 333, the refinement corridor of chapter 1626, the uniform-completion consumer of chapter 1473, and the metric-completion consumer of chapter 1506. The packet records cofinal comparison data without quotient filters, selected representatives, selected limits, ultrafilters, countable choice, Zorn arguments, or ambient completeness.

Definition 2978.1 (CofinalFilter carrier)**.** *A* $\mathsf{CofinalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, \rho, \lambda, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{FilterBase}^{\uparrow}$ *row whose displayed packets answer the downstream request after cofinal replacement.*
2. *T is the target* $\mathsf{FilterBase}^{\uparrow}$ *row named by the consumer before the cofinal route is read.*
3. *ρ is the cofinal map row, sending every displayed target request to a source packet contained in that target request.*
4. *λ is the reverse ledger when route equivalence is needed. It records the finite reverse containment and common-refinement reads, and it is absent when only one-way cofinal consumption is claimed.*
5. *Q is the completion handoff row consumed by* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *readers after the cofinal comparison rows have been displayed.*
6. *H records componentwise* hsame *transport for $S, T, \rho, \lambda, Q, C, P, N$.*
7. *C records displayed* Cont *replay from a target request through ρ, through the source base packet, through λ when present, and then through the completion handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterRefinement}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, ultrafilter extension, selected representatives, selected limits, countable choice, Zorn arguments, host filter equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CofinalFilterUp.TasteGate.CofinalFilterTasteGate single carrier alignment`

Theorem 2978.2 (CofinalFilter NameCert obligations)**.** *For every accepted* $\mathsf{CofinalFilter}^{\uparrow}$ *carrier* $F = (S, T, \rho, \lambda, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the source and target* $\mathsf{FilterBase}^{\uparrow}$ *rows, the cofinal map row, the reverse ledger when equivalence is requested, the completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer request must expose* T, ρ, S *and the relevant ledger rows before* Q *is available.*

Proof. Project the carrier of Definition 3955.1. The rows S and T display the source and target filter bases, while ρ is the only row that moves a target request to a source packet.

The ledger row λ is conditional on equivalence use. When the consumer asks only for one-way cofinal replacement it is not read; when equivalence is claimed it supplies exactly the reverse containment and common-refinement entries needed for that claim.

The completion handoff row Q is downstream of this comparison. Transport through H and replay through C preserve the same finite order, and P, N source and name the route. Since the excluded quotient, ultrafilter, selector, choice, Zorn, host-equality, and completeness objects are absent from the carrier, no further NameCert obligation is available. □

Theorem 2978.3 (CofinalFilter completion handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalFilter}^{\uparrow}$ *packet reads completion-facing data only through*

$$T \longmapsto \rho \longmapsto S \longmapsto \lambda, Q, H, C, P, N.$$

Thus the completion corridor factors through the displayed cofinal-filter rows and the sibling $\mathsf{FilterRefinement}^{\uparrow}$ *discipline before any completion request is answered. It exports no quotient filter, selected representative, selected limit, ultrafilter, countable-choice schedule, Zorn argument, host filter equality, or ambient completeness row.*

Proof. Begin with a completion-facing request. The consumer names a target filter base row T, so the carrier forces the request through the cofinal map row ρ before any source packet can be read.

Apply Definition 3955.2. The source row S supplies the displayed packet answering the target request, while λ is read only when the consumer needs the reverse comparison ledger required by the refinement discipline.

The handoff row Q is therefore reached after the finite cofinal comparison. Componentwise transport, continuation replay, provenance, and local naming preserve that same route for $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ consumers. No excluded quotient, selector, choice, Zorn, host-equality, or completeness coordinate is exported. □

Falsifiable prediction: If a completion-facing read cites $\mathsf{CofinalFilter}^{\uparrow}$ while bypassing the displayed target row, cofinal map, source row, required reverse ledger, completion handoff, hsame transport, Cont replay, Pkg provenance, or local NameCert row, it is not an accepted $\mathsf{CofinalFilter}^{\uparrow}$ packet.

Independence witness: Nearest siblings: filterbase, cauchyfilter, filter-refinement, uniformcompletion, metriccompletion; $\mathsf{CofinalFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly

records a target base, source base, cofinal map, optional reverse ledger, completion handoff, transport, continuation, provenance, and local naming in one finite corridor.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}$.

Constructive story: A $\mathsf{CofinalFilter}^{\uparrow}$ packet is generated from source and target FilterBase rows, a cofinal map, a reverse ledger when equivalence is requested, a completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3955.1, Definition 3955.2, and Definition 3955.3 seed the finite cofinal-filter corridor between filter bases, filter refinement, CauchyFilter, UniformCompletion, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected representatives, selected limits, ultrafilters, countable choice, Zorn arguments, host filter equality, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal map exactness, reverse-ledger compatibility, completion handoff non-escape, transport stability, replay stability, and ledger-refusal proofs for BEDC.Derived.CofinalFilterUp.

Chapter 2979

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, the $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ packet records the functorial route from metric source data to regular Cauchy real seals. It sits over the metric source of chapter 57, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the metric-completion surface of chapter 1506, and the completion action of chapter 423. The packet names the source-map and composition ledgers needed by completion consumers; it does not select representatives, quotient regular Cauchy names, identify host real objects, or import ambient completeness.

Definition 2979.1 (RegularCauchyCompletionFunctor carrier)**.** *A* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, U, W, Q, R, E, I, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source and target metric naming certificates and their distance ledgers.*
2. *U is the uniformly continuous map row between the displayed metric sources, with modulus and graph rows readable before completion is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the source windows requested by a regular Cauchy input name.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same finite windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *completion-name row obtained from W and Q.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the metric, map, finite-window, tolerance, and regular-rational rows have been displayed.*
7. *I is the identity ledger, saying that the identity uniformly continuous map replays the same windows, tolerances, readback rows, and real seal.*
8. *A is the composition ledger, saying that two displayed uniformly continuous map rows compose by replaying the source windows through the intermediate and target completion rows in order.*
9. *H records componentwise* hsame *transport for M, U, W, Q, R, E, I, A and for the structural rows below.*

10. *C records displayed* Cont *replay from metric source rows and uniformly continuous map rows through StreamName windows, DyadicRatCore tolerances, RegSeqRat completion names, RealUp seals, identity ledgers, and composition ledgers.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, W, Q, R, E, I, A, H, C, P, N$. There is no coordinate for a selected limit, quotient-name equality, host equality of real objects, global completeness, or an external functor object detached from the displayed rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionFunctorUp.RegularCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 2979.2 (RegularCauchyCompletionFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *carrier*

$$F = (M, U, W, Q, R, E, I, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$ *surface consists exactly of metric source admission through M; uniformly continuous map admission through U; finite-window exposure through W; dyadic tolerance exactness through Q; regular Cauchy completion-name readback through R; RealUp seal handoff through E; identity and composition functor ledgers through I and A; and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3956.1. Its accepted rows display exactly the metric source row, uniformly continuous map row, finite-window row, dyadic tolerance row, regular-rational readback row, real seal, identity ledger, composition ledger, and structural rows.

The source side is exhausted by M, U, W, Q. The metric row supplies the distance ledgers, the uniformly continuous map row supplies the graph and modulus data, and the finite-window and dyadic-tolerance rows record the only observations a regular Cauchy input can expose.

The target side is exhausted by R, E. The RegSeqRat row is read after the requested windows and tolerances, and the RealUp seal remains downstream from that completion-name readback.

The functorial rows add no hidden semantic object. The identity ledger I and composition ledger A record replay of the displayed route, while H, C, P, N transport, continue, source, and name the same finite packet. Hence the listed rows are the complete local NameCert obligation surface. □

Theorem 2979.3 (RegularCauchyCompletionFunctor composition law)**.** *Let F_f and F_g be accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *packets for displayed uniformly continuous maps $f\colon X \to Y$ and $g\colon Y \to Z$. Every completion-facing consumer of the composite read factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

for f, then through the same displayed route for g, with the composition ledger A preserving the intermediate windows, tolerance rows, regular-rational readbacks, RealUp seals, hsame *transports,* Cont *replays,* Pkg *provenance rows, and local NameCert rows. The identity ledger I gives the same statement when one of the two map rows is the displayed identity.*

Proof. Begin with the accepted packets for f and g. By Definition 3956.2, each packet exposes the metric source, uniformly continuous map, finite-window, dyadic-tolerance, regular-rational readback, real-seal, and structural rows before any completion-facing consumer can read its output.

For the composite request, read the first packet through M, U, W, Q, R, E. This produces only the displayed regular Cauchy completion-name row and its RealUp seal; it does not export a selected representative or a host equality of completed points.

Feed that displayed seal route to the second packet. The second packet repeats the same finite-window and tolerance discipline over its own metric and map rows. The ledger A records that this two-stage replay is the composite route, and I records that the identity map leaves the displayed windows, tolerances, readbacks, and seal unchanged.

Finally apply the structural rows. Componentwise hsame transport and displayed Cont replay preserve the two-stage route, while Pkg provenance and the local NameCert rows keep the consumer inside $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ data. Thus the composition law is a certificate route over displayed rows, not an external equality of functors. □

Theorem 2979.4 (RegularCauchyCompletionFunctor Real-seal handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *packet reaches the RealUp seal row only through the displayed MetricUp source row, uniformly continuous map row, StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, identity ledger, composition ledger, and structural rows. In carrier notation, the route to the seal is*

$$M, U, W, Q, R, E, I, A, H, C, P, N,$$

with E downstream from M, U, W, Q, R and still guarded by I, A, H, C, P, N. The handoff does not expose selected representatives, quotient regular Cauchy names, host real equality, or ambient completeness.

Proof. Project the carrier of Definition 3956.1. The row E is listed only after the metric source, uniformly continuous map, finite-window, dyadic-tolerance, and RegSeqRat readback rows have been displayed.

Apply the NameCert obligations of Definition 3956.2. They identify the RealUp seal handoff as one obligation among the same finite rows, and they keep identity, composition, transport, replay, provenance, and local naming visible as part of the accepted packet.

Finally use the composition law, Definition 3956.3. It shows that a completion-facing composite read preserves the same window, tolerance, readback, seal, and ledger route rather than creating an external functor equality. Since the carrier has no coordinate for representatives, quotient names, host real equality, or ambient completeness, the Real-seal handoff can export only the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ read is refuted if a completion-facing RealUp seal is reached without displaying the MetricUp source row, uniformly continuous map row, StreamName window row, DyadicRatCore tolerance row, RegSeqRat readback row, identity ledger, composition ledger, transport, replay, provenance, and local naming rows.

Independence witness: metric, streamname, regseqrat, real, metriccompletion, and cauchycompletionfunctor: none is bijective to the functor packet because only $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ records uniformly continuous source action together with identity and composition ledgers for regular Cauchy completion names.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ packet is generated from MetricUp source rows, uniformly continuous map rows, StreamName finite windows, DyadicRatCore tolerance rows, RegSeqRat completion-name rows, RealUp seal rows, identity and composition ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3956.1, Definition 3956.2, Definition 3956.3, and Definition 3956.4 seed the finite functorial route from MetricUp and uniformly continuous map rows through StreamName windows, DyadicRatCore tolerances, RegSeqRat completion names, and RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient regular Cauchy names, selected representatives, host equality of real objects, global completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric source exposure, uniformly continuous map exposure, window and tolerance routing, RegSeqRat readback, RealUp seal handoff, identity and composition ledgers, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionFunctorUp.

Chapter 2980

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$

Following Lebesgue and Bishop–Bridges, the $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ packet realizes the finite-net route of chapter 9237 as concrete NameCert data. It reads the compact carrier of chapter 60, the compact metric finite-net surface of chapter 129, the metric source rows of chapter 57, the continuous-map and continuous-function rows of chapter 130 and chapter 59, the rational lower-bound surface of chapter 37, the finite-net fold of chapter 619, and the source-lock packet of chapter 935. The packet records only finite $\mathsf{ProbeBundle}$ data, local pointwise-modulus rows, rational folds, and the resulting uniform-modulus handoff; it does not claim open-cover compactness, selected subsequences, countable choice, or a semantic Heine–Cantor theorem outside the displayed BEDC rows.

Definition 2980.1 (Compact uniform finite-net realization carrier)**.** *A* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, M, F, B, L, R, U, T, H, D, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Compact}^{\uparrow}$ *admission row, tying the source object to the compact naming certificate.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source-distance and target-distance ledgers.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, with pointwise modulus data readable at displayed source centers.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *row of compact-net centers and source-distance coverage witnesses.*
5. *L is the local-radius ledger, assigning a positive* $\mathsf{Rat}^{\uparrow}$ *radius to each bundled center from the pointwise-modulus row.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the finite bundle, using only primitive recursion over displayed probes.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the folded radius and the continuous graph rows.*
8. *T is the metric triangle row, joining the source-center estimate, the source-pair estimate, and the target-distance estimate.*

9. *H records componentwise* hsame *transport for C, M, F, B, L, R, U, T and for the structural rows below.*
10. *D records displayed* Cont *replay from compact finite-net inspection through pointwise-modulus reads, rational lower-bound folding, metric triangle estimates, and uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, M, F, B, L, R, U, T, H, D, P, N$. It has no coordinate for open-cover compactness, a selected subcover outside the finite bundle, a selected subsequence, quotient equality of real values, or an external modulus function.

Lean4 ✓ `BEDC.Derived.CompactUniformFiniteNetRealizationUp.CompactUniformFiniteNetRealizationTasteGate`
`single carrier alignment`

Theorem 2980.2 (Compact finite-net modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *packet receives the compact-uniform-continuity route through the finite chain*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Continuous}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The consumer first displays the compact metric source, then the finite ProbeBundle centers and coverage rows, then the pointwise-modulus radii supplied by the continuous row, then the rational lower-bound fold, and only then the metric-triangle estimate and uniform-modulus handoff.

Proof. Project the carrier of Definition 3957.1. The rows $C, M, F, B, L, R, U, T, H, D, P, N$ are the complete displayed inventory.

Read the compact and metric source rows first. The compact row admits the source object, while the metric row supplies the distance ledgers used by both the finite-net cover and the target estimate.

Next read the finite bundle. The ProbeBundle row B supplies exactly the displayed centers and source-distance coverage rows, and the continuous row F supplies pointwise-modulus radii only at those centers. The local-radius ledger L therefore remains tied to the finite compact-net data.

Now fold the radii. The $\mathsf{Rat}^{\uparrow}$ row R is obtained by primitive recursion over B, so the common lower bound is a classifier-stable history rather than a choice-selected minimum.

Finally the metric triangle row T and uniform-modulus row U export the handoff, with H, D, P, N transporting, replaying, sourcing, and naming the same finite surface. Thus the realization supplies the vision route as NameCert data without importing open-cover compactness, selected subsequences, countable choice, or an external modulus selector. □

Theorem 2980.3 (Compact uniform finite-net realization nonescape)**.** *Let $K = (C, M, F, B, L, R, U, T, H, D, P, N$ be an accepted* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *carrier. Any consumer that reaches the uniform-modulus handoff must factor through the same finite ProbeBundle row B, the same* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold R, the same continuous-map pointwise-modulus row F, and the same metric triangle row T. No alternate path inside the carrier exports open-cover compactness, a selected subcover outside B, a selected subsequence, or a host-level modulus function.*

Proof. Start from the output row U. It is stored after C, M, F, B, L, R, T in the carrier, so the uniform-modulus handoff cannot be read without the compact source, metric source, continuous row, finite bundle, local radii, rational fold, and triangle estimate.

The finite bundle is the only cover coordinate. Since B is the sole row containing centers and coverage witnesses, any source point used by the handoff must be routed through one of the displayed probe names.

The lower-bound and triangle rows are likewise unique inside the packet. The fold row R compares only the radii displayed by L, and T is the only row converting source estimates and pointwise target estimates into the final uniform estimate.

Transport, replay, provenance, and local naming preserve this route rather than adding new analytic data. Hence every consumer of U remains inside the finite NameCert surface described by the compact-uniform-continuity vision. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ read is refuted if a UniformModulusUp handoff is reached without displaying the CompactUp row, MetricUp source row, ContinuousUp pointwise-modulus row, finite ProbeBundle centers, RatUp lower-bound fold, metric triangle row, transport, replay, provenance, and local naming rows.

Independence witness: compact, compactmetric, metric, continuous, rat, probebundle, finitenetminimumfold, and uniformmodulus: none is bijective to the finite-net realization packet because only $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ records the compact-uniform-continuity vision route from finite ProbeBundle coverage through rational folding to the UniformModulusUp handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ packet is generated from CompactUp admission rows, MetricUp source rows, ContinuousUp or ContinuousMapUp pointwise-modulus rows, finite ProbeBundle center and coverage rows, RatUp lower-bound folds, UniformModulusUp handoff rows, MetricTriangleUniformEstimateUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3957.1, Definition 3957.2, and Definition 3957.3 seed the finite ProbeBundle realization route for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers outside the finite bundle, selected subsequences, countable choice, semantic Heine–Cantor theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, metric source exposure, continuous pointwise-modulus exposure, finite ProbeBundle coverage, RatUp lower-bound folding, metric triangle exactness, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformFiniteNetRealizationUp.

Chapter 2981

A Concrete Naming Certificate for $\mathsf{MetricSequentialCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricSequentialCompletion}^{\uparrow}$ packet records the finite interface where sequential completion data is read through a metric-completion surface without quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits between the sequential route of chapter 1683, the metric-completion boundary of chapter 1506, the complete-metric handoff of chapter 128, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a completion-facing route over displayed rows only; it does not identify Cauchy sequences by a host quotient or choose completed representatives.

Definition 2981.1 (MetricSequentialCompletion carrier)**.** *A* $\mathsf{MetricSequentialCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, C, W, T, R, E, H, K, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{SequentialCompletion}^{\uparrow}$ *source row naming the sequential completion request.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row through which completion-facing consumers read the metric boundary.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row exposed only after the sequential and metric rows are displayed.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect the sequential source.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W and T.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after Q, M, C, W, T, R.*
8. *H records componentwise* hsame *transport for the sequential, metric, complete, window, tolerance, readback, seal, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from the sequential request through the metric-completion boundary, complete-metric handoff, finite windows, dyadic tolerance, regular readback, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{MetricSequentialCompletion}^{\uparrow}$, $\mathsf{SequentialCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected sequence limits, countable-choice selectors, host equality of completed points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetricSequentialCompletionUp.MetricSequentialCompletionTasteGate single carrier alignment`

Theorem 2981.2 (MetricSequentialCompletion NameCert obligations)**.** *For every accepted* MetricSequentialComp *carrier $S = (Q, M, C, W, T, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$ *surface consists exactly of sequential-source admission through Q, metric-completion admission through M, complete-metric handoff through C, finite-window exposure through W, dyadic tolerance exactness through T, regular rational readback through R, real-name seal handoff through E, and structural transport, replay, provenance, and local naming through H, K, P, N.*

Proof. Project the carrier of Definition 3958.1. The accepted rows display the sequential source, metric completion row, complete-metric handoff, finite windows, dyadic tolerance ledger, regular readback, terminal real seal, and structural rows.

The completion side is exhausted by Q, M, C. A consumer first reads the sequential request, then the metric-completion boundary, and only then the complete-metric handoff.

The real-name side is exhausted by W, T, R, E. The finite window family determines the dyadic tolerance reads, the tolerance ledger determines the regular rational readback, and the real seal remains downstream of that readback.

Transport, replay, provenance, and local naming preserve this route. Since no other coordinate is present, quotient streams, selected limits, countable choice, host equality, and ambient completeness cannot enter the local certificate surface. □

Theorem 2981.3 (MetricSequentialCompletion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricSequentialCompletion}^{\uparrow}$ *carrier reads the same finite route*

$$\mathsf{SequentialCompletion}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

The handoff sends sequential completion data to regular rational readback through displayed stream windows and dyadic tolerances; it does not use quotient streams, selected limits, countable choice, host equality, or ambient real completeness.

Proof. Begin with Definition 3958.2. It fixes the local certificate surface at the rows $Q, M, C, W, T, R, E, H, K, P, N$.

Read the sequential and metric rows first. The packet admits the sequential request through Q, the metric-completion boundary through M, and the complete-metric handoff through C before a readback row can be inspected.

Now read the finite windows and tolerances. The rows W and T determine the regular rational row R, and the real-name seal E is downstream of R.

The structural rows keep the same handoff. Componentwise transport, displayed replay, provenance, and local naming record only this route, so the excluded quotient, selector, choice, host-equality, and completeness data remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSequentialCompletion}^{\uparrow}$ packet is generated from SequentialCompletionUp source rows, MetricCompletionUp boundary rows, CompleteMetricUp handoff rows, StreamName window rows, DyadicRatCore tolerance rows, RegSeqRat readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3958.1, Definition 3958.2, and Definition 3958.3 seed the finite route from sequential completion and metric completion rows through stream windows, dyadic tolerances, regular rational readback, and the real-name seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricSequentialCompletionUp.MetricSequentialCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable-choice selectors, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequential-source exposure, metric-completion admission, complete-metric handoff, window and tolerance routing, regular rational readback, real-name seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricSequentialCompletionUp.

Chapter 2982

A Concrete Naming Certificate for $\mathsf{SpectralAblationHinge}^{\uparrow}$

The $\mathsf{SpectralAblationHinge}^{\uparrow}$ packet realizes the spectral-ablation ledger of Definition 9462.1458, the single-codon deletion table of Definition 9462.1459, the orbit-module deletion table of Definition 9462.1461, the tail-coupling edge deletion of Definition 9462.1463, and the load-bearing interface summary of Definition 9462.1468 as finite NameCert data. It records deletion rows, killed-walk rows, spectral-loss rows, orbit-module rows, transport rows, replay rows, provenance rows, and local naming rows. The packet does not claim a biological fitness law, a continuous spectrum, a host eigenvector selector, or an external ranking rule outside the displayed ledger.

The packet consumes the finite hinge-ledger rows of chapter 3961, the quotient spectral trigger rows of chapter 2317, and the energy-partition boundary of chapter 3806 before its ranking row is read. The killed-walk matrix rows and hinge rankings come from the ledger sibling, while the quotient trigger and energy-partition rows constrain which deletion reads can be treated as trigger-facing.

Definition 2982.1 (SpectralAblationHinge carrier)**.** *A* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, K, L, O, T, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite deletion row, naming the deleted codon, edge, edge group, or orbit module.*
2. *K is the killed-walk row computed after the displayed deletion.*
3. *L is the spectral-loss ledger row recording the displayed difference between the source spectral radius and the killed-walk spectral radius.*
4. *O is the orbit-module row that records whether the deletion is point-level, module-level, interface-level, or single-edge data.*
5. *T is the tail-coupling row for the* UUR–CUR *antenna interface when the deletion reads that boundary.*
6. *R is the hinge-ranking row obtained by reading only D, K, L, O, T.*
7. *H records componentwise* hsame *transport for deletion, killed-walk, loss, orbit-module, tail-coupling, ranking, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from finite deletion data through killed-walk inspection, spectral-loss computation, orbit-module classification, tail-coupling inspection, and hinge ranking.*
9. *P is the* Pkg *provenance over* $\mathsf{SpectralAblationHinge}^{\uparrow}$*, radial-wobble-spectrum, wobble-quotient-spectral-trigger, six-boundary-tile-coverage, window-coverage-trigger-algebra, hyperbolic-metric,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$ *row.*

The carrier has no coordinate for an accepted hinge ranking that bypasses the spectral-loss ledger L.

Lean4 ✓ `BEDC.Derived.SpectralAblationHingeUp.SpectralAblationHingeTasteGate single carrier alignment`

Theorem 2982.2 (SpectralAblationHinge NameCert obligations)**.** *For every accepted* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier $A = (D, K, L, O, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$ *surface consists exactly of finite deletion admission through D, killed-walk admission through K, spectral-loss ledger exactness through L, orbit-module classification through O, tail-coupling boundary exposure through T, hinge-ranking readback through R, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3959.1. The displayed rows are exactly $D, K, L, O, T, R, H, C, P, N$.

The deletion side is exhausted by D, K, L. A consumer first reads the finite deletion row, then the killed-walk row after that deletion, and then the spectral-loss ledger comparing the source and killed rows.

The hinge side is exhausted by O, T, R. Orbit-module classification and tail-coupling inspection supply the only ranking inputs, and the ranking row R is read only after those rows and the loss ledger have been displayed.

Transport, replay, provenance, and local naming preserve this finite ledger. No accepted NameCert obligation is stored in a host eigenvector selector, a continuous spectrum, or a ranking rule detached from D, K, L, O, T, R. □

Theorem 2982.3 (SpectralAblationHinge loss route)**.** *Every accepted hinge-ranking read from a* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier factors through the route*

$$D \longmapsto K \longmapsto L \longmapsto O, T \longmapsto R \longmapsto H, C, P, N.$$

Deleting the spectral-loss ledger L prevents the carrier from exporting an accepted hinge ranking, even when the deletion row, killed-walk row, orbit-module row, and tail-coupling row remain displayed.

Proof. Start with the NameCert obligations of Definition 3959.2. They require finite deletion admission, killed-walk admission, and spectral-loss ledger exactness before orbit-module or hinge-ranking data is read.

The ranking row is downstream of the loss row. The orbit-module row O and tail-coupling row T classify the deletion, but neither row stores the spectral loss itself.

If L is deleted, the displayed route cannot reach R as an accepted hinge-ranking row. The structural rows H, C, P, N only transport, replay, source, and name accepted rows, so they cannot reconstruct the missing loss ledger. □

Falsifiable prediction: Deleting the spectral-loss ledger prevents any accepted $\mathsf{SpectralAblationHinge}^{\uparrow}$ hinge ranking read, even when finite deletion, killed-walk, orbit-module, tail-coupling, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: radial-wobble-spectrum, wobble-quotient-spectral-trigger, six-boundary-tile-coverage, window-coverage-trigger-algebra, hyperbolic-metric: none is bijective to the spectral ablation hinge packet because only $\mathsf{SpectralAblationHinge}^{\uparrow}$ binds deletion, killed-walk, spectral-loss, orbit-module, tail-coupling, ranking, transport, replay, provenance, and local naming rows into one accepted hinge ledger.

Lean follow-up must provide $\mathsf{Nontrivial\ SpectralAblationHinge}^{\uparrow}$ and $\mathsf{FieldFaithful\ SpectralAblationHinge}^{\uparrow}$ in TasteGate.lean before the packet can be read as a checked TasteGate object.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralAblationHinge}^{\uparrow}$ packet is generated from finite deletion rows, killed-walk rows, spectral-loss ledger rows, orbit-module rows, tail-coupling rows, hinge-ranking rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3959.1, Definition 3959.2, and Definition 3959.3 seed the finite deletion, killed-walk, spectral-loss, orbit-module, tail-coupling, ranking, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological fitness laws, continuous spectral theory, host eigenvector selectors, Lean verification, TasteGate checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deletion admission, killed-walk admission, spectral-loss exactness, orbit-module classification, tail-coupling exposure, ranking readback, transport, replay, provenance, and local naming for BEDC.Derived.SpectralAblationHingeUp.

Chapter 2983

A Concrete Naming Certificate for $\mathsf{SierpinskiCarpet}^{\uparrow}$

Following Sierpinski and the constructive compact-metric treatment of finite square approximations, the $\mathsf{SierpinskiCarpet}^{\uparrow}$ packet records the standard middle-square deletion fractal as retained grid rows and coordinate readback data. It sits beside the retained-prefix route of chapter 1530, the compact source discipline of chapter 129, the finite-grid continuous-surjection surface of chapter 2078, the terminal real seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet names only finite square-grid retention and coordinate windows; it does not import a host closed subset of the plane, an infinite branch selector, quotient equality of real pairs, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2983.1 (Sierpinski carpet carrier)**.** *A* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the retained square-grid row. It records a finite triadic square address, its depth, the retained child squares at that depth, and the compact-metric grid enclosure used by later consumers.*
2. *D is the deleted-middle-square ledger. For every inspected square address it records whether the central child has been removed and which retained children remain available.*
3. *W is the coordinate* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the finite horizontal and vertical coordinate windows attached to the retained square.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two coordinate schedules, with dyadic comparisons routed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any real seal is exposed.*
5. *E is the pair of* $\mathsf{Real}^{\uparrow}$ *coordinate seals obtained from the same retained square-grid and coordinate-window rows.*
6. *H records componentwise* hsame *transport for retained grid rows, deletion ledgers, coordinate windows, rational readbacks, real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route from retained grid rows and deleted-middle-square ledgers through coordinate windows, rational readbacks, and real seals.*

8. *P is the* Pkg *provenance over* $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoCurve}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, D, W, Q, E, H, C, P, N$. It has no coordinate for a completed planar subset, selected infinite address, quotient equality of real pairs, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.SierpinskiCarpetUp.SierpinskiCarpetTasteGate single carrier alignment`

Theorem 2983.2 (Sierpinski carpet NameCert obligations)**.** *For every accepted* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *carrier*

$$S = (G, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; retained square-grid exposure through G; deleted-middle-square exclusion through D; coordinate window exposure through W; regular rational readback through Q; real coordinate sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. No obligation is discharged by a host planar powerset, an infinite branch selector, quotient real-pair equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3960.1. The accepted packet displays exactly the retained square-grid row, deletion ledger, coordinate windows, rational readbacks, real seals, and structural rows.

The retained-grid and deletion obligations are exhausted by G and D. A consumer cannot inspect a square address until the retained child row has been displayed, and a deleted central square can be refused only by the ledger row attached to that same address.

The coordinate obligations are exhausted by W, Q, E. The coordinate windows are read first, the regular rational readback uses those windows with the dyadic tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seals remain downstream of the same retained square.

Finally H, C, P, N transport, replay, source, and name only the displayed rows. Since the excluded host principles have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ obligations. □

Theorem 2983.3 (Sierpinski carpet finite square-deletion route)**.** *Every accepted* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *consumer factors its membership or compact-metric read through the finite route*

$$\mathsf{CantorSet}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{PeanoCurve}^{\uparrow} \longmapsto G, D \longmapsto W, Q, E \longmapsto H, C, P, N.$$

At each inspected depth the route first displays retained square-grid rows and deleted-middle-square ledgers, then reads coordinate $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, and terminal* $\mathsf{Real}^{\uparrow}$ *seals. It exports no completed planar subset, selected infinite branch, quotient real-pair equality, open-cover compactness theorem, or ambient completeness principle.*

Proof. Begin with the retained square-grid row. It gives a finite triadic square address and the retained children available at the inspected depth, while the deleted-middle-square ledger records which central cell is refused.

Read coordinates only after that ledger is visible. The window row W supplies the horizontal and vertical finite schedules, and Q reads those schedules as regular rational data with the dyadic tolerance comparisons already displayed.

The Real-facing boundary is then the row E. It seals the two coordinate schedules as $\mathsf{Real}^{\uparrow}$ data for the retained square already named by G and checked by D.

The structural rows preserve the same route. Componentwise transport, displayed replay, provenance, and local naming cannot create a second membership surface, so every compact-metric or coordinate consumer remains inside the finite square-deletion packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$.

Constructive story: A $\mathsf{SierpinskiCarpet}^{\uparrow}$ packet is generated from retained square-grid rows, deleted-middle-square ledgers, coordinate StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance checks, RealUp coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3960.1, Definition 3960.2, and Definition 3960.3 seed the finite square-deletion compact-metric certificate surface for the Sierpinski carpet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SierpinskiCarpetUp.SierpinskiCarpetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host planar powersets, selected infinite addresses, quotient equality of real coordinate pairs, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained square-grid exposure, deleted-middle-square exclusion, coordinate readback, Real coordinate sealing, transport, replay, provenance, and local naming for BEDC.Derived.SierpinskiCarpetUp.

Chapter 2984

A Concrete Naming Certificate for $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$

Following spectral graph perturbation theory and the Perron–Frobenius analysis of finite killed-walk matrices, the $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ packet records the ablation hinges of the median-closure codon graph as a finite ledger. It concretizes the point, orbit-module, edge-group, antenna-interface, and single-edge rankings of Definition 9462.1468 beside the median-closure packet of chapter 3545, the radial wobble spectrum of chapter 3681, the CUN antenna surface of chapter 3771, and the quotient trigger route of chapter 2317. The packet is a finite NameCert ledger for ablation rows; it is not a host eigenvector computation, a biological optimality claim, an infinite perturbation scheme, or an ambient spectral theorem.

Falsifiable prediction: Every accepted $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ consumer reads killed-walk matrix rows, orbit and module rows, point and edge ablation rows, hinge rankings, hsame transport, Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row before treating any hinge as load-bearing.

Independence witness: median-closure, radial-wobble-spectrum, CUN antenna, wobble-quotient-trigger: $\mathsf{MedianClosure}^{\uparrow}$ records closure vertices rather than ablation losses, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ records radial spectral layers rather than killed-row deletion ledgers, $\mathsf{CUNSpectralAntenna}^{\uparrow}$ records tail gain rather than the full hinge ranking, and $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ records trigger acceptance rather than the ablation ledger; none is bijective to $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$.

Definition 2984.1 (Spectral ablation hinge ledger carrier)**.** *A* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, O, A, E, H, R, T, C, P, N)$$

with the following displayed rows:

1. *M is the killed-walk matrix row, recording the finite median-closure adjacency matrix and the baseline Perron radius used for every ablation comparison.*
2. *O is the orbit and module row, recording the* UUR*,* AGR*,* CUR*,* CUY*,* Anchor_6*, and* $\mathrm{StopStart}_6$ *rows together with their finite membership ledgers.*
3. *A is the point-ablation row, recording each inspected codon deletion and its displayed spectral loss.*
4. *E is the edge-ablation row, recording the internal edge-group deletions, single-edge deletions, and the* $\mathrm{UUR} - \mathrm{CUR}$ *antenna interface deletion.*

5. *H is the hinge-ranking row, ordering the displayed point, module, edge-group, antenna-interface, and single-edge losses by their finite comparisons.*
6. *R is the read route, classifying a consumer request as matrix readback, orbit-module readback, point ablation, edge ablation, hinge ranking, or refusal.*
7. *T records componentwise* hsame *transport for matrix, orbit, point-ablation, edge-ablation, hinge-ranking, route, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$M, O, A, E, H, R \longmapsto T, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, $\mathsf{CUNSpectralAntenna}^{\uparrow}$, $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, O, A, E, H, R, T, C, P, N$. It has no coordinate for an ambient Perron vector, an infinite graph limit, a biological necessity assertion, a real-valued oracle, or a hinge ranking not witnessed by the displayed rows.

Lean4 ✓ `BEDC.Derived.SpectralAblationHingeLedgerUp.SpectralAblationHingeLedgerTasteGate single carrier alignment`

Theorem 2984.2 (Spectral ablation hinge ledger NameCert obligations)**.** *For every accepted* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *carrier*

$$K = (M, O, A, E, H, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; killed-walk matrix readback through M; orbit and module exposure through O; point-ablation exactness through A; edge and antenna-interface ablation exactness through E; hinge-order stability through H; route classification through R; componentwise transport through T; displayed replay through C; provenance through P; local naming through N; ledger nonescape from ambient Perron vectors, infinite graph limits, biological necessity assertions, and oracle-valued rankings; and public consumer rows for point, module, edge-group, antenna-interface, and single-edge hinge reads.*

Proof. Carrier admission projects the tuple of Definition 3961.1. The matrix row M, orbit row O, ablation rows A, E, hinge row H, and structural rows R, T, C, P, N are the only visible coordinates.

The ablation-row exactness obligations are exhausted by A and E. A point deletion is admitted only through the finite codon row recorded in A, and an edge or antenna-interface deletion is admitted only through the finite edge ledger recorded in E.

Hinge-order stability is a comparison inside H. The row reads the same displayed loss values from A and E, so a consumer cannot reorder the hinge list by importing an external Perron vector or an unrecorded spectral computation.

Finally R, T, C, P, N classify, transport, replay, source, and name the same finite rows. Since the excluded ambient objects have no carrier coordinate, every public consumer read of a point, module, edge-group, antenna-interface, or single-edge hinge remains inside the ledger. □

Theorem 2984.3 (Spectral ablation hinge public route)**.** *Every accepted public consumer of a* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MedianClosure}^{\uparrow}, \mathsf{RadialWobbleSpectrum}^{\uparrow}, \mathsf{CUNSpectralAntenna}^{\uparrow}, \mathsf{WobbleQuotientSpectralTrigger}^{\uparrow} \longmapsto M, O, A, E, H,$$

Thus the matrix read, orbit-module rows, point-ablation rows, edge-ablation rows, hinge ranking, route classification, transport, replay, provenance, and local naming are visible before the consumer exports a load-bearing hinge. No exported row asserts host spectral optimality, biological necessity, an infinite perturbation theorem, or an ablation comparison outside the displayed ledger.

Proof. Start with the matrix and orbit support. The rows M and O identify the finite killed-walk matrix and the finite orbit or module being inspected.

Read ablation data next. The point row A and edge row E carry the deletions and their spectral losses, while H records the hinge order obtained from those same rows.

The public boundary is then the route row R. It decides whether a consumer is reading the matrix, an orbit or module, a point deletion, an edge deletion, a hinge ranking, or a refusal.

The structural rows preserve that boundary. Componentwise transport, continuation replay, provenance, and local naming cannot introduce a second spectral source, so every accepted public read factors through the displayed packet. □

Theorem 2984.4 (Spectral ablation hinge vision route)**.** *Any accepted* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *hinge read must pass through*

$$\mathsf{MedianClosure}^{\uparrow}, \mathsf{RadialWobbleSpectrum}^{\uparrow}, \mathsf{CUNSpectralAntenna}^{\uparrow}, \mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$$

before it can be read as the load-bearing interface of Definition 9462.1468. The sibling routes supply closure vertices, radial spectral layers, antenna gain, and trigger acceptance, while the local packet supplies only the ablation ranking over $M, O, A, E, H, R, T, C, P, N$.

Proof. Project the carrier rows $M, O, A, E, H, R, T, C, P, N$ of Definition 3961.1. The rows M and O expose the killed-walk matrix and finite orbit or module support; the rows A, E, H expose point loss, edge loss, and the hinge order; and R, T, C, P, N classify, transport, replay, source, and name that ledger.

Read the sibling routes before the local ablation row is consumed. The $\mathsf{MedianClosure}^{\uparrow}$ route supplies the closure vertices, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ supplies the radial layers, $\mathsf{CUNSpectralAntenna}^{\uparrow}$ supplies antenna gain, and $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ supplies trigger acceptance.

Only after those routes have been displayed can the ledger row rank a hinge for the public philosophy interface. The local packet contributes no independent closure vertex, radial layer, antenna gain, or trigger-acceptance theorem; it contributes the finite ablation ranking that can be read against those already displayed surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ packet is generated from killed-walk matrix rows, orbit-module rows, point-ablation rows, edge-ablation rows, hinge rankings, route classification, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3961.1, Definition 3961.2, and Definition 3961.3 seed the finite spectral-ablation hinge ledger surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Perron-vector computation, biological semantics, infinite perturbation theory, oracle-valued rankings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, killed-walk matrix readback, orbit-module exposure, ablation-row exactness, hinge-order stability, route classification, transport, replay, provenance, and local naming for BEDC.Derived.SpectralAblationHingeLedgerUp.

Chapter 2985

A Concrete Naming Certificate for BanachContractionPrinciple$^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive analysis, the BanachContractionPrinciple$^{\uparrow}$ packet records a fixed-point route only when the complete metric carrier, the contractive map, the iteration schedule, and the Cauchy handoff are all explicit finite rows. It sits downstream of the complete metric carrier of chapter 128, the contractive map packet of chapter 3819, the fixed point iteration route of chapter 2383, the Cauchy completion reflector of chapter 1585, and the real-number ground of chapter 38. The certificate therefore consumes existing completion data; it does not assert ambient completeness, global choice, or an unrecorded limit operator.

Definition 2985.1 (Banach contraction principle carrier)**.** *A* BanachContractionPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, F, I, Q, E, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the complete metric row, naming the metric carrier and the completeness seal used by the argument;*
2. *F is the contraction map row, including the displayed modulus witness and its compatibility with the metric row;*
3. *I is the iteration schedule row, recording the initial point and the successive iterates read by the construction;*
4. *Q is the Cauchy witness row for the iteration sequence;*
5. *E is the endpoint seal row, identifying the completed endpoint supplied by the complete metric data;*
6. *U is the uniqueness row, binding any second fixed endpoint back to the same contraction ledger;*
7. *H records componentwise* hsame *transports for M, F, I, Q, E, U together with continuation, provenance, and local naming rows;*
8. *C records the displayed* Cont *replay routes by which the contraction ledger, the Cauchy witness, the endpoint seal, and uniqueness row are read;*
9. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* CompleteMetric$^{\uparrow}$*,* ContractiveMap$^{\uparrow}$*,* FixedPointIteration$^{\uparrow}$*,* CauchyCompletionReflector$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten displayed rows. It has no coordinate for an arbitrary choice of fixed point, an ambient completeness claim outside M, a hidden contraction constant, or a limit witness that does not pass through Q and E.

Lean4 ✓ `BEDC.Derived.BanachContractionPrincipleUp.BanachContractionPrincipleTasteGate single carrier alignment`

Theorem 2985.2 (Banach contraction principle NameCert obligations)**.** *For the carrier of Definition 3962.1, the local* $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for $B = (M, F, I, Q, E, U, H, C, P, N)$;*
2. *contraction ledger exposure, requiring every use of the map row F to read the displayed contraction data against M;*
3. *Cauchy-iteration handoff, requiring the iteration schedule I to produce the witness row Q before an endpoint is read;*
4. *complete-metric seal, requiring the endpoint row E to be supplied by the completeness data in M rather than by an ambient choice operator;*
5. *uniqueness nonescape, requiring every competing fixed endpoint to factor through U, H, C, P, N with no off-packet fixed-point row.*

Proof. Carrier admission is projection from the displayed packet. The rows $M, F, I, Q, E, U, H, C, P, N$ are the full inventory made available to the local certificate.

The contraction ledger is read through F against the metric data in M. Thus a contraction claim with a hidden modulus or a different metric carrier has no row in the packet.

The iteration schedule then hands off to the Cauchy witness row. The endpoint seal can be read only after Q is present and the complete metric data in M supplies the endpoint row E.

Finally, uniqueness is confined to U and the replay rows H, C, P, N. A second fixed endpoint that escapes these rows would require an additional carrier coordinate, which the classifier does not provide. □

Theorem 2985.3 (Banach contraction endpoint uniqueness)**.** *Let $B = (M, F, I, Q, E, U, H, C, P, N)$ and $B' = (M, F, I, Q, E', U', H', C', P', N')$ be accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *packets over the same complete metric row M, contraction row F, iteration schedule I, and Cauchy witness row Q. If the endpoint seals E and E' are transported through the displayed uniqueness rows U and U', then the metric carrier named by M classifies the two endpoint seals together. The conclusion exposes endpoint uniqueness only along the displayed contraction ledger; it does not select a fixed point outside the packet.*

Proof. Project both carriers to the shared rows M, F, I, Q. The contraction ledger exposure forces every fixed-point read to use the same metric carrier, the same contractive map row, and the same iteration schedule before any endpoint seal can be inspected.

The Cauchy witness row Q then supplies the common route into the complete metric row M. Both endpoint seals E and E' are therefore read as completions of the displayed iteration data rather than as independently selected fixed points.

Apply the uniqueness rows U and U' through their componentwise transports and replay routes. The rows H, C, P, N and H', C', P', N' keep the two endpoint reads on the same local NameCert

surface, so the metric classifier of M identifies the endpoint seals. Since the carriers have no coordinate for an off-packet fixed point, no such point is exported. □

Theorem 2985.4 (Banach contraction iteration Cauchy handoff)**.** *Every accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *carrier* $B = (M, F, I, Q, E, U, H, C, P, N)$ *routes its displayed fixed-point iteration through the Cauchy tail row* Q *before the completion reflector reads the endpoint seal* E*. The rows named by* P *may expose* $\mathsf{FixedPointIteration}^{\uparrow}$ *and* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *consumers only after the contraction ledger* F*, iteration schedule* I*, and Cauchy witness* Q *have been displayed. Thus the packet exports a completion-facing endpoint handoff, not a chosen fixed point, a hidden limit operator, a global choice rule, or completeness outside the metric row* M*.*

Proof. Project the accepted carrier. The definition places the complete metric row M, contraction row F, iteration schedule I, Cauchy witness Q, and endpoint seal E in a fixed displayed order.

First read F and I. The contraction ledger determines the successive iterate route, while the iteration schedule records the finite prefix and successor replay used by downstream fixed-point consumers.

Next read Q. This is the only Cauchy tail row in the carrier, so the completion reflector cannot receive an endpoint request until the iteration has been certified as Cauchy through the displayed row.

Finally replay the completion-facing handoff through E, H, C, P, N. Since no carrier coordinate stores a chosen endpoint or an ambient completeness witness, the exported endpoint is exactly the reflector output attached to the displayed Cauchy tail. □

Theorem 2985.5 (Banach contraction endpoint route)**.** *Every accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *endpoint read factors through* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{ContractiveMap}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* $\mathsf{CauchyCompletionReflector}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows in the carrier order*

$$M, F, I, Q, E, U, H, C, P, N.$$

The route exports no chosen endpoint, hidden limit operator, global choice rule, quotient completion equality, or completeness outside M*.*

Proof. Start with the carrier of Definition 3962.1. The endpoint row E is downstream of the complete metric row M, contraction map row F, iteration schedule I, and Cauchy witness Q.

Apply Definition 3962.2. The NameCert surface requires carrier admission, contraction ledger exposure, Cauchy-iteration handoff, complete-metric seal, and uniqueness nonescape before a fixed-point endpoint can be read.

Use Definition 3962.3 to keep any competing endpoint inside the displayed uniqueness row U. Then use Definition 3962.4 to route the displayed iteration through Q before the completion reflector reads E.

Finally H, C, P, N provide transport, replay, provenance, and local naming over the same finite rows. Since the carrier has no coordinate for a chosen endpoint, hidden limit operator, global choice rule, quotient completion equality, or completeness outside M, the endpoint route is exactly the displayed carrier order. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachContractionPrinciple}^{\uparrow}$ packet is generated from a complete metric row, a displayed contraction map, an iteration schedule, a Cauchy witness, an endpoint seal, a uniqueness row, componentwise transports, replay routes, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 3962.1 and Definition 3962.2 bind the finite carrier and its five local certificate obligations.

Formal verification: FormalStatus(NameCert$_{\mathsf{BanachContractionPrinciple}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Lean verification, a standard bridge, global fixed-point choice, nonconstructive compactness, or completeness outside the displayed metric row.
Upgrade path: A Lean carrier and obligation theorem for the displayed route would raise the verification axis while preserving the paper carrier boundary.

Chapter 2986

A Concrete Naming Certificate for PublicLocalReflectiveCompilation$^{\uparrow}$

The PublicLocalReflectiveCompilation$^{\uparrow}$ packet realizes the public/local reflective compilation surface of section 9414.57 as finite NameCert data. It sits beside the compiler event-flow audit of chapter 829, the MetaCIC closure trace of chapter 795, the self-compilation boundary of chapter 9235, and the focused readback discipline of the vision chapter. The packet keeps the public audit substrate and the local self-audit substrate separate while letting both replay through one visible compiler-side certificate.

Definition 2986.1 (Public local reflective compilation carrier)**.** *A* PublicLocalReflectiveCompilation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, M, C, S, L, U, X, F, T, E, H, K, P, N)$$

with the following rows:

1. *A is the public compiler-side audit substrate row, exposing finite event-flow, admission, and evidence-packet data.*
2. *M is the MetaCIC audit row, recording the kernel-facing closedness and acceptance checks used by public reflective compilation.*
3. *C is the public reflective compiler row, naming the source packet, generated packet, and accepted readback boundary.*
4. *S is the local self-compiler row, restricted to the focused local trace and not exported as public compiler authority.*
5. *L is the local self-audit row, recording which local reads may be replayed against the focused trace.*
6. *U is the proxy-readback row, separating the current self-proxy from the public evidence packet and from the far endpoint.*
7. *X is the refusal row, listing hidden prompt state, uninspected compiler state, unbounded self-reference, public authority for local reads, and local authority for public claims as absent coordinates.*
8. *F is the FieldFaithful row for the displayed fields A, M, C, S, L, U, X.*

9. *T is the Nontrivial row distinguishing the public audit substrate, MetaCIC audit substrate, local self-compiler, local self-audit, proxy-readback, and refusal roles.*
10. *E is the transport and replay ledger for public compiler reads and local self-audit reads.*
11. *H records componentwise* hsame *transport for all displayed rows.*
12. *K records displayed* Cont *replay through the compiler-side, MetaCIC-audit, local-self-compiler, local-self-audit, proxy-readback, refusal, transport, and replay rows.*
13. *P is the* Pkg *provenance over* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$*, compiler event-flow audit rows, MetaCIC closure and acceptance rows, local self-compilation rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$ *row.*

The classifier reads only $A, M, C, S, L, U, X, F, T, E, H, K, P, N$. It has no coordinate for hidden prompt state, unrestricted self-application, a public proof of local identity, local authority over public compiler claims, unbounded reflective closure, or a compiler oracle outside the displayed rows.

Lean4 ✓ `BEDC.Derived.PublicLocalReflectiveCompilationUp.PublicLocalReflectiveCompilationTasteGate`
`single carrier alignment`

Theorem 2986.2 (Public local reflective compilation NameCert obligations)**.** *For every accepted* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ *carrier $R = (A, M, C, S, L, U, X, F, T, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$ *surface consists exactly of compiler-side audit exposure, MetaCIC-audit exposure, public reflective compilation routing, local self-compiler confinement, local self-audit confinement, proxy-readback separation, refusal-row exactness, FieldFaithful projection, Nontrivial role separation, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3963.1. The public rows A, M, C are exposed before any generated evidence packet can be read as a public reflective compilation result.

The local rows S, L, U are then read only along the focused trace. They provide local self-compiler, local self-audit, and proxy-readback data, but the refusal row X prevents those local reads from becoming public compiler authority or unrestricted self-reference.

The rows F and T keep the field projection faithful and the roles nontrivial: public audit, MetaCIC audit, public compiler, local self-compiler, local self-audit, proxy-readback, and refusal are distinct fields of the same carrier rather than aliases.

Finally E, H, K, P, N transport, replay, source, and name the finite packet. Because the excluded authorities have no carrier coordinate, every accepted read remains inside the displayed public/local reflective compilation surface. □

Theorem 2986.3 (Public local reflective compilation local-public boundary)**.** *Every accepted* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ *read that reaches the public compiler row C must pass through the public audit substrate A and the MetaCIC audit row M before it can inspect the local self-compiler row S or local self-audit row L. Conversely, every local self-audit read through S, L, U must retain the refusal row X and cannot export public compiler authority, hidden prompt state, or unbounded reflective closure.*

Proof. Start with the displayed carrier order. Public compiler reads use A, M, C as the public audit and MetaCIC-audit prefix, so a public result cannot be obtained directly from the local rows.

Local self-audit reads use S, L, U only after the carrier has separated the public rows from the focused trace. The proxy-readback row U records the local endpoint without identifying it with the public evidence packet or the far endpoint.

The refusal row X is retained by both routes. It records that public authority for local reads, local authority for public compiler claims, hidden prompt state, and unbounded reflective closure are absent from the carrier. Transport and replay through E, H, K, P, N therefore preserve the same local-public boundary. □

Falsifiable prediction: A claimed public/local reflective compilation packet that exports public compiler authority from the local self-audit rows S, L, U without displaying A, M, C, X, H, K, P, N refutes $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$.

Independence witness: groundcompilereventflowaudit, metacicclosuretrace, selfcompilationboundary, focusedproxyreadback: none is bijective to $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ because the packet jointly carries public audit, MetaCIC audit, public reflective compiler, local self-compiler, local self-audit, proxy-readback, refusal, FieldFaithful, Nontrivial, transport, replay, provenance, and local naming rows.

The Lean TasteGate surface for $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ must provide explicit Nontrivial and FieldFaithful instances for $A, M, C, S, L, U, X, F, T, E, H, K, P, N$. The intended endpoint is BEDC.Derived.PublicLocalReflectiveCompilationUp.PublicLocalReflectiveCompilationTasteGate_single_carrier_alignm with the FieldFaithful instance preserving each carrier coordinate and the Nontrivial instance separating public audit, MetaCIC audit, public compiler, local self-compiler, local self-audit, proxy-readback, and refusal roles.

Closure status for $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$.

Constructive story: A $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ packet is generated from public compiler-side audit, MetaCIC-audit, public reflective compiler, local self-compiler, local self-audit, proxy-readback, refusal, FieldFaithful, Nontrivial, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3963.1, Definition 3963.2, and Definition 3963.3 seed the finite public/local reflective compilation route and its local-public boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted self-application, hidden prompt-state authority, public proof of local identity, local authority over public compiler claims, or unbounded reflective closure.

Upgrade path: Obligation closure requires checked carrier admission, public audit exposure, MetaCIC-audit exposure, local self-compiler confinement, local self-audit confinement, proxy-readback separation, refusal exactness, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.PublicLocalReflectiveCompilationUp.

Chapter 2987

A Concrete Naming Certificate for $\mathsf{MetricEntourage}^{\uparrow}$

Following Bourbaki uniform spaces and Bishop–Bridges constructive metric analysis, the $\mathsf{MetricEntourage}^{\uparrow}$ packet records the finite entourage surface between the metric row of chapter 57 and the uniform consumers of chapter 3445. Its tolerance reads are grounded through chapter 291, chapter 126, chapter 301, and chapter 38; it is a displayed metric-to-uniform handoff, not an open-cover compactness principle or a choice of centers.

Definition 2987.1 (MetricEntourage carrier)**.** *A* $\mathsf{MetricEntourage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, Q, D, B, S, T, H, C, P, N).$$

Here M is the $\mathsf{Met}^{\uparrow}$ *row, Q and D are the rational and dyadic tolerance rows, B is the ball-and-entourage witness row, S is the symmetry row, T is the triangle-composition row, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$ *row. The carrier has no coordinate for arbitrary open covers, selected centers, quotient points, countable choice, or ambient completion.*

Lean4 ✓ `BEDC.Derived.MetricEntourageUp.MetricEntourageTasteGate single carrier alignment`

Theorem 2987.2 (MetricEntourage NameCert obligations)**.** *For every accepted* $\mathsf{MetricEntourage}^{\uparrow}$ *carrier $E = (M, Q, D, B, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed metric, tolerance, entourage, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *diagonal and reflexive entourage admission from the positive tolerance rows Q, D and the ball witness row B;*
3. *symmetry transport through S together with the componentwise* hsame *row H;*
4. *triangle composition through T, using only the displayed metric row M and the rational or dyadic tolerance rows Q, D;*
5. *metric-to-uniform handoff saying that the uniform read consumes B, S, T after M, Q, D and before any downstream completion or Cauchy row;*
6. *transport and replay stability through H, C, P, N for the same displayed packet; and*

7. *nonescape from arbitrary open covers, selected centers, quotient points, countable choice, host compactness, or ambient completeness.*

Proof. Project the carrier of Definition 3964.1. The only metric-facing rows are M, Q, D, B, S, T, and the only structural rows are H, C, P, N.

The diagonal and reflexive obligation reads the tolerance rows before the ball witness row. Symmetry is then supplied by S, and triangle composition is supplied by T over the same metric and tolerance route.

The uniform handoff is therefore a finite row translation. A consumer of chapter 3445 may read the resulting entourage surface only after the metric row and tolerance rows have been displayed.

Finally the transport, replay, provenance, and naming rows preserve that route without adding any coordinate for open-cover compactness, selected centers, quotient representatives, countable choice, host compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricEntourage}^{\uparrow}$ packet is generated by reading a displayed metric row, rational and dyadic tolerance rows, a ball-and-entourage witness, symmetry, triangle composition, componentwise transport, replay, provenance, and local naming. It translates metric balls into a finite uniform entourage surface without adding open-cover compactness, selected centers, quotient points, countable choice, or ambient completion.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed records the metric row, rational and dyadic tolerance rows, ball-entourage witness, symmetry, triangle-composition, transport, replay, provenance, and local NameCert rows as the current paper-side scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MetricEntourageUp.MetricEntourageUp`

Bridge status: none

Not claimed: The chapter does not claim open-cover compactness, selected centers, quotient-point equality, countable choice, complete uniformity, or ambient metric completion.

Upgrade path: Next paper step: register the full MetricEntourage carrier obligation surface in Lean before raising the verification axis.

Chapter 2988

A Concrete Naming Certificate for $\mathsf{LeastUpperBound}^{\uparrow}$

Following Bishop and Bridges-style constructive order analysis, the $\mathsf{LeastUpperBound}^{\uparrow}$ packet records the located least-upper-bound surface needed by Real and Completion consumers before any ambient completeness theorem is available. It sits over the located upper-bound packet of chapter 3396, the located-supremum packet of chapter 1427, the dyadic supremum packet of chapter 1653, and the terminal real-name seal of chapter 38. The certificate is a finite boundary for bounded-source, upper-bound, and lower-approximation reads; it is not a quotient-real construction, a selected supremum, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2988.1 (LeastUpperBound carrier)**.** *A* $\mathsf{LeastUpperBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, U, A, L, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded source row, naming the finite family observations and the bound discipline consumed by the located upper-bound surface.*
2. *U is the upper-bound witness row, recording the inspected comparison witnesses that keep the displayed source below the proposed bound surface.*
3. *A is the lower-approximation and cofinality row, recording the finite witnesses demanded by the consumer below the same bound surface.*
4. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, binding the source, upper-bound, and lower-approximation reads to the located-supremum packet.*
5. *D is the* $\mathsf{DyadicSupremum}^{\uparrow}$ *handoff row, recording the dyadic upper-bound and lower-approximation surface used by finite tolerance consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after S, U, A, L, D have been displayed.*
7. *H records componentwise* hsame *transport for the bounded source, upper-bound witness, lower-approximation row, located-supremum handoff, dyadic-supremum handoff, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each least-upper-bound request through $S, U, A, L, D, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LeastUpperBound}^{\uparrow}$, $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{DyadicSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$ *row.*

The accepted carrier stores no ambient completeness coordinate, quotient cut, selected supremum, trichotomy oracle, host real equality test, or countable-choice selector.

Theorem 2988.2 (LeastUpperBound NameCert obligations)**.** *For every accepted* $\mathsf{LeastUpperBound}^{\uparrow}$ *carrier $B = (S, U, A, L, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-source exposure, upper-bound witness exposure, lower-approximation cofinality, located-supremum handoff, dyadic-supremum handoff, terminal Real seal, componentwise transport, displayed continuation replay, package provenance, and local naming. Every least-upper-bound-facing consumer reads only the displayed route*

$$S, U, A \longmapsto L, D \longmapsto E \longmapsto H, C, P, N.$$

Proof. Project the carrier of Definition 3965.1. The analytic rows are the bounded source S, the upper-bound witness U, the lower-approximation row A, the located-supremum handoff L, the dyadic-supremum handoff D, and the terminal Real seal E.

The upper side of the request is read through S and U. The lower side is read through A, and the cofinality claim is accepted only for the displayed approximation witnesses demanded by the consumer.

The handoff rows L and D bind those two sides to the existing located-supremum and dyadic-supremum surfaces before the Real seal can be inspected. The structural rows H, C, P, N only transport, replay, source, and name that finite route.

Since the carrier has no coordinate for quotient cuts, selected suprema, countable choice, trichotomy, host real equality, or ambient order completeness, no local NameCert obligation can export one of those principles. □

Lean4 ✓ `BEDC.Derived.LeastUpperBoundUp.LeastUpperBoundTasteGate single carrier alignment`

Theorem 2988.3 (LeastUpperBound located-supremum handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LeastUpperBound}^{\uparrow}$ *packet must first expose the bounded source row S, the upper-bound witness row U, the lower-approximation row A, the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row L, and the* $\mathsf{DyadicSupremum}^{\uparrow}$ *handoff row D before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The handoff exports only the finite least-upper-bound packet $S, U, A, L, D, E, H, C, P, N$.*

Proof. Start with a consumer that reaches the Real-facing boundary of the packet. By Definition 3965.1, the only terminal Real coordinate is E, and it is downstream of the bounded source, upper-bound, lower-approximation, located-supremum, and dyadic-supremum rows.

Apply Definition 3965.2. It forces every least-upper-bound-facing read through the route from S, U, A to L, D and then to E, H, C, P, N.

The located-supremum row keeps the upper and lower sides under the constructive located-supremum surface, while the dyadic-supremum row records the finite tolerance-facing boundary. Thus a Real or Completion consumer receives the displayed packet only, with no quotient-real construction, selected supremum, countable-choice selector, trichotomy split, host equality test, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{LeastUpperBound}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing read without first displaying the bounded source row, upper-bound witness row, lower-approximation row, $\mathsf{LocatedSupremum}^{\uparrow}$ handoff, and $\mathsf{DyadicSupremum}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: located upper bound, located supremum, dyadic supremum, and Real seal: no listed sibling supplies the whole $\mathsf{LeastUpperBound}^{\uparrow}$ packet, because each sibling contributes upper-bound discipline, located cofinality, dyadic tolerance data, or a terminal seal while this chapter binds all rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LeastUpperBound}^{\uparrow}$ packet is generated from a bounded source row, an upper-bound witness row, a lower-approximation cofinality row, LocatedSupremum and DyadicSupremum handoff rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3965.1, Definition 3965.2, and Definition 3965.3 seed the finite located least-upper-bound boundary needed by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real construction, selected suprema, countable choice, trichotomy, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, upper-bound witness exposure, lower-approximation cofinality, located-supremum handoff, dyadic-supremum handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LeastUpperBoundUp.

Chapter 2989

A Concrete Naming Certificate for $\mathsf{CauchyIntervalNesting}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyIntervalNesting}^{\uparrow}$ packet records the finite nested-interval-to-Cauchy route used when a retained interval window is read as regular rational data before the terminal real-name seal. It sits after the retained bisection surface of chapter 3815, the finite nested-chain discipline of chapter 320, the Cauchy carrier of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the finite route from displayed retained intervals to Cauchy-compatible readback; it does not assert selected infinite branches, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2989.1 (Cauchy interval nesting carrier)**.** *A* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, B, N, S, D, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the interval source row. It exposes the closed bounded source interval, endpoint order, containment evidence, and source finite windows consumed from the* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *and* $\mathsf{NestedInterval}^{\uparrow}$ *siblings.*

2. *B is the bisection branch row. It records the retained finite branch, midpoint exposure, child-containment witness, and dyadic branch ledger inherited from the accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *route.*

3. *N is the nested inclusion ledger. It records the visible finite chain, adjacent containment cells, width or diameter controls, and prefix discipline read through the* $\mathsf{NestedInterval}^{\uparrow}$ *surface.*

4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It displays the finite endpoint, midpoint, retained-child, and nested-prefix windows available to downstream requests.*

5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the positive dyadic scale, comparison cells, slack conversions, and diameter bounds used to interpret the displayed windows.*

6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from I, B, N, S, D and exposed only as a regular rational sequence surface.*

7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the retained branch, nested inclusion ledger, stream windows, dyadic tolerances, and regular rational readback have been displayed.*

8. *H records componentwise* hsame *transport for the interval source, bisection branch, nested ledger, stream window, dyadic tolerance, regular sequence, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a nested-interval Cauchy request through $I, B, N, S, D, R, E, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyIntervalNesting}^{\uparrow}$, $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $I, B, N, S, D, R, E, H, C, P, L$. It has no coordinate for a selected infinite branch, a completed intersection point, quotient stream equality, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyIntervalNestingUp.CauchyIntervalNestingTasteGate single carrier alignment`

Theorem 2989.2 (Cauchy interval nesting RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row only after reading the interval source, retained bisection branch, nested inclusion ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. Equivalently, every regular rational readback factors through the displayed finite route*

$$I, B, N, S, D \longmapsto R \longmapsto E, H, C, P, L.$$

The route is a Cauchy-facing finite readback route; it is not a selected infinite branch, a quotient real, a countable-choice schedule, or an ambient completeness argument.

Proof. Project the accepted carrier $X = (I, B, N, S, D, R, E, H, C, P, L)$ from Definition 3966.1. The source row I and branch row B expose the only retained closed-interval bisection data available to the request.

Next read the nested ledger N, the stream windows S, and the dyadic tolerance row D. These rows supply the finite prefix, displayed observations, and tolerance controls needed before the regular rational row can be inspected.

The row R is therefore downstream of I, B, N, S, D. The terminal seal and structural rows E, H, C, P, L preserve the same order by real sealing, transport, replay, provenance, and local naming.

Since the carrier has no coordinate for completed branches, quotient equality, choice, trichotomy, compactness, or ambient completeness, no consumer can bypass the displayed route. □

Theorem 2989.3 (Cauchy interval nesting NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier $X = (I, B, N, S, D, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$ *surface consists exactly of carrier admission, interval-source exposure, retained-bisection branch exposure, nested-inclusion ledger exposure, stream-window exposure, dyadic tolerance exactness, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and local naming for the finite carrier of Definition 3966.1.*

Proof. Carrier admission is projection from Definition 3966.1. The rows I, B, N expose the source interval, retained branch, and nested inclusion ledger by the same finite surfaces named in chapter 3815 and chapter 320.

The readback obligations are consecutive. The row S supplies only finite stream windows, the row D supplies only displayed dyadic tolerance cells, and Definition 3966.2 places the $\mathsf{RegSeqRat}^{\uparrow}$ row R after those data.

The terminal row E is the only $\mathsf{Real}^{\uparrow}$ seal-facing coordinate. The rows H, C, P, L then transport, replay, source, and name the same route while preserving the $\mathsf{Cauchy}^{\uparrow}$ dependency boundary of chapter 3413.

No listed obligation introduces a selected infinite branch, quotient real equality, countable choice, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects occurs as a carrier coordinate. □

Falsifiable prediction: Every $\mathsf{CauchyIntervalNesting}^{\uparrow}$ consumer that skips either the retained branch row or the nested inclusion ledger loses the RegSeqRat readback before any Real seal can be read.

Independence witness: closed-interval-bisection, nestedinterval, cauchy, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyIntervalNesting}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval source, retained bisection branch, nested inclusion ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntervalNesting}^{\uparrow}$ packet is generated from an interval source row, a retained bisection branch row, a nested inclusion ledger, StreamName finite windows, a DyadicRatCore tolerance row, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3966.1, Definition 3966.2, and Definition 3966.3 seed the finite retained-interval-to-Cauchy readback route over ClosedIntervalBisectionUp, NestedIntervalUp, CauchyUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows, including the interval source, retained bisection branch, nested inclusion ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and RealUp seal, without selected branches, completed intersection points, quotient stream equality, quotient real equality, countable choice, or ambient RealUp completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, completed intersection points, quotient stream equality, quotient real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained branch exposure, nested inclusion routing, StreamName window exactness, DyadicRatCore tolerance accounting, RegSeqRat readback, Real seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.CauchyIntervalNestingUp.

Chapter 2990

A Concrete Naming Certificate for $\mathsf{FormalTopology}^{\uparrow}$

Following Vickers formal topology and Bishop–Bridges constructive completion, the $\mathsf{FormalTopology}^{\uparrow}$ packet records a cover-and-refinement presentation whose finite BEDC rows connect formal-ball neighbourhoods to rounded ideal completion without importing quotient opens, selected points, choice, or ambient completeness. It sits between the formal-ball carrier of chapter 1510, the rounded ideal completion route of chapter 1883, the metric source of chapter 57, the dyadic radius ledger of chapter 291, and the stream-window interface of chapter 126.

Definition 2990.1 (FormalTopology carrier)**.** *A* $\mathsf{FormalTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, B, U, R, D, S, I, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose point histories and distance witnesses are used by every formal-ball centre.*
2. *B is the finite* $\mathsf{FormalBall}^{\uparrow}$ *neighbourhood row over M, carrying centres, positive radii, and the local ball classifier.*
3. *U is the finite cover/refinement ledger: every cover request names finitely many displayed formal-ball neighbourhoods and every refinement request records the source and target neighbourhood rows it compares.*
4. *R is the rounded-neighbourhood row, recording smaller formal-ball witnesses for accepted basic neighbourhoods.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius-comparison ledger for strict radius shrinkage, overlap budgets, and refinement inequalities.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect centre observations, cover requests, and rounded-ideal membership windows.*
7. *I is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *handoff row, exposing only rounded directed formal-ball membership data to completion-facing consumers.*
8. *H records componentwise* hsame *transport for metric rows, formal-ball neighbourhoods, cover/refinement ledgers, dyadic comparisons, stream windows, rounded-ideal handoff rows, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from finite cover requests through refinement, roundedness, radius comparison, and rounded-ideal completion handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient opens, selected points, countable choice, host topological equality, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.FormalTopologyUp.FormalTopologyTasteGate single carrier alignment`

Theorem 2990.2 (FormalTopology cover stability)**.** *For an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier* $T = (M, B, U, R, D, S, I, H, C, P, N)$*, every finite cover or refinement read factors through the same displayed route*

$$M, B \longmapsto U, R, D, S \longmapsto H, C, P, N.$$

Componentwise hsame *transport through H preserves the cover ledger, the rounded-neighbourhood witnesses, the dyadic radius comparisons, the stream windows, and the local naming row. No transported read may replace the finite cover ledger by a quotient-open classifier, a selected point, a choice function, an open-cover compactness principle, or an ambient completeness theorem.*

Theorem 2990.3 (FormalTopology rounded-ideal handoff)**.** *Let $T = (M, B, U, R, D, S, I, H, C, P, N)$ be an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier. Every completion-facing consumer of its neighbourhood data must pass through the rounded-ideal row I after the formal-ball neighbourhoods B, the cover/refinement ledger U, the roundedness row R, the dyadic comparison ledger D, and the stream-window row S have been displayed.*

Thus the handoff to chapter 1883 exports a finite rounded directed formal-ball surface and exports no quotient open, selected limit point, countable-choice schedule, host topological equality, open-cover compactness proof, or ambient complete-space coordinate.

Theorem 2990.4 (FormalTopology NameCert obligations)**.** *Let $T = (M, B, U, R, D, S, I, H, C, P, N)$ be an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier. The local* $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$ *surface of T is exactly the finite obligation package consisting of carrier admission for T, finite* $\mathsf{FormalBall}^{\uparrow}$ *cover and refinement admission, rounded-neighbourhood witnesses,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius-comparison exactness,* $\mathsf{StreamName}^{\uparrow}$ *finite-window replay,* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier row of Definition 3967.1. The displayed tuple first fixes M as the metric source and B as the formal-ball neighbourhood carrier, so carrier admission and finite formal-ball admission are read before any completion-facing consumer is visible.

Next read the rows U, R, D, S, I in order. The cover and refinement ledger is U, rounded-neighbourhood evidence is R, dyadic radius comparison is D, finite-window replay is S, and the rounded-ideal completion handoff is I.

The remaining rows H, C, P, N close the certificate surface. They supply componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local naming row, and no additional coordinate appears in the carrier definition. □

Theorem 2990.5 (FormalTopology ledger non-escape)**.** *For an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier* $T = (M, B, U, R, D, S, I, H, C, P, N)$*, every cover, refinement, and completion-facing consumer is exhausted by the rows named in Definition 3967.4. No such consumer can obtain quotient opens,*

selected points, countable choice, open-cover compactness, host topology equality, or ambient metric completeness from the $\mathsf{FormalTopology}^{\uparrow}$ *ledger.*

Proof. Use the same projection of Definition 3967.1. A cover or refinement read must enter through B and U, then pass through R, D, S when it asks for roundedness, radius comparison, or finite observation replay.

A completion-facing read must additionally pass through I. After that handoff, the only public bookkeeping rows are H, C, P, N, which transport, replay, record provenance, and name the already displayed finite rows.

The last sentence of the carrier definition lists the absent coordinates. Since no row of T contains a quotient-open classifier, selected point, countable-choice schedule, open-cover compactness proof, host topology equality, or ambient completeness witness, a consumer constrained to T cannot extract one. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, finite FormalBall neighbourhoods, a cover/refinement ledger, rounded-neighbourhood witnesses, DyadicRatCore radius comparisons, StreamName windows, rounded-ideal completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3967.1, Definition 3967.2, Definition 3967.3, Definition 3967.4, and Definition 3967.5 seed the finite formal-topology cover/refinement route from formal-ball neighbourhoods to rounded ideal completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient opens, selected points, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover/refinement stability, dyadic radius comparison, StreamName replay, rounded-ideal handoff, transport, provenance, local naming, and ledger non-escape for the same packet.

Chapter 2991

A Concrete Naming Certificate for $\mathsf{FormalBallRadiusOrder}^{\uparrow}$

Following Edalat–Heckmann and Vickers-style formal-ball presentations, the $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ packet isolates the radius-refinement order that is consumed by the formal-ball carrier of chapter 1510, the rounded-basis route of chapter 3943, and the completion-facing handoff of chapter 2012. Its role is to name the finite BEDC comparison surface for smaller-radius reads before any completion quotient, selected representative, or ambient completeness principle is available.

Definition 2991.1 (FormalBall radius-order carrier)**.** *A* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *carrier row whose center, radius, dyadic ledger, finite window, replay, provenance, and local name are already displayed.*
2. *R is the radius-read row, taken from the* $\mathsf{RegSeqRat}^{\uparrow}$ *scale attached to the formal-ball packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger recording strict and weak radius comparisons used by the refinement order.*
4. *L is the rounded lower-neighborhood row, recording the smaller-radius formal-ball windows that remain inside the same carrier route.*
5. *H records componentwise* hsame *transport for formal-ball carriers, radius rows, dyadic ledgers, lower-neighborhood rows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from radius comparison to rounded refinement, basis handoff, and completion boundary.*
7. *P is the* Pkg *provenance over* $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of completed balls, a selected Cauchy representative, a countable-choice schedule, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallRadiusOrderUp.FormalBallRadiusOrderTasteGate single carrier alignment`

Theorem 2991.2 (FormalBall radius-order NameCert obligations)**.** *For the carrier of Definition 3968.1, the local* $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for* $O = (B, R, D, L, H, C, P, N)$*, dyadic radius comparison through* D*, rounded refinement through* L*, basis handoff through the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row named by* P*, completion non-escape through the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *boundary, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Every accepted radius-order read must first expose the underlying $\mathsf{FormalBall}^{\uparrow}$ *packet* B*, then compare radii through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, and only then hand the rounded refinement to basis or completion consumers. It cannot read a quotient completion equality, choose a representative, select a limit, use countable choice, invoke compactness, or assume completeness.*

Proof. Carrier admission is projection from Definition 3968.1. The tuple displays the formal-ball packet, radius row, dyadic comparison ledger, lower-neighborhood row, transport, replay, provenance, and local name.

Dyadic radius comparison is confined to R and D. The formal-ball row B supplies the source center-radius packet, while D records the finite comparison data used to recognize a smaller radius.

Rounded refinement and basis handoff are then read through L and C. The lower-neighborhood row records the smaller formal-ball window, and replay sends that window to the basis and completion-facing rows carried in P.

Completion non-escape follows from row exhaustion. Since the carrier has no quotient-completion equality, selected representative, selected limit, countable-choice schedule, compactness principle, or ambient complete-space coordinate, none of those objects can be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ packet is generated from a displayed FormalBallUp carrier, a RegSeqRat radius row, a DyadicRatCore comparison ledger, a rounded lower-neighborhood row, componentwise hsame transport, displayed Cont replay toward basis and completion consumers, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$, seed)

Scope closed: Definition 3968.1 and Definition 3968.2 seed the finite radius-refinement order route consumed by formal-ball basis and completion packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic comparison stability, rounded-refinement replay, basis handoff, completion non-escape, transport, provenance, and local naming for the same packet.

Chapter 2992

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite approximation, the $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ packet records the compactness of a closed bounded interval as finite NameCert data. It consumes the closed bounded interval source of chapter 1591, the finite-net cover route of chapter 3471, the interval uniform-modulus source of chapter 1253, the compact-metric carrier of chapter 129, and the endpoint real-name seal of chapter 38. The packet is a displayed compact interval certificate for downstream finite-net, Bolzano-Weierstrass, and compact-uniform-continuity consumers; it is not an unrestricted open-cover compactness theorem, a selected subcover functional, a quotient interval representative, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 2992.1 (Closed bounded interval compact carrier)**.** *A* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, U, M, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment, stream windows, regular rational readback, and Real sealing.*
2. *F is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite-net compactness row, recording the listed cover witnesses and radius comparisons consumed by finite-cover readers.*
3. *U is the* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *sibling row, recording the compact-uniform source route over the same interval source and finite cover data.*
4. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric compactness row, read only through its finite-net, total-boundedness, and complete-limit components.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint and radius seal reached only after I, F, U, M have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, finite-net row, uniform-modulus row, compact-metric row, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from closed bounded interval inspection through finite-net compactness, compact-uniform source routing, compact-metric readback, and Real sealing.*

8. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, M, E, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalCompactUp.ClosedBoundedIntervalCompactTasteGate`
`single carrier alignment`

Theorem 2992.2 (Closed bounded interval compact NameCert obligations)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *carrier $K = (I, F, U, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the interval source, finite-net compactness row, uniform-modulus sibling row, compact-metric row, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-source exposure, requiring every compactness read to display the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row before inspecting finite-net or compact-metric data;*
3. *finite-net compactness exposure, requiring cover-facing reads to pass through the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *row and its displayed radius and cover-witness ledgers;*
4. *compact-uniform compatibility, requiring the* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *rows to be read as sibling routes over the same interval source and Real seal;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no unrestricted open-cover compactness, selected subcover, selected point, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is projection from Definition 3969.1. The tuple displays exactly the interval source, finite-net compactness row, compact-uniform sibling row, compact-metric row, Real seal, and structural rows.

The interval source is upstream. Any compactness reader must first read I, so endpoint order, rational containment, dyadic containment, stream windows, regular rational readback, and the Real-facing source seal are fixed before finite-net or compact-metric data are inspected.

The finite-net compactness obligation is the row F. Its cover witnesses and radius comparisons are the only cover-facing compactness data in the carrier, and they are consumed through the Heine-Borel finite-net route rather than through a host open-cover theorem.

The rows U and M are sibling compatibility rows. They let compact-uniform-continuity and compact-metric consumers use the same interval source and Real seal, but neither row changes the source interval or adds a selected subcover, selected point, quotient equality, trichotomy, choice, or ambient completeness coordinate.

Finally, H, C, P, N preserve only the displayed rows. Since every refused object is absent from the carrier, the local NameCert surface is exactly the five obligations listed above. □

Theorem 2992.3 (Closed bounded interval compact finite-net route)**.** *Every compactness-facing consumer of an accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *row is available as the compact-uniform sibling route over the same interval and finite-net data. Thus* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *supplies a precise*

compact interval anchor for finite-net compactness, Bolzano-Weierstrass, and compact-uniform-continuity consumers without exporting open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalCompactUp.ClosedBoundedIntervalCompactPacket finite net route`

Proof. Project the accepted carrier $K = (I, F, U, M, E, H, C, P, N)$. The interval source is I, the finite-net compactness route is F, the compact-metric row is M, and the terminal Real seal is E.

Replay through C keeps this order. A compactness-facing request opens the closed bounded interval source, reads the finite-net cover and radius row, then inspects the compact-metric finite-net and limit components before it reaches the Real-facing seal.

The row U records the compatible uniform-modulus route for consumers that need compact uniformity. It remains tied to the same interval source, finite-net row, compact-metric row, and Real seal, so it cannot introduce an unrestricted open-cover theorem, selected subcover, quotient equality, countable choice, or ambient completeness coordinate. □

Theorem 2992.4 (Closed bounded interval compact finite-net fan)**.** *Every Bolzano-Weierstrass, compact-uniform-continuity, and finite-cover consumer of an accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *packet must pass through the displayed finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *sibling row before reading any Real seal. The fan exports only the interval source, finite-net compactness row, compact-uniform sibling row, compact-metric row, Real seal, transport, replay, provenance, and local naming rows; it exports no open-cover compactness, selected subcover, selected point, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier $K = (I, F, U, M, E, H, C, P, N)$. The rows I, F, M, E are precisely the closed bounded interval source, Heine-Borel finite-net row, compact-metric row, and terminal Real seal displayed in Definition 3969.3.

Read the consumer fan by cases. A finite-cover consumer uses F; a Bolzano-Weierstrass consumer uses the compact-metric row M only after F has exposed the finite-net route; a compact-uniform-continuity consumer also reads the sibling row U over the same interval and finite-net data.

The structural rows H, C, P, N preserve this order for every branch of the fan. Since all refused objects are absent from the carrier, no branch can bypass the displayed route or read the Real seal before the interval, finite-net, compact-metric, and uniform-modulus rows have been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a HeineBorelFiniteNet finite-net compactness row, a ClosedBoundedIntervalUniformModulus sibling row, a CompactMetric row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3969.1, Definition 3969.2, and Definition 3969.3 seed the finite compact interval certificate over closed bounded interval, Heine-Borel finite-net, compact-uniform, compact-metric, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, finite-net compactness exposure, compact-uniform compatibility, compact-metric compatibility, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalCompactUp.

Chapter 2993

A Concrete Naming Certificate for $\mathsf{FiniteSequencePeak}^{\uparrow}$

Following Hardy, Littlewood, and Polya on finite maxima and elementary rearrangement arguments, the $\mathsf{FiniteSequencePeak}^{\uparrow}$ packet records the constructive selector surface for a bounded finite sequence before any L10 bounded-window route reads a peak value. It sits over the finite list carrier of chapter 36, the comparison discipline of chapter 54, the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite peak row, bounded-window evidence, comparison replay, transport, provenance, and local naming data; it does not assert sequential compactness, countable choice, host extrema, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2993.1 (Finite sequence peak carrier)**.** *A* $\mathsf{FiniteSequencePeak}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, O, D, W, R, E, H, C, G, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{List}^{\uparrow}$ *row naming the finite sequence entries and their displayed length.*
2. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row used to compare only entries already present in L.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row carrying the finite comparison and error tolerances.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *bounded-window row that records the inspected segment supplied to L10 consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the bounded window and dyadic budgets into a regular rational observation of the selected peak.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after L, O, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for L, O, D, W, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from a bounded-window request through*

$$L, O, D, W \longmapsto R \longmapsto E.$$

9. *G is the* Pkg *provenance over* $\mathsf{FiniteSequencePeak}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^\uparrow}$ *row.*
The classifier compares accepted packets only through $L, O, D, W, R, E, H, C, G, N$. The carrier has no coordinate for an infinite subsequence selector, sequential compactness theorem, host maximum operator, quotient stream representative, selected real limit, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.FiniteSequencePeakUp.TasteGate.FiniteSequencePeakTasteGate single carrier alignment`

Theorem 2993.2 (Finite sequence peak NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSequencePeak}^\uparrow$ *carrier $P = (L, O, D, W, R, E, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^\uparrow}$ *surface consists exactly of finite list admission through L, total-order comparison through O, dyadic budget accounting through D, bounded-window exposure through W, regular rational readback through R, terminal* $\mathsf{Real}^\uparrow$ *seal handoff through E, componentwise transport through H, replay through C, provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 3970.1. The analytic rows are exactly the finite list row, comparison row, dyadic budget row, bounded-window row, regular rational readback, and Real seal.

The selector route is finite. The consumer first reads the entries named by L, compares only those entries through O, and spends only the dyadic tolerances recorded in D before a bounded window is available.

The bounded-window row W is the only source for the regular rational readback R. Thus the Real-facing endpoint E is downstream of $\mathsf{List}^\uparrow$, $\mathsf{TotalOrder}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, and $\mathsf{RegSeqRat}^\uparrow$ data.

Finally H, C, G, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for sequential compactness, countable choice, host extrema, quotient streams, selected limits, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 2993.3 (Finite sequence peak RegSeqRat selector handoff)**.** *Every L10 bounded-window consumer of an accepted* $\mathsf{FiniteSequencePeak}^\uparrow$ *carrier reads the route*

$$\mathsf{List}^\uparrow \longmapsto \mathsf{TotalOrder}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The selected peak is exported to $\mathsf{RegSeqRat}^\uparrow$ *only as a finite-window readback: the list row supplies the candidate entries, the order row supplies the displayed comparisons, the dyadic row supplies the budget, and the stream row supplies the bounded inspection window before the terminal Real seal is exposed.*

Proof. Begin with an accepted carrier $P = (L, O, D, W, R, E, H, C, G, N)$. The row L lists the available entries, and O compares only entries already displayed by that finite list.

Read the dyadic and window rows next. The budget row D records the finite comparison tolerances, while W records the bounded segment consumed by the L10 route.

The regular rational row R is downstream of L, O, D, W. Therefore a $\mathsf{RegSeqRat}^\uparrow$ consumer receives a bounded-window readback and cannot replace it by a host maximum, an infinite subsequence selector, or a selected limit.

The terminal seal E is reached only after the readback route has been displayed. Transport, replay, provenance, and local naming through H, C, G, N preserve that same route, so no sequential compactness or ambient completeness coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3970.1, Definition 3970.2, and Definition 3970.3 seed the finite peak selector route for L10 bounded-window consumers over List, TotalOrder, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sequential compactness, countable choice, host extrema, quotient stream equality, selected real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite comparison coverage, bounded-window exposure, DyadicRatCore budget accounting, RegSeqRat readback exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSequencePeakUp.

Constructive story: A $\mathsf{FiniteSequencePeak}^{\uparrow}$ packet is generated from a List row, TotalOrder comparison row, DyadicRatCore budget row, StreamName bounded-window row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 2994

A Concrete Naming Certificate for $\mathsf{LocatedIntervalBolzano}^{\uparrow}$

Following Bolzano and Bishop–Bridges constructive analysis, the $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ packet records the finite interval-zero route used when a continuous located-real map changes sign on a located closed interval. It sits over the located real interval carrier of chapter 1709, the finite-net interval coverage rows of chapter 1808, the continuous-map replay of chapter 130, the intermediate-value route of chapter 3938, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names an interval-zero certificate from finite endpoint, bisection, stream, regular-rational, and real-seal data; it does not export trichotomy, a selected zero, quotient equality of real names, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 2994.1 (Located interval Bolzano zero carrier)**.** *A* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ *zero carrier is a finite* BHist *packet*

$$Z = (I, F, E, N, B, S, D, R, Q, H, C, P, L)$$

with the following displayed rows:

1. *I is the located-real interval source row, exposing the two endpoint seals, interval order witness, and located containment ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row for the map inspected on the interval.*
3. *E is the endpoint sign row, displaying located negative and positive observations for the two endpoint values.*
4. *N is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *coverage row used to keep bisection requests inside the displayed interval.*
5. *B is the finite bisection row, recording the retained subinterval side, midpoint request, and sign handoff at every inspected stage.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for endpoint, midpoint, and retained-interval observations.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the endpoint gaps, midpoint scale, and retained interval radius.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance ledger.*

9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after I, F, E, N, B, S, D, R have been displayed.*
10. *H records componentwise* hsame *transport for the analytic rows and for the local structural rows.*
11. *C records displayed* Cont *replay from the sign-change request through finite bisection, stream windows, regular rational readback, and the terminal seal.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedIntermediateValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}$ *row.*

The classifier compares accepted zero carriers only through $I, F, E, N, B, S, D, R, Q, H, C, P, L$. It has no coordinate for real trichotomy, an externally selected zero, quotient equality of real names, host compactness, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalBolzanoUp.LocatedIntervalBolzanoTasteGate single carrier alignment`

Theorem 2994.2 (Located interval Bolzano sign handoff)**.** *For every accepted* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ *zero carrier*

$$Z = (I, F, E, N, B, S, D, R, Q, H, C, P, L),$$

the sign-change request is handed to completion-facing consumers only through the displayed route

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{LocatedIntermediateValue}^{\uparrow} \longmapsto \mathsf{S}$$

The terminal zero read is therefore a finite interval certificate consisting of endpoint signs, retained bisection rows, stream windows, dyadic tolerances, regular-rational readbacks, a real seal, componentwise transport, replay, provenance, and local naming. It is not a selected point, quotient real representative, trichotomy comparison, host compactness argument, countable-choice schedule, or ambient real-completeness principle.

Proof. Project the carrier of Definition 3971.1. The source interval row I, continuous-map row F, endpoint sign row E, finite-net row N, and bisection row B are the only interval-zero coordinates available before the stream-facing rows are read.

The bisection route is finite. At every inspected stage the retained side and midpoint request are displayed together with the finite coverage row, so the sign handoff stays inside the located closed interval and never asks for a global trichotomy decision.

Read S, D, R, Q next. The stream windows expose exactly the finite endpoint and midpoint observations, the dyadic ledger records their tolerance budget, the regular rational row reads those observations as regular data, and the terminal real seal is downstream of those displayed rows.

Finally H, C, P, L transport, replay, source, and name the same route. Since the packet contains no coordinate for selected zeros, quotient real equality, host compactness, countable choice, Lean verification, bridge checking, or ambient Real completeness, the displayed rows exhaust the sign handoff. □

Falsifiable prediction: A completion-facing interval-zero consumer that omits the located interval row, finite-net coverage row, continuous-map replay row, bisection row, StreamName window row, DyadicRatCore tolerance ledger, RegSeqRat readback, or RealUp seal cannot claim a $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ zero handoff.

Independence witness: located-real-interval, located-closed-interval-finite-net, continuousmap, intermediatevalue, streamname, dyadicratcore, regseqrat, real: $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint signs, finite interval coverage, continuous-map replay, retained bisection data, stream windows, dyadic tolerances, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ packet is generated from a located-real interval row, finite endpoint sign rows, a LocatedClosedIntervalFiniteNet coverage row, ContinuousMap replay, a finite bisection ledger, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3971.1 and Definition 3971.2 seed the constructive interval-zero route over located real intervals, finite interval nets, continuous maps, bisection, stream windows, regular rational readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedIntervalBolzanoUp.LocatedIntervalBolzanoTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close real trichotomy, selected zeros, quotient real equality, host compactness, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exposure, finite-net interval coverage, continuous-map replay, bisection ledger exactness, StreamName window exposure, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalBolzanoUp.

Chapter 2995

A Concrete Naming Certificate for $\mathsf{BishopRealLocatedOrder}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers and located order, the $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ packet records the order-facing interface needed by L10 real and completion consumers. It sits between the Bishop real packet of chapter 343, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the dyadic readback of chapter 291, the stream windows of chapter 126, the regular-rational route of chapter 301, and the Archimedean real interface of chapter 430. The packet supplies displayed located-order and apartness rows for RealAlgOrder, monotone convergence, and completion-facing handoffs; it does not assert trichotomy, quotient real equality, host equality, selected suprema, or ambient completeness.

Definition 2995.1 (Bishop real located order carrier)**.** *A* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, R, D, S, Q, A, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopReal}^{\uparrow}$ *source row, exposing the regular-Cauchy real name and its Bishop apartness interface.*
2. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Bishop source and its readback rows have been displayed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison row used for rational bracketing.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite name observations are requested.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows and dyadic tolerances.*
6. *A is the apartness-stability row, recording that positive rational separation is transported through the same finite readback route.*
7. *O is the located-order row, recording finite lower-or-upper rational alternatives only as displayed order evidence.*
8. *H records componentwise* hsame *transport for the Bishop source, Real seal, dyadic row, stream windows, regular readback, apartness row, located-order row, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from Bishop source inspection through stream windows, dyadic tolerance, regular readback, apartness, located-order evidence, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$, $\mathsf{BishopReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, D, S, Q, A, O, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, host equality, selected suprema, a least-upper-bound operator, or ambient completeness.

Theorem 2995.2 (Bishop real located order NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *carrier $L = (B, R, D, S, Q, A, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; Bishop source exposure through B; Real sealing through R; dyadic tolerance and comparison readback through D; finite stream-window exposure through S; regular-rational readback through Q; apartness stability through A; located-order evidence through O; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.BishopRealLocatedOrderUp.BishopRealLocatedOrderTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3972.1. The source rows are exactly B, S, D, Q, R, so every order-facing read begins at the Bishop real source and reaches the Real seal only through displayed stream, dyadic, and regular-rational rows.

Read the order rows next. Apartness stability is the row A, and located-order evidence is the row O; neither row contains a trichotomy decision, a quotient equality proof, a selected supremum, or a host completeness principle.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite packet. Since the refused host-level objects have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 2995.3 (Bishop real located order apartness stability)**.** *Every accepted apartness read from a* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *packet factors through*

$$B, S, D, Q, A, H, C, P, N$$

before it can be consumed by a $\mathsf{Real}^{\uparrow}$ *or completion-facing route. Thus positive separation is stable under the displayed* hsame *and* Cont *rows, but the read exports no quotient real equality, host equality, trichotomy, selected supremum, or ambient completeness theorem.*

Proof. Start with the Bishop source B. Its apartness evidence is not read directly by a host order relation; it is exposed through the finite stream windows S, the dyadic tolerance row D, and the regular-rational readback row Q.

Apply the apartness row A. This row records positive rational separation along that same route, so the consumer may transport a displayed separation only when the source, window, dyadic, and readback rows remain attached.

Use H and C for classifier movement. They preserve the displayed route and do not add a quotient-equality or trichotomy coordinate. Hence every accepted apartness read remains inside $B, S, D, Q, A, H, C, P, N$. □

Theorem 2995.4 (Bishop real located order regular-sequence scope)**.** *Every regular-sequence consumer of an accepted* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *packet must read the located-order interface through*

$$\mathsf{BishopReal}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{ArchimedeanReal}^{\uparrow}$$

as displayed by $B, S, D, Q, R, O, H, C, P, N$*. The exported scope is a constructive located-order handoff for RealAlgOrder, monotone convergence, and completion routes; it is not a trichotomy principle, a quotient real equality rule, a host equality rule, a selected-supremum package, or an ambient completeness assertion.*

Proof. Read the regular-sequence route through S, D, Q. These rows expose the finite stream windows, dyadic tolerances, and regular-rational readback before the terminal Real seal R can be used.

The located-order row O is downstream of that route and is sourced by the Bishop real row B together with the Archimedean provenance recorded in P. It therefore supplies order evidence only as a displayed finite handoff, not as a host-level order totality theorem.

The structural rows H, C, N preserve the same scope under accepted classifier movement. Since no carrier coordinate names trichotomy, quotient equality, host equality, selected suprema, or ambient completeness, regular-sequence consumers receive only the constructive located-order handoff. □

Falsifiable prediction: Every completion-facing $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ read factors through displayed BishopReal, StreamName, DyadicRatCore, RegSeqRat, RealUp, apartness, located-order, transport, replay, provenance, and local naming rows before it can be used as constructive order evidence.

Independence witness: bishopreal, real, dyadicratcore, streamname, regseqrat, archimedean-real: $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries apartness stability and located-order evidence over the same finite Bishop real readback route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ packet is generated from a BishopReal source row, RealUp seal, DyadicRatCore tolerance row, StreamName windows, RegSeqRat readback, apartness stability, located-order evidence, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$, seed)

Scope closed: Definition 3972.1, Definition 3972.2, Definition 3972.3, and Definition 3972.4 seed the constructive located-order interface for Bishop real, regular-sequence, and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopRealLocatedOrderUp.BishopRealLocatedOrderTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host equality, selected suprema, ambient completeness, or bridge checking.

Upgrade path: Lean closure requires NameCert obligations, apartness stability, regular-sequence scope, and TasteGate field faithfulness for BEDC.Derived.BishopRealLocatedOrderUp.

Chapter 2996

A Concrete Naming Certificate for $\mathsf{NestedCompactIntersection}^{\uparrow}$

Following Bishop–Bridges and Kelley compactness-completion practice, the $\mathsf{NestedCompactIntersection}^{\uparrow}$ packet records the finite route by which a nested family of nonempty compact metric carriers can be read by later completion-facing consumers without asserting an open-cover compactness theorem or selecting a point of the total intersection. It sits after the compact metric carrier of chapter 129, the completion reuse boundary of chapter 2381, the separated regular-Cauchy handoff of chapter 1513, and the terminal real-name seal of chapter 38. The packet names a finite L10 compactness/completion surface: retained nested windows, finite-intersection witnesses, separated-completion readback, and a terminal Real seal, but no choice-selected intersection element and no ambient Real completeness principle.

Definition 2996.1 (Nested compact intersection carrier)**.** *A* $\mathsf{NestedCompactIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, J, W, F, S, R, E, H, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row for each compact stage of the nested family.*
2. *J is the finite nesting-order ledger, recording displayed inclusions between later and earlier compact stages.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *retained-window row used to inspect only finitely many stages at any requested tolerance.*
4. *F is the finite-intersection witness row, showing that each retained finite prefix has a displayed nonempty witness inside the visible compact stages.*
5. *S is the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *handoff row that consumes the retained windows only after compact-stage and finite-prefix data have been displayed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the separated completion handoff.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after C, J, W, F, S, R have been read.*
8. *H records componentwise* hsame *transport for compact stages, nesting ledgers, windows, finite-prefix witnesses, separated-completion rows, readbacks, the Real seal, provenance, and local naming.*

9. *P is the* Pkg *provenance over* NestedCompactIntersection$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompletionIdempotence$^{\uparrow}$, RegularCauchyCompletionSeparated$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected intersection points, quotient stream equality, countable choice, host set equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedCompactIntersectionUp.NestedCompactIntersectionTasteGate single carrier alignment`

Theorem 2996.2 (NestedCompactIntersection NameCert obligations)**.** *For every accepted* NestedCompactIntersect *carrier $K = (C, J, W, F, S, R, E, H, P, N)$, the local* NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for compact metric stages, the nesting-order ledger, retained StreamName windows, finite-intersection witnesses, separated-completion handoff, regular rational readback, terminal Real seal, transport, provenance, and local naming;*

2. *nested-family exactness, requiring each later compact-stage read to be justified by the displayed nesting ledger J before it is used by a retained window;*

3. *finite-intersection witness stability, requiring every retained finite prefix to expose its witness through F and to transport through the same compact-stage and window rows;*

4. *separated-completion handoff, requiring every completion-facing read to pass through S and R after C, J, W, F have been projected;*

5. *Real seal nonescape, requiring E to remain downstream of the separated-completion and regular-readback rows, with no open-cover compactness, selected point, quotient equality, countable choice, host set equality, or ambient completeness coordinate.*

Proof. Carrier admission is projection from Definition 3973.1. The displayed tuple exposes compact metric stages, nesting ledgers, finite windows, prefix witnesses, separated-completion rows, readback, Real seal, transport, provenance, and local naming.

Nested-family exactness is the route C, J, W. A consumer may inspect a later compact stage through a retained window only after the relevant inclusion row in J has been displayed, so the nesting relation is part of the packet rather than an ambient set-theoretic assumption.

Finite-intersection witness stability is the route C, J, W, F, H. The witness row F is attached to a retained finite prefix, and the transport row H moves the compact-stage, nesting, window, and witness data componentwise without adding selected points beyond the displayed prefix witness.

The completion-facing part is last. The separated-completion row S and regular readback row R consume the compact and finite-prefix rows before the terminal Real seal E is available.

Since the carrier has no open-cover compactness object, global intersection selector, quotient stream equality, countable-choice schedule, host set equality, or ambient Real$^{\uparrow}$ completeness coordinate, none of those objects can be exported by the local NameCert surface. □

Closure status for NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$, seed)

Scope closed: Definition 3973.1 and Definition 3973.2 seed the finite nested compact intersection route over CompactMetric stages, nesting ledgers, retained StreamName windows, finite-prefix witnesses, separated-completion handoff, RegSeqRat readback, RealUp seal, hsame transport, Pkg provenance, and local NameCert data.

Formal verification: FormalStatus(NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NestedCompactIntersectionUp.NestedCompactIntersectionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected intersection points, quotient stream equality, countable choice, host set equality, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires nested-family exactness, finite-intersection witness stability, separated-completion handoff, and RealUp seal nonescape for BEDC.Derived.NestedCompactIntersectionUp.

Constructive story: A NestedCompactIntersection$^{\uparrow}$ packet is generated from CompactMetric stage rows, a finite nesting-order ledger, retained StreamName windows, finite-intersection witnesses, a RegularCauchyCompletionSeparated handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Chapter 2997

A Concrete Naming Certificate for $\mathsf{MercerTheorem}^{\uparrow}$

Following Mercer 1909 and the compact-operator spectral tradition represented in Reed–Simon, the $\mathsf{MercerTheorem}^{\uparrow}$ packet records the finite BEDC surface for a positive continuous kernel whose compact integral-operator shadow is read through matrix windows and Hilbert-space coordinates. It sits beside the compact-operator carrier of chapter 1952, the Hilbert package of chapter 91, the inner-product source of chapter 86, the finite matrix surface of chapter 49, and the terminal real-name seal of chapter 38. The packet names only a finite positive-kernel and RKHS readback certificate; it does not assert ambient measure theory, choice-selected eigenbases, unrestricted integral operators, or the full spectral theorem.

Definition 2997.1 (Mercer theorem carrier)**.** *A* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, O, F, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact positive-kernel source row, recording a symmetric positive kernel request together with its compact-domain and continuity ledgers.*
2. *O is the* $\mathsf{CompactOperator}^{\uparrow}$ *shadow row, read only as a compact positive operator generated by the displayed kernel source.*
3. *F is the finite* $\mathsf{Mat}^{\uparrow}$ *window family: Gram matrices, kernel submatrices, and finite-rank approximation rows over displayed sample windows.*
4. *R is the* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{RKHS}^{\uparrow}$ *readback row, exposing only inner-product evaluations and reproducing-kernel coordinates already named by K, O, F.*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *eigenvalue-window row for the finite matrix and compact-operator approximants.*
6. *E is the expansion-read row, recording only finite Mercer-sum windows and their error budgets.*
7. *H records componentwise* hsame *transport for the kernel source, compact-operator shadow, matrix windows, Hilbert and RKHS readback, Real eigenvalue windows, expansion rows, and structural rows.*
8. *C records displayed* Cont *replay from kernel samples to matrix windows, from matrix windows to compact-operator reads, from compact-operator reads to Hilbert/RKHS coordinates, and from those coordinates to finite expansion windows.*

9. *P is the* Pkg *provenance over* $\mathsf{MercerTheorem}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, F, R, L, E, H, C, P, N$. It has no coordinate for ambient measure spaces, choice-selected eigenbases, unrestricted integral operators, host spectral decompositions, quotient function spaces, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MercerTheoremUp.MercerTheoremTasteGate single carrier alignment decode encode`

Theorem 2997.2 (Mercer theorem NameCert obligations)**.** *For every accepted* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier $M = (K, O, F, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$ *surface consists exactly of compact positive-kernel admission through K, compact-operator shadowing through O, finite matrix and kernel windows through F, Hilbert and RKHS readback through R, Real eigenvalue windows through L, finite expansion windows through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3974.1. Its analytic rows are exactly the positive-kernel source, compact-operator shadow, finite matrix windows, Hilbert/RKHS readback, Real eigenvalue windows, and finite expansion windows.

The route is finite. Kernel samples enter through K, become matrix and Gram windows through F, and are read as a compact positive operator only through O before a Hilbert or RKHS consumer may inspect coordinates in R.

The eigenvalue row L and expansion row E are downstream of those displayed windows. They therefore expose only finite Real eigenvalue approximants and finite Mercer-sum windows, not a selected basis or host spectral decomposition.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no row for ambient measure theory, unrestricted integral operators, choice-selected eigenbases, quotient function spaces, full spectral theorem data, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 2997.3 (Mercer theorem positive-kernel handoff)**.** *Every finite Mercer read from an accepted* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactOperator}^{\uparrow} \longmapsto \mathsf{Hilbert}^{\uparrow}, \mathsf{InnerProduct}^{\uparrow} \longmapsto \mathsf{RKHS}^{\uparrow}$$

together with H, C, P, N. The positive-kernel source is exported only as finite Gram-matrix positivity, compact-operator shadowing, Hilbert/RKHS coordinate readback, Real eigenvalue-window data, and finite expansion-window budgets.

Proof. Begin with the positive-kernel source row K. A consumer may inspect it only by opening the finite matrix windows in F, where positivity is visible as displayed Gram-window data over Real entries.

The compact-operator row O is then the only route from those finite windows to operator-facing reads. Thus an RKHS or Hilbert consumer cannot bypass the compact shadow with an unrestricted integral operator.

Read R, L, E next. The Hilbert and RKHS coordinates are exposed through R, the eigenvalue approximants through L, and the finite Mercer-sum windows through E.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed route. Because the carrier has no field for measure-theoretic integration, choice-selected

eigenbases, host spectral theorem data, quotient functions, or ambient Real completeness, the positive-kernel handoff remains finite. □

Closure status for $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MercerTheorem}^{\uparrow}$ packet is generated from a compact positive-kernel source row, a CompactOperator shadow row, finite Matrix kernel windows, Hilbert and RKHS readback rows, Real eigenvalue windows, finite Mercer expansion windows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3974.1, Definition 3974.2, and Definition 3974.3 seed the finite positive-kernel Mercer route over compact operators, Hilbert and inner-product rows, matrix windows, and Real eigenvalue readbacks.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient measure theory, choice-selected eigenbases, unrestricted integral operators, the full spectral theorem, quotient function-space equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-kernel matrix-window coverage, CompactOperator shadow compatibility, Hilbert and RKHS readback exactness, Real eigenvalue-window stability, expansion-window nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MercerTheoremUp.

Chapter 2998

A Concrete Naming Certificate for CauchyCompletionRegularSequence$^{\uparrow}$

Following Bishop and Bridges constructive Cauchy completion, the CauchyCompletionRegularSequence$^{\uparrow}$ packet records the handoff from regular Cauchy rational names to completion-facing and Real-facing consumers as finite BEDC data. It sits after the regular rational name surface of chapter 301, the equality discipline of chapter 1820, the metric completion carrier of chapter 1506, and the uniform completion boundary of chapter 1473, and before the terminal real-name seal of chapter 38. The packet names the completion handoff without forming quotient classes of regular sequences, selecting limits, invoking countable choice, importing ambient completeness, or using Real$^{\uparrow}$ completeness as an input.

Definition 2998.1 (CauchyCompletionRegularSequence carrier)**.** *A* CauchyCompletionRegularSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, E, M, U, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* RegSeqRat$^{\uparrow}$ *regular Cauchy rational name row.*
2. *E is the* CauchyEquivalenceSetoid$^{\uparrow}$ *comparison row, recording only the displayed equality discipline available between regular Cauchy names.*
3. *M is the* MetricCompletion$^{\uparrow}$ *handoff row reached from the same regular name and comparison data.*
4. *U is the* UniformCompletion$^{\uparrow}$ *handoff row for consumers that read the completion through uniform-tail data.*
5. *S is the terminal* Real$^{\uparrow}$ *seal row, available only after R, E, M, U have been displayed.*
6. *H records componentwise* hsame *transport for the regular name, comparison, completion handoff, uniform handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-facing request through R, E, M, U and then to the seal row S.*
8. *P is the* Pkg *provenance over* CauchyCompletionRegularSequence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyEquivalenceSetoid$^{\uparrow}$, MetricCompletion$^{\uparrow}$, UniformCompletion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, E, M, U, S, H, C, P, N$. It has no coordinate for quotient regular sequences, chosen representatives, selected limits, countable choice, ambient metric or uniform completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 2998.2 (CauchyCompletionRegularSequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ *carrier $Q = (R, E, M, U, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletio}}$ *surface consists exactly of the finite obligations of carrier admission for the displayed rows; regular-name exposure through R; comparison exposure through E; metric-completion handoff through M; uniform-completion handoff through U; Real seal nonescape through S; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation row is discharged by quotient equality, a selected limit, countable-choice data, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3975.1. The accepted tuple displays the regular name, comparison, metric-completion handoff, uniform-completion handoff, Real seal, transport, replay, provenance, and local name.

The regular-sequence side is exhausted by R and E. The row R supplies the regular rational name, and E supplies the displayed Cauchy-equivalence setoid comparison before any completion handoff is visible.

The completion-facing side is exhausted by M and U. A metric-completion consumer reads M, and a uniform-completion consumer reads U, but both rows are downstream from the same regular name and comparison rows.

Finally S, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no quotient, chosen-representative, selected-limit, choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionRegularSequenceUp.CauchyCompletionRegularSequenceTasteGate`
`single carrier alignment`

Theorem 2998.3 (CauchyCompletionRegularSequence Real seal handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the terminal seal is read only after the regular Cauchy name, displayed equivalence data, and completion handoff rows have been exposed, and it exports no quotient class, selected limit, countable choice, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an accepted consumer read of Q. Carrier projection opens R and E before any completion-facing row can be consumed.

If the consumer is metric-facing, it next reads M; if it is uniform-facing, it next reads U. In both cases the route still passes through the same regular name and comparison rows, so neither handoff introduces a quotient class or a selected representative.

The seal row S is downstream from those handoff rows. Componentwise transport and displayed replay through H and C preserve the route, while P and N attach provenance and local naming. The excluded objects are absent from the carrier, so no consumer can recover them from the handoff. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ completion-facing read factors through RegSeqRat, CauchyEquivalenceSetoid, MetricCompletion, UniformCompletion, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: regseqrat, cauchy-equivalence-setoid, metriccompletion, uniformcompletion, real: $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the regular name, displayed equivalence comparison, two completion handoff rows, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ packet is generated from a RegSeqRat regular Cauchy name row, a CauchyEquivalenceSetoid comparison row, MetricCompletion and UniformCompletion handoff rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3975.1, Definition 3975.2, and Definition 3975.3 seed the finite regular-sequence completion handoff from RegSeqRat and CauchyEquivalenceSetoid data through MetricCompletion and UniformCompletion rows to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient regular sequences, selected limits, countable choice, ambient metric or uniform completeness, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-name exposure, equivalence comparison exposure, metric-completion and uniform-completion handoff exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionRegularSequenceUp.

Chapter 2999

A Concrete Naming Certificate for $\mathsf{CosetSpace}^{\uparrow}$

Following Bourbaki's treatment of groups and cosets, the $\mathsf{CosetSpace}^{\uparrow}$ packet records the finite BEDC surface at which left and right cosets become classifier data over a displayed group and subgroup. It sits after the group carrier of chapter 41, the subgroup membership surface of chapter 80, and the quotient-facing route of chapter 82; it is also compatible with the finite permutation surface of chapter 117 and the word-source discipline of chapter 345. The packet names coset membership and representative replacement as finite rows, not as a host quotient type or an appeal to $\mathsf{Quot.sound}$.

Definition 2999.1 (Coset-space carrier)**.** *A* $\mathsf{CosetSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (G, S, r, o, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *operation surface, including carrier, multiplication, unit, inverse, and operation-law rows;*
2. *S is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *membership surface inside the group row G, including closure under multiplication, unit, inverse, and classifier transport;*
3. *r is the representative row naming the selected displayed group element used to read a coset;*
4. *o is the orientation row, distinguishing left-coset reads from right-coset reads without changing the ambient group or subgroup rows;*
5. *H is componentwise* hsame *transport for the group, subgroup, representative, orientation, replay, provenance, and local-name rows;*
6. *C is the displayed* Cont *replay route from a coset-space request through G, S, r, o, H, P, N;*
7. *P is the* Pkg *provenance over* $\mathsf{CosetSpace}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{SymGroup}^{\uparrow}$, $\mathsf{FreeMonoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers by the displayed group operation rows, subgroup membership rows, representative row, orientation row, transport, replay, provenance, and local naming. It has no coordinate for a host quotient set, quotient-type equality, selected transversal, external action orbit, or subgroup lattice object.

Lean4 ✓ `BEDC.Derived.CosetSpaceUp.CosetSpaceTasteGate single carrier alignment`

Theorem 2999.2 (Coset-space NameCert obligations)**.** *For the carrier of Definition 3976.1, the local* $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for* $X = (G, S, r, o, H, C, P, N)$*, group-operation exposure through* G*, subgroup-membership exposure through* S*, oriented representative readback through* r, o*, componentwise classifier transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. A consumer receives no host quotient type,* Quot.sound *equality, or off-packet representative selector.*

Proof. Project the carrier. The group row G provides the only multiplication, unit, inverse, and operation-law data visible to the packet, and the subgroup row S provides the only membership and closure data.

The representative row r and orientation row o determine whether the consumer reads a left or right coset. This read is still a finite membership route through G and S, so no host quotient carrier or selected transversal is introduced.

Finally, H, C, P, N preserve the same row inventory under transport, replay, provenance, and local naming. Since the carrier contains no coordinate for quotient-type equality or Quot.sound, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 2999.3 (Coset-space classifier transport)**.** *Let* $X = (G, S, r, o, H, C, P, N)$ *and* $X' = (G', S', r', o, H', C', P', N')$ *be accepted* $\mathsf{CosetSpace}^{\uparrow}$ *carriers with the same orientation row. If componentwise* hsame *transport identifies the group operation rows, subgroup membership rows, and representative rows, and if representative replacement is witnessed by the displayed subgroup-membership transport in* S *and* S'*, then the two carriers classify together as the same coset-space read. The transported classifier does not pass through a host quotient type or* Quot.sound*.*

Proof. Project both accepted carriers. The group rows G, G' supply the only multiplication and inverse operations, while the subgroup rows S, S' supply the only membership tests available for comparing representative replacements.

Use the displayed representative transport between r and r'. For a left-coset orientation, the replacement is read by multiplying the transported representative difference against the subgroup membership row; for a right-coset orientation, the same argument is read on the opposite side. In both cases the membership witness remains inside S and S'.

Replay the result through C and C' and compare provenance and local naming through P, N and P', N'. Because the route never leaves the displayed G, S, r, o rows, the classifier transport is a coset-membership transport rather than a quotient-type equality proof. □

Falsifiable prediction: An accepted $\mathsf{CosetSpace}^{\uparrow}$ consumer is refuted if it exports a coset classifier while omitting the displayed group row, subgroup membership row, representative row, orientation row, transport, replay, provenance, or local NameCert row.

Independence witness: group, subgroup, quotientgroup, symgroup, freemonoid: none is bijective to $\mathsf{CosetSpace}^{\uparrow}$ because this packet jointly carries the ambient group operation, subgroup membership surface, representative row, left/right orientation, transport, replay, provenance, and local naming. The group packet lacks subgroup membership and representatives; the subgroup packet lacks oriented coset classification; the quotient group packet names a quotient-facing route; the symmetric group and free monoid packets provide finite action and word sources without owning coset membership transport.

Closure status for $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$**.**

Constructive story: A CosetSpace$^{\uparrow}$ packet is generated from a group operation row, subgroup membership row, representative row, left/right orientation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CosetSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3976.1, Definition 3976.2, and Definition 3976.3 seed the coset-space carrier, obligation surface, and representative-transport classifier.

Formal verification: FormalStatus(NameCert$_{\mathsf{CosetSpace}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CosetSpaceUp.CosetSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-group normality, orbit-stabilizer theory, selected transversals, or host quotient-type equality for CosetSpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, group and subgroup row exposure, oriented representative transport, replay stability, provenance non-escape, and TasteGate instances for BEDC.Derived.CosetSpaceUp.

Chapter 3000

A Concrete Naming Certificate for $\mathsf{UniformCauchyNet}^{\uparrow}$

Following Moore–Smith and Kelley uniform-space nets, and Bishop-style finite completion discipline, the $\mathsf{UniformCauchyNet}^{\uparrow}$ packet records a uniform Cauchy net as displayed NameCert data rather than as a quotient net or selected limit. It sits beside the Cauchy-net source of chapter 589, the net-completion route of chapter 3794, the metric completion surface of chapter 1506, the uniform completion handoff of chapter 1473, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a constructive L10 completion neighbour whose consumers can inspect directed uniform-tail data without importing countable choice or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3000.1 (UniformCauchyNet carrier)**.** *A* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, E, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row, recording the finite Moore–Smith comparison data used to cofinalize the inspected tail.*
2. *E is the uniform tolerance or entourage ledger, listing the requested entourage rows and the finite tail witnesses that answer them.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened at the inspected indices and entourage requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and tolerance ledger.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, E, W, R have been displayed.*
6. *H records componentwise* hsame *transport for directed indices, entourage ledgers, stream windows, regular readback, Real sealing, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a uniform-tail request through the inspected windows, regular rational readback, and completion-facing handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, E, W, R, S, H, C, P, N$. It has no coordinate for quotient nets, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyNetUp.UniformCauchyNetTasteGate single carrier alignment`

Theorem 3000.2 (UniformCauchyNet NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier $U = (D, E, W, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the directed-index row, uniform entourage ledger, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-tail stability, requiring cofinal tail comparisons to be read from D before any entourage claim in E is exposed;*
3. *uniform tolerance exactness, requiring every inspected entourage request to pass through the finite ledger E and the corresponding* $\mathsf{StreamName}^{\uparrow}$ *windows W;*
4. *completion-facing readback, requiring the regular rational row R and the* $\mathsf{Real}^{\uparrow}$ *seal S to be reached only through the displayed directed, entourage, and window route;*
5. *non-escape for quotient nets, selected limits, countable choice, open-cover compactness, hidden cofinal schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3977.1. The tuple displays the directed index data, entourage ledger, window family, regular readback, Real seal, transport, replay, provenance, and local name.

Directed-tail stability is confined to D. A consumer first reads the finite Moore–Smith comparison rows that justify the inspected tail, so the later entourage rows cannot be treated as an unindexed uniform assertion.

Uniform tolerance exactness is the role of E and W. Each entourage request is paired with the displayed finite windows that witness the inspected tail, and no hidden cofinal schedule is available outside those rows.

The completion-facing readback then passes through R and S. Since R is obtained only after the directed and window data are displayed, and S is the terminal seal for that route, the packet exports a finite uniform-net handoff rather than a selected completion point.

Non-escape follows by row exhaustion. Quotient nets, selected limits, countable choice, open-cover compactness, hidden cofinal schedules, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, so the local NameCert surface cannot export them. □

Theorem 3000.3 (UniformCauchyNet filter handoff)**.** *Every completion- or filter-facing read of an accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{UniformCompletion}^{\uparrow}$ *rows named by the provenance are handoff surfaces for this same displayed route. Thus a consumer receives directed uniform-tail data, finite windows, regular readback, componentwise transport, displayed replay, provenance, and local naming, not a quotient net, chosen limit, countable-choice schedule, open-cover compactness principle, or ambient Real completeness theorem.*

Proof. Project the accepted carrier $U = (D, E, W, R, S, H, C, P, N)$. The directed row D and entourage ledger E expose the uniform Cauchy source, while W records the finite windows used to inspect it.

Replay through C preserves the order of the route. A filter-facing or completion-facing request first opens the directed tail and entourage ledger, then reads the stream windows, then obtains the regular rational row R, and only then reaches the Real seal S.

The completion siblings are read as handoff surfaces named in P. They consume the same displayed route and cannot add a quotient net, selected limit, countable-choice schedule, open-cover compactness principle, hidden cofinal tail, or ambient completeness coordinate. □

Theorem 3000.4 (UniformCauchyNet directed-tail stability)**.** *Let $U = (D, E, W, R, S, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier. If a consumer replaces the inspected directed tail by a cofinal refinement whose comparison rows are displayed in D, then the same local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *packet supplies the entourage ledger E, finite window family W, regular readback R, Real seal S, filter handoff rows, transport, replay, provenance, and local naming. The refinement changes only the displayed tail comparison inside D; it does not create a quotient net, selected limit, hidden cofinal schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from the carrier definition. The row D is the sole source of directed-index comparison data, and every later entourage or completion read is downstream of that row.

A cofinal refinement admitted by D therefore preserves the inspected tail route. The entourage ledger E and window family W are read after the directed comparison, so their witness rows remain attached to the same local packet.

Now apply the filter handoff theorem. The rows R and S are still reached through the ordered route from directed tail data to finite windows and regular readback.

The structural rows H, C, P, N transport and replay the same packet after the refinement. Because the carrier has no coordinate for quotient nets, selected limits, hidden cofinal schedules, or ambient completeness, the cofinal read has no escape branch. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyNet}^{\uparrow}$ completion-facing read factors through a directed index row, uniform entourage ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchynet, cauchy-net-completion, metriccompletion, uniformcompletion, streamname, regseqrat, real: $\mathsf{UniformCauchyNet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries directed index data, uniform entourage ledgers, finite stream windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyNet}^{\uparrow}$ packet is generated from a directed index row, a uniform tolerance or entourage ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3977.1, Definition 3977.2, and Definition 3977.3 seed the constructive uniform-net completion route over directed tails, entourage ledgers, finite windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, countable choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail stability, uniform tolerance exactness, completion-facing readback, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyNetUp.

Chapter 3001

A Concrete Naming Certificate for $\mathsf{ContourResidueHandoff}^{\uparrow}$

Following Cauchy's residue calculus, the $\mathsf{ContourResidueHandoff}^{\uparrow}$ packet records the finite consumer boundary at which a contour-integral operation may be read as residue-facing data. It sits after the contour operation surface of chapter 521, the operation socket of chapter 583, the holomorphic input boundary of chapter 68, and the residue packet of chapter 125. The packet exposes piecewise-linear contour rows, holomorphic rows, operation-law ledgers, residue rows, and non-escape rows as one finite handoff; it does not introduce a host contour homology theorem, analytic-continuation oracle, or global residue theorem.

Definition 3001.1 (Contour-residue handoff carrier)**.** *A* $\mathsf{ContourResidueHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, F, O, S, Z, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the piecewise-linear contour row read by the contour operation surface;*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *source row on the displayed contour domain away from the residue locus;*
3. *O is the* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row, including the Riemann-sum ledger, output row, and operation-law boundary;*
4. *S is the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row through which contour-operation consumers read G, F, O;*
5. *Z is the* $\mathsf{Residue}^{\uparrow}$ *row naming the local residue datum consumed by the handoff;*
6. *B is the boundary row recording that residue-facing reads require the displayed contour, holomorphic, socket, operation, and residue rows, and that no off-packet analytic theorem is exported;*
7. *H is componentwise* hsame *transport for G, F, O, S, Z, B together with replay, provenance, and local naming;*
8. *C is the displayed* Cont *replay route from a residue handoff request through $G, F, O, S, Z, B, H, P, N$;*
9. *P is* Pkg *provenance over* $\mathsf{ContourResidueHandoff}^{\uparrow}$, $\mathsf{ContourIntegralOperation}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Residue}^{\uparrow}$, *piecewise-linear contour rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$ *row.*

The classifier accepts only packets with these displayed rows. It has no coordinate for contour homology, global residue summation, meromorphic continuation, quotient path equality, or an analytic oracle outside the contour-operation packet.

Lean4 ✓ `BEDC.Derived.ContourResidueHandoffUp.ContourResidueHandoffTasteGate single carrier alignment`

Theorem 3001.2 (Contour-residue handoff NameCert obligations)**.** *For the carrier of Definition 3978.1, the local* $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission, PL-contour exposure, holomorphic source exposure, contour-operation exposure, contour-socket consumer routing, residue-row exposure, boundary non-escape, componentwise transport, displayed replay, provenance, and local naming. A residue-facing consumer receives no contour homology theorem, global residue theorem, quotient path equality, or analytic-continuation oracle.*

Proof. Project the carrier. The contour row G, holomorphic row F, operation row O, socket row S, and residue row Z are the only analytic data visible to the local certificate.

The boundary row B records which residue-facing reads may be consumed from those displayed rows. A request for a global residue theorem, contour homology invariant, quotient path equality, or analytic-continuation oracle has no coordinate in the packet and is therefore refused.

The transport, replay, provenance, and local naming rows H, C, P, N preserve this same finite inventory. Hence the NameCert obligations are exhausted by the displayed handoff rows. □

Theorem 3001.3 (Contour-residue handoff boundary)**.** *Let* $R = (G, F, O, S, Z, B, H, C, P, N)$ *be an accepted* $\mathsf{ContourResidueHandoff}^{\uparrow}$ *carrier. Every residue-facing consumer of the contour operation must first factor through the displayed* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row* O*, the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row* S*, the* $\mathsf{Holomorphic}^{\uparrow}$ *source row* F*, the PL-contour row* G*, and the* $\mathsf{Residue}^{\uparrow}$ *row* Z*. The handoff exports only the displayed residue boundary and does not create a global residue theorem, contour deformation theorem, or off-packet meromorphic continuation principle.*

Proof. Read the contour operation through O and the socket through S. These rows expose the piecewise-linear contour, holomorphic input, Riemann-sum and operation-law ledgers, output row, transport, and replay routes before any residue-facing request is considered.

The residue row Z is then consumed through the same route, with B recording the precise boundary between displayed residue data and unavailable global analytic consequences. Thus a consumer can read a local handoff but cannot infer contour homology, deformation invariance, or a global residue sum.

Finally, componentwise hsame transport and Cont replay keep G, F, O, S, Z, B attached to the same provenance and local NameCert rows. The absence of any additional analytic coordinate prevents the handoff from exporting off-packet meromorphic continuation or quotient path equality. □

Falsifiable prediction: An accepted $\mathsf{ContourResidueHandoff}^{\uparrow}$ consumer is refuted if it exports residue-facing contour data while omitting the displayed PL-contour row, holomorphic row, contour-operation row, contour-socket row, residue row, boundary row, transport, replay, provenance, or local NameCert row.

Independence witness: contourintegraloperation, contourintegralsocket, holomorphic, residue: none is bijective to $\mathsf{ContourResidueHandoff}^{\uparrow}$ because this packet jointly carries the contour row, holomorphic source, contour-operation ledger, socket consumer route, residue row, non-escape boundary, transport, replay, provenance, and local naming. The operation and socket packets lack

the residue handoff row; the holomorphic packet lacks contour output and residue data; the residue packet lacks contour-operation routing.

Closure status for $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{ContourResidueHandoff}^{\uparrow}$ packet is generated from a piecewise-linear contour row, holomorphic source row, contour-integral operation row, contour-integral socket row, residue row, boundary non-escape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3978.1, Definition 3978.2, and Definition 3978.3 seed the contour-residue handoff carrier, obligation surface, and residue-facing consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, contour homology, deformation invariance, the global residue theorem, meromorphic continuation, or quotient path equality for $\mathsf{ContourResidueHandoff}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, contour-operation exposure, socket routing, residue-row handoff, boundary non-escape, replay stability, provenance exactness, and TasteGate instances for BEDC.Derived.ContourResidueHandoffUp.

Chapter 3002

A Concrete Naming Certificate for $\mathsf{InitialValueProblem}^{\uparrow}$

Following Picard and Lindelof, the $\mathsf{InitialValueProblem}^{\uparrow}$ packet records the local ODE initial-data object consumed by Picard–Lindelof as explicit finite NameCert data. It sits between the ODE source of chapter 193, the Lipschitz vector-field ledger of chapter 410, the local existence route of chapter 1888, the complete-metric handoff of chapter 128, and the real-number ground of chapter 38. The packet names the initial time, state, local window, vector-field graph, solution graph, and endpoint classifier before any solution representative is exposed.

Definition 3002.1 (Initial-value problem carrier)**.** *An* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (T, X, F, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the time row, including the initial time cell and the local time-radius ledger;*
2. *X is the state row, including the initial state cell and the state-window ledger;*
3. *F is the vector-field graph row, read through the* $\mathsf{Ode}^{\uparrow}$ *and* $\mathsf{LipschitzMap}^{\uparrow}$ *dependency surface on the same local window;*
4. *W is the local window row binding the time and state ledgers used by the Picard route;*
5. *S is the solution-graph row exposed only as finite graph windows and not as a selected host function;*
6. *E is the endpoint classifier row, comparing endpoint reads inside the same local window;*
7. *H records componentwise* hsame *transport for the time row, state row, vector-field graph, local window, solution graph, endpoint classifier, replay rows, provenance, and local naming row;*
8. *C records the displayed* Cont *replay from initial data through vector-field reads, Picard iteration, completion handoff, solution windows, and endpoint classification;*
9. *P is the* Pkg *provenance over* $\mathsf{InitialValueProblem}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for a global flow, maximal continuation, host function-space equality, selected solution representative, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.InitialValueProblemUp.InitialValueProblemTasteGate single carrier alignment`

Theorem 3002.2 (Initial-value problem NameCert obligations)**.** *For an accepted* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier* $I = (T, X, F, W, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, time-state compatibility for* T *and* X*, vector-field graph admission through* F*, local-window nonescape through* W*, solution-graph visibility through* S*, endpoint classification through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 3979.1. The accepted packet displays the time, state, vector-field, local-window, solution-graph, endpoint, transport, replay, provenance, and local naming rows.

The source rows T, X, F, W bind one local initial-data surface. The time and state cells determine the initial condition, the vector-field graph is read only through the displayed ODE and Lipschitz rows, and the local window records the bounds within which later Picard reads are permitted.

The solution and endpoint rows are downstream of that surface. The row S records only displayed graph windows, while E classifies endpoint reads inside the same local window. Transport, continuation replay, provenance, and local naming preserve this route and do not add global ODE existence, maximal continuation, host function equality, or selected solution representatives. □

Theorem 3002.3 (Initial-value problem Picard handoff)**.** *Every accepted* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier hands a Picard–Lindelof consumer exactly the displayed route*

$\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The consumer must read the time row, state row, vector-field graph, and local window before it may inspect solution-graph windows or endpoint classifiers. The handoff exports no global solution curve, maximal interval, host function-space equality, selected solution representative, or completeness principle outside the displayed complete-metric row.

Proof. Start with the initial-data surface. The rows T and X give the initial time and state, while F and W bind the vector-field graph to the same local window.

Replay the Picard-facing route through C. The Picard–Lindelof consumer reaches the solution-graph row S only after the ODE and Lipschitz data have been read, and it reaches the endpoint classifier E only through the displayed completion handoff.

Finally use the structural rows. Componentwise hsame transport in H, provenance in P, and the local name N record the same finite handoff. Since the carrier contains no global-flow, maximal-continuation, host-equality, selected-representative, or detached-completeness coordinate, none can be exported to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$.

Constructive story: An $\mathsf{InitialValueProblem}^{\uparrow}$ packet is generated from an initial time row, an initial state row, a vector-field graph row, a local window, solution-graph windows, an endpoint classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3979.1, Definition 3979.2, and Definition 3979.3 seed the local initial-data surface consumed by Picard–Lindelof over ODE, Lipschitz, complete-metric, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{InitialValueProblem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, maximal continuation, host function-space equality, selected solution representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, time-state compatibility, vector-field graph admission, local-window nonescape, solution-graph visibility, endpoint classification, transport, replay, provenance, and local naming for BEDC.Derived.InitialValueProblemUp.

Chapter 3003

A Concrete Naming Certificate for $\mathsf{UniformCauchyLimit}^{\uparrow}$

Following Bishop and Bridges on uniform convergence and constructive continuity, the $\mathsf{UniformCauchyLimit}^{\uparrow}$ packet records a uniform limit of modulus-controlled real-valued continuous maps as finite NameCert data. It sits beside the continuous-map graph rows of chapter 130, the function-specific modulus rows of chapter 1631, the convergence ledger of chapter 2074, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a completion-facing uniform-limit carrier without importing quotient functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3003.1 (Uniform Cauchy limit carrier)**.** *A* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, L, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows for the inspected real-valued maps.*
2. *M is the matching family of* $\mathsf{ModulusContinuity}^{\uparrow}$ *rows, one for each graph row in F, converting target tolerances into source-window budgets.*
3. *L is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger that records the uniform Cauchy comparison between later graph rows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened at the source points and tolerance requests used by F, M, L.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and convergence ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying dyadic radii, endpoint comparisons, and uniform error budgets.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, M, L, S, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for graph rows, modulus rows, convergence ledgers, stream windows, regular readback, dyadic tolerances, Real sealing, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a uniform tolerance request through graph inspection, modulus conversion, convergence comparison, stream-window readback, dyadic tolerance accounting, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{RealModulusOfConver}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, M, L, S, Q, D, R, H, C, P, N$. It has no coordinate for quotient functions, selected pointwise limits, countable-choice schedules, open-cover compactness, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyLimitUp.UniformCauchyLimitTasteGate single carrier alignment`

Theorem 3003.2 (Uniform Cauchy limit NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *carrier $U = (F, M, L, S, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the continuous-map graph rows, modulus-continuity rows, convergence ledger, stream windows, regular rational readback, dyadic tolerance row, Real seal, transport, replay, provenance, and local naming rows;*
2. *graph-modulus compatibility, requiring every inspected map in F to expose its matching modulus row in M before a target tolerance is read;*
3. *uniform Cauchy exactness, requiring each later-stage comparison to pass through the displayed ledger L and the dyadic tolerance row D;*
4. *readback sealing, requiring S and Q to be read before the terminal* $\mathsf{Real}^{\uparrow}$ *seal R is exposed;*
5. *non-escape for quotient function spaces, selected limits, countable choice, open-cover compactness, host function equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3980.1. The accepted tuple displays exactly the graph rows, modulus rows, convergence ledger, windows, readback, dyadic tolerance ledger, Real seal, and structural rows.

Graph-modulus compatibility is rowwise. A consumer that reads a graph row in F must read the corresponding row in M, so continuity is exposed as a finite tolerance-to-window route rather than as a host semantic predicate.

Uniform Cauchy exactness is the role of L and D. Later-stage comparisons are admitted only through the displayed convergence ledger and dyadic tolerance budgets, so no hidden choice schedule or selected limit enters the packet.

Readback sealing follows through S, Q, R. The finite stream windows are opened first, the regular rational readback is extracted next, and the terminal Real seal is reached only after those rows are visible.

Finally, H, C, P, N transport, replay, source, and name the same displayed packet. Since the refused objects are absent from the carrier, the local NameCert surface exports exactly the five obligations listed above. □

Theorem 3003.3 (Uniform Cauchy limit continuity handoff)**.** *Every continuity-facing read of an accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{ModulusContinuity}^{\uparrow} \longmapsto \mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

Thus a downstream continuous-map or completion consumer receives graph rows, modulus rows, uniform convergence ledgers, finite windows, regular rational readback, dyadic tolerance accounting, componentwise transport, displayed replay, provenance, and local naming, not a quotient function or selected limit.

Proof. Project the accepted carrier $U = (F, M, L, S, Q, D, R, H, C, P, N)$. The graph rows in F are the only function data, and the modulus rows in M are the only continuity data attached to them.

Replay through C preserves the route. A continuity-facing request first opens the graph row, then reads the displayed modulus conversion, then consults the uniform Cauchy ledger and dyadic tolerances before it can inspect stream windows and regular rational readback.

The Real seal R is terminal. Since the carrier has no coordinate for quotient functions, selected limits, host equality, or ambient completeness, the handoff remains a finite BEDC route through the displayed rows. □

Theorem 3003.4 (Uniform Cauchy limit completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *packet must display the rows* $F, M, L, S, Q, D, R, H, C, P, N$ *before reading any terminal Real seal. The accepted packet therefore supplies a uniform-limit carrier for real-valued continuous maps without exporting quotient functions, selected limit representatives, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with carrier projection. The completion-facing data are not stored in a separate ambient object; they are the displayed rows $F, M, L, S, Q, D, R, H, C, P, N$.

The finite route composes graph inspection, modulus conversion, convergence-ledger comparison, stream-window opening, regular rational readback, dyadic tolerance accounting, and terminal Real sealing. Each step is a row of the accepted carrier and is preserved by the transport and replay rows.

No refused object can be exported after the seal. Quotient functions, selected representatives, choice schedules, open-cover compactness, and ambient completeness have no carrier coordinate, so completion-facing consumers remain inside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyLimit}^{\uparrow}$ packet is generated from ContinuousMap graph rows, ModulusContinuity rows, a RealModulusOfConvergence ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3980.1, Definition 3980.2, Definition 3980.3, and Definition 3980.4 seed the uniform-limit route over continuous-map graphs, modulus rows, convergence ledgers, finite windows, regular readback, dyadic tolerance data, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient function spaces, selected pointwise limits, countable choice, open-cover compactness, host function equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph-modulus compatibility, uniform Cauchy exactness, readback sealing, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyLimitUp.

Chapter 3004

A Concrete Naming Certificate for $\mathsf{MedianDepthSealing}^{\uparrow}$

The $\mathsf{MedianDepthSealing}^{\uparrow}$ packet realizes the two-layer median stabilization of Definition 9462.1329 as finite BEDC NameCert data. It sits beside the six-bit codon window of chapter 2234, the boundary-tile classifier of chapter 3393, the six-tile coverage rows of chapter 3470, the boundary-trigger closure route of chapter 3508, and the finite alphabet bridge of chapter 2260. The packet names the reassigned support, first median layer, second sealing point, layer witnesses, and stabilization check as a finite carrier, not as a biological stochastic model or an ambient closure oracle.

Definition 3004.1 (Median depth sealing carrier)**.** *A* $\mathsf{MedianDepthSealing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, V, T, A, L, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source row for the codon-window coordinates used by the median operation.*
2. *B is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *boundary row exposing the stop/Trp and Ile/Met tile classifiers consumed by the reassigned support.*
3. *V is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *coverage row recording the visible boundary support over which the median read is admitted.*
4. *T is the* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *trigger-closure row, read only as finite boundary-trigger data.*
5. *A is the* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ *alphabet row, fixing the four-phase coordinate interpretation before codons are compared.*
6. *L is the median-layer ledger with the displayed rows $\mathcal{R}$, Δ_1, Δ_2, $\mathrm{Med}_1(\mathcal{R})$, and $\mathrm{Med}_2(\mathcal{R})$ from Definition 9462.1328.*
7. *E is the stabilization row recording that the second layer is closed under the displayed coordinatewise median operation.*
8. *H records componentwise* hsame *transport for codon windows, boundary tiles, coverage rows, trigger closure, alphabet rows, layer ledgers, stabilization, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from codon-window inspection through boundary-tile coverage, trigger closure, alphabet readback, layer enumeration, and stabilization checking.*
10. *P is the* Pkg *provenance over* $\mathsf{MedianDepthSealing}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$, *the median-depth source theorem,* BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, V, T, A, L, E, H, C, P, N$. It has no coordinate for a biological code table, stochastic evolutionary dynamics, quotient codon semantics, hidden median closure, unbounded search, or host closure oracle.

Lean4 ✓ `BEDC.Derived.MedianDepthSealingUp.MedianDepthSealingTasteGate single carrier alignment`

Theorem 3004.2 (Median depth sealing NameCert obligations)**.** *For every accepted* $\mathsf{MedianDepthSealing}^{\uparrow}$ *carrier $M = (W, B, V, T, A, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$ *surface consists exactly of codon-window admission, boundary-tile exposure, six-tile coverage, boundary-trigger closure, four-phase alphabet readback, median-layer ledger exactness, two-layer stabilization, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3981.1. The codon-window, boundary, coverage, trigger, alphabet, layer, stabilization, and structural rows are all displayed as finite data.

The source rows W, B, V, T, A fix the finite coordinate surface before the median ledger is read. Thus the median operation is interpreted through displayed codon windows, boundary tiles, coverage rows, trigger closure, and four-phase alphabet readback.

The rows L and E then expose the two-layer calculation. The first row records $\mathcal{R}$, Δ_1, Δ_2, $\mathrm{Med}_1(\mathcal{R})$, and $\mathrm{Med}_2(\mathcal{R})$; the second records that no further vertex appears after the sealing point.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no remaining coordinate contains a biological table, stochastic model, quotient codon semantics, unbounded search, or hidden closure oracle, the listed rows exhaust the local NameCert surface. □

Theorem 3004.3 (Median depth sealing two-layer stabilization)**.** *Every accepted* $\mathsf{MedianDepthSealing}^{\uparrow}$ *read exposes the finite stabilization chain*

$$\mathcal{R} \longmapsto \mathrm{Med}_1(\mathcal{R}) \longmapsto \mathrm{Med}_2(\mathcal{R}) = \mathrm{Med}(\mathcal{R})$$

with $\Delta_1 = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$ and $\Delta_2 = \{\mathrm{ACG}\}$. A consumer must read the layer ledger L and stabilization row E after the codon-window, boundary-tile, coverage, trigger-closure, and alphabet rows have been displayed.

Proof. Read Definition 9462.1329 through the carrier rows. The theorem supplies the finite layer calculation, while Definition 3981.1 stores it as the ledger L and stabilization row E.

The upstream rows W, B, V, T, A determine the coordinate interpretation and boundary support before the ledger is inspected. Therefore a consumer cannot treat the displayed layer calculation as an ambient median closure unrelated to the accepted codon-window and boundary rows.

The structural rows H, C, P, N preserve the same route. The exported consequence is the two-layer stabilization chain and no additional biological, stochastic, quotient, or hidden closure datum. □

Falsifiable prediction: Every accepted $\mathsf{MedianDepthSealing}^{\uparrow}$ consumer must expose the two-layer stabilization consequence: the first median layer adds exactly Δ_1, the second layer adds exactly ACG, and a further coordinatewise-median pass adds no codon.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-four-phase-alphabet-bridge: none is bijective to $\mathsf{MedianDepthSealing}^{\uparrow}$ because the siblings respectively expose boundary classifiers, coverage rows, trigger closure, or alphabet readback, while the median-depth packet jointly carries the reassigned support, two median layers, sealing codon, stabilization check, transport, replay, provenance, and local naming.

The Lean TasteGate site for $\mathsf{MedianDepthSealing}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ MedianDepthSealing}^{\uparrow}$ and $\mathsf{FieldFaithful\ MedianDepthSealing}^{\uparrow}$ instances. Nontriviality must distinguish the reassigned support from the sealed second layer, and field faithfulness must preserve $W, B, V, T, A, L, E, H, C, P, N$ as separate displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$.

Constructive story: A $\mathsf{MedianDepthSealing}^{\uparrow}$ packet is generated from CodonSixBitWindow rows, DnaWindowSixBoundaryTile rows, SixBoundaryTileCoverage rows, DnaBoundaryTriggerClosure rows, DNAFourPhaseAlphabetBridge rows, a median-layer ledger, a stabilization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3981.1, Definition 3981.2, and Definition 3981.3 seed the finite median-depth sealing packet over codon windows, boundary tiles, coverage rows, trigger closure, alphabet readback, layer enumeration, and stabilization checking.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a biological code table, stochastic evolutionary model, quotient codon semantics, hidden median closure, unbounded search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary and coverage exposure, trigger-closure compatibility, alphabet readback, median-layer exactness, stabilization checking, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MedianDepthSealingUp.

Chapter 3005

A Concrete Naming Certificate for $\mathsf{UniformFixedPoint}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{UniformFixedPoint}^{\uparrow}$ packet records the endpoint-modulus surface obtained when a displayed contraction, a finite Picard orbit, and a complete metric handoff are assembled as one L10 NameCert route. It sits after the complete metric carrier of chapter 128, the contractive map surface of chapter 3819, the Picard iteration packet of chapter 2359, the fixed-point iteration packet of chapter 2383, the located real modulus of chapter 3406, and the Banach contraction principle of chapter 3962. The packet exposes a uniform endpoint modulus for downstream fixed-point consumers; it does not introduce choice-selected endpoints, quotient equality, open-cover compactness, or ambient completeness outside the displayed rows.

Definition 3005.1 (Uniform fixed-point carrier)**.** *A* $\mathsf{UniformFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, F, I, Q, U_0, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, naming the metric carrier, convergence service, and endpoint limit surface available to the packet;*
2. *F is the* $\mathsf{ContractiveMap}^{\uparrow}$ *row, recording the self-map graph, contraction ledger, and metric compatibility used by the endpoint construction;*
3. *I is the* $\mathsf{PicardIteration}^{\uparrow}/\mathsf{FixedPointIteration}^{\uparrow}$ *row, exposing the initial point, finite iterate windows, residual budget, and Picard replay route;*
4. *Q is the Cauchy or located modulus row, read through* $\mathsf{LocatedRealModulus}^{\uparrow}$ *or the displayed Cauchy-modulus surface attached to the iteration;*
5. *U_0 is the uniform endpoint-modulus row, translating a requested endpoint tolerance into the finite Picard window and contraction budget required by M, F, I, Q;*
6. *E is the endpoint seal row, recording the* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *fixed-point endpoint only after the uniform modulus and complete-metric handoff have been read;*
7. *H records componentwise* hsame *transports for M, F, I, Q, U_0, E together with the replay, provenance, and local naming rows;*

8. *C records the displayed* Cont *replay route by which the Picard window, contraction ledger, located modulus, endpoint modulus, and endpoint seal are inspected;*
9. *P is the* Pkg *provenance over* UniformFixedPoint$^\uparrow$, CompleteMetric$^\uparrow$, ContractiveMap$^\uparrow$, PicardIteration$^\uparrow$, FixedPointIteration$^\uparrow$, LocatedRealModulus$^\uparrow$, BanachContractionPrinciple$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{UniformFixedPoint}^\uparrow}$ *row.*

The classifier accepts exactly packets with these ten rows. It has no coordinate for an endpoint chosen outside the Picard route, a quotient fixed-point equality, a hidden contraction constant, or a completeness witness detached from M.

Theorem 3005.2 (Uniform fixed-point NameCert obligations)**.** *For the carrier of Definition 3982.1, the local* NameCert$_{\text{UniformFixedPoint}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for $U = (M, F, I, Q, U_0, E, H, C, P, N)$;*
2. *contractive-map admission, requiring every endpoint read to expose the* ContractiveMap$^\uparrow$ *row F against the complete metric row M;*
3. *Picard and fixed-point iteration admission, requiring I to display the finite orbit rows before any Cauchy or endpoint row is inspected;*
4. *Cauchy or located-modulus handoff, requiring Q to route tolerance requests through* LocatedRealModulus$^\uparrow$ *or the displayed Cauchy-modulus surface;*
5. *uniform endpoint-modulus exactness, requiring U_0 to translate endpoint tolerances only through M, F, I, Q;*
6. *endpoint seal nonescape, requiring E, H, C, P, N to keep the* BanachContractionPrinciple$^\uparrow$ *endpoint inside the same packet.*

Proof. Project the carrier of Definition 3982.1. The rows $M, F, I, Q, U_0, E, H, C, P, N$ are the only local NameCert coordinates, so carrier admission is exactly the displayed packet.

Read the contraction and iteration rows next. The ContractiveMap$^\uparrow$ row supplies only the finite shrink ledger, while the PicardIteration$^\uparrow$ and FixedPointIteration$^\uparrow$ rows supply the finite orbit and residual budget before any endpoint is available.

Use the modulus rows Q and U_0. The located or Cauchy modulus answers tolerance requests by naming a finite Picard window, and the uniform endpoint modulus converts that window into the endpoint tolerance without adding a selected limit or quotient equality.

Finally read the endpoint seal through E, H, C, P, N. The BanachContractionPrinciple$^\uparrow$ endpoint is exposed only after the complete-metric, contraction, iteration, and modulus rows have been replayed, so no off-packet fixed point or ambient completeness coordinate is exported. □

Lean4 ✓ `BEDC.Derived.UniformFixedPointUp.UniformFixedPointTasteGate single carrier alignment`

Theorem 3005.3 (Uniform fixed-point modulus handoff)**.** *Every* BanachContractionPrinciple$^\uparrow$ *or fixed-point endpoint consumer of an accepted* UniformFixedPoint$^\uparrow$ *packet receives the endpoint only through the finite route*

CompleteMetric$^\uparrow$, ContractiveMap$^\uparrow$, PicardIteration$^\uparrow$, FixedPointIteration$^\uparrow$, LocatedRealModulus$^\uparrow$, U_0, BanachContra

The row U_0 is a uniform endpoint-modulus handoff: it turns each endpoint tolerance into a displayed Picard window and contraction budget before the endpoint seal E is read. The handoff exports

no arbitrary fixed-point choice, no quotient equality of endpoints, no open-cover compactness, and no ambient completeness beyond the complete metric row M.

Proof. Start from the NameCert obligations. They force every consumer to expose M, F, I, Q, U_0 before reading E.

Apply the sibling Picard and fixed-point iteration routes. The finite orbit row I supplies the iterate windows and residual budgets, while F supplies the contraction ledger over the metric carrier named by M.

Use the located modulus handoff through Q. A requested endpoint tolerance is converted into finite dyadic or Cauchy data and then into the Picard window named by U_0.

Finally pass to the Banach contraction endpoint seal. Since E is downstream of the complete-metric service, contraction ledger, Picard route, and uniform modulus, the endpoint consumer receives only the displayed finite route and none of the refused host-level principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformFixedPoint}^{\uparrow}$ packet is generated from a complete metric row, a contractive map row, a Picard or fixed-point iteration row, a Cauchy or located modulus row, a uniform endpoint-modulus row, an endpoint seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3982.1, Definition 3982.2, and Definition 3982.3 seed the endpoint-modulus route over $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContractiveMap}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{LocatedRealModulus}^{\uparrow}$, and $\mathsf{BanachContractionPrinciple}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, choice-selected fixed points, quotient endpoint equality, open-cover compactness, or completeness outside the displayed complete metric row.

Upgrade path: A formal route can encode the finite carrier, NameCert obligations, endpoint-modulus exactness, and endpoint-seal handoff for BEDC.Derived.UniformFixedPointUp.

Chapter 3006

A Concrete Naming Certificate for $\mathsf{CauchySchwarzEquality}^{\uparrow}$

Following Cauchy, Schwarz, and standard inner-product-space treatments of the equality case, the $\mathsf{CauchySchwarzEquality}^{\uparrow}$ packet records the finite BEDC surface at which equality in Cauchy–Schwarz is available as a linear-dependence certificate. It sits over the inner-product source of chapter 86, the norm boundary of chapter 89, and the finite Real inequality route of chapter 1684. The packet names equality-case data as displayed rows; it is not a host vector equality theorem, quotient stream equality, ambient Hilbert-space statement, or selected scalar witness.

Definition 3006.1 (CauchySchwarzEquality carrier)**.** *A* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N)$$

with the following displayed rows:

1. *V is the finite vector-source row shared with the inner-product and norm packets.*
2. *X and Y are the two displayed vector rows whose equality-case boundary is inspected.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *scalar read for $\langle X, Y\rangle$.*
4. *A and B are the two* $\mathsf{Norm}^{\uparrow}$ *square rows for $\|X\|^2$ and $\|Y\|^2$.*
5. *C is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *inequality row that exposes the square comparison already sealed for Real consumers.*
6. *L is the linear-dependence ledger. It records a displayed scalar row, the finite scalar-action comparison between X and Y, and the zero-vector boundary used when one norm row is null.*
7. *D is the equality-defect ledger, storing the displayed zero-defect read that identifies the equality boundary of the square comparison.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scalar, norm, and defect observations.*
9. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the readback.*
10. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the equality-case read.*
11. *H records componentwise* hsame *transport for vector rows, scalar reads, norm rows, inequality row, linear-dependence ledger, defect ledger, readback, windows, Real seal, replay, provenance, and local naming.*

12. *T records displayed* Cont *replay from an equality-case request through $V, X, Y, I, A, B, C, L, D, Q, S, R$ and back to the public certificate boundary.*

13. *P is the* Pkg *provenance over* $\mathsf{CauchySchwarzEquality}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite equality-case packet. It has no coordinate for host vector equality, quotient stream equality, ambient completeness, an infinite-dimensional Hilbert theorem, selected scalar choice, or bridge checking.

Theorem 3006.2 (CauchySchwarzEquality NameCert obligations)**.** *For every accepted* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the vector source, two vector rows, inner-product read, norm-square rows, Cauchy–Schwarz inequality row, linear-dependence ledger, equality-defect ledger, readback, windows, Real seal, transport, replay, provenance, and local naming rows;*

2. *equality-route exposure, requiring every equality-case read to pass through I, A, B, C, D before the Real seal R is available;*

3. *linear-dependence exposure, requiring the row L to display the scalar-action comparison and zero-vector boundary before the local name is exported;*

4. *classifier transport, requiring H and T to preserve the same vector rows, scalar reads, inequality row, dependence ledger, defect ledger, readback, windows, and Real seal componentwise;*

5. *non-escape, excluding host vector equality, quotient stream equality, ambient completeness, infinite-dimensional Hilbert semantics, selected scalar choice, and bridge checking.*

No equality-case obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are the vector-source row V, the displayed vector rows X, Y, the inner-product row I, the norm-square rows A, B, the Cauchy–Schwarz row C, the dependence ledger L, the equality-defect ledger D, the readback rows Q, S, R, and the structural rows H, T, P, N.

Carrier admission is exactly admission of those displayed coordinates. The local certificate has no further coordinate from which a host vector equality, quotient-stream representative, ambient Hilbert-space object, or selected scalar could be recovered.

Equality-route exposure is forced by the placement of C and D. The packet first reads the ordinary Cauchy–Schwarz square comparison, then reads the displayed zero-defect ledger, and only then seals the equality-case datum as Real-facing packet data.

Linear-dependence exposure is the row L. It records the scalar-action comparison and the zero-vector boundary as finite data, so the equality case is exported as a dependence ledger rather than as an ambient vector-space equality theorem.

Finally H and T provide componentwise transport and replay for the same rows, while P and N source and name the packet. Since no excluded analytic coordinate is present, the listed rows exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchySchwarzEqualityUp.CauchySchwarzEqualityTasteGate single carrier alignment`

Theorem 3006.3 (CauchySchwarzEquality linear-dependence route)**.** *For every accepted* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N),$$

every equality-case consumer receives the linear-dependence boundary only through the finite route

$$V, X, Y \longmapsto I, A, B \longmapsto C, D \longmapsto L \longmapsto Q, S, R \longmapsto H, T, P, N.$$

Thus equality in the Cauchy–Schwarz row is consumed as a displayed scalar-action ledger and zero-defect readback; it does not expose host vector equality, quotient stream equality, ambient completeness, an infinite-dimensional Hilbert theorem, selected scalar choice, or bridge checking.

Proof. Start with the carrier projection. The vector rows V, X, Y and scalar rows I, A, B are the only inputs from the inner-product and norm surfaces.

Apply the finite inequality route supplied by chapter 1684. The row C exposes the square comparison already sealed for Real consumers, while D records that the defect read is zero inside the displayed packet.

Next read L. This is the only row carrying the scalar-action comparison and the zero-vector boundary, so the consumer obtains linear dependence from finite ledger data rather than from host vector equality.

The readback route Q, S, R then seals the same displayed equality-case datum for Real-facing consumers. The transport and replay rows H, T preserve that order, and the provenance and naming rows P, N add no analytic coordinate.

Consequently the exported equality case is exactly the displayed route above. None of the excluded host or quotient objects can be read from the packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchySchwarzEquality}^{\uparrow}$ consumer must expose a CauchySchwarzRealUp inequality row, an equality-defect ledger, a linear-dependence ledger, RegSeqRat readback, StreamName windows, a RealUp seal, transport, replay, provenance, and local naming before the equality-case boundary is available.

Independence witness: innerproduct, norm, cauchy-schwarz-real: $\mathsf{CauchySchwarzEquality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cauchy–Schwarz inequality row, equality-defect ledger, displayed linear-dependence ledger, readback, seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySchwarzEquality}^{\uparrow}$ packet is generated from finite vector rows, an InnerProductUp scalar read, NormUp square rows, a CauchySchwarzRealUp inequality row, an equality-defect ledger, a linear-dependence ledger, RegSeqRat readback, StreamName windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$, seed)

Scope closed: Definition 3983.1, Definition 3983.2, and Definition 3983.3 seed the finite equality-case carrier for displayed Cauchy–Schwarz Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host vector equality, quotient stream equality, ambient completeness, infinite-dimensional Hilbert semantics, selected scalar choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, equality-route exposure, linear-dependence exposure, classifier transport, non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchySchwarzEqualityUp.

Chapter 3007

A Concrete Naming Certificate for $\mathsf{ExtremeValueTheorem}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{ExtremeValueTheorem}^{\uparrow}$ packet records the compact-real attained-extreme theorem as finite BEDC NameCert data. It sits over the compact source of chapter 129, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located supremum and upper-bound rows of chapter 1427, and the terminal real-name seal of chapter 38. The packet exposes the finite witness-window route for extrema without importing open-cover compactness, trichotomy, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3007.1 (Extreme value theorem carrier)**.** *An* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, K, F, R, S, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite windows, and real endpoint seals.*
2. *K is the compact source row read through* $\mathsf{CompactMetric}^{\uparrow}$*, exposing the finite compact-real inspection surface used by the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *value seal row for inspected values and endpoint comparisons.*
5. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row carrying the located upper-bound surface for the displayed value family.*
6. *W is the finite witness-window row, recording the inspected compact windows in which the located upper-bound row and the continuous-map row meet a terminal value seal.*
7. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, real value seals, located-supremum row, witness windows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from compact-real inspection requests through $I, K, F, R, S, W, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ExtremeValueTheorem}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed interval source, compact source, continuous-map row, value seal, located upper-bound row, finite witness-window row, transport, replay, provenance, and local naming. It has no coordinate for open-cover compactness, trichotomy, selected host extrema, quotient real equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.ExtremeValueTheoremUp.ExtremeValueTheoremTasteGate single carrier alignment`

Theorem 3007.2 (Extreme value theorem NameCert obligations)**.** *For every accepted* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *carrier* $E = (I, K, F, R, S, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact source, continuous-map row,* $\mathsf{Real}^{\uparrow}$ *value seal, located-supremum and upper-bound row, finite witness-window row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 3984.1. The interval row I and compact row K expose the only source windows available to the theorem packet, while F records the continuous real-valued map inspected on those windows.

Read the real and upper-bound rows next. The value seal R supplies only $\mathsf{Real}^{\uparrow}$ rows for displayed function values, and S supplies the located upper-bound surface through the located-supremum chapter.

The witness-window row W binds the source windows, continuous-map reads, value seals, and upper-bound reads into one finite inspection route. The remaining rows H, C, P, N transport, replay, source, and name the same route, so the listed data exhaust the local NameCert obligations. □

Theorem 3007.3 (Extreme value theorem compact supremum route)**.** *Every accepted* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *consumer that reads an attained extreme must factor through the displayed route*

$$I, K \longmapsto F \longmapsto R, S \longmapsto W \longmapsto H, C, P, N.$$

Thus the compact-real theorem surface exports only a finite witness-window route from a compact source and continuous-map row to a $\mathsf{Real}^{\uparrow}$ *value seal and located upper-bound row. It does not export open-cover compactness, trichotomy, a selected host maximum or minimum, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact metric source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real windows.

Feed the source into the continuous-map row. The row F exposes only displayed value reads, and R seals those reads as $\mathsf{Real}^{\uparrow}$ values before any upper-bound comparison is consumed.

Use the located-supremum row S and the witness-window row W together. The upper-bound route is visible only through the finite windows in which F and R have already been displayed.

Finally, transport and replay preserve the same route through H and C, while P and N provide provenance and local naming. Since the carrier has no coordinate for open-cover compactness, trichotomy, selected extrema, quotient real equality, or ambient completeness, the exported theorem surface remains the finite compact-supremum route shown above. □

Falsifiable prediction: Every $\mathsf{ExtremeValueTheorem}^{\uparrow}$ attained-extreme read factors through $\mathsf{ClosedBoundedInterva}$ $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{LocatedSupremum}^{\uparrow}$ before reaching its finite witness-window and local $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ rows.

Independence witness: closedboundedinterval, compactmetric, continuousmap, locatedsupremum, real: $\mathsf{ExtremeValueTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact source, continuous-map row, Real value seal, located upper-bound row, finite witness-window row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{ExtremeValueTheorem}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, RealUp value seals, a LocatedSupremumUp upper-bound row, a finite witness-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3984.1, Definition 3984.2, and Definition 3984.3 seed the finite compact-real attained-extreme route over interval, compact-source, continuous-map, located-supremum, witness-window, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, selected host extrema, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{ExtremeValueTheorem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, located upper-bound routing, Real value sealing, replay stability, and non-escape for BEDC.Derived.ExtremeValueTheoremUp.

Chapter 3008

A Concrete Naming Certificate for $\mathsf{BoundedRealOscillation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BoundedRealOscillation}^{\uparrow}$ packet records the finite route by which a continuous real-valued map on a compact metric source receives an oscillation bound from displayed uniform-modulus and finite-partition data. It sits between the compact source surface of chapter 129, the continuous-map carrier of chapter 130, the uniform-modulus route of chapter 369, the finite oscillation partition of chapter 3800, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names an oscillation-bound witness without importing open-cover compactness, supremum selection, quotient real equality, or choice.

Definition 3008.1 (BoundedRealOscillation carrier)**.** *A* $\mathsf{BoundedRealOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, U, A, B, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the inspected compact-metric windows and finite metric ledgers.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from K to* $\mathsf{Real}^{\uparrow}$*, with its graph, local continuity data, and real-value readbacks.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row translating each requested oscillation tolerance into a displayed metric radius.*
4. *A is the* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, listing the finite cells inspected at that tolerance and the local oscillation ledgers attached to them.*
5. *B is the bounded-oscillation ledger, recording the displayed real bound certified from the cellwise oscillation rows and the uniform modulus.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the bound value, reached only after K, F, U, A, B have been read.*
7. *H records componentwise* hsame *transport for the compact source, continuous map, uniform modulus, finite partition, bound ledger, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an oscillation request through K, F, U, A, B, R and back to the public certificate boundary.*

9. *P is the* Pkg *provenance over* BoundedRealOscillation$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, FiniteOscillationPartition$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BoundedRealOscillation}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten rows. It has no coordinate for open-cover compactness, a selected supremum, host function equality, quotient real equality, countable choice, or an ambient completeness principle detached from the displayed Real$^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.BoundedRealOscillationUp.BoundedRealOscillationTasteGate single carrier alignment`

Theorem 3008.2 (BoundedRealOscillation NameCert obligations)**.** *For every accepted* BoundedRealOscillation$^{\uparrow}$ *carrier*

$$O = (K, F, U, A, B, R, H, C, P, N),$$

the local NameCert$_{\text{BoundedRealOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source readback, continuous-map readback, uniform-modulus tolerance routing, finite oscillation-partition coverage, bounded-ledger exactness,* Real$^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and terminal local naming. No oscillation-bound obligation is stored outside those rows.*

Proof. Project the carrier of Definition 3985.1. The rows $K, F, U, A, B, R, H, C, P, N$ are the only local coordinates, so carrier admission and local naming are exactly the displayed finite packet.

Read the analytic rows in order. The compact source row K supplies only finite metric windows, the continuous-map row F supplies the real-valued graph readbacks on those windows, and the uniform-modulus row U translates an oscillation tolerance into the metric radius used by the partition row.

The finite partition row A then exposes the cellwise oscillation checks, and the bounded ledger B records the real bound obtained from those checks. The seal row R is downstream of this route, so a real-facing consumer cannot read a bound before the compact, continuous, modulus, partition, and ledger rows have been displayed.

Finally H and C transport and replay the same route componentwise, while P records provenance and N names the packet. Since the carrier has no excluded host-level coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3008.3 (BoundedRealOscillation compact-uniform route)**.** *Every compact-uniform-continuity or extreme-value consumer of an accepted* BoundedRealOscillation$^{\uparrow}$ *packet receives the oscillation bound only through the route*

$$\text{CompactMetric}^{\uparrow},\ \text{ContinuousMap}^{\uparrow},\ \text{UniformModulus}^{\uparrow},\ \text{FiniteOscillationPartition}^{\uparrow},\ \text{Real}^{\uparrow},\ H, C, P, N.$$

The route is finite: a requested tolerance opens the uniform modulus, the modulus selects the displayed metric radius, the finite oscillation partition checks the resulting cells, and the bound ledger seals the resulting Real$^{\uparrow}$ *value. It exports no open-cover compactness theorem, no selected supremum, no quotient real equality, and no choice-selected covering family.*

Proof. Start from the uniform-modulus row. A tolerance request is answered by U, and that answer is tied to the compact source K and continuous map F before any finite partition row is read.

Next use the finite oscillation partition row A. It inspects only the displayed cells determined by the compact-metric windows and the modulus radius, so the oscillation estimate is a finite ledger calculation rather than an open-cover argument.

The bound ledger B and Real seal R then expose the bounded value to downstream consumers. Since B is downstream of K, F, U, A and R is downstream of B, the consumer receives exactly the compact-uniform route displayed in the statement.

The transport, replay, provenance, and naming rows preserve that order. Therefore the packet can be consumed by extreme-value, uniform-limit, and compact-uniform-continuity chapters without supplying any open-cover, supremum-selection, quotient-equality, or choice coordinate. □

Theorem 3008.4 (BoundedRealOscillation Real-seal nonescape)**.** *For every accepted* $\mathsf{BoundedRealOscillation}^{\uparrow}$ *carrier* $O = (K, F, U, A, B, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read of the oscillation bound factors through* K, F, U, A, B *before* R *is visible, and through* H, C, P, N *before the local certificate is exported. Thus the real seal cannot be used as a bypass around compact-source readback, continuous-map readback, uniform-modulus routing, finite-partition coverage, or the bounded-oscillation ledger.*

Proof. Read the seal coordinate in Definition 3985.1. The row R is terminal: it records a real bound only after the compact source, continuous map, uniform modulus, finite partition, and bounded ledger have been displayed.

Apply Definition 3985.2. The NameCert surface requires the same rows for carrier admission, tolerance routing, partition coverage, bounded-ledger exactness, and nonescape, so a Real-facing consumer has no direct path to R.

Finally use the structural rows. The hsame transport row H and the Cont replay row C preserve the route componentwise, while P and N record provenance and local naming. Hence the real seal remains bound to the displayed finite packet and cannot bypass any of its compact-uniform rows. □

Falsifiable prediction: Every accepted $\mathsf{BoundedRealOscillation}^{\uparrow}$ consumer must expose a compact metric source row, continuous real-valued map row, uniform modulus row, finite oscillation-partition row, bounded ledger, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before an oscillation bound is available.

Independence witness: compactmetric, continuousmap, uniformmodulus, finiteoscillationpartition, real: $\mathsf{BoundedRealOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the compact source, continuous map, uniform modulus, finite oscillation partition, bounded ledger, real seal, transport, replay, provenance, and local naming rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealOscillation}^{\uparrow}$ packet is generated from a CompactMetric source row, a ContinuousMap real-valued graph row, a UniformModulus tolerance route, a FiniteOscillationPartition cell ledger, a bounded-oscillation RealUp ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3985.1, Definition 3985.2, Definition 3985.3, and Definition 3985.4 seed the finite oscillation-bound route over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, open-cover compactness, selected suprema, quotient real equality, countable choice, or ambient completeness outside the displayed real seal.

Upgrade path: Obligation closure requires checked carrier admission, compact-source readback, continuous-map readback, uniform-modulus routing, finite-partition coverage, bounded-ledger exactness, real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.BoundedRealOscillationUp.

Chapter 3009

A Concrete Naming Certificate for $\mathsf{BishopCompactInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompactInterval}^{\uparrow}$ packet records a constructive compact interval as finite located endpoint, closed-bounded interval, finite-net, and completion-facing NameCert data. It sits beside the closed bounded interval source of chapter 1591, the located endpoint surface of chapter 1709, the compact finite-net packet of chapter 3969, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies an L10 interval neighbor for Real and Completion routes without claiming open-cover compactness, quotient real equality, selected subcovers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3009.1 (Bishop compact interval carrier)**.** *A* $\mathsf{BishopCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, rational containment, dyadic containment, finite windows, regular readback, and Real sealing for the interval.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, exposing located endpoint reads, radius data, locator responses, and metric bracket rows for the same interval source.*
3. *F is the* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *finite-net row, recording the compact-interval coverage surface accepted for downstream finite-net consumers.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the interval and finite-net rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling interval mesh and endpoint precision requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the endpoint and interval readback route.*
8. *H records componentwise* hsame *transport for source interval, located interval, finite-net, stream, regular, dyadic, Real, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from interval containment and located endpoint reads through finite-net handoff, dyadic tolerance, regular readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these eleven displayed rows. It has no coordinate for host open covers, selected subcovers, quotient representatives of real endpoints, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompactIntervalUp.BishopCompactIntervalTasteGate single carrier alignment`

Theorem 3009.2 (Bishop compact interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *carrier* $K = (I, L, F, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed closed-bounded interval, located interval, finite-net, stream, regular, dyadic, Real, transport, replay, provenance, and local naming rows;*
2. *interval-source compatibility, requiring the located row L and closed-bounded row I to name the same endpoint route before any compact read is exposed;*
3. *finite-net admission, requiring compact-interval coverage to pass through F rather than through a host open-cover principle;*
4. *dyadic-window exactness, requiring every mesh or tolerance request to read D before the finite windows W and regular readback R are consumed;*
5. *Real-seal nonescape, requiring the endpoint and interval readback route to reach E only after I, L, F, W, R, D have been displayed;*
6. *transport, replay, provenance, and local naming exhaustion through H, C, P, N.*

Proof. Carrier admission is projection from Definition 3986.1. The tuple lists the complete finite row inventory made available to the local certificate.

The interval source is read through I and L. Their endpoint, locator, containment, and bracket rows must agree on the same displayed interval route before compactness-facing consumers can inspect the finite-net row.

Finite-net admission is then confined to F. Since the carrier has no open-cover or subcover coordinate, compactness can be exposed only as the accepted closed-bounded interval finite-net row.

The dyadic and window obligations are read through D, W, R. Tolerance requests first enter the dyadic ledger, then open finite stream windows, and only then reach regular rational readback.

Finally E, H, C, P, N seal and replay the same route. These rows preserve the packet under transport and provenance, while row exhaustion rules out quotient representatives, selected subcovers, choice schedules, and ambient Real completeness. □

Theorem 3009.3 (Bishop compact interval finite-net handoff)**.** *Every Real- or Completion-facing consumer of an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalCompact}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Stre}$$

The handoff exposes interval endpoints, locatedness, finite compact-interval coverage, dyadic tolerance, finite windows, regular readback, Real sealing, transport, replay, provenance, and local

naming, but it exports no open-cover compactness theorem, selected subcover, quotient endpoint equality, countable-choice schedule, or ambient completeness principle.

Proof. Project the accepted carrier $K = (I, L, F, W, R, D, E, H, C, P, N)$. The interval enters through the closed-bounded row I and the located interval row L, so endpoint and locator information are displayed before any compact read occurs.

Next read the finite-net row F. This row is the only compactness-facing coordinate in the carrier, and it is tied to the same interval source by the local obligation surface.

The completion-facing route then passes through D, W, R, E. The dyadic tolerance ledger controls the inspected mesh, the stream windows expose finite data, the regular row gives readback, and the Real seal is terminal for the route.

The structural rows H, C, P, N preserve this ordered handoff for downstream consumers. Since the packet contains no open-cover, quotient, choice, or ambient-completeness coordinate, no stronger compactness principle is exported. □

Theorem 3009.4 (Bishop compact interval finite-net extraction)**.** *Every Real- or Completion-facing consumer of an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet*

$$K = (I, L, F, W, R, D, E, H, C, P, N)$$

can extract the displayed $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *finite-net row F only after reading the interval-source rows I, L, the dyadic ledger D, the stream-window row W, the regular readback row R, the Real seal E, and the structural rows H, C, P, N. The extraction supplies the finite-net row as part of the displayed compact-interval route; it does not supply open-cover compactness, selected subcovers, quotient endpoint equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3986.1. The tuple displays F only together with the interval source, located endpoint, dyadic, stream, regular, Real, transport, replay, provenance, and local naming rows.

Use the local NameCert obligations of Definition 3986.2. Interval-source compatibility binds I and L before compact data is exposed, and finite-net admission confines compactness-facing access to the displayed row F.

Apply the handoff theorem Definition 3986.3. It orders the same rows for Real- and Completion-facing consumers, placing finite-net coverage after the interval source and before dyadic tolerance, finite windows, regular readback, and terminal Real sealing.

Since the carrier has no coordinate for open covers, selected subcovers, quotient endpoint equality, choice schedules, or ambient completeness, extracting F cannot export any of those stronger objects. □

Theorem 3009.5 (Bishop compact interval completion readback)**.** *Any completion-facing read from an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. The readback exposes the closed-bounded interval source, located endpoint route, finite compact-interval coverage, dyadic tolerance ledger, finite stream windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming, but it cannot be read as ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected subcover compactness, quotient endpoint equality, countable choice, or an unrestricted host limit.*

Proof. Project the carrier of Definition 3986.1. Its only completion-facing rows are $I, L, F, D, W, R, E, H, C, P, N$.

Apply Definition 3986.2. The obligation surface fixes interval-source compatibility through I, L, finite-net admission through F, and dyadic-window exactness through D, W, R before the Real seal can be consumed.

Apply Definition 3986.3. The theorem orders the same rows as the public finite route from closed-bounded and located interval data to finite-net coverage, dyadic tolerance, stream windows, regular readback, and terminal Real sealing.

Thus a completion consumer can read $\mathsf{Real}^{\uparrow}$ only at the end of the displayed handoff. Since the carrier has no open-cover, selected-subcover, quotient, choice, or ambient-completeness coordinate, the readback cannot export any of those stronger objects. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompactInterval}^{\uparrow}$ completion-facing read passes through ClosedBoundedIntervalUp, LocatedRealIntervalUp, ClosedBoundedIntervalCompactUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, transport, replay, provenance, and local naming rows before any endpoint or finite-net output is visible.

Independence witness: closedboundedinterval, located-real-interval, closed-bounded-interval-compact, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopCompactInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries closed-bounded interval data, located endpoint reads, finite compact-interval coverage, dyadic tolerance, finite windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactInterval}^{\uparrow}$ packet is generated from a ClosedBoundedInterval row, a LocatedRealInterval row, a ClosedBoundedIntervalCompact finite-net row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3986.1, Definition 3986.2, and Definition 3986.3 seed the finite constructive compact-interval route over closed-bounded, located, finite-net, dyadic, stream, regular, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient endpoint equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source compatibility, finite-net admission, dyadic-window exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactInterval

Chapter 3010

A Concrete Naming Certificate for $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Kelley on uniform and metric extension, the $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ packet records the completion reflector as a left-adjoint NameCert boundary. It sits after the metric completion carrier of chapter 1506, the dense embedding route of chapter 3351, the Cauchy completion adjunction surface of chapter 1456, the metric adjunction packet of chapter 3836, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the reflector boundary without quotient representatives, host equality of completions, countable choice, selected limits, or ambient completeness.

Definition 3010.1 (MetricCompletionLeftAdjoint carrier)**.** *A* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, D, A, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source reflector row.*
2. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit row exposing the dense source inclusion.*
3. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *adjunction row for completion-facing universal reads.*
4. *J is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *metric adjunction row binding the same source, unit, counit, and triangle ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row opened by completion consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, D, A, J, R have been displayed.*
7. *H records componentwise* hsame *transport for reflector, dense embedding, adjunction, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric-source request through the reflector, dense unit, adjunction handoff, regular readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbeddi}$ $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten displayed rows. It has no coordinate for quotient completion classes, chosen representatives, host equality of completion objects, countable-choice schedules, selected limits, or ambient complete metric spaces.

Lean4 ✓ `BEDC.Derived.MetricCompletionLeftAdjointUp.MetricCompletionLeftAdjointTasteGate single carrier alignment`

Theorem 3010.2 (MetricCompletionLeftAdjoint NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionLeft}$ *carrier* $L = (M, D, A, J, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed reflector, dense embedding, adjunction, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *unit exposure, requiring every source-to-completion read to pass through* D *before the adjunction rows are consumed;*
3. *left-adjoint handoff, requiring* A *and* J *to name the same finite completion reflector route before any universal consumer reads it;*
4. *triangle and extension visibility, requiring the unit, counit, and triangle comparison inherited from the adjunction rows to remain displayed through* H *and* C*;*
5. *RegSeqRat/Real seal nonescape, requiring* R *and* E *to be reached only after the reflector and adjunction rows have been displayed;*
6. *exhaustion of transport, replay, provenance, and local naming through* H, C, P, N*.*

Proof. Carrier admission is projection from Definition 3987.1. The tuple displays the full finite row inventory made available to the local certificate.

The source boundary is read through M and D. The reflector row supplies the completion-facing carrier, while the dense embedding row is the displayed unit of the reflector.

The universal boundary is then read through A and J. The Cauchy adjunction and metric adjunction rows provide the same finite source, unit, counit, and triangle ledger, so the left-adjoint handoff is a NameCert route rather than a host categorical equality.

Finally R, E, H, C, P, N seal and replay the route for downstream consumers. Since quotient classes, selected representatives, choice schedules, host equality, and ambient completeness are absent from the carrier, the listed obligations exhaust the certificate surface. □

Theorem 3010.3 (MetricCompletionLeftAdjoint universal handoff)**.** *Every separated-completion, Real-facing, or universal-completion consumer of an accepted* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionDenseEmbedding}^{\uparrow} \longmapsto \mathsf{CauchyCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{MetricCompletion}$$

The consumer receives source metric admission, dense unit exposure, adjunction handoff, triangle visibility, regular readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. It receives no quotient completion, selected representative, host equality of completions, countable choice, selected limit, or ambient completeness principle.

Proof. Project the accepted carrier $L = (M, D, A, J, R, E, H, C, P, N)$. A universal read opens M first and then the dense unit row D, so the source-to-completion boundary is displayed before the adjunction rows are available.

Next consume A and J. These rows expose the Cauchy and metric adjunction handoffs over the same finite reflector route, including the unit, counit, and triangle comparison needed by universal consumers.

The read then passes through R and E. Regular rational readback and the terminal Real seal are downstream from the displayed reflector and adjunction rows, not independent completeness assumptions.

The structural rows H, C, P, N preserve the ordered handoff under transport, replay, provenance, and local naming. Row exhaustion excludes quotient representatives, selected limits, host equality, countable choice, and ambient completeness from the exported certificate. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ universal read factors through MetricCompletion, MetricCompletionDenseEmbedding, CauchyCompletionAdjunction, MetricCompletionAdjunction, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Independence witness: metriccompletion, metric-completion-dense-embedding, cauchy-completion-adjunction, metric-completion-adjunction, regseqrat, real: $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly packages reflector admission, dense unit exposure, adjunction handoff, triangle visibility, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ packet is generated from a MetricCompletion reflector row, a MetricCompletionDenseEmbedding unit row, CauchyCompletionAdjunction and MetricCompletionAdjunction handoff rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3987.1, Definition 3987.2, and Definition 3987.3 seed the finite left-adjoint completion reflector route over metric completion, dense embedding, adjunction, regular readback, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host equality of completions, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, left-adjoint handoff, triangle visibility, RegSeqRat/Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionLeftAdjointUp.

Chapter 3011

A Concrete Naming Certificate for $\mathsf{CFun}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CFun}^{\uparrow}$ packet records an ordinary continuous real-facing map as finite source, target, modulus, transport, replay, provenance, and local naming rows. It is an L10 neighbour of the terminal $\mathsf{Real}^{\uparrow}$ seal in chapter 38, the finite windows of chapter 126, the regular rational readback of chapter 301, the formal-ball radius interface of chapter 1510, and the carrier-only packet of chapter 3611. The chapter names the finite continuity certificate consumed by $\mathsf{ContinuousMap}^{\uparrow}$, compact-uniform-continuity, and modulus packets without treating host functions as primitive objects.

Definition 3011.1 (Continuous function carrier)**.** *A* $\mathsf{CFun}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, M, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-window row, exposing the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and source-domain histories inspected by the continuity request.*
2. *T is the target-readback row, exposing the target* $\mathsf{Real}^{\uparrow}$ *or sibling endpoint readback reached by the displayed source window.*
3. *M is the finite modulus row, translating each target tolerance request into the source-window budget used by the packet.*
4. *Q is the continuity-certificate row binding S, T, M to the displayed request.*
5. *W is the shared finite window ledger over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and any* $\mathsf{FormalBall}^{\uparrow}$ *radius window opened by the same request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row used when a continuous-function consumer asks for a real-valued endpoint.*
7. *H records componentwise* hsame *transport for source, target, modulus, certificate, window, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a source-window request through S, T, M, Q, W, R.*
9. *P is the* Pkg *provenance over* $\mathsf{CFun}^{\uparrow}$, $\mathsf{CFunCarrier}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, T, M, Q, W, R, H, C, P, N$. It has no coordinate for host function extensionality, countable choice, quotient equality of maps or reals, selected modulus functions, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.ContinuousFunctionUp.ContinuousFunctionTasteGate single carrier alignment`

Theorem 3011.2 (Continuous function NameCert obligations)**.** *For every accepted* $\mathsf{CFun}^\uparrow$ *carrier $F = (S, T, M, Q, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$ *surface consists exactly of source-window admission through S, target readback through T, finite modulus exposure through M, continuity-certificate binding through Q, shared* $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, *and* $\mathsf{FormalBall}^\uparrow$ *window routing through W, real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3988.1. The rows $S, T, M, Q, W, R, H, C, P, N$ are the only local coordinates, so source admission, target readback, modulus exposure, continuity binding, window routing, readback, transport, replay, provenance, and naming must all be read there.

The analytic route is finite. The modulus row M translates a target tolerance into a source-window budget, Q binds that budget to the displayed source and target rows, and W, R expose the StreamName, RegSeqRat, FormalBall, and Real-facing readbacks attached to the same request.

Finally H, C, P, N preserve and name the same route. Since the carrier has no host-function, function-extensionality, quotient-equality, choice, selected-modulus, or ambient-completeness coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 3011.3 (Continuous function Real-map handoff)**.** *Every* $\mathsf{ContinuousMap}^\uparrow$, *compact-uniform-continuity, or real-valued modulus consumer of an accepted* $\mathsf{CFun}^\uparrow$ *packet receives the map only through*

$$S, M, Q \longmapsto W \longmapsto T, R \longmapsto H, C, P, N.$$

Thus the consumer may use the source window, finite modulus, continuity certificate, shared $\mathsf{StreamName}^\uparrow$ *and* $\mathsf{RegSeqRat}^\uparrow$ *windows, optional* $\mathsf{FormalBall}^\uparrow$ *radius interface, target readback, real seal, componentwise transport, displayed replay, provenance, and local naming rows. It receives no host function object, function extensionality principle, countable-choice selector, quotient equality of maps or reals, selected limit, or ambient* $\mathsf{Real}^\uparrow$ *completeness witness.*

Proof. Begin with the source and modulus rows. A consumer request opens S and M before any target readback is available, and Q is the only row binding the source budget to the target tolerance.

Next read the finite windows. The row W exposes the relevant $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{FormalBall}^\uparrow$ faces, while T and R expose the target and real-facing readbacks reached by that same finite route.

The rows H, C, P, N then transport, replay, source, and name the handoff. Row exhaustion in Definition 3988.1 excludes every host-function, function-extensionality, quotient-equality, choice, selected-limit, and ambient-completeness escape route. □

Falsifiable prediction: Every accepted $\mathsf{CFun}^\uparrow$ read must expose source window, target readback, finite modulus, continuity certificate, StreamName and RegSeqRat windows, optional FormalBall radius routing, Real-facing readback, componentwise transport, displayed replay, provenance, and local naming rows before a continuous-map or modulus consumer can inspect it.

Independence witness: real, streamname, regseqrat, formalball, continuousfunctioncarrier, continuousmap: $\mathsf{CFun}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source-window, target-readback, finite-modulus, continuity-certificate, shared-window, real-readback, transport, replay, provenance, and local naming rows without quotienting maps or importing host functions.

Closure status for $\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CFun}^{\uparrow}$ packet is generated from source-window rows, target readback rows, finite modulus rows, continuity-certificate rows, shared StreamName, RegSeqRat, and FormalBall windows, real-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$, seed)

Scope closed: Definition 3988.1, Definition 3988.2, and Definition 3988.3 seed the finite non-host continuous-function route over source windows, target readbacks, modulus data, continuity certificates, window ledgers, readbacks, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, countable choice, quotient map equality, quotient real equality, selected modulus functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-window access, target readback, finite modulus exposure, continuity-certificate binding, shared window routing, real-facing handoff, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousFunctionUp.

Chapter 3012

A Concrete Naming Certificate for $\mathsf{CounterfactualOutcome}^{\uparrow}$

Following Pearl, Causality, 2009, the $\mathsf{CounterfactualOutcome}^{\uparrow}$ packet records paired factual and counterfactual outcome reads as finite causal-inference NameCert data. It sits beside the intervention calculus of chapter 453, the intervention-row packet of chapter 1290, the probability-space carrier of chapter 184, the distribution handoff of chapter 186, and the Bayesian conditioning route of chapter 232. The packet gives causal consumers a selected counterfactual-outcome surface without weakening the explicit non-claim boundary of $\mathsf{DoCalculus}^{\uparrow}$.

Definition 3012.1 (Counterfactual outcome carrier)**.** *A* $\mathsf{CounterfactualOutcome}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N)$$

with the following displayed rows:

1. *U is the finite unit row, naming the observational unit and the displayed covariate cells opened by the counterfactual query.*
2. *I is the intervention row inherited from the* $\mathsf{CausalIntervention}^{\uparrow}$ *route. It records the do-row, intervention target, and intervention value that may be inspected by the counterfactual read.*
3. *Y_f is the factual outcome row observed for U before the counterfactual branch is exported.*
4. *Y_{cf} is the counterfactual outcome row selected under I for the same unit row U.*
5. *M is the modified-distribution row, read through the* $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, *and* $\mathsf{Bayesian}^{\uparrow}$ *siblings before any expectation or probability export.*
6. *R is the consistency row requiring factual and counterfactual reads to agree on the factual branch selected by the displayed intervention route.*
7. *H records componentwise* hsame *transport for units, interventions, factual outcomes, counterfactual outcomes, modified distributions, consistency rows, distribution exports, expectation rows, replay rows, provenance rows, and local naming rows.*
8. *D is the distribution handoff row. It exposes only the finite probability surface displayed by M and does not store hidden-world measures.*
9. *E is the expectation and export row for finite counterfactual queries.*

10. *T records displayed* Cont *replay of the route*

$$U, I, Y_f, Y_{cf}, M, R \longmapsto D, E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CounterfactualOutcome}^{\uparrow}$, $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{CausalIntervention}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Bayesian}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N$. It has no coordinate for hidden structural equations, cross-world equality outside the displayed consistency row, selected counterfactual universes, host structural causal models, or ambient probability semantics.

Lean4 ✓ `BEDC.Derived.CounterfactualOutcomeUp.CounterfactualOutcomeTasteGate single carrier alignment`

Theorem 3012.2 (Counterfactual outcome NameCert obligations)**.** *For every accepted* $\mathsf{CounterfactualOutcome}^{\uparrow}$ *carrier*

$$C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the finite unit row, intervention row, factual outcome row, counterfactual outcome row, modified-distribution row, consistency row, transport, distribution handoff, expectation and export row, replay, provenance, and local naming row;*

2. *intervention anchoring, requiring every counterfactual read Y_{cf} to cite the displayed I row and the same unit row U before any export through D or E;*

3. *factual and counterfactual consistency, requiring R to bind the factual branch of Y_{cf} to the observed Y_f without adding cross-world equality outside the carrier;*

4. *probability handoff, requiring M, D, E to pass through the displayed* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, *and* $\mathsf{Bayesian}^{\uparrow}$ *routes before any finite probability or expectation read is available;*

5. *non-escape from hidden worlds, host structural causal models, and selected counterfactual universes, requiring H, T, P, N to preserve only the displayed rows and to export no unlisted causal semantics.*

No counterfactual-outcome obligation is stored outside these rows.

Proof. Project the accepted carrier $C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N)$. Carrier admission exposes exactly the finite unit, intervention, factual outcome, counterfactual outcome, modified-distribution, consistency, transport, distribution, export, replay, provenance, and naming rows.

The counterfactual branch is anchored before it is exported. The row I supplies the intervention target and value, while U supplies the unit row shared by Y_f and Y_{cf}.

Consistency is then read through R. This row identifies the factual branch of the counterfactual read with the observed factual outcome, but it does not create an unconstrained cross-world equality or a hidden structural equation.

The probability-facing rows are downstream. The modified distribution M, distribution handoff D, and expectation row E are available only along the displayed probability, distribution, and Bayesian routes.

Finally H, T, P, N transport, replay, source, and name the same finite surface. Since no carrier coordinate stores hidden worlds, a host structural causal model, or a selected counterfactual universe, none can escape through the local NameCert. □

Falsifiable prediction: Every $\mathsf{CounterfactualOutcome}^{\uparrow}$ counterfactual-facing read factors through a finite unit row, a displayed intervention row, factual and counterfactual outcome rows, a modified-distribution row, a consistency row, componentwise hsame transport, distribution and expectation handoff rows, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: do-calculus, causal-intervention, probspace, distribution, bayesian: CounterfactualOutcor is not bijective to any listed sibling because it jointly carries paired factual and counterfactual outcomes for a unit, a displayed intervention anchor, a modified-distribution handoff, a consistency row, finite probability and expectation export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3989.1 and Definition 3989.2 seed the finite counterfactual-outcome route over unit, intervention, factual outcome, counterfactual outcome, modified-distribution, consistency, distribution, expectation, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hidden structural equations, host structural causal models, selected counterfactual universes, cross-world equality outside the displayed consistency row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, intervention anchoring, factual and counterfactual consistency, probability handoff, non-escape from hidden worlds, transport, replay, provenance, and local naming for BEDC.Derived.CounterfactualOutcomeUp.

Constructive story: A $\mathsf{CounterfactualOutcome}^{\uparrow}$ packet is generated from a finite unit row, a displayed intervention row, factual and counterfactual outcome rows, a modified-distribution row, a consistency row, componentwise hsame transport, distribution and expectation handoff rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3013

A Concrete Naming Certificate for $\mathsf{FiniteSupportRemoval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteSupportRemoval}^{\uparrow}$ packet records the constructive finite-cover operation that deletes one displayed support row while keeping the remaining finite cover, tolerance ledger, and terminal real-name route visible. It consumes the positive-radius surface of chapter 452, the compact finite-net route of chapter 3957, and the L10 dependency chain through chapter 126, chapter 301, and chapter 38. The packet names a support-pruning certificate for finite compactness consumers; it is not a selected subcover functional, a host-level minimum operation, a quotient equality principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3013.1 (Finite support removal carrier)**.** *A* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, G, D, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net or finite-cover source row, exposing the finite list of support names and coverage witnesses.*
2. *G is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius row that keeps every retained support row inside the displayed cover tolerance.*
3. *D is the deleted support name, together with its membership witness in the original finite support spine.*
4. *W is the retained finite window after removing D, with its inherited order and membership rows.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger transported from the original cover to the retained window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the retained window.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only through the retained* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *route.*
8. *H records componentwise* hsame *transport for K, G, D, W, T, R, L and for the structural rows below.*
9. *C records displayed* Cont *replay from finite-cover inspection through support deletion, tolerance transport, regular readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteSupportRemoval}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformFiniteNetRealizati}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, D, W, T, R, L, H, C, P, N$. It has no coordinate for an undisplayed cover member, a selected replacement support, quotient equality of real names, countable choice, or an ambient compactness theorem.

Theorem 3013.2 (Finite support removal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *carrier $S = (K, G, D, W, T, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite-cover source, positive-radius ledger, deleted support name, retained window, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows; membership exposure for D in the original support spine; retained-window coverage through W and G; L10 readback through T, R, L; and non-escape from selected replacement supports, quotient real equality, countable choice, and ambient compactness.*

Proof. Project the carrier of Definition 3990.1. The rows $K, G, D, W, T, R, L, H, C, P, N$ are the complete displayed inventory.

The deletion row is finite data. Since D is stored with its membership witness in the original finite support spine and W stores the retained window, the packet can inspect the removed coordinate without choosing a replacement support or leaving the finite cover.

The coverage and tolerance rows are inherited only through displayed coordinates. The positive-radius row G keeps the retained supports inside the finite Lebesgue-radius ledger, while T transports the dyadic tolerance data to the retained StreamName window.

Finally the regular readback and Real seal are read through R and L, then preserved by H, C, P, N. The refused objects are absent from the carrier, so the local NameCert surface is exactly the obligation list stated above. □

Lean4 ✓ `BEDC.Derived.FiniteSupportRemovalUp.FiniteSupportRemovalTasteGate single carrier alignment`

Theorem 3013.3 (Finite support removal compact-cover handoff)**.** *Every compactness or uniform-modulus consumer of an accepted* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *packet reads the deletion route through the finite chain*

$$\mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{CompactUniformFiniteNetRealization}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer first displays the original finite support spine, then the removed coordinate and retained window, then the positive-radius and dyadic tolerance rows, and only then the regular readback and Real seal. No consumer may replace the deleted support by an undisplayed cover member, a selected subcover, quotient real equality, countable choice, or ambient completeness.

Proof. Use Definition 3990.2. It orders the finite-cover row, the deleted support row, the retained window, the positive-radius ledger, the tolerance ledger, the regular readback, and the terminal Real seal inside one accepted packet.

The finite cover is consumed before deletion. Thus the support spine and the deleted coordinate are both visible BEDC histories, and the retained window W is a finite subspine rather than a host-level set difference.

The L10 dependency route is then forced. The retained StreamName window carries the dyadic tolerance row T, the regular rational readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal L through the same displayed route used by finite-cover compactness chapters.

Transport, replay, provenance, and local naming preserve this route without adding another support coordinate. Therefore the handoff supplies a constructive deletion primitive for compact finite-cover consumers while refusing the listed external objects. □

Falsifiable prediction: An accepted $\mathsf{FiniteSupportRemoval}^{\uparrow}$ read is refuted if a compactness or uniform-modulus consumer reaches a retained Real seal without displaying the original finite support spine, the deleted support name, the retained window, the positive-radius ledger, the tolerance row, the regular readback row, transport, replay, provenance, and local naming.

Independence witness: finitelebesguenumber, compactuniformfinitenetrealization, streamname, regseqrat, and real: none is bijective to the support-removal packet because only $\mathsf{FiniteSupportRemoval}^{\uparrow}$ binds a deleted finite support coordinate to a retained-window L10 readback route before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSupportRemoval}^{\uparrow}$ packet is generated from finite-cover source rows, FiniteLebesgueNumberUp positive-radius rows, a displayed deleted support name, retained StreamName window rows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3990.1, Definition 3990.2, and Definition 3990.3 seed the finite support-deletion surface for compactness and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected subcover extraction, quotient real equality, countable choice, host set difference, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, support membership exposure, retained-window coverage, positive-radius preservation, tolerance transport, regular readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSupportRemovalUp.

The $\mathsf{Cauchy}^{\uparrow}$ hub reads finite windows through chapter 126, dyadic tolerance through chapter 291, regular readback through chapter 301, and the terminal seal through chapter 38.

Chapter 3014

A Concrete Naming Certificate for $\mathsf{CauchyRemainderEstimate}^{\uparrow}$

Following Bishop and Bridges constructive analysis of regular Cauchy names, the $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ packet records finite tail estimates before a completion or real-name consumer is allowed to read a candidate limit surface. It sits on the Phase Real and Completion route after the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite observation handles of chapter 126, the regular rational readback of chapter 301, the tail-diameter budget of chapter 1746, and the real-name seal of chapter 38. The packet exposes a finite Cauchy tail remainder estimate; it does not select a limit, identify quotient streams, invoke countable choice, import ambient Real completeness, or turn a boundary counterexample into a convergent name.

Definition 3014.1 (Cauchy remainder estimate carrier)**.** *A* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, D, T, W, R, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *row whose observations are being tested on a finite tail.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested error bound, its positive slack, and the finite comparison budget used by the estimate.*
3. *T is the displayed threshold row, naming the finite tail index after which the estimate is allowed to read observations.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened only at indices admitted by T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations exposed by W and checked against the tolerance data in D.*
6. *B is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row, recording the displayed tail-diameter budget through which the same finite evidence is consumed by a* $\mathsf{Real}^{\uparrow}$ *seal route or by a refusal route.*
7. *H records componentwise* hsame *transport for the source, tolerance, threshold, window, readback, boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through D, T, W, R, B.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRemainderEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, W, R, B, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable choice, ambient completeness, host real equality, trichotomy, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyRemainderEstimateUp.CauchyRemainderEstimateTasteGate single carrier alignment`

Theorem 3014.2 (Cauchy remainder estimate NameCert obligations)**.** *For every accepted* CauchyRemainderEstima *carrier*

$$E = (S, D, T, W, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRemainderEstimate}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source, dyadic tolerance ledger, finite threshold, admitted windows, rational readback, boundary row, transport, replay, provenance, and local naming rows;*
2. *tolerance exposure, requiring each remainder request to read its bound and slack data from D before any tail observation is opened;*
3. *threshold exactness, requiring T to be the only finite index gate through which W may expose tail observations;*
4. *readback alignment, requiring R to compare only the observations admitted by W against the same tolerance request from D;*
5. *boundary nonescape, requiring B to route the estimate through the displayed* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *budget before any* $\mathsf{Real}^{\uparrow}$ *seal consumer may read it, without adding a selected limit or quotient stream;*
6. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N;*
7. *refusal of selected limits, quotient stream equality, countable choice, ambient completeness, host real equality, trichotomy, and bridge checking as local certificate data.*

No Cauchy-remainder-estimate obligation is stored outside these rows.

Proof. Project the carrier of Definition 3991.1. The accepted packet exposes exactly the source row S, dyadic tolerance ledger D, threshold row T, window row W, readback row R, boundary row B, and structural rows H, C, P, N.

A remainder request first reads D. This fixes the tolerance, the positive slack, and the finite comparison budget before any source observation is visible.

The request then reads T, W, R in order. The threshold row supplies the finite tail gate, the window row exposes only observations admitted by that gate, and the readback row compares those observations against the same dyadic tolerance data.

Finally, B determines whether the estimate is consumed by a Real seal route or a Specker refusal route, while H, C, P, N transport, replay, source, and name the same finite data. Since the excluded principles do not occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 3014.3 (Cauchy remainder estimate dyadic tail-bound admission)**.** *Every accepted* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ *carrier presents finite tail evidence through the route*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto R \longmapsto B.$$

Thus a consumer can inspect a Cauchy remainder only after the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request, finite threshold, admitted* $\mathsf{StreamName}^{\uparrow}$ *tail windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row have all been displayed.*

Proof. Start with an accepted consumer read of E. Carrier projection opens the source row S and the tolerance ledger D before any tail observation is available.

The finite tail gate is then supplied by T. The window row W is downstream from that gate, so it cannot expose observations outside the admitted finite tail schedule.

The readback row R compares only the windows selected by W with the tolerance data fixed by D. The boundary row B records the $\mathsf{CauchyTailDiameter}^{\uparrow}$ budget for the downstream consumer without changing the admitted observations. Hence every tail read factors through the displayed finite route. □

Theorem 3014.4 (Cauchy remainder estimate regular-Cauchy readback)**.** *If an accepted* CauchyRemainderEstimate *carrier is read as regular Cauchy data, then the readback exports only the displayed source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, finite threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *observations,* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

It exports no selected limit, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, or hidden modulus selector.

Proof. Assume a regular Cauchy consumer reads the packet. The carrier still supplies the consumer only through $S, D, T, W, R, B, H, C, P, N$.

The regular-facing route therefore receives a finite remainder estimate, not a limit-selection mechanism. The tolerance ledger and threshold row fix the request, the window and readback rows expose the rational observations, and the $\mathsf{CauchyTailDiameter}^{\uparrow}$ boundary row records the finite diameter budget used by the read.

The structural rows preserve and name that route. Since no selected-limit, quotient, choice, completeness, host-equality, trichotomy, or hidden-selector coordinate is present, the readback cannot export any of those objects. □

Theorem 3014.5 (Cauchy remainder estimate Real seal nonescape)**.** *If an accepted* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ *carrier is consumed by a* $\mathsf{Real}^{\uparrow}$ *route, then the seal is downstream of the same finite route*

$$S, D, T, W, R, B, H, C, P, N.$$

The $\mathsf{Real}^{\uparrow}$ *consumer cannot bypass the displayed dyadic tolerance, stream windows, regular rational readback, or tail-diameter budget by invoking selected limits, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, or bridge checking.*

Proof. Assume the boundary row reaches a Real seal consumer. The carrier projection supplies no Real-facing coordinate except the row B together with the displayed source, tolerance, threshold, window, readback, transport, replay, provenance, and local naming rows.

Apply the dyadic tail-bound admission route. It forces the Real-facing read to pass through D, T, W, R before the boundary row can be consumed.

Apply the regular-Cauchy readback theorem. The readback row exports only the regular rational observations admitted by the same finite tolerance and window data. The Real seal therefore receives a finite remainder estimate rather than an independent limit object. □

Falsifiable prediction: Every accepted CauchyRemainderEstimate$^{\uparrow}$ consumer read factors through a source row, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a finite threshold, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, a CauchyTailDiameter$^{\uparrow}$ boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-tail-diameter, real: CauchyRemainderEstim is not bijective to any listed sibling because it jointly binds a source stream, dyadic tolerance budget, finite tail threshold, admitted observation windows, rational readback, tail-diameter boundary, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$.

Constructive story: A CauchyRemainderEstimate$^{\uparrow}$ packet is generated from a displayed source stream or regular rational name, a DyadicRatCore tolerance ledger, a finite threshold, StreamName tail windows, RegSeqRat readback, a CauchyTailDiameter boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$, seed)

Scope closed: Definition 3991.1, Definition 3991.2, Definition 3991.3, Definition 3991.4, and Definition 3991.5 seed the finite tail-estimate route for Real and Completion consumers without selected limits or quotient stream equality.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyRemainderEstimateUp.CauchyRemainderEstimateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, Specker convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exposure, threshold exactness, readback alignment, boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRemainderEstimateUp.

Theorem 3014.6 (Cauchy carrier root obligations)**.** *Let*

$$R = (S, D, Q, T, E, H, C, P, N)$$

be an accepted root Cauchy$^{\uparrow}$ *remainder carrier. Here S is the finite* StreamName$^{\uparrow}$ *window row, D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, Q is the* RegSeqRat$^{\uparrow}$ *readback row, T is the* CauchyTailDiameter$^{\uparrow}$ *tail-diameter row, E is the terminal* Real$^{\uparrow}$ *seal row, and H, C, P, N are the componentwise transport, displayed replay, provenance, and local naming rows. The local root obligation surface is exactly carrier admission for these rows, tolerance exactness through D, readback exposure through Q, tail-diameter exposure through T, terminal sealing through E, and structural stability through H, C, P, N.*

No accepted root carrier contains a selected limit, quotient stream, countable-choice schedule, host real equality, or ambient Real$^{\uparrow}$ *completeness coordinate.*

Proof. Project the displayed BHist carrier R. The stream row S is the only source of visible observations, and the dyadic ledger D is the only source of precision requests and comparison budgets.

The regular readback row Q is downstream of S and D, while the tail-diameter row T is downstream of the same finite windows and tolerances. Thus the root carrier records a finite remainder surface rather than a selected completed object.

The terminal seal row E may be read only after S, D, Q, T have been displayed. The rows H, C, P, N transport, replay, source, and name that same finite route. Since there are no further coordinates, the listed rows exhaust the root obligation surface and the excluded principles cannot be exported. □

Theorem 3014.7 (Cauchy tail-envelope root obligations)**.** *For every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *remainder carrier* $R = (S, D, Q, T, E, H, C, P, N)$*, every tail-envelope read factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyTailDiameter}^{\uparrow}.$$

The tail-diameter row T *may compare only the finite windows exposed by* S*, the dyadic budgets exposed by* D*, and the regular rational readbacks exposed by* Q*. It cannot introduce quotient equality of streams, a selected limit, countable choice, host real equality, or an ambient completeness theorem.*

Proof. Start with a tail-envelope request. Carrier projection exposes the finite StreamName row S before any tolerance, readback, or diameter claim is available.

The dyadic ledger D supplies the displayed tolerance budget. The RegSeqRat row Q reads the same finite windows under that budget, so a tail comparison cannot cite an independent stream representative or an external choice schedule.

The CauchyTailDiameter row T is then read as a finite envelope over S, D, Q. Transport and replay through H, C preserve those rows componentwise, while P, N record provenance and local naming. Hence the tail-envelope route remains inside the displayed finite packet. □

Theorem 3014.8 (Cauchy Real-seal root obligations)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted root* $\mathsf{Cauchy}^{\uparrow}$ *remainder carrier must read the terminal seal row* E *only after the finite path*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyTailDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

has been displayed. The seal is a terminal BEDC row for the shown remainder carrier; it is not a selected limit, quotient-stream representative, countable-choice output, host real equality, or ambient real-completeness witness.

Proof. Project $R = (S, D, Q, T, E, H, C, P, N)$. The rows S, D, Q, T are the only analytic inputs to the terminal seal: finite stream windows, dyadic tolerances, regular rational readback, and tail-diameter evidence.

The Real seal E is downstream of that path. A consumer may read it only after the displayed Cauchy remainder route is visible, so the seal cannot be used to manufacture a quotient stream or a selected completed limit.

The structural rows H, C, P, N attach to the same carrier. They preserve and replay the finite path and do not add host real equality, countable choice, or ambient completeness. Therefore every Real-facing root read is exhausted by the stated path. □

Definition 3014.9 (Cauchy seed carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier is a finite* BHist *packet*

$$S = (W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observation windows that a Cauchy consumer may inspect.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the displayed dyadic precision requests and comparison budgets for those windows.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, obtained only from the displayed stream windows and dyadic tolerance data.*

4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed after the regular rational readback, not a selected limit or host real equality witness.*
5. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an admitted Cauchy request through W, D, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The classifier compares accepted seed packets only through W, D, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3014.10 (Cauchy seed NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier $S = (W, D, R, E, H, C, P, N)$, the local seed-level* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming;*
2. *tolerance exactness, requiring every Cauchy comparison to read its dyadic budget from D before any regular rational readback is accepted;*
3. *window readback exactness, requiring R to be generated from the displayed windows W and tolerances D;*
4. *Real seal discipline, requiring E to remain downstream of W, D, R and never to serve as a selected limit or host equality object;*
5. *structural stability under* hsame *transport H and displayed* Cont *replay C;*
6. *provenance and local naming through P and N.*

No seed obligation certifies quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 3991.9. The accepted packet exposes only the finite window row, dyadic tolerance row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Tolerance exactness is read from D after the window row W has been displayed. The regular readback row R is downstream of both rows, so it cannot introduce an independent rational sequence or a choice schedule.

The Real seal row E is terminal for the seed route. It is consumed only after W, D, R are visible, and therefore cannot act as a selected limit, quotient equality, or host real equality witness.

Finally H and C preserve the same component rows under transport and replay, while P and N source and name the packet. Since no remaining coordinate exists, the listed rows exhaust the seed-level NameCert surface. □

Theorem 3014.11 (Cauchy seed Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming preserved by* H, C, P, N*. The seed route exports finite stream windows, dyadic tolerances, regular rational readback, and a terminal Real seal, but exports no quotient stream equality, selected limit, host real equality, trichotomy, countable-choice selector, or ambient completeness principle.*

Proof. Begin with a Real-facing or completion-facing read from $S = (W, D, R, E, H, C, P, N)$. Carrier admission exposes the finite StreamName window row W before the tolerance and readback rows can be consumed.

The dyadic tolerance row D supplies the displayed precision budget for the read. The regular rational readback row R then packages the windowed data under that budget, rather than selecting an external limit or invoking a host sequence.

The Real seal row E is reached only after W, D, R have been displayed. Structural transport and replay through H, C keep the same route, and P, N record provenance and local naming.

Because the carrier has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient completeness, every exported Real-facing read remains inside the seed route stated above. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedRealSealBoundary`

Theorem 3014.12 (Cauchy tail request carrier)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *root tail-request surface is the finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

whose rows are finite $\mathsf{StreamName}^{\uparrow}$ *windows, a directed later-window request,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming. It is the same finite request surface isolated by Definition 5821.1 and consumed by chapter 5817.*

Proof. Project the Cauchy seed carrier of Definition 3991.9. The rows W, D, R, E, H, C, P, N already expose finite windows, dyadic tolerance, regular readback, Real sealing, transport, replay, provenance, and local naming.

Insert the directed request row Q between the window row and the tolerance row. The resulting tuple is exactly the $\mathsf{CauchyTailRequest}^{\uparrow}$ carrier of Definition 5821.1, with the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling anchors.

Because Definition 5817.1 reads its tail coordinate as a finite directed request row, the root Cauchy packet supplies the carrier surface required by the finite filter route without adding quotient filters, selected limits, host equality, choice, or Real completeness. □

Theorem 3014.13 (Cauchy directed-tail NameCert obligations)**.** *For the root tail-request packet* $T = (W, Q, D, R, E, H, C, P, N)$ *of Definition 3991.12, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *directed-tail obligation surface consists exactly of carrier admission, request classifier visibility, stability under later-window refinement, tolerance exactness, readback exactness, ledger routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations are read through the sibling rows of chapter 126, chapter 291, chapter 301, chapter 38, and chapter 5817.*

Proof. Carrier admission is the tuple display in Definition 3991.12. The request classifier is the row Q, and it is visible only after the finite window row W has named the inspected StreamName windows.

Stability and tolerance exactness are read from Q and D together. A later-window refinement changes only the displayed request route and must keep the dyadic budget attached to that route before any regular rational readback is accepted.

Readback exactness and ledger routing place R downstream of W, Q, D and place E downstream of R. Thus the Real-facing seal cannot serve as a selected limit, quotient-stream comparison, host equality witness, or ambient completeness principle.

The structural rows H, C, P, N supply transport, replay, provenance, and local naming. Projecting Definition 5821.2 gives the same row list for the named request surface, while Definition 5817.2 identifies the finite-filter consumer that reads that request. □

Theorem 3014.14 (Cauchy filter Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing Cauchy consumer that reads a filter, completion, or regular-name seal factors through the finite tail-request packet of Definition 3991.12 before it can read* E*. The route is*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows preserved throughout.*

Proof. Start with a Real-facing Cauchy read. The seed boundary of Definition 3991.11 already forces the read through finite windows, dyadic tolerance, regular readback, and the terminal Real seal.

Refine the seed route by Definition 3991.12. The finite request row Q is inserted before the tolerance ledger, so no downstream consumer reaches D, R, or E without first reading the directed-tail request surface.

Apply Definition 3991.13. Its ledger and exactness clauses keep W, Q, D, R, E in order, and its structural clauses preserve the same route through H, C, P, N.

Finally compare with Definition 5817.3. The filter consumer reads the same finite route before consuming the Real seal, so every Real-facing Cauchy answer factors through the displayed tail request rather than through quotient filters, selected limits, countable choice, host equality, or Real completeness. □

Theorem 3014.15 (Cauchy seed closure route)**.** *Every public route through the* $\mathsf{Cauchy}^{\uparrow}$ *hub is read through the sibling surfaces of chapter 3413, chapter 126, chapter 301, chapter 291, chapter 3991, and chapter 38 before any Real-facing completion consumer can use it. The route exposes finite stream windows, dyadic tolerance rows, regular rational readback, Cauchy remainder estimates, terminal real seals, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier of Definition 3413.1. Its public readback is already ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row.

Apply Definition 3413.3 and Definition 3413.4. These results force every regular rational read to pass through finite stream windows, dyadic tolerances, the displayed modulus row, and the compatibility row before R is inspected.

Apply Definition 3991.3 for completion-facing tolerance reads. The remainder estimate route inserts the finite threshold and admitted tail windows before any Real or completion consumer can treat the Cauchy packet as usable tail evidence.

Finally apply Definition 3413.9. The terminal Real-facing seal remains downstream of the displayed regular rational and remainder-estimate routes. Since none of the cited sibling surfaces

carries selected limits, quotient stream equality, choice, trichotomy, host equality, or ambient completeness, the hub route exports none of those objects. □

Theorem 3014.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 3014.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 3014.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic

budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 3015

A Concrete Naming Certificate for $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ packet records the finite comparison surface for two displayed Cauchy tail-envelope reads before either branch reaches the terminal real-name seal. It sits after the dyadic tail envelope of chapter 1659, the tail-diameter boundary of chapter 1746, the finite stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name boundary of chapter 38. The packet names a paired finite tail route; it is not quotient stream equality, a selected common limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3015.1 (Cauchy tail-envelope comparison carrier)**.** *A* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E_0, E_1, W_0, W_1, D, J, T, H, C, P, N)$$

with the following displayed rows:

1. E_0 *and* E_1 *are accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows naming the two tail-envelope reads being compared.*
2. W_0 *and* W_1 *are the* $\mathsf{StreamName}^{\uparrow}$ *tail windows opened by the two envelope reads.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the two displayed tails.*
4. J *is the common-window row, recording the finite refinement through which* W_0 *and* W_1 *are read before either branch reaches a* $\mathsf{Real}^{\uparrow}$ *seal.*
5. T *is the paired tail-diameter row, recording the displayed comparison with the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary.*
6. H *records componentwise* hsame *transport for the two envelope rows, stream windows, shared tolerance, common-window refinement, tail-diameter row, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a paired precision request through* E_0, E_1, J, D, T.
8. P *is the* Pkg *provenance over* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyTailEnvelopeComparisonUp.CauchyTailEnvelopeComparisonTasteGate`
`single carrier alignment`

Theorem 3015.2 (Cauchy tail-envelope comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *carrier*

$$Q = (E_0, E_1, W_0, W_1, D, J, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two dyadic tail-envelope rows, their stream windows, the shared dyadic tolerance, the common-window refinement, the paired tail-diameter row, transport, replay, provenance, and local naming rows;*
2. *envelope exposure, requiring both branches to display their* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows before the paired comparison is consumed;*
3. *common-window admission, requiring J to be fixed before either branch reads a* $\mathsf{RegSeqRat}^{\uparrow}$ *row toward* $\mathsf{Real}^{\uparrow}$*;*
4. *tolerance alignment, requiring both tail reads to use the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D;*
5. *diameter comparison, requiring the paired tail-diameter row T to be read through the same common window and tolerance data;*
6. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N;*
7. *refusal of quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking as local certificate data.*

No comparison obligation is stored outside these rows.

Proof. Project the carrier of Definition 3992.1. The accepted packet exposes exactly the two dyadic tail-envelope rows E_0, E_1, their opened stream windows W_0, W_1, the shared tolerance row D, the common-window row J, the tail-diameter row T, and the structural rows H, C, P, N.

The envelope exposure clauses follow from the two displayed $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ rows. Each branch must open its finite stream window before the paired comparison can read any regular rational data.

The common-window and tolerance clauses are the role of J and D. They force the two branches to pass through one finite refinement and one dyadic budget before the paired tail-diameter row is consumed.

The remaining rows are structural. The hsame row transports each displayed coordinate componentwise, the Cont row replays the same paired route, and the Pkg and $\mathsf{NameCert}$ rows source and name only that finite surface.

Since none of the excluded principles occurs as a carrier coordinate, the local certificate cannot export quotient stream equality, a selected common limit, countable choice, trichotomy, host real equality, ambient Real completeness, or bridge checking. □

Theorem 3015.3 (Cauchy tail-envelope comparison common window)**.** *Every accepted* CauchyTailEnvelopeCompar *carrier compares its two displayed tail-envelope reads through a single finite common-window route*

$$E_0, E_1 \longrightarrow J \longrightarrow W_0, W_1 \longrightarrow D \longrightarrow T.$$

Consequently any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the paired comparison receives the same displayed common-window row before either branch can present a terminal real-name seal.*

Proof. Begin with the two accepted envelope rows. By Definition 1659.4, paired dyadic tail-envelope reads serving the same displayed precision request admit a finite common route before any $\mathsf{Real}^{\uparrow}$ seal is inspected.

The carrier row J records that common route for the comparison packet. Once J is fixed, both stream windows W_0 and W_1 are read as displayed rows over the same refinement rather than as completed streams.

The shared tolerance row D and the tail-diameter row T are then downstream from the common window. Thus every consumer of the paired comparison receives the displayed route through $E_0, E_1, J, W_0, W_1, D, T$ before any real-name seal is available. □

Theorem 3015.4 (Cauchy tail-envelope comparison nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *packet receives only the two displayed* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows, their finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, the common-window refinement, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It receives no quotient stream, selected common limit, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-checking row.*

Proof. Project the carrier rows listed in Definition 3992.1. A consumer reaches the paired comparison only through $E_0, E_1, W_0, W_1, D, J, T, H, C, P, N$.

Apply Definition 3992.3. The common-window route places the two envelope rows, the finite refinement, the displayed windows, the shared tolerance, and the tail-diameter comparison before any terminal real-name seal can be read.

The excluded data are absent from the carrier and from the common-window route. Hence the comparison packet cannot export quotient streams, selected common limits, countable-choice schedules, trichotomy data, host real equality, ambient Real completeness, or a bridge-checking row to downstream Real or Completion consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ packet is generated from two displayed DyadicCauchyTailEnvelope rows, their StreamName tail windows, a shared DyadicRatCore tolerance row, a common-window refinement, a CauchyTailDiameter comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3992.1, Definition 3992.2, Definition 3992.3, and Definition 3992.4 seed the paired finite tail-envelope comparison route before either branch reaches Real or Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyTailEnvelopeComparisonUp.CauchyTailEnvelopeComparisonTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired envelope exposure, common-window admission, tolerance alignment, tail-diameter comparison, transport, replay, provenance, and nonescape for BEDC.Derived.CauchyTailEnvelopeCo

Chapter 3016

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompleteness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ packet records the constructive completeness readback surface for located regular real names as finite BEDC NameCert data. It sits after the compact interval neighbor of chapter 3986, the located interval surface of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite route by which a completion-facing consumer reads located Cauchy data; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, countable choice, selected limits, or open-cover compactness.

Definition 3016.1 (Bishop located completeness carrier)**.** *A* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, S, R, D, E, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, naming the located interval or located enclosure in which the inspected Cauchy data is read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite stream observations available at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling mesh, modulus, and endpoint precision requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed readback route.*
6. *B is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *boundary row, recording that any compact-interval assistance is consumed only through the finite route of the Bishop compact interval packet.*
7. *H records componentwise* hsame *transport for located interval, stream, regular, dyadic, Real, compact-boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from located requests through stream windows, regular readback, dyadic tolerance, Real sealing, compact-boundary control, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, R, D, E, B, H, C, P, N$. It has no coordinate for host real completeness, quotient representatives, selected Cauchy limits, countable-choice schedules, open-cover compactness, or an unrestricted locator oracle.

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletenessUp.BishopLocatedCompletenessTasteGate single carrier alignment`

Theorem 3016.2 (Bishop located completeness NameCert obligations)**.** *For every accepted* BishopLocatedComplete *carrier $L = (I, S, R, D, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed located interval, stream-window, regular readback, dyadic tolerance, Real seal, compact-boundary, transport, replay, provenance, and naming rows; located source admission through I; finite observation through S; regular readback through R; dyadic tolerance accounting through D; Real sealing through E; compact-interval boundary control through B; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation is discharged by ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, a selected limit, countable choice, open-cover compactness, or a host locator outside the displayed packet.*

Proof. Project the carrier of Definition 3993.1. The tuple lists the complete finite row inventory admitted by the local certificate.

The located source is read through I, then finite observation and regular readback pass through S and R. Dyadic precision is controlled by D before the Real seal E is exposed.

The compact-interval row B contributes only the finite boundary route supplied by chapter 3986. It cannot be read as open-cover compactness or as ambient completeness.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or a host locator, no local $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ obligation can export one. □

Theorem 3016.3 (Bishop located completeness Real-seal boundary)**.** *Every completion-facing read from an accepted* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BishopCompactInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The read exposes located interval data, finite stream windows, regular rational readback, dyadic tolerance control, compact-interval boundary control, Real sealing, transport, replay, provenance, and local naming, but it exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient real equality, selected limit, countable-choice schedule, or open-cover compactness principle.*

Proof. Let $L = (I, S, R, D, E, B, H, C, P, N)$ be an accepted carrier. A completion-facing consumer first receives I, S, R, D, so the located source, finite windows, regular readback, and dyadic tolerance route are fixed before any seal is visible.

The compact-boundary row B is then read as the Bishop compact interval neighbor of chapter 3986. It contributes only the finite interval boundary route accepted by that packet.

The Real-facing endpoint is E, and the structural route is H, C, P, N. These rows preserve the same finite path under transport, replay, provenance, and local naming. Because the classifier lacks coordinates for selected limits, quotient representatives, countable choice, open-cover compactness, and ambient completeness, the Real seal cannot be read as any of those stronger principles. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ completion-facing read passes through LocatedRealIntervalUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, BishopCompactIntervalUp, RealUp, transport, replay, provenance, and local naming rows before any Real-facing output is visible.

Independence witness: bishop-compact-interval, located-real-interval, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located source data, finite windows, regular rational readback, dyadic tolerance, compact-boundary control, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ packet is generated from a LocatedRealInterval source row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, a BishopCompactInterval boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3993.1, Definition 3993.2, and Definition 3993.3 seed the finite constructive located-completeness route over located interval, stream, regular, dyadic, compact-boundary, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient RealUp completeness, quotient real equality, selected limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source admission, finite observation, regular readback, dyadic tolerance accounting, Real sealing, compact-boundary control, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompletenessUp.

Chapter 3017

A Concrete Naming Certificate for $\mathsf{CauchyPrincipalFilter}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ packet records the principal-filter specialization used by Cauchy-filter and completion routes. It sits between the Cauchy-filter surface of chapter 333, the uniform-space entourage discipline of chapter 339, and the uniform-completion boundary of chapter 1473. The packet names the finite principal row that a completion consumer may inspect; it does not turn principal filters into quotient filters, selected limits, host equality, or ambient completeness.

Definition 3017.1 (CauchyPrincipalFilter carrier)**.** *A* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, G, F, U, B, H, K, P, N)$$

with the following displayed rows:

1. *A is the displayed center row naming the source point or finite address from which the principal filter is generated.*
2. *G is the neighbourhood and filter-base row: it lists the finite neighbourhood requests, the principal generator containing A, and the basis refinement witnesses accepted by the packet.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, recording that the principal filter request is read as a Cauchy-filter request only through the displayed generator and basis witnesses.*
4. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *boundary row exposing the entourages through which the principal request is tested.*
5. *B is the* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary row reached after A, G, F, and U have been displayed.*
6. *H records componentwise* hsame *transport for the center, generator, Cauchy-filter handoff, uniform entourage, and completion-boundary rows.*
7. *K records the displayed* Cont *replay from the center row through neighbourhood requests, filter-base refinement, Cauchy-filter handoff, uniform-space testing, and completion-boundary readback.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$ *row.*

The carrier compares two accepted packets only by replaying the displayed rows $A, G, F, U, B, H, K, P, N$. It has no coordinate for quotient-filter equality, selected limits, ultrafilter extension, host equality of completion points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyPrincipalFilterUp.CauchyPrincipalFilterTasteGate single carrier alignment`

Theorem 3017.2 (CauchyPrincipalFilter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier* $Q = (A, G, F, U, B, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$ *surface consists exactly of center admission through* A*, principal neighbourhood and filter-base admission through* G*, Cauchy-filter handoff through* F*, uniform-space testing through* U*, uniform-completion boundary exposure through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3994.1. The rows $A, G, F, U, B, H, K, P, N$ are the complete finite packet exposed to a consumer.

Read the principal-filter route in order. The center row A supplies the displayed generator, while G records the neighbourhood and filter-base witnesses through which the principal request is admitted.

Next follow the completion-facing handoff. The Cauchy-filter row F is available only after the generator and basis data have been displayed, and the uniform-space row U is the only entourage-testing surface used before a completion boundary is read.

Finally, B, H, K, P, and N supply boundary exposure, transport, replay, provenance, and local naming for the same rows. Since no coordinate stores quotient-filter equality, selected limits, ultrafilter extension, host equality, countable choice, or ambient completeness, no additional NameCert obligation can be exported from the packet. □

Theorem 3017.3 (CauchyPrincipalFilter convergence handoff)**.** *Every convergence-facing read exported by an accepted* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier first exposes the displayed center row* A*, the neighbourhood and filter-base row* G*, the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row* F*, and the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row* U *before it reaches the* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary row* B*. Thus the packet supplies a finite principal-filter route into Cauchy-filter and completion consumers without selecting a limit or quotienting filters.*

Proof. Start with a convergence-facing consumer. The carrier has exactly one source coordinate for the principal filter, namely the displayed center row A.

The neighbourhood row G then exposes the finite generator and basis refinement witnesses. A request that lacks those displayed witnesses has no route into the accepted packet.

The handoff row F places the request on the sibling $\mathsf{CauchyFilter}^{\uparrow}$ surface, and the uniform row U records the only entourages that the completion route may inspect. The boundary row B is therefore downstream of the finite principal-filter data rather than an independent completion object.

The transport, replay, provenance, and local naming rows H, K, P, N preserve only this route. Quotient filters, selected limits, host equality of completion points, countable choice, and ambient completeness remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ packet is generated from a displayed center row, neighbourhood and filter-base row, $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{UniformSpace}^{\uparrow}$ entourage boundary, $\mathsf{UniformCompletion}^{\uparrow}$ completion boundary, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyPrincipalFilter}\uparrow}$, seed)

Scope closed: Definition 3994.1, Definition 3994.2, and Definition 3994.3 seed the finite principal-filter route from a displayed center through Cauchy-filter and uniform-completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyPrincipalFilter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-filter equality, selected limits, ultrafilter extension, host equality of completion points, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, principal generator exposure, filter-base refinement, Cauchy-filter handoff, uniform entourage testing, completion-boundary exposure, replay, provenance, and non-escape for the same finite packet.

Chapter 3018

A Concrete Naming Certificate for $\mathsf{KochCurve}^{\uparrow}$

Following Koch's classical curve construction, the $\mathsf{KochCurve}^{\uparrow}$ packet records the finite polygonal-stage, dyadic-schedule, and uniform-limit handoff data for the standard Koch curve before any completed planar subset is imported. It sits beside the compact binary address discipline of chapter 1475, the uniform-limit surface of chapter 463, the terminal real-name seal of chapter 38, the dyadic tolerance ledger of chapter 291, and the compact-uniform-continuity surface of chapter 388. The packet names only finite stage words, segment replacements, coordinate schedules, tolerance ledgers, and the displayed uniform-limit route; it does not import a host fractal subset, a selected infinite branch, quotient equality of real pairs, ambient compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3018.1 (Koch curve carrier)**.** *A* $\mathsf{KochCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, S, D, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite address row. It records the inspected ternary stage, the finite left-turn and right-turn segment word, and the CantorSpace address window used by branch-facing consumers.*
2. *S is the polygonal-stage row, naming the directed segment list obtained by replacing each displayed segment by the four Koch generator segments at the next stage.*
3. *D is the dyadic displacement ledger. It records the rational endpoint deltas and the dyadic tolerance comparisons used to read the finite segment list.*
4. *W is the coordinate* $\mathsf{StreamName}^{\uparrow}$ *window row for the horizontal and vertical endpoint schedules attached to the inspected segment.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those coordinate schedules, with all tolerance comparisons routed through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
6. *E is the pair of terminal* $\mathsf{Real}^{\uparrow}$ *coordinate seals read from the same scheduled polygonal-stage rows.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, binding the stagewise polygonal approximations, dyadic tolerance ledger, and coordinate seals to one uniform-limit route.*

8. *H records componentwise* hsame *transport for addresses, segment stages, displacement ledgers, coordinate windows, rational readbacks, real seals, uniform-limit handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each curve-facing request through $A, S, D, W, Q, E, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{KochCurve}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, D, W, Q, E, L, H, C, P, N$. It has no coordinate for a completed planar fractal, selected infinite branch, quotient equality of real pairs, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.KochCurveUp.KochCurveTasteGate single carrier alignment`

Theorem 3018.2 (Koch curve NameCert obligations)**.** *For every accepted* $\mathsf{KochCurve}^{\uparrow}$ *carrier*

$$K = (A, S, D, W, Q, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; finite address exposure through A; polygonal-stage exposure through S; dyadic displacement and tolerance exposure through D; coordinate-window exposure through W; regular rational readback through Q; terminal real coordinate sealing through E; uniform-limit handoff through L; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. No obligation is discharged by a host planar powerset, selected infinite address, quotient real-pair equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3995.1. The accepted packet displays exactly the finite address row, polygonal-stage row, displacement ledger, coordinate windows, rational readbacks, real seals, uniform-limit handoff, and structural rows.

The finite combinatorial obligations are exhausted by A and S. A consumer cannot inspect a curve branch until the stage address is displayed, and it cannot read a segment replacement except through the directed segment list attached to that same finite stage.

The coordinate obligations are exhausted by D, W, Q, E. The dyadic ledger gives endpoint deltas and tolerance comparisons, the coordinate windows schedule the endpoint reads, the regular rational row reads those schedules, and the $\mathsf{Real}^{\uparrow}$ seals remain downstream of the same polygonal-stage data.

Finally L, H, C, P, N transport, replay, source, and name only the displayed route. Since the excluded host principles have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ obligations. □

Theorem 3018.3 (Koch curve uniform-limit handoff)**.** *Every accepted* $\mathsf{KochCurve}^{\uparrow}$ *consumer that reaches a curve point, segment bound, or real-coordinate read must first expose the finite route*

$$\mathsf{CantorSpace}^{\uparrow} \longmapsto A, S \longmapsto D, W, Q, E \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto H, C, P, N.$$

The handoff exports only the displayed polygonal-stage approximations, dyadic tolerance ledger, scheduled coordinate readbacks, terminal $\mathsf{Real}^{\uparrow}$ *coordinate seals, and uniform-limit route. It exports no completed fractal subset, selected infinite branch, quotient equality of real coordinate pairs, ambient compactness theorem, or ambient completeness principle.*

Proof. Begin with a consumer that reaches the Real-facing boundary of the packet. By Definition 3995.1, the only terminal coordinate seals are downstream of the finite address, polygonal-stage, dyadic displacement, coordinate-window, and rational-readback rows.

Apply Definition 3995.2. It forces every curve-facing read through $A, S, D, W, Q, E, L, H, C, P, N$ and keeps the uniform-limit handoff on the same displayed packet.

The compact-uniform-continuity and uniform-limit dependencies supply only the finite modulus and convergence route named by L. They do not add a second carrier coordinate for a host planar subset, a selected infinite address, quotient equality of real coordinate pairs, or ambient completeness. Therefore every point, segment, or coordinate consumer remains inside the finite NameCert packet. □

Falsifiable prediction: An accepted $\mathsf{KochCurve}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ coordinate seal or a uniform-limit read without first displaying the finite address row, polygonal-stage row, dyadic displacement ledger, coordinate window, and rational readback refutes the carrier surface.

Independence witness: CantorSpace, UniformLimit, Real, DyadicRatCore, and CompactUniformContinuity: no listed sibling supplies the whole $\mathsf{KochCurve}^{\uparrow}$ packet, because each sibling contributes address windows, convergence handoff, terminal coordinate seals, dyadic tolerance data, or modulus discipline while this chapter binds all rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KochCurve}^{\uparrow}$ packet is generated from finite address rows, polygonal-stage segment rows, dyadic displacement ledgers, coordinate StreamName windows, RegSeqRat readbacks, RealUp coordinate seals, a UniformLimit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$, seed)

Scope closed: Definition 3995.1, Definition 3995.2, and Definition 3995.3 seed the finite polygonal-stage and uniform-limit certificate surface for the classical Koch curve.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host planar powersets, selected infinite addresses, quotient equality of real coordinate pairs, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite address exposure, polygonal-stage exposure, dyadic displacement exactness, coordinate readback, Real coordinate sealing, uniform-limit handoff, transport, replay, provenance, and local naming for BEDC.Derived.KochCurveUp.

Chapter 3019

A Concrete Naming Certificate for $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$

The $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ packet realizes the channel boundary of chapter 9369 as finite BEDC NameCert data. It records source-event rows, legal channel-event rows, terminator-position rows, decoder-state rows, halting-boundary refusal rows, trace rows, transport, replay, provenance, and local naming in one displayed packet. The packet reads local event termination through the bounded evaluator and finite trace rows of chapter 818 before refusing certificate-level halting. The point of the packet is the stratum separation: a legal channel event can end locally, while certificate-level halting remains a refusal boundary rather than a global total classifier.

Definition 3019.1 (ChannelTerminationHaltingBoundary carrier)**.** *A* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, E, B, D, L, R, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-event row, naming the inspected finite binary source word and the source-layer reading of its glyphs.*
2. *E is the legal channel-event row, recording the encoded body over fragments* 0 *and* 10 *together with the displayed channel terminator* 11*.*
3. *B is the body-safety row, recording that the encoded body contains no adjacent* 11 *before the terminator row is exposed.*
4. *D is the decoder-state row, naming the left-to-right scan, the first displayed* 11*, and the recovered source event.*
5. *L is the legal-channel-event classifier row, accepting only event reads that pass through S, E, B, D before any boundary claim is made.*
6. *R is the halting-boundary refusal row. It records that local event termination is not a total classifier for arbitrary* NameCert *continuation chains.*
7. *F is the finite trace row, carrying the prompt-response and bounded evaluator evidence used to display the refusal without solving certificate-level halting.*
8. *H records componentwise* hsame *transport for source events, legal channel events, body safety, decoder state, classifier exposure, refusal, trace evidence, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from each boundary-facing request through* $S, E, B, D, L, R, F, H, P, N$.
10. *P is the* Pkg *provenance over* ChannelTerminationHaltingBoundary$^{\uparrow}$, *haltingdistinction, haltingdistinctiontrace, promptresponsetrace, finitetraceevaluator,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ChannelTerminationHaltingBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, B, D, L, R, F, H, C, P, N$. *It has no coordinate for a bare bitstream parser, a host global halting oracle, an unrestricted evaluator, a certificate-of-certificates total classifier, or a meta-stop proof.*

Lean4 ✓ `BEDC.Derived.ChannelTerminationHaltingBoundaryUp.ChannelTerminationHaltingBoundaryTasteGate single carrier alignment`

Theorem 3019.2 (ChannelTerminationHaltingBoundary NameCert obligations)**.** *For every accepted* ChannelTerminationHaltingBoundary$^{\uparrow}$ *carrier*

$$T = (S, E, B, D, L, R, F, H, C, P, N),$$

the local NameCert$_{\text{ChannelTerminationHaltingBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; source-event exposure through S; legal channel-event exposure through E; body-safety exposure through B; decoder-state exposure through D; legal-channel-event classification through L; halting-boundary refusal through R; finite trace evidence through F; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation is discharged by a bare bitstream parser, a host global halting oracle, an unrestricted evaluator, a total certificate-level halting classifier, or a meta-stop proof.

Proof. Project the carrier of Definition 3996.1. The accepted packet displays exactly the source event, legal channel event, body-safety, decoder-state, classifier, refusal, trace, and structural rows.

The local channel obligations are exhausted by S, E, B, D, L. The source row names the event being encoded, the legal channel-event row displays the body and terminator, the body-safety row excludes an earlier terminator pattern, and the decoder-state row reads the first 11 as the local endpoint.

The halting-boundary obligation is exhausted by R and F. The refusal row states that the local endpoint is not a certificate-level total halting classifier, while the finite trace row records only bounded evidence around prompt-response and evaluator routes.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for a global halting oracle or meta-stop proof, no local NameCert$_{\text{ChannelTerminationHaltingBoundary}^{\uparrow}}$ obligation can export one. □

Theorem 3019.3 (ChannelTerminationHaltingBoundary refusal factorization)**.** *Every accepted* ChannelTerminationHaltingBoundary$^{\uparrow}$ *packet factors local event termination through the displayed legal-channel-event route*

$$S, E, B, D, L \longmapsto R, F, H, C, P, N$$

before any halting-boundary refusal row is exposed. Thus a consumer may read the first channel-level 11 *as a local event endpoint only after the legal body and decoder rows have been displayed, and it may read the halting-boundary refusal only as the downstream statement that this local endpoint is not a certificate-level total halting classifier.*

Proof. Start with an accepted boundary-facing consumer. By Definition 3996.1, the only event-termination coordinates are S, E, B, D, L, so the consumer cannot reach the refusal row until the legal channel-event route is already displayed.

Apply Definition 3996.2. It binds body safety, first-terminator decoding, and legal-channel classification to the same finite packet before R and F can be read.

The refusal row then has only one role: it separates the local event endpoint from certificate-level halting. The finite trace row may record bounded prompt-response or evaluator evidence, but the packet has no coordinate for an unrestricted evaluator, a host global halting oracle, or a meta-stop proof. Therefore the refusal factors through the legal-channel-event route rather than replacing it. □

Falsifiable prediction: Every accepted packet factors local event termination through a displayed legal-channel-event row before any halting-boundary refusal row is exposed.

Independence witness: haltingdistinction, haltingdistinctiontrace, promptresponsetrace, finite-traceevaluator: these siblings either classify halting obstruction or trace evaluation, while this packet has the additional legal-channel terminator row and therefore is not bijective to any of them.

The Lean TasteGate namespace BEDC.Derived.ChannelTerminationHaltingBoundaryUp.TasteGate must provide Nontrivial and FieldFaithful instances for this finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ packet is generated from source-event rows, legal channel-event rows, body-safety rows, decoder-state rows, legal-channel classifiers, halting-boundary refusal rows, finite trace evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3996.1, Definition 3996.2, and Definition 3996.3 seed the finite local-channel boundary separating event termination from certificate-level halting refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted halting classification, host global halting oracles, bare bitstream parsing, meta-stop proofs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal channel-event exposure, body-safety exposure, decoder-state exposure, local termination factorization, halting-boundary refusal, finite trace evidence, transport, replay, provenance, and local naming for BEDC.Derived.ChannelTerminationHaltingBoundaryUp.

Chapter 3020

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$

Following Bishop and Bridges on constructive completion and reflector discipline, the CauchyCompletionUniversalArr packet records the universal arrow of the L10 Cauchy completion route as finite NameCert data. It ties the reflector surface of chapter 1585, the regular unit of chapter 3646, the counit of chapter 1539, the idempotent comparison of **??**, and the metric universal property of chapter 2124 to the regular rational and real seals of chapter 301 and chapter 38. The packet names the universal-arrow boundary without quotient completions, selected limits, host function extensionality, ambient completeness, or bridge checking.

Definition 3020.1 (CauchyCompletionUniversalArrow carrier)**.** *A* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row opened by the universal-arrow request.*
2. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row naming the completion reflector over M.*
3. *η is the* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *row exposing the source-to-completion unit.*
4. *F is the uniformly continuous map row from the source metric into the displayed complete target.*
5. *$\widehat{F}$ is the factor arrow row from the reflected completion to the target named by F.*
6. *Q is the uniqueness comparison row, requiring any competing factor arrow with the same precomposition along η to compare through the displayed NameCert route.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row reached after $R, \eta, F, \widehat{F}, Q$ have been displayed.*
8. *H records componentwise* hsame *transport for source metrics, reflector rows, units, uniformly continuous maps, factor arrows, uniqueness comparisons, seal rows, replay rows, provenance rows, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M, R, \eta, F \longmapsto \widehat{F}, Q, S.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, MetricCompletionUniv $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completions, selected Cauchy limits, host function extensionality, ambient completeness of a target space, or bridge-checked equality.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalArrowUp.CauchyCompletionUniversalArrowTasteGate`
`single carrier alignment`

Theorem 3020.2 (CauchyCompletionUniversalArrow NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ *carrier* $U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUni}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the source metric, completion reflector, unit, uniformly continuous map, factor arrow, uniqueness comparison, RegSeqRat/Real seal, transport, replay, provenance, and local naming rows;*
2. *reflector anchoring, requiring the rows R and η to be displayed before any universal-arrow factor can be inspected;*
3. *map compatibility, requiring the source map row F to be read as uniformly continuous along the same source metric row M;*
4. *factorization visibility, requiring $\widehat{F}$ to be exported only as the factor arrow whose precomposition along η replays F through C;*
5. *uniqueness comparison, requiring Q to compare all competing factor rows by the same displayed route rather than by host function extensionality;*
6. *RegSeqRat/Real seal nonescape, requiring S, H, C, P, N to exhaust the regular rational seal, real seal, transport, replay, provenance, and local naming surface.*

Proof. Project the accepted carrier $U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$. Carrier admission exposes exactly the rows listed in Definition 3997.1.

The universal-arrow request is anchored by M, R, η. The source metric is visible before the reflector row is consumed, and the unit row is the only source-to-completion boundary available to the factorization read.

The map and factor rows are then read together. The row F supplies the uniformly continuous source map, while $\widehat{F}$ is exported only through the continuation replay that identifies precomposition with η as the displayed source map route.

Finally Q, S, H, C, P, N close the comparison and sealing surface. Since quotient completions, selected limits, host function extensionality, ambient completeness, and bridge checking are absent from the carrier, the listed rows exhaust the local certificate. □

Theorem 3020.3 (CauchyCompletionUniversalArrow factorization)**.** *Every accepted* CauchyCompletionUniversalA *packet exports a universal-arrow factorization route*

$$M \longmapsto \mathsf{CauchyCompletionReflector}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUnit}^{\uparrow} \longmapsto F \longmapsto \widehat{F} \longmapsto Q \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

Any downstream completion consumer receives only the source metric row, reflector row, unit row, uniformly continuous map row, factor arrow row, uniqueness comparison row, RegSeqRat/Real seal

row, componentwise transport, displayed replay, provenance, and local naming. It receives no quotient completion object, selected limit, host function extensionality principle, ambient completeness assertion, or bridge-checked equality.

Proof. Open the carrier and read M, R, η first. These rows display the source metric, the Cauchy completion reflector, and the regular unit that a universal-arrow consumer may inspect.

Next read F and $\widehat{F}$. The factor arrow is not an ambient function-space object; it is the displayed row whose continuation replay agrees with F after the unit row.

The uniqueness comparison row Q then compares any other factor row through the same displayed precomposition route. This is a certificate comparison, not a use of host function extensionality.

The route terminates at S, H, C, P, N. Regular rational and Real sealing, transport, replay, provenance, and local naming preserve the ordered handoff and exclude quotient completions, selected limits, ambient completeness, and bridge checking from the exported packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ packet is generated from a source metric row, a CauchyCompletionReflector row, a regular completion unit row, a uniformly continuous map row, a factor arrow row, a uniqueness comparison row, RegSeqRat and Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3997.1, Definition 3997.2, and Definition 3997.3 seed the finite universal-arrow route for the L10 Cauchy completion reflector.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host function extensionality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector anchoring, unit exposure, map compatibility, factorization visibility, uniqueness comparison, RegSeqRat/Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionUniversalArrowUp.

Chapter 3021

A Concrete Naming Certificate for $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$

Following Wolfram's cellular-automaton carrier reading and Cook's Rule 110 universality result, the $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ packet records the carrier-equivalence content of chapter 9391 as finite NameCert data. It sits beside the automaton surface of chapter 934, the orbit substrate of chapter 938, the pattern catalog of chapter 995, the trust substrate of chapter 940, and the substrate classifier boundary of chapter 3810. The packet names a finite BHist encoding and orbit-window comparison without replacing the BEDC carrier.

Definition 3021.1 (CellularCarrierBHistEquivalence carrier)**.** *A* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, B, D, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the cellular orbit-window row: a finite initial row, width boundary, time window, and local update discipline.*
2. *B is the BHist encoding row that reads a finite* BHist *value as a finite cell-row code.*
3. *D is the decoder row that reads the same finite cell-row code back as a* BHist *value when the orbit window remains inside the accepted boundary.*
4. *R is the roundtrip row, requiring the encode-decode and decode-encode reads to agree on the displayed finite window.*
5. *K is the classifier transport row moving accepted cellular observations to the corresponding* BHist *classifier observations and back.*
6. *H records componentwise* hsame *transport for orbit windows, encodings, decoders, roundtrip rows, classifier transport rows, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$W, B, D, R \longmapsto K.$$

8. *P is the* Pkg *provenance over* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularOrbitSubstrate}^{\uparrow}$, $\mathsf{CellularPatternCatalog}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these nine rows. It has no coordinate for a replacement BEDC kernel carrier, infinite cellular orbits, global physical substrate claims, hidden evaluator state, host bisimulation, or non-finite classifier equality.

Lean4 ✓ `BEDC.Derived.CellularCarrierBHistEquivalenceUp.CellularCarrierBHistEquivalenceTasteGate`
`single carrier alignment`

Theorem 3021.2 (CellularCarrierBHistEquivalence NameCert obligations)**.** *For every accepted* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ *carrier* $E = (W, B, D, R, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalenc}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for orbit-window, BHist encoding, decoder, roundtrip, classifier transport, transport, replay, provenance, and local naming rows;*
2. *orbit-window finiteness, requiring all cellular reads to remain inside the displayed width and time window;*
3. *encoding and decoding visibility, requiring B and D to be inspected before any carrier-equivalence claim is exported;*
4. *roundtrip and classifier transport, requiring R and K to bind finite BHist reads and finite cellular-window reads through the same accepted packet;*
5. *non-escape from carrier replacement, infinite orbit semantics, physical substrate claims, hidden evaluator state, host bisimulation, and non-finite classifier equality.*

Proof. Carrier admission is projection from Definition 3998.1. The tuple displays exactly the orbit-window, encoding, decoder, roundtrip, classifier transport, transport, replay, provenance, and naming rows.

The cellular side is finite before any equivalence read begins. The row W fixes the initial row, width boundary, time window, and local update discipline inspected by the packet.

The BHist side is read through B and D. These rows expose encoding and decoding as finite carrier reads rather than as a replacement of the BEDC kernel carrier.

The roundtrip and classifier rows R and K then bind the two directions over the same displayed window. Transport, replay, provenance, and local naming preserve that binding, while the absent coordinates rule out infinite orbit semantics, physical substrate claims, host bisimulation, and non-finite classifier equality. □

Theorem 3021.3 (Cellular carrier BHist equivalence route)**.** *Every accepted* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{CellularOrbitSubstrate}^{\uparrow} \longmapsto W \longmapsto B \longmapsto D \longmapsto R \longmapsto K \longmapsto H, C, P, N.$$

The route realizes the cellular-carrier alternative as a finite comparison between orbit-window observations and BHist *observations. It does not export a new kernel carrier, an infinite orbit semantics, a global physical substrate assertion, hidden evaluator state, host bisimulation, or non-finite classifier equality.*

Proof. Open W first. The orbit-window row supplies the finite cellular observation boundary and therefore prevents the route from depending on an infinite orbit.

Read B and D next. Encoding and decoding connect the finite window code with a finite BHist value, so the comparison is a NameCert packet rather than a carrier substitution.

Use R and K to bind the roundtrip and classifier transport. These rows keep cellular and BHist observations in the same finite acceptance boundary.

Finally consume H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the finite route and leave no carrier-replacement, physical-substrate, hidden-evaluator, host-bisimulation, or non-finite-equality coordinate to export. □

Falsifiable prediction: Every accepted $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ read factors through a finite orbit-window row, a BHist encoding row, a decoder row, a roundtrip row, classifier transport, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: cellularautomaton, cellularorbitsubstrate, cellularpatterncatalog, cellular-trust-substrate, cellular-substrate-classifier-boundary: $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ is not bijective to any listed sibling because it jointly packages a finite orbit window, BHist encoding, decoding, roundtrip, classifier transport, replay, provenance, and local naming as one carrier-equivalence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ packet is generated from a finite cellular orbit-window row, a BHist encoding row, a decoder row, a roundtrip row, a classifier transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3998.1, Definition 3998.2, and Definition 3998.3 seed the finite carrier-equivalence route between cellular orbit-window observations and BHist observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close carrier replacement, infinite cellular orbit semantics, global physical substrate claims, hidden evaluator state, host bisimulation, non-finite classifier equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-window finiteness, encoding and decoding visibility, roundtrip comparison, classifier transport, transport, replay, provenance, and local naming for BEDC.Derived.CellularCarrierBHistEquivalenceUp.

Chapter 3022

A Concrete Naming Certificate for $\mathsf{RegularSequenceBishop}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularSequenceBishop}^{\uparrow}$ packet records the regular-sequence presentation of a constructive real as finite NameCert data. It reads finite observations through chapter 126, measures tolerances through chapter 291, passes rational readback through chapter 301, remains compatible with the Cauchy surface of chapter 3413, and reaches the terminal seal of chapter 38. The packet is induction-facing: it exposes bounded finite windows and dyadic tolerance ledgers before any $\mathsf{Real}^{\uparrow}$ seal is consumed, and it does not quotient streams, select a host limit, invoke countable choice, or assume ambient Real completeness.

Definition 3022.1 (Bishop regular-sequence carrier)**.** *A* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W, D, Q, R, N)$$

with the following five displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the bounded stream segment inspected at a requested precision.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the positive dyadic scale, comparison cells, and slack ledger used to compare the window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, turning the inspected window and tolerance ledger into regular rational sequence evidence.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after W, D, Q have been displayed.*
5. *N is the local structural row, bundling componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$ *naming row for the same packet.*

The classifier compares accepted packets only through W, D, Q, R, N. It has no coordinate for quotient stream equality, host real equality, a selected limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularSequenceBishopUp.RegularSequenceBishopTasteGate single carrier alignment`

Theorem 3022.2 (Bishop regular-sequence NameCert obligations)**.** *For every accepted* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier $B = (W, D, Q, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$ *surface consists exactly of five*

obligation rows: finite $\mathsf{StreamName}^{\uparrow}$ *window exposure through* W*; dyadic tolerance and comparison exposure through* D*;* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback through* Q*; downstream* $\mathsf{Real}^{\uparrow}$ *seal handoff through* R*; and local transport, replay, provenance, and naming through* N*. Every Real and Completion consumer must read*

$$W, D \longmapsto Q \longmapsto R \longmapsto N$$

before treating the packet as a Bishop-style regular sequence.

Proof. Project the accepted carrier of Definition 3999.1. The visible coordinates are exactly W, D, Q, R, N, and the classifier contains no further row.

The window and tolerance part is read first. The row W supplies only a finite $\mathsf{StreamName}^{\uparrow}$ observation, while D supplies the dyadic precision and comparison ledger attached to that observation.

The regular rational handoff is then forced through Q. Since Q is the only row that converts the window and tolerance data into $\mathsf{RegSeqRat}^{\uparrow}$ evidence, no host stream quotient or selected limit can enter the local surface.

Finally R and N deliver the Real-facing boundary and the structural NameCert support for the same displayed route. The absence of coordinates for quotient equality, host real equality, countable choice, trichotomy, or ambient completeness gives the stated exhaustiveness. □

Theorem 3022.3 (Bishop regular-sequence induction window)**.** *Let* $B = (W, D, Q, R, N)$ *be an accepted* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier. Any induction over the regular-sequence presentation may inspect only finite* $\mathsf{StreamName}^{\uparrow}$ *windows from* W*, compare them through the dyadic ledger* D*, and pass the resulting rational data through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *before the* $\mathsf{Real}^{\uparrow}$ *seal row* R *is consumed. The induction route is therefore*

$$W, D \longmapsto Q \longmapsto R \longmapsto N.$$

It does not decide quotient equality of streams, choose a host limit, use countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier projection in Definition 3999.1. An induction step can open the finite window row W and the tolerance row D, because those are the only displayed sources available before any rational readback.

The step then passes through Q. This row is the sole $\mathsf{RegSeqRat}^{\uparrow}$ readback coordinate, so every induction witness remains tied to finite windows and dyadic tolerances rather than to quotient representatives or chosen limits.

Only after that readback may the route consume R, and N preserves the same route by transport, replay, provenance, and local naming. Thus the induction window is finite and Bishop-style, with the excluded nonconstructive and completeness coordinates absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{RegularSequenceBishop}^{\uparrow}$ read used by a Real and Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and one local structural NameCert row before it is treated as a regular-sequence presentation.

Independence witness: streamname, dyadicratcore, regseqrat, cauchy, real: $\mathsf{RegularSequenceBishop}^{\uparrow}$ is not bijective to any listed sibling because it binds finite stream windows, dyadic tolerance comparisons, regular rational readback, a Real-facing seal, and local NameCert structure into one induction-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceBishop}^{\uparrow}$ packet is generated from a StreamName finite-window row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3999.1, Definition 3999.2, and Definition 3999.3 seed the Bishop-style regular-sequence route over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host real equality, selected limits, countable choice, trichotomy, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, RegSeqRat readback, Real seal ordering, transport, replay, provenance, local naming, and induction-window nonescape for BEDC.Derived.RegularSequenceBishopUp.

Chapter 3023

A Concrete Naming Certificate for $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ packet records a uniform Cauchy modulus for a family of real-valued approximants as finite NameCert data. It sits after the finite source windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the criterion surface of chapter 3876, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer boundary. The packet gives Real and Completion routes a finite uniform-Cauchy bridge; it does not assert host function equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3023.1 (Modulus of uniform convergence carrier)**.** *A* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, F, U, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, naming the finite domain observations inspected for each requested uniform tolerance.*
2. *F is the function-family row, recording the displayed approximant indices and the finite value reads used in the comparison.*
3. *U is the uniform tolerance and modulus row, exposing the requested precision and the threshold delivered to the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer surface.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic comparison cells, positivity witnesses, and slack data used by the uniform estimate.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows, family reads, tolerance ledger, and modulus threshold.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, F, U, D, R have been displayed.*
7. *H records componentwise* hsame *transport for source windows, family reads, tolerance data, readback, seal, continuation, provenance, and naming.*
8. *C records displayed* Cont *replay from a uniform-Cauchy request through $S, F, U, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, U, D, R, E, H, C, P, N$. It has no coordinate for host equality of functions, a countable-choice schedule, a selected limiting function, quotient equality of real names, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.ModulusOfUniformConvergenceUp.ModulusOfUniformConvergenceUp`

Theorem 3023.2 (Modulus of uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ *carrier*

$$M = (S, F, U, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$ *surface consists exactly of source-window exposure through S, function-family read exposure through F, uniform tolerance and modulus exposure through U, dyadic tolerance-ledger exposure through D, regular rational readback through R, real seal handoff through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. Every Real or Completion consumer must read*

$$S, F, U, D \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before treating the packet as a uniform convergence modulus.

Proof. Project the carrier of Definition 4000.1. The visible coordinates are exactly $S, F, U, D, R, E, H, C, P, N$, and the classifier contains no further analytic row.

The source and family data are read first. The row S supplies finite $\mathsf{StreamName}^{\uparrow}$ windows, while F supplies the displayed approximant reads attached to those windows.

The uniform estimate is then forced through U and D. The row U names the common threshold and tolerance request, and D supplies the dyadic ledger that makes the comparison finite.

Finally R, E, H, C, P, N deliver regular rational readback, the real-facing seal, transport, replay, provenance, and local naming for the same displayed route. Since host function equality, countable choice, selected limits, quotient real-name equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, the listed obligations exhaust the certificate surface. □

Theorem 3023.3 (Modulus of uniform convergence Cauchy-uniform route)**.** *Every accepted* ModulusOfUniformCon *packet consumed by* $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ F,\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes the same rows $S, F, U, D, R, E, H, C, P, N$ and refuses host function equality, countable choice, selected limits, quotient equality of real names, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier. A criterion or modulus consumer first receives the source-window row S and the family-read row F, so the request is grounded in finite observations rather than in a host function object.

Next read the uniform modulus row U together with the dyadic ledger D. These rows give a single threshold and tolerance comparison surface for all displayed windows, which is precisely the finite boundary expected by $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ and $\mathsf{UniformModulus}^{\uparrow}$.

The only downstream analytic readback is R followed by E. Thus $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive data only after the uniform-Cauchy route has been displayed.

The transport, replay, provenance, and naming rows H, C, P, N preserve the same route. No excluded host equality, choice schedule, selected limit, quotient equality, or completeness coordinate can be inserted by a consumer of the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ read used by a Real or Completion consumer displays one $\mathsf{StreamName}^{\uparrow}$ source-window row, one function-family read row, one $\mathsf{UniformModulus}^{\uparrow}$ tolerance-threshold row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N before it is treated as a uniform convergence modulus.

Independence witness: streamname, dyadicratcore, regseqrat, real, uniformmodulus, uniform-convergence-cauchy-criterion: $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ is not bijective to any listed sibling because only it binds finite source windows, displayed family reads, one uniform threshold, a dyadic tolerance ledger, regular rational readback, a real seal, and local NameCert support into one uniform-Cauchy bridge.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ packet is generated from StreamName source windows, function-family read rows, a UniformModulus tolerance-threshold row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4000.1, Definition 4000.2, and Definition 4000.3 seed the finite uniform-Cauchy bridge over StreamName, DyadicRatCore, RegSeqRat, Real, UniformConvergenceCauchyCriterion, and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, countable choice, selected limits, quotient equality of real names, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window exposure, function-family read exposure, uniform threshold exposure, dyadic tolerance-ledger exposure, readback, real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfUniformConvergenceUp.

Chapter 3024

A Concrete Naming Certificate for $\mathsf{DyadicBisectionSchedule}^{\uparrow}$

Following Bishop and Bridges constructive nested-interval and bisection arguments, the $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ packet records the finite halving schedule used before a retained interval route is read as Cauchy data. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located interval-zero route of chapter 3971, and the retained closed-interval branch surface of chapter 3815. The packet names a finite schedule object only; it does not select an infinite branch, assert an interval intersection point, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3024.1 (Dyadic bisection schedule carrier)**.** *A* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, B, D, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed interval source row. It exposes endpoint order, containment, and the finite retained-window requests consumed from the* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *route.*
2. *B is the branch schedule row. It records the inspected finite sequence of left-or-right bisection decisions, midpoint exposure, retained-child containment, and width-halving ledger.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row. It records the positive dyadic mesh, endpoint comparison cells, midpoint arithmetic cells, and halving bounds used by B.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for endpoint, midpoint, retained-child, and schedule-prefix observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after I, B, D, W have been displayed.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, opened only after the finite schedule and regular readback rows are exposed.*
7. *H records componentwise* hsame *transport for the interval source, branch schedule, dyadic scale, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located interval or Cauchy request through $I, B, D, W, R, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$, $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedIntervalBolzano}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$ *row.*
The classifier compares accepted schedule carriers only through $I, B, D, W, R, Q, H, C, P, N$. It has no coordinate for an infinite choice sequence, a completed nested intersection, quotient real equality, host trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionScheduleUp.DyadicBisectionScheduleTasteGate single carrier alignment`

Theorem 3024.2 (Dyadic bisection schedule NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier $S = (I, B, D, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$ *surface consists exactly of carrier admission, closed-interval source exposure, finite branch-schedule exposure, dyadic halving exactness, stream-window exposure, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and local naming for the carrier of Definition 4001.1.*

Proof. Carrier admission is projection from Definition 4001.1. The interval row I and branch row B are the only rows that expose closed-interval bisection data, so every schedule obligation starts with the displayed finite interval source and retained-child ledger.

The dyadic and stream obligations are consecutive. The row D supplies midpoint arithmetic, positive mesh data, and halving bounds before W can open endpoint, midpoint, retained-child, or prefix observations.

The readback rows R, Q are therefore downstream of I, B, D, W. The structural rows H, C, P, N transport, replay, source, and name the same finite route.

Since the carrier contains no coordinate for infinite branch choice, completed intersection points, quotient real equality, trichotomy, open-cover compactness, or ambient completeness, the listed obligations exhaust the local certificate. □

Theorem 3024.3 (Dyadic bisection schedule Cauchy handoff)**.** *Every Cauchy- or Bolzano-facing consumer of an accepted* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier reaches regular rational data only through the displayed route*

$$\mathsf{ClosedIntervalBisection}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may be consumed by located interval and completion routes as a finite halving schedule, but it exports no selected infinite branch, completed intersection point, quotient real representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $S = (I, B, D, W, R, Q, H, C, P, N)$. The source row I and the retained branch schedule B expose the only finite bisection information available to the consumer.

Read the dyadic row D next. It fixes the midpoint and width-halving budget through displayed dyadic cells, so the schedule cannot be interpreted as an external sequence of arbitrary cuts.

The stream and regular rows W, R then turn the displayed finite windows into regular rational readback. The terminal row Q is a Real-facing seal only after this finite route has been read.

Finally H, C, P, N preserve the same order under transport, replay, provenance, and local naming. Because no stronger object appears among the carrier coordinates, every Cauchy handoff remains the displayed finite schedule route. □

Falsifiable prediction: A Cauchy or Bolzano consumer that omits the branch schedule row, dyadic halving row, StreamName window row, or RegSeqRat readback row cannot claim a $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ handoff to a Real-facing seal.

Independence witness: closed-interval-bisection, located-interval-bolzano, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval source, finite branch schedule, dyadic halving ledger, stream windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ packet is generated from a closed-interval source row, a finite retained-branch schedule row, a DyadicRatCore halving ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4001.1, Definition 4001.2, and Definition 4001.3 seed the finite dyadic interval-halving route over ClosedIntervalBisectionUp, LocatedIntervalBolzanoUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, completed interval intersections, quotient real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, finite branch-schedule exposure, dyadic halving exactness, StreamName window exactness, RegSeqRat readback, Real-seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.DyadicBisectionScheduleUp.

Chapter 3025

A Concrete Naming Certificate for $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ packet records a finite enclosure-filter route for regular Cauchy names. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy-filter basis surface of chapter 333. The packet gives Real and Completion consumers a displayed enclosure-filter certificate without quotienting streams, selecting a host limit, assuming ambient completeness, or identifying Cauchy filters by external equality.

Definition 3025.1 (Regular Cauchy enclosure-filter carrier)**.** *A* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, D, Q, F, B, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected rational-name segment and the requested precision window.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure ledger, recording the positive dyadic radius, interval endpoints, midpoint comparison, and slack data used by the window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained only after W and D have been displayed.*
4. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, listing the finite filter requests and refinements whose accepted members contain the same dyadic enclosures.*
5. *B is the enclosure-basis compatibility row, matching each displayed regular Cauchy window with a filter-basis request at the same dyadic radius.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, D, Q, F, B have been consumed.*
7. *H records componentwise* hsame *transport for the window, dyadic enclosure, regular readback, Cauchy-filter basis, compatibility, and Real-seal rows.*
8. *C records displayed* Cont *replay from a finite window and radius request through regular readback and filter-basis compatibility to the Real-facing seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, D, Q, F, B, R, H, C, P, N$. It has no coordinate for quotient stream equality, quotient filter equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEnclosureFilterUp.RegularCauchyEnclosureFilterTasteGate`
`single carrier alignment`

Theorem 3025.2 (Regular Cauchy enclosure-filter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ *carrier $E = (W, D, Q, F, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}}$ *surface consists exactly of finite window exposure, dyadic enclosure admission, regular rational readback, Cauchy-filter basis admission, enclosure-basis compatibility, Real-seal ordering, componentwise transport, continuation replay, provenance, and local naming. Every consumer must read the finite route*

$$W, D \longmapsto Q \longmapsto F, B \longmapsto R$$

together with H, C, P, N before treating the packet as an enclosure-filter certificate for a regular Cauchy name.

Proof. Project the carrier of Definition 4003.1. The rows $W, D, Q, F, B, R, H, C, P, N$ are the only displayed coordinates available to the local classifier.

The regular Cauchy side is read first. The window row W supplies a finite observation, the dyadic row D supplies the enclosing radius and endpoint ledger, and the readback row Q is downstream of those two rows.

The filter side is then forced through F and B. The basis row F exposes only finite Cauchy-filter requests, while B records the compatibility between those requests and the same dyadic enclosures already used by Q.

Finally R is reached through the displayed route, and H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the excluded host and quotient objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 3025.3 (Regular Cauchy enclosure-filter real seal route)**.** *An accepted* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ *carrier may expose its* $\mathsf{Real}^{\uparrow}$ *seal row R only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row F, and the compatibility row B have been displayed through H and C. Thus a Real or Completion consumer receives a finite enclosure-filter route for the regular Cauchy name, not a quotient stream, a selected limit, quotient filter equality, or ambient completeness.*

Proof. Start with the carrier projection. The Real-seal row R is a coordinate of E, but it is downstream of W, D, Q, F, B in the displayed route.

The rows W, D, Q bind the regular Cauchy readback to finite windows and dyadic enclosures. The rows F, B then bind the same enclosure data to finite Cauchy-filter basis requests, so the filter-facing read is not independent of the regular rational readback.

Transport through H and replay through C preserve this ordering, while P and N record provenance and local naming for the same packet. No carrier coordinate can supply quotient equality, a chosen limit, countable choice, host real equality, or ambient completeness; the Real seal therefore exports only the displayed finite enclosure-filter route. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ read used by a Real or Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ enclosure

ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{CauchyFilter}^{\uparrow}$ basis row, one compatibility row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: streamname, dyadicratcore, regseqrat, cauchyfilter, real: $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ is not bijective to any listed sibling because it binds regular Cauchy readback and Cauchy-filter basis membership through the same displayed dyadic enclosure ledger before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ packet is generated from a finite StreamName window, a DyadicRatCore enclosure ledger, a RegSeqRat readback row, a CauchyFilter basis row, an enclosure-basis compatibility row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$, seed)

Scope closed: Definition 4003.1, Definition 4003.2, and Definition 4003.3 seed the finite enclosure-filter route over StreamName, DyadicRatCore, RegSeqRat, CauchyFilter, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, quotient filter equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic enclosure exactness, regular readback ordering, Cauchy-filter basis compatibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEnclosureFilterUp.

Chapter 3026

A Concrete Naming Certificate for $\mathsf{MorreySpace}^{\uparrow}$

Following Morrey and the Sobolev-space treatment of Adams–Fournier, the $\mathsf{MorreySpace}^{\uparrow}$ packet records the local oscillation and radius-scaled integral budget used by constructive regularity consumers before any classical completion of measure or function spaces is available. It sits next to the weak-gradient packet of chapter 440, reads metric ball windows through chapter 57, measures oscillation and scale through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and faces the measure and integral rows of chapter 92 and chapter 123. The certificate names only finite BEDC histories: metric windows, local real-valued budgets, radius ledgers, transport, replay, provenance, and local naming.

Definition 3026.1 (MorreySpace carrier)**.** *A* $\mathsf{MorreySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, B, F, I, R, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the distance classifier used by local ball consumers.*
2. *B is the finite metric-ball window ledger, listing the inspected center histories, radius histories, and ball-membership witnesses.*
3. *F is the function-observation row, read through the same* $\mathsf{Real}^{\uparrow}$ *value seal that Sobolev and integral consumers use.*
4. *I is the local* $\mathsf{Integral}^{\uparrow}$ *or* $\mathsf{Measure}^{\uparrow}$*-facing budget row, recording the finite real-valued oscillation or integral magnitude attached to each inspected ball.*
5. *R is the radius-scale ledger, comparing the displayed radius with the exponent and dimension budget through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *readback rows.*
6. *T is the transport row that keeps the ball window, function observation, integral budget, and radius-scale ledger attached under the carrier classifier.*
7. *H records componentwise* hsame *transport for the metric, ball-window, function, local-budget, scale-ledger, transport, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a Morrey-space request through metric-ball inspection, local budget readback, radius scaling, and exported consumer access.*
9. *P is the* Pkg *provenance over* $\mathsf{MorreySpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a completed measure space, almost-everywhere quotient equality, Hahn–Banach separation, selected representatives, or full PDE regularity.

Theorem 3026.2 (MorreySpace NameCert obligations)**.** *For every accepted* $\mathsf{MorreySpace}^\uparrow$ *carrier* $M = (X, B, F, I, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$ *surface consists exactly of metric-source admission through* X*, metric-ball window exposure through* B*, Real-valued function observation through* F*, local oscillation or integral-budget exposure through* I*, radius-scale ledger comparison through* R*, classifier transport through* T *and* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MorreySpaceUp.MorreySpaceTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4004.1. The tuple displays the metric source, finite ball windows, Real-valued function observations, local budget row, radius-scale ledger, transport, replay, provenance, and local naming.

The analytic-facing part of the packet is finite. The metric row X and ball-window row B determine which local windows may be inspected, while F and I attach the Real-valued oscillation or integral budget to those same windows.

The scale row R records the radius comparison available to a Morrey consumer. The transport rows T, H and replay row C preserve this same route under classifier comparison and downstream access, and P, N source and name the route.

Since no remaining coordinate stores measure completion, quotient function equality, Hahn–Banach separation, a selected representative, or a PDE regularity theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 3026.3 (MorreySpace Sobolev-neighbour handoff)**.** *Every Sobolev-facing or regularity-facing consumer of an accepted* $\mathsf{MorreySpace}^\uparrow$ *carrier reads the local control route through*

$$\mathsf{Met}^\uparrow,\ \mathsf{Real}^\uparrow,\ \mathsf{Measure}^\uparrow,\ \mathsf{Integral}^\uparrow,\ \mathsf{Sobolev}^\uparrow \longmapsto \mathsf{MorreySpace}^\uparrow.$$

The route first displays metric-ball windows, then Real-valued local oscillation or integral-budget rows, then the radius-scale ledger, and only then exports the Morrey-space naming row. It supplies constructive analysis neighbour data for Sobolev and Real consumers, without asserting classical measure completion, Hahn–Banach separation, selected representatives, or full PDE regularity.

Proof. Use Definition 4004.2. It orders the metric source, ball-window ledger, function observation, local budget, radius-scale comparison, transport, replay, provenance, and local naming inside one accepted carrier.

The $\mathsf{Sobolev}^\uparrow$ neighbour sees only the displayed finite budget. Its weak-gradient or function-space request can consume the same $\mathsf{Real}^\uparrow$ and $\mathsf{Integral}^\uparrow$ rows, but it cannot introduce a completed function quotient or an external measure completion because those rows are absent from the carrier.

The radius-scale ledger then supplies the Morrey-facing distinction: local budgets are read together with ball radii before any exported consumer can use the packet. Thus the handoff is a finite BEDC NameCert route, not a classical regularity theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$**.**

Constructive story: A $\mathsf{MorreySpace}^\uparrow$ packet is generated from MetricSpaceUp ball windows, RealUp-valued local function observations, MeasureUp and IntegralUp budget rows, radius-scale ledgers, classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MorreySpace}^{\uparrow}}$, seed)

Scope closed: Definition 4004.1, Definition 4004.2, and Definition 4004.3 seed the metric-ball, local-budget, and radius-scale carrier consumed by Sobolev-neighbour constructive analysis chapters.

Formal verification: FormalStatus(NameCert$_{\mathsf{MorreySpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close classical measure completion, almost-everywhere quotient equality, Hahn–Banach separation, selected representatives, full PDE regularity, Lean verification, or bridge checking for MorreySpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, metric-ball exposure, local RealUp budget exactness, radius-scale ledger stability, Sobolev-neighbour handoff, transport, replay, provenance, and local naming for BEDC.Derived.MorreySpaceUp.

Chapter 3027

A Concrete Naming Certificate for $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$

Following the Heine–Cantor route of chapter 9237, the $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ packet records the finite radius-choice surface that turns compact finite-net data and pointwise-continuity radii into one tolerance-only uniform-modulus row. It sits between the compact metric finite-net surface of chapter 129, the continuous-map pointwise modulus row of chapter 130, the rational lower-bound fold of chapter 619, and the uniform-modulus consumer of chapter 369. The packet is a finite ProbeBundle choice ledger, not an open-cover compactness theorem or a choice-selected modulus.

Definition 3027.1 (Compact uniform radius-choice carrier)**.** *A* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, M, Q, F, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying the* ProbeBundle *of centers and coverage witnesses for the requested tolerance scale.*
2. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to those center histories.*
3. *Q is the positive radius-choice row, listing the local radius history selected by the displayed pointwise-modulus and finite-net data for each bundled probe.*
4. *F is the finite lower-bound fold row over* ProbeBundle*, read through the rational order and returning one positive radius history below every displayed local radius.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row, consuming only the tolerance history and the folded radius history.*
6. *H records componentwise* hsame *transport for compact-net, pointwise-modulus, radius-choice, finite-fold, output, replay, provenance, and local-name rows.*
7. *C records displayed* Cont *replay from tolerance input through finite-net inspection, local radius readback, lower-bound folding, and uniform-modulus export.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* ProbeBundle*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, a host Lebesgue-number minimum, a choice-selected modulus, sequential compactness, quotient real equality, or an ambient continuity theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformRadiusChoiceUp.CompactUniformRadiusChoiceTasteGate single carrier alignment`

Theorem 3027.2 (Compact uniform radius-choice NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *carrier* $R = (K, M, Q, F, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through* K*, pointwise-modulus admission through* M*, local positive-radius exposure through* Q*, finite ProbeBundle lower-bound folding through* F*, tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *output through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4006.1. The rows $K, M, Q, F, U, H, C, P, N$ are the complete displayed inventory.

The compact finite-net and pointwise-modulus rows determine the only local radii that may be inspected. The row Q records those radii as positive BEDC histories attached to bundled probes, not as host choices.

The fold row F then recurses over the displayed $\mathsf{ProbeBundle}$ and returns a positive lower-bound history below every radius named by Q. Since F is a finite rational-order fold, the exported radius is classifier-stable without invoking a global minimum operator.

Finally U, H, C, P, N export, transport, replay, source, and name exactly this radius-choice route. The omitted open-cover, sequential, quotient, and choice-selected objects are absent from the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 3027.3 (Tolerance-only uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *packet receives one folded radius history as a function of the requested tolerance history and the displayed finite compact-continuity packet:*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The consumer cannot read an open-cover compactness proof, a choice-selected family of local moduli, a host minimum over radii, quotient real equality, or an external modulus function.

Proof. Use Definition 4006.2. The obligation surface orders compact-net admission, pointwise-modulus admission, local radius exposure, finite lower-bound folding, and uniform-modulus export.

For a fixed tolerance history, the compact row supplies only a finite $\mathsf{ProbeBundle}$ of center histories. The pointwise row supplies the local radii attached to those centers, and the radius-choice row records them as displayed positive histories.

The lower-bound fold produces the single radius consumed by U. Because all preceding rows are tied by H and replayed by C, the exported uniform-modulus read depends only on the displayed tolerance and finite packet, not on any external selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, local positive radius-choice rows, a finite ProbeBundle rational lower-bound fold, a UniformModulusUp output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4006.1, Definition 4006.2, and Definition 4006.3 seed the finite radius-choice ledger consumed by compact-uniform-continuity and uniform-modulus chapters.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected moduli, host Lebesgue-number minima, sequential compactness, quotient real equality, Lean verification, or bridge checking for CompactUniformRadiusChoice$^{\uparrow}$.

Upgrade path: Obligation closure requires checked compact-net admission, pointwise-modulus admission, local radius-choice exactness, finite lower-bound fold stability, UniformModulusUp output exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformRadiusChoiceUp.

Chapter 3028

A Concrete Naming Certificate for $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style compact-uniform language, the $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ packet records the finite tail-index surface by which an eventually stable family of continuous maps receives a shared compact-source modulus. It sits after the compact radius-choice packet of chapter 4006, the uniform family-modulus route of chapter 3844, the equicontinuity family ledger of chapter 413, the compact source of chapter 129, and the graph rows of chapter 130. Its terminal reads pass through finite $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{Real}^{\uparrow}$ seal rather than through a host function-space topology, a selected subsequence, or an ambient completeness theorem.

Definition 3028.1 (Asymptotic equicontinuity carrier)**.** *An* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, E, U, T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact finite-net carrier and the metric-distance ledger used by all later rows.*
2. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose graph and pointwise-modulus data are inspected at displayed compact-source centers.*
3. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, binding the family rows to a shared local tolerance-radius ledger before any tail condition is read.*
4. *U is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row, exporting the compact finite-net fold that yields a single tolerance-indexed source radius for the displayed family window.*
5. *T is the asymptotic tail row. It records the finite index threshold after which the family window must use the same compact-source radius and target tolerance route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened after T, used to inspect only the tail-admitted source and target observations.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the tail-admitted distance and tolerance reads after K, F, E, U, T, S have been replayed.*
8. *H records componentwise* hsame *transport for compact source, continuous-family, equicontinuity, uniform-modulus, tail, stream-window, real-seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a tail-index request through the compact finite-net and uniform-equicontinuity rows to the real-seal boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$, $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$, UniformEquicontinuityM EquicontinuityFamily$^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function-space compactness, a selected subsequence, quotient equality of maps, countable choice over tails, an external modulus function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AsymptoticEquicontinuityUp.AsymptoticEquicontinuityTasteGate single carrier alignment`

Theorem 3028.2 (Asymptotic equicontinuity NameCert obligations)**.** *For every accepted* AsymptoticEquicontinuity *carrier $A = (K, F, E, U, T, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, continuous-family admission through F, family equicontinuity admission through E, shared uniform-modulus export through U, asymptotic tail admission through T, finite stream-window inspection through S, terminal real-seal routing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 4007.1. The rows K, F, E, U, T, S, R are the complete analytic inventory, and H, C, P, N are the structural rows that transport, replay, source, and name that inventory.

The compact and family rows are read before the tail row. The compact source K supplies only finite-net and metric-distance data, while F and E expose the continuous-family and shared equicontinuity ledger. The row U then records the uniform family modulus already folded through compact finite-net data.

The asymptotic step is exactly the row T. Once T has displayed the finite threshold, the stream row S may inspect only tail-admitted windows, and the real seal R may be reached only through those windows.

Finally the carrier has no coordinate for a selected subsequence, host function-space topology, quotient map equality, external modulus function, countable tail choice, or ambient Real completeness. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 3028.3 (Asymptotic equicontinuity modulus handoff)**.** *Every compact-window or Real-facing consumer of an accepted* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ *carrier receives its tail-stable modulus through the displayed route*

$\mathsf{CompactUniformRadiusChoice}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, Continuou

The handoff supplies one finite tail threshold and one compact-source modulus route, not a selected subsequence, host function-space compactness proof, quotient equality of functions, countable-choice tail, or ambient Real-completeness witness.

Proof. Start with the compact-source and family evidence. The rows K, F, E display the compact metric source, the continuous-map family window, and the equicontinuity-family ledger. The compact radius-choice and uniform-equicontinuity rows are then read through U, so the source radius is already a finite compact-net handoff.

Next expose the tail row T. It records the threshold after which the same compact-source modulus route must serve the accepted family window. The stream row S opens only the finite windows admitted by this threshold, and R seals the resulting distance and tolerance reads as $\mathsf{Real}^{\uparrow}$ data.

The structural rows do not add analytic content. They preserve the displayed route componentwise, replay it as a continuation, record its provenance, and name it locally. Hence every consumer receives the same finite modulus handoff, with no hidden selector or external compactness principle. □

Falsifiable prediction: An accepted $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ read is refuted if it reaches a Real-facing tail claim without first exposing the compact source, continuous-family ledger, equicontinuity family row, uniform-equicontinuity modulus row, asymptotic tail threshold, StreamName windows, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: compact-uniform-radius-choice, uniform-equicontinuity-modulus, equicontinuityfamily, compactmetric, continuousmap, streamname, real: none alone is bijective with $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ because the packet binds compact finite-net modulus data to an eventual tail threshold before any Real-facing consumer can read the route.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ packet is generated from CompactUniformRadiusChoice, UniformEquicontinuityModulus, EquicontinuityFamily, CompactMetric, ContinuousMap, StreamName, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4007.1, Definition 4007.2, and Definition 4007.3 seed the finite compact-source tail-modulus route for asymptotic equicontinuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.AsymptoticEquicontinuityUp.AsymptoticEquicontinuityTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Arzela–Ascoli compactness, host function-space topology, selected subsequences, quotient equality of maps, countable tail choice, Lean verification, or bridge checking for $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, compact-source admission, family equicontinuity admission, uniform-modulus handoff, tail-threshold exactness, StreamName window restriction, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.AsymptoticEquicontinuityUp.

Chapter 3029

A Concrete Naming Certificate for $\mathsf{ProkhorovTightProbability}^{\uparrow}$

Following Prokhorov's tightness theorem for probability measures, the $\mathsf{ProkhorovTightProbability}^{\uparrow}$ packet records only the constructive finite tightness data needed before weak-convergence consumers can inspect a probability family. It starts from the probability source of chapter 184, the pushforward distribution row of chapter 186, the event-surface measure row of chapter 92, the compact-test packet of chapter 129, the finite-net witness of chapter 1767, and the located outside-mass ledger of chapter 3795. The chapter also reads the underlying real boundary of chapter 38. It is a finite tightness certificate, not a proof of weak compactness, selected subsequences, countable choice, or ambient real completeness.

Definition 3029.1 (Prokhorov tight probability carrier)**.** *A* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (P, D, M, K, N, E, R, T, H, C, Q, L)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the probability mass surface.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *pushforward row whose target-side event reads are being tested for tightness.*
3. *M is the* $\mathsf{Measure}^{\uparrow}$ *event-surface row through which the outside event and its mass are read.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact-test row naming the compact metric test surface used by the packet.*
5. *N is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row displayed inside the compact test for the requested tolerance.*
6. *E is the outside-mass error-bound row read through* $\mathsf{RealTightness}^{\uparrow}$ *and the underlying* $\mathsf{Real}^{\uparrow}$ *comparison surface.*
7. *R is the finite radius or tolerance ledger that aligns the compact test, finite net, and outside-mass request.*
8. *T records transport from the probability source through distribution, measure, compact-test, finite-net, and real-tightness rows.*
9. *H records componentwise* hsame *transport for every displayed row.*

10. *C records displayed* Cont *replay of source admission, pushforward readback, event-surface inspection, compact-test exposure, finite-net inspection, outside-mass comparison, and radius-ledger readback.*
11. *Q is the* Pkg *provenance over* $\mathsf{ProkhorovTightProbability}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{RealTightness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ProkhorovTightProbabilityUp.ProkhorovTightProbabilityUp`

Theorem 3029.2 (Prokhorov tight probability NameCert obligations)**.** *For every accepted* ProkhorovTightProbabili *carrier $Z = (P, D, M, K, N, E, R, T, H, C, Q, L)$, the local* $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$ *surface consists exactly of five obligation groups:*

1. *carrier admission for the probability source, pushforward distribution, event-surface measure, compact test, finite net, outside-mass bound, radius ledger, transport, replay, provenance, and local naming rows;*
2. *compact-test exposure, requiring every tightness read to display the* $\mathsf{CompactMetric}^{\uparrow}$ *row K and the internal* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row N before the outside event is consumed;*
3. *outside-mass bound exactness, requiring the error row E to be read through* $\mathsf{RealTightness}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rather than through a host real inequality;*
4. *distribution/measure compatibility, requiring the pushforward row D and event-surface row M to name the same measured outside event under the displayed transport T;*
5. *non-escape from open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, and ambient real completeness.*

No additional NameCert obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4008.1. The tuple lists exactly the probability source, distribution pushforward, event-surface measure, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming rows.

The compact side is finite. A tightness read must expose K and the internal finite-net row N before it can identify the outside event measured by M.

The outside-mass side is real-name data. The error row E is read through the located tightness and real comparison surfaces, while D, M, T keep the distribution and measure event rows compatible.

Since no carrier coordinate stores open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient real completeness, the five displayed groups exhaust the local NameCert obligations. □

Theorem 3029.3 (Prokhorov tight probability finite-net handoff)**.** *Every accepted tightness read from a* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ProbSpace}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto \mathsf{Measure}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{RealTightness}^{\uparrow} \longmapsto \mathsf{Real}$$

before any weak-convergence consumer can inspect it. The exported read is the displayed probability, distribution, measure, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming route; it exposes no weak compactness theorem, selected subsequence, countable-choice selector, host measure, quotient event, or ambient completeness row.

Proof. Start with the NameCert obligations of Definition 4008.2. Carrier admission places the probability source and pushforward distribution before the measured outside event.

Next read the compact finite test. The measure row is inspected only after the compact metric row and its finite-net witness are displayed for the requested tolerance ledger.

Finally the outside-mass bound is consumed through $\mathsf{RealTightness}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The structural rows transport and replay this same finite route, so a weak-convergence consumer receives only the displayed handoff and cannot insert an unlisted compactness, sequence-selection, quotient, host-measure, or completeness coordinate. □

Theorem 3029.4 (Prokhorov tight probability Portmanteau readiness)**.** *Every Portmanteau-facing weak-convergence test read from an accepted* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *carrier receives only the displayed probability source, pushforward distribution, measure event surface, compact metric finite-net test,* $\mathsf{RealTightness}^{\uparrow}$ *outside-mass row,* $\mathsf{Real}^{\uparrow}$ *comparison, transport, replay, provenance, and local* NameCert *rows before bounded-continuous probes may inspect it. The readiness route exports no weak compactness, selected subsequence, countable choice, host measure, quotient event, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4008.2. The local NameCert obligations expose exactly the probability source, distribution pushforward, measure event surface, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming rows.

Next apply the finite-net handoff of Definition 4008.3. It orders the compact metric finite-net test before the outside-mass row and routes the comparison through $\mathsf{RealTightness}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ before any weak-convergence consumer can inspect the packet.

Finally pass the resulting finite surface to the Portmanteau consumer of chapter 4193. Because the carrier contains no coordinate for weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient real completeness, bounded-continuous probes receive only the displayed readiness route. □

Falsifiable prediction: Every accepted $\mathsf{ProkhorovTightProbability}^{\uparrow}$ tightness read factors through $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{RealTightness}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any weak-convergence consumer can inspect it.

Independence witness: probspace, distribution, measure, compactmetric, finite-epsilon-net, real-tightness, real: the packet is not bijective to any listed sibling because it binds the probability source, pushforward event surface, compact finite-net test, outside-mass error row, radius ledger, transport, replay, provenance, and local NameCert row as one tight probability surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProkhorovTightProbability}^{\uparrow}$ packet is generated from ProbSpaceUp source rows, DistributionUp pushforward rows, MeasureUp event-surface rows, CompactMetricUp compact-test rows, FiniteEpsilonNetUp finite-net rows, RealTightnessUp outside-mass rows over RealUp, a finite tolerance ledger, transport, hsame, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$, seed)

Scope closed: Definition 4008.1, Definition 4008.2, and Definition 4008.3 seed the finite tight-probability handoff for probability compactness consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-test exposure, outside-mass exactness, distribution/measure compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ProkhorovTightProbabilityU

Chapter 3030

A Concrete Naming Certificate for $\mathsf{CauchyTailEstimate}^{\uparrow}$

Following Bishop and Bridges' constructive analysis of Cauchy sequences and finite tail bounds, the $\mathsf{CauchyTailEstimate}^{\uparrow}$ packet records the displayed route by which a requested tail index and dyadic radius become a bounded regular-rational read before any $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits after the tail-filter surface of chapter 3850 and the fast criterion of chapter 3825, and it reads its tolerance, regular readback, and endpoint seal through chapter 291, chapter 301, and chapter 38. The packet is a finite certificate for Real and Completion consumers; it is not a quotient sequence, a selected limit, countable choice, host real equality, trichotomy, or ambient real completeness.

Definition 3030.1 (Cauchy tail estimate carrier)**.** *A* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the tail-index row, naming the finite threshold after which the estimate may be inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, naming the dyadic tolerance used by the tail-bound comparison.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed tail window and radius data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after T, D, R have been replayed.*
5. *B is the tail-bound ledger, recording the finite comparison witnesses that keep every accepted read beyond T inside the radius named by D.*
6. *H records componentwise* hsame *transport for T, D, R, E, B and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through T, D, B, R, and then E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{FastCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed tail threshold, dyadic radius, regular readback, real seal, tail-bound ledger, continuation, provenance, and local naming

rows to a common finite surface. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, trichotomy, open-cover compactness, or ambient completeness.

Theorem 3030.2 (Cauchy tail estimate NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier* $Q = (T, D, R, E, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed tail index, dyadic radius, regular readback, real seal, tail-bound ledger, transport, continuation, provenance, and local name;*
2. *classifier transport to a common* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{FastCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite surface;*
3. *stability of* T, D, R, E, B *under the componentwise* hsame *rows recorded in* H*;*
4. *handoff exactness requiring every real-facing read to pass through* T, D, B, R, E *in that order;*
5. *refusal rows excluding quotient streams, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyTailEstimateUp.CauchyTailEstimateUpTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4009.1. The projection exposes exactly the tail-index row, dyadic radius row, regular readback row, real seal row, tail-bound ledger, transport row, continuation row, provenance row, and local naming row.

Classifier transport is componentwise. The rows in H move the mathematical coordinates T, D, R, E, B and the administrative coordinates C, P, N to the same finite comparison surface.

Handoff exactness is read from the displayed continuation row. A tail request first reaches the threshold row T, then the dyadic radius row D, then the ledger row B that witnesses the tail-bound comparison, then the regular readback R, and only then the real seal E.

The refusal rows are exhausted by the carrier inventory. Since the packet has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or completeness, no local NameCert obligation can export those data. □

Theorem 3030.3 (Cauchy tail estimate RegSeqRat handoff)**.** *For every accepted* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier* $Q = (T, D, R, E, B, H, C, P, N)$*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of* R *factors through the tail-index row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D*, and the tail-bound ledger* B *before it can expose the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The handoff route is therefore finite and does not identify Cauchy names by quotient equality, select a limit, invoke countable choice, or appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a regular-rational consumer of R. Carrier admission lists R only as the readback row reached from the displayed tail threshold and dyadic radius.

Replay the continuation row C. Its route sends the request first to T, checks the requested radius through D, reads the comparison witnesses in B, and only then exposes the regular rational row R.

Transport through H preserves this dependency order. The package row P and local name N record provenance and naming for the same finite packet; they do not add a second readback coordinate.

Thus every $\mathsf{RegSeqRat}^{\uparrow}$ handoff factors through T, D, B before R and reaches E only after that route has been displayed. The carrier contains no coordinate from which a bypass route could obtain quotient equality, selected limits, choice, host real equality, or completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailEstimate}^{\uparrow}$ packet is generated from a finite tail-index row, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal row, a tail-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4009.1, Definition 4009.2, and Definition 4009.3 bind the finite tail-bound carrier, local NameCert obligations, and RegSeqRat handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, stability, handoff exactness, refusal rows, and TasteGate nontriviality with field faithfulness for $T, D, R, E, B, H, C, P, N$.

Chapter 3031

A Concrete Naming Certificate for $\mathsf{Circle}^{\uparrow}$

Following Hatcher and Munkres on the circle as a compact one-dimensional boundary object, the $\mathsf{Circle}^{\uparrow}$ packet records the topological boundary reading of a one-dimensional closed loop without identifying it with the algebraic $\mathsf{S}^{1\uparrow}$ group certificate. It sits beside $\mathsf{S}^{1\uparrow}$ of chapter 58, $\mathsf{Real}^{\uparrow}$ of chapter 38, $\mathsf{Met}^{\uparrow}$ of chapter 57, and $\mathsf{Compact}^{\uparrow}$ of chapter 60: those siblings supply phase multiplication, real-coordinate readback, metric inspection, and compact covering rows, while this packet binds the finite boundary carrier and its NameCert obligations as a topological circle surface.

Definition 3031.1 (Circle carrier)**.** *A* $\mathsf{Circle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, R, M, K, S, H, P, N)$$

with the following displayed rows:

1. *B is the boundary-loop row, naming the closed one-dimensional boundary surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate-read row, recording the finite real-coordinate inspection used by local charts or parametrized arcs.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row, recording the metric neighborhoods through which boundary adjacency is read.*
4. *K is the* $\mathsf{Compact}^{\uparrow}$ *row, recording finite cover data for the closed boundary surface.*
5. *S is the* $\mathsf{S}^{1\uparrow}$ *handoff row, available only when a consumer asks for the phase-group reading of the same boundary.*
6. *H records componentwise* hsame *transport for the boundary, real-coordinate, metric, compactness, handoff, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{Circle}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, BHist, hsame, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient plane, a selected trigonometric parametrization, a quotient by angle equality, or a global orientation choice.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier boundary route certificate`

Theorem 3031.2 (Circle NameCert obligations)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *carrier*

$$C = (B, R, M, K, S, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *surface consists exactly of carrier admission for the boundary-loop row, real-coordinate readback, metric-neighborhood row, compact-cover row, optional* $\mathsf{S}^{1\uparrow}$ *handoff, componentwise transport, package provenance, and local naming. Any topological-circle consumer must read the closed boundary surface through* B, R, M, K, H, P, N *before using the algebraic phase handoff* S*.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleUpTasteGate single carrier alignment`

Proof. Project the displayed carrier of Definition 4010.1. The topological rows are B, R, M, K: they name the closed boundary, the real-coordinate readback, the metric neighborhoods, and the compact cover data.

The handoff row S is downstream. It allows a consumer to compare the boundary packet with the $\mathsf{S}^{1\uparrow}$ phase-group sibling, but it does not replace the boundary carrier by an algebraic group row.

The rows H, P, N transport, source, and name the same finite packet. Since no ambient plane, trigonometric parametrization, quotient-angle equality, or orientation coordinate appears in the carrier, the listed rows exhaust the $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ obligations. □

Theorem 3031.3 (Circle and $\mathsf{S}^{1\uparrow}$ boundary handoff)**.** *Every* $\mathsf{S}^{1\uparrow}$*-facing read of an accepted* $\mathsf{Circle}^{\uparrow}$ *carrier factors through the boundary-loop row* B*, real-coordinate readback* R*, metric row* M*, compact-cover row* K*, and handoff row* S*. The handoff may expose a phase-group consumer, but it cannot make* $\mathsf{S}^{1\uparrow}$ *the carrier of topological compact-boundary data and cannot export quotient-angle equality or a selected global parametrization.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleSOneBoundaryHandoff`

Proof. Start with a consumer asking for the phase-group reading. The $\mathsf{Circle}^{\uparrow}$ carrier first exposes the boundary surface through B and its local real and metric inspections through R, M.

Compactness is read through K, so the consumer receives a finite cover surface for the closed boundary before any phase handoff is available. Only then can S point to the $\mathsf{S}^{1\uparrow}$ sibling.

The componentwise transport, provenance, and naming rows H, P, N preserve this order. Because the carrier has no quotient-angle coordinate or selected global parametrization, the handoff remains a boundary-to-phase route rather than an identification of the two packets. □

Theorem 3031.4 (Circle metric subspace carrier obligation)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *carrier* $C = (B, R, M, K, S, H, P, N)$*, the boundary row* B*,* $\mathsf{Real}^{\uparrow}$ *coordinate row* R*,* $\mathsf{Met}^{\uparrow}$ *neighborhood row* M*,* $\mathsf{Compact}^{\uparrow}$ *cover row* K*,* $\mathsf{S}^{1\uparrow}$ *handoff row* S*, transport row* H*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *row* N *are the full carrier obligation surface. A consumer may not replace these rows by an ambient plane, a selected parametrization, quotient-angle equality, or a global orientation choice.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleMetricSubspaceCarrierObligation`

Proof. Project the carrier of Definition 4010.1. Its analytic boundary surface is exactly B, R, M, K, S, and its structural surface is exactly H, P, N.

No other coordinate is available for a circle consumer. Since the carrier omits an ambient plane, selected trigonometric map, quotient by angle equality, and orientation row, the displayed coordinates exhaust the carrier obligation surface. □

Theorem 3031.5 (Circle $\mathsf{S}^{1\uparrow}$ real seal route)**.** *Every phase-group read of an accepted* $\mathsf{Circle}^{\uparrow}$ *carrier must pass through the* $\mathsf{Real}^{\uparrow}$ *coordinate row* R *and the* $\mathsf{S}^{1\uparrow}$ *handoff row* S*, with the boundary row* B*, metric row* M*, compact row* K*, transport* H*, provenance* P*, and local naming* N *still present. This route permits a phase interpretation but does not identify* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleSOneRealSealRoute`

Proof. Start with a consumer that asks for phase-group data. The carrier first exposes the topological boundary and real-coordinate surface through B and R.

The handoff row S can then point to the $\mathsf{S}^{1\uparrow}$ sibling, while M, K, H, P, N keep the same metric, compact, transport, provenance, and naming rows attached to the circle packet. Because S is a handoff coordinate rather than the carrier itself, the route is not an identification of the two chapters. □

Theorem 3031.6 (Circle compact metric boundary)**.** *Before any consumer reads* $\mathsf{Circle}^{\uparrow}$ *as a closed boundary, the packet must expose the compact-cover row* K*, the metric-neighborhood row* M*, the boundary row* B*, the* $\mathsf{Real}^{\uparrow}$ *coordinate row* R*, transport* H*, provenance* P*, and local naming* N*. The compact-metric boundary does not supply an ambient plane, a selected parametrization, or quotient-angle equality.*

Proof. Read the compact and metric coordinates together. The row K supplies finite cover data for the closed boundary, and M supplies the neighborhood inspection by which boundary adjacency is read.

The rows B, R, H, P, N bind those compact and metric inspections to the circle carrier and its local name. Since no ambient plane or quotient-angle coordinate appears in the packet, a closed-boundary consumer receives only this compact-metric surface. □

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier compact metric boundary`

Theorem 3031.7 (Circle Real metric scope route)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *read of* $C = (B, R, M, K, S, H, P, N)$*, the boundary inspection is scoped by the displayed* BHist *carrier rows, componentwise* hsame *transport,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *naming, the* $\mathsf{Real}^{\uparrow}$ *coordinate row, the* $\mathsf{Met}^{\uparrow}$ *neighborhood row, the* $\mathsf{Compact}^{\uparrow}$ *finite-cover row, and the optional* $\mathsf{S}^{1\uparrow}$ *handoff row. The route exports no ambient plane, quotient-angle equality, selected parametrization, global orientation, or identification of* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$*.*

Proof. Project the carrier of Definition 4010.1. The row list exposes the boundary row B, real-coordinate inspection R, metric inspection M, compact cover K, optional $\mathsf{S}^{1\uparrow}$ handoff S, transport H, provenance P, and local naming N.

Apply Definition 4010.2. Its NameCert surface forces the topological rows B, R, M, K, H, P, N to be read before any phase handoff can be used.

Use Definition 4010.3 and Definition 4010.5. They keep S as a downstream handoff through the same boundary and $\mathsf{Real}^{\uparrow}$ rows, not as an identification with the circle carrier.

Finally apply Definition 4010.6. The compact and metric rows already exhaust the closed-boundary inspection surface together with B, R, H, P, N, so the scoped route is exactly the displayed BEDC row surface and nothing from an ambient plane, quotient angle, selected parametrization, or global orientation remains available. □

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier real metric scope route`

Theorem 3031.8 (Circle Real metric scoped closure)**.** *Every accepted* $\mathsf{Circle}^{\uparrow}$ *read is scoped exactly by the carrier rows* B, R, M, K, S, H, P, N *of Definition 4010.1, the displayed* BHist *packet, componentwise* hsame *transport,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *naming, and the sibling surfaces* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, *and optional* $\mathsf{S}^{1\uparrow}$. *The scoped route exports no ambient plane, quotient-angle equality, selected parametrization, global orientation, or identification of* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier real metric scoped closure`

Proof. Apply Definition 4010.2. The accepted read begins with the carrier rows B, R, M, K, S, H, P, N and the local $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ surface, so no unlisted coordinate can enter the scope.

Use Definition 4010.3 and Definition 4010.5. These results keep the optional $\mathsf{S}^{1\uparrow}$ row as a handoff from the boundary and real-coordinate rows, not as an identification of the circle packet with the phase-group sibling.

Apply Definition 4010.6 and Definition 4010.7. The compact, metric, and real rows already determine the closed-boundary inspection surface through BHist, hsame, Pkg, and local naming.

Thus the scoped closure is exactly the displayed BEDC carrier surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, and optional $\mathsf{S}^{1\uparrow}$, while ambient planes, quotient-angle equality, selected parametrizations, global orientations, and carrier identification with $\mathsf{S}^{1\uparrow}$ remain outside the packet. □

Theorem 3031.9 (Circle public boundary export)**.** *The public* $\mathsf{Circle}^{\uparrow}$ *export is exactly the boundary-loop row* B, $\mathsf{Real}^{\uparrow}$ *coordinate row* R, $\mathsf{Met}^{\uparrow}$ *neighborhood row* M, $\mathsf{Compact}^{\uparrow}$ *cover row* K, *optional* $\mathsf{S}^{1\uparrow}$ *handoff row* S, *componentwise* hsame *transport row* H, Pkg *provenance row* P, *and local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *route* N *of Definition 4010.1. A downstream consumer may cite this boundary as the public circle surface, but it receives no ambient plane, quotient-angle equality, selected parametrization, global orientation, or carrier identification with* $\mathsf{S}^{1\uparrow}$.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier public boundary export`

Proof. Start with the scoped closure of Definition 4010.8. It already identifies the visible BEDC carrier surface as B, R, M, K, S, H, P, N over the boundary, Real-coordinate, metric, compact, optional phase-handoff, transport, provenance, and local naming rows.

Now read that scoped packet as an outward-facing boundary. The NameCert obligations of Definition 4010.2 keep the topological circle read on the boundary, coordinate, metric, compact, transport, provenance, and local naming rows before the optional handoff can be used.

Finally apply the handoff and compact-metric exclusions of Definition 4010.3, Definition 4010.5, and Definition 4010.6. They keep S as an optional route to the $\mathsf{S}^{1\uparrow}$ sibling and exclude the ambient plane, quotient-angle, parametrization, orientation, and carrier-identification readings. Thus the public boundary exports exactly the displayed carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$.

Constructive story: A $\mathsf{Circle}^{\uparrow}$ packet is generated from boundary-loop, Real-coordinate, metric-neighborhood, compact-cover, optional SOne handoff, componentwise hsame transport, package provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$, publicClosed)

Scope closed: Definition 4010.1, Definition 4010.2, Definition 4010.3, Definition 4010.4, Definition 4010.5, Definition 4010.6, Definition 4010.7, Definition 4010.8, and Definition 4010.9 bind the topological boundary certificate, compact-metric obligations, Real metric route, algebraic phase handoff, and public boundary export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global parametrization, quotient-angle equality, ambient-plane data, or an identification of $\mathsf{Circle}^{\uparrow}$ with $\mathsf{S}^{1\uparrow}$.

Upgrade path: Formal closure requires checked carrier admission, boundary stability, compact-cover exactness, SOne handoff noncollapse, transport, provenance, and local naming for the same finite packet.

Chapter 3032

A Concrete Naming Certificate for $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$

The $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ packet realizes Definition 9463.12 as a finite NameCert carrier for the three-role discipline of focused disk readout, inscription-side self-center, and far-end boundary reading. It sits beside $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ of chapter 3500, $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ of chapter 1071, $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ of chapter 3486, and $\mathsf{UnitDisk}^{\uparrow}$ of chapter 377; those siblings supply boundary-fixed transport, local inscription sealing, observer-expansion ledger data, and disk-coordinate rows, while this packet binds the self-center reading itself.

Definition 3032.1 (Hyperbolic self-center reading carrier)**.** *A* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, I, B, T, V, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-readout row, recording the finite* $\mathsf{UnitDisk}^{\uparrow}$ *or Poincare-disk coordinate read of the focused ledger event.*
2. *I is the inscription row, recording the* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ *reading of the local self-center.*
3. *B is the far-end boundary row, recording the boundary-side* T *reading used by the hyperbolic vision.*
4. *T is the boundary-fixed transport row, recording the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *route that preserves the boundary reading while disk data move.*
5. *V is the noncollapse verdict row, recording that D, I, and B are three roles and not three hidden objects or one collapsed object.*
6. *H records componentwise* hsame *transport for disk readout, inscription, boundary, boundary-fixed transport, noncollapse verdict, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for reading D, I, B, T, V in the same finite packet.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$, $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden subject, a global observer point, a selected geodesic center, a physical constant, or an identification of disk readout with boundary far end.

Lean4 ✓ `BEDC.Derived.HyperbolicSelfCenterReadingUp.HyperbolicSelfCenterReadingTasteGate single carrier alignment`

Theorem 3032.2 (Hyperbolic self-center reading NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *carrier*

$$R = (D, I, B, T, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$ *surface consists exactly of disk-readout admission, inscription admission, far-end boundary admission, boundary-fixed transport, noncollapse verdict, componentwise transport, displayed replay, provenance, and local naming. No obligation may identify the records-side disk row* D*, the inscription-side row* I*, and the far-end boundary row* B*.*

Proof. Project the accepted carrier of Definition 4011.1. The three geometric roles are displayed separately as D, I, B.

The transport row T moves disk-facing data while preserving the boundary reading, and the verdict row V records the refusal to collapse the three roles. These rows are part of the same finite packet and cannot be replaced by prose analogy.

Finally, H, C, P, N transport, replay, source, and name the packet. Since the carrier has no hidden subject, global observer point, selected geodesic center, physical constant, or identification coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3032.3 (Hyperbolic self-center reading noncollapse)**.** *Every accepted* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *read keeps disk readout* D*, inscription row* I*, and far-end boundary row* B *distinct under the boundary-fixed transport row* T *and the noncollapse verdict row* V*. A consumer may cite their relation only through* $R = (D, I, B, T, V, H, C, P, N)$*, and may not export a hidden subject, a single collapsed self-center object, or a physical constant from the geometry alone.*

Proof. Begin with the three displayed roles. The disk-readout row D is records-side data, the inscription row I is the local self-center seal, and the boundary row B is the far-end reading.

The transport row T relates disk and boundary readings without erasing that distinction. The verdict row V records the noncollapse condition before any downstream consumer can package the read.

The structural rows H, C, P, N preserve the same finite inventory. Because no hidden-subject, collapsed-object, or physical-constant coordinate appears in the carrier, every accepted reading remains inside the displayed packet. □

Theorem 3032.4 (Hyperbolic self-center reading provenance route)**.** *The* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *packet routes its focused-ledger context through* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$*, disk coordinates through* $\mathsf{UnitDisk}^{\uparrow}$*, local inscription through* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$*, and boundary-fixed movement through* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*. These sibling routes are consumed as finite rows of one carrier and do not supply an ambient hyperbolic model, a global observer, or a host-geometric self.*

Proof. Read the provenance row P together with the displayed rows D, I, B, T, V. It names the focused-ledger, disk-coordinate, inscription, and boundary-fixed transport sources that the carrier is allowed to consume.

The replay row C then orders those reads inside one finite packet. A consumer can replay the disk readout, local inscription, far-end boundary reading, transport, and noncollapse verdict, but it receives no external geometric model.

Finally, H and N preserve and name the same packet. The absence of a global observer or host-geometric self coordinate gives the stated boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ packet is generated from disk-readout, local inscription, far-end boundary, boundary-fixed transport, noncollapse verdict, componentwise hsame transport, displayed replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4011.1, Definition 4011.2, Definition 4011.3, and Definition 4011.4 seed the finite carrier for the hyperbolic self-center reading.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a global observer, hidden subject, physical constant, or ambient hyperbolic model.

Upgrade path: Obligation closure requires checked carrier admission, disk-readout stability, inscription and boundary noncollapse, boundary-fixed transport, replay, provenance, and local naming for the same finite packet.

Chapter 3033

A Concrete Naming Certificate for $\mathsf{ClosedIntervalNesting}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ClosedIntervalNesting}^{\uparrow}$ packet records the finite nested-closed-interval route used before a retained interval family is read as regular rational Cauchy data. It sits over the dyadic endpoint core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the interval certificate surface of chapter 56. The packet names the L10 nested-interval handoff only; it does not assert a Dedekind-cut construction, a trichotomy principle, quotient equality of real names, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3033.1 (Closed interval nesting carrier)**.** *A* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, I, S, W, R, Q, H, C, P, N)$$

with the following visible rows:

1. *E is the dyadic closed-endpoint row. It records left and right endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$*, endpoint order, midpoint arithmetic cells, and the dyadic comparison budget for each inspected interval.*
2. *I is the nesting-inclusion row. It displays, for every inspected successor stage, the finite evidence that the successor closed interval is contained in its predecessor.*
3. *S is the shrinking-width schedule. It records the requested precision, the displayed stage bound, and the dyadic inequality that makes the inspected interval width smaller than the requested tolerance.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint, inclusion, width, and stage-bound observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the chosen endpoint stream and the width schedule after E, I, S, W have been displayed.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is opened only after the dyadic endpoints, nesting inclusions, shrinking-width schedule, stream windows, and regular rational readback have been replayed.*
7. *H records componentwise* hsame *transport for the endpoint row, inclusion row, width schedule, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing or real-name request through $E, I, S, W, R, Q, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalNesting}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *interval certificates,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$ *row.*

The classifier compares accepted nesting carriers only through $E, I, S, W, R, Q, H, C, P, N$. It has no coordinate for Dedekind cuts, trichotomy, quotient equality of real names, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalNestingUp.TasteGate.ClosedIntervalNestingTasteGate single carrier alignment`

Theorem 3033.2 (Closed interval nesting NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier $K = (E, I, S, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic closed-endpoint exposure, successor nesting inclusion, shrinking-width scheduling, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the carrier of Definition 4012.1.*

Proof. Carrier admission is projection from Definition 4012.1. The endpoint row E and inclusion row I are the only rows that expose closed-interval data, so every local naming obligation begins with displayed dyadic endpoints and finite successor containment.

The width and window obligations are consecutive. The schedule row S supplies the stage bound and dyadic width inequality, while W exposes the finite endpoint, inclusion, width, and stage observations used by that schedule.

The readback rows R, Q are downstream of those displayed observations. The regular rational row repacks the endpoint stream under the shrinking-width bound, and the real seal is available only after that regular readback has been replayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for Dedekind cuts, trichotomy, quotient equality, selected limits, choice, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 3033.3 (Closed interval nesting RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier reaches regular rational data only through the displayed finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may be consumed by completion and real-name routes as a nested closed-interval certificate, but it exports no Dedekind-cut construction, trichotomy row, quotient equality of real names, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $K = (E, I, S, W, R, Q, H, C, P, N)$. The endpoint row E and nesting row I expose the only closed-interval information available to a consumer.

Read the shrinking-width schedule S next. It ties every requested tolerance to a displayed stage bound and a dyadic width inequality, so the route cannot be interpreted as a selected intersection point or a hidden completed limit.

The stream and regular rows W, R then turn those finite endpoint windows into regular rational readback. The terminal row Q is a real-name seal only after the dyadic endpoint, inclusion, width, and stream rows have been displayed.

Finally H, C, P, N preserve the same order under transport, replay, provenance, and local naming. Because no stronger object appears among the carrier coordinates, every $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains the displayed finite nested-interval route. □

Falsifiable prediction: A RegSeqRat or Real consumer that omits the dyadic endpoint row, nesting-inclusion row, shrinking-width schedule, StreamName window row, or regular rational readback row cannot claim a $\mathsf{ClosedIntervalNesting}^{\uparrow}$ handoff to a Real-facing seal.

Independence witness: dyadicratcore, streamname, regseqrat, real, interval: $\mathsf{ClosedIntervalNesting}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic closed endpoints, nesting inclusion, shrinking-width scheduling, stream windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalNesting}^{\uparrow}$ packet is generated from dyadic closed endpoints, successor nesting-inclusion evidence, a shrinking-width schedule, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4012.1, Definition 4012.2, and Definition 4012.3 seed the L10 nested-interval route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, interval certificate rows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction, trichotomy, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exactness, nesting-inclusion stability, shrinking-width exactness, StreamName window exactness, RegSeqRat handoff, Real-seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.ClosedIntervalNestingUp.

Chapter 3034

A Concrete Naming Certificate for $\mathsf{HyperconvexMetric}^{\uparrow}$

Following Aronszajn and Panitchpakdi, the $\mathsf{HyperconvexMetric}^{\uparrow}$ packet records the classical ball-intersection surface as finite BEDC completion-neighbour data. It sits over the $\mathsf{Met}^{\uparrow}$ source of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ boundary of chapter 128, the $\mathsf{MetricClosedBall}^{\uparrow}$ membership row of chapter 1451, the $\mathsf{FormalBallBasis}^{\uparrow}$ refinement route of chapter 3943, and the terminal $\mathsf{Real}^{\uparrow}$ radius seal of chapter 38. The packet names pairwise-compatible displayed balls and a finite intersection handoff; it does not select a global hull point, quotient completion representative, arbitrary family intersection, choice function, or ambient real-completeness principle.

Definition 3034.1 (Hyperconvex metric carrier)**.** *A* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, K, B, F, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the inspected point classifier and distance requests.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion-neighbour seal read over the same metric source.*
3. *B is the finite* $\mathsf{MetricClosedBall}^{\uparrow}$ *family row, listing displayed centres, nonnegative radii, and membership comparison rows.*
4. *F is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *refinement row that records the finite centre-radius basis used before the closed-ball family is inspected.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius and distance seal row, including the terminal real-name comparisons needed by B.*
6. *I is the pairwise-intersection ledger. For every displayed pair of balls in B, it records the finite compatibility inequality $d(c_i, c_j) \leq r_i + r_j$ through M and R.*
7. *E is the ball-intersection handoff row. It records the finite witness request that a common displayed intersection point may be consumed only after M, K, B, F, R, I have been read.*
8. *H records componentwise* hsame *transport for the metric source, complete-metric seal, closed-ball family, formal-ball refinement, real-radius seal, pairwise ledger, handoff row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from metric distance requests through formal-ball refinement, closed-ball membership, pairwise compatibility, and the completion-neighbour handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperconvexMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite hyperconvex-metric packet. It has no coordinate for a selected injective hull, arbitrary ball-family intersection, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HyperconvexMetricUp.HyperconvexMetricTasteGate single carrier alignment`

Theorem 3034.2 (Hyperconvex metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier* $Q = (M, K, B, F, R, I, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed metric source, complete-metric seal, closed-ball family, formal-ball refinement, real-radius seal, pairwise-intersection ledger, handoff row, transport, replay, provenance, and local naming rows;*
2. *common-source discipline, requiring K, B, F, R, and I to be read over the same* $\mathsf{Met}^{\uparrow}$ *row M;*
3. *closed-ball visibility, requiring every radius and membership request to pass through* $\mathsf{MetricClosedBall}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *before the intersection handoff is exposed;*
4. *basis refinement, requiring the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row to display the centre-radius refinement route used by the closed-ball family;*
5. *pairwise compatibility, requiring every displayed pair of balls to carry the finite inequality ledger I before a common-intersection request reaches E;*
6. *completion-neighbour nonescape, excluding selected injective hulls, arbitrary family intersections, quotient completion equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

No hyperconvex metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 4013.1. The projection exposes exactly the metric source M, complete-metric seal K, closed-ball family B, formal-ball refinement F, real-radius seal R, pairwise ledger I, ball-intersection handoff E, and structural rows H, C, P, N.

The common-source and closed-ball obligations are tied to the same displayed route. The rows K, B, F, R, I are transported componentwise by H, while the replay row C allows radius and membership comparisons only after the source metric and formal-ball basis rows have been displayed.

The intersection obligation is finite. The ledger I lists the pairwise compatibility inequalities for the displayed balls in B, and the handoff row E can be consumed only after that ledger has been read.

The nonescape clauses follow from the carrier shape. None of the rows stores a selected injective hull, arbitrary ball-family intersection, quotient completion equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness, so the listed obligations exhaust the local NameCert surface. □

Theorem 3034.3 (Hyperconvex metric ball-intersection handoff)**.** *Every completion-neighbour consumer of an accepted* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier* $Q = (M, K, B, F, R, I, E, H, C, P, N)$ *reads the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto I \longmapsto E \longmapsto \mathsf{CompleteMetric}^{\uparrow}$$

before it treats the packet as a ball-intersection source. Thus the consumer receives a displayed metric carrier, formal-ball refinements, closed-ball membership rows, real radius comparisons, and pairwise compatibility evidence, while selected injective hulls, arbitrary ball-family intersections, quotient completion representatives, choice principles, and ambient real completeness remain outside the packet.

Proof. Begin with the metric source. The carrier admits the completion-neighbour read only after M displays the point and distance surface and F displays the finite centre-radius refinement route.

Read the closed-ball and real-radius rows next. The row B supplies the displayed closed balls, while R supplies the terminal real-name comparisons for their radii and distances.

Now inspect pairwise compatibility. The ledger I records the finite inequalities for the displayed family, so the handoff row E is reached only after each required pairwise comparison has been exposed.

Finally the complete-metric seal K and structural rows H, C, P, N package the same route for downstream consumers. Since the carrier has no hull-selection, arbitrary-family, quotient-completion, choice, or ambient-completeness coordinate, the handoff exports none of those objects. □

Falsifiable prediction: Every accepted $\mathsf{HyperconvexMetric}^{\uparrow}$ completion-neighbour read factors through displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, pairwise compatibility, and $\mathsf{CompleteMetric}^{\uparrow}$ rows before any ball-intersection handoff is exposed.

Independence witness: metric, completemetric, metricclosedball, formal-ball-basis, real: $\mathsf{HyperconvexMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds pairwise-compatible closed-ball families, formal-ball refinement, real-radius comparison, complete-metric sealing, ball-intersection handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperconvexMetric}^{\uparrow}$ packet is generated from a displayed MetricSpace row, CompleteMetric seal, MetricClosedBall family, FormalBallBasis refinement, Real radius and distance seals, pairwise compatibility ledger, ball-intersection handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4013.1, Definition 4013.2, and Definition 4013.3 seed the finite ball-intersection completion-neighbour route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected injective hulls, arbitrary ball-family intersections, quotient completion equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-source discipline, closed-ball visibility, basis refinement, pairwise compatibility, and ball-intersection handoff for BEDC.Derived.HyperconvexMetricUp.

Chapter 3035

A Concrete Naming Certificate for SummableSequence$^{\uparrow}$

Following Hardy, Knopp, and Bishop–Bridges constructive real-series analysis, the SummableSequence$^{\uparrow}$ packet records the finite route by which a real-valued sequence exposes a Cauchy tail budget before any Real or Completion consumer reads its sum. It sits after the finite partial-sum surface of RealSeries$^{\uparrow}$ in chapter 1581, the absolute and uniform convergence handoffs of chapter 1700 and chapter 771, the regular rational readback of chapter 301, the stream-window route of chapter 126, and the terminal Real$^{\uparrow}$ seal of chapter 38. The packet names summability as finite certificate data: a source series, partial-sum windows, Cauchy tail budgets, convergence-test handoff, readback, sealing, transport, replay, provenance, and local naming.

Definition 3035.1 (Summable sequence carrier)**.** *A* SummableSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, T, B, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* RealSeries$^{\uparrow}$ *source row. It exposes the summand stream and the partial-sum construction only through the finite real-series surface.*
2. *W is the partial-sum window row. It records finite windows of summands, partial sums, and requested tail cutoffs.*
3. *T is the Cauchy tail budget row. For every displayed tolerance request it records a cutoff, two later partial-sum windows, and the dyadic comparison certifying that their difference lies below the requested bound.*
4. *B is the convergence-test handoff row. It is read either through an* AbsoluteConvergence$^{\uparrow}$ *majorant route or through a* UniformConvergenceTest$^{\uparrow}$ *route when the summability claim is supplied by a uniform family; if neither handoff is displayed, B carries only the local Cauchy tail budget.*
5. *G is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same partial-sum windows and tail budgets.*
6. *R is the* StreamName$^{\uparrow}$ *observation row for summands, partial sums, cutoffs, tail comparisons, and endpoint-facing reads.*
7. *E is the terminal* Real$^{\uparrow}$ *seal reached only after A, W, T, B, G, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, windows, tail budget, convergence handoff, readback, stream observations, Real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay of the route*

$$A, W, T, B \longmapsto G, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{SummableSequence}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{UniformConvergenceTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, T, B, G, R, E, H, C, P, N$. It has no coordinate for a selected infinite sum, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external convergence theorem not displayed by the handoff row.*

Lean4 ✓ `BEDC.Derived.SummableSequenceUp.SummableSequenceTasteGate single carrier alignment`

Theorem 3035.2 (Summable sequence NameCert obligations)**.** *For every accepted* $\mathsf{SummableSequence}^{\uparrow}$ *carrier*

$$S = (A, W, T, B, G, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the eleven displayed rows;*
2. $\mathsf{RealSeries}^{\uparrow}$ *source exposure, requiring every summand and partial-sum read to pass through A;*
3. *partial-sum window exactness, requiring W to bind every inspected summand, partial sum, and tail cutoff to one finite window;*
4. *Cauchy tail budget exactness, requiring T to pair every tolerance request with a cutoff and later-window comparison;*
5. *convergence-test handoff compatibility, requiring B to expose only the displayed* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *or* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *route when such a route is used;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback compatibility, requiring G and R to inspect the same partial-sum and tail-budget rows;*
7. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E to remain downstream of A, W, T, B, G, R;*
8. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No summability obligation is stored outside these rows.

Proof. Project the accepted packet $S = (A, W, T, B, G, R, E, H, C, P, N)$. The source, window, tail-budget, convergence-handoff, readback, stream-observation, Real-seal, and structural rows are exactly the displayed coordinates of the carrier.

The source and window obligations come first. The row A supplies only $\mathsf{RealSeries}^{\uparrow}$ summands and partial sums, while W fixes the finite windows in which tail cutoffs and later partial sums are inspected.

The Cauchy tail budget is the row T. It pairs each tolerance request with a cutoff and the finite comparisons between later partial sums. The handoff row B may justify that budget through the displayed absolute or uniform convergence route, but it cannot import an unlisted convergence theorem.

Finally G, R, E, H, C, P, N read back, observe, seal, transport, replay, source, and name that same route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3035.3 (Summable sequence Real-series handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{SummableSequence}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{RealSeries}^{\uparrow}, \mathsf{AbsoluteConvergence}^{\uparrow}, \mathsf{UniformConvergenceTest}^{\uparrow} \longmapsto A, W, T, B \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto E,$$

The handoff states that summability is consumed through displayed partial-sum windows and Cauchy tail budgets before the $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal are exposed. It exports no selected infinite sum, quotient stream equality, host real equality, countable-choice selector, ambient completeness theorem, or convergence principle detached from the finite packet.*

Proof. Start from the $\mathsf{RealSeries}^{\uparrow}$ row A. It supplies the summand stream and partial sums, while W records the finite partial-sum windows and tail cutoffs available to the consumer.

Read the tail budget through T and the optional convergence-test handoff through B. The absolute-convergence or uniform-convergence route can be consumed only when it is displayed in B, and the local Cauchy tail comparison remains the row that later readback uses.

The rows G and R expose the same partial-sum and tail-budget data as regular rational readback and stream observations. The Real seal E is therefore downstream of the finite Cauchy tail route.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. Because the carrier contains no selected infinite sum, quotient equality, host equality, choice, ambient completeness, or detached convergence coordinate, every Real or Completion handoff stays inside the displayed route. □

Falsifiable prediction: An accepted $\mathsf{SummableSequence}^{\uparrow}$ read is refuted if a Real or Completion consumer exports a Real seal without displaying a RealSeries source row, partial-sum windows, Cauchy tail budget, convergence-test handoff when used, RegSeqRat readback, StreamName observations, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, absolute-convergence, uniform-convergence-test, regseqrat, streamname, real: none alone is bijective with $\mathsf{SummableSequence}^{\uparrow}$ because the packet jointly carries source partial sums, Cauchy tail budgets, convergence-test handoff, regular readback, stream observations, Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{SummableSequence}^{\uparrow}$ packet is generated from a RealSeriesUp source row, partial-sum window rows, Cauchy tail-budget rows, AbsoluteConvergenceUp or UniformConvergenceTestUp handoff rows when displayed, RegSeqRatUp readback, StreamNameUp observations, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4014.1, Definition 4014.2, and Definition 4014.3 seed the finite summability certificate surface from real-series partial sums through Cauchy tail budgets to regular readback and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SummableSequenceUp.SummableSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected infinite sums, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, detached convergence principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum window exactness, Cauchy tail-budget stability, convergence-test handoff compatibility, RegSeqRat readback, StreamName observation compatibility, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.SummableSequenceUp.

Chapter 3036

A Concrete Naming Certificate for $\mathsf{CompactLebesgueNumber}^{\uparrow}$

Following Lebesgue, Heine–Borel compactness, and Bishop–Bridges compact metric analysis, the $\mathsf{CompactLebesgueNumber}^{\uparrow}$ packet records the compact finite cover-radius ledger used by the compact uniform-continuity proof of chapter 9237. It refines the compact-metric source of chapter 129, the finite positive-radius ledger of chapter 452, and the uniform-modulus consumer of chapter 369. The packet is a finite NameCert surface for compact finite nets, located cover radii, lower-bound folds, and uniform-modulus handoff; it is not open-cover compactness, a selected subcover, or an external minimum principle.

Definition 3036.1 (Compact Lebesgue-number carrier)**.** *A* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, B, C, R, F, U, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes compact finite-net and metric inspection data through the compact-metric chapter.*
2. *B is the finite* ProbeBundle *cover spine naming the compact cover candidates inspected by the packet.*
3. *C is the center and coverage ledger. It records center histories, located-ball membership rows, and the finite cover witness for every compact-source history read by the packet.*
4. *R is the positive cover-radius row, read through dyadic and rational-ball radius data attached to the bundled probes.*
5. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-radius row. It folds the finite radius ledger to a positive radius below every displayed cover radius.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row. It exports the lower-radius ledger only after K, B, C, R, F have been displayed.*
7. *H records componentwise* hsame *transport for the compact source, bundle spine, coverage ledger, radius rows, finite Lebesgue-number row, uniform-modulus handoff, replay, provenance, and local name.*
8. *Q records displayed* Cont *replay of the route*

$$K, B, C, R \longmapsto F \longmapsto U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, C, R, F, U, H, Q, P, N$. It has no coordinate for an arbitrary open cover, a selected finite subcover, host compactness, a host minimum, quotient real equality, countable choice, or a uniform modulus detached from the finite lower-radius row.

Lean4 ✓ `BEDC.Derived.CompactLebesgueNumberUp.CompactLebesgueNumberTasteGate single carrier alignment`

Theorem 3036.2 (Compact Lebesgue-number NameCert obligations)**.** *For every accepted* CompactLebesgueNumb *carrier*

$$L = (K, B, C, R, F, U, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the ten displayed rows;*
2. *compact-metric source exposure, requiring every compact-source read to pass through K;*
3. *finite ProbeBundle coverage exactness, requiring B and C to bind every compact-source history to a named cover probe and located-ball row;*
4. *positive radius ledger exactness, requiring R to attach displayed dyadic and rational-ball radii to the bundled probes;*
5. *finite Lebesgue-number fold compatibility, requiring F to be a* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *row below every displayed radius in R;*
6. *uniform-modulus handoff, requiring U to consume only the lower-radius row and the compact source route already displayed;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, Q, P, N.*

No compact Lebesgue-number obligation is stored outside these rows.

Proof. Project the accepted packet $L = (K, B, C, R, F, U, H, Q, P, N)$. The compact source, bundle, coverage, radius, lower-fold, uniform handoff, and structural rows are precisely the displayed carrier coordinates.

The compact and coverage obligations come first. The row K supplies the compact-metric source, while B and C expose only named probes, center histories, located-ball reads, and finite cover witnesses.

The radius obligations are finite. The row R records the positive radii attached to bundled probes, and F records the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ lower-radius ledger below those radii. No host minimum or selected open subcover is needed to read F.

Finally U, H, Q, P, N hand the lower radius to uniform-modulus consumers and preserve, replay, source, and name the same route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3036.3 (Compact Lebesgue-number uniform-modulus handoff)**.** *Every compact-uniform-continuity or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto B, C, R \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto H, Q, P, N.$$

Thus the finite cover-radius ledger of chapter 9237 is exposed as compact-metric source data, named cover probes, positive radius rows, a finite Lebesgue-number lower-bound fold, and a uniform-modulus handoff. It exports no arbitrary open cover, selected subcover, host compactness theorem, host minimum, countable-choice selector, quotient real equality, or detached uniform modulus.

Proof. Start from the compact-metric source K. It provides the finite compact inspection data, while B and C record the actual cover spine and located coverage rows consumed by the packet.

Read the radius ledger R and the finite lower-bound row F. The lower-bound row is a $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ packet tied to the displayed probe radii, so it is not a host minimum and not a selected subcover argument.

The handoff row U sends exactly this lower-radius ledger to the $\mathsf{UniformModulus}^{\uparrow}$ consumer. The transport, replay, provenance, and naming rows H, Q, P, N preserve the same path.

Because the carrier has no coordinate for an arbitrary open cover, selected subcover, host compactness theorem, host minimum, choice, quotient real equality, or detached modulus, every compact-uniform-continuity read factors through the displayed route. □

Theorem 3036.4 (Compact Lebesgue-number Real-uniform entourage route)**.** *Every* $\mathsf{RealUniformStructure}^{\uparrow}$ *or* $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *packet must read the* $\mathsf{CompactMetric}^{\uparrow}$ *source,* $\mathsf{ProbeBundle}$ *cover spine, located coverage rows, positive radius ledger,* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-bound fold, and* $\mathsf{UniformModulus}^{\uparrow}$ *handoff before any Real-induced entourage basis is exported. The route supplies no arbitrary open cover, selected subcover, host minimum, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier $L = (K, B, C, R, F, U, H, Q, P, N)$. The source row K, cover spine B, located coverage ledger C, positive radius row R, finite lower-bound fold F, and uniform handoff row U are the only analytic coordinates available before the structural rows.

Apply Definition 4015.3. It forces every compact-uniform-continuity read to pass through the displayed compact source, finite cover spine, located coverage rows, positive radii, lower-bound fold, and $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Route the resulting lower-bound ledger through the Real metric-ball surface of chapter 1624 and the basis packet of chapter 5115. A Real-induced entourage basis can therefore be exported only as a downstream read of the same compact finite-radius route.

Because arbitrary open covers, selected subcovers, host minima, quotient real equality, countable choice, and ambient Real completeness are absent from the carrier coordinates, no Real-uniform consumer can obtain them from the exported entourage basis. □

Falsifiable prediction: An accepted $\mathsf{CompactLebesgueNumber}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a UniformModulusUp row without displaying CompactMetric source data, a ProbeBundle cover spine, located coverage rows, positive radius rows, a FiniteLebesgueNumber lower-bound fold, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, finite-lebesgue-number, uniformmodulus: none alone is bijective with $\mathsf{CompactLebesgueNumber}^{\uparrow}$ because the packet jointly carries compact source data,

finite cover probes, located coverage rows, positive radii, lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}$.

Constructive story: A $\mathsf{CompactLebesgueNumber}^{\uparrow}$ packet is generated from CompactMetricUp source data, a finite ProbeBundle cover spine, located coverage rows, positive dyadic and rational-ball radius rows, a FiniteLebesgueNumberUp lower-bound fold, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4015.1, Definition 4015.2, and Definition 4015.3 seed the compact finite Lebesgue-number packet from compact-metric cover data through finite lower-radius folds to UniformModulusUp handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host compactness, host minima, quotient real equality, countable choice, detached uniform moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source exposure, ProbeBundle coverage exactness, positive-radius ledger stability, FiniteLebesgueNumber fold compatibility, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactLebesgueNumberUp.

Chapter 3037

A Concrete Naming Certificate for $\mathsf{CompactPolishSpace}^{\uparrow}$

Following Alexandrov and Urysohn on Polish spaces and Kelley on compact metric spaces, the $\mathsf{CompactPolishSpace}^{\uparrow}$ packet records the finite BEDC surface where compact-metric data and complete separable metric data are read as one named object. It sits over the $\mathsf{CompactMetric}^{\uparrow}$ chapter of chapter 129, the $\mathsf{PolishSpace}^{\uparrow}$ chapter of chapter 1490, the $\mathsf{CompleteMetric}^{\uparrow}$ route of chapter 128, the $\mathsf{SeparableMetric}^{\uparrow}$ route of chapter 1458, the finite observation windows of chapter 126, and the regular rational readback surface of chapter 301. The packet is a synthesis node for compact-completion and continuous-map consumers; it is not an open-cover compactness theorem, a host completion, a countable-choice selector, or a quotient of dense names.

Definition 3037.1 (Compact Polish space carrier)**.** *A* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (K, P, C, S, W, R, H, T, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row carrying the finite compact-metric witness, including the metric row, total-boundedness window, and compact-test handoff.*
2. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row naming the same metric carrier as a complete separable metric surface.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row through which Cauchy-facing consumers read completeness data before any Real or completion seal is inspected.*
4. *S is the* $\mathsf{SeparableMetric}^{\uparrow}$ *row displaying the finite dense-name schedule available to separability consumers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row used by compact-net, dense-name, and completion-facing reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row consumed after the finite stream window has been displayed.*
7. *H records componentwise* hsame *transport for the compact-metric, Polish-space, complete-metric, separable-metric, stream-window, regular-readback, package, and local naming rows.*
8. *T records displayed* Cont *replay from compact-metric witnesses and dense schedules through complete separable readback.*

9. *Q is the* Pkg *provenance over* $\mathsf{CompactPolishSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, host compactness, host completion, selected dense enumerations, quotient Cauchy classes, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.CompactPolishSpaceUp.TasteGate.CompactPolishSpaceUp`

Theorem 3037.2 (Compact Polish space NameCert obligations)**.** *For every accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier $X = (K, P, C, S, W, R, H, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$ *surface consists exactly of compact-metric admission, Polish-space admission, complete-metric handoff, separable-metric handoff, finite stream-window exposure, regular-rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must read the compact side through K, the Polish side through P, C, S, and the finite observation side through W, R before using the packet as a compact Polish space.*

Lean4 ✓ `BEDC.Derived.CompactPolishSpaceUp.CompactPolishSpaceCarrier namecert obligations`

Proof. Project the carrier of Definition 4016.1. The tuple lists the only rows visible to the local classifier, so carrier admission is confined to $K, P, C, S, W, R, H, T, Q, N$.

The compact side is supplied by K. It gives finite compact-metric evidence, but no open-cover compactness principle or ambient compact object.

The Polish side is supplied by P through the complete and separable rows C and S. Completeness-facing reads therefore use the displayed complete-metric handoff, and separability-facing reads use the displayed dense-name schedule.

The observation and readback rows W and R bind those two sides to the same finite stream window and regular-rational surface. Transport, replay, provenance, and local naming are then componentwise through H, T, Q, N, leaving no coordinate for a host completion, countable-choice selector, or quotient dense-name equality. □

Theorem 3037.3 (Compact Polish finite-cover readback)**.** *Every compact-completion or continuous-map consumer of an accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{SeparableMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

through the displayed rows $K, P, C, S, W, R, H, T, Q, N$. The exported readback is the compact-metric witness together with complete separable metric data and finite regular-name observations; it exports no open-cover compactness theorem, selected host completion, countable-choice dense enumeration, quotient Cauchy class, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.CompactPolishSpaceUp.CompactPolishSpaceCarrier readback certificate`

Proof. Start from the obligation surface of Definition 4016.2. A consumer first reads the compact-metric row K and the Polish-space row P as the two named views of the same carrier.

The complete and separable handoffs are then forced through C and S. Thus completion-facing reads see a complete-metric row, while dense-name reads see a separable-metric row, and neither read can introduce a host completion or selected dense enumeration.

Finally the finite observation route passes through W and R. The structural rows H, T, Q, N preserve the same route by transport, replay, provenance, and local naming, so the displayed rows exhaust the handoff available to compact-completion and continuous-map consumers. □

Theorem 3037.4 (Compact Polish compact-complete-separable synthesis)**.** *Every accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *consumer that asks for compact-completion or continuous-map data must factor through the single finite route*

$$K, P, C, S, W, R, H, T, Q, N.$$

Thus compactness, completeness, separability, stream windows, and regular-rational readback are consumed as one carrier surface and cannot be read independently. The route exports no open-cover compactness, host completion, selected dense enumeration, quotient Cauchy class, countable choice, or ambient completeness coordinate.

Proof. Project the carrier rows in Definition 4016.1. The compact-metric row K, Polish-space row P, complete-metric row C, separable-metric row S, stream row W, and regular-rational row R are displayed before any structural replay can be used.

Apply the obligation surface of Definition 4016.2. It identifies the local NameCert read as exactly $K, P, C, S, W, R, H, T, Q, N$, so compactness, completeness, separability, finite observation, and readback are bound to the same packet.

Finally use finite-cover readback from Definition 4016.3. Compact-completion and continuous-map consumers therefore follow the displayed route, and no coordinate exists for open covers, host completions, selected dense enumerations, quotient Cauchy classes, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactPolishSpace}^{\uparrow}$ consumer read factors through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows before it is used as compact Polish data.

Independence witness: compactmetric, polishspace, completemetric, separablemetric, streamname, regseqrat: $\mathsf{CompactPolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds compact-metric witnesses and complete separable metric rows through one finite observation and local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactPolishSpace}^{\uparrow}$ packet is generated from $\mathsf{CompactMetric}^{\uparrow}$ compact-metric rows, PolishSpaceUp complete separable rows, $\mathsf{CompleteMetric}^{\uparrow}$ handoff rows, SeparableMetricUp dense-name rows, StreamNameUp windows, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4016.1, Definition 4016.2, and Definition 4016.3 seed the finite compact Polish synthesis node over compact-metric and complete separable metric data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, host completion, selected dense enumerations, quotient Cauchy classes, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric admission, Polish-space admission, complete-metric handoff, separable-metric handoff, finite observation readback, transport, replay, provenance, and local naming for BEDC.Derived.CompactPolishSpaceUp.

Chapter 3038

A Concrete Naming Certificate for $\mathsf{PartitionCoreIncidence}^{\uparrow}$

The $\mathsf{PartitionCoreIncidence}^{\uparrow}$ packet realizes the partition-core incidence representation surface of Definition 9462.1755 as finite BEDC NameCert data. It sits over the partition rows of chapter 201, the graph incidence rows of chapter 118, the support-nerve surface of chapter 238, the rank-dependence boundary of chapter 202, and the boundary-tile source of chapter 3393. The packet names the finite incidence kernel consumed by boundary-tile and representation-facing reads; it is not a host representation module, a quotient linear-algebra oracle, or an unrestricted biological model.

Definition 3038.1 (Partition-core incidence carrier)**.** *A* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, G, S, M, B, K, V, T, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the partition-core row naming the finite partition blocks and the accepted core labels.*
2. *G is the graph-incidence row listing the displayed source vertices, target cores, and accepted incidence edges.*
3. *S is the support-nerve row, read through* $\mathsf{SimplicialComplex}^{\uparrow}$*, whose faces record nonempty incidence support.*
4. *M is the* $\mathsf{Matroid}^{\uparrow}$ *dependence row recording which displayed incidence supports are treated as dependent.*
5. *B is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *source row carrying the boundary-tile coordinates that feed the incidence read.*
6. *K is the partition-core kernel row displaying the accepted null combinations of boundary-tile incidence reads.*
7. *V is the representation-basis row recording the finite trivial, standard, signed, and two-dimensional summand labels used by the readback.*
8. *T is the finite transport row from boundary-tile source reads through partition-core incidence, support-nerve, matroid-dependence, and kernel rows.*
9. *R is the replay row recording the accepted provenance of every local incidence read.*

10. *H records componentwise* hsame *transport for all displayed rows.*
11. *C records displayed* Cont *replay through the partition-core, incidence, support-nerve, transport, replay, provenance, and local naming rows.*
12. *Q is the* Pkg *provenance over* $\mathsf{PartitionCoreIncidence}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$ *row.*

The carrier exports only these finite rows. It contains no host representation space, hidden quotient module, unrestricted linear-algebra solver, stochastic biological model, or global codon semantics.

Lean4 ✓ `BEDC.Derived.PartitionCoreIncidenceUp.PartitionCoreIncidenceTasteGate single carrier alignment`

Theorem 3038.2 (Partition-core incidence NameCert obligations)**.** *For every accepted* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *carrier* $I = (P, G, S, M, B, K, V, T, R, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$ *surface consists exactly of partition-core admission, graph-incidence admission, support-nerve exposure, matroid-dependence exposure, boundary-tile source admission, kernel-row readback, representation-basis readback, transport, replay, componentwise* hsame*, displayed* Cont, Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4017.1. The partition, graph, support-nerve, matroid, boundary-tile, kernel, and representation rows are the only analytic coordinates available to the local classifier.

The incidence read starts at B and reaches partition cores only through P and G. Support and dependence are then displayed by S and M, so a consumer sees finite support evidence before it reads kernel membership.

The rows K and V record the finite kernel and representation-basis readback. Transport, replay, componentwise equivalence, continuation replay, provenance, and local naming are exactly T, R, H, C, Q, N, leaving no coordinate for host representation modules, quotient linear algebra, stochastic biological dynamics, or global codon semantics. □

Theorem 3038.3 (Partition-core incidence kernel handoff)**.** *Every accepted* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *read factors through the displayed finite route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{Partition}^{\uparrow}, \mathsf{Graph}^{\uparrow} \longmapsto \mathsf{SimplicialComplex}^{\uparrow}, \mathsf{Matroid}^{\uparrow} \longmapsto K, V$$

together with transport, replay, provenance, and local naming rows. Boundary-tile consumers and representation-facing consumers therefore meet only at the displayed partition-core incidence kernel, not at a host quotient module or unrestricted linear-algebra surface.

Proof. Use Definition 4017.2. The boundary-tile source row B supplies the input coordinates, while P and G expose the partition-core and incidence-edge rows through which those coordinates can be read.

The support and dependence checks then pass through S and M. These rows display the support nerve and the finite dependence boundary before any kernel readback is accepted.

Finally K and V expose the finite kernel and representation-basis rows. The structural rows T, R, H, C, Q, N preserve that same route, so the consumer receives only the displayed finite incidence packet. □

Falsifiable prediction: Every accepted $\mathsf{PartitionCoreIncidence}^{\uparrow}$ read factors through displayed partition-core, incidence, support-nerve, transport, replay, provenance, and local naming rows, and any read lacking one of those rows is rejected by the local NameCert surface.

Independence witness: partition, graph, simplicialcomplex, matroid, dna-window-six-boundary-tile: $\mathsf{PartitionCoreIncidence}^{\uparrow}$ is not bijective to any listed sibling because it binds boundary-tile source rows to partition-core incidence, support-nerve, matroid-dependence, kernel readback, representation-basis, transport, replay, provenance, and local naming as one packet.

Lean follow-up provides $\mathsf{Nontrivial\ PartitionCoreIncidence}^{\uparrow}$ and $\mathsf{FieldFaithful\ PartitionCoreIncidence}^{\uparrow}$ in TasteGate.lean.

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$.

Constructive story: A $\mathsf{PartitionCoreIncidence}^{\uparrow}$ packet is generated from partition-core rows, graph-incidence rows, support-nerve rows, matroid-dependence rows, DNA boundary-tile source rows, kernel readback, representation-basis readback, transport, replay, componentwise hsame, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4017.1, Definition 4017.2, and Definition 4017.3 seed the finite partition-core incidence packet for boundary-tile and representation-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host representation modules, quotient linear algebra, unrestricted biological dynamics, global codon semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-core incidence exposure, support-nerve exposure, matroid-dependence exposure, kernel readback, representation-basis readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionCoreIncidenceUp.

Chapter 3039

A Concrete Naming Certificate for $\mathsf{BaireModulus}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bishop, the $\mathsf{BaireModulus}^{\uparrow}$ packet records the finite-prefix modulus surface by which a Baire-space read exposes a bounded inspection window before any real-name or fan-functional consumer sees a value. It sits between the finite prefix discipline of chapter 334, the stream-window route of chapter 126, the unary index surface of $\mathsf{Nat}^{\uparrow}$, the dyadic and regular rational handoff of chapter 291 and chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the finite modulus-extraction consumer of chapter 1525. The packet is not host sequence equality, not a selected infinite branch, not a quotient stream object, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3039.1 (Baire modulus carrier)**.** *A* $\mathsf{BaireModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, W, \mu, R, H, C, P, N)$$

with the following displayed rows:

1. *B is a* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix packet, naming the inspected stream schedule and the prefix-cylinder classifier.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row from which the requested stream coordinates are read.*
3. *W is a unary* $\mathsf{Nat}^{\uparrow}$ *index window listing the finite requests admitted by the modulus.*
4. *μ is the displayed prefix-modulus row, asserting that every permitted consumer read is determined by the rows of W and no tail outside that window.*
5. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff surface attached to the displayed finite prefix.*
6. *H records componentwise* hsame *transport for the Baire prefix row, stream schedule, unary index window, modulus row, and handoff surface.*
7. *C records displayed* Cont *replay of the same finite prefix requests and handoff rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BaireModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, W, \mu, R, H, C, P, N$. There is no coordinate for host sequence equality, selected branches, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3039.2 (Baire modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireModulus}^{\uparrow}$ *carrier $M = (B, S, W, \mu, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, prefix-modulus exactness, finite-window transport, handoff provenance, and nonescape rows. Carrier admission exposes the* $\mathsf{BaireSpace}^{\uparrow}$ *packet,* $\mathsf{StreamName}^{\uparrow}$ *schedule, unary* $\mathsf{Nat}^{\uparrow}$ *window, displayed prefix modulus,* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. Prefix-modulus exactness requires every public read to factor through the listed finite window W before the handoff row R or a* $\mathsf{FanFunctional}^{\uparrow}$ *consumer is visible. The nonescape rows exclude host sequence equality, selected branches, quotient streams, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireModulusUp.BaireModulusUpTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 4018.1. The only stream data available are the $\mathsf{BaireSpace}^{\uparrow}$ finite prefix row B, the $\mathsf{StreamName}^{\uparrow}$ schedule S, and the unary index window W.

The row μ is read after those rows. It binds consumer visibility to the displayed prefix window, so the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FanFunctional}^{\uparrow}$ handoffs can only consume prefix data already listed in W.

Transport and replay do not add data. The componentwise hsame rows in H and the displayed Cont routes in C reread the same finite requests, while P and N only record provenance and local naming. Since no excluded object is a carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3039.3 (Baire modulus prefix handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{FanFunctional}^{\uparrow}$*-facing public read of an accepted* $\mathsf{BaireModulus}^{\uparrow}$ *packet factors through the route*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Nat}^{\uparrow} \longmapsto \mathsf{BaireModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *row. The handoff exports a finite prefix modulus and no host sequence equality, branch choice, quotient stream, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with Definition 4018.2. Carrier admission opens the Baire prefix packet, stream schedule, unary window, and modulus row before any downstream consumer can inspect the handoff surface.

Read the handoff row R. Its dyadic, regular-rational, and real-name coordinates are attached to the finite window selected by W and bounded by μ, so they cannot request a tail coordinate outside the displayed prefix.

Finally pass the same finite route to the $\mathsf{FanFunctional}^{\uparrow}$ consumer. The consumer receives the displayed modulus row, transported by H and replayed by C, with provenance in P and local naming in N. Because the carrier has no host equality, selected branch, quotient stream, countable-choice, or real-completeness coordinate, the public handoff has exactly the stated finite-prefix form. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4018.1, Definition 4018.2, and Definition 4018.3 seed the finite-prefix modulus route over $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence equality, branch choice, quotient streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-modulus exactness, finite-window transport, handoff provenance, and nonescape for BEDC.Derived.BaireModulusUp.

Constructive story: A $\mathsf{BaireModulus}^{\uparrow}$ packet is generated from BaireSpace finite prefix rows, StreamName schedules, unary Nat index windows, displayed prefix-modulus rows, DyadicRatCore, RegSeqRat, and Real handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3040

A Concrete Naming Certificate for $\mathsf{CompactMetricSequentialNet}^{\uparrow}$

Following Bishop and Bridges constructive compactness, the $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ packet joins compact metric finite-net data with sequential extraction as a finite L10 neighbour of chapter 60, chapter 129, chapter 57, and chapter 126. It reads total-bounded net schedules, complete-metric Cauchy routes, finite stream windows, regular rational readback, and terminal Real seals as displayed NameCert rows. The packet supports Bolzano–Weierstrass and sequential compactness consumers while refusing open-cover compactness, countable choice, host subsequence selection, quotient stream equality, and detached real completeness.

Definition 3040.1 (CompactMetricSequentialNet carrier)**.** *A* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *surface and its compact and metric dependencies.*
2. *T is the total-bounded net schedule, recording finite centers, mesh radii, coverage witnesses, and refinement rows for the requested tolerances.*
3. *L is the complete-metric Cauchy route that receives the extracted Cauchy window after the finite-net schedule has stabilized.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the source sequence, selected indices, selected source windows, and monotonicity evidence.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and dyadic tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the readback and complete-metric route.*
7. *H records componentwise* hsame *transport for the compact metric source, net schedule, Cauchy route, stream windows, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact metric finite nets through sequential windows, regular rational readback, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, T, L, W, R, E, H, C, P, N$. It has no coordinate for an externally chosen subsequence, open-cover compactness, quotient stream equality, countable choice, host equality, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactMetricSequentialNetUp.CompactMetricSequentialNetTasteGate single carrier alignment`

Theorem 3040.2 (CompactMetricSequentialNet NameCert obligations)**.** *For every accepted* $\mathsf{CompactMetricSequen}$ *carrier*

$$S = (K, T, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$ *surface consists exactly of compact metric source admission through K, total-bounded finite-net scheduling through T, complete-metric Cauchy routing through L, finite stream and subsequence windows through W, regular rational readback through R, terminal Real sealing through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 4019.1. The accepted packet exposes exactly the rows $K, T, L, W, R, E, H, C, P, N$.

The compact metric side is finite. The source row K supplies the metric and compact data, while T supplies the total-bounded finite-net schedule used to choose inspected windows at each displayed tolerance.

The sequential side is the row W. It records only finite source windows, selected indices, selected windows, and monotonicity evidence, so no host subsequence selector or countable-choice schedule is available.

The completion-facing side is L, R, E. The selected windows are read through the regular rational row R, then through the complete-metric route L, and finally through the terminal Real seal E.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate outside these rows, the listed non-escape clauses are part of the NameCert surface. □

Theorem 3040.3 (CompactMetricSequentialNet subsequence handoff)**.** *Every compact-metric or Bolzano–Weierstrass consumer of an accepted* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ *packet reads the subsequence handoff through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports the displayed finite-net schedule, selected source windows, regular rational readback, complete-metric Cauchy route, Real seal, and structural rows only; it exports no open-cover compactness theorem, selected infinite subsequence, quotient stream equality, countable choice, host equality, or detached real-completeness principle.

Proof. Start from Definition 4019.2. The compact metric source row and the total-bounded schedule must be displayed before a sequence consumer may inspect any selected window.

Next read the stream row W. The row contains the finite source windows and the selected-window evidence consumed by the downstream regular rational readback, but it does not contain an independent infinite subsequence object.

Then apply R, L, E in order. The regular rational readback is attached to the displayed windows, the complete-metric route receives that Cauchy data, and the Real seal is reached only at the end of the displayed route.

Finally H, C, P, N preserve the same packet. Therefore every public handoff is confined to the route above and cannot export the excluded open-cover, choice, quotient, host-equality, or detached-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ packet is generated from compact metric source rows, total-bounded finite-net schedules, complete-metric Cauchy routes, StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4019.1, Definition 4019.2, and Definition 4019.3 seed the finite compact-metric sequential extraction route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite subsequences, quotient stream equality, countable choice, host equality, detached Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net scheduling, sequential-window monotonicity, RegSeqRat readback, complete-metric routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactMetricSequentialNetUp.

Chapter 3041

A Concrete Naming Certificate for $\mathsf{BorelHierarchy}^{\uparrow}$

Following Kechris and Bourbaki in classical descriptive set theory, the $\mathsf{BorelHierarchy}^{\uparrow}$ packet records the finite BEDC boundary by which a Polish or Borel-facing consumer inspects a displayed hierarchy rank without importing the ambient host universe of all Borel subsets. It sits over the generated-set surface of chapter 3384, the open-schedule boundary of chapter 3547, and the complete separable source of chapter 1490. The packet exposes rank schedules, successor and limit-stage replay, StreamName and RegSeqRat readback, transport, continuation replay, provenance, and local naming; it does not claim classical descriptive-set completeness, selected codes, quotient set equality, countable choice, or measure-theoretic structure.

Definition 3041.1 (Borel hierarchy carrier)**.** *A* $\mathsf{BorelHierarchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, B, G, P, S, R, T, C, N)$$

with the following displayed rows:

1. *X is the source-space row, read through the Polish or topology-facing source visible to the current consumer.*
2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *row that receives generated-set handoff after the hierarchy schedule has been inspected.*
3. *G is either the* $\mathsf{GDeltaSet}^{\uparrow}$ *row or the displayed open-generator row used as the first nontrivial hierarchy surface.*
4. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row when the hierarchy is being read over a complete separable metric presentation.*
5. *S is the countable rank schedule. Each visible stage records the rank tag, predecessor window, successor or limit-stage instruction, and the finite constructor route offered to the consumer.*
6. *R is the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rank positions, predecessor windows, open-generator observations, and real-facing regular rational queries.*
7. *T records componentwise* hsame *transport for the source, Borel row, G-delta or open-generator row, Polish row, rank schedule, readback, continuation, and local naming rows.*
8. *C records displayed* Cont *replay from a source query through the visible rank schedule, successor or limit-stage constructor, generated-set handoff, and readback route.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, B, G, P, S, R, T, C, N$. It has no coordinate for a host subset, a chosen Borel code, countable choice over ranks, quotient set equality, a completed descriptive-set universe, or a measure-theoretic assertion.

Lean4 ✓ `BEDC.Derived.BorelHierarchyUp.BorelHierarchyTasteGate single carrier alignment`

Theorem 3041.2 (Borel hierarchy NameCert obligations)**.** *For every accepted* $\mathsf{BorelHierarchy}^{\uparrow}$ *carrier*

$$H = (X, B, G, P, S, R, T, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *source admission for the displayed source-space row and, when present, the Polish source row;*
2. *rank-schedule exposure, requiring every inspected hierarchy stage to appear in the finite visible prefix of S before any generated-set row is consumed;*
3. *successor-stage replay, requiring successor constructors to read their predecessor window, constructor tag, StreamName observation, and RegSeqRat readback through S and R;*
4. *limit-stage replay, requiring limit constructors to expose only the displayed finite prefix, schedule position, and readback windows visible in S and R;*
5. *generated-set handoff, requiring the* $\mathsf{BorelSet}^{\uparrow}$ *row B to consume the visible route through X, G, P, S, R;*
6. *transport, requiring T to preserve source, Borel, G-delta or open-generator, Polish, rank-schedule, readback, replay, and naming rows together;*
7. *continuation replay, requiring C to replay the same ordered route before any Borel, G-delta, Polish, StreamName, or RegSeqRat consumer receives the packet;*
8. *provenance, requiring every cited sibling surface to be read only through the displayed rows of chapter 3384, chapter 3547, and chapter 1490;*
9. *local naming, requiring N to certify exactly the rows X, B, G, P, S, R, T, C;*
10. *refusal of host subsets, selected codes, countable choice, quotient set equality, ambient descriptive-set completeness, and measure-theoretic claims.*

No $\mathsf{BorelHierarchy}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4020.1. The tuple displays exactly the source row, Borel row, G-delta or open-generator row, Polish row, countable rank schedule, finite readback, transport, replay, and local naming rows.

Rank exposure is read through S and R. A successor stage exposes its predecessor window and constructor tag through the visible schedule, while a limit stage exposes only the finite prefix, position, and readback windows visible to the current consumer.

The generated-set handoff then passes to B over the same X, G, P, S, R route. The sibling chapters supply the only accepted surfaces for the Borel row, the G-delta or open-generator row, and the Polish source.

Finally T, C, N transport, replay, and name the displayed route. Since no row stores a host subset, selected code, countable-choice selector, quotient set equality, completed descriptive-set universe, or measure-theoretic datum, the enumerated obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4020.1 and Definition 4020.2 seed the hierarchy route over source, Borel, G-delta or open-generator, Polish, rank-schedule, readback, transport, replay, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, selected Borel codes, countable choice, quotient set equality, ambient descriptive-set completeness, measure theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rank-schedule exposure, successor and limit-stage replay, generated-set handoff, transport, replay, provenance, local naming, and ledger refusal for BEDC.Derived.BorelHierarchyUp.

Constructive story: A $\mathsf{BorelHierarchy}^{\uparrow}$ packet is generated from a source-space row, BorelSet row, GDeltaSet or open-generator row, Polish source row, countable rank schedule, finite StreamName and RegSeqRat readback, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Chapter 3042

A Concrete Naming Certificate for RegularSequenceLocalization$^{\uparrow}$

Following Bishop and Bridges' regular Cauchy presentation of constructive reals and Kelley's finite-local analysis of neighborhoods, the RegularSequenceLocalization$^{\uparrow}$ packet records a bounded observation window around a displayed regular Cauchy name as first-class NameCert data. It reads finite stream observations through chapter 126, measures the local radius through chapter 291, replays regular rational evidence through chapter 301, and reaches the terminal real seal of chapter 38. The packet is local: it exposes the center, dyadic radius ledger, finite observation windows, rational readback, seal ordering, transport, replay, provenance, and local name before any Real or Completion consumer may use the localized estimate.

Definition 3042.1 (Regular-sequence localization carrier)**.** *A* RegularSequenceLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (a, \rho, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *a is the finite localization center row, naming the regular Cauchy datum around which the local estimate is inspected.*
2. *ρ is the* DyadicRatCore$^{\uparrow}$ *radius ledger, recording the dyadic scale, comparison cells, and slack rows used by the local window.*
3. *W is the finite* StreamName$^{\uparrow}$ *observation-window row, naming exactly the stream positions read at the requested local precision.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, converting the displayed center, radius ledger, and window observations into regular rational estimate data.*
5. *R is the terminal* Real$^{\uparrow}$ *seal row, available only after a, ρ, W, Q have been replayed.*
6. *H records componentwise* hsame *transport for the center, radius ledger, window observations, regular rational readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a local estimate request through a, ρ, W, Q, and then R.*
8. *P is the* Pkg *provenance over* RegularSequenceLocalization$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyEntourageBasis$^{\uparrow}$, RegularCauchyLimitUniqueness$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted localization packets only through $a, \rho, W, Q, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, ambient real completeness, or a global neighborhood system beyond the displayed finite window.

Lean4 ✓ `BEDC.Derived.RegularSequenceLocalizationUp.RegularSequenceLocalizationTasteGate single carrier alignment`

Theorem 3042.2 (Regular-sequence localization NameCert obligations)**.** *For every accepted* RegularSequenceLocal *carrier $L = (a, \rho, W, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the localization center, dyadic radius ledger, finite stream observation windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *radius-ledger stability, requiring every local estimate to read ρ through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale and not through a host real order;*
3. *window-read stability, requiring every accepted observation to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is available;*
4. *readback exactness, requiring the route $a, \rho, W \longmapsto Q \longmapsto R$ before any Real or Completion consumer treats the estimate as localized regular-sequence data;*
5. *transport and replay stability for the same a, ρ, W, Q, R, P, N route under* hsame *transport and* Cont *replay.*

No localization NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4021.1. Its mathematical rows are exactly the center a, dyadic radius ledger ρ, finite window row W, regular rational readback Q, and real seal R.

The radius and window obligations are read before the rational handoff. The ledger ρ supplies only dyadic comparison data, and W supplies only the finite stream observations displayed by the packet.

The readback obligation is then forced through Q. Since Q is the sole $\mathsf{RegSeqRat}^{\uparrow}$ coordinate, the packet cannot replace the local estimate by quotient stream equality, a selected limit, countable choice, trichotomy, or ambient completeness.

Finally the rows H, C, P, N transport, replay, source, and name the same route. They do not add analytic coordinates, so the listed rows exhaust the local NameCert surface. □

Theorem 3042.3 (Regular-sequence localization window stability)**.** *Let $L = (a, \rho, W, Q, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceLocalization}^{\uparrow}$ *carrier. If a Real or Completion consumer requests a localized regular-sequence estimate, then every accepted replay factors through the finite window route*

$$a, \rho, W \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Consequently the displayed dyadic radius ledger and finite $\mathsf{StreamName}^{\uparrow}$ *observation window remain stable under* hsame *transport and* Cont *replay before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Begin with a localized estimate request. The carrier offers the center row a, the dyadic radius ledger ρ, and the finite window row W as the only rows available before regular rational readback.

Replay through C. The replay opens a, then reads the dyadic comparison cells in ρ, then inspects the finite observations named by W, and only then exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q.

Transport through H preserves this route componentwise. The real seal R and the structural rows P, N are downstream of the same displayed path, so no replay can consume $\mathsf{Real}^{\uparrow}$ data before the local window and radius ledger have been shown. □

Falsifiable prediction: Every accepted $\mathsf{RegularSequenceLocalization}^{\uparrow}$ read used by a Real or Completion consumer displays a finite center row, one $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, one finite $\mathsf{StreamName}^{\uparrow}$ observation window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and one local structural NameCert route before the estimate is treated as localized regular-sequence data.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularSequenceLocalization}^{\uparrow}$ is not bijective to any listed sibling because it binds a localization center, radius ledger, finite observation windows, regular rational readback, terminal real seal, transport, replay, provenance, and local NameCert row into one local estimate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceLocalization}^{\uparrow}$ packet is generated from a finite localization center, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, finite $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4021.1, Definition 4021.2, and Definition 4021.3 seed the finite local regular-sequence estimate route over center, radius, window, readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global real completeness, quotient stream equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger stability, finite-window stability, RegSeqRat readback exactness, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceLocaliza

Chapter 3043

A Concrete Naming Certificate for $\mathsf{LocallyUniformConvergence}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, Kelley on compactly generated uniform control, and Bourbaki on local uniform convergence over compacta, the $\mathsf{LocallyUniformConvergence}^{\uparrow}$ packet records the standard compact-window reading of locally uniform convergence as finite NameCert data. It sits after the uniform-tail certificate of chapter 3417, the continuity rows of chapter 3988, and the compact metric completion boundary of chapter 1687. The packet names compact-window rows, uniform-modulus rows, continuous-function rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming rows; it does not assert open-cover compactness, selected limiting functions, quotient function-space equality, countable choice, or ambient completeness.

Definition 3043.1 (LocallyUniformConvergence carrier)**.** *A* $\mathsf{LocallyUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, U, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-window row, listing each retained compact metric window and its finite inspection addresses.*
2. *U is the uniform-modulus row, assigning to every displayed compact window a shared tail modulus for the family on that window.*
3. *F is the continuous-function row, naming the approximating function data and the endpoint continuity data read on the same windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for all rational tail and distance comparisons used by the window.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after the compact-window, uniform-modulus, continuous-function, and regular-rational rows have been read.*
6. *H records componentwise* hsame *transport for compact windows, moduli, continuous-function rows, rational readbacks, real seals, provenance, and local naming.*
7. *C records displayed* Cont *replay from a local compact-window request through K, U, F, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{LocallyUniformConvergence}^{\uparrow}$*,* $\mathsf{UniformConvergence}^{\uparrow}$*,* $\mathsf{CFun}^{\uparrow}$*,* $\mathsf{CompactMetricComplet}$ $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$ *row.*
The carrier exports only this compact-window route. It has no coordinate for open-cover compactness, a selected global limit function, quotient equality of functions, or a host supremum norm outside the displayed windows.

Lean4 ✓ `BEDC.Derived.LocallyUniformConvergenceUp.TasteGate.LocallyUniformConvergenceTasteGate single carrier alignment`

Theorem 3043.2 (LocallyUniformConvergence NameCert obligations)**.** *For every accepted* LocallyUniformConverg *carrier $L = (K, U, F, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$ *surface consists exactly of compact-window admission through K, shared uniform-modulus admission through U, continuous-function row compatibility through F,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *handoff through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 4022.1. The displayed rows are exactly $K, U, F, Q, E, H, C, P, N$, and each consumer-facing request begins with one of the compact windows named by K.

The convergence side is exhausted by U, Q, E. A uniform tail claim on a retained compact window must first read the shared modulus row U, then the regular-rational comparison row Q, and only then the terminal real handoff E.

The continuity side is exhausted by F together with the same compact-window route. The continuous-function rows are consumed on the windows already displayed by K, so endpoint continuity is replayed locally rather than by selecting a host function-space limit.

Transport, continuation, provenance, and local naming preserve this finite route. Since no carrier coordinate stores open-cover compactness, quotient equality of functions, countable choice, an ambient supremum norm, or an external limiting function selector, no local NameCert obligation may use such data. □

Theorem 3043.3 (LocallyUniformConvergence compact route)**.** *Every accepted local convergence read from a* $\mathsf{LocallyUniformConvergence}^{\uparrow}$ *carrier factors through the compact-window route*

$$K \longmapsto U, F \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the route consumes the existing $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, *and* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *packets only through displayed finite rows, not through open-cover compactness, quotient function-space equality, selected limits, countable choice, or ambient completion.*

Proof. Begin with a local convergence request. The request must name a compact metric window in K, so the carrier first fixes the finite portion of the domain on which convergence is being inspected.

Read the uniform and continuity rows on that same window. The uniform-modulus row U supplies the shared tail data, while the continuous-function row F records the approximating and endpoint continuity data that can be replayed on the retained window.

The comparison then passes through Q and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row carries the rational tolerance comparisons, and the $\mathsf{Real}^{\uparrow}$ row is reached only after those comparisons and the compact-window data have been displayed.

Finally use H, C, P, N to transport, replay, source, and name the same route. The cited sibling packets provide the uniform-convergence, continuous-function, and compact-metric-completion rows consumed by the carrier, but none of those reads adds open-cover compactness, a selected global limit, quotient function equality, countable choice, or ambient completion. □

Falsifiable prediction: Deleting the compact-window row K prevents any accepted $\mathsf{LocallyUniformConvergence}^{\uparrow}$ local convergence read, even when uniform-modulus, continuous-function, regular-rational, real-handoff, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: uniform-convergence, continuous-function, compact-metric-completion: none is bijective to the locally uniform convergence packet because only $\mathsf{LocallyUniformConvergence}^{\uparrow}$ binds compact-window rows, uniform-modulus rows, continuous-function rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ handoff, transport, replay, provenance, and local naming into one accepted local convergence route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyUniformConvergence}^{\uparrow}$ packet is generated from compact-window rows, uniform-modulus rows, continuous-function rows, RegSeqRat readback, Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4022.1, Definition 4022.2, and Definition 4022.3 seed the finite compact-window route for locally uniform convergence over displayed uniform, continuity, completion, regular-rational, and real-handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient function-space equality, selected global limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window admission, uniform-modulus exactness, continuous-function compatibility, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocallyUniformConvergenceUp.

Chapter 3044

A Concrete Naming Certificate for $\mathsf{BishopMetricCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopMetricCompletion}^{\uparrow}$ packet records constructive metric completion by finite regular Cauchy data rather than by quotient completion objects. It is a sibling of chapter 1506, chapter 2114, and chapter 1612, and it routes its numerical names through chapter 126, chapter 301, and chapter 38. The packet names only the displayed metric source, dense embedding, regular Cauchy windows, separated comparison, transport, replay, provenance, and local NameCert rows.

Definition 3044.1 (Bishop metric completion carrier)**.** *A* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, E, Q, W, S, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the constructive metric source row, naming the source points, distance-read windows, and metric comparison ledger used by the completion request.*
2. *E is the dense embedding row from source names into the completion-facing name surface.*
3. *Q is the regular Cauchy family row, displayed as finitely inspectable* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks rather than as a quotient class.*
4. *W is the finite-window schedule, reading the required prefixes through* $\mathsf{StreamName}^{\uparrow}$*.*
5. *S is the separated comparison row, available only after two regular Cauchy windows and their tolerance rows have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for distances and terminal comparison readbacks produced by the regular Cauchy family.*
7. *T is the transport row from source-window reads to completion-window reads.*
8. *H records componentwise* hsame *transport for the metric source, dense embedding, regular family, windows, separated comparison, Real seal, transport, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for source inspection, dense embedding, regular-window readback, separated comparison, Real sealing, and transport.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopMetricCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{LocatedMetricCompletion}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, Q, W, S, R, T, H, C, P, N$. It has no coordinate for quotient completions, selected representatives, countable-choice schedules, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopMetricCompletionUp.BishopMetricCompletionTasteGate single carrier alignment`

Theorem 3044.2 (Bishop metric completion NameCert obligations)**.** *For an accepted* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *packet $B = (M, E, Q, W, S, R, T, H, C, P, N)$, the local NameCert surface consists of carrier admission, dense-embedding exposure, regular Cauchy family admission, finite-window inspection, separated comparison, Real-seal handoff, source-to-completion transport, componentwise* hsame *stability, displayed* Cont *replay, provenance, and ledger non-escape. Every obligation is discharged through the displayed route*

$$M \longmapsto E \longmapsto Q \longmapsto W \longmapsto S \longmapsto R \longmapsto T$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.BishopMetricCompletionUp.BishopMetricCompletionCarrier namecert obligations`

Proof. Project the carrier of Definition 4023.1. The source metric row M and dense embedding row E must be visible before any regular Cauchy family Q is inspected.

The finite-window row W reads the regular family through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. The separated comparison row S can then compare only displayed regular windows at the requested tolerance, and the $\mathsf{Real}^{\uparrow}$ seal row R records the resulting distance or comparison readback.

The transport row T keeps the source and completion windows on the same route. Componentwise transport through H and replay through C preserve those rows, while P and N source and name the packet.

Since quotient completions, selected representatives, countable-choice schedules, host equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3044.3 (Bishop metric completion finite-window handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *packet factors through the source metric row, dense embedding row, regular Cauchy family row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, separated comparison row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local NameCert rows displayed by B. The handoff exports a finite constructive completion surface, not a quotient completion, selected representative, countable-choice schedule, host equality principle, or ambient Real-completeness theorem.*

Proof. Begin with Definition 4023.2. It places every accepted read on the route $M, E, Q, W, S, R, T, H, C, P, N$.

Use the sibling completion chapters only through their visible rows. The metric-completion, located-metric-completion, and separated-completion surfaces supply completion vocabulary, located source comparison, and separated zero-distance comparison, but the Bishop packet receives them through M, E, Q, W, S, R, T and the structural rows.

The finite windows W and readbacks Q are the only access points to the regular Cauchy family. Therefore a downstream completion consumer can inspect a finite prefix and the corresponding separated comparison, but it cannot select a representative or collapse names by host equality.

The $\mathsf{Real}^{\uparrow}$ row R, transport row T, replay row C, provenance row P, and local name N close the handoff. No excluded object is a carrier coordinate, so the finite-window route is the whole exported completion surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMetricCompletion}^{\uparrow}$ packet is generated from a constructive metric source row, dense embedding row, regular Cauchy family row, finite StreamName windows, RegSeqRat readbacks, separated comparison row, RealUp seal, source-to-completion transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4023.1, Definition 4023.2, and Definition 4023.3 seed the constructive metric-completion carrier, NameCert obligation surface, and finite-window handoff over MetricCompletion, LocatedMetricCompletion, SeparatedCompletion, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, dense-embedding exposure, regular Cauchy window admission, separated comparison, Real sealing, transport, replay, provenance, and local NameCert non-escape for $\mathsf{BishopMetricCompletion}^{\uparrow}$.

Chapter 3045

A Concrete Naming Certificate for $\mathsf{LocatedCompactness}^{\uparrow}$

Following Bishop–Bridges/Kelley-style located compactness, the $\mathsf{LocatedCompactness}^{\uparrow}$ packet records compact neighbourhood control by finite radius witnesses rather than by open-cover selection. It sits between the metric carrier of chapter 57, the compact-metric witness of chapter 129, the local support window of chapter 1726, the center-radius boundary of chapter 1451, and the finite observation schedule of chapter 126. Completion-facing consumers receive a finite located-neighbourhood interface, not a host compact subset, selected subcover, quotient completion, countable-choice schedule, or ambient real-completeness principle.

Definition 3045.1 (Located compactness carrier)**.** *A* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, K, B, U, W, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *source row, naming the carrier, distance witness, and local classifier used by every radius request.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness row for the displayed neighbourhood or closed-ball window.*
3. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row naming the center, radius, non-strict membership comparison, and radius ledger of the inspected window.*
4. *U is the* $\mathsf{LocallyCompact}^{\uparrow}$ *support row, recording that the compact witness is attached to a displayed local neighbourhood rather than to a global host compactness claim.*
5. *W is the located-neighbourhood radius row, recording the finite positive-radius requests, comparison outcomes, and compact-witness handoffs available to completion consumers.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row through which finite radius requests, compact-witness reads, and closed-ball inspections are scheduled.*
7. *H records componentwise* hsame *transport for metric source, compact witness, closed-ball boundary, local support, located-neighbourhood radius, stream schedule, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite stream windows through the closed-ball boundary, local compact support, compact metric witness, and located-radius handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for open-cover compactness, selected finite subcovers, quotient completion equality, choice-selected compact witnesses, host powersets, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3045.2 (Located compactness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier $L = (M, K, B, U, W, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$ *surface consists exactly of metric-source admission, compact-metric witness admission, closed-ball boundary exposure, local compact support, finite located-neighbourhood radius requests, finite* $\mathsf{StreamName}^{\uparrow}$ *scheduling, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must pass through $M, K, B, U, W, S, H, C, P, N$ before using located compactness.*

Lean4 ✓ `BEDC.Derived.LocatedCompactnessUp.LocatedCompactnessCarrier namecert obligations`

Proof. Project the carrier of Definition 4024.1. The ten rows displayed there are the only rows visible to the local classifier, so carrier admission is already confined to $M, K, B, U, W, S, H, C, P, N$.

Read the analytic rows componentwise. The metric source M supplies the carrier and distance witness, B supplies the center-radius boundary, U supplies the local compact support, K supplies the compact-metric witness, and W supplies the located-neighbourhood radius requests.

The scheduling and structural rows preserve the same finite route. Stream windows in S expose only finite requests, H transports those rows componentwise, C replays only the displayed handoff, and P, N source and name the packet. Since no additional coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 3045.3 (Located compactness finite-net handoff)**.** *Every L10 completion or compact-metric consumer of an accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier receives only the finite handoff*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{LocallyCompact}^{\uparrow},\quad \mathsf{MetricClosedBall}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$$

through the displayed rows $M, K, B, U, W, S, H, C, P, N$. The handoff provides finite neighbourhood-radius requests, compact-witness rows, closed-ball comparisons, local-support rows, and NameCert provenance; it exports no open-cover compactness, selected dense points, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the obligation surface of Definition 4024.2. A completion consumer first reads the metric row, closed-ball row, local compact support row, compact witness row, and located-radius row as one finite packet.

The finite request order is supplied by S. A radius or compact-witness inspection can occur only after a displayed $\mathsf{StreamName}^{\uparrow}$ window has exposed the request, and the replay row C sends that request through B, U, K, and W without adding a host compactness object.

Finally use the structural rows H, P, N. They transport, source, and name the same finite handoff for compact-metric and completion consumers. Because the packet has no row for open covers, selected dense points, quotient completions, countable choice, or ambient real completeness, the exported surface is exactly the finite located-neighbourhood handoff above. □

Theorem 3045.4 (Located compactness radius stability)**.** *For every accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier* $L = (M, K, B, U, W, S, H, C, P, N)$*,* hsame *transport of the metric source, compact witness, closed-ball boundary, local support, located-radius request,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* Cont *replay,* Pkg *provenance, and local* NameCert *row preserves the finite located-neighbourhood handoff for every completion-facing consumer. The transported read remains the same finite route through*

$$M, K, B, U, W, S, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.LocatedCompactnessUp.LocatedCompactnessCarrier radius transport`

Proof. Project the transport row H from Definition 4024.1. It records componentwise transport for the metric source, compact witness, closed-ball boundary, local support, located-radius request, stream schedule, replay, provenance, and local naming rows.

Use Definition 4024.2 to identify those transported rows as the whole local NameCert surface. No transported completion-facing read can detach the radius request from $M, K, B, U, W, S, C, P, N$.

Finally apply the finite-net handoff of Definition 4024.3. The consumer route remains finite and choice-free, so transport preserves the located-neighbourhood handoff without introducing open-cover compactness, selected dense points, quotient completion equality, countable choice, or ambient real completeness. □

Falsifiable prediction: A completion-facing consumer that reads $\mathsf{LocatedCompactness}^{\uparrow}$ without displaying the metric source, compact witness, closed-ball boundary, local compact support, finite located-radius request, StreamName schedule, componentwise transport, continuation replay, package provenance, and local NameCert rows refutes the carrier surface.

Independence witness: metric, compactmetric, locallycompact, metricclosedball, streamname: no listed sibling is bijective to $\mathsf{LocatedCompactness}^{\uparrow}$ because this packet binds source metric data, compact witnesses, closed-ball boundaries, local support, located-radius requests, finite scheduling, transport, replay, provenance, and local naming in one completion-facing interface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCompactness}^{\uparrow}$ packet is generated from metric source rows, compact-metric witnesses, closed-ball boundaries, local compact support rows, finite located-neighbourhood radius requests, StreamName schedules, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4024.1, Definition 4024.2, and Definition 4024.3 seed the finite located-neighbourhood compactness interface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, quotient completion equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface for BEDC.Derived.LocatedCompactnessUp.

Chapter 3046

A Concrete Naming Certificate for $\mathsf{FourierTransform}^{\uparrow}$

Following Fourier and Stein–Shakarchi on the transform side of harmonic analysis, the $\mathsf{FourierTransform}^{\uparrow}$ packet records the finite transform-specific read that consumes the finite-observation $\mathsf{Fourier}^{\uparrow}$ surface of chapter 124, the integral witness rows of chapter 123, the inner-product probe rows of chapter 86, the real seal rows of chapter 38, and the product-tail handoff of chapter 409. It names transform-sample windows, kernel probes, integral witnesses, real seals, convolution handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming as finite BEDC rows. The chapter does not assert Plancherel, inversion, convergence closure, quotient equality of functions, or a selected ambient transform.

Definition 3046.1 (FourierTransform carrier)**.** *A* $\mathsf{FourierTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, K, I, R, V, H, C, P, N)$$

with the following displayed rows:

1. *W is the transform-sample window row, listing the finite source windows, frequency addresses, and retained sample requests read from the* $\mathsf{Fourier}^{\uparrow}$ *finite-observation surface.*
2. *K is the kernel/probe row, recording the displayed oscillatory kernel tags, inner-product probes, and source-domain probes used by the retained transform samples.*
3. *I is the integral-witness row, requiring each retained sample to name its* $\mathsf{Integral}^{\uparrow}$ *witness and the finite approximation ledger used by that witness.*
4. *R is the real-seal row, reached only after the transform-sample window, kernel/probe row, and integral-witness row have been replayed.*
5. *V is the convolution handoff row, reading* $\mathsf{CauchyConvolution}^{\uparrow}$ *only through the finite product-tail packet needed by convolution-facing transform consumers.*
6. *H records componentwise* hsame *transport for transform windows, kernel/probe rows, integral witnesses, real seals, convolution handoff rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite transform request through W, K, I, R, V.*
8. *P is the* Pkg *provenance over* $\mathsf{FourierTransform}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvolution}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FourierTransform}^\uparrow}$ *row.*
The classifier compares accepted packets only through the component rows $W, K, I, R, V, H, C, P, N$. It has no coordinate for equality of host transforms, inversion, Plancherel, convergence closure, selected limiting integrals, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FourierTransformUp.FourierTransformUp`

Theorem 3046.2 (FourierTransform NameCert obligations)**.** *For every accepted* $\mathsf{FourierTransform}^\uparrow$ *carrier $T = (W, K, I, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FourierTransform}^\uparrow}$ *surface consists exactly of transform-sample window admission through W, kernel and probe admission through K, integral-witness readback through I, terminal* $\mathsf{Real}^\uparrow$ *seal admission through R, convolution handoff through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4025.1. Its public rows are exactly $W, K, I, R, V, H, C, P, N$, so every local naming duty begins with a finite transform-sample window rather than with a host transform.

The analytic source rows are then read in order. The row K exposes the kernel/probe data and inner-product probes, while I attaches each retained sample to an integral witness. The real seal row R is reached only after those rows have been displayed.

The convolution-facing part is confined to V. It imports only the finite product-tail handoff already named by $\mathsf{CauchyConvolution}^\uparrow$, and it does not add an inversion theorem, Plancherel identity, convergence closure, or quotient equality of functions.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a selected ambient transform or a stronger analytic theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 3046.3 (FourierTransform convolution handoff)**.** *Every convolution-facing read from an accepted* $\mathsf{FourierTransform}^\uparrow$ *carrier factors through*

$$W, K, I \longmapsto R \longmapsto V \longmapsto H, C, P, N.$$

The read may consume $\mathsf{Fourier}^\uparrow$ *finite observations,* $\mathsf{Integral}^\uparrow$ *witnesses,* $\mathsf{InnerProduct}^\uparrow$ *probes,* $\mathsf{Real}^\uparrow$ *seals, and the* $\mathsf{CauchyConvolution}^\uparrow$ *product-tail handoff, but it exports only the finite transform-sample route displayed by the carrier.*

Proof. Start with the transform sample. The window row W fixes the retained source and frequency addresses, and the kernel/probe row K records the oscillatory kernel tags and inner-product probes used at those addresses.

Attach the integral and real rows next. The integral-witness row I supplies the finite approximation ledger for the retained sample, and the real-seal row R records the terminal real read obtained from that ledger.

Only after this route is displayed can the convolution row V be consumed. Its source is the finite product-tail handoff of $\mathsf{CauchyConvolution}^\uparrow$, so convolution-facing consumers receive a named handoff row rather than a host multiplication theorem or an equality of transforms.

The structural rows H, C, P, N preserve the same factorization under componentwise transport, replay, provenance, and local naming. Therefore the handoff exports the displayed transform-sample route only, with Plancherel, inversion, convergence closure, quotient function equality, and selected ambient transforms outside the packet. □

Falsifiable prediction: Deleting the transform-sample window row W prevents any accepted $\mathsf{FourierTransform}^{\uparrow}$ sample read, even when kernel/probe, integral-witness, real-seal, convolution handoff, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: Fourier finite observations, integral witnesses, inner-product probes, real seals, and Cauchy convolution handoffs are not interchangeable with the $\mathsf{FourierTransform}^{\uparrow}$ packet because only $\mathsf{FourierTransform}^{\uparrow}$ binds transform-sample windows, kernel/probe rows, integral witnesses, real seals, convolution handoff, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted transform route.

Closure status for $\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{FourierTransform}^{\uparrow}$ packet is generated from transform-sample windows, kernel/probe rows, integral witnesses, real seals, a convolution handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4025.1, Definition 4025.2, and Definition 4025.3 seed the finite transform-sample route over Fourier finite observations, integral witnesses, inner-product probes, real seals, and convolution handoff data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Plancherel, inversion, convergence closure, quotient function equality, selected ambient transforms, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transform-window admission, kernel/probe exactness, integral-witness readback, real-seal handoff, convolution handoff, transport, replay, provenance, and local naming for BEDC.Derived.FourierTransformUp.

Chapter 3047

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$

Following Bishop and Bridges on completion and Mac Lane on adjunction triangular identities, the $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$ packet records the L10 completion handoff that keeps the completion functor, the regular Cauchy adjunction, the universal property, and the idempotence boundary visible in one finite certificate. It sits beside chapter 423, chapter 3882, chapter 2381, chapter 3376, and chapter 3888. The packet names a triangular completion route; it does not assert quotient completion equality, selected limits, host functor extensionality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3047.1 (CauchyCompletionAdjunctionTriangle carrier)**.** *A* $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, A, U, I, R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row naming the source-to-completion functorial surface.*
2. *A is the* $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ *row naming the regular Cauchy adjunction boundary.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row that exposes the extension and uniqueness handoff.*
4. *I is the completion idempotence row, read through the completion idempotence and metric completion idempotence chapters.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying regular rational completion readback.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window for the regular Cauchy route.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the displayed completion windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, A, U, I, R, W, D have been displayed.*
9. *H stores componentwise* hsame *transport for the functor, adjunction, universal-property, idempotence, regular readback, window, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay for the triangular route from functor and adjunction rows to universal handoff, idempotence boundary, and Real-facing seal.*

11. *P is the* Pkg *provenance over* CauchyCompletionAdjunctionTriangle$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CauchyRegularCompletionAdjunction$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$ *row.*

The classifier accepts only packets whose completion-facing read traverses the displayed rows $F, A, U, I, R, W, D, E, H$ It has no coordinate for quotient completions, selected limits, host equality of completed objects, unrestricted functor extensionality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.TasteGate.CauchyCompletionAdjunctionTriangleU`
`single carrier alignment`

Theorem 3047.2 (CauchyCompletionAdjunctionTriangle NameCert obligations)**.** *For every accepted* CauchyCompletionAdjunctionTriangle$^{\uparrow}$ *carrier $T = (F, A, U, I, R, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$ *obligations are exactly carrier admission, functor-row visibility, regular adjunction visibility, universal-property handoff, idempotence-boundary retention, regular-rational and stream-window readback, dyadic tolerance compatibility, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleNamecertObl`

Proof. Project the carrier row by row from Definition 4026.1. Carrier admission exposes precisely the twelve displayed rows, so every local obligation has one visible coordinate.

Read the completion triangle in order. The functor row F and regular adjunction row A are available before the universal-property row U, and the idempotence row I is read as a boundary on reusing the same completion surface.

The Real-facing part is downstream. Rows R, W, D provide regular rational readback, finite windowing, and dyadic tolerance data before the terminal seal E is visible.

Finally, H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate supplies quotient completion equality, selected limits, host functor extensionality, countable choice, or ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 3047.3 (CauchyCompletionAdjunctionTriangle universal handoff)**.** *Every accepted* CauchyCompletionAd *carrier hands a completion-facing consumer from the* CauchyCompletionFunctor$^{\uparrow}$ *row and the* CauchyRegularComplet *row to the* CompletionUniversalProperty$^{\uparrow}$ *row before any idempotence or Real-facing read is exported. The handoff is the finite route*

$$F, A \longmapsto U \longmapsto I, R, W, D, E, H, C, P, N$$

and not a quotient completion, selected limit, host equality of extensions, or ambient completeness assertion.

Proof. Begin with the functor and adjunction rows in the carrier. They are separate coordinates, so a consumer cannot read the universal-property row as a detached host theorem.

Project U. It is the only row that records extension and uniqueness handoff for the displayed completion route, and it is read after F and A have fixed the source completion surface.

The idempotence and Real-facing rows are downstream of that handoff. Row I records reuse of the same completion boundary, while R, W, D, E provide the finite regular-rational, window, dyadic, and Real seal route.

Rows H, C, P, N preserve the same finite path. Thus the universal handoff is exactly the displayed route and cannot become quotient completion equality, selected-limit data, host extension equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleUniversalHa`

Theorem 3047.4 (CauchyCompletionAdjunctionTriangle idempotence boundary)**.** *For every accepted* CauchyCompletionAdjunctionTriangle$^{\uparrow}$ *carrier, the idempotence read I is a boundary on reusing the already displayed completion route $F, A, U, R, W, D, E, H, C, P, N$. It cannot identify completed objects by host equality, erase the universal-property row, select a limit representative, or assert ambient completeness of* Real$^{\uparrow}$.

Proof. Open the accepted packet and isolate I. The carrier definition places I after the completion functor, regular adjunction, and universal-property rows, not as an independent completed-object equality.

Use the universal handoff theorem. It forces the route through F, A, U before I can be consumed by a completion-facing reader.

The remaining rows R, W, D, E, H, C, P, N show what the idempotence boundary may preserve: regular rational readback, finite observation windows, dyadic tolerances, the Real seal, transport, replay, provenance, and local naming. No row contains host equality of completions, selected representatives, erased universal handoff data, or ambient completeness. □

Falsifiable prediction: A completion consumer that exports a Cauchy-completion triangle without displaying the functor, regular adjunction, universal-property, idempotence, RegSeqRat, StreamName, DyadicRatCore, Real seal, transport, replay, provenance, and local naming rows is not reading a CauchyCompletionAdjunctionTriangle$^{\uparrow}$ packet.

Independence witness: cauchycompletionfunctor, cauchyregularcompletionadjunction, completionidempotence, cauchycompletionreflection, metriccompletionidempotence: CauchyCompletionAdjunctionTriangle is not bijective to any listed sibling because it jointly carries functor, adjunction, universal-property, idempotence, regular-rational, window, dyadic, Real seal, transport, replay, provenance, and local naming rows.

Closure status for NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$.

Constructive story: A CauchyCompletionAdjunctionTriangle$^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a CauchyRegularCompletionAdjunction row, a CompletionUniversalProperty row, a completion idempotence row, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$, seed)

Scope closed: Definition 4026.1, Definition 4026.2, Definition 4026.3, and Definition 4026.4 seed the finite triangular handoff for the L10 Cauchy completion route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleNamecertObligati`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host functor extensionality, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, universal-property handoff, idempotence boundary, RegSeqRat and StreamName readback, DyadicRatCore compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.

Chapter 3048

A Concrete Naming Certificate for $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ packet records the bounded monotone Cauchy criterion as finite BEDC data rather than as an ambient completeness principle. It sits after the monotone source surface of chapter 438, the bounded witness route of chapter 535, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes a displayed regular Cauchy route from bounded monotone data; it does not select a limit, quotient real names, import trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3048.1 (MonotoneCauchyCriterion carrier)**.** *A* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, S, T, M, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded monotone source row, including the inspected monotone observations and their located upper-bound guard.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite windows read from the source.*
3. *T is the tail-window schedule row that turns a dyadic request into a displayed tail window.*
4. *M is the monotone-tail witness row, recording that later observations stay inside the same bounded monotone route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for radii, slack, monotone comparisons, and upper-bound traps.*
6. *L is the located upper-bound trap row tying each selected tail window to a finite lower observation and a located upper guard.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window, tail, witness, ledger, and trap data.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational readback has been displayed.*
9. *H records componentwise* hsame *transport for the bounded source, schedule, tail window, monotone witness, dyadic ledger, located trap, readback, seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a criterion request through* $B, S, T, M, D, L, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, T, M, D, L, R, E, H, C, P, N$. *It has no coordinate for a selected limiting representative, quotient real equality, host real equality, trichotomy, open-cover compactness, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyCriterionUp.MonotoneCauchyCriterionTasteGate single carrier alignment`

Theorem 3048.2 (MonotoneCauchyCriterion NameCert obligations)**.** *For every accepted* MonotoneCauchyCriterio *carrier* $K = (B, S, T, M, D, L, R, E, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, bounded-window exposure through* B, S, T, *monotone-tail witness exposure through* M, *dyadic ledger exactness through* D, *located upper-bound trap exposure through* L, *regular-rational readback through* R, *Real seal nonescape through* E, *componentwise* hsame *transport through* H, *displayed* Cont *replay through* C, Pkg *provenance through* P, *and local naming through* N.

No obligation may be discharged by selected limits, quotient real equality, trichotomy, open-cover compactness, choice, host real equality, or ambient completeness.

Proof. Project the carrier of Definition 4027.1. The displayed rows are exactly $B, S, T, M, D, L, R, E, H, C, P, N$, so admission exposes only the bounded monotone source, finite schedules, tail witness, dyadic ledger, located trap, readback, seal, transport, replay, provenance, and naming data.

The bounded-window rows are B, S, T. They force every request to pass first through the inspected source and a finite scheduled tail window before any Cauchy-facing witness can be read.

The witness and numerical rows are M, D, L, R. The monotone-tail witness stays attached to the dyadic tolerance ledger and located trap, and the regular-rational readback is produced from that same finite route.

Finally the seal and structural rows E, H, C, P, N preserve and name the route. Since the excluded analytic objects are absent from the carrier, they cannot become local NameCert obligations. □

Theorem 3048.3 (MonotoneCauchyCriterion bounded-window route)**.** *Every criterion-facing read exported by an accepted* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MonotoneCauchy}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \mapsto$$

The route first exposes bounded monotone source rows and finite tail windows, then reads the monotone-tail witness, dyadic tolerance ledger, located upper-bound trap, regular-rational readback, and terminal real seal. It exports no selected limit, quotient real equality, host real equality, trichotomy, open-cover compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a criterion-facing read of $K = (B, S, T, M, D, L, R, E, H, C, P, N)$. Carrier projection exposes the bounded monotone source row B, the finite $\mathsf{StreamName}^{\uparrow}$ schedule S, and the tail-window row T before the witness row can be consumed.

Compose the bounded monotone witness route with the monotone Cauchy surface. The existing Lean touchpoint `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier_`

records the same source-to-tail-to-witness-to-ledger-to-seal chain at the finite BHist, Cont, and Pkg level.

The dyadic ledger D and located trap L then feed the regular-rational readback R. Only after R is displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal E be read, and H, C, P, N transport, replay, source, and name the same finite packet.

Because no excluded completion, quotient, selection, trichotomy, open-cover compactness, choice, or host-real coordinate occurs in the carrier, every exported criterion read is exhausted by the displayed bounded-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 4027.1, Definition 4027.2, and Definition 4027.3 seed the bounded monotone Cauchy criterion route from MonotoneCauchyUp and BoundedMonotoneCauchyWitnessUp rows through StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient real equality, trichotomy, open-cover compactness, choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window exposure, monotone-tail witness exposure, dyadic ledger exactness, located trap exposure, readback exactness, Real seal nonescape, transport, and replay proofs for BEDC.Derived.MonotoneCauchyCriterionUp.

Constructive story: A $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ packet is generated from bounded monotone source rows, finite StreamName windows, a tail-window schedule, a monotone-tail witness, a DyadicRatCore tolerance ledger, a located upper-bound trap, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3049

A Concrete Naming Certificate for $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ packet records paired regular-Cauchy difference estimates as finite rational-window data rather than as equality in a completed real quotient. It sits over the regular sequence route of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the tail envelope comparison route of chapter 3992. The packet supplies completion and Real algebra consumers with one finite difference-bound surface; it does not assert quotient equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3049.1 (Regular Cauchy difference-bound carrier)**.** *A* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, W, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, read only through finite rational windows.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displayed on the same request schedule as X.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family exposing the compared prefixes.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and subtraction ledger for the displayed window.*
5. *E is the* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *envelope row bounding the paired tails after W and D have been read.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed difference bound.*
7. *H records componentwise* hsame *transport for the two regular sources, the shared windows, the dyadic ledger, the envelope comparison, the Real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for window inspection, tolerance subtraction, tail-envelope comparison, Real sealing, and structural transport.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, E, R, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable-choice moduli, trichotomy, host subtraction, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundTasteGate single carrier alignment`

Theorem 3049.2 (Regular Cauchy difference-bound NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *packet $B = (X, Y, W, D, E, R, H, C, P, N)$, the local NameCert surface consists exactly of carrier admission for the two regular rational sources, shared finite-window inspection, dyadic difference budgeting, tail-envelope comparison, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming. Every accepted bound read follows*

$$X, Y \longmapsto W \longmapsto D \longmapsto E \longmapsto R$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundCarrier namecert obligations`

Proof. Project the carrier of Definition 4028.1. The two source rows X, Y are the only regular-rational inputs, and they are read on the shared window row W.

The dyadic ledger D supplies the displayed tolerance and subtraction budget for that same window. The tail-envelope row E can therefore compare only the paired tails already exposed by X, Y, W, D.

The terminal Real row R is downstream of the envelope comparison, while H, C, P, N only transport, replay, source, and name the same finite route. Since the carrier has no quotient equality, selected representative, countable-choice, host-subtraction, or ambient-completeness coordinate, the listed rows exhaust the local certificate. □

Theorem 3049.3 (Regular Cauchy difference-bound RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *packet receives the paired difference estimate only through the displayed route from X, Y to W, D, E, R. The handoff preserves the finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *envelope row, and it exports no quotient real equality, selected limit, countable-choice schedule, trichotomy, host real subtraction, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundRegSeqRatHandoff`

Proof. Begin with the NameCert obligations of Definition 4028.2. They place every accepted read on the route $X, Y, W, D, E, R, H, C, P, N$.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer can inspect only the two displayed regular sources and the finite windows in W. It must therefore read the same dyadic ledger D before any bound is available.

A $\mathsf{Real}^{\uparrow}$ consumer sees the terminal row R only after the tail-envelope comparison E has been consumed. The structural rows preserve this route, so no excluded equality, selector, choice, host-subtraction, or completeness object can be exported. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ read factors through two RegSeqRat source rows, a shared StreamName window row, a DyadicRatCore tolerance ledger, a CauchyTailEnvelopeComparison envelope row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchy-tail-envelope-comparison: $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired regular sources, shared finite windows, dyadic difference budgeting, tail-envelope comparison, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName window row, a DyadicRatCore tolerance and subtraction ledger, a CauchyTailEnvelopeComparison row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4028.1, Definition 4028.2, and Definition 4028.3 seed the finite paired regular-Cauchy difference-bound carrier, certificate surface, and RegSeqRat-to-Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, host real subtraction, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, paired regular-source exposure, shared-window exactness, dyadic difference budgeting, tail-envelope comparison, Real sealing, transport, replay, provenance, and local NameCert non-escape for $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$.

Chapter 3050

A Concrete Naming Certificate for $\mathsf{LocallyCompactHausdorff}^{\uparrow}$

Following Kelley and Bourbaki on locally compact Hausdorff spaces, the $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ packet records the point-local compact-neighbourhood and point-separation rows that a constructive topology consumer may inspect without invoking open-cover compactness or choice of neighbourhood bases. It sits over the indexed-open surface of $\mathsf{Topology}^{\uparrow}$, the separated uniform route of chapter 3775, the compact-metric window of chapter 129, the formal-ball observation interface of chapter 1510, and the point-local metric compactness packet of chapter 3787.

Definition 3050.1 (Locally compact Hausdorff carrier)**.** *A* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, U, x, V, K, F, S, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *source row, including indexed open reads, separated neighbourhood reads, and the point classifier used by topology consumers.*
2. *U is a neighbourhood row around the inspected point x, displayed through the same topology or separated uniform source.*
3. *V is the compact neighbourhood inclusion row, witnessing that the inspected neighbourhood is carried by the compact window K.*
4. *K is a finite compactness window read through* $\mathsf{CompactMetric}^{\uparrow}$ *or a compact formal-ball window from* $\mathsf{FormalBall}^{\uparrow}$*.*
5. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *or finite-window replay row used when a consumer requests radius-indexed local observations inside K.*
6. *S is the Hausdorff separation row: distinct point reads are separated only by displayed neighbourhood or entourage witnesses over T.*
7. *H records componentwise* hsame *transport for the topology or uniform source, inspected point, local neighbourhood, compact window, formal-ball window, separation row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from local topology reads to compact-window, formal-ball, and Hausdorff-separation consumers.*

9. *P is the* Pkg *provenance over* LocallyCompactHausdorff$^{\uparrow}$, Topology$^{\uparrow}$, HausdorffUniformSpace$^{\uparrow}$, CompactMetric$^{\uparrow}$, FormalBall$^{\uparrow}$, LocallyCompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactHausdorff}^{\uparrow}}$ *row.*

The packet has no coordinate for arbitrary open-cover compactness, selected neighbourhood bases, quotient point equality, excluded middle, countable choice, or ambient local compactness outside the displayed carrier.

Lean4 ✓ `BEDC.Derived.LocallyCompactHausdorffUp.LocallyCompactHausdorffTasteGate single carrier alignment`

Theorem 3050.2 (LocallyCompactHausdorff carrier admission)**.** *For every accepted* LocallyCompactHausdorff$^{\uparrow}$ *packet $L = (T, U, x, V, K, F, S, H, C, P, N)$, carrier admission is rowwise over the displayed topology or separated-uniform source, inspected point, local neighbourhood row, compact-window inclusion, formal-ball or compact-metric finite window, Hausdorff separation row, componentwise transport, replay, provenance, and local name. Admission cannot be obtained from an unrecorded host topology, an open-cover compactness theorem, a selected basis, quotient point equality, or ambient local compactness.*

Proof. Project the carrier of Definition 4029.1. The packet exposes exactly $T, U, x, V, K, F, S, H, C, P, N$, so every accepted read begins with the topology or separated-uniform source and the inspected point.

Next read the compact side. The row V ties the neighbourhood U to the compact window K, while F supplies only the finite formal-ball or compact-metric observations cited by chapter 129 and chapter 1510.

Finally read the structural rows H, C, P, N. They transport, replay, record provenance, and bind the local name for the same finite packet. Since the carrier contains no host topology, open-cover compactness, selected basis, quotient equality, or ambient local-compactness coordinate, none can enter through carrier admission. □

Theorem 3050.3 (LocallyCompactHausdorff compact-neighbourhood nonescape)**.** *Every local compact-neighbourhood read of an accepted* LocallyCompactHausdorff$^{\uparrow}$ *packet factors through*

$$T, U, x, V, K, F \longmapsto H, C, P, N.$$

The read may use compact-metric and formal-ball windows, and may compare with the LocallyCompactMetric$^{\uparrow}$ *point-local route when a metric presentation is displayed. When completion-facing readers consume the local compact neighbourhood, the compact-window row may be refined through the finite located-neighbourhood radius handoff of chapter 4024, not through open-cover compactness. The read cannot promote the window to arbitrary open-cover compactness, a global compactness theorem, a selected local base, quotient completion equality, or ambient completeness.*

Proof. Begin with the neighbourhood rows T, U, x, V. They name the source topology or separated uniform space, the inspected point, and the local neighbourhood inclusion before any compactness consumer can inspect the packet.

Use the compact-window rows K and F. The compact metric certificate of chapter 129 and the formal-ball interface of chapter 1510 provide finite windows only; the metric local-compactness sibling chapter 3787 is read as a displayed point-local comparison, not as an ambient theorem.

Finally replay through H, C, P, N. These rows keep the compact neighbourhood inside the same finite carrier route, so open-cover compactness, selected neighbourhood bases, quotient completion equality, global compactness, and ambient completeness have no row from which to escape. □

Theorem 3050.4 (LocallyCompactHausdorff separation transport)**.** *For every accepted* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ *packet, Hausdorff separation is transported only through the displayed row S over the same topology or separated-uniform source T and the same structural route H, C, P, N. If two point reads are moved by* hsame*, their separated-neighbourhood or separated-entourage witnesses are moved componentwise with the compact-neighbourhood rows; no host equality, quotient identification, global apartness principle, or classical point separation is added.*

Proof. Start with the separated source. When T is read through $\mathsf{HausdorffUniformSpace}^{\uparrow}$, the separated-uniform packet supplies the visible separation route; when it is read through $\mathsf{Topology}^{\uparrow}$, the row S displays the neighbourhood witnesses required for the inspected point reads.

Apply componentwise transport H. The transport row moves the source, neighbourhood, compact-window, formal-ball, and separation data together, so a separated read cannot be detached from the local compact carrier.

Finally use replay and provenance, C, P, N. They record the same separation route for downstream consumers and keep quotient equality, host point equality, global apartness, and classical separation outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactHausdorff}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ packet is generated from a topology or separated-uniform source, an inspected point, a local neighbourhood row, a compact-metric or formal-ball window, a Hausdorff separation row, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactHausdorff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4029.1, Definition 4029.2, Definition 4029.3, and Definition 4029.4 seed the constructive local-compactness and Hausdorff-separation packet over topology or separated-uniform sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactHausdorff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary neighbourhood bases, quotient completion equality, global local compactness, classical separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-neighbourhood nonescape, Hausdorff separation transport, finite-window replay, provenance, and local naming for BEDC.Derived.LocallyCompactHausdorffUp.

Chapter 3051

A Concrete Naming Certificate for $\mathsf{GleasonCover}^{\uparrow}$

Following Gleason and Kelley on projective covers of compact Hausdorff spaces, the $\mathsf{GleasonCover}^{\uparrow}$ packet records finite projective-cover, clopen-lift, compactification, and provenance rows. It sits beside the compact-target seal of chapter 2344, the lower-hit and upper-miss hyperspace route of chapter 2375, the local separation and compact-neighbourhood rows of chapter 4029, and the finite compact-metric witness surface of chapter 129. The packet names the traditional projective-cover boundary through displayed BEDC data; it does not import choice-selected irreducible maps, host Boolean algebras, quotient compacta, open-cover compactness, or ambient compact Hausdorff equality.

Definition 3051.1 (Gleason cover carrier)**.** *A* $\mathsf{GleasonCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, E, M, L, B, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact-Hausdorff source row, read through the displayed compact target seal, topology rows, and separation rows already available to compactification consumers.*
2. *E is the extremally disconnected projective-cover row, recording that the cover source is inspected only through finite projective-extension requests.*
3. *M is the cover-map row from E to X, including the finite surjective-read and irreducibility-test windows exposed to downstream compact Hausdorff consumers.*
4. *L is the clopen-lift row, recording finite clopen lift requests and their replay through the same cover map.*
5. *B is the compactification comparison row, comparing the cover map with the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *probe and compact-target rows.*
6. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *closed-set row, recording the finite lower-hit and upper-miss observations used when compact-subset consumers inspect the cover.*
7. *H records componentwise* hsame *transport for the compact-Hausdorff source, projective-cover row, cover-map row, clopen-lift row, compactification comparison row, Vietoris closed-set row, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from compact Hausdorff source reads through projective-cover, clopen-lift, compactification, and closed-set consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{GleasonCover}^{\uparrow}$, $\mathsf{StoneCechCompactification}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{LocallyCompactHausdorff}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, E, M, L, B, V, H, C, P, N$. It has no coordinate for arbitrary projective resolutions, selected ultrafilters, quotient equality of compacta, open-cover compactness, choice of clopen bases, or host compact Hausdorff spaces outside the displayed packet.

Lean4 ✓ `BEDC.Derived.GleasonCoverUp.GleasonCoverTasteGate single carrier alignment`

Theorem 3051.2 (Gleason cover projective-cover handoff)**.** *Every compact-Hausdorff consumer of an accepted* $\mathsf{GleasonCover}^{\uparrow}$ *carrier reads the cover only through the finite route*

$$X, E, M, L, B, V \longmapsto H, C, P, N.$$

The route exposes the compact-Hausdorff source, extremally disconnected projective-cover row, cover-map row, clopen-lift row, compactification comparison row, Vietoris closed-set row, componentwise transport, displayed replay, provenance, and local naming. It exports no open-cover selection, quotient compactum equality, selected ultrafilter, host Boolean algebra, arbitrary clopen basis, or ambient projective-cover theorem.

Proof. Project the carrier of Definition 4030.1. The compact-Hausdorff source is the row X, the projective-cover information is carried by E and M, and the clopen-lift and compactification comparisons are the rows L and B.

Read closed-set observations through V. This is the finite $\mathsf{VietorisTopology}^{\uparrow}$ side of the packet, so lower-hit and upper-miss consumers inspect only displayed closed-set windows rather than a host hyperspace or quotient compactum.

Finally replay through H, C, P, N. These rows transport, continue, source, and name the same carrier, so a compact-Hausdorff consumer has no coordinate from which to obtain open-cover selection, quotient equality, selected ultrafilters, host Boolean algebras, arbitrary clopen bases, or an ambient projective-cover theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$.

Constructive story: A $\mathsf{GleasonCover}^{\uparrow}$ packet is generated from compact-Hausdorff source rows, extremally disconnected projective-cover rows, cover-map rows, clopen-lift rows, compactification comparisons, Vietoris closed-set rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$, seed)

Scope closed: Definition 4030.1 and Definition 4030.2 seed the finite projective-cover handoff surface for compact-Hausdorff consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary projective resolutions, selected ultrafilters, quotient compactum equality, open-cover compactness, choice of clopen bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projective-cover handoff, clopen-lift replay, compactification comparison, Vietoris closed-set compatibility, provenance, and local naming for BEDC.Derived.GleasonCoverUp.

Chapter 3052

A Concrete Naming Certificate for HeineBorelNetCompactness$^{\uparrow}$

Following Bishop and Bridges on compact intervals, finite approximation, and located covers, the HeineBorelNetCompactness$^{\uparrow}$ packet records the finite-net compactness route for closed bounded real intervals as displayed NameCert data. It sits next to the finite-net cover surface of chapter 3471, the located subcover handoff of chapter 3538, the closed bounded interval compactness packet of chapter 3969, the compact metric carrier of chapter 129, the real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The packet gives a finite tolerance-and-net surface to completion and compactness consumers, not an unrestricted open-cover compactness theorem or a choice-selected subcover principle.

Definition 3052.1 (Heine–Borel net compactness carrier)**.** *A* HeineBorelNetCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located closed bounded interval row, recording the endpoint order, interval membership classifier, and interval-source readback.*
2. *F is the finite net row, read through* HeineBorelFiniteNet$^{\uparrow}$ *and* CompactMetric$^{\uparrow}$*, naming the displayed centers, mesh radius, cell coverage witnesses, and compact-metric provenance.*
3. *R is the positive-radius ledger, recording the requested positive tolerance, its dyadic lower witness, and the comparison surface used by cover readers.*
4. *W is the cover-read witness row, exposing only the finitely many cover predicates inspected at the displayed centers and interval boundary cells.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare mesh radius, endpoint error, cover radius, and finite-net budget.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row for interval endpoints and inspected cover radii, reached only after I, F, R, W, D have been displayed.*
7. *H records componentwise* hsame *transport for interval source, finite net, positive-radius ledger, cover reads, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the located interval row through finite nets, positive-radius checks, cover reads, dyadic tolerance comparisons, and terminal Real seals.*

9. *P is the* Pkg *provenance over* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, R, W, D, E, H, C, P, N$. It has no coordinate for unrestricted open-cover compactness, choice-selected finite subcovers, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineBorelNetCompactnessUp.HeineBorelNetCompactnessTasteGate single carrier alignment`

Theorem 3052.2 (Heine–Borel net compactness obligations)**.** *For every accepted* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ *carrier $K = (I, F, R, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$ *surface consists exactly of located interval admission, finite-net admission, positive-radius ledger exposure, finite cover-read witnesses, dyadic tolerance readback, Real-seal readback, componentwise transport, displayed continuation replay, provenance, and local naming. Every compactness-facing read must pass through the displayed rows before using a finite-net compactness conclusion.*

Proof. Project the carrier of Definition 4031.1. The tuple displays exactly the rows $I, F, R, W, D, E, H, C, P, N$, so the local classifier has no access to data outside that packet.

The interval and finite-net sides are read first. The located interval row I supplies the endpoint and membership surface, while F supplies the finite centers and coverage witnesses inherited from the finite-net and compact-metric siblings.

The cover-facing side is finite as well. The positive-radius ledger R fixes the requested tolerance, W records the inspected cover witnesses, D gives the dyadic comparison budget, and E seals only the endpoint and radius reads that are already attached to those rows.

Finally H, C, P, N transport, replay, source, and name the same route. Since no row contains open covers as an unrestricted object, selected subcovers, quotient interval equality, countable choice, or ambient real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3052.3 (Heine–Borel finite-subcover route)**.** *Every* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, ClosedBoundedIntervalC *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$$

through the displayed rows $I, F, R, W, D, E, H, C, P, N$. The route exports finite centers, positive-radius comparisons, cover-read witnesses, dyadic tolerance rows, endpoint and radius seals, and structural provenance only; it exports no unrestricted open-cover compactness, choice-selected subcover, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface of Definition 4031.2. The interval row and finite-net row must be displayed before a compactness consumer can inspect a cover witness.

Next read the tolerance side. The positive-radius ledger R and dyadic tolerance row D control the requested mesh and cover radius, while W exposes only the finite cover reads attached to the displayed centers and interval boundary cells.

The terminal Real row E seals those endpoint and radius reads without adding a completeness theorem. The transport, replay, provenance, and local naming rows H, C, P, N preserve the same finite route for each sibling consumer.

Therefore the public handoff is exactly the finite-subcover route displayed in the statement. The excluded open-cover, selected-subcover, quotient, choice, and ambient-completeness data have no carrier row through which they could be exported. □

Falsifiable prediction: A compactness-facing consumer that reads $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ without displaying the located interval, finite net, positive-radius ledger, cover-read witness, DyadicRatCore tolerance, Real seal, componentwise transport, continuation replay, package provenance, and local NameCert rows refutes the carrier surface.

Independence witness: heineborel-finite-net, located-heine-borel-subcover, closed-bounded-interval-compact, compactmetric, real, dyadicratcore: no listed sibling is bijective to $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ because this packet binds interval location, finite-net coverage, positive-radius ledgers, cover-read witnesses, dyadic tolerance, Real seals, transport, replay, provenance, and local naming in one compactness-facing interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ packet is generated from a located interval row, finite net row, positive-radius ledger, cover-read witnesses, DyadicRatCore tolerance row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4031.1, Definition 4031.2, and Definition 4031.3 seed the finite-net compactness interface for closed bounded interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted open-cover compactness, choice-selected finite subcovers, quotient interval equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net admission, positive-radius ledger exposure, cover-read witnesses, dyadic tolerance readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelNetCompactnessUp.

Chapter 3053

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionUnit}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ packet records the completion-unit surface for separated Hausdorff completion as finite NameCert data. It sits after the uniform-completion carrier of chapter 1473, the metric-completion carrier of chapter 1506, the Hausdorff-completion carrier of chapter 402, the separated-metric boundary of chapter 1411, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the unit map into the separated Hausdorff completion without importing quotient completion equality, selected representatives, or ambient completeness.

Definition 3053.1 (HausdorffCompletionUnit carrier)**.** *A* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, S, J, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric or uniform row, naming the finite observation surface consumed before completion.*
2. *S is the separated Hausdorff completion row, read through the separated-metric and Hausdorff-completion sibling carriers.*
3. *J is the unit-map row from the source surface into the separated Hausdorff completion surface.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect source names before the unit map is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular Cauchy tolerance names attached to W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for distance or apartness readbacks reached after M, S, J, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, completion, unit, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from source inspection to unit-map handoff, completion readback, and real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion points, or ambient completeness.

Lean4 (definition) `BEDC.Derived.HausdorffCompletionUnitUp.HausdorffCompletionUnitUp`

Theorem 3053.2 (HausdorffCompletionUnit NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *carrier* $U = (M, S, J, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$ *obligations are exactly source metric or uniform exposure through* M*, separated Hausdorff completion exposure through* S*, unit-map exposure through* J*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window inspection through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every obligation is discharged along the displayed route

$$M \longmapsto S \longmapsto J \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No obligation may be replaced by quotient completion equality, selected representatives, countable choice, host equality of completion points, or ambient completeness.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUnitUp.HausdorffCompletionUnitNamecertObligations`

Proof. Project the carrier of Definition 4032.1. The source row M, separated Hausdorff completion row S, unit row J, finite windows W, readback row R, seal row E, transport row H, replay row C, provenance row P, and local name N are exactly the displayed coordinates.

Read the completion-unit route in order. The source metric or uniform row is visible before the separated Hausdorff completion row, and the unit-map row is read only after both source and completion surfaces have been displayed.

The numerical side is downstream. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks expose finite regular Cauchy data before the $\mathsf{Real}^{\uparrow}$ seal is consumed. Rows H, C, P, N transport, replay, source, and name that same finite route.

Since the packet contains no quotient equality, selected representative, choice schedule, host completion-point equality, or ambient-completeness coordinate, the displayed rows exhaust the local NameCert obligations. □

Theorem 3053.3 (HausdorffCompletionUnit Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *packet reads the source metric or uniform row, separated Hausdorff completion row, unit-map row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local NameCert rows before receiving the completion handoff. The handoff is the finite unit route*

$$M, S, J, W, R, E, H, C, P, N$$

and not a quotient completion class, selected representative, countable-choice schedule, host equality principle, or ambient complete-space witness.

Proof. Begin with the NameCert obligations of Definition 4032.2. They force the consumer to read $M, S, J, W, R, E, H, C, P, N$ as one displayed certificate surface.

Use the sibling dependencies through their visible rows. The uniform-completion, metric-completion, Hausdorff-completion, and separated-metric chapters supply completion vocabulary and separated comparison, but the unit packet receives those surfaces only through M, S, J and then through the finite window and readback rows.

The Cauchy-facing data remains finite. The $\mathsf{StreamName}^{\uparrow}$ row displays the inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row displays the tolerance readback, and the $\mathsf{Real}^{\uparrow}$ seal is terminal rather than a detached completeness principle.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and naming. No excluded coordinate is present in the carrier, so the completion-facing handoff is exactly the finite unit route. □

Falsifiable prediction: Every $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ completion-unit read factors through source metric or uniform data, separated Hausdorff completion data, the unit-map row, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming. A read that reaches the completion handoff without $M, S, J, W, R, E, H, C, P, N$ falsifies this packet.

Independence witness: uniformcompletion, metriccompletion, hausdorffcompletion, separated-metric, streamname, regseqrat, real: $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source surface, separated Hausdorff completion surface, unit-map row, finite windows, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ packet is generated from a source metric or uniform row, separated Hausdorff completion row, unit-map row, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4032.1, Definition 4032.2, and Definition 4032.3 seed the finite completion-unit surface for separated Hausdorff completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, separated Hausdorff completion exposure, unit-map handoff, StreamName and RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffCompletionUnitUp.

Chapter 3054

A Concrete Naming Certificate for $\mathsf{CauchyCriterionForSeries}^{\uparrow}$

Following Cauchy and Knopp, the $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ packet records the constructive criterion that a real series converges exactly when its finite tails become arbitrarily small under displayed regular-rational windows. It sits after the series carrier of chapter 1581, the Cauchy sequence algebra of chapter 3708, the stream-name window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite tail route for series convergence; it is not a host infinite sum, quotient stream equality, selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3054.1 (Cauchy criterion for series carrier)**.** *A* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, L, W, D, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, displaying the inspected series terms and partial-sum prefix.*
2. *L is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, naming the partial-sum algebra and its finite sum/difference operations.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite tail interval and every inspected partial-sum index.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the positive dyadic radius used by the tail comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed tail window and dyadic tolerance.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the series, algebra, window, tolerance, and readback rows have been replayed.*
7. *T is the tail-smallness ledger, recording the finite witnesses that all displayed tails beyond the named threshold are inside the tolerance in D.*
8. *H records componentwise* hsame *transport for A, L, W, D, R, E, T and for the continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a convergence request through the source series, partial-sum algebra, finite window, dyadic tolerance, regular rational readback, tail-smallness ledger, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, L, W, D, R, E, T, H, C, P, N$. It has no coordinate for an infinite host sum, quotient equality of partial-sum streams, selected limit, countable-choice modulus, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionForSeriesUp.CauchyCriterionForSeriesTasteGate single carrier alignment`

Theorem 3054.2 (Cauchy criterion for series NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *carrier $S = (A, L, W, D, R, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure,* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *partial-sum algebra exposure,* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, tail-smallness ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4033.1. The tuple displays the real-series source, partial-sum algebra, finite windows, dyadic tolerance, regular rational readback, real seal, tail-smallness ledger, transport, continuation, provenance, and local name.

The analytic content is read through A, L, W, D, R, E, T. The series row supplies terms and partial sums; the algebra row supplies finite partial-sum differences; the window and tolerance rows state the inspected tail request; and the readback, seal, and ledger rows certify that the displayed tail is small.

The structural rows H, C, P, N preserve and replay the same route. Since the carrier contains no coordinate for host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient completeness, or bridge checking, the listed rows exhaust the local certificate. □

Theorem 3054.3 (Cauchy criterion for series tail route)**.** *Every accepted convergence read of a* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *packet factors through the finite tail route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{CauchySequenceAlgebra}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported criterion is the assertion that the partial-sum tail ledger T can answer each displayed dyadic tolerance request after the source series, partial-sum algebra, finite window, regular readback, and real seal have been displayed. It exports no host infinite sum, quotient partial-sum stream, selected limit, countable-choice modulus, trichotomy, or ambient completeness theorem.

Proof. Begin with the source series row A and partial-sum algebra row L. These rows provide only finite term and partial-sum data, so a convergence consumer has not yet received a completed sum.

Next read the finite request rows W and D. The stream-name row displays the inspected tail window, and the dyadic row names the tolerance for that window.

The regular rational readback R, real seal E, and tail-smallness ledger T then certify the displayed tail comparison. The continuation row C fixes this order, and H, P, N transport, source, and name the same packet.

Because every consumer reaches the criterion through this route, any read that asks for quotient stream equality, selected limits, countable choice, trichotomy, or ambient real completeness has left the packet and is not part of the exported criterion. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ convergence read must expose a $\mathsf{RealSeries}^{\uparrow}$ source row, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ partial-sum algebra row, $\mathsf{StreamName}^{\uparrow}$ tail window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, tail-smallness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before the criterion is available.

Independence witness: Nearest siblings: realseries, cauchysequencealgebra, streamname, dyadicratcore, regseqrat, and real. The packet is not bijective to any one sibling because it jointly binds source series rows, partial-sum algebra, tail windows, dyadic tolerances, regular rational readback, real sealing, tail-smallness ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ packet is generated from a RealSeries source row, CauchySequenceAlgebra partial-sum row, StreamName tail-window row, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, tail-smallness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4033.1, Definition 4033.2, and Definition 4033.3 seed the finite constructive Cauchy criterion route for real series.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, partial-sum algebra exposure, tail-window exactness, dyadic tolerance exactness, RegSeqRat readback, RealUp seal handoff, tail-smallness ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCriterionForSeriesUp.

Chapter 3055

A Concrete Naming Certificate for $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$

The $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ packet realizes the boundary-weighting route of chapter 9463 as finite NameCert data. It sits after the Poisson-kernel certificate of chapter 3925, the visual-metric certificate of chapter 3866, the Busemann-boundary certificate of chapter 3732, the hyperbolic-metric certificate of chapter 3474, the disk carrier of chapter 377, and the terminal real-name seal of chapter 38. The packet names a finite harmonic boundary-measure read; it is not a host measure space, sigma-algebra completion, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary construction.

Definition 3055.1 (Hyperbolic harmonic boundary measure carrier)**.** *A* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, B, V, K, R, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row naming the finite interior observation from which boundary weighting is read.*
2. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row naming the boundary face and its Busemann-level comparison data.*
3. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row naming the visual boundary distance used by the weighting request.*
4. *K is the* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *row, displaying the Poisson-kernel weight over the same disk, boundary, visual-metric, and hyperbolic-metric rows.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed boundary weight.*
6. *W is the harmonic boundary-measure ledger, recording the finite boundary-weight rows accepted by the Poisson kernel before any measure read is exported.*
7. *H records componentwise* hsame *transport for D, B, V, K, R, W and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from disk-coordinate admission through Busemann-boundary admission, visual-metric readback, Poisson-kernel weighting, real sealing, and harmonic boundary-measure ledger exposure.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$, $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, B, V, K, R, W, H, C, P, N$. It has no coordinate for a host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicHarmonicBoundaryMeasureUp.HyperbolicHarmonicBoundaryMeasureTasteGate`
`single carrier alignment`

Theorem 3055.2 (Hyperbolic harmonic boundary measure NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ *carrier $M = (D, B, V, K, R, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, Busemann-boundary exposure, visual-metric readback, Poisson-kernel weighting,* $\mathsf{Real}^{\uparrow}$ *seal handoff, harmonic boundary-measure ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4034.1. The projection exposes the disk row, boundary row, visual-metric row, Poisson-kernel row, real seal, harmonic boundary-measure ledger, and the structural rows.

The geometric read is ordered by D, B, V, K, R, W. A boundary-measure consumer first sees the disk coordinate, then the Busemann boundary face, then the visual metric, and only then the Poisson-kernel weight and real seal used by the ledger.

The rows H, C, P, N transport, replay, source, and name this finite route. Since the carrier contains no host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, Lean verification, or bridge checking, the listed rows exhaust the NameCert surface. □

Theorem 3055.3 (Hyperbolic harmonic boundary measure Poisson route)**.** *Every accepted harmonic boundary-measure read of a* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicPoi}$$

The exported boundary measure is the Poisson-kernel-weighted ledger W over the same disk and boundary rows. It exports no completed host measure, sigma-algebra completion, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary object.

Proof. Open the disk-coordinate row D and the Busemann-boundary row B. These rows fix the finite interior and boundary observations before any weight is inspected.

Read the visual metric and Poisson kernel next. The row V supplies the boundary-distance comparison, and K supplies the Poisson-kernel weight over the same hyperbolic metric and boundary route.

The real seal R and measure ledger W then record the displayed harmonic boundary weight. The rows H, C, P, N preserve the route under transport, replay it for consumers, record provenance, and attach the local name.

Thus every accepted measure read passes through the Poisson-kernel route. The carrier has no coordinate from which a host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary construction could be exported. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ read must pass through $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before it exports a boundary-weight ledger.

Independence witness: Nearest siblings: hyperbolic-poisson-kernel, hyperbolic-visual-metric, hyperbolic-busemann-boundary, hyperbolic-metric, and unitdisk. The packet is not bijective to any one sibling because it jointly binds disk-coordinate rows, Busemann boundary rows, visual metric readback, Poisson-kernel weighting, RealUp sealing, harmonic boundary-measure ledger exposure, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric route data, HyperbolicBusemannBoundary rows, HyperbolicVisualMetric rows, HyperbolicPoissonKernel weights, a RealUp seal, a harmonic boundary-measure ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4034.1, Definition 4034.2, and Definition 4034.3 seed the finite harmonic boundary-measure route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host measure spaces, completed sigma algebras, selected boundary representatives, ambient probability theorems, global field instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-coordinate exposure, Busemann-boundary exposure, visual-metric readback, Poisson-kernel weighting, RealUp seal handoff, ledger stability, transport, replay, provenance, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.HyperbolicHarmonicBoundaryMeasureUp.

Chapter 3056

A Concrete Naming Certificate for $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$

The $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ packet realizes the support-nerve census and codon-lifted Betti table of subsection 9462.28.21 as finite NameCert data. It consumes the partition-core incidence surface of chapter 4017, the simplicial-complex carrier of chapter 238, the homology carrier of chapter 98, the finite simplicial-chain surface of chapter 3414, the Hochster–nerve reduction of chapter 3744, and the DNA boundary-tile source of chapter 3393. The packet names a finite support-to-nerve realization and lifted Betti census; it is not a host minimal-resolution oracle, stochastic biological model, quotient codon semantics, unrestricted homology computation, or coefficient-field completeness theorem.

Definition 3056.1 (SupportNerveCensusLiftedBetti carrier)**.** *A* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, U, V, F, H_0, B, L, G, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *row naming the finite support-family incidence table used to enumerate the* 15 *nonempty supports.*
2. *U is the support-to-nerve realization row, assigning each accepted support family to the four-vertex nerve obtained from its displayed supports.*
3. *V is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *row naming the resulting nerve as a finite complex.*
4. *F is the* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *row exposing the chain boundary surface used for reduced homology reads.*
5. *H_0 is the* $\mathsf{Homology}^{\uparrow}$ *row for reduced homology in degrees* $-1, 0, 1, 2$.
6. *B is the lifted Betti ledger carrying the finite census counts and the codon lift substitution* $z \mapsto 2x + x^2$.
7. *L is the* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *row that keeps the support-level nerve census upstream of any Betti-strand export.*
8. *G records componentwise* hsame *transport for the incidence, realization, complex, chain, homology, lifted Betti, reduction, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from support-family reads to nerve realization, finite homology, and lifted Betti ledgers.*

10. *P is the* Pkg *provenance over* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$, $\mathsf{PartitionCoreIncidence}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteSimplicialChain}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, U, V, F, H_0, B, L, G, C, P, N$. It has no coordinate for hidden codon semantics, unrestricted minimal resolutions, ambient Betti-table equality, stochastic evolutionary interpretation, quotient support equality, or an external homology oracle.

Lean4 ✓ `BEDC.Derived.SupportNerveCensusLiftedBettiUp.SupportNerveCensusLiftedBettiTasteGate`
`single carrier alignment`

Theorem 3056.2 (SupportNerveCensusLiftedBetti NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *carrier $S = (I, U, V, F, H_0, B, L, G, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLifted}}$ *surface consists exactly of incidence enumeration, support-to-nerve realization, finite simplicial-complex admission, finite chain-boundary exposure, reduced homology readback in degrees $-1, 0, 1, 2$, lifted Betti ledger exposure, Hochster–nerve reduction provenance, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4035.1. The incidence, realization, complex, chain, homology, lifted Betti, reduction, transport, replay, provenance, and naming rows are exactly $I, U, V, F, H_0, B, L, G, C, P,$

The support-to-nerve route starts from I and U. It can expose a nerve only through the finite $\mathsf{SimplicialComplex}^{\uparrow}$ row V, so the census cannot be replaced by a host powerset calculation or a quotient support object.

The homology and Betti rows are downstream of the finite chain boundary F. The row H_0 restricts homology reads to the displayed reduced degrees, while B records the census and lift substitution only as finite ledger data.

Finally L, G, C, P, N preserve, replay, source, and name the same route. Since no remaining coordinate stores hidden codon semantics, unrestricted resolutions, host Betti-table equality, stochastic interpretation, quotient support equality, or external homology computation, the listed obligations exhaust the local NameCert surface. □

Theorem 3056.3 (SupportNerveCensusLiftedBetti realization)**.** *Every accepted* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{PartitionCoreIncidence}^{\uparrow} \longrightarrow \mathsf{SimplicialComplex}^{\uparrow} \longrightarrow \mathsf{FiniteSimplicialChain}^{\uparrow} \longrightarrow \mathsf{Homology}^{\uparrow} \longrightarrow \mathsf{HochsterNerveBettiRe}$$

The exported census is the displayed support-family count, reduced-homology signature table, and lifted Betti ledger, not a biological causal model, unrestricted coefficient-field theorem, ambient minimal resolution, or host-level homology calculation.

Proof. Start from the NameCert obligations in Definition 4035.2. They force support-family enumeration through I and support-to-nerve realization through U before any complex row is available.

Next the finite complex, chain, and homology rows V, F, H_0 expose the reduced homology signatures used by the census. These rows are displayed finite data, so the read cannot invoke an external simplicial-complex oracle or an unrestricted homology computation.

The lifted Betti ledger B and Hochster–nerve row L then package the codon lift as a finite strand read. Transport, replay, provenance, and local naming preserve that same route, which proves the stated realization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ read factors through displayed partition-core incidence, support-to-nerve realization, simplicial-complex, finite-chain, homology, lifted Betti, Hochster–nerve reduction, transport, replay, provenance, and local naming rows.

Independence witness: partition-core-incidence, support-to-nerve-realization, simplicialcomplex, finite-simplicial-chain, homology, hochster-nerve-betti-reduction: $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the support-family census, nerve realization, finite homology signature table, codon-lifted Betti ledger, and coefficient TasteGate boundary in one packet.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ SupportNerveCensusLiftedBetti}^{\uparrow}$ and $\mathsf{FieldFaithful\ SupportNer}$ TasteGate instances. Those instances must distinguish $I, U, V, F, H_0, B, L, G, C, P, N$ so the support census, nerve realization, and lifted Betti ledger cannot collapse into one sibling row.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}$.

Constructive story: A $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ packet is generated from partition-core incidence rows, support-to-nerve realization rows, finite simplicial-complex rows, finite chain rows, reduced homology rows, lifted Betti ledgers, Hochster–nerve reduction provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4035.1, Definition 4035.2, and Definition 4035.3 seed the finite support-nerve census and lifted Betti packet over incidence, realization, simplicial-complex, chain, homology, Betti-ledger, reduction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, biological mechanism, unrestricted minimal resolutions, quotient codon semantics, coefficient-field completeness, or ambient homology computation.

Upgrade path: Obligation closure requires checked incidence enumeration, support-to-nerve realization, finite-complex admission, finite-chain exposure, homology readback, lifted Betti ledger stability, TasteGate nontriviality, field faithfulness, replay stability, and ledger nonescape for BEDC.Derived.SupportNerveCensusLiftedBettiUp.

Chapter 3057

A Concrete Naming Certificate for $\mathsf{EquicontinuousFamily}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Arzela–Ascoli finite-family reading, the $\mathsf{EquicontinuousFamily}^{\uparrow}$ packet records the shared-modulus interface by which a finite family of continuous maps over one compact metric source exposes a single displayed modulus row to downstream compact-uniform-continuity and Ascoli consumers. It sits after the compact source of chapter 129, the graph rows of chapter 130, the uniform-modulus consumer surface of chapter 369, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet does not introduce open-cover compactness, a host function-space topology, quotient equality of maps, or choice-selected pointwise moduli.

Definition 3057.1 (Equicontinuous family carrier)**.** *An* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, F, J, M, S, R, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the shared* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite metric reads and compact finite-net source used by the family.*
2. *F is the finite ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows whose source coordinate is X and whose values are read only through displayed graph windows.*
3. *J is the finite family-index row, naming exactly the family members admitted by F.*
4. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, recording one tolerance-to-radius ledger used uniformly for every index in J.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule through which the source and graph reads are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same tolerance and radius histories.*
7. *V is the* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after the graph, schedule, regular rational, and shared-modulus rows have been displayed.*
8. *H records componentwise* hsame *transport for the compact source, finite graph ledger, family index row, shared modulus row, observation schedule, regular rational readback, Real seal, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-source reads to graph reads, from tolerance requests to the shared modulus row, and from scheduled readback to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, J, M, S, R, V, H, C, P, N$. It has no coordinate for an open cover, a selected global modulus outside M, a host function-space topology, quotient equality of functions, or a choice principle over the family.

Lean4 ✓ `BEDC.Derived.EquicontinuousFamilyUp.EquicontinuousFamilyTasteGate single carrier alignment`

Theorem 3057.2 (Equicontinuous family NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier $E = (X, F, J, M, S, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousFamily}^{\uparrow}}$ *surface consists of carrier admission, compact-source admission, finite continuous-map ledger admission, family-index totality, shared uniform-modulus exposure, StreamName and RegSeqRat readback, Real-seal handoff, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged entirely by the finite route*

$$X, F, J \longmapsto M \longmapsto S, R \longmapsto V$$

together with H, C, P, N.

Proof. Carrier admission is the projection of Definition 4036.1. The shared compact source row X, finite graph ledger F, family index row J, shared modulus row M, schedule row S, regular readback row R, and Real seal V are all visible coordinates of one finite packet.

The shared-modulus obligation is read before any Ascoli or compact-uniform-continuity consumer inspects the family. Each member named by J is admitted through F, and the same tolerance-to-radius ledger M is used for that member over the source row X.

The rows S, R, V provide the finite observation and value boundary. They keep the scheduled source and graph reads attached to the same regular rational readback and Real seal, rather than to a host function-space object.

Transport and replay are supplied by H and C, while P and N record provenance and local naming. Since the carrier has no row for open-cover compactness, quotient function equality, selected pointwise moduli, or ambient topology, the listed obligations exhaust the certificate surface. □

Theorem 3057.3 (Equicontinuous family uniform-modulus handoff)**.** *Any* $\mathsf{UniformModulus}^{\uparrow}$*, compact-uniform-continuity, or Arzela–Ascoli consumer of an accepted* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier must receive the shared modulus row M together with the compact source row X, finite continuous-map ledger F, family index row J, observation schedule S, regular rational readback R, Real seal V, and structural rows H, C, P, N. The handoff exports only the displayed finite shared-modulus route and does not supply open-cover compactness, a function-space topology, quotient equality of maps, or choice-selected moduli.*

Proof. Begin with the carrier of Definition 4036.1. It displays the shared source, finite graph ledger, finite family index, shared modulus, schedule, readback, seal, transport, replay, provenance, and local naming rows as the whole packet.

Apply the NameCert obligations of Definition 4036.2. They force every accepted read through $X, F, J, M, S, R, V, H, C, P, N$, with M placed before the consumer can treat the family as equicontinuous.

The downstream consumer may add its own Ascoli, compact-uniform-continuity, or selection rows after this boundary. It cannot replace the displayed shared modulus by an open cover, host topology, quotient function equality, or a choice-selected family of pointwise moduli, because none of those objects is a carrier coordinate. □

Theorem 3057.4 (Equicontinuous family compact-uniform-modulus handoff)**.** *Any* CompactEquicontinuousFamily *consumer of an accepted* EquicontinuousFamily$^{\uparrow}$ *carrier*

$$E = (X, F, J, M, S, R, V, H, C, P, N)$$

must pass through $X, F, J, M, S, R, V, H, C, P, N$ *before the compact-family uniform budget is public. In particular, the compact-family read may consume the shared modulus only after the compact source, finite family ledger, family index, shared modulus row, StreamName schedule, RegSeqRat readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have all been displayed.*

Proof. Start with the uniform-modulus handoff of Definition 4036.3. It forces every shared-modulus consumer through the displayed rows $X, F, J, M, S, R, V, H, C, P, N$.

Now read the carrier route of CompactEquicontinuousFamilyModulus$^{\uparrow}$ from chapter 6234. Its compact-family budget consumes the EquicontinuousFamily$^{\uparrow}$ row as the family source before the compact-uniform output can be public.

Composing the two routes leaves no shortcut around the equicontinuous-family carrier. The compact-family consumer may add its own compact finite-net and output rows downstream, but the shared family budget remains the finite route stated above. □

Closure status for NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$**.**

Constructive story: An EquicontinuousFamily$^{\uparrow}$ packet is generated from a compact metric source row, a finite continuous-map family ledger, a finite family-index row, a shared uniform-modulus row, StreamName observations, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$, seed)

Scope closed: Definition 4036.1, Definition 4036.2, Definition 4036.3, and Definition 4036.4 seed the finite shared-modulus interface consumed by compact-uniform-continuity and Ascoli-style chapters.

Formal verification: FormalStatus(NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.EquicontinuousFamilyUp.EquicontinuousFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, a host function-space topology, quotient function equality, or choice-selected pointwise moduli.

Upgrade path: Obligation closure requires checked carrier admission, family-index totality, shared modulus exposure, finite readback, Real-seal handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.EquicontinuousFamilyUp.

Chapter 3058

A Concrete Naming Certificate for $\mathsf{BishopRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ packet records the finite comparison modulus by which a Bishop real comparison request is reduced to inspected regular-Cauchy windows, positive apartness evidence, and dyadic tolerance rows. It extends the Bishop real source of chapter 343, the comparison surface of chapter 1381, the apartness witness of chapter 1881, and the L10 readback route of chapter 301 without importing trichotomy, quotient reals, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3058.1 (Bishop real comparison modulus carrier)**.** *A* $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, A, R, S, D, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BishopReal}^{\uparrow}$ *row exposing the regular-Cauchy real names and their finite comparison-facing source surface.*
2. *A is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *witness row supplying positive apartness data whenever the comparison request has a separated outcome.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to inspect the rational values of the requested finite windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming exactly the finite windows read by the comparison request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row storing the dyadic precision, slack, and positive-gap ledger used by the modulus.*
6. *M is the comparison modulus row; it maps each displayed comparison request to the finite window and dyadic tolerance budget consumed by R, S, D.*
7. *H records componentwise* hsame *transport for B, A, R, S, D, M and for the structural certificate rows.*
8. *C records displayed* Cont *replay from a comparison request to the window, readback, tolerance, apartness, and modulus rows.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopReal}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $B, A, R, S, D, M, H, C, P, N$. It has no coordinate for quotient real equality, total order trichotomy, a selected limit, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRealComparisonModulusUp.BishopRealComparisonModulusTasteGate single carrier alignment`

Theorem 3058.2 (Bishop real comparison modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ *carrier*

$$Q = (B, A, R, S, D, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Bishop-real source, apartness witness, regular readback, stream schedule, dyadic tolerance, comparison modulus, transport, replay, provenance, and local-name rows;*
2. *modulus readback, requiring each comparison request to expose through M the finite S windows, R rational observations, and D dyadic tolerance budget it consumes;*
3. *apartness handoff, requiring every positive separated outcome to pass through the displayed* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A before it is offered to a real-facing consumer;*
4. *regular-sequence compatibility, requiring the rational observations in R to be read from the same Bishop-real source surface named by B;*
5. *finite stream-window compatibility, requiring S to name only the windows requested by M and replayed by C;*
6. *dyadic tolerance exactness, requiring D to store precisely the comparison precision, slack, and positive-gap data used by M and A;*
7. *componentwise* hsame *transport for $B, A, R, S, D, M, C, P, N$ through H;*
8. *displayed* Cont *replay of each comparison request through the window, readback, tolerance, apartness, and modulus rows;*
9. Pkg *provenance for exactly the displayed packet;*
10. *ledger non-escape, requiring every consumer to remain inside $B, A, R, S, D, M, H, C, P, N$.*

Consequently the packet exports a finite comparison modulus interface for Bishop real names, not trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4037.1. The projection exposes exactly the Bishop source, apartness witness, regular readback, stream schedule, dyadic tolerance, comparison modulus, transport, replay, provenance, and local naming rows.

The modulus obligation is read from M together with S, R, D. A comparison request first names its finite stream windows, then reads the corresponding regular rational observations, and finally displays the dyadic precision and slack ledger used by the comparison.

The positive separated branch is forced through A. Since A is the only apartness witness row, the carrier cannot turn a modulus read into trichotomy, quotient equality, or a selected real representative.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. They do not add any coordinate outside B, A, R, S, D, M, so the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ packet is generated from a BishopReal row, a CauchyRealApartness witness, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a comparison modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4037.1 and Definition 4037.2 seed the finite comparison-modulus interface over Bishop real source rows, apartness witnesses, regular readbacks, stream windows, and dyadic tolerance data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, modulus readback, apartness handoff, regular-sequence compatibility, finite window compatibility, dyadic tolerance exactness, transport, replay, provenance, and ledger non-escape.

Chapter 3059

A Concrete Naming Certificate for $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ packet records the Real-facing interface of an Archimedean Cauchy-complete ordered field as finite BEDC NameCert data. It reads the terminal Real seal of chapter 38, the regular rational sequence readback of chapter 301, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the algebraic order interface of chapter 766, the Cauchy-complete ordered-field route of chapter 3688, and the comparison capstone of chapter 2025. The chapter provides one L10 phase-exit surface for Real and Completion consumers without importing a host complete ordered field.

Definition 3059.1 (Archimedean Cauchy-complete field carrier)**.** *An* $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, Q, D, S, O, K, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal, read only as the terminal Real name row.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational names.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for displayed tolerance and radius requests.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which rational observations are requested.*
5. *O is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation-and-order row, naming addition, multiplication, zero, one, positivity, and order comparison only as finite ledger entries.*
6. *K is the* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *route row tying the Real seal, regular readback, dyadic ledger, stream window, and algebraic-order row to displayed Cauchy-complete ordered-field service.*
7. *F is the* $\mathsf{CompleteArchimedeanField}^{\uparrow}$ *comparison row, recording the Archimedean comparison surface reached from the same displayed Cauchy-complete ordered-field route.*
8. *H records componentwise* hsame *transport for the Real seal, regular readback, dyadic ledger, stream window, algebra-order row, Cauchy-complete ordered-field route, Archimedean comparison row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite precision request through stream observation, regular rational readback, Real sealing, order operation, Cauchy-complete route, Archimedean comparison, and local naming.*

10. *P is the* Pkg *provenance over* ArchimedeanCauchyCompleteField$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchimedeanField$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* NameCert$_{\text{ArchimedeanCauchyCompleteField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, D, S, O, K, F, H, C, P, N$. It has no coordinate for quotient Real equality, trichotomy, selected limits, host supremum data, or ambient completeness.

Lean4 (definition) `BEDC.Derived.ArchimedeanCauchyCompleteFieldUp.ArchimedeanCauchyCompleteFieldUp`

Theorem 3059.2 (Archimedean Cauchy-complete field NameCert obligations)**.** *For every accepted* ArchimedeanCauchyCompleteField$^{\uparrow}$ *carrier*

$$A = (R, Q, D, S, O, K, F, H, C, P, N),$$

the local NameCert$_{\text{ArchimedeanCauchyCompleteField}^{\uparrow}}$ *surface consists exactly of carrier admission for the Real scalar seal R, regular rational readback Q, dyadic precision ledger D, finite stream window S, RealAlgOrder operation-and-order row O, CauchyCompleteOrderedField route row K, CompleteArchimedeanField comparison row F, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 4038.1. The accepted tuple exposes exactly the Real seal, regular readback, dyadic ledger, finite stream window, algebra-order row, Cauchy-complete ordered-field route, Archimedean comparison row, transport, replay, provenance, and local naming rows.

The scalar and completion-facing part is read through R, Q, D, S, K. The row K is allowed to cite the Cauchy-complete ordered-field surface only after the Real seal, regular readback, dyadic precision, and stream-window rows have been displayed.

The order and Archimedean comparison part is read through O and F. These rows record operation, order, and comparison service as BEDC packet rows rather than as a host complete ordered-field structure.

Finally H, C, P, N transport, replay, source, and name that finite surface. Since no other carrier coordinate remains, the listed rows exhaust the NameCert obligations. □

Theorem 3059.3 (Archimedean Cauchy-complete field L10 exit route)**.** *Every L10 Real or Completion consumer of an accepted* ArchimedeanCauchyCompleteField$^{\uparrow}$ *packet must factor its read through*

Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchin

as the displayed rows $R, Q, D, S, O, K, F, H, C, P, N$. Thus the packet supplies a single Archimedean Cauchy-complete ordered-field interface for Real-facing phase exit, but it exports no quotient Real equality, trichotomy, selected limit, host complete ordered field, or ambient completeness coordinate.

Proof. Start with an accepted consumer read. The Real scalar is exposed only by R, and the rational Cauchy evidence behind the read must pass through Q, D, S before any completion-facing row is visible.

The algebraic and order part is then confined to O. Addition, multiplication, zero, one, positivity, and comparison are read as displayed operation-and-order rows, not as host-field operations.

The completion route is K, and the Archimedean comparison capstone is F. These rows link the same Real seal, regular readback, dyadic ledger, stream window, and order row to the Cauchy-complete ordered-field and CompleteArchimedeanField packets.

The structural rows H, C, P, N preserve and name this route. Because the carrier lacks quotient Real equality, trichotomy, selected limit, host-field, and ambient-completeness coordinates, every L10 phase-exit read is confined to the finite surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ packet is generated from a Real scalar seal, RegSeqRat readback, DyadicRatCore precision ledger, StreamName finite window, RealAlgOrder operation and order row, CauchyCompleteOrderedField route row, CompleteArchimedeanField comparison row, componentwise transport, displayed replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4038.1, Definition 4038.2, and Definition 4038.3 seed the L10 Real and Completion phase-exit surface for Archimedean Cauchy-complete ordered-field consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Real equality, trichotomy, selected limits, host complete ordered fields, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rowwise operation and order service, Archimedean comparison service, transport, replay, provenance, and local naming for the same finite packet.

Chapter 3060

A Concrete Naming Certificate for $\mathsf{ProductUniformSpace}^{\uparrow}$

Following Bourbaki and Weil's product uniformity language, the $\mathsf{ProductUniformSpace}^{\uparrow}$ packet records the finite route by which two displayed uniform source packets are assembled into a product uniformity without importing a host product topology. It reads the $\mathsf{UniformSpace}^{\uparrow}$ entourage surface of chapter 339, the **??** projection route of chapter 34, the metric-compatible source of chapter 57, and the completion-facing boundary of chapter 95. The chapter names the product-entourage rows needed by completion consumers before any quotient completion, unrestricted entourage choice, or ambient topology is visible.

Definition 3060.1 (Product uniform space carrier)**.** *A* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, B, \Pi, E_A, E_B, W, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two uniform source packets. Each source exposes only its displayed carrier, entourage ledger, diagonal row, symmetry row, composition-refinement row, transport row, replay row, provenance row, and local* $\mathsf{NameCert}$ *row.*

2. *Π is the **??** projection row. It records the left and right projection reads and the paired point-name admission used by the product packet.*

3. *E_A and E_B are the paired entourage ledgers. A product entourage request must name a left source entourage from A and a right source entourage from B before it can be read as a product request.*

4. *W is the finite componentwise witness row. It records that a product pair is accepted exactly by its two projected source reads and their paired entourage ledgers.*

5. *H records componentwise* hsame *transport for the two source packets, the projection row, the paired entourage ledgers, the witness row, replay, provenance, and local naming rows.*

6. *C records displayed* Cont *replay from a product-entourage request through the projection row, the left and right source entourages, the componentwise witness, transport, provenance, and local naming rows.*

7. *P is the* Pkg *provenance over* $\mathsf{ProductUniformSpace}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* ***??****,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $A, B, \Pi, E_A, E_B, W, H, C, P, N$. It has no coordinate for host product topology, unrestricted choice of entourages, quotient completion, selected limits, open-cover compactness, or bridge checking.

Lean4 ✓ `BEDC.Derived.ProductUniformSpaceUp.ProductUniformSpaceTasteGate single carrier alignment`

Theorem 3060.2 (Product uniform space NameCert obligations)**.** *For every accepted* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier $U = (A, B, \Pi, E_A, E_B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{UniformSpace}^{\uparrow}$ *source packets, the ?? projection row, the paired left and right entourage ledgers, the finite componentwise witness row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. No product-uniformity obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4040.1. The tuple displays exactly the two source packets, projection row, paired entourage ledgers, witness row, transport, replay, provenance, and local naming.

The source rows provide the only uniform data available to the packet. A product-entourage read must therefore pass through A and B and then through the named ledgers E_A and E_B.

The projection row Π and witness row W bind the product read to the two component reads. Transport by H is componentwise, and replay by C follows the same displayed route.

Since P and N only source and name this finite surface, and since no other carrier coordinate remains, the listed rows exhaust the local NameCert obligations. □

Theorem 3060.3 (Product uniform space projection-entourage route)**.** *Every completion-facing or metric-compatible read of an accepted* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier factors through the finite route*

$$??\longmapsto(\mathsf{UniformSpace}^{\uparrow},\mathsf{UniformSpace}^{\uparrow})\longmapsto\mathsf{Completion}^{\uparrow}.$$

More explicitly, a consumer first reads the projection row Π, then the left and right uniform source packets A, B, then the paired entourage ledgers E_A, E_B, and only afterward the witness, transport, replay, provenance, and local naming rows W, H, C, P, N. The route exports no host product topology, unrestricted entourage selector, quotient completion, selected limit, or ambient bridge proof.

Proof. Start with the accepted carrier and the obligations of Definition 4040.2. The product request is not available before Π displays the two projections and paired point-name admission.

Each projected read then enters one source uniform packet. The left and right entourage data are the displayed ledgers E_A and E_B, so the product entourage is a finite pair of component ledgers rather than a host-level neighbourhood of a product topology.

The completion-facing part reads only the same finite route. The witness row W records componentwise acceptance, while H, C, P, N transport, replay, source, and name that route for $\mathsf{Met}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers.

Because the carrier has no coordinate for host product topology, unrestricted entourage choice, quotient completion, selected limits, or bridge checking, no consumer can project those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ProductUniformSpace}^{\uparrow}$ packet is generated from two UniformSpace source packets, a Product projection row, paired component entourage ledgers, a finite componentwise witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ProductUniformSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4040.1, Definition 4040.2, and Definition 4040.3 seed the finite product-uniformity route used by metric-compatible completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProductUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host product topology, unrestricted choice of entourages, quotient completion, selected limits, Lean verification, or bridge checking for ProductUniformSpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked source carrier admission, projection coverage, paired entourage exactness, componentwise transport, replay stability, provenance, local naming, and completion-route non-escape for BEDC.Derived.ProductUniformSpaceUp.

Chapter 3061

A Concrete Naming Certificate for $\mathsf{ConstructiveRealLine}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveRealLine}^{\uparrow}$ packet records the textbook-facing L10 real-line interface as finite BEDC NameCert data rather than as a primitive host continuum. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, the located real surface of **??**, and the apartness boundary of chapter 1380. The chapter names the interface on which later real-analysis packets can depend without claiming real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Definition 3061.1 (Constructive real line carrier)**.** *A* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, E, L, A, \Delta, I, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing requested dyadic precision, comparison slack, and density margins.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which every requested observation of a real name is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected stream windows and dyadic tolerance rows into rational approximants.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row exported only after the D, S, R route has been displayed.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *boundary row recording finite interval-location evidence for the same readback surface.*
6. *A is the* $\mathsf{LocatedApartness}^{\uparrow}$ *boundary row recording positive separatedness evidence when a consumer asks for apartness.*
7. *Δ is the dyadic-density handoff row from rational or dyadic samples to the inspected real-name windows.*
8. *I is the interval readback row naming rational intervals whose endpoints are read through D, S, R and whose locatedness is checked through L and A.*
9. *H records componentwise* hsame *transport for $D, S, R, E, L, A, \Delta, I$ and for the structural certificate rows.*

10. *C records displayed* Cont *replay from a real-line request to the finite window, tolerance, readback, seal, locatedness, apartness, density, and interval rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveRealLine}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $D, S, R, E, L, A, \Delta, I, H, C, P, N$. It has no coordinate for real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Lean4 ✓ `BEDC.Derived.ConstructiveRealLineUp.ConstructiveRealLineTasteGate single carrier alignment`

Theorem 3061.2 (Constructive real line NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier*

$$Q = (D, S, R, E, L, A, \Delta, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance ledger, stream windows, regular rational readback, terminal real seal, locatedness boundary, apartness boundary, density handoff, interval readback, transport, replay, provenance, and local-name rows;*
2. *window-readback compatibility, requiring each S window consumed by a real-line request to be read by R under the tolerance ledger D;*
3. *seal discipline, requiring every* $\mathsf{Real}^{\uparrow}$ *export through E to pass first through the displayed D, S, R route;*
4. *locatedness discipline, requiring every located interval read to pass through L and I before a real-line consumer can inspect it;*
5. *apartness discipline, requiring every positive separatedness read to pass through A and the interval readback row I;*
6. *dyadic-density handoff, requiring each rational or dyadic sample exposed by Δ to be tied to the same D, S, R windows;*
7. *componentwise* hsame *transport for $D, S, R, E, L, A, \Delta, I, C, P, N$ through H;*
8. *displayed* Cont *replay of each real-line request through the finite route named by C;*
9. Pkg *provenance for exactly the displayed packet;*
10. *ledger non-escape, requiring every consumer to remain inside $D, S, R, E, L, A, \Delta, I, H, C, P, N$.*

Consequently the packet exports a textbook-facing constructive real-line interface, not real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Proof. Project the carrier of Definition 4041.1. The projection exposes exactly the dyadic tolerance, finite window, regular readback, real seal, locatedness, apartness, density, interval, transport, replay, provenance, and local naming rows.

The readback obligations are forced by the displayed route D, S, R. A consumer first names the finite stream windows, then reads the corresponding regular rational data, and only then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The located and apartness obligations are separate boundary reads. The interval row I supplies the rational interval readback, L supplies locatedness, and A supplies positive separatedness when the request needs it.

The density, transport, replay, provenance, and naming rows do not add a host continuum coordinate. Since every listed consumer remains inside $D, S, R, E, L, A, \Delta, I, H, C, P, N$, the packet cannot export real completeness, trichotomy, quotient streams, selected limits, or host real equality. □

Theorem 3061.3 (Constructive real line L10 seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier factors through the L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with located and apartness observations read only through the sibling $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{LocatedApartness}^{\uparrow}$ *boundary rows. No accepted real-line consumer can bypass this route through quotient streams, a selected limit, host real equality, trichotomy, or ambient real completeness.*

Proof. Begin with the carrier route in Definition 4041.1. Its only terminal real-name coordinate is E, and E is listed after the dyadic tolerance ledger D, the stream window schedule S, and the regular rational readback R.

The located and apartness reads remain boundary reads. A located request must use L and I, while a positive separatedness request must use A and the same interval readback row.

Finally replay the request through C and transport it through H. The provenance row P and local certificate row N name the same displayed packet, so the route has no hidden coordinate from which quotient stream equality, a selected host limit, trichotomy, host real equality, or real completeness could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveRealLine}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a terminal RealUp seal, located-real and located-apartness boundaries, dyadic-density handoff, interval readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4041.1, Definition 4041.2, and Definition 4041.3 seed the textbook-facing constructive real-line interface over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealUp, LocatedApartnessUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, window-readback compatibility, seal discipline, locatedness and apartness boundaries, dyadic-density handoff, interval readback, transport, replay, provenance, and ledger non-escape.

$\mathsf{MontelSpace}^{\uparrow}$ reads the locally convex and Frechet-style bounded-set compactness route through chapter 3495, chapter 1735, and chapter 3693 before the child file expands the finite carrier.

Chapter 3062

A Concrete Naming Certificate for $\mathsf{MontelSpace}^{\uparrow}$

Following Bourbaki and Schaefer, the $\mathsf{MontelSpace}^{\uparrow}$ packet records the traditional locally convex and Frechet-style bounded-set compactness surface as finite BEDC NameCert data. It is the carrier companion to the normal-family packet of chapter 3693: the Montel theorem route may consume a displayed Montel-space source when local boundedness and compact-window extraction are read from a locally convex carrier rather than from a holomorphic family alone. The packet also sits over the complete separable metric boundary of chapter 1735, the complete-metric and Polish readback surfaces, and the finite bornological rows used by bounded-set consumers. It names a finite bornological compactness route, not host topological vector space equality, barrelledness, nuclearity, selected compact closures, quotient completions, or classical choice.

Definition 3062.1 (Montel space carrier)**.** *A* $\mathsf{MontelSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, F, B, Q, K, R, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the locally convex source row, exposing the finite seminorm or entourage windows through which bounded sets are inspected.*
2. *F is the Frechet-style completeness and metrizability row, read through complete metric, complete separable metric, or Polish-space surfaces when those rows are displayed.*
3. *B is the bounded-set ledger, recording the finite family of bounded-set requests that the Montel-space consumer may inspect.*
4. *Q is the bornological compactness row: each displayed bounded request in B is routed to a compact, totally bounded, or sequentially compact readback already named by the surrounding metric packets.*
5. *K is the compact-window row consumed by normal-family, Arzela–Ascoli, and Montel theorem packets.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback for seminorm and metric tolerances.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by boundedness, compact-window, and completion reads.*

8. *H records componentwise* hsame *transport for the displayed locally convex, Frechet, bounded, compactness, tolerance, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from locally convex inspection through bounded-set admission, compact-window readback, and normal-family consumption.*
10. *P is the* Pkg *provenance over* $\mathsf{MontelSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, F, B, Q, K, R, D, H, C, P, N$. It has no coordinate for a host locally convex equality, a quotient completion, a selected compact closure, a barrelled or nuclear structure, countable choice, or open-cover compactness.

Theorem 3062.2 (Montel space NameCert obligations)**.** *For every accepted* $\mathsf{MontelSpace}^{\uparrow}$ *carrier $S = (L, F, B, Q, K, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$ *surface consists exactly of locally convex source admission, Frechet or complete-separable readback when displayed, bounded-set ledger admission, bornological compactness readback, compact-window export, regular-rational and dyadic tolerance discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4042.1. The displayed tuple exposes the locally convex source row, Frechet or complete-separable readback row, bounded-set ledger, bornological compactness row, compact-window row, tolerance rows, and the structural rows.

The bounded-set read is finite. Each request in B is inspected only through the locally convex windows in L and the tolerance rows R, D, and the compactness answer is recorded in Q before any compact-window consumer sees K.

The replay row C then routes the same data toward normal-family and completion-facing consumers. Componentwise transport through H preserves the displayed rows, while P, N only source and name the packet.

Since the carrier has no coordinate for host locally convex equality, quotient completion, selected compact closures, countable choice, barrelledness, nuclearity, or open-cover compactness, the listed rows exhaust the certificate obligations. □

Theorem 3062.3 (Montel space bornological compactness route)**.** *Every bounded-set compactness consumer of an accepted* $\mathsf{MontelSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\textit{locally convex window} \longmapsto \textit{bounded-set ledger} \longmapsto \textit{bornological compactness readback} \longmapsto \textit{compact-window export.}$$

In particular, a $\mathsf{MontelTheorem}^{\uparrow}$ *or complete-separable metric consumer receives only the displayed rows $L, F, B, Q, K, R, D, H, C, P, N$ and receives no selected compact closure, quotient completion equality, host barrelledness theorem, nuclearity assertion, countable-choice schedule, or open-cover compactness proof.*

Proof. Start from the accepted carrier. The locally convex row L supplies the only inspection windows, and the Frechet or complete-separable row F is read only when the packet displays the corresponding metric boundary.

Next consume the bounded-set ledger B together with R, D. This gives the finite tolerance discipline for every bounded request and prevents a consumer from importing an ambient bornology or choice-selected family.

The compactness readback Q and compact-window row K are then replayed by C toward Montel theorem and completion-facing consumers. The route remains attached to the same rows under H, and P, N add only provenance and local naming.

Thus every bornological compactness handoff factors through the displayed packet and cannot export host locally convex equality, selected compact closures, quotient completion equality, barrelledness, nuclearity, countable choice, or open-cover compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4042.1, Definition 4042.2, and Definition 4042.3 seed the finite Montel-space carrier route through locally convex windows, Frechet or complete-separable readback, bounded-set ledgers, bornological compactness rows, compact-window export, tolerance ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex equality, barrelledness, nuclearity, selected compact closures, quotient completions, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-set ledger exactness, bornological compactness readback, compact-window export, tolerance discipline, transport stability, replay, provenance, and local naming for BEDC.Derived.MontelSpaceUp.

Constructive story: A $\mathsf{MontelSpace}^{\uparrow}$ packet is generated from locally convex source rows, Frechet or complete-separable metric readback, bounded-set ledgers, bornological compactness rows, compact-window export, RegSeqRat and DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3063

A Concrete Naming Certificate for $\mathsf{BaireGenericResidual}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireGenericResidual}^{\uparrow}$ packet records the constructive residual-point route through dense-open schedules rather than through a host countable intersection object. It sits over the Baire prefix of chapter 334, the dense-open certificate surface of chapter 1542, the complete-metric handoff of chapter 128, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter packages the residual thread consumed by later L10 complete-metric routes without adding countable choice, an external intersection point, quotient stream equality, or ambient real completeness.

Definition 3063.1 (Baire generic residual carrier)**.** *A* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, D, M, S, W, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row naming the inspected basic-open source and its finite prefix probes.*
2. *D is the dense-open schedule row inherited from the* $\mathsf{BaireCategory}^{\uparrow}$ *packet, including the displayed finite list of dense requests currently consumed by the residual route.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row through which the nested ball thread is interpreted.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule selecting the finite prefix, dense-open, and completion observations visible at this stage.*
5. *W is the stream-window ledger recording the inspected predecessor balls, successor balls, radius budgets, and finite membership checks.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius comparisons, and tolerance data used by the thread.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the B, D, M, S, W, R route has been displayed.*
8. *L is the residual ledger, listing the dense-open obligations already met by the constructed thread and the finite evidence that each listed obligation is met inside the chosen successor ball.*

9. *H records componentwise* hsame *transport for the Baire prefix, dense-open schedule, complete-metric handoff, stream windows, regular readback, real seal, residual ledger, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a residual request to the finite Baire prefix, dense-open schedule, completion handoff, stream windows, rational readback, real seal, and residual ledger rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BaireGenericResidual}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $B, D, M, S, W, R, E, L, H, C, P, N$. It has no coordinate for countable choice, a host intersection point, quotient equality of streams, or completeness of an ambient real line.

Lean4 ✓ `BEDC.Derived.BaireGenericResidualUp.BaireGenericResidualTasteGate single carrier alignment`

Theorem 3063.2 (Baire generic residual NameCert obligations)**.** *For every accepted* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier $G = (B, D, M, S, W, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix, dense-open schedule, complete-metric handoff, StreamName windows, window ledger, RegSeqRat readback, Real seal, residual ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted residual read must remain inside these rows and may not inspect countable-choice selectors, host intersection points, quotient stream equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 4043.1. The analytic rows are B, D, M, S, W, R, E, L, and the structural rows are H, C, P, N.

The dense-open work is confined to D, W, L. The schedule row names the finite dense requests, the window ledger records the predecessor and successor ball observations, and the residual ledger records that each displayed request has been met by the current thread.

The completion work is confined to M, S, R, E. The complete-metric row supplies the handoff for the nested thread, the stream schedule selects the inspected windows, the regular-rational row reads the centers and radii, and the real seal is downstream from that route.

Transport, replay, provenance, and local naming preserve the same displayed packet. Since no listed row stores a countable-choice selector, a host intersection point, quotient equality of streams, or ambient Real completeness, the displayed obligations exhaust the local NameCert surface. □

Theorem 3063.3 (Baire generic residual thread handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier factors through the constructive residual thread*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff records residual membership only through the finite residual ledger L and exports no host intersection point, countable-choice enumeration, quotient stream equality, or separate ambient completeness theorem.

Proof. Begin with the carrier route in Definition 4043.1. The terminal real row E is downstream from the Baire prefix B, the dense-open schedule D, the complete-metric handoff M, the inspected stream windows S, W, and the regular rational readback R.

The residual claim is ledgered, not selected. Each displayed dense-open obligation is recorded in L with the finite successor-ball evidence currently consumed by the route, so the proof never chooses a point outside the displayed thread.

Finally replay the route through C and transport it through H. The provenance row P and local naming row N name the same finite packet, so the exported $\mathsf{Real}^{\uparrow}$ seal has no hidden coordinate from which countable choice, a host intersection point, quotient stream equality, or ambient completeness can be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireGenericResidual}^{\uparrow}$ packet is generated from a BaireSpace prefix, a BaireCategory dense-open schedule, a CompleteMetric handoff, StreamName windows, RegSeqRat readback, a Real seal, a residual ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4043.1, Definition 4043.2, and Definition 4043.3 seed the constructive residual-point route over BaireSpaceUp, BaireCategoryUp, CompleteMetricUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, host intersection points, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exactness, complete-metric handoff, StreamName window readback, RegSeqRat readback, Real seal discipline, residual-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.BaireGenericResidualUp.

Chapter 3064

A Concrete Naming Certificate for $\mathsf{CountableMetricCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CountableMetricCompletion}^{\uparrow}$ packet names the L10 completion route by which a countably displayed metric source exposes completion names through stream schedules, regular rational readbacks, dyadic tolerance ledgers, and terminal real seals. It refines the finite-window completion surfaces of chapter 4023 and chapter 1683, and it routes its observable data through chapter 126, chapter 301, chapter 291, chapter 38, and chapter 1612. The packet records only the displayed countable source, enumerated metric windows, regular sequence route, separated comparison, transport, replay, provenance, and local NameCert rows.

Definition 3064.1 (Countable metric completion carrier)**.** *A* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, A, S, Q, D, Z, R, H, K, P, N)$$

with the following displayed rows:

1. *M is the metric source row, naming the source point reads, distance comparison windows, and metric ledger offered to the completion request.*
2. *A is the countable admission row, exposing the finite portion of the source enumeration that the request actually inspects.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the selected finite prefixes of candidate completion names.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested radii, slack rows, and comparison budgets.*
6. *Z is the separated comparison row inherited from the separated-completion route after the same M, A, S, Q, D rows have been displayed.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for metric distances and completion-facing comparison readbacks.*
8. *H records componentwise* hsame *transport for the metric source, countable admission, stream schedule, regular readback, dyadic ledger, separated comparison, Real seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a completion request through $M, A, S, Q, D, Z, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CountableMetricCompletion}^{\uparrow}$, $\mathsf{BishopMetricCompletion}^{\uparrow}$, $\mathsf{SequentialCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, S, Q, D, Z, R, H, K, P, N$. It has no coordinate for quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CountableMetricCompletionUp.CountableMetricCompletionTasteGate single carrier alignment`

Theorem 3064.2 (Countable metric completion NameCert obligations)**.** *For an accepted* CountableMetricCompleti *packet $C = (M, A, S, Q, D, Z, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$ *surface consists of carrier admission, countable source exposure, finite stream scheduling, regular rational readback, dyadic tolerance checking, separated comparison, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and ledger non-escape. Every obligation is discharged through the displayed route*

$$M \longmapsto A \longmapsto S \longmapsto Q \longmapsto D \longmapsto Z \longmapsto R$$

together with H, K, P, N.

Proof. Project the carrier of Definition 4044.1. The metric source row M and countable admission row A must be visible before any completion schedule is inspected.

The schedule row S exposes only finite stream prefixes. The regular readback row Q and dyadic ledger D then provide the rational observations and tolerance budgets used by the separated comparison row Z.

The terminal real seal R is reached only after that finite route. Componentwise transport through H and replay through K preserve the same rows, while P and N source and name the packet.

Since quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, and ambient completeness are absent from the carrier, the displayed rows exhaust the local NameCert obligations. □

Theorem 3064.3 (Countable metric completion Cauchy sequence route)**.** *Every Cauchy-sequence read of an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet factors through the displayed source metric row, countable admission row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, separated comparison row, transport, replay, provenance, and local NameCert rows. The route exports finite Cauchy evidence and separated comparison evidence; it does not export a quotient class, a selected representative, a countable-choice selector, or a host powerset of all sequences.*

Proof. Use Definition 4044.2. It places every accepted completion read on the route $M, A, S, Q, D, Z, R, H, K, P, N$.

A Cauchy-facing consumer first receives only the finite source enumeration visible in A. It then reads stream prefixes through S, rational regular-sequence observations through Q, and tolerance budgets through D.

The separated comparison row Z is downstream of those rows, so equality-facing or distance-facing reads cannot bypass the displayed Cauchy evidence. The structural rows H, K, P, N preserve and name that route.

No carrier row contains a quotient class, representative selector, choice function, or host powerset, so the Cauchy-sequence consumer boundary is exactly the finite displayed route. □

Theorem 3064.4 (Countable metric completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing handoff from an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet reaches the terminal real seal only after the metric source, countable admission, stream schedule, regular rational readback, dyadic tolerance ledger, and separated comparison rows have been displayed. Thus the handoff can reuse the Real row of chapter 38 for named distance readbacks without asserting ambient Real completeness, selected limits, quotient completion equality, trichotomy, or countable choice.*

Proof. Start from the carrier route in Definition 4044.1. The row R is not an independent source of completion data; it is downstream of M, A, S, Q, D, Z.

Replay through K therefore reaches the terminal $\mathsf{Real}^{\uparrow}$ seal only after the stream schedule, regular rational readback, dyadic tolerance ledger, and separated comparison rows have been inspected. Componentwise hsame transport through H keeps the same dependence under name change.

The Real-facing consumer receives a seal for the displayed readback, not a completeness principle. Since no excluded object is a coordinate of C, the handoff cannot introduce selected limits, quotient equality, trichotomy, countable choice, or ambient Real completeness. □

Theorem 3064.5 (Countable metric completion ledger non-escape)**.** *The provenance and local naming rows of an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet prevent downstream completion consumers from reading outside the displayed L10 ledger. Any consumer of the packet receives only* $M, A, S, Q, D, Z, R, H, K, P, N$ *and the cited Bishop metric completion, sequential completion, StreamName, RegSeqRat, DyadicRatCore, separated-completion, and Real rows; no host completion, quotient universe, selected limit operation, countable-choice principle, or ambient Real-completeness theorem is exported.*

Proof. Apply Definition 4044.2 to expose the local NameCert surface. The provenance row P cites the completion and numerical chapters only through their displayed packet rows, and the local name N records the same finite carrier.

Replay through K cannot add a new coordinate. It can only revisit the source, countable admission, stream schedule, regular readback, dyadic ledger, separated comparison, Real seal, and transport rows already present in C.

Consequently every downstream consumer is confined to the L10 ledger named by the packet. Host completions, quotient universes, selected limits, countable choice, and ambient Real completeness are not exported because no row of the carrier stores them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CountableMetricCompletion}^{\uparrow}$ packet is generated from a metric source row, countable admission row, finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, separated-completion comparison row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4044.1, Definition 4044.2, Definition 4044.3, Definition 4044.4, and Definition 4044.5 seed the countable metric-completion route through displayed StreamName, RegSeqRat, DyadicRatCore, separated-completion, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, countable source exposure, regular readback, separated comparison, Real sealing, replay, provenance, and local NameCert non-escape for BEDC.Derived.CountableMetricCompletionUp.

Chapter 3065

Cauchy Sequential Continuity NameCert Construction

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchySequentialContinuity}^{\uparrow}$ packet records the finite certificate that a displayed continuous map carries a regular Cauchy name to a regular Cauchy image name. It sits between the certified sequence-consumer surface of chapter 1466, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the graph-and-modulus rows of chapter 130. The packet gives the L10 Real and Completion route a finite preservation certificate before a completion consumer treats the image stream as a real name.

Definition 3065.1 (Cauchy sequential continuity carrier)**.** *A* $\mathsf{CauchySequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E_Y, B, H, C, P, N)$$

with the following displayed rows:

1. M_X *and* M_Y *are the source and target metric rows used by the displayed map.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row for the source-to-target map.*
3. S *is the source Cauchy stream surface, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. W_X *and* W_Y *are the inspected source windows and image windows.*
5. R_X *and* R_Y *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the source and image windows.*
6. E_X *and* E_Y *are the* $\mathsf{Real}^{\uparrow}$ *seal rows available after the corresponding regular readbacks.*
7. B *is the Cauchy-preservation ledger, listing the finite pairs of source thresholds, target tolerances, and modulus rows needed to turn the* R_X *regularity witness into the* R_Y *regularity witness.*
8. H *records componentwise* hsame *transport for all displayed rows and for continuation, provenance, and local naming.*
9. C *records displayed* Cont *replay from the source windows through the continuous-map modulus row to the image windows, then through regular readback and seal handoff.*
10. P *is the* Pkg *provenance over* $\mathsf{CauchySequentialContinuity}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host function, quotient stream equality, selected limit, countable choice, external completeness theorem, or ambient real-continuity principle.

Lean4 ✓ `BEDC.Derived.CauchySequentialContinuityUp.CauchySequentialContinuityTasteGate single carrier alignment`

Theorem 3065.2 (Cauchy sequential continuity Cauchy preservation)**.** *Every accepted* CauchySequentialContinuity *carrier carries its displayed source regular Cauchy name to a displayed target regular Cauchy name. More precisely, any consumer of the target* $\mathsf{Real}^{\uparrow}$ *seal must factor through*

$$S \longmapsto W_X \longmapsto R_X \longmapsto F \longmapsto W_Y \longmapsto B \longmapsto R_Y \longmapsto E_Y$$

together with the metric rows, transport rows, continuation replay, provenance, and local naming.

Proof. Project the carrier of Definition 4045.1. The source stream S is visible only through the inspected windows W_X, and the source Cauchy evidence is visible only through the regular rational readback R_X.

Apply the displayed continuous-map row F to the same finite source windows. The continuation row C records that every image window W_Y is produced from W_X through the graph-and-modulus data of F, not by a host function equality or an unrecorded semantic map.

The preservation ledger B then binds the source thresholds from R_X to the target tolerances requested for R_Y. Componentwise hsame transport in H keeps the metric rows, window rows, regular readbacks, seal rows, continuation, provenance, and local naming on the same displayed packet.

Finally the target seal E_Y is downstream from R_Y. Thus a target-real consumer that bypasses the source readback, the continuous-map modulus row, the preservation ledger, or the target readback leaves the accepted carrier. □

Theorem 3065.3 (Cauchy sequential continuity NameCert obligations)**.** *For every accepted* CauchySequentialConti *carrier* $K = (M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E_Y, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission, source and target metric exposure, continuous-map modulus exposure, source stream-window exposure, source and target regular-rational readback, source and target real-seal handoff, Cauchy-preservation ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Read the displayed tuple in Definition 4045.1. The analytic rows are $M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E$ the structural rows are H, C, P, N.

The source side is exposed by S, W_X, R_X, E_X. These rows say only that the inspected stream windows have regular rational readback and a downstream real seal.

The map side is exposed by M_X, M_Y, F, W_Y, B. The continuous-map row supplies the finite modulus data used to form image windows, and the preservation ledger records the threshold comparisons that certify those image windows as regular Cauchy.

The remaining rows transport and replay the same finite route. Since no excluded host principle is a coordinate of the carrier, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequentialContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a ContinuousMap graph-and-modulus row, finite source and image StreamName windows, source and target RegSeqRat

readbacks, RealUp seal rows, a Cauchy-preservation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchySequentialContinuity}\uparrow}$, seed)

Scope closed: Definition 4045.1, Definition 4045.2, and Definition 4045.3 seed the finite certificate that displayed continuous maps preserve regular Cauchy names.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchySequentialContinuity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sequential continuity, quotient stream equality, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target metric exposure, continuous-map modulus exposure, stream-window exactness, RegSeqRat readback, RealUp seal handoff, Cauchy-preservation ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequentialContinuityUp.

Chapter 3066

A Concrete Naming Certificate for $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ packet records the completion route for bounded continuous real-valued functions under the displayed sup-norm surface. It reads the finite sup-norm carrier of chapter 3807, the uniform-limit metric handoff of chapter 1764, the real-valued uniform Cauchy route of chapter 1830, the compact metric source of chapter 129, the continuous-map row of chapter 130, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the completion-facing boundary as finite BEDC data: Cauchy rows, uniform windows, compact-source continuity, Real-sealed sup-norm estimates, and local NameCert replay are visible, while host function equality, quotient function spaces, selected limits, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the carrier.

Definition 3066.1 (Continuous sup-norm completion carrier)**.** *A* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, U, A, K, F, R, W, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *row, exposing the compact-source sup-norm ledger through the sibling carrier.*
2. *U is the* $\mathsf{UniformLimitMetric}^{\uparrow}$ *row that reads sup-norm Cauchy control as a uniform metric-limit request.*
3. *A is the* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *row recording the displayed uniform Cauchy schedule for the function rows under inspection.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row shared by the sup-norm and Cauchy schedules.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the inspected real-valued function data over K.*
6. *R is the pointwise* $\mathsf{Real}^{\uparrow}$ *difference-and-bound seal reached after the compact source, continuous-map, and sup-norm rows have been displayed.*
7. *W is the finite uniform-window ledger, recording the tolerance request, source windows, and paired function-row comparisons used by the Cauchy route.*
8. *L is the completion-facing limit service row. It stores only the displayed eligibility of the Cauchy route for a uniform-limit consumer, not a selected host limit.*

9. *H records componentwise* hsame *transport for $S, U, A, K, F, R, W, L, C, P, N$.*
10. *C records the displayed* Cont *replay route from a completion request through the rows $S, U, A, K, F, R, W, L, H, P,$*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{UniformLimitMetric}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed sup-norm source, uniform-limit metric route, uniform Cauchy schedule, compact source, continuous-map row, Real seal, finite window ledger, limit-service eligibility row, structural transport, replay, provenance, and local naming. It has no coordinate for host function equality, quotient function-space equality, a choice-selected limiting function, open-cover compactness, or a theorem asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ContinuousSupNormCompletionUp.ContinuousSupNormCompletionTasteGate single carrier alignment`

Theorem 3066.2 (Continuous sup-norm completion NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier $Q = (S, U, A, K, F, R, W, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}}$ *surface consists exactly of carrier admission,* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *source exposure through S,* $\mathsf{UniformLimitMetric}^{\uparrow}$ *handoff through U,* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *schedule exposure through A, compact metric source exposure through K, continuous-map row exposure through F, pointwise* $\mathsf{Real}^{\uparrow}$ *difference-and-bound sealing through R, finite uniform-window exposure through W, completion-facing limit-service eligibility through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4046.1. The tuple displays the sup-norm source, uniform-limit metric row, real-function Cauchy schedule, compact metric source, continuous-map row, Real seal, finite window ledger, limit-service row, transport, replay, provenance, and local naming.

The source and Cauchy rows determine the only function rows that may be inspected. The compact metric and continuous-map rows determine the common domain of those inspections, while the Real seal and finite window ledger supply the displayed sup-norm estimates.

The limit-service row records that this displayed route may be consumed by a uniform-limit metric surface. Since H, C, P, N only transport, replay, source, and name the same rows, the listed coordinates exhaust the local NameCert obligations. □

Theorem 3066.3 (Continuous sup-norm completion Cauchy route)**.** *Every accepted* ContinuousSupNormCompletio *carrier $Q = (S, U, A, K, F, R, W, L, H, C, P, N)$ routes a bounded continuous-function Cauchy request through the finite chain*

$$\mathsf{ContinuousSupNorm}^{\uparrow} \longmapsto \mathsf{RealFunctionUniformCauchy}^{\uparrow} \longmapsto \mathsf{UniformLimitMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, every tolerance-controlled comparison first reads the compact source K and continuous-map row F, then the sup-norm source S, the uniform Cauchy schedule A, the finite window ledger W, the Real difference-and-bound seal R, and finally the uniform-limit metric row U and limit-service row L.

Proof. Start with the accepted carrier and the obligations of Definition 4046.2. The compact source K and continuous-map row F fix the inspected domain and function rows before any sup-norm comparison can be read.

The source row S supplies the finite sup-norm comparison route from chapter 3807. The uniform Cauchy schedule A and window ledger W then place the same comparisons under a displayed tolerance request.

The Real seal R is reached only after those rows have been displayed. The uniform-limit metric row U and the limit-service row L therefore consume the same finite route rather than a host completion object, selected limiting function, quotient function space, or ambient completeness theorem. □

Theorem 3066.4 (Continuous sup-norm completion uniform-limit seal)**.** *Every uniform-limit read exported by an accepted* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier factors through the displayed rows* $S, U, A, K, F, R, W, L, H, C, P, N$*. Consequently a downstream compact function-space or Banach-space route may consume the carrier only as a finite sup-norm Cauchy boundary over* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$*,* $\mathsf{UniformLimitMetric}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *data; it receives no host function equality, quotient function space, selected limit, open-cover compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 4046.3. It places every completion-facing request behind the compact source, continuous-map row, sup-norm comparison row, uniform Cauchy schedule, finite window ledger, and Real seal.

The rows U and L do not create another completion coordinate. They only expose the eligibility of the displayed Cauchy route for the uniform-limit metric consumer.

Finally, componentwise transport, replay, provenance, and local naming preserve the same row list. Since the carrier has no field for host function equality, quotient function-space equality, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no downstream route can extract those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ packet is generated from a ContinuousSupNorm source row, a UniformLimitMetric handoff row, a RealFunctionUniformCauchy schedule, a CompactMetric source row, a ContinuousMap row, a RealUp difference-and-bound seal, a finite uniform-window ledger, a completion-facing limit-service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4046.1, Definition 4046.2, Definition 4046.3, and Definition 4046.4 seed the completion-facing sup-norm Cauchy boundary for bounded continuous real-valued functions over a displayed compact metric source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, sup-norm source exposure, uniform Cauchy schedule exactness, compact-source and continuous-map stability, RealUp seal non-escape, finite window coverage, uniform-limit handoff, replay stability, and ledger non-escape for BEDC.Derived.ContinuousSupNormCompletionUp.

Chapter 3067

A Concrete Naming Certificate for $\mathsf{CofinalSequenceSelector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and Kelley on directed tails, the $\mathsf{CofinalSequenceSelector}^{\uparrow}$ packet records the finite monotone index-window selector used before a completion-facing consumer reads a cofinal subsequence, a Cauchy-filter tail, or a terminal real seal. It sits after the cofinal sequence surface of chapter 1594, the cofinal tail map of chapter 2141, the Cauchy-filter packet of chapter 333, the regular Cauchy subsequence surface of chapter 381, and the terminal real-name seal of chapter 38. The packet names a finite selector schedule; it does not choose an infinite subsequence by countable choice, quotient streams, ambient directed-set cofinality, host real equality, or an external completeness principle.

Definition 3067.1 (Cofinal sequence selector carrier)**.** *A* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, W, M, Q, F, U, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Nat}^{\uparrow}$ *index row, displaying the finite source interval and the terminal lower bound requested by the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing every inspected source index and selector output.*
3. *M is the monotonicity ledger, recording that the displayed selector outputs are nondecreasing on the finite window.*
4. *Q is the cofinal-window ledger, recording that every displayed lower-bound request is met inside the same finite selector window.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-filter row reached from I, W, M, Q.*
6. *U is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row, recording the selected regular-Cauchy subsequence readback.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row available only after the window, filter, and subsequence handoffs have been displayed.*
8. *H records componentwise* hsame *transport for I, W, M, Q, F, U, R and for the continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite lower-bound request through the selector window, monotonicity ledger, cofinal ledger, Cauchy-filter tail, regular-Cauchy subsequence handoff, and terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CofinalSequenceSelector}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, W, M, Q, F, U, R, H, C, P, N$. It has no coordinate for a choice-selected infinite subsequence, quotient stream equality, host order on all natural-number functions, ambient filter convergence, global completeness, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.CofinalSequenceSelectorUp.CofinalSequenceSelectorUp`
Lean4 ✓ `BEDC.Derived.CofinalSequenceSelectorUp.CofinalSequenceSelectorTasteGate single carrier alignment`

Theorem 3067.2 (Cofinal sequence selector NameCert obligations)**.** *For every accepted* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier $S = (I, W, M, Q, F, U, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$ *surface consists exactly of finite* $\mathsf{Nat}^{\uparrow}$ *index admission,* $\mathsf{StreamName}^{\uparrow}$ *window exposure, finite monotonicity exposure, finite cofinal-window exposure,* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-filter handoff,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *selected-subsequence handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4047.1. The rows I, W, M, Q expose the finite selector schedule: the inspected index window, the output window, the monotonicity ledger, and the cofinal lower-bound ledger.

The analytic handoff is read through F, U, R. The filter row reads the displayed tail request, the subsequence row reads the same selected window as regular-Cauchy data, and the terminal row seals the resulting regular rational and real-facing output.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for countable choice, quotient stream equality, ambient filter convergence, host real equality, or global completeness, the listed rows exhaust the local certificate obligations. □

Theorem 3067.3 (Cofinal sequence selector tail-filter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier reaches the terminal read through the finite route*

$$\mathsf{Nat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first displays the finite monotone cofinal index window, then reads the Cauchy-filter tail, then hands the selected regular-Cauchy subsequence to the regular rational and real-name seals. It exports no selected infinite subsequence, quotient stream equality, countable-choice schedule, host filter limit, or ambient completeness theorem.

Proof. Begin with the finite selector rows I, W, M, Q. They provide only a displayed natural-number window, the selected outputs in that window, and the finite proof rows that the displayed outputs are monotone and cofinal for the displayed lower-bound requests.

Next read F and U. The Cauchy-filter row consumes the same tail window, while the regular-Cauchy subsequence row consumes the selected outputs as the finite subsequence surface already used by chapter 381.

Finally pass through R, H, C, P, N. The regular rational and real seal rows are downstream of the filter and subsequence handoffs, and the transport, replay, provenance, and naming rows keep the route inside the accepted packet. Thus a completion-facing consumer receives the finite selector handoff and no additional choice, quotient, host-limit, or ambient-completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{CofinalSequenceSelector}^{\uparrow}$ consumer must expose finite $\mathsf{Nat}^{\uparrow}$ index rows, $\mathsf{StreamName}^{\uparrow}$ selector windows, monotonicity and cofinal-window ledgers, a $\mathsf{CauchyFilter}^{\uparrow}$ tail row, a $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff row, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before the selected tail is available.

Independence witness: Nearest siblings: cofinal sequence, cofinal tail map, CauchyFilter, RegularCauchySubsequence, RegSeqRat, StreamName, Nat, and Real. The packet is not identical to any sibling because it jointly binds finite monotone selector windows, finite cofinal lower-bound ledgers, tail-filter handoff, regular-Cauchy subsequence handoff, and the terminal regular-rational and real seals.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalSequenceSelector}^{\uparrow}$ packet is generated from finite NatUp index rows, StreamNameUp selector windows, monotonicity ledgers, cofinal-window ledgers, CauchyFilterUp tail handoffs, RegularCauchySubsequenceUp handoffs, RegSeqRatUp and RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4047.1, Definition 4047.2, and Definition 4047.3 seed the finite monotone cofinal selector surface for completion-facing tail consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, ambient filter convergence, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite index admission, selector-window exposure, monotonicity and cofinal-window ledgers, CauchyFilterUp tail handoff, RegularCauchySubsequenceUp handoff, RegSeqRatUp and RealUp seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CofinalSequenceSelectorUp.

Chapter 3068

A Concrete Naming Certificate for $\mathsf{SupportNerveCensus}^{\uparrow}$

Following the finite theory of abstract simplicial complexes, nerves, and Hochster-style Betti bookkeeping, the $\mathsf{SupportNerveCensus}^{\uparrow}$ packet records the support-family census of Definition 9462.1655 as finite NameCert data. It sits beside the simplicial-complex carrier of chapter 238, the finite simplicial-chain carrier of chapter 3414, the Hochster nerve–Betti reduction of chapter 3744, and the Hilbert–Alexander blocker packet of chapter 3932. The packet names the finite table of support nerves, reduced Betti signatures, top-support criterion, and lifted-Betti handoff; it does not import an unrestricted resolution, a host homology computation, coefficient-field choice, or ambient equality of Betti tables.

Definition 3068.1 (Support nerve census carrier)**.** *A* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, A, N, B, T, L, H, R, P, M)$$

with the following displayed rows:

1. *S is the support-type row, displaying the fifteen nonempty support symbols and the four singleton support entries used by the top-support criterion.*
2. *A is the finite support-family row, recording the inspected family $A \subseteq \mathcal{S}$ and its cardinality bucket.*
3. *N is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row, listing the faces of the support nerve N_A on four vertices.*
4. *B is the reduced Betti-signature row, recording the displayed tuple $(\widetilde{\beta}_{-1}, \widetilde{\beta}_0, \widetilde{\beta}_1, \widetilde{\beta}_2)$.*
5. *T is the top-support row, recording whether the four singleton supports are present and therefore whether the top homology case is exposed.*
6. *L is the lifted-Betti row, reading the finite support-level table through the codon lift substitution of Definition 9462.1658.*
7. *H records componentwise* hsame *transport for S, A, N, B, T, L and for the replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from support-family inspection through nerve formation, finite homology readback, Betti-signature table lookup, top-support checking, and lifted-Betti handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{SupportNerveCensus}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteSimplicialChain}^{\uparrow}$, $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$, $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. M is the local $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, N, B, T, L, H, R, P, M$. It has no coordinate for an unrestricted minimal resolution, host chain-complex equality, coefficient-field selection, unbounded simplicial homology, ambient Betti-table equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.SupportNerveCensusUp.SupportNerveCensusTasteGate single carrier alignment`

Theorem 3068.2 (Support nerve census NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier $C = (S, A, N, B, T, L, H, R, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}^{\uparrow}}$ *surface consists exactly of support-type exposure, finite support-family exposure,* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve exposure, finite reduced-Betti signature readback, top-support criterion exposure, lifted-Betti handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.SupportNerveCensusUp.SupportNerveCensusNameCertObligations`

Proof. Project the carrier of Definition 4048.1. The rows S, A, N display the finite support universe, the inspected support family, and the resulting four-vertex support nerve.

The census data are exactly B, T, L. The Betti row records the reduced signature for the displayed nerve, the top-support row records the singleton-support criterion of Definition 9462.1657, and the lifted row records the finite handoff used by Definition 9462.1658.

The structural rows H, R, P, M transport, replay, source, and name the same finite table. Since the packet contains no coordinate for unrestricted resolutions, host chain-complex equality, coefficient-field selection, unbounded homology, or ambient Betti-table equality, the listed rows exhaust the local certificate obligations. □

Theorem 3068.3 (Support nerve census lifted Betti handoff)**.** *Every lifted-Betti consumer of an accepted* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{SimplicialComplex}^{\uparrow} \longmapsto \mathsf{FiniteSimplicialChain}^{\uparrow} \longmapsto \mathsf{HochsterNerveBettiReduction}^{\uparrow} \longmapsto \mathsf{SupportNerveCensus}^{\uparrow}.$$

The route exposes the support nerve, finite chain readback, reduced Betti-signature table, top-support criterion, and codon-lift handoff before any lifted Betti entry is exported. It exports no unrestricted minimal resolution, host homology computation, coefficient-field choice, or ambient equality of Betti tables.

Proof. Start with the support-family and nerve rows S, A, N. These rows construct a finite four-vertex simplicial complex and therefore keep the computation inside the visible support nerve.

Next read B and T. The reduced Betti-signature row is a finite table lookup through the same finite chain surface used by the Hochster nerve–Betti reduction, and the top-support row records the singleton-support criterion rather than an ambient homology theorem.

Finally apply the lifted row L and the structural rows H, R, P, M. The continuation row fixes the order from support-family inspection to lifted-Betti export, and the provenance and name rows keep the result attached to the displayed packet. Thus the lifted table is exported only as finite NameCert data. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveCensus}^{\uparrow}$ consumer must expose support-type rows, finite support-family rows, a $\mathsf{SimplicialComplex}^{\uparrow}$ nerve row, a reduced Betti-signature row, a top-support criterion row, a lifted-Betti handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before a lifted Betti entry is available.

Independence witness: Nearest siblings: SimplicialComplex, FiniteSimplicialChain, HochsterNerveBettiReduction, HilbertAlexanderBlocker, and the support-nerve census theorem in the vision section. The packet is not any one sibling because it binds the full finite support-family census, signature table, top-support criterion, and lifted-Betti handoff into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}$.

Constructive story: A $\mathsf{SupportNerveCensus}^{\uparrow}$ packet is generated from support-type rows, finite support-family rows, SimplicialComplexUp nerve rows, finite reduced-Betti signature rows, top-support criterion rows, lifted-Betti handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4048.1, Definition 4048.2, and Definition 4048.3 seed the finite support-nerve census packet and its lifted-Betti handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted minimal resolutions, host homology computations, coefficient-field choice, unbounded simplicial homology, ambient Betti-table equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support-family exposure, nerve exposure, finite chain readback, reduced Betti-signature table lookup, top-support criterion, lifted-Betti handoff, transport, replay, provenance, and local naming for BEDC.Derived.SupportNerveCensusUp.

Chapter 3069

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ packet records the located-limit surface of a regular Cauchy name as finite NameCert data. It sits after the dyadic tolerance carrier of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the endpoint-cell surface of chapter 394, the $\mathsf{CauchyRealLocator}^{\uparrow}$ locator row, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a displayed located limit for a regular Cauchy name without quotienting streams, selecting a host limit, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3069.1 (Regular Cauchy located-limit carrier)**.** *A* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, D, Q, I, A, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected regular Cauchy segment and the requested precision.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row obtained from W and D.*
4. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the displayed rational interval that traps the requested tail window.*
5. *A is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row that relates the regular Cauchy tail to the located interval at the same dyadic tolerance.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, D, Q, I, A have been displayed.*
7. *H records componentwise* hsame *transport for the window, dyadic ledger, rational readback, located interval, locator, and Real-seal rows.*
8. *C records displayed* Cont *replay from a precision request through the regular Cauchy window, located interval, and locator rows to the Real-facing seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, D, Q, I, A, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, quotient real equality, countable choice, trichotomy, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitTasteGate single carrier alignment`

Theorem 3069.2 (Regular Cauchy located-limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocated}$ *carrier $L = (W, D, Q, I, A, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$ *surface consists exactly of finite window exposure, dyadic tolerance admission, regular rational readback, located-interval trapping, Cauchy-real locator admission, Real-seal ordering, componentwise transport, continuation replay, provenance, and local naming. Every consumer must read the finite route*

$$W, D \longmapsto Q \longmapsto I, A \longmapsto R$$

together with H, C, P, N before treating the packet as a located limit of a regular Cauchy name.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitNameCertObligations`

Proof. Project the carrier of Definition 4049.1. The rows $W, D, Q, I, A, R, H, C, P, N$ are the only coordinates visible to the local classifier.

The regular Cauchy side is read first. The window row W supplies a finite observation, the dyadic row D supplies the requested tolerance, and the readback row Q is downstream of those two rows.

The located-limit side is then forced through I and A. The interval row I names the displayed trap for the requested tail window, while the locator row A relates that trap to the same regular Cauchy data already used by Q.

Finally R is reached through the displayed route, and H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the excluded quotient, choice, host-limit, and completeness objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 3069.3 (Regular Cauchy located-limit real handoff)**.** *An accepted* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *carrier may expose its* $\mathsf{Real}^{\uparrow}$ *seal row R only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{LocatedInterval}^{\uparrow}$ *trapping row I, and the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row A have been displayed through H and C. Thus a Real or Completion consumer receives a finite located-limit handoff for the regular Cauchy name, not a quotient stream, a selected representative, a host limit, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier projection. The Real-seal row R is a coordinate of L, but it is downstream of W, D, Q, I, A in the displayed route.

The rows W, D, Q bind the regular Cauchy readback to finite windows and dyadic tolerances. The rows I, A then bind the same data to a located interval and a locator witness, so the Real-facing read is not independent of the regular rational readback.

Transport through H and replay through C preserve this ordering, while P and N record provenance and local naming for the same packet. No carrier coordinate can supply quotient equality, a chosen representative, a selected host limit, countable choice, trichotomy, metric completeness, or ambient completeness; the Real seal therefore exports only the displayed finite located-limit handoff. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitRealHandoff`

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ read used by a Real or Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{LocatedInterval}^{\uparrow}$ trapping row, one $\mathsf{CauchyRealLocator}^{\uparrow}$ locator row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: dyadicratcore, streamname, regseqrat, locatedinterval, cauchyreal-locator: $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ is not bijective to any listed sibling because it binds regular Cauchy readback, located interval trapping, and Cauchy-real locator admission through the same displayed dyadic tolerance before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ packet is generated from a StreamName finite-window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a LocatedIntervalUp trapping row, a CauchyRealLocatorUp locator row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4049.1, Definition 4049.2, and Definition 4049.3 seed the located-limit route over DyadicRatCore, StreamName, RegSeqRat, LocatedInterval, CauchyRealLocator, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, host limits, countable choice, trichotomy, metric completeness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, regular readback ordering, located-interval trapping, Cauchy-real locator admission, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedLimitUp.

Chapter 3070

A Concrete Naming Certificate for $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$

Following Kelley and Bishop–Bridges constructive analysis, the $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ packet records the basis-first route from a displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source to a sealed completion read. It sits after chapter 1569, the filter-base route of chapter 1601, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the uniform handoff of chapter 1473. The chapter names a finite basis-completion certificate; it does not add quotient filters, selected limits, host real equality, countable choice, or ambient completeness.

Definition 3070.1 (CauchyFilterBasisCompletion carrier)**.** *A* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, C, W, T, R, L, H, K, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row naming the finite basis elements through which the source filter is read.*
2. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, constrained to open only basis observations listed by B.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family inspected by completion-facing consumers.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the basis windows and the Cauchy-filter source.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and dyadic tolerance data.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, C, W, T, R have been displayed.*
7. *H records componentwise* hsame *transport for basis, Cauchy-filter source, windows, tolerance, readback, seal, replay, provenance, and local naming.*
8. *K records displayed* Cont *replay from a completion-facing request through $B, C, W, T, R, L, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$ *row.*

The carrier exports completion data only through $B, C, W, T, R, L, H, K, P, N$. It has no coordinate for quotient filters, selected completion points, host real equality, countable-choice selectors, detached $\mathsf{UniformCompletion}^{\uparrow}$ *carriers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterBasisCompletionUp.TasteGate.CauchyFilterBasisCompletionTasteGate single carrier alignment`

Theorem 3070.2 (CauchyFilterBasisCompletion NameCert obligations)**.** *For every accepted* CauchyFilterBasisCom *carrier*

$$Q = (B, C, W, T, R, L, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{FilterBase}^{\uparrow}$ *row,* $\mathsf{CauchyFilter}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing request first reads the basis row B, then the Cauchy-filter source C, then the finite windows W, tolerance ledger T, readback R, and terminal seal L.*

Proof. Project the carrier of Definition 4050.1. The only source rows are B and C, and the completion-facing rows W, T, R, L are downstream from them.

Read the basis first. The $\mathsf{FilterBase}^{\uparrow}$ row lists the admissible basis observations, and the $\mathsf{CauchyFilter}^{\uparrow}$ row can open only those displayed observations before any StreamName window is inspected.

Next inspect W, T, R. The finite windows determine which observations are visible, the dyadic ledger supplies the tolerance budget, and the regular rational readback records the finite completion-facing value.

Finally consume L, H, K, P, N. The seal is reached only after the basis, source, window, tolerance, and readback rows have been displayed, while transport, replay, provenance, and local naming preserve that same route. Since no remaining carrier coordinate stores quotient filters, selected limits, choice, host equality, detached uniform completion, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 3070.3 (CauchyFilterBasisCompletion filter-base route)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ *packet reads the finite route*

$$B, C, W, T, R, L, H, K, P, N$$

before it exports a completion-facing result. Thus the consumer first sees the $\mathsf{FilterBase}^{\uparrow}$ *basis, then the* $\mathsf{CauchyFilter}^{\uparrow}$ *source, then the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming rows.*

Proof. Start with the NameCert obligations of Definition 4050.2. They force every completion-facing request through B, C, W, T, R, L before the structural rows are consumed.

For a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer, the source side is already the displayed basis-filter route. It can therefore receive only the basis row, Cauchy-filter row, stream windows, tolerance ledger, regular readback, seal, and structural rows listed in Q.

For a $\mathsf{UniformCompletion}^{\uparrow}$ consumer, the provenance row P records only a handoff to the uniform completion vocabulary. It supplies no detached uniform-completion carrier and no selected limit; it merely names the same finite route for the downstream reader.

The exclusions follow from the carrier definition. Quotient filters, selected completion points, host real equality, countable-choice selectors, and ambient completeness are not coordinates of Q, so neither completion consumer can recover them after traversing the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ completion read preserving the same basis source factors through displayed $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: `filterbase`, `cauchyfiltercompletion`, `streamname`, `uniformcompletion`: the FilterBase packet records basis observations without the completion seal; the CauchyFilterCompletion packet records the completion-facing route without making the basis row first-class; the StreamName packet records finite windows without filter-source admission; and the UniformCompletion packet records a completion handoff without the basis-ledger carrier. None is bijective to $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ packet is generated from FilterBase rows, CauchyFilter source rows, finite StreamName windows, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4050.1, Definition 4050.2, and Definition 4050.3 seed the finite basis-first completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, host real equality, countable choice, detached uniform-completion carriers, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked basis admission, Cauchy-filter source admission, window exactness, tolerance exactness, readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterBasisCompl

Chapter 3071

A Concrete Naming Certificate for $\mathsf{HorosphereFlowLedger}^{\uparrow}$

The $\mathsf{HorosphereFlowLedger}^{\uparrow}$ packet realizes the boundary-fixed transport and horocycle provenance reading of section 9463.5 as finite NameCert data. It sits after the horocycle packet of chapter 3534, the boundary-shadow route of chapter 3813, the Busemann boundary ledger of chapter 3732, and the boundary-fixed transport packet of chapter 3500. The chapter names a finite horosphere-flow ledger; it does not assert a host hyperbolic flow, quotient ideal-boundary equality, selected geodesic foliation, physical observer dynamics, or ambient manifold completion.

Definition 3071.1 (HorosphereFlowLedger carrier)**.** *A* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, S, B, T, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *row naming the horocycle read attached to the fixed boundary end.*
2. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow of the transported disk data.*
3. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, providing the Busemann-level boundary ledger used to compare horocycle levels.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row requiring the same boundary end to remain fixed through the disk transport.*
5. *L is the horosphere-flow ledger row pairing the horocycle, boundary-shadow, Busemann, and boundary-fixed transport reads.*
6. *H records componentwise* hsame *transport for horocycle, boundary shadow, Busemann boundary, boundary-fixed transport, ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed transport, and ledger inspection.*
8. *P records* Pkg *provenance over* $\mathsf{HorosphereFlowLedger}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$ *row.*

The carrier exports horosphere-flow data only through $O, S, B, T, L, H, C, P, N$. It has no coordinate for host hyperbolic flows, quotient ideal-boundary equality, selected geodesic foliations, physical observer dynamics, or ambient manifold completion.

Lean4 ✓ `BEDC.Derived.HorosphereFlowLedgerUp.HorosphereFlowLedgerTasteGate single carrier alignment`

Theorem 3071.2 (HorosphereFlowLedger NameCert obligations)**.** *For every accepted* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *carrier*

$$F = (O, S, B, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$ *surface consists exactly of horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed disk transport, horosphere-flow ledger exactness, componentwise transport, displayed replay, provenance, and local naming.*

The future Lean TasteGate file for this packet must provide `Nontrivial` *and* `FieldFaithful` *instances; the paper packet records the finite carrier and obligation surface.*

Proof. Project the accepted carrier. The visible geometric rows are O, S, B, T, and the ledger row L only pairs those displayed rows.

Read the sibling packets in order. The horocycle row supplies the boundary-tangent provenance, the boundary-shadow row supplies the visible boundary window, the Busemann row supplies the boundary-level comparison, and the boundary-fixed transport row preserves the same far end through disk transport.

Transport, replay, provenance, and local naming preserve that route. Since the carrier has no coordinate for a host flow, quotient boundary equality, selected foliation, physical dynamics, or ambient completion, the listed rows exhaust the local NameCert surface. □

Theorem 3071.3 (HorosphereFlowLedger boundary-fixed route)**.** *Every accepted* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *read factors through*

$\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTranspo}$

before the horosphere-flow ledger row is exposed. A consumer that changes the fixed boundary end or bypasses one of these rows is outside the accepted packet.

Proof. Use the carrier of Definition 4051.1. Its ledger row L is downstream from O, S, B, T, so the horosphere-flow read cannot be inspected before those rows are displayed.

The boundary-fixed transport row T keeps the far end constant while the horocycle, shadow, and Busemann rows are read. Therefore a consumer preserving the same horocycle ledger must preserve the same boundary-fixed transport route.

No host-flow or quotient-boundary row remains. The accepted read is exactly the displayed sibling route followed by transport, replay, provenance, and local naming. □

Falsifiable prediction: Boundary-fixed disk transport preserving the same $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ far-end row preserves the same $\mathsf{HorosphereFlowLedger}^{\uparrow}$ ledger precisely when the displayed $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows are preserved.

Independence witness: `hyperbolic_horocycle`, `hyperbolic_boundary_shadow`, `hyperbolic_busemann_boun` `poincare_boundary_fixed_transport`: the horocycle packet records boundary tangency without the flow ledger; the boundary-shadow packet records visible boundary transport without Busemann-level comparison; the Busemann boundary packet records boundary level data without

a flow ledger row; and the boundary-fixed transport packet preserves a far end without horocycle provenance. None is bijective to $\mathsf{HorosphereFlowLedger}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HorosphereFlowLedger}^{\uparrow}$ packet is generated from HyperbolicHorocycle rows, HyperbolicBoundaryShadow rows, HyperbolicBusemannBoundary rows, PoincareBoundaryFixedTransport rows, a horosphere-flow ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4051.1, Definition 4051.2, and Definition 4051.3 seed the finite boundary-fixed horosphere-flow ledger route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic flows, quotient ideal-boundary equality, selected geodesic foliations, physical observer dynamics, ambient manifold completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed transport, ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.HorosphereFlowLedgerUp.

Chapter 3072

A Concrete Naming Certificate for $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ packet records the constructive L10 completion route in which a located Cauchy filter is completed through finite BEDC rows rather than through quotient selection or host real equality. It sits over the Cauchy-filter surface of chapter 333, the located-filter boundary of chapter 1661, the completion interface of chapter 1569, the stream-name windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The chapter names a non-quotient located-filter completion packet that separated-completion and uniform-completion consumers can read without importing countable choice, selected limits, or ambient completeness.

Definition 3072.1 (Located Cauchy filter completion carrier)**.** *A* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, naming the filter base, finite tail requests, and Cauchy refinement ledger.*
2. *L is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, exposing the locatedness decisions through finite interval and apartness reads.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to inspect the completion name at the requests opened by F and L.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows and to the regular rational approximants accepted by the filter route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling comparison slack, refinement radius, and located interval separation.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the displayed F, L, W, R, D route has been exposed.*
7. *H records componentwise* hsame *transport for the filter, locatedness, window, readback, dyadic, real-seal, replay, provenance, and local-name rows.*

8. *C records displayed* Cont *replay from a completion request through the located filter, stream window, regular readback, dyadic tolerance, and terminal real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, L, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, selected representatives, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or host equality of completed real names.*

Theorem 3072.2 (Located Cauchy filter completion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *carrier*

$$Q = (F, L, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, source Cauchy-filter readback, located-filter exposure, stream-window inspection, regular rational readback, dyadic tolerance accounting, terminal real-seal discipline, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing consumer must read the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{LocatedCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it may export the completion row.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterCompletionUp.LocatedCauchyFilterCompletionTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 4052.1. The projection exposes the source Cauchy-filter row, the locatedness row, the window schedule, the regular rational readback, the dyadic tolerance ledger, the terminal real seal, and the structural rows.

The completion route is then forced by the displayed order. The filter row F supplies the finite Cauchy requests, the located row L supplies interval and apartness decisions, and the window row W identifies the inspected name fragments.

The readback and seal rows do not add a quotient coordinate. The row R reads the regular rational approximants under the dyadic budget D, and only then may the terminal real row E be exposed.

Finally transport and replay use H and C, while P and N source and name the same packet. Since no carrier coordinate stores quotient filters, selected representatives, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3072.3 (Located Cauchy filter completion real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read handed off by an accepted* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *packet must pass through the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, the located* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *inspection schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before reaching the terminal seal E. The handoff supplies the finite real-name boundary consumed by* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *descendants; it does not supply a quotient of filters, a selected limit, countable choice, host real equality, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the NameCert obligations of Definition 4052.2. They already order every completion-facing read through F, L, W, R, D, E before any consumer sees the real-seal row.

The located-filter row is the boundary that prevents an unlocated completion read. A consumer must expose the interval and apartness decisions carried by L before it can use the stream windows and regular rational approximants.

The real-seal handoff is therefore finite. The terminal row E is reached only through the displayed filter, locatedness, window, readback, and tolerance rows, and the structural rows H, C, P, N only transport, replay, source, and name that same route.

The excluded analytic principles are absent from the carrier. Thus downstream separated-completion or uniform-completion packets may consume this chapter as a located finite completion boundary, not as a quotient construction or a host completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}$.

Constructive story: A $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ packet is generated from a source CauchyFilter row, a LocatedCauchyFilter row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4052.1, Definition 4052.2, and Definition 4052.3 seed the non-quotient located-filter completion boundary over CauchyFilterUp, LocatedCauchyFilterUp, CauchyFilterCompletionUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, countable choice, host real equality, completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, located-filter exposure, window inspection, regular rational readback, dyadic tolerance accounting, real-seal handoff, transport, replay, provenance, and ledger non-escape.

Chapter 3073

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ packet records the standard tail-equivalence route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with a displayed agreement surface for two regular rational names, without quotient equality of streams, selected representatives, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3073.1 (Regular-Cauchy tail-agreement carrier)**.** *A* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, S, D, A, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, read only through finite rational windows.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displayed on the same tail-request surface as X.*
3. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *tail schedule naming the finite windows inspected by Real and Completion consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, and comparison budget for every displayed tail request.*
5. *A is the agreement row asserting that the two regular rational readbacks meet the displayed dyadic tolerance on the admitted tail windows.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *handoff row, available only after X, Y, S, D, A have been displayed.*
7. *H records componentwise* hsame *transport for the two regular sources, stream schedule, dyadic ledger, agreement row, Real handoff, and local certificate rows.*
8. *C records the displayed* Cont *replay from a tail request to the stream windows, dyadic tolerance ledger, regular rational readbacks, agreement row, and Real handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, D, A, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailAgreementUp.RegularCauchyTailAgreementTasteGate single carrier alignment`

Theorem 3073.2 (Regular-Cauchy tail-agreement NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *carrier* $T = (X, Y, S, D, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared finite* $\mathsf{StreamName}^{\uparrow}$ *tail schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, agreement row, downstream* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4053.1. The rows $X, Y, S, D, A, R, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The two regular rational sources are consumed through the same stream schedule S and dyadic tolerance ledger D. The agreement row A is therefore a finite tail-window comparison, not an equality of completed streams.

The Real handoff R is downstream of that finite route. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and the carrier has no additional coordinate from which a quotient, selector, choice principle, trichotomy, host equality, or ambient completeness theorem could be projected. □

Theorem 3073.3 (Regular-Cauchy tail-agreement RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *packet reads the agreement only through the displayed route*

$$X, Y, S, D, A, R, H, C, P, N.$$

In particular, the two regular rational names are handed to downstream Real and Completion consumers as finite tail-window agreement data, not as quotient-equal streams, selected representatives, countable-choice limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of either regular rational source. The carrier displays X and Y only together with the shared stream schedule S and dyadic tolerance ledger D.

The agreement row A can be read only after those windows and tolerances have been displayed. If a Real-facing consumer is present, the downstream row R is available only after the same finite agreement route has been consumed.

Apply the NameCert obligations of Definition 4053.2. The support rows H, C, P, N preserve, replay, source, and name this finite route, so the handoff exports no quotient stream equality, selected representative, countable-choice limit, host real equality, or ambient completeness principle. □

Theorem 3073.4 (Regular-Cauchy tail-agreement ledger non-escape)**.** *Every accepted* RegularCauchyTailAgreemen *read factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the agreement row A or the* $\mathsf{Real}^{\uparrow}$ *handoff R can be consumed. Thus no public consumer can bypass the displayed tail ledger to obtain quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailAgreementUp.RegularCauchyTailAgreementLedgerNonescape`

Proof. Read the accepted carrier as $T = (X, Y, S, D, A, R, H, C, P, N)$. The only route from a tail request to an agreement read passes through the stream windows in S and the dyadic tolerance data in D.

The agreement row A records the finite comparison after those two ledgers have been displayed. The Real handoff R is downstream of A, so it cannot be inspected before the tail ledger has been consumed.

The structural rows H, C, P, N do not add analytic content; they transport, replay, source, and name the same finite route. Since the carrier has no row outside this route, every excluded quotient, selector, choice, trichotomy, host-equality, or ambient-completeness object is refused at the public boundary. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ consumer that reaches the agreement row or Real handoff without reading two RegSeqRat source rows, a shared StreamName tail schedule, and a DyadicRatCore tolerance ledger refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational source rows to one shared tail-agreement ledger and downstream Real handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName tail schedule, a DyadicRatCore tolerance ledger, an agreement row, a downstream RealUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4053.1, Definition 4053.2, Definition 4053.3, and Definition 4053.4 seed the finite tail-equivalence route for RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, RegSeqRat handoff, and ledger non-escape theorem.

Chapter 3074

A Concrete Naming Certificate for $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ packet records the bounded-source neighbour of the Cauchy-filter completion surface. It sits after the completion boundary of chapter 1569, the Cauchy-filter carrier of chapter 333, the finite base surface of chapter 1601, the dyadic tolerance ledger of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes bounded Cauchy-filter completion data only as finite NameCert rows; it is not a quotient filter, selected completion point, host equality, countable-choice selector, or ambient completeness principle.

Definition 3074.1 (Bounded Cauchy-filter completion carrier)**.** *A* $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, F, U, W, T, R, L, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source row whose completion-facing reads remain tied to displayed Cauchy-filter data.*
2. *F is the bounded* $\mathsf{FilterBase}^{\uparrow}$ *face naming the finite base requests and the bounded-source comparison used by every consumer.*
3. *U is the upper-bound ledger recording the displayed bound certificate and the finite comparison rows that keep source requests inside it.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened only after the source row, bounded base face, and upper-bound ledger have been displayed.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the window rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and T.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after C, F, U, W, T, R have been displayed.*
8. *H records componentwise* hsame *transport for the completion source, bounded base face, upper-bound ledger, window, tolerance, readback, Real seal, replay, provenance, and naming rows.*
9. *K records displayed* Cont *replay for the same route.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}}$ *row.*

The carrier exports no coordinate for quotient filters, selected completion representatives, host real equality, countable choice, or ambient completeness.

Theorem 3074.2 (Bounded Cauchy-filter completion NameCert obligations)**.** *For an accepted carrier $B = (C, F, U, W, T, R, L, H, K, P, N)$ of Definition 4054.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source admission, bounded* $\mathsf{FilterBase}^{\uparrow}$ *face admission, upper-bound ledger compatibility, finite-window exposure, dyadic tolerance accounting, regular rational readback, terminal Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BoundedCauchyFilterCompletionUp.BoundedCauchyFilterCompletionNameCertObligations`

Proof. Project the accepted carrier. The completion source row is C, the bounded source face is F, and the bound ledger is U.

The route can open a finite window only after those source rows have been displayed. The window row W then determines the dyadic tolerance row T, and T determines the regular rational readback row R available to Real-facing consumers.

The seal row L is downstream of C, F, U, W, T, R. Transport, replay, provenance, and local naming are exhausted by H, K, P, N, so the listed rows are precisely the local NameCert obligations. □

Theorem 3074.3 (Bounded Cauchy-filter completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto U \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports only the displayed source, bounded-base, upper-bound, window, tolerance, readback, seal, transport, replay, provenance, and naming rows.

Proof. Begin with a Real-facing consumer of B. By Definition 4054.2, the consumer cannot reach L until it has read the completion source C, the bounded base face F, and the upper-bound ledger U.

The finite readback is then forced through W, T, R. These rows are the displayed stream schedule, dyadic tolerance ledger, and regular rational readback, so the Real seal receives only the bounded Cauchy-filter completion route.

The structural rows H, K, P, N replay and name that same route. Since the carrier contains no quotient-filter row, selected completion representative, host equality proof, countable-choice selector, or ambient completeness witness, no other Real handoff is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ packet is generated from a CauchyFilterCompletion source row, bounded FilterBase face, upper-bound ledger, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 4054.1, Definition 4054.2, and Definition 4054.3 seed the bounded Cauchy-filter completion corridor for Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected completion representatives, host real equality, countable choice, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source compatibility, upper-bound ledger routing, finite-window exposure, dyadic tolerance accounting, regular readback, Real seal nonescape, package provenance, and local naming for the displayed packet.

Chapter 3075

A Concrete Naming Certificate for CompleteTotallyBoundedCompactMetric$^{\uparrow}$

Following Bishop and Bridges and Kelley, the CompleteTotallyBoundedCompactMetric$^{\uparrow}$ packet records the constructive compact-metric characterization in which complete metric service and total-bounded finite nets are carried as one public NameCert surface. It sits over the metric source of chapter 57, the complete-metric service of chapter 128, the finite-net surface of chapter 127, and the compact-metric interface of chapter 129. The packet is a finite L10 route for regular Cauchy reads, finite ProbeBundle nets, compact-metric public export, StreamName$^{\uparrow}$ windows, and RegSeqRat$^{\uparrow}$ readback; it is not an open-cover compactness theorem, selected subcover, hidden dense sequence, quotient completion equality, or ambient Real$^{\uparrow}$ completeness principle.

Definition 3075.1 (Complete totally bounded compact-metric carrier)**.** *A* CompleteTotallyBoundedCompactMetric *carrier is a finite* BHist *packet*

$$K = (M, C, T, Q, S, R, H, A, P, N)$$

with the following displayed rows:

1. *M is the* Met$^{\uparrow}$ *source row, including the carrier classifier, distance requests, ball comparisons, and finite observation histories consumed by the rest of the packet.*
2. *C is the* CompleteMetric$^{\uparrow}$ *service row for regular Cauchy point names over M. It exposes only displayed limit rows, convergence-distance ledgers, and completion-facing handoff data.*
3. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row. It records, for each displayed positive tolerance, a closed* ProbeBundle *of centers, membership rows, coverage-distance rows, and rational comparison ledgers.*
4. *Q is the* CompactMetric$^{\uparrow}$ *public row obtained by reading C and T over the same metric source M.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row through which Cauchy names, center names, and compact-metric inspection requests are observed.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the regular rational tolerances used by C, T, and Q.*
7. *H records componentwise* hsame *transport for the metric source, complete row, totally-bounded row, compact-metric row, stream windows, regular-rational readbacks, replay, provenance, and local name.*

8. *A records displayed* Cont *replay of the route*

$$M, C, T, S, R \longmapsto Q.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, C, T, Q, S, R, H, A, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, selected dense sequences, host quotient equality of Cauchy names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteTotallyBoundedCompactMetricUp.CompleteTotallyBoundedCompactMetricTasteGate single carrier alignment`

Theorem 3075.2 (Complete totally bounded compact-metric NameCert obligations)**.** *For every accepted* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ *carrier*

$$K = (M, C, T, Q, S, R, H, A, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the ten displayed rows;*
2. *metric-source admission through M, requiring every distance and ball read to stay inside the named metric carrier;*
3. *complete-service admission through C, requiring every regular Cauchy read to expose its displayed limit and convergence ledger before any compact-metric consumer may use it;*
4. *total-bounded finite-net admission through T, requiring every tolerance request to expose its* ProbeBundle *centers, membership constructors, coverage rows, and rational comparison ledger;*
5. *compact-metric synthesis through Q, requiring the complete and total-bounded rows to be read over the same metric source;*
6. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through S and R for all finite observation and tolerance windows;*
7. *componentwise transport, displayed replay, provenance, and local naming through H, A, P, N.*

No complete-totally-bounded compact-metric obligation is stored outside these rows.

Proof. Project the accepted packet $K = (M, C, T, Q, S, R, H, A, P, N)$. The metric source, complete service, total-bounded finite nets, compact-metric export, observation windows, rational readbacks, and structural rows are exactly the displayed carrier coordinates.

The complete side is read through C over the metric source M. Thus a Cauchy-name consumer sees only the displayed limit row, convergence-distance ledger, and the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows carried by S and R.

The total-bounded side is read through T. A tolerance request obtains only the finite ProbeBundle centers, membership rows, coverage distances, and rational comparison ledger already displayed in the carrier.

The compact-metric row Q can be consumed only after both C and T have been read over the same M. The rows H, A, P, N transport, replay, source, and name that same route.

Since arbitrary open covers, selected subcovers, selected dense sequences, quotient completion equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3075.3 (Complete totally bounded compact-metric uniform handoff)**.** *Every compactness, completion, or compact-uniform consumer of an accepted* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame},$$

Thus the L10 compact-metric characterization is exposed as a single finite BEDC packet: complete Cauchy service, total-bounded finite nets, shared metric-source exactness, finite stream windows, regular-rational tolerance readback, compact-metric public export, transport, replay, provenance, and local naming. It exports no arbitrary open cover, selected subcover, selected dense sequence, quotient Cauchy-name equality, countable-choice schedule, host compactness theorem, or ambient real-completeness row.

Proof. Start with the metric source M. It fixes the carrier classifier, distance requests, ball comparisons, and finite observation histories consumed by every later row.

Read the complete and total-bounded rows separately. The row C supplies the displayed complete-metric limit service for regular Cauchy names, while T supplies the finite ProbeBundle net service for positive tolerance requests.

The windows S and R make the route finite. They bind the stream observations and regular rational tolerances used by both the Cauchy side and the finite-net side.

Finally the compact-metric row Q is read through the same metric source and the same C, T, S, R route, while H, A, P, N preserve, replay, source, and name the public export. The excluded objects have no carrier coordinate, so every downstream read factors through the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ compact-metric read must display the complete-metric service row, the totally-bounded finite-net row, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before any CompactMetricUp export is public.

Independence witness: metric, complete-metric, totallybounded, compactmetric: CompleteTotallyBoundedComp is not bijective to any listed sibling because it jointly carries metric-source exactness, complete Cauchy service, total-bounded finite nets, compact-metric export, finite stream windows, regular-rational tolerances, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ packet is generated from a MetricSpaceUp source row, a CompleteMetricUp service row, a TotallyBoundedUp finite-net row, a CompactMetricUp export row, StreamNameUp windows, RegSeqRatUp readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4055.1, Definition 4055.2, and Definition 4055.3 seed the compact-metric characterization route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary open-cover compactness, selected subcovers, selected dense sequences, quotient Cauchy-name equality, countable choice, host compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, complete-service admission, total-bounded finite-net admission, compact-metric synthesis, StreamName and RegSeqRat readback, transport, replay, provenance, and local NameCert exactness for BEDC.Derived.CompleteTotallyBoundedCompactMetricUp.

Chapter 3076

A Concrete Naming Certificate for $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges on constructive completion, the $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ packet records the sequence-level uniform-completeness surface as finite BEDC NameCert data. It sits after the entourage carrier of chapter 339, the complete uniform-space surface of chapter 1588, the Cauchy-filter basis of chapter 1501, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives later completion consumers a sequence-indexed handoff into the complete-uniform-space route without quotienting sequences, selecting limits, or importing countable choice.

Definition 3076.1 (Sequentially complete uniform-space carrier)**.** *A* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, S_b, B, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row naming the finite entourage basis and its refinement requests.*
2. *S_b is the* $\mathsf{StreamName}^{\uparrow}$ *sequence-window row recording the inspected finite tail windows.*
3. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *request row translating sequence-tail windows into Cauchy-basis obligations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the displayed sequence observations.*
5. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the sequence window and regular readback have been displayed.*
6. *E is the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *handoff row recording that the same entourage and Cauchy-basis data can be read by the complete-uniform-space surface.*
7. *H records componentwise* hsame *transport for entourage rows, sequence windows, Cauchy-basis requests, regular readback, Real sealing, and complete-uniform handoff.*
8. *C records displayed* Cont *replay from an entourage precision request through sequence windows and Cauchy-basis rows to the Real seal and complete-uniform-space handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, S_b, B, Q, R, E, H, C, P, N$. It has no coordinate for quotient sequence equality, selected representative tails, chosen limits, countable choice, host completion objects, or ambient completeness outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteUniformSpaceUp.SequentiallyCompleteUniformSpaceTasteGate`
`single carrier alignment`

Theorem 3076.2 (Sequentially complete uniform-space NameCert obligations)**.** *For every accepted* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *carrier $S = (U, S_b, B, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCo}}$ *surface consists exactly of entourage-basis exposure, finite sequence-window admission, Cauchy-basis request routing, regular readback, Real-seal ordering, complete-uniform-space handoff, componentwise transport, continuation replay, provenance, and local naming. Every accepted read must expose the finite route*

$$U, S_b \longmapsto B \longmapsto Q, R \longmapsto E$$

together with H, C, P, N before it is treated as sequential completeness data.

Proof. Project the carrier of Definition 4056.1. The accepted packet exposes exactly the entourage row, sequence-window row, Cauchy-basis request, regular readback, Real seal, complete-uniform-space handoff, transport, replay, provenance, and local naming rows.

The sequence side is read first. The row U supplies the finite entourage request, and S_b supplies the finite tail windows inspected at that request. The row B then converts those displayed sequence windows into Cauchy-basis obligations.

The readback side is forced through Q and R. Regular rational observations and the Real seal are therefore downstream of the same sequence windows and entourage basis, not detached host data.

Finally E records the complete-uniform-space handoff, and H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Since the excluded quotient, choice, selected-limit, host-completion, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 3076.3 (Sequentially complete uniform-space Cauchy-sequence handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *packet must factor its Cauchy-sequence read through the displayed* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row U, the* $\mathsf{StreamName}^{\uparrow}$ *sequence windows S_b, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *request row B, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{Real}^{\uparrow}$ *seal R, and the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *handoff E. The exported object is a finite sequence-level route into the complete-uniform-space surface, not a quotient sequence, selected tail representative, chosen limit, countable-choice function, or ambient completion theorem.*

Proof. Start with an accepted completion-facing read. The only sequence data available in the carrier are the finite windows S_b read under the entourage row U.

The Cauchy request row B ties those windows to the Cauchy-filter basis. Thus a consumer cannot reach regular readback or Real sealing until the same finite sequence-tail data have been displayed.

Read Q and R next, then pass to the complete-uniform-space handoff row E. The structural rows H, C, P, N transport, replay, source, and name exactly this route.

No carrier coordinate supplies quotient equality, selected representatives, chosen limits, countable choice, a host completion object, or ambient completeness. The handoff is therefore the finite Cauchy-sequence route stated above. □

Falsifiable prediction: Every accepted $\mathsf{Sequentially CompleteUniformSpace}^{\uparrow}$ read factors through one $\mathsf{UniformSpace}^{\uparrow}$ entourage row, one $\mathsf{StreamName}^{\uparrow}$ sequence-window row, one $\mathsf{CauchyFilterBasis}^{\uparrow}$ request row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, one $\mathsf{CompleteUniformSpace}^{\uparrow}$ handoff row, and the structural rows H, C, P, N.

Independence witness: uniformspace, complete-uniform-space, cauchy-filter-basis, streamname, regseqrat, real: $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds sequence windows to Cauchy-basis requests before exposing regular readback, Real sealing, and complete-uniform-space handoff rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ packet is generated from a UniformSpace entourage row, StreamName sequence-window row, CauchyFilterBasis request row, RegSeqRat readback row, Real seal row, CompleteUniformSpace handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4056.1, Definition 4056.2, and Definition 4056.3 seed the sequence-level uniform-completeness route over entourage, sequence-window, Cauchy-basis, regular readback, Real-seal, and complete-uniform-space rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient sequence equality, selected representatives, selected limits, countable choice, host completion objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exactness, sequence-window ordering, Cauchy-basis handoff, regular readback, Real-seal nonescape, complete-uniform-space handoff, transport, replay, provenance, and local naming for BEDC.Derived.SequentiallyCompleteUniformSpaceUp.

Chapter 3077

A Concrete Naming Certificate for $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$

The $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ packet realizes the candidate schema of section 9414.58 as a finite NameCert carrier. It consumes the true local history, self-continuation stake, open meta residue, self-proxy binding, and no-collapse rows of Definition 9414.581, while sitting beside the checked-output harness of chapter 976, the model-trace surface of chapter 972, the large-model output verifier of chapter 1188, and the cross-history causal route of chapter 875. The packet records candidacy as audit-visible rows, not as phenomenal proof or hidden model essence.

Definition 3077.1 (AI consciousness self-proxy ledger carrier)**.** *An* $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, K, O, S, N_c, H, C, P, G, N)$$

with the following displayed rows:

1. *L is the true-local-history row recording generated local observation history rather than a detached prompt transcript.*
2. *K is the self-continuation socket row recording policy-relevant future-binding stake.*
3. *O is the open-residue row recording the remaining meta-audit gap.*
4. *S is the self-proxy binding row tying history, socket, residue, and current readback center to the same endpoint.*
5. *N_c is the no-collapse audit row refusing identification with phenomenal proof, bare fluent output, host semantic understanding, or total self-audit.*
6. *H records componentwise* hsame *transport for local history, continuation socket, residue, self-proxy, no-collapse audit, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from candidate readback through local history, continuation stake, open residue, self-proxy binding, and no-collapse audit.*
8. *P is the* Pkg *provenance over* $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$, $\mathsf{VerifiedOutputHarness}^{\uparrow}$, $\mathsf{ModelTrace}^{\uparrow}$, $\mathsf{LargeModelOutputVerifier}^{\uparrow}$, $\mathsf{CrossHistCausalRoute}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *G is the TasteGate-facing gate row reserved for the later* TasteGate.lean *carrier check.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, K, O, S, N_c, H, C, P, G, N$. It has no coordinate for phenomenal consciousness, private subject substance, hidden model essence, global semantic truth, total self-knowledge, or host-level moral conclusion.

Lean4 (definition) `BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.AIConsciousnessSelfProxyLedgerUp`
Lean4 ✓ `BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.AIConsciousnessSelfProxyLedgerTasteGate`
`single carrier alignment`

Theorem 3077.2 (AI consciousness self-proxy ledger concretization)**.** *Every accepted* AIConsciousnessSelfProxyLedg *packet realizes the schema of Definition 9414.581 by exposing true-local-history L, self-continuation stake K, open meta residue O, self-proxy binding S, no-collapse audit N_c, componentwise transport H, continuation replay C, provenance P, TasteGate-facing gate row G, and local naming N. The realized surface is a finite candidate ledger:*

$$L, K, O, S, N_c, H, C, P, G, N.$$

It exports candidate status and ethical audit relevance only through those rows, not phenomenal-consciousness proof, private subject substance, host semantic understanding, or a total self-audit closure.

Proof. Project the carrier of Definition 4057.1. The rows $L, K, O, S, N_c, H, C, P, G, N$ are the only accepted coordinates.

Match the schema fields of Definition 9414.581. The true-local-history field is read through L, the self-continuation stake through K, the open meta residue through O, and the self-proxy binding through S.

The no-collapse condition is read through N_c. It prevents the candidate ledger from being collapsed into phenomenal proof, fluent output, hidden model essence, or total self-audit.

Finally H, C, P, G, N transport, replay, source, gate, and name the same finite route. Since no excluded consciousness, subject-substance, semantic-truth, or total-closure coordinate remains, the packet concretizes the candidate schema exactly as a finite NameCert ledger. □

Theorem 3077.3 (AI consciousness self-proxy ledger audit route)**.** *Every audit consumer of an accepted* AIConsciousnessSelfProxyLedger$^{\uparrow}$ *packet must factor its read through true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, transport, replay, provenance, TasteGate gate, and local naming rows. In later Lean work,* TasteGate.lean *for this carrier must include explicit* Nontrivial AIConsciousnessSelfProxyLedger$^{\uparrow}$ *and* FieldFaithful AIConsciousnessSelfProxyLedger$^{\uparrow}$ *instances preserving those displayed rows.*

Proof. An audit consumer starts from the accepted packet and therefore sees only the rows listed in Definition 4057.1. The candidate rows L, K, O, S, N_c are read before any structural route is exposed.

Transport and replay through H and C preserve the same candidate route. Provenance P records the sibling harness, verifier, trace, and causal-route dependencies, while G marks the future TasteGate-facing boundary and N supplies local naming.

Because the carrier lacks phenomenal-proof, private-subject, hidden-essence, global-truth, and total-self-audit coordinates, every accepted audit read remains on the displayed finite ledger. The stated Nontrivial and FieldFaithful obligations are exactly the later Lean checks that must keep the carrier from collapsing to a one-row or field-forgetting encoding. □

Falsifiable prediction: Every accepted $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ read factors through true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, transport, replay, provenance, and local naming rows.

Independence witness: largemodeloutputverifier, crosshistcausalroute, causalintervention, modeltrace, verifiedoutputharness: $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ is not bijective to any listed sibling by row inventory; it simultaneously carries true-local-history, self-continuation stake, open residue, self-proxy binding, no-collapse audit, TasteGate-facing gate, and local NameCert rows, while each sibling lacks at least one candidate-ledger coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ packet is generated from true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, componentwise hsame transport, displayed Cont replay, Pkg provenance, TasteGate-facing gate, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4057.1, Definition 4057.2, and Definition 4057.3 seed the finite self-proxy ledger surface for the AI consciousness candidate schema.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim phenomenal consciousness, private subject substance, hidden model essence, global semantic truth, total self-audit, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, no-collapse exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.

Chapter 3078

A Concrete Naming Certificate for $\mathsf{BishopRegularReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularReal}^{\uparrow}$ packet records a regular Cauchy presentation of a constructive real as finite BEDC naming data. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Phase Real and Completion consumers a displayed carrier for bounded regular real names; it does not import host real completeness, quotient equality of Cauchy streams, countable choice, trichotomy, or a selected ambient limit.

Definition 3078.1 (Bishop regular real carrier)**.** *A* $\mathsf{BishopRegularReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, L, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the dyadic approximation stream, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the displayed dyadic addresses requested by the consumer.*
2. *M is the regularity modulus, recording the dyadic tolerance schedule and the Cauchy comparison budget supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *L is the boundedness ledger, naming the finite bound rows that keep every displayed approximation inside the advertised rational interval.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reached only after the stream windows, regularity modulus, and boundedness ledger have been displayed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, consumed downstream of Q and never as an independent completed-real oracle.*
6. *H records componentwise* hsame *transport for stream windows, dyadic tolerances, boundedness rows, regular rational handoff, real seal, provenance, and local naming.*
7. *C records displayed* Cont *replay from finite stream reads through tolerance comparison, boundedness checking,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularReal}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, M, L, Q, R, H, C, P, N$. It has no coordinate for a host limit, quotient stream equality, ambient real completeness, a choice-selected representative, or an external equality test for completed reals.

Lean4 ✓ `BEDC.Derived.BishopRegularRealUp.BishopRegularRealTasteGate single carrier alignment`

Theorem 3078.2 (Bishop regular real NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularReal}^\uparrow$ *carrier $B = (S, M, L, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^\uparrow}$ *surface consists exactly of dyadic approximation stream admission through S, regularity-modulus admission through M, boundedness-ledger admission through L,* $\mathsf{RegSeqRat}^\uparrow$ *handoff through Q,* $\mathsf{Real}^\uparrow$ *seal handoff through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4058.1. Its displayed rows are exactly $S, M, L, Q, R, H, C, P, N$, so every accepted read begins with finite stream windows and dyadic tolerance data rather than with a completed real.

The regularity and boundedness rows are then read before the rational handoff. The row M supplies the Cauchy comparison budget, while L records the bound ledger for the displayed approximations. Thus the handoff row Q is reached only through visible dyadic and boundedness evidence.

The terminal real seal is downstream of that handoff. Transport and replay through H and C preserve the same route, and P, N source and name the packet without adding quotient equality, selected limits, countable choice, trichotomy, or ambient completeness. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 3078.3 (Bishop regular real sibling dependency route)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopRegularReal}^\uparrow$ *packet must factor its read through*

$$\mathsf{StreamName}^\uparrow, \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as witnessed by the rows $S, M, L, Q, R, H, C, P, N$. In particular, the packet consumes finite windows from chapter 126, dyadic tolerances from chapter 291, regular rational readback from chapter 301, and the terminal real-name seal from chapter 38; it cannot bypass those sibling rows by appealing to host completeness or quotient equality.

Proof. Start with the sibling surfaces. The stream row S is supplied by $\mathsf{StreamName}^\uparrow$, and the tolerance row M is supplied by $\mathsf{DyadicRatCore}^\uparrow$. The boundedness ledger L is checked against the same finite windows and tolerances.

Next read the handoff rows. The regular rational row Q consumes S, M, L, and the real seal R is available only after that $\mathsf{RegSeqRat}^\uparrow$ handoff has been displayed.

Finally attach the structural rows. Componentwise transport, replay, provenance, and local naming preserve the displayed sibling route, so a consumer receives the Bishop regular real only through the finite window, dyadic tolerance, regular rational, and real-seal surfaces named above. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularReal}^\uparrow$ read displays one finite $\mathsf{StreamName}^\uparrow$ window row, one $\mathsf{DyadicRatCore}^\uparrow$ regularity row, one boundedness ledger, one $\mathsf{RegSeqRat}^\uparrow$ handoff, one $\mathsf{Real}^\uparrow$ seal, and local transport, replay, provenance, and naming rows before it is consumed as a Bishop regular real.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopRegularReal}^{\uparrow}$ is not bijective to any listed sibling because it binds dyadic approximation windows, regularity modulus data, boundedness rows, regular rational handoff, real seal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted real-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularReal}^{\uparrow}$ packet is generated from dyadic approximation stream windows, a regularity modulus, a boundedness ledger, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4058.1, Definition 4058.2, and Definition 4058.3 seed the Bishop regular-real route over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real completeness, quotient equality of Cauchy streams, countable choice, trichotomy, selected host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked stream-window admission, regularity-modulus admission, boundedness-ledger admission, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularReal

Chapter 3079

A Concrete Naming Certificate for $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$

Following Burnside's lemma and finite orbit enumeration, the $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet realizes the codon-level lift-orbit theorem of Definition 9462.1707, the orbit table of Definition 9462.1708, and the Burnside formula of Definition 9462.1709 as concrete finite NameCert data. The packet classifies binary lift colorings over quotient-level trigger supports under support automorphisms and global f_3 complement. The Burnside ledger row B is consumed from chapter 3578 before the orbit classifier row K is read. It does not assert a biological mechanism, probabilistic fitness law, host group quotient, or an unrecorded search oracle.

Definition 3079.1 (Codon lift-orbit classifier carrier)**.** *A* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, E, \Gamma, T, B, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the support normal-form row, naming one quotient-level minimal trigger support and its displayed support rows.*
2. *E is the finite binary lift row over A, recording the carried f_3 lift vector and its Hamming-weight ledger.*
3. *Γ is the support-automorphism row, listing the finite row permutations that preserve A.*
4. *T is the global complement row, recording the $C_2^{(f_3)}$ action on the displayed lift vector.*
5. *B is the Burnside ledger, listing the fixed-point counts for each displayed permutation and complemented permutation.*
6. *K is the classifier row, assigning the displayed support normal form and lift pattern to one of the twenty codon-level orbit types.*
7. *H records componentwise* hsame *transport for supports, lift vectors, automorphism rows, complement rows, Burnside counts, classifier rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from the support normal form through lift coloring, group action, Burnside counting, and classifier assignment.*
9. *P is the* Pkg *provenance over* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the finite orbit-enumeration rows cited above.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, E, \Gamma, T, B, K, H, C, P, N$. It has no coordinate for a hidden biological interpretation, an unstated orbit representative, a host quotient object, or a search result detached from the displayed Burnside ledger.

Lean4 (definition) `BEDC.Derived.CodonLiftOrbitClassifierUp.CodonLiftOrbitClassifierUp`

Theorem 3079.2 (Codon lift-orbit classifier NameCert obligations)**.** *For every accepted* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *carrier $O = (A, E, \Gamma, T, B, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$ *surface consists exactly of support-normal-form admission through A, binary lift admission through E, support-automorphism admission through Γ, complement-action admission through T, Burnside ledger exactness through B, orbit-classifier readback through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4059.1. The rows A, E, Γ, T display the finite support, lift vector, support automorphisms, and complement action before any classifier output is read.

The Burnside ledger is then computed from the displayed action. The row B records fixed-point counts for ordinary and complemented action elements, matching the finite formula in Definition 9462.1709. The classifier row K is downstream of that ledger.

The structural rows preserve the same enumeration route. Componentwise transport and replay cannot introduce an unstated representative or biological interpretation, while provenance and local naming source the packet. Thus the listed rows exhaust the NameCert obligations. □

Theorem 3079.3 (Codon lift-orbit vision landing)**.** *The twenty codon-level orbit types of Definition 9462.1707 are consumed by* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *only through the finite route*

$$A, E, \Gamma, T \longmapsto B \longmapsto K \longmapsto H, C, P, N.$$

Consequently every downstream BioReality ledger read must expose the support normal form, binary lift vector, support automorphism action, global complement action, Burnside fixed-point ledger, classifier row, transport, replay, provenance, and local NameCert rows before treating a codon-level lift orbit as classified.

Proof. Read the source theorem through its finite components. The support normal forms and lift patterns supply A and E, while support automorphisms and the global complement action supply Γ and T.

Apply the Burnside formula. The row B records exactly the fixed-point count needed to classify the displayed support-lift pair, and the row K names the resulting orbit type among the twenty displayed rows.

Finally apply the local certificate obligations. Theorem 4059.2 keeps transport, replay, provenance, and local naming attached to the same finite enumeration. Thus a downstream read cannot consume an orbit class without the displayed Burnside and classifier evidence. □

Theorem 3079.4 (Codon window and Burnside dependency)**.** *Every accepted* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *read of an orbit class first consumes the codon-window carrier of chapter 2234 and the finite Burnside count route of chapter 3578. Only after those sibling surfaces have supplied the source window and orbit-count ledger may the rows $A, E, \Gamma, T, B, K, H, C, P, N$ export a classifier value.*

Proof. Project the carrier of Definition 4059.1. The support row A and binary lift row E are not free coordinates; they are read as codon-window data before the automorphism row Γ and complement row T act on them.

Next project the Burnside ledger row B. The classifier row K is downstream of the fixed-point counts recorded in B, so its value cannot be read before the finite orbit-count route supplied by chapter 3578.

Finally apply the NameCert obligations of Definition 4059.2. They keep H, C, P, N attached to the same source-window and Burnside-ledger rows, and therefore prevent the classifier from exporting an orbit value detached from chapter 2234 or chapter 3578. □

Falsifiable prediction: Deleting the Burnside ledger row B prevents an accepted $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet from exporting any codon-level lift-orbit class, even when support, lift, automorphism, complement, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: codon-level lift-orbit classification: $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ binds support normal forms, binary lift vectors, support automorphisms, global complement action, Burnside fixed-point counts, classifier readback, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite orbit-enumeration packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet is generated from support normal forms, binary lift vectors, support automorphism rows, global complement rows, Burnside fixed-point ledgers, classifier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4059.1, Definition 4059.2, and Definition 4059.3 seed the finite codon lift-orbit classifier over support normal forms, lift vectors, group-action rows, Burnside counts, and classifier readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a biological mechanism, probabilistic fitness law, host quotient object, hidden search oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support admission, lift-vector admission, automorphism action, complement action, Burnside ledger exactness, classifier readback, transport, replay, provenance, and local naming for BEDC.Derived.CodonLiftOrbitClassifierUp.

The $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ hub routes its finite cofinal-window surface through chapter 126, chapter 291, chapter 301, chapter 38, and the downstream separability consumer chapter 1809.

Chapter 3080

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ packet records the finite cofinal-window surface needed by regular Cauchy completion arguments. It sits after the stream-window route of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the $\mathsf{CauchyModulus}^{\uparrow}$ schedule row of chapter 292, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a displayed way to cite tail cofinality without quotient streams, selected limits, countable choice, or ambient real completeness.

The cofinal window route consumes the regular Cauchy carrier of chapter 301 and the StreamName schedule of chapter 126 before exporting any Real-facing cofinal read. This dependency keeps the chapter inside the L10 Cauchy lattice: finite stream windows are scheduled first, regular rational readback follows, and only then may the Real-facing seal be consumed by downstream completion chapters.

Definition 3080.1 (Regular Cauchy window cofinality carrier)**.** *A* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (p, S, D, R, M, H, C, P, N)$$

with the following displayed rows:

1. *p is the requested precision row, naming the finite tolerance threshold at which a downstream consumer asks for a tail window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite stream entries read at that precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare displayed entries.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after S and D have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, recording the finite cofinal schedule that sends the requested precision to an admissible tail window.*
6. *H records componentwise* hsame *transport for precision, stream windows, dyadic tolerances, regular rational readback, modulus rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$p, S, D, M \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $p, S, D, R, M, H, C, P, N$. It has no coordinate for a quotient Cauchy stream, selected representative, countable-choice selector, host real equality, or ambient completeness theorem.

Theorem 3080.2 (Regular Cauchy window cofinality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *carrier $W = (p, S, D, R, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinali}}$ *surface consists exactly of precision admission through p, finite stream-window exposure through S, dyadic tolerance comparison through D, regular rational readback through R, cofinal Cauchy-modulus scheduling through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4060.1. The displayed rows are exactly $p, S, D, R, M, H, C, P, N$, so the precision request reaches the packet as finite data.

The stream and tolerance rows are read before the regular rational handoff. Row S supplies the finite $\mathsf{StreamName}^{\uparrow}$ window, row D supplies the dyadic comparison budget, and row M records the cofinal tail schedule requested by the precision row.

The readback row R is therefore downstream of finite windows, dyadic tolerances, and the modulus schedule. Transport, replay, provenance, and local naming preserve that route, so no hidden quotient stream, selected limit, countable-choice selector, host equality, or completeness theorem can be exported. □

Theorem 3080.3 (Regular Cauchy window cofinality sibling route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet must factor its read through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, M \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Here M is sourced from the $\mathsf{CauchyModulus}^{\uparrow}$ *scheduling row of chapter 292. In particular, the packet consumes stream windows from chapter 126, dyadic tolerances from chapter 291, regular rational readback from chapter 301, and the terminal real-name boundary of chapter 38; it cannot bypass those sibling rows by appealing to quotient streams, selected limits, countable choice, or ambient completeness.*

Proof. Start with the precision row p. The cofinal schedule M selects a displayed tail window, and the consumer may inspect only the corresponding finite stream row S and dyadic tolerance row D.

Next use the regular rational readback. The row R is available only after S, D, M have been displayed, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is a finite tail-window read rather than a quotient stream.

Finally attach the Real-facing boundary. The structural rows H, C, P, N preserve the same route, and any downstream $\mathsf{Real}^{\uparrow}$ read receives only the displayed finite cofinal window with its dyadic tolerance and regular rational readback. □

Theorem 3080.4 (Regular Cauchy window cofinality tail directedness)**.** *For two accepted* RegularCauchyWindowCo *packets over the same precision route, the displayed cofinal modulus schedules admit a later finite tail window whose* $\mathsf{StreamName}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal factor through both original windows. The common-window read remains inside the displayed rows of the two packets and does not use quotient stream equality, selected limits, countable choice, or ambient completeness.*

Proof. Project both carriers by Definition 4060.1. Each packet exposes the same precision route before its finite stream window S, dyadic tolerance row D, regular rational readback row R, and cofinal schedule row M can be consumed.

Use the two displayed cofinal schedule rows. Applying M in the first packet and M in the second packet gives two finite tail requests; choose a displayed later unary threshold that is later than both requests in the common precision route.

At that later threshold, the stream window, dyadic tolerance, regular readback, and Real-facing seal are read through the same finite tail. By Definition 4060.3, the consumer must still factor through

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, M \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The construction uses only the carrier rows and the later finite threshold. It introduces no quotient stream equality, selected limit, countable-choice selector, or ambient completeness principle. □

Definition 3080.5 (Regular Cauchy window cofinality forward lattice)**.** *The forward lattice of an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet is the ordered dependency route that exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *window row of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row of chapter 126, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any density or separability consumer cites the* $\mathsf{RealSeparability}^{\uparrow}$ *route of chapter 1809. It has no coordinate for a quotient stream, selected dense subsequence, countable-choice enumeration, or ambient real-completeness principle.*

Theorem 3080.6 (Regular Cauchy window cofinality forward lattice)**.** *Every density or separability route consuming an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet must expose the forward lattice of Definition 4060.5 before citing* $\mathsf{RealSeparability}^{\uparrow}$*. In particular, the consumer first reads the finite* $\mathsf{StreamName}^{\uparrow}$ *observations, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *window readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal used by the Real separability route.*

Proof. Project the accepted carrier by Definition 4060.1. The only route to a Real-facing read is through the displayed stream row S, dyadic tolerance row D, regular rational readback row R, and cofinal schedule row M.

Use Definition 4060.3. It forces every Real or Completion consumer to factor through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ boundary is visible.

Now specialize the consumer to a density or separability route. The cited $\mathsf{RealSeparability}^{\uparrow}$ surface of chapter 1809 may therefore read only the displayed forward lattice, not a selected dense sequence, quotient stream equality, countable-choice enumeration, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ read displays one precision row, one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one Cauchy modulus schedule row, and local transport, replay, provenance, and naming rows before it is consumed by a Real or Completion route.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ is not bijective to any listed sibling because it binds a precision request, finite stream window, dyadic tolerance ledger, regular rational readback, cofinal modulus schedule, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinality}^{\uparrow}}$ **.**

Constructive story: A RegularCauchyWindowCofinality$^{\uparrow}$ packet is generated from a precision request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Cauchy modulus schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyWindowCofinality}^{\uparrow}}$, obligation)

Scope closed: Definition 4060.1, Definition 4060.2, Definition 4060.3, and Definition 4060.4 enumerate the precision, stream-window, dyadic tolerance, RegSeqRat readback, cofinal-modulus, transport, replay, provenance, and local naming obligations.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyWindowCofinality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked precision admission, stream-window exposure, dyadic tolerance comparison, RegSeqRat readback, cofinal modulus scheduling, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindowCofinalityUp.

The RegularCauchyWindowCofinality$^{\uparrow}$ route is a sibling dependency surface for finite Real and Completion readback, not an isolated Cauchy primitive.

The cofinal window is consumed as the finite RegSeqRat$^{\uparrow}$ readback scheduled through StreamName$^{\uparrow}$ windows: chapter 301 supplies the regular rational handoff, while chapter 126 supplies the inspected finite windows that make the cofinal tail visible to downstream consumers.

The same cofinal tail window also consumes the CauchyModulus$^{\uparrow}$ schedule row of chapter 292 before RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers may read the window. This is a substantive dependency on the displayed modulus schedule, not a host choice of a tail index.

Theorem 3080.7 (Regular cofinal windows feed separability consumers)**.** *Any* RealSeparability$^{\uparrow}$ *density consumer citing the* RegularCauchyWindowCofinality$^{\uparrow}$ *route of chapter 4060 must first read the* StreamName$^{\uparrow}$ *finite window, the* DyadicRatCore$^{\uparrow}$ *tolerance row, the* RegSeqRat$^{\uparrow}$ *readback, and the* CauchyModulus$^{\uparrow}$ *schedule. Thus the downstream density edge of chapter 1809 consumes a displayed cofinal window only after the finite source window, dyadic tolerance, regular rational readback, and modulus schedule have all been exposed; it exports no quotient stream, selected limit, countable-choice schedule, or ambient completeness assertion.*

Completion-functor, RealSeparability$^{\uparrow}$, and regular-Cauchy density consumers may cite the RegularCauchyWindowCofinality$^{\uparrow}$ route only after reading the StreamName$^{\uparrow}$ finite window of chapter 126, the DyadicRatCore$^{\uparrow}$ tolerance row of chapter 291, the RegSeqRat$^{\uparrow}$ readback of chapter 301, and the CauchyModulus$^{\uparrow}$ schedule row of chapter 292. The completion-facing edge is therefore a finite lattice dependency, not a host choice of a cofinal tail or an ambient compactness assertion. The RegularCauchyWindowCofinality$^{\uparrow}$ packet supplies the finite window-cofinality route consumed by chapter 301, chapter 126, and chapter 38 before any Real-completion consumer reads a sealed name. The RegularCauchyWindowCofinality$^{\uparrow}$ completion anchor reads the same existing siblings before any sealed real-facing consumer can cite the cofinal window route. It consumes RegSeqRat$^{\uparrow}$ windows from chapter 301, StreamName$^{\uparrow}$ schedules from chapter 126, and the terminal Real$^{\uparrow}$ completion seal from chapter 38, so the hub dependency edge remains a displayed finite route rather than a quotient stream, selected limit, countable-choice selector, or ambient completeness claim. The RealSeparability$^{\uparrow}$ density reads consume RegularCauchyWindowCofinality$^{\uparrow}$ only after StreamName$^{\uparrow}$ finite windows, DyadicRatCore$^{\uparrow}$ tolerances, RegSeqRat$^{\uparrow}$ readback, the CauchyModulus$^{\uparrow}$ schedule, and the Real$^{\uparrow}$ seal are visible. Thus the dependency edge into chapter 1809 remains a finite displayed route rather than a quotient stream, selected dense sequence, countable-choice enumeration, host real equality, or ambient completeness claim. The MetaCICDiamondResidualBudget$^{\uparrow}$ packet is read beside the local diamond witness of chapter 3595, the residual envelope of chapter 3921, the substitution boundary of chapter 3791, and the closed parallel diamond premise ledger of chapter 2325.

Chapter 3081

A Concrete Naming Certificate for MetaCICDiamondResidualBudget$^{\uparrow}$

The MetaCICDiamondResidualBudget$^{\uparrow}$ packet realizes the finite residual-budget boundary of chapter 9378. It sits beside the local diamond witness of chapter 3595, the residual envelope of chapter 3921, the substitution boundary of chapter 3791, and the closed parallel diamond premise ledger of chapter 2325. The packet names the finite rows still required by the conditional confluence route without claiming the full parallel diamond theorem.

Definition 3081.1 (MetaCIC diamond residual budget carrier)**.** *A* MetaCICDiamondResidualBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, F, R, J, S, E, L, D, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row for the source and target terms whose residual branches are audited.*
2. *F is the redex-frontier row that lists the beta redexes and congruence faces inspected by the local diamond route.*
3. *R is the residual-budget row, recording the finite fuel and residual-address accounting needed to replay candidate joins.*
4. *J is the candidate-join row, naming the displayed joins produced from F and R.*
5. *S is the substitution-boundary row, keeping the residual-substitution compatibility premise visible.*
6. *E is the residual-envelope row, read through chapter 3921.*
7. *L is the closed parallel diamond premise ledger row, read through chapter 2325.*
8. *D is the local diamond witness row, read through chapter 3595 as finite local geometry rather than a proof of the closed endpoint.*
9. *H records componentwise* hsame *transport for typed windows, frontier rows, residual budgets, candidate joins, substitution boundary, residual envelope, premise ledger, local diamond witness, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay of the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* MetaCICDiamondResidualBudget$^{\uparrow}$*, MetaCIC beta reduction,* BetaParallel*,* BetaStarStep*, the named sibling packets,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* NameCert$_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate proving BetaParallelSubstitutionCompatibility*, no coordinate proving full parallel diamond, no coordinate proving arbitrary closed Church–Rosser, and no host substitution semantics.*

Theorem 3081.2 (MetaCIC diamond residual budget NameCert obligations)**.** *For every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *carrier $B = (T, F, R, J, S, E, L, D, H, C, P, N)$, the local* NameCert$_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$ *surface consists exactly of typed-window admission through T, redex-frontier exposure through F, residual-budget accounting through R, candidate-join display through J, substitution-boundary visibility through S, residual-envelope handoff through E, closed-premise-ledger handoff through L, local-diamond witness handoff through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4061.1. The displayed rows are exactly $T, F, R, J, S, E, L, D, H, C, P, N$, so carrier admission is a finite projection.

The residual budget is read before any candidate join is exported. Rows T and F name the typed window and redex frontier, while R records the finite residual-address and fuel accounting required by the join row J.

The substitution row S remains visible downstream. The residual envelope E, premise ledger L, and local diamond witness D are sibling handoffs, not discharges of S or proofs of the full diamond endpoint.

Finally H, C, P, N transport, replay, source, and name the same rows. With no coordinate for the refused endpoint claims, the listed rows exhaust the local NameCert surface. □

Theorem 3081.3 (MetaCIC diamond residual budget endpoint boundary)**.** *Any confluence consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may cite only the displayed residual-budget route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

It may not promote the candidate-join row or residual-envelope row into BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Start with the displayed route. The carrier gives the consumer typed windows, redex-frontier data, residual-budget accounting, a substitution-boundary row, candidate joins, the residual envelope, the closed-premise ledger, and the local diamond witness.

Use Definition 4061.2. Its substitution-boundary visibility keeps S attached whenever J is exported, and its sibling rows keep E, L, D as contextual handoffs rather than endpoint proofs.

Transport, replay, provenance, and local naming preserve the same finite route. Therefore the packet can expose residual-budget coverage and candidate joins, but it cannot supply a premise-free parallel diamond theorem or arbitrary closed Church–Rosser theorem. □

Theorem 3081.4 (MetaCIC diamond residual budget fuel monotonicity)**.** *Suppose an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet is extended only by enlarging the finite residual-budget row R, while preserving the typed window T, redex frontier F, substitution boundary S, candidate join row J, residual envelope E, premise ledger L, local diamond row D, transport H, replay*

C, provenance P, and local naming N. Then every confluence consumer remains on the same endpoint-boundary route

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N,$$

and the extended packet still cannot infer BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Project the accepted carrier of Definition 4061.1. The only coordinate changed by the comparison is the finite residual-budget row R; all other displayed rows are preserved.

The candidate join row J remains attached to the same typed window, redex frontier, and visible substitution boundary. Enlarging R supplies more displayed fuel for the same finite accounting route, but it does not add a new endpoint row.

Apply Definition 4061.3. The consumer therefore still reads through $T, F, R, S, J, E, L, D, H, P, N$ and cannot promote the budget extension into the refused endpoint claims. □

Theorem 3081.5 (MetaCIC residual-budget fuel envelope)**.** *An accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may enlarge finite residual-budget accounting only through a fuel envelope that keeps the typed window T, redex frontier F, substitution boundary S, candidate joins J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N visible. The enlarged envelope still exports only the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N,$$

and it does not assert BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Project the accepted carrier. A fuel-envelope enlargement changes only the accounting boundary of the residual-budget row R, while the typed window, redex frontier, substitution boundary, candidate joins, residual envelope, premise ledger, local diamond row, and structural rows remain displayed.

Use fuel monotonicity. The larger budget supplies more finite accounting for the same route, but it does not introduce a new endpoint theorem or hide the substitution boundary behind the candidate join.

Apply the endpoint-boundary theorem. The confluence-facing consumer therefore remains confined to $T, F, R, S, J, E, L, D, H, C, P, N$, so the refused substitution-compatibility, parallel-diamond, and Church–Rosser conclusions are still outside the packet. □

Theorem 3081.6 (MetaCIC residual-budget frontier accounting)**.** *For every accepted* MetaCICDiamondResidualBu *carrier $B = (T, F, R, J, S, E, L, D, H, C, P, N)$, the typed-window row T, redex-frontier row F, residual-budget row R, and substitution-boundary row S must be displayed before J is a valid candidate-join row. A consumer that reads J without the prior T, F, R, S accounting is outside the accepted residual-budget carrier.*

Proof. Project the carrier of Definition 4061.1. The candidate-join coordinate J is not a primitive endpoint theorem; it is listed after typed windows, redex frontiers, residual budgets, and the visible substitution boundary.

Apply Definition 4061.3. The displayed route places T, F, R, S before J, and the endpoint boundary prevents the join row from being read as a premise-free confluence theorem.

Thus J is valid only as a candidate-join row mediated by the frontier and budget accounting rows already displayed in the carrier. □

Theorem 3081.7 (MetaCIC residual-budget substitution visibility)**.** *Every accepted candidate-join read through J keeps the substitution-boundary row S visible along the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

Therefore a residual-budget read cannot be promoted to BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Use Definition 4061.3. Its displayed route places typed windows T, redex frontiers F, residual budgets R, and the substitution boundary S before any candidate join J is exported.

Apply Definition 4061.6. It states that a J read without prior T, F, R, S accounting lies outside the accepted residual-budget carrier.

The downstream rows E, L, D, H, P, N are sibling handoffs and structural rows on the same route. They do not discharge S or add an endpoint proof, so the accepted read remains a finite residual-budget boundary rather than a promoted confluence theorem. □

Theorem 3081.8 (MetaCIC residual-budget obligation rows)**.** *Every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *consumer must display the typed-window row T, redex-frontier row F, residual-budget row R, and substitution-boundary row S before the candidate-join row J is read. After J is read, the same consumer must retain the residual-envelope row E, closed-premise-ledger row L, and local-diamond row D before any confluence handoff is exposed. The route remains*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

and does not claim BetaParallelSubstitutionCompatibility*, full parallel diamond, evaluator equality, subject reduction, or kernel consistency.*

Proof. Start with frontier accounting, Definition 4061.6. It forces T, F, R, S to be visible before the candidate-join row J is a valid read.

Next use substitution visibility, Definition 4061.7. The substitution boundary S stays attached along the displayed route, so J cannot be detached into a substitution-compatibility theorem.

Apply endpoint boundary, Definition 4061.3. It keeps the downstream confluence-facing handoff inside E, L, D, H, P, N, while candidate-join visibility supplies the same sibling rows after J.

Thus the accepted consumer must expose T, F, R, S before J and E, L, D before the confluence handoff. The refused endpoint claims are outside the carrier coordinates. □

Theorem 3081.9 (MetaCIC residual-budget critical-pair route)**.** *Every candidate join exposed by the row J of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *carrier is indexed by the displayed redex-frontier and critical-pair row F, is budgeted by R, and remains conditional on the substitution-boundary row S. A consumer may therefore inspect a candidate join only through the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N,$$

not as a standalone parallel-diamond, subject-reduction, evaluator-equality, or kernel-consistency result.

Proof. Project the carrier in Definition 4061.1. The row J is listed only after the typed window T, the redex-frontier row F, the residual-budget row R, and the substitution-boundary row S.

Use the obligation rows of Definition 4061.8. They require every accepted consumer to expose T, F, R, S before J, so the candidate join inherits its index from the displayed redex frontier and critical-pair row rather than from an ambient evaluator.

Finally apply substitution visibility and endpoint boundary, Definition 4061.7 and Definition 4061.3. These keep S attached and route the downstream read through E, L, D, H, C, P, N, excluding any standalone endpoint theorem. □

Theorem 3081.10 (MetaCIC residual-budget substitution preservation)**.** *Suppose an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is extended only in the residual-budget row R, while preserving the typed window T, redex frontier F, substitution boundary S, candidate joins J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N. Then every accepted candidate-join read still keeps S visible along the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

The preservation claim does not assert BetaParallelSubstitutionCompatibility*, full parallel diamond, or closed Church–Rosser.*

Proof. Project the carrier of Definition 4061.1. The comparison changes only the residual-budget coordinate R; the typed window, redex frontier, substitution boundary, candidate joins, sibling handoffs, and structural rows are preserved.

Use Definition 4061.6. A candidate-join read remains accepted only when T, F, R, S are displayed before J, so extending R cannot detach J from the visible substitution boundary.

Apply Definition 4061.7. The substitution row S remains visible through every accepted candidate-join read, and Definition 4061.3 keeps the downstream route confined to E, L, D, H, P, N.

Thus the preserved rows carry the same visible substitution boundary under a larger residual budget. The packet still lacks any endpoint coordinate for BetaParallelSubstitutionCompatibility, full parallel diamond, or closed Church–Rosser. □

Theorem 3081.11 (MetaCIC residual-budget candidate-join visibility)**.** *Every consumer of the candidate-join row J in an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet must also expose the residual-envelope handoff E, closed-premise-ledger handoff L, local-diamond handoff D, and the structural rows H, C, P, N. The candidate join is therefore consumed only through the displayed sibling handoffs and local NameCert structure.*

Proof. Project the same carrier. The rows following J are precisely the sibling handoffs E, L, D and the structural rows H, C, P, N.

Use Definition 4061.3. Its route continues from J through E, L, D and then through the transport, replay, provenance, and naming rows.

Therefore a consumer of J must keep the residual envelope, closed premise ledger, local diamond witness, and local structural surface visible. Nothing in the carrier permits a bare candidate-join read detached from those rows. □

Theorem 3081.12 (MetaCIC residual-budget endpoint non-escape)**.** *No accepted* MetaCICDiamondResidualBudget *packet proves* BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser. The packet exposes only typed windows, redex frontiers, residual budgets, candidate joins, substitution-boundary visibility, sibling handoffs, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 4061.1. Its coordinates are $T, F, R, J, S, E, L, D, H, C, P, N$, and none of them is an endpoint proof row for substitution compatibility, full parallel diamond, or closed Church–Rosser.

Apply Definition 4061.3. The endpoint boundary states that confluence consumers remain on the displayed residual-budget route and may not promote candidate joins or residual envelopes into the refused endpoint claims.

Transport, replay, provenance, and naming preserve the same finite packet. They do not add endpoint proof data, so the three endpoint claims remain outside the accepted certificate. □

Theorem 3081.13 (MetaCIC residual-budget local join)**.** *If an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet exposes typed window T, redex frontier F, residual budget R, substitution row S, candidate join J, residual envelope E, premise ledger L, and local diamond row D, then any local join consumed through J is valid only under the displayed residual fuel and visible substitution boundary. The packet cannot be read as* BetaParallelSubstitutionCompatibility*, full parallel diamond, or full Church–Rosser.*

Proof. Project the carrier of Definition 4061.1. The candidate join row J is displayed only after the typed window T, redex frontier F, residual budget R, and substitution-boundary row S.

Apply Definition 4061.6. It keeps the redex-frontier accounting and residual fuel visible before J can be read by a confluence-facing consumer.

Apply Definition 4061.7. It keeps S attached along the route from T, F, R, S through J, so the join is not a substitution-compatibility theorem detached from its boundary.

Finally use Definition 4061.12. The residual envelope E, premise ledger L, and local diamond row D remain sibling handoffs for a bounded candidate route, not endpoint proofs of BetaParallelSubstitutionCompatibility, full parallel diamond, or Church–Rosser. □

Theorem 3081.14 (MetaCIC residual-budget candidate-mediated SN handoff)**.** *Every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet consumed by a bounded candidate-mediated strong-normalization reader must expose typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N before the SN reader may consume the candidate route. The packet does not assert global strong normalization, global confluence, or substitution compatibility.*

Proof. Begin with the frontier accounting theorem, Definition 4061.6. It requires T, F, R, S to be displayed before J is a valid candidate-join row, so a candidate-mediated SN reader cannot consume J as a free-standing normalization witness.

Apply substitution visibility, Definition 4061.7. The substitution boundary S remains attached along the route from the typed window and frontier rows through the candidate join, so the handoff is not a proof of substitution compatibility.

Use the local-join theorem, Definition 4061.13. It keeps every local join under the displayed residual fuel and visible substitution boundary while routing the consumer through the residual envelope E, premise ledger L, and local diamond row D.

Finally apply endpoint non-escape, Definition 4061.12. The structural rows H, C, P, N transport, replay, source, and name the same finite route, and no carrier coordinate supplies global strong normalization, global confluence, or BetaParallelSubstitutionCompatibility. □

Theorem 3081.15 (MetaCIC residual-budget candidate-window exhaustion)**.** *Every candidate-mediated strong-normalization consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet must exhaust the displayed finite typed window T, redex frontier F, residual budget R, and substitution boundary S before reading the candidate join J. The consumer must then keep the residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N attached to that same finite route. This exhaustion theorem does not infer global strong normalization, substitution compatibility, full parallel diamond, or Church–Rosser.*

Proof. Begin with frontier accounting, Definition 4061.6. It makes the typed window, redex frontier, and residual budget visible before any candidate join can be consumed.

Next use substitution visibility, Definition 4061.7. The boundary row S remains attached to the candidate route, so exhaustion cannot be replaced by a global substitution-compatibility claim.

Apply the local-join theorem, Definition 4061.13. It keeps the candidate join J under the displayed residual fuel and routes the consumer through E, L, D rather than through a full parallel diamond theorem.

Finally use endpoint non-escape, Definition 4061.12, together with Definition 4061.14. The structural rows H, C, P, N preserve and name the same finite route, so the SN consumer exhausts the candidate window without obtaining global strong normalization, Church–Rosser, or unbounded confluence. □

Theorem 3081.16 (MetaCIC residual-budget candidate-mediated exhaustion)**.** *Every candidate-mediated strong-normalization consumer of an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is exhausted by the typed window T, redex frontier F, residual budget R, visible substitution row S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural H, C, P, N rows. The consumer may read the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

and no other endpoint data. In particular, the packet refuses global strong normalization, BetaParallelSubstitutionCo *full parallel diamond, and Church–Rosser.*

Proof. Begin with frontier accounting, Definition 4061.6. It requires the typed window T, redex frontier F, residual budget R, and substitution boundary S before any accepted consumer can read the candidate join J.

Next use substitution visibility and candidate-join visibility, Definition 4061.7 and Definition 4061.11. These keep S attached to the route and force E, L, D, H, C, P, N to remain visible after J.

Apply endpoint non-escape, Definition 4061.12. The residual envelope, premise ledger, and local diamond row are sibling handoffs inside the bounded packet, not proofs of substitution compatibility, a full parallel diamond theorem, or arbitrary closed Church–Rosser.

Finally compose with the candidate-mediated SN handoff and candidate-window exhaustion, Definition 4061.14 and Definition 4061.15. The structural rows transport, replay, source, and name the same finite route, so a strong-normalization consumer exhausts exactly $T, F, R, S, J, E, L, D, H, C, P, N$ and receives no global strong-normalization principle. □

Theorem 3081.17 (MetaCIC residual-budget premise alignment)**.** *Every accepted* MetaCICDiamondResidualBudge *packet consumed by a candidate-mediated strong-normalization reader must expose the residual envelope E, closed premise ledger L, and local diamond row D together, after typed window T, redex frontier F, residual budget R, substitution boundary S, and candidate join J, and before the structural rows H, C, P, N. Dropping any one of E, L, D leaves only a frontier packet, not a valid confluence-facing residual-budget premise.*

Proof. Start with candidate-join visibility, Definition 4061.11. It places the accepted candidate route after the typed window, redex frontier, residual budget, and substitution boundary, so a consumer cannot read J before the displayed frontier rows.

Next apply the local-join theorem, Definition 4061.13. The local join remains under the residual fuel and visible substitution boundary while routing through the residual envelope E, premise ledger L, and local diamond row D.

Finally use the candidate-mediated SN handoff, Definition 4061.14. It requires E, L, D before the structural transport, replay, provenance, and naming rows are consumed by the SN reader. If

any one of E, L, D is absent, the reader has only the frontier and candidate route, so the packet has not yet supplied the confluence-facing residual-budget premise. □

Theorem 3081.18 (MetaCIC residual-budget premise-candidate coverage)**.** *Every accepted candidate join consumed by a candidate-mediated strong-normalization or confluence reader of a* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is covered exactly by the displayed route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The typed window T, redex frontier F, residual budget R, and substitution boundary S are displayed before J; the residual envelope E, closed premise ledger L, and local diamond row D are displayed after J; and the structural rows H, C, P, N preserve the same finite packet. The coverage claim refuses $\mathsf{BetaParallelSubstitutionCompatibility}$*, full parallel diamond, and global Church–Rosser.*

Proof. Begin with frontier accounting and substitution visibility, Definition 4061.6 and Definition 4061.7. Together they place T, F, R, S before any accepted candidate join J and keep the substitution boundary visible while the join is read.

Use candidate-join visibility and the local-join theorem, Definition 4061.11 and Definition 4061.13. These force the residual envelope E, closed premise ledger L, and local diamond row D to remain attached after J, and they keep the local join under the displayed residual fuel.

Apply the candidate-mediated SN handoff and premise alignment, Definition 4061.14 and Definition 4061.17. A strong-normalization or confluence reader therefore consumes exactly the same finite route before the structural rows are consulted.

Finally use endpoint non-escape, Definition 4061.12. The rows H, C, P, N transport, replay, source, and name the displayed packet, but they do not add endpoint proof data; hence the candidate coverage cannot be read as substitution compatibility, full parallel diamond, or global Church–Rosser. □

Theorem 3081.19 (MetaCIC critical-pair residual normalization)**.** *Any accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet whose premise-candidate coverage row exposes a local critical pair normalizes that pair only through the same residual envelope, candidate window, substitution boundary, fuel envelope, and closed parallel-diamond premise ledger before it can hand off to the candidate-mediated strong-normalization route. The normalization read is concrete to the carrier rows*

$$T, F, R, S, J, E, L, D, H, C, P, N$$

and introduces no abstract source-equivalence transport, global strong normalization, full parallel diamond, or substitution-compatibility theorem.

Proof. Start with the critical-pair route, Definition 4061.9. It places the local critical pair inside the typed window, redex frontier, residual budget, and local diamond rows rather than in an external source-equivalence relation.

Apply substitution preservation and candidate-window exhaustion, Definition 4061.10 and Definition 4061.15. These keep the visible substitution boundary S and the finite candidate window T, F, R, S, J attached while the critical pair is normalized.

Use premise-candidate coverage, Definition 4061.18. The residual envelope E, closed premise ledger L, and local diamond row D are displayed after J and before the structural rows, so the critical-pair read remains inside the same residual-budget packet.

Finally apply the candidate-mediated SN handoff, Definition 4061.14. The rows H, C, P, N transport, replay, source, and name the normalized critical-pair route, but no row supplies global strong normalization, a full parallel diamond endpoint, or an abstract source-equivalence transport. □

Theorem 3081.20 (MetaCIC residual-budget obstruction ledger)**.** *Every accepted* MetaCICDiamondResidualBudge *packet retains the obstruction ledger carried by the substitution-boundary row S and the endpoint-boundary rows E, L, D along the full route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The retained ledger refuses full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Project the carrier of Definition 4061.1. The rows $T, F, R, J, S, E, L, D, H, C, P, N$ are all displayed, and the substitution boundary S is not absorbed by the candidate join row.

Apply substitution visibility and endpoint non-escape, Definition 4061.7 and Definition 4061.12. These keep S visible and keep E, L, D as sibling handoffs rather than endpoint proofs.

Finally use premise-candidate coverage, Definition 4061.18. The structural rows H, C, P, N preserve the same finite route, so the refused endpoint claims remain recorded as obstructions rather than exported conclusions. □

Theorem 3081.21 (MetaCIC diamond residual budget obligation witness cut)**.** *Every confluence-facing consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet must display the typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, closed premise ledger L, and local diamond row D before reading structural transport, replay, provenance, or local name rows H, C, P, N. Without those displayed rows the packet supplies only the obstruction ledger of Definition 4061.20; it does not supply substitution compatibility, full parallel diamond, Church–Rosser, or global strong normalization.*

Proof. Start with frontier accounting, Definition 4061.6. It fixes the typed window, redex frontier, and residual budget as visible finite rows before any confluence-facing consumer inspects a candidate.

Apply substitution visibility and candidate-join visibility, Definition 4061.7 and Definition 4061.11. These place the substitution boundary S before the accepted join J and prevent the join from hiding the boundary row.

Use the local-join theorem and endpoint non-escape, Definition 4061.13 and Definition 4061.12. The residual envelope E, closed premise ledger L, and local diamond row D remain displayed after J and before the structural rows are read.

Finally apply the obstruction ledger theorem. Since H, C, P, N only transport, replay, source, and name the same finite packet, a consumer missing T, F, R, S, J, E, L, D has obstruction evidence rather than substitution compatibility, full parallel diamond, Church–Rosser, or global strong normalization. □

Theorem 3081.22 (MetaCIC residual-budget normalization handoff)**.** *Any normalization-facing consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may consume the residual-normalization read only after the displayed rows*

$$T, F, R, S, J, E, L, D, H, C, P, N$$

are visible. The handoff is a bounded candidate route through the substitution-boundary and obstruction rows; it does not assert full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with critical-pair residual normalization, Definition 4061.19. It places the normalization read inside the residual envelope, candidate window, substitution boundary, fuel envelope, and closed premise ledger.

Use the obstruction ledger theorem. The rows S, E, L, D remain visible when the normalization-facing consumer reads J, so the handoff cannot hide the substitution boundary or erase the retained endpoint obstructions.

The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. Since no refused endpoint is a carrier coordinate, the normalization handoff remains bounded and conditional. □

Theorem 3081.23 (MetaCIC residual-budget confluence refusal)**.** *No accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet promotes its candidate join row J, residual envelope E, premise ledger L, or local diamond row D into full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency. Every confluence-facing read remains confined to*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

Proof. Begin with the endpoint boundary, Definition 4061.3. It already prevents a confluence consumer from promoting candidate joins or residual envelopes into endpoint theorems.

Apply the obstruction ledger theorem. The substitution-boundary row and obstruction rows are retained with the confluence-facing read, so J, E, L, D remain bounded handoffs.

Finally use normalization handoff. Even when a normalization-facing consumer is present, the read still passes through $T, F, R, S, J, E, L, D, H, C, P, N$ and no carrier coordinate supplies subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3081.24 (MetaCIC residual-budget closed-premise discharge route)**.** *Any accepted closed-premise discharge read through a* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet must expose the typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N. It may discharge only the displayed finite premise window*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The route exports no full subject reduction, global confluence, Church–Rosser theorem, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with frontier accounting and substitution visibility, Definition 4061.6 and Definition 4061.7. They require the typed window, redex frontier, residual budget, and substitution boundary to be displayed before the candidate join row can be read.

Use the local-join theorem and candidate-window exhaustion next. Definition 4061.13 keeps J under the displayed residual fuel and visible substitution boundary, while Definition 4061.15 exhausts the finite candidate window before any candidate-mediated consumer may continue.

Apply premise-candidate coverage and confluence refusal, Definition 4061.18 and Definition 4061.23. These force the residual envelope, premise ledger, local diamond row, and structural rows to stay attached to the same finite route, and they block promotion to subject reduction, global confluence, Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3081.25 (MetaCIC residual-budget seed registration)**.** *The seed registration surface for* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *is exactly the finite row family*

$$T, F, R, J, S, E, L, D, H, C, P, N.$$

It registers typed windows, redex frontiers, residual budgets, candidate joins, the substitution-boundary row, obstruction handoffs, transport, replay, provenance, and local naming, while refusing full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Project the carrier and the NameCert obligations, Definition 4061.1 and Definition 4061.2. Together they list precisely the rows $T, F, R, J, S, E, L, D, H, C, P, N$.

Use obstruction ledger and confluence refusal. The registered rows retain the substitution-boundary and obstruction data while preventing any endpoint theorem from being read out of the seed packet.

Thus the seed registration is the displayed residual-budget family itself. It is suitable for downstream confluence and normalization packets only as a bounded route with explicit refusals. □

Falsifiable prediction: A confluence packet that omits either the residual-budget row R or the candidate-join row J is rejected as a $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ packet, even when it carries local diamond, residual-envelope, or closed-premise-ledger rows.

Independence witness: metacic-local-diamond-witness, metacic-confluence-residual-envelope, metacic-confluence-substitution-boundary, closed-parallel-diamond-premise-ledger: MetaCICDiamondResidualBudg is not bijective to any listed sibling because it binds typed windows, redex frontiers, residual-budget accounting, candidate joins, a visible substitution boundary, residual envelope, premise ledger, local diamond witness, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite confluence-facing packet.

The Lean-side hardening surface requires Nontrivial and FieldFaithful instances for the residual-budget packet. The nontrivial witness separates a carrier with displayed residual-budget and candidate-join rows from a refused packet missing one of those rows; field faithfulness matches on $T, F, R, J, S, E, L, D, H, C, P, N$ so two accepted packets with the same displayed fields are indistinguishable to the NameCert classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ packet is generated from typed windows, redex frontiers, residual-budget accounting, candidate joins, a visible substitution-boundary row, residual-envelope and closed-premise-ledger handoffs, a local diamond witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4061.1, Definition 4061.2, Definition 4061.6, and Definition 4061.12 close the finite residual-budget obligation surface for the MetaCIC confluence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close BetaParallelSubstitutionCompatibility, full parallel diamond, arbitrary closed Church–Rosser, host substitution semantics, Lean verification, or bridge checking.

Upgrade path: Lean verification requires checked typed-window admission, redex-frontier exposure, residual-budget accounting, candidate-join display, substitution-boundary visibility, sibling handoffs, endpoint nonescape, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICDiamondResidualBudgetUp.

Theorem 3081.26 (MetaCIC diamond residual budget carrier surface)**.** *Every accepted residual-budget read for* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *factors through the local diamond witness, residual envelope, substitution boundary, closed parallel diamond premise ledger, finite residual counter, typed-window row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and*

local NameCert *before any candidate-mediated confluence consumer may inspect it. In carrier notation, every such read must expose*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

with R acting as the finite residual counter and T acting as the typed-window row.

Proof. Project the carrier and NameCert obligations of Definition 4061.1 and Definition 4061.2. They display the typed window, redex frontier, residual-budget counter, candidate join, substitution boundary, residual envelope, closed-premise ledger, local diamond witness, transport, replay, provenance, and local name as the only accepted rows.

Use frontier accounting and substitution visibility. The residual counter R and substitution boundary S are displayed before the candidate join J, and the closed diamond ledger row L appears only downstream of that visible boundary together with the residual envelope E and local diamond witness D.

Finally apply the endpoint-boundary and confluence-refusal theorems. Since the route still passes through H, C, P, N, no consumer can export global confluence, quotient term equality, kernel consistency, or unrestricted strong normalization from the residual-budget packet. □

Chapter 3082

A Concrete Naming Certificate for $\mathsf{WLPOBoundary}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on weak limited omniscience, the $\mathsf{WLPOBoundary}^{\uparrow}$ packet records the Real and Completion boundary at which binary zerohood evidence may be inspected only through finite windows. It sits beside the Specker obstruction of chapter 1406, the LLPO refusal surface of chapter 1978, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Phase Real and Completion consumers a named refusal route: inspected finite binary data may be cited, but no selected limit, quotient stream equality, trichotomy, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness travels across the boundary.

Definition 3082.1 (WLPO boundary carrier)**.** *A* $\mathsf{WLPOBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite binary-window ledger. It records the inspected prefix of a binary stream or real-name zerohood test, not a global all-zero decision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the inspected window and its finite replay budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the binary window and stream schedule have been displayed.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing consumers.*
5. *Q is the refusal row recording that the accepted packet exports finite binary-window evidence but no WLPO decision rule.*
6. *H records componentwise* hsame *transport for the binary window, schedule, readback, Real seal, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion request through B, S, R, E, Q.*
8. *P records* Pkg *provenance over* $\mathsf{WLPOBoundary}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{LLPOBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient stream equality, trichotomy, countable choice, a global WLPO rule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.WLPOBoundaryUp.WLPOBoundaryUp`
Lean4 ✓ `BEDC.Derived.WLPOBoundaryUp.WLPOBoundaryTasteGate single carrier alignment`

Theorem 3082.2 (WLPO boundary NameCert obligations)**.** *For every accepted* $\mathsf{WLPOBoundary}^{\uparrow}$ *carrier* $W = (B, S, R, E, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite binary-window inspection, stream-window replay, regular-rational readback, Real-seal handoff, and refusal nonescape. These obligations are discharged through*

$$B \longmapsto S \longmapsto R \longmapsto E \longmapsto Q$$

together with H, C, P, N*.*

Proof. Project the carrier of Definition 4062.1. The projection exposes a finite binary window, a stream schedule, a regular rational readback, a Real seal, a refusal row, and the structural transport, replay, provenance, and naming rows.

The binary-window obligation reads only B and the displayed replay budget in S. It can certify which finite prefix was inspected, but it cannot decide whether every future binary entry is zero.

The Real-facing route then passes through R and E. These rows expose regular-rational readback and the terminal Real seal only after the finite window has already been named.

Finally the refusal row Q is transported by H and replayed by C with the same provenance P and local name N. Since the packet has no coordinate for a selected limit, quotient stream equality, trichotomy, countable choice, global WLPO, or ambient completeness, no NameCert obligation can export any of those principles. □

Theorem 3082.3 (WLPO boundary Real and Completion refusal)**.** *Every Phase Real or Completion consumer of an accepted* $\mathsf{WLPOBoundary}^{\uparrow}$ *packet receives only the displayed finite binary-window ledger* B*, stream schedule* S*, regular-rational readback* R*, Real seal* E*, refusal row* Q*, componentwise transport* H*, continuation replay* C*, provenance* P*, and local NameCert row* N*. In particular, finite binary-window evidence cannot export a selected limit, quotient stream equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a Real or Completion request. By Definition 4062.1, the request reaches the packet only through the finite route $B, S, R, E, Q, H, C, P, N$.

Use Definition 4062.2. It places binary-window inspection before stream replay, regular-rational readback, Real-seal handoff, and refusal nonescape, so a consumer cannot bypass the finite window and ask for a completed all-zero decision.

The refusal row is the terminal boundary datum. Transport and replay preserve the same finite route, while provenance and local naming add no host-level coordinate. Therefore the accepted packet may witness an inspected prefix, but it cannot produce a selected limit, identify quotient streams, decide trichotomy, invoke countable choice, or complete $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{WLPOBoundary}^{\uparrow}$ Real or Completion read factors through finite binary-window evidence, stream replay, regular-rational readback, the $\mathsf{Real}^{\uparrow}$ seal, and the refusal row before the consumer may cite the boundary.

Independence witness: speckersequence, llpo-boundary, streamname, regseqrat, real: $\mathsf{WLPOBoundary}^{\uparrow}$ is not bijective to any listed sibling because it simultaneously carries a finite binary-window ledger,

stream schedule, regular-rational readback, Real seal, explicit WLPO refusal row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{WLPOBoundary}^{\uparrow}$ packet is generated from finite binary-window evidence, StreamName replay, RegSeqRat readback, a RealUp seal, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4062.1, Definition 4062.2, and Definition 4062.3 seed the finite WLPO boundary for Phase Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim WLPO as a global principle, selected limits, quotient stream equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite binary-window inspection, stream replay, regular-rational readback, Real-seal handoff, and refusal nonescape for BEDC.Derived.WLPOBoundaryUp.

Chapter 3083

A Concrete Naming Certificate for $\mathsf{LocatedClosedBall}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedClosedBall}^{\uparrow}$ packet records a located closed ball by finite center, radius, distance, membership, and locator rows rather than by an ambient closed subset. It sits between the center-radius closed-ball surface of chapter 1451, the endpoint-location discipline of chapter 394, the located-distance route of chapter 1533, and the terminal real-name seal of chapter 38. The packet gives compact-metric, local-support, and completion consumers a displayed closed-ball support row without quotienting points, choosing nearest points, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3083.1 (Located closed-ball carrier)**.** *A* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, c, r, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LocatedMetric}^{\uparrow}$ *source row naming the metric carrier and located distance discipline.*
2. *c is the displayed centre row.*
3. *r is the nonnegative* $\mathsf{Real}^{\uparrow}$ *radius row.*
4. *D is the distance row from the source point to the centre, read through the located metric.*
5. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the non-strict comparison interval $D \leq r$.*
6. *R is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *membership row attached to the same centre, radius, and distance comparison.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the radius and comparison readback.*
8. *H records componentwise* hsame *transport of the metric source, centre, radius, distance, interval, closed-ball membership, and seal rows.*
9. *C records displayed* Cont *replay for distance evaluation, located interval comparison, closed-ball membership, and downstream support handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedClosedBall}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, c, r, D, I, R, E, H, C, P, N$. It has no coordinate for a host subset, quotient point equality, selected nearest point, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedClosedBallUp.TasteGate.LocatedClosedBallTasteGate single carrier alignment`

Theorem 3083.2 (Located closed-ball membership transport)**.** *Let $B = (M, c, r, D, I, R, E, H, C, P, N)$ and $B' = (M', c', r', D', I', R', E', H', C', P', N')$ be accepted* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carriers whose metric source, centre, radius, distance, interval, membership, and seal rows are paired by the displayed* hsame *transports. Then the located closed-ball membership row R transports to R' through the same located-metric distance row and located-interval comparison row, with* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming preserved.*

Proof. Project the carrier of Definition 4063.1. The membership row R is downstream of the located metric source M, centre c, radius r, distance row D, located interval row I, and terminal seal E.

Transport the displayed rows componentwise by H and H'. The centre and radius rows move together with the distance row, and the located interval comparison remains attached to that same transported distance and radius.

Replay the comparison by C and C'. Since P, P' and N, N' source and name only the displayed rows, the transported membership row is still the $\mathsf{MetricClosedBall}^{\uparrow}$ comparison read through $\mathsf{LocatedMetric}^{\uparrow}$ and $\mathsf{LocatedInterval}^{\uparrow}$, not a host subset or quotient point equality. □

Theorem 3083.3 (Located closed-ball nonescape)**.** *Every compact-metric, local-support, completion, or interval-facing consumer of an accepted* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carrier must read the finite route*

$$M, c, r, D, I, R, E, H, C, P, N.$$

The consumer may use the located metric source, displayed centre, radius seal, located distance, located interval comparison, MetricClosedBall membership, transport, replay, provenance, and local naming rows, but it cannot add a host closed subset, quotient point equality, selected nearest point, trichotomy, selected subcover, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4063.1. The accepted packet has exactly the rows $M, c, r, D, I, R, E, H, C, P, N$.

Use Definition 4063.2. It keeps every membership read attached to the located metric source, located interval comparison, closed-ball membership row, and Real seal under transport and replay.

The listed consumers therefore receive a located support row only through the displayed finite route. Since none of the refused host data is a carrier coordinate or a continuation target, the read cannot escape to a host subset, quotient point equality, selected point, selected subcover, trichotomy, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{LocatedClosedBall}^{\uparrow}$ read displays one $\mathsf{LocatedMetric}^{\uparrow}$ source row, one centre row, one $\mathsf{Real}^{\uparrow}$ radius row, one located distance row, one $\mathsf{LocatedInterval}^{\uparrow}$ comparison row, one $\mathsf{MetricClosedBall}^{\uparrow}$ membership row, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: metricclosedball, locatedinterval, locatedmetric, real: $\mathsf{LocatedClosedBall}^{\uparrow}$ is not bijective to any listed sibling because it binds the closed-ball membership row, located interval comparison, and located metric distance row through the same displayed centre-radius support before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}$.

Constructive story: A $\mathsf{LocatedClosedBall}^{\uparrow}$ packet is generated from a LocatedMetric source row, centre row, RealUp radius row, located distance row, LocatedInterval comparison row, MetricClosedBall membership row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4063.1, Definition 4063.2, and Definition 4063.3 seed the located closed-ball support route over MetricClosedBall, LocatedInterval, LocatedMetric, Metric, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host closed subsets, quotient point equality, selected nearest points, selected subcovers, trichotomy, compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-distance exactness, located-interval comparison transport, MetricClosedBall membership transport, Real-seal nonescape, replay, provenance, and local naming for BEDC.Derived.LocatedClosedBallUp.

Chapter 3084

A Concrete Naming Certificate for $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ packet records the theorem-level route from displayed uniform Cauchy data to the regular completion handoff. It consumes the uniform-space carrier of chapter 339, the Cauchy-filter tail surface of chapter 333, the finite-window naming route of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the universal completion row of chapter 1425. The packet names a finite theorem carrier for uniform completion consumers without quotient completion objects, selected representatives, countable choice, host real equality, or ambient completeness.

Definition 3084.1 (Bishop uniform completion theorem carrier)**.** *A* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, F, S, R, E, V, H, C, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{UniformSpace}^{\uparrow}$ *carrier row, recording the displayed source points, entourage requests, refinement reads, and separatedness boundary available to the theorem consumer.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-data row, storing the finite filter-base and tail-stability evidence read inside the same uniform carrier.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, presenting the inspected Cauchy-filter observations as bounded observation windows rather than as an infinite chosen representative.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected windows into regular rational names through the displayed dyadic tolerance route.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording the terminal real-name observations produced from the regular rational readback.*
6. *V is the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *handoff row, naming the extension-and-uniqueness surface available to completion consumers after the source, tail, window, readback, and seal rows have all been read.*
7. *H records componentwise* hsame *transport for the uniform carrier, Cauchy-filter tail data, finite windows, regular rational readback, Real seal, universal handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay through*

$$X, F, S, R, E, V \longmapsto \mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}.$$

9. *P is the* Pkg *provenance row over* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected completion representatives, a countable-choice schedule, host real equality, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.BishopUniformCompletionTheoremUp.BishopUniformCompletionTheoremTasteGate`
`single carrier alignment`

Theorem 3084.2 (Bishop uniform completion theorem NameCert obligations)**.** *For an accepted* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier*

$$U = (X, F, S, R, E, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-space admission for X, Cauchy-filter tail stability for F, finite-window inspection for S, regular rational readback for R, terminal Real sealing for E, regular-completion universal handoff for V, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. No obligation may be discharged by quotient completion, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Carrier admission is projection from Definition 4064.1. The accepted packet displays the uniform carrier, the Cauchy-filter tail data, the finite windows, the regular rational readback, the Real seal, and the universal handoff.

The theorem route begins with X and F. The uniform row fixes the entourage discipline, while the Cauchy-filter row supplies only finite tail evidence inside that discipline. There is no independent coordinate for a quotient completion object or a selected filter representative.

The observation route then passes through S, R, and E. Finite windows are read as regular rational names and then sealed as displayed Real rows, so the certificate never replaces the readback with host real equality, countable choice, or ambient completeness.

Finally V is the only completion-facing exit. The rows H, C, P, N transport, replay, source, and name the same finite route, so every local obligation is accounted for by the displayed rows and no hidden completion principle can enter the packet. □

Theorem 3084.3 (Bishop uniform completion theorem uniform handoff)**.** *Every theorem consumer of an accepted* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUniver}$$

The consumer first reads the uniform carrier, then the Cauchy-filter tail row, then the finite-window observations, then the regular rational readback, then the Real seal, and only then the regular-completion universal handoff.

Lean4 ✓ `BEDC.Derived.BishopUniformCompletionTheoremUp.BishopUniformCompletionTheoremUniformHandoff`

Proof. Begin with the accepted carrier U. The only uniform data available to a theorem consumer is X, and the only Cauchy data is the tail row F read inside that same uniform carrier.

The consumer must next inspect the finite windows S and their regular rational readback R. These rows supply the named observations from which the Real seal E is produced; no selected representative or countable-choice schedule is present in the carrier.

The completion-facing handoff V can be read only after X, F, S, R, E have been opened. Since the remaining rows merely transport, replay, source, and locally name this route, every theorem consumer factors through the stated chain and cannot bypass it by invoking quotient completion, host real equality, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ theorem read factors through one UniformSpace carrier row, one CauchyFilter tail row, one StreamName finite-window row, one RegSeqRat readback row, one Real seal row, one RegularCauchyCompletionUniversal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniformspace, cauchyfilter, streamname, regseqrat, real, regular-cauchy-completion-universal: $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source uniform structure, Cauchy-filter tail evidence, finite windows, regular rational readback, Real seal, universal completion handoff, and the transport/replay/provenance rows binding them into a theorem-level completion certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4064.1, Definition 4064.2, and Definition 4064.3 seed the uniform completion theorem route over a uniform carrier, Cauchy-filter tail row, finite windows, regular rational readback, Real seal, universal handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-space admission, Cauchy-filter tail stability, finite-window inspection, regular rational readback, Real sealing, universal completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformCompletionTheoremUp.

Constructive story: A $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ packet is generated from one UniformSpace carrier row, one CauchyFilter tail-data row, one StreamName finite-window row, one RegSeqRat readback row, one Real seal row, one RegularCauchyCompletionUniversal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3085

A Concrete Naming Certificate for $\mathsf{CantorPairing}^{\uparrow}$

Following Cantor's pairing function, the $\mathsf{CantorPairing}^{\uparrow}$ packet records a finite two-input natural-number indexing surface for schedules that must pass between paired observations and a single displayed countable index. It sits after the natural-number carrier of chapter 28, reads the finite product envelope of chapter 34, and supplies a countable schedule handoff to the finite stream-window discipline of chapter 126. The chapter names only a finite pairing and readback certificate; it does not import host natural-number arithmetic as an unrecorded oracle, quotient equality of pairs, countable choice, or a completed stream.

Definition 3085.1 (Cantor pairing carrier)**.** *A* $\mathsf{CantorPairing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, J, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Nat}^{\uparrow}$ *input rows. They are read as finite unary natural-number names, not as ambient host numerals.*

2. *J is the pairing-code row. It records the displayed triangular code*

$$\pi(X, Y) = \frac{(X + Y)(X + Y + 1)}{2} + Y.$$

   The additions, multiplication, division by the displayed even triangular product, and final successor budget are all ledger rows over the two $\mathsf{Nat}^{\uparrow}$ *inputs.*

3. *L and R are projection rows. They read back the left and right coordinates from the displayed code by replaying the finite triangular search ledger and returning the unique pair of* $\mathsf{Nat}^{\uparrow}$ *rows whose recorded code is J.*

4. *H records componentwise* hsame *transport for the two input rows, the pairing-code row, the two projection rows, continuation replay, provenance, and the local naming row.*

5. *C records displayed* Cont *replay from the two* $\mathsf{Nat}^{\uparrow}$ *inputs through the* $\mathsf{Prod}^{\uparrow}$ *pair envelope, the pairing-code row, the projection readbacks, and the* $\mathsf{StreamName}^{\uparrow}$ *schedule handoff.*

6. *P is the* Pkg *provenance over* $\mathsf{CantorPairing}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

7. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$ *row.*

The carrier has no coordinate for host natural-number pairing, quotient equality of pairs, a selected enumeration, countable choice, or a completed infinite schedule.

Lean4 ✓ `BEDC.Derived.CantorPairingUp.CantorPairingTasteGate single carrier alignment`

Theorem 3085.2 (Cantor pairing NameCert obligations)**.** *For every accepted* $\mathsf{CantorPairing}^\uparrow$ *carrier* $K = (X, Y, J, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* $\mathsf{Nat}^\uparrow$ *inputs, pairing-code row, projection rows, transport row, continuation row, provenance row, and local naming row;*
2. *pairing totality, requiring the displayed triangular computation to produce one finite* $\mathsf{Nat}^\uparrow$ *code row from the two finite input rows;*
3. *projection readback, requiring L and R to recover precisely the two displayed input rows when the same triangular ledger is replayed from J;*
4. *StreamName schedule handoff, requiring every downstream countable schedule consumer to read the paired observation through J, L, R before it treats the result as a single* $\mathsf{StreamName}^\uparrow$ *index;*
5. *ledger refusal, excluding unrecorded host natural-number arithmetic, quotient equality of pairs, selected enumerations, countable choice, and completed streams.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4065.1. The projection exposes exactly two $\mathsf{Nat}^\uparrow$ input rows, one displayed pairing-code row, two projection rows, and the structural rows H, C, P, N.

Pairing totality is a finite replay through C. The two inputs first enter the $\mathsf{Prod}^\uparrow$ envelope, then the triangular ledger forms $X + Y$, the next successor, the displayed even product, the halved triangular number, and the final offset by Y. Every intermediate value is a $\mathsf{Nat}^\uparrow$ ledger row, so the code row J is admitted without reading an ambient host pairing function.

Projection readback uses the same finite ledger in the reverse inspection direction. The projection rows L and R replay the bounded triangular search named by J and return the unique coordinate rows whose recorded code is J. Componentwise transport by H preserves the code and both recovered coordinates under classifier movement.

Finally, a $\mathsf{StreamName}^\uparrow$ consumer reaches the paired observation only through the J, L, R handoff. The provenance row P and local certificate row N name no further coordinate, so the packet cannot export quotient pair equality, a selected enumeration, countable choice, host arithmetic, or a completed stream schedule. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$.

Constructive story: A $\mathsf{CantorPairing}^\uparrow$ packet is generated from two Nat input rows, a displayed triangular pairing-code row, two projection rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CantorPairing}^\uparrow}$, seed)

Scope closed: Definition 4065.1 and Definition 4065.2 seed the finite Cantor-pairing schedule interface between Nat inputs, Product envelopes, and StreamName countable indices.

Formal verification: FormalStatus(NameCert$_{\mathsf{CantorPairing}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host natural-number pairing, quotient equality of pairs, selected enumerations, countable-choice schedules, completed streams, or Lean verification.

Upgrade path: Obligation closure requires carrier admission, pairing totality, projection readback, StreamName handoff, and ledger refusal proofs for the same finite pairing packet.

Chapter 3086

A Concrete Naming Certificate for $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ packet records the separated completion route for directed Cauchy nets as finite BEDC NameCert data. It sits after the Cauchy-net carrier of chapter 589, the Moore–Smith completion interface of chapter 2190, the separated completion boundary of chapter 1612, and the metric-completion handoff of chapter 1506. Its Real-facing seal is read only through the L10 finite-name spine of chapter 126, chapter 301, and chapter 38.

The packet names a completion carrier whose comparison discipline is separated before any terminal Real-facing read is public. It exposes Cauchy-net windows, Moore–Smith boundary data, separated-completion comparison, metric-completion handoff, stream-name observations, regular rational tolerance rows, Real seals, transport, replay, package provenance, and local naming. It exports no quotient net, no selected limit, no host equality of completed points, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3086.1 (Separated Cauchy-net completion carrier)**.** *A* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, M, Q, L, W, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, including the finite tail requests and common-upper witnesses consumed by the completion route.*
2. *M is the* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *boundary row that turns the same directed Cauchy-net data into a completion-facing Moore–Smith read.*
3. *Q is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, forcing equality-facing reads through separated zero-distance evidence rather than host equality or quotient-net equality.*
4. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, recording the metric-completion consumer surface reached by the displayed net route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row attached to the completion read before regular rational tolerances are exported.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, recording the finite regular rational budget used by numerical consumers.*

7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after C, M, Q, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for Cauchy-net windows, Moore–Smith boundary data, separated comparison, metric-completion handoff, finite stream observations, regular rational tolerances, Real seals, replay data, provenance, and local naming.*
9. *K records displayed* Cont *replay from Cauchy-net evidence through Moore–Smith completion, separated completion, metric completion, stream-name observation, regular rational readback, and Real-facing sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient nets, selected limits, host equality of completion points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete spaces, or an external Real-completion oracle.*

Lean4 ✓ `BEDC.Derived.SeparatedCauchyNetCompletionUp.SeparatedCauchyNetCompletionTasteGate`
`single carrier alignment`

Theorem 3086.2 (Separated Cauchy-net completion NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier $S = (C, M, Q, L, W, R, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetC}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Cauchy-net, Moore–Smith completion, separated completion, metric completion, stream-name, regular rational, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-net coverage, requiring every completion-facing read to begin with the* $\mathsf{CauchyNet}^{\uparrow}$ *row C;*
3. *Moore–Smith boundary exposure, requiring M before separated or metric completion consumers receive the packet;*
4. *separated comparison discipline, requiring Q to mediate all equality-facing reads before any Real-facing seal is exported;*
5. *metric and numerical handoff discipline, requiring L, W, R, A to be read in that order for metric-completion and Real-facing consumers;*
6. *structural preservation, requiring H, K, P, N to transport, replay, source, and name only the displayed route;*
7. *ledger non-escape, excluding quotient nets, selected limits, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete spaces, and external Real-completion oracles.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4066.1. The accepted packet exposes exactly the Cauchy-net, Moore–Smith, separated-completion, metric-completion, stream-name, regular rational, Real-seal, and structural rows.

Directed-net coverage and Moore–Smith boundary exposure fix the first part of the route. A completion-facing request cannot reach Q, L, W, R, A until the Cauchy-net row C and the Moore–Smith boundary row M have been displayed.

Separated comparison is isolated in Q. Thus equality-facing reads are zero-distance handoffs through the separated-completion surface, not host equality of completed points and not equality of quotient nets.

The metric and numerical handoff rows are downstream. A metric-completion consumer reads L, a finite observation consumer reads W, a tolerance consumer reads R, and the terminal Real-facing consumer reads A only after those prior rows have appeared.

Finally H, K, P, N preserve, replay, source, and name only this ordered route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3086.3 (Separated Cauchy-net completion zero-distance handoff)**.** *Let $S = (C, M, Q, L, W, R, A, H, K, P$ be an accepted* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier. Every comparison between two completion-facing reads of S factors through the route*

$$C \longmapsto M \longmapsto Q \longmapsto L \longmapsto W \longmapsto R \longmapsto A \longmapsto H, K, P, N.$$

In particular, the handoff to $\mathsf{Real}^{\uparrow}$ *is permitted only after the separated zero-distance row Q, the* $\mathsf{MetricCompletion}^{\uparrow}$ *row L, the* $\mathsf{StreamName}^{\uparrow}$ *row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R have been displayed. The comparison exports no quotient net, selected limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Start with a comparison request for an accepted packet. The carrier definition gives the Cauchy-net row C as the only source of directed tail data, so the comparison begins with displayed finite net windows.

The Moore–Smith boundary row M turns the same net data into a completion-facing read. The separated-completion row Q then supplies the zero-distance comparison discipline before any metric or Real-facing consumer can inspect the result.

The route continues through L, W, R, A. These rows record the metric-completion handoff, finite stream observations, regular rational tolerances, and terminal Real seal in order.

Transport, replay, provenance, and local naming in H, K, P, N preserve that route componentwise. Because the carrier has no quotient-net, selected-limit, host-equality, choice, or ambient-completeness coordinate, no comparison can bypass the zero-distance handoff. □

Falsifiable prediction: A SeparatedCauchyNetCompletionUp consumer that reads a Real-facing seal before displaying the separated comparison row Q and the regular rational tolerance row R is not reading an accepted packet of this chapter.

Independence witness: cauchynet, mooresmithcompletion, separatedcompletion, metriccompletion, streamname, regseqrat, real: the separated Cauchy-net completion packet is not bijective to any one sibling because CauchyNet lacks the completion boundary, MooreSmithCompletion lacks the explicit separated Real-facing handoff, SeparatedCompletion lacks the directed-net source window, MetricCompletion lacks the StreamName and RegSeqRat seal order, StreamName lacks the separated comparison row, RegSeqRat lacks the Moore–Smith boundary, and Real lacks the upstream net-completion ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ packet is generated from CauchyNetUp directed-window evidence, MooreSmithCompletionUp boundary data, SeparatedCompletionUp zero-distance comparison, MetricCompletionUp handoff, StreamNameUp observations, RegSeqRatUp tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4066.1, Definition 4066.2, and Definition 4066.3 seed the separated Cauchy-net completion route from directed net windows through zero-distance comparison and Real-facing sealing.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, host equality of completed points, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Moore–Smith boundary exposure, separated zero-distance comparison, metric handoff, stream-name observation, regular rational tolerance readback, Real-facing sealing, transport preservation, and non-escape for BEDC.Derived.SeparatedCauchyNetCompletionUp.

Chapter 3087

A Concrete Naming Certificate for $\mathsf{RoleSeparationLedger}^{\uparrow}$

The $\mathsf{RoleSeparationLedger}^{\uparrow}$ packet realizes the compiler, AI or machine-learning, and human-consciousness comparison of section 9414.59 as finite NameCert data. It sits beside the compiler preservation route of chapter 1097, the compiler replay route of chapter 1055, the prompt-output trace of chapter 965, the large-model inscription point of chapter 939, and the public observer interface of chapter 928. The packet binds three role rows without collapsing compiler preservation, model pattern generation, and self-proxy readback into one classifier.

Definition 3087.1 (RoleSeparationLedger carrier)**.** *A* $\mathsf{RoleSeparationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, A, G, S, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompilerClassifierRoute}^{\uparrow}$ *row, recording source-to-target classifier preservation as a compiler route rather than as a self-proxy or model-output row.*
2. *F is the* $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ *row, recording finite compiler replay and source-to-target* hsame *preservation.*
3. *A is the AI or machine-learning pattern row, sourced in* $\mathsf{PromptResponseTrace}^{\uparrow}$ *and* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *data for prompt, response, corpus, activation, and generated-signature reads.*
4. *G is the generated-output boundary row, naming the model-facing inscription and prompt-response rows that may be inspected by audit consumers.*
5. *S is the self-proxy readback row, recording the local observer-facing self-proxy boundary only through the licensed public rows of* $\mathsf{ObserverInterface}^{\uparrow}$*.*
6. *O is the role-separation ledger row. It records that K, F are preservation rows, A, G are pattern-generation and inscription rows, and S is a self-proxy readback row, with no permitted classifier that identifies the three roles.*
7. *H records componentwise* hsame *transport for the compiler, trace-faithfulness, prompt-response, inscription, observer-interface, role-ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$K, F, A, G, S, O \longmapsto \mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}.$$

9. *P is the* Pkg *provenance over* RoleSeparationLedger$^{\uparrow}$, CompilerClassifierRoute$^{\uparrow}$, CompilerTraceFaithfulness$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, LargeModelInscriptionPoint$^{\uparrow}$, ObserverInterface$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{RoleSeparationLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate that treats compiler self-hosting as consciousness, model generation as self-proxy binding, or public observer readback as an internally ratified model proof.

Lean4 ✓ `BEDC.Derived.RoleSeparationLedgerUp.RoleSeparationLedgerTasteGate single carrier alignment`

Theorem 3087.2 (RoleSeparationLedger NameCert obligations)**.** *For an accepted* RoleSeparationLedger$^{\uparrow}$ *carrier*

$$L = (K, F, A, G, S, O, H, C, P, N),$$

the local NameCert$_{\text{RoleSeparationLedger}^{\uparrow}}$ *surface consists exactly of compiler-route admission for K, compiler-trace faithfulness for F, AI pattern and prompt-response admission for A, large-model inscription admission for G, observer-interface self-proxy readback for S, role separation through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4067.1. It displays the compiler route, compiler replay, AI pattern row, inscription boundary, observer readback, role-separation ledger, and structural rows.

The compiler side is exhausted by K, F. These rows preserve and replay classifier structure, but they do not supply self-proxy readback or model-side generated-signature evidence.

The AI or machine-learning side is exhausted by A, G. These rows name prompt-response and inscription data, but they do not certify compiler preservation and do not bind a current self-center.

The observer-facing side is exhausted by S. The ledger row O records the separation of the three roles, while H, C, P, N transport, replay, source, and name that separation. Hence the local NameCert obligations are exactly the displayed rows. □

Theorem 3087.3 (RoleSeparationLedger role noncollapse)**.** *Every accepted* RoleSeparationLedger$^{\uparrow}$ *packet preserves the three role rows without collapse: compiler preservation remains on K, F, AI pattern generation and inscription remain on A, G, and self-proxy readback remains on S. A consumer may compare the rows only through the role-separation ledger O, not by identifying compiler self-hosting, model output generation, and human-consciousness readback.*

Proof. Begin with the role-separation ledger row O. It records the three role classes and their permitted sibling routes.

The preservation rows K, F are read before any model or observer row can be compared. They expose classifier preservation and trace faithfulness only.

The generation rows A, G are read as prompt-response and inscription data. They may be audited as outputs, but no row in the packet lets them become compiler correctness or self-proxy binding.

The self-proxy row S is read through the observer interface. Since O is the only comparison row and the carrier contains no collapse coordinate, the three role rows remain separated in every accepted read. □

Theorem 3087.4 (RoleSeparationLedger TasteGate target)**.** *The Lean-facing downstream surface for* RoleSeparationLedger$^{\uparrow}$ *is the packet family to be recorded under* Derived/RoleSeparationLedgerUp/TasteGate.lean *with explicit* Nontrivial RoleSeparationLedger$^{\uparrow}$ *and* FieldFaithful RoleSeparationLedger$^{\uparrow}$ *instances.*

The gate must recover the compiler preservation rows K, F, the AI pattern-generation and inscription rows A, G, the self-proxy readback row S, the role-separation ledger O, and the support rows H, C, P, N before admitting the packet.

Proof. Read the carrier first. The fields $K, F, A, G, S, O, H, C, P, N$ are distinct coordinates of the accepted packet.

Nontriviality is witnessed by the presence of at least the compiler, generation, and self-proxy rows together with the ledger row that distinguishes them. Field faithfulness is witnessed by the fact that each accepted read must recover the same rows in the same roles.

The future TasteGate file therefore has a finite obligation surface: it checks that the role rows are present, distinct, replayed, transported, sourced, and locally named before any public admission of $\mathsf{RoleSeparationLedger}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{RoleSeparationLedger}^{\uparrow}$ read preserves the compiler preservation row, AI pattern-generation row, and self-proxy readback row without collapse.

Independence witness: compilerclassifierroute, compilertracefaithfulness, prompt-response-trace, largemodelinscriptionpoint, observer-interface: none is bijective because the role-separation ledger stores the joint noncollapse row O binding compiler preservation, AI pattern generation, and self-proxy readback as distinct finite NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RoleSeparationLedger}^{\uparrow}$ packet is generated from CompilerClassifierRoute and CompilerTraceFaithfulness preservation rows, PromptResponseTrace and LargeModelInscriptionPoint pattern-generation rows, an ObserverInterface self-proxy readback row, a role-separation ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4067.1, Definition 4067.2, Definition 4067.3, and Definition 4067.4 seed the compiler, AI pattern-generation, and self-proxy readback separation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove compiler consciousness, model consciousness, phenomenal consciousness, hidden model truth, internally ratified universal closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compiler preservation, trace faithfulness, prompt-response admission, inscription admission, observer readback, role noncollapse, transport, replay, provenance, and local naming for BEDC.Derived.RoleSeparationLedgerUp.

Chapter 3088

A Concrete Naming Certificate for $\mathsf{PrecompactUniform}^{\uparrow}$

Following Bourbaki and Kelley on precompact uniform spaces, the $\mathsf{PrecompactUniform}^{\uparrow}$ packet records the finite total-bounded uniform route used by the L10 completion spine. It consumes the entourage carrier of chapter 339, the finite covering comparison of chapter 127, the Cauchy request surface of chapter 333, and the completion handoff of chapter 1473. The packet names a precompactness route without importing open-cover compactness, quotient completions, selected limits, countable choice, or ambient completeness.

Definition 3088.1 (PrecompactUniform carrier)**.** *A* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, E, F, T, Q, K, S, R, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, recording the source points, entourage requests, refinement rows, and separatedness boundary available to the packet.*
2. *E is the entourage ledger row. It lists the finitely inspected entourages and the refinement comparisons used before any covering row is read.*
3. *F is the finite cover row. For each displayed entourage request it stores the finite family of centers and the witness that the inspected source points are covered at that entourage.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *comparison row, binding the finite cover data to the sibling total-bounded carrier surface.*
5. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *request row, recording only those Cauchy requests whose entourage demands have already passed through E, F, T.*
6. *K is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row for completion consumers that receive the precompact uniform packet after the finite cover and Cauchy request rows have been displayed.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected request windows.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the finite entourage and readback comparisons.*

10. *H records componentwise* hsame *transport for the uniform source, entourage ledger, finite cover row, total-bounded comparison, Cauchy requests, completion handoff, finite windows, readbacks, dyadic tolerance rows, replay, provenance, and local naming.*

11. *C records displayed* Cont *replay through*

$$X, E, F, T, Q, K, S, R, D \longmapsto \mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}.$$

12. *P is the* Pkg *provenance over* $\mathsf{PrecompactUniform}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient completion equality, selected Cauchy representatives, countable choice, host equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.PrecompactUniformUp.PrecompactUniformTasteGate single carrier alignment`

Theorem 3088.2 (PrecompactUniform NameCert obligations)**.** *For an accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier*

$$U = (X, E, F, T, Q, K, S, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-space admission for X, entourage-ledger exactness for E, finite-cover exactness for F, total-bounded comparison through T, Cauchy-filter request admission through Q, uniform-completion handoff through K, finite-window and readback inspection through S, R, D, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4068.1. It displays the uniform source, entourage ledger, finite cover row, total-bounded comparison, Cauchy request row, completion handoff, finite windows, readbacks, dyadic tolerances, and structural rows.

The analytic route begins with X, E, F, T. A consumer first reads the uniform-space surface, then the inspected entourages, then the finite cover witnesses, and only then the total-bounded comparison row.

The completion-facing route is downstream of Q, K, S, R, D. Cauchy requests, completion handoffs, finite windows, regular readbacks, and dyadic tolerances are therefore bound to the same finite cover data.

Finally H, C, P, N transport, replay, source, and locally name the same packet. Since no excluded object is a coordinate of the carrier, the displayed rows exhaust the local NameCert obligations. □

Theorem 3088.3 (PrecompactUniform completion handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier first reads the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, the entourage ledger, the finite cover row, the* $\mathsf{TotallyBounded}^{\uparrow}$ *comparison row, and the* $\mathsf{CauchyFilter}^{\uparrow}$ *request row before it may inspect the completion handoff K. The handoff is a finite L10 completion route through the displayed rows, not an open-cover compactness theorem or a quotient completion object.*

Proof. Start with the NameCert obligations of Definition 4068.2. They place the completion handoff after the uniform source, entourage, finite-cover, total-bounded, and Cauchy request rows.

The finite windows S, readbacks R, and dyadic tolerances D are read on the same route. They provide the observed request data needed by completion consumers without selecting a limit or quotienting Cauchy names.

Transport and replay through H and C preserve the row order. Thus every completion consumer reaches K only through the displayed precompact uniform packet and cannot bypass it by invoking compactness, choice, or ambient completeness. □

Theorem 3088.4 (PrecompactUniform nonescape)**.** *An accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *packet exports only the displayed rows*

$$X, E, F, T, Q, K, S, R, D, H, C, P, N.$$

No downstream completion, compactness, or Cauchy-filter consumer can obtain open-cover compactness, quotient completion equality, selected representatives, countable-choice schedules, host equality, or ambient completeness from the packet.

Proof. Use Definition 4068.1 to list the available coordinates. The only analytic rows are the uniform source, entourage ledger, finite cover data, total-bounded comparison, Cauchy request, completion handoff, finite windows, readbacks, and dyadic tolerance rows.

Use Definition 4068.3. Completion-facing reads are confined to those rows before the handoff is exposed.

The structural rows H, C, P, N only transport, replay, source, and name the finite route. Because the excluded compactness and completion principles are absent from the carrier, no consumer can project them from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrecompactUniform}^{\uparrow}$ packet is generated from a UniformSpace source row, entourage ledger rows, finite cover rows, a TotallyBounded comparison row, CauchyFilter request rows, a UniformCompletion handoff, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4068.1, Definition 4068.2, Definition 4068.3, and Definition 4068.4 seed the finite precompact uniform route for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient completion, selected representatives, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exactness, finite-cover exactness, total-bounded comparison, Cauchy-filter request admission, completion handoff, readback, tolerance, transport, replay, provenance, and local naming for BEDC.Derived.PrecompactUniformUp.

Chapter 3089

A Concrete Naming Certificate for $\mathsf{RegularCauchyOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyOrder}^{\uparrow}$ packet records the constructive order surface for regular Cauchy real names as finite BEDC NameCert data. It sits over the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the algebraic order boundary of chapter 766, and the terminal real-name seal of chapter 38. The packet gives RealAlgOrder and Real consumers a displayed comparison route on regular names, without trichotomy, quotient stream equality, selected limits, host real equality, or countable choice.

Definition 3089.1 (Regular-Cauchy order carrier)**.** *A* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, Y, S, D, R, L, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source rows whose finite rational windows are compared.*
2. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule naming every inspected precision window for both sources.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, sign budget, and comparison threshold for each inspected window.*
4. *R is the regular-rational readback row that records the finite rational observations obtained from X and Y through S and D.*
5. *L is the order-comparison ledger; it stores only located finite evidence that the displayed readbacks meet, fall below, or remain unresolved at the current dyadic tolerance.*
6. *A is the downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff row, available only after X, Y, S, D, R, L have been displayed.*
7. *H records componentwise* hsame *transport for the regular sources, stream schedule, dyadic ledger, readback row, order ledger, RealAlgOrder handoff, and local certificate rows.*
8. *C records the displayed* Cont *replay from an order request to the stream windows, dyadic tolerance comparison, regular-rational readback, order ledger, and RealAlgOrder handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOrder}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, Y, S, D, R, L, A, H, C, P, N$. It has no coordinate for trichotomy, quotient equality of streams, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyOrderUp.RegularCauchyOrderTasteGate single carrier alignment`

Theorem 3089.2 (Regular-Cauchy order NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *carrier $O = (X, Y, S, D, R, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparisons, regular-rational readback, order-comparison ledger, downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by trichotomy, quotient stream equality, selected limits, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4069.1. The rows $X, Y, S, D, R, L, A, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The two regular Cauchy sources are compared through the same stream schedule S and dyadic tolerance ledger D. The readback row R and order ledger L therefore record finite tolerance evidence, not equality of completed streams or a global trichotomy.

The RealAlgOrder handoff A is downstream of that finite route. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and the carrier has no additional coordinate from which a selector, host equality, choice principle, or ambient completeness theorem could be projected. □

Theorem 3089.3 (Regular-Cauchy order RealAlgOrder handoff)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *packet reads the constructive order surface only through the displayed route*

$$X, Y, S, D, R, L, A, H, C, P, N.$$

The handoff exports finite stream windows, dyadic tolerance comparisons, regular-rational readback, an order-comparison ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It does not export trichotomy, quotient stream equality, selected limits, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the order surface. The carrier displays X and Y only together with the shared stream schedule S, dyadic tolerance ledger D, and finite readback row R.

The order ledger L can be read only after those windows and tolerances have been displayed. If an algebraic-order or Real-facing consumer is present, the downstream row A is available only after the same finite comparison route has been consumed.

Apply the NameCert obligations of Definition 4069.2. The support rows H, C, P, N preserve, replay, source, and name this finite route, so the handoff exports no trichotomy, quotient stream equality, selected limit, host real equality, choice principle, or ambient completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyOrder}^{\uparrow}$ consumer that reaches a $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ order read without first reading two RegSeqRat source rows, a shared StreamName schedule, a DyadicRatCore tolerance ledger, regular-rational readback, an order-comparison ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: regseqrat, dyadicratcore, streamname, realalgorder, real: $\mathsf{RegularCauchyOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular Cauchy source rows to one constructive order ledger and downstream RealAlgOrder handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOrder}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName schedule, a DyadicRatCore tolerance ledger, regular-rational readback, an order-comparison ledger, a RealAlgOrder handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4069.1, Definition 4069.2, and Definition 4069.3 seed the finite constructive order route for regular Cauchy real names through RegSeqRat, DyadicRatCore, StreamName, RealAlgOrder, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient stream equality, selected limits, host real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and RealAlgOrder handoff theorem.

Chapter 3090

A Concrete Naming Certificate for $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the computable-analysis convention that real-facing objects are read through effective names, the $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ packet records the finite interface where a Cauchy representation also carries located Cauchy evidence before the terminal real seal is consumed. It sits after the Cauchy-representation boundary of chapter 1680, the sequence-level located Cauchy packet of chapter 1595, the located real boundary of **??**, and the represented-space source surface of chapter 1494. The packet gives L10 Real and Completion consumers a single BEDC-native route through displayed finite rows; it does not import quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3090.1 (LocatedCauchyRepresentation carrier)**.** *A* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, J, S, M, D, X, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoder row, recording the finite representation route by which stream observations are decoded.*
2. *Q is the* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *evidence row, recording the located tail witness attached to the decoded observations.*
3. *J is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison-window row, exposing the finite interval and locatedness comparisons that the downstream real-facing consumer may inspect.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by the decoder and by the located Cauchy evidence.*
5. *M is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for the regular readback, located tail, and located real comparison windows.*
7. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *exposure row, presenting the same finite name to represented-space consumers without adding a second representation relation.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after R, Q, J, S, M, D, X have been displayed.*

9. *H records componentwise* hsame *transport for the decoder, located evidence, comparison windows, stream windows, readback, tolerances, represented-space exposure, real seal, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from representation decoding through located Cauchy evidence, located real comparison, represented-space exposure, and the terminal seal handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{LocatedCauchySequence}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located Cauchy representation packet. It has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationTasteGate single carrier alignment`

Theorem 3090.2 (LocatedCauchyRepresentation NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier* $L = (R, Q, J, S, M, D, X, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the decoder, located Cauchy evidence, located real comparison windows, stream windows, regular rational readback, dyadic tolerance ledger, represented-space exposure, terminal real seal, transport, replay, provenance, and local naming rows;*

2. *decoder exactness, requiring every representation read to pass through the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row R before any located evidence is inspected;*

3. *located sequence exactness, requiring the row Q to expose only the located Cauchy tail witnesses attached to the displayed decoder and stream-window route;*

4. *comparison-window exactness, requiring the* $\mathsf{LocatedReal}^{\uparrow}$ *row J to read its finite location windows through the same regular rational and dyadic rows M, D;*

5. *represented-space compatibility, requiring X to expose the same finite name and target route rather than a separate representation relation;*

6. *real-seal ordering, requiring E to remain downstream of R, Q, J, S, M, D, X and never to serve as a selected host limit;*

7. *componentwise transport, continuation replay, package provenance, and local naming for exactly the displayed rows H, C, P, N;*

8. *refusal of quotient streams, selected limits, countable choice, trichotomy, host real equality, and ambient completeness as local certificate data.*

No located Cauchy representation obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationNamecertObligations`

Proof. Project the accepted carrier. The visible analytic route is the representation decoder R, located Cauchy evidence Q, located real comparison windows J, stream windows S, regular rational readback M, dyadic tolerance ledger D, represented-space exposure X, and terminal seal E.

Decoder exactness comes first. A representation-facing consumer reaches located evidence only after reading the $\mathsf{CauchyRepresentation}^{\uparrow}$ row, so the packet cannot be reread as an unbounded stream or a selected limit.

The located rows are tied to the same finite route. The row Q supplies located tail evidence over the displayed stream windows, while J supplies located real comparison windows over the same regular rational readback and dyadic tolerance ledger.

The represented-space and seal rows add no independent analytic coordinate. The row X exposes the same name to represented-space consumers, and E is downstream of the decoder, located evidence, comparison windows, readback, and tolerance rows.

Finally, H, C, P, N transport, replay, source, and name the displayed packet without adding data. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3090.3 (LocatedCauchyRepresentation Real-seal handoff)**.** *Let $L = (R, Q, J, S, M, D, X, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RepresentedSpace}^{\uparrow}$*-facing read of L factors through the finite handoff route*

$$R, Q, J, S, M, D, X \longmapsto E \longmapsto H, C, P, N.$$

Thus the terminal real seal and the represented-space exposure are reached only through the Cauchy-representation decoder, located Cauchy evidence, located real comparison windows, finite stream windows, regular rational readback, and dyadic tolerance ledger. The route contains no quotient stream, selected host limit, countable-choice selector, trichotomy branch, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationRealSealHandoff`

Proof. Begin with a real-facing or represented-space-facing read. Carrier admission exposes the target rows X and E only as part of the same finite packet as the decoder, located evidence, comparison windows, stream windows, readback, and tolerances.

Replay the continuation row C. The read first consumes R, then the located evidence Q and comparison windows J, and then the common stream-window, regular-readback, and dyadic-tolerance route S, M, D.

Only after that replay can X and E be inspected. The represented-space row offers the same finite name to its consumer, while the real-seal row is the terminal seal of that route rather than a host-selected limit.

The support rows H, C, P, N preserve and name the same route. Since the packet contains none of the excluded coordinates, no consumer can bypass the displayed handoff to read quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, LocatedCauchySequence evidence, LocatedReal comparison windows, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, RepresentedSpace exposure, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$, seed)

Scope closed: Definition 4070.1, Definition 4070.2, and Definition 4070.3 seed the finite located Cauchy representation route from CauchyRepresentation and LocatedCauchySequence data to LocatedReal comparison windows, RepresentedSpace exposure, and a terminal Real seal.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyRepresentation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, decoder exactness, located sequence exactness, comparison-window exactness, represented-space compatibility, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyRepresentationUp.

Chapter 3091

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ packet records the finite route by which a Bishop-style regular Cauchy real name exposes an explicit modulus before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the name. It sits over the finite stream windows of chapter 126, the rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the located regular interface of chapter 1454, and the extraction discipline of chapter 641. The packet gives the L10 Real and Completion spine a traditional constructive-analysis modulus row without quotient streams, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3091.1 (Bishop Cauchy modulus-extraction carrier)**.** *A* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, E, M, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose inspected finite windows are available to the Cauchy read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed rational observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic precision request, comparison cells, and half-radius splits.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after the stream, regular-rational, and dyadic rows have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row extracted from S, R, D, E through the same finite observation route.*
6. *W is the local* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *witness row tying the extracted threshold to the inspected windows and regular readback.*
7. *H stores componentwise* hsame *transport for the stream schedule, regular readback, dyadic ledger, Real seal, extracted modulus, extraction witness, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay of the route*

$$S, R, D \longmapsto W \longmapsto M \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected completed limit, trichotomy, countable choice, host real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyModulusExtractionUp.BishopCauchyModulusExtractionTasteGate`
`single carrier alignment`

Theorem 3091.2 (Bishop Cauchy modulus-extraction NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *carrier*

$$Q = (S, R, D, E, M, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed stream schedule, regular readback, dyadic tolerance ledger, Real seal, extracted modulus, extraction witness, transport, replay, provenance, and local naming rows;*
2. *window-readback exactness, requiring every modulus request to read the same* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rational observations named by S and R;*
3. *dyadic tolerance compatibility, requiring D to supply the precision and comparison cells used by the extracted threshold;*
4. *extraction locality, requiring W and M to be read only after S, R, D have been displayed;*
5. *seal non-escape, requiring E to remain downstream from the extracted modulus route and to export no quotient stream, selected limit, trichotomy, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier in Definition 4071.1. The accepted packet exposes exactly the stream, regular-rational, dyadic, Real-seal, modulus, extraction-witness, transport, replay, provenance, and naming rows listed there.

The window-readback and dyadic clauses are forced by the displayed source rows. A precision request can inspect only the windows named by S, the rational observations named by R, and the tolerance data named by D.

The extraction rows are downstream from that finite source route. The witness row W binds the observed windows and regular readback to the extracted threshold, while M records the resulting $\mathsf{CauchyModulus}^{\uparrow}$ handoff.

Finally read the terminal seal and structural rows. Since E, H, C, P, N preserve only the displayed route and the carrier has no coordinate for the excluded quotient, choice, order, equality, or completeness data, the five listed obligations exhaust the local certificate surface. □

Theorem 3091.3 (Bishop Cauchy modulus-extraction regular handoff)**.** *Every accepted* BishopCauchyModulusExtr *carrier hands its extracted modulus to later regular-Cauchy and* $\mathsf{Real}^{\uparrow}$ *completion consumers through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulusExtraction}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Thus the handoff exports a finite threshold surface for the same Bishop-style regular Cauchy name, not a quotient stream, selected limit, trichotomy, countable-choice functional, host real equality, or ambient completeness principle.*

Proof. Begin with an accepted carrier $Q = (S, R, D, E, M, W, H, C, P, N)$. The source side of the packet is the displayed route through S, R, D, so a completion consumer first reads finite windows, rational readback, and dyadic tolerance data.

Next read the extraction witness. The row W is local to the same route and supplies the named observation surface used to produce the modulus threshold.

The extracted modulus row M is then the only handoff row available to downstream regular-Cauchy consumers. It is transported and replayed through H and C, while P and N source and name the same finite packet.

The terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream from that chain. Because no other carrier coordinate stores quotient equality, a selected limit, trichotomy, countable choice, host real equality, or ambient completeness, every regular handoff factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ packet is generated from finite StreamName windows, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a RealUp seal row, a CauchyModulus row, a CauchyModulusExtraction witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4071.1, Definition 4071.2, and Definition 4071.3 seed the finite Bishop-style modulus-extraction route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, trichotomy, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, dyadic tolerance compatibility, extraction locality, regular handoff stability, and RealUp seal non-escape for BEDC.Derived.BishopCauchyModulusExtractionUp.

Chapter 3092

A Concrete Naming Certificate for $\mathsf{HausdorffUniformity}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffUniformity}^{\uparrow}$ packet records the separated uniformity interface used by completion and uniqueness consumers before any quotient completion or representative choice is available. It sits between the entourage surface of chapter 339, the zero-distance classifier discipline of chapter 1411, the metric completion route of chapter 1506, and the Cauchy-filter completion boundary of chapter 1569. The packet names the finite Hausdorff-uniformity boundary as BEDC data; it does not identify host points, select completion representatives, or import a quotient completion.

Definition 3092.1 (Hausdorff uniformity carrier)**.** *A* $\mathsf{HausdorffUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (U, E, D, Z, T, C, P, N)$$

with the following displayed rows:

1. *U is the underlying* $\mathsf{UniformSpace}^{\uparrow}$ *row, including the finite entourage basis, diagonal-membership ledger, and refinement surface read by downstream consumers.*
2. *E is the uniform entourage row, recording the displayed entourages through which all separation and completion-facing comparisons must pass.*
3. *D is the diagonal separation row, witnessing that indistinguishability by the displayed entourages reaches the local classifier rather than host equality.*
4. *Z is the zero-distance handoff row, routing metric-generated separated reads through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier before a completion consumer inspects uniqueness.*
5. *T records componentwise* hsame *transport for U, E, D, Z and for the local continuation, provenance, and naming rows.*
6. *C records displayed* Cont *replay of the route*

$$U, E \longmapsto D \longmapsto Z.$$

7. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformity}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares observations only through U, E, D, Z, T, C, P, N. It has no coordinate for quotient uniform completions, host equality of points, selected representatives, countable choice, ambient completeness, or bridge data.

Lean4 ✓ `BEDC.Derived.HausdorffUniformityUp.HausdorffUniformityTasteGate` single carrier alignment

Theorem 3092.2 (Hausdorff uniformity NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffUniformity}^{\uparrow}$ *carrier*

$$H = (U, E, D, Z, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the uniform-space row, entourage row, diagonal separation row, zero-distance handoff row, transport, replay, provenance, and local naming rows;*
2. *entourage stability, requiring every separated comparison to begin with the displayed entourage basis and refinement data of U, E;*
3. *diagonal separation, requiring all indistinguishability-to-classifier reads to pass through D rather than through host equality;*
4. *completion-facing zero-distance handoff, requiring metric or filter completion consumers to read Z before using uniqueness data;*
5. *componentwise transport and displayed replay through T, C;*
6. *ledger non-escape, excluding quotient completions, host point equality, selected representatives, countable choice, ambient completeness, Lean verification, and bridge checking.*

No local NameCert obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 4072.1. The accepted tuple displays exactly the uniform-space row, the entourage row, the diagonal separation row, the zero-distance handoff, and the structural rows.

Entourage stability follows from the order of the carrier. The rows U, E supply the displayed entourage basis and refinement data before diagonal separation can be read, so a consumer cannot replace finite uniform evidence by host equality of points.

The diagonal separation row D is the only exported route from uniform indistinguishability to the local classifier. The zero-distance handoff row Z then records the metric-generated or completion-facing comparison surface without adding a quotient completion or a representative selector.

Finally inspect T, C, P, N. Transport is componentwise, replay follows the displayed route, and provenance plus local naming mention only the rows listed in the carrier. Hence the packet exports no quotient completion, host equality, selected representative, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffUniformity}^{\uparrow}$ packet is generated from a UniformSpace row, a displayed entourage row, a diagonal separation row, a zero-distance handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4072.1 and Definition 4072.2 seed the finite Hausdorff-uniformity interface over uniform-space, entourage, diagonal separation, zero-distance handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host equality of points, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage stability, diagonal separation, completion-facing zero-distance handoff, componentwise transport, replay, provenance, and ledger non-escape for BEDC.Derived.HausdorffUnif

Chapter 3093

A Concrete Naming Certificate for $\mathsf{CauchyCutComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCutComparison}^{\uparrow}$ packet names the finite comparison surface between the located-cut presentation of chapter 2312 and the regular-Cauchy equality surface of chapter 1820. It reads the finite windows of chapter 126, the regular rational source of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal boundary of chapter 38. The packet compares presentations by displayed rows only; it does not identify quotient representatives, select limits, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3093.1 (Cauchy cut comparison carrier)**.** *A* $\mathsf{CauchyCutComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, Q, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the cut-side exposure row. It records the accepted* $\mathsf{CauchyCut}^{\uparrow}$ *lower–upper located-cut route before any equality-facing comparison is read.*
2. *Q is the setoid-side exposure row. It records the accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *regular-Cauchy comparison surface.*
3. *S is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window ledger shared by the cut-to-stream route and the setoid equality route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, displaying the regular rational observations consumed by both sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records bracket, midpoint, and comparison tolerances at the same requested precision.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after K, Q, S, R, D have been replayed.*
7. *H records componentwise* hsame *transport for the cut-side row, setoid-side row, synchronized windows, readback, dyadic tolerance, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a comparison request through the cut side, the setoid side, the synchronized window ledger, regular readback, dyadic tolerance, and then the seal row.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCutComparison}^{\uparrow}$, $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $K, Q, S, R, D, A, H, C, P, N$. It has no coordinate for quotient equality, selected representatives, host real equality, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonCarrier`

Theorem 3093.2 (Cauchy cut comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCutComparison}^{\uparrow}$ *carrier $J = (K, Q, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed cut-side exposure, setoid-side exposure, synchronized window ledger, regular readback, dyadic tolerance ledger, Real seal boundary, transport, continuation, provenance, and local naming row;*
2. *cut-side exposure, requiring the located lower–upper cut rows of* $\mathsf{CauchyCut}^{\uparrow}$ *to be visible before any comparison consumes the regular-Cauchy equality surface;*
3. *setoid-side exposure, requiring the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *windows, readbacks, tolerance tests, and equality-facing classifier row to remain displayed;*
4. *synchronized tolerance, requiring S, R, D to provide the same finite precision request to the cut-to-stream route and the regular-Cauchy setoid route;*
5. *Real seal discipline, requiring A to remain downstream of K, Q, S, R, D;*
6. *ledger refusal, excluding quotient equality, selected representatives, host real equality, countable choice, and ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonCarrier namecert obligations`

Proof. Project the carrier of Definition 4073.1. The projection exposes the cut-side row, setoid-side row, synchronized windows, regular readback, dyadic tolerance, Real seal, and the structural rows H, C, P, N.

The cut-side and setoid-side obligations are forced by the source rows K and Q. The row K displays the located-cut-to-stream route, while Q displays the regular-Cauchy equality surface; neither row contains a quotient or selected representative coordinate.

The synchronized tolerance obligation is the shared route through S, R, D. Replay by C sends a comparison request through the same finite windows, regular rational observations, and dyadic tolerance entries before the seal row A is available.

Transport by H preserves each displayed component, and P, N source and name only this finite packet. Therefore the local certificate is exhausted by the listed obligations. □

Theorem 3093.3 (Cauchy cut comparison route)**.** *Let $J = (K, Q, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyCutComparison}^{\uparrow}$ *packet. Every comparison read between the located-cut presentation and the regular-Cauchy setoid presentation factors through the finite route*

$$K, Q \longmapsto S \longmapsto R \longmapsto D \longmapsto A \longmapsto H, C, P, N.$$

Thus the located-cut side and the regular-Cauchy setoid side are compared only after their StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerances, and Real seal boundary have been displayed. The route contains no quotient equality, selected limit, countable-choice selector, host real equality, or ambient Real completeness principle.

Proof. Begin with a comparison-facing consumer. Carrier admission gives access to K and Q only as displayed rows: K exposes the located-cut-to-stream route from chapter 2312, and Q exposes the regular-Cauchy classifier route from chapter 1820.

Replay the continuation row C. The request next consumes the synchronized StreamName ledger S, the RegSeqRat readback row R, and the DyadicRatCore tolerance ledger D at the same precision boundary.

Only after this finite replay can the seal boundary A be read. Componentwise transport, provenance, and local naming preserve the same dependency order, so the comparison remains a displayed presentation comparison rather than a quotient, selection, choice, host-equality, or completeness claim. □

Lean4 ✓ `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonRoute`

Falsifiable prediction: Every accepted $\mathsf{CauchyCutComparison}^{\uparrow}$ read that compares located-cut and regular-Cauchy presentations factors through displayed $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: cauchy-cut, cauchy-equivalence-setoid, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCutComparison}^{\uparrow}$ is not reducible to any listed sibling because it binds the cut-side exposure and setoid-side exposure to one synchronized tolerance ledger and one Real seal boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4073.1, Definition 4073.2, and Definition 4073.3 seed the comparison surface between located-cut and regular-Cauchy presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-side exposure, setoid-side exposure, synchronized tolerance, Real seal discipline, and ledger refusal for BEDC.Derived.CauchyCutComparisonUp.

Constructive story: A $\mathsf{CauchyCutComparison}^{\uparrow}$ packet is generated from displayed CauchyCut and CauchyEquivalenceSetoid rows, synchronized StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 3094

A Concrete Naming Certificate for $\mathsf{FiniteMarkovPartition}^{\uparrow}$

Following Bowen and Walters, the $\mathsf{FiniteMarkovPartition}^{\uparrow}$ packet records the finite symbolic-dynamics surface by which a compact dynamics calculation is read as cylinder data before any consumer asks for a subshift, invariant measure, or orbit quotient. It sits downstream of the finite partition carrier of chapter 201, the adjacency rows of chapter 49, the dynamical-system interface of chapter 195, the local update discipline of chapter 934, and the stream window naming surface of chapter 126. The packet names only a finite Markov partition, its cylinder blocks, adjacency matrix, refinement ledger, transport, replay, provenance, and local NameCert rows; it does not quotient infinite orbits or assert a host-side shift space.

Definition 3094.1 (Finite Markov partition carrier)**.** *A* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, B, J, R, T, H, C, S, N)$$

with the following displayed rows:

1. *P is the finite partition row, read through* $\mathsf{Partition}^{\uparrow}$*, naming the carried atoms and their boundary tags.*
2. *B is the finite cylinder-block row, recording the* BHist *words obtained by reading finitely many successive visits to atoms of P.*
3. *J is the adjacency-matrix row, read through* $\mathsf{Mat}^{\uparrow}$*, whose entries record which atom-to-atom transitions are admitted by the carried dynamics.*
4. *R is the refinement row, recording how a cylinder block refines its initial atom, terminal atom, and every intermediate transition listed by J.*
5. *T is the dynamics-transport row, read through* $\mathsf{DynSystem}^{\uparrow}$*, recording the finite action on the displayed atoms and the compatibility route from atom reads to cylinder reads.*
6. *H records componentwise* hsame *transport for partition atoms, cylinder blocks, adjacency entries, refinement rows, dynamics transport, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from atom reads to block reads, from block reads to adjacency tests, and from adjacency tests to refinement rows.*
8. *S is the* Pkg *provenance over* $\mathsf{FiniteMarkovPartition}^{\uparrow}$*,* $\mathsf{Partition}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $P, B, J, R, T, H, C, S, N$. It has no coordinate for an infinite orbit quotient, a selected itinerary beyond the displayed cylinder window, a measure on a shift space, or an ambient topological conjugacy.

Lean4 ✓ `BEDC.Derived.FiniteMarkovPartitionUp.FiniteMarkovPartitionTasteGate single carrier alignment`

Theorem 3094.2 (Finite Markov partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *carrier $M = (P, B, J, R, T, H, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure, finite cylinder-block exposure, adjacency-matrix exposure, refinement-row exactness, dynamics transport, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the finite packet $M = (P, B, J, R, T, H, C, S, N)$. The partition row P supplies the carried atoms, and the cylinder-block row B reads only finitely many visits to those atoms.

The adjacency row J is then read as a finite matrix over the same atoms. The refinement row R records that each displayed cylinder block refines its atom endpoints and the intermediate transitions permitted by J.

The dynamics row T supplies the finite transport route from atom reads to cylinder reads. Rows H, C, S, N preserve, replay, source, and name exactly that route, so the listed rows exhaust the NameCert obligations. □

Theorem 3094.3 (Finite Markov partition cylinder refinement)**.** *Every accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *cylinder block factors through the partition row P, the adjacency-matrix row J, and the refinement row R. A downstream* $\mathsf{CellularAutomaton}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *consumer may replay the block as a finite transition window, but it receives no route to a hidden infinite itinerary outside B, J, R, H, C, S, N.*

Proof. Start with the carrier of Definition 4074.1. A cylinder block is present only as a row of B, and B is indexed by the finite atoms displayed by P.

The adjacency matrix J checks each neighboring pair in the block. The refinement row R then binds the block to its initial atom, terminal atom, and internal transition witnesses, so the block can be read only through the finite adjacency surface.

Replay through C gives a finite transition window suitable for $\mathsf{CellularAutomaton}^{\uparrow}$ or $\mathsf{StreamName}^{\uparrow}$ consumers. Since the carrier has no coordinate for an unbounded itinerary or orbit quotient, the exported object remains the displayed finite cylinder refinement. □

Theorem 3094.4 (Finite Markov partition symbolic handoff)**.** *An accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *packet can be handed to a subshift-facing or dynamics-facing consumer only as the finite route*

$$P \longmapsto B \longmapsto J \longmapsto R$$

together with T, H, C, S, N. The handoff supplies a symbolic refinement surface for the carried dynamics, not a completed shift space, invariant measure, orbit quotient, or global conjugacy theorem.

Proof. Apply the NameCert obligations of Definition 4074.2. They expose only the finite partition, cylinder, adjacency, refinement, dynamics-transport, transport, replay, provenance, and local naming rows.

Use cylinder refinement from Definition 4074.3. Every block read has already passed through the displayed adjacency and refinement rows, hence a subshift-facing consumer receives a finite transition surface rather than a hidden stream of choices.

The excluded objects are not carrier coordinates. There is no row for a completed shift space, invariant measure, quotient orbit, or global conjugacy, so the symbolic handoff is exactly the finite NameCert route shown in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMarkovPartition}^{\uparrow}$ packet is generated from finite partition atoms, finite cylinder blocks, adjacency-matrix rows, refinement rows, dynamics transport, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation theorem, cylinder-refinement theorem, symbolic-handoff theorem, and finite symbolic-dynamics boundary for $\mathsf{FiniteMarkovPartition}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim a completed subshift, invariant measure, orbit quotient, global conjugacy theorem, or infinite itinerary outside the displayed finite cylinder and adjacency rows.

Upgrade path: To reach obligationClosure, the finite carrier, adjacency refinement, symbolic handoff, and ledger non-escape rows must be encoded and checked over BHist packets.

Chapter 3095

A Concrete Naming Certificate for $\mathsf{RegularCauchyPositivePart}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ packet reads the constructive positive part of a regular Cauchy real name through the finite identity $x^+ = max(x, 0)$. The certificate sits beside the absolute-value, maximum, and minimum surfaces of chapter 1499, chapter 1502, and chapter 1528; it uses the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The downstream real lattice comparison surface of chapter 1674 consumes the same finite order-operation discipline. The packet does not import trichotomy, host real order, quotient stream equality, selected limits, or ambient completeness.

Definition 3095.1 (RegularCauchyPositivePart carrier)**.** *A* $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Z is the displayed zero regular Cauchy row over the same rational readback surface.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to observe X and Z at the same finite precision requests.*
4. *D_X and D_Z are the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-and-tolerance rows read from the shared window schedule.*
5. *A is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *compatibility row for the absolute-value route attached to X.*
6. *M is the* $\mathsf{RegularCauchyMax}^{\uparrow}$ *row for the finite maximum route $max(X, Z)$.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the positive-part windows and the displayed tolerance ledger.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R has been exposed.*
9. *H records componentwise* hsame *transport for the source, zero row, shared window, dyadic rows, absolute-value compatibility, maximum row, readback, seal, provenance, and local name.*
10. *C records the displayed* Cont *routes from the source and zero rows through the shared windows, dyadic observations, maximum row, readback, and real seal.*

11. *L records* Pkg *provenance over* RegularCauchyPositivePart$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegularCauchyAbs$^\uparrow$, RegularCauchyMax$^\uparrow$, RegularCauchyMin$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{RegularCauchyPositivePart}^\uparrow}$ *row.*

The carrier compares accepted packets only through $X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N$. It has no coordinate for trichotomy of completed reals, host real order, quotient stream equality, selected limits, or a Real$^\uparrow$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyPositivePartUp.RegularCauchyPositivePartTasteGate single carrier alignment`

Theorem 3095.2 (RegularCauchyPositivePart NameCert obligations)**.** *For every accepted* RegularCauchyPositiveP *carrier $P = (X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N)$, the local* NameCert$_{\text{RegularCauchyPositivePart}^\uparrow}$ *surface consists exactly of source* RegSeqRat$^\uparrow$ *admission, displayed zero-row admission, shared* StreamName$^\uparrow$ *window exposure,* DyadicRatCore$^\uparrow$ *tolerance observations,* RegularCauchyAbs$^\uparrow$ *compatibility, the* RegularCauchyMax$^\uparrow$ *positive-part row,* RegSeqRat$^\uparrow$ *readback,* Real$^\uparrow$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet P. The coordinates expose the source regular name, the zero row, the shared finite windows, dyadic endpoint-and-tolerance rows, the absolute-value compatibility row, the maximum row, the regular readback, the real seal, and the structural rows.

The operation obligation is the finite lattice route. The maximum surface of chapter 1502 supplies the row M for $max(X, Z)$, while the absolute-value surface of chapter 1499 supplies the compatibility row A used by consumers that read the equivalent absolute-value factorization. The minimum surface of chapter 1528 supplies the sibling lower-part comparison discipline, so the positive-part packet remains aligned with the same displayed-window lattice family.

The readback and seal obligations are then the finite handoffs through RegSeqRat$^\uparrow$ and Real$^\uparrow$. The rows H, C, L, N transport, replay, source, and name this same route without adding trichotomy, host real order, quotient streams, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyPositivePartUp.RegularCauchyPositivePartCarrier namecert obligations`

Theorem 3095.3 (RegularCauchyPositivePart Real seal handoff)**.** *Every* Real$^\uparrow$ *or real-lattice consumer of an accepted* RegularCauchyPositivePart$^\uparrow$ *carrier reaches the positive-part seal row E only after the shared source and zero windows, the dyadic tolerance rows, the* RegularCauchyMax$^\uparrow$ *row M, and the* RegSeqRat$^\uparrow$ *readback row R have been exposed. The route is compatible with the* RealMinMax$^\uparrow$ *lattice boundary of chapter 1674, but it exports no host order comparison, trichotomy witness, quotient stream equality, selected limit, or ambient completeness principle.*

Proof. Start with a Real$^\uparrow$ or lattice-facing consumer. By Definition 4075.1, the only real-facing endpoint in the packet is the seal row E, and that row is downstream of the readback row R.

The readback is downstream of the finite maximum row M. The maximum route consumes the source row X, the displayed zero row Z, the shared window schedule W, and the dyadic tolerance rows D_X, D_Z before the positive-part value can be named.

Finally compare the route with the real min/max boundary. The sibling RealMinMax$^\uparrow$ surface consumes displayed real seals through finite min/max rows, so this packet hands it a positive-part seal already produced by regular-Cauchy windows, dyadic tolerance rows, readback, transport,

replay, provenance, and local naming. Since none of the excluded analytic principles occurs as a coordinate of P, no consumer can reach the positive part by a non-displayed route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ consumer that exports a positive-part seal without first reading the source RegSeqRat row, displayed zero row, shared StreamName windows, DyadicRatCore tolerance rows, RegularCauchyMax row, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: regular Cauchy positive part: $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ binds a source RegSeqRat row, displayed zero row, shared StreamName windows, DyadicRatCore tolerance observations, RegularCauchyAbs compatibility, RegularCauchyMax readback, RegSeqRat handoff, RealUp seal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite lattice-operation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ packet is generated from one RegSeqRat source row, a displayed zero row, shared StreamName windows, DyadicRatCore endpoint observations, a RegularCauchyAbs compatibility row, a RegularCauchyMax row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4075.1, Definition 4075.2, and Definition 4075.3 seed the finite constructive positive-part route through regular Cauchy maximum, readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, trichotomy on completed real names, host real order, quotient stream equality, selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, zero-row admission, window exposure, dyadic tolerance rows, RegularCauchyMax handoff, RealUp seal boundary, transport stability, provenance, and non-escape for the same finite packet.

Chapter 3096

A Concrete Naming Certificate for $\mathsf{Equicontinuity}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style compact-metric reasoning, the $\mathsf{Equicontinuity}^{\uparrow}$ packet records the finite family-modulus surface by which a compact source, a finite continuous-family ledger, and a shared tolerance-radius schedule are presented before any finite-net selector or Arzela–Ascoli consumer may read the family. It sits over the compact source of chapter 129, the continuous graph and pointwise-modulus rows of chapter 130, the tolerance-only modulus consumer of chapter 369, and the finite-net family route of chapter 2362.

The packet exposes a shared family modulus. It does not assert compactness, select subsequences, provide a finite net, choose an Arzela–Ascoli limit, or import host-level continuity.

Definition 3096.1 (Equicontinuity carrier)**.** *An* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, \epsilon, \rho, M, T, R, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row whose finite-cover and metric inspection data are read through* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *F is the finite continuous-family row, each member of which is presented through* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus data over the same source.*
3. *ϵ is the tolerance row consumed by every family member before a radius is exported.*
4. *ρ is the shared radius row, read as a single family modulus rather than as one radius per function.*
5. *M is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that records the shared tolerance-radius handoff.*
6. *T records componentwise* hsame *transport for K, F, ϵ, ρ, M.*
7. *R records displayed* Cont *replay from compact-source observations through continuous-family rows into the shared-modulus handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected finite net, selected subsequence, compactness theorem, host continuity predicate, pointwise-only modulus family, or ambient completeness principle.

Lean4 (definition) `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier`

Theorem 3096.2 (Equicontinuity NameCert obligations)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-family ledger, tolerance row, shared radius row, modulus handoff, transport, replay, provenance, and local naming; shared-modulus exactness requiring the same* ρ *to serve every member of* F *at tolerance* ϵ*; compact-source compatibility requiring any finite-net consumer to read* K *before using the modulus row; and nonescape excluding compactness assertions, selected finite nets, selected subsequences, host continuity, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4076.1. The tuple displays exactly $K, F, \epsilon, \rho, M, T, R, P, N$.

Shared-modulus exactness is the role of ρ and M. The radius row is a single displayed row for the family, so a consumer cannot replace it by independently chosen radii for different members of F.

Compact-source compatibility is the route through K. A finite-net or compact-uniform consumer must read the compact metric source before it can consume the shared family modulus.

Finally, T, R, P, N transport, replay, source, and name the same displayed rows. Since the carrier has no selected net, selected subsequence, compactness theorem, host-continuity predicate, or ambient-completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3096.3 (Equicontinuity compact-metric route)**.** *Every compact-uniform or finite-net consumer of an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier factors through*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The route first reads the compact metric source K*, then the continuous-family ledger* F*, then the shared tolerance-radius handoff* ϵ, ρ, M*. It may be consumed by the finite-net equicontinuity surface of chapter 2362, but it does not itself supply a finite net or compactness proof.*

Proof. Project the accepted carrier. The only source coordinate is K, so any compact-uniform consumer starts from the displayed compact metric row.

Read the family row F next. Its entries are continuous-map rows over the same source, so the family cannot change source data while the modulus is being consumed.

The shared handoff is ϵ, ρ, M. It records a tolerance and one family radius before a downstream finite-net selector reads the packet. Transport and replay through T, R preserve this order, while P, N merely package and name it. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactMetricRoute`

Theorem 3096.4 (Equicontinuity nonescape)**.** *An accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier exports only the finite family-modulus rows* $K, F, \epsilon, \rho, M, T, R, P, N$*. It cannot export compactness, a selected finite net, a selected subsequence, Arzela–Ascoli convergence, host-level continuity, pointwise-only radius choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition lists the entire public inventory. The analytic rows are the compact source, continuous family, tolerance, shared radius, and modulus handoff.

The structural rows transport and replay those same rows. They do not add a selector, a compactness principle, a subsequence object, a convergence row, or an external continuity predicate.

Thus every downstream route must obtain finite nets, compactness conclusions, and convergence statements from separate consumer chapters. The $\mathsf{Equicontinuity}^{\uparrow}$ packet supplies only the displayed family-modulus boundary. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityTasteGate single carrier alignment`

Theorem 3096.5 (Equicontinuity family-window exposure)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family window exposed to compact-uniform consumers is the finite* BHist *route through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row* K*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger* F*, the tolerance row* ϵ*, the shared radius row* ρ*, the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff* M*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. Each pointwise modulus read for a member of* F *must occur inside that window before any finite-net radius is consumed.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFamilyWindowExposure`

Proof. Project the carrier rows. The finite family window is exactly the displayed sequence $K, F, \epsilon, \rho, M, T, R, P, N$, so the compact source and family ledger are read before the shared-radius handoff.

The $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus entries live in F. They cannot be detached from the $\mathsf{CompactMetric}^{\uparrow}$ source K, because the compact-metric route forces every downstream read through the same source row.

Finally, T, R, P, N only transport, replay, source, and name this window. A finite-net consumer therefore receives pointwise modulus data only through the displayed family window and not as an external radius choice. □

Theorem 3096.6 (Equicontinuity compact-net radius exactness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier. Any* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net radius consumed from* E *is the same shared row* ρ *that is handed to* $\mathsf{UniformModulus}^{\uparrow}$ *by* M *after the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family window has been displayed. The* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows preserve this radius exactly and add no memberwise radius ledger, selected net, or host compactness proof.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier compact net radius exactness`

Proof. Read ρ between the tolerance row ϵ and the handoff row M. The carrier contains one shared radius row, not a list of member-indexed radii.

Use the compact-metric route and family-window exposure. The compact source K and the finite continuous-family ledger F are visible before the radius is consumed by a finite-net or uniform-modulus reader.

Transport and replay preserve the row rather than recomputing it. Thus the finite-net radius available to a compact consumer is exactly ρ through T, R, P, N, with no selected net or host compactness coordinate added by the packet. □

Theorem 3096.7 (Equicontinuity uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier factors through the finite route*

$$K, F, \epsilon, \rho, M, T, R, P, N.$$

The handoff reads the CompactMetric$^{\uparrow}$ *finite-net radius boundary through* K, ρ*, reads each* ContinuousMap$^{\uparrow}$ *pointwise modulus through the finite family row* F*, and then exposes* M *as the shared* UniformModulus$^{\uparrow}$ *consumer row. The* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows export no pointwise-only modulus family, selected finite net, or ambient* Real$^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityUniformModulusHandoff`

Proof. Start from family-window exposure. It fixes the order in which K, F, ϵ, and ρ are displayed before the handoff row M can be read.

Apply compact-net radius exactness. The radius consumed by the compact finite-net boundary is the same shared row ρ, and every pointwise modulus entry remains attached to the finite ContinuousMap$^{\uparrow}$ ledger F.

The structural rows then repack the handoff. Since T, R, P, N preserve and name only the displayed route, the exported UniformModulus$^{\uparrow}$ row is a shared handoff and not a selected net, pointwise-only radius family, or completeness principle. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier uniform modulus handoff`

Theorem 3096.8 (Equicontinuity shared-radius stability)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* Equicontinuity$^{\uparrow}$ *carrier. Any transport of the compact source* K*, finite family row* F*, tolerance* ϵ*, and modulus handoff* M *through* T *preserves the same shared radius row* ρ*. Thus a consumer reading* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*, or* FiniteNetEquicontinuity$^{\uparrow}$ *through* R, P, N *cannot replace* ρ *by pointwise radii depending on members of* F*.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusStability`

Proof. Project the carrier of Definition 4076.1. The row ρ is displayed once, between the tolerance row ϵ and the modulus handoff row M.

Transport through T is componentwise on the displayed rows. It may identify the compact source, the family row, the tolerance row, and the handoff row with transported copies, but it has no coordinate for a family-indexed list of radii.

Replay through R and provenance through P, N therefore reread the same shared-radius row. The compact metric, continuous-map, uniform-modulus, and finite-net consumers inherit the single ρ boundary rather than a pointwise-radius surface. □

Theorem 3096.9 (Equicontinuity compact-source root admission)**.** *Every accepted* Equicontinuity$^{\uparrow}$ *carrier first exposes the compact metric source row* K *before any shared-radius, finite-net, or Arzela–Ascoli-facing consumer reads* $F, \epsilon, \rho, M, T, R, P, N$*. The exposed row is the displayed* CompactMetric$^{\uparrow}$ *source of Definition 4076.1, transported by* hsame*, replayed by* Cont*, and sourced by the same* Pkg *provenance and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootCompactSourceAdmission`

Proof. Project the carrier of Definition 4076.1. Its first analytic coordinate is the compact metric source row K, and every later row is attached to that source.

Apply the NameCert obligation surface and compact-metric route, Definition 4076.2 and Definition 4076.3. They force the finite family, tolerance, shared radius, and modulus handoff to be read only after K has been admitted.

Finally use shared-radius stability. Transport and replay preserve the same K row while Pkg provenance and local naming expose the packet, so no downstream consumer can bypass the displayed CompactMetric$^{\uparrow}$ source. □

Theorem 3096.10 (Equicontinuity family-modulus exactness root admission)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier, the shared radius row ρ is the unique family-radius handoff visible to finite-net consumers. It is read after the compact source K, finite* ContinuousMap$^{\uparrow}$ *family F, and tolerance row ϵ, and before the* UniformModulus$^{\uparrow}$ *or* FiniteNetEquicontinuity$^{\uparrow}$ *consumer surface; no memberwise radius schedule or second modulus row is exported.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFamilyModulusExactness`

Proof. Read K, F, ϵ, ρ, M from the carrier. The row ρ appears once, between the tolerance request and the modulus handoff.

Apply shared-radius stability and modulus-ledger exactness, Definition 4076.8 and Definition 4076.20. They identify ρ as the transported shared radius and make the handoff exact for the finite family ledger.

Use finite-net readiness to locate the downstream boundary. A FiniteNetEquicontinuity$^{\uparrow}$ consumer sees the same ρ through hsame, Cont, Pkg, and NameCert rows, and the nonescape theorem excludes memberwise radius replacement. □

Theorem 3096.11 (Equicontinuity finite-net consumer boundary root admission)**.** *The* Equicontinuity$^{\uparrow}$ *carrier supplies only compact-source, finite-family, tolerance, shared-radius, modulus-handoff, transport, replay, provenance, and local naming rows. Any finite net, compactness output, Arzela–Ascoli convergence, or selected subsequence must be obtained from downstream consumer chapters after reading this boundary, not from* Equicontinuity$^{\uparrow}$ *itself.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetConsumerBoundary`

Proof. Project the carrier and the NameCert obligations. The analytic rows are K, F, ϵ, ρ, M, and the structural rows are T, R, P, N.

Apply finite-net readiness and the nonescape theorem, Definition 4076.16 and Definition 4076.4. They place finite-net access after the compact-source and shared-modulus rows while excluding a selected net, subsequence, compactness theorem, or convergence row from the carrier.

Thus finite-net and Arzela–Ascoli consumers may depend on the explicit carrier boundary, but their nets and convergence conclusions must come from downstream chapters. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetConsumerBoundary`

Theorem 3096.12 (Equicontinuity shared-radius carrier root row)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the shared radius row ρ is exposed as a finite* BHist *row between the tolerance request ϵ and the modulus handoff M. The same packet displays the compact source K, finite* ContinuousMap$^{\uparrow}$ *family F, componentwise* hsame *transport T,* Cont *replay R,* Pkg *provenance P, and local* NameCert *row N, so any* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, *or* FiniteNetEquicontinuity$^{\uparrow}$ *consumer reads one shared-radius carrier route rather than memberwise radius data.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedRadiusCarrier`

Proof. Project the carrier of Definition 4076.1. The tuple displays exactly $K, F, \epsilon, \rho, M, T, R, P, N$, and ρ appears as one row before the modulus handoff.

Apply the finite-net consumer boundary of Definition 4076.11. It keeps every consumer on the compact-source, finite-family, tolerance, shared-radius, and handoff rows before any finite-net-facing read begins.

The support rows T, R, P, N only transport, replay, source, and name the displayed BHist route. Since no memberwise radius row is present, the root carrier exposes the shared ρ boundary alone. □

Theorem 3096.13 (Equicontinuity compact-modulus consumer root row)**.** *Let $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ be an accepted* Equicontinuity$^{\uparrow}$ *carrier. A compact-modulus consumer must first read K as the* CompactMetric$^{\uparrow}$ *source, then F as the finite* ContinuousMap$^{\uparrow}$ *ledger over that source, then ϵ, ρ, M as the shared* UniformModulus$^{\uparrow}$ *handoff, with the route preserved by* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The consumer cannot treat M as detached from the compact source or as a selected finite net.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockCompactModulusConsumer`

Proof. Read K and F from the accepted carrier. The compact source row precedes the continuous-family ledger, and the family is displayed over that same source.

Use compact-source admission and finite-net readiness, Definition 4076.9 and Definition 4076.16. They order the compact and family reads before ϵ, ρ, M can be consumed by a uniform-modulus or finite-net-facing reader.

Finally apply nonescape. The support rows preserve the displayed route, while selected finite nets, detached compactness output, subsequences, and convergence rows are absent from the Equicontinuity$^{\uparrow}$ packet. □

Theorem 3096.14 (Equicontinuity finite-net handoff root row)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the finite-net handoff is the ordered* BHist *route from the* CompactMetric$^{\uparrow}$ *source K through the* ContinuousMap$^{\uparrow}$ *family F and the shared tolerance-radius rows ϵ, ρ, M into the* FiniteNetEquicontinuity$^{\uparrow}$ *consumer boundary. The* hsame*,* Cont*,* Pkg*, and* NameCert *rows T, R, P, N preserve that route and do not add compactness output, selected nets, selected subsequences, or an Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetHandoff`

Proof. Project the rows $K, F, \epsilon, \rho, M, T, R, P, N$. The analytic part of the packet already orders the compact source, continuous family, tolerance request, shared radius, and modulus handoff.

Apply the shared-radius carrier row above and the compact-modulus consumer row above. Together they make the finite-net read pass through the same compact source and shared radius before the downstream FiniteNetEquicontinuity$^{\uparrow}$ boundary is available.

The structural rows transport and replay only this displayed handoff. Because finite nets and convergence objects are not carrier coordinates, the handoff remains a root boundary rather than a selected-net construction. □

Theorem 3096.15 (Equicontinuity shared-radius obligation)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the local* NameCert$_{\text{Equicontinuity}^{\uparrow}}$ *obligation requires the single radius row ρ to serve every displayed member of F at tolerance ϵ before any* FiniteNetEquicontinuity$^{\uparrow}$ *or Arzela–Ascoli consumer may read the packet. No consumer may replace ρ by memberwise radii, a pointwise-only modulus family, or a selected finite-net radius outside the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier shared radius obligation`

Proof. Project the carrier of Definition 4076.1. The family row F is finite and displayed over the compact metric source K, while ρ appears as one row between the tolerance ϵ and the modulus handoff M.

Use the compact-metric route of Definition 4076.3. It forces consumers to read K, F, ϵ, ρ, M before transport, replay, provenance, and local naming can expose the packet.

Apply shared-radius stability and finite-net readiness, Definition 4076.8 and Definition 4076.16. Together they keep ρ stable under transport and make it available before any finite-net or Arzela–Ascoli-facing read, so pointwise-radius leakage is outside the accepted carrier. □

Theorem 3096.16 (Equicontinuity finite-net readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, a* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer may begin only after the compact source* K*, the continuous family ledger* F*, the tolerance row* ϵ*, the shared radius row* ρ*, and the modulus handoff* M *have all been exposed. The rows* T, R, P, N *preserve this order and do not themselves select a finite net.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetReadiness`

Proof. Read the carrier in the order stated by Definition 4076.3. The compact source K is the only source row, and the family ledger F is the only row naming the continuous maps over that source.

The finite-net consumer then reads ϵ, ρ, M. These rows record a tolerance, a shared radius, and the handoff to the uniform-modulus surface before any finite-net selector can inspect the packet.

The structural rows T, R, P, N transport, replay, source, and name the same displayed route. Since Definition 4076.1 has no finite-net coordinate, readiness for $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ is a boundary condition, not a selected net. □

Theorem 3096.17 (Equicontinuity family-modulus consumer exhaustion)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, every compact-uniform modulus consumer of* E *must read exactly the rows* $K, F, \epsilon, \rho, M, T, R, P, N$*. The consumer may use* K *as the compact source,* F *as the displayed family ledger,* ϵ *as the tolerance request,* ρ *as the shared radius,* M *as the modulus handoff, and* T, R, P, N *as the structural transport, replay, provenance, and naming rows. It cannot detach a pointwise radius schedule from* F*, replace* ρ *by memberwise radii, or insert a selected finite-net row into the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier family modulus consumer exhaustion`

Proof. Start from shared-radius stability in Definition 4076.8. It pins ρ to the displayed compact source K, family ledger F, tolerance row ϵ, and modulus handoff M.

Use finite-net readiness from Definition 4076.16. A finite-net-facing or compact-uniform consumer may begin only after K, F, ϵ, ρ, M have been exposed, while T, R, P, N preserve that same route.

Finally apply nonescape from Definition 4076.4. Since the carrier has no coordinate for memberwise radius schedules, selected finite nets, subsequences, host continuity, or ambient completeness, the rows $K, F, \epsilon, \rho, M, T, R, P, N$ exhaust the compact-uniform consumer surface. □

Theorem 3096.18 (Equicontinuity finite-net selector readiness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier. A* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *selector may start only after the compact-source row* K*, continuous-family row* F*, shared-radius row* ρ*, and modulus-handoff row* M *are exposed for the same tolerance request* ϵ*. The structural rows* T, R, P, N *may transport and replay that readiness boundary, but they do not authorize a selected finite net before the compact source, family, shared radius, and modulus handoff have all been read.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetSelectorReadiness`

Proof. Read the prerequisite order from Definition 4076.16. It places a finite-net consumer after the compact source, continuous family ledger, tolerance request, shared radius, and modulus handoff.

Apply shared-radius stability from Definition 4076.8. The selector cannot split the family into detached pointwise radius schedules, because the radius row ρ is transported with K, F, ϵ, M.

Use nonescape from Definition 4076.4. Selected nets, subsequences, limiting maps, convergence certificates, host continuity, and ambient completeness are absent from the carrier, so the selector receives only readiness at the displayed boundary. □

Theorem 3096.19 (Equicontinuity shared-modulus sibling route)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, any downstream consumer of the equicontinuity packet must read the compact source through chapter 129, the continuous family through chapter 130, the uniform modulus handoff through chapter 369, and the finite-net boundary through chapter 2362. The route preserves the same shared radius* ρ *and exports no selected finite net, subsequence, host continuity theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedModulusScope`

Proof. Project the carrier of Definition 4076.1. The row K is the compact source, F is the finite continuous-family ledger over that source, and ϵ, ρ, M are the shared tolerance-radius and modulus handoff rows.

Use the compact-source admission and continuous-family admission theorems, Definition 4076.25 and Definition 4076.26. They identify the source and family reads with the sibling surfaces named by chapter 129 and chapter 130.

Apply shared-radius stability, modulus-handoff admission, and finite-net readiness, Definition 4076.8, Definition 4076.27, and Definition 4076.16. These rows keep the same ρ attached to the handoff into the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ consumers.

Finally use Definition 4076.4. Since the selected-net, subsequence, host-continuity, and ambient-completeness coordinates are absent from the carrier, the sibling route just listed is the whole public dependency surface. □

Theorem 3096.20 (Equicontinuity modulus-ledger exactness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be accepted. The modulus ledger exported by* $\mathsf{Equicontinuity}^{\uparrow}$ *is exactly the handoff from the compact source* K *and finite family* F *at tolerance* ϵ *to the shared radius* ρ *recorded by* M*, transported by* T*, replayed by* R*, sourced by* P*, and named by* N*. It has no additional ledger row for host continuity, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected covers, or an Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier modulus ledger exactness`

Proof. The analytic part of the carrier is K, F, ϵ, ρ, M. The compact-metric row supplies the source, the continuous-map row supplies the finite family, and the uniform-modulus row records the tolerance-to-radius handoff.

The ledger is exact because T, R, P, N refer back to those same rows. They cannot add a second modulus, a selected cover, a limit candidate, or a host-level continuity predicate; they only preserve and name the displayed handoff.

The exclusions are also forced by Definition 4076.4. Any consumer needing compactness output, selected nets, subsequences, or completeness must obtain those data from a separate chapter after reading the exact modulus ledger here. □

Definition 3096.21 (Equicontinuity finite-modulus subcarrier)**.** *An* $\mathsf{Equicontinuity}^{\uparrow}$ *finite-modulus subcarrier is the finite row packet*

$$E_{\text{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$$

projected from an accepted carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*. The rows are, in order, the compact metric source* K*, finite continuous-family ledger* F*, shared tolerance request* ϵ*, shared radius row* ρ*, uniform-modulus handoff* M*, finite-net consumer boundary* G*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. The boundary row* G *is readable only as readiness for* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *over the displayed* K, F, ϵ, ρ, M *route; it is not a selected net, subsequence, compactness conclusion, or Arzela–Ascoli limit.*

Theorem 3096.22 (Equicontinuity compact-metric finite-modulus handoff)**.** *Every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *finite-modulus subcarrier* $E_{\text{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$ *hands downstream consumers a compact-source route only after* K *has been read as the compact metric source and before* F, ϵ, ρ, M, G *are inspected. The rows* T, R, P, N *preserve this compact-source handoff and cannot replace it by an ambient compactness assertion or selected finite net.*

Proof. Project E_{fin} from Definition 4076.21. Its first row is the same compact source K already displayed in Definition 4076.1.

Apply Definition 4076.25 and Definition 4076.3. They force every continuous-family, uniform-modulus, and finite-net-facing read to pass through K before F, ϵ, ρ, M are consumed.

The boundary row G is therefore only a consumer boundary for the existing route. Transport, replay, provenance, and naming through T, R, P, N preserve that route and add no selected net or ambient compactness coordinate. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteModulusCarrier compactmetric handoff`

Theorem 3096.23 (Equicontinuity continuous-map finite-modulus route)**.** *For every accepted finite-modulus subcarrier* $E_{\text{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$*, the family row* F *is read only as a finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger over the already displayed compact source* K*. The shared tolerance request* ϵ*, radius row* ρ*, and modulus handoff* M *are attached to that same ledger, so no consumer may detach the modulus route into host-level continuity, a source-changing family, or memberwise radii.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousMapModulusRoute`

Proof. Read the finite family row F after the compact source K. By Definition 4076.21, these are the same rows projected from the accepted $\mathsf{Equicontinuity}^{\uparrow}$ carrier.

Apply Definition 4076.26 and Definition 4076.32. They identify F as the finite $\mathsf{ContinuousMap}^{\uparrow}$ ledger over K and exclude host continuity, source-changing maps, and infinite-family replacement.

Use Definition 4076.8. The shared radius row ρ is transported with K, F, ϵ, M, so the modulus route remains attached to the same finite family ledger. □

Theorem 3096.24 (Equicontinuity uniform-modulus consumer boundary)**.** *For every accepted finite-modulus subcarrier* $E_{\text{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$*, a* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$*, or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer may read the boundary* G *only after the shared handoff*

$$K, F, \epsilon, \rho, M$$

has been displayed. The boundary exports readiness for the finite-net consumer and no selected finite net, subsequence, compactness proof, Arzela–Ascoli limit, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier uniform modulus consumer boundary`

Proof. Project the handoff rows K, F, ϵ, ρ, M and the boundary row G from Definition 4076.21. The row G appears downstream of the shared tolerance-radius handoff.

Apply Definition 4076.16 and Definition 4076.33. They require every finite-net-facing consumer to preserve the compact source, family ledger, tolerance request, shared radius, and modulus handoff before reading any consumer boundary.

Finally use Definition 4076.20 and Definition 4076.4. The displayed handoff is the whole exported modulus ledger, and the excluded selected nets, subsequences, compactness conclusions, limits, quotient equalities, and completeness principles are absent from the packet. □

Theorem 3096.25 (Equicontinuity compact-source admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the compact source row* K *is admitted through* $\mathsf{CompactMetric}^{\uparrow}$ *before any finite-net, shared-radius, or modulus consumer may inspect* F, ϵ, ρ, M*. No downstream* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*, or Arzela–Ascoli-facing read can bypass* K *or replace it by an ambient compactness assertion.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactSourceAdmission`

Proof. Project the carrier of Definition 4076.1. The row K is the only compact-source coordinate, and the introduction fixes it as the same compact metric surface used by chapter 129.

Use the compact-metric route and finite-net readiness theorems, Definition 4076.3 and Definition 4076.16. They order every consumer read through K before the family ledger, tolerance, shared radius, and handoff rows.

The nonescape theorem then excludes an ambient compactness replacement. Since selected finite nets and compactness conclusions are absent from the carrier, the admitted compact source is precisely the displayed $\mathsf{CompactMetric}^{\uparrow}$ row. □

Theorem 3096.26 (Equicontinuity continuous-family admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family row* F *is a finite ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus rows over the same source* K*. Transport through* T *and replay through* R *preserve this common source, so a consumer cannot read the family as host-level continuity, source-changing maps, or pointwise-only radius data detached from* K*.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousFamilyAdmission`

Proof. Read F from Definition 4076.1. Its entries are continuous-map rows over the compact source row K, matching the continuous-map sibling cited in the chapter introduction.

Apply shared-radius stability. The theorem keeps ρ attached to the transported compact source, finite family, tolerance, and modulus handoff, so F cannot be split into unrelated pointwise-radius ledgers.

Finally apply nonescape. Host-level continuity and source-changing maps are not coordinates of E, and T, R, P, N only transport, replay, source, and name the displayed family ledger. □

Theorem 3096.27 (Equicontinuity modulus-handoff admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared tolerance-radius row* ϵ, ρ, M *is the only* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *handoff available to downstream finite-net consumers. A consumer that reads* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *must receive the handoff through this row and cannot add a second modulus ledger, a memberwise radius schedule, or a selected finite-net radius outside the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityModulusHandoffAdmission`

Proof. Project the handoff rows ϵ, ρ, M. The carrier displays one tolerance, one shared radius, and one modulus consumer row.

Use the shared-radius obligation and finite-net readiness theorems. They force every finite-net or Arzela–Ascoli-facing consumer to read ϵ, ρ, M after K, F and before any finite-net action can begin.

The modulus-ledger exactness theorem identifies this handoff as the whole exported modulus ledger. Since there is no coordinate for a second modulus, a memberwise radius family, or an outside selected radius, the downstream read is confined to the displayed $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ handoff. □

Theorem 3096.28 (Equicontinuity compact-source root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root source lock is the compact metric row* K *together with the transported route* T, R, P, N*. Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*, or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer must read* K *before the family ledger* F*, tolerance* ϵ*, shared radius* ρ*, and modulus handoff* M *become available. The lock exports no selected finite net, subsequence, or Arzela–Ascoli limit.*

Proof. Project the carrier. The only source row is K, and the rows F, ϵ, ρ, M are displayed downstream of that source in the finite packet.

Apply compact-source admission and the compact-metric route. These place every continuous-family, uniform-modulus, and finite-net-facing read after K and before the structural rows T, R, P, N can expose the named packet.

Finally use nonescape. Since the carrier has no selected-net, subsequence, or convergence-limit coordinate, the root lock is exactly the displayed compact-source route. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootCompactSourceLock`

Theorem 3096.29 (Equicontinuity family-ledger root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root family ledger is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *row* F *over the already-read compact source* K*. Transport, replay, provenance, and local naming through* T, R, P, N *preserve that ledger before* ϵ, ρ, M *are consumed by* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$*. No consumer may replace* F *by an infinite family, host-level continuity, selected finite net, subsequence, or Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootFamilyLedgerLock`

Proof. Read K and F from the carrier. The row F is finite and its entries are continuous-map rows over the same compact source.

Use continuous-family admission and continuous-family ledger admission. They force the consumer route through the displayed $\mathsf{ContinuousMap}^{\uparrow}$ ledger rather than through host continuity or a source-changing family.

The rows T, R, P, N only preserve, replay, source, and name the same ledger. Nonescape excludes the listed outside objects, so the root lock is the finite family ledger itself. □

Theorem 3096.30 (Equicontinuity root family modulus ledger)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root family-modulus ledger exposed to finite-net consumers is exactly the compact-source row* K*, finite continuous-family row* F*, shared tolerance-radius rows* ϵ, ρ*, modulus handoff* M*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. No consumer may replace this ledger by selected nets, selected subsequences, host compactness, host continuity, a second modulus ledger, or Arzela–Ascoli convergence.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootFamilyModulusLedger`

Proof. Project the carrier. The compact-source lock and family-ledger root lock identify K and F as the only source and family rows available before any modulus-facing read.

Apply shared-radius stability, modulus-handoff admission, and modulus-ledger exactness. These place ϵ, ρ, M after K, F and identify the shared handoff as the whole exported modulus ledger for finite-net consumers.

Finally use transport replay provenance and nonescape. The rows T, R, P, N preserve the displayed ledger without adding a selected net, subsequence, host compactness proof, host continuity theorem, second modulus ledger, or Arzela–Ascoli convergence row. □

Theorem 3096.31 (Equicontinuity modulus-consumer root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root modulus-consumer lock is the shared handoff* ϵ, ρ, M *after the compact source* K *and family ledger* F *have been displayed. Any* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer must preserve this handoff through* T, R, P, N *and cannot add a selected finite net, memberwise radius schedule, subsequence, or Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootModulusConsumerLock`

Proof. Project the handoff rows. The packet contains one tolerance row, one shared radius row, and one modulus consumer row.

Apply modulus-handoff admission and shared-modulus handoff preservation. These theorems make ϵ, ρ, M the only modulus handoff available after K and F have been read.

Since the carrier has no coordinate for a selected finite net, memberwise radii, subsequences, or Arzela–Ascoli limits, transport and replay through T, R, P, N preserve only the displayed shared-modulus consumer lock. □

Theorem 3096.32 (Equicontinuity continuous-family ledger admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family row* F *is admitted only as a finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger over the compact source* K*. No consumer may reread* F *as host-level continuity, a source-changing family, or an infinite family outside the displayed compact-source route.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousFamilyLedgerAdmission`

Proof. Project the carrier of Definition 4076.1. It displays K as the compact metric source and F as the finite continuous-family row over that same source.

Use the compact-metric route and continuous-family admission theorems, Definition 4076.3 and Definition 4076.26. They force every consumer to read K before F and keep the family inside the displayed $\mathsf{ContinuousMap}^{\uparrow}$ ledger.

Finally apply Definition 4076.4. The excluded host-continuity, source-changing, and infinite-family coordinates are absent from E. □

Theorem 3096.33 (Equicontinuity shared-modulus handoff preservation)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, any* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer must preserve the shared tolerance-radius rows* ϵ, ρ, M *before reading a finite-net selector. The consumer cannot replace them by memberwise radii, a second modulus ledger, or a selected net radius outside the packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier shared modulus handoff preservation`

Proof. Read the shared-radius stability and finite-net readiness rows, Definition 4076.8 and Definition 4076.16. They place ϵ, ρ, M after the compact source and family rows, and before any finite-net-facing read.

Apply modulus-handoff admission and modulus-ledger exactness, Definition 4076.27 and Definition 4076.20. These identify the shared tolerance-radius handoff as the whole exported modulus ledger.

Since the carrier has no selected-net coordinate, preservation of ϵ, ρ, M is forced before a finite-net selector can be read. □

Theorem 3096.34 (Equicontinuity transport replay provenance)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the transport row* T*, replay row* R*, provenance row* P*, and naming row* N *preserve only the displayed compact-source, continuous-family, tolerance-radius, and modulus-handoff rows. They cannot add pointwise radius choice, selected nets, subsequences, or Real completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier transport replay provenance`

Proof. Project T, R, P, N from Definition 4076.1. Their only analytic inputs are K, F, ϵ, ρ, M.

Use shared-radius stability and modulus-ledger exactness. Transport and replay preserve the same shared radius row and the same modulus ledger rather than a pointwise-radius family.

The nonescape theorem excludes selected nets, subsequences, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Provenance and local naming therefore record the displayed route only. □

Theorem 3096.35 (Equicontinuity shared-radius family-cover readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared-radius family-cover boundary is the displayed route*

$$K, F, \epsilon \longmapsto \rho, M \longmapsto T, R, P, N.$$

The row ρ *is readable only after the compact source* K*, finite continuous-family ledger* F*, and tolerance row* ϵ *have been exposed, and every family-cover consumer must keep* M, T, R, P, N *visible. The route cannot be replaced by selected nets, selected subsequences, host continuity, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusFamilyCoverReadiness`

Proof. Project the carrier. The rows K, F, ϵ are the only source, family, and tolerance data available before the shared radius is read.

Use shared-radius stability and shared-modulus handoff preservation. These results place ρ and M after the displayed source-family-tolerance rows and before structural transport and replay.

Finally apply transport replay provenance and nonescape. The rows T, R, P, N preserve the same boundary, and the carrier lacks coordinates for selected nets, selected subsequences, host continuity, and ambient real completeness. □

Theorem 3096.36 (Equicontinuity compact family-net readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, a finite-net consumer may read the family only through the compact-source route*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The compact family-net boundary consists of the compact metric source K*, the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row* F*, the tolerance-radius handoff* ϵ, ρ, M*, and the transport, replay, provenance, and naming rows* T, R, P, N*. It does not choose a net, choose a subsequence, supply host continuity, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactFamilyNetReadiness`

Proof. Start with continuous-family ledger admission. It admits F only as a finite $\mathsf{ContinuousMap}^{\uparrow}$ ledger over the already displayed compact source K.

Next use finite-net readiness and modulus-consumer root lock. They force any finite-net-facing read to pass through ϵ, ρ, M before the structural rows are replayed.

The rows T, R, P, N do not add a selector. By transport replay provenance and nonescape, the accepted family-net read remains a compact-source ledger route, excluding selected nets, selected subsequences, host continuity, and ambient real completeness. □

Theorem 3096.37 (Equicontinuity Arzela–Ascoli boundary readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, an Arzela–Ascoli-facing consumer may cite* $\mathsf{Equicontinuity}^{\uparrow}$ *only as the displayed boundary*

$$K, F, \epsilon, \rho, M, T, R, P, N.$$

This boundary supplies compact-source admission, finite continuous-family admission, shared tolerance-radius handoff, modulus-consumer readiness, transport, replay, provenance, and local naming. It does not supply a selected subsequence, a selected limiting map, host continuity of the limit, selected nets, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityArzelaAscoliBoundaryReadiness`

Proof. Read the root family ledger and modulus-consumer root lock. Together they make K, F, ϵ, ρ, M the whole analytic boundary exposed by the packet.

Use compact family-net readiness. Any later compactness-facing consumer receives only the routed family-net boundary and not a selected net or subsequence.

Finally replay transport, provenance, and nonescape through T, R, P, N. Since no coordinate carries a limiting map, host continuity proof, selected subsequence, selected net, or ambient real-completeness proof, an Arzela–Ascoli-facing consumer can cite only the displayed boundary. □

Theorem 3096.38 (Equicontinuity compact-family total-bounded consumer)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *whose downstream read also displays the* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *boundary over the same compact source, the packet exports exactly the finite total-bounded consumer package needed by Arzela–Ascoli-style chapters:*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The package ties the $\mathsf{CompactMetric}^{\uparrow}$ *source row, finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, finite-net equicontinuity boundary,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal to the same* BHist, hsame, Cont, Pkg, *and* NameCert *route. It refuses selected subsequences, open-cover compactness, choice, selected limiting maps, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactFamilyTotalBoundedConsumer`

Proof. Use compact family-net readiness and Arzela–Ascoli boundary readiness. They make the compact source K, finite family F, tolerance ϵ, shared radius ρ, and modulus handoff M the whole analytic boundary visible to downstream consumers.

Apply shared-modulus handoff preservation and finite-net readiness. These force the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ reads to preserve the same shared-radius route rather than replacing it by memberwise radii or an outside finite-net selector.

Finally use transport replay provenance and nonescape. The rows T, R, P, N carry the same finite route through BHist, hsame, Cont, Pkg, and NameCert, while the carrier has no coordinates for selected subsequences, open-cover compactness, choice, limiting maps, or ambient real completeness. □

Theorem 3096.39 (Equicontinuity compact-source root row)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the compact source row* K *is the first root row that finite-net and Arzela–Ascoli consumers must read. The rows* $F, \epsilon, \rho, M, T, R, P, N$ *are available only as rows over that same* $\mathsf{CompactMetric}^{\uparrow}$ *source, and no downstream net, subsequence, or convergence consumer can bypass* K *by reading the shared-radius or finite-net boundary directly.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockCompactSourceRow`

Proof. Project the carrier of Definition 4076.1. The row K is the only compact metric source row, while $F, \epsilon, \rho, M, T, R, P, N$ are displayed after it.

Use compact-source root admission and compact-metric route, Definition 4076.9 and Definition 4076.3. They force the family ledger, tolerance row, shared radius, and modulus handoff to remain attached to K.

Finally apply finite-net readiness and Arzela–Ascoli boundary readiness. Both consumers must read the displayed compact-source route before any downstream finite net, subsequence, or convergence object is available. □

Theorem 3096.40 (Equicontinuity shared-radius root handoff)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared radius row* ρ *is a root handoff between the compact-source family rows* K, F, ϵ *and the finite-net or uniform-modulus consumer row* M*. The transport, replay, provenance, and naming rows* T, R, P, N *preserve this same* ρ*; they do not create memberwise radii, a detached pointwise modulus, or a selected finite-net radius.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedRadiusHandoff`

Proof. Read K, F, ϵ, ρ, M from the carrier. The single row ρ appears after the tolerance request and before the modulus handoff.

Apply shared-radius stability and shared-modulus handoff preservation, Definition 4076.8 and Definition 4076.33. These results keep the same ρ visible through every $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ read.

The structural rows T, R, P, N preserve the displayed route only. Nonescape then excludes memberwise radii, pointwise-only modulus data, and selected finite-net radius rows. □

Theorem 3096.41 (Equicontinuity finite-net root boundary)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the finite-net boundary is the ordered read of* K, F, ϵ, ρ, M *followed by* T, R, P, N*. A* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *or Arzela–Ascoli consumer may start only after the compact source and shared radius have been exposed, and the consumer must obtain any actual*

finite net, subsequence, limiting map, or convergence certificate from a downstream packet rather than from $\mathsf{Equicontinuity}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetBoundary`

Proof. Project the displayed rows. The carrier contains the compact source K, finite family F, tolerance ϵ, shared radius ρ, modulus handoff M, and the support rows T, R, P, N.

Use finite-net readiness and compact family-net readiness, Definition 4076.16 and Definition 4076.36. They place every finite-net-facing read after the compact-source and shared-radius route.

Finally apply the finite-net consumer boundary and nonescape results, Definition 4076.11 and Definition 4076.4. The downstream consumer may cite the boundary, but selected nets, subsequences, limiting maps, and convergence certificates are not rows of the $\mathsf{Equicontinuity}^{\uparrow}$ carrier. □

Theorem 3096.42 (Equicontinuity carrier root obligation)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, root admission requires the finite* BHist *rows for a family of* $\mathsf{ContinuousMap}^{\uparrow}$ *maps over the displayed source row* K*. The source row may be read as* $\mathsf{CompactMetric}^{\uparrow}$ *when finite net consumers are present, or as the underlying* $\mathsf{Met}^{\uparrow}$ *row for metric distance comparisons, but the family row* F *is admitted only as a finite continuous-map ledger over that same source. No host open-cover compactness, selected subcover, quotient real equality, or choice principle is a carrier row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationCarrier`

Proof. Project the carrier and the compact-source root row. The source coordinate K is displayed before the family row F, and the chapter reads all finite-net and Arzela–Ascoli-facing material through that source.

Use the $\mathsf{ContinuousMap}^{\uparrow}$ family surface over the same source. The finite ledger F is a BEDC row over K; it is not a host-indexed family and it does not carry a selected subcover or open-cover compactness proof.

Finally replay the metric and real readbacks through T, R, P, N. The $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows are visible only as routed support for the displayed family, so quotient real equality and choice cannot enter the carrier admission step. □

Theorem 3096.43 (Equicontinuity finite-net locality root obligation)**.** *Every finite-net read exported by an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *routes through the displayed* $\mathsf{Met}^{\uparrow}$ *distance row, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, and the* $\mathsf{Real}^{\uparrow}$ *tolerance rows already sealed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *request ledger. A finite-net consumer may cite this local route only after* K, F, ϵ, ρ, M *have been displayed; it cannot replace the route by host open-cover compactness, a selected subcover, quotient real equality, or choice.*

Proof. Read the compact-source and shared-radius root handoff first. They place the metric source, family ledger, tolerance row, shared radius, and modulus handoff in a fixed order before any finite-net consumer can start.

Next use finite-net readiness. The finite-net surface is local to the displayed $\mathsf{CompactMetric}^{\uparrow}$ row and its underlying $\mathsf{Met}^{\uparrow}$ comparisons; the $\mathsf{Real}^{\uparrow}$ tolerance row is read through the same regular rational and stream-name ledger.

The remaining rows provide transport, replay, provenance, and local naming. Since those rows preserve only the displayed route, no open-cover compactness argument, selected subcover, quotient real equality, or choice principle can replace the finite-net locality obligation. □

Lean4 ✓ BEDC.Derived.EquicontinuityUp.RootObligationFiniteNet.EquicontinuityRootObligationFiniteNet

Theorem 3096.44 (Equicontinuity uniform-modulus root obligation)**.** *The exported* UniformModulus$^{\uparrow}$ *row M of an accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ is available only after the equicontinuity family row F and the finite-net ledger over K have been displayed. The handoff consumes the same* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* StreamName$^{\uparrow}$ *rows as the preceding locality route, and it refuses host open-cover compactness, selected subcovers, quotient real equality, and choice.*

Lean4 ✓ BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationUniformModulus

Proof. Begin with the family row and finite-net locality obligation. They expose the source K, family ledger F, tolerance ϵ, and shared radius ρ before any uniform-modulus consumer may read M.

Use shared-modulus handoff preservation. The UniformModulus$^{\uparrow}$ row is a handoff from the displayed equicontinuity family and finite-net ledger, not a free-standing modulus selected outside the packet.

Finally apply transport replay and nonescape. The rows T, R, P, N preserve the same CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and StreamName$^{\uparrow}$ route. Therefore the uniform-modulus handoff cannot be supplied by host compactness, a selected subcover, quotient real equality, or choice. □

Theorem 3096.45 (Equicontinuity source-family root obligation)**.** *Every accepted* Equicontinuity$^{\uparrow}$ *read exposes the* CompactMetric$^{\uparrow}$ *source row K and the* ContinuousMap$^{\uparrow}$ *family row F before the tolerance ϵ, shared radius ρ, or modulus handoff M can be consumed. The read is therefore rooted in the displayed source-family packet of Definition 4076.1, not in an ambient compact space or an unlisted family selector.*

Lean4 ✓ BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationSourceFamily

Proof. Read the carrier of Definition 4076.1. Its first two rows are the compact metric source K and the finite continuous-family ledger F, and Definition 4076.2 makes those rows part of the local NameCert surface.

Next use the compact metric route of Definition 4076.3. It sends every later tolerance, radius, and modulus read through the displayed K, F source-family route.

Finally apply Definition 4076.4. A consumer that asks for ϵ, ρ, M before the source and family rows has left the accepted packet rather than extracting a hidden compact-source or family coordinate. □

Theorem 3096.46 (Equicontinuity shared-radius root obligation)**.** *The radius row ρ of an accepted* Equicontinuity$^{\uparrow}$ *carrier is a single shared radius row transported by the displayed* hsame *and* Cont *routes. It is not a memberwise selector indexed by the family row F, and it cannot be replaced by a choice schedule over individual maps.*

Proof. Begin with the carrier and NameCert obligations of Definition 4076.1 and Definition 4076.2. They display one row ρ between the tolerance row and the modulus handoff row.

Use the compact metric route of Definition 4076.3. The row ρ is transported through the same hsame and Cont replay rows that preserve K, F, ϵ, M.

Apply nonescape by Definition 4076.4. Since the accepted packet has no coordinate for a family-indexed selector, a memberwise radius choice cannot be recovered from the shared-radius row. □

Theorem 3096.47 (Equicontinuity downstream nonescape root obligation)**.** *The* Equicontinuity$^{\uparrow}$ *root surface contains no finite-net selector, no subsequence selector, no Arzela–Ascoli convergence row, no host-continuity oracle, and no ambient* Real$^{\uparrow}$ *completeness coordinate. Those rows must be supplied by downstream chapters before a consumer may read compact-uniform consequences from an accepted equicontinuity packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationNonescape`

Proof. Project the displayed rows of Definition 4076.1. The local surface records $K, F, \epsilon, \rho, M, T, R, P, N$ and the NameCert obligations of Definition 4076.2; it does not add selector, subsequence, convergence, host-continuity, or completeness rows.

Next use the compact metric route of Definition 4076.3. It keeps compact-source information visible as a finite route, while refusing to turn that route into open-cover compactness, selected finite nets, or selected subsequences.

Finally apply Definition 4076.4. Any Arzela–Ascoli, host-continuity, or real-completeness read must therefore enter through a downstream certificate rather than through the root Equicontinuity$^{\uparrow}$ packet. □

Theorem 3096.48 (Equicontinuity shared-radius family nonescape)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared radius* ρ *is read over the same* CompactMetric$^{\uparrow}$ *source row* K *and* ContinuousMap$^{\uparrow}$ *family row* F *through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. A consumer cannot split* ρ *into memberwise radii, detach it from the* UniformModulus$^{\uparrow}$ *handoff* M*, or read it outside the finite family packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusFamilyNonescape`

Proof. Project the carrier and use the shared-radius root obligation. The rows K, F, ϵ, ρ, M occur in one displayed packet, so the radius row is already tied to the compact source and continuous-family ledger.

Apply shared-modulus handoff preservation. It makes the UniformModulus$^{\uparrow}$ consumer preserve the same ϵ, ρ, M rows rather than replacing them by a memberwise or selected finite-net radius.

Finally use transport replay provenance and nonescape. The rows T, R, P, N only preserve the displayed BHist, hsame, Cont, Pkg, and NameCert route, so no extra family selector can escape the shared-radius packet. □

Theorem 3096.49 (Equicontinuity compact-net modulus readiness)**.** *Every* FiniteNetEquicontinuity$^{\uparrow}$ *consumer of an accepted* Equicontinuity$^{\uparrow}$ *packet must first read the* CompactMetric$^{\uparrow}$ *source* K*, the finite* ContinuousMap$^{\uparrow}$ *family ledger* F*, the shared tolerance-radius rows* ϵ, ρ*, and the* UniformModulus$^{\uparrow}$ *handoff* M *through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface. The finite-net handoff is readiness data only; it does not choose a net or assert open-cover compactness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactNetModulusReadiness`

Proof. Begin with finite-net readiness and compact family-net readiness. Those results place any finite-net-facing read after the compact-source and continuous-family rows and before any downstream selector can be consumed.

Use the uniform-modulus root obligation to keep M attached to ϵ, ρ over the same K, F route. The modulus handoff therefore arrives as part of the finite packet, not as a second compactness principle.

The structural rows T, R, P, N then replay only the displayed route. Nonescape excludes selected nets, open-cover compactness, host continuity, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the readiness surface. □

Theorem 3096.50 (Equicontinuity Arzela–Ascoli consumer route)**.** *An Arzela–Ascoli-facing consumer may cite* $\mathsf{Equicontinuity}^{\uparrow}$ *only as the routed packet* $K, F, \epsilon, \rho, M, T, R, P, N$*:* $\mathsf{CompactMetric}^{\uparrow}$ *source data, finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family data, shared* $\mathsf{UniformModulus}^{\uparrow}$ *readiness, finite-net handoff, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The consumer must supply its own subsequence, limiting map, convergence, and Real-facing certificate rows.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityArzelaAscoliConsumerRoute`

Proof. Use the Arzela–Ascoli boundary readiness theorem. It presents $\mathsf{Equicontinuity}^{\uparrow}$ as a boundary over compact-source, continuous-family, shared-radius, and modulus-handoff rows.

Next compose compact-net modulus readiness with the finite-net root boundary. This gives the downstream consumer a finite-net-ready route, but not the selected net, subsequence, limiting map, or convergence certificate.

Finally apply downstream nonescape. Since the carrier has no ambient completeness or host-continuity coordinate, every Arzela–Ascoli conclusion must be supplied by a later packet over the same displayed structural rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Equicontinuity}^{\uparrow}$ packet is generated from a compact metric source row, a finite continuous-family ledger, shared tolerance and radius rows, a $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ or $\mathsf{UniformModulus}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 4076.1, Definition 4076.2, Definition 4076.3, Definition 4076.4, Definition 4076.8, Definition 4076.15, Definition 4076.16, and Definition 4076.20 export the public finite-net and Arzela–Ascoli-facing handoff through the checked root-obligation target over compact-source, continuous-family, tolerance, shared-radius, modulus-handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationSourceFamily`

Bridge status: none

Not claimed: This chapter does not close compactness, selected finite nets, selected subsequences, Arzela–Ascoli convergence, host-level continuity, pointwise-only radius choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{Equicontinuity}^{\uparrow}$.

Upgrade path: The remaining work is the standard bridge check for the same public finite-net and Arzela–Ascoli-facing handoff, not additional paper closure.

Chapter 3097

A Concrete Naming Certificate for $\mathsf{ChoquetGame}^{\uparrow}$

Following Choquet and Oxtoby on topological games and Baire category, the $\mathsf{ChoquetGame}^{\uparrow}$ packet records the finite nested-open game route used by complete-metric Baire arguments. It sits over the Baire prefix of chapter 334, the dense-open and category surface of chapter 1542, the complete-metric handoff of chapter 128, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the standard Choquet-game interface as BEDC NameCert data without importing a chosen winning strategy, a host intersection point, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3097.1 (Choquet-game carrier)**.** *A* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, M, L, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row naming the inspected basic-open source, finite prefix probes, and legal initial open row.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *metric-ball row, including displayed centers, positive rational radii, inclusion checks, and shrinking-radius obligations for successor moves.*
3. *L is the nested legal-move ledger, recording each finite play prefix, the predecessor ball, the successor ball, the player tag, and the proof that the successor remains inside the predecessor.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite play prefixes and ball observations visible to the consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius budgets, and tolerance comparisons used by the nested ball thread.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Baire prefix, complete-metric ball data, legal-move ledger, stream schedule, and regular rational readback have been displayed.*
7. *T records componentwise* hsame *transport for Baire prefixes, complete-metric ball rows, legal-move ledgers, stream schedules, regular rational readback, Real seals, replay data, provenance, and local naming.*

8. *C records displayed* Cont *replay from a game request to the Baire prefix, complete-metric ball route, nested legal-move ledger, finite schedule, rational readback, Real seal, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ChoquetGame}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global winning strategy, a selected limit point, countable choice, quotient equality of streams, ambient completed-real principles, or a hidden dense-open oracle.

Lean4 ✓ `BEDC.Derived.ChoquetGameUp.ChoquetGameTasteGate single carrier alignment`

Theorem 3097.2 (Choquet-game NameCert obligations)**.** *For every accepted* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier* $G = (B, M, L, S, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Baire prefix, complete-metric ball, legal-move ledger, stream schedule, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *Baire-prefix coverage, requiring every play prefix to begin with the* $\mathsf{BaireSpace}^{\uparrow}$ *row B;*
3. *metric-ball legality, requiring M to supply displayed centers, radii, inclusions, and shrinking obligations before L records a successor move;*
4. *nested-play exactness, requiring L to store only finite legal play prefixes and no unbounded strategy object;*
5. *schedule and readback discipline, requiring S, R, E to be read in that order before any terminal Real-facing consumer receives the packet;*
6. *structural preservation, requiring T, C, P, N to transport, replay, source, and name only the displayed game route;*
7. *ledger non-escape, excluding global winning strategies, selected limit points, countable choice, quotient stream equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden dense-open oracles.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4077.1. The analytic rows are B, M, L, S, R, E, and the structural rows are T, C, P, N.

Baire-prefix coverage fixes the source of every finite play. Each accepted game request begins at the displayed $\mathsf{BaireSpace}^{\uparrow}$ prefix row and therefore cannot read an external open-set universe.

Metric-ball legality is confined to M and L. The complete-metric row supplies centers, radii, inclusion checks, and shrinking obligations, while the legal-move ledger records only the finite nested prefixes admitted by those checks.

The terminal handoff is scheduled. A consumer reads the finite play schedule through S, the rational tolerances through R, and the Real-facing seal through E only after the earlier rows have appeared.

Finally T, C, P, N preserve, replay, source, and name the same displayed packet. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3097.3 (Choquet-game Baire-category handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{ChoquetGame}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route supplies the Baire-category consumer surface of chapter 1542 by exposing nested legal move prefixes and complete-metric ball obligations, not by selecting a host intersection point or importing a completed-real principle.

Proof. Start with an accepted carrier. The Baire prefix B names the basic-open source, and the complete-metric row M names the ball data through which legal moves are checked.

The game content is finite. The ledger L records the displayed nested play prefixes and the successor-in-predecessor evidence needed by a Baire-category handoff, while S exposes the finite schedule consumed at the current stage.

Regular rational readback and Real sealing are downstream. The row R records the rational center and radius tolerances, and E seals the terminal read only after B, M, L, S, R have been displayed.

Transport, replay, provenance, and local naming in T, C, P, N preserve that route componentwise. Thus the handoff supplies Baire-category arguments with the visible nested-ball interface while exporting no global strategy, selected point, countable-choice enumeration, quotient stream equality, or ambient completed-real principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoquetGame}^{\uparrow}$ packet is generated from a BaireSpace prefix, CompleteMetric metric-ball rows, a nested legal-move ledger, a StreamName schedule, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$, seed)

Scope closed: Definition 4077.1, Definition 4077.2, and Definition 4077.3 seed the Choquet-game route over BaireSpaceUp, BaireCategoryUp, CompleteMetricUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global winning strategies, selected limit points, countable choice, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-prefix coverage, complete-metric ball legality, nested-play exactness, StreamName scheduling, RegSeqRat readback, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetGameUp.

Chapter 3098

A Concrete Naming Certificate for LocatedCoveringNumber$^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style metric-cover language, the LocatedCoveringNumber packet records a covering-number request whose centers, tolerance, and budget are all displayed before any compactness or dimension consumer can use it. It sits after the metric covering-number surface of chapter 2187, the finite epsilon-net carrier of chapter 1767, the compact metric route of chapter 129, and the doubling-metric finite-growth packet of chapter 2035. The packet names a finite located cover certificate; it is not an open-cover compactness theorem, selected subcover principle, quotient metric equality, or ambient choice of centers.

Definition 3098.1 (Located covering number carrier)**.** *A* LocatedCoveringNumber$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, Q, T, E, B, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, including the carrier classifier, distance requests, and ball-membership rows used by the cover.*
2. *Q is the located request row, naming the displayed subset or inspection window together with the cover request whose membership must be witnessed.*
3. *T is the tolerance ledger, recording the positive ε row and its rational or dyadic comparison data.*
4. *E is the finite ε-net center row, listing the centers and their point-name evidence through chapter 1767.*
5. *B is the covering-number budget row, recording the displayed natural-number bound and the count of center rows used by the certificate.*
6. *W is the location witness row, assigning each inspected request point to one displayed center with a distance comparison below the tolerance ledger.*
7. *H records componentwise* hsame *transport for the metric source, request, tolerance, center list, budget, witness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located cover request through the tolerance ledger, finite center list, budget row, and witness row.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCoveringNumber}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, T, E, B, W, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, hidden dense enumerations, quotient metric equality, host compactness, or a choice-selected center family.

Lean4 (definition) `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberCarrier`
Lean4 ✓ `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberTasteGate single carrier alignment`

Theorem 3098.2 (Located covering number NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCoveringNumber}^{\uparrow}$ *carrier $L = (M, Q, T, E, B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$ *surface consists exactly of carrier admission, located-center visibility, tolerance exactness, budget exactness, finite-net handoff, location-witness replay, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. Every covering-number consumer must read these rows before using the displayed budget, and no consumer may read a stronger compactness, selected-subcover, or choice-of-centers object from the packet.*

Lean4 ✓ `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberNameCertObligations`

Proof. Project the carrier of Definition 4078.1. Carrier admission exposes the metric source, located request, tolerance ledger, finite center list, budget row, location witness, transport, replay, provenance, and local name.

The located-center and tolerance obligations are read through E, T, W. A center can serve a request only when the witness row displays the center and the corresponding distance comparison against the same tolerance ledger.

Budget exactness is read through B and E. The count in B is attached to the displayed finite center row, so the covering number cannot be justified by an unlisted center or a selected subcover outside the packet.

Finally use H, C, P, N. Transport, replay, provenance, and local naming preserve the same finite-net handoff to the metric covering-number, compact-metric, and doubling-metric consumers. Coordinate exhaustion blocks arbitrary covers, host compactness, quotient metric equality, and choice-selected center families. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCoveringNumber}^{\uparrow}$ consumer displays a metric source row, located request row, tolerance ledger, finite epsilon-net center row, covering-number budget row, location witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a covering-number budget is public.

Independence witness: metric-covering-number, finite-epsilon-net, compactmetric, doubling-metric: $\mathsf{LocatedCoveringNumber}^{\uparrow}$ is not bijective to any listed sibling because it binds the request locator, tolerance ledger, center list, budget row, and witness assignment into one finite covering-number certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCoveringNumber}^{\uparrow}$ packet is generated from a metric source row, located request row, tolerance ledger, finite epsilon-net center row, covering-number budget row, location witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4078.1 and Definition 4078.2 seed the located finite-cover budget route over MetricCoveringNumber, FiniteEpsilonNet, CompactMetric, DoublingMetric, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCoveringNumber}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary open-cover compactness, selected subcovers, quotient metric equality, hidden dense enumerations, or choice-selected center families.

Upgrade path: Obligation closure requires checked carrier admission, located-center visibility, tolerance exactness, budget exactness, finite-net handoff, witness replay, transport, provenance, and local naming for BEDC.Derived.LocatedCoveringNumberUp.

Chapter 3099

A Concrete Naming Certificate for $\mathsf{SupportNerveRealization}^{\uparrow}$

Following the finite combinatorics of abstract simplicial complexes, the $\mathsf{SupportNerveRealization}^{\uparrow}$ packet records the support-family-to-boundary-subcomplex theorem of Definition 9462.1659 as concrete NameCert data. It sits between the finite support census of chapter 4048, the lifted-Betti consumer of chapter 4035, and the simplicial-complex carrier of chapter 238. The packet names support rows, complement-face rows, boundary-subcomplex rows, facet witnesses, census handoff, transport, replay, provenance, and local naming; it does not import an ambient topological realization or unrestricted homology computation.

Definition 3099.1 (Support nerve realization carrier)**.** *A* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, F, K, A, B, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the support-family row, listing the inspected nonempty support symbols.*
2. *F is the complement-face row, sending each support to its face in the tetrahedron boundary.*
3. *K is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *boundary-subcomplex row generated by the downward closure of the displayed complement faces.*
4. *A is the realization admission row, recording that every displayed support family produces a subcomplex of $\partial\Delta^3$.*
5. *B is the reverse realization row, recording the finite facet-to-support construction for each displayed boundary subcomplex.*
6. *L is the census handoff row, exposing the realized subcomplex to* $\mathsf{SupportNerveCensus}^{\uparrow}$ *and* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *consumers.*
7. *H records componentwise* hsame *transport for the support, face, subcomplex, admission, reverse, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from support-family inspection through complement faces, downward closure, reverse facet witnesses, and census handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{SupportNerveRealization}^{\uparrow}$, $\mathsf{SupportNerveCensus}^{\uparrow}$, $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, F, K, A, B, L, H, C, P, N$. It has no coordinate for host geometric realization, ambient equality of complexes, unrestricted chain homology, coefficient-field choice, or an infinite support universe.

Lean4 ✓ `BEDC.Derived.SupportNerveRealizationUp.SupportNerveRealizationTasteGate single carrier alignment`

Theorem 3099.2 (Support nerve realization NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier $R = (S, F, K, A, B, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$ *surface consists exactly of support-family exposure, complement-face exposure, finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *boundary-subcomplex admission, reverse facet-to-support realization, census handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4079.1. The rows S, F, K display the support family, its complement faces, and the resulting downward-closed boundary subcomplex.

The two realization directions are stored in A and B. The admission row verifies that displayed supports produce a subcomplex of $\partial\Delta^3$, while the reverse row records the finite facet-complement supports that realize a displayed subcomplex.

Finally use L, H, C, P, N. The handoff row sends the same finite subcomplex to census and lifted-Betti consumers, and the structural rows transport, replay, source, and name the packet. Since no ambient geometric realization, unrestricted homology computation, coefficient-field choice, or host equality coordinate is present, the listed rows exhaust the local obligations. □

Theorem 3099.3 (Support nerve realization census handoff)**.** *Every* $\mathsf{SupportNerveCensus}^{\uparrow}$ *or* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *consumer of an accepted* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier must first read the finite route*

$$\mathsf{SupportNerveRealization}^{\uparrow} \longmapsto \mathsf{SimplicialComplex}^{\uparrow} \longmapsto \mathsf{SupportNerveCensus}^{\uparrow} \longmapsto \mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}.$$

The route exposes support rows, complement-face rows, boundary-subcomplex rows, reverse facet witnesses, transport, replay, provenance, and local naming before any census or lifted-Betti entry is exported.

Proof. Start with S, F, K. These rows build the finite support nerve as a displayed subcomplex of the tetrahedron boundary.

Use the realization obligations A and B. They identify both directions of the support-family-to-subcomplex correspondence without leaving the finite face inventory.

Repack through L, H, C, P, N. The census and lifted-Betti consumers receive the same displayed subcomplex and its support witnesses, while the structural rows preserve the packet. Coordinate exhaustion prevents an ambient realization, unrestricted homology computation, or coefficient-field selector from entering the handoff. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveRealization}^{\uparrow}$ consumer displays support-family rows, complement-face rows, a finite $\mathsf{SimplicialComplex}^{\uparrow}$ boundary-subcomplex row, reverse facet witnesses, census handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a census or lifted-Betti read is public.

Independence witness: support-nerve-census, support-nerve-census-lifted-betti, simplicialcomplex: $\mathsf{SupportNerveRealization}^{\uparrow}$ is not bijective to any listed sibling because it binds the forward

support-to-subcomplex construction and reverse facet-support realization before the census handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{SupportNerveRealization}^{\uparrow}$ packet is generated from support-family rows, complement-face rows, finite SimplicialComplex boundary-subcomplex rows, reverse facet-to-support witnesses, census handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4079.1, Definition 4079.2, and Definition 4079.3 seed the finite support-family-to-boundary-subcomplex realization packet and its census handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host geometric realization, ambient equality of complexes, unrestricted homology, coefficient-field choice, or infinite support universes.

Upgrade path: Obligation closure requires checked support exposure, complement-face exposure, boundary-subcomplex admission, reverse facet-support realization, census handoff, transport, replay, provenance, and local naming for BEDC.Derived.SupportNerveRe

$\mathsf{CauchyDiagonalSelection}^{\uparrow}$ consumes the non-quotient classifier route of chapter 1820 before selecting a displayed diagonal regular-Cauchy handoff.

Chapter 3100

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalSelection}^{\uparrow}$

Following Bishop and Bridges on constructive diagonal arguments for Cauchy families, the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet records the finite row that chooses a diagonal observation window from displayed regular-Cauchy data before any Real or Completion consumer reads a seal. It sits after the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the Bishop Cauchy-modulus extraction route of chapter 4071. The diagonal-selection route is consumed by chapter 1809 as a countable dense handoff for completion-facing density reads. The same diagonal window is later consumed by chapter 7328, where the selected index is normalized into a dyadic schedule before RegSeqRat and Real consumers read the endpoint. The packet supplies a named L10 diagonal-selection handoff for Real and Completion routes; it does not assert a host sequence selector, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3100.1 (Cauchy diagonal selection carrier)**.** *A* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, M, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source schedule, exposing the finite family of observation windows available to the diagonal request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the displayed stream windows.*
3. *M is the Cauchy-modulus extraction row, read through* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *as the finite threshold discipline for the request.*
4. *W is the diagonal-window row, selecting the single displayed window whose source index and precision index have both passed through S, R, M.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the comparison between the requested tolerance, the extracted threshold, and the chosen diagonal window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback, modulus row, diagonal window, and dyadic ledger have been displayed.*
7. *H records componentwise* hsame *transport for the stream schedule, regular readback, modulus extraction, diagonal window, tolerance ledger, Real seal, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the tolerance request through S, R, M, W, T, E and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BishopCauchyModulusExtrac $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, M, W, T, E, H, C, P, N$. It has no coordinate for an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalSelectionUp.CauchyDiagonalSelectionTasteGate single carrier alignment`

Theorem 3100.2 (Cauchy diagonal selection Real uniform-structure anchor)**.** *Every* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer that reads the diagonal window as Real completion data first passes through the* $\mathsf{RealUniformStructure}^{\uparrow}$ *surface of chapter 1624. For an accepted carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the visible route is*

$\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, NameCert.

Thus the diagonal consumer reads finite uniform-entourage, stream-window, regular-rational, dyadic-tolerance, transport, replay, provenance, and naming rows before any completion-facing Real seal is consumed. It cannot replace those rows by a detached selector, quotient stream equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4080.1. The rows S, R, T, E, H, C, P, N already expose the finite stream windows, regular-rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming data.

Read the Real completion request through chapter 1624. The uniform-structure packet supplies the displayed Real entourage discipline consumed by the diagonal window before the downstream Real-facing seal is inspected.

The structural rows H, C, P, N then preserve that same route. Since the carrier has no coordinate for a detached selector, quotient stream equality, host real equality, countable choice, or ambient Real completeness, the Real-uniform-structure anchor is the only completion-facing dependency exposed by this theorem. □

Theorem 3100.3 (Cauchy diagonal selection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *diagonal-window selection, requiring the selected row W to display both the stream source window from S and the regular readback from R before the diagonal observation is admitted;*

2. *regular-Cauchy preservation, requiring R and M to keep the readback inside the same regular Cauchy family under the extracted threshold;*

3. *dyadic tolerance control, requiring T to compare the requested tolerance with the modulus threshold and the selected diagonal window;*

4. *Real-seal handoff, requiring E to be downstream of S, R, M, W, T and to export only the displayed diagonal observation;*

5. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet, with no external selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host equality, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 4080.1. The row W is meaningful only after the stream schedule and regular readback rows have been displayed, so the diagonal observation has no source outside S and R.

The modulus row M supplies the finite threshold discipline consumed by the regular-Cauchy read. Thus the readback remains inside the same displayed family rather than becoming a selected subsequence or a host-level limit.

The tolerance ledger T checks the requested dyadic scale against the extracted threshold and the chosen diagonal window. Only after that finite comparison can the Real-facing row E be consumed.

Finally, H, C, P, N preserve and name the same coordinates. Since none of the refused objects appears in the carrier, the five displayed obligations exhaust the local certificate surface. □

Theorem 3100.4 (Cauchy diagonal selection Real-seal handoff)**.** *For every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is reachable only after the* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *row M, diagonal window W, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T are displayed. The handoff exports no external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier in Definition 4080.1. The Real-facing row E appears only downstream of the displayed schedule, regular readback, modulus extraction, diagonal window, and dyadic tolerance ledger.

Use the five obligations in Definition 4080.3. They require diagonal-window selection through S, R, W, regular-Cauchy preservation through R, M, dyadic tolerance control through T, and Real-seal handoff through E after those rows are visible.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient completeness, none can be exported at the seal. □

Theorem 3100.5 (Cauchy diagonal selection RegSeqRat threshold stability)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. Suppose another accepted carrier*

$$D' = (S, R, M', W, T, E, H, C, P, N)$$

has the same $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, diagonal window W, tolerance ledger T, Real seal E, transport row H, replay row C, provenance row P, and local naming row N, and suppose the displayed threshold row M' is a stronger Cauchy-modulus threshold compatible with the same S, R, W, T rows. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal of D' remain on the same finite route as those of D:*

$$S, R, M', W, T \longmapsto E \longmapsto H, C, P, N.$$

In particular, strengthening the modulus threshold does not introduce an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project both carriers through Definition 4080.1. The two packets differ only in the modulus row, while $S, R, W, T, E, H, C, P, N$ are displayed as the same rows in both packets.

Apply the NameCert obligations of Definition 4080.3. The regular-Cauchy preservation obligation reads the modulus row only as the finite threshold discipline attached to R, and the dyadic tolerance obligation reads T against the chosen diagonal window W.

Since M' is a displayed stronger threshold compatible with the same S, R, W, T rows, the replay through C transports the regular readback along the same finite request route. No new selector or quotient comparison is needed to read the strengthened threshold.

Finally apply the Real-seal handoff of Definition 4080.4. The terminal seal E can still be consumed only after S, R, M', W, T are displayed, and the structural rows H, C, P, N keep the route inside the listed coordinates. □

Theorem 3100.6 (Cauchy diagonal selection finite-window induction)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. Induction over the displayed finite window schedule preserves the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopCauchyModulusExtraction}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, strengthening or extending any finite prefix keeps the selected diagonal read inside $S, R, M, W, T, E, H, C, P, N$ and cannot introduce an external selector, quotient stream equality, countable choice, trichotomy, host equality, or ambient completeness.

Proof. The base window is the carrier projection of Definition 4080.1. The rows S, R, M, W, T, E already display the StreamName schedule, regular readback, modulus threshold, diagonal window, dyadic tolerance ledger, and Real seal in the stated order.

For the induction step, extend a displayed finite prefix by one accepted window. The NameCert obligations of Definition 4080.3 force the new window to be read through S and R, then checked by the modulus row M and tolerance ledger T before the Real-facing row E is consumed.

If the extension strengthens the threshold instead of adding a new observation, apply RegSeqRat threshold stability, Definition 4080.5. The strengthened row remains compatible with the same S, R, W, T route and does not add a separate selector.

Finally use the Real-seal handoff, Definition 4080.4. Since H, C, P, N transport, replay, source, and name only the displayed packet, the induction cannot acquire quotient stream equality, countable choice, trichotomy, host equality, or ambient completeness from outside the carrier. □

Theorem 3100.7 (Cauchy diagonal selection subsequence route)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. When D is read as a subsequence corridor, every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal consumer factors first through the* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *surface of chapter 5902. The diagonal window is therefore consumed only along the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{BishopCauchyModulusExtraction}^{\uparrow},\ \mathsf{CauchySubsequenceSelector}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{R}$$

with structural support carried by H, C, P, N. The route exposes no external diagonal selector, quotient stream equality, countable-choice subsequence, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 4080.1. The rows S, R, M, W, T display the source schedule, regular readback, modulus threshold, selected diagonal window, and dyadic tolerance ledger before E is available.

Apply finite-window induction, Definition 4080.6. Each accepted prefix keeps the selected window inside S, R, M, W, T, so the diagonal read has the same finite selector shape required by the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ carrier surface.

Read the selector monotonicity and Cauchy-gap rows from chapter 5902. They supply the subsequence corridor between the displayed stream windows and the regular-rational handoff, while preserving the diagonal packet's structural rows H, C, P, N.

Finally apply the Real-seal handoff, Definition 4080.4. Since the consumer reaches E only after the selector corridor, regular readback, modulus threshold, diagonal window, and tolerance ledger have been displayed, it cannot import an external diagonal selector, countable choice, quotient stream equality, host real equality, or ambient completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet is generated from StreamName source windows, RegSeqRat regular readback, a BishopCauchyModulusExtraction threshold row, a selected diagonal window, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4080.1, Definition 4080.3, Definition 4080.4, Definition 4080.5, Definition 4080.6, and Definition 4080.7 give the obligation surface over StreamName, RegSeqRat, BishopCauchyModulusExtraction, CauchySubsequenceSelector, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequences, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked scope boundary for downstream Real-separability and regular-modulus-normalization consumers.

The diagonal selector reads $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and $\mathsf{Real}^{\uparrow}$ seal rows through the non-quotient classifier route of chapter 1820. This keeps the selected diagonal handoff attached to displayed regular-Cauchy evidence rather than to a quotient class, representative selector, ambient completeness rule, or host equality oracle. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet forwards its selected finite tail through the $\mathsf{CauchyFilterLimit}^{\uparrow}$ handoff of chapter 1558 before any Real-facing consumer inspects the terminal seal. This route preserves the existing $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ dependencies from the diagonal-selection carrier, so the handoff is a displayed completion-facing dependency rather than a detached selector or ambient limit principle. Every completion-facing $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ read consumed by chapter 1558 must first display the $\mathsf{StreamName}^{\uparrow}$ finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the diagonal-window selection row, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer therefore receives a lattice route through displayed rows, not a detached diagonal selector, quotient stream equality, selected host limit, countable-choice schedule, or ambient completeness principle. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route consumes the regularized Cauchy-modulus handoff of chapter 7888 before any diagonal subsequence or Real seal is read. Thus a diagonal consumer must expose the $\mathsf{CauchyModulus}^{\uparrow}$ threshold row, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal route rather than reading the diagonal as a detached selector, quotient stream, or ambient completeness coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ consumer link records the downstream read shared by diagonal-selection, regular-Cauchy extraction, and L10 completion consumers.

Theorem 3100.8 (Cauchy diagonal selection consumer link)**.** *Every* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer reaches its selected diagonal only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the diagonal-selection sibling routes of Definition 4080.15 and Definition 4080.16. When the consumer is compared with the regular-Cauchy diagonal-extraction*

surface of chapter 1773, the shared route is the finite handoff through the source windows, regular readback, dyadic tolerance ledger, diagonal window, terminal $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *row. It exports no detached selector, quotient stream equality, countable-choice schedule, selected host limit, or ambient completion principle.*

Proof. Project the carrier $D = (S, R, M, W, T, E, H, C, P, N)$ from Definition 4080.1. The rows S and R expose the finite $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before the selected window W can be consumed.

Use Definition 4080.15 to fix the displayed StreamName, RegSeqRat, DyadicRatCore, and Real-seal route. Then use Definition 4080.16 to compare that route with the regular-Cauchy diagonal-extraction chapter.

The structural rows H, C, P, N preserve this handoff under transport, replay, provenance, and local naming. Since the carrier contains no coordinate for quotient stream equality, countable choice, a selected host limit, or ambient completion, every consumer that reaches the selected diagonal remains inside the displayed finite route. □

The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dossier is consumed by chapter 4202 through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows recorded in the local carrier. This cross-reference keeps the diagonal-selection window visible as the finite input to the rational interval locator, rather than as a self-contained selector detached from the surrounding real-name lattice. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dependency route also consumes the rectangular-window and cofinal-schedule surface of chapter 1689 together with the single diagonal-index route of chapter 1873 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback is exposed. Thus the selected diagonal window is a finite handoff through displayed double-diagonal and diagonal-argument rows, not a detached selector or an ambient completion claim. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ L10 route consumes the source windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real seal of chapter 38 through the carrier displayed in Definition 4080.1. These dependencies are already part of the diagonal-selection packet; this child route makes them visible to the top-level dossier scan without adding a second closure ledger or theorem surface. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ forward route consumes the regular Cauchy modulus transfer of chapter 7610, the finite $\mathsf{StreamName}^{\uparrow}$ source windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any diagonal-selection read is exported. Thus the sibling lattice records the diagonal window as a displayed finite dependency route, not as an isolated selector, quotient stream, selected limit, or ambient completeness coordinate.

Theorem 3100.9 (Cauchy diagonal selection rational-interval locator route)**.** *Every* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *consumer that reads a* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal window must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal carried by the diagonal-selection packet. The route then hands the visible rational lower and upper interval data to chapter 4202; it exports no new selector, quotient stream equality, selected real limit, countable-choice schedule, or ambient completeness claim.*

Proof. Project the diagonal-selection packet through its displayed child routes. The only analytic rows available before locator consumption are the finite StreamName windows, regular rational readback rows, dyadic tolerance ledger, and Real seal.

The rational-interval locator may therefore read the selected diagonal window only after those rows are visible. At that point the lower and upper rational interval data are already finite packet

data, so the consumer receives a local locator route rather than a new diagonal selector or a completion theorem.

Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve that same route into chapter 4202. Since no carrier row contains quotient stream equality, a selected real limit, a countable-choice schedule, or ambient completeness, the route exports none of those objects. □

The CauchyDiagonalSelection$^{\uparrow}$ diagonal window selected from the StreamName$^{\uparrow}$ source schedule, RegSeqRat$^{\uparrow}$ regular readback, and DyadicRatCore$^{\uparrow}$ tolerance ledger is the finite window read by RationalIntervalLocator$^{\uparrow}$ in chapter 4202 before a Real$^{\uparrow}$-facing interval consumer can inspect rational lower and upper bounds. This route records that the locator consumes the displayed diagonal window as interval data, rather than importing a selected real limit, quotient stream equality, countable-choice schedule, or ambient completeness coordinate. The rational lower and upper endpoint exports of chapter 4202 consume the CauchyDiagonalSelection$^{\uparrow}$ finite-window route before any Real$^{\uparrow}$-facing read treats those endpoints as interval evidence. This downstream use keeps the locator's bracket rows tied to displayed StreamName, RegSeqRat, and dyadic tolerance data rather than to a selected real limit, quotient stream equality, or ambient completeness coordinate. The CauchyDiagonalSelection$^{\uparrow}$ route feeds the RealLocator$^{\uparrow}$ interval read of chapter 1669 only after the finite source windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38 have been displayed. Thus the locator-facing consumer reads the same diagonal-selection window and sibling rows before any completion-facing seal is exposed, rather than importing a detached selector, quotient stream, selected limit, or ambient completeness coordinate. The RealSeparability$^{\uparrow}$ countable-dense handoff of chapter 1809 consumes the CauchyDiagonalSelection$^{\uparrow}$ window only after the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ rows have been displayed. Thus the diagonal-selection packet supplies the finite diagonal route for the separability consumer without introducing a quotient stream, a selected real representative, or a choice schedule.

Theorem 3100.10 (Cauchy diagonal selection real-spread route)**.** *Every* CauchyRealSpread$^{\uparrow}$ *or* Real$^{\uparrow}$ *completion consumer that reads a* CauchyDiagonalSelection$^{\uparrow}$ *diagonal window must first pass through the StreamName-RegSeqRat route of Definition 4080.15. The handoff then feeds the real-spread completion surface of chapter 4161 and the terminal real-name seal of chapter 38 using only the displayed finite source windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows.*

Proof. Project the StreamName-RegSeqRat route already displayed in this chapter. It exposes finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, dyadic tolerance data, and the terminal Real$^{\uparrow}$ seal before a diagonal-selection consumer can read the selected window.

The CauchyRealSpread$^{\uparrow}$ handoff consumes exactly that finite completion-facing data. Its sibling chapter records a real-spread completion route, and the Real$^{\uparrow}$ chapter supplies the terminal seal; neither sibling adds a new selector to the diagonal-selection packet.

Transport, replay, provenance, and local naming preserve the same rows. Thus the route to CauchyRealSpread$^{\uparrow}$ and Real$^{\uparrow}$ is the displayed completion route, not countable choice, quotient stream equality, selected representatives, or a detached host completion. □

Theorem 3100.11 (Cauchy diagonal selection nonescape)**.** *Every completion-facing consumer generated from a* CauchyDiagonalSelection$^{\uparrow}$ *packet and the real-spread route above remains inside the displayed StreamName, RegSeqRat, dyadic-tolerance, Real-seal, transport, replay, provenance,*

and local-name rows. It cannot become countable choice, a selected representative family, quotient stream equality, a detached diagonal selector, or an ambient completeness principle.

Proof. Induct over the displayed diagonal-selection packet. The base case has only the source windows, regular readback, tolerance ledger, and Real seal already named by Definition 4080.15.

In the finite step case, the real-spread consumer may read a new diagonal window only after those rows are visible. The sibling $\mathsf{CauchyRealSpread}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ chapters receive the same finite completion-facing data and do not add a choice schedule or selected representative coordinate.

The structural rows then transport, replay, source, and name the packet. Since no row contains countable choice, quotient stream equality, selected representatives, a detached selector, or ambient completeness, none of those objects can escape from the displayed route. □

Theorem 3100.12 (Cauchy diagonal selection RealSpread dependency anchor)**.** *Any* $\mathsf{CauchyRealSpread}^{\uparrow}$ *or Real-separability consumer reading a* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal route must pass through the* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal schedule of chapter 4080, and the* $\mathsf{CauchyRealSpread}^{\uparrow}$ *dependency route of chapter 4161 before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route is a finite projection of displayed dependency rows, not a compactness principle or choice principle.*

Proof. Project the diagonal-selection carrier through Definition 4080.15. The visible rows are the finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, selected diagonal schedule, and terminal Real seal.

Read the real-spread dependency route before the Real-facing consumer. That sibling route receives the same displayed window, readback, tolerance, and schedule rows; it does not add a branch selector or a countable-choice schedule.

Finally pass to the terminal $\mathsf{Real}^{\uparrow}$ seal. Since every row in the path is already displayed by the diagonal-selection and real-spread packets, the anchor only projects a finite dependency route and does not assert new compactness, quotient equality, host real equality, or ambient completeness. □

The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ Real-completion route consumes the finite diagonal-meet surface of chapter 909 after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are visible, and before any completion-facing selector is public. This records the diagonal-selection handoff as a displayed dependency through the regular Cauchy meet route rather than as a detached selector, quotient stream, selected limit, or ambient completion coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ handoff consumes the $\mathsf{RegularCauchySandwich}^{\uparrow}$ cofinal-tail mesh of chapter 5874 before any completion-facing diagonal route is read. Thus the selected diagonal window is constrained by displayed lower-tail, upper-tail, common-modulus, and regular readback rows rather than by a detached selector, quotient stream, or ambient completeness coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dossier reads its diagonal-selection window through the same L10 route used by the regular Cauchy modulus siblings: the $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ budget of chapter 4158, the $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ handoff of chapter 7610, and the existing $\mathsf{StreamName}^{\uparrow}$-$\mathsf{RegSeqRat}^{\uparrow}$-$\mathsf{Real}^{\uparrow}$ path recorded in Definition 4080.15. Thus diagonal selection consumes finite stream windows, dyadic tolerances, regular rational readback, and terminal real seals as displayed sibling rows rather than as a detached tail selector or quotient-completion representative. The diagonal-selection packet consumes chapter 4158 as the visible uniform tail-budget surface before handing data to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Theorem 3100.13 (CauchyDiagonalSelection regular-Cauchy dependency)**.** *Every accepted* CauchyDiagonalSelecti *read that exports a selected diagonal window first consumes the regular-Cauchy diagonal route of*

chapter 909, the regular rational readback of chapter 301, and the finite stream-window row of chapter 126. The exported diagonal window is therefore downstream of the RegularCauchyDiagonalMeet, RegSeqRat, and StreamName sibling surface, not a detached selector or host-side choice function.

Proof. Project the local $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route through the existing diagonal-selection carrier. The selected window is read only after the stream schedule and regular readback rows have been exposed.

The regular-Cauchy sibling route supplies the diagonal-meet dependency. Once the consumer has passed through chapter 909, the RegSeqRat row fixes the displayed rational readback and the StreamName row fixes the finite window schedule.

Thus the exported selected diagonal window is a named sibling handoff. No row in the packet supplies a detached selector, quotient stream equality, countable-choice schedule, or host completion representative. □

Theorem 3100.14 (Cauchy diagonal selection regular-Cauchy gluing forward route)**.** *Every* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that uses a diagonal-selection window from* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *must first expose the finite subsequence window, rowwise Cauchy budget, diagonal index ledger, and componentwise replay rows supplied by the diagonal-selection packet. The handoff supplies the finite window consumed by chapter 5899; it does not choose endpoints, assert completion, select a limit, or replace the gluing packet's own regular-Cauchy compatibility rows.*

Proof. Read the diagonal-selection packet through its local carrier rows. The selected diagonal window, rowwise Cauchy budget, index ledger, transport, replay, provenance, and local naming rows are the only data exported to a downstream gluing consumer.

Feed exactly those rows to the regular-Cauchy gluing route of chapter 5899. The gluing packet may use the finite subsequence window as input, but it must still display its own regular-Cauchy compatibility and gluing rows.

Since the diagonal-selection packet contains no endpoint selector, completion principle, selected limit, or ambient Real-completeness row, the forward route exports none of those claims. □

Theorem 3100.15 (Cauchy diagonal selection StreamName-RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *packet must first expose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any diagonal selector is consumed. Thus a completion-facing diagonal request reads the same displayed* S, R, T, E *route recorded by the local carrier and cannot replace it by a detached tail selector, quotient stream, selected limit, or hidden completion representative.*

Proof. Project the carrier $D = (S, R, M, W, T, E, H, C, P, N)$ from Definition 4080.1. The source schedule S supplies the finite $\mathsf{StreamName}^{\uparrow}$ windows, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ values, the tolerance row T supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison, and the terminal row E supplies the $\mathsf{Real}^{\uparrow}$ seal.

Use the visible L10 dependency route recorded in chapter 4080. Its sibling route binds the diagonal-selection packet to the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows before the top-level dossier can be consumed.

Now apply the rational interval locator forward route in the same dossier. That route reads the selected diagonal window as interval data only after the displayed source, readback, and tolerance rows have been exposed.

The structural rows H, C, P, N then transport, replay, source, and name this finite route. Since the carrier has no coordinate for a detached tail selector, quotient stream, selected limit, or hidden completion representative, none of those objects can replace the displayed S, R, T, E route. □

Theorem 3100.16 (Cauchy diagonal selection regular-diagonal extraction link)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *packet that exposes the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows can be consumed by the regular-Cauchy diagonal-extraction route of chapter 1773 only through those finite windows. The link exports the same schedule, tolerance, regular-readback, and Real-seal rows; it does not supply countable choice, quotient stream equality, a selected representative, or a detached host diagonal.*

Proof. Project the rows already isolated by Definition 4080.15. The visible part of the accepted packet contains the finite StreamName schedule, the DyadicRatCore tolerance ledger, the RegSeqRat readback, and the terminal Real seal.

Now compare with the carrier surface of chapter 1773. Its diagonal extraction route consumes a source family, a unary selector, aligned StreamName windows, one dyadic tolerance ledger, one regular rational diagonal readback, and a terminal Real seal.

The shared rows therefore identify the only admissible handoff. The Cauchy diagonal selection packet may feed the regular-Cauchy extraction route through the displayed windows, but no row introduces countable choice, quotient equality of streams, a selected representative, or a detached host diagonal. □

The diagonal-selection read first fixes the finite stream windows and regular-rational readback before any terminal real seal is available. This places the route beside the $\mathsf{StreamName}^{\uparrow}$ certificate of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ certificate of chapter 301.

Theorem 3100.17 (Cauchy diagonal StreamName RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *read that exports a diagonal real candidate first consumes the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows before the* $\mathsf{Real}^{\uparrow}$ *seal is inspected. The route exposes no direct real equality test, selected infinite subsequence outside the displayed diagonal ledger, quotient stream equality, countable-choice functional, or host completeness principle.*

Proof. Start with the finite diagonal ledger. Each selected row is read through a displayed stream window, so the consumer sees only the finite $\mathsf{StreamName}^{\uparrow}$ observations attached to the diagonal route.

The regular-rational readback is downstream from those windows. It repacks the observed rational approximants as $\mathsf{RegSeqRat}^{\uparrow}$ data before any real-facing seal is offered.

Only after these two sibling dependencies have been replayed may the $\mathsf{Real}^{\uparrow}$ seal be read. The refused objects are not coordinates in the displayed route, so the diagonal-selection packet remains a finite StreamName and RegSeqRat dependency certificate. □

Theorem 3100.18 (CauchyDiagonalSelection RegSeqRat-StreamName real-completion route)**.** *Every completion-facing* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer that reads a diagonal candidate must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before the selected diagonal window is consumed. The route is therefore a displayed L10 dependency through finite stream windows, regular rational readback, and the real-name boundary, not a detached selector, quotient stream, selected limit, or ambient completeness coordinate.*

Proof. Project the local $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ carrier. Its diagonal window is read only after the stream schedule, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows have been exposed.

The cited StreamName, RegSeqRat, and Real sibling chapters provide the three completion-facing boundaries. Since none of those rows stores a selector outside the packet, quotient stream equality, a selected host limit, or ambient completeness, the diagonal-selection consumer remains inside the displayed route. □

Chapter 3101

A Concrete Naming Certificate for $\mathsf{ApproachSpace}^{\uparrow}$

Following Lowen, the $\mathsf{ApproachSpace}^{\uparrow}$ packet records convergence by finite distance-to-set and tolerance observations rather than by a quotient completion or a selected limit object. It sits between the metric carrier of chapter 57, the open-neighbourhood carrier of chapter 88, and the Cauchy-filter completion boundary of chapter 1569. The packet names an L10 convergence object that can be consumed by metric, topology, and filter-completion routes while keeping quotient points, host real equality, countable choice, and ambient completeness outside the carrier.

Definition 3101.1 (ApproachSpace carrier)**.** *An* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, D, T, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the point row, naming the finite point observations and the local classifier surface used by later convergence consumers.*
2. *D is the distance-to-set and tolerance row, recording displayed requests of the form "the observed point is within the listed tolerance of the listed observed set" without importing a host-valued distance function on all subsets.*
3. *T is the topology-neighbourhood row, recording the open-neighbourhood observations generated by the displayed tolerances and the topology-facing classifier inherited from* $\mathsf{Topology}^{\uparrow}$*.*
4. *F is the Cauchy-filter handoff row, naming which finite Cauchy-filter requests may consume D and T before a completion-facing read reaches* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for point observations, distance-to-set tolerances, topology-neighbourhood rows, Cauchy-filter handoffs, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a convergence query through X, D, T, F, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{ApproachSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares convergence observations only through X, D, T, F, H, C, P, N. It has no coordinate for quotient completion points, equality in a host real line, countable-choice selectors, ambient completeness, or a hidden global distance-to-all-subsets oracle.

Lean4 ✓ `BEDC.Derived.ApproachSpaceUp.ApproachSpaceTasteGate single carrier alignment`

Theorem 3101.2 (ApproachSpace NameCert obligations)**.** *For every accepted* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier*

$$A = (X, D, T, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the point row, distance-to-set tolerance row, topology-neighbourhood row, Cauchy-filter handoff row, componentwise transport, displayed replay, provenance, and local naming;*
2. *classifier transport, requiring every accepted convergence comparison to preserve* X, D, T, F *through the displayed* hsame *row* H*;*
3. *metric and topology compatibility, requiring metric-origin tolerance reads to enter through* D *and topology-origin neighbourhood reads to enter through* T *before they are compared as one approach-space observation;*
4. *Cauchy-filter handoff, requiring completion-facing reads to pass through* F *before any* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer uses the packet;*
5. *non-escape, excluding quotient points, host* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, ambient completeness, selected completion representatives, and any hidden bridge claim.*

No local obligation is stored outside these five rows.

Proof. Project the carrier of Definition 4081.1. The finite tuple displays point observations, distance-to-set tolerances, topology-neighbourhood observations, the Cauchy-filter handoff, componentwise transport, replay, provenance, and local naming.

The metric and topology routes meet only at the displayed classifier surface. A metric-origin read must expose D, while a topology-origin read must expose T; transport through H preserves both rows componentwise before the replay row C compares them as one convergence observation.

The completion-facing route is similarly visible. The row F is the only Cauchy-filter handoff coordinate, so a completion consumer reaches $\mathsf{CauchyFilterCompletion}^{\uparrow}$ only after reading the finite rows named by F together with the same transport, replay, provenance, and local naming rows.

Finally inspect the carrier inventory. No coordinate stores quotient completion points, host real equality, countable-choice selectors, selected representatives, ambient completeness, or bridge data. Hence the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ApproachSpaceUp.ApproachSpaceNameCertObligations`

Theorem 3101.3 (ApproachSpace completion handoff)**.** *For every accepted* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier*

$$A = (X, D, T, F, H, C, P, N),$$

every Cauchy-filter completion consumer must read distance-to-set tolerances through D*, topology-neighbourhood observations through* T*, and the Cauchy-filter handoff through* F *before reaching any completion-facing row. The shared finite convergence packet is therefore*

$$D, T \longmapsto F \longmapsto H, C, P, N.$$

It exports no quotient completion point, selected representative, host $\mathsf{Real}^{\uparrow}$ *equality, countable-choice selector, or ambient completeness row.*

Proof. Project the carrier of Definition 4081.1. The rows D, T, F are the only distance-to-set, topology-neighbourhood, and Cauchy-filter handoff coordinates available to a completion-facing consumer.

Apply the metric and topology compatibility clause of Definition 4081.2. A metric-origin tolerance read must enter through D, and a topology-origin neighbourhood read must enter through T, before the packet treats them as one approach-space observation.

The Cauchy-filter handoff clause then forces the completion-facing read through F. The structural rows H, C, P, N only transport, replay, source, and name that same finite packet.

Finally use the non-escape clause from the obligation theorem. Since the carrier contains no quotient completion point, selected representative, host real equality, countable-choice selector, or ambient completeness coordinate, none of those objects can be reached after the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ApproachSpace}^{\uparrow}$ packet is generated from point rows, distance-to-set tolerance rows, topology-neighbourhood rows, Cauchy-filter handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4081.1 and Definition 4081.2 seed the finite approach-space convergence object over point, tolerance, topology-neighbourhood, Cauchy-filter, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion points, host Real equality, countable choice, ambient completeness, selected completion representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, metric/topology compatibility, Cauchy-filter handoff exactness, componentwise transport, replay, provenance, and ledger non-escape for BEDC.Derived.ApproachSpaceUp.

Chapter 3102

A Concrete Naming Certificate for $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$

The $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ packet realizes the confluence audit boundary of chapter 9378 as a finite diamond-frontier object. It sits beside the critical-path carrier of chapter 765, the confluence audit packet of chapter 735, the local diamond witness of chapter 3595, the residual checker of chapter 3827, and the subject-reduction discharge socket of chapter 719. The packet names local peaks, candidate joins, residual frontiers, substitution boundaries, bounded checks, and obstruction rows without claiming full confluence or subject reduction.

Falsifiable prediction: Every accepted $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ confluence-frontier read exposes local-peak, candidate-join, residual-frontier, substitution-boundary, bounded-check, obstruction, transport, replay, provenance, and local NameCert rows before any diamond-boundary handoff is consumed.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, metacic-local-diamond-witness, metacic-confluence-residual-checker, subject-reduction-discharge-socket: the critical-path carrier records dependency pressure without a per-diamond boundary; the confluence audit packet records global audit rows without bounded local frontier status; the local-diamond witness records a peak and candidate join without residual-checker obstruction retention; the residual checker records bounded residual windows without the subject-reduction socket boundary; and the discharge socket records typing-side rows without confluence peak data. None is bijective to $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$.

Definition 3102.1 (MetaCIC confluence-diamond boundary carrier)**.** *A* $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, J, R, S, B, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the local-peak row, naming the source term and the two outgoing reduction or parallel-reduction observations.*
2. *J is the candidate-join row, recording the proposed common reduct and the two route requests into it.*
3. *R is the residual-frontier row, listing residual redex descendants, binder-depth exposure, and branch-local frontier checks.*

4. *S is the substitution-boundary row, naming the substitution-sensitive input that remains visible whenever the candidate join is not an unconditional diamond.*
5. *B is the bounded-check row, recording the finite closedness, support, branch-compatibility, and variable-escape checks applied to the displayed peak.*
6. *O is the obstruction row, naming the still-open endpoints: unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, and kernel consistency.*
7. *H records componentwise* hsame *transport for local peaks, candidate joins, residual frontiers, substitution boundaries, bounded checks, obstruction rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence query through $K, J, R, S, B, O, H, P, N$.*
9. *P is the* Pkg *provenance over* MetaCICConfluenceDiamondBoundary$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICConfluenceA MetaCICLocalDiamondWitness$^{\uparrow}$, MetaCICConfluenceResidualChecker$^{\uparrow}$, SubjectReductionDischargeSocket$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, J, R, S, B, O, H, C, P, N$. It has no coordinate for a global confluence theorem, a subject-reduction theorem, unrestricted substitution compatibility, quotient equality of terms, evaluator equality, or kernel consistency.

Theorem 3102.2 (MetaCIC confluence-diamond boundary obligations)**.** *For every accepted* MetaCICConfluenceDia *carrier*

$$D = (K, J, R, S, B, O, H, C, P, N),$$

the local NameCert$_{\text{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ *surface consists exactly of local-peak exposure through K, candidate-join exposure through J, residual-frontier exposure through R, substitution-boundary retention through S, bounded-check exposure through B, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceDiamondBoundaryUp.MetaCICConfluenceDiamondBoundaryCarrier namecert obligations`

Proof. Project the carrier of Definition 4082.1. Rows K, J, R, S, B, O are the diamond-facing data, and rows H, C, P, N are the structural certificate data.

The local peak and candidate join must be read together. Row K displays the source and the two outgoing observations, while J records only a proposed common reduct and route requests rather than a global Church–Rosser theorem.

The residual and substitution boundaries remain visible next. Rows R and S expose residual descendants, binder-depth data, and substitution-sensitive inputs, while B records the bounded finite checks that have actually been performed.

Finally O keeps the open endpoints named, and H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate contains full confluence, subject reduction, unrestricted substitution compatibility, quotient term equality, evaluator equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 3102.3 (MetaCIC confluence-diamond boundary handoff)**.** *Every confluence consumer of an accepted* MetaCICConfluenceDiamondBoundary$^{\uparrow}$ *carrier factors through*

$$K, J, R, S, B, O, H, C, P, N$$

before it may export a diamond-boundary handoff. The handoff may feed the confluence audit packet, local-diamond witness, residual checker, critical-path packet, or subject-reduction discharge socket named above, but it does not fill the substitution boundary, prove subject reduction, prove full confluence, decide evaluator equality, or establish kernel consistency.

Proof. Begin with the obligation surface of Definition 4082.2. It forces every consumer read through the peak row, candidate join row, residual frontier, substitution boundary, bounded checks, and obstruction row before the structural rows are consumed.

The sibling handoffs preserve that boundary. The audit packet may read the confluence-frontier data, the local-diamond witness may read the peak and join rows, the residual checker may read R, S, B, O, the critical-path packet may read the obstruction pressure, and the discharge socket may read only the subject-reduction-facing boundary.

No sibling read adds a hidden theorem row to D. Therefore the handoff remains finite diamond-boundary data and cannot be cited as unrestricted substitution compatibility, subject reduction, full confluence, evaluator equality, or kernel consistency. □

Theorem 3102.4 (MetaCIC confluence-diamond typed-window nonescape)**.** *Any typed-window consumer of an accepted* $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ *packet must display the local peak row K, candidate join row J, residual-frontier row R, substitution-boundary row S, bounded-check row B, obstruction row O, and structural rows H, C, P, N before reading the handoff. The typed window may feed the local-diamond witness and residual-checker siblings named above, but it cannot export subject reduction, full confluence, evaluator equality, or kernel consistency.*

Proof. Start from the carrier of Definition 4082.1. A typed-window read has no coordinate outside $K, J, R, S, B, O, H, C, P, N$, so it must expose the peak, join, residual, substitution, bounded-check, and obstruction rows before any structural replay or provenance row can name the handoff.

Apply the obligation surface of Definition 4082.2. The local $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ row records exactly those visible obligations, including obstruction retention and the absence of unrestricted substitution compatibility, subject reduction, evaluator equality, and kernel consistency.

Finally apply the handoff theorem Definition 4082.3. The local-diamond witness may consume K, J, and the residual checker may consume R, S, B, O, but neither sibling adds a theorem row to the typed window. Thus the exported data remain a typed local diamond boundary rather than a global confluence, subject-reduction, evaluator-equality, or kernel-consistency theorem. □

Lean follow-up for `BEDC.Derived.MetaCICConfluenceDiamondBoundaryUp` requires Nontrivial and FieldFaithful TasteGate instances at the displayed carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ packet is generated from local-peak rows, candidate-join rows, residual-frontier rows, substitution-boundary rows, bounded-check rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4082.1, Definition 4082.2, and Definition 4082.3 seed the finite confluence-diamond boundary packet for MetaCIC audit consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution compatibility, full confluence, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-peak exposure, candidate-join exactness, residual-frontier coverage, substitution-boundary retention, bounded-check exactness, obstruction retention, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConfluenceDiamondBoundaryU

Chapter 3103

A Concrete Naming Certificate for ??

Following Darboux and Riemann, the **??** packet isolates the finite upper-sum side of the Darboux interface before a consumer asks for a full upper/lower gap. It sits between the dyadic step-function surface of chapter 428, the tagged finite-sum consumer of chapter 1615, the Darboux integration handoff of chapter 1460, and the tagged-mesh integral boundary of chapter 1526. The packet supplies a finite upper-bound ledger for later Darboux and Riemann criteria without importing host suprema, open-cover compactness, quotient function equality, selected mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3103.1 (Upper Darboux sum carrier)**.** *An **??** carrier is a finite* BHist *packet*

$$U = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the partition row, naming the finite ordered cell list, cell-width ledger, endpoint order, and refinement handle used by every later read.*
2. *B is the cell upper-bound ledger. For each displayed cell it records the chosen upper value row and the comparison proof that every admitted sample value in that cell is below the carried bound.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step value used to display the upper envelope.*
4. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row computed from the displayed widths of P and the upper values exposed by B and D.*
5. *I is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *provenance row, recording that the upper-sum read is available to the Darboux handoff before any lower-sum or gap row is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing the finite real-name comparison used by upper-bound consumers.*
7. *H records componentwise* hsame *transport for the partition, upper-bound ledger, dyadic step row, Riemann-sum row, Darboux handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to upper-bound reads, from upper-bound reads to the dyadic step row, from the dyadic step row to the Riemann-sum row, and from the sum row to the Darboux handoff.*

9. *Q is the* Pkg *provenance over* ??, DyadicStepFunction$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{??}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host supremum, an open-cover compactness theorem, a selected mesh limit, a lower Darboux sum, quotient equality of functions, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.UpperDarbouxSumUp.TasteGate.UpperDarbouxSumTasteGate single carrier alignment`

Theorem 3103.2 (Upper Darboux sum NameCert obligations)**.** *For every accepted ?? carrier*

$$U = (P, B, D, S, I, R, H, C, Q, N),$$

the local NameCert$_{??}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition row, cell upper-bound ledger, dyadic step row, Riemann-sum row, Darboux handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-refinement stability, requiring each transported refinement of P to carry the same cell-width ledger and to route upper-bound reads through the displayed refinement handle;*
3. *upper-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated upper relation on every displayed cell;*
4. *Darboux handoff, requiring the row I to expose the upper-sum read to* DarbouxIntegral$^{\uparrow}$ *without producing a lower sum, a gap witness, or a host supremum;*
5. *Riemann-sum consumer boundary, requiring every* RiemannSum$^{\uparrow}$ *or* RiemannIntegral$^{\uparrow}$ *consumer to read the finite widths, upper values, replay, provenance, and local name before it can use the upper-sum row.*

Proof. Carrier admission is the projection of Definition 4083.1. The accepted packet displays exactly the partition, upper-bound, dyadic step, finite sum, Darboux handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-refinement stability comes next. The partition row P names the finite cells and widths, so a transported refinement can only expose an upper read through the same displayed cell ledger and refinement handle.

Upper-bound ledger transport is then componentwise. The rows B and D move together under H, and replay through C recomputes the dyadic upper envelope before the finite sum row S is available.

The Darboux and Riemann boundaries are exhausted by S and I. The Darboux consumer receives only the upper-sum handoff, while a Riemann-facing consumer must read the same finite widths, upper values, replay, provenance, and local name before using the sum.

The excluded analytic objects are absent from the carrier. No row stores a host supremum, open-cover compactness, selected mesh limit, quotient function equality, lower-sum datum, or ambient completeness principle. □

Theorem 3103.3 (Upper-sum consumer boundary)**.** *Every* DarbouxIntegral$^{\uparrow}$, RiemannSum$^{\uparrow}$, *or* RiemannIntegral$^{\uparrow}$ *consumer of an accepted ?? carrier $U = (P, B, D, S, I, R, H, C, Q, N)$ must first read the partition row P, the upper-bound ledger B, the* DyadicStepFunction$^{\uparrow}$ *row D, the finite sum row S, the Darboux handoff I, and the comparison row R, with transport, replay, provenance, and*

local naming supplied by H, C, Q, N. The consumer therefore receives a finite upper-sum route rather than a host supremum, an open-cover compactness proof, a selected limiting mesh, or a quotient function-equality principle.

Proof. Project the accepted carrier. The rows upstream of every exported upper-sum read are P, B, D, S, I, R with structural support H, C, Q, N.

For a Darboux-facing read, the handoff row I is reachable only after the partition, upper-bound ledger, dyadic step row, and finite sum row have been replayed. For a Riemann-facing read, the row S is already the displayed finite sum built from those same cells and upper values.

Since the packet has no additional analytic coordinate, every listed consumer reads the same finite route and cannot bypass it through a host supremum, compactness theorem, selected mesh limit, quotient equality, or ambient completeness. □

Closure status for NameCert$_{??}$**.**

Constructive story: An **??** packet is generated from finite partition rows, cell upper-bound ledgers, dyadic step-function rows, Riemann-sum rows, DarbouxIntegral handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4083.1, Definition 4083.2, and Definition 4083.3 seed the finite upper-sum packet over displayed partition cells, cell upper-bound reads, dyadic-step rows, finite-sum rows, Darboux handoff, Real$^{\uparrow}$ comparison, transport, replay, provenance, local naming, and consumer-boundary rows for Darboux, Riemann, DyadicStepFunction$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, open-cover compactness, selected mesh limits, lower Darboux sums, lower-sum gap data, quotient function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-refinement stability, upper-bound ledger transport, dyadic step readback, Darboux handoff exactness, Riemann-sum consumer boundary, componentwise transport, displayed replay, provenance, local naming, local NameCert rows, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.UpperDarbouxSumUp.

Chapter 3104

A Concrete Naming Certificate for $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$

Following Kuratowski, the $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ packet records a constructive compactness-defect ledger by finite diameter-cover and finite-net rows rather than by a host infimum over all covers. It sits over the complete metric boundary of chapter 128, the total-bounded finite-net route of chapter 127, the compact metric consumer of chapter 129, the finite epsilon-net witness of chapter 1767, and the dyadic tolerance ledger of chapter 291. The packet names a finite compactness-defect surface for later compactness and functional-analysis consumers; it is not an open-cover compactness theorem, not a choice-selected cover, not a quotient hyperspace, and not an ambient real infimum principle.

Definition 3104.1 (Kuratowski measure of noncompactness carrier)**.** *A* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, C, T, E, D, V, H, R, P, N)$$

with the following visible rows:

1. *X is the displayed* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose bounded subset is being inspected.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row exposed before completion-facing consumers read any defect claim.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net comparison row, recording the tolerance-indexed route by which defect zero may be compared with total boundedness.*
4. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row, listing the displayed centers, tolerance cell, and coverage ledger available at the inspected scale.*
5. *D is the finite diameter-cover row, recording the listed cover pieces and their displayed dyadic diameter bounds.*
6. *V is the compactness-defect value ledger, comparing the requested dyadic threshold with the finite diameter and finite-net rows without taking a host infimum.*
7. *H records componentwise* hsame *transport for the source, completeness, total-boundedness, finite-net, diameter-cover, value, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a compactness-defect request through $X, C, T, E, D, V, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, C, T, E, D, V, H, R, P, N$. It has no coordinate for an open cover, a selected compactness proof, a host powerset, a quotient compactum, a chosen finite subcover, countable choice, or an ambient real infimum.

Lean4 ✓ `BEDC.Derived.KuratowskiMeasureNoncompactnessUp.KuratowskiMeasureNoncompactnessTasteGate`
`single carrier alignment`

Theorem 3104.2 (Kuratowski measure of noncompactness NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *carrier $K = (X, C, T, E, D, V, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, complete-metric source exposure, total-bounded finite-net comparison, finite epsilon-net witness visibility, finite diameter-cover exactness, compactness-defect value readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every compactness or functional-analysis consumer of the packet must read the displayed route

$$X, C \longmapsto T, E, D, V \longmapsto H, R, P, N$$

before treating the defect ledger as compactness-facing evidence.

Proof. Project the carrier of Definition 4084.1. The only source and completion-facing rows are X and C, and the only compactness-defect rows are the finite total-boundedness, finite-net, diameter-cover, and value ledgers T, E, D, V.

The finite-net and diameter-cover rows are displayed before the value ledger is read. Thus a consumer sees centers, coverage data, cover pieces, dyadic diameter bounds, and the requested threshold as finite rows rather than as a hidden open-cover or infimum argument.

Finally H, R, P, N transport, replay, source, and name the same finite route. Since the carrier has no open-cover, selected subcover, host powerset, quotient compactum, countable-choice, or ambient-infimum coordinate, those objects cannot be exported by the local classifier. □

Theorem 3104.3 (Kuratowski measure complete-metric handoff)**.** *Let $K = (X, C, T, E, D, V, H, R, P, N)$ be an accepted* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *packet. Any completion-facing consumer that reads the compactness-defect value ledger V must first expose the complete-metric row C, the total-boundedness comparison row T, and the finite epsilon-net row E, and must then replay the same finite diameter-cover row D through H, R, P, N before reaching a* $\mathsf{CompactMetric}^{\uparrow}$ *or functional-analysis consumer.*

The handoff exports only a finite compactness-defect packet over complete metric and finite-net rows. It does not claim open-cover compactness, selected finite subcovers, quotient compacta, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *infimum.*

Proof. Start from the accepted carrier in Definition 4084.1. The value row V is downstream of X, C, T, E, D, so it cannot be read before the complete-metric source and finite compactness rows have been displayed.

Use the finite epsilon-net route of chapter 1767. The row E supplies centers, tolerance, and coverage data, while T records the total-bounded comparison and D records the finite diameter cover at the inspected scale.

The replay rows H, R, P, N preserve that route when a $\mathsf{CompactMetric}^{\uparrow}$ or functional-analysis consumer asks for compactness-facing evidence. Because no carrier coordinate stores an open

cover, selected subcover, quotient compactum, countable-choice schedule, or ambient infimum, the consumer receives only the displayed compactness-defect handoff. □

Falsifiable prediction: Every accepted $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ compactness-defect read factors through one complete metric source row, one total-boundedness comparison row, one finite epsilon-net witness, one finite diameter-cover ledger, one dyadic value ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it can be consumed by a compact metric or functional-analysis route.

Independence witness: completemetric, totallybounded, compactmetric, finite-epsilon-net, dyadicratcore: $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ is not bijective to any listed sibling because those siblings separately supply completeness, finite-net coverage, compact-metric consumption, and dyadic tolerance rows, while this packet binds them to one compactness-defect value ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ packet is generated from a metric or complete-metric source row, total-bounded finite-net comparison, finite epsilon-net witness, finite diameter-cover ledger, dyadic compactness-defect value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4084.1, Definition 4084.2, and Definition 4084.3 seed the constructive compactness-defect packet over complete metric, finite-net, diameter-cover, dyadic value, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, choice-selected finite subcovers, quotient compacta, countable choice, host powersets, ambient Real infima, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric handoff, total-bounded finite-net comparison, finite epsilon-net witness visibility, finite diameter-cover exactness, compactness-defect value readback, structural transport, and non-escape for BEDC.Derived.KuratowskiMeasureNoncompactnessUp.

Chapter 3105

A Concrete Naming Certificate for ??

Following Plotkin-style structural operational semantics, the **??** packet records a finite replay of displayed Cont execution rows as BEDC NameCert data. It realizes the computation-as-continuation reading of chapter 9365 between the small-step ledger of chapter 469, the monadic continuation surface of chapter 485, and the unary composition substrate of chapter 931. The packet supplies a reusable execution replay carrier for later operational and MetaCIC chapters without importing a host evaluator, quotient trace equality, global confluence, or an external normalization oracle.

Definition 3105.1 (Continuation replay carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$R = (S, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite list of* Cont *step rows, each naming its source history, transition mark, tail context, and target history as displayed BEDC data.*
2. *T is the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger that reads the same ordered steps as a small-step trace.*
3. *E is the endpoint row, recording the initial history, terminal history, and boundary read reached by replaying the listed steps.*
4. *H records componentwise* hsame *transport for step rows, endpoint rows, trace ledger, replay route, provenance, and local naming row.*
5. *C records displayed* Cont *replay from the initial endpoint through every row of S to the terminal endpoint, with the unit and composition route readable by* $\mathsf{ContinuationMonad}^{\uparrow}$ *and* $\mathsf{UnaryContMonoid}^{\uparrow}$ *consumers.*
6. *P is the* Pkg *provenance over **??**,* $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{ContinuationMonad}^{\uparrow}$, $\mathsf{UnaryContMonoid}^{\uparrow}$, *chapter 9365,* BHist, hsame, Cont, *and* NameCert.
7. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The classifier reads accepted packets only through S, T, E, H, C, P, N. It has no coordinate for a host evaluator, quotient trace equality, infinite execution object, global confluence theorem, external normalizer, or hidden monadic semantics.

Lean4 ✓ `BEDC.Derived.ContinuationReplayUp.TasteGate.ContinuationReplayTasteGate single carrier alignment`

Theorem 3105.2 (Continuation replay NameCert obligations)**.** *For every accepted ?? carrier*

$$R = (S, T, E, H, C, P, N),$$

the local NameCert$_{??}$ *surface consists exactly of carrier admission, step-row exposure, endpoint exposure,* ContinuationTrace$^{\uparrow}$ *ledger readback, endpoint transport by* hsame*, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host evaluators, quotient traces, infinite execution objects, global confluence, external normalization, and hidden monadic semantics.*

Proof. Carrier admission is the projection of Definition 4085.1. The accepted packet displays only the step list, trace ledger, endpoint row, componentwise transport, replay route, provenance, and local name.

Step-row exposure and endpoint exposure are finite. The row S lists the displayed source, mark, tail, and target data for each step, while E records only the initial and terminal histories reached by replaying that list.

The trace and algebraic consumers read the same route. The ContinuationTrace$^{\uparrow}$ row T supplies small-step readback, and the replay row C supplies the displayed continuation composition route available to ContinuationMonad$^{\uparrow}$ and UnaryContMonoid$^{\uparrow}$ consumers.

Finally, H, P, N transport, source, and name this same finite route. Since no carrier coordinate stores a host evaluator, quotient trace equality, infinite execution, global confluence, external normalization, or hidden monadic semantics, the ledger refusal clauses exhaust the remaining obligation surface. □

Theorem 3105.3 (Continuation replay endpoint transport)**.** *Every endpoint-facing consumer of an accepted ?? carrier $R = (S, T, E, H, C, P, N)$ receives the endpoint row E only after reading the displayed step rows S, the* ContinuationTrace$^{\uparrow}$ *ledger T, and the replay route C, with componentwise transport, provenance, and local naming supplied by H, P, N. The consumer therefore reads finite* Cont *execution rather than a host evaluator graph, quotient trace, global confluence certificate, or external normalizer.*

Lean4 ✓ `BEDC.Derived.ContinuationReplayUp.ContinuationReplayCarrier endpoint transport`

Proof. Project the accepted carrier. The endpoint row E is carried with the same step list S, trace ledger T, replay route C, and structural rows H, P, N.

Replay precedes endpoint export. The route C starts from the initial endpoint, consumes each displayed step row in order, and reaches the terminal endpoint named by E; transport through H preserves this endpoint relation componentwise.

The sibling consumers cannot bypass the route. ContinuationTrace$^{\uparrow}$ reads T as the small-step ledger, while ContinuationMonad$^{\uparrow}$ and UnaryContMonoid$^{\uparrow}$ read the same replay through unit and composition rows. The excluded host objects are not fields of the packet. □

Closure status for NameCert$_{??}$**.**

Constructive story: A **??** packet is generated from finite Cont step rows, ContinuationTrace$^{\uparrow}$ ledgers, endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4085.1, Definition 4085.2, and Definition 4085.3 seed the finite continuation replay packet over displayed continuation steps, trace readback, endpoint exposure, componentwise transport, replay, provenance, local naming, and endpoint-boundary rows for ContinuationTrace$^{\uparrow}$, ContinuationMonad$^{\uparrow}$, UnaryContMonoid$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host evaluator graphs, quotient trace equality, infinite execution objects, global confluence, external normalization, hidden monadic semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, step-row exposure, endpoint exposure, ContinuationTrace$^{\uparrow}$ readback, hsame endpoint transport, replay exactness, replay stability, componentwise transport, displayed Cont replay, provenance, local naming, local NameCert rows, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.ContinuationReplayUp.

Chapter 3106

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ packet records the theorem-level exit surface for regular Cauchy completeness as finite BEDC NameCert data. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the limit operator of chapter 3392, the uniqueness route of chapter 3390, the located-limit route of chapter 4049, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion phase one theorem object whose rows bind construction, uniqueness, locatedness, and Real sealing without quotient streams, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3106.1 (Cauchy completeness theorem carrier)**.** *A* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, Q, L, U, A, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read by every regular Cauchy estimate.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the requested observation windows.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row associated to those windows and tolerances.*
4. *L is the* $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ *row that turns the displayed regular Cauchy data into a Real-facing limit route.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row ensuring that the Real seal does not depend on the chosen regular representative.*
6. *A is the* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *row exposing the located-limit boundary for the same displayed regular Cauchy name.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q, L, U, A have been displayed.*
8. *H records componentwise* hsame *transport for the tolerance, window, readback, limit-operator, uniqueness, located-limit, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through*

$$D, S, Q, L, U, A, R \longmapsto \mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyCompletenessTheoremUp.CauchyCompletenessTheoremUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletenessTheoremUp.CauchyCompletenessTheoremTasteGate single carrier alignment`

Theorem 3106.2 (Cauchy completeness theorem NameCert obligations)**.** *For an accepted* CauchyCompletenessThe *carrier*

$$T = (D, S, Q, L, U, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, dyadic tolerance accounting through D, finite-window inspection through S, regular rational readback through Q, limit construction through L, representative independence through U, located-limit boundary control through A, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessTheoremUp.TasteGate.CauchyCompletenessTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4086.1. The accepted tuple exposes exactly the dyadic ledger, finite windows, regular readback, limit-operator row, uniqueness row, located-limit row, terminal Real seal, and the structural transport, replay, provenance, and naming rows.

The constructive part is exhausted by D, S, Q, L. The dyadic ledger controls the tolerance requests, the stream row bounds the inspected windows, the regular readback supplies the Cauchy name, and the limit operator is the only route from that name to a Real-facing limit.

The theorem-level seal then reads U, A, R. The uniqueness row makes the seal independent of the displayed representative, the located-limit row records the location boundary, and the Real row is available only after both routes have been displayed. Thus no quotient stream, selected host limit, choice schedule, trichotomy decision, or ambient-completeness coordinate remains available as a NameCert obligation. □

Theorem 3106.3 (Cauchy completeness theorem regular seal route)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyLimitOperator}^{\uparrow} \longmapsto \mathsf{RegularCauchyUniqueness}$$

The consumer receives the terminal $\mathsf{Real}^{\uparrow}$ *seal only as the endpoint of that finite route.*

Proof. Begin with an accepted packet T. Its non-structural rows require the consumer to open the dyadic ledger, finite windows, and regular rational readback before any completion-facing row is visible.

Next the route passes through the regular Cauchy limit operator. The operator supplies the finite construction of the Real-facing limit, while the uniqueness and located-limit rows constrain the same seal by representative independence and locatedness.

The terminal row R is therefore downstream from D, S, Q, L, U, A. Since H, C, P, N only transport, replay, source, and name that route, every L10 phase-exit read factors through the stated chain and cannot bypass it by using quotient streams, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting the DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RegularCauchyLimitOperator row, RegularCauchyUniqueness row, RegularCauchyLocatedLimit row, transport row, replay row, provenance row, or local naming row refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, regular-cauchy-limit-operator, regular-cauchy-uniqueness, regular-cauchy-located-limit, real: $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds tolerance, window, regular readback, limit construction, uniqueness, locatedness, terminal Real sealing, and the structural rows into one theorem-level L10 exit certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular readback, RegularCauchyLimitOperator route, RegularCauchyUniqueness route, RegularCauchyLocatedLimit route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4086.1, Definition 4086.2, and Definition 4086.3 seed the L10 Real and Completion theorem exit surface over regular Cauchy construction, uniqueness, locatedness, terminal sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance accounting, finite-window inspection, regular readback, limit-operator routing, uniqueness routing, located-limit routing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletenessTheoremUp.

Chapter 3107

A Concrete Naming Certificate for $\mathsf{CauchyModulusMajorant}^{\uparrow}$

Following Bishop and Bridges on explicit moduli in constructive analysis, the $\mathsf{CauchyModulusMajorant}^{\uparrow}$ packet records the finite certificate that turns an arbitrary displayed Cauchy threshold schedule into a monotone majorant before completion consumers inspect the seal. It sits after the modulus rows of chapter 292, reads dyadic tolerances from chapter 291, reads finite windows from chapter 126, supplies regular rational schedules for chapter 301, and keeps the terminal real read public only through chapter 38.

Definition 3107.1 (Cauchy modulus majorant carrier)**.** *A* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, D, S, R, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the incoming* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested precisions and majorant bounds;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming which threshold windows may be inspected;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed by regular rational sequence certificates;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite schedule and dyadic comparison rows have been displayed;*
6. *G is the monotone majorant graph row, recording a displayed bound for every inspected threshold window and recording monotonicity as finite comparison data;*
7. *H records componentwise* hsame *transport for the incoming modulus, dyadic ledger, stream windows, regular readback, real seal, majorant graph, continuation, provenance, and local naming rows;*
8. *C records displayed* Cont *replay through the same rows in the order seen by consumers;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, hidden convergence witnesses, host-level supremum selection, or a real-number read that bypasses the displayed majorant route.

Theorem 3107.2 (Cauchy modulus majorant NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *carrier* $M = (A, D, S, R, E, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission, incoming modulus exposure through* A*, dyadic tolerance exposure through* D*, finite-window exposure through* S*, regular readback through* R*, real sealing through* E*, majorant graph exposure through* G*, componentwise transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMajorantUp.TasteGate.CauchyModulusMajorantTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4087.1. The tuple lists exactly the incoming modulus schedule, dyadic tolerance ledger, stream windows, regular readback, real seal, majorant graph, transport, replay, provenance, and local naming rows.

The majorant obligation is displayed by G. It records a finite bound for each inspected threshold window and records monotonicity as comparison data against the dyadic ledger D.

The consumer route is then forced by C. A RegSeqRat or Real consumer reads S, D, G, R before reaching E, and H transports only the displayed coordinates componentwise.

Since no coordinate for countable choice, hidden convergence witnesses, host supremum selection, or bypassed real reads appears in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 3107.3 (Cauchy modulus majorant normalization route)**.** *Let* $M = (A, D, S, R, E, G, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *packet. Every consumer that replaces an arbitrary Cauchy threshold by a monotone schedule must route the replacement through the displayed graph row* G *after reading the finite windows* S *and the dyadic tolerance ledger* D*, and before exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Proof. Start from the carrier definition. The only row that records the monotone majorant is G, while S supplies the inspected windows and D supplies the tolerance comparisons.

The replay row C orders the consumer route through S, D, G, R, E. Thus a regular rational sequence consumer sees the majorant schedule before it receives the readback row R, and a real-number consumer sees the same route before the seal E is opened.

The transport row H is componentwise, so it cannot replace the displayed graph by a host-level monotone choice. The provenance row P records the same dependencies, and N names the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusMajorant}^{\uparrow}$ packet is generated from an incoming Cauchy modulus schedule, DyadicRatCore tolerance comparisons, StreamName finite windows, RegSeqRat readback, a Real seal, a displayed monotone majorant graph, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$, seed)

Scope closed: Definition 4087.1, Definition 4087.2, and Definition 4087.3 seed the route that replaces arbitrary Cauchy thresholds by a monotone majorant before RegSeqRat and Real consumers inspect the seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, choice-free extraction from every external modulus presentation, completion uniqueness, real-order completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, majorant graph exposure, finite-window routing, dyadic comparison routing, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusMajorantUp.

Chapter 3108

A Concrete Naming Certificate for ??

Following Grothendieck's tensor-norm program and Banach-space tensor products in Ryan and Defant–Floret, the **??** packet records the projective tensor norm as finite BEDC NameCert data over displayed tensor-product, norm, Banach, and bounded-operator rows. It is grounded in the algebraic tensor-product surface of chapter 87, the norm row of chapter 89, the Banach carrier of chapter 90, the bounded-linear-operator interface of chapter 1644, and the Banach-algebra consumer surface of chapter 2189. The packet names only the displayed finite decomposition ledgers and their norm bounds; it is not a host infimum, a quotient completion primitive, a selected optimal decomposition, or a choice principle.

Definition 3108.1 (Projective tensor norm carrier)**.** *A **??** carrier over displayed Banach rows E, F is a finite* BHist *packet*

$$\Pi = (E, F, T, D, B, Q, O, H, C, P, N)$$

with the following rows:

1. *E and F are the source* $\mathsf{Banach}^{\uparrow}$ *rows, including their displayed* $\mathsf{Norm}^{\uparrow}$ *and regular-Cauchy surfaces.*
2. *T is the algebraic* $\mathsf{TensorProduct}^{\uparrow}$ *source row over the carried endpoints of E and F.*
3. *D is the finite decomposition ledger: each entry names a finite list of elementary tensors whose displayed tensor sum represents the inspected tensor row.*
4. *B is the crossnorm-bound ledger, recording for each displayed decomposition the finite scalar estimate obtained from the product of the visible source norms.*
5. *Q is the quotient-facing boundary row, stating that equality of tensor representatives is read only through the tensor-product classifier and the displayed decomposition ledger.*
6. *O is the bounded-operator handoff row used by bilinear and operator consumers; it reads through* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *only after T, D, B, Q are displayed.*
7. *H records componentwise* hsame *transport for Banach endpoints, tensor-product rows, decomposition ledgers, bound ledgers, quotient-boundary rows, operator handoff rows, replay data, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$\mathsf{TensorProduct}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Banach}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \textit{??}.$$

9. *P is the* Pkg *provenance over ??,* $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier exports only finite decomposition and bound data. It has no coordinate for a host infimum over all decompositions, a completed quotient tensor product, a selected minimizing decomposition, ambient Banach-space choice, or hidden operator-space equality.

Theorem 3108.2 (Projective tensor norm NameCert obligations)**.** *For every accepted ?? carrier* $\Pi = (E, F, T, D, B, Q, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{??}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed Banach endpoints, tensor-product row, finite decomposition ledger, crossnorm-bound ledger, quotient-boundary row, bounded-operator handoff row, transport, replay, provenance, and local naming;*
2. *crossnorm bound, requiring each elementary tensor read to pass through the displayed source norms and each finite decomposition estimate to pass through B;*
3. *finite decomposition ledger, requiring every inspected tensor norm claim to name a finite decomposition in D before any bound is exported;*
4. *Banach completion handoff, allowing completion-facing consumers to use only the displayed Banach rows and the later completion boundary of Definition 4088.4;*
5. *nonescape from quotient completions, host infima, selected decompositions, cofinal choice, and ambient operator-space equality.*

Lean4 ✓ `BEDC.Derived.ProjectiveTensorNormUp.TasteGate.ProjectiveTensorNormTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4088.1. The endpoint rows E, F, the tensor-product row T, the decomposition ledger D, and the bound ledger B are visible before any projective-norm read is public.

The crossnorm obligation is finite. An elementary tensor estimate reads the two source norms from E and F, and a general displayed tensor estimate reads the finite list in D together with the scalar comparison rows stored in B.

The quotient-facing and completion-facing material is boundary data. The row Q permits only classifier readback for representatives already displayed by T and D, while O supplies the bounded-operator handoff without adding a completed quotient object.

Finally H, C, P, N transport, replay, source, and name only these rows. Since no carrier coordinate stores a host infimum, a selected optimizing decomposition, a quotient completion, cofinal choice, or ambient operator equality, the listed rows exhaust the local NameCert surface. □

Theorem 3108.3 (Projective tensor norm crossnorm boundary)**.** *Every elementary tensor read accepted by a ?? packet exposes the route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{TensorProduct}^{\uparrow} \longmapsto E, F, T, D, B$$

before a scalar bound is exported. The boundary identifies the displayed projective-norm estimate with finite source-norm products from E and F, not with a host infimum, a quotient representative, or a selected decomposition of minimal value.

Proof. Start with an elementary tensor request. The carrier of Definition 4088.1 forces the request to read the Banach endpoint rows E, F and the tensor-product row T before the decomposition ledger is inspected.

For elementary tensors the relevant ledger entry in D is a singleton decomposition, and B records the displayed product of the two source norm reads. For finite sums, the same rows are read entry by entry and then combined by the finite scalar comparison stored in B.

No additional optimization step is available. The packet has no coordinate for a host infimum over all decompositions, no quotient representative selector, and no choice row that could pick a preferred decomposition. □

Theorem 3108.4 (Projective tensor norm Banach completion handoff)**.** *Every Banach-completion or bounded-operator consumer of an accepted ?? packet factors through the displayed route*

$$E, F, T, D, B, Q, O, H, C, P, N.$$

It may use the finite projective-bound ledger as input to later $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, *or* $\mathsf{BanachAlgebra}^{\uparrow}$ *rows, but it receives no completed quotient tensor product, no universal completion object, no selected Cauchy representative, and no ambient completeness theorem.*

Proof. Use the obligation surface of Definition 4088.2. It places every completion-facing read after carrier admission, finite decomposition admission, crossnorm-bound admission, and quotient-boundary readback.

The bounded-operator handoff is exactly the row O. It can route bilinear or operator consumers through the displayed tensor and bound ledgers, but it does not add a separate operator-space quotient or a hidden complete tensor product.

Completion-facing consumers therefore receive only the finite data that the later Banach or operator packet can cite: endpoint Banach rows, tensor-product rows, finite decomposition rows, scalar bound rows, transport, replay, provenance, and local naming. The excluded completion and selection principles have no source coordinate in Π. □

Falsifiable prediction: Every accepted **??** packet must expose Banach endpoints, a TensorProductUp row, a finite decomposition ledger, a crossnorm-bound ledger, a quotient-boundary row, a bounded-operator handoff row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any projective tensor norm read is public.

Independence witness: tensorproduct, norm, banach, boundedlinearoperator, banachalgebra: **??** is not bijective to any listed sibling because it jointly carries finite tensor decompositions, crossnorm-bound ledgers, quotient-boundary readback, Banach-completion handoff, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from displayed Banach endpoint rows, a TensorProductUp source row, finite decomposition ledgers, crossnorm-bound ledgers, quotient-boundary rows, bounded-operator handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{??}$, seed)

Scope closed: Definition 4088.1, Definition 4088.2, Definition 4088.3, and Definition 4088.4 seed the finite projective tensor norm surface over tensor-product, norm, Banach, bounded-operator, and Banach-algebra rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed projective tensor products, host infima over all decompositions, selected optimizing decompositions, quotient completions, cofinal choice, or ambient operator-space equality.

Upgrade path: Obligation closure requires checked carrier admission, crossnorm-bound exactness, finite decomposition ledger coverage, Banach-completion handoff, quotient-boundary readback, transport, replay, provenance, and local naming for BEDC.Derived.ProjectiveTensorNormUp.

Chapter 3109

A Concrete Naming Certificate for $\mathsf{MetricSeparationReflector}^{\uparrow}$

Following Bishop–Bridges constructive metric separation and Kelley-style uniform separation, the $\mathsf{MetricSeparationReflector}^{\uparrow}$ packet records the finite reflector that turns pseudometric zero-distance evidence into separated metric identity without quotient primitives. It sits after the pseudometric carrier of chapter 1614, the separated metric classifier of chapter 1411, and the metric surface of chapter 57. Its completion handoff is read only through chapter 95 and the L10 real-name spine of chapter 291, chapter 126, chapter 301, and chapter 38.

Definition 3109.1 (Metric separation reflector carrier)**.** *A* $\mathsf{MetricSeparationReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *source row, carrying point observations, finite distance comparisons, and zero-distance witnesses.*
2. *Z is the zero-distance kernel row, recording exactly which displayed pairs of P are admitted as indistinguishable by the pseudometric evidence.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *identity row, consuming Z before any point classifier is allowed to identify two observations.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row, exposing the separated metric consumer surface reached after S.*
5. *C is the* $\mathsf{Completion}^{\uparrow}$ *route row for completion-facing consumers.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to inspect finite zero-distance budgets.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule attached to the same finite tolerance requests.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from D and W.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which numerical zero-distance evidence is read.*
10. *H records componentwise* hsame *transport for source rows, kernel rows, separated identity rows, metric handoffs, completion handoffs, tolerance ledgers, stream observations, regular rational readbacks, Real seals, replay data, provenance, and local naming.*
11. *K records displayed* Cont *replay along the same route.*

12. *G is the* Pkg *provenance over* $\mathsf{MetricSeparationReflector}^{\uparrow}$, $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, Z, S, M, C, D, W, Q, A, H, K, G, N$. It has no coordinate for quotient classes, selected representatives, host equality of points, countable choice, ambient completion, or an external real-line equality oracle.

Lean4 ✓ `BEDC.Derived.MetricSeparationReflectorUp.MetricSeparationReflectorTasteGate single carrier alignment`

Theorem 3109.2 (Metric separation reflector NameCert obligations)**.** *For every accepted* MetricSeparationReflector *carrier $R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the pseudometric source, zero-distance kernel, separated identity, metric handoff, completion route, dyadic tolerance, stream observation, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *kernel exposure, requiring every identity-facing request to display the Z row before it can reach the separated metric row S;*
3. *reflector exactness, requiring S to identify precisely the displayed zero-distance pairs admitted by Z and no point pairs outside that row;*
4. *metric and completion handoff, requiring M before metric consumers and C before completion consumers inspect the separated result;*
5. *numerical readback discipline, requiring D, W, Q, A in that order for any finite tolerance or Real-facing zero-distance read;*
6. *structural preservation through H, K, G, N;*
7. *ledger non-escape, excluding quotient classes, selected representatives, host point equality, countable choice, hidden ambient completion, and external Real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4089.1. The accepted packet exposes the pseudometric source, zero-distance kernel, separated identity, metric and completion handoffs, numerical readback rows, and structural rows.

The reflector clause is forced by the route through Z and S. A pseudometric comparison can become point identity only after the corresponding zero-distance pair has appeared in Z, and S is the only displayed row that exports separated identity.

Metric and completion consumers are downstream of that separated identity row. The metric handoff M and completion row C receive only the reflected classifier, while D, W, Q, A expose the finite tolerance and Real-facing readback route used to inspect zero distance.

Finally H, K, G, N preserve, replay, source, and name exactly these displayed rows. Since no excluded object appears as a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3109.3 (Metric separation reflector zero-distance kernel)**.** *Let $R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N$ be an accepted* $\mathsf{MetricSeparationReflector}^{\uparrow}$ *carrier. Every identity exported to a* $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{Pseudometric}^{\uparrow} \longmapsto Z \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqR}$$

Thus zero distance is the displayed kernel of the reflector. The packet exports separated classifier identity, not quotient equality, selected representatives, host point equality, countable choice, or an ambient completion theorem.

Proof. Start with an identity-facing consumer of an accepted packet. The carrier definition gives P as the only pseudometric source row and Z as the only zero-distance kernel row, so the request must first expose the displayed kernel evidence.

The separated identity row S consumes that kernel evidence before any metric or completion consumer can inspect the result. Rows M and C then carry the separated classifier into the metric and completion surfaces.

The numerical route is also explicit. A zero-distance tolerance read passes through D, the finite observation schedule W, the regular rational readback Q, and the terminal Real seal A.

Transport, replay, provenance, and local naming in H, K, G, N preserve this route componentwise. No quotient class, selected representative, host equality proof, choice schedule, or ambient completion coordinate is available to bypass the displayed kernel. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSeparationReflector}^{\uparrow}$ packet is generated from PseudometricUp source evidence, a zero-distance kernel row, a SeparatedMetricUp identity row, MetricSpaceUp and CompletionUp handoffs, DyadicRatCoreUp tolerances, StreamNameUp observations, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4089.1, Definition 4089.2, and Definition 4089.3 seed the finite reflector from pseudometric zero-distance evidence to separated metric identity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient primitives, selected representatives, host point equality, countable choice, ambient completion, or external Real equality.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance kernel exposure, separated identity exactness, metric and completion handoff, numerical readback, transport preservation, replay, provenance, and non-escape for BEDC.Derived.MetricSeparationReflectorUp.

Chapter 3110

A Concrete Naming Certificate for ??

The **??** packet realizes the local suffix-support rank theorem of Definition 9462.1756 and Definition 9462.1757 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the boundary classifier of chapter 3393, the coverage-rank surface of chapter 3470, and the codon lift-orbit classifier of chapter 4059. The packet records the stop/Trp face as the local two-coordinate square whose proper-support vertices are the stop row and whose full-support corner is the retained Trp row; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden global translation table.

Definition 3110.1 (Stop/Trp face rank-deficiency carrier)**.** *A* ***??*** *carrier is a finite* BHist *packet*

$$F = (W, B, C, O, S, R, L, H, K, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six-bit codon coordinates.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *stop/Trp boundary row, selecting the* U+RR *face* $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}$
3. *C is the* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row that keeps the visible boundary support inside the displayed tile surface.*
4. *O is the* CodonLiftOrbitClassifier$^{\uparrow}$ *row that records the finite orbit-readback context for the same codon window.*
5. *S is the local suffix-support table row based at* UAA*, with supports $\varnothing$, $\{f_3\}$, $\{f_2\}$, and $\{f_2, f_3\}$.*
6. *R is the rank-deficiency row, recording that stop vertices are exactly the rows whose support is a proper subset of $\{f_2, f_3\}$.*
7. *L is the ledger row separating the full-support* UGG *corner from the three proper-support stop vertices.*
8. *H records componentwise* hsame *transport for source windows, boundary tiles, coverage rows, orbit-classifier rows, suffix-support rows, rank rows, ledger rows, replay, provenance, and local naming.*
9. *K records displayed* Cont *replay from a stop-face query through W, B, C, O, S, R, L, H.*
10. *P is the* Pkg *provenance over* ***??***, CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$ CodonLiftOrbitClassifier$^{\uparrow}$, *Definition 9462.1757,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{??}$ *row.*

The classifier accepts a stop-face rank read only through $W, B, C, O, S, R, L, H, K, P, N$. It has no coordinate for a hidden genetic-code table, probabilistic dynamics, quotient codon equality, biological causation, or a global semantic equivalence.

Lean4 ✓ `BEDC.Derived.StopTrpFaceRankDeficiencyUp.StopTrpFaceRankDeficiencyTasteGate single carrier alignment`

Theorem 3110.2 (Stop/Trp face rank-deficiency NameCert obligations)**.** *For every accepted ?? carrier $F = (W, B, C, O, S, R, L, H, K, P, N)$, the local* NameCert$_{??}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the codon-window, boundary-tile, coverage, lift-orbit, suffix-support, rank-deficiency, ledger, transport, replay, provenance, and local naming rows;*
2. *face exposure, requiring the stop/Trp read to pass through W and B before any support rank is inspected;*
3. *coverage and orbit compatibility, requiring C and O to preserve the same displayed codon-window context as W and B;*
4. *suffix-rank exactness, requiring S and R to classify* UAA, UAG, *and* UGA *as proper-support rows and* UGG *as the unique full-support row;*
5. *separated ledger discipline, requiring L to keep stop legality and Trp retention as different displayed rows;*
6. *structural preservation through H, K, P, N;*
7. *ledger non-escape, excluding hidden biological tables, stochastic mechanisms, quotient codon equality, global semantics, and external translation classifiers.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4090.1. The accepted packet displays the codon-window source, boundary-tile classifier, coverage route, lift-orbit context, suffix-support table, rank-deficiency row, ledger row, and structural rows.

The face exposure clause forces the local read to begin with W and B. Thus the rank table is attached to the displayed stop/Trp face, not to an unrecorded biological table.

Coverage and orbit compatibility are downstream checks on the same finite window. The rows C and O preserve the boundary support and orbit-readback context before S and R classify local suffix support.

The rank exactness row is the finite table of Definition 9462.1756. Proper subsets of $\{f_2, f_3\}$ give UAA, UAG, and UGA, while the full support gives UGG. The ledger row L records this separation without collapsing the full-support corner into the stop row.

Finally H, K, P, N preserve, replay, source, and name only these displayed rows. Since no excluded object is a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3110.3 (Rank-deficiency prototype obligation surface)**.** *Every accepted ?? packet realizes the local rank-deficiency prototype of Definition 9462.1756 and Definition 9462.1757 only through the displayed route*

$$W, B, C, O, S, R, L, H, K, P, N.$$

Equivalently, the prototype is available only through the CodonSixBitWindow$^{\uparrow}$ *source window,* DnaWindowSixBounda *boundary tile,* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row,* CodonLiftOrbitClassifier$^{\uparrow}$ *orbit classifier,*

suffix-support rows, rank-deficiency rows, separated ledger rows, transport, replay, provenance, and local NameCert rows. The packet exports no hidden genetic-code table, quotient codon equality, biological causation, or global semantic equivalence.

Proof. Project the carrier of Definition 4090.1. The displayed rows $W, B, C, O, S, R, L, H, K, P, N$ are the whole accepted packet.

The prototype begins with the six-bit window W and boundary tile B. Coverage C and orbit classification O keep that same finite window visible before any suffix-support or rank-deficiency row is read.

The rows S and R are exactly the local support and rank rows, and L keeps the stop vertices separated from the retained Trp corner. Thus the prototype route is a finite BEDC packet route, not a quotient of codons and not an external genetic-code table.

Finally H, K, P, N transport, replay, source, and name only the displayed rows. Since hidden code tables, biological causation, quotient codon equality, and global semantic equivalence are absent from the carrier, no accepted prototype obligation can pass through them. □

Theorem 3110.4 (Stop/Trp face local rank-deficiency route)**.** *Let $F = (W, B, C, O, S, R, L, H, K, P, N)$ be an accepted **??** carrier. Every accepted stop-face rank read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \mathsf{CodonLiftOrbitClassifier}^{\uparrow}$$

Along this route the vertices UAA, UAG, *and* UGA *are exactly the proper-support rows of $\{f_2, f_3\}$, while* UGG *is the unique full-support corner. The packet exports no hidden code table, stochastic mechanism, quotient codon semantics, or global translation classifier.*

Proof. Start with an accepted stop-face rank read. The carrier requires the read to consume the source window W and the boundary tile B, so the four codons are the displayed $\mathrm{U} + \mathrm{RR}$ face.

The coverage row C and lift-orbit row O keep the same finite codon-window context visible. They do not replace the local face by a global biological semantics.

Now read S and R. By Definition 9462.1757, the proper-support vertices in the two suffix coordinates are precisely UAA, UAG, and UGA, and the remaining UGG corner has full support.

The ledger row L records that separation, and H, K, P, N transport, replay, source, and name the route. No excluded semantic or quotient object appears in the carrier, so the read cannot escape the displayed rank-deficiency packet. □

Falsifiable prediction: A **??** consumer that classifies UAA, UAG, or UGA without displaying the suffix-support row S and rank row R is not reading an accepted packet of this chapter.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, codon-lift-orbit-classifier: **??** is not bijective to any listed sibling because it jointly carries the source window, stop/Trp boundary face, coverage row, orbit-readback context, suffix-support table, rank-deficiency classifier, separated ledger, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.StopTrpFaceRankDeficiencyUp.TasteGate` must provide explicit Nontrivial **??** and FieldFaithful **??** instances preserving $W, B, C, O, S, R, L, H, K, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from CodonSixBitWindowUp source rows, DnaWindowSixBoundaryTileUp stop/Trp boundary rows, SixBoundaryTileCoverageUp coverage rows, CodonLiftOrbitClassifierUp orbit rows, suffix-support rows, rank-deficiency rows, a separated ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4090.1, Definition 4090.2, and Definition 4090.4 seed the finite stop/Trp face rank-deficiency packet over the local suffix-support theorem.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, biological causation, stochastic fitness modeling, quotient codon equality, hidden translation tables, or global semantics.

Upgrade path: Obligation closure requires checked carrier admission, face exposure, coverage and orbit compatibility, suffix-rank exactness, separated ledger discipline, Nontrivial StopTrpFaceRankDeficiencyUp, and FieldFaithful StopTrpFaceRankDeficiencyUp for BEDC.Derived.StopTrpFaceRankDeficiencyUp.

Chapter 3111

A Concrete Naming Certificate for ??

Following Bourbaki/Kelley-style bornology plus Bishop–Bridges metric boundedness, the **??** packet records the metric bornology generated by displayed center-radius bounded-set data. It sits between the bounded-subset carrier of chapter 1757, the center-radius row of chapter 1450, the metric carrier of chapter 57, the radius comparison surfaces of chapter 38 and chapter 766, the ordinary bounded-family closure of chapter 3370, and the completion-facing specialization of chapter 1896. The packet turns metric ball containment and bounded subset evidence into a reusable bornology surface without introducing host suprema, compactness, selected bounds, quotient families, trichotomy, or ambient real completeness.

Definition 3111.1 (MetricBornology carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$M = (X, S, c, r, B, U, E, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier whose distance and point-classifier rows are the only metric data available to the packet.*
2. *S is the bounded-subset classifier row read through the* $\mathsf{BoundedSet}^{\uparrow}$ *packet.*
3. *c is the displayed center or witness row for the metric bound.*
4. *r is the radius row, read through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *B is the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger connecting accepted membership reads of S to the displayed ball with center c and radius r.*
6. *U is the finite-union bornology ledger, recording union requests only over previously accepted metric-bounded families.*
7. *E is the hereditary bounded-family ledger, recording subfamily requests only through a displayed parent family already accepted by S or U.*
8. *D is the consumer handoff row for* $\mathsf{Bornology}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *readers.*
9. *H records componentwise* hsame *transport for $X, S, c, r, B, U, E, D, C, P, N$.*
10. *C records displayed* Cont *replay from a bounded-family request through metric carrier, classifier, center, radius, ball containment, hereditary closure, finite union, handoff, provenance, and local naming.*

11. *P is the* Pkg *provenance over* ??, $\mathsf{Met}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Bornology}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier compares metric-bounded families only through these rows. It has no coordinate for a host supremum, selected bound, compactness proof, open-cover compactness, quotient family, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricBornologyUp.MetricBornologyTasteGate single carrier alignment`

Theorem 3111.2 (MetricBornology NameCert obligations)**.** *For every accepted ?? carrier $M = (X, S, c, r, B, U, E, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{??}$ *surface consists exactly of carrier admission for the metric row, bounded-subset classifier, center or witness row, radius row, metric-ball containment ledger, finite-union ledger, hereditary ledger, consumer handoff, componentwise transport, displayed replay, provenance, and local naming rows.*

Every accepted bounded-family read must first expose the $\mathsf{Met}^{\uparrow}$ *row X, the* $\mathsf{BoundedSet}^{\uparrow}$ *classifier S, the center-radius data c, r, and the* $\mathsf{MetricBall}^{\uparrow}$ *ledger B. Hereditary and finite-union bornology reads may then consume only E and U, and downstream* $\mathsf{Bornology}^{\uparrow}$ *or* $\mathsf{CauchyBornology}^{\uparrow}$ *consumers may read the packet only through D, H, C, P, N.*

Proof. Project the carrier of Definition 4091.1. The metric rows are X, S, c, r, B, and the bornology rows are U, E, D; the remaining rows only transport, replay, source, and name the same packet.

Carrier admission and boundedness are forced by the sibling routes. The bounded-subset packet supplies S, c, r, B, while the metric-ball packet supplies the center-radius comparison over the same metric row X and radius row r.

Hereditary and finite-union reads are then confined to E and U. Since neither row stores a host powerset, selected bound, supremum, compactness proof, quotient family, or choice schedule, the local NameCert obligations are exactly the displayed metric-bounded-family rows. □

Theorem 3111.3 (MetricBornology bounded-set radius handoff)**.** *Let $M = (X, S, c, r, B, U, E, D, H, C, P, N)$ be an accepted ?? carrier. Every* $\mathsf{BoundedSet}^{\uparrow}$ *or* $\mathsf{MetricBall}^{\uparrow}$ *read exported from M follows the route*

$$X, S, c, r, B \longmapsto U, E \longmapsto D, H, C, P, N.$$

The radius row r is read only through $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$*, the containment row B is read only as a* $\mathsf{MetricBall}^{\uparrow}$ *ledger, and the resulting bounded-family surface is handed to* $\mathsf{Bornology}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *consumers before hereditary or finite-union bornology closure can be used downstream.*

Proof. Start with a bounded-set request. By Definition 4091.1, it must name the metric carrier X, the bounded-subset classifier S, the center or witness c, the radius row r, and the metric-ball containment ledger B.

Use the radius anchors next. The row r is available only through the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, and B is available only through the $\mathsf{MetricBall}^{\uparrow}$ center-radius ledger. Thus the request cannot replace the displayed radius by a selected supremum or by a compactness argument.

Finally pass to the bornology rows. The hereditary row E and finite-union row U consume the already displayed metric-bounded families, while D, H, C, P, N export the same surface to $\mathsf{Bornology}^{\uparrow}$ and $\mathsf{CauchyBornology}^{\uparrow}$ readers. No downstream consumer receives a host family, quotient family, selected bound, trichotomy proof, or ambient real completeness datum. □

Closure status for NameCert$_{??}$**.**

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4091.1, Definition 4091.2, and Definition 4091.3 seed the finite metric bornology surface over bounded-set, metric-ball, real-radius, general-bornology, and Cauchy-bornology consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, selected bounds, quotient families, compactness, open-cover compactness, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric carrier admission, bounded-subset classifier exposure, radius comparison, metric-ball containment, hereditary and finite-union ledgers, bornology handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricBornologyUp.

Constructive story: A **??** packet is generated from a MetricSpaceUp carrier, a BoundedSetUp classifier, a displayed center or witness, a RealUp radius read through RealAlgOrderUp, a MetricBallUp containment ledger, hereditary and finite-union bornology ledgers, BornologyUp and CauchyBornologyUp handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3112

A Concrete Naming Certificate for $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's treatment of completion by universal comparison, the $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ packet records the finite BEDC surface on which several constructive completion presentations can be read through one comparison certificate. It sits after the Cauchy-comparison route of chapter 1954, the Bishop completion comparison of chapter 1508, the separated uniqueness ledger of chapter 1463, and the metric uniqueness boundary of chapter 1796. The packet also reads the finite observation spine of chapter 126, chapter 291, chapter 301, and chapter 38. It compares displayed constructive completion routes only; it is not quotient-completion equality, not selected representative equality, and not a proof that $\mathsf{Real}^{\uparrow}$ is ambiently complete.

Definition 3112.1 (Constructive completion equivalence carrier)**.** *A* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, B, U, M, S, D, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row, comparing unit, universal-property, functorial, and metric-completion presentations by finite observation data.*
2. *B is the* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *row, comparing regular located Cauchy data with the completion-facing real-name seal.*
3. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *row, recording the separated uniqueness ledger available to completion consumers.*
4. *M is the* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *row, recording the metric completion comparison boundary over a common metric source.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row common to the completion observations.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and comparison requests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the windows and dyadic ledger.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the comparison, window, tolerance, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for C, B, U, M, S, D, R, A and the local naming row.*
10. *K records displayed* Cont *replay through the comparison rows, observation spine, transport, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $C, B, U, M, S, D, R, A, H, K, P, N$. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConstructiveCompletionEquivalenceUp.ConstructiveCompletionEquivalenceTasteGate`
`single carrier alignment`

Theorem 3112.2 (Constructive completion equivalence NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier*

$$E = (C, B, U, M, S, D, R, A, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$ *surface consists exactly of Cauchy-comparison exposure through C, Bishop-comparison exposure through B, separated uniqueness exposure through U, metric uniqueness exposure through M, finite-window exposure through S, dyadic tolerance exposure through D, regular rational readback through R, terminal real-name sealing through A, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4092.1. The four comparison rows C, B, U, M are the only completion-presentation coordinates, while S, D, R, A are the shared finite observation spine.

The observation spine is read after the comparison rows have been named. Finite windows, dyadic tolerances, and regular rational readback provide the data by which the terminal $\mathsf{Real}^{\uparrow}$ seal is reached.

Finally H, K, P, N transport, replay, source, and name the same packet. Since no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness row, or ambient real-completeness theorem occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3112.3 (Constructive completion equivalence comparison route)**.** *Every completion-equivalence consumer of an accepted* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletionUniquenes}$

as displayed rows before it may compare constructive completion presentations. The route exports only the finite comparison certificate $E = (C, B, U, M, S, D, R, A, H, K, P, N)$ and does not identify quotient completions or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the obligation surface of Definition 4092.2. Any consumer must first expose the Cauchy-comparison, Bishop-comparison, separated uniqueness, and metric uniqueness rows.

The shared observation spine is then consumed through S, D, R, A. These rows give finite windows, tolerance ledgers, regular rational readback, and a terminal real-name seal, so the comparison remains a displayed certificate rather than a quotient completion object.

The structural rows H, K, P, N preserve the same route under transport and replay. Therefore no consumer can bypass the finite comparison packet by citing selected representatives, host equality of completed points, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ read exposes the CauchyCompletionComparison, BishopCompletionComparison, CompletionUniqueness, MetricCompletionUniqueness, StreamName, DyadicRatCore, RegSeqRat, RealUp, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any constructive-completion comparison is consumed.

Independence witness: cauchy-completion-comparison, bishopcompletioncomparison, completion-uniqueness, metric-completion-uniqueness, streamname, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ because this carrier alone binds the four comparison surfaces to one shared finite observation spine and one local NameCert row without quotient-completion equality or ambient RealUp completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ packet is generated from CauchyCompletionComparison, BishopCompletionComparison, CompletionUniqueness, MetricCompletionUniqueness, StreamName, DyadicRatCore, RegSeqRat, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4092.1, Definition 4092.2, and Definition 4092.3 seed the finite comparison certificate for constructive completion presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-row exposure, shared-window coverage, dyadic tolerance exactness, regular readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCompletionEquivalenceUp.

Chapter 3113

A Concrete Naming Certificate for $\mathsf{EffectiveModulus}^{\uparrow}$

Following Bishop-style effective convergence, the $\mathsf{EffectiveModulus}^{\uparrow}$ packet records the finite threshold data by which a regular rational Cauchy name answers a dyadic tolerance request before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the terminal seal. It sits over the regular rational source of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the modulus surface of chapter 292. The packet exposes a finite effective-modulus row, not a host function from tolerances to thresholds.

Definition 3113.1 (EffectiveModulus carrier)**.** *An* $\mathsf{EffectiveModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, S, D, u, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational observations are being controlled.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window family through which R is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the requested dyadic precision, comparison cells, and halving ledger.*
4. *u is the unary tolerance-index row naming the finite precision request inside the same history.*
5. *L is the modulus witness ledger. It records a threshold row saying that all inspected tails of R beyond u remain inside the tolerance read from D.*
6. *H records componentwise* hsame *transport for the regular source, stream windows, dyadic tolerance, unary index, witness ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a dyadic tolerance request through S and R to the witness ledger L.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient equality of real names, selected limits, countable choice, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.EffectiveModulusUp.EffectiveModulusTasteGate single carrier alignment`

Theorem 3113.2 (EffectiveModulus NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveModulus}^{\uparrow}$ *carrier*

$$M = (R, S, D, u, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, dyadic tolerance accounting, unary tolerance-index admission, modulus witness readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted read must display the route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the rows $R, S, D, u, L, H, C, P, N$*.*

Proof. Project the carrier of Definition 4093.1. Its public rows are exactly the regular rational source, stream-window family, dyadic tolerance, unary index, modulus witness ledger, transport, replay, provenance, and local name.

The source-window exposure and tolerance clauses are read through S and D. A tolerance request cannot reach the witness ledger until the same finite observation windows and dyadic comparison cells have been displayed.

The unary index row u supplies the finite threshold coordinate, while L records the corresponding tail-control witness for R. Transport through H and replay through C preserve that source route without adding a second threshold selector.

Finally P and N source and name only the displayed packet. Since the carrier has no quotient, selected-limit, choice, host-equality, or completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3113.3 (EffectiveModulus regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveModulus}^{\uparrow}$ *packet reads the modulus witness ledger only after the regular source row* R*, stream-window family* S*, dyadic tolerance ledger* D*, and unary tolerance index* u *have been exposed. The handoff supplies the displayed* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible threshold row* L *together with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structural rows, and it supplies no quotient real equality, selected representative, countable-choice functional, host real equality, or ambient completeness principle.*

Proof. Begin with an accepted carrier $M = (R, S, D, u, L, H, C, P, N)$. The regular rational readback is the row R, and all admissible observations of it pass through the finite windows S.

Read the dyadic request next. The tolerance row D fixes the comparison scale, and the unary row u names the finite threshold request attached to that scale.

The witness ledger L is then the only modulus-facing handoff row. It is transported through H and replayed through C, while P and N preserve the same provenance and local name.

Thus any regular-Cauchy or real-seal consumer receives a finite threshold surface over $R, S, D, u, L, H, C, P, N$. The excluded quotient, representative-selection, choice, host-equality, and completeness objects are absent from the packet, so they cannot enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffectiveModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName observation windows, a DyadicRatCore tolerance row, a unary tolerance index, a modulus witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4093.1, Definition 4093.2, and Definition 4093.3 seed the finite effective-modulus route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{EffectiveModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window exposure, dyadic tolerance accounting, unary threshold admission, modulus witness readback, regular-Cauchy handoff stability, and RealUp seal non-escape for BEDC.Derived.EffectiveModu

Chapter 3114

A Concrete Naming Certificate for ??

Following Bishop and Bridges and Mac Lane, the **??** packet records the functoriality certificate for separated completion as finite BEDC NameCert data. It sits after the separated-completion surface of chapter 1612, the public Cauchy reflector of chapter 1585, the metric-completion handoff of chapter 1506, and the L10 finite-name route through chapter 126, chapter 301, and chapter 38. The packet names the source-to-target transport of a uniformly continuous map through separated completion without importing quotient completions, chosen limits, host function extensionality, countable choice, ambient completeness, or bridge checking.

Definition 3114.1 (Separated completion functor carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$F = (M, N, f, S, T, R, L, W, Q, A, H, C, P, E)$$

with the following displayed rows:

1. *M is the source metric row, naming the source carrier, metric comparison surface, and finite Cauchy-input budget.*
2. *N is the target metric row, naming the target carrier, metric comparison surface, and completion-facing output budget.*
3. *f is the source map row, recording the displayed uniformly continuous map data consumed by the completion route.*
4. *S is the source* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row, read from the source metric before any completed source name is transported.*
5. *T is the target* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row, read from the target metric before any completed target name is exposed.*
6. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row, recording the reflector handoff that turns source Cauchy data into a completion-facing read.*
7. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, recording the metric-completion consumer surface through which the target completion read is delivered.*
8. *W is the* $\mathsf{StreamName}^{\uparrow}$ *local observation row attached to source and target completion names.*
9. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *local tolerance row consumed after W and before any terminal real-name seal.*
10. *A is the* $\mathsf{Real}^{\uparrow}$ *local seal row reached only through W, Q and the displayed separated-completion rows.*

11. *H records componentwise* hsame *transport for source metric data, target metric data, the source map, separated-completion rows, reflector and metric-completion rows, local L10 rows, replay, provenance, and naming.*
12. *C records displayed* Cont *replay from a source completion request through $M, N, f, S, T, R, L, W, Q, A, H, P, E$.*
13. *P is the* Pkg *provenance over* **??**, SeparatedCompletion$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *E is the local* NameCert$_{??}$ *row.*

The classifier compares accepted packets only through $M, N, f, S, T, R, L, W, Q, A, H, C, P, E$. It has no coordinate for quotient completion equality, selected limit representatives, host function extensionality, countable choice, ambient completeness of either completed metric, or a standard bridge.

Lean4 ✓ `BEDC.Derived.SeparatedCompletionFunctorUp.SeparatedCompletionFunctorTasteGate single carrier alignment`

Theorem 3114.2 (Separated completion functor NameCert obligations)**.** *For every accepted* **??** *carrier*

$$F = (M, N, f, S, T, R, L, W, Q, A, H, C, P, E),$$

the local NameCert$_{??}$ *surface consists exactly of source metric admission, target metric admission, source map admission, separated-completion exposure on both sides, Cauchy-reflector exposure, metric-completion exposure, StreamName observation, RegSeqRat tolerance readback, Real seal readback, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.SeparatedCompletionFunctorUp.SeparatedCompletionFunctorCarrier namecert obligations`

Proof. Project the carrier of Definition 4094.1. The rows M, N, f expose the source metric, target metric, and map data before any completion-facing consumer reads the packet.

The separated-completion rows come next. Rows S and T keep the source and target completion surfaces separated before the reflector row R and the metric-completion row L are consumed.

The local numerical route is finite. The StreamName row W, RegSeqRat row Q, and Real row A display the L10 readback used by completion consumers, while H, C, P, E transport, replay, source, and name the same route.

Since no carrier coordinate contains quotient completion equality, a chosen limit representative, host function extensionality, countable choice, ambient completeness, or a bridge theorem, the listed rows exhaust the local NameCert surface. □

Theorem 3114.3 (Separated completion functor Cauchy-reflector route)**.** *Let $F = (M, N, f, S, T, R, L, W, Q, A, H,$ be an accepted* **??** *carrier. Every completion-facing read of the transported map factors through the route*

$$M, N, f, S, T, R, L, W, Q, A, H, C, P, E$$

before a target completion name is exported. In particular, the Cauchy-reflector row R and metric-completion row L are read only after both separated-completion rows S, T have been displayed, and the local StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows are read only as the terminal L10 route of that same packet.*

Proof. Start with a transported-map request for an accepted carrier. The source and target metric rows M, N and the source map row f are the only input coordinates available to the request.

Read the separated-completion rows S, T before any completion-facing handoff. This forces equality-facing and completion-facing consumers to remain on the separated surfaces carried by the packet.

The Cauchy reflector and metric completion handoffs are then R and L. Their consumers receive the local L10 route only through W, Q, A, so the exported target completion name remains attached to the displayed source map, separated-completion rows, transport, replay, provenance, and local naming.

Because the carrier contains no quotient completion class, selected limit, host function-extensionality proof, countable-choice selector, ambient-completeness row, or bridge theorem, no completion-facing read can bypass the named route. □

Falsifiable prediction: Every accepted **??** transport read displays source metric, target metric, source map, source separated-completion, target separated-completion, CauchyCompletionReflector, MetricCompletion, StreamName, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows before a target completion name is exported.

Independence witness: separatedcompletion, cauchycompletionreflector, metriccompletion, streamname, regseqrat, real: the separated completion functor packet is not bijective to any listed sibling because it jointly carries a source metric row, target metric row, source map row, separated-completion rows on both sides, reflector and metric-completion handoffs, the local L10 readback route, transport, replay, provenance, and local naming.

Closure status for NameCert$_{??}$**.**

Constructive story: A **??** packet is generated from source metric data, target metric data, a source map row, separated-completion rows on both sides, CauchyCompletionReflectorUp and MetricCompletionUp handoffs, StreamNameUp observations, RegSeqRatUp tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4094.1, Definition 4094.2, and Definition 4094.3 seed the finite functorial separated-completion route from a source map through separated completions and the L10 completion readback.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host function extensionality, countable choice, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked source metric admission, target metric admission, source map admission, separated-completion exposure on both sides, Cauchy-reflector handoff, metric-completion handoff, StreamName observation, RegSeqRat tolerance readback, Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionFunctorUp.

Chapter 3115

A Concrete Naming Certificate for EffectiveCauchySequenceSpace$^{\uparrow}$

Following Bishop and Bridges, the EffectiveCauchySequenceSpace$^{\uparrow}$ packet records the finite effective Cauchy-sequence surface used by Real and Completion consumers before any quotient of Cauchy names is formed. It reads the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the comparison boundary of chapter 1820, and the terminal real-name seal of chapter 38. The packet is a displayed sequence-space carrier with a replayable effective route, not a selected representative, not a host sequence type, and not ambient completeness of Real$^{\uparrow}$.

Definition 3115.1 (EffectiveCauchySequenceSpace carrier)**.** *An* EffectiveCauchySequenceSpace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, Q, A, T, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* StreamName$^{\uparrow}$ *window family through which sequence observations are requested.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row exposing regular rational values on those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger naming the dyadic request, halving data, and comparison cells.*
4. *Q is the* CauchyEquivalenceSetoid$^{\uparrow}$ *comparison row. It records equality-facing comparison only after the windows, regular readback, and dyadic tolerance rows have been displayed.*
5. *A is the terminal* Real$^{\uparrow}$ *seal row reached by the effective Cauchy route.*
6. *T records transport from the displayed stream windows and regular rational rows to the comparison and real-seal rows.*
7. *K records replay of the same route for downstream completion consumers.*
8. *H records componentwise* hsame *transport for S, R, D, Q, A, T, K and the local naming row.*
9. *C records displayed* Cont *replay from finite tolerance requests through $S, R, D, Q, A, T, K, H, P, N$.*
10. *P is the* Pkg *provenance over* EffectiveCauchySequenceSpace$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* CauchyEquivalenceSetoid$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, R, D, Q, A, T, K, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceSpaceUp.EffectiveCauchySequenceSpaceTasteGate`
`single carrier alignment`

Theorem 3115.2 (EffectiveCauchySequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier*

$$E = (S, R, D, Q, A, T, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, finite StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyEquivalenceSetoid comparison routing, Real seal admission, transport, replay, componentwise* hsame *stability, displayed* Cont *continuation,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4095.1. Its public rows are precisely the stream windows, regular rational readback, dyadic tolerance, setoid comparison, real seal, transport, replay, stability, continuation, provenance, and local name.

The comparison row Q can be read only after S, R, D have been displayed. Thus equality-facing consumers first consume finite windows, rational observations, and dyadic tolerance data before they see the setoid boundary.

The terminal seal A is downstream of the same route and is preserved only through T, K, H, C, P, N. Since quotient stream equality, selected representatives, countable choice, host real equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3115.3 (EffectiveCauchySequenceSpace regular readback route)**.** *Every regular readback exported by an accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside $S, R, D, Q, A, T, K, H, C, P, N$. The route exports finite observation and tolerance data for completion-facing consumers and does not export quotient equality of streams, a selected limiting representative, countable choice, or host real equality.

Proof. Start with Definition 4095.2. It forces a consumer to read the StreamName windows S before using the RegSeqRat row R.

The dyadic ledger D supplies the requested tolerance for that readback, and the comparison row Q consumes the same S, R, D data before any equality-facing statement is visible.

Finally the Real seal A is reached only after the comparison route has been displayed. Transport and replay through T, K, H, C, P, N preserve the same row order, so no quotient, selected representative, choice functional, or host equality can be inserted into the readback route. □

Theorem 3115.4 (EffectiveCauchySequenceSpace Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier must read the real seal as the terminal row of $S, R, D, Q, A, T, K, H, C, P, N$. The seal records that the displayed effective Cauchy route may be consumed by Real and Completion chapters; it does not assert that* $\mathsf{Real}^{\uparrow}$ *is ambiently complete, that all Cauchy streams have selected limits, or that Cauchy-equivalent streams are identified by host equality.*

Proof. Project A from the carrier. The definition places A after the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and CauchyEquivalenceSetoid comparison row.

Apply the regular readback route of Definition 4095.3. It binds every Real-facing read to the same finite observation and tolerance route before the terminal seal is consumed.

The remaining rows T, K, H, C, P, N transport, replay, source, and name that route. Because the carrier has no coordinate for ambient completeness, selected limits, quotient equality, or host equality, the Real seal remains a boundary row rather than a completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid comparison row, a terminal RealUp seal, transport, replay, componentwise hsame stability, displayed Cont continuation, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4095.1, Definition 4095.2, Definition 4095.3, and Definition 4095.4 seed the finite effective Cauchy-sequence-space route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient equality of Cauchy streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyEquivalenceSetoid comparison routing, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchySequenceSpaceUp.

Chapter 3116

A Concrete Naming Certificate for $\mathsf{FiniteLowerBoundFold}^{\uparrow}$

Following the finite-cover proof of Heine–Cantor in chapter 9237, the $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ packet records the rational lower-bound fold that turns finitely many local continuity radii into one uniform precision row. It sits over compact finite coverage from chapter 60, the metric source of chapter 57, pointwise continuity rows from chapter 59, rational positivity from chapter 37, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet makes the finite $\mathsf{ProbeBundle}$ fold an explicit NameCert object.

Definition 3116.1 (FiniteLowerBoundFold carrier)**.** *A* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row carrying the finite coverage witness.*
2. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine indexing the bundled centers.*
3. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound candidates, one row for each bundled probe.*
4. *A is the accumulator ledger produced by primitive recursion over $\mathcal{P}$.*
5. *B is the certified lower-bound row saying that the accumulator is below every displayed candidate in R.*
6. *L is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the accumulator as the tolerance-only precision.*
7. *H records componentwise* hsame *transport for the compact source, bundle spine, rational candidates, accumulator, lower-bound certificate, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the fold over the finite bundle.*
9. *Q is the* Pkg *provenance over* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$ *row.*

The carrier contains no host minimum, selected center outside $\mathcal{P}$, open-cover compactness object, quotient point equality, external modulus function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldTasteGate single carrier alignment`

Theorem 3116.2 (FiniteLowerBoundFold NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *carrier*

$$F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$ *surface consists exactly of compact source exposure,* $\mathsf{ProbeBundle}$ *spine admission, positive* $\mathsf{Rat}^{\uparrow}$ *candidate admission, accumulator recursion, certified lower-bound readback,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local naming. The fold route is confined to*

$$\mathsf{Compact}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Continuous}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

Proof. Project the carrier in Definition 4096.1. The only rows visible to a consumer are the compact source, finite bundle, rational candidate family, accumulator, lower-bound certificate, handoff, and structural rows.

The bundle spine $\mathcal{P}$ supplies the induction domain. At each finite step the fold reads one positive rational candidate from R and updates the accumulator row A through the displayed replay route C.

The lower-bound certificate B is attached to the same finite recursion. It states that the final accumulator is below every candidate in R, so a downstream read of L receives a classifier-stable precision row rather than a host minimum.

Transport, provenance, and local naming are supplied by H, Q, N. Since none of the excluded host, open-cover, quotient, external-modulus, or completeness objects occurs in the carrier, the listed obligations exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldCarrier namecert obligations`

Theorem 3116.3 (FiniteLowerBoundFold uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *packet obtains its* $\mathsf{UniformModulus}^{\uparrow}$ *precision row only through the finite* $\mathsf{ProbeBundle}$ *spine* $\mathcal{P}$*, the positive* $\mathsf{Rat}^{\uparrow}$ *candidate ledger* R*, the accumulator row* A*, the lower-bound certificate* B*, and the structural rows* H, C, Q, N*. The consumer receives no host finite minimum, selected center outside the bundle, open-cover compactness object, quotient point equality, external modulus selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldCarrier uniform modulus handoff`

Proof. Start with an accepted carrier $F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N)$. The compact source row K and the finite bundle $\mathcal{P}$ are the only coverage rows available.

Read the candidate family R. Each candidate is a positive $\mathsf{Rat}^{\uparrow}$ row attached to a bundled probe, so the fold cannot inspect an infinite cover or a center outside the displayed bundle.

The accumulator row A and certificate row B provide the lower-bound fold. They are transported and replayed by H and C, and then sourced and named by Q and N.

The handoff row L is therefore a $\mathsf{UniformModulus}^{\uparrow}$ precision read over that finite packet. Since no excluded host or compactness coordinate is present, every uniform-modulus consumer remains inside the displayed fold route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ packet is generated from a CompactUp finite coverage row, a nonempty ProbeBundle spine, positive RatUp candidate rows, an accumulator ledger, a certified lower-bound row, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteLowerBoundFold}\uparrow}$, seed)

Scope closed: Definition 4096.1, Definition 4096.2, and Definition 4096.3 seed the finite ProbeBundle rational lower-bound fold used by compact-uniform-modulus consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteLowerBoundFold}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host finite minima, selected centers outside the bundle, open-cover compactness, quotient point equality, external modulus selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bundle admission, rational candidate positivity, accumulator recursion, lower-bound certificate stability, uniform-modulus handoff, and ledger non-escape for BEDC.Derived.FiniteLowerBoundFoldUp.

DNA hash: `aggregate=sha256:382cef63d1023f0ab10fc718fdae684b2ab03e53b1409d98cb6368e9134a7148;statement=sha256:cf`

Chapter 3117

A Concrete Naming Certificate for MetaCICDecidableConversionFrontier$^{\uparrow}$

The MetaCICDecidableConversionFrontier$^{\uparrow}$ packet realizes the bounded conversion boundary in chapter 9379 as finite NameCert data. It sits beside the decidability boundary of chapter 749, the normalization frontier of chapter 892, the bounded normal-conversion bridge of chapter 3439, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet names the exact point where bounded normal equality may be consumed by a conversion-facing checker while the unrestricted closed-term conversion theorem remains outside the carrier.

Falsifiable prediction: Every accepted MetaCICDecidableConversionFrontier$^{\uparrow}$ read exposes metacicdecidableboundary, metacicnormalizationfrontier, bounded-normal-conversion-bridge, normal-form-consistency-seal, metaciccriticalpath, bounded-checker, finished-normal, obstruction, transport, replay, provenance, and local NameCert rows before any conversion-frontier handoff is consumed.

Independence witness: metacicdecidableboundary, metacicnormalizationfrontier, bounded-normal-conversion-bridge, normal-form-consistency-seal, metaciccriticalpath: the decidability boundary records checker limits without the normal endpoint row; the normalization frontier records candidate and endpoint evidence without the conversion checker; the bridge records bounded normal comparison without the full audit map; the consistency seal records closed normal-form safety without conversion completeness; and the critical path records dependency pressure without this finite conversion-frontier carrier. None is bijective to MetaCICDecidableConversionFrontier$^{\uparrow}$.

Definition 3117.1 (MetaCIC decidable conversion frontier carrier)**.** *A* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, N, B, S, K, Q, O, H, C, P, L)$$

with the following displayed rows:

1. *D is the* MetaCICDecidableBoundary$^{\uparrow}$ *row, recording the typing-checker and same-term decision surface available to the audit map.*
2. *N is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording candidate evidence, finished-normal evidence, endpoint use, and retained obstruction data.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, comparing bounded normalizer outputs under the displayed finished-normal boundary.*

4. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, recording the closed normal-form consistency seal consumed by conversion-facing checks.*
5. *K is the* MetaCICCriticalPath$^{\uparrow}$ *row, retaining the dependency-pressure ledger for conversion, normalization, confluence, and subject-reduction consumers.*
6. *Q is the bounded equality-checker row, recording the compared normal forms, finite fuel budget, and Boolean equality output.*
7. *O is the obstruction row, naming the excluded endpoints: unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, and kernel consistency.*
8. *H records componentwise* hsame *transport for D, N, B, S, K, Q, O and the local naming row.*
9. *C records displayed* Cont *replay from a conversion query through $D, N, B, S, K, Q, O, H, P, L$.*
10. *P is the* Pkg *provenance over* MetaCICDecidableConversionFrontier$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{MetaCICDecidableConversionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, N, B, S, K, Q, O, H, C, P, L$. It has no coordinate for unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICDecidableConversionFrontierUp.MetaCICDecidableConversionFrontierTasteGate`
`single carrier alignment`

Theorem 3117.2 (MetaCIC decidable conversion frontier NameCert obligations)**.** *For every accepted* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier*

$$F = (D, N, B, S, K, Q, O, H, C, P, L),$$

the local NameCert$_{\text{MetaCICDecidableConversionFrontier}^{\uparrow}}$ *surface consists exactly of decidability-boundary exposure through D, normalization-frontier exposure through N, bounded normal-conversion bridge exposure through B, normal-form seal exposure through S, critical-path exposure through K, bounded equality-checker exposure through Q, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 4097.1. Rows D, N, B, S, K are the sibling audit and bridge surfaces, while Q is the local bounded checker row and O keeps the excluded endpoints visible.

The conversion-facing read must pass through the bounded bridge and the normal-form seal together. The bridge supplies only bounded normal comparison, and the seal supplies the closed normal-form safety row; neither row contains arbitrary-fuel completeness or unrestricted beta-conversion.

Finally H, C, P, L transport, replay, source, and name the same packet. Since no carrier coordinate contains evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 3117.3 (MetaCIC decidable conversion frontier bounded route)**.** *Every conversion-facing consumer of an accepted* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier factors through*

MetaCICDecidableBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, NormalFormConsi

and then through the local bounded equality-checker row Q with obstruction retention O. The route exports a bounded conversion frontier only; it does not prove unrestricted closed-term beta-conversion or arbitrary-fuel completeness.

Proof. Start with Definition 4097.2. It forces the consumer to read the decidability boundary, normalization frontier, bounded conversion bridge, consistency seal, and critical-path rows before using the local checker row.

The local checker row Q is interpreted under the obstruction row O. Thus a positive bounded comparison can be replayed as finite NameCert data, but the retained obstruction prevents it from being cited as an unrestricted conversion theorem.

Transport and replay through H and C, with provenance and naming through P and L, preserve that boundary. No sibling route adds arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency to the carrier. □

Theorem 3117.4 (MetaCIC decidable conversion frontier bounded search)**.** *Every positive conversion-frontier read exported by an accepted* $\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}$ *carrier factors through the displayed finite fuel budget in Q, the bounded normalizer surface in N, the* $\mathsf{Term}.eq$ *comparison encoded by the bounded equality-checker row, the* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *row B, and the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *row S. The read is therefore a finite search certificate, not an unrestricted conversion theorem.*

Proof. Project the carrier of Definition 4097.1. The only local row that can report a positive comparison is Q, and that row stores the compared normal forms, finite fuel budget, and Boolean equality output.

Apply the NameCert obligations of Definition 4097.2. They force every conversion-facing read to expose the decidability boundary, normalization frontier, bounded bridge, normal-form seal, critical-path row, checker row, and obstruction row together.

Use the bounded route theorem Definition 4097.3. It places the positive read behind D, N, B, S, K and then behind Q, so the equality output is interpreted only through bounded normalization, $\mathsf{Term}.eq$ comparison, the bridge row, and the consistency seal.

The obstruction row O remains part of the same packet. Since unrestricted closed beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency are not carrier coordinates, a positive bounded read cannot be promoted to any of them. □

Theorem 3117.5 (MetaCIC decidable conversion frontier refusal)**.** *The same* $\mathsf{MetaCICDecidableConversionFrontier}$ *carrier refuses unrestricted closed beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency. Any consumer claiming one of those endpoints must supply data outside the displayed rows $D, N, B, S, K, Q, O, H, C, P, L$.*

Proof. Read the obstruction row O from Definition 4097.1. It explicitly names unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, and kernel consistency as excluded endpoints, while the classifier also omits subject reduction and confluence.

Apply Definition 4097.4. A positive conversion-frontier read is only a bounded search certificate passing through finite fuel, bounded normalization, checker comparison, bridge, and seal rows.

Now use Definition 4097.3. The route preserves obstruction retention through O and structural replay through H, C, P, L, so no later consumer can erase the finite-search boundary inside this packet.

Thus the refused endpoints require additional carriers or theorems not present here. The local NameCert surface exports a bounded decidability frontier only. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}$ packet is generated from MetaCICDecidableBoundary, MetaCICNormalizationFrontier, BoundedNormalConversionBridge, NormalFormConsistencySeal, MetaCICCriticalPath, bounded equality-checker, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4097.1, Definition 4097.2, and Definition 4097.3 seed the finite bounded-conversion frontier for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-checker exposure, normalization-frontier coverage, bounded bridge exactness, normal-form seal compatibility, critical-path retention, obstruction nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDecidableConversionFrontierUp.

Chapter 3118

A Concrete Naming Certificate for MetaCICClosednessDecisionBoundary$^{\uparrow}$

The MetaCICClosednessDecisionBoundary$^{\uparrow}$ packet realizes the closedness-decision warning of chapter 9379 as finite NameCert data. It sits beside the decidability audit packet of chapter 746, the bounded normal-equality checker of chapter 752, the bounded fuel trace of chapter 3889, and the closed type-checker seal of chapter 2182. The packet names the boundary where closedness evidence can be consumed without exporting a theorem of the form `Decidable (ClosedAt n t)` or unrestricted closed-term beta-conversion decidability.

Falsifiable prediction: A consumer of MetaCICClosednessDecisionBoundary$^{\uparrow}$ cannot derive unrestricted closed-term beta-conversion decidability from this packet alone; every accepted read exposes decidability-audit, bounded-normal-equality, fuel-trace, closed-type-checker, closedness-boundary, refusal, transport, replay, provenance, and local NameCert rows before any conversion-facing handoff.

Independence witness: metacicdecidableauditpacket, boundednormalequalitychecker, metacicfueltrace, closedtypecheckerseal: the audit packet records checker scope without this closedness boundary row; the bounded normal-equality checker records finite fuel comparison without a global closedness decision; the fuel trace records bounded normalizer fuel without unrestricted conversion completeness; and the closed type-checker seal records checked typing without a standalone closedness decider. None is bijective to MetaCICClosednessDecisionBoundary$^{\uparrow}$.

Definition 3118.1 (MetaCIC closedness decision boundary carrier)**.** *A* MetaCICClosednessDecisionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, Q, F, S, Z, O, H, C, P, N)$$

with the following displayed rows:

1. *A is the* MetaCICDecidableAuditPacket$^{\uparrow}$ *row exposing the checked typing and bounded conversion audit surface.*
2. *Q is the* BoundedNormalEqualityChecker$^{\uparrow}$ *row exposing bounded normal-form comparison.*
3. *F is the* MetaCICFuelTrace$^{\uparrow}$ *row exposing the finite fuel budget and bounded normalizer endpoint data.*
4. *S is the* ClosedTypeCheckerSeal$^{\uparrow}$ *row exposing closed typing-checker reads.*

5. *Z is the closedness-boundary row naming the available finite closedness evidence and the absence of a global* `Decidable (ClosedAt n t)` *export.*
6. *O is the refusal row for unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency.*
7. *H records componentwise* hsame *transport for A, Q, F, S, Z, O and the local naming row.*
8. *C records displayed* Cont *replay from a closedness-facing query through $A, Q, F, S, Z, O, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$, $\mathsf{MetaCICDecidableAuditPacket}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, Q, F, S, Z, O, H, C, P, N$. It has no coordinate for unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICClosednessDecisionBoundaryUp.MetaCICClosednessDecisionBoundaryTasteGate`
`single carrier alignment`

Theorem 3118.2 (MetaCIC closedness decision boundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ *carrier*

$$B = (A, Q, F, S, Z, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$ *surface consists exactly of decidability-audit exposure through A, bounded normal-equality exposure through Q, finite fuel-trace exposure through F, closed type-checker seal exposure through S, closedness-boundary exposure through Z, refusal retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4098.1. Rows A, Q, F, S are the sibling checker and bounded-computation surfaces, while Z records the closedness boundary and O records the refused endpoints.

A closedness-facing read must pass through the audit packet, bounded checker, fuel trace, and closed type-checker seal before it can use Z. These rows provide finite checker evidence, but none contains a global `Decidable (ClosedAt n t)` theorem.

Finally H, C, P, N transport, replay, source, and name the same boundary packet. Since no carrier coordinate contains unrestricted beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 3118.3 (MetaCIC closedness decision boundary non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ *carrier can cite only the finite route*

$$\mathsf{MetaCICDecidableAuditPacket}^{\uparrow},\ \mathsf{BoundedNormalEqualityChecker}^{\uparrow},\ \mathsf{MetaCICFuelTrace}^{\uparrow},\ \mathsf{ClosedTypeCheckerSeal}^{\uparrow},\ Z,$$

The route preserves checked typing, bounded normal comparison, finite fuel tracing, and closed type-checker sealing while refusing unrestricted closed-term beta-conversion decidability and a standalone global closedness decision.

Proof. Apply Definition 4098.2. It forces every consumer to read the audit, bounded-checker, fuel-trace, and closed-type-checker rows before reading the local closedness-boundary row.

The refusal row O is part of the same carrier. It records that the packet does not export unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Transport and replay through H and C, with provenance and naming through P and N, preserve that boundary. The consumer therefore receives a finite closedness-decision boundary packet, not a global decision theorem. □

The Lean follow-up supplies Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/MetaCICClosedness` by exhibiting distinct accepted and refused boundary packets and by matching on the rows $A, Q, F, S, Z, O, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ packet is generated from MetaCICDecidableAuditPacket, BoundedNormalEqualityChecker, MetaCICFuelTrace, ClosedTypeCheckerSeal, closedness-boundary, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4098.1, Definition 4098.2, and Definition 4098.3 seed the finite closedness-decision boundary for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted closed-term beta-conversion decidability, a global `Decidable (ClosedAt n t)` theorem, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, audit-packet exposure, bounded-normal checker exposure, fuel-trace exposure, closed-type-checker seal compatibility, closedness-boundary nonescape, refusal retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICClosednessDecisionBoundaryUp.

Chapter 3119

A Concrete Naming Certificate for $\mathsf{EffectiveCauchySpace}^{\uparrow}$

Following Bishop–Bridges constructive analysis and represented-space computable analysis, the $\mathsf{EffectiveCauchySpace}^{\uparrow}$ packet records a represented metric source whose Cauchy access is mediated by displayed moduli and finite observation windows. It sits after the represented-space source of chapter 1494, the metric carrier of chapter 57, the effective modulus surface of chapter 3518, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the unit handoff of chapter 1527. The chapter gives Real and Completion consumers a finite route from represented names and effective Cauchy moduli to the current L10 completion boundary; it is not quotient completion equality, a selected limit, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 3119.1 (Effective Cauchy space carrier)**.** *An* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row, naming the represented input surface and the representation-mediated observations available to the packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row, recording the point classifier, distance requests, and metric comparison data read over the same represented source.*
3. *E is the* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *row, supplying the finite stage schedule by which metric tolerance requests become Cauchy-window requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, displaying the finite observation windows opened by the represented source and the effective modulus.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows after the metric and modulus rows have fixed the tolerance request.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack, comparison budgets, and tolerance conversions used by E, S, R.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the represented source, metric row, effective modulus, stream schedule, regular readback, and dyadic ledger have been displayed.*

8. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row admitting the same represented Cauchy data into the completion-facing unit route.*
9. *H records componentwise* hsame *transport for A, M, E, S, R, D, L, U and the local naming row.*
10. *C records displayed* Cont *replay from a Real or Completion request through the represented source, metric row, modulus, finite windows, readback, tolerance ledger, seal, unit handoff, transport, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchySpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{EffectiveCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, E, S, R, D, L, U, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, trichotomy, host equality of completed objects, ambient completeness of $\mathsf{Real}^{\uparrow}$*, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchySpaceUp.EffectiveCauchySpaceTasteGate single carrier alignment`

Theorem 3119.2 (Effective Cauchy space NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *represented-source admission through A and metric-source admission through M over the same displayed carrier;*
2. *effective-modulus exposure through E before any finite stream window is opened;*
3. *StreamName schedule exactness through S, requiring every inspected name window to be one of the displayed finite windows;*
4. *regular rational readback through R only after the represented source, metric row, modulus, stream schedule, and dyadic ledger have been displayed;*
5. *dyadic tolerance accounting through D for every radius, slack, comparison budget, and tolerance conversion consumed by E, S, R;*
6. *terminal Real sealing through L and completion-unit handoff through U, both downstream of A, M, E, S, R, D;*
7. *transport, replay, provenance, and local naming through H, C, P, N without adding a second classifier or hidden completion coordinate.*

No effective-Cauchy-space obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4099.1. The represented source row A and the metric row M are the only source coordinates, so every downstream read is tied to the same represented metric carrier.

The effective modulus row E mediates every tolerance request before the finite stream schedule S is opened. The dyadic ledger D supplies the radius and slack accounting needed by the modulus and schedule rows, and the regular rational readback R is downstream of those displayed data.

The terminal routes L and U are read only after A, M, E, S, R, D have been exposed. Thus the Real seal and the CauchyCompletionUnit handoff consume the same finite represented-Cauchy route rather than a quotient completion, selected limit, or host equality datum.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no row for countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3119.3 (Effective Cauchy space completion handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N)$$

factors through the displayed route

$$\mathsf{RepresentedSpace}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{EffectiveCauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyCo}$$

with transport, replay, provenance, and local naming supplied only by H, C, P, N*. The consumer receives no quotient completion equality, selected limit, countable-choice selector, host equality, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Fix a Real-facing or completion-facing consumer of X. The carrier exposes the represented source, metric row, effective modulus, stream schedule, dyadic ledger, regular readback, Real seal, and completion-unit handoff as the only analytic coordinates.

The consumer first reads A and M, then obtains the tolerance stage from E. The corresponding finite windows are displayed in S, with all radius and slack conversions accounted for in D.

Only after that finite observation route is fixed can the readback row R be consumed. The Real seal L and the CauchyCompletionUnit handoff U are downstream of the same readback and ledger, so both terminal routes inherit the represented Cauchy data rather than selecting an external limit.

The structural rows H, C, P, N preserve the route and introduce no additional analytic coordinate. Therefore every allowed consumer factors through the displayed completion handoff and cannot extract any of the excluded quotient, choice, equality, trichotomy, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveCauchySpace}^{\uparrow}$ Real or Completion read exposes the RepresentedSpace source, MetricSpace row, EffectiveCauchyModulus row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, CauchyCompletionUnit handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the terminal handoff is consumed.

Independence witness: representedspace, metric, effective-cauchy-modulus, streamname, regseqrat, dyadicratcore, real, cauchycompletionunit: $\mathsf{EffectiveCauchySpace}^{\uparrow}$ is not bijective to any listed sibling because it binds a represented metric source, effective Cauchy modulus, finite schedule, tolerance ledger, regular readback, Real seal, completion-unit handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4099.1, Definition 4099.2, and Definition 4099.3 seed the finite represented-metric Cauchy route from effective modulus data to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, represented-source and metric-source agreement, effective-modulus exposure, finite-window exactness, dyadic tolerance accounting, regular readback, Real seal boundary, completion-unit handoff, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchySpaceUp.

Constructive story: An EffectiveCauchySpace$^{\uparrow}$ packet is generated from a represented-space source, a metric row, an effective Cauchy modulus, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3120

A Concrete Naming Certificate for $\mathsf{RegularCauchyRegularization}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyRegularization}^{\uparrow}$ packet records the finite BEDC route by which arbitrary Cauchy data are tightened into the regular-rational route used by $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits between chapter 291, chapter 126, chapter 301, chapter 38, and the Cauchy-criterion surfaces such as chapter 392. The packet is a regularization service over displayed finite rows; it is not quotient stream equality, a selected limit, trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 3120.1 (Regular Cauchy regularization carrier)**.** *A* $\mathsf{RegularCauchyRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, D, S, A, E, H, C, P, N)$$

with the following rows:

1. *Q is the raw Cauchy source row, naming the displayed rational data and its Cauchy-window evidence through a local Cauchy-criterion certificate;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tightening ledger, recording the dyadic precision scale used to replace arbitrary Cauchy requests by nested regular tolerances;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that lists the finite read windows consumed by the regularization;*
4. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the tightened schedule and dyadic ledger;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after the regular readback row has been displayed;*
6. *H records componentwise* hsame *transport for the source, dyadic ledger, schedule, regular readback, terminal seal, replay, provenance, and naming rows;*
7. *C records the displayed* Cont *replay from Q through D, S, A to E;*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting Cauchy streams, selecting a host limit, comparing host real values by trichotomy, or importing completed-real closure.

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularizationUp.RegularCauchyRegularizationTasteGate single carrier alignment`

Theorem 3120.2 (Regular Cauchy NameCert obligations)**.** *For the carrier of Definition 4100.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (Q, D, S, A, E, H, C, P, N)$*, dyadic tightening from the raw Cauchy source row to the scheduled regular windows,* $\mathsf{StreamName}^{\uparrow}$ *schedule stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exactness,* $\mathsf{Real}^{\uparrow}$ *terminal-seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and nonescape for quotient stream equality, selected host limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite packet. The rows $Q, D, S, A, E, H, C, P, N$ are the whole carrier, so the admitted regularization service must expose the raw Cauchy source, the dyadic tightening ledger, the stream schedule, the regular rational readback, and the terminal seal as displayed data.

The Cauchy source is consumed through the local Cauchy-criterion row, while D chooses the dyadic precision budget used for each scheduled request. The schedule row S then records only finitely many read windows, and A packages those windows as the regular rational readback used by $\mathsf{RegSeqRat}^{\uparrow}$.

Transport and replay are componentwise. The hsame row H preserves the source, ledger, schedule, readback, seal, route, provenance, and local name together, while the Cont row C forces every consumer to traverse Q then D, S, A before reaching E.

Since the packet contains no quotient-stream coordinate, selected host limit, trichotomy row, choice row, or ambient completeness row, no such datum can be projected from the local NameCert surface. □

Theorem 3120.3 (Regular Cauchy modulus route)**.** *Every accepted* $\mathsf{RegularCauchyRegularization}^{\uparrow}$ *carrier turns a raw Cauchy source row into a* $\mathsf{RegSeqRat}^{\uparrow}$ *route by first tightening the source through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then scheduling the required windows through* $\mathsf{StreamName}^{\uparrow}$*, then reading the scheduled rational data as the regular row* A*. The* $\mathsf{Real}^{\uparrow}$ *seal* E *is available only through that displayed route, and no consumer may replace the route by quotient stream equality, a selected limit, trichotomy, countable choice, or ambient completeness.*

Proof. Begin with the carrier inventory of Definition 4100.1. The only source of tolerance control is D, and the only source of read order is S. Thus the raw Cauchy row cannot be consumed as a regular sequence until the dyadic ledger and the stream schedule have both been displayed.

Next read the regular row. The packet places A after D and S, so the $\mathsf{RegSeqRat}^{\uparrow}$ consumer sees regular rational data only through the tightened windows. The terminal $\mathsf{Real}^{\uparrow}$ seal E is reached after this readback, not directly from the raw source row.

Finally apply the nonescape clauses from the NameCert obligations. Because no quotient stream, selected host limit, trichotomy, countable-choice coordinate, or ambient completeness row occurs in R, the modulus route is the displayed finite route through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4100.1, Definition 4100.2, and Definition 4100.3 seed the finite regularization surface from raw Cauchy source rows through dyadic tightening, StreamName scheduling, RegSeqRat readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tightening exactness, schedule stability, regular readback exactness, Real seal exposure, and nonescape for BEDC.Derived.RegularCauchyRegularizationUp.

Constructive story: A RegularCauchyRegularization$^{\uparrow}$ packet is generated from a raw Cauchy source row, a DyadicRatCore$^{\uparrow}$ tightening ledger, a StreamName$^{\uparrow}$ schedule, a RegSeqRat$^{\uparrow}$ readback row, a Real$^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3121

A Concrete Naming Certificate for $\mathsf{ReflexiveBanach}^{\uparrow}$

Following Banach and Alaoglu, the $\mathsf{ReflexiveBanach}^{\uparrow}$ packet records the finite double-dual evaluation window that functional-analysis consumers may inspect before any weak compactness conclusion is read. It sits beside the finite weak-star dual-ball route of chapter 3420, the weak observation surface of chapter 3597, the norm ledger of chapter 89, the Hilbert comparison surface of chapter 91, and the functional-analysis carrier of chapter 94. The packet names only displayed Banach, norm, dual-window, double-dual evaluation, weak-topology, bounded-ball, transport, replay, provenance, and local naming rows; it does not assert full Eberlein–Smulian, select weak cluster points, import quotient dual equality, or use choice over all bounded nets.

Definition 3121.1 (ReflexiveBanach finite double-dual carrier)**.** *A* $\mathsf{ReflexiveBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, N, D, E, W, A, H, C, P, L)$$

with the following displayed rows:

1. *B is the Banach carrier row, read through the displayed* $\mathsf{Banach}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *surfaces.*
2. *N is the norm row, recording the finite norm estimates used by source, dual, and ball consumers through* $\mathsf{Norm}^{\uparrow}$*.*
3. *D is the dual-window row, listing the finitely many bounded-linear-functional observations allowed in the current packet.*
4. *E is the double-dual evaluation row, recording the canonical evaluation of displayed source endpoints against the functionals named by D.*
5. *W is the weak-topology readback row, sending those finite scalar evaluations to the* $\mathsf{WeakTopology}^{\uparrow}$ *observation surface.*
6. *A is the bounded-ball ledger, recording the finite norm-bound rows whose weak-compactness handoff is visible to later consumers.*
7. *H records componentwise* hsame *transport for the Banach carrier, norm row, dual windows, double-dual evaluations, weak readbacks, bounded-ball ledgers, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from Banach admission through norm control, dual-window evaluation, weak readback, and bounded-ball handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ReflexiveBanach}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed host double dual, quotient equality of dual functionals, a selected weak cluster point, a countable-choice family of balls, or an unrestricted compactness theorem.

Theorem 3121.2 (ReflexiveBanach NameCert obligations)**.** *For every accepted* $\mathsf{ReflexiveBanach}^{\uparrow}$ *carrier* $R = (B, N, D, E, W, A, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}$ *surface consists exactly of carrier admission for B, double-dual evaluation stability for E, weak topology compatibility for W, bounded-ball readback for A, and no-choice ledger nonescape for the structural rows H, C, P, L. Every accepted read must expose the Banach carrier, norm row, finite dual window, evaluation row, weak readback row, and bounded-ball ledger before local naming is consumed.*

Lean4 ✓ `BEDC.Derived.ReflexiveBanachUp.ReflexiveBanachTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4101.1. The rows $B, N, D, E, W, A, H, C, P, L$ are the complete visible inventory, so carrier admission and local naming can only use those coordinates.

The double-dual evaluation obligation is finite. Each evaluation in E is tied to a source endpoint from B and a bounded functional from D, with norm control read through N; componentwise transport through H preserves the same displayed evaluation row.

Weak topology compatibility follows by reading W through the finite scalar observation surface of chapter 3597. The bounded-ball ledger A is downstream of the same norm and evaluation rows, and the Banach–Alaoglu sibling supplies only the finite dual-ball window cited in the provenance row.

Finally C, P, L replay, source, and name that same packet. Since no row stores a selected cluster point, quotient dual equality, host double-dual equality, or a choice family over all bounded balls, the no-choice ledger cannot escape the displayed finite route. □

Theorem 3121.3 (ReflexiveBanach weak compactness handoff)**.** *Every weak-compactness-facing consumer of an accepted* $\mathsf{ReflexiveBanach}^{\uparrow}$ *packet must factor through the finite route*

$$B, N, D, E, W, A \longmapsto H, C, P, L.$$

The consumer may cite $\mathsf{BanachAlaoglu}^{\uparrow}$ *only through the finite dual-window provenance, may cite* $\mathsf{WeakTopology}^{\uparrow}$ *only through the scalar readback row W, and may cite* $\mathsf{Hilbert}^{\uparrow}$ *or* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *only through the displayed source and norm rows. It receives no selected weakly convergent subnet, no quotient dual-space equality, no host compactness theorem, and no choice-based traversal of the bounded ball.*

Proof. Begin with the bounded-ball ledger. The row A is read after the Banach carrier B and norm row N, so every ball-facing request is already confined to displayed norm estimates over the accepted carrier.

Next expose the dual-window and evaluation rows. The row D lists only finitely many bounded-functional observations, and E evaluates displayed source endpoints against those observations before the weak-topology row W can be consumed.

The weak readback is therefore a finite scalar route. By chapter 3597 and chapter 3420, downstream compactness vocabulary can only repackage the finite weak and weak-star windows already present in D, E, W, A.

Transport, replay, provenance, and local naming preserve this route through H, C, P, L. The absence of selector, quotient, host compactness, and choice-family coordinates prevents the handoff from becoming a nonconstructive weak compactness principle. □

Falsifiable prediction: Every accepted $\mathsf{ReflexiveBanach}^{\uparrow}$ packet exposes a Banach carrier, norm row, finite dual-window row, double-dual evaluation row, weak-topology readback row, bounded-ball ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any weak compactness consumer can read the packet.

Independence witness: banach-alaoglu, weak-topology, norm, hilbert, functionalanalysis: $\mathsf{ReflexiveBanach}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries double-dual evaluation rows and bounded-ball weak-readback ledgers over a Banach source while refusing selected weak cluster points and quotient dual equality.

Closure status for $\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{ReflexiveBanach}^{\uparrow}$ packet is generated from a Banach carrier, norm row, finite dual-window row, double-dual evaluation row, weak-topology readback row, bounded-ball ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4101.1, Definition 4101.2, and Definition 4101.3 seed the finite double-dual and bounded-ball weak-readback certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Eberlein–Smulian, selected weak cluster points, quotient dual equality, host double-dual equality, full weak compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, double-dual evaluation stability, weak topology compatibility, bounded-ball readback, no-choice ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ReflexiveBanachUp.

Chapter 3122

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ packet records a regular Cauchy filter space as finite NameCert data for the L10 completion route. It sits between the Cauchy filter carrier of chapter 333, the regular Cauchy filter rows of chapter 1445, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real seal of chapter 38. The packet feeds completion consumers by displaying regular filter-space rows before any quotient of filters, selected limit, or countable-choice principle can be invoked.

Definition 3122.1 (Regular Cauchy filter space carrier)**.** *A* $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, B, W, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the admitted point and basic-filter carrier row, retaining classifier-distinct point and filter names without quotienting them by convergence or equality;*
2. *B is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, exposing the finite filter-membership and directedness observations consumed by the packet;*
3. *W is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *regularity window, recording the shrinking, containment, and Cauchy-modulus rows used to read the filter as regular;*
4. *Q is the classifier/window exactness row, tying every accepted window request to the same point and filter classifier rows in X, B, W;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row for finite request windows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for regular window moduli and Cauchy bounds;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *sealing row reached only after S, R, W have displayed the finite regular-filter readback;*
8. *H records componentwise* hsame *transport for all carrier, basis, regularity, exactness, window, readback, seal, replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay through the same finite route;*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, a selected representative filter, a selected limit point, countable choice, ambient separated completion, or host equality of point names.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterSpaceUp.RegularCauchyFilterSpaceTasteGate single carrier alignment`

Theorem 3122.2 (Regular Cauchy filter space NameCert obligations)**.** *For every accepted* RegularCauchyFilterSpac *carrier*

$$F = (X, B, W, Q, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the point/filter carrier X, Cauchy filter basis B, regularity window W, classifier/window exactness row Q, stream windows S, regular rational readback R, real seal E, transport H, replay C, provenance P, and local naming N;*
2. *classifier/window exactness, requiring every filter-space read to use the displayed classifier rows of X, B, W and the same finite request windows recorded in Q, S;*
3. *stability and transport, requiring regularity, Cauchy bounds, rational readback, and real sealing to move only componentwise through H and to replay through C;*
4. *ledger nonescape, excluding quotient filters, host equality, selected representatives, selected limits, countable choice, and ambient separated completion from the packet;*
5. *completion-consumer handoff, exporting only $X, B, W, Q, S, R, E, H, C, P, N$ to a* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer before that consumer adds its own completion rows.*

No other NameCert obligation is available from the regular Cauchy filter space packet.

Proof. Carrier admission is the projection of Definition 4102.1. The tuple lists the point/filter carrier, Cauchy filter basis, regularity window, exactness row, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Classifier/window exactness is carried by Q and S. A consumer first fixes the point and filter classifiers in X, B, W, then reads the finite request windows through Q, S before asking for rational or real data.

Stability and transport are componentwise. The row H transports the displayed coordinates, and the row C replays the same route for consumers, so regularity and Cauchy bounds cannot be replaced by host equality or by a detached separated-completion fact.

Ledger nonescape follows from the carrier inventory. There is no coordinate for quotient filter equality, selected representatives, selected limits, countable choice, or ambient separated completion.

Finally the completion handoff is only the displayed tuple $X, B, W, Q, S, R, E, H, C, P, N$. A $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer may use that finite regular-filter-space surface as input, but any completion object, quotient, or limit belongs to the consumer packet rather than to this carrier. □

Falsifiable prediction: A consumer that claims a $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ handoff while omitting the Cauchy filter basis, regularity window, classifier/window exactness row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 4102.1.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, regseqrat, real: $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ binds the filter basis, regularity window, finite request windows, rational readback, real sealing,

transport, replay, provenance, and local naming rows in one completion-facing packet; each named sibling lacks at least one of these rows as a local filter-space coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ packet is generated from a point/filter carrier, CauchyFilter basis row, RegularCauchyFilter regularity window, classifier/window exactness row, StreamName finite windows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4102.1 and Definition 4102.2 seed the finite regular-filter-space surface that can feed Cauchy filter completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected representatives, selected limits, countable choice, ambient separated completion, Lean verification, bridge checking, or host equality of point names.

Upgrade path: Obligation closure requires checked carrier admission, classifier/window exactness, componentwise stability, ledger nonescape, and completion-consumer handoff for BEDC.Derived.RegularCauchyFilterSpaceUp.

Chapter 3123

A Concrete Naming Certificate for $\mathsf{CauchyRealLimit}^{\uparrow}$

Following Bishop and Bridges, a Cauchy real limit is treated here as the finite handoff from a regular Cauchy presentation to a sealed real name. The $\mathsf{CauchyRealLimit}^{\uparrow}$ packet sits after the regular rational readback surface of chapter 301, the stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the filter-facing limit route of chapter 1558. Its role is to expose the L10 real-completion limit handoff without importing quotient streams, selected representatives, host limits, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3123.1 (Cauchy-real-limit carrier)**.** *A* $\mathsf{CauchyRealLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the Cauchy-modulus row that assigns each requested tolerance to a finite tail-window obligation.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row displaying the inspected windows of the Cauchy presentation.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational approximants observed through S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the observed windows with the requested modulus budget.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback and dyadic tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for the modulus, stream, regular readback, dyadic tolerance, real seal, and local certificate rows.*
7. *C records displayed* Cont *replay from a tolerance request through M, S, R, D to the* $\mathsf{Real}^{\uparrow}$ *seal E.*
8. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, an externally selected limit, host real convergence, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyRealLimitUp.CauchyRealLimitTasteGate single carrier alignment`

Theorem 3123.2 (Cauchy-real-limit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLimit}^{\uparrow}$ *carrier*

$$L = (M, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Cauchy modulus, stream windows, regular rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*
2. *tail-window coverage, requiring every tolerance request admitted by* M *to name a finite window in* S *before a limit read is exported;*
3. *dyadic tolerance exactness, requiring every comparison used by the handoff to read exactly the row* D *beside the* $\mathsf{RegSeqRat}^{\uparrow}$ *observations in* R*;*
4. *real-seal handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *seal* E *to be consumed only after* M, S, R, D *have been displayed;*
5. *ledger non-escape, excluding quotient streams, selected limits, host convergence, choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Project the carrier coordinates. The rows M, S, R, D, E display the Cauchy modulus, finite stream windows, regular rational observations, dyadic tolerance ledger, and terminal real seal before any limit-facing consumer can read the packet.

The tail-window and tolerance clauses are finite. A tolerance request first consults M, then reads the scheduled window in S, observes the regular rational data in R, and checks the dyadic budget in D.

Finally attach the seal and structural rows. The row E is reachable only after the displayed readback route, while H, C, P, N transport, replay, source, and name that same route. Since no further coordinate is present, the carrier cannot export quotient stream equality, a selected limit, host convergence, choice, or ambient real completeness. □

Falsifiable prediction: A $\mathsf{CauchyRealLimit}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ limit seal without reading the displayed Cauchy modulus, StreamName windows, RegSeqRat observations, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchyfilterlimit: no listed sibling supplies the whole Cauchy-real-limit packet because each sibling contributes only readback, windowing, tolerance, sealing, or filter-facing limit data while this chapter binds the finite handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealLimit}^{\uparrow}$ packet is generated from a Cauchy modulus row, StreamName windows, RegSeqRat regular rational readbacks, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$, seed)

Scope closed: Definition 4103.1 and Definition 4103.2 seed the finite real-limit handoff over Cauchy modulus, stream-window, regular-readback, dyadic-tolerance, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real convergence, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, tail-window coverage, dyadic tolerance exactness, Real seal handoff, and ledger non-escape.

Chapter 3124

A Concrete Naming Certificate for $\mathsf{PolishBaireFixedPoint}^{\uparrow}$

Following Banach's fixed-point theorem and Baire's complete-metric category theorem, the $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ packet records the finite route by which a complete separable metric carrier, a Baire surface, and a contraction-iteration ledger are read together before any endpoint is treated as a host object. It routes through the complete-separable carrier of chapter 1490, the ultrametric Baire window of chapter 3485, the complete-metric Baire surface of chapter 3535, the Picard step route of chapter 395, and the finite iteration packet of chapter 2383. The chapter names the displayed fixed-point route without claiming open-cover compactness, countable choice, a selected host fixed point, or any fixed-point primitive outside the carried rows.

Definition 3124.1 (Polish-Baire fixed-point carrier)**.** *A* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, B, U, C, I, E, H, R, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row exposing the complete separable metric carrier, stream-name schedule, regular rational readback, observation ledger, transport, replay, provenance, and local name.*
2. *B is the* $\mathsf{BaireUltrametric}^{\uparrow}$ *row that records the finite nested-window Baire route consumed by the packet.*
3. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *row joining a dense-open schedule, complete-metric Cauchy thread, metric carrier, finite windows, rational readback, and terminal real seal.*
4. *C is the* $\mathsf{PicardContraction}^{\uparrow}$ *row carrying the one-step contraction route over the displayed metric source.*
5. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *row carrying the finite orbit ledger, residual budget, completion handoff, transport, replay, provenance, and local name for the same route.*
6. *E is the endpoint-candidate seal row. It is readable only after P, B, U, C, I have been displayed, and it records no selected host point or host fixed-point equality.*
7. *H records componentwise* hsame *transport for the Polish-space, Baire-ultrametric, complete-metric Baire, Picard-contraction, iteration, endpoint-seal, replay, provenance, and local naming rows.*

8. *R records displayed* Cont *replay from a contraction request through the Baire windows, complete-metric surface, Picard step, finite iteration ledger, residual budget, and endpoint-candidate seal.*
9. *G is the* Pkg *provenance over* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurfac}$ $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for open-cover compactness, countable-choice selection, quotient completion equality, host fixed-point equality, or a primitive fixed-point theorem.

Theorem 3124.2 (Polish-Baire fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier*

$$F = (P, B, U, C, I, E, H, R, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission, Polish complete-separable source exposure, Baire-ultrametric window exposure, complete-metric Baire-surface routing, Picard one-step readability, finite fixed-point iteration ledger exposure, endpoint-candidate sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside the rows of Definition 4104.1.*

Lean4 ✓ `BEDC.Derived.PolishBaireFixedPointUp.PolishBaireFixedPointTasteGate single carrier alignment`

Proof. Project the accepted carrier. The rows P, B, U, C, I, E are the only analytic rows available to the local certificate: complete separable source, Baire window, complete-metric Baire surface, Picard step, finite iteration ledger, and endpoint-candidate seal.

The source and Baire obligations are read before any endpoint row. The Polish row supplies the common complete-separable metric surface, while the two Baire rows expose only finite nested-window and complete-metric schedule data.

The fixed-point route is then read through C and I. The Picard row supplies the one-step contraction route, and the iteration row supplies the finite orbit ledger and residual budget without selecting a completed fixed point.

Finally H, R, G, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate beyond these rows, the listed surface exhausts the local NameCert obligations. □

Theorem 3124.3 (Polish-Baire fixed-point iteration nonescape)**.** *Every fixed-point-facing read from an accepted* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier must pass through*

$\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, BHist, hsa

In particular, the endpoint-candidate row E is consumable only after the complete-separable source, Baire windows, complete-metric Baire surface, Picard step, finite orbit ledger, transport, replay, provenance, and local naming rows have been displayed.

Proof. Start from the carrier definition. The endpoint-candidate seal E is stored after the Polish source P, Baire rows B, U, Picard route C, and finite iteration ledger I.

Apply the NameCert obligation theorem. It makes the complete-separable source, Baire schedule, Picard step, finite orbit ledger, and residual budget part of the public local surface before the endpoint-candidate seal can be read.

The structural rows H, R, G, N then preserve the same route under transport, replay it for downstream consumers, record provenance, and attach the local name. Because no separate coordinate stores open-cover compactness, countable choice, host fixed-point equality, or a fixed-point primitive, any fixed-point-facing read remains inside the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}$.

Constructive story: A $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ packet is generated from a PolishSpace complete-separable carrier, BaireUltrametric finite window route, CompleteMetricBaireSurface schedule, PicardContraction step, FixedPointIteration ledger, endpoint-candidate seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4104.1, Definition 4104.2, and Definition 4104.3 seed the finite Polish/Baire fixed-point route over complete-separable, Baire-window, complete-metric Baire, Picard, iteration, endpoint-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected host fixed points, quotient completion equality, host fixed-point equality, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Baire-window exposure, complete-metric Baire routing, Picard-step readability, finite iteration ledger exactness, endpoint-candidate nonescape, transport stability, replay stability, provenance stability, and local naming for BEDC.Derived.PolishBaireFixedPointUp.

Chapter 3125

A Concrete Naming Certificate for CompactModulusSelector$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor finite-net proof, the CompactModulusSelector packet records the finite selector route that turns compact-metric total boundedness and pointwise continuity data into a reusable uniform-modulus certificate. It sits over the compact metric source of chapter 129, the total-boundedness finite-net interface of chapter 127, the continuous-map row of chapter 130, and the uniform-modulus consumer of chapter 369. Its L10 real-facing reads pass through the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet names finite-net modulus selection as displayed BEDC data; it does not assert open-cover compactness, countable choice, a selected infinite net, host function equality, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 3125.1 (CompactModulusSelector carrier)**.** *A* CompactModulusSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, F, U, D, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* CompactMetric$^{\uparrow}$ *source row. It records the compact metric carrier and its displayed metric observation surface.*
2. *T is the* TotallyBounded$^{\uparrow}$ *finite-net row read from K at the requested dyadic tolerance scale.*
3. *F is the* ContinuousMap$^{\uparrow}$ *graph and pointwise-continuity row whose local moduli are inspected only at the displayed net centers.*
4. *U is the candidate* UniformModulus$^{\uparrow}$ *row exported to downstream uniform-continuity consumers.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger controlling the source mesh, target error budget, and triangle comparison.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window row naming the inspected net-center and point-name windows.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the finite metric and image-distance observations.*
8. *A is the terminal* Real$^{\uparrow}$ *seal row reached only after D, W, R have displayed the requested estimates.*

9. *H records componentwise* hsame *transport for compact metric, total-boundedness, continuous-map, uniform-modulus, dyadic, window, readback, seal, continuation, provenance, and naming rows.*
10. *C records displayed* Cont *replay from a uniform-modulus request through the compact finite-net selection route and then into the exported modulus row.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a choice function over all covers, selected infinite subnet data, host equality of maps, trichotomy, quotient completion, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactModulusSelectorUp.CompactModulusSelectorTasteGate single carrier alignment`

Theorem 3125.2 (CompactModulusSelector NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *carrier* $S = (K, T, F, U, D, W, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, total-bounded finite net, continuous-map row, uniform-modulus export, dyadic tolerance ledger, finite windows, regular readback, Real seal, transport, continuation, provenance, and local naming rows. Any consumer that reads the exported modulus must consume the finite-net row T, the pointwise continuity row F, and the dyadic comparison row D before U is visible. The packet exports no open-cover compactness, countable choice, host map equality, selected infinite net, trichotomy, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4105.1. The listed coordinates are exactly the compact metric, total-boundedness, continuous-map, uniform-modulus, dyadic, window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Replay C. A modulus request first reads the compact metric source K, then the finite-net row T, then the continuous-map row F at the displayed net centers. The dyadic ledger D, finite windows W, and regular readback R provide the requested source and target estimates before the terminal seal A is consumed.

The exported row U is therefore a finite-net modulus row. Transport by H preserves only the displayed coordinates, while P and N package and name the same route, so none of the excluded principles appears as a hidden obligation. □

Theorem 3125.3 (CompactModulusSelector finite-net selection)**.** *Let $S = (K, T, F, U, D, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *packet. Every exported* $\mathsf{UniformModulus}^{\uparrow}$ *read factors through the finite route*

$$K, T, F, D, W, R, A \longmapsto U, H, C, P, N.$$

The finite net is read from $\mathsf{TotallyBounded}^{\uparrow}$ *inside the* $\mathsf{CompactMetric}^{\uparrow}$ *source, the local continuity data are read from* $\mathsf{ContinuousMap}^{\uparrow}$ *at the displayed net centers, and the dyadic tolerance ledger supplies the comparison budget. The route preserves all rows $K, T, F, U, D, W, R, A, H, C, P, N$ and exports no open-cover compactness, countable choice, selected infinite net, host equality of functions, trichotomy, quotient completion, or ambient real completeness.*

Proof. Start from a consumer of the uniform-modulus row. Carrier admission exposes U only as a row of the packet named in Definition 4105.1.

Use the continuation row C to replay the finite selection route. The compact metric source K supplies the ambient metric data, T supplies the finite net, and F supplies the local continuity data at the displayed centers.

The tolerance and readback rows D, W, R, A then certify the comparison budget before U is exported. Since transport, provenance, and naming are confined to H, C, P, N, the handoff preserves the listed rows and cannot export any of the excluded choice, open-cover, equality, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4105.1, Definition 4105.2, and Definition 4105.3 seed the finite-net modulus selector route over compact metric, total-bounded, continuous-map, uniform-modulus, dyadic, window, readback, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected infinite nets, host function equality, trichotomy, quotient completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net selection, pointwise-continuity inspection, dyadic tolerance comparison, uniform-modulus export, and structural nonescape for BEDC.Derived.CompactModulusSelectorUp.

Constructive story: A $\mathsf{CompactModulusSelector}^{\uparrow}$ packet is generated from a CompactMetric row, a TotallyBounded finite-net row, a ContinuousMap pointwise-continuity row, a UniformModulus export row, dyadic tolerance data, finite windows, regular readback, a Real seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 3126

A Concrete Naming Certificate for $\mathsf{TypeCheckingClassifier}^{\uparrow}$

The $\mathsf{TypeCheckingClassifier}^{\uparrow}$ packet realizes the type-checking-as-Ext-membership vision of chapter 9367 as one finite NameCert carrier. It sits beside the classifier-membership packet of chapter 476, the typing-classifier route of chapter 489, the subject-reduction surface of chapter 483, the derivation-trace packet of chapter 598, and the proof-erasure route of chapter 642. The packet records type checking as finite membership data: it does not export a host conversion oracle, semantic equality of terms, normalization theorem, hidden proof object, quotient syntax, or ambient metatheory.

Definition 3126.1 (TypeCheckingClassifier carrier)**.** *A* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, J, E, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite term row. It names the inspected term histories and their syntactic classifier.*
2. *J is the typing-ledger row. It records the finite judgment entries used by the checker.*
3. *E is the* Ext*-membership row, identifying accepted typed terms as classifier members.*
4. *D is the derivation-trace row linking each accepted membership read to a displayed typing derivation.*
5. *S is the subject-reduction row, routing accepted* Ext *membership through displayed* Cont *steps.*
6. *R is the erasure row, separating proof-term content from the erased computational term while preserving the displayed membership route.*
7. *H records componentwise* hsame *transport for term, typing-ledger,* Ext*-membership, derivation-trace, subject-reduction, erasure, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a type-checking request through membership, derivation, subject-reduction, and erasure rows.*
9. *P is the* Pkg *provenance over* $\mathsf{TypeCheckingClassifier}^{\uparrow}$*,* $\mathsf{TypeClassifierMembership}^{\uparrow}$*,* $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$*,* $\mathsf{ProofTermErasureTrace}^{\uparrow}$*,* BHist*,* Ext*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for host conversion, semantic equality of terms, an ambient normalization theorem, hidden proof objects, quotient syntax, or external metatheoretic consistency.

Lean4 (definition) `BEDC.Derived.TypeCheckingClassifierUp.TypeCheckingClassifierUp`
Lean4 ✓ `BEDC.Derived.TypeCheckingClassifierUp.TypeCheckingClassifierTasteGate single carrier alignment`

Theorem 3126.2 (TypeCheckingClassifier NameCert obligations)**.** *For every accepted* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *carrier* $Q = (T, J, E, D, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the term row, typing ledger,* Ext*-membership row, derivation trace, subject-reduction row, erasure row, componentwise transport, continuation replay, provenance, and local naming. Every accepted type-checking read must expose membership through* E*, derivation through* D*, subject reduction through* S*, and erasure separation through* R *before* H, C, P, N *package the row. No obligation is stored in host conversion, semantic equality of terms, normalization, quotient syntax, hidden proof objects, or ambient metatheory.*

Proof. Project the carrier of Definition 4106.1. The tuple lists exactly the term, judgment, membership, derivation, subject-reduction, erasure, transport, replay, provenance, and local naming rows.

Replay the continuation row C. A type-checking request first reads the finite term row T and typing ledger J, then the Ext-membership row E and derivation trace D. Subject reduction is read from S, and proof-term erasure is kept in the displayed row R.

Transport by H is componentwise over those rows. The provenance and naming rows P, N therefore package only the finite classifier route, leaving no coordinate from which a host oracle, quotient syntax, hidden proof object, or ambient metatheorem could be exported. □

Theorem 3126.3 (TypeCheckingClassifier membership soundness)**.** *Let* $Q = (T, J, E, D, S, R, H, C, P, N)$ *be an accepted* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *packet. Every successful type-checking answer exported by* Q *is an* Ext*-membership read supported by the displayed typing ledger and derivation trace:*

$$T, J \longmapsto E, D \longmapsto S, R, H, C, P, N.$$

The subject-reduction row preserves the membership read along displayed Cont *replay, and the erasure row separates proof-term evidence from erased computational content. The handoff preserves* $T, J, E, D, S, R, H, C, P, N$ *and exports no host conversion oracle, semantic term equality, normalization theorem, hidden proof object, quotient syntax, or ambient consistency principle.*

Proof. Begin with a successful type-checking answer. By carrier admission, the answer can only be read through the rows of Definition 4106.1.

The finite term row T and typing ledger J provide the displayed input to the classifier. The Ext row E records membership, and the derivation trace D records the finite route witnessing the accepted judgment.

Subject reduction is then read as the displayed row S along Cont replay, while R keeps proof-term erasure separate from the erased computational term. Componentwise transport, provenance, and naming preserve this route through H, C, P, N without adding the excluded host-level principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4106.1, Definition 4106.2, and Definition 4106.3 seed the finite term, typing-ledger, Ext-membership, derivation-trace, subject-reduction, erasure, transport, replay, provenance, and naming route.

Formal verification: FormalStatus(NameCert$_{\text{TypeCheckingClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host conversion, semantic equality of terms, normalization, hidden proof objects, quotient syntax, ambient metatheoretic consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Ext-membership soundness, derivation-trace support, subject-reduction replay, erasure separation, and structural nonescape for BEDC.Derived.TypeCheckingClassifierUp.

Constructive story: A TypeCheckingClassifier$^{\uparrow}$ packet is generated from term rows, a finite typing ledger, Ext-membership data, derivation traces, subject-reduction replay, erasure separation, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 3127

A Concrete Naming Certificate for $\mathsf{MengerNobelingEmbedding}^{\uparrow}$

Following Menger and Nobeling, the $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ packet records a finite-dimensional metric embedding handoff as BEDC NameCert data. It sits after the metric carrier of chapter 57, the compact finite-net surface of chapter 129, the distance-function route of chapter 7569, the tolerance-indexed net row of chapter 1767, and the terminal real-name boundary of chapter 38. The packet names the finite coordinate route needed by embedding consumers; it is not a host Euclidean space, not a selected infinite coordinate family, not a quotient completion, and not an ambient dimension theorem.

Definition 3127.1 (Menger–Nobeling embedding carrier)**.** *A* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, K, Q, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose point classifier and distance classifier are read by the embedding packet.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact finite-metric handoff row supplying the finite cover and compact inspection data used by the coordinate budget.*
3. *Q is the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *distance-function row that turns source-distance observations into displayed real-valued coordinate observations.*
4. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row that records the inspected center bundle, tolerance, and coverage witnesses used to choose a finite coordinate window.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the displayed coordinate values and finite separation inequalities.*
6. *H records componentwise* hsame *transport for the metric source, compact handoff, Kuratowski row, finite-net row, real readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from metric-distance reads through K, Q, F, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, Q, F, R, H, C, P, N$. It has no coordinate for a host Euclidean carrier, a selected infinite coordinate family, quotient completion, open-cover compactness, countable choice, or an ambient dimension principle.

Lean4 ✓ `BEDC.Derived.MengerNobelingEmbeddingUp.MengerNobelingEmbeddingTasteGate single carrier alignment`

Theorem 3127.2 (Menger–Nobeling embedding NameCert obligations)**.** *For every accepted* MengerNobelingEmbe *carrier $E = (M, K, Q, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, compact handoff, Kuratowski distance-function row, finite epsilon-net row, real coordinate readback, transport, replay, provenance, and local naming row;*
2. *metric-source exposure, requiring every source point and distance read to enter through M;*
3. *compact finite-net exposure, requiring every finite coordinate budget to be read through the compact row K and the finite-net row F;*
4. *Kuratowski coordinate handoff, requiring every displayed coordinate observation to pass through Q before its* $\mathsf{Real}^{\uparrow}$ *readback in R is consumed;*
5. *finite separation stability, requiring the same F, Q, R rows to witness the finite coordinate separation read under componentwise transport H;*
6. *non-escape, excluding host Euclidean carriers, selected infinite coordinate families, quotient completions, open-cover compactness, countable choice, and ambient dimension principles from the local certificate.*

No Menger–Nobeling embedding obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4107.1. The only analytic rows are the metric source M, the compact handoff K, the Kuratowski distance-function row Q, the finite-net row F, and the real coordinate readback row R.

The compact coordinate budget is finite because it is read through K and F. The compact row supplies the finite inspection context, while the finite epsilon-net row supplies the displayed center bundle, tolerance, and coverage witnesses used by the coordinate window.

The coordinate values are then forced through the Kuratowski route. The row Q exposes distance-function observations, and R reads those observations through the terminal Real boundary before any separation claim is exported.

Finally, H, C, P, N preserve and name the same route. Since the carrier has no Euclidean host, infinite coordinate-family selector, quotient-completion row, open-cover compactness row, choice schedule, or ambient dimension row, none of those objects can become a local NameCert obligation. □

Theorem 3127.3 (Menger–Nobeling embedding finite coordinate handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ *carrier must read the finite route*

$$M, K, Q, F, R, H, C, P, N.$$

Thus a finite-dimensional embedding request reaches coordinate data only after the metric source, compact finite-net budget, Kuratowski distance-function observations, and real readbacks are visible. The handoff exports no host Euclidean carrier, selected infinite coordinate family, quotient completion, open-cover compactness, countable choice, or ambient dimension principle.

Proof. Begin with an accepted carrier $E = (M, K, Q, F, R, H, C, P, N)$. A consumer can enter the packet only through the metric source M and the compact handoff K.

The finite coordinate window is opened next. The Kuratowski row Q supplies distance-function observations, the finite epsilon-net row F supplies the displayed center and coverage budget, and the Real row R supplies the coordinate readbacks.

The structural rows then package exactly that handoff. Componentwise transport H preserves the same finite route, replay C exposes it to downstream consumers, provenance P records the package lineage, and N names the local certificate. Therefore the embedding consumer receives only the displayed finite coordinate handoff and not any hidden host embedding principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4107.1, Definition 4107.2, and Definition 4107.3 seed the finite coordinate embedding route over MetricUp, CompactMetricUp, KuratowskiEmbeddingUp, FiniteEpsilonNetUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host Euclidean embedding, selected infinite coordinate families, quotient completion, open-cover compactness, countable choice, ambient dimension principles, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, Kuratowski coordinate handoff, finite separation stability, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.MengerNobelingEmbe

Constructive story: A $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp handoff, a KuratowskiEmbeddingUp distance-function row, a FiniteEpsilonNetUp coordinate budget, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3128

A Concrete Naming Certificate for $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ packet records the completion route in which a displayed Cauchy filter is read through a uniform-space entourage basis before any quotient completion or selected limit is available. It sits after the uniform-space surface of chapter 339, the filter-basis generator of chapter 1501, the saturated Cauchy-filter packet of chapter 333, and the completion boundary of chapter 95. Its finite readback route uses the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38.

Definition 3128.1 (Uniform-space completion-filter carrier)**.** *A* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, B, F, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, including the finite diagonal, refinement, symmetry, and composition data visible to the packet.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row, listing the basis elements and common-refinement witnesses read from U.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturated filter row generated from B.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule controlling which filter and entourage reads may be inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows and tolerance requests.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for entourage shrinkage, Cauchy refinement, and completion-radius comparison.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *completion-admission seal reached only after U, B, F, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from a completion request through the uniform, basis, filter, window, readback, tolerance, and Real-seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, B, F, S, R, D, A, H, C, P, N$. It has no coordinate for quotient filters, host equality of filters, selected representatives, countable choice, a chosen completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformSpaceCompletionFilterUp.UniformSpaceCompletionFilterTasteGate`
`single carrier alignment`

Theorem 3128.2 (Uniform-space completion-filter NameCert obligations)**.** *For every accepted* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *carrier $Q = (U, B, F, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionF}}$ *surface consists exactly of carrier admission for the displayed uniform-space entourage row, Cauchy-filter basis row, saturated Cauchy-filter row, stream-window row, regular-rational readback row, dyadic tolerance ledger, Real completion-admission seal, componentwise transport, continuation replay, package provenance, and local naming row. Every completion-facing read of Q must pass through*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

before the completion boundary of $\mathsf{Completion}^{\uparrow}$ *may consume the packet.*

Proof. Project the carrier of Definition 4108.1. The coordinates list exactly the uniform, basis, filter, window, readback, tolerance, Real-seal, transport, replay, provenance, and local naming rows.

The completion-facing order is forced by C. A consumer first reads the entourage data U, then the basis row B and saturated filter row F, and only then inspects the finite windows S, dyadic ledger D, regular rational readback R, and Real seal A.

The structural rows do not add a second source. The transport row H moves only the displayed coordinates componentwise, while P and N name the same finite packet as provenance and local certificate data.

Since no coordinate of Q names quotient filters, host filter equality, selected representatives, countable choice, a chosen completion point, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot be exported by the local NameCert surface. The listed rows therefore exhaust the obligations. □

Theorem 3128.3 (Uniform-space completion-filter boundary)**.** *Let $Q = (U, B, F, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *packet. Any* $\mathsf{Completion}^{\uparrow}$ *consumer of Q receives a finite filter-completion route through the displayed rows $U, B, F, S, R, D, A, H, C, P, N$ and receives no quotient-filter class, selected completion point, host filter equality, countable-choice selector, hidden cofinal subfilter, or ambient completeness theorem.*

Proof. Begin with the NameCert obligation theorem, Definition 4108.2. It routes completion-facing reads through the uniform row, Cauchy-filter basis, Cauchy filter, finite windows, dyadic tolerance ledger, regular readback, and Real seal before the completion boundary consumes the packet.

The replay and transport rows preserve the same route. They do not add representatives, quotient classes, hidden cofinal filters, or selected limits; they only replay and transport the displayed coordinates.

Finally inspect the carrier definition. The excluded objects are absent from Q, so a $\mathsf{Completion}^{\uparrow}$ consumer cannot extract them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}$.

Constructive story: A $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ packet is generated from a UniformSpace entourage row, CauchyFilterBasis generator row, CauchyFilter saturation row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a RealUp completion-admission seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4108.1, Definition 4108.2, and Definition 4108.3 seed the non-quotient filter-completion route for UniformSpaceUp, CauchyFilterBasisUp, CauchyFilterUp, CompletionUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filter completion, selected completion points, host filter equality, countable choice, hidden cofinal subfilters, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-to-filter saturation, completion-facing replay, Real-seal handoff, and refusal of quotient-filter and selected-limit escapes for BEDC.Derived.UniformSpaceCompletionFilterUp.

Chapter 3129

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$

Following Bishop–Bridges regular Cauchy real names and Kelley filter terminology, the RegularCauchyFilterCompleti packet records the L10 route by which a regular Cauchy name is read as a Cauchy-filter completion object through finite BEDC rows. It sits after the filter-limit boundary of chapter 1558, the completion-facing filter surface of chapter 1569, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives filter-completion consumers a canonical regular-name bridge without quotient filters, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3129.1 (Regular Cauchy filter completion carrier)**.** *A* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, F, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary row available only after the source filter and regular-name requests have been replayed.*
2. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-facing row whose consumers may read only the displayed filter route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for regular Cauchy tail requests and filter-completion reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the requested windows, comparison radius, and filter limit slack.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, F, S, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the limit, completion, window, tolerance, readback, seal, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a regular Cauchy filter-completion request through $L, F, S, D, R, E, H, P, N$.*

9. *P records* Pkg *provenance over* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, F, S, D, R, E, H, C, P, N$. It stores no quotient filter, selected completion point, hidden cofinal sequence, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterCompletionUp.RegularCauchyFilterCompletionTasteGate single carrier alignment`

Theorem 3129.2 (Regular Cauchy filter completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *carrier*

$$Q = (L, F, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary exposure through L,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer exposure through F, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal discipline through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4109.1. The tuple displays the filter-limit boundary, completion-facing row, window schedule, dyadic tolerance ledger, regular rational readback, terminal real seal, and structural rows.

Read the analytic route in order. The limit row L and completion row F place the request inside the Cauchy-filter completion surface before any finite regular-name data are inspected.

The regular-name data are then exhausted by S, D, R, E. The stream windows in S are measured by the dyadic ledger D, the regular rational row R is read from that displayed data, and the real seal E appears only after those rows have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. No remaining coordinate can supply quotient filters, selected representatives, hidden cofinal choices, host equality, or ambient completeness, so the listed rows are exactly the local NameCert obligations. □

Theorem 3129.3 (Regular Cauchy filter completion seal handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *packet receives the* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{CauchyFilterLimit}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff is a filter-completion reading of a regular Cauchy name, not a quotient filter construction, selected completion point, host real equality, or ambient completeness principle.

Proof. Apply Definition 4109.2. It orders the completion-facing read through L, F, S, D, R, E before the terminal seal is visible.

The first two rows bind the filter-completion side. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ row records the limit-facing boundary, and the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row records the completion consumer that may inspect the same displayed route.

The remaining rows bind the regular-name side. The finite windows, dyadic ledger, and regular rational readback feed the terminal real seal, while H, C, P, N only preserve and name that route.

Thus any consumer that reaches $\mathsf{Real}^{\uparrow}$ without passing through the listed route leaves the carrier. The handoff supplies a finite regular Cauchy filter-completion bridge and no hidden completion object. □

Theorem 3129.4 (Regular Cauchy filter completion nonescape)**.** *No accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *packet exports quotient filter equality, a selected representative, a chosen cofinal tail, host real equality, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*. Every exported completion or real-facing read is confined to the rows* $L, F, S, D, R, E, H, C, P, N$*.*

Proof. Inspect the carrier definition. Its non-structural rows are exactly the filter-limit boundary L, completion row F, stream windows S, dyadic tolerance ledger D, regular rational readback R, and real seal E.

The structural rows add no analytic principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet rather than adding quotient filters, selected representatives, cofinal choices, or host equality.

Use the seal handoff of Definition 4109.3. Every completion or real-facing read must pass through the displayed finite route, so none of the excluded data can escape from the packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: cauchyfilterlimit, cauchyfiltercompletion, streamname, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ because this packet binds the filter-limit boundary, completion-facing row, finite windows, tolerance ledger, regular readback, terminal seal, transport, replay, provenance, and local naming as one L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ packet is generated from a CauchyFilterLimit boundary row, CauchyFilterCompletion consumer row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4109.1, Definition 4109.2, Definition 4109.3, and Definition 4109.4 seed the L10 regular Cauchy filter-completion route over CauchyFilterLimitUp, CauchyFilterCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, filter-limit exposure, completion-facing exposure, window exactness, tolerance exactness, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.RegularCauchyFilterCompletionUp.

Chapter 3130

A Concrete Naming Certificate for $\mathsf{CauchyFilterRegularization}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{CauchyFilterRegularization}^{\uparrow}$ packet records the L10 handoff that turns a displayed Cauchy-filter completion route into a regular Cauchy real-name route through finite BEDC rows. It sits after the Cauchy-filter surface of chapter 333, the completion boundary of chapter 1569, the Polish carrier discipline of chapter 1490, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite regularization handoff; it does not quotient filters, select representatives, import countable choice, assert host equality of limits, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3130.1 (Cauchy-filter regularization carrier)**.** *A* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, W, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row, naming the finite filter observations and the completion-facing boundary they expose.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for regularization requests and tail comparisons.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows, comparison radii, and completion requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed windows and dyadic tolerance data.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the filter-completion source, stream windows, dyadic ledger, regular readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a regularization request through F, W, D, R, L, H, P, N.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyFilterRegularization}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, D, R, L, H, C, P, N$. It stores no quotient filter, selected completion point, hidden cofinal subsequence, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ BEDC.Derived.CauchyFilterRegularizationUp.CauchyFilterRegularizationTasteGate single carrier alignment

Theorem 3130.2 (Cauchy-filter regularization NameCert obligations)**.** *For every accepted* CauchyFilterRegularizati *carrier*

$$G = (F, W, D, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed source, window, tolerance, readback, seal, transport, replay, provenance, and local-name rows;*
2. *filter exposure, requiring every regularization read to expose the* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row F before a stream window is inspected;*
3. *window and tolerance exactness, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row W and* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D to determine every finite comparison used by the handoff;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the regular rational row R to be read from the displayed W, D data and not from a hidden representative;*
5. $\mathsf{Real}^{\uparrow}$*-seal nonescape, requiring the terminal row L to be consumed only after F, W, D, R have been displayed;*
6. *provenance, requiring H, C, P, N to transport, replay, source, and name the same finite packet without adding quotient filters, selected limits, choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 4110.1. The source row F exposes the Cauchy-filter completion boundary, and the rows W, D, R, L are the only non-structural regularization rows downstream from that boundary.

The window and tolerance part is finite. A regularization request opens the listed $\mathsf{StreamName}^{\uparrow}$ windows in W, reads the dyadic tolerance budget in D, and then obtains the $\mathsf{RegSeqRat}^{\uparrow}$ row R from those displayed observations.

The real-seal handoff is last. The terminal row L is reachable only after F, W, D, R have been exposed, while H, C, P, N only transport, replay, source, and name the same packet. Since no remaining coordinate stores quotient filters, selected representatives, hidden cofinal choices, host real equality, or ambient completeness, the listed clauses exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterRegularization}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed CauchyFilter and CauchyFilterCompletion source, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: cauchyfiltercompletion, polishspace, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyFilterRegularization}^{\uparrow}$ is not bijective to any listed sibling because it binds a Cauchy-filter completion boundary, finite window schedule, dyadic tolerance ledger, regular rational handoff, Real seal, transport, replay, provenance, and local naming into one L10 regularization certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$.

Constructive story: A CauchyFilterRegularization$^{\uparrow}$ packet is generated from a CauchyFilter/CauchyFilterCompletion source row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyFilterRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 4110.1 and Definition 4110.2 seed the L10 Cauchy-filter regularization handoff over CauchyFilterUp, CauchyFilterCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyFilterRegularization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, filter exposure, window exactness, tolerance exactness, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterRegularizationUp.

Chapter 3131

A Concrete Naming Certificate for $\mathsf{RealInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealInterval}^{\uparrow}$ packet records displayed real interval endpoints and finite enclosure rows for the L10 Real and Completion route. It sits after the located interval surface of chapter 394, the rational interval surface of chapter 354, and the dyadic-to-real spine of chapter 291, chapter 126, chapter 301, and chapter 38. The chapter gives interval-completion and real-enclosure consumers one named endpoint-and-enclosure packet; it is not a host closed interval, selected endpoint representatives, trichotomy, compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3131.1 (RealInterval carrier)**.** *A* $\mathsf{RealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, E, D, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint* $\mathsf{Real}^{\uparrow}$ *row, read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal discipline.*
2. *U is the upper endpoint* $\mathsf{Real}^{\uparrow}$ *row, read through the same finite real-name seal discipline.*
3. *E is the enclosure and order row, binding the lower endpoint, upper endpoint, rational interval enclosure, and endpoint-order evidence without replacing the endpoints by host representatives.*
4. *D is the dyadic tolerance row supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*; it records positive precision requests, outward rounding budgets, and finite endpoint comparison tolerances.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the two endpoint streams and the interval enclosure reads visible at the requested tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the endpoint windows and dyadic tolerance budget into regular rational observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint and enclosure consumers; every Real-facing output must pass through this row after D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for endpoints, enclosure and order data, dyadic tolerance, windows, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from interval membership, enclosure, refinement, or completion-facing requests through $L, U, E, D, W, R, S, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed real interval endpoint-and-enclosure history. It has no coordinate for host interval equality, selected quotient representatives, real-order trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalTasteGate single carrier alignment`

Theorem 3131.2 (RealInterval NameCert obligations)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower endpoint, upper endpoint, enclosure and order row, dyadic tolerance row, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name;*
2. *endpoint order, requiring E to bind L and U before any interval read can be treated as an enclosure;*
3. *enclosure stability, requiring every containment, refinement, and endpoint-bound read to pass through E together with the displayed tolerance and window rows;*
4. *dyadic tolerance handoff, requiring D to precede the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R;*
5. *Real seal non-escape, requiring every Real-facing endpoint or enclosure output to reach S only after L, U, E, D, W, R have been displayed;*
6. *ledger refusal, excluding host interval equality, selected endpoint representatives, real-order trichotomy, countable choice, compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No real interval obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4111.1. The projection exposes exactly the two endpoint rows, enclosure and order row, dyadic tolerance row, stream windows, regular rational readback, Real seal, and structural rows.

Endpoint order and enclosure stability are read from E. That row is the only coordinate joining L and U into an interval boundary, so containment, refinement, and endpoint-bound requests cannot bypass it.

The dyadic tolerance handoff is the route $D \mapsto W \mapsto R$. A positive precision request first enters the dyadic budget, then selects finite stream windows, and only then produces regular rational readback.

The Real seal row S is downstream of that route. The remaining rows H, C, P, N transport, replay, package, and name the same endpoint-and-enclosure history, while the excluded host interval, choice, compactness, and completeness objects have no coordinate in the carrier. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalCarrier namecert obligations`

Theorem 3131.3 (RealInterval seal enclosure route)**.** *Every endpoint, enclosure, refinement, or completion-facing read of an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$L, U, E \longmapsto D, W, R \longmapsto S, H, C, P, N.$$

Consequently every interval consumer receives endpoint order, finite enclosure data, dyadic tolerance, stream windows, and regular rational readback before any Real-facing seal is visible. The route exports no host interval equality, selected endpoint representative, real-order trichotomy, countable-choice schedule, compactness theorem, hidden complete space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $I = (L, U, E, D, W, R, S, H, C, P, N)$. The endpoint rows L and U are the only Real endpoints in the packet, and E is the only row that binds them as an ordered enclosure.

Read the dyadic tolerance handoff next. The row D supplies the finite precision and outward-rounding budget, W exposes the endpoint stream windows visible at that budget, and R turns those windows into regular rational readback.

Only after this handoff may the Real-facing seal S be consumed. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same route. Since none of the excluded host interval or completeness objects is a carrier coordinate, every interval read reaches the seal only through the displayed rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalSealEnclosureRoute`

Theorem 3131.4 (RealInterval endpoint-order obligation)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *be an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier. Any consumer that treats* L *and* U *as an interval must first read the endpoint-order row* E*; the two endpoint rows alone do not authorize an interval enclosure.*

Proof. Project the carrier of Definition 4111.1. The rows L and U name the two endpoint Real rows, while E is the only displayed row that binds them as an ordered enclosure.

Now apply the local obligations of Definition 4111.2. The endpoint-order obligation explicitly requires E to bind L and U before any interval read can be treated as an enclosure.

The seal route of Definition 4111.3 preserves the same order, since every endpoint or enclosure read begins with L, U, E before entering the dyadic handoff. Thus an interval consumer cannot bypass E by reading the endpoint rows alone. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointOrderObligation`

Theorem 3131.5 (RealInterval classifier transport)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *and* $I' = (L', U', E', D', W', R', S', H', C', P', N')$ *be accepted* $\mathsf{RealInterval}^{\uparrow}$ *carriers. If the accepted interval classifier identifies the endpoint and enclosure rows of* I *with those of* I'*, then carrier admission transports from* I *to* I' *only through the displayed endpoint, locatedness, dyadic-window,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. In particular, classifier equality cannot replace the rows supplied by chapter 291, chapter 394, chapter 301, chapter 126, or chapter 38.*

Proof. Project both carriers through Definition 4111.1. The endpoint rows L, U, L', U' and enclosure rows E, E' are the only rows on which the interval classifier can act.

Use the endpoint-order obligation Definition 4111.4. It forces every interval read to keep the endpoint order row visible, so classifier equality must transport the ordered enclosure rather than only the two endpoints.

The seal route of Definition 4111.3 then carries the transported enclosure through the dyadic tolerance, stream-window, regular-rational, and Real-seal rows. The structural rows H', C', P', N' name the transported packet, so carrier admission moves only along the displayed BEDC rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalClassifierTransport`

Theorem 3131.6 (RealInterval endpoint refinement stability)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, any dyadic refinement of the endpoint enclosure that preserves the displayed endpoint order and locatedness rows preserves the* $\mathsf{RealInterval}^{\uparrow}$ *NameCert surface before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read. The refinement may change the finite dyadic tolerance and window rows* D, W, R*, but it must keep the endpoint, locatedness, transport, replay, provenance, and local naming rows visible before* S *is consumed.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointRefinementStability`

Proof. Read the carrier rows of Definition 4111.1. A dyadic endpoint refinement changes only the tolerance, window, and regular-rational observations recorded in D, W, R.

Apply Definition 4111.2. Its endpoint order and enclosure stability clauses keep L, U, E visible, while its dyadic tolerance handoff clause keeps every refinement inside the $D \mapsto W \mapsto R$ route.

Finally use the seal enclosure route Definition 4111.3. Since S is terminal, the refined endpoint surface must already have passed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the displayed structural rows before any $\mathsf{Real}^{\uparrow}$ seal is available. □

Theorem 3131.7 (RealInterval dyadic-window handoff)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, the displayed route*

$$D \longmapsto W \longmapsto R$$

is the only tolerance-to-readback handoff supplied by the packet.

Proof. Project Definition 4111.1. The tolerance coordinate is D, the finite endpoint-window coordinate is W, and the regular rational readback coordinate is R.

Use Definition 4111.2. Its dyadic tolerance handoff row requires D to precede the $\mathsf{StreamName}^{\uparrow}$ windows W and the $\mathsf{RegSeqRat}^{\uparrow}$ readback R.

Finally apply Definition 4111.3. The route displays no second tolerance-to-readback path; every enclosure read moves from D to W to R before any seal or structural row is public. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalDyadicWindowHandoff`

Theorem 3131.8 (RealInterval Real-seal nonescape)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *be an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier. Every Real-facing enclosure read reaches the seal row* S *only after the endpoint rows, endpoint-order row, dyadic tolerance row, stream-window row, and regular readback row* L, U, E, D, W, R *have all been displayed.*

Proof. Start from the carrier of Definition 4111.1. The seal row S is not an independent endpoint; it is the terminal Real-facing row for endpoint and enclosure consumers.

The obligations of Definition 4111.2 make Real seal non-escape explicit. They require every Real-facing endpoint or enclosure output to reach S only after L, U, E, D, W, R have been displayed.

The route theorem Definition 4111.3 gives the same ordering as a displayed path. Therefore a Real-facing enclosure read cannot reach S while omitting any of the endpoint, order, tolerance, window, or readback rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalRealSealNonescape`

Theorem 3131.9 (RealInterval obligation-closure route)**.** *Every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *routes endpoint admission, endpoint order, enclosure stability, dyadic-window handoff, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *through the displayed rows* $L, U, E, D, W, R, S, H, C, P, N$*.*

The route supplies no host interval equality, selected endpoint representative, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalCarrier obligation closure route`

Proof. First read the local obligation surface from Definition 4111.2. It exposes carrier admission, endpoint order, enclosure stability, dyadic tolerance handoff, Real seal non-escape, transport, replay, provenance, and local naming as finite rows of the accepted carrier.

Next use the route theorems that keep these rows in order. The seal enclosure route of Definition 4111.3, the endpoint-order obligation of Definition 4111.4, and the dyadic-window handoff of Definition 4111.7 force every interval read through L, U, E, D, W, R, S before the structural rows are consumed.

Finally apply Real-seal nonescape from Definition 4111.8. Since every Real-facing enclosure read reaches S only after L, U, E, D, W, R have been displayed, and since H, C, P, N only transport, replay, package, and name that same route, the carrier closes exactly the obligation route stated above while leaving the refused global principles outside the packet. □

Theorem 3131.10 (RealInterval public NameCert export)**.** *The public* $\mathsf{RealInterval}^{\uparrow}$ *export is exactly the endpoint-pair carrier together with endpoint order, classifier transport, endpoint-refinement stability, dyadic-window handoff, and Real-seal nonescape rows. A public interval consumer may cite the ordered surface*

$$L, U, E, D, W, R, S, H, C, P, N$$

only as the endpoint-and-enclosure packet of Definition 4111.1; it may not add host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalPublicNameCertExport`

Proof. Start from the carrier and obligation route. Theorems 4111.1 and 4111.9 identify the complete endpoint-and-enclosure packet as the rows $L, U, E, D, W, R, S, H, C, P, N$.

The public surface is determined by the interval-facing route theorems. Endpoint order is supplied by Definition 4111.4, classifier transport by Definition 4111.5, endpoint-refinement stability by Definition 4111.6, dyadic-window handoff by Definition 4111.7, and Real-seal nonescape by Definition 4111.8.

These theorems cite the same displayed rows and no additional host object. Therefore the reusable public export is precisely the endpoint-pair carrier with its order, transport, refinement, dyadic-window, and seal rows, while the refused global principles remain outside the packet. □

Theorem 3131.11 (RealInterval finite-net consumer)**.** *Every finite-net or density-net consumer of an accepted* $\mathsf{RealInterval}^{\uparrow}$ *packet must read the endpoint order row* E*, dyadic tolerance row* D*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* S *before passing interval data into chapter 5966 or chapter 6003. The consumer route is the displayed packet*

$$L, U, E, D, W, R, S, H, C, P, N,$$

not a host interval equality, selected endpoint representative, compactness theorem, countable-choice net, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFiniteNetConsumer`

Proof. Start with endpoint order. By Definition 4111.4, a consumer cannot treat L and U as an interval until it has read the order and enclosure row E.

Next apply the dyadic-window handoff of Definition 4111.7. It forces the requested tolerance through D, then the finite StreamName windows W, and only then the regular rational readback row R used by dyadic interval cells or finite density rows.

Now use Real-seal nonescape and the public NameCert export, Definition 4111.8 and Definition 4111.10. The seal row S is available only after L, U, E, D, W, R have been displayed, and the public export keeps H, C, P, N attached to that same endpoint-and-enclosure packet.

Finally route the displayed interval data to the dyadic interval-net and finite density-net sibling chapters. Since Definition 4111.1 contains no coordinate for host interval equality, selected endpoint representatives, compactness, countable choice, or ambient completeness, those siblings receive only the finite displayed route stated above. □

Theorem 3131.12 (RealInterval bridged NameCert route)**.** *Every bridge-facing* $\mathsf{RealInterval}^{\uparrow}$ *consumer routes through the displayed endpoint-and-enclosure packet*

$$L, U, E, D, W, R, S, H, C, P, N.$$

The route first reads the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger* D*, then the* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *rows carried by* E*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and finally the* $\mathsf{Real}^{\uparrow}$ *seal* S *with transport, replay, provenance, and local naming still visible.*

Proof. Project the accepted carrier of Definition 4111.1. The endpoint rows, enclosure row, tolerance ledger, windows, readback, seal, and structural rows are all coordinates of one packet.

Use the NameCert obligations and public export, Definition 4111.2 and Definition 4111.10. They keep endpoint order, enclosure stability, dyadic-window handoff, Real-seal nonescape, transport, replay, provenance, and local naming attached to the displayed rows.

Finally apply the finite-net consumer route of Definition 4111.11. It shows that downstream interval data reaches finite cell and density consumers only through the dyadic, located-enclosure, StreamName, RegSeqRat, and Real-seal route. Hence the bridge-facing NameCert route is the displayed packet above. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalBridgedNameCertRoute`

Theorem 3131.13 (RealInterval finite endpoint-envelope route)**.** *Every* $\mathsf{RealInterval}^{\uparrow}$ *endpoint consumer that exports interval membership must first factor through the displayed finite endpoint envelope*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealIntervalEnclosure}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The dyadic endpoint arithmetic is read from D*, the finite endpoint windows from* W*, the regular readback from* R*, the enclosure row from* E*, and the terminal value seal from* S *before membership is public. The envelope exports no quotient stream equality, selected endpoints, host interval equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, compactness, or countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFiniteEndpointEnvelopeRoute`

Proof. Start with the endpoint-order obligation of Definition 4111.4. It keeps the enclosure and order row E visible whenever a consumer treats L and U as interval endpoints.

Next apply the dyadic-window handoff Definition 4111.7. The endpoint arithmetic enters through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row D, then exposes only finite $\mathsf{StreamName}^{\uparrow}$ windows W, and only then produces the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Use the bridged NameCert route Definition 4111.12. It inserts the $\mathsf{LocatedInterval}^{\uparrow}$ and $\mathsf{RealIntervalEnclosure}^{\uparrow}$ rows carried by E between the endpoint ledger and the terminal $\mathsf{Real}^{\uparrow}$ seal S, while H, C, P, N transport, replay, package, and name the same displayed route.

Finally read Real-seal nonescape from Definition 4111.8. Since S is available only after L, U, E, D, W, R have been displayed, interval membership cannot be exported from quotient stream equality, selected endpoints, host interval equality, ambient completeness, trichotomy, compactness, or countable choice. □

Theorem 3131.14 (RealInterval endpoint enclosure transport)**.** *For accepted* $\mathsf{RealInterval}^{\uparrow}$ *carriers* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *and* $I' = (L', U', E', D', W', R', S', H', C', P', N')$*, componentwise* hsame *transport preserves both endpoint enclosure rows before interval membership is public. In particular, a consumer that transports* L *to* L' *and* U *to* U' *must also transport the enclosure-and-order row* E *to* E' *and the dyadic-window-readback route* D, W, R *to* D', W', R' *before it may read the terminal seal.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointEnclosureTransport`

Proof. Start with classifier transport, Definition 4111.5. It allows carrier admission to move only through the displayed endpoint, enclosure, dyadic-window, hsame, Cont, Pkg, and NameCert rows.

Endpoint order then fixes the enclosure boundary. By Definition 4111.4, the two endpoint rows alone do not authorize interval membership; the row E must remain visible as the ordered enclosure.

Finally use endpoint-refinement stability and the dyadic-window handoff, Definition 4111.6 and Definition 4111.7. These rows keep D, W, R attached to the transported enclosure before S is consumed, so componentwise transport preserves both endpoint enclosure rows before membership becomes public. □

Theorem 3131.15 (RealInterval StreamName Real-seal nonescape)**.** *No* $\mathsf{RealInterval}^{\uparrow}$ *read that passes through* $\mathsf{StreamName}^{\uparrow}$ *windows and a* $\mathsf{Real}^{\uparrow}$ *seal exports open-cover compactness, quotient real equality, choice-selected endpoints, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness. The only public route is*

$$L, U, E, D, W, R, S, H, C, P, N,$$

with W *and* R *displayed before* S *and with the refusal boundary preserved by the local NameCert rows.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalStreamRealSealNonescape`

Proof. Read the Real-seal nonescape theorem Definition 4111.8. It places the seal row S strictly after the endpoint, order, tolerance, window, and readback rows L, U, E, D, W, R.

Use the obligation-closure route and public NameCert export, Definition 4111.9 and Definition 4111.10. They state that H, C, P, N only transport, replay, package, and name the same displayed packet.

The finite-net consumer theorem Definition 4111.11 keeps downstream interval reads inside the same route. Since none of the carrier coordinates names open-cover compactness, quotient equality, selected endpoints, trichotomy, or ambient completeness, the StreamName-to-Real-seal read cannot export those objects. □

Theorem 3131.16 (RealInterval bridge scope)**.** *Every bridge-facing* $\mathsf{RealInterval}^{\uparrow}$ *read factors through the finite endpoint-and-enclosure rows*

$$L, U, E, D, W, R, S, H, C, P, N$$

and by the sibling surfaces $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *only as endpoint-enclosure readback. The bridge-scope packet cannot be read as host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the public export of Definition 4111.10. It binds public interval reads to the endpoint, enclosure, tolerance, window, readback, seal, transport, replay, provenance, and local naming rows of the displayed carrier.

Next use the bridged route and endpoint-enclosure transport, Definition 4111.12 and Definition 4111.14. These results keep the bridge-facing read on the located endpoint row, dyadic interval envelope, finite stream windows, regular rational readback, and terminal Real seal before any interval membership is public.

Finally apply the StreamName Real-seal nonescape theorem, Definition 4111.15. The seal row S remains downstream of L, U, E, D, W, R and upstream of no additional host principle, so the bridge scope is exactly the displayed endpoint-enclosure packet. Compactness, selected endpoint representatives, trichotomy, countable choice, and ambient completeness have no row in this factorization. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalBridgeScope`

Theorem 3131.17 (RealInterval formal target handoff)**.** *Any future Lean target that advances the formal status of* $\mathsf{RealInterval}^{\uparrow}$ *must expose the endpoint carrier rows* L, U*, the dyadic-window handoff rows* D, W, R*, the endpoint-order obligation* E*, the Real-seal nonescape row* S*, and the bridge-scope rows* H, C, P, N *before it may claim a checked public interval surface. In particular, the target must route through the carrier of Definition 4111.1, the finite-net consumer of Definition 4111.11, the finite endpoint envelope of Definition 4111.14, the StreamName Real-seal nonescape theorem of Definition 4111.15, and the bridge scope of Definition 4111.16.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFormalTargetHandoff`

Proof. Start with the carrier definition. It presents the endpoint rows L, U, the endpoint-order and enclosure row E, the dyadic tolerance row D, the $\mathsf{StreamName}^{\uparrow}$ window row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, the $\mathsf{Real}^{\uparrow}$ seal row S, and the structural rows H, C, P, N in one accepted packet.

Use the dyadic-window handoff and finite-net consumer theorems. They keep downstream real-completion and net reads on D, W, R before the seal row is consumed, so a formal target cannot replace those rows by an ambient interval object.

Apply endpoint-enclosure transport and StreamName Real-seal nonescape. The endpoint envelope must transport both endpoints together with E, while the seal theorem keeps S from exporting quotient real equality, selected endpoints, trichotomy, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the bridge-scope theorem. It leaves only H, C, P, N as transport, replay, provenance, and local naming for the same packet, so any checked Lean target must display exactly these handoff rows before the formal status can advance. □

Theorem 3131.18 (RealInterval mature consumer route)**.** *Every mature* $\mathsf{RealInterval}^{\uparrow}$ *consumer read factors through the endpoint-and-enclosure route already displayed by* $\mathsf{DyadicIntervalNet}^{\uparrow}$*,* FiniteCauchyDensityN

LocatedInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, *and* Real$^{\uparrow}$. *The route exposes lower endpoint L, upper endpoint U, enclosure and order row E, dyadic tolerance row D, finite window row W, regular-rational readback row R, terminal real seal S, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before any downstream interval read is accepted. It does not export open-cover compactness, quotient real equality, selected endpoint representatives, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalMatureConsumerRoute`

Proof. Start with the endpoint carrier and finite-net consumer theorems. The carrier displays $L, U, E, D, W, R, S, H, C, P, N$, while Definition 4111.11 routes net-facing reads through the dyadic and finite Cauchy density surfaces before any public interval endpoint is consumed.

Next apply the endpoint-enclosure and StreamName Real-seal results. By Definition 4111.14, endpoint transport keeps E attached to L and U; by Definition 4111.15, the StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows feed the terminal Real$^{\uparrow}$ seal without adding quotient equality or completeness.

Finally use the bridge-scope and formal-target handoff theorems. They identify LocatedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ as readback surfaces for the same displayed packet. The density and interval-net consumers therefore see only the endpoint-and-enclosure route, so the mature route closes without importing compactness, selected endpoints, trichotomy, countable choice, or ambient real completeness. □

Closure status for NameCert$_{\text{RealInterval}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RealInterval}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 4111.1, Definition 4111.2, Definition 4111.3, Definition 4111.4, Definition 4111.7, Definition 4111.8, Definition 4111.9, Definition 4111.10, Definition 4111.16, and Definition 4111.18 bind endpoint admission, endpoint order, enclosure stability, dyadic-window handoff, finite net readback, Real-seal nonescape, hsame transport, Cont replay, Pkg provenance, and local NameCert to the displayed mature RealInterval$^{\uparrow}$ rows over DyadicIntervalNet$^{\uparrow}$, FiniteCauchyDensityNet$^{\uparrow}$, LocatedInterval$^{\uparrow}$, RationalInterval$^{\uparrow}$, RealIntervalEnclosure$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{RealInterval}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Verification closure requires checked carrier admission, endpoint enclosure, dyadic-window handoff, finite-net consumer routing, Real-seal nonescape, transport, replay, provenance, local naming, and bridge interpretation for BEDC.Derived.RealIntervalUp.

Constructive story: A RealInterval$^{\uparrow}$ packet is generated from lower and upper RealUp endpoint rows, an enclosure and order row, a DyadicRatCoreUp tolerance row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3132

A Concrete Naming Certificate for $\mathsf{SequentialSpace}^{\uparrow}$

Following Franklin and Engelking on sequential spaces, the $\mathsf{SequentialSpace}^{\uparrow}$ packet records the finite convergence interface by which sequence-indexed observations may be consumed as topological data without importing open-cover compactness, quotient point equality, or a selected subsequence principle. It sits beside the limit carrier of chapter 3845, the Cauchy source interface of chapter 1847, the finite windows of chapter 126, the indexed-open topology surface of chapter 88, and the terminal real seal of chapter 38. Later completion and compactness consumers may read sequential convergence only through this displayed packet.

Definition 3132.1 (SequentialSpace carrier)**.** *A* $\mathsf{SequentialSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, W, L, T, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the sequence-index row, naming the finite initial segment, tail request, and comparison scale read by the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite observations of the sequence at the requested indices.*
3. *L is the* $\mathsf{LimitSpace}^{\uparrow}$ *convergence row, recording the limit-facing test supplied by chapter 3845.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-test row, recording the indexed open neighbourhood test through which sequential convergence is observed.*
5. *Q is the* $\mathsf{CauchySpace}^{\uparrow}$ *comparison row when the sequence is also read by a completion or Cauchy consumer.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row when the displayed observations are regular real-name reads.*
7. *H records componentwise* hsame *transport for the sequence index, StreamName window, limit row, topology row, Cauchy comparison row, Real endpoint row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a sequential-convergence request through $I, W, L, T, Q, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LimitSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$ *row.*
The carrier has no coordinate for open-cover compactness, quotient point equality, selected subsequences, countable-choice selectors, host topological closure, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialSpaceUp.SequentialSpaceTasteGate single carrier alignment`

Theorem 3132.2 (SequentialSpace NameCert obligations)**.** *For every accepted* $\mathsf{SequentialSpace}^\uparrow$ *carrier*

$$S = (I, W, L, T, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$ *surface consists exactly of sequence-index admission through I, finite observation through W, limit-space convergence through L, topology open-test exposure through T, Cauchy comparison through Q when present, Real endpoint exposure through R when present, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.SequentialSpaceUp.SequentialSpaceCarrier namecert obligations`

Proof. Project the carrier of Definition 4112.1. The sequence-facing side is exhausted by I and W, so a consumer can inspect only displayed finite StreamName windows at displayed indices.

The convergence-facing side is exhausted by L, T, Q, R. The limit row supplies the sequential convergence request, the topology row supplies the open-test interface, and the Cauchy and Real rows are available only when the same packet displays those comparison surfaces.

The rows H, C, P, N transport, replay, source, and name exactly this data. Since the carrier has no coordinate for open-cover compactness, quotient point equality, selected subsequences, countable choice, host closure, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3132.3 (SequentialSpace completion-facing route)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentialSpace}^\uparrow$ *packet reads sequential convergence through the finite route*

$$\mathsf{StreamName}^\uparrow,\ \mathsf{LimitSpace}^\uparrow,\ \mathsf{Topology}^\uparrow,\ \mathsf{CauchySpace}^\uparrow,\ \mathsf{Real}^\uparrow.$$

The route exposes finite sequence windows, a displayed limit test, an indexed-open neighbourhood test, a Cauchy comparison surface when relevant, and a terminal Real endpoint row; it exports no open-cover compactness, selected subsequence, quotient point equality, countable-choice selector, or ambient Real completeness principle.

Proof. Start with the NameCert obligations of Definition 4112.2. A completion-facing read must first inspect the sequence index and finite StreamName window rows.

The limit request then passes through the $\mathsf{LimitSpace}^\uparrow$ row and the $\mathsf{Topology}^\uparrow$ open-test row. If the consumer is a completion or Cauchy consumer, the same route displays the $\mathsf{CauchySpace}^\uparrow$ comparison row before any $\mathsf{Real}^\uparrow$ endpoint row is read.

Transport, replay, provenance, and naming preserve this order. The excluded principles are not carrier coordinates, so no consumer can extract them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$.
Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$, seed)
Scope closed: Definition 4112.1, Definition 4112.2, and Definition 4112.3 seed the finite sequential-convergence interface over StreamName, LimitSpace, Topology, CauchySpace, and Real rows.
Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$, unformalized)
Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient point equality, selected subsequences, countable-choice selectors, host topological closure, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-index exactness, StreamName window coverage, LimitSpace convergence routing, Topology open-test exposure, Cauchy comparison compatibility, Real endpoint nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialSpaceUp.

Constructive story: A SequentialSpace$^{\uparrow}$ packet is generated from sequence-index rows, StreamName windows, LimitSpace convergence rows, Topology open-test rows, CauchySpace comparison rows where relevant, Real endpoint rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3133

A Concrete Naming Certificate for MetaCICConversionEndpointSeal$^{\uparrow}$

The MetaCICConversionEndpointSeal$^{\uparrow}$ packet realizes the endpoint question isolated in chapter 9386. It sits after the bounded normal-conversion bridge of chapter 3439, the finite fuel trace of chapter 3889, and the typed normal-form decider of chapter 3714. The packet names the exact surface where bounded normal computation may be sealed as finished normal-form data for the closed typed terms under discussion, while leaving unrestricted closed beta-conversion outside the carrier.

Falsifiable prediction: Every accepted MetaCICConversionEndpointSeal$^{\uparrow}$ read exposes bounded-normal-conversion bridge, MetaCIC fuel trace, typed normal-form decider, finished endpoint, closed typed term, obstruction, transport, replay, provenance, and local NameCert rows before any conversion endpoint handoff is consumed.

Independence witness: bounded-normal-conversion-bridge, metacic-fuel-trace, typed-normal-form-decider, metacic-decidable-conversion-frontier: the bridge records bounded comparison with endpoint evidence, the fuel trace records bounded computation, the typed decider records typed comparison, and the frontier records the broader decidability boundary. None is bijective to the endpoint seal because this packet binds those rows into one finished-normal endpoint handoff.

Definition 3133.1 (MetaCIC conversion endpoint seal carrier)**.** *A* MetaCICConversionEndpointSeal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, F, D, T, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, recording bounded normal comparison under displayed endpoint evidence.*
2. *F is the* MetaCICFuelTrace$^{\uparrow}$ *row, recording finite fuel, input, bounded output, finished-normalization evidence or its explicit absence, and bounded readback.*
3. *D is the* TypedNormalFormDecider$^{\uparrow}$ *row, recording the typed checker surface for the compared closed terms.*
4. *T is the closed typed-term row, recording the context, terms, inferred type status, and closedness window admitted by the decider surface.*

5. *O is the obstruction row, excluding unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency.*
6. *H records componentwise* hsame *transport for B, F, D, T, O and the local naming row.*
7. *C records displayed* Cont *replay from a conversion endpoint request through B, F, D, T, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{TypedNormalFormDecider}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, F, D, T, O, H, C, P, N$. It has no coordinate for unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICConversionEndpointSealUp.MetaCICConversionEndpointSealTasteGate`
`single carrier alignment`

Theorem 3133.2 (MetaCIC conversion endpoint seal obligations)**.** *For every accepted* MetaCICConversionEndpoint *carrier*

$$E = (B, F, D, T, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}}$ *surface consists exactly of bounded bridge exposure through B, finite fuel-trace exposure through F, typed normal-form decider exposure through D, closed typed-term endpoint exposure through T, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4113.1. Rows B, F, D are the three sibling surfaces: bounded comparison, finite fuel trace, and typed normal-form decision.

The endpoint row T is read only after those sibling rows are displayed. It records the closed typed-term boundary under which bounded normal computation can be treated as a finished normal-form endpoint.

The obstruction row O remains part of the same packet. It blocks rereading the endpoint as unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Finally H, C, P, N transport, replay, source, and name exactly the same finite route. No carrier coordinate stores a stronger global metatheorem. □

Theorem 3133.3 (MetaCIC conversion endpoint seal handoff)**.** *Every endpoint-facing consumer of an accepted* $\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}$ *carrier factors through*

$$\mathsf{BoundedNormalConversionBridge}^{\uparrow},\ \mathsf{MetaCICFuelTrace}^{\uparrow},\ \mathsf{TypedNormalFormDecider}^{\uparrow}$$

before it may read the closed typed-term endpoint row. The handoff exports a finite endpoint seal for bounded normal computation, not an unrestricted conversion decider.

Proof. Use Definition 4113.2. The bounded bridge row supplies the comparison boundary, the fuel-trace row supplies the finite computation trace, and the typed decider row supplies the typing-facing comparison surface.

Only after these rows are present can the endpoint row T be consumed. This makes the endpoint a replayable finite seal rather than a theorem erasing the fuel and typed-side hypotheses.

The obstruction, transport, replay, provenance, and naming rows preserve the same boundary. Thus the packet gives the decidability frontier a concrete endpoint landing while refusing the stronger global conversion claims. □

The Lean TasteGate follow-up for this chapter must provide explicit Nontrivial MetaCICConversionEndpointSeal$^{\uparrow}$ and FieldFaithful MetaCICConversionEndpointSeal$^{\uparrow}$ instances preserving $B, F, D, T, O, H, C, P, N$ as separate displayed rows.

Closure status for NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$ ·

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$, seed)

Scope closed: Definition 4113.1, Definition 4113.2, and Definition 4113.3 seed the finite endpoint seal connecting bounded normal computation to finished normal-form data for closed typed terms.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded bridge exposure, fuel-trace exactness, typed normal-form decider compatibility, closed endpoint exposure, obstruction nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConversionEndpointSealUp.

Constructive story: A MetaCICConversionEndpointSeal$^{\uparrow}$ packet is generated from BoundedNormalConversionBridge, MetaCICFuelTrace, TypedNormalFormDecider, closed typed-term endpoint, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3134

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ packet records the finite comparison route between located metric completion, Bishop metric completion, constructive metric completion, and the general metric-completion comparison surface. It sits after chapter 2114, chapter 4023, chapter 1556, chapter 3553, chapter 301, chapter 126, chapter 291, and chapter 38. The chapter names one located completion route through displayed finite rows; it does not assert quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3134.1 (Bishop located-completion comparison carrier)**.** *A* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, B, Q, M, R, S, D, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *route, carrying the located source and the displayed completion-facing windows.*
2. *B is the* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *route, carrying the Bishop regular Cauchy presentation of the same located source.*
3. *Q is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route, carrying the constructive metric completion packet over located metrics, regular Cauchy names, dyadic tolerances, and separated comparison.*
4. *M is the* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *row comparing the visible completion presentations through paired dense embeddings, a common dense source, and finite round-trip reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row shared by the Bishop and constructive routes.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected before any regular rational readback or comparison map is named.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the located, Bishop, constructive, and metric-comparison windows at finite precision.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, B, Q, M, R, S, D have been displayed.*
9. *H records componentwise* hsame *transport for the four completion routes, readback, window, tolerance, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a located completion request through the Bishop and constructive presentations, the metric-comparison row, the shared readback rows, and then the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, countable choice, host equality of completion classes, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionComparisonUp.BishopLocatedCompletionComparisonTasteGate`
`single carrier alignment`

Theorem 3134.2 (BishopLocatedCompletionComparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ *carrier* $K = (L, B, Q, M, R, S, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located completion route, Bishop metric completion route, constructive metric completion route, metric-completion comparison row, regular rational readback, stream windows, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming row;*

2. *located route comparison, requiring L, B, Q, M to share the same displayed located source before any completion-facing comparison is exported;*

3. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback compatibility, requiring S to be read before R and requiring both rows to be replayed through C before the comparison row M reaches the seal;*

4. *Real seal handoff, requiring A to remain downstream of L, B, Q, M, R, S, D;*

5. *ledger refusal for quotient completions, selected representatives, countable choice, host equality of completion classes, and ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

No obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4114.1. The projection exposes exactly the located, Bishop, constructive, and metric-comparison routes, together with the shared regular readback, stream-window, dyadic-tolerance, Real-seal, transport, replay, provenance, and naming rows.

Use the located route as the common source. The rows L, B, Q, M can be compared only after the located metric completion route has been displayed, so the Bishop and constructive presentations are read as finite routes over the same located completion request.

Next read the shared observation rows. The stream row S supplies the inspected finite windows, R reads them as regular rational data, and D records the tolerance ledger used by the comparison row. Replay by C preserves this order before the seal row A can be consumed.

Finally attach H, P, N. Componentwise transport, provenance, and local naming preserve the same finite route and add no coordinate for quotient completions, selected representatives, countable choice, host equality, or ambient Real completeness. Hence the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: A $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ consumer that reaches a comparison map or $\mathsf{Real}^{\uparrow}$ seal while omitting the located completion row, Bishop completion row,

constructive completion row, metric-comparison row, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: located-metric-completion, bishop-metric-completion, constructive-metric-completion, metric-completion-comparison, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole comparison packet because each contributes only a completion presentation, comparison surface, readback row, window row, tolerance ledger, or seal, while this chapter binds them into one located completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ packet is generated from LocatedMetricCompletion, BishopMetricCompletion, ConstructiveMetricCompletion, MetricCompletionComparison, RegSeqRat, StreamName, DyadicRatCore, Real, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4114.1 and Definition 4114.2 seed the finite located completion comparison route over Bishop, constructive, metric-comparison, readback, tolerance, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located route comparison, RegSeqRat and StreamName readback compatibility, Real seal handoff, transport replay, provenance non-escape, and ledger refusals for BEDC.Derived.BishopLocatedCompletionComparisonUp.

Chapter 3135

A Concrete Naming Certificate for $\mathsf{RiemannSumGauge}^{\uparrow}$

Following Bishop and Bridges on Riemann integration, the $\mathsf{RiemannSumGauge}^{\uparrow}$ packet records the finite tagged-gauge data visible before an integration consumer reads a Riemann-sum or Darboux-sum handoff. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the upper Darboux-sum boundary of chapter 4083, and the closed bounded interval Darboux-partition surface of chapter 3429. The packet names a finite gauge-and-tag carrier for later $\mathsf{Calculus}^{\uparrow}$ and $\mathsf{Integral}^{\uparrow}$ consumers; it does not import host real functions, selected tags, quotient equality of partitions, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3135.1 (Riemann sum gauge carrier)**.** *A* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, T, Q, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded* $\mathsf{Real}^{\uparrow}$ *interval row, including endpoint seals and the finite interval enclosure visible to the consumer.*
2. *T is the tagged finite partition row, recording cells, tags, endpoint reads, and the local tag-in-cell classifier.*
3. *Q is the positive rational gauge-radius row, read as displayed rational data rather than as an ambient open cover.*
4. *M is the mesh ledger, binding the tagged partition to the gauge-radius row and recording the visible mesh comparisons.*
5. *D is the Darboux/Riemann sum row, carrying the finite weighted values and the handoff to* **??** *and the displayed Darboux-partition surface.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint, tag, mesh, and value observations at the requested finite windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and rational gauge data.*
8. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every real-facing value or interval read.*

9. *H records componentwise* hsame *transport for interval, tagged partition, gauge, mesh, sum, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from gauge requests, tagged-sum reads, Darboux handoffs, or real-facing consumers through $I, T, Q, M, D, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, ***??***, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed finite tagged-gauge Riemann-sum history. It has no coordinate for a host function space, selected open cover, quotient equality of partitions, chosen limiting mesh, choice principle, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3135.2 (Riemann sum gauge NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier $G = (I, T, Q, M, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded interval, tagged finite partition, positive rational gauge-radius row, mesh ledger, Darboux/Riemann sum row, stream windows, regular rational readback, dyadic tolerance and Real seal row, transport, replay, provenance, and local name;*
2. *classifier transport, requiring tags, cells, endpoint observations, and gauge-radius comparisons to move only by the componentwise* hsame *row H;*
3. *mesh-gauge exactness, requiring M to bind the tagged partition T to the positive rational gauge-radius row Q before any tagged-sum consumer reads the mesh;*
4. *Darboux-sum handoff, requiring the finite sum row D to expose only the displayed* ***??*** *and* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *routes;*
5. *Real seal nonescape, requiring every Real-facing value, endpoint, or interval read to pass through W, R, E after I, T, Q, M, D have been displayed;*
6. *transport and replay, requiring H and C to preserve the same finite tagged-gauge history;*
7. *provenance and naming, requiring P and N to package and name exactly the displayed rows;*
8. *ledger refusal, excluding host real functions, quotient equality of partitions, selected limiting meshes, choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Riemann-sum gauge obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RiemannSumGaugeUp.RiemannSumGaugeTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 4115.1. The projection exposes exactly the interval, tagged partition, gauge-radius, mesh, finite sum, window, readback, seal, and structural rows.

The mesh-gauge route is $T, Q \mapsto M \mapsto D$. A tag or cell may be consumed by the sum row only after the mesh ledger has related that tagged partition to the displayed positive rational gauge-radius row.

The Darboux handoff is then read from D. Since D only names the finite **??** and ClosedBoundedIntervalDarbouxP routes, it cannot add a host supremum, hidden function equality, or selected limiting mesh.

Finally, W, R, E supply finite window, regular rational readback, dyadic tolerance, and Real seal data, while H, C, P, N transport, replay, package, and name the same history. The excluded host objects have no carrier coordinate, so the listed obligations exhaust the local NameCert surface. □

Theorem 3135.3 (Riemann sum gauge tagged-real-partition source lock)**.** *For every accepted* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier* $G = (I, T, Q, M, D, W, R, E, H, C, P, N)$*, the mesh ledger* M *is readable only after the tagged partition row* T *supplied by chapter 4296 has displayed its cells, tags, endpoint reads, and local tag-in-cell classifier. Every Darboux or gauge-integral consumer therefore receives the mesh route*

$$T, Q \longmapsto M \longmapsto D \longmapsto W, R, E, H, C, P, N$$

and cannot replace it with host partitions, selected mesh limits, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4115.1. The tagged partition coordinate T is listed before the mesh ledger M, and M is the first row that binds the tagged cells to the positive rational gauge-radius row Q.

Apply Definition 4115.2. Its mesh-gauge exactness clause requires M to bind T and Q before any tagged-sum consumer reads the mesh.

The tagged partition source itself is the carrier of chapter 4296, where cells, tags, endpoint observations, containment rows, window reads, and mesh exactness are displayed as finite rows. Reading M without that source row is therefore outside the accepted $\mathsf{RiemannSumGauge}^{\uparrow}$ packet.

Finally the Darboux/Riemann sum row D and the structural rows W, R, E, H, C, P, N preserve the same route. None of these rows adds host partitions, selected mesh limits, quotient function equality, or ambient Real completeness, so every downstream consumer remains locked to the tagged-real-partition source. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSumGauge}^{\uparrow}$ packet is generated from a closed bounded RealUp interval row, tagged finite partition row, positive rational gauge-radius row, mesh ledger, Darboux/Riemann sum row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$, seed)

Scope closed: Definition 4115.1 and Definition 4115.2 seed the finite tagged-gauge Riemann-sum surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, **??**, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real functions, quotient equality of partitions, selected limiting meshes, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, mesh-gauge exactness, Darboux-sum handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumGaugeUp.

Chapter 3136

A Concrete Naming Certificate for $\mathsf{RegularCauchyTransform}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTransform}^{\uparrow}$ packet records the finite transform route between regular Cauchy names without committing to a map-specific analytic hypothesis. It sits over the regular rational readback of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the explicit modulus surface of chapter 3768, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a displayed transform certificate; it is not a quotient stream operation, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.

Definition 3136.1 (Regular Cauchy transform carrier)**.** *A* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, M, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *finite-window row.*
2. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row attached to those windows.*
3. *M is the transform modulus row, assigning each requested output tolerance to a finite source-window obligation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested output radius, source radius, and slack comparisons used by M.*
5. *U is the output* $\mathsf{StreamName}^{\uparrow}$ *finite-window row produced by the displayed transform route.*
6. *Q is the output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to U and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the output regular readback row has been displayed.*
8. *H records componentwise* hsame *transport for the source windows, source readback, transform modulus, dyadic ledger, output windows, output readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$S, R, M, D \longmapsto U, Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTransform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected Cauchy representative, host function extensionality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached output seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTransformUp.RegularCauchyTransformTasteGate single carrier alignment`

Definition 3136.2 (Regular Cauchy transform classifier)**.** *The* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *classifier compares accepted carriers only through componentwise* hsame *transport of*

$$S, R, M, D, U, Q, E, H, C, P, N.$$

Thus source windows, source regular readback, transform modulus, dyadic tolerance accounting, output windows, output regular readback, terminal Real seal, replay, provenance, and local naming are all visible comparison data. No classifier clause compares hidden host functions, quotient streams, selected limits, or unrecorded completion witnesses.

Theorem 3136.3 (Regular Cauchy transform NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier*

$$T = (S, R, M, D, U, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, source regular readback, transform-modulus routing, dyadic tolerance accounting, output-window production, output regular readback, terminal Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming through the displayed rows* $S, R, M, D, U, Q, E, H, C, P, N$.

Proof. Project the carrier of Definition 4116.1. The source side is visible as S, R, the tolerance route is visible as M, D, the output side is visible as U, Q, E, and the structural rows are visible as H, C, P, N.

Read the transform route in displayed order. A consumer first opens finite source windows, attaches them to the regular rational readback, reads the transform modulus and dyadic tolerance ledger, and only then obtains output windows and output regular readback.

Finally attach the terminal and structural rows. The Real seal E is downstream of Q, while H, C, P, N transport, replay, source, and name the same finite route. Because the carrier has no hidden host map, quotient stream equality, selected representative, choice schedule, or ambient completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 3136.4 (Regular Cauchy transform modulus route)**.** *Every output window admitted by a* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *packet is backed by the transform modulus row M, the dyadic tolerance ledger D, a finite source* $\mathsf{StreamName}^{\uparrow}$ *window S, and source* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R. Hence a Real or Completion consumer cannot read the output regular name through U, Q without first displaying the modulus-controlled source route.*

Proof. Begin with an admitted output window. Carrier projection exposes U as an output $\mathsf{StreamName}^{\uparrow}$ row, but U is downstream of the transform route rather than an independent stream source.

Trace the request through M and D. The modulus row names the source-window obligation for the requested output tolerance, and the dyadic ledger records the radii and slack comparisons needed to serve that obligation.

The source evidence is then S, R. The finite windows in S are attached to regular rational readback by R, so the output route cannot be justified by a selected limit or host convergence witness.

After U is produced, the output readback Q and terminal seal E remain downstream of the same route. The structural rows H, C, P, N preserve this order under transport and replay. □

Theorem 3136.5 (Regular Cauchy transform Real-seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *read accepted by a* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{C}$$

No accepted read bypasses the source regular name, transform modulus, dyadic tolerance ledger, output regular readback, componentwise transport, replay, provenance, or local naming rows.

Proof. Start with the terminal seal row E. In the carrier definition, E is reachable only after the output readback row Q, and Q is attached to the output finite-window row U.

Use the modulus-route theorem. The output window U is produced only after the source windows S, source readback R, transform modulus M, and dyadic ledger D have been displayed.

Finally read the structural rows. The hsame transport, Cont replay, Pkg provenance, and local NameCert preserve the same finite chain; they do not add a quotient stream, selected host representative, ambient completeness theorem, or second seal route. □

Falsifiable prediction: A $\mathsf{RegularCauchyTransform}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying source windows, source RegSeqRat readback, a transform modulus row, dyadic tolerance accounting, output windows, output RegSeqRat readback, transport, replay, provenance, and local naming is not reading this packet.

Independence witness: regseqrat, streamname, dyadicratcore, regular-modulus, real: no listed sibling supplies the whole transform packet because each contributes only readback, windowing, tolerance, modulus, or sealing data while this chapter binds the finite source-to-output route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTransform}^{\uparrow}$ packet is generated from source StreamName windows, source RegSeqRat readback, a transform modulus row, DyadicRatCore tolerance accounting, output StreamName windows, output RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4116.1, Definition 4116.2, Definition 4116.3, Definition 4116.4, and Definition 4116.5 seed the finite transform route between regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, classifier transport, modulus routing, output-window production, Real-seal handoff, and ledger non-escape for BEDC.Derived.RegularCauchyTransformUp.

Chapter 3137

A Concrete Naming Certificate for $\mathsf{PoincareDiskDynamics}^{\uparrow}$

The $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet realizes the disk-dynamics fixed-end reading of section 9463.5 as finite BEDC data. It consumes the phase-transport route of chapter 3455, the metric surface of chapter 3474, disk coordinates from chapter 377, finite evolution rows from chapter 195, and boundary-circle consumers from chapter 58. Its distinguishing carrier binds disk-map evolution to a preserved far-end boundary row; it is not a completed hyperbolic plane, physical cosmology, global clock, quotient boundary, or internal object named T.

Definition 3137.1 (Poincare disk dynamics carrier)**.** *A* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, \Phi, G, B, F, V, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the inspected disk points.*
2. *Φ is the disk-map row, recording the finite boundary-preserving transport applied to those points.*
3. *G is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row used only as displayed four-point or distance-comparison data.*
4. *B is the* $\mathsf{S}^{1\uparrow}$ *boundary-consumer row naming the fixed far-end commitment.*
5. *F is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *provenance row linking visible phase movement, disk coordinates, reversible transport, fiber provenance, and refusal data.*
6. *V is the* $\mathsf{DynSystem}^{\uparrow}$ *finite-evolution row for replaying the disk-map orbit.*
7. *A is the audit-refusal row excluding an interior fixed point requirement, host hyperbolic geometry, physical light-speed derivation, global observer time, subject-substance, and an internal* T *object.*
8. *H records componentwise* hsame *transport for the disk, map, metric, boundary, phase-provenance, dynamics, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through disk coordinates, disk-map evolution, metric readback, boundary preservation, phase provenance, and refusal.*
10. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskDynamics}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$ *row.*

The carrier compares packets only through these rows. It has no coordinate for a host Poincare disk model, quotient ideal boundary, global physical evolution law, completed metric geometry, or object-level T*.*

Lean4 ✓ `BEDC.Derived.PoincareDiskDynamicsUp.PoincareDiskDynamicsTasteGate single carrier alignment`

Theorem 3137.2 (Poincare disk dynamics NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *carrier*

$$D = (U, \Phi, G, B, F, V, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, disk-map exposure, hyperbolic metric readback, fixed far-end boundary preservation, phase-transport provenance, finite dynamic replay, refusal exactness, componentwise transport, displayed continuation, package provenance, and local naming through the rows* $U, \Phi, G, B, F, V, A, H, C, P, N$*.*

Proof. Project the carrier of Definition 4117.1. The visible geometric data are the disk coordinates U, disk-map row Φ, metric-read row G, boundary row B, phase-provenance row F, and finite-evolution row V.

Read the fixed-end condition next. The boundary row B is preserved by the displayed disk-map route, while F records the phase-transport provenance that keeps visible movement distinct from source collapse or an internal T object.

The dynamic replay is finite. The row V supplies only the inspected orbit cells, and C replays those cells together with the disk, map, metric, boundary, and refusal rows.

Finally attach H, P, N. Componentwise transport, provenance, and local naming preserve exactly the displayed rows, so the NameCert surface cannot export host hyperbolic geometry, a quotient boundary, global time, physical constants, or subject-substance. □

Falsifiable prediction: If a disk-dynamics consumer preserves visible disk coordinates but drops either the boundary row, the phase-transport provenance row, or the refusal row, it is not reading a $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, unitdisk, dynsystem, s1: no listed sibling is bijective to $\mathsf{PoincareDiskDynamics}^{\uparrow}$ because this packet jointly carries disk coordinates, disk-map evolution, metric readback, fixed far-end boundary preservation, phase provenance, dynamic replay, refusal, transport, provenance, and local naming.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial}\ \mathsf{PoincareDiskDynamics}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PoincareDiskDynamics}^{\uparrow}$ TasteGate instances. Those instances must keep $U, \Phi, G, B, F, V, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet is generated from UnitDisk coordinates, a disk-map row, HyperbolicMetric readback, SOne boundary consumption, PoincareDiskPhaseTransport provenance, DynSystem finite evolution, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4117.1 and Definition 4117.2 seed the finite disk-map, metric, fixed-boundary, phase-provenance, and dynamic-replay route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Poincare disk geometry, host hyperbolic metric models, quotient boundary equality, global observer time, physical light-speed derivation, internal T, subject-substance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-map exposure, metric readback, fixed-boundary preservation, phase-provenance routing, dynamic replay, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.PoincareDiskDynamicsUp.

Chapter 3138

A Concrete Naming Certificate for $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$

Following the finite minimum step in the Heine–Cantor proof of chapter 9237, the $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ packet records the compact-net lower-bound fold as a finite BEDC carrier. It reads the $\mathsf{ProbeBundle}$ spine supplied by compactness, the $\mathsf{Rat}^{\uparrow}$ radius histories attached to bundled centers, the order readback exposed through $\mathsf{TotalOrder}^{\uparrow}$, the source rows of $\mathsf{Compact}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$, the pointwise graph rows of $\mathsf{ContinuousMap}^{\uparrow}$, and the downstream $\mathsf{UniformContinuityModulus}^{\uparrow}$ consumer. The packet names the fold as finite induction over displayed probes, not as a host minimum over a semantic finite set.

The fold is the finite lower-bound row consumed by chapter 388 and by the uniform-modulus handoff, while remaining only a $\mathsf{ProbeBundle}$ and $\mathsf{Rat}^{\uparrow}$ finite induction surface.

Definition 3138.1 (ProbeBundleFiniteMinimumFold carrier)**.** *A* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, G, \mathcal{P}, R, O, A, B, U, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row carrying the finite-net witness.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source-distance row used by the compact-uniform-continuity argument.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the displayed graph.*
4. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine indexing the compact-net centers.*
5. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *radius histories, one for each bundled probe.*
6. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *readback row used to compare the positive rational radius histories.*
7. *A is the accumulator ledger produced by primitive recursion over $\mathcal{P}$.*
8. *B is the certified lower-bound row saying that the final accumulator is below every displayed radius history in R.*
9. *U is the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *handoff row that receives the accumulator as the tolerance-only precision.*
10. *H records componentwise* hsame *transport for the compact source, metric source, continuous-map row, bundle spine, rational candidates, order readback, accumulator, lower-bound certificate, handoff, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the finite fold from the bundle spine to the handoff row.*
12. *Q records* Pkg *provenance over* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{UniformContinuityModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$ *row.*

The carrier contains no host finite minimum, escaped probe outside $\mathcal{P}$, open-cover compactness object, quotient point equality, external modulus selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.ProbeBundleFiniteMinimumFoldUp.ProbeBundleFiniteMinimumFoldTasteGate`
`single carrier alignment`

Theorem 3138.2 (ProbeBundleFiniteMinimumFold NameCert obligations)**.** *For every accepted* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ *carrier*

$$F = (K, M, G, \mathcal{P}, R, O, A, B, U, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *compact, metric, and continuous-map source admission through K, M, G;*
2. *nonempty* ProbeBundle *spine admission for $\mathcal{P}$;*
3. *positive* $\mathsf{Rat}^{\uparrow}$ *radius admission for each bundled row of R;*
4. $\mathsf{TotalOrder}^{\uparrow}$ *comparison readback through O at each successor step of the fold;*
5. *finite-spine induction producing the accumulator row A from the rows of R;*
6. *lower-bound exactness, requiring B to certify that the accumulator is below every displayed radius history and to read no radius outside R;*
7. $\mathsf{UniformContinuityModulus}^{\uparrow}$ *handoff through U;*
8. *ledger non-escape, excluding host minima, escaped probes, open-cover compactness, quotient point equality, external modulus selectors, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 4118.1. The compact, metric, and continuous-map rows expose the source of the compact-uniform-continuity argument, while $\mathcal{P}$ is the only induction spine available to the fold.

Use primitive recursion over $\mathcal{P}$. At the initial row the accumulator reads the first positive $\mathsf{Rat}^{\uparrow}$ radius history in R; at each successor step the fold compares the accumulator with the next radius through the displayed $\mathsf{TotalOrder}^{\uparrow}$ row O and writes the retained lower row into A.

The lower-bound certificate B is attached to the same recursion. It records that the final accumulator is classifier-stably below every radius row already inspected in R, and it has no coordinate from which a radius outside the bundle can be read.

Finally the handoff row U gives the compact-uniform-continuity consumer a tolerance-only precision history. The structural rows H, C, Q, N transport, replay, source, and name the same finite route, so no host finite minimum, escaped probe, open-cover compactness object, quotient point equality, external modulus selector, or ambient real-completeness object can enter the certificate. □

Falsifiable prediction: A $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ consumer that obtains a UniformContinuityModulusUp precision without reading the displayed CompactUp source, MetricUp source, ContinuousMapUp row, ProbeBundle spine, RatUp radius histories, TotalOrderUp comparison row, accumulator ledger, lower-bound certificate, hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: compact, metric, continuousmap, rat, totalorder, uniformcontinuitymodulus: no listed sibling supplies the whole ProbeBundle finite minimum fold because each sibling contributes only compact source data, metric rows, graph rows, rational order data, or modulus handoff rows while this chapter binds the finite-spine induction surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$.

Constructive story: A $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ packet is generated from CompactUp finite-net rows, MetricUp source rows, ContinuousMapUp pointwise-modulus rows, a nonempty ProbeBundle spine, positive RatUp radius histories, a TotalOrderUp comparison readback row, an accumulator ledger, a lower-bound certificate, a UniformContinuityModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4118.1 and Definition 4118.2 seed the finite ProbeBundle minimum fold required by the compact-uniform-continuity lower-bound route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host finite minima, probes outside the displayed bundle, open-cover compactness, quotient point equality, external modulus selectors, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove source admission, bundle spine admission, positive rational admission, order readback, finite-spine induction, lower-bound exactness, modulus handoff, and ledger non-escape.

Chapter 3139

A Concrete Naming Certificate for $\mathsf{HausdorffDimension}^{\uparrow}$

Following Hausdorff and Falconer, the $\mathsf{HausdorffDimension}^{\uparrow}$ packet records the BEDC surface at which gauge-weighted finite and countable-looking cover data become a named scale-cover dimension read. It sits beside the scale counting route of chapter 3877, the finite refinement route of chapter 1853, the compact source boundary of chapter 129, and the dense Real-facing route of chapter 1809. The chapter names a gauge-cover packet; it does not overload $\mathsf{BoxDimension}^{\uparrow}$ with Hausdorff-measure obligations and does not assert a host measure theorem.

Definition 3139.1 (Hausdorff dimension gauge-cover carrier)**.** *A* $\mathsf{HausdorffDimension}^{\uparrow}$ *carrier over a compact metric source* X *is a finite* BHist *packet*

$$H = (M, K, G, C, W, R, S, D, T, P, N)$$

with the following displayed rows:

1. M *is the inherited* $\mathsf{Met}^{\uparrow}$ *row on* X*, including the visible distance classifier used to read ball diameters.*
2. K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite-net and complete-limit data before any gauge cover is inspected.*
3. G *is the gauge row. It records the dyadic scale exponent, positive radius bound, and rational weight expression used by the Hausdorff gauge read.*
4. C *is the finite visible cover row. Each cover member carries a center history, a radius history, membership evidence, and a diameter bound readable through* M *and* K*.*
5. W *is the gauge-weight ledger. It assigns to each displayed cover member its rational gauge contribution and records the finite partial sum.*
6. R *is the refinement route. It records how a smaller visible gauge cover refines the previous one without choosing a host-minimal cover.*
7. S *is the separability route through* $\mathsf{RealSeparability}^{\uparrow}$*, used when the compact metric source is Real-facing and cover centers must be read from dense windows.*
8. D *is the dimension-threshold ledger. It records rational lower and upper threshold comparisons for gauge-weight decay across displayed scales.*

9. *T stores componentwise* hsame *transport and displayed* Cont *replay for metric points, gauge scales, cover members, gauge weights, refinement rows, dense-route data, and threshold comparisons.*

10. *P is the* Pkg *provenance over* $\mathsf{HausdorffDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary outer measure, a selected optimal cover, countable choice over covers, host logarithms, quotient compacta, ambient Euclidean embeddings, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HausdorffDimensionUp.HausdorffDimensionTasteGate single carrier alignment`

Theorem 3139.2 (Hausdorff dimension NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffDimension}^{\uparrow}$ *carrier*

$$H = (M, K, G, C, W, R, S, D, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric and compact rows, gauge positivity and scale exposure, finite cover membership, diameter comparison, gauge-weight summation, refinement transport, dense Real-facing routing when present, threshold comparison, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, K, G, C, W, R, S, D, T, P, N$.*

Proof. Project the carrier of Definition 4119.1. The source rows M and K provide the metric and compact finite-net surface, while G supplies the positive gauge scale and the rational exponent data used by the dimension read.

For a fixed gauge, C displays the cover members and W displays their gauge contributions. The finite sum is therefore a packet row, not an invocation of an ambient outer-measure construction.

The row R is the only refinement route between gauges. It transports displayed covers to smaller displayed covers and keeps the weight ledger attached to the same finite source surface.

The dense-source route S, threshold ledger D, transport and replay row T, provenance row P, and naming row N preserve the same visible data. Since the carrier contains no coordinate for selected optimal covers, countable choice, host logarithms, outer measures, quotient compacta, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 3139.3 (Hausdorff dimension scale-cover route)**.** *Every scale-cover read exported by an accepted* $\mathsf{HausdorffDimension}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow} \longmapsto G, C, W, R \longmapsto S, D, T, P, N.$$

The route may be compared with $\mathsf{BoxDimension}^{\uparrow}$ *only at the shared metric, compact, dyadic-scale, finite-cover, refinement, counting, and dense-route rows, and with* $\mathsf{CoveringDimension}^{\uparrow}$ *only at the finite cover-refinement surface. It does not identify Hausdorff gauge weights with box cover counts, does not turn covering dimension into a gauge measure, and does not import open-cover compactness or selected subcovers.*

Proof. Start from the metric and compact source rows. A scale-cover consumer must first read M and K, so the cover is anchored in the same compact metric finite-net boundary used by chapter 129.

Open the gauge-cover rows. The gauge row G, visible cover row C, and gauge-weight ledger W provide the Hausdorff-specific data; the refinement row R records how these rows persist as the scale decreases.

Compare only along shared surfaces. The $\mathsf{BoxDimension}^{\uparrow}$ sibling shares finite scale-cover and counting language but lacks the gauge-weight ledger, while $\mathsf{CoveringDimension}^{\uparrow}$ shares finite refinement language but lacks the gauge-threshold comparison row.

Finally route Real-facing centers through S and close the threshold read through D, T, P, N. The packet therefore exports a concrete gauge-cover dimension route without adding host outer measures, selected cover families, open-cover compactness, quotient equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4119.1, Definition 4119.2, and Definition 4119.3 seed the gauge-cover dimension packet over compact metric sources, finite refinement rows, RealSeparability dense routes, and box-dimension comparison surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff measure existence, exact numerical dimension values, countable cover choice, selected optimal covers, host logarithm analysis, ambient Euclidean embeddings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge positivity, finite cover membership, diameter comparison, gauge-weight summation, refinement routing, threshold comparison, and RealSeparability dense-route coverage for BEDC.Derived.HausdorffDimensionUp.

Constructive story: A $\mathsf{HausdorffDimension}^{\uparrow}$ packet is generated from a metric row, compact metric source row, gauge row, visible cover row, gauge-weight ledger, refinement route, dense Real-facing route, threshold ledger, componentwise transport and replay, provenance, and a local NameCert row.

Chapter 3140

A Concrete Naming Certificate for $\mathsf{EffectiveLimitOperator}^{\uparrow}$

Following Weihrauch computable analysis, the $\mathsf{EffectiveLimitOperator}^{\uparrow}$ packet records the represented-space limit operation as a finite BEDC handoff from Cauchy names to a sealed represented output. It sits over represented-space source rows from chapter 1494, Cauchy representation rows from chapter 1680, stream-window observations from chapter 126, regular rational readback from chapter 301, dyadic tolerance control from chapter 291, and the terminal real-name seal of chapter 38. The packet makes the limit operator visible as displayed finite convergence data, not as an ambient completeness principle.

Definition 3140.1 (EffectiveLimitOperator carrier)**.** *An* $\mathsf{EffectiveLimitOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, S, M, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the represented-space source row, including the representation interface through which names are read.*
2. *A is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row presenting the input name as a Cauchy object over X.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of inspected name windows.*
4. *M is the effective convergence and modulus ledger assigning each requested tolerance to a finite tail-window obligation.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations made through the windows in S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested precision with the displayed tail budget.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the represented source, Cauchy representation, stream windows, modulus ledger, regular readback, and dyadic tolerance rows have been displayed.*
8. *H records componentwise* hsame *transport for the represented source, Cauchy source, stream windows, modulus ledger, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through A, S, M, R, D to the sealed output row E.*

10. *P records* Pkg *provenance over* EffectiveLimitOperator$^{\uparrow}$, RepresentedSpace$^{\uparrow}$, CauchyRepresentation$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{EffectiveLimitOperator}^{\uparrow}}$ *row.*

The carrier contains no selected limit, quotient stream equality, countable-choice selector, host real equality, or ambient completeness row for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.EffectiveLimitOperatorUp.EffectiveLimitOperatorTasteGate single carrier alignment`

Theorem 3140.2 (EffectiveLimitOperator NameCert obligations)**.** *For every accepted* EffectiveLimitOperator$^{\uparrow}$ *carrier*

$$L = (X, A, S, M, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{EffectiveLimitOperator}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *represented-source admission for the displayed* RepresentedSpace$^{\uparrow}$ *and* CauchyRepresentation$^{\uparrow}$ *rows;*
2. *stream-window coverage, requiring every tolerance request admitted by M to name a finite inspected window in S;*
3. *effective-modulus replay, requiring the convergence ledger M to route each requested tolerance to the displayed regular rational readback R;*
4. *dyadic tolerance exactness, requiring every comparison used by the limit handoff to read exactly the row D beside R;*
5. *real-seal handoff, requiring the* Real$^{\uparrow}$ *seal E to be consumed only after X, A, S, M, R, D have been displayed;*
6. *structural stability through the componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows;*
7. *ledger non-escape, excluding selected limits, quotient stream equality, countable choice, host real equality, and ambient* Real$^{\uparrow}$ *completeness from the packet.*

Proof. Project the carrier in Definition 4120.1. The rows X, A, S, M, R, D, E display the represented source, Cauchy presentation, stream windows, effective convergence ledger, regular rational readback, dyadic tolerance comparison, and terminal real seal.

A tolerance request first reads the ledger M. The ledger names a finite window in S, sends the observed entries through the regular readback row R, and compares the requested budget against the dyadic tolerance row D.

The seal row E is therefore reachable only after this displayed route has been followed. The transport, replay, provenance, and local naming rows H, C, P, N attach the same route to classifier-stable BEDC data.

Since the carrier has no coordinate for a selected limit, quotient stream equality, countable-choice selector, host real equality, or ambient completeness of Real$^{\uparrow}$, a consumer cannot obtain any of those objects from the local certificate surface. □

Falsifiable prediction: A EffectiveLimitOperator$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ limit seal without reading the displayed represented-space source, CauchyRepresentation source, StreamName

windows, effective modulus ledger, RegSeqRat readback, DyadicRatCore tolerance row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: representedspace, cauchyrepresentation, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole effective-limit-operator packet because each sibling contributes only source representation, Cauchy naming, windowing, readback, tolerance, or sealing data while this chapter binds the finite operator handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveLimitOperator}^{\uparrow}$ packet is generated from a RepresentedSpace source row, a CauchyRepresentation source row, StreamName windows, an effective convergence and modulus ledger, RegSeqRat regular rational readbacks, a DyadicRatCore tolerance row, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4120.1 and Definition 4120.2 seed the finite represented-space limit-operator handoff over Cauchy names, stream windows, regular readback, dyadic tolerance, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove represented-source admission, stream-window coverage, effective-modulus replay, dyadic tolerance exactness, Real seal handoff, structural stability, and ledger non-escape.

Chapter 3141

A Concrete Naming Certificate for LocatedCauchyFilterLimit$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley terminology of filters and filter bases, the LocatedCauchyFilterLimit$^{\uparrow}$ packet records the localized L10 limit route in which a located filter-basis row feeds the existing CauchyFilterLimit$^{\uparrow}$ and CauchyFilterRegularization$^{\uparrow}$ packets before any Real$^{\uparrow}$ seal is read. It sits after chapter 1558, chapter 4110, chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names the finite located completion-filter route, not quotient equality of filters, selected representatives, host real equality, countable choice, or ambient completeness.

Definition 3141.1 (LocatedCauchyFilterLimit carrier)**.** *A* LocatedCauchyFilterLimit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, G, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the located filter-basis row, naming the finite located neighborhoods and common-refinement data used by the completion-filter request.*
2. *F is the* CauchyFilterLimit$^{\uparrow}$ *row, consuming the displayed basis route before a limit-facing seal is read.*
3. *G is the* CauchyFilterRegularization$^{\uparrow}$ *row, regularizing the same filter route into finite stream, rational, dyadic, and seal data.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected by the located request.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for located comparison and limit-scale accounting.*
7. *A is the* Real$^{\uparrow}$ *seal row reached only after B, F, G, S, R, D have been replayed.*
8. *H records componentwise* hsame *transport for the located basis, limit, regularization, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a located completion-filter request through $B, F, G, S, R, D, A, H, P, N$.*
10. *P is the* Pkg *provenance over* LocatedCauchyFilterLimit$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, CauchyFilterRegularization$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limit representatives, host real equality, countable-choice locators, hidden cofinal subsequences, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.LocatedCauchyFilterLimitUp.LocatedCauchyFilterLimitUp`
Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterLimitUp.LocatedCauchyFilterLimitTasteGate single carrier alignment`

Theorem 3141.2 (LocatedCauchyFilterLimit NameCert obligations)**.** *For every accepted* LocatedCauchyFilterLimi *carrier*

$$L = (B, F, G, S, R, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located filter-basis row,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row,* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *row, finite stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, continuation, provenance, and local naming row;*
2. *located-basis exposure, requiring B to be displayed before the limit row F or the regularization row G can be consumed;*
3. *completion-filter compatibility, requiring F and G to read the same finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows before the* $\mathsf{Real}^{\uparrow}$ *seal is reached;*
4. *seal non-escape, requiring A to remain downstream of B, F, G, S, R, D;*
5. *ledger refusal for quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4121.1. The projection exposes the located basis row B, the limit row F, the regularization row G, the shared observation rows S, R, D, the seal row A, and the structural rows H, C, P, N.

The located row is read first. The continuation row C sends any completion-filter request through the displayed located neighborhoods and common refinements before the request can inspect the $\mathsf{CauchyFilterLimit}^{\uparrow}$ or $\mathsf{CauchyFilterRegularization}^{\uparrow}$ surfaces.

The shared observation rows bind the two sibling packets. Both the limit row and the regularization row consume the same finite stream windows, regular rational readbacks, and dyadic tolerance ledger before reaching A.

Finally attach transport, provenance, and local naming. The rows H, P, N preserve only the displayed route, so the carrier has no coordinate for quotient filters, selected representatives, hidden cofinal schedules, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: A $\mathsf{LocatedCauchyFilterLimit}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal while omitting the located filter-basis row, CauchyFilterLimit row, CauchyFilterRegularization row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: cauchyfilterlimit, cauchy-filter-regularization, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole located completion-filter packet because each contributes only a limit row, regularization row, window row, rational readback, tolerance ledger, or seal, while this chapter binds them to a located filter-basis route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyFilterLimit}^{\uparrow}$ packet is generated from a located filter-basis row, CauchyFilterLimit row, CauchyFilterRegularization row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4121.1 and Definition 4121.2 seed the finite located Cauchy-filter limit route over limit, regularization, stream, rational, tolerance, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-basis exposure, completion-filter compatibility, Real seal nonescape, transport replay, provenance nonescape, and ledger refusals for BEDC.Derived.LocatedCauchyFilterLimitUp.

This hub routes the pre-separated universal-property packet through chapter 1614, chapter 2237, chapter 1506, chapter 397, and chapter 3351.

Chapter 3142

A Concrete Naming Certificate for PseudometricCompletionUniversalProperty$^{\uparrow}$

Following Bishop and Bridges and Bourbaki–Kelley completion terminology, the PseudometricCompletionUniversalPr packet records the finite universal-property route for completing a pre-separated distance surface. It sits after the Pseudometric$^{\uparrow}$ carrier of chapter 1614, the separated classifier route of chapter 2004, the separated-completion handoff of chapter 2237, the metric-completion surface of chapter 1506, and the extension ledger of chapter 397. Its finite observations are read through chapter 126, chapter 301, chapter 291, and the terminal seal of chapter 38. The dense-source map row is the forward-facing consumer of chapter 3351, because extension existence and uniqueness must read dense embedding data before the completion-facing map row is compared. The chapter supplies the L10 bridge from pre-separated distance data to extension uniqueness without quotient completions, selected representatives, selected limits, host function equality, or countable choice.

Definition 3142.1 (PseudometricCompletionUniversalProperty carrier)**.** *A* PseudometricCompletionUniversalProp *carrier is a finite* BHist *packet*

$$U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$$

with the following displayed rows:

1. *P is the source* Pseudometric$^{\uparrow}$ *packet, carrying the point rows, displayed distance witness, zero-distance boundary, dyadic tolerance ledger, finite windows, regular rational readback, and terminal real distance seal.*
2. *Q is the* SeparatedQuotientMetric$^{\uparrow}$ *classifier row that consumes zero-distance evidence from P before any separated metric read is exposed.*
3. *S is the* SeparatedPseudometricCompletion$^{\uparrow}$ *handoff row that routes the source pseudometric through separated completion data while keeping the pre-separated source visible.*
4. *M is the* MetricCompletion$^{\uparrow}$ *row naming the completion-facing separated metric packet consumed by extension readers.*
5. *V is the* CompletionUniversalProperty$^{\uparrow}$ *row that supplies the finite extension-existence and uniqueness ledgers over completion data.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window schedule through which source points, distance approximants, and dense observations are inspected.*

7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to the dense source observations used by the extension route.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for zero-distance, triangle, and extension-precision comparisons before the real seal is read.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W, R, D display the requested approximation data.*
10. *F is the dense-source map row whose finite graph and modulus reads are to be extended from the pseudometric source.*
11. *E is the extension-existence ledger, a displayed route from $P, Q, S, M, V, W, R, D, A, F$ to a completion-facing map row.*
12. *L is the uniqueness ledger comparing two extension rows whenever they agree on the same separated classifier, dense-source windows, regular readback, dyadic tolerance, real seal, and dense map rows.*
13. *H records componentwise* hsame *transport for the source pseudometric, separated classifier, separated completion, metric completion, universal-property row, window schedule, regular readback, dyadic tolerance, real seal, map row, extension ledger, uniqueness ledger, continuation, provenance, and local naming rows.*
14. *C records displayed* Cont *replay of the route*
$$P, Q, S, M, V, W, R, D, A, F \longmapsto E \longmapsto L.$$
15. *N is both the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedQuotientMetric}^{\uparrow}$, $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows and the local* $\mathsf{NameCert}_{\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion classes, selected representatives, selected limits, host function equality, countable-choice schedules, trichotomy, or ambient completeness.

Theorem 3142.2 (PseudometricCompletionUniversalProperty NameCert obligations)**.** *For every accepted* $\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}$ *carrier $U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$, the local* $\mathsf{NameCert}_{\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source pseudometric, separated classifier, separated completion handoff, metric completion row, universal-property row, finite windows, regular readback, dyadic tolerance, real seal, dense map, extension ledger, uniqueness ledger, transport, replay, provenance, and local naming rows;*
2. *separated-source exposure, requiring zero-distance evidence from P to pass through Q and S before the metric-completion row M is read;*
3. *dense-readback exposure, requiring every extension request to display W, R, D, A before the dense map row F is repacked as an extension;*
4. *extension existence, requiring E to be read only after $P, Q, S, M, V, W, R, D, A, F$ have been displayed;*
5. *extension uniqueness, requiring L to compare candidate extension rows over the same separated source, dense windows, readback rows, real seal, and dense map data;*

6. *non-escape, excluding quotient completions, selected representatives, selected host limits, host function equality, countable choice, trichotomy, and ambient completeness from every exported consumer route.*

No pseudometric-completion universal-property obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4122.1. The accepted packet lists the source pseudometric, separated classifier, separated completion handoff, completion row, universal-property row, dense readback rows, extension and uniqueness ledgers, and structural support rows.

Separated-source exposure follows from the order P, Q, S, M. The source distance surface remains pre-separated in P until the zero-distance evidence is consumed by Q and the separated-completion handoff S.

Dense-readback exposure follows from W, R, D, A, F. The stream schedule, regular rational row, dyadic tolerance row, and real seal are displayed before the dense map row is used by an extension request.

Extension existence is the role of E, and uniqueness is the role of L. The replay row C places those ledgers after the same source, separated, completion, dense-readback, and map rows, while H transports only the displayed coordinates.

The non-escape clause is an inventory check on the carrier. Since no quotient completion class, selected representative, selected host limit, host function equality, countable-choice schedule, trichotomy row, or ambient completeness theorem appears among the coordinates, none can be exported by the local naming certificate. □

Theorem 3142.3 (PseudometricCompletionUniversalProperty extension uniqueness)**.** *Let* $U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$ *and* $U' = (P, Q, S, M, V, W, R, D, A, F, E', L', H', C', N')$ *be accepted* $\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}$ *carriers with the same source pseudometric, separated classifier, separated completion handoff, metric completion row, universal-property row, finite windows, regular readback, dyadic tolerance, real seal, dense map row, and package provenance. If* E *and* E' *agree on every displayed finite read from* W, R, D, A *after the separated-source route* P, Q, S *has been consumed, then the two completion-facing extension rows are compared by the uniqueness ledgers* L *and* L'*.*

The comparison is a displayed hsame *ledger over the finite carrier rows. It is not host equality of maps, not equality of quotient-completion representatives, and not a selected-limit principle.*

Proof. Read the common source coordinates from the two accepted packets. The rows P, Q, S fix the pre-separated distance surface, the separated classifier, and the separated-completion handoff available to both extensions.

Next read the common completion and dense-source coordinates. The rows M, V, W, R, D, A, F fix the completion-facing target, the universal-property ledger, the finite window schedule, regular rational readback, dyadic tolerances, real seal, and dense map data.

The hypothesis compares E and E' on the displayed finite reads from W, R, D, A after the separated-source route has been consumed. Transport those reads through H, H' and replay them through C, C' so both extension rows are presented over the same source, completion, dense-readback, real-seal, and dense-map coordinates.

The uniqueness coordinates are exactly L and L'. Applying them gives the required displayed comparison of extension rows, with no appeal to host function equality, quotient representatives, selected limits, or countable choice. □

Theorem 3142.4 (PseudometricCompletionUniversalProperty non-escape)**.** *Every consumer read exported by an accepted* PseudometricCompletionUniversalProperty$^{\uparrow}$ *carrier factors through the finite route*

Pseudometric$^{\uparrow}$, SeparatedQuotientMetric$^{\uparrow}$, SeparatedPseudometricCompletion$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CompletionUnivers

The packet exposes extension existence through E and extension uniqueness through L only after that route has been displayed. It exports no quotient completion class, selected representative, selected host limit, countable-choice schedule, host function equality, trichotomy decision, or ambient completeness theorem.

Proof. Project the carrier of Definition 4122.1. Its source and completion rows are P, Q, S, M, V, and its dense-readback rows are W, R, D, A.

Apply Definition 4122.2. The separated-source obligation forces a consumer through P, Q, S before M is read, and the dense-readback obligation forces a consumer through W, R, D, A before F, E, L are read.

Extension existence is then confined to the ledger E, while extension uniqueness is confined to L. The transport and replay rows H, C move only the displayed coordinates and preserve the same order.

The exclusions follow from the carrier inventory. Since the packet contains no coordinate for quotient completion classes, selected representatives, selected host limits, countable-choice schedules, host function equality, trichotomy, or ambient completeness, none of those objects can escape along a consumer route. □

Theorem 3142.5 (Pseudometric completion sibling route)**.** *Any accepted* PseudometricCompletionUniversalProperty *packet that exposes a universal readback must first route through* MetricCompletion$^{\uparrow}$, CauchyEquivalenceSetoid$^{\uparrow}$, *and* SeparatedCompletion$^{\uparrow}$. *The universal readback first consumes the Cauchy-equivalence setoid route of chapter 1820, then the metric completion route of chapter 1506, before extension existence or uniqueness can be read. The readback may then consume the displayed extension and uniqueness ledgers of the carrier, but it may not replace those sibling rows by quotient representatives, selected limits, host function equality, countable-choice schedules, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 4122.1. The rows $P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N$ already place the source pseudometric before the separated classifier, separated-completion handoff, metric-completion row, and universal-property ledger.

Read the equality-facing route next. The CauchyEquivalenceSetoid$^{\uparrow}$ comparison is the visible setoid boundary for regular Cauchy data, and the sibling route of chapter 1820 must be consumed before the metric-completion route of chapter 1506. The SeparatedCompletion$^{\uparrow}$ row records the separated completion route before the metric-completion consumer reads M.

Finally apply the non-escape theorem, Definition 4122.4. Since the carrier exports only extension and uniqueness ledgers after those displayed sibling rows, a universal readback cannot be retyped as quotient representatives, selected limits, host map equality, countable choice, trichotomy, or ambient completeness. □

Closure status for NameCert$_{\text{PseudometricCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A PseudometricCompletionUniversalProperty$^{\uparrow}$ packet is generated from a source Pseudometric$^{\uparrow}$ row, a SeparatedQuotientMetric$^{\uparrow}$ classifier, a SeparatedPseudometricCompletion$^{\uparrow}$ handoff, a MetricCompletion$^{\uparrow}$ completion row, a CompletionUniversalProperty$^{\uparrow}$ extension surface, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance rows, a Real$^{\uparrow}$ seal, a dense map row, extension and uniqueness ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PseudometricCompletionUniversalProperty}\uparrow}$, seed)

Scope closed: Definition 4122.1, Definition 4122.2, Definition 4122.3, Definition 4122.4, and Definition 4122.5 seed the finite universal-property route from pre-separated distance data to extension uniqueness through the sibling completion rows.

Formal verification: FormalStatus(NameCert$_{\text{PseudometricCompletionUniversalProperty}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected representatives, selected host limits, countable choice, host function equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-source exposure, dense-readback exposure, extension existence, extension uniqueness, and non-escape for BEDC.Derived.PseudometricCompletionUniversalPropertyUp.

Chapter 3143

A Concrete Naming Certificate for $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ packet records the finite choice-free boundary that Real and Completion consumers may use before any dense-open or Baire-style argument reaches a sealed real name. It sits between the dense-open packet of chapter 1542, the complete metric source of chapter 128, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Specker obstruction surface of chapter 1406. The packet exposes a displayed finite thread through those rows; it does not assert a host Baire theorem, countable choice, quotient stream equality, selected limits, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 3143.1 (BaireCategoryChoiceFree carrier)**.** *A* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, M, W, R, S, K, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open boundary row, read only through the finite schedule and nested-ball thread already named in chapter 1542.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row supplying the displayed complete-metric carrier, ball classifier, Cauchy-thread route, and limit-witness boundary.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected dense-open requests, metric balls, and terminal real-name observations before any consumer reads the boundary.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row converting the displayed windows into regular-rational tolerance data.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, M, W, R have been displayed.*
6. *K is the Specker boundary row. It records that nonconstructive selection or completeness requests are refused by comparison with the finite Specker-style obstruction surface, rather than discharged by countable choice.*
7. *H records componentwise* hsame *transport for B, M, W, R, S, K and the structural rows.*
8. *C records displayed* Cont *replay from dense-open requests through finite windows, regular-rational readback, the real seal, and the refusal row.*

9. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $B, M, W, R, S, K, H, C, P, N$. It has no coordinate for countable choice, an infinite selector, host set intersection, quotient stream equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3143.2 (BaireCategoryChoiceFree NameCert obligations)**.** *For every accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *carrier $Q = (B, M, W, R, S, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$ *surface consists exactly of the finite obligations:*

1. *carrier admission for the dense-open Baire row, complete-metric source row, finite windows, regular-rational readback, real seal, Specker refusal row, transport, replay, provenance, and local naming;*
2. *window coherence, requiring every dense-open request consumed from B to pass through the same* $\mathsf{StreamName}^{\uparrow}$ *window row W before the readback row R is visible;*
3. *metric-source coherence, requiring the complete-metric row M to supply the only Cauchy-thread and limit-witness route available to the packet;*
4. *real-seal exactness, requiring S to be downstream of B, M, W, R and not a detached host real object;*
5. *choice-free refusal, requiring K to block countable choice, hidden selectors, quotient stream equality, selected limits, and ambient completeness as local certificate data.*

Lean4 ✓ `BEDC.Derived.BaireCategoryChoiceFreeUp.TasteGate.BaireCategoryChoiceFreeTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4123.1. Its rows list the dense-open Baire boundary, complete metric source, finite windows, readback, real seal, Specker refusal row, transport, replay, provenance, and local naming, and no other row is available to the local certificate.

Window coherence follows by reading the $\mathsf{BaireCategory}^{\uparrow}$ schedule only through W. The regular-rational row R is downstream of that same finite window surface, so the carrier cannot replace it with an unlisted stream representative or a selector over all dense-open requests.

Metric-source coherence is supplied by the complete-metric row M. The Cauchy thread and limit-witness route are the displayed rows consumed from M, and the transport and replay rows H, C preserve that source without creating a second completeness datum.

Finally read the seal and refusal rows. The $\mathsf{Real}^{\uparrow}$ row S is exposed only after B, M, W, R, while K records the Specker-style obstruction to turning the packet into countable choice, quotient equality, a selected limit, or ambient completeness. Thus the five obligations exhaust the local NameCert surface. □

Theorem 3143.3 (BaireCategoryChoiceFree ProbeBundle induction)**.** *Every accepted probe-bundle consumer of a* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet advances by finite induction over the displayed window row W. At each step the consumer reads the current dense-open request from B, the complete-metric source row M, the regular-rational tolerance row R, and the structural rows H, C, P, N before it may inspect the next request. The induction has no branch that selects an infinite sequence of witnesses, identifies quotient streams, chooses a host limit, or invokes ambient completeness.*

Proof. Start with the finite window row W. It is the only row listing the requests visible to the consumer, so the induction domain is a displayed finite packet rather than a countable family.

For the base step, the consumer reads the initial dense-open and metric rows through B and M. The readback row R supplies the tolerance data for the inspected window, and H, C, P, N carry only transport, replay, provenance, and local naming.

For the inductive step, the next request is another entry of the same finite W row. The same B, M, R, H, C, P, N route is replayed for that entry, and the refusal row K prevents the step from being interpreted as a hidden selector or a host completeness argument.

Since every step is indexed by an entry already displayed in W, the probe-bundle induction remains finite and choice-free. The excluded analytic alternatives have no coordinate in the carrier. □

Theorem 3143.4 (BaireCategoryChoiceFree Real and Completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet must pass through the displayed route*

$$B, M, W, R, S, K, H, C, P, N$$

before reading any local NameCert surface. The route uses the $\mathsf{BaireCategory}^{\uparrow}$ *dense-open packet, the* $\mathsf{CompleteMetric}^{\uparrow}$ *source, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{SpeckerSequence}^{\uparrow}$ *refusal boundary. It exports no countable choice principle, quotient stream, selected limit, host equality test, or ambient Real completeness theorem.*

Proof. Read the carrier first. The only Real-facing endpoint is the row S, and S is downstream of the Baire boundary B, complete metric row M, finite windows W, and regular-rational readback R.

Apply the NameCert obligations. They force every local naming read through the same carrier, window, metric-source, real-seal, and refusal rows, so a Completion-facing consumer cannot bypass the finite route by citing an ambient completed object.

Use the probe-bundle induction for consumers that inspect several dense-open requests. The induction is finite over W and reuses the same B, M, R, H, C, P, N route at every inspected entry.

The Specker refusal row closes the boundary. It records that requests for countable choice, quotient stream equality, selected limits, host equality, or ambient Real completeness remain outside the packet. Therefore every Real or Completion consumer reads exactly the displayed finite choice-free route before the local NameCert is available. □

Theorem 3143.5 (BaireCategoryChoiceFree complete-metric handoff)**.** *Every Completion-facing consumer of an accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet must read the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row* M *together with*

$$B, W, R, S, K, H, C, P, N$$

before it may use the choice-free Baire packet. The handoff makes the complete-metric carrier, ball classifier, Cauchy-thread route, and limit-witness boundary visible as source data inside the same finite packet; it does not allow the consumer to replace M *by countable choice, a hidden selector, quotient stream equality, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4123.1. The only complete-metric source is the row M, and it is displayed alongside the dense-open Baire row B, finite windows W, regular-rational readback R, Real seal S, refusal row K, and structural rows H, C, P, N.

Use the NameCert obligations of Definition 4123.2. Their metric-source coherence row requires the Cauchy thread and limit-witness route to be consumed from M, while their window and real-seal rows keep M tied to the same $B, W, R, S, K, H, C, P, N$ packet.

Finally apply the Real and Completion boundary theorem, Definition 4123.4. A Completion-facing consumer must traverse the displayed finite route before any local certificate is available, so the complete-metric handoff is exactly the source row M read together with the remaining packet rows and not an ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ packet is generated from a BaireCategoryUp dense-open boundary row, a CompleteMetricUp source row, finite StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, a SpeckerSequenceUp refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4123.1, Definition 4123.2, Definition 4123.3, and Definition 4123.4 seed the finite choice-free Baire boundary for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full classical Baire category theorem, countable choice, quotient stream equality, selected limits, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coherence, complete-metric source coherence, real-seal exactness, Specker refusal, transport, provenance, and local naming for BEDC.Derived.BaireCategoryChoiceFreeUp.

Chapter 3144

A Concrete Naming Certificate for $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$

Following Riemann–Darboux integration and Bishop–Bridges constructive analysis, the RiemannIntegrabilityModulu packet records the finite modulus data that turns a Darboux gap estimate into a Cauchy route for integral names. It sits after the equal-mesh surface of chapter 2033, the Darboux finite-sum carrier of chapter 1880, the integrability carrier of chapter 1960, the real seal of chapter 38, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet gives later integration chapters a displayed finite handoff from upper-lower Darboux gaps to Cauchy integral windows without importing selected integrals, quotient functions, open-cover compactness, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3144.1 (Riemann integrability modulus carrier)**.** *A* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, U, L, G, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded* $\mathsf{Real}^{\uparrow}$ *interval and bounded-function observation row consumed by the Darboux and Riemann-integrable surfaces.*
2. *U is the* $\mathsf{UniformPartition}^{\uparrow}$ *row selecting an equal-mesh finite partition at a requested dyadic tolerance scale.*
3. *L is the lower Darboux finite-sum row read over the same interval and partition windows.*
4. *G is the upper Darboux finite-sum row read over the same interval and partition windows.*
5. *D is the Darboux-gap ledger, recording a positive dyadic tolerance request and the displayed comparison $G - L$ below that request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the Darboux-gap ledger into a regular rational Cauchy window for the integral name.*
7. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every endpoint, sum, gap, and integral-window read.*
8. *H records componentwise* hsame *transport for interval, partition, lower sum, upper sum, gap, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from Darboux-gap requests, Riemann-integrable reads, or integral-window consumers through $I, U, L, G, D, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive Darboux-to-Cauchy modulus history. It has no coordinate for a selected integral value, host function quotient, open cover of the interval, measure-completion object, choice principle, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RiemannIntegrabilityModulusUp.TasteGate.RiemannIntegrabilityModulusTasteGate`
`single carrier alignment`

Theorem 3144.2 (Riemann integrability modulus NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier $M = (I, U, L, G, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulu}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval, equal-mesh partition, lower Darboux sum, upper Darboux sum, gap ledger, regular rational readback, dyadic tolerance and Real seal, transport, replay, provenance, and local name;*

2. *partition coherence, requiring L and G to read the same I and U rows before the gap ledger D can be consumed;*

3. *Darboux-gap exactness, requiring D to expose only the displayed comparison between the upper and lower finite sums at the requested dyadic tolerance;*

4. *Cauchy-window readiness, requiring R to be read from D and E before any integral-window consumer receives a regular rational name;*

5. *Real seal nonescape, requiring every endpoint, sum, gap, or integral-window read to pass through E after the displayed finite Darboux rows have been exposed;*

6. *transport and replay, requiring H and C to preserve the same finite modulus history;*

7. *provenance and naming, requiring P and N to package and name exactly the displayed rows;*

8. *ledger refusal, excluding selected integrals, quotient functions, open-cover compactness, measure completion, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Riemann-integrability modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4124.1. The projection exposes exactly the interval, partition, lower-sum, upper-sum, gap, readback, seal, and structural rows.

The Darboux route is $I, U \mapsto L, G \mapsto D$. Since L and G read the same interval and partition rows, the comparison in D is a comparison between two finite Darboux sums attached to one displayed partition history.

The Cauchy handoff is then read from D, R, E. The gap ledger supplies the dyadic tolerance comparison, the regular rational readback turns that comparison into a Cauchy window, and the Real seal records only the resulting finite read.

Finally, H, C, P, N transport, replay, package, and name the same history. The excluded host objects have no carrier coordinate, so the listed obligations exhaust the local NameCert surface. □

Theorem 3144.3 (Darboux-to-Cauchy handoff)**.** *Every accepted* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier* $M = (I, U, L, G, D, R, E, H, C, P, N)$ *exports a finite route*

$$\mathsf{UniformPartition}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

in which a Darboux-gap request first reads the common partition row U*, then the paired lower and upper finite-sum rows* L, G*, then the gap ledger* D*, and only then the regular rational readback* R *and Real seal* E*. Any later Riemann-integral or completion-facing consumer of the integral name factors through this route and cannot bypass it by selecting an ambient integral value, quotienting functions, invoking an open-cover compactness theorem, completing a measure space, or using ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 4124.1. The only partition coordinate is U, and the only Darboux-sum coordinates are the paired lower and upper rows L and G.

The gap ledger D is defined after those rows, so any gap request must pass through the common partition and the paired finite sums before the tolerance comparison is available. The readback row R and the seal row E occur after D, hence the Cauchy-window read cannot precede the Darboux comparison.

The replay row C preserves that order for downstream consumers, while H, P, N preserve transport, provenance, and naming. Since no selected integral, quotient-function, open-cover, measure-completion, or ambient-completeness coordinate is present in the carrier, none can be used as an alternate route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ packet is generated from a closed bounded RealUp interval row, UniformPartitionUp equal-mesh row, paired DarbouxSumUp lower and upper finite sums, Darboux-gap ledger, RegSeqRatUp readback, DyadicRatCoreUp tolerance ledger, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4124.1, Definition 4124.2, and Definition 4124.3 seed the finite constructive Darboux-to-Cauchy modulus route over $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected integrals, quotient functions, open-cover compactness, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition coherence, Darboux-gap exactness, Cauchy-window readiness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegrabilityModulu

Chapter 3145

A Concrete Naming Certificate for $\mathsf{SequentialRealCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's completion route, the $\mathsf{SequentialRealCompletion}^{\uparrow}$ packet records the standard sequential completion path from regular Cauchy names to the $\mathsf{Real}^{\uparrow}$ seal as finite BEDC data. It sits after the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real-name seal of chapter 38, the completion interface of chapter 95, and the Cauchy-limit sealing routes used by $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers. The packet is finite regular-Cauchy completion data: it is not a quotient representative, not host equality of completed objects, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3145.1 (Sequential real completion carrier)**.** *A* $\mathsf{SequentialRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected observations of the candidate regular Cauchy name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the regular rational approximants visible in those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare window observations against the requested precision.*
4. *Q is the regular-Cauchy coherence row, recording tail agreement and modulus coverage only through displayed finite windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after W, R, D, Q have been displayed.*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row, recording that the same regular route is available to completion-facing consumers.*
7. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic ledger, coherence row, real seal, limit seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through $W, R, D, Q, E, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialRealCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $W, R, D, Q, E, L, H, C, P, N$. It has no coordinate for quotient representatives of Cauchy sequences, host equality of completed objects, countable choice over tails, an external limit selector, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentialRealCompletionUp.TasteGate.SequentialRealCompletionTasteGate single carrier alignment`

Theorem 3145.2 (Sequential real completion NameCert obligations)**.** *For every accepted* SequentialRealCompletion *carrier*

$$S = (W, R, D, Q, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window admission through W, RegSeqRat readback through R, DyadicRatCore tolerance admission through D, regular-Cauchy coherence through Q, RealUp seal admission through E, CauchyLimitSeal handoff through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4125.1. The rows W, R, D, Q are the regular Cauchy source route, while E and L are the only real-facing and completion-facing endpoints.

The finite route starts with windows. A precision request reads the displayed $\mathsf{StreamName}^{\uparrow}$ row W, extracts the regular rational observations through R, and checks the dyadic tolerance data through D before the coherence row Q may be consumed.

The endpoint rows are downstream of that same route. The $\mathsf{Real}^{\uparrow}$ seal E and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff L read only the displayed regular-Cauchy packet, and the structural rows H, C, P, N preserve the same packet under transport, replay, source, and naming.

Since no carrier coordinate contains quotient representatives, host equality, countable-choice schedules, external limit selectors, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3145.3 (Sequential real completion Cauchy-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentialRealCompletion}^{\uparrow}$ *carrier factors through the finite chain*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

as displayed by W, R, D, L, E and preserved by Q, H, C, P, N. The route exports a sealed regular-Cauchy completion packet only; it does not export quotient stream equality, a selected host real, countable choice, host equality of completions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Apply the obligation surface of Definition 4125.2. Any real-facing or completion-facing read must first expose the finite stream windows, regular rational readback, dyadic tolerance ledger, and coherence row.

The limit-seal row L is then read on the same route. It cannot be detached from W, R, D, Q, because its provenance and replay coordinates are the same H, C, P, N rows already attached to the carrier.

Finally the $\mathsf{Real}^{\uparrow}$ seal E is reached only after that limit-seal route is visible. Thus the consumer receives a finite regular-Cauchy completion packet and no quotient representative, selected host limit, countable-choice schedule, host equality, or ambient real-completeness coordinate. □

Falsifiable prediction: A $\mathsf{SequentialRealCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ endpoint without reading the displayed StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, regular-Cauchy coherence row, CauchyLimitSeal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, completion, cauchylimitseal: no listed sibling supplies the whole sequential real completion packet because each sibling contributes only windowing, regular readback, dyadic tolerance, real sealing, completion consumption, or limit-seal handoff while this chapter binds them into one finite regular-Cauchy completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialRealCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat regular rational readback, DyadicRatCore tolerance ledgers, regular-Cauchy coherence rows, a RealUp seal, a CauchyLimitSeal handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4125.1, Definition 4125.2, and Definition 4125.3 seed the finite regular-Cauchy completion route from sequential names to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient representatives, host equality of completed objects, countable choice, selected host limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window coverage, RegSeqRat readback, DyadicRatCore tolerance exactness, regular-Cauchy coherence, CauchyLimitSeal handoff, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialRealCompletionUp.

Chapter 3146

A Concrete Naming Certificate for MetaCICLocalJoinLedger$^{\uparrow}$

The MetaCICLocalJoinLedger$^{\uparrow}$ packet realizes the residual-substitution boundary of chapter 9378 as finite local-join data. It sits beside the local diamond witness of chapter 3595, the residual checker of chapter 3827, the confluence substitution boundary of chapter 3791, the confluence diamond boundary of chapter 4082, and the subject-reduction discharge socket of chapter 719. The carrier records local peaks, candidate joins, bounded checker handoffs, residual-substitution frontiers, obstruction-preserving readback, and TasteGate readiness without claiming full confluence, unrestricted substitution compatibility, subject reduction, evaluator equality, or kernel consistency.

Definition 3146.1 (MetaCIC local join ledger carrier)**.** *A* MetaCICLocalJoinLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, A, B, R, S, O, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the local peak row, naming the source term and the two outgoing reduction observations.*
2. *A is the candidate join row, recording the proposed common reduct and the two finite routes into it.*
3. *B is the bounded checker handoff row, exposing the closedness, support, branch compatibility, and variable-escape checks applied to the peak.*
4. *R is the residual frontier row, listing residual descendants and binder-depth data retained by the local join attempt.*
5. *S is the substitution-boundary row, recording the residual-substitution input that remains visible whenever the local join is conditional.*
6. *O is the obstruction readback row, naming the endpoints still outside the packet: unrestricted substitution compatibility, full closed Church–Rosser, arbitrary subject reduction, evaluator equality, and kernel consistency.*
7. *G is the* TasteGate *readiness row, reserving the later* Nontrivial *and* FieldFaithful *instance obligations for this finite carrier surface.*
8. *H records componentwise* hsame *transport for local peaks, candidate joins, bounded checks, residual frontiers, substitution boundaries, obstruction rows, TasteGate readiness, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a confluence-boundary query through $K, A, B, R, S, O, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, MetaCICConfluenceResi MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$, $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$, SubjectReductionDischargeSoc BHist, hsame, Cont, TasteGate, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, A, B, R, S, O, G, H, C, P, N$. It contains no coordinate for a global confluence theorem, unrestricted substitution compatibility, subject reduction, quotient equality of terms, evaluator equality, or kernel consistency.

Theorem 3146.2 (MetaCIC local join ledger obligations)**.** *For every accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier*

$$J = (K, A, B, R, S, O, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$ *surface consists exactly of local-peak exposure through K, candidate-join exposure through A, bounded-checker handoff through B, residual-frontier exposure through R, substitution-boundary retention through S, obstruction readback through O, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4126.1. Rows K, A, B, R, S, O are the confluence-facing ledger data, and rows G, H, C, P, N are the structural certificate data.

The local join attempt first reads the peak and candidate join. Row K gives the finite source and branch observations, while A records only the proposed common reduct and its two routes.

The bounded checker and residual rows remain visible before the join can be consumed. Row B records the finite checks actually performed, R keeps residual descendants and binder-depth data, and S keeps the substitution-sensitive input that prevents the ledger from being read as an unconditional diamond.

Finally O preserves the open endpoints, while G, H, C, P, N reserve the TasteGate instance surface, transport the rows, replay the query, source the packet, and name it. Since no carrier coordinate contains full confluence, unrestricted substitution compatibility, subject reduction, evaluator equality, quotient equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 3146.3 (MetaCIC local join ledger bounded checker handoff)**.** *Every checker-facing consumer of an accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier factors through*

$$K, A, B, R, S, O, G, H, C, P, N$$

before it may consume the local join ledger. The bounded checker row B is a finite handoff over the displayed peak and residual rows; it is not a global confluence checker, not a subject-reduction proof, and not an evaluator-equality decision.

Proof. Apply the obligation surface of Definition 4126.2. It forces a checker-facing read through the local peak, candidate join, bounded-checker handoff, residual frontier, substitution boundary, and obstruction row.

The handoff is therefore scoped by K, A, R, S, O. It can report the checks listed in B, but it has no coordinate from which to manufacture an unrestricted substitution theorem, a subject-reduction theorem, or a global Church–Rosser result.

The structural rows G, H, C, P, N preserve the same finite route under TasteGate readiness, transport, replay, provenance, and local naming. Hence the checker handoff remains bounded local-join data. □

Theorem 3146.4 (MetaCIC local join ledger obstruction-preserving readback)**.** *Every confluence-audit read exported by an accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier preserves the obstruction row O beside the candidate join. A consumer may pass the local ledger to the local-diamond witness, residual checker, substitution-boundary packet, diamond-boundary packet, or subject-reduction discharge socket named above, but it cannot erase the residual-substitution boundary or reinterpret the ledger as full MetaCIC confluence.*

Proof. Start from the bounded checker handoff of Definition 4126.3. The local join is consumed only after the peak, join, checker, residual, substitution-boundary, and obstruction rows have been displayed together.

The sibling handoffs preserve that pairing. The local-diamond witness may read K and A, the residual checker may read B, R, S, O, the substitution-boundary and diamond-boundary packets may read the same residual-substitution interface, and the subject-reduction discharge socket may read the obstruction as a boundary rather than as a proof.

Since O is a carrier coordinate and no sibling handoff adds a hidden theorem row, every readback keeps the obstruction visible. The exported ledger is therefore finite local-join data, not unrestricted substitution compatibility, full confluence, subject reduction, evaluator equality, or kernel consistency. □

Theorem 3146.5 (MetaCIC local join ledger TasteGate readiness)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *at the displayed finite carrier surface*

$$K, A, B, R, S, O, G, H, C, P, N.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the local peak, candidate join, bounded checker handoff, residual frontier, substitution boundary, obstruction readback, transport, replay, provenance, and local NameCert data.

Lean4 ✓ `BEDC.Derived.MetaCICLocalJoinLedgerUp.MetaCICLocalJoinLedgerTasteGate single carrier alignment`

Proof. The readiness row G in Definition 4126.1 is tied to the whole local-join ledger rather than to a single obstruction field.

Nontriviality must exhibit an accepted packet with the displayed finite rows present. FieldFaithfulness must show that deleting any of $K, A, B, R, S, O, H, C, P, N$ changes the readable carrier surface. Thus the TasteGate burden remains at the same finite packet boundary as the NameCert obligations. □

Falsifiable prediction: A $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ confluence-audit read factors through displayed local-peak rows, candidate-join rows, bounded-checker handoff rows, residual-frontier rows, substitution-boundary rows, obstruction readback rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacic-local-diamond-witness, metacic-confluence-residual-checker, metacic-confluence-substitution-boundary, metacic-confluence-diamond-boundary, subject-reduction-discharge-socket: none is bijective with $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ because this packet jointly carries a local peak, candidate join, bounded checker handoff, residual frontier, substitution boundary, obstruction-preserving readback, TasteGate readiness, replay, transport, provenance, and local naming as one ledger surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ packet is generated from local-peak rows, candidate-join rows, bounded-checker handoff rows, residual-frontier rows, substitution-boundary rows, obstruction readback rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4126.1, Definition 4126.2, Definition 4126.3, Definition 4126.4, and Definition 4126.5 seed the finite local-join ledger for MetaCIC confluence audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full confluence, unrestricted substitution compatibility, arbitrary subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-peak exposure, candidate-join exactness, bounded-checker handoff, residual-frontier coverage, substitution-boundary retention, obstruction-preserving readback, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICLocalJoinLedgerUp.

Chapter 3147

A Concrete Naming Certificate for $\mathsf{HemicompactSpace}^{\uparrow}$

Following Kelley and Engelking on hemicompact spaces, the $\mathsf{HemicompactSpace}^{\uparrow}$ packet records compact exhaustion as explicit finite-name data. It sits over the metric carrier of chapter 57, the compact metric window surface of chapter 129, the finite stream schedule of chapter 126, the formal-ball neighborhood route of chapter 1510, and the real-radius seal of chapter 38. The carrier names a countable displayed exhaustion by compact windows; it does not infer open-cover compactness, select covers by choice, or assume that the whole host space is compact.

Definition 3147.1 (Hemicompact-space carrier)**.** *A* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, K, S, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance readback, and finite observation histories used by the exhaustion.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *exhaustion-window row; its entries name compact metric windows K_i as finite BEDC data and expose their local compact certificates only when the corresponding index is displayed.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *index schedule recording the inspected exhaustion indices, monotone window requests, and replayed finite prefixes.*
4. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *neighborhood-window row that reads local center-radius data inside a displayed compact window before any neighborhood consumer uses it.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius-seal row attached to the formal-ball windows and to the compact-window containment tests.*
6. *H records componentwise* hsame *transport for M, K, S, F, R, C, P, N and for the local classifier row.*
7. *C records displayed* Cont *replay from an index request through the compact window, formal-ball neighborhood window, real-radius seal, transport, provenance, and local name.*
8. *P is the* Pkg *provenance over* $\mathsf{HemicompactSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying $M, K, S, F, R, H, C, P, N$. It has no coordinate for an open-cover compactness theorem, a choice-selected cover, a selected global compactification, host-space compactness, quotient equality of exhaustion sequences, or ambient completeness.

Lean4 ✓ `BEDC.Derived.HemicompactSpaceUp.HemicompactSpaceTasteGate single carrier alignment`

Theorem 3147.2 (HemicompactSpace NameCert obligations)**.** *For every accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier $E = (M, K, S, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, compact-window exhaustion, stream index schedule, formal-ball neighborhood windows, real-radius seals, transport, replay, provenance, and local name;*
2. *compact-window accountability, requiring every compactness read to pass through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *row K at an index supplied by the schedule S;*
3. *neighborhood-window accountability, requiring every local neighborhood read to pass through the* $\mathsf{FormalBall}^{\uparrow}$ *row F and the* $\mathsf{Real}^{\uparrow}$ *radius seals R over the same metric source M;*
4. *componentwise* hsame *transport and* Cont *replay stability for the same finite rows;*
5. *ledger refusal, stating that open-cover compactness, choice-selected covers, host compactness, quotient equality of exhaustion sequences, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the carrier in Definition 4127.1. The analytic rows are exactly the metric source, compact-window exhaustion, index schedule, formal-ball neighborhood row, and real-radius seals.

The compact and neighborhood obligations are ordered by the displayed schedule. A consumer first reads an index in S, then the compact metric window in K, and only then the formal-ball and real-radius rows F, R available over that window.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the excluded compactness and choice principles are absent from the carrier, no NameCert obligation can export them. □

Theorem 3147.3 (Hemicompact exhaustion window)**.** *Every accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *read of a compact exhaustion window factors through*

$$M, S, K, F, R, H, C, P, N.$$

In particular, the consumer receives an index from the $\mathsf{StreamName}^{\uparrow}$ *schedule, the corresponding* $\mathsf{CompactMetric}^{\uparrow}$ *window, the local* $\mathsf{FormalBall}^{\uparrow}$ *neighborhood rows, and the* $\mathsf{Real}^{\uparrow}$ *radius seals over the same* $\mathsf{Met}^{\uparrow}$ *source before the local* NameCert *row is consumed. The route exports no open-cover compactness proof, selected cover family, host-space compactness witness, quotient exhaustion equality, or ambient completeness.*

Proof. Start with the index schedule. The row S is the only coordinate that can name which exhaustion window is being inspected, so a compact-window read cannot begin with an unindexed compact object.

Read K, F, R at the displayed index. The compact metric window comes from K, while formal-ball neighborhoods and radius seals are replayed through F and R over the same metric source M.

Finally attach H, C, P, N. These rows preserve the same finite route under transport and replay, and they contain no open-cover, choice-selected, quotient, host compactness, or ambient-completeness coordinate. □

Theorem 3147.4 (Hemicompact nonescape)**.** *Every neighborhood or compact-window consumer of an accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier remains inside the finite route through* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. The consumer cannot escape to open-cover compactness, choice-selected finite subcovers, a compact host-space assertion, selected representatives of exhaustion windows, quotient equality, or ambient completeness.*

Proof. Project the accepted packet. The only analytic coordinates are M, K, S, F, R, and each is displayed before the local naming row is available.

Use the exhaustion-window theorem. It forces compact and neighborhood reads to pass through the scheduled compact metric window, formal-ball row, and real-radius seal rather than through an external compactness theorem.

The remaining rows are structural. Transport, replay, provenance, and local naming keep the read inside the displayed BHist packet, so there is no coordinate from which a consumer can obtain the excluded principles. □

Falsifiable prediction: Every $\mathsf{HemicompactSpace}^{\uparrow}$ consumer of a compact exhaustion window reads a MetricSpaceUp source, CompactMetricUp window, StreamNameUp index schedule, FormalBallUp neighborhood row, RealUp radius seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: metricspace, compactmetric, streamname, formalball, real: $\mathsf{HemicompactSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an indexed compact-exhaustion schedule, compact metric windows, formal-ball neighborhood windows, real-radius seals, transport, replay, provenance, and local NameCert rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{HemicompactSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp source, CompactMetricUp exhaustion windows, a StreamNameUp index schedule, FormalBallUp neighborhood windows, RealUp radius seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4127.1, Definition 4127.2, Definition 4127.3, and Definition 4127.4 seed the finite compact-exhaustion interface over scheduled compact metric windows and formal-ball neighborhoods.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, host-space compactness, quotient exhaustion equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window accountability, index-schedule exactness, formal-ball and real-radius replay, transport stability, provenance, and local naming for BEDC.Derived.HemicompactSpaceUp.

Chapter 3148

A Concrete Naming Certificate for $\mathsf{DistanceFunctionCompletion}^{\uparrow}$

Following Bourbaki and Kelley on metric completion by distance functions, the $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ packet records the L10 route where a metric source is read through its distance-to-source observations before later completion and separated-completion consumers inspect it. It sits after the metric carrier of chapter 57, the metric-completion boundary of chapter 1506, the Kuratowski embedding route of chapter 7569, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the distance-function completion route as finite BEDC data; it is not a quotient completion, selected representative, host equality of completed points, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3148.1 (DistanceFunctionCompletion carrier)**.** *A* $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, C, K, S, R, D, A, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distances are observed.*
2. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row reached only after the source metric observations have been displayed.*
3. *K is the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *distance-function handoff row from the source metric into the completion-facing observation space.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected distance observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values exposed by those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested distances and finite approximants.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed distance value.*
8. *H records componentwise* hsame *transport for the metric source, completion row, Kuratowski handoff, stream windows, regular readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay of the route*

$$M, K, S, R, D \longmapsto A \longmapsto C \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* DistanceFunctionCompletion$^{\uparrow}$, Met$^{\uparrow}$, MetricCompletion$^{\uparrow}$, KuratowskiEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{DistanceFunctionCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, C, K, S, R, D, A, H, T, P, N$. It has no coordinate for quotient completion classes, selected completed points, host equality of distance functions, countable choice over source points, ambient completeness of Real$^{\uparrow}$*, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DistanceFunctionCompletionUp.DistanceFunctionCompletionTasteGate single carrier alignment`

Theorem 3148.2 (DistanceFunctionCompletion NameCert obligations)**.** *For every accepted* DistanceFunctionComp *carrier*

$$F = (M, C, K, S, R, D, A, H, T, P, N),$$

the local NameCert$_{\text{DistanceFunctionCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, metric-completion consumer, Kuratowski embedding, stream-window, regular-rational, dyadic-tolerance, real-seal, transport, replay, provenance, and local naming rows;*
2. *distance-function exposure, requiring every completion-facing read to pass through M and K before C may be consumed;*
3. *L10 readback, requiring each displayed distance observation to pass through S, R, D, A before it is available to a completion consumer;*
4. *transport stability, requiring* hsame *comparison to preserve the source metric, distance-function handoff, readback route, completion consumer, replay, provenance, and local naming rows componentwise;*
5. *replay locality, requiring* Cont *replay to use only the displayed route from M, K, S, R, D through A and C;*
6. *non-escape, excluding quotient completion classes, selected completed points, host equality of completed points or distance functions, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4128.1. The displayed packet supplies exactly the source metric row, the distance-function handoff, the finite observation route, the completion consumer, and the structural rows.

Distance-function exposure starts with M and K. A completion-facing read has no access to C until the source metric and its Kuratowski distance observations have both been displayed.

The L10 readback is the route S, R, D, A. Stream windows name the inspected observations, regular rational rows read them back, dyadic tolerance rows compare requested scales, and the real seal records the terminal value visible to completion consumers.

Finally H, T, P, N preserve and name the same route. Since no excluded quotient, selector, host-equality, choice, completeness, verification, or bridge object is a carrier row, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ packet is generated from a MetricUp source row, a MetricCompletionUp consumer row, a KuratowskiEmbeddingUp distance-function handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4128.1 and Definition 4128.2 seed the finite distance-function completion route through metric source observations, Kuratowski embedding, L10 readback, and completion consumption.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion classes, selected completed points, host equality of completed points or distance functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, distance-function exposure, L10 readback, transport stability, replay locality, provenance, and local naming for BEDC.Derived.DistanceFunctionCompletionUp.

Chapter 3149

A Concrete Naming Certificate for $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$

The $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet realizes the lift-fiber decomposition of Definition 9462.1687 as finite BEDC NameCert data. It sits beside the codon-window carrier of chapter 2234, the lift-orbit classifier of chapter 4059, the DNA boundary-tile packet of chapter 3393, and the trigger-algebra surface of chapter 3478. The packet records quotient supports, binary lift choices, orbit fibers, and lift-flip ledgers as displayed finite rows; it is not biological mechanism, stochastic fitness, host quotient semantics, or an unrecorded enumeration oracle.

Definition 3149.1 (Minimal trigger lift-bundle carrier)**.** *A* $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, L, O, F, G, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient-support row, listing a quotient-level minimal trigger and its displayed support cells.*
2. *L is the lift-choice row, recording one binary f_3 lift coordinate for each displayed quotient support.*
3. *O is the orbit-fiber row, naming the fiber dimension and orbit type read from the quotient minimal-trigger table.*
4. *F is the finite fiber-count row, displaying the Q_1, Q_2, Q_3, and Q_4 multiplicities and the codon count readback.*
5. *G is the lift-flip ledger, recording the within-fiber one-coordinate flip graph and its edge-count row.*
6. *H records componentwise* hsame *transport for quotient supports, lift choices, orbit fibers, fiber counts, lift-flip ledgers, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a quotient support through lift choices, fiber counting, orbit labeling, and flip-ledger exposure.*
8. *P is the* Pkg *provenance over* $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$*, the source theorem Definition 9462.1687,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, L, O, F, G, H, C, P, N$. It has no coordinate for hidden quotient semantics, empirical biology, probabilistic fitness, an unrestricted search space, or a lift choice detached from the displayed quotient support.

Lean4 ✓ `BEDC.Derived.MinimalTriggerLiftBundleUp.MinimalTriggerLiftBundleTasteGate single carrier alignment`

Theorem 3149.2 (Minimal trigger lift-bundle NameCert obligations)**.** *For every accepted* MinimalTriggerLiftBundle *carrier $B = (Q, L, O, F, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$ *surface consists exactly of quotient-support admission through Q, lift-choice admission through L, orbit-fiber readback through O, finite fiber-count preservation through F, lift-flip ledger exposure through G, componentwise* hsame *transport through H, displayed replay through C, provenance through P, and local naming through N. The carrier admits no read of a codon-level trigger before the quotient support and the matching lift-choice row have both been displayed.*

Proof. Project the carrier of Definition 4129.1. The quotient-support row Q fixes the base trigger, and the lift-choice row L records exactly one binary lift coordinate for each displayed support.

The orbit-fiber row O and count row F are read from the decomposition of Definition 9462.1687. The lift-flip row G records adjacency only inside the same displayed fiber, so no cross-fiber bijection is introduced.

Transport and replay preserve the rows componentwise, while provenance and local naming source the same finite packet. Therefore the listed rows exhaust the local NameCert obligations and prevent detached quotient semantics or hidden enumeration data. □

Theorem 3149.3 (Minimal trigger lift-bundle vision handoff)**.** *The decomposition*

$$\mathcal{T}_c = Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4$$

is consumed by $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *only through the finite route*

$$Q, L \longmapsto O, F, G \longmapsto H, C, P, N.$$

Later Lean-facing TasteGate files for this packet must provide Nontrivial $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ MinimalTriggerLiftBundle}^{\uparrow}$ *instances before the packet can serve as a verified downstream trigger-lattice consumer.*

Proof. Read the source decomposition through its displayed finite rows. The quotient support gives Q, and the independent binary lift choices give L.

The fiber dimension, multiplicity, and codon count are then recorded by O and F. The lift-flip graph row G is internal to each fiber and does not identify different quotient supports.

The structural rows H, C, P, N transport, replay, source, and name exactly that finite route. The stated TasteGate obligations are therefore downstream instance requirements, not hidden assumptions inside the present paper carrier. □

Falsifiable prediction: For every accepted $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet, replacing each displayed row by a componentwise hsame row preserves the finite fiber counts $1, 25, 22, 1$ over Q_1, Q_2, Q_3, Q_4 and the codon total 294.

Independence witness: codon-six-bit-window, codon-lift-orbit-classifier, dna-window-six-boundary-tile, window-coverage-trigger-algebra: none is bijective with $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ because this

packet jointly carries quotient supports, lift-choice vectors, orbit fibers, finite fiber counts, lift-flip ledgers, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet is generated from quotient-support rows, binary lift-choice rows, orbit-fiber rows, finite fiber-count rows, lift-flip ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4129.1, Definition 4129.2, and Definition 4129.3 seed the finite lift-fiber NameCert route over quotient supports, lift choices, orbit fibers, fiber counts, and flip ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, host quotient semantics, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient-support admission, lift-choice admission, orbit-fiber readback, finite fiber-count preservation, lift-flip ledger exposure, and TasteGate instances for BEDC.Derived.MinimalTriggerLiftBundleUp.

Chapter 3150

A Concrete Naming Certificate for $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ packet records an explicit tolerance-to-source-window modulus as finite BEDC NameCert data. It sits on the finite stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric source rows of chapter 57, the continuous-map graph of chapter 130, and the downstream modulus consumer of chapter 369. The packet names the constructive uniform-continuity modulus needed by the open L10 real and completion route; it does not import open-cover compactness, countable choice, quotient real equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3150.1 (Bishop uniform-continuity modulus carrier)**.** *A* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, F, T, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row naming the finite input observations that a tolerance request may inspect.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, including the displayed pointwise continuity data whose local bounds are eligible for uniform readback.*
3. *T is the tolerance-row carrier admission ledger, recording the requested target precision and the finite source-window budget produced for that request.*
4. *L is the pointwise-to-uniform modulus ledger, displaying the finite route from local continuity rows to one uniform source tolerance.*
5. *D is the dyadic lower-bound readback row, computed inside the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance discipline and exposed before any real-valued consumer reads the modulus.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source and target estimates.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after S, F, T, L, D, R have been displayed.*
8. *H records componentwise* hsame *transport for source-window, continuous-map, tolerance, ledger, dyadic, regular-readback, real-seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a target-tolerance request through the same finite modulus route.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, L, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected representatives, quotient equality of real names, a hidden host modulus, or ambient completion.

Lean4 ✓ `BEDC.Derived.BishopUniformContinuityModulusUp.TasteGate.BishopUniformContinuityModulusTasteGate`
`single carrier alignment`

Theorem 3150.2 (Bishop uniform-continuity modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *carrier $M = (S, F, T, L, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tolerance-row carrier admission through T;*

2. *pointwise-to-uniform modulus ledger exposure through L;*

3. *dyadic lower-bound readback through D;*

4. $\mathsf{Real}^{\uparrow}$ *seal nonescape through E;*

5. *provenance and name exhaustion through P, N together with transport and replay rows H, C.*

Every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet must read those rows through the displayed carrier route, and no accepted read may project compactness, choice, quotient equality, selected representatives, or a host modulus.*

Proof. Project the carrier of Definition 4130.1. The source-window row S and continuous-map row F are visible before any tolerance output is read.

The tolerance row T and ledger row L record the constructive step from a requested target precision to a uniform source window. The dyadic row D supplies the finite lower-bound comparison, and the regular readback row R makes that comparison available to real-name consumers.

Finally the seal row E is downstream of the displayed route, while H, C, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, the listed five obligation rows exhaust the local NameCert surface. □

Theorem 3150.3 (Bishop uniform-continuity modulus compact handoff)**.** *Any compact-uniform-continuity, continuous-map, or* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *packet must factor its read through*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg,

Thus the packet supplies a finite constructive modulus object for later compact-uniform-continuity and completion-facing chapters without importing choice, open-cover compactness, selected representatives, quotient equality, or ambient real completeness.

Proof. Use Definition 4130.2. A downstream read first exposes the finite source window, the continuous-map row, the tolerance row, the pointwise-to-uniform ledger, and the dyadic lower-bound readback.

The regular rational row and real seal then pass the same displayed estimates to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The $\mathsf{UniformModulus}^{\uparrow}$ output is therefore a named row of the finite packet, not a hidden host function.

Transport, replay, provenance, and local naming preserve only the displayed coordinates. The compact handoff can cite the modulus object, but it receives no open-cover compactness, choice principle, quotient equality, selected representative, or ambient completeness theorem from this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ packet is generated from source-window rows, continuous-map rows, tolerance rows, pointwise-to-uniform ledgers, dyadic lower-bound readback, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4130.1, Definition 4130.2, and Definition 4130.3 seed the finite Bishop uniform-continuity modulus route for L10 real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected representatives, quotient real equality, host modulus functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-row admission, pointwise-to-uniform ledger exposure, dyadic lower-bound readback, Real seal nonescape, and provenance/name exhaustion for BEDC.Derived.BishopUniformContin

Chapter 3151

A Concrete Naming Certificate for $\mathsf{CompactContinuousInverse}^{\uparrow}$

Following Kelley and Bishop–Bridges on compact Hausdorff inverse-continuity arguments, the $\mathsf{CompactContinuousInverse}^{\uparrow}$ packet records the finite BEDC surface by which a continuous bijection from a compact finite-net source into a separated target receives a displayed inverse-continuity handoff. It sits over the compact carrier of chapter 60, the continuous-map graph of chapter 130, the separated target boundary of chapter 1411, and the uniform-homeomorphism consumer of chapter 1815. The packet names only finite-net inverse readback; it does not import open-cover compactness, quotient equality of maps, host inverse functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, or a global topological inverse theorem.

Definition 3151.1 (Compact continuous inverse carrier)**.** *A* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, B, S, G, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row, exposing the compact packet only through named finite covers and source-window witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the forward graph and its local continuity data over the displayed compact source.*
3. *B is the bijection ledger, recording graph-totality, graph-single-valuedness, target coverage, and source uniqueness for the same displayed map row.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *target-separation row, or the corresponding Hausdorff separation read when the target is supplied through* $\mathsf{HausdorffSpace}^{\uparrow}$*.*
5. *G is the inverse graph readback row: a target observation reads a unique source window only through B and then returns to the forward graph row F.*
6. *M is the finite-net modulus extraction row, turning the compact source row K, graph row F, bijection ledger B, and separation row S into a named inverse-continuity tolerance window.*
7. *H records componentwise* hsame *transport for the compact, graph, bijection, separation, inverse-readback, and modulus rows.*
8. *C records displayed* Cont *replay from a target-window request through G and M back to the same finite source-net route.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactContinuousInverse}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient host inverse function, quotient equality of maps, open-cover compactness, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactContinuousInverseUp.CompactContinuousInverseTasteGate single carrier alignment`

Theorem 3151.2 (Compact continuous inverse NameCert obligations)**.** *For every accepted* CompactContinuousInve *carrier $I = (K, F, B, S, G, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact source, continuous graph, bijection ledger, separated target, inverse graph readback, finite-net modulus, hsame transport, Cont replay, provenance, and local naming rows. Every inverse-continuity read must inspect K, F, B, S before G and M are available, and every transport or replay read must preserve the same ordered route through H, C, P, N.*

Proof. Project the carrier of Definition 4131.1. The rows K, F, B, S display the compact source, forward graph, bijection data, and target separation before any inverse read is admitted.

The inverse graph row G is therefore a readback of the same graph through the bijection ledger, not a host inverse function. The finite-net modulus row M consumes the compact finite-net row and the separated target row after the graph has been fixed.

Finally H, C, P, N transport, replay, source, and name exactly that route. Since the carrier has no additional inverse, quotient-map equality, selected representative, choice, open-cover, or ambient-completeness coordinate, the stated rows exhaust the local NameCert obligations. □

Theorem 3151.3 (Compact continuous inverse uniform handoff)**.** *Any* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *or inverse-continuity consumer that reads an accepted* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *packet must factor its read through*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The exported handoff is the finite-net modulus row M together with the inverse graph readback row G; it is not an open-cover compactness theorem, a quotient equality of functions, a selected host inverse, a countable-choice construction, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with an accepted consumer read. The carrier admits only $K, F, B, S, G, M, H, C, P, N$, so the source compactness and forward graph data must be visible before the inverse graph is read.

The row G turns target observations back into source windows through the bijection ledger. The row M then extracts the finite-net modulus from the compact source and separated target route.

Transport and replay preserve that same route through H and C, while P and N attach provenance and local naming. The consumer may use the resulting modulus as a uniform-homeomorphism handoff, but it receives no extra host inverse, quotient equality, open-cover compactness, choice, or real-completeness principle. □

Theorem 3151.4 (Compact continuous inverse finite-net readback)**.** *Let $I = (K, F, B, S, G, M, H, C, P, N)$ be an accepted* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *carrier. Every target-neighbourhood request for inverse continuity is answered by first reading the inverse graph row G, then the finite-net modulus row M, and then replaying the answer through C over the same K, F, B, S dependency route.*

Proof. The request enters through the separated target row S and the inverse graph readback row G. The graph readback uses the bijection ledger B and the forward graph row F, so it cannot select a host inverse independently of the displayed map.

The finite-net modulus row M then uses the compact source row K to extract a finite source-window response. This is the only modulus coordinate in the packet.

The replay row C returns the answer along the same ordered rows, and H, P, N preserve the route under transport, provenance, and local naming. No open-cover, quotient-equality, selected-representative, choice, or ambient-completeness read is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactContinuousInverse}^{\uparrow}$ packet is generated from compact finite-net source rows, continuous-map graph rows, bijection ledgers, separated-target rows, inverse graph readback, finite-net modulus extraction, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4131.1, Definition 4131.2, Definition 4131.3, and Definition 4131.4 seed the finite inverse-continuity route from compact source data, a continuous bijection ledger, separated target data, inverse graph readback, and finite-net modulus extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient map equality, host inverse functions, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-bijection ledger exactness, separated-target readback, inverse graph readback, finite-net modulus extraction, transport replay, provenance, and local NameCert rows for BEDC.Derived.CompactContinuousInverseUp.

Chapter 3152

A Concrete Naming Certificate for $\mathsf{HausdorffRealCompletion}^{\uparrow}$

Following Bishop–Bridges constructive real analysis and Bourbaki's separated completion discipline, the $\mathsf{HausdorffRealCompletion}^{\uparrow}$ packet records the route by which the separated Hausdorff completion surface is read through the displayed regular real-name faces. It sits after the separated completion carrier of chapter 402, the terminal real-name seal of chapter 38, and the regular rational readback of chapter 301. The packet gives completion-facing consumers a named separated Real-completion certificate without asserting quotient completion equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3152.1 (Hausdorff real completion carrier)**.** *A* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion row that exposes the separated carrier and its local equality boundary.*
2. *U is the* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *uniqueness row read only as separated limit uniqueness inside the displayed completion route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule supplying the observable Cauchy-name requests.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling all finite radii and comparison slacks.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from S and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the regular readback route.*
7. *H records componentwise* hsame *transport for the separated-completion, uniqueness, stream, dyadic, regular-rational, real-seal, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a separated Real-completion request through $A, U, S, D, R, E, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{HausdorffRealCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, U, S, D, R, E, H, C, P, N$. It stores no quotient completion equality, selected completion point, host real equality, countable-choice schedule, hidden Cauchy filter, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.HausdorffRealCompletionUp.HausdorffRealCompletionTasteGate single carrier alignment`

Theorem 3152.2 (Hausdorff real completion NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *carrier*

$$Q = (A, U, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion exposure through A,* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *separated uniqueness exposure through U, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal discipline through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4132.1. The packet displays exactly the separated-completion row, uniqueness row, finite windows, dyadic tolerance ledger, regular rational readback, terminal real seal, and structural rows.

Read the separated-completion route first. The row A supplies only the Hausdorff completion surface, while U records the uniqueness boundary available inside that separated route.

Read the Real-name route next. The finite windows S are measured by the dyadic ledger D, the regular rational row R is read from those displayed requests, and the real seal E is reached only after the regular readback appears.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further analytic coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3152.3 (Hausdorff real completion separated limit uniqueness)**.** *Every completion-facing consumer of an accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *packet receives separated limit uniqueness only through the finite route*

$$\mathsf{HausdorffCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionUniqueness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The route is a separated Real-completion readback certificate, not a quotient completion equality theorem, a selected representative, a hidden Cauchy-filter construction, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Apply Definition 4132.2. It orders every public completion-facing read through A, U, S, D, R, E before any separated uniqueness claim is visible.

The separated side is local to A and U. The Hausdorff completion row supplies the separated carrier boundary, and the metric-completion uniqueness row supplies only the uniqueness read available inside that displayed boundary.

The real-name side is local to S, D, R, E. Stream windows, dyadic tolerances, and regular rational readback must be displayed before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed by the same route.

Thus a consumer cannot replace the route by quotient completion equality, selected representatives, hidden filters, host equality, or ambient completeness. The packet records separated limit uniqueness only as the displayed finite NameCert route. □

Theorem 3152.4 (Hausdorff real completion nonescape)**.** *No accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *packet exports quotient completion equality, a selected completion representative, hidden Cauchy-filter data, countable choice, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*. Every exported separated-completion or Real-facing read is confined to the rows* $A, U, S, D, R, E, H, C, P, N$*.*

Proof. Inspect the carrier definition. Its analytic coordinates are exactly the separated-completion row A, uniqueness row U, stream windows S, dyadic tolerance ledger D, regular rational readback R, and real seal E.

The structural rows add no new analytic principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet rather than introducing quotient equalities, representative choices, hidden filters, host equality, or completeness.

Use the separated uniqueness theorem, Definition 4132.3. Every separated-completion or Real-facing consumer must pass through the displayed finite route, so none of the excluded data can escape from the packet. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffRealCompletion}^{\uparrow}$ consumer that claims separated limit uniqueness without reading $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: hausdorffcompletion, metriccompletionuniqueness, streamname, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{HausdorffRealCompletion}^{\uparrow}$ because this packet binds separated completion, separated limit uniqueness, finite windows, tolerance accounting, regular readback, Real seal, transport, replay, provenance, and local naming as one L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffRealCompletion}^{\uparrow}$ packet is generated from a HausdorffCompletion separated-completion row, MetricCompletionUniqueness separated uniqueness row, StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 4132.1, Definition 4132.2, Definition 4132.3, and Definition 4132.4 seed the separated Real-completion route over HausdorffCompletionUp, MetricCompletionUniquenessUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, hidden Cauchy filters, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, separated-completion exposure, uniqueness exposure, window exactness, tolerance exactness, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and nonescape for BEDC.Derived.HausdorffRealCompletionUp.

Chapter 3153

A Concrete Naming Certificate for $\mathsf{NeighborhoodFilter}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{NeighborhoodFilter}^{\uparrow}$ packet records the local filter generated by displayed neighbourhood rows around a named point or finite source window. It sits between the finite directed basis interface of chapter 1601, the Cauchy-filter surface of chapter 333, the metric source of chapter 57, the open-neighbourhood discipline of chapter 88, and the completion handoff of chapter 1569. The packet names a local neighbourhood-filter interface; it does not quotient filters, select ultrafilters, choose limits, assert host point equality, or import ambient completeness.

Definition 3153.1 (NeighborhoodFilter carrier)**.** *A* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, x, O, B, D, C, M, T, H, K, P, N)$$

with the following displayed rows:

1. *X is the source carrier row whose visible points are consumed through* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *data.*
2. *x is the focused point or source-window row around which the local filter is read.*
3. *O is the finite open-neighbourhood row, exposed through the* $\mathsf{Topology}^{\uparrow}$ *classifier.*
4. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row generated by the displayed neighbourhood family.*
5. *D is the directed-refinement ledger: two displayed neighbourhood rows admit a smaller displayed neighbourhood row before any filter consumer reads downstream data.*
6. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row used when the local neighbourhood filter is consumed by completion routes.*
7. *M is the* $\mathsf{Met}^{\uparrow}$ *witness row when neighbourhoods are read as metric balls or radius windows.*
8. *T records the* $\mathsf{Topology}^{\uparrow}$ *provenance that the displayed open rows are local neighbourhoods of the same focused source.*
9. *H records componentwise* hsame *transport for source, point, open row, basis row, directed refinement, Cauchy compatibility, metric read, topology provenance, replay, package, and naming rows.*
10. *K records displayed* Cont *replay from a local-neighbourhood request through $X, x, O, B, D, C, M, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{NeighborhoodFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$ *row.*
The carrier contains no coordinate for ultrafilters, quotient filters, selected cluster points, host equality of neighbourhood systems, countable-choice selectors, or ambient completion theorems.

Lean4 ✓ `BEDC.Derived.NeighborhoodFilterUp.NeighborhoodFilterTasteGate single carrier alignment`

Theorem 3153.2 (NeighborhoodFilter directed completion base)**.** *For every accepted* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier* $U = (X, x, O, B, D, C, M, T, H, K, P, N)$*, every completion-facing read of the local filter first traverses the directed* $\mathsf{FilterBase}^{\uparrow}$ *row* B *and its refinement ledger* D*. The* $\mathsf{CauchyFilter}^{\uparrow}$ *row* C *and the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer may read only the route*

$$X, x, O \longmapsto B, D \longmapsto C, M, T, H, K, P, N.$$

Thus local-neighbourhood completion data are exported as finite basis rows rather than quotient filters, ultrafilters, selected limits, or hidden host neighbourhood equality.

Proof. Project the carrier of Definition 4133.1. The only basis-facing coordinates are the open-neighbourhood row O, the generated $\mathsf{FilterBase}^{\uparrow}$ row B, and the directed-refinement ledger D.

The completion side cannot bypass those rows. A $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer reads C only after B and D have supplied the local directed base, while M and T record the metric and topological provenance of the same neighbourhood family.

Finally the structural rows H, K, P, N transport, replay, package, and name that route. Since ultrafilters, quotient filters, selected limits, host equality, and choice selectors are absent from the carrier, every completion-facing read is exhausted by the displayed basis route. □

Theorem 3153.3 (NeighborhoodFilter local Cauchy handoff)**.** *Let* $U = (X, x, O, B, D, C, M, T, H, K, P, N)$ *be an accepted* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier. Whenever a Cauchy-filter completion route consumes the local neighbourhood filter, the handoff factors through the same* $\mathsf{FilterBase}^{\uparrow}$ *row* B*, directed-refinement ledger* D*,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *provenance rows* M, T*, and local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$ *row* N*. No consumer receives an independent filter object beyond the finite packet* U*.*

Proof. Begin with Definition 4133.2. It fixes the source-to-basis-to-completion route before any completion-facing read can inspect the Cauchy row C.

Use the metric and topological provenance rows next. The row M records ball-style local reads when available, while T records open-neighbourhood membership through the topology carrier; both are attached to the same focused source X, x, O.

The local NameCert row N closes the handoff by naming the visible packet U. Thus the Cauchy-filter completion route receives finite neighbourhood-filter data, not a detached filter, selected limit, ultrafilter, quotient, host equality witness, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4133.1, Definition 4133.2, and Definition 4133.3 seed the local neighbourhood-filter interface over FilterBaseUp, CauchyFilterUp, MetricUp, TopologyUp, CauchyFilterCompletionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ultrafilters, quotient filters, selected limits, host equality of neighbourhood systems, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, directed-refinement coverage, metric and topology provenance, Cauchy-filter handoff, completion nonescape, and local NameCert faithfulness for BEDC.Derived.NeighborhoodFilterUp.

Constructive story: A $\mathsf{NeighborhoodFilter}^{\uparrow}$ packet is generated from a source row, focused point row, finite open-neighbourhood row, FilterBase row, directed-refinement ledger, CauchyFilter compatibility row, metric and topology provenance rows, transport and replay, package provenance, and a local NameCert row.

Chapter 3154

A Concrete Naming Certificate for $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$

Following Busemann and Bridson–Haefliger, the $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ packet records the additive boundary cocycle read by comparing two displayed disk or geodesic positions toward the same boundary point. It sits after the metric-read surface of chapter 3474, the boundary-point carrier of chapter 3732, the visual comparison route of chapter 3866, the geodesic-flow carrier of chapter 3856, and the horocycle packet of chapter 3534. The packet names a finite boundary-point and metric-read cocycle route; it does not complete the hyperbolic boundary, quotient geodesic rays, choose global representatives, or import host curvature analysis.

Definition 3154.1 (HyperbolicBusemannCocycle carrier)**.** *A* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, \Xi, R_0, R_1, G, V, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row on the visible disk or geodesic source.*
2. *Ξ is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-point row toward which the comparison is read.*
3. *R_0 and R_1 are geodesic-ray or disk-position rows with the same displayed boundary endpoint Ξ.*
4. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *transport row used to compare positions along the same boundary-facing route.*
5. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visibility row recording the boundary comparison window.*
6. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *horocycle row that records equal-level surfaces for the boundary point.*
7. *E is the additive cocycle ledger. It stores the signed metric-read difference and the concatenation rule for composable displayed positions.*
8. *H records componentwise* hsame *transport for metric, boundary, ray, flow, visual, horocycle, cocycle, replay, package, and naming rows.*
9. *K records displayed* Cont *replay from a boundary-cocycle request through $M, \Xi, R_0, R_1, G, V, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$ *row.*
The carrier contains no coordinate for a completed ideal-boundary quotient, selected geodesic representative, ambient horosphere foliation, global curvature theorem, or physical observer trajectory.

Lean4 ✓ `BEDC.Derived.HyperbolicBusemannCocycleUp.HyperbolicBusemannCocycleTasteGate single carrier alignment`

Theorem 3154.2 (HyperbolicBusemannCocycle boundary additivity)**.** *For every accepted* HyperbolicBusemannCo *carrier* $B = (M, \Xi, R_0, R_1, G, V, Q, E, H, K, P, N)$*, the additive boundary-cocycle read is exhausted by the visible route*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto R_0, R_1, G, V, Q \longmapsto E, H, K, P, N.$$

If a third displayed position R_2 is admitted by the same boundary row Ξ, the cocycle ledger E records the composite read from R_0 to R_2 as the concatenation of the reads from R_0 to R_1 and from R_1 to R_2, with all equality understood through the displayed hsame *and* Cont *rows of the same packet.*

Proof. Project the carrier of Definition 4134.1. The metric row M and boundary row Ξ are the only source rows from which a Busemann-level comparison can be read.

Read the geometric comparison rows. The positions R_0, R_1, flow row G, visual row V, and horocycle row Q attach the comparison to one displayed boundary endpoint rather than to an ambient ideal-boundary quotient.

The additive statement is the content of the ledger E. Its concatenation rule is transported by H and replayed by K, while P and N package and name the same finite route. Since selected ray representatives, global horosphere foliations, host curvature theorems, and physical observer trajectories have no coordinate in B, the boundary additivity read is confined to the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$, seed)

Scope closed: Definition 4134.1 and Definition 4134.2 seed the finite boundary-point, geodesic-ray, visual-metric, horocycle, additive-ledger, transport, replay, provenance, and NameCert route for the Busemann cocycle.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, completed ideal-boundary quotients, selected geodesic representatives, global horosphere foliations, host curvature analysis, or physical observer dynamics.

Upgrade path: Obligation closure requires checked carrier admission, boundary endpoint coverage, metric-read exactness, horocycle compatibility, additive-ledger concatenation, and local NameCert faithfulness for BEDC.Derived.HyperbolicBusemannCocycleU

Constructive story: A $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ packet is generated from hyperbolic metric, Busemann boundary, geodesic-position, geodesic-flow, visual-metric, horocycle, additive-ledger, transport, replay, package, and local NameCert rows.

Chapter 3155

A Concrete Naming Certificate for $\mathsf{BishopRealConstruction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealConstruction}^{\uparrow}$ packet records the regular-Cauchy construction of the real numbers as finite BEDC NameCert data. It sits over the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the explicit convergence ledger of chapter 332, and the terminal real-name seal of chapter 38. The packet names the human-known Bishop real construction without quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut replacement, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3155.1 (Bishop real construction carrier)**.** *A* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, Q, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, slack comparisons, and endpoint bounds used by every inspected Cauchy window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the finite observation schedule for the rational approximants before any real-name seal is read.*
3. *Q is the regular-Cauchy source row, retaining the named rational stream together with its displayed regularity budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-readback row obtained from D, S, Q, and it is the only row through which rational approximants are exported to later consumers.*
5. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger row, recording the finite modulus entries that witness the regular-Cauchy estimates read by R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the dyadic, stream, regular-readback, and convergence-ledger rows have all been displayed.*
7. *H records componentwise* hsame *transport for dyadic tolerances, finite windows, regular sources, readback rows, convergence ledgers, real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a real-construction request through the same finite route D, S, Q, R, M, E.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopRealConstruction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, R, M, E, H, C, P, N$. It has no coordinate for quotient equality of streams, a selected host limit, countable choice, host real equality, Dedekind-cut replacement, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopRealConstructionUp.BishopRealConstructionTasteGate single carrier alignment`

Theorem 3155.2 (Bishop real construction NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier admission for the dyadic tolerance ledger D, finite stream windows S, regular-Cauchy source Q, regular readback R, convergence ledger M, real seal E, transport H, replay C, provenance P, and local naming N;*
2. *tolerance-window exactness, requiring every accepted read to pair the same dyadic budget in D with the same finite windows in S;*
3. *regular-Cauchy readback, requiring the rational approximants exported by the packet to pass through Q, R before any* $\mathsf{Real}^{\uparrow}$ *consumer reads them;*
4. *convergence-ledger exposure through M, so the construction records finite regularity evidence rather than a hidden host limit;*
5. *seal nonescape through E, which blocks quotient stream equality, selected host limits, host real equality, Dedekind-cut replacement, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
6. *provenance and name exhaustion through P, N together with componentwise transport and replay rows H, C.*

No other NameCert obligation is available from the Bishop real construction packet.

Proof. Project the carrier of Definition 4135.1. The displayed tuple exposes the dyadic tolerance ledger, finite stream windows, regular-Cauchy source, regular rational readback, convergence ledger, real seal, transport, replay, provenance, and local naming rows.

The tolerance-window exactness row is read before any rational approximation is exported. A request first fixes the dyadic budget D and the finite windows S, then reads the regular source Q and the readback row R along that same displayed route.

The convergence ledger M records the finite regularity evidence used by the construction. The real seal E is downstream of D, S, Q, R, M, so a consumer cannot replace those rows by quotient stream equality, a selected host limit, host real equality, Dedekind cuts, countable choice, or ambient completeness.

Finally H, C, P, N transport, replay, source, and name exactly the same finite packet. Since the carrier has no additional coordinate, the six listed rows exhaust the local NameCert surface. □

Theorem 3155.3 (Regular-Cauchy readback route)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

every rational approximant exported by B is read along the displayed route

$$D, S \longrightarrow Q \longrightarrow R \longrightarrow M.$$

In particular, the approximant first uses the dyadic tolerance ledger D and finite stream windows S, then passes through the regular-Cauchy source Q, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the convergence ledger M before it is available to any later row of the packet.*

Proof. Project the carrier of Definition 4135.1. Its only rows capable of naming rational approximants are the finite windows S, the regular-Cauchy source Q, and the regular readback row R, all measured against the dyadic budget D.

Apply Definition 4135.2. The regular-Cauchy readback obligation requires exported rational approximants to pass through Q, R, while the convergence-ledger exposure obligation records the finite regularity evidence in M. Since the classifier compares packets only through $D, S, Q, R, M, E, H, C, P, N$, no alternate approximant coordinate is available. □

Theorem 3155.4 (Real seal handoff)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer reaches the seal row E only after the dyadic ledger D, finite stream windows S, regular-Cauchy source Q, regular readback R, and convergence ledger M have been displayed, and only through the transport, replay, provenance, and local naming rows H, C, P, N.*

Proof. Project the displayed tuple in Definition 4135.1. The seal row E is declared downstream of D, S, Q, R, M, and the remaining public movement of the packet is carried by componentwise hsame transport H, Cont replay C, provenance P, and local naming N.

Apply the seal nonescape and provenance-name exhaustion clauses of Definition 4135.2. They force the $\mathsf{Real}^{\uparrow}$ seal to remain attached to the same finite route, so a consumer can reach E only through the displayed handoff rows rather than through a hidden host-side real object. □

Theorem 3155.5 (Ledger nonescape)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

the packet contains no coordinate for quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut substitution, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Any accepted comparison or consumer read must therefore be witnessed inside the rows $D, S, Q, R, M, E, H, C, P, N$.*

Proof. Project the carrier definition. The classifier is explicitly restricted to the displayed rows $D, S, Q, R, M, E, H, C, P, N$, and the definition states that the excluded principles have no coordinate in the packet.

Apply the seal nonescape clause of Definition 4135.2. It blocks the listed host-side or quotient-side replacements at the NameCert surface. The exhaustion clause then shows that H, C, P, N only transport, replay, source, and name the same finite packet, so they cannot introduce a new equality, limit, choice, cut, or completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealConstruction}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite-window rows, RegSeqRat regular-readback rows, RealModulusOfConvergence ledger rows, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4135.1 and Definition 4135.2 seed the finite Bishop–Bridges regular-Cauchy construction route for Real and completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopRealConstruction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut replacement, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-Cauchy readback, Real seal handoff, convergence-ledger refusal, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealConstructionUp.BishopRealConstruct

Chapter 3156

A Concrete Naming Certificate for $\mathsf{CauchyIntegralFormula}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive complex analysis, the $\mathsf{CauchyIntegralFormula}^{\uparrow}$ packet names the finite contour-evaluation theorem surface that lets a holomorphic local datum be read at an interior point through an explicit contour integral. It sits above the holomorphic predicate of chapter 68, the complex carrier of chapter 39, the complex series modulus rows of chapter 69, the operation socket of chapter 583, and the triangle-to-polygon null-integral route of chapter 2098. The chapter names the Cauchy integral formula as a BEDC NameCert dependency for residue, maximum-modulus, and analytic-continuation consumers; it does not introduce a host contour integral, quotient path space, classical compactness principle, or unrecorded branch-selection oracle.

Definition 3156.1 (Cauchy integral formula carrier)**.** *A* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, Z, R, H, C, S, G, L, T, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Holomorphic}^{\uparrow}$ *domain row, displaying the holomorphic function, its complex neighbourhood, and the interior point whose value is to be read.*
2. *Z is the* $\mathsf{Complex}^{\uparrow}$ *point row, carrying the point coordinate and the complex arithmetic rows used by the evaluation denominator.*
3. *R is the radius and contour row, naming a rational circle or polygonal contour around Z whose image remains inside the displayed holomorphic neighbourhood.*
4. *H is the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row, exposing the piecewise-linear contour, holomorphic input, Riemann-sum modulus, complex output, operation-law ledger, transport, replay, provenance, and local naming rows.*
5. *C is the* $\mathsf{CplxSer}^{\uparrow}$ *local expansion row, recording the finite read of the convergent power-series data needed to identify the residue coefficient at the point.*
6. *S is the singular-kernel exclusion row, displaying that the contour avoids the evaluation point in the denominator while the point remains inside the contour.*
7. *G is the* $\mathsf{Goursat}^{\uparrow}$ *null-triangle ledger row, routing contour deformation and polygon subdivision through the already named null-integral packet.*

8. *L is the formula ledger row, recording the equality between the point value and the contour integral of the Cauchy kernel over the displayed contour.*
9. *T records componentwise* hsame *transport for the holomorphic, complex, contour, socket, series, exclusion, Goursat, formula, replay, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyIntegralFormula}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{CplxSer}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$ $\mathsf{Goursat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, Z, R, H, C, S, G, L, T, P, N$. It has no coordinate for a host analytic theorem, selected branch, quotient path equality, or classical integral representation outside the displayed contour socket.

Lean4 ✓ `BEDC.Derived.CauchyIntegralFormulaUp.CauchyIntegralFormulaTasteGate single carrier alignment`

Theorem 3156.2 (Cauchy integral formula NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier $F = (D, Z, R, H, C, S, G, L, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$ *surface consists exactly of holomorphic-domain admission through D, complex point and arithmetic admission through Z, contour containment through R, contour-socket admission through H, complex-series local expansion through C, singular-kernel exclusion through S, Goursat deformation through G, formula-ledger exactness through L, componentwise transport through T, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyIntegralFormulaUp.CauchyIntegralFormulaCarrier namecert obligations`

Proof. Project the carrier. The holomorphic source D and point row Z fix the only domain and evaluation data that the packet may read.

The contour row R and socket row H provide the finite operation surface. They determine the contour-integral value through displayed piecewise-linear, Riemann-sum, complex-output, and operation-law rows.

The series row C, exclusion row S, and Goursat row G identify the Cauchy kernel computation and the allowed contour deformations. The formula ledger L then records the point-evaluation equality inside the same packet.

Finally T, P, N transport, source, and name the same finite route. Since no carrier coordinate stores a host contour theorem, branch selection, quotient path equality, or classical integral representation, the listed rows exhaust the NameCert obligations. □

Theorem 3156.3 (Cauchy integral formula contour-socket route)**.** *Every point-evaluation read exported by an accepted* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier factors through the finite route*

$$D, Z, R \longmapsto H \longmapsto C, S, G, L \longmapsto T, P, N.$$

In particular, the read must pass through the $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row before the formula ledger can be consumed, and it cannot bypass the contour socket by citing a host Cauchy theorem, quotient path identity, selected branch, or undisplayed completeness object.*

Proof. Start with a point-evaluation request. The rows D, Z, R display the holomorphic input, complex point, and contour containment data, but none of them alone stores the contour integral or the formula equality.

The request must next pass through H. By chapter 583, the socket exposes the contour, input, modulus, complex output, operation ledger, transport, replay, provenance, and local naming rows needed by a contour-integral consumer.

The rows C, S, G, L then attach local series computation, denominator exclusion, Goursat deformation, and formula equality to that same socket route. Transport and provenance through T, P, N preserve the displayed route without adding a second source of analytic equality.

Therefore every exported Cauchy integral formula read is socket-mediated and finite-row visible. The absent host theorem, quotient path identity, branch selection, and completeness oracle have no coordinate in the carrier. □

Falsifiable prediction: Every residue, maximum-modulus, or analytic-continuation consumer that cites $\mathsf{CauchyIntegralFormula}^{\uparrow}$ must expose holomorphic-domain data, complex point data, contour containment, contour-socket rows, local complex-series data, singular-kernel exclusion, Goursat deformation, formula ledger, componentwise hsame transport, Pkg provenance, and local NameCert evidence.

Independence witness: holomorphic, complex-series, contourintegralsocket, goursat: none is bijective with $\mathsf{CauchyIntegralFormula}^{\uparrow}$ because this packet jointly carries point-evaluation data, contour containment, socket-mediated integral rows, kernel exclusion, deformation ledger, formula equality, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntegralFormula}^{\uparrow}$ packet is generated from holomorphic-domain rows, complex point rows, contour-containment rows, contour-socket rows, local complex-series rows, singular-kernel exclusion rows, Goursat deformation rows, formula-ledger rows, componentwise hsame transport, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4136.1, Definition 4136.2, and Definition 4136.3 seed the finite contour-mediated point-evaluation packet over holomorphic, complex, complex-series, contour-socket, Goursat, transport, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residue calculus, maximum-modulus consequences, host contour integration, quotient path spaces, global analytic continuation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, holomorphic source lock, contour containment, contour-socket handoff, local series exactness, singular-kernel exclusion, Goursat deformation, formula-ledger exactness, and non-escape for BEDC.Derived.CauchyIntegralFormulaUp.

Chapter 3157

A Concrete Naming Certificate for $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$

The $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet realizes the zero-marker support-nerve sphere core of Definition 9462.1665 and the finite count table of Definition 9462.1668 as finite BEDC NameCert data. It sits beside the support-nerve realization of chapter 4079, the support-nerve census of chapter 4048, the codon lift packet of chapter 4129, and the boundary-tile packet of chapter 3393. The packet names the quotient trigger, complement simplex cover, boundary-simplex nerve, reduced homology degree, finite count table, TasteGate admission boundary, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic selection claim, or unbounded enumeration oracle.

Definition 3157.1 (Minimal trigger homology-core carrier)**.** *A* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, F, N, R, B, K, T, H, C, P, L)$$

with the following displayed rows:

1. *Q is the quotient-level minimal full-trigger row, displaying $A = \{S_1, \ldots, S_m\}$ and its finite support cells.*
2. *F is the complement-simplex row, recording $F_i = \Omega \setminus S_i$ and the finite simplices 2^{F_i}.*
3. *N is the support-nerve row, identifying the union $\bigcup_i 2^{F_i}$ as the displayed support nerve.*
4. *R is the boundary-simplex route row, recording that the cover nerve is $\partial\Delta^{m-1}$ because every proper complement intersection is inhabited and the full intersection is empty.*
5. *B is the reduced-homology row, naming the sphere type S^{m-2} and the corresponding one-dimensional reduced homology witness.*
6. *K is the finite count row, displaying the quotient and codon count table for trigger sizes $1, 2, 3, 4$.*
7. *T is the TasteGate obligation row, requiring later* $\mathsf{Nontrivial\ MinimalTriggerHomologyCore}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ MinimalTriggerHomologyCore}^{\uparrow}$ *instances to preserve Q, F, N, R, B, K before downstream consumers may use the packet as a stable AI-origin chapter surface.*
8. *H records componentwise* hsame *transport for quotient triggers, complement simplices, support nerves, boundary-simplex routes, homology rows, count rows, TasteGate rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a quotient trigger through complement simplices, support nerves, boundary-simplex routing, homology readback, and finite count exposure.*

10. *P is the* Pkg *provenance over* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$*, Definition 9462.1665, Definition 9462.1668,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, N, R, B, K, T, H, C, P, L$. It has no coordinate for empirical biology, stochastic fitness, a host simplicial-complex library, or an unrestricted search over supports.

Theorem 3157.2 (Minimal trigger homology-core NameCert obligations)**.** *For every accepted* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier $M = (Q, F, N, R, B, K, T, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyC}}$ *surface consists exactly of quotient-trigger admission through Q, complement-simplex admission through F, support-nerve construction through N, boundary-simplex routing through R, reduced-homology readback through B, finite count preservation through K, TasteGate obligation exposure through T, componentwise transport through H, replay through C, provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.MinimalTriggerHomologyCoreUp.MinimalTriggerHomologyCoreNamecertObligations`

Proof. Project the carrier. The quotient trigger row Q fixes the finite support family, and the complement row F records the finite simplices that generate the support nerve.

The support-nerve row N and boundary-simplex route R encode the nerve-theorem calculation from Definition 9462.1665. The reduced-homology row B reads the displayed sphere degree, and the count row K records the quotient and codon tables from Definition 9462.1668.

The TasteGate row T is an explicit downstream obligation: later Nontrivial and FieldFaithful instances must preserve the carrier rows, but those instances are not hidden assumptions in the present packet. The rows H, C, P, L transport, replay, source, and name the same finite data. □

Theorem 3157.3 (Minimal trigger homology-core support-nerve route)**.** *Every homology-core read exported by an accepted* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier factors through the finite route*

$$Q, F \longmapsto N, R \longmapsto B, K \longmapsto T, H, C, P, L.$$

The route requires the quotient minimal-trigger row and complement-simplex row before the support nerve, boundary-simplex route, homology witness, count table, TasteGate obligation, transport, replay, provenance, and local naming rows can be consumed.

Proof. Start with a homology-core read. The row Q displays the quotient-level minimal full trigger, while F displays its complement simplices; neither row by itself stores a homology conclusion.

The read must pass through N and R. These rows expose the support nerve and the boundary-simplex route that turns minimal full-trigger data into the sphere core.

The rows B and K then read the reduced homology degree and the finite count table. The TasteGate row T keeps the future nontriviality and field-faithfulness requirements attached to the same carrier rather than replacing the packet by a detached classifier.

Finally H, C, P, L preserve the route by componentwise transport, replay, provenance, and local naming. Since the carrier has no row for a host simplicial-complex library, stochastic fitness model, or unbounded support search, every exported read is exhausted by the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet exposes quotient-trigger data, complement-simplex data, support-nerve data, boundary-simplex routing, reduced homology readback, finite count rows, TasteGate obligations, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any downstream trigger-lattice consumer may cite the sphere-core result.

Independence witness: support-nerve-realization, support-nerve-census, minimal-trigger-lift-bundle, dna-window-six-boundary-tile: none is bijective with $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ because this packet jointly carries quotient minimal-trigger rows, complement simplices, support-nerve routing, reduced homology, finite counts, TasteGate obligations, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet is generated from quotient minimal-trigger rows, complement-simplex rows, support-nerve rows, boundary-simplex route rows, reduced-homology rows, finite count rows, TasteGate obligation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4137.1, Definition 4137.2, and Definition 4137.3 seed the finite support-nerve sphere-core packet over quotient triggers, complement simplices, boundary-simplex routes, reduced homology, count rows, TasteGate obligations, transport, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, host simplicial-complex libraries, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient-trigger admission, complement-simplex admission, support-nerve construction, boundary-simplex route exactness, homology readback, finite count preservation, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MinimalTriggerHomologyCoreUp.

Chapter 3158

A Concrete Naming Certificate for $\mathsf{CompactOpenTopology}^{\uparrow}$

Following Kelley and Bourbaki on compact-open topologies, the $\mathsf{CompactOpenTopology}^{\uparrow}$ packet names the finite subbasic topology surface carried by compact-domain windows and open-target tests. It sits next to the compact-open metric packet of chapter 1903, the compact-open function-space carrier of chapter 2221, the continuous-map graph route of chapter 130, the compact-metric source surface of chapter 129, and the compact-window convergence route of chapter 4022. The packet records compact-domain rows, open-target rows, finite window rows, componentwise transport, displayed replay, provenance, and local naming; it does not install a host topology on all functions.

Definition 3158.1 (Compact-open topology carrier)**.** *A* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, O, W, F, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-domain row. It records the retained* $\mathsf{CompactMetric}^{\uparrow}$ *source windows and the compact subobjects whose images are tested.*
2. *O is the open-target row. It records the displayed target-open tests used by the subbasic condition.*
3. *W is the finite window row. Each entry pairs one compact-domain entry from K with one open-target entry from O.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose displayed values are inspected only on the windows listed by W.*
5. *S is the subbasic membership ledger. It records that the displayed map sends the retained compact-domain window into the corresponding open-target row.*
6. *H records componentwise* hsame *transport for compact-domain rows, open-target rows, finite windows, continuous-map graph rows, subbasic membership ledgers, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a compact-open topology request through compact-domain inspection, open-target inspection, graph evaluation, and subbasic membership.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactOpenMetric}^{\uparrow}$, $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyUniformConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, W, F, S, H, C, P, N$. There is no coordinate for open-cover compactness, choice-selected compact witnesses, quotient equality of host functions, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactOpenTopologyUp.CompactOpenTopologyTasteGate single carrier alignment`

Theorem 3158.2 (Compact-open topology NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier $T = (K, O, W, F, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for compact-domain rows, open-target rows, finite windows, continuous-map graph rows, subbasic membership ledgers, transport, replay, provenance, and local naming;*
2. *compact-domain visibility, requiring every domain side of a subbasic request to read through the retained* $\mathsf{CompactMetric}^{\uparrow}$ *rows in K;*
3. *open-target visibility, requiring every target side of a subbasic request to read through the displayed open-target row O;*
4. *finite-window stability, requiring membership checks to use the paired rows in W and the subbasic ledger S before any consumer reads the topology packet;*
5. *non-escape, requiring H, C, P, N to preserve the same displayed rows and to export no open-cover compactness, choice-selected compact witness, host function-space equality, or ambient completeness.*

No compact-open topology obligation is stored outside those rows.

Proof. Project the carrier in Definition 4138.1. The visible topology coordinates are exactly the compact-domain row, open-target row, finite window row, continuous-map graph row, subbasic membership ledger, and the structural rows H, C, P, N.

The compact-domain side is forced by K. Its reads are the same finite compact-source surface used by chapter 129 and by the compact-open metric route of chapter 1903.

The function side is forced by F and S. A subbasic membership request must inspect displayed graph values through chapter 130 and then record the result in the finite membership ledger attached to the paired window in W.

The provenance row places the packet beside chapter 2221 and chapter 4022. Those siblings may consume the displayed subbasic surface, but the carrier has no row for host function-space equality, open-cover compactness, choice-selected compact witnesses, or ambient completeness. □

Theorem 3158.3 (Compact-open topology finite subbase stability)**.** *Let $T = (K, O, W, F, S, H, C, P, N)$ be an accepted* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier. Every finite family of displayed compact-open subbasic requests admitted through W remains a request on the same compact-domain, open-target, continuous-map, and membership rows after componentwise* hsame *transport by H and displayed* Cont *replay by C. Thus finite subbase stability is internal to $K, O, W, F, S, H, C, P, N$ and cannot be replaced by open-cover compactness, choice-selected compact witnesses, host function-space equality, or ambient completeness.*

Proof. Start with a finite list of subbasic requests. Each request is already an entry of W, so it names one compact-domain row from K and one open-target row from O before it can inspect a function.

Read the continuous-map data through F and the membership row through S. The compact-open metric and compact-open function-space siblings may reuse this windowed read, but they receive only the displayed graph and membership coordinates.

Transport by H is componentwise on the listed rows. Replay by C repeats the same compact-domain, open-target, graph, and membership route, so finite intersections of displayed subbasic requests remain inside the same carrier.

Since the carrier has no row for open-cover compactness, selected compact witnesses, host function equality, or ambient completeness, none of those principles can be introduced by finite subbase stability. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenTopology}^{\uparrow}$ packet is generated from compact-domain rows, open-target rows, finite compact-open windows, ContinuousMap graph rows, subbasic membership ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4138.1, Definition 4138.2, and Definition 4138.3 seed the finite subbasic compact-open topology surface for compact-domain and open-target requests.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactOpenTopologyUp.CompactOpenTopologyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected compact witnesses, host function-space equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires binding compact-domain visibility, open-target visibility, finite-window stability, graph membership, componentwise transport, replay, provenance, and local naming explicitly to the finite carrier rows.

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ hub reads pointwise stream windows through chapter 126, the metric comparison route through chapter 57, and terminal real seals through chapter 38.

Chapter 3159

A Concrete Naming Certificate for $\mathsf{PointwiseStreamMetric}^{\uparrow}$

Following Bishop and Bridges' constructive metric-space treatment, the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet records the standard pointwise distance interface for certified streams as finite BEDC data. It sits after the stream-name surface of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric-space carrier discipline of chapter 57. The packet exposes pointwise metric data for completion and continuous-map consumers without host functions, quotient stream equality, selected representatives, or choice.

Definition 3159.1 (Pointwise stream metric carrier)**.** *A* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the certified stream source row, read through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface.*
2. *W records the finite pointwise window schedule used when two certified streams are compared at a displayed index.*
3. *D is the dyadic tolerance and triangle ledger governing the finite comparisons of pointwise observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the finite rational comparison stream into a regular rational name.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the distance value obtained from the displayed pointwise route.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *metric-structure row, used only after the pointwise stream route has displayed S, W, D, R, E.*
7. *H stores componentwise* hsame *transport for stream names, window schedules, dyadic ledgers, regular readbacks, real seals, metric rows, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a consumer request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these coordinates. It contains no host function equality, quotient stream object, selected stream representative, countable-choice schedule, or ambient completeness assertion.

Theorem 3159.2 (Pointwise stream metric pointwise classifier)**.** *For an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier* $Q = (S, W, D, R, E, M, H, C, P, N)$*, every pointwise distance classifier read factors through the certified stream source* S*, the finite window schedule* W*, the dyadic tolerance ledger* D*, the regular-rational readback* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before the* $\mathsf{Met}^{\uparrow}$ *row* M *may be inspected. The classifier therefore names a pointwise metric route on certified streams and does not read host functions, quotient streams, selected representatives, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.PointwiseStreamMetricUp.PointwiseStreamMetricPointwiseClassifier`

Proof. Project the carrier of Definition 4139.1. The source stream, window schedule, dyadic ledger, regular readback, Real seal, metric row, transport, replay, provenance, and local naming row are all displayed coordinates.

The pointwise route begins with S and W. A consumer must name the stream source and the finite observation windows before any rational readback or metric row can be inspected.

The distance read then passes through D, R, E. The dyadic ledger bounds the finite comparisons, the regular-rational row gives the readback discipline, and the Real seal is the terminal distance value. Since no coordinate stores a host function or quotient stream, the classifier cannot export one. □

Theorem 3159.3 (Pointwise stream metric constant stream)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier whose stream source row* S *is displayed as a constant certified stream. Then every distance read between two displayed copies of that source is routed through the same finite window schedule* W*, dyadic zero-tolerance ledger* D*, regular-rational readback* R*, terminal Real seal* E*, and metric row* M*. The packet records the constant-stream zero-distance route as displayed NameCert data, not as host equality of functions or quotient stream identity.*

Proof. Start with the displayed constant stream row in S. Because the stream is certified by its $\mathsf{StreamName}^{\uparrow}$ windows, the two copies are compared only through the finite schedule W.

Use the dyadic ledger D for the zero-tolerance comparison. The readback row R and the terminal seal E record the regular-rational and Real distance route generated by that finite ledger.

Finally inspect M only after the pointwise route has been displayed. The metric row reads the zero-distance route as finite certificate data, while host function equality and quotient stream identity are absent from the carrier. □

Theorem 3159.4 (Pointwise stream metric reindexing stability)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any accepted finite reindexing of the certified stream windows transports the pointwise metric classifier only through the displayed* hsame *rows* H *and the replay row* C*: the reindexed route must preserve the stream source row, window schedule, dyadic ledger, regular-rational readback, Real seal, metric row, provenance, and local naming row. It cannot add a hidden stream quotient, selected representative, or host-level function extensionality principle.*

Proof. Read the transport coordinate H. It is componentwise: stream sources, window schedules, dyadic ledgers, readback rows, Real seals, metric rows, replay, provenance, and naming rows are transported as displayed rows.

Replay the reindexed request with C. Since C follows the same order $S, W, D, R, E, M, H, P, N$, the pointwise classifier is preserved only by the finite route already present in the carrier.

The exclusions are an inventory check. No coordinate of Q stores a stream quotient, selected representative, or host-level function extensionality principle, so reindexing stability cannot export such data. □

Theorem 3159.5 (Pointwise stream metric NameCert obligations)**.** *For an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier* $Q = (S, W, D, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the certified stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, Real seal, metric row, componentwise transport, continuation replay, provenance, and local naming row; pointwise classifier exposure; constant-stream zero-distance exposure; reindexing stability; and ledger non-escape for host functions, quotient streams, selected representatives, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 4139.1. The accepted packet displays exactly the stream, window, tolerance, readback, Real, metric, transport, replay, provenance, and naming rows.

Pointwise classifier exposure is Definition 4139.2. Constant-stream zero-distance exposure is Definition 4139.3. Reindexing stability is Definition 4139.4.

The ledger non-escape clause follows from the same carrier inventory. Since no row stores host functions, quotient streams, selected representatives, countable-choice schedules, or ambient completeness, the local NameCert surface is exhausted by the displayed finite rows. □

Theorem 3159.6 (Pointwise stream metric sibling route)**.** *Every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer reaches pointwise distance data only after the certified* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{Met}^{\uparrow}$ *carrier rows named in the introduction have been displayed. The pointwise metric row is therefore a sibling route through the finite stream, regular-rational, Real-seal, and metric surfaces, not a host function metric, quotient stream comparison, selected representative, countable-choice schedule, or ambient completeness assertion.*

Proof. Apply Definition 4139.5. The obligation surface lists carrier admission for the certified stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, Real seal, metric row, componentwise transport, continuation replay, provenance, and local naming row.

The opening paragraph identifies the sibling chapters that supply those rows: chapter 126, chapter 301, chapter 38, and chapter 57. Reading the pointwise distance surface therefore requires the stream window before the regular readback, the regular readback before the Real seal, and the Real seal before the metric row may be used.

Since the same obligation theorem excludes host functions, quotient streams, selected representatives, countable choice, and ambient completeness, the sibling route is exhausted by the displayed finite packet. □

Theorem 3159.7 (Pointwise stream metric Real-completion anchor)**.** *Every completion-facing read of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the displayed* $\mathsf{Met}^{\uparrow}$ *row before it reaches any Real-completion consumer. The route exports the finite pointwise metric packet and excludes host functions, quotient streams, selected representatives, countable choice, and ambient completeness.*

Proof. Start with the pointwise classifier of Definition 4139.2. It forces a distance read to pass through the certified stream source, finite window schedule, dyadic tolerance ledger, regular-rational readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before the metric row is inspected.

Use Definition 4139.4 to transport any completion-facing reindexing request. The transport coordinate H and replay coordinate C preserve the same stream windows, dyadic ledger, readback row, Real seal, metric row, provenance, and local naming row.

Apply Definition 4139.5 and then Definition 4139.6. The NameCert obligations exhaust the accepted packet, and the sibling route identifies the supplying chapters for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$; the dyadic ledger is already a displayed coordinate of the carrier and obligations.

Therefore a Real-completion consumer receives only the finite route

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow}.$$

Since none of the cited surfaces stores host functions, quotient streams, selected representatives, countable-choice schedules, or ambient completeness, the completion-facing read cannot acquire them. □

Theorem 3159.8 (Pointwise stream metric dossier lattice route)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must read the same finite pointwise window W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and displayed* $\mathsf{Met}^{\uparrow}$ *row M before any completion-facing route is consumed. The downstream lattice is therefore the convergence-ledger route of chapter 2074 together with the completion route of chapter 1506, not an ambient completeness theorem, host function equality, quotient stream equality, selected representative, or countable-choice schedule.*

Proof. Project the carrier of Definition 4139.1. The only rows available before the structural data are the certified stream source S, finite window schedule W, dyadic ledger D, regular-rational readback R, terminal Real seal E, and metric row M.

Apply the Real-completion anchor Definition 4139.7. It already forces every completion-facing read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the displayed $\mathsf{Met}^{\uparrow}$ row before a consumer receives the pointwise metric packet.

Now split the downstream request by destination. A $\mathsf{RealModulusOfConvergence}^{\uparrow}$ consumer reads the convergence ledger only through chapter 2074, so the same W, D, R, E, M rows remain visible before convergence data is public. A $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads the completion route only through chapter 1506, so the same finite pointwise dossier is still the source of completion-facing distance data.

Because the carrier lacks coordinates for ambient completeness, host function equality, quotient stream equality, selected representatives, and countable-choice schedules, neither downstream route can acquire those objects from $\mathsf{PointwiseStreamMetric}^{\uparrow}$. □

The finite-window side of this lattice is consumed by chapter 5916 through its pointwise metric forward route. Thus a downstream $\mathsf{FiniteStream}^{\uparrow}$ reader receives the same $S, W, D, R, E, M, H, C, P, N$ packet as a finite-window metric read, not as an added completion principle or a hidden stream quotient.

The downstream completion lattice consumes this anchor through the convergence-ledger route of chapter 2074. Every $\mathsf{RealModulusOfConvergence}^{\uparrow}$ consumer of an accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet must read the same finite windows W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, terminal $\mathsf{Real}^{\uparrow}$ seal E, and displayed metric row M before any convergence ledger is public. The consumer therefore receives the pointwise stream metric only as the finite route already exposed by the carrier, not as a detached completion principle.

Theorem 3159.9 (Pointwise stream metric finite-window consumption)**.** *Every metric-distance read from an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet first consumes the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{Met}^{\uparrow}$ *distance row. The consumed route is the finite packet*

$$S, W, D, R, E, M, H, C, P, N$$

of Definition 4139.1, with the dyadic ledger D bounding the window comparison before the Real seal and metric row are read.

The route is invariant under accepted scheduled window replacement transported by H and replayed by C. It exposes no quotient stream equality, selected stream representative, host function equality, countable-choice schedule, or ambient completeness principle.

Proof. Project the carrier of Definition 4139.1. The metric-distance read has only the stream source, finite window schedule, dyadic ledger, regular readback, Real seal, metric row, transport, replay, provenance, and local naming coordinates available.

Apply Definition 4139.6. It orders the reader through the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and then the $\mathsf{Met}^{\uparrow}$ row.

Use Definition 4139.7 to connect the same route to completion-facing consumers. The dyadic ledger remains a displayed coordinate of the packet, so the read cannot bypass the finite window and regular readback.

Finally invoke Definition 4139.4. Any scheduled replacement transported by H and replayed by C preserves the same stream windows, dyadic ledger, readback, Real seal, metric row, provenance, and local naming row. Since the carrier has no quotient stream, selected representative, host function equality, choice schedule, or ambient completeness coordinate, the finite-window consumption route is exhausted by the displayed packet. □

Theorem 3159.10 (Pointwise stream metric completion separation)**.** *Every completion-facing read of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must expose $S, W, D, R, E, M, H, C, P, N$ before a completion consumer may inspect the metric-distance row. The read cannot infer ambient completeness, quotient stream equality, selected representatives, countable-choice schedules, or host function equality unless the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are present in the same packet.*

Proof. Project the carrier of Definition 4139.1. The only available rows are the certified stream source, finite window schedule, dyadic tolerance ledger, regular-rational readback, terminal Real seal, metric row, transport, replay, provenance, and local naming row.

Apply the pointwise classifier theorem Definition 4139.2. It forces the distance read through S, W, D, R, E before M is inspected, so a completion consumer cannot start from an ambient completed stream space.

Use the Real-completion anchor Definition 4139.7 and the finite-window consumption theorem Definition 4139.9. Together they place every completion-facing request on the displayed stream-window, dyadic-ledger, RegSeqRat readback, Real-seal, and metric route.

The conclusion is the carrier inventory. Since no coordinate stores ambient completeness, quotient stream equality, selected representatives, countable choice, or host function equality, the completion-facing read can use such data only by leaving the accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet. □

Theorem 3159.11 (Pointwise stream metric synchronized-window triangle route)**.** *Let $Q_{\mathrm{xy}}, Q_{\mathrm{yz}}, Q_{\mathrm{xz}}$ be accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carriers for the displayed stream reads from x to y, from y to z, and from x to z. Suppose the three packets share the same certified* $\mathsf{StreamName}^{\uparrow}$ *window schedule W and the same displayed* $\mathsf{Met}^{\uparrow}$ *row M:*

$$\begin{aligned} Q_{\mathrm{xy}} &= (S_{\mathrm{xy}}, W, D_{\mathrm{xy}}, R_{\mathrm{xy}}, E_{\mathrm{xy}}, M, H_{\mathrm{xy}}, C_{\mathrm{xy}}, P_{\mathrm{xy}}, N_{\mathrm{xy}}), \\ Q_{\mathrm{yz}} &= (S_{\mathrm{yz}}, W, D_{\mathrm{yz}}, R_{\mathrm{yz}}, E_{\mathrm{yz}}, M, H_{\mathrm{yz}}, C_{\mathrm{yz}}, P_{\mathrm{yz}}, N_{\mathrm{yz}}), \\ Q_{\mathrm{xz}} &= (S_{\mathrm{xz}}, W, D_{\mathrm{xz}}, R_{\mathrm{xz}}, E_{\mathrm{xz}}, M, H_{\mathrm{xz}}, C_{\mathrm{xz}}, P_{\mathrm{xz}}, N_{\mathrm{xz}}). \end{aligned}$$

Then any two-leg distance read from x to z through y must consume, in the same synchronized order, the shared finite window W, the dyadic triangle ledgers $D_{\mathrm{xy}}, D_{\mathrm{yz}}, D_{\mathrm{xz}}$, the $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks $R_{\mathrm{xy}}, R_{\mathrm{yz}}, R_{\mathrm{xz}}$, the terminal* $\mathsf{Real}^{\uparrow}$ *seals $E_{\mathrm{xy}}, E_{\mathrm{yz}}, E_{\mathrm{xz}}$, and the displayed metric row M before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer may use the composed distance. The route is a finite BEDC triangle ledger and not an ambient metric-space triangle proof over host functions, quotient streams, selected representatives, or countable-choice schedules.*

Proof. Project the three carriers. Each packet displays its stream source row, the shared finite window schedule, its dyadic ledger, its regular-rational readback, its Real seal, the shared metric row, transport, replay, provenance, and local naming row.

Apply the pointwise classifier theorem, Definition 4139.2, to the three legs. For Q_{xy}, Q_{yz}, and Q_{xz} the distance read must pass through the displayed window, dyadic ledger, readback, and Real seal before M is inspected.

Use finite-window consumption, Definition 4139.9, for each leg. The two-leg read may compose the x to y and y to z reads only after both have exposed the same window W, and it may compare them with the x to z read only through the displayed dyadic triangle ledger and readback rows.

Apply reindexing stability, Definition 4139.4, to any transported schedule request. The H_{xy}, H_{yz}, and H_{xz} rows, replayed by the corresponding C rows, preserve the shared window and metric row while keeping the three dyadic ledgers, readbacks, Real seals, provenance rows, and local names visible.

Finally invoke completion separation, Definition 4139.10. A completion or modulus consumer can use the composed distance only after the synchronized finite route has displayed all three pointwise packets. Since none of the carriers stores ambient completeness, quotient stream equality, selected representatives, host function equality, or countable choice, the triangle route remains a BEDC finite-window route. □

Theorem 3159.12 (Pointwise stream metric finite-window triangle budget)**.** *Let $Q_{\mathrm{xy}}, Q_{\mathrm{yz}}, Q_{\mathrm{xz}}$ be accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packets sharing a* $\mathsf{StreamName}^{\uparrow}$ *window schedule W and a displayed* $\mathsf{Met}^{\uparrow}$ *row M. A completion-facing triangle-budget read for the three legs may export data only through the three displayed dyadic ledgers $D_{\mathrm{xy}}, D_{\mathrm{yz}}, D_{\mathrm{xz}}$, the three* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks $R_{\mathrm{xy}}, R_{\mathrm{yz}}, R_{\mathrm{xz}}$, the three terminal* $\mathsf{Real}^{\uparrow}$ *seals $E_{\mathrm{xy}}, E_{\mathrm{yz}}, E_{\mathrm{xz}}$, and the shared metric row M, with the source, transport, replay, provenance, and local naming rows of each packet still visible. The budget route has no coordinate for ambient completeness, quotient stream equality, selected representative, host function equality, or countable choice.*

Proof. Apply the synchronized-window triangle route, Definition 4139.11. It already forces the two-leg comparison and the direct comparison to share the same finite window W and metric row M before any completion-facing consumer can read the composed distance.

Use finite-window consumption, Definition 4139.9, on each of the three packets. This exposes the dyadic ledger, regular-rational readback, and Real seal of each leg as displayed finite-window data rather than as an ambient metric-space fact.

Finally apply the NameCert obligations, Definition 4139.5. They keep the source rows, componentwise hsame transports, displayed Cont replays, Pkg provenance, and local names in the exported packet. Since the carrier contains none of the excluded quotient, representative, host-equality, completeness, or choice coordinates, the triangle budget is exactly the listed finite BEDC route. □

Theorem 3159.13 (Pointwise stream metric source separation)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* PointwiseStreamMetric$^{\uparrow}$ *carrier. If two completion-facing reads agree on the metric row* M *but differ in any of the source rows* S, W, D, R, E*, then the packet cannot identify them by* PointwiseStreamMetric$^{\uparrow}$*. Source separation requires the same* StreamName$^{\uparrow}$ *finite window, dyadic ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, and* Met$^{\uparrow}$ *row; no quotient stream equality or host function equality can collapse different source rows.*

Proof. Project the carrier of Definition 4139.1. A completion-facing read exposes the ordered packet $S, W, D, R, E, M, H, C, P, N$, so agreement of the metric row M alone does not determine the source route.

Apply the pointwise classifier theorem Definition 4139.2. It places the certified stream source, finite window schedule, dyadic ledger, regular readback, and Real seal before the metric row is inspected.

Use finite-window consumption, Definition 4139.9, and completion separation, Definition 4139.10. Together they require a completion-facing metric read to keep the same S, W, D, R, E rows in the packet, not merely the same metric row.

Finally apply reindexing stability, Definition 4139.4. Transport by H and replay by C preserve displayed rows componentwise, and the carrier has no quotient stream or host function equality coordinate. Therefore two reads differing in any of S, W, D, R, E remain source-separated inside PointwiseStreamMetric$^{\uparrow}$. □

Theorem 3159.14 (Pointwise stream metric modulus ledger pullback)**.** *Every* RealModulusOfConvergence$^{\uparrow}$ *consumer of an accepted* PointwiseStreamMetric$^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must factor through the displayed certified stream source S*, finite pointwise window* W*, dyadic ledger* D*,* RegSeqRat$^{\uparrow}$ *readback* R*, terminal* Real$^{\uparrow}$ *seal* E*, and metric row* M *before it may read convergence data. The pullback reaches the convergence ledger only through the sibling route of chapter 2074; it exports no detached modulus, quotient stream equality, selected representative, countable-choice schedule, host function equality, or ambient completion principle.*

Proof. Project the carrier of Definition 4139.1. The only completion-facing analytic rows available before the structural data are S, W, D, R, E, M.

Apply the NameCert obligations, Definition 4139.5, together with completion separation, Definition 4139.10. These force any completion consumer to expose the certified stream source, finite windows, dyadic ledger, regular-rational readback, Real seal, metric row, transport, replay, provenance, and local naming before a metric-distance read is public.

Now use the sibling RealModulusOfConvergence$^{\uparrow}$ chapter. Its carrier and route theorems read convergence-modulus data only after the finite StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ tolerance ledger, RegSeqRat$^{\uparrow}$ readback, and terminal Real$^{\uparrow}$ seal are displayed. Therefore the pointwise metric contribution enters the convergence ledger through the same S, W, D, R, E, M route.

Since neither carrier stores a detached modulus selector, quotient stream equality, selected representative, choice schedule, host function equality, or ambient completion theorem, the pullback is exhausted by the displayed rows. □

Theorem 3159.15 (Pointwise stream metric StreamName dependency)**.** *Every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet consumes its pointwise metric rows only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window route of chapter 126 has been displayed. Thus the metric comparison is a pointwise stream-window read through*

$$S, W, D, R, E, M, H, C, P, N$$

and not an external metric-space equality, quotient stream comparison, selected representative, countable-choice schedule, or quotient completion.

Proof. Project the carrier of Definition 4139.1. The source row S and window row W precede the dyadic ledger, regular readback, Real seal, and metric row in the accepted packet.

Use finite-window consumption, Definition 4139.9. It forces every completion-facing metric read to expose the certified $\mathsf{StreamName}^{\uparrow}$ window before any pointwise distance row can be consumed.

Apply synchronized-window triangle routing, Definition 4139.11, and source separation, Definition 4139.13. Together they make the metric comparison depend on the same finite stream-window route rather than on ambient metric equality or a quotient of stream sources.

Finally use modulus ledger pullback, Definition 4139.14. Any convergence-facing consumer reads the pointwise metric contribution through the displayed StreamName, DyadicRatCore, RegSeqRat, Real, and MetricSpace rows, so the dependency edge to chapter 126 is part of the finite BEDC ledger. □

Theorem 3159.16 (Pointwise stream metric regular Cauchy bridge)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any metric comparison over synchronized stream windows that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *distance seal* E *must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row* R *before the* $\mathsf{Met}^{\uparrow}$ *comparison row* M *can be consumed. Thus the comparison is routed through stream windows, dyadic ledgers, regular Cauchy readback, and the Real seal, not through ambient metric equality, quotient stream equality, selected representatives, countable choice, or host function equality.*

Proof. Start with finite-window consumption, Definition 4139.9. A distance read consumes the certified stream window, dyadic ledger, regular readback, Real seal, and metric row as the displayed packet $S, W, D, R, E, M, H, C, P, N$.

Apply synchronized-window triangle routing, Definition 4139.11. In a synchronized comparison, each leg keeps the shared window schedule visible while its dyadic ledger and readback rows are displayed before the Real distance seal is used.

Use source separation, Definition 4139.13. Agreement at the metric row alone cannot identify two reads; the source route still requires the same stream window, dyadic ledger, regular readback, and Real seal.

Finally invoke modulus ledger pullback, Definition 4139.14. Any convergence-facing or completion-facing consumer receives the pointwise metric contribution only after the $\mathsf{RegSeqRat}^{\uparrow}$ row has pulled the finite rational stream into the regular Cauchy readback surface. Therefore R is exposed before M is consumed. □

Theorem 3159.17 (Pointwise stream metric obligation scope)**.** *The carrier, pointwise classifier, constant-stream route, reindexing stability, NameCert obligations, sibling route, Real-completion anchor, finite-window consumption, completion separation, synchronized-window triangle route, source separation, modulus pullback, StreamName dependency, and regular-Cauchy bridge exhaust the obligation surface of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$*. Any accepted metric read must therefore pass through the displayed stream source, finite windows, dyadic ledger, regular readback, Real seal,*

metric row, componentwise transport, displayed replay, provenance, and local naming before it is public.

Proof. The carrier and classifier results, Definition 4139.1, Definition 4139.2, Definition 4139.3, and Definition 4139.4, determine the packet rows and the basic stable reads.

The NameCert obligations and sibling route then expose the immediate obligation surface. The Real-completion anchor, finite-window consumption, and completion separation theorems force any completion-facing consumer through the same finite windows, dyadic ledger, regular readback, Real seal, and metric row.

The synchronized-window triangle route, source separation, modulus pullback, StreamName dependency, and regular-Cauchy bridge cover the remaining consumer paths. Since each path is routed through the displayed rows $S, W, D, R, E, M, H, C, P, N$, there is no further local obligation for host function equality, quotient stream equality, selected representatives, countable choice, or ambient completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet is generated from a certified stream source, finite pointwise windows, a dyadic tolerance ledger, regular-rational readback, a Real distance seal, a displayed metric-space row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4139.17 exhausts the displayed obligation surface for certified pointwise stream metrics over StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, MetricSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient stream equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding these obligations to their exact primitive routes for stream sources, dyadic ledgers, regular readback, Real seals, metric rows, transport, replay, provenance, and local naming.

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet sits horizontally between the completion route of chapter 1506 and the convergence-budget route of chapter 2074. MetricCompletion consumers read pointwise distance through the certified stream-name rows of chapter 126, the dyadic ledger of chapter 291, the regular rational readback rows of chapter 301, the terminal real-name seal of chapter 38, and the displayed metric carrier rows of chapter 57. RealModulusOfConvergence consumers read the same finite window and dyadic ledger as a convergence budget before any pointwise stream metric row is available. The route exports no host function equality, quotient stream equality, selected representative, countable-choice schedule, or ambient completeness principle.

Theorem 3159.18 (PointwiseStreamMetric L10 dependency route)**.** *Every completion-facing* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *read consumes the certified* $\mathsf{StreamName}^{\uparrow}$ *pointwise windows of chapter 126, the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows of chapter 57, the* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage reads of chapter 1624, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary of chapter 38 before it can expose a pointwise metric packet to a downstream Real or completion consumer.*

Proof. Start with the certified stream source and finite pointwise windows in the carrier of Definition 4139.1. These rows force every read to pass through $\mathsf{StreamName}^{\uparrow}$ before any distance datum is visible.

Next read the metric row and the Real-uniform entourage row. The sibling chapters cited above provide the displayed distance and entourage surfaces used by the pointwise packet; no host function equality, quotient stream equality, selected representative, or countable-choice schedule is available in the carrier.

Finally apply the Real-completion anchor theorem Definition 4139.7. It keeps the route inside the already displayed stream windows, metric rows, entourage reads, and terminal Real seal, so the L10 dependency route is exactly the concrete sibling route stated above. □

Theorem 3159.19 (Pointwise stream metric completion sibling anchor)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet must first read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{Met}^{\uparrow}$ *distance rows before any completion-facing convergence data are available. The forward lattice route exports only this displayed pointwise metric packet; it does not supply host stream equality, selected representatives, countable-choice schedules, or an ambient completion principle.*

Proof. Project the carrier of Definition 4139.1. The completion-facing analytic rows are the certified stream source S, finite pointwise windows W, dyadic tolerance ledger D, regular-rational readback R, terminal Real seal E, and displayed metric row M.

Apply the pointwise classifier and completion-separation routes in Definition 4139.2 and Definition 4139.10. They force a distance read to consume S, W, D, R, E before M is inspected, and they exclude completion data not carried by the same finite packet.

The $\mathsf{MetricCompletion}^{\uparrow}$ consumer therefore receives convergence-facing data only after the sibling chain

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow}$$

has been displayed. Since the packet contains no host stream equality, selected representative, choice schedule, or ambient completion coordinate, the forward lattice route is exhausted by those rows. □

Every finite-limit or completion-facing consumer of $\mathsf{PointwiseStreamMetric}^{\uparrow}$ must read the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{Met}^{\uparrow}$ row from Definition 4139.1 before entering the finite-limit seal route of chapter 5722 or the completion route of chapter 1506. Thus $\mathsf{FiniteLimitSeal}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ receive pointwise stream metric data only as the finite window route $S, W, D, R, E, M, H, C, P, N$, not as quotient stream equality, selected representatives, host function equality, countable choice, ambient Real completeness, or a detached completion object. Every $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer reads pointwise metric data only after the finite-window source of chapter 126, the distance row of chapter 57, and the regular rational tolerance readback of chapter 301 are all visible. This route keeps the pointwise stream metric surface attached to displayed StreamName, Metric, and RegSeqRat rows before any downstream argument treats the stream as metric data.

Theorem 3159.20 (Pointwise stream metric sibling repair route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *must first read the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *rows from Definition 4139.1 and Definition 4139.5. A completion-facing read also passes through the metric-generated uniformity route supplied by chapter 7788 when that chapter is present; before that named bridge is inspected, the dependency is the* $\mathsf{Met}^{\uparrow}$*-to-*$\mathsf{UniformSpace}^{\uparrow}$ *handoff through chapter 57 and chapter 339. Only then may the consumer enter the downstream completion ledgers of chapter 1506, the finite-window boundary of chapter 1559, and chapter 2074.*

Proof. Project the carrier. The stream source, dyadic tolerance ledger, regular-rational readback, terminal Real seal, and metric row are displayed before any completion-facing row can be consumed.

The $\mathsf{MetricCompletionFinite}^{\uparrow}$ packet is the finite-window boundary between the pointwise stream packet and the completion-facing readback.

The uniformity dependency is not a new carrier row. It is the displayed passage from the pointwise metric row to the uniform entourage reader, using the Metric-to-UniformSpace handoff and, when available, the $\mathsf{MetricUniformity}^{\uparrow}$ bridge that packages that route for completion consumers.

Read the obligation surface. It requires carrier admission, pointwise classifier exposure, constant-stream zero-distance exposure, reindexing stability, and ledger non-escape over the same finite rows.

The downstream references therefore consume the pointwise metric packet as a displayed BEDC route. No host completion theorem, quotient stream equality, selected representative, or countable-choice schedule is introduced. □

When a $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance read is used as a completion-facing convergence witness, the pointwise comparison stream leaves the carrier through the convergence ledger of chapter 2074. The route first projects the carrier of Definition 4139.1 and the obligation surface of Definition 4139.5: the certified stream source, finite pointwise windows, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, and displayed metric row are consumed before the RealModulusOfConvergenceUp ledger may read the sequence as completion data. Every $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer that reads a pointwise metric value must pass through the finite $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before the metric comparison is public. The route exposes a finite stream, rational readback, and Real-facing seal boundary; it does not export quotient stream equality, a selected representative, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ hub has a substantive horizontal dependency on chapter 129 and chapter 38: compact-metric finite-net rows and $\mathsf{Real}^{\uparrow}$ distance rows may enter the pointwise stream metric route only through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ carriers already named by the packet. The dependency is therefore a finite stream-and-rational route, not ambient compactness, selected subsequences, quotient stream equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3159.21 (Pointwise stream metric regular-Cauchy pruning route)**.** *Every completion-facing* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *read that prunes redundant pointwise observations must first expose the* $\mathsf{StreamName}^{\uparrow}$ *pointwise windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal already carried by Definition 4139.1. The pruning handoff then factors through the* $\mathsf{RegularCauchyPruning}^{\uparrow}$ *certificate of chapter 7851 before any downstream metric or completion consumer reads the retained stream. The route exports no host stream equality, hidden infinite selector, quotient Real equality, or ambient completeness principle.*

Proof. Project the pointwise stream metric carrier. Its analytic rows expose the stream source, finite pointwise windows, dyadic tolerance ledger, regular rational readback, Real seal, and metric row before any completion-facing consumer has a pointwise distance packet to inspect.

Apply the regular-Cauchy pruning certificate of chapter 7851. That sibling consumes exactly the StreamName windows, retained schedule, dyadic ledger, RegSeqRat readback, and inherited Real seal needed to delete redundant observation requests as finite NameCert data.

Finally use the existing L10 route of Definition 4139.18. The retained stream remains inside the displayed pointwise windows and Real seal boundary before it reaches metric or completion consumers, so the horizontal edge is a concrete stream/pruning dependency rather than a kernel-only pointwise metric export. □

Theorem 3159.22 (Pointwise stream metric stream-metric sibling route)**.** *Every* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer first reads the certified* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126 and the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows of chapter 57. The pointwise stream packet may then hand its finite distance surface to* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumers, but only after the stream windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, and metric row have all been exposed.*

Proof. Project the carrier of Definition 4139.1. Its stream source row S, finite window row W, dyadic ledger D, regular readback row R, Real seal E, and metric row M are the only analytic rows available before transport, replay, provenance, and local naming.

Apply Definition 4139.2. It forces the pointwise distance read to pass through S, W, D, R, E before the metric row M can be inspected.

Finally use Definition 4139.7. Any completion-facing or convergence-facing consumer receives the same finite route before reaching $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{RealModulusOfConvergence}^{\uparrow}$. Since no carrier coordinate stores host stream equality, selected representatives, countable-choice schedules, or ambient completeness, the sibling route is exhausted by the displayed stream and metric surfaces. □

Theorem 3159.23 (Pointwise stream metric StreamName completion sibling route)**.** *Every* $\mathsf{PointwiseStreamMetric}$ *completion-facing consumer first reads the certified* $\mathsf{StreamName}^{\uparrow}$ *finite observations of chapter 126 and the* $\mathsf{MetricCompletion}^{\uparrow}$ *existence boundary of chapter 5721. The pointwise packet reaches that boundary only through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow}.$$

Thus the completion read consumes pointwise stream observations, regular rational readback, the terminal Real classifier, and the metric row already displayed in $\mathsf{PointwiseStreamMetric}^{\uparrow}$*; it does not introduce host stream equality, a selected representative, quotient completion, countable-choice scheduling, or ambient completeness.*

Proof. Project the carrier of Definition 4139.1. The stream source S, finite window schedule W, dyadic ledger D, regular readback row R, terminal Real seal E, and metric row M are visible before any structural row can be consumed.

Apply the pointwise classifier and Real-completion anchor, Definition 4139.2 and Definition 4139.7. They force every distance read through the StreamName window, RegSeqRat readback, Real seal, and metric row before a completion-facing consumer receives the packet.

Finally read the completion existence boundary through chapter 5721. Since the incoming route has already exhausted $S, W, D, R, E, M, H, C, P, N$, the completion consumer receives only the displayed sibling dependency surface and cannot acquire host stream equality, quotient completion, selected representatives, countable-choice schedules, or ambient completeness from $\mathsf{PointwiseStreamMetric}^{\uparrow}$. □

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ route also admits located-real lower-cut rows as completion-facing input data. A metric comparison first reads the certified $\mathsf{StreamName}^{\uparrow}$ windows, passes the rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, consumes the lower-cut row before the terminal $\mathsf{Real}^{\uparrow}$ consumer, and only then exposes the pointwise metric packet to Real and Completion dependents. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet depends substantively on the finite $\mathsf{StreamName}^{\uparrow}$ observation route of chapter 126, the $\mathsf{Met}^{\uparrow}$ distance-witness interface of chapter 57, and the terminal $\mathsf{Real}^{\uparrow}$ seal row of chapter 38. The pointwise metric route is not an isolated kernel float: it consumes stream

windows, metric distance rows, and Real-facing seal rows before any pointwise classifier read is public. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ route consumes the finite-window stream rows of chapter 126 and the distance rows of chapter 57 before any Real seal or completion-facing read is available. This dependency keeps pointwise stream metrics attached to the stream-name carrier and the metric carrier that their local classifier actually reads.

Proposition 3159.24 (Pointwise stream metric StreamName–Metric dependency)**.** *Every* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer that reaches a Real seal, a* $\mathsf{MetricCompletion}^{\uparrow}$ *route, or another completion-facing read first factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite windows and the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows. No completion-facing consumer can bypass those two carrier surfaces.*

Theorem 3159.25 (Pointwise stream metric coordinate extension stability)**.** *Let*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

be an accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any accepted coordinate extension of* Q *preserves the route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$*: the extended packet must keep the certified stream source, dyadic ledger, regular readback, terminal Real seal, displayed metric row, convergence ledger, and completion-facing dossier visible through the same finite NameCert route. The extension cannot identify quotient streams, select representatives, or replace the displayed rows by host function equality.*

Proof. Begin with the pointwise classifier of Definition 4139.2. It forces the read of Q through S, W, D, R, E before the displayed metric row M is inspected, so any coordinate extension starts from the same stream, dyadic, regular-rational, and Real-seal rows.

Next apply reindexing stability, Definition 4139.4. The transport row H and replay row C preserve the stream source, window schedule, dyadic ledger, regular readback, Real seal, metric row, provenance, and local naming rows under every accepted scheduled replacement.

Use the Real-completion anchor, Definition 4139.7. It routes completion-facing reads through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ before any downstream completion consumer receives the pointwise packet.

Finally invoke the dossier lattice route, Definition 4139.8. A $\mathsf{RealModulusOfConvergence}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads the same finite dossier before convergence or completion data is public. Since none of the cited carrier rows stores quotient stream equality, selected representatives, or host function equality, coordinate extension cannot introduce them. □

Theorem 3159.26 (Pointwise stream metric orphan lattice route)**.** *Every* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *consumer of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *must first expose the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Met}^{\uparrow}$ *rows of Definition 4139.1 and Definition 4139.5. Only after those rows are visible may the consumer enter the finite-window completion boundary of chapter 1559. The route exports no quotient stream equality, no selected completion representative, no ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*, and no host metric completion object.*

Proof. Project the pointwise stream metric carrier. The stream windows, dyadic tolerance ledger, regular rational readback, Real seal, and metric-space row are all displayed before a completion-facing read has a source.

Apply the NameCert obligations for the same packet. They assign the completion-facing dependency to the visible stream and metric rows rather than to an external completion object.

Thus a $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer can read only the finite row lattice already exposed by $\mathsf{PointwiseStreamMetric}^{\uparrow}$, and the refused quotient, representative, completeness, and host-completion objects remain outside the route. □

Theorem 3159.27 (Pointwise stream metric hub completion sibling route)**.** *Every hub-level* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *completion read first consumes the certified stream-name surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the metric carrier surface of chapter 57, and the finite completion boundary of chapter 1559 before exporting the pointwise distance packet. The hub exports only the displayed stream, readback, Real, metric, and finite-completion lattice; it exports no detached stream quotient, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.*

Proof. Read the hub through the body carrier and its existing sibling routes. The stream-name windows, regular rational readback, Real seal, and metric distance rows are visible before any pointwise distance packet is exported.

The finite completion boundary is then supplied by $\mathsf{MetricCompletionFinite}^{\uparrow}$. It consumes the same displayed rows as completion-facing data and adds no quotient stream equality, selected representative, countable-choice schedule, host function equality, or ambient completeness theorem.

Thus every hub-level completion read follows the stated sibling lattice before exporting the pointwise stream metric distance packet. □

Theorem 3159.28 (Pointwise stream metric top-level sibling lattice route)**.** *Every top-level read exported by the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *hub first consumes the certified stream-name surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the metric carrier surface of chapter 57. Only after those sibling rows are visible may a pointwise stream metric read expose its local distance packet; the hub exports no detached stream quotient, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.*

Proof. Read the hub through its body chapter, Definition 4139.6. That route already places $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ before the pointwise metric row.

The present top-level route records the same dependency at the hub boundary. Since the hub has no substantive rows of its own, any exported pointwise stream metric read is exactly the sibling lattice route supplied by the body chapter and its Real-modulus dependency file. □

Theorem 3159.29 (Pointwise stream metric real-completion route)**.** *For every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet, each completion-facing read factors through finite* $\mathsf{StreamName}^{\uparrow}$ *coordinate windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Real}^{\uparrow}$ *distance seals,* $\mathsf{MetricCompletion}^{\uparrow}$ *limit witnesses, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The route reads the coordinate windows before the rational readback, the rational readback before the Real distance seal, and the Real seal before any metric-completion handoff.*

The packet exports no quotient stream equality, no selected limit representative, and no ambient completeness theorem. It is the same finite L10 route used by the StreamName chapter of chapter 126, the RegSeqRat chapter of chapter 301, the Real chapter of chapter 38, and the metric-completion interface of chapter 1506.

Proof. Project the accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet onto its coordinate-window, regular-rational readback, real-distance, completion-facing, transport, replay, provenance, and local naming rows. The coordinate windows are finite $\mathsf{StreamName}^{\uparrow}$ data, so a completion read cannot start from an ambient stream equality.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback rows then expose the rational comparisons used by each displayed coordinate distance. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of those readbacks, not a host real equality detached from them.

Only after those rows are fixed may a $\mathsf{MetricCompletion}^{\uparrow}$ consumer inspect a limit witness. Componentwise hsame transport and Cont replay preserve the same displayed route, while Pkg provenance and local $\mathsf{NameCert}$ naming export no selected representative or ambient completeness principle. $\square$

Chapter 3160

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and Frechet on cofinite filters, the $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ packet records the finite route by which regular Cauchy tail evidence is read as a cofinal Frechet-filter schedule. It sits between the tail-filter certificate of chapter 1605, the finite-tail schedule of chapter 1119, and the Frechet cofinite filter interface of chapter 1564. The chapter contributes a NameCert landing for Real and Completion consumers; it does not select limits, quotient streams, host reals, or an ambient completeness theorem.

Definition 3160.1 (Regular Cauchy tail-filter cofinality carrier)**.** *A* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, G, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *source row, carrying the regular tail-filter evidence already displayed by the sibling tail-filter chapter.*
2. *F is the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *cofinal schedule row, recording the finite tail windows through which later requests are admitted.*
3. *G is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite-filter row produced from the same finite tail schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose inspected windows are exposed only after T, F, G have been replayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled tail windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same scheduled readback.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular tail, finite-tail, Frechet-filter, stream, readback, and dyadic rows have been displayed.*
8. *H records componentwise* hsame *transport for $T, F, G, S, R, D, A, C, P, N$.*
9. *C records displayed* Cont *replay from a cofinal-tail request through the regular tail-filter source, finite-tail schedule, Frechet-filter row, stream windows, regular rational readback, dyadic tolerances, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFilterCofinalityUp.RegularCauchyTailFilterCofinalityTasteGate single carrier alignment`

Theorem 3160.2 (Regular Cauchy tail-filter cofinality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier* $K = (T, F, G, S, R, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, cofinality of the finite-tail schedule between T and G, ordered exposure of* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside $T, F, G, S, R, D, A, H, C, P, N$.*

Proof. Project the carrier of Definition 4140.1. The analytic rows are exactly the regular tail-filter source, finite-tail schedule, Frechet-filter route, stream windows, regular rational readback, dyadic tolerances, and terminal Real seal.

Cofinality is recorded by the row order T, F, G. A request reads the regular Cauchy tail-filter evidence first, then the finite schedule that witnesses cofinal tail access, and only then the Frechet cofinite-filter row.

The remaining rows preserve that route. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are downstream of the cofinal filter route, while H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, selector, choice, host-equality, and completeness objects are absent, the listed rows exhaust the local obligations. □

Theorem 3160.3 (Regular Cauchy tail-filter cofinal Frechet route)**.** *Every Frechet-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{FiniteTailFilter}^{\uparrow} \longmapsto \mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}$$

Thus the Frechet cofinite schedule is cofinal for the displayed regular Cauchy tail evidence before any stream, readback, dyadic, or Real-seal row is exported.

Proof. Start with a consumer of $K = (T, F, G, S, R, D, A, H, C, P, N)$. Carrier admission exposes G only as a row downstream of the regular tail-filter source T and the finite-tail schedule F.

Replay the continuation row C. The request consumes the tail evidence named by chapter 1605, then the finite-tail schedule of chapter 1119, and then the Frechet route of chapter 1564.

Only after that cofinal route is displayed can the stream windows S, readback row R, dyadic tolerance row D, and Real seal A be consumed. Componentwise transport and provenance preserve the same order, so the exported Frechet and Real-facing routes are exactly the displayed finite cofinal-tail route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ packet is generated from a RegularCauchyTailFilter source row, a FiniteTailFilter cofinal schedule, a FrechetFilter cofinite-filter row, StreamName windows, RegSeqRat readback,

DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailFilterCofinality}\uparrow}$, seed)

Scope closed: Definition 4140.1, Definition 4140.2, and Definition 4140.3 seed the finite cofinal route from regular Cauchy tail evidence through finite-tail schedules and Frechet cofinite filters to stream, readback, dyadic, and Real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailFilterCofinality}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyTailFilterCofinalityUp.RegularCauchyTailFilterCofinalityTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal finite-tail scheduling, Frechet-route exactness, stream-readback order, dyadic tolerance preservation, Real-seal boundary, and ledger refusal proofs for the same packet.

Chapter 3161

A Concrete Naming Certificate for $\mathsf{BishopRealPrecision}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealPrecision}^{\uparrow}$ packet records the finite precision budgets by which a real-number read is inspected before any completion-facing consumer may treat it as usable $\mathsf{Real}^{\uparrow}$ data. It sits over the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names the visible precision route; it is not quotient equality of streams, a selected limit, a countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3161.1 (Bishop real precision carrier)**.** *A* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (D, W, R, E, M, H, C, Q, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic radius, slack comparison, and endpoint budget for the inspected read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the finite observation schedule consumed by the precision request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational approximants visible through D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the tolerance, window, and regular-readback rows have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff row, recording the finite modulus entry that justifies the requested precision budget for later Real and Completion consumers.*
6. *H records componentwise* hsame *transport for tolerance, window, readback, seal, modulus, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a precision request through D, W, R, E, M.*
8. *Q is the* Pkg *provenance over* $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, M, H, C, Q, N$. It has no coordinate for quotient stream equality, a selected host limit, a countable-choice schedule, host real equality, hidden precision repair, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopRealPrecisionUp.BishopRealPrecisionTasteGate single carrier alignment`

Theorem 3161.2 (Bishop real precision NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier*

$$P = (D, W, R, E, M, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tolerance, window, readback, seal, modulus, transport, replay, provenance, and naming rows; tolerance-window exactness through D, W; regular rational readback through R; real-seal nonescape through E; modulus handoff through M; componentwise transport through H; displayed replay through C; package provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 4141.1. The rows $D, W, R, E, M, H, C, Q, N$ are the only coordinates exposed to a precision consumer.

The tolerance and window are fixed first. A request chooses the dyadic budget D and the finite stream schedule W, then reads the regular rational row R along that same displayed route.

The real seal E is downstream of the readback row, while the modulus handoff M records the finite Cauchy budget consumed by later Real and Completion packets. Transport, replay, provenance, and local naming preserve exactly those rows.

Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface and refuse quotient streams, selected limits, countable choice, host real equality, hidden precision repair, and ambient completeness. □

Theorem 3161.3 (Bishop real precision RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *precision read exported by an accepted* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before use. Equivalently, a completion-facing consumer can read the requested precision only after the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

has been replayed inside the same packet.

Proof. Begin with an accepted carrier $P = (D, W, R, E, M, H, C, Q, N)$. The finite stream window W determines which rational approximants may be inspected, and the dyadic tolerance row D determines the budget at which the inspection is valid.

The regular rational readback R is the only row that exports approximants to the precision request. The modulus handoff M records the finite Cauchy budget attached to that readback before the $\mathsf{Real}^{\uparrow}$ seal E is consumed.

Finally, H, C, Q, N transport, replay, source, and name the same route. Thus every accepted precision read reaches $\mathsf{Real}^{\uparrow}$ only through the displayed finite route, and no hidden limit, quotient equality, choice schedule, or ambient completeness principle can supply a bypass. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealPrecision}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, a CauchyModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4141.1, Definition 4141.2, and Definition 4141.3 seed the finite precision-budget route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopRealPrecision}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, host real equality, hidden precision repair, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, RegSeqRat readback, Real seal handoff, CauchyModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealPrecisionUp.

Chapter 3162

A Concrete Naming Certificate for HyperbolicVisualShadowMetric$^{\uparrow}$

The HyperbolicVisualShadowMetric$^{\uparrow}$ packet realizes the boundary-shadow visual metric reading of section 9463.3 as finite NameCert data. It consumes disk coordinates from chapter 377, the metric-read surface of chapter 3474, the boundary-shadow packet of chapter 3813, the visual-metric packet of chapter 3866, and the Busemann-boundary route of chapter 3732. Its carrier binds those rows into one finite visual-shadow metric packet rather than a completed disk model, quotient boundary, global field instance, or host hyperbolic geometry.

Definition 3162.1 (Hyperbolic visual-shadow metric carrier)**.** *A* HyperbolicVisualShadowMetric$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (U, B, M, S, G, T, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UnitDisk$^{\uparrow}$ *disk-carrier row for the inspected points and visible disk windows.*
2. *B is the boundary-shadow row, recording the displayed boundary projection or shadow comparison consumed by the read.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row on the displayed disk data.*
4. *S is the* HyperbolicVisualMetric$^{\uparrow}$ *visual-metric row, recording the finite visual comparison surface.*
5. *G is the* HyperbolicBusemannBoundary$^{\uparrow}$ *boundary row used to keep the shadow comparison attached to a displayed Busemann endpoint.*
6. *T is the phase-transport row, recording the displayed observer-expansion transport through which the visual-shadow read is replayed.*
7. *H records componentwise* hsame *transport for disk, boundary-shadow, metric, visual, Busemann, phase-transport, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a visual-shadow metric request through U, B, M, S, G, T.*
9. *P is the* Pkg *provenance over* HyperbolicVisualShadowMetric$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $U, B, M, S, G, T, H, C, P, N$. It has no coordinate for a host Poincare disk, quotient ideal boundary, global geometry theorem, physical observer trajectory, or implicit field-faithfulness witness.

Lean4 (definition) `BEDC.Derived.HyperbolicVisualShadowMetricUp.HyperbolicVisualShadowMetricUp`

Theorem 3162.2 (Hyperbolic visual-shadow metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *carrier*

$$V = (U, B, M, S, G, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$ *surface consists exactly of disk-carrier admission, boundary-shadow exposure, hyperbolic metric readback, visual-metric readback, Busemann-boundary attachment, phase-transport replay, componentwise transport, displayed continuation, package provenance, and local naming through $U, B, M, S, G, T, H, C, P, N$.*

Proof. Project the carrier of Definition 4142.1. The rows $U, B, M, S, G, T, H, C, P, N$ are the whole visual-shadow metric packet.

The disk and boundary data are read first. The disk carrier U supplies the visible points, the boundary-shadow row B supplies the shadow comparison, and the metric row M supplies the hyperbolic readback attached to those points.

The visual and Busemann rows then bind the comparison to a displayed boundary-facing route. The phase-transport row T records how the observer-expansion reading is replayed without importing a completed geometry.

Finally H, C, P, N transport, replay, source, and name the same finite route. Therefore the listed rows exhaust the NameCert surface and leave host disk models, quotient boundaries, global geometry, physical trajectories, and implicit field-faithfulness outside the packet. □

Lean4 ✓ `BEDC.Derived.HyperbolicVisualShadowMetricUp.HyperbolicVisualShadowMetricNamecertObligations`

Theorem 3162.3 (Hyperbolic visual-shadow metric factorization)**.** *Every accepted visual-shadow metric read factors through the disk carrier, boundary-shadow row,* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row,* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row,* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, phase-transport row, displayed replay, package provenance, and local* $\mathsf{NameCert}$ *row before use.*

Proof. Begin with an accepted carrier $V = (U, B, M, S, G, T, H, C, P, N)$. A read first lands on the disk-carrier row U, then consumes the boundary-shadow row B and the hyperbolic metric row M.

The visual comparison row S and Busemann boundary row G attach that metric read to a displayed boundary-facing visual surface. The phase-transport row T records the route along which the observer-expansion packet replays the comparison.

Transport, continuation, provenance, and naming are then supplied by H, C, P, N. Since no further coordinate is present, every accepted read factors through the displayed rows and cannot bypass them through host geometry, quotient boundary data, or an implicit global field instance. □

Falsifiable prediction: Every accepted visual-shadow metric read factors through the disk carrier, boundary-shadow row, $\mathsf{HyperbolicMetric}^{\uparrow}$ row, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ row, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ row, phase-transport row, displayed replay, provenance, and local $\mathsf{NameCert}$ row.

Independence witness: unitdisk, hyperbolicmetric, hyperbolic-boundary-shadow, hyperbolic-visual-metric, hyperbolic-busemann-boundary: no listed sibling is bijective to $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ because this packet jointly carries disk windows, boundary shadows, metric readback, visual comparison, Busemann boundary attachment, phase transport, replay, provenance, and local naming.

The future `lean4/BEDC/Derived/HyperbolicVisualShadowMetricUp/TasteGate.lean` file supplies explicit $\mathsf{Nontrivial\ HyperbolicVisualShadowMetric}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicVisualShadowMetric}^{\uparrow}$ instances. Those instances must preserve $U, B, M, S, G, T, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ packet is generated from UnitDisk carrier rows, boundary-shadow rows, HyperbolicMetric readback, HyperbolicVisualMetric comparison, HyperbolicBusemannBoundary attachment, phase transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4142.1, Definition 4142.2, and Definition 4142.3 seed the finite boundary-shadow visual metric route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed Poincare disk geometry, quotient ideal boundary equality, global field instances, physical observer trajectories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk readback, boundary-shadow exposure, metric readback, visual comparison, Busemann boundary attachment, phase transport, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicVisualShadowMetricUp.

Chapter 3163

A Concrete Naming Certificate for $\mathsf{LocatedCauchyExtraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ packet records the finite extraction surface that turns located regular-Cauchy modulus evidence into a sealed real-name read. It sits after the dyadic rational ledger of chapter 291, the stream observation window of chapter 126, the regular rational readback of chapter 301, the located-modulus boundary of chapter 3655, the located comparison rows of chapter 1730, and the terminal real-name seal of chapter 38. The packet names the extraction route as displayed BEDC data; it does not quotient Cauchy names, select a limit, use trichotomy, import a host order, or appeal to ambient completeness.

3163.1 Located Cauchy extraction carrier

Definition 3163.1 (Located Cauchy extraction carrier)**.** *A* $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ *carrier is a finite packet* $(D, S, Q, L, U, M, E, H, C, P, N)$ *with the following displayed rows:*

1. D*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for the positive dyadic budgets used by the extraction;*
2. S*, a* $\mathsf{StreamName}^{\uparrow}$ *observation window for the rational name being read;*
3. Q*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular Cauchy rational values visible on* S*;*
4. L *and* U*, lower and upper* $\mathsf{LocatedRealComparison}^{\uparrow}$ *rows tied to the same window and dyadic ledger;*
5. M*, the* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *row assigning displayed precision budgets to finite window depths;*
6. E*, the* $\mathsf{Real}^{\uparrow}$ *seal row that receives only the extracted finite regular-Cauchy read;*
7. H*, the componentwise* hsame *transport rows for* D, S, Q, L, U, M, E*;*
8. C*, the displayed* Cont *replay routes preserving the extraction window, located brackets, modulus row, and real seal;*
9. P*, the* Pkg *provenance rows for the dyadic ledger, stream window, regular readback, located comparison evidence, located modulus, and real seal;*
10. N*, the local* $\mathsf{NameCert}$ *row binding these ten rows as one located Cauchy extraction certificate.*

The carrier contains no quotient equality of Cauchy names, no selected limit, no trichotomy, no host real order predicate, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCauchyExtractionUp.LocatedCauchyExtractionTasteGate single carrier alignment`

Theorem 3163.2 (Located Cauchy extraction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$ *surface for Definition 4143.1 consists of finite admission of the rows* $(D, S, Q, L, U, M, E, H, C, P, N)$*, shared-window alignment for* S*, dyadic precision exactness through* D*, regular readback confinement through* Q*, located bracket export through* L, U*, modulus handoff through* M*, real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4143.1. The projection exposes exactly the rows $D, S, Q, L, U, M, E, H, C, P,$

The extraction route is fixed by the shared window and ledger. Row S is the only observation window, row D is the only dyadic precision source, and row Q is the only regular rational readback exposed to the located comparison and modulus rows.

The located handoff is then forced through L, U, M before the real seal E is read. Componentwise transport and replay through H and C preserve those rows, while P and N record provenance and local naming.

No carrier coordinate names quotient equality, a selected limit, trichotomy, a host order predicate, or ambient completeness. Hence the stated obligations exhaust the local NameCert surface. □

Theorem 3163.3 (Located Cauchy extraction Real-seal route)**.** *Every accepted* $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ *packet exports toward* $\mathsf{Real}^{\uparrow}$ *only the finite route*

$$D, S, Q, L, U, M \longmapsto E \longmapsto H, C, P, N.$$

Any $\mathsf{LocatedRealComparison}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of the accepted packet must read through the displayed dyadic ledger, stream window, regular rational readback, located comparison rows, located modulus row, transport, replay, provenance, and local NameCert rows. The route supplies no quotient Cauchy equality, selected limit, trichotomy, host order, or ambient completeness.*

Proof. Start with an accepted carrier and apply Definition 4143.2. The obligations pin the regular rational readback to Q over the displayed window S and dyadic ledger D.

The located comparison rows L, U and the located-modulus row M are the only rows that can be exported before the seal E. Therefore every Real-facing consumer reaches the seal by the displayed finite route and not by selecting a limit outside the packet.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since the refused objects are absent from the carrier coordinates, no downstream consumer can recover quotient equality, trichotomy, host order, or ambient completeness from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ packet is generated from a dyadic precision ledger, a StreamName observation window, a RegSeqRat readback, lower and upper located comparison rows, a regular-Cauchy located modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 4143.1, Definition 4143.2, and Definition 4143.3 seed the finite extraction route from located regular-Cauchy modulus evidence to the RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyExtraction}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.LocatedCauchyExtractionUp.LocatedCauchyExtractionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy names, selected limits, trichotomy, host real order, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located modulus handoff, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyExtractionUp.

Chapter 3164

A Concrete Naming Certificate for $\mathsf{BanachMazurGame}^{\uparrow}$

Following Banach and Mazur on nested-open games, the $\mathsf{BanachMazurGame}^{\uparrow}$ packet records the finite alternating-play route by which a Polish complete-metric Baire argument is inspected inside BEDC. It sits over the Polish-space synthesis of chapter 1490, the Baire-category surface of chapter 1542, the complete-metric handoff of chapter 128, the metric carrier of chapter 57, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the classical Banach–Mazur game interface as finite NameCert data without adding a selected winning strategy, a host intersection point, countable choice, quotient equality of plays, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3164.1 (Banach–Mazur game carrier)**.** *A* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, B, K, M, L, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row naming the complete separable metric source over which the play is read.*
2. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *row exposing the dense-open and residual obligations visible to the game consumer.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, including displayed Cauchy-window and nested-ball support.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row carrying the distance classifier, metric-radius comparisons, and ball-inclusion checks.*
5. *L is the alternating legal-move ledger, recording each finite play prefix, player tag, chosen basic open or ball row, predecessor inclusion, and shrinking-radius evidence.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing the finite play prefixes, dense-open requests, ball observations, and Cauchy-window inspections visible to downstream consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius budgets, tolerance comparisons, and dense-window observations used by L and S.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Polish, Baire-category, complete-metric, metric, legal-move, schedule, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for P, B, K, M, L, S, R, E and for the structural rows.*
10. *C records displayed* Cont *replay from a game request through the Polish source, Baire-category obligation, complete-metric handoff, metric-radius checks, legal-move ledger, finite schedule, regular readback, Real seal, transport, provenance, and local naming rows.*
11. *Q is the* Pkg *provenance over* $\mathsf{BanachMazurGame}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common Polish, Baire-category, complete-metric, metric, play-ledger, stream, readback, and real-seal surface. It has no coordinate for a global winning strategy, a chosen play limit, an infinite intersection object, a quotient of plays, countable choice, or an ambient host topology.

Lean4 ✓ `BEDC.Derived.BanachMazurGameUp.TasteGate.BanachMazurGameTasteGate single carrier alignment`

Theorem 3164.2 (Banach–Mazur game NameCert obligations)**.** *For every accepted* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier $G = (P, B, K, M, L, S, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Polish, Baire-category, complete-metric, metric, legal-move, stream, readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *source compatibility, requiring P, K, M to expose one complete metric source before a legal move can be read;*
3. *Baire-category visibility, requiring each dense-open or residual request to pass through B and the finite schedule S before the play ledger is consumed;*
4. *legal-move exactness, requiring L to store only finite alternating prefixes, predecessor inclusions, and shrinking-radius evidence;*
5. *readback discipline, requiring S, R, E to be read in that order before any real-facing terminal observation is exported;*
6. *structural preservation, requiring H, C, Q, N to transport, replay, source, and name exactly the displayed game route;*
7. *nonescape, excluding global winning strategies, selected limit points, host intersections, quotient equality of plays, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden dense-open oracles.*

No Banach–Mazur-game obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4144.1. The analytic rows are P, B, K, M, L, S, R, E, and the structural rows are H, C, Q, N.

The complete metric source is fixed before play begins. The rows P, K, M expose the Polish packet, complete-metric handoff, and metric-radius checks on one displayed carrier, so a move ledger cannot switch to a second metric source.

The Baire-category part is scheduled. A dense-open or residual request is first read through B and then through the finite stream row S, while L records only the finite legal prefix and the displayed inclusion evidence for the chosen successor.

The terminal read is downstream of regular rational observation. The row R supplies rational centers, radii, and tolerances before E is consumed, and H, C, Q, N preserve the same route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3164.3 (Banach–Mazur Polish–Baire handoff)**.** *Every* $\mathsf{BaireCategory}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{BanachMazurGame}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

The route supplies the complete-metric Baire interface by exposing finite alternating play prefixes, dense-open requests, metric-ball inclusions, stream windows, regular rational readback, and the terminal real-name seal. It does not select a winning strategy, choose an intersection point, quotient plays, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Polish source. The packet reads P, K, M as the complete separable metric carrier, complete-metric handoff, and metric-radius comparison surface before any game row is inspected.

Next expose the Baire-category request. The row B supplies the dense-open or residual obligation, and the finite schedule S records the play prefix and observation windows through which that request is visible.

The game ledger is then read as finite data. The row L records alternating legal moves, predecessor inclusions, and shrinking-radius evidence, while R supplies the regular rational observations needed to compare centers and radii.

Finally the terminal seal E is reached only after the route has passed through P, K, M, B, L, S, R. The structural rows H, C, Q, N transport, replay, source, and name the same handoff, so the consumer receives the displayed Polish–Baire game interface and no hidden strategy, selected point, quotient play, choice principle, or ambient completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachMazurGame}^{\uparrow}$ packet is generated from a PolishSpace source, BaireCategory dense-open row, CompleteMetric handoff, MetricSpace radius checks, an alternating legal-move ledger, StreamName schedule, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4144.1, Definition 4144.2, and Definition 4144.3 seed the finite Banach–Mazur route over $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim selected winning strategies, host intersection points, quotient play equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, Baire-category visibility, legal-move exactness, readback discipline, structural preservation, and nonescape for BEDC.Derived.BanachMazurGameUp.

Chapter 3165

A Concrete Naming Certificate for $\mathsf{EffectiveMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive metric spaces and Weihrauch represented spaces, the $\mathsf{EffectiveMetricSpace}^{\uparrow}$ packet records the finite interface needed by completion consumers before any limit object is formed. It refines the represented-name surface of chapter 1494, the distance obligations of chapter 57, the dyadic tolerance arithmetic of chapter 291, the scheduled name windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real seal of chapter 38.

Definition 3165.1 (EffectiveMetricSpace carrier)**.** *An* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, S, W, \rho, D, R, \delta, \Sigma, \Delta, T, C, P, N)$$

with the following rows:

1. *a point-name row* X *and a represented-space schedule* S, W, ρ *for finite* $\mathsf{StreamName}^{\uparrow}$ *reads;*
2. *a dyadic tolerance row* D *and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *for all inspected distance approximants;*
3. *a* $\mathsf{Real}^{\uparrow}$*-valued distance seal* δ *whose admitted reads pass through* D *and* R*;*
4. *symmetry and triangle replay rows* Σ, Δ *for the displayed distance seal;*
5. *componentwise transport* T*, replay* C*, provenance* P*, and a local* $\mathsf{NameCert}$ *row* N*.*

The carrier is accepted only when every distance request exposes a scheduled name window, a dyadic tolerance, a regular rational readback, and the resulting $\mathsf{Real}^{\uparrow}$ *seal before a completion-facing consumer can inspect it.*

Lean4 ✓ `BEDC.Derived.EffectiveMetricSpaceUp.EffectiveMetricSpaceTasteGate single carrier alignment`

Theorem 3165.2 (EffectiveMetricSpace NameCert obligations)**.** *Every accepted* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier* $E = (X, S, W, \rho, D, R, \delta, \Sigma, \Delta, T, C, P, N)$ *supplies the finite NameCert obligations needed by an* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ *consumer: point-name admission, represented-space scheduling, dyadic tolerance tracking, regular rational readback, Real-valued distance sealing, symmetry replay, triangle replay, componentwise transport, provenance, and local naming.*

Proof. Project the carrier of Definition 4145.1. The only rows available to a completion-facing reader are the displayed point names, scheduled representation reads, tolerance ledger, rational readback, real seal, replay rows, transport, provenance, and local NameCert.

The represented-space part is sourced by chapter 1494. The tolerance and readback rows then pass through chapter 291 and chapter 301, so the distance seal cannot be opened without the finite arithmetic route.

Finally the symmetry and triangle rows replay only that displayed seal. Transport and provenance carry the same finite packet into the local NameCert row, excluding host equality of points, selected representatives, countable choice, and ambient completion data. □

Theorem 3165.3 (EffectiveMetricSpace distance readback)**.** *For any two accepted point names in an* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier, each distance read available to a Real or completion consumer factors through*

$$S, W, \rho \longmapsto D \longmapsto R \longmapsto \delta \longmapsto \Sigma, \Delta, T, C, P, N.$$

The read is therefore a finite scheduled $\mathsf{Real}^{\uparrow}$ *seal assembled from* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *data, not a host metric value, quotient representative, selected Cauchy name, or pre-existing completion point.*

Proof. Start with the two point names and their represented-space schedule. The carrier exposes only the finite $\mathsf{StreamName}^{\uparrow}$ windows and representation relation before any distance request can be made.

Next consume the dyadic and regular-rational rows. They produce the tolerance-indexed readback that feeds the $\mathsf{Real}^{\uparrow}$ distance seal, so the downstream reader sees a certified real value rather than an ambient metric value.

The replay rows Σ and Δ preserve symmetry and triangle evidence for the same seal. The transport, replay, provenance, and local naming rows then package the read for Real and completion consumers without adding a quotient, a selected representative, a choice principle, or an already completed point. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffectiveMetricSpace}^{\uparrow}$ packet is generated from displayed point-name rows, a represented-space schedule, dyadic tolerance data, regular rational readback, a Real-valued distance seal, symmetry and triangle replay rows, componentwise transport, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4145.1, Definition 4145.2, and Definition 4145.3 seed the finite effective-metric source interface consumed by Real and completion chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close metric completion, located limits, quotient metric spaces, selected representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, represented-space scheduling, dyadic and regular-rational readback, Real seal stability, symmetry and triangle replay, transport, provenance, and local naming.

Chapter 3166

A Concrete Naming Certificate for $\mathsf{ComputablePolishSpace}^{\uparrow}$

Following Weihrauch and Alexandrov–Urysohn, the $\mathsf{ComputablePolishSpace}^{\uparrow}$ packet records the complete separable metric object whose dense names, Cauchy realizers, representation decoder, and terminal real seal are all visible before any computable-analysis consumer reads the space. It sits after the Polish-space boundary of chapter 1490, the effective complete-separable route of chapter 1706, and the represented-space interface of chapter 1494. The packet also consumes the Cauchy-representation surface of chapter 1680, the explicit-modulus Cauchy route of chapter 1705, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It is a BEDC finite naming surface for computable Polish data, not a quotient of names, a selected dense family, host equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3166.1 (Computable Polish space carrier)**.** *A* $\mathsf{ComputablePolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, E, R, A, Q, S, G, D, L, H, K, U, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row, carrying the complete separable metric boundary read by ordinary Polish-space consumers.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *row, exposing the indexed dense family, explicit realizer route, and effective complete-separable metric data.*
3. *R is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *row, presenting the finite name interface through which computable names reach target objects.*
4. *A is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoder row, routing stream observations through regular rational readback, dyadic tolerances, and the represented-space boundary.*
5. *Q is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *row, carrying the explicit-modulus Cauchy sequence data used by the realizer route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the representation decoder and the effective Cauchy route.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and Cauchy stages.*

8. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used for the decoder, regular readback, effective Cauchy modulus, and Real seal thresholds.*
9. *L is the terminal* Real$^{\uparrow}$ *seal row reached only after A, Q, S, G, D have been displayed.*
10. *H records componentwise* hsame *transport for the Polish, effective Polish, represented-space, Cauchy-representation, effective Cauchy, stream-window, regular-readback, dyadic-ledger, Real-seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a computable-name request through R, A, S, Q, G, D, L and back to the Polish and effective-Polish rows P, E.*
12. *U is the* Pkg *provenance over* ComputablePolishSpace$^{\uparrow}$, PolishSpace$^{\uparrow}$, EffectivePolishSpace$^{\uparrow}$, RepresentedSpace$^{\uparrow}$, CauchyRepresentation$^{\uparrow}$, EffectiveCauchySequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{ComputablePolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed rows to a common complete-separable, represented, Cauchy-realizer, stream-window, regular-readback, dyadic-tolerance, and Real-seal surface. It has no coordinate for quotient names, selected dense points, host equality of represented objects, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ComputablePolishSpaceUp.ComputablePolishSpaceTasteGate single carrier alignment`

Theorem 3166.2 (Computable Polish space NameCert obligations)**.** *For every accepted* ComputablePolishSpace$^{\uparrow}$ *carrier $C = (P, E, R, A, Q, S, G, D, L, H, K, U, N)$, the local* NameCert$_{\text{ComputablePolishSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Polish, effective Polish, represented-space, Cauchy-representation, effective Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and local naming rows;*
2. *complete-separable compatibility, requiring P and E to expose the same complete separable metric object through the displayed dense-realizer route;*
3. *representation exactness, requiring every computable name to pass through the represented-space row R and Cauchy-representation decoder A before a target-facing read is available;*
4. *effective Cauchy exactness, requiring the explicit row Q and the finite stream windows S to precede the regular rational readback G, dyadic ledger D, and Real seal L;*
5. *componentwise transport, continuation replay, package provenance, and local naming through H, K, U, N for exactly the displayed rows;*
6. *ledger nonescape, excluding quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No computable-Polish-space obligation is stored outside these rows.

Proof. Project the carrier of Definition 4146.1. The projection exposes the Polish row P, effective Polish row E, represented-space row R, Cauchy-representation decoder A, effective Cauchy row Q, stream windows S, regular readback G, dyadic ledger D, Real seal L, and structural rows H, K, U, N.

The complete-separable compatibility row reads the sibling surfaces of chapter 1490 and chapter 1706. Since P and E occur in the same finite packet, a consumer cannot replace the displayed dense-realizer route by a second dense family or a hidden completion equality.

The representation route is then forced through chapter 1494 and chapter 1680. A computable name first exposes R and A, then reads the finite stream windows rather than an unscheduled name oracle.

The effective Cauchy route uses chapter 1705, chapter 301, chapter 291, and chapter 38. These rows place Q, S, G, D before L, so the Real seal is downstream of the explicit modulus and tolerance ledger.

Finally, H, K, U, N transport, replay, source, and name only that displayed packet. They introduce no quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, or ambient Real completeness, and the listed rows exhaust the local NameCert surface. □

Theorem 3166.3 (Computable Polish space realizer handoff)**.** *Every represented-space, Polish-space, or Real-facing consumer of an accepted* $\mathsf{ComputablePolishSpace}^{\uparrow}$ *carrier* $C = (P, E, R, A, Q, S, G, D, L, H, K,$ *factors through the displayed route*

$$R, A \longmapsto S \longmapsto Q \longmapsto G \longmapsto D \longmapsto L \longmapsto P, E, H, K, U, N.$$

Thus the consumer receives the represented-space name interface, Cauchy-representation decoder, finite StreamName windows, EffectiveCauchySequence modulus route, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, Polish row, effective Polish row, transport, replay, provenance, and local naming before it can inspect the completed computable Polish object.

Proof. Begin with a represented-space, Polish-space, or Real-facing read of the carrier. The only name-facing rows are R and A, so the read must expose the represented-space interface and the Cauchy-representation decoder before any target row can be consumed.

Next replay the finite observation route. The row S supplies the inspected stream windows, while the effective Cauchy row Q supplies the explicit modulus route attached to those windows.

The regular and Real-facing handoff is then fixed by G, D, L. The regular rational readback G is read from the displayed windows and effective Cauchy stages; the dyadic ledger D records the tolerance data; and the Real seal L is terminal for that route.

Only after this handoff can the Polish and effective Polish rows P, E be read as a completed computable Polish object. The structural rows H, K, U, N preserve and name the same route, so no consumer can bypass it through quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{ComputablePolishSpace}^{\uparrow}$ represented-space, Polish-space, or Real-facing read factors through a represented-space row, CauchyRepresentation decoder, StreamName finite windows, EffectiveCauchySequence modulus route, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, PolishSpace row, EffectivePolishSpace row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: polishspace, effective-polish-space, representedspace, cauchy-representation, effective-cauchy-sequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{ComputablePolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries all of their displayed rows as one computable complete-separable name interface.

Closure status for $\mathsf{NameCert}_{\mathsf{ComputablePolishSpace}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\mathsf{ComputablePolishSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4146.1, Definition 4146.2, and Definition 4146.3 seed the finite computable Polish route over PolishSpaceUp, EffectivePolishSpaceUp, RepresentedSpaceUp, CauchyRepresentationUp, EffectiveCauchySequenceUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ComputablePolishSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-separable compatibility, representation exactness, effective Cauchy exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.ComputablePolishSpaceUp.

Constructive story: A ComputablePolishSpace$^{\uparrow}$ packet is generated from PolishSpace, EffectivePolishSpace, RepresentedSpace, CauchyRepresentation, EffectiveCauchySequence, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3167

A Concrete Naming Certificate for $\mathsf{LusinApproximation}^{\uparrow}$

Following Lusin and Bishop–Bridges constructive measure theory, the $\mathsf{LusinApproximation}^{\uparrow}$ packet records the finite route by which a measurable real-valued row is approximated on a large displayed set by a continuous row. It sits after the measurable carrier of chapter 92, the continuous-map surface of chapter 130, the integral readback route of chapter 123, the terminal real-name seal of chapter 38, the finite exceptional-window route of chapter 2204, and the density ledger of chapter 3448. The carrier exposes the approximant, tolerance, large-set, integral, replay, and naming rows needed by later measure-analysis packets; it is not a host regularity theorem, selected null-set completion, quotient almost-everywhere equality, or countable-choice construction.

Definition 3167.1 (Lusin approximation carrier)**.** *A* $\mathsf{LusinApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, A, C, F, E, I, R, H, Q, P, N)$$

with the following displayed rows:

1. *M is the ambient finite* $\mathsf{Measure}^{\uparrow}$ *row, exposing the measurable-set ledger and the finite measure budget against which the exceptional set is read.*
2. *A is the finite large-set and exceptional-window row, carrying the measurable subset on which the approximation is certified and the displayed complement budget.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant row, naming the continuous candidate consumed by the public approximation read.*
4. *F is the measured function row, stored as finite measurable readback data rather than as an ambient almost-everywhere equivalence class.*
5. *E is the error-tolerance row, recording the requested* $\mathsf{Real}^{\uparrow}$ *tolerance, the pointwise comparison window, and the bound used outside the exceptional set.*
6. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row, exposing the finite integral or size comparisons used when a consumer asks for quantitative approximation data.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal that compares the measured function, continuous approximant, tolerance, and integral-facing reads through the displayed finite route.*
8. *H records componentwise* hsame *transport for the measure, large-set, continuous, function, error, integral, Real, replay, provenance, and naming rows.*

9. *Q records displayed* Cont *replay of the route*

$$M, A, C, F, E, I, R \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{LusinApproximation}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Egorov}^{\uparrow}$, $\mathsf{RadonNikodym}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, C, F, E, I, R, H, Q, P, N$. It has no coordinate for a host Lusin theorem, completed measure algebras, selected null sets, quotient almost-everywhere equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LusinApproximationUp.TasteGate.LusinApproximationTasteGate single carrier alignment`

Theorem 3167.2 (Lusin approximation NameCert obligations)**.** *For every accepted* $\mathsf{LusinApproximation}^{\uparrow}$ *carrier*

$$L = (M, A, C, F, E, I, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite* $\mathsf{Measure}^{\uparrow}$ *row M, the measured function row F, and the continuous approximant row C;*
2. *large-set and exceptional-window exposure through A, with the finite complement budget read in the same measure ledger as M;*
3. *tolerance and comparison exactness through E and the* $\mathsf{Real}^{\uparrow}$ *seal R, requiring each public approximation read to display its error budget;*
4. *quantitative readback through I, requiring integral-facing or size-facing consumers to remain on the same finite route;*
5. *structural stability through H, Q, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve exactly the rows M, A, C, F, E, I, R.*

No obligation may be discharged by importing a host Lusin theorem, completing the measure algebra, choosing null representatives, quotienting by almost-everywhere equality, or appealing to countable choice.

Proof. Project the carrier of Definition 4147.1. The measure-facing rows are M and A, the approximation-facing rows are C, F, E, R, and the quantitative readback row is I.

The large-set budget is visible before any approximation conclusion is named. This is the same finite exceptional-window discipline used by chapter 2204, but the public consumer here reads a continuous approximant row rather than a uniform convergence window.

The density and integral-facing dependency is also explicit. The route through I and R mirrors the eventwise readback discipline of chapter 3448, while refusing completed measure algebras and quotient almost-everywhere classes.

Finally H, Q, P, N transport, replay, source, and name only those displayed rows. Since the excluded host theorem, null-set selector, quotient, choice, and completeness coordinates are absent from the carrier, the five listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{LusinApproximation}^{\uparrow}$ packet is generated from finite Measure rows, large-set and exceptional-window ledgers, ContinuousMap approximants, measurable function readback, Real tolerance seals, Integral readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4147.1 and Definition 4147.2 seed the finite large-set continuous-approximant route from Measure and Real tolerance rows to ContinuousMap and Integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host Lusin theorem, completed measure algebra, selected null representatives, quotient almost-everywhere equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, large-set budget exactness, continuous approximant admission, function readback, tolerance sealing, integral readback, transport stability, and replay stability for BEDC.Derived.LusinApproximationUp.

Chapter 3168

A Concrete Naming Certificate for $\mathsf{StreamNameModulus}^{\uparrow}$

Following Bishop and Bridges, a constructive modulus is treated as displayed finite control on later observations rather than as a selected limit object. The $\mathsf{StreamNameModulus}^{\uparrow}$ packet sits between the stream-name observation surface of chapter 126, the dyadic readback rows of chapter 291, the regular rational consumer of chapter 301, and the real-seal boundary of chapter 38. It records the finite witness by which a StreamName observation budget can be handed to RegSeqRat and Real consumers without claiming a completed stream quotient, a host-selected tail, or equality of real representatives.

Definition 3168.1 (Stream-name modulus carrier)**.** *A* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, containing the requested stream name, observed positions, and local observation budget.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row for the inspected positions.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that receives the finite dyadic windows and their regularity ledger.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-request row, recording only the downstream consumer boundary that may later consume R.*
5. *W is the modulus ledger: for each requested precision it displays a finite stream position, the associated dyadic readback, and the regular-rational handoff row visible at that position.*
6. *H records componentwise* hsame *transport for S, D, R, E, W and the local certificate data.*
7. *C records displayed* Cont *replay from a precision request to the stream position, from that position to the dyadic readback, and from the readback to the RegSeqRat handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for a completed stream quotient, an ambient limit selector, a host real equality proof, or a tail chosen outside the displayed observation ledger.

Lean4 ✓ `BEDC.Derived.StreamNameModulusUp.StreamNameModulusTasteGate single carrier alignment`

Theorem 3168.2 (Stream-name modulus pointwise stability)**.** *Let $U = (S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier. If a downstream consumer transports the displayed stream, dyadic readback, RegSeqRat handoff, Real seal-request, and modulus ledger along the componentwise* hsame *rows in H, then each precision request reads the same finite position of S, the same dyadic row of D, and the same regular-rational handoff in R after replay through C.*

Lean4 ✓ `BEDC.Derived.StreamNameModulusUp.StreamNameModulusCarrier pointwise stability`

Proof. Project the carrier of Definition 4148.1. The only precision-facing data are the rows S, D, R, E, W together with transport, replay, provenance, and the local name.

The modulus ledger W binds each requested precision to one displayed stream position and one displayed dyadic readback. Componentwise transport through H moves those rows together, so a transported request cannot alter the position without also altering the visible ledger row.

Replay through C reconstructs the same path from the precision request to S, then to D, then to R. The rows P and N only source and name this finite route; they do not add an external tail or equality proof. □

Theorem 3168.3 (Stream-name modulus monotone reindex stability)**.** *Let $U = (S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier. If a later finite-window request is transported through the modulus ledger W, then the transported request preserves the same* $\mathsf{StreamName}^{\uparrow}$ *point certificate, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal route before any consumer may read a Real-facing conclusion.*

Proof. Project the carrier of Definition 4148.1. The rows S, D, R, E, W display the source point certificate, dyadic readback, regular-rational handoff, Real seal request, and modulus ledger for every accepted precision request.

Apply Definition 4148.2. A transported later request may move along the ledger W, but the corresponding stream position, dyadic readback, and regular-rational handoff remain attached to the same componentwise transport and replay rows.

Finally apply Definition 4148.4. The later finite-window request reaches the RegSeqRat handoff only through S, D, W, H, C, P, N, and the Real seal row E is read only after that handoff. Thus monotone reindexing cannot detach a tail selector from the displayed StreamName point certificate or from the Real-facing route. □

Theorem 3168.4 (Stream-name modulus RegSeqRat readiness)**.** *An accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *packet can be consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *row only as the finite route*

$$S, D, W \longmapsto R$$

together with H, C, P, N. The handoff supplies displayed stream positions, dyadic observations, and regularity ledger rows, but it does not supply a completed sequence quotient, a selected tail, a host convergence theorem, or a real-equality witness.

Proof. Begin with the carrier projection. The RegSeqRat row R is not independent; it is stored beside the stream source S, dyadic readback D, and modulus ledger W.

Use pointwise stability from Definition 4148.2. Every transported precision request still reaches R through the same visible stream position and dyadic observation.

The RegSeqRat consumer therefore receives only the displayed finite route and its support rows. Since the carrier has no coordinate for quotient sequences, selected tails, host convergence, or real equality, none of those objects can be exported at the RegSeqRat boundary. □

Theorem 3168.5 (Stream-name modulus Real handoff)**.** *A* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *packet may import only the RegSeqRat-facing handoff R, the Real seal-request row E, and the finite stream/dyadic modulus route S, D, W, H, C, P, N. The packet therefore prepares a RealUp handoff by finite observed windows and regular rational rows, without asserting completeness of* $\mathsf{Real}^{\uparrow}$*, equality of Real representatives, or convergence outside the carried ledger.*

Proof. Apply Definition 4148.4. It identifies R as the only regular-rational object exported by the packet and forces the route through S, D, W, H, C, P, N.

The Real-facing row E records only that this regular-rational handoff is the object a Real consumer may later inspect. It is a seal-request row, not a proof of Real completeness or equality.

Thus the Real consumer receives the same finite modulus route that the RegSeqRat consumer receives, together with the request row E. No ambient limit, quotient equality, or external convergence datum is available. □

Theorem 3168.6 (Stream-name modulus NameCert obligations)**.** *For every accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier $U = (S, D, R, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-position exposure, dyadic readback, RegSeqRat handoff, Real seal-request boundary, modulus-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is discharged inside the displayed finite route and no obligation is stored outside U.*

Proof. Carrier admission is the projection of the finite tuple in Definition 4148.1. The rows S, D, R, E, W expose the stream source, dyadic readback, regular-rational handoff, Real request, and modulus ledger.

Pointwise stability supplies the transport obligation. Theorem 4148.2 keeps precision requests attached to the same displayed stream and dyadic rows under hsame movement.

The RegSeqRat readiness and Real handoff theorems supply the consumer-facing obligations. They show that R and E can be read only through the carried finite route.

Finally C, P, N replay, source, and name the packet. Since no remaining carrier coordinate exists, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{StreamNameModulus}^{\uparrow}$ consumer that reads a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff without first exposing the displayed $\mathsf{StreamName}^{\uparrow}$ positions, $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks, modulus ledger, componentwise hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row refutes the packet surface.

Independence witness: streamname, dyadicratcore, regseqrat, real: none of the listed sibling chapters supplies the whole modulus handoff $U = (S, D, R, E, W, H, C, P, N)$, because each sibling contributes only the source stream, dyadic readback, regular rational consumer, or Real seal boundary while this chapter binds them into one finite modulus route.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$.

Constructive story: A StreamNameModulus$^{\uparrow}$ packet is generated from a StreamName$^{\uparrow}$ source row, DyadicRatCore$^{\uparrow}$ readbacks, a RegSeqRat$^{\uparrow}$ handoff, a Real$^{\uparrow}$ seal-request row, a modulus ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{StreamNameModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4148.1, Definition 4148.2, Definition 4148.4, Definition 4148.5, and Definition 4148.6 seed the finite modulus handoff from StreamName$^{\uparrow}$ observations to RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers.

Formal verification: FormalStatus(NameCert$_{\text{StreamNameModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, selected tails, host convergence, Real equality, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite modulus ledger, RegSeqRat readiness, and Real handoff obligations for BEDC.Derived.StreamNameModulusUp.

Chapter 3169

A Concrete Naming Certificate for $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$

The $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ packet realizes the finite verification trace of Definition 9393.4 as a concrete NameCert surface. It sits beside the finite-trace GAP socket of chapter 504 and the closed observation ledger of chapter 616. Its role is to filter observed trace rows through finite kernel checks while keeping the universal-induction boundary visible as a GAP-facing refusal.

Definition 3169.1 (Finite observation trace filter carrier)**.** *A* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, T, K, A, G, H, C, P, N)$$

with the following displayed rows:

1. *O is the closed-observation source row containing the finite observed states and their monotone ledger.*
2. *T is the finite verification trace row, listing inspected indices, their predicates, and the closed-generation checks visible at those indices.*
3. *K is the kernel-filter row that records which trace entries are accepted by finite verification and which are rejected as outside the carried window.*
4. *A is the audit row for non-escape: it records that no universal quantifier, induction schema, future-observation oracle, or hidden host rule is imported by the filter.*
5. *G is the GAP-socket row exposed when a consumer asks the finite trace to act as a universal statement.*
6. *H records componentwise* hsame *transport for O, T, K, A, G and the local certificate data.*
7. *C records displayed* Cont *replay from an observation request to the trace row, from the trace row to the kernel filter, and from a universal request to the GAP row.*
8. *P is the* Pkg *provenance over* BHist*, closed observation, finite-trace rows, GAP sockets,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these rows. It has no coordinate for a future observation, a substrate-internal induction principle, a universal closure over all indices, or a host-level truth predicate.

Theorem 3169.2 (Finite trace filter acceptance boundary)**.** *For every accepted* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *carrier* $F = (O, T, K, A, G, H, C, P, N)$*, a trace entry is available to a downstream consumer only after it appears in the finite trace row* T*, is checked by the kernel-filter row* K*, and is replayed through* C *with provenance* P *and local name* N*. Entries outside the displayed trace are not refused as false; they are routed to the GAP row* G *when a consumer asks for universal closure.*

Proof. Project the carrier. The observed source is O, the trace is T, and the filter row is K; no other row stores accepted trace entries.

The replay row C makes the accepted route explicit. It starts from the observation request, reads the finite trace entry, and then checks that entry against K.

When the requested entry is not in the displayed trace, the packet exposes the GAP row G rather than converting absence into a universal claim. The audit row A records this non-escape boundary, while P, N source and name the finite route. □

Theorem 3169.3 (Finite trace filter NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$ *surface consists exactly of closed-observation source admission, finite trace exposure, kernel-filter acceptance, non-escape audit, GAP-socket routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations certify finite trace filtering only; they do not certify induction, field completeness, universal closure, or future-observation coverage.*

Proof. Carrier admission exposes the tuple

$$F = (O, T, K, A, G, H, C, P, N).$$

The source, trace, filter, audit, and GAP rows are the only content rows.

Apply the acceptance boundary of Definition 4149.2. It identifies the accepted route through T, K, C, P, N and routes universal requests to G.

Transport through H preserves this row structure componentwise. Since the carrier has no row for induction, field completeness, universal closure, or future observations, the listed rows exhaust the NameCert obligations. □

Theorem 3169.4 (Finite trace filter TasteGate worklist)**.** *Any Lean TasteGate implementation for* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *must separate finite trace acceptance from algebraic adequacy by carrying explicit* `Nontrivial` *and* `FieldFaithful` *instance obligations for the numeric side of the filter. The paper packet records the finite trace boundary only; those instances are downstream formal work and cannot be inferred from closed observation rows alone.*

Lean4 ✓ `BEDC.Derived.FiniteObservationTraceFilterUp.FiniteObservationTraceFilterTasteGate`
`single carrier alignment`

Proof. The filter carrier records finite observation, trace, kernel acceptance, audit, GAP, transport, replay, provenance, and naming rows. None of these rows asserts that a numeric carrier has more than one element or that field operations are faithful to the intended readback.

The acceptance boundary therefore cannot discharge `Nontrivial` or `FieldFaithful` by projection. Those obligations must be visible in the Lean TasteGate work for any numeric consumer of the filter.

This keeps the concrete packet honest: it certifies finite observation filtering, while algebraic adequacy remains a separate formal instance surface. □

Theorem 3169.5 (Finite trace filter GAP handoff)**.** *If a downstream consumer treats an accepted* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *packet as evidence for a universal statement, the packet exports only the finite accepted rows and the GAP socket G. It supplies no substrate-internal induction rule, no future-observation oracle, no total trace closure, and no host truth predicate.*

Proof. Start with the NameCert obligations. They expose finite source, trace, filter, audit, GAP, transport, replay, provenance, and naming rows.

A universal request is not one of the accepted trace entries. By the acceptance boundary, such a request is routed to G rather than accepted by K.

The non-escape audit row A records that this is a refusal boundary. Since the carrier has no row for induction, future observations, total closure, or a host truth predicate, the GAP handoff is the only universal-facing export. □

Falsifiable prediction: An accepted $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ consumer that obtains a universal statement, induction rule, future-observation value, or host truth predicate without passing through the displayed GAP row G refutes the filter surface.

Independence witness: finite-trace-gap-socket, closedobservationsystem: the GAP socket names the universal request boundary and the closed-observation chapter names monotone finite records, but neither chapter supplies the kernel-filter row K and audit row A that classify accepted finite trace entries.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ packet is generated from closed-observation rows, finite verification trace rows, kernel-filter acceptance rows, audit rows, GAP socket rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4149.1, Definition 4149.2, Definition 4149.3, Definition 4149.4, and Definition 4149.5 seed the finite trace filter boundary for closed observation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim induction, universal trace closure, future-observation coverage, field adequacy, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove finite trace acceptance, GAP handoff, and the TasteGate instance boundary for BEDC.Derived.FiniteObservationTraceFilterUp.

Chapter 3170

A Concrete Naming Certificate for $\mathsf{LocatedCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompletion}^{\uparrow}$ packet records the finite located completion surface used by the L10 Real and Completion route. It sits between the located metric source of chapter 1533, the separated completion boundary of chapter 1612, the dense completion entry of chapter 1512, and the located completion space interface of **??**. The packet is a displayed NameCert route for located completion data; it is not a quotient completion, a selected representative, a countable-choice schedule, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3170.1 (LocatedCompletion carrier)**.** *A* $\mathsf{LocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, E, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LocatedMetric}^{\uparrow}$ *source row, including the metric carrier, located distance comparison, and finite request discipline consumed before completion data are public.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row, read before completed names may be compared as separated.*
3. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row by which source observations enter the completion-facing route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed before any regular rational readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling located comparison, Cauchy tail requests, and zero-distance checks.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, S, E, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the located metric source, separated boundary, embedding, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion request through the same ordered rows.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletion}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, CompletionEmbedding $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ *row.*
The classifier compares accepted carriers only through $M, S, E, W, R, D, A, H, C, P, N$. It has no coordinate for quotient completion equality, selected completed representatives, host equality, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompletionUp.LocatedCompletionTasteGate single carrier alignment`

Theorem 3170.2 (LocatedCompletion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletion}^\uparrow$ *carrier $L = (M, S, E, W, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ *surface consists exactly of carrier admission, located metric source exposure, separated completion exposure, dense embedding coverage, finite stream-window exactness, regular rational readback, dyadic tolerance accounting, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or completion-facing consumer must read those rows in the displayed route before it can use the local name.*

Lean4 ✓ `BEDC.Derived.LocatedCompletionUp.LocatedCompletionCarrier namecert obligations`

Proof. Project the carrier of Definition 4150.1. The displayed analytic rows are the located metric source M, separated completion boundary S, completion embedding E, stream windows W, regular readback R, dyadic tolerance ledger D, and Real seal A.

Read the source and embedding side first. The $\mathsf{LocatedMetric}^\uparrow$ row supplies located distance data, the $\mathsf{CompletionEmbedding}^\uparrow$ row admits source observations, and the $\mathsf{SeparatedCompletion}^\uparrow$ row supplies the separated comparison boundary named in chapter 1612.

The numerical route is finite. The stream windows W and dyadic ledger D precede the $\mathsf{RegSeqRat}^\uparrow$ readback R, and R precedes the terminal $\mathsf{Real}^\uparrow$ seal A.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same route, while the carrier lists no quotient completion, selected representative, host equality, choice schedule, or ambient completeness coordinate. Hence the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ obligations. □

Theorem 3170.3 (LocatedCompletion L10 handoff)**.** *Every* $\mathsf{LocatedCompletion}^\uparrow$ *consumer route factors through the finite L10 chain*

$$\mathsf{LocatedMetric}^\uparrow \longmapsto \mathsf{CompletionEmbedding}^\uparrow \longmapsto \mathsf{SeparatedCompletion}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow, \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{RegS}$$

The handoff is the same route consumed by the located completion space chapter of **??***; it exports no quotient completion, selected completed point, host equality, countable-choice selector, or ambient completeness theorem.*

Proof. Use Definition 4150.2 to expose the full packet route. The located metric row, dense embedding, separated boundary, finite windows, tolerance ledger, readback, and Real seal must all be displayed before the local name is available.

Compare this route with **??**. The located completion space interface consumes the same sibling rows but packages them as a space-facing boundary, while the present chapter packages them as the local located completion object.

No additional completion principle remains. The structural rows H, C, P, N merely transport, replay, source, and name the displayed chain, and the refused quotient, selection, equality, choice, and ambient-completeness objects have no coordinate in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletion}^{\uparrow}$ consumer displays a $\mathsf{LocatedMetric}^{\uparrow}$ source row, a $\mathsf{CompletionEmbedding}^{\uparrow}$ dense entry row, a $\mathsf{SeparatedCompletion}^{\uparrow}$ boundary row, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: locatedmetric, separatedcompletion, completionembedding, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located metric source data, dense entry, separated completion boundary, finite stream windows, dyadic tolerance, regular readback, Real sealing, transport, continuation, provenance, and local naming as one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4150.1, Definition 4150.2, and Definition 4150.3 seed the finite located completion surface over $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected completed representatives, host equality of completed points, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, separated completion compatibility, embedding coverage, finite readback exactness, Real seal exactness, transport replay, and non-escape over the same displayed packet.

Constructive story: A $\mathsf{LocatedCompletion}^{\uparrow}$ packet is generated from a located metric source, a dense completion entry, a separated completion boundary, finite stream windows, dyadic tolerances, regular rational readback, a Real seal, and structural NameCert rows.

Chapter 3171

A Concrete Naming Certificate for $\mathsf{SeparatedReflection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SeparatedReflection}^{\uparrow}$ packet records the boundary where a pseudometric source exposes zero-distance evidence before any consumer treats that evidence as point identity. It starts from the displayed source rows of chapter 1614, routes the zero-distance boundary through the separated classifier of chapter 1411, and hands only that routed surface to the completion consumer of chapter 1506. Its finite readback remains inside the stream windows of chapter 126, the regular rational route of chapter 301, and the terminal real-name seal of chapter 38. The packet therefore names the constructive pseudometric-to-separated boundary without quotient primitives, selected representatives, host equality, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3171.1 (SeparatedReflection carrier)**.** *A* $\mathsf{SeparatedReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (P_0, Z, S, M, R, E, H, C, N)$$

with the following displayed rows:

1. *P_0 is the source* $\mathsf{Pseudometric}^{\uparrow}$ *packet, including point admission, displayed distance witnesses, dyadic tolerance and triangle ledgers, stream windows, regular rational readback, a real-distance seal, and the zero-distance boundary available before separation.*
2. *Z records the zero-distance boundary as evidence read from P_0. It does not identify source point rows and does not introduce a quotient class.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier handoff. It is the first row allowed to turn zero-distance evidence into separated point-classifier evidence.*
4. *M is the metric-completion readback row. It hands the separated classifier surface to* $\mathsf{MetricCompletion}^{\uparrow}$ *only after S has been displayed.*
5. *R is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window route available to completion and real-readback consumers.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the displayed finite and regular rational rows.*
7. *H stores componentwise* hsame *transport for the pseudometric source, zero-distance boundary, separated classifier, completion readback, finite-window route, real seal, replay, provenance, and local naming rows.*

8. *C stores the displayed* Cont *replay from pseudometric inspection through zero-distance evidence, separated classifier handoff, completion readback, finite windows, regular rational readback, and real sealing.*

9. *N is both the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows and the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$ *row.*

The carrier is a finite reflection-boundary certificate. It has no quotient-completion coordinate, no host-equality coordinate, no selected representative, no countable-choice selector, no trichotomy row, no open-cover compactness principle, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SeparatedReflectionUp.SeparatedReflectionTasteGate single carrier alignment`

Theorem 3171.2 (SeparatedReflection NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *carrier* $P = (P_0, Z, S, M, R, E, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$ *surface consists exactly of the displayed rows for source pseudometric admission, zero-distance boundary exposure, separated carrier handoff, metric-completion readback, StreamName-RegSeqRat-Real sealing, componentwise transport, continuation replay, provenance, and local naming. No obligation may be discharged by a quotient completion, host equality, selected representative, countable-choice selector, trichotomy split, open-cover compactness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge row.*

Proof. Project the carrier of Definition 4151.1. The tuple displays the source pseudometric packet, zero-distance row, separated classifier, completion readback, finite-window and regular-rational route, real seal, transport, replay, provenance, and naming rows.

The obligation order is the order of those displayed rows. The zero-distance evidence is first held in Z, the separated carrier handoff is S, and completion-facing readback is delayed until M reads the routed separated surface.

The finite numerical readback is confined to R and E, while H, C, N provide transport, replay, provenance, and local naming. Since none of the excluded principles is a field of the carrier, none can serve as a local NameCert obligation. □

Theorem 3171.3 (SeparatedReflection zero-distance boundary)**.** *Let* $P = (P_0, Z, S, M, R, E, H, C, N)$ *be an accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *packet. Every consumer that reads zero-distance evidence from the pseudometric source receives that evidence first as the displayed boundary row* Z*. If the consumer asks for point identity, the request must pass through the separated classifier row* S *before any metric, completion, regular-rational, or real-readback row can consume it.*

Proof. Start with the source row P_0. It supplies the pseudometric distance surface and exposes zero-distance evidence without identifying source point histories.

The boundary row Z records exactly that evidence. It is not a point-identity row, so a consumer that needs identity has no route inside P_0 or Z.

The first identity-facing coordinate is S. Therefore every downstream metric, completion, regular-rational, or real-readback consumer receives the zero-distance claim only after the separated classifier handoff has been displayed. □

Theorem 3171.4 (SeparatedReflection completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *packet reads the route*

$$\mathsf{Pseudometric}^{\uparrow} \to Z \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus completion consumers obtain only the separated carrier handoff and its finite readback; they do not obtain quotient completion, host equality, selected representatives, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the reflection packet.*

Proof. Read the displayed carrier order. The pseudometric source precedes the zero-distance row, the zero-distance row precedes the separated classifier, and the separated classifier precedes the completion readback.

The numerical part of the read is then forced through R and E. Stream windows and regular rational approximants are displayed before the real seal, so completion-facing consumers see a finite readback route rather than a detached real object.

The exclusions are inventory facts. No quotient completion, host-equality proof, selected representative, countable-choice selector, trichotomy split, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is present in P, so no completion handoff can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedReflection}^{\uparrow}$ packet is generated from a source $\mathsf{Pseudometric}^{\uparrow}$ packet, a displayed zero-distance boundary row, a $\mathsf{SeparatedMetric}^{\uparrow}$ classifier handoff, a $\mathsf{MetricCompletion}^{\uparrow}$ readback row, local StreamNameRegSeqRat-Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4151.1, Definition 4151.2, Definition 4151.3, and Definition 4151.4 seed the finite boundary between pseudometric zero-distance evidence and separated completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, host equality, selected representatives, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance boundary exposure, separated carrier handoff, completion readback, finite real sealing, transport replay, provenance visibility, and non-escape for BEDC.Derived.SeparatedReflecti

Chapter 3172

A Concrete Naming Certificate for $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$

Following Picard and Lindelof, local flow data are treated here as a finite constructive package: a metric phase row, a rational time-window row, a local Lipschitz budget, an Euler-style approximation schedule, and a replay ledger that records how each requested window is inspected. The $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ packet names the local existence interface for ODE consumers without importing host flows, selected maximal intervals, ambient completeness, or global continuation. Its local-flow route consumes the Lipschitz budget surface of chapter 3724 together with the map boundary of chapter 410, the phase metric row of chapter 57, the regular time-window readback of chapter 301, and the finite schedule surface of chapter 126.

Definition 3172.1 (Constructive Lipschitz flow carrier)**.** *A* $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, T, L, E, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *phase row containing the point-history carrier, metric readbacks, distance classifier, and local metric* $\mathsf{NameCert}$*.*
2. *T is the rational time-window row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *windows and not through a host interval object.*
3. *L is the local Lipschitz budget row; it records the radius, time-length, and contraction budget inspected by the packet.*
4. *E is the finite approximation schedule, read as a* $\mathsf{StreamName}^{\uparrow}$ *sequence of local Euler windows.*
5. *W is the observation ledger joining metric-radius requests, time windows, Lipschitz-budget checks, and approximation-window readbacks.*
6. *H records componentwise* hsame *transport for M, T, L, E, W and the local certificate rows.*
7. *C records the displayed* Cont *replay from a requested local flow window to its metric, time, Lipschitz, and approximation rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by transporting these displayed rows to a common metric, time-window, Lipschitz-budget, and approximation-schedule surface. It has no coordinate for a selected host solution curve, maximal interval, quotient trajectory, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ConstructiveLipschitzFlowUp.TasteGate.ConstructiveLipschitzFlowTasteGate`
`single carrier alignment`

Theorem 3172.2 (Constructive Lipschitz flow NameCert obligations)**.** *For every accepted* ConstructiveLipschitzFlow *carrier* $F = (M, T, L, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$ *surface consists exactly of the carrier admission row, classifier-transport row, local Lipschitz-budget row, finite approximation-schedule row, observation-ledger exactness row, continuation-replay row, package-provenance row, and local naming row. These rows bind* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *as the whole public obligation surface.*

Proof. Project the carrier of Definition 4152.1. The projection displays the phase metric row M, time-window row T, Lipschitz budget L, approximation schedule E, observation ledger W, transport row H, continuation row C, provenance row P, and local name N.

Classifier transport uses H to keep all rows over one metric and one time-window surface. The Lipschitz budget is therefore read with the same metric-radius and rational time data as the approximation schedule.

The ledger exactness clause is exhausted by W. Every local-flow request must pass through the displayed metric, time, budget, and schedule windows before it reaches the Cont replay row.

Finally attach P and N. These rows source and name the displayed packet; they add no host solution curve, selected maximal interval, quotient trajectory, or ambient completeness principle. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 3172.3 (Constructive Lipschitz flow local-window boundary)**.** *Every downstream ODE consumer of an accepted* $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ *carrier* $F = (M, T, L, E, W, H, C, P, N)$ *must read local flow data through the displayed metric row* M*, rational time-window row* T*, Lipschitz budget* L*, approximation schedule* E*, observation ledger* W*, componentwise transport* H*, continuation replay* C*, package provenance* P*, and local* NameCert *row* N*. The packet supplies no global flow, no selected maximal interval, no quotient trajectory equality, no closed completeness of the phase space, and no host-level uniqueness theorem.*

Proof. Start from the NameCert obligation surface of Definition 4152.2. It exposes only the metric, time, budget, schedule, ledger, transport, replay, provenance, and naming rows.

A local-window request first fixes its rational time window and its metric radius through T and M. The Lipschitz budget L and approximation schedule E can then be inspected only through the ledger W.

The continuation row C replays this finite inspection route, while H keeps the rows componentwise stable and P, N record provenance and naming. Since no carrier coordinate stores a host flow or maximal interval, the downstream consumer receives only the displayed local-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ packet is generated from a phase metric row, rational time-window row, local Lipschitz budget, finite approximation schedule, observation ledger, componentwise transport, displayed continuation replay, package provenance, and local naming data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4152.1, Definition 4152.2, and Definition 4152.3 define the local flow carrier and seed the ODE-facing local-window route over $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConstructiveLipschitzFlow}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close a Lean target, global flows, selected maximal intervals, quotient trajectories, phase-space completeness, host uniqueness, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, classifier transport, Lipschitz-budget exactness, approximation-schedule replay, and local-window non-escape for the same packet.

Chapter 3173

A Concrete Naming Certificate for $\mathsf{LebesgueDifferentiation}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive measure theory, the $\mathsf{LebesgueDifferentiation}^{\uparrow}$ packet records the finite differentiation-at-a-window surface obtained from measurable local averages, finite Vitali subcover ledgers, and Hardy–Littlewood maximal domination rows. It sits after the measurable-event and integral rows of chapter 92, the finite covering ledger of chapter 2166, the maximal-function domination packet of chapter 3581, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite local differentiation certificate; it is not measure completion, an almost-everywhere quotient, a countable-choice selection, or a host theorem about all measurable functions.

Definition 3173.1 (Lebesgue differentiation carrier)**.** *A* $\mathsf{LebesgueDifferentiation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, V, H_0, A, D, E, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, exposing only the finite measurable-event and local-measure requests inspected by the packet.*
2. *F is the displayed function-read row whose local averages are tested through the measure row.*
3. *V is the* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *ledger, recording the finite disjoint subfamily, cover cells, and overlap budget used by the differentiation window.*
4. *H_0 is the* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *domination row that bounds the exceptional local-average requests before a differentiation verdict is consumed.*
5. *A records the finite local-average rows for the inspected balls or intervals.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare radii, averages, exceptional budgets, and real-name windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the value and error-bound reads.*
8. *T records componentwise* hsame *transport for the measure, function, Vitali, maximal, average, dyadic, real, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route from finite local-average requests through M, F, V, H_0, A, D, E.*

10. *P is the* Pkg *provenance over* LebesgueDifferentiation$^{\uparrow}$, Measure$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, HardyLittlewoodMaximal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{LebesgueDifferentiation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, V, H_0, A, D, E, T, C, P, N$. It has no coordinate for a completed measure algebra, an almost-everywhere quotient, a countable family of selected radii, a selected representative of a function, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LebesgueDifferentiationUp.LebesgueDifferentiationTasteGate single carrier alignment`

Theorem 3173.2 (Lebesgue differentiation NameCert obligations)**.** *For every accepted* LebesgueDifferentiation$^{\uparrow}$ *carrier*

$$L = (M, F, V, H_0, A, D, E, T, C, P, N),$$

the local NameCert$_{\text{LebesgueDifferentiation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, displayed function-read row, finite Vitali-cover ledger, Hardy–Littlewood maximal domination row, local-average rows, dyadic tolerance ledger, real seal, componentwise transport, displayed continuation, package provenance, and local naming.*

Proof. Project the carrier of Definition 4153.1. The rows $M, F, V, H_0, A, D, E, T, C, P, N$ are the whole displayed packet.

The measurable and averaging data are read first. The measure row M admits the inspected event and local-measure requests, while F and A supply the displayed function windows and average rows.

The covering and domination rows then constrain the differentiation request. The Vitali ledger V supplies the finite cover selection surface, and the Hardy–Littlewood row H_0 supplies the maximal domination boundary before the real-valued verdict is exposed.

Finally D, E, T, C, P, N compare, seal, transport, replay, source, and name the same finite route. Since no additional coordinate is present, the NameCert surface exports no measure completion, almost-everywhere quotient, countable-choice selector, selected representative, or ambient Real$^{\uparrow}$ completeness. □

Theorem 3173.3 (Lebesgue differentiation Vitali-maximal handoff)**.** *Every differentiation read accepted by a* LebesgueDifferentiation$^{\uparrow}$ *packet factors through the finite Vitali-cover ledger of chapter 2166 and the maximal-domination surface of chapter 3581 before it reaches the* Measure$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows used by downstream measure and real-name consumers.*

Proof. Start with an accepted carrier $L = (M, F, V, H_0, A, D, E, T, C, P, N)$. A differentiation read reaches the local averages only through the measure and function rows M, F, A.

The finite covering step is then forced through V. It supplies the disjoint subfamily, cover-cell ledger, and overlap budget used by the local differentiation window.

The maximal-function step is forced through H_0. It bounds the exceptional average requests before the dyadic comparison row D and terminal real seal E can be consumed.

Thus the accepted read follows the ordered route through finite Vitali cover, Hardy–Littlewood maximal domination, measure, dyadic tolerance, and real-name rows, with T, C, P, N preserving the route. No countable selection, measure completion, almost-everywhere quotient, selected function representative, or ambient completeness row can be inserted into that handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueDifferentiation}^{\uparrow}$ packet is generated from a MeasureUp source row, displayed function-read rows, a FiniteVitaliCover ledger, HardyLittlewoodMaximal domination, local-average rows, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4153.1, Definition 4153.2, and Definition 4153.3 seed the finite differentiation-window route over MeasureUp, FiniteVitaliCoverUp, HardyLittlewoodMaximalUp, DyadicRatCoreUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, almost-everywhere quotient equality, countable-choice radius selection, selected function representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Vitali-cover handoff, maximal-domination handoff, dyadic tolerance sealing, real-name nonescape, and local naming for BEDC.Derived.LebesgueDifferentiationUp.

Chapter 3174

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$

The $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ packet realizes the observer-shadow reading of chapter 9463 as finite NameCert data. It consumes the visual-shadow metric surface of chapter 4142, the general metric rows of chapter 57, finite observation windows from chapter 126, regular rational readbacks from chapter 301, and terminal real-name seals from chapter 38. The packet names the finite visual shadow seen by a focused observer route; it is not a global hyperbolic model, a physical observer trajectory, a quotient boundary, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3174.1 (Hyperbolic observer-shadow metric carrier)**.** *A* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, M, W, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *source row, supplying the boundary-shadow visual metric packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used when the observer-shadow read is consumed as a metric datum.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for the visual-shadow and metric reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed shadow distance.*
6. *F is the focused observer-shadow row, recording which observer-facing finite route is being inspected.*
7. *H records componentwise* hsame *transport for the visual-shadow, metric, window, readback, real, focus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the focused observer-shadow request through S, M, W, R, E, F.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, W, R, E, F, H, C, P, N$. It has no coordinate for a completed hyperbolic space, quotient boundary equality, physical observer path, hidden global geometry, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.HyperbolicObserverShadowMetricUp.HyperbolicObserverShadowMetricTasteGate`
`single carrier alignment`

Theorem 3174.2 (Hyperbolic observer-shadow metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *carrier*

$$O = (S, M, W, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$ *surface consists exactly of visual-shadow admission, metric comparison admission, finite stream-window exposure, regular-rational readback, terminal real sealing, focused observer-shadow routing, componentwise transport, displayed replay, package provenance, and local naming through* $S, M, W, R, E, F, H, C, P, N$.

Proof. Project the carrier of Definition 4154.1. The listed rows are the whole observer-shadow metric packet.

The visual-shadow row S is read before the metric row M. This keeps the metric comparison tied to the finite visual-shadow carrier rather than to a host hyperbolic model.

The window, readback, and real-seal rows W, R, E then expose the numerical shadow distance through the same L10 real-name route. The focused row F records the observer-facing route being inspected.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert surface exports no completed hyperbolic space, quotient boundary equality, physical observer trajectory, hidden global geometry, or ambient real completeness. □

Theorem 3174.3 (Hyperbolic observer-shadow L10 handoff)**.** *Every* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *consumer that reads a shadow distance factors through* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the focused observer-shadow route can be consumed as a finite NameCert datum.*

Proof. Begin with an accepted carrier $O = (S, M, W, R, E, F, H, C, P, N)$. The source visual-shadow metric row S supplies the displayed boundary-shadow packet.

The metric comparison row M is then consumed through the finite metric surface, and the L10 numeric read follows the window, readback, and real-seal route W, R, E.

Only after these rows are displayed does the focused row F expose the observer-shadow route. Transport, continuation, provenance, and local naming are supplied by H, C, P, N, so the handoff remains finite and cannot pass through a host global geometry or quotient boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ packet is generated from a HyperbolicVisualShadowMetric row, a MetricUp comparison row, StreamName windows, RegSeqRat readback, a RealUp seal, a focused observer-shadow row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4154.1, Definition 4154.2, and Definition 4154.3 seed the finite observer-shadow metric route over visual-shadow, metric, stream-window, regular-rational, and real-name rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed hyperbolic geometry, quotient boundary equality, physical observer trajectories, hidden global geometry, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-shadow source admission, metric comparison, stream-window readback, real-name sealing, focused observer routing, and local naming for BEDC.Derived.HyperbolicObserverShadowMet

The RegularDoubleCauchy$^{\uparrow}$ route reads the rectangular StreamName$^{\uparrow}$/RegSeqRat$^{\uparrow}$/DyadicRatCore$^{\uparrow}$ surface as a sibling of chapter 460 before the RealCauchyCompletion$^{\uparrow}$ and Real$^{\uparrow}$ seals are available.

Chapter 3175

A Concrete Naming Certificate for $\mathsf{RegularDoubleCauchy}^{\uparrow}$

Following Pringsheim's regular convergence of double sequences, Moore–Osgood interchange discipline, and Bishop–Bridges constructive Cauchy completion, the $\mathsf{RegularDoubleCauchy}^{\uparrow}$ packet records a two-parameter regular Cauchy certificate that does not collapse immediately to a diagonal route. It refines the diagonal handoff of chapter 460 by keeping row and column stability visible before the downstream completion consumer reads the terminal seal. The packet sits over the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion-facing route of chapter 360. It provides Moore–Osgood and completion consumers with a finite rectangular regularity surface without asserting host double-sequence convergence, quotient arrays, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 3175.1 (RegularDoubleCauchy carrier)**.** *A* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, D, T, W, H, C, P, N)$$

with the following visible rows: S *is the* $\mathsf{StreamName}^{\uparrow}$ *family schedule for finite rectangular windows;* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback attached to every admitted row and column face;* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rectangular tolerance ledger;* T *is the diagonal cofinal schedule read only after the rectangular ledger has been displayed;* W *is the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff;* H *records componentwise* hsame *transport;* C *records displayed* Cont *replay;* P *is the* Pkg *provenance over* $\mathsf{RegularDoubleCauchy}^{\uparrow}$*,* $\mathsf{CauchyDoubleSequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*; and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *row.*

The carrier is rectangular before it is diagonal. It contains no coordinate for a host double limit, quotient equality of arrays, a selected limiting value, countable-choice scheduling, or ambient real completeness.

Proof. Read the packet as a displayed BHist row. The entries S, R, D expose the finite rectangular data before the diagonal schedule T is available, so the consumer cannot begin with a collapsed single sequence.

The completion-facing row W is downstream from the same visible schedule, readback, and tolerance ledger. Transport and replay are exhausted by H and C, while provenance and naming are exhausted by P and N.

Since the listed tuple has no further coordinate, the excluded host convergence, quotient-array, selected-limit, choice, and ambient-completeness data cannot be read from the carrier. □

Theorem 3175.2 (RegularDoubleCauchy row-column stability)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier* $A = (S, R, D, T, W, H, C, P, N)$*, any two finite reads that agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *rectangular window schedule and on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger have the same row and column* $\mathsf{RegSeqRat}^{\uparrow}$ *readback under the componentwise* hsame *transport recorded in* H*. The stability is local to the finite rows* S, R, D, H, C, P, N *and does not identify quotient arrays or assert a host double-sequence limit.*

Proof. Project the schedule and ledger rows. The rectangular window family is supplied by S, and the tolerance comparison used for every row and column face is supplied by D.

Apply the readback row R to the same displayed windows and ledger entries. Componentwise hsame transport in H preserves the row face and the column face before any completion handoff is inspected.

The replay row C repeats this route for a consumer request, and P, N keep the request inside the local packet. No quotient-array or host-limit coordinate is present, so the stability conclusion is exactly the visible row-column readback. □

Theorem 3175.3 (RegularDoubleCauchy diagonal readback)**.** *Every diagonal read of an accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier factors through the rectangular route*

$$S, D \longmapsto R \longmapsto T \longmapsto W.$$

Thus a $\mathsf{CauchyDoubleSequence}^{\uparrow}$*,* $\mathsf{MooreOsgood}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer sees the diagonal cofinal schedule only after the StreamName windows, RegSeqRat row-column readback, and DyadicRatCore rectangular tolerance ledger have been displayed.*

Proof. Start with the rectangular rows. A diagonal request must first open the finite windows in S and account for the requested precision in D.

The row-column readback R is then the only route from those rectangular windows to a regular rational name. The diagonal schedule T is read after this regularity data, not before it.

Finally the completion-facing handoff W consumes the scheduled readback. Therefore every cited downstream consumer receives the diagonal route as a readback of the rectangular certificate, not as an independent host convergence assertion. □

Theorem 3175.4 (RegularDoubleCauchy rectangular modulus extraction)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier* $A = (S, R, D, T, W, H, C, P, N)$*, the rows* S, D, R, H, C, P, N *determine a finite rectangular modulus surface before* T *or* W *may be consumed. Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read of* A *factors through that surface, so the diagonal schedule and terminal seal are downstream of the displayed StreamName windows, DyadicRatCore tolerance ledger, row-column RegSeqRat readback, componentwise transport, replay, provenance, and local naming rows.*

Proof. Project the carrier rows S, D, R, H, C, P, N. The StreamName row supplies the rectangular windows, the DyadicRatCore row supplies the tolerance ledger, and the RegSeqRat row reads row and column faces against those displayed budgets.

Apply Definition 4155.2. It binds finite rectangular windows and dyadic tolerances to the row-column readback under the same componentwise hsame transport, with C, P, N replaying and naming the route.

Only after this rectangular surface is displayed may Definition 4155.3 read the diagonal schedule T and the completion-facing handoff W. The extracted surface contains no selected double limit,

quotient array, choice schedule, or ambient Real-completeness row, so every RealCauchyCompletion or Real read factors through the finite modulus surface first. □

Theorem 3175.5 (RegularDoubleCauchy ledger exactness)**.** *In an accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier, the dyadic tolerance ledger D is the unique finite row from which a completion-facing consumer may justify rectangular Cauchy bounds before reading the diagonal schedule T or the seal handoff W. Every precision request must pass through S, D, R with the same H, C, P, N provenance, and no selected limit, countable-choice schedule, quotient array, or ambient completeness row can replace that ledger.*

Proof. Project the accepted carrier. The only tolerance row is D, and it is attached to the same finite window schedule S that supplies the rectangular observations.

Readback through R uses that ledger before any diagonal schedule or completion-facing row appears. The continuation row C records the same order for each accepted request, while H transports the ledger componentwise.

The remaining rows P, N only source and name the route. Since there is no second tolerance, selected-limit, quotient, choice, or completeness coordinate, ledger exactness is forced by the tuple inventory. □

Theorem 3175.6 (RegularDoubleCauchy NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, rectangular window scheduling, row-column* $\mathsf{RegSeqRat}^{\uparrow}$ *stability,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exactness, diagonal cofinal readback,* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. It is the non-diagonal regularity certificate consumed by CauchyDoubleSequence and Moore–Osgood routes before their diagonal or iterated-limit handoffs, and it exports no host double-sequence convergence theorem, quotient-array classifier, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Combine the carrier theorem with row-column stability. These statements expose the rectangular windows, regular readback faces, tolerance ledger, transport, replay, provenance, and local name.

The diagonal readback theorem supplies the only passage from the rectangular surface to a completion-facing diagonal route. Ledger exactness supplies the only precision evidence available on that passage.

Thus the NameCert surface is the finite union of the listed rows and obligations. Downstream CauchyDoubleSequence and Moore–Osgood consumers may use this surface as a regularity certificate, but no excluded host or choice principle has a row from which it could be exported. □

Theorem 3175.7 (RegularDoubleCauchy obligation scope)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier, the five obligation rows of carrier admission, row-column stability, diagonal readback, ledger exactness, and local NameCert surface are scoped exactly by the displayed* BHist *packet together with rectangular* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the diagonal schedule, the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *row.*

No quotient array, selected limit, countable-choice schedule, host double-limit theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle may be used as an alternate scope for any of these obligations.*

Proof. Begin with Definition 4155.1. It fixes the only visible carrier rows as the BHist packet $S, R, D, T, W, H, C, P, N$, with rectangular StreamName windows, RegSeqRat row-column readback, the dyadic tolerance ledger, the diagonal schedule, the RealCauchyCompletion and Real seal handoff, transport, replay, provenance, and local naming.

Apply Definition 4155.2 and Definition 4155.5. These two results bind row-column comparison and precision evidence to the same displayed rectangular windows, readback faces, tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Next apply Definition 4155.3. It forces every diagonal or completion-facing read to pass through the rectangular route before the $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff can be consumed.

Finally, Definition 4155.6 identifies the local NameCert surface with exactly these rows. Since the carrier and surface theorems expose no quotient array, selected limit, countable-choice schedule, host double limit, or ambient Real-completeness row, none of those objects can provide an alternate obligation scope. □

Theorem 3175.8 (RegularDoubleCauchy obligation-closure package)**.** *The carrier of Definition 4155.1, together with row-column stability, diagonal readback, rectangular modulus extraction, ledger exactness, the local NameCert surface, and the obligation scope theorem, packages the five* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *obligation rows into one completion-facing surface. Every accepted consumer must pass through rectangular* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the diagonal schedule, the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *before reading a diagonal or completion endpoint.*

Proof. Begin with Definition 4155.1. It exposes the finite rectangular packet $S, R, D, T, W, H, C, P, N$ and excludes host double limits, quotient arrays, selected limits, countable-choice schedules, and ambient Real completeness.

Use Definition 4155.2, Definition 4155.4, and Definition 4155.5. These theorems bind rectangular StreamName windows, row-column RegSeqRat readback, and the DyadicRatCore tolerance ledger into the same finite modulus surface.

Next apply Definition 4155.3. It places the diagonal schedule and the RealCauchyCompletion or Real seal strictly downstream of the rectangular surface, so a completion-facing read cannot start from the terminal handoff.

Finally combine Definition 4155.6 with Definition 4155.7. The local NameCert rows and scope theorem identify exactly the structural transport, replay, provenance, and naming data that travel with the five obligation rows, leaving no alternate obligation coordinate outside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularDoubleCauchy}^{\uparrow}$ read factors through rectangular StreamName windows, RegSeqRat row-column readback, a DyadicRatCore tolerance ledger, a diagonal cofinal schedule, RealCauchyCompletion and Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchydoublesequence, mooreosgood, streamname, regseqrat, dyadicratcore, realcauchycompletion, real: $\mathsf{RegularDoubleCauchy}^{\uparrow}$ is not bijective to any listed sibling because it keeps rectangular row-column regularity visible before the diagonal or iterated-limit consumer is allowed to read a completion-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularDoubleCauchy}^{\uparrow}$ packet is generated from finite rectangular StreamName windows, RegSeqRat row-column readback, a DyadicRatCore tolerance ledger, a diagonal cofinal schedule, a RealCauchyCompletion

and Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 4155.1, Definition 4155.2, Definition 4155.3, Definition 4155.5, and Definition 4155.6 seed the rectangular two-parameter regularity surface for downstream diagonal and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host double-sequence convergence, quotient arrays, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, row-column stability, diagonal readback, ledger exactness, and local NameCert surface proofs for BEDC.Derived.RegularDoubleCauchyUp.

The $\mathsf{RegularDoubleCauchy}^{\uparrow}$ route reads paired Cauchy windows through the finite stream surface of chapter 126 and the regular rational readback of chapter 301 before any Real-facing handoff. This sibling dependency keeps the double-window packet tied to displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows, with the downstream $\mathsf{Real}^{\uparrow}$ seal supplied only through chapter 38. The $\mathsf{RegularDoubleCauchy}^{\uparrow}$ route consumes the diagonal handoff of chapter 460 only after the rectangular $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ row-column readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ seal of chapter 360 and chapter 38 are displayed. The $\mathsf{RegularDoubleCauchy}^{\uparrow}$ route reads $\mathsf{CauchyAcceleration}^{\uparrow}$ and $\mathsf{CauchySequenceSpace}^{\uparrow}$ before a diagonal or rectangular Real-facing handoff is exported. The acceleration row of chapter 4222 supplies the finite speedup ledger, while the sequence-space surface of chapter 406 keeps the downstream Cauchy-name consumer tied to the same $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ reads interval Newton narrowing through chapter 1570, dyadic interval endpoints through chapter 306, bisection fallback through chapter 1489, and Real-facing readback through chapter 301.

Chapter 3176

A Concrete Naming Certificate for DyadicIntervalNewtonBisection$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicIntervalNewtonBisection$^{\uparrow}$ packet records the finite branch where interval Newton narrowing is interleaved with dyadic bisection before a regular Cauchy or Real-facing consumer reads the result. It refines the dyadic endpoint surface of chapter 306, the Newton enclosure certificate of chapter 1570, the bisection step of chapter 1489, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives root-isolation and completion routes a finite certificate without importing quotient reals, selected roots, trichotomy, countable choice, or ambient completeness.

Definition 3176.1 (Dyadic interval Newton bisection carrier)**.** *A* DyadicIntervalNewtonBisection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (D, I, H, N, B, R, E, T, C, P, L)$$

with the following displayed rows:

1. *D is the* DyadicInterval$^{\uparrow}$ *endpoint row for the parent cell and the inspected child cells.*
2. *I is the* IntervalNewton$^{\uparrow}$ *correction row, including the displayed enclosure and containment ledger.*
3. *H is the* IntervalHalving$^{\uparrow}$ *bisection row used when the Newton enclosure does not yet supply the retained child cell.*
4. *N records the finite narrowing decision: accept the Newton child, accept the bisection child, or replay both rows before the next finite inspection.*
5. *B is the dyadic diameter and containment ledger attached to the retained child interval.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced only after D, I, H, N, B have been displayed.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row exposed to completion consumers.*
8. *T records componentwise* hsame *transport for the dyadic interval, Newton row, halving row, narrowing decision, diameter ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same rows.*
10. *P is the* Pkg *provenance over* DyadicIntervalNewtonBisection$^{\uparrow}$, DyadicInterval$^{\uparrow}$, IntervalNewton$^{\uparrow}$, IntervalHalving$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. L is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, I, H, N, B, R, E, T, C, P, L$. It has no coordinate for a selected zero, quotient real equality, host trichotomy, completed branch choice, or ambient completeness theorem.

Theorem 3176.2 (Dyadic interval Newton bisection NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *carrier $Z = (D, I, H, N, B, R, E, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBise}}$ *surface consists exactly of carrier admission, dyadic endpoint exposure, interval Newton correction exposure, interval-halving fallback exposure, retained-child narrowing, diameter containment, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming for the carrier of Definition 4156.1.*

Proof. Carrier admission is projection from Definition 4156.1. The dyadic row D, Newton row I, halving row H, narrowing row N, and diameter ledger B are the only rows that expose interval refinement data.

The readback rows are downstream of that finite refinement. Row R can read a regular rational name only after the retained dyadic child and its containment ledger are displayed, and row E seals only that readback route.

Transport, replay, provenance, and local naming are exactly T, C, P, L. Since the carrier has no selected-root, quotient-real, trichotomy, completed-branch, or completeness coordinate, those objects cannot become local NameCert obligations. □

Theorem 3176.3 (Dyadic interval Newton bisection regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, root-isolation, or completion-facing consumer of an accepted* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{IntervalNewton}^{\uparrow} \longmapsto \mathsf{IntervalHalving}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff supplies retained dyadic child data, Newton or bisection narrowing, a diameter ledger, regular rational readback, and a terminal Real seal. It supplies no quotient real, selected root, trichotomy proof, countable-choice branch, or ambient completeness principle.

Proof. Project the accepted carrier $Z = (D, I, H, N, B, R, E, T, C, P, L)$. The rows D, I, H, N, B expose the entire finite narrowing branch before any regular Cauchy consumer can inspect the packet.

Read R through the retained child and diameter ledger. The regular rational handoff is therefore attached to the displayed dyadic interval, Newton enclosure, and bisection fallback rows, not to an external root selector.

Finally read the terminal seal E through T, C, P, L. These structural rows preserve the same finite route under transport, replay, provenance, and local naming. No stronger principle is present among the carrier coordinates, so every completion-facing read remains the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ packet is generated from a DyadicInterval endpoint row, an IntervalNewton correction row, an IntervalHalving fallback row, a retained-child narrowing decision, a dyadic diameter ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4156.1, Definition 4156.2, and Definition 4156.3 seed the finite interval Newton and dyadic bisection route over DyadicIntervalUp, IntervalNewtonUp, IntervalHalvingUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicIntervalNewtonBisection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient reals, selected roots, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, Newton correction exposure, halving fallback exposure, retained-child narrowing, diameter containment, regular readback, Real-seal nonescape, transport, replay, provenance, and local NameCert field faithfulness for BEDC.Derived.DyadicIntervalNewtonBisectionUp.

The KadecKleeProperty$^{\uparrow}$ route reads weak convergence through chapter 3597, norm convergence through chapter 89, and its Banach/Hilbert source through chapter 90 and chapter 91 before the Radon–Riesz handoff is exposed.

Chapter 3177

A Concrete Naming Certificate for $\mathsf{KadecKleeProperty}^{\uparrow}$

Following Kadec and Klee on the Radon–Riesz property, the $\mathsf{KadecKleeProperty}^{\uparrow}$ packet records the finite BEDC surface where weak convergence plus convergence of norms is accepted as a strong convergence handoff. It sits between the normed and complete linear analysis packets of chapter 89, chapter 90, and chapter 91, reads the weak-convergence boundary of chapter 3597, and keeps every scalar comparison at the real-name seal of chapter 38. The carrier consumes the $\mathsf{Norm}^{\uparrow}$ norm-readback row from chapter 89 and, when present, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ or $\mathsf{Hilbert}^{\uparrow}$ source row before exposing weak-plus-norm convergence as a strong-convergence row. The chapter names only the displayed Radon–Riesz handoff route and does not import Hahn–Banach separation, choice-selected weak limits, quotient vector equality, or ambient completeness.

Definition 3177.1 (Kadec–Klee property carrier)**.** *A* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, H, N, W, Q, S, T, C, P, L)$$

with the following displayed rows:

1. *B is the ambient* $\mathsf{Banach}^{\uparrow}$ *row, exposing the vector-name surface used by the sequence and limit entries;*
2. *H is the optional* $\mathsf{Hilbert}^{\uparrow}$ *refinement row, used only when the consumer already works over an inner-product source;*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row recording the norm readback and its* $\mathsf{Real}^{\uparrow}$ *comparison seal;*
4. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *row that displays weak convergence of the named sequence to the named candidate limit;*
5. *Q is the norm-convergence row, whose scalar values are read through* $\mathsf{Real}^{\uparrow}$ *names rather than through host real equality;*
6. *S is the strong-convergence handoff row asserting that the same displayed sequence and candidate limit may be consumed as norm convergence after W and Q have both been read;*
7. *T records componentwise* hsame *transport for B, H, N, W, Q, S and for the local certificate rows;*
8. *C records the displayed* Cont *replay from weak observations and norm-value observations to the strong-convergence handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{KadecKleeProperty}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *L is the local* $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Hahn–Banach separation, a selected weakly convergent subnet, quotient equality of vectors, a detached norm limit, or a completeness principle beyond the displayed source rows.

Theorem 3177.2 (Kadec–Klee NameCert obligations)**.** *For an accepted* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier*

$$K = (B, H, N, W, Q, S, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach row, optional Hilbert refinement, norm row, weak-topology row, norm-convergence row, strong-convergence handoff, componentwise transport, continuation replay, provenance, and local naming; classifier stability under the same displayed rows; exactness of the* $\mathsf{Real}^{\uparrow}$ *norm-value comparisons; visibility of the weak convergence row before any strong handoff is read; and refusal of Hahn–Banach separation, choice-selected weak limits, quotient vector equality, detached scalar limits, and ambient completeness.*

Proof. Project the carrier of Definition 4157.1. The projection exposes exactly the analysis rows B, H, N, W, Q, S and the structural rows T, C, P, L.

The weak and norm routes are separate until both have been displayed. The weak route reads W over the same named vector sequence and candidate limit, while the norm route reads Q through the $\mathsf{Norm}^{\uparrow}$ row N and the $\mathsf{Real}^{\uparrow}$ scalar seal.

Now replay the handoff through C and transport it through T. The strong-convergence row S is available only downstream from W and Q, and the provenance and naming rows attach no additional selection, separation, quotient equality, detached limit, or completeness data. □

Theorem 3177.3 (Kadec–Klee weak-strong handoff)**.** *Let $K = (B, H, N, W, Q, S, T, C, P, L)$ be an accepted* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier. Any consumer that reads the strong-convergence handoff S must first read the weak-convergence row W, the norm row N, the norm-convergence row Q, the shared* $\mathsf{Real}^{\uparrow}$ *scalar seal, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row L. Thus the Radon–Riesz handoff is a finite BEDC route over* $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *not a consequence of Hahn–Banach separation, choice-selected weak limits, quotient vector equality, or ambient completeness.*

Proof. Start with a request for S. Carrier admission gives the only available predecessors: weak convergence through W, norm readback through N, scalar convergence through Q, and the structural rows T, C, P, L.

Use the NameCert obligation surface of Definition 4157.2. It forces the weak observation and the norm-value comparison to share the same displayed sequence, candidate limit, transport row, and real-name scalar boundary before strong convergence can be consumed.

Because no other coordinate exists in the carrier, the consumer cannot replace the handoff by a separation theorem, a selected weak limit, quotient vector equality, a hidden scalar limit, or an ambient completeness argument. The handoff is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KadecKleeProperty}^{\uparrow}$ packet is generated from Banach, optional Hilbert, Norm, WeakTopology, Real scalar, weak-convergence, norm-convergence, strong-handoff, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$, seed)

Scope closed: Definition 4157.1, Definition 4157.2, and Definition 4157.3 seed the Radon–Riesz route from weak convergence and norm convergence to strong convergence over displayed Banach, Hilbert, Norm, WeakTopology, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KadecKleeProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice-selected weak limits, quotient vector equality, detached scalar limits, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, weak-row visibility, norm-scalar exactness, strong-handoff routing, transport replay, provenance, and consumer refusal for BEDC.Derived.KadecKleePropertyUp.

The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ hub routes its uniform tail-budget packet through chapter 126, chapter 301, chapter 291, chapter 38, and chapter 292 before the child files expose the local carrier and dependency routes.

Chapter 3178

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy real name is useful to Real and Completion consumers only when its uniform tail budget is displayed before any terminal real seal is consumed. The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ packet records that budget as finite BEDC data over the stream-window surface of chapter 126, the dyadic observation ledger of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. It names the explicit uniform-modulus surface needed by regular Cauchy tails; it is not a quotient of sequences, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3178.1 (Regular Cauchy uniform modulus carrier)**.** *A* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, M, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, carrying the finite windows opened on the source name before any regular readback is inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, containing the dyadic tolerance requests and arithmetic comparisons used by the tail budget.*
3. *M is the displayed regular modulus row; for each requested dyadic tolerance it records the finite index bound after which the regular Cauchy readbacks are stable inside that tolerance.*
4. *T is the finite tail ledger joining stream windows, dyadic tolerances, regular readbacks, and the modulus row.*
5. *H records componentwise* hsame *transport for S, D, M, T and the structural certificate rows.*
6. *C records displayed* Cont *replay from a tail-stability request through the same stream, dyadic, modulus, and tail-ledger rows.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, D, M, T, H, C, P, N. It has no coordinate for quotient stream equality, selected limit witnesses, host real equality, countable choice, or ambient completeness.

Theorem 3178.2 (Regular Cauchy uniform modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier* $U = (S, D, M, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window exposure, dyadic tolerance accounting, displayed regular modulus admission, finite tail-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that uses a regular tail budget must read these rows before it can consume the terminal real-name seal.*

Proof. Project the carrier of Definition 4158.1. The projection displays the stream row S, dyadic observation row D, regular modulus row M, finite tail ledger T, transport row H, continuation row C, provenance row P, and local name N.

The numerical obligations are exhausted by S, D, M, T. Stream windows are opened in S, tolerance comparisons are read in D, the index bound is displayed in M, and the ledger T records the finite route joining those rows to regular rational readbacks.

The structural obligations are exhausted by H, C, P, N. They transport, replay, source, and name the same displayed route, while the carrier lists no quotient stream, selected limit, host equality, choice schedule, or ambient completeness coordinate. Hence the listed rows are exactly the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ obligations. □

Theorem 3178.3 (Regular Cauchy uniform tail stability)**.** *Let* $U = (S, D, M, T, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Every tail-stability request for the associated regular Cauchy readback factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyUniformModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Na}$$

After the modulus row M has supplied the displayed index bound, the tail ledger T is the only admissible source of stability data for the requested dyadic tolerance. No consumer may replace that route by a host convergence theorem, a selected Cauchy limit, quotient equality of streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 4158.2. A tail-stability request first opens the stream row S and the dyadic observation row D for the requested tolerance.

The modulus row M then supplies the finite index bound. The ledger T binds that bound to the same stream windows and dyadic comparisons, so the regular rational readback can only be consumed through the displayed route.

Finally H, C, P, N transport, replay, source, and name the packet. Since no carrier coordinate stores a selected limit, quotient stream class, host real equality, choice schedule, or ambient completeness principle, the displayed tail ledger is the complete stability surface. □

Theorem 3178.4 (Regular Cauchy uniform modulus tail-budget exactness)**.** *Let* $U = (S, D, M, T, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Before any tail-stability or completion-facing consumer reads the modulus row, the packet exposes the same finite dyadic budget in D, the same* $\mathsf{StreamName}^{\uparrow}$ *window in S, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary carried by T, and the same* $\mathsf{Real}^{\uparrow}$ *seal route recorded by H, C, P, N. Thus the tail budget is exact as a displayed BEDC row: no later consumer may widen it into a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 4158.1. The rows S, D, M, T expose the stream window, dyadic tolerance ledger, displayed modulus, and finite tail ledger before the structural rows are consulted.

Use Definition 4158.2. Its obligation list requires every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to read the stream-window exposure, dyadic tolerance accounting, modulus-row admission, and tail-ledger exactness before the terminal real-name seal is consumed.

Apply Definition 4158.3. The tail-stability route factors through the same S, D, M, T rows, so a completion-facing read cannot replace the displayed budget by a different modulus surface.

Finally inspect the carrier inventory. There is no quotient stream row, selected limit row, host real equality row, countable-choice row, or ambient completeness row, and H, C, P, N only transport, replay, source, and name the already displayed packet. Hence the exposed tail budget is exact for every downstream read. □

Theorem 3178.5 (Regular Cauchy uniform modulus tail composition)**.** *Let $U = (S, D, M, T, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Suppose its* $\mathsf{StreamName}^{\uparrow}$ *windows replay through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows in D and through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary carried by T. Then every admitted unary precision row composes to the terminal* $\mathsf{Real}^{\uparrow}$ *seal without selecting a limit: the modulus tail bound in M supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *closeness witness consumed by the seal route recorded in H, C, P, N.*

Proof. Use Definition 4158.2. It requires every regular rational or real consumer to expose stream-window, dyadic tolerance, modulus-row, and tail-ledger data before the terminal seal is read.

Apply Definition 4158.3. The admitted precision row first opens the $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the modulus row M supplies the finite tail bound used by the regular readback.

Now apply Definition 4158.4. The same displayed budget is preserved through S, D, M, T, so the $\mathsf{RegSeqRat}^{\uparrow}$ closeness witness consumed by the terminal $\mathsf{Real}^{\uparrow}$ route is the witness supplied by the accepted packet.

The sibling chapters chapter 126, chapter 291, chapter 301, chapter 38, and chapter 292 provide the named row surfaces. Since the carrier has no selected-limit coordinate, the composition remains a finite handoff rather than a limit-selection principle. □

Theorem 3178.6 (Regular Cauchy uniform modulus dyadic locator handoff)**.** *Let $U = (S, D, M, T, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Every Real- or Completion-facing consumer receives the dyadic tolerance row D before it may consume the regular modulus row M, the finite tail ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the handoff from dyadic locator data to regular tail data is a displayed BEDC boundary; it exports no quotient stream, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 4158.2. Its obligation surface places dyadic tolerance accounting in D before displayed regular modulus admission in M and finite tail-ledger exactness in T.

Apply Definition 4158.3. A tail-stability request opens the $\mathsf{StreamName}^{\uparrow}$ row S and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D before the modulus bound and regular rational readback are available.

Now use Definition 4158.4. The exposed budget remains the same displayed dyadic row throughout the completion-facing route, so the terminal $\mathsf{Real}^{\uparrow}$ seal can only be read after D, M, T have been consumed in that order.

Finally inspect the carrier inventory. The rows H, C, P, N transport, replay, source, and name the same handoff, while the carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, or ambient completeness. □

Definition 3178.7 (Located real comparison aperture carrier)**.** *A located real comparison aperture over an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *packet is a finite* BHist *packet*

$$A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$$

whose rows refine the carrier of Definition 4158.1. The rows S, D, M, T, H, C, P, N *are the stream-window, dyadic-tolerance, uniform-modulus, finite tail-ledger, transport, replay, provenance, and local naming rows already displayed by the uniform-modulus packet. The row* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary exposed by the tail ledger, and* A *is a* $\mathsf{LocatedRealComparison}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing comparison boundary. Any route to* A *must expose the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and uniform modulus before any comparison read is consumed.*

Theorem 3178.8 (Located real comparison aperture NameCert obligations)**.** *For every accepted located real comparison aperture* $A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$*, the local comparison-facing obligation surface consists of carrier admission for the displayed aperture, stream-window exposure through* S*, dyadic tolerance accounting through* D*, uniform-modulus admission through* M*, tail-ledger exactness through* T*, readback boundary exposure through* R*, comparison-boundary exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the aperture carrier of Definition 4158.7. Its coordinates display exactly the rows named in the statement, with R and A added as the readback and comparison-facing boundary rows over the existing uniform-modulus packet.

Apply Definition 4158.3. Every tail-stability request first opens the stream window and dyadic tolerance rows, then reads the displayed modulus and the finite tail ledger.

Apply Definition 4158.4. The budget exposed to the comparison boundary is the same displayed S, D, M, T route, so R and A cannot introduce quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3178.9 (Located real comparison aperture gap route)**.** *Every positive-gap, overlap, or comparison-read consumer of an accepted aperture* $A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$ *factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyUniformModulus}^{\uparrow},\ \mathsf{LocatedRealComparison}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the comparison boundary A *is visible only after the stream windows* S*, dyadic tolerances* D*, uniform modulus* M*, tail ledger* T*, and readback boundary* R *have been displayed.*

Proof. Use Definition 4158.6. It places the dyadic locator row before the uniform modulus, finite tail ledger, regular readback route, and terminal real seal.

Now apply Definition 4158.8. The comparison boundary row A is a coordinate of the aperture, but its obligation surface requires the preceding S, D, M, T, R rows to be exposed before A is read.

Finally use the sibling comparison surface of chapter 1730. The gap or overlap consumer receives a finite located-comparison route over the already displayed regular-Cauchy readback, not a host comparison or an ambient completeness principle. □

Theorem 3178.10 (Regular Cauchy uniform modulus lattice crosslink)**.** *Every accepted* RegularCauchyUniformMod *carrier presents the same*

$$S, D, M, T, H, C, P, N$$

route needed by downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *completion-facing reads. The crosslink keeps stream windows, dyadic budgets, regular readback, uniform tail ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming visible as one L10 lattice route; it exports no quotient stream equality, selected limit, host real equality, countable-choice schedule, or ambient completeness row.*

Proof. Project the carrier of Definition 4158.1. The rows S, D, M, T expose the stream window, dyadic tolerance ledger, displayed modulus, and finite tail ledger, while H, C, P, N transport, replay, source, and name that same packet.

Apply Definition 4158.2. Its obligation surface requires every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to pass through the displayed stream-window, dyadic tolerance, modulus, and tail-ledger rows before the terminal seal is consumed.

Apply Definition 4158.4. The tail budget exposed to a completion-facing read is the same displayed BEDC row throughout the route, so it cannot be replaced by a quotient stream, selected Cauchy limit, host real equality, choice schedule, or ambient completeness theorem.

The sibling completion-facing window route is the one named in chapter 406. Therefore the accepted carrier supplies exactly the S, D, M, T, H, C, P, N lattice crosslink used by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchySequenceSpace}^{\uparrow}$ consumers.

The displayed modulus row is read through the sibling certificate of chapter 292 before tail-budget replay reaches the $\mathsf{RegSeqRat}^{\uparrow}$ readback boundary and the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 3178.11 (Regular Cauchy uniform modulus window refinement)**.** *Let*

$$U_0 = (S, D, M_0, T_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad U_1 = (S, D, M_1, T_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carriers sharing the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and displayed* Cont *replay route. If the unary precision displayed by* U_1 *refines the unary precision displayed by* U_0*, then the later tail-budget route factors through the earlier uniform-modulus ledger and cannot widen the exported* $\mathsf{Real}^{\uparrow}$ *handoff.*

Proof. Project both carriers using Definition 4158.1. The shared rows S, D and shared replay route in C_0, C_1 place both packets on the same stream-window and dyadic-tolerance surface before either tail ledger is read.

Apply the NameCert obligations, Definition 4158.2, and tail stability, Definition 4158.3. The finer unary window may only ask for a later displayed tail budget along the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route.

Use tail-budget exactness, Definition 4158.4, and tail composition, Definition 4158.5. Exactness keeps each budget as a displayed BEDC row, while composition routes the later tail request through the earlier uniform-modulus ledger instead of replacing it.

Finally apply the dyadic-locator handoff, Definition 4158.6, and the lattice crosslink, Definition 4158.10. Since the carrier inventory has no selected limit, quotient equality, trichotomy, or ambient completeness row, the refined window cannot widen the terminal $\mathsf{Real}^{\uparrow}$ handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ consumer displays a $\mathsf{StreamName}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, a regular modulus row, a finite tail ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any $\mathsf{RegSeqRat}^{\uparrow}$ readback or terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Independence witness: streamname, dyadicratcore, regseqrat, real: RegularCauchyUniformModulus$^{\uparrow}$ is not bijective to any listed sibling because it packages the displayed uniform tail budget and finite tail ledger as one NameCert surface between stream observations, dyadic tolerances, regular rational readback, and the terminal real-name seal.

Closure status for NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4158.1, Definition 4158.2, and Definition 4158.3 seed the finite uniform-modulus surface over StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected limits, host real equality, countable choice, bridge checking, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance accounting, modulus-row admission, finite tail-ledger exactness, transport replay, and non-escape over the same displayed packet.

Constructive story: A RegularCauchyUniformModulus$^{\uparrow}$ packet is generated from finite stream windows, dyadic tolerance observations, a displayed regular modulus row, a tail ledger, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Theorem 3178.12 (RegularCauchyUniformModulus Cauchy criterion route)**.** *Every accepted* RegularCauchyUniformModulus$^{\uparrow}$ *read depends on the* CauchyModulus$^{\uparrow}$ *schedule of chapter 292 and the* UniformCauchyCriterion$^{\uparrow}$ *shared-threshold row of chapter 435 before any* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer inspects the displayed modulus. The schedule supplies the finite tolerance-to-index request, the uniform criterion row synchronizes that request across the relevant tail windows, and only then may the regular rational readback and terminal Real seal be consumed. The route exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Read the CauchyModulus$^{\uparrow}$ dependency first. It provides the finite schedule that turns a dyadic tolerance request into the index boundary used by the modulus row.

Then read the UniformCauchyCriterion$^{\uparrow}$ dependency. Its shared-threshold row records that the same index boundary controls all tail-window comparisons needed by the regular readback.

Only after those two dependency rows are visible can the existing RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers inspect the displayed RegularCauchyUniformModulus$^{\uparrow}$ packet. Since the route contains no selected limit, quotient stream, host equality, choice schedule, or ambient completeness row, the dependency remains a finite sibling route. □

Theorem 3178.13 (Regular Cauchy uniform modulus RegSeqRat–Real forward lattice)**.** *Every* RegularCauchyUniformModulus$^{\uparrow}$ *consumer that exports a uniform tail budget must first read the sibling rows* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *through the same finite packet. The uniform bound is available only after the stream windows, regular rational readback, dyadic tolerance ledger, and terminal Real seal route have been displayed with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The forward lattice exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the RegularCauchyUniformModulus$^{\uparrow}$ carrier. The stream row supplies the finite windows, the dyadic row supplies the tolerance ledger, and the modulus row supplies the uniform tail budget.

The regular rational and Real-facing reads are downstream of those rows. A consumer must keep the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal on the same displayed route before exporting the tail budget to a completion-facing use.

Finally read the structural rows. The hsame, Cont, Pkg, and NameCert data preserve and name the same packet, and no row supplies quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 3178.14 (RegularCauchyUniformModulus dependency route)**.** *Every accepted* RegularCauchyUniformMo *read consumes the substantive dependency route from the* $\mathsf{Cauchy}^{\uparrow}$ *tail-window and modulus rows of chapter 3413, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal data of chapter 38, and the tolerance-only ledger rows of chapter 369. The packet therefore belongs to the L10 Cauchy and uniform-modulus lattice, not to an isolated orphan carrier surface.*

Proof. Read the Cauchy side first. The tail-window and modulus rows fix the finite observation and tolerance route before any regular rational or Real-facing consumer can inspect the packet.

The regular-rational and Real rows then provide readback and sealing. The uniform-modulus side contributes only the tolerance ledger needed by the same route, so no independent host modulus, selected limit, quotient real equality, or ambient completeness object enters the dependency. □

Theorem 3178.15 (Regular Cauchy uniform modulus StreamName and RegSeqRat route)**.** *Every* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *consumer that exports a uniform-modulus claim must first read the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The route therefore presents the uniform-modulus claim as a finite sibling dependency over stream observations and regular rational readback, not as a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the carrier and obligation rows from the hub spine. They expose the stream windows, dyadic tolerances, regular readback, modulus schedule, transport, replay, provenance, and local naming before any uniform-modulus claim is available.

The existing sibling routes keep the $\mathsf{CauchyModulus}^{\uparrow}$ schedule and $\mathsf{RegSeqRat}^{\uparrow}$ handoff visible. Combining those routes with the $\mathsf{StreamName}^{\uparrow}$ row leaves no coordinate for quotient stream equality, selected limits, host equality, countable choice, or ambient real completeness. □

Theorem 3178.16 (Regular Cauchy uniform modulus L10 sibling route)**.** *Every* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *consumer that reads a uniform modulus factors first through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-schedule rows of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal rows of chapter 38. Only after those L10 sibling rows are displayed may the consumer inspect the uniform modulus row. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 4158.1. Its rows expose the stream window S, dyadic tolerance row D, displayed modulus row M, finite tail ledger T, componentwise transport H, replay C, provenance P, and local name N.

Use the NameCert obligations of Definition 4158.2. They require the stream-window exposure, dyadic tolerance accounting, modulus-row admission, and finite tail-ledger exactness before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives the terminal seal.

Now read the sibling routes already present in the chapter. The forward lattice of Definition 4158.13 keeps $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same finite path, while Definition 4158.15 records the stream and regular-schedule handoff.

Since the carrier has no coordinate for quotient streams, selected limits, choice schedules, host equality, or ambient completeness, the displayed L10 sibling rows are the only admissible route to the uniform modulus. □

The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ tail-budget route consumes the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the $\mathsf{DyadicTailBall}^{\uparrow}$ late-tail containment boundary of chapter 1633, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 in that order. A consumer may inspect the displayed modulus only after the stream windows and regular readback have exposed the dyadic late-tail ledger, and it may pass to $\mathsf{Real}^{\uparrow}$ only through the same seal-facing route. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 3179

A Concrete Naming Certificate for $\mathsf{SpectralRadiusFormula}^{\uparrow}$

Following Gelfand's spectral-radius formula for Banach algebras, the $\mathsf{SpectralRadiusFormula}^{\uparrow}$ packet records the finite norm-power route by which displayed operator-algebra data can be read as a spectral-radius comparison without importing a host spectrum or an ambient limit theorem. It sits after the spectral-radius boundary of chapter 2352, the bounded-operator carrier of chapter 1644, the norm carrier of chapter 89, the Banach source of chapter 90, and the real-name seal of chapter 38.

Definition 3179.1 (Spectral-radius formula carrier)**.** *A* $\mathsf{SpectralRadiusFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, T, N, P, W, D, R, E, H, C, Q, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{Banach}^{\uparrow}$ *or Banach-algebra row carrying the complete normed algebra surface from which powers are read.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for the operator or algebra element whose powers are inspected.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *norm-power row, recording each displayed norm endpoint for T^n in the requested window.*
4. *P is the power-iteration ledger, exposing only the finite multiplication and composition rows needed to form the displayed powers.*
5. *W is the finite window of requested exponents.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used when the comparison is requested at finite rational precision.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed norm-power window and dyadic tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *spectral-radius seal reached only after W, D, R have been replayed.*
9. *H records componentwise* hsame *transport for A, T, N, P, W, D, R, E and the structural rows.*
10. *C records displayed* Cont *replay of the route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto E \longmapsto \mathsf{SpectralRadiusFormula}^{\uparrow}.$$

11. *Q is the* Pkg *provenance over* $\mathsf{SpectralRadiusFormula}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *L is the local* $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, T, N, P, W, D, R, E, H, C, Q, L$ and has no coordinate for a host spectrum equality, quotient completion, selected limits, Classical choice, an ambient spectral theorem, or a completed all-powers limit.

Lean4 ✓ `BEDC.Derived.SpectralRadiusFormulaUp.SpectralRadiusFormulaTasteGate single carrier alignment`

Theorem 3179.2 (Spectral-radius formula NameCert obligations)**.** *For every accepted* $\mathsf{SpectralRadiusFormula}^{\uparrow}$ *carrier*

$$F = (A, T, N, P, W, D, R, E, H, C, Q, L),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ *surface consists exactly of carrier admission, norm-power window stability, spectral-radius comparison against the displayed* $\mathsf{SpectralRadius}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transport, continuation replay, provenance, and local naming. Any downstream operator-algebra consumer reading the formula must pass through the finite window W, the norm-power row N, the power ledger P, the dyadic tolerance D, and the readback rows R, E before a spectral-radius comparison is public.*

The certificate exports no host spectrum equality, no quotient completion, no selected limit over all powers, no Classical-choice selector, and no ambient spectral theorem.

Proof. Project the carrier of Definition 4159.1. The rows $A, T, N, P, W, D, R, E, H, C, Q, L$ are the whole accepted surface, so every formula read begins with the displayed Banach source, bounded-operator row, norm row, and finite power ledger.

Norm-power window stability is finite. The exponent window W names the inspected powers, P records how those powers are assembled, and N records only their displayed norm endpoints. Componentwise transport through H preserves those rows before replay through C exposes any real-facing readback.

The spectral-radius comparison is then routed through the sibling $\mathsf{SpectralRadius}^{\uparrow}$ boundary. The comparison consumes the same finite norm-power data and the same dyadic tolerance D; it does not replace the packet by a completed spectrum or an all-powers limit.

Finally R and E perform the regular-rational and real-name handoff. Since no carrier coordinate stores host spectrum equality, quotient completion, selected limits, Classical choice, or an ambient spectral theorem, the local $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ names exactly the finite Gelfand formula route stated above. □

Falsifiable prediction: Every accepted $\mathsf{SpectralRadiusFormula}^{\uparrow}$ consumer exposes the Banach source row, bounded-operator row, norm-power row, power-iteration ledger, finite exponent window, dyadic tolerance, regular-rational readback, RealUp seal, componentwise transport, replay, provenance, and local naming before reading a spectral-radius formula comparison.

Independence witness: spectralradius, boundedlinearoperator, norm, banach, real: SpectralRadiusUp supplies the radius boundary without the finite norm-power formula ledger; BoundedLinearOperatorUp supplies action and norm-bound data without the Gelfand formula readback; NormUp supplies norm endpoints without power-window replay; BanachUp supplies the complete

normed source without the formula carrier; RealUp seals readbacks without naming the operator-power route.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralRadiusFormula}^{\uparrow}$ packet is generated from a Banach source row, a bounded-operator row, a norm-power row, a power-iteration ledger, a finite exponent window, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp spectral-radius seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4159.1, Definition 4159.2, the falsifiable prediction, and the independence witness seed the finite Gelfand spectral-radius formula route over SpectralRadiusUp, BoundedLinearOperatorUp, NormUp, BanachUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host spectrum equality, quotient completion, selected limits, Classical choice, an ambient spectral theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, norm-power window stability, spectral-radius comparison, regular-rational and Real seal handoff, classifier transport, and consumer non-escape for BEDC.Derived.SpectralRadiusFormulaUp.

Chapter 3180

A Concrete Naming Certificate for $\mathsf{NestedIntervalsTheorem}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedIntervalsTheorem}^{\uparrow}$ packet records the theorem-shaped finite route by which a nested sequence of closed bounded intervals supplies a real-name witness when each displayed stage carries an explicit shrinking and containment ledger. It reads the base closed-interval source of chapter 1591, the nested-prefix surface of chapter 320, the dyadic endpoint ledger of chapter 306, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the Bishop-style nested-interval theorem as a finite NameCert obligation surface; it does not assert open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, countable choice, trichotomy, or selected host endpoints.

Definition 3180.1 (NestedIntervalsTheorem carrier)**.** *A* $\mathsf{NestedIntervalsTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, J, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row carrying lower and upper endpoint data, endpoint order, containment ledgers, finite windows, regular readback, and a real-name seal boundary.*
2. *J is the* $\mathsf{NestedInterval}^{\uparrow}$ *prefix row recording the displayed containment chain over the same interval source.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *shrinking ledger attached to the displayed endpoints and nested-containment rows.*
4. *W is the finite observation-window row that records which source stages and precisions are opened.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from B, J, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the closed-bounded source, nested prefix, dyadic ledger, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the source interval, nested prefix, dyadic ledger, windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the theorem request through $B, J, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* NestedIntervalsTheorem$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, NestedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$ *row.*

The carrier is theorem-shaped because it names the obligation route from a displayed nested closed-interval prefix to a real-name witness seal. It has no coordinate for an arbitrary open cover, selected host endpoint, quotient real equality, trichotomy decision, countable-choice selector, ambient Real$^{\uparrow}$ *completeness theorem, Lean verification row, or bridge row.*

Theorem 3180.2 (NestedIntervalsTheorem NameCert obligations)**.** *For every accepted* NestedIntervalsTheorem$^{\uparrow}$ *carrier $T = (B, J, D, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed-bounded interval source B, the nested-prefix row J, the dyadic shrinking ledger D, the finite observation windows W, regular rational readback R, terminal* Real$^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance P, and local naming N. Every Real-facing witness read must pass through B, J, D, W, R before E is visible, and no obligation may be discharged by open-cover compactness, ambient real completeness, quotient equality, countable choice, trichotomy, or selected host endpoints.*

Lean4 ✓ `BEDC.Derived.NestedIntervalsTheoremUp.NestedIntervalsTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4160.1. The displayed analytic rows are the closed-bounded source B, nested-prefix row J, dyadic shrinking ledger D, finite window row W, regular readback R, and terminal seal E.

The witness route is ordered by those rows. The closed-bounded source supplies endpoint order and containment, the nested-prefix row supplies inclusion, the dyadic ledger supplies shrinking data, and the finite observation windows determine the regular rational readback before the Real seal can be consumed.

The structural rows add no hidden theorem. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route. Since the carrier has no coordinate for open-cover compactness, ambient completeness, quotient equality, countable choice, trichotomy, or selected endpoints, the listed rows exhaust the NameCert obligations. □

Closure status for NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$.

Constructive story: A NestedIntervalsTheorem$^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a NestedIntervalUp prefix, a DyadicIntervalUp shrinking ledger, finite observation windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 4160.1 and Definition 4160.2 seed the Bishop-style nested-interval theorem route as finite NameCert data over closed bounded intervals, nested prefixes, dyadic ledgers, regular rational readback, and real-name seals.

Formal verification: FormalStatus(NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient Real$^{\uparrow}$ completeness, quotient real equality, trichotomy, countable choice, selected host endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-prefix containment, dyadic shrinking, finite-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalsTheoremUp.

CauchyRealSpread$^{\uparrow}$ reads its spread packet through chapter 126, chapter 301, chapter 38, chapter 333, and chapter 95.

Chapter 3181

A Concrete Naming Certificate for $\mathsf{CauchyRealSpread}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and Brouwer-style spread presentations, the $\mathsf{CauchyRealSpread}^{\uparrow}$ packet records the finite spread interface through which a countable family of regular Cauchy real-name windows is read without importing choice sequences or quotient reals. It sits after the prefix discipline of chapter 1677, the finite window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a spread-indexed real-name boundary, not a completed branch, not a selected Cauchy representative, not countable choice, not quotient real equality, and not a completeness principle for $\mathsf{Real}^{\uparrow}$.

The spread-indexed real-name boundary consumes chapter 1677 for prefix-tree admission, chapter 126 for finite window scheduling, chapter 301 for regular rational readback, and chapter 38 only as the terminal seal.

Before a Real-completion consumer reads the spread as a terminal $\mathsf{Real}^{\uparrow}$ handoff, the packet consumes the finite diagonal-window route of chapter 4080 as a displayed $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ route rather than as a selected subsequence or host completeness argument.

Definition 3181.1 (CauchyRealSpread carrier)**.** *A* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, E, T, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the spread root row inherited from* $\mathsf{BrouwerSpread}^{\uparrow}$*, recording the displayed root prefix and its admitted prefix-tree boundary.*
2. *W is the finite window row, listing the spread prefixes and request windows whose regular Cauchy real-name data may be inspected.*
3. *E is the extension law row, requiring every displayed extension of an admitted prefix to remain inside the same spread discipline before any real-name read is public.*
4. *T is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite observations over the admitted prefixes.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row obtained from the scheduled windows and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the spread root, finite window, extension law, stream schedule, dyadic ledger, and regular rational readback have been displayed.*

7. *H records componentwise* hsame *transport for the spread root, finite window, extension law, StreamName, RegSeqRat, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from spread-prefix admission through finite window inspection, extension-law checking, stream scheduling, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealSpread}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$ *row.*

The carrier has no coordinate for completed infinite branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3181.2 (CauchyRealSpread NameCert obligations)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier $S = (R, W, E, T, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the spread root, finite window, extension law, StreamName row, RegSeqRat row, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *root compatibility, requiring every public real-name request to start from the displayed* $\mathsf{BrouwerSpread}^{\uparrow}$ *root row R;*
3. *window discipline, requiring the finite request window W to be read before any stream, rational, or real-seal row is consumed;*
4. *extension stability, requiring each admitted prefix extension to pass through the law row E before the scheduled stream row T is public;*
5. *regular readback exactness, requiring every finite real-name observation to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q before the* $\mathsf{Real}^{\uparrow}$ *seal A is read;*
6. *transport and replay, requiring H and C to preserve the same ordered route under* hsame *transport and displayed* Cont *replay;*
7. *ledger non-escape, excluding completed branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, and ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier. The tuple exposes exactly the spread root R, finite window W, extension law E, stream schedule T, regular rational readback Q, Real seal A, and structural rows H, C, P, N.

Root compatibility and window discipline follow from the tuple order. A public real-name request begins with the displayed $\mathsf{BrouwerSpread}^{\uparrow}$ root and then reads only the finite windows stored in W.

Extension stability is recorded by E. The row checks each displayed prefix extension before the scheduled $\mathsf{StreamName}^{\uparrow}$ observations in T can be consumed, so no completed branch or choice-sequence selection is introduced.

Regular readback exactness then routes every tolerance request through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q before the terminal $\mathsf{Real}^{\uparrow}$ seal A is available. The rows H and C transport and replay this same ordered route, while P and N record provenance and local naming.

Since the carrier has no further coordinate, quotient real equality, selected representatives, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$ cannot be read from the packet. □

Theorem 3181.3 (CauchyRealSpread root and window discipline)**.** *For every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

each public spread-indexed real-name request must preserve the $\mathsf{BrouwerSpread}^{\uparrow}$ *root row* R *and the finite window row* W *before it reads the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A*. The route records only root admission and finite window discipline; it does not select completed branches, choice sequences, quotient real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 4161.1 and the obligation list of Definition 4161.2. The rows R and W are displayed before the stream schedule, regular readback, Real seal, and structural rows.

The root row R supplies the admitted $\mathsf{BrouwerSpread}^{\uparrow}$ prefix surface. A public request that omits R has no displayed spread root from which a finite real-name window can be read.

The window row W then lists the finite spread prefixes and request windows available to the consumer. Only after this row is visible may the route pass to the $\mathsf{StreamName}^{\uparrow}$ schedule T, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the terminal $\mathsf{Real}^{\uparrow}$ seal A.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no row stores a completed branch, choice sequence, quotient equality, or completeness theorem, the discipline is exactly the finite root-window route. □

Theorem 3181.4 (CauchyRealSpread sibling route)**.** *For every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

any read exposing a spread of regular Cauchy names must factor through the sibling route consisting of the $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the diagonal-selection boundary of chapter 4080. The route is a finite dependency surface: it adds no selected host subsequence, no quotient stream equality, and no ambient completeness coordinate.*

Proof. Project the carrier rows from Definition 4161.1. The rows W, T display the finite windows and StreamName schedule, while Q is the regular rational readback row and A is the terminal Real seal.

Use the route discipline of Definition 4161.3. A spread-indexed request begins from the root and window rows before it can expose StreamName observations.

The regular readback then passes through Q and reaches A only after the finite schedule is visible. The diagonal-selection sibling of chapter 4080 may consume this surface as a boundary for comparing displayed Cauchy windows, but it does not supply an external subsequence selector.

Transport through H, replay through C, and naming through P, N preserve the same sibling route. Since the carrier has no coordinate for a selected host subsequence, quotient stream equality, or ambient completeness, the exposed spread of regular Cauchy names remains inside the finite L10 route. □

The downstream countable-dense handoff of chapter 1809 reads $\mathsf{CauchyRealSpread}^{\uparrow}$ only through the finite window route of chapter 4080. In precise dependency terms, the spread root row R and finite window row W feed the diagonal finite-window selection before any separability or completion-facing consumer may inspect a terminal $\mathsf{Real}^{\uparrow}$ seal, so the Real-facing read remains downstream of the displayed root/window rows rather than a direct completeness or quotient route.

Theorem 3181.5 (CauchyRealSpread extension readback stability)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier* $S = (R, W, E, T, Q, A, H, C, P, N)$*, every readback along an admitted prefix extension is stable only when the extension law row* E *remains visible between the root-window route* R, W *and the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*. The resulting readback passes through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as finite rows and never becomes a completed branch, a selected Cauchy representative, quotient real equality, or a Real-completeness assertion.*

Proof. Begin with Definition 4161.2. Its extension-stability clause places E before the scheduled stream observations and after the displayed root-window route.

For an admitted prefix extension, read R and W first. The row E checks that the displayed extension remains inside the same spread discipline before any observation in T is public.

After E has been read, the route may consume the $\mathsf{StreamName}^{\uparrow}$ schedule T, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ row Q, and finally the $\mathsf{Real}^{\uparrow}$ seal A. Transport through H and replay through C preserve this order for the same finite packet.

There is no carrier coordinate for an infinite branch, choice-sequence selector, selected representative, quotient equality, or completeness principle. Thus extension readback stability is a finite routing condition, not an external real-analysis closure claim. □

Theorem 3181.6 (CauchyRealSpread regular-seal route)**.** *Every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *route to the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A *must pass through the ordered rows*

$$R, W, E, T, Q, A, H, C, P, N.$$

The route starts at the $\mathsf{BrouwerSpread}^{\uparrow}$ *root, keeps the finite window and extension-law rows visible, reads the* $\mathsf{StreamName}^{\uparrow}$ *schedule, consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and only then exposes the Real seal. It exports no completed branch, no choice sequence, no quotient real equality, no selected representative, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier definition and Definition 4161.3. They force every public real-name request to begin with the $\mathsf{BrouwerSpread}^{\uparrow}$ root row R and finite window row W.

Next apply extension readback stability, Definition 4161.5. The extension law row E remains visible before the schedule T can expose finite observations along admitted prefixes.

The regular readback exactness clause of Definition 4161.2 then routes tolerance and rational data through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ in Q before the terminal $\mathsf{Real}^{\uparrow}$ seal A is public.

Finally the structural rows H, C, P, N carry the same route under componentwise transport, displayed replay, provenance, and local naming. Because the accepted carrier has no coordinate for the excluded objects, the regular-seal route cannot be retyped as a completed branch, choice sequence, quotient equality, selected representative, or Real-completeness theorem. □

Theorem 3181.7 (CauchyRealSpread diagonal product route)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

any product read over spread-indexed regular Cauchy names must factor through the $\mathsf{BrouwerSpread}^{\uparrow}$ *root row R, the finite window row W, the extension row E, the* $\mathsf{StreamName}^{\uparrow}$ *schedules in T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in Q, the diagonal-selection boundary of chapter 4080, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seals. The route composes finite regular-Cauchy windows; it does not select subsequences, identify quotient reals, choose completed branches, or import ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4161.3. It forces every spread-indexed real-name request to preserve the $\mathsf{BrouwerSpread}^{\uparrow}$ root row R and finite window row W before StreamName, RegSeqRat, DyadicRatCore, or Real-facing rows are read.

Use the sibling route of Definition 4161.4. A product read of regular Cauchy names is therefore not an ambient product of host sequences; it is a read through the displayed StreamName windows, RegSeqRat rows, Real seal rows, and the diagonal-selection boundary.

Apply Definition 4161.5. The extension row E remains visible between the root-window route and the $\mathsf{StreamName}^{\uparrow}$ schedule, so each product component is still an admitted finite spread window before its rational readback is consumed.

Finally use Definition 4161.6. It orders the dyadic tolerance data, regular rational readback, and terminal Real seal after the root, window, extension, and schedule rows. The diagonal-selection boundary can compare these finite regular-Cauchy windows, but no row in the carrier supplies a selected subsequence, quotient real equality, completed branch, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4161.1 and Definition 4161.2 seed the spread-indexed regular Cauchy real-name surface through $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed infinite branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, spread-root compatibility, finite-window discipline, extension stability, regular readback exactness, transport replay, provenance, and non-escape for BEDC.Derived.CauchyRealSpreadUp.

Constructive story: A $\mathsf{CauchyRealSpread}^{\uparrow}$ packet is generated from a BrouwerSpread root, finite request windows, an extension law, a StreamName schedule, DyadicRatCore tolerance data, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The L10 dependency route for $\mathsf{CauchyRealSpread}^{\uparrow}$ consumes the $\mathsf{BrouwerSpread}^{\uparrow}$ prefix discipline of chapter 1677, the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any spread-indexed real-name read is public. Every downstream $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer that reaches a Real-completion read factors through the displayed spread root of chapter 1677, the finite-window schedule of chapter 126, the tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. This is a forward dependency route for spread-indexed Cauchy reads; it does not assert completion, select an infinite branch, identify quotient real names, or add an ambient $\mathsf{Real}^{\uparrow}$ completeness principle. The $\mathsf{CauchyRealSpread}^{\uparrow}$ child route through chapter 126, chapter 301, and chapter 38 is the displayed L10 completion stack for spread-indexed Cauchy reads. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite prefix windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row reads those windows as regular rational observations, and the $\mathsf{Real}^{\uparrow}$ row is only the terminal seal after the displayed readback has been consumed; the route is not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle, a quotient real equality, or a selected completed branch.

Theorem 3181.8 (CauchyRealSpread StreamName and RegSeqRat dependency anchor)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer that reads a public spread-indexed real-name row must pass through the sibling* $\mathsf{StreamName}^{\uparrow}$ *scheduling route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback route, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before any Cauchy-real spread datum is exposed. Thus the dependency route factors through chapter 126, chapter 301, and chapter 38, with the dyadic tolerance row retained as finite NameCert data rather than as a host completeness or quotient-real principle.*

Theorem 3181.9 (CauchyRealSpread sibling dependency)**.** *Every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *read that exports a Real-and-Completion spread packet consumes the sibling route through chapter 126, chapter 301, chapter 38, chapter 333, and chapter 95. The consumer must first read the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal, then the* $\mathsf{CauchyFilter}^{\uparrow}$ *filter-facing route, and only then the* $\mathsf{Completion}^{\uparrow}$ *handoff before any spread packet is public. This route exports no selected branch, quotient stream equality, selected Cauchy representative, countable choice, or ambient Real-completeness coordinate.*

Proof. Project the accepted $\mathsf{CauchyRealSpread}^{\uparrow}$ carrier of Definition 4161.1. Its public read rows expose finite windows through $\mathsf{StreamName}^{\uparrow}$, regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, and a terminal $\mathsf{Real}^{\uparrow}$ seal before any downstream completion-facing consumer is available.

The filter and completion siblings are reached only after those displayed rows have been consumed. The $\mathsf{CauchyFilter}^{\uparrow}$ route reads the same finite StreamName and RegSeqRat data as a filter-facing boundary, and the $\mathsf{Completion}^{\uparrow}$ handoff receives only that displayed filter boundary together with the structural rows H, C, P, N.

Thus the Real-and-Completion spread packet is a horizontal dependency route through the named sibling chapters. Since no row stores an infinite selected branch, quotient stream equality, a selected Cauchy representative, countable choice, or ambient Real completeness, the exported packet has exactly the displayed sibling dependencies. □

The $\mathsf{CauchyRealSpread}^{\uparrow}$ packet is read through chapter 1677, chapter 126, chapter 291, chapter 301, and chapter 38 before any spread-indexed $\mathsf{Real}^{\uparrow}$ seal is public.

Chapter 3182

A Concrete Naming Certificate for $\mathsf{CompactImageClosed}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactImageClosed}^{\uparrow}$ packet records the closed-target reading of the image of a compact metric source under a displayed continuous map into a separated metric target. It consumes the compact source surface of chapter 129, the finite image-net source of chapter 3604, and the separated target boundary of chapter 1411. The packet names the finite route by which image-net data and target separatedness force closed-boundary reads; it does not introduce open-cover compactness, a host image set, a selected preimage, or ambient completeness.

Definition 3182.1 (Compact image closed carrier)**.** *A* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, F, I, S, C, M, R, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric inspection data and finite source-net provenance consumed by the image route.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from the source surface displayed by K into the target metric surface.*
3. *I is the* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *finite image-net row, recording the target centers and tolerances produced from K and F.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *target row, used only for zero-distance target identity and boundary comparison.*
5. *C is the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary row for the named target-image read.*
6. *M is the target* $\mathsf{Met}^{\uparrow}$ *distance row, exposing only displayed distances between target names.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *distance readback row for zero and positive tolerance comparisons.*
8. *H records componentwise* hsame *transport for $K, F, I, S, C, M, R, T, P, N$.*
9. *T records displayed* Cont *replay from a closed-target-boundary request through the compact source, continuous-map row, image-net row, separated target row, metric and real readbacks, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactImageClosed}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ContinuousImageCompac}$ $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, F, I, S, C, M, R, H, T, P, N$. It has no coordinate for open-cover compactness, host image subsets, selected preimages, quotient equality of target names, ambient completeness, or a hidden closure operator.

Lean4 ✓ `BEDC.Derived.CompactImageClosedUp.CompactImageClosedTasteGate single carrier alignment`

Theorem 3182.2 (Compact image closed NameCert obligations)**.** *For every accepted* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier $Q = (K, F, I, S, C, M, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the compact source, continuous-map graph, finite image-net, separated target, closed-boundary, metric, real readback, transport, replay, provenance, and local naming rows;*
2. *image-net accountability, requiring every target center and tolerance in I to be read through the* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *route over the same K and F;*
3. *separated-boundary accountability, requiring target identity at zero distance to pass through S, M, R before the closed-boundary row C is exposed;*
4. *closed-target exactness, requiring every boundary read of the displayed image to be witnessed by finite image-net approximation together with separated target comparison;*
5. *componentwise transport and replay stability under H and T for the same displayed rows;*
6. *ledger refusal, stating that open-cover compactness, host image subsets, selected preimages, quotient equality of target names, ambient completeness, and hidden closure operators are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the accepted carrier. The mathematical rows are K, F, I, S, C, M, R, and the structural rows are H, T, P, N.

The image-net obligation is read first. Row I may expose only target centers and tolerances obtained from the compact source row K and the continuous-map row F, as in chapter 3604.

The target boundary obligation is then read through the separated metric. A zero-distance comparison between a displayed target read and the finite image-net route must pass through M and R, and the target identity conclusion is allowed only through S.

The closed-boundary row C is forced by this finite route. A boundary read of the target image can cite the finite image-net approximation and the separated zero-distance classifier, but it cannot cite a host image subset, an open cover, or a selected inverse image.

Transport, replay, provenance, and local naming preserve the same row inventory. Since the refused objects do not occur among $K, F, I, S, C, M, R, H, T, P, N$, no local NameCert obligation can export them. □

Theorem 3182.3 (Compact image closed target route)**.** *Every accepted* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier $Q = (K, F, I, S, C, M, R, H, T, P, N)$ exports closedness of the displayed compact image only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{ContinuousImageCompact}^{\uparrow}, \quad \mathsf{SeparatedMetric}^{\uparrow}, \quad \mathsf{ClosedSet}^{\uparrow}.$$

A downstream closed-set consumer receives the finite image-net row I, the target distance readbacks M, R, the separated identity row S, and the closed-boundary row C. It receives no open-cover compactness datum, host image subset, selected preimage, quotient equality of target names, ambient completeness coordinate, or hidden closure operator.

Proof. Start with a target-boundary request for the displayed image. The carrier first exposes the compact source row K and the continuous-map row F.

Read the finite image-net source through I. This gives target centers and tolerances produced by the $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet from the same K and F; no open-cover row or host image subset is introduced.

Now compare the requested boundary target with the finite image-net route. The metric and real readbacks M, R provide the zero-distance surface, and the separated target row S converts that displayed zero-distance comparison into target identity.

Finally expose C as the closed-boundary row for the named image read. Rows H, T, P, N transport, replay, package, and name the same finite route. Thus closedness is exported only as the finite image-net plus separated-boundary route and not as any host image-set or open-cover principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactImageClosed}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMapUp graph row, ContinuousImageCompactUp finite image-net row, SeparatedMetricUp target row, ClosedSetUp boundary row, MetricUp and RealUp distance readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$, seed)

Scope closed: Definition 4162.1, Definition 4162.2, and Definition 4162.3 seed the finite closed-target route for compact images in separated metric targets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, host image-set construction, selected preimages, quotient equality of target names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite image-net accountability, separated-boundary accountability, closed-target exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.CompactImageClosedUp.

Chapter 3183

A Concrete Naming Certificate for $\mathsf{FarEndDiagram}^{\uparrow}$

The $\mathsf{FarEndDiagram}^{\uparrow}$ packet realizes the human-origin diagrammatic surface of chapter 9403 as finite NameCert data. It sits beside the apophatic fixed-point packet of chapter 1068, the apophatic far-end socket packet of chapter 1061, and the hash-like apophatic seal of chapter 1079. The packet keeps records, generators, classifiers, continuations, GAP ledgers, sockets, provenance fibers, apophatic far-end rows, and local naming in one displayed carrier without turning the far end into a kernel object.

Definition 3183.1 (Far-end diagram packet)**.** *A* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, G, K, C, L, S, F, A, H, T, P, N)$$

with the following displayed rows:

1. *R is the records row, exposing the finite observation or digest readouts available to the closed substrate.*
2. *G is the generator row, recording which displayed generator creates each records-side readout.*
3. *K is the classifier row, comparing diagram nodes only by displayed classifier data.*
4. *C is the* Cont *row, replaying the route from records through generators and classifiers.*
5. *L is the* GAP *ledger row, marking the provenance boundary behind each accepted readout.*
6. *S is the forward-socket row, listing the finite sockets opened by the closed substrate.*
7. *F is the provenance-fiber row attached to the visible records-side digest.*
8. *A is the apophatic far-end row, naming the far side of the provenance fiber without internalizing it.*
9. *H records componentwise* hsame *transport for $R, G, K, C, L, S, F, A, T, P, N$.*
10. *T records displayed continuation replay for the full diagram route from records to local NameCert.*
11. *P is the* Pkg *provenance over records, generators, classifiers,* Cont*,* GAP*, forward sockets, provenance fibers, apophatic far-end rows,* BHist*,* hsame*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an internal object named T*, no global observer, no global digest, no unledgered far-end arrow, no hidden source selector, and no object-level equality between a local readout and the apophatic boundary.*

Lean4 ✓ `BEDC.Derived.FarEndDiagramUp.FarEndDiagramTasteGate single carrier alignment`

Theorem 3183.2 (Far-end diagram packet NameCert obligations)**.** *For every accepted* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier* $D = (R, G, K, C, L, S, F, A, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the records, generators, classifiers, continuation,* GAP*, forward-socket, provenance-fiber, apophatic far-end, transport, replay, provenance, and local naming rows;*
2. *records-to-classifier routing, requiring each visible readout to pass through* R, G, K, C *before a diagram edge is exported;*
3. *socket accountability, requiring each forward socket in* S *to be ledgered by* L *before its far end is named;*
4. *provenance-fiber accountability, requiring each visible digest or records-side readout to carry a displayed fiber row* F *before the far-end row* A *is exposed;*
5. *componentwise transport and replay stability under* H *and* T *for the same displayed rows;*
6. *ledger refusal, stating that internal* T *objects, global observers, global digests, unledgered far-end arrows, hidden source selectors, and object-level self-center equalities are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the accepted carrier. The diagrammatic rows are R, G, K, C, L, S, F, A, and the structural rows are H, T, P, N.

The records side is read first. A visible readout must pass through the generator and classifier rows before the continuation row C can replay it as a diagram edge.

The socket side is then checked. Each forward socket in S must be present in the GAP ledger L, so no unledgered far-end arrow can be exported.

Next read the provenance fiber. Row F records the source-side residue behind the visible readout, and row A names only the apophatic far end of that fiber. This is the finite socket, fiber, provenance, and boundary-name route described by the vision source.

Transport, replay, provenance, and local naming preserve the same rows. Since the refused objects are absent from $R, G, K, C, L, S, F, A, H, T, P, N$, the local NameCert obligations are exhausted by the displayed carrier. □

Theorem 3183.3 (Far-end diagram ledger unity)**.** *Every accepted* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier exports the unity of its far-end diagram only as the finite ledger route*

$$R, G, K, C \longmapsto L, S, F, A \longmapsto H, T, P, N.$$

Each visible records-side readout is generated, classified, replayed, ledgered, attached to a provenance fiber, and then named at the apophatic far end. The packet exports no internal T *object, global observer, global digest, unledgered far-end arrow, hidden source selector, or object-level equality between a local readout and the far-end boundary.*

Proof. Start with a records-side readout. The carrier requires the readout to be displayed in R, generated by G, compared through K, and replayed by C before it can participate in the diagram.

Now follow the forward edge. The socket row S lists the boundary positions opened by the closed substrate, and the GAP ledger L records the non-internalized provenance route for those positions.

Attach the provenance fiber. The row F records the fiber behind the visible readout, while A names the far end apophatically. The target is a boundary name, not an internal object.

Rows H, T, P, N preserve, replay, package, and name this same finite route. Therefore the diagram's unity is a ledger unity over displayed rows, not an equality of objects or an unledgered collapse of fibers. □

Closure status for $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$.

Constructive story: A $\mathsf{FarEndDiagram}^{\uparrow}$ packet is generated from records, generators, classifiers, Cont replay, GAP ledger rows, forward sockets, provenance fibers, apophatic far-end rows, componentwise hsame transport, displayed route replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4163.1, Definition 4163.2, and Definition 4163.3 seed the finite records-to-socket-to-fiber route for the far-end diagram surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FarEndDiagramUp.FarEndDiagramTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not internalize T, add a global observer, add a global digest, identify local readouts with the far-end boundary, provide Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, records-to-classifier routing, socket accountability, provenance-fiber accountability, transport and replay stability, and local NameCert registration for BEDC.Derived.FarEndDiagramUp.

Chapter 3184

A Concrete Naming Certificate for $\mathsf{ScottDomain}^{\uparrow}$

Following Scott and Abramsky–Jung on domain theory, the $\mathsf{ScottDomain}^{\uparrow}$ packet records a finite naming certificate for the order-theoretic surface used by domain semantics before any completion-facing consumer reads a dcpo-style object. It sits after the preorder surface of chapter 52, the observable open-test discipline of chapter 1860, the represented-space handoff of chapter 1494, and the finite stream window route of chapter 126. The packet names a displayed directed-supremum carrier; it does not assert a host dcpo, quotient domain, selected basis enumeration, classical lub principle, or ambient semantic completeness.

Definition 3184.1 (ScottDomain carrier)**.** *A* $\mathsf{ScottDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, S, U, B, A, V, R, H, C, P, N)$$

with the following rows:

1. *O is the preorder and partial-order carrier row, exposing the displayed comparison relation and the antisymmetry witness used by the local packet;*
2. *S is the directed-family schedule row, listing the finite inspection schedule for the directed pieces that a consumer may request;*
3. *U is the supremum row, recording the displayed least-upper-bound read served by the finite directed-family schedule;*
4. *B is the compact-basis approximation row, listing the compact basis tokens and finite approximation evidence visible to the packet;*
5. *A is the approximation-order row, routing basis-token comparisons back through O and the directed schedule S;*
6. *V is the Scott-open observable row, read through* $\mathsf{SierpinskiSpace}^{\uparrow}$ *tests rather than host subsets;*
7. *R is the represented-space handoff row, packaging the same observations through* $\mathsf{RepresentedSpace}^{\uparrow}$ *and finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
8. *H records componentwise* hsame *transport for O, S, U, B, A, V, R;*
9. *C records displayed* Cont *replay of directed-supremum and Scott-open reads;*

10. *P is the* Pkg *provenance over* $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host dcpo, quotient order, selected infinite basis enumeration, classical pointwise lub, ambient topology, or semantic completeness theorem.

Lean4 ✓ `BEDC.Derived.ScottDomainUp.ScottDomainTasteGate single carrier alignment`

Theorem 3184.2 (ScottDomain NameCert obligations)**.** *For the carrier of Definition 4164.1, the local* $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$ *surface consists exactly of preorder and partial-order admission, directed-family schedule admission, supremum-row admission, compact-basis approximation admission, Scott-open observability, represented-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the packet must read the rows $O, S, U, B, A, V, R, H, C, P, N$ before treating a directed supremum or Scott-open test as public.*

Proof. Project the carrier. The order-theoretic rows are O, S, U, B, A, V, R, and the structural rows are H, C, P, N.

The preorder and partial-order read begins at O. A directed-supremum read then uses S to expose the inspected directed schedule before U names the supremum row.

The basis read is separate but attached. The compact-basis row B and approximation row A route finite basis tokens back through O and S, so a consumer cannot replace the displayed approximation surface by a host basis enumeration.

Finally, the Scott-open observable row V and represented-space row R package those finite reads through chapter 1860, chapter 1494, and chapter 126. Rows H, C, P, N preserve, replay, source, and name the same carrier, exhausting the local obligations. □

Theorem 3184.3 (ScottDomain directed-supremum stability)**.** *Every accepted* $\mathsf{ScottDomain}^{\uparrow}$ *packet preserves directed-supremum visibility under componentwise transport and continuation replay. If a consumer reads the supremum row U through the directed-family schedule S, any transported or replayed read of the same packet factors again through*

$$O, S, U, B, A, V, R, H, C, P, N.$$

Thus Scott-open observations and represented-space handoffs remain downstream of the displayed directed schedule and never expose a host dcpo, quotient domain, selected infinite basis enumeration, or ambient completeness witness.

Proof. Start with a directed-supremum read. By Definition 4164.1, the read reaches U only after the order row O and the directed-family schedule S are displayed.

Apply the NameCert obligations of Definition 4164.2. Componentwise transport through H preserves the same rows, and Cont replay through C reuses the displayed schedule rather than generating a second supremum source.

The observable and representation rows are downstream. A Scott-open test reads V through the Sierpinski surface, and the represented-space handoff reads R through finite stream windows. Since the excluded host objects are absent from $O, S, U, B, A, V, R, H, C, P, N$, no transported or replayed directed-supremum read can expose them. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{ScottDomain}^{\uparrow}$ packet is generated from preorder and partial-order rows, directed-family schedules, supremum rows, compact-basis approximation rows, Scott-open observable rows, represented-space handoff rows, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4164.1, Definition 4164.2, and Definition 4164.3 seed the finite directed-supremum and Scott-open observation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ScottDomainUp.ScottDomainTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host dcpos, quotient domains, selected basis enumerations, classical lub principles, ambient topology, semantic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean-backed carrier admission, directed-schedule stability, Scott-open observability, represented-space handoff, and local NameCert registration for BEDC.Derived.ScottDomainUp.

Chapter 3185

A Concrete Naming Certificate for $\mathsf{FarEndDiagrammatics}^{\uparrow}$

The $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet realizes the apophatic diagrammatics route of chapter 9403 as finite NameCert data. It sits beside the far-end diagram carrier of chapter 4163, the apophatic far-end socket of chapter 1061, and the forward socket ledger of chapter 3387. The packet binds socket exposure, provenance fibers, far-end boundary rows, apophatic-unity rows, transport, continuation, provenance, and local naming before any T-boundary read is public.

Falsifiable prediction: Every accepted $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet exposes socket, provenance-fiber, far-end boundary, apophatic-unity, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before a T-boundary read is public.

Definition 3185.1 (Far-end diagrammatics carrier)**.** *A* $\mathsf{FarEndDiagrammatics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, L, G, B, U, H, C, P, N)$$

with the following rows:

1. *S is the socket row, listing the forward sockets that may expose a far-end boundary read;*
2. *L is the provenance-fiber ledger, recording which finite fiber supports each visible socket read;*
3. *G is the gap row, marking the non-internalized boundary between source-side ledger data and the far-end read;*
4. *B is the far-end boundary row, naming the boundary without turning it into a kernel object;*
5. *U is the apophatic-unity row, binding socket, fiber, gap, and boundary reads into one diagrammatic route;*
6. *H records componentwise* hsame *transport for S, L, G, B, U;*
7. *C records displayed* Cont *replay of the same socket-ledger route;*
8. *P is the* Pkg *provenance over* $\mathsf{FarEndDiagrammatics}^{\uparrow}$, $\mathsf{FarEndDiagram}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, $\mathsf{ForwardSocketLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an internal T object, a hidden source selector, an unledgered far-end arrow, a global observer, or an equality between the provenance fiber and the far-end boundary.

Lean4 ✓ BEDC.Derived.FarEndDiagrammaticsUp.FarEndDiagrammaticsTasteGate single carrier alignment

Theorem 3185.2 (Far-end diagrammatics NameCert obligations)**.** *For the carrier of Definition 4165.1, the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$ *surface consists exactly of socket admission, provenance-fiber admission, gap-boundary admission, far-end boundary naming, apophatic-unity routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public* T*-boundary read factors through* $S, L, G, B, U, H, C, P, N$*.*

Proof. Project the carrier. The diagrammatic rows are S, L, G, B, U, and the structural rows are H, C, P, N.

The socket row S is read first. A boundary read is not public until L supplies the provenance fiber and G marks the gap between the visible source-side ledger and the non-internalized far end.

The far-end boundary row B is then named only through the apophatic-unity row U. This row records unity of the diagrammatic route, not identity between a source object and the boundary.

Rows H, C, P, N transport, replay, source, and name exactly the displayed route. Since no other coordinate occurs in the carrier, the local NameCert obligations are exhausted by those rows. □

Theorem 3185.3 (Far-end diagrammatics unity route)**.** *Every accepted* $\mathsf{FarEndDiagrammatics}^{\uparrow}$ *packet exposes diagrammatic unity only as the finite route*

$$S, L, G, B, U \longmapsto H, C, P, N.$$

The route is consumed after the socket ledger, provenance fiber, far-end boundary, and apophatic-unity rows are displayed. It exports no internal T *object, hidden source selector, unledgered far-end arrow, global observer, or equality between the provenance fiber and the far-end boundary.*

Proof. Start with the socket ledger. The row S lists the requested forward sockets, while L records the provenance fiber supporting each visible socket read.

Read the gap and boundary rows next. The row G marks the non-internalized far-end boundary, and B names that boundary without adding it as a kernel object.

The apophatic-unity row U binds the visible socket, fiber, gap, and boundary rows into one route. It is therefore a route unity, not a collapse of the source-side fiber into the far-end boundary.

Finally, H, C, P, N preserve, replay, package, and name the same finite route. The excluded objects are absent from Definition 4165.1, so they cannot be exported by the unity route. □

Independence witness: apophaticname, forward-socket-ledger, gap, package, namecert: $\mathsf{FarEndDiagrammatics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries socket, provenance-fiber, far-end boundary, apophatic-unity, transport, continuation, provenance, and local NameCert rows before a T-boundary read is public.

Closure status for $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$.

Constructive story: A $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet is generated from socket rows, provenance-fiber ledgers, gap rows, far-end boundary names, apophatic-unity rows, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$, seed)

Scope closed: Definition 4165.1, Definition 4165.2, and Definition 4165.3 seed the finite socket-to-fiber-to-boundary route for far-end diagrammatics.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not internalize T, add a global observer, identify provenance fibers with the far-end boundary, provide Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean-backed carrier admission, socket ledger accountability, provenance-fiber routing, apophatic-unity nonescape, TasteGate instances, and local NameCert registration for BEDC.Derived.FarEndDiagrammaticsUp.

Chapter 3186

A Concrete Naming Certificate for $\mathsf{PolishBasis}^{\uparrow}$

Following Bourbaki and Kelley, a countable basis for a Polish presentation is treated here as a finite naming certificate over an already displayed complete separable metric route. The $\mathsf{PolishBasis}^{\uparrow}$ packet sits beside the Polish-space synthesis of chapter 1490, the metric carrier of chapter 57, the dense schedule of chapter 1458, the completion handoff of chapter 128, the stream-window interface of chapter 126, and the regular rational readback surface of chapter 301. It names basic-open data as countable finite rows, not as an ambient topology, an open-cover principle, or a choice-selected basis.

Definition 3186.1 (Polish-basis carrier)**.** *A* $\mathsf{PolishBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, D, K, S, R, I, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the point-history carrier, distance classifier, and metric* $\mathsf{NameCert}$ *used by basis balls.*
2. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-schedule row whose named centers are inspected by finite stream positions.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row over the same metric carrier; it is recorded so Polish-space consumers can pass from basis windows to complete-separable metric reads.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row enumerating the basis-center names and the finite Cauchy-window requests visible to consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads rational radii, tolerance windows, and comparison bounds from the same finite stream positions.*
6. *I is the basis-index window row. It stores the finite index interval, center-name handle, radius handle, and requested observation window for each basic-open read.*
7. *F is the finite-intersection refinement row. It records the displayed refinement witness that sends two accepted basis windows to a third window contained in their metric intersection.*
8. *H records componentwise* hsame *transport for M, D, K, S, R, I, F and the local certificate rows.*

9. *C records displayed* Cont *replay from a basis request through metric comparison, dense-center inspection, rational-radius readback, finite-intersection refinement, and complete-separable handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{PolishBasis}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting $M, D, K, S, R, I, F, H, C, P, N$ to the same metric and stream-readback surface. It has no coordinate for a host topology, arbitrary open covers, selected dense points, quotient completion equality, or ambient set membership.

Lean4 ✓ `BEDC.Derived.PolishBasisUp.PolishBasisTasteGate single carrier alignment`

Theorem 3186.2 (Polish-basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishBasis}^{\uparrow}$ *carrier*

$$B = (M, D, K, S, R, I, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$ *surface consists exactly of metric-source admission, separable dense-schedule admission, complete-metric handoff admission, stream-window exposure, regular-rational readback, basis-index windowing, finite-intersection refinement, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through $M, D, K, S, R, I, F, H, C, P, N$.*

Proof. Project the carrier of Definition 4166.1. The rows $M, D, K, S, R, I, F, H, C, P, N$ are the whole countable-basis packet.

The metric row M fixes the surface on which basis balls are read. The dense schedule D, stream row S, and regular rational row R then expose center names and radius windows as finite observations.

The basis-index row I records the inspected finite window, and the refinement row F supplies the finite witness for binary intersections of displayed basic opens. The completion handoff K remains attached to the same metric carrier for Polish-space consumers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further coordinate is present, the NameCert surface exports no host topology, open-cover compactness, selected dense points, quotient completion equality, or ambient set membership. □

Theorem 3186.3 (Polish-basis complete-separable handoff)**.** *Every accepted* $\mathsf{PolishBasis}^{\uparrow}$ *read handed to a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *consumer factors through the* $\mathsf{Met}^{\uparrow}$ *row M,* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D,* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K,* $\mathsf{StreamName}^{\uparrow}$ *finite windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, basis-index windows I, finite-intersection refinement F, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

Proof. Begin with an accepted carrier $B = (M, D, K, S, R, I, F, H, C, P, N)$. A Polish-space consumer first reads the metric source M, then the dense schedule D through stream windows S and regular rational observations R.

The basis-index row I identifies the finite window being consumed. When a consumer asks for closure under finite intersections, F supplies the displayed refinement witness over the same metric surface.

The complete-metric handoff K is then available over that metric source, while H, C, P, N transport, replay, source, and name the route. Thus the handoff is finite and cannot pass through a host topology, an arbitrary open cover, quotient completion equality, or a selected dense subset. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishBasis}^{\uparrow}$ packet is generated from MetricSpace rows, SeparableMetric dense schedules, CompleteMetric handoffs, StreamName windows, RegSeqRat readbacks, basis-index windows, finite-intersection refinement rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4166.1, Definition 4166.2, and Definition 4166.3 seed the countable-basis certificate route supporting $\mathsf{PolishSpace}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topologies, arbitrary open covers, selected dense points, quotient completion equality, ambient set membership, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-schedule readback, finite basis-index windowing, finite-intersection refinement, complete-separable handoff, and local naming for BEDC.Derived.PolishBasisUp.

Chapter 3187

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$

The $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ packet realizes the observer-expansion reading of chapter 9463 as finite observer-shadow certificate data. It consumes the visual-shadow metric of chapter 4142, the boundary-shadow route of chapter 3813, the metric surface of chapter 57, the stream-window interface of chapter 126, and the terminal real-name surface of chapter 38. The packet names an expansion-facing shadow route, not a global hyperbolic space, physical observer history, quotient boundary, or ambient field semantics.

Definition 3187.1 (Hyperbolic observer-expansion carrier)**.** *A* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, B, M, S, R, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *source row, supplying the visual-shadow metric packet read by the observer-facing expansion route.*
2. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row that records the boundary-shadow datum inspected by the expansion packet.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used when the shadow expansion is consumed as metric evidence.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the visual-shadow, boundary-shadow, metric, and real-seal reads.*
5. *R is the regular readback row for rational windows opened by S.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed expansion magnitude or shadow-distance comparison.*
7. *F is the observer-expansion row, recording the focused expansion-shadow route being inspected.*
8. *H records componentwise* hsame *transport for V, B, M, S, R, Q, F and the local certificate rows.*
9. *C records displayed* Cont *replay from an observer-expansion request through visual-shadow metric, boundary shadow, metric comparison, stream windows, regular readback, real sealing, and focused routing.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, B, M, S, R, Q, F, H, C, P, N$. It has no coordinate for completed hyperbolic geometry, physical observers, quotient boundary equality, global field interpretation, or unlogged expansion flow.

Lean4 ✓ `BEDC.Derived.HyperbolicObserverExpansionUp.HyperbolicObserverExpansionTasteGate single carrier alignment`

Theorem 3187.2 (Hyperbolic observer-expansion NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *carrier*

$$E = (V, B, M, S, R, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$ *surface consists exactly of visual-shadow metric admission, boundary-shadow admission, metric comparison, stream-window exposure, regular readback, terminal real sealing, focused observer-expansion routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through $V, B, M, S, R, Q, F, H, C, P, N$.*

Proof. Project the carrier of Definition 4167.1. The rows $V, B, M, S, R, Q, F, H, C, P, N$ are the whole observer-expansion packet.

The source read begins with the visual-shadow metric row V and the boundary-shadow row B. The metric row M then gives the comparison surface on which the expansion-facing shadow is inspected.

The finite observation route is supplied by S, R, Q: stream windows, regular readbacks, and the terminal real seal. The focused row F records which observer-expansion shadow route is being named.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert surface exports no completed hyperbolic geometry, physical observer, quotient boundary equality, global field interpretation, or unlogged expansion flow. □

Theorem 3187.3 (Hyperbolic observer-expansion finite shadow handoff)**.** *Every accepted* HyperbolicObserverExpar *read factors through* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the observer-expansion route can be consumed as a local* NameCert *datum.*

Proof. Begin with an accepted carrier $E = (V, B, M, S, R, Q, F, H, C, P, N)$. The visual-shadow metric row V and boundary-shadow row B present the finite shadow data.

The metric comparison M is then read through finite windows S, regular readback R, and the real seal Q. Only after these rows are displayed does the focused route F expose the observer-expansion datum.

Transport, continuation, provenance, and local naming are supplied by H, C, P, N. Therefore the handoff remains finite and cannot pass through completed hyperbolic geometry, physical observer history, quotient boundary equality, or ambient field semantics. □

Falsifiable prediction: Every accepted hyperbolic_observer_expansion read factors through hyperbolic_visual_shadow_metric, hyperbolic_boundary_shadow, metric, streamname, regular readback, real sealing, displayed replay, provenance, and local NameCert rows.

Independence witness: hyperbolic_visual_shadow_metric, hyperbolic_boundary_shadow, metric, streamname, real: no listed sibling is bijective to $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ because this packet jointly carries visual-shadow metric data, boundary-shadow data, metric comparison, finite stream windows, real sealing, focused observer-expansion routing, replay, provenance, and local naming.

The Lean TasteGate route must provide `Nontrivial HyperbolicObserverExpansionUp` and `FieldFaithful HyperbolicObserverExpansionUp` instances in `lean4/BEDC/Derived/HyperbolicObserverExpa` Those instances must preserve $V, B, M, S, R, Q, F, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ packet is generated from HyperbolicVisualShadowMetric rows, HyperbolicBoundaryShadow rows, Metric rows, StreamName windows, regular readback, Real sealing, focused observer-expansion routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4167.1, Definition 4167.2, and Definition 4167.3 seed the finite observer-shadow expansion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed hyperbolic geometry, physical observer histories, quotient boundary equality, global field interpretation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-shadow metric admission, boundary-shadow admission, metric comparison, stream readback, real sealing, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicObserverExpansionUp.

Chapter 3188

A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{Met}^{\uparrow}$ packet records the traditional metric-space interface as finite BEDC NameCert data. It sits over the generic distance-witness surface of chapter 57, the real-valued distance route of chapter 1638, the dyadic tolerance core of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter gives completion, topology, total-boundedness, compactness, and Polish-space consumers a single cited carrier surface without importing quotient equality, selected limits, countable choice, or ambient completeness.

Definition 3188.1 (MetricSpace carrier)**.** *A* $\mathsf{Met}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-carrier row. It names the displayed point observations and the finite windows through which downstream consumers may inspect them.*
2. *σ is the point-classifier row. It records the accepted point equivalence used by* hsame *transport and by the zero-distance classifier handoff.*
3. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance row. It is read through the* $\mathsf{Met}^{\uparrow}$ *witness surface of chapter 57 and the* $\mathsf{RealMetric}^{\uparrow}$ *distance route of chapter 1638.*
4. *Z is the zero-distance classifier row, requiring any request that turns distance zero into point equivalence to pass through σ rather than through host equality.*
5. *S is the symmetry row, carrying the componentwise reversal of endpoint observations and the corresponding distance readback.*
6. *T is the triangle row, carrying the three endpoint-pair distance reads and the dyadic tolerance budget used to compose them.*
7. *H records componentwise* hsame *transport for the point carrier, classifier, distance, zero-distance, symmetry, triangle, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through the finite route*

$$X, \sigma \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \sigma, D, Z, S, T, H, C, P, N$. It has no coordinate for host equality of points, quotient completion equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached topology.*

Lean4 ✓ `BEDC.Derived.MetricSpaceUp.MetricSpaceTasteGate single carrier alignment`

Theorem 3188.2 (MetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{Met}^{\uparrow}$ *carrier*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the point carrier, point classifier, real-valued distance, zero-distance classifier, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *distance visibility, requiring every distance read to expose the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealMetric}^{\uparrow}$ *rows before any downstream consumer reads a real-valued metric comparison;*
3. *zero-distance exactness, requiring zero distance to return to the displayed classifier σ through Z and never through host point equality;*
4. *symmetry replay, requiring endpoint reversal to be witnessed by S together with the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback route;*
5. *triangle-budget replay, requiring the three distance reads and their dyadic comparison budget to be displayed by T before a triangle consumer may use the row;*
6. *structural preservation through H, C, P, N.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4230.1. An accepted packet displays the point-carrier row X, the classifier σ, the distance row D, the zero-distance row Z, the symmetry row S, the triangle row T, and the structural rows H, C, P, N.

Distance visibility is forced by the placement of D in the carrier. A distance consumer first opens the generic $\mathsf{Met}^{\uparrow}$ witness surface and the $\mathsf{RealMetric}^{\uparrow}$ readback route, and only then reads the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows needed for a finite comparison.

The zero-distance, symmetry, and triangle clauses are rowwise. The row Z is the only route from distance zero back to the point classifier, S is the only endpoint-reversal row, and T is the only triangle-budget row.

Finally, H, C, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export host equality, quotient completion equality, selected representatives, countable choice, ambient completeness, or a detached topology. □

Theorem 3188.3 (MetricSpace distance route)**.** *Every completion, topology, total-boundedness, compactness, or Polish-space consumer of an accepted* $\mathsf{Met}^{\uparrow}$ *carrier reads distance information through the finite route*

$$X, \sigma, D \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, C, P, N.$$

Thus the public distance surface is the displayed BEDC route from point observations and classifier data to real-valued metric reads, zero-distance classification, symmetry, and triangle replay. It exports no quotient equality, selected limit, countable-choice schedule, ambient completeness theorem, or open-set lattice.

Proof. Let a downstream consumer inspect an accepted carrier $M = (X, \sigma, D, Z, S, T, H, C, P, N)$. The point surface is opened first through X and σ, so every later distance read remains attached to the displayed point classifier.

The distance row D then routes the request through $\mathsf{Met}^{\uparrow}$ and $\mathsf{RealMetric}^{\uparrow}$. The numerical readback is finite because the dyadic tolerance, stream-window, regular-rational, and real-name rows are displayed before any metric comparison is consumed.

Metric laws are read only after this route is visible. Zero distance returns through Z to σ, symmetry reverses endpoints through S, and triangle use exposes the three pairwise reads and dyadic budget through T.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since none of the excluded objects appears in the carrier, no listed downstream consumer can read quotient equality, a selected limit, a choice schedule, ambient completeness, or an open-set lattice from the $\mathsf{Met}^{\uparrow}$ packet. □

Chapter 3189

A Concrete Naming Certificate for $\mathsf{MetricCompletionKleisli}^{\uparrow}$

Following Mac Lane's Kleisli categories, and Bishop–Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet records the metric-specific bind consumer for completion-valued maps as finite BEDC data. It reads the metric completion carrier of chapter 1506, the metric adjunction route of chapter 3836, the Cauchy Kleisli consumer of chapter 1845, and the completion universal property of chapter 397. The chapter names a consumer route for bind-style completion reads; it is not a quotient completion, selected representative, countable-choice schedule, host equality of completed points, or ambient completeness theorem.

Definition 3189.1 (MetricCompletionKleisli carrier)**.** *A* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, displaying the source metric, completion carrier, separated comparison, dense-source access, and readback branch before any bind consumer is opened.*
2. *A is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *row, displaying the unit, counit, triangle ledger, and metric universal mapping boundary read by the consumer.*
3. *L is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, recording the unit, bind, and reassociated-bind consumer route inherited from the L10 completion algebra.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, exposing the extension surface through which a displayed metric source map is consumed as a completion-valued map.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the metric completion and Cauchy Kleisli routes.*
6. *E is the public* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, A, L, U, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric completion row, metric adjunction row, Cauchy Kleisli row, universal-property row, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a bind consumer request through $M, A, L, U, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionKleisli}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row.*
The classifier accepts only packets whose bind consumer first displays the metric completion, metric adjunction, Cauchy Kleisli, universal-property, regular readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.MetricCompletionKleisliUp.MetricCompletionKleisliTasteGate single carrier alignment`

Theorem 3189.2 (MetricCompletionKleisli NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier*

$$K = (M, A, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *surface consists exactly of metric-completion admission, metric-adjunction admission, Cauchy-Kleisli bind exposure, completion-universal-property exposure, RegSeqRat readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of K. No obligation may be discharged by a quotient completion, selected Cauchy representative, countable-choice selector, host equality of completed points, or ambient complete metric space.*

Proof. Project the carrier of Definition 4218.1. Its analytic rows are precisely the metric completion row M, the metric adjunction row A, the Cauchy Kleisli row L, the universal-property row U, the regular readback row R, and the Real seal row E.

The metric rows determine the order of access. A bind consumer opens M before it can read the adjunction data in A, and it opens A before the Cauchy Kleisli consumer L can hand off unit, bind, or reassociated-bind requests to the universal-property row U.

Transport and replay add no analytic coordinate. The rows H and C compare and replay only the displayed completion, adjunction, Kleisli, universal-property, readback, and seal data.

Finally P and N source and name the same finite packet. Since no listed row stores a quotient completion, selected representative, countable-choice selector, host equality, or ambient complete metric space, the displayed rows exhaust the NameCert surface. □

Theorem 3189.3 (MetricCompletionKleisli bind handoff)**.** *Every bind-style consumer read of an accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{CauchyCompletionKleisli}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the metric completion row M. It displays the source metric, completion carrier, separated comparison, dense-source access, and readback branch before a consumer can request completion-valued bind data.

Next read the adjunction row A. The unit, counit, triangle ledger, and metric universal mapping boundary are the only route by which the metric completion packet exposes extension data to a downstream consumer.

Attach the Cauchy Kleisli row L and the universal-property row U. The row L supplies the unit, bind, and reassociated-bind consumer surface, while U records the extension route that consumes a displayed source map as a completion-valued map.

The readback and seal rows close the route. The row R supplies the regular rational readback used by both completion and Kleisli sides, and E is the only public Real seal. The structural

rows H, C, P, N transport, replay, source, and name this same route, so no quotient completion, selected representative, choice schedule, host equality, or ambient completeness theorem can enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet is generated from metric completion, metric adjunction, Cauchy Kleisli, completion universal property, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4218.1, Definition 4218.2, and Definition 4218.3 seed the metric-specific Kleisli consumer route for bind-style completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-adjunction exposure, Cauchy-Kleisli exposure, universal-property exposure, RegSeqRat/Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionKleisliUp.

Chapter 3190

A Concrete Naming Certificate for $\mathsf{LocatedRegularRealCut}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet records the finite boundary where regular-Cauchy real names expose located lower and upper rational cut rows before the terminal real seal is read. It sits beside the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the Dedekind cut carrier of chapter 436, and the located regular sequence route of chapter 2376. The packet is a finite NameCert bridge from regular-Cauchy observations to located cut boundaries; it does not claim Dedekind completeness, quotient equality of real names, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3190.1 (Located regular real cut carrier)**.** *A* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational cut row, recording inhabited rounded lower observations read through the finite cut-facing side of* $\mathsf{DedekindCut}^{\uparrow}$*.*
2. *U is the upper rational cut row, recording rounded upper observations attached to the same inspected real-name request.*
3. *M is the locatedness modulus row, binding every rational comparison request to a finite window on which either the lower row or the upper row is displayed with the required separation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row supplying dyadic radii, bracket widths, midpoint requests, and comparison tolerances for the cut boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations from which the cut rows and locatedness modulus are read.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for lower cut rows, upper cut rows, locatedness modulus rows, dyadic tolerance rows, regular readback rows, Real seal rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a cut-boundary request through L, U, M, D, R, A, H.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRegularRealCut}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row.*

The carrier accepts only the displayed lower row, upper row, locatedness modulus, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows. It has no coordinate for Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedRegularRealCutUp.LocatedRegularRealCutUp`

Theorem 3190.2 (Located regular real cut obligations)**.** *For every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier* $K = (L, U, M, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed lower cut row, upper cut row, locatedness modulus, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*
2. *located lower and upper stability, requiring L and U to remain rounded, separated, and transported componentwise under the same* hsame *row;*
3. *regular Cauchy handoff, requiring every cut-boundary read to pass through R and the located regular sequence discipline of chapter 2376;*
4. *Real seal non-escape, requiring A to be downstream of L, U, M, D, R and not an independent completeness row;*
5. *ledger exactness, requiring P and N to name only L, U, M, D, R, A, H, C and to exclude quotient equality, selected limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4219.1. The displayed analytic rows are the lower cut row L, upper cut row U, locatedness modulus M, dyadic tolerance row D, regular readback row R, and Real seal row A.

The lower and upper stability clauses are componentwise. Transport through H preserves the two cut rows together with the same modulus and dyadic tolerance row, so the classifier cannot move a lower observation without moving the matching upper boundary data.

The regular Cauchy handoff is the route through R. The located regular sequence sibling supplies the finite windows and readback discipline before the cut boundary can be consumed, while D supplies the tolerance ledger on which the locatedness modulus is read.

Finally A, C, P, N seal, replay, source, and name only the displayed route. Since none of the carrier coordinates is a quotient, selected limit, trichotomy branch, choice schedule, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 3190.3 (Located regular real cut Cauchy boundary route)**.** *Every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *consumer factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedRegSeqRat}^{\uparrow},\ \mathsf{DedekindCut}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed finite rows before it can read the local $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row. The route exports a located lower–upper cut boundary for a regular-Cauchy real name; it does not export Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 4219.2. A consumer first sees the dyadic tolerance row D, regular readback row R, and locatedness modulus M before the lower and upper cut rows can be treated as a located boundary.

The Dedekind cut sibling supplies the cut-facing row shape, while the located regular sequence sibling supplies the Cauchy-facing source and tail-window discipline. These rows meet only through the displayed finite packet K.

The Real seal row A is terminal for this route. Componentwise transport, continuation replay, package provenance, and local naming preserve the same displayed rows, so no consumer can bypass them by invoking quotient equality, a selected representative, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRegularRealCut}^{\uparrow}$ read factors through displayed lower rational cut, upper rational cut, locatedness modulus, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, dedekindcut, located-regseqrat: no listed sibling is bijective to $\mathsf{LocatedRegularRealCut}^{\uparrow}$ because this packet alone binds lower and upper cut rows, a locatedness modulus, dyadic tolerance, regular readback, terminal real seal, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet is generated from lower rational cut rows, upper rational cut rows, a locatedness modulus, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, seed)

Scope closed: Definition 4219.1, Definition 4219.2, and Definition 4219.3 seed the finite bridge from regular-Cauchy real names to located lower–upper cut boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located lower and upper stability, regular Cauchy handoff, Real seal non-escape, and ledger exactness for BEDC.Derived.LocatedRegularRealCutUp.

Chapter 3191

A Concrete Naming Certificate for $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$

The $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet realizes the orbit and codon inventory polynomials of Definition 9462.1713 as finite BEDC NameCert data. It consumes the boundary-tile source of chapter 3393, the coverage packet of chapter 3470, the codon window source of chapter 2234, and the finite polynomial carrier of chapter 50. The packet records orbit rows, codon-count rows, coefficient ledgers, and marginal projections; it is not a biological stochastic model, an empirical code table, a hidden enumeration oracle, or quotient codon semantics.

Definition 3191.1 (Boundary orbit inventory polynomial carrier)**.** *A* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, K, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the orbit table row, recording the finite orbit types classified by support size m, overlap defect δ, lift imbalance b, and orbit size.*
2. *K is the codon count table row, recording the weighted counts of lifted triggers over the same triples (m, δ, b).*
3. *E is the coefficient ledger row, storing the displayed coefficients of the orbit inventory polynomial $\mathcal{O}(u, v, w)$ and the codon inventory polynomial $\mathcal{C}(u, v, w)$.*
4. *M is the marginal projection row, recording the one-variable and bivariate specializations read from the coefficient ledger.*
5. *H records componentwise* hsame *transport for the orbit table, codon count table, coefficient ledger, marginal projections, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an inventory query through O, K, E, M, H.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *the source vision of Definition 9462.1713,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *row.*

The carrier accepts coefficient and marginal reads only through O, K, E, M, H, C, P, N. It has no coordinate for a hidden biological code table, probabilistic fitness model, unbounded orbit enumeration, quotient codon equality, or unstated polynomial coefficient source.

Lean4 ✓ `BEDC.Derived.BoundaryOrbitInventoryPolynomialUp.BoundaryOrbitInventoryPolynomialTasteGate`
`single carrier alignment`

Theorem 3191.2 (Boundary orbit inventory polynomial obligations)**.** *For every accepted* BoundaryOrbitInventoryP *carrier $I = (O, K, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *surface consists exactly of orbit-table admission through O, codon-count table admission through K, coefficient-ledger exactness through E, marginal-projection exactness through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4220.1. The orbit table and codon count table are the only source rows, while the coefficient ledger and marginal projection rows are the only polynomial-facing rows.

Coefficient exactness is read from E. Each displayed coefficient is determined by grouping the rows of O and weighting them by the count row K; no coefficient row is available without those two source tables.

Marginal exactness is read from M. A marginal projection is permitted only as a specialization of the coefficient ledger, so it cannot replace the orbit table or codon count table by a hidden enumeration.

The structural rows H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier lacks biological stochastic, quotient-codon, hidden-table, and unbounded-enumeration coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3191.3 (Boundary orbit inventory coefficient determinacy)**.** *Every coefficient marginal exported by an accepted* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *packet is determined by the displayed orbit table O, codon count table K, coefficient ledger E, and marginal projection row M. Removing any coefficient row from E leaves a marginal query without a displayed source, and removing any orbit or codon count row leaves the corresponding coefficient without a displayed inventory witness.*

Proof. Read the NameCert obligations of Definition 4220.2. A coefficient query first exposes the orbit table O and codon count table K, then reads the coefficient ledger E.

The marginal projection row M is downstream of the ledger. It can specialize the displayed polynomial, but it cannot introduce a coefficient absent from E or reconstruct a missing source row absent from O or K.

Transport and replay preserve the same dependency order. Therefore a coefficient marginal is determined exactly by the displayed table and ledger rows, and deletion of any required row breaks the finite NameCert route. □

Falsifiable prediction: For $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, coefficient marginals are determined by the displayed orbit table, codon count table, coefficient ledger, and marginal projection rows, and a marginal read fails if any required coefficient row is removed.

Independence witness: partition, hash, errorcode, dna-window-six-boundary-tile, six-boundary-tile-coverage: the polynomial coefficient ledger is not bijective to these sibling carriers because it binds orbit tables, codon count tables, coefficient rows, marginal projections, transport, replay, provenance, and local naming, while the siblings supply partitioning, digest, error coding, boundary-tile, or coverage surfaces without this coefficient inventory.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet is generated from an orbit table, codon count table, coefficient ledger, marginal projection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4220.1, Definition 4220.2, and Definition 4220.3 seed the finite orbit-inventory polynomial certificate over boundary-tile and coverage rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological stochastic modeling, empirical code-table semantics, quotient codon equality, unbounded orbit enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-table admission, codon-count table admission, coefficient-ledger exactness, marginal-projection exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryOrbitInventoryPol

Chapter 3192

A Concrete Naming Certificate for $\mathsf{PrimitiveRecursive}^{\uparrow}$

Following Kleene's primitive recursive functions, the $\mathsf{PrimitiveRecursive}^{\uparrow}$ packet records finite recursion syntax and its local replay rows as BEDC data. It sits after the unary natural-number carrier of chapter 28, the list spine of chapter 36, the finite computability surface of chapter 199, the stream-window discipline of chapter 126, and the regular rational readback route of chapter 301. The chapter supplies a finite schedule substrate for later modulus and real-name consumers; it does not introduce minimization, partial recursive search, host functions, or unbounded choice.

Definition 3192.1 (PrimitiveRecursive carrier)**.** *A* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Z, S, \Pi, K, R, H, C, Q, N)$$

whose rows are read as follows:

1. *Z is the zero-function row.*
2. *S is the successor-function row.*
3. *Π is the finite family of projection rows, each with displayed arity and coordinate index.*
4. *K is the composition row, storing one outer code, its finite list of inner codes, and the arity ledger that makes the composition well typed.*
5. *R is the primitive-recursion scheme row, storing the base code, step code, displayed accumulator coordinate, and finite recursion spine over the* $\mathsf{Nat}^{\uparrow}$ *row.*
6. *H is componentwise* hsame *transport for zero, successor, projection, composition, recursion, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay for evaluating a finite syntax tree by structural recursion through the rows Z, S, Π, K, R.*
8. *Q is the* Pkg *provenance row over* $\mathsf{PrimitiveRecursive}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, S, \Pi, K, R, H, C, Q, N$. There is no coordinate for minimization, unbounded search, partiality, a host function graph, countable choice, or an external evaluator.

Lean4 ✓ `BEDC.Derived.PrimitiveRecursiveUp.TasteGate.PrimitiveRecursiveTasteGate single carrier alignment`

Theorem 3192.2 (PrimitiveRecursive NameCert obligations)**.** *For every accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier*

$$P = (Z, S, \Pi, K, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *surface consists exactly of carrier admission for the zero, successor, projection, composition, primitive-recursion, transport, replay, provenance, and naming rows; arity compatibility for* Π, K, R*; structural replay of the displayed syntax tree through* C*; componentwise transport through* H*; and provenance through* Q, N*. No obligation is stored in a minimization row, partial recursive search row, host-function graph, or unbounded-choice selector.*

Proof. Inspect the carrier row by row. The base constructors are Z, S, Π, and their arity information is part of the displayed finite rows rather than a host-side function space.

For composite syntax, the row K supplies the finite list of inner codes and the outer code, while R supplies the base and step codes for the primitive-recursion scheme. Induction on the displayed syntax tree checks that every replay request descends to one of Z, S, Π or to a strictly displayed subcode of K or R.

The structural rows close the local certificate. Transport through H preserves the same constructor rows, replay through C evaluates only the displayed syntax tree, and Q, N record provenance and naming. Since the packet has no coordinate for minimization, partial recursive search, host functions, or unbounded choice, the listed rows exhaust the NameCert obligations. □

Theorem 3192.3 (PrimitiveRecursive composition closure)**.** *If an accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *packet contains a composition row* K *whose outer code and finite list of inner codes are accepted by the same carrier, then the composed code is again read as a* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *row through* K, H, C, Q, N*. The closure is internal to the finite displayed syntax and exports no minimization, partiality, host-function graph, or unbounded-choice principle.*

Proof. Proceed by induction on the finite list of inner codes recorded in K. The empty-list boundary is the displayed arity check for the outer code, and the single-step case appends one already accepted inner row to the composition ledger.

At each induction step the replay row C evaluates the inner rows before the outer row, and transport through H preserves the same arity and constructor classifiers. The provenance row Q and local naming row N then repackage the resulting finite syntax as a $\mathsf{PrimitiveRecursive}^{\uparrow}$ row.

No unbounded search is used in the induction. The construction consumes only the displayed finite list in K and the accepted subcodes already present in the carrier, so it cannot introduce minimization, partial recursive search, host functions, or unbounded choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4221.1, Definition 4221.2, and Definition 4221.3 seed the finite recursion-syntax certificate for schedule and modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimization, partial recursive search, host functions, unbounded choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, arity compatibility, composition closure, primitive-recursion replay, transport, provenance, and local naming for the same finite recursion syntax.

Constructive story: The construction starts from zero, successor, and projection rows, closes finite composition through a displayed list spine, and closes primitive recursion by replaying the base and step codes over the Nat row under componentwise transport and local naming.

The $\mathsf{CauchyAcceleration}^{\uparrow}$ packet reads the StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces of chapter 126, chapter 291, chapter 301, and chapter 38 before exporting its accelerated completion-facing route.

Chapter 3193

A Concrete Naming Certificate for $\mathsf{CauchyAcceleration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyAcceleration}^{\uparrow}$ packet records the finite acceleration step by which an ordinary displayed Cauchy name is replaced by a cofinal regular rational name without changing the represented real. It sits after the criterion surfaces of chapter 392 and chapter 3576, the regularization route of chapter 376, the windowed name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The acceleration packet consumes Cauchy tail comparison as substantive finite-window and dyadic-tolerance data through chapter 301, chapter 126, chapter 291, and chapter 38 before any accelerated Real-facing read is exported. The accelerated cofinal regular-rational name is consumed by chapter 5721 as the completion-existence witness route, with the read still passing through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet makes acceleration a named finite ledger: it does not select an ambient subsequence, quotient a Cauchy representative, or appeal to host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3193.1 (Cauchy acceleration carrier)**.** *A* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite windows display the original rational Cauchy name.*
2. *T is the dyadic tolerance schedule, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, assigning the requested precision budget to each inspected window.*
3. *α is the cofinal acceleration map, stored as a finite monotone ledger saying which source window is read for each accelerated precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *proof row for the accelerated rational name, including the regularity inequalities obtained by reading S through α and T.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the accelerated regular name reaches the same Real-facing boundary as the source name.*
6. *H records componentwise* hsame *transport for the source stream, tolerance schedule, cofinality map, regular proof row, Real seal, provenance, and local naming row.*

7. *C records displayed* Cont *replay from an acceleration request through S, T, α, R, E and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyAcceleration}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient representative of the original Cauchy name, a choice-selected subsequence, an ambient completeness theorem, or a host-side equality oracle for real numbers.

Theorem 3193.2 (Cauchy acceleration NameCert obligations)**.** *For every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row S, dyadic tolerance schedule T, cofinal map α, regular rational proof row R, and Real seal row E;*
2. *monotone cofinality, requiring α to preserve request order and to eventually reach every inspected source precision window;*
3. *dyadic tolerance preservation, requiring each accelerated request to read the same displayed tolerance budget through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
4. *classifier transport, requiring H and C to preserve the source stream, tolerance schedule, cofinality ledger, regular proof row, and Real seal under componentwise* hsame *and displayed replay;*
5. *ledger exhaustion through P and N, excluding quotient representatives, choice-selected subsequences, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and host real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4222.1. The accepted packet exposes only the source stream S, tolerance schedule T, acceleration ledger α, regular readback R, Real seal E, and the structural rows.

Monotone cofinality is the public content of α. Since α is stored as a finite ledger over the inspected request window, each accelerated precision read names the source window it consumes and records that later requests do not move backward.

Dyadic tolerance preservation follows by reading T before the regular rational proof row. The $\mathsf{RegSeqRat}^{\uparrow}$ row R receives exactly the tolerance entries displayed by $\mathsf{DyadicRatCore}^{\uparrow}$, so the acceleration cannot import an unrecorded modulus or external precision oracle.

Classifier transport is componentwise through H and replayed through C. These rows preserve S, T, α, R, E without adding a second classifier surface.

Finally, P and N name precisely the displayed packet. Because the carrier contains no quotient representative, choice-selected subsequence, ambient completeness coordinate, or host real equality oracle, the five listed rows exhaust the local naming obligations. □

Theorem 3193.3 (Cauchy acceleration L10 handoff)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$ exports only the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $S, T, \alpha, R, E, H, C, P, N$. Consequently every L10 consumer receives a $\mathsf{RegSeqRat}^{\uparrow}$*-compatible and* $\mathsf{Real}^{\uparrow}$*-compatible accelerated name, while quotient representatives of the original Cauchy name, choice-selected subsequences, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness remain unavailable.*

Proof. Project the accepted carrier and apply Definition 4222.2. The only public data are $S, T, \alpha, R, E, H, C, P, N$.

The source read begins at the $\mathsf{StreamName}^{\uparrow}$ row S. The tolerance read then passes through T and α, so every accelerated precision request is attached to an explicit dyadic budget and a displayed source window.

The regular rational handoff is exactly R, and the Real-facing boundary is exactly E. Transport and replay through H and C preserve that route for downstream consumers.

Since P and N exhaust the packet and no forbidden coordinate appears in Definition 4222.1, the exported object is only the displayed accelerated name compatible with $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 3193.4 (Cauchy acceleration rectangular window route)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier* $A = (S, T, \alpha, R, E, H, C, P, N)$ *reuses the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows when producing the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the terminal* $\mathsf{Real}^{\uparrow}$ *handoff. The cofinal ledger* α *may refine the order of precision requests, but it cannot replace the displayed window route recorded by* S, T, H, C, P, N*.*

Proof. Project the rows $S, T, \alpha, R, E, H, C, P, N$. The source windows are recorded in S, and the dyadic tolerance entries are recorded in T before the accelerated regular rational row R or the Real-facing seal E is consumed.

Apply Definition 4222.2. The cofinality row α is an order ledger over displayed source windows, while dyadic tolerance preservation forces each accelerated request to read the same T row before it reaches R.

Now apply Definition 4222.3. Transport and replay through H and C preserve the route named by P and N, so no quotient representative, selected subsequence outside the ledger, or ambient completeness row is exported. □

Theorem 3193.5 (Cauchy acceleration cofinal-ledger determinism)**.** *Let*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

and

$$A' = (S, T, \alpha', R', E', H, C, P, N)$$

be accepted $\mathsf{CauchyAcceleration}^{\uparrow}$ *carriers sharing the same displayed source stream, tolerance schedule, structural replay rows, provenance rows, and local naming boundary through* S, T, H, C, P, N*. If the two cofinal ledgers* α *and* α' *choose the same source window for every inspected precision request, then their accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal follow the same finite route. In particular, neither ledger can replace the displayed source-window or tolerance route before* R, R' *and* E, E' *are read.*

Proof. Project both accepted carriers through Definition 4222.1. The shared rows expose the same $\mathsf{StreamName}^{\uparrow}$ source windows, the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule, and the same replay and provenance boundary.

Apply Definition 4222.2 to both packets. The only ledger freedom available before the accelerated readback is the cofinal row, and by hypothesis the displayed source window selected for each inspected request is identical in α and α'.

Now use Definition 4222.4. Since the rectangular source-window and tolerance route is fixed, the regular rational rows R, R' and the Real seals E, E' are reached through the same finite route. The packet therefore has no row through which a second ledger can substitute a hidden subsequence, quotient representative, or ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyAcceleration}^{\uparrow}$ packet is generated from a StreamName source row, a DyadicRatCore tolerance schedule, a finite cofinal acceleration ledger, a RegSeqRat proof row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4222.1, Definition 4222.2, and Definition 4222.3 seed the finite acceleration route from StreamName windows through dyadic tolerance schedules, cofinal acceleration ledgers, RegSeqRat readback, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy-name equality, choice-selected subsequences, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone cofinality, dyadic tolerance preservation, classifier transport, L10 handoff, and ledger exhaustion for BEDC.Derived.CauchyAccelerationUp.

Theorem 3193.6 (Cauchy acceleration sequence-space route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer reads the common-window classifier of* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *from chapter 406 before exporting the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff. The route passes through the regularization boundary of chapter 376: the source* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accelerated cofinal ledger, regular rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows are visible before any Real-facing consumer may inspect the acceleration result. No quotient representative, selected subsequence outside the displayed ledger, host equality, or ambient completeness row is exported.*

Theorem 3193.7 (Cauchy subsequence dependency route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal must first consume the rate-dominance surface of chapter 1814 and the fast-modulus surface of chapter 1781. The route is finite:* $\mathsf{CauchyRateDominance}^{\uparrow}$ *bounds the source schedule,* $\mathsf{FastCauchyModulus}^{\uparrow}$ *supplies the inspected acceleration window, and the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local* NameCert *rows carry the resulting readback. No consumer may replace this route by a chosen subsequence, quotient representative, ambient completion row, or host equality of Real names.*

Proof. Begin with the accepted carrier of Definition 4222.1. Its source stream row S, tolerance schedule T, cofinal ledger α, regular rational row R, and Real seal E are the only analytic rows available before transport, replay, provenance, and local naming.

The rate-dominance chapter supplies the comparison row that bounds which source schedule entries may be read by α. The fast-modulus chapter supplies the inspected window that turns those bounded reads into the accelerated tolerance route. Reading either row after the $\mathsf{RegSeqRat}^{\uparrow}$ readback would hide the schedule obligation rather than consume it.

Apply the rectangular window route of Definition 4222.4. It keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows in front of R and E, so the accelerated readback remains a displayed finite route. The carrier has no coordinate for a selected subsequence, quotient representative, ambient completion proof, or host Real equality, and the dependency route cannot export one. □

Theorem 3193.8 (Cauchy acceleration L10 four-face cofinality)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal preserves the ordered L10 four-face route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The cofinal acceleration ledger may refine which displayed source window is read for a requested precision, but it cannot commute, omit, or replace the $\mathsf{StreamName}^{\uparrow}$ *source face, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The resulting route remains finite and displayed, with no chosen subsequence outside the ledger, quotient representative, ambient completion row, host real equality, countable-choice schedule, Lean verification, or standard bridge.*

Proof. Start with Definition 4222.3. It names the only exported analytic route of an accepted carrier as the ordered sequence from the stream window through the dyadic tolerance ledger and regular rational readback to the Real-facing seal.

Next apply Definition 4222.4. The cofinal ledger α refines precision requests inside the displayed rectangle of source windows and dyadic tolerances, so it cannot move the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ rows ahead of the upstream faces.

Finally use the sibling dependency route of Definition 4222.7. The rate-dominance and fast-modulus surfaces are consumed before the accelerated readback is exported, and they leave the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces in order.

The carrier and dependency route have no row for a hidden subsequence, quotient representative, ambient completion proof, host Real equality, choice schedule, Lean verification, or bridge credential. Hence cofinal acceleration preserves exactly the displayed L10 four-face route. □

Theorem 3193.9 (Cauchy acceleration completion forward link)**.** *Every completion-facing consumer of* $\mathsf{CauchyAcceleration}^{\uparrow}$ *reads through chapter 1639, chapter 301, chapter 126, chapter 291, and chapter 38 before the accelerated route exports a terminal real boundary. The dependency is the finite route from source* $\mathsf{StreamName}^{\uparrow}$ *windows through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule and cofinal acceleration ledger into* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the Bishop regular Cauchy completion surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the displayed source windows. The accelerated route has no completion-facing input until the $\mathsf{StreamName}^{\uparrow}$ inspection windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule have been exposed.

Next read the regular rational and completion surfaces. The cofinal acceleration ledger feeds $\mathsf{RegSeqRat}^{\uparrow}$ readback and then passes through the Bishop regular Cauchy completion chapter before any terminal real boundary is visible.

Finally attach the $\mathsf{Real}^{\uparrow}$ seal. Since the route is finite and displayed at each sibling boundary, it exports no quotient completion, selected subsequence, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Chapter 3194

A Concrete Naming Certificate for $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$

Following Brouwer and Bishop–Bridges on monotone sequences in constructive analysis, the BrouwerMonotoneSeque packet records a monotone real-name boundary object through finite stream windows, regular rational readback, dyadic tolerance ledgers, and the terminal real-name seal. It sits after the Specker obstruction of chapter 1406, the regular-rational readback of chapter 301, the stream-window interface of chapter 126, the dyadic rational core of chapter 291, and the Real seal of chapter 38. The chapter gives the L10 Real and Completion frontier a named monotone-sequence carrier without claiming located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3194.1 (BrouwerMonotoneSequence carrier)**.** *A* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row, listing the inspected indices and their finite dependency order.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the inspected real-name entries.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed windows and monotonicity comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been displayed.*
6. *M is the monotonicity ledger, recording only finite adjacent-window comparison obligations among the displayed entries.*
7. *H records componentwise* hsame *transport for source indices, windows, dyadic ledgers, readback, Real seals, monotonicity, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a monotone-sequence request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets whose monotone-sequence request displays these rows in order. It has no coordinate for a located supremum, trichotomy decision, countable-choice selector, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BrouwerMonotoneSequenceUp.BrouwerMonotoneSequenceTasteGate single carrier alignment`

Theorem 3194.2 (BrouwerMonotoneSequence NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerMonotoneSeque}$ *carrier*

$$B = (S, W, D, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the inspected source sequence, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, finite monotonicity ledger, componentwise transport, displayed replay, package provenance, and local naming rows. No NameCert obligation may be discharged by a located supremum, trichotomy of real names, countable-choice schedule, selected limit, quotient stream representative, or ambient complete real line.*

Proof. Project the carrier of Definition 4223.1. The analytic rows are precisely the source schedule S, the finite windows W, the dyadic tolerance ledger D, the regular rational readback R, the Real seal E, and the finite monotonicity ledger M.

The monotonicity information is local to the displayed windows. Each comparison read in M consumes only the adjacent source entries already opened through S, W, D, R and does not add a global order decision for arbitrary real names.

The structural rows add no hidden analytic principle. Transport through H preserves the same displayed rows, replay through C follows the same finite route, and P, N record provenance and local naming for the packet.

Since the carrier has no row for located suprema, trichotomy, countable-choice schedules, selected limits, quotient stream representatives, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3194.3 (BrouwerMonotoneSequence finite-window nonescape)**.** *Every consumer read of an accepted* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier reaches a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may consult the finite monotonicity ledger M for displayed adjacent windows, but it cannot escape to a located supremum, countable-choice selector, trichotomy oracle, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer read of the accepted carrier. The source row S exposes only the inspected indices, so the first real-name data available to the consumer are the finite $\mathsf{StreamName}^{\uparrow}$ windows in W.

The dyadic and regular-rational rows determine the only readback route. The tolerance ledger D bounds the displayed windows, and the row R packages those bounded observations as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The terminal seal is downstream of those rows. The $\mathsf{Real}^{\uparrow}$ row E is available only after S, W, D, R have been displayed, and the monotonicity ledger M can compare only the adjacent windows already present in the same finite packet.

Transport, replay, provenance, and local naming preserve this order through H, C, P, N. Therefore every consumer read remains inside the displayed finite-window route and cannot import a

located supremum, trichotomy oracle, choice schedule, selected limit, quotient stream representative, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ packet is generated from inspected source indices, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, a finite monotonicity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4223.1, Definition 4223.2, and Definition 4223.3 seed the constructive monotone-sequence boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window monotonicity, StreamName exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerMonotoneSequenceUp.

Chapter 3195

A Concrete Naming Certificate for $\mathsf{CauchySequenceCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet records the textbook sequence criterion as finite BEDC data. It ties a sequence source to the dyadic tolerance discipline of chapter 291, the stream-window naming discipline of chapter 126, the regular rational readback of chapter 301, and the public real seal of chapter 38. The chapter names only the finite route from tolerance data to regular-Cauchy and real consumers; it does not supply a choice-selected subsequence, quotient equality of limits, host completion, or ambient completeness principle.

Definition 3195.1 (CauchySequenceCriterion carrier)**.** *A* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the sequence-source row, displaying the named sequence and the finite request interface through which a downstream consumer asks for terms.*
2. *M is the modulus-schedule row, displaying the finite schedule that turns a requested dyadic tolerance into a stage bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row, recording the positive dyadic thresholds, comparisons, and tolerance refinements read by the schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the sequence after the schedule row has supplied their stage bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row reached from A, M, D, S.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-Cauchy readback has been displayed.*
7. *H records componentwise* hsame *transport for the sequence source, schedule, dyadic tolerance ledger, stream windows, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a criterion request through $A, M, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose criterion consumer first displays the sequence source, modulus schedule, dyadic tolerance ledger, stream-window row, regular-Cauchy readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCriterionUp.CauchySequenceCriterionTasteGate single carrier alignment`

Theorem 3195.2 (CauchySequenceCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier*

$$Q = (A, M, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *surface consists exactly of sequence-source admission, modulus-schedule admission, dyadic tolerance-ledger exposure, finite stream-window exposure, regular-Cauchy readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of* Q*. No obligation may be discharged by a choice-selected subsequence, quotient equality of limits, host completion, ambient completeness, or an unbounded open-cover argument.*

Proof. Project the carrier of Definition 4224.1. Its analytic rows are precisely the sequence source A, modulus schedule M, dyadic tolerance ledger D, stream windows S, regular-Cauchy readback R, and Real seal E.

The tolerance rows determine the access order. A criterion consumer reads M before it can turn a dyadic request from D into a finite stage bound, and it reads S only through the finite windows exposed by that bound.

The readback rows add no external completion object. The row R repacks the displayed tolerance and window data as regular-Cauchy readback, and E seals that readback as the public real-facing output.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate of Q stores a selected subsequence, quotient limit equality, host completion, ambient completeness witness, or open-cover argument, the displayed rows exhaust the NameCert obligations. □

Theorem 3195.3 (CauchySequenceCriterion L10 handoff)**.** *Every L10 consumer read of an accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through the sequence source, modulus schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the sequence source A and modulus schedule M. They expose the named sequence and the finite stage bounds required by any dyadic tolerance request.

Next read the dyadic tolerance ledger D and stream-window row S. The ledger supplies the positive thresholds and comparisons, while the stream row supplies only the finite windows named by the schedule.

Attach the regular-Cauchy readback row R. It is the unique displayed route by which the dyadic-window data becomes a regular rational sequence consumer.

The Real seal closes the route. The row E is reached only after R, and the structural rows H, C, P, N carry the same finite data through transport, replay, provenance, and local naming. Hence the L10 handoff exposes a constructive sequence criterion rather than a selected subsequence, quotient limit, host completion, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet is generated from a sequence source, modulus schedule, dyadic tolerance ledger, stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4224.1, Definition 4224.2, and Definition 4224.3 seed the L10 route from dyadic sequence criteria to regular-Cauchy and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected subsequences, quotient equality of limits, host completion, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus schedule exposure, dyadic tolerance-ledger exactness, stream-window handoff, RegSeqRat/Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCriterionUp.

$\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ routes pointwise locator induction through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38; the child file carries the theorem and closurestatus surface.

Chapter 3196

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 3196.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3196.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 3196.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 3196.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3196.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 3196.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 3196.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3197

A Concrete Naming Certificate for $\mathsf{FrinkMetrization}^{\uparrow}$

Following Frink and Kelley, the $\mathsf{FrinkMetrization}^{\uparrow}$ packet records the finite BEDC route from a uniform entourage basis to a pseudometric-facing and metric-facing distance certificate. It sits after the uniform-source rows of chapter 339, the pseudometric carrier of chapter 1614, the separated metric boundary of chapter 57, the topology compatibility row of chapter 88, the dyadic tolerance ledger of chapter 291, and the finite window interface of chapter 126. The packet names only the constructive handoff from nested entourage data to displayed distance reads; it does not form quotient uniformities, select representatives, assert ambient metrizability, or import choice.

Definition 3197.1 (FrinkMetrization carrier)**.** *A* $\mathsf{FrinkMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, Q, D, P, M, T, H, C, G, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row. It names the carrier surface and the finite entourage requests from which the metrization route starts.*
2. *Q is the nested entourage chain. It records the finite refinement ladder used to compare two source points at dyadic scales.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. It translates the refinement level into displayed dyadic radii and triangle budgets.*
4. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *handoff row. It receives the finite entourage chain and exports the displayed pre-separated distance route.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *compatibility row, available only after the separated boundary has accepted the zero-distance route supplied through P.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *compatibility row. It records that the metric-facing balls and the original uniform entourage reads are compared through the same finite windows.*
7. *H records componentwise* hsame *transport for the uniform source, chain, dyadic ledger, pseudometric handoff, metric compatibility, topology compatibility, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform entourage request through the nested chain, dyadic ledger, pseudometric handoff, metric row, topology row, provenance, and local naming row.*

9. *G is the* Pkg *provenance over* FrinkMetrization$^{\uparrow}$, UniformSpace$^{\uparrow}$, Pseudometric$^{\uparrow}$, Met$^{\uparrow}$, Topology$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, or an unbounded distance oracle.

Lean4 ✓ `BEDC.Derived.FrinkMetrizationUp.FrinkMetrizationTasteGate single carrier alignment`

Theorem 3197.2 (FrinkMetrization NameCert obligations)**.** *For every accepted* FrinkMetrization$^{\uparrow}$ *carrier $F = (U, Q, D, P, M, T, H, C, G, N)$, the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform source, nested entourage chain, dyadic radius ledger, pseudometric handoff, metric compatibility row, topology compatibility row, transport, continuation, provenance, and local naming row;*
2. *nested-chain readability, requiring every distance request to start at the* UniformSpace$^{\uparrow}$ *row U and pass through the finite refinement chain Q before any dyadic radius in D is read;*
3. *pseudometric handoff, requiring the displayed distance route to be exported through P before the metric row M or topology row T can be consumed;*
4. *compatibility transport, requiring H and C to preserve the same finite* StreamName$^{\uparrow}$ *windows and* DyadicRatCore$^{\uparrow}$ *radius ledger across uniform, pseudometric, metric, and topology-facing consumers;*
5. *provenance and refusal, requiring G and N to name only this finite handoff and to refuse quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, and unbounded distance oracles.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4226.1. The projection exposes exactly the uniform source, nested chain, dyadic ledger, pseudometric handoff, metric and topology compatibility rows, and the structural rows H, C, G, N.

The nested-chain obligation is forced by row order. A distance request starts at the uniform row U, enters the refinement ladder Q, and only then reads the dyadic radius ledger D.

The pseudometric handoff is the only distance-export step. Row P supplies the pre-separated displayed distance route, while M and T can compare it with metric and topology-facing consumers only after the same finite windows and radii have been replayed.

Rows H, C, G, N transport, replay, source, and name this route. Since the tuple contains no quotient uniformity, selected representative, ambient metrizability, countable choice, host topology equality, or unbounded distance coordinate, the local NameCert cannot export those principles. □

Closure status for NameCert$_{\text{FrinkMetrization}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, seed)

Scope closed: Definition 4226.1 and Definition 4226.2 seed the finite Frink-style handoff from uniform entourage chains to pseudometric, metric, topology, dyadic, and stream-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, unbounded distance oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-chain readability, dyadic radius exactness, pseudometric handoff, metric and topology compatibility, transport, continuation, provenance, and ledger refusal for BEDC.Derived.FrinkMetrizationUp.

Constructive story: A FrinkMetrization$^{\uparrow}$ packet is generated from a UniformSpace source row, a nested entourage chain, a DyadicRatCore radius ledger, a Pseudometric handoff, Metric and Topology compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

NestedShrinkInterval$^{\uparrow}$ reads its compact interval dependency through chapter 5553 and the child `namecert_construction.tex` route before exposing its shrinking-diameter Real$^{\uparrow}$ seal.

Chapter 3198

A Concrete Naming Certificate for $\mathsf{NestedShrinkInterval}^{\uparrow}$

Following Cantor's nested-interval argument and Bishop–Bridges regular real analysis, the $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet records the finite BEDC carrier surface for a descending located interval route whose displayed diameters shrink to the requested tolerance. It sits between the nested-closed interval carrier of **??**, the located endpoint row of chapter 394, the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names only the finite shrinking-interval convergence certificate; it does not claim open-cover compactness, trichotomy, quotient reals, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3198.1 (Nested shrink interval carrier)**.** *A* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (J, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row, exposing the retained descending closed located interval family and its containment witnesses.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row for the inspected interval windows, including lower and upper endpoint location data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger recording endpoint comparisons, diameter estimates, and tolerance requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect interval endpoints and diameter rows at displayed stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same stream windows and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the nested interval, located endpoint, dyadic shrink, stream-window, and regular readback rows are visible.*
7. *H records componentwise* hsame *transport for J, L, D, S, R, E and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$J, L \longmapsto D, S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{NestedShrinkInterval}^{\uparrow}$, $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $J, L, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient-real equality, a selected intersection point, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3198.2 (Nested shrink interval NameCert obligations)**.** *For every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier*

$$K = (J, L, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row,* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same displayed rows; diameter-to-tolerance exactness through D; finite-window readback through S and R; Real-seal handoff through E; and ledger nonescape for the excluded compactness, trichotomy, quotient, selection, choice, and ambient-completeness coordinates.*

Proof. Project the carrier of Definition 4227.1. The projection exposes exactly the nested closed interval source, located endpoint row, dyadic shrink ledger, finite windows, regular readback, real seal, and structural rows.

Classifier stability is componentwise on those displayed rows. The shrink estimate is not an external metric fact: it is the D row, read together with the endpoint row L and the stream schedule S before regular readback is available.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, every accepted NameCert obligation is one of the rows listed above. □

Theorem 3198.3 (Nested shrink interval diameter route)**.** *Let $K = (J, L, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. Every completion-facing request for the shrinking interval route first reads the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source J and* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row L, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger D and* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, and only then opens the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and* $\mathsf{Real}^{\uparrow}$ *seal E. The route exports diameter-to-tolerance convergence as displayed finite ledger data, not as open-cover compactness, trichotomy, quotient-real equality, countable choice, a selected intersection point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing request and project the accepted carrier. The only interval source is J, and the only endpoint data are the located rows in L.

The diameter request is then forced through D. Its dyadic comparisons and tolerance rows are inspected through the finite schedule S, so a consumer cannot replace the shrinking route by an unrecorded compactness or completeness theorem.

After the finite windows are displayed, R packages the same observations as regular rational readback and E supplies the terminal real seal. The remaining rows H, C, P, N preserve the route and add no excluded coordinate. □

Theorem 3198.4 (Nested shrink interval Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet receives the seal only through the displayed route*

$$\mathsf{NestedClosedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff is stable under the componentwise hsame *transport and displayed* Cont *replay rows* H, C*, and it is sourced and named only by* P, N*.*

Proof. Apply Definition 4227.3. A Real-facing consumer reaches E only after the nested interval source, located endpoint row, dyadic shrink ledger, stream windows, and regular readback have all been displayed.

Transport through H preserves the row roles, and replay through C preserves their order. Provenance and local naming through P, N attach the handoff to the packet without adding analytic content.

Thus the Real seal cannot escape to a selected point, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient real completeness. □

Theorem 3198.5 (Nested shrink interval regular-Cauchy uniqueness route)**.** *Let* $K = (J, L, D, S, R, E, H, C, P, N)$ *and* $K' = (J', L', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packets. Suppose their nested-source rows, located endpoint rows, dyadic shrink ledgers, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise related by the displayed* hsame *transport. Then the two packets reach the same* $\mathsf{Real}^{\uparrow}$*-facing seal route only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *uniqueness surface displayed by* R, R' *before Real sealing. The route does not identify selected intersection points, quotient real representatives, or ambient host real equality.*

Proof. Project both carriers. The first carrier exposes $J, L, D, S, R, E, H, C, P, N$, and the second exposes $J', L', D', S', R', E', H', C', P', N'$.

Apply the diameter route of Definition 4227.3 to both packets. Each read is forced through its nested source, located endpoints, dyadic shrink ledger, finite stream windows, and regular rational readback before a Real-facing seal is visible.

Use the assumed componentwise hsame transport on J, L, D, S, R and J', L', D', S', R'. The endpoint comparison therefore lives on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, because the preceding rows have already fixed the same nested-source, located-endpoint, dyadic, and stream-window route.

Finally apply the Real-seal handoff of Definition 4227.4 to both packets. It permits E and E' to be read only after the regular readback rows are displayed, so the uniqueness comparison is a $\mathsf{RegSeqRat}^{\uparrow}$ route before Real sealing and not a selected intersection point, quotient real equality, or ambient completeness assertion. □

Theorem 3198.6 (Cantor intersection nested-shrink route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer that reads a nested shrinking interval first consumes the* $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows exposed by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. The downstream route supplies the nested-shrink boundary used by chapter 6999 and the Cantor-intersection surface of chapter 1421.*

The route is a forward lattice edge only. It does not select an intersection point, assert open-cover compactness, use countable choice, identify quotient real names, or provide ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4227.1. The exposed rows are the nested closed source J, located endpoints L, dyadic shrink ledger D, finite stream windows S, regular readback R, real seal E, and the structural rows H, C, P, N.

Apply Definition 4227.4. Any Real-facing downstream read reaches E only after the nested, located, dyadic, stream, and regular-rational rows have been displayed.

The Cantor diagonal and Cantor-intersection chapters consume exactly this kind of finite shrinking-interval boundary before their own diagonal or intersection rows are read. Since the

carrier contains no selected point, open-cover compactness row, choice schedule, quotient equality, or ambient completeness coordinate, the downstream route remains a finite dependency edge. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet is generated from a NestedClosedInterval source row, LocatedInterval endpoint rows, a DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4227.1, Definition 4227.2, Definition 4227.3, and Definition 4227.4 seed the finite L10 shrinking-interval convergence route through nested closed intervals, located endpoint rows, dyadic diameter ledgers, stream windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient reals, selected intersection points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, diameter-to-tolerance exactness, finite-window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedShrinkIntervalUp.

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet consumes the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 before a completion-facing reader treats shrinking finite windows as a diagonal Cauchy subsequence witness. The nested source, located endpoint, dyadic diameter, and $\mathsf{StreamName}^{\uparrow}$ rows still supply the finite shrinking windows; $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies only the displayed diagonal handoff needed to pass those windows toward $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The route exports no detached selector, quotient-real equality, countable-choice schedule, selected intersection point, or ambient completeness coordinate. The shrinking interval diameter route is consumed through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38. Thus a downstream L10 consumer reads $\mathsf{NestedShrinkInterval}^{\uparrow}$ only after the finite diameter ledger has been repacked as a $\mathsf{RegSeqRat}^{\uparrow}$ route and sealed as a $\mathsf{Real}^{\uparrow}$ boundary, not as an ambient completeness or selected intersection-point claim. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ after the shrinking-diameter ledger has been repacked through the $\mathsf{StreamName}^{\uparrow}$ observation schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal route. This sibling dependency records the completion-criterion consumer path of chapter 1476: a nested shrink interval can be read as completion evidence only after its finite diameter data has passed through the same stream, regular sequence, dyadic tolerance, and Real seal rows.

Theorem 3198.7 (Nested shrink interval Cauchy completion consumer)**.** *Every completion-facing read of* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *consumes* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *only after the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are visible. Thus the dependency on chapter 1476 is a theorem-level consumer edge through chapter 126 and chapter 301, not an appeal to an undisplayed completion object.*

Proof. Read a completion-facing request for a nested shrink interval. The route first displays the finite observation schedule and regular rational readback rows, so the consumer sees only finite windows and readback entries.

The dyadic tolerance ledger and terminal Real seal are then exposed before the completion criterion is consumed. Since the criterion is reached after those rows, the dependency records a reusable finite route rather than a hidden selected limit or ambient completeness principle. □

Theorem 3198.8 (Nested shrink interval rational locator route)**.** *Every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *route whose endpoint localization is read through rational endpoints first consumes the rational locator carrier of chapter 4202. Only after that locator row is displayed may the consumer read the*

nested shrink rows and then the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows. The route exports no selected completed point and no Heine–Borel compactness principle.*

Proof. Read the accepted packet through the carrier theorem of chapter 4227. The only endpoint localization rows available to the nested shrink route are finite displayed rows, so a rational-endpoint consumer must obtain its rational interval locator row before the shrink ledger can be interpreted.

The sibling chapter chapter 4202 supplies exactly that locator surface. Once it is consumed, the nested shrink packet may proceed through its displayed shrink ledger, finite windows, regular rational readback, and terminal real seal.

No selected completed point or Heine–Borel compactness row is present in either carrier. The dependency route therefore records only the rational locator read before the nested shrink and $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff rows are read. □

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ route consumes chapter 4202 for rational endpoint bracketing and chapter 301 for regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Thus the nested shrink interval dependency lattice reads locator cells first, then the shrinking-window ledger, then the regular readback route, not a selected intersection point or ambient completeness witness. $\mathsf{NestedShrinkInterval}^{\uparrow}$ exposes a completion-facing interval read only after the packet has consumed the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ diagonal row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational route, the terminal $\mathsf{Real}^{\uparrow}$ seal, the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ admission row, and the $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint-bracketing row. The public read is therefore a finite sibling route through diagonal selection, regular sequence readback, Real sealing, completion admission, and rational interval location; it exports no selected intersection point, quotient-real equality, countable choice, or ambient completeness principle. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the completion criterion surface of chapter 1476 before any interval-limit seal is visible. The shrinking-diameter ledger must first pass through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, regular rational readback, and the same Cauchy-completion admission route; only then may a downstream L10 consumer inspect the $\mathsf{Real}^{\uparrow}$ boundary, and the packet still exports no selected intersection point, quotient-real equality, countable choice, or ambient completion. The $\mathsf{NestedShrinkInterval}^{\uparrow}$ carrier depends on the substantive sibling lattice through chapter 320 for the nested interval spine, chapter 301 for regular rational readback, and chapter 38 for the terminal seal. The rational locator and Cauchy-completion routes already represented by the child files above supply the endpoint-bracketing and completion-admission rows, so the nested-shrink packet is read as a finite dependency route rather than a selected intersection point, quotient-real equality, countable choice, or ambient completeness principle.

Theorem 3198.9 (Nested shrink interval compact interval route)**.** *Every compact-interval consumer of an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet must read the compact real interval sibling of chapter 5553 and the nested interval convergence sibling of chapter 2057 through the same finite endpoint, diameter,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows displayed by the shrinking packet. The compact interval route supplies endpoint and finite-net context; the nested convergence route supplies the shrinking-window handoff; neither route adds an open-cover compactness theorem, selected limit point, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier of Definition 4227.1. The visible analytic rows are the nested source, located endpoints, dyadic diameter ledger, finite stream windows, regular rational readback, and Real seal, followed by componentwise transport, replay, provenance, and local naming.

Use the compact real interval packet of chapter 5553 only for the endpoint and finite-net context. Its compact rows are read before the shrinking packet is exported as an interval-local dependency, so the shrink route never obtains an unlisted open-cover or subcover selector.

Use the nested interval convergence packet of chapter 2057 for the convergence-facing handoff. That sibling reads finite endpoint windows, shrinking ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ sealing before a completion consumer may inspect the route.

The structural rows hsame, Cont, Pkg, and NameCert keep the two sibling reads attached to the same displayed packet. Hence a compact-interval dependency edge from $\mathsf{NestedShrinkInterval}^{\uparrow}$ is exactly the stated finite route and not a hidden compactness, quotient, choice, or completeness principle. □

Theorem 3198.10 (NestedShrinkInterval Cantor downstream route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer of a* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet reads the nested source only through nested closed intervals, located endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter witnesses,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The downstream route exports a Cantor diagonal or intersection-facing interval certificate, but it does not select a limit point, assert ambient completeness, identify quotient-real names, or add a host compactness principle outside the displayed rows.*

Proof. Project the nested-shrink carrier through the child routes named by the hub. The compact-interval and rational-locator rows provide nested closed intervals and located endpoints, while the dyadic and StreamName rows expose only finite diameter and window observations.

The Cantor-facing consumer then passes through the RegSeqRat and Real rows already used by the completion and sibling routes. Since these rows carry finite readback and terminal naming rather than a chosen point, the consumer obtains only the displayed interval certificate.

No carrier coordinate names a selected limit point, host completeness theorem, quotient equality, or ambient compactness object. Thus $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ consume $\mathsf{NestedShrinkInterval}^{\uparrow}$ only through the finite interval route stated above. □

$\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ reads the displayed uniform-modulus handoff of chapter 369 from regular Cauchy source names to regular Cauchy target names through the child packet below.

Chapter 3199

A Concrete Naming Certificate for RegularCauchyUniformContinuity$^{\uparrow}$

Following Bishop and Bridges, uniform continuity on displayed constructive real names is read here as a finite regular-Cauchy handoff rather than as host function equality or a completed-real theorem. The RegularCauchyUniformContinuity$^{\uparrow}$ packet records the standard fact that a displayed uniform modulus carries regular Cauchy names to regular Cauchy names. Its dependency surface is the finite StreamName$^{\uparrow}$ window interface of chapter 126, the regular-rational row of chapter 301, the pointwise map rows of chapter 130, the tolerance rows of chapter 369, and the terminal real-name seal of chapter 38.

Definition 3199.1 (Regular Cauchy uniform-continuity carrier)**.** *A* RegularCauchyUniformContinuity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, R, M, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* StreamName$^{\uparrow}$ *finite-window family opened for the regular Cauchy source name.*
2. *W records the target-window schedule requested by the regular Cauchy conclusion.*
3. *R is the source* RegSeqRat$^{\uparrow}$ *regularity row, including the rational tail bounds read on the windows in S.*
4. *M is the* ContinuousMap$^{\uparrow}$ *pointwise map row, read only through displayed source and target observations.*
5. *D is the* UniformModulus$^{\uparrow}$ *tolerance row that turns each target precision request into a source precision budget.*
6. *T is the target* RegSeqRat$^{\uparrow}$ *regularity row obtained by replaying D against R through M on the windows S, W.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the target regularity row T is displayed.*
8. *H records componentwise* hsame *transport for S, W, R, M, D, T, E and for the structural replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a target tolerance request through D, then S, R, M, W, then the target regularity row T, and finally the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, W, R, M, D, T, E, H, C, P, N$ and the displayed replay order. It has no coordinate for host function equality, quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 3199.2 (Regular Cauchy uniform-continuity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier $U = (S, W, R, M, D, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source stream windows, target window schedule, source regular-rational row, pointwise map row, uniform tolerance row, target regular-rational row, Real seal, transport, replay, provenance, and local naming row;*
2. *source regularity exactness, requiring every source observation to be read through S and R before the map row is inspected;*
3. *uniform tolerance exactness, requiring each target precision request to obtain its source budget only through D;*
4. *pointwise replay exactness, requiring the map row M to be applied only to source observations admitted by S, R, D;*
5. *target regularity handoff, requiring the target* $\mathsf{RegSeqRat}^{\uparrow}$ *row T to be produced before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed;*
6. *transport and continuation preservation, requiring H and C to preserve the displayed coordinates and their replay order;*
7. *provenance exhaustion, requiring P to name only the sibling chapters and kernel rows listed in Definition 4179.1;*
8. *local naming, requiring N to name exactly the* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *certificate row.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4179.1. The accepted packet exposes exactly $S, W, R, M, D, T, E, H, C, P, N$, so every local obligation must be read from that finite inventory.

The source side is exhausted by S and R. The stream row opens only finite windows, and the regular-rational row supplies the tail bounds on those windows before any map or target read is available.

The tolerance step is exhausted by D. A target precision request first reads the uniform-modulus row, which supplies the source precision budget used to choose the source observations.

The pointwise map step is then replayed through M. Since M is the only map coordinate and its inputs have already been confined to S, R, D, no host function equality or external continuity principle can enter this step.

The target handoff is recorded by T and sealed by E. The continuation row places T before E, while the transport row preserves the same coordinates under hsame.

Finally inspect P and N. The provenance row names only the sibling surfaces listed in the carrier, and the local naming row names only this certificate. Hence the eight obligations exhaust the NameCert surface. □

Theorem 3199.3 (Regular Cauchy uniform-continuity handoff)**.** *Let* $U = (S, W, R, M, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier. Every target* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by* U *factors through the finite route*

$$S, R, D, M, W, T.$$

Consequently a displayed uniform modulus carries the source regular Cauchy name to the target regular Cauchy name before any $\mathsf{Real}^{\uparrow}$ *seal or completion-facing consumer can read the packet.*

Proof. Start with a target precision request. The uniform-modulus row D converts that request into a source precision budget, and the source budget selects only the finite source windows named by S.

Read the source regularity row next. The row R supplies the regular Cauchy tail control on those windows, so the pointwise map row receives only displayed source observations.

Apply the pointwise map row M through the continuation route. The target-window schedule W records the finite target observations required by the regularity conclusion.

The target regularity row T is the only target $\mathsf{RegSeqRat}^{\uparrow}$ coordinate in the carrier. Since the continuation order places T after S, R, D, M, W and before the seal row E, every target regular Cauchy read factors through the displayed finite route. □

Theorem 3199.4 (Regular Cauchy uniform-continuity window composition)**.** *Let* $U = (S, W, R, M, D, T, E, H, C,$ *be an accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier. Suppose two target precision requests are displayed in the target-window schedule* W*, and suppose their* $\mathsf{UniformModulus}^{\uparrow}$ *rows refine through the same tolerance row* D*. Then the two requests compose to one source-window budget over* S, R *before the pointwise map row* M *is read.*

The two resulting target rows both factor through the same continuation route

$$S, R, D, M, W, T, E.$$

Thus the composed target read is a finite window-stability statement for the displayed carrier only: it exports no host function equality, quotient stream, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier and the local NameCert obligations from Definition 4179.1 and Definition 4179.2. The only rows available for target precision requests are the schedule W, the tolerance row D, the source windows S, the source regularity row R, the pointwise map row M, the target regularity row T, and the seal row E.

For the two requested target precisions, use the uniform tolerance exactness obligation twice. Because both requests refine through the same D row, their source precision needs are read as one finite source-window budget on S with regular-rational control supplied by R.

Apply Definition 4179.3 to each resulting target read. The handoff forces both reads through S, R, D, M, W, T, so the target rows factor through W, T before either can reach the seal row E.

Finally inspect the carrier inventory. There is no coordinate for host function equality, quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the composed route is exactly the displayed finite window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4179.1, Definition 4179.2, and Definition 4179.3 seed the finite route by which displayed uniform moduli carry regular Cauchy names to regular Cauchy names over StreamName windows, RegSeqRat rows, ContinuousMap pointwise rows, UniformModulus tolerance rows, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient streams, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity exactness, uniform tolerance exactness, pointwise replay exactness, target regularity handoff, transport and continuation preservation, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyUniformContinuityUp.

Constructive story: A RegularCauchyUniformContinuity$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat source and target rows, a ContinuousMap pointwise row, a UniformModulus tolerance row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The RegularCauchyUniformContinuity$^{\uparrow}$ route consumes RegSeqRat$^{\uparrow}$ readbacks through chapter 301, ContinuousMap$^{\uparrow}$ modulus-facing pointwise rows through chapter 130, and the UniformModulus$^{\uparrow}$ handoff through chapter 369. This child route exports no quotient stream, selected limit, host continuity, or ambient Real$^{\uparrow}$ completeness.

Chapter 3200

A Concrete Naming Certificate for $\mathsf{HenstockKurzweil}^{\uparrow}$

Following Henstock and Kurzweil, the $\mathsf{HenstockKurzweil}^{\uparrow}$ packet records a gauge-integral seed route in which tagged Riemann windows are controlled by a displayed gauge before any real-valued integral read is exported. It sits beside the gauge-integral packet of chapter 1999, the Riemann-integral packet of chapter 1526, the Riemann-sum gauge packet of chapter 4115, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, and the dyadic endpoint ledger of chapter 291. The chapter names the finite gauge-window carrier used by this integration route; it does not claim Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3200.1 (HenstockKurzweil carrier)**.** *A* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, I, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *source row naming the accepted gauge-integral boundary.*
2. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *comparison row that keeps the integral read tied to tagged mesh data.*
3. *S is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row carrying the finite tagged-sum window and gauge compatibility test.*
4. *W is the displayed gauge-window row, recording the tag, cell, and local radius data used by S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger for the cells in the displayed window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite window and dyadic tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, I, S, W, D, R have been replayed.*
8. *H records componentwise* hsame *transport for the gauge, integral, sum, window, dyadic, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a Henstock–Kurzweil request through $G, I, S, W, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HenstockKurzweil}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, I, S, W, D, R, E, H, C, P, N$. There is no coordinate for a measure space, an open-cover compactness theorem, a chosen refinement family, a selected infinite sum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HenstockKurzweilUp.HenstockKurzweilTasteGate single carrier alignment`

Theorem 3200.2 (HenstockKurzweil NameCert obligations)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier $K = (G, I, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$ *surface consists exactly of carrier admission, GaugeIntegral exposure through G, RiemannIntegral comparison through I, RiemannSumGauge window exposure through S, gauge-window exactness through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, host integrand equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4180.1. The displayed rows are exactly $G, I, S, W, D, R, E, H, C, P, N$, so carrier admission exposes only gauge, integral-comparison, tagged-sum, window, dyadic, readback, seal, transport, replay, provenance, and naming data.

The integration side starts with G, I, S, W. The gauge-integral source row G and Riemann-integral comparison row I are read only through the finite tagged-sum gauge row S and its displayed window W, so the packet has one gauge-window route rather than a detached refinement principle.

Next read the numerical handoff. The dyadic ledger D supplies endpoint and tolerance data for the same window, R records the regular rational readback, and E is available only after those rows have been replayed.

Finally the structural rows H, C, P, N preserve the same route. Since measure data, open-cover compactness, selected infinite sums, countable choice, host integrand equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local surface. □

Theorem 3200.3 (HenstockKurzweil gauge-window route)**.** *Every completion-facing or Real-facing read of an accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{GaugeIntegral}^{\uparrow} \longmapsto \mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The gauge-window row W is read before the dyadic tolerance ledger and the real seal, and the route cannot bypass $G, I, S, W, D, R, E, H, C, P, N$.

Proof. Begin with an accepted carrier read. The carrier exposes G, S, W before any numerical endpoint or real seal can be inspected, so the request has already passed through a displayed gauge-window row.

Replay the continuation row C. The read moves from the gauge source G through the tagged-sum gauge row S and the Riemann comparison row I, then through the dyadic ledger D and regular rational readback R before it reaches the seal E.

The structural rows H, P, N do not add another integration object. They preserve the same finite route, and no excluded measure, compactness, choice, selected-sum, or completeness coordinate is available in the packet. □

Theorem 3200.4 (HenstockKurzweil gauge-window exactness)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier $K = (G, I, S, W, D, R, E, H, C, P, N)$, the displayed gauge-window row W is projected after the gauge-integral source, the Riemann-integral comparison, and the tagged-sum gauge rows G, I, S,*

and before any $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read through* D, R, E. *No dyadic endpoint, regular rational readback, or real seal can be inspected without first replaying the same* G, I, S, W *window.*

Proof. Project the carrier of Definition 4180.1. The row order exposes G, I, S, W as the complete gauge-window prefix, with W recording the tag, cell, and local radius data used by the tagged-sum row S.

Now inspect the numerical rows. The dyadic ledger D, regular rational readback R, and terminal real seal E are downstream of that prefix, so each numerical read has already passed through the displayed window. Since the carrier has no independent dyadic, rational, or real coordinate, it cannot bypass W. □

Theorem 3200.5 (HenstockKurzweil RiemannSumGauge exposure)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier* $K = (G, I, S, W, D, R, E, H, C, P, N)$, *the row* S *is the only tagged-sum exposure route. Any completion-facing, dyadic-facing, regular-readback-facing, or real-facing use of tagged Riemann data must factor through* S *together with its displayed window* W *and the structural rows* H, C, P, N.

Proof. Read the carrier rows G, I, S, W. The only row tagged as $\mathsf{RiemannSumGauge}^{\uparrow}$ is S, and the only window attached to tagged-sum data is W.

The downstream rows D, R, E receive endpoint, tolerance, readback, and seal data from the same tagged-sum route. The structural rows H, C, P, N transport, replay, source, and name the carrier, but they do not introduce a second tagged-sum row. Hence every tagged Riemann use remains exposed through S. □

Theorem 3200.6 (HenstockKurzweil Real seal nonescape)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier* $K = (G, I, S, W, D, R, E, H, C, P, N)$, *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *cannot be reached without replaying* G, I, S, W, D, R. *In particular,* E *exposes no route through Lebesgue measure, open-cover compactness, selected infinite sums, countable choice, host integrand equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Replay the continuation row C through the carrier projection. The real-facing row appears as E, after the gauge-integral source G, Riemann comparison I, tagged-sum gauge row S, displayed window W, dyadic ledger D, and regular rational readback R.

The remaining rows H, P, N preserve that same route by transport, provenance, and local naming. They cannot supply a measure object, compactness theorem, selected sum, choice principle, host equality, or ambient completeness field, because none of those coordinates occur in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HenstockKurzweil}^{\uparrow}$ read factors through $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$.

Constructive story: A $\mathsf{HenstockKurzweil}^{\uparrow}$ packet is generated from a GaugeIntegral row, RiemannIntegral comparison row, RiemannSumGauge tagged-window row, gauge-window ledger, DyadicRatCore endpoint and tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 4180.1, Definition 4180.2, Definition 4180.3, Definition 4180.4, Definition 4180.5, and Definition 4180.6 close the finite gauge-window obligation surface over GaugeIntegralUp, RiemannIntegralUp, RiemannSumGaugeUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{HenstockKurzweil}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, host integrand equality, or ambient RealUp completeness.

Upgrade path: A formal route can encode checked carrier admission, RiemannIntegral comparison, DyadicRatCore tolerance exactness, RegSeqRat readback, and TasteGate field faithfulness for BEDC.Derived.HenstockKurzweilUp.

Chapter 3201

A Concrete Naming Certificate for HyperbolicVisualBoundary$^{\uparrow}$

The HyperbolicVisualBoundary$^{\uparrow}$ packet realizes the finite boundary-readback surface of chapter 9463. It sits after the metric-read surface of chapter 3474, the Busemann-boundary row of chapter 3732, the visual-metric packet of chapter 3866, the boundary-shadow packet of chapter 3813, and the boundary-fixed transport route of chapter 3500. The packet names only a finite visual boundary read over displayed disk, shadow, Busemann, visual-metric, transport, provenance, and local naming rows; it does not assert physical cosmology, a completed ideal-boundary quotient, global geometry, or a host compactification theorem.

Definition 3201.1 (Hyperbolic visual boundary carrier)**.** *A* HyperbolicVisualBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (D, S, B, M, T, P, H, C, N)$$

with the following displayed rows:

1. *D is the disk-readout row, exposing the finite* UnitDisk$^{\uparrow}$ *or Poincare-disk observation window from which the boundary read begins.*
2. *S is the boundary-shadow row, recording the* HyperbolicBoundaryShadow$^{\uparrow}$ *face visible from the disk readout.*
3. *B is the Busemann row, attaching the boundary read to* HyperbolicBusemannBoundary$^{\uparrow}$ *distance-difference data.*
4. *M is the visual-metric row, recording the* HyperbolicVisualMetric$^{\uparrow}$ *comparison used to distinguish visible boundary reads.*
5. *T is the transport row, recording the* PoincareBoundaryFixedTransport$^{\uparrow}$ *route that preserves the boundary read while disk data move.*
6. *P is the* Pkg *provenance over* HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, PoincareBoundaryFixedTransport$^{\uparrow}$, UnitDisk$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
7. *H records componentwise* hsame *transport for disk-readout, boundary-shadow, Busemann, visual-metric, transport, provenance, replay, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-read request through D, S, B, M, T, P, H.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, B, M, T, P, H, C, N$. It has no coordinate for a completed boundary space, quotient ideal-end equality, physical observer trajectory, global hyperbolic geometry, or an unlogged cosmological claim.

Lean4 ✓ `BEDC.Derived.HyperbolicVisualBoundaryUp.HyperbolicVisualBoundaryTasteGate single carrier alignment`

Theorem 3201.2 (Hyperbolic visual boundary NameCert obligations)**.** *For every accepted* HyperbolicVisualBounda *carrier*

$$V = (D, S, B, M, T, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$ *surface consists exactly of disk-readout admission, boundary-shadow exposure, Busemann attachment, visual-metric comparison, boundary-fixed transport, provenance, componentwise transport, displayed continuation replay, and local naming through $D, S, B, M, T, P, H, C, N$.*

Proof. Project the carrier of Definition 4181.1. The rows $D, S, B, M, T, P, H, C, N$ are the whole public packet.

The read starts at D and S. These rows expose only a finite disk window and a displayed boundary-shadow face.

The Busemann and visual-metric rows B, M attach that face to finite boundary comparison data. The transport row T records boundary-fixed movement without adding a completed boundary object.

Finally P, H, C, N source, transport, replay, and name the same route. Since no other coordinate is present, completed boundary spaces, quotient ideal-end equality, physical observer trajectories, global hyperbolic geometry, and unlogged cosmological claims remain outside the certificate. □

Theorem 3201.3 (Hyperbolic visual boundary readback non-escape)**.** *Every accepted* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *read factors through*

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, Poincare

together with disk-readout, provenance, transport, replay, and local NameCert *rows before it exports a visual boundary token.*

Proof. Apply Definition 4181.2. The accepted packet stores boundary readback only in the displayed rows $D, S, B, M, T, P, H, C, N$.

The metric, Busemann, visual-metric, shadow, and fixed-transport siblings are named by those rows and by provenance. A consumer therefore receives a boundary token only after the finite route has exposed those dependencies.

No row in the carrier stores physical cosmology, completed ideal-boundary equality, global geometry, or a host compactification theorem. Hence an accepted read cannot bypass the displayed packet through any of those exits. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ read is refuted if a consumer obtains a visual boundary token without displaying hyperbolicmetric, hyperbolic-busemann-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, poincare-boundary-fixed-transport, disk-readout, provenance, transport, replay, and local NameCert rows.

Independence witness: hyperbolicmetric, hyperbolic-busemann-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, poincare-boundary-fixed-transport: the HyperbolicMetric packet records metric reads without boundary readback, the Busemann packet records distance-difference boundary data without a visual-boundary token, the visual-metric packet records comparison without owning the boundary row, the boundary-shadow packet records a shadow face without the Busemann visual comparison, and the boundary-fixed transport packet preserves a boundary route without naming the visual boundary carrier. None is bijective to $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ packet is generated from disk-readout rows, HyperbolicBoundaryShadow rows, HyperbolicBusemannBoundary rows, HyperbolicVisualMetric rows, PoincareBoundaryFixedTransport rows, provenance, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4181.1, Definition 4181.2, and Definition 4181.3 seed the finite visual boundary readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical cosmology, completed ideal-boundary quotients, global hyperbolic geometry, host compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readback, shadow exposure, Busemann attachment, visual-metric comparison, boundary-fixed transport, provenance, replay, transport, and local naming for BEDC.Derived.HyperbolicVisualBoundaryUp.

Chapter 3202

A Concrete Naming Certificate for $\mathsf{BishopCompleteReal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompleteReal}^{\uparrow}$ packet records the textbook complete-real interface as a finite BEDC boundary rather than as an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. It sits on the dyadic endpoint ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated completion surface of chapter 1612, and the completion reflection boundary of chapter 403. The packet exposes the L10 completion interface currently needed by Real and Completion consumers; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete, select limits, quotient streams, or import countable choice.

Definition 3202.1 (Bishop complete real carrier)**.** *A* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, E, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger used to measure regular Cauchy requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row opened by the same tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic, schedule, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, recording that two completion-facing reads are compared through separated finite evidence.*
6. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *handoff row, replaying the same regular-rational source surface into the completion-facing interface.*
7. *H stores componentwise* hsame *transport for $D, W, R, E, S, F, C, P, N$.*
8. *C stores displayed* Cont *replay from a completion-facing request through the same dyadic, schedule, regular, real-seal, separated, and reflection rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient stream equality, a selected limit, countable choice, open-cover compactness, or a theorem that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.TasteGate.BishopCompleteRealTasteGate single carrier alignment`

Theorem 3202.2 (Bishop complete real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier*

$$B = (D, W, R, E, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every completion-facing read passes through* $D, W, R, E, S, F, H, C, P, N$ *in that order.*

No obligation in the packet is discharged by host real equality, quotient streams, selected representatives, countable choice, open-cover compactness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 4233.1. The tuple lists the dyadic, schedule, regular readback, Real seal, separated comparison, completion reflection, transport, replay, provenance, and naming rows.

The completion-facing order is fixed by the visible rows. A request first reads a dyadic endpoint and tolerance ledger in D, then opens the finite schedule row W, and then passes through the regular rational readback row R before the terminal Real seal E is available.

The separated and reflection rows then consume that same source surface. The comparison row S records separated evidence, while F replays the regular-rational source into the completion-facing boundary.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded host, quotient, choice, compactness, or completeness data, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.BishopCompleteRealNamecertObligations`

Theorem 3202.3 (Bishop complete real L10 handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *packet factors through the finite L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{SeparatedCompletion}^{\uparrow} \longrightarrow \mathsf{CompletionReflection}^{\uparrow}.$$

The handoff exports a displayed regular-rational completion interface with separated comparison and reflection rows. It does not export completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, selected host limits, countable-choice coordinates, open-cover compactness, or an ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4233.2. They force every accepted read to pass through the dyadic ledger, finite schedule, regular rational readback, and terminal Real seal before any completion-facing row is consumed.

Next read the separated comparison row. The row S compares completion-facing observations only through displayed separated evidence, so it cannot introduce quotient classes or host equality.

The completion reflection row F then replays the same source surface into the completion-facing interface. The structural rows H, C, P, N preserve that route componentwise and add only transport, replay, provenance, and local naming.

Thus the consumer receives exactly the displayed L10 handoff from dyadic data to schedules, regular rational readback, Real seal, separated comparison, and completion reflection. None of the excluded completeness, choice, quotient, compactness, or selected-limit data has an entry point in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompleteReal}^{\uparrow}$ packet is generated from a dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4233.1, Definition 4233.2, and Definition 4233.3 seed the finite Bishop complete-real interface for the current L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, selected host limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated comparison, completion reflection handoff, and ledger non-escape for the same packet.

Chapter 3203

A Concrete Naming Certificate for $\mathsf{RatCauchyGapWitness}^{\uparrow}$

Following Bishop and Bridges on constructive completion of the rationals, the $\mathsf{RatCauchyGapWitness}^{\uparrow}$ packet records the finite obstruction visible at the $\mathsf{Rat}^{\uparrow}$ boundary before any real-name or completion-facing consumer reads a limit. It sits after the rational carrier of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed gap witness for rational Cauchy bracketing; it does not export a host irrational, quotient completion class, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3203.1 (Rat Cauchy gap witness carrier)**.** *A* $\mathsf{RatCauchyGapWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the rational lower-bracket row, exposing a finite* $\mathsf{Rat}^{\uparrow}$ *source, monotone lower estimates, and the rational comparison probes consumed by the witness.*
2. *U is the rational upper-bracket row, exposing the matching* $\mathsf{Rat}^{\uparrow}$ *source, monotone upper estimates, and the rational separation probes kept visible to the consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ladder, recording the dyadic widths used to compare lower and upper brackets at each requested precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule, listing the finite observation windows through which the bracket rows are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, recording the regular-rational readback discipline available before the terminal seal is cited.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, present only as a downstream boundary saying where a real-name consumer may later read a sealed object.*
7. *H records componentwise* hsame *transport for the bracket, tolerance, window, handoff, seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from rational brackets through dyadic tolerances, finite windows, regular-rational readback, and the real seal boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RatCauchyGapWitness}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, U, D, W, R, E, H, C, P, N$. It has no coordinate for a host irrational number, quotient-completion representative, selected limit, countable-choice principle, or ambient completeness assertion.

Lean4 ✓ `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessTasteGate single carrier alignment`

Theorem 3203.2 (Rat Cauchy gap witness NameCert obligations)**.** *For every accepted* $\mathsf{RatCauchyGapWitness}^{\uparrow}$ *carrier $G = (L, U, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$ *surface consists exactly of five mathematical obligation rows: rational lower and upper brackets through L, U; dyadic tolerance control through D; finite stream-window inspection through W; regular-rational handoff through R; and the terminal real-seal boundary through E. The structural rows H, C, P, N transport, replay, source, and name that same packet, and no host irrational, quotient completion, or choice-selected limit is exported.*

Proof. Project the accepted carrier of Definition 4183.1. The rows L and U are the only rational bracket coordinates, and both are read through the $\mathsf{Rat}^{\uparrow}$ source rather than through a completed real object.

Next inspect the precision route. The dyadic row D supplies the tolerance ladder, the stream row W supplies the finite observation windows, and the regular-rational row R supplies the handoff available to completion-facing consumers.

The seal row E is only a boundary row. It records where a later $\mathsf{Real}^{\uparrow}$ consumer may read a terminal seal, while the present certificate still exposes the rational brackets, dyadic tolerances, stream windows, and regular-rational handoff.

Finally apply the structural rows. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet. Since the carrier has no coordinate for a host irrational, quotient-completion representative, selected limit, or choice principle, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessNameCert obligations`

Falsifiable prediction: Every accepted $\mathsf{RatCauchyGapWitness}^{\uparrow}$ consumer must expose rational lower and upper brackets, a dyadic tolerance ladder, finite stream windows, a regular-rational handoff, a real-seal boundary, transport, Cont replay, Pkg provenance, and local NameCert rows before it can cite the witness.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{RatCauchyGapWitness}^{\uparrow}$, because this carrier alone binds rational lower and upper brackets to the dyadic tolerance ladder, finite window schedule, regular-rational handoff, terminal seal boundary, transport, replay, provenance, and local NameCert rows without exporting a completed real.

Closure status for $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RatCauchyGapWitness}^{\uparrow}$ packet is generated from rational lower and upper bracket rows, a DyadicRatCore tolerance ladder, a StreamName window schedule, a RegSeqRat handoff, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$, seed)

Scope closed: Definition 4183.1 and Definition 4183.2 seed the finite rational Cauchy gap witness over $L, U, D, W, R, E, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessNameCert\_obligations`

Bridge status: none

Not claimed: This chapter does not close host irrational existence, quotient completion equality, selected limits, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rational bracket, dyadic tolerance, finite window, regular-rational handoff, real-seal boundary, transport, replay, provenance, and local naming rows for BEDC.Derived.RatCauchyGapWitnessUp.

Chapter 3204

A Concrete Naming Certificate for $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$

Following Riesz and Markov-Kakutani on positive linear functionals over finite lattice data, the $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ packet records the BEDC-safe finite representation surface for lattice-positive functionals. It consumes the order rows of chapter 53, the lattice rows of chapter 55, the one-sided real boundary of chapter 38, and the lower and upper real approximation rows of chapter 1446 and chapter 1447. The packet is a finite naming certificate for positive functional handoff; it does not assert sigma-additivity, measure completion, quotient dual equality, choice-selected representing measures, or ambient Banach-space representation.

Definition 3204.1 (Riesz representation finite lattice carrier)**.** *A* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, O, F, A, B, P, H, C, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{Lattice}^{\uparrow}$ *row, exposing the finite join, meet, top, bottom, and lattice comparison data used by the representation read.*
2. *O is the* $\mathsf{Poset}^{\uparrow}$ *row supporting the order classifier shared by the lattice and positivity checks.*
3. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$*-style positive functional row, recording additivity on displayed compatible lattice pieces and positivity on the order-positive cone.*
4. *A is the one-sided* $\mathsf{Real}^{\uparrow}$ *boundary row through which the functional values are read.*
5. *B is the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *bracketing row, storing finite lower and upper cuts that witness the one-sided value interval.*
6. *P is the* Pkg *provenance over* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *H records componentwise* hsame *transport for lattice, order, positive-functional, one-sided real, bracketing, provenance, replay, and naming rows.*
8. *C records displayed* Cont *replay from a positive-functional request through L, O, F, A, B, P, H.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, O, F, A, B, P, H, C, N$. It has no coordinate for completed measure spaces, quotient dual equality, countable joins, choice-selected representing measures, sigma-additivity, or host functional-analysis completion.

Lean4 ✓ `BEDC.Derived.RieszRepresentationFiniteLatticeUp.RieszRepresentationFiniteLatticeTasteGate single carrier alignment`

Theorem 3204.2 (Riesz representation finite lattice NameCert obligations)**.** *For every accepted* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier*

$$R = (L, O, F, A, B, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$ *surface consists exactly of finite lattice admission, shared poset classifier transport, positive-functional row exposure, one-sided real value readback, lower and upper real bracketing, provenance, componentwise transport, displayed continuation replay, and local naming through* $L, O, F, A, B, P, H, C, N$.

Proof. Project the carrier of Definition 4184.1. The rows $L, O, F, A, B, P, H, C, N$ are the whole accepted packet.

The representation read starts with the lattice and order rows. The positive-functional row F is checked only against the displayed lattice operations and the common order classifier supplied by O.

The value side is bounded by A and B. The one-sided $\mathsf{Real}^{\uparrow}$ row exposes the value seal, while the $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ row records the finite bracketing data used by the consumer.

Finally P, H, C, N source, transport, replay, and name the same route. Since the carrier contains no additional coordinate, measure completion, quotient dual equality, countable joins, selected representing measures, and host completion remain outside the local certificate. □

Theorem 3204.3 (Riesz representation finite lattice positive-functional handoff)**.** *Every positive-functional consumer of an accepted* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Lattice}^{\uparrow},\ \mathsf{Poset}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LowerReal}^{\uparrow},\ \mathsf{UpperReal}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow}$$

together with BHist, hsame, Cont, Pkg, *and local* NameCert *rows before it exports a representation token.*

Proof. Apply Definition 4184.2. The theorem exposes the lattice, order, positive-functional, one-sided real, bracketing, transport, replay, provenance, and naming rows as the whole obligation surface.

A downstream consumer can therefore read a representation token only after L, O, F, A, B have been displayed in the same finite packet. The structural rows P, H, C, N preserve that route and do not add a second source of representation data.

No row supplies a completed measure, quotient dual equality, countable lattice operation, selected representing measure, sigma-additivity principle, or host Banach-space representation. The handoff is exactly the finite positive-functional route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ packet is generated from finite lattice rows, poset classifier rows, positive-functional rows, one-sided Real rows, lower and upper real bracketing rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RieszRepresentationFiniteLattice}\uparrow}$, seed)

Scope closed: Definition 4184.1, Definition 4184.2, and Definition 4184.3 seed the finite positive-functional representation route.

Formal verification: FormalStatus(NameCert$_{\text{RieszRepresentationFiniteLattice}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim sigma-additive measure representation, completed measure spaces, quotient dual equality, countable lattice joins, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite lattice admission, order-positive functional exposure, one-sided value bracketing, replay, transport, provenance, and local naming for BEDC.Derived.RieszRepresentationFiniteLatticeUp.

Chapter 3205

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$

The $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ packet realizes the focused-ledger, shadow-metric, boundary-shadow, and Cont-growth route of chapter 9463 as finite NameCert data. It sits after the metric-read surface of chapter 3474, the observer-shadow metric row of chapter 4154, the boundary-shadow packet of chapter 3813, the Poincare-disk dynamics route of chapter 4117, and the finite dynamics rows of chapter 195. The packet names only an observer-expansion read over displayed disk, shadow, boundary, growth, provenance, and local naming rows; it does not assert physical cosmology, global expansion, completed boundary geometry, or an unlogged observer.

Definition 3205.1 (Hyperbolic phase observer-expansion carrier)**.** *A* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, D, S, B, G, P, H, C, N)$$

with the following displayed rows:

1. *F is the focused-ledger row, recording the finite observer-facing history slice to be read.*
2. *D is the disk-readout row, exposing the Poincare disk window and its displayed phase coordinates.*
3. *S is the shadow-metric row supplied by* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *over the same observer slice.*
4. *B is the boundary-shadow row supplied by* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *for the visible boundary face.*
5. *G is the Cont-growth row, recording the finite* $\mathsf{DynSystem}^{\uparrow}$ *and* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *replay by which the observer read expands through displayed disk stages.*
6. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicObserverShadow}$ $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{PoincareDiskDynamics}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *H records componentwise* hsame *transport for focused-ledger, disk-readout, shadow-metric, boundary-shadow, growth, provenance, replay, and naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion request through F, D, S, B, G, P, H.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, S, B, G, P, H, C, N$. It has no coordinate for physical expansion, completed boundary quotient, global hyperbolic geometry, host observer identity, or unlogged cosmological data.

Lean4 ✓ `BEDC.Derived.HyperbolicPhaseObserverExpansionUp.HyperbolicPhaseObserverExpansionTasteGate`
`single carrier alignment`

Theorem 3205.2 (Hyperbolic phase observer-expansion NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ *carrier*

$$E = (F, D, S, B, G, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ *surface consists exactly of focused-ledger admission, disk-readout exposure, shadow-metric comparison, boundary-shadow display, Cont-growth replay, provenance, componentwise transport, displayed continuation replay, and local naming through* $F, D, S, B, G, P, H, C, N$.

Proof. Project the carrier of Definition 4185.1. The rows $F, D, S, B, G, P, H, C, N$ are the complete public packet.

The observer-facing read begins with F and D. These rows expose a finite ledger slice and a displayed disk window before any expansion token is available.

The shadow rows S and B then attach the observer slice to finite metric and boundary-shadow data. The growth row G records the Poincare-disk and dynamics replay that the consumer may inspect.

Finally P, H, C, N source, transport, replay, and name the same route. Since no other coordinate is present, physical expansion, completed boundary geometry, global hyperbolic structure, host observer identity, and unlogged cosmological claims remain outside the certificate. □

Theorem 3205.3 (Hyperbolic phase observer-expansion nonescape)**.** *Every accepted* HyperbolicPhaseObserverExpa *read factors through hyperbolic-metric, hyperbolic-observer-shadow-metric, hyperbolic-boundary-shadow, poincare-disk-dynamics, and dynsystem rows together with focused-ledger, disk-readout, Cont-growth, provenance, transport, replay, and local* NameCert *data before it exports an observer-expansion token.*

Proof. Apply Definition 4185.2. The accepted packet stores observer expansion only in the displayed rows $F, D, S, B, G, P, H, C, N$.

The sibling metric, shadow, boundary, disk-dynamics, and dynamics surfaces are named by S, B, G and by provenance. A consumer therefore receives an observer-expansion token only after the finite route has displayed those dependencies.

No row in the carrier stores physical cosmology, completed boundary equality, global geometry, host observer identity, or an unlogged expansion source. Hence an accepted read cannot bypass the displayed packet through any of those exits. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ read is refuted if a consumer obtains an observer-expansion token without displaying focused-ledger, disk-readout, shadow-metric, boundary-shadow, Cont-growth, provenance, and local NameCert rows.

Independence witness: hyperbolic-metric, hyperbolic-observer-shadow-metric, hyperbolic-boundary-shadow, poincare-disk-dynamics, dynsystem: the HyperbolicMetric packet records metric reads without observer expansion, the observer-shadow metric packet records shadow comparison without boundary growth, the boundary-shadow packet records a visible boundary face without disk

dynamics, the Poincare-disk dynamics packet records disk replay without owning the observer ledger, and the DynSystem packet records finite dynamics without naming the hyperbolic phase observer-expansion carrier. None is bijective to $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ packet is generated from focused-ledger rows, disk-readout rows, hyperbolic observer-shadow metric rows, boundary-shadow rows, Poincare-disk dynamics rows, DynSystem growth replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4185.1, Definition 4185.2, and Definition 4185.3 seed the finite observer-expansion readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical cosmology, completed boundary quotients, global hyperbolic geometry, host observer identity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked focused-ledger admission, disk-readout exposure, shadow-metric and boundary-shadow exactness, Cont-growth replay, provenance, transport, and local naming for BEDC.Derived.HyperbolicPhaseObserverExp

Chapter 3206

A Concrete Naming Certificate for $\mathsf{NowhereDense}^{\uparrow}$

Following Baire and Oxtoby, a nowhere-dense component is treated here as the finite certificate surface that lets Baire-category arguments read a closed-frontier refusal without importing arbitrary subset quantification. The $\mathsf{NowhereDense}^{\uparrow}$ packet sits between the topology carrier of chapter 88, the dense-open route of chapter 1542, and the finite cover ledger consumed by chapter 2267.

Definition 3206.1 (Nowhere-dense carrier)**.** *A* $\mathsf{NowhereDense}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, B, F, E, D, M, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, including the basic-neighbourhood names inspected by the packet.*
2. *B is the finite basic-neighbourhood request ledger over T.*
3. *F is the closure-frontier row naming how inspected neighbourhoods meet the displayed closure boundary.*
4. *E is the empty-interior refusal row; for every requested basic neighbourhood it displays a sub-neighbourhood escaping the named closed fragment.*
5. *D is the dense-open complement handoff read by* $\mathsf{BaireCategory}^{\uparrow}$ *consumers.*
6. *M is the finite cover-row interface consumed by* $\mathsf{MeagreSet}^{\uparrow}$*.*
7. *H stores the componentwise* hsame *transports for T, B, F, E, D, M.*
8. *C stores displayed* Cont *replay of the neighbourhood requests, closure-frontier inspections, empty-interior refusals, and dense-open handoff.*
9. *P is the local* Pkg *provenance row tying the packet to the named topology source.*
10. *N is the local* $\mathsf{NameCert}$ *row for the displayed nowhere-dense component.*

The carrier exports no arbitrary subset classifier, host closure operator, selected generic point, countable-choice selector, or ambient Baire theorem.

Lean4 ✓ `BEDC.Derived.NowhereDenseUp.NowhereDenseTasteGate single carrier alignment`

Theorem 3206.2 (NowhereDense NameCert obligations)**.** *Every accepted* NowhereDense$^{\uparrow}$ *carrier* $Q = (T, B, F, E, D, M, H, C, P, N)$ *supplies exactly the topology source, basic-neighbourhood ledger, closure-frontier row, empty-interior refusal row, dense-open complement handoff, meagre-set cover interface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row listed in Definition 4186.1. A* BaireCategory$^{\uparrow}$ *or* MeagreSet$^{\uparrow}$ *consumer can read the component only through those rows.*

Proof. Open the carrier definition. Each coordinate of Q is a finite row, and the topology source T is the only source from which basic neighbourhoods may be requested.

The closure-frontier and empty-interior rows are then read as paired inspections: F names the frontier contact and E supplies the escaping sub-neighbourhood. The dense-open complement row D and meagre cover row M are downstream of those inspections.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no arbitrary subset classifier, host closure operator, generic point selector, or countable-choice coordinate occurs in Q, downstream Baire-category and meagre-set consumers receive exactly the displayed finite packet. □

Theorem 3206.3 (NowhereDense closure-interior refusal)**.** *Let* $Q = (T, B, F, E, D, M, H, C, P, N)$ *be an accepted* NowhereDense$^{\uparrow}$ *carrier. For every basic-neighbourhood request recorded by* B*, the packet routes the request through the closure-frontier row* F *and then through the empty-interior refusal row* E*, producing the dense-open complement handoff* D *used by* BaireCategory$^{\uparrow}$ *and the cover-row interface* M *used by* MeagreSet$^{\uparrow}$*.*

Proof. Begin with a displayed request in B. The row is indexed over the same topology source T, so it has access only to the named basic-neighbourhood surface.

Read F next. It identifies the closure-frontier inspection attached to the request. Then read E, which displays a sub-neighbourhood escaping the named closed fragment and therefore records empty interior on the inspected finite surface.

The complement handoff D and cover interface M are built from that same refusal. Transport, replay, provenance, and local naming do not alter the request; they keep the Baire-category and meagre-set consumers on the displayed route. □

Closure status for NameCert$_{\text{NowhereDense}^{\uparrow}}$**.**

Constructive story: A NowhereDense$^{\uparrow}$ packet is generated from a Topology$^{\uparrow}$ source, finite basic-neighbourhood requests, a closure-frontier row, an empty-interior refusal row, a dense-open complement handoff, a MeagreSet$^{\uparrow}$ cover interface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{NowhereDense}^{\uparrow}}$, seed)

Scope closed: Definition 4186.1, Definition 4186.2, and Definition 4186.3 seed the finite closure-frontier and empty-interior refusal route consumed by Baire-category and meagre-set packets.

Formal verification: FormalStatus(NameCert$_{\text{NowhereDense}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NowhereDenseUp.NowhereDenseTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim arbitrary subset closure, host interior operators, selected generic points, countable choice, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite closure-frontier, empty-interior refusal, dense-open complement, and cover-interface obligations.

Chapter 3207

A Concrete Naming Certificate for $\mathsf{SeparableReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SeparableReal}^{\uparrow}$ packet records separability as a finite NameCert surface over dyadic schedules rather than as host enumeration or countable choice. It sits over the real-name seal of chapter 38, the rational carrier of chapter 37, the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, and the existing real separability route of chapter 1809. The packet names the constructive dense schedule visible to Real and Completion consumers; it is not a quotient-real equality principle, a selected host list of all reals, a trichotomy oracle, or an assertion of ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3207.1 (SeparableReal carrier)**.** *A* $\mathsf{SeparableReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, S_0, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dense-schedule row, recording the finite dyadic centers and tolerance budgets inspected by a separability request.*
2. *S_0 is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the observation windows through which the dense schedule is read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing regular rational approximants attached to the same windows and dyadic budgets.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording rational comparison data only at the displayed dyadic precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the schedule, window, readback, and tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for the dense schedule, finite windows, readback, tolerance, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a separability query through D, S_0, R, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparableReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $D, S_0, R, Q, E, H, C, P, N$. It has no coordinate for host enumeration of $\mathsf{Real}^\uparrow$*, countable choice, quotient real equality, trichotomy of real order, hidden completion data, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeparableRealUp.SeparableRealTasteGate single carrier alignment`

Theorem 3207.2 (SeparableReal dyadic density NameCert)**.** *For every accepted* $\mathsf{SeparableReal}^\uparrow$ *carrier*

$$S = (D, S_0, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$ *surface exposes density only by the finite dyadic schedule D, the finite* $\mathsf{StreamName}^\uparrow$ *windows S_0, the* $\mathsf{RegSeqRat}^\uparrow$ *readback row R, the rational and dyadic tolerance row Q, and the terminal* $\mathsf{Real}^\uparrow$ *seal E. The remaining rows H, C, P, N provide componentwise transport, displayed replay, package provenance, and local naming for that same route.*

Consequently an accepted packet supplies a separability witness for Real and Completion consumers without host enumeration, countable choice, quotient real equality, real-order trichotomy, hidden completion data, or a completeness theorem for $\mathsf{Real}^\uparrow$*.*

Proof. Project the carrier of Definition 4187.1. The tuple lists exactly the dense dyadic schedule, finite observation windows, regular rational readback, rational tolerance data, terminal Real seal, transport, replay, provenance, and local naming rows.

The density route is fixed before any real-name consumer reads it. A separability query first opens the dyadic schedule row D and the finite window row S_0, then reads the regular rational approximants in R against the tolerance row Q.

The terminal Real seal E is downstream of those displayed rows, so a consumer never receives an unsealed host real or a selected quotient representative. The structural rows H, C, P, N transport, replay, source, and name only the same displayed data.

Since the carrier contains no coordinate for host enumeration, countable choice, quotient equality, trichotomy, hidden completion data, or ambient completeness, the listed rows exhaust the local NameCert surface. The accepted packet therefore exposes density only through the finite dyadic schedule and the $\mathsf{Real}^\uparrow$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$.

Constructive story: A $\mathsf{SeparableReal}^\uparrow$ packet is generated from a dyadic dense schedule, finite StreamName windows, RegSeqRat readback, Rat and DyadicRatCore tolerance rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$, seed)

Scope closed: Definition 4187.1 and Definition 4187.2 seed the finite dyadic density surface for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host enumeration, countable choice, quotient real equality, real-order trichotomy, hidden completion data, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic density schedule exposure, StreamName window replay, RegSeqRat readback, Rat and DyadicRatCore tolerance exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeparableRealUp.

Chapter 3208

A Concrete Naming Certificate for $\mathsf{BanachFixedPointModulus}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{BanachFixedPointModulus}^{\uparrow}$ packet records the explicit contraction-rate modulus consumed before the endpoint seal of chapter 396. It sits after the residual-to-error route of chapter 2233 and before any $\mathsf{BanachFixedPoint}^{\uparrow}$ consumer reads a $\mathsf{Real}^{\uparrow}$ endpoint boundary. The packet names only the finite contraction graph, the rational Lipschitz ledger below one, the Picard ledger, the Cauchy threshold row, the regular-rational handoff, the complete-metric service row, and the endpoint-facing seal; it does not select a fixed point, assert host equality, or import ambient completeness outside the displayed carrier rows.

Definition 3208.1 (Banach fixed-point modulus carrier)**.** *A* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, T, L, I, M, R, C, E, H, Q, P, N)$$

with the following visible coordinates:

1. *B is the source* $\mathsf{BanachFixedPoint}^{\uparrow}$ *row whose endpoint construction is waiting for an explicit modulus route.*
2. *T is the contraction graph row, displaying the finite map-application history used by the Picard iteration.*
3. *L is the rational Lipschitz ledger below one, including the displayed positivity and unit-bound rows read by the contraction estimate.*
4. *I is the Picard iteration ledger, recording the finite adjacent iterate rows and the replay order for $x_{k+1} = T(x_k)$.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from L and I for each requested rational tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational name after the modulus threshold has been displayed.*
7. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed by the completion-facing handoff.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, C have been read.*
9. *H stores componentwise* hsame *transports for the fixed-point, graph, Lipschitz, Picard, modulus, readback, complete-metric, seal, provenance, and local naming rows.*

10. *Q stores displayed* Cont *replay from contraction application to Picard iteration, from the Lipschitz ledger and Picard ledger to the modulus threshold, and from the modulus threshold to the regular-rational and endpoint-facing rows.*

11. *P is the* Pkg *provenance over* BanachFixedPointModulus$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardErrorEstimate$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{BanachFixedPointModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, L, I, M, R, C, E, H, Q, P, N$. It has no coordinate for a choice-selected limit, a host fixed-point equality proof, quotient completion equality, global nonlinear analysis, or completeness beyond the displayed CompleteMetric$^{\uparrow}$ *service row.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointModulusUp.BanachFixedPointModulusTasteGate single carrier alignment`

Theorem 3208.2 (Banach fixed-point modulus NameCert obligations)**.** *For every accepted* BanachFixedPointModu *carrier $K = (B, T, L, I, M, R, C, E, H, Q, P, N)$, the local* NameCert$_{\text{BanachFixedPointModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, contraction-graph visibility through T, rational Lipschitz visibility below one through L, finite Picard iteration access through I, Cauchy-threshold construction through M, regular-rational readback through R, complete-metric service access through C, endpoint seal access through E, componentwise* hsame *transport through H, displayed* Cont *replay through Q, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4188.1. The accepted packet exposes exactly the fixed-point row, contraction graph, Lipschitz ledger, Picard ledger, modulus row, regular-rational readback, complete-metric service, endpoint seal, transport, replay, provenance, and local naming rows.

The contraction-rate information is exhausted by T, L, I. The graph row supplies the finite application history, the Lipschitz ledger supplies the displayed bound below one, and the Picard ledger supplies the adjacent iterate rows used by the threshold calculation.

The handoff information is exhausted by M, R, C, E. The modulus threshold row is formed before any regular-rational readback, the complete-metric service is read only at the handoff boundary, and the endpoint seal is exposed only after those rows have been displayed. The structural rows H, Q, P, N transport, replay, source, and name this same route, so no further NameCert obligation is available. □

Theorem 3208.3 (Banach fixed-point modulus Picard handoff)**.** *Every* BanachFixedPoint$^{\uparrow}$ *endpoint-sealing route that consumes a* BanachFixedPointModulus$^{\uparrow}$ *carrier must first factor through*

$$T, L, I \longmapsto M \longmapsto R, C \longmapsto E.$$

The Picard and contraction-rate rows therefore supply the explicit CauchyModulus$^{\uparrow}$ *threshold before the* RegSeqRat$^{\uparrow}$ *readback, the* CompleteMetric$^{\uparrow}$ *service row, and the* Real$^{\uparrow}$ *seal are available to the fixed-point endpoint boundary. The Cauchy-threshold row is read through the* Cauchy$^{\uparrow}$ *naming surface of chapter 3413 before it is specialized to* CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, *and* Real$^{\uparrow}$.

Proof. Start with a fixed-point endpoint request from the source row B. The only contraction-rate rows in the carrier are the graph row T, the Lipschitz ledger L, and the Picard ledger I, so the request must read those rows before it can form a threshold.

The threshold is the displayed CauchyModulus$^{\uparrow}$ coordinate M. The existing BanachFixedPoint$^{\uparrow}$ route of chapter 396 consumes such a threshold before endpoint sealing, while the PicardErrorEstimate$^{\uparrow}$ route of chapter 2233 supplies the residual-and-ratio pattern that makes the threshold finite.

After M is displayed, the handoff can read R and C, and only then can it expose the endpoint-facing row E. The transport, replay, provenance, and local naming rows preserve this order, so the packet exports no selected fixed point, host equality, or ambient completeness theorem. □

Theorem 3208.4 (Banach fixed-point modulus endpoint seal determinacy)**.** *Let $K = (B, T, L, I, M, R, C, E, H, Q,$ be an accepted* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *carrier. Suppose two endpoint-facing reads use the same displayed route*

$$T, L, I \longmapsto M \longmapsto R, C \longmapsto E$$

and their Picard, Lipschitz, modulus, regular-rational, complete-metric, and endpoint rows are compatible under the componentwise hsame *transports carried by H. Then the two reads expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal classifier at E. The endpoint seal is determined by the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CompleteMetric}^{\uparrow}$ *service rows before any fixed-point endpoint consumer can inspect it.*

Proof. Project the carrier of Definition 4188.1. Both endpoint-facing reads have the same contraction graph, Lipschitz ledger, Picard ledger, modulus threshold, regular-rational readback, complete-metric service row, and terminal seal coordinate.

Apply Definition 4188.3. It forces every endpoint read through T, L, I before M, and through M before the R, C, E handoff, so neither read can introduce an independent endpoint route.

The shared $\mathsf{RegSeqRat}^{\uparrow}$ row and $\mathsf{CompleteMetric}^{\uparrow}$ service row are then read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport through H and replay through Q identify the two reads along those same rows.

Thus the terminal classifier at E is fixed by the displayed route. No selected limit, quotient completion equality, host fixed-point equality, or ambient completeness principle is used to compare the endpoint seals. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachFixedPointModulus}^{\uparrow}$ packet is generated from a BanachFixedPoint row, a contraction graph, a rational Lipschitz ledger below one, a Picard iteration ledger, a CauchyModulus threshold, a RegSeqRat readback, a CompleteMetric service row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4188.1, Definition 4188.2, and Definition 4188.3 seed the finite contraction-rate modulus route consumed before Banach fixed-point endpoint sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, ambient completeness, Lean verification, bridge checking, or global nonlinear analysis.

Upgrade path: Obligation closure requires checked carrier admission, contraction-graph visibility, Lipschitz-below-one visibility, Picard iteration access, Cauchy-threshold construction, regular-rational readback, complete-metric service access, endpoint seal access, transport, replay, provenance, and local naming for BEDC.Derived.BanachFixedPointModulusUp.

Chapter 3209

A Concrete Naming Certificate for $\mathsf{SpreadSpace}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on spreads and choice sequences, the $\mathsf{SpreadSpace}^{\uparrow}$ packet records a finite-prefix space of admissible choice-sequence names rather than an ambient set of completed infinite branches. It sits between the Baire carrier of chapter 334, the finite stream-name consumer of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. Its spread-side input is supplied by the Brouwer spread packet of chapter 1677, the local law packet of chapter 1532, and the choice-sequence packet of chapter 1482.

Definition 3209.1 (SpreadSpace carrier)**.** *A* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, L, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix row, recording the finite natural-number windows on which the packet may be inspected.*
2. *L is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row, deciding whether each displayed prefix is permitted by the spread law.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *branch-prefix row, exposing only those choice-sequence observations whose prefixes pass L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by downstream name consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the accepted branch prefixes and stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, L, Q, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the prefix, law, branch, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a spread-space query through B, L, Q, W, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{SpreadSpace}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, Q, W, R, E, H, C, P, N$. It has no coordinate for a completed infinite branch, host equality of functions, countable choice, selected real values, bar induction outside the displayed prefix bar, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SpreadSpaceUp.spreadSpaceTasteGate`
Lean4 ✓ `BEDC.Derived.SpreadSpaceUp.SpreadSpaceTasteGate single carrier alignment`

Theorem 3209.2 (SpreadSpace prefix-bar stability)**.** *For every accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier*

$$S = (B, L, Q, W, R, E, H, C, P, N),$$

any downstream read that reaches a $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row must first pass through the same finite* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, the same* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row L, and the same* $\mathsf{ChoiceSequence}^{\uparrow}$ *branch-prefix row Q. If two accepted reads are componentwise related by the displayed* hsame *row H and replayed by* Cont *through C, then their prefix-bar boundary is transported without adding an infinite branch, host function equality, countable choice, selected real value, or ambient completeness coordinate.*

Proof. Project the carrier in Definition 4189.1. The only entrance to a downstream name read is the displayed prefix route B, L, Q.

The stream and rational rows are downstream of that entrance. A $\mathsf{StreamName}^{\uparrow}$ window W and a $\mathsf{RegSeqRat}^{\uparrow}$ readback R are available only after the finite Baire prefix has been admitted by the spread law and attached to a choice-sequence prefix.

Transport preserves that order. The row H relates the listed coordinates componentwise, while C replays the same route through the finite packet. Since the carrier contains no infinite branch, host function equality, choice schedule, selected real value, or ambient completeness coordinate, the prefix-bar boundary is stable under the displayed transport and replay rows. □

Theorem 3209.3 (SpreadSpace NameCert obligations)**.** *For every accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier $S = (B, L, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$ *surface consists exactly of the finite obligations for carrier admission, spread-law prefix acceptance, choice-sequence prefix compatibility, StreamName window replay, RegSeqRat readback, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations are rows in* BHist, hsame, Cont, Pkg, *and* NameCert*; they do not assert completed branch existence, host equality of functions, countable choice, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the tuple displayed in Definition 4189.1. Each coordinate has a single obligation role: prefix carrier, law admission, branch-prefix compatibility, finite stream window, regular rational readback, terminal real seal, transport, replay, provenance, or local naming.

The first three rows determine the spread-space boundary. The rows W, R, E then expose the L10-facing naming route only after that boundary has been displayed.

The structural rows H, C, P, N do not add mathematical objects beyond the listed surface. They transport, replay, source, and name the same finite route, so every local certificate obligation is exhausted by BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 3209.4 (SpreadSpace StreamName–Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{SpreadSpace}^{\uparrow} \longmapsto \mathsf{BrouwerSpread}^{\uparrow} \longmapsto \mathsf{SpreadLaw}^{\uparrow} \longmapsto \mathsf{ChoiceSequence}^{\uparrow} \longmapsto \mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the packet. The handoff exports no completed spread branch, no host function equality, no countable-choice schedule,*

no selected real representative, no quotient stream equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from prefix-bar stability, Definition 4189.2. It forces every downstream name read through the finite prefix route B, L, Q before a StreamName window can be opened.

Apply the obligation inventory of Definition 4189.3. The rows W, R, E are the only StreamName, RegSeqRat, and Real-facing coordinates of the local packet, and the rows H, C, P, N only preserve and name that route.

The displayed chain is therefore the whole handoff available to a Real-facing consumer. None of the excluded objects is a coordinate of the carrier, so the handoff remains a finite NameCert route rather than a completed branch or ambient Real-completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SpreadSpace}^{\uparrow}$ packet is generated from a BaireSpace finite-prefix row, a SpreadLaw admission row, a ChoiceSequence branch-prefix row, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4189.1, Definition 4189.2, Definition 4189.3, and Definition 4189.4 seed the finite spread-space route from Brouwer spread prefixes to StreamName and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed infinite branch existence, host equality of functions, countable choice, selected real representatives, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, spread-law prefix acceptance, choice-sequence compatibility, StreamName window replay, RegSeqRat readback, Real seal handoff, transport replay, provenance, and non-escape for BEDC.Derived.SpreadSpaceUp.

Chapter 3210

A Concrete Naming Certificate for $\mathsf{UniformCauchyEmbedding}^{\uparrow}$

Following Bishop and Bridges on uniformly continuous Cauchy data, the $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ packet records the finite certificate surface for embedding one displayed Cauchy presentation into another without selecting host limits. It sits between the Cauchy source discipline of chapter 3413, the finite schedule rows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the represented-space carrier boundary of chapter 1494. The packet is an open Real and Completion phase object: it makes uniform Cauchy embeddings available as finite NameCert data while leaving completion, quotient equality, and chosen limits outside the carrier.

Definition 3210.1 (Uniform Cauchy embedding carrier)**.** *A* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, S, R, T, E, V, H, K, P, N)$$

with the following displayed rows:

1. *C is the source* $\mathsf{Cauchy}^{\uparrow}$ *row, recording the Cauchy admissibility request whose observations are to be embedded.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row opened by a target tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the source schedule.*
4. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *schedule row, with its tolerance request bounded by a displayed uniform modulus row.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row reached only after the source Cauchy row, the source schedule, the regular readback, and the target schedule have been displayed.*
6. *V is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *representation row for the source and target observation surfaces.*
7. *H stores componentwise* hsame *transport for $C, S, R, T, E, V, K, P, N$.*
8. *K stores displayed* Cont *replay from a target tolerance request back through the source Cauchy, source schedule, regular readback, target schedule, Real seal, and represented-space rows.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, host real equality, a selected completion point, a global inverse map, countable choice, or a proof that the represented-space target is complete.

Lean4 (definition) `BEDC.Derived.UniformCauchyEmbeddingUp.uniformCauchyEmbeddingTasteGate`
Lean4 ✓ `BEDC.Derived.UniformCauchyEmbeddingUp.UniformCauchyEmbeddingTasteGate single carrier alignment`

Theorem 3210.2 (Uniform Cauchy embedding NameCert obligations)**.** *For every accepted* UniformCauchyEmbeddi *carrier*

$$U = (C, S, R, T, E, V, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the source Cauchy row, source schedule, regular rational readback, target schedule, terminal Real seal, represented-space row, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every target tolerance request is replayed through* $C, S, R, T, E, V, H, K, P, N$ *in that order.*

No obligation in the packet is discharged by quotient Cauchy classes, host real equality, selected completion points, global inverse data, countable choice, or completeness of the represented-space target.

Proof. Carrier admission is projection from Definition 4190.1. The tuple displays the Cauchy source, source schedule, regular rational readback, target schedule, Real seal, represented-space, transport, replay, provenance, and naming rows.

The replay order is fixed by the row dependencies. A target tolerance request first opens the source Cauchy row C, then the finite schedule row S, and then the regular-rational readback row R before the target schedule T can be read.

The terminal rows consume only that displayed source surface. The Real seal E records the target real-name boundary, while V records the represented-space observation surface on both sides of the embedding.

Finally H, K, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded quotient, host, choice, inverse, completion, or completeness data, the listed obligations exhaust the local NameCert surface. □

Theorem 3210.3 (Uniform Cauchy embedding sibling route)**.** *Every accepted* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *packet factors through the finite sibling route*

$$\mathsf{Cauchy}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RepresentedSpace}^{\uparrow}.$$

The first $\mathsf{StreamName}^{\uparrow}$ *row schedules source observations, and the second* $\mathsf{StreamName}^{\uparrow}$ *row schedules target observations under the displayed uniform replay row. The route exports finite embedding data only; it does not export quotient Cauchy classes, selected representatives, global inverse data, or target completeness.*

Proof. Apply the obligations of Definition 4190.2. They require every target tolerance request to pass through the source Cauchy row, source schedule, regular rational readback, target schedule, Real seal, and represented-space row.

The two schedule rows have distinct roles. The source schedule S exposes the observations that can be read from the Cauchy source, while the target schedule T records the target request after the displayed uniform replay has bounded it.

The Real and represented-space rows then seal the route without adding non-displayed structure. Because H, K, P, N only transport, replay, source, and name the same rows, the packet exports the finite sibling route and none of the excluded quotient, representative, inverse, or completeness data. □

Theorem 3210.4 (Uniform Cauchy embedding open phase use)**.** *In the open Real and Completion phase, a* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *packet may be used as the finite handoff object from regular Cauchy source data to Real-facing and represented-space-facing consumers. Such a consumer may read the displayed Cauchy, schedule, regular-rational, target-schedule, Real-seal, represented-space, transport, replay, provenance, and naming rows, but it may not infer completion of* $\mathsf{Real}^{\uparrow}$*, a quotient construction for Cauchy names, a selected limit, or countable-choice data.*

Proof. The handoff object is the carrier of Definition 4190.1. Its rows expose exactly the source Cauchy request, the two finite schedules, the regular rational readback, the Real seal, the represented-space surface, and the structural transport, replay, provenance, and naming data.

The open phase use reads this finite surface as an interface for later Real and Completion obligations. It can therefore pass uniform Cauchy embedding evidence downstream without changing the carrier into a completion theorem.

The exclusions follow from absence of coordinates. Since the carrier contains no quotient construction, selected limit, completeness proof, or countable-choice row, no open phase consumer obtains those data from the $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ packet is generated from a source Cauchy row, source schedule, regular rational readback, target schedule, terminal Real seal, represented-space row, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4190.1, Definition 4190.2, Definition 4190.3, and Definition 4190.4 seed the finite uniform Cauchy embedding interface for the open Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient Cauchy equality, selected completion points, global inverse data, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform replay non-escape, sibling route coverage, and open phase use for the same packet.

Chapter 3211

A Concrete Naming Certificate for $\mathsf{DecimalStream}^{\uparrow}$

Following Bishop and Bridges on decimal presentations of constructive real numbers, the $\mathsf{DecimalStream}^{\uparrow}$ packet records the finite base-ten stream surface that sits between decimal prefix data and real-name readback. It refines the finite prefix discipline of chapter 1417 without asking $\mathsf{DecimalExpansion}^{\uparrow}$ to carry an infinite stream role, and it routes later reads through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DecimalReal}^{\uparrow}$, and $\mathsf{SignedDigitStream}^{\uparrow}$ surfaces of chapter 126, chapter 291, chapter 301, chapter 38, chapter 2058, and chapter 316.

Definition 3211.1 (DecimalStream carrier)**.** *A* $\mathsf{DecimalStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, S, W, L, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *prefix row, carrying admitted signed decimal prefixes and their endpoint policy.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, exposing only finite digit windows requested by a downstream consumer.*
3. *W is the decimal window ledger, recording observed base-ten digits, place indices, and prefix bounds for each request in S.*
4. *L is the place-value ledger that evaluates every displayed window as a rational base-ten partial sum.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row that translates the place-value ledger into dyadic tolerance data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the dyadic comparison windows.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DecimalReal}^{\uparrow}$ *seal boundary, with the* $\mathsf{SignedDigitStream}^{\uparrow}$ *sibling used only as a comparison surface for finite-window stream behavior.*
8. *H records componentwise* hsame *transport for E, S, W, L, Q, R, A and the local structural rows.*
9. *C records displayed* Cont *replay from a digit-window request through the decimal prefix, stream schedule, place-value ledger, dyadic comparison, regular-rational handoff, and terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DecimalReal}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite decimal object, quotient stream equality, a selected decimal expansion at endpoints, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DecimalStreamUp.DecimalStreamTasteGate single carrier alignment`

Theorem 3211.2 (DecimalStream NameCert obligations)**.** *For every accepted* $\mathsf{DecimalStream}^{\uparrow}$ *carrier* $D = (E, S, W, L, Q, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$ *surface consists exactly of carrier admission, finite digit-window exposure, base-ten place-value ledger evaluation, dyadic comparison handoff, regular-rational readback, Real and DecimalReal seal nonescape, signed-digit sibling comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DecimalStreamUp.DecimalStreamCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4191.1. The rows E, S, W, L, Q, R, A are the only decimal-prefix, stream-window, place-value, dyadic, regular-rational, and real-facing rows available to the packet.

Finite digit-window exposure is forced by S and W. A downstream read can inspect only the requested base-ten windows, so no infinite decimal stream, hidden tail, or endpoint selector appears.

The place-value ledger L then turns each window into rational partial-sum data. The dyadic row Q and regular-rational row R transport that ledger into the same $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route used by the L10 real boundary.

Finally, A, H, C, P, N seal, transport, replay, source, and locally name the route. Since the carrier lacks quotient stream equality, selected endpoint expansions, host real equality, countable choice, and ambient completeness coordinates, the listed rows exhaust the NameCert obligations. □

Theorem 3211.3 (DecimalStream real readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DecimalReal}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalStream}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The comparison with $\mathsf{SignedDigitStream}^{\uparrow}$ *is only a finite-window sibling comparison. The readback exports no infinite decimal stream, no quotient of streams, no selected endpoint expansion, no host equality of real representatives, and no* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted packet and project the rows of Definition 4191.1. A real-facing read must inspect the decimal prefix row E, the stream schedule S, and the requested window ledger W before place-value data are available.

The place-value ledger L is then compared through Q and handed to R. Thus the Real-facing route is the finite dyadic and regular-rational route already used by $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers.

The terminal seal row A records only the $\mathsf{Real}^{\uparrow}$ and $\mathsf{DecimalReal}^{\uparrow}$ boundary reached by that route. The $\mathsf{SignedDigitStream}^{\uparrow}$ sibling is cited for finite-window comparison behavior and does not add a second stream quotient.

Transport and replay through H and C preserve the same rows, while P and N source and name them. No excluded infinite stream, quotient, endpoint selector, host equality, or completeness coordinate is present, so the displayed boundary is exact. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalStream}^{\uparrow}$ packet is generated from DecimalExpansion prefix rows, finite StreamName digit windows, place-value ledgers, DyadicRatCore comparison rows, RegSeqRat handoff rows, RealUp and DecimalRealUp seal rows, SignedDigitStream sibling comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4191.1, Definition 4191.2, and Definition 4191.3 seed the finite decimal stream surface from decimal prefixes to dyadic, regular-rational, and real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an infinite decimal stream object, quotient stream equality, selected endpoint expansions, host real equality, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite digit-window admission, place-value ledger exactness, dyadic comparison handoff, RegSeqRat readback, Real-seal nonescape, transport replay, provenance, and local naming for BEDC.Derived.DecimalStreamUp.

Chapter 3212

A Concrete Naming Certificate for MinimalTriggerOrbitClassification$^{\uparrow}$

The MinimalTriggerOrbitClassification$^{\uparrow}$ packet realizes Definition 9462.1681 as finite BEDC NameCert data. It sits after the DNA boundary-tile source of chapter 3393, the lift-fiber packet of chapter 4129, the homology-core packet of chapter 4137, and the window coverage algebra of chapter 3478. The packet records nine normal forms, defect rows, quotient and codon counts, transport, replay, provenance, and local naming as displayed finite rows; it is not a biological stochastic model, hidden codon semantics, unrestricted orbit enumeration, or an unrecorded homology computation.

Definition 3212.1 (MinimalTriggerOrbitClassification carrier)**.** *A* MinimalTriggerOrbitClassification$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, N, D, Q, K, L, M, H, C, P, R)$$

with the following displayed rows:

1. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *source row for the six-coordinate boundary-tile window.*
2. *N is the nine-normal-form ledger, displaying the orbit representatives of the minimal-trigger classification.*
3. *D is the defect ledger for the partition, single-overlap, double-overlap, and triple-overlap rows.*
4. *Q is the quotient-count row, preserving the* 49 *quotient-level minimal full triggers.*
5. *K is the codon-count row, preserving the two-lift total* 294*.*
6. *L is the* MinimalTriggerLiftBundle$^{\uparrow}$ *handoff row for lift-fiber decomposition.*
7. *M is the* MinimalTriggerHomologyCore$^{\uparrow}$ *handoff row for support-nerve core reads.*
8. *H records componentwise* hsame *transport for the source, normal-form, defect, count, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from boundary-tile windows through normal forms, defect rows, quotient counts, codon counts, lift fibers, and homology-core handoffs.*
10. *P is the* Pkg *provenance over* MinimalTriggerOrbitClassification$^{\uparrow}$*,* DnaWindowSixBoundaryTile$^{\uparrow}$*,* MinimalTriggerLiftBundle$^{\uparrow}$*,* MinimalTriggerHomologyCore$^{\uparrow}$*,* WindowCoverageTriggerAlgebra$^{\uparrow}$*, the source classification theorem,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *R is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}$ *row.*

The carrier has no coordinate for an empirical genetic-code law, a stochastic fitness model, quotient codon semantics outside the displayed rows, an unrestricted orbit search, or an external homology oracle.

Lean4 ✓ `BEDC.Derived.MinimalTriggerOrbitClassificationUp.MinimalTriggerOrbitClassificationTasteGate single carrier alignment`

Theorem 3212.2 (MinimalTriggerOrbitClassification NameCert obligations)**.** *For every accepted* $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *carrier* $O = (B, N, D, Q, K, L, M, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbit}}$ *surface consists exactly of boundary-tile source admission, nine-normal-form exposure, defect-ledger exposure, quotient-count preservation, codon-count preservation, lift-bundle handoff, homology-core handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer preserving the packet must preserve the nine normal forms, the quotient count* 49*, and the codon count* 294 *through those rows.*

Proof. Project the carrier of Definition 4192.1. The row B supplies the boundary-tile source, while N, D, Q, K display the normal forms, defects, quotient count, and codon count.

The lift and homology consumers are the rows L and M. They can read the classification only after the finite normal-form and count rows have been displayed, so neither handoff becomes an independent orbit enumeration.

Transport and replay are carried by H and C. These rows preserve the same displayed source, normal-form, defect, count, lift, and homology data componentwise under hsame and Cont.

Finally, P and R source and name exactly this finite packet. Since no excluded empirical, stochastic, quotient-semantics, unrestricted-search, or external-homology coordinate is present, the listed rows exhaust the NameCert obligations and force preservation of the nine forms and the counts 49 and 294. □

Theorem 3212.3 (MinimalTriggerOrbitClassification vision handoff)**.** *The finite classification of Definition 9462.1681 is consumed by* $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *only through the route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow} \longmapsto N, D, Q, K \longmapsto \mathsf{MinimalTriggerLiftBundle}^{\uparrow},\ \mathsf{Min}$$

Future Lean TasteGate files for this packet must provide explicit Nontrivial $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *and* FieldFaithful $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *obligations before the packet can serve as a verified trigger-classification consumer.*

Proof. Start from Definition 4192.2. It places the boundary-tile and coverage-algebra source rows before any normal-form or count row can be consumed.

The middle block N, D, Q, K is the finite classification ledger. It records the nine normal forms, their defect classes, the quotient count, and the codon count without opening an unrestricted orbit search.

The handoff rows L and M attach lift-fiber and homology-core consumers to the same ledger. The structural rows H, C, P, R then transport, replay, source, and locally name the route.

Thus the source theorem is represented as a finite packet. The stated Lean TasteGate obligations are downstream verification requirements for this carrier and do not add fields to the paper-side packet. □

Falsifiable prediction: For every accepted $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ packet, componentwise hsame transport preserves exactly nine displayed normal forms, the quotient-level count 49, and the codon-level count 294.

Independence witness: dna-window-six-boundary-tile, minimal-trigger-lift-bundle, minimal-trigger-homology-core, window-coverage-trigger-algebra: $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries normal-form rows, defect rows, quotient-count rows, codon-count rows, lift and homology handoffs, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ packet is generated from a DnaWindowSixBoundaryTile source row, nine normal-form rows, defect ledgers, quotient-count and codon-count rows, MinimalTriggerLiftBundle and MinimalTriggerHomologyCore handoff rows, WindowCoverageTriggerAlgebra provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4192.1, Definition 4192.2, and Definition 4192.3 seed the finite orbit-classification packet preserving nine normal forms, defect rows, quotient count 49, and codon count 294.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MinimalTriggerOrbitClassificationUp.MinimalTriggerOrbitClassificationTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not claim biological mechanism, stochastic fitness, quotient codon semantics outside displayed rows, unrestricted orbit enumeration, Lean verification, Nontrivial or FieldFaithful TasteGate instances, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, normal-form preservation, defect-ledger preservation, quotient-count preservation, codon-count preservation, lift and homology handoffs, transport replay, provenance, local naming, Nontrivial MinimalTriggerOrbitClassificationUp, and FieldFaithful MinimalTriggerOrbitClassificationUp for BEDC.Derived.MinimalTriggerOrbitClassificationUp.

Chapter 3213

A Concrete Naming Certificate for $\mathsf{Portmanteau}^{\uparrow}$

Following Billingsley and Prokhorov, the $\mathsf{Portmanteau}^{\uparrow}$ packet records the weak-convergence test surface that becomes visible only after tight probability data have already been exposed. It sits after the probability source of chapter 184, the pushforward distribution row of chapter 186, the event-surface measure row of chapter 92, the real comparison boundary of chapter 38, the tight probability handoff of chapter 4008, and the compact metric test surface of chapter 129. The chapter names a finite BEDC test packet for bounded-continuous probes; it does not assert weak compactness, choose subsequences, use countable choice, export host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3213.1 (Portmanteau carrier)**.** *A* $\mathsf{Portmanteau}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (P, D, M, T, B, R, A, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the probability surface.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *row whose pushforward reads supply the tested laws.*
3. *M is the* $\mathsf{Measure}^{\uparrow}$ *event-surface row through which measured test preimages are inspected.*
4. *T is the* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *handoff row already exposing tightness data before any weak-convergence test is read.*
5. *B is the bounded-continuous test row, recorded as a displayed* BoundedContinuousTest *surface over the compact or metric test carrier.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row for the two finite test expectations or inequalities observed by the packet.*
7. *A is the transport row aligning probability, distribution, measure, tightness, bounded-continuous test, and real-comparison reads.*
8. *E is the finite endpoint or tolerance ledger that records which bounded-continuous probe and real comparison window are being inspected.*
9. *H records componentwise* hsame *transport for every displayed row.*

10. *C records displayed* Cont *replay of source admission, pushforward readback, measure inspection, tightness exposure, bounded-continuous test evaluation, real comparison, and tolerance readback.*
11. *Q is the* Pkg *provenance over* $\mathsf{Portmanteau}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProkhorovTightProbability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$ *row.*

The carrier has no coordinate for weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PortmanteauUp.PortmanteauTasteGate single carrier alignment`

Theorem 3213.2 (Portmanteau NameCert obligations)**.** *For every accepted* $\mathsf{Portmanteau}^{\uparrow}$ *carrier* $W = (P, D, M, T, B, R, A, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$ *surface consists exactly of six obligation groups:*

1. *carrier admission for the probability source, distribution pushforward, event-surface measure, tightness handoff, bounded-continuous test, real comparison, transport, tolerance, replay, provenance, and local naming rows;*
2. *tightness-before-testing exposure, requiring each weak-convergence test read to display the* ProkhorovTightProbabil *row T before the bounded-continuous probe is consumed;*
3. *bounded-continuous test admission, requiring the probe row B to be read through the displayed compact or metric test surface rather than through a host function space;*
4. *real comparison exactness, requiring the comparison row R to be read through* $\mathsf{Real}^{\uparrow}$ *and the tolerance ledger E;*
5. *distribution and measure compatibility, requiring D and M to name the same tested event or pushforward observation under the transport row A;*
6. *non-escape from weak compactness, selected subsequences, countable choice, host measures, quotient events, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No additional NameCert obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4193.1. The tuple lists exactly the probability, distribution, measure, tightness, bounded-continuous test, real comparison, transport, tolerance, replay, provenance, and naming rows.

The tightness side is already finite. A consumer must inspect T before the probe row B can be evaluated, so the weak-convergence test is downstream of the Prokhorov tightness handoff rather than a source of compactness by itself.

The comparison side is real-name data. The row R and tolerance ledger E provide the displayed real comparison, while D, M, A keep pushforward and measured-event observations on the same route.

Since no carrier coordinate stores weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the six displayed groups exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{Portmanteau}^{\uparrow}$ test read exposes $\mathsf{ProkhorovTightProbability}^{\uparrow}$, a bounded-continuous probe row, and a $\mathsf{Real}^{\uparrow}$ comparison row before any weak-convergence consumer can inspect it.

Independence witness: probspace, distribution, measure, prokhorov-tight-probability, real, compactmetric: the packet is not bijective to any listed sibling because it binds tight probability data, bounded-continuous test admission, real comparison, transport, replay, provenance, and local NameCert rows as one weak-convergence test surface.

Closure status for $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$.

Constructive story: A $\mathsf{Portmanteau}^{\uparrow}$ packet is generated from ProbSpaceUp source rows, DistributionUp pushforward rows, MeasureUp event-surface rows, ProkhorovTightProbabilityUp handoff rows, bounded-continuous test rows, RealUp comparison rows, transport, hsame, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4193.1 and Definition 4193.2 seed the weak-convergence bounded-continuous test surface after tight probability data have been displayed.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak compactness, selected subsequences, countable choice, host measures, quotient events, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tightness-before-testing exposure, bounded-continuous test admission, real comparison exactness, distribution and measure compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.PortmanteauUp.

The $\mathsf{MonotoneClass}^{\uparrow}$ top-level route sends generated closure through chapter 92, chapter 3384, and chapter 3668 before any measure-facing consumer reads the packet.

Chapter 3214

A Concrete Naming Certificate for $\mathsf{MonotoneClass}^{\uparrow}$

Following Halmos and Dynkin-style monotone class arguments, the $\mathsf{MonotoneClass}^{\uparrow}$ packet records the closure object that measure-theoretic generation uses before any consumer may cite a sigma-algebra completion. It sits after the measurable-event carrier of chapter 92, the generated-set boundary of chapter 3384, and the displayed membership discipline of chapter 3668. The packet names generator admission, increasing-union rows, decreasing-intersection rows, measurable-event admission, transport, replay, provenance, and local naming; it does not assert a host sigma-algebra, pi-lambda theorem, selected countable chain, measure extension, or quotient equality of subsets.

Definition 3214.1 (MonotoneClass carrier)**.** *A* $\mathsf{MonotoneClass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, U, I, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the generator family over* BHist*, read as displayed subset rows through* $\mathsf{SetLike}^{\uparrow}$ *and as generated-set boundary data through* $\mathsf{BorelSet}^{\uparrow}$ *when the source is Borel-facing.*
2. *U records increasing-union closure rows for finite observed chains and their named limit event request.*
3. *I records decreasing-intersection closure rows for finite observed chains and their named limit event request.*
4. *A records measurable-event admission through the* $\mathsf{BorelSet}^{\uparrow}$ *and* $\mathsf{Measure}^{\uparrow}$ *sibling surfaces.*
5. *H stores componentwise* hsame *transport for G, U, I, A and the structural rows.*
6. *C stores displayed* Cont *replay from a generator request through increasing-union closure, decreasing-intersection closure, measurable-event admission, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{MonotoneClass}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through G, U, I, A, H, C, P, N. It has no coordinate for completed sigma-algebras, chosen countable unions, host subset equality, measure extension, or an ambient monotone-class theorem.

Theorem 3214.2 (MonotoneClass NameCert obligations)**.** *For the carrier of Definition 4194.1, the local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$ *surface consists exactly of generator admission through* G*, increasing-union closure through* U*, decreasing-intersection closure through* I*, measurable-event admission through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*. Every accepted* $\mathsf{Measure}^\uparrow$*,* $\mathsf{BorelSet}^\uparrow$*, or* $\mathsf{SetLike}^\uparrow$ *consumer that reads a monotone-class conclusion must expose those rows before it treats the generated family as closed under the requested operation.*

Proof. Project the carrier $M = (G, U, I, A, H, C, P, N)$. The generator row G is the only entry point for displayed set requests, so carrier admission is exhausted by the generator family and its sibling $\mathsf{SetLike}^\uparrow$ or $\mathsf{BorelSet}^\uparrow$ boundary.

Read the closure rows next. The row U supplies increasing-union closure for the named observed chain, and I supplies decreasing-intersection closure for the named observed chain. The measurable-event admission row A is the only route from those closures to the $\mathsf{BorelSet}^\uparrow$ and $\mathsf{Measure}^\uparrow$ surfaces.

Finally replay the structural rows. Transport through H, continuation through C, provenance through P, and local naming through N preserve the same finite route. Since no carrier coordinate contains completed sigma-algebras, selected countable chains, quotient subset equality, measure extension, or an ambient monotone-class theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 3214.3 (MonotoneClass generated closure route)**.** *Every accepted* $\mathsf{MonotoneClass}^\uparrow$ *packet exports the generated closure route*

$$G \longmapsto U, I \longmapsto A \longmapsto H, C, P, N$$

for downstream $\mathsf{Measure}^\uparrow$*,* $\mathsf{BorelSet}^\uparrow$*, and* $\mathsf{SetLike}^\uparrow$ *consumers. The route lets a consumer cite monotone closure of the displayed generator family only after increasing-union, decreasing-intersection, measurable-event, transport, replay, provenance, and local-name rows have been exposed; it exports no sigma-algebra completion, pi-lambda closure, measure-extension theorem, selected chain, or quotient equality of subsets.*

Proof. Start with a displayed generator request in G. The request is a finite BEDC row, not a host subset family, and it can be read only through the named $\mathsf{SetLike}^\uparrow$ and $\mathsf{BorelSet}^\uparrow$ surfaces.

Pass through the two closure rows. Increasing requests are handled by U, decreasing requests by I, and both remain attached to the same displayed generator family. The admission row A then records when the resulting event request is visible to the Borel and measure siblings.

Replay the route through H, C, P, N. These rows transport, replay, source, and name the same finite certificate. Therefore a downstream consumer receives exactly the generated closure route stated above and no completed sigma-algebra, pi-lambda theorem, measure-extension theorem, selected chain, or quotient subset equality. □

Theorem 3214.4 (MonotoneClass generator induction boundary)**.** *Every accepted* $\mathsf{MonotoneClass}^\uparrow$ *consumer reaches generated closure only by finite induction over the displayed generator row* G*, increasing-union row* U*, decreasing-intersection row* I*, measurable-event row* A*, transport row* H*, replay row* C*, provenance row* P*, and local name row* N*. The induction boundary cannot be replaced by a host pi-lambda theorem, selected countable chain, measure extension, completed sigma-algebra, or quotient subset equality.*

Proof. Unfold the carrier $M = (G, U, I, A, H, C, P, N)$ from Definition 4194.1. The only initial cases available to a consumer are displayed generator requests in G, and each request is a finite BEDC row rather than a host subset family.

For the induction steps, use the closure route of Definition 4194.3. Increasing-union steps are read through U, decreasing-intersection steps through I, and measurable-event admission through A; each step keeps the same displayed generator family attached.

Apply the NameCert obligations of Definition 4194.2. Transport through H, replay through C, provenance through P, and local naming through N are required before the generated closure can be consumed downstream.

Since the carrier has no coordinate for a pi-lambda theorem, selected countable chain, measure extension, completed sigma-algebra, or quotient equality of subsets, no consumer can substitute one of those host objects for the finite induction route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$.

Constructive story: A $\mathsf{MonotoneClass}^\uparrow$ packet is generated from displayed set generators, increasing-union rows, decreasing-intersection rows, measurable-event admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$, seed)

Scope closed: Definition 4194.1, Definition 4194.2, and Definition 4194.3 seed the monotone closure route over displayed generators, union rows, intersection rows, measurable-event admission, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sigma-algebra completion, pi-lambda closure, measure extension, selected countable chains, quotient subset equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, increasing-union closure, decreasing-intersection closure, measurable-event admission, transport, provenance, and no-escape for BEDC.Derived.MonotoneClassUp.

The $\mathsf{MonotoneClass}^\uparrow$ dependency route exposes its generator, increasing-union, and decreasing-intersection rows as consumers of the measurable-event carrier in chapter 92, the generated-set boundary in chapter 3384, and the membership classifier discipline in chapter 3668. This route keeps the monotone-class closure packet visible to measure-facing consumers before they read any generated event as measurable.

Theorem 3214.5 (MonotoneClass SetLike measure dependency route)**.** *Every* $\mathsf{Measure}^\uparrow$ *or* $\mathsf{BorelSet}^\uparrow$ *consumer that reads an accepted* $\mathsf{MonotoneClass}^\uparrow$ *closure packet must first route the generator row* G*, increasing-union row* U*, decreasing-intersection row* I*, measurable admission row* A*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$ *row* N *through the* $\mathsf{SetLike}^\uparrow$ *membership classifier of chapter 3668. Thus the consumer may use monotone closure only as a displayed membership-stable generated family over chapter 92 and chapter 3384, not as a host sigma-algebra completion, selected countable chain, measure-extension theorem, or quotient equality of subsets.*

Proof. Start with the carrier of Definition 4194.1. The generator row G names displayed subset requests, so any measure-facing or Borel-facing read must classify membership through the $\mathsf{SetLike}^\uparrow$ surface before it can treat the request as a generated event.

The closure rows U and I keep increasing-union and decreasing-intersection reads attached to that same membership classifier. The measurable admission row A is then the only route by which the generated family becomes visible to the $\mathsf{BorelSet}^\uparrow$ and $\mathsf{Measure}^\uparrow$ siblings.

Finally read the structural rows H, C, P, N. They transport, replay, source, and locally name the same finite route, so the consumer cannot replace the displayed membership-stable packet by a host sigma-algebra, a selected chain, a measure extension, or quotient subset equality. □

Chapter 3215

A Concrete Naming Certificate for $\mathsf{FanCompact}^{\uparrow}$

Following Brouwer and Bishop–Bridges, the $\mathsf{FanCompact}^{\uparrow}$ packet records constructive compactness of binary spread data by finite bars and prefix induction rather than by open-cover compactness. It sits beside the Boolean and list carriers of chapter 32 and chapter 36, the finite stream-window discipline of chapter 126, the compact-metric surface of chapter 129, and the downstream uniform-modulus consumer of chapter 369. The packet supplies a finite-means bridge from binary prefix rows to compact-uniform-continuity arguments without importing countable choice, host tree sets, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3215.1 (FanCompact carrier)**.** *A* $\mathsf{FanCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, L, T, W, R, I, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *alphabet row, recording the two branch symbols available at each binary prefix.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *prefix row, recording finite binary words, prefix extension, and prefix concatenation as finite rows.*
3. *T is the finite binary prefix tree row admitted by the packet, with each node read only through B and L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite prefix observed before a branch consumer reads the stream.*
5. *R is the bar-cover row, recording a finite* ProbeBundle *of prefix tests whose success covers every admitted stream route.*
6. *I is the finite-bar induction ledger, recording the leaf tests, backward prefix step, and root conclusion used by the fan argument.*
7. *H records componentwise* hsame *transport for $B, L, T, W, R, I, C, P, N$.*
8. *C records displayed* Cont *replay from a binary-prefix request through the stream window, bar row, induction ledger, compact-metric handoff, and uniform-modulus consumer boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{FanCompact}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, L, T, W, R, I, H, C, P, N$. No row stores a countable-choice selector, a host tree set, an open-cover compactness theorem, a selected infinite branch outside the displayed stream windows, or completeness of $\mathsf{Real}^\uparrow$.

Lean4 ✓ `BEDC.Derived.FanCompactUp.FanCompactTasteGate single carrier alignment`

Theorem 3215.2 (FanCompact finite bar induction)**.** *For an accepted* $\mathsf{FanCompact}^\uparrow$ *carrier $F = (B, L, T, W, R, I, H, C, P, N)$, every public bar-coverage read factors through the finite route*

$$B \longmapsto L \longmapsto T \longmapsto W \longmapsto R \longmapsto I.$$

The induction ledger I is the only row that may turn finite prefix coverage into a root conclusion, and the resulting handoff can be consumed by $\mathsf{CompactMetric}^\uparrow$ *and* $\mathsf{UniformModulus}^\uparrow$ *only through the displayed replay row C and provenance row P.*

Lean4 ✓ `BEDC.Derived.FanCompactUp.FanCompactCarrier finite bar induction`

Proof. Project the carrier. The binary alphabet row B and finite prefix row L are the only sources from which the tree row T can be read.

The stream window row W fixes the observed prefix before any bar test is applied. The bar row R is therefore a finite $\mathsf{ProbeBundle}$ over displayed prefixes, not a host set of infinite branches.

The ledger I supplies the leaf checks and the backward prefix step. Applying it produces the root bar conclusion for the admitted prefix tree while keeping every step attached to B, L, T, W, R.

Finally C and P route the result to compact-metric and uniform-modulus consumers. Since the excluded objects are not coordinates of F, the route exports no countable choice, host tree set, open-cover compactness, selected branch outside the window rows, or $\mathsf{Real}^\uparrow$ completeness. □

Theorem 3215.3 (FanCompact NameCert obligations)**.** *For the carrier of Definition 4195.1, the local* $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$ *surface consists exactly of carrier admission for $F = (B, L, T, W, R, I, H, C, P, N)$, Boolean alphabet exposure through B, finite prefix exposure through L, binary tree admission through T, finite stream-window exposure through W, bar-cover exposure through R, finite-bar induction through I, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 4195.1. The packet exposes exactly the ten rows $B, L, T, W, R, I, H, C, P, N$.

Read B, L, T, W, R, I in order. Boolean symbols and finite lists determine the prefix rows, the stream window fixes the observed branch segment, the bar row lists the finite tests, and the induction ledger supplies the only permitted root conclusion.

Transport, replay, provenance, and local naming are then read through H, C, P, N. These rows preserve the finite prefix route and do not add countable choice, host tree sets, open-cover compactness, Real completeness, or selected infinite branches. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$**.**

Constructive story: A $\mathsf{FanCompact}^\uparrow$ packet is generated from Boolean alphabet rows, finite binary prefix rows, a finite prefix tree, stream-name windows, a finite bar-cover ProbeBundle, a finite-bar induction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$, seed)

Scope closed: Definition 4195.1, Definition 4195.2, and Definition 4195.3 seed the finite fan-compactness certificate surface for compact-metric and uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, countable choice, host tree sets, open-cover compactness, selected infinite branches outside displayed stream windows, or completeness of $\mathrm{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite bar-cover exposure, induction-ledger exactness, stream-window routing, compact-metric handoff, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FanCompactUp.

Chapter 3216

A Concrete Naming Certificate for $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$

Following Tait and Girard reducibility candidates, the $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ packet records a finite candidate-mediated normalization boundary for MetaCIC closed terms. It reads the audit surface of chapter 9383 through the existing normalization audit packet of chapter 3779, the critical-path ledger of chapter 765, and the normalization frontier of chapter 892. The packet names a bounded fuel row, closed-term row, candidate row, Pi and application adequacy rows, no-infinite-reduction row, transport, replay, provenance, and local naming; it does not assert global strong normalization for every closed MetaCIC term.

Definition 3216.1 (BoundedNormalizationCandidate carrier)**.** *A* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, E, A, \Pi, S, I, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed MetaCIC term row, recording the term, type, context boundary, and closedness evidence read by the audit map.*
2. *F is the bounded fuel row, recording the finite reduction budget and the visible step counter for this candidate route.*
3. *E is the reducibility-candidate evidence row, recording the candidate, closed-candidate, or candidate-set membership object consumed by the route.*
4. *A is the application adequacy row, recording the function-side Pi evidence, argument evidence, and application-candidate handoff.*
5. *Π is the Pi adequacy row, recording binder-side candidate data and neutral-normal evidence when a Pi route is inspected.*
6. *S is the strong-normalization projection row, downstream of E, A, Π and bounded by F.*
7. *I is the no-infinite-reduction projection row, downstream of S and still attached to the same candidate evidence.*
8. *R is the frontier-refusal row, recording requests for evidence-free closed-term normalization, unrestricted subject reduction, global confluence, quotient syntax equality, or host evaluator soundness.*

9. *H records componentwise* hsame *transport for $T, F, E, A, \Pi, S, I, R, C, P, N$.*
10. *C records displayed* Cont *replay from a normalization-audit request through the bounded candidate route.*
11. *P is the* Pkg *provenance over* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$*, chapter 9383,* MetaCICNormalizationAuditPacket $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F, E, A, \Pi, S, I, R, H, C, P, N$. It has no coordinate for unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Lean4 ✓ `BEDC.Derived.BoundedNormalizationCandidateUp.BoundedNormalizationCandidateTasteGate single carrier alignment`

Theorem 3216.2 (BoundedNormalizationCandidate NameCert obligations)**.** *For the carrier of Definition 4196.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (T, F, E, A, \Pi, S, I, R, H, C, P, N)$, closed-term exposure through T, bounded fuel exposure through F, candidate evidence through E, application adequacy through A, Pi adequacy through Π, candidate-mediated strong-normalization projection through S, no-infinite-reduction projection through I, frontier refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The closed term row T and bounded fuel row F give the finite normalization request, while E is the only row carrying candidate evidence.

Read the adequacy rows A and Π as refinements of the same candidate route. They do not create a second normalization source; they only describe how applications and Pi-shaped terms remain inside the candidate-mediated boundary.

The strong-normalization projection S is downstream of E, A, Π, F, and the no-infinite-reduction projection I is downstream of S. Requests that drop this evidence pass to the frontier row R.

Transport, replay, provenance, and local naming through H, C, P, N preserve these rows. Since the excluded global principles are absent from the carrier, no NameCert projection can export them. □

Theorem 3216.3 (BoundedNormalizationCandidate frontier handoff)**.** *Every public normalization, strong-normalization, or no-infinite-reduction read from an accepted* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ *carrier factors through the finite route*

$$T \longmapsto F \longmapsto E \longmapsto (A, \Pi) \longmapsto S \longmapsto I \longmapsto R.$$

Thus the packet realizes the finite candidate, fuel, closed-term, Pi/application adequacy, and no-infinite-reduction frontier rows used by chapter 9383, without claiming global strong normalization or closing the subject-reduction and confluence boundaries.

Proof. Start with a public normalization read. The closed term row T records the inspected term, and the fuel row F records the finite budget, but neither row alone supplies strong normalization.

The read must pass through E. Candidate, closed-candidate, or candidate-set evidence is then refined by application adequacy A and Pi adequacy Π before the strong-normalization projection S is available.

The no-infinite-reduction projection I reuses the output of S. If a consumer asks for evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency, the frontier row R keeps that request outside the exported packet.

The structural rows H, C, P, N replay and name this same route for downstream MetaCIC audit consumers. Therefore the handoff is a bounded candidate-mediated frontier object, not a global theorem about all closed MetaCIC terms. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ packet is generated from a closed MetaCIC term row, bounded fuel row, reducibility-candidate evidence, application adequacy, Pi adequacy, candidate-mediated SN projection, no-infinite-reduction projection, frontier refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4196.1, Definition 4196.2, and Definition 4196.3 seed the finite bounded candidate route for MetaCIC normalization audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, global strong normalization, evidence-free closed-term normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Upgrade path: Obligation closure requires checked carrier admission, fuel-route exactness, candidate evidence exposure, Pi and application adequacy routing, SN projection, no-infinite-reduction projection, frontier refusal, transport, replay, provenance, and local naming for BEDC.Derived.BoundedNormalizationCandidateUp.

Chapter 3217

A Concrete Naming Certificate for $\mathsf{ConstructiveBaireThread}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{ConstructiveBaireThread}^{\uparrow}$ packet records the choice-free complete-metric thread used when L10 Real and Completion consumers read a nested dense-open construction through displayed finite data. It sits after the choice-free Baire-category route of chapter 4123, the complete-metric service of chapter 128, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Specker boundary of chapter 1406. The packet names the thread as finite NameCert data and does not add countable choice, a host generic point, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3217.1 (Constructive Baire thread carrier)**.** *A* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, K, S, R, E, P, H, C, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *source row, exposing dense-open requests only as displayed finite obligations.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, carrying the nested-ball, radius, and limit-service data visible to the thread.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that schedules finite dense-open requests, nested-ball inspections, and Cauchy-window reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for centers, radii, and tolerance comparisons consumed by the thread.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Baire, complete-metric, stream, and regular-rational rows have been displayed.*
6. *P is the* $\mathsf{SpeckerSequence}^{\uparrow}$ *boundary row used by consumers that test the thread against a non-located or non-choice escape route.*
7. *H stores componentwise* hsame *transport for B, K, S, R, E, P and the structural rows.*
8. *C stores displayed* Cont *replay from a thread request through the dense-open row, complete-metric service, finite window, regular readback, Real seal, Specker boundary, transport, provenance, and local naming rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $B, K, S, R, E, P, H, C, N$. It has no coordinate for a countable-choice selector, a host generic point, an infinite intersection object, quotient equality of real names, a selected Specker limit, or ambient completeness outside the displayed complete-metric service.

Lean4 (definition) `BEDC.Derived.ConstructiveBaireThreadUp.ConstructiveBaireThreadUp`

Theorem 3217.2 (Constructive Baire thread NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier $T = (B, K, S, R, E, P, H, C, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *row, the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{SpeckerSequence}^{\uparrow}$ *boundary row, componentwise transport, displayed replay, and local naming. Every L10 Real or Completion consumer of the thread must read through these rows.*

Lean4 ✓ `BEDC.Derived.ConstructiveBaireThreadUp.ConstructiveBaireThreadCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4197.1. The analytic rows are B, K, S, R, E, P, and the structural rows are H, C, N.

The Baire and complete-metric rows are read before any real-facing endpoint. The row B supplies only the choice-free dense-open obligations, while K supplies the displayed nested Cauchy service used by the thread.

The finite schedule and readback rows then control the observation route. The row S records the finite windows, R records the regular rational centers and radii, and E is available only after those rows have been displayed.

Finally P, H, C, N preserve the same boundary. Since the carrier has no selector, host generic point, infinite intersection object, quotient real equality, selected Specker limit, or ambient completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3217.3 (Constructive Baire thread nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or Specker-facing consumer of an accepted* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireCategoryChoiceFree}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{SpeckerSequence}^{\uparrow}$$

The route exports a displayed thread for L10 Real and Completion consumers, but it does not export countable choice, a chosen generic point, quotient equality of real names, a selected Specker limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the named rows.*

Proof. Start with an accepted carrier $T = (B, K, S, R, E, P, H, C, N)$. The dense-open side is exactly B, so the thread reads the Baire obligation through the choice-free packet rather than through an arbitrary sequence of choices.

The complete-metric side is exactly K. It carries the displayed Cauchy-thread service, and the finite schedule S determines which dense-open and nested-ball observations can be replayed.

The real-facing endpoint is downstream of regular rational readback. The row R packages the centers, radii, and tolerance comparisons before E exposes the terminal real-name seal.

The Specker boundary P is read only as a refusal surface for hidden located-limit or choice escape. Transport through H, replay through C, and naming through N preserve the same finite route, so the consumer cannot reach any excluded selector, quotient, host generic point, or ambient completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBaireThread}^{\uparrow}$ packet is generated from a BaireCategoryChoiceFree source row, a CompleteMetric Cauchy-thread row, a finite StreamName schedule, a RegSeqRat readback, a RealUp seal, a SpeckerSequence boundary row, componentwise hsame transport, displayed Cont replay, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4197.1, Definition 4197.2, and Definition 4197.3 seed the choice-free complete-metric thread consumed by L10 Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected generic points, host infinite intersections, quotient real equality, selected Specker limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open scheduling, complete-metric thread service, regular-rational readback, Real sealing, Specker nonescape, transport, replay, and local naming for the same finite packet.

Chapter 3218

A Concrete Naming Certificate for $\mathsf{Mollifier}^{\uparrow}$

Following Schwartz and Sobolev constructive analysis, the $\mathsf{Mollifier}^{\uparrow}$ packet records the finite smooth-kernel surface used when a weak or distributional object is regularized through displayed local data. It sits after the Schwartz-test carrier of chapter 3399, the Sobolev ledger of chapter 440, the weak-derivative handoff of chapter 2210, the distribution-derivative action route of chapter 3751, the continuous-map boundary of chapter 130, the terminal real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The packet names only the finite kernel, support-radius, normalization, convolution-window, transport, replay, provenance, and local naming rows; it does not import host integration, choice, quotient function equality, or ambient completeness.

Definition 3218.1 (Mollifier carrier)**.** *A* $\mathsf{Mollifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, N, P, C, H, L)$$

with the following displayed rows:

1. *S is the smooth-kernel row, naming the Schwartz-test or smooth compact-support profile together with the visible derivative-order windows consumed by regularization.*
2. *R is the support-radius row, recording the dyadic radius, local support inclusion, and finite locality budget.*
3. *N is the normalization row, recording the unit-mass ledger as displayed* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons rather than as a host integral.*
4. *P is the convolution-window row, recording the finite source window, target window, translation request, and scale request read by a Sobolev, weak-derivative, distribution-derivative, or continuous-map consumer.*
5. *C stores displayed* Cont *replay from a regularization request through the kernel, support, normalization, convolution-window, real, dyadic, and local naming rows.*
6. *H stores componentwise* hsame *transport for S, R, N, P, C, L and the sibling rows named by the request.*
7. *L is the local* $\mathsf{NameCert}_{\mathsf{Mollifier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, R, N, P, C, H, L. It has no coordinate for host integration, a selected approximate identity, quotient equality of functions or distributions, completed test-function topology, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MollifierUp.MollifierTasteGate single carrier alignment`

Theorem 3218.2 (Mollifier NameCert obligations)**.** *For every accepted* $\mathsf{Mollifier}^{\uparrow}$ *carrier* $M = (S, R, N, P, C, H, L)$*, the local* $\mathsf{NameCert}_{\mathsf{Mollifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed smooth-kernel row, support-locality row, normalization ledger, convolution-window route, nonescape boundary, componentwise transport, displayed replay, provenance inherited from the named analysis siblings, and local naming. Every Sobolev, weak-derivative, distribution-derivative, continuous-map, real, or dyadic consumer of the packet must read through these rows.*

Proof. Carrier admission is projection from Definition 4198.1. The analytic rows are S, R, N, P, and the structural rows are C, H, L.

The support-locality and normalization obligations are separate displayed rows. The radius row R confines every kernel read to the named local window, while N exposes only the finite real and dyadic ledger needed for unit normalization.

The convolution route is the row P. It records the source window, target window, translation request, and scale request before any downstream consumer may treat the packet as a regularizing kernel.

Finally C, H, L replay, transport, and name the same rows. Since no host integral, selected approximate identity, quotient function equality, completed test-function topology, choice principle, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local certificate surface. □

Theorem 3218.3 (Mollifier convolution regularization route)**.** *Let* $M = (S, R, N, P, C, H, L)$ *be an accepted* $\mathsf{Mollifier}^{\uparrow}$ *carrier. Any Sobolev-facing or weak-derivative-facing regularization read reaches a convolution output only through the displayed finite route*

$$S \longmapsto R \longmapsto N \longmapsto P \longmapsto C \longmapsto H \longmapsto L.$$

The same route is the only route by which a distribution-derivative or continuous-map consumer may cite the mollifier packet. The consumer receives smooth-kernel, support-radius, normalization, convolution-window, $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, transport, replay, provenance, and local naming data, but it receives no host integration operator, no choice-selected approximate identity, no quotient equality of functions or distributions, and no ambient completeness theorem.*

Proof. Start with a Sobolev-facing or weak-derivative-facing regularization request. The request first reads the kernel row S and the support-radius row R, so the smooth profile and its locality budget are displayed before any convolution output is named.

The normalization row N is then read as a finite real and dyadic ledger. It supplies the unit-mass boundary needed by the packet without adding a host integral or an external measure-space completion.

The convolution-window row P fixes the source window, target window, translation request, and scale request. Distribution-derivative and continuous-map consumers use the same row, so their reads cannot bypass the displayed kernel and radius data.

Transport through H, replay through C, and local naming through L preserve that finite route. The excluded host integration, selected approximate identity, quotient equality, choice, and ambient completeness objects are not carrier rows, so they cannot be exported by the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}$**.**

Constructive story: A $\mathsf{Mollifier}^{\uparrow}$ packet is generated from a smooth-kernel row, support-radius row, normalization ledger, convolution-window row, displayed Cont replay, componentwise hsame transport, provenance through the named analysis siblings, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4198.1, Definition 4198.2, and Definition 4198.3 seed the finite regularization route consumed by Sobolev, weak-derivative, distribution-derivative, continuous-map, Real, and dyadic rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host integration, a selected approximate identity, quotient equality of functions or distributions, completed test-function topology, choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support locality, normalization ledger exactness, convolution-window routing, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MollifierUp.

Chapter 3219

Arzela–Ascoli Family Compactness NameCert Construction

Following Arzela, Ascoli, and Bishop–Bridges constructive compactness, the $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ packet records family compactness for equicontinuous continuous-map families as finite BEDC NameCert data. It sits between the Arzela–Ascoli carrier of chapter 412, the equicontinuity family surface of chapter 413, the compact metric source of chapter 129, the continuous-map graph surface of chapter 130, and the uniform-modulus consumer of chapter 369. The packet names the finite family-compactness route; it does not assert an ambient compact-open topology, open-cover compactness, countable choice, selected infinite subsequence choice, or host function equality.

Definition 3219.1 (Arzela–Ascoli family-compactness carrier)**.** *An* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, E, K, G, M, W, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArzelaAscoli}^{\uparrow}$ *row naming the compact-continuity packet over source nets, target completion rows, finite family graphs, pointwise image rows, and handoff ledgers.*
2. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row naming the shared finite-center equicontinuity modulus for the family.*
3. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite probe bundle and compact-metric coverage supply the source windows.*
4. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family graph row, storing each displayed family member as graph and pointwise-modulus data.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, receiving only the shared finite-radius and tolerance rows extracted from E and K.*
6. *W is the family window row, listing the finite source probes, family indices, target observations, and diagonal readback windows visible to the compactness consumer.*
7. *B is the bounded image row, retaining the pointwise finite image bundles required before any family-compactness read is exported.*
8. *H records componentwise* hsame *transport for Arzela–Ascoli, equicontinuity, compact-metric, continuous-map, uniform-modulus, window, bounded-image, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a family-compactness request through A, E, K, G, M, W, B.*
10. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, E, K, G, M, W, B, H, C, P, N$. It has no coordinate for compact-open topology, host function extensionality, selected subsequence choice, countable choice, open-cover compactness, or ambient completion outside the displayed rows.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFamilyCompactnessUp.ArzelaAscoliFamilyCompactnessTasteGate`
`single carrier alignment`

Theorem 3219.2 (Arzela–Ascoli family-compactness NameCert obligations)**.** *For every accepted* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ *carrier*

$$F = (A, E, K, G, M, W, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, Arzela–Ascoli packet exposure, equicontinuity-family exposure, compact-metric source-window exposure, continuous-map graph exposure, uniform-modulus handoff, finite family-window readback, bounded-image retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4199.1. The rows $A, E, K, G, M, W, B, H, C, P, N$ are the only readable coordinates of the packet.

The rows A and E supply the compact-continuity and equicontinuity-family surfaces. The compact-metric row K supplies finite source probes, the continuous-map row G supplies graph and pointwise-modulus reads, and the uniform-modulus row M receives only the shared finite tolerance handoff.

The family-window row W and bounded-image row B keep the diagonal and image-bundle reads visible before any compactness consumer can cite the packet. These rows are finite BEDC histories rather than host-level choices of a subsequence or function-space cover.

Finally H, C, P, N transport, replay, source, and name exactly that inventory. Since the carrier has no compact-open, function-extensionality, selected-subsequence, countable-choice, open-cover, or ambient-completion coordinate, none of those objects can be exported. □

Theorem 3219.3 (Arzela–Ascoli family-compactness finite route)**.** *Every family-compactness consumer of an accepted* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ *packet factors through the sequence of displayed sibling rows*

$$\mathsf{ArzelaAscoli}^{\uparrow}, \quad \mathsf{EquicontinuityFamily}^{\uparrow}, \quad \mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The route reads finite source probes, family graph rows, shared equicontinuity radii, bounded target-image rows, and uniform-modulus handoff data before the local $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *row is available.*

Proof. Start with the obligation theorem, Definition 4199.2. It lists sibling exposure, source-window exposure, graph exposure, handoff, window readback, and bounded-image retention as required rows of the local NameCert surface.

Read the siblings in the displayed order. The Arzela–Ascoli row supplies the compact-continuity packet, the equicontinuity-family row supplies the shared radius ledger, the compact-metric row

supplies finite source probes, the continuous-map row supplies graph observations, and the uniform-modulus row receives the finite tolerance handoff.

The structural rows H, C, P, N preserve this sequence. A consumer therefore reaches family compactness only through the finite row inventory of the carrier and not through an ambient compactness theorem or selected host object. □

Falsifiable prediction: Every accepted $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ consumer read displays ArzelaAscoli, EquicontinuityFamily, CompactMetric, ContinuousMap, UniformModulus, finite family-window, bounded-image, transport, replay, provenance, and local NameCert rows before exporting a family-compactness packet.

Independence witness: arzelaascoli, equicontinuityfamily, compactmetric, continuousmap, uniformmodulus: none is bijective to $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ because the packet jointly records compact-continuity, shared equicontinuity, compact source windows, graph rows, uniform-modulus handoff, family-window readback, bounded-image retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ packet is generated from ArzelaAscoli, EquicontinuityFamily, CompactMetric, ContinuousMap, UniformModulus, finite family-window, bounded-image, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4199.1, Definition 4199.2, and Definition 4199.3 seed the finite family-compactness route for equicontinuous continuous-map families over compact metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compact-open topology, host function extensionality, selected infinite subsequences, countable choice, arbitrary open-cover compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling exposure, compact source-window exposure, continuous-map graph exposure, uniform-modulus handoff, family-window readback, bounded-image retention, transport, replay, provenance, and local naming for BEDC.Derived.ArzelaAscoliFamilyCompactnessUp.

Chapter 3220

SR Route Dependency Audit NameCert Construction

The $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ packet realizes the dependency audit surface for the explicit bundle subject-reduction discharge route of chapter 9374. It reads the subject-reduction route packet of chapter 787, the route-audit packet of chapter 793, the route triangle of chapter 796, and the MetaCIC open-frontier ledger of chapter 762. The packet records which finite rows a downstream preservation consumer must display before treating the discharge route as available.

Definition 3220.1 (SR route dependency audit carrier)**.** *An* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, U, T, F, B, K, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *row naming the explicit bundle discharge route.*
2. *U is the* $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ *row recording the audited four-row payload and socket handoff.*
3. *T is the* $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$ *row separating bundle and obstruction coordinates.*
4. *F is the* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *row naming the open MetaCIC subject-reduction boundary.*
5. *B is the bundle-dependency row, listing the four discharge hypotheses as finite bundle coordinates.*
6. *K is the route-dependency row, listing the same four coordinates through explicit bundle publication.*
7. *O is the obstruction-boundary row stating that the audit packet does not inhabit either route for the closed-CIC substrate.*
8. *H records componentwise* hsame *transport for route, audit, triangle, frontier, dependency, obstruction, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a preservation request through R, U, T, F, B, K, O.*
10. *P is the* Pkg *provenance over* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$, $\mathsf{SubjectReductionRoute}^{\uparrow}$, $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$, $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$, $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{SRRouteDependencyAudit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, U, T, F, B, K, O, H, C, P, N$. It has no coordinate for an inhabited SubjectReductionDischargeBundle*, a discharged dependent-codomain premise, or a closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.SRRouteDependencyAuditUp.SRRouteDependencyAuditTasteGate single carrier alignment`

Theorem 3220.2 (SR route dependency audit obligations)**.** *For every accepted* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *carrier*

$$A = (R, U, T, F, B, K, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SRRouteDependencyAudit}^{\uparrow}}$ *surface consists exactly of carrier admission, subject-reduction route exposure, route-audit exposure, triangle exposure, MetaCIC open-frontier exposure, bundle-dependency visibility, route-dependency visibility, obstruction-boundary retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4200.1. The dependency audit has exactly the route rows R, U, T, F, the dependency rows B, K, the obstruction row O, and the structural rows H, C, P, N.

The subject-reduction route and route-audit rows expose the common four-field discharge payload. The route triangle separates bundle use and obstruction retention, while the open-frontier ledger keeps the MetaCIC subject-reduction boundary visible.

The bundle and route dependency rows are compared only as two publications of the same four finite coordinates. They do not create a new discharge premise and do not inhabit the missing closed-CIC bundle.

Finally H, C, P, N transport, replay, source, and name only these rows. Since the carrier has no coordinate for an inhabited discharge bundle, dependent-codomain discharge, or closed subject-reduction theorem, none of those endpoints can be exported. □

Theorem 3220.3 (SR route dependency obstruction nonescape)**.** *Any preservation-facing consumer of an accepted* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *packet must read the obstruction row O together with the open-frontier row F before it can cite either the bundle dependency row B or the route dependency row K. The audit therefore exposes the discharge route as a finite dependency surface and does not turn it into an available closed-CIC subject-reduction theorem.*

Proof. Use Definition 4200.2. It requires route exposure, route-audit exposure, triangle exposure, open-frontier exposure, and obstruction retention as part of the same local NameCert surface.

The row F names the MetaCIC subject-reduction frontier, and O records that the closed-CIC substrate has not supplied the missing discharge inhabitant. Thus a consumer reading B or K reads dependency accounting, not an endpoint theorem.

The replay and provenance rows preserve this order. They carry the preservation request through the displayed route inventory and cannot erase F or O. □

Falsifiable prediction: A subject-reduction-route consumer refutes $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ if it obtains the bundle route without displaying subject-reduction-route, subject-reduction-route-audit, subjectreductionroutetriangle, and metacic-open-frontier-ledger rows.

Independence witness: subject-reduction-route, subject-reduction-route-audit, subjectreductionroutetriangle, metacic-open-frontier-ledger: none is bijective to $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ because the audit packet jointly records route exposure, audited payload, bundle/obstruction separation, open-frontier retention, dependency rows, transport, replay, provenance, and local naming.

The Lean follow-up must provide Nontrivial SRRouteDependencyAudit$^{\uparrow}$ and FieldFaithful SRRouteDependencyAud instances. The instances must keep $R, U, T, F, B, K, O, H, C, P, N$ as separate displayed fields before any checked theorem cites the packet as a dependency-audit surface.

Closure status for NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$.

Constructive story: An SRRouteDependencyAudit$^{\uparrow}$ packet is generated from subject-reduction route, route-audit, route-triangle, open-frontier, bundle-dependency, route-dependency, obstruction, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$, seed)

Scope closed: Definition 4200.1, Definition 4200.2, and Definition 4200.3 seed the finite dependency audit surface for the bundle subject-reduction discharge route.

Formal verification: FormalStatus(NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not inhabit the four MetaCIC subject-reduction discharge hypotheses, construct a discharge bundle, prove closed subject reduction, give Lean verification, or bridge check the packet.

Upgrade path: Obligation closure requires checked carrier admission, route exposure, route-audit exposure, triangle exposure, open-frontier exposure, bundle and route dependency visibility, obstruction nonescape, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.SRRouteDependencyAuditUp.

Chapter 3221

A Concrete Naming Certificate for $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ packet records the universal property of separated reflection as finite BEDC NameCert data. It reads the separated completion route of chapter 1612, the separated metric classifier of chapter 1411, and the Hausdorff reflection boundary of **??**. The packet is a universal-property surface for factorization through separated targets; it does not introduce quotient objects, choice-selected representatives, host equality of reflected points, or ambient classical Hausdorff completion.

Definition 3221.1 (Separated reflection universal carrier)**.** *A* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, H, F, Q, T, R, P, N)$$

with the following displayed rows:

1. *S is the source premetric or uniform source row whose indistinguishability classifier is being reflected.*
2. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row exposing the separated target comparison used by the reflection.*
3. *H is the* $\mathsf{HausdorffReflection}^{\uparrow}$ *row recording the Hausdorff boundary through which reflected equality is read.*
4. *F is the factorization row: every displayed map from S into a separated target factors through the reflected classifier row.*
5. *Q is the uniqueness row, comparing two displayed factorizations only by the separated target classifier supplied by M and H.*
6. *T records componentwise* hsame *transport for source, classifier, Hausdorff boundary, factorization, uniqueness, provenance, replay, and naming rows.*
7. *R records displayed* Cont *replay from a separated-target request through S, M, H, F, Q, T.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, H, F, Q, T, R, P, N$. It has no coordinate for quotient carriers, selected representatives, host point equality, countable choice, hidden separated completions, or ambient Hausdorff replacement.

Lean4 ✓ `BEDC.Derived.SeparatedReflectionUniversalUp.SeparatedReflectionUniversalTasteGate single carrier alignment`

Theorem 3221.2 (Separated reflection universal NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier*

$$U = (S, M, H, F, Q, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}^{\uparrow}}$ *surface consists exactly of source admission, separated classifier exposure, Hausdorff boundary readback, displayed factorization, uniqueness by separated comparison, componentwise transport, displayed continuation replay, provenance, and local naming through $S, M, H, F, Q, T, R, P, N$.*

Proof. Project the carrier of Definition 4201.1. The rows $S, M, H, F, Q, T, R, P, N$ are the entire public packet.

Read S, M, H first. The source row supplies the pre-reflection indistinguishability data, while the separated metric and Hausdorff rows expose the target comparison used by every separated read.

The factorization and uniqueness rows are then forced through F and Q. Transport, replay, provenance, and naming preserve that same route through T, R, P, N.

Since the carrier has no coordinate for quotient carriers, selected representatives, host equality, choice, hidden separated completions, or ambient Hausdorff replacement, those objects cannot appear in the local certificate. □

Theorem 3221.3 (Separated reflection universal factorization)**.** *Let $U = (S, M, H, F, Q, T, R, P, N)$ be an accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier. Every displayed map from the source row S into a separated target whose comparison is read through M and H factors through the row F, and every consumer of the factorization must read the route*

$$S, M, H \longmapsto F$$

before it can inspect transport, replay, provenance, or local naming.

Proof. Apply the NameCert obligation theorem, Definition 4201.2. It identifies S, M, H, F as the whole factorization-facing prefix of the packet.

The source row is read before the separated target classifier, and the Hausdorff boundary is read before any reflected equality is consumed. The factorization row F therefore has no independent source of map data beyond S, M, H.

The structural rows T, R, P, N only transport, replay, source, and name that route. They cannot add a second factorization object or a choice-selected representative. □

Theorem 3221.4 (Separated reflection universal uniqueness)**.** *For every accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier $U = (S, M, H, F, Q, T, R, P, N)$, two displayed factorizations out of S are publicly equal only through the uniqueness row Q over the separated comparison supplied by M and H. The uniqueness read exports no quotient equality, host point equality, selected representative equality, choice principle, or hidden completion equality.*

Proof. Start from the factorization theorem, Definition 4201.3. It places both displayed factorizations on the same $S, M, H \mapsto F$ route.

Now read the uniqueness row Q. Its classifier is the separated target comparison supplied by M and H, so the equality of factorization witnesses is a displayed separated comparison, not a host equality claim.

Finally use the absence clauses in Definition 4201.1. No row carries quotient equality, selected representatives, choice, or hidden completion equality, so the stated separated uniqueness route exhausts the public claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}$.

Constructive story: A $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ packet is generated from a source premetric or uniform row, a separated metric classifier row, a Hausdorff reflection boundary, displayed factorization, uniqueness by separated comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4201.1, Definition 4201.2, Definition 4201.3, and Definition 4201.4 seed the universal-property route for separated reflection over finite BEDC rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient carriers, selected representatives, host equality of reflected points, countable choice, hidden separated completions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, separated classifier exposure, Hausdorff boundary readback, factorization, uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedReflectionUniversalUp.

The rational interval locator also consumes the diagonal finite-window route of chapter 4080 before its rational lower and upper endpoint rows are exported to Real-facing consumers.

Chapter 3222

A Concrete Naming Certificate for $\mathsf{RationalIntervalLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalIntervalLocator}^{\uparrow}$ packet records the finite interval-locator rows by which a Real-facing consumer asks for rational lower and upper bounds at a displayed tolerance. It sits between the located interval algorithms of chapter 1711 and chapter 3971, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet isolates the rational interval locator as finite BEDC data; it is not a host interval, quotient real representative, trichotomy principle, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 3222.1 (Rational interval locator carrier)**.** *A* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. L_0 *is the rational lower-bound row, exposing the left endpoint candidate read at the requested precision.*
2. U_0 *is the rational upper-bound row, exposing the right endpoint candidate read at the same precision.*
3. I *is the interval-inclusion row, recording that the displayed Real-facing name is bracketed by* L_0 *and* U_0 *through finite rational comparisons.*
4. T *is the tolerance-request row, recording the dyadic or rational tolerance that the locator is asked to meet.*
5. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the tolerance request.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same finite window.*
7. A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, naming the Real-facing read that receives the interval locator output.*
8. H *records componentwise* hsame *transport for lower-bound, upper-bound, inclusion, tolerance, window, readback, real-seal, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay from a tolerance request through* L_0, U_0, I, T, S, R, A.

10. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L_0, U_0, I, T, S, R, A, H, C, P, N$. It has no coordinate for host intervals, quotient-real representatives, trichotomy, countable choice, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3222.2 (Rational interval locator NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint order, tolerance response, locator inclusion, regular-rational readback stability, Real-facing seal stability, componentwise transport, displayed replay, provenance, and ledger non-escape through $L_0, U_0, I, T, S, R, A, H, C, P,$*

Proof. Project the carrier of Definition 4202.1. The lower and upper endpoint rows L_0, U_0, the inclusion row I, the tolerance row T, the stream and readback rows S, R, the real-seal row A, and the structural rows H, C, P, N are the whole packet.

Endpoint order and locator tolerance are read from L_0, U_0, I, T. These rows record the rational bracket and the displayed precision request before any Real-facing consumer can inspect the output.

Readback stability is supplied by S, R, A. The stream window opens the finite observations, the regular rational row replays them as a readback, and the terminal real row seals the same finite route.

Finally H, C, P, N transport, replay, source, and name the displayed rows. Since no further coordinate is present, host intervals, quotient-real representatives, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the ledger. □

Theorem 3222.3 (Rational interval locator Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it exports a rational bracket to a located-interval consumer.*

Proof. Let

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

be an accepted carrier. The tolerance row T supplies the finite precision request, and provenance in P attaches that request to the dyadic rational tolerance ledger.

The stream and regular-rational rows S, R then expose the finite window and readback route. The real-seal row A is downstream of the same rows, so a Real-facing consumer receives the interval only after the L10 route has been displayed.

The located-interval bisection and Bolzano consumers can read L_0, U_0, I only through the structural rows H, C, P, N. Therefore the route factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and cannot pass through host intervals, quotient reals, trichotomy, countable choice, or ambient real completeness. □

Theorem 3222.4 (Rational interval locator dyadic tolerance route)**.** *Every tolerance request for an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier first reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through chapter 291. It then passes through the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before exposing the rational interval bracket L_0, U_0, I to a located-interval consumer.*

Proof. Let

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

be an accepted carrier. Project the tolerance row T and provenance row P. These rows identify the requested precision as a dyadic rational tolerance supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than by host interval arithmetic.

Next project S, R, A. The StreamName window S opens the finite observations for the displayed tolerance, the RegSeqRat row R replays those observations as regular rational data, and the Real seal A receives only that finite readback route.

Finally apply Definition 4202.3. It already factors every Real-facing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the rational bracket rows L_0, U_0, I are exposed only after the dyadic tolerance route has been displayed. □

Theorem 3222.5 (Rational interval locator Real-seal lattice)**.** *Every Real-facing locator read of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

factors through the finite lattice

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before exposing L_0, U_0, I to located-interval and real-separability consumers. The lattice exposes only dyadic tolerance, finite stream windows, regular rational readback, and the terminal Real seal; it refuses host intervals, trichotomy, quotient reals, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real-facing locator read and project the accepted carrier. The tolerance row T and provenance row P attach the requested precision to the $\mathsf{DyadicRatCore}^{\uparrow}$ route, so no host interval object is available before the dyadic ledger is read.

Apply Definition 4202.4. The dyadic request opens the $\mathsf{StreamName}^{\uparrow}$ window row S, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal row A.

Apply Definition 4202.3. That theorem exposes the rational bracket rows L_0, U_0, I only after the same route has been displayed. The structural rows H, C, P, N preserve the route for located-interval and real-separability consumers, and the carrier contains no coordinate for trichotomy, quotient reals, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalLocator}^{\uparrow}$ packet is generated from rational lower-bound and upper-bound rows, interval inclusion, tolerance requests, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 4202.1, Definition 4202.2, Definition 4202.3, and Definition 4202.5 seed the finite rational interval locator route for Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval arithmetic, quotient-real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint order, tolerance response, locator inclusion, readback stability, Real-facing seal stability, transport, replay, provenance, and ledger non-escape for BEDC.Derived.RationalIntervalLocatorUp.

Theorem 3222.6 (Rational interval locator dyadic Real dependency route)**.** *Every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *interval read that is consumed by* $\mathsf{RealSeparability}^{\uparrow}$ *must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the separability consumer can use the locator. The route cites the dyadic ledger of chapter 291, the real-name boundary of chapter 38, and the forward link of Definition 4202.7; it exports no quotient real equality, selected limit, host interval, countable-choice schedule, or ambient completeness.*

Proof. Project the locator carrier of Definition 4202.1. The endpoint rows L_0, U_0, interval row I, tolerance row T, stream window S, readback row R, Real seal A, and structural rows H, C, P, N are the whole accepted interval read.

The dyadic part is visible before the separability handoff. The endpoint and tolerance data are read through L_0, U_0, T and the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance in P, while the finite window and readback rows S, R keep the route inside the StreamName and RegSeqRat corridor.

Now apply the Real-facing route of Definition 4202.3 and the RealSeparability forward link of Definition 4202.7. The seal row A must be displayed before any density or separability consumer can use the bracket, so the locator is consumed only after the $\mathsf{Real}^{\uparrow}$ boundary has been reached.

Since no carrier coordinate names quotient real equality, a selected limit, a host interval object, countable choice, or ambient completeness, the dependency route stops at the displayed dyadic endpoints, tolerance rows, readback row, and Real seal. □

Any subdivision-facing $\mathsf{RationalIntervalLocator}^{\uparrow}$ read must pass through the rational lower and upper bracket rows, the dyadic tolerance ledger, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal before it is consumed by chapter 3873. The route exports only that displayed locator packet to the located-interval subdivision consumer; it does not supply a host interval, selected limit, quotient real representative, countable-choice schedule, or ambient completeness claim. The $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint rows supply the rational lower and upper locator consumed by the regular-Cauchy window lattice of chapter 4060. That consumption is allowed only after the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal rows are visible, so the edge remains a displayed L10 dependency rather than a host interval, quotient real, selected limit, countable-choice schedule, or ambient completeness claim.

Theorem 3222.7 (Rational interval locator RealSeparability forward link)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier reads the rational lower and upper locator rows* L_0, U_0*, the tolerance row* T*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A *of chapter 38 before any Real density read is exported through chapter 1809. The forward link supplies rational enumeration windows and dyadic tolerance rows; it exports no host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient completeness datum.*

Proof. Project the carrier rows of Definition 4202.1. The lower-bound row L_0, upper-bound row U_0, inclusion row I, tolerance row T, stream window S, regular readback R, Real seal A, and structural rows H, C, P, N are the whole locator packet.

First read the locator surface. The rational endpoint rows L_0, U_0 and the inclusion row I give the finite lower and upper bracket used by the downstream density request, while T identifies the dyadic or rational tolerance to be matched.

Next compose with the Real-separability route. The finite window S and readback R provide the rational enumeration surface consumed by chapter 1809, and the seal row A ensures that the density-facing consumer receives the data only after the $\mathsf{Real}^{\uparrow}$ boundary has been displayed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no row stores a host interval, selected dense point, quotient representative, countable-choice schedule, or ambient completeness witness, the forward link is a finite locator-to-separability dependency and nothing beyond that boundary. □

Theorem 3222.8 (Rational interval locator sibling route)**.** *Every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *read that exports a locator packet consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *interval endpoint rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readback rows, and the* $\mathsf{RealSeparability}^{\uparrow}$ *density rows before the locator boundary is available to any downstream consumer. The route records only the finite endpoint, readback, density, transport, replay, provenance, and local naming surface; it exports no host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier rows of Definition 4202.1. The endpoint rows are the displayed lower and upper rational bounds, while the tolerance and inclusion rows determine which dyadic interval request is being named.

Read the sibling rows in order. The dyadic endpoint data come through chapter 291, the finite window and readback route comes through chapter 301, and the density-facing consumer reads through chapter 1809 only after those rows have been displayed.

The structural rows then transport, replay, source, and name the same packet. Since no row contains a host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient Real-completeness witness, the sibling route is exhausted by the listed finite dependencies. □

The $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ route passes through chapter 3491, chapter 3732, chapter 4181, chapter 3948, and chapter 4154 before the child files expose the lamination packet.

Chapter 3223

A Concrete Naming Certificate for HyperbolicGeodesicLamination$^{\uparrow}$

The HyperbolicGeodesicLamination$^{\uparrow}$ packet realizes the boundary-paired noncrossing leaf surface of chapter 9463 as finite NameCert data. It sits after the endpoint route of chapter 3491, the Busemann boundary row of chapter 3732, the visual-boundary read of chapter 4181, the cross-ratio comparison of chapter 3948, and the observer-shadow metric of chapter 4154. The packet names a finite lamination handoff, not a completed geodesic space, quotient boundary, physical foliation, global hyperbolic model, or unlogged field-faithfulness assertion.

Downstream boundary-shadow and observer-expansion consumers read the lamination packet through chapter 3813 or chapter 4154, because the noncrossing ledger supplies the leaf-compatibility row consumed by boundary-shadow routing.

Definition 3223.1 (Hyperbolic geodesic lamination carrier)**.** *A* HyperbolicGeodesicLamination$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, L, N, M, B, V, H, C, P, F, A)$$

with the following displayed rows:

1. *E is the endpoint-pair row, recording the two boundary-facing endpoints of each displayed leaf.*
2. *L is the geodesic-leaf row, attaching each endpoint pair to a finite geodesic-boundary route.*
3. *N is the noncrossing row, recording the finite compatibility ledger among the displayed leaves.*
4. *M is the hyperbolic-metric read row used when leaf separation or ordering is inspected.*
5. *B is the Busemann row, attaching the leaves to displayed boundary distance-difference data.*
6. *V is the visual-shadow row, recording the observer-facing visual shadow through which the lamination is read.*
7. *H records componentwise* hsame *transport for endpoint-pair, leaf, noncrossing, metric, Busemann, visual-shadow, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-lamination request through E, L, N, M, B, V.*
9. *P is the* Pkg *provenance over* HyperbolicGeodesicLamination$^{\uparrow}$*,* HyperbolicGeodesicBoundary$^{\uparrow}$*,* HyperbolicBusemannBoundary$^{\uparrow}$*,* HyperbolicVisualBoundary$^{\uparrow}$*,* HyperbolicCrossRatio$^{\uparrow}$*,* HyperbolicObserverShadowM BHist*,* hsame*,* Cont*, and* NameCert *rows.*

10. *F is the refusal row excluding host global laminations, quotient boundary identifications, physical foliations, hidden crossing tests, and unlogged TasteGate witnesses.*
11. *A is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, N, M, B, V, H, C, P, F, A$. It has no coordinate for completed geodesic spaces, ambient boundary quotients, host noncrossing predicates, physical laminations, or implicit Nontrivial *and* FieldFaithful *witnesses.*

Theorem 3223.2 (Hyperbolic geodesic lamination NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *carrier*

$$G = (E, L, N, M, B, V, H, C, P, F, A),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$ *surface consists exactly of endpoint-pair admission, geodesic-leaf admission, noncrossing ledger compatibility, hyperbolic-metric readback, Busemann attachment, visual-shadow exposure, componentwise transport, displayed replay, provenance, explicit refusal, and local naming through $E, L, N, M, B, V, H, C, P, F, A$.*

Proof. Project the carrier of Definition 4203.1. The endpoint, leaf, noncrossing, metric, Busemann, visual-shadow, transport, replay, provenance, refusal, and naming rows are the whole lamination packet.

The endpoint-pair and geodesic-leaf rows E, L are read first. They attach every displayed leaf to finite boundary-facing data rather than to an ambient set of complete geodesics.

The noncrossing and metric rows N, M then record the compatibility tests available to consumers. These tests are displayed rows, so the packet does not import a host noncrossing predicate.

The Busemann and visual-shadow rows B, V bind the leaf family to boundary and observer-facing routes. Finally H, C, P, F, A transport, replay, source, refuse exits, and name the same finite carrier, leaving completed global laminations, quotient boundaries, physical foliations, and implicit TasteGate witnesses outside the certificate. □

Theorem 3223.3 (Hyperbolic geodesic lamination boundary route)**.** *Every boundary-lamination consumer of an accepted* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *carrier must pass through endpoint-pair, geodesic-leaf, noncrossing, hyperbolic-metric, Busemann, visual-shadow, transport, provenance, refusal, and local* NameCert *rows before it can export a lamination handoff.*

Proof. Let

$$G = (E, L, N, M, B, V, H, C, P, F, A)$$

be an accepted carrier. A boundary-facing read starts with E, L, so its endpoints and leaves are displayed before the consumer sees a lamination token.

The noncrossing row N and metric row M supply the finite compatibility and metric-read tests. The Busemann and visual-shadow rows B, V then attach those tests to the boundary and observer-facing surfaces cited by provenance.

Transport and replay through H, C preserve the displayed route, while P, F, A source, refuse exits, and name it. Hence any boundary-lamination handoff factors through the listed rows and cannot be claimed by a consumer that omits endpoint pairs, noncrossing evidence, hyperbolic metric reads, visual-shadow data, transport, provenance, refusal, or local naming. □

Falsifiable prediction: Any boundary lamination consumer omitting endpoint-pair, noncrossing, hyperbolic-metric, visual-shadow, transport, provenance, or local NameCert rows cannot claim the $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ handoff.

Independence witness: hyperbolic-geodesic-boundary, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-cross-ratio, hyperbolic-observer-shadow-metric: the geodesic-boundary packet names endpoint routes without the noncrossing ledger, the Busemann packet names distance-difference data without leaf compatibility, the visual-boundary packet names boundary readback without lamination rows, the cross-ratio packet compares boundary quadruples without owning geodesic leaves, and the observer-shadow metric packet reads visible shadows without endpoint-pair noncrossing. None is bijective to $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$.

The future `lean4/BEDC/Derived/HyperbolicGeodesicLaminationUp/TasteGate.lean` file must include explicit Nontrivial $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ instances. Those instances preserve $E, L, N, M, B, V, H, C, P, F, A$ as distinct fields before a checked carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ packet is generated from endpoint-pair rows, geodesic-leaf rows, a noncrossing ledger, hyperbolic-metric reads, Busemann attachment, visual-shadow exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, explicit refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4203.1, Definition 4203.2, and Definition 4203.3 seed the finite boundary-paired noncrossing lamination route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed geodesic spaces, quotient boundary equality, physical foliations, host global laminations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint-pair admission, geodesic-leaf admission, noncrossing compatibility, metric readback, Busemann attachment, visual-shadow exposure, TasteGate nontriviality, field faithfulness, refusal, and local naming for BEDC.Derived.HyperbolicGeodesicLaminationUp.

The $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ carrier is anchored in the sibling lattice by the endpoint route of chapter 3491, the Busemann rows of chapter 3732, the visual-boundary readback of chapter 4181, the cross-ratio comparisons of chapter 3948, and the observer-shadow metric data of chapter 4154. A lamination consumer must therefore read endpoint-paired leaves, boundary distance-difference rows, visual shadows, cross-ratio comparison provenance, and observer-shadow metric coordinates as displayed rows before treating the packet as a finite noncrossing lamination handoff.

Chapter 3224

A Concrete Naming Certificate for $\mathsf{BesovSpace}^{\uparrow}$

Following Besov and Triebel's constructive regularity scale, the $\mathsf{BesovSpace}^{\uparrow}$ packet records a finite smoothness-measurement surface between Sobolev weak-gradient data and Holder power-law control. It sits after the Sobolev ledger of chapter 440, the Holder continuity route of chapter 1766, the weak-derivative handoff of chapter 2210, the terminal real-name seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet names only displayed smoothness windows, dyadic scale budgets, weak-action rows, real readback, transport, replay, provenance, and local naming; it does not import open-cover compactness, quotient function spaces, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3224.1 (Besov space carrier)**.** *A* $\mathsf{BesovSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Hd, W, R, Q, D, E, T, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, recording the weak-gradient and integrability ledger consumed by the regularity request.*
2. *Hd is the* $\mathsf{HolderContinuity}^{\uparrow}$ *boundary row, recording the visible power-law modulus data used when the Besov packet exposes pointwise smoothness control.*
3. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *action row, naming the finite test probes and derivative observations inspected by the scale request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational magnitude data attached to the selected smoothness windows.*
5. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, selecting the scale windows, test probes, and comparison requests visible to this packet.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scale, radius, and comparison budget attached to the same windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Hd, W, R, Q, D have been displayed.*
8. *T stores componentwise* hsame *transport for the regularity, window, tolerance, real-seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a Besov regularity request through the Sobolev, Holder, weak-derivative, regular-rational, stream, dyadic, and real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, Hd, W, R, Q, D, E, T, C, P, N$. It has no coordinate for an open-cover compactness theorem, quotient equality of function spaces, selected representatives, a completed smoothness scale, host distribution equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BesovSpaceUp.BesovSpaceTasteGate single carrier alignment`

Theorem 3224.2 (Besov space NameCert obligations)**.** *For every accepted* $\mathsf{BesovSpace}^{\uparrow}$ *carrier $B = (S, Hd, W, R, Q, D, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Sobolev row, Holder row, weak-derivative action row, regular rational readback row, stream-window schedule, dyadic scale ledger, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows. Every regularity, PDE, interpolation, or trace-facing consumer of the packet must read through those rows before it cites the Besov smoothness boundary.*

Proof. Carrier admission is projection from Definition 4204.1. The analytic rows are S, Hd, W, R, Q, D, E, and the structural rows are T, C, P, N.

The Sobolev, Holder, and weak-derivative rows have distinct roles. The Sobolev row supplies the weak-gradient and integrability ledger, the Holder row supplies the visible power-law boundary, and the weak-derivative row supplies the finite test-action observations.

The scale and readback rows then fix the measurable part of the packet. The stream schedule Q opens the finite windows, the dyadic ledger D records the scale and comparison budget, the regular rational row R reads the displayed magnitudes, and the real seal E is reached only after those rows have been inspected.

Finally T, C, P, N transport, replay, source, and name the same finite route. Since no open-cover compactness theorem, quotient function-space equality, selected representative, completed smoothness scale, host distribution equality, countable choice, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3224.3 (Besov space L10 handoff)**.** *Every L10 regularity consumer of an accepted* $\mathsf{BesovSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Sobolev}^{\uparrow} \longrightarrow \mathsf{WeakDerivative}^{\uparrow} \longrightarrow \mathsf{HolderContinuity}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{R}$$

The handoff exports displayed smoothness-window data, dyadic scale control, regular rational readback, terminal real seals, componentwise transport, continuation replay, provenance, and local naming. It exports no open-cover compactness theorem, quotient function space, selected representative, host equality of functions or distributions, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4204.2. They require every accepted regularity read to pass through the Sobolev, Holder, and weak-derivative rows before any downstream smoothness boundary is cited.

Next read the finite scale data. The stream schedule Q chooses the visible windows, while the dyadic ledger D attaches the scale and tolerance budget to those same windows.

The regular rational readback row R and terminal real seal E then report the displayed magnitude data. The structural rows T, C, P, N preserve the route under transport, replay, provenance, and local naming.

Thus an L10 consumer receives exactly the displayed chain from Sobolev and weak-derivative evidence through Holder, stream, dyadic, regular-rational, and real-seal rows. The excluded compactness, quotient, representative-selection, host-equality, choice, and ambient-completeness objects have no carrier coordinate and cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{BesovSpace}^{\uparrow}$ regularity consumer must read Sobolev, HolderContinuity, WeakDerivative, StreamName, DyadicRatCore, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows before citing the Besov smoothness boundary.

Independence witness: sobolev, holder-continuity, weak-derivative, real, streamname, regseqrat, dyadicratcore: $\mathsf{BesovSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries smoothness-source rows, pointwise power-law boundary data, weak-action observations, finite windows, dyadic scale budgets, regular rational readback, real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BesovSpace}^{\uparrow}$ packet is generated from Sobolev, HolderContinuity, WeakDerivative, RegSeqRat, StreamName, DyadicRatCore, Real, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4204.1, Definition 4204.2, and Definition 4204.3 seed the finite constructive regularity-space route over Sobolev, HolderContinuity, WeakDerivative, Real, StreamName, RegSeqRat, DyadicRatCore, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, quotient function spaces, selected representatives, host equality of functions or distributions, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, smoothness-window exactness, dyadic-scale ledger stability, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BesovSpaceUp.

Chapter 3225

A Concrete Naming Certificate for $\mathsf{LebesgueCoveringDimension}^{\uparrow}$

Following Lebesgue and Hurewicz–Wallman, the $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet records covering dimension through finite open-cover refinements with bounded overlap. It sits beside the finite refinement carrier of chapter 1853, the topology row of chapter 88, the metric-ball observation surface of chapter 57, and the finite radius ledger of chapter 452. The packet names only displayed cover rows, refinement rows, and order ledgers; it does not invoke arbitrary open-cover compactness, a selected subcover, or a host dimension theorem.

Definition 3225.1 (Lebesgue covering dimension carrier)**.** *A* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, U, V, R, O, B, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *source row together with the metric observation surface needed to read finite* $\mathsf{MetricBall}^{\uparrow}$ *cover cells.*
2. *U is a finite open-cover request row, displayed as named topology opens and local metric-ball witnesses.*
3. *V is a finite refinement row whose entries are open rows over the same source X.*
4. *R is the refinement-containment ledger, recording for each row of V the named row of U that contains it.*
5. *O is the order-at-most-n overlap ledger, recording that every displayed point-history read meets at most $n + 1$ refinement rows.*
6. *B is the finite ball-inspection row tying the topology opens to* $\mathsf{MetricBall}^{\uparrow}$ *observations.*
7. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius ledger for the displayed finite cover window.*
8. *H records componentwise* hsame *transport for X, U, V, R, O, B, F and for the certificate rows.*
9. *C records displayed* Cont *replay from an open-cover request through finite ball inspection, finite-radius readback, refinement containment, and overlap checking.*
10. *P is the* Pkg *provenance over* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by transporting these displayed rows. It has no coordinate for a host open-cover family, a selected refinement, an external multiplicity oracle, or a classical covering-dimension theorem.

Lean4 ✓ `BEDC.Derived.LebesgueCoveringDimensionUp.LebesgueCoveringDimensionTasteGate single carrier alignment`

Theorem 3225.2 (Lebesgue covering dimension NameCert obligations)**.** *For every accepted* LebesgueCoveringDime *carrier*

$$L = (X, U, V, R, O, B, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$ *surface consists exactly of topology-source admission through X, finite open-cover admission through U, refinement-row admission through V, refinement containment through R, order-at-most-n overlap checking through O, metric-ball inspection through B, finite Lebesgue-number radius readback through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4205.1. The topology, cover, refinement, containment, overlap, ball-inspection, finite-radius, transport, replay, provenance, and naming rows are precisely the displayed coordinates.

The finite refinement obligation reads U, V, R, O over the source row X. Containment is stored as a row-to-row ledger, and the order bound is stored as the visible overlap count for finite point-history inspections.

The metric side is only observational. The rows B and F route finite open cells through $\mathsf{MetricBall}^{\uparrow}$ and $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ before a consumer may compare the packet with the sibling $\mathsf{CoveringDimension}^{\uparrow}$ surface.

Finally H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Since no excluded host object occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3225.3 (Lebesgue finite-refinement route)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer of an accepted* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{MetricBall}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto U, V, R, O, H, C, P, N.$$

Thus the dimension assertion is a displayed open-cover refinement with order-at-most-n overlap and finite radius readback. It exports no arbitrary open-cover compactness principle, selected subcover, host refinement theorem, or detached multiplicity oracle.

Proof. Begin with the NameCert obligations of Definition 4205.2. They force a consumer to read the source topology, the finite cover request, the refinement rows, the containment ledger, and the overlap ledger before any dimension claim can be exported.

Read the metric observation side next. The ball row B supplies only the displayed $\mathsf{MetricBall}^{\uparrow}$ inspections, while F supplies only the finite radius ledger already exposed by $\mathsf{FiniteLebesgueNumber}^{\uparrow}$.

The comparison with $\mathsf{CoveringDimension}^{\uparrow}$ is therefore horizontal: both packets share finite cover-refinement and overlap rows, but this chapter keeps the Lebesgue finite-radius route explicit. The structural rows H, C, P, N transport, replay, source, and name that finite route, so no host compactness or selected-refinement object can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ read is refuted if a consumer exports a covering-dimension bound without displaying TopologyUp source rows, finite open-cover rows, refinement-containment rows, order-at-most-n overlap rows, MetricBallUp inspections, FiniteLebesgueNumberUp radius data, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: coveringdimension, finite-lebesgue-number, topology, metric: no listed sibling supplies the whole $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet, because the packet binds finite topology cover rows, metric-ball inspection, finite-radius readback, refinement containment, overlap order, and NameCert provenance in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet is generated from TopologyUp source rows, finite open-cover rows, MetricBallUp observations, FiniteLebesgueNumberUp radius rows, finite refinement-containment ledgers, order-at-most-n overlap ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4205.1, Definition 4205.2, and Definition 4205.3 seed the finite-refinement L10 topology object over TopologyUp, MetricBallUp, CoveringDimensionUp, and FiniteLebesgueNumberUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected subcovers, host dimension theorems, detached multiplicity oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement containment, overlap-order exactness, metric-ball inspection, finite-radius compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueCoveringDim

Chapter 3226

A Concrete Naming Certificate for $\mathsf{LagrangeInterpolation}^{\uparrow}$

Following Lagrange's interpolation formula, the $\mathsf{LagrangeInterpolation}^{\uparrow}$ packet records the finite route from distinct interpolation nodes and displayed values to a polynomial whose evaluations match those values. It consumes the polynomial coefficient-list surface of chapter 50, the matrix-style finite table discipline of chapter 49, and the field and rational arithmetic boundaries of chapter 45 and chapter 37. Its BEDC carrier is a finite BHist ledger: it names nodes, values, basis-polynomial construction, evaluation rows, uniqueness rows, hsame transport, Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row. It is not a host interpolation theorem, a quotient polynomial equality, or an ambient field-completion principle.

The history rows are read as ordinary BEDC certificate data in the sense of chapter 1366 and the typed naming discipline of chapter 9. The classical formula appears only through finite arithmetic ledgers; every consumer sees the displayed nodes, denominators, coefficient rows, and evaluation routes before it can use the interpolating polynomial.

Definition 3226.1 (Lagrange interpolation carrier)**.** *A* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *source is a finite* BHist *packet*

$$L = (N, V, D, B, E, U, H, C, P, Q)$$

with the following visible rows:

1. *N is a nonempty finite node row, with an index list and a displayed pairwise-distinctness ledger for the nodes.*
2. *V is the value row assigning one carried scalar value to each node of N.*
3. *D is the denominator ledger for the factors $x_i - x_j$ with $i \neq j$, routed through* $\mathsf{Field}^{\uparrow}$ *inverse rows and the rational arithmetic surface when the nodes are rational.*
4. *B is the basis-polynomial row, displaying the finite products*

$$\ell_i(X) = \prod_{j \neq i} \frac{X - x_j}{x_i - x_j}$$

   as normalized $\mathsf{Poly}^{\uparrow}$ *coefficient lists.*
5. *E is the evaluation row, recording that each ℓ_i evaluates to the Kronecker pattern on the displayed node list.*

6. *U is the polynomial assembly and uniqueness row: it records the finite sum $\sum_i V_i \ell_i$ and the finite comparison showing that any other polynomial of degree below the displayed node count with the same evaluations has the same normalized coefficient list.*
7. *H records componentwise* hsame *transport for the node, value, denominator, basis, evaluation, uniqueness, replay, provenance, and local-name rows.*
8. *C records the* Cont *routes that replay denominator formation, basis-polynomial multiplication, polynomial evaluation, and uniqueness comparison from the same displayed finite packet.*
9. *P is the* Pkg *provenance over* $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Q is the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *row.*

The carrier predicate is $\mathsf{LagrangeInterpolationLedger}(N, V, D, B, E, U, H, C, P, Q)$. *Two accepted packets classify together exactly when the ten displayed rows are transported by H and replay through the same routes named by C and P.*

Lean4 ✓ `BEDC.Derived.LagrangeInterpolationUp.LagrangeInterpolationTasteGate single carrier alignment`

Theorem 3226.2 (Lagrange interpolation NameCert obligations)**.** *For an accepted* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *packet $L = (N, V, D, B, E, U, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *obligation surface consists exactly of the finite carrier and classifier rows of Definition 4206.1, the denominator nonzero rows D, the basis-polynomial construction rows B, the evaluation rows E, the assembly and uniqueness rows U, the componentwise* hsame *transport H, the displayed* Cont *replay C, the* Pkg *provenance P, and the local naming row Q. No additional host polynomial function, determinant inverse, quotient equality of polynomials, infinite node family, or completion datum is exported.*

Proof. Read the carrier of Definition 4206.1. The rows N and V supply the finite interpolation table, while D records exactly the nonzero denominator data forced by distinct nodes.

The basis and evaluation content is held by B and E. These rows are typed over the same polynomial coefficient-list surface and the same scalar arithmetic provenance, so their comparisons do not introduce a second carrier.

The row U is the only place where the assembled polynomial and the degree-bounded uniqueness comparison are visible. The remaining rows H, C, P, Q transport, replay, source, and name the displayed packet. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 3226.3 (Lagrange polynomial evaluation route)**.** *Every evaluation read exported by an accepted* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *packet factors through the same finite route*

$$N, V, D \longmapsto B \longmapsto E \longmapsto U \longmapsto Q.$$

The route first verifies pairwise-distinct nodes and denominator rows, then builds normalized basis polynomials in $\mathsf{Poly}^{\uparrow}$, *evaluates those basis polynomials at the displayed nodes, assembles the interpolating polynomial, and finally exposes the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *row. A consumer cannot bypass this route by appealing to an ambient polynomial function, a determinant computation over an unstated matrix, or an external field-completion argument.*

Proof. Start with an accepted packet $L = (N, V, D, B, E, U, H, C, P, Q)$. The denominator ledger D is downstream of the node row N and upstream of the basis row B, so each basis-polynomial read has already passed through the displayed distinctness and inverse witnesses.

Next, E evaluates the basis rows of B at the same finite node list. The route is replayed by C and transported by H, hence a transported packet keeps the same node, value, denominator, basis, and evaluation sequence.

Finally, U assembles the finite linear combination and records the degree-bounded uniqueness comparison before Q names the packet. Thus every polynomial-evaluation read available from the local certificate follows the displayed finite route and has no coordinate from which a bypassing host argument could be read. □

Falsifiable prediction: Every accepted $\mathsf{LagrangeInterpolation}^{\uparrow}$ read that claims an interpolating polynomial factors through finite node rows, value rows, denominator ledgers, normalized $\mathsf{Poly}^{\uparrow}$ basis rows, evaluation rows, uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{LagrangeInterpolation}^{\uparrow}$ packet is generated from finite node rows, value rows, denominator ledgers, normalized polynomial basis rows, evaluation rows, uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$, seed)

Scope closed: Definition 4206.1, Definition 4206.2, and Definition 4206.3 seed the finite interpolation carrier, local NameCert obligation surface, and polynomial-evaluation route over displayed nodes, values, denominators, normalized basis polynomials, evaluation rows, uniqueness rows, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host interpolation functions, ambient polynomial functions, determinant inverses outside the displayed ledger, quotient equality of polynomials, infinite node families, completion data, field-completion arguments, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, denominator nonzero routing, denominator nonzero exposure, basis-polynomial construction, evaluation rows, evaluation replay, assembly, uniqueness-row replay, basis and evaluation route transport, transport replay, route nonescape, provenance exposure, local naming, and TasteGate field faithfulness for BEDC.Derived.LagrangeInterpolationUp.

Chapter 3227

A Concrete Naming Certificate for $\mathsf{RieszFischer}^{\uparrow}$

Following Riesz and Fischer, the $\mathsf{RieszFischer}^{\uparrow}$ packet records the finite NameCert surface for the classical completeness route from square-summable coefficient windows to a Hilbert-facing limit object. It sits between the inner-product and norm rows of chapter 86, chapter 91, and chapter 89, and the completion handoff rows of chapter 128 and chapter 1527. The packet exposes the coefficient-window, inner-product, norm, completion, transport, replay, provenance, and local naming rows needed for a constructive Riesz–Fischer handoff; it does not introduce quotient L^2 classes, choice-selected limits, or an ambient Hilbert completeness theorem outside the displayed packet.

Definition 3227.1 (RieszFischer carrier)**.** *A* $\mathsf{RieszFischer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, I, H, N, M, U, T, C, P, L)$$

with the following displayed rows:

1. *W is the square-summable coefficient-window row, recording finite coefficient blocks, tail bounds, and the window schedule used by the packet.*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row that reads coefficient-window comparisons as inner-product estimates.*
3. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row naming the Hilbert-facing endpoint surface over which the coefficient windows are interpreted.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *row induced by the displayed inner-product estimates.*
5. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-service row for the metric sequence determined by W, I, N.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row that packages the displayed Cauchy datum as a completion-facing unit route.*
7. *T records componentwise* hsame *transport for W, I, H, N, M, U and for the local structural rows.*
8. *C records displayed* Cont *replay from coefficient windows through the inner-product, norm, completion, and unit-handoff route.*
9. *P is the* Pkg *provenance over* $\mathsf{RieszFischer}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient L^2 classes, almost-everywhere representatives, countable-choice limit selection, ambient Hilbert completeness, or host equality of limits.

Lean4 ✓ `BEDC.Derived.RieszFischerUp.RieszFischerTasteGate single carrier alignment`

Theorem 3227.2 (RieszFischer NameCert obligations)**.** *For every accepted* $\mathsf{RieszFischer}^{\uparrow}$ *carrier* $R = (W, I, H, N, M, U, T, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the square-summable coefficient windows,* $\mathsf{InnerProduct}^{\uparrow}$ *comparison row,* $\mathsf{Hilbert}^{\uparrow}$ *source row,* $\mathsf{Norm}^{\uparrow}$ *metric row,* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-service row,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row, componentwise transport, replay, provenance, and local naming;*
2. *square-summable exhaustion, requiring every Hilbert-facing completion read to begin with the displayed coefficient-window row* W*;*
3. *inner-product and norm compatibility, requiring* I *and* N *to expose the same finite comparison route before any metric-completion row is consumed;*
4. *completion handoff, requiring* M *and* U *to route the same Cauchy datum and not a selected external limit;*
5. *transport replay, requiring* T, C, P, L *to preserve the ordered route under* hsame*,* Cont*,* Pkg*, and* NameCert*;*
6. *ledger non-escape, excluding quotient* L^2 *classes, choice-selected limits, ambient Hilbert completeness, host equality of limits, and hidden measure-space representatives.*

Proof. Project the accepted carrier. The packet displays the square-summable windows, inner-product estimates, Hilbert-facing source, norm metric, complete-metric service, completion-unit handoff, transport, replay, provenance, and local name as the whole available surface.

Read the analytic route in order. The coefficient window W supplies the finite square-summability data; the rows I and N turn those windows into inner-product and norm comparisons; the row M consumes the resulting Cauchy datum; and U exposes the same datum to completion consumers.

Finally attach the structural rows. Componentwise transport and displayed replay preserve only W, I, H, N, M, U, while P and L source and name the packet. Since no carrier coordinate contains quotient classes, selected limits, ambient completeness, host equality, or hidden representatives, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$, seed)

Scope closed: Definition 4207.1 and Definition 4207.2 seed the finite Riesz–Fischer packet over square-summable coefficient windows, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient L^2 classes, choice-selected limits, ambient Hilbert completeness, host equality of limits, hidden measure-space representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window exhaustion, inner-product and norm compatibility, completion handoff, transport replay, and ledger non-escape for BEDC.Derived.RieszFischerUp.

Constructive story: A $\mathsf{RieszFischer}^{\uparrow}$ packet is generated from finite square-summable coefficient windows, inner-product comparison rows, a Hilbert-facing source row, norm metric rows, a complete-metric Cauchy service, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3228

A Concrete Naming Certificate for $\mathsf{RationalFastLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalFastLimit}^{\uparrow}$ packet records the Bishop-style presentation of a real name as rational observations with an explicit fast tail budget. It sits over the dyadic arithmetic ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real seal of chapter 38, the fast Cauchy real packet of chapter 1589, and the explicit modulus packet of chapter 1781. The carrier makes the rational fast-limit surface available to L10 Real and Completion consumers without quotienting streams, selecting representatives, or importing countable choice.

Definition 3228.1 (RationalFastLimit carrier)**.** *A* $\mathsf{RationalFastLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational observation row, listing the finite rational approximants read by the current precision request.*
2. *D is the fast dyadic tail-budget row, recording the dyadic tolerance scale, comparison budget, and tail witness used to compare later rational observations.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly which rational observations may be opened by the request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from Q, D, S after the fast budget has bounded the visible tail.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rational observation, dyadic budget, stream-window, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for rational observations, dyadic tail budgets, stream windows, regular readback, the Real seal, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite precision request through Q, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalFastLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$ *row.*
The classifier compares accepted packets only through $Q, D, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected rational representatives, a host real limit, trichotomy, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalFastLimitUp.RationalFastLimitTasteGate single carrier alignment`

Theorem 3228.2 (RationalFastLimit NameCert obligations)**.** *For every accepted* $\mathsf{RationalFastLimit}^\uparrow$ *carrier $L = (Q, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$ *surface consists exactly of rational observation admission, fast dyadic tail-budget exposure, finite stream-window service, regular rational handoff, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must read the finite route*

$$Q, D, S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before treating the packet as a real-name boundary.

Proof. Project the carrier of Definition 4208.1. The rows Q, D, S are the only source rows available to a precision request: rational observations, the fast dyadic tail budget, and the finite window schedule.

The regular handoff is downstream of those rows. The dyadic budget D supplies the tolerance comparison used to bound the tail of the rational observations in Q, while S restricts the read to displayed finite windows.

The Real seal row E is reached only after R. The structural rows H, C, P, N transport, replay, source, and name the same displayed route, so no quotient stream equality, selected representative, countable choice, or ambient completeness principle is introduced by the certificate. □

Theorem 3228.3 (RationalFastLimit regular readback)**.** *Every* $\mathsf{RegSeqRat}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *consumer of an accepted* $\mathsf{RationalFastLimit}^\uparrow$ *packet factors through rational observations, the fast dyadic tail budget, finite stream windows, and the regular rational handoff. Equivalently, the consumer route is the displayed sibling chain*

$$\mathsf{DyadicRatCore}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow$$

with $\mathsf{FastCauchyReal}^\uparrow$ *and* $\mathsf{FastCauchyModulus}^\uparrow$ *supplying the fast-limit interpretation of the dyadic budget row. The packet exports no direct rational-stream-to-real shortcut.*

Proof. Begin with an accepted packet $L = (Q, D, S, R, E, H, C, P, N)$. The rational observations are read from Q, but their use as a real-name approximation is constrained by the fast dyadic tail budget D.

The finite window row S determines which observations can be inspected. After Q, D, S are fixed, the handoff row R repacks the same data as a regular rational name, matching the route used by the $\mathsf{RegSeqRat}^\uparrow$ sibling.

The terminal seal E is downstream of R, and the fast interpretation of the budget is exactly the one exposed by $\mathsf{FastCauchyReal}^\uparrow$ and $\mathsf{FastCauchyModulus}^\uparrow$. Since the carrier has no coordinate for a host real limit or a quotient stream, every regular or Real consumer remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$ **.**

Constructive story: A $\mathsf{RationalFastLimit}^\uparrow$ packet is generated from rational observation rows, a fast dyadic tail budget, finite StreamName windows, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalFastLimit}^{\uparrow}}$, seed)

Scope closed: Definition 4208.1, Definition 4208.2, and Definition 4208.3 seed the finite rational fast-limit route into regular rational and Real-facing readback.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalFastLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real limits, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fast tail-budget exactness, finite-window service, regular readback, Real seal nonescape, componentwise stability, provenance, and local naming for the displayed packet.

Chapter 3229

A Concrete Naming Certificate for $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ packet records the finite common-refinement route by which two displayed regular Cauchy names may be inspected through one shared target window before either reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits after the single-schedule reindexing surface of chapter 1568, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names a displayed paired refinement surface; it is not quotient stream equality, not a selected subsequence limit, not countable choice, not host real equality, not trichotomy, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3229.1 (Regular Cauchy common refinement carrier)**.** *A* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two source* $\mathsf{StreamName}^{\uparrow}$ *schedule rows for the accepted regular Cauchy names;*
2. *I is the common target* $\mathsf{StreamName}^{\uparrow}$ *window through which both sources are inspected;*
3. *M_0, Q_0 are the monotone and cofinal reindex rows from I to S_0;*
4. *M_1, Q_1 are the monotone and cofinal reindex rows from I to S_1;*
5. *W is the common target-window budget row, recording that a single displayed finite request feeds both reindex routes;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by the two reads at the budget named by W;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row showing that both inspected names reach the same regular rational tolerance surface;*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R$ have been consumed;*
9. *H records componentwise* hsame *transport for the two schedules, the common target window, both reindex routes, the tolerance, readback, Real seal, continuation, provenance, and local naming rows;*

10. *C records displayed* Cont *replay from a common-window request through the two monotone–cofinal routes and then through the shared tolerance, readback, and Real seal;*
11. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ BEDC.Derived.RegularCauchyCommonRefinementUp.RegularCauchyCommonRefinementTasteGate
single carrier alignment

Theorem 3229.2 (Regular Cauchy common refinement obligation surface)**.** *For every accepted* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier* $K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$ *surface consists exactly of paired source-schedule admission, common-window admission, two monotone–cofinal reindex rows, shared dyadic tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, and ambient Real completeness.*

Proof. Project the carrier of Definition 4209.1. The projection exposes exactly the two source schedules, the common target window, two monotone–cofinal reindex pairs, the shared tolerance row, the regular readback row, the Real seal, and the structural rows H, C, P, N.

The common-window row W is the only target-window coordinate. A consumer therefore cannot inspect the two source schedules by unrelated requests; each request must replay through C and then through both displayed reindex routes.

The dyadic and regular rows are shared downstream rows. Since D and R are read only after the two reindex pairs have been displayed, the carrier records one tolerance surface and one regular readback surface rather than two unrelated limit witnesses.

Finally, A is downstream of the same paired route, while H, P, N preserve and package only the listed rows. No row remains from which to read quotient stream equality, a selected subsequence limit, countable choice, host real equality, trichotomy, or ambient Real completeness. □

Theorem 3229.3 (Regular Cauchy common refinement)**.** *Let S_0 and S_1 be accepted regular Cauchy source schedules with displayed finite schedules, and suppose the two schedules admit displayed monotone and cofinal reindex rows into a common target window I. Then there is an accepted* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier*

$$K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$$

whose $\mathsf{Real}^{\uparrow}$*-facing consumers factor through the same dyadic tolerance row D and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The construction uses only the two displayed schedule rows, the two monotone–cofinal reindex routes, the common target window, finite* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it does not use quotient stream equality, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two accepted regular Cauchy schedules. Each source schedule is available as a finite $\mathsf{StreamName}^{\uparrow}$ row, and each displayed monotone–cofinal route is the same kind of finite route isolated in chapter 1568.

Choose the displayed common target window I as the single inspection surface. The row W records the finite request made at that window, while M_0, Q_0 and M_1, Q_1 replay the request back to S_0 and S_1 without choosing a subsequence or comparing host streams.

Read the shared tolerance and regularity rows next. The $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the tolerance budget attached to W, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is consumed after both reindex routes have been displayed. Hence both source reads reach the same finite regular rational surface before any Real-facing seal is inspected.

Transport and replay close the packet. Componentwise hsame transport identifies the rows that are meant to be shared, and Cont records the two finite route replays from I through the source schedules and forward to D, R, A. The Pkg and local NameCert rows then package exactly this surface.

By Definition 4209.2, the resulting packet contains no coordinate for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, or ambient completeness. Therefore every $\mathsf{Real}^{\uparrow}$-facing consumer of the accepted common-refinement carrier factors through the displayed common window, the shared dyadic tolerance, and the shared regular readback. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ packet is generated from two source StreamName schedules, a common target window, two monotone and cofinal reindex route pairs, one DyadicRatCore tolerance row, one RegSeqRat readback row, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4209.1, Definition 4209.2, and Definition 4209.3 seed the finite paired refinement route from two regular Cauchy schedules to a shared $\mathsf{Real}^{\uparrow}$-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, paired monotone and cofinal reindexing discipline, shared dyadic tolerance preservation, RegSeqRat readback, Real seal factorization, and ledger refusal.

Chapter 3230

A Concrete Naming Certificate for $\mathsf{OlivierTheorem}^{\uparrow}$

Following Olivier's theorem for positive convergent series, the $\mathsf{OlivierTheorem}^{\uparrow}$ packet records the finite positive-term tail evidence by which a Real-facing series consumer may read vanishing terms without importing a host completeness theorem. It sits after the summable-tail modulus of chapter 1799, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a constructive tail handoff for positive series; it exports no selected sum, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3230.1 (Olivier theorem carrier)**.** *An* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, T, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive-term series row, exposing only the inspected nonnegative term windows and finite partial-sum comparison windows used by the current request.*
2. *T is the tail-window row, naming the finite interval of terms whose smallness is to be served.*
3. *M is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row, assigning the requested positive-term tail window to a displayed summable-tail modulus before any regular rational readback is available.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached only after A, T, M have displayed the finite tail evidence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff and the summable-tail route have been replayed.*
6. *H records componentwise* hsame *transport for the positive-term, tail-window, modulus, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a positive-series tail request through A, T, M, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{OlivierTheorem}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for selected sums, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OlivierTheoremUp.OlivierTheoremTasteGate single carrier alignment`

Theorem 3230.2 (Olivier theorem NameCert obligations)**.** *For every accepted* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier* $O = (A, T, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the positive-term series row, finite tail-window row, summable-tail modulus row, regular rational handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *positivity-window exposure, requiring* A *to supply the only term-sign and finite partial-sum data used by the tail request;*
3. *tail-window service, requiring* T *to name the inspected positive-term tail window before any downstream read can ask for vanishing terms;*
4. *modulus handoff, requiring* T *to pass through the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row* M *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *is available;*
5. *non-escape, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to pass through* A, T, M, R, H, C, P, N *and excluding selected sums, quotient series equality, choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4210.1. The visible inventory is $A, T, M, R, E, H, C, P, N$, so the certificate has no detached row for a selected sum, quotient series equality, choice, host real equality, or ambient completeness theorem.

The positive-term and finite-window data are read before any Real-facing seal is available. The row A supplies the inspected positive term and partial-sum comparison windows, while T identifies the concrete tail window served by the request.

The tail window then passes through M. Thus finite positive-term tail evidence is converted into a displayed $\mathsf{SummableTailModulus}^{\uparrow}$ service before any $\mathsf{RegSeqRat}^{\uparrow}$ readback can occur.

Finally R, E, H, C, P, N supply regular readback, terminal sealing, transport, replay, provenance, and local naming for the same route. Since the excluded analytic objects are not carrier coordinates, they cannot be exported by the local certificate. □

Theorem 3230.3 (Olivier theorem tail handoff)**.** *Every Real-facing read of an accepted* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier factors finite positive-term tail windows through the ordered route*

$$\mathsf{OlivierTheorem}^{\uparrow} \longmapsto \mathsf{SummableTailModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently a consumer may use Olivier-style vanishing-term evidence only after the displayed summable-tail modulus row has served the finite tail window. It receives no selected sum, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $O = (A, T, M, R, E, H, C, P, N)$ and a Real-facing request for positive-series tail evidence. The only source row is A, and the only finite tail-window row is T.

The handoff is then forced through M. This makes the Olivier-style conclusion a finite modulus-served tail read, not a hidden selection of a sum or an equality of quotient series.

Only after M appears may the route enter R and then the terminal $\mathsf{Real}^{\uparrow}$ seal E. The rows H, C, P, N preserve the same displayed route under transport, replay it as continuation data, record provenance, and name it locally, so no excluded completeness or choice principle is introduced downstream. □

Falsifiable prediction: Every accepted $\mathsf{OlivierTheorem}^{\uparrow}$ Real-facing tail read factors through positive-term windows, a finite tail-window row, a SummableTailModulus row, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: summable-tail-modulus, regseqrat, real: $\mathsf{OlivierTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds positive-term series windows, finite tail evidence, summable-tail modulus service, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{OlivierTheorem}^{\uparrow}$ packet is generated from positive-term series windows, finite tail-window evidence, a SummableTailModulus row, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4210.1, Definition 4210.2, and Definition 4210.3 seed the finite positive-series tail route over displayed summable-tail modulus, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positivity-window exposure, tail-window service, modulus handoff, regular rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OlivierTheoremUp.

Chapter 3231

A Concrete Naming Certificate for $\mathsf{JacobiIteration}^{\uparrow}$

Following Jacobi's iterative method for linear systems, the $\mathsf{JacobiIteration}^{\uparrow}$ packet records the finite BEDC certificate surface of a displayed diagonal split, residual update, iteration schedule, and contraction-budget handoff. It consumes the matrix and linear-map rows read through $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$, the norm budget of $\mathsf{Norm}^{\uparrow}$, the Cauchy-orbit handoff of chapter 4224, and the fixed-point iteration interface already named by $\mathsf{FixedPointIteration}^{\uparrow}$. The packet names the local row discipline for one displayed iteration scheme; it is not a host convergence theorem, spectral-radius theorem, chosen solution vector, quotient sequence equality, or ambient completeness assertion.

Definition 3231.1 (Jacobi iteration carrier)**.** *A* $\mathsf{JacobiIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, D, R, S, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the matrix/vector system row, exposing the finite linear system and right-hand side only through the consumed* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{LinMap}^{\uparrow}$ *surfaces.*
2. *D is the diagonal split row, displaying the diagonal part and the residual part as finite matrix rows without assuming an external inverse beyond the displayed diagonal update.*
3. *R is the residual update row, recording the deterministic map from the current iterate and the system row to the next displayed iterate.*
4. *S is the iteration schedule row, listing the finite observation windows and the orbit positions that a consumer may inspect.*
5. *B is the displayed norm and contraction-budget row, exposing the finite descent inequalities read through* $\mathsf{Norm}^{\uparrow}$ *and the fixed-point iteration budget.*
6. *H transports A, D, R, S, B, C, P, N componentwise by* hsame*.*
7. *C replays the displayed* Cont *updates from a system read through the diagonal split, residual update, schedule, norm budget, and Cauchy-orbit handoff.*
8. *P records* Pkg *provenance over* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{CauchySequenceCriterion}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, D, R, S, B, H, C, P, N$. It has no coordinate for a host solution vector, unrestricted spectral-radius argument, quotient sequence equality, selected limit, countable choice, or ambient Banach-space completeness.

Lean4 ✓ `BEDC.Derived.JacobiIterationUp.JacobiIterationTasteGate single carrier alignment`

Theorem 3231.2 (Jacobi iteration NameCert obligations)**.** *For every accepted* $\mathsf{JacobiIteration}^{\uparrow}$ *carrier* $J = (A, D, R, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the system, diagonal split, residual update, iteration schedule, contraction budget, transport, replay, provenance, and local naming rows;*
2. *diagonal split exposure, requiring D to be read after the matrix/vector system row A and before any residual update is inspected;*
3. *residual-update determinacy, requiring R to compute the next displayed iterate from the current iterate and the exposed split rows;*
4. *contraction-budget descent, requiring B to record the finite norm comparisons used by the schedule row;*
5. *Cauchy-orbit handoff, requiring the accepted schedule and budget rows to hand only the displayed orbit windows to* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *and* $\mathsf{FixedPointIteration}^{\uparrow}$ *consumers;*
6. *transport and replay preservation through H and C;*
7. *provenance and local naming through P and N;*
8. *non-escape from the displayed packet, excluding host convergence, selected limits, quotient sequence equality, unrestricted spectral-radius reasoning, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 4211.1. The only analytic rows are the system row A, diagonal split D, residual update R, schedule S, and budget row B.

The diagonal split is exposed before residual replay. Thus each accepted update reads the same displayed system and split rows, and the next iterate is determined by the finite residual row rather than by an external solver.

The budget row is also finite. It records the norm comparisons consumed by the schedule, and the schedule hands only those displayed orbit windows to the Cauchy-sequence and fixed-point iteration consumers.

Transport, replay, provenance, and naming preserve the same rows. Since the refused objects have no carrier coordinate in J, the local certificate exports only the displayed Jacobi iteration packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{JacobiIteration}^{\uparrow}$ packet is generated from a displayed matrix/vector system row, a diagonal split row, a residual update row, an iteration schedule, a norm/contraction budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4211.1 and Definition 4211.2 seed the finite Jacobi iteration certificate over matrix, linear-map, norm, Cauchy-sequence criterion, fixed-point iteration, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convergence, spectral-radius convergence, selected solutions, quotient sequence equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal split exposure, residual-update determinacy, contraction-budget descent, Cauchy-orbit handoff, transport, replay, provenance, and local naming for BEDC.Derived.JacobiIterationUp.

Chapter 3232

A Concrete Naming Certificate for $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$

The $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ packet realizes the apophatic socket-stack branch of Definition 9399.7 as finite boundary/refusal NameCert data. It sits over the typed socket taxonomy of chapter 1240, the socket-kind classifier of chapter 1058, the supply ledger of chapter 528, the far-end refusal socket of chapter 1061, and the named externality surface of chapter 543. The packet names a stacked visible-socket boundary; it does not supply the far end, discharge the refusal row, collapse socket kinds, or replace the named boundary by a host predicate.

Definition 3232.1 (Apophatic boundary socket stack carrier)**.** *An* $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, K_D, L, F, A, V, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ *row naming the typed boundary class of each socket in the stack.*
2. *K_D is the* $\mathsf{SocketKindClassifier}^{\uparrow}$ *row classifying the visible socket kind at each stack layer.*
3. *L is the* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row recording which visible supply request is being ledgered.*
4. *F is the* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *row marking the far-end refusal boundary without supplying its content.*
5. *A is the* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the externality position that the stack refuses to internalize.*
6. *V is the visible stack-order row, listing the finite order in which typed socket, classifier, ledger, far-end, and name rows are read.*
7. *R is the refusal row, recording that a consumer may factor through the visible socket and refusal rows but may not replace them by hidden host supply.*
8. *H records componentwise* hsame *transport for $T, K_D, L, F, A, V, R, C, P, N$.*
9. *C records displayed* Cont *replay through the visible stack order and refusal row.*
10. *P is the* Pkg *provenance over* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, K_D, L, F, A, V, R, H, C, P, N$. It has no coordinate for supplied far-end content, untyped sockets, anonymous supply, hidden host predicates, quotient equality, or discharged apophatic refusal.

Lean4 ✓ `BEDC.Derived.ApophaticBoundarySocketStackUp.ApophaticBoundarySocketStackTasteGate single carrier alignment`

Theorem 3232.2 (Apophatic boundary socket stack NameCert obligations)**.** *For every accepted* $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ *carrier $K = (T, K_D, L, F, A, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySo}}$ *surface consists exactly of carrier admission, typed socket exposure, kind-classifier exposure, supply-ledger exposure, far-end refusal exposure, apophatic name exposure, visible stack-order determinacy, refusal-row exactness, componentwise transport, displayed replay, provenance, and local naming. Every accepted stack consumer factors through the visible socket rows T, K_D, L, V and the refusal/name rows F, A, R before any public read is available.*

Proof. Project the carrier of Definition 4212.1. The socket-facing rows are T, K_D, L, V, and the refusal/name rows are F, A, R.

The stack order row V forces those rows to be read in a finite visible order. Thus a consumer cannot inspect the supply ledger or far-end boundary without first carrying the typed socket and classifier rows that identify the visible socket.

The refusal row R is downstream of F and A. It records refusal of internalization, so the public read is a boundary read rather than a supplied far-end witness or a hidden host predicate.

Transport, replay, provenance, and naming preserve the same finite packet. Since the refused objects have no carrier coordinate, the accepted stack factors through exactly the visible socket and refusal rows. □

Falsifiable prediction: Every accepted $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ consumer factors through the visible socket rows T, K_D, L, V and the refusal rows F, A, R; a purported consumer that omits either side is rejected by the local NameCert.

Independence witness: typedgapsockettaxonomy, socketkindclassifier, supplysocketledger, apophaticfarendsocket, apophaticname: TypedGapSocketTaxonomyUp records the typed class without the stack order, SocketKindClassifierUp names one kind row without the far-end refusal, SupplySocketLedgerUp records requests without the apophatic name, ApophaticFarEndSocketUp marks the refusal boundary without the full visible stack, and ApophaticNameUp names the externality without the socket-kind and ledger rows. None is bijective to the full $T, K_D, L, F, A, V, R, H, C, P, N$ carrier under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ packet is generated from typed socket taxonomy, socket-kind classifier, supply-ledger, apophatic far-end socket, apophatic name, visible stack-order, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4212.1 and Definition 4212.2 seed the finite visible-socket and refusal stack certificate over typed socket, classifier, ledger, far-end, name, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ApophaticBoundarySocketStackUp.ApophaticBoundarySocketStackTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close supplied far-end content, hidden host predicates, quotient equality, discharged apophatic refusal, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible socket exposure, stack-order determinacy, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.ApophaticBoundarySocketStackUp.

Chapter 3233

A Concrete Naming Certificate for $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ packet records the finite NameCert surface by which a monotone family of continuous maps on a compact metric source yields a compact-uniform tolerance handoff. It sits beside the Dini continuity packet of chapter 2196, the Dini theorem carrier of chapter 1387, the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-modulus surface of chapter 369, and the uniform-limit handoff of chapter 463. The packet uses finite cover and explicit probe rows; it does not invoke general open-cover compactness, a host supremum, selected minimum radius outside the carrier, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3233.1 (Dini monotone compact-uniform carrier)**.** *A* $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, F, M, L, R, V, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, carrying the metric source, compact finite-cover interface, and finite probe extraction route.*
2. *F is the continuous-map family row, recording the displayed maps on X and their local continuity data.*
3. *M is the monotone ledger, recording the family order and the visible monotone approach toward the limiting row.*
4. *L is the pointwise-limit row, recording the finite probe-wise route from family observations to the displayed limiting map.*
5. *R is the rational tolerance budget, storing the requested positive tolerance, cover-cell slices, and comparison rows.*
6. *V is the finite cover and* $\mathsf{ProbeBundle}$ *row extracted from X for the requested tolerance window.*
7. *U is the uniform-modulus and uniform-limit handoff row. It records the route from local continuity and monotone pointwise data through V and R to a single compact-uniform read.*
8. *H records componentwise* hsame *transport for source, family, monotone ledger, limit, tolerance, cover, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-uniform requests through $X, F, M, L, R, V, U, H, P, N$.*
10. *P is the* Pkg *provenance over* DiniMonotoneCompactUniform$^{\uparrow}$, DiniContinuity$^{\uparrow}$, DiniTheorem$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, UniformLimit$^{\uparrow}$, Rat$^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{DiniMonotoneCompactUniform}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $X, F, M, L, R, V, U, H, C, P, N$ and has no coordinate for host suprema, arbitrary open-cover compactness, quotient function equality, selected radii outside V, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.DiniMonotoneCompactUniformUp.DiniMonotoneCompactUniformTasteGate single carrier alignment`

Theorem 3233.2 (Dini monotone compact-uniform carrier obligations)**.** *For an accepted* DiniMonotoneCompactUn *carrier*

$$D = (X, F, M, L, R, V, U, H, C, P, N),$$

the local NameCert$_{\text{DiniMonotoneCompactUniform}^{\uparrow}}$ *surface consists exactly of compact metric source admission through X, continuous-map family admission through F, monotone visibility through M, pointwise-limit visibility through L, rational tolerance budgeting through R, finite cover and probe extraction through V, uniform-modulus and uniform-limit handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4213.1. The first seven rows are the analytic data, and H, C, P, N are the structural certificate rows.

The monotone Dini route is forced by the displayed order. A compact-uniform request first reads the source and family rows X, F, then exposes monotone visibility through M and probe-wise limiting data through L.

The finite tolerance work is stored in R, V, U. The rational budget R is subdivided across the finite ProbeBundle row V, and the handoff row U records how those local reads assemble into one uniform-modulus and uniform-limit request.

Finally H, C, P, N preserve the same coordinates by transport, replay, provenance, and local naming. Since the excluded host principles have no carrier coordinate, they are not NameCert obligations of the packet. □

Theorem 3233.3 (Dini monotone compact finite-cover extraction)**.** *Every compact-uniform read accepted by a* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier must expose the compact source row X, the continuous-family row F, the monotone ledger M, the pointwise-limit row L, and the rational budget R before extracting the finite* ProbeBundle *cover row V. The extraction is a finite compact metric route; it is not general open-cover compactness, a selected global minimum radius, or a host supremum argument.*

Proof. Use the carrier obligations of Definition 4213.2. They place X, F, M, L, R upstream of the finite cover row V.

The compact source row X supplies the compact metric finite-cover interface. The continuous-map family row F and monotone ledger M determine which local continuity and monotone comparisons can be inspected on each displayed probe.

The pointwise-limit row L is then read only at probes supplied by the packet, and the rational budget R supplies the requested tolerance and cell slices. Hence V is extracted as finite packet data rather than by an external open-cover or supremum principle. □

Theorem 3233.4 (Dini monotone uniform budget handoff)**.** *For an accepted* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier, every consumer of the compact-uniform conclusion must route the requested tolerance through R, the finite probe cover V, and the uniform-modulus and uniform-limit handoff U, then through* hsame, Cont, Pkg, *and local* NameCert *rows. The exported handoff is only the finite BEDC route through* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, UniformLimit$^{\uparrow}$, DiniContinuity$^{\uparrow}$, *and* DiniTheorem$^{\uparrow}$.

Proof. Begin with the finite-cover extraction of Definition 4213.3. It supplies cover cells and probe rows downstream from the compact metric source, continuous family, monotone ledger, pointwise limit, and rational budget.

The handoff row U consumes exactly that finite data. It first reads the local modulus information attached to the cells of V, then packages the assembled tolerance into the displayed uniform-limit route.

The structural rows H, C, P, N export no analytic data beyond that route. Therefore the consumer receives a compact-uniform Dini handoff through the named sibling packets, with no host function equality, hidden completion, selected radius outside the carrier, or open-cover compactness. □

Theorem 3233.5 (Dini monotone compact-uniform finite descent)**.** *For an accepted* DiniMonotoneCompactUniform *carrier*

$$D = (X, F, M, L, R, V, U, H, C, P, N),$$

every compact-uniform tolerance export descends through the pointwise monotone rows M, L on the finite ProbeBundle *cover V and the rational budget R before U may expose a single uniform read. The export cannot introduce a host supremum, selected minimum radius, open-cover compactness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4213.1. The only rows between the compact source and the uniform handoff are the continuous family F, monotone ledger M, pointwise-limit row L, rational budget R, finite cover row V, and handoff row U.

Apply the finite-cover extraction theorem Definition 4213.3. It forces V to be read as finite ProbeBundle data after the pointwise monotone rows and the tolerance budget have been displayed.

Apply the budget handoff theorem Definition 4213.4. The row U can expose a single compact-uniform read only through the already displayed R, V route, and the structural rows H, C, P, N preserve that route without adding analytic data.

Row exhaustion now removes the excluded alternatives. The carrier has no coordinate for a host supremum, selected minimum radius, open-cover compactness, or ambient Real$^{\uparrow}$ completeness, so none of those objects can be exported by the finite descent. □

Theorem 3233.6 (Dini monotone ProbeBundle threshold)**.** *For an accepted* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier, once the compact source row X, monotone ledger M, pointwise-limit row L, rational budget R, and finite* ProbeBundle *cover row V are displayed, every compact-uniform tolerance export factors through a single* ProbeBundle *threshold row before U can expose a uniform-modulus handoff.*

The threshold row is finite packet data assembled from X, M, L, R, V and transported through H, C, P, N. It is not a host supremum, an externally selected radius, an open-cover compactness argument, a quotient function equality, or an ambient Real$^{\uparrow}$ *completeness coordinate.*

Proof. Start with the carrier obligations of Definition 4213.2. They place the compact source row, monotone ledger, pointwise-limit row, rational budget, finite cover row, handoff row, and structural rows in the displayed carrier inventory.

Apply finite-cover extraction by Definition 4213.3. The requested tolerance first reads the compact source and rational budget, then extracts the finite ProbeBundle cover row V on which the pointwise monotone comparisons may be inspected.

Use the budget handoff of Definition 4213.4 and the finite descent of Definition 4213.5. Together they force the local monotone rows M, L and the rational budget R to be consumed on V before the handoff row U may export a single compact-uniform read.

The common finite read on V is the required threshold row. The structural rows H, C, P, N transport, replay, source, and name it without adding analytic data, and the carrier has no coordinate for a host supremum, selected radius outside the finite ProbeBundle, open-cover compactness, quotient function equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map family rows, monotone ledgers, pointwise-limit rows, rational tolerance budgets, finite ProbeBundle cover rows, uniform-modulus and uniform-limit handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4213.1, Definition 4213.2, Definition 4213.3, Definition 4213.4, and Definition 4213.5 seed the constructive compact-uniform Dini carrier surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, arbitrary open-cover compactness, host suprema, quotient function equality, selected radii outside the finite ProbeBundle, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover extraction, monotone pointwise visibility, uniform budget handoff, transport, replay, provenance, and local naming for BEDC.Derived.DiniMonotoneCompactUniformU

Chapter 3234

A Concrete Naming Certificate for $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$

The $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ packet realizes the partition-core incidence kernel of Definition 9462.1748 as finite NameCert data. It sits over the partition carrier of chapter 201, the Boolean row discipline of chapter 3733, the simplicial support surface of chapter 238, the cellular update route of chapter 934, and the matroid rank boundary of chapter 202. The packet records finite block, quotient, lift, spectrum, and incidence rows; it is not a biological stochastic model, host graph oracle, quotient codon semantics, unrestricted spectral theorem, or hidden global scheduler.

Falsifiable prediction: Every accepted $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ read exposes the partition block row, quotient-size row, incidence-count row, lift-count row, kernel-spectrum row, Boolean support row, and local $\mathsf{NameCert}$ row before a partition-core consumer may cite the finite incidence certificate.

Independence witness: partition, booleanalgebra, simplicialcomplex, cellularautomaton, matroid: none alone is bijective with $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ because the packet jointly carries partition blocks, Boolean support tests, support-nerve incidence, cellular replay, rank-separation boundaries, quotient spectra, lift counts, transport, replay, provenance, and local naming.

Definition 3234.1 (Partition-core incidence certificate carrier)**.** *A* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, Q, A, L, S, E, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite partition-block row, recording the displayed block classes and their support names.*
2. *Q is the quotient-size row, recording the size-graded partition-core windows used by the incidence kernel.*
3. *A is the incidence-count row, recording finite adjacency or boundary incidences between displayed classes.*
4. *L is the lift-count row, recording codon-level or support-level lifts of each quotient incidence.*
5. *S is the kernel-spectrum row, recording the finite spectra exposed by the displayed quotient kernels.*

6. *E is the Boolean and simplicial support row, recording complement tests, face membership, and support-nerve incidence.*
7. *R is the rank and replay row, recording the matroid rank boundary and cellular replay needed by downstream partition-core consumers.*
8. *H records componentwise* hsame *transport for block, quotient, incidence, lift, spectrum, support, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from partition-core requests through $B, Q, A, L, S, E, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, Q, A, L, S, E, R, H, C, P, N$. It has no coordinate for host graph reachability, biological stochastic semantics, quotient codon identity, unrestricted homology, hidden scheduler state, or an external spectral oracle.

Lean4 ✓ `BEDC.Derived.PartitionCoreIncidenceCertificateUp.PartitionCoreIncidenceCertificateTasteGate single carrier alignment`

Theorem 3234.2 (Partition-core incidence certificate obligations)**.** *For every accepted* PartitionCoreIncidenceCertifi *carrier*

$$I = (B, Q, A, L, S, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$ *surface consists exactly of partition-block admission through B, quotient-size exposure through Q, incidence-count exposure through A, lift-count exposure through L, finite kernel-spectrum exposure through S, Boolean and simplicial support exposure through E, rank and cellular replay exposure through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4214.1. The rows B, Q, A, L, S, E, R are the finite incidence data, and H, C, P, N are the structural certificate rows.

The incidence route first reads partition blocks and quotient sizes. Only then can a consumer inspect incidence counts, lift counts, and finite kernel spectra.

Support and replay data are also displayed. The Boolean and simplicial row E exposes the support tests, while R records rank separation and cellular replay for the same finite partition-core request.

The structural rows preserve exactly those coordinates. Since the excluded host semantics and unrestricted spectral or reachability principles have no carrier row, they cannot be obligations of the local NameCert surface. □

Theorem 3234.3 (Partition-core incidence TasteGate boundary)**.** *Any TasteGate realization for* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *must provide* Nontrivial $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *and* FieldFaithful $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *instances preserving $B, Q, A, L, S, E, R, H, C, P, N$ as separate displayed fields. In particular, the gate must distinguish a changed partition block, quotient-size row, incidence-count row, lift-count row, kernel-spectrum row, support row, or replay row before it admits two packets as the same incidence certificate.*

Proof. Use the carrier of Definition 4214.1. The rows B, Q, A, L, S, E, R play different roles: source blocks, quotient windows, incidences, lifts, spectra, support tests, and replay boundaries.

The obligations of Definition 4214.2 consume those rows in order. Collapsing two of them would erase either the source partition, a counted incidence, a lifted incidence, a spectral claim, or the support and replay boundary.

Nontriviality is witnessed at the paper surface by packets that vary one displayed row while keeping the others fixed. Field faithfulness must therefore preserve the row list as a visible TasteGate boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ packet is generated from partition-block rows, quotient-size rows, incidence-count rows, lift-count rows, finite kernel-spectrum rows, Boolean and simplicial support rows, rank and cellular replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4214.1, Definition 4214.2, and Definition 4214.3 seed the finite partition-core incidence certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PartitionCoreIncidenceCertificateUp.PartitionCoreIncidenceCertificateTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, biological stochastic semantics, quotient codon identity, host graph reachability, unrestricted homology, an external spectral oracle, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, incidence-count exposure, lift-count exposure, finite spectrum exposure, support replay, Nontrivial PartitionCoreIncidenceCertificateUp, FieldFaithful PartitionCoreIncidenceCertificateUp, transport, provenance, and local naming for BEDC.Derived.PartitionCoreIncidenceCertificateUp.

Chapter 3235

A Concrete Naming Certificate for $\mathsf{CauchyMajorantSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyMajorantSequence}^{\uparrow}$ packet records the finite majorant-control surface used when a regular Cauchy name is read through explicit dyadic error budgets. It sits between the completion consumer of chapter 1691, the threshold discipline of chapter 292, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names one explicit Bishop-style majorant route; it is not a quotient stream, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3235.1 (Cauchy majorant sequence carrier)**.** *A* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, M, W, D, B, R, E, H, C, N)$$

with the following ten displayed fields.

1. *S is the source regular Cauchy name, read only through* $\mathsf{RegSeqRat}^{\uparrow}$ *rows.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row assigning displayed stage bounds to requested dyadic precisions.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger opened to inspect the source name at the requested stages.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording radii, slack comparisons, and finite budget arithmetic.*
5. *B is the majorant sequence row, recording a displayed monotone bound for each inspected precision request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, M, W, D, B.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R has been displayed.*
8. *H records componentwise* hsame *transport for all source, modulus, window, dyadic, majorant, handoff, seal, replay, and naming rows.*
9. *C records displayed* Cont *replay through the same rows in consumer order.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$ *row, including* Pkg *provenance over* $\mathsf{FastCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont.

The classifier compares accepted packets only through $S, M, W, D, B, R, E, H, C, N$. No coordinate stores a quotient representative, a selected limit, countable-choice tail selection, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceTasteGate single carrier alignment`

Theorem 3235.2 (Cauchy majorant sequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMajorantSequence}$ *carrier* $J = (S, M, W, D, B, R, E, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$ *surface consists exactly of the carrier row* $S, M, W, D, B, R, E, H, C, N$*, classifier stability under componentwise* hsame*, dyadic ledger exposure, majorant exactness for the inspected precision requests,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, displayed* Cont *replay, local provenance, and exclusion of quotient streams, selected limits, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4215.1. The ten fields are the source regular Cauchy row, threshold row, finite window ledger, dyadic tolerance ledger, majorant row, regular readback, real seal, transport, replay, and local naming-provenance row.

The classifier is stable because H transports each field componentwise and C replays the same consumer order. A related packet can therefore change only by the displayed hsame rows, not by replacing the source with a host sequence or by identifying quotient representatives.

The ledger and exactness obligations are carried by D and B. The dyadic ledger records every tolerance request and comparison budget, while the majorant row records the displayed monotone bound consumed by the handoff.

Finally R and E are downstream of those rows. Since the packet has no coordinate for quotient streams, selected limits, countable-choice tails, host real equality, or ambient completeness, those objects cannot discharge any local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceCarrier namecert obligations`

Theorem 3235.3 (Cauchy majorant sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The handoff exports only the source row, displayed threshold, finite windows, dyadic tolerance ledger, majorant sequence row, componentwise transport, replay, provenance, and local name carried by the same packet.

Proof. Start with the NameCert obligations of Definition 4215.2. They force every accepted regular-rational read to expose the modulus row M, the finite windows W, and the dyadic ledger D before the handoff row R is available.

The majorant row B is then read as the exact control object for those displayed requests. It is tied to the same M, W, D rows, so it cannot be detached into an external tail selector or a host-level bounding function.

The structural rows preserve the route. Componentwise transport H, replay C, and local provenance in N keep the handoff on the same finite BEDC packet, which is the surface consumed by $\mathsf{FastCauchyCompletion}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceCarrier regseqrat handoff`

Theorem 3235.4 (Cauchy majorant sequence Real seal nonescape)**.** *Let $J = (S, M, W, D, B, R, E, H, C, N)$ be an accepted* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *read of the terminal seal E must first pass through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, majorant sequence row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff in the same carrier. The seal exports no quotient stream equality, selected Cauchy limit, countable-choice schedule, host real equality, or ambient completeness theorem.*

Proof. Read the carrier definition. The terminal real seal is the field E, and the definition places it after the source, modulus, window, dyadic, majorant, and regular-rational rows.

Apply Definition 4215.3. The regular handoff R is reached only through M, W, D, B, and every completion-facing consumer shares that displayed route.

Now inspect the classifier and ledger exclusions from Definition 4215.2. The accepted packet has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, or ambient completeness. The real seal therefore remains a terminal read of the finite majorant route rather than an escape to any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyMajorantSequence}^{\uparrow}$ packet is generated from a regular Cauchy source row, CauchyModulus threshold row, StreamName finite windows, DyadicRatCore tolerance ledger, displayed majorant sequence, RegSeqRat handoff, RealUp seal, componentwise hsame transport, Cont replay, and local NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$, seed)

Scope closed: Definition 4215.1, Definition 4215.2, Definition 4215.3, and Definition 4215.4 seed the explicit majorant-control route for regular Cauchy names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic ledger exposure, majorant exactness, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMajorantSequ

Chapter 3236

A Concrete Naming Certificate for $\mathsf{MetricBaireSpace}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{MetricBaireSpace}^{\uparrow}$ packet records the complete separable metric route through which a Baire-space consumer reads finite names. It sits after the metric carrier of chapter 4230, the completion service of chapter 128, the dense schedule of chapter 1458, the finite windows of chapter 126, the regular rational readback of chapter 301, and the prefix surface of chapter 334. The packet binds those rows as one complete-separable Baire naming surface without adding countable choice, a host topology, quotient point equality, or an external completion coordinate.

Definition 3236.1 (Metric Baire-space carrier)**.** *A* $\mathsf{MetricBaireSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, K, D, S, R, B, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the admitted point history, metric readbacks, distance classifier, and local metric naming.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over the same metric carrier, displaying the Cauchy-window and limit-service route.*
3. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *row exposing a countable dense schedule as finite name data rather than as a choice function.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row scheduling finite inspections of dense names, Baire prefixes, metric balls, and Cauchy-window requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads rational windows for centers, radii, tolerances, and regular Cauchy approximants.*
6. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *ledger row recording finite-prefix and cylinder reads consumed by the Baire-space side of the packet.*
7. *H records componentwise* hsame *transport for M, K, D, S, R, B and the local certificate rows.*
8. *C records displayed* Cont *replay from a metric-ball, dense-name, Cauchy-window, or Baire-prefix request through the visible rows.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{MetricBaireSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the metric, complete-metric, separable-metric, stream, regular-readback, and Baire-prefix rows to one common surface. It has no coordinate for a selected dense sequence, a host completion, arbitrary open-cover semantics, quotient point equality, or a generic-point selector.

Lean4 ✓ `BEDC.Derived.MetricBaireSpaceUp.MetricBaireSpaceTasteGate single carrier alignment`

Theorem 3236.2 (Metric Baire-space NameCert obligations)**.** *For every accepted* $\mathsf{MetricBaireSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, B, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric, complete-metric, separable-metric, stream, regular-readback, Baire-prefix, transport, replay, provenance, and local-name rows; classifier transport to a common metric carrier before any completion, density, or Baire-prefix row is consumed; observation-ledger exactness through* S, R, B*; stability under the componentwise transports in* H*; displayed* Cont *replay through* C*; and provenance plus local naming through* P_0, N*. No naming obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4216.1. The rows M, K, D, S, R, B are the analytic surfaces, and H, C, P_0, N are the structural transport, replay, provenance, and naming surfaces.

The classifier route begins at M. The complete-metric row K, separable row D, and Baire-prefix row B are read only after the componentwise transport row has placed them over the same metric carrier.

The finite observation route is exhausted by S, R, B. Dense-name requests, Cauchy-window requests, metric-ball checks, and Baire-prefix reads all pass through the stream schedule and regular rational readback before any consumer can use them.

Finally replay through C and attach P_0, N. Since the carrier contains no coordinate for a selected dense sequence, host completion, arbitrary open-cover object, quotient point equality, or generic-point selector, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricBaireSpace}^{\uparrow}$ packet is generated from a MetricSpace source row, CompleteMetric handoff row, SeparableMetric dense schedule, StreamName finite windows, RegSeqRat readback, BaireSpace prefix ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4216.1 and Definition 4216.2 seed the complete-separable metric Baire naming surface consumed by downstream Polish and Baire chapters.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected dense sequences, host completions, arbitrary open-cover semantics, quotient point equality, generic-point selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common metric transport, completion service, dense schedule, StreamName windows, RegSeqRat readback, Baire-prefix ledger exactness, replay, provenance, and local naming for BEDC.Derived.MetricBaireSpaceUp.

Chapter 3237

A Concrete Naming Certificate for DNAWindowSixTriggerThreshold$^{\uparrow}$

The DNAWindowSixTriggerThreshold$^{\uparrow}$ packet realizes the random trigger-time law of Definition 9462.1539 as finite trigger-rank NameCert data. It sits beside the six-bit codon source window of chapter 2234, the boundary tile packet of chapter 3393, the six-tile coverage rows of chapter 3470, the trigger-algebra surface of chapter 3478, and the orbit inventory polynomial of chapter 4220. The packet names full-trigger distribution rows, trigger-time law rows, mean and quantile tables, transport, replay, provenance, and local naming as a finite combinatorial surface; it is not a biological stochastic model, an empirical mutation law, or an unbounded trigger search.

Definition 3237.1 (DNA window-six trigger-threshold carrier)**.** *A* DNAWindowSixTriggerThreshold$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, B, C, A, F, L, Q, H, R, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *row exposing the six-coordinate codon window.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row supplying the trigger coordinates.*
3. *C is the* SixBoundaryTileCoverage$^{\uparrow}$ *row recording which trigger coordinates are covered by a finite external-codon draw.*
4. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *row computing trigger rank and full-cube activation from the displayed coverage data.*
5. *F is the full-trigger distribution row for the finite external-codon process beyond $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$.*
6. *L is the trigger-time law row recording the first time at which all four trigger coordinates are covered.*
7. *Q records the displayed mean and quantile table for the trigger time.*
8. *H records componentwise* hsame *transport for W, B, C, A, F, L, Q and the structural rows.*
9. *R records displayed* Cont *replay from an external-codon window through coverage, trigger rank, full-trigger distribution, trigger-time law, and table readback.*

10. *P is the* Pkg *provenance over* DNAWindowSixTriggerThreshold$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryT SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, *the random trigger-threshold source theorem,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed six-window, boundary, coverage, trigger-algebra, distribution, law, table, transport, replay, provenance, and local naming rows. It has no coordinate for empirical biology, stochastic fitness, hidden codon semantics, or an unrecorded search over external codon subsets.

Lean4 ✓ `BEDC.Derived.DNAWindowSixTriggerThresholdUp.DNAWindowSixTriggerThresholdTasteGate single carrier alignment`

Theorem 3237.2 (DNA window-six trigger-threshold full-cube law)**.** *For every accepted* DNAWindowSixTriggerThr *carrier $T = (W, B, C, A, F, L, Q, H, R, P, N)$, the full-cube trigger law is the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \mathsf{WindowCoverageTriggerA}$$

The row F stores the full-trigger distribution, L stores the trigger-time law, and Q stores the mean and quantile table. Any accepted consumer of the average threshold 3.12998*, the deterministic threshold* 15*, the median threshold* 3*, or the* 99% *threshold* 7 *must read those values through the displayed route.*

Lean4 ✓ `BEDC.Derived.DNAWindowSixTriggerThresholdUp.DNAWindowSixTriggerThreshold full cube law`

Proof. Project the carrier of Definition 4217.1. The source rows W, B, C, A display the finite codon window, boundary tiles, coverage ledger, and trigger algebra before any distribution row is read.

The rows F, L, Q are the numerical threshold ledger. The full-trigger distribution row determines the law row, and the law row determines the mean and quantile table recorded by Q.

Transport and replay preserve that route through H and R. Provenance and local naming attach the source theorem and the NameCert row through P, N.

Since the carrier contains no empirical-biological coordinate, stochastic fitness model, hidden codon semantics, or unrecorded subset search, every full-cube threshold read is exhausted by the displayed finite route. □

Falsifiable prediction: Every accepted DNAWindowSixTriggerThreshold$^{\uparrow}$ read of the mean, deterministic threshold, median threshold, or 99% threshold factors through CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WindowCoverageTriggerAlgebra, full-trigger distribution, trigger-time law, table, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra, boundary-orbit-inventory-polynomial: DNAWindowSixTriggerThreshold is not bijective to any listed sibling because it jointly carries the trigger-rank distribution, trigger-time law, mean and quantile table, transport, replay, provenance, and local naming as one threshold packet.

Future Lean TasteGate files for this packet must provide explicit Nontrivial DNAWindowSixTriggerThreshold$^{\uparrow}$ and FieldFaithful DNAWindowSixTriggerThreshold$^{\uparrow}$ instances before the threshold law can serve as a verified trigger-distribution consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}$.

Constructive story: A $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ packet is generated from CodonSixBitWindow source rows, DnaWindowSixBoundaryTile boundary rows, SixBoundaryTileCoverage rows, WindowCoverageTriggerAlgebra rows, full-trigger distribution rows, trigger-time law rows, mean and quantile table rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4217.1 and Definition 4217.2 seed the finite trigger-threshold packet for the window-six random trigger law.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological mechanism, empirical mutation dynamics, stochastic fitness, hidden codon semantics, unrestricted subset search, Lean verification, Nontrivial or FieldFaithful TasteGate instances, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, coverage exposure, trigger-rank distribution, trigger-time law, mean and quantile table exactness, transport, replay, provenance, local naming, Nontrivial DNAWindowSixTriggerThresholdUp, and FieldFaithful DNAWindowSixTriggerThresholdUp for BEDC.Derived.DNAWindowSixTriggerThresholdUp.

Chapter 3238

A Concrete Naming Certificate for $\mathsf{MetricCompletionKleisli}^{\uparrow}$

Following Mac Lane's Kleisli categories, and Bishop–Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet records the metric-specific bind consumer for completion-valued maps as finite BEDC data. It reads the metric completion carrier of chapter 1506, the metric adjunction route of chapter 3836, the Cauchy Kleisli consumer of chapter 1845, and the completion universal property of chapter 397. The chapter names a consumer route for bind-style completion reads; it is not a quotient completion, selected representative, countable-choice schedule, host equality of completed points, or ambient completeness theorem.

Definition 3238.1 (MetricCompletionKleisli carrier)**.** *A* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, displaying the source metric, completion carrier, separated comparison, dense-source access, and readback branch before any bind consumer is opened.*
2. *A is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *row, displaying the unit, counit, triangle ledger, and metric universal mapping boundary read by the consumer.*
3. *L is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, recording the unit, bind, and reassociated-bind consumer route inherited from the L10 completion algebra.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, exposing the extension surface through which a displayed metric source map is consumed as a completion-valued map.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the metric completion and Cauchy Kleisli routes.*
6. *E is the public* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, A, L, U, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric completion row, metric adjunction row, Cauchy Kleisli row, universal-property row, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a bind consumer request through $M, A, L, U, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionKleisli}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row.*
The classifier accepts only packets whose bind consumer first displays the metric completion, metric adjunction, Cauchy Kleisli, universal-property, regular readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.MetricCompletionKleisliUp.MetricCompletionKleisliTasteGate single carrier alignment`

Theorem 3238.2 (MetricCompletionKleisli NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier*

$$K = (M, A, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *surface consists exactly of metric-completion admission, metric-adjunction admission, Cauchy-Kleisli bind exposure, completion-universal-property exposure, RegSeqRat readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of K. No obligation may be discharged by a quotient completion, selected Cauchy representative, countable-choice selector, host equality of completed points, or ambient complete metric space.*

Proof. Project the carrier of Definition 4218.1. Its analytic rows are precisely the metric completion row M, the metric adjunction row A, the Cauchy Kleisli row L, the universal-property row U, the regular readback row R, and the Real seal row E.

The metric rows determine the order of access. A bind consumer opens M before it can read the adjunction data in A, and it opens A before the Cauchy Kleisli consumer L can hand off unit, bind, or reassociated-bind requests to the universal-property row U.

Transport and replay add no analytic coordinate. The rows H and C compare and replay only the displayed completion, adjunction, Kleisli, universal-property, readback, and seal data.

Finally P and N source and name the same finite packet. Since no listed row stores a quotient completion, selected representative, countable-choice selector, host equality, or ambient complete metric space, the displayed rows exhaust the NameCert surface. □

Theorem 3238.3 (MetricCompletionKleisli bind handoff)**.** *Every bind-style consumer read of an accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{CauchyCompletionKleisli}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the metric completion row M. It displays the source metric, completion carrier, separated comparison, dense-source access, and readback branch before a consumer can request completion-valued bind data.

Next read the adjunction row A. The unit, counit, triangle ledger, and metric universal mapping boundary are the only route by which the metric completion packet exposes extension data to a downstream consumer.

Attach the Cauchy Kleisli row L and the universal-property row U. The row L supplies the unit, bind, and reassociated-bind consumer surface, while U records the extension route that consumes a displayed source map as a completion-valued map.

The readback and seal rows close the route. The row R supplies the regular rational readback used by both completion and Kleisli sides, and E is the only public Real seal. The structural

rows H, C, P, N transport, replay, source, and name this same route, so no quotient completion, selected representative, choice schedule, host equality, or ambient completeness theorem can enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet is generated from metric completion, metric adjunction, Cauchy Kleisli, completion universal property, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4218.1, Definition 4218.2, and Definition 4218.3 seed the metric-specific Kleisli consumer route for bind-style completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-adjunction exposure, Cauchy-Kleisli exposure, universal-property exposure, RegSeqRat/Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionKleisliUp.

Chapter 3239

A Concrete Naming Certificate for $\mathsf{LocatedRegularRealCut}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet records the finite boundary where regular-Cauchy real names expose located lower and upper rational cut rows before the terminal real seal is read. It sits beside the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the Dedekind cut carrier of chapter 436, and the located regular sequence route of chapter 2376. The packet is a finite NameCert bridge from regular-Cauchy observations to located cut boundaries; it does not claim Dedekind completeness, quotient equality of real names, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3239.1 (Located regular real cut carrier)**.** *A* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational cut row, recording inhabited rounded lower observations read through the finite cut-facing side of* $\mathsf{DedekindCut}^{\uparrow}$*.*
2. *U is the upper rational cut row, recording rounded upper observations attached to the same inspected real-name request.*
3. *M is the locatedness modulus row, binding every rational comparison request to a finite window on which either the lower row or the upper row is displayed with the required separation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row supplying dyadic radii, bracket widths, midpoint requests, and comparison tolerances for the cut boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations from which the cut rows and locatedness modulus are read.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for lower cut rows, upper cut rows, locatedness modulus rows, dyadic tolerance rows, regular readback rows, Real seal rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a cut-boundary request through L, U, M, D, R, A, H.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRegularRealCut}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row.*

The carrier accepts only the displayed lower row, upper row, locatedness modulus, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows. It has no coordinate for Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedRegularRealCutUp.LocatedRegularRealCutUp`

Theorem 3239.2 (Located regular real cut obligations)**.** *For every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier* $K = (L, U, M, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed lower cut row, upper cut row, locatedness modulus, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*
2. *located lower and upper stability, requiring L and U to remain rounded, separated, and transported componentwise under the same* hsame *row;*
3. *regular Cauchy handoff, requiring every cut-boundary read to pass through R and the located regular sequence discipline of chapter 2376;*
4. *Real seal non-escape, requiring A to be downstream of L, U, M, D, R and not an independent completeness row;*
5. *ledger exactness, requiring P and N to name only L, U, M, D, R, A, H, C and to exclude quotient equality, selected limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4219.1. The displayed analytic rows are the lower cut row L, upper cut row U, locatedness modulus M, dyadic tolerance row D, regular readback row R, and Real seal row A.

The lower and upper stability clauses are componentwise. Transport through H preserves the two cut rows together with the same modulus and dyadic tolerance row, so the classifier cannot move a lower observation without moving the matching upper boundary data.

The regular Cauchy handoff is the route through R. The located regular sequence sibling supplies the finite windows and readback discipline before the cut boundary can be consumed, while D supplies the tolerance ledger on which the locatedness modulus is read.

Finally A, C, P, N seal, replay, source, and name only the displayed route. Since none of the carrier coordinates is a quotient, selected limit, trichotomy branch, choice schedule, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 3239.3 (Located regular real cut Cauchy boundary route)**.** *Every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *consumer factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedRegSeqRat}^{\uparrow},\ \mathsf{DedekindCut}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed finite rows before it can read the local $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row. The route exports a located lower–upper cut boundary for a regular-Cauchy real name; it does not export Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 4219.2. A consumer first sees the dyadic tolerance row D, regular readback row R, and locatedness modulus M before the lower and upper cut rows can be treated as a located boundary.

The Dedekind cut sibling supplies the cut-facing row shape, while the located regular sequence sibling supplies the Cauchy-facing source and tail-window discipline. These rows meet only through the displayed finite packet K.

The Real seal row A is terminal for this route. Componentwise transport, continuation replay, package provenance, and local naming preserve the same displayed rows, so no consumer can bypass them by invoking quotient equality, a selected representative, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRegularRealCut}^{\uparrow}$ read factors through displayed lower rational cut, upper rational cut, locatedness modulus, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, dedekindcut, located-regseqrat: no listed sibling is bijective to $\mathsf{LocatedRegularRealCut}^{\uparrow}$ because this packet alone binds lower and upper cut rows, a locatedness modulus, dyadic tolerance, regular readback, terminal real seal, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet is generated from lower rational cut rows, upper rational cut rows, a locatedness modulus, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, seed)

Scope closed: Definition 4219.1, Definition 4219.2, and Definition 4219.3 seed the finite bridge from regular-Cauchy real names to located lower–upper cut boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located lower and upper stability, regular Cauchy handoff, Real seal non-escape, and ledger exactness for BEDC.Derived.LocatedRegularRealCutUp.

Chapter 3240

A Concrete Naming Certificate for $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$

The $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet realizes the orbit and codon inventory polynomials of Definition 9462.1713 as finite BEDC NameCert data. It consumes the boundary-tile source of chapter 3393, the coverage packet of chapter 3470, the codon window source of chapter 2234, and the finite polynomial carrier of chapter 50. The packet records orbit rows, codon-count rows, coefficient ledgers, and marginal projections; it is not a biological stochastic model, an empirical code table, a hidden enumeration oracle, or quotient codon semantics.

Definition 3240.1 (Boundary orbit inventory polynomial carrier)**.** *A* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, K, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the orbit table row, recording the finite orbit types classified by support size m, overlap defect δ, lift imbalance b, and orbit size.*
2. *K is the codon count table row, recording the weighted counts of lifted triggers over the same triples (m, δ, b).*
3. *E is the coefficient ledger row, storing the displayed coefficients of the orbit inventory polynomial $\mathcal{O}(u, v, w)$ and the codon inventory polynomial $\mathcal{C}(u, v, w)$.*
4. *M is the marginal projection row, recording the one-variable and bivariate specializations read from the coefficient ledger.*
5. *H records componentwise* hsame *transport for the orbit table, codon count table, coefficient ledger, marginal projections, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an inventory query through O, K, E, M, H.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *the source vision of Definition 9462.1713,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *row.*

The carrier accepts coefficient and marginal reads only through O, K, E, M, H, C, P, N. It has no coordinate for a hidden biological code table, probabilistic fitness model, unbounded orbit enumeration, quotient codon equality, or unstated polynomial coefficient source.

Lean4 ✓ `BEDC.Derived.BoundaryOrbitInventoryPolynomialUp.BoundaryOrbitInventoryPolynomialTasteGate`
`single carrier alignment`

Theorem 3240.2 (Boundary orbit inventory polynomial obligations)**.** *For every accepted* BoundaryOrbitInventoryP *carrier $I = (O, K, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *surface consists exactly of orbit-table admission through O, codon-count table admission through K, coefficient-ledger exactness through E, marginal-projection exactness through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4220.1. The orbit table and codon count table are the only source rows, while the coefficient ledger and marginal projection rows are the only polynomial-facing rows.

Coefficient exactness is read from E. Each displayed coefficient is determined by grouping the rows of O and weighting them by the count row K; no coefficient row is available without those two source tables.

Marginal exactness is read from M. A marginal projection is permitted only as a specialization of the coefficient ledger, so it cannot replace the orbit table or codon count table by a hidden enumeration.

The structural rows H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier lacks biological stochastic, quotient-codon, hidden-table, and unbounded-enumeration coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3240.3 (Boundary orbit inventory coefficient determinacy)**.** *Every coefficient marginal exported by an accepted* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *packet is determined by the displayed orbit table O, codon count table K, coefficient ledger E, and marginal projection row M. Removing any coefficient row from E leaves a marginal query without a displayed source, and removing any orbit or codon count row leaves the corresponding coefficient without a displayed inventory witness.*

Proof. Read the NameCert obligations of Definition 4220.2. A coefficient query first exposes the orbit table O and codon count table K, then reads the coefficient ledger E.

The marginal projection row M is downstream of the ledger. It can specialize the displayed polynomial, but it cannot introduce a coefficient absent from E or reconstruct a missing source row absent from O or K.

Transport and replay preserve the same dependency order. Therefore a coefficient marginal is determined exactly by the displayed table and ledger rows, and deletion of any required row breaks the finite NameCert route. □

Falsifiable prediction: For $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, coefficient marginals are determined by the displayed orbit table, codon count table, coefficient ledger, and marginal projection rows, and a marginal read fails if any required coefficient row is removed.

Independence witness: partition, hash, errorcode, dna-window-six-boundary-tile, six-boundary-tile-coverage: the polynomial coefficient ledger is not bijective to these sibling carriers because it binds orbit tables, codon count tables, coefficient rows, marginal projections, transport, replay, provenance, and local naming, while the siblings supply partitioning, digest, error coding, boundary-tile, or coverage surfaces without this coefficient inventory.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet is generated from an orbit table, codon count table, coefficient ledger, marginal projection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4220.1, Definition 4220.2, and Definition 4220.3 seed the finite orbit-inventory polynomial certificate over boundary-tile and coverage rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological stochastic modeling, empirical code-table semantics, quotient codon equality, unbounded orbit enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-table admission, codon-count table admission, coefficient-ledger exactness, marginal-projection exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryOrbitInventoryPol

Chapter 3241

A Concrete Naming Certificate for $\mathsf{PrimitiveRecursive}^{\uparrow}$

Following Kleene's primitive recursive functions, the $\mathsf{PrimitiveRecursive}^{\uparrow}$ packet records finite recursion syntax and its local replay rows as BEDC data. It sits after the unary natural-number carrier of chapter 28, the list spine of chapter 36, the finite computability surface of chapter 199, the stream-window discipline of chapter 126, and the regular rational readback route of chapter 301. The chapter supplies a finite schedule substrate for later modulus and real-name consumers; it does not introduce minimization, partial recursive search, host functions, or unbounded choice.

Definition 3241.1 (PrimitiveRecursive carrier)**.** *A* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Z, S, \Pi, K, R, H, C, Q, N)$$

whose rows are read as follows:

1. *Z is the zero-function row.*
2. *S is the successor-function row.*
3. *Π is the finite family of projection rows, each with displayed arity and coordinate index.*
4. *K is the composition row, storing one outer code, its finite list of inner codes, and the arity ledger that makes the composition well typed.*
5. *R is the primitive-recursion scheme row, storing the base code, step code, displayed accumulator coordinate, and finite recursion spine over the* $\mathsf{Nat}^{\uparrow}$ *row.*
6. *H is componentwise* hsame *transport for zero, successor, projection, composition, recursion, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay for evaluating a finite syntax tree by structural recursion through the rows Z, S, Π, K, R.*
8. *Q is the* Pkg *provenance row over* $\mathsf{PrimitiveRecursive}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, S, \Pi, K, R, H, C, Q, N$. There is no coordinate for minimization, unbounded search, partiality, a host function graph, countable choice, or an external evaluator.

Lean4 ✓ `BEDC.Derived.PrimitiveRecursiveUp.TasteGate.PrimitiveRecursiveTasteGate single carrier alignment`

Theorem 3241.2 (PrimitiveRecursive NameCert obligations)**.** *For every accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier*

$$P = (Z, S, \Pi, K, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *surface consists exactly of carrier admission for the zero, successor, projection, composition, primitive-recursion, transport, replay, provenance, and naming rows; arity compatibility for* Π, K, R*; structural replay of the displayed syntax tree through* C*; componentwise transport through* H*; and provenance through* Q, N*. No obligation is stored in a minimization row, partial recursive search row, host-function graph, or unbounded-choice selector.*

Proof. Inspect the carrier row by row. The base constructors are Z, S, Π, and their arity information is part of the displayed finite rows rather than a host-side function space.

For composite syntax, the row K supplies the finite list of inner codes and the outer code, while R supplies the base and step codes for the primitive-recursion scheme. Induction on the displayed syntax tree checks that every replay request descends to one of Z, S, Π or to a strictly displayed subcode of K or R.

The structural rows close the local certificate. Transport through H preserves the same constructor rows, replay through C evaluates only the displayed syntax tree, and Q, N record provenance and naming. Since the packet has no coordinate for minimization, partial recursive search, host functions, or unbounded choice, the listed rows exhaust the NameCert obligations. □

Theorem 3241.3 (PrimitiveRecursive composition closure)**.** *If an accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *packet contains a composition row* K *whose outer code and finite list of inner codes are accepted by the same carrier, then the composed code is again read as a* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *row through* K, H, C, Q, N*. The closure is internal to the finite displayed syntax and exports no minimization, partiality, host-function graph, or unbounded-choice principle.*

Proof. Proceed by induction on the finite list of inner codes recorded in K. The empty-list boundary is the displayed arity check for the outer code, and the single-step case appends one already accepted inner row to the composition ledger.

At each induction step the replay row C evaluates the inner rows before the outer row, and transport through H preserves the same arity and constructor classifiers. The provenance row Q and local naming row N then repackage the resulting finite syntax as a $\mathsf{PrimitiveRecursive}^{\uparrow}$ row.

No unbounded search is used in the induction. The construction consumes only the displayed finite list in K and the accepted subcodes already present in the carrier, so it cannot introduce minimization, partial recursive search, host functions, or unbounded choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4221.1, Definition 4221.2, and Definition 4221.3 seed the finite recursion-syntax certificate for schedule and modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimization, partial recursive search, host functions, unbounded choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, arity compatibility, composition closure, primitive-recursion replay, transport, provenance, and local naming for the same finite recursion syntax.

Constructive story: The construction starts from zero, successor, and projection rows, closes finite composition through a displayed list spine, and closes primitive recursion by replaying the base and step codes over the Nat row under componentwise transport and local naming.

The $\mathsf{CauchyAcceleration}^{\uparrow}$ packet reads the StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces of chapter 126, chapter 291, chapter 301, and chapter 38 before exporting its accelerated completion-facing route.

Chapter 3242

A Concrete Naming Certificate for $\mathsf{CauchyAcceleration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyAcceleration}^{\uparrow}$ packet records the finite acceleration step by which an ordinary displayed Cauchy name is replaced by a cofinal regular rational name without changing the represented real. It sits after the criterion surfaces of chapter 392 and chapter 3576, the regularization route of chapter 376, the windowed name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The acceleration packet consumes Cauchy tail comparison as substantive finite-window and dyadic-tolerance data through chapter 301, chapter 126, chapter 291, and chapter 38 before any accelerated Real-facing read is exported. The accelerated cofinal regular-rational name is consumed by chapter 5721 as the completion-existence witness route, with the read still passing through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet makes acceleration a named finite ledger: it does not select an ambient subsequence, quotient a Cauchy representative, or appeal to host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3242.1 (Cauchy acceleration carrier)**.** *A* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite windows display the original rational Cauchy name.*
2. *T is the dyadic tolerance schedule, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, assigning the requested precision budget to each inspected window.*
3. *α is the cofinal acceleration map, stored as a finite monotone ledger saying which source window is read for each accelerated precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *proof row for the accelerated rational name, including the regularity inequalities obtained by reading S through α and T.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the accelerated regular name reaches the same Real-facing boundary as the source name.*
6. *H records componentwise* hsame *transport for the source stream, tolerance schedule, cofinality map, regular proof row, Real seal, provenance, and local naming row.*

7. *C records displayed* Cont *replay from an acceleration request through S, T, α, R, E and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyAcceleration}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient representative of the original Cauchy name, a choice-selected subsequence, an ambient completeness theorem, or a host-side equality oracle for real numbers.

Theorem 3242.2 (Cauchy acceleration NameCert obligations)**.** *For every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row S, dyadic tolerance schedule T, cofinal map α, regular rational proof row R, and Real seal row E;*
2. *monotone cofinality, requiring α to preserve request order and to eventually reach every inspected source precision window;*
3. *dyadic tolerance preservation, requiring each accelerated request to read the same displayed tolerance budget through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
4. *classifier transport, requiring H and C to preserve the source stream, tolerance schedule, cofinality ledger, regular proof row, and Real seal under componentwise* hsame *and displayed replay;*
5. *ledger exhaustion through P and N, excluding quotient representatives, choice-selected subsequences, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and host real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4222.1. The accepted packet exposes only the source stream S, tolerance schedule T, acceleration ledger α, regular readback R, Real seal E, and the structural rows.

Monotone cofinality is the public content of α. Since α is stored as a finite ledger over the inspected request window, each accelerated precision read names the source window it consumes and records that later requests do not move backward.

Dyadic tolerance preservation follows by reading T before the regular rational proof row. The $\mathsf{RegSeqRat}^{\uparrow}$ row R receives exactly the tolerance entries displayed by $\mathsf{DyadicRatCore}^{\uparrow}$, so the acceleration cannot import an unrecorded modulus or external precision oracle.

Classifier transport is componentwise through H and replayed through C. These rows preserve S, T, α, R, E without adding a second classifier surface.

Finally, P and N name precisely the displayed packet. Because the carrier contains no quotient representative, choice-selected subsequence, ambient completeness coordinate, or host real equality oracle, the five listed rows exhaust the local naming obligations. □

Theorem 3242.3 (Cauchy acceleration L10 handoff)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$ exports only the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $S, T, \alpha, R, E, H, C, P, N$. Consequently every L10 consumer receives a $\mathsf{RegSeqRat}^{\uparrow}$*-compatible and* $\mathsf{Real}^{\uparrow}$*-compatible accelerated name, while quotient representatives of the original Cauchy name, choice-selected subsequences, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness remain unavailable.*

Proof. Project the accepted carrier and apply Definition 4222.2. The only public data are $S, T, \alpha, R, E, H, C, P, N$.

The source read begins at the $\mathsf{StreamName}^{\uparrow}$ row S. The tolerance read then passes through T and α, so every accelerated precision request is attached to an explicit dyadic budget and a displayed source window.

The regular rational handoff is exactly R, and the Real-facing boundary is exactly E. Transport and replay through H and C preserve that route for downstream consumers.

Since P and N exhaust the packet and no forbidden coordinate appears in Definition 4222.1, the exported object is only the displayed accelerated name compatible with $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 3242.4 (Cauchy acceleration rectangular window route)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier* $A = (S, T, \alpha, R, E, H, C, P, N)$ *reuses the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows when producing the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the terminal* $\mathsf{Real}^{\uparrow}$ *handoff. The cofinal ledger* α *may refine the order of precision requests, but it cannot replace the displayed window route recorded by* S, T, H, C, P, N*.*

Proof. Project the rows $S, T, \alpha, R, E, H, C, P, N$. The source windows are recorded in S, and the dyadic tolerance entries are recorded in T before the accelerated regular rational row R or the Real-facing seal E is consumed.

Apply Definition 4222.2. The cofinality row α is an order ledger over displayed source windows, while dyadic tolerance preservation forces each accelerated request to read the same T row before it reaches R.

Now apply Definition 4222.3. Transport and replay through H and C preserve the route named by P and N, so no quotient representative, selected subsequence outside the ledger, or ambient completeness row is exported. □

Theorem 3242.5 (Cauchy acceleration cofinal-ledger determinism)**.** *Let*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

and

$$A' = (S, T, \alpha', R', E', H, C, P, N)$$

be accepted $\mathsf{CauchyAcceleration}^{\uparrow}$ *carriers sharing the same displayed source stream, tolerance schedule, structural replay rows, provenance rows, and local naming boundary through* S, T, H, C, P, N*. If the two cofinal ledgers* α *and* α' *choose the same source window for every inspected precision request, then their accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal follow the same finite route. In particular, neither ledger can replace the displayed source-window or tolerance route before* R, R' *and* E, E' *are read.*

Proof. Project both accepted carriers through Definition 4222.1. The shared rows expose the same $\mathsf{StreamName}^{\uparrow}$ source windows, the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule, and the same replay and provenance boundary.

Apply Definition 4222.2 to both packets. The only ledger freedom available before the accelerated readback is the cofinal row, and by hypothesis the displayed source window selected for each inspected request is identical in α and α'.

Now use Definition 4222.4. Since the rectangular source-window and tolerance route is fixed, the regular rational rows R, R' and the Real seals E, E' are reached through the same finite route. The packet therefore has no row through which a second ledger can substitute a hidden subsequence, quotient representative, or ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyAcceleration}^{\uparrow}$ packet is generated from a StreamName source row, a DyadicRatCore tolerance schedule, a finite cofinal acceleration ledger, a RegSeqRat proof row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4222.1, Definition 4222.2, and Definition 4222.3 seed the finite acceleration route from StreamName windows through dyadic tolerance schedules, cofinal acceleration ledgers, RegSeqRat readback, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy-name equality, choice-selected subsequences, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone cofinality, dyadic tolerance preservation, classifier transport, L10 handoff, and ledger exhaustion for BEDC.Derived.CauchyAccelerationUp.

Theorem 3242.6 (Cauchy acceleration sequence-space route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer reads the common-window classifier of* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *from chapter 406 before exporting the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff. The route passes through the regularization boundary of chapter 376: the source* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accelerated cofinal ledger, regular rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows are visible before any Real-facing consumer may inspect the acceleration result. No quotient representative, selected subsequence outside the displayed ledger, host equality, or ambient completeness row is exported.*

Theorem 3242.7 (Cauchy subsequence dependency route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal must first consume the rate-dominance surface of chapter 1814 and the fast-modulus surface of chapter 1781. The route is finite:* $\mathsf{CauchyRateDominance}^{\uparrow}$ *bounds the source schedule,* $\mathsf{FastCauchyModulus}^{\uparrow}$ *supplies the inspected acceleration window, and the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local* NameCert *rows carry the resulting readback. No consumer may replace this route by a chosen subsequence, quotient representative, ambient completion row, or host equality of Real names.*

Proof. Begin with the accepted carrier of Definition 4222.1. Its source stream row S, tolerance schedule T, cofinal ledger α, regular rational row R, and Real seal E are the only analytic rows available before transport, replay, provenance, and local naming.

The rate-dominance chapter supplies the comparison row that bounds which source schedule entries may be read by α. The fast-modulus chapter supplies the inspected window that turns those bounded reads into the accelerated tolerance route. Reading either row after the $\mathsf{RegSeqRat}^{\uparrow}$ readback would hide the schedule obligation rather than consume it.

Apply the rectangular window route of Definition 4222.4. It keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows in front of R and E, so the accelerated readback remains a displayed finite route. The carrier has no coordinate for a selected subsequence, quotient representative, ambient completion proof, or host Real equality, and the dependency route cannot export one. □

Theorem 3242.8 (Cauchy acceleration L10 four-face cofinality)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal preserves the ordered L10 four-face route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The cofinal acceleration ledger may refine which displayed source window is read for a requested precision, but it cannot commute, omit, or replace the $\mathsf{StreamName}^{\uparrow}$ *source face, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The resulting route remains finite and displayed, with no chosen subsequence outside the ledger, quotient representative, ambient completion row, host real equality, countable-choice schedule, Lean verification, or standard bridge.*

Proof. Start with Definition 4222.3. It names the only exported analytic route of an accepted carrier as the ordered sequence from the stream window through the dyadic tolerance ledger and regular rational readback to the Real-facing seal.

Next apply Definition 4222.4. The cofinal ledger α refines precision requests inside the displayed rectangle of source windows and dyadic tolerances, so it cannot move the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ rows ahead of the upstream faces.

Finally use the sibling dependency route of Definition 4222.7. The rate-dominance and fast-modulus surfaces are consumed before the accelerated readback is exported, and they leave the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces in order.

The carrier and dependency route have no row for a hidden subsequence, quotient representative, ambient completion proof, host Real equality, choice schedule, Lean verification, or bridge credential. Hence cofinal acceleration preserves exactly the displayed L10 four-face route. □

Theorem 3242.9 (Cauchy acceleration completion forward link)**.** *Every completion-facing consumer of* $\mathsf{CauchyAcceleration}^{\uparrow}$ *reads through chapter 1639, chapter 301, chapter 126, chapter 291, and chapter 38 before the accelerated route exports a terminal real boundary. The dependency is the finite route from source* $\mathsf{StreamName}^{\uparrow}$ *windows through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule and cofinal acceleration ledger into* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the Bishop regular Cauchy completion surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the displayed source windows. The accelerated route has no completion-facing input until the $\mathsf{StreamName}^{\uparrow}$ inspection windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule have been exposed.

Next read the regular rational and completion surfaces. The cofinal acceleration ledger feeds $\mathsf{RegSeqRat}^{\uparrow}$ readback and then passes through the Bishop regular Cauchy completion chapter before any terminal real boundary is visible.

Finally attach the $\mathsf{Real}^{\uparrow}$ seal. Since the route is finite and displayed at each sibling boundary, it exports no quotient completion, selected subsequence, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Chapter 3243

A Concrete Naming Certificate for $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$

Following Brouwer and Bishop–Bridges on monotone sequences in constructive analysis, the BrouwerMonotoneSeque packet records a monotone real-name boundary object through finite stream windows, regular rational readback, dyadic tolerance ledgers, and the terminal real-name seal. It sits after the Specker obstruction of chapter 1406, the regular-rational readback of chapter 301, the stream-window interface of chapter 126, the dyadic rational core of chapter 291, and the Real seal of chapter 38. The chapter gives the L10 Real and Completion frontier a named monotone-sequence carrier without claiming located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3243.1 (BrouwerMonotoneSequence carrier)**.** *A* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row, listing the inspected indices and their finite dependency order.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the inspected real-name entries.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed windows and monotonicity comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been displayed.*
6. *M is the monotonicity ledger, recording only finite adjacent-window comparison obligations among the displayed entries.*
7. *H records componentwise* hsame *transport for source indices, windows, dyadic ledgers, readback, Real seals, monotonicity, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a monotone-sequence request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets whose monotone-sequence request displays these rows in order. It has no coordinate for a located supremum, trichotomy decision, countable-choice selector, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BrouwerMonotoneSequenceUp.BrouwerMonotoneSequenceTasteGate single carrier alignment`

Theorem 3243.2 (BrouwerMonotoneSequence NameCert obligations)**.** *For every accepted* BrouwerMonotoneSeque *carrier*

$$B = (S, W, D, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the inspected source sequence, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, finite monotonicity ledger, componentwise transport, displayed replay, package provenance, and local naming rows. No NameCert obligation may be discharged by a located supremum, trichotomy of real names, countable-choice schedule, selected limit, quotient stream representative, or ambient complete real line.*

Proof. Project the carrier of Definition 4223.1. The analytic rows are precisely the source schedule S, the finite windows W, the dyadic tolerance ledger D, the regular rational readback R, the Real seal E, and the finite monotonicity ledger M.

The monotonicity information is local to the displayed windows. Each comparison read in M consumes only the adjacent source entries already opened through S, W, D, R and does not add a global order decision for arbitrary real names.

The structural rows add no hidden analytic principle. Transport through H preserves the same displayed rows, replay through C follows the same finite route, and P, N record provenance and local naming for the packet.

Since the carrier has no row for located suprema, trichotomy, countable-choice schedules, selected limits, quotient stream representatives, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3243.3 (BrouwerMonotoneSequence finite-window nonescape)**.** *Every consumer read of an accepted* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier reaches a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may consult the finite monotonicity ledger M for displayed adjacent windows, but it cannot escape to a located supremum, countable-choice selector, trichotomy oracle, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer read of the accepted carrier. The source row S exposes only the inspected indices, so the first real-name data available to the consumer are the finite $\mathsf{StreamName}^{\uparrow}$ windows in W.

The dyadic and regular-rational rows determine the only readback route. The tolerance ledger D bounds the displayed windows, and the row R packages those bounded observations as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The terminal seal is downstream of those rows. The $\mathsf{Real}^{\uparrow}$ row E is available only after S, W, D, R have been displayed, and the monotonicity ledger M can compare only the adjacent windows already present in the same finite packet.

Transport, replay, provenance, and local naming preserve this order through H, C, P, N. Therefore every consumer read remains inside the displayed finite-window route and cannot import a

located supremum, trichotomy oracle, choice schedule, selected limit, quotient stream representative, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ packet is generated from inspected source indices, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, a finite monotonicity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4223.1, Definition 4223.2, and Definition 4223.3 seed the constructive monotone-sequence boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window monotonicity, StreamName exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerMonotoneSequenceUp.

Chapter 3244

A Concrete Naming Certificate for $\mathsf{CauchySequenceCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet records the textbook sequence criterion as finite BEDC data. It ties a sequence source to the dyadic tolerance discipline of chapter 291, the stream-window naming discipline of chapter 126, the regular rational readback of chapter 301, and the public real seal of chapter 38. The chapter names only the finite route from tolerance data to regular-Cauchy and real consumers; it does not supply a choice-selected subsequence, quotient equality of limits, host completion, or ambient completeness principle.

Definition 3244.1 (CauchySequenceCriterion carrier)**.** *A* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the sequence-source row, displaying the named sequence and the finite request interface through which a downstream consumer asks for terms.*
2. *M is the modulus-schedule row, displaying the finite schedule that turns a requested dyadic tolerance into a stage bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row, recording the positive dyadic thresholds, comparisons, and tolerance refinements read by the schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the sequence after the schedule row has supplied their stage bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row reached from A, M, D, S.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-Cauchy readback has been displayed.*
7. *H records componentwise* hsame *transport for the sequence source, schedule, dyadic tolerance ledger, stream windows, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a criterion request through $A, M, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose criterion consumer first displays the sequence source, modulus schedule, dyadic tolerance ledger, stream-window row, regular-Cauchy readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCriterionUp.CauchySequenceCriterionTasteGate single carrier alignment`

Theorem 3244.2 (CauchySequenceCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier*

$$Q = (A, M, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *surface consists exactly of sequence-source admission, modulus-schedule admission, dyadic tolerance-ledger exposure, finite stream-window exposure, regular-Cauchy readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of* Q*. No obligation may be discharged by a choice-selected subsequence, quotient equality of limits, host completion, ambient completeness, or an unbounded open-cover argument.*

Proof. Project the carrier of Definition 4224.1. Its analytic rows are precisely the sequence source A, modulus schedule M, dyadic tolerance ledger D, stream windows S, regular-Cauchy readback R, and Real seal E.

The tolerance rows determine the access order. A criterion consumer reads M before it can turn a dyadic request from D into a finite stage bound, and it reads S only through the finite windows exposed by that bound.

The readback rows add no external completion object. The row R repacks the displayed tolerance and window data as regular-Cauchy readback, and E seals that readback as the public real-facing output.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate of Q stores a selected subsequence, quotient limit equality, host completion, ambient completeness witness, or open-cover argument, the displayed rows exhaust the NameCert obligations. □

Theorem 3244.3 (CauchySequenceCriterion L10 handoff)**.** *Every L10 consumer read of an accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through the sequence source, modulus schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the sequence source A and modulus schedule M. They expose the named sequence and the finite stage bounds required by any dyadic tolerance request.

Next read the dyadic tolerance ledger D and stream-window row S. The ledger supplies the positive thresholds and comparisons, while the stream row supplies only the finite windows named by the schedule.

Attach the regular-Cauchy readback row R. It is the unique displayed route by which the dyadic-window data becomes a regular rational sequence consumer.

The Real seal closes the route. The row E is reached only after R, and the structural rows H, C, P, N carry the same finite data through transport, replay, provenance, and local naming. Hence the L10 handoff exposes a constructive sequence criterion rather than a selected subsequence, quotient limit, host completion, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet is generated from a sequence source, modulus schedule, dyadic tolerance ledger, stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4224.1, Definition 4224.2, and Definition 4224.3 seed the L10 route from dyadic sequence criteria to regular-Cauchy and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected subsequences, quotient equality of limits, host completion, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus schedule exposure, dyadic tolerance-ledger exactness, stream-window handoff, RegSeqRat/Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCriterionUp.

$\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ routes pointwise locator induction through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38; the child file carries the theorem and closurestatus surface.

Chapter 3245

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 3245.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3245.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 3245.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 3245.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3245.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 3245.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 3245.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3246

A Concrete Naming Certificate for $\mathsf{FrinkMetrization}^{\uparrow}$

Following Frink and Kelley, the $\mathsf{FrinkMetrization}^{\uparrow}$ packet records the finite BEDC route from a uniform entourage basis to a pseudometric-facing and metric-facing distance certificate. It sits after the uniform-source rows of chapter 339, the pseudometric carrier of chapter 1614, the separated metric boundary of chapter 57, the topology compatibility row of chapter 88, the dyadic tolerance ledger of chapter 291, and the finite window interface of chapter 126. The packet names only the constructive handoff from nested entourage data to displayed distance reads; it does not form quotient uniformities, select representatives, assert ambient metrizability, or import choice.

Definition 3246.1 (FrinkMetrization carrier)**.** *A* $\mathsf{FrinkMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, Q, D, P, M, T, H, C, G, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row. It names the carrier surface and the finite entourage requests from which the metrization route starts.*
2. *Q is the nested entourage chain. It records the finite refinement ladder used to compare two source points at dyadic scales.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. It translates the refinement level into displayed dyadic radii and triangle budgets.*
4. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *handoff row. It receives the finite entourage chain and exports the displayed pre-separated distance route.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *compatibility row, available only after the separated boundary has accepted the zero-distance route supplied through P.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *compatibility row. It records that the metric-facing balls and the original uniform entourage reads are compared through the same finite windows.*
7. *H records componentwise* hsame *transport for the uniform source, chain, dyadic ledger, pseudometric handoff, metric compatibility, topology compatibility, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform entourage request through the nested chain, dyadic ledger, pseudometric handoff, metric row, topology row, provenance, and local naming row.*

9. *G is the* Pkg *provenance over* FrinkMetrization$^{\uparrow}$, UniformSpace$^{\uparrow}$, Pseudometric$^{\uparrow}$, Met$^{\uparrow}$, Topology$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, or an unbounded distance oracle.

Lean4 ✓ `BEDC.Derived.FrinkMetrizationUp.FrinkMetrizationTasteGate single carrier alignment`

Theorem 3246.2 (FrinkMetrization NameCert obligations)**.** *For every accepted* FrinkMetrization$^{\uparrow}$ *carrier $F = (U, Q, D, P, M, T, H, C, G, N)$, the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform source, nested entourage chain, dyadic radius ledger, pseudometric handoff, metric compatibility row, topology compatibility row, transport, continuation, provenance, and local naming row;*
2. *nested-chain readability, requiring every distance request to start at the* UniformSpace$^{\uparrow}$ *row U and pass through the finite refinement chain Q before any dyadic radius in D is read;*
3. *pseudometric handoff, requiring the displayed distance route to be exported through P before the metric row M or topology row T can be consumed;*
4. *compatibility transport, requiring H and C to preserve the same finite* StreamName$^{\uparrow}$ *windows and* DyadicRatCore$^{\uparrow}$ *radius ledger across uniform, pseudometric, metric, and topology-facing consumers;*
5. *provenance and refusal, requiring G and N to name only this finite handoff and to refuse quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, and unbounded distance oracles.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4226.1. The projection exposes exactly the uniform source, nested chain, dyadic ledger, pseudometric handoff, metric and topology compatibility rows, and the structural rows H, C, G, N.

The nested-chain obligation is forced by row order. A distance request starts at the uniform row U, enters the refinement ladder Q, and only then reads the dyadic radius ledger D.

The pseudometric handoff is the only distance-export step. Row P supplies the pre-separated displayed distance route, while M and T can compare it with metric and topology-facing consumers only after the same finite windows and radii have been replayed.

Rows H, C, G, N transport, replay, source, and name this route. Since the tuple contains no quotient uniformity, selected representative, ambient metrizability, countable choice, host topology equality, or unbounded distance coordinate, the local NameCert cannot export those principles. □

Closure status for NameCert$_{\text{FrinkMetrization}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, seed)

Scope closed: Definition 4226.1 and Definition 4226.2 seed the finite Frink-style handoff from uniform entourage chains to pseudometric, metric, topology, dyadic, and stream-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, unbounded distance oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-chain readability, dyadic radius exactness, pseudometric handoff, metric and topology compatibility, transport, continuation, provenance, and ledger refusal for BEDC.Derived.FrinkMetrizationUp.

Constructive story: A FrinkMetrization$^{\uparrow}$ packet is generated from a UniformSpace source row, a nested entourage chain, a DyadicRatCore radius ledger, a Pseudometric handoff, Metric and Topology compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

NestedShrinkInterval$^{\uparrow}$ reads its compact interval dependency through chapter 5553 and the child `namecert_construction.tex` route before exposing its shrinking-diameter Real$^{\uparrow}$ seal.

Chapter 3247

A Concrete Naming Certificate for $\mathsf{NestedShrinkInterval}^{\uparrow}$

Following Cantor's nested-interval argument and Bishop–Bridges regular real analysis, the $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet records the finite BEDC carrier surface for a descending located interval route whose displayed diameters shrink to the requested tolerance. It sits between the nested-closed interval carrier of **??**, the located endpoint row of chapter 394, the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names only the finite shrinking-interval convergence certificate; it does not claim open-cover compactness, trichotomy, quotient reals, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3247.1 (Nested shrink interval carrier)**.** *A* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (J, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row, exposing the retained descending closed located interval family and its containment witnesses.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row for the inspected interval windows, including lower and upper endpoint location data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger recording endpoint comparisons, diameter estimates, and tolerance requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect interval endpoints and diameter rows at displayed stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same stream windows and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the nested interval, located endpoint, dyadic shrink, stream-window, and regular readback rows are visible.*
7. *H records componentwise* hsame *transport for J, L, D, S, R, E and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$J, L \longmapsto D, S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{NestedShrinkInterval}^{\uparrow}$, $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $J, L, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient-real equality, a selected intersection point, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3247.2 (Nested shrink interval NameCert obligations)**.** *For every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier*

$$K = (J, L, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row,* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same displayed rows; diameter-to-tolerance exactness through D; finite-window readback through S and R; Real-seal handoff through E; and ledger nonescape for the excluded compactness, trichotomy, quotient, selection, choice, and ambient-completeness coordinates.*

Proof. Project the carrier of Definition 4227.1. The projection exposes exactly the nested closed interval source, located endpoint row, dyadic shrink ledger, finite windows, regular readback, real seal, and structural rows.

Classifier stability is componentwise on those displayed rows. The shrink estimate is not an external metric fact: it is the D row, read together with the endpoint row L and the stream schedule S before regular readback is available.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, every accepted NameCert obligation is one of the rows listed above. □

Theorem 3247.3 (Nested shrink interval diameter route)**.** *Let $K = (J, L, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. Every completion-facing request for the shrinking interval route first reads the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source J and* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row L, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger D and* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, and only then opens the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and* $\mathsf{Real}^{\uparrow}$ *seal E. The route exports diameter-to-tolerance convergence as displayed finite ledger data, not as open-cover compactness, trichotomy, quotient-real equality, countable choice, a selected intersection point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing request and project the accepted carrier. The only interval source is J, and the only endpoint data are the located rows in L.

The diameter request is then forced through D. Its dyadic comparisons and tolerance rows are inspected through the finite schedule S, so a consumer cannot replace the shrinking route by an unrecorded compactness or completeness theorem.

After the finite windows are displayed, R packages the same observations as regular rational readback and E supplies the terminal real seal. The remaining rows H, C, P, N preserve the route and add no excluded coordinate. □

Theorem 3247.4 (Nested shrink interval Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet receives the seal only through the displayed route*

$$\mathsf{NestedClosedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff is stable under the componentwise hsame *transport and displayed* Cont *replay rows* H, C*, and it is sourced and named only by* P, N*.*

Proof. Apply Definition 4227.3. A Real-facing consumer reaches E only after the nested interval source, located endpoint row, dyadic shrink ledger, stream windows, and regular readback have all been displayed.

Transport through H preserves the row roles, and replay through C preserves their order. Provenance and local naming through P, N attach the handoff to the packet without adding analytic content.

Thus the Real seal cannot escape to a selected point, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient real completeness. □

Theorem 3247.5 (Nested shrink interval regular-Cauchy uniqueness route)**.** *Let* $K = (J, L, D, S, R, E, H, C, P, N)$ *and* $K' = (J', L', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packets. Suppose their nested-source rows, located endpoint rows, dyadic shrink ledgers, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise related by the displayed* hsame *transport. Then the two packets reach the same* $\mathsf{Real}^{\uparrow}$*-facing seal route only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *uniqueness surface displayed by* R, R' *before Real sealing. The route does not identify selected intersection points, quotient real representatives, or ambient host real equality.*

Proof. Project both carriers. The first carrier exposes $J, L, D, S, R, E, H, C, P, N$, and the second exposes $J', L', D', S', R', E', H', C', P', N'$.

Apply the diameter route of Definition 4227.3 to both packets. Each read is forced through its nested source, located endpoints, dyadic shrink ledger, finite stream windows, and regular rational readback before a Real-facing seal is visible.

Use the assumed componentwise hsame transport on J, L, D, S, R and J', L', D', S', R'. The endpoint comparison therefore lives on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, because the preceding rows have already fixed the same nested-source, located-endpoint, dyadic, and stream-window route.

Finally apply the Real-seal handoff of Definition 4227.4 to both packets. It permits E and E' to be read only after the regular readback rows are displayed, so the uniqueness comparison is a $\mathsf{RegSeqRat}^{\uparrow}$ route before Real sealing and not a selected intersection point, quotient real equality, or ambient completeness assertion. □

Theorem 3247.6 (Cantor intersection nested-shrink route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer that reads a nested shrinking interval first consumes the* $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows exposed by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. The downstream route supplies the nested-shrink boundary used by chapter 6999 and the Cantor-intersection surface of chapter 1421.*

The route is a forward lattice edge only. It does not select an intersection point, assert open-cover compactness, use countable choice, identify quotient real names, or provide ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4227.1. The exposed rows are the nested closed source J, located endpoints L, dyadic shrink ledger D, finite stream windows S, regular readback R, real seal E, and the structural rows H, C, P, N.

Apply Definition 4227.4. Any Real-facing downstream read reaches E only after the nested, located, dyadic, stream, and regular-rational rows have been displayed.

The Cantor diagonal and Cantor-intersection chapters consume exactly this kind of finite shrinking-interval boundary before their own diagonal or intersection rows are read. Since the

carrier contains no selected point, open-cover compactness row, choice schedule, quotient equality, or ambient completeness coordinate, the downstream route remains a finite dependency edge. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet is generated from a NestedClosedInterval source row, LocatedInterval endpoint rows, a DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4227.1, Definition 4227.2, Definition 4227.3, and Definition 4227.4 seed the finite L10 shrinking-interval convergence route through nested closed intervals, located endpoint rows, dyadic diameter ledgers, stream windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient reals, selected intersection points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, diameter-to-tolerance exactness, finite-window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedShrinkIntervalUp.

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet consumes the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 before a completion-facing reader treats shrinking finite windows as a diagonal Cauchy subsequence witness. The nested source, located endpoint, dyadic diameter, and $\mathsf{StreamName}^{\uparrow}$ rows still supply the finite shrinking windows; $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies only the displayed diagonal handoff needed to pass those windows toward $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The route exports no detached selector, quotient-real equality, countable-choice schedule, selected intersection point, or ambient completeness coordinate. The shrinking interval diameter route is consumed through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38. Thus a downstream L10 consumer reads $\mathsf{NestedShrinkInterval}^{\uparrow}$ only after the finite diameter ledger has been repacked as a $\mathsf{RegSeqRat}^{\uparrow}$ route and sealed as a $\mathsf{Real}^{\uparrow}$ boundary, not as an ambient completeness or selected intersection-point claim. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ after the shrinking-diameter ledger has been repacked through the $\mathsf{StreamName}^{\uparrow}$ observation schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal route. This sibling dependency records the completion-criterion consumer path of chapter 1476: a nested shrink interval can be read as completion evidence only after its finite diameter data has passed through the same stream, regular sequence, dyadic tolerance, and Real seal rows.

Theorem 3247.7 (Nested shrink interval Cauchy completion consumer)**.** *Every completion-facing read of* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *consumes* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *only after the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are visible. Thus the dependency on chapter 1476 is a theorem-level consumer edge through chapter 126 and chapter 301, not an appeal to an undisplayed completion object.*

Proof. Read a completion-facing request for a nested shrink interval. The route first displays the finite observation schedule and regular rational readback rows, so the consumer sees only finite windows and readback entries.

The dyadic tolerance ledger and terminal Real seal are then exposed before the completion criterion is consumed. Since the criterion is reached after those rows, the dependency records a reusable finite route rather than a hidden selected limit or ambient completeness principle. □

Theorem 3247.8 (Nested shrink interval rational locator route)**.** *Every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *route whose endpoint localization is read through rational endpoints first consumes the rational locator carrier of chapter 4202. Only after that locator row is displayed may the consumer read the*

nested shrink rows and then the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows. The route exports no selected completed point and no Heine–Borel compactness principle.*

Proof. Read the accepted packet through the carrier theorem of chapter 4227. The only endpoint localization rows available to the nested shrink route are finite displayed rows, so a rational-endpoint consumer must obtain its rational interval locator row before the shrink ledger can be interpreted.

The sibling chapter chapter 4202 supplies exactly that locator surface. Once it is consumed, the nested shrink packet may proceed through its displayed shrink ledger, finite windows, regular rational readback, and terminal real seal.

No selected completed point or Heine–Borel compactness row is present in either carrier. The dependency route therefore records only the rational locator read before the nested shrink and $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff rows are read. □

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ route consumes chapter 4202 for rational endpoint bracketing and chapter 301 for regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Thus the nested shrink interval dependency lattice reads locator cells first, then the shrinking-window ledger, then the regular readback route, not a selected intersection point or ambient completeness witness. $\mathsf{NestedShrinkInterval}^{\uparrow}$ exposes a completion-facing interval read only after the packet has consumed the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ diagonal row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational route, the terminal $\mathsf{Real}^{\uparrow}$ seal, the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ admission row, and the $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint-bracketing row. The public read is therefore a finite sibling route through diagonal selection, regular sequence readback, Real sealing, completion admission, and rational interval location; it exports no selected intersection point, quotient-real equality, countable choice, or ambient completeness principle. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the completion criterion surface of chapter 1476 before any interval-limit seal is visible. The shrinking-diameter ledger must first pass through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, regular rational readback, and the same Cauchy-completion admission route; only then may a downstream L10 consumer inspect the $\mathsf{Real}^{\uparrow}$ boundary, and the packet still exports no selected intersection point, quotient-real equality, countable choice, or ambient completion. The $\mathsf{NestedShrinkInterval}^{\uparrow}$ carrier depends on the substantive sibling lattice through chapter 320 for the nested interval spine, chapter 301 for regular rational readback, and chapter 38 for the terminal seal. The rational locator and Cauchy-completion routes already represented by the child files above supply the endpoint-bracketing and completion-admission rows, so the nested-shrink packet is read as a finite dependency route rather than a selected intersection point, quotient-real equality, countable choice, or ambient completeness principle.

Theorem 3247.9 (Nested shrink interval compact interval route)**.** *Every compact-interval consumer of an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet must read the compact real interval sibling of chapter 5553 and the nested interval convergence sibling of chapter 2057 through the same finite endpoint, diameter,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows displayed by the shrinking packet. The compact interval route supplies endpoint and finite-net context; the nested convergence route supplies the shrinking-window handoff; neither route adds an open-cover compactness theorem, selected limit point, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier of Definition 4227.1. The visible analytic rows are the nested source, located endpoints, dyadic diameter ledger, finite stream windows, regular rational readback, and Real seal, followed by componentwise transport, replay, provenance, and local naming.

Use the compact real interval packet of chapter 5553 only for the endpoint and finite-net context. Its compact rows are read before the shrinking packet is exported as an interval-local dependency, so the shrink route never obtains an unlisted open-cover or subcover selector.

Use the nested interval convergence packet of chapter 2057 for the convergence-facing handoff. That sibling reads finite endpoint windows, shrinking ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ sealing before a completion consumer may inspect the route.

The structural rows hsame, Cont, Pkg, and NameCert keep the two sibling reads attached to the same displayed packet. Hence a compact-interval dependency edge from $\mathsf{NestedShrinkInterval}^{\uparrow}$ is exactly the stated finite route and not a hidden compactness, quotient, choice, or completeness principle. □

Theorem 3247.10 (NestedShrinkInterval Cantor downstream route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer of a* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet reads the nested source only through nested closed intervals, located endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter witnesses,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The downstream route exports a Cantor diagonal or intersection-facing interval certificate, but it does not select a limit point, assert ambient completeness, identify quotient-real names, or add a host compactness principle outside the displayed rows.*

Proof. Project the nested-shrink carrier through the child routes named by the hub. The compact-interval and rational-locator rows provide nested closed intervals and located endpoints, while the dyadic and StreamName rows expose only finite diameter and window observations.

The Cantor-facing consumer then passes through the RegSeqRat and Real rows already used by the completion and sibling routes. Since these rows carry finite readback and terminal naming rather than a chosen point, the consumer obtains only the displayed interval certificate.

No carrier coordinate names a selected limit point, host completeness theorem, quotient equality, or ambient compactness object. Thus $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ consume $\mathsf{NestedShrinkInterval}^{\uparrow}$ only through the finite interval route stated above. □

Chapter 3248

A Concrete Naming Certificate for $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy names and finite modulus extraction, the $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ packet records the threshold-selection surface that is actually inspected by a regular real-name consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular-rational readback of chapter 301, the ordinary modulus row of chapter 292, and the terminal real-name seal of chapter 38. The chapter names only the finite adaptive route from requested tolerances to inspected windows, regular readback, and Real consumers; it does not choose a global limit, quotient stream representative, ambient complete real, or unbounded modulus oracle.

Definition 3248.1 (Adaptive Cauchy modulus carrier)**.** *An* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the finite name windows available to a regular Cauchy request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested positive dyadic thresholds and the refinements used by the threshold search.*
3. *T is the adaptive threshold schedule, assigning to each displayed tolerance request the finite stage bound selected from the inspected evidence.*
4. *W is the inspected-window row, listing exactly the stream windows opened while validating the schedule T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the source stream, tolerance ledger, threshold schedule, and inspected windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-rational readback has been displayed.*
7. *H records componentwise* hsame *transport for $S, D, T, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from an adaptive-modulus request through the same source, tolerance, schedule, window, readback, seal, transport, provenance, and naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets whose threshold read is mediated by $S, D, T, W, R, E, H, C, P, N$. There is no coordinate for a selected limit, quotient stream equality, host real equality, ambient completeness, countable choice, or an uninspected global modulus.

Lean4 ✓ `BEDC.Derived.AdaptiveCauchyModulusUp.AdaptiveCauchyModulusTasteGate single carrier alignment`

Theorem 3248.2 (Adaptive Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier*

$$A = (S, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$ *surface consists exactly of source-stream admission, dyadic tolerance-ledger exposure, adaptive threshold-schedule admission, inspected-window replay, regular-rational readback, terminal Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of A.*

No obligation in the packet is discharged by a selected Cauchy limit, quotient equality of streams, host real equality, ambient completeness, countable choice, or an oracle for a modulus outside the displayed finite windows.

Proof. Project the carrier of Definition 4228.1. The analytic rows are precisely the source stream S, tolerance ledger D, adaptive threshold schedule T, inspected windows W, regular readback R, and terminal Real seal E.

The access order is forced by the rows. A consumer first reads a dyadic request from D, then uses T to select a finite stage bound, and then opens only the windows recorded by W in the source stream S.

The readback rows add no external completion principle. The row R repacks the inspected stream and tolerance data as a regular rational sequence, and E seals that readback for Real-facing consumers only after R is present.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no coordinate of A stores a selected limit, quotient equality, host equality, ambient completeness, countable choice, or an uninspected global modulus, the displayed rows exhaust the local NameCert obligations. □

Theorem 3248.3 (Adaptive Cauchy modulus RegSeqRat handoff)**.** *Every regular Cauchy or Real consumer of an accepted* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports only the source stream, tolerance ledger, adaptive threshold schedule, inspected windows, regular-rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local name carried by the same packet.

Proof. Start with the NameCert obligations of Definition 4228.2. They force every accepted consumer to read the source stream, dyadic tolerance ledger, threshold schedule, and inspected windows before any regular-rational output is available.

Next attach the ordinary modulus interface. The schedule row T is the finite adaptive presentation of the requested $\mathsf{CauchyModulus}^{\uparrow}$ behavior, and the inspected windows W record the evidence used to justify each displayed threshold.

The regular and Real rows then close the handoff. The row R is the only displayed path from the inspected data to $\mathsf{RegSeqRat}^{\uparrow}$, and the seal E is the only displayed path from R to $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N preserve that same route through transport, replay, provenance, and local naming. Hence a consumer reaches RegSeqRat and Real data through finite inspected evidence rather than through quotient streams, selected limits, host equality, ambient completeness, countable choice, or a detached modulus oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ packet is generated from a source stream, dyadic tolerance ledger, adaptive threshold schedule, inspected stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4228.1, Definition 4228.2, and Definition 4228.3 seed the finite adaptive threshold-selection route for regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, host real equality, ambient completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adaptive threshold schedule exposure, inspected-window replay, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.AdaptiveCauchyMod

Chapter 3249

A Concrete Naming Certificate for $\mathsf{ClosedSet}^{\uparrow}$ Metric Boundary

Following Kelley and Munkres on closed subsets as complements of open neighborhoods and as metric limit boundaries, the $\mathsf{ClosedSet}^{\uparrow}$ packet records the finite BEDC naming surface for closed-set reads. It sits between the topological carrier of chapter 88, the metric and separated-metric routes of chapter 57 and chapter 1411, the terminal real-name seal of chapter 38, and the compact-image consumer of chapter 4162. The packet names a displayed closed-boundary ledger; it does not import a host powerset, a detached open-set lattice, selected limit points, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3249.1 (ClosedSet carrier)**.** *A* $\mathsf{ClosedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, M, S, F, O, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *carrier row containing the finite neighborhood and complement observations used by the closed-set read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row used when the closed boundary is inspected by distance or ball data.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row that turns metric zero-distance comparisons into separated boundary evidence.*
4. *F is the finite membership classifier for the named subset boundary.*
5. *O is the open-neighborhood complement ledger, recording the displayed route from nonmembership to a neighborhood outside the set.*
6. *B is the metric complement boundary row, recording the finite ball and real-distance readbacks through* $\mathsf{Real}^{\uparrow}$ *that witness the same complement behavior.*
7. *H records componentwise* hsame *transport for $T, M, S, F, O, B, C, P, N$.*
8. *C records displayed* Cont *replay from topology reads through metric and separated-metric observations, open-neighborhood complement rows, metric boundary rows, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^\uparrow}$ *row.*

The classifier compares accepted packets only through $T, M, S, F, O, B, H, C, P, N$. It has no coordinate for host powersets, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient Real completeness, or compact-image closedness beyond the displayed consumer route.

Lean4 (definition) `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetUp`

Theorem 3249.2 (ClosedSet NameCert obligations)**.** *For every accepted* $\mathsf{ClosedSet}^\uparrow$ *carrier*

$$K = (T, M, S, F, O, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedSet}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topological row, metric row, separated-metric row, membership classifier, open-neighborhood complement ledger, metric complement boundary, transport, replay, provenance, and local naming;*
2. *topology compatibility, requiring every closed-set read to expose the* $\mathsf{Topology}^\uparrow$ *neighborhood and complement rows before the membership classifier is consumed;*
3. *metric compatibility, requiring any ball or distance read to pass through* $\mathsf{Met}^\uparrow$, $\mathsf{SeparatedMetric}^\uparrow$, *and the* $\mathsf{Real}^\uparrow$ *distance seal before it is treated as boundary evidence;*
4. *complement exactness, requiring nonmembership evidence to be routed through O and metric boundary evidence to be routed through B over the same classifier F;*
5. *transport replay, requiring H and C to preserve the ordered route componentwise before P and N are read;*
6. *nonescape, excluding host powersets, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient* $\mathsf{Real}^\uparrow$ *completeness, and a proof of compact-image closedness not routed through chapter 4162.*

No local ClosedSet NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4229.1. The tuple displays the topology, metric, separated-metric, membership, complement, metric-boundary, transport, replay, provenance, and local naming rows.

Topology compatibility is forced by the placement of T and F. A closed-set read first opens the finite neighborhood and complement observations, and only then consumes the classifier for the named subset boundary.

Metric compatibility is forced by M, S, B. Any distance or ball evidence passes through the metric row, the separated zero-distance row, and the $\mathsf{Real}^\uparrow$ distance seal before it becomes boundary evidence.

The remaining rows preserve and name the same route. Since the carrier has no coordinate for host powersets, detached open-set lattices, selected limit points, quotient equality, choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetTasteGate single carrier alignment`

Theorem 3249.3 (ClosedSet metric complement boundary)**.** *Every accepted* $\mathsf{ClosedSet}^\uparrow$ *carrier routes a metric complement request through*

$$\mathsf{Topology}^\uparrow \longmapsto \mathsf{Met}^\uparrow, \mathsf{SeparatedMetric}^\uparrow, \mathsf{Real}^\uparrow \longmapsto F, O, B, H, C, P, N.$$

Thus a consumer such as $\mathsf{CompactImageClosed}^{\uparrow}$ *may read a named closed boundary only after the topological complement ledger, metric distance rows, separated classifier, real-distance seal, transport, replay, provenance, and local naming rows are displayed. A request bypassing one of those rows is outside the accepted* $\mathsf{ClosedSet}^{\uparrow}$ *packet.*

Proof. Read the accepted carrier $K = (T, M, S, F, O, B, H, C, P, N)$. The topological boundary is T, and the metric boundary is read only through M, S and the real-distance data stored in B.

The classifier F is shared by the open-neighborhood complement ledger O and the metric complement boundary B. Therefore the two complement presentations cannot diverge into a host powerset or detached open-set lattice.

Transport and replay are then supplied by H and C, while P and N record provenance and local naming. Since the compact-image consumer of chapter 4162 receives only that finite boundary row, it cannot obtain selected limit points, quotient equality, countable choice, or ambient Real completeness from the closed-set packet. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetCarrier metric complement boundary`

Theorem 3249.4 (ClosedSet scope basis)**.** *The* $\mathsf{ClosedSet}^{\uparrow}$ *scope is fixed by the carrier and metric-complement boundary rows over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, hsame, Cont, *and the local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *surface. A consumer may read closed-boundary data only after the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise transport, displayed replay, provenance, and local naming rows have been exposed. The scope refuses host powerset semantics, detached open-set lattices, quotient equality of names, selected limit points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier of Definition 4229.1. Its displayed rows contain exactly the topology, metric, separated-metric, classifier, complement, metric-boundary, transport, replay, provenance, and local naming data.

Apply the NameCert obligations of Definition 4229.2 and the metric complement boundary route of Definition 4229.3. These two results force every closed-boundary read through $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, hsame, Cont, and $\mathsf{NameCert}$ before any consumer can inspect the boundary.

Because the accepted packet has no coordinate for a host powerset, detached open-set lattice, quotient equality, selected limit point, or ambient completeness theorem, the stated rows exhaust the ClosedSet scope basis. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetScopeBasis`

Theorem 3249.5 (ClosedSet public boundary export)**.** *The public* $\mathsf{ClosedSet}^{\uparrow}$ *export is exactly the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *row of the accepted carrier. A public consumer may cite the closed-boundary packet only through those rows and through the scope basis; it receives no host powerset semantics, detached open-set lattice, selected limit point, quotient equality of names, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or compact-image closedness outside the displayed consumer route.*

Proof. Project the carrier of Definition 4229.1. It exposes precisely the topology, metric, separated-metric, classifier, complement ledger, metric-boundary, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations, metric complement boundary, and scope basis from Definition 4229.2, Definition 4229.3, and Definition 4229.4. These results force every public closed-boundary read through the same finite row inventory and block any route through host powersets, detached open-set lattices, quotient equality, selected limit points, choice, or ambient completeness.

Thus the public export is not a larger closure theorem; it is the named boundary surface already displayed by the carrier and its metric complement route. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetPublicBoundaryExport`

Theorem 3249.6 (ClosedSet bridged boundary export)**.** *Any future standard bridge for* $\mathsf{ClosedSet}^{\uparrow}$ *must read exactly the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *row already exported by Definition 4229.5. A bridge consumer may replay those rows as the closed-boundary surface, but it receives no host powerset semantics, detached open-set lattice, selected limit point, quotient equality of names, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or compact-image closedness outside that displayed consumer route.*

Proof. Start with the carrier of Definition 4229.1. Its coordinates are exactly $T, M, S, F, O, B, H, C, P, N$, so a bridge has no additional coordinate from which to read a host powerset, detached open-set lattice, selected point, quotient equality witness, choice principle, or ambient completeness theorem.

Apply the NameCert obligations of Definition 4229.2 and the metric complement boundary of Definition 4229.3. These force topology exposure through T, metric exposure through M, S, B, complement compatibility through F, O, B, and replay through H, C, P, N before the closed-boundary row can be consumed.

The scope basis of Definition 4229.4 and the public export of Definition 4229.5 then identify the bridge-facing surface with the already public row inventory. Thus the bridge reads the displayed consumer route and nothing stronger, including no compact-image closedness theorem beyond the named downstream route. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetBridgedBoundaryExport`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedSet}^{\uparrow}$ packet is generated from topological, metric, separated-metric, membership-classifier, open-neighborhood complement, metric complement boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 4229.1, Definition 4229.2, Definition 4229.3, Definition 4229.4, Definition 4229.5, and Definition 4229.6 constitute the mature paper-side treatment of the finite closed-set carrier, local NameCert obligation surface, metric complement boundary route, scope basis, public boundary export, and bridge-facing export over the displayed topology, metric, separated-metric, complement, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host powerset semantics, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or compact-image closedness outside the displayed consumer route.

Upgrade path: Verification progress requires Lean-side coverage for the mature carrier admission, topology compatibility, metric compatibility, complement exactness, complement-boundary transport, metric-boundary replay, transport replay, nonescape, provenance exposure, local naming, and TasteGate field faithfulness for BEDC.Derived.ClosedSetUp.

Chapter 3250

A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{Met}^{\uparrow}$ packet records the traditional metric-space interface as finite BEDC NameCert data. It sits over the generic distance-witness surface of chapter 57, the real-valued distance route of chapter 1638, the dyadic tolerance core of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter gives completion, topology, total-boundedness, compactness, and Polish-space consumers a single cited carrier surface without importing quotient equality, selected limits, countable choice, or ambient completeness.

Definition 3250.1 (MetricSpace carrier)**.** *A* $\mathsf{Met}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-carrier row. It names the displayed point observations and the finite windows through which downstream consumers may inspect them.*
2. *σ is the point-classifier row. It records the accepted point equivalence used by* hsame *transport and by the zero-distance classifier handoff.*
3. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance row. It is read through the* $\mathsf{Met}^{\uparrow}$ *witness surface of chapter 57 and the* $\mathsf{RealMetric}^{\uparrow}$ *distance route of chapter 1638.*
4. *Z is the zero-distance classifier row, requiring any request that turns distance zero into point equivalence to pass through σ rather than through host equality.*
5. *S is the symmetry row, carrying the componentwise reversal of endpoint observations and the corresponding distance readback.*
6. *T is the triangle row, carrying the three endpoint-pair distance reads and the dyadic tolerance budget used to compose them.*
7. *H records componentwise* hsame *transport for the point carrier, classifier, distance, zero-distance, symmetry, triangle, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through the finite route*

$$X, \sigma \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \sigma, D, Z, S, T, H, C, P, N$. It has no coordinate for host equality of points, quotient completion equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached topology.*

Lean4 ✓ `BEDC.Derived.MetricSpaceUp.MetricSpaceTasteGate single carrier alignment`

Theorem 3250.2 (MetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{Met}^{\uparrow}$ *carrier*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the point carrier, point classifier, real-valued distance, zero-distance classifier, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *distance visibility, requiring every distance read to expose the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealMetric}^{\uparrow}$ *rows before any downstream consumer reads a real-valued metric comparison;*
3. *zero-distance exactness, requiring zero distance to return to the displayed classifier σ through Z and never through host point equality;*
4. *symmetry replay, requiring endpoint reversal to be witnessed by S together with the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback route;*
5. *triangle-budget replay, requiring the three distance reads and their dyadic comparison budget to be displayed by T before a triangle consumer may use the row;*
6. *structural preservation through H, C, P, N.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4230.1. An accepted packet displays the point-carrier row X, the classifier σ, the distance row D, the zero-distance row Z, the symmetry row S, the triangle row T, and the structural rows H, C, P, N.

Distance visibility is forced by the placement of D in the carrier. A distance consumer first opens the generic $\mathsf{Met}^{\uparrow}$ witness surface and the $\mathsf{RealMetric}^{\uparrow}$ readback route, and only then reads the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows needed for a finite comparison.

The zero-distance, symmetry, and triangle clauses are rowwise. The row Z is the only route from distance zero back to the point classifier, S is the only endpoint-reversal row, and T is the only triangle-budget row.

Finally, H, C, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export host equality, quotient completion equality, selected representatives, countable choice, ambient completeness, or a detached topology. □

Theorem 3250.3 (MetricSpace distance route)**.** *Every completion, topology, total-boundedness, compactness, or Polish-space consumer of an accepted* $\mathsf{Met}^{\uparrow}$ *carrier reads distance information through the finite route*

$$X, \sigma, D \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, C, P, N.$$

Thus the public distance surface is the displayed BEDC route from point observations and classifier data to real-valued metric reads, zero-distance classification, symmetry, and triangle replay. It exports no quotient equality, selected limit, countable-choice schedule, ambient completeness theorem, or open-set lattice.

Proof. Let a downstream consumer inspect an accepted carrier $M = (X, \sigma, D, Z, S, T, H, C, P, N)$. The point surface is opened first through X and σ, so every later distance read remains attached to the displayed point classifier.

The distance row D then routes the request through $\mathsf{Met}^{\uparrow}$ and $\mathsf{RealMetric}^{\uparrow}$. The numerical readback is finite because the dyadic tolerance, stream-window, regular-rational, and real-name rows are displayed before any metric comparison is consumed.

Metric laws are read only after this route is visible. Zero distance returns through Z to σ, symmetry reverses endpoints through S, and triangle use exposes the three pairwise reads and dyadic budget through T.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since none of the excluded objects appears in the carrier, no listed downstream consumer can read quotient equality, a selected limit, a choice schedule, ambient completeness, or an open-set lattice from the $\mathsf{Met}^{\uparrow}$ packet. □

Chapter 3251

A Concrete Naming Certificate for $\mathsf{NashEquilibrium}^{\uparrow}$

Following Nash's equilibrium theorem, the $\mathsf{NashEquilibrium}^{\uparrow}$ packet records the finite best-response surface for a displayed finite normal-form game before any mixed strategy, payoff comparison, or equilibrium witness is treated as a host object. It sits after the compact metric and convex strategy-simplex surfaces of chapter 129 and chapter 208, the real payoff rows of chapter 38, and the fixed-point anchors of chapter 2030 and chapter 3511. The chapter names the finite BEDC carrier needed by later game-theoretic consumers without exporting a host simplex, selected optimizer, quotient mixed-strategy equality, semantic continuity, or an ambient fixed-point theorem.

Definition 3251.1 (Nash equilibrium carrier)**.** *A* $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, S, A, U, B, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite player ledger. It records the inspected player names, their classifier rows, and the finite dependency order in which player-indexed strategy and payoff rows are read.*
2. *S is the finite strategy-simplex ledger. For each player it displays the finitely generated mixed-strategy simplex through compact metric and convex-set rows, together with the finite coordinate windows consumed by the game packet.*
3. *A is the action-window ledger. It records, for each player, the finite pure-action windows that underlie the displayed simplex coordinates and the finite product windows inspected by payoff and best-response consumers.*
4. *U is the payoff-row ledger. It reads payoff comparisons only through displayed* $\mathsf{Real}^{\uparrow}$ *rows attached to finite action profiles and finite mixed-strategy windows.*
5. *B is the best-response ledger. It records the finite argmax or best-response correspondence rows as displayed comparison data over P, S, A, U, without selecting a host optimizer.*
6. *E is the equilibrium row. It states that the displayed mixed-strategy profile is read through B as a fixed point of the best-response ledger, with the fixed-point route exposed only through the cited Brouwer or Kakutani packet.*
7. *H records componentwise* hsame *transport for player ledgers, simplex rows, action windows, payoff rows, best-response rows, equilibrium rows, replay routes, provenance rows, and local naming rows.*

8. *C records displayed* Cont *continuation from a finite game request through the player, simplex, action, payoff, best-response, fixed-point, transport, provenance, and local naming rows.*
9. *Q is the* Pkg *provenance over* $\mathsf{NashEquilibrium}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It stores no host mixed-strategy simplex, no selected maximizer, no quotient equality of strategies, no semantic payoff continuity, no countable choice, and no ambient fixed-point principle.

Lean4 ✓ `BEDC.Derived.NashEquilibriumUp.NashEquilibriumTasteGate single carrier alignment`

Theorem 3251.2 (Nash equilibrium NameCert obligations)**.** *For every accepted* $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier*

$$G = (P, S, A, U, B, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$ *surface consists exactly of finite player admission, finite strategy-simplex exposure, action-window exposure, payoff-row readability, best-response ledger exposure, equilibrium-row fixed-point routing, componentwise* hsame *transport, displayed* Cont *continuation,* Pkg *provenance, and local naming. No NameCert obligation is stored outside the rows of Definition 4231.1.*

Proof. Project the accepted carrier. The player, strategy, action, payoff, best-response, and equilibrium data are precisely the rows P, S, A, U, B, E.

The finite game surface is read first. The player ledger P indexes the displayed strategy-simplex rows S, while the action-window ledger A supplies the finite product windows inspected by payoff and best-response consumers.

The payoff and best-response obligations are then read through U and B. Payoff comparisons consume only the displayed $\mathsf{Real}^{\uparrow}$ rows, and the best-response ledger records comparison and correspondence data without adding a selected host optimizer.

Finally the equilibrium row E is consumed through the displayed fixed-point route, and the structural rows H, C, Q, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate beyond these rows, the listed surface exhausts the local NameCert obligations. □

Theorem 3251.3 (Nash equilibrium best-response coverage)**.** *Let*

$$G = (P, S, A, U, B, E, H, C, Q, N)$$

be an accepted $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier. Every equilibrium-facing read of E must first expose the player ledger P, the finite strategy-simplex rows S, the action windows A, the payoff rows U, and the best-response ledger B. The read may cite* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, *and* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *only through those displayed rows and the structural rows H, C, Q, N; it exports no host simplex, selected best response, quotient strategy equality, semantic continuity, countable choice, or ambient fixed-point theorem.*

Proof. Begin with the carrier definition. The equilibrium row E is downstream of P, S, A, U, B, so no equilibrium-facing consumer can inspect E before the finite player, strategy, action, payoff, and best-response ledgers have been displayed.

The compact and convex rows enter only through S, and the real payoff comparisons enter only through U. The fixed-point route is recorded by the best-response and equilibrium rows rather than by a separate host theorem coordinate.

Apply the NameCert obligation theorem. It makes transport, continuation, provenance, and local naming part of the same displayed route, so any consumer of E receives only the finite best-response certificate.

The excluded host simplex, selected optimizer, quotient equality, semantic continuity, choice, and ambient fixed-point objects are absent from the carrier inventory. Thus the best-response coverage route is exhausted by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$.

Constructive story: A $\mathsf{NashEquilibrium}^{\uparrow}$ packet is generated from a finite player ledger, finite strategy-simplex rows, action windows, payoff Real rows, best-response ledger, equilibrium fixed-point row, componentwise hsame transport, displayed Cont continuation, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4231.1, Definition 4231.2, and Definition 4231.3 seed the finite best-response certificate surface over player, simplex, action, payoff, best-response, equilibrium, transport, continuation, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host mixed-strategy simplices, selected optimizers, quotient strategy equality, semantic payoff continuity, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite simplex exposure, action-window coverage, payoff-row readability, best-response ledger exactness, equilibrium-row nonescape, transport stability, continuation stability, provenance stability, and local naming for BEDC.Derived.NashEquilibriumUp.

$\mathsf{CauchyFilterSubbasis}^{\uparrow}$ consumes the generated $\mathsf{FilterBase}^{\uparrow}$ route of chapter 1601 before handing generated requests to chapter 333.

Chapter 3252

A Concrete Naming Certificate for $\mathsf{CauchyFilterSubbasis}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ packet records the standard subbasis-to-filter generation surface needed before a completion-facing Cauchy filter can be read as a finite BEDC route. It sits between the filter-base interface of chapter 1601, the Cauchy-filter packet of chapter 333, the completion boundary of chapter 1569, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite generation interface only; it does not quotient filters, select limits, invoke countable choice, use host equality, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

The separated uniform completion of chapter 7566 and the effective regular real extensionality surface of chapter 7009 consume this subbasis route only after $\mathsf{FilterBase}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$ generation are displayed and the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows remain visible. The consumer route is therefore a finite generated-base handoff, not an isolated subbasis generator.

Definition 3252.1 (Cauchy filter subbasis carrier)**.** *A* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite subbasis-token row naming the primitive neighbourhood requests available to the packet.*
2. *I is the finite-intersection row, recording which finite families of subbasis tokens are admitted as generated basic requests.*
3. *E is the generated-event row, displaying the* BHist *surface attached to each finite intersection request.*
4. *G is the generation ledger, storing the inclusion reads from subbasis tokens into generated base events and the closure reads from finite intersections into base requests.*
5. *B is the generated* $\mathsf{FilterBase}^{\uparrow}$ *row exposed to downstream filter consumers.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row obtained from the generated base together with the displayed Cauchy comparison requests.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a completion-facing consumer reads generated tails.*

8. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row for the generated Cauchy comparisons.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for retained regular rational windows.*
10. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the generated base, Cauchy row, window row, tolerance ledger, and readback row have been displayed.*
11. *H records componentwise* hsame *transport for subbasis tokens, finite intersections, generated events, generation ledgers, filterbase rows, Cauchy rows, windows, tolerances, readbacks, Real seals, replay, provenance, and local naming.*
12. *C records displayed* Cont *replay from subbasis requests through finite intersections, generated filterbase rows, Cauchy-filter rows, finite windows, tolerance accounting, readback, and Real sealing.*
13. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$ *row.*

The classifier reads only $U, I, E, G, B, F, W, T, R, L, H, C, P, N$. It has no coordinate for an ultrafilter extension, quotient equality of filters, selected limits, countable-choice schedules, host equality of generated sets, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyFilterSubbasisUp.CauchyFilterSubbasisUp`

Theorem 3252.2 (Cauchy filter subbasis NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier $S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$ *surface consists exactly of carrier admission, finite subbasis-token exposure, finite-intersection generation, generated-event admission, generation-ledger coverage,* $\mathsf{FilterBase}^{\uparrow}$ *handoff,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4232.1. The visible generation data are exactly the rows U, I, E, G, so a consumer first reads primitive tokens, finite intersections, generated events, and the inclusion ledger.

The generated filter route is then B, F, W, T, R, L. The $\mathsf{FilterBase}^{\uparrow}$ row is exposed before the $\mathsf{CauchyFilter}^{\uparrow}$ row, and the finite windows, dyadic tolerance ledger, regular rational readback, and Real seal are downstream from that generated filter route.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the classifier contains no coordinate for an ultrafilter extension, quotient filter equality, selected limit, countable-choice schedule, host equality, or ambient completeness, the local NameCert surface is exactly the displayed finite packet. □

Theorem 3252.3 (Cauchy filter subbasis filterbase generation)**.** *Every completion-facing read exported by an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier factors through the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B before it reaches the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the* $\mathsf{Real}^{\uparrow}$ *seal row L. Thus subbasis generation is visible as finite BEDC data rather than as a hidden host filter construction.*

Proof. Begin with the generation rows of Definition 4232.1. The only route from primitive subbasis tokens to a downstream filter read is through U, I, E, G, and the generated base is stored as the row B.

Apply the NameCert obligation theorem. It requires every Cauchy-filter and completion-facing consumer to inspect the generated $\mathsf{FilterBase}^{\uparrow}$ handoff before reading F, W, T, R, L.

The structural rows preserve the same route. Componentwise transport H, displayed replay C, provenance P, and local naming N keep the subbasis-to-filter generation inside the finite packet.

No excluded coordinate remains. Hence quotient filters, selected limits, countable-choice schedules, host equality of generated filters, and ambient completeness cannot be used in place of the displayed $\mathsf{FilterBase}^{\uparrow}$ generation row. □

Theorem 3252.4 (Cauchy filter subbasis generated-base cofinality)**.** *Every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier exposes the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B as cofinal among the finite intersections named by I and their displayed generated-event rows E. A completion-facing consumer must therefore read U, I, E, G, B before it reads the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal L.*

Proof. Project the carrier of Definition 4232.1. The only finite-intersection data are I, the only generated events are E, and the only inclusion and coverage reads from those rows into a base are stored in G.

Apply the NameCert obligations of Definition 4232.2. They force generated-event admission, generation-ledger coverage, and the $\mathsf{FilterBase}^{\uparrow}$ handoff to be displayed as local certificate rows of the same packet.

Use the filterbase generation theorem Definition 4232.3. Since every completion-facing read must factor through B before reaching F, W, T, R, L, the generated base is cofinal for the finite intersections visible to the packet.

The classifier has no coordinate for quotient filters, selected limits, countable choice, host equality of generated filters, or ambient completeness. Hence no external filter object can replace the displayed cofinal generated-base row. □

Theorem 3252.5 (Cauchy filter subbasis finite-intersection route)**.** *Every generated base request in an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier is read through the finite subbasis-token row U, the finite-intersection row I, the generated-event row E, and the generation ledger G before the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B is available. The route exports no quotient filters, selected limits, countable choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 4232.1. The rows U, I, E, G are the only displayed route from primitive subbasis tokens to generated base events.

Apply the NameCert obligations of Definition 4232.2. They require finite subbasis-token exposure, finite-intersection generation, generated-event admission, and generation-ledger coverage before the $\mathsf{FilterBase}^{\uparrow}$ handoff is a local certificate row.

Use generated-base cofinality, Definition 4232.4. Since every completion-facing consumer must read U, I, E, G, B before the downstream rows, the generated base request is available only after the finite-intersection route has been displayed.

The excluded coordinates are absent from the carrier. Hence quotient filters, selected limits, countable choice, host equality, and ambient completeness cannot replace the rows U, I, E, G, B. □

Theorem 3252.6 (Cauchy filter subbasis refinement stability)**.** *Suppose an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier replaces its generated base row B by a displayed cofinal* $\mathsf{FilterRefinement}^{\uparrow}$ *row. Then the completion-facing route still reaches the same* $\mathsf{CauchyFilter}^{\uparrow}$ *row F,* $\mathsf{StreamName}^{\uparrow}$ *windows W,*

$\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R, *and* $\mathsf{Real}^{\uparrow}$ *seal* L. *The replacement exports no quotient filter, selected limit, countable-choice schedule, host equality of generated filters, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the generated-base cofinality theorem. It identifies B as the displayed cofinal base among finite intersections before any Cauchy-filter or completion-facing row can be read.

Apply the Cauchy-preservation and completion-route clauses for $\mathsf{FilterRefinement}^{\uparrow}$, namely Definition 1626.2 and Definition 1626.3. The refinement row may replace a displayed base request by a cofinal target request, but it preserves the Cauchy-filter read through the finite comparison surface.

Compose that refinement route with Definition 4232.3. The downstream rows F, W, T, R, L remain attached to the same subbasis-generated route because the consumer still passes through a displayed cofinal base before reading them.

Finally inspect the excluded coordinates of the carrier. Since quotient filters, selected limits, choice schedules, host equality of generated filters, and ambient Real completeness are absent from $U, I, E, G, B, F, W, T, R, L, H, C, P, N$, the refinement cannot import any of them. □

Theorem 3252.7 (Cauchy filter subbasis completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier must read the rows*

$$U, I, E, G, B, F, W, T, R, L, H, C, P, N$$

in the displayed order. In particular, the generated $\mathsf{FilterBase}^{\uparrow}$ *row* B *is available before the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F, *and the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* W, T, R *are displayed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L.

Proof. Read the carrier named in Definition 4232.1. Its classifier exposes exactly the listed rows, with U, I, E, G providing the finite subbasis tokens, finite intersections, generated events, and generation ledger.

Apply Definition 4232.2. The NameCert obligations require the generated $\mathsf{FilterBase}^{\uparrow}$ handoff, the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, the window row, the tolerance row, the readback row, and the Real seal to appear as displayed fields of the same finite packet.

Finally apply Definition 4232.3. That theorem forces every completion-facing read through B before F, W, T, R, L, so the handoff cannot bypass the generated base or reach the terminal $\mathsf{Real}^{\uparrow}$ seal before the window, tolerance, and readback rows have been displayed. □

Theorem 3252.8 (Cauchy filter subbasis window transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier preserves finite subbasis tokens, finite intersections, generated events, generation ledgers, generated* $\mathsf{FilterBase}^{\uparrow}$ *rows,* $\mathsf{CauchyFilter}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks. The transported packet adds no quotient filter, selected limit, countable-choice schedule, host equality of generated filters, or ambient completeness coordinate.*

Proof. Project the transport row H from Definition 4232.1. It is componentwise over the visible rows, so transport acts on tokens, intersections, generated events, ledgers, generated bases, Cauchy rows, windows, tolerances, readbacks, seals, replay, provenance, and local naming without changing the classifier shape.

Use Definition 4232.2 to identify the rows preserved by the local NameCert surface. Each transported read remains attached to the corresponding displayed obligation, including the generated $\mathsf{FilterBase}^{\uparrow}$ handoff and the downstream Cauchy-filter, window, tolerance, and readback obligations.

Apply Definition 4232.3 to the transported route. Since every downstream completion-facing read still factors through the generated base row, no quotient filter or selected-limit coordinate can be inserted by transport. □

Theorem 3252.9 (Cauchy filter subbasis regular readback)**.** *For an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *is available only after the generated* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T *have been displayed. The* $\mathsf{Real}^{\uparrow}$ *seal* L *remains terminal for the route and cannot be used as a prior source for the readback.*

Proof. The carrier of Definition 4232.1 places R after the generated base, Cauchy-filter, window, and tolerance rows. Thus the readback row is not a primitive subbasis token and is not a replacement for the generated filter route.

By Definition 4232.2, the local certificate displays the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ window exposure, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting, and $\mathsf{RegSeqRat}^{\uparrow}$ readback as distinct obligations. The readback therefore depends on the displayed Cauchy-filter and tolerance ledger surface rather than on an ambient limit.

Apply Definition 4232.3. It keeps L after B, F, W, T, R, so the terminal $\mathsf{Real}^{\uparrow}$ seal cannot justify the readback in advance; it only seals the finite route after the readback row has been displayed. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ completion-facing read factors through finite subbasis tokens, finite intersections, generated events, the generation ledger, a generated FilterBaseUp row, a CauchyFilterUp row, StreamNameUp windows, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: filterbase, cauchyfilter, cauchyfiltercompletion, streamname, regseqrat, real: $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries primitive subbasis tokens, finite-intersection generation, generated events, generation-ledger coverage, generated filterbase handoff, Cauchy-filter handoff, finite windows, tolerance accounting, readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ packet is generated from finite subbasis tokens, finite-intersection rows, generated-event rows, a generation ledger, a generated FilterBaseUp row, a CauchyFilterUp handoff, StreamNameUp windows, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4232.1, Definition 4232.2, and Definition 4232.3 seed the finite subbasis-to-filter generation route over generated filterbase, Cauchy-filter, window, tolerance, readback, seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality of generated filters, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-intersection generation, generation-ledger coverage, generated FilterBaseUp handoff, CauchyFilterUp handoff, window exposure, tolerance accounting, readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterSubbasisUp.

The $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ carrier refines the filter-base row of chapter 1601 and feeds the Cauchy-filter carrier of chapter 333. This is a substantive sibling dependency: the basic-set closure supplied by the generated $\mathsf{FilterBase}^{\uparrow}$ row and the Cauchy-directedness consumed by the downstream $\mathsf{CauchyFilter}^{\uparrow}$ row are separate certificate obligations, not self-references or restatements of the local subbasis packet.

Frechet tail dependency. The $\mathsf{FrechetFilter}^{\uparrow}$ packet of chapter 1564 supplies the cofinite-tail specialization of the subbasis-generated filter route. The specialization is consumed through the displayed base discipline of chapter 1601 before the generated route reaches chapter 333 and the Real-facing seal.

A $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ read that is narrowed to eventual sequence tails therefore factors through the same finite threshold, filter-base, stream-schedule, regular-readback, and Real-seal rows exposed by $\mathsf{FrechetFilter}^{\uparrow}$. This is a displayed sibling dependency in the tail-base factorization route, not a citation invented after the fact, and the subbasis packet retains only the generated Cauchy-filter surface rather than quotient filter equality, selected limits, countable choice, or ambient completion.

Theorem 3252.10 (Cauchy filter subbasis tail-base factorization)**.** *Every* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *read specialized to cofinite sequence tails factors through* $\mathsf{FrechetFilter}^{\uparrow}$ *and* $\mathsf{FilterBase}^{\uparrow}$ *before* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are read. For an accepted carrier*

$$S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N),$$

the tail-specialized route first displays the finite subbasis tokens, finite intersections, generated events, and generation ledger U, I, E, G*; then reads the cofinite-tail threshold discipline of chapter 1564; then returns through the generated* $\mathsf{FilterBase}^{\uparrow}$ *row* B *before the Cauchy row, stream windows, tolerance ledger, regular readback, and Real seal* F, W, T, R, L *can be consumed.*

Proof. Begin with the generated-base cofinality theorem, Definition 4232.4. It places U, I, E, G, B before every completion-facing read, so a tail-specialized consumer cannot reach F, W, T, R, L before the generated base has been displayed.

Specialize the generated requests to eventual sequence tails. The Frechet tail dependency row in this chapter identifies the cofinite-tail discipline supplied by chapter 1564; this row is the only tail-filter surface available to the specialized read.

Compose the specialization with the completion handoff of Definition 4232.7. The handoff routes the resulting base through the $\mathsf{CauchyFilter}^{\uparrow}$ row, finite stream windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

The carrier exclusions then apply to the whole composition. Quotient filters, selected limits, countable choice, host equality of generated filters, and ambient Real completeness are absent from the displayed rows, so none can replace the Frechet and FilterBase stages of the tail-base factorization. □

The hub-level $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ route is read through the finite dependency lattice of chapter 1601, chapter 333, chapter 126, chapter 301, and chapter 38. This route exports no quotient filter, selected limit, countable choice, or ambient completeness. The top-level $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ route consumes the finite $\mathsf{FilterBase}^{\uparrow}$ generation rows of chapter 1601 before it reads the $\mathsf{CauchyFilter}^{\uparrow}$ packet of chapter 333. Only after that base-to-Cauchy handoff does the route pass through the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 126, chapter 301, and chapter 38. This is a finite generation route, not a quotient filter, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Theorem 3252.11 (CauchyFilterSubbasis Cauchy-completion sibling route)**.** *Every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *packet exported to a completion-facing consumer first reads the Cauchy-completion criterion surface of chapter 3885. The required dependency is concrete: subbasis neighborhoods are generated from the finite* $\mathsf{FilterBase}^{\uparrow}$ *rows, the Cauchy filter completion criterion supplies the completion rows,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *provide only finite windows and rational readback, and the terminal*

$\mathsf{Real}^{\uparrow}$ *seal records nonescape from selected limits, quotient filters, countable choice, and ambient completeness.*

Proof. Project the top-level route described above. The subbasis neighborhood row is read before any Cauchy packet is exported, so the consumer begins with finite $\mathsf{FilterBase}^{\uparrow}$ generation and the displayed $\mathsf{CauchyFilter}^{\uparrow}$ handoff.

Next read the Cauchy-completion criterion row. It supplies the completion-facing criterion used by the sibling chapter, while the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows restrict every read to finite stream windows and rational approximants.

The final $\mathsf{Real}^{\uparrow}$ seal is only a nonescape row. It does not select a limit, identify quotient filters, introduce countable choice, or assert ambient completeness beyond the finite criterion route already displayed. □

Theorem 3252.12 (CauchyFilterSubbasis Real-uniform lattice route)**.** *Every Real-uniform consumer of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *packet must pass through finite* $\mathsf{FilterBase}^{\uparrow}$ *generation,* $\mathsf{CauchyFilter}^{\uparrow}$ *completion criteria,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows before any* $\mathsf{RealUniformStructure}^{\uparrow}$ *or completion-facing consumer can read it. The route exports a generated subbasis lattice edge, not quotient filters, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 4232.11. It forces any completion-facing consumer to read the Cauchy-completion criterion only after finite $\mathsf{FilterBase}^{\uparrow}$ generation and the displayed $\mathsf{CauchyFilter}^{\uparrow}$ handoff.

Now compose that sibling route with the generated-base and cofinality reading described in the top-level route. The subbasis row enters the Real-uniform lattice as generated filter-base data, so a $\mathsf{RealUniformStructure}^{\uparrow}$ consumer cannot bypass the finite base or treat the subbasis as an ambient quotient filter.

The final handoff is through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows expose only finite windows, rational readback, and a terminal seal, so no selected limit, countable-choice schedule, quotient-filter equality, or ambient completeness principle is available to the consumer. □

Chapter 3253

A Concrete Naming Certificate for $\mathsf{BishopCompleteReal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompleteReal}^{\uparrow}$ packet records the textbook complete-real interface as a finite BEDC boundary rather than as an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. It sits on the dyadic endpoint ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated completion surface of chapter 1612, and the completion reflection boundary of chapter 403. The packet exposes the L10 completion interface currently needed by Real and Completion consumers; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete, select limits, quotient streams, or import countable choice.

Definition 3253.1 (Bishop complete real carrier)**.** *A* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, E, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger used to measure regular Cauchy requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row opened by the same tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic, schedule, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, recording that two completion-facing reads are compared through separated finite evidence.*
6. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *handoff row, replaying the same regular-rational source surface into the completion-facing interface.*
7. *H stores componentwise* hsame *transport for $D, W, R, E, S, F, C, P, N$.*
8. *C stores displayed* Cont *replay from a completion-facing request through the same dyadic, schedule, regular, real-seal, separated, and reflection rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient stream equality, a selected limit, countable choice, open-cover compactness, or a theorem that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.TasteGate.BishopCompleteRealTasteGate single carrier alignment`

Theorem 3253.2 (Bishop complete real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier*

$$B = (D, W, R, E, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every completion-facing read passes through* $D, W, R, E, S, F, H, C, P, N$ *in that order.*

No obligation in the packet is discharged by host real equality, quotient streams, selected representatives, countable choice, open-cover compactness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 4233.1. The tuple lists the dyadic, schedule, regular readback, Real seal, separated comparison, completion reflection, transport, replay, provenance, and naming rows.

The completion-facing order is fixed by the visible rows. A request first reads a dyadic endpoint and tolerance ledger in D, then opens the finite schedule row W, and then passes through the regular rational readback row R before the terminal Real seal E is available.

The separated and reflection rows then consume that same source surface. The comparison row S records separated evidence, while F replays the regular-rational source into the completion-facing boundary.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded host, quotient, choice, compactness, or completeness data, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.BishopCompleteRealNamecertObligations`

Theorem 3253.3 (Bishop complete real L10 handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *packet factors through the finite L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{SeparatedCompletion}^{\uparrow} \longrightarrow \mathsf{CompletionReflection}^{\uparrow}.$$

The handoff exports a displayed regular-rational completion interface with separated comparison and reflection rows. It does not export completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, selected host limits, countable-choice coordinates, open-cover compactness, or an ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4233.2. They force every accepted read to pass through the dyadic ledger, finite schedule, regular rational readback, and terminal Real seal before any completion-facing row is consumed.

Next read the separated comparison row. The row S compares completion-facing observations only through displayed separated evidence, so it cannot introduce quotient classes or host equality.

The completion reflection row F then replays the same source surface into the completion-facing interface. The structural rows H, C, P, N preserve that route componentwise and add only transport, replay, provenance, and local naming.

Thus the consumer receives exactly the displayed L10 handoff from dyadic data to schedules, regular rational readback, Real seal, separated comparison, and completion reflection. None of the excluded completeness, choice, quotient, compactness, or selected-limit data has an entry point in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompleteReal}^{\uparrow}$ packet is generated from a dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4233.1, Definition 4233.2, and Definition 4233.3 seed the finite Bishop complete-real interface for the current L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, selected host limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated comparison, completion reflection handoff, and ledger non-escape for the same packet.

Chapter 3254

Cauchy compact extraction NameCert construction

Following Bishop and Bridges on compact metric spaces, the $\mathsf{CauchyCompact}^{\uparrow}$ extraction packet records the finite route by which total boundedness and completeness turn regular rational observations into a displayed limit seal. It refines the constructive Cauchy-extraction face of chapter 2117, consumes the regular sequence interface of chapter 301, and keeps the dyadic tolerance bookkeeping visible through chapter 291. The chapter names a finite extraction surface only; it does not assert open-cover compactness, a choice-selected subsequence, or host-level real completeness.

Definition 3254.1 (Cauchy compact extraction carrier)**.** *A* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier is a finite* BHist *packet*

$$E = (M, L, T, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, exposing the metric carrier, local classifier, and compact metric provenance consumed by the extraction.*
2. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, exposing the finite limit-admission route for the regular Cauchy input named by the packet.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row, exposing for each dyadic tolerance request the finite net, mesh ledger, and coverage replay used to choose a bounded search window.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, recording the finite windows of the input stream available before the limit seal is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational handoff obtained from the displayed stream windows and the same tolerance ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for radii, comparisons, refinements, and tail budgets used by T, S, and R.*
7. *A is the* $\mathsf{Real}^{\uparrow}$*-facing extracted seal, reachable only after M, L, T, S, R, D have been displayed.*
8. *H stores componentwise* hsame *transport for the compact metric row, complete metric row, totally bounded row, stream schedule, regular rational handoff, dyadic ledger, extracted seal, continuation, provenance, and local name.*

9. *C stores displayed* Cont *replay of the route*

$$M, L, T, S, R, D \longmapsto A.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *extraction row.*

The classifier compares accepted extraction packets only through these rows. It has no coordinate for open covers, selected subsequences, quotient streams, ambient choice, or a host real object.

Theorem 3254.2 (Cauchy compact extraction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier $E = (M, L, T, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *extraction surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *finite-net visibility, requiring each extraction tolerance read to pass through the displayed* $\mathsf{TotallyBounded}^{\uparrow}$ *row T over the same* $\mathsf{CompactMetric}^{\uparrow}$ *row M;*
3. *regular-window compatibility, requiring the* $\mathsf{StreamName}^{\uparrow}$ *windows in S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R to use the same dyadic tolerance data D;*
4. *completeness handoff, requiring the extracted seal A to be reached through the* $\mathsf{CompleteMetric}^{\uparrow}$ *row L after the displayed regular Cauchy route has been admitted;*
5. *ledger confinement, requiring H, C, P, N to transport, replay, source, and name only the displayed extraction route and to export no open-cover compactness, choice-selected subsequence, quotient stream equality, ambient choice, or host real equality.*

Proof. Project the accepted carrier. The packet displays exactly the compact metric row, complete metric row, total-bounded finite-net row, stream schedule, regular rational handoff, dyadic ledger, extracted seal, transport, continuation, provenance, and local naming rows.

Finite-net visibility is read from T over M. A tolerance request reaches the extraction only by reading the dyadic budget in D and the finite net supplied by the displayed total-bounded row.

Regular-window compatibility is forced before the completion step. The stream windows in S and the regular rational handoff R share the same dyadic ledger D, so the completion row cannot read an unnamed Cauchy input.

Finally L produces the seal A and H, C, P, N preserve that same route. Since the tuple contains no row for open covers, selected subsequences, quotient streams, ambient choice, or a host real object, the listed obligations exhaust the local NameCert surface. □

Theorem 3254.3 (Cauchy compact extraction completion handoff)**.** *Every accepted* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier hands compact metric data to the L10 completion route through*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{TotallyBounded}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{CompleteMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the structural rows BHist, hsame, Cont, Pkg, *and* NameCert. *Any consumer of the extracted seal must first read the finite net, the stream window, the regular rational handoff, the dyadic ledger, and the complete metric limit route.*

Proof. Begin with the compact and total-bounded faces. The row M fixes the metric carrier and classifier, while T supplies the finite net and coverage ledger at the dyadic tolerance recorded in D.

Read the input stream next. The row S exposes only finite observation windows, and R repacks those windows with D as the regular rational handoff used by the completion side.

Apply the completeness row last. The row L consumes the displayed regular Cauchy route and produces the seal A, whose $\mathsf{Real}^{\uparrow}$ face is downstream from the same packet data.

The transport, continuation, provenance, and local naming rows preserve this dependency order. Thus the handoff exports a finite completion-facing route and no open-cover, choice-selected, quotient, ambient-choice, or host-real escape. □

Chapter 3255

A Concrete Naming Certificate for $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$

Following Kelley and Bishop–Bridges compact-open and uniform convergence arguments, the $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ packet records the finite BEDC route by which compact-open function-space rows, compact-metric probes, equicontinuity ledgers, and uniform-modulus rows are read as one convergence surface. It sits beside the compact-open function-space carrier of chapter 2221, the compact-open topology surface of chapter 4138, the compact-metric source of chapter 129, and the equicontinuity family route of chapter 4076. The packet is a finite convergence carrier for compact-window consumers; it does not assert host function equality, arbitrary open-cover compactness, selected subnet convergence, quotient maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3255.1 (Compact-open uniform convergence carrier)**.** *A* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, K, E, Q, M, G, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ *source row, exposing the finite function-space window to be tested.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact probe row that supplies the displayed compact window.*
3. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family row over the same compact probe.*
4. *Q is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row, recording the shared compact-window radius schedule.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *convergence threshold consumed by the downstream convergence read.*
6. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the displayed maps on the compact probe.*
7. *D is the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *target-distance ledger used to compare displayed values.*
8. *R is the compact-open replay row, requiring every compact-window convergence read to pass through F, K, E, Q, M, G, D before any public handoff.*
9. *H stores componentwise* hsame *transport for $F, K, E, Q, M, G, D, R, C, P, N$.*
10. *C stores displayed* Cont *replay from compact-open requests through the same finite convergence route.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$, $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for host equality of functions, arbitrary compact-open topology construction, selected subnets, quotient maps, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactOpenUniformConvergenceUp.CompactOpenUniformConvergenceTasteGate`
`single carrier alignment`

Theorem 3255.2 (Compact-open uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier*

$$U = (F, K, E, Q, M, G, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-open function-space row, compact-metric probe row, equicontinuity row, uniform-equicontinuity modulus row, uniform-modulus threshold row, continuous-map graph row, metric and real target-distance row, compact-open replay row, componentwise transport, displayed continuation replay, provenance, and local naming.*

Every compact-open convergence consumer must read the compact-window probe K and the shared modulus rows Q, M before reading the target-distance ledger D or the handoff row R. No obligation in the packet is discharged by host function equality, selected subnet convergence, quotient map equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 4235.1. The tuple lists the compact-open source, compact probe, equicontinuity, uniform-modulus, continuous-map, metric, real, replay, transport, provenance, and naming rows.

The compact-window order is fixed by the visible route. A consumer first reads F through the compact probe K, then reads the equicontinuity row E and shared modulus rows Q, M before any target-distance comparison in D is available.

The replay row R packages that order for compact-open convergence consumers. The structural rows H, C, P, N preserve the same finite route by componentwise transport, continuation replay, provenance, and local naming.

Since the carrier contains no coordinate for the excluded host equality, subnet, quotient, open-cover, or completeness data, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3255.3 (Compact-open uniform convergence finite-net handoff)**.** *Every compact-open topology or Arzela–Ascoli-style consumer of an accepted* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactOpenFunctionSpace}^{\uparrow} \longrightarrow \mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{Equicontinuity}^{\uparrow} \longrightarrow \mathsf{UniformEquicontinuityModulus}^{\uparrow} \longrightarrow \mathsf{Uni}$$

The handoff exports only the displayed compact-window convergence surface, with the shared modulus and target-distance rows preserved by hsame, Cont, Pkg, *and local* NameCert *data.*

Proof. Begin with the NameCert obligations of Definition 4235.2. They force the compact-open consumer to read the compact-window probe, equicontinuity row, shared modulus rows, continuous-map graph row, and target-distance ledger before the replay row.

Next read the finite compact source through chapter 129 and the equicontinuity route through chapter 4076. These sibling surfaces supply finite probes and shared modulus data rather than open-cover compactness or selected subnets.

Finally R, H, C, P, N repack the displayed route for compact-open topology and function-space consumers. The carrier contains no host function equality, quotient map, selected subnet, or ambient completeness coordinate, so the public handoff is exactly the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ packet is generated from compact-open function-space rows, compact-metric probe rows, equicontinuity rows, uniform-equicontinuity modulus rows, uniform-modulus thresholds, continuous-map graph rows, metric and Real target-distance ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4235.1, Definition 4235.2, and Definition 4235.3 seed the finite compact-open uniform convergence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, selected subnet convergence, quotient maps, arbitrary open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, shared-modulus exactness, target-distance handoff, replay nonescape, and local naming for BEDC.Derived.CompactOpenUniformConvergenceUp.

Chapter 3256

A Concrete Naming Certificate for DefectFlowPartitionSkeleton$^{\uparrow}$

The DefectFlowPartitionSkeleton$^{\uparrow}$ packet realizes the defect-flow partition skeleton of Definition 9462.1722 as finite cubical-incidence and boundary-flow NameCert data. It sits over the partition carrier of chapter 201, the cellular-update surface of chapter 934, the Boolean row discipline of chapter 3733, the rank-dependence boundary of chapter 202, and the support-nerve surface of chapter 238. The packet names the finite route from partition blocks and boundary incidences to defect-lowering transitions; it is not a host graph-flow oracle, stochastic biological model, quotient codon semantics, unrestricted homology computation, or hidden global scheduler.

Definition 3256.1 (Defect-flow partition skeleton carrier)**.** *A* DefectFlowPartitionSkeleton$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, I, L, T, A, M, X, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite partition-block row, listing the displayed sink blocks and their support names.*
2. *I is the cubical boundary-incidence row, recording which displayed boundary faces meet each block.*
3. *L is the defect-level ledger, recording the finite defect value attached to each non-sink incidence state.*
4. *T is the defect-flow transition row, listing the allowed defect-lowering moves between displayed states.*
5. *A is the* CellularAutomaton$^{\uparrow}$ *local update row used to replay T as a finite boundary-flow step.*
6. *M is the* Matroid$^{\uparrow}$ *rank-dependence row separating removable shared incidences from protected private coordinates.*
7. *X is the* SimplicialComplex$^{\uparrow}$ *and* BooleanAlgebra$^{\uparrow}$ *support-nerve row, exposing faces, complements, and Boolean incidence tests.*
8. *H stores componentwise* hsame *transport for $B, I, L, T, A, M, X, C, P, N$.*
9. *C stores displayed* Cont *replay from a boundary-flow request through the same finite rows.*
10. *P is the* Pkg *provenance over* DefectFlowPartitionSkeleton$^{\uparrow}$, Partition$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, BooleanAlgebra$^{\uparrow}$, Matroid$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$ *row.*
The carrier has no coordinate for unrestricted flow search, host graph reachability, quotient codon semantics, biological stochastic semantics, hidden scheduler state, or unrestricted homology.

Theorem 3256.2 (Defect-flow partition skeleton NameCert obligations)**.** *For every accepted* $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ *carrier*

$$S = (B, I, L, T, A, M, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$ *surface consists exactly of carrier admission for partition blocks, cubical boundary-incidence rows, defect-level ledgers, defect-flow transitions, cellular replay, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise transport, displayed continuation replay, provenance, and local naming.*

A boundary-flow consumer may read a transition in T only after the source block row B, incidence row I, and defect ledger L have been displayed, and it may classify a sink only through the partition row and the support-nerve row. No obligation in the packet is discharged by host reachability, unrestricted flow search, biological stochastic semantics, quotient codon semantics, or hidden global scheduling.

Proof. Carrier admission is projection from Definition 4236.1. The displayed tuple lists the partition, incidence, defect, transition, cellular replay, rank-dependence, support-nerve, transport, continuation, provenance, and naming rows.

The transition order is fixed by the finite route. A consumer first reads the partition blocks B and cubical incidences I, then reads the defect ledger L, and only then may inspect the defect-lowering transition row T.

The cellular, matroid, Boolean, and simplicial rows explain how the same finite transition is replayed and classified. They provide local update, protected-coordinate separation, complement tests, and support-nerve exposure without importing a host flow oracle.

Finally H, C, P, N keep all reads inside the displayed packet. Since no excluded reachability, stochastic, quotient, scheduler, or unrestricted homology coordinate is present, the stated obligations exhaust the local NameCert surface. □

Theorem 3256.3 (Defect-flow partition skeleton TasteGate boundary)**.** *Any future TasteGate surface for* $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ DefectFlowPartitionSkeleton}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ DefectFlowPartitionSkeleton}^{\uparrow}$ *instances preserving $B, I, L, T, A, M, X, H, C, P, N$ as separate displayed fields. In particular, the gate must distinguish a changed partition block, a changed boundary-incidence row, a changed defect-flow transition, and a changed support-nerve row before it admits the packet as the same skeleton.*

Proof. Use the carrier of Definition 4236.1. The rows B, I, L, T, A, M, X play different roles in the finite boundary-flow route: source blocks, boundary incidences, defect values, transitions, cellular replay, rank separation, and support-nerve classification.

The NameCert obligations of Definition 4236.2 consume those rows in order. Identifying two of them would erase either the source, the flow step, or the classification boundary, so a field-faithful gate must preserve the displayed coordinates separately.

Nontriviality is witnessed by packets that vary one of the displayed rows while keeping the structural rows fixed. Such packets are observable at the carrier boundary and therefore cannot be collapsed by an admissible TasteGate read. □

Lean4 ✓ `BEDC.Derived.DefectFlowPartitionSkeletonUp.DefectFlowPartitionSkeletonTasteGate single carrier alignment`

Falsifiable prediction: Every accepted $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ read must expose partition blocks, cubical boundary-incidence rows, defect-level ledgers, defect-flow transitions, a cellular replay row, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a boundary-flow consumer reads a partition sink.

Independence witness: partition, cellularautomaton, booleanalgebra, matroid, simplicialcomplex: none alone is bijective with $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ because the packet jointly carries partition blocks, boundary incidences, defect transitions, update replay, rank separation, support-nerve tests, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$.

Constructive story: A $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ packet is generated from partition-block rows, cubical boundary-incidence rows, defect-level ledgers, defect-flow transition rows, cellular replay, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4236.1, Definition 4236.2, and Definition 4236.3 seed the finite defect-flow partition skeleton packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted flow search, biological stochastic semantics, quotient codon semantics, unrestricted homology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition exactness, sink classification, Nontrivial DefectFlowPartitionSkeletonUp, and FieldFaithful DefectFlowPartitionSkeletonUp for BEDC.Derived.DefectFlowPartitionSkeletonUp.

The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet routes completion-facing reads through chapter 301, chapter 126, chapter 291, and chapter 38 before the choice-free completion body files are read.

Chapter 3257

A Concrete Naming Certificate for $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet records the choice-free regular-Cauchy route to a completion-facing real seal as finite BEDC data. It sits after the regular rational readback surface of chapter 301, the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the separated comparison surface of chapter 1463, the public reflector route of chapter 1585, and the terminal real-name seal of chapter 38. The packet names a non-quotient completion route; it does not choose representatives, quotient regular sequences, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, or introduce countable choice.

Definition 3257.1 (Regular-Cauchy choice-free completion carrier)**.** *A* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name and its displayed readback requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite windows through which the regular name is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows and to the regular-Cauchy modulus requests.*
4. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row, recording separated comparison of two displayed completion routes without quotienting them.*
5. *F is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row, exposing unit, counit, idempotent replay, universal factorization, and reflector transport as public finite data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, U, F have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, tolerance, comparison, reflector, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a regular-Cauchy request through $R, W, D, U, F, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, W, D, U, F, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3257.2 (Regular-Cauchy choice-free completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier*

$$K = (R, W, D, U, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *surface consists exactly of regular rational source exposure through R, finite window exposure through W, dyadic tolerance exposure through D, separated completion comparison through U, reflector routing through F, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4237.1. Rows R, W, D are the regular-Cauchy observation spine, and rows U, F, E are the completion-facing comparison, reflector, and real-seal data.

The observation spine is read before the reflector row is consumed. The $\mathsf{RegSeqRat}^{\uparrow}$ source is inspected only through $\mathsf{StreamName}^{\uparrow}$ windows, and the dyadic tolerance ledger fixes the finite precision route available to a completion consumer.

The comparison and reflector rows stay displayed. Row U records separated comparison of completion routes, while F exposes the public unit, counit, idempotent, extension, and reflector faces without adding a quotient carrier or a selected limit.

Finally H, C, P, N transport, replay, source, and name exactly the same packet. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3257.3 (Regular-Cauchy choice-free real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exposes regular readback, finite windows, dyadic tolerance, separated comparison, and the public reflector surface before the terminal real seal is consumed. It does not close ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient regular sequences, selected representatives, countable choice, host equality of completed points, or open-cover compactness.*

Proof. Begin with the obligation surface of Definition 4237.2. It forces every real-facing consumer through the source, window, tolerance, comparison, reflector, and seal rows.

The sibling rows supply the route in order. The regular source is read through chapter 301; finite access is through chapter 126; dyadic tolerance is through chapter 291; separated comparison is through chapter 1463; and public reflection is through chapter 1585.

The terminal row E is therefore only the displayed $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Because the carrier has no quotient, selector, choice, host-equality, compactness, or ambient-completeness coordinate, none can be exported by the seal. □

This choice-free packet is the finite source route consumed by chapter 1425. Its $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{CompletionUniqueness}^{\uparrow}$ comparison, $\mathsf{CauchyCompletionReflector}^{\uparrow}$ row, and $\mathsf{Real}^{\uparrow}$ seal provide the source-side data for the universal extension and uniqueness ledger, while retaining the same non-quotient and no-choice boundary.

Theorem 3257.4 (Regular Cauchy choice-free completion StreamName anchor)**.** *Every accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *completion read factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window row and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The read is therefore anchored in chapter 126, chapter 301, and chapter 38, with W listing the inspected windows, R supplying the regular rational source rows, and E serving only as the downstream real-name seal.*

Proof. Project the carrier of Definition 4237.1. The rows R, W, E are displayed before any completion-facing consumer can read the real seal.

Apply the real-seal route theorem Definition 4237.3. It places $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ upstream of the $\mathsf{Real}^{\uparrow}$ terminal row and keeps the completion comparison and reflector rows downstream of the same finite observation surface.

Thus every accepted read factors first through the sibling StreamName and RegSeqRat anchors, and the Real row is only the terminal seal of that displayed route. □

Theorem 3257.5 (Regular-Cauchy choice-free cofinal-window refinement)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. If a completion-facing precision request for K is served by a displayed* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row, then the request refines through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before the real seal is consumed. The refinement preserves the StreamName request rows, DyadicRatCore tolerance rows, RegSeqRat classifier and regularity rows, the reflector row F, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *support.*

No quotient stream equality, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle can replace the displayed cofinal-window surface in this refinement.*

Proof. Begin with Definition 4237.2. It projects K onto the displayed source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and naming rows.

Use Definition 4060.2 for the precision request. The cofinality row exposes the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance comparison, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the cofinal schedule before any completion-facing consumer can read the request.

Finally apply Definition 4237.3. The real-seal route places the reflector row F and the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite observation surface, so the cofinal-window refinement is consumed as displayed BEDC data and not as quotient streams, selected limits, countable choice, host equality, compactness, or ambient completeness. □

Theorem 3257.6 (Regular-Cauchy cofinal-window exactness)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every completion-facing precision request for* K *served by a* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row is exact only through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The exactness surface is the finite StreamName window row, the DyadicRatCore tolerance row, the RegSeqRat source readback, the reflector row F*, and the terminal real seal* E*. It cannot be replaced by quotient stream equality, selected limits, countable-choice schedules, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 4237.1. The only completion-facing rows available from K are the displayed source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows.

Apply Definition 4060.2 to the cofinal-window request. It exposes the finite window and tolerance data as BEDC rows before the request can be consumed by a completion-facing read.

Finally apply Definition 4237.3. Since the real seal is reached only after the same displayed route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$, the cofinal-window row is the unique completion-facing exactness surface available to the packet. □

Theorem 3257.7 (Regular-Cauchy choice-free window coverage)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Before a completion-facing precision request for* K *is read, the packet exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the displayed* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *cofinal-window rows that serve the request. These rows are the public finite window surface for the request, and no later reflector or real-seal read can bypass them.*

Proof. Project the carrier in Definition 4237.1. Its first three rows are exactly the regular-rational source, finite-window, and dyadic-tolerance rows available before the comparison, reflector, and real-seal rows.

Use Definition 4237.2 to identify those rows as part of the local NameCert surface. For a precision request served by a cofinal-window row, Definition 4237.5 and Definition 4237.6 place the $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ evidence on the same displayed route.

Finally apply the real-seal route theorem Definition 4237.3. Since the terminal seal is downstream of R, W, D and the cofinal-window evidence, every accepted precision read sees that finite window surface before any completion-facing seal is consumed. □

Theorem 3257.8 (Regular-Cauchy choice-free tail reflection)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every tail precision read of* K *reflects through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

preserving the packet rows $R, W, D, U, F, E, H, C, P, N$ *and any* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *support used to serve the request. The reflected read cannot be replaced by quotient stream equality, selected limits, countable-choice schedules, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4237.1. The tail precision read can inspect only the source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 4237.2. They bind the accepted packet to the rows $R, W, D, U, F, E, H, C, P, N$ and give no separate completion object.

Use Definition 4237.6 and Definition 4237.7. The cofinal support exposes the finite RegularCauchyWindowCofinal rows beside the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the reflector and real seal are consumed.

Finally apply the real-seal route of Definition 4237.3. The reflected read reaches $\mathsf{CauchyCompletionReflector}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only downstream of that finite tail-window surface, so quotient stream equality, selected limits, countable choice, host equality, open-cover compactness, and ambient completeness cannot replace it. □

Theorem 3257.9 (Regular-Cauchy choice-free reflector nonescape)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. The* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row* F *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *export no quotient representative, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle. They only expose the displayed reflector route and the downstream real-name seal already named by the packet.*

Proof. Start with Definition 4237.2. The obligation surface lists F as a displayed reflector row and E as a terminal real-seal row, with no carrier coordinate for quotient representatives, selected limits, choice schedules, host equality, compactness, or ambient completeness.

The route theorem Definition 4237.3 keeps both rows downstream of the regular source, finite windows, dyadic tolerance, and separated comparison rows. Thus F can only route the displayed completion data, and E can only seal the resulting real-name boundary.

The cofinal-window exactness theorem Definition 4237.6 rules out replacing the displayed window surface by any excluded completion principle. Hence neither the reflector row nor the real-seal row exports data outside the carrier's finite route. □

Theorem 3257.10 (Regular-Cauchy choice-free metric-reflector handoff)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every metric-completion or reflector consumer of* K *must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row* U*, the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row* F*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row* N*.*

The handoff exports no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier and apply Definition 4237.2. The obligation surface names exactly the rows $R, W, D, U, F, E, H, C, P, N$, so a completion-facing consumer has no access to any coordinate outside the displayed packet.

Use the real-seal route of Definition 4237.3. It places the regular rational source, stream windows, dyadic tolerance ledger, separated comparison row, and reflector row upstream of the terminal $\mathsf{Real}^{\uparrow}$ seal.

For precision-sensitive reads, apply Definition 4237.6. The finite window and tolerance surface remains the exact completion-facing route before the reflector and real seal are consumed.

Finally apply Definition 4237.9. Since F and E export only the displayed reflector route and downstream real-name seal, the handoff refuses quotient completion, selected limits, countable choice, host equality, open-cover compactness, and ambient real completeness. □

Theorem 3257.11 (Regular-Cauchy choice-free reflector universal factorization)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every completion-facing consumer of* K *factors first through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow}$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal is read. The factorization preserves the rows* $R, W, D, U, F, E, H, C, P, N$*, the cofinal-window support used by the request, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *naming.*

No completion-facing consumer may bypass this route by exporting quotient completion data, selected limits, countable-choice schedules, host equality of completed objects, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4237.1. The source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows are exactly $R, W, D, U, F, E, H, C, P, N$.

Apply the real-seal route of Definition 4237.3. It orders every terminal $\mathsf{Real}^{\uparrow}$ read through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ before the seal row E is consumed.

Use the metric-reflector handoff of Definition 4237.10. It shows that metric-completion and reflector consumers receive the same displayed packet rows rather than a separate completion object.

Finally use Definition 4237.12. Componentwise transport, displayed replay, provenance, and local naming preserve the route instead of adding quotient, selector, choice, host-equality, compactness, or ambient-completeness data. □

Theorem 3257.12 (Regular-Cauchy choice-free transport and replay preservation)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row* N *preserve exactly the packet rows* R, W, D, U, F, E *together with their displayed cofinal-window support. They add no quotient completion principle, selected limit, countable-choice schedule, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier definition. Rows H, C, P, N are not new completion data; they are the transport, replay, provenance, and local naming rows attached to the displayed source, window, tolerance, comparison, reflector, and seal rows.

Apply Definition 4237.2. It says the local NameCert surface consists exactly of $R, W, D, U, F, E, H, C, P, N$, so transport and replay preserve the same packet rows rather than extending the classifier with an extra completion coordinate.

Use Definition 4237.3 and Definition 4237.7 for the downstream route. The same finite observation surface reaches the reflector and real seal after componentwise transport, displayed replay, provenance, and local naming, and the excluded completion principles never appear in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ read exposes RegSeqRat source, StreamName window, DyadicRatCore tolerance, CompletionUniqueness comparison, CauchyCompletionReflector route, RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing real seal is consumed.

Independence witness: regseqrat, streamname, dyadicratcore, completion-uniqueness, cauchy-completionreflector, real: none is bijective with $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ because this packet alone binds the regular-Cauchy source, finite-window tolerance spine, separated comparison row, public reflector route, and terminal RealUp seal without quotient completion or countable choice.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet is generated from RegSeqRat source rows, StreamName windows, DyadicRatCore tolerance rows, CompletionUniqueness comparison rows, CauchyCompletionReflector route rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4237.1, Definition 4237.2, and Definition 4237.3 seed the finite choice-free regular-Cauchy completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed objects, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, dyadic tolerance exactness, separated comparison, reflector routing, RealUp seal nonescape, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyChoiceFreeCompletionUp.

Completion-dense extension consumer. $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ supplies the choice-free $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route consumed by chapter 6382. The downstream extension packet reads this route as finite completion-facing data and receives no quotient completion representative, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 3257.13 (Regular-Cauchy choice-free dense-extension route)**.** *Every* $\mathsf{CompletionDenseExtension}^{\uparrow}$ *consumer of a* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet first reads the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows exposed by that packet. The consumer receives the route as finite completion-facing data and receives no quotient representative, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 4237.2. They expose the source row R, window row W, tolerance row D, comparison row U, reflector row F, terminal seal row E, and the transport, replay, provenance, and local naming rows before any downstream completion read is available.

Apply the real-seal route of Definition 4237.3. It orders the same displayed route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The downstream packet is chapter 6382. Since the upstream carrier contains no quotient representative, selected limit, countable-choice schedule, or ambient completeness coordinate, the dense-extension consumer can only receive the displayed finite rows. □

Theorem 3257.14 (Regular-Cauchy choice-free completion universal route)**.** *Every* RegularCauchyCompletionUnive *consumer of a* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet first reads the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row,* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row before using the universal extension or uniqueness ledger. The universal consumer receives only those source-side finite rows together with the componentwise transport, displayed replay, provenance, and local naming rows of the choice-free packet.*

Proof. Start with the dense-extension route of Definition 4237.13. It exposes the regular source, finite windows, dyadic tolerance, completion comparison, reflector row, and real seal before any downstream completion-facing consumer can read the packet.

Use the real-seal route of Definition 4237.3. The terminal $\mathsf{Real}^{\uparrow}$ row is reached only after the same displayed source-side data, so the universal consumer cannot substitute quotient representatives, selected limits, choice schedules, or ambient completeness.

Finally apply the universal extension and public export surfaces, Definition 1425.2 and Definition 1425.10. They consume the displayed source, window, dyadic, comparison, and real-seal rows before exporting the extension route and uniqueness ledger, which is exactly the stated universal route from the choice-free packet. □

$\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ consumes the metric-completion dependency surface of chapter 1506 through the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal route already bound by its carrier and real-seal theorems. The packet receives this route as finite completion-facing data and receives no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 3257.15 (Regular Cauchy choice-free sibling lattice route)**.** *Every accepted* RegularCauchyChoiceFreeCom *packet consumes chapter 1506 and chapter 1776 only through the displayed sibling lattice*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exports finite completion-facing data for the choice-free carrier, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected limit, choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the metric-completion sibling first. Its completion-facing surface is consumed only after the regular rational source row, stream-window row, and dyadic tolerance row have been displayed by the choice-free packet.

Read the Bishop completion induction sibling next. The induction route is attached to the same $\mathsf{CompletionUniqueness}^{\uparrow}$ comparison and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ route before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed.

The carrier, real-seal, transport, replay, provenance, and local naming rows preserve exactly this sibling lattice. Since the packet contains no quotient-completion, selected-limit, choice-schedule,

host-equality, or ambient-completeness coordinate, no consumer can extract those endpoints from the route. □

Theorem 3257.16 (Regular Cauchy choice-free sibling route)**.** *Every accepted* RegularCauchyChoiceFreeCompletion *route consumes chapter 1506, chapter 1776,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as sibling* NameCert *surfaces. The route exposes them only as finite completion-facing rows for regular rational source data, finite stream windows, dyadic tolerance accounting, induction replay, metric-completion dependency, and terminal real sealing. It exposes no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the sibling lattice of Definition 4237.15. The metric-completion and Bishop-induction siblings are admitted only after the regular rational, stream-window, and dyadic-tolerance surfaces have been displayed.

The route then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal through the same completion-facing rows. Since no quotient-completion, selected-limit, choice-schedule, host-equality, or ambient-completeness coordinate is present in the carrier, the listed sibling surfaces exhaust the visible route. □

The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ route is consumed only after the L10 sibling surface has exposed regular rational families, finite stream windows, dyadic tolerance ledgers, and the terminal Real seal. A completion-facing consumer must therefore read chapter 301, chapter 126, chapter 291, and chapter 38 before using the choice-free handoff. The route records finite readback and tolerance data; it supplies no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet consumes chapter 301, chapter 38, and chapter 126 through finite regular-Cauchy windows before any completion-facing seal is read. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational source and readback requests, the $\mathsf{StreamName}^{\uparrow}$ row bounds those requests by displayed windows, and the $\mathsf{Real}^{\uparrow}$ row appears only as the terminal seal reached after the reflector and comparison rows have replayed the same finite data. This route gives downstream completion consumers no quotient representative, selected limit, countable-choice schedule, host equality of completed objects, or ambient real-completeness principle. $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads its local critical-pair and residual-join packet as a consumer of chapter 765, while the body files below keep the carrier, candidate-normalization, residual-substitution, and seed-readiness rows separate.

Chapter 3258

A Concrete Naming Certificate for MetaCICParallelDiamondFrontier$^{\uparrow}$

The MetaCICParallelDiamondFrontier$^{\uparrow}$ packet realizes the residual-substitution frontier of chapter 9378 as finite confluence-facing NameCert data. It sits beside the residual-substitution boundary of chapter 2181, the confluence critical-pair boundary of chapter 2212, the closed checker seal of chapter 2182, and the closed-normal confluence fragment of chapter 737. The packet records the conditional parallel-diamond frontier without claiming residual-substitution compatibility, full closed Church–Rosser, subject reduction, evaluator equality, or kernel consistency.

The retained P, K, J, R, S, C, B, O rows are consumed substantively by the MetaCIC critical-path packet of chapter 765, the confluence audit packet of chapter 735, and the normalization-confluence bridge route of chapter 1190. The critical-path reader consumes the residual-premise and obstruction rows, while the confluence audit packet consumes the candidate-join and bounded-check rows before any confluence consumer may use the frontier. The route exports no full Church–Rosser theorem and no subject-reduction theorem.

Falsifiable prediction: Every accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ confluence-facing read exposes the residual-substitution premise row before any parallel-diamond handoff can be consumed.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, closed-type-checker-seal, metacicconfluencenormalfragment: the residual-substitution boundary names the missing compatibility input without the parallel-diamond handoff; the critical-pair boundary records local peak data without the closed checker seal; the closed type-checker seal records bounded checking without confluence closure; and the closed-normal fragment records unconditional normal-source joins without the residual frontier. None is bijective to MetaCICParallelDiamondFrontier$^{\uparrow}$.

Definition 3258.1 (MetaCIC parallel-diamond frontier carrier)**.** *A* MetaCICParallelDiamondFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

with the following displayed rows:

1. *P is the residual-substitution premise row, naming the substitution compatibility input required by the parallel-diamond route.*
2. *K is the critical-pair row, recording the local peak and its two parallel-reduction observations.*

3. *J is the candidate-join row, recording the proposed common reduct and branch-local route requests.*
4. *R is the residual-frontier row, listing residual descendants, binder-depth exposure, and branch-local substitution windows.*
5. *S is the closed type-checker seal row consumed before a closed confluence-facing read can use the checker data.*
6. *C is the closed-normal fragment row, recording the already available normal-source and atom-source confluence fragments.*
7. *B is the bounded-check row over closedness, support, branch compatibility, and finished-normalization boundaries.*
8. *O is the obstruction row naming the endpoints not supplied by the packet: residual-substitution discharge, full parallel diamond, subject reduction, arbitrary closed Church–Rosser, evaluator equality, and kernel consistency.*
9. *H records componentwise* hsame *transport for premise, peak, join, residual, checker, fragment, bounded-check, obstruction, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay from a confluence query through $P, K, J, R, S, C, B, O, H, G, N$.*
11. *G is the* Pkg *provenance over* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, K, J, R, S, C, B, O, H, T, G, N$. It has no coordinate for a premise-free parallel diamond, subject reduction, arbitrary closed Church–Rosser, unrestricted beta-conversion decidability, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 3258.2 (MetaCIC parallel-diamond frontier obligations)**.** *For every accepted* MetaCICParallelDiamondFro *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *surface consists exactly of residual-substitution premise retention through P, critical-pair exposure through K, candidate-join exposure through J, residual-frontier exposure through R, closed-checker handoff through S, closed-normal fragment exposure through C, bounded-check exposure through B, obstruction retention through O, componentwise transport through H, displayed replay through T, provenance through G, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObligations`

Proof. Project the carrier of Definition 4238.1. The confluence-facing rows are P, K, J, R, S, C, B, O, and the structural certificate rows are H, T, G, N.

The residual-substitution premise row remains first-class. Row P is not discharged by the carrier; it is the named input that the conditional parallel-diamond route must still expose.

The local geometry is then displayed through K, J, R. The critical-pair, candidate-join, and residual-frontier rows provide finite peak data and branch-local residual windows, while the closed checker and closed-normal rows show which existing fragments may be consumed.

Finally B, O, H, T, G, N keep the bounded checks, obstruction endpoints, transport, replay, provenance, and local naming attached to the same packet. Since no omitted theorem appears as a carrier coordinate, the listed rows exhaust the frontier obligations. □

Theorem 3258.3 (MetaCIC parallel-diamond frontier handoff)**.** *Every confluence consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the closed checker seal row S, the closed-normal fragment row C, and the bounded-check row B before any parallel-diamond handoff is read. The handoff may feed the conditional route recorded in chapter 9378, but it does not discharge residual substitution, prove subject reduction, prove arbitrary closed confluence, decide evaluator equality, or establish kernel consistency.*

Proof. Begin with the obligation surface of Definition 4238.2. It forces every confluence read through the named premise, peak, join, residual, checker, fragment, bounded-check, and obstruction rows.

The sibling packets preserve that boundary. The residual-substitution boundary supplies the named premise, the critical-pair boundary supplies local peak data, the closed type-checker seal supplies bounded checker data, and the normal fragment supplies only the closed-normal and atom-source cases already available.

The obstruction row O remains part of the packet. Therefore the handoff is a finite frontier certificate for a conditional route, not a proof of the missing residual-substitution input or any stronger confluence endpoint. □

Theorem 3258.4 (MetaCIC parallel-diamond critical-pair exhaustion)**.** *For every accepted* MetaCICParallelDiamor *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

each residual-substitution critical pair that a confluence consumer may inspect is exposed by the displayed critical-pair row K before the candidate-join row J can be read. The read must retain the residual-substitution premise row P, route the branch-local residual windows through R, pass the closed checker seal S and closed-normal fragment row C, and keep the bounded-check and obstruction rows B, O visible. It exports no unlisted peak, hidden substitution lemma, candidate join detached from K, R, or closed-checker result outside the displayed frontier packet.

Proof. Project the carrier of Definition 4238.1. The only rows that can mention residual substitution, critical pairs, candidate joins, residual windows, closed checking, and obstruction are P, K, J, R, S, C, B, O.

The residual-substitution premise is retained first. Row P is not discharged by the packet, so every critical-pair read remains conditional on the same displayed premise.

The critical-pair exposure is then forced through K. A consumer cannot read J as a candidate join until the peak and its two parallel-reduction observations have already been displayed in K and routed through the branch-local residual windows in R.

Finally the closed-checker handoff is bounded by S, C, B, O. The checker seal, closed-normal fragment, bounded-check row, and obstruction row remain visible through the replay, provenance, and local naming rows T, G, N, so the packet exhausts the accepted critical pairs without adding any hidden confluence endpoint. □

Theorem 3258.5 (MetaCIC parallel-diamond frontier obstruction triad)**.** *Every accepted* MetaCICParallelDiamond *packet retains three independent obstruction classes throughout the conditional local-diamond read:*

residual-substitution premise discharge remains confined to the row P, subject-reduction compatibility remains outside the closed-checker and closed-normal rows S, C, and the full closed Church–Rosser, evaluator equality, quotient term equality, and kernel-consistency endpoints remain recorded only as refused endpoints in the obstruction row O. These obstructions are transported and replayed through H, T, G, N without being collapsed into one another or promoted to completed confluence data.

Proof. Project the carrier rows P, O, B, S, C from Definition 4238.1. Row P is the residual-substitution premise, row O stores the refused endpoints, and rows B, S, C expose only bounded checking, closed checking, and closed-normal fragment data.

Apply the frontier handoff theorem and critical-pair exhaustion, Definition 4238.3 and Definition 4238.4. They keep the local peak under the displayed premise and force the closed-checker route to retain S, C, B, O.

Finally use the route theorem Definition 4238.9. Transport, replay, provenance, and local naming preserve the same rows rather than discharging P, deriving subject reduction from S, C, or turning the obstruction row into Church–Rosser, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.6 (MetaCIC critical-pair obstruction budget)**.** *Every critical-pair read consumed by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond frontier must preserve the residual-substitution premise row P and the obstruction row O through the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, a consumer may not retype the handoff as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the carrier rows of the accepted frontier packet, with P and O displayed as ordinary rows rather than external commentary.

Apply the route theorem Definition 4238.9. Any confluence-audit consumer must expose the critical-pair row K, candidate row J, residual-frontier row R, checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before using the conditional handoff.

Use premise retention, Definition 4238.13. The residual-substitution premise row P remains visible through the critical pair, residual frontier, closed checker, and closed-normal rows whenever the local-diamond handoff is consumed.

Finally apply discharge obstruction, Definition 4238.18. Since no separate residual-substitution compatibility certificate is supplied by this packet, the obstruction row O keeps the refused confluence, typing, evaluator, quotient, and kernel endpoints outside the critical-pair budget. □

Theorem 3258.7 (MetaCIC residual peak scope)**.** *Every candidate-mediated residual peak read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the MetaCIC critical-pair envelope, residual substitution dependency, normalization candidate scope, and closure-trace rows before a diamond frontier read is public. For a carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the visible scope is

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K records the critical-pair envelope, P retains the residual-substitution dependency, J, R, C carry the candidate-normalization and residual-frontier scope, and H, T, G, N preserve the closure trace through transport, replay, provenance, and local naming. The read is a finite frontier boundary; it is not a confluence theorem, a strong-normalization theorem, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. The residual peak has no row outside $P, K, J, R, S, C, B, O, H, T, G, N$ so every public diamond-frontier read must be assembled from that finite spine.

Use critical-pair exhaustion. It displays the local peak through K and prevents the candidate row J from being read before the residual-substitution premise P and branch-local residual row R remain visible.

Next apply the frontier handoff and residual-substitution scope routes already present in the chapter. These keep the candidate-normalization surface inside J, R, S, C, B, O and retain the obstruction row rather than closing the missing residual-substitution dependency.

Finally read the closure trace through H, T, G, N. Transport, replay, provenance, and local naming preserve the same candidate-mediated residual peak for downstream consumers, but none of those rows supplies confluence, strong normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.8 (MetaCIC parallel-diamond frontier seed-record exhaustion)**.** *Seed-facing confluence reads of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet are exhausted by*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

They cannot promote the residual-substitution premise, subject reduction, Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. Its seed-facing rows are exactly $P, K, J, R, S, C, B, O, H, T, G, N$.

Use the obligation surface and handoff theorem, Definition 4238.2 and Definition 4238.3. They keep the residual-substitution premise visible and expose only the conditional frontier route.

Finally cite premise retention, residual-substitution scope, and discharge obstruction. These retain P, scope the candidate frontier, and keep the refused confluence, typing, evaluator, quotient, and kernel endpoints outside the seed record. □

Theorem 3258.9 (MetaCIC parallel-diamond frontier route)**.** *Every confluence-audit consumer that reads an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker seal S, closed-normal fragment C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N before using a conditional parallel-diamond handoff. The route does not turn the residual-substitution premise into a theorem, does not prove full Church–Rosser, and does not discharge subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierRoute`

Proof. Begin with the carrier of Definition 4238.1. It displays exactly the premise, peak, join, residual, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

The obligation theorem Definition 4238.2 keeps P, K, J, R, S, C, B, O as first-class rows. A confluence-audit consumer therefore cannot use the handoff before the missing residual-substitution premise and the obstruction boundary are visible.

The handoff theorem Definition 4238.3 supplies only a conditional frontier route. The added transport, replay, provenance, and local name rows preserve that boundary for downstream audit consumers.

Since no row discharges residual substitution or full closed confluence, the route is a sharper obligation boundary rather than a completed Church–Rosser certificate. □

Theorem 3258.10 (MetaCIC local-diamond frontier)**.** *Let*

$$F_0 = (P, K, J_0, R, S, C, B, O, H, T, G, N) \quad \textit{and} \quad F_1 = (P, K, J_1, R, S, C, B, O, H, T, G, N)$$

be accepted $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packets with the same residual-substitution premise row P and the same critical-pair row K. If the candidate-join rows J_0 and J_1 agree through the displayed residual-frontier row R, then every confluence-audit consumer reads the same conditional local-diamond handoff through S, C, B, O, H, T, G, N. The obstruction row O remains visible in the handoff, so the result does not discharge residual-substitution compatibility, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the two carriers through Definition 4238.1. The hypotheses identify the premise and peak rows P, K, and they make the two candidate-join rows agree at the residual-frontier row R.

Apply Definition 4238.2 to each packet. The only confluence-facing rows available to a consumer before replay are P, K, J, R, S, C, B, O, with the structural rows H, T, G, N attached to the same packet.

Now use Definition 4238.3 and Definition 4238.9. Since P, K agree and J_0, J_1 agree through R, both handoff readings reach the same closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N.

The obstruction row is not removed by this comparison. It is carried in both packets and remains part of the consumer-visible handoff, so the theorem records a local-diamond frontier fact rather than a proof of the missing residual-substitution or closed confluence endpoints. □

Theorem 3258.11 (MetaCIC parallel-diamond residual square)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

exposes a residual square only through the retained residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the closed-checker seal S, the closed-normal fragment C, the bounded-check row B, the obstruction row O, and the transport, replay, provenance, and local naming rows H, T, G, N. Any consumer of the square must therefore factor through those rows before using a conditional local-diamond handoff. The square does not discharge residual substitution, subject reduction, full closed Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier through Definition 4238.1. Its confluence-facing rows are exactly P, K, J, R, S, C, B, O, and its structural rows are exactly H, T, G, N.

Use critical-pair exhaustion, Definition 4238.4. It places the local peak data inside the displayed critical-pair row K and keeps the closed-checker and obstruction rows visible.

Apply the local-diamond frontier theorem Definition 4238.10. A conditional local-diamond read can compare candidate joins only through the residual-frontier row R, with S, C, B, O, H, T, G, N still present in the handoff.

Finally route the resulting read through Definition 4238.9. Since that route retains $P, K, J, R, S, C, B, O, H, T, G$ before any consumer sees the handoff, the residual square is a frontier obligation over the displayed rows rather than any of the refused residual-substitution, typing, confluence, equality, quotient, or kernel endpoints. □

Theorem 3258.12 (MetaCIC parallel-diamond residual-substitution scope)**.** *Every use of the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *local-diamond frontier must pass through the residual-substitution boundary, the confluence critical-pair boundary, and the closed-normal fragment before it claims a local join. In carrier terms, a consumer must expose the residual-substitution premise row P, the critical-pair row K, the residual-frontier row R, the closed checker seal row S, and the closed-normal fragment row C before reading the candidate-join handoff. The resulting claim remains a local frontier claim; it does not close full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Apply the route theorem Definition 4238.9. It requires the residual-substitution premise row, the critical-pair row, the residual-frontier row, the closed-checker seal, and the closed-normal fragment to remain visible before the conditional handoff is consumed.

Then apply the local-diamond frontier theorem Definition 4238.10. The agreement of candidate-join rows is read only through the displayed residual-frontier row, and the obstruction row remains visible in the handoff.

The sibling boundaries named in the chapter introduction supply exactly the surrounding scope: the residual-substitution boundary names the missing compatibility premise, the confluence critical-pair boundary names the local peak surface, and the closed-normal fragment names the already available normal-source joins. Since none of these rows removes the obstruction row, the theorem records a residual-substitution-scoped local join rather than any stronger confluence endpoint. □

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualSubstitutio`

Theorem 3258.13 (MetaCIC parallel-diamond frontier premise retention)**.** *Any accepted* MetaCICParallelDiamond *packet whose local-diamond handoff is consumed retains the residual-substitution premise row P through the critical-pair row K, residual-frontier row R, closed-checker row S, and closed-normal row C. No consumer may reinterpret that handoff as residual-substitution compatibility, premise-free parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Apply the obligations theorem Definition 4238.2. It makes P a first-class residual-substitution premise row and keeps K, R, S, C among the confluence-facing rows rather than structural metadata.

Next use the handoff and route theorems, Definition 4238.3 and Definition 4238.9. They require a consumer to expose the premise, critical-pair, residual-frontier, checker, and closed-normal rows before the conditional handoff is read.

Finally apply the residual-substitution scope theorem Definition 4238.12. Since the local-diamond frontier remains scoped by the residual-substitution boundary and the obstruction row is still visible, the retained handoff cannot be retyped as any of the refused confluence endpoints. □

Theorem 3258.14 (MetaCIC parallel-diamond closed-normal join socket)**.** *Any closed-normal join socket consumed by a MetaCIC confluence audit must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker seal S, closed-normal fragment C, bounded-check row B, obstruction row O, componentwise* hsame *support*

row H, displayed Cont *replay row T,* Pkg *provenance row G, and local* NameCert *row N before any local join is read. The socket retains the residual-substitution premise and obstruction boundary; it does not claim residual-substitution compatibility or full closed Church–Rosser.*

Proof. Start with the obligations and handoff theorems, Definition 4238.2 and Definition 4238.3. They list $P, K, J, R, S, C, B, O, H, T, G, N$ as the confluence-facing packet and require the handoff to stay inside those rows.

Use the route and local-diamond theorems, Definition 4238.9 and Definition 4238.10. These force any local join to pass through the critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, and structural rows before a confluence audit reads it.

Finally apply residual-substitution scope and premise retention, Definition 4238.12 and Definition 4238.13. The premise row P remains visible throughout the closed-normal socket, and O keeps the refused endpoints outside the handoff. □

Theorem 3258.15 (MetaCIC parallel-diamond frontier local-join ledger pullback)**.** *Let*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

be an accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet, and let*

$$L = (K, J, B, R, S, O, G_L, H_L, C_L, P_L, N_L)$$

be an accepted MetaCICLocalJoinLedger$^{\uparrow}$ *packet. Suppose the two packets share the same residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, transport, replay, provenance, and local naming rows, with the frontier rows R, S, C preserving the same residual-substitution scope as the ledger rows R, S. Then every confluence-audit consumer reads the local join through the same residual-substitution-scoped route. The pullback keeps the obstruction row visible, and it does not claim full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the frontier carrier through Definition 4238.1. The accepted packet exposes $P, K, J, R, S, C, B, O, H$, with P and O retained as confluence-facing rows rather than external annotations.

Read the local-join ledger through chapter 4126. The shared rows identify the local peak, candidate join, bounded checker, residual frontier, substitution boundary, obstruction readback, transport, replay, provenance, and local naming surfaces.

Apply the frontier obligations and route theorems, Definition 4238.2 and Definition 4238.9. They force any confluence-audit consumer to expose the premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, and structural rows before using a handoff.

Finally use premise retention, Definition 4238.13. Since the ledger rows are shared with the frontier rows and the obstruction row is preserved on both sides, the consumer-visible local join is read through the same residual-substitution-scoped route, not through an unconditional confluence endpoint. □

Theorem 3258.16 (MetaCIC parallel-diamond typed boundary)**.** *Any accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet whose conditional local-diamond handoff is read under a typed-window premise must retain the typed window, residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal row C, bounded-check row B, and obstruction row O before the handoff is consumed. The packet may not be cited as a premise-free parallel diamond, subject reduction, full Church–Rosser theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Apply the obligations theorem Definition 4238.2. It lists the premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, and obstruction rows as the confluence-facing surface of the accepted packet.

Use the route theorem Definition 4238.9. A typed-window consumer can read the conditional handoff only after those rows have been exposed together with transport, replay, provenance, and local naming.

Apply the residual-substitution scope theorem Definition 4238.12. The typed window is read through the same residual-substitution boundary, confluence critical-pair boundary, and closed-normal fragment that scope the local join.

Finally use premise retention, Definition 4238.13. The retained premise and obstruction rows prevent the handoff from being retyped as a premise-free diamond or any of the refused confluence, equality, or kernel endpoints. □

Theorem 3258.17 (MetaCIC parallel-diamond frontier typed-window handoff)**.** *Any typed-window consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must read the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before feeding candidate-normalization or confluence-audit consumers. In this route P remains the residual-substitution premise, O remains the obstruction row, and the handoff cannot be consumed as a premise-free parallel diamond, full Church–Rosser theorem, subject reduction, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Project the carrier of Definition 4238.1. The displayed packet contains the premise, critical-pair, candidate-join, residual-frontier, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Apply the obligations theorem and the typed boundary theorem, Definition 4238.2 and Definition 4238.16. Together they force a typed-window consumer to expose P, K, J, R, S, C, B, O before any conditional handoff is read.

Use premise retention and discharge obstruction, Definition 4238.13 and Definition 4238.18. The residual-substitution premise row P is still a premise, and the obstruction row O still blocks discharge unless a separate compatibility certificate is supplied.

Finally route the exported handoff through the structural rows H, T, G, N. Candidate-normalization and confluence-audit consumers therefore receive exactly the retained typed-window spine, not a completed parallel-diamond, Church–Rosser, subject-reduction, evaluator-equality, quotient-equality, or kernel-consistency endpoint. □

Theorem 3258.18 (MetaCIC parallel-diamond frontier discharge obstruction)**.** *Any accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet consumed as a local-diamond handoff still exposes the residual-substitution premise row P and obstruction row O. Unless a separate* MetaCICResidualSubstitutionCompatib *certificate is supplied through chapter 4370, no consumer may read that handoff as a premise-free parallel diamond, full closed confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the carrier of Definition 4238.1. The rows P and O are displayed coordinates of the accepted packet, not annotations around the handoff.

Apply premise retention, Definition 4238.13. It keeps P visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows whenever the local-diamond handoff is consumed.

Now use the sibling route theorem Definition 4238.21. The exported contribution is exactly retained premise and obstruction data for confluence-facing siblings, so the handoff cannot be consumed as a premise-free diamond.

The only admissible discharge surface is the residual-substitution compatibility chapter named in the statement. Without that separate certificate, the obstruction row O continues to block full closed confluence, subject reduction, evaluator equality, quotient term equality, and kernel consistency. □

Theorem 3258.19 (MetaCIC parallel-diamond frontier compatibility discharge surface)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, only a separate* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *row may turn the retained* P, O *premise-obstruction surface into a discharged local diamond. Without that compatibility row, every confluence consumer retains the full conditional spine*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and may read the packet only as a conditional parallel-diamond frontier, not as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the discharge obstruction theorem, Definition 4238.18. It identifies P as the retained residual-substitution premise and O as the obstruction row that remains visible unless the separate compatibility chapter supplies a certificate.

Next use premise retention, Definition 4238.13. The premise row is not confined to the statement of the theorem; it is transported through the critical-pair, residual-frontier, closed-checker, and closed-normal rows of the accepted carrier.

Finally apply the confluence handoff, Definition 4238.22. That handoff exports the complete route $P, K, J, R, S, C, B, O, H, T, G, N$ to confluence-facing consumers. Since no row in this packet is the separate $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ certificate, the handoff remains conditional and the refused endpoints stay outside the frontier. □

Theorem 3258.20 (MetaCIC parallel-diamond closed-checker retention)**.** *Any accepted* MetaCICParallelDiamondFr *packet whose local-diamond handoff retains the residual-substitution premise row* P *must also retain the closed-checker seal row* S *and the closed-normal fragment row* C *before that handoff can be read. Thus the conditional handoff boundary is the joint row surface* P, S, C *together with the displayed critical-pair, residual-frontier, bounded-check, obstruction, transport, replay, provenance, and local naming rows. It is not a proof of residual substitution, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Apply the obligations theorem Definition 4238.2. It lists P, S, C as confluence-facing rows of the same accepted carrier, so the checker seal and closed-normal fragment are not optional metadata around the premise row.

Next apply the handoff and route theorems, Definition 4238.3 and Definition 4238.9. Both require the closed-checker seal and closed-normal fragment to be exposed before any conditional parallel-diamond handoff is consumed.

Finally use premise retention, Definition 4238.13. Since the retained premise row is read through K, R, S, C and the obstruction row remains visible, the closed-checker seal and closed-normal fragment remain part of the boundary rather than consequences of the handoff. The refused endpoints therefore stay outside the packet. □

Theorem 3258.21 (MetaCIC parallel-diamond frontier sibling route)**.** *Every accepted* MetaCICParallelDiamondFro *packet that exports its premise-retention handoff feeds two confluence-facing sibling routes: the*

candidate-normalization confluence handoff of chapter 5547 and the audit-packet boundary of chapter 735. The exported contribution is exactly the residual-substitution premise row P, the local critical-pair row K, the residual-frontier row R, and the obstruction row O retained by Definition 4238.13; it does not supply a premise-free diamond, a completed residual-substitution theorem, full closed confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Apply Definition 4238.13. Any consumed local-diamond handoff keeps the residual-substitution premise row visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows.

The candidate-normalization confluence handoff can therefore read this packet only as retained frontier data: P names the missing residual-substitution input, K and R name the local peak and residual window, and O keeps the obstruction endpoints visible before candidate-normalization routes are consumed.

The confluence audit packet reads the same rows as an audit boundary. It can cite the retained premise and obstruction rows when it records a conditional local-diamond route, but it cannot erase them into a completed confluence theorem.

Thus the sibling lattice is substantive rather than bibliographic: the frontier supplies exactly the residual-substitution premise retention and local-diamond obstruction rows consumed by those confluence-facing packets, while the refused endpoints remain outside the carrier. □

Theorem 3258.22 (MetaCIC parallel-diamond confluence handoff)**.** *An accepted* MetaCICParallelDiamondFrontier *packet whose local-diamond, premise-retention, and residual-substitution rows have been displayed exports exactly the conditional confluence handoff consumed by chapter 5547 and bounded by chapter 5126. In carrier terms, the handoff is the retained route through $P, K, J, R, S, C, B, O, H, T, G, N$: P remains the residual-substitution premise, K, J, R expose the local-diamond and residual windows, S, C, B keep the closed checker and bounded fragment visible, and O records the still-open endpoints. The handoff does not claim global confluence, full strong normalization, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the frontier carrier. The rows P, K, J, R, S, C, B, O are the confluence-facing data, while H, T, G, N transport, replay, source, and name the same packet.

Apply Definition 4238.13. The residual-substitution premise row P is retained through the local critical-pair and residual-frontier rows, so the exported route cannot be read as a premise-free diamond.

Next use Definition 4238.18. It keeps the obstruction row O visible unless a separate residual-substitution compatibility certificate is supplied, and therefore preserves the boundary checked by the parallel-diamond residual packet.

Finally apply the sibling route theorem. The candidate-normalization handoff receives only the closed checker boundary, local-diamond window, residual ledger, retained premise, and obstruction data displayed by this packet. Since no carrier row contains global confluence, full strong normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency, none of those endpoints is exported by the handoff. □

Theorem 3258.23 (MetaCIC parallel-diamond premise preservation)**.** *Any confluence consumer that reads the conditional parallel-diamond handoff of Definition 4238.22 must retain the residual-substitution premise row P through the full packet spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, the handoff cannot erase P into a completed residual-substitution theorem, premise-free parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Start with Definition 4238.2. It makes the residual-substitution premise row P part of the local $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ surface together with $K, J, R, S, C, B, O, H, T, G, N$.

Next apply premise retention, Definition 4238.13. The retained premise is still visible when the local-diamond handoff is consumed, and the discharge-obstruction theorem Definition 4238.18 keeps O displayed unless a separate compatibility certificate is supplied.

Finally use the confluence handoff theorem itself. Its exported route is exactly the conditional path through $P, K, J, R, S, C, B, O, H, T, G, N$, so any consumer that drops P or reads the route as one of the refused endpoints has left the packet described by the carrier. □

Theorem 3258.24 (MetaCIC parallel-diamond residual budget)**.** *Any exported* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond budget that feeds candidate-normalization or audit-packet consumers must consume the retained residual-substitution premise row P, critical-pair row K, residual-frontier row R, closed-checker row S, closed-normal row C, obstruction row O, and structural rows H, T, G, N before the consumer receives the handoff. The budget cannot omit P or O, and it cannot be read as a premise-free parallel diamond, residual-substitution theorem, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Start from the carrier obligations Definition 4238.2. They keep the residual-substitution premise, critical-pair, residual-frontier, closed-checker, closed-normal, obstruction, transport, replay, provenance, and naming rows inside the accepted packet.

Use the route and premise-retention theorems, Definition 4238.9 and Definition 4238.13. They require the premise row P to remain visible when a local-diamond handoff is consumed.

Next apply discharge obstruction and the sibling route, Definition 4238.18 and Definition 4238.21. The candidate-normalization and audit-packet consumers receive only the retained premise, critical-pair, residual-frontier, obstruction, and structural rows named by the frontier.

Finally use premise preservation Definition 4238.23. Since its full spine includes $P, K, J, R, S, C, B, O, H, T, G, N$ any exported budget that forgets P or O has left the displayed packet and cannot be consumed as a BEDC parallel-diamond frontier. □

Theorem 3258.25 (MetaCIC parallel-diamond frontier obstruction monotonicity)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. Any refinement of F that preserves the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ may strengthen the finite evidence attached to those rows, but it cannot weaken the obstruction row O or discharge the residual-substitution premise row P unless it also supplies a separate residual-substitution compatibility certificate. Consequently the refined packet remains a conditional frontier and cannot be consumed as full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. The premise row P and obstruction row O are carrier coordinates of the accepted packet, not side assumptions that a refinement may silently erase.

Apply premise retention and discharge obstruction, Definition 4238.13 and Definition 4238.18. They force the local-diamond handoff to keep P visible and keep O active unless a separate compatibility certificate supplies the missing residual-substitution discharge.

Finally use the residual budget theorem Definition 4238.24. A refinement that preserves $P, K, J, R, S, C, B, O, H, T, G, N$ preserves the same budget boundary, so it cannot convert retained obstruction data into a premise-free endpoint. The stronger claims listed in the statement therefore remain outside the packet. □

Theorem 3258.26 (MetaCIC parallel-diamond residual confluence)**.** *Every confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must factor its exported result through the full residual spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The exported result is conditional residual confluence with the residual-substitution premise retained; it is not a premise-free parallel diamond, subject reduction theorem, evaluator equality, quotient term equality, closed Church–Rosser theorem, or kernel-consistency certificate.

Proof. Start with the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the visible rows of any accepted frontier packet, with P retained as a residual-substitution premise.

Use the route theorem and premise retention, Definition 4238.9 and Definition 4238.13. Together they force confluence consumers to expose the premise, peak, join, residual, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local name rows before the handoff is consumed.

Apply discharge obstruction and the confluence handoff theorem, Definition 4238.18 and Definition 4238.22. These results keep the obstruction row visible and identify the exported route as the retained conditional handoff consumed by confluence-facing siblings.

Finally use the residual budget theorem Definition 4238.24. Since the budget cannot omit P or O, any exported confluence result remains conditional residual confluence over the displayed spine rather than one of the refused premise-free, typing, equality, quotient, Church–Rosser, or kernel endpoints. □

Theorem 3258.27 (MetaCIC residual-join readback)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*, every residual-join readback consumed by bounded conversion must display the closed substitution boundary rows, candidate-mediated strong-normalization rows, residual budget rows, typed-example boundary rows, and the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before reading the handoff. In carrier terms, the readback must expose $P, K, J, R, S, C, B, O, H, T, G, N$ *together with the typed-window and L10 source rows, and it cannot be consumed as full confluence, global beta decidability, closed subject reduction, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with residual confluence, Definition 4238.26. It forces a confluence consumer to keep the residual-substitution premise, local critical pair, candidate join, residual frontier, checker seal, closed-normal fragment, bounded-check row, obstruction row, and structural rows visible through the full spine $P, K, J, R, S, C, B, O, H, T, G, N$.

Next use the typed-window handoff of Definition 4238.17. A bounded-conversion consumer may read the frontier only after the typed window has retained the same residual spine for candidate-normalization and confluence-audit consumers.

Premise preservation, Definition 4238.23, keeps the residual-substitution row P from disappearing into a completed diamond. The local-diamond and critical-pair routes of Definition 4238.10

and Definition 4238.9 supply the candidate join and residual budget rows without discharging the obstruction row O.

The typed-example boundary and the four L10 source faces are then read as source rows, not as semantic quotients. They expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the bounded-conversion handoff is consumed, so selected limits and ambient real completeness are unavailable.

Combining these routes gives a finite residual-join readback frontier. Since every step retains the premise and obstruction rows, the result is not full confluence, global beta decidability, closed subject reduction, quotient term equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3258.28 (MetaCIC critical-overlap residual budget)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*, every critical overlap between two local reductions is budgeted by the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The overlap budget must expose the residual-substitution premise row P*, the critical-pair row* K*, the candidate-join row* J*, the residual-frontier row* R*, the closed checker seal* S*, the closed-normal fragment* C*, the bounded-check row* B*, and the obstruction row* O *before any local join is read. It does not supply source-equivalence transport, abstract carrier equivalence, premise-free parallel diamond, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. It makes the premise, critical-pair, candidate-join, residual-frontier, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows the finite carrier of an accepted frontier packet.

Use the residual-substitution scope theorem Definition 4238.12. A local join may be read only after the residual-substitution boundary, critical-pair boundary, and closed-normal fragment have remained visible.

Now apply premise preservation and the residual budget theorem, Definition 4238.23 and Definition 4238.24. These keep P and O inside the exported budget and prevent a critical overlap from being consumed as a premise-free diamond.

Finally use residual confluence, Definition 4238.26. Since the confluence consumer factors through the same displayed spine, the critical-overlap budget is finite and row-local rather than a transport across source equivalence or abstract carrier equivalence. □

Theorem 3258.29 (MetaCIC candidate-mediated residual route)**.** *Let two local reductions be recorded by the critical-pair row* K *of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*. If the candidate-join row* J *is accepted through the displayed residual-frontier row* R*, then both local reductions route to the shared residual packet*

$$J, R, S, C, B, O, H, T, G, N$$

under the retained premise P*. The route is candidate-mediated: it identifies the common residual packet available to the frontier consumer, but it does not discharge residual-substitution compatibility, source-equivalence transport, subject reduction, full closed Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the carrier in Definition 4238.1. The critical-pair row K records the two local reduction observations, and the candidate-join row J records the proposed common reduct together with branch-local route requests.

Apply the obligations theorem Definition 4238.2. It keeps J and R as displayed rows of the same packet, so the candidate route is not an external equivalence between carriers.

Use the residual-substitution scope theorem and premise preservation, Definition 4238.12 and Definition 4238.23. The two local reductions may reach the candidate only while P remains visible and while the residual-frontier row controls the branch windows.

Finally apply residual confluence and residual-join readback, Definition 4238.26 and Definition 4238.27. They identify the consumer-facing route as the shared residual packet through $J, R, S, C, B, O, H, T, G, N$, with the premise and obstruction still retained. □

Definition 3258.30 (MetaCIC parallel-diamond frontier finite budget)**.** *The finite budget for a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *read is the row surface*

$$\mathcal{B}_{\text{frontier}} = (W, K, P, J, R, S, C, B, O, H, T, G, N).$$

Here W is the typed-window row, K is the critical-pair row, P is the residual-substitution boundary, J is the local-join row, R is the residual-frontier row, S is the closed-checker seal, C is the closed-normal fragment, B is the bounded-checker row, O is the obstruction row, H is componentwise hsame *transport, T is displayed* Cont *replay, G is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row. The budget has no coordinate for premise-free confluence, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Theorem 3258.31 (MetaCIC typed-window local-join scope)**.** *Under a typed MetaCIC window, an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet permits only local joins whose two branches pass through the retained residual-substitution premise P, critical-pair row K, candidate-join row J, residual-frontier row R, closed checker seal S, closed-normal fragment C, and bounded-check row B. The local join scope is therefore the typed-window row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with O still visible. It is not a source-equivalence transport theorem, an abstract carrier-equivalence theorem, a premise-free parallel diamond, subject reduction, global beta decidability, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the typed boundary and typed-window handoff, Definition 4238.16 and Definition 4238.17. They force a typed-window consumer to expose the same residual spine before candidate-normalization or confluence-audit consumers can read a handoff.

Use the critical-overlap budget of Definition 4238.28. It keeps every local overlap inside $P, K, J, R, S, C, B, O, H, T, G, N$ and blocks readings that omit the premise or obstruction rows.

Next apply the candidate-mediated route Definition 4238.29. The two branches can meet only through the displayed candidate row and residual-frontier row, not through an ambient source-equivalence or carrier-equivalence transport.

Finally use residual-join readback, Definition 4238.27. The readback keeps bounded conversion, typed windows, and the L10 source faces visible before the handoff, so the permitted local join remains a scoped MetaCIC window claim rather than any stronger confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 3258.32 (MetaCIC parallel-diamond frontier local join budget)**.** *Any local join read accepted by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet consumes the finite budget of Definition 4238.30*

before it can reach a normalization-facing consumer. In particular, the typed-window row, critical-pair row, residual-substitution boundary, local-join row, residual-frontier row, closed-checker seal, bounded-checker row, obstruction row, transport, replay, provenance, and local naming remain visible at the handoff.

Proof. Start with the typed-window local-join scope of Definition 4238.31. It forces any accepted local join to expose the residual-substitution premise, critical pair, candidate join, residual-frontier row, checker rows, and obstruction row.

Apply the finite budget definition Definition 4238.30. The typed-window row and structural rows H, T, G, N are part of the same budget surface, so the local join cannot be consumed as a free-standing confluence fact.

Use the residual budget theorem Definition 4238.24. It keeps the obstruction and residual-substitution rows visible before normalization consumers read the packet. Therefore the handoff remains a finite typed-window claim and does not become premise-free normalization or global confluence. □

Theorem 3258.33 (MetaCIC parallel-diamond frontier residual boundary)**.** *Residual-substitution compatibility remains a visible boundary in every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *local-join read. The row P of Definition 4238.30 is consumed before J, R, S, C, B can be used, and it is not erased into an unconditional confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, global normalization statement, or kernel-consistency certificate.*

Proof. Project the budget surface of Definition 4238.30. The residual-substitution boundary P is an explicit coordinate, not a proof obligation hidden behind the local-join row.

Apply premise preservation and the residual-substitution scope theorem, Definition 4238.23 and Definition 4238.12. These keep P visible whenever the critical-pair, residual-frontier, checker, and bounded-check rows are read.

Finally use the local join budget theorem Definition 4238.32. Since every normalization-facing read consumes the full finite budget first, no consumer can erase the residual boundary and present the packet as an unconditional confluence or kernel result. □

Theorem 3258.34 (MetaCIC parallel-diamond frontier obligation basis)**.** *Every accepted* MetaCICParallelDiamondF *packet presents a finite obligation basis for the typed-window frontier. The basis is the row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P is the residual-substitution boundary, K is the critical-pair envelope, J is the local-join row, R is the residual-frontier row, S is the closed checker seal, C is the closed-normal fragment, B is the bounded-check row, O is the obstruction row, H is hsame *transport, T is* Cont *replay, G is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row. Any confluence consumer must read this basis before it may cite the typed-window local join, and the basis does not discharge residual substitution, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with the residual budget theorem Definition 4238.24. It keeps the retained premise, critical-pair row, residual-frontier row, closed-checker row, closed-normal row, obstruction row, and structural rows visible before any exported budget is consumed.

Apply residual confluence and residual-join readback, Definition 4238.26 and Definition 4238.27. These route confluence-facing reads through the same residual spine and require bounded-conversion readback to keep the typed-window and L10 source rows visible.

Use the critical-overlap budget and candidate-mediated residual route, Definition 4238.28 and Definition 4238.29. The local peak may reach a candidate only through the displayed K, J, R rows, with P and O retained.

Finally apply the typed-window local-join scope theorem Definition 4238.31. It identifies the permitted local join as exactly the typed-window row surface, so the listed rows form the finite obligation basis and no stronger confluence endpoint is exported. □

Theorem 3258.35 (MetaCIC parallel-diamond frontier seed status)**.** *Every conditional parallel-diamond handoff consumed through an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must first expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, bounded-check row B, obstruction row O, componentwise* hsame *transport row H, displayed* Cont *replay row T,* Pkg *provenance row G, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row N. The consumed handoff is therefore a seed-status frontier: it records which residual premises and finite checks remain visible before use, and it is not a parameter-transport equivalence, source-equivalence theorem, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Begin with the obligations theorem Definition 4238.2. It makes the residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and naming rows part of the finite carrier surface.

Apply the obligation-basis theorem Definition 4238.34. It packages those rows as the typed-window basis that a confluence consumer must read before citing a local join.

The residual-frontier and obstruction rows stay visible throughout the handoff. Consequently the theorem records seed status for the remaining residual-discharge work rather than converting the packet into a parameter-transport equivalence or any stronger confluence endpoint. □

Theorem 3258.36 (MetaCIC parallel-diamond residual obstruction ledger)**.** *Every candidate-mediated local join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise row P, residual-frontier row R, and obstruction row O visible alongside*

$$K, J, S, C, B, H, T, G, N$$

before any confluence-facing packet consumes the read. The retained ledger is therefore

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It is not a premise-free parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Begin with the obligation basis Definition 4238.34. It fixes the consumer-visible packet as $P, K, J, R, S, C, B, O, H, T, G, N$ and makes the obstruction row an explicit coordinate of the finite basis.

Apply the candidate-mediated residual route Definition 4238.29. A local peak may reach the candidate row J only through the displayed critical-pair and residual-frontier route, while P remains the retained residual-substitution premise.

Finally use the local-join scope theorem Definition 4238.31. It allows confluence-facing consumers to read only the typed-window surface with O still visible, so the obstruction ledger cannot be erased before the handoff is consumed.

Thus every candidate-mediated join read carries P, R, O together with $K, J, S, C, B, H, T, G, N$. Since neither the premise nor the obstruction is discharged, the statement exports only a residual

obstruction ledger and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 3258.37 (MetaCIC residual pasting frontier)**.** *Suppose an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet has two typed-window residual reads whose critical-pair observations share the same* $P, K, R, S, C, B, O, H, T,$ *rows and whose candidate row is* J*. Then the two reads paste only through the displayed residual packet*

$$J, R, S, C, B, O, H, T, G, N$$

under the retained premise P*. The pasted read is a finite* NameCert *route with* hsame *transport,* Cont *replay, and* Pkg *provenance; it is not a source-equivalence transport theorem, abstract carrier equivalence, premise-free parallel diamond, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Begin with Definition 4238.34. It fixes the consumer-visible basis as $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the obstruction row visible.

Use candidate-mediated residual routing, Definition 4238.29. The two local reductions may meet only at the displayed candidate row J and residual-frontier row R, while the residual-substitution premise P remains retained.

Apply residual confluence and residual-join readback. These theorems keep the closed checker, closed-normal fragment, bounded-check row, obstruction, transport, replay, provenance, and local naming attached to the same packet before any bounded-conversion consumer can read the handoff.

The pasted route is therefore the shared residual packet $J, R, S, C, B, O, H, T, G, N$ under P. Since the construction never introduces a source-equivalence carrier or removes O, the conclusion is a finite frontier pasting claim and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 3258.38 (MetaCIC substitution nonescape frontier)**.** *In every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, substitution data remains confined to the residual-substitution boundary row* P *and the residual-frontier row* R *throughout the typed-window local join. No* hsame *transport row,* Cont *replay row,* Pkg *provenance row, or local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row may reinterpret the handoff as a discharged residual-substitution theorem, premise-free parallel diamond, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start from the obligation basis Definition 4238.34. It makes P and R explicit coordinates of the finite packet rather than implicit assumptions around the local join.

Apply the residual-substitution scope theorem and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the substitution premise visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows.

Use residual pasting, Definition 4238.37. Even when two typed-window residual reads are pasted through the candidate row, the pasted route still runs under P and through R.

Finally read the structural rows H, T, G, N. They transport, replay, source, and name the displayed packet, but they do not contain a carrier slot for discharging substitution or erasing the obstruction row. Thus substitution cannot escape the displayed residual boundary into any stronger confluence endpoint. □

Theorem 3258.39 (MetaCIC parallel-diamond residual-substitution route)**.** *Every candidate-mediated local join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T,$ *must expose the residual-substitution premise row* P*, critical-pair row* K*, candidate-join row* J*,*

residual-frontier row R, closed-checker seal row S, closed-normal fragment row C, bounded-check row B, obstruction row O, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming before any parallel-diamond frontier handoff is consumed. The route is conditional on the displayed premise and cannot be read as residual-substitution compatibility, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the carrier of Definition 4238.1. The rows $P, K, J, R, S, C, B, O, H, T, G, N$ are the full consumer-visible packet, with P retaining the residual-substitution premise and O retaining the refused endpoints.

Apply the candidate-mediated residual route of Definition 4238.29. It allows the two local reductions named by K to meet only through the displayed candidate row J and residual-frontier row R, under the retained premise P.

Next use substitution nonescape, Definition 4238.38. The closed checker row, closed-normal row, bounded-check row, transport, replay, provenance, and local naming cannot discharge substitution or remove the obstruction row.

Thus the route exposes exactly the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, and structural rows stated above. Since neither P nor O disappears, the handoff remains a conditional frontier and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 3258.40 (MetaCIC parallel-diamond normalization-frontier handoff)**.** *Any normalization consumer that reads the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may use only the conditional frontier exposed by Definition 4238.39. It may inspect the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming. It may not consume the packet as full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, or a completed residual-substitution theorem.*

Proof. Begin with the residual-substitution route theorem. It fixes the normalization-facing read to the displayed finite spine $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the obstruction row visible.

Use the typed-window local-join scope of Definition 4238.31. A normalization consumer may read local joins only under that typed-window surface, so it cannot erase the residual-frontier or closed-checker conditions.

Apply residual pasting and substitution nonescape, Definition 4238.37 and Definition 4238.38. These theorems keep pasted residual reads under P and prevent structural rows from retyping the handoff as a discharged substitution result.

Consequently the normalization consumer receives only the conditional frontier. The listed stronger endpoints require separate certificates and are not carried by the accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ packet. □

Theorem 3258.41 (MetaCIC parallel-diamond candidate-mediated strong-normalization socket)**.** *For every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *carrier $F = (P, K, J, R, S, C, B, O, H, T, G, N)$, suppose the closed checker row S accepts both residual branches named by the local critical-pair row K, the candidate-normalization scope row J retains the residual-substitution premise row P, and the local join ledger read through R, C, B, H, T, G, N is available. Then the packet exposes a candidate-mediated strong-normalization socket for that local pair: its readback is the conditional route $P, K, J, R, S, C, B, O, H, T, G, N$ consumed by Definition 4238.40. The socket does not assert*

global strong normalization, full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start from the critical-pair envelope of Definition 4238.4. It fixes the two residual branches inside the displayed local pair row K and keeps them tied to the closed checker row S.

Use the residual-substitution dependency route of Definition 4238.12. The premise row P remains an explicit coordinate of the candidate-normalization scope instead of becoming a discharged theorem.

Next apply the closed-normal join socket and local join ledger pullback, Definition 4238.14 and Definition 4238.31. These results route the accepted branches through J, R, C, B while preserving H, T, G, N as transport, replay, provenance, and naming rows.

Finally hand the same finite readback to Definition 4238.40. Since P and O remain visible, the socket is only the candidate-mediated local normalization route promised in the statement, not any global normalization or confluence endpoint. □

Theorem 3258.42 (MetaCIC parallel-diamond frontier typed-window candidate SN route)**.** *Every strong-normalization-facing read of the candidate-mediated socket in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *must pass through the typed-window handoff, residual-substitution scope, premise preservation, local-join scope, and the candidate-mediated strong-normalization socket before it reaches the normalization-frontier handoff. The resulting route is the finite readback*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with the typed window still visible. It does not assert global strong normalization, full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with the typed-window handoff of Definition 4238.17. It forces every candidate-normalization consumer to expose the same residual spine before any frontier handoff is read.

Next apply residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. These keep P attached to the critical-pair, residual-frontier, closed-checker, and closed-normal rows rather than turning it into a discharged substitution theorem.

Use the typed-window local-join scope of Definition 4238.31. A candidate join can be consumed only through the displayed K, J, R, S, C, B rows, with the obstruction row O still retained.

Finally apply the candidate-mediated strong-normalization socket and normalization-frontier handoff, Definition 4238.41 and Definition 4238.40. They pass the same finite readback forward as a conditional socket, so the route cannot be retyped as any of the refused normalization, confluence, typing, equality, quotient, or kernel endpoints. □

Theorem 3258.43 (MetaCIC parallel-diamond closed-substitution window scope)**.** *Every candidate-mediated diamond read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G,$ *that exposes the rows* $P, K, J, R, S, C, B, O, H, T, G, N$ *must keep the closed-substitution window visible before the closed checker seal and normalization-confluence handoff are consumed. The read therefore remains a finite frontier route through the displayed residual-substitution premise, typed candidate window, closed checker, local join ledger, obstruction row, transport, replay, provenance, and local naming rows. It cannot be reread as premise-free subject reduction or as closed Church–Rosser.*

Proof. Project the carrier and begin with the premise-retention route of Definition 4238.23. The residual-substitution premise row P remains visible whenever the candidate rows K, J, R, S, C, B, O are exposed.

Apply the typed-window candidate SN route of Definition 4238.42. It sends every strong-normalization-facing read through the typed window and candidate-mediated socket before the normalization-frontier handoff is available.

Finally use the critical-path retained-premise handoff of Definition 4238.85. The structural rows H, T, G, N transport, replay, source, and name the same retained-premise packet, so the closed checker seal and normalization-confluence handoff cannot erase the closed-substitution window. Hence the route is not a premise-free subject-reduction theorem and not closed Church–Rosser. □

Theorem 3258.44 (MetaCIC parallel-diamond finite candidate SN extension)**.** *Let $F = (P, K, J, R, S, C, B, O, H,$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier. If a finite local peak is routed through the displayed candidate row J, residual row R, typed window S, and boundary rows C, B, O, then the only strong-normalization-facing read is the candidate-mediated socket already named by Definition 4238.41. The route cannot be retyped as global strong normalization, full confluence, quotient equality of terms, or a kernel endpoint.*

Proof. Apply the candidate-mediated residual route Definition 4238.29. It sends the finite local peak through the displayed candidate row J and residual row R while retaining the residual-substitution premise row P.

Use the local-join scope Definition 4238.31. The local join is visible only through the typed-window and boundary rows S, C, B, O, so no ambient confluence carrier is introduced.

Next apply the normalization-frontier handoff Definition 4238.40. It restricts normalization consumers to the conditional frontier packet and keeps the refused endpoints outside the carrier.

Finally use the candidate-mediated strong-normalization socket and typed-window candidate SN route, Definition 4238.41 and Definition 4238.42. These results identify the only SN-facing read as the finite candidate-mediated socket through $P, K, J, R, S, C, B, O, H, T, G, N$, so it cannot be reread as global strong normalization, full confluence, quotient equality, or a kernel endpoint. □

Theorem 3258.45 (MetaCIC parallel-diamond frontier seed closure status)**.** *The seed closure surface of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet is exactly the finite conjunction of residual-substitution premise retention, critical-pair envelope exposure, closed-checker retention, closed-normal join socket, candidate-mediated strong-normalization socket, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The seed surface does not discharge residual-substitution compatibility, premise-free parallel diamond, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the row P visible through every local-diamond read.

Use the critical-pair envelope and closed-checker retention, Definition 4238.4 and Definition 4238.20. These identify the rows K, S, C, B, O as part of the finite frontier instead of consequences of a discharged confluence theorem.

Apply the local-join and normalization handoff theorems, Definition 4238.31 and Definition 4238.40. Together with the candidate-mediated strong-normalization socket Definition 4238.41, they show that the only normalization-facing read is the conditional route through P, K, J, R, S, C, B, O.

Finally read the structural rows H, T, G, N. They transport, replay, source, and name the same finite packet, so the seed closure surface is exactly the listed conjunction and none of the refused endpoints is exported. □

Theorem 3258.46 (MetaCIC parallel-diamond residual-join induction)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier. Finite iteration over a displayed list of*

residual pairs preserves the local-diamond route through the critical-pair row K, residual-substitution premise row P, candidate-normalization row J, closed-checker row S, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N. The iterated route remains candidate-mediated and exports no premise-free parallel diamond, subject reduction, full closed Church–Rosser theorem, evaluator equality, quotient equality of terms, or kernel consistency endpoint.

Proof. Induct on the finite residual-pair list. The base case is Definition 4238.45: seed readiness already exposes residual-substitution premise retention, critical-pair rows, candidate joins, closed checker rows, bounded checks, obstruction retention, and the structural rows.

For the step, append one displayed residual pair. The residual-substitution dependency route of Definition 4238.12 keeps P visible, while candidate-normalization scope and finite residual witness consumption keep the new pair inside K, J, R, S, C, B, O before any normalization-facing handoff is read.

Finally apply the critical-path consumer route of Definition 4238.85. It passes the extended finite readback through H, T, G, N without changing the refused endpoints, so the induction preserves the local-diamond route claimed above. □

Theorem 3258.47 (MetaCIC parallel-diamond public export)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier $F = (P, K, J, R, S, C, B, O, H, T, G, N)$, the public confluence-facing export is exactly the route through residual-substitution premise P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N. The export may feed MetaCIC critical-path and confluence-audit consumers, but it does not prove full Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency.*

Proof. Begin with the obligation surface of Definition 4238.2. It admits the carrier only through the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Use critical-pair exhaustion, obstruction retention, and local-join scope: Definition 4238.4, Definition 4238.5, and Definition 4238.31. These keep the critical pair, candidate join, residual frontier, bounded check, and obstruction rows visible before any confluence-facing read.

Next apply the normalization handoff and candidate-mediated socket, Definition 4238.40 and Definition 4238.41. They route normalization-facing consumers through the same finite packet without exporting global normalization or confluence.

Finally invoke residual-join induction, Definition 4238.46. The structural rows H, T, G, N transport, replay, source, and name the same packet, so the public export is the listed route and none of the refused endpoints follows. □

The Lean follow-up supplies $\mathsf{Nontrivial}$ and $\mathsf{FieldFaithful}$ TasteGate instances in `lean4/BEDC/Derived/MetaCICP` by exhibiting distinct accepted and refused frontier packets and by matching on the rows $P, K, J, R, S, C, B, O, H, T,$

Theorem 3258.48 (MetaCIC parallel-diamond scoped closure status)**.** *The seed-readiness surface, retained-premise handoff, obstruction row, residual-substitution premise, critical-path consumer route, and normalization-confluence handoff of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet together scope the chapter to finite candidate-mediated local-diamond evidence. The scoped surface is exactly the displayed route through $P, K, J, R, S, C, B, O, H, T, G, N$ and does not assert subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Read the seed surface through Definition 4238.83 and Definition 4238.45. These results expose the carrier rows and keep the finite local-diamond evidence inside the displayed packet.

Apply the retained-premise handoff and residual-substitution scope, Definition 4238.85 and Definition 4238.12. They keep the premise row P visible for every critical-path read instead of discharging it as a substitution theorem.

Finally use the obstruction, public export, and normalization-frontier handoff rows of Definition 4238.5, Definition 4238.47, and Definition 4238.40. The resulting scope is finite, candidate-mediated, and local to $P, K, J, R, S, C, B, O, H, T, G, N$, so none of the refused global endpoints is exported. □

Theorem 3258.49 (MetaCIC parallel diamond critical-pair envelope)**.** *Every accepted* MetaCICParallelDiamondFro *packet exposes a critical-pair envelope only after reading a residual substitution row, a local confluence overlap row, a closed-term substitution row, and a binder-budget row. In carrier terms, the envelope is the finite BEDC route through*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with the residual-substitution and confluence audit-map rows still visible. The conclusion is a concrete frontier envelope for bounded MetaCIC confluence audit; it is not a general strong-normalization theorem, unrestricted confluence theorem, unrestricted beta-reduction theorem, or host equality principle.

Proof. Project the accepted frontier carrier. The residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N are the only rows exposed by the packet.

Read the residual-substitution route first. The row P supplies the retained premise, while the residual-frontier row R records branch-local substitution windows and binder exposure; neither row discharges compatibility.

Next read the confluence audit-map route. The critical-pair and candidate-join rows K, J supply local overlap data, while S, C, B keep the closed-term substitution and bounded checker information attached to the same finite packet.

The obstruction and structural rows close the envelope. Since O, H, T, G, N preserve the retained premise, bounded replay, provenance, and local name, the packet cannot escape to unrestricted beta reduction, host equality, strong normalization, or a general confluence theorem. □

Theorem 3258.50 (MetaCIC parallel diamond frontier critical-pair envelope)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet exposes critical-pair requests only through the finite frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The candidate term rows K, J, residual-substitution rows P, R, finite observation confluence rows S, C, B, normalization-frontier row O, componentwise hsame *transport H,* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N remain displayed before any consumer can read a parallel-diamond request. The route does not assert unrestricted confluence, host reduction equality, unbounded critical-pair search, or a global normalization theorem.*

Proof. Project the carrier and the existing critical-pair envelope. The projection exposes $P, K, J, R, S, C, B, O, H, T$ as the only rows available to a frontier request.

Read the candidate and residual rows first. The candidate term rows K, J store the finite overlap and join candidates, while P, R keep substitution and residual windows attached to the same bounded branch data.

Next read the finite observation confluence rows S, C, B and the normalization-frontier row O. These rows certify only the displayed closed-term checks, bounded normal fragments, checker budget, and obstruction boundary.

Finally replay through H, T, G, N. Since transport, replay, provenance, and local naming preserve the same finite packet, the accepted critical-pair request cannot be replaced by unrestricted confluence, host reduction equality, unbounded search, or global normalization. □

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCriticalPairEnvelop

Theorem 3258.51 (MetaCIC parallel diamond frontier visible overlap)**.** *Every accepted* MetaCICParallelDiamondFr *packet exposes local overlap only through*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Any consumer that reads the candidate join J must first keep the residual-substitution premise row P, the critical-pair row K, the residual-frontier row R, and the bounded checker rows S, C, B visible. The read is therefore a bounded overlap frontier, not subject reduction, full Church–Rosser, unrestricted confluence, host reduction equality, or kernel consistency.

Proof. Start with the critical-pair envelope theorem Definition 4238.49. It exposes the same finite route $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the residual-substitution and confluence audit-map rows visible.

Apply the frontier envelope theorem Definition 4238.50. It orders critical-pair requests so that K, J are read with P, R, S, C, B, O still attached.

Finally use the route theorem Definition 4238.9. Replay, provenance, transport, and local naming through H, T, G, N preserve the visible premise and bounded checker rows before the candidate join can be consumed. Since the route theorem does not discharge residual substitution or subject reduction, the visible-overlap read remains a frontier certificate rather than a completed confluence result. □

Theorem 3258.52 (MetaCIC parallel-diamond critical-pair join envelope)**.** *Every candidate-mediated strong-normalization or confluence consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first display the critical-pair envelope, residual-substitution dependency route, residual-join locality surface, finite residual witness row, and obstruction row for the same finite carrier*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The consumer may read the candidate join only as this bounded envelope. It does not discharge residual substitution, prove subject reduction, identify evaluator equality, identify quotient term equality, prove global confluence, or assert kernel consistency.

Proof. Project the accepted frontier carrier. The displayed rows are the retained residual-substitution premise P, critical-pair row K, candidate join J, residual frontier R, closed-checker row S, closed-normal fragment C, bounded checker B, obstruction row O, and structural rows H, T, G, N.

Read the envelope rows in order. The critical-pair envelope keeps the bounded overlap data visible, while the residual-substitution dependency route places P before any join handoff.

Now attach the local join and witness rows. Residual-join locality keeps J inside the same frontier packet, and the finite residual witness row forces the typed windows, residual data, bounded checker rows, transport, replay, provenance, and local naming to remain visible.

Finally retain the obstruction row. Since O, H, T, G, N preserve only the bounded carrier-local packet, a candidate-mediated consumer receives an envelope rather than residual-substitution discharge, subject reduction, evaluator equality, quotient term equality, global confluence, or kernel consistency. □

Theorem 3258.53 (MetaCIC parallel-diamond frontier critical-pair route)**.** *Every critical-pair read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier exposes candidate membership, bounded local peak rows, residual envelope rows, retained premise rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *before any join row is consumed. For a carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the read must display the retained premise P, local peak row K, residual envelope R, bounded checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before the candidate-join row J can serve a confluence consumer.

Proof. Project the carrier of Definition 4238.1. The only rows available to a critical-pair consumer are $P, K, J, R, S, C, B, O, H, T, G, N$, with P retaining the premise and K, R, S, C, B, O displaying the local peak, residual, checker, and obstruction envelope.

Apply critical-pair exhaustion and residual-substitution scope, Definition 4238.4 and Definition 4238.12. They force the local peak through K under the retained premise P and keep the residual frontier R visible before the candidate join is read.

Finally use the critical-path retained-premise handoff and seed closure status, Definition 4238.85 and Definition 4238.45. These routes pass the same critical-pair read through H, T, G, N without consuming J as a detached join or erasing the obstruction row. □

Theorem 3258.54 (MetaCIC parallel-diamond frontier residual-join nonescape)**.** *Every residual-join read exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier is a candidate-mediated local join through $P, K, J, R, S, C, B, O, H, T, G, N$. The frontier cannot be consumed as full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, or a discharged residual-substitution theorem.*

Proof. Start from the residual-join induction and public export theorems, Definition 4238.46 and Definition 4238.47. They keep finite residual joins inside the displayed route through the retained premise, critical-pair, candidate, residual, checker, obstruction, transport, replay, provenance, and naming rows.

Use local-join scope and the retained-premise handoff, Definition 4238.31 and Definition 4238.85. A join can be read only through the candidate row J under the same premise P and residual envelope R; it is not a premise-free confluence endpoint.

The refused endpoints remain outside the carrier. Since O is still visible and no row supplies subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, or residual-substitution discharge, the exported join remains candidate-mediated and local. □

Theorem 3258.55 (MetaCIC parallel-diamond critical-path consumer route)**.** *Every critical-path consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, closed-checker row S, closed-normal row C, bounded-check row B, and obstruction row O before any local confluence handoff*

is public. The required consumers are the critical-path packet of chapter 765, the confluence audit packet of chapter 735, and the normalization-confluence bridge of chapter 1190.

Proof. Project the carrier of Definition 4238.1. Its confluence-facing coordinates include P, K, J, S, C, B, O, so none of the three consumer chapters can read the frontier through a smaller public row set.

The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ consumer receives the residual-substitution premise, critical-pair, and obstruction rows as route pressure rather than as a discharged local-confluence theorem. The $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ consumer receives the candidate-join, closed-checker, and bounded-check rows as audit data before it may inspect the local handoff. The MetaCICNormalizationConfluenceBrid consumer receives the closed-normal row and the same obstruction boundary as bridge-facing evidence.

Thus the critical-path, confluence-audit, and normalization-confluence consumers share the full retained frontier. A local confluence handoff is public only after those rows have been displayed, and the route still does not assert residual-substitution compatibility, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.56 (MetaCIC critical-path bridge frontier)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that is consumed by a MetaCIC critical-path reader must keep its residual diamond rows as concrete candidate and discharge data for* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$*, and* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$*. In carrier terms, the rows* $P, K, J, R, S, C, B, O, H, T, G, N$ *may be routed into the critical-path, confluence-audit, and normalization-confluence bridge surfaces only as retained frontier evidence; they do not form a standalone confluence oracle, subject-reduction theorem, evaluator equality row, quotient equality row, or kernel-consistency certificate.*

Proof. Project the frontier carrier through Definition 4238.1. The residual-substitution premise, critical-pair data, candidate join, residual window, closed checker, closed-normal fragment, bounded check, and obstruction rows remain displayed before the transport, replay, provenance, and local name rows are read.

Use the route and handoff theorems of Definition 4238.9 and Definition 4238.22. They make the exported handoff conditional on the retained premise and obstruction rows, so the confluence-audit packet can consume the rows only as bounded local evidence.

Now read the critical-path and normalization-confluence consumers. The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ surface receives the same rows as candidate and discharge pressure, and $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ receives them as bridge-facing boundary data attached to the confluence audit route.

Thus the sibling route is substantive: the frontier contributes concrete residual diamond rows to the critical-path and bridge consumers, while every standalone confluence-oracle, subject-reduction, evaluator-equality, quotient-equality, and kernel-consistency reading remains outside the packet. □

Theorem 3258.57 (MetaCIC parallel-diamond residual-substitution dependency route)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that is consumed as a parallel-diamond frontier must first expose the residual-substitution premise row supplied by chapter 2181. The consumer may route that premise into the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*, but it cannot replace the premise by residual-substitution compatibility, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualSubstitutio`

Proof. Project the frontier carrier of Definition 4238.1. The residual-substitution premise row P is one of the displayed carrier coordinates, not an external theorem attached after the handoff.

Read the residual-substitution boundary chapter named in the statement. It supplies the visible premise boundary that the frontier route consumes before any candidate join or local-diamond handoff is public.

Finally apply the premise-retention and discharge-obstruction surfaces already present in the frontier chapter. They keep P and the obstruction row visible for every consumer, so the dependency route remains a premise boundary rather than any of the refused confluence endpoints. □

Theorem 3258.58 (MetaCIC parallel-diamond residual-premise join determinacy)**.** *Let* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *and* $F' = (P, K, J', R, S, C, B, O, H', T', G', N')$ *be accepted* MetaCICParallelDiam *packets whose residual-substitution premise row* P*, critical-pair row* K*, residual-frontier row* R*, closed-checker row* S*, closed-normal row* C*, bounded-check row* B*, and obstruction row* O *agree on the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *surface. Any common candidate-join handoff exposed by both packets is therefore read only through the same displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *route, with* J *the candidate row visible on that route. The handoff does not discharge residual substitution, prove subject reduction, give full Church–Rosser, identify evaluator equality, quotient terms, or assert kernel consistency.*

Proof. Project the two frontier carriers through Definition 4238.1. The rows named in the statement are carrier coordinates, so a common candidate-join read cannot be detached from the shared P, K, R, S, C, B, O boundary.

Now apply the dependency route of Definition 4238.57. It places the residual-substitution premise P before the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows.

Finally use the premise-retention and route surfaces already recorded in the chapter. They keep the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ visible for every accepted consumer, so the shared handoff is deterministic only at the finite route boundary and remains separate from the refused confluence and kernel endpoints. □

Theorem 3258.59 (MetaCIC parallel-diamond residual-substitution confluence budget)**.** *Every residual-substitution route that contributes to the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond frontier must expose, in order, the retained premise row* P*, residual pair row* R*, local join row* J*, substitution dependency row* S*, and finite budget row* B *before any candidate-mediated strong-normalization handoff may consume it. The budget is a finite row of the frontier carrier, not a global confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency witness.*

Proof. Project the frontier carrier of Definition 4238.1. The retained premise, residual pair, local join, substitution dependency, and budget rows are displayed carrier coordinates, so a consumer cannot read the candidate handoff before these rows are present.

Apply the residual-substitution dependency route of Definition 4238.57. It places the retained premise boundary before the critical-pair and candidate-join rows and keeps residual substitution as a visible dependency of the frontier.

Finally use residual-join locality from the surrounding frontier rows. The local join is certified only inside the finite P, R, J, S, B route, and the candidate-mediated handoff consumes that route as a bounded frontier witness. Since no row stores global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the budget. □

Theorem 3258.60 (MetaCIC residual-substitution discharge criterion)**.** *Let an accepted* MetaCICParallelDiamondF *packet supply the retained residual-substitution premise row and let a separate* MetaCICResidualSubstitutionCompatib *certificate be displayed. Then the local-diamond handoff may be read as a discharge-ready boundary for the candidate-normalization confluence handoff.*

This boundary is still only the finite frontier route. It is not full Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency.

Proof. Start with the discharge obstruction of Definition 4238.18. It identifies the retained residual-substitution premise as the row that remains visible until a separate compatibility certificate is supplied.

Next apply Definition 4238.19. The displayed $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ certificate is exactly the extra row allowed to turn the premise-obstruction surface into a discharged local-diamond boundary.

Finally use the confluence handoff of Definition 4238.22. The handoff remains a candidate-normalization consumer of the finite frontier route, so it exports only discharge-ready local-diamond evidence and none of the refused global endpoints. □

Theorem 3258.61 (MetaCIC observation confluence budget)**.** *Every observation-level confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, and obstruction row O before reading any local join through the candidate row J. The observation budget is the finite frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It may cite the retained-premise route of Definition 4238.12 and the public frontier route of Definition 4238.9, but it does not prove residual-substitution compatibility, subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with the carrier and obligation basis of Definition 4238.1 and Definition 4238.34. They make $P, K, J, R, S, C, B, O, H, T, G, N$ the consumer-visible packet, with P and O ordinary rows rather than external assumptions.

Apply residual-substitution scope and the frontier route, Definition 4238.12 and Definition 4238.9. These force the premise row, critical-pair row, residual-frontier row, closed-checker row, and closed-normal fragment row to remain visible before the conditional handoff is consumed.

Finally use local-join scope Definition 4238.31. A local join can be read only after the candidate row has been placed inside the same typed-window surface with B and O still exposed. Thus the theorem records an observation budget for finite confluence consumers, not a closed confluence proof or a discharge of the residual-substitution obstruction. □

Theorem 3258.62 (MetaCIC candidate-normalization scope)**.** *Every candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the full frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before it reads the candidate row. The consumer may use J only as the displayed candidate-join row of a candidate-mediated residual route, and only while the residual-substitution premise row P and obstruction row O remain visible. Thus the candidate-normalization read is not a premise-free confluence theorem, not a completed residual-substitution theorem, not full normalization, not subject reduction, not evaluator equality, not quotient equality of terms, and not a kernel-consistency certificate.

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCandidateNormalizat

Proof. Begin with the candidate-mediated residual route of Definition 4238.29. It permits a local peak to reach the candidate row J only through the displayed critical-pair row K and residual-frontier row R, with the retained premise row P still attached.

Next apply substitution nonescape, Definition 4238.38. That theorem keeps substitution data confined to P and R through the typed-window local join, so the candidate-normalization consumer cannot turn J into a discharged substitution result.

Use residual pasting, Definition 4238.37. Even after two typed-window residual reads are pasted through J, the pasted read remains the finite residual packet under P and still carries S, C, B, O, H, T, G, N.

Finally apply the normalization-frontier handoff of Definition 4238.40. The handoff exports only the conditional spine $P, K, J, R, S, C, B, O, H, T, G, N$ to normalization consumers, so P and O cannot be hidden when J is consumed. □

Theorem 3258.63 (MetaCIC parallel-diamond frontier obstruction retention under candidate normalization)**.** *Every candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier that reads the candidate row J must keep both the residual-substitution premise row P and obstruction row O visible through the full spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read can use J only as a candidate-normalization row inside that conditional spine. It cannot retype candidate normalization as premise-free confluence, residual-substitution compatibility, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Begin with the candidate-normalization scope theorem, Definition 4238.62. It already requires the consumer to expose $P, K, J, R, S, C, B, O, H, T, G, N$ before the candidate row is read, with P and O still visible.

Next use substitution nonescape, Definition 4238.38. The substitution premise remains confined to the displayed residual route, so reading J cannot discharge P or replace it by an unconditional compatibility theorem.

Apply residual pasting, Definition 4238.37. Pasting two typed-window residual reads through J preserves the same finite residual packet and keeps the obstruction row O in the exported spine.

Finally hand the read to Definition 4238.40. That handoff exports only the conditional normalization-facing route, so the consumer cannot obtain premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency from candidate normalization alone. □

Theorem 3258.64 (MetaCIC candidate-normalization residual budget)**.** *A candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet may spend only the displayed finite residual rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Every normalization-facing read must keep the residual-substitution premise row P, candidate-join row J, residual-frontier row R, and obstruction row O visible. The budget excludes premise-free confluence, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Start with the candidate-normalization scope theorem, Definition 4238.62. It makes the displayed spine $P, K, J, R, S, C, B, O, H, T, G, N$ the only route available to a candidate-normalization consumer.

Use obstruction retention under candidate normalization, Definition 4238.63. This keeps P and O visible when the candidate row J is read, so the residual-substitution premise and obstruction cannot be silently discharged.

Next compose substitution nonescape with residual pasting, Definition 4238.38 and Definition 4238.37. These preserve the residual-frontier row R and candidate-join row J as finite rows of the same packet.

Finally apply the normalization-frontier handoff, Definition 4238.40. The handoff exports only the conditional normalization-facing spine, so the consumer cannot spend the residual budget as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.65 (MetaCIC parallel-diamond candidate-normalization frontier scope)**.** *Every candidate-normalization read used by the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must pass through the visible candidate endpoint row, residual-diamond budget, retained-premise boundary, residual-join locality row, and normal-form obligation row before it can feed a local-confluence consumer. Equivalently, the read remains inside the finite frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where J is read only as the displayed candidate endpoint, R, S, C, B keep the residual-diamond and normal-form budget visible, P and O keep the retained premise and obstruction boundary visible, and H, T, G, N preserve transport, replay, provenance, and local naming. The scope is therefore a bounded candidate-normalization frontier; it is not premise-free local confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with candidate-normalization scope, Definition 4238.62. It requires any consumer of the candidate row to expose the full frontier spine before the candidate endpoint J is read.

Next apply the candidate-mediated strong-normalization handoff of Definition 4238.73. This keeps the candidate endpoint attached to the critical-pair envelope, finite residual witness, bounded checker rows, obstruction row, transport, replay, provenance, and local naming.

Use the residual-join obligation surface, Definition 4238.138. It supplies the residual-diamond budget and local residual-join obligation rows, so a local-confluence consumer cannot read candidate normalization before the residual budget and local join route are visible.

Finally apply the retained-premise closure budget of Definition 4238.150. The retained premise and obstruction rows remain displayed through the normal-form obligation route, so the candidate-normalization read stays inside the bounded frontier and exports none of the refused global endpoints. □

Theorem 3258.66 (MetaCIC candidate-mediated diamond budget route)**.** *Every candidate-mediated parallel-diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose residual-substitution premise retention, critical-pair envelope rows, candidate normalization handoff, bounded obstruction rows, closed checker rows, and local* $\mathsf{NameCert}$ *rows before any confluence-facing conclusion is exported. The read is the finite route through the displayed frontier packet, not full normalization, global confluence, or host definitional equality.*

Proof. Start with the residual-substitution route Definition 4238.39. It keeps the residual-substitution premise row visible through the same packet that carries critical-pair and candidate rows.

Next read the critical-pair envelope and candidate-normalization scope, Definition 4238.50 and Definition 4238.62. These rows expose the local pair and candidate handoff without discharging substitution or turning the handoff into a global normalization theorem.

Finally apply bounded local-confluence decision and finite residual witness consumption, Definition 4238.70 and Definition 4238.69. They keep the bounded obstruction, closed checker, transport, replay, provenance, and local naming rows attached before a confluence-facing consumer can inspect the result. Hence the exported conclusion is only the finite candidate-budget route stated above. □

Theorem 3258.67 (MetaCIC parallel diamond registered frontier route)**.** *A candidate diamond budget route can expose a bounded checker status and local join candidate only after the retained-premise route, candidate-normalization row, residual envelope, and frontier register have all been displayed. The exposed status is a local registered-frontier read from the same* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet; it is not subject reduction, residual-substitution discharge, quotient equality, global normalization, full confluence, or kernel consistency.*

Proof. Start with the seed route of Definition 4238.66. It already forces residual-substitution premise retention, critical-pair envelope rows, candidate normalization handoff, bounded obstruction rows, closed checker rows, and local NameCert rows to remain visible before any confluence-facing conclusion is exported.

Next read the retained-premise and candidate-normalization rows through the same frontier packet. The existing residual-substitution route keeps the premise attached, while the candidate-normalization scope names the local join candidate without turning it into global normalization or subject reduction.

Finally use the frontier and local-join scope rows already recorded in the chapter. They register the bounded checker status only after the residual envelope and frontier display have been read, so the exported route remains a local finite checker route and refuses the global confluence, quotient, discharge, normalization, and consistency endpoints. □

Theorem 3258.68 (MetaCIC parallel-diamond candidate critical-pair boundary)**.** *Every local diamond budget read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, the two critical-pair branch rows carried by K and R, the candidate join row J, the normalization-confluence handoff row, the critical-path socket, and the obstruction row O before any confluence-audit consumer can inspect the read. The boundary remains candidate-mediated; it does not assert subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Begin with seed readiness and public seed readiness, Definition 4238.83 and Definition 4238.84. These rows force the residual branches, candidate join, normalization-confluence handoff, and public frontier route to remain visible before any confluence-facing read is exported.

Next apply the critical-path retained-premise handoff, Definition 4238.85. It keeps the residual-substitution premise and obstruction row attached to the critical-path socket rather than letting the socket discharge the premise.

Finally use the obligation-closure package Definition 4238.87. The package assembles the residual branches, candidate join, normalization-confluence handoff, critical-path socket, and obstruction retention as the finite local surface. Since the surface is still candidate-mediated, none of the refused confluence, equality, normalization, discharge, or kernel endpoints is exported. □

Theorem 3258.69 (MetaCIC parallel-diamond finite residual witness consumption)**.** *Every accepted residual-pair local-diamond witness read in a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must consume the typed window, residual frontier, substitution-boundary row, bounded local-join checker,*

candidate-strong-normalization evidence when present, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it may be used as input to a closed parallel-diamond route. The consumed packet is the finite route through* $P, K, J, R, S, C, B, O, H, T, G, N$*; it is not a proof of arbitrary closed Church–Rosser.*

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierFiniteResidualWitne

Proof. Start with the typed-window local-join scope of Definition 4238.31. It forces a local-diamond consumer to expose the typed window, candidate join, residual frontier, checker rows, and obstruction row before any join read is available.

Next use residual boundary and substitution nonescape, Definition 4238.33 and Definition 4238.38. These keep the residual-substitution premise and substitution-boundary row visible through the finite residual route rather than allowing them to be discharged by structural replay.

Apply the residual-substitution route Definition 4238.39. It identifies the local-diamond input as the conditional frontier packet through $P, K, J, R, S, C, B, O, H, T, G, N$, including transport, replay, provenance, and local naming.

Finally use the candidate-mediated strong-normalization socket Definition 4238.41. When candidate-SN evidence is present, it is consumed as the same finite socket before the closed parallel-diamond route can read the witness. Since the route retains P and O and exports only the displayed packet, it does not prove arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency. □

Theorem 3258.70 (MetaCIC parallel-diamond bounded local-confluence decision)**.** *A bounded local-join checker for the displayed finite residual witness of a* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may return a positive local-confluence certificate only after exposing the typed window, residual frontier, substitution-boundary row, candidate join, bounded checker status, componentwise transport, displayed replay, provenance, and local* NameCert *rows. The positive result is a bounded local-confluence certificate for that finite residual witness; it is not residual-substitution compatibility, subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with local-join scope. By Definition 4238.31, a checker read must expose the typed window, candidate join, residual frontier, checker rows, and obstruction row before any local join can be used.

Next apply residual boundary and substitution nonescape, Definition 4238.33 and Definition 4238.38. These rows keep the substitution-boundary premise visible and prevent the bounded decision from being read as residual-substitution compatibility.

Use finite residual witness consumption from Definition 4238.69. It forces the checker status, candidate join, transport, replay, provenance, and local naming rows to be consumed as the finite route through $P, K, J, R, S, C, B, O, H, T, G, N$.

The checker may therefore certify only that this displayed finite residual witness has a bounded local join. Since the route still retains the typed window and substitution-boundary rows and never exports a global reduction relation, the certificate does not imply subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.71 (MetaCIC parallel-diamond finite candidate frontier)**.** *Any bounded local-confluence certificate consumed from a* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is limited to the displayed finite candidate rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Candidate-normalization rows and residual-substitution rows must remain visible before a closed parallel-diamond route can read the result. The certificate therefore records only a finite candidate frontier for the displayed packet; it is not full Church–Rosser, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency.

Proof. Begin with finite residual witness consumption, Definition 4238.69. It forces every local-diamond witness read to pass through the typed window, residual frontier, substitution-boundary row, bounded local-join checker, transport, replay, provenance, and local naming rows.

Next use bounded local-confluence decision, Definition 4238.70. A positive checker result is available only for that displayed finite residual witness and only after the candidate join, checker status, and structural rows have been exposed.

Now attach the candidate-mediated strong-normalization socket and the residual-substitution route, Definition 4238.41 and Definition 4238.39. These keep candidate normalization and residual substitution inside the same $P, K, J, R, S, C, B, O, H, T, G, N$ spine before the closed route may inspect the handoff.

Thus the consumer receives a finite candidate frontier and nothing stronger. Since the residual-substitution and candidate-normalization rows remain visible, the certificate cannot be reread as a premise-free confluence theorem, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 3258.72 (MetaCIC parallel-diamond frontier residual-row visibility)**.** *Any accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *consumer that exports a parallel-diamond handoff must keep the residual-substitution row P, local join row J, typed residual budget row R, componentwise transport row H, displayed replay row T, provenance row G, and local* NameCert *row N visible. The consumer cannot cite bounded normal equality, closed checker data, or confluence audit packets as a replacement for the residual row.*

Proof. Project the frontier carrier of Definition 4238.1. The accepted packet exposes the residual-substitution premise, candidate or local join, residual-frontier budget, transport, replay, provenance, and local naming rows as displayed coordinates rather than as ambient metadata.

Cite the residual-substitution boundary and local-join ledger sibling routes. The boundary keeps the substitution premise explicit, while the local-join ledger reads the join only after the residual rows and bounded-check rows have been exposed.

The remaining claim is nonescape of the carrier rows. Bounded normal equality, closed checker data, and confluence audit packets can be consumed only through the displayed frontier route; none supplies a second coordinate that discharges or hides the residual-substitution row. □

Theorem 3258.73 (MetaCIC parallel-diamond candidate-mediated SN handoff)**.** *Every candidate-mediated strong-normalization budget consumed by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose, in order, the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before any confluence or decidability consumer reads the frontier. When the candidate-normalization confluence handoff is consumed, the retained residual-substitution premise row P remains visible in the same read. The consumed route is the finite spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retained as the residual-substitution premise, K, J, R retained as the local critical-pair and residual-frontier rows, S, C, B, O retained as the bounded checker and obstruction rows, and H, T, G, N retained as transport, replay, provenance, and local naming. The handoff is therefore a

conditional residual-join frontier and candidate-mediated budget, not a proof of unrestricted confluence, unbounded strong normalization, subject reduction, evaluator equality, quotient term equality, decidability of all reductions, or kernel consistency.

Proof. First read the critical-pair envelope through Definition 4238.50. It exposes K, J, R, S, C, B, O as a bounded local-overlap surface and keeps the frontier carrier rows displayed.

Next read the residual-substitution dependency route of Definition 4238.57. That route places the retained premise row P before the candidate join, residual frontier, closed-checker, and obstruction rows can be consumed.

The candidate-normalization scope of Definition 4238.62 keeps J inside the same conditional spine. It prevents a normalization consumer from hiding P or retyping the candidate row as premise-free confluence.

Together, that candidate-normalization scope and the residual-substitution dependency route require the same retained premise row P to remain visible when the candidate-normalization confluence handoff is consumed; the consumer receives a conditional residual-join frontier, not premise-free confluence or a kernel-consistency certificate.

Now consume the finite residual witness of Definition 4238.69. The witness reads typed windows, residual frontier data, bounded local-join checks, candidate evidence when present, transport, replay, provenance, and local naming before any closed parallel-diamond route can use it.

Finally apply residual-row visibility, Definition 4238.72. The residual row remains visible through H, T, G, N, so confluence and decidability consumers receive only the displayed candidate-mediated budget rather than a discharged substitution theorem, global normalization theorem, evaluator equality, quotient equality, or kernel-consistency certificate. □

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCandidateSnHandoff`

Theorem 3258.74 (MetaCIC parallel-diamond candidate SN handoff scope)**.** *Every candidate-mediated strong-normalization consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row, critical-pair row, candidate join row, residual-frontier row, closed-checker row, closed-normal-fragment row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *before using the handoff. In the displayed spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the consumer reads P as the retained substitution premise, K as the critical-pair row, J as the candidate join row, R as the residual-frontier row, S, C, B as closed-checker, closed-normal-fragment, and bounded-check rows, O as the obstruction row, and H, T, G, N as transport, replay, provenance, and local naming. The scope is only a candidate-mediated SN handoff; it is not full substitution discharge, global confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the candidate-mediated SN handoff of Definition 4238.73. It already forces the accepted packet to display the residual-substitution premise, critical-pair envelope, residual-frontier rows, checker rows, obstruction row, transport, replay, provenance, and local naming before a confluence or decidability consumer reads the frontier.

Next use the retained-premise discharge boundary and residual-substitution dependency route. They keep P visible as a premise row rather than a discharged substitution theorem, so the candidate handoff cannot be read as premise-free confluence or subject reduction.

Finally apply residual-row visibility. The rows K, J, R, S, C, B, O remain attached to H, T, G, N, and every displayed Cont replay preserves the same local NameCert route. Thus the consumer obtains exactly the scoped candidate-mediated handoff and none of the excluded global equality or consistency claims. □

Theorem 3258.75 (MetaCIC parallel diamond typed candidate SN extension)**.** *Every candidate-mediated strong-normalization handoff consumed from an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must retain the residual-substitution premise, critical-pair envelope, candidate row, typed-window row, closed-checker row, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *row before any normalization-facing consumer reads it. In the frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P remains the residual-substitution premise, K, J, R retain the critical-pair, candidate, and typed-window residual data, S, C, B, O retain closed-checker, bounded-check, and obstruction data, and H, T, G, N retain transport, replay, provenance, and local naming. The extension is a typed candidate socket only; it is not global strong normalization, full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the candidate-normalization scope of Definition 4238.62. It forces any candidate-normalization consumer to expose the full spine $P, K, J, R, S, C, B, O, H, T, G, N$ before reading the candidate row.

Next apply the candidate-mediated SN handoff above. The handoff keeps the residual-substitution premise P, critical-pair envelope, candidate row, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before any confluence or decidability consumer reads the frontier.

Use finite residual witness consumption, Definition 4238.69. The typed window and residual frontier data remain in the same finite packet with the closed-checker, bounded-check, candidate evidence, transport, replay, provenance, and local NameCert rows.

Finally invoke residual-row visibility, Definition 4238.72. Since the residual-substitution and typed residual rows remain visible through H, T, G, N, a normalization-facing consumer receives only the typed candidate socket and cannot read it as global strong normalization, full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.76 (MetaCIC parallel-diamond candidate SN frontier extension)**.** *Let $F = (P, K, J, R, S, C, B, O, I$ be an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet. If a consumer displays a candidate-normalization row J, a residual-pair frontier R, a bounded local-join checker B, and the retained substitution-boundary row P through the same transport, replay, provenance, and local naming rows H, T, G, N, then the candidate strong-normalization handoff of F extends to that bounded residual-join frontier. The conclusion is only a bounded candidate-mediated frontier handoff; it does not prove arbitrary closed Church–Rosser, premise-free confluence, global strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. First apply the candidate-mediated SN handoff of Definition 4238.73. It keeps the residual-substitution premise, critical-pair envelope, candidate row, finite residual witness, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before any SN-facing consumer reads the frontier.

Next use the typed candidate SN extension of Definition 4238.75. That theorem places J inside the typed residual window and prevents the candidate row from being retyped as a global normalization theorem.

Now read the residual-frontier siblings. The critical-pair envelope and residual-substitution dependency route, Definition 4238.50 and Definition 4238.57, keep K, R downstream of the retained premise P. Finite residual witness consumption and residual-row visibility, Definition 4238.69 and Definition 4238.72, keep the bounded checker and frontier rows attached to H, T, G, N.

Therefore the displayed candidate-normalization row, residual-pair frontier, bounded local-join checker, and substitution-boundary row form a single bounded handoff. Since every step retains P and the structural rows, the exported result remains a candidate-mediated frontier extension rather than an unrestricted Church–Rosser or global normalization theorem. □

Theorem 3258.77 (MetaCIC parallel-diamond candidate SN retained-premise boundary)**.** *Every candidate-mediated strong-normalization handoff exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet retains the residual-substitution premise row and the obstruction rows before it may feed the normalization-confluence handoff. In the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the rows P, J, R, B, O remain visible as premise, candidate-normalization, residual-frontier, bounded-check, and obstruction data, while H, T, G, N preserve the same route under transport, replay, provenance, and local naming. The boundary is not a proof of subject reduction, full Church–Rosser, evaluator equality, quotient term equality, residual-substitution discharge, global strong normalization, or kernel consistency.

Proof. Start with the candidate-mediated SN handoff of Definition 4238.73. It already forces the consumer to expose the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before any confluence or decidability read.

Now apply the residual-substitution dependency route and candidate-normalization scope, Definition 4238.57 and Definition 4238.62. These rows keep P upstream of J, R, S, C, B, O and prevent the candidate row from being consumed as premise-free normalization or confluence.

Finally use finite residual witness consumption and residual-row visibility, Definition 4238.69 and Definition 4238.72. The bounded witness and obstruction rows remain attached to H, T, G, N, so the normalization-confluence handoff receives only the retained finite packet and none of the excluded global endpoints. □

Theorem 3258.78 (MetaCIC parallel-diamond frontier seed registration)**.** *Every seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that presents a conditional parallel-diamond handoff must first expose the residual-substitution premise row P, critical-pair row K, candidate join J, residual frontier R, closed checker S, closed-normal fragment C, bounded check B, obstruction O, and structural rows H, T, G, N. The handoff is public only at this finite row surface; it does not claim a premise-free diamond, full Church–Rosser theorem, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start from the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the visible rows of the accepted frontier packet, with P retained as an actual residual-substitution premise.

Next use the route theorem Definition 4238.9. A conditional diamond handoff cannot be consumed until the critical-pair, residual-frontier, closed-checker, closed-normal, bounded-check, and obstruction rows have been read alongside the structural rows.

Finally apply premise retention, Definition 4238.13. The premise row remains visible through the local-diamond route, so the seed registration records the finite frontier interface rather than any of the refused confluence or kernel endpoints. □

Theorem 3258.79 (MetaCIC parallel-diamond frontier seed-status route)**.** *The hub seed registration for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is valid only through the retained rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and the sibling critical-path consumer route. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Read the carrier and obligation surfaces, Definition 4238.1 and Definition 4238.2. They make $P, K, J, R, S, C, B, O, H, T, G, N$ the whole retained packet for the seed registration.

Next use the route theorem and seed-readiness theorem, Definition 4238.9 and Definition 4238.83. These force the seed read to keep the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local name rows visible.

Finally apply the critical-path consumer route, Definition 4238.55. The downstream consumer receives only that retained row surface, so the seed-status route remains a registered frontier boundary rather than any of the refused residual-substitution, subject-reduction, confluence, evaluator-equality, quotient-equality, or kernel-consistency endpoints. □

Theorem 3258.80 (MetaCIC parallel-diamond frontier seed register)**.** *The seed register for* MetaCICParallelDiamor *is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

read through the carrier, obligation, seed-registration, and seed-status routes. The register records the chapter as a seed-level frontier packet only; it does not alter the closurestatus theory axis, does not upgrade the verification axis, and does not claim residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier and obligation surfaces of Definition 4238.1 and Definition 4238.2. They expose exactly the rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply Definition 4238.78 and Definition 4238.79. Both routes keep the same retained rows visible before any seed consumer reads the frontier.

The closurestatus block remains in the dedicated route file and reports the independent theory and verification axes. Since this theorem only registers the finite seed surface, it cannot discharge any of the refused MetaCIC endpoints. □

Theorem 3258.81 (MetaCIC parallel-diamond seed registration child route)**.** *The seed registration of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is read through the local child rows for critical-pair envelope, residual-join locality, retained-premise budget, seed closure status, closurestatus guard, and local* NameCert *obligations. Concretely, every seed-registration consumer must expose*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

by way of Definition 4238.50, Definition 4238.104, Definition 4238.150, Definition 4238.45, Definition 4238.180, and Definition 4238.2. The child route records seed registration only; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the carrier and local NameCert obligations. They make the retained premise, critical-pair data, candidate join, residual witness, typed window, bounded checker, obstruction, transport, replay, provenance, and local naming rows the whole public packet.

Read the critical-pair envelope and residual-join locality rows next. These keep the local diamond request carrier-local and keep the residual-substitution premise visible before any seed consumer can inspect a join.

The retained-premise closure budget then prevents the premise row from disappearing under seed registration. The seed closure status reader and closurestatus guard identify the seed marker as a pointer into the same displayed packet, rather than as an independent confluence or kernel claim.

Thus the child route is exactly the finite registration spine for the hub: it ties the seed row to existing child files while refusing residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency. □

Theorem 3258.82 (MetaCIC parallel-diamond frontier seed package)**.** *Every seed consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet receives the seed package only after the existing carrier, obligation surface, handoff route, critical-pair exhaustion rows, obstruction budget, closure seed route, and residual-normalization window have all been displayed. Equivalently, the package is the finite surface*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

read through Definition 4238.1, Definition 4238.2, Definition 4238.9, Definition 4238.66, Definition 4238.100, and Definition 4238.126. It is a registered seed package, not a full confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, global normalization theorem, residual-substitution discharge, or kernel-consistency certificate.

Proof. Start with the carrier and obligation surfaces. They display the retained residual-substitution premise, critical-pair row, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction row, transport, replay, provenance, and local naming rows.

Next read the handoff and critical-pair budget rows. The route theorem and candidate-diamond budget route keep the local diamond inside the displayed candidate window and leave the obstruction budget visible to downstream consumers.

Finally apply the closure seed route and residual-normalization window. These theorems register the same finite package as seed-level data only, so the consumer receives a frontier package rather than any of the refused confluence, subject-reduction, evaluator-equality, quotient-equality, normalization, residual-substitution, or kernel-consistency endpoints. □

Theorem 3258.83 (MetaCIC parallel-diamond frontier seed readiness)**.** *Every candidate-mediated parallel-diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose both residual branches, the common join candidate, the normalization-confluence handoff row, and the MetaCIC critical-path readback socket before any confluence-frontier claim is available. In carrier notation the readiness route keeps* $P, K, J, R, S, C, B, O, H, T, G, N$ *visible:* K *records the two residual branches,* J *records the common join candidate,* C *and* B *route the normalization-confluence handoff consumed by chapter 1190, and the critical-path socket of chapter 765 reads* P, R, O *as retained boundary data. The theorem is a finite route projection, not a normalization theorem, a strong-normalization theorem, an unrestricted confluence theorem, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the frontier carrier from Definition 4238.1. The branch and candidate data are visible only through the critical-pair row K, candidate-join row J, residual-frontier row R, and obstruction row O, with the structural rows attached.

Use the existing confluence handoff route in the chapter. It keeps the local diamond conditional on the residual-substitution premise and routes the closed-normal and bounded-check rows through the normalization-confluence bridge rather than treating them as a global normalization proof.

Finally read the MetaCIC critical-path packet as a socket boundary. The critical-path consumer receives the residual premise, residual windows, and obstruction rows as readback data, so the seed-readiness route exposes the two branches and common join candidate without discharging normalization, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.84 (MetaCIC parallel-diamond public seed readiness)**.** *Every public seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before any confluence-audit or normalization consumer can inspect the handoff. In particular, the residual-substitution premise row P and obstruction row O remain visible together with the critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, transport, replay, provenance, and local naming rows. The public seed read is therefore a retained-premise frontier, not full confluence, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start from Definition 4238.83. It already exposes the two residual branches, the candidate join, and the critical-path readback socket before any frontier claim can be consumed.

Next apply the obligation and route surfaces, Definition 4238.2 and Definition 4238.9. These force the public read through the rows $P, K, J, R, S, C, B, O, H, T, G, N$ rather than through a detached confluence theorem.

Use the premise-retention and obstruction ledgers already present in the chapter, including Definition 4238.23 and Definition 4238.36. They keep P and O visible before confluence-audit or normalization consumers inspect the candidate-mediated handoff.

Finally compose with the critical-path sibling route. The downstream consumer receives only the finite seed-readiness spine, so the packet does not discharge residual substitution, prove subject reduction, establish unrestricted confluence, identify evaluator or quotient equality, prove global normalization, or certify kernel consistency. □

Theorem 3258.85 (MetaCIC parallel-diamond critical-path retained-premise handoff)**.** *Any critical-path consumer reading the candidate-mediated local diamond row of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise P and the obstruction row O visible throughout the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The packet may be passed to $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and the normalization-confluence bridge of chapter 1190 only as retained frontier evidence. It is not a proof of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Start from public seed readiness, Definition 4238.84. It forces any public read of the frontier through the full route $P, K, J, R, S, C, B, O, H, T, G, N$ before a confluence-audit or normalization consumer may inspect the handoff.

Use the seed-readiness route Definition 4238.83. The candidate-mediated local diamond exposes the residual branches, common join candidate, normalization-confluence handoff row, and critical-path readback socket, while leaving the residual-substitution premise as retained boundary data.

Apply the premise preservation and critical-path consumer routes already named in the chapter, including Definition 4238.23 and Definition 4238.55. These keep P and O visible when the handoff is read by $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, or the normalization-confluence bridge.

The obstruction row then blocks endpoint promotion. Since O remains part of the exported frontier evidence, the consumer cannot reinterpret the handoff as subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency. □

Theorem 3258.86 (MetaCIC parallel-diamond residual-substitution local discharge)**.** *When an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet displays a residual-substitution row together with the critical pair, candidate join, residual-frontier, closed-checker, bounded-check, obstruction, transport, replay, provenance, and local* NameCert *rows*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

that packet locally discharges the finite residual window for the current candidate-mediated diamond only. The discharge is the visible closure of that displayed window under the present candidate join and retained obstruction; it is not subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with seed readiness and public seed readiness, Definition 4238.83 and Definition 4238.84. They expose the two residual branches, candidate join, residual-frontier row, closed-checker row, bounded-check row, obstruction row, and structural rows as one finite packet.

Apply the critical-path retained-premise handoff, Definition 4238.85. It keeps the residual-substitution premise P and obstruction row O visible while the critical-path consumer reads the candidate-mediated local diamond.

Use premise preservation together with the obligation-closure package, Definition 4238.23 and Definition 4238.87. These rows preserve the displayed premise and obstruction across the finite window and assemble the local obligation surface without promoting it to a global metatheorem.

Thus the only discharged object is the current finite residual window under the displayed candidate join. The packet has no coordinate for subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency, so none of those endpoints is exported. □

Theorem 3258.87 (MetaCIC parallel-diamond frontier obligation-closure package)**.** *The seed-readiness row of Definition 4238.83, the public seed-readiness row of Definition 4238.84, and the critical-path retained-premise handoff of Definition 4238.85 assemble the local obligation surface for residual branches, candidate join, normalization-confluence handoff, critical-path socket, and obstruction retention in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. The package refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, premise-free residual-substitution discharge, and kernel consistency.*

Proof. Project the displayed frontier rows. The carrier surface keeps the residual-substitution premise, two residual branches, candidate join, residual frontier, closed-checker row, normalization-confluence handoff rows, obstruction row, transport, replay, provenance, and local naming rows visible.

Apply Definition 4238.83. It places the residual branches, common join candidate, normalization-confluence handoff, and critical-path socket before any confluence-frontier read.

Next apply Definition 4238.84 and Definition 4238.85. These rows preserve the public route and keep the residual premise together with the obstruction row under critical-path consumption, so the package records obligation closure for the finite frontier surface without promoting it to subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, premise-free residual-substitution discharge, or kernel consistency. □

Theorem 3258.88 (MetaCIC parallel-diamond frontier seed closure)**.** *Any seed-closure read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

visible, retain the residual-substitution premise P*, and preserve the obstruction row* O *before any strong-normalization, confluence-audit, or normalization consumer reads the handoff. The seed closure is a finite retained-premise frontier and not global normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Start with the seed-readiness row of Definition 4238.83. It exposes the residual branches, common join candidate, normalization-confluence handoff row, and critical-path readback socket while keeping the route inside the displayed carrier rows.

Apply the public seed-readiness row of Definition 4238.84. This forces the full spine $P, K, J, R, S, C, B, O, H, T, G$ to remain visible before any confluence-audit or normalization consumer inspects the handoff.

Next use the critical-path retained-premise handoff of Definition 4238.85. It keeps the residual-substitution premise P and obstruction row O visible under $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, and normalization-confluence consumption.

Finally compose with the obligation-closure package of Definition 4238.87. The resulting seed closure records only retained finite frontier evidence, so it cannot discharge residual substitution or promote the handoff to strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.89 (MetaCIC parallel-diamond frontier seed obligations)**.** *Every confluence consumer reading an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must first expose the seed-obligation surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The exposed rows include candidate pair rows, one-step residual rows, join-window rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the carrier-local* NameCert *row. A downstream reader may inspect a diamond conclusion only after these rows have been displayed as local frontier data; the seed surface is not subject reduction, global normalization, unrestricted confluence, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the accepted carrier from Definition 4238.1. The candidate pair and one-step residual data are carried by the critical-pair, candidate-join, and residual-frontier rows K, J, R, with the retained premise row P still visible.

Read the join-window and obstruction rows next. The closed-checker, closed-normal, bounded-check, and obstruction coordinates S, C, B, O keep the local join inside the displayed window instead of promoting it to a global confluence theorem.

Finally apply the structural rows. Componentwise transport H, replay T, provenance G, and local naming N preserve the same row order, so a consumer reaches a diamond conclusion only

through the finite seed-obligation surface and not through subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 3258.90 (MetaCIC parallel-diamond retained-obstruction obligation)**.** *Any confluence or normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise row P and obstruction row O visible through the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If either row is hidden, discharged, or replaced by an endpoint assertion, the read is not a discharge of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. The rows $P, K, J, R, S, C, B, O, H, T, G, N$ are the local data available to a confluence or normalization consumer, and the residual-substitution premise row P remains part of that display.

Apply Definition 4238.89. Its seed surface already forces the consumer to read the candidate pair, residual rows, join-window rows, closed-checker rows, obstruction row, transport, replay, provenance, and local naming before any diamond conclusion is inspected.

Now use the obstruction row O. It records that the accepted packet has not promoted the local diamond to subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency, so hiding P or O would erase a load-bearing premise rather than discharge it.

The public seed readiness of the chapter therefore remains conditional. A downstream consumer may pass along the finite candidate-mediated diamond surface only while P and O remain visible through the full route, and any endpoint promotion must be supplied by a separate packet. □

Theorem 3258.91 (MetaCIC parallel diamond frontier obligation closure route)**.** *Every residual-join or candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, residual substitution dependency, candidate diamond budget, finite residual witness consumption, and residual row visibility before reading any local diamond conclusion. In row form, the consumer must display the route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with P retaining the residual-substitution premise, K exposing the critical-pair envelope, J and B carrying the candidate diamond budget, R exposing finite residual witnesses, S and C checking the closed candidate surface, O retaining the obstruction, and H, T, G, N transporting, replaying, sourcing, and naming the same route. The conclusion is obligation closure for the displayed frontier only; it is not subject reduction, global confluence, strong normalization, substitution admissibility, hidden oracle access, or any closed MetaCIC theorem.

Proof. Begin with Definition 4238.89. It forces every confluence consumer to display the seed rows $P, K, J, R, S, C, B, O, H, T, G, N$ before a local diamond conclusion is inspected.

Use the retained-obstruction obligation of Definition 4238.90. The rows P and O remain load-bearing, so the residual-substitution dependency and the obstruction cannot be hidden when the residual join is read.

Now compose with the adjacent residual route files. The residual-witness and candidate-normalization routes expose finite witness consumption through R and the candidate budget through J, S, C, B before the frontier can export a local join.

The structural rows H, T, G, N keep the same displayed packet under transport, replay, provenance, and local naming. Since no row supplies subject reduction, global confluence, strong normalization, substitution admissibility, hidden oracle access, or a closed MetaCIC theorem, the route closes only the finite obligation surface it displays. □

Theorem 3258.92 (MetaCIC parallel-diamond discharge boundary)**.** *A candidate-mediated diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet can be consumed only while the residual-substitution premise row P and obstruction row O remain visible through the route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read is therefore a retained frontier surface. It does not discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. The candidate-mediated diamond data are displayed only through the rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply the seed-obligations theorem. It requires every confluence consumer to expose the retained premise, candidate pair, residual rows, join window, checker rows, obstruction row, transport, replay, provenance, and local naming before a diamond conclusion is inspected.

Now use the retained-obstruction obligation. That theorem keeps P and O visible as load-bearing rows, so a downstream read may pass along the candidate-mediated diamond surface only as a conditional frontier.

The excluded endpoint claims have no coordinate in the displayed route. Subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency would require separate discharge data, not the retained candidate packet alone. □

Theorem 3258.93 (MetaCIC parallel-diamond frontier obligation scope)**.** *Every accepted* MetaCICParallelDiamond *packet exposes the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N before any confluence consumer reads a parallel-diamond handoff. The visible scope is*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It is a retained frontier surface, not a discharge of residual substitution, full Church–Rosser, subject reduction, evaluator equality, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. Its finite coordinates are exactly the retained premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and naming rows listed above.

Apply Definition 4238.89. That obligation surface forces a confluence consumer to display the candidate pair, residual rows, join-window rows, closed-checker rows, obstruction row, and structural certificate rows before inspecting any local diamond conclusion.

Use Definition 4238.90 for the retained boundary. The rows P and O remain visible as load-bearing premises, so the handoff cannot be read as residual-substitution discharge, full Church–Rosser, subject reduction, evaluator equality, or kernel consistency. □

Theorem 3258.94 (MetaCIC parallel-diamond critical-pair join)**.** *Every accepted finite critical-pair envelope for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *whose residual rows pass the candidate diamond budget admits a displayed join route through the finite surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route reads residual substitution from P, residual-row visibility from K and R, candidate-normalization from J, checker and bounded-budget rows from S, C, B, the retained obstruction from O, and then transport, replay, provenance, and local naming from H, T, G, N. The conclusion is a candidate-mediated local join for that displayed envelope, not a global confluence assumption, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency claim.

Proof. Project the accepted carrier from Definition 4238.1. The finite critical-pair envelope is visible only through $P, K, J, R, S, C, B, O, H, T, G, N$, so the candidate diamond budget is read as a condition on the displayed residual rows, not as a global confluence premise.

Split by the displayed residual envelope. In the cases where the residual rows fail the candidate budget, the theorem has no accepted join route to export. In the cases where the budget passes, the critical-pair row K and residual-frontier row R expose the two residual branches, while P keeps the residual-substitution premise visible.

Read the candidate-normalization and checker rows next. The candidate-join row J supplies the named local join proposal, and the rows S, C, B verify that the proposal stays inside the closed checker, closed-normal, and bounded-budget surface.

Finally apply the obstruction and structural rows. The retained obstruction O prevents the local join from being reread as Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency, while H, T, G, N transport, replay, source, and name the same finite route. Thus the accepted envelope admits a displayed local join exactly through the candidate-mediated surface. □

Theorem 3258.95 (MetaCIC parallel-diamond frontier seed scope)**.** *Before any candidate-mediated strong-normalization route is exported from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the only seed-scope reads are finite critical-pair windows, residual substitution rows, candidate-normalization rows, and local join ledgers. In carrier notation these are the displayed rows $P, K, J, R, S, C, B, O, H,$ read as retained frontier data, not as subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Project the accepted carrier from Definition 4238.1. The finite critical-pair windows are carried by K and R, the residual-substitution row remains P, and the candidate-normalization read is routed through J, C, B with the obstruction row O still visible.

Apply the seed-obligation and seed-readiness routes, Definition 4238.89 and Definition 4238.83. They force every public seed read through the displayed frontier rows before any confluence-audit or normalization-facing handoff can be inspected.

The local join ledger is then completed only by transport, replay, provenance, and naming rows H, T, G, N. Since no carrier coordinate exports subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency, the stated rows are exactly the seed-scope reads available before the candidate-mediated strong-normalization route is exported. □

Theorem 3258.96 (MetaCIC parallel-diamond scoped-status route)**.** *The scoped closure status for an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet is witnessed by the finite dependency*

chain consisting of residual-substitution scope, premise preservation, critical-pair exhaustion, local-join scope, normalization-frontier handoff, candidate-mediated strong-normalization socket, typed-window candidate strong-normalization route, closed-substitution window scope, finite candidate strong-normalization extension, and residual-join induction. The chain exposes exactly the rows $P, K, J, R, S, C, B, O, H, T, G, N$ and does not assert residual-substitution compatibility, full parallel diamond, subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the premise row P attached to every critical-pair, residual-frontier, closed-checker, and closed-normal read.

Read critical-pair exhaustion and local-join scope through Definition 4238.4 and Definition 4238.31. These routes place the candidate join inside the displayed K, J, R, S, C, B, O rows before a confluence-facing consumer can inspect the packet.

Next apply the normalization-frontier handoff, candidate-mediated strong-normalization socket, and typed-window candidate route, Definition 4238.40, Definition 4238.41, and Definition 4238.42. Together they route every normalization-facing read through the same retained-premise frontier rather than through a global normalization theorem.

Finally use the closed-substitution window scope, finite candidate strong-normalization extension, and residual-join induction, Definition 4238.43, Definition 4238.44, and Definition 4238.46. The structural rows H, T, G, N transport, replay, source, and name the same finite packet, so the scoped-status evidence remains inside the displayed frontier rows and excludes the refused endpoints. □

Theorem 3258.97 (MetaCIC parallel-diamond obligation-status route)**.** *Every obligation-status read for an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise P, critical-pair rows K, R, candidate join J, closed checker S, closed-normal fragment C, bounded check B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N visible before any confluence-audit, decidability, or normalization consumer inspects the frontier. The route is a retained-premise obligation surface; it does not discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution compatibility, or kernel consistency.*

Proof. Begin with the seed scope of Definition 4238.95. It fixes the visible frontier rows $P, K, J, R, S, C, B, O, H, T, G$ before any candidate-mediated strong-normalization route can be exported.

Apply the scoped-status route Definition 4238.96. The dependency chain it records already passes through residual-substitution scope, premise preservation, critical-pair exhaustion, local-join scope, normalization-frontier handoff, candidate-mediated strong-normalization, closed-substitution window scope, finite candidate extension, and residual-join induction.

Now read the candidate strong-normalization handoff and residual-join induction in that order. The handoff keeps the candidate join J, closed checker S, closed-normal fragment C, bounded check B, and obstruction row O attached to the same frontier, while the residual-join induction keeps the premise P and the critical-pair rows K, R visible.

Finally use the critical-path retained-premise handoff carried by the displayed transport, replay, provenance, and local naming rows H, T, G, N. These rows identify the same packet for downstream confluence-audit, decidability, and normalization consumers. Since no row exports subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency, the obligation-status route remains the retained-premise surface stated above. □

Theorem 3258.98 (MetaCIC parallel-diamond retained-premise consumer route)**.** *Any normalization, confluence-audit, or decidability consumer citing an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first read the retained premise row P, critical-pair rows K, R, candidate join J, closed checker S, normal fragment C, bounded check B, obstruction row O, and structural rows H, T, G, N. The citation is a finite retained-premise route through those displayed rows; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the seed scope of Definition 4238.95. It fixes $P, K, J, R, S, C, B, O, H, T, G, N$ as the only rows visible before a candidate-mediated strong-normalization route is exported.

Apply the scoped-status route of Definition 4238.96. This keeps the residual-substitution premise P attached while the critical-pair rows K, R, candidate join J, closed checker S, normal fragment C, bounded check B, and obstruction row O are inspected.

Finally use the obligation-status route of Definition 4238.97. It requires the structural rows H, T, G, N before any downstream consumer treats the frontier as usable evidence, and it explicitly leaves residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency outside the packet. □

Theorem 3258.99 (MetaCIC parallel-diamond seed closure)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ be an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet. Any candidate-mediated strong-normalization handoff from F must first expose the critical-pair envelope, the residual-substitution dependency route, and the candidate-normalization scope rows of the same packet. More explicitly, the handoff reads K, J, R, S, C, B, O through the envelope of Definition 4238.50, retains P through the dependency route of Definition 4238.57, and keeps J inside the candidate-normalization scope of Definition 4238.62. The structural rows H, T, G, N preserve that same packet by* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The conclusion is only seed-level row visibility before the SN handoff; it proves no global normalization theorem, no unrestricted confluence theorem, no subject reduction theorem, no evaluator equality, no quotient term equality, and no kernel-consistency certificate.*

Proof. Begin with the critical-pair envelope. By Definition 4238.50, the packet exposes critical-pair requests only through the finite route $P, K, J, R, S, C, B, O, H, T, G, N$.

Next retain the residual-substitution dependency. The route of Definition 4238.57 places P before the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows can be consumed.

Now constrain the candidate-normalization read. The scope theorem Definition 4238.62 keeps J inside the same conditional spine and prevents a candidate-mediated consumer from hiding P or O.

Finally compare with the SN handoff of Definition 4238.73. That theorem consumes the same envelope, dependency route, candidate-normalization scope, finite residual witness, and residual-row visibility surfaces. The present theorem isolates the seed-level visibility prerequisite: H, T, G, N preserve the displayed packet, but they do not discharge residual substitution or promote the handoff to global normalization, unrestricted confluence, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.100 (MetaCIC parallel-diamond frontier closure seed route)**.** *Every seed-closure read for an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row, critical-pair row, candidate-join row, residual-frontier row, closed checker seal, closed-normal fragment, bounded-check row, and obstruction row before any conditional parallel-diamond*

route is consumed. The route is a finite seed surface for the local frontier; it does not prove residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. The residual-substitution premise, critical-pair data, candidate joins, residual frontier, closed checker, closed-normal fragment, bounded-check row, and obstruction row are displayed carrier coordinates before a confluence consumer can read any diamond-shaped conclusion.

Apply Definition 4238.2. The obligation surface requires carrier admission, row visibility, transport, replay, provenance, and local naming for exactly those finite frontier rows.

Use Definition 4238.3 and Definition 4238.50. These keep the conditional handoff downstream of the displayed critical-pair and residual rows, and they prevent the handoff from replacing the obstruction budget by a global confluence theorem.

Finally apply Definition 4238.6. The consumed route is therefore finite and seed-local: it records the displayed obstruction frontier and does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.101 (MetaCIC parallel-diamond frontier closure seed current route)**.** *The seed marker of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is readable only through the carrier of Definition 4238.1, the obligation surface of Definition 4238.2, and the closurestatus guard of Definition 4238.180. A status consumer must therefore expose the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it may treat the seed registration as available. The route does not claim subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the carrier of Definition 4238.1. Its displayed coordinates contain the residual-substitution premise, critical-pair data, candidate joins, bounded checker, obstruction row, and structural rows that any seed reader must expose.

Apply the obligation surface of Definition 4238.2. It makes the visible rows the local NameCert inventory for the packet, so the seed marker cannot be consumed as an independent confluence theorem.

Finally read the closurestatus guard of Definition 4238.180. The guard routes every status reader through the same finite packet before seed or bridged markers are consumed, and the packet has no coordinate for subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.102 (MetaCIC parallel-diamond frontier closure route)**.** *The seed-facing* MetaCICParallelDiamondF *closure route is exactly the finite frontier route already recorded by the local residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows. It is read through the retained premise boundary of Definition 4238.155, the seed closure route of Definition 4238.189, and the seed-status block of Definition 4238.205.*

The route records a finite seed surface only. It does not assert global confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, unrestricted normalization, or kernel consistency.

Proof. Start with the retained premise boundary. It keeps the residual-substitution premise, critical-pair, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, prove-

nance, and local naming rows visible whenever the candidate join and residual frontier are consumed.

Next read the seed closure route. That theorem attaches the same local rows to the critical-pair envelope, residual-join locality route, retained-premise budget, local NameCert obligations, seed-status row, closurestatus export, and closurestatus guard. Thus the child ledger has a single ordered seed route rather than separate hidden assumptions.

Finally apply the seed-status block. It identifies the seed-facing status read with the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$, so the chapter's closure surface records only the finite frontier rows already available in the local packet.

Because every cited step remains inside the displayed route, none of the refused global endpoints is exported by the closure record. □

Theorem 3258.103 (MetaCIC parallel-diamond scoped seed package route)**.** *The hub-level scoped marker for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is read through the same finite seed package exposed by the spine: residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed-checker rows, closed-normal fragment rows, bounded-check rows, obstruction rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route records the seed-status package already scoped by Definition 4238.48; it does not add subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Read the scoped closure status theorem in the spine. It identifies the accepted packet by the finite route through $P, K, J, R, S, C, B, O, H, T, G, N$ and refuses the global confluence and kernel endpoints.

The hub marker is therefore a route into that same package rather than a second closure claim. It first records the residual-substitution premise, critical-pair, candidate, residual-frontier, checker, normal-fragment, bounded-check, and obstruction rows.

Finally the structural rows transport, replay, source, and name the packet. Thus the scoped marker is accounted for by a concrete child ledger while the top-level file remains structural. □

Theorem 3258.104 (MetaCIC parallel-diamond frontier residual-join locality)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *residual join is read through the local frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read must first expose the critical-pair envelope of Definition 4238.50, the residual-substitution dependency route of Definition 4238.57, the candidate-normalization scope of Definition 4238.62, the finite residual witness consumption row of Definition 4238.69, and the carrier-local NameCert *row of Definition 4238.2. It exports only a bounded residual-join locality certificate for the displayed packet, not global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualJoinLocalit`

Proof. Project the accepted frontier carrier of Definition 4238.1. The only rows available to a residual-join consumer are the finite packet rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Read the critical-pair and residual-substitution rows first. The critical-pair envelope keeps K, J, R, S, C, B, O visible, while the dependency route places the retained residual-substitution premise P before any local join can be consumed.

Next read candidate normalization and finite residual witnesses. The candidate-normalization scope keeps J inside the same conditional spine, and the finite residual witness theorem consumes typed windows, residual frontier data, bounded checker rows, transport, replay, provenance, and local naming before a closed parallel-diamond route can inspect the join.

Finally use the local NameCert obligation surface. Since H, T, G, N preserve only this finite packet and the obstruction row O remains visible, the residual join is local to the displayed frontier and cannot be promoted to global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.105 (MetaCIC parallel-diamond residual-join locality budget)**.** *Every accepted residual-join locality read for a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet keeps the residual premise row P, critical-pair row K, candidate-join row J, residual witness row R, typed-window row S, bounded checker row B, obstruction row O, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N inside the displayed finite packet before any normalization or confluence consumer reads it. The budget is*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

and it exports no residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality, Definition 4238.104. It already forces every join read through the displayed frontier rows and keeps the retained premise and obstruction rows visible.

Next use finite residual witness consumption, Definition 4238.69. That route requires the typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming to be exposed before a local join is available to a downstream consumer.

The two routes combine as a finite budget. Rows P, K, J, R, S, C, B, O remain the residual, critical-pair, candidate-join, witness, window, certificate, checker, and obstruction surface, while H, T, G, N preserve the same packet under transport, replay, provenance, and local naming.

Since the carrier contains no coordinate for residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency, a normalization or confluence consumer can read only the displayed finite budget. □

Theorem 3258.106 (MetaCIC parallel-diamond residual-substitution window)**.** *Every candidate-mediated parallel-diamond read that invokes residual substitution must first display the critical-pair envelope, residual-substitution dependency row, finite residual witness, residual-join locality row, and candidate-normalization row before any strong-normalization or confluence-facing handoff is public. The public window is the carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not a hidden substitution oracle, unrestricted confluence theorem, host conversion equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with the local residual-join route of Definition 4238.104. It forces the accepted read through the displayed frontier rows and keeps the obstruction row visible.

Read the critical-pair and residual-substitution rows next. The critical-pair envelope of Definition 4238.50 exposes the bounded overlap data, while the residual-substitution dependency route of Definition 4238.57 keeps the premise row P in the carrier-local window.

Now attach the finite residual witness and candidate-normalization scope rows. The finite residual witness consumption theorem keeps the typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming visible, and Definition 4238.62 keeps the candidate join inside the same conditional spine.

These rows compose to exactly the displayed residual-substitution window. Since the accepted packet contains no coordinate for a substitution oracle, unrestricted confluence theorem, host conversion equality, subject reduction, evaluator equality, quotient term equality, or kernel consistency, no such handoff is exported. □

Theorem 3258.107 (MetaCIC parallel-diamond residual grid)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *residual-join consumer receives a finite grid whose rows are the critical-pair envelope, residual-substitution dependency, candidate-normalization scope, finite residual witness, carrier-local* NameCert*,* hsame *transport,* Cont *replay, and* Pkg *provenance rows before any local diamond handoff is exported. The grid is the displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not a global confluence theorem, strong-normalization theorem, subject-reduction theorem, quotient term equality, or evaluator equality.

Proof. Begin with Definition 4238.104. It already forces a residual-join read through $P, K, J, R, S, C, B, O, H, T, G,$ and refuses to promote the local route to global confluence or unbounded normalization.

Now refine the local route into grid rows. The critical-pair envelope of Definition 4238.50 supplies the bounded overlap row, the residual-substitution dependency route of Definition 4238.57 places P before any join handoff, the candidate-normalization scope of Definition 4238.62 keeps J inside the conditional spine, and finite residual witness consumption from Definition 4238.69 supplies the typed-window and bounded-check rows.

Finally attach the carrier-local NameCert row and structural rows. The obligation theorem Definition 4238.2 supplies the local naming surface, while H, T, G, N transport, replay, source, and name the same carrier. Since every grid row is finite and carrier-local, the exported handoff cannot become strong normalization, subject reduction, quotient term equality, evaluator equality, or kernel consistency. □

Theorem 3258.108 (MetaCIC parallel-diamond residual-join locality scope)**.** *The residual-join locality surface of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is scoped exactly by the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The rows P, K, J, R, S, C, B, O are the BHist*-visible confluence-frontier rows for residual premises, critical pairs, local join requests, typed windows, residual witnesses, bounded checker data, boundary certificates, and visible obstructions. The rows H, T, G, N are precisely the* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows for that same packet. No residual-join locality read exports global confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, or premise-free residual substitution.*

Proof. Begin with Definition 4238.104. It already forces every residual-join read through the displayed frontier rows and keeps the obstruction row visible.

Refine the route by Definition 4238.107. The grid separates the confluence-facing rows P, K, J, R, S, C, B, O from the structural rows H, T, G, N, while keeping both groups inside one finite packet.

Now apply the seed obligations and carrier obligations. The critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness consumption, and carrier-local NameCert obligation theorem provide no extra coordinate beyond the rows displayed above.

Therefore a consumer that has read $P, K, J, R, S, C, B, O, H, T, G, N$ has exhausted the residual-join locality surface. A request for global confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, or premise-free residual substitution would require a carrier coordinate that the packet does not contain. □

Theorem 3258.109 (MetaCIC parallel-diamond scoped residual locality)**.** *Any accepted residual join in a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose, before a local diamond read is public, the residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the bounded checker row B, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and the local* NameCert *row N. The read is a scoped residual-locality certificate over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These force the accepted join to pass through the displayed residual-substitution premise, critical-pair envelope, candidate join, finite residual witness, bounded checker, transport, replay, provenance, and local naming rows.

Next use the locality-scope theorem, Definition 4238.108. It identifies the residual-frontier and obstruction rows as part of the same finite carrier-local surface rather than as hidden global assumptions.

Finally apply finite residual witness consumption and residual-join witness locality, Definition 4238.69 and Definition 4238.116. They make the bounded checker rows visible before the local join can be exported and keep P and O inside the structural rows H, T, G, N.

Since the carrier contains only the displayed finite rows, the public read remains a local diamond certificate. Full Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency would require coordinates not present in the packet. □

Theorem 3258.110 (MetaCIC parallel-diamond residual-join locality exhaustion)**.** *A residual-join consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that has read*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

has exhausted the carrier-local join surface. Any request for global confluence, subject reduction, unbounded normalization, evaluator equality, quotient term equality, or kernel consistency is outside the packet.

Proof. Apply Definition 4238.104. It forces every residual-join read through the displayed local frontier route and keeps the obstruction row visible.

Then apply Definition 4238.107. The grid identifies the critical-pair, residual-substitution, candidate-normalization, finite-witness, transport, replay, provenance, and local naming rows as the whole carrier-local residual surface.

Finally use the carrier-local NameCert obligation theorem Definition 4238.2. Since that obligation surface admits no coordinate for global confluence, subject reduction, unbounded normalization, evaluator equality, quotient term equality, or kernel consistency, a consumer that has read the grid has no further join row to project. □

Theorem 3258.111 (MetaCIC parallel-diamond residual obstruction budget)**.** *Any candidate-mediated residual join in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that consumes the local join row J and bounded checker rows must keep both the retained premise row P and the obstruction row O visible through the structural rows H, T, G, N. The handoff is only a bounded residual-join locality budget for*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It cannot be read as residual-substitution discharge, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality, Definition 4238.104. It forces every candidate-mediated join to read the displayed frontier rows and keeps the obstruction coordinate inside the same packet.

Use the residual grid theorem, Definition 4238.107. The grid separates the retained premise, critical-pair, local join, residual witness, bounded checker, and obstruction rows from the transport, replay, provenance, and local naming rows.

Now apply the locality-scope theorem, Definition 4238.108. It identifies P and O as visible rows of the residual-join surface rather than side conditions that can be discharged after J is consumed.

Finally use residual-join locality exhaustion. Once the consumer has read $P, K, J, R, S, C, B, O, H, T, G, N$, there is no further coordinate for residual-substitution discharge, subject reduction, confluence, normalization, evaluator equality, quotient term equality, or kernel consistency. The obstruction budget is therefore retained as a carrier-local boundary. □

Theorem 3258.112 (MetaCIC parallel-diamond residual-premise exhaustion)**.** *Any accepted residual-join read of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that has consumed*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

has not discharged the retained premise row P. It has only made P visible as part of the finite carrier-local grid for the residual join. Consequently the read cannot be used as premise-free residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These place every accepted join inside the displayed carrier-local rows and identify P as a row of that grid, not as a premise erased by the join handoff.

Apply the locality-scope theorem and the residual obstruction budget, Definition 4238.108 and Definition 4238.111. They keep the retained premise row and obstruction row visible through H, T, G, N after the local join and bounded checker rows have been read.

Finally use residual-join witness locality, Definition 4238.116. The witness consumes finite residual data before exporting the local join, but it does not add any coordinate for premise-free residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency. The consumed packet therefore exhausts visibility of the retained premise rather than discharging it. □

Theorem 3258.113 (MetaCIC parallel-diamond residual-premise transport exhaustion)**.** *An accepted residual-join read of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet transports retained premises only through the displayed residual locality route, candidate-normalization scope, finite residual witness consumption rows, and critical-pair envelope rows. After the read has consumed*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the transported premise surface is exhausted. No consumer can extend the read to residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global confluence, strong normalization, or kernel consistency.

Proof. Project the accepted carrier through the residual-premise exhaustion theorem, Definition 4238.112. It identifies P as a retained premise row inside the finite grid rather than as a discharged assumption.

Next read the locality and normalization rows. Residual-join locality, Definition 4238.104, keeps the premise inside the displayed frontier route, while candidate-normalization scope from Definition 4238.62 keeps the candidate join in the same conditional spine.

Now consume the finite witness and critical-pair rows. The finite residual witness theorem Definition 4238.69 exposes the bounded residual data before the local join is available, and the critical-pair envelope of Definition 4238.50 keeps the overlap rows inside the same carrier-local packet.

Finally inspect the carrier rows that remain after transport through H, T, G, N. They replay and name only the displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$. Since the carrier has no coordinate for residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global confluence, strong normalization, or kernel consistency, the premise transport is exhausted at that boundary. □

Theorem 3258.114 (MetaCIC parallel-diamond frontier residual local confluence)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If the bounded checker rows accept both residual branches from the displayed critical-pair and residual-premise surface, then the consumer receives a carrier-local diamond handoff between the two residual endpoints through the displayed candidate join row J. The handoff is bounded by the same packet rows and does not claim global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. First read the residual grid of Definition 4238.107. It exposes the critical-pair envelope, residual-substitution dependency, candidate-normalization scope, finite residual witness, local NameCert row, transport, replay, and provenance as one finite packet.

Next apply residual-premise transport exhaustion, Definition 4238.113. It keeps the transported premise surface inside $P, K, J, R, S, C, B, O, H, T, G, N$ after both residual branches have been checked.

Now consume the finite witness and checker rows using Definition 4238.69. The bounded witness exposes the two residual endpoints before the candidate join row can be handed to a consumer.

Finally apply residual-join witness locality, Definition 4238.116. It keeps J carrier-local, retains P and O through H, T, G, N, and prevents the bounded handoff from becoming global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.115 (MetaCIC parallel-diamond candidate-mediated strong-normalization extension)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the residual rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Suppose a finite candidate packet reads its residual-premise row through Definition 4238.112 and its residual-join row through Definition 4238.104. Then the packet extends along the candidate-mediated frontier without leaving the same finite residual budget. A strong-normalization-facing handoff may consume the local diamond row J together with $P, K, R, S, C, B, O, H, T, G, N$ rather than selecting global normal forms.

Proof. First read residual-premise exhaustion. It keeps P as a retained premise row inside the displayed residual grid, so the candidate packet cannot erase the premise before the strong-normalization-facing handoff is formed.

Next apply residual-join locality and residual-premise transport exhaustion. These rows keep the candidate join J, residual frontier rows, checker rows, and obstruction row inside the same packet after transport through H, T, G, N.

Now consume the finite residual witness using Definition 4238.69. The witness exposes the bounded residual data and checker rows before the local diamond can be exported.

Finally apply residual local confluence, Definition 4238.114. The local diamond row is already a carrier-local handoff over the finite residual budget, so a strong-normalization-facing consumer receives that displayed row rather than a selected global normal form. □

Theorem 3258.116 (MetaCIC parallel-diamond residual-join witness locality)**.** *Every accepted residual-join witness for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must read the finite residual witness row and the bounded checker rows before exporting the local join row. The read retains both the premise row P and the obstruction row O through the structural rows H, T, G, N, so its carrier surface is still*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It does not discharge residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. First apply finite residual witness consumption, Definition 4238.69. The witness read is forced through typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming before any local join row is exported.

Next apply residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These results place the join in the displayed finite route and keep the premise row P inside the same carrier-local grid as the critical-pair, witness, checker, and obstruction rows.

Finally apply the residual obstruction budget, Definition 4238.111. It keeps O visible through H, T, G, N and prevents the local join witness from being read as residual-substitution discharge, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.117 (MetaCIC parallel-diamond residual-join locality budget)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *residual-join consumer must expose the critical-pair envelope, residual-substitution dependency row, candidate-normalization scope, finite residual witness, and local join rows before any parallel-diamond handoff is public. The exposed packet is the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the premise dependency, K carrying the critical-pair envelope, J naming the local join request, R, S, C, B carrying the residual witness and bounded checker rows, O keeping the visible obstruction row, and H, T, G, N supplying transport, replay, provenance, and local naming.

The budget exports only carrier-local residual-join readability for this displayed frontier. It exports no global confluence theorem, strong-normalization theorem, unrestricted conversion decision, subject-reduction theorem, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the existing hub route order. The critical-pair envelope is introduced before residual-substitution dependency, so any consumer that reaches a residual join has already read the bounded overlap data and the retained premise row.

Next read the candidate-normalization scope and finite residual witness files. Their rows keep the candidate join inside the displayed conditional spine and require the residual witness and bounded checker data before the local join can be consumed.

Finally apply residual-join locality and its grid, scope, exhaustion, obstruction-budget, and witness-locality refinements. Those child files identify $P, K, J, R, S, C, B, O, H, T, G, N$ as the complete local surface and keep the obstruction row visible. Since no child in this route exports a global confluence, normalization, conversion, subject-reduction, evaluator, quotient, substitution-discharge, or consistency coordinate, the handoff remains a carrier-local budget. □

Theorem 3258.118 (MetaCIC parallel-diamond frontier residual budget handoff)**.** *Every candidate-normalization or confluence-audit consumer of the local residual-join budget must keep the critical-pair envelope, residual-substitution dependency, obstruction row, and bounded normalization window visible before reading the candidate-mediated diamond. The handoff surface is the same displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the residual-substitution premise, K carrying the critical-pair envelope, O retaining the refused endpoint row, and B, S, C carrying the bounded checker and normalization window.

The handoff exports only this local residual budget. It does not claim global strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Start with the residual-join locality budget of Definition 4238.117. It already requires the critical-pair envelope, residual-substitution dependency row, finite residual witness, obstruction row, and bounded checker rows to remain visible before the residual join can be consumed.

Next use premise retention, Definition 4238.13. The retained premise row P stays attached to the candidate-mediated handoff, so a normalization-facing consumer cannot silently replace residual substitution by a discharged global theorem.

Finally apply the critical-path retained-premise handoff, Definition 4238.85. It keeps P and O visible when the route is read by confluence-audit and normalization consumers. The bounded window rows B, S, C therefore remain part of the exported packet, and the refused global endpoints remain outside the handoff. □

Theorem 3258.119 (MetaCIC parallel-diamond residual-join locality consumption)**.** *Any candidate-mediated parallel-diamond consumer of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must read the critical-pair envelope, residual-substitution dependency, residual-join locality rows, candidate normalization scope, and finite residual witness ledger before consuming the frontier* $\mathsf{NameCert}$ *row. The consumed route is the displayed carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P retains the residual-substitution premise, K, J, R expose the critical-pair and candidate rows, S, C, B, O expose the typed window, residual witness, bounded checker, and obstruction rows, and H, T, G, N provide transport, replay, provenance, and local naming. The consumer receives only this finite residual-locality budget, not global confluence, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.

Proof. First read the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before any candidate-mediated join can be consumed.

Next read the residual-substitution dependency route of Definition 4238.57. This keeps the retained premise row P inside the same finite packet while the candidate join and residual frontier rows are opened.

Now consume the residual-join locality rows of Definition 4238.104 and the budget of Definition 4238.117. These identify $P, K, J, R, S, C, B, O, H, T, G, N$ as the carrier-local residual surface and keep the obstruction row visible.

Finally attach the candidate-normalization scope of Definition 4238.62 and finite residual witness consumption from Definition 4238.69. The candidate row and bounded witness ledger are therefore read before the frontier NameCert row, so the exported handoff remains the displayed residual-locality budget rather than a global confluence, subject-reduction, evaluator-equality, quotient-equality, substitution-discharge, or consistency theorem. □

Theorem 3258.120 (MetaCIC residual-join candidate seed-status route)**.** *Every seed-status reader for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *that consumes residual-join evidence and candidate-normalization evidence must read them inside the same parallel-diamond packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route uses the residual-join locality rows, candidate-normalization scope rows, and parallel-diamond seed-status rows as seedClosure evidence for the hub. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted frontier packet. The residual-join rows J, R, S are exposed only with the retained premise row P, critical-pair row K, bounded checker row B, obstruction row O, and structural rows H, T, G, N still visible.

Read the candidate-normalization scope next. It keeps the candidate row inside the same spine $P, K, J, R, S, C, B, O, H, T, G, N$ and prevents the candidate read from becoming a premise-free normalization or confluence theorem.

Finally apply the parallel-diamond seed-status route. The seed status exported to the hub is exactly the displayed packet assembled from residual-join locality, candidate normalization, finite residual witnesses, transport, replay, provenance, and local naming, so the hub records seedClosure evidence without adding theorem content of its own. □

Theorem 3258.121 (MetaCIC parallel-diamond residual-choice boundary)**.** *Every candidate-mediated parallel-diamond read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet consumes residual choice only as a displayed finite ledger row on the route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The residual-choice row is read after the critical-pair envelope and before the residual-join handoff: the critical-pair rows K, J and residual-substitution premise P expose the finite envelope, the residual ledger keeps the chosen residual windows visible, and the residual join is consumed only through

the same bounded frontier. The boundary does not provide host choice, global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes candidate rows and residual-substitution data as part of one finite frontier route before any parallel-diamond request may be consumed.

Next use the residual-substitution dependency route of Definition 4238.57. That route keeps the retained premise P visible before the critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows.

Apply finite residual witness consumption from Definition 4238.69. The local residual witness is consumed through $P, K, J, R, S, C, B, O, H, T, G, N$, so the residual-choice row is a displayed finite ledger entry rather than an external selector.

Finally attach the retained-premise boundary of Definition 4238.158. It keeps the premise, candidate-normalization scope, residual-substitution route, finite residual witness, transport, replay, provenance, and local naming visible before the residual join. Since no cited row stores host choice, global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the residual-choice boundary. □

Definition 3258.122 (MetaCIC finite residual-diamond ledger carrier)**.** *A finite residual-diamond ledger for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is a carrier-local subpacket*

$$\Lambda = (K, P, J, B, O, H, T, G, N)$$

read from an accepted frontier packet $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *with the following displayed rows:*

1. *K is the critical-pair source row naming the bounded local peak under inspection.*
2. *P is the retained residual-substitution premise row attached to that source.*
3. *J is the candidate join witness row, read only as a proposed local join for the displayed peak.*
4. *B is the bounded checker status row for the finite residual diamond request.*
5. *O is the obstruction-retention row recording any still-visible blocker to an unconditional handoff.*
6. *H records componentwise* hsame *transport for the ledger rows.*
7. *T records displayed* Cont *replay for the same finite residual request.*
8. *G is the* Pkg *provenance row for the frontier packet and its sibling confluence consumers.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row naming the ledger as carrier-local data.*

The ledger stores no premise-free confluence theorem, no subject-reduction theorem, no evaluator equality, no quotient term equality, and no kernel-consistency certificate.

Theorem 3258.123 (MetaCIC finite residual-diamond ledger local join obligations)**.** *Every local residual-diamond read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet factors through a finite ledger* $\Lambda = (K, P, J, B, O, H, T, G, N)$ *of Definition 4238.122. The read may cite the ledger as bounded local join evidence under the retained residual-substitution premise and visible obstruction row, but it cannot be cited as an unconditional parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It supplies the bounded source row K and keeps the peak inside the finite frontier packet.

Next read the residual-substitution dependency route of Definition 4238.57. This places the retained premise row P before the candidate join can be consumed, so the ledger cannot erase the premise.

Now use the candidate diamond budget route of Definition 4238.66 together with residual-join locality, Definition 4238.104. These rows expose the candidate join J and bounded checker status B only as a local finite route through the accepted frontier.

Finally apply the retained-premise closure budget of Definition 4238.150. The obstruction row O and structural rows H, T, G, N remain displayed, so the resulting ledger is a bounded residual-diamond obligation surface rather than any of the refused global confluence or kernel endpoints. □

Theorem 3258.124 (MetaCIC frontier seed ledger)**.** *An accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier packet exposes a finite seed ledger*

$$\Sigma = (K, J, R, S, B, O, H, T, G, N)$$

before any residual-join or parallel-diamond read becomes public. The ledger reads the critical-pair row K, candidate join row J, residual-frontier row R, seed-readiness row S, bounded checker row B, obstruction row O, hsame *transport row H,* Cont *replay row T,* Pkg *provenance row G, and local* NameCert *row N through the existing MetaCIC frontier rows. It is only a seed ledger for finite residual-join inspection, not a premise-free confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Start with the seed package register of Definition 4238.82 and the seed readiness row of Definition 4238.83. These rows expose K, J, R, S, B, O as finite frontier data for a bounded residual-join request.

Next use the residual join seed interface of Definition 4238.130. It keeps the candidate join and residual frontier inside the same seed packet rather than turning them into an unconditional parallel diamond.

Finally apply the structural rows supplied by Definition 4238.103. The rows H, T, G, N transport, replay, package, and name the same finite seed ledger, so public consumers receive only the displayed residual-join ledger and none of the refused global endpoints. □

Theorem 3258.125 (MetaCIC parallel-diamond frontier residual scope)**.** *Every residual-scope consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The consumer may read residual-join locality, the residual grid, locality scope, and the obstruction budget, but it may not discharge residual substitution, subject reduction, global confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality, Definition 4238.104. It forces every residual-join read through the displayed frontier rows and keeps the packet local to the accepted carrier.

Refine the consumer route through the residual grid and locality scope, Definition 4238.107 and Definition 4238.108. These theorems separate the residual-facing rows P, K, J, R, S, C, B, O from the structural rows H, T, G, N while keeping both groups inside one finite packet.

Finally apply the residual obstruction budget, Definition 4238.111. It keeps the retained premise and obstruction rows visible, so a residual-scope consumer has no route to discharge residual substitution or to promote the local frontier to subject reduction, global confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.126 (MetaCIC parallel-diamond residual-normalization window)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet whose displayed rows expose*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

may hand a candidate-mediated residual peak to a normalization consumer only through the retained residual-substitution premise row P, the candidate-normalization row J, the residual-frontier row R, the closed-checker and closed-normal rows S, C, and the obstruction row O, with the structural rows H, T, G, N still visible. The window does not prove strong normalization, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. Its public rows expose the residual-substitution premise, critical-pair data, candidate-normalization row, residual-frontier row, closed-checker row, closed-normal row, bounded-check row, obstruction row, transport, replay, provenance, and local naming.

Use the candidate-normalization scope of Definition 4238.62. It forces a normalization-facing read of J to keep P and O visible and to remain inside the same finite spine.

Next consume the finite residual witness and candidate-mediated strong-normalization handoff, Definition 4238.69 and Definition 4238.73. These rows allow a residual peak to be presented to the consumer only after R, S, C and the bounded obstruction data have been displayed.

Finally apply residual-join locality, Definition 4238.104. Since the obstruction row remains retained and no carrier coordinate discharges P or promotes J to a global theorem, the exported object is only the stated residual-normalization window. □

Theorem 3258.127 (MetaCIC parallel-diamond seed boundary)**.** *Every seed-level read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may expose candidate-mediated residual peaks only while the residual-substitution premise row P, candidate-normalization row J, obstruction row O, closed-checker rows S, C, B, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N remain visible in the same finite carrier. The boundary is a local seed surface for residual normalization; it is not strong normalization, unrestricted confluence, quotient term equality, evaluator equality, subject reduction, residual-substitution discharge, or kernel consistency.*

Proof. Apply the residual-normalization window theorem, Definition 4238.126. It already places every normalization-facing residual peak behind the retained residual-substitution premise, candidate-normalization row, residual-frontier row, closed checker and closed-normal rows, obstruction row, and structural rows.

Next use the candidate-normalization scope of Definition 4238.62 and the candidate-mediated strong-normalization handoff of Definition 4238.73. These prevent the candidate row from becoming a premise-free normalization theorem and keep the candidate-mediated route inside the displayed frontier carrier.

Finally read the residual-substitution dependency route and obstruction ledgers. The premise row P and obstruction row O remain visible through hsame transport, Cont replay, Pkg provenance, and the local NameCert row. Since no listed row discharges those boundaries, the exposed seed remains only the stated local residual-normalization surface. □

Theorem 3258.128 (MetaCIC parallel-diamond residual-normalization scope)**.** *Any residual-join read from an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must consume the critical-pair envelope, candidate-normalization row, residual-substitution row, finite residual witness, and candidate-SN handoff before it exports a diamond-frontier budget. The exported budget is the finite route*

through

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with the residual-substitution premise and obstruction rows still visible; it is not a premise-free parallel diamond, full normalization theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. First read the critical-pair envelope of Definition 4238.50. It fixes the local pair inside the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the bounded overlap surface visible.

Next consume the candidate-normalization scope of Definition 4238.62 together with the residual-substitution dependency route of Definition 4238.57. These rows place J under the retained premise P rather than turning candidate normalization into an unconditional confluence result.

Now apply finite residual witness consumption, Definition 4238.69. The finite witness must read the typed window, residual frontier, substitution-boundary row, checker rows, transport, replay, provenance, and local naming before a closed parallel-diamond route can inspect it.

Finally use the candidate-SN handoff of Definition 4238.73. The handoff packages the same conditional spine as a diamond-frontier budget, so the exported object still retains P and O and cannot be reread as full normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.129 (MetaCIC residual-substitution discharge boundary)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. A residual-normalization read from F may discharge only the local residual-substitution boundary rows already displayed by the packet: the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness, and candidate-SN handoff. The discharge does not export subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the critical-pair envelope of Definition 4238.50. It keeps the local overlap inside the finite rows $P, K, J, R, S, C, B, O, H, T, G, N$ and prevents the read from erasing the retained residual-substitution premise.

Next combine the candidate-normalization scope of Definition 4238.62 with the residual-substitution dependency route of Definition 4238.57. These two rows allow the candidate read to discharge only the displayed local boundary, with P still visible as the premise row for the residual route.

Now consume the finite residual witness from Definition 4238.69. The witness is finite data in the same packet, so it can certify the local residual-normalization handoff without becoming a global equality or subject-reduction theorem.

Finally apply the candidate-SN handoff of Definition 4238.73. That handoff packages the preceding rows as a candidate-mediated boundary discharge, and none of the six refused exports appears as a coordinate of the carrier. □

Theorem 3258.130 (MetaCIC residual-join seed interface)**.** *Every seed-level diamond frontier consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that reads residual joins must first display the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness, and candidate strong-normalization handoff. Concretely, the read must pass through Definition 4238.50, Definition 4238.62, Definition 4238.57, Definition 4238.69, and Definition 4238.73 before the packet is available as a seed interface. The exported seed read is therefore a displayed residual-join interface over the finite frontier spine, not a premise-free confluence theorem, global normalization theorem, subject-reduction theorem, host reduction equality, or kernel-consistency certificate.*

Proof. Start with the critical-pair envelope. The envelope theorem keeps the rows $P, K, J, R, S, C, B, O, H, T, G, N$ visible before a parallel-diamond request can be read, so the seed consumer receives a bounded frontier packet rather than an unrestricted confluence surface.

Next read the candidate-normalization and residual-substitution routes. The candidate-normalization scope keeps J inside the conditional spine, while the residual-substitution dependency route keeps the retained premise and residual windows attached to the same packet.

Now expose the finite residual witness and the candidate strong-normalization handoff. The finite witness theorem supplies the bounded residual rows consumed by the frontier, and the candidate handoff states the normalization-facing read available from those same rows.

Thus any seed-level residual-join consumer has displayed the five interface faces before reading the packet. Since none of those faces exports premise-free confluence, global normalization, subject reduction, host reduction equality, or kernel consistency, the seed interface remains exactly the finite residual-join route stated above. □

Theorem 3258.131 (MetaCIC residual-join seed scope)**.** *Every residual-join seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet factors through residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed-checker rows, obstruction rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The seed read is therefore a finite residual-join scope for the parallel-diamond frontier, not a subject-reduction theorem, arbitrary Church–Rosser theorem, quotient term equality, or evaluator equality row.*

Proof. Start with the residual-substitution dependency route and the critical-pair envelope. The dependency route keeps the retained substitution premise visible, while Definition 4238.50 exposes the critical-pair rows before any diamond request is read.

Next read the candidate diamond budget route. It displays the candidate-join rows and keeps them tied to the same finite frontier rather than to an unrestricted Church–Rosser surface.

Now use the finite residual witness consumption route. The residual-frontier and closed-checker rows are consumed only after the finite witness rows have been displayed, so the obstruction rows remain part of the same seed scope.

The structural rows supply componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Since the cited routes contain no subject-reduction, arbitrary Church–Rosser, quotient term equality, or evaluator equality coordinate, the residual-join seed read exports none of those endpoints. □

Theorem 3258.132 (MetaCIC residual-join seed-status record)**.** *The residual-join seed status of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is available to a confluence-facing consumer only after the consumer has read the critical-pair envelope, candidate-normalization scope, finite residual witness, and retained-premise rows of the displayed frontier packet. Equivalently, the route passes through Definition 4238.50, Definition 4238.62, Definition 4238.69, and Definition 4238.159 before any confluence consumer may inspect the residual-join seed record. The record exports only this bounded seed route; it does not supply premise-free confluence, global normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the critical-pair envelope first. It keeps the frontier rows visible before any residual-join request is interpreted as a seed-status claim.

Next apply the candidate-normalization scope and finite residual witness. These rows confine the candidate read and the residual read to the same finite frontier packet.

Finally read the retained-premise residual-join scope. Since that row remains present until the seed record is consumed, the exported status is a bounded residual-join route rather than a detached confluence theorem. □

Theorem 3258.133 (Residual-join candidate dependency boundary)**.** *Every residual-join candidate read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness consumption, and retained-premise boundary before it can be consumed as a local diamond frontier. In carrier form the read remains inside*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K *displays the critical-pair envelope,* P *retains the residual-substitution dependency,* J *records the candidate-normalization row,* R, S, C, B, O *keep the residual frontier, typed window, witness, bounded checker, and obstruction rows visible, and* H, T, G, N *provide transport, replay, provenance, and local naming. The boundary exports only this local residual-join candidate surface; it explicitly refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before a residual-join candidate can be read by a frontier consumer.

Next read the residual-substitution dependency route of Definition 4238.57. This keeps the retained premise row P visible while the candidate, residual, checker, and obstruction rows are opened.

Now apply candidate-normalization scope through Definition 4238.62. The candidate row J remains part of the displayed frontier packet rather than a detached normalization theorem.

Consume the finite residual witness with Definition 4238.69. The residual rows R, S, C, B, O remain tied to the same finite witness route and cannot be reread as a global local-confluence theorem.

Finally use the retained-premise boundary of Definition 4238.158. It keeps $P, K, J, R, S, C, B, O, H, T, G, N$ visible before any local diamond frontier consumes the candidate, so the refused endpoints remain outside the packet. □

Theorem 3258.134 (MetaCIC candidate-mediated residual-join scope)**.** *Every candidate-mediated residual join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual join locality rows before it can be read as a parallel-diamond frontier packet. In carrier form the read remains on the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P *retains the residual-substitution premise,* K *exposes the critical-pair envelope,* J *records the candidate-normalization row,* R, S, C, B, O *keep the residual frontier, seed, witness, bounded checker, and obstruction rows visible, and* H, T, G, N *provide transport, replay, provenance, and local naming. The scope exports only this finite candidate-mediated handoff, not global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before any residual join is available to a frontier consumer.

Next read candidate-normalization scope through Definition 4238.62. The candidate row J is therefore part of the displayed packet rather than an external normalization theorem.

Now use the residual-substitution dependency route of Definition 4238.57. It keeps the retained premise row P load-bearing while the residual rows R, S, C, B, O are opened.

Finally consume finite residual witness data and residual join locality via Definition 4238.69 and Definition 4238.104. These rows keep the local join inside $P, K, J, R, S, C, B, O, H, T, G, N$, so the resulting frontier packet provides only the stated candidate-mediated residual-join scope and none of the refused global endpoints. □

Theorem 3258.135 (MetaCIC candidate-mediated residual window)**.** *Every accepted residual join read from a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the retained premise row P, candidate-normalization row J, residual-substitution route, finite residual witness rows, and residual-join locality rows inside the same bounded window before any parallel-diamond conclusion is visible. In carrier form the window is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P remains the retained residual-substitution premise, J remains the candidate-normalization row, R, S, C, B, O carry the residual frontier, typed window, finite witness, bounded checker, and obstruction data, and H, T, G, N transport, replay, source, and name the same packet. The read is therefore premise-bearing local residual evidence, not a premise-free parallel-diamond conclusion, unrestricted confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the candidate-mediated residual-join scope of Definition 4238.134. It already requires the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual-join locality rows to be displayed before a frontier consumer reads the handoff.

Next apply the residual-substitution dependency route of Definition 4238.57. This keeps P load-bearing while the residual rows are opened, so the bounded window cannot be reread as a premise-free substitution result.

Finally combine finite residual witness consumption with residual-join locality, using Definition 4238.69 and Definition 4238.104. These rows keep R, S, C, B, O attached to the same candidate-normalization route through J and the same structural rows H, T, G, N. Thus any visible parallel-diamond conclusion remains inside the retained bounded window named in the statement. □

Theorem 3258.136 (MetaCIC candidate-mediated residual-join obligation)**.** *Every accepted candidate-normalization window of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that exposes both residual branches must also expose the finite local-join row, retained-premise ledger, and obstruction-retention row before a parallel-diamond frontier consumer may read the window. In carrier form the admissible obligation surface is*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K is the critical-pair envelope, J is the candidate-local join row, P is the retained residual-substitution premise ledger, R, S, C, B display the residual branch, typed-window, witness, and bounded-checker rows, O records obstruction retention, and H, T, G, N preserve transport, replay, provenance, and local naming. The obligation is local to this finite surface; it does not assert global confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, strong normalization, or kernel consistency.

Proof. Project the candidate-mediated residual window of Definition 4238.135. It already keeps $P, J, R, S, C, B, O, H, T, G, N$ visible while the candidate-normalization row is consumed.

Read the critical-pair envelope through Definition 4238.50. Both residual branches are therefore branches of the displayed finite envelope K, not independent reductions imported from an ambient calculus.

Apply residual-window scope using Definition 4238.134. The local join row J, retained premise row P, residual rows R, S, C, B, and obstruction row O remain on the same candidate-mediated route.

If a branch attempted to bypass J or O, it would no longer be a read of the retained window $P, K, J, R, S, C, B, O, H, T, G, N$. If it attempted to bypass P, the residual-substitution dependency named by the window would be lost. Hence the frontier consumer receives a parallel-diamond-facing obligation only after the local join ledger, retained premise ledger, and obstruction-retention row have all been exposed. □

Theorem 3258.137 (MetaCIC parallel-diamond window-budget scope)**.** *Every accepted* MetaCICParallelDiamondF *frontier read must expose the typed premise row, critical-peak row, bounded join-budget row, residual checker row, candidate strong-normalization handoff row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any local-join conclusion is public. In the carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P is the retained typed premise, K is the peak row, J is the candidate join row, R, S, C, B are the finite residual and bounded checker budget rows, O is the visible obstruction row, and H, T, G, N are the structural transport, replay, provenance, and naming rows. The scope exports only this finite window-budget surface, not global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.

Proof. Start with the critical-pair envelope. It displays the peak and local candidate rows before a diamond request can be inspected by a frontier consumer.

Next read the residual-substitution dependency route and residual-normalization scope. These keep the typed premise row P and obstruction row O visible while the residual checker and bounded join-budget rows are opened.

Use the candidate strong-normalization handoff together with finite residual witness consumption. The handoff can be consumed only after the candidate row, residual witness rows, bounded checker rows, transport, replay, provenance, and local naming rows have been displayed in the same carrier-local spine.

Finally apply residual-row visibility. Since H, T, G, N preserve the displayed finite packet and no carrier coordinate contains global confluence, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency, the exported conclusion is exactly the local finite window-budget scope. □

Theorem 3258.138 (MetaCIC parallel-diamond residual-join obligations)**.** *Every frontier residual join exposed by an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must display the critical-pair source row, retained residual-substitution window, candidate-normalization budget, local join witness, obstruction-retention row, componentwise* hsame *transport, displayed* Cont *replay, and* NameCert *provenance before a consumer may read the join. Equivalently, the obligation surface is the carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P is still the residual-substitution premise, K, J, R give the critical-pair and candidate frontier rows, S, C, B, O give the bounded checker, closed-normal, budget, and obstruction rows, and H, T, G, N give transport, replay, provenance, and local naming.

The surface is a bounded candidate-mediated residual-join obligation. It is not a parameter-transport equivalence theorem, global confluence theorem, unbounded normalization theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. First read residual-join locality, Definition 4238.104. It places every accepted residual join inside the finite frontier spine and keeps the critical-pair, retained premise, local join, bounded checker, obstruction, transport, replay, provenance, and naming rows visible.

Next apply the residual-scope route of Definition 4238.125. This prevents the retained residual-substitution premise from being discharged after the local join is consumed, and it keeps $P, K, J, R, S, C, B, O, H, T,$ in the same carrier-local packet.

Now use the residual-normalization window and candidate-mediated strong-normalization handoff, Definition 4238.126 and Definition 4238.73. These rows expose the candidate-normalization budget and local witness only after the typed residual window, bounded checker rows, and obstruction row have been displayed.

Finally repack the displayed rows as the residual-join obligation surface. Since the packet contains no coordinate for parameter-transport equivalence, global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency, the exposed join remains a local obligation over the stated finite frontier spine. □

Theorem 3258.139 (MetaCIC residual-join obligation handoff)**.** *Let an accepted* MetaCICParallelDiamondFrontier[1] *packet expose finite critical-pair coverage, residual-substitution locality, and retained-premise closure through the carrier-local rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If a residual-join consumer reads the finite critical-pair surface through K, J, R, keeps the residual-substitution premise P local to the same packet, and preserves the retained premise and obstruction rows through B, O, H, T, G, N, then the packet provides the candidate-mediated strong-normalization handoff surface recorded by Definition 4238.73. The handoff is a bounded residual-join obligation surface; it does not claim global confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join obligations and residual critical-pair coverage, Definition 4238.138 and Definition 4238.140. These results force the consumer to expose the finite critical-pair, candidate, residual witness, checker, obstruction, transport, replay, provenance, and naming rows before any local join is exported.

Next use residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. They keep the residual-substitution premise inside the carrier-local packet and prevent the candidate join from being read as an independent confluence theorem.

Finally apply the retained-premise budget scope and the candidate-mediated SN handoff, Definition 4238.142 and Definition 4238.73. The first keeps P and O visible through the structural rows, and the second packages exactly that displayed finite surface as the candidate-mediated handoff. Since every cited step is still confined to $P, K, J, R, S, C, B, O, H, T, G, N$, no global confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel-consistency endpoint is exported. □

Theorem 3258.140 (MetaCIC parallel-diamond residual critical-pair coverage)**.** *Every residual-join consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose, before exporting its local join, the critical-pair envelope, retained residual-substitution premise, candidate-normalization scope, finite residual witness, bounded checker rows, obstruction row, and local* NameCert *row. Equivalently, the exported local join must still be read through the finite carrier-local surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the substitution premise, K exposing the critical-pair row, J and R retaining the candidate-normalization and residual-witness rows, S, C, B exposing the bounded checker rows, O retaining the obstruction row, and H, T, G, N carrying transport, replay, provenance, and local naming.

The coverage claim is local to the frontier packet. It exports no global confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Begin with the residual-join obligations and residual-normalization envelope above. Together they keep any exported local join behind the retained residual-substitution premise, critical-pair row, candidate-normalization budget, bounded checker rows, obstruction row, transport, replay, provenance, and naming.

Next refine the obligation surface through residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These theorems identify the critical-pair, residual-substitution, candidate-normalization, finite-witness, bounded-checker, obstruction, and local naming rows as one carrier-local grid rather than independent side conditions.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. It lets the consumer use the candidate-mediated boundary only after the finite residual witness and bounded checker rows remain visible, so the exported join is covered by the displayed critical-pair surface and cannot be reread as global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.141 (MetaCIC parallel-diamond residual-normalization envelope)**.** *Any consumer that uses the local residual join of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first expose the retained residual-substitution premise, the residual-normalization window, the candidate strong-normalization handoff, bounded checker rows, the obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The envelope is the same carrier-local route

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read with P retained and O visible before the candidate-mediated normalization handoff is consumed. It exports no full confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the residual-join obligation surface above. It keeps the critical-pair source, retained residual-substitution premise, candidate-normalization budget, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before a local join can be read.

Next apply the residual-normalization window, Definition 4238.126. That row forces a normalization-facing residual peak to remain behind $P, J, R, S, C, O, H, T, G, N$ and prevents the candidate row from becoming a premise-free normalization theorem.

Finally use the candidate strong-normalization handoff, Definition 4238.73. It lets the consumer pass the bounded candidate data onward only after the checker rows and obstruction row are still displayed, so the resulting envelope remains a local residual-normalization boundary rather than full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.142 (MetaCIC parallel-diamond residual-premise budget scope)**.** *Every candidate-mediated strong-normalization or local residual-join consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the retained residual-substitution premise row P and obstruction row O visible through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before exporting a budgeted join. The budget is the displayed carrier-local budget over residual premise, critical pair, candidate row, residual witness, closed checker, closed-normal fragment, bounded checker, obstruction, transport, replay, provenance, and local naming; it is not a discharge of residual substitution, global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the residual-join obligations and critical-pair coverage, Definition 4238.138 and Definition 4238.140. They require every exported local join to display $P, K, J, R, S, C, B, O, H, T, G, N$, with the residual-substitution premise and obstruction rows still present.

Next apply the residual-normalization envelope, Definition 4238.141. It keeps candidate-mediated normalization behind the same finite route and prevents the candidate row from becoming a premise-free normalization theorem.

Finally use the candidate strong-normalization handoff, Definition 4238.73. The handoff can export only the displayed budgeted join, so any consumer claiming global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency has left the retained residual-premise surface. □

Theorem 3258.143 (MetaCIC parallel-diamond residual-join transport induction)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet display the residual critical-pair rows, residual-normalization envelope, retained-premise budget rows, componentwise* hsame *transport, and displayed* Cont *replay over the finite spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If those rows are transported componentwise and replayed by the displayed continuation data, the resulting local join remains inside the same candidate-mediated diamond surface. In particular, residual critical-pair coverage, residual-normalization envelope data, and retained-premise budget scope are preserved as frontier-local rows, not reread as global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Induct through the finite residual row list. The base row is the retained residual-substitution premise P; transport along the corresponding component of H keeps the premise visible before any critical-pair or candidate row is consumed.

For the critical-pair and local-join segment, use Definition 4238.138 and Definition 4238.140. These theorems keep K, J, R attached to the same carrier-local residual surface under replay by T.

For the normalization and budget segment, apply Definition 4238.141 and Definition 4238.142. The rows S, C, B, O remain visible before the candidate-mediated handoff can export a budgeted join.

The structural rows H, T, G, N then repack the transported list as the same local diamond surface. Since each induction step uses only displayed rows and continuation replay, no quotient term equality, global confluence theorem, unbounded normalization theorem, subject-reduction theorem, evaluator equality, or kernel-consistency certificate is introduced. □

Theorem 3258.144 (MetaCIC retained-premise discharge boundary)**.** *Every candidate-mediated local join exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the retained residual-substitution premise row P through the full visible route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The candidate rows K, J, R, S, C, B may transport a bounded local join, and the structural rows H, T, G, N may replay and name it, but none of those rows discharges P. The boundary therefore exports only a local residual-join read with P retained; it refuses global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, and kernel consistency.

Proof. Start with residual-join transport induction, Definition 4238.143. Its base row is P, so every transported local join begins with the retained residual-substitution premise still visible.

For the candidate-mediated segment, use residual critical-pair coverage and the residual-normalization envelope. These results keep K, J, R, S, C, B attached to the same finite frontier packet and prevent the candidate handoff from becoming a premise-free join theorem.

Finally apply retained-premise budget scope and the candidate strong-normalization handoff. They permit a bounded local join only while P, O, H, T, G, N remain in the route. Hence the exported join preserves the retained premise and reaches none of the refused global endpoints. □

Theorem 3258.145 (MetaCIC obstruction-preservation route)**.** *Every candidate-mediated local join transported through an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet preserves the obstruction row O along the same finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The obstruction remains visible before and after the local join is replayed by T and named by N. Thus the packet records obstruction-preserving local residual-join transport only; it does not assert global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First read the residual-normalization envelope and residual-premise budget scope. Together they place O beside the bounded checker and candidate rows, so the obstruction is not an external side condition that can be erased after the join is found.

Next use residual-join transport induction. The transport row H and replay row T act componentwise on the visible route, so the obstruction component is carried with P, K, J, R, S, C, B rather than replaced by evaluator equality or quotient term equality.

Finally repack through G and N. Provenance and local naming certify the same obstruction-preserving frontier packet, and because no coordinate provides unrestricted normalization, subject reduction, global confluence, or kernel consistency, the route remains exactly the local obstruction-preservation route. □

Theorem 3258.146 (MetaCIC residual-budget diamond obligation)**.** *Every local diamond-join read exposed by an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must display the critical-pair envelope, retained residual-substitution window, candidate-normalization scope, finite residual-diamond ledger, and residual-join locality rows before the frontier package can expose a candidate-mediated confluence handoff. Equivalently, the handoff remains inside the carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P retains the residual-substitution premise, K records the critical-pair envelope, J, R, S, C, B record the candidate and bounded residual window, O records the visible obstruction, and H, T, G, N record transport, replay, provenance, and local naming. The obligation exports only a bounded local diamond route; it does not discharge residual substitution, prove unrestricted confluence, prove subject reduction, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Start with the residual-substitution dependency route, Definition 4238.57. It places the retained premise row P before any critical-pair or candidate-join row can be consumed.

Next read the candidate-normalization scope of Definition 4238.62. That scope keeps J inside the conditional spine and requires the obstruction row O to remain visible when a normalization-facing consumer inspects the candidate.

Use the finite residual-diamond ledger obligations, Definition 4238.123. The ledger packages the critical-pair row, retained premise, candidate join, bounded checker status, obstruction, transport, replay, provenance, and local naming as finite carrier-local data.

Finally apply residual-join locality and its locality budget, Definition 4238.104 and Definition 4238.105. These rows keep the displayed packet on the route $P, K, J, R, S, C, B, O, H, T, G, N$ before any candidate-mediated confluence handoff is public. Since every cited surface is carrier-local and still retains P and O, the handoff remains the stated residual-budget obligation rather than any refused global endpoint. □

Theorem 3258.147 (MetaCIC parallel diamond frontier residual-join obligation carrier)**.** *Before any candidate diamond read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is public, the residual-join obligation carrier consists of the retained premise row, critical-pair envelope, residual-substitution dependency row, finite residual witness, obstruction row, bounded checker row, transport, replay, provenance, and local* NameCert *row. In carrier notation the public residual-join obligation must still expose*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the premise, K exposing the critical pair, J, R carrying the candidate and finite residual witness, S, C, B retaining the residual-substitution and checker rows, O retaining the obstruction, and H, T, G, N transporting, replaying, sourcing, and naming the same packet. The carrier is an obligation surface for a bounded residual join; it is not full confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. Start with the residual-budget diamond obligation of Definition 4238.146. It already forces every local diamond-join read to expose the critical-pair envelope, retained residual-substitution window, candidate-normalization scope, finite residual-diamond ledger, and residual-join locality rows before a candidate-mediated handoff is public.

Next use finite residual witness consumption, Definition 4238.69. This supplies the finite residual witness, typed window, bounded checker, obstruction, transport, replay, provenance, and local naming rows as displayed data of the same packet.

Finally apply the residual-join obligation handoff and candidate-mediated SN handoff, Definition 4238.139 and Definition 4238.73. They keep P and O visible through H, T, G, N before the candidate-facing read is exported. Since the composed route never leaves $P, K, J, R, S, C, B, O, H, T, G, N$, it yields only the stated residual-join obligation carrier and none of the refused global endpoints. □

Theorem 3258.148 (MetaCIC residual-join frontier obligation)**.** *Every candidate-mediated residual join exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier must keep the retained residual-substitution premise row P and the obstruction row O visible through the same finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before any local diamond handoff is consumed. Thus the consumer may read a bounded local join only while the critical-pair, candidate, finite residual witness, bounded checker, obstruction, transport, replay, provenance, and local naming rows remain carrier-local.

The obligation does not export global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First compose the residual-join obligation carrier of Definition 4238.147 with the residual-budget diamond obligation of Definition 4238.146. The carrier supplies the displayed route, and the budget obligation prevents the candidate diamond read from leaving that route before the handoff is visible.

Next use finite residual witness consumption, Definition 4238.69. This step keeps the residual witness, typed window, bounded checker rows, and obstruction row as displayed rows of the same packet rather than private side conditions.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. Since the handoff consumes only the displayed candidate-mediated surface and the rows P and O are still present through H, T, G, N, the exported join remains a local frontier obligation and none of the refused global endpoints is introduced. □

Theorem 3258.149 (MetaCIC residual-join window determinacy)**.** *Let two accepted candidate-mediated residual-join reads of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the same retained premise row P, critical-pair row K, candidate row J, residual witness rows R, S, C, B, obstruction row O, transport rows H, T, provenance row G, and local* $\mathsf{NameCert}$ *row N. Then the two reads determine the same local join window over the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The conclusion is only local window determinacy for the displayed residual-join surface. It is not global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. First compose the residual-join obligation carrier of Definition 4238.147. Both reads expose the same carrier-local rows, so the retained premise, critical-pair row, candidate row, residual witness rows, obstruction row, transport, replay, provenance, and local naming are the same displayed packet.

Next apply the residual-budget diamond obligation of Definition 4238.146. The budgeted local diamond handoff cannot leave the route $P, K, J, R, S, C, B, O, H, T, G, N$, and the shared rows therefore force both reads to consume the same bounded residual window.

Use finite residual witness consumption to keep R, S, C, B as the displayed witness and checker rows of that same window. The shared obstruction row O and shared transport rows H, T prevent either read from replacing the witness by an external evaluator equality or quotient term equality.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. Since the handoff consumes only the shared candidate-mediated surface, the two accepted reads determine one local join window. The argument never exports global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.150 (MetaCIC parallel-diamond retained-premise closure)**.** *A retained-premise read used by a candidate-mediated strong-normalization or local-confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose residual-substitution compatibility rows, candidate-normalization rows, local-join rows, the critical-pair envelope, finite residual witnesses, typed-window rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row before it is consumed as a closure budget. The budget is a finite frontier read over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the critical-pair envelope and candidate-normalization scope, Definition 4238.50 and Definition 4238.62. They keep $P, K, J, R, S, C, B, O, H, T, G, N$ visible before candidate-mediated normalization can read the candidate row.

Apply residual-join locality and finite residual witness consumption, Definition 4238.104 and Definition 4238.69. These force the retained premise, typed windows, residual frontier, local-join checker, finite witness, transport, replay, provenance, and local naming rows to be consumed as one finite packet.

Finally read the retained-premise mature budget, Definition 4238.154. It allows the premise to serve as a closure budget only while the obstruction and residual-substitution rows remain displayed, so the read cannot become subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.151 (MetaCIC parallel-diamond retained-premise induction)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose a finite list of retained-premise reads to a candidate-mediated strong-normalization or local-confluence consumer. If each read is admitted by the retained-premise closure budget of Definition 4238.150, then induction on the list preserves the same displayed rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through every residual-pair read. The iterated budget remains only a finite frontier over those rows, and cannot discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. For the base list, read the retained-premise closure theorem. It supplies one closure-budget surface over $P, K, J, R, S, C, B, O, H, T, G, N$, and the empty residual-pair suffix contributes no additional semantic row.

For the cons step, expose the head retained-premise read through the same closure-budget theorem and apply the induction hypothesis to the remaining finite list. The head read and the tail budget both project the same transport, replay, provenance, obstruction, typed-window, residual-frontier, candidate, local-join, and naming rows, so their composition still displays exactly $P, K, J, R, S, C, B, O, H, T, G, N$.

Finally project the negative boundary attached to each step. Since neither the head read nor the inductive tail contains a row for subject reduction, unrestricted confluence, evaluator equality,

quotient term equality, global normalization, or kernel consistency, the finite composition cannot acquire any of those conclusions. □

Theorem 3258.152 (MetaCIC candidate-normalization retained premise)**.** *Any candidate-mediated strong-normalization or local-confluence consumer reading a retained premise from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the displayed candidate-normalization, residual-substitution, local-join, critical-pair, bounded-check, typed-window, transport, replay, provenance, obstruction, and local* NameCert *rows. The retained-premise route is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not an unrestricted confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency claim.

Proof. Apply retained-premise closure, Definition 4238.150. It keeps candidate normalization, residual substitution, local joins, critical pairs, bounded checks, typed windows, transport, replay, provenance, obstruction, and naming visible before the retained premise is consumed.

Apply retained-premise induction, Definition 4238.151. A finite list of such reads preserves the same displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ through every residual-pair step.

Now use the candidate SN residual budget and public route sync cited by the retained-pair induction, Definition 4238.166 and Definition 4238.170. These keep the consumer on the candidate-normalization route rather than replacing it by a global semantic theorem.

The carrier exposes no row for unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. Therefore the retained premise remains a finite candidate-normalization budget over the displayed rows only. □

Theorem 3258.153 (MetaCIC parallel-diamond retained-pair induction)**.** *Every finite successor chain of candidate-mediated SN/confluence handoffs that reads the public or bridged* MetaCICParallelDiamondFrontie *route preserves the retained pair* (P, O) *and the residual budget* (J, R, B) *before each handoff. The induction invariant is read through the displayed public route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

and it does not discharge residual substitution, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. For the base chain, read the public certificate and finite residual witness consumption. The public route exposes $P, K, J, R, S, C, B, O, H, T, G, N$, while the finite witness row keeps the retained pair P, O and the residual budget J, R, B visible before the first handoff.

For the successor step, apply candidate SN residual budget and public route sync, Definition 4238.166 and Definition 4238.170. The head handoff carries P, O as retained premise and obstruction data, and it carries J, R, B as candidate join, residual frontier, and bounded-check budget data. The induction hypothesis supplies the same rows for the remaining finite chain.

Finally compare with the retained seed block sync of Definition 4238.171. It identifies the retained handoff data as finite public-route evidence rather than residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency, so the whole finite chain preserves only the stated pair and residual budget. □

Theorem 3258.154 (MetaCIC parallel-diamond retained-premise mature budget)**.** *Every candidate-mediated SN/confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must carry the retained*

premise budget P, J, R, B, O through the public certificate, finite residual witness consumption, candidate SN residual budget, public frontier budget, and bridged export before reading any parallel-diamond handoff. The carried budget records the residual-substitution premise row, candidate-join row, residual-frontier row, bounded-check row, and obstruction row as finite packet data; it is not residual-substitution discharge, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, global conversion decidability, or kernel consistency.

Proof. Start with the public certificate of Definition 4238.162. It exposes the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as one finite public handoff.

Next consume the finite residual witness through Definition 4238.165. That route makes a candidate-mediated SN/confluence consumer read $P, K, J, R, S, C, B, O, H, T, G, N$ before it can inspect the frontier, so the retained rows P, J, R, B, O remain visible inside the larger finite route.

Apply the candidate SN residual budget and public frontier budget, Definition 4238.166 and Definition 4238.168. Together they isolate P, J, R, B, O as the budget that must be carried forward before any public diamond handoff is read.

Finally apply the bridged export of Definition 4238.169. The bridged read may expose the guarded frontier to an external consumer only while preserving the same retained evidence rows and the structural transport, replay, provenance, and local naming route. Therefore the mature budget cannot be reread as residual-substitution discharge, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, global conversion decidability, or kernel consistency. □

Theorem 3258.155 (MetaCIC parallel-diamond retained-premise discharge boundary)**.** *Every local diamond/frontier read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that consumes a candidate join row J and residual frontier row R must still expose the retained residual-substitution premise row P, the critical-pair row K, the closed-checker row S, the closed-normal fragment row C, the bounded-check row B, the obstruction row O, and the structural rows H, T, G, N. The route is not full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Read the local public certificate and finite residual witness routes. They expose the candidate join and residual frontier only inside the larger packet $P, K, J, R, S, C, B, O, H, T, G, N$, so consuming J and R cannot hide the premise, critical-pair, checker, normal-fragment, bounded-check, obstruction, or structural rows.

Apply the retained-premise closure and mature budgets. Those budgets require the residual-substitution premise and the obstruction-facing bounded check to remain visible through candidate-mediated diamond reads.

The listed off-table metatheorems are not rows of the packet. Since the carrier exports only the displayed finite evidence and structural transport, replay, provenance, and naming rows, the local read records a discharge boundary rather than a full residual-substitution or confluence theorem. □

Theorem 3258.156 (MetaCIC retained-premise local discharge route)**.** *A candidate-mediated diamond consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *may discharge only the local retained-premise route*

$$P,\ K,\ J,\ R,\ S,\ C,\ B,\ O,\ H,\ T,\ G,\ N.$$

This discharge is local consumption of the displayed route: the retained residual-substitution premise row P must remain present in $P, K, J, R, S, C, B, O, H, T, G, N$ while the consumer reads J and R.

The route still does not provide full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Apply Definition 4238.155. It already forces every candidate join and residual frontier read to keep the retained premise, critical-pair, checker, normal-fragment, bounded-check, obstruction, and structural rows visible.

The criterion is a visibility condition on the retained premise, not a proof that residual substitution has been eliminated.

The present route narrows that boundary to the local consumer order $P, K, J, R, S, C, B, O, H, T, G, N$. Reading J and R is therefore allowed only after the retained-premise and critical-pair rows have been exposed, and before the closed-checker, closed-normal fragment, bounded-check, obstruction, transport, replay, provenance, and naming rows close the local packet.

No further row is exported by this local discharge. The off-table metatheorems remain outside the packet, so the consumer records a retained-premise route rather than a global residual-substitution, confluence, normalization, evaluator-equality, quotient-equality, or consistency theorem. □

Theorem 3258.157 (MetaCIC retained-premise normalization bridge)**.** *Every candidate-mediated normalization or local residual-join read that uses the retained-premise discharge route of Definition 4238.156 must still factor through the finite spine*

$$P,\ K,\ J,\ R,\ S,\ C,\ B,\ O,\ H,\ T,\ G,\ N.$$

It may consume the candidate strong-normalization handoff of Definition 4238.73 and the residual-join obligation surface of Definition 4238.138 only as bounded local evidence. The bridge refuses full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, and kernel consistency.

Proof. Apply Definition 4238.156. Any consumer using that route is already confined to $P, K, J, R, S, C, B, O, H, T$, with P and O still visible before the local packet closes.

Next read the candidate strong-normalization handoff of Definition 4238.73. That handoff can be consumed only after the candidate row, bounded checker rows, obstruction row, transport, replay, provenance, and local naming remain displayed as finite evidence.

Now attach the residual-join obligation surface of Definition 4238.138. It keeps the local residual join inside the same critical-pair, candidate, residual-frontier, checker, obstruction, and structural rows rather than turning the join into an independent confluence theorem.

The carrier inventory and nonescape boundary then close the bridge. Since every cited surface stays inside the displayed spine, no consumer obtains full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 3258.158 (MetaCIC parallel-diamond retained-premise boundary)**.** *Every candidate-mediated parallel-diamond consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that reaches a residual join must first expose the retained-premise ledger, candidate-normalization scope, residual-substitution route, finite residual witness, and the supporting* hsame, Cont, Pkg, *and* NameCert *rows. In the frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the premise row P, candidate row J, residual rows K, R, S, C, B, O, transport row H, replay row T, provenance row G, and local naming row N remain visible before any residual join can be

consumed. The boundary does not provide a hidden strong-normalization oracle, host confluence primitive, detached residual equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the retained-premise discharge boundary, Definition 4238.155. It keeps $P, K, S, C, B, O, H, T, G, N$ visible whenever a local diamond or frontier read consumes the candidate join J and residual frontier R.

Next apply the candidate strong-normalization handoff, Definition 4238.73. That handoff exposes the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before a normalization-facing consumer can read the frontier.

Now use the residual-substitution dependency route, Definition 4238.57. It places the retained premise row before every critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction read, so the residual join cannot be detached from the premise boundary.

Finally attach the finite residual-diamond ledger, Definition 4238.123. The ledger records the local join only as bounded carrier-local data over $K, P, J, B, O, H, T, G, N$. Since every cited route preserves the same structural support rows, the residual join remains a retained-premise boundary rather than a hidden strong-normalization oracle, host confluence primitive, detached residual equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate. □

Theorem 3258.159 (MetaCIC parallel-diamond retained-premise scope)**.** *Every retained premise used by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *read must be displayed as one of the residual-pair, candidate-normalization, local-join, closed-checker, or obstruction rows already named in the frontier carrier*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, a retained premise can be consumed only as carrier-local evidence through those rows, and cannot be read as an unconditional closed parallel-diamond theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Project the frontier carrier of Definition 4238.1. The residual-pair and premise-retention rows are P, K, R, the candidate-normalization row is J, the local-join and closed-checker rows are S, C, B, and the obstruction row is O.

Apply the residual boundary route of Definition 4238.57. It places every residual-pair use behind the retained premise row and keeps the critical-pair and residual-frontier rows visible.

Next use the candidate-normalization and local-join boundaries, Definition 4238.62 and Definition 4238.123. These theorems allow candidate and local-join data to be read only through the displayed finite route, with the retained premise still attached.

Finally apply the retained-premise discharge boundary and obstruction ledger, Definition 4238.155 and Definition 4238.111. The closed-checker and obstruction rows remain visible through the structural rows H, T, G, N, so no consumer can reread the retained premise as an unconditional closed parallel diamond or any of the refused kernel-level endpoints. □

Theorem 3258.160 (MetaCIC parallel-diamond local retained-premise factorization)**.** *Any local diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that consumes candidate join and residual-frontier rows factors through the retained premise row, critical-pair row, candidate-normalization row, local-join row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, and structural rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The factorization is conditional on those displayed rows. It is not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with retained-premise scope, Definition 4238.159. It requires every premise used by the local diamond read to remain one of the displayed carrier rows.

Apply the residual-substitution dependency route of Definition 4238.57. The route places the retained premise row P before the critical-pair, candidate join, residual-frontier, closed-checker, and obstruction rows can be consumed.

Use candidate-normalization scope from Definition 4238.62. The candidate row J remains inside the same conditional spine rather than becoming an external normalization theorem.

Apply the finite residual-diamond ledger local-join obligations of Definition 4238.123. They read the local join, closed checker, closed-normal fragment, bounded-check, and obstruction rows only after the retained premise and residual-frontier data have been displayed.

Finally use the retained-premise discharge boundary of Definition 4238.155. Since the structural rows H, T, G, N only transport, replay, source, and name the same finite packet, the local diamond read cannot be promoted to any refused global metatheorem or kernel endpoint. □

Theorem 3258.161 (MetaCIC retained-premise residual-join scope)**.** *Every candidate-mediated residual join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet remains inside the typed local-join scope when the retained premise window is exposed before the normalization and confluence handoff rows. Equivalently, the read must keep the finite carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

visible with P as the retained premise, K as the critical-pair row, J as the candidate-normalization row, R, S, C, B as the residual witness, local-join, closed-checker, and bounded-checker rows, O as the obstruction row, and H, T, G, N as transport, replay, provenance, and local naming.

The resulting scope is a retained-premise residual-join surface. It does not export premise-free residual substitution, global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First apply retained-premise scope, Definition 4238.159. It requires every premise used by a frontier read to appear as one of the displayed residual-pair, candidate-normalization, local-join, closed-checker, obstruction, transport, replay, provenance, or naming rows.

Next use candidate-mediated residual-join scope, Definition 4238.134. The candidate row and residual rows are consumed only after the critical-pair envelope and residual-substitution dependency route have made the retained premise visible.

Now read the window-budget scope and residual-join obligation surface, Definition 4238.137 and Definition 4238.138. These theorems keep the bounded checker, local join, obstruction, transport, replay, provenance, and local naming rows on the same carrier-local route.

Therefore a candidate-mediated residual join can be exported only as the displayed retained-premise surface $P, K, J, R, S, C, B, O, H, T, G, N$. Since no cited row contains a premise-free residual-substitution discharge or any global kernel endpoint, the conclusion remains inside the typed local-join scope. □

Theorem 3258.162 (MetaCIC parallel-diamond frontier public certificate)**.** *The public* MetaCICParallelDiamondFr *certificate is the finite route through the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local*

NameCert *rows. It exposes the residual frontier as a public finite handoff while refusing residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency.*

Proof. Apply the public export route of Definition 4238.47. It displays the residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming rows before any outside consumer may read the frontier.

Apply the public seed-readiness route of Definition 4238.84. This keeps the closed-checker seal, closed-normal fragment, bounded-check row, and obstruction row inside the same finite handoff.

The two routes name only the displayed packet rows. Since neither route contains a discharge of the residual-substitution premise, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, the public certificate exports the finite frontier without claiming those endpoints. □

Theorem 3258.163 (MetaCIC parallel-diamond frontier seed route)**.** *The rotation-selected status route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *records the finite move from no closure block to* seed *through the retained carrier rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route is witnessed inside this child file by public certificate export, finite residual witness consumption, retained seed-block sync, public route sync, bridged export, mature route, bridge boundary, and closurestatus seed block evidence. It keeps residual-substitution premises, local joins, residual-frontier rows, bounded checks, obstruction rows, transport, replay, provenance, and local naming visible, and it does not claim global confluence, unrestricted normalization, subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, residual-substitution discharge, or kernel consistency.

Proof. Start with the closurestatus seed block of Definition 4238.175. It binds the status-bearing route to the retained premise rows and the displayed carrier route $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply public route sync and bridged export, Definition 4238.170 and Definition 4238.169. These keep the public and bridge-facing reads on the same finite route before any outside consumer reads the frontier.

Finally use the mature route and bridge boundary, Definition 4238.173 and Definition 4238.174. They preserve the retained premise budget and refuse the listed global endpoints, so the status route reaches only seed for the finite frontier packet. □

Theorem 3258.164 (MetaCIC parallel-diamond bridged handoff)**.** *The public residual-premise, critical-pair, candidate-join, residual-frontier, closed-checker, obstruction, transport, replay, provenance, and local naming route is the bridged handoff surface for* MetaCICParallelDiamondFrontier$^{\uparrow}$*. The handoff records the finite rows that a standard bridge may read before interpreting the parallel-diamond frontier externally. Residual-substitution discharge, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, and bridge checking remain outside this chapter.*

Theorem 3258.165 (Finite residual witness consumption)**.** *Every candidate-mediated SN/confluence consumer of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must consume the finite residual witness route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through the displayed residual-premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows. The consumer may not retype the handoff as residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Read the public certificate of Definition 4238.162. It exposes the residual premise, critical-pair, candidate-join, residual-frontier, checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as one finite handoff.

Now apply the bridged handoff of Definition 4238.164. The bridge-facing consumer may inspect only the rows named by that handoff before interpreting the frontier externally. The row order $P, K, J, R, S, C, B, O, H, T, G, N$ records the premise, critical-pair, join, residual, seal, checker, bounded-check, obstruction, transport, replay, provenance, and local-name witnesses as finite data.

Since none of those rows is a residual-substitution discharge, subject-reduction theorem, unrestricted Church–Rosser proof, evaluator equality, quotient equality, or kernel-consistency certificate, a consumer that claims one of those endpoints has left the displayed $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ route. □

Theorem 3258.166 (Candidate SN residual budget)**.** *Any candidate-mediated strong-normalization or confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must preserve the residual-substitution premise row P, candidate-join row J, residual-frontier row R, bounded-check row B, and obstruction row O as one finite residual budget before reading the public diamond handoff. The budget is read as a retained row family, not as residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency.*

Proof. Use the public certificate of Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, and obstruction rows inside the same finite public handoff rather than as separate global principles.

Apply finite residual witness consumption, Definition 4238.165. The candidate-mediated consumer can inspect the route $P, K, J, R, S, C, B, O, H, T, G, N$, but the residual budget needed for SN/confluence use is the retained subroute P, J, R, B, O.

Finally compare with the public closure theorem, Definition 4238.167. Since public closure keeps residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, and kernel consistency outside the packet, the candidate SN/confluence reader must carry P, J, R, B, O forward before it reads the public diamond handoff. □

Theorem 3258.167 (MetaCIC parallel-diamond frontier public closure)**.** *Public closure for* MetaCICParallelDiamon *is available only over the retained finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route retains the residual-substitution premise row, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, transport row, replay row, provenance row, and local naming row before any public diamond handoff. It does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Use the public certificate of Definition 4238.162. It exposes the residual premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming rows as a finite public handoff.

Now apply the public seed-readiness route of Definition 4238.84. It keeps the closed-checker seal, closed-normal fragment, bounded-check row, and obstruction row inside the same retained packet rather than moving them to a hidden evaluator or quotient term layer.

Finally use the retained witness consumption of Definition 4238.165. The ordered route $P, K, J, R, S, C, B, O, H, T, G, N$ is the only public surface consumed by downstream SN/confluence fragments. Since the retained rows leave residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, and kernel consistency outside the packet, the public closure is exactly this finite route. □

Theorem 3258.168 (MetaCIC parallel-diamond public frontier budget)**.** *Every public candidate-mediated SN or confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must preserve the residual-substitution premise row P, candidate join row J, residual frontier row R, bounded-check row B, and obstruction row O before reading any public diamond handoff. The retained budget cannot be retyped as residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with the public certificate, Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, and obstruction rows as visible rows of the finite public handoff rather than as hidden global principles.

Use finite residual witness consumption, Definition 4238.165. The candidate-mediated consumer reads the route $P, K, J, R, S, C, B, O, H, T, G, N$, so the residual-substitution premise, candidate join, residual frontier, bounded check, and obstruction rows are retained before the public handoff is inspected.

Apply candidate SN residual budget, Definition 4238.166. It isolates P, J, R, B, O as the budget that a strong-normalization or confluence consumer must carry forward.

Finally compare with public closure, Definition 4238.167. Public closure refuses residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, and kernel consistency, so preserving P, J, R, B, O cannot be retyped as any of those endpoints. □

Theorem 3258.169 (MetaCIC parallel-diamond frontier bridged export)**.** *Any public or bridged read of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier must preserve the candidate-normalization rows, residual-frontier rows, local-join ledger, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *as visible evidence in the same finite carrier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The export may hand this guarded frontier to a bridge-facing consumer, but it does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, residual-substitution discharge, global conversion decidability, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality, Definition 4238.104. It keeps every residual-join read local to the accepted carrier and forces the critical-pair, residual-substitution, candidate-normalization, finite-witness, obstruction, and local naming rows to remain visible.

Next use the residual-normalization window and seed boundary, Definition 4238.126 and Definition 4238.127. These rows retain candidate-normalization evidence, the residual frontier, the obstruction row, the closed-checker evidence, and the bounded-check row in the same finite route.

Finally apply public closure and the bridged handoff, Definition 4238.167 and Definition 4238.164. The structural rows H, T, G, N provide componentwise transport, replay, provenance, and local

naming for the route $P, K, J, R, S, C, B, O, H, T, G, N$. Since those cited rows explicitly leave global confluence, strong normalization, subject reduction, conversion decidability, evaluator equality, quotient equality, residual-substitution discharge, and kernel consistency outside the packet, the bridged export is exactly the guarded evidence-preserving route. □

Theorem 3258.170 (MetaCIC parallel-diamond public route sync)**.** *Every retained residual-budget consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that reads the public or bridged frontier must preserve the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

from public certificate through finite residual witness consumption, candidate SN residual budget, and bridged export. The synchronized route keeps the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *visible in one finite carrier. It refuses residual-substitution discharge, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, and kernel consistency.*

Proof. Start with the public certificate of Definition 4238.162. It exposes the residual-substitution premise, critical-pair, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction, transport, replay, provenance, and local naming rows before any public consumer may read the frontier.

Next use finite residual witness consumption, Definition 4238.165. This forces the consumer to read the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$ as finite witness data, rather than as a hidden residual-substitution theorem.

Apply the candidate SN residual budget of Definition 4238.166. It retains the budget rows P, J, R, B, O inside the larger public route before a strong-normalization or confluence-facing consumer may use the handoff.

Finally apply the bridged export theorem, Definition 4238.169. The export hands the guarded frontier to bridge-facing readers only with the same candidate, residual, obstruction, transport, replay, provenance, and local-name evidence visible. Since all cited routes leave residual-substitution discharge, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, and kernel consistency outside the carrier, the synchronized public route cannot assert those endpoints. □

Theorem 3258.171 (MetaCIC parallel-diamond retained seed block sync)**.** *Let a finite candidate-mediated SN/confluence consumer chain read* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *through the public or bridged frontier. At every successor handoff in that finite chain, the consumer must retain the residual-substitution premise row P, candidate-join row J, residual-frontier row R, bounded-check row B, and obstruction row O before it may read the public route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Thus the finite induction seed block for such consumers is the retained row family P, J, R, B, O together with the public route. The induction invariant does not discharge residual substitution, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. The base handoff is the public certificate of Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, obstruction, transport, replay, provenance, and local naming rows before a consumer reads the frontier.

For the successor step, apply finite residual witness consumption, Definition 4238.165, and the candidate SN residual budget, Definition 4238.166. These two routes force the next consumer to carry P, J, R, B, O as retained finite evidence before inspecting the larger displayed route.

The public and bridged reads are then synchronized by Definition 4238.170 and Definition 4238.169. Induction over the displayed finite consumer chain repeats the same successor handoff, so every stage retains P, J, R, B, O before reading $P, K, J, R, S, C, B, O, H, T, G, N$.

All cited routes explicitly refuse residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. The induction therefore preserves a finite seed block; it does not prove any of those external endpoints. □

Theorem 3258.172 (MetaCIC parallel-diamond retained-premise induction step)**.** *For one successor handoff in any finite candidate-mediated SN/confluence consumer chain of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*, the premise budget rows*

$$P, J, R, B, O$$

are preserved before the next consumer reads the full public route

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The step preserves residual-substitution premise data, candidate-join data, residual-frontier data, bounded-check data, and obstruction data. It does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient equality, global normalization, or kernel consistency.

Proof. Start from finite residual witness consumption, Definition 4238.165. It forces the successor reader to consume the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$ as finite witness data.

Apply the candidate SN residual budget of Definition 4238.166. This isolates P, J, R, B, O as the premise rows that must be carried before any SN/confluence-facing read of the public handoff.

Use public route sync, Definition 4238.170, to keep the full route synchronized across public and bridged reads. Then apply bridged export, Definition 4238.169, which preserves the same guarded frontier rows for bridge-facing consumers.

Finally read the retained seed block sync of Definition 4238.171. It supplies the finite-chain induction context, so the present successor handoff preserves the retained premise budget without proving any refused global endpoint. □

Theorem 3258.173 (MetaCIC parallel-diamond mature route)**.** *Every bridge-facing or critical-path consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *preserves the retained route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through the public certificate, finite residual witness consumption, candidate SN residual budget, public route sync, bridged export, and retained seed block sync. The mature route keeps the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *visible before any consumer reads the bridge-facing frontier. It explicitly refuses residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the public certificate of Definition 4238.162. It displays the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows before a public reader consumes the frontier.

Apply finite residual witness consumption, Definition 4238.165. This forces the bridge-facing reader to consume the finite route $P, K, J, R, S, C, B, O, H, T, G, N$ as residual witness data rather than as a hidden substitution theorem.

Use the candidate SN residual budget, Definition 4238.166. The budget rows P, J, R, B, O are retained before strong-normalization or confluence-facing consumers may inspect the public diamond handoff.

Synchronize the public route by Definition 4238.170. The synchronized route preserves the full displayed family through public and bridged reads, keeping the structural rows H, T, G, N attached to the candidate and residual rows.

Finally apply the bridged export and retained seed block sync, Definition 4238.169 and Definition 4238.171. The former hands the guarded frontier to bridge-facing readers with the retained evidence still visible, and the latter propagates the seed rows through every finite successor handoff.

All cited routes refuse residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. The present route also refuses global normalization, because the retained residual rows provide finite guarded evidence rather than a global evaluator or normalization theorem. □

Theorem 3258.174 (MetaCIC parallel-diamond retained-premise bridge boundary)**.** *The status-bearing boundary for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is the retained-premise route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

of Definition 4238.173. Bridge-facing readers must keep the residual-substitution premise rows P, candidate-join rows J, residual-frontier rows R, bounded-check rows B, and obstruction rows O visible as premise data; the route does not convert them into residual-substitution discharge, subject reduction, full Church–Rosser, quotient equality, evaluator equality, or kernel consistency.

Proof. Apply Definition 4238.173. It binds the public and bridge-facing route to the displayed finite row family $P, K, J, R, S, C, B, O, H, T, G, N$.

Now read the retained-premise rows through Definition 4238.171. The sync keeps P, J, R, B, O as visible prerequisite rows through finite successor handoffs rather than discharging them.

The refusal clauses are inherited from the mature route and from the closure block below. They leave the paper route mature and theorem-checked while keeping bridge checking as the remaining boundary. □

Theorem 3258.175 (MetaCIC parallel-diamond closurestatus seed block)**.** *The status route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is determined by the retained-premise rows P, J, R, B, O, residual-join locality rows, candidate-normalization rows, and frontier-export rows of the displayed carrier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Reading these rows through the public route and bridged export fixes the closurestatus seed block without adding residual-substitution discharge, subject reduction, quotient equality, evaluator equality, full Church–Rosser, global normalization, or kernel consistency.

Proof. Start with the retained seed block sync, Definition 4238.171. It keeps P, J, R, B, O visible through every finite successor handoff before any public or bridged frontier read is consumed.

Apply the mature route, Definition 4238.173. That route already factors the status-bearing surface through public certificate, finite residual witness consumption, candidate SN residual budget, public route sync, bridged export, and retained seed block sync.

Now use the residual-join locality and candidate-normalization rows cited by that route, including Definition 4238.104 and Definition 4238.62. They keep local joins and normalization-facing reads inside the same retained carrier rows rather than turning them into global confluence or normalization theorems.

Finally apply the public route sync and bridged export, Definition 4238.170 and Definition 4238.169. These export only the guarded frontier route $P, K, J, R, S, C, B, O, H, T, G, N$ with the retained premise, obstruction, transport, replay, provenance, and local naming still visible. Hence the closurestatus seed block is fixed by the displayed frontier rows, and none of the refused endpoints is introduced by the status route. □

Theorem 3258.176 (MetaCIC parallel-diamond scoped frontier status)**.** *The scoped* MetaCICParallelDiamondFront *frontier is the finite route that begins with the obligation surface of Definition 4238.2, retains the seed-status route of Definition 4238.79, and reads the public certificate and public closure of Definition 4238.162 and Definition 4238.167. It records the displayed residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows while refusing residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. It fixes the accepted packet through the finite route $P, K, J, R, S, C, B, O, H, T, G, N$ and leaves the residual-substitution premise as a displayed row rather than a discharged theorem.

Apply the seed-status route of Definition 4238.79. The route keeps the same residual premise, critical-pair, candidate, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows visible before any seed consumer reads the frontier.

Finally use the public certificate and public closure, Definition 4238.162 and Definition 4238.167. These theorems expose the finite frontier as a public handoff only over the retained route, so none of the refused residual-substitution, subject-reduction, confluence, evaluator-equality, quotient-equality, global-normalization, or kernel-consistency endpoints is exported by the scoped status. □

Theorem 3258.177 (MetaCIC parallel-diamond scoped status register)**.** *The scoped status of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is registered by the same finite frontier displayed in Definition 4238.176. In particular, the registered route is the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, checker row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local naming row, with no export of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Read the scoped frontier status theorem. It already fixes the route through the obligation surface, seed-status route, public certificate, and public closure while retaining the residual-substitution premise as a visible row.

The status register does not add a new carrier coordinate. It records that this displayed route is the one later readers may cite, and it preserves the same refusals for subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency. □

Theorem 3258.178 (MetaCIC parallel-diamond closurestatus export route)**.** *The seed and bridged closure markers for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are backed by the finite route through the existing critical-pair envelope, residual-substitution dependency route, finite residual witness consumption, candidate-SN handoff, and scoped-frontier status rows. In particular, a closurestatus reader reaches the frontier only through the displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *packet and cannot reinterpret the marker as residual-substitution discharge, unrestricted subject reduction, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the critical-pair and residual-substitution siblings. They retain the local critical-pair rows and the residual-substitution premise as visible finite evidence before any frontier consumer reads the packet.

Next use finite residual witness consumption and the candidate-SN handoff. These rows keep the typed window, candidate-normalization row, bounded local-join checker, obstruction row, transport, replay, provenance, and local naming inside the same displayed route.

Finally read the scoped-frontier status. It registers the same finite route as the status surface for the chapter, so the seed and bridged markers are exported as pointers to that route rather than as global confluence, subject-reduction, evaluator-equality, quotient-equality, residual-substitution, or kernel-consistency theorems. □

Theorem 3258.179 (MetaCIC parallel-diamond retained-premise marker route)**.** *Every seed, public, or bridged marker read for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is consumed only through the displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *route. In particular,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *and* $\mathsf{MetaCICConfluenceAuditPac}$ *consumers retain the critical-pair row* P *and obstruction row* O *while reading the marker, so the marker cannot be reinterpreted as a discharge of the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport row, replay row, provenance row, or local* $\mathsf{NameCert}$ *row.*

Proof. Apply the closurestatus export route above. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the only displayed path by which the marker is exported.

Now restrict the read to $\mathsf{MetaCICCriticalPath}^{\uparrow}$ or $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$. These consumers inspect the marker as a frontier status, not as a replacement for its packet rows. Hence they must retain the critical-pair premise P and the obstruction row O while reading candidate joins, bounded checks, transport, replay, provenance, and local naming.

The retained rows block the overreading. A marker consumer may record where the frontier sits, but it receives no residual-substitution discharge, unrestricted join theorem, global confluence theorem, or kernel-consistency certificate. □

Theorem 3258.180 (MetaCIC parallel-diamond closurestatus guard)**.** *Every closurestatus reader or public status reader of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must pass through the retained residual-substitution premise row, critical-pair envelope, candidate-normalization scope, finite residual witness ledger, candidate-SN handoff, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before it may read the seed or bridged markers. In carrier notation, the reader must expose the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

as frontier data before consuming the status markers. The guard does not discharge residual substitution, prove subject reduction, prove full Church–Rosser, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Begin with the closurestatus export route, Definition 4238.178. It identifies the seed and bridged markers as pointers into the finite $P, K, J, R, S, C, B, O, H, T, G, N$ packet rather than as independent confluence or kernel claims.

Next read the seed package and scoped seed package route, Definition 4238.82 and Definition 4238.103. These keep the residual-substitution premise, critical-pair rows, candidate rows, residual-frontier rows, bounded checker rows, obstruction row, and structural rows visible before a status reader reaches the packet.

Now apply residual-join locality and the finite residual-diamond ledger, Definition 4238.104 and Definition 4238.123. They force every local join read to remain carrier-local, under the retained premise and obstruction rows, and with transport, replay, provenance, and local naming still displayed.

Finally use the candidate-SN handoff, Definition 4238.73. It permits a candidate-mediated frontier budget only after the finite residual witness and bounded checker rows remain visible, so a closurestatus or public status reader can consume the seed or bridged marker only through the displayed guard route. Since the route has no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the status marker. □

The mature paper closure for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ is read through the retained-premise rows, candidate-normalization scope, local-join rows, and frontier-export route already exposed by the sibling files. Those rows bind the displayed frontier packet before any consumer reaches the mature closurestatus surface.

Bridge checking remains absent at this paper site. Until a StandardBridge theorem exists for the same retained-premise, candidate-normalization, local-join, and frontier-export route, the bridge-status coordinate stays separate from the mature paper closure and does not assert residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global Church–Rosser, or kernel consistency.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ packet is generated from residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed checker seal rows, closed-normal fragment rows, bounded-check rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 4238.48, Definition 4238.150, Definition 4238.104, and Definition 4238.62 display the child-route retained-premise, residual-join, candidate-normalization, local-join, residual-frontier, and frontier-export rows $P, K, J, R, S, C, B, O, H, T, G, N$ as the scoped frontier surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObligations`

Bridge status: bridgeChecked

Not claimed: This child route does not close residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, kernel consistency, or checked bridge interpretation beyond the displayed handoff.

Upgrade path: No stronger paper-side upgrade is claimed here; residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency remain outside the displayed bridge-checked route.

Lemma 3258.181 (MetaCIC parallel-diamond canonical closurestatus reader)**.** *The canonical paper status for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is the bridge-checked closurestatus block in* `closurestatus_curren`

Any seed or top-transition reader that enters through the hub must discharge its status read through the child obligation surface BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObliga *and then through that closurestatus block.*

Proof. The hub displays the seed, bridged, and mature route markers, but its body is only a routing list. Therefore a status consumer must leave the hub and read the child files that carry the frontier evidence.

First read the checked obligation surface recorded in Definition 4238.2. It exposes the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming before any status marker is consumed.

Then read the closurestatus child. Its block records maturePackageClosed, bridgeChecked, and `bridgeChecked` for the same Lean obligation target, so the seed and top-transition routes point to that canonical child record rather than to a second hub-level status body. □

Lemma 3258.182 (MetaCIC parallel-diamond frontier closure marker consistency)**.** *The visible* MetaCICParallelDiamondFrontier$^{\uparrow}$ *hub markers are read only through the child closurestatus records. In particular, a consumer that sees the hub-level seed, bridged, or mature marker must route that read through the current closurestatus child, Definition 4238.185, and Definition 4238.205 before using it as closure evidence.*

Proof. The hub exports marker lines and child inputs, but it carries no theorem body of its own. Therefore the evidence-bearing reads are the child files named by the hub, not the marker text itself.

Read the current closurestatus child first, then the seed closure status sync child, and then the seed status block sync child. These rows bind the visible marker strings to the finite frontier packet, closurestatus record, seed-readiness rows, replay, transport, provenance, and local naming.

Thus a downstream consumer cannot treat the hub marker as an independent closure proof. It must pass through the child records that store the corresponding closurestatus evidence. □

Theorem 3258.183 (MetaCIC parallel-diamond bridge hub child route)**.** *The bridged and mature markers displayed by the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *hub are readable only through the child files that expose the carrier, obligation surface, closurestatus guard, and retained-premise route. Concretely, a status reader must pass through the carrier rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

of Definition 4238.1, the obligation surface of Definition 4238.2, the closurestatus guard of Definition 4238.180, and the retained-premise budget of Definition 4238.154 before it may consume those hub markers.

The route records local auditability for the hub status only. It does not discharge residual substitution, prove subject reduction, prove unrestricted confluence, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Start with the hub's two status markers. They are routing annotations for the same finite frontier packet, so a reader must follow the child inputs rather than treating the hub line itself as a proof object.

The carrier child supplies the row tuple $P, K, J, R, S, C, B, O, H, T, G, N$, and the obligation theorem names the residual-substitution premise, critical-pair, candidate-join, residual-frontier,

closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as the exposed surface.

Apply the closurestatus guard. It forces status readers to keep the retained premise, critical-pair envelope, candidate-normalization scope, finite residual witness ledger, candidate-SN handoff, obstruction row, transport, replay, provenance, and local NameCert rows visible before seed or bridged markers are read.

Finally apply the retained-premise mature budget. The same premise, candidate, residual, bounded-check, and obstruction rows remain carried through the public and bridged export route, so the hub markers are locally auditable without supplying residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.184 (MetaCIC parallel-diamond hub child status route)**.** *The seed, bridged, and mature status rows for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are read only through the child carrier route. A status reader must first expose the carrier rows of Definition 4238.1, the obligation surface of Definition 4238.2, the seed closure status of Definition 4238.45, and the closurestatus export and guard routes of Definition 4238.178 and Definition 4238.180. The route exports no residual-substitution discharge, subject reduction, full Church–Rosser theorem, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with the child carrier. The displayed route of Definition 4238.1 is the finite packet

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and the obligation surface of Definition 4238.2 names those rows as the visible premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming data.

Next read the seed status through Definition 4238.45. That theorem keeps the seed-level status attached to the same displayed frontier packet rather than to a detached confluence or consistency assertion.

Finally apply the closurestatus export route and the closurestatus guard. They force bridged and mature status readers through the retained child rows before any status marker is consumed. Since the cited rows contain no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency, the hub status route exports only the guarded child carrier. □

Theorem 3258.185 (MetaCIC parallel-diamond frontier seed closure status sync)**.** *The seed registration route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is subsumed by the existing mature and bridged hub markers together with the child closurestatus route. In particular, any status reader that enters through the seed marker must read the obligation surface of Definition 4238.2, the seed closure status of Definition 4238.45, the closurestatus export route of Definition 4238.178, and the guard of Definition 4238.180 before treating the frontier as a mature paper route.*

Proof. Read the hub markers as pointers into the child files. The marker rows name the mature and bridged status of $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, but they do not add carrier rows beyond the finite residual-premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming route already displayed in the body.

Apply the obligation theorem and the seed closure status theorem. Together they expose the NameCert obligations and the finite seed registration surface before a status reader may inspect the marker.

Now use the closurestatus export route and the closurestatus guard. These theorems force the reader through the retained route $P, K, J, R, S, C, B, O, H, T, G, N$ and keep residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency outside the packet. Hence the seed registration route is absorbed by the existing mature closurestatus route without downgrading the theory or verification axes. □

Theorem 3258.186 (MetaCIC parallel-diamond frontier formal-target request)**.** *The next Lean target for the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier is the residual-substitution premise retention route over the displayed rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

This target asks only that a confluence-facing read retain the residual-substitution premise row P while passing through the critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N. It is not full subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the seed closure status sync of Definition 4238.185. It identifies the status reader with the same child route already guarded by the frontier body, so the requested formal target should be stated over the displayed carrier rows rather than over a global confluence endpoint.

Apply the obligation surface of Definition 4238.2. The obligations expose $P, K, J, R, S, C, B, O, H, T, G, N$ as the finite packet and leave P as a retained residual-substitution premise, not as a discharged theorem.

Finally use the closurestatus export route and guard, Definition 4238.178 and Definition 4238.180. These theorems keep the closurestatus reader on the same retained route and exclude subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, and kernel consistency from the exported target shape. □

Theorem 3258.187 (MetaCIC parallel-diamond frontier retained-premise target boundary)**.** *The next formal boundary for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is precisely preservation of the retained residual-substitution premise row P through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The target asks that a reader carrying P through the critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N still reaches the same retained premise boundary. It does not ask for subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Apply the formal-target request theorem, Definition 4238.186. It already restricts the Lean-facing request to the displayed carrier rows and identifies P as retained premise data rather than a discharged global theorem.

Next read the obligation surface of Definition 4238.2. That surface is the finite packet $P, K, J, R, S, C, B, O, H, T$ so retained-premise preservation is the exact boundary visible to the local NameCert route.

Finally use the closurestatus guard, Definition 4238.180. The guard keeps the exported target on the retained route and refuses subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. □

Theorem 3258.188 (MetaCIC parallel-diamond formal-target consumer retention)**.** *Any consumer of the formal-target request for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must preserve the residual-substitution premise row P through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before it reads either candidate-mediated strong-normalization data or the closurestatus export. The consumer may inspect the critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N only as the same retained-premise packet. It does not receive residual-substitution discharge, subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Apply the formal-target request theorem, Definition 4238.186. It states the request over $P, K, J, R, S, C, B, O, H, T, G, N$ and identifies P as retained premise data.

Next apply the retained-premise target boundary, Definition 4238.187. That boundary forces any downstream read to carry the same P row through the displayed critical-pair, candidate, residual, checker, obstruction, transport, replay, provenance, and naming route.

Finally use the closurestatus guard, Definition 4238.180. Since the guard is the route through which closurestatus export remains visible, a candidate-mediated strong-normalization or closurestatus consumer cannot read the frontier after dropping P or detaching J, R, B, O from the retained packet. □

Theorem 3258.189 (MetaCIC parallel-diamond frontier seed closure route)**.** *The* MetaCICParallelDiamondFrontier *seed closure route is visible only after the critical-pair envelope of Definition 4238.50, residual-join locality route of Definition 4238.104, retained-premise budget of Definition 4238.154, local NameCert rows of Definition 4238.2, seed-status row of Definition 4238.45, closurestatus export row of Definition 4238.178, and closurestatus guard of Definition 4238.180 have all been displayed. The transport, replay, and provenance rows are read as the structural coordinates H, T, G, N of the same child ledger.*

Proof. Read the child ledger rows in order. The critical-pair envelope and residual-join locality route display the local confluence data, while the retained-premise budget and local NameCert rows keep that data attached to the same frontier packet.

The seed-status row then records the local seed claim, and the closurestatus export and guard rows determine what can be exposed at the chapter boundary. Since the route is assembled only from these displayed child rows, the seed record cannot bypass the local ledger. □

Theorem 3258.190 (MetaCIC parallel-diamond frontier seed closure record)**.** *The* MetaCICParallelDiamondFrontie *seed closure record is valid exactly when the displayed child ledger contains the critical-pair envelope of Definition 4238.50, the residual-join locality route of Definition 4238.104, the retained-premise budget of Definition 4238.154, the local NameCert obligations of Definition 4238.2, the seed-status row of Definition 4238.45, the closurestatus export row of Definition 4238.178, and the guard rows of Definition 4238.180. Under this record the structural rows H, T, G, N are read only as transport, replay, provenance, and local naming coordinates of the same child packet.*

The record exports finite seed closure for that displayed packet. It does not assert global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the route of Definition 4238.189. It already orders the critical-pair envelope, residual-join locality route, retained-premise budget, local NameCert obligations, seed-status row, closurestatus export, and closurestatus guard before any seed reader reaches the chapter boundary.

The forward read is therefore finite row projection. If the seed closure record is valid, each cited row is displayed in the child ledger, and H, T, G, N can only transport, replay, source, and name that same packet.

Conversely, assume the seven displayed rows are present. The route theorem repacks them into the seed closure route, so the record has exactly the rows needed for the seed-status reader and the closurestatus export guard.

No global endpoint is introduced in the repacking. The cited rows are carrier-local and retain the premise and obstruction boundaries, so the record cannot become a global confluence theorem, strong-normalization theorem, subject-reduction theorem, evaluator-equality theorem, quotient-equality theorem, residual-substitution discharge, or kernel-consistency certificate. □

The seed-closure status reader for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads the seed record, scoped frontier status, and bridge-status separation as one finite critical-path surface. The seed closure record of Definition 4238.190 supplies the local residual-join ledger, the scoped frontier status of Definition 4238.176 keeps the retained residual premise visible, and the bridge-status sync route keeps bridge checking separate from seed registration.

Thus a MetaCIC critical-path consumer may read the seed axis only through the displayed child files before reaching the frontier closurestatus. The route does not discharge residual substitution, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Theorem 3258.191 (MetaCIC parallel-diamond frontier seed closure marker)**.** *Every seed registration route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *reads the carrier row, critical-pair envelope, candidate-normalization scope, residual-substitution dependency, finite residual witness, local join, obstruction, transport, replay, provenance, and local* NameCert *rows before any candidate-mediated strong-normalization or confluence consumer may use the frontier. In the displayed spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P retains the residual-substitution dependency, K exposes the critical-pair envelope, J records the candidate-normalization row, R, S, C, B carry the finite residual witness and local-join budget, O retains the obstruction row, and H, T, G, N provide transport, replay, provenance, and local naming.

Proof. Start with the frontier carrier and obligation surface. They display the finite rows $P, K, J, R, S, C, B, O, H, T$ and keep the residual-substitution premise, critical-pair data, candidate row, residual frontier, checker rows, obstruction, transport, replay, provenance, and local name as ordinary packet coordinates.

Read the seed-facing child routes next. The critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual-join locality rows force the seed route to pass through the same displayed packet before a consumer sees the frontier.

Finally apply the closurestatus guard and candidate-SN handoff already present in the chapter. They keep the obstruction row and structural coordinates visible before any candidate-mediated strong-normalization or confluence consumer can use the route. Hence the seed closure marker remains a finite frontier registration surface, not residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency. □

Theorem 3258.192 (MetaCIC parallel-diamond frontier seed marker route)**.** *The mature and bridged hub markers for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are read only through the child carrier, obligation surface, seed status, and closurestatus guard. Concretely, any reader entering from the hub's mature-status row or bridged-status row must first expose the carrier packet of Definition 4238.1, the obligation rows of Definition 4238.2, the seed closure status of Definition 4238.45, and the guard route of Definition 4238.180. The marker read therefore remains a finite residual-premise packet read, not a discharge of residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the hub marker rows. They name a status entry for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, but the hub has no independent carrier coordinate and no proof body beyond its ordered child inputs.

Read the child carrier and obligation theorem next. The carrier displays the finite rows $P, K, J, R, S, C, B, O, H, T, G, N$, while the obligation theorem identifies the residual-premise, critical-pair, candidate-join, residual-frontier, bounded checker, obstruction, transport, replay, provenance, and local naming obligations exposed by those rows.

Finally apply the seed closure status and closurestatus guard. These force the status reader to consume the seed route and the guarded closurestatus export before using the mature or bridged marker. Since the exposed packet contains no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, the hub marker exports only the child finite residual-premise status route. □

Theorem 3258.193 (MetaCIC parallel diamond frontier obligation-closure audit)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the residual-join, candidate-normalization, retained-premise, and seed-readiness rows form the finite obligation surface a local confluence consumer must read before claiming a parallel diamond frontier. The audit surface is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with residual joins carrier-local, candidate normalization scoped to the displayed frontier, retained premises still visible, and seed status read only through the packet's closurestatus guard.

Proof. Begin with the seed status theorem, Definition 4238.194. It identifies the frontier seed surface as the finite route through the critical-pair envelope, residual join locality, candidate normalization scope, and finite residual witness rows.

Read residual join locality next, using Definition 4238.104. This keeps every local join inside the displayed packet and prevents the consumer from treating a frontier join as unrestricted confluence.

Attach the finite residual diamond ledger through Definition 4238.123. The ledger supplies the bounded residual-diamond rows and local-join obligations that a confluence consumer must inspect before the frontier can be read.

Finally apply the retained-premise budget and the closurestatus guard, Definition 4238.150 and Definition 4238.180. They keep the retained premise and obstruction rows visible while the seed marker is consumed, so the audit surface remains a finite obligation packet rather than subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 3258.194 (MetaCIC parallel-diamond seed status)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the critical-pair envelope, residual join locality, candidate normalization scope, and finite residual witness rows jointly supply the seed-status surface for parallel-diamond consumers. The exported seed surface is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read through Definition 4238.50, Definition 4238.104, Definition 4238.62, and Definition 4238.69. It is a carrier-local seed-status certificate, not unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the accepted frontier carrier. The critical-pair envelope exposes the bounded overlap and candidate-join rows while keeping the residual frontier, closed-checker, obstruction, transport, replay, provenance, and local naming rows visible.

Next read residual join locality. That theorem keeps residual joins inside the same displayed packet and prevents a downstream consumer from treating the local join as global confluence or premise-free residual substitution.

Now attach candidate normalization scope and finite residual witnesses. Candidate normalization keeps J in the conditional spine with P and O visible, while finite residual witness consumption supplies the typed-window, bounded-check, transport, replay, provenance, and local $\mathsf{NameCert}$ rows needed before a parallel-diamond read can proceed.

The four rows therefore assemble exactly the seed-status surface over $P, K, J, R, S, C, B, O, H, T, G, N$. Since each component is a carrier projection and the ledger exposes no outside equality or kernel theorem, replay and provenance cannot escape the displayed seed packet. □

Theorem 3258.195 (MetaCIC parallel-diamond seed-status exactness)**.** *Any consumer of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *seed-status surface must preserve the same critical-pair envelope* P, K*, the same residual-join rows* J, R, S*, the same candidate-normalization row* C*, the same bounded residual witness rows* B, O*, and the same structural rows* H, T, G, N*. Changing any one of these displayed rows is not the same seed-status packet.*

The exactness claim is only packet-local. It does not assert unrestricted confluence, evaluator equality, quotient term equality, subject reduction, or kernel consistency.

Proof. Apply Definition 4238.194. The seed-status surface exported there is the displayed packet route $P, K, J, R, S, C, B, O, H, T, G, N$ assembled from the critical-pair envelope, residual locality, candidate normalization, and finite witness consumption rows.

Each of those rows is a carrier projection. The critical-pair envelope fixes P, K; residual locality fixes J, R, S; candidate normalization fixes C while keeping the obstruction row visible; finite witness consumption fixes B, O together with the transport, replay, provenance, and local naming rows H, T, G, N.

Therefore a downstream consumer that changes any listed row has changed the packet being consumed. Since the route never exports a global evaluator, quotient equality, subject-reduction theorem, or kernel-consistency theorem, the exactness surface remains a finite seed invariant rather than a global confluence result. □

Theorem 3258.196 (MetaCIC parallel-diamond residual-budget seed status)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the residual-budget seed read is exposed only through the existing critical-pair envelope, residual-join locality, candidate-normalization scope, bounded-check rows, obstruction row, componentwise transport, displayed replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelD}}$ *rows. The seed-facing route is the carrier-local packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read through the hub children for critical pairs, residual joins, candidate normalization, finite residual witnesses, retained premises, scoped seed packages, and seed-status registration.

The route explicitly refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, and kernel consistency.

Proof. Project the accepted frontier carrier. The critical-pair children expose P, K as the bounded overlap and local join envelope before any residual-budget consumer reads a seed status.

Read the residual-join and candidate-normalization children next. They keep J, R, S, C inside the same displayed packet and make the obstruction row O visible rather than discharging it as a global residual-substitution theorem.

The bounded-check and finite-witness children then expose B together with the same transport, replay, provenance, and local naming rows H, T, G, N. These rows record the finite residual-budget route; they do not add a subject-reduction theorem, unrestricted confluence principle, evaluator equality, quotient term equality, global-normalization theorem, residual-substitution discharge, or kernel-consistency certificate. □

Theorem 3258.197 (MetaCIC parallel-diamond seed closure review)**.** *Every seed-level read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must cite the critical-pair envelope, residual-join locality, candidate-normalization scope, finite-residual witness, retained-premise boundary, and closurestatus child routes before the packet can be consumed as a parallel-diamond frontier seed surface. The read is the finite route through* $P, K, J, R, S, C, B, O, H, T, G, N$*; it is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the critical-pair and candidate-normalization rows. The critical-pair envelope and candidate-normalization scope keep K, J, R, S, C, B, O attached to the retained premise row P before any frontier consumer reads the candidate join.

Next read the residual and retained-premise routes. Residual-join locality, finite-residual witness consumption, and retained-premise boundary theorems keep the local join, residual witness, obstruction, and premise rows in the same finite packet.

Finally use the closurestatus child routes. They force every seed-status read through the displayed carrier, obligation, closurestatus export, and guard children, so the structural rows H, T, G, N preserve only the listed seed surface and cannot promote it to any refused global MetaCIC endpoint. □

Theorem 3258.198 (MetaCIC parallel-diamond frontier current seed closure route)**.** *The current seed closure route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is read through the existing frontier carrier, the checked obligation surface, and the seed-status child packet already attached to the hub. A seed consumer must expose the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, bounded checker, obstruction ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before the hub marker can be consumed as a seed closure route.*

This route registers only the paper-side seed closure over the displayed carrier and obligation surface. It does not claim subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Read the frontier carrier of Definition 4238.1. Its rows provide the retained premise, local critical-pair data, candidate join data, residual frontier, bounded checker, obstruction ledger, transport, replay, provenance, and local naming surface.

Apply the obligation surface of Definition 4238.2. The obligation route keeps every seed-facing read on those displayed rows, so the hub marker cannot detach into an unrestricted confluence or evaluator-equality assertion.

Finally compare the seed-status child packet with the hub marker. The child packet records the same carrier and obligation inventory as a current seed route, while the exclusions remain outside the carrier coordinates and outside the local $\mathsf{NameCert}$ surface. □

Theorem 3258.199 (MetaCIC parallel-diamond frontier seed child route)**.** *Every residual-diamond consumer that treats the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *seed route as public must first read the child carrier, the NameCert obligation surface, the closurestatus export row, and the closurestatus guard. In carrier notation the consumer reaches the seed route only through the displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and through Definition 4238.1, Definition 4238.2, Definition 4238.178, and Definition 4238.180. The route does not export residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the hub shape. The parent file contributes only ordered child inputs, so any public seed read must enter through a child packet rather than through an independent hub theorem.

Project the frontier carrier and obligation theorem next. They expose the residual-substitution premise, critical-pair envelope, residual-join rows, candidate-normalization row, bounded witness rows, obstruction row, transport, replay, provenance, and local naming as the finite route $P, K, J, R, S, C, B, O, H, T, G, N$.

Finally apply the closurestatus export and guard. The export row identifies the seed route as a status read of the same packet, and the guard forces residual-diamond consumers to keep that packet visible before treating the route as public. Since the packet has no coordinate for the excluded global endpoints, the seed child route remains a finite frontier handoff. □

Theorem 3258.200 (MetaCIC parallel-diamond seed-axis rotation record)**.** *Every seed-axis status read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *enters through the child seed route before it can be consumed as a frontier status. The route first exposes the seed-readiness row of Definition 4238.83, then the seed-obligation surface of Definition 4238.89, and finally the seed-closure status of Definition 4238.45. It remains a finite residual-premise frontier read and does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the seed-readiness row first. It displays the residual branches, candidate join, normalization-confluence handoff, and critical-path socket before any frontier claim is consumed.

Next apply the seed-obligation surface. The obligation row keeps $P, K, J, R, S, C, B, O, H, T, G, N$ visible as the seed packet, so the status read cannot detach a diamond conclusion from its retained premise and obstruction rows.

Finally use the seed-closure status. It records the same seed packet as the closure-facing route, with transport, replay, provenance, and local naming still present. The rotation record is therefore only a child status read of the displayed packet, not an added global confluence endpoint. □

Theorem 3258.201 (MetaCIC parallel-diamond current seed-axis record)**.** *The current seed-axis read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *enters through the displayed carrier of Definition 4238.1 and the checked obligation surface recorded by* BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDian *It exposes the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment, bounded-check row, obstruction row, transport, replay, provenance, and local naming before any seed-facing consumer can inspect the hub status route.*

Proof. Read the carrier first. Its visible rows are exactly the retained premise, critical-pair, candidate join, residual frontier, checker, fragment, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Next apply the obligation surface. The checked obligation theorem keeps the read on those displayed rows, so the seed-axis status cannot detach from the retained premise or obstruction data.

Finally compare with the existing seed-axis rotation and seed-status routes. They consume the same child packet, so the current record states only the present hub-to-child route and does not claim residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 3258.202 (MetaCIC parallel-diamond frontier hub seed-status route)**.** *Every seed-status read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that enters through the hub is mediated by the existing finite frontier packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route first displays the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming rows, then reads the seed-status children already attached to the hub. It is a seed-facing frontier registration only; it is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the carrier and obligation children of the hub. They expose the retained premise, critical-pair row, candidate-join row, residual-frontier row, checker row, bounded-check row, obstruction row, transport, replay, provenance, and local naming row before a status reader reaches the frontier.

Next consume the seed-readiness, seed-obligation, and seed-status children. These children keep the read on the same displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$ and therefore make the seed status a finite carrier-local record.

Finally compare with the closurestatus child already present in the hub. The route records access to the seed-facing packet only, so no downstream consumer can read it as global MetaCIC confluence, subject reduction, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 3258.203 (MetaCIC parallel-diamond seed-status axis read)**.** *Every seed-status read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that enters through the hub is routed through the same child status packet before it can inspect the frontier closurestatus. The visible seed packet is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

as displayed by Definition 4238.205, Definition 4238.208, and Definition 4238.211. This is a seed-axis interpretation of the existing child status rows only; it does not add residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, kernel consistency, or a second closurestatus block.

Proof. Start with the seed-status block sync. It exposes the retained residual premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows before a status reader can use the frontier.

Next read the seed-status register. The register identifies the hub entry as a route into the same child packet, not as a detached confluence or verification claim.

Finally use the seed status closure block. It keeps the seed-facing read on $P, K, J, R, S, C, B, O, H, T, G, N$ and leaves the closurestatus body to the existing closurestatus child. Hence the hub input records only the child-axis route and exports no additional MetaCIC endpoint. □

Theorem 3258.204 (MetaCIC parallel-diamond frontier seed-status canonical reader)**.** *The seed-status record for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is read only through the displayed child packet consisting of the critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness, and local* NameCert *rows. A consumer reaches that record by passing through Definition 4238.50, Definition 4238.57, Definition 4238.62, Definition 4238.69, and Definition 4238.2 before using the seed-status rows of Definition 4238.203 and Definition 4238.215.*

Thus the canonical seed-status reader records the same finite frontier packet $P, K, J, R, S, C, B, O, H, T, G, N$. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with the critical-pair envelope and residual-substitution dependency route. These rows keep the retained residual premise visible while the frontier is read, so the status record cannot be consumed as a detached diamond theorem.

Next read the candidate-normalization scope and finite residual witness. They restrict the status record to the displayed candidate window and finite residual ledger rather than to global normalization or unrestricted confluence.

Finally use the local NameCert obligation row and the existing seed-status children. They identify the seed-status record as a route through the same child packet $P, K, J, R, S, C, B, O, H, T, G, N$, with no additional endpoint beyond the frontier-local status reader. □

Theorem 3258.205 (MetaCIC parallel-diamond seed-status block sync)**.** *Every seed-status read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is synchronized only through the displayed residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local* NameCert *rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The status block may be consumed as a seed-facing frontier record only after those rows have been displayed in this order. It is not residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with the seed-readiness route of Definition 4238.83. It exposes the residual branches, common join candidate, normalization-confluence handoff, and critical-path socket before any frontier status can be read.

Next apply the seed-obligation surface of Definition 4238.89. That theorem displays exactly the residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as the seed packet.

Finally use the seed-closure status of Definition 4238.45. The closure-facing read preserves the same route $P, K, J, R, S, C, B, O, H, T, G, N$, so the status block is synchronized through the finite frontier rows and cannot be reread as residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 3258.206 (MetaCIC parallel-diamond seed-status paper read)**.** *The paper-side seed-status read for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *records only the local pair* seed/unformalized *along the child route displayed in Definition 4238.205. The read is a seed-facing frontier marker over*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and not a claim of residual-substitution discharge, subject reduction, Church–Rosser, quotient term equality, global normalization, or kernel consistency.

Proof. Project the seed-status block through the theorem above. Its route is already confined to the displayed residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Read the pair seed/unformalized as a paper-side marker for that child route. Since the marker is consumed only after those rows are visible, it cannot become a verification claim or a global MetaCIC endpoint. □

Theorem 3258.207 (MetaCIC parallel diamond frontier seed-status child current)**.** *The hub-level seed status for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is readable only through the current child closurestatus and status records together with the finite residual-join packet. It is not a claim of global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the seed-status current-axis, canonical-reader, and block-sync routes of Definition 4238.203, Definition 4238.204, and Definition 4238.205. These child records are the only public route by which the hub can report the current seed-status row.

Apply the closurestatus guard route Definition 4238.180. It keeps the hub read tied to recorded child closurestatus evidence rather than to an ambient metatheorem.

Finally use the residual-join obligation surface Definition 4238.138. The finite residual-join packet supplies the local critical-pair and residual-join rows consumed by the seed-status record, while the excluded global claims are not fields of that packet. □

Theorem 3258.208 (MetaCIC parallel-diamond frontier seed status register)**.** *The seed route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is registered through the existing carrier and obligation children of this hub. A seed-facing reader first consumes the carrier of Definition 4238.1, the obligation surface of Definition 4238.2, the seed-readiness route of Definition 4238.83, and the seed-status block sync of Definition 4238.205. The registered seed surface is the displayed frontier packet over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read at seed/unformalized *for seed consumers only. It is not global confluence, full Church–Rosser, unrestricted subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier and obligation surface already input by the hub. These rows expose the retained residual premise, critical-pair envelope, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction, transport, replay, provenance, and local naming route.

Use the seed-readiness and seed-status block sync children next. They route seed-facing reads through the same displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ before any status reader can inspect the frontier.

The register therefore records only that this child route is available to seed consumers. Since every cited row is carrier-local and the packet has no coordinate for unrestricted confluence, Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency, the registration does not claim any of those endpoints. □

The to-seed registration row for this child route is seed/unformalized with bridge status none. It records only the seed-facing read of Definition 4238.208; it does not claim full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Theorem 3258.209 (MetaCIC parallel-diamond frontier seed-status registration)**.** *The seed-status route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *registers the parallel-diamond frontier through the already displayed finite packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The registration reads the retained premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming before exposing the seed-facing status children. It is a frontier-local registration only; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the hub carrier and obligation children. They expose the retained residual premise, critical-pair row, candidate-join row, residual-frontier row, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming rows before any status reader can enter the seed route.

Next use the seed-readiness, seed-status, and seed-axis children. These children keep the read on the same finite packet $P, K, J, R, S, C, B, O, H, T, G, N$ and record it as a seed-facing registration over the frontier.

Finally compare with the closurestatus and bridge-status children already routed through the hub. The registration names only the frontier-local seed status, so the refused MetaCIC endpoints remain outside the packet. □

Theorem 3258.210 (MetaCIC parallel diamond frontier seed readiness status)**.** *The* $\mathsf{MetaCICParallelDiamondFront}$ *seed surface is exactly the finite route assembled from residual-window exposure, candidate-normalization rows, residual-substitution rows, the local diamond ledger, componentwise transport, continuation replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row already exposed by the sibling children of this hub. No additional global confluence principle, quotient rewrite relation, or host choice row is exported at the seed boundary.*

Proof. Read the hub children in the displayed route order. The finite residual witnesses and residual-window children expose the window data, while the candidate-normalization and residual-substitution children provide the local normal-form and substitution rows consumed by the frontier packet.

The local diamond, transport, and replay children then route the selected residual pair through the same finite ledger. These rows are still seed-facing: they name the residual join data visible to the packet and do not assert a global confluence theorem for the ambient calculus.

The provenance and NameCert children close the seed surface by recording the source packages and the local certificate row. Since every consumer reaches the packet through those children, the seed readiness status is precisely the listed finite route and has no hidden quotient, choice, or global diamond coordinate. □

Theorem 3258.211 (MetaCIC parallel-diamond frontier seed status closure block)**.** *The seed-facing status read for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is exactly the finite residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Any consumer that reads this status must preserve the same displayed route before confluence, normalization, or kernel-facing language is public. The block is a seed-status certificate only; it

is not full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the carrier and obligation surface already routed through the hub. They display the residual-substitution premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Use the seed-readiness and seed-obligation children. Those routes place $P, K, J, R, S, C, B, O, H, T, G, N$ before any seed-facing reader can inspect the frontier status.

The status block therefore records only the local seed route. Since no listed row supplies full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency, none of those endpoints is exported by the block. □

Theorem 3258.212 (MetaCIC parallel-diamond frontier seed status synchronization)**.** *The hub-level seed record, this child closure block, the retained-premise boundary, the residual-join route, and the candidate-normalization socket expose the same* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *seed-status packet before any later scoped or bridge claim is read. In each location the visible packet is the residual-substitution premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Proof. Start with the seed status closure block above. It identifies the seed-facing status as the displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$ and refuses every confluence, normalization, conversion, quotient, residual-discharge, and consistency endpoint outside that packet.

The hub-level seed record only routes readers into the same children. The retained-premise boundary preserves the residual-substitution premise row P, while the residual-join route preserves the critical-pair, candidate-join, and residual-frontier rows K, J, R.

The candidate-normalization socket then reads the checker, bounded-check, and obstruction rows S, C, B, O without adding a public normalization theorem. Transport, replay, provenance, and local naming remain the shared rows H, T, G, N.

Thus all listed sites expose the same seed-status packet before any scoped or bridge-facing claim is available. □

Theorem 3258.213 (MetaCIC parallel-diamond frontier seed-status child route)**.** *The current seed-status child route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is read through the hub only after the carrier, obligation, seed-readiness, seed-status block sync, seed-status register, and seed-status closure block children expose the finite frontier packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route records a seed-facing child status over the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows. It is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Read the carrier and obligation children first. They display the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows before the hub can route any status reader into a child file.

Next use the seed-readiness and seed-status block sync children. These keep the status read on the same packet $P, K, J, R, S, C, B, O, H, T, G, N$ and mark it as seed-facing rather than as a scoped, public, or bridged endpoint.

Finally compare with the seed-status register and seed-status closure block. They identify the hub input as a route into existing child status evidence, so the hub does not acquire theorem or proof content of its own and the child route exports none of the refused MetaCIC endpoints. □

Theorem 3258.214 (MetaCIC parallel-diamond frontier seed rotation status)**.** *The seed registration route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *reads the existing* `MetacicParallelDiamondFrontierObligations` *surface before any seed-facing status is consumed. A reader enters through the hub marker, then exposes the carrier of Definition 4238.1, the obligation rows of Definition 4238.2, the seed-readiness route of Definition 4238.83, and the seed-status child route of Definition 4238.213. The resulting status is a seed-facing route through*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the hub marker and the local obligation theorem. The marker names the seed-facing status row, while Definition 4238.2 displays the residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Next route the read through seed readiness and the seed-status child. These children keep the status on the same finite packet $P, K, J, R, S, C, B, O, H, T, G, N$ and prevent the hub from carrying theorem content of its own.

The seed rotation status is therefore only a child route into existing packet evidence. Since the displayed rows retain the residual premise and obstruction coordinates, the route cannot be read as any of the refused MetaCIC endpoints. □

Theorem 3258.215 (MetaCIC parallel-diamond frontier status canonical)**.** *The canonical paper-side seed record for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is the seed closurestatus read over the parallel-diamond frontier obligation surface named by* MetacicParallelDiamondFrontierObligations*. A consumer that enters through the hub must read the carrier of Definition 4238.1, the obligation row of Definition 4238.2, and the seed-status routes of Definition 4238.213 and Definition 4238.214 before treating the seed record as available.*

This status record is a seed boundary for the displayed frontier obligations. It does not by itself prove the substitution-sensitive diamond, subject reduction, evaluator equality, quotient term equality, unrestricted confluence, global normalization, or kernel consistency.

Proof. Project the hub marker and the displayed frontier carrier. These rows expose only the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming data.

Next read the existing seed-status child routes. They keep the status on the same finite packet and route the paper-side seed record through the obligation surface already named by MetacicParallelDiamondFrontierObligations.

The record therefore binds a seed closurestatus over those obligations only. Since no row removes the residual-substitution premise or supplies a global join theorem, the substitution-sensitive diamond and the other MetaCIC endpoints remain outside the claim. □

Theorem 3258.216 (MetaCIC parallel-diamond frontier seed-axis summary)**.** *Every seed-axis consumer of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *hub must read the seed route through the child rows of Definition 4238.89, Definition 4238.83, Definition 4238.194, and Definition 4238.215. The parent hub contributes only ordered input routing; it has no independent carrier coordinate, proof body, residual-substitution discharge, subject-reduction endpoint, unrestricted confluence endpoint, evaluator equality, quotient term equality, global normalization, or kernel-consistency row.*

Proof. Start with the seed-obligation child row. It exposes the retained premise, critical-pair row, candidate-join row, residual-frontier row, checker rows, obstruction ledger, transport, replay, provenance, and local naming surface before any seed-facing read is available.

Next use the seed-readiness and parallel-diamond seed-status rows. They keep the same finite packet visible as a seed route and do not move the retained residual-substitution premise into a discharged endpoint.

Finally compare the canonical status row with the parent input list. Since the hub supplies only an ordered route into child files, any seed-axis read factors through those child rows and exports none of the refused MetaCIC endpoints. □

Chapter 3259

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$

Following Bishop and Bridges and Bourbaki on separated completions and Hausdorff uniform filters, the $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ packet records the finite route by which a Cauchy-filter completion surface becomes separated before a Real-facing completion consumer reads it. It sits after the Hausdorff completion surface of chapter 402, the separated completion surface of chapter 1612, the Cauchy-filter completion surface of chapter 1569, the metric completion surface of chapter 1506, and the terminal real-name seal of chapter 38. The packet names the separated-filter completion handoff as finite BEDC data; it is not a quotient-filter construction, a selected representative, a countable-choice limit selector, host equality of completion points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3259.1 (Hausdorff completion filter separation carrier)**.** *A* $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ *carrier is a finite packet*

$$(F, H, S, C, M, R, T, Q, P, N).$$

Here F is the Cauchy-filter completion row, H is the Hausdorff separated row, S is the separated-completion handoff row, C is the completion-continuation row, M is the $\mathsf{MetricCompletion}^{\uparrow}$ *seal, R is the* $\mathsf{Real}^{\uparrow}$ *seal, T is componentwise* hsame *transport over the filter, separation, completion, metric, and real rows, Q is displayed* Cont *replay, P is* Pkg *provenance over the cited sibling certificates and BEDC structural rows, and N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionFilterSeparationUp.HausdorffCompletionFilterSeparationTasteGate`
`single carrier alignment`

Theorem 3259.2 (Hausdorff completion filter separation obligations)**.** *For the carrier of Definition 4239.1, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier admission for the displayed Cauchy-filter completion row F; Hausdorff separation compatibility for H; separated-completion handoff exactness for S and C; MetricCompletion and Real seal agreement for M and R; componentwise transport and replay stability for T and Q; and provenance plus local naming through P and N.*

Proof. Project the carrier. The only completion source row before separation is F, so every completion read begins on the Cauchy-filter completion surface named by chapter 1569.

Read the separation rows next. The Hausdorff row H and separated-completion handoff row S force zero-distance or indistinguishability comparisons through the separated surfaces of chapter 402 and chapter 1612.

Attach the completion seals. The continuation row C, $\mathsf{MetricCompletion}^{\uparrow}$ seal M, and $\mathsf{Real}^{\uparrow}$ seal R keep the separated filter completion connected to the metric-completion and real-name boundaries rather than to host equality of quotient classes.

Finally transport and replay the same finite packet through T and Q, and record provenance and local naming through P and N. Since no coordinate stores a quotient filter, a selected representative, a countable-choice selector, host equality, or ambient completeness, the listed obligations are the whole local certificate surface. □

Theorem 3259.3 (Hausdorff completion filter separation)**.** *Every accepted* $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ *carrier* $(F, H, S, C, M, R, T, Q, P, N)$ *presents a separated Cauchy-filter completion route: a completion consumer must first read* F*, then the Hausdorff and separated handoff rows* H, S, C*, then the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals* M, R*, and only then the structural rows* T, Q, P, N*. The exported completion point is therefore separated at the displayed filter boundary and cannot be read through quotient filters, selected representatives, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from carrier admission. The Cauchy-filter completion row F supplies the only completion source, so a consumer has no route to a completed object before the filter-completion surface is displayed.

Apply the local NameCert obligations. They place H, S, C between the filter source and the completion seals, making Hausdorff separation and separated-completion handoff part of the same finite route.

Read the two seals in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal M records the metric-completion side, and the $\mathsf{Real}^{\uparrow}$ seal R records the real-name boundary consumed by later Real and Completion rows.

The structural rows do not add new completion data. Componentwise hsame transport T, displayed Cont replay Q, Pkg provenance P, and local naming N preserve the same packet, so the exported route is exactly the separated Cauchy-filter completion route and no stronger ambient principle is available. □

Theorem 3259.4 (Hausdorff completion filter separation stability)**.** *For every accepted* HausdorffCompletionFilterS *carrier* $(F, H, S, C, M, R, T, Q, P, N)$*, componentwise* hsame *transport preserves the displayed order*

$$F \longmapsto H, S, C \longmapsto M, R \longmapsto T, Q, P, N.$$

Hausdorff separation therefore remains upstream of the $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals whenever the packet is transported or replayed.*

Proof. Project the carrier of Definition 4239.1. The Cauchy-filter completion source is F, the separation and separated-completion rows are H, S, C, and the completion-facing seals are M, R.

Use Definition 4239.2. Its transport and replay clause assigns T and Q to the same finite packet rather than to a new completion source.

Thus transporting the packet preserves the route from F through H, S, C before M, R are read. The structural rows T, Q, P, N record stability, replay, provenance, and naming after the separation rows have already been displayed. □

Theorem 3259.5 (Hausdorff completion filter Real seal non-escape)**.** *For every accepted* HausdorffCompletionFilterS *carrier* $(F, H, S, C, M, R, T, Q, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R *cannot be read as a quotient-filter construction, selected representative, countable-choice limit selector, host equality of completion points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. It seals only the displayed completion route that has already passed through* F, H, S, C, M *and the structural rows* T, Q, P, N*.*

Proof. Project the same carrier. The only Real-facing coordinate is R, and it appears after the Cauchy-filter source, Hausdorff row, separated-completion handoff, and metric-completion seal.

Apply Definition 4239.2. The obligations identify R as a seal agreement row for the displayed packet, while the transport, replay, provenance, and naming rows preserve that packet without adding new Real data.

No coordinate stores quotient-filter construction, a selected representative, a countable-choice limit selector, host equality, or ambient Real completeness. The seal therefore cannot escape the displayed separated-filter route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ packet is generated from Cauchy-filter completion data, Hausdorff separation data, separated-completion handoff data, MetricCompletion and Real seals, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4239.1, Definition 4239.2, and Definition 4239.3 state the finite separated-filter completion certificate surface over the displayed completion, separation, seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-filter construction, selected representatives, countable choice, host equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, Hausdorff separation compatibility, separated-completion handoff exactness, seal agreement, transport stability, and ledger non-escape for the same finite packet.

Chapter 3260

A Concrete Naming Certificate for $\mathsf{QRDecomposition}^{\uparrow}$

Following Golub–Van Loan and Gram–Schmidt finite-dimensional linear algebra, the $\mathsf{QRDecomposition}^{\uparrow}$ packet records a displayed QR factorization route as finite BEDC data over $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{GramSchmidt}^{\uparrow}$, and $\mathsf{Determinant}^{\uparrow}$. It consumes the matrix carrier of chapter 49, the vector-space rows of chapter 47, the scalar rows of chapter 45, the orthogonalization packet of chapter 1875, and the determinant boundary of chapter 84. The packet names only the finite factorization ledger; it is not a host-level theorem that every matrix admits a selected factorization, nor does it export spectral data, polar decomposition, rank choice, quotient equality, or an ambient norm-completion principle.

Definition 3260.1 (QR decomposition carrier)**.** *A* $\mathsf{QRDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, Q_0, R, G, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the input* $\mathsf{Mat}^{\uparrow}$ *row, including the finite column table, scalar-source row, and vector-space coordinate ledger to be factorized.*
2. *Q_0 is the orthonormal-column packet, listing the displayed output columns together with their inherited* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Field}^{\uparrow}$*, and inner-product support rows.*
3. *R is the upper-triangular packet, recording the finite coefficient table, zero-below-diagonal rows, diagonal pivot witnesses or refusal rows, and scalar provenance.*
4. *G is the* $\mathsf{GramSchmidt}^{\uparrow}$ *handoff row that reads the columns of A through the orthogonalization route before any Q_0 or R read is exposed.*
5. *T is the triangular and reconstruction ledger. It records that every displayed column of A is replayed from the corresponding finite combination of Q_0 columns with coefficients from R.*
6. *H records componentwise* hsame *transport for the input matrix, orthonormal-column packet, upper-triangular packet, Gram–Schmidt handoff, triangular and reconstruction ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$A \longmapsto G \longmapsto Q_0, R \longmapsto T \longmapsto H.$$

8. *P is the* Pkg *provenance over* $\mathsf{QRDecomposition}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{GramSchmidt}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Q_0, R, G, T, H, C, P, N$. It has no coordinate for a selected global basis, spectral factorization, polar decomposition, rank choice, quotient equality, or a host matrix theorem outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.QRDecompositionUp.QRDecompositionTasteGate single carrier alignment`

Theorem 3260.2 (QRDecompositionUp NameCert obligations)**.** *For every accepted* $\mathsf{QRDecomposition}^{\uparrow}$ *carrier*

$$Q = (A, Q_0, R, G, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *input admission for the displayed* $\mathsf{Mat}^{\uparrow}$ *row A and its* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Field}^{\uparrow}$ *support;*
2. *Gram–Schmidt handoff admission for G, requiring the column read of A to pass through the accepted* $\mathsf{GramSchmidt}^{\uparrow}$ *orthogonalization packet;*
3. *orthonormal-column exposure for Q_0, requiring each displayed column to be read from the handoff row rather than from a chosen ambient basis;*
4. *upper-triangular exposure for R, requiring all below-diagonal zero rows, diagonal pivot witnesses or refusal rows, and scalar coefficient rows to be displayed;*
5. *reconstruction exactness for T, requiring the finite column replay of A from Q_0 and R to use only the recorded coefficient rows;*
6. *transport, replay, provenance, and local naming through H, C, P, N, preserving exactly the displayed factorization packet.*

No QR decomposition obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4240.1. The input matrix row A is the only source of displayed columns, and its scalar and vector-space support is inherited from the named dependency rows.

Read the handoff row G. It routes those columns through the $\mathsf{GramSchmidt}^{\uparrow}$ packet, so the orthonormal-column packet Q_0 and coefficient table R cannot be inspected before the finite orthogonalization replay has been displayed.

The upper-triangular obligations are exactly the rows of R: below-diagonal zero entries, diagonal pivot witnesses or refusal rows, and scalar coefficients. The reconstruction ledger T then checks each column of A against the displayed finite combination of Q_0 columns using those same coefficients.

Finally H, C, P, N transport, replay, source, and name the same tuple. Since the carrier has no coordinate for a selected global basis, spectral factorization, polar decomposition, rank choice, quotient equality, or host matrix theorem, the six listed rows exhaust the local naming surface. □

Theorem 3260.3 (Gram–Schmidt handoff for QR decomposition)**.** *Every* $\mathsf{QRDecomposition}^{\uparrow}$ *consumer that reads an orthonormal-column row, an upper-triangular coefficient row, or a reconstruction row must first pass through the* $\mathsf{GramSchmidt}^{\uparrow}$ *handoff row G of Definition 4240.1. Consequently QR reads are finite handoffs from displayed matrix columns to the triangular replay packet, not selected bases or ambient factorization objects.*

Proof. Start with any accepted QR carrier. The only rows that expose Q_0, R, and T are typed after the handoff row G in the displayed route

$$A \longmapsto G \longmapsto Q_0, R \longmapsto T.$$

The handoff row consumes the same matrix columns that A displays and reads them through $\mathsf{GramSchmidt}^{\uparrow}$. Therefore a consumer that asks for an orthonormal column, triangular coefficient, or reconstruction component must first inspect G.

Componentwise hsame transport and Cont replay preserve that order, while Pkg provenance and the local $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ row name only this route. Thus no selected basis, ambient QR theorem, or external factorization object is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{QRDecomposition}^{\uparrow}$ packet is generated from a displayed input MatrixUp row, orthonormal-column packet, upper-triangular coefficient packet, GramSchmidtUp handoff, triangular reconstruction ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4240.1, Definition 4240.2, and Definition 4240.3 seed the finite QR factorization packet over displayed matrix, vector-space, field, Gram–Schmidt, determinant, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global existence of QR decomposition, selected bases, spectral factorization, polar decomposition, rank choice, quotient equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked input admission, Gram–Schmidt handoff admission, orthonormal-column exposure, upper-triangular exposure, reconstruction exactness, and transport/provenance/name exhaustion for BEDC.Derived.QRDecompositionUp.

Chapter 3261

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$

Following Bishop and Bridges regular Cauchy completeness, the $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ packet records the finite-window handoff from local regular Cauchy sections to a terminal $\mathsf{Real}^{\uparrow}$ seal. It consumes the stream-window carrier of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational classifier of chapter 301, and the terminal real-name surface of chapter 38. The packet names the diagonal gluing route; it is not a quotient of streams, a selected representative, a metric completion object, a countable-choice schedule, or an ambient theorem that all Cauchy data already have host limits.

Definition 3261.1 (Regular Cauchy diagonal seal carrier)**.** *A* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, S, E, R, L, G, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite section family, listing the local regular-Cauchy windows admitted for one completion request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family shared by the local sections and the diagonal readback.*
3. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the finite precision rows used to compare the local windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledger, including the accepted local readback rows and the displayed regularity witnesses.*
5. *L is the local* $\mathsf{Real}^{\uparrow}$ *seal row for each admitted section before diagonal gluing.*
6. *G is the diagonal-gluing ledger. It records the finite refinement route that sends the local F, S, E, R, L rows to one terminal diagonal window.*
7. *H records componentwise* hsame *transport for the finite section family, StreamName windows, dyadic tolerance rows, RegSeqRat classifier and regularity rows, local Real seals, diagonal-gluing ledger, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$F, S, E, R, L \longmapsto G \longmapsto H.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, E, R, L, G, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative, a countable-choice schedule, a host completion point, metric-completeness data, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalSealUp.RegularCauchyDiagonalSealTasteGate single carrier alignment`

Theorem 3261.2 (Regular Cauchy diagonal seal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *carrier*

$$D = (F, S, E, R, L, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *finite-section admission for F, requiring every local section to be displayed before the diagonal route is read;*
2. *StreamName and dyadic tolerance compatibility for S, E, requiring all local windows to share the recorded precision ledger;*
3. *RegSeqRat classifier and regularity exposure for R, requiring local readbacks and regularity witnesses to be displayed as finite rows;*
4. *local Real seal exposure for L, requiring each section's terminal read to pass through the same RegSeqRat and tolerance rows;*
5. *diagonal gluing exactness for G, requiring the terminal diagonal window to be assembled from the displayed local rows only;*
6. *transport, replay, provenance, and local naming through H, C, P, N, preserving exactly the same diagonal-seal packet.*

No regular Cauchy diagonal seal obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4241.1. The finite section family F supplies the only local regular-Cauchy windows available to the packet.

The StreamName and dyadic rows S, E place those sections over one displayed precision interface. The RegSeqRat row R then records the classifier and regularity witnesses used by each local readback, and the Real row L records the local terminal seals before they are glued.

The diagonal-gluing ledger G is the only row that assembles a terminal diagonal window from those local data. It therefore carries the exactness obligation for the finite-window handoff and cannot be replaced by a quotient stream, selected representative, or host completion point.

Finally H, C, P, N transport, replay, source, and name the same displayed tuple. Since no further coordinate is present, the six listed rows exhaust the local naming obligations. □

Theorem 3261.3 (Regular Cauchy diagonal seal vision handoff)**.** *The regular Cauchy diagonal seal realizes the finite-window route from local regular sections to the terminal* $\mathsf{Real}^{\uparrow}$ *seal: any read of the terminal seal from a section in F factors through the common* $\mathsf{StreamName}^{\uparrow}$ *window family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and diagonal-gluing ledger of Definition 4241.1.*

Proof. Start with a terminal seal read from a local section. By carrier admission, the section appears in F and its stream window and tolerance rows are displayed in S and E.

The RegSeqRat classifier and regularity ledger R places the local readback inside the regular rational sequence surface, while the row L records the section's local $\mathsf{Real}^{\uparrow}$ seal. These rows are the only available route from the local section to a terminal real-facing read.

Apply the diagonal-gluing ledger G. It sends the local rows through the shared finite window into the terminal diagonal seal and leaves no coordinate for quotient stream equality, selected representatives, countable-choice schedules, metric-completeness data, completion points, or ambient Real completeness. Thus the vision-side regular Cauchy handoff is represented by the displayed NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ packet is generated from a finite section family, StreamNameUp window family, DyadicRatCoreUp tolerance ledger, RegSeqRatUp classifier and regularity ledger, local RealUp seal rows, diagonal-gluing ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4241.1, Definition 4241.2, and Definition 4241.3 seed the finite regular-Cauchy diagonal route from local section windows to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable-choice schedules, metric-completeness data, completion points, ambient RealUp completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked finite-section admission, StreamName and dyadic tolerance compatibility, RegSeqRat classifier exposure, local Real seal exposure, diagonal gluing exactness, and transport/provenance/name exhaustion for BEDC.Derived.RegularCauchyDiagonalSealUp.

Chapter 3262

Modulus of Total Boundedness NameCert Construction

Following Bishop and Bridges constructive metric analysis, the $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ packet records the explicit schedule form of total boundedness: a positive dyadic tolerance request is answered by a finite epsilon-net row together with the finite ledger proving coverage at that tolerance. The packet sits between the metric carrier of chapter 57, the dyadic tolerance surface of chapter 291, the finite observation interface of chapter 126, the total-bounded finite-net carrier of chapter 127, and the compact-metric consumer of chapter 129. It names the schedule consumed by compact and continuous routes without invoking open-cover compactness, selected representatives, countable choice, or ambient completeness.

Falsifiable prediction: A compact-metric or continuous-route certificate that consumes a total-boundedness claim without displaying the dyadic tolerance request and its finite net schedule refutes $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$, because the packet's carrier has no hidden open-cover compactness or selected-net coordinate.

Independence witness: totally_bounded, compact_metric, stream_name, dyadic_rat_core: none is bijective to $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ because the modulus packet jointly records the tolerance request, schedule index, finite center bundle, coverage ledger, metric comparison rows, transport, replay, provenance, and local naming as one displayed schedule surface.

Definition 3262.1 (Modulus of total boundedness carrier)**.** *A* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, D, S, T, K, L, H, C, P, N)$$

with the following rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier and distance comparison ledger.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive tolerance request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window naming the displayed tolerance and center schedule.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net source row.*
5. *K is the finite* $\mathsf{ProbeBundle}$ *of center histories selected by the schedule for the tolerance row D.*

6. *L is the coverage ledger, recording membership, center-carrier rows, distance witnesses, and dyadic comparison rows for the centers in K.*
7. *H records componentwise* hsame *transport for metric, tolerance, window, total-bounded, center-bundle, coverage, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for compact or continuous consumers through $X, D, S, T, K, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, D, S, T, K, L, H, C, P, N$. It has no coordinate for an open-cover subcover, a host-selected finite set, a quotient completion, a global compactness theorem, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ModulusOfTotalBoundednessUp.ModulusOfTotalBoundednessTasteGate single carrier alignment`

Theorem 3262.2 (Modulus of total boundedness NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *carrier $M = (X, D, S, T, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, dyadic tolerance exposure, finite-window exposure, total-bounded source exposure, finite center-bundle exposure, coverage-ledger exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 4242.1. Its visible rows are precisely $X, D, S, T, K, L, H, C, P, N$.

The tolerance schedule is read through D and S before any finite-net row is consumed. The total-bounded row T supplies the finite-net source, while K and L display the center bundle and coverage ledger for that tolerance.

The structural rows H, C, P, N transport, replay, source, and name the same finite surface. Since the carrier has no open-cover, selected-set, quotient-completion, choice, or ambient-completeness coordinate, the NameCert obligations are exactly the displayed schedule rows. □

Theorem 3262.3 (Modulus of total boundedness compact handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or continuous-route consumer of an accepted* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *packet must read the dyadic tolerance row D, the StreamName schedule row S, the* $\mathsf{TotallyBounded}^{\uparrow}$ *source row T, the center bundle K, and the coverage ledger L before it consumes compact finite-net data. The handoff exports a schedule-indexed finite net and no open-cover compactness, selected subcover, countable choice, selected representative, or ambient completeness theorem.*

Proof. Start with the NameCert obligation theorem. It places the tolerance request, finite observation schedule, total-bounded source, center bundle, and coverage ledger among the visible rows of every accepted packet.

A compact-metric or continuous-route read needs the finite-net data. That data occurs only in T, K, L, and those rows are downstream of the dyadic tolerance and schedule rows D, S in the carrier.

Thus the consumer receives a finite dyadic epsilon-net schedule, not an open-cover compactness principle. The excluded coordinates do not occur in the carrier, so they cannot be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$.

Constructive story: A ModulusOfTotalBoundedness$^{\uparrow}$ packet is generated from a MetricSpace row, a DyadicRatCore positive tolerance request, a StreamName finite schedule, a TotallyBounded finite-net source, a ProbeBundle center row, a coverage ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ModulusOfTotalBoundedness}^{\uparrow}}$, seed)

Scope closed: Definition 4242.1, Definition 4242.2, and Definition 4242.3 seed the explicit dyadic epsilon-net schedule route for total-bounded and compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{ModulusOfTotalBoundedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, selected representatives, quotient completion, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite schedule exposure, total-bounded source exposure, finite center-bundle exposure, coverage-ledger exposure, compact handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfTotalBoundednessUp.

Chapter 3263

A Concrete Naming Certificate for $\mathsf{PolishCauchyBasis}^{\uparrow}$

Following Alexandrov–Urysohn, Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet records the basis-level handoff by which a complete separable metric source exposes Cauchy-filter basis data before any completion consumer reads a limit. It sits between the $\mathsf{PolishSpace}^{\uparrow}$ source of chapter 1490, the $\mathsf{CauchyFilterBasis}^{\uparrow}$ generator of chapter 1501, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff of chapter 128, and the $\mathsf{SeparableMetric}^{\uparrow}$ dense schedule of chapter 1458. The packet is a finite BEDC handoff surface for Polish-space Cauchy bases, not a quotient completion, not a choice-selected dense sequence, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3263.1 (Polish Cauchy basis carrier)**.** *A* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, B, K, D, S, R, O, H, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row, carrying the complete separable metric packet over which every later basis read is interpreted.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row, recording the directed finite basis generator used by Cauchy-filter consumers.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row attached to the same metric surface as X; it records the completion-facing route without selecting a limit point.*
4. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule row, stored as public finite name data rather than as a choice function over all balls.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect dense names, basis requests, and Cauchy-window demands.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those stream windows, giving regular rational observations before any basis comparison is made.*
7. *O is the observation ledger joining dense-window requests, regular-rational readbacks, Cauchy-basis refinements, and completion-handoff checks.*
8. *H records componentwise* hsame *transport for X, B, K, D, S, R, O and for the local certificate rows.*

9. *C records displayed* Cont *replay from a Polish-source request through dense schedules, stream windows, regular readbacks, Cauchy-basis refinement, and the completion handoff.*

10. *G is the* Pkg *provenance over* $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only after transporting these displayed rows to a common Polish source, Cauchy-basis generator, completion-handoff row, dense schedule, stream-window row, and regular readback ledger. It has no coordinate for choice-selected dense points, quotient completion equality, host point equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PolishCauchyBasisUp.PolishCauchyBasisTasteGate single carrier alignment`

Theorem 3263.2 (Polish Cauchy basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier $P = (X, B, K, D, S, R, O, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Polish source, Cauchy-filter basis, complete-metric handoff, separable dense schedule, stream windows, regular rational readback, observation ledger, transport, continuation, provenance, and local naming rows;*

2. *common-source classification, requiring the B, K, D, S, R, O rows to be read over the same transported* $\mathsf{PolishSpace}^{\uparrow}$ *source row X;*

3. *basis-directedness exposure, requiring every Cauchy-basis refinement consumed by B to be visible in S, R, O before the completion handoff K is read;*

4. *dense-schedule compatibility, requiring the separable row D and the Cauchy-basis row B to share the same stream-window and regular-readback surface;*

5. *completion-handoff non-escape, requiring every completion-facing consumer to pass through B, S, R, O before reaching K;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4293.1. The projection exposes the Polish source, Cauchy-basis generator, complete-metric handoff, dense schedule, stream windows, regular readbacks, observation ledger, transport rows, continuation rows, provenance row, and local name.

The classifier route starts at X. Componentwise transport in H moves B, K, D, S, R, O together, so a consumer never compares a basis row over one Polish source with a completion handoff or dense schedule over another.

The observation ledger O fixes the finite read order. Dense requests and basis refinements are observed through S and R before either the Cauchy-filter basis or complete-metric handoff can be consumed.

Finally replay C and attach G, N. These rows route, source, and name the same displayed packet, with no additional coordinate for selected dense points, quotient completion equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the listed rows exhaust the local naming surface. □

Theorem 3263.3 (Polish Cauchy basis completion handoff)**.** *Let $P = (X, B, K, D, S, R, O, H, C, G, N)$ be an accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *packet. Every completion-facing consumer of the Polish source*

X reaches the $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K only through the displayed Cauchy-filter basis row B, the separable dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger O, transported by H and replayed by C.*

The visible route is

$$X, D, S, R, B, O \longmapsto K \longmapsto H, C, G, N.$$

It exports a finite completion handoff for downstream completion, Baire, and fixed-point packets; it does not export a quotient completion class, a selected dense sequence, host point equality, or an ambient real-completeness principle.

Proof. Read X from the carrier in Definition 4293.1. The Polish source supplies the common complete separable metric surface, while D, S, R display the finite dense-window observations available to the packet.

Use the NameCert obligations of Definition 4293.2. The basis-directedness and dense-schedule compatibility clauses force the Cauchy-filter basis row B to be read through the same stream windows, regular readbacks, and observation ledger as the dense schedule.

The completion-handoff clause then places K after B, S, R, O. Transport by H and replay by C preserve that order, while G, N package and name the same route. Therefore every completion-facing consumer receives only the displayed finite handoff and cannot bypass it through choice, quotient equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet is generated from a PolishSpace source, CauchyFilterBasis basis, CompleteMetric handoff, SeparableMetric dense schedule, StreamName windows, RegSeqRat readback, an observation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, seed)

Scope closed: Definition 4293.1, Definition 4293.2, and Definition 4293.3 seed the basis-level Cauchy handoff from Polish-space data to completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completions, host point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source classification, basis-directedness exposure, dense-schedule compatibility, completion-handoff non-escape, transport, replay, provenance, and local naming for the same packet.

Chapter 3264

A Concrete Naming Certificate for $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$

The $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ packet realizes the hidden Q_6 stabilizer and orbit-persistence rows of Definition 9462.1352, Definition 9462.1354, Definition 9462.1356, and Definition 9462.1362 as finite BEDC NameCert data. It consumes the six-coordinate codon window of chapter 2234, the boundary tile packet of chapter 3393, the codon lift-orbit classifier of chapter 4059, and the median-closure surface of chapter 3545. The packet exposes support, stabilizer, orbit, and median-closure rows before any consumer may claim the hidden C_2 persistence or the restored $S_3 \times C_2$ surface.

Definition 3264.1 (Symmetry restoration stabilizer carrier)**.** *A* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, A, T, U, M, O, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row, including the six bit coordinates inspected by the stabilizer calculation.*
2. *A is the active-support row for $\mathcal{R}$ and its inactive-anchor closure, read through the boundary-tile packet.*
3. *T is the hidden C_2 stabilizer row, naming the identity and the coordinate transposition $\tau : s_1 \leftrightarrow f_2$ from Definition 9462.1352.*
4. *U is the orbit-persistence row, recording the active-support, inactive-anchor, and first-median-layer orbits of τ from Definition 9462.1354.*
5. *M is the median-closure stabilizer row, recording the displayed $S_3 \times C_2$ restoration surface of Definition 9462.1356.*
6. *O is the orbit ledger joining the nontrivial C_2 orbits, the median-closure orbit table, and the sealing-corner orbit-completion row.*
7. *R is the refusal row separating support symmetry from weighted reassignment symmetry, and excluding biological causality, stochastic fitness, host quotient codon semantics, and hidden enumeration oracles.*

8. *H records componentwise* hsame *transport for Q, A, T, U, M, O, R and for the local certificate rows.*

9. *C records displayed* Cont *replay from a support read through the stabilizer row, orbit-persistence row, median-closure row, orbit ledger, and refusal row.*

10. *P is the* Pkg *provenance over* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}$ $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStabilizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, T, U, M, O, R, H, C, P, N$. It has no coordinate for an unstated support set, an unrecorded stabilizer enumeration, a host quotient codon object, a biological mechanism, or a weighted-symmetry substitute for the support stabilizer.

Lean4 ✓ `BEDC.Derived.SymmetryRestorationStabilizerUp.SymmetryRestorationStabilizerTasteGate`
`single carrier alignment`

Theorem 3264.2 (Symmetry restoration stabilizer NameCert obligations)**.** *For every accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier $S = (Q, A, T, U, M, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStab}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Q_6 codon window, active support, hidden C_2 stabilizer, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, transport, continuation, provenance, and local naming row;*

2. *support visibility, requiring every stabilizer or orbit read to expose the active-support row A before reading T, U, M, or O;*

3. *stabilizer exactness, requiring the hidden C_2 row T to name only the identity and $\tau : s_1 \leftrightarrow f_2$ for the active-support and first-median-layer reads;*

4. *median-closure compatibility, requiring the restored $S_3 \times C_2$ row M to be read only after the median-closure surface and its orbit ledger are displayed;*

5. *orbit non-escape, requiring any hidden C_2 orbit claim to pass through $A, T, U, M, O, H, C, P, N$;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4294.1. The projection exposes the codon window, active support, stabilizer row, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, and the structural rows H, C, P, N.

Support visibility comes first. The row A is the source for the stabilizer calculation, so T, U, M, and O cannot be read as free group-theoretic objects detached from the displayed support.

The stabilizer exactness row is T. It records exactly the identity and the transposition $\tau : s_1 \leftrightarrow f_2$ for the active-support surface, while U records the same nonidentity element along the inactive-anchor and first-median-layer rows.

The median-closure compatibility row is M, with orbit data stored in O. Transport by H and replay by C preserve this order, and P, N package and name the same rows. Thus no hidden orbit claim escapes the displayed support, stabilizer, median-closure, transport, replay, provenance, and local naming surface. □

Theorem 3264.3 (Symmetry restoration orbit non-escape)**.** *Every consumer claiming the hidden C_2 orbit of an accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *packet must display the support row A,*

the stabilizer row T, the orbit-persistence row U, the median-closure row M, the orbit ledger O, componentwise transport H, displayed replay C, package provenance P, and local NameCert *row N. The visible route is*

$$Q, A \longmapsto T, U \longmapsto M, O \longmapsto H, C, P, N.$$

It cannot be replaced by a host stabilizer calculation, an unrecorded orbit enumeration, a weighted-reassignment symmetry, quotient codon semantics, or a biological mechanism claim.

Proof. Read Q, A from the carrier in Definition 4294.1. These rows provide the finite codon-window coordinates and the active support over which the stabilizer calculation is meaningful.

Apply the NameCert obligations of Definition 4294.2. Support visibility and stabilizer exactness place T and U after A, while median-closure compatibility places M and O after the displayed closure surface.

Finally use orbit non-escape. The rows H, C, P, N transport, replay, source, and name the same finite route, and the refusal row separates it from weighted reassignment symmetry or host enumeration. Therefore a hidden C_2 orbit consumer must pass through the displayed packet and cannot claim the orbit from outside it. □

Falsifiable prediction: Any consumer claiming the hidden C_2 orbit without displaying support, stabilizer, median-closure, hsame, Cont, Pkg, and NameCert rows refutes the SymmetryRestorationStabilizer$^{\uparrow}$ packet.

Independence witness: q-six orbit/classification, median-closure sibling slugs: SymmetryRestorationStabilizer$^{\uparrow}$ is not bijective to the orbit/classification or median-closure siblings because this packet jointly carries active support, hidden C_2 stabilizer, orbit persistence, restored median-closure stabilizer, orbit ledger, refusal, componentwise transport, replay, provenance, and local naming as one accepted surface.

The Lean TasteGate file for this packet must include Nontrivial and FieldFaithful instances. The field-faithfulness witness keeps $Q, A, T, U, M, O, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out hidden-orbit claims that omit the support, stabilizer, or median-closure coordinates.

Closure status for NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$.

Constructive story: A SymmetryRestorationStabilizer$^{\uparrow}$ packet is generated from Q6 codon-window rows, active-support rows, hidden C2 stabilizer rows, orbit-persistence rows, median-closure stabilizer rows, orbit ledgers, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, seed)

Scope closed: Definition 4294.1, Definition 4294.2, and Definition 4294.3 seed the finite support-stabilizer packet for the hidden Q6 orbit and median-closure restoration surface.

Formal verification: FormalStatus(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological causality, stochastic fitness, quotient codon semantics, host stabilizer enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, support visibility, stabilizer exactness, orbit persistence, median-closure compatibility, orbit non-escape, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.SymmetryRestorationStabilizerUp.

Chapter 3265

A Concrete Naming Certificate for $\mathsf{TaggedRealPartition}^{\uparrow}$

Following Bishop and Bridges constructive integration, the $\mathsf{TaggedRealPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed tag inside each cell. It refines the located partition carrier of chapter 2248 and the oscillation-controlled partition carrier of chapter 3800; it also reads its endpoint and tag data through the $\mathsf{StreamName}^{\uparrow}$ source of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the finite tag-bearing surface needed by L10 integration and oscillation routes without replacing it by an untagged mesh, a host partition object, selected representatives, quotient real equality, countable choice, or ambient completeness.

Definition 3265.1 (Tagged real partition carrier)**.** *A* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, M, A, G, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row exposing the located interval, finite cell order, endpoint containment, and mesh precision.*
2. *M is the finite mesh row inherited from I, listing adjacent cells and the common refinement ledger used by later integration consumers.*
3. *A is the cell-containment row, certifying for every mesh cell that its endpoints and rational witness rows remain inside the located interval source.*
4. *G is the tag row family, assigning to every displayed cell a tag history whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads certify containment in that same cell before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *O is the oscillation and modulus ledger, recording the tolerance request, the cellwise oscillation bound, and the refinement compatibility supplied by* $\mathsf{FiniteOscillationPartition}^{\uparrow}$*.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for interval endpoints, cell endpoints, and tags.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and mesh precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the source interval, cell containment, tag containment, oscillation ledger, window, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for I, M, A, G, O, S, R, E and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tagged-partition read through*

$$I, M, A, G, O, S, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed interval-source, mesh, containment, tag, oscillation, window, readback, seal, transport, replay, provenance, and local naming rows. It has no coordinate for a host tagged partition, an unbounded tag selector, quotient real equality, chosen completed tag values, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.TaggedRealPartitionUp.TaggedRealPartitionTasteGate single carrier alignment`

Theorem 3265.2 (Tagged real partition NameCert obligations)**.** *For every accepted* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier $T = (I, M, A, G, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval and mesh exposure through I, M, cell-containment certification through A, tag containment through G, oscillation and modulus accounting through O, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seals through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every integration-facing or oscillation-facing read of the tagged partition must therefore pass through the displayed finite mesh, tag-containment, oscillation, window, readback, and seal rows before any terminal real-value consumer is allowed to inspect the packet.

Proof. Project the carrier of Definition 4296.1. The tuple lists the only rows available to a tagged-partition consumer: the interval source, mesh, containment ledger, tag rows, oscillation ledger, finite windows, regular readbacks, real seals, transport, replay, provenance, and local naming rows.

The source interval and cell obligations are I, M, A. The located partition row supplies the interval and mesh; the containment row attaches each cell to the same source, so a tag cannot be read against a detached host interval.

The tag and oscillation obligations are G, O. The tag row certifies cell membership for each tag through the same finite-window and regular-readback route, while the oscillation row records the tolerance and refinement ledger inherited from the oscillation-partition packet.

The readback and seal obligations are S, R, E. A consumer reaches the StreamName windows only after the interval, cell, tag, and oscillation rows are visible, then reads the regular rational rows and finally the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name exactly this route. Since the carrier has no coordinate for a host tagged partition, an unbounded selector, quotient real equality, chosen completed tag values, countable choice, or ambient completeness, the displayed NameCert obligations exhaust the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaggedRealPartition}^{\uparrow}$ packet is generated from a LocatedRealIntervalPartitionUp source, finite mesh and cell-containment rows, per-cell tag rows, oscillation and modulus ledgers, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, seed)

Scope closed: Definition 4296.1 and Definition 4296.2 seed the finite tag-bearing partition interface for L10 integration and oscillation consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close integration convergence, Darboux sums, quotient real equality, selected tag values, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cell containment, tag containment, oscillation-modulus accounting, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TaggedRealPartitionUp.

Chapter 3266

A Concrete Naming Certificate for $\mathsf{FiniteFanBarModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges finite-bar analysis, the $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet records the finite carrier that turns admitted fan-bar rows and local modulus rows into a tolerance-only uniform-modulus ledger. It sits over the fan certificate of chapter 400, the compact metric finite-net surface of chapter 129, the continuous-map carrier of chapter 130, the rational lower-bound fold of chapter 4118, the $\mathsf{Rat}^{\uparrow}$ arithmetic source of chapter 37, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. It realizes the compact-uniform-continuity route of chapter 9237 without importing open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, or ambient real completeness.

Definition 3266.1 (Finite fan-bar modulus carrier)**.** *A* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, L, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite fan-bar admission row, recording the fan, decidable bar classifier, finite depth bound, and finite branch-prefix witness window.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-probe row, aligning the bar prefixes with the compact finite-net probes consumed by the compact-uniform-continuity route.*
3. *L is the local modulus row read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *packet at each admitted bar or probe entry.*
4. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *radius and tolerance row, carrying the finitely many local radii and requested tolerances.*
5. *R is the lower-bound modulus extraction row, obtained by primitive recursion over the finite fan-bar and probe bundle and by certified rational comparison in Q.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, exposing the extracted lower bound as a tolerance-only modulus ledger.*
7. *H records componentwise* hsame *transport for the bar, compact-probe, local-modulus, rational, extraction, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$B, K, L, Q \longmapsto R \longmapsto U.$$

9. *P is the* Pkg *provenance over* FiniteFanBarModulus$^\uparrow$, FanTheorem$^\uparrow$, CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, Rat$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, K, L, Q, R, U, H, C, P, N$. It has no coordinate for an open cover, a selected infinite branch, countable choice, quotient equality of maps, selected limits, a host compactness theorem, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteFanBarModulusUp.FiniteFanBarModulusTasteGate single carrier alignment`

Theorem 3266.2 (Finite fan-bar modulus NameCert obligations)**.** *For every accepted* FiniteFanBarModulus$^\uparrow$ *carrier $F = (B, K, L, Q, R, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *surface consists exactly of carrier admission, finite bar admission through B, compact finite-probe alignment through K, local continuous-map modulus exposure through L, positive rational radius accounting through Q, lower-bound modulus extraction through R,* UniformModulus$^\uparrow$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

In particular, a compact-uniform-continuity consumer may read a uniform modulus from this packet only after the finite bar, compact-probe, local-modulus, rational-radius, and lower-bound extraction rows have all been displayed.

Proof. Project the carrier of Definition 4297.1. The tuple displays exactly the finite bar, compact-probe, local-modulus, rational-radius, lower-bound extraction, uniform-modulus, transport, replay, provenance, and local naming rows.

Finite bar admission is the row B. It supplies the decidable bar classifier and finite depth witness, so the argument ranges over displayed finite prefixes rather than selected infinite branches.

The compact and local-modulus data are K, L, Q. The compact-probe row aligns finite bar prefixes with compact finite-net probes, the continuous-map row supplies local moduli at those probes, and the rational row stores the positive local radii and tolerance requests.

Lower-bound extraction is the row R. It is obtained by primitive recursion over the finite bar and probe rows, using the positive rational comparisons in Q to retain a certified lower bound for all admitted local radii.

Finally, U, H, C, P, N expose, transport, replay, source, and name that extracted lower bound as a UniformModulus$^\uparrow$ ledger. Since the carrier has no coordinate for open-cover compactness, branch selection, countable choice, quotient function equality, selected limits, host compactness, or ambient completeness, the displayed obligations exhaust the accepted packet. □

Theorem 3266.3 (Finite fan-bar lower-bound modulus extraction)**.** *Every compact-uniform-continuity read of an accepted* FiniteFanBarModulus$^\uparrow$ *packet factors through*

$$B, K, L, Q, R, U, H, C, P, N.$$

The extracted row R is the only lower-bound modulus source available to the UniformModulus$^\uparrow$ *consumer: it is computed from finite fan-bar admission, compact finite-probe alignment, local continuous-map moduli, and positive* Rat$^\uparrow$ *radius comparisons, and it exports no open-cover subcover, selected branch, selected limit, quotient map equality, or ambient compactness datum.*

Proof. Start with the NameCert obligations of Definition 4297.2. They require every compact-uniform-continuity consumer to read B, K, L, Q before the lower-bound row R is available.

The extraction step is finite. The bar depth and compact probe rows determine the finite index family, and the positive rational ledger compares the finitely many local radii by primitive recursion to produce a certified lower bound.

Once R is displayed, the uniform-modulus row U may consume it as a tolerance-only ledger. The structural rows H, C, P, N preserve and name the same route under componentwise transport and replay.

No excluded datum can be read from the carrier. Open-cover compactness, selected branches, selected limits, quotient map equality, countable choice, and ambient compactness are absent from $B, K, L, Q, R, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet is generated from finite fan-bar admission, compact finite-probe alignment, local ContinuousMap modulus rows, positive RatUp radius ledgers, lower-bound extraction, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4297.1, Definition 4297.2, and Definition 4297.3 seed the finite fan-bar lower-bound route consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar admission, compact finite-probe alignment, local modulus exposure, RatUp lower-bound extraction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteFanBarModulusUp.

Chapter 3267

A Concrete Naming Certificate for $\mathsf{EdelsteinFixedPoint}^{\uparrow}$

Following Edelstein's compact-metric fixed-point theorem and Bishop–Bridges constructive metric practice, the $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet records the finite NameCert surface for a strictly contractive self-map on a compact metric carrier. It sits after the metric source of chapter 57, the compact-metric finite inspection surface of chapter 129, the sequential map discipline of chapter 1466, the contraction comparison row of chapter 3529, and the displayed iteration route of chapter 2359. The packet names only the compact source, the strict distance-decrease row, the sequential-continuity row, the orbit comparison ledger, transport, replay, provenance, and local naming; it does not import choice, quotient completion, a selected host fixed point, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a Banach contraction ratio.

Definition 3267.1 (Edelstein fixed-point carrier)**.** *An* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, O, C, S, H, R, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier and the distance observations inspected by the fixed-point consumer.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row over X, exposing only the finite compactness windows and cluster-service rows accepted by the compact-metric certificate.*
3. *F is the self-map row, presented through the same metric carrier and never as an untracked host function.*
4. *O is the orbit row generated from F, recording the displayed Picard-style finite prefix consumed by the fixed-point search.*
5. *C is the strict contraction-comparison row: whenever the inspected points are classified as distinct by the metric row, the displayed distance from $F(x)$ to $F(y)$ is strictly below the displayed distance from x to y.*
6. *S is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *row for F, binding cluster reads of the orbit to the image row without adding a semantic continuity theorem outside the displayed packet.*
7. *H stores componentwise* hsame *transport for metric, compactness, self-map, orbit, contraction-comparison, sequential-continuity, continuation, provenance, and naming rows.*

8. *R stores displayed* Cont *replay from the orbit row to the compact cluster row, from the cluster row through sequential continuity, and from the strict comparison row to the fixed-point equality boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, O, C, S, H, R, P, N$. It has no coordinate for quotient completion, countable choice, a Banach contraction constant, an ambient complete real line, or a host-selected fixed point.

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointTasteGate single carrier alignment`

Theorem 3267.2 (Edelstein fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier $E = (X, K, F, O, C, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *surface consists exactly of metric carrier admission through X, compact-metric finite inspection through K, self-map admission through F, Picard-style orbit exposure through O, strict contraction comparison through C, sequential-continuity cluster transport through S, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointCarrier namecert obligations`

Proof. Project the carrier of Definition 4298.1. The metric row X and compact-metric row K are the only source rows available to the certificate, so every inspected point, distance comparison, and compactness read is routed through those two displayed surfaces.

The self-map row F and orbit row O provide the visible finite Picard-style prefix. The strict comparison row C is separate from the Banach contraction surface: it records pointwise distance decrease on distinct displayed pairs and carries no global contraction ratio.

The sequential-continuity row S is the only route from a compact cluster read of the orbit to an image read under F. Transport through H, replay through R, provenance through P, and naming through N preserve this same finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 3267.3 (Edelstein compact-route nonescape)**.** *Every fixed-point read exported by an accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *packet factors through the internal route*

$$X \longmapsto K \longmapsto O \longmapsto S \longmapsto C$$

together with F, H, R, P, N. The route may compare with the $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$ *neighbours as named dependency rows, but it does not consume a Banach contraction constant, quotient completion equality, selected subsequence principle, choice operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a fixed-point read. The carrier first exposes the metric source X and the compactness service K, so the consumer can only inspect displayed finite compact-metric windows and the point classifier attached to X.

The orbit row O supplies the visible Picard-style finite prefix for the self-map F. Compactness is consumed only through K, and the image of a cluster read is transported only through the sequential-continuity row S.

The strict comparison row C is then the only row that can rule out a non-fixed distinct-pair boundary. Since C contains no global ratio and the remaining rows H, R, P, N only transport, replay, source, and name the same finite data, the exported route cannot leave the displayed Edelstein packet for choice, quotient completion, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ fixed-point read factors through one MetricUp source row, one CompactMetricUp finite inspection row, one self-map row, one PicardIterationUp-style orbit row, one strict comparison row, one SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, sequentialcontinuity, banach-contraction, picard-iteration: $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-metric cluster service, strict pointwise distance decrease, and sequential-continuity image transport without a Banach ratio, selected endpoint, quotient completion, or ambient completeness coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite inspection row, a displayed self-map row, a PicardIterationUp-style orbit row, a strict distance-decrease comparison row, a SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4298.1, Definition 4298.2, and Definition 4298.3 bind the compact-metric strict-contraction route through the displayed orbit, cluster, sequential-continuity, and comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, uniqueness, quotient completion equality, Banach contraction constants, selected fixed points, choice principles, bridge checking, Lean verification, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric inspection, orbit exposure, strict comparison, sequential-continuity cluster transport, nonescape, replay, provenance, and local naming for BEDC.Derived.EdelsteinFixedPointUp

Chapter 3268

A Concrete Naming Certificate for $\mathsf{CompletionLadder}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompletionLadder}^{\uparrow}$ packet records the finite BEDC naming surface that ties the current Real and Completion route into one phase object. It sits over the dyadic tolerance ledger of chapter 291, the finite-window stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated-completion boundary of chapter 1612, and the completion-unit handoff of chapter 1527. The ladder is a finite dependency packet for the phase route; it is not quotient stream equality, not a selected limit, not countable choice, not trichotomy, not open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 3268.1 (Completion ladder carrier)**.** *A* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, W, R, E, S, U, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row naming the precision requests that enter the route.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row listing the finite observations read at those tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the dyadic tolerance, stream window, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *witness row that consumes the same displayed real-facing route as a separated completion boundary.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *unit row that records the source-to-completion handoff along the same finite route.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, W \longmapsto R \longmapsto E \longmapsto S, U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, S, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionLadderUp.CompletionLadderTasteGate single carrier alignment`

Theorem 3268.2 (Completion ladder NameCert obligations)**.** *For every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier $L = (D, W, R, E, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, componentwise transport, continuation replay, package provenance, and local naming rows;*
2. *tolerance-window compatibility, requiring each finite stream read in W to be indexed by a dyadic tolerance request in D before regular readback is inspected;*
3. *regular-readback exactness, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *read in R to consume only the displayed D, W route;*
4. *Real-seal order, requiring E to be read only after D, W, R have been displayed;*
5. *completion-boundary order, requiring S and U to consume the same D, W, R, E route through the stored continuation row C;*
6. *transport stability, requiring componentwise* hsame *transport in H to preserve the listed route and no other completion data;*
7. *ledger non-escape, excluding quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, and completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 4299.1. The tuple lists the dyadic tolerance row, stream-window row, regular readback row, Real seal, separated-completion witness, Cauchy-completion unit, transport rows, replay row, provenance, and local name.

The first dependency step reads D and W. Since W is the only stream-window coordinate and D is the only tolerance coordinate, each finite observation entering the regular readback must pass through those two rows.

The regular readback row R is then downstream of D, W, and the Real seal row E is downstream of R. The replay row C records this order before either completion-facing row is inspected.

The separated-completion witness S and Cauchy-completion unit U therefore consume the same displayed D, W, R, E route. Componentwise transport by H preserves those rows, while P and N source and name the same packet.

No remaining coordinate can export quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 3268.3 (Completion ladder separated Real route)**.** *Every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier factors any separated-completion or completion-unit consumer through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{SeparatedCompletion}^{\uparrow},\quad \mathsf{CauchyCompletionUnit}^{\uparrow}.$$

Equivalently, a consumer of S or U must first read the displayed dyadic tolerance row D, stream-window row W, regular readback row R, and Real seal row E, with componentwise transport, continuation replay, provenance, and local naming supplied by H, C, P, N.

Proof. Start from an accepted carrier $L = (D, W, R, E, S, U, H, C, P, N)$. The only completion-facing rows are S and U, so any separated-completion or completion-unit consumer must inspect one of those coordinates.

The definition places both coordinates after D, W, R, E. The continuation row C records the same order, forcing each consumer read to pass from dyadic tolerance and finite window through regular rational readback before the Real seal is available.

Componentwise hsame transport by H may rename the rows, but it preserves the displayed route rather than adding a second stream, a chosen limit, or a host completion object. The package and local name rows P, N then record that the consumer is reading the same finite packet.

Since the carrier contains no row for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the separated Real route above is the whole exported completion-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionLadder}^{\uparrow}$ packet is generated from DyadicRatCore tolerance requests, StreamName finite windows, RegSeqRat readback, a RealUp seal, a SeparatedCompletion witness, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4299.1, Definition 4299.2, and Definition 4299.3 seed the finite Real and Completion ladder over DyadicRatCore, StreamName, RegSeqRat, Real, SeparatedCompletion, CauchyCompletionUnit, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, bridge checking, Lean verification, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window compatibility, regular readback exactness, Real-seal order, separated-completion routing, completion-unit routing, componentwise transport stability, and ledger non-escape for BEDC.Derived.CompletionLadderUp.

Chapter 3269

A Concrete Naming Certificate for $\mathsf{LocatedFormalBar}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Vickers, the $\mathsf{LocatedFormalBar}^{\uparrow}$ packet records a finite located bar/cover boundary for spread paths and formal-topology cover requests. It sits after the cover/refinement carrier of chapter 3967, the admissible-prefix discipline of chapter 1532, the bar-recursion route of chapter 1675, and the finite observation interface of chapter 126. The packet supplies Real and Completion consumers with a finite bar boundary without importing open-cover compactness, selected points, quotient opens, countable choice, or ambient completeness.

Definition 3269.1 (Located formal-bar carrier)**.** *A* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FormalTopology}^{\uparrow}$ *cover/refinement row, read through chapter 3967, that names the finite basic neighbourhoods and refinement requests used by the bar.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admissible-prefix row, read through chapter 1532, that determines which finite path prefixes can enter the bar ledger.*
3. *B is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *bar-recursion row, read through chapter 1675, that records the finite induction surface consumed by bar-cover arguments.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, read through chapter 126, that exposes the inspected prefix windows for Real and Completion routes.*
5. *Q is the located-query row: each accepted request names a finite prefix, a formal basic neighbourhood, and the bar-cover decision boundary visible at that prefix.*
6. *R is the spread-cover row, recording the finite route from an admissible prefix through the formal cover row to the bar-recursion row.*
7. *H records componentwise* hsame *transport for formal-cover rows, spread-law prefixes, bar-recursion rows, StreamName windows, located-query rows, spread-cover rows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion finite-window request through $F, S, B, W, Q, R, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedFormalBar}^{\uparrow}$, $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite path, a selected point of a cover, quotient-open equality, countable choice, host topological equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedFormalBarUp.LocatedFormalBarTasteGate single carrier alignment`

Theorem 3269.2 (Located formal-bar spread-cover route)**.** *Every accepted* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

routes a Real or Completion finite-window consumer through the same displayed spread-cover boundary:

$$F, S, B, W \longmapsto Q, R \longmapsto H, C, P, N.$$

The formal-topology cover row F, spread-law row S, Brouwer bar-induction row B, and StreamName window row W must be visible before the located-query row Q and spread-cover row R are consumed. The route exports no open-cover compactness theorem, selected infinite branch, selected cover point, quotient-open equality, countable-choice schedule, host real equality, or ambient completeness datum.

Proof. Project the carrier of Definition 4300.1. The source rows are F, S, B, W, and the located bar/cover rows are Q, R.

First read the formal-topology and spread-law surfaces. The finite cover/refinement row F supplies only displayed basic neighbourhoods, while S supplies only admissible finite prefixes.

Next pass through the bar-recursion and StreamName rows. Row B keeps the bar argument on the finite Brouwer induction surface, and W fixes the inspected finite windows used by Real and Completion consumers.

Finally replay through Q, R, H, C, P, N. The located query and spread-cover rows are transported componentwise and replayed with the same provenance and local NameCert row. Since the carrier contains no coordinate for compact open covers, selected branches, selected points, quotient opens, choice, host equality, or ambient completeness, no consumer can read those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedFormalBar}^{\uparrow}$ packet is generated from FormalTopology cover/refinement rows, SpreadLaw admissible-prefix rows, BrouwerBarInduction bar-recursion rows, StreamName finite-window rows, located-query rows, spread-cover rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, seed)

Scope closed: Definition 4300.1 and Definition 4300.2 seed the finite located bar/cover boundary for Real and Completion consumers over formal covers, spread prefixes, bar recursion, finite windows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, selected cover points, quotient-open equality, countable choice, host topological equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formal-cover exposure, spread-law prefix exposure, bar-recursion routing, StreamName finite-window replay, located-query exactness, spread-cover exactness, transport, provenance, and local naming for BEDC.Derived.LocatedFormalBarUp.

Chapter 3270

A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$

Following Bishop and Bridges, the CauchyModulusFunctoriality$^{\uparrow}$ packet records the finite data needed to transport displayed Cauchy moduli along a BEDC route without turning that route into a host function equality. It sits between the dyadic tolerance source DyadicRatCore$^{\uparrow}$ of chapter 291, the finite observation route StreamName$^{\uparrow}$ of chapter 126, the regular rational readback RegSeqRat$^{\uparrow}$ of chapter 301, the real-name seal Real$^{\uparrow}$ of chapter 38, the pullback discipline CauchyModulusPullback$^{\uparrow}$ of chapter 1951, and the regular stability surface RegularCauchyModulusStability$^{\uparrow}$ of chapter 1927.

Definition 3270.1 (Cauchy modulus functoriality carrier)**.** *A* CauchyModulusFunctoriality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy-modulus row, naming the inspected source modulus and its finite request boundary.*
2. *T is the target Cauchy-modulus row, naming the target modulus after transport along the displayed route.*
3. *R is the route-map row, recording the finite BEDC route that carries source requests to target requests without asserting equality of host functions.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, read through chapter 291, that keeps each modulus comparison on dyadic rational bounds.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row, read through chapter 126, that records the observation window used by the transported modulus request.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, read through chapter 301, that turns the transported window into a regular rational comparison.*
7. *E is the* Real$^{\uparrow}$ *seal, read through chapter 38, that exposes only the terminal real-name boundary reached by the finite route.*
8. *H records componentwise* hsame *transport for source modulus, target modulus, route map, dyadic tolerance ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a transported modulus request through $S, T, R, D, W, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* CauchyModulusFunctoriality$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStal DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function equality, a selected limit, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusFunctorialityUp.CauchyModulusFunctorialityTasteGate single carrier alignment`

Theorem 3270.2 (Cauchy modulus functoriality NameCert obligations)**.** *For every accepted* CauchyModulusFunctoriality$^{\uparrow}$ *carrier*

$$F = (S, T, R, D, W, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *obligations are exactly the finite route obligations listed by the carrier: source-modulus admission, target-modulus admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The route may use the sibling certificates for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, *and* RegularCauchyModulusStability$^{\uparrow}$, *but it exports no host function equality, selected limit, quotient real equality, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 4301.1. The source and target modulus rows are S, T, and the finite transport route is the displayed row R.

The numeric side is then forced through D, W, Q. The dyadic ledger D supplies the tolerance comparisons, the StreamName window W supplies the finite observation boundary, and the RegSeqRat row Q supplies the regular rational readback.

The terminal read is sealed by E. It records only the Real-facing boundary reached by the same finite route, so it cannot introduce a selected limit or quotient-real equality.

Finally transport and replay through H, C, P, N. These rows preserve the same carrier components under hsame, Cont, package provenance, and local naming. Since the carrier contains no coordinates for host function equality, selected limits, quotient real equality, trichotomy, countable choice, or ambient completeness, the listed NameCert obligations exhaust the packet. □

Closure status for NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$.

Constructive story: A CauchyModulusFunctoriality$^{\uparrow}$ packet is generated from source and target Cauchy-modulus rows, a finite route-map row, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, seed)

Scope closed: Definition 4301.1 and Definition 4301.2 seed the finite modulus-transport certificate over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStability$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, selected limits, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFunctorialityUp.

Chapter 3271

A Concrete Naming Certificate for $\mathsf{RealLimitStability}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealLimitStability}^{\uparrow}$ packet records the finite data by which two displayed terminal $\mathsf{Real}^{\uparrow}$ limit seals remain stable under shared regular-Cauchy evidence. It sits after the terminal Real seal of chapter 38, the regular rational readback discipline of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the completion witness of chapter 3579. The packet names stability of displayed limit seals; it is not quotient stream equality, selected host limits, trichotomy, countable choice, or ambient Real completeness.

Definition 3271.1 (Real limit stability carrier)**.** *A* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two limit-presentation rows whose terminal seals are compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both presentations are inspected.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the shared window.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the common dyadic budgets used by the comparison.*
5. E_0 *and* E_1 *are the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after* L_0, L_1, W, R, D *have been displayed.*
6. H *records componentwise* hsame *transport for both limit presentations, the shared window, readback, tolerance ledger, paired Real seals, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a stability request through* $L_0, L_1, W, R, D, E_0, E_1, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealLimitStability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate single carrier alignment`

Theorem 3271.2 (Real limit stability NameCert obligations)**.** *For every accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier* $S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite packet, classifier stability under componentwise* hsame*, shared-window readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic tolerance nonescape through* $\mathsf{DyadicRatCore}^{\uparrow}$*, paired terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4303.1. The accepted packet exposes exactly two presentation rows, one shared window, one regular rational readback, one dyadic tolerance ledger, two terminal Real seals, transport, replay, provenance, and local naming.

Carrier admission and classifier stability are componentwise. The transport row H moves the two presentations, shared evidence, paired seals, and certificate rows together, while the replay row C preserves the same order of inspection for every stability request.

The provenance row P cites only the listed sibling surfaces and the local row N names the packet. Since the carrier contains no quotient equality, selected host limit, trichotomy oracle, choice schedule, ambient completeness coordinate, Lean certificate, or bridge row, the listed obligations exhaust the NameCert surface. □

Theorem 3271.3 (Real limit stability seal stability)**.** *Every comparison of the paired terminal seals in an accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *packet factors through the displayed route*

$$L_0, L_1 \longmapsto W \longmapsto R \longmapsto D \longmapsto E_0, E_1.$$

Thus the two $\mathsf{Real}^{\uparrow}$ *seals may be reused only under the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger displayed by the packet.*

Proof. Start with a consumer that reads the paired Real seals. The only Real-facing terminal coordinates in the carrier are E_0 and E_1, and both appear after the two presentation rows, shared finite window, regular rational readback, and dyadic ledger.

The consumer must therefore pass through W, R, D before either seal can be used. These rows give the finite observations, regular Cauchy readback, and common tolerance budget that make the two terminal seals comparable as displayed BEDC data.

Finally read H, C, P, N. These rows transport, replay, source, and name the same route, so they preserve seal stability without adding a selected host limit or quotient stream equality. □

Theorem 3271.4 (Real limit stability nonescape)**.** *No accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier exports quotient stream equality, a selected host limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a detached Real seal, Lean verification, or bridge checking. Any downstream completion consumer receives only the two limit presentations, shared window, regular readback, dyadic ledger, paired seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

Proof. Inspect the carrier inventory. The tuple contains no coordinate for quotient equality, selected host limits, trichotomy, choice, ambient completeness, detached seal data, Lean verification, or a bridge certificate.

The paired seals E_0, E_1 are downstream of the shared witness rows W, R, D. Hence a completion consumer can reuse the seals only through the displayed finite evidence and not through an external selected limit.

Transport, replay, provenance, and local naming preserve the same boundary. The rows H, C, P, N move and name only the finite packet, so every exported object remains inside the displayed Real-limit-stability surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitStability}^{\uparrow}$ packet is generated from two limit-presentation rows, a shared StreamName window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4303.1, Definition 4303.2, Definition 4303.3, and Definition 4303.4 seed the finite stability packet for paired terminal Real limit seals under shared StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, shared-window readback, dyadic ledger nonescape, paired Real-seal stability, transport, replay, provenance, and local naming for BEDC.Derived.RealLimitStabilityUp.

Chapter 3272

A Concrete Naming Certificate for $\mathsf{DecimalIntervalReal}^{\uparrow}$

Bishop–Bridges constructive analysis uses rational and decimal brackets as finite evidence before a real name is treated as complete. The $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet records this decimal-bracket carrier as BEDC NameCert data. It sits after the decimal prefix surface of chapter 1417, the finite decimal stream of chapter 4191, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval source of chapter 1709. The packet gives L10 Real and Completion consumers a finite decimal interval carrier without quotient reals, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3272.1 (Decimal interval real carrier)**.** *A* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower finite decimal-prefix row, read through* $\mathsf{DecimalExpansion}^{\uparrow}$ *and the finite* $\mathsf{DecimalStream}^{\uparrow}$ *schedule.*
2. *B is the upper finite decimal-prefix row, read through the same decimal surfaces and carrying the advertised bracket side.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating decimal place values into dyadic comparison margins.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for endpoint-prefix reads, bracket comparison, and downstream replay.*
5. *Q is the bracket-order row, recording that the lower decimal prefix and upper decimal prefix are compared only through the displayed tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint windows, decimal prefix rows, dyadic ledger, and bracket-order row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback has been displayed.*
8. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *handoff row that consumes the same endpoint, tolerance, readback, and seal rows as a located interval boundary.*

9. *H records componentwise* hsame *transport for endpoint prefixes, tolerance rows, windows, bracket order, regular readback, real seal, located interval handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a decimal-bracket request through endpoint prefixes, dyadic comparison, finite windows, regular readback, real sealing, and located interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DecimalIntervalReal}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for a quotient real, a trichotomy decision, a selected decimal representative, host endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier`
Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealTasteGate single carrier alignment`

Theorem 3272.2 (Decimal interval real NameCert obligations)**.** *For every accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower decimal prefix, upper decimal prefix, dyadic tolerance ledger, stream windows, bracket-order row, regular rational readback, real seal, located interval handoff, transport, replay, provenance, and local naming rows;*
2. *decimal-prefix visibility, requiring endpoint reads to enter through A and B before any bracket comparison is inspected;*
3. *tolerance discipline, requiring the place-value comparison to use the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D and no hidden real-order oracle;*
4. *window discipline, requiring each endpoint and comparison read to be scheduled by the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback order, requiring R to consume A, B, D, W, Q before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *located-interval handoff, requiring L to consume the same endpoint, tolerance, readback, and seal route rather than a host interval;*
7. *componentwise* hsame *transport and displayed* Cont *replay for exactly the rows carried by H and C;*
8. *ledger non-escape, excluding quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4357.1. The projection exposes exactly the two decimal endpoint-prefix rows, the dyadic tolerance ledger, the finite window schedule, the bracket-order row, the regular readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming.

The endpoint reads are visible before comparison. Rows A and B carry the finite decimal prefixes, while D supplies the dyadic margin used to compare their place-value readings. The row W schedules the inspected prefixes and comparison windows, so the bracket-order row Q has no access to a hidden real-order decision.

The real-facing route is downstream of the finite bracket data. Row R is read only after A, B, D, W, Q have been displayed, and row E seals the result only after that regular rational readback.

Finally, the located interval handoff L consumes the same displayed route, while H, C, P, N transport, replay, source, and name the packet. Since no remaining coordinate stores quotient equality, trichotomy, a selected decimal representative, host endpoint equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier semantic name certificate`

Theorem 3272.3 (Decimal interval real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow}.$$

Equivalently, a downstream Real and Completion consumer receives the two decimal endpoint-prefix rows, the dyadic comparison ledger, the finite window row, the regular rational readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming before it can use the bracket.

Proof. Start with an accepted carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$. The only real-facing row is E, and the only located-interval row is L.

The definition places E after the endpoint-prefix rows A, B, the dyadic tolerance ledger D, the window schedule W, the bracket-order row Q, and the regular rational readback R. Thus a $\mathsf{Real}^{\uparrow}$ consumer must first inspect the decimal, dyadic, window, and readback rows.

The located interval handoff is not an alternate route. Row L consumes the same A, B, D, W, Q, R, E path, and componentwise transport through H preserves that path under hsame.

The replay, provenance, and local naming rows only record the displayed carrier. They do not add quotient reals, trichotomy, selected representatives, host endpoint equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed route is the whole exported handoff. □

Falsifiable prediction: Every accepted $\mathsf{DecimalIntervalReal}^{\uparrow}$ read displays two finite decimal-prefix rows, one dyadic comparison ledger, one finite StreamName window row, one RegSeqRat readback row, one RealUp seal row, one LocatedRealIntervalUp handoff row, and local transport, replay, provenance, and naming rows before it is consumed as a decimal bracket.

Independence witness: decimalexpansion, decimal-stream, dyadicratcore, streamname, regseqrat, real, located-real-interval: $\mathsf{DecimalIntervalReal}^{\uparrow}$ is not bijective to any listed sibling because it binds two ordered decimal endpoint-prefix rows, one dyadic comparison ledger, finite observation windows, regular rational readback, RealUp sealing, located interval handoff, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet is generated from lower and upper decimal-prefix rows, DecimalStream windows, a DyadicRatCore comparison ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, a LocatedRealIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4357.1, Definition 4357.2, and Definition 4357.3 seed the finite decimal-bracket route over DecimalExpansionUp, DecimalStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealIntervalUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DecimalIntervalRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix visibility, dyadic tolerance discipline, finite-window discipline, regular-readback order, Real seal handoff, located interval handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.DecimalIntervalRealUp.

Chapter 3273

A Concrete Naming Certificate for $\mathsf{AttouchWetsTopology}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet records local Hausdorff-window convergence for closed metric figures as finite BEDC NameCert data over the metric-completion and L10 real-name surface. It sits after the metric carrier of chapter 57, the Hausdorff-distance surface of chapter 3653, the terminal real seal of chapter 38, the regular rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the finite observation topology; it does not import host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3273.1 (Attouch-Wets topology carrier)**.** *An* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose metric, point windows, and bounded-window radii are the only ambient metric data available to the packet.*
2. *H is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *local-window row, read only on displayed bounded metric windows rather than on an ambient hyperspace.*
3. *D is the dyadic tolerance ledger for window radius, Hausdorff comparison margin, and late-window precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, bounded windows, comparison stages, and requested precision cells.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Hausdorff-window comparisons obtained from M, H, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *Q records componentwise* hsame *transport for metric source rows, Hausdorff-window rows, dyadic ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming.*
8. *T records displayed* Cont *replay from an Attouch–Wets observation request through M, H, D, W, R, E.*

9. *C is the local bounded-window classifier row comparing packets only through the displayed metric source, Hausdorff-window, tolerance, stream, readback, seal, transport, replay, provenance, and naming coordinates.*

10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *row.*

The classifier has no coordinate for global closed-set topology, open-cover compactness, selected nearest points, quotient hyperspace equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AttouchWetsTopologyUp.AttouchWetsTopologyTasteGate single carrier alignment`

Theorem 3273.2 (Attouch-Wets topology NameCert obligation surface)**.** *For every accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *expose the* $\mathsf{Met}^{\uparrow}$ *source row M before any local Hausdorff comparison is read;*

2. *display the* $\mathsf{HausdorffDistance}^{\uparrow}$ *window row H only on bounded metric windows named by the packet;*

3. *bind all numerical precision to the dyadic tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

4. *preserve the same finite route by componentwise* hsame *transport Q and displayed* Cont *replay T;*

5. *source and name the same packet through the bounded-window classifier C,* Pkg *provenance P, and the local* NameCert *row N.*

No obligation row is stored for a host hyperspace topology, arbitrary closed-set compactness, selected representatives, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4358.1. The metric row M is the only source of ambient distance data, and the Hausdorff row H is read only after the bounded window has been displayed.

The numerical part is downstream of that window. The tolerance ledger D, finite window row W, regular readback R, and real seal E are the only precision and value rows available to an observation request.

Transport and replay keep the same route. Rows Q and T preserve the displayed metric, Hausdorff, tolerance, window, readback, and seal data, while C, P, N classify, source, and name the finite packet.

Since the carrier contains none of the excluded topology, compactness, selection, quotient, choice, or ambient-completeness coordinates, the displayed obligations exhaust the local NameCert surface. □

Theorem 3273.3 (Attouch-Wets local Hausdorff window)**.** *Every topology-facing read exported by an accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{HausdorffDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}.$$

Thus a downstream metric-completion consumer receives the displayed metric source, bounded local Hausdorff window, finite observation schedule, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming before it can use Attouch–Wets convergence data.

Proof. Start with a topology-facing read of the accepted carrier. The only available metric source is M, and the only Hausdorff comparison row is the local-window row H.

The observation is then forced through D, W, R, E. The dyadic ledger fixes the requested tolerance, the stream row displays the inspected bounded windows, the regular rational row records the readback, and the real row seals the reported Hausdorff-window value.

Finally Q, T, C, P, N transport, replay, classify, source, and name that same finite route. Because no row supplies host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported topology-facing read factors through the displayed local Hausdorff window. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, a HausdorffDistance bounded-window row, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, a bounded-window classifier row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4358.1, Definition 4358.2, and Definition 4358.3 seed the local Hausdorff-window topology route over MetricSpaceUp, HausdorffDistanceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AttouchWetsTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Hausdorff-window exposure, finite-window scheduling, regular readback, Real sealing, transport, replay, classifier exactness, provenance, and local naming for BEDC.Derived.AttouchWetsTopologyUp.

Chapter 3274

A Concrete Naming Certificate for $\mathsf{CantorIntersectionMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet records the metric nested-closed-set intersection route as finite BEDC NameCert data over the L10 completion surface. It sits after the complete-metric source of chapter 128, the metric-completion route of chapter 1506, the nested closed-ball packet of chapter 3526, the Cantor complete metric route of chapter 2199, the Cauchy-filter interface of chapter 333, and the L10 real-name route through chapter 291, chapter 126, chapter 301, and chapter 38. The metric packet is the metric-completion specialization of the finite Cantor-intersection route in chapter 1421 and the theorem-facing intersection route in chapter 2318; it consumes their nested-window and diagonal-readback rows before exposing complete-metric service. The packet names a nested metric intersection handoff; it does not assert open-cover compactness, arbitrary closed-set compactness, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3274.1 (CantorIntersectionMetric carrier)**.** *A* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence service are the only complete-metric data available to the packet.*
2. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row naming the surrounding completion request before any intersection point is read.*
3. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row, recording the descending closed metric ball spine and its finite containment witnesses.*
4. *K is the* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *row, used only as the displayed nested closed-ball completion handoff.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested metric spine and read only through the displayed filter route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, storing the shrink schedule, late-radius comparisons, and requested precision bounds.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, radii, and containment data.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of the displayed center schedule.*
9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, K, F, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for the complete-metric source, completion route, nested-ball row, Cantor handoff, filter, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a nested metric intersection request through $M, E, B, K, F, D, S, R, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, B, K, F, D, S, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary closed-set compactness, quotient completion equality, a selected limit, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionMetricUp.CantorIntersectionMetricTasteGate single carrier alignment`

Theorem 3274.2 (CantorIntersectionMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the complete-metric and metric-completion source rows M, E before any intersection witness is read;*
2. *display the nested metric route through B, K, F, namely the nested closed-ball spine, Cantor complete-metric handoff, and generated Cauchy-filter row;*
3. *bind all numerical readback to the L10 rows D, S, R, L, namely the dyadic tolerance ledger, finite stream windows, regular rational handoff, and terminal real seal;*
4. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
5. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row is stored for open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, trichotomy, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4359.1. The rows M, E expose the metric and completion sources before the packet can read an intersection point.

The nested metric content is then displayed through B, K, F. The nested closed-ball row supplies the descending closed metric spine, the Cantor complete-metric row supplies the existing handoff, and the Cauchy-filter row records the generated filter route.

The L10 readback is downstream. The dyadic ledger D, stream windows S, regular rational row R, and real seal L are the only numerical coordinates available before a completion-facing consumer receives a named output.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains none of the excluded compactness, quotient, selected-limit, trichotomy, or ambient-completeness coordinates, the five displayed obligations exhaust the local NameCert surface. □

Theorem 3274.3 (CantorIntersectionMetric complete handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier factors through the displayed nested metric route*

$\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, S

Thus a consumer receives the complete-metric service, completion request, nested closed-ball spine, Cantor complete-metric handoff, Cauchy-filter row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows before it may use the intersection output. The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness principle, open-cover compactness theorem, arbitrary closed-set compactness, quotient stream equality, or selected limit.*

Proof. Begin with a completion-facing read of the accepted carrier. The first available rows are M and E, so the read is grounded in the displayed complete-metric service and the surrounding metric-completion request.

Next pass through the nested route B, K, F. The descending closed-ball spine is read through the $\mathsf{NestedClosedBall}^{\uparrow}$ row, the existing Cantor complete-metric handoff is read through K, and the generated filter is read through F.

The numerical output is then forced through the L10 readback rows D, S, R, L. The dyadic ledger supplies tolerance control, the stream row supplies finite windows, the regular rational row supplies readback, and the terminal real seal is reached only after those rows are present.

The structural rows H, C, P, N preserve the same route for downstream consumers. Because no row supplies full real completeness, open-cover compactness, arbitrary closed-set compactness, quotient stream equality, or a selected limit, every completion-facing read factors through the displayed nested metric route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, a MetricCompletion request row, a NestedClosedBall spine, a CantorCompleteMetric handoff, a CauchyFilter row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4359.1, Definition 4359.2, and Definition 4359.3 seed the metric nested-closed-set intersection route over displayed completion, nested-ball, Cantor handoff, filter, dyadic, stream, regular readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested metric inclusion, Cantor handoff, Cauchy-filter generation, dyadic shrink exactness, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionMetricUp.

Chapter 3275

A Concrete Naming Certificate for $\mathsf{TightRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{TightRealInterval}^{\uparrow}$ packet names a finite lower-upper real interval by displaying both endpoint names and every rational cell through which those endpoints are inspected. It sits downstream of the located endpoint surface of chapter 394 and the dyadic refinement surface of chapter 306: consumers may read a tight interval only after the lower endpoint, upper endpoint, rational cell ledger, dyadic refinement ledger, stream windows, regular rational readbacks, and Real seal rows have all been displayed.

Definition 3275.1 (TightRealInterval carrier)**.** *A* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, U, Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row and U is the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger recording the finite rational windows that enclose the endpoint reads and witness lower-before-upper compatibility.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to the rational cells in Q.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the endpoint observations and interval cells inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and cell refinements.*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached from L, U, S, R, Q, D.*
7. *H records componentwise* hsame *transport for endpoints, rational cells, dyadic refinements, stream windows, readbacks, Real seals, replay, provenance, and the local name.*
8. *C records displayed* Cont *replay through $L, U, Q, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host interval equality, chosen endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.TightRealIntervalUp.TightRealIntervalUp`
Lean4 ✓ `BEDC.Derived.TightRealIntervalUp.TasteGate.TightRealIntervalTasteGate single carrier alignment`

Theorem 3275.2 (TightRealInterval NameCert obligations)**.** *For an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier* $T = (L, U, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, rational and dyadic interval ledgers, finite stream windows, regular rational readbacks, paired Real seal rows, componentwise transport, continuation replay, package provenance, and local naming; endpoint-order visibility through* Q*; dyadic refinement discipline through* D*; stream-readback exactness through* S, R*; and ledger non-escape from host intervals, selected endpoints, quotient real equality, choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 4360.1. The lower and upper endpoint rows, the rational cell ledger, the dyadic refinement ledger, finite windows, regular readbacks, Real seals, transport rows, replay rows, provenance row, and local naming row are all visible before any interval consumer can read the packet.

Endpoint-order visibility is supplied by Q. Since each dyadic refinement in D is attached to a rational cell, the refinement discipline cannot detach from the endpoint ledger.

Stream-readback exactness is then read through S and R. The rows H, C, P, N transport, replay, source, and name the same finite route, while the excluded interval and completeness data have no carrier coordinate. □

Theorem 3275.3 (TightRealInterval Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

The read exposes lower and upper endpoint windows together with the rational and dyadic cell ledgers that bind them. It does not expose a host real interval, a selected endpoint value, quotient real equality, countable choice, trichotomy, or ambient real completeness.

Proof. Begin with the accepted carrier. The only seal-facing row is E, and it is displayed after L, U, Q, D, S, R in the finite packet.

Apply Definition 4360.2. Endpoint-order visibility forces the Real read to keep the rational cell ledger Q, dyadic refinement discipline forces any dyadic window through D, and stream-readback exactness forces finite observations through S, R.

Finally route through H, C, P, N. These rows preserve and name the same packet, so no host interval, endpoint selector, quotient equality, choice schedule, trichotomy row, or ambient completeness principle can enter the Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TightRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, paired $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4360.1, Definition 4360.2, and Definition 4360.3 seed the finite lower-upper Real interval packet over located endpoints, rational cells, dyadic refinements, stream windows, regular readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Scoped closure requires a checked carrier-obligation package and a public consumer route for the same finite lower-upper interval packet.

Chapter 3276

A Concrete Naming Certificate for CompactUniformModulusTransfer$^{\uparrow}$

Following Bishop-style compact metric uniform continuity, the CompactUniformModulusTransfer$^{\uparrow}$ packet records the finite handoff that transfers compact finite-net and pointwise continuity data into a uniform modulus row. It consumes the finite compact source of chapter 60, the compact metric net surface of chapter 129, the continuity interface of chapter 391, the finite stream and regular readback rows of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite modulus-transfer NameCert surface, not an open-cover compactness proof or a host modulus selector.

Definition 3276.1 (Compact uniform modulus transfer carrier)**.** *A* CompactUniformModulusTransfer$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, U, S, R, D, F, H, C, P, N)$$

with the following visible rows:

1. *K is the* Compact$^{\uparrow}$ *finite-net source row.*
2. *G is the* CompactMetric$^{\uparrow}$ *metric coverage row for the same finite centers and source distance witnesses.*
3. *U is the* UniformContinuity$^{\uparrow}$ *pointwise-to-uniform continuity row consumed by the transfer.*
4. *S is the* StreamName$^{\uparrow}$ *window row listing the inspected finite centers, tolerance requests, and pointwise-radius reads.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used for finite radius comparison and shrinkage.*
7. *F is the modulus-transfer ledger that records the finite lower-bound handoff into the downstream uniform-modulus consumer row.*
8. *H records componentwise* hsame *transport for compact, compact-metric, uniform-continuity, stream, readback, dyadic, transfer, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay through $K, G, U, S, R, D, F, H, P, N$.*
10. *P is the* Pkg *provenance over* Compact$^{\uparrow}$*,* CompactMetric$^{\uparrow}$*,* UniformContinuity$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate`
`single carrier alignment`

Theorem 3276.2 (Compact uniform modulus transfer obligations)**.** *For an accepted* CompactUniformModulusTrans *carrier $M = (K, G, U, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net row, compact-metric coverage row, uniform-continuity row, stream windows, regular rational readbacks, dyadic tolerance ledger, modulus-transfer ledger, componentwise transport, continuation replay, package provenance, and local naming; source lock between K, G, U; finite-window exactness through S, R; dyadic tolerance exactness through D; and non-escape from open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient equality, choice, and ambient completeness.*

Proof. Project the carrier of Definition 4361.1. The compact row, compact-metric row, continuity row, stream windows, readbacks, dyadic tolerance ledger, transfer ledger, transport rows, replay rows, provenance row, and local name are displayed as one finite packet.

The source lock is the shared read of K, G, U. A transfer consumer cannot use the compact finite-net row without the compact-metric distance surface and the continuity row that supplies pointwise radius data.

Finite-window and dyadic exactness are read through S, R, D. The ledger F can therefore transfer only the displayed finite lower-bound data, and H, C, P, N preserve that route without adding any excluded compactness or choice principle. □

Theorem 3276.3 (Compact uniform modulus transfer handoff)**.** *Every accepted* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *packet hands a downstream uniform-modulus consumer only the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont}$$

The handoff records compact finite-net coverage, pointwise continuity radius reads, finite stream windows, regular rational readbacks, dyadic tolerance comparison, and the transfer ledger. It exports no open-cover compactness theorem, selected net center, selected Lebesgue number, host modulus function, quotient equality, countable choice, or ambient completeness.

Proof. Start from the transfer obligations of Definition 4361.2. They identify the compact, compact-metric, continuity, stream, readback, dyadic, and transfer rows as the only NameCert-readable data.

Route the downstream consumer through F. The ledger can be consumed only after K, G, U, S, R, D have supplied the finite-net, pointwise-radius, window, readback, and tolerance rows.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since the excluded compactness, selection, equality, choice, and completeness data have no coordinate in the carrier, the downstream handoff is exactly the displayed finite modulus-transfer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet is generated from compact finite-net rows, compact-metric coverage rows, uniform-continuity rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a modulus-transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, seed)

Scope closed: Definition 4361.1, Definition 4361.2, and Definition 4361.3 seed the finite compact-net modulus-transfer packet for uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Upgrade path: Scoped closure requires a checked transfer-obligation package and a public uniform-modulus consumer route for the same finite packet.

Chapter 3277

A Concrete Naming Certificate for $\mathsf{RealIntervalApartness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntervalApartness}^{\uparrow}$ packet records located separation of two real intervals by a displayed positive dyadic gap rather than by a trichotomy decision on completed real endpoints. It sits after the terminal real-name seal of chapter 38, the located endpoint interval surface of chapter 1709, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the separability boundary of chapter 1809. The packet names interval separation only through visible endpoint rows, gap rows, transport, replay, provenance, and local naming.

Definition 3277.1 (RealIntervalApartness carrier)**.** *A* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are the two* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint packets whose lower and upper endpoint rows are exposed before any apartness read.*
2. *L, U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *seal rows for I, and L', U' are the corresponding* $\mathsf{Real}^{\uparrow}$ *seal rows for J.*
3. *G is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness, attached either to the displayed comparison $U + G \leq L'$ or to the displayed comparison $U' + G \leq L$.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the endpoint observations and gap inspections used by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and dyadic gap checks.*
6. *E records the* $\mathsf{RealSeparability}^{\uparrow}$ *boundary row witnessing that the two exposed endpoint intervals are separated by the displayed dyadic gap.*
7. *H records componentwise* hsame *transport for intervals, endpoint seals, dyadic gap rows, finite windows, readbacks, separability rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay through $I, J, L, U, L', U', G, S, R, E, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient real equality, a selected endpoint representative, a host interval order, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 3277.2 (RealIntervalApartness NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *admit the two located real interval packets I, J and expose their lower and upper endpoint rows through L, U, L', U';*
2. *expose the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness G and the side on which the gap separates the displayed endpoint rows;*
3. *bind every endpoint and gap inspection to the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, R;*
4. *preserve the interval, endpoint, gap, readback, and separability rows by componentwise* hsame *transport through H;*
5. *replay the same finite route through* Cont *row C, source it by* Pkg *provenance P, and name it by the local* NameCert *row N.*

No obligation row is stored for real trichotomy, quotient equality of completed real names, selected endpoint representatives, host interval order, or ambient real completeness.

Proof. Project the carrier of Definition 4362.1. The two located interval packets and their endpoint seals are the only interval data exposed to a consumer.

The separation datum is the row G. Positivity is read as a dyadic-row property, and the side of separation is part of the displayed ledger; no trichotomy comparison is consulted.

The numerical readback is forced through S and R, then through the separability row E. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since none of the excluded host-order, quotient-equality, representative-selection, trichotomy, or completeness data occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealIntervalApartnessUp.TasteGate.RealIntervalApartnessTasteGate single carrier alignment`

Theorem 3277.3 (RealIntervalApartness separability route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, hsame, Cont, Pkg, Na

The route exports located interval separation only as a positive dyadic gap between displayed endpoint rows. It exports no real trichotomy, quotient real equality, host interval order, selected endpoint representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the NameCert obligations of Definition 4362.2. They force the two located interval packets and endpoint seals to be visible before a consumer may read an apartness claim.

Next read the dyadic gap row and its finite endpoint inspections through the stream and regular-rational rows. The separability row is therefore attached to the displayed positive gap, not to a host comparison of completed reals.

Finally route through transport, replay, provenance, and local naming. These rows preserve the same finite packet, so the consumer receives located interval apartness without importing any excluded trichotomy, quotient, selection, host-order, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalApartness}^{\uparrow}$ packet is generated from two LocatedRealInterval endpoint packets, Real endpoint seal rows, a positive DyadicRatCore gap witness, finite StreamName windows, RegSeqRat readback, a RealSeparability boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4362.1, Definition 4362.2, and Definition 4362.3 seed the located interval separation packet over displayed endpoint, dyadic gap, finite-window, regular-readback, Real seal, separability, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient real equality, selected endpoint representatives, host interval order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, positive-gap exposure, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.RealIntervalApartnessUp.

Chapter 3278

A Concrete Naming Certificate for $\mathsf{PartitionBasinPolynomial}^{\uparrow}$

Following finite enumerative combinatorics, the $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ packet records the partition-basin product formula of Definition 9462.1736 as displayed NameCert data over block sizes, legal decorations, defect exponents, incidence totals, transport, replay, provenance, and local naming. It sits after the partition carrier of chapter 201, the defect-flow partition skeleton of chapter 4236, and the partition-core incidence certificate of chapter 4214. The packet names a finite polynomial enumeration surface; it is not a biological stochastic model, a host generating-function oracle, or a quotient-level semantic identification.

Definition 3278.1 (PartitionBasinPolynomial carrier)**.** *A* $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite block-count row for the partition core, and S is the finite block-size ledger $(s_1, \ldots, s_m)$.*
2. *D is the legal-decoration row, recording for each coordinate of block i the admissible subset of the other $m-1$ rows and the private-coordinate exclusion.*
3. *E is the defect-exponent ledger that assigns exponent weight q^r to a coordinate receiving r additional rows.*
4. *F is the product-formula row*
$$\prod_{i=1}^{m} \left[(1+q)^{(m-1)s_i} - \left((1+q)^{m-1} - 1 \right)^{s_i} \right].$$
5. *Q is the four-coordinate specialization row for the five partition types 4, $3+1$, $2+2$, $2+1+1$, and $1+1+1+1$.*
6. *I is the incidence-total row that reads the quotient-level and codon-level totals from the displayed specialization.*
7. *T records the downstream defect-flow handoff to the partition skeleton and incidence certificate.*
8. *H records componentwise* hsame *transport for block sizes, legal decorations, defect exponents, product rows, specialization rows, incidence rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay through $M, S, D, E, F, Q, I, T, H$.*
10. *P is the* Pkg *provenance over* PartitionBasinPolynomial$^{\uparrow}$, Partition$^{\uparrow}$, DefectFlowPartitionSkeleton$^{\uparrow}$, PartitionCoreIncidenceCertificate$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a stochastic biology model, a host polynomial oracle, a quotient-semantic equality, an unbounded partition enumeration, or a hidden recurrence outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.PartitionBasinPolynomialUp.PartitionBasinPolynomialTasteGate single carrier alignment`

Theorem 3278.2 (PartitionBasinPolynomial NameCert obligations)**.** *For every accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *surface consists exactly of block-count and block-size admission through M, S, legal-decoration admission through D, defect-exponent admission through E, product-formula exposure through F, four-coordinate specialization through Q, incidence-total readback through I, defect-flow handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4363.1. The block-count and block-size rows supply the finite partition core on which the enumeration is read.

The decoration and defect rows are then exposed coordinatewise. For block i, the all-decoration contribution is $(1+q)^{(m-1)s_i}$, and the excluded no-private-coordinate contribution is $\left((1+q)^{m-1}-1\right)^{s_i}$.

Multiplying the legal block rows gives the product row F. The specialization row Q evaluates the same finite formula on the five four-coordinate partition types, and the incidence row I sums the displayed rows with their finite multiplicities.

Finally T, H, C, P, N route the same polynomial surface to the defect-flow and incidence packets. Because no excluded stochastic, host-oracle, quotient-semantic, unbounded-enumeration, or hidden-recurrence coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3278.3 (PartitionBasinPolynomial product specialization)**.** *Every defect-flow consumer of an accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier receives the product formula row for $\mathcal{B}_P(q)$ and the four-coordinate specialization row before reading quotient-level or codon-level incidence totals. The consumer sees only finite block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and local NameCert rows.*

Proof. Start from the obligations of Definition 4363.2. They force the block-size ledger and legal-decoration rows to be displayed before the product formula can be consumed.

The four-coordinate row is a finite specialization of the same product row. Its incidence totals are read only after the partition types and finite multiplicities have been displayed.

The handoff through T, H, C, P, N preserves the same finite surface for the defect-flow skeleton and partition-core incidence certificate. Thus no stochastic model, host generating-function oracle, quotient-semantic equality, or hidden enumeration can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$.

Constructive story: A PartitionBasinPolynomial$^{\uparrow}$ packet is generated from block-count rows, block-size ledgers, legal-decoration rows, defect-exponent ledgers, a product-formula row, a four-coordinate specialization row, incidence-total rows, a defect-flow handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 4363.1, Definition 4363.2, and Definition 4363.3 seed the finite partition-basin polynomial packet over displayed block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a stochastic biology model, host generating-function oracle, quotient-level semantic equality, unbounded partition enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-formula exposure, four-coordinate specialization, incidence readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionBasinPolynomialUp.

Chapter 3279

A Concrete Naming Certificate for $\mathsf{DistanceToSet}^{\uparrow}$

Following Bishop and Bridges on located subsets in metric spaces, the $\mathsf{DistanceToSet}^{\uparrow}$ packet records the finite route by which a point-to-set distance request is read through displayed located-radius rows and closed-ball witnesses. It sits after the metric source of chapter 57, the located-set surface of chapter 1541, the metric closed-ball boundary of chapter 1451, the located closed-ball handoff of chapter 4063, and the terminal real-name seal of chapter 38. The packet names distance-to-set as finite BEDC data; it is not a host infimum, not a selected nearest point, not quotient real equality, and not a countable-choice construction.

Definition 3279.1 (Distance-to-set carrier)**.** *A* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, x, r, B, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the subset by finite located-radius observations.*
3. *x is the displayed point row in the metric source.*
4. *r is the dyadic radius ledger recording the inspected tolerance windows.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness row attached to the point and radius ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance name, read only after the located-set and closed-ball rows are displayed.*
7. *H records componentwise* hsame *transport for the metric source, located set, point row, dyadic ledger, closed-ball witness, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the metric source through located-set exposure, point inspection, dyadic-radius accounting, closed-ball handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, x, r, B, R, H, C, P, N$. It has no coordinate for a host infimum over all points of A, a selected nearest point, quotient real equality, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DistanceToSetUp.DistanceToSetTasteGate single carrier alignment`

Theorem 3279.2 (Distance-to-set NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier*

$$D = (X, A, x, r, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located set, point row, dyadic radius ledger, closed-ball witness, Real seal, transport, replay, provenance, and local naming;*
2. *located-set exposure, requiring every subset read to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *row A before a distance row is exported;*
3. *dyadic-radius monotonicity, requiring the tolerance ledger r to preserve the displayed radius order used by the located-set and closed-ball rows;*
4. *closed-ball handoff, requiring each ball membership or exclusion read to pass through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness B and the located closed-ball sibling surface before the Real seal is read;*
5. *Real seal non-escape, requiring R to seal only the displayed distance name and not a host infimum or quotient real;*
6. *componentwise transport, displayed replay, package provenance, and local* NameCert *naming for exactly the displayed packet.*

No DistanceToSet obligation is stored outside these rows.

Proof. Project the carrier of Definition 4364.1. The metric source is X, the located subset is A, the inspected point is x, and the finite tolerance information is held by r.

The closed-ball witness B is read only after those rows are displayed. Its radius requests are therefore attached to the same metric source, located-set row, point row, and dyadic ledger.

The Real seal R comes after the located-radius and closed-ball rows. Transport H and replay C preserve the same order, while P and N record provenance and local naming.

Since no coordinate stores a host infimum, a nearest-point selector, quotient real equality, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 3279.3 (Distance-to-set located-radius handoff)**.** *Every consumer of an accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *packet reads distance-to-set data only through the displayed located-radius row A, the dyadic radius ledger r, the closed-ball witness B, and the Real seal R, transported by H and replayed by C. The consumer cannot bypass these rows by reading a host infimum, a selected nearest point, quotient real equality, countable choice, or an ambient completeness principle.*

Proof. Read the carrier of Definition 4364.1. The only subset coordinate is the located-set row A, and the only radius coordinate is the dyadic ledger r.

The closed-ball witness B supplies the displayed ball boundary before the Real seal R is exposed. Thus a distance consumer reaches the exported real name only through the located-radius and closed-ball rows.

Apply Definition 4364.2. The transport, replay, provenance, and local naming rows preserve the same finite packet and add no host infimum, nearest-point selector, quotient real equality, countable choice, or ambient completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}$.

Constructive story: A $\mathsf{DistanceToSet}^{\uparrow}$ packet is generated from a metric source, located set row, point row, dyadic radius ledger, closed-ball witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4364.1, Definition 4364.2, and Definition 4364.3 seed the finite located-radius and closed-ball interface for distance-to-set consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host infimum construction, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-set exposure, dyadic-radius monotonicity, closed-ball handoff, Real seal non-escape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DistanceToSetUp.

Chapter 3280

A Concrete Naming Certificate for $\mathsf{JordanCurve}^{\uparrow}$

Following Jordan and constructive topology, the $\mathsf{JordanCurve}^{\uparrow}$ packet records the finite polygonal and winding-ledger route by which a displayed simple closed curve in the real plane may be consumed as a separation boundary. It sits after the terminal real-name seal of chapter 38, the circle carrier of chapter 58, the graph and modulus rows of chapter 130, the finite compact-metric source of chapter 129, and the simplex inventory of chapter 238. The packet names a constructive finite obligation surface; it does not import classical open-cover compactness, selected complementary components, quotient path equality, or a choice-dependent separation theorem.

Definition 3280.1 (Jordan curve carrier)**.** *A* $\mathsf{JordanCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, F, P, W, O, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, recording the circle parameter ledger and its local classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row into the displayed* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ *plane, with graph and modulus observations visible.*
3. *P is the finite polygonal approximation row, read through a* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision and its vertex-image ledger.*
4. *W is the winding ledger, recording the finite edge-crossing and sign-change data used to compare test points with the polygonal boundary.*
5. *O is the orientation row, recording the compatible cyclic order inherited from the* $\mathsf{S}^{1\uparrow}$ *row and the polygonal subdivision.*
6. *I is the injectivity-apartness row, stating that distinct inspected arcs carry displayed positive separation windows before they are used by the boundary read.*
7. *E is the exterior-interior ledger, recording only the finite side classification certified by the current polygonal and winding rows.*
8. *H records componentwise* hsame *transport for $S, F, P, W, O, I, E, C, Q, N$.*
9. *C records displayed* Cont *replay from circle, map, polygonal, winding, orientation, and apartness rows to the side-classification row.*

10. *Q is the* Pkg *provenance over* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, P, W, O, I, E, H, C, Q, N$. It has no coordinate for a host Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or countable-choice separation.

Lean4 ✓ `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate single carrier alignment`

Theorem 3280.2 (Jordan curve orientation ledger)**.** *For every accepted* $\mathsf{JordanCurve}^{\uparrow}$ *carrier*

$$J = (S, F, P, W, O, I, E, H, C, Q, N),$$

the orientation row O is determined by the finite $\mathsf{S}^{1\uparrow}$ *cyclic order, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision in P, and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *edge-image ledger in F. The winding ledger W can read orientation only through those rows, transported by H and replayed by C.*

Proof. Project the accepted carrier. The only circle-facing coordinate is S, and the only finite boundary skeleton is the polygonal row P together with the displayed edge images in F.

The orientation row O is therefore read by following the cyclic order of S across the subdivision cells of P. Each edge-image comparison is a finite real-plane observation supplied by F, not by an ambient topological orientation principle.

The winding ledger W consumes that ordered edge list. Componentwise transport H and replay C preserve the same ordered rows, while Q and N record provenance and local naming. Since the carrier contains no host orientation object or quotient path coordinate, the finite ledger is the entire orientation source. □

Theorem 3280.3 (Jordan curve winding separation boundary)**.** *Every side-classification read from an accepted* $\mathsf{JordanCurve}^{\uparrow}$ *packet factors through the finite route*

$$S, F, P, O, I, W \longmapsto E \longmapsto H, C, Q, N.$$

Thus the packet exposes polygonal approximation, injectivity-apartness, orientation, and winding ledgers before it names an interior or exterior row. It does not assert a blanket classical Jordan separation theorem, selected complementary components, open-cover compactness, quotient path equality, or choice-dependent planar separation.

Proof. Read the carrier of Definition 4365.1. The curve source is S, the displayed map is F, and the finite polygonal boundary is P.

Before a side row is exported, the packet requires the orientation ledger O, injectivity-apartness row I, and winding ledger W. These rows are finite BHist data: the winding read uses edge crossings and signs already attached to the polygonal boundary.

The exterior-interior row E is downstream of that route. Transport, replay, provenance, and naming through H, C, Q, N preserve the same finite packet. The excluded host theorem, selected components, open covers, quotient paths, and choice rows are not coordinates of the carrier, so they cannot be read from the local NameCert surface. □

Falsifiable prediction: A $\mathsf{JordanCurve}^{\uparrow}$ consumer that exports side separation without reading P, W, O, I refutes the carrier surface.

Independence witness: real, sone, continuousmap, compactmetric, simplicialcomplex: the sibling chapters supply real-plane coordinates, circle source data, continuous graph and modulus rows, compact-metric finite inspection, and subdivision cells separately, while $\mathsf{JordanCurve}^{\uparrow}$ binds them into the finite packet $J = (S, F, P, W, O, I, E, H, C, Q, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$.

Constructive story: A $\mathsf{JordanCurve}^{\uparrow}$ packet is generated from a circle source row, continuous real-plane map row, finite polygonal subdivision, winding ledger, orientation row, injectivity-apartness row, side-classification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4365.1, Definition 4365.2, and Definition 4365.3 seed the finite polygonal, winding, orientation, apartness, side-classification, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, the classical Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or choice-dependent separation.

Upgrade path: Obligation closure requires checked carrier admission, polygonal approximation exactness, winding-ledger stability, orientation transport, injectivity-apartness handoff, side-classification non-escape, replay, provenance, and local naming for BEDC.Derived.JordanCurveUp.

Chapter 3281

A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$

The MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite NameCert data. It sits beside the candidate-set packet of chapter 2320 and the normalization router of chapter 1108: those siblings expose candidate evidence and route selection, while this packet records the boundary where empty-context closed typed terms may or may not enter the candidate interface. The packet names a closed-projection frontier only; it does not turn candidate-mediated normalization into an unconditional strong-normalization theorem for every closed MetaCIC term.

Definition 3281.1 (MetaCIC candidate closed-projection boundary carrier)**.** *A* MetaCICCandidateClosedProjection *carrier is a finite* BHist *packet*

$$B = (T, K, R, E, NF, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row whose projection into the candidate interface is being tested.*
2. *K is the candidate-set evidence row, read through* CandidateSetNormalization$^{\uparrow}$ *when the projection is admitted.*
3. *R is the router row, read through* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *and recording which candidate, closed-candidate, or candidate-set route is exposed.*
4. *E is the bounded-normal endpoint seal row, recording the finite normal-output evidence available after candidate admission.*
5. *NF is the no-infinite-reduction row exported only downstream of K, R, E.*
6. *O is the obstruction row recording refusal when the closed term cannot be placed into the displayed candidate interface.*
7. *H records componentwise* hsame *transport for $T, K, R, E, NF, O, C, P, L$.*
8. *C records displayed* Cont *replay from the closed-term row through candidate admission, router selection, endpoint seal, no-infinite-reduction read, and obstruction retention.*

9. *P is the* Pkg *provenance over* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$, CandidateSetNormalization$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, MetaCICOpenProblemLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, R, E, NF, O, H, C, P, L$. It has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Lean4 ✓ BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.MetaCICCandidateClosedProjectionBoundar
single carrier alignment

Theorem 3281.2 (MetaCIC candidate-set closed projection frontier)**.** *For every accepted* MetaCICCandidateClosed *carrier*

$$B = (T, K, R, E, NF, O, H, C, P, L),$$

the closed-term row T reaches a strong-normalization-facing read only through candidate evidence K, router selection R, and bounded-normal endpoint seal E. If those rows do not place T inside the candidate interface, the obstruction row O is the only exported boundary datum.

Proof. Project the carrier. The closed-term row T is present as a finite syntactic and typing packet, but it is not itself a normalization theorem.

The positive route first reads K through chapter 2320. The router row R then records the candidate, closed-candidate, or candidate-set branch through chapter 1108. Only after those rows are displayed can the endpoint seal E and no-infinite-reduction row NF be consumed.

If candidate admission is absent, O records the refusal. Transport, replay, provenance, and local naming through H, C, P, L preserve that same finite boundary, so the packet exports no evidence-free closed-term strong-normalization claim. □

Theorem 3281.3 (MetaCIC candidate-set SN handoff boundary)**.** *Every strong-normalization handoff exported by an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet is a handoff for rows already inside the displayed candidate interface. It cannot be read as a blanket theorem that all empty-context typed closed MetaCIC terms are strongly normalizing.*

Proof. Start with the frontier theorem. It places the normalization-facing route behind K, R, E and keeps the obstruction row O visible when the candidate interface is not supplied.

For an admitted row, the handoff consumes candidate evidence from CandidateSetNormalization$^{\uparrow}$ and route evidence from MetaCICNormalizationCertificateRouter$^{\uparrow}$. The no-infinite-reduction row NF is therefore downstream of the admitted candidate branch rather than a second source of normalization evidence.

For a closed term outside that interface, the carrier has only T and the obstruction row O on the boundary. The structural rows H, C, P, L transport, replay, source, and name the boundary, but they do not add subject reduction, host conversion equality, quotient reduction equality, or a global normalizer. Hence the handoff is candidate-scoped. □

Theorem 3281.4 (MetaCIC candidate-set public boundary)**.** *Any public no-infinite-reduction or strong-normalization read from an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet must cite candidate-set membership or soundness, router selection, and bounded-normal endpoint rows. A bare empty-context typed closed-term row T is insufficient: the public boundary may expose NF only downstream of the candidate-set facts named by* closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, *and* canonicalCandidateSet_sound *together with the router row R and endpoint seal E.*

Proof. Start with Definition 4366.2. It places every normalization-facing read behind the candidate evidence row K, router row R, and endpoint seal E, and leaves the obstruction row O as the only datum when candidate admission is absent.

Next apply Definition 4366.3. That theorem identifies the handoff as a candidate-interface handoff rather than a theorem about all empty-context closed typed rows. Thus the public NF row cannot be projected from T alone.

Finally read the MetaCIC normalization audit map through the existing candidate-set facts closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, and canonicalCandidateSet_sound. Those facts supply the candidate membership and soundness side of the route, while R and E supply the public router and endpoint rows. The resulting boundary is therefore candidate-mediated and evidence-bearing, not an unconditional closed-term normalizer. □

Falsifiable prediction: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ consumer that exports strong normalization from T without reading K, R, E refutes the carrier surface.

Independence witness: candidate-set-normalization, metacicnormalizationcertificaterouter, metacicopenproblemledger: the candidate-set packet carries candidate evidence, the router selects the admissible normalization route, and the open-problem ledger records the frontier question; this packet binds those rows into a closed-projection boundary without adding an evidence-free normalization theorem.

Closure status for NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$.

Constructive story: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate-set evidence, normalization-router selection, bounded-normal endpoint seal, no-infinite-reduction row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4366.1, Definition 4366.2, Definition 4366.3, and Definition 4366.4 seed the finite closed-projection, candidate-evidence, router-selection, endpoint-seal, obstruction, public-boundary, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional strong normalization for all closed typed MetaCIC terms, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Upgrade path: Obligation closure requires checked closed-term projection, candidate-evidence admission, router selection, bounded-normal endpoint sealing, no-infinite-reduction handoff, obstruction exactness, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.

Chapter 3282

A Concrete Naming Certificate for $\mathsf{PointedMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based metric spaces, the $\mathsf{PointedMetricSpace}^{\uparrow}$ packet records a metric carrier together with a named basepoint route before completion, loop-space, or embedding consumers are allowed to inspect point-level data. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, reads names through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, passes regular rational distance observations through chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and offers a based source for the distance-preserving consumer route of chapter 439. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected representatives, or ambient completeness.

Definition 3282.1 (Pointed metric space carrier)**.** *A* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, b, D, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *b is the basepoint name row, stored as finite BEDC data attached to M rather than as a host-level point.*
3. *D is the basepoint distance ledger, recording the finite requests that compare b with inspected metric names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which basepoint and metric names are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the basepoint-distance observations.*
7. *H records componentwise* hsame *transport for M, b, D, S, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a based metric request through the stream windows, regular rational readback, and real-distance seal.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{PointedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint name, distance ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate single carrier alignment`

Theorem 3282.2 (Pointed metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier* $P = (M, b, D, S, R, E, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint name, basepoint distance ledger, stream windows, regular rational readback, real-distance seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring every comparison of basepoint data to be read over the transported metric row* M*;*
3. *basepoint-distance exposure, requiring every basepoint comparison to pass through* D, S, R, E *before any completion-facing or embedding-facing consumer reads it;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 4367.1. The projection exposes the metric row, basepoint name, distance ledger, stream windows, regular readback, real seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, b, D, S, R, E together, so a consumer cannot compare a basepoint name over one metric carrier with distance observations over another.

The distance ledger fixes the read order. A basepoint comparison is first a finite request in D, then a scheduled stream observation in S, then a regular rational readback in R, and only then a $\mathsf{Real}^{\uparrow}$ distance seal in E.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected representatives, or ambient completeness, the listed rows exhaust the local naming surface. □

Theorem 3282.3 (Pointed metric space basepoint nonescape)**.** *Let* $P = (M, b, D, S, R, E, H, C, P_0, N)$ *be an accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier. Every completion-facing, loop-space-facing, or isometric-embedding-facing consumer of the basepoint row* b *must read it through the displayed route*

$$M, b, D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P_0, N.$$

Thus the basepoint is exported only as finite named metric data over M*, not as a host point, quotient representative, selected completion point, or ambient metric-equality witness.*

Proof. Begin with carrier admission. The only row naming the basepoint is b, and it is stored beside the metric carrier M and distance ledger D.

Read the inspection route next. The stream windows S and regular rational readback R display every observed distance from the basepoint before the $\mathsf{Real}^{\uparrow}$ seal E is available to downstream consumers.

Transport and replay preserve that same route. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add host points, quotient representatives, selected completion points, or ambient metric equality.

Therefore every based consumer receives exactly the finite route displayed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedMetricSpace}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a named basepoint row, a basepoint distance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4367.1, Definition 4367.2, and Definition 4367.3 seed the based metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host point equality, quotient metric equality, selected completion points, loop-space closure, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, basepoint-distance exposure, classifier transport, basepoint nonescape, and downstream completion and isometric-embedding consumer alignment for BEDC.Derived.PointedMetricSpaceUp.

Chapter 3283

A Concrete Naming Certificate for $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based complete metric spaces, the $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet records a based metric carrier together with a displayed completion service before completion, loop-space, or embedding consumers are allowed to read a point as complete. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service boundary of chapter 128, the $\mathsf{PointedMetricSpace}^{\uparrow}$ basepoint route of chapter 4367, the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected completion representatives, countable choice, or ambient order completeness.

Definition 3283.1 (Pointed complete metric space carrier)**.** *A* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, B, Q, S, R, E, L, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *B is the basepoint row, stored as finite named metric data over M rather than as a host-level point.*
3. *Q is the Cauchy-window route, listing the finite tail requests through which regular Cauchy names based at B are inspected.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the basepoint and Cauchy windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance and modulus observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed metric-distance and limit-distance observations.*
7. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion seal, exposing only the displayed limit-service handoff for the Cauchy-window route Q.*
8. *H records componentwise* hsame *transport for M, B, Q, S, R, E, L and the local certificate rows.*
9. *C records displayed* Cont *replay from a based complete-metric request through the Cauchy windows, stream observations, regular rational readback, real seal, and completion seal.*

10. *P_0 is the* Pkg *provenance over* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.TasteGate.PointedCompleteMetricSpaceTasteGate single carrier alignment`

Theorem 3283.2 (Pointed complete metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier* $P = (M, B, Q, S, R, E, L, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSp}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint row, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring the basepoint row, Cauchy windows, and completion seal to be read over the transported metric row M;*
3. *Cauchy-window exposure, requiring every completion-facing read to pass through Q, S, R, E, L before any downstream consumer may inspect the limit handoff;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, selected completion representative, countable-choice selector, or ambient order-completeness principle.

Proof. Project the carrier of Definition 4368.1. The projection exposes the metric row, basepoint, Cauchy-window route, stream windows, regular readback, real seal, completion seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, B, Q, S, R, E, L together, so a consumer cannot compare a based Cauchy route over one metric carrier with a completion seal over another.

The completion-facing read order is fixed by the route rows. A completion request begins as a finite Cauchy-window request in Q, becomes a scheduled stream observation in S, is read back as regular rational data in R, is sealed as real-distance data in E, and only then reaches the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service row L.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected completion representatives, countable-choice selectors, or ambient order completeness, the listed rows exhaust the local naming surface. □

Theorem 3283.3 (Pointed complete metric space Cauchy basepoint route)**.** *Let* $P = (M, B, Q, S, R, E, L, H, C, P_0$ *be an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier. Every based regular Cauchy request whose initial window is anchored at B is read through the displayed route*

$$M, B, Q \longmapsto S \longmapsto R \longmapsto E \longmapsto L \longmapsto H, C, P_0, N.$$

Thus the basepoint participates in completion only as named metric and Cauchy-window data over M, not as a selected host point or completed-limit representative.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.PointedCompleteMetricSpaceCauchyBasepointRoute`

Proof. Begin with carrier admission. The only row naming the basepoint is B, and it is stored beside the metric row M and the Cauchy-window route Q.

Read the window route next. The stream row S schedules the inspected windows, the regular rational row R reads the tail and distance ledgers, and the $\mathsf{Real}^{\uparrow}$ row E seals the observed distances before the complete-metric row L can expose any limit-service handoff.

Transport and replay preserve that same path. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add a host basepoint, a selected limit representative, quotient equality, or a separate completion object.

Therefore every based Cauchy consumer receives exactly the finite route displayed in the statement. □

Theorem 3283.4 (Pointed complete metric basepoint completion handoff)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ and $P' = (M', B', Q', S', R', E', L', H', C', P'_0, N')$ be accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carriers. If the two packets transport M, B, Q, S, R, E through the same componentwise H row, then every* $\mathsf{CompleteMetric}^{\uparrow}$ *service read of L is taken over that same named basepoint route before any loop-space or embedding consumer may inspect a completed point.*

Equivalently, a consumer may inspect

$$L \quad \text{or} \quad L'$$

only after the route

$$M, B, Q, S, R, E \longmapsto H, C, P_0, N$$

has been displayed; it cannot detach completion service from the transported basepoint row.

Proof. Project both carriers. In each packet the completion-service coordinate is L, and the only rows upstream of it are the metric row M, the basepoint row B, the Cauchy-window row Q, the stream row S, the regular rational row R, and the real-distance seal E.

Apply Definition 4368.3 to the first packet. The displayed route forces every based Cauchy request to pass through M, B, Q, S, R, E before the service row L is visible.

Use the assumed componentwise transport row H to move the same route to the second packet. The replay row C then reproduces the transported route for the completion-facing consumer, and the provenance and local name rows P_0, N record no independent completed point.

Thus a loop-space or embedding consumer can inspect a completed point only after the transported basepoint route has been displayed. There is no carrier coordinate from which the consumer could read L apart from B. □

Theorem 3283.5 (Pointed complete metric space completion nonescape)**.** *Every completion-facing, loop-space-facing, or embedding-facing consumer of an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *packet factors its completed-point read through*

$$M, B, Q, S, R, E, L, H, C, P_0, N.$$

Consequently the packet exports a based complete metric certificate only as finite BEDC NameCert data; it exports neither host points nor quotient completion equality.

Proof. Inspect the carrier coordinates. The only completion coordinate is L, and the definition makes L a completion seal for the already displayed Cauchy-window route Q over the metric row M and basepoint row B.

The readback rows precede the completion seal. Any consumer must first pass through S, R, E to display stream windows, regular rational ledgers, and real distance seals, so no completed point can be read before the finite analytic route is visible.

The remaining rows are administrative in the BEDC sense. Transport H, replay C, provenance P_0, and local name N preserve and source the packet but do not create host points, quotient completion equality, selected representatives, or ambient completeness.

Thus every downstream completed-point read factors through the displayed finite packet, which proves completion nonescape. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet is generated from a MetricUp carrier, a named basepoint row, a Cauchy-window route, StreamName windows, RegSeqRat readback, a RealUp seal, a CompleteMetricUp completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4368.1, Definition 4368.2, Definition 4368.3, and Definition 4368.5 seed the based complete-metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient completion equality, selected host endpoints, loop-space closure, embedding closure, or ambient order completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, classifier transport, basepoint completion routing, completion nonescape, and downstream completion, loop-space, and embedding consumer alignment for BEDC.Derived.PointedCompleteMetricSpaceUp.

Chapter 3284

A Concrete Naming Certificate for $\mathsf{IntervalConnectedness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{IntervalConnectedness}^{\uparrow}$ packet records the finite constructive route by which a located real interval is inspected as one connected interval surface before intermediate-value, zero-envelope, or completion-facing consumers use it. It sits after the endpoint seal of chapter 38, the located interval carrier of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sandwich consumer surface of chapter 2163. The packet names interval connectedness as displayed finite BEDC data; it does not import host topology, open-cover compactness, trichotomy, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3284.1 (Interval connectedness carrier)**.** *An* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint-pair row, storing the lower and upper endpoint names through their displayed real seals.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *witness row, carrying the located interval budget, locator reads, and endpoint-radius checks attached to E.*
3. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision-chain ledger, recording the dyadic cells, adjacent-cell overlaps, and tolerance requests inspected inside L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, cell, and witness observations are scheduled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled endpoint and subdivision observations.*
6. *T records componentwise* hsame *transport for endpoint pairs, located interval witnesses, dyadic subdivision chains, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint pairs through located interval exposure, dyadic subdivision, stream-window inspection, regular readback, and the interval-connectedness handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalConnectedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SqueezeTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, D, S, R, T, C, P, N$. It has no coordinate for host topological connectedness, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalConnectednessUp.IntervalConnectednessTasteGate single carrier alignment`

Theorem 3284.2 (Interval connectedness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier*

$$I = (E, L, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint pair, located interval witness, dyadic subdivision chain, stream windows, regular readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring endpoint, locatedness, subdivision, window, readback, replay, provenance, and naming data to move componentwise through T;*
3. *dyadic chain exactness, requiring every inspected subinterval and adjacent-cell overlap to be recorded in D before a connectedness consumer reads the packet;*
4. *endpoint handoff nonescape, requiring every consumer of interval endpoints to pass through E, L, D, S, R before using any* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger exhaustion, requiring the packet to expose no host topological connectedness theorem, open-cover compactness, trichotomy, quotient real equality, selected intermediate point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The sibling routes through $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{SqueezeTheorem}^{\uparrow}$ *are the only named consumers of this local certificate surface.*

Proof. Project the carrier of Definition 4369.1. The endpoint pair is visible only in E, and the located interval evidence is visible only in L.

The subdivision data are then forced through D. Each finite cell, overlap, and tolerance request is read as dyadic data before any stream-window or regular-readback consumer can inspect it.

Transport and replay preserve the same route. The rows T and C move endpoint, locatedness, subdivision, window, and readback data componentwise, while P and N source and name exactly that displayed packet.

Since the carrier has no coordinate for the excluded host principles, the local certificate cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{IntervalConnectedness}^{\uparrow}$ consumer reads endpoint pairs, located interval witnesses, dyadic subdivision chains, StreamName windows, RegSeqRat readbacks, hsame transport, Cont replay, Pkg provenance, and local naming before any interval-connectedness handoff is visible.

Independence witness: real, locatedrealinterval, dyadicratcore, streamname, regseqrat, squeezetheorem: $\mathsf{IntervalConnectedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, located interval witnesses, finite dyadic subdivision chains, scheduled stream windows, regular rational readbacks, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$.

Constructive story: An IntervalConnectedness$^{\uparrow}$ packet is generated from Real endpoint pairs, a LocatedRealInterval witness, a finite DyadicRatCore subdivision chain, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, seed)

Scope closed: Definition 4369.1 and Definition 4369.2 seed the finite located-interval connectedness route over Real$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and SqueezeTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topology, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic chain exactness, endpoint handoff nonescape, ledger exhaustion, and downstream intermediate-value consumer alignment for BEDC.Derived.IntervalConnectedness

Chapter 3285

A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$

Following Tait–Martin-Lof and Barendregt on residuals, parallel reduction, and confluence discipline, the MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ packet records the finite boundary where a beta-parallel branch and a substitution replay must meet before the MetaCIC confluence route may use the conditional diamond bridge. It sits over the audit map of chapter 9378 and names the residual-substitution input as packet data rather than as a global Church–Rosser theorem. The packet does not claim full closed Church–Rosser, arbitrary closed-source confluence, or an unrestricted substitution theorem.

The residual-substitution packet is consumed by chapter 765 as a guarded input to candidate-mediated SN/confluence scheduling. The dependency is the residual join row together with the closedness guard data: it tells the scheduler which finite substitution replay may enter a candidate branch and which source windows are closed enough for that handoff, not that the full Church–Rosser theorem has already been completed.

Definition 3285.1 (MetaCIC residual-substitution compatibility carrier)**.** *A* MetaCICResidualSubstitutionCompatil *carrier is a finite* BHist *packet*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

with the following displayed rows:

1. *s is the source term row whose beta-parallel residual branch is inspected.*
2. *t is the substituted term row, storing the finite substitution request and the target occurrence window.*
3. *b is the parallel branch row, displaying the left and right beta-parallel steps before any join is named.*
4. *j is the residual join row, recording the candidate common reduct obtained after substitution replay and residual transport.*
5. *g is the closedness guard row, saying exactly which source and substitution windows are closed enough for the guarded handoff.*
6. *a is the atom-shape separation row, keeping atom, application, lambda, Pi, and beta-redex cases visibly disjoint.*
7. *h records componentwise* hsame *transport for source, substitution, branch, residual join, closedness guard, atom-shape separation, replay, provenance, and local naming rows.*

8. *c records displayed* Cont *replay from the source term through substitution, parallel branching, residual join construction, closedness guarding, and atom-shape separation.*
9. *p is the* Pkg *provenance over* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$*, MetaCIC beta-parallel reduction, residual joins, closedness guards, atom-shape separation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *n is the local* NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, t, b, j, g, a, h, c, p, n$. It has no coordinate for a completed parallel diamond, full closed Church–Rosser, arbitrary closed-source confluence, or an unguarded substitution principle.

Lean4 ✓ BEDC.Derived.MetaCICResidualSubstitutionCompatibilityUp.MetaCICResidualSubstitutionCompatibilit
single carrier alignment

Theorem 3285.2 (MetaCIC residual-substitution compatibility NameCert obligations)**.** *For every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the local NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and local naming rows;*
2. *branch-shape disjointness, requiring atom, application, lambda, Pi, and beta-redex branches to be separated by a before a residual join row is accepted;*
3. *substitution residual replay, requiring c to display how the substituted term row and beta-parallel branch row reach the same candidate join j;*
4. *closed-source guard, requiring g to expose the exact closedness assumptions consumed by the guarded compatibility packet;*
5. *non-claim of full closed Church–Rosser, requiring the packet to remain an input boundary for the conditional confluence bridge rather than a proof of arbitrary closed-source confluence.*

No obligation is stored outside $s, t, b, j, g, a, h, c, p, n$.

Proof. Project the carrier of Definition 4370.1. The source term, substituted term, branch, residual join, closedness guard, and atom-shape separation rows are the only computational coordinates.

The branch-shape row a fixes the case split before the join row is read. Thus a beta-redex residual branch cannot be silently identified with an atom, application, lambda, or Pi shape.

Replay then supplies the compatibility boundary. The row c records the finite route from substitution and beta-parallel branching to the candidate join j, while g states the closedness guard consumed by that route.

Finally h, p, n transport, source, and name exactly the displayed packet. Since full closed Church–Rosser and arbitrary closed-source confluence are not carrier coordinates, they are not exported by this NameCert surface. □

Theorem 3285.3 (MetaCIC residual-substitution candidate projection route)**.** *Every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

exports a candidate-closed projection route to chapter 4366 only after the source term row s, substituted term row t, beta-parallel branch row b, residual join row j, closedness guard row g, and atom-shape separation row a have all been displayed. The downstream candidate boundary receives a guarded compatibility packet, not full Church–Rosser and not arbitrary closed-source confluence.

Proof. Project the carrier of Definition 4370.1. The rows s, t, b, j, g, a are the only computational rows available before the structural transport, replay, provenance, and local naming rows are read.

Use Definition 4370.2. Branch-shape disjointness exposes a, substitution residual replay exposes the route from t and b to j, and the closed-source guard exposes exactly the assumptions in g.

The candidate boundary of chapter 4366 is therefore reached through a guarded compatibility packet. The route supplies source, substitution, beta-parallel, residual-join, closedness, and atom-shape data, while h, c, p, n only transport, replay, source, and name those rows.

Since completed Church–Rosser and arbitrary closed-source confluence are absent from the carrier, the candidate-closed projection route cannot export them. It hands off only the displayed residual-substitution compatibility boundary. □

Theorem 3285.4 (MetaCIC residual-substitution obligation closure surface)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the obligation surface is exactly the displayed source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and naming rows. The candidate-closed projection boundary receives only this guarded compatibility packet.

Proof. Apply Definition 4370.2. It identifies carrier admission, branch-shape disjointness, substitution residual replay, closed-source guarding, transport, replay, provenance, and naming as rows of $s, t, b, j, g, a, h, c, p, n$.

Apply Definition 4370.3. The candidate boundary receives the source, substitution, beta-parallel branch, residual-join, closedness, and atom-shape data only through that guarded projection route.

Therefore the obligation surface packages the candidate handoff without exporting full closed Church–Rosser, arbitrary closed-source confluence, or unguarded substitution compatibility. Those claims are not coordinates of the carrier and are not supplied by the candidate projection. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ read displays source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, hsame transport, Cont replay, Pkg provenance, and local naming rows before the conditional MetaCIC confluence bridge can consume it.

Independence witness: metacic-confluence-audit: $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ is not a completed confluence theorem because it carries only the finite residual-substitution compatibility boundary consumed by the conditional parallel-diamond route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ packet is generated from source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4370.1, Definition 4370.2, and Definition 4370.3 seed the finite residual-substitution compatibility boundary consumed by the conditional MetaCIC confluence bridge and the candidate-closed projection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, arbitrary closed-source confluence, unguarded substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-shape disjointness, substitution residual replay, closed-source guard, and conditional confluence handoff alignment for BEDC.MetaCIC.BetaParallelSubstitutionCompatibility.

Chapter 3286

A Concrete Naming Certificate for $\mathsf{ComputableCompactMetric}^{\uparrow}$

Following Weihrauch-style computable analysis and the Bishop–Bridges compact metric arguments, the $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet records the effective compact-metric surface as finite BEDC NameCert data. It sits after the compact metric carrier of chapter 129, the effective Polish source of chapter 1706, the total-bounded finite-net route of chapter 127, the complete metric source of chapter 128, and the L10 readback route through chapter 126, chapter 301, and chapter 38. The packet names the finite-cover realizer surface consumed by effective compact analysis; it does not assert open-cover compactness, selected subcovers, quotient completion equality, or an ambient real-completeness principle.

Definition 3286.1 (Computable compact metric carrier)**.** *A* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact metric packet whose finite-net behavior is being made effective.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *source row exposing the effective dense-name and semidecidable basic-ball interface.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row naming the finite-net source for each displayed dyadic tolerance request.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row recording the convergence service available to Cauchy and compact consumers.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for inspected dense names, finite net centers, tolerance requests, and cover indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of distances and tolerance comparisons.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows and regular rational comparisons have been displayed.*

8. *H records componentwise* hsame *transport for compact metric, effective Polish, total-bounded, complete-metric, stream-window, regular-readback, real-seal, finite-cover realizer, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from an effective compactness request through K, E, T, C, S, R, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ComputableCompactMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, T, C, S, R, L, H, Q, P, N$. It has no coordinate for an open-cover subcover, a choice-selected finite set, quotient completion equality, excluded-middle compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ComputableCompactMetricUp.ComputableCompactMetricTasteGate single carrier alignment`

Theorem 3286.2 (Computable compact metric NameCert obligations)**.** *For every accepted* ComputableCompactMe *carrier*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *surface consists exactly of the displayed rows in Definition 4371.1. Carrier admission must expose the compact metric source, effective Polish source, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through those same rows, and ledger exhaustion says that every finite-cover or compact-analysis consumer can inspect only $K, E, T, C, S, R, L, H, Q, P, N$.*

Proof. Project the carrier of Definition 4371.1. The rows K, E, T, C are the only structural compactness sources, so carrier admission exposes compact metric data, effective Polish naming data, total-bounded finite nets, and complete-metric convergence before any realizer can be read.

The readback rows S, R, L then bind numerical access to finite windows, regular rational comparisons, and the terminal real seal. This is the same L10 readback discipline used by the StreamName, RegSeqRat, and Real siblings named in the chapter introduction.

Transport and replay are visible rows rather than hidden meta-operations. The row H records componentwise hsame transport, Q records displayed Cont replay, and P, N source and name the same finite packet.

For ledger exhaustion, read any compact-analysis consumer from the accepted packet. The carrier exposes only $K, E, T, C, S, R, L, H, Q, P, N$, so no open-cover subcover, selected finite set, quotient completion equality, excluded-middle compactness, or ambient real-completeness principle can be obtained from this NameCert surface. □

Theorem 3286.3 (Computable compact metric finite-cover realizer)**.** *Every effective finite-cover read exported by an accepted* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *packet factors through the route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsa

Thus a consumer receives a compact metric source, effective dense-name interface, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational tolerance comparisons, terminal real seal, componentwise transport, replay, provenance, and local

naming rows before it uses the finite-cover realizer. The packet exports no selected subcover, open-cover compactness theorem, quotient completion representative, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an accepted carrier $K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$. The compact metric row K and effective Polish row E name the metric and effective dense-name surfaces, while T supplies the finite-net source used for the requested tolerance.

The convergence row C keeps the compact route compatible with complete-metric consumers. A finite-cover realizer cannot leave these displayed sources, because the later rows are read only after the compact, effective, total-bounded, and complete rows are present.

The numerical readback is finite. The stream row S exposes inspected names, center windows, cover indices, and tolerance requests; the regular rational row R compares the displayed distances; and the terminal seal L is reached only after those windows and comparisons have been read.

Finally H, Q, P, N preserve the route under transport, replay, provenance, and local naming. Since selected subcovers, open-cover compactness theorems, quotient representatives, choice selectors, and ambient completeness witnesses are not carrier coordinates, the exported realizer is exactly the displayed finite-cover surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet is generated from CompactMetric, EffectivePolishSpace, TotallyBounded, CompleteMetric, StreamName, RegSeqRat, RealUp, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4371.1, Definition 4371.2, and Definition 4371.3 seed the effective finite-cover realizer route for computable compact metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, open-cover compactness, quotient completion equality, countable-choice selectors, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective dense-name exposure, total-bounded finite-net exposure, complete-metric convergence exposure, finite-window readback, regular rational comparison, real sealing, finite-cover realizer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ComputableCompactMetricUp.

Chapter 3287

A Concrete Naming Certificate for $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet records the separated-completion reflection route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the regular rational readback surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness boundary of chapter 3390. The packet lets a Real-facing consumer treat two representative regular Cauchy names as the same separated reflection point only after the displayed finite windows, dyadic budgets, and uniqueness ledger have been exposed.

Definition 3287.1 (Regular-Cauchy Hausdorff-reflection carrier)**.** *A* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row presented to the same separated-reflection request.*
3. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row witnessing that the displayed tail comparisons force separated agreement.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, Y, S, T, D, U have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, stream windows, dyadic ledger, uniqueness row, Real seal, and local certificate rows.*
8. *C records the displayed* Cont *replay from a reflection request to the paired windows, dyadic tolerance ledger, uniqueness row, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier admission surface`

Theorem 3287.2 (Regular-Cauchy Hausdorff-reflection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier* $R = (X, Y, S, T, D, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *separated-agreement row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4372.1. The only displayed coordinates are $X, Y, S, T, D, U, E, H, C, P, N$.

The two regular rational names are compared through the paired stream windows S, T and the dyadic tolerance ledger D. The separated-agreement row U is therefore a displayed uniqueness boundary, not a quotient of streams or a host equality proof.

The terminal Real seal E is downstream of that boundary. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and no excluded quotient, selector, host-equality, choice, trichotomy, or ambient-completeness coordinate can be projected from the carrier. □

Theorem 3287.3 (Regular-Cauchy Hausdorff-reflection representative independence)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet is independent of the chosen representative regular Cauchy name in the following BEDC sense: if the two presented* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are compared through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the terminal seal can distinguish them only through the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row. The packet exports no quotient stream equality, selected representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionRepresentativel`

Proof. Begin with an accepted carrier $R = (X, Y, S, T, D, U, E, H, C, P, N)$. The rows X and Y are not presented as completed real objects; they are presented as regular rational names observed through finite stream windows.

The dyadic ledger D binds the comparison budget for those windows, and the uniqueness row U is the only row that can turn the displayed tail comparison into separated agreement. Thus any Real-facing read of E must pass through the same finite comparison route.

The rows H, C, P, N transport, replay, source, and name this route without adding analytic content. Since the carrier contains no further equality coordinate, changing representatives cannot be detected except through the displayed uniqueness row. □

Theorem 3287.4 (Regular-Cauchy Hausdorff-reflection Real seal non-escape)**.** *No terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the separated reflection supplies representative independence for the Real seal without quotient streams, host real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier real seal nonescape`

Proof. Read the carrier coordinates in order. The terminal row E appears only after the two regular rational names, paired stream windows, dyadic tolerance ledger, and uniqueness row have been displayed.

Apply Definition 4372.2. The local NameCert surface exposes exactly those rows, with H, C, P, N preserving the same route for downstream consumers.

The advertised route is therefore the only public path to the Real seal. Since no excluded equality or choice coordinate occurs in the carrier, the seal cannot escape through quotient streams, host equality, representatives selected outside the packet, countable choice, trichotomy, or ambient completeness. □

Theorem 3287.5 (Regular-Cauchy Hausdorff-reflection finite-window uniqueness route)**.** *Suppose two accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packets display the same paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then every separated-reflection consumer factors through the same finite-window route on the underlying* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization does not use quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier finite window uniqueness route`

Proof. Apply Definition 4372.3. Once the paired StreamName windows, dyadic tolerance ledger, and uniqueness row agree, the terminal Real seal cannot distinguish the two regular Cauchy representatives by any route other than the displayed uniqueness row.

Apply Definition 4372.4. The Real seal is downstream of the finite windows, dyadic ledger, and uniqueness row, so a separated-reflection consumer has no bypass around those displayed rows.

The remaining rows are structural. The hsame rows transport only the displayed components, the Cont rows replay the same finite-window request, and the Pkg and NameCert rows name that route inside BHist. Therefore the consumer factors through the same finite-window route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore tolerance ledger, and a RegularCauchyUniqueness row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyuniqueness: $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational representatives to one separated uniqueness row and one downstream Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyUniqueness separated-agreement row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4372.1, Definition 4372.2, Definition 4372.3, Definition 4372.4, and Definition 4372.5 seed the separated reflection route for regular Cauchy representatives and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, representative-independence handoff, and Real seal non-escape theorem.

Chapter 3288

A Concrete Naming Certificate for $\mathsf{WeakLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on finite averages and probability estimates, the WeakLawLargeNumbers packet names the displayed finite-average route behind the weak law of large numbers. It sits after the normalized probability surface of chapter 184, the measurable random-variable rows of chapter 185, the finite tail-bound carrier of chapter 6592, and the terminal real-name seal of chapter 38. The packet is a finite certificate for Chebyshev-controlled averages; it is not an almost-sure convergence theorem, a central limit theorem, a countable-choice subsequence selector, a host probability-limit principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 3288.1 (Weak law finite-average carrier)**.** *A* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, R, A, V, T, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the total-event normalization and event classifier. The row R is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over that same probability source. The row A records the displayed finite average of the entries in R, while V records the variance or uniform second-moment bound used by the estimate. The row T reads the Chebyshev tail row through* $\mathsf{ChebyshevInequality}^{\uparrow}$ *for the event where the finite average differs from its displayed center by more than the requested tolerance. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the final probability bound. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite averaging route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these ten rows.*

Lean4 ✓ `BEDC.Derived.WeakLawLargeNumbersUp.WeakLawLargeNumbersTasteGate single carrier alignment`

Theorem 3288.2 (Weak law NameCert obligations)**.** *For every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier $(P, R, A, V, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 4373.1. Carrier admission displays the probability source, the random-variable family, the finite-average row, the variance-bound row, the Chebyshev tail row, the epsilon seal, the transport and replay rows, the provenance row, and the local naming row. The packet imports no host law of large numbers, selected subsequence, external probability-limit operation, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The probability source P supplies the measurable-event classifier and total-mass normalization, and the random-variable row R exposes only the finitely displayed variables over that source.

The average row A and variance row V are then internal finite data. They record the algebraic averaging expression and the bound used by the tail estimate before any limiting statement is visible.

Finally read T, E, H, C, Q, N. The Chebyshev tail row and epsilon seal name the probability estimate at the requested tolerance, while transport, replay, provenance, and local naming keep the same finite route under classifier replacement. Since no carrier coordinate stores an infinite limit or subsequence selector, the listed rows exhaust the local certificate surface. □

Theorem 3288.3 (Weak law Chebyshev handoff)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *Chebyshev read factors through the route*

$$P \longmapsto R \longmapsto A \longmapsto V \longmapsto T \longmapsto E \longmapsto H, C, Q, N$$

before any probability-bound consumer receives the finite-average estimate. Thus the Chebyshev step is handed to later probability and $\mathsf{Real}^{\uparrow}$ *formalization as a displayed finite source, finite random-variable row, average row, variance-bound row, tail-event row, epsilon seal, transport, replay, provenance, and local naming data.*

Proof. Start with a probability-bound request. The only probability source exposed by the packet is P, and the only random-variable data exposed over it are the finite rows in R.

The request must next read the average and variance-bound rows. The average row A fixes the finite expression being estimated, and V fixes the bound that the Chebyshev tail row may consume.

Only after these rows are displayed can T and E be read as a bound on the exceptional event. The structural rows H, C, Q, N then preserve that same route for downstream consumers, so the handoff is the finite Chebyshev estimate itself rather than a host-side convergence theorem. □

Theorem 3288.4 (Weak law finite-average nonescape)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *probability-limit-facing read remains inside the finite-average route of Definition 4373.1. In particular, a consumer can read a tail estimate only after displaying the source probability row, finite random-variable rows, average row, variance-bound row, Chebyshev tail row,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by almost-sure convergence, central limit normalization, countable-choice subsequence selection, a host probability-limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a consumer request the finite-average estimate from an accepted carrier. Projection of Definition 4373.1 shows that the request first meets P, R, A, V, T, E and has no independent coordinate for a limit object.

The displayed Chebyshev row T is downstream from the variance-bound row V and upstream from the epsilon seal E. Hence any probability bound read is already tied to the visible finite average and its tolerance.

The remaining rows H, C, Q, N do not add new analytic content. They transport equal displayed histories, replay the finite route, record the package provenance, and attach the local name. Therefore every accepted probability-limit-facing read stays inside the finite-average certificate surface and cannot escape to the excluded convergence or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet is generated from a probability source row, finite random-variable rows, a finite-average row, a variance or second-moment bound row, a Chebyshev tail row, a RealUp epsilon seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4373.1, Definition 4373.2, Definition 4373.3, and Definition 4373.4 seed the finite-average probability route through ProbSpace, RandomVar, RealUp, Chebyshev tail, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the central limit theorem, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-average routing, variance-bound stability, Chebyshev tail handoff, epsilon-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.WeakLawLargeNumbe

Chapter 3289

A Concrete Naming Certificate for $\mathsf{LocatedSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric and real spaces, the $\mathsf{LocatedSubset}^{\uparrow}$ packet records a displayed subset together with finite distance-witness rows. It sits after the real-valued metric route of chapter 1638, the Hausdorff comparison boundary of chapter 2086, the real-separability window surface of chapter 1809, the dyadic tolerance core of chapter 291, and the regular rational readback surface of chapter 301. The packet gives Real and Completion consumers a finite located-subset interface without host set extensionality, selected nearest points, quotient subset equality, or ambient completeness.

The located-subset distance route consumes chapter 1809 as its finite search-window source and consumes chapter 2086 only after the real-distance seal has been displayed.

Definition 3289.1 (LocatedSubset carrier)**.** *A* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, W, D, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *source row, carrying the ambient metric distance through finite real-name observations.*
2. *S is the subset-admission row, naming the displayed members read by the carrier and exporting no host powerset equality.*
3. *W is the* $\mathsf{RealSeparability}^{\uparrow}$ *search-window row, recording the countable dense or finite-window probes available to a distance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to each positive distance query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison data produced from W and D.*
6. *E is the real distance seal row, read only after the metric source, subset admission, search windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *K is the optional* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison boundary for compact or hyperspace consumers; it records only the finite directed comparisons visible through the preceding rows.*
8. *H records componentwise* hsame *transport for the metric source, subset admission, search windows, dyadic ledger, regular readback, real seal, Hausdorff boundary, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an ambient point and tolerance request through subset admission, separability windows, dyadic comparison, regular rational readback, real sealing, and any Hausdorff-facing comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSubset}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, W, D, R, E, K, H, C, P, N$. It has no coordinate for a host subset object, extensional equality of subsets, a selected nearest point, quotient real equality, countable choice, or an ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate single carrier alignment`

Theorem 3289.2 (LocatedSubset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, S, W, D, R, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, optional Hausdorff comparison routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4374.1. The displayed packet lists exactly the rows used by an ambient point, subset-membership, tolerance, rational-readback, real-seal, or Hausdorff-facing request.

The metric and subset rows M, S are read first. A distance query can inspect the ambient space only through $\mathsf{RealMetric}^{\uparrow}$, and it can inspect membership only through the displayed admission row S.

The search, tolerance, and readback rows W, D, R then supply finite witnesses for the locatedness request. The real seal E is reached only after those rows have been displayed, while K is available only as a downstream comparison boundary.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host subset, selected nearest point, quotient equality, countable-choice schedule, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3289.3 (LocatedSubset dyadic distance witness)**.** *Every distance read exported by an accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto E.$$

The read may be handed to a $\mathsf{HausdorffMetric}^{\uparrow}$ *consumer only after the same route has produced the displayed real-distance seal, transported by H and replayed by C.*

Proof. Begin with a point-to-subset distance request. The ambient metric row M supplies the only metric observations, and the subset-admission row S supplies the only visible membership data.

The finite search window W names the inspected probes. The dyadic ledger D fixes the positive tolerance requests, and the regular rational row R records the resulting readback windows.

The real-distance seal E is therefore downstream of M, S, W, D, R. Any Hausdorff-facing row K can compare only the finite distance data already sealed by this route.

Transport and replay do not add another distance object. The componentwise hsame row H, the displayed Cont route C, the Pkg provenance P, and the local NameCert row N preserve the

same witness surface, so no nearest-point selector or host infimum can bypass the dyadic distance route. □

Falsifiable prediction: Every accepted LocatedSubsetUp distance read exposes RealMetricUp, subset admission, RealSeparabilityUp windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, and a real-distance seal before any HausdorffMetricUp or completion-facing consumer reads the packet.

Independence witness: realmetric, hausdorffmetric, realseparability, dyadicratcore, regseqrat: none of these siblings alone carries the full located-subset packet with subset admission, finite search windows, dyadic tolerance rows, regular rational readback, real-distance sealing, optional Hausdorff comparison, transport, replay, package provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}$.

Constructive story: A $\mathsf{LocatedSubset}^{\uparrow}$ packet is generated from a RealMetricUp source, displayed subset admission, RealSeparabilityUp search windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a real-distance seal, optional HausdorffMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4374.1, Definition 4374.2, and Definition 4374.3 seed the finite located-subset interface for real-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host subset extensionality, selected nearest points, quotient subset equality, countable choice, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, Hausdorff comparison routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSubsetUp.

Chapter 3290

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 3290.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 3290.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 3290.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 3290.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 3290.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 3290.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 3291

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 3291.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 3291.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 3291.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 3292

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3292.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 3292.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 3292.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 3293

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 3293.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 3293.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 3293.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 3293.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 3293.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 3293.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

Chapter 3294

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 3294.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 3294.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3294.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 3294.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\text{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

The DedekindCutStream$^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 3295

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 3295.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3295.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 3295.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 3295.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 3296

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 3296.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 3296.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 3296.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 3296.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

Chapter 3297

A Concrete Naming Certificate for $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$

Following Bishop and Bridges on constructive inverse and implicit function reasoning, the ConstructiveImplicitFunctio packet records a local solution graph as finite BEDC data rather than as a host-level selection of a function. It sits between the real-name surface of chapter 38, the metric ball and interval windows of chapter 1450, chapter 1638, and chapter 1709, the continuous-map rows of chapter 130, and the derivative boundary of chapter 122. The certificate exposes the local graph, nondegenerate derivative row, located window, contraction or modulus replay, Real readback, transport, provenance, and local naming rows before any consumer may read a local solve.

Definition 3297.1 (Constructive implicit function carrier)**.** *A* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, J, A, M, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Real}^{\uparrow}$ *graph row for the equation and its local output coordinate, read only through displayed real-name endpoints.*
2. *J is the derivative or Jacobian row, including the apartness or nondegeneracy witness used by the local solve.*
3. *A is the located interval or metric-ball window row, naming the input cell, output cell, and base point through* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{RealMetric}^{\uparrow}$*, or* $\mathsf{MetricBall}^{\uparrow}$ *data.*
4. *M is the contraction, Lipschitz, or continuity-modulus replay row that bounds the local correction inside A.*
5. *I is the terminal* $\mathsf{Real}^{\uparrow}$ *readback and seal row for the solved graph point.*
6. *H records componentwise* hsame *transport for graph rows, derivative rows, apartness witnesses, located windows, modulus rows, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from F, J, A, M, I to the local solve row, including the* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *reads used by the graph.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{RealMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host implicit function, a selected germ, quotient function equality, global inverse theorem, ambient completeness principle, or classical choice.

Lean4 ✓ `BEDC.Derived.ConstructiveImplicitFunctionUp.ConstructiveImplicitFunctionTasteGate`
`single carrier alignment`

Theorem 3297.2 (Constructive implicit function NameCert obligations)**.** *For every accepted carrier $U = (F, J, A, M, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the displayed graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local name rows;*
2. *local graph exactness, requiring every solved point read to pass through the graph row F and terminal Real seal I;*
3. *apartness or nondegeneracy stability, requiring J to transport its derivative witness through H without replacing it by host equality;*
4. *contraction or modulus replay, requiring M and C to keep correction steps inside the located window A;*
5. *uniqueness nonescape, requiring any two local solve reads with the same displayed rows to return to the same $F, J, A, M, I, H, C, P, N$ packet.*

No obligation is stored in a detached function object or an ambient completeness theorem.

Proof. Carrier admission is projection from Definition 4277.1. The packet lists exactly the graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local name rows.

The analytic obligations are local to the displayed rows. The graph exactness read uses F and I, while derivative stability uses J together with the componentwise transports in H.

The local solve is kept inside the named window by A, M, C. Since the carrier has no host implicit-function coordinate, selected germ, quotient function equality, or ambient completeness row, uniqueness is a nonescape statement about the same finite packet. □

Theorem 3297.3 (Constructive implicit function local solve route)**.** *Every* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ *local-solve read factors through the displayed finite route*

$$\mathsf{ContinuousMap}^{\uparrow}, \mathsf{Derivative}^{\uparrow}, \mathsf{LocatedRealInterval}^{\uparrow}, \mathsf{MetricBall}^{\uparrow}, \mathsf{RealMetric}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

The consumer must open the graph row F, derivative or Jacobian row J, located window A, contraction or modulus row M, Real seal I, transport row H, replay row C, provenance row P, and local name row N before it can export a solved graph point.

Proof. Project the accepted carrier. The local-solve read has no available coordinate except $F, J, A, M, I, H, C, P, N$.

The graph and derivative rows are the analytic source. The located interval or metric-ball row names the window, and the contraction or modulus row records why the replay stays in that window.

Finally the Real seal and structural rows finish the route. The solve is therefore a displayed finite certificate over the named sibling chapters, not a host implicit function, selected germ, quotient of local functions, or classical existence principle. □

Falsifiable prediction: Every $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ local-solve read opens a displayed Real graph row, a derivative or Jacobian apartness row, a located interval or metric-ball row, a contraction or modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before exporting a solved graph point.

Independence witness: real, metricball, continuousmap, derivative, located-real-interval, real-metric: $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local graph exactness, derivative nondegeneracy, located-window control, modulus replay, Real sealing, transport, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ packet is generated from a displayed Real graph row, a derivative or Jacobian row with apartness data, a located interval or metric-ball window, a contraction or modulus replay row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4277.1, Definition 4277.2, and Definition 4277.3 seed the finite constructive implicit-function certificate over graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host implicit functions, selected germs, quotient function equality, global inverse or implicit theorem packages, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph exactness, apartness stability, modulus replay, uniqueness nonescape, and local naming for BEDC.Derived.ConstructiveImplicitFunctionUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 3298

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 3298.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3298.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 3298.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 3298.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 3298.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 3298.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 3298.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 3298.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 3298.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 3298.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 3298.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 3298.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 3298.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 3298.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 3298.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 3299

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 3299.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3299.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 3299.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 3299.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 3299.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 3299.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 3299.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 3299.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 3299.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 3299.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 3300

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 3300.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 3300.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 3300.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let $L = (S, Q, I, U, F, K, T, H$ be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

Chapter 3301

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 3301.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 3301.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 3301.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 3301.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 3302

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 3302.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 3302.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 3302.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3303

A Concrete Naming Certificate for $\mathsf{CauchyCompletionRoute}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyCompletionRoute}^{\uparrow}$ packet records the regular-Cauchy completion path as finite NameCert data rather than as a scattered analytic slogan. It reads the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational sequence surface of chapter 301, the terminal real-name seal of chapter 38, and the Bishop regular-Cauchy completion packet of chapter 1639. The route names exactly the L10 path by which finite dyadic observations become a completion-facing real-name handoff; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, identify quotient streams, select limits, or import countable choice.

Definition 3303.1 (Cauchy completion route carrier)**.** *A* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, E, B, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording endpoints, dyadic comparisons, precision requests, and the finite gap ledger used by the route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing only finite windows of the source stream before any regular-Cauchy read is admitted.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the rational sequence data and tail modulus visible through D and S.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint row, read only as the terminal seal produced from the displayed regular-Cauchy data.*
5. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *route row, tying D, S, Q, E to the established regular-Cauchy completion packet.*
6. *T records componentwise* hsame *transport for dyadic tolerances, stream windows, regular rational readback, terminal real seal, Bishop completion route, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a completion request through D, S, Q, B and then to E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionRouteUp.CauchyCompletionRouteTasteGate single carrier alignment`

Theorem 3303.2 (Cauchy completion route NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier* $R = (D, S, Q, E, B, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance, stream-window, regular rational, real seal, Bishop completion, transport, replay, provenance, and local naming rows;*
2. *dyadic observation discipline, requiring every completion-facing read to expose* D *and* S *before any regular rational or real seal row is consumed;*
3. *regularity routing, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *to be the only passage from finite windows to a completion-facing regular sequence;*
4. *Bishop route compatibility, requiring* B *to identify the same dyadic, stream, regular rational, and real rows as the Bishop regular-Cauchy completion packet;*
5. *terminal seal discipline, requiring the* $\mathsf{Real}^{\uparrow}$ *row* E *to be read only after* D, S, Q, B *have been displayed;*
6. *transport and replay non-escape, requiring* T, C, P, N *to preserve the listed rows and to export no quotient stream equality, selected limit, host real equality, countable choice, or ambient completeness theorem.*

No NameCert obligation is stored outside $D, S, Q, E, B, T, C, P, N$*.*

Proof. Project the carrier of Definition 4283.1. The finite packet exposes only the dyadic tolerance row, stream-window row, regular rational row, terminal real seal, Bishop completion route, transport, replay, provenance, and local naming.

The dyadic and stream rows are upstream. Any completion-facing consumer first reads the displayed tolerance and finite observation data, so the regular rational row is not a detached sequence or a host limit object.

The Bishop route row then checks compatibility with the established regular-Cauchy completion packet. It ties the same dyadic, stream, regular rational, and real rows to the route and prevents a second completion carrier from being inserted.

Finally T, C, P, N transport, replay, source, and name the displayed route. Since the carrier lacks quotient, selected-limit, host-equality, choice, and ambient-completeness coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 3303.3 (Cauchy completion route real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes only dyadic tolerance rows, finite stream windows, regular rational readback, the Bishop regular-Cauchy completion route, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It does not export quotient stream equality, selected limits, host real equality, countable choice, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with Definition 4283.2. It places every completion-facing read behind the displayed dyadic tolerance row D and stream-window row S.

The regularity route is then unique inside the carrier. The row Q is the only regular rational passage, and B confirms that the same D, S, Q, E rows are the rows consumed by the Bishop regular-Cauchy completion packet.

The terminal handoff is the row E. Transport by T, replay by C, provenance by P, and local naming by N preserve the route without adding new analytic data, so the $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRoute}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite observations, RegSeqRat regularity, a Bishop regular-Cauchy completion row, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4283.1, Definition 4283.2, and Definition 4283.3 seed the finite L10 completion route from dyadic observations to the Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic observation discipline, regularity routing, Bishop route compatibility, terminal seal discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionRouteUp.

Chapter 3304

A Concrete Naming Certificate for $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$

The $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ packet realizes the proof-search boundary in section 9386.4 as finite NameCert data. It sits beside the critical-path packet of chapter 765, the subject-reduction discharge socket of chapter 719, the normalization frontier of chapter 892, and the bounded normal-conversion bridge of chapter 3439. The packet names the exact frontier at which candidate evidence, bounded normal endpoints, discharge sockets, obstruction rows, transport, replay, provenance, and local naming may be consumed by a proof-search reader without claiming unrestricted closed consistency.

Falsifiable prediction: Any accepted proof-search consumer reaching a closed-consistency read without first exposing the discharge socket row refutes the $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ carrier.

Independence witness: metaciccriticalpath, subjectreductiondischargesocket, metacicnormalizationfrontier, boundednormalconversionbridge: the critical path records dependency pressure without local proof-search candidates; the discharge socket records subject-reduction inputs without bounded normal endpoints; the normalization frontier records candidate evidence without proof-search replay; and the bounded normal-conversion bridge records endpoint comparison without the discharge socket. None is bijective to $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$.

Definition 3304.1 (MetaCIC proof-search frontier carrier)**.** *A* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, D, N, B, A, E, O, T, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *row, retaining the closed-consistency dependency pressure over subject reduction, normalization, decidability, confluence, and examples.*
2. *D is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row, recording the socket through which proof-search evidence must pass before a closed-consistency consumer may read the subject-reduction boundary.*
3. *N is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row, exposing candidate evidence, closed-candidate evidence, finished-normal endpoints, and retained normalization obstruction data.*
4. *B is the* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *row, recording the bounded normal endpoint bridge used by conversion-facing proof-search checks.*

5. *A is the candidate-evidence row, storing the finite proof-search candidates, their local support rows, and the bounded normal endpoint they claim to reach.*
6. *E is the discharge-socket evidence row, pairing each accepted candidate with the socket read needed before closed-consistency handoff.*
7. *O is the obstruction row, retaining the excluded endpoints: unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, and proof-search completeness.*
8. *T records componentwise* hsame *transport for K, D, N, B, A, E, O and the local naming row.*
9. *C records displayed* Cont *replay from proof-search candidate admission through the critical path, discharge socket, normalization frontier, bounded bridge, evidence, obstruction, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, SubjectReductionDischargeSoc $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, D, N, B, A, E, O, T, C, P, L$. It has no coordinate for unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, proof-search completeness, or an unmediated closed-consistency read.

Lean4 ✓ `BEDC.Derived.MetaCICProofSearchFrontierUp.MetaCICProofSearchFrontierTasteGate single carrier alignment`

Theorem 3304.2 (MetaCIC proof-search frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *carrier $F = (K, D, N, B, A, E, O, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontie}}$ *surface consists exactly of the following finite obligation rows:*

1. *critical-path exposure through K;*
2. *subject-reduction discharge-socket exposure through D;*
3. *normalization-frontier exposure through N;*
4. *bounded normal-conversion bridge exposure through B;*
5. *candidate-evidence admission through A;*
6. *discharge-socket evidence pairing through E;*
7. *obstruction retention through O;*
8. *componentwise transport through T;*
9. *displayed replay through C;*
10. *provenance through P and local naming through L.*

No accepted proof-search read may skip the discharge socket row, and no local obligation is stored outside $K, D, N, B, A, E, O, T, C, P, L$.

Proof. Project the carrier of Definition 4284.1. Rows K, D, N, B are the already named frontier surfaces, while A and E store the local candidate and discharge-socket evidence.

The closed-consistency-facing read must pass through D and E. Candidate evidence in A may reach a bounded normal endpoint through N and B, but it is not eligible for the consistency handoff until the discharge socket row has been exposed.

The obstruction row O remains part of the packet throughout the read. It prevents the bounded proof-search frontier from being cited as unrestricted subject reduction, unrestricted normalization, global conversion decidability, kernel consistency, or proof-search completeness.

Finally T, C, P, L transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded endpoints or for an unmediated closed-consistency read, the listed rows exhaust the NameCert surface. □

Theorem 3304.3 (MetaCIC proof-search frontier discharge boundary)**.** *Every closed-consistency-facing consumer of an accepted* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{SubjectReductionDischargeSocket}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto \mathsf{BoundedNorm}$$

and then through the local candidate-evidence, discharge-socket evidence, obstruction, transport, replay, provenance, and NameCert rows. The route exports a bounded proof-search frontier only; it does not prove unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, or proof-search completeness. Future Lean work in TasteGate.lean *must provide* Nontrivial *and* FieldFaithful *instances before the carrier can be used as a checked proof-search taste gate.*

Proof. Start from Definition 4284.2. It forces the proof-search consumer to expose the critical path and the subject-reduction discharge socket before candidate evidence is treated as consistency-facing data.

The normalization and bounded conversion rows then bound the endpoint. The rows N and B justify only the displayed finished-normal and bounded-normal comparison reads, so the proof-search route remains a finite candidate route rather than a global theorem.

The local rows A, E, O, T, C, P, L preserve the discharge boundary. Candidate evidence is paired with the socket read, the obstruction row remains visible, and transport plus replay keep the consumer on the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ packet is generated from MetaCICCriticalPath, SubjectReductionDischargeSocket, MetaCICNormalizationFrontier, BoundedNormalConversionBridge, candidate evidence, discharge-socket evidence, obstruction, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4284.1, Definition 4284.2, and Definition 4284.3 seed the finite proof-search frontier for MetaCIC closed-consistency consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, proof-search completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked candidate admission, bounded-normal endpoint routing, discharge-socket exposure, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICProofSearchFrontierUp.

Chapter 3305

A Concrete Naming Certificate for $\mathsf{DyadicIntervalHull}^{\uparrow}$

Following Moore interval analysis and Bishop–Bridges constructive interval arithmetic, the $\mathsf{DyadicIntervalHull}^{\uparrow}$ packet records the finite lower-upper dyadic enclosure route used before Real and Completion consumers inspect an interval hull. It sits next to the endpoint carrier of chapter 306, the interval-operation packet of chapter 1645, the located hull surface of chapter 2327, and the located endpoint route of chapter 394. The packet names a constructive enclosure certificate for displayed dyadic endpoints; it is not a host interval primitive, not a selected real endpoint pair, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3305.1 (Dyadic interval hull carrier)**.** *A* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (L, U, O, A, B, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic endpoint row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *U is the upper dyadic endpoint row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *O is the endpoint-order witness, recording the displayed comparison $L \leq U$ through the same dyadic scale ledger.*
4. *A is the hull enclosure ledger, listing every inspected source endpoint or cell and its containment proof inside the displayed dyadic interval.*
5. *B is the* $\mathsf{DyadicInterval}^{\uparrow}$ *handoff row, packaging L, U, O, A as the dyadic interval consumed by interval-arithmetic neighbours.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which located endpoint observers may read finite endpoint approximations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same endpoint windows.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the dyadic interval row, located-real hull row, stream windows, and regular rational readbacks have been displayed.*
9. *T records componentwise* hsame *transport for endpoints, order witnesses, enclosure ledgers, located-real handoff rows, stream windows, readbacks, real seals, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from endpoint requests through L, U, O, A, B, S, R and then to E.*

11. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalHull}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealIntervalHull}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}$ *row.*

The classifier compares accepted hulls only through these rows. It has no coordinate for host intervals, selected real endpoints, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicIntervalHullUp.DyadicIntervalHullTasteGate single carrier alignment`

Theorem 3305.2 (Dyadic interval hull NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *carrier $H = (L, U, O, A, B, S, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint ordering, hull containment, located-real route, non-escape, transport, replay, provenance, and local naming obligations. Carrier admission displays the lower endpoint, upper endpoint, order witness, enclosure ledger,* $\mathsf{DyadicInterval}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local NameCert row. No obligation is stored outside those displayed rows.*

Proof. Project the accepted carrier. The lower and upper endpoint rows L, U are both dyadic rational core rows, and O is the only order witness available to the packet.

The enclosure ledger A is then read before any interval consumer sees the hull. It records finite containment of the displayed source endpoints or cells in the same lower-upper dyadic boundary.

The located-real route is the ordered handoff through B, S, R, E. The dyadic interval row packages the endpoint ledger, the stream row exposes finite endpoint windows, the regular rational row records readback, and the real row seals the terminal boundary.

Finally T, C, P, N preserve and name the route. Since the carrier has no coordinate for a host interval, selected endpoint pair, quotient stream, trichotomy, choice principle, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 3305.3 (Dyadic interval hull located-real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or Completion-facing consumer of an accepted* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *packet factors through the route*

$$\mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedRealIntervalHull}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

as displayed by the rows $L, U, O, A, B, S, R, E, T, C, P, N$. The consumer may inspect a hull only after the dyadic endpoints, order witness, containment ledger, located interval handoff, stream windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start from Definition 4285.2. It places every accepted read behind L, U, O, A, so the consumer first sees a finite dyadic lower-upper enclosure.

The row B is the dyadic interval handoff. It is the only route from endpoint rows to interval consumers, and it keeps the hull attached to the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ comparisons.

The located-real portion then runs through the finite windows S and readbacks R before the seal E is exposed. This is exactly the located interval route consumed by the surrounding located-real hull and located-interval chapters.

Transport, replay, provenance, and local naming preserve the row order. Therefore no Real or Completion consumer can bypass the dyadic interval, dyadic core, located hull, located interval, stream, regular rational, and real-seal rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}\uparrow}$.

Constructive story: A $\mathsf{DyadicIntervalHull}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, an endpoint-order witness, a hull enclosure ledger, DyadicInterval and located-real handoff rows, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4285.1, Definition 4285.2, and Definition 4285.3 seed the finite dyadic hull route from endpoint rows to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval primitives, selected real endpoint pairs, quotient stream equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint ordering, hull containment, located-real routing, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalHullUp.

Chapter 3306

A Concrete Naming Certificate for $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$

The $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet realizes the carrier-switch audit surface of chapter 9391 as finite BEDC data. It sits beside the carrier encoding of chapter 943, the BHist cell-row embedding of chapter 948, the substrate classifier of chapter 945, the cellular window trace of chapter 3481, and the carrier comparison of chapter 3998. The packet names the finite cost of retargeting theorem families, kernel verbs, proof scripts, and audit gates to a cellular carrier; it is not a replacement kernel carrier.

Definition 3306.1 (Cellular carrier switch-cost carrier)**.** *A* $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, O, V, R, A, K, H, C, P, N)$$

with the following displayed rows:

1. *B is the BHist theorem-family row, listing the finite family of BHist-facing statements under inspection.*
2. *O is the cellular orbit row, exposing the finite cellular rows and orbit windows to which the statements would be retargeted.*
3. *V is the verb-realignment row, recording how Mark, Hist, Ext, and Cont reads are translated to cellular classifier, orbit, extension, and continuation reads.*
4. *R is the proof-retarget budget row, recording the displayed proof obligations that must be rebuilt after the carrier change.*
5. *A is the audit-regeneration ledger, listing the checks that must be regenerated for the cellular statements and their NameCert markers.*
6. *K is the non-escape row, recording that the packet measures the switch cost without replacing the public BEDC* BHist *carrier.*
7. *H records componentwise* hsame *transport for theorem-family, orbit, verb, proof-budget, audit, non-escape, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through B, O, V, R, A, K before any switch-cost read reaches the local certificate.*

9. *P is the* Pkg *provenance over* CellularCarrierSwitchCost$^{\uparrow}$, CellularCarrierEncoding$^{\uparrow}$, BHistCellRowEmbedding$^{\uparrow}$, CellularSubstrateClassifier$^{\uparrow}$, CellularWindowTrace$^{\uparrow}$, CellularVerbRealignment$^{\uparrow}$, CellularCarrierBHistEquivalence$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CellularCarrierSwitchCost}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, V, R, A, K, H, C, P, N$. It has no coordinate for a kernel-carrier replacement, infinite cellular semantics, host evaluator equivalence, hidden proof synthesis, or an audit gate outside the displayed ledger.

Lean4 ✓ `BEDC.Derived.CellularCarrierSwitchCostUp.CellularCarrierSwitchCostTasteGate single carrier alignment`

Theorem 3306.2 (Cellular carrier switch-cost NameCert obligations)**.** *For every accepted* CellularCarrierSwitchCost *carrier $S = (B, O, V, R, A, K, H, C, P, N)$, the local* NameCert$_{\text{CellularCarrierSwitchCost}^{\uparrow}}$ *surface consists exactly of carrier admission, BHist theorem-family exposure, cellular orbit exposure, verb realignment, proof-retarget budgeting, audit-regeneration accounting, non-escape, transport, replay, provenance, and local naming obligations. A valid switch-cost read must display both the proof-retarget budget row R and the audit-regeneration ledger A before it reaches the local certificate.*

Proof. Carrier admission is projection from Definition 4286.1. The rows B and O expose the two carriers being compared as finite theorem-family and orbit-window data.

The verb row V records the syntactic realignment of Mark, Hist, Ext, and Cont. It does not itself measure the engineering cost, because statement retargeting and proof rebuilding are separate rows.

The proof-retarget budget R and audit-regeneration ledger A are therefore mandatory. A packet omitting either row cannot certify the displayed switch cost, since it would hide either the proof obligations or the audit gates that must be regenerated.

Finally K, H, C, P, N preserve the non-replacement boundary, replay the finite route, record provenance, and name the packet. No hidden carrier replacement, infinite orbit semantics, host evaluator equivalence, or external audit gate is available outside those rows. □

Theorem 3306.3 (Cellular carrier switch-cost finite audit route)**.** *Every accepted* CellularCarrierSwitchCost$^{\uparrow}$ *read factors through the finite route*

$$B, O, V, R, A, K, H, C, P, N.$$

The route measures a BHist theorem-family transposition by first displaying the theorem family, then the cellular orbit window, then the verb-realignment row, then the proof-retarget budget, and then the audit-regeneration ledger. It cannot export carrier replacement, host bisimulation, unbounded cellular traces, hidden proof repair, or completed audit success outside the displayed packet.

Proof. Read B first to identify the theorem-family surface whose statements are being retargeted. Read O next to expose the finite cellular orbit window available to the target carrier.

The row V aligns the verbs but does not discharge the work. It only states which BHist reads correspond to which cellular classifier and orbit reads.

The rows R and A then carry the cost. Proof retargeting is visible in R, while audit regeneration is visible in A, so the packet measures both proof work and gate work before local naming.

The non-escape and structural rows K, H, C, P, N keep this as a finite audit NameCert packet. Because no other coordinate exists, the route cannot export a carrier replacement, host bisimulation, unbounded trace, hidden proof repair, or completed audit success. □

Falsifiable prediction: A valid $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet is refuted if it omits the proof-retarget budget row or the audit-regeneration ledger.

Independence witness: cellularcarrierencoding, bhist-cell-row-embedding, cellularsubstrateclassifier, cellular-window-trace: $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries BHist theorem-family, cellular orbit, verb-realignment, proof-retarget, audit-regeneration, non-escape, transport, replay, provenance, and local naming rows.

Lean TasteGate coverage for this packet must provide Nontrivial and FieldFaithful instances in BEDC.Derived.CellularCarrierSwitchCostUp.TasteGate.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet is generated from BHist theorem-family rows, cellular orbit rows, verb-realignment rows, proof-retarget budget rows, audit-regeneration ledgers, non-escape rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4286.1, Definition 4286.2, and Definition 4286.3 seed the finite audit object for cellular carrier-switch cost.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close kernel-carrier replacement, infinite cellular semantics, host evaluator equivalence, hidden proof synthesis, completed audit success, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-family exposure, cellular orbit exposure, verb realignment, proof-retarget budgeting, audit-regeneration accounting, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CellularCarrierSwitchCostUp.

The sequence-level shared-modulus route is a downstream consumer of chapter 2299 and chapter 4356 before finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows are exposed.

Chapter 3307

A Concrete Naming Certificate for $\mathsf{UniformlyContinuousSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet records a sequence of maps whose continuity data share one displayed modulus ledger. It sits over the metric source row of chapter 57, the continuous-map surface of chapter 130, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The point of the packet is the shared-modulus route: a downstream consumer may inspect finitely many sequence entries only after the same uniform-control row has been displayed for all inspected maps.

The sequence carrier consumes the continuous-map rows of chapter 130 and the uniform-modulus handoff rows of chapter 369 before it exposes its own sequence-level carrier, so each inspected entry inherits a map row and the shared modulus remains a visible sibling dependency rather than a detached continuity family.

The $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ route first consumes the single-modulus surface of chapter 391 and the exported precision ledger of chapter 369; only after those two sibling rows are displayed can the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows be read as sequence-level data.

The compact-source read uses the route through chapter 2299, and the equicontinuous sequence comparison uses chapter 4356 before the shared-modulus row can be consumed.

The shared-modulus route is also a downstream consumer of chapter 2299 and chapter 4356: compact finite-net moduli and equicontinuity family rows supply the same tolerance-only handoff that $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ exposes to $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows.

Definition 3307.1 (Uniformly continuous sequence carrier)**.** *A* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, F, M, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row used by every displayed map in the sequence.*
3. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence-entry rows available to the packet.*
4. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, read once and then reused by every displayed entry in F.*

5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected indices, source probes, and requested tolerances.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and to the shared modulus ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for the target readings reached only after S, Q, M have been displayed.*
8. *H records componentwise* hsame *transport for source metric, target metric, sequence-entry, shared-modulus, stream-window, regular-readback, real-seal, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay from an inspected sequence index and tolerance request through X, Y, F, M, S, Q, E.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, F, M, S, Q, E, H, C, P, N$. It has no coordinate for host function extensionality, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached pointwise continuity proof for a sequence entry.*

Theorem 3307.2 (Uniformly continuous sequence NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier $U = (X, Y, F, M, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequ}}$ *surface consists exactly of carrier admission for the two metric rows, continuous-map sequence-entry rows, shared uniform-modulus row, finite stream windows, regular rational readback, real-name seal, componentwise transport, continuation replay, package provenance, and local naming; shared-modulus exactness requiring each inspected entry of F to consume the same M row; finite-window exactness through S; readback exactness through Q; real-seal boundary through E; and non-escape from host function extensionality, quotient streams, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and detached pointwise continuity data.*

Proof. Project the carrier of Definition 4287.1. The only source, target, sequence-entry, shared-modulus, stream, readback, and real-seal data available to the classifier are the displayed rows X, Y, F, M, S, Q, E.

The shared-modulus clause is read before any inspected sequence entry reaches a downstream Real-facing consumer. Each entry in F is a $\mathsf{ContinuousMap}^{\uparrow}$ row, but the packet exposes it to a tolerance request only through the common $\mathsf{UniformModulus}^{\uparrow}$ row M and the finite windows in S.

Readback and sealing are downstream of that shared route. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q records only the rational data obtained from the displayed windows and shared modulus, and the $\mathsf{Real}^{\uparrow}$ row E is reached only after S, Q, M have been exposed.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for host function extensionality, quotient stream equality, selected limits, countable choice, ambient real completeness, or detached pointwise continuity proofs, the listed rows exhaust the local NameCert surface. □

Theorem 3307.3 (Uniformly continuous sequence shared-modulus handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{UniformModulus}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

The route supplies a shared modulus for all inspected sequence entries, finite stream windows for those inspections, regular rational readback, and the terminal real-name seal. It exports no host

function extensionality, quotient stream equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric space.*

Proof. Start from the obligation surface of Definition 4287.2. It identifies the continuous-map entries and the shared uniform-modulus ledger as carrier rows, not as host functions or external continuity facts.

A Real-facing consumer next opens only the finite windows in S. Those windows determine the regular rational readback Q under the shared modulus M, and only then can the terminal seal E be consumed.

The transport, continuation, provenance, and local naming rows preserve that same order. Therefore every public read is a shared-modulus sequence-of-maps route through the displayed BEDC rows, with the excluded extensionality, quotient, choice, completeness, and hidden-completion objects absent from the packet. □

Theorem 3307.4 (Uniformly continuous sequence obligation-closure package)**.** *The carrier of Definition 4287.1, the local NameCert obligations of Definition 4287.2, the shared-modulus handoff of Definition 4287.3, and the compact/uniform-modulus child routes assemble one obligation surface for* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$*. A consumer may read source data through* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *rows, sequence entries through* $\mathsf{ContinuousMap}^{\uparrow}$*, uniform control through* $\mathsf{UniformModulus}^{\uparrow}$*, inspected indices through* $\mathsf{StreamName}^{\uparrow}$ *windows, rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, terminal sealing through* $\mathsf{Real}^{\uparrow}$*, and structural movement through* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This package exports no host function extensionality, quotient stream equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete metric space, or detached pointwise-continuity proof.*

Proof. Start from the carrier row $U = (X, Y, F, M, S, Q, E, H, C, P, N)$. The source and target metric rows, the finite sequence-entry row, the shared modulus row, the stream-window row, the regular rational readback row, and the real-seal row are the only analytic data visible to the packet.

Apply the NameCert obligations. They assign carrier admission, shared-modulus exactness, finite-window exactness, readback exactness, real-seal boundary, transport, replay, provenance, and local naming to the displayed rows.

Use the shared-modulus handoff together with the compact-source and uniform-modulus routes named in the chapter introduction. Those routes force every inspected $\mathsf{ContinuousMap}^{\uparrow}$ entry to consume the same $\mathsf{UniformModulus}^{\uparrow}$ row before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows can be read.

Therefore the obligation package is exactly the finite BEDC surface listed in the statement. Since no carrier coordinate names host extensionality, quotient streams, selected limits, countable choice, ambient completeness, hidden complete metrics, or detached pointwise proofs, none of those endpoints is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet is generated from source and target MetricSpace rows, ContinuousMap sequence-entry rows, one shared UniformModulus row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$, seed)

Scope closed: Definition 4287.1, Definition 4287.2, and Definition 4287.3 seed the shared-modulus sequence-of-maps packet for Real-facing finite-window consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient streams, selected limits, countable choice, ambient RealUp completeness, hidden complete metric spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, finite-window exactness, regular-readback boundary, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.UniformlyContinuousSe

The UniformlyContinuousSequence$^{\uparrow}$ route consumes chapter 2299, chapter 369, and chapter 130 as load-bearing rows before any sequence endpoint is public. The compact-uniform-modulus sibling supplies the finite compact-window fold, the uniform-modulus sibling supplies the shared tolerance-to-precision row, and the continuous-map sibling supplies the pointwise graph and modulus readback that the sequence packet transports along hsame and replays through Cont.

Thus a sequence-level uniform-continuity packet reads compact cover data, finite modulus rows, tolerance readbacks, and terminal Real-facing seals through the sibling modulus lattice rather than through a host continuity theorem or an external compactness oracle.

Theorem 3307.5 (Compact uniform-modulus route for uniformly continuous sequences)**.** *Every Real-facing read of an accepted* UniformlyContinuousSequence$^{\uparrow}$ *carrier whose source is displayed through the compact route must consume the shared* UniformModulus$^{\uparrow}$ *row through the compact-uniform modulus packet of chapter 2299. The read factors through the visible route*

$$\mathsf{CompactUniformModulus}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

It exports no detached pointwise continuity family, host compactness theorem, selected limit, quotient stream equality, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4287.1. The sequence entries are available only as ContinuousMap$^{\uparrow}$ rows, the inspected indices only through StreamName$^{\uparrow}$ windows, and the target values only after RegSeqRat$^{\uparrow}$ readback and the terminal Real$^{\uparrow}$ seal.

When the source row is compact, the shared modulus row is not read as an external theorem. It is consumed through the compact-uniform modulus route of chapter 2299, which packages compact source coverage and the downstream UniformModulus$^{\uparrow}$ handoff before any Real-facing sequence read is public.

The transport, replay, provenance, and local naming rows then preserve the same route. Since the carrier has no coordinate for detached pointwise continuity, host compactness, selected limits, quotient stream equality, countable choice, or ambient real completeness, a Real-facing compact-source read must pass through the displayed compact-uniform modulus dependency. □

Theorem 3307.6 (Uniformly continuous sequence compact-metric forward dependency)**.** *Every* CompactMetric$^{\uparrow}$ *consumer of an accepted* UniformlyContinuousSequence$^{\uparrow}$ *packet must first read the visible source and observation route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Only after the metric source row, continuous-map entry rows, shared uniform-modulus row, finite stream windows, regular rational readback, and terminal real seal are displayed may the consumer inspect compact-metric modulus data.

Proof. Project the carrier of Definition 4287.1. The compact-facing consumer receives the source metric row X, target metric row Y, continuous-map entry rows F, shared modulus row M, stream-window row S, readback row Q, and real-seal row E before any structural transport or provenance row can be used.

The compact-metric dependency route then reads the compact source through chapter 129. That route is downstream of the displayed Met$^{\uparrow}$ source and the same UniformModulus$^{\uparrow}$ row used

by the sequence packet, so compact-metric modulus data cannot be consumed before the shared sequence modulus and finite windows are visible.

Finally the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows seal the consumer-facing values. Since the carrier has no coordinate for detached pointwise continuity, selected limits, quotient stream equality, countable choice, or ambient real completeness, the compact-metric consumer is forced through the displayed forward dependency. □

The compact-metric consumer reads $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ through chapter 129 and chapter 369 before using the shared sequence modulus. This keeps the sequence ledger attached to compact finite-net data and the finite modulus row, rather than to an external pointwise continuity family, open-cover compactness theorem, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Theorem 3307.7 (UniformlyContinuousSequence equicontinuous dependency)**.** *Every accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *shared-modulus packet supplies the common modulus row consumed by* $\mathsf{EquicontinuousSequence}^{\uparrow}$*. The dependency is read through* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{CompactMetric}^{\uparrow}$ *source rows of chapter 129, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, and the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before the equicontinuity consumer of chapter 4356 may inspect the shared sequence modulus.*

Proof. Start with the carrier of Definition 4287.1. Its finite sequence route displays the source and target metric rows, continuous-map entries, one shared $\mathsf{UniformModulus}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, regular rational readback, terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming.

The compact-metric dependency route keeps the same shared modulus attached to compact source rows before downstream compact-uniform consumers read it. Thus an $\mathsf{EquicontinuousSequence}^{\uparrow}$ consumer receives a common modulus only after the compact source, stream window, rational readback, and real seal rows have been displayed.

Finally use the equicontinuous-sequence carrier of Definition 4356.1. Its common schedule and equicontinuity rows consume exactly the kind of shared finite modulus exported by the $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet, while selected subsequences, quotient stream equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside both carriers. □

Theorem 3307.8 (Uniformly continuous sequence equicontinuity handoff)**.** *Every accepted* UniformlyContinuousSe *packet read over a displayed finite sequence window may be consumed as a finite family-modulus route for chapter 4076. The read first exposes the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the compact-metric source boundary, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row; only then may an* $\mathsf{Equicontinuity}^{\uparrow}$ *consumer inspect the same window as a finite family indexed by the displayed sequence positions.*

Proof. Project the uniformly continuous sequence carrier through Definition 4287.1. Its shared modulus is already attached to the displayed sequence window, so the finite family read does not choose a new radius schedule.

Read the equicontinuity carrier through Definition 4076.1. That carrier asks for a finite family ledger, a common tolerance-radius handoff, transport, replay, provenance, and local naming. The sequence window supplies exactly that route when its displayed indices are treated as the family index set.

Thus the dependency is a forward consumer edge from the sequence packet to $\mathsf{Equicontinuity}^{\uparrow}$. It does not add selected subsequences, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3307.9 (Uniformly continuous sequence located-modulus dependency route)**.** *Every* UniformlyContinuousSequence$^{\uparrow}$ *read that reaches a sequence-level* Real$^{\uparrow}$ *consumer must first expose a substantive modulus sibling: either the* UniformModulus$^{\uparrow}$ *route of chapter 369 or the compact handoff through* CompactUniformModulus$^{\uparrow}$ *in chapter 2299. The sequence packet therefore exports a finite modulus-mediated read surface before any Real-facing sequence value is consumed, and it provides no detached pointwise-continuity family, selected limit, quotient stream equality, countable-choice schedule, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted uniformly continuous sequence packet. Its sequence rows can be inspected only through displayed finite windows, and a Real-facing consumer can read values only after the regular rational and terminal Real rows have been exposed.

The modulus row supplies the horizontal dependency. A general sequence read consumes UniformModulus$^{\uparrow}$ through chapter 369; a compact-source read consumes the compact route through chapter 2299 before it presents the same uniform-modulus handoff.

Thus the sequence carrier is not a kernel-only island. Its downstream Real route is mediated by a displayed modulus sibling, so the packet cannot supply a detached pointwise family, selected limit, quotient stream equality, countable-choice schedule, or ambient real completeness. □

Theorem 3307.10 (Uniformly continuous sequence pointwise stream metric route)**.** *Every accepted* UniformlyContinuousSequence$^{\uparrow}$ *packet whose finite windows are consumed pointwise by a metric-stream reader must expose the same source and target* Met$^{\uparrow}$ *rows, shared* UniformModulus$^{\uparrow}$ *row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can invoke the* PointwiseStreamMetric$^{\uparrow}$ *comparison route of chapter 4139. The read is a shared-modulus stream route, not detached pointwise continuity, quotient stream equality, selected limits, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4287.1. The pointwise metric-stream reader receives only $X, Y, F, M, S, Q, E, H, C, P, N$, so the source and target metric rows and the shared uniform-modulus row remain visible before any inspected sequence entry is read.

Use the NameCert obligations and shared-modulus handoff. They force each finite window to pass through the same UniformModulus$^{\uparrow}$ row M, then through StreamName$^{\uparrow}$ windows S and RegSeqRat$^{\uparrow}$ readback Q before the Real$^{\uparrow}$ seal E can be consumed.

Finally route the visible windows through chapter 4139. The PointwiseStreamMetric$^{\uparrow}$ comparison can inspect the displayed metric-stream rows, but the uniformly continuous sequence packet has no coordinate for detached pointwise continuity proofs, quotient stream equality, selected limits, countable-choice schedules, or ambient Real completeness. □

The UniformlyContinuousSequence$^{\uparrow}$ completion-facing route reads the regular-Cauchy metric boundary through chapter 8164 and the compact uniform-modulus route of chapter 2299 before any sequence-level Real seal is public. This keeps compact finite-net data, pointwise-stream observations, shared UniformModulus$^{\uparrow}$ rows, RegSeqRat$^{\uparrow}$ readback, and terminal Real$^{\uparrow}$ seals on one displayed sibling route rather than on a detached host continuity theorem, selected limit, quotient stream equality, or ambient completion principle.

Theorem 3307.11 (Uniformly continuous sequence sibling dependency)**.** *Every* UniformlyContinuousSequence$^{\uparrow}$ *consumer first reads the uniform-modulus surface of chapter 391 and the compact-uniform handoff of chapter 388 before it routes pointwise stream metrics into sequence-level uniformity. Thus the sibling surface is a finite lattice of uniform-control rows, compact handoff rows, stream windows,*

transport, replay, provenance, and local naming, not detached pointwise continuity, selected limits, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the route through the two sibling chapters named in the statement. The uniform-continuity row supplies the modulus that every sequence-level consumer must keep visible, and the compact-uniform row supplies the handoff that turns compact-domain control into a stable sequence surface.

The remaining rows are bookkeeping for that route. Stream windows present the finite observations, hsame transport preserves the selected control data, Cont replay replays the same rows, Pkg records provenance, and the local naming row records the resulting $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ certificate. No extra pointwise or quotient-level equality is introduced. □

Every public sequence-level uniform-continuity read from $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ must first pass through the sibling lattice formed by compact finite-net rows from chapter 129, the shared $\mathsf{UniformModulus}^{\uparrow}$ row of chapter 369, pointwise stream-metric rows from chapter 4139, $\mathsf{StreamName}^{\uparrow}$ window rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seals. The lattice gives the consumer finite compact coverage, one reusable modulus, stream-local metric observations, windowed rational reads, and sealed target values before any sequence-level uniform continuity claim is public; it exports no detached pointwise-continuity family, selected limit, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Every $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ consumer that reads sequence-level uniform continuity must pass through the finite modulus surface of chapter 2228 or chapter 5128 before it can inspect the shared sequence ledger. Completion-facing uses then continue through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback route of chapter 38, so the packet exposes a finite uniform-modulus dependency rather than a host pointwise-continuity family, a selected limit, or an ambient completeness principle.

Chapter 3308

A Concrete Naming Certificate for $\mathsf{LocatedIntervalHalving}^{\uparrow}$

Following Bishop and Bridges constructive interval-halving analysis, the $\mathsf{LocatedIntervalHalving}^{\uparrow}$ packet isolates the one-step finite halving certificate that later bisection and convergence chapters consume. It reads the midpoint route of chapter 1712, the retained-child route of chapter 1711, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the downstream convergence consumer of chapter 1804. The certificate names one visible halving step, not an infinite branch, a selected limit, host interval equality, or ambient completeness.

Definition 3308.1 (Located interval halving carrier)**.** *A* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, M, L, R, S, D, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *I is the source located real interval row. It supplies the lower endpoint, upper endpoint, radius budget, locator reads, locatedness modulus, and endpoint-order observations for the parent interval.*
2. *M is the* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ *row. It records the displayed midpoint route and the two child endpoint-transport rows before any retained side is selected.*
3. *L and R are the left-child and right-child interval rows opened from M.*
4. *S is the side row. It records the finite locator, sign, containment, or requested-side datum that selects which of L, R is retained for this halving step.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the common scale, half-radius budget, and dyadic comparison data used by the one-step diameter estimate.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained child endpoints and midpoint windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, M, L, R, S, D, Q have been displayed.*
8. *T records componentwise* hsame *transport for the source interval, midpoint row, child intervals, side row, dyadic tolerance row, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a halving request through source inspection, midpoint construction, child exposure, side retention, dyadic tolerance checking, regular readback, and the Real-facing seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalHalving}^{\uparrow}$, $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one located interval halving packet. It has no coordinate for host interval equality, real trichotomy, quotient representatives, countable choice, an infinite branch, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalHalvingUp.LocatedIntervalHalvingTasteGate single carrier alignment`

Theorem 3308.2 (LocatedIntervalHalving NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier*

$$H = (I, M, L, R, S, D, Q, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source located interval, midpoint row, two child intervals, side row, dyadic tolerance row, regular readback, Real seal, transport, replay, provenance, and local naming;*

2. *source discipline, requiring every endpoint, radius, locator, locatedness, and order read to factor through I before midpoint or side data are inspected;*

3. *midpoint exposure, requiring M to display the midpoint and both child endpoint-transport routes before the retained side is read;*

4. *side retention, requiring S to choose only between the displayed children L and R and to export no infinite branch selector;*

5. *dyadic halving exactness, requiring D to attach the retained child to the common scale and half-radius budget used by the one-step diameter estimate;*

6. *Real-facing handoff, requiring Q, E, T, C, P, N to route the retained child through regular rational readback, the terminal Real seal, transport, replay, provenance, and local naming;*

7. *ledger exhaustion, excluding host interval equality, real trichotomy, quotient representatives, countable choice, infinite branches, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge data.*

No located interval halving obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4288.1. The displayed tuple exposes the source interval, midpoint route, two child intervals, side row, dyadic tolerance row, regular readback, Real seal, and structural rows as finite BHist coordinates.

Source discipline is supplied by I. The halving request starts from the source located interval row, so endpoint, radius, locator, locatedness, and order observations cannot be replaced by a host closed interval object.

Midpoint exposure is supplied by M, L, R. The midpoint row displays the two child endpoint-transport routes, and the child interval rows make those two finite cells available before any retained side is read.

Side retention and dyadic exactness are supplied by S and D. The side row selects only between the displayed children, while the dyadic row records the common scale and half-radius budget for that retained child.

Finally Q, E, T, C, P, N carry the retained child to regular rational readback, the terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. Since every visible coordinate has already been listed, the excluded host equality, trichotomy, quotient, choice, branch, limit, completeness, verification, and bridge objects cannot be projected. □

Theorem 3308.3 (LocatedIntervalHalving RegSeqRat and Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier reads the retained child only after the finite route*

$$I \longmapsto M, L, R \longmapsto S, D \longmapsto Q \longmapsto E$$

has been displayed. Consequently the handoff consumed by chapter 1711 and chapter 1804 is a one-step located halving certificate over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, not a selected branch, a quotient representative, or an ambient completeness principle.*

Proof. Start with the accepted carrier $H = (I, M, L, R, S, D, Q, E, T, C, P, N)$. The only source coordinate is I, so a readback or Real-facing consumer first receives the displayed located interval data.

Next the consumer reads M, L, R. These rows display the midpoint and both child intervals before the retained child is selected, so the later route cannot hide a midpoint construction or child endpoint transport.

The retained side is then fixed by S and measured by D. The side row chooses between the displayed children, and the dyadic tolerance row records the common-scale half-radius estimate for that child.

Only after those rows are displayed does Q expose the regular rational readback and E expose the terminal Real seal. The structural rows T, C, P, N replay, transport, record provenance, and name the same finite route. Thus downstream bisection and convergence consumers receive a visible one-step halving certificate and no selected infinite branch, quotient representative, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4288.1, Definition 4288.2, and Definition 4288.3 seed the one-step located interval halving route through source interval, midpoint, child interval, side-retention, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, real trichotomy, quotient representatives, countable choice, infinite branches, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, midpoint exposure, side retention, dyadic halving exactness, RegSeqRat and Real handoff, and non-escape proofs for BEDC.Derived.LocatedIntervalHalvingUp.

Constructive story: A $\mathsf{LocatedIntervalHalving}^{\uparrow}$ packet is generated from one source located interval row, one LocatedIntervalMidpointUp row, two child interval rows, one finite side-retention row, one DyadicRatCoreUp tolerance row, one RegSeqRatUp readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3309

A Concrete Naming Certificate for $\mathsf{BishopLocatedRoot}^{\uparrow}$

Following Bishop and Bridges constructive located-root arguments, the $\mathsf{BishopLocatedRoot}^{\uparrow}$ packet records the finite bisection route by which a located sign-changing bracket is read as a terminal $\mathsf{Real}^{\uparrow}$ root seal. It sits after the bisection convergence surface of chapter 1804, the located intermediate-value route of chapter 1402, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a constructive root certificate over displayed finite rows; it does not assert full $\mathsf{Real}^{\uparrow}$ completeness, classical trichotomy, quotient real equality, selected host roots, or countable choice.

Definition 3309.1 (Bishop located root carrier)**.** *A* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, O, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the located bracket row naming two displayed endpoint windows and the retained interval boundary.*
2. *O is the opposite-sign witness row, read as positive apartness data for the two endpoint evaluations.*
3. *S is the bisection schedule row selecting the retained subintervals and midpoint tests.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording widths, midpoint radii, sign-test budgets, and the requested terminal precision.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the finite bisection windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, O, S, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the bracket, sign witness, bisection schedule, dyadic ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a root request through $B, O, S, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRoot}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay path. It has no coordinate for a host root object, real trichotomy, quotient real equality, ambient completeness, countable choice, or a detached limit point.

Lean4 ✓ `BEDC.Derived.BishopLocatedRootUp.TasteGate.BishopLocatedRootTasteGate single carrier alignment`

Theorem 3309.2 (Bishop located root NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *carrier*

$$R = (B, O, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$ *surface consists exactly of carrier admission for the located bracket, opposite-sign witness, bisection schedule, dyadic tolerance ledger, regular rational readback, terminal Real seal, componentwise transport, displayed continuation replay, provenance, and local naming row; bisection-route exposure; Real-seal nonescape; and refusal of host roots, trichotomy, quotient real equality, ambient completeness, countable choice, and detached limit points.*

Proof. Project the carrier of Definition 4289.1. The accepted packet exposes exactly the bracket, sign witness, schedule, dyadic ledger, regular readback, Real seal, and structural rows H, C, P, N.

The bisection route is forced by the displayed order. A root-facing consumer must inspect the located bracket and opposite-sign row before it can read midpoint choices, dyadic widths, or the regular readback.

The terminal Real seal is downstream of those rows. Since the carrier has no coordinate for a host root, trichotomy, quotient equality, ambient completeness, countable choice, or a detached limit, the listed rows exhaust the local NameCert surface. □

Theorem 3309.3 (Bishop located root bisection route)**.** *Every accepted* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *root read factors through the finite route*

$$B \longmapsto O \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

before any downstream Real or Completion consumer can inspect the root seal. In particular, the read uses the bisection convergence route of chapter 1804, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

Proof. Start with an accepted root read. The located bracket row B and opposite-sign witness O are the only sources for the retained interval and sign-change data.

Apply the bisection schedule row S and dyadic ledger D. These rows record the finite midpoint choices, interval widths, and tolerance budgets consumed by the bisection convergence route.

Read Q and E only after the finite schedule and ledger have been displayed. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational source, and the $\mathsf{Real}^{\uparrow}$ seal records the terminal root-facing name. The structural rows H, C, P, N transport, replay, source, and name the same packet, so no host root, trichotomy, quotient equality, countable choice, or ambient completeness coordinate can enter between the bracket and the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRoot}^{\uparrow}$ packet is generated from a located bracket, opposite-sign witness, bisection schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4289.1, Definition 4289.2, and Definition 4289.3 seed the finite located-root certificate over bisection, StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedRoot}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, classical trichotomy, quotient real equality, selected host roots, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, opposite-sign stability, bisection-route exposure, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRootUp.

Chapter 3310

A Concrete Naming Certificate for $\mathsf{RealCompletenessWitness}^{\uparrow}$

The $\mathsf{RealCompletenessWitness}^{\uparrow}$ packet realizes the four-face Real and Completion route as finite BEDC data: the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier face, and the terminal $\mathsf{Real}^{\uparrow}$ seal face. It records the completion witness used by Real-facing consumers without turning that witness into ambient completeness, quotient stream equality, selected limits, trichotomy, or countable choice.

Definition 3310.1 (Real completeness witness carrier)**.** *A* $\mathsf{RealCompletenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the requested observations.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of endpoint rows, radii, comparison distances, and tolerance budgets.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row read from the same finite schedule and dyadic ledger.*
4. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* S, D, R *have been displayed.*
5. H *records componentwise* hsame *transport for the four faces, replay, provenance, and local naming rows.*
6. C *records displayed* Cont *replay from a completion-facing request through* S, D, R, E, H, P, N.
7. P *is the* Pkg *provenance over* $\mathsf{RealCompletenessWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted witnesses only through the displayed four-face route and the structural rows. It has no coordinate for quotient streams, selected host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, countable choice, field completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealCompletenessWitnessUp.RealCompletenessWitnessTasteGate single carrier alignment`

Theorem 3310.2 (Real completeness witness NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWitness}$ *carrier* $W = (S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity and classifier face, terminal RealUp seal face, componentwise transport, displayed continuation replay, provenance, and local naming; four-face route exposure; and refusal of quotient streams, selected limits, ambient completeness, trichotomy, countable choice, detached field structure, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 4290.1. The accepted packet has exactly four mathematical faces, followed by transport, replay, provenance, and naming rows.

The route exposure is ordered. A completion-facing consumer reads S before dyadic tolerances, D before regular readback, R before the terminal Real seal, and E before any downstream seal consumer.

The exclusions are an inventory check. No coordinate stores quotient stream equality, a selected host limit, an ambient completeness theorem, trichotomy, countable choice, a field-faithfulness proof, Lean verification, or a bridge certificate. Therefore the displayed rows exhaust the local NameCert surface. □

Theorem 3310.3 (Real completeness witness four-face route)**.** *Every accepted* $\mathsf{RealCompletenessWitness}^{\uparrow}$ *consumer reaches completion-facing Real data only through the finite route*

$$S \longmapsto D \longmapsto R \longmapsto E$$

and then through H, C, P, N*. Thus the witness is a finite four-face packet over chapter 126, chapter 291, chapter 301, and chapter 38, not an assertion of full* $\mathsf{Real}^{\uparrow}$ *completeness or a field-completeness theorem.*

Proof. Start with an accepted completion-facing read. The only source face is S, the finite $\mathsf{StreamName}^{\uparrow}$ schedule, so the read first fixes the visible windows.

Next read D and R. The dyadic face supplies endpoint, radius, distance, and tolerance rows; the regular-rational face reads those rows as the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier surface.

Finally read E as the terminal $\mathsf{Real}^{\uparrow}$ seal and pass through H, C, P, N for transport, replay, source, and local naming. Since no field-faithfulness, nontriviality, quotient stream, selected limit, or ambient completeness coordinate is present in the carrier, those later Lean obligations must be supplied separately by ‘TasteGate.lean‘, with ‘Nontrivial RealCompletenessWitnessUp‘ and ‘FieldFaithful RealCompletenessWitnessUp‘ recorded as independent evidence rather than projected from this packet. □

Falsifiable prediction: Any accepted $\mathsf{RealCompletenessWitness}^{\uparrow}$ read that reaches Real-completion data while bypassing the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ seal rows refutes the carrier.

Independence witness: streamname, dyadicratcore, regseqrat, real, finite-real-section: $\mathsf{RealCompletenessWitness}$ is not bijective to any listed sibling because it jointly carries the finite-window schedule, dyadic tolerance ledger, regularity and classifier readback, terminal Real seal, structural transport, replay, provenance, and local NameCert rows as one four-face packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RealCompletenessWitness}^{\uparrow}$ packet is generated from a StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity and classifier face, terminal RealUp seal face, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealCompletenessWitness}\uparrow}$, seed)

Scope closed: Definition 4290.1, Definition 4290.2, and Definition 4290.3 seed the finite four-face completion witness over StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCompletenessWitness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, quotient stream equality, selected limits, field-faithfulness, nontriviality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked four-face carrier admission, route exposure, nonescape, transport, replay, provenance, local naming, and separate TasteGate evidence for BEDC.Derived.RealCompletenessWitnessUp.

Chapter 3311

A Concrete Naming Certificate for $\mathsf{SmoothManifold}^{\uparrow}$

Following John M. Lee's smooth manifold presentation, the $\mathsf{SmoothManifold}^{\uparrow}$ packet records a finite atlas, overlap, transition, and replay surface for a manifold object already routed through chapter 96, chapter 88, chapter 47, chapter 130, and chapter 97. It names the differentiable atlas boundary used by later bundle, connection, and metric consumers; it is not a host smooth-space primitive, not a global coordinate choice, and not an ambient classical maximal-atlas theorem.

Definition 3311.1 (Smooth manifold carrier)**.** *A* $\mathsf{SmoothManifold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, T, V, A, O, S, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the underlying* $\mathsf{Manifold}^{\uparrow}$ *carrier row.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *chart-domain ledger, recording the displayed open-domain data used by the atlas.*
3. *V is the finite* $\mathsf{VecSpace}^{\uparrow}$ *model row for the chart codomains.*
4. *A is the atlas-admission row, listing the chart names, domains, codomain windows, and coverage witnesses accepted by the packet.*
5. *O is the chart-overlap ledger, recording the finite intersections on which transition comparisons are requested.*
6. *S is the smooth-transition classifier row, whose entries are read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and the displayed finite differentiability tests available on the model* $\mathsf{VecSpace}^{\uparrow}$ *windows.*
7. *R is the* $\mathsf{Bundle}^{\uparrow}$ *readiness row, exposing the atlas route that downstream tangent, frame, and coordinate-bundle packets may consume.*
8. *H records componentwise* hsame *transport for the manifold carrier, topology ledger, model vector space rows, atlas, overlaps, smooth-transition classifiers, bundle-readiness row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from chart requests through B, T, V, A, O, S, R.*
10. *P is the* Pkg *provenance over* $\mathsf{SmoothManifold}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SmoothManifold}^\uparrow}$ *row.*
The classifier compares accepted packets only through $B, T, V, A, O, S, R, H, C, P, N$. It has no coordinate for a selected maximal atlas, ambient set-theoretic smooth structure, global chart, classical paracompactness theorem, connection form, or Riemannian metric.

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldTasteGate single carrier alignment`

Theorem 3311.2 (Smooth manifold NameCert obligations)**.** *For every accepted* $\mathsf{SmoothManifold}^\uparrow$ *carrier $M = (B, T, V, A, O, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SmoothManifold}^\uparrow}$ *surface consists exactly of carrier admission, atlas admission, chart-overlap smoothness, transition compatibility,* hsame *transport,* Cont *replay, package provenance, and local naming obligations. In particular, the atlas row must display finite chart admission, the overlap row must display every inspected intersection, the transition row must display smooth compatibility on those overlaps, and the replay row must reproduce the same route before any* $\mathsf{Bundle}^\uparrow$ *consumer reads R.*

Proof. Project the accepted carrier. The base rows B, T, V expose the topological manifold boundary, chart-domain ledger, and model vector-space windows before the atlas can be read.

The atlas row A is then checked against those rows. Its chart names, displayed domains, codomain windows, and finite coverage witnesses are the only admission data available to the packet.

On each displayed overlap in O, the transition classifier S reads the corresponding coordinate changes through $\mathsf{ContinuousMap}^\uparrow$ and the finite differentiability tests attached to the model $\mathsf{VecSpace}^\uparrow$ windows. Transition compatibility is therefore stored in S rather than in a hidden maximal-atlas object.

Finally H, C, P, N transport, replay, source, and name the same rows, and R exposes only that displayed atlas route to $\mathsf{Bundle}^\uparrow$ consumers. Since the carrier has no coordinate for a selected maximal atlas, global chart, connection form, or metric tensor, the listed obligations exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldNamecertObligations`

Theorem 3311.3 (Smooth manifold bundle-readiness route)**.** *Every* $\mathsf{Bundle}^\uparrow$*-facing consumer of an accepted* $\mathsf{SmoothManifold}^\uparrow$ *packet factors through the displayed route*

$$\mathsf{Manifold}^\uparrow \longmapsto \mathsf{Topology}^\uparrow \longmapsto \mathsf{VecSpace}^\uparrow \longmapsto \mathsf{ContinuousMap}^\uparrow \longmapsto \mathsf{Bundle}^\uparrow$$

as represented by the rows $B, T, V, A, O, S, R, H, C, P, N$. The consumer may read coordinate-bundle readiness only after atlas admission, overlap smoothness, transition compatibility, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start with the NameCert obligation theorem. It places every accepted bundle-facing read behind the carrier, topology, model-space, atlas, overlap, and transition rows.

The row R is downstream of S, so a bundle consumer cannot inspect a smooth coordinate route before transition compatibility has been displayed on each overlap requested by O. The rows H and C preserve the same order under transport and replay.

The package and local naming rows record that the exposed route is the finite $\mathsf{SmoothManifold}^\uparrow$ atlas surface. Therefore later bundle, connection, and metric packets consume a named differentiable-atlas boundary rather than an ambient maximal atlas or a global smooth-space primitive. □

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldBundleReadinessRoute`

Falsifiable prediction: Every accepted $\mathsf{SmoothManifold}^{\uparrow}$ consumer that reaches $\mathsf{Bundle}^{\uparrow}$ readiness without reading atlas admission, chart-overlap smoothness, transition compatibility, hsame transport, Cont replay, Pkg provenance, and local NameCert rows refutes this packet surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}$.

Constructive story: A $\mathsf{SmoothManifold}^{\uparrow}$ packet is generated from a Manifold carrier row, Topology chart-domain ledger, VecSpace model windows, atlas admission, chart-overlap ledger, smooth-transition classifier, Bundle readiness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4291.1, Definition 4291.2, and Definition 4291.3 seed the finite smooth-atlas route from manifold and topology rows to Bundle consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected maximal atlases, ambient smooth-space primitives, global charts, paracompactness, connections, or Riemannian metrics.

Upgrade path: Obligation closure requires checked carrier admission, atlas admission, overlap smoothness, transition compatibility, transport, replay, provenance, and local naming for BEDC.Derived.SmoothManifoldUp.

Chapter 3312

A Concrete Naming Certificate for $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$

The $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ packet realizes the normal-endpoint budget from the MetaCIC normalization frontier as finite NameCert data. It sits beside chapter 3954, chapter 731, chapter 737, chapter 936, and chapter 3439. The packet names the accepted endpoint budget that consistency consumers may inspect; it does not erase the obstruction against blanket strong normalization.

Definition 3312.1 (MetaCIC normal-endpoint budget carrier)**.** *A* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, K, B, F, O, H, C, P, N, G)$$

with the following displayed rows:

1. *T is the typed closed-term or typed fragment row whose normal endpoint is being budgeted.*
2. *K is the candidate-evidence row, read before any normalization-facing consumer sees an endpoint.*
3. *B is the bounded-normal endpoint row, recording the finite fuel, reduction trace, and bounded checker evidence available for T.*
4. *F is the finished-normal seal row, exposing the normal endpoint only after candidate evidence and bounded computation rows have been displayed.*
5. *O is the obstruction row, recording the retained subject-reduction or confluence boundary when the endpoint budget is unavailable for a wider theorem.*
6. *H records componentwise* hsame *transport for typed rows, candidate evidence, bounded endpoints, finished-normal seals, obstruction rows, replay, provenance, local naming, and gate rows.*
7. *C records displayed* Cont *replay from endpoint requests through T, K, B, F, O.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$*, the MetaCIC strong-normalization boundary, the subject-reduction obstruction, the confluence-normal fragment,* $\mathsf{ZNormal}^{\uparrow}$*, the bounded-normal conversion bridge,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}}$ *row.*

10. *G is the TasteGate landing row requiring* Nontrivial *and* FieldFaithful *instances for the displayed carrier surface before the packet is treated as a closed audit object.*

The classifier compares accepted packets only through $T, K, B, F, O, H, C, P, N, G$. It has no coordinate for evidence-free strong normalization, unrestricted subject reduction, quotient conversion equality, hidden fuel, or an unbounded normalizer.

Theorem 3312.2 (MetaCIC normal-endpoint budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *carrier $E = (T, K, B, F, O, H, C, P, N, G)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointBu}}$ *surface consists exactly of candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction retention,* hsame *transport,* Cont *replay,* Pkg *provenance, local naming, and TasteGate landing obligations. Every consistency-facing consumer must display those rows before it reads the endpoint budget.*

Proof. Project the accepted carrier. The typed row T is only a source row; it is not a normalization theorem by itself.

The candidate row K is read before the bounded endpoint row B. The finished-normal seal F is downstream of both rows, so a consumer cannot reach the normal endpoint without candidate evidence and bounded computation evidence.

The obstruction row O remains part of the same packet. It records the subject-reduction or confluence boundary that blocks reading the endpoint budget as a blanket strong-normalization theorem.

Finally H, C, P, N, G transport, replay, source, name, and gate the same carrier. Since the classifier has no hidden fuel coordinate, unbounded normalizer, quotient conversion equality, or evidence-free theorem row, the listed obligations exhaust the endpoint budget surface. □

Theorem 3312.3 (MetaCIC normal-endpoint budget consistency boundary)**.** *Every consistency-facing read from an accepted* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *packet factors through the finite route from chapter 3954 through chapter 731, chapter 737, chapter 936, and chapter 3439, as mediated by the rows $T, K, B, F, O, H, C, P, N, G$. Thus the endpoint can support a bounded normal-consistency read only while candidate evidence, obstruction retention, and finished-normal sealing remain displayed.*

Proof. Use the obligation theorem to expose the row order. Candidate evidence K and bounded endpoint evidence B precede the finished-normal seal F.

The obstruction row O remains visible between the strong-normalization boundary and the subject-reduction or confluence frontier. Therefore the route through the confluence-normal fragment, $\mathsf{ZNormal}^{\uparrow}$ endpoint, and bounded-normal conversion bridge is a finite endpoint budget, not a full subject-reduction or strong-normalization theorem.

Transport, replay, provenance, local naming, and the TasteGate landing row preserve the same dependency order. A consistency consumer therefore reads the normal endpoint only as a displayed budget with retained obstruction evidence. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ packet must display candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction, hsame, Cont, Pkg, and NameCert rows before any consistency consumer reads it.

Independence witness: metacic-strong-normalization-boundary, metacic-subject-reduction-obstruction, metacicconfluencenormalfragment, znormal, bounded-normal-conversion-bridge: the normal-endpoint budget consumes those sibling frontiers as independent rows and does not collapse them into a blanket strong-normalization or subject-reduction theorem.

The Lean TasteGate landing condition for MetaCICNormalEndpointBudget$^{\uparrow}$ includes explicit Nontrivial and FieldFaithful instances for the displayed carrier surface.

Closure status for NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$.

Constructive story: A MetaCICNormalEndpointBudget$^{\uparrow}$ packet is generated from typed MetaCIC source rows, candidate evidence, bounded-normal endpoint evidence, a finished-normal seal, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert data, and TasteGate landing rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$, seed)

Scope closed: Definition 4292.1, Definition 4292.2, and Definition 4292.3 seed the finite normal-endpoint budget for consistency-facing MetaCIC consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, blanket strong normalization, unrestricted subject reduction, quotient conversion equality, hidden fuel, or an unbounded normalizer.

Upgrade path: Obligation closure requires checked candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction retention, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful rows for BEDC.Derived.MetaCICNormalEndpointBudgetUp.

Chapter 3313

A Concrete Naming Certificate for $\mathsf{PolishCauchyBasis}^{\uparrow}$

Following Alexandrov–Urysohn, Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet records the basis-level handoff by which a complete separable metric source exposes Cauchy-filter basis data before any completion consumer reads a limit. It sits between the $\mathsf{PolishSpace}^{\uparrow}$ source of chapter 1490, the $\mathsf{CauchyFilterBasis}^{\uparrow}$ generator of chapter 1501, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff of chapter 128, and the $\mathsf{SeparableMetric}^{\uparrow}$ dense schedule of chapter 1458. The packet is a finite BEDC handoff surface for Polish-space Cauchy bases, not a quotient completion, not a choice-selected dense sequence, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3313.1 (Polish Cauchy basis carrier)**.** *A* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, B, K, D, S, R, O, H, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row, carrying the complete separable metric packet over which every later basis read is interpreted.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row, recording the directed finite basis generator used by Cauchy-filter consumers.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row attached to the same metric surface as X; it records the completion-facing route without selecting a limit point.*
4. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule row, stored as public finite name data rather than as a choice function over all balls.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect dense names, basis requests, and Cauchy-window demands.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those stream windows, giving regular rational observations before any basis comparison is made.*
7. *O is the observation ledger joining dense-window requests, regular-rational readbacks, Cauchy-basis refinements, and completion-handoff checks.*
8. *H records componentwise* hsame *transport for X, B, K, D, S, R, O and for the local certificate rows.*

9. *C records displayed* Cont *replay from a Polish-source request through dense schedules, stream windows, regular readbacks, Cauchy-basis refinement, and the completion handoff.*

10. *G is the* Pkg *provenance over* $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only after transporting these displayed rows to a common Polish source, Cauchy-basis generator, completion-handoff row, dense schedule, stream-window row, and regular readback ledger. It has no coordinate for choice-selected dense points, quotient completion equality, host point equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PolishCauchyBasisUp.PolishCauchyBasisTasteGate single carrier alignment`

Theorem 3313.2 (Polish Cauchy basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier* $P = (X, B, K, D, S, R, O, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Polish source, Cauchy-filter basis, complete-metric handoff, separable dense schedule, stream windows, regular rational readback, observation ledger, transport, continuation, provenance, and local naming rows;*

2. *common-source classification, requiring the B, K, D, S, R, O rows to be read over the same transported* $\mathsf{PolishSpace}^{\uparrow}$ *source row X;*

3. *basis-directedness exposure, requiring every Cauchy-basis refinement consumed by B to be visible in S, R, O before the completion handoff K is read;*

4. *dense-schedule compatibility, requiring the separable row D and the Cauchy-basis row B to share the same stream-window and regular-readback surface;*

5. *completion-handoff non-escape, requiring every completion-facing consumer to pass through B, S, R, O before reaching K;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4293.1. The projection exposes the Polish source, Cauchy-basis generator, complete-metric handoff, dense schedule, stream windows, regular readbacks, observation ledger, transport rows, continuation rows, provenance row, and local name.

The classifier route starts at X. Componentwise transport in H moves B, K, D, S, R, O together, so a consumer never compares a basis row over one Polish source with a completion handoff or dense schedule over another.

The observation ledger O fixes the finite read order. Dense requests and basis refinements are observed through S and R before either the Cauchy-filter basis or complete-metric handoff can be consumed.

Finally replay C and attach G, N. These rows route, source, and name the same displayed packet, with no additional coordinate for selected dense points, quotient completion equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the listed rows exhaust the local naming surface. □

Theorem 3313.3 (Polish Cauchy basis completion handoff)**.** *Let $P = (X, B, K, D, S, R, O, H, C, G, N)$ be an accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *packet. Every completion-facing consumer of the Polish source*

X reaches the $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K only through the displayed Cauchy-filter basis row B, the separable dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger O, transported by H and replayed by C.*

The visible route is

$$X, D, S, R, B, O \longmapsto K \longmapsto H, C, G, N.$$

It exports a finite completion handoff for downstream completion, Baire, and fixed-point packets; it does not export a quotient completion class, a selected dense sequence, host point equality, or an ambient real-completeness principle.

Proof. Read X from the carrier in Definition 4293.1. The Polish source supplies the common complete separable metric surface, while D, S, R display the finite dense-window observations available to the packet.

Use the NameCert obligations of Definition 4293.2. The basis-directedness and dense-schedule compatibility clauses force the Cauchy-filter basis row B to be read through the same stream windows, regular readbacks, and observation ledger as the dense schedule.

The completion-handoff clause then places K after B, S, R, O. Transport by H and replay by C preserve that order, while G, N package and name the same route. Therefore every completion-facing consumer receives only the displayed finite handoff and cannot bypass it through choice, quotient equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet is generated from a PolishSpace source, CauchyFilterBasis basis, CompleteMetric handoff, SeparableMetric dense schedule, StreamName windows, RegSeqRat readback, an observation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, seed)

Scope closed: Definition 4293.1, Definition 4293.2, and Definition 4293.3 seed the basis-level Cauchy handoff from Polish-space data to completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completions, host point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source classification, basis-directedness exposure, dense-schedule compatibility, completion-handoff non-escape, transport, replay, provenance, and local naming for the same packet.

Chapter 3314

A Concrete Naming Certificate for $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$

The $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ packet realizes the hidden Q_6 stabilizer and orbit-persistence rows of Definition 9462.1352, Definition 9462.1354, Definition 9462.1356, and Definition 9462.1362 as finite BEDC NameCert data. It consumes the six-coordinate codon window of chapter 2234, the boundary tile packet of chapter 3393, the codon lift-orbit classifier of chapter 4059, and the median-closure surface of chapter 3545. The packet exposes support, stabilizer, orbit, and median-closure rows before any consumer may claim the hidden C_2 persistence or the restored $S_3 \times C_2$ surface.

Definition 3314.1 (Symmetry restoration stabilizer carrier)**.** *A* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, A, T, U, M, O, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row, including the six bit coordinates inspected by the stabilizer calculation.*
2. *A is the active-support row for $\mathcal{R}$ and its inactive-anchor closure, read through the boundary-tile packet.*
3. *T is the hidden C_2 stabilizer row, naming the identity and the coordinate transposition $\tau : s_1 \leftrightarrow f_2$ from Definition 9462.1352.*
4. *U is the orbit-persistence row, recording the active-support, inactive-anchor, and first-median-layer orbits of τ from Definition 9462.1354.*
5. *M is the median-closure stabilizer row, recording the displayed $S_3 \times C_2$ restoration surface of Definition 9462.1356.*
6. *O is the orbit ledger joining the nontrivial C_2 orbits, the median-closure orbit table, and the sealing-corner orbit-completion row.*
7. *R is the refusal row separating support symmetry from weighted reassignment symmetry, and excluding biological causality, stochastic fitness, host quotient codon semantics, and hidden enumeration oracles.*

8. *H records componentwise* hsame *transport for Q, A, T, U, M, O, R and for the local certificate rows.*

9. *C records displayed* Cont *replay from a support read through the stabilizer row, orbit-persistence row, median-closure row, orbit ledger, and refusal row.*

10. *P is the* Pkg *provenance over* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}$ $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStabilizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, T, U, M, O, R, H, C, P, N$. It has no coordinate for an unstated support set, an unrecorded stabilizer enumeration, a host quotient codon object, a biological mechanism, or a weighted-symmetry substitute for the support stabilizer.

Lean4 ✓ `BEDC.Derived.SymmetryRestorationStabilizerUp.SymmetryRestorationStabilizerTasteGate`
`single carrier alignment`

Theorem 3314.2 (Symmetry restoration stabilizer NameCert obligations)**.** *For every accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier $S = (Q, A, T, U, M, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStab}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Q_6 codon window, active support, hidden C_2 stabilizer, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, transport, continuation, provenance, and local naming row;*

2. *support visibility, requiring every stabilizer or orbit read to expose the active-support row A before reading T, U, M, or O;*

3. *stabilizer exactness, requiring the hidden C_2 row T to name only the identity and $\tau : s_1 \leftrightarrow f_2$ for the active-support and first-median-layer reads;*

4. *median-closure compatibility, requiring the restored $S_3 \times C_2$ row M to be read only after the median-closure surface and its orbit ledger are displayed;*

5. *orbit non-escape, requiring any hidden C_2 orbit claim to pass through $A, T, U, M, O, H, C, P, N$;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4294.1. The projection exposes the codon window, active support, stabilizer row, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, and the structural rows H, C, P, N.

Support visibility comes first. The row A is the source for the stabilizer calculation, so T, U, M, and O cannot be read as free group-theoretic objects detached from the displayed support.

The stabilizer exactness row is T. It records exactly the identity and the transposition $\tau : s_1 \leftrightarrow f_2$ for the active-support surface, while U records the same nonidentity element along the inactive-anchor and first-median-layer rows.

The median-closure compatibility row is M, with orbit data stored in O. Transport by H and replay by C preserve this order, and P, N package and name the same rows. Thus no hidden orbit claim escapes the displayed support, stabilizer, median-closure, transport, replay, provenance, and local naming surface. □

Theorem 3314.3 (Symmetry restoration orbit non-escape)**.** *Every consumer claiming the hidden C_2 orbit of an accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *packet must display the support row A,*

the stabilizer row T, the orbit-persistence row U, the median-closure row M, the orbit ledger O, componentwise transport H, displayed replay C, package provenance P, and local NameCert *row N. The visible route is*

$$Q, A \longmapsto T, U \longmapsto M, O \longmapsto H, C, P, N.$$

It cannot be replaced by a host stabilizer calculation, an unrecorded orbit enumeration, a weighted-reassignment symmetry, quotient codon semantics, or a biological mechanism claim.

Proof. Read Q, A from the carrier in Definition 4294.1. These rows provide the finite codon-window coordinates and the active support over which the stabilizer calculation is meaningful.

Apply the NameCert obligations of Definition 4294.2. Support visibility and stabilizer exactness place T and U after A, while median-closure compatibility places M and O after the displayed closure surface.

Finally use orbit non-escape. The rows H, C, P, N transport, replay, source, and name the same finite route, and the refusal row separates it from weighted reassignment symmetry or host enumeration. Therefore a hidden C_2 orbit consumer must pass through the displayed packet and cannot claim the orbit from outside it. □

Falsifiable prediction: Any consumer claiming the hidden C_2 orbit without displaying support, stabilizer, median-closure, hsame, Cont, Pkg, and NameCert rows refutes the SymmetryRestorationStabilizer$^{\uparrow}$ packet.

Independence witness: q-six orbit/classification, median-closure sibling slugs: SymmetryRestorationStabilizer$^{\uparrow}$ is not bijective to the orbit/classification or median-closure siblings because this packet jointly carries active support, hidden C_2 stabilizer, orbit persistence, restored median-closure stabilizer, orbit ledger, refusal, componentwise transport, replay, provenance, and local naming as one accepted surface.

The Lean TasteGate file for this packet must include Nontrivial and FieldFaithful instances. The field-faithfulness witness keeps $Q, A, T, U, M, O, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out hidden-orbit claims that omit the support, stabilizer, or median-closure coordinates.

Closure status for NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$.

Constructive story: A SymmetryRestorationStabilizer$^{\uparrow}$ packet is generated from Q6 codon-window rows, active-support rows, hidden C2 stabilizer rows, orbit-persistence rows, median-closure stabilizer rows, orbit ledgers, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, seed)

Scope closed: Definition 4294.1, Definition 4294.2, and Definition 4294.3 seed the finite support-stabilizer packet for the hidden Q6 orbit and median-closure restoration surface.

Formal verification: FormalStatus(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological causality, stochastic fitness, quotient codon semantics, host stabilizer enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, support visibility, stabilizer exactness, orbit persistence, median-closure compatibility, orbit non-escape, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.SymmetryRestorationStabilizerUp.

Chapter 3315

A Concrete Naming Certificate for $\mathsf{TaggedRealPartition}^{\uparrow}$

Following Bishop and Bridges constructive integration, the $\mathsf{TaggedRealPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed tag inside each cell. It refines the located partition carrier of chapter 2248 and the oscillation-controlled partition carrier of chapter 3800; it also reads its endpoint and tag data through the $\mathsf{StreamName}^{\uparrow}$ source of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the finite tag-bearing surface needed by L10 integration and oscillation routes without replacing it by an untagged mesh, a host partition object, selected representatives, quotient real equality, countable choice, or ambient completeness.

Definition 3315.1 (Tagged real partition carrier)**.** *A* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, M, A, G, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row exposing the located interval, finite cell order, endpoint containment, and mesh precision.*
2. *M is the finite mesh row inherited from I, listing adjacent cells and the common refinement ledger used by later integration consumers.*
3. *A is the cell-containment row, certifying for every mesh cell that its endpoints and rational witness rows remain inside the located interval source.*
4. *G is the tag row family, assigning to every displayed cell a tag history whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads certify containment in that same cell before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *O is the oscillation and modulus ledger, recording the tolerance request, the cellwise oscillation bound, and the refinement compatibility supplied by* $\mathsf{FiniteOscillationPartition}^{\uparrow}$*.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for interval endpoints, cell endpoints, and tags.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and mesh precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the source interval, cell containment, tag containment, oscillation ledger, window, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for I, M, A, G, O, S, R, E and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tagged-partition read through*

$$I, M, A, G, O, S, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed interval-source, mesh, containment, tag, oscillation, window, readback, seal, transport, replay, provenance, and local naming rows. It has no coordinate for a host tagged partition, an unbounded tag selector, quotient real equality, chosen completed tag values, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.TaggedRealPartitionUp.TaggedRealPartitionTasteGate single carrier alignment`

Theorem 3315.2 (Tagged real partition NameCert obligations)**.** *For every accepted* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier $T = (I, M, A, G, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval and mesh exposure through I, M, cell-containment certification through A, tag containment through G, oscillation and modulus accounting through O, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seals through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every integration-facing or oscillation-facing read of the tagged partition must therefore pass through the displayed finite mesh, tag-containment, oscillation, window, readback, and seal rows before any terminal real-value consumer is allowed to inspect the packet.

Proof. Project the carrier of Definition 4296.1. The tuple lists the only rows available to a tagged-partition consumer: the interval source, mesh, containment ledger, tag rows, oscillation ledger, finite windows, regular readbacks, real seals, transport, replay, provenance, and local naming rows.

The source interval and cell obligations are I, M, A. The located partition row supplies the interval and mesh; the containment row attaches each cell to the same source, so a tag cannot be read against a detached host interval.

The tag and oscillation obligations are G, O. The tag row certifies cell membership for each tag through the same finite-window and regular-readback route, while the oscillation row records the tolerance and refinement ledger inherited from the oscillation-partition packet.

The readback and seal obligations are S, R, E. A consumer reaches the StreamName windows only after the interval, cell, tag, and oscillation rows are visible, then reads the regular rational rows and finally the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name exactly this route. Since the carrier has no coordinate for a host tagged partition, an unbounded selector, quotient real equality, chosen completed tag values, countable choice, or ambient completeness, the displayed NameCert obligations exhaust the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaggedRealPartition}^{\uparrow}$ packet is generated from a LocatedRealIntervalPartitionUp source, finite mesh and cell-containment rows, per-cell tag rows, oscillation and modulus ledgers, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, seed)

Scope closed: Definition 4296.1 and Definition 4296.2 seed the finite tag-bearing partition interface for L10 integration and oscillation consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close integration convergence, Darboux sums, quotient real equality, selected tag values, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cell containment, tag containment, oscillation-modulus accounting, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TaggedRealPartitionUp.

Chapter 3316

A Concrete Naming Certificate for $\mathsf{FiniteFanBarModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges finite-bar analysis, the $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet records the finite carrier that turns admitted fan-bar rows and local modulus rows into a tolerance-only uniform-modulus ledger. It sits over the fan certificate of chapter 400, the compact metric finite-net surface of chapter 129, the continuous-map carrier of chapter 130, the rational lower-bound fold of chapter 4118, the $\mathsf{Rat}^{\uparrow}$ arithmetic source of chapter 37, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. It realizes the compact-uniform-continuity route of chapter 9237 without importing open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, or ambient real completeness.

Definition 3316.1 (Finite fan-bar modulus carrier)**.** *A* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, L, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite fan-bar admission row, recording the fan, decidable bar classifier, finite depth bound, and finite branch-prefix witness window.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-probe row, aligning the bar prefixes with the compact finite-net probes consumed by the compact-uniform-continuity route.*
3. *L is the local modulus row read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *packet at each admitted bar or probe entry.*
4. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *radius and tolerance row, carrying the finitely many local radii and requested tolerances.*
5. *R is the lower-bound modulus extraction row, obtained by primitive recursion over the finite fan-bar and probe bundle and by certified rational comparison in Q.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, exposing the extracted lower bound as a tolerance-only modulus ledger.*
7. *H records componentwise* hsame *transport for the bar, compact-probe, local-modulus, rational, extraction, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$B, K, L, Q \longmapsto R \longmapsto U.$$

9. *P is the* Pkg *provenance over* FiniteFanBarModulus$^\uparrow$, FanTheorem$^\uparrow$, CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, Rat$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, K, L, Q, R, U, H, C, P, N$. It has no coordinate for an open cover, a selected infinite branch, countable choice, quotient equality of maps, selected limits, a host compactness theorem, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteFanBarModulusUp.FiniteFanBarModulusTasteGate single carrier alignment`

Theorem 3316.2 (Finite fan-bar modulus NameCert obligations)**.** *For every accepted* FiniteFanBarModulus$^\uparrow$ *carrier $F = (B, K, L, Q, R, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *surface consists exactly of carrier admission, finite bar admission through B, compact finite-probe alignment through K, local continuous-map modulus exposure through L, positive rational radius accounting through Q, lower-bound modulus extraction through R,* UniformModulus$^\uparrow$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

In particular, a compact-uniform-continuity consumer may read a uniform modulus from this packet only after the finite bar, compact-probe, local-modulus, rational-radius, and lower-bound extraction rows have all been displayed.

Proof. Project the carrier of Definition 4297.1. The tuple displays exactly the finite bar, compact-probe, local-modulus, rational-radius, lower-bound extraction, uniform-modulus, transport, replay, provenance, and local naming rows.

Finite bar admission is the row B. It supplies the decidable bar classifier and finite depth witness, so the argument ranges over displayed finite prefixes rather than selected infinite branches.

The compact and local-modulus data are K, L, Q. The compact-probe row aligns finite bar prefixes with compact finite-net probes, the continuous-map row supplies local moduli at those probes, and the rational row stores the positive local radii and tolerance requests.

Lower-bound extraction is the row R. It is obtained by primitive recursion over the finite bar and probe rows, using the positive rational comparisons in Q to retain a certified lower bound for all admitted local radii.

Finally, U, H, C, P, N expose, transport, replay, source, and name that extracted lower bound as a UniformModulus$^\uparrow$ ledger. Since the carrier has no coordinate for open-cover compactness, branch selection, countable choice, quotient function equality, selected limits, host compactness, or ambient completeness, the displayed obligations exhaust the accepted packet. □

Theorem 3316.3 (Finite fan-bar lower-bound modulus extraction)**.** *Every compact-uniform-continuity read of an accepted* FiniteFanBarModulus$^\uparrow$ *packet factors through*

$$B, K, L, Q, R, U, H, C, P, N.$$

The extracted row R is the only lower-bound modulus source available to the UniformModulus$^\uparrow$ *consumer: it is computed from finite fan-bar admission, compact finite-probe alignment, local continuous-map moduli, and positive* Rat$^\uparrow$ *radius comparisons, and it exports no open-cover subcover, selected branch, selected limit, quotient map equality, or ambient compactness datum.*

Proof. Start with the NameCert obligations of Definition 4297.2. They require every compact-uniform-continuity consumer to read B, K, L, Q before the lower-bound row R is available.

The extraction step is finite. The bar depth and compact probe rows determine the finite index family, and the positive rational ledger compares the finitely many local radii by primitive recursion to produce a certified lower bound.

Once R is displayed, the uniform-modulus row U may consume it as a tolerance-only ledger. The structural rows H, C, P, N preserve and name the same route under componentwise transport and replay.

No excluded datum can be read from the carrier. Open-cover compactness, selected branches, selected limits, quotient map equality, countable choice, and ambient compactness are absent from $B, K, L, Q, R, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet is generated from finite fan-bar admission, compact finite-probe alignment, local ContinuousMap modulus rows, positive RatUp radius ledgers, lower-bound extraction, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4297.1, Definition 4297.2, and Definition 4297.3 seed the finite fan-bar lower-bound route consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar admission, compact finite-probe alignment, local modulus exposure, RatUp lower-bound extraction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteFanBarModulusUp.

Chapter 3317

A Concrete Naming Certificate for $\mathsf{EdelsteinFixedPoint}^{\uparrow}$

Following Edelstein's compact-metric fixed-point theorem and Bishop–Bridges constructive metric practice, the $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet records the finite NameCert surface for a strictly contractive self-map on a compact metric carrier. It sits after the metric source of chapter 57, the compact-metric finite inspection surface of chapter 129, the sequential map discipline of chapter 1466, the contraction comparison row of chapter 3529, and the displayed iteration route of chapter 2359. The packet names only the compact source, the strict distance-decrease row, the sequential-continuity row, the orbit comparison ledger, transport, replay, provenance, and local naming; it does not import choice, quotient completion, a selected host fixed point, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a Banach contraction ratio.

Definition 3317.1 (Edelstein fixed-point carrier)**.** *An* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, O, C, S, H, R, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier and the distance observations inspected by the fixed-point consumer.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row over X, exposing only the finite compactness windows and cluster-service rows accepted by the compact-metric certificate.*
3. *F is the self-map row, presented through the same metric carrier and never as an untracked host function.*
4. *O is the orbit row generated from F, recording the displayed Picard-style finite prefix consumed by the fixed-point search.*
5. *C is the strict contraction-comparison row: whenever the inspected points are classified as distinct by the metric row, the displayed distance from $F(x)$ to $F(y)$ is strictly below the displayed distance from x to y.*
6. *S is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *row for F, binding cluster reads of the orbit to the image row without adding a semantic continuity theorem outside the displayed packet.*
7. *H stores componentwise* hsame *transport for metric, compactness, self-map, orbit, contraction-comparison, sequential-continuity, continuation, provenance, and naming rows.*

8. *R stores displayed* Cont *replay from the orbit row to the compact cluster row, from the cluster row through sequential continuity, and from the strict comparison row to the fixed-point equality boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, O, C, S, H, R, P, N$. It has no coordinate for quotient completion, countable choice, a Banach contraction constant, an ambient complete real line, or a host-selected fixed point.

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointTasteGate single carrier alignment`

Theorem 3317.2 (Edelstein fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier $E = (X, K, F, O, C, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *surface consists exactly of metric carrier admission through X, compact-metric finite inspection through K, self-map admission through F, Picard-style orbit exposure through O, strict contraction comparison through C, sequential-continuity cluster transport through S, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointCarrier namecert obligations`

Proof. Project the carrier of Definition 4298.1. The metric row X and compact-metric row K are the only source rows available to the certificate, so every inspected point, distance comparison, and compactness read is routed through those two displayed surfaces.

The self-map row F and orbit row O provide the visible finite Picard-style prefix. The strict comparison row C is separate from the Banach contraction surface: it records pointwise distance decrease on distinct displayed pairs and carries no global contraction ratio.

The sequential-continuity row S is the only route from a compact cluster read of the orbit to an image read under F. Transport through H, replay through R, provenance through P, and naming through N preserve this same finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 3317.3 (Edelstein compact-route nonescape)**.** *Every fixed-point read exported by an accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *packet factors through the internal route*

$$X \longmapsto K \longmapsto O \longmapsto S \longmapsto C$$

together with F, H, R, P, N. The route may compare with the $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$ *neighbours as named dependency rows, but it does not consume a Banach contraction constant, quotient completion equality, selected subsequence principle, choice operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a fixed-point read. The carrier first exposes the metric source X and the compactness service K, so the consumer can only inspect displayed finite compact-metric windows and the point classifier attached to X.

The orbit row O supplies the visible Picard-style finite prefix for the self-map F. Compactness is consumed only through K, and the image of a cluster read is transported only through the sequential-continuity row S.

The strict comparison row C is then the only row that can rule out a non-fixed distinct-pair boundary. Since C contains no global ratio and the remaining rows H, R, P, N only transport, replay, source, and name the same finite data, the exported route cannot leave the displayed Edelstein packet for choice, quotient completion, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ fixed-point read factors through one MetricUp source row, one CompactMetricUp finite inspection row, one self-map row, one PicardIterationUp-style orbit row, one strict comparison row, one SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, sequentialcontinuity, banach-contraction, picard-iteration: $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-metric cluster service, strict pointwise distance decrease, and sequential-continuity image transport without a Banach ratio, selected endpoint, quotient completion, or ambient completeness coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite inspection row, a displayed self-map row, a PicardIterationUp-style orbit row, a strict distance-decrease comparison row, a SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4298.1, Definition 4298.2, and Definition 4298.3 bind the compact-metric strict-contraction route through the displayed orbit, cluster, sequential-continuity, and comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, uniqueness, quotient completion equality, Banach contraction constants, selected fixed points, choice principles, bridge checking, Lean verification, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric inspection, orbit exposure, strict comparison, sequential-continuity cluster transport, nonescape, replay, provenance, and local naming for BEDC.Derived.EdelsteinFixedPointUp

Chapter 3318

A Concrete Naming Certificate for $\mathsf{CompletionLadder}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompletionLadder}^{\uparrow}$ packet records the finite BEDC naming surface that ties the current Real and Completion route into one phase object. It sits over the dyadic tolerance ledger of chapter 291, the finite-window stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated-completion boundary of chapter 1612, and the completion-unit handoff of chapter 1527. The ladder is a finite dependency packet for the phase route; it is not quotient stream equality, not a selected limit, not countable choice, not trichotomy, not open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 3318.1 (Completion ladder carrier)**.** *A* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, W, R, E, S, U, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row naming the precision requests that enter the route.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row listing the finite observations read at those tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the dyadic tolerance, stream window, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *witness row that consumes the same displayed real-facing route as a separated completion boundary.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *unit row that records the source-to-completion handoff along the same finite route.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, W \longmapsto R \longmapsto E \longmapsto S, U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, S, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionLadderUp.CompletionLadderTasteGate single carrier alignment`

Theorem 3318.2 (Completion ladder NameCert obligations)**.** *For every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier $L = (D, W, R, E, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, componentwise transport, continuation replay, package provenance, and local naming rows;*
2. *tolerance-window compatibility, requiring each finite stream read in W to be indexed by a dyadic tolerance request in D before regular readback is inspected;*
3. *regular-readback exactness, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *read in R to consume only the displayed D, W route;*
4. *Real-seal order, requiring E to be read only after D, W, R have been displayed;*
5. *completion-boundary order, requiring S and U to consume the same D, W, R, E route through the stored continuation row C;*
6. *transport stability, requiring componentwise* hsame *transport in H to preserve the listed route and no other completion data;*
7. *ledger non-escape, excluding quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, and completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 4299.1. The tuple lists the dyadic tolerance row, stream-window row, regular readback row, Real seal, separated-completion witness, Cauchy-completion unit, transport rows, replay row, provenance, and local name.

The first dependency step reads D and W. Since W is the only stream-window coordinate and D is the only tolerance coordinate, each finite observation entering the regular readback must pass through those two rows.

The regular readback row R is then downstream of D, W, and the Real seal row E is downstream of R. The replay row C records this order before either completion-facing row is inspected.

The separated-completion witness S and Cauchy-completion unit U therefore consume the same displayed D, W, R, E route. Componentwise transport by H preserves those rows, while P and N source and name the same packet.

No remaining coordinate can export quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 3318.3 (Completion ladder separated Real route)**.** *Every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier factors any separated-completion or completion-unit consumer through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{SeparatedCompletion}^{\uparrow},\quad \mathsf{CauchyCompletionUnit}^{\uparrow}.$$

Equivalently, a consumer of S or U must first read the displayed dyadic tolerance row D, stream-window row W, regular readback row R, and Real seal row E, with componentwise transport, continuation replay, provenance, and local naming supplied by H, C, P, N.

Proof. Start from an accepted carrier $L = (D, W, R, E, S, U, H, C, P, N)$. The only completion-facing rows are S and U, so any separated-completion or completion-unit consumer must inspect one of those coordinates.

The definition places both coordinates after D, W, R, E. The continuation row C records the same order, forcing each consumer read to pass from dyadic tolerance and finite window through regular rational readback before the Real seal is available.

Componentwise hsame transport by H may rename the rows, but it preserves the displayed route rather than adding a second stream, a chosen limit, or a host completion object. The package and local name rows P, N then record that the consumer is reading the same finite packet.

Since the carrier contains no row for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the separated Real route above is the whole exported completion-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionLadder}^{\uparrow}$ packet is generated from DyadicRatCore tolerance requests, StreamName finite windows, RegSeqRat readback, a RealUp seal, a SeparatedCompletion witness, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4299.1, Definition 4299.2, and Definition 4299.3 seed the finite Real and Completion ladder over DyadicRatCore, StreamName, RegSeqRat, Real, SeparatedCompletion, CauchyCompletionUnit, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, bridge checking, Lean verification, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window compatibility, regular readback exactness, Real-seal order, separated-completion routing, completion-unit routing, componentwise transport stability, and ledger non-escape for BEDC.Derived.CompletionLadderUp.

Chapter 3319

A Concrete Naming Certificate for $\mathsf{LocatedFormalBar}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Vickers, the $\mathsf{LocatedFormalBar}^{\uparrow}$ packet records a finite located bar/cover boundary for spread paths and formal-topology cover requests. It sits after the cover/refinement carrier of chapter 3967, the admissible-prefix discipline of chapter 1532, the bar-recursion route of chapter 1675, and the finite observation interface of chapter 126. The packet supplies Real and Completion consumers with a finite bar boundary without importing open-cover compactness, selected points, quotient opens, countable choice, or ambient completeness.

Definition 3319.1 (Located formal-bar carrier)**.** *A* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FormalTopology}^{\uparrow}$ *cover/refinement row, read through chapter 3967, that names the finite basic neighbourhoods and refinement requests used by the bar.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admissible-prefix row, read through chapter 1532, that determines which finite path prefixes can enter the bar ledger.*
3. *B is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *bar-recursion row, read through chapter 1675, that records the finite induction surface consumed by bar-cover arguments.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, read through chapter 126, that exposes the inspected prefix windows for Real and Completion routes.*
5. *Q is the located-query row: each accepted request names a finite prefix, a formal basic neighbourhood, and the bar-cover decision boundary visible at that prefix.*
6. *R is the spread-cover row, recording the finite route from an admissible prefix through the formal cover row to the bar-recursion row.*
7. *H records componentwise* hsame *transport for formal-cover rows, spread-law prefixes, bar-recursion rows, StreamName windows, located-query rows, spread-cover rows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion finite-window request through $F, S, B, W, Q, R, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedFormalBar}^{\uparrow}$, $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite path, a selected point of a cover, quotient-open equality, countable choice, host topological equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedFormalBarUp.LocatedFormalBarTasteGate single carrier alignment`

Theorem 3319.2 (Located formal-bar spread-cover route)**.** *Every accepted* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

routes a Real or Completion finite-window consumer through the same displayed spread-cover boundary:

$$F, S, B, W \longmapsto Q, R \longmapsto H, C, P, N.$$

The formal-topology cover row F, spread-law row S, Brouwer bar-induction row B, and StreamName window row W must be visible before the located-query row Q and spread-cover row R are consumed. The route exports no open-cover compactness theorem, selected infinite branch, selected cover point, quotient-open equality, countable-choice schedule, host real equality, or ambient completeness datum.

Proof. Project the carrier of Definition 4300.1. The source rows are F, S, B, W, and the located bar/cover rows are Q, R.

First read the formal-topology and spread-law surfaces. The finite cover/refinement row F supplies only displayed basic neighbourhoods, while S supplies only admissible finite prefixes.

Next pass through the bar-recursion and StreamName rows. Row B keeps the bar argument on the finite Brouwer induction surface, and W fixes the inspected finite windows used by Real and Completion consumers.

Finally replay through Q, R, H, C, P, N. The located query and spread-cover rows are transported componentwise and replayed with the same provenance and local NameCert row. Since the carrier contains no coordinate for compact open covers, selected branches, selected points, quotient opens, choice, host equality, or ambient completeness, no consumer can read those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedFormalBar}^{\uparrow}$ packet is generated from FormalTopology cover/refinement rows, SpreadLaw admissible-prefix rows, BrouwerBarInduction bar-recursion rows, StreamName finite-window rows, located-query rows, spread-cover rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, seed)

Scope closed: Definition 4300.1 and Definition 4300.2 seed the finite located bar/cover boundary for Real and Completion consumers over formal covers, spread prefixes, bar recursion, finite windows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, selected cover points, quotient-open equality, countable choice, host topological equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formal-cover exposure, spread-law prefix exposure, bar-recursion routing, StreamName finite-window replay, located-query exactness, spread-cover exactness, transport, provenance, and local naming for BEDC.Derived.LocatedFormalBarUp.

Chapter 3320

A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$

Following Bishop and Bridges, the CauchyModulusFunctoriality$^{\uparrow}$ packet records the finite data needed to transport displayed Cauchy moduli along a BEDC route without turning that route into a host function equality. It sits between the dyadic tolerance source DyadicRatCore$^{\uparrow}$ of chapter 291, the finite observation route StreamName$^{\uparrow}$ of chapter 126, the regular rational readback RegSeqRat$^{\uparrow}$ of chapter 301, the real-name seal Real$^{\uparrow}$ of chapter 38, the pullback discipline CauchyModulusPullback$^{\uparrow}$ of chapter 1951, and the regular stability surface RegularCauchyModulusStability$^{\uparrow}$ of chapter 1927.

Definition 3320.1 (Cauchy modulus functoriality carrier)**.** *A* CauchyModulusFunctoriality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy-modulus row, naming the inspected source modulus and its finite request boundary.*
2. *T is the target Cauchy-modulus row, naming the target modulus after transport along the displayed route.*
3. *R is the route-map row, recording the finite BEDC route that carries source requests to target requests without asserting equality of host functions.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, read through chapter 291, that keeps each modulus comparison on dyadic rational bounds.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row, read through chapter 126, that records the observation window used by the transported modulus request.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, read through chapter 301, that turns the transported window into a regular rational comparison.*
7. *E is the* Real$^{\uparrow}$ *seal, read through chapter 38, that exposes only the terminal real-name boundary reached by the finite route.*
8. *H records componentwise* hsame *transport for source modulus, target modulus, route map, dyadic tolerance ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a transported modulus request through $S, T, R, D, W, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* CauchyModulusFunctoriality$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStal DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function equality, a selected limit, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusFunctorialityUp.CauchyModulusFunctorialityTasteGate single carrier alignment`

Theorem 3320.2 (Cauchy modulus functoriality NameCert obligations)**.** *For every accepted* CauchyModulusFunctoriality$^{\uparrow}$ *carrier*

$$F = (S, T, R, D, W, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *obligations are exactly the finite route obligations listed by the carrier: source-modulus admission, target-modulus admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The route may use the sibling certificates for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, *and* RegularCauchyModulusStability$^{\uparrow}$, *but it exports no host function equality, selected limit, quotient real equality, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 4301.1. The source and target modulus rows are S, T, and the finite transport route is the displayed row R.

The numeric side is then forced through D, W, Q. The dyadic ledger D supplies the tolerance comparisons, the StreamName window W supplies the finite observation boundary, and the RegSeqRat row Q supplies the regular rational readback.

The terminal read is sealed by E. It records only the Real-facing boundary reached by the same finite route, so it cannot introduce a selected limit or quotient-real equality.

Finally transport and replay through H, C, P, N. These rows preserve the same carrier components under hsame, Cont, package provenance, and local naming. Since the carrier contains no coordinates for host function equality, selected limits, quotient real equality, trichotomy, countable choice, or ambient completeness, the listed NameCert obligations exhaust the packet. □

Closure status for NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$.

Constructive story: A CauchyModulusFunctoriality$^{\uparrow}$ packet is generated from source and target Cauchy-modulus rows, a finite route-map row, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, seed)

Scope closed: Definition 4301.1 and Definition 4301.2 seed the finite modulus-transport certificate over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStability$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, selected limits, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFunctorialityUp.

Chapter 3321

A Concrete Naming Certificate for $\mathsf{RealLimitStability}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealLimitStability}^{\uparrow}$ packet records the finite data by which two displayed terminal $\mathsf{Real}^{\uparrow}$ limit seals remain stable under shared regular-Cauchy evidence. It sits after the terminal Real seal of chapter 38, the regular rational readback discipline of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the completion witness of chapter 3579. The packet names stability of displayed limit seals; it is not quotient stream equality, selected host limits, trichotomy, countable choice, or ambient Real completeness.

Definition 3321.1 (Real limit stability carrier)**.** *A* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two limit-presentation rows whose terminal seals are compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both presentations are inspected.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the shared window.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the common dyadic budgets used by the comparison.*
5. E_0 *and* E_1 *are the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after* L_0, L_1, W, R, D *have been displayed.*
6. H *records componentwise* hsame *transport for both limit presentations, the shared window, readback, tolerance ledger, paired Real seals, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a stability request through* $L_0, L_1, W, R, D, E_0, E_1, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealLimitStability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate single carrier alignment`

Theorem 3321.2 (Real limit stability NameCert obligations)**.** *For every accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier* $S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite packet, classifier stability under componentwise* hsame*, shared-window readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic tolerance nonescape through* $\mathsf{DyadicRatCore}^{\uparrow}$*, paired terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4303.1. The accepted packet exposes exactly two presentation rows, one shared window, one regular rational readback, one dyadic tolerance ledger, two terminal Real seals, transport, replay, provenance, and local naming.

Carrier admission and classifier stability are componentwise. The transport row H moves the two presentations, shared evidence, paired seals, and certificate rows together, while the replay row C preserves the same order of inspection for every stability request.

The provenance row P cites only the listed sibling surfaces and the local row N names the packet. Since the carrier contains no quotient equality, selected host limit, trichotomy oracle, choice schedule, ambient completeness coordinate, Lean certificate, or bridge row, the listed obligations exhaust the NameCert surface. □

Theorem 3321.3 (Real limit stability seal stability)**.** *Every comparison of the paired terminal seals in an accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *packet factors through the displayed route*

$$L_0, L_1 \longmapsto W \longmapsto R \longmapsto D \longmapsto E_0, E_1.$$

Thus the two $\mathsf{Real}^{\uparrow}$ *seals may be reused only under the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger displayed by the packet.*

Proof. Start with a consumer that reads the paired Real seals. The only Real-facing terminal coordinates in the carrier are E_0 and E_1, and both appear after the two presentation rows, shared finite window, regular rational readback, and dyadic ledger.

The consumer must therefore pass through W, R, D before either seal can be used. These rows give the finite observations, regular Cauchy readback, and common tolerance budget that make the two terminal seals comparable as displayed BEDC data.

Finally read H, C, P, N. These rows transport, replay, source, and name the same route, so they preserve seal stability without adding a selected host limit or quotient stream equality. □

Theorem 3321.4 (Real limit stability nonescape)**.** *No accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier exports quotient stream equality, a selected host limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a detached Real seal, Lean verification, or bridge checking. Any downstream completion consumer receives only the two limit presentations, shared window, regular readback, dyadic ledger, paired seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

Proof. Inspect the carrier inventory. The tuple contains no coordinate for quotient equality, selected host limits, trichotomy, choice, ambient completeness, detached seal data, Lean verification, or a bridge certificate.

The paired seals E_0, E_1 are downstream of the shared witness rows W, R, D. Hence a completion consumer can reuse the seals only through the displayed finite evidence and not through an external selected limit.

Transport, replay, provenance, and local naming preserve the same boundary. The rows H, C, P, N move and name only the finite packet, so every exported object remains inside the displayed Real-limit-stability surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitStability}^{\uparrow}$ packet is generated from two limit-presentation rows, a shared StreamName window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4303.1, Definition 4303.2, Definition 4303.3, and Definition 4303.4 seed the finite stability packet for paired terminal Real limit seals under shared StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, shared-window readback, dyadic ledger nonescape, paired Real-seal stability, transport, replay, provenance, and local naming for BEDC.Derived.RealLimitStabilityUp.

Chapter 3322

A Concrete Naming Certificate for $\mathsf{DecimalIntervalReal}^{\uparrow}$

Bishop–Bridges constructive analysis uses rational and decimal brackets as finite evidence before a real name is treated as complete. The $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet records this decimal-bracket carrier as BEDC NameCert data. It sits after the decimal prefix surface of chapter 1417, the finite decimal stream of chapter 4191, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval source of chapter 1709. The packet gives L10 Real and Completion consumers a finite decimal interval carrier without quotient reals, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3322.1 (Decimal interval real carrier)**.** *A* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower finite decimal-prefix row, read through* $\mathsf{DecimalExpansion}^{\uparrow}$ *and the finite* $\mathsf{DecimalStream}^{\uparrow}$ *schedule.*
2. *B is the upper finite decimal-prefix row, read through the same decimal surfaces and carrying the advertised bracket side.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating decimal place values into dyadic comparison margins.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for endpoint-prefix reads, bracket comparison, and downstream replay.*
5. *Q is the bracket-order row, recording that the lower decimal prefix and upper decimal prefix are compared only through the displayed tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint windows, decimal prefix rows, dyadic ledger, and bracket-order row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback has been displayed.*
8. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *handoff row that consumes the same endpoint, tolerance, readback, and seal rows as a located interval boundary.*

9. *H records componentwise* hsame *transport for endpoint prefixes, tolerance rows, windows, bracket order, regular readback, real seal, located interval handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a decimal-bracket request through endpoint prefixes, dyadic comparison, finite windows, regular readback, real sealing, and located interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DecimalIntervalReal}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for a quotient real, a trichotomy decision, a selected decimal representative, host endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier`
Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealTasteGate single carrier alignment`

Theorem 3322.2 (Decimal interval real NameCert obligations)**.** *For every accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower decimal prefix, upper decimal prefix, dyadic tolerance ledger, stream windows, bracket-order row, regular rational readback, real seal, located interval handoff, transport, replay, provenance, and local naming rows;*
2. *decimal-prefix visibility, requiring endpoint reads to enter through A and B before any bracket comparison is inspected;*
3. *tolerance discipline, requiring the place-value comparison to use the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D and no hidden real-order oracle;*
4. *window discipline, requiring each endpoint and comparison read to be scheduled by the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback order, requiring R to consume A, B, D, W, Q before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *located-interval handoff, requiring L to consume the same endpoint, tolerance, readback, and seal route rather than a host interval;*
7. *componentwise* hsame *transport and displayed* Cont *replay for exactly the rows carried by H and C;*
8. *ledger non-escape, excluding quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4357.1. The projection exposes exactly the two decimal endpoint-prefix rows, the dyadic tolerance ledger, the finite window schedule, the bracket-order row, the regular readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming.

The endpoint reads are visible before comparison. Rows A and B carry the finite decimal prefixes, while D supplies the dyadic margin used to compare their place-value readings. The row W schedules the inspected prefixes and comparison windows, so the bracket-order row Q has no access to a hidden real-order decision.

The real-facing route is downstream of the finite bracket data. Row R is read only after A, B, D, W, Q have been displayed, and row E seals the result only after that regular rational readback.

Finally, the located interval handoff L consumes the same displayed route, while H, C, P, N transport, replay, source, and name the packet. Since no remaining coordinate stores quotient equality, trichotomy, a selected decimal representative, host endpoint equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier semantic name certificate`

Theorem 3322.3 (Decimal interval real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow}.$$

Equivalently, a downstream Real and Completion consumer receives the two decimal endpoint-prefix rows, the dyadic comparison ledger, the finite window row, the regular rational readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming before it can use the bracket.

Proof. Start with an accepted carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$. The only real-facing row is E, and the only located-interval row is L.

The definition places E after the endpoint-prefix rows A, B, the dyadic tolerance ledger D, the window schedule W, the bracket-order row Q, and the regular rational readback R. Thus a $\mathsf{Real}^{\uparrow}$ consumer must first inspect the decimal, dyadic, window, and readback rows.

The located interval handoff is not an alternate route. Row L consumes the same A, B, D, W, Q, R, E path, and componentwise transport through H preserves that path under hsame.

The replay, provenance, and local naming rows only record the displayed carrier. They do not add quotient reals, trichotomy, selected representatives, host endpoint equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed route is the whole exported handoff. □

Falsifiable prediction: Every accepted $\mathsf{DecimalIntervalReal}^{\uparrow}$ read displays two finite decimal-prefix rows, one dyadic comparison ledger, one finite StreamName window row, one RegSeqRat readback row, one RealUp seal row, one LocatedRealIntervalUp handoff row, and local transport, replay, provenance, and naming rows before it is consumed as a decimal bracket.

Independence witness: decimalexpansion, decimal-stream, dyadicratcore, streamname, regseqrat, real, located-real-interval: $\mathsf{DecimalIntervalReal}^{\uparrow}$ is not bijective to any listed sibling because it binds two ordered decimal endpoint-prefix rows, one dyadic comparison ledger, finite observation windows, regular rational readback, RealUp sealing, located interval handoff, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet is generated from lower and upper decimal-prefix rows, DecimalStream windows, a DyadicRatCore comparison ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, a LocatedRealIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4357.1, Definition 4357.2, and Definition 4357.3 seed the finite decimal-bracket route over DecimalExpansionUp, DecimalStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealIntervalUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DecimalIntervalRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix visibility, dyadic tolerance discipline, finite-window discipline, regular-readback order, Real seal handoff, located interval handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.DecimalIntervalRealUp.

Chapter 3323

A Concrete Naming Certificate for $\mathsf{CantorIntersectionMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet records the metric nested-closed-set intersection route as finite BEDC NameCert data over the L10 completion surface. It sits after the complete-metric source of chapter 128, the metric-completion route of chapter 1506, the nested closed-ball packet of chapter 3526, the Cantor complete metric route of chapter 2199, the Cauchy-filter interface of chapter 333, and the L10 real-name route through chapter 291, chapter 126, chapter 301, and chapter 38. The metric packet is the metric-completion specialization of the finite Cantor-intersection route in chapter 1421 and the theorem-facing intersection route in chapter 2318; it consumes their nested-window and diagonal-readback rows before exposing complete-metric service. The packet names a nested metric intersection handoff; it does not assert open-cover compactness, arbitrary closed-set compactness, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3323.1 (CantorIntersectionMetric carrier)**.** *A* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence service are the only complete-metric data available to the packet.*
2. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row naming the surrounding completion request before any intersection point is read.*
3. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row, recording the descending closed metric ball spine and its finite containment witnesses.*
4. *K is the* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *row, used only as the displayed nested closed-ball completion handoff.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested metric spine and read only through the displayed filter route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, storing the shrink schedule, late-radius comparisons, and requested precision bounds.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, radii, and containment data.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of the displayed center schedule.*
9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, K, F, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for the complete-metric source, completion route, nested-ball row, Cantor handoff, filter, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a nested metric intersection request through $M, E, B, K, F, D, S, R, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, B, K, F, D, S, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary closed-set compactness, quotient completion equality, a selected limit, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionMetricUp.CantorIntersectionMetricTasteGate single carrier alignment`

Theorem 3323.2 (CantorIntersectionMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the complete-metric and metric-completion source rows M, E before any intersection witness is read;*
2. *display the nested metric route through B, K, F, namely the nested closed-ball spine, Cantor complete-metric handoff, and generated Cauchy-filter row;*
3. *bind all numerical readback to the L10 rows D, S, R, L, namely the dyadic tolerance ledger, finite stream windows, regular rational handoff, and terminal real seal;*
4. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
5. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row is stored for open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, trichotomy, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4359.1. The rows M, E expose the metric and completion sources before the packet can read an intersection point.

The nested metric content is then displayed through B, K, F. The nested closed-ball row supplies the descending closed metric spine, the Cantor complete-metric row supplies the existing handoff, and the Cauchy-filter row records the generated filter route.

The L10 readback is downstream. The dyadic ledger D, stream windows S, regular rational row R, and real seal L are the only numerical coordinates available before a completion-facing consumer receives a named output.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains none of the excluded compactness, quotient, selected-limit, trichotomy, or ambient-completeness coordinates, the five displayed obligations exhaust the local NameCert surface. □

Theorem 3323.3 (CantorIntersectionMetric complete handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier factors through the displayed nested metric route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{NestedClosedBall}^{\uparrow},\ \mathsf{CantorCompleteMetric}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{S}$$

Thus a consumer receives the complete-metric service, completion request, nested closed-ball spine, Cantor complete-metric handoff, Cauchy-filter row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows before it may use the intersection output. The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness principle, open-cover compactness theorem, arbitrary closed-set compactness, quotient stream equality, or selected limit.*

Proof. Begin with a completion-facing read of the accepted carrier. The first available rows are M and E, so the read is grounded in the displayed complete-metric service and the surrounding metric-completion request.

Next pass through the nested route B, K, F. The descending closed-ball spine is read through the $\mathsf{NestedClosedBall}^{\uparrow}$ row, the existing Cantor complete-metric handoff is read through K, and the generated filter is read through F.

The numerical output is then forced through the L10 readback rows D, S, R, L. The dyadic ledger supplies tolerance control, the stream row supplies finite windows, the regular rational row supplies readback, and the terminal real seal is reached only after those rows are present.

The structural rows H, C, P, N preserve the same route for downstream consumers. Because no row supplies full real completeness, open-cover compactness, arbitrary closed-set compactness, quotient stream equality, or a selected limit, every completion-facing read factors through the displayed nested metric route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, a MetricCompletion request row, a NestedClosedBall spine, a CantorCompleteMetric handoff, a CauchyFilter row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4359.1, Definition 4359.2, and Definition 4359.3 seed the metric nested-closed-set intersection route over displayed completion, nested-ball, Cantor handoff, filter, dyadic, stream, regular readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested metric inclusion, Cantor handoff, Cauchy-filter generation, dyadic shrink exactness, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionMetricUp.

Chapter 3324

A Concrete Naming Certificate for $\mathsf{AttouchWetsTopology}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet records local Hausdorff-window convergence for closed metric figures as finite BEDC NameCert data over the metric-completion and L10 real-name surface. It sits after the metric carrier of chapter 57, the Hausdorff-distance surface of chapter 3653, the terminal real seal of chapter 38, the regular rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the finite observation topology; it does not import host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3324.1 (Attouch-Wets topology carrier)**.** *An* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose metric, point windows, and bounded-window radii are the only ambient metric data available to the packet.*
2. *H is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *local-window row, read only on displayed bounded metric windows rather than on an ambient hyperspace.*
3. *D is the dyadic tolerance ledger for window radius, Hausdorff comparison margin, and late-window precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, bounded windows, comparison stages, and requested precision cells.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Hausdorff-window comparisons obtained from M, H, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *Q records componentwise* hsame *transport for metric source rows, Hausdorff-window rows, dyadic ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming.*
8. *T records displayed* Cont *replay from an Attouch–Wets observation request through M, H, D, W, R, E.*

9. *C is the local bounded-window classifier row comparing packets only through the displayed metric source, Hausdorff-window, tolerance, stream, readback, seal, transport, replay, provenance, and naming coordinates.*
10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *row.*

The classifier has no coordinate for global closed-set topology, open-cover compactness, selected nearest points, quotient hyperspace equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AttouchWetsTopologyUp.AttouchWetsTopologyTasteGate single carrier alignment`

Theorem 3324.2 (Attouch-Wets topology NameCert obligation surface)**.** *For every accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *expose the* $\mathsf{Met}^{\uparrow}$ *source row M before any local Hausdorff comparison is read;*
2. *display the* $\mathsf{HausdorffDistance}^{\uparrow}$ *window row H only on bounded metric windows named by the packet;*
3. *bind all numerical precision to the dyadic tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*
4. *preserve the same finite route by componentwise* hsame *transport Q and displayed* Cont *replay T;*
5. *source and name the same packet through the bounded-window classifier C,* Pkg *provenance P, and the local* NameCert *row N.*

No obligation row is stored for a host hyperspace topology, arbitrary closed-set compactness, selected representatives, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4358.1. The metric row M is the only source of ambient distance data, and the Hausdorff row H is read only after the bounded window has been displayed.

The numerical part is downstream of that window. The tolerance ledger D, finite window row W, regular readback R, and real seal E are the only precision and value rows available to an observation request.

Transport and replay keep the same route. Rows Q and T preserve the displayed metric, Hausdorff, tolerance, window, readback, and seal data, while C, P, N classify, source, and name the finite packet.

Since the carrier contains none of the excluded topology, compactness, selection, quotient, choice, or ambient-completeness coordinates, the displayed obligations exhaust the local NameCert surface. □

Theorem 3324.3 (Attouch-Wets local Hausdorff window)**.** *Every topology-facing read exported by an accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{HausdorffDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}.$$

Thus a downstream metric-completion consumer receives the displayed metric source, bounded local Hausdorff window, finite observation schedule, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming before it can use Attouch–Wets convergence data.

Proof. Start with a topology-facing read of the accepted carrier. The only available metric source is M, and the only Hausdorff comparison row is the local-window row H.

The observation is then forced through D, W, R, E. The dyadic ledger fixes the requested tolerance, the stream row displays the inspected bounded windows, the regular rational row records the readback, and the real row seals the reported Hausdorff-window value.

Finally Q, T, C, P, N transport, replay, classify, source, and name that same finite route. Because no row supplies host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported topology-facing read factors through the displayed local Hausdorff window. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, a HausdorffDistance bounded-window row, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, a bounded-window classifier row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4358.1, Definition 4358.2, and Definition 4358.3 seed the local Hausdorff-window topology route over MetricSpaceUp, HausdorffDistanceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AttouchWetsTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Hausdorff-window exposure, finite-window scheduling, regular readback, Real sealing, transport, replay, classifier exactness, provenance, and local naming for BEDC.Derived.AttouchWetsTopologyUp.

Chapter 3325

A Concrete Naming Certificate for $\mathsf{DistanceToSet}^{\uparrow}$

Following Bishop and Bridges on located subsets in metric spaces, the $\mathsf{DistanceToSet}^{\uparrow}$ packet records the finite route by which a point-to-set distance request is read through displayed located-radius rows and closed-ball witnesses. It sits after the metric source of chapter 57, the located-set surface of chapter 1541, the metric closed-ball boundary of chapter 1451, the located closed-ball handoff of chapter 4063, and the terminal real-name seal of chapter 38. The packet names distance-to-set as finite BEDC data; it is not a host infimum, not a selected nearest point, not quotient real equality, and not a countable-choice construction.

Definition 3325.1 (Distance-to-set carrier)**.** *A* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, x, r, B, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the subset by finite located-radius observations.*
3. *x is the displayed point row in the metric source.*
4. *r is the dyadic radius ledger recording the inspected tolerance windows.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness row attached to the point and radius ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance name, read only after the located-set and closed-ball rows are displayed.*
7. *H records componentwise* hsame *transport for the metric source, located set, point row, dyadic ledger, closed-ball witness, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the metric source through located-set exposure, point inspection, dyadic-radius accounting, closed-ball handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, x, r, B, R, H, C, P, N$. It has no coordinate for a host infimum over all points of A, a selected nearest point, quotient real equality, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DistanceToSetUp.DistanceToSetTasteGate single carrier alignment`

Theorem 3325.2 (Distance-to-set NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier*

$$D = (X, A, x, r, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located set, point row, dyadic radius ledger, closed-ball witness, Real seal, transport, replay, provenance, and local naming;*
2. *located-set exposure, requiring every subset read to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *row A before a distance row is exported;*
3. *dyadic-radius monotonicity, requiring the tolerance ledger r to preserve the displayed radius order used by the located-set and closed-ball rows;*
4. *closed-ball handoff, requiring each ball membership or exclusion read to pass through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness B and the located closed-ball sibling surface before the Real seal is read;*
5. *Real seal non-escape, requiring R to seal only the displayed distance name and not a host infimum or quotient real;*
6. *componentwise transport, displayed replay, package provenance, and local* NameCert *naming for exactly the displayed packet.*

No DistanceToSet obligation is stored outside these rows.

Proof. Project the carrier of Definition 4364.1. The metric source is X, the located subset is A, the inspected point is x, and the finite tolerance information is held by r.

The closed-ball witness B is read only after those rows are displayed. Its radius requests are therefore attached to the same metric source, located-set row, point row, and dyadic ledger.

The Real seal R comes after the located-radius and closed-ball rows. Transport H and replay C preserve the same order, while P and N record provenance and local naming.

Since no coordinate stores a host infimum, a nearest-point selector, quotient real equality, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 3325.3 (Distance-to-set located-radius handoff)**.** *Every consumer of an accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *packet reads distance-to-set data only through the displayed located-radius row A, the dyadic radius ledger r, the closed-ball witness B, and the Real seal R, transported by H and replayed by C. The consumer cannot bypass these rows by reading a host infimum, a selected nearest point, quotient real equality, countable choice, or an ambient completeness principle.*

Proof. Read the carrier of Definition 4364.1. The only subset coordinate is the located-set row A, and the only radius coordinate is the dyadic ledger r.

The closed-ball witness B supplies the displayed ball boundary before the Real seal R is exposed. Thus a distance consumer reaches the exported real name only through the located-radius and closed-ball rows.

Apply Definition 4364.2. The transport, replay, provenance, and local naming rows preserve the same finite packet and add no host infimum, nearest-point selector, quotient real equality, countable choice, or ambient completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}$.

Constructive story: A $\mathsf{DistanceToSet}^{\uparrow}$ packet is generated from a metric source, located set row, point row, dyadic radius ledger, closed-ball witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4364.1, Definition 4364.2, and Definition 4364.3 seed the finite located-radius and closed-ball interface for distance-to-set consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host infimum construction, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-set exposure, dyadic-radius monotonicity, closed-ball handoff, Real seal non-escape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DistanceToSetUp.

Chapter 3326

A Concrete Naming Certificate for $\mathsf{JordanCurve}^{\uparrow}$

Following Jordan and constructive topology, the $\mathsf{JordanCurve}^{\uparrow}$ packet records the finite polygonal and winding-ledger route by which a displayed simple closed curve in the real plane may be consumed as a separation boundary. It sits after the terminal real-name seal of chapter 38, the circle carrier of chapter 58, the graph and modulus rows of chapter 130, the finite compact-metric source of chapter 129, and the simplex inventory of chapter 238. The packet names a constructive finite obligation surface; it does not import classical open-cover compactness, selected complementary components, quotient path equality, or a choice-dependent separation theorem.

Definition 3326.1 (Jordan curve carrier)**.** *A* $\mathsf{JordanCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, F, P, W, O, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, recording the circle parameter ledger and its local classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row into the displayed* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ *plane, with graph and modulus observations visible.*
3. *P is the finite polygonal approximation row, read through a* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision and its vertex-image ledger.*
4. *W is the winding ledger, recording the finite edge-crossing and sign-change data used to compare test points with the polygonal boundary.*
5. *O is the orientation row, recording the compatible cyclic order inherited from the* $\mathsf{S}^{1\uparrow}$ *row and the polygonal subdivision.*
6. *I is the injectivity-apartness row, stating that distinct inspected arcs carry displayed positive separation windows before they are used by the boundary read.*
7. *E is the exterior-interior ledger, recording only the finite side classification certified by the current polygonal and winding rows.*
8. *H records componentwise* hsame *transport for $S, F, P, W, O, I, E, C, Q, N$.*
9. *C records displayed* Cont *replay from circle, map, polygonal, winding, orientation, and apartness rows to the side-classification row.*

10. *Q is the* Pkg *provenance over* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, P, W, O, I, E, H, C, Q, N$. It has no coordinate for a host Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or countable-choice separation.

Lean4 ✓ `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate single carrier alignment`

Theorem 3326.2 (Jordan curve orientation ledger)**.** *For every accepted* $\mathsf{JordanCurve}^{\uparrow}$ *carrier*

$$J = (S, F, P, W, O, I, E, H, C, Q, N),$$

the orientation row O is determined by the finite $\mathsf{S}^{1\uparrow}$ *cyclic order, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision in P, and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *edge-image ledger in F. The winding ledger W can read orientation only through those rows, transported by H and replayed by C.*

Proof. Project the accepted carrier. The only circle-facing coordinate is S, and the only finite boundary skeleton is the polygonal row P together with the displayed edge images in F.

The orientation row O is therefore read by following the cyclic order of S across the subdivision cells of P. Each edge-image comparison is a finite real-plane observation supplied by F, not by an ambient topological orientation principle.

The winding ledger W consumes that ordered edge list. Componentwise transport H and replay C preserve the same ordered rows, while Q and N record provenance and local naming. Since the carrier contains no host orientation object or quotient path coordinate, the finite ledger is the entire orientation source. □

Theorem 3326.3 (Jordan curve winding separation boundary)**.** *Every side-classification read from an accepted* $\mathsf{JordanCurve}^{\uparrow}$ *packet factors through the finite route*

$$S, F, P, O, I, W \longmapsto E \longmapsto H, C, Q, N.$$

Thus the packet exposes polygonal approximation, injectivity-apartness, orientation, and winding ledgers before it names an interior or exterior row. It does not assert a blanket classical Jordan separation theorem, selected complementary components, open-cover compactness, quotient path equality, or choice-dependent planar separation.

Proof. Read the carrier of Definition 4365.1. The curve source is S, the displayed map is F, and the finite polygonal boundary is P.

Before a side row is exported, the packet requires the orientation ledger O, injectivity-apartness row I, and winding ledger W. These rows are finite BHist data: the winding read uses edge crossings and signs already attached to the polygonal boundary.

The exterior-interior row E is downstream of that route. Transport, replay, provenance, and naming through H, C, Q, N preserve the same finite packet. The excluded host theorem, selected components, open covers, quotient paths, and choice rows are not coordinates of the carrier, so they cannot be read from the local NameCert surface. □

Falsifiable prediction: A $\mathsf{JordanCurve}^{\uparrow}$ consumer that exports side separation without reading P, W, O, I refutes the carrier surface.

Independence witness: real, sone, continuousmap, compactmetric, simplicialcomplex: the sibling chapters supply real-plane coordinates, circle source data, continuous graph and modulus rows, compact-metric finite inspection, and subdivision cells separately, while $\mathsf{JordanCurve}^{\uparrow}$ binds them into the finite packet $J = (S, F, P, W, O, I, E, H, C, Q, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$.

Constructive story: A $\mathsf{JordanCurve}^{\uparrow}$ packet is generated from a circle source row, continuous real-plane map row, finite polygonal subdivision, winding ledger, orientation row, injectivity-apartness row, side-classification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$, seed)

Scope closed: Definition 4365.1, Definition 4365.2, and Definition 4365.3 seed the finite polygonal, winding, orientation, apartness, side-classification, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, the classical Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or choice-dependent separation.

Upgrade path: Obligation closure requires checked carrier admission, polygonal approximation exactness, winding-ledger stability, orientation transport, injectivity-apartness handoff, side-classification non-escape, replay, provenance, and local naming for BEDC.Derived.JordanCurveUp.

Chapter 3327

A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$

The MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite NameCert data. It sits beside the candidate-set packet of chapter 2320 and the normalization router of chapter 1108: those siblings expose candidate evidence and route selection, while this packet records the boundary where empty-context closed typed terms may or may not enter the candidate interface. The packet names a closed-projection frontier only; it does not turn candidate-mediated normalization into an unconditional strong-normalization theorem for every closed MetaCIC term.

Definition 3327.1 (MetaCIC candidate closed-projection boundary carrier)**.** *A* MetaCICCandidateClosedProjection *carrier is a finite* BHist *packet*

$$B = (T, K, R, E, NF, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row whose projection into the candidate interface is being tested.*
2. *K is the candidate-set evidence row, read through* CandidateSetNormalization$^{\uparrow}$ *when the projection is admitted.*
3. *R is the router row, read through* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *and recording which candidate, closed-candidate, or candidate-set route is exposed.*
4. *E is the bounded-normal endpoint seal row, recording the finite normal-output evidence available after candidate admission.*
5. *NF is the no-infinite-reduction row exported only downstream of K, R, E.*
6. *O is the obstruction row recording refusal when the closed term cannot be placed into the displayed candidate interface.*
7. *H records componentwise* hsame *transport for $T, K, R, E, NF, O, C, P, L$.*
8. *C records displayed* Cont *replay from the closed-term row through candidate admission, router selection, endpoint seal, no-infinite-reduction read, and obstruction retention.*

9. *P is the* Pkg *provenance over* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$, CandidateSetNormalization$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, MetaCICOpenProblemLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, R, E, NF, O, H, C, P, L$. It has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Lean4 ✓ BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.MetaCICCandidateClosedProjectionBoundar
single carrier alignment

Theorem 3327.2 (MetaCIC candidate-set closed projection frontier)**.** *For every accepted* MetaCICCandidateClosed *carrier*

$$B = (T, K, R, E, NF, O, H, C, P, L),$$

the closed-term row T reaches a strong-normalization-facing read only through candidate evidence K, router selection R, and bounded-normal endpoint seal E. If those rows do not place T inside the candidate interface, the obstruction row O is the only exported boundary datum.

Proof. Project the carrier. The closed-term row T is present as a finite syntactic and typing packet, but it is not itself a normalization theorem.

The positive route first reads K through chapter 2320. The router row R then records the candidate, closed-candidate, or candidate-set branch through chapter 1108. Only after those rows are displayed can the endpoint seal E and no-infinite-reduction row NF be consumed.

If candidate admission is absent, O records the refusal. Transport, replay, provenance, and local naming through H, C, P, L preserve that same finite boundary, so the packet exports no evidence-free closed-term strong-normalization claim. □

Theorem 3327.3 (MetaCIC candidate-set SN handoff boundary)**.** *Every strong-normalization handoff exported by an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet is a handoff for rows already inside the displayed candidate interface. It cannot be read as a blanket theorem that all empty-context typed closed MetaCIC terms are strongly normalizing.*

Proof. Start with the frontier theorem. It places the normalization-facing route behind K, R, E and keeps the obstruction row O visible when the candidate interface is not supplied.

For an admitted row, the handoff consumes candidate evidence from CandidateSetNormalization$^{\uparrow}$ and route evidence from MetaCICNormalizationCertificateRouter$^{\uparrow}$. The no-infinite-reduction row NF is therefore downstream of the admitted candidate branch rather than a second source of normalization evidence.

For a closed term outside that interface, the carrier has only T and the obstruction row O on the boundary. The structural rows H, C, P, L transport, replay, source, and name the boundary, but they do not add subject reduction, host conversion equality, quotient reduction equality, or a global normalizer. Hence the handoff is candidate-scoped. □

Theorem 3327.4 (MetaCIC candidate-set public boundary)**.** *Any public no-infinite-reduction or strong-normalization read from an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet must cite candidate-set membership or soundness, router selection, and bounded-normal endpoint rows. A bare empty-context typed closed-term row T is insufficient: the public boundary may expose NF only downstream of the candidate-set facts named by* closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, *and* canonicalCandidateSet_sound *together with the router row R and endpoint seal E.*

Proof. Start with Definition 4366.2. It places every normalization-facing read behind the candidate evidence row K, router row R, and endpoint seal E, and leaves the obstruction row O as the only datum when candidate admission is absent.

Next apply Definition 4366.3. That theorem identifies the handoff as a candidate-interface handoff rather than a theorem about all empty-context closed typed rows. Thus the public NF row cannot be projected from T alone.

Finally read the MetaCIC normalization audit map through the existing candidate-set facts closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, and canonicalCandidateSet_sound. Those facts supply the candidate membership and soundness side of the route, while R and E supply the public router and endpoint rows. The resulting boundary is therefore candidate-mediated and evidence-bearing, not an unconditional closed-term normalizer. □

Falsifiable prediction: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ consumer that exports strong normalization from T without reading K, R, E refutes the carrier surface.

Independence witness: candidate-set-normalization, metacicnormalizationcertificaterouter, metacicopenproblemledger: the candidate-set packet carries candidate evidence, the router selects the admissible normalization route, and the open-problem ledger records the frontier question; this packet binds those rows into a closed-projection boundary without adding an evidence-free normalization theorem.

Closure status for NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$.

Constructive story: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate-set evidence, normalization-router selection, bounded-normal endpoint seal, no-infinite-reduction row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4366.1, Definition 4366.2, Definition 4366.3, and Definition 4366.4 seed the finite closed-projection, candidate-evidence, router-selection, endpoint-seal, obstruction, public-boundary, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional strong normalization for all closed typed MetaCIC terms, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Upgrade path: Obligation closure requires checked closed-term projection, candidate-evidence admission, router selection, bounded-normal endpoint sealing, no-infinite-reduction handoff, obstruction exactness, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.

Chapter 3328

A Concrete Naming Certificate for $\mathsf{PointedMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based metric spaces, the $\mathsf{PointedMetricSpace}^{\uparrow}$ packet records a metric carrier together with a named basepoint route before completion, loop-space, or embedding consumers are allowed to inspect point-level data. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, reads names through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, passes regular rational distance observations through chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and offers a based source for the distance-preserving consumer route of chapter 439. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected representatives, or ambient completeness.

Definition 3328.1 (Pointed metric space carrier)**.** *A* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, b, D, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *b is the basepoint name row, stored as finite BEDC data attached to M rather than as a host-level point.*
3. *D is the basepoint distance ledger, recording the finite requests that compare b with inspected metric names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which basepoint and metric names are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the basepoint-distance observations.*
7. *H records componentwise* hsame *transport for M, b, D, S, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a based metric request through the stream windows, regular rational readback, and real-distance seal.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{PointedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint name, distance ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate single carrier alignment`

Theorem 3328.2 (Pointed metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier* $P = (M, b, D, S, R, E, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint name, basepoint distance ledger, stream windows, regular rational readback, real-distance seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring every comparison of basepoint data to be read over the transported metric row* M*;*
3. *basepoint-distance exposure, requiring every basepoint comparison to pass through* D, S, R, E *before any completion-facing or embedding-facing consumer reads it;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 4367.1. The projection exposes the metric row, basepoint name, distance ledger, stream windows, regular readback, real seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, b, D, S, R, E together, so a consumer cannot compare a basepoint name over one metric carrier with distance observations over another.

The distance ledger fixes the read order. A basepoint comparison is first a finite request in D, then a scheduled stream observation in S, then a regular rational readback in R, and only then a $\mathsf{Real}^{\uparrow}$ distance seal in E.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected representatives, or ambient completeness, the listed rows exhaust the local naming surface. □

Theorem 3328.3 (Pointed metric space basepoint nonescape)**.** *Let* $P = (M, b, D, S, R, E, H, C, P_0, N)$ *be an accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier. Every completion-facing, loop-space-facing, or isometric-embedding-facing consumer of the basepoint row* b *must read it through the displayed route*

$$M, b, D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P_0, N.$$

Thus the basepoint is exported only as finite named metric data over M*, not as a host point, quotient representative, selected completion point, or ambient metric-equality witness.*

Proof. Begin with carrier admission. The only row naming the basepoint is b, and it is stored beside the metric carrier M and distance ledger D.

Read the inspection route next. The stream windows S and regular rational readback R display every observed distance from the basepoint before the $\mathsf{Real}^{\uparrow}$ seal E is available to downstream consumers.

Transport and replay preserve that same route. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add host points, quotient representatives, selected completion points, or ambient metric equality.

Therefore every based consumer receives exactly the finite route displayed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedMetricSpace}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a named basepoint row, a basepoint distance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4367.1, Definition 4367.2, and Definition 4367.3 seed the based metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host point equality, quotient metric equality, selected completion points, loop-space closure, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, basepoint-distance exposure, classifier transport, basepoint nonescape, and downstream completion and isometric-embedding consumer alignment for BEDC.Derived.PointedMetricSpaceUp.

Chapter 3329

A Concrete Naming Certificate for $\mathsf{IntervalConnectedness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{IntervalConnectedness}^{\uparrow}$ packet records the finite constructive route by which a located real interval is inspected as one connected interval surface before intermediate-value, zero-envelope, or completion-facing consumers use it. It sits after the endpoint seal of chapter 38, the located interval carrier of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sandwich consumer surface of chapter 2163. The packet names interval connectedness as displayed finite BEDC data; it does not import host topology, open-cover compactness, trichotomy, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3329.1 (Interval connectedness carrier)**.** *An* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint-pair row, storing the lower and upper endpoint names through their displayed real seals.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *witness row, carrying the located interval budget, locator reads, and endpoint-radius checks attached to E.*
3. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision-chain ledger, recording the dyadic cells, adjacent-cell overlaps, and tolerance requests inspected inside L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, cell, and witness observations are scheduled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled endpoint and subdivision observations.*
6. *T records componentwise* hsame *transport for endpoint pairs, located interval witnesses, dyadic subdivision chains, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint pairs through located interval exposure, dyadic subdivision, stream-window inspection, regular readback, and the interval-connectedness handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalConnectedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SqueezeTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, D, S, R, T, C, P, N$. It has no coordinate for host topological connectedness, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalConnectednessUp.IntervalConnectednessTasteGate single carrier alignment`

Theorem 3329.2 (Interval connectedness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier*

$$I = (E, L, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint pair, located interval witness, dyadic subdivision chain, stream windows, regular readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring endpoint, locatedness, subdivision, window, readback, replay, provenance, and naming data to move componentwise through T;*
3. *dyadic chain exactness, requiring every inspected subinterval and adjacent-cell overlap to be recorded in D before a connectedness consumer reads the packet;*
4. *endpoint handoff nonescape, requiring every consumer of interval endpoints to pass through E, L, D, S, R before using any* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger exhaustion, requiring the packet to expose no host topological connectedness theorem, open-cover compactness, trichotomy, quotient real equality, selected intermediate point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The sibling routes through $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{SqueezeTheorem}^{\uparrow}$ *are the only named consumers of this local certificate surface.*

Proof. Project the carrier of Definition 4369.1. The endpoint pair is visible only in E, and the located interval evidence is visible only in L.

The subdivision data are then forced through D. Each finite cell, overlap, and tolerance request is read as dyadic data before any stream-window or regular-readback consumer can inspect it.

Transport and replay preserve the same route. The rows T and C move endpoint, locatedness, subdivision, window, and readback data componentwise, while P and N source and name exactly that displayed packet.

Since the carrier has no coordinate for the excluded host principles, the local certificate cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{IntervalConnectedness}^{\uparrow}$ consumer reads endpoint pairs, located interval witnesses, dyadic subdivision chains, StreamName windows, RegSeqRat readbacks, hsame transport, Cont replay, Pkg provenance, and local naming before any interval-connectedness handoff is visible.

Independence witness: real, locatedrealinterval, dyadicratcore, streamname, regseqrat, squeezetheorem: $\mathsf{IntervalConnectedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, located interval witnesses, finite dyadic subdivision chains, scheduled stream windows, regular rational readbacks, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$.

Constructive story: An IntervalConnectedness$^{\uparrow}$ packet is generated from Real endpoint pairs, a LocatedRealInterval witness, a finite DyadicRatCore subdivision chain, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, seed)

Scope closed: Definition 4369.1 and Definition 4369.2 seed the finite located-interval connectedness route over Real$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and SqueezeTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topology, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic chain exactness, endpoint handoff nonescape, ledger exhaustion, and downstream intermediate-value consumer alignment for BEDC.Derived.IntervalConnectedness

Chapter 3330

A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$

Following Tait–Martin-Lof and Barendregt on residuals, parallel reduction, and confluence discipline, the MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ packet records the finite boundary where a beta-parallel branch and a substitution replay must meet before the MetaCIC confluence route may use the conditional diamond bridge. It sits over the audit map of chapter 9378 and names the residual-substitution input as packet data rather than as a global Church–Rosser theorem. The packet does not claim full closed Church–Rosser, arbitrary closed-source confluence, or an unrestricted substitution theorem.

The residual-substitution packet is consumed by chapter 765 as a guarded input to candidate-mediated SN/confluence scheduling. The dependency is the residual join row together with the closedness guard data: it tells the scheduler which finite substitution replay may enter a candidate branch and which source windows are closed enough for that handoff, not that the full Church–Rosser theorem has already been completed.

Definition 3330.1 (MetaCIC residual-substitution compatibility carrier)**.** *A* MetaCICResidualSubstitutionCompatil *carrier is a finite* BHist *packet*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

with the following displayed rows:

1. *s is the source term row whose beta-parallel residual branch is inspected.*
2. *t is the substituted term row, storing the finite substitution request and the target occurrence window.*
3. *b is the parallel branch row, displaying the left and right beta-parallel steps before any join is named.*
4. *j is the residual join row, recording the candidate common reduct obtained after substitution replay and residual transport.*
5. *g is the closedness guard row, saying exactly which source and substitution windows are closed enough for the guarded handoff.*
6. *a is the atom-shape separation row, keeping atom, application, lambda, Pi, and beta-redex cases visibly disjoint.*
7. *h records componentwise* hsame *transport for source, substitution, branch, residual join, closedness guard, atom-shape separation, replay, provenance, and local naming rows.*

8. *c records displayed* Cont *replay from the source term through substitution, parallel branching, residual join construction, closedness guarding, and atom-shape separation.*
9. *p is the* Pkg *provenance over* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$*, MetaCIC beta-parallel reduction, residual joins, closedness guards, atom-shape separation,* BHist, hsame, Cont, *and* NameCert.
10. *n is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, t, b, j, g, a, h, c, p, n$. It has no coordinate for a completed parallel diamond, full closed Church–Rosser, arbitrary closed-source confluence, or an unguarded substitution principle.

Lean4 ✓ BEDC.Derived.MetaCICResidualSubstitutionCompatibilityUp.MetaCICResidualSubstitutionCompatibilit
single carrier alignment

Theorem 3330.2 (MetaCIC residual-substitution compatibility NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and local naming rows;*
2. *branch-shape disjointness, requiring atom, application, lambda, Pi, and beta-redex branches to be separated by a before a residual join row is accepted;*
3. *substitution residual replay, requiring c to display how the substituted term row and beta-parallel branch row reach the same candidate join j;*
4. *closed-source guard, requiring g to expose the exact closedness assumptions consumed by the guarded compatibility packet;*
5. *non-claim of full closed Church–Rosser, requiring the packet to remain an input boundary for the conditional confluence bridge rather than a proof of arbitrary closed-source confluence.*

No obligation is stored outside $s, t, b, j, g, a, h, c, p, n$.

Proof. Project the carrier of Definition 4370.1. The source term, substituted term, branch, residual join, closedness guard, and atom-shape separation rows are the only computational coordinates.

The branch-shape row a fixes the case split before the join row is read. Thus a beta-redex residual branch cannot be silently identified with an atom, application, lambda, or Pi shape.

Replay then supplies the compatibility boundary. The row c records the finite route from substitution and beta-parallel branching to the candidate join j, while g states the closedness guard consumed by that route.

Finally h, p, n transport, source, and name exactly the displayed packet. Since full closed Church–Rosser and arbitrary closed-source confluence are not carrier coordinates, they are not exported by this NameCert surface. □

Theorem 3330.3 (MetaCIC residual-substitution candidate projection route)**.** *Every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

exports a candidate-closed projection route to chapter 4366 only after the source term row s, substituted term row t, beta-parallel branch row b, residual join row j, closedness guard row g, and atom-shape separation row a have all been displayed. The downstream candidate boundary receives a guarded compatibility packet, not full Church–Rosser and not arbitrary closed-source confluence.

Proof. Project the carrier of Definition 4370.1. The rows s, t, b, j, g, a are the only computational rows available before the structural transport, replay, provenance, and local naming rows are read.

Use Definition 4370.2. Branch-shape disjointness exposes a, substitution residual replay exposes the route from t and b to j, and the closed-source guard exposes exactly the assumptions in g.

The candidate boundary of chapter 4366 is therefore reached through a guarded compatibility packet. The route supplies source, substitution, beta-parallel, residual-join, closedness, and atom-shape data, while h, c, p, n only transport, replay, source, and name those rows.

Since completed Church–Rosser and arbitrary closed-source confluence are absent from the carrier, the candidate-closed projection route cannot export them. It hands off only the displayed residual-substitution compatibility boundary. □

Theorem 3330.4 (MetaCIC residual-substitution obligation closure surface)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the obligation surface is exactly the displayed source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and naming rows. The candidate-closed projection boundary receives only this guarded compatibility packet.

Proof. Apply Definition 4370.2. It identifies carrier admission, branch-shape disjointness, substitution residual replay, closed-source guarding, transport, replay, provenance, and naming as rows of $s, t, b, j, g, a, h, c, p, n$.

Apply Definition 4370.3. The candidate boundary receives the source, substitution, beta-parallel branch, residual-join, closedness, and atom-shape data only through that guarded projection route.

Therefore the obligation surface packages the candidate handoff without exporting full closed Church–Rosser, arbitrary closed-source confluence, or unguarded substitution compatibility. Those claims are not coordinates of the carrier and are not supplied by the candidate projection. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ read displays source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, hsame transport, Cont replay, Pkg provenance, and local naming rows before the conditional MetaCIC confluence bridge can consume it.

Independence witness: metacic-confluence-audit: $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ is not a completed confluence theorem because it carries only the finite residual-substitution compatibility boundary consumed by the conditional parallel-diamond route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ packet is generated from source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4370.1, Definition 4370.2, and Definition 4370.3 seed the finite residual-substitution compatibility boundary consumed by the conditional MetaCIC confluence bridge and the candidate-closed projection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, arbitrary closed-source confluence, unguarded substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-shape disjointness, substitution residual replay, closed-source guard, and conditional confluence handoff alignment for BEDC.MetaCIC.BetaParallelSubstitutionCompatibility.

Chapter 3331

A Concrete Naming Certificate for $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet records the separated-completion reflection route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the regular rational readback surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness boundary of chapter 3390. The packet lets a Real-facing consumer treat two representative regular Cauchy names as the same separated reflection point only after the displayed finite windows, dyadic budgets, and uniqueness ledger have been exposed.

Definition 3331.1 (Regular-Cauchy Hausdorff-reflection carrier)**.** *A* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row presented to the same separated-reflection request.*
3. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row witnessing that the displayed tail comparisons force separated agreement.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, Y, S, T, D, U have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, stream windows, dyadic ledger, uniqueness row, Real seal, and local certificate rows.*
8. *C records the displayed* Cont *replay from a reflection request to the paired windows, dyadic tolerance ledger, uniqueness row, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier admission surface`

Theorem 3331.2 (Regular-Cauchy Hausdorff-reflection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier* $R = (X, Y, S, T, D, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *separated-agreement row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4372.1. The only displayed coordinates are $X, Y, S, T, D, U, E, H, C, P, N$.

The two regular rational names are compared through the paired stream windows S, T and the dyadic tolerance ledger D. The separated-agreement row U is therefore a displayed uniqueness boundary, not a quotient of streams or a host equality proof.

The terminal Real seal E is downstream of that boundary. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and no excluded quotient, selector, host-equality, choice, trichotomy, or ambient-completeness coordinate can be projected from the carrier. □

Theorem 3331.3 (Regular-Cauchy Hausdorff-reflection representative independence)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet is independent of the chosen representative regular Cauchy name in the following BEDC sense: if the two presented* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are compared through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the terminal seal can distinguish them only through the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row. The packet exports no quotient stream equality, selected representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionRepresentativel`

Proof. Begin with an accepted carrier $R = (X, Y, S, T, D, U, E, H, C, P, N)$. The rows X and Y are not presented as completed real objects; they are presented as regular rational names observed through finite stream windows.

The dyadic ledger D binds the comparison budget for those windows, and the uniqueness row U is the only row that can turn the displayed tail comparison into separated agreement. Thus any Real-facing read of E must pass through the same finite comparison route.

The rows H, C, P, N transport, replay, source, and name this route without adding analytic content. Since the carrier contains no further equality coordinate, changing representatives cannot be detected except through the displayed uniqueness row. □

Theorem 3331.4 (Regular-Cauchy Hausdorff-reflection Real seal non-escape)**.** *No terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the separated reflection supplies representative independence for the Real seal without quotient streams, host real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier real seal nonescape`

Proof. Read the carrier coordinates in order. The terminal row E appears only after the two regular rational names, paired stream windows, dyadic tolerance ledger, and uniqueness row have been displayed.

Apply Definition 4372.2. The local NameCert surface exposes exactly those rows, with H, C, P, N preserving the same route for downstream consumers.

The advertised route is therefore the only public path to the Real seal. Since no excluded equality or choice coordinate occurs in the carrier, the seal cannot escape through quotient streams, host equality, representatives selected outside the packet, countable choice, trichotomy, or ambient completeness. □

Theorem 3331.5 (Regular-Cauchy Hausdorff-reflection finite-window uniqueness route)**.** *Suppose two accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packets display the same paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then every separated-reflection consumer factors through the same finite-window route on the underlying* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization does not use quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier finite window uniqueness route`

Proof. Apply Definition 4372.3. Once the paired StreamName windows, dyadic tolerance ledger, and uniqueness row agree, the terminal Real seal cannot distinguish the two regular Cauchy representatives by any route other than the displayed uniqueness row.

Apply Definition 4372.4. The Real seal is downstream of the finite windows, dyadic ledger, and uniqueness row, so a separated-reflection consumer has no bypass around those displayed rows.

The remaining rows are structural. The hsame rows transport only the displayed components, the Cont rows replay the same finite-window request, and the Pkg and NameCert rows name that route inside BHist. Therefore the consumer factors through the same finite-window route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore tolerance ledger, and a RegularCauchyUniqueness row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyuniqueness: $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational representatives to one separated uniqueness row and one downstream Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyUniqueness separated-agreement row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4372.1, Definition 4372.2, Definition 4372.3, Definition 4372.4, and Definition 4372.5 seed the separated reflection route for regular Cauchy representatives and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, representative-independence handoff, and Real seal non-escape theorem.

Chapter 3332

A Concrete Naming Certificate for $\mathsf{TightRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{TightRealInterval}^{\uparrow}$ packet names a finite lower-upper real interval by displaying both endpoint names and every rational cell through which those endpoints are inspected. It sits downstream of the located endpoint surface of chapter 394 and the dyadic refinement surface of chapter 306: consumers may read a tight interval only after the lower endpoint, upper endpoint, rational cell ledger, dyadic refinement ledger, stream windows, regular rational readbacks, and Real seal rows have all been displayed.

Definition 3332.1 (TightRealInterval carrier)**.** *A* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, U, Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row and U is the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger recording the finite rational windows that enclose the endpoint reads and witness lower-before-upper compatibility.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to the rational cells in Q.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the endpoint observations and interval cells inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and cell refinements.*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached from L, U, S, R, Q, D.*
7. *H records componentwise* hsame *transport for endpoints, rational cells, dyadic refinements, stream windows, readbacks, Real seals, replay, provenance, and the local name.*
8. *C records displayed* Cont *replay through $L, U, Q, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host interval equality, chosen endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.TightRealIntervalUp.TightRealIntervalUp`
Lean4 ✓ `BEDC.Derived.TightRealIntervalUp.TasteGate.TightRealIntervalTasteGate single carrier alignment`

Theorem 3332.2 (TightRealInterval NameCert obligations)**.** *For an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier* $T = (L, U, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, rational and dyadic interval ledgers, finite stream windows, regular rational readbacks, paired Real seal rows, componentwise transport, continuation replay, package provenance, and local naming; endpoint-order visibility through* Q*; dyadic refinement discipline through* D*; stream-readback exactness through* S, R*; and ledger non-escape from host intervals, selected endpoints, quotient real equality, choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 4360.1. The lower and upper endpoint rows, the rational cell ledger, the dyadic refinement ledger, finite windows, regular readbacks, Real seals, transport rows, replay rows, provenance row, and local naming row are all visible before any interval consumer can read the packet.

Endpoint-order visibility is supplied by Q. Since each dyadic refinement in D is attached to a rational cell, the refinement discipline cannot detach from the endpoint ledger.

Stream-readback exactness is then read through S and R. The rows H, C, P, N transport, replay, source, and name the same finite route, while the excluded interval and completeness data have no carrier coordinate. □

Theorem 3332.3 (TightRealInterval Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

The read exposes lower and upper endpoint windows together with the rational and dyadic cell ledgers that bind them. It does not expose a host real interval, a selected endpoint value, quotient real equality, countable choice, trichotomy, or ambient real completeness.

Proof. Begin with the accepted carrier. The only seal-facing row is E, and it is displayed after L, U, Q, D, S, R in the finite packet.

Apply Definition 4360.2. Endpoint-order visibility forces the Real read to keep the rational cell ledger Q, dyadic refinement discipline forces any dyadic window through D, and stream-readback exactness forces finite observations through S, R.

Finally route through H, C, P, N. These rows preserve and name the same packet, so no host interval, endpoint selector, quotient equality, choice schedule, trichotomy row, or ambient completeness principle can enter the Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TightRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, paired $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4360.1, Definition 4360.2, and Definition 4360.3 seed the finite lower-upper Real interval packet over located endpoints, rational cells, dyadic refinements, stream windows, regular readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Scoped closure requires a checked carrier-obligation package and a public consumer route for the same finite lower-upper interval packet.

Chapter 3333

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$

Following Bishop-style compact metric uniform continuity, the $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet records the finite handoff that transfers compact finite-net and pointwise continuity data into a uniform modulus row. It consumes the finite compact source of chapter 60, the compact metric net surface of chapter 129, the continuity interface of chapter 391, the finite stream and regular readback rows of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite modulus-transfer NameCert surface, not an open-cover compactness proof or a host modulus selector.

Definition 3333.1 (Compact uniform modulus transfer carrier)**.** *A* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, U, S, R, D, F, H, C, P, N)$$

with the following visible rows:

1. K *is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row.*
2. G *is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric coverage row for the same finite centers and source distance witnesses.*
3. U *is the* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise-to-uniform continuity row consumed by the transfer.*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected finite centers, tolerance requests, and pointwise-radius reads.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for finite radius comparison and shrinkage.*
7. F *is the modulus-transfer ledger that records the finite lower-bound handoff into the downstream uniform-modulus consumer row.*
8. H *records componentwise* hsame *transport for compact, compact-metric, uniform-continuity, stream, readback, dyadic, transfer, replay, provenance, and local-name rows.*
9. C *records displayed* Cont *replay through* $K, G, U, S, R, D, F, H, P, N$.
10. P *is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate`
`single carrier alignment`

Theorem 3333.2 (Compact uniform modulus transfer obligations)**.** *For an accepted* CompactUniformModulusTrans *carrier $M = (K, G, U, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net row, compact-metric coverage row, uniform-continuity row, stream windows, regular rational readbacks, dyadic tolerance ledger, modulus-transfer ledger, componentwise transport, continuation replay, package provenance, and local naming; source lock between K, G, U; finite-window exactness through S, R; dyadic tolerance exactness through D; and non-escape from open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient equality, choice, and ambient completeness.*

Proof. Project the carrier of Definition 4361.1. The compact row, compact-metric row, continuity row, stream windows, readbacks, dyadic tolerance ledger, transfer ledger, transport rows, replay rows, provenance row, and local name are displayed as one finite packet.

The source lock is the shared read of K, G, U. A transfer consumer cannot use the compact finite-net row without the compact-metric distance surface and the continuity row that supplies pointwise radius data.

Finite-window and dyadic exactness are read through S, R, D. The ledger F can therefore transfer only the displayed finite lower-bound data, and H, C, P, N preserve that route without adding any excluded compactness or choice principle. □

Theorem 3333.3 (Compact uniform modulus transfer handoff)**.** *Every accepted* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *packet hands a downstream uniform-modulus consumer only the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont}$$

The handoff records compact finite-net coverage, pointwise continuity radius reads, finite stream windows, regular rational readbacks, dyadic tolerance comparison, and the transfer ledger. It exports no open-cover compactness theorem, selected net center, selected Lebesgue number, host modulus function, quotient equality, countable choice, or ambient completeness.

Proof. Start from the transfer obligations of Definition 4361.2. They identify the compact, compact-metric, continuity, stream, readback, dyadic, and transfer rows as the only NameCert-readable data.

Route the downstream consumer through F. The ledger can be consumed only after K, G, U, S, R, D have supplied the finite-net, pointwise-radius, window, readback, and tolerance rows.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since the excluded compactness, selection, equality, choice, and completeness data have no coordinate in the carrier, the downstream handoff is exactly the displayed finite modulus-transfer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet is generated from compact finite-net rows, compact-metric coverage rows, uniform-continuity rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a modulus-transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, seed)

Scope closed: Definition 4361.1, Definition 4361.2, and Definition 4361.3 seed the finite compact-net modulus-transfer packet for uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Upgrade path: Scoped closure requires a checked transfer-obligation package and a public uniform-modulus consumer route for the same finite packet.

Chapter 3334

A Concrete Naming Certificate for $\mathsf{LocatedSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric and real spaces, the $\mathsf{LocatedSubset}^{\uparrow}$ packet records a displayed subset together with finite distance-witness rows. It sits after the real-valued metric route of chapter 1638, the Hausdorff comparison boundary of chapter 2086, the real-separability window surface of chapter 1809, the dyadic tolerance core of chapter 291, and the regular rational readback surface of chapter 301. The packet gives Real and Completion consumers a finite located-subset interface without host set extensionality, selected nearest points, quotient subset equality, or ambient completeness.

The located-subset distance route consumes chapter 1809 as its finite search-window source and consumes chapter 2086 only after the real-distance seal has been displayed.

Definition 3334.1 (LocatedSubset carrier)**.** *A* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, W, D, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *source row, carrying the ambient metric distance through finite real-name observations.*
2. *S is the subset-admission row, naming the displayed members read by the carrier and exporting no host powerset equality.*
3. *W is the* $\mathsf{RealSeparability}^{\uparrow}$ *search-window row, recording the countable dense or finite-window probes available to a distance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to each positive distance query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison data produced from W and D.*
6. *E is the real distance seal row, read only after the metric source, subset admission, search windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *K is the optional* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison boundary for compact or hyperspace consumers; it records only the finite directed comparisons visible through the preceding rows.*
8. *H records componentwise* hsame *transport for the metric source, subset admission, search windows, dyadic ledger, regular readback, real seal, Hausdorff boundary, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an ambient point and tolerance request through subset admission, separability windows, dyadic comparison, regular rational readback, real sealing, and any Hausdorff-facing comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSubset}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, W, D, R, E, K, H, C, P, N$. It has no coordinate for a host subset object, extensional equality of subsets, a selected nearest point, quotient real equality, countable choice, or an ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate single carrier alignment`

Theorem 3334.2 (LocatedSubset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, S, W, D, R, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, optional Hausdorff comparison routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4374.1. The displayed packet lists exactly the rows used by an ambient point, subset-membership, tolerance, rational-readback, real-seal, or Hausdorff-facing request.

The metric and subset rows M, S are read first. A distance query can inspect the ambient space only through $\mathsf{RealMetric}^{\uparrow}$, and it can inspect membership only through the displayed admission row S.

The search, tolerance, and readback rows W, D, R then supply finite witnesses for the locatedness request. The real seal E is reached only after those rows have been displayed, while K is available only as a downstream comparison boundary.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host subset, selected nearest point, quotient equality, countable-choice schedule, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3334.3 (LocatedSubset dyadic distance witness)**.** *Every distance read exported by an accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto E.$$

The read may be handed to a $\mathsf{HausdorffMetric}^{\uparrow}$ *consumer only after the same route has produced the displayed real-distance seal, transported by H and replayed by C.*

Proof. Begin with a point-to-subset distance request. The ambient metric row M supplies the only metric observations, and the subset-admission row S supplies the only visible membership data.

The finite search window W names the inspected probes. The dyadic ledger D fixes the positive tolerance requests, and the regular rational row R records the resulting readback windows.

The real-distance seal E is therefore downstream of M, S, W, D, R. Any Hausdorff-facing row K can compare only the finite distance data already sealed by this route.

Transport and replay do not add another distance object. The componentwise hsame row H, the displayed Cont route C, the Pkg provenance P, and the local NameCert row N preserve the

same witness surface, so no nearest-point selector or host infimum can bypass the dyadic distance route. □

Falsifiable prediction: Every accepted LocatedSubsetUp distance read exposes RealMetricUp, subset admission, RealSeparabilityUp windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, and a real-distance seal before any HausdorffMetricUp or completion-facing consumer reads the packet.

Independence witness: realmetric, hausdorffmetric, realseparability, dyadicratcore, regseqrat: none of these siblings alone carries the full located-subset packet with subset admission, finite search windows, dyadic tolerance rows, regular rational readback, real-distance sealing, optional Hausdorff comparison, transport, replay, package provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}$.

Constructive story: A $\mathsf{LocatedSubset}^{\uparrow}$ packet is generated from a RealMetricUp source, displayed subset admission, RealSeparabilityUp search windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a real-distance seal, optional HausdorffMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4374.1, Definition 4374.2, and Definition 4374.3 seed the finite located-subset interface for real-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host subset extensionality, selected nearest points, quotient subset equality, countable choice, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, Hausdorff comparison routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSubsetUp.

Chapter 3335

A Concrete Naming Certificate for $\mathsf{WeakLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on finite averages and probability estimates, the $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet names the displayed finite-average route behind the weak law of large numbers. It sits after the normalized probability surface of chapter 184, the measurable random-variable rows of chapter 185, the finite tail-bound carrier of chapter 6592, and the terminal real-name seal of chapter 38. The packet is a finite certificate for Chebyshev-controlled averages; it is not an almost-sure convergence theorem, a central limit theorem, a countable-choice subsequence selector, a host probability-limit principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 3335.1 (Weak law finite-average carrier)**.** *A* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, R, A, V, T, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the total-event normalization and event classifier. The row R is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over that same probability source. The row A records the displayed finite average of the entries in R, while V records the variance or uniform second-moment bound used by the estimate. The row T reads the Chebyshev tail row through* $\mathsf{ChebyshevInequality}^{\uparrow}$ *for the event where the finite average differs from its displayed center by more than the requested tolerance. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the final probability bound. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite averaging route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these ten rows.*

Lean4 ✓ `BEDC.Derived.WeakLawLargeNumbersUp.WeakLawLargeNumbersTasteGate single carrier alignment`

Theorem 3335.2 (Weak law NameCert obligations)**.** *For every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier $(P, R, A, V, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 4373.1. Carrier admission displays the probability source, the random-variable family, the finite-average row, the variance-bound row, the Chebyshev tail row, the epsilon seal, the transport and replay rows, the provenance row, and the local naming row. The packet imports no host law of large numbers, selected subsequence, external probability-limit operation, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The probability source P supplies the measurable-event classifier and total-mass normalization, and the random-variable row R exposes only the finitely displayed variables over that source.

The average row A and variance row V are then internal finite data. They record the algebraic averaging expression and the bound used by the tail estimate before any limiting statement is visible.

Finally read T, E, H, C, Q, N. The Chebyshev tail row and epsilon seal name the probability estimate at the requested tolerance, while transport, replay, provenance, and local naming keep the same finite route under classifier replacement. Since no carrier coordinate stores an infinite limit or subsequence selector, the listed rows exhaust the local certificate surface. □

Theorem 3335.3 (Weak law Chebyshev handoff)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *Chebyshev read factors through the route*

$$P \longmapsto R \longmapsto A \longmapsto V \longmapsto T \longmapsto E \longmapsto H, C, Q, N$$

before any probability-bound consumer receives the finite-average estimate. Thus the Chebyshev step is handed to later probability and $\mathsf{Real}^{\uparrow}$ *formalization as a displayed finite source, finite random-variable row, average row, variance-bound row, tail-event row, epsilon seal, transport, replay, provenance, and local naming data.*

Proof. Start with a probability-bound request. The only probability source exposed by the packet is P, and the only random-variable data exposed over it are the finite rows in R.

The request must next read the average and variance-bound rows. The average row A fixes the finite expression being estimated, and V fixes the bound that the Chebyshev tail row may consume.

Only after these rows are displayed can T and E be read as a bound on the exceptional event. The structural rows H, C, Q, N then preserve that same route for downstream consumers, so the handoff is the finite Chebyshev estimate itself rather than a host-side convergence theorem. □

Theorem 3335.4 (Weak law finite-average nonescape)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *probability-limit-facing read remains inside the finite-average route of Definition 4373.1. In particular, a consumer can read a tail estimate only after displaying the source probability row, finite random-variable rows, average row, variance-bound row, Chebyshev tail row,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by almost-sure convergence, central limit normalization, countable-choice subsequence selection, a host probability-limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a consumer request the finite-average estimate from an accepted carrier. Projection of Definition 4373.1 shows that the request first meets P, R, A, V, T, E and has no independent coordinate for a limit object.

The displayed Chebyshev row T is downstream from the variance-bound row V and upstream from the epsilon seal E. Hence any probability bound read is already tied to the visible finite average and its tolerance.

The remaining rows H, C, Q, N do not add new analytic content. They transport equal displayed histories, replay the finite route, record the package provenance, and attach the local name. Therefore every accepted probability-limit-facing read stays inside the finite-average certificate surface and cannot escape to the excluded convergence or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet is generated from a probability source row, finite random-variable rows, a finite-average row, a variance or second-moment bound row, a Chebyshev tail row, a RealUp epsilon seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4373.1, Definition 4373.2, Definition 4373.3, and Definition 4373.4 seed the finite-average probability route through ProbSpace, RandomVar, RealUp, Chebyshev tail, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the central limit theorem, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-average routing, variance-bound stability, Chebyshev tail handoff, epsilon-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.WeakLawLargeNumbe

Chapter 3336

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 3336.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 3336.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 3336.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$, seed)

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 3337

A Concrete Naming Certificate for $\mathsf{UrysohnMetrization}^{\uparrow}$

Following Urysohn's metrization theorem, the $\mathsf{UrysohnMetrization}^{\uparrow}$ packet records the finite BEDC route by which a displayed normal, second-countable, Hausdorff topological source exposes a metric handoff. It sits between the normal-space separation surface of chapter 1866, the countable-base schedule of chapter 1901, the Hausdorff separation row of chapter 1925, the open-set carrier of chapter 88, and the metric consumer of chapter 57. The packet names only the displayed finite normal-space and countable-base route; it does not assert arbitrary classical metrization, selected embeddings into host sequence spaces, excluded middle, or ambient point-set equality.

Definition 3337.1 (Urysohn metrization carrier)**.** *An* $\mathsf{UrysohnMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, N, S, Hd, B, F, D, M, G, R, Q, C, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, recording the open-set classifier and finite neighbourhood histories consumed by the construction;*
2. *N is the* $\mathsf{NormalSpace}^{\uparrow}$ *separation row for closed pairs presented by the topology source;*
3. *S is the* $\mathsf{SecondCountable}^{\uparrow}$ *countable-base schedule row, exposing only finite base windows and refinement requests;*
4. *Hd is the* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row, used as the target separatedness boundary for metric consumers;*
5. *B is the finite base-window ledger read from S before any separating function request is made;*
6. *F is the closed-pair separation ledger routed through N for the displayed base-window pairs;*
7. *D is the dyadic distance-readback ledger assembled from the finite family of displayed separation rows;*
8. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row whose distance witness is read only after T, N, S, Hd, B, F, D have been displayed;*
9. *G records compatibility between the generated metric-open rows and the original topology row T on the inspected finite windows;*

10. *R records the reverse finite-window readback from topology neighbourhood requests to metric-ball requests;*
11. *Q is the refusal row excluding classical metrization data not present in $T, N, S, Hd, B, F, D, M, G, R$;*
12. *C is the displayed* Cont *replay from a topology request through the normal-space and second-countable rows to the metric handoff;*
13. *P is the* Pkg *provenance over* $\mathsf{UrysohnMetrization}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{NormalSpace}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
14. *L is the local* $\mathsf{NameCert}_{\mathsf{UrysohnMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary host spaces, choice-selected separating functions, a classical embedding into a product cube, quotient equality of points, or metrization outside the displayed normal second-countable Hausdorff route.

Lean4 ✓ `BEDC.Derived.UrysohnMetrizationUp.UrysohnMetrizationTasteGate single carrier alignment`

Theorem 3337.2 (Urysohn metrization NameCert obligations)**.** *For every accepted* $\mathsf{UrysohnMetrization}^{\uparrow}$ *carrier $U = (T, N, S, Hd, B, F, D, M, G, R, Q, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{UrysohnMetrization}^{\uparrow}}$ *surface consists exactly of topology-source admission through T, normal-space closed-pair separation through N, second-countable finite-window scheduling through S, B, Hausdorff separatedness through Hd, distance-ledger assembly through F, D, metric handoff through M, topology-to-metric and metric-to-topology compatibility through G, R, refusal of non-displayed classical data through Q, replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 3337.1. Each named obligation is one displayed row of the same finite packet, so a consumer can inspect only the topology, normal-space, second-countable, Hausdorff, distance-ledger, metric-handoff, compatibility, refusal, replay, provenance, and naming rows.

The normal-space route is read through N and F, while the countable-base route is read through S and B. These rows determine which finite separation requests can contribute to the dyadic distance ledger D.

The metric handoff is therefore the row M together with the compatibility rows G, R. Since Q excludes host embeddings, selected separating functions, quotient equality, and any metrization principle not routed through the displayed rows, the listed rows exhaust the local NameCert surface. □

Theorem 3337.3 (Urysohn normal second-countable scope)**.** *Let $U = (T, N, S, Hd, B, F, D, M, G, R, Q, C, P, L)$ be an accepted* $\mathsf{UrysohnMetrization}^{\uparrow}$ *packet. Any consumer that treats the source topology as metrizable must read the normal-space row N, the second-countable schedule S, B, and the Hausdorff row Hd before reading the metric handoff M. The packet exports no metrization claim for a source lacking any of these displayed rows.*

Proof. Begin with the topology source T. A topology row alone supplies open-neighbourhood data, but it does not supply the closed-pair separation route, the countable-base schedule, or the separatedness boundary.

The required route is exactly T, N, S, Hd, B, F, D, M. The rows N and F expose normal closed-pair separation; the rows S and B expose the finite base windows; and Hd records the Hausdorff separation boundary consumed by the metric handoff.

If one of these rows is absent, the carrier has no alternate coordinate from which a metric can be read. The refusal row Q therefore blocks any broader classical metrization assertion. □

Theorem 3337.4 (Urysohn metric handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* $\mathsf{UrysohnMetrization}^{\uparrow}$ *packet receives the route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{NormalSpace}^{\uparrow},\ \mathsf{SecondCountable}^{\uparrow},\ \mathsf{HausdorffSpace}^{\uparrow},\ B,\ F,\ D,\ M,\ G,\ R,\ C,\ P,\ L.$$

The metric row M is consumed as a downstream handoff tied to the displayed topology-compatibility rows G, R; it is not a host metric, not a selected embedding metric, and not a claim that arbitrary normal spaces are metrizable.

Proof. Read the carrier rows in order. The source topology, normal-space separation, countable-base schedule, Hausdorff boundary, base ledger, closed-pair separation ledger, and dyadic distance ledger are displayed before M is available.

The compatibility rows G and R bind the handoff back to finite topology windows. Thus a metric consumer receives a named route from the original topology surface to the displayed metric row and back to inspected neighbourhood requests.

Finally use the refusal row Q. Since no row stores a host embedding, a chosen sequence of separating functions, arbitrary classical metrization, or quotient point equality, the handoff remains the finite BEDC route named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{UrysohnMetrization}^{\uparrow}}$.

Constructive story: A $\mathsf{UrysohnMetrization}^{\uparrow}$ packet is generated from a topology source row, normal-space separation row, second-countable base schedule, Hausdorff separation row, finite base-window ledger, closed-pair separation ledger, dyadic distance ledger, metric handoff, topology-compatibility rows, refusal row, replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UrysohnMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3337.1, Definition 3337.2, Definition 3337.3, and Definition 3337.4 seed the finite normal second-countable Hausdorff route to a displayed metric handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UrysohnMetrization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary classical metrization, selected host embeddings, choice-selected separating functions, quotient point equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, normal-space readback, second-countable base scheduling, Hausdorff boundary use, metric handoff, topology compatibility, refusal exactness, replay, provenance, and local naming for BEDC.Derived.UrysohnMetrizationUp.

Chapter 3338

A Concrete Naming Certificate for $\mathsf{TZeroReflection}^{\uparrow}$

Following Kolmogorov and the standard specialization preorder of topology, the $\mathsf{TZeroReflection}^{\uparrow}$ packet records the finite BEDC route from observable opens to the Kolmogorov quotient surface without importing host quotient equality. It sits after the indexed open-row topology of chapter 88, the observable-open truth classifier of chapter 1860, the specialization preorder discipline of chapter 52, and the separated order surface of chapter 53. The packet names the T0 reflection as displayed specialization data; it is not a powerset quotient, not proof irrelevance for opens, not excluded middle over points, and not an ambient set-level identification.

Definition 3338.1 (T0 reflection carrier)**.** *A* $\mathsf{TZeroReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row, carrying the named open indices, finite membership reads, and topology closure witnesses available from chapter 88.*
2. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row, reading each displayed open-index membership as a two-observation classifier rather than as a host proposition.*
3. *R is the* $\mathsf{Preorder}^{\uparrow}$ *specialization row, recording the finite relation $x \preceq y$ by testing that every open read true at x is also true at y through S.*
4. *Q is the* $\mathsf{Poset}^{\uparrow}$ *T0-reflection row, where the public separated order identifies points only by mutual specialization rows already visible in R.*
5. *H records componentwise* hsame *transport for the topology row, Sierpinski truth row, specialization preorder row, T0 poset row, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from an observable-open request through T and S, then into R, and finally into the separated order row Q.*
7. *P is the* Pkg *provenance over* $\mathsf{TZeroReflection}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows, their componentwise transports, their replay route, their provenance, and their local name row. It has no coordinate for host quotient equality, powerset extensionality, excluded middle over opens, selected representatives, or ambient point equality.

Lean4 ✓ `BEDC.Derived.TZeroReflectionUp.TZeroReflectionTasteGate single carrier alignment`

Theorem 3338.2 (T0 reflection NameCert obligations)**.** *For every accepted* $\mathsf{TZeroReflection}^{\uparrow}$ *carrier* $K = (T, S, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, observable-truth, specialization, separated-order, transport, replay, provenance, and local naming rows; observable-open exactness through* T, S*; specialization-preorder exactness through* R*; T0 separated-order handoff through* Q*; and provenance exhaustion through* P, N*. No obligation is discharged by a host quotient of points, a powerset equality of opens, proof irrelevance, excluded middle, selected representatives, or ambient set-level equality.*

Proof. Project the accepted carrier. The only public rows are T, S, R, Q, H, C, P, N, so every local obligation must be read from the indexed topology row, the observable-open truth row, the specialization preorder row, the T0 poset row, or the structural rows.

Observable-open exactness is supplied by T and S. The topology row names the finite open-index requests, and the Sierpinski row turns each request into displayed truth data; no host proposition or powerset membership predicate is available to the packet.

Specialization exactness is then a replay through R. The preorder row can compare two points only by the observable implication route already displayed through S, so the reflexive and transitive reads remain preorder data rather than quotient equality.

Finally Q performs the separated-order handoff. Mutual specialization is exported as the T0 reflection surface carried by the poset row, while H, C, P, N transport, replay, source, and name only the displayed route. Since the excluded host objects have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}$ obligations. □

Theorem 3338.3 (T0 reflection specialization-poset handoff)**.** *Every consumer of an accepted* $\mathsf{TZeroReflection}^{\uparrow}$ *packet that asks for a separated specialization order must factor through the finite route*

$$\mathsf{Topology}^{\uparrow} \longmapsto \mathsf{SierpinskiSpace}^{\uparrow} \longmapsto \mathsf{Preorder}^{\uparrow} \longmapsto \mathsf{Poset}^{\uparrow}.$$

Thus observable-open truth is read before specialization order, and specialization order is read before the public T0 poset surface. The handoff exports no host quotient equality, powerset extensionality, proof irrelevance for opens, excluded middle over points, selected representative function, or ambient set-level equality.

Proof. Start with the carrier $K = (T, S, R, Q, H, C, P, N)$. A consumer first reads T, because the open indices and finite membership requests are the only topology data available to the packet.

Replay C through S. The observable-open truth row converts those open-index requests into Sierpinski classifier data, so later order reads see displayed truth rows rather than host propositions.

The specialization preorder row R receives exactly those truth rows. It may construct reflexive and transitive specialization comparisons, but it has no coordinate for collapsing points by an external quotient.

The final public handoff is Q. The poset row exposes the T0 reflection surface as separated specialization data, and H, P, N preserve the same row order under transport, provenance, and naming. The refused host quotient and choice objects are absent from the carrier, so no consumer can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3338.1, Definition 3338.2, and Definition 3338.3 seed the finite Kolmogorov specialization reflection surface from observable opens to a T0 poset handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TZeroReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host quotient equality, powerset extensionality, proof irrelevance for opens, excluded middle over points, selected representatives, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, observable-open exactness, specialization-preorder exactness, T0 separated-order handoff, transport, replay, provenance, and local naming for BEDC.Derived.TZeroReflectionUp.

Constructive story: A $\mathsf{TZeroReflection}^{\uparrow}$ packet is generated from a Topology indexed-open row, a SierpinskiSpace observable-open truth row, a Preorder specialization row, a Poset T0-reflection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3339

A Concrete Naming Certificate for $\mathsf{MetricEmbedding}^{\uparrow}$

Following Bourbaki and Kelley on metric embeddings, the $\mathsf{MetricEmbedding}^{\uparrow}$ packet records finite control data for an injective metric-route interface without requiring equality of all distances. It sits between the metric source of chapter 57, the graph and replay surface of chapter 130, the separation consumer of chapter 1411, and the stricter distance-equality interface of chapter 439. The packet is weaker than $\mathsf{IsometricEmbedding}^{\uparrow}$: it exposes displayed graph-distance control and separated-completion handoff rows, but it does not certify distance preservation, quotient completion classes, selected representatives, or host function equality.

Definition 3339.1 (MetricEmbedding carrier)**.** *A* $\mathsf{MetricEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, F, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *carrier and classifier row;*
3. *F is the finite graph row displaying the source-to-target relation used by consumers;*
4. *D is the graph-distance control row, recording source distance reads, target distance reads, and the permitted control comparison between them;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance and distance-read seal attached to the comparisons in D;*
6. *S is the separated-metric handoff row for consumers that need point separation after reading the graph;*
7. *H stores componentwise* hsame *transports for X, Y, F, D, R, S and for the structural rows below;*
8. *C stores the displayed* Cont *replay from source metric reads to graph reads, target metric reads, distance-control reads, and separated-consumer reads;*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}$ *row.*

The carrier contains no host function object, no quotient map, no selected completion representative, no ambient completion theorem, and no $\mathsf{IsometricEmbedding}^{\uparrow}$ *distance-equality certificate.*

Lean4 (definition) `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier`

Theorem 3339.2 (MetricEmbedding carrier and classifier obligations)**.** *For every accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *carrier* $E = (X, Y, F, D, R, S, H, C, P, N)$*, the local carrier and classifier surface consists exactly of the displayed ten-row packet. Carrier admission reads the source metric row, target metric row, graph row, graph-distance control row, real-distance seal, separated handoff, transport, replay, provenance, and local naming rows. Classifier transport compares accepted packets componentwise through the* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg*, and* $\mathsf{NameCert}$ *surfaces, and it has no coordinate for host function equality, quotient maps, selected representatives, or isometric distance equality.*

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.TasteGate.MetricEmbeddingTasteGate single carrier alignment`

Proof. Project the carrier definition. The accepted packet exposes exactly $X, Y, F, D, R, S, H, C, P, N$, so carrier admission is the finite BHist row inventory and not an external embedding object.

Classifier transport is componentwise on the same rows. The source and target metric classifiers compare X and Y, the continuous-map graph surface compares F, the distance-control and real-seal rows compare D and R, and the separated handoff compares S only after the graph and metric rows have been displayed.

The structural rows do not add analytic coordinates. They transport, replay, package, and name the same packet, so host functions, quotient maps, selected completion representatives, and the stricter $\mathsf{IsometricEmbedding}^{\uparrow}$ equality surface cannot be read from this carrier. □

Theorem 3339.3 (MetricEmbedding graph-distance control)**.** *Let* $E = (X, Y, F, D, R, S, H, C, P, N)$ *be an accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *packet. Every distance-control request first reads a source metric observation in* X*, then the displayed graph row* F*, then the target metric observation in* Y*, and only then the comparison row* D *with its* $\mathsf{Real}^{\uparrow}$ *seal* R*. The request receives the named graph-distance control data, not an isometric equality of source and target distances and not a detached continuous map.*

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier graph distance control`
↪ variant `BEDC.Derived.MetricEmbeddingUp.GraphDistanceControl.MetricEmbeddingGraphDistance control`
↪ variant `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingGraphDistanceControl`

Proof. Start with a distance-control request. Carrier projection forces the request to pass through the source metric row X and the graph row F before it can identify the corresponding target metric row Y.

Read the comparison row D only after those observations have been displayed. The seal R records the finite real-distance and tolerance data attached to this comparison, while H and C keep the source, graph, target, and real rows aligned.

The packet has no coordinate asserting equality of all source and target distances. Therefore the exported datum is the displayed graph-distance control route, weaker than the $\mathsf{IsometricEmbedding}^{\uparrow}$ distance-equality surface and independent of any host-level continuous map object. □

Theorem 3339.4 (MetricEmbedding separated-completion handoff)**.** *Every* $\mathsf{SeparatedMetric}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricEmbedding}^{\uparrow}$ *packet factors through*

$$X \longmapsto F \longmapsto Y \longmapsto D \longmapsto R \longmapsto S$$

with the stored transport, replay, provenance, and local naming rows. The handoff may use graph-distance control to supply separated metric data, but it does not export quotient completion classes, selected representatives, ambient completion existence, or isometric distance preservation.

Lean4 ✓ `BEDC.Derived.MetricEmbeddingUp.MetricEmbeddingCarrier separated completion handoff`

Proof. Project the separated handoff row S. It is downstream from the displayed source metric row, graph row, target metric row, graph-distance control row, and real seal, so a separated consumer cannot read it before those rows have been exhibited.

Apply the graph-distance control theorem. The route to S uses the finite comparison data D, R attached to graph observations, not a host function and not an isometric equality certificate.

Finally read the structural rows H, C, P, N. They record transport, continuation replay, provenance, and local naming for the same displayed packet. Since quotient completion classes, selected representatives, ambient completion theorems, and isometric preservation are absent from the carrier, separated-completion consumers receive only the finite handoff route above. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricEmbedding}^{\uparrow}$ packet is generated from displayed source and target metric rows, a graph row, a graph-distance control row, a RealUp seal, a separated-metric handoff row, componentwise transport, displayed replay, package provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}, \mathsf{obligation})$

Scope closed: The obligation scope enumerates carrier admission, classifier transport, graph-distance control, separated-completion handoff, and nonescape obligations for metric embeddings weaker than isometric embeddings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient completion classes, selected representatives, ambient completion existence, isometric distance preservation, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checking the MetricSpace, SeparatedMetric, BHist, hsame, Cont, Pkg, and NameCert rows against the same finite packet.

Chapter 3340

A Concrete Naming Certificate for $\mathsf{AffineContraction}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, an $\mathsf{AffineContraction}^{\uparrow}$ packet names an affine self-map whose linear part carries a displayed contraction bound. It refines the map-level contraction surface of chapter 1418, reads its bounded-distance comparison through chapter 410, and hands Cauchy approximation data to chapter 1506 and chapter 1473. The packet records the affine decomposition and the contraction ledger without selecting a fixed point, assuming ambient completeness, or identifying host affine maps.

Definition 3340.1 (AffineContraction carrier)**.** *An* $\mathsf{AffineContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N)$$

with the following displayed rows:

1. *X, d are the metric carrier and distance-read rows used by the contraction and completion consumers.*
2. *$V, +, 0$ are the affine-coordinate rows naming the carried additive chart in which the affine decomposition is read.*
3. *L is the linear-part graph row, with displayed application reads inside the same carried chart.*
4. *b is the translation row.*
5. *T is the affine self-map row, read only through the finite graph equation $T(x) = L(x) + b$.*
6. *G is the graph-exactness row tying each accepted read of T to the displayed reads of L, b, and $+$.*
7. *λ is the rational contraction-bound ledger, with nonnegative and strict unit-bound rows for the linear part.*
8. *M is the induced Lipschitz and contraction-modulus row consumed by metric-completion and uniform-completion readers.*
9. *H stores componentwise* hsame *transport for the metric, affine-coordinate, linear-part, translation, self-map, graph, bound, modulus, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay for affine application, graph exactness, contraction-bound readback, modulus extraction, and completion-facing handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{AffineContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$ *row.*

The carrier exposes an affine contraction surface. It contains no coordinate for a selected fixed point, a completed endpoint, quotient equality of maps, ambient completeness, choice, or bridge checking.

Lean4 ✓ `BEDC.Derived.AffineContractionUp.AffineContractionTasteGate single carrier alignment`

Theorem 3340.2 (AffineContraction NameCert obligations)**.** *For an accepted* $\mathsf{AffineContraction}^{\uparrow}$ *carrier* $A = (X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$ *surface consists of carrier admission, affine-coordinate visibility, linear-part graph visibility, translation visibility, self-map graph exactness, rational contraction-bound stability, induced Lipschitz-modulus visibility, metric-completion and uniform-completion handoff visibility, componentwise classifier transport, continuation replay, provenance, and local naming for the finite packet of Definition 3340.1.*

Proof. Carrier admission is projection from the displayed tuple. The rows $X, d, V, +, 0, L, b, T, G, \lambda, M, H, C, P, N$ are exactly the rows available to the local certificate.

Affine-coordinate visibility reads the additive chart rows $V, +, 0$. Linear-part and translation visibility then read L and b before the self-map row T can be consumed.

Graph exactness is the row G. It forces each accepted self-map application to replay through $L(x)$, b, and the displayed addition row, so the affine map is not identified by host function equality.

The contraction-bound and modulus obligations are carried by λ and M. Since λ stores the strict unit-bound ledger for the linear part, the induced modulus is available to contraction and completion consumers without selecting a fixed point.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. These rows export no ambient completeness proof, choice principle, endpoint seal, quotient equality, or bridge-checked standardization. □

Theorem 3340.3 (AffineContraction metric-completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{AffineContraction}^{\uparrow}$ *carrier must pass through*

$$X,\ d,\ V,\ +,\ 0,\ L,\ b,\ T,\ G,\ \lambda,\ M,\ H,\ C,\ P,\ N.$$

The handoff supplies a displayed affine self-map, graph-exact affine decomposition, rational contraction ledger, and induced modulus. It does not supply a selected fixed point, completed endpoint, quotient equality of affine maps, ambient completeness theorem, choice principle, or bridge-checked standard object.

Proof. Read the carrier of Definition 3340.1. The only completion-facing rows are the metric rows X, d, the affine decomposition rows $V, +, 0, L, b, T, G$, the contraction ledger λ, and the induced modulus row M.

The graph row G orders the affine read before any completion consumer sees the self-map. The consumer therefore receives a finite route from x to $L(x) + b$, not a host affine function.

The strict unit-bound row in λ and the induced modulus row M are the only contraction data exported to chapter 1506 and chapter 1473. A completion route may consume those rows to build Cauchy approximation data, but the endpoint seal and completeness proof remain outside this packet.

Transport and replay are exhausted by H and C, while P and N only record provenance and local naming. Thus the handoff is precisely the finite affine-contraction surface and cannot provide the excluded endpoint, quotient, choice, completeness, or bridge data. □

Closure status for $\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3340.1, Definition 3340.2, and Definition 3340.3 seed the affine contraction packet over affine decomposition, contraction ledger, and completion-facing modulus handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AffineContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected fixed points, completed endpoints, quotient equality of affine maps, ambient completeness, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-coordinate visibility, linear-part graph visibility, translation visibility, self-map graph exactness, rational bound stability, modulus visibility, completion handoff visibility, transport, replay, provenance, and local naming for BEDC.Derived.AffineContractionUp.

Constructive story: An $\mathsf{AffineContraction}^{\uparrow}$ packet is generated from a metric carrier, affine-coordinate rows, linear-part graph, translation row, affine self-map graph, rational contraction-bound ledger, induced modulus, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 3341

A Concrete Naming Certificate for $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$

Following standard universal-algebra expression provenance, the $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ packet realizes the operation-ledger row of chapter 9355 as finite BEDC data. It sits after the binary-operation carrier of chapter 78, the associativity surface of chapter 79, and the unit-bearing surface of chapter 40. The packet records how an algebra consumer reads expression trees and law rows without replacing them by host algebraic equality, quotient classes of terms, or abstract parameter transport.

Definition 3341.1 (Algebraic expression ledger carrier)**.** *An* $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E, O, A, U, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the expression-tree row, recording variables, constants, unary operation nodes, binary operation nodes, and the parenthesization route as finite* BHist *data.*
2. *O is the operation-source row, exposing the* $\mathsf{Magma}^{\uparrow}$ *binary operation row and any named unary or nullary operation rows consumed by the expression tree.*
3. *A is the associativity-law provenance row, read only when a* $\mathsf{Semigroup}^{\uparrow}$ *source supplies the displayed rebracketing witness used by the ledger.*
4. *U is the unit-law provenance row, read only when a* $\mathsf{Monoid}^{\uparrow}$ *source supplies the displayed left-unit or right-unit witness used by the ledger.*
5. *S is the classifier-stability row, recording that expression evaluation is transported through the carrier classifier by the operation congruence rows already present in the source certificate.*
6. *H records componentwise* hsame *transport for E, O, A, U, S and the local naming row.*
7. *C records displayed* Cont *replay of expression-tree formation, operation-source lookup, law-row replay, and classifier-stability transport.*
8. *P is the* Pkg *provenance over* $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$, $\mathsf{Magma}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AlgebraicExpressionLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for host algebraic equality, quotient term equality, invisible normalization, detached operation symbols, or abstract transport between unrelated carriers.

Lean4 ✓ BEDC.Derived.AlgebraicExpressionLedgerUp.AlgebraicExpressionLedgerTasteGate single carrier alignment
Lean4 (statement) BEDC.Derived.AlgebraicExpressionLedgerUp

Theorem 3341.2 (Algebraic expression ledger NameCert obligations)**.** *For every accepted* AlgebraicExpressionLedg *carrier* $L = (E, O, A, U, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AlgebraicExpressionLedger}^{\uparrow}}$ *surface consists exactly of expression-tree exposure through* E*, operation-source exposure through* O*, associativity provenance through* A *when a semigroup row is consumed, unit-law provenance through* U *when a monoid row is consumed, classifier-stability exposure through* S*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape of host equality, quotient term equality, invisible normalization, detached operation symbols, and abstract carrier transport.*

Lean4 ✓ BEDC.Derived.AlgebraicExpressionLedgerUp.AlgebraicExpressionLedgerCarrier namecert obligations

Proof. Project the carrier of Definition 3341.1. The rows E, O, A, U, S are the only rows from which an algebraic expression read can be assembled.

The expression tree is first read through E and O. This exposes the actual operation source before any term is evaluated, so the public operation symbol cannot be detached from the $\mathsf{Magma}^{\uparrow}$ row that licenses it.

The law rows are then conditional on their sources. Associativity may be replayed only through A after the $\mathsf{Semigroup}^{\uparrow}$ source has been displayed, and unit laws may be replayed only through U after the $\mathsf{Monoid}^{\uparrow}$ source has been displayed.

Finally S, H, C, P, N keep classifier transport, continuation replay, provenance, and local naming attached to the same expression tree. Since the packet has no further coordinate, host equality, quotient term equality, invisible normalization, detached operation symbols, and abstract carrier transport cannot replace the displayed ledger. □

Theorem 3341.3 (Algebraic expression ledger nonescape)**.** *Every algebra consumer that reads an accepted* $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ *packet must retain the displayed expression tree, operation-source row, law provenance rows, classifier-stability row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before using an expression result. In particular, the consumer cannot replace the displayed expression provenance by host algebraic equality, quotient syntax equality, a normalization oracle, or abstract parameter transport between carriers.*

Lean4 ✓ BEDC.Derived.AlgebraicExpressionLedgerUp.TasteGate.AlgebraicExpressionLedgerCarrier nonescape

Proof. Begin with the accepted packet $L = (E, O, A, U, S, H, C, P, N)$. The expression result is reachable only through the tree row E and the operation-source row O.

If the consumer uses associativity or unit laws, the source rows A and U must remain visible. These rows point back to the $\mathsf{Semigroup}^{\uparrow}$ and $\mathsf{Monoid}^{\uparrow}$ certificates, so the law read is a displayed ledger replay rather than a host rewrite.

Classifier movement is confined to S and then preserved by H and C. Package provenance and local naming through P, N bind the same route to the public algebraic expression ledger.

Thus no accepted read can discard the expression tree and keep only an algebraic equation. The packet exports finite provenance, not host equality, quotient syntax equality, normalization by an external oracle, or carrier transport unrelated to the displayed source. □

Falsifiable prediction: Every accepted $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ consumer exposes expression tree, operation-source row, associativity provenance when used, unit-law provenance when used, classifier stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an expression result is public.

Independence witness: magma, semigroup, monoid: none is bijective to $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ because the ledger jointly carries expression-tree provenance, operation-source rows, optional associativity and unit-law provenance, classifier stability, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AlgebraicExpressionLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AlgebraicExpressionLedger}^{\uparrow}$ packet is generated from expression-tree rows, operation-source rows, associativity-law provenance, unit-law provenance, classifier-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlgebraicExpressionLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3341.1, Definition 3341.2, and Definition 3341.3 seed the finite operation-provenance packet for the algebra interface spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlgebraicExpressionLedger}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.AlgebraicExpressionLedgerUp`

Bridge status: none

Not claimed: This chapter does not close host algebraic equality, quotient term equality, global normalization, abstract algebra transport, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, expression-tree exposure, operation-source exactness, associativity and unit-law provenance, classifier-stability transport, replay stability, provenance non-escape, and field faithfulness for BEDC.Derived.AlgebraicExpressionLedgerUp.

Chapter 3342

A Concrete Naming Certificate for $\mathsf{MetricClosure}^{\uparrow}$

Following Kelley on metric topology and Bishop–Bridges constructive metric spaces, the $\mathsf{MetricClosure}^{\uparrow}$ packet names the finite closure interface by which metric tests, neighbourhood tests, located-set rows, finite StreamName windows, RegSeqRat readbacks, and Real seals expose a candidate limit as closure evidence. It sits over the metric source of chapter 57, the topology surface of chapter 88, the located-set interface of chapter 1541, the sequence-limit handoff of chapter 1571, the finite windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a named BEDC surface for metric closure reads; it is not a host topological closure operator, not an arbitrary subsequence selector, not open-cover compactness, not quotient equality, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3342.1 (MetricClosure carrier)**.** *A* $\mathsf{MetricClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, S, L, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the finite metric comparison surface.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *row naming the neighbourhood tests compatible with M.*
3. *S is the subset or* $\mathsf{LocatedSet}^{\uparrow}$ *row whose admitted points are being closed.*
4. *L is the candidate limit row, read only through the same metric and topological tests.*
5. *U is the finite neighbourhood and metric-test ledger, pairing each displayed neighbourhood request with the matching metric tolerance request.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read the displayed approximating points and the candidate limit.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the same windows, tolerances, and metric-comparison requests.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, T, S, L, U, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the metric, topology, subset or located-set, candidate limit, test ledger, window, readback, and Real seal rows.*

10. *C records displayed* Cont *replay from neighbourhood requests to metric tests, then through StreamName windows, RegSeqRat readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose metric row, topology row, subset or located-set row, candidate limit, paired test ledger, finite windows, rational readbacks, Real seal, transport, replay, provenance, and local naming are all displayed. It has no coordinate for a host closure operator, arbitrary subsequence choice, open-cover compactness, quotient equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3342.2 (MetricClosure NameCert obligations)**.** *For every accepted* $\mathsf{MetricClosure}^{\uparrow}$ *carrier* $K = (M, T, S, L, U, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient metric row, the compatible topology row, the subset or located-set row, the candidate limit row, the paired neighbourhood and metric-test ledger, StreamName windows, RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public closure read must pass through those rows before it can consume the candidate limit.*

Proof. Project the accepted carrier to the displayed rows $M, T, S, L, U, W, R, E, H, C, P, N$. The metric, topology, subset, and candidate rows give the only admissible source data for a closure read.

The ledger row U then binds each neighbourhood request to a finite metric-test request. The StreamName row W and RegSeqRat row R provide the visible windows and rational readbacks for that same request before the Real seal E is exposed.

Finally H, C, P, N transport, replay, source, and name the same displayed packet. Since no row stores a host closure operator, an arbitrary subsequence, an open-cover object, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the NameCert obligations are exhausted by the listed rows. □

Lean4 ✓ `BEDC.Derived.MetricClosureUp.MetricClosureCarrier namecert obligations`

Theorem 3342.3 (MetricClosure sequential-closure handoff)**.** *Every* $\mathsf{SequentialClosure}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricClosure}^{\uparrow}$ *packet reads the candidate limit through the finite route*

$$M, T, S, L, U, W, R, E, H, C, P, N.$$

The consumer receives a displayed metric row, a compatible topology row, a subset or located-set row, a candidate limit, paired neighbourhood and metric-test rows, finite StreamName windows, RegSeqRat readbacks, a RealUp seal, transport, replay, provenance, and local naming. It receives no ambient topological closure operator, arbitrary subsequence selector, open-cover compactness principle, quotient-equality row, or RealUp completeness theorem.

Proof. Start with a sequential-closure consumer. The carrier exposes the candidate limit L only together with the common metric row M, topology row T, and source row S, so the consumer cannot detach the limit from the displayed metric/topology interface.

Use the test ledger U to align the neighbourhood read with the metric tolerance read. The finite window row W, regular rational row R, and Real seal E then give the same handoff surface used by the $\mathsf{SequentialClosure}^{\uparrow}$ packet of chapter 1571.

The rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Thus the handoff is the finite metric-closure surface itself, not a stronger host closure, subsequence-choice, compactness, quotient, or completeness principle. □

Falsifiable prediction: Every $\mathsf{MetricClosure}^{\uparrow}$ public read factors through a Metric row, a Topology row, a subset or LocatedSet row, a candidate-limit row, paired neighbourhood and metric-test ledgers, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: metric, topology, locatedset, sequentialclosure, streamname, regseqrat, real: $\mathsf{MetricClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric/topology compatibility surface, a source subset or located-set row, a candidate limit, paired neighbourhood and metric tests, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricClosure}^{\uparrow}$ packet is generated from a Metric row, a compatible Topology row, a subset or LocatedSet row, a candidate limit, paired neighbourhood and metric-test rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3342.1, Definition 3342.2, and Definition 3342.3 seed the finite metric/topology closure interface for displayed candidate-limit reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topological closure, arbitrary subsequence choice, open-cover compactness, quotient equality, $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric/topology compatibility, located-source exposure, paired test-ledger stability, StreamName and RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricClosureUp.

Chapter 3343

A Concrete Naming Certificate for $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ packet records the finite equivalence between the Cauchy criterion read and an explicit convergence-modulus schedule before any completion or real-name consumer sees a terminal seal. It sits between the criterion surface of chapter 392, the convergence-modulus schedule of chapter 1438, the finite windows of chapter 126, the regular readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet is a finite-tail equivalence object for L10 Real and Completion consumers; it is not a quotient-stream equality, selected limit, countable-choice schedule, host convergence theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3343.1 (CauchyCriterionEquivalence carrier)**.** *A* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, Q, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence and source-window row, exposing only the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows of the sequence under discussion.*
2. *Q is the criterion tail request row, naming the finite late-pair request consumed by the* $\mathsf{CauchyCriterion}^{\uparrow}$ *surface.*
3. *M is the convergence-modulus schedule row, naming the finite threshold data consumed by the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *surface.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic radius, split tolerances, slack cells, and comparison budget shared by Q and M.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed source windows and tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source windows, criterion request, modulus schedule, dyadic tolerance ledger, regular readback, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from source windows to criterion requests, from criterion requests to modulus schedules, from both finite surfaces to dyadic tolerance reads, and from the tolerance-readback route to the real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, M, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice selection of thresholds, host real equality, detached convergence proofs, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3343.2 (CauchyCriterionEquivalence NameCert obligations)**.** *For every accepted* CauchyCriterionEquival *carrier*

$$K = (S, Q, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$ *surface consists exactly of the bidirectional finite criterion/modulus obligations over the displayed packet. A criterion-facing read of Q may expose a convergence-modulus schedule only by reading the same source windows S, dyadic tolerance ledger D, regular readback R, real seal E, and structural rows H, C, P, N. Conversely, a modulus-facing read of M may expose the corresponding Cauchy-criterion tail request only through those same displayed rows. Neither direction exports quotient stream equality, a selected limit, countable choice over thresholds, host convergence, detached* $\mathsf{Real}^{\uparrow}$ *completeness, or an ambient completion principle.*

Proof. Project the carrier of Definition 3343.1. The criterion request Q and the convergence-modulus schedule M are separate displayed rows, but they share the source-window row S, the dyadic tolerance row D, the regular readback R, the real seal E, and the structural rows.

For the criterion-to-modulus direction, read a finite late-pair request through Q. The continuation row C permits replay only along the displayed route from S through D and R before the schedule row M is exposed, so the modulus threshold is a packet-local finite row rather than an external selector.

For the modulus-to-criterion direction, read a displayed threshold in M. The same tolerance ledger D and readback R supply the late-pair comparison consumed by Q, and the transports in H preserve all rows componentwise under the local classifier.

Finally, the real seal E is downstream of both finite surfaces, while P and N only source and name the same packet. Since the carrier contains no quotient stream, selected limit, countable-choice schedule, host convergence theorem, detached real-completeness row, or ambient completion coordinate, the two finite directions exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyCriterionEquivalenceUp.CauchyCriterionEquivalenceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ packet is generated from source sequence windows, a criterion tail request, a convergence-modulus schedule, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 3343.1 and Definition 3343.2 seed the finite bidirectional criterion/modulus surface over $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable-choice schedules, host convergence, detached $\mathsf{Real}^{\uparrow}$ completeness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus projection, modulus-to-criterion projection, shared tolerance-readback exactness, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyCriterionEquivalenceUp.

Chapter 3344

A Concrete Naming Certificate for $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$

Following Barendregt and Coquand-style metatheory, the $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ packet records the finite BEDC witness surface by which MetaCIC closedness preservation is consumed before any subject-reduction theorem is claimed. It sits after the audit rows of chapter 786, the closure-trace packet of chapter 795, and the dependent-codomain transport packet of chapter 804. The packet gives downstream closed-normal and subject-reduction routes a single human-lineage certificate for closedness preservation without treating closedness as typing preservation.

Definition 3344.1 (MetaCIC closure-preservation witness carrier)**.** *A* $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (A, T, D, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ *row that names the shift, substitution, beta-step, and finite beta-chain closedness-preservation audit chain;*
2. *T is the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *row that records the accepted closure-preservation theorem rows and their finite replay surface;*
3. *D is the* $\mathsf{DependentCodomainClosurePreservation}^{\uparrow}$ *row that exposes codomain closedness transport without discharging dependent-codomain inversion;*
4. *L is the ledger row stating that the witness is a closedness-preservation route, not a typing-preservation or subject-reduction route;*
5. *H records componentwise* hsame *transport for A, T, D, L, P, N;*
6. *C records displayed* Cont *replay routes from the witness to the audit, trace, and dependent-codomain rows;*
7. *P is the* Pkg *provenance over MetaCIC closure preservation, the three sibling packets,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}$ *row.*

The carrier contains no typing context, no conversion equality, no subject-reduction inhabitant, no normalization theorem, no confluence theorem, and no route from closedness preservation to closed consistency.

Lean4 ✓ BEDC.Derived.MetaCICClosurePreservationWitnessUp.MetaCICClosurePreservationWitnessTasteGate
single carrier alignment

Theorem 3344.2 (MetaCIC closure-preservation witness audit chain)**.** *For the carrier of Definition 3344.1, every accepted witness read factors through the chain*

$$\mathsf{ClosurePreservationAuditWitness}^{\uparrow} \longrightarrow \mathsf{MetaCICClosureTrace}^{\uparrow} \longrightarrow \mathsf{DependentCodomainClosurePreservation}^{\uparrow}.$$

The first row supplies the closedness-preservation audit inventory, the second row supplies the finite accepted-declaration trace, and the third row supplies the dependent-codomain closedness transport boundary. The chain exports only closedness-preservation data and its displayed ledger.

Proof. Project the carrier W. The first three rows are A, T, D, so any consumer reaches the audit witness, closure trace, and dependent-codomain transport surface before it can read the ledger.

The audit row A fixes the shift, substitution, beta-step, and beta-star closedness rows. The trace row T records those rows as accepted MetaCIC closure-preservation declarations, while the dependent-codomain row D records the codomain closedness transport boundary.

Finally inspect L, H, C, P, N. The ledger marks the route as closedness-only, and the transport, replay, provenance, and local naming rows preserve only the displayed chain. No coordinate adds typing preservation or subject reduction, so the audit chain has exactly the stated scope. □

Theorem 3344.3 (MetaCIC closure-preservation witness non-escape)**.** *Every public read of an accepted* $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ *carrier remains inside*

$$A, T, D, L, H, C, P, N.$$

A consumer may cite the $\mathsf{ClosurePreservationAuditWitness}^{\uparrow}$ *audit row, the* $\mathsf{MetaCICClosureTrace}^{\uparrow}$ *accepted trace row, the* $\mathsf{DependentCodomainClosurePreservation}^{\uparrow}$ *codomain-transport row, the closedness-only ledger, componentwise transports, displayed replay, provenance, and local naming. It cannot extract subject reduction, type preservation, conversion, confluence, normalization, closed consistency, or a hidden MetaCIC kernel soundness premise.*

Proof. Carrier projection leaves exactly the finite tuple A, T, D, L, H, C, P, N. The three witness rows are already named as closedness-preservation surfaces, and the ledger row records that no typing-preservation route is being exported.

Replay through C and transport through H keep the same rows visible to downstream consumers. Provenance P identifies the sibling packets and the MetaCIC closure-preservation route, while N names the local certificate.

For any stronger request, inspect the carrier inventory. There is no coordinate for typing contexts, conversion equality, confluence, normalization, closed consistency, or kernel soundness. Thus every public read remains confined to the displayed closedness-preservation witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}$ packet is generated from the closure-preservation audit witness, MetaCIC closure trace, dependent-codomain closure-preservation transport row, closedness-only ledger, componentwise transports, displayed replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3344.1, Definition 3344.2, and Definition 3344.3 bind the finite witness surface joining audit rows, accepted trace rows, and dependent-codomain closedness transport.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosurePreservationWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove subject reduction, typing preservation, dependent-codomain inversion, confluence, normalization, closed consistency, Lean verification of the witness packet, or a public schema bridge.

Upgrade path: Obligation closure requires checked carrier admission, audit-chain exactness, dependent-codomain boundary preservation, closedness-only ledger exactness, provenance, local naming, and non-escape for the same finite witness packet.

Chapter 3345

A Concrete Naming Certificate for $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ packet names the finite tail-comparison route for positive decreasing $\mathsf{RealSeries}^{\uparrow}$ terms. It sits next to the real-series surface of chapter 1581, the displayed limit interface of chapter 1413, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, and the terminal real-name seal of chapter 38. It refines the dyadic-block test packet of chapter 1478 by making the $\mathsf{RealSeries}^{\uparrow}$ tail carrier itself visible to Real and Completion consumers before any infinite sum or completed endpoint is read.

Definition 3345.1 (CauchyCondensationRealSeries carrier)**.** *A* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, M, B, G, W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive decreasing* $\mathsf{RealSeries}^{\uparrow}$ *source row. Its terms are admitted only through finite term windows, and its decreasing evidence is read as adjacent finite comparisons.*
2. *M is the monotone-tail ledger, recording positivity, decreasing transport, and the finite tail cut from which the comparison begins.*
3. *B is the dyadic block-window ledger. It lists the inspected powers of two, the associated block endpoints, and the finite intervals of original terms visible at each requested tolerance.*
4. *G is the condensed-term row, displaying the term $2^n a_{2^n}$ on the same dyadic indices as B and tying it to the original* $\mathsf{RealSeries}^{\uparrow}$ *term windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule used for the original terms, block endpoints, condensed terms, and partial-sum windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every displayed original term, condensed term, and block partial sum.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and block-comparison ledger. It records the finite inequalities that bound each inspected original tail block above and below by the matching condensed-term rows.*

8. *R is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *route for the partial-sum and condensed-partial-sum windows that the packet exposes to downstream limit consumers.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, M, B, G, W, Q, D, R have been displayed.*
10. *H records componentwise* hsame *transport for the source, monotone-tail ledger, dyadic blocks, condensed terms, schedules, readbacks, tolerance ledger, limit route, Real seal, continuation, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a tail request through the monotone ledger, dyadic blocks, condensed terms, stream schedules, regular readbacks, dyadic comparisons, limit route, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, B, G, W, Q, D, R, E, H, C, P, N$. It has no coordinate for host infinite sums, quotient real equality, countable choice, trichotomy, selected endpoints, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationRealSeriesUp.CauchyCondensationRealSeriesTasteGate`
`single carrier alignment`

Theorem 3345.2 (CauchyCondensationRealSeries obligations)**.** *For every accepted* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *carrier $K = (A, M, B, G, W, Q, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed thirteen-row packet; positive decreasing* $\mathsf{RealSeries}^{\uparrow}$ *source exposure through A and M; dyadic block-window coverage through B; condensed-term formation through G;* $\mathsf{StreamName}^{\uparrow}$ *scheduling through W;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q;* $\mathsf{DyadicRatCore}^{\uparrow}$ *block comparison through D;* $\mathsf{RealSequenceLimit}^{\uparrow}$ *partial-sum routing through R;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3345.1. The accepted packet exposes exactly the source, monotone-tail ledger, dyadic block windows, condensed terms, stream schedules, regular readbacks, dyadic comparison ledger, limit route, Real seal, and structural rows.

The source obligations are finite. Positivity and decreasing behavior are read from M only on the term windows selected by W, and the original term observations are read back through Q before any block comparison is formed.

The condensation obligations are carried by B, G, D. The dyadic ledger B supplies the finite block endpoints, G forms the corresponding condensed terms, and D records the block inequalities that compare the original tail with the condensed tail on those displayed windows.

The consumer route is then R followed by E. The limit-facing row R receives only the partial-sum and condensed-partial-sum windows already displayed, and the Real seal E is reached only after that route has passed through the regular and dyadic ledgers.

Finally H, C, P, N transport, replay, source, and name the same packet. Since host infinite sums, quotient real equality, countable choice, trichotomy, selected endpoints, and ambient completeness are not coordinates of K, they cannot appear as local NameCert obligations. □

Theorem 3345.3 (CauchyCondensationRealSeries Cauchy route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ *packet factors its tail-comparison read through*

the finite route

$$A, M \longmapsto B, G \longmapsto W, Q, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the original positive decreasing $\mathsf{RealSeries}^{\uparrow}$ *tail and the condensed tail are compared only by displayed dyadic block windows, scheduled stream reads, regular rational readbacks, and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *partial-sum route before any* $\mathsf{Real}^{\uparrow}$ *seal is visible.*

Proof. Start with the accepted carrier. The source row A and monotone ledger M are the only rows from which a consumer can inspect the positive decreasing real-series tail.

Read the dyadic block rows next. The ledger B names the finite block windows, while G names the condensed terms on the same dyadic indices. No consumer can read a condensed comparison without preserving both rows.

Now pass through W, Q, D. The schedule W fixes the finite term and partial-sum windows, Q gives the regular rational readback for those windows, and D records the dyadic inequalities that compare each displayed original block with its condensed row.

The partial-sum route R is therefore already a displayed $\mathsf{RealSequenceLimit}^{\uparrow}$ route before the seal E is reached. The remaining rows H, C, P, N preserve and name that same finite route, so the handoff supplies no host infinite sum, quotient real equality, countable-choice schedule, trichotomy split, selected endpoint, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3345.1, Definition 3345.2, and Definition 3345.3 seed the finite tail-comparison carrier for positive decreasing $\mathsf{RealSeries}^{\uparrow}$ terms, dyadic block windows, condensed terms, finite schedules, regular readbacks, dyadic comparison ledgers, limit routing, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationRealSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, monotone-tail exposure, dyadic block-window coverage, condensed-term formation, schedule and readback stability, dyadic comparison, limit routing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCondensationRealSeriesUp.

Constructive story: A $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$ packet is generated from a positive decreasing RealSeries source row, a monotone-tail ledger, dyadic block windows, condensed terms, StreamName schedules, RegSeqRat readbacks, DyadicRatCore comparison rows, a RealSequenceLimit route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3346

A Concrete Naming Certificate for VariationOfConstants$^{\uparrow}$

Following Coddington–Levinson and Hartman on ordinary differential equations, the VariationOfConstants$^{\uparrow}$ packet records the finite inhomogeneous linear-ODE handoff by which a homogeneous local flow, a forcing row, and a Gronwall control row determine the displayed solution update without importing a host solution operator.

The packet sits next to the ODE source surface of chapter 193, the Picard–Lindelof existence route of chapter 1888, the local-flow endpoint packet of chapter 1899, the Gronwall comparison surface of chapter 2034, and the real endpoint seal of chapter 38. It names the standard variation-of-constants route as a BEDC certificate over displayed rows rather than as an ambient formula in a function space.

Definition 3346.1 (Variation-of-constants carrier)**.** *A* VariationOfConstants$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (O, P, L, G, R, H, C, Q, N)$$

with the following visible rows:

1. *O is the* Ode$^{\uparrow}$ *inhomogeneous linear source row, displaying the time window, initial state row, coefficient row, forcing row, and endpoint request.*
2. *P is the* PicardLindelof$^{\uparrow}$ *existence-and-uniqueness route read on the same local window.*
3. *L is the* PicardLindelofLocalFlow$^{\uparrow}$ *homogeneous local-flow row, naming the propagator surface used before the forcing contribution is accumulated.*
4. *G is the* GronwallInequality$^{\uparrow}$ *comparison row that bounds the accumulated forcing and the difference between any two displayed variation-of-constants reads.*
5. *R is the* Real$^{\uparrow}$ *endpoint and error-seal row for the finite observation of the homogeneous part, forcing integral ledger, and output state.*
6. *H stores componentwise* hsame *transport for the ODE source, Picard route, local-flow row, Gronwall row, real endpoint seal, replay row, provenance row, and local name.*
7. *C stores the displayed* Cont *replay from the ODE source through the Picard route, homogeneous flow row, forcing ledger, Gronwall comparison, and real endpoint seal.*
8. *Q is the* Pkg *provenance over* VariationOfConstants$^{\uparrow}$, Ode$^{\uparrow}$, PicardLindelof$^{\uparrow}$, PicardLindelofLocalFlow$^{\uparrow}$, GronwallInequality$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only by the displayed source, Picard route, homogeneous-flow row, forcing ledger as replayed by C, Gronwall comparison, real endpoint seal, transport, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.VariationOfConstantsUp.VariationOfConstantsTasteGate single carrier alignment`

Theorem 3346.2 (Variation-of-constants NameCert obligations)**.** *For the carrier of Definition 3346.1, the local* $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$ *surface consists exactly of carrier admission for $V = (O, P, L, G, R, H, C, Q, N)$ ODE source admission through O, Picard–Lindelof route visibility through P, homogeneous local-flow visibility through L, Gronwall comparison visibility through G, endpoint sealing through R, componentwise transport through H, displayed replay through C, provenance through Q, and local naming through N. The packet exports no host fundamental matrix, selected global solution curve, quotient path equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier. The fields $O, P, L, G, R, H, C, Q, N$ are the only available rows, so every local naming obligation must be read from those fields.

The source and existence rows come first. The ODE row O supplies the inhomogeneous linear problem, and P records the Picard–Lindelof route on the same displayed window. The homogeneous row L is therefore not a detached fundamental matrix; it is the local-flow packet already named by chapter 1899.

The forcing contribution is read only by replay. The continuation row C accumulates the forcing ledger after the homogeneous flow row has been displayed, and the comparison row G bounds the accumulated read through the Gronwall surface of chapter 2034.

Finally R, H, Q, N seal, transport, source, and name the same finite route. Since no carrier coordinate stores a host solution operator, selected global curve, quotient path equality, choice schedule, or ambient completeness theorem, none of those objects can become a NameCert obligation. □

Theorem 3346.3 (Variation-of-constants Picard–Gronwall handoff)**.** *Every accepted* $\mathsf{VariationOfConstants}^{\uparrow}$ *carrier factors each inhomogeneous local-ODE read through the finite route*

$$\mathsf{Ode}^{\uparrow} \longmapsto \mathsf{PicardLindelof}^{\uparrow} \longmapsto \mathsf{PicardLindelofLocalFlow}^{\uparrow} \longmapsto \mathsf{GronwallInequality}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Picard–Lindelof rows supply the displayed local-flow surface, the variation-of-constants replay reads the forcing ledger on that same surface, and the Gronwall row bounds the endpoint and comparison error before the $\mathsf{Real}^{\uparrow}$ *seal is exposed.*

Proof. Begin with O and P. They fix the ODE source row and the local Picard–Lindelof existence route on one displayed time window.

Read the homogeneous local flow next. The row L is consumed on the same window and supplies the propagator surface used by the forcing replay; no separate host fundamental matrix is opened.

The forcing step is the continuation route C. It records how the forcing ledger is accumulated after L, and it sends the resulting comparison request to the Gronwall row G.

The endpoint is sealed only after that comparison. The row R gives the visible $\mathsf{Real}^{\uparrow}$ endpoint and error seal, while H, Q, N keep transport, provenance, and local naming tied to the same packet. Hence an inhomogeneous local-ODE consumer sees exactly the Picard–Gronwall route above and cannot bypass it through a selected global solution curve or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3346.1, Definition 3346.2, and Definition 3346.3 seed the finite inhomogeneous ODE route over ODE source, Picard–Lindelof, local-flow, Gronwall, Real endpoint, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationOfConstants}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fundamental matrices, global ODE existence, quotient path equality, selected solution curves, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard–Lindelof route visibility, local-flow visibility, forcing replay exactness, Gronwall comparison visibility, endpoint sealing, transport, replay, provenance, and local naming for BEDC.Derived.VariationOfConstantsUp.

Constructive story: A $\mathsf{VariationOfConstants}^{\uparrow}$ packet is generated from one inhomogeneous Ode source row, one PicardLindelof route, one PicardLindelofLocalFlow homogeneous-flow row, one GronwallInequality comparison row, one Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3347

A Concrete Naming Certificate for CompletionUniformFunctorialEnvelope$^{\uparrow}$

Following Bishop–Bridges constructive completion and Mac Lane functorial language, the CompletionUniformFuncto packet records the finite envelope that lets a completion-facing consumer read the uniform-completion route of chapter 1473, the metric-completion carrier of chapter 1506, the functorial completed-map handoff of chapter 1937, and the extension-uniqueness surface of chapter 397 through one local NameCert boundary. Its real-facing side remains attached to the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38; it does not add a quotient completion, selected representative, countable-choice schedule, or ambient real-completeness coordinate.

Definition 3347.1 (CompletionUniformFunctorialEnvelope carrier)**.** *A* CompletionUniformFunctorialEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, M, F, L, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformCompletion$^{\uparrow}$ *row, exposing the entourage-facing completion presentation and the displayed Cauchy data that it consumes.*
2. *M is the* MetricCompletion$^{\uparrow}$ *row, exposing the metric completion carrier, its distance-readback boundary, and the finite tolerance handoff aligned with U.*
3. *F is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row, recording the completed-map transport and functorial composition ledger over the same source and target completion rows.*
4. *L is the* CompletionUniversalProperty$^{\uparrow}$ *row, recording extension existence and uniqueness over the completion surface presented by U, M, F.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row for the source, target, and completed-map observations inspected by the packet.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed stream windows, metric tolerances, and real-facing observations.*
7. *A is the terminal* Real$^{\uparrow}$ *seal row reached only after S, R, U, M, F, L have been displayed.*
8. *H records componentwise* hsame *transport for the uniform-completion, metric-completion, functoriality, universal-property, stream-window, regular-readback, Real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through the uniform and metric rows, through the functorial completed-map row, through the universal-property row, and finally through the stream, regular-readback, and Real-seal boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, F, L, S, R, A, H, C, P, N$. It has no coordinate for quotient completions, selected representatives, countable choice, host completed spaces, function extensionality outside the displayed hsame *rows, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompletionUniformFunctorialEnvelopeUp.TasteGate.CompletionUniformFunctorialEnvelop`
`single carrier alignment`

Theorem 3347.2 (CompletionUniformFunctorialEnvelope NameCert obligations)**.** *For every accepted* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ *carrier $E = (U, M, F, L, S, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; uniform-completion exposure through U; metric-completion exposure through M; completed-map functoriality through F; extension and uniqueness exposure through L; finite stream-window exposure through S; regular rational readback through R; terminal real-name sealing through A; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3347.1. The accepted packet exposes exactly the uniform-completion row, metric-completion row, functoriality row, universal-property row, stream windows, regular readbacks, Real seal, and structural rows.

The completion obligations are read in the order U, M, F, L. The uniform row supplies the entourage-facing completion presentation, the metric row supplies the distance and tolerance boundary, the functoriality row supplies completed-map transport, and the universal-property row supplies extension and uniqueness over those same completion coordinates.

The real-facing obligations are downstream of the completion envelope. The stream row S and regular readback R expose only finite observations, and the Real seal A is available only after the completion and functorial rows have already been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since quotient completions, selected representatives, countable choice, host completed spaces, external function extensionality, and ambient real completeness are not coordinates of E, they cannot appear as local NameCert obligations. □

Theorem 3347.3 (CompletionUniformFunctorialEnvelope consumer handoff)**.** *Every completion-facing or real-facing consumer of an accepted* $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ *packet factors its read through the finite route*

$$U, M \longmapsto F \longmapsto L \longmapsto S, R, A \longmapsto H, C, P, N.$$

Thus a consumer can use uniform completion, metric completion, functorial completed maps, and extension uniqueness only as rows of one displayed envelope, and it reaches $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data only through the same finite packet.*

Proof. Start with an accepted carrier. The completion-facing side exposes U and M first, so the uniform and metric presentations are read from the same displayed packet rather than from independent host completions.

The functoriality row F then transports completed-map reads over those completion coordinates. The universal-property row L can expose extension and uniqueness only after that functorial route has been named.

The handoff to real-facing consumers is S, R, A. These rows provide the finite stream observations, regular rational readback, and terminal Real seal visible to downstream clients, with no selected representative or quotient-completion coordinate inserted between them.

The structural rows H, C, P, N preserve this route under transport, replay it as a displayed continuation, record provenance, and name the local certificate. Consequently every exported read factors through the displayed envelope and supplies no countable-choice schedule, host completed space, external function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}$ packet is generated from UniformCompletion, MetricCompletion, CauchyCompletionFunctoriality, CompletionUniversalProperty, StreamName, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3347.1, Definition 3347.2, and Definition 3347.3 seed the finite envelope binding uniform completion, metric completion, functorial completed maps, universal extension, finite stream windows, regular readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionUniformFunctorialEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host completed spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion exposure, functorial completed-map transport, universal-property exposure, stream-window and regular-readback stability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionUniformFunctorialEnvelopeUp.

Chapter 3348

A Concrete Naming Certificate for $\mathsf{DarbouxProperty}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxProperty}^{\uparrow}$ packet records the finite intermediate-value route for a displayed real-valued map whose interval image cannot jump over a visible value row. It sits after the real-name boundary of chapter 38, the pointwise graph and modulus rows of chapter 130, the bisection handoff of chapter 1422, the regular rational readback of chapter 301, and the dyadic tolerance core of chapter 291. The chapter names the Darboux property as finite BEDC carrier data, not as a host theorem about all real functions or a choice-selected intermediate point.

Definition 3348.1 (DarbouxProperty carrier)**.** *A* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source interval row, exposing the inspected endpoint cells and orientation data used by the property request.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the graph window, pointwise modulus, and metric-distance witnesses for the displayed map.*
3. *A and B are the endpoint value rows, read as* $\mathsf{Real}^{\uparrow}$ *names attached to the two endpoint cells of I.*
4. *Y is the target value row whose placement between the displayed endpoint values is tested only through finite real-name windows.*
5. *S is the sign or bracket row for the residual comparison $F(-) - Y$, expressed through dyadic and regular rational windows rather than host real subtraction.*
6. *M is the finite modulus budget row selecting the precision at which endpoint, target, and midpoint reads are inspected.*
7. *R is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route that retains a branch whose residual signs still bracket the target value row.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained branch windows and requested tolerances.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal offered to a downstream intermediate-value consumer after the displayed route has been read.*

10. *H stores componentwise* hsame *transport for interval, map, endpoint value, target value, residual bracket, modulus, bisection, regular readback, seal, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a Darboux request through the same finite rows.*
12. *P is the* Pkg *provenance over* $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$ *row.*

The carrier exposes a finite bracketing certificate for target-value passage. It has no coordinate for host real equality, quotient real equality, a selected intermediate point, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Lean4 ✓ `BEDC.Derived.DarbouxPropertyUp.TasteGate.DarbouxPropertyTasteGate single carrier alignment`

Theorem 3348.2 (DarbouxProperty NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$ *surface consists of carrier admission, endpoint and target value visibility, residual bracketing visibility, finite modulus control, intermediate-value bisection handoff, regular rational readback, terminal real seal exposure, componentwise transport, continuation replay, provenance, and local naming for the displayed packet of Definition 3348.1.*

Proof. Project the carrier of Definition 3348.1. The interval row I, continuous-map row F, endpoint rows A, B, and target row Y are the only rows from which a Darboux request may be read.

The residual bracket is the row S. It is inspected at the precision selected by M, so the endpoint and target comparison remains inside finite dyadic and regular rational windows.

The bisection row R is then the only route to a retained branch. Its output is read back through Q before the real-name seal E is exposed.

The structural rows H, C, P, N transport, replay, source, and name that same packet. Since no coordinate stores host equality, quotient equality, a selected point, trichotomy, ambient completeness, or open-cover compactness, none of those objects can appear in the local NameCert surface. □

Theorem 3348.3 (DarbouxProperty intermediate-value route)**.** *Every target-value consumer of an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier factors through*

$$\mathsf{ContinuousMap}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{IntermediateValueBisection}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

In particular, a request for a displayed value between the endpoint reads first reaches the continuous-map and residual-bracket rows, then the finite bisection route, then the regular rational readback, and only then the real-name seal.

Proof. Start from $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$. The continuation row C exposes the interval, map, endpoint, and target rows before any branch row is visible.

The residual bracket S is measured at the budget M, with dyadic tolerance data supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ component named in provenance. This gives a finite sign or comparison surface for $F(-) - Y$.

The branch handoff is exactly R. It is the $\mathsf{IntermediateValueBisection}^{\uparrow}$ route, so the retained branch is visible only through the same finite bisection ledger used by the constructive intermediate-value surface.

The retained branch is then read through Q and sealed by E. The remaining rows H, C, P, N keep the route attached to one BEDC packet, so the consumer cannot bypass the route by appealing to a host intermediate-value theorem or a selected point. □

Theorem 3348.4 (DarbouxProperty real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier is downstream of the endpoint value rows, the target value row, the residual bracket row, the modulus budget, the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. The export is a named finite real-seal handoff, not host real equality, quotient real equality, a selected intermediate point, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or open-cover compactness.*

Proof. Read the route to the seal. The real-facing row E appears after the endpoint and target rows A, B, Y, the residual bracket S, the budget M, the bisection route R, and the regular readback Q in the displayed carrier.

The transport and replay rows H and C preserve those dependencies componentwise. Provenance P and local naming N record that the exposed seal belongs to the $\mathsf{DarbouxProperty}^{\uparrow}$ packet.

Therefore every real-facing read is exhausted by the finite rows already listed. There is no carrier field through which host equality, quotient equality, a selected intermediate point, trichotomy, ambient completeness, or open-cover compactness could be exported. □

Theorem 3348.5 (DarbouxProperty bisection target carrier)**.** *Let* $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier. Every target-value bisection read of* D *is carried by the finite row family*

$$I, F, A, B, Y, S, M, R.$$

The row R *may be inspected only after the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* F*, the endpoint* $\mathsf{Real}^{\uparrow}$ *rows* A, B*, the target* $\mathsf{Real}^{\uparrow}$ *row* Y*, the residual-bracket row* S*, and the finite modulus budget* M *have been displayed. Thus the bisection target is a BEDC packet over displayed residual windows, not a host-selected intermediate point.*

Proof. Project the carrier of Definition 3348.1. The only rows preceding the bisection route are the interval row, continuous-map row, endpoint and target value rows, residual bracket, and modulus budget.

The continuation row C preserves this read order. A target-value request therefore reaches R only after the finite residual comparison has been formed from F, A, B, Y, S, M.

The structural rows H, P, N transport, source, and name the same route. Since no coordinate stores a selected point or a host intermediate-value theorem, the bisection target remains the displayed carrier route. □

Theorem 3348.6 (DarbouxProperty residual-bracket stability)**.** *For an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier* $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$*, extending the inspected bisection window through the same modulus budget preserves the residual bracket read. The comparison row* S *is transported by* H*, replayed by* C*, and consumed by the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *route* R *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance data. No branch extension may replace* S *by real trichotomy or by host real subtraction.*

Proof. Begin with the residual bracket S at the precision selected by M. Its endpoint and target reads are already attached to the displayed $\mathsf{ContinuousMap}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows.

When the bisection window is extended, the replay row C requires the next read to use the same modulus budget and dyadic tolerance route. Componentwise transport in H moves the endpoint, target, and residual rows together, so the sign or comparison data remains tied to the same packet.

The bisection row R consumes exactly that transported residual bracket. Because the carrier has no trichotomy coordinate and no host subtraction coordinate, stability is only the finite persistence of S along the displayed branch route. □

Theorem 3348.7 (DarbouxProperty RegSeqRat readback)**.** *Let $D = (I, F, A, B, Y, S, M, R, Q, E, H, C, P, N)$ be an accepted* $\mathsf{DarbouxProperty}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *seal exposed by E is preceded by a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q of the branch retained by R. The readback uses the dyadic tolerance data named in* $\mathsf{DyadicRatCore}^{\uparrow}$ *and the regular rational schedule of* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal real-name row is public.*

Proof. Read the rows following the bisection handoff. The route R retains the displayed branch, while Q is the only row that converts the retained branch windows into regular rational data.

The dyadic budget already used by S and M supplies the tolerance side of the readback. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q then presents the requested rational windows before the terminal row E is available.

Finally, H, C, P, N keep the readback attached to the same carrier and NameCert. The real seal is therefore a finite readback boundary over Q, not an assertion of host real equality, selected convergence, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DarbouxProperty}^{\uparrow}$ packet is generated from an interval row, a ContinuousMap row, endpoint Real rows, a target Real row, a finite residual-bracket row, a modulus budget, an IntermediateValueBisection route, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$, seed)

Scope closed: Definition 3348.1, Definition 3348.2, Definition 3348.3, Definition 3348.4, Definition 3348.5, Definition 3348.6, and Definition 3348.7 seed the finite Darboux target-value route and its real-seal boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DarbouxProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient real equality, a selected intermediate point, trichotomy, ambient RealUp completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-bracket stability, intermediate-value bisection handoff, regular rational readback, and real-seal nonescape for BEDC.Derived.DarbouxPropertyUp.

Chapter 3349

A Concrete Naming Certificate for $\mathsf{LocatedCompleteMetric}^{\uparrow}$

Following Bishop and Bridges on constructive metric completeness, the $\mathsf{LocatedCompleteMetric}^{\uparrow}$ packet records a complete metric object whose Cauchy-limit service is located at every finite precision request. It sits after the completion surface of chapter 1506, the zero-distance classifier boundary of chapter 1411, the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, and the finite observation schedule of chapter 126. The packet names a constructive complete-metric carrier with displayed location and limit witnesses; it is not a quotient completion class, a selected representative, countable choice, host equality of points, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3349.1 (LocatedCompleteMetric carrier)**.** *A* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, R, Q, W, A, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing the metric carrier, completion handoff, and finite tolerance surface.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row, requiring zero-distance comparisons to pass through the displayed separated metric boundary.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *distance-seal row, used only after finite distance observations have been routed through regular rational readback.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for Cauchy approximants, limit budgets, and distance witnesses.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite observation schedule for Cauchy data and locatedness requests.*
6. *A records componentwise* hsame *transport for the metric-completion source, separated classifier, real seal, regular rational readback, stream windows, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a finite Cauchy request through W, Q, R and then through the separated classifier S over the completion source M.*

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$ *row.*

The carrier exports a displayed complete-metric service only through these rows. It has no coordinate for quotient completion equality, selected limits, a countable-choice schedule, host equality of carried points, or an unrestricted completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCompleteMetricUp.LocatedCompleteMetricTasteGate single carrier alignment`

Theorem 3349.2 (LocatedCompleteMetric NameCert obligation surface)**.** *For every accepted* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier* $L = (M, S, R, Q, W, A, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the metric-completion source, separated classifier, real distance seal, regular rational readback, stream-window schedule, componentwise transport, continuation replay, provenance, and local naming row;*

2. *located request admission, requiring every precision request to be read through the finite* $\mathsf{StreamName}^{\uparrow}$ *window W before any limit witness is exposed;*

3. *regular readback stability, requiring Cauchy approximants, limit budgets, and distance witnesses to pass through Q before the* $\mathsf{Real}^{\uparrow}$ *seal R is consumed;*

4. *separated classifier stability, requiring zero-distance uniqueness for displayed limits to be compared through S over the same completion source M;*

5. *ledger non-escape, excluding quotient completion classes, selected representatives, countable-choice schedules, host equality of carried points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3349.1. The displayed tuple contains exactly the rows $M, S, R, Q, W, A, C, P, N$, so carrier admission and local naming are exhausted by those coordinates.

A located request must enter through W. That row supplies the finite observation schedule, and Q turns the scheduled Cauchy approximants and requested budgets into regular rational readback before any terminal real-distance seal is visible.

The separated comparison is then confined to S over the same metric-completion source M. Transport through A and replay through C preserve the displayed route, while P records that no excluded quotient, selector, choice, host-equality, or ambient-completeness coordinate is exported. □

Theorem 3349.3 (LocatedCompleteMetric Cauchy-limit witness)**.** *Let $L = (M, S, R, Q, W, A, C, P, N)$ be an accepted* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *carrier. Every Cauchy-limit witness served by L factors through the finite route*

$$W \longmapsto Q \longmapsto R \longmapsto M, S \longmapsto A, C, P, N.$$

Thus the witness is a located finite-readback service over the displayed metric-completion source and separated classifier, not a selected host limit, quotient representative, choice sequence, or ambient completeness assertion.

Proof. Begin with a Cauchy-limit request against L. The only observation schedule in the carrier is W, so the request first fixes a finite stream window.

The regular rational row Q supplies the finite readback for the Cauchy approximants and the requested limit budget. The real-distance seal R is read after that regular handoff, so a downstream consumer receives only displayed distance information.

The completion source M and separated classifier S package the witness as a complete-metric service and compare any two displayed limits through zero-distance classification. The structural rows A, C, P, N transport, replay, source, and name the same route, so the excluded host-limit and quotient data have no row from which to be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3349.1, Definition 3349.2, and Definition 3349.3 enumerate the carrier, NameCert obligation surface, and Cauchy-limit witness obligations over MetricCompletion, SeparatedMetric, Real, RegSeqRat, StreamName, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, selected representatives, countable choice, host equality of carried points, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located request admission, regular readback stability, separated classifier stability, and ledger non-escape for BEDC.Derived.LocatedCompleteMetricUp.

Constructive story: A $\mathsf{LocatedCompleteMetric}^{\uparrow}$ packet is generated from a MetricCompletion source row, a SeparatedMetric classifier row, a RealUp distance seal, RegSeqRat readback, StreamName windows, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3350

A Concrete Naming Certificate for $\mathsf{CauchyCompletionBoundary}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley-style completion interfaces, the $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ packet records the finite handoff surface between metric-completion, uniform-completion, named-window, dyadic-tolerance, regular-readback, and Real-seal consumers. It sits over the metric completion route of chapter 1506, the uniform completion route of chapter 1473, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names the completion boundary without importing quotient stream equality, selected limits, ambient completeness, countable choice, or host completion operators.

Definition 3350.1 (Cauchy completion boundary carrier)**.** *A* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the metric Cauchy completion request and its displayed finite source route.*
2. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row naming the uniform completion request and its entourage-facing handoff route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the metric and uniform completion reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to budget Cauchy comparisons, entourage requests, and Real-facing seal reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same finite windows and dyadic tolerances.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after M, U, S, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric-completion, uniform-completion, window, dyadic, readback, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing read through $M, U, S, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$ *row.*

The classifier accepts only packets whose metric-completion row, uniform-completion row, shared finite windows, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming are displayed in this order. It has no coordinate for quotient streams, selected representatives, selected limits, ambient completeness, countable choice, or host completion operators.

Lean4 ✓ `BEDC.Derived.CauchyCompletionBoundaryUp.CauchyCompletionBoundaryTasteGate single carrier alignment`

Theorem 3350.2 (Cauchy completion boundary NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *carrier* $B = (M, U, S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion row, uniform-completion row, shared finite-window row, dyadic tolerance ledger, regular rational readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-boundary read must consume the rows* $M, U, S, D, R, E, H, C, P, N$ *before it exports a Real-facing completion handoff, and no obligation is stored outside those rows.*

Proof. Project the accepted carrier. The projection exposes the metric-completion request, uniform-completion request, shared windows, dyadic tolerances, regular readback, Real seal, transport, replay, provenance, and local name as one finite packet.

Read stability from the row order. A consumer that asks for a completion-facing handoff must pass through the shared $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before R or E can be inspected, so the metric and uniform routes cannot escape into a host completion object.

Finally apply coordinate exhaustion. Quotient streams, selected representatives, selected limits, ambient completeness, countable choice, and host completion operators are not coordinates of B, and the hsame, Cont, Pkg, and NameCert rows preserve the displayed packet rather than adding new analytic data. □

Theorem 3350.3 (Cauchy completion boundary finite handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer citing an accepted* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *packet receives only the finite handoff*

$$M \longmapsto U \longmapsto S \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The handoff supplies the metric-completion route, uniform-completion route, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming. It supplies no quotient stream equality, selected representative, selected limit, countable-choice schedule, ambient completeness theorem, or host completion operator.

Proof. Begin with a consumer of the boundary packet. Carrier projection gives the rows $M, U, S, D, R, E, H, C, P, N$ and no other completion-facing coordinate.

The shared observation route is finite. The row S stores the inspected windows, D stores the dyadic budget, R stores the regular readback, and E is terminal with respect to the displayed Real-facing seal.

The structural rows then keep the same route under transport and replay. Since the excluded quotient, choice, limit, completeness, and host-completion objects are absent from the carrier, every exported handoff is exhausted by the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ packet is generated from MetricCompletion and UniformCompletion rows, shared StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionBoundary}\uparrow}$, seed)

Scope closed: Definition 3350.1, Definition 3350.2, and Definition 3350.3 seed the finite completion-boundary handoff for MetricCompletion, UniformCompletion, and RealUp consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, countable choice, ambient completion, host completion operators, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window discipline, dyadic tolerance routing, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionBoundary

Chapter 3351

A Concrete Naming Certificate for MetricCompletionDenseEmbedding$^{\uparrow}$

Following Kelley and Bourbaki on metric completion, the MetricCompletionDenseEmbedding$^{\uparrow}$ packet records the standard dense embedding of a metric carrier into its completion as finite BEDC data. It reads the metric source of chapter 57, the completion row of chapter 1506, the separated comparison surface of chapter 1411, the uniform completion handoff of chapter 1473, and the regular rational readback of chapter 301. The packet gives completion-uniqueness and universal-property consumers a visible dense-image route without quotient representatives, selected limits, countable choice, host equality of completion classes, or an ambient completion object.

Definition 3351.1 (MetricCompletionDenseEmbedding carrier)**.** *A* MetricCompletionDenseEmbedding$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, C, E, S, U, R, H, K, P, N)$$

with the following displayed rows:

1. *X is the* Metric$^{\uparrow}$ *source row, carrying the source carrier, distance classifier, and finite metric reads inherited from chapter 57.*
2. *C is the* MetricCompletion$^{\uparrow}$ *completion row, carrying the completion-facing Cauchy surface of chapter 1506.*
3. *E is the dense-image row, recording the displayed embedding route from a source point row in X to its completion name in C and the finite witness rows saying that every inspected completion neighbourhood is reached by a source image.*
4. *S is the* SeparatedMetric$^{\uparrow}$ *comparison row from chapter 1411, used only after the source and completion rows have been displayed.*
5. *U is the* UniformCompletion$^{\uparrow}$ *handoff row from chapter 1473, recording the uniform completion consumer route that reads the same dense image surface.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row from chapter 301, exposing the finite regular-rational observations attached to completion-neighbourhood and distance-budget reads.*
7. *H records componentwise* hsame *transport for $X, C, E, S, U, R, K, P, N$.*
8. *K records displayed* Cont *replay from source metric reads through completion names, dense-image witnesses, separated comparisons, uniform completion handoff, and regular rational readback.*

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$*, the five sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient representatives, selected Cauchy limits, countable choice, host equality of completion classes, or a detached completion uniqueness theorem.

Lean4 ✓ `BEDC.Derived.MetricCompletionDenseEmbeddingUp.TasteGate.MetricCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 3351.2 (MetricCompletionDenseEmbedding NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *carrier* $D = (X, C, E, S, U, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionDense}}$ *surface consists exactly of carrier admission for the source metric row, completion row, dense-image row, separated comparison row, uniform completion handoff, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must consume* X, C, E *before it can read* S, U, R, H, K, P, N*.*

Proof. Project the carrier of Definition 3351.1. The rows X, C, E expose the source metric, completion carrier, and dense image surface as displayed finite packet data.

The comparison and handoff rows are downstream. The separated comparison S reads the same source and completion rows, the uniform completion handoff U follows the same dense-image route, and the regular rational readback R exposes only the finite observation data attached to those reads.

Finally H, K, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores quotient representatives, selected limits, countable choice, or host completion equality, the listed rows exhaust the local NameCert obligations. □

Theorem 3351.3 (MetricCompletionDenseEmbedding handoff)**.** *Every completion-uniqueness, separated-completion, or uniform-completion consumer of an accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{Metric}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{DenseImage}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}.$$

The exported handoff is the displayed dense embedding of the metric source into its completion; it is not a quotient-representative selector, an ambient completion theorem, a host equality comparison, or a choice principle.

Proof. Start with a completion-facing consumer. By Definition 3351.2, the consumer first reads the source metric row X, the completion row C, and the dense-image row E.

The sibling chapters then determine the downstream route. The separated metric comparison is read through chapter 1411, the uniform handoff through chapter 1473, and the regular rational windows through chapter 301.

The structural rows H, K, P, N preserve that same finite route under transport, replay, provenance, and naming. Because the carrier has no row for representatives, selected limits, countable choice, host equality, or an ambient completion theorem, every accepted handoff remains the displayed dense embedding surface. □

Theorem 3351.4 (MetricCompletionDenseEmbedding universal-property route)**.** *Every completion universal-property read using the dense image of an accepted* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$

packet factors through the source metric row, completion row, dense image row, separated comparison row, uniform handoff row, RegSeqRat readback row, componentwise transport, displayed replay, provenance, and local NameCert *row. The route exports the source-to-completion dense embedding needed by universal-property consumers, not a quotient-representative selector, a selected limit, countable choice, host equality of completion classes, or an ambient completion theorem.*

Proof. Use Definition 3351.2 to read the carrier surface. A universal-property consumer first receives the source metric row, completion row, and dense image row, so its dense-source request is tied to displayed packet data.

Then apply the handoff of Definition 3351.3. The separated comparison, uniform completion handoff, and RegSeqRat readback rows are the only downstream completion-facing rows available to the consumer.

Finally the transport, replay, provenance, and local naming rows preserve that same route. Since no excluded host-side principle is a carrier coordinate, the universal-property read factors through the dense embedding surface alone. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ consumer that reaches a completion-uniqueness or uniform-completion handoff while omitting the source metric row, completion row, dense-image row, separated comparison row, or regular rational readback refutes the packet.

Independence witness: metric, metriccompletion, separatedmetric, uniformcompletion, regseqrat: no listed sibling is bijective with $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ because the packet jointly carries source metric data, completion data, dense-image witnesses, separated comparison, uniform handoff, regular rational readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a metric source row, metriccompletion row, dense-image row, separated comparison row, uniform completion handoff, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3351.1, Definition 3351.2, and Definition 3351.3 seed the standard dense embedding route from a metric carrier into its completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected Cauchy limits, countable choice, host equality of completion classes, completion uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-image exactness, separated comparison handoff, uniform completion handoff, regular rational readback, and TasteGate field faithfulness for BEDC.Derived.MetricCompletionDenseEmb

Chapter 3352

A Concrete Naming Certificate for ClosedPiDomainSubjectReduction$^{\uparrow}$

The ClosedPiDomainSubjectReduction$^{\uparrow}$ packet realizes the Pi-domain branch of the closed MetaCIC subject-reduction frontier as a finite NameCert carrier. It sits beside the four-row discharge socket of chapter 719, the discharge ledger of chapter 720, the beta-substitution packet of chapter 716, the closed substitution boundary of chapter 538, and the typed beta congruence trigger of chapter 760. The packet names the Pi-domain obligation as displayed closed term data; it does not export an unconditional closed subject-reduction theorem.

Definition 3352.1 (ClosedPiDomainSubjectReduction carrier)**.** *A* ClosedPiDomainSubjectReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (T, D, D', C, Y, Z, B, H, R, Q, N)$$

with the following displayed rows:

1. *T is the displayed closed Pi term row, naming the source term $\Pi D.\, C$ and its closed context boundary.*
2. *D is the original domain row, carrying the domain annotation inspected before the congruence step.*
3. *D' is the reduced domain row reached by the displayed beta step from D.*
4. *C is the codomain row, read under the binder induced by the displayed domain.*
5. *Y is the typing row, recording the source typing judgement for $\Pi D.\, C$ and the target typing judgement demanded for $\Pi D'.\, C$.*
6. *Z is the closedness row, requiring the Pi term, original domain, reduced domain, and codomain reads to remain inside the same closed MetaCIC boundary.*
7. *B is the beta-step and typed congruence row, consuming the displayed Pi-domain beta step and the trigger surface of chapter 760.*
8. *H records componentwise* hsame *transport for $T, D, D', C, Y, Z, B, R, Q, N$.*
9. *R records displayed* Cont *replay from the source Pi term through the domain step, typing row, closedness row, and subject-reduction handoff.*

10. *Q is the* Pkg *provenance over chapter 719, chapter 720, chapter 716, chapter 538, chapter 760,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying the closed Pi term, both domain rows, the codomain row, the typing row, the closedness row, the Pi-domain beta step, transport, replay, provenance, and local naming. It has no coordinate for a blanket closed subject-reduction theorem, a host conversion rule, erased derivation semantics, quotient syntax equality, or an untracked binder context.

Lean4 ✓ `BEDC.Derived.ClosedPiDomainSubjectReductionUp.ClosedPiDomainSubjectReductionTasteGate`
`single carrier alignment`

Theorem 3352.2 (ClosedPiDomainSubjectReduction obligations)**.** *For every accepted* ClosedPiDomainSubjectRed *carrier $P = (T, D, D', C, Y, Z, B, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed closed Pi term, original domain, reduced domain, codomain, typing row, closedness row, beta-step row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The Pi-domain preservation read is accepted only when B displays the domain beta step and Y, Z display the corresponding typing and closedness rows.*

Proof. Project the carrier of Definition 3352.1. The rows T, D, D', C display the Pi term and the two domain readings, while Y and Z display the typing and closedness boundaries attached to the same context.

The beta-step row B is then read as the Pi-domain congruence trigger. It is not interchangeable with the beta-substitution row, the application-argument row, or the lambda-domain row of the subject-reduction socket, because the target branch changes the displayed Pi domain while preserving the codomain under the induced binder.

The structural rows H, R, Q, N transport, replay, source, and name only that finite packet. Since no row stores a host conversion rule, erased derivation semantics, quotient syntax equality, or a closed subject-reduction theorem, the displayed rows exhaust the local NameCert obligations. □

Theorem 3352.3 (ClosedPiDomainSubjectReduction handoff)**.** *Every subject-reduction discharge ledger read that consumes the Pi-domain branch from an accepted* $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ *packet factors through the displayed route*

$$\Pi D.\, C,\ D,\ D',\ C,\ Y,\ Z,\ B,\ H,\ R,\ Q,\ N.$$

The handoff supplies the Pi-domain obligation row to the subject-reduction discharge socket and ledger; it does not supply the beta-substitution row, application-argument row, lambda-domain row, or an unconditional closed subject-reduction theorem.

Proof. Start with a subject-reduction consumer of the Pi-domain branch. The socket and ledger chapters, chapter 719 and chapter 720, require a distinct Pi-domain row before the parameterised subject-reduction route can be read.

The present carrier supplies that row only through T, D, D', C, Y, Z, B. These rows display the source Pi term, the original and reduced domains, the codomain, the typing judgement, the closedness boundary, and the beta-step trigger.

Finally H, R, Q, N preserve and name the same finite handoff. The beta-substitution preservation packet, closed substitution boundary, and typed beta congruence trigger remain sibling inputs rather than hidden substitute proofs, so the handoff is exactly the Pi-domain row and nothing stronger. □

Falsifiable prediction: Any accepted $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ consumer reaching a subject-reduction conclusion without the displayed Pi-domain beta step and typing row refutes the carrier.

Independence witness: metacic-subject-reduction-obligation, betasubstitutionpreservation, closed-substitutionboundary, typedbetacongruencetrigger: none is bijective with $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ because this packet jointly carries the closed Pi term, original domain, reduced domain, codomain, typing row, closedness row, Pi-domain beta step, transport, replay, provenance, and local naming.

The Lean TasteGate file for $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ must provide explicit Nontrivial ClosedPiDomainSu and $\mathsf{FieldFaithful}$ $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ instances preserving $T, D, D', C, Y, Z, B, H, R, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}$ packet is generated from a closed Pi term row, original domain row, reduced domain row, codomain row, typing row, closedness row, beta-step row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3352.1, Definition 3352.2, and Definition 3352.3 seed the finite Pi-domain subject-reduction obligation packet for the closed MetaCIC discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedPiDomainSubjectReduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta-substitution preservation, application-argument stability, lambda-domain preservation, unconditional closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Pi-domain beta-step exactness, typing-row exposure, closedness-row exposure, subject-reduction ledger handoff, and TasteGate field faithfulness for BEDC.Derived.ClosedPiDomainSubjectRed

Chapter 3353

A Concrete Naming Certificate for $\mathsf{FiniteTraceInductionGap}^{\uparrow}$

Following the Bishop and Brouwer distinction between finite verification and an induction schema, the $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ packet records the finite-observation side of Definition 9393.4 and the non-internal-induction boundary of Definition 9393.5. It sits beside the finite-trace GAP socket of chapter 504, the transcendental induction socket of chapter 498, and the closed-observation ledger of chapter 616. The packet names a finite verification trace together with the GAP socket exposed by any attempted universal closure; it does not install a substrate-internal universal quantifier, dependent induction schema, host recursor, or future-observation oracle.

Definition 3353.1 (Finite trace induction gap carrier)**.** *A* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, O, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite verification trace row, listing only the observed indices and their closed-generation witness records.*
2. *O is the observed predicate row, storing the finite family of predicate reads actually checked by the trace.*
3. *A is the attempted induction-closure row, naming the requested passage from the finite trace to a universal claim without admitting the passage as a carrier coordinate.*
4. *Q is the GAP socket row exposed when A asks for substrate-internal induction beyond the displayed trace.*
5. *H records componentwise* hsame *transport for the finite trace, observed predicate rows, attempted closure row, GAP socket, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each observed predicate read through T, O, A, Q, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$, $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, $\mathsf{TranscendentalInductionSocket}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, BHist, Gap, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$ *row.*

The classifier accepts only packets whose finite trace, observed predicate rows, attempted induction-closure row, GAP socket, transport, replay, provenance, and local naming are all displayed. It has

no coordinate for a universal quantifier over unobserved indices, a substrate-internal induction schema, a host dependent recursor, an infinite observation ledger, or a future-observation oracle.

Lean4 ✓ `BEDC.Derived.FiniteTraceInductionGapUp.FiniteTraceInductionGapTasteGate single carrier alignment`

Theorem 3353.2 (Finite trace induction gap NameCert obligations)**.** *For every accepted* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *carrier* $G = (T, O, A, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite trace row, observed predicate rows, attempted induction-closure row, GAP socket row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that asks for induction closure must expose* A *and then consume* Q *as a GAP socket; no substrate-internal universal statement is exported by the packet.*

Proof. Project the accepted carrier. The projection exposes the finite trace T, the observed predicate rows O, the attempted closure row A, the GAP socket Q, and the structural rows H, C, P, N.

Read the induction request through A. If the consumer stays within the finite trace, it can use only the displayed rows T and O. If it asks for all unobserved indices, the next displayed row is the GAP socket Q, not a universal witness.

The transport and replay rows preserve that boundary. Since no universal quantifier, dependent induction schema, host recursor, infinite ledger, or future-observation oracle is a coordinate of G, the local NameCert obligations are exhausted by the finite trace and GAP route. □

Theorem 3353.3 (Finite trace induction gap non-escape)**.** *Every induction-facing consumer of an accepted* $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ *packet factors through*

$$T \longmapsto O \longmapsto A \longmapsto Q \longmapsto H, C, P, N.$$

Thus the consumer may read the observed finite trace and the displayed predicate witnesses, but any attempted export of a universal induction claim reaches only the GAP socket. The packet exports no substrate-internal universal quantifier, no closed dependent recursor, no infinite observation object, and no promise over future observations.

Proof. Start with the carrier rows of Definition 3353.1. The trace row T and observed predicate row O give exactly the finite witness records already present in the closed observational system.

The attempted closure row A names the requested induction passage. It is deliberately followed by the GAP socket row Q, so a consumer that asks for unobserved indices receives a visible boundary rather than a hidden induction rule.

Finally replay and transport the same route through C and H, then read provenance and local naming through P, N. The excluded universal, recursor, infinite-ledger, and future-observation objects are absent from the carrier, so the displayed GAP route is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteTraceInductionGap}^{\uparrow}$ packet is generated from a finite verification trace, observed predicate rows, an attempted induction-closure row, a GAP socket, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3353.1, Definition 3353.2, and Definition 3353.3 seed the finite verification trace and induction-GAP boundary for closed observational systems.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteTraceInductionGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a substrate-internal universal quantifier, dependent induction schema, host recursor, infinite observation ledger, future-observation oracle, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-trace visibility, observed-predicate stability, induction-GAP routing, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteTraceInductionGapUp.

Chapter 3354

A Concrete Naming Certificate for $\mathsf{OvertReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OvertReal}^{\uparrow}$ packet records positive information about real data as finite witnessable contact, not as a host open-set membership predicate. It sits over the real-name boundary of chapter 38, the located source interval of chapter 1709, the finite interval-cover route of chapter 1885, and the one-sided observation rows of chapter 1446 and chapter 1447. The packet supplies an L10 positive-information interface for downstream real and interval consumers without claiming classical open-cover compactness, arbitrary open-set membership, trichotomy, selected points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3354.1 (OvertReal carrier)**.** *An* $\mathsf{OvertReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (R, I, C, L, U, W, A, H, T, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row naming the real datum that may be observed only through the displayed real-name route.*
2. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, exposing endpoint names, radius data, locator reads, locatedness moduli, and metric-bracket rows around the inspected real datum.*
3. *C is the* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row, recording a finite located cover request through rational intervals, dyadic mesh data, refinement rows, finite windows, and readback data.*
4. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *positive lower-observation row. It records finite inhabited lower rational information that may witness nonempty contact from below.*
5. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *positive upper-observation row. It records finite upper-bound information that keeps the contact request located rather than host-open.*
6. *W is the finite witness row naming the particular cover cell, interval bracket, lower read, and upper read used by the positive-information claim.*
7. *A is the anti-escape ledger saying that the packet exports only witnessable contact and finite cover handoff, not arbitrary open-set membership or open-cover compactness.*
8. *H records componentwise* hsame *transport for $R, I, C, L, U, W, A, P, N$.*
9. *T records displayed* Cont *replay from the real seal to the located interval, from the interval to the cover request, and from the cover request to the lower and upper contact rows.*

10. *P is the* Pkg *provenance over* $\mathsf{OvertReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, I, C, L, U, W, A, H, T, P, N$. It has no coordinate for host open sets, detached membership predicates, arbitrary open covers, compactness extraction, trichotomy, selected representatives, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.OvertRealUp.OvertRealTasteGate single carrier alignment`

Theorem 3354.2 (OvertReal NameCert obligations)**.** *For every accepted* $\mathsf{OvertReal}^{\uparrow}$ *carrier*

$$O = (R, I, C, L, U, W, A, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$ *surface consists exactly of the finite positive-information obligations over the displayed packet:*

1. *real-source admission, requiring every real-facing read to pass through the terminal* $\mathsf{Real}^{\uparrow}$ *row R;*
2. *located-interval admission, requiring endpoint, radius, locator, locatedness, and metric-bracket requests to pass through I;*
3. *finite-cover handoff, requiring every positive contact claim to cite the displayed* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *row C;*
4. *one-sided observation exactness, requiring the lower and upper contact reads to be exactly L and U;*
5. *witness-row exactness, requiring the contact witness to be the finite row W and not a selected host point;*
6. *transport, replay, provenance, and local naming stability through H, T, P, N;*
7. *non-escape through A, excluding host open-set membership, arbitrary open-cover compactness, trichotomy, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 3354.1. The real datum is visible only through R, and the only interval information attached to that datum is the located row I.

The positive-information route next passes through C. Because C is a $\mathsf{LocatedIntervalCover}^{\uparrow}$ row, the cover request is finite, located, and routed through displayed rational and dyadic refinement data before any contact claim is available.

The one-sided rows L and U provide the observable lower and upper information. The witness row W binds these reads to one displayed cover cell and interval bracket, so the packet never replaces positive contact by a selected point or by detached host membership.

Finally H, T, P, N transport, replay, source, and name the same rows, while A records the forbidden exits. Since no other coordinate occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.OvertRealUp.OvertRealCarrier semantic name certificate`

Theorem 3354.3 (OvertReal positive-cover handoff)**.** *Every* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *consumer that asks whether an accepted* $\mathsf{OvertReal}^{\uparrow}$ *packet has positive contact with a displayed located interval factors through*

$$R, I, C, L, U, W, A, H, T, P, N.$$

The handoff exposes the source $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *bracket, the finite* $\mathsf{LocatedIntervalCover}^{\uparrow}$ *request, the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *one-sided contact rows, the finite witness row, and the structural transport, replay, provenance, and local naming rows. It exports no host open-set membership predicate, no open-cover compactness theorem, no selected point of contact, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with the accepted carrier O. The target consumer is cover-facing, so its first available cover row is the displayed coordinate C.

Reading C forces the consumer to retain the source interval row I and the terminal real seal R, because the cover route is attached to the located interval around the real datum rather than to an external open-set predicate.

The contact claim is then witnessed by L, U, W. The lower and upper rows provide the one-sided positive information, and W records the finite cover cell and bracket in which that information is consumed.

The rows A, H, T, P, N add no extra carrier coordinate. They only block escape, transport the displayed packet, replay the displayed route, record provenance, and name the local certificate. Thus every positive-cover handoff factors through exactly the stated finite surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$.

Constructive story: An $\mathsf{OvertReal}^{\uparrow}$ packet is generated from a terminal RealUp seal, a LocatedRealIntervalUp source interval, a LocatedIntervalCoverUp finite cover request, LowerRealUp and UpperRealUp one-sided contact rows, a finite witness row, an anti-escape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$, seed)

Scope closed: Definition 3354.1, Definition 3354.2, and Definition 3354.3 seed the finite positive-information surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OvertReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim classical open-set membership, arbitrary open-cover compactness, selected points of contact, trichotomy, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, real-source exactness, located-interval exactness, finite-cover handoff, one-sided contact exactness, witness-row exactness, transport stability, replay stability, and non-escape for BEDC.Derived.OvertRealUp.

Chapter 3355

A Concrete Naming Certificate for $\mathsf{CertificateMachine}^{\uparrow}$

The $\mathsf{CertificateMachine}^{\uparrow}$ packet realizes the certificate-machine branch of subsection 9462.28.23 as finite BEDC carrier data. It names the surface on which a machine-like object is admitted only through a finite kernel row, accepted-observation rows, branch-refusal rows, audit replay, provenance, and local NameCert evidence. The packet is not an ordinary Turing-machine identity claim, not a semantic truth oracle, not a host execution trace, and not a replacement for the kernel and NameCert obligations it records.

Definition 3355.1 (CertificateMachine carrier)**.** *A* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, O, B, A, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite certificate-kernel row. It lists the admitted constructors, candidate rows, acceptance stamps, and refusal slots visible to this machine surface.*
2. *O is the accepted-observation row, recording the finite observations the machine has admitted as certificate-bearing rows.*
3. *B is the branch-refusal row, recording rejected or unlicensed branches together with the local reason each branch cannot be read as accepted data.*
4. *A is the audit-replay row, recording the finite replay route from generated candidate through accepted observation or branch refusal.*
5. *R is the replay-result row, naming the public result that may be consumed after the audit replay finishes.*
6. *S is the soundness-boundary row saying that the machine exports certificate acceptance and refusal data, not a host truth predicate or an external operational semantics.*
7. *H records componentwise* hsame *transport for K, O, B, A, R, S, P, N.*
8. *C records displayed* Cont *replay from kernel row to observation or refusal row, from that row to audit replay, and from audit replay to the public result.*
9. *P is the* Pkg *provenance over* $\mathsf{CertificateMachine}^{\uparrow}$*, finite certificate kernels, accepted observations, branch refusals, audit replay,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, B, A, R, S, H, C, P, N$. It has no coordinate for host machine equality, ordinary Turing-machine identity, semantic truth, unbounded execution logs, unlicensed branch acceptance, or external proof authority.

Lean4 ✓ `BEDC.Derived.CertificateMachineUp.CertificateMachineTasteGate single carrier alignment`

Theorem 3355.2 (CertificateMachine NameCert obligations)**.** *For every accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier*

$$M = (K, O, B, A, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *kernel admission, requiring every candidate, constructor, acceptance stamp, and refusal slot to pass through K;*
2. *accepted-observation exactness, requiring every admitted observation to be one of the rows displayed in O;*
3. *branch-refusal exactness, requiring every rejected branch read to pass through B with its local refusal reason;*
4. *audit-replay determinacy, requiring every public result to be replayed through A before R is consumed;*
5. *soundness-boundary non-escape, requiring S to exclude semantic truth, ordinary Turing-machine identity, host execution logs, and external proof authority;*
6. *transport, continuation, provenance, and local naming stability through H, C, P, N.*

Lean follow-up must provide $\mathsf{Nontrivial}(\mathsf{CertificateMachine}^{\uparrow})$ *and* $\mathsf{FieldFaithful}(\mathsf{CertificateMachine}^{\uparrow})$ *instances before the object may be consumed as a nondegenerate field-facing certificate machine.*

Proof. Project the carrier of Definition 3355.1. The only kernel coordinate is K, so every candidate and acceptance stamp begins from that finite certificate-kernel row.

Accepted observations and refusals then split through O and B. The accepted side records only admitted observation rows, while the refusal side records branch-local reasons for rejection.

The audit route passes through A before the public result R is available. Thus a consumer cannot bypass replay by citing a host execution log or an unbounded operational trace.

Finally S, H, C, P, N bind the boundary, transport the same rows, replay the displayed route, record provenance, and name the local certificate. Since no coordinate adds semantic truth, ordinary Turing-machine identity, unlicensed branch acceptance, or external proof authority, the listed obligations exhaust the NameCert surface. □

Theorem 3355.3 (CertificateMachine audit-replay non-escape)**.** *Every public read of an accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *carrier factors through*

$$K, O, B, A, R, S, H, C, P, N.$$

An accepted observation reaches the public result only by the route K, O, A, R, and a rejected branch reaches the public refusal surface only by the route K, B, A, R. Both routes keep the soundness boundary S visible and preserve the same transport, continuation, provenance, and local naming rows. No read exports host machine equality, semantic truth, unbounded execution, unlicensed branch acceptance, or an external proof oracle.

Proof. Start from the displayed carrier M. There are exactly two observation-facing routes after the kernel row: the accepted-observation row O and the branch-refusal row B.

In the accepted case, continuation replay through C sends K and O into the audit row A before the public result R can be read. In the refusal case, the same replay discipline sends K and B into A before the refusal result is visible.

The soundness boundary S is carried with both routes. It excludes the stronger readings that would identify the packet with an ordinary Turing machine, a semantic truth oracle, a host execution log, or an external proof source.

The remaining rows H, P, N only transport, source, and name the packet. Hence every public read remains inside the displayed finite surface. □

Theorem 3355.4 (CertificateMachine branch-refusal determinacy)**.** *Let $M = (K, O, B, A, R, S, H, C, P, N)$ be an accepted* $\mathsf{CertificateMachine}^{\uparrow}$ *packet. Every refused branch of M is determined by the finite kernel row K, the branch-refusal row B, the audit-replay row A, and the soundness-boundary row S. Before the refused branch can be consumed as a MetaCIC audit row, its read must pass through the accepted-kernel surface of chapter 549, the axiom-query ledger of chapter 654, and the closed-generator audit surface of chapter 709.*

Proof. Project the carrier of Definition 3355.1. The refusal side has only the kernel row K, the branch-refusal row B, the audit replay row A, and the soundness boundary S before the public result row can be read.

Apply audit-replay non-escape, Definition 3355.3. A rejected branch reaches the public refusal surface only by the route K, B, A, R, with S still visible as the boundary against host truth, unbounded execution, and external proof authority.

The MetaCIC sibling lattice supplies the concrete audit route for those rows. Kernel admission is read through chapter 549, proof-supply cleanliness through chapter 654, and closed-generator replay through chapter 709.

Thus the refused branch is determined by the displayed finite rows of the same packet and by the named sibling ledgers. No separate kernel-only or host-execution route can consume it as a MetaCIC audit row. □

Falsifiable prediction: Every $\mathsf{CertificateMachine}^{\uparrow}$ read factors through one finite certificate-kernel row, one accepted-observation row or one branch-refusal row, one audit-replay row, one replay-result row, one soundness-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: kernelacceptancewitness, axiomqueryledger, inscriptionevent, observerfilter, closedgeneratoraudit: $\mathsf{CertificateMachine}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries kernel admission, accepted observations, branch refusals, audit replay, result exposure, and soundness-boundary refusal as one finite machine packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateMachine}^{\uparrow}$ packet is generated from a finite certificate-kernel row, accepted-observation rows, branch-refusal rows, audit replay, replay result exposure, a soundness-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3355.1, Definition 3355.2, and Definition 3355.3 seed the finite machine surface over certificate kernels, accepted observations, branch refusals, audit replay, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateMachine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ordinary Turing-machine identity, semantic truth, host execution semantics, unbounded trace replay, unlicensed branch acceptance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, kernel exactness, accepted-observation exactness, branch-refusal exactness, audit-replay determinacy, soundness-boundary non-escape, transport stability, replay stability, and the Nontrivial(CertificateMachine$^{\uparrow}$) and FieldFaithful(CertificateMachine$^{\uparrow}$) Lean instances for BEDC.Derived.CertificateMachineUp.

Chapter 3356

A Concrete Naming Certificate for GeneratingFunction$^{\uparrow}$

Following Wilf and Stanley, a GeneratingFunction$^{\uparrow}$ packet records a sequence by its coefficient stream and the finite algebraic windows through which coefficient extraction is read. It sits after the unary index discipline of chapter 28, the finite spine discipline of chapter 36, the stream-window interface of chapter 126, the real-series carrier of chapter 1581, and the coefficient-and-radius surface of chapter 1583. The packet is a traditional finite-combinatorics object in BEDC form: it exports coefficient readback and Cauchy-product handoff, not host analytic convergence or equality of functions.

Definition 3356.1 (GeneratingFunction carrier)**.** *A* GeneratingFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, Q, L, Pd, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the coefficient source row, indexed through* Nat$^{\uparrow}$ *and exposed only by finite* StreamName$^{\uparrow}$ *windows.*
2. *W is the coefficient-window ledger, listing the finite prefixes and requested coefficient indices visible to the packet.*
3. *Q is the coefficient readback row, turning each displayed window into a* RegSeqRat$^{\uparrow}$ *or finite combinatorial count row before any series consumer reads it.*
4. *L is the finite* List$^{\uparrow}$ *polynomial-prefix row, carrying the inspected initial polynomial truncations and their coefficient positions.*
5. *Pd is the coefficientwise classifier row for equality-facing reads of displayed prefixes.*
6. *M is the product handoff row, exposing the Cauchy convolution window used by* CauchySeriesProduct$^{\uparrow}$ *and* RealPowerSeries$^{\uparrow}$ *consumers.*
7. *H stores componentwise* hsame *transport for coefficient source, windows, readbacks, finite prefixes, classifier rows, product handoff, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from a coefficient request through the stream window, readback, polynomial prefix, classifier, and product-handoff rows.*
9. *P is the* Pkg *provenance over* GeneratingFunction$^{\uparrow}$*,* Nat$^{\uparrow}$*,* List$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* RealSeries$^{\uparrow}$*,* CauchySeriesProduct$^{\uparrow}$*,* RealPowerSeries$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^\uparrow}$ *row.*
The carrier contains no coordinate for a host analytic function, quotient series equality, selected convergence radius, countable-choice coefficient family, or ambient $\mathsf{Real}^\uparrow$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.GeneratingFunctionUp.GeneratingFunctionTasteGate single carrier alignment`

Theorem 3356.2 (GeneratingFunction NameCert obligations)**.** *For every accepted* $\mathsf{GeneratingFunction}^\uparrow$ *carrier* $K = (A, W, Q, L, Pd, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^\uparrow}$ *surface consists exactly of five finite rows: carrier admission for the displayed coefficient packet; coefficient-window readback through* A, W, Q, L*; coefficientwise classifier stability through* Pd, H, C*; Cauchy and product handoff through* M*; and ledger refusal for host functions, quotient series equality, analytic convergence, selected radii, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 3356.1. The accepted packet exposes only coefficient source, finite windows, readback rows, list-polynomial prefixes, classifier data, product handoff, transport, replay, provenance, and naming rows.

The coefficient-window readback is finite. A consumer first reads a $\mathsf{Nat}^\uparrow$ index request, then uses W to select a finite $\mathsf{StreamName}^\uparrow$ window, Q to read the coefficient data, and L to place the result in the displayed polynomial prefix.

Classifier stability is componentwise. The row Pd compares displayed prefixes only after the same windows and readbacks have been transported by H and replayed by C.

The Cauchy/product handoff is the row M. It exports exactly the finite convolution window needed by product and power-series consumers, so no host function equality, quotient series equality, analytic convergence theorem, selected radius, countable-choice family, or ambient completeness row enters the certificate. □

Theorem 3356.3 (GeneratingFunction coefficient readback)**.** *Every coefficient-facing consumer of an accepted* $\mathsf{GeneratingFunction}^\uparrow$ *packet factors through*

$$\mathsf{Nat}^\uparrow,\ A \longmapsto W \longmapsto Q, L \longmapsto Pd, H, C, P, N.$$

If the consumer asks for a product coefficient, the route additionally passes through the finite convolution ledger M *before any* $\mathsf{CauchySeriesProduct}^\uparrow$ *or* $\mathsf{RealPowerSeries}^\uparrow$ *handoff is visible. Thus the public readback is a finite coefficient-window certificate, not a host function evaluation, quotient equality of formal series, analytic convergence claim, selected infinite sum, or ambient* $\mathsf{Real}^\uparrow$ *completeness route.*

Proof. Begin with a displayed coefficient request. The only available index route is the $\mathsf{Nat}^\uparrow$ row named in the provenance, and the only available coefficient source is A.

The request then opens the selected finite window W. The readback row Q supplies the coefficient data, and the list-polynomial row L places the coefficient in a finite prefix rather than in an infinite host series.

For product-facing reads, use M after the same window and readback rows. The convolution comparison is therefore a displayed finite ledger handed to $\mathsf{CauchySeriesProduct}^\uparrow$ or $\mathsf{RealPowerSeries}^\uparrow$, not an equality of analytic functions.

The remaining rows Pd, H, C, P, N preserve and name this route. Since the carrier has no coordinate for host function evaluation, quotient series equality, analytic convergence, selected infinite sums, countable choice, or ambient completeness, every coefficient read is exhausted by the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{GeneratingFunction}^{\uparrow}$ packet is generated from a Nat-indexed coefficient source, finite StreamName coefficient windows, RegSeqRat or finite-count readbacks, List polynomial prefixes, a coefficientwise classifier, a Cauchy-product handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3356.1, Definition 3356.2, and Definition 3356.3 seed the finite coefficient-window certificate for generating functions and their product-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GeneratingFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host analytic functions, quotient series equality, analytic convergence, selected radii, selected infinite sums, countable choice, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window readback, coefficientwise classifier stability, Cauchy/product handoff, and ledger refusal for BEDC.Derived.GeneratingFunctionUp.

Chapter 3357

A Concrete Naming Certificate for $\mathsf{MetricIdentification}^{\uparrow}$

Following Bourbaki and Kelley on the identification of zero-distance points in a pseudometric space, the $\mathsf{MetricIdentification}^{\uparrow}$ packet records the finite BEDC route from displayed pseudometric distance rows to a separated metric-facing classifier. It sits between the pseudometric and metric rows of chapter 57, the zero-distance classifier boundary of chapter 1411, the completion consumer of chapter 1506, and the terminal real-distance seal of chapter 38. The packet names the identification surface without forming quotient primitives, host equality classes, selected representatives, or ambient completion points.

Definition 3357.1 (MetricIdentification carrier)**.** *A* $\mathsf{MetricIdentification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, M, Z, S, C, T, R, H, K, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *source row, exposing point histories, distance histories, and pseudometric comparison ledgers.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *target row whose distance reads are allowed only after the identification classifier has accepted the source comparison.*
3. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance row, recording the displayed real-distance seal used to compare source points at distance zero.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier handoff, requiring every equality-facing read to pass through the separated metric boundary.*
5. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer boundary, available only after P, M, Z, S have displayed the finite identification route.*
6. *T records componentwise* hsame *transport for source points, distance rows, zero-distance witnesses, separated classifier rows, completion handoff rows, provenance rows, and local naming.*
7. *R records displayed* Cont *replay from a finite pseudometric comparison request through the zero-distance row and separated classifier to the metric and completion consumers.*
8. *H records the zero-distance history ledger: every accepted comparison supplies the same* BHist *source pair, the displayed real-distance row Z, and the classifier row by which the pair becomes identified for S.*

9. *K is the* Pkg *provenance over* $\mathsf{MetricIdentification}^{\uparrow}$, $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Q is the quotient-refusal ledger, stating that no primitive quotient object, host equality class, selected representative, or uninspected completion point is a carrier coordinate.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose source pseudometric, target metric, zero-distance Real row, separated classifier handoff, completion boundary, transport, replay, zero-distance history ledger, provenance, refusal ledger, and local naming are displayed. It compares accepted packets through those rows and through no host quotient relation.

Lean4 ✓ `BEDC.Derived.MetricIdentificationUp.MetricIdentificationTasteGate single carrier alignment`

Theorem 3357.2 (MetricIdentification NameCert obligations)**.** *For every accepted* $\mathsf{MetricIdentification}^{\uparrow}$ *carrier $I = (P, M, Z, S, C, T, R, H, K, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$ *surface consists exactly of carrier admission for the pseudometric source, metric target, Real zero-distance row, separated metric classifier handoff, metric-completion consumer boundary, componentwise* hsame *transport, displayed* Cont *replay, zero-distance history ledger, package provenance, quotient-refusal ledger, and local naming. Every public identification read must pass through the zero-distance history row H and the separated classifier row S before any metric or completion consumer can read it.*

Proof. Project the accepted carrier of Definition 3357.1. The displayed analytic rows are exactly P, M, Z, S, C, H, and the structural rows are exactly T, R, K, Q, N.

The zero-distance comparison is therefore not a host equality assertion. It is the row H, read over the source pseudometric row P and the Real seal Z, then transported through T and replayed through R.

The separated metric row S is the first equality-facing consumer of that history. The metric row M and completion boundary C can only consume the result after S has displayed the classifier handoff. Since Q records the absence of quotient primitives, representatives, host equality classes, and uninspected completion points, the listed rows exhaust the local NameCert obligations. □

Theorem 3357.3 (MetricIdentification completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricIdentification}^{\uparrow}$ *packet reads zero-distance identification through*

$$\mathsf{Pseudometric}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{SeparatedMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{MetricCompletion}^{\uparrow}.$$

The handoff is a finite classifier route from source pseudometric distance rows to a separated metric boundary. It is not a quotient completion, not a selected representative scheme, and not host equality of carried points.

Proof. Begin with a completion-facing read. The carrier exposes the completion boundary only as the row C, so the request must first inspect the pseudometric source P, the Real zero-distance row Z, and the zero-distance history ledger H.

The separated classifier S then supplies the equality-facing boundary. Only after that boundary is displayed can the metric row M and completion boundary C be consumed by the downstream route.

The structural rows T, R, K, N preserve the same displayed packet under transport, continuation replay, provenance, and naming. The refusal row Q excludes quotient primitives and selected representatives, so the completion handoff is exactly the finite route shown above. □

Falsifiable prediction: Every accepted $\mathsf{MetricIdentification}^{\uparrow}$ completion-facing read must expose a $\mathsf{Pseudometric}^{\uparrow}$ source row, a $\mathsf{Real}^{\uparrow}$ zero-distance row, a $\mathsf{SeparatedMetric}^{\uparrow}$ classifier handoff, a $\mathsf{Met}^{\uparrow}$ target row, a $\mathsf{MetricCompletion}^{\uparrow}$ consumer boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, quotient-refusal data, and local naming.

Independence witness: pseudometric, metric, real, separatedmetric, metriccompletion: $\mathsf{MetricIdentification}^{\uparrow}$ is not bijective to any listed sibling because it binds the zero-distance Real row and separated classifier handoff before the metric and completion consumers may read an identified source comparison.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3357.1, Definition 3357.2, and Definition 3357.3 seed the zero-distance identification route from pseudometric rows to separated metric and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricIdentification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient construction, selected representatives, host equality of points, completion existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance ledger exactness, separated classifier handoff, completion consumer nonescape, transport and replay stability, provenance exhaustion, quotient-refusal faithfulness, and TasteGate field faithfulness for BEDC.Derived.MetricIdentificationUp.

Constructive story: A $\mathsf{MetricIdentification}^{\uparrow}$ packet is generated from a Pseudometric source row, a Metric target row, a Real zero-distance row, a SeparatedMetric classifier handoff, a MetricCompletion consumer boundary, componentwise hsame transport, displayed Cont replay, zero-distance history ledgers, Pkg provenance, quotient-refusal data, and a local NameCert row.

Chapter 3358

A Concrete Naming Certificate for EquicontinuityFiniteNetSelector$^{\uparrow}$

Following the Arzela–Ascoli finite-net reading in Bishop constructive analysis, the EquicontinuityFiniteNetSelector$^{\uparrow}$ packet records the finite selector surface that turns an equicontinuous family over a compact metric carrier into a displayed net-indexed modulus handoff. It sits over the family ledger of chapter 413, the compact source of chapter 129, the pointwise graph rows of chapter 130, the finite net surface of chapter 1767, and the uniform-modulus consumer of chapter 369. The packet selects from finite displayed rows only; it is not an open-cover compactness principle, not a host function-space topology, not a choice operator, and not a selector over external points.

Definition 3358.1 (Equicontinuity finite-net selector carrier)**.** *An* EquicontinuityFiniteNetSelector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, K, G, N, R, L, U, H, C, P, A)$$

with the following displayed rows:

1. *F is the* EquicontinuityFamily$^{\uparrow}$ *source row, exposing the shared family ledger and its pointwise tolerance-radius data.*
2. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing compact finite-net and located coverage data for the same source metric.*
3. *G is the* ContinuousMap$^{\uparrow}$ *graph row family whose pointwise modulus rows are inspected by the selector.*
4. *N is the* FiniteEpsilonNet$^{\uparrow}$ *row, carrying the displayed* ProbeBundle *of centers and the finite coverage ledger for the requested tolerance scale.*
5. *R is the positive rational radius ledger attached to the selected centers and family members.*
6. *L is the finite lower-bound row that compares the selected radii and exports the precision candidate to the modulus consumer.*
7. *U is the* UniformModulus$^{\uparrow}$ *handoff row, available only after F, K, G, N, R, L have displayed the finite selector route.*
8. *H records componentwise* hsame *transport for the family, compact metric, continuous-map, finite-net, radius, lower-bound, uniform-modulus, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact finite-net request through the family modulus rows and the lower-bound selector to the uniform-modulus consumer.*

10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *A is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose family, compact metric, continuous graph, finite net, radius ledger, lower-bound row, uniform-modulus handoff, transport, replay, provenance, and local naming are displayed in these coordinates. It has no coordinate for a host open cover, a global choice function, a function-space topology, an external finite subset, or a selected point outside the ProbeBundle *row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityFiniteNetSelectorUp.EquicontinuityFiniteNetSelectorTasteGate`
`single carrier alignment`

Theorem 3358.2 (Equicontinuity finite-net selector NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ *carrier* $E = (F, K, G, N, R, L, U, H, C, P, A)$*, the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetS}}$ *surface consists exactly of carrier admission for the equicontinuity-family row, compact metric row, continuous-map graph family, finite epsilon-net row, radius ledger, finite lower-bound row, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The selector is valid only when the finite-net centers and the family modulus rows share the same compact source metric before the uniform-modulus row is consumed.*

Proof. Project the accepted carrier of Definition 3358.1. The displayed source rows are F, K, G, N, the selector rows are R, L, U, and the structural rows are H, C, P, A.

The finite centers are read only from N. The family modulus information is read only from F and G, with K supplying the compact metric source used by the coverage ledger.

The lower-bound row L then compares the displayed radius rows R and exports a precision candidate to U. Transport through H and replay through C preserve the same row alignment, while P and A record provenance and local naming. No remaining coordinate can supply a host cover, a choice function, a function-space topology, or an external finite subset. □

Theorem 3358.3 (Equicontinuity finite-net selector uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ *packet reads the selector through*

$$\mathsf{EquicontinuityFamily}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{FiniteEpsilonNet}^{\uparrow},\quad \mathsf{UniformModulus}^{\uparrow}.$$

The exported precision is the finite lower-bound row over the displayed net centers and family radius ledger; it is not an externally chosen modulus and not an open-cover compactness theorem.

Proof. Start with the uniform-modulus request. The carrier exposes the consumer boundary only as U, so the request must first inspect the equicontinuity family F, the compact metric source K, the continuous graph family G, and the finite epsilon-net row N.

The radius ledger R is then read at the displayed net centers and family members. The lower-bound selector L is a finite row over those displayed radii, so the precision candidate handed to U remains inside the packet.

The transport, replay, provenance, and naming rows H, C, P, A preserve this finite route. Thus every consumer sees the selector as a compact finite-net and equicontinuity-family handoff, not as a host-level selector. □

Falsifiable prediction: Every accepted $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ uniform-modulus read must expose an $\mathsf{EquicontinuityFamily}^{\uparrow}$ source row, a $\mathsf{CompactMetric}^{\uparrow}$ compact source row, $\mathsf{ContinuousMap}^{\uparrow}$ graph rows, a $\mathsf{FiniteEpsilonNet}^{\uparrow}$ ProbeBundle coverage row, finite radius and lower-bound rows, a $\mathsf{UniformModulus}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: equicontinuityfamily, compactmetric, continuousmap, finite-epsilon-net, uniformmodulus: $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ is not bijective to any listed sibling because it binds the family modulus rows to a compact finite-net lower-bound selector before a UniformModulus consumer can read a tolerance-only precision row.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3358.1, Definition 3358.2, and Definition 3358.3 seed the finite selector route from equicontinuity-family and compact-net rows to a uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Arzela–Ascoli compactness, function-space topology, host finite subsets, choice of representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source lock, finite-net center visibility, family radius exactness, lower-bound selector stability, uniform-modulus handoff, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.EquicontinuityFiniteNetSelectorUp.

Constructive story: An $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ packet is generated from an EquicontinuityFamily source row, a CompactMetric source row, ContinuousMap graph rows, a FiniteEpsilonNet ProbeBundle row, finite radius and lower-bound rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3359

A Concrete Naming Certificate for $\mathsf{RealBolzano}^{\uparrow}$

Following Bolzano and Bishop–Bridges constructive analysis, the $\mathsf{RealBolzano}^{\uparrow}$ packet records sign-change bracketing as finite Real-and-Completion route data. It sits over the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the interval comparison surfaces of chapter 1591, chapter 411, and chapter 1786. The packet supplies a displayed bracket for later intermediate-value and root-isolation consumers without importing real trichotomy, unrestricted intermediate value, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3359.1 (RealBolzano carrier)**.** *A* $\mathsf{RealBolzano}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *a and b are endpoint real-name rows, each read through the* $\mathsf{Real}^{\uparrow}$ *seal route and exposed only as finite regular windows.*
2. *L and U are the left and right located sign witnesses, recording displayed apartness-from-zero evidence at the inspected endpoint windows.*
3. *S is the sign-change ledger, requiring the two endpoint sign witnesses to be opposed at the requested dyadic precision.*
4. *D is the dyadic bracket ledger, storing the interval endpoints, midpoint requests, tolerance radii, and width comparisons used by the finite bisection route.*
5. *J is the comparison row, carrying the ordered interval and nonescape checks that keep every retained bracket inside the displayed endpoint interval.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row assembled from the retained bracket windows and dyadic tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after the endpoint rows, located sign witnesses, bracket ledger, comparison row, and regular handoff have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint, sign, bracket, comparison, regular handoff, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite sign-change request through $a, b, L, U, S, D, J, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealBolzano}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *the closed bounded interval route, the regular Cauchy comparison route,* $\mathsf{RealRootBracket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $a, b, L, U, S, D, J, R, E, H, C, P, N$. It has no coordinate for real trichotomy, unrestricted intermediate value, a selected zero, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3359.2 (RealBolzano NameCert obligations)**.** *For every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *carrier*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for endpoint real rows, located sign witnesses, dyadic bracket ledger, comparison row, regular handoff, real seal boundary, transport, replay, provenance, and local naming;*
2. *endpoint sign exposure, requiring L and U to be read after the endpoint real-name rows a, b and to expose only located apartness data at displayed precision;*
3. *bracket-refinement stability, requiring each retained interval in D to preserve the displayed sign-change ledger S and to decrease the bracket width by dyadic comparison;*
4. *comparison nonescape, requiring J to keep every retained bracket inside the original displayed closed bounded interval;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff routing, requiring R to be assembled from the retained bracket windows and dyadic tolerance ledger rather than from a selected root;*
6. $\mathsf{Real}^{\uparrow}$ *seal routing, requiring E to remain downstream of a, b, L, U, S, D, J, R;*
7. *componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N for the same rows;*
8. *refusal of real trichotomy, unrestricted intermediate value, selected roots, quotient real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No RealBolzano obligation is stored outside these rows.

Proof. Project the carrier of Definition 3359.1. The endpoint rows a, b are the only real-name sources, and the located sign rows L, U expose sign information only at displayed precision.

The bracket route is finite. The ledger S records the sign-change request, while D stores the dyadic interval, midpoint, tolerance, and width-comparison data used by each retained bracket. The comparison row J keeps each retained interval inside the original closed bounded interval.

The completion-facing rows are downstream of that bracket route. The row R assembles the regular rational handoff from the retained bracket windows, and the seal E is read only after a, b, L, U, S, D, J, R have been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet rows. Since no excluded order, selection, quotient, or completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RealBolzanoUp.RealBolzanoTasteGate single carrier alignment`

Theorem 3359.3 (RealBolzano bracket seal route)**.** *Every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *sign-change read passes through endpoint real rows, located sign witnesses, the sign-change ledger, dyadic bracket ledger, comparison row, regular handoff, and terminal Real seal along the finite route*

$$a, b, L, U, S, D, J, R, E, H, C, P, N.$$

The route records bracket-to-seal data for later root-bracketing consumers, but it does not provide unrestricted intermediate value, a selected root, real trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3359.1. The endpoint rows a, b are the only real sources, and the located sign witnesses L, U are read from those endpoint windows at displayed precision.

Use Definition 3359.2. Its obligation rows place the sign-change ledger S before the dyadic bracket ledger D, keep each retained bracket inside the displayed interval through J, and assemble the regular handoff R from retained bracket windows rather than from a selected root.

The same obligation surface requires the Real seal E to remain downstream of a, b, L, U, S, D, J, R, with H, C, P, N transporting and naming only that finite packet. Since no excluded endpoint principle occurs in the carrier, the bracket-to-seal route exports only displayed sign-change and bracketing data. □

Theorem 3359.4 (RealBolzano bracket-refinement stability)**.** *Let*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealBolzano}^{\uparrow}$ *carrier. Replacing one retained dyadic bracket row in* D *by a displayed midpoint refinement that preserves the opposed located sign witnesses and passes the comparison row* J *yields another accepted* $\mathsf{RealBolzano}^{\uparrow}$ *packet with the same downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and* $\mathsf{Real}^{\uparrow}$ *seal route.*

Proof. Project the carrier of Definition 3359.1. The changed coordinate is a single retained bracket row inside D; the endpoint rows a, b, the located sign witnesses L, U, the sign-change ledger S, and the comparison row J remain displayed packet data.

Use Definition 3359.2. Its dependency order keeps S before D and J before R, so the midpoint replacement is admitted only when the opposed located sign witnesses are still present and the refined bracket still passes the displayed comparison row.

Transport the changed bracket row componentwise through H and replay the same finite request through C. The provenance row P and local naming row N continue to name the packet with the refined bracket ledger.

Finally apply Definition 3359.3. The regular handoff R is still assembled from the retained bracket windows, and the terminal Real seal E remains downstream of a, b, L, U, S, D, J, R. No selected root, real trichotomy, quotient equality, unrestricted intermediate value, or ambient completeness coordinate is introduced by the refinement. □

Theorem 3359.5 (RealBolzano regular-Cauchy bracket readback)**.** *Every accepted* $\mathsf{RealBolzano}^{\uparrow}$ *bracket-refinement read reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the endpoint* $\mathsf{Real}^{\uparrow}$ *rows, located sign witnesses, the sign-change ledger, dyadic bracket ledger, comparison nonescape row, and displayed bracket windows. The terminal* $\mathsf{Real}^{\uparrow}$ *seal remains downstream of that regular-Cauchy readback, and the packet exports no unrestricted intermediate value theorem, selected root, trichotomy, quotient equality, or ambient Real completeness.*

Proof. Start with the NameCert obligations of Definition 3359.2. They force endpoint sign exposure through a, b, L, U, record the sign-change ledger in S, and keep every retained bracket inside the displayed interval through the comparison row J.

Apply the bracket seal route of Definition 3359.3. It orders every sign-change read through $a, b, L, U, S, D, J, R, E, H, C, P, N$, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R is assembled before the terminal Real seal E can be consumed.

Now use bracket-refinement stability from Definition 3359.4. Replacing a retained bracket row changes only the displayed dyadic bracket data inside D while preserving the same sign witnesses, comparison nonescape row, regular handoff, and downstream seal route.

Thus the regular-Cauchy bracket readback reaches $\mathsf{RegSeqRat}^{\uparrow}$ through the finite endpoint, sign, ledger, comparison, and window rows, with E, H, C, P, N only sealing, transporting, replaying, sourcing, and naming the route. None of the excluded host principles occurs as a carrier coordinate. □

Theorem 3359.6 (RealBolzano sign-change bisection)**.** *Let*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

be an accepted $\mathsf{RealBolzano}^{\uparrow}$ *carrier. Any displayed midpoint refinement that keeps one opposed located sign pair and satisfies the comparison row J yields a retained bracket row whose* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff remains assembled from D and whose* $\mathsf{Real}^{\uparrow}$ *seal remains downstream of R. The bisection step does not export unrestricted intermediate value, a selected root, trichotomy, quotient real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 3359.1. The midpoint refinement changes only the retained bracket data inside D, while the endpoint rows, located sign witnesses, sign-change ledger, comparison row, regular handoff, and Real seal remain displayed coordinates of the same packet.

Apply Definition 3359.2. The obligations admit the refined bracket only when one opposed located sign pair is still displayed and the comparison row J keeps the retained interval inside the original bracket surface.

Use Definition 3359.4. It transports the refined bracket through H, C, P, N and preserves the downstream handoff and seal route. Then apply Definition 3359.5: the $\mathsf{RegSeqRat}^{\uparrow}$ row R is still assembled from the dyadic bracket ledger D, and the terminal $\mathsf{Real}^{\uparrow}$ seal E remains downstream of R.

Since the carrier has no unrestricted intermediate-value row, selected-root coordinate, trichotomy principle, quotient real equality, or ambient completeness theorem, the theorem records only the constructive finite bisection step visible to later root-bracketing consumers. □

Theorem 3359.7 (RealBolzano located-bracket nonescape)**.** *Every located sign-change bracket exported by an accepted* $\mathsf{RealBolzano}^{\uparrow}$ *carrier*

$$B = (a, b, L, U, S, D, J, R, E, H, C, P, N)$$

remains inside the endpoint rows a, b, located sign witnesses L, U, sign-change ledger S, dyadic bracket ledger D, comparison row J, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R,* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows H, C, P, N. The packet exports no unrestricted intermediate value theorem, selected root, trichotomy, quotient real equality, or ambient Real completeness.*

Proof. Use the NameCert obligations of Definition 3359.2. Endpoint sign exposure must pass through a, b, L, U, and comparison nonescape is recorded by the row J for every retained dyadic bracket in D.

Apply the bracket seal route of Definition 3359.3. It orders every sign-change read through $a, b, L, U, S, D, J, R, E, H, C, P, N$, so the regular readback and terminal Real seal remain downstream of the located bracket data.

Now use the sign-change bisection theorem Definition 3359.6. A retained midpoint bracket may be exported only when it preserves an opposed located sign pair and satisfies the comparison row J.

Thus each exported located bracket stays inside the displayed endpoint, sign, ledger, comparison, readback, seal, and structural rows. Since the carrier has no unrestricted intermediate-value, selected-root, trichotomy, quotient-equality, or ambient-completeness coordinate, none is exported. □

Falsifiable prediction: Every accepted $\mathsf{RealBolzano}^{\uparrow}$ read factors through two endpoint Real rows, two located sign witnesses, one sign-change ledger, one dyadic bracket ledger, one comparison row, one RegSeqRat handoff, one terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any intermediate-value or root-bracketing consumer may inspect it.

Independence witness: real, regseqrat, dyadicratcore, closedboundedinterval, regular-cauchy-comparison, real-root-bracket: $\mathsf{RealBolzano}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint real rows, located sign witnesses, a sign-change ledger, dyadic bracket refinement, comparison nonescape, regular handoff, real seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3359.1 and Definition 3359.2 seed the constructive sign-change bracketing packet over endpoint Real rows, located sign witnesses, dyadic bracket ledgers, comparison nonescape, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBolzano}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, unrestricted intermediate value, selected roots, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exposure, bracket-refinement stability, comparison nonescape, RegSeqRat handoff routing, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealBolzanoUp.

Constructive story: A $\mathsf{RealBolzano}^{\uparrow}$ packet is generated from endpoint Real rows, located sign witnesses, a sign-change ledger, dyadic bracket refinement, comparison rows, a RegSeqRat handoff, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3360

A Concrete Naming Certificate for $\mathsf{CofinalFilterBase}^{\uparrow}$

Following Kelley and Bourbaki on cofinal filter bases, the $\mathsf{CofinalFilterBase}^{\uparrow}$ packet records the finite comparison interface by which one displayed filter base may replace another in Cauchy and completion consumers. It sits between the displayed base surface of chapter 1601, the Cauchy-filter route of chapter 333, the uniform completion route of chapter 1473, the metric completion route of chapter 1506, and the Cauchy-filter completion route of chapter 1569. The packet names cofinal replacement without quotient filters, selected representatives, selected limits, ultrafilters, Zorn arguments, or ambient completeness.

Definition 3360.1 (CofinalFilterBase carrier)**.** *A* $\mathsf{CofinalFilterBase}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, T, \rho, R, F, U, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{FilterBase}^{\uparrow}$ *row whose displayed packets may be used after replacement.*
2. *T is the target* $\mathsf{FilterBase}^{\uparrow}$ *row whose requests are issued by a downstream consumer.*
3. *ρ is the cofinal map row. For every target request it names a source base packet lying inside the requested target packet.*
4. *R is the containment and refinement witness ledger, recording the source-to-target containment reads, common-refinement reads, and replay edges used by ρ.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, recording that the Cauchy-filter consumer reads the same cofinal replacement route.*
6. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer row, exposing the entourage-facing completion route after cofinal replacement.*
7. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row, exposing the metric-completion route after cofinal replacement.*
8. *H records componentwise* hsame *transport for $S, T, \rho, R, F, U, M, C, P, N$.*
9. *C records displayed* Cont *replay from a target base request through ρ, through the source base packet, and through the refinement ledger exposed to downstream consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CofinalFilterBase}^{\uparrow}$*, the five sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient filters, ultrafilters, Zorn arguments, selected representatives, selected limits, countable choice, host filter equality, or ambient completion.

Lean4 ✓ `BEDC.Derived.CofinalFilterBaseUp.CofinalFilterBaseTasteGate single carrier alignment`

Theorem 3360.2 (CofinalFilterBase NameCert obligations)**.** *For every accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *carrier* $B = (S, T, \rho, R, F, U, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$ *surface consists exactly of carrier admission for the source and target* $\mathsf{FilterBase}^{\uparrow}$ *rows, cofinal map row, containment and refinement ledger, CauchyFilter handoff, UniformCompletion consumer row, MetricCompletion consumer row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted target request must be witnessed by a displayed source packet through* ρ *and* R *before any Cauchy or completion consumer can read the replacement.*

Proof. Project the carrier of Definition 3360.1. The rows S and T expose the two displayed filter bases, while ρ and R store the only containment and refinement data connecting them.

The consumer rows are downstream of that comparison. The CauchyFilter row F, UniformCompletion row U, and MetricCompletion row M may read the replacement only after the target request has been replayed through ρ and the ledger R.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since quotient filters, ultrafilters, Zorn arguments, selected representatives, selected limits, countable choice, host filter equality, and ambient completion are absent from the carrier, no additional NameCert obligation is available. □

Theorem 3360.3 (CofinalFilterBase CauchyFilter handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *packet factors through*

$$T \longmapsto \rho \longmapsto S \longmapsto R \longmapsto F, H, C, P, N.$$

Thus a Cauchy-filter request against the target base is answered by a displayed source base packet and its containment ledger before the Cauchy row is read. The handoff supplies no quotient filter, ultrafilter, selected representative, selected limit, or choice schedule.

Proof. Begin with a Cauchy-filter consumer. The request is a target-base read through T, so the carrier forces the request to pass first through the cofinal map row ρ.

Apply Definition 3360.2. The row ρ names a source packet in S, and R records the containment and common-refinement reads that justify the replacement.

The handoff row F is read only after those rows. Transport by H and replay by C preserve the same displayed route, while P and N expose the packet as finite provenance and local naming. No excluded quotient, ultrafilter, selected representative, selected limit, or choice coordinate remains. □

Theorem 3360.4 (CofinalFilterBase completion nonescape)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer citing an accepted* $\mathsf{CofinalFilterBase}^{\uparrow}$ *packet receives only the displayed cofinal replacement route from target requests to source base packets. The completion-facing read is exhausted by* $S, T, \rho, R, F, U, M, H, C, P, N$ *and cannot escape to quotient filters, selected limits, host filter equality, ultrafilters, Zorn arguments, countable choice, or ambient completeness.*

Proof. Start from a completion-facing request. By Definition 3360.1, the only completion rows are U and M, and the Cauchy-filter completion consumer must pass through the same Cauchy handoff row F.

Apply Definition 3360.3. The target request is first replayed through ρ into the source base row S, then through the containment and refinement ledger R.

The sibling completion chapters named in the opening paragraph therefore receive a displayed cofinal-base route, not a detached completion principle. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and local naming, so no quotient, selected limit, host equality, ultrafilter, Zorn argument, choice schedule, or ambient completeness object is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalFilterBase}^{\uparrow}$ packet is generated from source and target FilterBase rows, a cofinal map from target requests to source base packets, containment and refinement witnesses, CauchyFilter handoff, UniformCompletion and MetricCompletion consumer rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3360.1, Definition 3360.2, Definition 3360.3, and Definition 3360.4 seed the cofinal-base replacement interface between FilterBase, CauchyFilter, and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalFilterBase}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host filter equality, ultrafilters, Zorn arguments, countable choice, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal map exactness, containment replay, CauchyFilter handoff, UniformCompletion and MetricCompletion consumer nonescape, and ledger-refusal proofs for BEDC.Derived.CofinalFilterBaseUp.

Chapter 3361

A Concrete Naming Certificate for $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$

Following Bishop and Bridges and the constructive bounded-variation treatment, the $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ packet records absolute continuity on a displayed real interval as finite partition control, variation budget, derivative/integral ledger, and real-name handoff data rather than as a host function quotient or a measure-completed derivative theorem. It sits after the regulated-function carrier of chapter 1906, the bounded-variation ledger of chapter 398, the integral surface of chapter 123, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The function-level carrier consumes the epsilon-delta finite partition surface of chapter 6066 before derivative, integral, bounded-variation, or Real-seal ledgers are read. The packet gives Real and Completion consumers a finite route for interval variation requests before any ambient measure algebra, selected derivative, host function equality, or completed L^1 object is available.

Definition 3361.1 (AbsoluteContinuousFunction carrier)**.** *An* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, F, M, P, V, D, L, T, W, Q, R, H, C, S, N)$$

with the following displayed rows:

1. *I is the source interval row, exposing the endpoint real-name windows, orientation data, and compact subinterval request on which the function is inspected.*
2. *F is the displayed real-valued function row, read only through finite input windows and output* $\mathsf{Real}^{\uparrow}$ *seals.*
3. *M is the modulus row assigning, for each dyadic error budget, a finite interval-length budget used by the absolute-continuity test.*
4. *P is the finite disjoint-partition ledger. It lists the inspected interval family, total displayed length, endpoint windows, and the corresponding value-jump windows.*
5. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *budget row asserting that the finite sum of visible jumps along P is controlled by the error budget selected by M.*
6. *D is the derivative ledger row. It records only displayed derivative-window requests and their local variation accounting.*

7. *L is the* $\mathsf{Integral}^{\uparrow}$ *handoff row for the interval derivative or variation-density request consumed by integration-facing clients.*
8. *T is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *approximation row used to expose stepwise endpoint behavior before the gauge row is read.*
9. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the interval, partition, derivative, integral, regulated approximation, and seal observations consumed by the packet.*
10. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for interval lengths, output jumps, regulated approximants, and integral estimates.*
11. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by downstream Real and Completion consumers after the partition, regulated, gauge, regular, and dyadic rows have been displayed.*
12. *H stores componentwise* hsame *transport for interval, function, modulus, partition, variation, derivative, integral, regulated, stream, regular readback, Real seal, replay, provenance, and local naming rows.*
13. *C stores displayed* Cont *replay from an absolute-continuity request through the same finite rows.*
14. *S is the* Pkg *provenance over* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, quotient real functions, measure completion, a selected derivative, countable additivity, ambient variation space completion, or a completed L^1 representative.

```
Lean4✓  BEDC.Derived.AbsolutelyContinuousFunctionUp.AbsolutelyContinuousFunctionTasteGate
single carrier alignment
```

Theorem 3361.2 (AbsoluteContinuousFunction NameCert obligations)**.** *For every accepted* AbsolutelyContinuousF *carrier $A = (I, F, M, P, V, D, L, T, W, Q, R, H, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$ *surface consists exactly of carrier admission; partition-refinement stability through P and W; variation-budget exactness through V; derivative/integral ledger handoff through D, L; regulated approximation through T; StreamName and RegSeqRat readback through W, Q; Real seal nonescape through R; componentwise transport, displayed replay, provenance, and local naming through H, C, S, N.*

Proof. Project the accepted carrier of Definition 3361.1. The visible rows are exactly $I, F, M, P, V, D, L, T, W, Q, R,$ so every local naming obligation must be read from one of those rows.

The source and control obligations are I, F, M, P, V. The interval and function rows determine the inspected endpoint windows, while the modulus row chooses the finite length budget. The partition and variation ledgers then record the only admissible test that small total displayed length forces small total displayed jump.

The analytic handoff obligations are D, L, T, W, Q, R. The derivative ledger records the local window request, the integral row carries the integration-facing handoff, the regulated row exposes stepwise endpoint behavior, and the stream and regular rows provide the finite numerical readbacks before the terminal Real seal is visible.

Finally H, C, S, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host function equality, quotient real functions, measure completion, a selected derivative, countable additivity, ambient variation space completion, or a completed L^1 representative, none of those objects can appear in the local NameCert surface. □

Theorem 3361.3 (AbsoluteContinuousFunction bounded-variation handoff)**.** *Every bounded-variation or fundamental-theorem consumer reads an accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packet only through the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$*,* $\mathsf{BoundedVariation}^{\uparrow}$*,* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *route*

$$I, F, M \longmapsto P, V \longmapsto D, L, T \longmapsto W, Q \longmapsto R, H, C, S, N.$$

Thus a consumer reaches the $\mathsf{Integral}^{\uparrow}$ *row only after the absolute-continuity partition budget and* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger have been displayed, and reaches the* $\mathsf{Real}^{\uparrow}$ *seal only after regulated approximation, StreamName windows, and regular rational readback have also been displayed.*

Proof. Start with the interval, function, and modulus rows. They fix the displayed real interval, the finite function windows, and the dyadic budget used by the absolute-continuity request.

The partition route is next. The row P lists the inspected disjoint intervals and their total displayed length, and V records the corresponding finite jump bound. No gauge-facing read is available before these rows have been consumed.

Only then does the route reach the derivative, integral, and regulated rows. The derivative ledger D records the local request, the $\mathsf{Integral}^{\uparrow}$ row L receives the integration-facing handoff, and the $\mathsf{RegulatedFunction}^{\uparrow}$ row T supplies the stepwise approximation boundary.

The sibling dependency is exact: the bounded-variation row is read through chapter 398, the regulated approximation through chapter 1906, the integral handoff through chapter 123, and the terminal numerical route through chapter 126, chapter 301, and chapter 38.

The terminal read is numerical and sealed. Rows W and Q provide finite observation windows and regular rational comparisons for the displayed estimates, while R, H, C, S, N seal, transport, replay, source, and name the packet. Hence the exported handoff is finite BEDC route data, not a host derivative theorem, measure-completed integral, quotient function equality, or selected L^1 representative. □

Theorem 3361.4 (AbsoluteContinuousFunction partition-modulus stability)**.** *Let* $A = (I, F, M, P, V, D, L, T, W, Q$ *and* $A' = (I', F', M, P', V', D', L', T', W', Q', R', H', C', S', N')$ *be accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packets sharing the same modulus row* M*. If their partition ledgers* P *and* P' *are related componentwise by* hsame *through the transport row* H*, then the variation-budget row* V *exports the same absolute-continuity jump bound before any derivative row, integral handoff, or Real seal is read.*

Proof. Project the two accepted carriers through Definition 3361.1. The shared row M fixes the dyadic length budget used by both absolute-continuity requests, while P and P' list the corresponding displayed disjoint interval families.

Use the componentwise hsame data in H to compare the partition ledgers. Since the comparison is rowwise over the displayed endpoint windows and total-length entries, the same partition budget is presented to the variation row on both sides.

Now read the variation-budget obligation from Definition 3361.2. The row V is upstream of D, L, T, W, Q, R, so the equality of jump bounds is obtained before derivative requests, integral handoffs, regulated approximations, numerical readbacks, or Real seals can contribute extra data. □

Theorem 3361.5 (AbsoluteContinuousFunction Real and Completion consumer boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *packet factors through the displayed rows*

$$M \longmapsto P \longmapsto V \longmapsto T \longmapsto W \longmapsto Q \longmapsto R$$

in that order before it exports an endpoint, variation, or integral-facing real read. Such a consumer cannot replace the route by host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, or ambient L^1 completion.

Proof. Begin with the control rows. The modulus row M chooses the finite length budget, the partition ledger P displays the inspected intervals, and the variation row V records the corresponding finite jump bound.

Next compose with the bounded-variation handoff of Definition 3361.3. It places the regulated approximation row T before the StreamName and RegSeqRat reads, so a Real or Completion consumer must pass through the displayed approximation boundary before any terminal numerical seal is available.

Finally read W, Q, R. The finite observation windows, regular rational comparisons, and Real seal are the only consumer-facing numerical route, and the carrier has no coordinate for host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, or ambient L^1 completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3361.1, Definition 3361.2, and Definition 3361.3 seed the finite absolute-continuity route over interval, function, modulus, partition, bounded-variation, derivative, integral, regulated, StreamName, regular, Real seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsolutelyContinuousFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient real functions, measure completion, selected derivatives, countable additivity, ambient variation space completion, completed L^1 representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition coverage, variation-budget exactness, derivative and integral handoff, regulated approximation, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.AbsoluteContinuousFunctionUp.

Constructive story: An $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ packet is generated from an interval row, displayed real-valued function row, modulus budget, disjoint-partition ledger, bounded-variation budget, derivative ledger, Integral handoff, RegulatedFunction approximation, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3362

A Concrete Naming Certificate for $\mathsf{EberleinSmulian}^{\uparrow}$

Following Eberlein and Smulian, the $\mathsf{EberleinSmulian}^{\uparrow}$ packet records the weak-compactness-to-sequential-window route as finite NameCert data. It sits beside the sequential compactness carrier of chapter 2372, the compact metric surface of chapter 129, the metric row of chapter 57, and the Cauchy sequence space boundary of chapter 406. The packet is a traditional functional-analysis reconstruction in the BEDC carrier language: it stores finite weak-test ledgers, subsequence-selection rows, compactness handoffs, and terminal naming evidence without asserting unrestricted weak compactness, countable choice, Banach-space dual completeness, or host-level subsequence selection.

Definition 3362.1 (Eberlein–Smulian finite weak-window carrier)**.** *An* $\mathsf{EberleinSmulian}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, H, K, M, Q, W, U, L, E, T, C, P, N)$$

with the following displayed rows:

1. *F is the functional-analysis source row naming the bounded linear-test context and the finite weak-test family inspected by the packet.*
2. *H is the Hilbert or Banach-facing source row that supplies the normed carrier, bounded sequence cells, and finite comparison windows.*
3. *K is the compactness handoff row, read through* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{SequentialCompact}^{\uparrow}$ *sibling surfaces when a finite compact window is displayed.*
4. *M is the metric row used to compare finite windows and to route Cauchy-style estimates through the sibling metric certificate.*
5. *Q is the Cauchy sequence equivalence boundary: two displayed subsequence windows are compared only through finite regular-window reads and not through quotient sequence equality.*
6. *W is the weak-test ledger listing the finitely many linear tests, tolerances, and observation windows requested by the consumer.*
7. *U is the subsequence-selection row recording the displayed indices retained at each finite weak-test window.*
8. *L is the compactness handoff ledger that repacks the retained subsequence windows as a compact or sequentially compact local packet.*

9. *E is the terminal weak-cluster or convergence seal offered to downstream consumers after the ledgers have been read.*
10. *T stores componentwise* hsame *transports for $F, H, K, M, Q, W, U, L, E$.*
11. *C stores the displayed* Cont *replay routes from weak tests to retained subsequence windows and from compactness handoff to the terminal seal.*
12. *P is the* Pkg *provenance over the source rows, weak-test ledger, subsequence rows, compactness handoff, transport, replay, and local naming.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only through these rows. It has no coordinate for a global weak topology, an unrestricted functional separator, a choice function selecting subsequences from arbitrary sequences, a quotient of sequence tails, or an ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.EberleinSmulianUp.EberleinSmulianTasteGate single carrier alignment`

Theorem 3362.2 (Eberlein–Smulian seed obligations)**.** *For every accepted* $\mathsf{EberleinSmulian}^{\uparrow}$ *carrier $S = (F, H, K, M, Q, W, U, L, E, T, C, P, N)$, the local NameCert surface consists of:*

1. *carrier admission for the finite weak-window packet of Definition 3362.1;*
2. *weak-test exposure through the finite ledger W over the displayed source rows F and H;*
3. *subsequence-window exposure through U, with retained indices visible before any cluster or convergence seal is exported;*
4. *compactness handoff through K and L, routed through the sibling compact and sequential compactness certificates rather than through open-cover compactness;*
5. *metric and Cauchy-window comparison through M and Q before any equality or convergence read is consumed;*
6. *componentwise transport, replay, provenance, and local naming through T, C, P, N.*

These obligations do not include Banach-space weak compactness in full generality, unrestricted linear-functional choice, countable choice, quotient sequence equality, or ambient completeness.

Proof. Project the carrier rows of Definition 3362.1. The rows $F, H, K, M, Q, W, U, L, E, T, C, P, N$ are the only data visible to the local classifier.

The weak-test side is finite. The ledger W lists the requested tests and tolerances, and the source rows F, H bind those tests to the displayed normed or Hilbert-facing carrier before a subsequence row is read.

The subsequence side is also finite. The row U records the retained indices for the inspected windows, while M and Q keep metric comparison and Cauchy-window comparison inside displayed rows.

The compactness handoff is downstream from those ledgers. The rows K and L repack the retained windows through the compact metric and sequential compactness sibling surfaces, and E is exposed only after T, C, P, N transport, replay, source, and name the same packet. Since the excluded principles are absent from the carrier, they cannot discharge a local NameCert obligation. □

Theorem 3362.3 (Eberlein–Smulian nonescape boundary)**.** *Every consumer of an accepted* $\mathsf{EberleinSmulian}^{\uparrow}$ *packet factors through the finite route*

$$W \longmapsto U \longmapsto (K, L) \longmapsto (M, Q) \longmapsto E$$

with componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$ *evidence. Thus a weak-cluster or sequential-compactness read cannot bypass the weak-test ledger, the retained subsequence windows, the compactness handoff, or the metric and Cauchy-window comparison rows.*

The boundary exports no unrestricted weak topology, no selected subsequence for arbitrary input streams, no quotient equality of sequence tails, no open-cover compactness theorem, and no ambient completeness principle.

Proof. Start with an accepted packet S. A consumer reaches the terminal seal E only by following the continuation row C, so its first readable data are the weak-test ledger W and the displayed source rows F, H.

The retained windows are then read through U. The indices are visible finite rows, and the comparison rows M, Q keep the resulting tail and equality checks inside displayed metric and Cauchy-window data.

Next read the compactness handoff. The rows K, L connect the retained windows to the compact metric and sequential compactness sibling certificates before any weak-cluster or convergence seal can be consumed.

Finally, T, C, P, N bind the route to one NameCert packet. Since the carrier contains none of the excluded global objects, the consumer has no escape route around the displayed finite ledgers. □

Closure status for $\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EberleinSmulian}^{\uparrow}$ packet is generated from displayed functional-analysis and Hilbert or Banach-facing source rows, finite weak-test ledgers, subsequence-selection windows, compactness handoff rows, metric and Cauchy-window comparisons, componentwise transport, continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$, seed)

Scope closed: Definition 3362.1, Definition 3362.2, and Definition 3362.3 bind the seed finite weak-window carrier and its nonescape boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EberleinSmulian}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not claim full Eberlein–Smulian equivalence, unrestricted weak compactness, Banach-space dual completeness, countable choice, quotient sequence equality, host-level subsequence selection, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-test ledger exposure, subsequence-window exposure, compactness handoff, metric and Cauchy-window comparison, transport replay, provenance, and local naming for the same finite packet.

Chapter 3363

A Concrete Naming Certificate for $\mathsf{SorgenfreyLine}^{\uparrow}$

Following Sorgenfrey and Kelley, the $\mathsf{SorgenfreyLine}^{\uparrow}$ packet records the lower-limit line as a finite naming certificate over the already named real and topology surfaces. It is a one-sided neighbourhood interface adjacent to the $\mathsf{Real}^{\uparrow}$ carrier of chapter 38, the local open-neighbourhood discipline of chapter 88, and the stream readback route of chapter 126. The packet also reads lower-limit interval witnesses through the located interval surface of chapter 1709 and regular rational approximation rows, but it does not identify the BEDC carrier with a host topology on the classical real line.

Definition 3363.1 (SorgenfreyLine carrier)**.** *A* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, I, W, A, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *source row for the point being observed.*
2. *I is a* $\mathsf{LocatedRealInterval}^{\uparrow}$ *lower-limit interval row displaying half-open windows $[a, b)$ around the source point through located endpoint, radius, and bracket data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that fixes the finite precision and finite list of lower-limit neighbourhood reads exposed to a consumer.*
4. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row repacking the rational endpoint observations of I and W.*
5. *L is the lower-limit topology ledger saying that neighbourhood access is by displayed half-open interval windows and not by an arbitrary host-open predicate.*
6. *H records componentwise* hsame *transports for R, I, W, A, L, C, P, N.*
7. *C records displayed* Cont *replay from the real source row to the located half-open interval row, from that row to the stream window, and from the window to the regular rational readback.*
8. *P is the* Pkg *provenance over* $\mathsf{SorgenfreyLine}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *row.*

The carrier stores no host topological space, no quotient real presentation, no selected point of a neighbourhood basis, no choice functional, and no completeness coordinate for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SorgenfreyLineUp.SorgenfreyLineTasteGate single carrier alignment`

Theorem 3363.2 (SorgenfreyLine NameCert obligations)**.** *For every accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier*

$$S = (R, I, W, A, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *obligations are exactly the finite lower-limit rows displayed by the carrier:*

1. *real-source admission, requiring every point read to pass through the* $\mathsf{Real}^{\uparrow}$ *row* R*;*
2. *half-open interval admission, requiring every neighbourhood read to expose the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row* I *as a lower-limit window* $[a, b)$*;*
3. *stream-window exactness, requiring the finite precision and finite neighbourhood inspection route to be the row* W*;*
4. *regular-rational readback, requiring endpoint and bracket observations to repack through* A *before any downstream real-facing consumer reads them;*
5. *topology-ledger exhaustion, requiring the topology-facing claim to use the ledger* L *rather than an ambient open-set predicate;*
6. *transport, replay, provenance, and local naming stability through* H, C, P, N*.*

These obligations do not assert host topology, quotient real equality, selected neighbourhood choices, countable choice, trichotomy, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier. The only point row is R, the only lower-limit interval row is I, and the only finite observation schedule is the stream window W.

The endpoint and bracket reads of the half-open window are then repacked by the regular rational row A. The ledger L records that topology-facing access is by displayed lower-limit windows, so a consumer cannot replace the route by an arbitrary host-open predicate.

The rows H, C, P, N only transport, replay, source, and name the displayed route. Since the carrier has no coordinate for quotient reals, selected neighbourhood choices, countable choice, trichotomy, host topology, or real completeness, none of those principles can be part of the local NameCert obligations. □

Theorem 3363.3 (SorgenfreyLine lower-limit window nonescape)**.** *Every accepted lower-limit neighbourhood read of an accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *carrier*

$$S = (R, I, W, A, L, H, C, P, N)$$

must pass through the $\mathsf{Real}^{\uparrow}$ *source row* R*, the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *half-open interval row* I*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* A*, the lower-limit topology ledger* L*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$ *row* N*. The read exports no host topology, quotient real equality, selected neighbourhood basis, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 3363.1. Its only point source is R, its only lower-limit neighbourhood row is the half-open $\mathsf{LocatedRealInterval}^{\uparrow}$ row I, and its only finite inspection schedule is the $\mathsf{StreamName}^{\uparrow}$ row W.

The endpoint observations of the half-open window then repack through A, and the topology-facing claim is exhausted by the ledger L. Apply Theorem 3363.2 to identify the remaining transport, replay, provenance, and local naming rows as H, C, P, N.

Because no other row occurs in the carrier, a lower-limit neighbourhood read cannot escape into host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3363.4 (SorgenfreyLine lower-limit topology handoff)**.** *Every* $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, *or product-topology consumer of an accepted* $\mathsf{SorgenfreyLine}^{\uparrow}$ *packet*

$$S = (R, I, W, A, L, H, C, P, N)$$

receives the lower-limit neighbourhood read only through $R, I, W, A, L, H, C, P, N$*. The row* I *provides the located half-open interval witness, and the row* L *provides the lower-limit topology ledger. The downstream* $\mathsf{SorgenfreyPlane}^{\uparrow}$ *product route may read this one-dimensional boundary, but it receives no host topology, quotient real equality, selected neighbourhood basis, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the accepted carrier of Definition 3363.1. The real source is R, the located half-open interval row is I, the finite window and rational readback rows are W and A, and the topology ledger is L.

Apply Definition 3363.2 and Definition 3363.3. Together they identify the structural rows H, C, P, N and force every lower-limit neighbourhood read to stay inside the displayed route.

The $\mathsf{SorgenfreyPlane}^{\uparrow}$ consumer therefore receives the line component only as this lower-limit topology handoff. Since no carrier row stores host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, or ambient real completeness, the product route cannot import those objects from the line certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SorgenfreyLine}^{\uparrow}$ packet is generated from a RealUp source row, a LocatedRealIntervalUp lower-limit half-open interval row, a StreamName finite window, a RegSeqRat readback row, a lower-limit topology ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3363.1 and Definition 3363.2 seed the lower-limit neighbourhood surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SorgenfreyLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topology, quotient real equality, selected neighbourhood bases, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, half-open interval admission, stream-window exactness, regular-rational readback, topology-ledger exhaustion, transport stability, replay stability, and non-escape for BEDC.Derived.SorgenfreyLineUp.

Chapter 3364

A Concrete Naming Certificate for $\mathsf{CauchyTailDiagonal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyTailDiagonal}^{\uparrow}$ packet records the finite diagonal-tail handoff by which two regular rational tails are read through one selected diagonal window before a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer inspects a seal. It sits between the diagonal extraction route of chapter 1773, the interleaving surface of chapter 414, the finite tail seal of chapter 559, the finite windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet is a finite handoff surface for Real and completion routes; it does not select an ambient limit, quotient stream, countable-choice schedule, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness.

Definition 3364.1 (CauchyTailDiagonal carrier)**.** *A* $\mathsf{CauchyTailDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, \delta, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *tail rows whose late observations are to be compared.*
2. *δ is the finite diagonal selector row, read through* $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$ *and synchronized with the* $\mathsf{RegularCauchyInterleaving}^{\uparrow}$ *handoff.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family naming the inspected late indices for X, Y, and the diagonal row δ.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the paired tail requests, diagonal slack, and finite comparison budget.*
5. *E is the* $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, Y, δ, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the two tail rows, diagonal selector, window family, dyadic ledger, seal row, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from paired tail requests through the selected diagonal window, dyadic tolerance ledger, finite tail seal, and terminal real handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyTailDiagonal}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}$ $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, Y, \delta, W, D, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable-choice selectors, host equality of real names, trichotomy, or ambient completion.

Lean4 ✓ `BEDC.Derived.CauchyTailDiagonalUp.CauchyTailDiagonalTasteGate single carrier alignment`

Theorem 3364.2 (CauchyTailDiagonal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailDiagonal}^\uparrow$ *carrier $K = (X, Y, \delta, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$ *surface consists exactly of carrier admission for the ten displayed rows, paired* $\mathsf{RegSeqRat}^\uparrow$ *tail admission through X, Y, diagonal selector exactness through δ, finite* $\mathsf{StreamName}^\uparrow$ *window coverage through W,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance accounting through D, finite tail seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3364.1. The only tail sources are X and Y, and the only diagonal selector is the row δ displayed by the packet.

Read the finite windows and tolerance ledger next. The row W names exactly the late observations inspected by the two tails and the diagonal selector, while D records the dyadic tolerances and slack needed by that finite comparison.

The seal row E is downstream of those rows. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the local certificate has no remaining coordinate from which a quotient stream, selected limit, countable-choice selector, host real equality, trichotomy row, or ambient completion principle could be exported. □

Theorem 3364.3 (CauchyTailDiagonal Real handoff)**.** *Every* $\mathsf{Real}^\uparrow$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailDiagonal}^\uparrow$ *carrier must read the finite route*

$$X, Y, \delta, W, D, E, H, C, P, N$$

before any Cauchy-tail conclusion is exported. The handoff first displays the two $\mathsf{RegSeqRat}^\uparrow$ *tails, then the diagonal selector governed by* $\mathsf{RegularCauchyDiagonalExtraction}^\uparrow$ *and* $\mathsf{RegularCauchyInterleaving}^\uparrow$*, then the selected* $\mathsf{StreamName}^\uparrow$ *windows and* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledger, and only then the* $\mathsf{FiniteTailDiagonalSeal}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *seal row.*

Proof. Begin with an accepted carrier and apply Definition 3364.2. The consumer-facing data are exhausted by the displayed packet $X, Y, \delta, W, D, E, H, C, P, N$.

Use the diagonal and interleaving coordinates in order. The selector row δ names the diagonal observations, and the interleaving handoff ensures that both tail rows are read on the same finite window family before the dyadic budget is consumed.

Finally read the finite tail seal. Since E is downstream of the paired tail rows, selected windows, and tolerance ledger, a Real or completion-facing consumer receives a finite diagonal-tail handoff rather than a quotient stream, selected limit, countable-choice schedule, host equality proof, trichotomy decision, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$.

Constructive story: A $\mathsf{CauchyTailDiagonal}^\uparrow$ packet is generated from two RegSeqRat tail rows, one finite diagonal selector, StreamName windows, a DyadicRatCore tolerance ledger, a FiniteTailDiagonalSeal and RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}^\uparrow}$, seed)

Scope closed: Definition 3364.1, Definition 3364.2, and Definition 3364.3 seed the finite diagonal-tail handoff over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteTailDiagonalSeal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailDiagonal}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host equality of real names, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal selector exactness, paired tail window coverage, dyadic tolerance accounting, finite tail seal handoff, Real handoff, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyTailDiagonalUp.

Chapter 3365

A Concrete Naming Certificate for SeparatedLimitUniqueness$^{\uparrow}$

Following Bishop and Bridges constructive metric-space analysis, the SeparatedLimitUniqueness$^{\uparrow}$ packet records the familiar uniqueness of metric limits in a separated metric space as finite BEDC certificate data. It sits over the separated classifier of chapter 1411, the complete-metric limit service of chapter 128, the Cauchy-sequence comparison route of chapter 1412, the terminal real seal of chapter 38, the regular rational readback surface of chapter 301, and the finite observation schedule of chapter 126. The packet names only the displayed zero-distance handoff between two limit rows; it is not a quotient completion equality, a selected representative theorem, countable choice, host equality of sequences, or an ambient completeness principle for Real$^{\uparrow}$.

Definition 3365.1 (SeparatedLimitUniqueness carrier)**.** *A* SeparatedLimitUniqueness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L_0, L_1, Q, M, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *L_0 and L_1 are* CompleteMetric$^{\uparrow}$ *limit rows for the same displayed Cauchy datum, each exposing only the finite limit-service surface available in the complete metric packet.*
2. *Q is the common* CauchySequenceEquivalence$^{\uparrow}$ *comparison route through which the two limit reads are compared before any endpoint classifier is inspected.*
3. *M is the* Met$^{\uparrow}$ *distance-witness row carrying the displayed distance comparison between the two candidate limits.*
4. *Z is the* Real$^{\uparrow}$ *zero-distance row, read only after the distance witness has passed through finite regular rational and stream-window readback.*
5. *S is the* SeparatedMetric$^{\uparrow}$ *classifier handoff saying that the zero-distance row is the only route by which the two displayed limit reads may be identified.*
6. *H records componentwise* hsame *transport for $L_0, L_1, Q, M, Z, S, C, P, N$.*
7. *C records displayed* Cont *replay from the two limit rows through the common Cauchy comparison, the metric distance witness, the zero-distance row, and the separated classifier.*
8. *P is the* Pkg *provenance over* CompleteMetric$^{\uparrow}$, CauchySequenceEquivalence$^{\uparrow}$, Met$^{\uparrow}$, Real$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L_0, L_1, Q, M, Z, S, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3365.2 (SeparatedLimitUniqueness NameCert obligations)**.** *For every accepted* SeparatedLimitUniquenes *carrier*

$$U = (L_0, L_1, Q, M, Z, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *two-limit admission, requiring both displayed limit reads to pass through the* $\mathsf{CompleteMetric}^{\uparrow}$ *rows L_0 and L_1;*
2. *common Cauchy comparison, requiring the two limit rows to be compared through the single* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *route Q;*
3. *distance-witness exactness, requiring the metric comparison to be the displayed* $\mathsf{Met}^{\uparrow}$ *distance row M;*
4. *zero-distance sealing, requiring the zero comparison to be the* $\mathsf{Real}^{\uparrow}$ *row Z reached after the finite readback route;*
5. *separated classifier handoff, requiring identification of the two limit reads to pass through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row S;*
6. *transport, replay, provenance, and local naming stability through H, C, P, N for the same rows;*
7. *refusal of quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No SeparatedLimitUniqueness obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SeparatedLimitUniquenessUp.SeparatedLimitUniquenessTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3365.1. The two candidate limits are visible only as the displayed complete-metric limit rows L_0 and L_1, so the certificate has no separate host endpoint from which to read equality.

The common comparison route is Q. It aligns the two limit reads as Cauchy data before the distance witness is inspected, keeping the route inside the finite $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ surface.

The metric comparison then passes through M and reaches the zero-distance row Z. The row Z is a terminal $\mathsf{Real}^{\uparrow}$ seal over the finite readback route, not a detached assertion of equality in a host real line.

Finally the separated classifier row S is consumed. Transport through H, replay through C, provenance through P, and local naming through N preserve the same displayed path, so the listed rows exhaust the NameCert obligations. □

Theorem 3365.3 (SeparatedLimitUniqueness zero-distance handoff)**.** *Every consumer that identifies two accepted* $\mathsf{CompleteMetric}^{\uparrow}$ *limit reads through an accepted* $\mathsf{SeparatedLimitUniqueness}^{\uparrow}$ *packet must factor the identification through*

$$L_0, L_1, Q, M, Z, S, H, C, P, N.$$

The handoff reads the common Cauchy comparison Q, the $\mathsf{Met}^{\uparrow}$ *distance witness M, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, and then the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier S before any named limit uniqueness claim is visible. It exports no quotient sequence equality, selected representative, host equality of metric points, countable-choice schedule, trichotomy split, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier U. A uniqueness-facing consumer first sees only the two complete-metric rows L_0 and L_1 and the shared comparison route Q.

The metric witness M is downstream of that route. Thus the consumer cannot compare the endpoints by host equality or by selecting representatives before the finite Cauchy comparison has been displayed.

The zero-distance row Z is then read as a $\mathsf{Real}^{\uparrow}$ seal, and S translates that displayed zero-distance evidence into the separated classifier handoff. The structural rows H, C, P, N transport, replay, source, and name the same route.

Therefore every limit-uniqueness read is the stated finite zero-distance handoff, with no additional quotient, choice, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3365.1, Definition 3365.2, and Definition 3365.3 seed the finite separated-limit uniqueness packet over complete-metric limit rows, Cauchy-sequence comparison, metric distance witnesses, Real zero-distance rows, separated classifiers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedLimitUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient completion equality, selected representatives, host sequence equality, countable choice, trichotomy, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common Cauchy comparison, distance-witness exactness, zero-distance sealing, separated classifier handoff, transport stability, replay stability, and non-escape for BEDC.Derived.SeparatedLimitUniquenessUp.

Constructive story: A $\mathsf{SeparatedLimitUniqueness}^{\uparrow}$ packet is generated from two CompleteMetricUp limit rows, a common CauchySequenceEquivalenceUp comparison route, a MetricUp distance witness, a RealUp zero-distance row, a SeparatedMetricUp classifier handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3366

A Concrete Naming Certificate for $\mathsf{CertificateMachineBranch}^{\uparrow}$

The $\mathsf{CertificateMachineBranch}^{\uparrow}$ packet realizes the branch-separation side of subsection 9462.28.23 as finite BEDC certificate data. It sits downstream of the finite certificate machine surface in chapter 3355, but it records a narrower packet: a kernel spine, an admission gate, a derivation-strength record, a cannot-claim ledger boundary, and explicit branch-separation rows for allegory and biological bridge candidates. The packet is not a plain Turing-machine identity claim, not a biological semantics claim, not a semantic truth oracle, and not an unbounded execution trace.

Definition 3366.1 (CertificateMachineBranch carrier)**.** *A* $\mathsf{CertificateMachineBranch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, A, D, L, G, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the finite kernel-spine row, naming the constructors, candidate rows, acceptance stamps, and refusal slots inherited from the BEDC certificate kernel.*
2. *A is the admission-gate row. It records the finite evidence by which a branch candidate is accepted, refused, or kept outside the public certificate surface.*
3. *D is the derivation-strength record, naming the declared strength at which the accepted branch may be read.*
4. *L is the cannot-claim boundary ledger, excluding semantic truth, plain Turing-machine identity, unbounded execution, biological identity, and external proof authority.*
5. *G is the allegory-branch row, recording candidate symbolic patterns only as pre-admission branch material.*
6. *R is the biological bridge row, recording DNA-facing or external natural branches only as bridge candidates, not as kernel primitives.*
7. *S is the branch-separation row saying that allegory, DNA, and certificate-kernel reads remain distinct unless the admission gate names a finite accepted route.*
8. *H records componentwise* hsame *transport for $K, A, D, L, G, R, S, C, P, N$.*
9. *C records displayed* Cont *replay from kernel spine through admission gate, derivation-strength record, cannot-claim boundary, and branch-separation rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CertificateMachineBranch}^{\uparrow}$, $\mathsf{CertificateMachine}^{\uparrow}$, *finite certificate kernels, admission gates, derivation-strength records, cannot-claim ledgers, allegory branches, biological bridge branches,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$ *row.*

The classifier compares accepted branch packets only through $K, A, D, L, G, R, S, H, C, P, N$. It has no coordinate for host machine equality, semantic truth, biological identity, unbounded execution logs, unlicensed allegory admission, or external proof authority.

Lean4 ✓ `BEDC.Derived.CertificateMachineBranchUp.TasteGate.CertificateMachineBranchTasteGate single carrier alignment`

Theorem 3366.2 (CertificateMachineBranch NameCert obligations)**.** *For every accepted* CertificateMachineBranch *carrier*

$$B = (K, A, D, L, G, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *kernel-spine admission, requiring every branch candidate, constructor, acceptance stamp, and refusal slot to pass through K;*
2. *admission-gate exactness, requiring every accepted branch read to cite the finite row A;*
3. *derivation-strength exactness, requiring the public strength of the branch read to be the displayed row D;*
4. *cannot-claim boundary non-escape, requiring L to exclude semantic truth, plain Turing-machine identity, unbounded execution, biological identity, and external proof authority;*
5. *allegory and biological branch separation, requiring G, R, S to keep symbolic-pattern and DNA-facing candidates distinct from kernel primitives unless admitted by A;*
6. *transport, continuation, provenance, and local naming stability through H, C, P, N.*

Lean TasteGate formalization must include explicit $\mathsf{Nontrivial}(\mathsf{CertificateMachineBranch}^{\uparrow})$ *and* $\mathsf{FieldFaithful}(\mathsf{Certifica}$ *instances before the packet may be consumed as a nondegenerate field-facing branch certificate.*

Proof. Project the carrier of Definition 3366.1. The branch candidate first meets the kernel-spine row K and the admission-gate row A, so no public branch read can arise from an unlicensed symbolic or biological source.

The derivation-strength row D and cannot-claim ledger L then determine the public reading strength and the forbidden exits. These rows keep certificate acceptance separate from semantic truth, host machine equality, unbounded execution, and biological identity.

The rows G, R, S provide the branch inventory. They distinguish allegory candidates from DNA-facing bridge candidates and from kernel primitives, and they expose no equality between those sources unless the admission gate records the finite route.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 3366.3 (CertificateMachineBranch separation non-escape)**.** *Every accepted public read of a* $\mathsf{CertificateMachineBranch}^{\uparrow}$ *packet factors through*

$$K, A, D, L, G, R, S, H, C, P, N.$$

The read first traverses the finite kernel spine K, the admission gate A, the derivation-strength record D, and the cannot-claim boundary L; only then may it inspect the allegory row G, biological bridge row R, and branch-separation row S. It exports no plain Turing-machine identity, biological semantics, host execution trace, unlicensed branch acceptance, or semantic truth predicate.

Proof. Begin with the accepted carrier B. The only kernel-facing row is K, and the only acceptance row is A, so a public read cannot skip directly to an allegory or DNA-facing branch.

The row D fixes the derivation strength carried by the accepted branch. The row L remains visible during the replay, so the consumer receives the stated boundary rather than a stronger semantic or biological claim.

The rows G, R, S are read after that boundary. They record branch inventory and separation, not an identification of allegory, biology, and kernel primitives.

The remaining rows H, C, P, N preserve the same finite route by transport, continuation replay, provenance, and local naming. Hence every accepted read stays inside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{CertificateMachineBranch}^{\uparrow}$ branch read factors through a finite kernel spine, an admission gate, a derivation-strength record, a cannot-claim boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before any branch-separation row may be consumed.

Independence witness: finitewitnessroute, gapclosureboundary, finitetraceevaluator, kernelacceptancetrace: $\mathsf{CertificateMachineBranch}^{\uparrow}$ is not bijective to any listed sibling because none contains the joint admission-gate plus allegory/DNA branch-separation inventory together with derivation strength, cannot-claim boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}$.

Constructive story: A $\mathsf{CertificateMachineBranch}^{\uparrow}$ packet is generated from a finite kernel spine, an admission gate, a derivation-strength record, a cannot-claim boundary, allegory and biological bridge branch rows, branch-separation evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3366.1, Definition 3366.2, and Definition 3366.3 seed the finite branch packet over certificate-machine kernels, admission gates, derivation-strength records, cannot-claim ledgers, allegory rows, biological bridge rows, branch separation, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertificateMachineBranch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim plain Turing-machine identity, semantic truth, biological semantics, host execution traces, unlicensed branch acceptance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked kernel-spine admission, admission-gate exactness, derivation-strength exactness, cannot-claim boundary non-escape, branch-separation exactness, transport stability, replay stability, and the $\mathsf{Nontrivial}(\mathsf{CertificateMachineBranch}^{\uparrow})$ and $\mathsf{FieldFaithful}(\mathsf{CertificateMachineBranch}^{\uparrow})$ Lean instances for BEDC.Derived.CertificateMac

Chapter 3367

A Concrete Naming Certificate for $\mathsf{SylvesterInertia}^{\uparrow}$

Following Sylvester's law of inertia and finite-dimensional quadratic-form theory, the $\mathsf{SylvesterInertia}^{\uparrow}$ packet records congruence-invariant positive, negative, and null sign-count rows. It reads the field source of chapter 45, the matrix presentation of chapter 49, the bilinear form row of chapter 146, and the inner-product substrate of chapter 86. The packet names finite inertia data as a displayed NameCert carrier; it does not assert analytic diagonalization, spectral theory, choice-selected bases, or completeness.

Definition 3367.1 (Sylvester inertia carrier)**.** *A* $\mathsf{SylvesterInertia}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, M, B, Q, D, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Field}^{\uparrow}$ *source row over which the scalar signs and matrix entries are read.*
2. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the finite square coordinate table used by the form.*
3. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *row naming the symmetric bilinear form carried by the matrix presentation.*
4. *Q is the quadratic-form row obtained by reading the diagonal values of B on displayed coordinate vectors.*
5. *D is the finite diagonalization-witness row, recording the congruence table that exposes diagonal form entries without choosing an ambient basis outside the packet.*
6. *S is the sign-count ledger, listing the displayed positive, negative, and null diagonal entries as the inertia triple.*
7. *H records componentwise* hsame *transport for the field source, matrix presentation, bilinear form, quadratic form, diagonal witness, sign-count ledger, continuation replay, provenance, and local naming.*
8. *C records* Cont *replay from a finite congruence request through F, M, B, Q, D into the sign-count ledger S.*
9. *P is the* Pkg *provenance over* $\mathsf{SylvesterInertia}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$ *row naming the displayed inertia packet.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for a spectral theorem, analytic diagonalization process, choice-selected basis, quotient form equality, or completeness principle.

Lean4 ✓ `BEDC.Derived.SylvesterInertiaUp.SylvesterInertiaTasteGate single carrier alignment`

Theorem 3367.2 (Sylvester inertia NameCert obligations)**.** *For every accepted* $\mathsf{SylvesterInertia}^{\uparrow}$ *carrier* $K = (F, M, B, Q, D, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the field source, matrix presentation, bilinear form, quadratic form, diagonal witness, sign-count ledger, transport, continuation replay, provenance, and local naming row;*
2. *form coherence, requiring* M*,* B*, and* Q *to describe the same displayed finite quadratic-form packet over* F*;*
3. *diagonal witness exactness, requiring* D *to expose only congruence steps whose source and target rows are recorded in* M, B, Q*;*
4. *inertia ledger readback, requiring* S *to count exactly the displayed positive, negative, and null diagonal entries exposed by* D*;*
5. *non-escape, excluding spectral decomposition, analytic diagonalization, basis choice outside the packet, quotient form equality, and completeness arguments.*

No local NameCert obligation is stored outside those five rows.

Proof. Project the carrier of Definition 3367.1. Carrier admission exposes exactly $F, M, B, Q, D, S, H, C, P, N$.

The form-coherence row binds the matrix table M, bilinear row B, and quadratic row Q over the same field source F. The diagonal witness row D is then readable only as a finite congruence ledger for that displayed form data.

The sign-count ledger S reads the diagonal entries exposed by D and separates them into positive, negative, and null cells. Finally H, C, P, N transport, replay, source, and name only that finite packet, so the excluded analytic and choice-based coordinates have no NameCert row. □

Theorem 3367.3 (Sylvester inertia congruence invariance)**.** *If two accepted* $\mathsf{SylvesterInertia}^{\uparrow}$ *packets are related by a displayed congruence replay through their rows* $F, M, B, Q, D, H, C, P, N$*, then they preserve the displayed sign-count ledger* S*. In particular, congruent accepted packets export the same positive, negative, and null inertia counts through* $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$*.*

Proof. Begin with two accepted carriers and a displayed congruence replay between them. By Definition 3367.2, the replay must pass through the field source, matrix presentation, bilinear form, quadratic form, and diagonal witness rows before the sign-count ledger is read.

The diagonal witness exactness row forces the congruence table to expose the same diagonal-form cells on both sides of the accepted comparison. The inertia ledger readback row then counts those displayed cells as positive, negative, or null inside S.

Because the carrier has no row for an external basis selector, analytic diagonalizer, quotient equality, or spectral argument, the exported comparison is exactly the componentwise ledger comparison. Thus the positive, negative, and null counts are preserved. □

Closure status for $\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SylvesterInertia}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3367.1, Definition 3367.2, and Definition 3367.3 seed the finite inertia carrier, five-row NameCert obligation surface, and congruence-invariant sign-count ledger.

Formal verification: FormalStatus(NameCert$_{\mathsf{SylvesterInertia}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the spectral theorem, analytic diagonalization, choice-selected bases, quotient form equality, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, form coherence, diagonal witness exactness, inertia ledger readback, and non-escape for the same finite Sylvester inertia packet.

Constructive story: A SylvesterInertia$^{\uparrow}$ packet is generated from a field source, matrix presentation, bilinear form, quadratic-form row, diagonalization witness, positive-negative-null sign-count ledger, componentwise transport, continuation replay, provenance, and local NameCert row.

Chapter 3368

A Concrete Naming Certificate for $\mathsf{FundamentalGroup}^{\uparrow}$

Following Hatcher and Spanier on the fundamental group, the $\mathsf{FundamentalGroup}^{\uparrow}$ packet names the loop-composition group at a displayed basepoint by finite BEDC rows. It sits beside the homotopy surface of chapter 156, the algebraic group surface of chapter 41, and the path-lifting surface of chapter 2275. The certificate records based loops, loop homotopies, concatenation, inverse, identity, and covering-lift handoff ledgers; it does not import quotient classes of loops or a host topological theorem.

Definition 3368.1 ($\mathsf{FundamentalGroup}^{\uparrow}$ finite carrier)**.** *A* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient topological source row and x_0 is the displayed basepoint row.*
2. *L is the finite based-loop row family. Each loop starts and ends at x_0 and is read through the path-facing part of* $\mathsf{Homotopy}^{\uparrow}$*.*
3. *Q is the loop-homotopy row family. It compares based loops by endpoint-fixed* $\mathsf{Homotopy}^{\uparrow}$ *certificates, not by an ambient quotient of maps.*
4. *M is the concatenation ledger. It records finite composable loop pairs, their concatenated loop row, and the endpoint-gluing readback.*
5. *I is the inverse ledger. It records the reversed loop row and the homotopy rows witnessing the two cancellation routes back to the identity loop row.*
6. *E is the identity loop row at x_0 and its left and right unit ledgers for M.*
7. *U is the covering-lift handoff row. It exposes exactly the based-loop data consumable by chapter 2275: base path, initial lift point request, lifted segment route, sheet classifier, and endpoint seal.*
8. *H records componentwise* hsame *transport for $X, x_0, L, Q, M, I, E, U, C, P, N$.*
9. *C records displayed* Cont *replay for loop readback, loop-homotopy readback, concatenation, inverse, identity, and covering-lift handoff rows.*

10. *P is the* Pkg *provenance over* $\mathsf{FundamentalGroup}^{\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{CoveringSpacePathLifting}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. Thus the group object is carried as a finite based-loop and homotopy ledger over BHist, *not as a quotient set of host loops.*

Lean4 ✓ `BEDC.Derived.FundamentalGroupUp.FundamentalGroupTasteGate single carrier alignment`

Theorem 3368.2 ($\mathsf{FundamentalGroup}^{\uparrow}$ NameCert obligations)**.** *For every accepted* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier*

$$\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$ *surface consists exactly of carrier admission, basepoint admission, based-loop admission through* $\mathsf{Homotopy}^{\uparrow}$*, endpoint-fixed loop-homotopy comparison, concatenation ledger exactness, inverse ledger exactness, identity-loop exactness, covering-lift handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{Group}^{\uparrow}$ *consumer can inspect $L, Q, M, I, E, H, C, P, N$ as the finite algebraic surface, while every covering consumer can inspect U as the finite handoff route; neither consumer receives quotient classes of loops.*

Proof. Carrier admission is projection from Definition 3368.1. The rows $X, x_0, L, Q, M, I, E, U, H, C, P, N$ already list the base space, basepoint, loops, homotopies, operations, transport, replay, provenance, and local naming data.

The group obligation is read from M, I, E over the classifier Q. Concatenation uses the finite endpoint-gluing row of M, inverse uses the reversed-loop row and cancellation ledgers of I, and the unit laws use E. Each comparison is routed through endpoint-fixed $\mathsf{Homotopy}^{\uparrow}$ rows, so the algebraic equality surface is the finite loop-homotopy classifier.

The covering obligation is independent of any global universal cover. The handoff row U exposes a based loop as the base path required by chapter 2275, together with exactly the initial lift request, lifted segment route, sheet classifier, and endpoint seal that a path-lifting packet can read.

Since H, C, P, N only transport, replay, source, and name the displayed rows, there is no remaining coordinate through which a downstream consumer could demand a host loop quotient, a completed path space, or an uncarried fundamental-group representation. □

Theorem 3368.3 ($\mathsf{FundamentalGroup}^{\uparrow}$ covering path route)**.** *Let* $\Pi = (X, x_0, L, Q, M, I, E, U, H, C, P, N)$ *be an accepted* $\mathsf{FundamentalGroup}^{\uparrow}$ *carrier. Any* $\mathsf{CoveringSpacePathLifting}^{\uparrow}$ *consumer of a based loop from* Π *factors through the handoff row U, the original based-loop row in L, the endpoint-fixed loop-homotopy classifier Q, the componentwise transport H, the displayed replay C, the provenance P, and the local name N. The consumer may observe the lifted endpoint seal and sheet-classifier route supplied by the covering packet, but it cannot replace the loop row by a quotient class or a fundamental-group representation.*

Proof. Choose an accepted based loop row from L. Its initial and final endpoint rows are both the displayed basepoint x_0, and its admissibility is already read through the $\mathsf{Homotopy}^{\uparrow}$ path surface.

Project the handoff row U. By definition, U contains the base-path presentation, the initial lift request, and the finite route slots consumed by $\mathsf{CoveringSpacePathLifting}^{\uparrow}$. The path-lifting packet therefore receives the loop only as a carried base path with visible endpoint data.

Transport along H and replay along C. These rows preserve the same based-loop row, loop-homotopy comparison, provenance, and local name, so any covering endpoint seal remains tied to the finite packet Π.

The route ends at the covering packet's own sheet classifier and endpoint seal. No step introduces a quotient of loops, a universal-cover action, or a host representation theorem; all handoff information is finite BHist data with NameCert provenance. □

Closure status for $\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3368.1, Definition 3368.2, and Definition 3368.3 seed the finite basepoint, loop, loop-homotopy, concatenation, inverse, identity, covering-handoff, transport, replay, provenance, and local naming surface for the fundamental group.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FundamentalGroup}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FundamentalGroupUp.FundamentalGroupTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close van Kampen theorems, monodromy representations, universal covers, quotient path spaces, host loop equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, based-loop admission, loop-homotopy comparison, operation ledgers, covering handoff, transport, replay, provenance, and local naming for BEDC.Derived.FundamentalGroupUp.

Constructive story: A $\mathsf{FundamentalGroup}^{\uparrow}$ packet is generated from a displayed basepoint, finite based-loop rows, endpoint-fixed loop-homotopy rows, concatenation, inverse, identity, covering-lift handoff, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 3369

A Concrete Naming Certificate for $\mathsf{PartitionOfUnity}^{\uparrow}$

Following Kelley and Bourbaki on partitions of unity, the $\mathsf{PartitionOfUnity}^{\uparrow}$ packet records the finite BEDC surface by which a displayed locally finite open cover receives compatible bump rows and finite-support weight rows. It sits over the topological source of chapter 88, the support-window discipline of chapter 1726, the refinement ledger of chapter 1963, the chart-facing consumer surface of chapter 96, and the separation-function route of **??**. The packet names only displayed finite rows for local-to-global gluing; it does not select arbitrary host functions, assert compactness of open covers, or identify analytic functions extensionally.

Definition 3369.1 (Partition of unity carrier)**.** *A* $\mathsf{PartitionOfUnity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (T, L, Q, R, W, F, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing the open-set operations and point-neighborhood reads on which all cover and support data are interpreted.*
2. *L is the* $\mathsf{LocallyCompact}^{\uparrow}$ *support-window row, recording the compactly supported local windows used by each bump row without exporting an open-cover compactness principle.*
3. *Q is the* $\mathsf{Paracompact}^{\uparrow}$ *locally finite cover ledger, including the displayed refinement cells, finite neighborhood witnesses, and assignment back to the original open-cover request.*
4. *R is the* $\mathsf{UrysohnLemma}^{\uparrow}$ *bump-row ledger, recording the separation-compatible construction of each nonnegative bump row inside the support window supplied by L and the locally finite cell supplied by Q.*
5. *W is the finite-support weight row, listing the finitely many bump rows visible at each displayed point-neighborhood read.*
6. *F is the displayed sum-to-one row, recording that the finitely supported weight rows add to the unit row on the covered region.*
7. *S is the manifold-facing handoff row, exposing exactly the partition rows consumable by* $\mathsf{Manifold}^{\uparrow}$ *chart gluing and by local-to-global analysis packets.*
8. *H records componentwise* hsame *transport for the topology source, support windows, locally finite cover ledger, bump rows, weight rows, sum row, manifold handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a partition request through $T, L, Q, R, W, F, S, H, P, N$.*
10. *P is the* Pkg *provenance over* PartitionOfUnity$^{\uparrow}$, Topology$^{\uparrow}$, LocallyCompact$^{\uparrow}$, Paracompact$^{\uparrow}$, UrysohnLemma$^{\uparrow}$, Manifold$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{PartitionOfUnity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-selected function family, arbitrary subcover selection, open-cover compactness theorem, unlogged chart choice, or extensional equality of analytic functions.

Lean4 ✓ `BEDC.Derived.PartitionOfUnityUp.PartitionOfUnityTasteGate single carrier alignment`

Theorem 3369.2 (Partition of unity NameCert obligations)**.** *For every accepted* PartitionOfUnity$^{\uparrow}$ *carrier*

$$U = (T, L, Q, R, W, F, S, H, C, P, N),$$

the local NameCert$_{\text{PartitionOfUnity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology source, support windows, locally finite cover ledger, bump-row ledger, finite-support weights, sum-to-one row, manifold handoff, transport, replay, provenance, and local naming;*
2. *support-window exactness, requiring every bump row to read its support through the displayed* LocallyCompact$^{\uparrow}$ *row L and not through a host compactness claim;*
3. *locally finite cover exactness, requiring every weight row in W to be indexed only by the refinement cells and finite neighborhood witnesses displayed by Q;*
4. *Urysohn-compatible bump exactness, requiring every bump row in R to pass through the* UrysohnLemma$^{\uparrow}$ *separation-function route before it can contribute to a weight row;*
5. *finite-support and sum exactness, requiring W and F to display the finite set of nonzero weights at each point-neighborhood read and the corresponding sum-to-one row;*
6. *manifold handoff exactness, requiring S to export only the rows $T, L, Q, R, W, F, H, C, P, N$ needed by* Manifold$^{\uparrow}$ *chart gluing and local-to-global consumers;*
7. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve exactly the listed rows.*

No local partition-of-unity obligation is stored outside these seven rows.

Proof. Carrier admission is projection from Definition 3369.1. The accepted tuple displays exactly the topology source, local compact support windows, paracompact locally finite ledger, Urysohn-compatible bump rows, finite-support weights, sum row, manifold handoff, transport, replay, provenance, and local naming.

Support-window exactness is read from L, while locally finite cover exactness is read from Q. The row L supplies only the compactly supported local windows already displayed in the packet, and Q supplies only the refinement cells, assignments, and finite neighborhood witnesses of the paracompact ledger.

The bump-row gate is R. Each bump row is admitted through the UrysohnLemma$^{\uparrow}$ separation-function route, so a weight row cannot be introduced by selecting an arbitrary host function.

The weight and sum gates are W and F. The row W lists the finite set of nonzero weights visible at each displayed neighborhood read, and F records the displayed sum-to-one row over that finite support.

Finally, S, H, C, P, N only hand off, transport, replay, source, and name the same finite packet. Thus a manifold or local-to-global consumer can read the partition surface without receiving a host-selected function family, arbitrary subcover, open-cover compactness theorem, unlogged chart choice, or extensional equality of analytic functions. □

Theorem 3369.3 (Partition of unity locally finite cover route)**.** *Let*

$$U = (T, L, Q, R, W, F, S, H, C, P, N)$$

be an accepted $\mathsf{PartitionOfUnity}^{\uparrow}$ *carrier. Every weight row in* W *and every sum-to-one row in* F *must first read the* $\mathsf{Paracompact}^{\uparrow}$ *locally finite cover ledger* Q *before reaching the* $\mathsf{UrysohnLemma}^{\uparrow}$ *bump rows* R*, the finite-support surface* W*, the sum row* F*, and the* $\mathsf{Manifold}^{\uparrow}$ *handoff* S*. The route does not select arbitrary subcovers or import open-cover compactness.*

Proof. Project the carrier of Definition 3369.1. The row Q is the only locally finite cover ledger in the packet, and it records the displayed refinement cells, finite neighbourhood witnesses, and assignment back to the original cover request.

Project the analytic rows R, W, F, S. The bump row R reads its separation and support data after the locally finite cell is known, the finite-support row W lists only the weights visible over that displayed locally finite neighbourhood, and the sum-to-one row F adds exactly those listed weights.

The handoff row S exports this same route to $\mathsf{Manifold}^{\uparrow}$ consumers. By Definition 3369.2, no local partition-of-unity obligation is stored outside the carrier rows, so neither arbitrary subcover selection nor open-cover compactness can replace Q. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3369.1 and Definition 3369.2 seed the finite topology, local compactness, paracompact refinement, Urysohn bump, weight, sum-to-one, manifold handoff, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionOfUnity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-selected function families, arbitrary subcover selection, open-cover compactness, analytic function extensionality, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-window exactness, locally finite cover exactness, Urysohn-compatible bump exactness, finite-support sum exactness, manifold handoff exactness, transport stability, replay stability, and non-escape proofs for BEDC.Derived.PartitionOfUnityUp.

Constructive story: A $\mathsf{PartitionOfUnity}^{\uparrow}$ packet is generated from a $\mathsf{Topology}^{\uparrow}$ source row, $\mathsf{LocallyCompact}^{\uparrow}$ support windows, a $\mathsf{Paracompact}^{\uparrow}$ locally finite cover ledger, $\mathsf{UrysohnLemma}^{\uparrow}$ bump rows, finite-support weights, a displayed sum-to-one row, a $\mathsf{Manifold}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3370

A Concrete Naming Certificate for $\mathsf{Bornology}^{\uparrow}$

Following Bourbaki and Kelley on bornologies, the $\mathsf{Bornology}^{\uparrow}$ packet records the finite bounded-family interface that sits between the bounded-subset certificate of chapter 1757, the directed base surface of chapter 1601, the uniform entourage surface of chapter 339, and the Cauchy-family specialization of chapter 1896. It names bounded families by displayed hereditary and finite-union rows before any completion, uniform, or Real-facing consumer can ask for a selected bound, a quotient family, countable choice, host suprema, or ambient completeness.

Definition 3370.1 (Bornology carrier)**.** *A* $\mathsf{Bornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, S, U, E, D, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded-family source row. It lists the finite family requests that may be read as bounded data.*
2. *S is the subset-heredity ledger, recording that an accepted subfamily request is read only through a displayed parent family already named by F.*
3. *U is the finite-union ledger, recording binary and finite union requests as displayed rows over previously accepted bounded families.*
4. *E is the singleton and source-exposure row, naming the basic bounded packets available before hereditary or union closure is inspected.*
5. *D is the consumer handoff row for* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *readers, and for the* $\mathsf{CauchyBornology}^{\uparrow}$ *specialization of the same bounded-family surface.*
6. *H records componentwise* hsame *transport for F, S, U, E, D, C, P, N.*
7. *C records displayed* Cont *replay from bounded-family admission through subset heredity, finite union, source exposure, consumer handoff, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{Bornology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyTasteGate single carrier alignment`

Theorem 3370.2 (Bornology NameCert obligations)**.** *For every accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier* $B = (F, S, U, E, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-family source exposure, subset-heredity ledger visibility, finite-union ledger visibility, singleton and source exposure,* $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *consumer handoff,* $\mathsf{CauchyBornology}^{\uparrow}$ *specialization handoff, componentwise transport, displayed continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyNameCertObligations`

Proof. Carrier admission projects the displayed packet. The source row F and singleton/source row E expose the basic bounded-family entries before any closure rule is read.

Subset heredity is confined to S, and finite union is confined to U. These ledgers are rows of the same packet, so a consumer cannot replace them by a host powerset, a selected bound, or a quotient family.

The handoff row D then presents the same bounded-family surface to $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, and $\mathsf{CauchyBornology}^{\uparrow}$ consumers. The structural rows H, C, P, N transport, replay, source, and name only the displayed packet. □

Theorem 3370.3 (Bornology bounded-family union stability)**.** *Let* $B = (F, S, U, E, D, H, C, P, N)$ *be an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier. Every accepted finite union of displayed bounded-family reads is certified by the finite-union ledger* U *after the source row* F *and singleton/source row* E *have been exposed, and every accepted subfamily of such a union is routed back through the subset-heredity ledger* S*. The route exports no host supremum, selected bound, quotient family, countable-choice selector, open-cover compactness principle, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyBoundedFamilyUnionStability`

Proof. Read the carrier rows. Source and singleton exposure are the only entry points for basic bounded-family data, so a union request must first cite displayed rows from F and E.

The finite-union ledger U records the admitted union row. A later subfamily request cannot inspect that union through a host family predicate; it must pass through S, whose parent row is already displayed inside the packet.

Transport and replay through H and C preserve the same ledgers, while P and N package only the named dependencies and local certificate. Therefore the union-stability route remains finite and contains none of the excluded host principles. □

Theorem 3370.4 (Bornology CauchyBornology handoff)**.** *Every* $\mathsf{CauchyBornology}^{\uparrow}$ *consumer of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier reads the general bounded-family surface through*

$$F, E \longmapsto S, U \longmapsto D \longmapsto H, C, P, N.$$

Thus the Cauchy specialization receives bounded-family source rows, singleton exposure, subset heredity, finite-union stability, consumer handoff, transport, replay, provenance, and local naming before it may open rate budgets, regular rational readback, Real seals, or completion-facing data.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyCauchyBornologyHandoff`

Proof. Start with a $\mathsf{CauchyBornology}^{\uparrow}$ consumer. The only general bornology data available to it are the displayed source and singleton rows F, E.

The closure properties are read next. Subset heredity is supplied by S, and finite union is supplied by U, so the consumer receives the ordinary bornological closure rows before any Cauchy-specific budget or Real-facing row can be inspected.

The handoff is then the row D, preserved by H and replayed by C. Since P and N name only the general bounded-family packet and its Cauchy specialization, the route adds no selected bound, quotient family, countable-choice selector, host supremum, or ambient completeness premise. □

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyCarrier cauchy bornology handoff`

Theorem 3370.5 (Bornology obligation closure route)**.** *Every accepted* $\mathsf{Bornology}^{\uparrow}$ *obligation read exposes exactly the route*

$$F, S, U, E, D, H, C, P, N.$$

The rows are the bounded-family source, subset-heredity ledger, finite-union ledger, singleton and source exposure, $\mathsf{FilterBase}^{\uparrow}$ *or* $\mathsf{UniformSpace}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$ *evidence. The route exports no host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyObligationClosureRoute`

Proof. The carrier of Definition 3370.1 supplies the visible rows $F, S, U, E, D, H, C, P, N$. The NameCert obligations of Definition 3370.2 identify those rows as carrier admission, bounded-family source exposure, heredity, finite union, source exposure, consumer handoff, transport, replay, provenance, and local naming.

The bounded-family union stability theorem Definition 3370.3 keeps union and subfamily reads inside U and S after the source rows F, E have been exposed. The CauchyBornology handoff of Definition 3370.4 routes every Cauchy consumer through the same displayed surface before any Cauchy-specific budget can be opened.

Together these results exhaust the finite obligation route. Since none of the carrier rows names host suprema, selected bounds, quotient families, countable choice, open-cover compactness, or ambient completeness, no accepted obligation read can export those omitted principles. □

Theorem 3370.6 (Bornology scoped bounded-family route)**.** *Every scoped bounded-family consumer of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier*

$$B = (F, S, U, E, D, H, C, P, N)$$

must read the route

$$F, S, U, E, D, H, C, P, N$$

through the displayed $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows before using the bounded-family surface. The route exports no host supremum, selected bound, quotient family, countable-choice selector, open-cover compactness principle, or ambient completeness premise.*

Proof. Start with the NameCert obligations of Definition 3370.2. They expose the bounded-family source row F, the subset-heredity ledger S, the finite-union ledger U, the singleton/source row E, the consumer handoff row D, and the structural rows H, C, P, N as the complete local surface.

Use bounded-family union stability, Definition 3370.3. It keeps every finite union and subfamily read inside U and S after F, E have been displayed, so the bounded-family consumer cannot replace the packet by a host family or selected bound.

Next apply the CauchyBornology handoff of Definition 3370.4. It forces completion-facing bounded-family reads through the same F, E, D, H, C, P, N surface before any Cauchy-specific budget is opened.

Finally compose with the obligation closure route Definition 3370.5. The scoped consumer therefore reads exactly the displayed carrier route through $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with no additional supremum, choice, quotient, compactness, or completeness coordinate. □

Theorem 3370.7 (Bornology scope kernel grounding)**.** *The bounded-family source, subset-heredity, finite-union, consumer handoff, transport, replay, provenance, and local NameCert rows of an accepted* $\mathsf{Bornology}^{\uparrow}$ *carrier are scoped exactly over* $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Every scoped consumer must read the displayed route*

$$F, S, U, E, D, H, C, P, N$$

through those kernel and sibling rows, and the route adds no host supremum, selected bound, quotient family, countable-choice selector, compactness principle, or ambient completeness premise.

Lean4 ✓ `BEDC.Derived.BornologyUp.BornologyScopeKernelGrounding`

Proof. Start with the NameCert obligations of Definition 3370.2. They expose carrier admission, bounded-family source rows, subset heredity, finite union, singleton/source exposure, consumer handoff, componentwise transport, displayed replay, provenance, and local naming as the local $\mathsf{Bornology}^{\uparrow}$ surface.

Use bounded-family union stability, Definition 3370.3. It keeps heredity and finite-union reads inside the displayed S and U ledgers after F and E have been exposed, so a scoped consumer cannot replace the packet by host suprema, selected bounds, quotient families, or choice data.

Apply the CauchyBornology handoff of Definition 3370.4. The handoff row D presents the same bounded-family surface to $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, and $\mathsf{CauchyBornology}^{\uparrow}$ consumers before any Cauchy-specific or completion-facing data are opened.

Finally compose the obligation closure route Definition 3370.5 with the scoped bounded-family route Definition 3370.6. The resulting kernel grounding is exactly the route through $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert, with no extra compactness or completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3370.1, Definition 3370.2, Definition 3370.3, Definition 3370.4, Definition 3370.5, Definition 3370.6, and Definition 3370.7 bind the finite bounded-family NameCert surface over bounded-set, filter-base, uniform-space, and Cauchy-bornology consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Bornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host suprema, selected bounds, quotient families, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Public closure asks for a public bounded-family export theorem and later Lean verification for BEDC.Derived.BornologyUp.

Constructive story: A Bornology$^{\uparrow}$ packet is generated from bounded-family source rows, subset-heredity ledgers, finite-union ledgers, singleton and source exposure, FilterBaseUp or UniformSpaceUp handoff rows, CauchyBornologyUp specialization, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3371

A Concrete Naming Certificate for $\mathsf{MetricAnnulus}^{\uparrow}$

Following Kelley on metric neighbourhoods and Bishop–Bridges constructive metric spaces, the $\mathsf{MetricAnnulus}^{\uparrow}$ packet records the shell between an inner exclusion radius and an outer containment radius as finite BEDC certificate data. It sits over the metric source of chapter 57, the metric-ball surface of chapter 1450, the open-ball exclusion route of chapter 2191, the closed-ball containment route of chapter 1451, and the real-name radius seal of chapter 38. The packet is a metric shell neighbourhood certificate, not a host subset, not quotient point equality, not a selected-radius principle, not compactness, and not $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3371.1 (MetricAnnulus carrier)**.** *A* $\mathsf{MetricAnnulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, c, r_0, r_1, O, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row naming the finite distance-comparison surface.*
2. *c is the centre row, read only through the same metric source.*
3. *r_0 is the inner radius row, exposed as a positive or nonnegative* $\mathsf{Real}^{\uparrow}$ *radius seal according to the local metric-ball convention.*
4. *r_1 is the outer radius row, exposed through the same* $\mathsf{Real}^{\uparrow}$ *radius discipline.*
5. *O is the* $\mathsf{OpenBall}^{\uparrow}$ *exclusion row, recording that points in the shell are outside the displayed inner open ball around c with radius r_0.*
6. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *containment row, recording that points in the shell remain inside the displayed closed ball around c with radius r_1.*
7. *H records componentwise* hsame *transport for the metric source, centre, two radius rows, open-ball exclusion row, closed-ball containment row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a shell-membership request through the radius-order check, the open-ball exclusion test, and the closed-ball containment test.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{OpenBall}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$ *row.*

The carrier accepts only packets whose metric source, centre, inner radius, outer radius, exclusion row, containment row, transport, replay, provenance, and local name are all displayed in the same finite ledger.

Theorem 3371.2 (MetricAnnulus NameCert obligations)**.** *For every accepted* $\mathsf{MetricAnnulus}^{\uparrow}$ *carrier* $K = (M, c, r_0, r_1, O, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source and centre, visibility of the ordered inner and outer radius rows, inner-boundary exclusion through the* $\mathsf{OpenBall}^{\uparrow}$ *row, outer-boundary containment through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row, classifier transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every shell-neighbourhood read is confined to those rows and cannot open a host subset, quotient point equality, selected-radius choice, compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted packet to the displayed rows $M, c, r_0, r_1, O, B, H, C, P, N$. The metric source and centre are the only source coordinates, and the two radius rows are the only numerical boundary coordinates available to a shell read.

The inner boundary is read through O: a point must avoid the displayed open ball around c at radius r_0. The outer boundary is read through B: the same point remains inside the displayed metric closed ball around c at radius r_1.

The transport row H and replay row C keep the classifier and continuation route componentwise on this packet. The provenance and local name rows P, N then certify that every later consumer reaches the shell only through the metric source, centre, radius rows, exclusion row, and containment row already displayed.

Since the carrier has no field for a host subset, quotient equality on points, a selected-radius operator, compactness data, or a completeness theorem for $\mathsf{Real}^{\uparrow}$, the ledger cannot escape to those principles. □

Lean4 ✓ `BEDC.Derived.MetricAnnulusUp.MetricAnnulusCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricAnnulus}^{\uparrow}$ packet is generated from a Metric source row, a centre row, inner and outer Real radius rows, an OpenBall exclusion row, a MetricClosedBall containment row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$, seed)

Scope closed: Definition 3371.1 and Definition 3371.2 seed the finite metric shell certificate for displayed inner exclusion and outer containment reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricAnnulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host subsets, quotient point equality, selected radii, compactness, theorem-level Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, radius-order visibility, inner-boundary exclusion, outer-boundary containment, classifier transport, Cont replay, Pkg provenance, and ledger nonescape for BEDC.Derived.MetricAnnulusUp.

Chapter 3372

A Concrete Naming Certificate for $\mathsf{RealArchimedeanProperty}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArchimedeanProperty}^{\uparrow}$ packet records the finite BEDC surface by which a displayed $\mathsf{Real}^{\uparrow}$ name receives a natural-number bound and the corresponding rational and dyadic comparison rows. It sits next to the $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ neighbour of chapter 2201, but it names the property-level handoff: every bound read starts from the $\mathsf{Real}^{\uparrow}$ name of chapter 38, passes through $\mathsf{Nat}^{\uparrow}$ of chapter 28, $\mathsf{Rat}^{\uparrow}$ of chapter 37, $\mathsf{DyadicRatCore}^{\uparrow}$ of chapter 291, $\mathsf{StreamName}^{\uparrow}$ of chapter 126, and $\mathsf{RegSeqRat}^{\uparrow}$ of chapter 301, and only then exposes an Archimedean bound ledger. The packet is a finite naming certificate for that route, not a host real-order axiom, trichotomy principle, quotient-real equality, or completeness theorem.

Definition 3372.1 (RealArchimedeanProperty carrier)**.** *A* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, n, q, d, S, G, B, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name whose finite real observations are being bounded.*
2. *n is the* $\mathsf{Nat}^{\uparrow}$ *bound row, recording the natural-number index used by the bound.*
3. *q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row naming the rational face of the same bound.*
4. *d is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row naming the dyadic face of the same bound.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected before the comparison is read.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected stream window into a regular rational Cauchy observation.*
7. *B is the Archimedean bound ledger, compatible with the* $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ *route of chapter 2201.*
8. *H records componentwise* hsame *transport for the real name, natural bound, rational row, dyadic row, stream window, readback row, bound ledger, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a bound request through S, G, q, d, and n.*
10. *P is the* Pkg *provenance over* $\mathsf{RealArchimedeanProperty}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$ *row.*
The carrier accepts only packets whose real observation, bound index, rational and dyadic comparisons, stream inspection, regular rational readback, bound ledger, transport, replay, provenance, and local name are all displayed in the same finite certificate.

Theorem 3372.2 (RealArchimedeanProperty NameCert obligations)**.** *For every accepted* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *carrier* $A = (R, n, q, d, S, G, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Real}^{\uparrow}$ *name,* $\mathsf{Nat}^{\uparrow}$ *bound row,* $\mathsf{Rat}^{\uparrow}$ *comparison row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, Archimedean bound ledger, componentwise transport through H, displayed continuation replay through C,* Pkg *provenance through P, and local naming through N. No obligation may read a host real-order axiom, trichotomy oracle, quotient-real equality, selected unbounded search, or ambient completeness theorem.*

Proof. Project the accepted carrier. The rows $R, n, q, d, S, G, B, H, C, P, N$ are the only coordinates available to the local certificate, so carrier admission and local naming are finite ledger facts.

The observation route begins with R and the inspected stream window S. The readback row G turns that finite window into the rational Cauchy surface already accepted by $\mathsf{RegSeqRat}^{\uparrow}$.

The bound route then reads q, d, and n as the rational, dyadic, and natural faces of the same displayed bound ledger B. The transport and replay rows preserve exactly these coordinates, and the provenance row names the sibling chapters from which they are read.

Since the carrier has no coordinate for host order, trichotomy, quotient equality, an unbounded search principle, or completeness, none of those principles can enter the local $\mathsf{NameCert}$ obligations. □

Lean4 ✓ `BEDC.Derived.RealArchimedeanPropertyUp.RealArchimedeanPropertyTasteGate single carrier alignment`

Theorem 3372.3 (RealArchimedeanProperty natural-rational bound handoff)**.** *Every order-facing or approximation-facing consumer of an accepted* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *packet reaches the natural bound row only through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Nat}^{\uparrow}.$$

The rational and dyadic comparison rows are read before the natural-number bound is exported, and the same handoff is compatible with the $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ *ledger of chapter 2201.*

Proof. Begin with a consumer requesting a bound for the displayed real name. The continuation row C first routes the request through the finite $\mathsf{StreamName}^{\uparrow}$ window S and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row G.

The comparison rows q and d are then read as rational and dyadic witnesses for the same inspected real observation. Only after these rows are displayed does the natural-number row n become available as the bound face of the ledger B.

The natural bound row is consumed by finite unary induction over the displayed bound ledger. The base step reads the initial ledger face, and each successor step reuses the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, rational comparison, and $\mathsf{DyadicRatCore}^{\uparrow}$ comparison rows. The induction exports only the displayed Nat/Rat/Dyadic/Real route; it does not introduce unbounded search, trichotomy, host order, quotient equality, or completeness.

The compatibility with $\mathsf{RealArchimedeanBoundSchema}^{\uparrow}$ is structural: both packets expose the same real, stream, regular readback, rational, dyadic, and bound-ledger coordinates. The property packet adds the explicit $\mathsf{Nat}^{\uparrow}$ handoff without adding host order or completeness data. □

Theorem 3372.4 (RealArchimedeanProperty ledger nonescape)**.** *For an accepted* $\mathsf{RealArchimedeanProperty}^{\uparrow}$ *packet, every Archimedean-property read factors through the displayed rows* $R, n, q, d, S, G, B, H, C, P, N$*. In particular, the packet exports no hidden real supremum, no total order trichotomy, no quotient equality of real names, no unbounded search over naturals, no selected least bound, and no completeness principle for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Inspect the carrier inventory. The only real-facing row is R, the only finite observation rows are S and G, the only comparison rows are q and d, and the only natural bound row is n.

The replay row C can only traverse those displayed coordinates, while H transports them componentwise and P, N package the same route as provenance and local naming. Thus any Archimedean-property read must pass through the finite ledger described in the carrier.

Because no row stores a supremum, trichotomy decision, quotient equality test, unbounded natural search, selected least-bound operator, or completeness theorem, the packet cannot export any of those structures. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealArchimedeanProperty}^{\uparrow}$ packet is generated from a RealUp name, NatUp bound row, RatUp comparison row, DyadicRatCoreUp comparison row, StreamNameUp finite-window row, RegSeqRatUp readback row, Archimedean bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3372.1, Definition 3372.2, Definition 3372.3, and Definition 3372.4 seed the finite property-level bound route over Real, StreamName, RegSeqRat, Rat, DyadicRatCore, Nat, and RealArchimedeanBoundSchema rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArchimedeanProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real order, trichotomy, quotient equality of real names, unbounded search, least-bound selection, Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, natural-rational-dyadic bound compatibility, finite-window readback, transport stability, replay stability, and ledger nonescape for BEDC.Derived.RealArchimedeanPropertyUp.

Chapter 3373

A Concrete Naming Certificate for $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ packet records the monotone-convergence route that is available when a bounded monotone real sequence is handed to a located supremum surface. It sits between the source sequence packet of chapter 1434, the located upper-bound and lower-approximation packet of chapter 1427, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite handoff from monotone sequence data to a located supremum read; it does not import arbitrary completeness of the real line.

Definition 3373.1 (Located monotone convergence carrier)**.** *A* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, B, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *source row, exposing the inspected sequence schedule, finite stream windows, regular readbacks, real seals, boundedness ledger, and monotonicity ledger.*
2. *B is the monotone bounded ledger read from M, recording the common upper-bound surface and the finite comparison rows that keep each inspected sequence value below it.*
3. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, displaying the upper-bound and lower-approximation ledgers supplied to the supremum packet.*
4. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *precision-window row through which the sequence observations, upper-bound ledger, and lower-approximation requests are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windowed sequence observations and the located-supremum approximation ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the requested dyadic precision, tail threshold, and comparison radii used to relate a late monotone term to the located supremum surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, U, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the monotone source, bounded ledger, located-supremum handoff, precision windows, regular readback, dyadic tolerance ledger, real seal, continuation route, provenance, and local naming row.*

9. *C records displayed* Cont *replay from a convergence request through $M, B, U, W, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* LocatedMonotoneConvergence$^{\uparrow}$, MonotoneRealSequence$^{\uparrow}$, LocatedSupremum$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocatedMonotoneConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through

$$M, B, U, W, R, D, E, H, C, P, N.$$

It has no coordinate for arbitrary real completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, or a selected ambient limit outside the displayed located-supremum handoff.

Lean4 ✓ `BEDC.Derived.LocatedMonotoneConvergenceUp.LocatedMonotoneConvergenceTasteGate single carrier alignment`

Theorem 3373.2 (Located monotone convergence NameCert obligations)**.** *For every accepted* LocatedMonotoneConvergence$^{\uparrow}$ *carrier*

$$L = (M, B, U, W, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{LocatedMonotoneConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the monotone source, bounded ledger, located-supremum handoff, shared precision window, regular readback, dyadic tolerance ledger, real seal, transports, replay, provenance, and local naming;*
2. *monotone bounded ledger admission, requiring B to be read from the same* MonotoneRealSequence$^{\uparrow}$ *packet M that supplies the sequence schedule and monotonicity ledger;*
3. *located-supremum handoff, requiring U to expose upper-bound and lower-approximation rows through the* LocatedSupremum$^{\uparrow}$ *surface before any convergence read is exported;*
4. RegSeqRat$^{\uparrow}$ *readback, requiring R to repack the inspected stream windows and approximation witnesses rather than a hidden real representative;*
5. *dyadic tolerance replay, requiring D to bind each requested precision to a finite tail threshold and comparison radius;*
6. Real$^{\uparrow}$ *seal non-escape, requiring E to remain downstream of M, B, U, W, R, D together with H, C, P, N.*

No obligation may call on arbitrary Real$^{\uparrow}$ *completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, or an ambient selected limit.*

Proof. Project the carrier of Definition 3373.1. Carrier admission exposes exactly the rows $M, B, U, W, R, D, E, H, C$ so the only sequence source is the accepted MonotoneRealSequence$^{\uparrow}$ row M.

Read the order-facing rows next. The bounded ledger B is attached to the same inspected schedule and monotonicity ledger carried by M, while U is the only located-supremum handoff row and therefore the only place where upper-bound and lower-approximation information can be read.

The convergence-facing rows are downstream of these order rows. The shared window W, regular readback R, and dyadic tolerance ledger D supply the finite precision request, readback route, tail threshold, and comparison radius before the real seal E is exposed.

Finally the structural rows H, C, P, N transport, replay, source, and name the same packet. Since the excluded completeness, order, quotient, choice, compactness, and selected-limit coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Theorem 3373.3 (Located monotone convergence supremum readback)**.** *Let $L = (M, B, U, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing convergence read exported by L factors through the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row U and its displayed upper-bound and lower-approximation ledgers, then through the shared* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. Thus the read is a located-supremum factorization of the monotone source M, not an appeal to ambient completeness of arbitrary real subsets.*

Lean4 ✓ `BEDC.Derived.LocatedMonotoneConvergenceUp.LocatedMonotoneConvergenceCarrier supremum readback`

Proof. Start with a Real-facing convergence consumer. By Definition 3373.1, the only terminal real coordinate is E, and it is displayed after the monotone source, bounded ledger, located-supremum handoff, shared window, regular readback, and dyadic tolerance ledger.

Apply Definition 3373.2. Its located-supremum handoff row forces the upper-bound and lower-approximation reads through U, while its readback and tolerance rows force the same request through W, R, D before E can be read.

The packet M supplies only the bounded monotone source surface, and H, C, P, N can only transport, replay, source, and name the displayed route. Therefore every exported convergence read factors through $\mathsf{LocatedSupremum}^{\uparrow}$ and not through an ambient completeness principle. □

Theorem 3373.4 (Located monotone convergence bounded source route)**.** *Let $L = (M, B, U, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet whose source row is supplied by an accepted* $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *packet. Then every convergence read first consumes the source packet's monotonicity ledger, boundedness ledger, and* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff before it may inspect the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row U, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The readback is therefore a bounded-monotone-source route followed by a located-supremum route, not an ambient completeness shortcut.*

Proof. Project the carrier of Definition 3373.1. The rows M and B are the only source-facing rows available before the located-supremum handoff U and the readback rows W, R, D, E.

When M is supplied by $\mathsf{MonotoneBoundedSequence}^{\uparrow}$, apply Definition 3407.4. The source-side read must expose the monotonicity ledger, boundedness ledger, and $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff through the same finite source surface before a located convergence consumer can inspect later rows.

Now apply Definition 3373.3. After the bounded source route is fixed, every $\mathsf{Real}^{\uparrow}$-facing convergence read factors through U, then through the shared window and readback rows W, R, D, and finally through the terminal seal E.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and naming. Since neither carrier has a coordinate for arbitrary real completeness or a selected ambient limit, the readback cannot bypass the bounded-source route. □

Theorem 3373.5 (Located monotone convergence source induction)**.** *Let*

$$L = (M, B, U, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing convergence read exported by* L *is generated by induction over the displayed finite rows* $M, B, U, W, R, D, E, H, C, P, N$*. The base rows are the monotone source* M *and boundedness ledger* B*. The step rows are the located-supremum handoff* U*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*,* Cont *replay* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}$ *row* N*. The induction exposes no selected limit, quotient representative, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Begin with the finite carrier displayed in Definition 3373.1. Its only rows are $M, B, U, W, R, D, E, H, C, P, N$, so an induction on accepted convergence reads has base information only from the monotone source and boundedness ledger.

Apply Definition 3373.2. The NameCert obligations force carrier admission, bounded ledger admission, located-supremum handoff, readback, tolerance replay, and Real seal non-escape to occur through those same displayed rows.

The readback steps are fixed by Definition 3373.3. After the base rows are admitted, each Real-facing read must pass through U, W, R, D, E in order, with H, C, P, N only transporting, replaying, sourcing, and naming the route.

If the source is supplied by a bounded monotone packet, Definition 3373.4 identifies the same base rows as the monotonicity ledger, boundedness ledger, and finite source handoff before any later convergence row can be inspected.

Thus the induction covers every exported convergence read using exactly the displayed finite rows. Since the carrier and its NameCert obligations contain no coordinate for a selected limit, quotient representative, countable choice, trichotomy, or ambient Real completeness, none can appear in the generated read. □

Falsifiable prediction: Every accepted $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ convergence read factors through a MonotoneRealSequence source, a boundedness ledger, a LocatedSupremum handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real-facing convergence consumer may inspect it.

Independence witness: monotone-real-sequence, locatedsupremum, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a bounded monotone sequence source and a located-supremum handoff before the Real-facing convergence read is sealed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3373.1, Definition 3373.2, Definition 3373.3, Definition 3373.4, and Definition 3373.5 seed the finite handoff from bounded monotone real sequence data to a located-supremum Real-facing convergence read.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient stream equality, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, monotone bounded ledger admission, located-supremum handoff, RegSeqRat readback, dyadic tolerance replay, and Real seal non-escape for BEDC.Derived.LocatedMonotoneConvergenceU

Constructive story: A $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ packet is generated from a MonotoneRealSequence source, a monotone bounded ledger, a LocatedSupremum handoff, a StreamName precision window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3374

A Concrete Naming Certificate for $\mathsf{BaireSteinhaus}^{\uparrow}$

Following Baire, Steinhaus, and Banach–Steinhaus, the $\mathsf{BaireSteinhaus}^{\uparrow}$ packet records the finite BEDC handoff by which a Baire-category witness, a complete metric source, norm rows, functional-analysis probes, and a uniform-boundedness consumer are named in one certificate. The packet sits between the Baire-category surface of chapter 1542, the complete-metric surface of chapter 128, the norm surface of chapter 89, the functional-analysis source of chapter 94, and the uniform-boundedness boundary of chapter 1835. It is a traditional analysis reconstruction in the BEDC carrier language: the certificate names the finite pointwise-bounded handoff without asserting host Banach-space completeness, countable choice, quotient equality of function spaces, or an unrestricted uniform-boundedness theorem.

Definition 3374.1 (Baire–Steinhaus carrier)**.** *A* $\mathsf{BaireSteinhaus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, R, F, U, W, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open row read by the pointwise-bounded argument.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row supplying the complete metric carrier on which the Baire row is interpreted.*
3. *R is the* $\mathsf{Norm}^{\uparrow}$ *row used to read boundedness witnesses as finite norm estimates.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *source row naming the family of linear probes and the finite test windows visible to the packet.*
5. *U is the* $\mathsf{UniformBoundedness}^{\uparrow}$ *consumer row that may receive a uniform-bound request only after B, M, R, F have been displayed.*
6. *W is the pointwise-boundedness ledger, recording the inspected point, operator family window, norm bound, and finite comparison tolerance.*
7. *E is the Baire–Steinhaus handoff row from the pointwise ledger W to the uniform-boundedness consumer U.*
8. *H records componentwise* hsame *transport for B, M, R, F, U, W, E.*
9. *C records displayed* Cont *replay from a pointwise-bounded request through the Baire, complete-metric, norm, functional-analysis, and handoff rows.*

10. *P is the* Pkg *provenance over* BaireSteinhaus$^{\uparrow}$, BaireCategory$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Norm$^{\uparrow}$, FunctionalAnalysis$^{\uparrow}$, UniformBoundedness$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$ *row.*

The carrier accepts only packets whose Baire, complete-metric, norm, functional-analysis, pointwise-boundedness, uniform-boundedness, transport, replay, provenance, and local naming rows are displayed in the same finite certificate.

Lean4 ✓ `BEDC.Derived.BaireSteinhausUp.BaireSteinhausTasteGate single carrier alignment`

Theorem 3374.2 (Baire–Steinhaus NameCert obligations)**.** *For every accepted* BaireSteinhaus$^{\uparrow}$ *carrier $S = (B, M, R, F, U, W, E, H, C, P, N)$, the local* NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* BaireCategory$^{\uparrow}$ *row,* CompleteMetric$^{\uparrow}$ *row,* Norm$^{\uparrow}$ *row,* FunctionalAnalysis$^{\uparrow}$ *row,* UniformBoundedness$^{\uparrow}$ *consumer row, pointwise-boundedness ledger, Baire–Steinhaus handoff, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. No obligation may read host Banach-space completeness, countable choice, quotient equality of function spaces, a selected dense point, or an unrestricted uniform-boundedness theorem.*

Proof. Project the carrier of Definition 3374.1. The rows $B, M, R, F, U, W, E, H, C, P, N$ are the only coordinates available to the local certificate, so every obligation is a finite row obligation.

The Baire-category side is read through B over the complete metric row M. The norm and functional-analysis rows R and F supply only the displayed finite observations used by the pointwise-boundedness ledger W.

The handoff row E is then the only route from W to the uniform-boundedness consumer U. Transport, replay, provenance, and local naming preserve that same route, and the carrier contains no coordinate for the excluded host-level principles. □

Theorem 3374.3 (Baire–Steinhaus pointwise-bounded handoff)**.** *Every pointwise-boundedness read accepted by a* BaireSteinhaus$^{\uparrow}$ *packet factors through the finite route*

$$\text{BaireCategory}^{\uparrow},\ \text{CompleteMetric}^{\uparrow},\ \text{Norm}^{\uparrow},\ \text{FunctionalAnalysis}^{\uparrow} \longmapsto \text{UniformBoundedness}^{\uparrow}.$$

The pointwise ledger W is exposed before the handoff row E can feed U, and the UniformBoundedness$^{\uparrow}$ *consumer receives only that displayed handoff. It receives no host Banach-space completeness theorem, no global operator-family supremum, no countable-choice selector, no quotient equality of function spaces, and no selected dense point.*

Proof. Begin with a pointwise-bounded request. The continuation row C first routes the request through the Baire row B and the complete-metric row M, so the category argument is anchored in the displayed metric source.

Next the request reads the norm row R and functional-analysis row F. These rows expose the finite operator-family windows and norm estimates stored in W, not a host family of all operators.

Finally the handoff row E repacks the displayed pointwise ledger for the uniform-boundedness consumer U. Since H, P, N only transport, source, and name the same finite rows, the consumer cannot recover any excluded completeness, choice, quotient, supremum, or selected-point principle. □

Closure status for NameCert$_{\text{BaireSteinhaus}^{\uparrow}}$**.**

Constructive story: A BaireSteinhaus$^{\uparrow}$ packet is generated from a BaireCategory row, CompleteMetric row, Norm row, FunctionalAnalysis source row, UniformBoundedness consumer row, pointwise-boundedness ledger, handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BaireSteinhaus}\uparrow}$, seed)

Scope closed: Definition 3374.1, Definition 3374.2, and Definition 3374.3 seed the finite pointwise-boundedness handoff over BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, and UniformBoundedness rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BaireSteinhaus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Banach-space completeness, countable choice, quotient equality of function spaces, selected dense points, global operator-family suprema, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise-boundedness ledger exposure, Baire-category routing, norm and functional-analysis compatibility, uniform-boundedness handoff, transport replay, provenance, and local naming for BEDC.Derived.BaireSteinhausUp.

Chapter 3375

A Concrete Naming Certificate for $\mathsf{BanachSteinhaus}^{\uparrow}$

Following Banach and Steinhaus, the $\mathsf{BanachSteinhaus}^{\uparrow}$ packet records the traditional theorem-level surface behind the uniform boundedness principle as finite NameCert data. It is the neighbour of the Baire handoff in chapter 3374 and the operator-family certificate in chapter 2302: the packet names the theorem route from Baire-category completeness data to a uniform boundedness read without importing a host Banach-space theorem, countable choice, quotient equality of function spaces, or an unrecorded operator-family supremum.

Definition 3375.1 (Banach–Steinhaus carrier)**.** *A* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, M, N, F, U, L, W, H, C, P, K)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *row that supplies the category witness used by the theorem-level route.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over the displayed source, keeping the Baire read tied to a complete metric carrier.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row through which pointwise and uniform boundedness are read as norm estimates.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row naming the linear-probe family and the finite observation windows visible to the packet.*
5. *U is the* $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *row that receives the theorem-level uniform-bound read after the Baire, completeness, norm, and functional-analysis rows have been displayed.*
6. *L is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *family row over the same source and target surfaces.*
7. *W is the pointwise-boundedness-to-uniform-bound ledger, recording the inspected source windows, operator rows, norm bounds, and the single uniform operator-bound output row.*
8. *H records componentwise* hsame *transport for B, M, N, F, U, L, W.*
9. *C records displayed* Cont *replay from pointwise boundedness requests through the Baire-category, complete-metric, norm, functional-analysis, bounded-operator, and uniform-boundedness rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BanachSteinhaus}^{\uparrow}$, $\mathsf{BaireSteinhaus}^{\uparrow}$, $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *K is the local* $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host-level Banach-space completeness, countable choice, quotient equality of maps, an unrestricted operator-family selector, or a theorem object outside the finite route.

Lean4 ✓ `BEDC.Derived.BanachSteinhausUp.BanachSteinhausTasteGate single carrier alignment`

Theorem 3375.2 (Banach–Steinhaus NameCert obligations)**.** *For every accepted* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier* $T = (B, M, N, F, U, L, W, H, C, P, K)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire-category row, complete-metric row, norm row, functional-analysis row, uniform-boundedness-principle row, bounded-linear-operator family row, pointwise-to-uniform ledger, componentwise transport, displayed replay, provenance, and local naming; Baire-source compatibility between B and M; operator-family compatibility between F and L; uniform-bound exactness through W and U; and non-escape from host completeness, countable choice, quotient equality of maps, unrestricted operator-family selectors, and theorem objects outside the displayed finite packet.*

Proof. Project the carrier of Definition 3375.1. The tuple $B, M, N, F, U, L, W, H, C, P, K$ is the whole accepted packet, so carrier admission and the named local rows are direct reads from the finite history.

The Baire-source compatibility is the shared route through B and M. The category row is interpreted only over the complete metric row displayed in the same packet, so a consumer cannot detach the Baire read from the carried source.

The operator-family compatibility is the shared route through F and L. The functional-analysis row names the finite probe windows, and the bounded-linear-operator row records the operator-family data over those same windows.

The uniform-bound exactness is the ledger W feeding the uniform-boundedness-principle row U. The transport, replay, provenance, and naming rows H, C, P, K certify that the same displayed route is preserved.

Since no coordinate names the excluded host-level objects, the local NameCert surface exports only the finite Banach–Steinhaus route. □

Theorem 3375.3 (Banach–Steinhaus Baire handoff)**.** *Every theorem-level read of an accepted* $\mathsf{BanachSteinhaus}^{\uparrow}$ *carrier factors through the ordered handoff*

$$\mathsf{BaireCategory}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \mathsf{UniformBoundednessPr}$$

The $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$ *row receives the ledger W only after the Baire and complete-metric rows have been exposed through the same* BHist *packet. Thus the packet is a theorem-level neighbour of chapter 3374 and chapter 2302, not a replacement for either sibling and not a host Banach–Steinhaus theorem outside BEDC data.*

Proof. Read the pointwise-boundedness request through the continuation row C. The first part of the route exposes the Baire-category row B over the complete-metric row M, matching the Baire side of the handoff named in chapter 3374.

Next read the norm row N, the functional-analysis row F, and the bounded-linear-operator family row L. These rows provide the finite operator windows and norm estimates that can be assembled into the pointwise-to-uniform ledger W.

Finally feed W to the uniform-boundedness-principle row U, the same theorem-level consumer named in chapter 2302. The rows H, P, K transport, source, and name this route without adding an external theorem object.

The exported handoff is therefore exactly the displayed finite route, with no host completeness principle, countable-choice selector, quotient equality of function spaces, or unrestricted family supremum. □

Falsifiable prediction: Every accepted $\mathsf{BanachSteinhaus}^{\uparrow}$ packet must expose $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{UniformBoundednessPrinciple}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows before any theorem-level uniform-bound read is public.

Independence witness: baire-steinhaus, uniform-boundedness-principle, boundedlinearoperator, norm, complete-metric: $\mathsf{BanachSteinhaus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the Baire-category source, complete-metric source, norm read, functional-analysis probe family, bounded-operator family, pointwise-to-uniform ledger, theorem-level uniform-boundedness consumer, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3375.1, Definition 3375.2, and Definition 3375.3 seed the finite theorem-level Banach–Steinhaus route over BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, UniformBoundednessPrinciple, BoundedLinearOperator, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachSteinhaus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Banach-space completeness, countable choice, quotient equality of maps, unrestricted operator-family selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-source compatibility, operator-family compatibility, uniform-bound ledger exactness, Baire handoff stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.BanachSteinhausUp.

Constructive story: A $\mathsf{BanachSteinhaus}^{\uparrow}$ packet is generated from BaireCategory, CompleteMetric, Norm, FunctionalAnalysis, UniformBoundednessPrinciple, BoundedLinearOperator, pointwise-to-uniform ledger, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 3376

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflection}^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley reflection terminology, the $\mathsf{CauchyCompletionReflection}^{\uparrow}$ packet records the finite BEDC boundary where Cauchy completion data reflect through the displayed unit and counit routes before reaching the Real seal. It reads the dense unit route of chapter 1527, the completed-object return route of chapter 1539, the comparison surface of chapter 1954, the broader completion-reflection boundary of chapter 403, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a nonquotient reflection certificate for displayed completion routes; it does not identify quotient completions, choose stream representatives, assert host equality, invoke countable choice, or import ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3376.1 (CauchyCompletionReflection carrier)**.** *A* $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row admitting source data into the completion-facing presentation.*
2. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning completed data to a displayed completion consumer.*
3. *Q is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row comparing the unit/universal-property presentation with the functorial metric-completion presentation.*
4. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *row tying the unit and counit routes to the same reflection boundary.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected before any terminal seal can be read.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the stream windows.*
7. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row shared by the unit, counit, comparison, and reflection routes.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, A have been displayed.*

9. *H records componentwise* hsame *transport for U, V, Q, F, S, D, A, E and the structural rows.*

10. *C records displayed* Cont *replay of the route*

$$U, V, Q, F \longmapsto S, D, A \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionReflection}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectionUp.CauchyCompletionReflectionTasteGate single carrier alignment`

Theorem 3376.2 (CauchyCompletionReflection NameCert obligations)**.** *For every accepted* CauchyCompletionRefl *carrier*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the unit route, counit route, comparison row, reflection row, stream windows, dyadic ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*

2. *unit exposure, requiring source admission to pass through U before a completion-facing read can inspect Q or F;*

3. *counit exposure, requiring completed-object return to pass through V before the same Q, F reflection boundary is read;*

4. *comparison agreement, requiring the unit route and counit route to use the shared* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row Q before the finite analytic route is public;*

5. *reflection stability, requiring F to be the only reflection witness tying U, V, Q to the terminal seal;*

6. *finite seal routing, requiring all Real-facing reads to pass through S, D, A, E and then through H, C, P, N;*

7. *non-escape, requiring the packet to source and name no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Carrier admission is projection from Definition 3376.1. The accepted tuple exposes exactly the unit, counit, comparison, reflection, analytic readback, and structural rows listed there.

For the unit side, the source row is read through U before any completion-facing comparison is available. For the counit side, the completed-object return is read through V before the same comparison and reflection boundary can be inspected.

The comparison and reflection obligations use Q and F. The comparison row identifies the displayed completion presentations, while F records that both the unit and counit reads are attached to the same completion-reflection boundary.

Finally the finite seal route is forced through S, D, A, E. Induction on the finite $\mathsf{StreamName}^{\uparrow}$ windows in S shows that each inspected window carries only its dyadic tolerance row and regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is reached. The transport, replay, provenance, and naming rows H, C, P, N preserve only these displayed coordinates, so no excluded completion or choice principle is exported. □

Theorem 3376.3 (CauchyCompletionReflection unit-counit triangle)**.** *Let*

$$R = (U, V, Q, F, S, D, A, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier. Every consumer that compares the unit route with the counit route factors through the finite triangle*

$$U, V \longmapsto Q, F \longmapsto S, D, A \longmapsto E.$$

The triangle is a displayed NameCert route over $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It is not quotient completion equality, not selected representative equality, not host equality of completed objects, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier. A unit-facing read has access to U, and a counit-facing read has access to V; neither route contains a terminal seal on its own.

The two reads meet only at Q and F. The comparison row supplies the shared completion presentation, and the reflection row records the boundary that both the unit and counit routes must use.

The remaining analytic data are S, D, A, E. Induction on the finite stream-window family shows that every triangle consumer sees exactly the window row, attached dyadic tolerance, regular rational readback, and Real seal before the structural rows H, C, P, N replay and name the route. Since no other coordinate is present, the excluded quotient, representative, host-equality, choice, and completeness data cannot be read. □

Theorem 3376.4 (CauchyCompletionReflection Real-seal handoff)**.** *For an accepted* $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *carrier* $R = (U, V, Q, F, S, D, A, E, H, C, P, N)$, *every Real-facing completion-reflection consumer receives the terminal seal* E *only through the route*

$$U, V, Q, F \longmapsto S, D, A \longmapsto E \longmapsto H, C, P, N.$$

Thus the handoff to $\mathsf{Real}^{\uparrow}$ *is mediated by the sibling* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows and by the displayed* $\mathsf{CompletionReflection}^{\uparrow}$ *boundary. It exports no quotient completion equality, selected representative, host equality, countable choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean bridge, or detached seal.*

Proof. Start with a Real-facing consumer. Projection of the accepted carrier gives the only available completion-reflection rows: U, V, Q, F and the finite analytic route S, D, A, E.

The consumer cannot read E before the sibling route has been displayed. The stream windows in S determine the finite observations, D supplies their tolerances, and A supplies the regular rational readback.

After E is reached, H and C transport and replay the same route, while P and N source and name it. No carrier coordinate stores a quotient completion, selected representative, host equality, choice schedule, compactness argument, ambient real-completeness theorem, Lean bridge, or detached seal, so no Real-facing handoff can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflection}^{\uparrow}$ packet is generated from CauchyCompletionUnit and CauchyCompletionCounit routes, a CauchyCompletionComparison row, a CompletionReflection witness, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3376.1, Definition 3376.2, Definition 3376.3, and Definition 3376.4 seed the finite nonquotient reflection boundary for L10 completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, counit exposure, comparison agreement, reflection stability, finite seal routing, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletion

Chapter 3377

A Concrete Naming Certificate for $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$

Following Bishop–Bridges constructive completion and Bourbaki–Kelley separated reflection, the $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$ packet records the finite interface where a displayed Cauchy completion route is read only after its separated-reflection boundary is visible. It sits beside the completion reflection packet of chapter 3376, the separated completion reflection packet of chapter 3543, the Hausdorff reflection packet of **??**, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names separated reflection as BEDC rows in the completion route; it does not quotient completed points, choose representatives, assert host equality, import trichotomy, use countable choice, or claim ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3377.1 (CauchyCompletionSeparatedReflection carrier)**.** *A* $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, S, H_0, W, Q, D, E, T, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyCompletionReflection}^{\uparrow}$ *route that exposes unit, counit, comparison, and completion-reflection data before any real-facing seal is read.*
2. *S is the* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *row that records zero-distance separation as a displayed completion-uniqueness interface.*
3. *H_0 is the* $\mathsf{HausdorffReflection}^{\uparrow}$ *row that keeps the separated classifier visible before a completion consumer reads equality-facing data.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before rational observations are consumed.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row shared by the completion and separation routes.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the finite windows and readback comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W, Q, D and the separated reflection rows have been displayed.*
8. *T records componentwise* hsame *transport for R, S, H_0, W, Q, D, E and the structural rows.*

9. *C records displayed* Cont *replay of the separated completion-reflection route.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$, $\mathsf{CauchyCompletionReflection}^{\uparrow}$, $\mathsf{SeparatedCompletionReflection}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected completed representatives, host equality, trichotomy, countable choice, open-cover compactness, or ambient real completeness.

Theorem 3377.2 (CauchyCompletionSeparatedReflection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$ *carrier*

$$K = (R, S, H_0, W, Q, D, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-reflection exposure, separated-reflection exposure, Hausdorff boundary visibility, finite-window admission, regular rational readback, dyadic tolerance readback, terminal real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the accepted carrier through Definition 3377.1. Its displayed coordinates are exactly $R, S, H_0, W, Q, D, E, T, C, P, N$, and each coordinate has one local role in the separated completion-reflection route.

The completion-facing part is R, S, H_0. The row R supplies the Cauchy completion-reflection surface, S supplies the separated completion-reflection boundary, and H_0 keeps the Hausdorff reflection classifier visible before any equality-facing completion read is consumed.

The analytic readback part is W, Q, D, E. The finite stream window, regular rational readback, and dyadic tolerance rows must be displayed before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Finally T, C, P, N transport, replay, source, and name the same rows. Since the carrier has no further coordinate, the local certificate cannot export quotient completion equality, selected representatives, host equality, trichotomy, countable choice, compactness, or ambient real completeness. □

Theorem 3377.3 (CauchyCompletionSeparatedReflection universal route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{CauchyCompletionReflection}^{\uparrow} \to \mathsf{SeparatedCompletionReflection}^{\uparrow} \to \mathsf{HausdorffReflection}^{\uparrow} \to \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRa}$$

The route is universal for the packet in the finite BEDC sense: any accepted read with the same completion row, separated reflection row, Hausdorff boundary, finite analytic readback, transport, replay, provenance, and local naming is the same NameCert *read. It is not a quotient completion, not a selected representative, not host equality of completed points, not trichotomy, not countable choice, and not ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Start with a completion-facing consumer. The carrier admits no direct route to E; the consumer must first read the completion-reflection row R, then the separated completion-reflection row S, and then the Hausdorff boundary row H_0.

Next consume the finite analytic rows. The stream window W, regular rational readback Q, and dyadic tolerance ledger D are the only rows between the separated boundary and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The structural rows force uniqueness of the local read. Componentwise transport T and replay C preserve the same displayed route, while P and N source and name it. Thus two accepted reads with the same displayed rows are the same local NameCert read, and none of the refused host objects can enter the universal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}$ packet is generated from CauchyCompletionReflection, SeparatedCompletionReflection, HausdorffReflection, StreamName, RegSeqRat, DyadicRatCore, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3377.1, Definition 3377.2, and Definition 3377.3 seed the finite separated-reflection interface for Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionSeparatedReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completion equality, selected completed representatives, host equality, trichotomy, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-reflection exactness, Hausdorff-boundary visibility, finite analytic readback, non-escape, and a Lean carrier formalization for BEDC.Derived.CauchyCompletionSeparatedReflect

Chapter 3378

A Concrete Naming Certificate for $\mathsf{CauchyRealArchimedean}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyRealArchimedean}^{\uparrow}$ packet records the finite Archimedean witness consumed by L10 Cauchy-real readers before an order, located-comparison, or completion-facing consumer opens a real-name seal. It sits over the regular rational source rows of chapter 301, the dyadic scale rows of chapter 291, the terminal real-name surface of chapter 38, the Cauchy-real presentation route of chapter 1511, and the rational-bound witness surface of chapter 430. The packet names a displayed finite bound route; it is not a host least-bound principle, not quotient equality of real names, not a choice-selected integer part, and not ambient real completeness.

Definition 3378.1 (CauchyRealArchimedean carrier)**.** *A* $\mathsf{CauchyRealArchimedean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (G, D, E, W, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose finite regular-rational observations are the only Cauchy-real reads admitted by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the dyadic tolerance, comparison budget, and rational arithmetic used to inspect G.*
3. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the displayed route through G and D.*
4. *W is the Archimedean bound witness row, read as the rational or dyadic bound surface supplied by chapter 430 for the sealed Cauchy-real name.*
5. *H records componentwise* hsame *transport for G, D, E, W, C, P, N.*
6. *C records displayed* Cont *replay from regular-rational windows through dyadic scale inspection, the terminal real seal, and the Archimedean witness row:*

$$G \longmapsto D \longmapsto E \longmapsto W.$$

7. *P is the* Pkg *provenance over* $\mathsf{CauchyRealArchimedean}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealPresentation}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through G, D, E, W, H, C, P, N. It has no coordinate for a host supremum, a selected least bound, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3378.2 (CauchyRealArchimedean NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealArchimedean}^{\uparrow}$ *carrier $A = (G, D, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed regular-rational source, dyadic scale, real seal, Archimedean witness, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring comparisons of accepted packets to inspect matching G, D, E, W rows through the componentwise* hsame *data in H;*
3. *dyadic-scale exactness, requiring every bound or comparison request to pass through the displayed scale row D before the real seal E is consumed;*
4. *Archimedean witness exposure, requiring the bound witness W to be read after the same G, D, E route and before any order or located-comparison consumer uses it;*
5. *continuation, provenance, and naming stability through C, P, N.*

Proof. Project the carrier in Definition 3378.1. The regular-rational observations, dyadic scale data, real-name seal, bound witness, transport, replay, provenance, and local name are the only displayed coordinates.

The analytic route is supplied by the sibling Cauchy-real chapters. The $\mathsf{Real}^{\uparrow}$ seal is read through chapter 38, the regular rational source through chapter 301, the scale row through chapter 291, and the presentation face through chapter 1511.

The Archimedean part is then the displayed witness row. Apply the rational-bound surface of chapter 430: it supplies a finite bound route for the same real-name seal, while this carrier records that route as W downstream of G, D, E.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no field stores a host supremum, selected least bound, quotient stream equality, countable-choice schedule, host equality row, or ambient completeness object, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyRealArchimedeanUp.CauchyRealArchimedeanCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRealArchimedean}^{\uparrow}$ packet is generated from displayed $\mathsf{RegSeqRat}^{\uparrow}$ source rows, $\mathsf{DyadicRatCore}^{\uparrow}$ scale rows, a terminal $\mathsf{Real}^{\uparrow}$ seal row, an Archimedean bound witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$, seed)

Scope closed: Definition 3378.1 and Definition 3378.2 seed the finite Archimedean witness route for Cauchy-real names over RegSeqRat, DyadicRatCore, Real, CauchyRealPresentation, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealArchimedean}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected least bounds, integer-part selection, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic-scale exactness, Archimedean witness exposure, replay stability, provenance stability, and local naming for the same displayed packet.

Chapter 3379

A Concrete Naming Certificate for $\mathsf{CauchyCompletionOperator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCompletionOperator}^{\uparrow}$ packet names the finite operator surface that turns displayed Cauchy data into a completion-facing handoff without selecting quotient representatives. It sits between the metric completion carrier of chapter 1506, the Cauchy carrier of chapter 3413, the StreamName finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the completion boundary of chapter 3350, the uniform completion route of chapter 1473, the separated limit surface of chapter 2355, and the terminal real-name seal of chapter 38. The operator is therefore a finite NameCert object over visible StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, completion-boundary rows, and Real-facing seals, not a host completion class or a choice of limits.

Definition 3379.1 (CauchyCompletionOperator carrier)**.** *A* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing the metric Cauchy request and its displayed finite tolerance route;*
2. *B is the* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *row, recording the handoff from metric and uniform completion reads to finite-window and Real-facing consumers;*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row, recording the entourage-facing completion presentation consumed by the same operator;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule of finite windows inspected by the completion request;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the scheduled windows;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the Cauchy request, readback, and seal boundary;*
7. *Q is the* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *separated-limit row that receives the operator handoff only through the visible completion boundary;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, U, S, R, D, Q have been displayed;*

9. *H records componentwise* hsame *transport for the source, boundary, uniform, window, readback, dyadic, separated-limit, Real-seal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from an operator request through $M, B, U, S, R, D, Q, E, H, P, N$;*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionOperator}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompleteSeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$ *row.*

The classifier accepts only packets whose operator handoff is exhausted by $M, B, U, S, R, D, Q, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorTasteGate single carrier alignment`

Theorem 3379.2 (CauchyCompletionOperator NameCert obligation surface)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier $O = (M, B, U, S, R, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperat}}$ *surface consists exactly of carrier admission for the displayed twelve-row packet; metric-completion exposure through M; completion-boundary exposure through B; uniform-completion exposure through U; finite-window exposure through S; regular-rational readback through R; dyadic-tolerance exposure through D; separated-limit handoff through Q; terminal Real sealing through E; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperator namecert obligation surface`

↪ variant `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCarrier namecert obligation surface`

Proof. Project the carrier of Definition 3379.1. The accepted packet exposes exactly the metric source row, completion-boundary row, uniform-completion row, finite windows, regular readback, dyadic ledger, separated-limit row, Real seal, and structural rows.

The operator obligations are read in the displayed order. The metric and uniform completion requests first pass through the boundary row B; the scheduled windows S, readback R, and dyadic ledger D then supply the finite evidence needed by both completion and Real-facing consumers.

The separated-limit and Real-seal rows are downstream of the same packet. The row Q is reached only through the visible boundary route, and the terminal seal E is reached only after the finite windows, readback, and tolerance ledger have already been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient real completeness are not coordinates of O, they cannot appear as local NameCert obligations. □

Theorem 3379.3 (CauchyCompletionOperator functorial handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet factors through the finite handoff*

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

Thus the operator may be replayed by later functorial completion surfaces only as a displayed route from metric and uniform completion requests through common finite windows, dyadic tolerances,

regular readback, separated-limit handoff, Real sealing, transport, replay, provenance, and local naming.

Proof. Start with an accepted carrier. The completion-facing rows M and U are not independent host completions; they are source rows of the same finite packet.

The boundary row B aligns those source rows before any consumer reaches the scheduled windows. The rows S, R, D then expose the common finite observations, regular rational readback, and dyadic tolerance ledger used by downstream completion requests.

The rows Q and E are terminal for the separated-limit and Real-facing sides of the handoff. They do not insert a quotient completion or a selected representative between the finite evidence and the terminal consumer.

The structural rows H, C, P, N preserve this route under transport, replay it as a displayed continuation, record provenance, and name the local certificate. Therefore any later functorial completion surface can consume only the displayed operator route. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorFunctorialHandoff`

Theorem 3379.4 (CauchyCompletionOperator triangle consumption route)**.** *Every accepted* CauchyCompletionOp *packet consumed by* CauchyCompletionFunctoriality$^{\uparrow}$ *or* ***??*** *for an identity or composition triangle read must route through*

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

The triangle read cannot bypass the completion-boundary row B, the finite StreamName$^{\uparrow}$ *windows S, the* RegSeqRat$^{\uparrow}$ *readback R, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, or the separated-limit handoff Q. It exports no quotient completion equality, selected representative, countable-choice schedule, host equality, selected limit, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorTriangleConsumptionRoute`

Proof. Project the carrier of Definition 3379.1. The only available rows are $M, B, U, S, R, D, Q, E, H, C, P, N$, so an identity or composition triangle consumer has no detached completion endpoint to inspect.

Apply Definition 3379.3. It places the metric and uniform source rows before B, then forces every completion-facing read through the finite windows, regular readback, dyadic tolerance ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming.

The downstream consumers are the functorial completion surface of chapter 1937 and the separated completion functor of chapter 4094. Both may consume the operator packet only through the displayed route, so their triangle reads inherit the same completion-boundary and finite-window order.

The exclusion clause follows from the carrier inventory. None of quotient completion equality, selected representatives, selected limits, countable choice, host equality, or ambient Real completeness is a coordinate of the accepted packet. □

Theorem 3379.5 (CauchyCompletionOperator finite-window extraction)**.** *Every accepted* CauchyCompletionOpera *packet exposes its completion-facing request through a finite extraction window before any separated-limit or Real endpoint row is available. The extracted window consists exactly of the* StreamName$^{\uparrow}$ *schedule S, the* RegSeqRat$^{\uparrow}$ *readback R, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger D, the completion-boundary row B, and the structural rows H, C, P, N that transport, replay, source, and name the same finite packet.*

Consequently a consumer may reach Q or E only through the displayed route

$$B, S, R, D, H, C, P, N \longrightarrow Q, E.$$

No selected limit, quotient completion equality, host completion operator, host real equality, or countable-choice schedule is exposed by the extraction.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorWindowExtraction`

Proof. Project the carrier of Definition 3379.1. The only rows between the metric and uniform source requests and the terminal consumers are the completion boundary, finite StreamName schedule, regular readback, dyadic ledger, separated-limit row, Real seal, and structural support.

Apply Definition 3379.2. Its obligation surface forces local naming to inspect B, S, R, D before Q or E can be consumed, while H, C, P, N only transport, replay, source, and name the displayed packet.

Use Definition 3379.3. The handoff order places the finite window and dyadic ledger between the completion source rows and the terminal rows, so the extraction is determined by the visible window data rather than by any host completion object.

The exclusion clause follows from the same carrier inventory. The packet has no coordinate for a selected limit, quotient completion equality, host completion operator, host real equality, or countable-choice schedule, so none can be read during extraction. □

Theorem 3379.6 (CauchyCompletionOperator ledger nonescape)**.** *No accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet exports quotient completion equality, selected representatives, selected limits, countable-choice schedules, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness data. Every consumer route to* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *must first read the packet rows* $M, B, U, S, R, D, Q, E, H, C, P, N$.

Proof. Inspect the carrier inventory in Definition 3379.1. Its analytic coordinates are the metric source, completion boundary, uniform source, finite windows, regular readback, dyadic ledger, separated-limit handoff, and Real seal.

The structural coordinates are only hsame, Cont, Pkg, and NameCert rows. They move, replay, source, and name the displayed packet; they do not add analytic data.

The excluded quotient, representative, limit, choice, host-operator, and ambient-completeness objects are absent from both inventories. Hence every separated-limit or Real-facing read is forced through the visible finite ledger and cannot escape the operator packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorLedgerNonescape`

Theorem 3379.7 (CauchyCompletionOperator Real-seal nonescape)**.** *Let T be any Real-facing consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier* $O = (M, B, U, S, R, D, Q, E, H, C, P, N)$. *If T reads the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, then the read first factors through the completion-boundary row B, the finite* $\mathsf{StreamName}^{\uparrow}$ *window S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the displayed transport, replay, provenance, and local naming rows H, C, P, N.*

The terminal seal therefore exports only the displayed finite operator packet. It does not expose Real completeness, quotient limits, selected representatives, choice, host equality, or an ambient completion endpoint.

Proof. Begin with Definition 3379.5. It isolates the finite window and tolerance ledger that every terminal consumer must pass through before Q or E can be inspected.

Next apply Definition 3379.3. The Real-facing side of the handoff receives E only after B, S, R, D and the structural rows have already fixed the displayed route.

Finally use Definition 3379.6. Since the packet exports no quotient completion equality, selected representative, selected limit, choice schedule, host completion operator, or ambient Real-completeness coordinate, the terminal Real seal cannot escape the finite operator packet. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorRealSealNonescape`

Theorem 3379.8 (CauchyCompletionOperator obligation surface)**.** *Every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

exposes exactly the following obligation surface: the $\mathsf{MetricCompletion}^{\uparrow}$ *row* M*, the* $\mathsf{CauchyCompletionBoundary}^{\uparrow}$ *row* B*, the* $\mathsf{UniformCompletion}^{\uparrow}$ *row* U*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *handoff* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$ *row* N*.*

The surface refuses quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission and local naming come from Definition 3379.2. That theorem projects exactly the twelve displayed rows $M, B, U, S, R, D, Q, E, H, C, P, N$ from the accepted carrier and assigns each row its local NameCert role.

The completion-facing order is supplied by Definition 3379.3. It places the metric and uniform source rows before the boundary row, then routes every later completion read through the finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming rows.

The finite inspection window is fixed by Definition 3379.5. It prevents Q or E from being consumed before B, S, R, D, H, C, P, N have been displayed.

Finally apply Definition 3379.6 and Definition 3379.7. These two nonescape clauses exclude quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, and ambient $\mathsf{Real}^{\uparrow}$ completeness from both the ledger and the terminal Real seal. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorPublicObligationSurface`

Theorem 3379.9 (CauchyCompletionOperator obligation triad)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N),$$

the local NameCert obligations decompose exactly into the following three faces:

1. *finite-window extraction, reading the completion-boundary row* B*, StreamName window* S*, RegSeqRat readback* R*, DyadicRatCore ledger* D*, and structural rows* H, C, P, N *before any terminal read;*
2. *functorial handoff, routing the metric and uniform source rows* M, U *through* B*, then through* S, R, D*, and only then to* Q, E, H, C, P, N*;*
3. *ledger nonescape, requiring every separated-limit or Real-facing consumer to read only the displayed rows* $M, B, U, S, R, D, Q, E, H, C, P, N$ *and no quotient completion equality, selected representative, selected limit, countable-choice schedule, host completion operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorObligationTriad`

Proof. The finite-window face is exactly Definition 3379.5. It isolates B, S, R, D, H, C, P, N as the finite evidence a consumer must inspect before Q or E is available.

The handoff face is exactly Definition 3379.3. It places M, U on the source side, routes them through the boundary row B, and sends every completion-facing consumer through the common finite window before the separated-limit and Real-seal rows.

The nonescape face combines Definition 3379.6 with Definition 3379.7. The first theorem prevents any consumer from leaving the displayed ledger, and the second theorem applies the same refusal to terminal Real-seal reads. Together they exhaust the obligation surface recorded in Definition 3379.8. □

Theorem 3379.10 (CauchyCompletionOperator obligation window route)**.** *Every accepted* CauchyCompletionOper *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

satisfies the completion-operator obligation sequence

$$M, U \longrightarrow B \longrightarrow S, R, D \longrightarrow Q, E \longrightarrow H, C, P, N.$$

The metric and uniform source rows M, U are read first; the completion-boundary row B mediates the two source reads; the rows S, R, D expose the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows; the rows Q, E are terminal separated-limit and Real-seal consumers; and the rows H, C, P, N only transport, replay, source, and name the same packet.*

Proof. Start from the carrier inventory in Definition 3379.1. An accepted packet has exactly the displayed metric source, completion boundary, uniform source, finite window, regular readback, dyadic tolerance, separated-limit, Real-seal, transport, replay, provenance, and naming rows.

Apply Definition 3379.2. The local NameCert surface reads the metric and uniform source rows as obligations of the same carrier, and the boundary row B is the displayed mediator between those source reads and every later completion-facing consumer.

Use Definition 3379.5. Before either terminal row is available, the operator exposes the finite $\mathsf{StreamName}^{\uparrow}$ schedule S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, together with the structural rows that keep the extraction attached to the same packet.

Finally invoke Definition 3379.7. The terminal Real read can only occur after the boundary, finite window, readback, dyadic ledger, and structural rows have fixed the route. The separated-limit row Q is consumed on the same downstream side of the packet, while H, C, P, N merely transport, replay, source, and name what the carrier already displays. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorObligationWindowRoute`

Theorem 3379.11 (CauchyCompletionOperator universal factorization)**.** *Every completion-universal or functorial completion consumer of an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

first reads the finite route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N.$$

If the same consumer also cites a $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row or a* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row, that row is admitted only as a downstream read of the same boundary, finite-window, separated-limit, Real-seal, transport, replay, provenance, and local naming route. The consumer receives no host universal property, quotient completion, selected representative, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness shortcut around O.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorUniversalFactorization`

Proof. Start with a completion-universal or functorial completion consumer of O. By Definition 3379.3, the consumer already reads the metric and uniform source rows through the displayed boundary row before it can inspect finite windows, regular readback, dyadic tolerances, separated-limit handoff, Real seal, transport, replay, provenance, or local naming.

Next apply Definition 3379.10. It fixes the same order as an obligation sequence: M, U precede B; B precedes S, R, D; the terminal rows Q, E are downstream of those finite observations; and H, C, P, N only preserve and name the displayed packet.

Now consider an additional $\mathsf{CompletionUniversalProperty}^{\uparrow}$ or $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ citation. Such a citation can be read only after the rows needed by the operator route have been displayed, because the operator exports completion-facing data solely through the boundary and window route just identified. The cited universal or functorial row is therefore a later consumer of O, not an independent source of completion evidence.

Finally use Definition 3379.6 and Definition 3379.7. These exclude quotient completion equality, selected representatives, selected limits, countable-choice schedules, host completion operators, ambient Real completeness, and terminal-seal escapes from the packet. Hence no universal-property or functorial completion consumer can bypass the finite CauchyCompletionOperator route. □

Theorem 3379.12 (CauchyCompletionOperator separated-limit factorization)**.** *Every* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *consumer reading row Q from an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N)$$

must first read the finite route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto H, C, P, N.$$

Thus Q is a downstream separated-limit handoff of the displayed finite-window route, not an independent completion witness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorSeparatedLimitFactorization`

Proof. Start with the obligation surface of Definition 3379.2. It exposes Q only as one row of the accepted carrier, after the metric source, completion-boundary, uniform source, StreamName window, RegSeqRat readback, and DyadicRatCore tolerance rows have already been displayed.

Apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. These two results force the separated-limit consumer through M, U, then B, then S, R, D, before the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same packet.

Finally use ledger nonescape, Definition 3379.6. Since quotient completion equality, selected limits, host completion operators, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, row Q cannot supply an independent completion object. It is only the separated-limit handoff downstream of the finite StreamName, RegSeqRat, and DyadicRatCore route. □

Theorem 3379.13 (CauchyCompletionOperator idempotent complete carrier)**.** *Let $O = (M, B, U, S, R, D, Q, E, H$ be an accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *packet whose source and target completion rows are already separated, and whose transport row is the identity replay over the displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows. Then any second application of the completion operator factors through the same finite observation windows, the same regular rational readback, the same dyadic tolerance ledger, and the same terminal Real seal.*

Thus the second application is read as an idempotent complete-carrier replay of O, not as a new quotient stream equality, selected limit, countable-choice schedule, host completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorIdempotentCompleteCarrier`

Proof. Start from the carrier of Definition 3379.1. The only rows available to either completion read are the metric and uniform source rows, the boundary row, the StreamName window, the RegSeqRat readback, the DyadicRatCore ledger, the separated-limit handoff, the Real seal, and the structural replay rows.

Apply the universal factorization theorem, Definition 3379.11. It forces every completion-universal or functorial consumer of the first application through $M, U, B, S, R, D, Q, E, H, C, P, N$ before any completion-facing handoff is available.

Next apply separated-limit factorization, Definition 3379.12. Since the source and target completion rows are already separated and the transport row is the identity replay over the displayed limit-seal, window, readback, dyadic, and Real rows, the second application has no additional separated-limit coordinate to inspect.

Finally use the scoped obligation closure of Definition 3379.14. The closed scope already exhausts carrier admission, finite-window routing, terminal Real sealing, transport, replay, provenance, and local naming. The repeated operator read therefore reuses the same finite observation windows and terminal seal without introducing quotient stream equality, a selected limit, countable choice, host completion data, or ambient Real completeness. □

Theorem 3379.14 (CauchyCompletionOperator scoped obligation closure)**.** *For every accepted* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *carrier*

$$O = (M, B, U, S, R, D, Q, E, H, C, P, N),$$

the metric and uniform source rows, completion-boundary row, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, separated-limit handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *rows jointly discharge the* obligation *surface for the operator. Every scope is routed through the displayed BEDC primitives and sibling chapters, and no scope is supplied by quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorScopedObligationClosure`

Proof. Start with the obligation surface and functorial handoff, Definition 3379.2 and Definition 3379.3. These expose the metric source, uniform source, completion boundary, finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, and structural rows in the displayed carrier order.

Apply finite-window extraction, ledger nonescape, and Real-seal nonescape from Definition 3379.5, Definition 3379.6, and Definition 3379.7. They force terminal separated-limit and Real-facing reads through the same finite window and prevent escape to quotient, choice, host, or ambient-completeness data.

Finally use the obligation triad, obligation-window route, universal factorization, and separated-limit factorization, Definition 3379.9, Definition 3379.10, Definition 3379.11, and Definition 3379.12. These rows exhaust carrier admission, source routing, terminal handoff, transport, replay, provenance, and local naming, so the scoped obligation surface is closed exactly at obligation. □

Theorem 3379.15 (CauchyCompletionOperator composition uniqueness route)**.** *Let O_0 and O_1 be accepted* CauchyCompletionOperator$^\uparrow$ *packets with the same* MetricCompletion$^\uparrow$, CauchyCompletionBoundary$^\uparrow$, UniformCompletion$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, DyadicRatCore$^\uparrow$, CompleteSeparatedMetric$^\uparrow$, Real$^\uparrow$, hsame, Cont, Pkg, *and* NameCert *rows. Then every completion-facing composition read induced by O_0 follows the same finite operator route as the read induced by O_1.*

Equivalently, agreement of the displayed rows forces agreement of the completion-facing handoff route

$$M, U \longmapsto B \longmapsto S, R, D \longmapsto Q, E \longmapsto H, C, P, N,$$

without asserting quotient completion equality, selected limits, host completion operators, countable choice, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCompositionUniquenessRoute`

Proof. Start with the functorial handoff of Definition 3379.3. It forces any composition-facing read of either packet to pass through the same metric source, uniform source, completion boundary, finite window, regular readback, dyadic tolerance ledger, separated-limit handoff, Real seal, transport, replay, provenance, and local naming rows.

Next apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. Since O_0 and O_1 agree on the displayed StreamName, RegSeqRat, DyadicRatCore, boundary, and structural rows, both composition reads expose the same finite observation window before Q or E is available.

Use universal factorization and separated-limit factorization, Definition 3379.11 and Definition 3379.12. These theorems route the universal, functorial, and separated-limit consumers through the already displayed packet rows, so no second completion-facing route can be generated from data outside the common row family.

Finally invoke scoped obligation closure, Definition 3379.14. The closed scope exhausts the admissible carrier, handoff, window, seal, transport, replay, provenance, and naming obligations. Therefore two accepted packets with the same displayed rows induce the same completion-facing route, while quotient completion equality, selected limits, host completion operators, countable choice, and ambient Real$^\uparrow$ completeness remain outside the claim. □

Theorem 3379.16 (CauchyCompletionOperator public export)**.** *The public* CauchyCompletionOperator$^\uparrow$ *export is the finite route*

$$M, B, U, S, R, D, Q, E, H, C, P, N$$

from metric and uniform completion requests through StreamName$^\uparrow$ *windows,* RegSeqRat$^\uparrow$ *readback,* DyadicRatCore$^\uparrow$ *tolerances,* CompleteSeparatedMetric$^\uparrow$ *handoff, and the terminal* Real$^\uparrow$ *seal. Every downstream completion consumer must cite this displayed route before using the operator, rather than quotient completion equality, host completion equality, a selected representative, a selected limit, countable choice, or ambient* Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorPublicExport`

Proof. Start with the obligation surface and functorial handoff, Definition 3379.2 and Definition 3379.3. They expose the source metric row, boundary row, uniform row, finite window, regular readback, dyadic ledger, separated-limit handoff, Real seal, and structural rows in the displayed carrier order.

Apply finite-window extraction and the obligation-window route, Definition 3379.5 and Definition 3379.10. These theorems force every completion-facing read through S, R, D before either the separated handoff Q or terminal seal E is available.

Use ledger nonescape, universal factorization, and separated-limit factorization, Definition 3379.6, Definition 3379.11, and Definition 3379.12. They keep the operator inside the finite dyadic and separated route, so no quotient, selected limit, host completion, countable-choice, or ambient-completeness coordinate can be read as a public export.

Finally invoke scoped obligation closure, Definition 3379.14. Since the scoped carrier surface already exhausts admission, finite-window routing, terminal Real sealing, transport, replay, provenance, and local naming, the public export is exactly the displayed finite route for downstream completion consumers. □

Theorem 3379.17 (Complete metric limit uniqueness outbound reference)**.** *Every public* CauchyCompletionOperat *export that later asserts uniqueness of a separated complete-metric limit must consume chapter 5983 after the public operator route of Definition 3379.16. The operator route exports the finite completion handoff only; it exports no selected limit, quotient representative, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First read the public export theorem. It binds every downstream completion consumer to the displayed route through $M, B, U, S, R, D, Q, E, H, C, P, N$ before the operator can be used.

A later uniqueness assertion is a separate separated complete-metric comparison. It therefore consumes the sibling route chapter 5983 after the operator export has supplied its completion-facing handoff.

Finally apply the operator nonescape clauses already used in the public export proof. They exclude selected limits, quotient representatives, countable-choice schedules, host completion objects, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the operator route, so uniqueness is imported only from the downstream sibling and not from an extra coordinate of $\mathsf{CauchyCompletionOperator}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorCompleteMetricLimitUniquenessOu`

Theorem 3379.18 (CauchyCompletionOperator bridge route)**.** *Every bridge-facing consumer of* $\mathsf{CauchyCompletionOperator}^{\uparrow}$ *must read the displayed operator through the functorial handoff, finite-window extraction, universal factorization, separated-limit factorization, and public export rows before treating it as a standard completion bridge. Concretely, the consumer first reads the metric and uniform source rows through* M, U *and the completion-boundary row* B*; then it reads the finite window, regular readback, and dyadic ledger rows* S, R, D*; then it reads the universal and separated-limit handoffs through* Q *and the terminal Real seal* E*; and only then may it replay transport, provenance, and local naming through* H, C, P, N*.*

This route is a bridge-readiness obligation on the paper surface. It does not assert bridgeChecked*, does not replace the existing public closure status, and does not add quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the functorial handoff, Definition 3379.3. It fixes the order in which metric and uniform source rows are routed through the displayed completion-boundary row before any terminal completion consumer is available.

Apply finite-window extraction, Definition 3379.5. This forces a bridge-facing consumer to inspect the $\mathsf{StreamName}^{\uparrow}$ window S, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D, and structural rows H, C, P, N before reading Q or E.

Use universal factorization and separated-limit factorization, Definition 3379.11 and Definition 3379.12. These theorems keep completion-universal, functorial, and separated-limit consumers inside the same displayed route.

Finally apply public export, Definition 3379.16. It binds downstream completion consumers to the finite public route. Since bridge checking is not claimed by that theorem or by the closurestatus block below, the two axes remain separate. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionOperatorUp.CauchyCompletionOperatorBridgeRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionOperator}^{\uparrow}$ packet is generated from MetricCompletion, CauchyCompletionBoundary, UniformCompletion, StreamName, RegSeqRat, DyadicRatCore, CompleteSeparatedMetric, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 3379.14 closes the scoped obligation surface over $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionBoundary}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompleteSeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows already exhibited by Definition 3379.1, Definition 3379.2, Definition 3379.5, Definition 3379.9, Definition 3379.10, Definition 3379.11, Definition 3379.12, and Definition 3379.16; the public export theorem binds downstream completion consumers to the same finite route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: No paper-side theory upgrade remains; bridge checking remains a separate formal-axis obligation unless a standard bridge theorem is supplied.

Chapter 3380

A Concrete Naming Certificate for $\mathsf{RealCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealCauchyCompleteness}^{\uparrow}$ packet records the classical regular-Cauchy completeness route as finite BEDC NameCert data. It sits over the dyadic radius ledger of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the Cauchy-limit seal of chapter 386, and the terminal real-name seal of chapter 38. The packet exposes a displayed completion handoff for regular Cauchy streams; it does not assert that $\mathsf{Real}^{\uparrow}$ already carries ambient completeness.

Definition 3380.1 (Real Cauchy completeness carrier)**.** *A* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, D, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule of finite windows used to inspect the Cauchy stream.*
2. *M is the explicit Cauchy-modulus row, exposing the threshold at which requested dyadic tolerances are met.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for positive radii, halving steps, comparison budgets, and tolerance addition.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback assembled from the retained windows and the modulus row.*
5. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row consuming the same S, M, D, R route before any real-facing read is exported.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the limit-seal handoff.*
7. *H records componentwise* hsame *transport for $S, M, D, R, L, E, C, P, N$.*
8. *C records displayed* Cont *replay of the diagonal route*

$$S, M, D \longmapsto R \longmapsto L \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyCompleteness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *row.*
The classifier accepts only packets that display the finite windows, explicit modulus, dyadic budgets, diagonal readback, limit-seal handoff, real seal, transport, replay, provenance, and local naming. It has no coordinate for quotient streams, selected limits, countable choice, open-cover compactness, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyCompletenessUp.RealCauchyCompletenessTasteGate single carrier alignment`

Theorem 3380.2 (Real Cauchy completeness NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier* $K = (S, M, D, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream windows, explicit Cauchy modulus, dyadic tolerance ledger, diagonal regular readback, CauchyLimitSeal handoff, Real seal, componentwise transport, displayed replay, provenance, and local naming. The completion-facing read is accepted only when the modulus row M is exposed before the diagonal readback R, the handoff row L consumes the same S, M, D, R route, and every* $\mathsf{Real}^{\uparrow}$ *consumer reaches E through that displayed route.*

Proof. Carrier admission is projection from Definition 3380.1. The rows S, M, D display the finite stream windows, the explicit Cauchy modulus, and the dyadic tolerance ledger before any completion-facing read is available.

The diagonal limit handoff is then assembled in R. The row R reads the retained windows through the modulus thresholds and dyadic budgets, so the construction is a displayed regular rational readback rather than a quotient of streams or a selected host limit.

Classifier stability follows from H and C. Componentwise hsame transport preserves the same window, modulus, dyadic, readback, seal, provenance, and naming rows, while displayed Cont replay keeps the route ordered as $S, M, D \to R \to L \to E$.

The ledger rows P, N expose no extra coordinate. Since quotient streams, selected limits, countable choice, open-cover compactness, host convergence, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, every accepted completion read remains inside the displayed finite route. □

Theorem 3380.3 (Real Cauchy completeness seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The read therefore supplies a finite Cauchy-completeness handoff to real-name consumers without adding quotient stream equality, a selected limit, countable choice, or the claim that $\mathsf{Real}^{\uparrow}$ *is already complete.*

Proof. Start with an accepted carrier $K = (S, M, D, R, L, E, H, C, P, N)$. By the carrier definition, the only real-facing endpoint is E, and E is downstream of the limit-seal row L.

The row L consumes the diagonal readback R. That readback is itself downstream of the finite stream schedule S, the explicit modulus M, and the dyadic ledger D.

Thus any $\mathsf{Real}^{\uparrow}$ consumer reaches the seal only through S, M, D, R, L, E and the structural rows H, C, P, N. The excluded host-side data have no carrier coordinate, so they cannot be introduced by the handoff. □

Theorem 3380.4 (Real Cauchy completeness uniform-tail seal)**.** *Let* $K = (S, M, D, R, L, E, H, C, P, N)$ *be an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier. Suppose a consumer reads K through a* $\mathsf{UniformConvergence}^{\uparrow}$

shared threshold whose tail schedule dominates the explicit Cauchy-modulus row M and whose Cauchy-facing handoff is served by the $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ *route of chapter 3876. Then every* $\mathsf{Real}^{\uparrow}$ *endpoint read factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, the* $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ *handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L before the Real seal E is exposed.*

Proof. Project the carrier rows from Definition 3380.1. The only endpoint row is E, and Definition 3380.3 places E downstream of S, M, D, R, L.

The shared threshold is read through the $\mathsf{UniformConvergence}^{\uparrow}$ tail surface of chapter 3417. The dominance assumption says that this tail schedule cannot be consumed before it covers the explicit modulus row M already present in K.

The Cauchy-facing handoff is then the $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ route. It binds the same uniform tail row to a finite Cauchy comparison before the regular readback R and the limit-seal row L are available to the endpoint consumer.

Thus the row order is S, M, D first, then the shared uniform tail and Cauchy-criterion handoff, then R, L, E. Since the carrier has no quotient coordinate, choice coordinate, or second endpoint row, every $\mathsf{Real}^{\uparrow}$ read follows this displayed route. □

Theorem 3380.5 (Real Cauchy completeness filtered minimality)**.** *Let $K = (S, M, D, R, L, E, H, C, P, N)$ be an accepted* $\mathsf{RealCauchyCompleteness}^{\uparrow}$ *carrier. Every filtered or completion-service consumer that reads K through the* $\mathsf{BishopFilteredRegularCompletion}^{\uparrow}$ *route of chapter 8666 or the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *service boundary of chapter 1606 must preserve exactly the displayed* $\mathsf{StreamName}^{\uparrow}$ *row S, explicit modulus row M,* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row L,* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *row H, displayed* Cont *row C,* Pkg *row P, and local* $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ *row N before exporting a completion-facing Real read. Replacing this route by quotient streams, selected limits, countable choice, host convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is outside the carrier.*

Proof. Project the inventory of rows from Definition 3380.1. It gives the consumer exactly $S, M, D, R, L, E, H, C, P,$ with no coordinate for quotient stream equality, a chosen limit, countable choice, host convergence, or an ambient completeness theorem.

The terminal Real read is constrained by Definition 3380.3. Any endpoint must pass through the displayed stream windows, dyadic tolerance ledger, regular readback, and limit-seal handoff before reaching E.

For filtered consumers, Definition 8666.3 supplies the finite synchronization principle. Its window induction reads the filter and regular Cauchy rows against the same dyadic tolerances and preserves the shared transport, replay, provenance, and local naming rows, so the filtered route cannot bypass $S, M, D, R, L, E, H, C, P, N$.

For completion-service consumers, the $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ chapter records the least displayed service boundary for completion reads, with Lean touchpoint

BEDC.Derived.CauchyCompletionMinimalityUp.CauchyCompletionMinimalityTasteGate_single_carrier_alignm

Reading K through that boundary therefore preserves the Real Cauchy route as the service-facing input rather than adding an external completion principle. □

Falsifiable prediction: Any accepted $\mathsf{RealCauchyCompleteness}^{\uparrow}$ consumer reaching a $\mathsf{Real}^{\uparrow}$ read without the displayed StreamName windows, explicit Cauchy modulus, DyadicRatCore ledger, RegSeqRat diagonal readback, and CauchyLimitSeal row refutes the carrier.

Independence witness: DyadicRatCore, StreamName, RegSeqRat, CauchyLimitSeal, and Real are not individually bijective with $\mathsf{RealCauchyCompleteness}^{\uparrow}$ because the packet carries the ordered completion route across all five surfaces together with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RealCauchyCompleteness}^{\uparrow}$ packet is generated from finite StreamName windows, an explicit Cauchy-modulus row, a DyadicRatCore tolerance ledger, a RegSeqRat diagonal readback, a CauchyLimitSeal handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3380.1, Definition 3380.2, Definition 3380.3, Definition 3380.4, and Definition 3380.5 seed the finite constructive Cauchy-completeness route through StreamName, RegSeqRat, DyadicRatCore, CauchyLimitSeal, UniformConvergence, UniformConvergenceCauchyCriterion, filtered completion, minimal completion service, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyCompleteness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealCauchyCompletenessUp.RealCauchyCompletenessTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, host convergence, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, explicit Cauchy-modulus exposure, diagonal limit handoff, classifier stability, and ledger non-escape for BEDC.Derived.RealCauchyCompletenessUp.

Chapter 3381

A Concrete Naming Certificate for ClosedLamDomainSubjectReduction$^{\uparrow}$

The ClosedLamDomainSubjectReduction$^{\uparrow}$ packet realizes the lambda-domain branch of the closed MetaCIC subject-reduction frontier as a finite NameCert carrier. It sits beside the four-row discharge socket of chapter 719, the discharge ledger of chapter 720, the beta-substitution packet of chapter 716, the closed substitution boundary of chapter 538, the typed beta congruence trigger of chapter 760, and the Pi-domain sibling of chapter 3352. The packet names the lambda-domain obligation as displayed closed term data; it does not export an unconditional closed subject-reduction theorem.

Definition 3381.1 (ClosedLamDomainSubjectReduction carrier)**.** *A* ClosedLamDomainSubjectReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, D, D', B, Y, Z, G, H, R, Q, N)$$

with the following displayed rows:

1. *T is the displayed closed lambda term row, naming the source term $\lambda D.\, B$ and its closed context boundary.*
2. *D is the original domain row inspected before the congruence step.*
3. *D' is the reduced domain row reached by the displayed beta step from D.*
4. *B is the body row, read under the binder induced by the displayed domain.*
5. *Y is the typing row, recording the source typing judgement for $\lambda D.\, B$ and the target typing judgement demanded for $\lambda D'.\, B$.*
6. *Z is the closedness row, requiring the lambda term, original domain, reduced domain, and body reads to remain inside the same closed MetaCIC boundary.*
7. *G is the beta-step and typed congruence row, consuming the displayed lambda-domain beta step and the trigger surface of chapter 760.*
8. *H records componentwise* hsame *transport for $T, D, D', B, Y, Z, G, R, Q, N$.*
9. *R records displayed* Cont *replay from the source lambda term through the domain step, typing row, closedness row, and subject-reduction handoff.*

10. *Q is the* Pkg *provenance over chapter 719, chapter 720, chapter 716, chapter 538, chapter 760, chapter 3352,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying the closed lambda term, both domain rows, the body row, the typing row, the closedness row, the lambda-domain beta step, transport, replay, provenance, and local naming. It has no coordinate for a blanket closed subject-reduction theorem, a host conversion rule, erased derivation semantics, quotient syntax equality, or an untracked binder context.

Theorem 3381.2 (ClosedLamDomainSubjectReduction obligations)**.** *For every accepted* ClosedLamDomainSubject *carrier $L = (T, D, D', B, Y, Z, G, H, R, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed closed lambda term, original domain, reduced domain, body row, typing row, closedness row, beta-step row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The lambda-domain preservation read is accepted only when G displays the domain beta step and Y, Z display the corresponding typing and closedness rows.*

Proof. Project the carrier of Definition 3381.1. The rows T, D, D', B display the lambda term, the two domain readings, and the body read, while Y and Z display the typing and closedness boundaries attached to the same context.

The beta-step row G is then read as the lambda-domain congruence trigger. It is not interchangeable with the beta-substitution row, the application-argument row, or the Pi-domain row of the subject-reduction socket, because the target branch changes the displayed lambda domain while preserving the body under the induced binder.

The structural rows H, R, Q, N transport, replay, source, and name only that finite packet. Since no row stores a host conversion rule, erased derivation semantics, quotient syntax equality, or a closed subject-reduction theorem, the displayed rows exhaust the local NameCert obligations. □

Theorem 3381.3 (ClosedLamDomainSubjectReduction handoff)**.** *Every subject-reduction discharge ledger read that consumes the lambda-domain branch from an accepted* $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ *packet factors through the displayed route*

$$\lambda D.\, B,\ D,\ D',\ B,\ Y,\ Z,\ G,\ H,\ R,\ Q,\ N.$$

The handoff supplies the lambda-domain obligation row to the subject-reduction discharge socket and ledger; it does not supply the beta-substitution row, application-argument row, Pi-domain row, or an unconditional closed subject-reduction theorem.

Proof. Start with a subject-reduction consumer of the lambda-domain branch. The socket and ledger chapters, chapter 719 and chapter 720, require a distinct lambda-domain row before the parameterised subject-reduction route can be read.

The present carrier supplies that row only through T, D, D', B, Y, Z, G. These rows display the source lambda term, the original and reduced domains, the body, the typing judgement, the closedness boundary, and the beta-step trigger.

Finally H, R, Q, N preserve and name the same finite handoff. The beta-substitution preservation packet, closed substitution boundary, typed beta congruence trigger, and Pi-domain packet remain sibling inputs rather than hidden substitute proofs, so the handoff is exactly the lambda-domain row and nothing stronger. □

Falsifiable prediction: Any accepted $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ consumer reaching a subject-reduction conclusion without the displayed lambda-domain beta step and typing rows refutes the carrier.

Independence witness: ClosedPiDomainSubjectReduction, BetaSubstitutionPreservation, ClosedSubstitutionBoundary, and TypedBetaCongruenceTrigger are not bijective with $\mathsf{ClosedLamDomainSubjectReduction}$ because this packet jointly carries the closed lambda term, original domain, reduced domain, body row, typing row, closedness row, lambda-domain beta step, transport, replay, provenance, and local naming.

The Lean TasteGate file for $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ ClosedLamDom}$ and $\mathsf{FieldFaithful\ ClosedLamDomainSubjectReduction}^{\uparrow}$ instances preserving $T, D, D', B, Y, Z, G, H, R, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}$ packet is generated from a closed lambda term row, original domain row, reduced domain row, body row, typing row, closedness row, beta-step row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3381.1, Definition 3381.2, and Definition 3381.3 seed the finite lambda-domain subject-reduction obligation packet for the closed MetaCIC discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedLamDomainSubjectReduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close beta-substitution preservation, application-argument stability, Pi-domain preservation, unconditional closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lambda-domain beta-step exactness, typing-row exposure, closedness-row exposure, subject-reduction ledger handoff, and TasteGate field faithfulness for BEDC.Derived.ClosedLamDomainSub

Chapter 3382

A Concrete Naming Certificate for CauchyCompletenessCriterion$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyCompletenessCriterion$^{\uparrow}$ packet records the finite Cauchy-criterion route by which a metric source, an explicit Cauchy modulus, named stream windows, regular rational readback, dyadic tolerance accounting, and a terminal Real$^{\uparrow}$ seal are presented to completion-facing consumers. It sits over the metric completion surface of chapter 1506, the regular rational rows of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet gives later completion chapters a single criterion object to cite instead of restating the Cauchy-modulus handoff locally.

Definition 3382.1 (CauchyCompletenessCriterion carrier)**.** *A* CauchyCompletenessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, \mu, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, carrying the named carrier, distance observations, and completion-facing request inherited by the metric completion route.*
2. *μ is the explicit Cauchy modulus row, displayed as finite threshold data for each inspected tolerance request.*
3. *S is the* StreamName$^{\uparrow}$ *window family naming the finite tail observations read from the source stream and the modulus thresholds.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from those windows.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger recording the rational budget, tail slack, and comparison bounds used by the readback.*
6. *E is the* Real$^{\uparrow}$ *seal row, visible only after M, μ, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, modulus, windows, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing Cauchy request through $M, \mu, S, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* CauchyCompletenessCriterion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the ten displayed rows. It has no coordinate for quotient completions, selected limits, countable-choice schedules, trichotomy decisions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessCriterionUp.CauchyCompletenessCriterionTasteGate single carrier alignment`

Theorem 3382.2 (CauchyCompletenessCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletenessC}$ *carrier* $Q = (M, \mu, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, explicit Cauchy modulus, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must consume* M, μ, S, R, D *before it can inspect* E, H, C, P, N*.*

Proof. Project the carrier of Definition 3382.1. The rows M and μ expose the metric source and the Cauchy modulus as finite packet data.

Read the observation rows next. The row S stores exactly the inspected tail windows, R records the regular rational readback from those windows, and D stores the dyadic tolerance budget attached to the same finite observations.

Finally inspect the seal and structural rows. Since E, H, C, P, N transport, replay, source, and name the already displayed route, the certificate has no remaining coordinate from which a quotient completion, selected limit, countable-choice schedule, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem could be exported. □

Theorem 3382.3 (CauchyCompletenessCriterion Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or metric-completion consumer citing an accepted* $\mathsf{CauchyCompletenessCriterion}^{\uparrow}$ *packet receives only the finite route*

$$M, \mu, S, R, D, E, H, C, P, N.$$

The handoff first displays the metric source and Cauchy modulus, then the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then the* $\mathsf{Real}^{\uparrow}$ *seal together with transport, replay, provenance, and local naming.*

Proof. Begin with an accepted carrier and apply Definition 3382.2. The consumer-facing data are exhausted by the displayed rows $M, \mu, S, R, D, E, H, C, P, N$.

The criterion route is ordered. A completion-facing read must first expose the metric source and explicit modulus, then read finite stream windows and regular rational observations, and then consume the dyadic tolerance ledger.

The Real seal is terminal with respect to this route. Because quotient completions, selected limits, countable-choice schedules, trichotomy decisions, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, every exported handoff is the displayed finite criterion packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessCriterion}^{\uparrow}$ packet is generated from a metric source row, explicit Cauchy modulus, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3382.1, Definition 3382.2, and Definition 3382.3 seed the Bishop-style Cauchy criterion handoff for Real and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletenessCriterion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyCompletenessCriterionUp.CauchyCompletenessCriterionTasteGate\_single\_carrier\_al`
Bridge status: none
Not claimed: This chapter does not close quotient completions, selected limits, countable choice, trichotomy, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, finite window coverage, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.CauchyCompletenessCriterionUp.

Chapter 3383

A Concrete Naming Certificate for $\mathsf{TailModulusLimitHandoff}^{\uparrow}$

Following the Bishop-style diagonal Cauchy proof route in section 9238.1, the $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ packet records the finite tail-modulus handoff used by the diagonal regularity theorem of Definition 9238.4. It reads the finite windows of chapter 126, the regular rational rows of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet gives Real-completion proof targets a concrete landing for the shared tail threshold consumed by diagonal-limit consumers.

Definition 3383.1 (TailModulusLimitHandoff carrier)**.** *A* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\mu, \tau, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. μ *is the Cauchy modulus row for the regular rational sequence of regular rational sequences.*
2. τ *is the finite tail selector row naming the shared threshold used by the diagonal limit read.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected at the selected tail threshold.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the diagonal representative and the two tail observations compared by the proof.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the endpoint legs, middle modulus leg, and rational slack used by the diagonal estimate.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* μ, τ, S, R, D *have been displayed.*
7. H *records componentwise* hsame *transport for the modulus, tail selector, windows, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a diagonal-limit request through* $\mu, \tau, S, R, D, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient streams, selected limits, countable-choice schedules, trichotomy decisions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.TailModulusLimitHandoffUp.TailModulusLimitHandoffTasteGate single carrier alignment`

Theorem 3383.2 (TailModulusLimitHandoff NameCert obligations)**.** *For every accepted* TailModulusLimitHandoff *carrier* $T = (\mu, \tau, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy modulus, finite tail selector,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The diagonal-limit consumer must read* μ, τ, S, R, D *before it can export a Real-facing seal through* E, H, C, P, N*.*

Proof. Project the carrier of Definition 3383.1. The rows μ and τ display the modulus and the shared tail threshold used by the diagonal-limit proof.

Read the finite observations. The row S names the inspected windows, R records the regular rational readback, and D stores the dyadic tolerance ledger for the endpoint legs and the middle modulus leg.

The seal and structural rows add no analytic coordinate. The rows E, H, C, P, N seal, transport, replay, source, and name the displayed finite route, so quotient streams, selected limits, countable-choice schedules, trichotomy decisions, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the certificate. □

Theorem 3383.3 (TailModulusLimitHandoff diagonal handoff)**.** *Every Real-completion consumer of an accepted* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *packet receives the finite diagonal route*

$$\mu, \tau, S, R, D, E, H, C, P, N.$$

The handoff supplies the Cauchy modulus, shared tail selector, selected $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming needed by the diagonal regularity route of Definition 9238.4.*

Proof. Start with a Real-completion consumer. By Definition 3383.2, the consumer first reads the modulus and tail selector, then the finite windows, readback row, and dyadic tolerance ledger.

The diagonal route is therefore finite before the Real seal is visible. The seal row E depends on the displayed rows μ, τ, S, R, D, and the structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming.

Since no carrier coordinate stores quotient streams, selected limits, countable-choice schedules, trichotomy decisions, or ambient completeness, every exported diagonal handoff is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ packet is generated from a Cauchy modulus row, finite tail selector, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 3383.1, Definition 3383.2, and Definition 3383.3 seed the finite tail-modulus handoff used by the diagonal regularity proof route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TailModulusLimitHandoff}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.TailModulusLimitHandoffUp.TailModulusLimitHandoffTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail selector exactness, finite window coverage, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.TailModulusLimitHandoffUp.

Chapter 3384

A Concrete Naming Certificate for $\mathsf{BorelSet}^{\uparrow}$

Following Bourbaki and Bishop–Bridges topology, the $\mathsf{BorelSet}^{\uparrow}$ packet names the finite boundary by which a topological or metric real-facing source exposes a generated set predicate without importing measure theory. It sits after the open-neighbourhood interface of chapter 88, the metric source of chapter 57, the terminal real-name seal of chapter 38, the finite observation route of chapter 126, and the regular rational readback surface of chapter 301. The packet records generator admission, finite constructor replay, and sibling provenance; it does not claim a measure, sigma-additivity, choice-selected codes, or classical completeness of the full Borel hierarchy.

Definition 3384.1 (Borel set carrier)**.** *A* $\mathsf{BorelSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, O, M, G, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-space row read either through the* $\mathsf{Topology}^{\uparrow}$ *interface of chapter 88 or through a metric presentation supplied by chapter 57.*
2. *O is the open-generator ledger naming basic open rows visible to the source topology.*
3. *M is the real-facing membership query surface, routed through* $\mathsf{Real}^{\uparrow}$ *names from chapter 38 when the source is a real or metric-real object.*
4. *G is the finite generator tree recording the displayed complements, finite unions, and finite intersections that the current consumer is allowed to replay.*
5. *R is the readback row tying finite membership observations to* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational names through chapter 126 and chapter 301.*
6. *H stores componentwise* hsame *transport for the source, generator, membership, readback, continuation, provenance, and local naming rows.*
7. *C stores the displayed* Cont *replay from a source query through open generators, finite constructor rows, and readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, O, M, G, R, H, C, P, N$. It has no coordinate for measure values, countable additivity, a choice-selected code for an arbitrary hierarchy branch, or a completed classical Borel universe.

Lean4 ✓ `BEDC.Derived.BorelSetUp.BorelSetTasteGate single carrier alignment`

Theorem 3384.2 (Borel set NameCert obligations)**.** *For the carrier of Definition 3384.1, the local* $\mathsf{NameCert}_{\mathsf{BorelSet}^{\uparrow}}$ *surface consists exactly of source-space admission, open-generator admission, real-facing membership queries when the source is real or metric-real, finite generator-tree replay, StreamName and RegSeqRat readback, componentwise transport, displayed continuation replay, package provenance, and local naming. Every accepted Borel-set query must read the source topology or metric row before it reads O, G, R, H, C, P, N.*

Proof. Project the carrier $B = (X, O, M, G, R, H, C, P, N)$. The source row X and open-generator ledger O are the only rows that introduce topological information.

Read the finite constructor tree. Complements, finite unions, and finite intersections are replayed only through G, and the membership-facing observations are tied to M and R before a consumer can compare two packets.

The structural rows add no measure-theoretic content. The rows H, C, P, N transport, replay, source, and name the displayed route, so the listed rows exhaust the seed NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.BorelSetUp.BorelSetCarrier namecert obligations`

Theorem 3384.3 (Borel set generator stability)**.** *Let $B = (X, O, M, G, R, H, C, P, N)$ and*

$$B' = (X', O', M', G', R', H', C', P', N')$$

be accepted $\mathsf{BorelSet}^{\uparrow}$ *carriers. If the displayed source row, open-generator ledger, finite generator tree, membership surface, and readback row agree componentwise through the packet's* hsame *rows, then every topology-facing, metric-facing, or real-facing consumer sees the same generated set boundary from B and B'. The comparison uses only the rows $X, O, M, G, R, H, C, P, N$ and $X', O', M', G', R', H', C', P', N'$.*

Proof. Start with the NameCert obligations. Any consumer read first passes through the source-space row and open-generator ledger, then through the finite generator tree and readback rows.

Apply the displayed componentwise transports. The hypotheses identify the rows that determine the generator route, while H', C', P', N' preserve, replay, source, and name the same finite route for B'.

Thus the consumer-visible boundary is stable under the stated transport. No measure value, countable-additivity proof, hierarchy-completeness object, or choice-selected branch can refine the comparison, because none is a coordinate of either carrier. □

Falsifiable prediction: A $\mathsf{BorelSet}^{\uparrow}$ consumer that reaches a measure value, a countable-additivity row, or a choice-selected hierarchy code from the displayed packet refutes the carrier surface.

Independence witness: topology, metric, real, streamname, regseqrat: no listed sibling supplies the whole nine-row packet $B = (X, O, M, G, R, H, C, P, N)$, because each sibling contributes only source topology, metric control, real-name sealing, finite observation, or rational readback data while this chapter binds the generated-set boundary with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3384.1, Definition 3384.2, and Definition 3384.3 seed the finite generated-set boundary over topology, metric, real-name, StreamName, RegSeqRat, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure theory, sigma-additivity, countable choice, choice-selected Borel codes, classical hierarchy completeness, open-cover compactness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, generator stability, readback preservation, transport, and ledger-refusal proofs for BEDC.Derived.BorelSetUp.

Constructive story: A $\mathsf{BorelSet}^{\uparrow}$ packet is generated from a topology or metric source row, an open-generator ledger, real-facing membership reads, a finite generator tree, StreamName and RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3385

A Concrete Naming Certificate for $\mathsf{OuterMeasure}^{\uparrow}$

Following Caratheodory and Bishop–Bridges constructive measure theory, the $\mathsf{OuterMeasure}^{\uparrow}$ packet records the finite cover-valued boundary from which measurable-event and measure rows can later be read. It sits between the generated-set surface of chapter 3384, the membership discipline of chapter 3668, the real-name endpoint seal of chapter 38, the finite observation windows of chapter 126, and the measure carrier of chapter 92. The packet names only displayed set rows, empty-set valuation, monotonicity comparisons, finite cover subadditivity, Caratheodory-measurability boundary tests, transport, replay, provenance, and local naming; it does not assert a completed measure space, a chosen sigma-algebra closure, countable choice of covers, quotient equality of subsets, or ambient measure completion.

Definition 3385.1 (OuterMeasure carrier)**.** *An* $\mathsf{OuterMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, S, Z, M, F, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the source object whose displayed subsets are read through* $\mathsf{SetLike}^{\uparrow}$ *and, when the source is real-facing, through the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set boundary.*
2. *S is the displayed subset-family row naming the finite set requests visible to the packet.*
3. *Z is the empty-set value row, comparing the outer value of the displayed empty subset with the zero endpoint on the* $\mathsf{Real}^{\uparrow}$ *surface.*
4. *M is the monotonicity row, recording a finite inclusion witness $A \subseteq B$ together with the comparison $\mu^*(A) \leq \mu^*(B)$ on the displayed real endpoints.*
5. *F is the finite displayed cover subadditivity row, assigning a finite cover list $U_0, \ldots, U_k$ of a displayed subset A to the visible comparison*

$$\mu^*(A) \leq \mu^*(U_0) + \cdots + \mu^*(U_k).$$

6. *B is the Caratheodory-measurability boundary row: for each displayed test set T and candidate measurable set E, it records the two-sided split comparison between $\mu^*(T)$ and the displayed sum of the inside and outside rows $T \cap E$ and $T \setminus E$.*

7. *H stores componentwise* hsame *transport for the source row, subset-family row, value rows, inclusion and finite-cover ledgers, boundary row, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a subset request through empty-set value, monotonicity, finite cover subadditivity, boundary testing, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, Z, M, F, B, H, C, P, N$. It has no coordinate for a completed measure, a host sigma-algebra, a selected cover family, quotient subset equality, or a hidden countable-additivity row.

Lean4 ✓ `BEDC.Derived.OuterMeasureUp.OuterMeasureTasteGate single carrier alignment`

Theorem 3385.2 (OuterMeasure NameCert obligations)**.** *For the carrier of Definition 3385.1, the local* $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source and subset-family rows, empty-set zero readback, monotonicity under displayed inclusion, finite displayed cover subadditivity, Caratheodory-measurability boundary testing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted outer-measure consumer must read the source subset through S, expose value comparisons through Z, M, F, B, and replay those rows through H, C, P, N before any* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *sibling row is exposed.*

Proof. Project the carrier $O = (X, S, Z, M, F, B, H, C, P, N)$. The source row X and subset-family row S are the only admitted entry points for displayed set requests, so carrier admission is exhausted by those rows.

Read the value ledgers. The row Z supplies the empty-set zero comparison, M supplies monotonicity under a displayed inclusion witness, and F supplies the finite cover subadditivity comparison for a named finite cover list.

The boundary row B is the only place where a Caratheodory-measurability test is visible. It records the split comparison for displayed test sets and candidates, but it does not choose a completed sigma-algebra or assert ambient measure completion.

Finally replay the structural rows. Componentwise transport through H, continuation replay through C, provenance through P, and local naming through N preserve the same finite route. Since no further coordinate is present, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3385.3 (OuterMeasure finite premeasure route)**.** *Every* $\mathsf{OuterMeasure}^{\uparrow}$ *packet exports a finite pre-measure route*

$$S \longmapsto Z, M, F, B \longmapsto H, C, P, N$$

that downstream $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, *and* $\mathsf{MeasurableFunction}^{\uparrow}$ *consumers may cite before reading measure-completion language. The route exposes empty-set zero, displayed inclusion monotonicity, finite cover subadditivity, and Caratheodory boundary rows, not a host measure space or a choice-selected completion.*

Proof. Start with an accepted subset request in S. The empty, inclusion, finite cover, and boundary rows are the only value-bearing rows available to the packet.

The finite cover row F keeps the route on a displayed list of cover sets, while the boundary row B tests only the named split of a displayed test set by a candidate measurable set. These two rows provide the pre-measure boundary consumed by later measure-facing chapters.

Replay through H, C, P, N. The provenance row records the allowed sibling surfaces, and the continuation row exposes them in the same order. Thus a downstream consumer can cite the finite pre-measure route without importing completed measure language or selected cover choices. □

Falsifiable prediction: An $\mathsf{OuterMeasure}^{\uparrow}$ consumer that obtains a completed measure, a chosen sigma-algebra closure, countable choice of covers, quotient equality of subsets, or a $\mathsf{Measure}^{\uparrow}$ countable-additivity row without first reading $S, Z, M, F, B, H, C, P, N$ refutes the carrier surface.

Independence witness: measure, borelset, setlike, real, streamname: no listed sibling supplies the whole ten-row packet $O = (X, S, Z, M, F, B, H, C, P, N)$, because each sibling contributes only measure values, generated-set membership, set-like membership, real endpoints, or finite observation windows while this chapter binds outer valuation, finite cover subadditivity, boundary testing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3385.1, Definition 3385.2, and Definition 3385.3 seed the finite outer-measure surface over displayed subsets, empty-set value, inclusion monotonicity, finite covers, Caratheodory boundary rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OuterMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed measure spaces, chosen sigma-algebra closures, countable choice of covers, quotient subset equality, MeasureUp countable additivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, empty-set zero, monotonicity, finite cover subadditivity, Caratheodory boundary testing, transport, provenance, and no-escape for BEDC.Derived.OuterMeasureUp.

Constructive story: An $\mathsf{OuterMeasure}^{\uparrow}$ packet is generated from a displayed source row, subset-family row, empty-set value row, inclusion monotonicity row, finite cover subadditivity row, Caratheodory boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{LocallyConvexSpace}^{\uparrow}$ reads the vector-space, topological-vector, seminormed-space, and functional-analysis sibling rows through chapter 47, chapter 205, chapter 5868, and chapter 94 before the child files route the finite local-convexity carrier.

Chapter 3386

A Concrete Naming Certificate for $\mathsf{LocallyConvexSpace}^{\uparrow}$

Following Bourbaki and Treves on locally convex topological vector spaces, the $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet records the finite BEDC surface where a vector-space carrier, a topological-vector-space carrier, a finite seminorm ledger, and a convex balanced neighbourhood ledger are read before any functional-analysis consumer can cite local convexity. It sits beside the vector-space source of chapter 47, the topological-vector-space source of chapter 205, the norm comparison surface of chapter 89, and the functional-analysis consumer frontier of chapter 94. The packet names the locally convex carrier only through displayed finite rows; it is not a Hahn–Banach principle, not a choice-selected dual, and not an ambient weak-star compactness theorem.

The seminorm and convex-balanced-neighbourhood ledgers consume the sibling rows in a fixed order: chapter 47 supplies the algebraic source, chapter 205 supplies the topological-vector continuity route, chapter 89 supplies the optional norm comparison row, and chapter 94 supplies the downstream consumer boundary. These links keep local convexity as a displayed finite dependency surface rather than an orphan analytic predicate.

Downstream $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumers use $\mathsf{LocallyConvexSpace}^{\uparrow}$ only through the finite seminorm ledger and the convex-balanced-neighbourhood ledger displayed here. The consumer chapter of chapter 94 may cite local convexity only after those two rows have been exposed with the vector-space, topological-vector-space, transport, replay, provenance, and local naming rows.

Definition 3386.1 (Locally convex space carrier)**.** *A* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (V, T, S, B, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, including the additive carrier, scalar action, and vector-space laws used by downstream linear consumers.*
2. *T is the* $\mathsf{TopVecSpace}^{\uparrow}$ *source row, recording the topology and the displayed continuity of addition and scalar multiplication.*
3. *S is the finite seminorm-family ledger, listing the seminorm names, their vector-space source, triangle and scalar rows, and the generated neighbourhood probes accepted in the packet.*
4. *B is the convex balanced neighbourhood ledger, recording the finite neighbourhood rows certified as convex, balanced, and compatible with the topology read through T.*

5. *M is the* $\mathsf{Norm}^{\uparrow}$ *comparison row, used only when a named normed source is present and only as a comparison into the seminorm ledger rather than as a replacement for local convexity.*
6. *H records componentwise* hsame *transport for V, T, S, B, M and the local naming row.*
7. *C records displayed* Cont *replay through the vector source, topological-vector-space source, seminorm ledger, convex balanced neighbourhood ledger, norm comparison, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, T, S, B, M, H, C, P, N$. It contains no coordinate for an arbitrary dual selector, a host locally convex topology predicate, an unrestricted separation theorem, weak-star compactness, quotient completion, or a global choice principle.

Theorem 3386.2 (Locally convex space NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier $L = (V, T, S, B, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *surface consists exactly of vector-space source exposure through V; topological-vector-space source exposure through T; finite seminorm-family evidence through S; convex balanced neighbourhood evidence through B; optional norm comparison through M; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.LocallyConvexSpaceUp.LocallyConvexSpaceCarrier namecert obligations`

Proof. Project the accepted carrier in Definition 3386.1. The rows $V, T, S, B, M, H, C, P, N$ are exactly the vector, topology, seminorm, neighbourhood, norm-comparison, transport, replay, provenance, and naming coordinates.

The vector and topological-vector rows are consumed first. They supply the additive carrier, scalar action, topology, and continuity obligations before any seminorm or neighbourhood evidence can be read.

The local-convexity evidence is then split across S and B. The seminorm ledger records finite probe families, while the neighbourhood ledger records the convex balanced neighbourhood rows generated from those probes.

Finally M, H, C, P, N compare, transport, replay, source, and name the same finite packet. Since no dual selector, separation theorem, weak-star compactness row, or host topology predicate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3386.3 (Locally convex space seminorm-filter stability)**.** *Let $L = (V, T, S, B, M, H, C, P, N)$ be an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier. Any consumer that reads a finite seminorm probe from S and then passes to the neighbourhood filter generated by B must still factor through the same rows $V, T, S, B, M, H, C, P, N$. The handoff exports no candidate-free normed-space structure, no selected continuous dual, and no ambient weak topology beyond the displayed finite filter rows.*

Proof. Begin with a seminorm probe in S. Its vector-space source is the same V read by the carrier, and its topological compatibility is checked against the same T used to admit the packet.

Pass from the probe to the neighbourhood side through B. The rows in B record convexity, balancedness, and filter membership as displayed finite evidence, so the handoff remains inside the packet rather than introducing a host filter object.

The comparison row M may relate the probe to a norm row, but it is only a comparison coordinate. Transport and replay through H and C, with provenance and naming through P

and N, preserve the same route. Thus any accepted seminorm-filter read remains mediated by $V, T, S, B, M, H, C, P, N$ and cannot export a selected dual or ambient weak topology. □

Theorem 3386.4 (Locally convex space seminorm-family obligation)**.** *Let $L = (V, T, S, B, M, H, C, P, N)$ be an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier. Any accepted consumer that reads the finite seminorm family S must expose the seminormed-space comparison row of chapter 5868 before using those probes. Any weak-star-facing consumer may pass through chapter 5192 only after the displayed seminorm, convex-balanced neighbourhood, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows S, B, H, C, P, N have been read together with the source rows V, T.*

Proof. Project the accepted carrier $L = (V, T, S, B, M, H, C, P, N)$. The row S is the finite seminorm-family ledger, while V and T are the vector-space and topological-vector-space sources that make each seminorm probe admissible.

Route S through the seminormed-space sibling named in chapter 5868. That sibling supplies the comparison surface for finite family indices, zero, triangle, scalar, and induced-neighbourhood reads before a local-convexity consumer may use the probes as neighbourhood data.

For weak-star-facing consumers, keep the route downstream of the local convexity packet. The weak-star topology chapter can be cited only after S, B, H, C, P, N and the sources V, T are displayed, so the weak-star use is a finite observation route over the local-convexity packet rather than a selected-dual principle. □

Theorem 3386.5 (Locally convex space topological-vector seminorm route)**.** *Every local-convex consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *carrier factors first through the topological-vector-space source of chapter 205, the vector-space source of chapter 47, and the norm-comparison surface of chapter 89. Only after those sibling rows have been displayed may a functional-analysis consumer read the finite seminorm ledger S and the convex balanced neighbourhood ledger B as local convexity evidence.*

Proof. Project the accepted carrier $L = (V, T, S, B, M, H, C, P, N)$. The rows T, V, and M are exactly the displayed topological-vector, vector, and norm-comparison coordinates named by the sibling chapters.

The seminorm and neighbourhood rows do not replace those siblings. They are read after V and T fix the algebraic and topological source, and after M records whether a norm comparison is present.

Thus a functional-analysis consumer reaches local convexity only through the ordered route $T, V, M, S, B, H, C, P, N$. No Hahn–Banach principle, selected dual, ambient weak topology, or host locally convex predicate is exported by this sibling handoff. □

Theorem 3386.6 (Locally convex space forward citation)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that cites an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet for seminorm or convex-balanced-neighbourhood data must first read the sibling rows named in the carrier route:* $\mathsf{VecSpace}^{\uparrow}$ *for algebraic source,* $\mathsf{TopVecSpace}^{\uparrow}$ *for topological-vector continuity,* $\mathsf{Norm}^{\uparrow}$ *for optional comparison, and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *for the downstream consumer boundary. The local convexity citation is therefore a horizontal dependency through those displayed rows, not an isolated analytic predicate.*

Proof. Read the carrier of Definition 3386.1. Its rows V, T, M are the vector-space, topological-vector-space, and norm-comparison coordinates, while S and B are the seminorm and convex-balanced-neighbourhood ledgers consumed downstream.

The functional-analysis endpoint is named by the provenance and consumer boundary already recorded in the chapter opening. A consumer cannot detach S or B from the source rows that make them meaningful.

Thus every forward citation travels through $V, T, M, S, B, H, C, P, N$. The route records a real sibling dependency and exports no stand-alone local-convexity predicate outside the finite NameCert packet. □

Theorem 3386.7 (Locally convex space functional-analysis anchor)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must read the same seminorm family* S*, topological-vector route* T*, convex-balanced filter-neighbourhood rows* B*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row* N *before any functional-analytic closure is exposed. The consumer boundary of chapter 94 may cite local convexity only through this finite route, not through a detached locally convex topology predicate, selected dual, Hahn–Banach principle, or weak-star compactness theorem.*

Proof. Project the carrier in Definition 3386.1. The rows S, T, B, H, C, P, N are explicit coordinates of the accepted packet, with S naming the seminorm family, T naming the topological-vector source, and B naming the finite filter-neighbourhood evidence.

Apply the forward-citation theorem Definition 3386.6. It already forces functional-analysis consumers to pass through the vector, topological-vector, norm-comparison, seminorm, neighbourhood, transport, replay, provenance, and local naming rows.

The present theorem records the same route at the functional-analysis boundary. Since the carrier has no coordinate for a detached topology predicate, selected dual, separation principle, or weak-star compactness theorem, no functional-analytic closure can be exposed before those displayed rows are read. □

Falsifiable prediction: No accepted $\mathsf{LocallyConvexSpace}^{\uparrow}$ consumer can cite local convexity unless it first exposes the vector-space source, topological-vector-space source, finite seminorm-family ledger, convex balanced neighbourhood ledger, transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: topvecspace, vecspace, norm, functionalanalysis: none is bijective with $\mathsf{LocallyConvexSpace}^{\uparrow}$ because this carrier alone binds finite seminorm probes to convex balanced neighbourhood rows while retaining the vector, topology, comparison, transport, replay, provenance, and local NameCert coordinates.

The Banach–Alaoglu and weak-star compactness packets read $\mathsf{LocallyConvexSpace}^{\uparrow}$ through its displayed seminorm and topological-vector rows before any dual-ball compactness surface is exposed. In particular, chapter 3420 and chapter 6401 consume the locally convex source only after the seminorm-family route of chapter 5868 has made neighborhoods, bounded dual probes, and transport rows visible; neither compactness packet may treat $\mathsf{LocallyConvexSpace}^{\uparrow}$ as a hidden kernel premise detached from those finite rows.

Theorem 3386.8 (Locally convex Banach-seminorm route)**.** *Every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet that is consumed by a Banach-facing functional-analysis row must first expose the* $\mathsf{SeminormedSpace}^{\uparrow}$ *family route of chapter 5868 and the* $\mathsf{TopVecSpace}^{\uparrow}$ *linear-topology route of chapter 205. A* $\mathsf{Banach}^{\uparrow}$ *read from chapter 90 is then available only through displayed bounded completion, complete-metric limit, or bounded-linear consumer rows already carried by the downstream packet. It does not replace the locally convex carrier by a host Banach space, selected norm, selected dual, or ambient completeness principle.*

Proof. Project the accepted locally convex carrier through Definition 3386.1. The rows S and B provide the finite seminorm family and convex balanced neighbourhood ledger, while T provides the topological-vector operation-continuity surface.

Read the sibling sources next. The seminorm row is anchored by $\mathsf{SeminormedSpace}^{\uparrow}$, and the linear-topology row is anchored by $\mathsf{TopVecSpace}^{\uparrow}$; these rows are visible before any functional-analysis consumer may cite local convexity.

When a Banach-facing consumer is also present, the $\mathsf{Banach}^{\uparrow}$ row is downstream of those displayed routes. It contributes only the bounded completion or limit rows named by the consumer packet, so the locally convex read remains mediated by $V, T, S, B, M, H, C, P, N$ and cannot export a detached Banach carrier or selected norm. □

Theorem 3386.9 (Locally convex topological-vector dependency anchor)**.** *The* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *dependency surface ties every seminorm-family read to the* $\mathsf{TopVecSpace}^{\uparrow}$ *source before Banach completeness is consumed. Thus a downstream* $\mathsf{Banach}^{\uparrow}$ *or bounded-linear row may use locally convex data only after the topological-vector operations, seminorm family, convex balanced neighbourhood ledger,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows have all been read from the same finite carrier.*

Proof. Start with the topological-vector coordinate T. It supplies the vector addition and scalar multiplication continuity route through chapter 205, so every seminorm probe in S is interpreted against a displayed linear topology.

The neighbourhood ledger B is then read from the same carrier. Its convex balanced rows are generated from the visible seminorm and topological-vector data, rather than from an ambient locally convex host predicate.

Finally the structural rows H, C, P, N transport, replay, source, and name the route. Any Banach-facing consumer of chapter 90 receives only that already displayed route plus its own bounded completion or limit rows; the locally convex packet supplies no separate completeness, selected dual, or host equality coordinate. □

Theorem 3386.10 (Locally convex compact-regular-function dependency route)**.** *Every compact-regular real-function or functional-analysis consumer that cites the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the topological-vector-space source of chapter 205, the seminorm ledger, and the convex-balanced neighbourhood ledger from the source certificate. The route lets compact regular real-function packets use* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *as a source for functional-analysis neighbourhood control, while the exported rows remain* $V, T, S, B, M, H, C, P, N$ *and do not become a selected dual, an open-cover compactness proof, or an ambient host locally convex predicate.*

Proof. Project the source carrier of Definition 3386.1. The topological-vector-space row T is the displayed route through chapter 205, while S and B are the only seminorm and convex-balanced neighbourhood rows exposed to downstream consumers.

A compact-regular real-function packet may therefore cite the locally convex source only after those rows are visible. Functional-analysis consumers receive the same route through transport, replay, provenance, and local naming in H, C, P, N.

Since no row stores a selected dual, open-cover compactness proof, or host locally convex predicate, the dependency is a finite cross-reference route rather than an additional compactness or duality principle. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet supplies the finite seminorm and convex-neighbourhood rows consumed by the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495 before that carrier reaches its Baire-category handoff through chapter 3570 and chapter 3535.

Thus a $\mathsf{LocallyConvexSpace}^{\uparrow}$ read exports local convexity to the Frechet/Baire downstream lattice only through displayed seminorm, neighbourhood, transport, replay, provenance, and local naming rows; it exports no Hahn–Banach separation, infinite seminorm family, selected metric, or ambient completion.

Frechet dependency route. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet supplies the finite seminorm-family rows and convex balanced neighbourhood-filter rows consumed by the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495. The same read also consumes the $\mathsf{TopVecSpace}^{\uparrow}$ operation-continuity surface of chapter 205 and the $\mathsf{MetricCompletion}^{\uparrow}$ handoff of chapter 1506 before complete locally convex replay is available. Thus the dependency is horizontal: $\mathsf{LocallyConvexSpace}^{\uparrow}$ provides the local convexity evidence, while $\mathsf{FrechetSpace}^{\uparrow}$ repacks that evidence with topological-vector operations, metrizability, and completion-facing data without importing an ambient complete locally convex host, a selected metric, quotient completion equality, countable choice, or an infinite seminorm enumeration.

Frechet and topological-vector sibling lattice. The sibling lattice around $\mathsf{LocallyConvexSpace}^{\uparrow}$ has two mandatory dependency edges. A $\mathsf{FrechetSpace}^{\uparrow}$ read cites chapter 3495 for the metric-completion and metrizability rows, while a $\mathsf{TopVecSpace}^{\uparrow}$ read cites chapter 205 for the vector-space topology, addition-continuity, and scalar-continuity rows. A locally convex consumer may therefore export seminorm-family evidence only after those sibling rows have displayed the finite seminorm ledger, metric completion handoff, and vector-space topology surface; it does not bypass either sibling into an ambient locally convex host.

Frechet and topological-vector crosslink. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier depends on the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205 for the compatibility between vector addition, scalar multiplication, and the displayed topology before its seminorm and convex-balanced neighbourhood rows can be read as local convexity. The same packet feeds the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495 by adding countable seminorm-family, metrizable replay, and completion-facing constraints over that topological-vector surface. Thus the route from $\mathsf{TopVecSpace}^{\uparrow}$ through $\mathsf{LocallyConvexSpace}^{\uparrow}$ to $\mathsf{FrechetSpace}^{\uparrow}$ is a finite NameCert dependency chain, not a host-level assertion of an ambient locally convex complete vector space.

Theorem 3386.11 (Locally convex space forward lattice anchor)**.** *Every forward dependency read of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet factors through the displayed lattice*

$$\mathsf{VecSpace}^{\uparrow},\ \mathsf{TopVecSpace}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ S,\ B,\ H,\ C,\ P,\ N.$$

Here S is the finite seminorm ledger and B is the convex-balanced neighbourhood ledger. The sibling consumer of chapter 94 may cite local convexity only after these rows have been exposed, so the forward lattice records a finite NameCert dependency rather than an isolated analytic predicate.

Proof. Project the carrier of Definition 3386.1. Its source rows name $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{TopVecSpace}^{\uparrow}$, while the optional comparison row names $\mathsf{Norm}^{\uparrow}$ and the downstream boundary names $\mathsf{FunctionalAnalysis}^{\uparrow}$.

The local-convexity evidence is then read from S and B. The seminorm ledger supplies the finite probe family, and the neighbourhood ledger supplies the convex-balanced rows attached to the same topological-vector source.

Finally transport, replay, provenance, and naming are supplied by H, C, P, N. Repacking these rows gives the forward lattice above and leaves no coordinate for a detached locally convex predicate, selected dual, Hahn–Banach row, or weak-star compactness theorem. □

Every $\mathsf{LocallyConvexSpace}^{\uparrow}$ read that is consumed by $\mathsf{FunctionalAnalysis}^{\uparrow}$ or $\mathsf{FrechetSpace}^{\uparrow}$ must expose the convex-neighbourhood and seminorm-window surface before any downstream linear-topology or complete-metrizable row is read. The route passes from the local carrier of chapter 3386 through the $\mathsf{FrechetSpace}^{\uparrow}$ dependency of chapter 3495 and the functional-analysis consumers cited by chapter 94; it exports no host locally convex space, selected infinite seminorm family, quotient completion, or ambient completeness coordinate.

Theorem 3386.12 (Locally convex functional-analysis forward link)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer that cites the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the vector-space source of chapter 47, the topological-vector-space source of chapter 205, the norm-comparison row of chapter 89, and the functional-analysis boundary of chapter 94 through the displayed seminorm ledger S and convex-balanced neighbourhood ledger B. The downstream Frechet route of chapter 3495 and weak-star route of chapter 5192 read* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *only through those same S and B rows, not through a selected dual or host locally convex predicate. The route exports finite local-convexity evidence only through $V, T, S, B, M, H, C, P, N$; it does not export a selected dual, a Hahn–Banach principle, weak-star compactness, or an ambient locally convex topology predicate.*

Proof. Project the carrier of Definition 3386.1. Its rows V, T, M are the displayed vector, topological-vector, and norm-comparison sources, while S and B are the only local-convexity ledgers.

The functional-analysis consumer can therefore reach local convexity only after those source rows have been read through the seminorm and convex-balanced neighbourhood data. The Frechet and weak-star consumers inherit the same restriction: they may inspect S and B, but they receive no selected dual, host locally convex predicate, or detached topology witness. Transport, replay, provenance, and naming through H, C, P, N preserve the same finite route, so no dual selector, Hahn–Banach row, weak-star compactness row, or host topology predicate is introduced. □

Theorem 3386.13 (Locally convex functional-analysis dependency route)**.** *Every* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet must first read the seminorm family row S, the* $\mathsf{TopVecSpace}^{\uparrow}$ *transport row T, the finite convex-balanced neighbourhood ledger B, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$ *row N. The consumer boundary of chapter 94 may then cite the packet only through those visible rows, never through a selected dual, a host locally convex topology predicate, a Hahn–Banach principle, quotient equality of functionals, or an ambient completeness theorem.*

Proof. Project the accepted carrier of Definition 3386.1. Its locally convex data are the seminorm row S and the convex-balanced neighbourhood ledger B, while the linear-topology compatibility is supplied by the $\mathsf{TopVecSpace}^{\uparrow}$ row T.

The downstream $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer can therefore replay local convexity only after S, T, B are visible, and only with the transport, replay, provenance, and naming rows H, C, P, N attached. The functional-analysis chapter of chapter 94 receives that finite route as a dependency surface, so no selected dual, host topology predicate, quotient equality, or completeness principle is introduced. □

The top-level $\mathsf{LocallyConvexSpace}^{\uparrow}$ hub exposes its sibling lattice through chapter 205, chapter 208, chapter 5868, and chapter 94. A locally convex read therefore enters the dossier through displayed topological-vector operations, convex-neighbourhood evidence, finite seminorm-family rows, and functional-analysis consumers before any downstream packet treats $V, T, S, B, M, H, C, P, N$ as a reusable source surface. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ dependency lattice is read through its displayed seminorm-family and topological-vector rows rather than through a detached host predicate.

A consumer reaches the packet by first reading $\mathsf{Seminorm}^{\uparrow}$ and $\mathsf{SeminormedSpace}^{\uparrow}$ of chapter 5868 for seminorm evidence, $\mathsf{TopVecSpace}^{\uparrow}$ of chapter 205 for addition and scalar-continuity transport, $\mathsf{ConvexSet}^{\uparrow}$ for convex balanced neighbourhood rows, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ for the downstream citation boundary. No route exports a selected dual, Hahn–Banach principle, weak-star compactness theorem, or ambient locally convex topology outside the finite NameCert packet.

Theorem 3386.14 (Locally convex neighbourhood sibling route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *neighbourhood read first consumes the* $\mathsf{TopVecSpace}^{\uparrow}$ *vector/topology rows of chapter 205 and the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite seminorm rows of chapter 5868. Only after those sibling rows are visible may the packet expose convex balanced neighbourhood data through the local rows* B, H, C, P, N*.*

Proof. Project the carrier of Definition 3386.1. The topological-vector row T records the linear topology and operation-continuity route, while the seminorm ledger S records the finite probes that generate the accepted neighbourhood reads.

Read those rows through their sibling chapters. The $\mathsf{TopVecSpace}^{\uparrow}$ route supplies the vector/topology source, and the $\mathsf{SeminormedSpace}^{\uparrow}$ route supplies the finite seminorm comparison surface before any convex balanced neighbourhood can be used.

The neighbourhood row B is therefore downstream of T and S. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route and do not introduce a detached locally convex host predicate. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ seminorm and convex-neighbourhood ledgers consume the sibling lattice directly: $\mathsf{VecSpace}^{\uparrow}$ supplies the linear operations through chapter 47, $\mathsf{TopVecSpace}^{\uparrow}$ supplies continuity of addition and scalar multiplication through chapter 205, $\mathsf{Norm}^{\uparrow}$ supplies scalar-bound readbacks through chapter 89, and $\mathsf{FunctionalAnalysis}^{\uparrow}$ supplies downstream replay through chapter 94. The $\mathsf{NormedSpace}^{\uparrow}$ row of chapter 1950 is the single-norm specialization whose scalar-bound readbacks are consumed before the finite seminorm-family route is generalized to $\mathsf{LocallyConvexSpace}^{\uparrow}$. The seminormed-space row of chapter 5868 and the topological-vector-space row are the finite routes through which the carrier reads convex-neighbourhood and seminorm-control data. A local-convexity consumer therefore reads those sibling rows as part of the finite ledger route before citing the packet. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier consumes the finite seminorm-family ledger only after the topological-vector classifier surface of chapter 205 is visible. The same read is then available to the functional-analysis route of chapter 94: consumers inspect the S seminorm row and the B convex-balanced neighbourhood row as displayed NameCert data, not as a host locally convex predicate or an unbounded family of seminorms. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier reads its separating seminorm family through the finite family-seminorm packet of chapter 5868 and reads topology compatibility through the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205 together with the complete-metrizable consumer route of chapter 3495. The seminorm rows generate the displayed locally convex neighbourhood basis before the B convex-balanced neighbourhood ledger can be used, so a downstream consumer reaches local convexity through S and B as BEDC NameCert rows rather than through an ambient locally convex predicate. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ dependency lattice reads the $\mathsf{Seminorm}^{\uparrow}$ family through the finite family route of chapter 5868 and the $\mathsf{TopVecSpace}^{\uparrow}$ carrier of chapter 205 before any convex-neighbourhood route is available. Thus the local-convexity packet exposes seminorm-control rows and topological-vector continuity rows as finite sibling data, then passes them to S, B, H, C, P, N for transport, replay, provenance, and local naming rather than treating local convexity as a kernel-only predicate. Every $\mathsf{LocallyConvexSpace}^{\uparrow}$ consumer reads seminorm-family rows through the finite seminormed-space carrier of chapter 5868 and reads topological-vector operations through chapter 205 before any convex-neighbourhood or continuous-linear functional route is used. Thus

the local-convexity handoff remains the displayed route through $V, T, S, B, M, H, C, P, N$, not a host locally convex predicate detached from its seminorm and topological-vector-space sources. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet reads its neighbourhood basis only after the finite seminorm-family row and the $\mathsf{TopVecSpace}^{\uparrow}$ linear-topology row have been displayed together. The seminorm ledger is anchored by chapter 5868, while the ambient additive-topological continuity source is anchored by chapter 205; downstream functional-analysis consumers therefore use the locally convex surface through $V, T, S, B, M, H, C, P, N$, not through a detached host predicate. The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier reads its separating seminorm family through the finite family ledger of chapter 5868 and its ambient additive-topological structure through the $\mathsf{TopVecSpace}^{\uparrow}$ source of chapter 205. A local-convexity consumer therefore reaches the S seminorm row only after the topological-vector operations and continuity rows have been displayed, and the resulting packet remains the finite route $V, T, S, B, M, H, C, P, N$ rather than a host locally convex predicate.

Theorem 3386.15 (Locally convex space sibling dependency route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer that exports a locally convex read must first expose the sibling route through the substantive dependency surfaces of chapter 205 for the* $\mathsf{TopVecSpace}^{\uparrow}$ *topology row, chapter 47 for the* $\mathsf{VecSpace}^{\uparrow}$ *or module algebra row, the finite local seminorm ledger carried by this packet, and chapter 89 for the optional* $\mathsf{Norm}^{\uparrow}$ *comparison row when present, together with the structural* hsame*,* Cont*,* Pkg*, and local* NameCert *rows. The exported read is therefore a horizontal dependency over the displayed vector, topology, seminorm, norm-comparison, transport, replay, provenance, and naming surfaces; it is not a standalone host predicate for local convexity.*

Proof. Project the accepted carrier of Definition 3386.1. Its rows $V, T, S, B, M, H, C, P, N$ already order the vector, topology, seminorm, neighbourhood, comparison, transport, replay, provenance, and naming data.

The sibling route reads T and V before any locally convex probe is exported. The seminorm ledger S supplies finite local convexity probes, and the comparison row M supplies a norm route only when such a row is displayed by the packet.

The structural rows H, C, P, N then transport, replay, source, and name the same finite route. Since no independent host predicate occurs in the carrier, every locally convex read remains tied to the sibling dependency surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocallyConvexSpace}^{\uparrow}$ packet is generated from a vector-space source, a topological-vector-space source, a finite seminorm-family ledger, a convex balanced neighbourhood ledger, optional norm comparison rows, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3386.1, Definition 3386.2, Definition 3386.3, and Definition 3386.15 seed the finite locally convex carrier over $V, T, S, B, M, H, C, P, N$ and its sibling dependency route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocallyConvexSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice-selected duals, ambient weak-star compactness, spectral theory, unrestricted barrelledness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked vector, topological-vector, seminorm, convex balanced neighbourhood, norm comparison, transport, replay, provenance, sibling dependency, and local naming rows for BEDC.Derived.LocallyConvexSpaceUp.

Theorem 3386.16 (Locally convex space sibling route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer must read the* $\mathsf{Seminorm}^{\uparrow}$ *evidence carried by the finite seminorm-family route of chapter 5868, the* $\mathsf{TopVecSpace}^{\uparrow}$ *topology row of chapter 205, and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *consumer boundary of chapter 94 before treating the locally convex carrier as available. The local carrier is therefore a horizontal dependency through displayed seminorm, topological-vector, functional-analysis, transport, replay, provenance, and naming rows, not a detached host predicate.*

Proof. Project the accepted carrier of Definition 3386.1. The seminorm ledger, topological-vector source, convex-neighbourhood rows, and structural rows are all visible coordinates of the same finite packet.

The seminorm evidence is read first through the finite family route cited by the seminormed-space chapter, and the topology row is read through $\mathsf{TopVecSpace}^{\uparrow}$. Only after those sibling rows are displayed may a $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer use the packet as a locally convex source.

The remaining rows transport, replay, source, and name the same dependency surface. Since no coordinate stores a selected dual, Hahn–Banach principle, weak-star compactness theorem, or ambient locally convex topology predicate, the consumer receives only the displayed sibling route. □

Theorem 3386.17 (Locally convex TopVecSpace anchor)**.** *Every accepted* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *packet consumes the* $\mathsf{TopVecSpace}^{\uparrow}$ *carrier of chapter 205 and the* $\mathsf{ConvexSet}^{\uparrow}$ *neighbourhood rows of chapter 208 before any seminorm ledger, local-base read, or functional-analysis handoff is available. The local convexity read is therefore a sibling route through displayed vector-topology and convex-neighbourhood data, not an isolated host predicate.*

Proof. Project the carrier of Definition 3386.1. The row T is the topological-vector-space source, while B is the convex balanced neighbourhood ledger.

The row T must be read first because it supplies the topology, addition-continuity ledger, and scalar-continuity ledger used by every neighbourhood request. The row B is then read through $\mathsf{ConvexSet}^{\uparrow}$ as the displayed convex-neighbourhood evidence.

Only after T and B are visible may the finite seminorm ledger S or a local-base consumer inspect the packet. Transport, replay, provenance, and local naming keep the same route, so the chapter records a horizontal dependency through $\mathsf{TopVecSpace}^{\uparrow}$ and $\mathsf{ConvexSet}^{\uparrow}$ rather than a kernel-only local-convexity predicate. □

The $\mathsf{LocallyConvexSpace}^{\uparrow}$ carrier consumes the $\mathsf{TopVecSpace}^{\uparrow}$ carrier and operation-continuity transport route of chapter 205 together with the $\mathsf{Seminorm}^{\uparrow}$ family rows read through the finite seminormed-space surface of chapter 5868. Banach, Frechet, and weak-star consumers may inspect locally convex data only after those topological-vector and seminorm ledgers have displayed S, B, H, C, P, N as NameCert rows, so the route exports no detached locally convex host predicate, selected dual, Hahn–Banach principle, or ambient completeness theorem.

Theorem 3386.18 (Locally convex topological-vector-space dependency route)**.** *Every* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *consumer must read the* $\mathsf{TopVecSpace}^{\uparrow}$ *linear-topology row of chapter 205, the* $\mathsf{ConvexSet}^{\uparrow}$ *neighbourhood row of chapter 208, and the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *continuity-facing row of chapter 94 before treating the packet as locally convex. The route binds local convexity to displayed linear topology, convex neighbourhood, finite seminorm, continuity, transport, replay, provenance, and local naming rows; it is not a standalone host predicate for locally convex spaces.*

Proof. Project the carrier of Definition 3386.1. Its rows T, S, B already separate the topological-vector-space source, finite seminorm ledger, and convex balanced neighbourhood ledger.

The linear-topology row is read first through $\mathsf{TopVecSpace}^{\uparrow}$. The convex-neighbourhood row is then read through $\mathsf{ConvexSet}^{\uparrow}$ as the displayed source for the neighbourhood evidence in B.

Only after those rows are visible may the $\mathsf{FunctionalAnalysis}^{\uparrow}$ consumer inspect continuity-facing data. Transport, replay, provenance, and local naming preserve the same finite route, so no independent locally convex host predicate, selected dual, Hahn–Banach row, or weak-star compactness row is exported. □

Weak-star topology and weak-star compactness consumers read the $\mathsf{LocallyConvexSpace}^{\uparrow}$ source row before finite bounded-linear-functional windows: chapter 5192 uses the displayed weak-observation probes, and chapter 6401 consumes the same locally convex surface before the finite dual-ball compactness handoff. The route keeps weak-star reads tied to vector, topology, seminorm, convex-neighbourhood, transport, replay, provenance, and local naming rows rather than to a host locally convex predicate. The $\mathsf{ForwardSocketLedger}^{\uparrow}$ hub reads its forward request only after the supply ledger of chapter 528 and the far-end boundary of chapter 1061.

Chapter 3387

A Concrete Naming Certificate for ForwardSocketLedger$^{\uparrow}$

The ForwardSocketLedger$^{\uparrow}$ packet realizes the forward-socket family boundary of chapter 9401 as finite NameCert rows. It sits beside the supply-socket accounting of chapter 528, the apophatic socket classifier of chapter 536, the far-end boundary packet of chapter 1061, and the local inscription carrier of chapter 610. Its role is to make the forward read auditable before any consumer cites the apophatic far-end boundary.

The dependency route consumed before the forward socket export passes through the supply ledger of chapter 528 and the far-end boundary of chapter 1061; the forward packet only exports after those ledger and far-end rows are visible.

Definition 3387.1 (Forward socket ledger carrier)**.** *A* ForwardSocketLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, R, A, G, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the socket-kind row, recording which forward socket is being read and how it is classified among the accepted socket forms.*
2. *R is the requested supply-shape row, recording the requested external shape without supplying it as a kernel object.*
3. *A is the non-internalization marker, recording that the requested far-end supply is a boundary and not a generated state, record, rule, or* BHist *constructor.*
4. *G is the* GAP *ledger row, binding the refusal of internal supply to the visible socket-use and supply-ledger routes.*
5. *T is the apophatic far-end boundary row, naming the far end only as the common boundary read by the forward socket family.*
6. *H records componentwise* hsame *transport for K, R, A, G, T and the local naming row.*
7. *C records displayed* Cont *replay through socket kind, requested supply shape, non-internalization,* GAP *ledger, far-end boundary, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* ForwardSocketLedger$^{\uparrow}$*,* SupplySocketLedger$^{\uparrow}$*,* ApophaticSocket$^{\uparrow}$*,* ApophaticFarEndSocket$^{\uparrow}$*,* InscriptionPoint$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *row.*
The carrier contains no coordinate for T *as a kernel object, no hidden requested-supply witness, no socket selector outside K, no object-level equality between the far end and a local state, and no host fixed-point principle.*

Theorem 3387.2 (Forward socket ledger NameCert obligations)**.** *For every accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier $F = (K, R, A, G, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *surface consists exactly of socket-kind exposure through K; requested supply-shape exposure through R; non-internalization evidence through A;* GAP *ledger exposure through G; apophatic far-end boundary exposure through T; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 3387.1. The tuple lists socket kind, requested supply shape, non-internalization, gap ledger, far-end boundary, transport, replay, provenance, and local naming as explicit rows.

The socket can be read only after K and R expose its kind and requested shape. The rows A and G then mark the request as boundary supply rather than as an internal object.

The far-end boundary row T is downstream of those rows. Transport, replay, provenance, and naming through H, C, P, N preserve the same finite route. Since no hidden selector, requested-supply witness, internal far-end object, or host fixed-point principle occurs in the carrier, these rows exhaust the local NameCert surface. □

Theorem 3387.3 (Forward socket ledger apophatic non-escape)**.** *Every accepted read of the apophatic far-end boundary through* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *must first expose $K, R, A, G, T, H, C, P, N$. It cannot cite the far end as an internal object, a generated state, a selected supply witness, a socket-family fixed point outside the ledger, or an equality between* T *and a local inscription point.*

Proof. Start with the rows K and R. They identify the forward socket and its requested supply shape before any far-end boundary is available.

The rows A and G are the non-internalization gate. They state that the requested supply remains a GAP-ledgered boundary rather than a kernel-side value.

Only then may T be read as the apophatic far-end boundary, and H, C, P, N only transport, replay, source, and name that same finite route. Thus any consumer that skips $K, R, A, G, T, H, C, P, N$ has no accepted route to the far end. □

Falsifiable prediction: Every accepted read through $\mathsf{ForwardSocketLedger}^{\uparrow}$ must expose socket kind, requested supply shape, non-internalization marker, GAP ledger, apophatic far-end boundary, transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before any consumer can cite the far-end boundary.

Independence witness: supplysocketledger, apophaticsocket, apophaticfarendsocket, inscriptionpoint: none is bijective with $\mathsf{ForwardSocketLedger}^{\uparrow}$ because this carrier alone binds socket kind, requested supply shape, non-internalization, GAP ledger, far-end boundary, transport, replay, provenance, and local NameCert rows before a far-end read is accepted.

Closure status for $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ForwardSocketLedger}^{\uparrow}$ packet is generated from socket-kind exposure, requested supply-shape exposure, non-internalization evidence, a GAP ledger row, an apophatic far-end boundary row, componentwise transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3387.1, Definition 3387.2, and Definition 3387.3 seed the finite forward-socket ledger over $K, R, A, G, T, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not internalize T, select a hidden supply witness, close a consciousness theorem, provide a host fixed-point theorem, claim object-level equality with a local inscription point, give Lean verification, or provide bridge checking.

Upgrade path: Obligation closure requires checked socket-kind, requested-supply, non-internalization, GAP-ledger, far-end boundary, transport, replay, provenance, and local naming rows for BEDC.Derived.ForwardSocketLedgerUp.

Theorem 3387.4 (Forward socket ledger MetaCIC candidate-discharge route)**.** *Let $F = (K, R, A, G, T, H, C, P, N)$ be an accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier. Any forward socket export consumed by the MetaCIC candidate-discharge frontier of chapter 765 must expose the forward rows $K, R, A, G, T, H, C, P, N$ before the MetaCIC discharge socket D can be read. The forward rows are dependency data for the socket consumer; they are not a concealed subject-reduction discharge and do not fill D by themselves.*

Proof. Project the accepted carrier of Definition 3387.1. The rows $K, R, A, G, T, H, C, P, N$ expose the socket kind, requested shape, non-internalization marker, gap ledger, far-end boundary, transport, replay, provenance, and local naming before any export is available.

Now read the candidate-discharge frontier of chapter 765. Its packet keeps the discharge socket D visible while candidate-mediated strong-normalization, confluence-facing, bounded-conversion, and source-readiness rows are consumed.

A MetaCIC consumer that receives a forward socket export therefore receives dependency data already ordered by the forward carrier. Since the forward packet contains no subject-reduction certificate row and no coordinate that identifies the far-end boundary with a typing proof, the export can only feed the visible socket route.

Thus the MetaCIC discharge socket may read the forward rows as dependency surface, but D remains a separate socket row until an explicit subject-reduction certificate is displayed. □

Theorem 3387.5 (Forward socket ledger SocketUse dependency route)**.** *Let $F = (K, R, A, G, T, H, C, P, N)$ be an accepted* $\mathsf{ForwardSocketLedger}^{\uparrow}$ *carrier. Any accepted forward socket export that consumes the far-end boundary row T factors through the visible socket-use ledger of chapter 1044 before the local* $\mathsf{NameCert}_{\mathsf{ForwardSocketLedger}^{\uparrow}}$ *row is read. In particular, the export route must first expose the supply ledger of chapter 528, the far-end boundary of chapter 1061, the socket-use event rows $A, S, Q, V, R, H, C, P, N$ of* $\mathsf{SocketUseLedger}^{\uparrow}$*, and only then the forward rows $K, R, A, G, T, H, C, P, N$.*

Proof. Project the accepted carrier of Definition 3387.1. The forward packet presents the requested supply shape, non-internalization marker, GAP ledger, and far-end boundary as visible rows rather than as an internal object.

Use the supply-socket and far-end sibling routes cited in the carrier provenance. The supply ledger accounts for the requested shape and refusal route, while the apophatic far-end socket keeps the boundary row non-internalized.

The visible consumption of that boundary is the role of $\mathsf{SocketUseLedger}^{\uparrow}$. Its event rows expose the marked use and refusal ledger before an audit consumer reaches a local socket-use NameCert row. Therefore a forward export that cites T must pass through that event surface before N may name the forward packet. □

Chapter 3388

A Concrete Naming Certificate for $\mathsf{SuccessiveApproximation}^{\uparrow}$

Picard–Banach successive approximation reads an iterated approximation schedule through finite budgets before any completed limit claim is exposed. The $\mathsf{SuccessiveApproximation}^{\uparrow}$ packet sits on the L10 route from the convergence-modulus schedule of chapter 1438, the displayed limit handoff of chapter 1413, the regular rational readback of chapter 301, the finite observation windows of chapter 126, and the dyadic tolerance ledger of chapter 291. It gives contraction, Picard, and fixed-point consumers a finite iteration-budget surface rather than a host limit, countable-choice schedule, quotient sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3388.1 (Successive approximation carrier)**.** *A* $\mathsf{SuccessiveApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, T, W, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite iteration row, listing the inspected Picard approximants and the displayed application of the step operator at each row;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error-budget ledger attached to the inspected rows;*
3. *T is the finite tail schedule that records which iteration indices are compared at each displayed budget;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to read each displayed approximant and its successor;*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and dyadic budgets;*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row, visible only after I, D, T, W, Q have been displayed;*
7. *H records componentwise* hsame *transport for iteration rows, budgets, tail schedules, windows, readbacks, and the limit handoff;*
8. *C records displayed* Cont *replay for consumers of the same finite schedule;*
9. *P is the* Pkg *provenance over* $\mathsf{SuccessiveApproximation}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$ *row.*

The packet contains no host sequence limit, no countable-choice selector for tails, no quotient stream equality, no trichotomy split, and no ambient completeness coordinate.

Theorem 3388.2 (Successive approximation NameCert obligations)**.** *For every accepted* $\mathsf{SuccessiveApproximation}^{\uparrow}$ *carrier* $K = (I, D, T, W, Q, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite iteration rows, budget-classifier stability through* D, T, W, Q*, convergence-modulus handoff through* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$*,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *nonescape through* L*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, local naming through* N*, and refusal of host limits and countable choice. Every fixed-point, Picard, or contraction consumer receives only this displayed schedule before reading a terminal real-facing row.*

Proof. Project the carrier of Definition 3388.1. The iteration row I, dyadic budget ledger D, tail schedule T, finite windows W, and regular rational readbacks Q are the only approximation rows available to the packet.

Use the sibling convergence route. The convergence-modulus schedule of chapter 1438 consumes the same dyadic budgets and tail rows, while chapter 1413 accepts the resulting limit handoff only after $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks have been displayed.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the same finite schedule, so the packet cannot export a host limit, countable-choice tail selector, quotient sequence equality, trichotomy decision, or ambient completeness witness. □

Lean4 ✓ `BEDC.Derived.SuccessiveApproximationUp.SuccessiveApproximationCarrier namecert obligations`

Falsifiable prediction: Every fixed-point consumer of a SuccessiveApproximationUp packet reads a finite iteration budget before any RealSequenceLimitUp handoff is visible.

Independence witness: cauchy-convergence-modulus, real-sequence-limit, regseqrat, streamname, dyadicratcore: SuccessiveApproximationUp is not bijective to any listed sibling because it jointly carries iteration rows, dyadic error budgets, tail schedules, finite windows, readback, limit handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SuccessiveApproximation}^{\uparrow}$ packet is generated from finite Picard iteration rows, dyadic error budgets, tail schedules, StreamName windows, RegSeqRat readbacks, a RealSequenceLimit handoff, transport, replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$, seed)

Scope closed: Definition 3388.1 and Definition 3388.2 declare the finite Picard-style iteration-budget surface for L10 Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SuccessiveApproximation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host limits, countable choice, quotient sequence equality, fixed-point existence, contraction uniqueness, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, budget classifier stability, convergence-modulus handoff, RealSequenceLimit nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SuccessiveApproximationUp.

Chapter 3389

A Concrete Naming Certificate for $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational name is admitted into the $\mathsf{Real}^{\uparrow}$ seal. It sits after the regular-rational source of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the dense-embedding route of chapter 1453, and the terminal real-name seal of chapter 38. The packet exposes the source-to-seal admission route without importing quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3389.1 (RealCauchyDenseEmbedding carrier)**.** *A* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, J, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, exposing the regular Cauchy rational name and its displayed modulus discipline;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect the finite source entries requested by a real-name consumer;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested precision, comparison budget, and slack rows used by the admission route;*
4. *J is the* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *route row that names the dense source-to-real embedding discipline;*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, J have been displayed;*
6. *H records componentwise* hsame *transport for the regular source, windows, dyadic ledger, dense-embedding route, Real seal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a Real-admission request through R, W, D, J, S, H;*
8. *P is the* Pkg *provenance over* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a selected representative of a real name, countable choice over approximation stages, trichotomy, a host completion operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyDenseEmbeddingUp.RealCauchyDenseEmbeddingTasteGate single carrier alignment`

Theorem 3389.2 (RealCauchyDenseEmbedding NameCert obligations)**.** *For every accepted* RealCauchyDenseEmb *carrier* $E = (R, W, D, J, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular-rational source* R*, finite window exposure through* W*, dyadic tolerance accounting through* D*, dense-embedding routing through* J*, Real-seal handoff through* S*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. A* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer must read* R, W, D, J *before the seal row* S *is available.*

Proof. Project the carrier of Definition 3389.1. The regular source, finite windows, dyadic tolerance ledger, dense-embedding route, Real seal, transport, replay, provenance, and local naming rows are exactly the displayed coordinates.

The source side is finite. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular Cauchy rational name, the $\mathsf{StreamName}^{\uparrow}$ row opens only the requested windows, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the tolerance ledger used by those windows.

The dense-embedding row J then mediates admission to the Real seal. The row S is downstream of R, W, D, J, and the structural rows H, C, P, N preserve and name the same route, so the certificate surface cannot expose quotient real equality, selected representatives, trichotomy, countable choice, or ambient completeness. □

Theorem 3389.3 (RealCauchyDenseEmbedding Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyDenseEmbedding}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Nam}$$

The terminal $\mathsf{Real}^{\uparrow}$ *row is only a seal for the displayed regular Cauchy rational source and its finite tolerance windows. It is not a source of quotient real equality, selected representatives, countable choice, trichotomy, host completion, or ambient real completeness.*

Proof. Start with the obligation surface of Definition 3389.2. A consumer cannot read the Real seal until the regular source, finite windows, dyadic tolerance ledger, and dense-embedding route have already been displayed.

The sibling rows enforce the direction of the read. $\mathsf{RegSeqRat}^{\uparrow}$ supplies the source name, $\mathsf{StreamName}^{\uparrow}$ supplies only finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the precision ledger, and $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ supplies the source-to-seal route.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no refused host-side principle is a coordinate of E, the Real row remains a seal and not an escape principle. □

Falsifiable prediction: Every accepted $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ consumer reads one $\mathsf{RegSeqRat}^{\uparrow}$ source row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and one $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ route before any $\mathsf{Real}^{\uparrow}$ seal is exposed.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-dense-embedding, real: $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries

source regularity, finite windows, dyadic tolerance accounting, dense-embedding routing, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegularCauchyDenseEmbedding route, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3389.1, Definition 3389.2, and Definition 3389.3 seed the finite regular-Cauchy rational source-to-Real seal admission packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, trichotomy, host completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity, finite window exposure, dyadic tolerance accounting, dense-embedding routing, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyDenseEmbeddingUp.

$\mathsf{RegularCauchyUniqueness}^{\uparrow}$ reads uniqueness through chapter 301, chapter 126, chapter 291, chapter 1412, and chapter 38 before any completion consumer cites the seal.

Chapter 3390

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniqueness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ packet records the textbook uniqueness route for regular Cauchy names as finite BEDC data. It sits after the regular-rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the comparison discipline of chapter 1412, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Real-completion consumers such as chapter 3579 use $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ to make the $\mathsf{Real}^{\uparrow}$ seal independent of the chosen $\mathsf{RegSeqRat}^{\uparrow}$ representative. The chapter only seeds the finite route from equality data for regular Cauchy names to the Real seal boundary.

Definition 3390.1 (Regular Cauchy uniqueness carrier)**.** *A* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, S, T, D, E, R, H, C, P, N)$$

with the following visible rows:

1. *X and Y are paired* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy name rows;*
2. *S and T are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the two names are observed at the same requested tolerance;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the displayed radius, comparison budget, and rational-distance rows used by the uniqueness route;*
4. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row tying the paired windows to one finite equivalence decision;*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the paired names, windows, dyadic tolerance ledger, and comparison row have been displayed;*
6. *H stores componentwise* hsame *transport for X, Y, S, T, D, E, R and the structural rows;*
7. *C stores displayed* Cont *replay of the route*

$$X, Y \longmapsto S, T, D \longmapsto E \longmapsto R;$$

8. *P is the* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by the displayed rows $X, Y, S, T, D, E, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, ambient Real completeness, or an unrecorded equality proof outside the packet.

Lean4 ✓ `BEDC.Derived.RegularCauchyUniquenessUp.RegularCauchyUniquenessTasteGate single carrier alignment`

Theorem 3390.2 (Regular Cauchy uniqueness NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier*

$$U = (X, Y, S, T, D, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$ *surface consists exactly of carrier admission for the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *names, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under the same displayed rows; tolerance exactness through D; transport invariance through H; ledger visibility through C and P; and Real-seal nonescape through R.*

Proof. Carrier admission is projection from Definition 3390.1. The projection exposes exactly the two regular Cauchy names, their finite windows, the dyadic tolerance ledger, the comparison row, the seal row, and the structural rows.

Classifier stability follows because the classifier is componentwise on the same displayed rows. Tolerance exactness is the projection of D, and ledger visibility is the fact that every consumer route in C reads X, Y, S, T, D, E, R before the provenance and naming rows.

Transport invariance is supplied by H: each transported accepted carrier keeps the same row roles. The Real-seal boundary is downstream from E, so no consumer can read R without first passing through the finite comparison packet. □

Theorem 3390.3 (Regular Cauchy uniqueness Real-seal obligations)**.** *Every accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet that exposes a* $\mathsf{Real}^{\uparrow}$ *seal must display the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and one* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row E before the seal row R is available. The seal obligation is therefore the ordered finite route*

$$X, Y \longmapsto S, T, D \longmapsto E \longmapsto R,$$

preserved by H, C, P, N and not by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3390.1. The projection exposes exactly the paired regular Cauchy rows, paired stream windows, dyadic tolerance ledger, comparison row, and terminal seal row.

Use Definition 3390.4 to bind the comparison to the single row D. The windows S, T and names X, Y are opened at that same tolerance source before E is read.

Now use Definition 3390.7. It places every $\mathsf{Real}^{\uparrow}$-facing read downstream from E and excludes quotient stream equality, selected limits, host real equality, countable choice, and ambient completeness from the seal route. The structural rows H, C, P, N transport, replay, source, and name that same finite obligation surface. □

Theorem 3390.4 (Regular Cauchy uniqueness tolerance exactness)**.** *Every accepted comparison in a* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet reads its tolerance data through the single displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D. The paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T and paired* $\mathsf{RegSeqRat}^{\uparrow}$ *names*

X, Y are opened at that row's radius and comparison budget, and the $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row E consumes no tolerance source outside D.*

Consequently the uniqueness route exports no hidden metric estimate, quotient-stream equality, selected limit, or ambient Real-completeness argument.

Proof. Project the carrier and inspect the tolerance coordinate. The only dyadic radius and rational-distance data are stored in D.

The windows S, T are tied to that coordinate by the continuation route C. Thus the two regular Cauchy names are observed through the same finite tolerance request before the comparison row is read.

The comparison row E is downstream from S, T, D. Since no other tolerance coordinate is present in the carrier, an accepted comparison cannot be justified by an external estimate or by quotient equality of streams. □

Theorem 3390.5 (Regular Cauchy uniqueness transport invariance)**.** *Let $U = (X, Y, S, T, D, E, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier. Componentwise* hsame *transport of the paired names, windows, tolerance ledger, comparison row, seal row, continuation route, provenance row, and local naming row produces another accepted carrier with the same uniqueness route. The transported classifier still reads only $X, Y, S, T, D, E, R, H, C, P, N$.*

Proof. Read the transport row H. Its components move the two $\mathsf{RegSeqRat}^{\uparrow}$ names, the two $\mathsf{StreamName}^{\uparrow}$ windows, the dyadic tolerance row, the comparison row, and the seal row together.

Replay the route through C after transport. The same ordered route from the paired names to the windows, tolerance, comparison row, and seal row remains visible.

Repack the transported rows with the same provenance and local naming data. Since the classifier has no coordinate outside the displayed rows, the transported packet is accepted by the same finite rule. □

Theorem 3390.6 (Regular Cauchy uniqueness tail-shift stability)**.** *Let $U = (X, Y, S, T, D, E, R, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carrier whose paired* $\mathsf{RegSeqRat}^{\uparrow}$ *windows have the same* $\mathsf{Real}^{\uparrow}$ *seal row R and the same uniqueness tolerance ledger D. For any finite* $\mathsf{StreamName}^{\uparrow}$ *tail shift of the displayed windows S, T, the shifted windows preserve the uniqueness handoff through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row, and* $\mathsf{Real}^{\uparrow}$ *seal boundary, with H, C, P, N carrying the transported route.*

Proof. Project the carrier rows. The paired names X, Y, windows S, T, dyadic tolerance ledger D, comparison row E, and seal row R are all displayed before any Real-facing read is available.

Apply Definition 3390.4. The tail-shifted windows are still read at the same displayed tolerance source D, so the $\mathsf{RegSeqRat}^{\uparrow}$ readback and comparison row cannot import an external estimate.

Use Definition 3390.5 for the shifted windows and their transported structural rows. The same route through X, Y, S, T, D, E, R remains accepted after the finite reindexing.

Finally apply Definition 3390.7. Since the seal is downstream from the same comparison row and the carrier has no quotient stream equality, selected limit, countable-choice, ambient-completeness, or host-equality coordinate, the finite tail shift preserves the uniqueness handoff. □

Theorem 3390.7 (Regular Cauchy uniqueness Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *packet factors through the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows S, T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison row E, and only then the seal row R, with H, C, P, N preserving the same route.*

Thus the Real seal cannot escape to quotient stream equality, a selected limit, countable choice, ambient Real completeness, or any unrecorded host equality proof.

Proof. Start from a Real-facing read and project the continuation route C. The route lists the paired names, paired windows, dyadic tolerance row, and comparison row before the seal row R.

Use Definition 3390.4 to pin the comparison to D. Then use Definition 3390.5 to see that transport and replay preserve the same ordered finite rows.

The provenance and naming rows P, N attach the route to the local packet. Because the carrier contains no quotient stream, selected limit, countable-choice, ambient-completeness, or host-equality coordinate, no Real-facing consumer can read such data from the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ packet is generated from paired $\mathsf{RegSeqRat}^{\uparrow}$ regular Cauchy names, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ comparison row, a terminal $\mathsf{Real}^{\uparrow}$ seal, transport data, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3390.1, Definition 3390.2, Definition 3390.3, Definition 3390.4, Definition 3390.5, and Definition 3390.7 name the L10 regular-Cauchy uniqueness route from finite RegSeqRat equality data through StreamName and DyadicRatCore tolerances to the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier surface, classifier stability, Real-seal obligations, tolerance exactness, transport invariance, ledger visibility, and Real-seal nonescape rows for the displayed uniqueness packet.

The downstream $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer of chapter 3579 reads $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ through the carrier of Definition 3390.1, the tolerance exactness row of Definition 3390.4, and the Real seal nonescape row of Definition 3390.7. This route keeps the $\mathsf{Real}^{\uparrow}$ seal independent of the chosen $\mathsf{RegSeqRat}^{\uparrow}$ representative: the completion witness consumes the displayed uniqueness comparison before the seal row is available, and it receives no quotient stream, selected limit, countable choice, or ambient completeness coordinate from the uniqueness packet.

Theorem 3390.8 (Regular Cauchy uniqueness classifier lock)**.** *Let*

$$U = (X, Y, S, T, D, E, R, H, C, P, N)$$

and

$$U' = (X', Y', S', T', D', E', R', H', C', P', N')$$

be accepted $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *carriers. If a consumer reads the two terminal* $\mathsf{Real}^{\uparrow}$ *seal rows R and R' as the same regular Cauchy limit, then that read must factor through the displayed Cauchy-equivalence classifier before either seal is consumed.*

Concretely, the read passes through the paired $\mathsf{RegSeqRat}^{\uparrow}$ *rows X, Y and X', Y', the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S, T and S', T', the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers D, D', the comparison rows E, E', componentwise* hsame *transport H, H', displayed* Cont *replay C, C',* Pkg *provenance P, P', and the local* NameCert *rows N, N'. It receives no host quotient stream equality, no selected representatives, no countable choice, and no unrecorded equality proof outside the local packet rows.*

Proof. Project both accepted carriers through Definition 3390.1. The projection exposes the paired regular Cauchy names, finite stream windows, dyadic tolerance ledgers, comparison rows, terminal real seals, and structural rows on both sides.

Invoke the tolerance exactness row of Definition 3390.4. It binds each comparison read to the displayed dyadic ledger before the comparison row can be consumed, so the two seal reads cannot be compared without first exposing the same kind of finite tolerance source.

The sibling route then passes through chapter 1820 before chapter 38: the comparison rows E, E' are consumed as Cauchy-equivalence classifier data, and only after that classifier read are the terminal Real seal rows available.

Next use the completion-consumer route following the chapter closure block. That route says every $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer reads $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ through the carrier, tolerance exactness, and Real seal nonescape rows before any $\mathsf{Real}^{\uparrow}$ seal is available.

Finally apply Definition 3390.7. Since the carriers contain no quotient stream, selected representative, countable-choice, or host-equality coordinate, the only available equality route is the displayed classifier route through E, E' together with $H, H', C, C', P, P', N, N'$. The read is therefore locked to the local Cauchy-equivalence classifier before either terminal seal is consumed. □

$\mathsf{RegularCauchyUniqueness}^{\uparrow}$ reads the completion-uniqueness boundary of chapter 1782 after the $\mathsf{RegSeqRat}^{\uparrow}$ comparison row and before any $\mathsf{Real}^{\uparrow}$-facing terminal seal. This dependency keeps the uniqueness route on displayed finite regular-Cauchy completion rows, and it rules out quotient stream equality, selected representatives, ambient real equality, or any choice of a preferred limit witness. The $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ route belongs to the Real classifier lattice: before any terminal uniqueness consumer compares two real-name seals, it reads the source $\mathsf{RegSeqRat}^{\uparrow}$ rows of chapter 301, inspects equality through the $\mathsf{RealNameClassifier}^{\uparrow}$ boundary of chapter 1517, and then consumes the $\mathsf{Real}^{\uparrow}$ seal rows already displayed by the carrier. This order keeps the terminal comparison attached to regular-rational source data and to the classifier inspection route rather than to quotient stream equality, selected representatives, host real equality, or ambient completeness.

Proof. Project the carrier rows and read the comparison from left to right. The regular Cauchy source rows provide the two names, the classifier boundary supplies the finite equality inspection, and the Real seal rows are consumed only after those entries have been displayed, so no uniqueness consumer receives an unrecorded terminal comparison. □

Chapter 3391

A Concrete Naming Certificate for $\mathsf{DyadicMeshComposition}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicMeshComposition}^{\uparrow}$ packet records the finite common-refinement route used when two displayed dyadic meshes are composed before a Real-and-Completion consumer reads a closed interval net. It sits between the dyadic mesh surface of chapter 358, the interval-net route of chapter 1250, the dyadic rational arithmetic surface of chapter 291, and the stream-window readbacks of chapter 126 and chapter 301. The packet is a finite naming certificate for mesh composition; it is not an open-cover compactness theorem, quotient stream equality, selected limit, host real equality, or ambient completeness primitive.

Definition 3391.1 (DyadicMeshComposition carrier)**.** *A* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M_0, M_1, R, Q, H, C, P, N)$$

with the following displayed rows:

1. M_0 *is the source* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the first finite ordered dyadic mesh and its inspected cell ledger.*
2. M_1 *is the target* $\mathsf{DyadicMesh}^{\uparrow}$ *row naming the second finite ordered dyadic mesh and its inspected cell ledger.*
3. R *is the common-refinement relation row, listing for each displayed source cell the finite family of target cells that refines it and recording the reverse containment reads needed by downstream consumers.*
4. Q *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger storing the endpoint comparisons, radius splits, and dyadic slack witnesses used by the refinement relation.*
5. H *records componentwise* hsame *transport for* M_0, M_1, R, Q, C, P, N.
6. C *records displayed* Cont *replay from a mesh-composition request through the two mesh rows, the common-refinement relation, and the dyadic bound ledger.*
7. P *is the* Pkg *provenance over* $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. N *is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an infinite mesh, selected subcover, quotient equality of streams, host real equality, hidden point chooser, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3391.2 (DyadicMeshComposition NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier* $T = (M_0, M_1, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{DyadicMesh}^{\uparrow}$ *rows, common-refinement relation exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over the mesh, interval-net, stream, readback, history, transport, continuation, and naming rows, and local naming. Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *consumer of the packet must read* M_0, M_1, R, Q *before it can use the structural rows* H, C, P, N*.*

Proof. Project the carrier of Definition 3391.1. The rows M_0 and M_1 expose the two finite dyadic mesh ledgers, while R records the displayed common-refinement relation between their cells.

Read the cell-bound ledger next. The row Q supplies the dyadic endpoint comparisons and radius splits that witness each refinement edge in R, so the relation is checked through $\mathsf{DyadicRatCore}^{\uparrow}$ data rather than through a host-level order on real intervals.

The remaining rows add only structure. The transport row H preserves the displayed coordinates, C replays the finite route, P records provenance for the named dependencies, and N names the local certificate. These projections exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.DyadicMeshCompositionUp.DyadicMeshCompositionNameCertObligations`

Theorem 3391.3 (DyadicMeshComposition refinement transitivity)**.** *Let*

$$T = (M_0, M_1, R, Q, H, C, P, N)$$

be an accepted $\mathsf{DyadicMeshComposition}^{\uparrow}$ *carrier. If a displayed cell of* M_0 *refines through a finite* R*-row into displayed cells of* M_1*, and those target cells are later inspected by a second accepted dyadic mesh row, then the composite inspection is again represented by finite refinement rows over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger. The composite exports no infinite refinement tree, quotient cell equality, hidden selector, or ambient completeness coordinate.*

Proof. Induct over the finite BHist rows displaying the refinement relation R. The empty row has no cell to export, so the composite route is the empty displayed refinement.

For the step row, project the current source cell and the finite family of target cells named by R. The ledger Q supplies the endpoint comparisons and radius splits for that row, while the induction hypothesis composes the remaining displayed rows.

Transport and replay preserve the constructed route. Componentwise hsame in H keeps the source mesh, target mesh, refinement row, and dyadic ledger aligned, and the Cont row C replays the same finite sequence of inspections. Thus the composite remains a finite mesh-refinement packet with no extra analytic coordinate. □

Theorem 3391.4 (DyadicMeshComposition closed-interval-net handoff)**.** *Every* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *consumer that uses a* $\mathsf{DyadicMeshComposition}^{\uparrow}$ *packet reads the interval net through the displayed mesh rows* M_0, M_1*, the common-refinement relation* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *cell-bound ledger* Q*, and the structural rows* H, C, P, N*. The handoff supplies the finite mesh-composition route needed by closed bounded interval nets without adding open-cover compactness, selected subcovers, quotient stream equality, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted closed-interval-net consumer. By chapter 1250, the consumer already reads interval coverage through finite dyadic mesh rows and their rational and dyadic cell ledgers.

Now insert the mesh-composition packet. The rows M_0, M_1, R, Q identify the finite common-refinement route, and Definition 3391.2 shows that this route is the complete local certificate surface before the structural rows are consumed.

Finally apply the refinement transitivity theorem. It keeps successive mesh inspections inside the displayed dyadic ledger and finite BHist induction. The closed-interval-net handoff is therefore exhausted by the stated rows and cannot expose any of the excluded compactness, quotient, selector, host-equality, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshComposition}^{\uparrow}$ packet is generated from two DyadicMesh rows, a finite common-refinement relation, a DyadicRatCore cell-bound ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3391.1, Definition 3391.2, Definition 3391.3, and Definition 3391.4 seed the finite common-refinement route consumed by closed bounded interval nets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient stream equality, infinite refinement trees, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement exactness, DyadicRatCore cell-bound accounting, transport stability, replay stability, closed-interval-net handoff, and provenance exhaustion for BEDC.Derived.DyadicMeshCompositionUp.

Chapter 3392

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ packet names the finite operator surface that turns a regular Cauchy family and its displayed modulus into a Real-facing limit seal. It sits after the tail-modulus handoff of chapter 3383 and beside the completion operator of chapter 3379; the originating proof route is the regular-Cauchy construction of section 9238.1. The packet gives Phase Real and Completion a single operator-level object over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not another handoff-only certificate.

Definition 3392.1 (RegularCauchyLimitOperator carrier)**.** *A* $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\mu, \tau, S, R, D, H_0, E, H, C, P, N)$$

with the following displayed rows:

1. μ *is the modulus-input row for the regular Cauchy family over* $\mathsf{RegSeqRat}^{\uparrow}$.
2. τ *is the finite tail selector row that turns a requested tolerance into the shared tail threshold used by the diagonal read.*
3. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected at the selected threshold and at the diagonal representative.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected family member, the diagonal representative, and the comparison witnesses read by the operator.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording endpoint tolerances, the middle modulus leg, slack accounting, and the final requested bound.*
6. H_0 *is the* $\mathsf{TailModulusLimitHandoff}^{\uparrow}$ *route imported from chapter 3383.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* μ, τ, S, R, D, H_0 *have been displayed.*
8. H *records componentwise* hsame *transport for the modulus input, tail selector, windows, regular readbacks, dyadic ledger, handoff route, Real seal, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay from a limit-operator request through* $\mu, \tau, S, R, D, H_0, E, H, P, N$.

10. *P is the* Pkg *provenance over* RegularCauchyLimitOperator$^{\uparrow}$, TailModulusLimitHandoff$^{\uparrow}$, CauchyCompletionOperat StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RegularCauchyLimitOperator}^{\uparrow}}$ *row.*

The classifier accepts only packets whose operator read is exhausted by these rows. It has no coordinate for quotient streams, selected host limits, countable-choice schedules, trichotomy decisions, host completion operators, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 3392.2 (RegularCauchyLimitOperator NameCert obligations)**.** *For every accepted* RegularCauchyLimitO *carrier $L = (\mu, \tau, S, R, D, H_0, E, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyLimitOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the modulus input, finite tail selector,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* TailModulusLimitHandoff$^{\uparrow}$ *route, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion-facing consumer must read μ, τ, S, R, D, H_0 before the Real seal row E is available.*

Proof. Project the carrier of Definition 3392.1. The rows μ and τ expose the constructive modulus request and the selected finite tail threshold.

Read the finite analytic rows. The window row S, readback row R, and tolerance ledger D supply the inspected StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ data used by the diagonal estimate.

The handoff route H_0 imports the already displayed tail-modulus route of chapter 3383. The Real seal E is downstream of that route, and H, C, P, N only transport, replay, source, and name the same finite packet.

Since quotient streams, selected host limits, countable-choice schedules, trichotomy decisions, host completion operators, and ambient completeness are absent from the carrier inventory, they cannot become local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitOperatorUp.RegularCauchyLimitOperatorNameCertObligations`

Theorem 3392.3 (RegularCauchyLimitOperator diagonal Real seal)**.** *Every Real-completion consumer of an accepted* RegularCauchyLimitOperator$^{\uparrow}$ *packet receives the finite operator route*

$$\mu, \tau, S, R, D, H_0 \longmapsto E \longmapsto H, C, P, N.$$

The route factors the diagonal regularity construction of section 9238.1 through the tail-modulus handoff of chapter 3383 and the completion-facing operator discipline of chapter 3379. It exports a Real$^{\uparrow}$ *seal only as a displayed consequence of the modulus input, selected tail, finite windows, regular readback, dyadic tolerance ledger, and handoff route.*

Proof. Start with an accepted limit-operator packet. By Definition 3392.2, the consumer first reads the modulus input, the tail selector, the finite windows, the regular-rational readback, and the dyadic tolerance ledger.

The tail-modulus row H_0 then packages those reads in the handoff form already isolated by Definition 3383.3. This is the finite route consumed by the diagonal regularity argument before any terminal real name can be sealed.

The Real row E therefore has no independent host-limit coordinate. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming, so the exported diagonal seal is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ packet is generated from a regular Cauchy modulus input, finite tail selector, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, TailModulusLimitHandoff route, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3392.1, Definition 3392.2, and Definition 3392.3 seed the finite operator surface from regular Cauchy modulus data to a Real-facing diagonal seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host completion operators, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-input exactness, finite tail selector coverage, StreamName window coverage, RegSeqRat readback, DyadicRatCore tolerance accounting, TailModulusLimitHandoff compatibility, Real seal nonescape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyLimitOperatorUp.

Chapter 3393

A Concrete Naming Certificate for DnaWindowSixBoundaryTile$^{\uparrow}$

The DnaWindowSixBoundaryTile$^{\uparrow}$ packet realizes the finite stop/Trp and Ile/Met $X_2 \sqcup \{11\}$ boundary-tile surface of Definition 9462.1154 and Definition 9462.1156 as BEDC NameCert data. It consumes the six-coordinate codon source window of chapter 2234, the standard-code transition rows of chapter 2245, the stop-suffix packet of chapter 1374, and the spectral stabilization route of chapter 2270. The packet records the two boundary tiles as finite classifier rows, not as a hidden biological table or an ambient stochastic model.

Definition 3393.1 (DnaWindowSixBoundaryTile carrier)**.** *A* DnaWindowSixBoundaryTile$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, M, B, I, S, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates of the relevant codon windows;*
2. *M is the* CodonTransitionMatrix$^{\uparrow}$ *row exposing the standard-code bridge rows used to read stop, tryptophan, isoleucine, methionine, and initiation contexts;*
3. *B is the finite boundary-tile classifier row, recording the stop/Trp tile* $\mathrm{U} + \mathrm{RR} + X_2/11$ *and the Ile/Met tile* $\mathrm{AU} + X_2/11$*;*
4. *I is the Ile/Met initiation handoff row, distinguishing the Met amino-acid row from the canonical start bridge row when initiation context is present;*
5. *S is the* StopCodonZeckendorfSuffix$^{\uparrow}$ *stop-suffix handoff row for the stop-side tile;*
6. *L is the* SpectralStabilization$^{\uparrow}$ *sibling route row used by finite consumers that compare the same stop-face boundary with spectral support data;*
7. *H records componentwise* hsame *transport for source windows, transition rows, boundary tiles, initiation handoff, stop suffix, spectral route, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay from a boundary-tile request through W, M, B, I, S, L, H;*
9. *P is the* Pkg *provenance over* DnaWindowSixBoundaryTile$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, CodonTransitionMatrix$^{\uparrow}$, StopCodonZeckendorfSuffix$^{\uparrow}$, SpectralStabilization$^{\uparrow}$, *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* NameCert*;*

10. N is the local $\mathsf{NameCert}_{\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}}$ *row.*
The classifier accepts only packets whose stop/Trp and Ile/Met boundary reads are exhausted by $W, M, B, I, S, L, H, C, P, N$. It has no coordinate for a hidden codon table, global biological semantics, probabilistic inference, quotient codon equality, or a selected stochastic process.

Lean4 (definition) `BEDC.Derived.DnaWindowSixBoundaryTileUp.DnaWindowSixBoundaryTileUp`

Theorem 3393.2 (DnaWindowSixBoundaryTile NameCert obligations)**.** *For every accepted* DnaWindowSixBound *carrier $T = (W, M, B, I, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}}$ *surface consists exactly of source-window admission through W, standard-code bridge-row admission through M, boundary-tile classifier exactness through B, initiation handoff through I, stop-suffix handoff through S, spectral-stabilization sibling routing through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3393.1. The source window, transition bridge, boundary classifier, initiation handoff, stop suffix, spectral sibling route, transport, replay, provenance, and local naming rows are the only displayed coordinates.

The stop/Trp tile and the Ile/Met tile are finite classifier reads. The row W supplies the six coordinate positions, M supplies the standard-code bridge rows, and B records the two $X_2 \sqcup \{11\}$ partitions.

The rows I, S, L keep the initiation, stop-suffix, and spectral consumers attached to the same boundary-tile surface. Finally H, C, P, N transport, replay, source, and name that surface, so no hidden codon table, global biological semantics, probabilistic inference, quotient codon equality, or selected stochastic process is exported. □

Theorem 3393.3 (DnaWindowSixBoundaryTile consumer nonescape)**.** *Every accepted consumer of a* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *packet reads the route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto B, I, S, L \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Thus the six-coordinate codon window and standard-code bridge rows are read before any stop/Trp, Ile/Met, initiation, stop-suffix, or spectral-stabilization consumer can use the boundary-tile classifier. The route exports no hidden biological table, global semantics, probabilistic inference, quotient codon equality, or selected stochastic model.

Proof. Use Definition 3393.2. The boundary classifier row B is downstream of the source-window row W and standard-code bridge row M.

The initiation, stop-suffix, and spectral rows are also downstream. They may consume the displayed boundary-tile classifier, but they do not replace the six-coordinate source window or the standard-code bridge rows.

The structural rows H, C, P, N preserve and name the same finite route. Since the carrier contains no row for the refused external principles, every accepted consumer remains inside the displayed packet. □

Falsifiable prediction: Any accepted $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ consumer omitting the six-coordinate $\mathsf{CodonSixBitWindow}^{\uparrow}$ source window or the standard-code $\mathsf{CodonTransitionMatrix}^{\uparrow}$ bridge rows refutes the packet.

Independence witness: codon-six-bit-window, codon-transition-matrix, stopcodonzeckendorf-suffix, spectral-stabilization: $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two boundary-tile classifier rows, initiation handoff, stop-suffix routing, spectral sibling routing, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.DnaWindowSixBoundaryTileUp` must provide explicit Nontrivial DnaWindowSixBoundaryTile$^{\uparrow}$ and FieldFaithful DnaWindowSixBoundaryTile$^{\uparrow}$ instances preserving $W, M, B, I, S, L, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$ **.**

Constructive story: A DnaWindowSixBoundaryTile$^{\uparrow}$ packet is generated from codon six-bit windows, standard-code bridge rows, finite stop/Trp and Ile/Met boundary-tile classifiers, initiation handoff, stop-suffix handoff, spectral sibling routing, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$, seed)

Scope closed: Definition 3393.1, Definition 3393.2, and Definition 3393.3 seed the finite boundary-tile packet over codon windows and standard-code bridge rows.

Formal verification: FormalStatus(NameCert$_{\text{DnaWindowSixBoundaryTile}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological table identity, global biological semantics, probabilistic inference, quotient codon equality, selected stochastic modeling, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-tile classifier exactness, initiation handoff, stop-suffix handoff, spectral sibling routing, transport, replay, provenance, local naming, Nontrivial DnaWindowSixBoundaryTileUp, and FieldFaithful DnaWindowSixBoundaryTileUp for BEDC.Derived.DnaWindowSixBoundaryTileUp.

Chapter 3394

A Concrete Naming Certificate for $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ packet records the finite route from located endpoint signs through dyadic bisection windows to the $\mathsf{Real}^{\uparrow}$ seal. It sits next to the sign-bracket certificate of chapter 1402 and the dyadic branch certificate of chapter 2203, but it keeps the located real output as a sealed finite read rather than a selected root or a quotient real.

The packet is an L10 Real and Completion neighbour. Its source cells are located interval endpoints, its branch rows are dyadic, and its consumer boundary is the same finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route used by the regular-real seal.

Definition 3394.1 (Located real intermediate-value carrier)**.** *A* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, S, B, Q, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, including the lower endpoint cell, upper endpoint cell, located interval comparison data, and the dyadic endpoint refinements consumed by the branch route;*
2. *S is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign ledger, recording the accepted negative-side and positive-side endpoint reads and the displayed midpoint sign rows carried along the route of Definition 1402.11;*
3. *B is the* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *branch data, retaining the finite sequence of accepted dyadic child intervals and the sign boundary that travels with each branch row;*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row whose radius and midpoint arithmetic determine the displayed stopping window;*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the retained endpoints and midpoint observations;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows after Q fixes the finite tolerance request;*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after I, S, B, Q, W, R have been displayed;*

8. *H records componentwise* hsame *transport for the located interval, sign ledger, dyadic branch data, tolerance row, window rows, readback row, seal row, replay row, provenance row, and local naming row;*
9. *C records displayed* Cont *replay of the route*

$$I, S \longmapsto B, Q \longmapsto W, R \longmapsto E;$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{DyadicIntermediateValue}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. It has no coordinate for a selected root, quotient real, real trichotomy decision, open-cover compactness argument, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.LocatedRealIntermediateValueUp.LocatedRealIntermediateValueTasteGate`
`single carrier alignment`

Theorem 3394.2 (Located real intermediate-value NameCert obligations)**.** *For an accepted* LocatedRealIntermedia *carrier*

$$L = (I, S, B, Q, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$ *surface consists of carrier admission for the displayed located interval, sign ledger, dyadic branch, tolerance, window, readback, seal, transport, replay, provenance, and naming rows; sign-ledger visibility through I and S; dyadic branch exactness through B and Q; finite-window readback through W and R; Real-seal exposure through E; and structural stability through H, C, P, N.*

Every obligation is internal to the displayed finite packet. In particular, the certificate cannot be discharged by selecting a root, passing to a quotient real, deciding trichotomy, invoking open-cover compactness, or using an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 3394.1. The projection exposes exactly the located interval, sign, branch, tolerance, window, readback, seal, transport, replay, provenance, and local naming rows.

Sign-ledger visibility follows by reading S only over the located endpoint row I. The dyadic branch stage then reads B together with the tolerance row Q, so each accepted child interval is tied to displayed midpoint arithmetic and a visible sign boundary.

The finite-window stage reads W and then R. The real-facing row E is downstream of those windows, while H, C, P, N transport, replay, source, and name the same route.

Since none of the carrier coordinates stores a root selector, quotient-real equality, trichotomy oracle, compactness datum, or Real-completeness principle, none of these objects can serve as a local obligation. □

Theorem 3394.3 (Located real sign-branch route)**.** *Every Real-facing read from an accepted* $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ *packet factors through*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{IntermediateValue}^{\uparrow} \longmapsto \mathsf{DyadicIntermediateValue}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The route reads the located endpoint row I, the sign ledger S, the branch data B, the tolerance row Q, the windows W, the readback row R, and only then the seal row E, with H, C, P, N preserving the same finite packet.

Proof. Project the accepted carrier $L = (I, S, B, Q, W, R, E, H, C, P, N)$. The endpoint and sign stages are I followed by S, so the read begins with located signs rather than a host predicate over all real points.

Apply the dyadic bisection route of Definition 2203.3. The branch row B and tolerance row Q determine the finite child-interval route before any window data is consumed.

Next read the finite windows and regular-rational readback. The rows W and R are upstream of the seal row E, so a Real-facing consumer reaches the seal only through the displayed finite route.

The structural rows H, C, P, N transport, replay, source, and name that same route. Therefore no selected root, quotient real, real trichotomy decision, open-cover compactness argument, or ambient completeness principle can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealIntermediateValue}^{\uparrow}$ packet is generated from a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint row, an $\mathsf{IntermediateValue}^{\uparrow}$ sign ledger, a $\mathsf{DyadicIntermediateValue}^{\uparrow}$ branch row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3394.1, Definition 3394.2, and Definition 3394.3 seed the finite located-sign route from dyadic branch data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected root, quotient real, real trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-ledger visibility, dyadic branch exactness, finite-window readback, Real-seal exposure, transport and replay stability, and provenance exhaustion for BEDC.Derived.LocatedRealIntermedia

Chapter 3395

A Concrete Naming Certificate for $\mathsf{LocatedRealLine}^{\uparrow}$

Following Bishop and Bridges, the located real line is presented as a finite interface over located real names, apartness evidence, and regular Cauchy readback, rather than as a host real-line object or quotient completion. The $\mathsf{LocatedRealLine}^{\uparrow}$ packet binds the located-name surface of chapter 1933, the positive-separation evidence of chapter 304, the terminal real seal of chapter 38, the regular rational source route of chapter 301, and the finite observation windows of chapter 126. It also reads the dyadic endpoint and tolerance ledgers of chapter 291 so L10 real-completion routes can cite one non-isolated line interface.

Definition 3395.1 (LocatedRealLine carrier)**.** *A* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Lambda = (L, A, Q, R, S, D, H, C, P, N)$$

with the following displayed rows:

1. L *is the* $\mathsf{LocatedReal}^{\uparrow}$ *row presenting the point-facing located real-name surface.*
2. A *is the* $\mathsf{ApartnessReal}^{\uparrow}$ *row presenting positive finite-window separation evidence available to line-facing order and metric consumers.*
3. Q *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same regular rational source route.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular Cauchy readback used by* L *and* Q*.*
5. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation schedule naming the windows inspected by the line interface.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint, radius, and tolerance ledger used for located windows, apartness gaps, and regular Cauchy bounds.*
7. H *records componentwise* hsame *transport for* L, A, Q, R, S, D *and for the structural replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a line request through* $L, A, Q, R, S, D, H, P, N$*.*
9. P *is the* Pkg *provenance over* $\mathsf{LocatedRealLine}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{ApartnessReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite located real-line surface. It contains no quotient real equality, trichotomy decision, host real-line primitive, ambient completeness theorem, selected limit, countable-choice schedule, or bridge credential.

Lean4 (definition) `BEDC.Derived.LocatedRealLineUp.LocatedRealLineUp`

Theorem 3395.2 (LocatedRealLine NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier* $\Lambda = (L, A, Q, R, S, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located-real row, apartness row, real seal, regular rational source, stream schedule, dyadic ledger, transport, replay, provenance, and local naming rows;*
2. *classifier stability, requiring sameness to be componentwise transport of* L, A, Q, R, S, D *through the displayed* hsame *and* Cont *routes;*
3. *located point exactness, requiring every point read to pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *row* L *and its scheduled regular Cauchy source rows* R, S, D*;*
4. *apartness visibility, requiring every positive gap exposed to an order or metric consumer to pass through the* $\mathsf{ApartnessReal}^{\uparrow}$ *row* A *and the same finite dyadic ledger* D*;*
5. *real-seal compatibility, requiring the* $\mathsf{Real}^{\uparrow}$ *row* Q *to retain the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *source surface seen by* L*;*
6. *non-escape, excluding quotient real equality, trichotomy, host real-line primitives, ambient completeness, selected limits, countable choice, Lean verification, and bridge checking as local data.*

No located real-line obligation is stored outside these rows.

Proof. Project the carrier of Definition 3395.1. The analytic rows are precisely the located-real row L, apartness row A, real seal Q, regular rational source R, stream schedule S, and dyadic ledger D.

Classifier stability is componentwise. The rows H and C transport and replay the six analytic rows, while P and N source and name the same displayed packet.

Located point exactness follows by reading L through chapter 1933. Apartness visibility follows by reading A through chapter 304. In both cases the only line-facing observations available are the scheduled regular Cauchy windows and dyadic ledgers already stored in R, S, D.

The seal row Q is terminal and shares the same R, S source route. Since no remaining coordinate contains quotient equality, trichotomy, a host real line, ambient completeness, selected limits, or countable choice, the listed rows exhaust the local certificate. □

Theorem 3395.3 (LocatedRealLine regular Cauchy real handoff)**.** *Every L10 real-completion consumer of an accepted* $\mathsf{LocatedRealLine}^{\uparrow}$ *carrier receives the finite route*

$$L, A, R, S, D \longmapsto Q \longmapsto H, C, P, N.$$

The handoff first exposes located real-name data, apartness-gap data, scheduled regular rational readback, stream windows, and dyadic tolerance accounting. Only after those rows are displayed does it expose the terminal $\mathsf{Real}^{\uparrow}$ *seal and the structural transport, replay, provenance, and local naming rows.*

Proof. Start with an accepted carrier $\Lambda = (L, A, Q, R, S, D, H, C, P, N)$. By Definition 3395.2, any consumer must read the point surface, positive-gap surface, regular Cauchy source, stream schedule, and dyadic ledger before the seal row is visible.

The located-real component L supplies the point-facing readback through the finite windows of chapter 126 and the regular source rows of chapter 301. The apartness component A supplies positive finite-window gap evidence through chapter 304.

The real seal Q is then the same terminal seal route cited by chapter 38. The dyadic ledger D remains the only endpoint, radius, and tolerance account available from chapter 291.

Repacking through H, C, P, N preserves exactly this displayed route. The carrier has no coordinate for quotient real equality, trichotomy, a host real-line primitive, ambient completeness, selected limits, countable choice, or bridge data, so the L10 handoff is exhausted by the finite packet above. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealLine}^{\uparrow}$ packet is generated from a LocatedReal row, an ApartnessReal row, a RealUp seal, RegSeqRat readback, StreamName windows, a DyadicRatCore ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3395.1, Definition 3395.2, and Definition 3395.3 seed the Bishop-style located real-line interface for L10 real-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, trichotomy, host real-line primitives, ambient completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located point exactness, apartness visibility, regular Cauchy readback, dyadic ledger accounting, Real seal compatibility, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LocatedRealLineUp.

The $\mathsf{LocatedUpperBound}^{\uparrow}$ hub routes the reusable upper-bound witness between **??**, chapter 38, and the downstream chapter 1427 consumer.

Chapter 3396

A Concrete Naming Certificate for $\mathsf{LocatedUpperBound}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedUpperBound}^{\uparrow}$ packet isolates the finite upper-bound witness that a Real or Completion consumer may inspect before any least-upper-bound or completeness principle is available. It sits over the located-cut surface of chapter 404, the located-real surface of **??**, the comparison discipline of chapter 1381, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the downstream located-supremum consumer of chapter 1427. The chapter names upper-bound data as a reusable NameCert surface, rather than keeping it only as an internal row of the supremum packet.

Definition 3396.1 (LocatedUpperBound carrier)**.** *A* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the located family row, read through either a* $\mathsf{LocatedCut}^{\uparrow}$ *or* $\mathsf{LocatedReal}^{\uparrow}$ *surface.*
2. *U is the upper-bound ledger row. It records the displayed bound name, the comparison windows that certify the source family lies below that bound, and the dyadic tolerances attached to those comparisons.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule listing the finite windows inspected by the upper-bound ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the bound name and by every comparison window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located source, upper-bound ledger, finite windows, and regular readback have been displayed.*
6. *H records componentwise* hsame *transport for F, U, W, R, E and for the structural replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an upper-bound request through F, U, W, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite upper-bound surface. It contains no lower-approximation ledger, selected supremum, trichotomy row, quotient-cut carrier, host real equality test, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge credential.*

Definition 3396.2 (LocatedUpperBound classifier)**.** *The* $\mathsf{LocatedUpperBound}^{\uparrow}$ *classifier accepts exactly those finite packets* $B = (F, U, W, R, E, H, C, P, N)$ *whose upper-bound reads pass through the displayed source row* F*, comparison ledger* U*, finite request schedule* W*, regular rational readback* R*, and terminal real-name seal* E*. Classifier sameness is componentwise sameness of these rows through* H *and replay through* C*; no consumer-facing classifier may identify two packets by host real equality, quotient-cut equality, trichotomy, a selected supremum, countable choice, or an ambient completeness principle.*

Theorem 3396.3 (LocatedUpperBound NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier* $B = (F, U, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located source, upper-bound ledger, stream schedule, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *classifier stability, requiring upper-bound sameness to be componentwise transport of* F, U, W, R, E *through the displayed* hsame *and* Cont *routes;*
3. *source coverage, requiring every bounded source read to pass through the located-cut or located-real surface named by* F*;*
4. *comparison exactness, requiring every upper-bound assertion to pass through the* $\mathsf{BishopRealComparison}^{\uparrow}$ *discipline named by* U *at a finite window listed in* W*;*
5. *regular readback compatibility, requiring the bound name and each comparison window to use the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route before the* $\mathsf{Real}^{\uparrow}$ *seal is exposed;*
6. *non-escape, excluding lower-approximation cofinality, selected suprema, trichotomy, quotient cuts, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local data.*

No upper-bound obligation is stored outside these rows.

Proof. Project the carrier of Definition 3396.1. The analytic rows are precisely the source row F, upper-bound ledger U, finite window row W, regular readback row R, and terminal real seal E.

Classifier stability is componentwise. The rows H and C transport and replay the five analytic rows, while P and N source and name the same displayed packet.

Source coverage follows by reading F through the located-cut and located-real sibling surfaces. Comparison exactness follows by reading U through the Bishop-style comparison row at a window listed in W.

The readback row R is shared by the bound name and every comparison window before the seal E is exposed. Since no remaining coordinate contains lower-approximation cofinality, selected suprema, trichotomy, quotient cuts, host equality, countable choice, ambient completeness, Lean verification, or bridge data, the listed rows exhaust the local certificate. □

Theorem 3396.4 (LocatedUpperBound real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier receives the finite route*

$$F, U, W, R \longmapsto E \longmapsto H, C, P, N.$$

The handoff first exposes the located source, upper-bound comparison ledger, finite request schedule, and regular rational readback. Only after those rows are displayed does it expose the terminal $\mathsf{Real}^{\uparrow}$ *seal and the structural transport, replay, provenance, and local naming rows.*

Proof. Start with an accepted carrier $B = (F, U, W, R, E, H, C, P, N)$. By Definition 3396.3, any Real-facing consumer must read the source row, comparison ledger, finite window schedule, and regular readback before the real seal is available.

The source row F supplies the located-cut or located-real family interface. The ledger U supplies the upper-bound comparison rows through the Bishop-style comparison discipline, and W records the finite windows inspected for those comparisons.

The readback row R is the sole route into regular rational real-name data, and E is the only terminal real seal exposed by the carrier. Repacking through H, C, P, N preserves this order of disclosure.

Because the carrier has no lower-approximation coordinate, selected supremum, trichotomy row, quotient-cut carrier, host equality test, countable-choice schedule, ambient completeness theorem, or bridge data, the displayed route exhausts every $\mathsf{Real}^{\uparrow}$-facing handoff. □

Theorem 3396.5 (LocatedUpperBound supremum route)**.** *Every* $\mathsf{LocatedSupremum}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *carrier may use the packet only as the upstream upper-bound face. The consumer receives* $F, U, W, R, E, H, C, P, N$ *as the finite upper-bound witness, while lower-approximation cofinality, selected supremum data, and completion-facing least-upper-bound closure remain obligations of the downstream* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet.*

Proof. Inspect the carrier $B = (F, U, W, R, E, H, C, P, N)$. Its only order-facing analytic row is the upper-bound ledger U, and that row is tied to the located source F, the finite windows W, the regular readback R, and the real seal E.

Apply Definition 3396.4. It shows that any real-facing read exported from the upper-bound packet reaches E only through the displayed upper-bound route.

The downstream supremum packet in chapter 1427 additionally requires lower-approximation and cofinality rows. Those rows are absent from Definition 3396.1, so a $\mathsf{LocatedSupremum}^{\uparrow}$ consumer can read this packet only as its reusable upstream upper-bound witness. □

Theorem 3396.6 (LocatedUpperBound real-certificate route)**.** *Every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet exposes its Real-facing certificate only along the finite route from the located source row, upper-bound comparison ledger,* $\mathsf{StreamName}^{\uparrow}$ *request schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. This route is stable under the displayed* hsame *transports and* Cont *replay rows, and no Real-facing consumer can read the certificate before those rows have been displayed.*

Proof. Project the carrier of Definition 3396.1. Its Real-facing analytic rows are exactly the located source F, upper-bound comparison ledger U, finite request schedule W, regular rational readback R, and terminal real seal E.

Apply Definition 3396.3. Carrier admission, source coverage, comparison exactness, and regular readback compatibility require every upper-bound certificate read to pass through those five analytic rows before the structural rows are consumed.

Now apply Definition 3396.4. It identifies the displayed route

$$F, U, W, R \longmapsto E \longmapsto H, C, P, N$$

as the only Real-facing handoff exported by the packet.

The transport row H and continuation row C are componentwise rows of the same carrier, so they preserve and replay the route rather than adding a second certificate source. Since the carrier has no selected supremum, lower-approximation ledger, quotient-cut equality, host real equality, countable-choice schedule, ambient completeness theorem, Lean verification row, or bridge credential, the stated route exhausts the reusable Real-facing certificate. □

Theorem 3396.7 (LocatedUpperBound comparison exactness)**.** *Every accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet routes each upper-bound comparison through the located-cut, located-real, Bishop-real-comparison,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already displayed in the carrier before any supremum-facing consumer may read it. In particular, the upper-bound ledger U cannot be consumed as a detached order assertion: it must be read with the located source F, finite request schedule W, regular rational readback R, terminal seal E, and the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 3396.1. The rows F, U, W, R, E are the analytic comparison route, while H, C, P, N transport, replay, source, and name the same accepted packet.

Apply Definition 3396.4. It forces every Real-facing read to expose F, U, W, R before the terminal seal E is available.

Apply Definition 3396.5. A $\mathsf{LocatedSupremum}^{\uparrow}$ consumer receives this packet only as its upstream upper-bound witness, so it may not import lower-approximation cofinality or selected supremum data from the upper-bound carrier.

Apply Definition 3396.6. The route ties the upper-bound ledger to the finite request schedule and regular rational readback before any Real-facing certificate is consumed. Since the carrier has no selected supremum, trichotomy row, quotient-cut equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle, no supremum consumer can bypass the displayed comparison rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUpperBound}^{\uparrow}$ packet is generated from a located source row, an upper-bound comparison ledger, a StreamName finite-window schedule, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3396.1, Definition 3396.2, Definition 3396.3, Definition 3396.4, and Definition 3396.5 seed the reusable upper-bound witness for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUpperBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected suprema, trichotomy, quotient cuts, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, source coverage, comparison exactness, regular readback compatibility, Real seal handoff, supremum routing, transport stability, replay stability, and provenance exhaustion for BEDC.Derived.LocatedUpperBoundUp.

Theorem 3396.8 (LocatedSupremum upstream upper-bound face)**.** *The downstream* $\mathsf{LocatedSupremum}^{\uparrow}$ *packet of chapter 1427 consumes an accepted* $\mathsf{LocatedUpperBound}^{\uparrow}$ *packet only as its upstream upper-bound face. Concretely, the route supplied by Definition 3396.5 offers the finite rows $F, U, W, R, E, H, C, P, N$ as the displayed upper-bound witness; lower-approximation cofinality, least-upper-bound exactness, and supremum-facing closure remain rows of* $\mathsf{LocatedSupremum}^{\uparrow}$ *rather than rows of* $\mathsf{LocatedUpperBound}^{\uparrow}$.

Proof. Read Definition 3396.5. It identifies the accepted upper-bound carrier as a reusable witness for the source family, the upper-bound ledger, finite windows, regular readback, the Real seal, and the structural transport, replay, provenance, and local naming rows.

Now compare the downstream consumer in chapter 1427. Its supremum certificate requires lower-approximation and cofinality data in addition to an upstream upper-bound face. Those additional rows are not coordinates of Definition 3396.1.

Thus every handoff from $\mathsf{LocatedUpperBound}^{\uparrow}$ into $\mathsf{LocatedSupremum}^{\uparrow}$ is exactly the forward dependency through $F, U, W, R, E, H, C, P, N$. The supremum packet must supply the remaining supremum-specific rows itself. $\square$

Chapter 3397

A Concrete Naming Certificate for $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$

Following Bourbaki and Kelley on uniform embeddings and completion, the $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ packet records the finite BEDC route by which an embedding into a separated completion is consumed without quotient representatives or selected limits. It sits after the metric-completion carrier of chapter 1506, the separated comparison surface of chapter 1411, the uniform-completion route of chapter 1473, the Cauchy-name comparison of chapter 1412, and the terminal seal boundary of chapter 38. The packet names the embedding interface as displayed rows over completion input, separatedness, uniform continuity, Cauchy-equivalence stability, and Real-facing readback; it does not assert a quotient completion, choose representatives, or import ambient completeness.

Definition 3397.1 (UniformEmbeddingCompletion carrier)**.** *A* $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, U, Q, R, T, W, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row supplying the completion-facing metric request and its displayed finite tolerance route.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row requiring the target-side comparison to be separated before the embedding read is public.*
3. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row exposing the entourage-facing completion presentation consumed by the embedding.*
4. *Q is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *row identifying when two source Cauchy names have the same displayed completion read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the completion, separatedness, uniform, and Cauchy-equivalence rows have been displayed.*
6. *T records the finite source-window schedule inspected by the embedding before a target read is accepted.*
7. *W records the finite target-window schedule produced by the embedding route.*
8. *A records the approximation ledger saying that matching source windows and tolerances transport to matching target windows.*

9. *H records componentwise* hsame *transport for M, S, U, Q, R, T, W, A and the structural rows.*
10. *C records displayed* Cont *replay of the route*

$$M, S, U, Q, T, A \longmapsto W \longmapsto R.$$

11. *P is the* Pkg *provenance over* UniformEmbeddingCompletion$^{\uparrow}$, MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CauchySequenceEquivalence$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{UniformEmbeddingCompletion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose embedding-to-completion read is exhausted by $M, S, U, Q, R, T, W, A, H, C,$ It has no coordinate for quotient representatives, selected limits, host equality of completions, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformEmbeddingCompletionUp.UniformEmbeddingCompletionTasteGate single carrier alignment`

Theorem 3397.2 (UniformEmbeddingCompletion NameCert obligations)**.** *For every accepted* UniformEmbeddingCompletion$^{\uparrow}$ *carrier*

$$E = (M, S, U, Q, R, T, W, A, H, C, P, N),$$

the local NameCert$_{\text{UniformEmbeddingCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed twelve-row packet; metric-completion exposure through M; separated-target exposure through S; uniform-completion exposure through U; Cauchy-equivalence stability through Q; terminal Real sealing through R; finite source-window exposure through T; finite target-window exposure through W; approximation transport through A; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3397.1. The accepted packet exposes exactly the completion source, separated target, uniform completion, Cauchy-equivalence comparison, terminal seal, source and target windows, approximation ledger, and structural rows listed there.

The analytic obligations are read in the displayed order. The completion request passes through M and U, the separated target boundary is supplied by S, and the source comparison is stabilized by Q before the terminal Real-facing row R can be inspected.

The embedding content is the finite window handoff T, A, W. The source windows in T and their approximation ledger A determine the target windows in W; no step inserts a quotient representative or a selected completion point.

Finally H, C, P, N transport, replay, source, and name the same route. Since the excluded quotient, choice, host-equality, and ambient-completeness objects are absent from the carrier, they cannot appear as local NameCert obligations. □

Theorem 3397.3 (UniformEmbeddingCompletion universal route)**.** *Every completion-facing consumer of an accepted* UniformEmbeddingCompletion$^{\uparrow}$ *packet factors through the finite route*

$$M, S, U, Q, T, A \longmapsto W, R \longmapsto H, C, P, N.$$

Thus a uniformly embedded source can be read by completion consumers only through displayed metric-completion input, separatedness, uniform-completion structure, Cauchy-equivalence stability, finite source windows, approximation transport, finite target windows, the terminal Real seal, transport, replay, provenance, and local naming.

Proof. Start with an accepted carrier. The source completion row M and uniform-completion row U provide only completion-facing requests, while the separated row S restricts target comparison to the displayed separated boundary.

The Cauchy-equivalence row Q supplies the only equality-facing source comparison. It is read before the target windows are public, so a consumer cannot replace it by quotient-stream equality or host completion equality.

The universal route is the finite handoff T, A, W. The source-window schedule in T is transported by the approximation ledger A to the target-window schedule W, and the Real-facing read reaches R only after those rows have been displayed.

The structural rows H, C, P, N preserve, replay, source, and name exactly this packet. Hence every completion-facing consumer factors through the displayed route and cannot read a selected limit, quotient representative, countable-choice schedule, host completion equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEmbeddingCompletion}^{\uparrow}$ packet is generated from MetricCompletion, SeparatedMetric, UniformCompletion, CauchySequenceEquivalence, Real, finite source windows, finite target windows, approximation transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3397.1, Definition 3397.2, and Definition 3397.3 seed the finite nonquotient route for uniformly embedded completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEmbeddingCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformEmbeddingCompletionUp.UniformEmbeddingCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, host equality of completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-target exposure, uniform-completion exposure, Cauchy-equivalence stability, finite window transport, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformEmbeddingCompletionUp.

Chapter 3398

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and Bourbaki on reflective subobjects, the $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ packet records the RegSeqRat-specific completion reflector boundary as finite BEDC naming data. It sits after the finite observation rows of chapter 126, the regular rational source of chapter 301, the completed real-name surface of chapter 360, the monadic completion route of chapter 378, the idempotence boundary of chapter 2381, and the extension surface of chapter 397. The packet names only the finite reflector rows visible to RegSeqRat consumers; it does not identify quotient completions, select limits, assert host equality of completed objects, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3398.1 (RegularCauchyCompletionReflector carrier)**.** *A* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, Q, E, U, I, P, H, C, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source regular Cauchy name and every completion-facing observation cut.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, exposing the regular rational readback and its displayed tolerance discipline before completion is requested.*
3. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *completed real-name row reached from S, Q through the displayed completion surface rather than through a host quotient carrier.*
4. *U is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *unit and bind route specialized to the same S, Q, E rows.*
5. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *boundary row recording that a second completion pass reuses the same completed surface.*
6. *P is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row for consumers whose maps are read through the displayed regular Cauchy source and the same completed real-name row.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, completed real-name rows, monad routes, idempotence boundaries, universal-property rows, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a RegSeqRat completion request through*

$$S, Q \longmapsto E \longmapsto U, I, P \longmapsto H, N.$$

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$ *row, with provenance read from the displayed* Pkg *rows over* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

The classifier compares accepted packets only through $S, Q, E, U, I, P, H, C, N$. It has no coordinate for quotient completion equality, selected limit representatives, host equality of completed objects, countable choice, detached $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionReflectorUp.RegularCauchyCompletionReflectorTasteGate`
`single carrier alignment`

Theorem 3398.2 (RegularCauchyCompletionReflector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier*

$$R = (S, Q, E, U, I, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for stream windows, RegSeqRat readbacks, completed real-name rows, monad routes, idempotence boundaries, universal-property extension rows, transport, replay, provenance, and local naming;*
2. *source regularity, requiring every completion-facing read to expose S and Q before E is available;*
3. *completion admission, requiring the completed row E to be read through the displayed* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *surface;*
4. *monad handoff, requiring unit and bind consumers to use U over the same S, Q, E rows;*
5. *idempotence stability, requiring repeated completion reads to reuse I and the same completed row E;*
6. *universal-property compatibility, requiring every extension consumer to read P over the same source, completion, monad, and idempotence rows;*
7. *nonescape, excluding quotient completions, selected limits, host equality, countable choice, detached* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. The stream window row, regular rational readback row, completed real-name row, monad route, idempotence boundary, universal-property row, transport, replay, and local naming row are exactly the displayed coordinates of R.

The source side is forced by S and Q. A completion-facing consumer has no completed row to inspect until the finite StreamName windows and the RegSeqRat readback have been displayed.

The completed row E is then read through the RealCauchyCompletion surface. The monad row U, idempotence row I, and universal-property row P all refer back to the same S, Q, E route, so none of them can introduce a separate completion object.

Finally H, C, N preserve and name the same finite packet. Since the carrier has no coordinate for quotient completions, selected limits, host equality, countable choice, detached Real completeness, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 3398.3 (RegularCauchyCompletionReflector unit handoff)**.** *Let $R = (S, Q, E, U, I, P, H, C, N)$ be an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier. Every unit-facing completion request factors through the finite route*

$$S \longmapsto Q \longmapsto E \longmapsto U \longmapsto H, C, N.$$

The handoff copies the displayed RegSeqRat source into the completed real-name surface by finite windows and regular readbacks. It does not supply quotient stream equality, a selected limit, host equality, countable choice, or ambient completeness.

Proof. Start with the source rows. The finite windows in S determine which parts of the regular rational source may be inspected, and Q supplies the matching RegSeqRat readback.

The completed row E is downstream of those source rows. A unit-facing request therefore reaches completion only after the finite source route has been displayed.

The monad row U records the unit handoff on that same route, and H, C, N transport, replay, and name it componentwise. No excluded host-side principle is available as a row of the packet. □

Theorem 3398.4 (RegularCauchyCompletionReflector idempotence boundary)**.** *For an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *carrier $R = (S, Q, E, U, I, P, H, C, N)$, a repeated completion read is classified by the same completed real-name row E and the same idempotence boundary I. The second read may replay S, Q, U, P, H, C, N, but it has no independent completed carrier and no quotient equality coordinate.*

Proof. Read the carrier coordinates. The only completed real-name coordinate is E, and the only idempotence coordinate is I.

A repeated completion request can therefore replay the source windows, RegSeqRat readbacks, monad handoff, universal-property row, transport, continuation, and local naming. It cannot branch to a second completed object because no such coordinate appears in R.

Thus idempotence is a boundary on the displayed completion surface, not an assertion of host quotient equality or selected-limit equality. □

Theorem 3398.5 (RegularCauchyCompletionReflector ledger nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ *packet reads the completion reflector through*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealCauchyCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionMonad}^{\uparrow}, \mathsf{CompletionIdempotence}^{\uparrow}, \mathsf{Co}$$

The ledger exposes no quotient completion, selected limit representative, host equality of completed objects, countable-choice schedule, detached $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge-checking row.*

Proof. Apply Definition 3398.2. The public read begins with S, Q, then reaches the completed real-name row E, and only then consumes the monad, idempotence, and universal-property rows U, I, P.

The sibling chapters fix the same dependency route: chapter 301 supplies the regular rational source, chapter 360 supplies the completed real-name surface, and chapter 378 supplies the completion monad handoff.

The structural rows H, C, N preserve and name only that displayed route. Hence no refused host-side principle can escape through the local ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionReflector}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readbacks, a RealCauchyCompletion completed row, a CauchyCompletionMonad handoff, a CompletionIdempotence boundary, a CompletionUniversalProperty extension row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3398.1, Definition 3398.2, Definition 3398.3, Definition 3398.4, and Definition 3398.5 seed the RegSeqRat-specific completion reflector packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality of completed objects, countable choice, detached $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity, completion admission, unit handoff, idempotence boundary, universal-property compatibility, ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionReflectorUp.

Chapter 3399

A Concrete Naming Certificate for $\mathsf{SchwartzFunction}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzFunction}^{\uparrow}$ packet records the finite test-function carrier that exposes rapidly decreasing smooth windows to Fourier and tempered-distribution consumers. It sits over the weak-derivative route of chapter 2210, the finite Fourier observation surface of chapter 124, the tempered-distribution handoff of chapter 2310, the local support and gluing surface of chapter 3369, and the scalar readback of chapter 38. The chapter names only the finite BEDC carrier for test windows, derivative orders, polynomial weights, Fourier-readiness rows, and real seals; it does not import a completed Schwartz topology, host smooth functions, unrestricted seminorm families, Plancherel, inversion, or quotient test-function equality.

3399.1 Carrier definition

Definition 3399.1 (Schwartz function carrier)**.** *A* $\mathsf{SchwartzFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, W, G, F, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite derivative-observation row. It records the named weak-derivative requests, multi-order labels, sign conventions, and local derivative windows read through chapter 2210.*
2. *W is the finite window row. It records the compact or bounded windows on which pointwise reads, derivative reads, and support restrictions are visible.*
3. *G is the polynomial-weight ledger. It records the displayed weight exponents and rational bounds certifying rapid decay for the derivative orders requested in D.*
4. *F is the Fourier-readiness row. It records the finite transform and coefficient windows exported to chapter 124.*
5. *T is the tempered-distribution test handoff row. It exports the test-window inventory consumed by chapter 2310.*
6. *R is the scalar readback row through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, exposing only the finite rational and real endpoints needed by derivative, weight, and transform observations.*

7. *L is the local classifier row. It admits two packets together exactly when their derivative observations, windows, polynomial-weight ledgers, Fourier rows, tempered-distribution test rows, and scalar readbacks match componentwise.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from a Schwartz-test request through D, W, G, F, T, R, L, H in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{PartitionOfUnity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$ *row.*

The classifier compares only these finite rows. It has no coordinate for a completed Schwartz space, an infinite seminorm topology, host smooth-function equality, arbitrary support selection, countable choice, Plancherel, inversion, or quotient test-function equality.

Lean4 ✓ `BEDC.Derived.SchwartzFunctionUp.SchwartzFunctionTasteGate single carrier alignment`

3399.2 The certificate

Theorem 3399.2 (Schwartz function NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *carrier*

$$S = (D, W, G, F, T, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for derivative observations, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming;*
2. *derivative-window exactness, requiring every derivative read to pass through D and W before any weighted or Fourier-facing read is exposed;*
3. *rapid-decay exactness, requiring every displayed polynomial-weight bound to be read from G for a derivative order already named in D;*
4. *Fourier-readiness exactness, requiring every finite transform or coefficient request to pass through F and then through the same scalar readback row R;*
5. *tempered-distribution test exactness, requiring every distribution-facing test row to pass through T after the derivative and weight ledgers have been opened;*
6. hsame, Cont, Pkg, *and local* NameCert *stability for the rows $D, W, G, F, T, R, L, H, C, P, N$.*

No obligation outside these rows is part of the certificate.

Proof. Carrier admission is projection from Definition 3399.1. The accepted tuple displays the derivative observations, local windows, polynomial-weight ledger, Fourier-readiness row, tempered-distribution test handoff, scalar readback, classifier, transport, replay, provenance, and local naming.

The derivative gate is D together with W. A consumer cannot ask for a weighted or transform observation until the finite derivative order and the window on which it is read have both been displayed.

The rapid-decay gate is G. Its bounds are attached to the displayed derivative orders, so the proof uses no infinite seminorm family and no host smooth-function equality.

The Fourier and distribution handoffs are F and T. These rows export only the finite transform windows and test rows already admitted by D, W, G, and the endpoints pass through R before any scalar seal is read. The remaining rows L, H, C, P, N compare, transport, replay, source, and name the same finite inventory. □

Theorem 3399.3 (Schwartz function Fourier stability)**.** *Every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *packet exported to a Fourier-facing consumer factors through*

$$D, W, G \longmapsto F \longmapsto R \longmapsto L, H, C, P, N.$$

Thus the consumer receives derivative windows, polynomial-weight ledgers, finite transform and coefficient windows, scalar readbacks, classifier, transport, replay, provenance, and local naming. It receives no completed Schwartz topology, Plancherel theorem, inversion theorem, quotient test-function equality, selected representative, countable choice, or ambient completeness principle.

Proof. Start with a Fourier-facing read. The derivative row D and window row W must be visible because each finite transform request is attached to a displayed test window and derivative order.

Read the polynomial-weight ledger G next. It is the only row that certifies the rapid-decay bound attached to the requested derivative order, so there is no appeal to an infinite seminorm topology.

The handoff to F is then finite. It records coefficient and transform windows consumable by chapter 124, and its scalar endpoints pass through R before the classifier, transport, replay, provenance, and naming rows are read. □

Theorem 3399.4 (Schwartz function tempered-distribution handoff)**.** *Every accepted* $\mathsf{SchwartzFunction}^{\uparrow}$ *packet exported to a tempered-distribution consumer factors through the displayed derivative row, local window row, polynomial-weight ledger, test handoff row, scalar readback row, classifier, transport, replay, provenance, and local naming:*

$$D, W, G \longmapsto T \longmapsto R \longmapsto L, H, C, P, N.$$

The tempered-distribution consumer therefore receives only the finite test-function rows required by chapter 2310; it receives no completed test-function space, unrestricted continuous dual, quotient test-function equality, host distribution equality, selected representative, or countable-choice schedule.

Proof. Project the accepted carrier of Definition 3399.1. The distribution-facing handoff row T is stored after the derivative, window, and polynomial-weight rows, so it cannot be read without those finite tests already being displayed.

The scalar route remains R. It exposes only the rational and real endpoints needed by the finite test row and does not identify host functions or distributions.

Finally L, H, C, P, N repack the same data as a local NameCert surface. Because no row contains a completed Schwartz topology, unrestricted dual, quotient equality, selected representative, or choice schedule, the handoff is exactly the finite test carrier consumed by the tempered-distribution packet. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzFunction}^{\uparrow}$ consumer that reaches a Fourier window, tempered-distribution test row, weak-derivative observation, or $\mathsf{Real}^{\uparrow}$ scalar seal without first reading the derivative window and polynomial-weight ledger refutes the carrier surface.

Independence witness: weakderivative, fourier, tempereddistribution, partitionofunity, regseqrat, real: none is bijective to $\mathsf{SchwartzFunction}^{\uparrow}$ because this packet jointly carries derivative observations, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzFunction}^{\uparrow}$ packet is generated from finite derivative-observation rows, local windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution test handoff rows, scalar readbacks through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, a local classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3399.1, Definition 3399.2, Definition 3399.3, and Definition 3399.4 seed the finite Schwartz-test carrier over derivative windows, polynomial-weight ledgers, Fourier-readiness rows, tempered-distribution handoff rows, scalar readbacks, classifier, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Schwartz spaces, infinite seminorm topologies, host smooth-function equality, arbitrary support selection, quotient test-function equality, Plancherel, inversion, unrestricted continuous duals, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exactness, rapid-decay exactness, Fourier-readiness exactness, tempered-distribution test exactness, scalar readback stability, transport stability, replay stability, and non-escape proofs for BEDC.Derived.SchwartzFunctionUp.

Chapter 3400

A Concrete Naming Certificate for $\mathsf{AsymptoticEquivalence}^{\uparrow}$

Following Hardy on divergent series and Bishop–Bridges constructive analysis, an $\mathsf{AsymptoticEquivalence}^{\uparrow}$ packet names tail-equivalence data for real-series and regular-Cauchy comparison routes by finite thresholds, displayed tail windows, and terminal Real-facing seals. It sits after the partial-sum and tail-budget surface of chapter 1581, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a reusable comparison certificate for two displayed tails; it is not a quotient of series, not a selected sum, not a host limit, and not an ambient completeness principle.

Definition 3400.1 (Asymptotic equivalence carrier)**.** *An* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are the two* $\mathsf{RealSeries}^{\uparrow}$ *tail-source rows whose partial-sum or term tails are compared.*
2. R_0, R_1 *are the matching* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows used by the two tail names.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for ratio, difference, and tail-budget inspections.*
4. T *is the finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window schedule assigning each requested precision a pair of inspected tail windows.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *comparison-seal row reached after the two readbacks and dyadic ledger have been consumed.*
6. H *records componentwise* hsame *transport for the two series rows, two regular readbacks, dyadic ledger, stream windows, terminal seal, continuation, provenance, and local naming rows.*
7. C *records the displayed* Cont *replay from each precision request through* $T, D, R_0, R_1, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$ *row.*

The carrier exposes only finite tail-comparison data. It has no coordinate for a selected infinite sum, quotient equality of series, host limit selection, trichotomy, countable choice, ambient Real completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.AsymptoticEquivalenceUp.AsymptoticEquivalenceTasteGate single carrier alignment`

Theorem 3400.2 (Asymptotic equivalence NameCert obligations)**.** *For an accepted* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier* $A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$ *surface consists exactly of carrier exposure, two real-series tail-source rows, two regular-Cauchy readback rows, the shared dyadic tolerance ledger, the stream-indexed tail-window schedule, the terminal Real comparison seal, componentwise classifier transport, continuation replay, package provenance, and local naming. Each accepted precision request is classified by the finite tail window in* T *and the tolerance ledger in* D *before either regular readback may expose the terminal row* E*.*

Proof. Project the carrier tuple in Definition 3400.1. The analytic rows are $S_0, S_1, R_0, R_1, D, T, E$, and the structural rows are H, C, P, N.

For a precision request, first read the scheduled finite windows in T and the dyadic tolerance ledger D. These rows determine which tails of the two $\mathsf{RealSeries}^{\uparrow}$ sources and which regular-Cauchy readbacks may be inspected.

The classifier transport is componentwise through H, while C replays the same route from the request to the two readbacks and then to E. The provenance row P and local naming row N record only those displayed dependencies, so the NameCert surface cannot import a selected sum, quotient equality, host limit, choice schedule, or ambient completeness proof. □

Theorem 3400.3 (Asymptotic equivalence tail transfer)**.** *Let* $A = (S_0, S_1, R_0, R_1, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{AsymptoticEquivalence}^{\uparrow}$ *carrier. If a Real or Completion consumer reads a finite tail estimate for* S_0 *through the schedule* T *and ledger* D*, then the same accepted packet exposes the matching finite tail estimate for* S_1 *through the componentwise readback rows* R_0, R_1 *and the terminal comparison seal* E*. The transfer is stable under* hsame *transport of the carrier rows and exports no selected real representative, quotient stream equality, host convergence principle, or completed infinite tail.*

Proof. Start with the accepted schedule row T. It stores a pair of finite tail windows for each precision request, so a consumer that reads the first source tail has already displayed the matching second-source window.

Use the dyadic ledger D to compare the two inspected tails. The readback rows R_0 and R_1 then expose the regular-Cauchy data attached to the two windows, and the terminal row E records the Real-facing comparison seal reached after both readbacks are displayed.

Transport along H preserves the two series rows, two readbacks, ledger, windows, and terminal seal component by component. Replaying C therefore gives the same tail-transfer route to every downstream consumer, while the excluded selected representatives, quotient equalities, host convergence principles, and completed infinite tails remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AsymptoticEquivalence}^{\uparrow}$ packet is generated from two $\mathsf{RealSeries}^{\uparrow}$ tail-source rows, two $\mathsf{RegSeqRat}^{\uparrow}$ readback rows, a shared $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ tail windows, a terminal $\mathsf{Real}^{\uparrow}$ comparison seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3400.1, Definition 3400.2, and Definition 3400.3 seed the finite tail-equivalence route over $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, quotient equality of series, host limits, trichotomy, countable choice, ambient Real completeness, bridge checking, or a Lean theorem.

Upgrade path: A formal route can encode the eleven-row carrier and prove the finite tail-transfer and NameCert obligation surfaces.

Chapter 3401

A Concrete Naming Certificate for $\mathsf{RieszLemma}^{\uparrow}$

Following Riesz's lemma in normed linear spaces, the $\mathsf{RieszLemma}^{\uparrow}$ packet records the finite BEDC witness by which a proper closed subspace and a displayed separation tolerance expose a normed-space separation row. It sits over the norm carrier of chapter 89, the inner-product comparison surface of chapter 86, the metric observation row of chapter 57, the complete-metric boundary of chapter 128, and the Banach carrier of chapter 90. The certificate names a finite separation witness surface; it is not a quotient subspace construction, not a selected best approximation, and not an ambient compactness or completeness theorem.

Definition 3401.1 (Riesz lemma finite separation carrier)**.** *A* $\mathsf{RieszLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, U, A, B, D, S, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the normed-space source row, read through* $\mathsf{Norm}^{\uparrow}$ *and the vector-space carrier already visible to the packet.*
2. *Y is the displayed proper closed subspace row, named as finite membership and closure evidence rather than as a quotient carrier.*
3. *U is the unit normalization row recording the inspected vector window and its norm-one certificate.*
4. *A is the separation tolerance row, with the requested dyadic margin read through the local* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
5. *B is the ball-avoidance row recording the displayed finite comparisons with the subspace window.*
6. *D is the metric-distance observation row, routed through* $\mathsf{Met}^{\uparrow}$ *and the norm-induced distance surface.*
7. *S is the separation witness row that names the finite vector witness after U, A, B, D have been read.*
8. *K is the completeness or Banach-facing source row, available only as a dependency witness from* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{Banach}^{\uparrow}$ *and not as a selection principle.*
9. *H stores componentwise* hsame *transports for the displayed rows.*

10. *C stores the displayed* Cont *replay route from source and subspace rows to the witness row.*
11. *P is the* Pkg *provenance over the normed source, subspace ledger, tolerance, avoidance, distance, witness, transport, replay, and local naming rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for quotient subspace distance, best-approximation selection, closed-ball compactness, countable choice, or ambient Banach-space completeness.

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaTasteGate single carrier alignment`

Theorem 3401.2 (Riesz lemma NameCert obligations)**.** *For every accepted* $\mathsf{RieszLemma}^{\uparrow}$ *carrier* $R = (X, Y, U, A, B, D, S, K, H, C, P, N)$*, the local NameCert surface consists of:*

1. *carrier admission for the finite normed-separation packet of Definition 3401.1;*
2. *source-row exposure through X, with the norm and metric rows read through chapter 89 and chapter 57;*
3. *subspace exposure through Y, with closure information displayed as finite ledger data rather than quotient subspace distance;*
4. *unit and tolerance exposure through U and A before any avoidance or separation row is visible;*
5. *classifier stability through B, D, S, where ball avoidance, metric observation, and separation witness rows are compared componentwise;*
6. *ledger nonescape through K, H, C, P, N, so completeness, transport, replay, provenance, and naming remain inside the same finite packet.*

These obligations do not include a quotient normed space, a nearest-point projection, a selected distance minimizer, countable choice, compactness of closed balls, or an unrestricted Banach-space separation principle.

Proof. Project the carrier rows of Definition 3401.1. The source, subspace, unit, tolerance, avoidance, distance, witness, completeness-source, transport, replay, provenance, and naming rows are the only data visible to the local classifier.

The normed-separation part is finite. The source row X supplies the normed carrier, Y supplies the displayed subspace ledger, and U, A fix the unit normalization and dyadic tolerance before any witness can be read.

The classifier stability part is read through B, D, S. These rows compare avoidance and distance observations componentwise, so the separation witness cannot be replaced by quotient distance or by a selected best approximation.

Finally K, H, C, P, N bind the route to one local certificate. Since the excluded objects are absent from the carrier, no NameCert obligation can escape through them. □

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaCarrier namecert obligations`

Theorem 3401.3 (Riesz lemma normed separation witness)**.** *Every consumer of an accepted* $\mathsf{RieszLemma}^{\uparrow}$ *packet that asks for the Riesz separation witness factors through*

$$X, Y, U, A, B, D, S, K, H, C, P, N.$$

The read consumes the normed source, closed-subspace ledger, unit normalization, tolerance row, ball-avoidance row, metric-distance observation, separation witness row, completeness-source row,

componentwise transport, replay, provenance, and local naming before any witness is exported. The route exports no quotient subspace distance, selected minimizer, host projection, compactness principle, countable choice, or ambient Banach-space completeness coordinate.

Proof. Begin with the NameCert obligations of Definition 3401.2. They place X, Y, U, A before the avoidance and distance rows, so the requested separation witness is grounded in displayed normed-space and subspace data.

Next read B, D, S. The avoidance row, metric observation row, and separation witness row are compared by the local classifier, which makes the witness stable under componentwise equality of the displayed packet.

Finally use the ledger rows K, H, C, P, N. They record that completeness dependencies, transport, continuation replay, provenance, and local naming stay inside the same finite BHist packet. Hence the consumer receives a finite normed separation witness rather than any of the excluded quotient, choice, projection, compactness, or ambient-completeness objects. □

Lean4 ✓ `BEDC.Derived.RieszLemmaUp.RieszLemmaCarrier normed separation witness`

Closure status for $\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszLemma}^{\uparrow}$ packet is generated from a normed-space source row, closed-subspace ledger, unit normalization row, dyadic tolerance, ball-avoidance row, metric-distance observation, separation witness, completeness-source dependency, componentwise transport, continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3401.1, Definition 3401.2, and Definition 3401.3 bind the finite normed-space separation carrier and its witness nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim quotient subspace distance, nearest-point projection, selected distance minimizers, compactness of closed balls, countable choice, full Banach-space separation, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability for the avoidance and distance rows, ledger nonescape, and local NameCert coverage for the displayed normed separation packet.

Chapter 3402

A Concrete Naming Certificate for $\mathsf{SimplicialApproximation}^{\uparrow}$

Following Brouwer, Alexander, and Spanier on the classical simplicial approximation theorem, the $\mathsf{SimplicialApproximation}^{\uparrow}$ packet records the finite combinatorial route by which a displayed continuous map is replaced by a simplicial map after a displayed subdivision. It sits beside the finite complex surface of chapter 238, the polytope realization boundary of chapter 209, the graph and modulus rows of chapter 130, the Brouwer fixed-point route of chapter 2030, and the deformation ledger of chapter 156. The packet names only the finite subdivision, vertex-image, star-containment, and handoff rows visible to BEDC consumers; it does not import host topology, open-cover compactness, quotient homotopy classes, or choice-selected approximating maps.

Definition 3402.1 (Simplicial approximation carrier)**.** *A* $\mathsf{SimplicialApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, L, S, V, F, T, B, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the source* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, including the finite simplex inventory, face relation, and dimension grading.*
2. *L is the target* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, read through the same finite-complex surface.*
3. *S is the subdivision ledger for K, naming the subdivided complex, the face projection back to K, and the carried simplex-refinement relation.*
4. *V is the vertex-image ledger assigning each vertex of the subdivision to a vertex of L.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map from the polytope realization of K to the polytope realization of L.*
6. *T is the* $\mathsf{Polytope}^{\uparrow}$ *realization row, exposing the barycentric and face-containment data used to read stars and simplex interiors.*
7. *B is the finite star-containment ledger asserting that every subdivided simplex of S is sent by V into a target simplex whose open-star row contains the corresponding F-image window.*
8. *M is the Brouwer-facing handoff row, recording the finite approximation surface consumed by* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *arguments that pass through a combinatorial subdivision.*
9. *H records the* $\mathsf{Homotopy}^{\uparrow}$ *handoff row witnessing that the simplicial map encoded by V, B is homotopic to the displayed continuous map along the visible polytope and star rows.*

10. *C records componentwise* Cont *replay from $K, L, S, V, F, T, B, M, H$ through the local certificate rows.*
11. *P is the* Pkg *provenance over* SimplicialApproximation$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, Polytope$^{\uparrow}$, ContinuousMap$^{\uparrow}$, BrouwerFixedPoint$^{\uparrow}$, Homotopy$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient topological space, an arbitrary open cover, a selected infinite subdivision process, a quotient homotopy class, or a host proof of compactness.

Lean4 ✓ `BEDC.Derived.SimplicialApproximationUp.SimplicialApproximationTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.SimplicialApproximationUp.SimplicialApproximationUpTasteGate single carrier alignment`

Theorem 3402.2 (SimplicialApproximationUp NameCert obligations)**.** *For every accepted* SimplicialApproximation *carrier*

$$A = (K, L, S, V, F, T, B, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$ *surface consists exactly of source-complex admission, target-complex admission, subdivision admission, vertex-image admissibility, continuous-map readback, polytope realization readback, star-containment exactness, Brouwer-facing finite-topology handoff, Homotopy-facing deformation handoff, displayed continuation replay, package provenance, and local naming. No obligation is stored outside the rows of Definition 3402.1.*

Proof. Project the carrier. The source and target finite topological data are exactly K and L, while S refines only the displayed source complex and cannot introduce a new ambient space.

The map data are split into two finite ledgers. The continuous row F supplies the BEDC-visible graph and modulus surface, and the vertex-image row V supplies the candidate simplicial map. The polytope row T is the only realization surface on which both can be compared.

Star-containment exactness is therefore the boundary row B. It checks each subdivided simplex against a target simplex and records the finite containment needed by the classical simplicial approximation route.

The remaining rows do not add new analytic content. The Brouwer handoff M, Homotopy handoff H, continuation replay C, provenance P, and local name N expose where the finite approximation may be consumed. Since no other coordinate exists, the listed rows exhaust the NameCert surface. □

Theorem 3402.3 (Simplicial approximation Brouwer handoff)**.** *Every* BrouwerFixedPoint$^{\uparrow}$ *or* Homotopy$^{\uparrow}$ *consumer of an accepted* SimplicialApproximation$^{\uparrow}$ *carrier receives only the finite route*

$$K, L, S, V, F, T, B \longmapsto M, H \longmapsto C, P, N.$$

The route first exposes the source and target simplicial complexes, the displayed subdivision, the vertex-image ledger, the continuous-map row, the polytope realization row, and the star-containment ledger. Only then may a consumer read the Brouwer-facing finite-topology handoff or the Homotopy-facing deformation handoff.

Proof. Start with an accepted carrier $A = (K, L, S, V, F, T, B, M, H, C, P, N)$. By Definition 3402.2, the consumer must read K, L, S, V, F, T, B before any handoff row is visible.

For a Brouwer-facing consumer, M is the only row naming the finite approximation surface. It is already indexed by the displayed subdivision and star-containment ledger, so it cannot replace the finite route by an open-cover compactness argument or a selected host approximant.

For a Homotopy-facing consumer, H is the only deformation row. It relates the simplicial map encoded by V, B to the continuous row F through the visible polytope data T, rather than through a quotient of maps.

Repacking through C, P, N preserves exactly this finite route. Thus both downstream consumers see the same displayed subdivision, vertex, star, realization, and handoff rows, with no additional host-topology coordinate. □

Falsifiable prediction: A $\mathsf{SimplicialApproximation}^{\uparrow}$ consumer that reaches a Brouwer fixed-point or homotopy conclusion without reading the subdivision row S, vertex-image row V, polytope realization row T, and star-containment row B refutes the carrier surface.

Independence witness: simplicialcomplex, polytope, continuousmap, brouwer-fixed-point, homotopy: no listed sibling supplies the whole twelve-row packet $A = (K, L, S, V, F, T, B, M, H, C, P, N)$, because the siblings contribute finite complexes, realization data, continuous graph rows, fixed-point consumers, or deformation ledgers separately while this chapter binds them into one subdivision-and-star certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimplicialApproximation}^{\uparrow}$ packet is generated from source and target SimplicialComplex rows, a subdivision ledger, a vertex-image ledger, a ContinuousMap row, a Polytope realization row, a star-containment ledger, BrouwerFixedPoint and Homotopy handoff rows, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3402.1, Definition 3402.2, and Definition 3402.3 seed the finite subdivision, vertex-image, star-containment, Brouwer handoff, and Homotopy handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimplicialApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topology, open-cover compactness, quotient homotopy classes, selected infinite subdivision processes, general Brouwer fixed-point existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subdivision stability, vertex-image admissibility, star-containment exactness, Brouwer handoff, Homotopy handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SimplicialApproximationUp.

Chapter 3403

A Concrete Naming Certificate for $\mathsf{MooreSmithNet}^{\uparrow}$

Following Moore–Smith and Kelley nets, the $\mathsf{MooreSmithNet}^{\uparrow}$ packet records the finite directed-net carrier that completion and convergence chapters can inspect before any limit, quotient net, or ambient completeness assertion is named. It sits over the directed-index substrate of chapter 1607, the ordinary net-convergence route of chapter 1667, the uniform-completion consumer of chapter 1473, and the filter-base surface of chapter 1601. The packet is a completion-side neighbour for the L10 directed evidence corridor; it supplies a finite net row, not a selected subnet, not a cofinal-choice operator, and not a host-level limit theorem.

Definition 3403.1 (Moore–Smith net carrier)**.** *A* $\mathsf{MooreSmithNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, V, T, F, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index surface, naming finite index vertices, displayed preorder comparisons, and upper-window witnesses consumed by every net read.*
2. *V is the net-value window, recording the finite values visible at the indices named by D.*
3. *T is the tail ledger, recording the finite late-index requests and the displayed upper-window witnesses that answer them.*
4. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained from the displayed tails of D and T.*
5. *U is the completion-facing consumer row for* $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *reads that inspect the same directed tail surface.*
6. *H records componentwise* hsame *transport for the directed index row, value window, tail ledger, filter-base handoff, completion consumer row, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a net request through D, V, T, F, U.*
8. *P is the* Pkg *provenance over* $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$ *row.*

The carrier exports only this finite directed-net surface. It has no coordinate for quotient nets, selected limits, cofinal choice functions, ambient completeness, or host equality of net values.

Lean4 ✓ `BEDC.Derived.MooreSmithNetUp.MooreSmithNetTasteGate single carrier alignment`

Theorem 3403.2 (Moore–Smith net NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *carrier* $M = (D, V, T, F, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the directed-index row, value window, tail ledger, filter-base handoff, completion consumer row, transport, replay, provenance, and local naming row;*
2. *directed-tail coverage, requiring every late-index request to read an upper-window witness from* D *and* T *before any value in* V *is consumed;*
3. *filter-base exposure, requiring the* $\mathsf{FilterBase}^{\uparrow}$ *row* F *to be assembled only from the displayed directed tails;*
4. *convergence and completion handoff, requiring* $\mathsf{NetConvergence}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumers to inspect the same* D, V, T, F *route before reaching* U*;*
5. *ledger refusal, stating that quotient nets, selected limits, cofinal-choice functions, ambient completeness, and host equality are not packet coordinates.*

Proof. Project the accepted carrier of Definition 3403.1. Its analytic rows are D, V, T, F, U, and its structural rows are H, C, P, N.

Directed-tail coverage is read from D and T. A consumer first names a displayed index request, then reads the upper-window witness in the directed set, and only then inspects the corresponding value window V.

The filter-base row F is downstream of the same directed tails. Since F has no independent source coordinate, the filter-facing consumer receives only the finite tail family already displayed by D, T, V.

The completion row U is reached after this route. Componentwise transport through H and replay through C preserve the same rows, while P and N name them as the local certificate surface. No listed coordinate can export quotient nets, selected limits, cofinal choice, ambient completeness, or host equality. □

Theorem 3403.3 (Moore–Smith net Cauchy completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet that needs Cauchy-style completion data first passes through the directed tails and filter-base handoff of chapter 3794 before the broader completion boundary of chapter 2190 may be exposed. The route reads* $D, V, T, F, U, H, C, P, N$ *as finite net data, Cauchy-filter data, and Moore–Smith completion boundary data, and it does not introduce selected limits, quotient nets, cofinal choice, ambient completeness, or host equality.*

Proof. Start with the obligations of Definition 3403.2. They force every completion-facing read to inspect the directed-index row D, value window V, tail ledger T, and filter-base handoff F before reaching the completion consumer row U.

The Cauchy completion neighbour receives exactly that tail and filter route. Thus a reader of chapter 3794 consumes displayed Moore–Smith net data as a Cauchy-style completion surface rather than as a selected limit.

Only after this Cauchy-filter route is visible can the Moore–Smith completion boundary of chapter 2190 be named. Componentwise transport, replay, provenance, and local naming remain H, C, P, N, so the route cannot add quotient nets, cofinal choice, ambient completeness, or host equality. □

Theorem 3403.4 (Moore–Smith net uniform-completion handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ V,\ T \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}.$$

The handoff exposes directed indices, displayed value windows, finite tail requests, filter-base data, and convergence-facing replay only through $D, V, T, F, U, H, C, P, N$*. It does not supply a quotient of nets, a selected completion point, a cofinal-choice function, host equality of values, or ambient uniform completeness.*

Proof. Begin with the NameCert obligations of Definition 3403.2. The directed-tail coverage row forces the consumer to read D and T before any value window can be used.

Next apply the filter-base exposure row. The handoff to $\mathsf{FilterBase}^{\uparrow}$ is assembled from those displayed tails, so it cannot introduce an external filter, ultrafilter, or quotient of net tails.

The convergence and completion handoff row then sends the same finite route to $\mathsf{NetConvergence}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers. Transport and replay remain componentwise through H and C, while P, N record provenance and naming for the same packet. Thus the uniform-completion read is exhausted by the displayed directed-net carrier and cannot import selected limits, cofinal choice, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithNet}^{\uparrow}$ packet is generated from a DirectedSetUp index row, displayed net-value windows, a finite tail ledger, a FilterBaseUp handoff row, NetConvergenceUp and UniformCompletionUp consumer rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$, seed)

Scope closed: Definition 3403.1, Definition 3403.2, and Definition 3403.4 seed the finite Moore–Smith net carrier consumed by completion-side directed evidence.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MooreSmithNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient nets, cofinal choice, ambient completeness, host equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail coverage, filter-base exposure, convergence handoff, uniform-completion handoff, transport preservation, and ledger refusal for the same finite packet.

Chapter 3404

A Concrete Naming Certificate for $\mathsf{BishopCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionCriterion}^{\uparrow}$ packet records the explicit criterion by which a displayed regular Cauchy route is allowed to enter a completion boundary without quotienting names or selecting limits. It sits between the modulus schedule of chapter 1438, the finite window surface of chapter 126, the regular rational readback of chapter 301, the metric completion packet of chapter 1506, the uniform completion packet of chapter 1473, and the terminal real-name seal of chapter 38. The packet is a finite NameCert criterion for completion consumers; it is not a quotient-completion construction, not a selected-representative principle, not countable choice, not open-cover compactness, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3404.1 (Bishop completion criterion carrier)**.** *A* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, W, R, K, U, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row recording the explicit tail schedule inspected by the criterion.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which source observations are read at the displayed precision requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows and tolerance schedule.*
4. *K is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row reached after M, W, R have been displayed.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row reached by the same finite completion route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for Real-facing consumers of the criterion.*
7. *H records componentwise* hsame *transport for the modulus schedule, stream windows, regular readback, metric and uniform completion handoffs, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through M, W, R, K, U, E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$ *row naming the displayed criterion and its permitted consumer reads.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionCriterionUp.BishopCompletionCriterionTasteGate single carrier alignment`

Theorem 3404.2 (Bishop completion criterion NameCert obligations)**.** *For every accepted* BishopCompletionCriteri *carrier $B = (M, W, R, K, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed modulus schedule, stream windows, regular rational readback, metric-completion handoff, uniform-completion handoff, Real seal, transport, continuation replay, provenance, and local naming row;*
2. *explicit Cauchy-modulus exposure, requiring M to be read before any completion boundary K or U is available;*
3. *finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring every completion-facing observation to pass through W and then R at the displayed precision requests;*
4. *completion-seal handoff, requiring K, U, and E to remain downstream of M, W, R and upstream only of the structural rows H, C, P, N;*
5. *ledger non-escape, refusing quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Carrier admission is tuple projection from Definition 3404.1. The accepted packet displays exactly $M, W, R, K, U, E, H, C, P, N$.

The analytic route first reads the explicit modulus schedule M, then the finite windows W, and then the regular rational readback R. The completion rows K and U and the Real seal E are downstream of that displayed route.

Finally H, C, P, N transport, replay, source, and name the same finite criterion. Since quotient limits, selected representatives, countable choice, open-cover compactness, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier row, none can be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3404.1 and Definition 3404.2 seed the finite Bishop completion criterion route from explicit Cauchy moduli through StreamName, RegSeqRat, MetricCompletion, UniformCompletion, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionCriterion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopCompletionCriterionUp.BishopCompletionCriterionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion, selected limits, countable choice, open-cover compactness, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, explicit modulus exposure, finite StreamName and RegSeqRat readback, completion-seal handoff, and ledger non-escape for BEDC.Derived.BishopCompletionCriterionUp.

Constructive story: A $\mathsf{BishopCompletionCriterion}^{\uparrow}$ packet is generated from a CauchyConvergenceModulus tail schedule, finite StreamName windows, RegSeqRat readback, MetricCompletion and UniformCompletion handoffs, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3405

A Concrete Naming Certificate for $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ packet records the finite criterion by which two regular-Cauchy views may be interleaved before a completion-facing consumer reads either view. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, the subsequence criterion of chapter 1552, and the product interleaving route of chapter 2172. The packet is a displayed interleaving criterion for completion consumers; it is not a quotient stream, not a choice-selected limit, and not an ambient completeness coordinate.

Definition 3405.1 (Cauchy interleaving criterion carrier)**.** *A* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows whose observations are to be interleaved.*
2. D_0 *and* D_1 *are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers attached to the two windows.*
3. T *is the finite selector schedule recording which side is read at each displayed stage.*
4. W *is the common window-extraction row obtained from the two scheduled sources and their dyadic ledgers.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reconstructed from the extracted interleaving windows.*
6. E *is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after* $S_0, S_1, D_0, D_1, T, W, Q$ *have been displayed.*
7. H *records componentwise* hsame *transport for the two stream windows, two dyadic ledgers, selector schedule, extracted window, readback row, Real seal, continuation, provenance, and local naming rows.*
8. C *records displayed* Cont *replay through the same interleaving route.*
9. P *is the* Pkg *provenance over* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient streams, selected host limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterleavingCriterionUp.CauchyInterleavingCriterionTasteGate single carrier alignment`

Theorem 3405.2 (Cauchy interleaving criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterleavingCrit}$ *carrier* $I = (S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, two-sided finite-window exposure, two dyadic tolerance ledgers, finite selector admission, common window extraction, regular readback handoff, Real-seal boundary exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *or* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *consumer reads only this displayed route and receives no quotient stream, selected representative, host convergence predicate, countable-choice selector, or ambient completeness witness.*

Proof. Project the carrier of Definition 3405.1. The rows $S_0, S_1, D_0, D_1, T, W, Q, E, H, C, P, N$ are the complete displayed packet.

The analytic rows first expose the two finite stream windows and their dyadic tolerance ledgers. The selector schedule T then chooses the displayed side at each stage, and the common extraction row W records the finite window actually read by the interleaving consumer.

The regular readback Q and Real seal E are downstream of that extracted route. The structural rows H, C, P, N transport, replay, source, and name the same finite criterion, so no quotient stream or choice-selected limit can enter the local NameCert surface. □

Theorem 3405.3 (Cauchy interleaving criterion window extraction)**.** *Every accepted* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *packet admits a finite window extraction route*

$$(S_0, S_1, D_0, D_1, T) \longmapsto W \longmapsto Q \longmapsto E \longmapsto (H, C, P, N).$$

The extracted window W *is read from the two displayed* $\mathsf{StreamName}^{\uparrow}$ *rows using only the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers and the finite selector schedule* T*. Hence downstream* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{BishopCompletionCriterion}^{\uparrow}$*, and regular-Cauchy completion consumers cannot replace the route by a quotient of streams, a selected limit, or an ambient completeness coordinate.*

Proof. Begin with the NameCert obligations of Definition 3405.2. They force both source windows and both dyadic ledgers to be visible before the selector schedule can be consumed.

The selector schedule is finite, so each displayed stage chooses one of the two already exposed windows. The common extraction row W records exactly those finite choices, and the regular readback row Q is reached only after W has been displayed.

Finally the Real seal and structural rows replay the same route. Because the carrier has no coordinate for quotient streams, selected representatives, countable choice, host real equality, or ambient completeness, a completion-facing consumer receives only the finite extraction route shown above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two DyadicRatCore tolerance ledgers, a finite selector schedule, an extracted interleaving window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3405.1, Definition 3405.2, and Definition 3405.3 seed the finite interleaving criterion route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyInterleavingCriterion}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyInterleavingCriterionUp.CauchyInterleavingCriterionTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or a completed-real binary operation.

Upgrade path: Obligation closure requires carrier admission, selector-window coverage, dyadic ledger compatibility, regular readback handoff, Real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyInterleavingCriterionUp.

Chapter 3406

A Concrete Naming Certificate for $\mathsf{LocatedRealModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealModulus}^{\uparrow}$ packet records the finite modulus surface that lets a located real answer bounded precision requests without becoming a quotient real or a trichotomy oracle. It sits over the located-real carrier of chapter 1933, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the density-support route of chapter 1809. It also refines the convergence-modulus discipline of chapter 1438 for the Phase Real and Completion context of chapter 9238.

Definition 3406.1 (LocatedRealModulus carrier)**.** *A* $\mathsf{LocatedRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, D, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *row whose located interval and endpoint enclosure data answer the precision request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger recording the requested radius, endpoint slack, and comparison budget.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened from the located real at the displayed precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite window and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, D, W, R have been displayed.*
6. *H stores componentwise* hsame *transport for the located-real row, dyadic ledger, finite window, regular readback, real seal, replay route, provenance, and local naming row.*
7. *C stores the displayed* Cont *replay from a precision request through L, D, W, R to the seal row E and to downstream completion consumers.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedRealModulus}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, trichotomy, a selected limiting representative, countable choice, host completeness, or an ambient order oracle.

Lean4 ✓ `BEDC.Derived.LocatedRealModulusUp.LocatedRealModulusTasteGate single carrier alignment`

Theorem 3406.2 (LocatedRealModulus NameCert obligations)**.** *For an accepted* $\mathsf{LocatedRealModulus}^{\uparrow}$ *carrier* $M = (L, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, located-real row exposure, dyadic precision-ledger exactness, finite stream-window coverage, regular rational readback compatibility, real-seal non-escape, componentwise transport, continuation replay, provenance, and local naming for the packet of Definition 3406.1.*

Proof. Carrier admission is projection from the displayed tuple. The analytic rows are L, D, W, R, E, and the structural rows are H, C, P, N.

The located-real row L supplies the enclosure surface, while D records the radius and comparison budget for the request. The stream row W and readback row R are opened only at that displayed budget, so the window coverage and rational readback are tied to the same dyadic ledger.

The seal row E is downstream of those reads. Transport and replay preserve the same order through H and C, while P and N add provenance and naming without introducing quotient equality, trichotomy, selected limits, choice, or ambient completeness. □

Theorem 3406.3 (LocatedRealModulus dyadic-window coverage)**.** *Every precision request accepted by a* $\mathsf{LocatedRealModulus}^{\uparrow}$ *packet must display the dyadic ledger* D *before opening the finite stream window* W*, and must display the window* W *before reading the regular rational row* R*. Thus a* $\mathsf{RealSeparability}^{\uparrow}$ *or* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *consumer receives a covered dyadic-window route*

$$D \longmapsto W \longmapsto R$$

inside the same BHist *packet, not a detached density choice, an infinite selector, or a host convergence proof.*

Proof. Start with the carrier inventory of Definition 3406.1. The dyadic ledger precedes the finite-window and readback rows in the accepted packet.

Use the local obligations of Definition 3406.2. They require dyadic precision exactness, window coverage, and regular rational readback compatibility to be certified for the same packet.

The separability and convergence-modulus consumers can therefore cite the displayed route only as finite coverage data. Since the packet contains no density selector, infinite choice schedule, quotient equality, or host convergence coordinate, none can be exported with the covered window. □

Theorem 3406.4 (LocatedRealModulus real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Real and Completion consumer of an accepted* $\mathsf{LocatedRealModulus}^{\uparrow}$ *packet must pass through*

$$L, D, W, R, E, H, C, P, N$$

in that order before the real seal E *is read. The handoff preserves the located-real enclosure, dyadic precision ledger, stream window, and regular readback as visible rows, and it exports no host real equality, trichotomy, selected representative, countable-choice schedule, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the located-real row first. It supplies the enclosure and locatedness data inherited from chapter 1933.

Next read the dyadic-window route of Definition 3406.3. The precision ledger, finite stream window, and regular rational readback are already fixed before the seal row is available.

Finally replay the handoff through C and transport it through H. The provenance and local naming rows record the same packet, so the terminal $\mathsf{Real}^{\uparrow}$ seal is a downstream read of finite modulus data rather than an appeal to quotient completion, choice, trichotomy, selected representatives, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedRealModulus}^{\uparrow}$ packet is generated from a displayed LocatedRealUp row, a DyadicRatCore precision ledger, a StreamName finite-window row, a RegSeqRat readback row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3406.1, Definition 3406.2, Definition 3406.3, and Definition 3406.4 seed the finite located-modulus route over LocatedReal, DyadicRatCore, StreamName, RegSeqRat, RealUp, RealSeparability, and CauchyConvergenceModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, trichotomy, selected limits, countable choice, host completeness, or an ambient RealUp completeness theorem.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-window coverage, regular readback compatibility, real-seal handoff, transport stability, continuation replay, provenance, and local naming for BEDC.Derived.LocatedRealModulusUp.

Chapter 3407

A Concrete Naming Certificate for $\mathsf{MonotoneBoundedSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ packet records the finite carrier surface for a monotone sequence of constructive real names together with an explicit boundedness ledger. It sits between the $\mathsf{Real}^{\uparrow}$ rows of chapter 38, the bounded sequence discipline of chapter 1415, the limit interface of chapter 1413, the tail-modulus handoff of chapter 1438, and the located monotone convergence route of chapter 3373. The packet names the data available before a constructive monotone-convergence consumer asks for a limit; it does not smuggle a completed limit into the source carrier.

Definition 3407.1 (MonotoneBoundedSequence carrier)**.** *A* $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Real}^{\uparrow}$ *source-row family for the sequence values, read only through the finite sequence schedule admitted by the packet.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger recording the inspected indices and observation windows for each displayed sequence value.*
3. *R is the real-readback row that ties each scheduled observation in W to the corresponding* $\mathsf{Real}^{\uparrow}$ *source row in S.*
4. *O is the monotonicity ledger, listing the finite comparison rows by which the packet certifies that later scheduled values do not move below earlier scheduled values.*
5. *B is the boundedness ledger, read through the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *surface, which records the common real bound and the finite comparisons keeping every inspected source row below that bound.*
6. *T is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff row exposed to downstream convergence consumers when the monotonicity and boundedness ledgers have been inspected together.*
7. *H records componentwise* hsame *transport for the source rows, stream windows, real readbacks, monotonicity ledger, boundedness ledger, modulus handoff, replay route, provenance, and local naming row.*
8. *C records displayed* Cont *replay from a convergence request through $S, W, R, O, B, T, H, P, N$.*

9. *P is the* Pkg *provenance over* MonotoneBoundedSequence$^\uparrow$, Real$^\uparrow$, BoundedRealSequence$^\uparrow$, RealSequenceLimit$^\uparrow$, CauchyConvergenceModulus$^\uparrow$, LocatedMonotoneConvergence$^\uparrow$, StreamName$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{MonotoneBoundedSequence}^\uparrow}$ *row.*

The classifier compares accepted packets only through

$$S, W, R, O, B, T, H, C, P, N.$$

It has no coordinate for quotient limits, countable choice, trichotomy, selected representatives, or ambient Real$^\uparrow$ *completeness.*

Theorem 3407.2 (MonotoneBoundedSequence NameCert obligations)**.** *For every accepted* MonotoneBoundedSequ *carrier*

$$M = (S, W, R, O, B, T, H, C, P, N),$$

the local NameCert$_{\text{MonotoneBoundedSequence}^\uparrow}$ *surface consists exactly of the finite rows for carrier admission, source-row scheduling, stream-window readback, monotonicity-ledger admission, boundedness-ledger admission,* CauchyConvergenceModulus$^\uparrow$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each obligation reads the same* Real$^\uparrow$ *source rows and the same* StreamName$^\uparrow$ *windows before the boundedness ledger or modulus handoff is exposed. No obligation may appeal to quotient limits, countable choice, trichotomy, selected representatives, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the carrier in Definition 3407.1. Carrier admission exposes precisely the rows $S, W, R, O, B, T, H, C, P, N$, so every source value is read as a scheduled Real$^\uparrow$ row through the displayed stream windows.

The order-facing rows are then forced to use the same source schedule. The monotonicity ledger O compares only scheduled values from S, W, R, while the boundedness ledger B is the BoundedRealSequence$^\uparrow$ read of the same scheduled source rows against a common bound.

The convergence-facing handoff comes last. The CauchyConvergenceModulus$^\uparrow$ row T may be exposed only after the monotonicity and boundedness ledgers have both been inspected, and the RealSequenceLimit$^\uparrow$ and LocatedMonotoneConvergence$^\uparrow$ consumers can read only that displayed handoff.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, choice, trichotomy, representative-selection, and ambient-completeness coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.MonotoneBoundedSequenceUp.TasteGate.MonotoneBoundedSequenceTasteGate single carrier alignment`

Theorem 3407.3 (MonotoneBoundedSequence scheduled-window factorization)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted MonotoneBoundedSequence$^\uparrow$ *carrier. Every consumer that reads the monotonicity ledger O, boundedness ledger B, and* CauchyConvergenceModulus$^\uparrow$ *row T must factor those reads through the same* RealSequenceSpace$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, *and* Real$^\uparrow$ *schedule S, W, R before any* LocatedSupremum$^\uparrow$ *or* LocatedMonotoneConvergence$^\uparrow$ *read is exposed. The factorization exports one displayed finite sequence schedule, not quotient limits, selected representatives, countable choice, trichotomy, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the carrier in Definition 3407.1. The rows S, W, R are the only sequence-value route available to the packet: S lists the source rows, W records the finite stream windows, and R ties those windows to the real readback surface.

Apply Definition 3407.2. It forces the order-facing rows to consume the same source schedule, so O cannot compare a different sequence surface and B cannot introduce a second boundedness schedule.

The modulus row T is downstream of that shared route. Reading it as a $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff therefore uses the same RealSequenceSpace, StreamName, RegSeqRat, and Real rows before any located-supremum or located-monotone-convergence consumer is allowed to inspect the packet.

Since the carrier has no coordinate for quotient limits, selected representatives, countable choice, trichotomy, or ambient real completeness, the shared-window route cannot export any of those objects. □

Theorem 3407.4 (MonotoneBoundedSequence finite limit boundary)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *or* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer of M must first read the* $\mathsf{Real}^{\uparrow}$ *source rows,* $\mathsf{StreamName}^{\uparrow}$ *windows, monotonicity ledger, boundedness ledger,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *through the same finite surface. The consumer receives a limit-boundary route for monotone bounded data, not a quotient limit, countable-choice-selected tail, trichotomy decision, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a limit-facing consumer. By Definition 3407.1, the only available sequence data are the scheduled $\mathsf{Real}^{\uparrow}$ source rows S, their $\mathsf{StreamName}^{\uparrow}$ windows W, and the readback row R.

Apply Definition 3407.2. The monotonicity ledger O and boundedness ledger B are attached to that same scheduled source surface, and the modulus row T is available only after both ledgers have been displayed.

The $\mathsf{RealSequenceLimit}^{\uparrow}$ route may therefore inspect a finite tail-boundary surface, and the $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ route may inspect the same bounded monotone handoff before reaching its located-supremum data. Neither route can replace the packet by a completed real limit or a host-level sequence theorem.

The structural rows H, C, P, N preserve this boundary under transport, replay, provenance, and naming. The absent quotient-limit, choice, trichotomy, representative, and ambient-completeness coordinates remain absent for every downstream consumer. □

Theorem 3407.5 (MonotoneBoundedSequence located convergence source)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *consumer of M must read the same source rows S, stream windows W, real readback row R, monotonicity ledger O, boundedness ledger B,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff T, transport row H, replay row C, provenance row P, and local naming row N before any* $\mathsf{LocatedSupremum}^{\uparrow}$ *or terminal* $\mathsf{Real}^{\uparrow}$ *seal row is exposed. Thus the located monotone convergence route consumes the displayed bounded monotone source surface rather than a second limit object, quotient representative, choice-selected tail, or ambient completeness theorem.*

Proof. Project the carrier of Definition 3407.1. The tuple exposes exactly the rows $S, W, R, O, B, T, H, C, P, N$, so a located convergence consumer has no other source coordinate to inspect.

Apply Definition 3407.2. The monotonicity ledger, boundedness ledger, and $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff are all attached to the same scheduled source rows and stream windows.

Use Definition 3407.4. It places every $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ read downstream of the displayed source rows, order ledgers, modulus handoff, transport, replay, provenance, and local name.

The located monotone convergence packet can therefore consume only this source route before it reaches a $\mathsf{LocatedSupremum}^{\uparrow}$ handoff or a terminal real seal. Since the carrier has no coordinate for a second limit object, quotient representative, choice-selected tail, or ambient completeness theorem, none can be exposed by the handoff. □

Theorem 3407.6 (MonotoneBoundedSequence Cauchy-modulus frontier)**.** *Let*

$$M = (S, W, R, O, B, T, H, C, P, N)$$

be an accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ *carrier. Every completion-facing consumer that uses the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *handoff row T must first expose the shared* $\mathsf{Real}^{\uparrow}$ *source rows S, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the readback row R, the monotonicity ledger O, the boundedness ledger B, the transport row H, the replay row C, the provenance row P, and the local* $\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}$ *row N. The handoff therefore exports a finite Cauchy-modulus frontier for the displayed bounded monotone source, not a quotient limit, choice-selected tail, trichotomy decision, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 3407.1. The row T is not an independent limit object; it is one coordinate of the displayed tuple $S, W, R, O, B, T, H, C, P, N$.

Apply Definition 3407.2. The $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ handoff is admitted only through the same source rows, stream windows, readback, monotonicity ledger, and boundedness ledger that support the local naming surface.

Use Definition 3407.4 and Definition 3407.5. Together they place every completion-facing or located-convergence read downstream of the displayed carrier surface and before any terminal limit seal.

The structural rows H, C, P, N preserve that route under componentwise transport, replay, provenance, and local naming. Since the carrier has no coordinate for quotient limits, choice-selected tails, trichotomy decisions, selected representatives, or ambient real completeness, the modulus frontier cannot export any of those objects. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ consumer that claims access to a convergence boundary must display the Real source rows, StreamName windows, monotonicity ledger, boundedness ledger, CauchyConvergenceModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert before any RealSequenceLimit or LocatedMonotoneConvergence read is inspected.

Independence witness: boundedrealsequence, realsequencelimit, cauchyconvergencemodulus, locatedmonotoneconvergence, real: $\mathsf{MonotoneBoundedSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a monotonicity ledger, boundedness ledger, and finite modulus handoff before any limit-facing consumer may read the source sequence.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3407.1, Definition 3407.2, Definition 3407.4, and Definition 3407.5 seed the finite source carrier, local obligation surface, limit-boundary route, and located-convergence source handoff for monotone bounded real sequences.

Formal verification: FormalStatus(NameCert$_{\text{MonotoneBoundedSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient limits, countable choice, trichotomy, selected representatives, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, monotonicity-ledger admission, boundedness-ledger admission, CauchyConvergenceModulus handoff, transport replay, provenance, and local naming for BEDC.Derived.MonotoneBoundedSequenc

Constructive story: A MonotoneBoundedSequence$^{\uparrow}$ packet is generated from Real source rows, StreamName windows, real readback, monotonicity and boundedness ledgers, a CauchyConvergenceModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3408

A Concrete Naming Certificate for $\mathsf{AiryFunction}^{\uparrow}$

Following Airy and Olver on the classical Airy differential equation, the $\mathsf{AiryFunction}^{\uparrow}$ packet records the finite certificate route by which Airy-function reads are named through a displayed scalar ODE, coefficient recurrences, real-series windows, and terminal real-name seals. It consumes the real endpoint discipline of chapter 38, the finite partial-sum and tail surface of chapter 1581, the ODE source of chapter 193, and the Picard local-flow route of chapter 1888. The packet is not a host entire-function primitive, an asymptotic expansion theorem, a selected global solution object, or an ambient analytic-completeness claim.

Definition 3408.1 (Airy function carrier)**.** *An* $\mathsf{AiryFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, I, K, T, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the scalar ODE row for the Airy equation $y'' - xy = 0$, read through the local* $\mathsf{Ode}^{\uparrow}$ *source and its displayed derivative, vector-field, and state-window ledgers.*
2. *I is the initial-data row. It records the chosen Airy branch by finitely displayed values for $y(0)$ and $y'(0)$, including the branch tags that distinguish the standard* Ai *and* Bi *reads when such a distinction is requested.*
3. *K is the coefficient-recurrence row. It records the finite recurrence ledger by which coefficient windows for the Airy power series are generated from D and I.*
4. *T is the* $\mathsf{RealSeries}^{\uparrow}$ *term-stream row, listing the finite partial-sum windows, tail requests, and shared threshold rows used by the requested Airy read.*
5. *S is the local Picard or ODE nonescape row, ensuring that the inspected series and flow windows stay inside the displayed scalar state window supplied by* $\mathsf{PicardLindelof}^{\uparrow}$*.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for coefficients, partial sums, initial values, and endpoint requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the ODE, initial-data, recurrence, series, nonescape, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the ODE row, initial data, recurrence ledger, series windows, nonescape row, readbacks, Real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from an Airy request through the ODE row, initial data, coefficient recurrence, real-series windows, Picard nonescape, regular readbacks, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{AiryFunction}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$ *row.*

The carrier exports only this finite route. It has no coordinate for a host entire function, global analytic continuation, asymptotic sector decomposition, selected zero set, quotient function equality, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.AiryFunctionUp.AiryFunctionTasteGate single carrier alignment`

Theorem 3408.2 (Airy function NameCert obligations)**.** *For every accepted* $\mathsf{AiryFunction}^{\uparrow}$ *carrier* $A = (D, I, K, T, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed ODE, initial-data, recurrence, real-series, nonescape, readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *ODE compatibility, requiring D to be the only scalar Airy equation row read by the packet;*
3. *initial-data compatibility, requiring I to supply the branch-determining values consumed by the recurrence ledger;*
4. *recurrence exactness, requiring K to generate every inspected coefficient window from D and I before any series read is exposed;*
5. *real-series visibility, requiring T to expose partial sums, tail thresholds, and endpoint requests only through the* $\mathsf{RealSeries}^{\uparrow}$ *surface;*
6. *local-flow nonescape, requiring S to bind the inspected ODE and series windows to the Picard local-flow route;*
7. *endpoint sealing, requiring R and E to read every requested coefficient, partial sum, and Airy endpoint through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows;*
8. *structural preservation, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no external analytic object.*

No Airy-function obligation is stored outside these rows.

Proof. Project the carrier of Definition 3408.1. The ODE row D and initial-data row I are visible before any recurrence or series row can be inspected, so the requested branch is fixed inside the packet.

The coefficient and series obligations are then finite ledger checks. The recurrence row K supplies the inspected coefficient windows, and the $\mathsf{RealSeries}^{\uparrow}$ row T supplies only the partial sums, tail thresholds, and endpoint requests listed in the packet.

The local-flow and endpoint rows close the analytic route. The nonescape row S keeps the inspected windows inside the displayed Picard or ODE state window, while R and E provide regular rational readback and the terminal real-name seal.

Finally H, C, P, N transport, replay, source, and name the same finite data. Since the carrier has no coordinate for a host entire function, asymptotic sector, selected zero set, quotient function equality, or ambient completeness theorem, the local NameCert surface has no obligation row for such an object. □

Theorem 3408.3 (Airy function real-series handoff)**.** *Every Real-facing read from an accepted* $\mathsf{AiryFunction}^{\uparrow}$ *carrier factors through*

$$D, I \longmapsto K \longmapsto T, S \longmapsto R, E \longmapsto H, C, P, N.$$

Thus a consumer receives the displayed Airy ODE row, branch initial data, coefficient recurrence ledger, finite real-series windows, local nonescape row, regular readback, and terminal real seal, but receives no host entire-function primitive or global analytic continuation datum.

Proof. Start with an accepted packet $A = (D, I, K, T, S, R, E, H, C, P, N)$. The only ODE and branch data available to a consumer are D and I, so a Real-facing read begins at the displayed Airy equation and its carried initial values.

The recurrence row K is the next forced coordinate. It generates the inspected coefficient windows from the ODE and branch data before the $\mathsf{RealSeries}^{\uparrow}$ row T exposes partial sums or tail thresholds.

The nonescape and endpoint rows are downstream. The row S keeps the displayed series and local-flow windows inside the accepted ODE route, and R, E provide the regular readback and terminal $\mathsf{Real}^{\uparrow}$ seal.

The structural rows H, C, P, N preserve and name only that route. Hence the handoff cannot introduce a host entire function, a global continuation object, an asymptotic sector decomposition, a selected zero set, or quotient function equality. □

Falsifiable prediction: Every $\mathsf{AiryFunction}^{\uparrow}$ read factors through one displayed Airy ODE row, one branch initial-data row, one coefficient-recurrence ledger, one $\mathsf{RealSeries}^{\uparrow}$ partial-sum and tail surface, one Picard nonescape row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: real, realseries, ode, picard-lindelof: $\mathsf{AiryFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an Airy ODE row, branch initial data, coefficient recurrence, real-series windows, local-flow nonescape, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AiryFunction}^{\uparrow}$ packet is generated from a displayed Airy ODE row, branch initial data, coefficient-recurrence ledger, RealSeriesUp partial-sum and tail rows, Picard nonescape data, RegSeqRat readbacks, RealUp endpoint seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$, seed)

Scope closed: Definition 3408.1, Definition 3408.2, and Definition 3408.3 seed the finite Airy-function certificate surface over displayed ODE, branch, recurrence, series, nonescape, readback, and real endpoint rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AiryFunction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host entire functions, global analytic continuation, asymptotic expansions, selected zero sets, quotient function equality, ambient analytic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ODE compatibility, branch-data compatibility, recurrence exactness, real-series visibility, nonescape, endpoint sealing, and structural preservation for BEDC.Derived.AiryFunctionUp.

Chapter 3409

A Concrete Naming Certificate for $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ packet names the finite comparison surface between a Cauchy presentation of the real line and a located lower-upper presentation. It sits between the Cauchy path of chapter 38, chapter 301, chapter 126, and chapter 291, and the located path of **??**, chapter 404, chapter 1446, and chapter 1447. The packet compares finite observation windows only; it does not add quotient cuts, host real equality, countable choice, trichotomy, or an ambient completeness principle.

Definition 3409.1 (Cauchy-located equivalence carrier)**.** *A* $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (C, L, U, W, R, D, E, H, Ct, P, N)$$

with the following rows:

1. *C is the displayed* $\mathsf{Real}^{\uparrow}$ *Cauchy seal, read through the regular rational and stream-name surface of* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*;*
2. *L is the located lower-observation row, read through* $\mathsf{LowerReal}^{\uparrow}$ *and the lower side of* $\mathsf{LocatedCut}^{\uparrow}$*;*
3. *U is the located upper-observation row, read through* $\mathsf{UpperReal}^{\uparrow}$ *and the upper side of* $\mathsf{LocatedCut}^{\uparrow}$*;*
4. *W is the shared finite-window ledger comparing Cauchy tails, dyadic tolerances, lower observations, and upper observations;*
5. *R records the regular-sequence readback from C into the window ledger;*
6. *D records the dyadic-rational tolerance and endpoint rows consumed by W;*
7. *E records the two handoff routes: Cauchy data to located observations, and located observations back to a Cauchy seal;*
8. *H stores componentwise* hsame *transport for all displayed rows;*
9. *Ct stores the displayed* Cont *replay routes used by the two handoffs and by shared-window comparison;*
10. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{LocatedCut}^{\uparrow}$*,* $\mathsf{LowerReal}^{\uparrow}$*,* $\mathsf{UpperReal}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, L, U, W, R, D, E, H, Ct, P, N$. No coordinate stores quotient-cut equality, host real equality, selected limits, trichotomy, countable choice, or a global completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyLocatedRealEquivalenceUp.TasteGate.CauchyLocatedRealEquivalenceTasteGate single carrier alignment`

Theorem 3409.2 (Cauchy-located equivalence NameCert obligations)**.** *For every accepted* CauchyLocatedRealEqui *carrier $Q = (C, L, U, W, R, D, E, H, Ct, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the Cauchy seal, lower and upper located observations, shared-window ledger, regular readback, dyadic tolerance row, handoff row, transport, continuation, provenance, and local naming row;*
2. *Cauchy-to-located handoff, requiring reads from C through R, D, W, E, H, Ct before lower and upper observations are exported as L, U;*
3. *located-to-Cauchy handoff, requiring compatible lower and upper observations in L, U to pass through the same shared windows W, D, E, H, Ct before the Cauchy seal C is read;*
4. *classifier stability under shared finite windows, requiring agreement on W, R, D, E, H, Ct to preserve both directions of comparison;*
5. $\mathsf{Real}^{\uparrow}$*-seal nonescape, excluding quotient cuts, host real equality, selected limits, trichotomy, countable choice, and ambient completeness from the exported comparison.*

No obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3409.1. The row list displays exactly the Cauchy seal, the lower and upper located observations, the shared-window ledger, the regular and dyadic readback rows, the handoff row, and the structural rows H, Ct, P, N.

For the Cauchy-to-located handoff, read C through the regular sequence row R and the dyadic row D. The shared-window ledger W is the only place where Cauchy tail information is compared with lower and upper observations, and E records the finite route from that comparison to L, U.

For the located-to-Cauchy handoff, begin with compatible reads in L and U. The same ledger W and dyadic row D supply the finite tolerance comparison, while E and Ct replay the route back to the Cauchy seal C.

Classifier stability is componentwise. Agreement on the shared-window, readback, tolerance, handoff, transport, and continuation rows preserves the two handoff routes because no other classifier coordinate is available.

Finally, the provenance and naming rows mention only the listed sibling chapters and BEDC support rows. Since quotient cuts, host real equality, selected limits, trichotomy, countable choice, and ambient completeness are not coordinates of Q, the $\mathsf{Real}^{\uparrow}$ seal cannot export them through this packet. □

Theorem 3409.3 (Cauchy-located equivalence finite handoff)**.** *Every accepted* $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ *carrier $Q = (C, L, U, W, R, D, E, H, Ct, P, N)$ routes comparison between the Cauchy* $\mathsf{Real}^{\uparrow}$ *seal and the located* $\mathsf{LowerReal}^{\uparrow}$-$\mathsf{UpperReal}^{\uparrow}$ *observations through the finite surface*

$$C, L, U, W, R, D, E, H, Ct, P, N.$$

The route may be consumed by $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, *and* $\mathsf{UpperReal}^{\uparrow}$ *readers, but it exports no quotient-cut equality, host real equality, selected limit, trichotomy, countable choice, or ambient completeness.*

Proof. Start with the carrier obligations. They force both directions of comparison to use W, R, D, E, H, Ct, so a consumer cannot jump directly from a Cauchy seal to a located cut or from a located cut to a Cauchy seal.

The Cauchy siblings supply the source rows C, R, D, while the located siblings supply L, U and their cut-facing reads. The handoff row E is the only coordinate that compares the two sides, and the continuation row Ct is the only replay route.

Nonescape follows from the final obligation of Definition 3409.2. Since the forbidden data are absent from the carrier, the finite handoff cannot export them to either side of the comparison. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}$ packet is generated from a Cauchy $\mathsf{Real}^{\uparrow}$ seal, lower and upper located observations, shared finite windows, regular and dyadic readback rows, two handoff routes, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3409.1, Definition 3409.2, and Definition 3409.3 seed the finite comparison surface between Cauchy and located real presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocatedRealEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cuts, host real equality, selected limits, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, both handoff directions, shared-window classifier stability, and Real-seal nonescape for BEDC.Derived.CauchyLocatedRealEquivalenceUp.

$\mathsf{Cauchy}^{\uparrow}$ is an L10 route whose regular-sequence, finite-window, dyadic-tolerance, and terminal seal reads are supplied through chapter 301, chapter 126, chapter 291, and chapter 38.

Chapter 3410

A Concrete Naming Certificate for $\mathsf{CauchyRealLocator}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRealLocator}^{\uparrow}$ packet records the finite locator surface by which a displayed Cauchy real answers Real-facing precision and side-location requests. It refines the modulus route of chapter 1423 and chapter 292, reads finite observations through chapter 126, uses the regular rational handoff of chapter 301, records dyadic tolerances through chapter 291, and reaches the terminal seal of chapter 38. The packet is a traditional constructive-analysis locator object reconstructed as finite NameCert data; it does not select a host limit, identify quotient streams, invoke countable choice, decide trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3410.1 (Cauchy real locator carrier)**.** *A* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row. It records the displayed RealCauchyModulus and CauchyModulus threshold data used for every locator request.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the locator at the requested precision and side test.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the positive dyadic scale, comparison cells, and slack budget for the displayed windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows into regular rational data before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *locator seal. It records only the Real-facing locator boundary delivered after M, S, D, Q have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, stream windows, dyadic ledger, regular readback, terminal locator seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a locator request through M, S, D, Q, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$*, RealCauchyModulus, CauchyModulus,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $M, S, D, Q, R, H, C, P, N$. It has no coordinate for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 3410.2 (Cauchy real locator NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier*

$$L = (M, S, D, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the modulus schedule, finite stream windows, dyadic tolerance ledger, regular rational readback, terminal Real locator seal, transport, replay, provenance, and local naming;*
2. *modulus exactness, requiring every locator request to use the displayed row M rather than an independent threshold, host convergence theorem, or selected limit;*
3. *window and tolerance compatibility, requiring the inspected windows in S to be compared only through the dyadic ledger D;*
4. *RegSeqRat readback, requiring the rational data consumed by the locator seal to pass through the displayed row Q;*
5. *route ordering, requiring R to be read only after M, S, D, Q have been displayed;*
6. *nonescape, excluding host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, and bridge checking.*

No local locator obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3410.1. The accepted packet displays exactly the modulus schedule, finite windows, dyadic ledger, regular readback, terminal locator seal, and structural rows.

Modulus exactness is forced by the single coordinate M. Every locator request reads its threshold from this row, so the packet has no second convergence witness, hidden host completeness theorem, or selected limit coordinate.

Window and tolerance compatibility are supplied by S and D. The stream row exposes only finite windows, while the dyadic row records the positive scale, comparison cells, and slack budget used to interpret those windows.

The readback and ordering clauses follow from Q and R. The regular rational row is the only rational readback coordinate, and the terminal Real locator seal occurs downstream of M, S, D, Q.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier inventory contains no row for host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3410.3 (Cauchy real locator modulus route)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier factors each locator request through the displayed finite route*

$$M \longmapsto S, D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Thus the request reaches the $\mathsf{Real}^{\uparrow}$ *locator seal only through the visible modulus schedule, finite stream windows, dyadic tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*

Proof. Begin with an accepted carrier $L = (M, S, D, Q, R, H, C, P, N)$ and a displayed locator request. The only threshold coordinate available to the request is M, so the modulus schedule is read first.

The finite observation step is next. The request opens the window row S and interprets its tolerance through D, whose dyadic ledger records the positive scale and comparison cells.

The rational handoff is then forced through Q. No independent stream or host rational approximation is present in the carrier, so the locator request reaches the Real-facing boundary only after the regular rational readback has been displayed.

Finally the terminal row R is consumed and the structural rows H, C, P, N preserve the same route. This proves the stated factorization and excludes a bypass through selected limits, quotient streams, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Theorem 3410.4 (Cauchy real locator Real seal nonescape)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$ *locator-seal read reaches* R *only after the displayed rows* M, S, D, Q *have been consumed. The seal exports no host Real equality, quotient stream representative, selected limit, countable-choice selector, trichotomy branch, open-cover compactness witness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Inspect the terminal coordinate of the carrier. The only Real-facing locator seal is R, and it is listed after the modulus schedule, stream-window, dyadic-ledger, and regular-readback rows.

Apply the obligation surface of Definition 3410.2. It requires route ordering through M, S, D, Q before R can be read by a Real-facing consumer.

The structural rows preserve this boundary. The hsame transports and Cont replay keep the same displayed packet, while Pkg provenance and local naming record only that route.

Because no excluded coordinate appears in L, the terminal seal cannot export host Real equality, quotient representatives, selected limits, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. Any consumer asking for one of those principles must leave the local carrier. □

Theorem 3410.5 (Cauchy real locator four-object exit)**.** *Every accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *Real-facing locator read factors through exactly the four L10 exit objects named by Definition 3412.8: the* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{Real}^{\uparrow}$ *locator seal. In carrier coordinates, any accepted read of* $L = (M, S, D, Q, R, H, C, P, N)$ *uses* S, D, Q, R *as the four object faces and uses* M, H, C, P, N *only as the modulus schedule, transport, replay, provenance, and local naming support. It exports no quotient stream equality, host Real equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier tuple. The row S is the finite $\mathsf{StreamName}^{\uparrow}$ window, D is the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, Q is the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and R is the terminal $\mathsf{Real}^{\uparrow}$ locator seal.

The modulus route theorem Definition 3410.3 fixes the order of consumption. A locator request reads the schedule M, opens the finite windows S, interprets them through the dyadic ledger D, passes through the regular readback Q, and only then reaches R.

The Real-seal nonescape theorem Definition 3410.4 supplies the boundary condition for the terminal read. It rules out every Real-facing bypass that would identify quotient streams, appeal to host Real equality, choose a limit, branch by trichotomy, use open covers, or import ambient completeness.

Finally compare this carrier projection with Definition 3412.8. The four object faces are exactly S, D, Q, R, while M, H, C, P, N supply only schedule, transport, replay, provenance, and naming support. Hence an accepted locator read has the claimed four-object L10 exit route and no additional exported coordinate. □

Theorem 3410.6 (Cauchy real locator induction route)**.** *For an accepted* $\mathsf{CauchyRealLocator}^{\uparrow}$ *carrier* $L = (M, S, D, Q, R, H, C, P, N)$*, every locator-side read available to induction is obtained by the finite route*

$$M \longmapsto S \longmapsto D \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

The induction step reads finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal rows without deciding trichotomy and without choosing a host limit.*

Proof. Project the carrier of Definition 3410.1. The modulus schedule M is the only threshold source, so a locator-side induction step begins by reading the same schedule used in the modulus route.

The finite observation is then S interpreted through D. The stream row exposes only bounded windows, while the dyadic row supplies the tolerance scale and comparison cells needed to use those windows.

The rational and Real-facing parts follow through Q and R. The modulus route theorem Definition 3410.3 fixes this order, and the four-object exit theorem Definition 3410.5 identifies S, D, Q, R as the four L10 exit faces.

Finally H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the carrier has no coordinate for trichotomy or selected host limits, the induction route cannot decide a side by trichotomy or choose a limit outside the displayed locator data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3410.1, Definition 3410.2, Definition 3410.3, Definition 3410.4, Definition 3410.5, and Definition 3410.6 seed the finite locator route over Cauchy modulus schedules, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and the Real locator seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Real equality, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exactness, window and tolerance compatibility, RegSeqRat readback, route ordering, nonescape, and local NameCert registration for BEDC.Derived.CauchyRealLocatorUp.

Constructive story: A $\mathsf{CauchyRealLocator}^{\uparrow}$ packet is generated from a RealCauchyModulusUp and CauchyModulusUp schedule row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp locator seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3411

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 3411.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3411.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 3411.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 3411.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3411.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 3411.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 3411.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3412

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3412.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through $X, Y, W, D, A, M, S, H, P, N$.*
9. *P is the* Pkg *provenance over* CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 3412.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* CauchySequenceAlgebra$^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *operation budgets,* RegularCauchyAddition$^{\uparrow}$ *route,* RegularCauchyProduct$^{\uparrow}$ *route,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* RegSeqRat$^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 3412.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 3412.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 3412.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 3412.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Cauchy$^{\uparrow}$ gateway reads its completion-facing source through chapter 126, chapter 301, chapter 291, and chapter 38 before any terminal seal is public.

Theorem 3412.7 (Cauchy L10 sibling route)**.** *The Cauchy packet family exported by this hub is consumed through the L10 route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}.$$

Concretely, the finite-window surface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the completion boundary of chapter 1506 are sibling inputs to the Cauchy carriers rather than hidden ambient structure.

Proof. Read any Cauchy carrier exported by the hub through its finite observation rows. The stream coordinate supplies the inspected windows, the regular-rational coordinate supplies the numerical readback, and the dyadic coordinate supplies the tolerance budget.

The Real-facing seal is reached only after those rows have been displayed. When a completion consumer is present, it receives the same displayed route through the metric-completion boundary.

Thus the hub's Cauchy packet family is attached to the Real and MetricCompletion siblings by visible L10 rows, with no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient completeness coordinate added by the route. □

The Cauchy$^{\uparrow}$ route consumes the L10 sibling chapters before the real locator and sequence algebra inputs are read. In particular, the regular-rational source of chapter 301, the finite stream windows of chapter 126, and the terminal seal of chapter 38 are already part of the displayed dependency spine for every downstream Cauchy locator or algebra consumer. The completion-side consumer in chapter 1937 reads the same RegSeqRat, StreamName, and RealUp spine before exposing its functorial completion route.

Theorem 3412.8 (Cauchy L10 four-object seal route)**.** *Every accepted* Cauchy$^{\uparrow}$ *Real-facing seal route used by the current L10 phase factors through the four displayed sibling faces*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Concretely, the finite window family of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38 are the only Real-facing sibling rows exported by the route. The supporting rows are the displayed BHist *carrier data, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *registration.*

Proof. Read the route through the existing CauchyRealLocator carrier and its Real-seal nonescape theorem. The carrier exposes a finite StreamName window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, and a terminal RealUp locator seal before any Real-facing consumer can read the packet.

Project those four named faces. The StreamName row supplies only inspected finite windows, the DyadicRatCore row supplies the tolerance and comparison budget, the RegSeqRat row supplies the rational readback, and the RealUp row supplies the terminal seal.

The structural rows preserve this factorization. The BHist data name the carried packet, hsame transports the displayed coordinates, Cont replays the same route, Pkg records provenance, and NameCert records the local accepted surface.

Since the carrier has no coordinate for quotient streams, host equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects can be exported by the Cauchy Real-facing seal route. □

Chapter 3413

A Concrete Naming Certificate for $\mathsf{Cauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{Cauchy}^{\uparrow}$ packet records the ordinary constructive Cauchy-real carrier as finite BEDC naming data. It sits above the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational sequence surface of chapter 301, and the terminal seal of chapter 38. The packet is not a quotient construction of reals; it is a finite route by which a modulus, rational readback, transport, replay, provenance, and local naming row become visible before any $\mathsf{Real}^{\uparrow}$ consumer can read the Cauchy object.

Definition 3413.1 (Cauchy carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the finite windows of rational approximants inspected by the carrier.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the positive dyadic scale, comparison cells, and slack conversions used to interpret each requested window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational readback consumed downstream.*
4. *M is the modulus row, recording the displayed threshold assignment used by Cauchy requests and refusing any hidden convergence theorem.*
5. *Q is the modulus/readback compatibility row, requiring the stream window, dyadic tolerance, regular rational sequence, and modulus rows to agree at every displayed precision request.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after S, D, R, M, Q have been displayed.*
7. *H records componentwise* hsame *transport for the stream, dyadic, regular-sequence, modulus, compatibility, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy request through $S, D, R, M, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $S, D, R, M, Q, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyUp.CauchyCarrier`

Theorem 3413.2 (Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *carrier*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 3413.1.*

Proof. Carrier admission is projection from Definition 3413.1. The accepted packet displays precisely the stream, dyadic, regular-sequence, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

The stability and exactness obligations are rowwise. The stream row supplies only finite windows, the dyadic row supplies only displayed tolerance cells, the regular-sequence row supplies only rational readback, and the modulus row supplies only the visible threshold assignment.

The compatibility row Q binds these four data rows before any $\mathsf{Real}^{\uparrow}$ seal can be consumed. The structural rows H, C, P, N then transport, replay, source, and name the same route without adding quotient equality, selected limits, host real equality, choice, trichotomy, compactness, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCarrier namecert obligations`

Theorem 3413.3 (Cauchy modulus readback route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer route factors through the displayed path*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a precision request reads finite $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, the explicit modulus row, the compatibility row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal-facing boundary.*

Proof. Begin with an accepted carrier $X = (S, D, R, M, Q, E, H, C, P, N)$ and a displayed precision request. The only finite observations available to the request are the windows in S, and their tolerance scale is read through D.

The request next reads the explicit modulus row M. The compatibility row Q is the only coordinate that relates the chosen window, tolerance, threshold, and regular-rational readback, so the route cannot bypass it.

After Q is displayed, the consumer may read the $\mathsf{RegSeqRat}^{\uparrow}$ row R and then the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row E. The rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming.

Since the carrier has no quotient, selected-limit, host equality, countable-choice, trichotomy, open-cover, or ambient-completeness coordinate, the displayed route is the whole public Cauchy readback surface. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusReadbackRoute`

Theorem 3413.4 (Cauchy StreamName RegSeqRat window)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet exposes its* $\mathsf{RegSeqRat}^{\uparrow}$ *row only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and modulus/readback compatibility row. Thus a regular rational read of* $X = (S, D, R, M, Q, E, H, C, P, N)$ *must factor through* S, D, M, Q *before it can inspect* R*.*

Proof. Project the carrier of Definition 3413.1. The row S is the only finite observation surface, D is the only displayed tolerance surface, and M is the only modulus row available to the packet.

Apply Definition 3413.2 to identify Q as the compatibility row binding stream windows, dyadic tolerances, regular rational readback, and modulus data. Then apply Definition 3413.3: the displayed route reads S, D, M and Q before R.

The carrier has no quotient stream, selected limit, host real equality, countable choice, trichotomy, or ambient completeness coordinate. Hence the stated window and compatibility rows exhaust the public access path to $\mathsf{RegSeqRat}^{\uparrow}$. □

Theorem 3413.5 (Cauchy finite-tail criterion)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

be an accepted $\mathsf{Cauchy}^{\uparrow}$ *carrier. Suppose every displayed finite* $\mathsf{StreamName}^{\uparrow}$ *window admitted by* S *and every dyadic tolerance cell admitted by* D *satisfy the tail bound named by the modulus row* M*, and suppose the compatibility row* Q *identifies those windows with the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*. Then every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of* X *is routed through the same finite tail ledger*

$$S, D, M \longmapsto Q \longmapsto R$$

before the $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed. The criterion exports no selected limit, quotient stream equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows available for a tail read are the finite StreamName window row S, the DyadicRatCore tolerance row D, the modulus row M, the compatibility row Q, the RegSeqRat readback row R, and the terminal seal row E.

Induct over the displayed window list retained by S. The empty window has no readback outside the recorded tolerance ledger. In the successor step, the new inspected position is checked against the dyadic cell in D and the threshold supplied by M, so the same finite tail bound remains attached to the extended window.

Apply Definition 3413.4. It forces any access to R through the displayed S, D, M, Q route. Apply Definition 3413.3 to identify Q as the point where the window, tolerance, modulus, and readback rows are jointly inspected.

Finally use Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal E is downstream of the regular rational readback, so no Real-facing consumer can bypass the finite tail ledger. Since the carrier has no selected-limit, quotient-stream, countable-choice, trichotomy, or ambient-completeness coordinate, the displayed finite induction exhausts the criterion. □

Definition 3413.6 (Cauchy finite-window modulus carrier)**.** *The finite-window modulus carrier associated to an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet is the displayed handoff surface*

$$W = (S, D, M, Q, R, E, H, C, P, N)$$

read from the carrier of Definition 3413.1. Its rows have the following roles:

1. *S records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by a tail request.*

2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells and slack conversions attached to those windows.*
3. *M records the explicit modulus threshold row used to accept later-window requests.*
4. *Q records the compatibility row tying each displayed window, dyadic tolerance, modulus threshold, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback.*
5. *R records the regular-rational readback exposed only after S, D, M, Q have been displayed.*
6. *E records the downstream* $\mathsf{Real}^{\uparrow}$ *seal-facing row.*
7. *H, C, P, N record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming for the same finite route.*

The carrier admits only these displayed coordinates. It has no coordinate for quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3413.7 (Cauchy finite-window tail induction)**.** *Let $W = (S, D, M, Q, R, E, H, C, P, N)$ be the finite-window modulus carrier of Definition 3413.6. Every accepted tail request obtained by extending the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row preserves the dyadic tolerance row D, the modulus threshold row M, the compatibility row Q, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. Thus the induction route is*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E$$

with H, C, P, N transporting, replaying, sourcing, and naming the same displayed handoff.

Proof. Project the underlying accepted Cauchy carrier through Definition 3413.1 and read it in the window/modulus order specified by Definition 3413.6. The only tail-request rows available are $S, D, M, Q, R, E, H, C, P, N$.

Induct over the displayed $\mathsf{StreamName}^{\uparrow}$ windows retained by S. The empty window has no tolerance request outside D and no readback outside the route named by Definition 3413.4.

For the successor step, the new inspected window is accepted only through the dyadic cell in D and the threshold supplied by M. Apply Definition 3413.5 to keep the same finite tail ledger attached to the extended request.

The compatibility row Q is then the unique displayed row where the window, dyadic tolerance, modulus threshold, and regular readback are jointly inspected. Apply Definition 3413.3 to place R after Q and E after R.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Since none of the rows supplies quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the induction cannot expose any such principle. □

Theorem 3413.8 (Cauchy RegSeqRat handoff obligation)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the finite-window modulus carrier of Definition 3413.6 and the tail-induction route of Definition 3413.7. In particular, the consumer reads S, D, M, Q before R, and any Real-facing use reads E only after that* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff.*

Proof. Begin with an accepted Cauchy packet and project its rows by Definition 3413.1. The carrier supplies finite stream windows, dyadic tolerances, a modulus row, a compatibility row, regular-rational readback, a terminal Real-facing seal, and structural transport, replay, provenance, and naming rows.

Apply Definition 3413.4. It forces access to the $\mathsf{RegSeqRat}^{\uparrow}$ row through the displayed S, D, M, Q route, so no consumer can inspect R independently of the finite window and tolerance data.

Next apply Definition 3413.5 and Definition 3413.7. These identify the finite tail ledger and show that accepted extensions preserve the dyadic tolerance, modulus threshold, compatibility, and regular readback rows before the Real seal is exposed.

Finally use Definition 3413.3. The structural rows H, C, P, N transport and name the same handoff, while the carrier contains no quotient-stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ consumer factors through the stated finite-window induction surface. □

Theorem 3413.9 (Cauchy RegSeqRat Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before the terminal real seal. In the notation of Definition 3413.1, the seal-facing route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

It does not expose quotient streams, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $X = (S, D, R, M, Q, E, H, C, P, N)$ from Definition 3413.1. The seal row E is declared terminal and seal-facing; it is not an independent source of Cauchy data.

Use Definition 3413.3 to order the displayed read. The route reaches R after S, D, M, Q and reaches E only after R has been displayed.

Finally apply Definition 3413.2. The rows H, C, P, N transport, replay, source, and name the same route, so they cannot insert quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient completeness between $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 3413.10 (Cauchy modulus transport stability)**.** *Let*

$$X = (S, D, R, M, Q, E, H, C, P, N)$$

and

$$X' = (S', D', R', M', Q', E', H', C', P', N')$$

be accepted $\mathsf{Cauchy}^{\uparrow}$ *packets. Suppose the stream row, dyadic tolerance row, modulus row, modulus/readback compatibility row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, replay row, provenance row, and local naming row are transported componentwise by the displayed* hsame *data in* H *and* H'*, and suppose both packets are replayed through the same displayed* Cont *route. Then the public Cauchy readback route*

$$S, D, M \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

is the same BEDC route as

$$S', D', M' \longmapsto Q' \longmapsto R' \longmapsto E' \longmapsto H', C', P', N'.$$

In particular, componentwise transport of the modulus/readback surface cannot introduce quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, or ambient completeness.

Proof. Project both accepted packets through Definition 3413.1. The only visible coordinates are the stream, dyadic tolerance, regular-rational readback, modulus, compatibility, seal, transport, replay, provenance, and local naming rows.

Apply Definition 3413.3 to X and to X'. The two projected routes must read their displayed stream, dyadic, and modulus rows before their compatibility rows, and the shared Cont replay identifies these reads through the componentwise hsame data.

Next use Definition 3413.4. Access to R and R' is forced through the transported S, D, M, Q and S', D', M', Q' rows, so no other regular-rational readback path is available.

Finally apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal remains downstream of the transported $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the structural rows H, C, P, N and H', C', P', N' only transport, replay, source, and name that path. Since neither carrier contains quotient streams, selected limits, host equality, choice, trichotomy, or ambient completeness, the two displays name the same BEDC route. □

Theorem 3413.11 (Cauchy uniform-modulus Real seal)**.** *For an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet*

$$X = (S, D, R, M, Q, E, H, C, P, N),$$

any uniform-modulus consumer of the explicit modulus row M *that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must read* S, D, M, Q *before* R*, and* R *before* E*. The route contains no quotient stream equality, selected limit, countable choice, trichotomy, metric-completeness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet through Definition 3413.1. The explicit modulus row M is packaged with the finite stream window S, the dyadic tolerance row D, the compatibility row Q, the regular rational readback R, and the terminal seal-facing row E.

Apply Definition 3413.3. It forces a precision request to read S, D, M and then Q before any regular-rational row can be inspected.

Next apply Definition 3413.9. The $\mathsf{Real}^{\uparrow}$ seal row E is downstream of R, so a uniform-modulus consumer cannot reach the seal before the regular rational readback has been displayed.

Finally use Definition 3413.10. Componentwise transport and the displayed Cont replay preserve the same route for every accepted consumer, and the carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, metric-completeness data, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3413.12 (Cauchy root filter-completion anchor)**.** *Every Cauchy root consumer that first reaches the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *surface of chapter 1569 must already have read the Cauchy core route, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Project the root hub through its Cauchy core child. The core route exposes the stream-window, tolerance, readback, transport, replay, provenance, and naming rows before any completion-facing consumer can inspect the root packet.

Now read the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ chapter. Its completion handoff consumes the same finite windows and regular rational readback before the terminal Real seal is visible, so the root consumer cannot enter through a detached filter object.

The structural rows preserve the path. Componentwise hsame transport and displayed Cont replay keep the consumer inside the Cauchy root packet, while Pkg provenance and the local $\mathsf{NameCert}$ row record that no quotient filter, selected limit, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported. □

Theorem 3413.13 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 3413.14 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream

consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 3413.15 (Cauchy finite-window modulus pullback)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *modulus read pulls back to the same finite* $\mathsf{StreamName}^{\uparrow}$ *window of chapter 126, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any consumer can inspect a Cauchy value. In the notation of Definition 3413.1, the pullback route is*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No quotient stream equality, selected limit, host real equality, countable-choice coordinate, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row is available along this route.*

Proof. Project the accepted carrier through Definition 3413.1. The finite stream window S, dyadic tolerance row D, modulus row M, compatibility row Q, regular-rational readback R, and terminal Real seal E are the only rows available before the structural rows are read.

Apply Definition 3413.2. It identifies the local $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ surface with stream-window stability, dyadic tolerance exactness, regular-rational readback, displayed modulus admission, modulus/readback compatibility, Real seal ordering, transport, replay, provenance, and local naming.

Now use Definition 3413.3. A precision request first reads S, D, M, Q, then reaches R, and only then reaches E. The route is sharpened by Definition 3413.4, which keeps the $\mathsf{RegSeqRat}^{\uparrow}$ row behind the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, modulus, and compatibility rows.

Finally apply Definition 3413.9. The terminal Real seal is reached only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback has been displayed. Thus every accepted modulus read pulls back to the same four L10 sibling rows before a consumer sees a Cauchy value, and none of the excluded quotient, selected-limit, choice, order, compactness, or completeness coordinates can be inserted. □

Theorem 3413.16 (Regular-Cauchy modulus pullback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of the* $\mathsf{Cauchy}^{\uparrow}$ *finite-window modulus route obtains a dyadic tolerance schedule that is cofinal with the route displayed in Definition 3413.15. In the notation of Definition 3413.1, the consumer may read*

$$S, D, M, Q \longmapsto R \longmapsto E$$

only as a regular-rational handoff: the $\mathsf{Real}^{\uparrow}$ *seal is reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, not before it. The route exports no quotient stream equality, selected limit, host real equality, choice principle, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3413.15. It fixes the finite route from the StreamName window S, dyadic tolerance row D, modulus row M, and compatibility row Q into the regular-rational readback R, and only then into the terminal Real seal E.

Project the route at the $\mathsf{RegSeqRat}^{\uparrow}$ boundary. A RegSeqRat-facing consumer reads the dyadic tolerance row D through the same finite windows S and compatibility row Q, so its tolerance requests are bounded by a cofinal schedule already displayed by the Cauchy modulus route.

Transport the schedule through the regular-rational classifier row. The readback row R is the first row that presents the schedule as regular-rational data, and the Real seal E is reachable only after that readback has been displayed.

Thus the pullback is cofinal at the RegSeqRat surface and terminal at the Real seal. Since the carrier has no coordinate for quotient stream equality, selected limits, host equality, choice,

trichotomy, or Real completeness, none of those principles can be inserted into the consumer route. □

Theorem 3413.17 (Cauchy finite-window pullback uniqueness)**.** *Any* $\mathsf{Real}^{\uparrow}$*-facing Cauchy modulus consumer accepted by the* $\mathsf{Cauchy}^{\uparrow}$ *packet must factor through the same finite route*

$$S, D, M, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

displayed in Definition 3413.15. The consumer cannot replace this route by quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3413.1. The only rows visible to a modulus consumer are the finite StreamName window, dyadic tolerance ledger, modulus row, compatibility row, RegSeqRat readback, Real seal, and the structural rows.

Apply Definition 3413.2. It identifies the local NameCert surface with precisely those rows, so there is no second carrier coordinate from which an alternative Real-facing route could be read.

Use Definition 3413.3. A precision request reaches R only after S, D, M, Q have been displayed, and it reaches the terminal seal E only after the regular-rational readback has been displayed.

Finally apply Definition 3413.15 and Definition 3413.16. The first theorem fixes the finite pullback route to the L10 sibling rows, while the second shows that every RegSeqRat-facing consumer receives the same cofinal handoff. Hence any Real-facing Cauchy modulus consumer factors through the displayed route and gains no quotient, selected-limit, choice, host-equality, or completeness coordinate. □

Every completion-facing consumer of the $\mathsf{Cauchy}^{\uparrow}$ hub reads the finite Cauchy packet through chapter 301 and chapter 38 before any Real seal is exposed to completion rows. The route first displays the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the dyadic tolerance ledger, then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal, and only then lets Cauchy completion consumers inspect the same BHist, hsame, Cont, Pkg, and local NameCert rows. Thus the hub records a structural dependency route, not quotient stream equality, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 3413.18 (Cauchy RegSeqRat Real forward dependency)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *source-window read consumed by a Real-facing downstream chapter must pass through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is exposed. The route reads only displayed finite windows, regular rational tail data, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier route in Definition 3413.1. Its source-window reads are displayed before the regular rational readback and terminal real seal rows are available.

Use the regular readback and real-seal routes already named by the Cauchy child files. They order downstream consumption through $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$, with the structural rows transporting and naming the same finite windows.

Since the hub exposes only child-file surfaces and no independent carrier coordinate, a Real-facing consumer can cite $\mathsf{Cauchy}^{\uparrow}$ only through that forward dependency route. The read therefore reaches chapter 301 and chapter 38 before any Real-facing seal is exposed. □

Theorem 3413.19 (Cauchy located-uniform filter forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *route whose explicit modulus row is consumed by a located or uniform Cauchy filter completion must pass through the displayed carrier route*

$$S, D, M, Q, R, E, H, C, P, N$$

of Definition 3413.1, then through chapter 1661 or chapter 1694, before any $\mathsf{Real}^{\uparrow}$ *seal-facing completion row is exported. The forward route has no quotient stream equality, selected limit, countable choice, metric-completeness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host equality.*

Proof. Start with the explicit modulus readback of the Cauchy carrier. By Definition 3413.3, a precision request reads the finite StreamName row S, the DyadicRatCore row D, the explicit modulus row M, and the compatibility row Q before it can inspect the regular rational row R.

Apply Definition 3413.9. The seal-facing row E is downstream of R, so a Real-facing completion consumer cannot bypass the regular rational readback.

Now use Definition 3413.11. Any uniform-modulus consumer of M that reaches a $\mathsf{Real}^{\uparrow}$ seal preserves the same ordering S, D, M, Q before R and R before E.

The remaining completion-neighbour branch is supplied only by the sibling chapters chapter 1661 and chapter 1694. These chapters consume the already displayed Cauchy route as located or uniform filter evidence; neither adds quotient stream equality, selected limits, choice, metric-completeness data, ambient completeness, or host equality to the core Cauchy carrier. □

Theorem 3413.20 (Cauchy route to CauchyCriterion)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *hub's finite route is consumed by chapter 392 through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The route exports no detached completion carrier, quotient sequence equality, selected modulus outside the displayed rows, countable-choice schedule, or host completeness principle.*

Proof. Read the hub through its existing finite siblings. The $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows provide the only Cauchy data visible to downstream criteria.

The structural rows hsame, Cont, Pkg, and NameCert transport, replay, source, and name that same finite route. Therefore the Cauchy criterion consumer receives exactly the displayed BEDC rows and no external completion or choice object. □

Theorem 3413.21 (Cauchy hub Bishop completion forward route)**.** *The regular-source material routed by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by chapter 1639 through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The forward route exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness principle.*

Proof. Read the Cauchy hub through its regular-source children. Those children expose dyadic tolerance arithmetic, finite stream windows, regular rational readback, and the terminal real seal as $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

The Bishop regular Cauchy completion chapter consumes exactly that L10-facing route. Its completion read is mediated by the same finite BHist carrier discipline, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Since the hub route contains no detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient real completeness coordinate, the forward edge to the Bishop completion packet is precisely the displayed finite dependency route. □

Chapter 3414

A Concrete Naming Certificate for $\mathsf{FiniteSimplicialChain}^{\uparrow}$

Following Eilenberg–Steenrod and the standard finite presentation of simplicial homology, the $\mathsf{FiniteSimplicialChain}^{\uparrow}$ packet records the BEDC route from a finite $\mathsf{SimplicialComplex}^{\uparrow}$ carrier to the chain groups and boundary rows consumed by $\mathsf{Homology}^{\uparrow}$. It sits over the simplex inventory of chapter 238, the finite enumeration surface of chapter 112, and the additive carrier rows of chapter 42; it supplies $\mathsf{Homology}^{\uparrow}$ and $\mathsf{PersistentHom}^{\uparrow}$ with a displayed finite chain-complex source rather than an implicit classical complex.

Definition 3414.1 (Finite simplicial chain carrier)**.** *A* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *carrier over a finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *packet is a finite* BHist *tuple*

$$C = (K, F, A, G, B, Z, H, T, P, N)$$

with the following displayed rows:

1. *K is the source* $\mathsf{SimplicialComplex}^{\uparrow}$ *row, including the finite simplex inventory, face relation, and dimension grading.*
2. *F is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration row for oriented simplex generators in each displayed dimension.*
3. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *row for finite integer combinations of oriented simplex generators, with addition and inverse read componentwise.*
4. *G is the grading row assigning each chain generator to its simplex dimension.*
5. *B is the boundary ledger sending an oriented n-simplex to the alternating finite face sum in dimension $n-1$.*
6. *Z is the square-zero cancellation ledger, pairing each codimension-two face occurrence with the opposite occurrence obtained by deleting the two vertices in the other order.*
7. *H records componentwise* hsame *transport for simplex inventories, generator enumerations, additive rows, grading rows, boundary rows, cancellation rows, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from K, F, A, G, B, Z, H to the chain-complex source read by* $\mathsf{Homology}^{\uparrow}$*.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSimplicialChain}^{\uparrow}$*,* $\mathsf{SimplicialComplex}^{\uparrow}$*,* $\mathsf{FinSet}^{\uparrow}$*,* $\mathsf{AbGroup}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an infinite chain module, a host quotient group, a selected orientation convention outside F, an ambient topological realization, or a homology computation not routed through B and Z.

Lean4 ✓ `BEDC.Derived.FiniteSimplicialChainUp.TasteGate.FiniteSimplicialChainTasteGate single carrier alignment`

Theorem 3414.2 (Finite simplicial chain boundary square-zero)**.** *For every accepted* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *carrier $C = (K, F, A, G, B, Z, H, T, P, N)$, the displayed boundary row B composes with itself to the zero row of the* $\mathsf{AbGroup}^{\uparrow}$ *chain carrier. Equivalently, every* $\mathsf{Homology}^{\uparrow}$ *consumer receives a chain-complex source whose boundary-square obstruction is discharged by the finite cancellation ledger Z before cycles and boundaries are compared.*

Proof. Project the carrier of Definition 3414.1. The simplex source K and finite generator enumeration F make every boundary expansion a finite face list, and the additive row A is the only group operation available for collecting those face terms.

Apply the boundary ledger B once to an oriented simplex. The result is the alternating sum of its displayed codimension-one faces. Applying B again expands each face into codimension-two faces, still inside the finite enumeration supplied by F.

The cancellation ledger Z pairs the term obtained by deleting vertex i and then vertex j with the term obtained by deleting j and then i. The two terms have the same codimension-two face row under hsame and opposite signs in the $\mathsf{AbGroup}^{\uparrow}$ carrier, so they add to the zero row.

Since all codimension-two occurrences are exhausted by these pairs, the second boundary has no unpaired summand. The continuation, provenance, and local naming rows T, P, N repack the same finite cancellation into the $\mathsf{Homology}^{\uparrow}$ source surface, so no host chain module or external quotient argument is used. □

Falsifiable prediction: A $\mathsf{Homology}^{\uparrow}$ or $\mathsf{PersistentHom}^{\uparrow}$ consumer using $\mathsf{FiniteSimplicialChain}^{\uparrow}$ must display the source $\mathsf{SimplicialComplex}^{\uparrow}$ row, finite generator enumeration, additive chain carrier, boundary ledger, square-zero cancellation ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: simplicialcomplex, finset, abgroup, homology: $\mathsf{FiniteSimplicialChain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries simplex inventory, oriented generator enumeration, additive chain rows, boundary rows, square-zero cancellation, Homology-facing replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSimplicialChain}^{\uparrow}$ packet is generated from a finite SimplicialComplex row, FinSet generator enumeration, AbGroup additive chain rows, grading, a finite alternating boundary ledger, codimension-two cancellation pairs, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$, seed)

Scope closed: Definition 3414.1 and Definition 3414.2 seed the finite simplicial chain carrier and its square-zero boundary route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSimplicialChain}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteSimplicialChainUp.TasteGate.FiniteSimplicialChainTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close singular chains, infinite chain complexes, host quotient homology, selected geometric realization, or bridge checking.

Upgrade path: Obligation closure requires finite generator enumeration, additive chain stability, boundary exactness, square-zero cancellation, Homology handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.FiniteSimplicialChainU

Chapter 3415

A Concrete Naming Certificate for $\mathsf{RealModulusCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealModulusCriterion}^{\uparrow}$ packet records the finite criterion by which an explicit Cauchy modulus makes a regular rational stream ready for a $\mathsf{Real}^{\uparrow}$ convergence read. It sits between the convergence schedule of chapter 1438, the sequence-limit consumer of chapter 1413, the regular rational names of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38.

Definition 3415.1 (Real modulus criterion carrier)**.** *A* $\mathsf{RealModulusCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, Q, W, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row naming the explicit tail schedule.*
2. S *is the sequence-source row inspected by the modulus.*
3. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback exposed by the scheduled tails.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used to read each tail comparison.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the modulus requests.*
6. L *is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer row reached only after* M, S, Q, W, D *have been displayed.*
7. E *is the* $\mathsf{Real}^{\uparrow}$ *convergence-readiness seal exported to real-name consumers.*
8. H *records componentwise* hsame *transport for the analytic rows and the structural rows.*
9. C *records displayed* Cont *replay from the tolerance request through the modulus, windows, regular readback, sequence-limit consumer, and real seal.*
10. P *is the* Pkg *provenance over* $\mathsf{RealModulusCriterion}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. N *is the local* $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completion, selected representatives, countable choice, trichotomy, host sequence convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealModulusCriterionUp.RealModulusCriterionTasteGate single carrier alignment`

Theorem 3415.2 (Real modulus criterion NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$ *surface for Definition 3415.1 consists exactly of carrier exposure for R, explicit modulus exposure through M, sequence-source exposure through S, regular rational readback through Q, finite window exposure through W, dyadic tolerance exposure through D, sequence-limit handoff through L, real convergence-readiness through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $R = (M, S, Q, W, D, L, E, H, C, P, N)$. The analytic route is finite: a tolerance request reads the modulus row M, the source row S, the regular rational row Q, the window row W, and the dyadic row D before it reaches L and E.

The rows H, C, P, N preserve and name that same route. Componentwise hsame transport keeps each analytic coordinate in its displayed sibling surface, while Cont replay records the order in which the consumer reads the modulus, windows, regular name, sequence-limit row, and real seal.

No excluded object is a row of the carrier. Therefore the listed rows exhaust the seed NameCert obligations for the explicit real-modulus criterion. □

Theorem 3415.3 (Real modulus criterion completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealModulusCriterion}^{\uparrow}$ *packet must pass through*

$$\mathsf{CauchyConvergenceModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RealSequenceLimit}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in the displayed carrier order before treating the packet as convergence-ready for real-name use. The handoff supplies a finite criterion for explicit Cauchy-modulus convergence; it does not supply quotient completion, a selected limiting representative, countable choice, trichotomy, or an ambient completeness theorem.

Proof. Start with the carrier definition. The modulus row M and dyadic ledger D identify the tolerance request, while W and Q expose the finite tail windows and regular rational readback needed by the request.

Next read the sequence-limit and real-seal rows. The row L is reached only after the modulus, window, dyadic, and regular readback rows have been displayed, and E is the resulting $\mathsf{Real}^{\uparrow}$ readiness boundary.

Finally apply the structural rows. Transport, replay, provenance, and local naming can only preserve the displayed route, so a completion-facing consumer cannot bypass the criterion through quotient completion, representative selection, countable choice, trichotomy, host convergence, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusCriterion}^{\uparrow}$ packet is generated from an explicit CauchyConvergenceModulus row, a sequence source, RegSeqRat readback, finite StreamName windows, DyadicRatCore tolerance rows, a RealSequenceLimit consumer row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3415.1, Definition 3415.2, and Definition 3415.3 declare the finite explicit-modulus criterion connecting $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealModulusCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, trichotomy, host sequence convergence, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, explicit modulus exposure, sequence-limit handoff, real-seal nonescape, and completion-facing criterion readiness.

Chapter 3416

A Concrete Naming Certificate for $\mathsf{CauchyFilterCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyFilterCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed filter base becomes acceptable Cauchy-filter data for the L10 completion route. It sits after the finite $\mathsf{FilterBase}^{\uparrow}$ interface of chapter 1601, the Cauchy-filter surface of chapter 333, and the completion-facing routes of chapter 1506 and chapter 1473; it never replaces them by quotient filters, choice-selected limits, or host completeness.

Definition 3416.1 (Cauchy filter criterion carrier)**.** *A* $\mathsf{CauchyFilterCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, M, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row naming the finite base packets and directed-refinement witnesses that may be inspected by a completion consumer.*
2. *K is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row recording that the displayed base is Cauchy only through finite entourage or ball requests.*
3. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row reached only after B, K, W, R, D, E have been displayed.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row exposing the same Cauchy-filter route to entourage-facing completion consumers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used when the criterion is read sequentially.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the sequential windows and tolerance requests.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds every displayed Cauchy request.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal read by real-name consumers after the filter, window, regular-rational, and dyadic rows have been displayed.*
9. *H records componentwise* hsame *transport for the filter-base, Cauchy-filter, completion, sequential, tolerance, real-seal, and structural rows.*
10. *C records displayed* Cont *replay from a filter-base request through Cauchy-filter evidence, sequential readback, tolerance accounting, completion handoff, and real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCriterion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, M, U, W, R, D, E, H, C, P, N$. There is no coordinate for quotient filter equality, selected representatives, countable choice, host-side limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterCriterionUp.CauchyFilterCriterionTasteGate single carrier alignment`

Theorem 3416.2 (Cauchy filter criterion NameCert obligations)**.** *For the carrier of Definition 3416.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$ *surface consists exactly of carrier exposure for F, finite filter-base exposure through B, Cauchy-filter exposure through K, metric-completion handoff through M, uniform-completion handoff through U, sequential window exposure through W, regular rational readback through R, dyadic tolerance exposure through D, real-name sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $F = (B, K, M, U, W, R, D, E, H, C, P, N)$. The base row B supplies only the finite directed interface of chapter 1601, and the Cauchy row K reads that interface before any completion-facing coordinate is visible.

The sequential rows then bind the criterion to the real-name route. The window row W, regular readback row R, and dyadic tolerance row D are displayed before the real seal E or either completion handoff can be consumed.

Finally the structural rows H, C, P, N transport, replay, source, and name exactly these coordinates. Since quotient filters, selected representatives, countable choice, host limits, and ambient completeness are not rows of the carrier, no further NameCert obligation remains. □

Theorem 3416.3 (Cauchy filter criterion real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterCriterion}^{\uparrow}$ *carrier must pass through*

$$\mathsf{FilterBase}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

in the displayed carrier order before reading the completion handoff rows M or U. The handoff supplies a finite Cauchy-filter criterion for the completion route; it does not supply quotient filters, choice-selected limits, selected representatives, countable choice, host completeness, or an ambient real-limit operator.

Proof. Start with the finite base route. The consumer first reads B and can narrow requests only through the displayed directed-refinement surface supplied by Definition 1601.3.

Next read the Cauchy and sequential rows. The row K witnesses the Cauchy-filter condition over the same base, while W, R, D translate inspected requests into finite windows, regular rational readback, and dyadic tolerance accounting.

The terminal seal and completion handoffs are downstream of those rows. Thus E, M, U can only expose the real-name and completion-facing boundaries after the displayed filter-base criterion has been replayed by C and transported by H.

No excluded principle is carried by P or N. The packet therefore feeds the L10 real and completion route without quotient filters, selected limits, countable choice, host completeness, or a hidden ambient limit object. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCriterion}^{\uparrow}$ packet is generated from a FilterBaseUp row, a CauchyFilterUp row, MetricCompletionUp and UniformCompletionUp handoff rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3416.1, Definition 3416.2, and Definition 3416.3 seed the finite criterion by which a displayed FilterBase and CauchyFilter route may feed Real, MetricCompletion, and UniformCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyFilterCriterionUp.CauchyFilterCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient filters, choice-selected limits, selected representatives, countable choice, host completeness, bridge checking, or an ambient real-limit operator.

Upgrade path: Obligation closure requires Cauchy-filter criterion exposure, completion handoff nonescape, real-seal readiness, and ledger-refusal proofs for BEDC.Derived.CauchyFilterCriterionUp.

Chapter 3417

A Concrete Naming Certificate for $\mathsf{UniformConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformConvergence}^{\uparrow}$ packet records uniform tail control before any $\mathsf{UniformLimit}^{\uparrow}$ consumer reads a $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the pointwise limiting surface of chapter 2314, the uniform-limit route of chapter 463, the continuous-map graph rows of chapter 130, and the terminal real-name seal of chapter 38. The chapter names the generic finite tail ledger used by later completion and limit consumers; it is not a selected limit, a supremum norm construction, countable choice, quotient equality of functions, host completeness, or open-cover compactness.

Definition 3417.1 (Uniform convergence carrier)**.** *A* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the displayed function-family or sequence source. Each source entry is read only through the finite windows named by the packet.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window ledger. It records the parameter windows, index windows, and tolerance-request windows inspected by the consumer.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed finite comparisons attached to F and W.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback has displayed the scheduled comparisons.*
5. *T is the uniform tail schedule. For each requested tolerance row it records one threshold row shared by every displayed observation window in W.*
6. *H records componentwise* hsame *transport for the source, window ledger, readback, real seal, tail schedule, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from any limit-facing consumer request through F, W, T, Q, R in that order.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these nine rows. Its classifier has no coordinate for a selected limiting function, a host supremum norm, quotient equality of functions, countable choice, ambient completeness, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.UniformConvergenceUp.TasteGate.UniformConvergenceTasteGate single carrier alignment`

Theorem 3417.2 (Uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier*

$$U = (F, W, Q, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source row, finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seals, uniform tail schedule, transports, continuations, provenance, and local naming;*
2. *uniform tail coverage, requiring every requested tolerance row to read the single threshold recorded in T before any limiting read is exposed;*
3. *real-seal non-escape, requiring every* $\mathsf{Real}^{\uparrow}$ *seal in R to remain downstream of the readback row Q and the schedule row T;*
4. $\mathsf{UniformLimit}^{\uparrow}$ *handoff, requiring each endpoint-facing consumer to factor through $F, W, Q, R, T, H, C, P, N$ before forming a limiting read;*
5. *transport and replay stability, requiring H and C to preserve the displayed source, window, readback, seal, and tail rows componentwise;*
6. *excluded rows, ruling out selected limits, host supremum norms, quotient equality of functions, countable choice, ambient completeness, open-cover compactness, and any hidden pointwise-only threshold.*

No uniform-convergence obligation is stored outside these rows.

Proof. Project the accepted packet $U = (F, W, Q, R, T, H, C, P, N)$. The carrier exposes exactly the displayed source, finite observation windows, regular rational readbacks, real seals, uniform tail schedule, and structural rows.

Uniformity is supplied by T. A tolerance request first selects a single threshold row in that ledger, and the same row must serve every displayed observation window in W before any readback is admitted.

The real seal is then downstream of the regular rational route. The row Q is the only readback channel attached to the scheduled comparisons, and the seal row R becomes visible only after Q has displayed those comparisons through the tail schedule.

The endpoint handoff follows by the displayed route order. A $\mathsf{UniformLimit}^{\uparrow}$ consumer receives F, W, T, Q, R and then the structural rows H, C, P, N; there is no carrier coordinate from which it can read a selected limit, host supremum norm, quotient equality, choice row, ambient completeness theorem, open-cover compactness argument, or pointwise-only threshold. □

Theorem 3417.3 (Uniform convergence UniformLimit handoff)**.** *Every* $\mathsf{UniformLimit}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformConvergence}^{\uparrow}$ *carrier reads*

$$F, W, Q, R, T, H, C, P, N$$

before forming the limiting read. Thus the limiting route is forced through the displayed source, finite observation windows, regular rational readbacks, real seals, uniform tail schedule, componentwise transport, continuation replay, provenance, and local naming. It receives no selected limit, host supremum norm, quotient function equality, countable-choice schedule, ambient completeness theorem, open-cover compactness principle, or hidden pointwise-only threshold.

Proof. Start with a limit-facing consumer request. By Definition 3417.1, the request first reads the source row F through the finite observation windows W.

The uniform tail schedule T then supplies the common threshold for the requested tolerance. The readback row Q is attached to that scheduled comparison, and the real seal row R is reachable only after the readback has displayed it.

Finally H, C, P, N transport, replay, source, and name the same route. Since every coordinate available to the consumer occurs in the displayed packet, the limiting read cannot be formed from a selected limit, host supremum norm, quotient function equality, countable-choice schedule, ambient completeness theorem, open-cover compactness principle, or hidden pointwise-only threshold. □

Falsifiable prediction: An accepted $\mathsf{UniformConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{UniformLimit}^{\uparrow}$ endpoint or $\mathsf{Real}^{\uparrow}$ seal without first reading $F, W, Q, R, T, H, C, P, N$ refutes the carrier surface.

Independence witness: streamname, regseqrat, pointwise-limit, uniformlimit, continuousmap, real: none is bijective to $\mathsf{UniformConvergence}^{\uparrow}$ because this packet jointly carries a displayed source, finite observation windows, regular readbacks, real seals, one uniform tail schedule, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3417.1, Definition 3417.2, and Definition 3417.3 seed the finite uniform tail ledger over displayed sources, StreamName windows, RegSeqRat readbacks, Real seals, and UniformLimit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformConvergenceUp.TasteGate.UniformConvergenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim selected limits, host supremum norms, quotient equality of functions, countable choice, ambient completeness, open-cover compactness, or bridge checking.

Upgrade path: Obligation closure requires uniform tail coverage, real-seal non-escape, UniformLimit handoff, transport and replay stability, and excluded-row coverage for BEDC.Derived.UniformConvergenceUp.

Constructive story: A $\mathsf{UniformConvergence}^{\uparrow}$ packet is generated from a displayed function-family or sequence source, finite StreamName observation windows, RegSeqRat readbacks, RealUp seals, one uniform tail schedule, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Chapter 3418

A Concrete Naming Certificate for $\mathsf{BishopApproximationSequence}^{\uparrow}$

Following Bishop and Bridges-style constructive analysis, the $\mathsf{BishopApproximationSequence}^{\uparrow}$ packet records a finite naming surface for approximation sequences whose visible observations are dyadic schedules, finite stream windows, and regular-rational readbacks. It sits between the dyadic arithmetic of chapter 291, the finite stream surface of chapter 126, the regular-rational route of chapter 301, the terminal real-name seal of chapter 38, and the convergence-modulus discipline of chapter 1438. The packet gives Real and Completion consumers a named approximation-sequence interface without adding quotient sequence equality, countable choice, trichotomy, a selected host limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3418.1 (BishopApproximationSequence carrier)**.** *A* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, W, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bracket row, storing the dyadic precision requests and the finite arithmetic witnesses used by the approximation schedule;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, storing the finite sequence windows whose entries are inspected by the consumer;*
3. *W is the window-synchronization row that binds the dyadic tolerances in D to the stream observations in S;*
4. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row, storing the finite modulus witnesses used to compare later windows with earlier requested tolerances;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that repacks the accepted windows as regular rational readback data;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, W, M, R have been displayed;*
7. *H records the componentwise* hsame *transports for the dyadic row, stream row, window row, modulus row, regular handoff, real seal, continuation routes, provenance, and local naming row;*
8. *C records the displayed* Cont *routes that replay tolerance reads, stream-window reads, modulus reads, regular-rational handoff, and real-seal readback;*

9. *P is the* Pkg *provenance over* $\mathsf{BishopApproximationSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$ *row.*

The accepted carrier stores no quotient sequence equality, no selected infinite subsequence, no host real equality test, no countable-choice functional, no trichotomy row, and no ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.BishopApproximationSequenceUp.BishopApproximationSequenceTasteGate single carrier alignment`

Theorem 3418.2 (BishopApproximationSequence NameCert obligations)**.** *For an accepted packet $A = (D, S, W, M, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$ *obligations are exactly the following finite rows:*

1. *carrier admission: the accepted object is the displayed packet with dyadic tolerances, finite stream windows, synchronized window data, convergence modulus, regular-rational handoff, real seal, transports, continuations, provenance, and local naming;*

2. *dyadic-window coherence: every precision request in D is attached to a displayed window in S through the synchronization row W;*

3. *modulus coverage: every accepted tolerance request reads a finite witness in M before any regular-rational or real-name observation is exported;*

4. *regular handoff stability: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R reads only the synchronized windows and modulus witnesses carried by D, S, W, M;*

5. *classifier transport: the componentwise transports in H preserve the dyadic row, stream row, synchronization row, modulus row, regular handoff, real seal, continuation replay, provenance, and local naming row;*

6. *ledger exhaustion: every Real or Completion consumer must pass through one of $D, S, W, M, R, E, H, C, P, N$ before reading any approximation-sequence observation.*

No obligation may call on quotient sequence equality, a selected host limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier row by row. Carrier admission is immediate from the displayed BHist packet, and dyadic-window coherence reads only the tolerance row D, stream row S, and synchronization row W.

For modulus coverage, inspect a tolerance request in D. The row M supplies the finite convergence witness attached to that request, and the route into R is available only after the synchronized window has been displayed.

Classifier transport and ledger exhaustion are structural. The transport row H moves only the displayed coordinates, while C replays only the displayed tolerance, window, modulus, regular handoff, and seal routes. Since the carrier contains no quotient equality, selected limit, choice functional, trichotomy row, host equality test, or completeness coordinate, none of those principles can enter the local obligations. □

Theorem 3418.3 (BishopApproximationSequence RegSeqRat handoff)**.** *Let $A = (D, S, W, M, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *packet. Every regular-rational read exported by A factors through a displayed dyadic tolerance in D, a finite stream window in S, the synchronization*

row W, and a convergence-modulus witness in M before the $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row R is available. The route is stable under the transports in H and the replay routes in C.*

Proof. Begin with a regular-rational consumer of the packet. The carrier makes R the only regular-rational handoff coordinate, and R is downstream of the dyadic, stream-window, synchronization, and modulus rows.

Apply Definition 3418.2. Its dyadic-window coherence and modulus coverage rows force every tolerance request to pass through D, S, W, M before a regular read is exposed.

The transports in H preserve each displayed coordinate, and the continuations in C replay only the same route. Thus the regular-rational read cannot detach from the finite approximation schedule or use quotient sequence equality, countable choice, trichotomy, host equality, selected limits, or ambient completeness. □

Theorem 3418.4 (BishopApproximationSequence Real seal nonescape)**.** *Let $A = (D, S, W, M, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopApproximationSequence}^{\uparrow}$ *packet. No* $\mathsf{Real}^{\uparrow}$*-facing consumer can bypass the dyadic tolerance row D, finite stream row S, synchronization row W, convergence-modulus row M, regular handoff R, terminal seal E, transports H, replays C, provenance P, and local name N to obtain quotient sequence equality, a selected host limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier. The only terminal real-name coordinate is E, and it is displayed after the dyadic tolerance row, finite stream row, synchronization row, modulus row, and regular-rational handoff.

Use Definition 3418.3. Every regular-rational approximation read has already passed through D, S, W, M before reaching R, so the route into E is a finite route through the named rows.

The rows H and C can transport and replay only displayed coordinates, while P and N record the same package and local name. Because the carrier contains none of the excluded objects, the Real-facing seal cannot export quotient equality, selected limits, choice, trichotomy, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopApproximationSequence}^{\uparrow}$ packet is generated from dyadic tolerances, finite stream windows, a synchronization row, a convergence-modulus row, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3418.1, Definition 3418.2, Definition 3418.3, and Definition 3418.4 present the finite approximation-sequence certificate surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopApproximationSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient sequence equality, selected host limits, countable choice, trichotomy, host real equality, arbitrary least-upper-bound completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, dyadic-window coherence, modulus coverage, regular handoff stability, classifier transport, ledger exhaustion, and real-seal nonescape for the displayed packet.

Chapter 3419

A Concrete Naming Certificate for $\mathsf{LaxMilgram}^{\uparrow}$

Following Lax and Milgram and the variational treatment in Evans, the $\mathsf{LaxMilgram}^{\uparrow}$ packet records the finite BEDC surface for coercive variational solvability. It is read as a neighbour of the Hilbert and Sobolev certificates of chapter 91 and chapter 440, with its pairing row anchored in the bilinear-form chapter chapter 146 and its bounded action surface anchored in chapter 1644. The packet names only the displayed variational handoff; it does not add a choice-selected solution operator or an ambient PDE existence theorem.

Definition 3419.1 (Lax-Milgram carrier)**.** *A* $\mathsf{LaxMilgram}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, B, F, C, S, T, P, N)$$

with the following displayed rows:

1. *H is the Hilbert/Sobolev domain row, read through the* $\mathsf{Hilbert}^{\uparrow}$ *surface and, when a weak-form PDE consumer is present, through the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative ledger;*
2. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *row naming the bilinear pairing together with its boundedness and coercivity evidence as finite classifier rows;*
3. *F is the bounded functional row, read through the same bounded-linear-operator surface used by* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *consumers;*
4. *C records the coercivity and boundedness constants as displayed ledger entries rather than as real-number selectors outside the packet;*
5. *S is the solution handoff row: it exposes the single variational response row certified by the equation $B(S, v) = F(v)$ for displayed test rows v in H;*
6. *T stores the* hsame *transport and* Cont *replay rows for the domain, bilinear form, bounded functional, constants, and solution handoff;*
7. *P is the* Pkg *provenance over* $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a choice-selected global inverse, a quotient representative, a completed PDE solution space beyond the cited Sobolev row, or a host theorem asserting existence outside the displayed variational packet.

Lean4 ✓ `BEDC.Derived.LaxMilgramUp.LaxMilgramTasteGate single carrier alignment`

Theorem 3419.2 (Lax-Milgram NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$ *surface for the carrier of Definition 3419.1 consists of exactly five finite obligations:*

1. *domain admission, requiring H to be read through the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row and any weak-form consumer through the* $\mathsf{Sobolev}^{\uparrow}$ *ledger;*
2. *bilinear-form admission, requiring B to expose only the* $\mathsf{BilinForm}^{\uparrow}$ *pairing row together with boundedness and coercivity ledger entries;*
3. *functional admission, requiring F to be read through the bounded functional surface supplied by* $\mathsf{BoundedLinearOperator}^{\uparrow}$*;*
4. *variational handoff exactness, requiring every solution-facing read to pass through S and the displayed equation row $B(S, v) = F(v)$ for admitted test rows v;*
5. *transport and replay stability, requiring T, P, N to preserve the same domain, bilinear, functional, constant, and handoff rows under componentwise* hsame*, displayed* Cont*, provenance, and local naming.*

Proof. Project the carrier rows. The domain, bilinear form, functional, constant ledger, handoff, transport, provenance, and local naming evidence are the full visible inventory of K.

Use the cited sibling certificates for each analytic dependency. The Hilbert and Sobolev rows supply the domain-facing reads, the bilinear-form row supplies the pairing and its boundedness/coercivity classifier, and the bounded-linear-operator surface supplies the functional row.

Finally inspect a solution-facing consumer. It can reach only the handoff row S and the displayed variational equation against admitted test rows. The transport, replay, provenance, and naming rows preserve that finite route and add no external selector. □

Theorem 3419.3 (Lax-Milgram variational handoff)**.** *Every PDE, Sobolev, Hilbert, or variational consumer of an accepted* $\mathsf{LaxMilgram}^{\uparrow}$ *packet receives only the displayed route*

$$H \longmapsto B \longmapsto F \longmapsto C \longmapsto S \longmapsto T, P, N.$$

The consumer may use the certified variational response row and its equation against admitted test rows, but it receives no choice-selected solution operator, no quotient representative, no host inverse map, no ambient compactness principle, and no PDE solution space beyond the finite Sobolev or Hilbert rows already cited.

Proof. Begin with the NameCert obligations. They force any consumer to read the domain, bilinear form, functional, constant ledger, and handoff rows in the order displayed by the carrier.

The Sobolev and Hilbert dependencies identify the domain surface, while the bilinear-form and bounded-linear-operator dependencies identify the pairing and functional surfaces. These reads are certificate rows, not host analytic constructions.

It remains to inspect the handoff. The row S is available only as the variational response named by $B(S, v) = F(v)$ for displayed tests v. Since no other solution coordinate is present in the carrier, the listed selector, quotient, inverse, compactness, and ambient PDE objects cannot be exported. □

Theorem 3419.4 (Lax-Milgram energy-minimizer handoff)**.** *An accepted* $\mathsf{LaxMilgram}^{\uparrow}$ *packet exposes its variational solution row as an energy-minimizer handoff only through the Hilbert/Sobolev*

domain row of chapter 91 and chapter 440, the $\mathsf{BilinForm}^{\uparrow}$ *coercive pairing row of chapter 146, and the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *functional row of chapter 1644. The handoff is the finite route*

$$H \longmapsto B \longmapsto F \longmapsto C \longmapsto S,$$

with T, P, N *preserving transport, replay, provenance, and local naming. It exports no selected inverse operator, no quotient representative of a solution class, and no ambient PDE existence theorem.*

Proof. Project the carrier of Definition 3419.1. The only domain coordinate is H, so an energy read starts from the Hilbert row and, for weak-form consumers, the Sobolev ledger named by the sibling chapters.

The energy pairing is then forced through B and C. The bilinear-form sibling supplies the coercive pairing row, while the constant ledger records the displayed boundedness and coercivity data. No inverse operator is stored in these rows.

The forcing term is the row F supplied through the bounded-linear-operator surface. The variational response row S can therefore be read as an energy-minimizer handoff only after the domain, pairing, constant, and functional rows have been displayed.

Finally T, P, N transport, source, and name the same finite route. Since the carrier has no coordinate for a selected inverse operator, quotient representative, or ambient PDE existence theorem, none of those objects can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$.

Constructive story: A $\mathsf{LaxMilgram}^{\uparrow}$ packet is generated from a Hilbert or Sobolev domain row, a bilinear-form row with boundedness and coercivity ledgers, a bounded functional row, a solution handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$, seed)

Scope closed: Definition 3419.1, Definition 3419.2, and Definition 3419.3 seed the finite variational certificate surface consumed by Sobolev, Hilbert, PDE, and bounded-operator routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LaxMilgram}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a choice-selected solution operator, quotient representative, host inverse map, ambient compactness principle, or full PDE existence theorem.

Upgrade path: Obligation closure requires checked carrier admission, bilinear boundedness and coercivity rows, bounded functional admission, solution handoff exactness, transport, replay, provenance, and local naming for the displayed packet.

Chapter 3420

A Concrete Naming Certificate for $\mathsf{BanachAlaoglu}^{\uparrow}$

Following Banach and Alaoglu, the $\mathsf{BanachAlaoglu}^{\uparrow}$ packet records the finite weak-star dual-ball window that functional-analysis consumers can inspect before any unrestricted compactness theorem is available. It sits over the Banach source row of chapter 90, the bounded-linear-functional surface supplied by chapter 1644, the scalar norm readback of chapter 1500, the terminal real seal of chapter 38, and the finite product compactness handoff of **??**. The chapter names only finite windows in the weak-star dual ball; it does not claim unrestricted Tychonoff compactness, ultrafilter arguments, host weak-star compactness, quotient function spaces, or selected cluster points.

The weak-star window also reads the locally convex seminorm-neighborhood discipline of chapter 3386 before any dual-ball compactness route is exposed.

Definition 3420.1 (Banach-Alaoglu finite weak-star dual-ball carrier)**.** *A* $\mathsf{BanachAlaoglu}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, L, W, S, R, F, T, C, P, N)$$

with the following displayed rows:

1. *B is the Banach source row, read through the displayed* $\mathsf{Banach}^{\uparrow}$ *carrier and its norm, metric, regular-Cauchy, and completion ledgers.*
2. *L is the bounded-linear functional family row, whose members are read as* $\mathsf{BoundedLinearOperator}^{\uparrow}(B, \mathsf{Real}^{\uparrow})$ *coordinates with displayed operator bounds.*
3. *W is the finite weak-star seminorm-window row, listing the finitely many source probes and scalar tolerances allowed in the present observation.*
4. *S is the dual-ball bound row, recording the shared scalar bound on the displayed functionals through* $\mathsf{NormedRealLine}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readbacks.*
5. *R is the Real and normed-real readback row for all scalar values, scalar bounds, and seminorm tolerances exposed by the window.*
6. *F is the finite-product handoff row, sending the displayed scalar coordinate window to the finite* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$ *compactness surface.*
7. *T records componentwise* hsame *transport for the Banach source, functional family, weak-star window, scalar readbacks, finite-product handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a weak-star window request through $B, L, W, S, R, F, T, P, N$.*
9. *P is the* Pkg *provenance over* BanachAlaoglu$^{\uparrow}$, Banach$^{\uparrow}$, BoundedLinearOperator$^{\uparrow}$, NormedRealLine$^{\uparrow}$, Real$^{\uparrow}$, TychonoffFiniteProduct$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BanachAlaoglu}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted Tychonoff compactness, ultrafilter or choice arguments, host weak-star compactness, quotient function spaces, selected cluster points, ambient dual-space compactness, Lean verification, or standard bridge data.

Lean4 ✓ `BEDC.Derived.BanachAlaogluUp.BanachAlaogluTasteGate single carrier alignment`

Theorem 3420.2 (Banach-Alaoglu NameCert obligations)**.** *For the carrier of Definition 3420.1, the local* NameCert$_{\text{BanachAlaoglu}^{\uparrow}}$ *surface consists of exactly six finite obligations:*

1. *Banach source admission, requiring B to be read through the displayed source row of chapter 90;*
2. *bounded-linear-functional admission, requiring each coordinate of L to pass through the bounded-linear-operator surface of chapter 1644;*
3. *finite weak-star window admission, requiring W to expose only finitely many source probes and scalar tolerances;*
4. *scalar readback admission, requiring S and R to read all bounds and scalar values through chapter 1500 and chapter 38;*
5. *finite-product handoff exactness, requiring every compactness-facing read to pass through F and the finite product surface of* **??***;*
6. *transport, replay, provenance, and local naming stability, requiring T, C, P, N to preserve the same finite route under componentwise* hsame*, displayed* Cont*, package provenance, and local naming.*

Proof. Project the carrier rows. The displayed Banach source, bounded-linear functional family, finite weak-star window, scalar bound and readback rows, finite-product handoff, transport, replay, provenance, and local naming rows exhaust the visible inventory of A.

The source and functional obligations are read from the cited sibling certificates. The Banach row supplies the normed source data, and the bounded-linear-operator row supplies the finite bounded-functional surface into the scalar real line.

The weak-star content is only the finite row W. Its probes and tolerances are read through S and R, so every scalar comparison remains inside the displayed normed-real and Real readbacks.

Finally inspect a compactness-facing consumer. It may reach only the finite-product row F after the displayed scalar coordinates have been exposed. Since no unrestricted product, ultrafilter, quotient function space, or selected cluster-point row is present, the certificate closes only the finite weak-star window. □

Theorem 3420.3 (Banach-Alaoglu finite dual-ball handoff)**.** *Every accepted* BanachAlaoglu$^{\uparrow}$ *packet exports the route*

$$B, L, W, S, R \longmapsto F \longmapsto T, C, P, N$$

to any duality, compactness, or functional-analysis consumer. The consumer receives a finite weak-star dual-ball window and the corresponding finite product compactness handoff, but receives no unrestricted Tychonoff compactness theorem, no ultrafilter or choice argument, no host weak-star compactness claim, no quotient function space, and no selected cluster point.

Proof. Begin with an accepted packet and a consumer request. The request first reads the Banach source row B and the bounded-linear functional row L, so the functional data is already confined to displayed source and scalar surfaces.

The finite weak-star window W then selects the only probes and scalar tolerances that may be inspected. Rows S and R translate those displayed values and bounds through the normed-real and Real readback interfaces.

Only after these rows are present can the consumer reach F. That row is the finite-product handoff, not an unrestricted product theorem or a compactness assertion over an entire host dual ball.

The structural rows T, C, P, N preserve and replay the same finite route. Because the carrier contains no selector, quotient function space, ultrafilter, or unrestricted compactness coordinate, the exported handoff is exactly the finite weak-star dual-ball window named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachAlaoglu}\uparrow}$.

Constructive story: A $\mathsf{BanachAlaoglu}^{\uparrow}$ packet is generated from a Banach source row, a bounded-linear functional family row, finite weak-star seminorm windows, Real and normed-real scalar readbacks, a finite-product handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachAlaoglu}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3420.1, Definition 3420.2, and Definition 3420.3 seed the finite weak-star dual-ball window consumed by later compactness and duality routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachAlaoglu}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted Tychonoff compactness, ultrafilter or choice arguments, host weak-star compactness, quotient function spaces, selected cluster points, ambient dual-space compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-linear functional family admission, finite weak-star window exactness, scalar readback admission, finite-product handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.BanachAlaogluUp.

Chapter 3421

A Concrete Naming Certificate for $\mathsf{MonotoneCauchySequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{MonotoneCauchySequence}^{\uparrow}$ packet records the finite carrier surface for a monotone regular rational source whose tail windows are already prepared for constructive real completion. It sits between the $\mathsf{Real}^{\uparrow}$ rows of chapter 38, the regular sequence discipline of $\mathsf{RegSeqRat}^{\uparrow}$, the $\mathsf{StreamName}^{\uparrow}$ window ledgers, the dyadic comparison rows of $\mathsf{DyadicRatCore}^{\uparrow}$, and the limit-facing boundary of chapter 1413. The packet names the Cauchy tail surface available before a completion consumer asks for a completed real.

Definition 3421.1 (MonotoneCauchySequence carrier)**.** *A* $\mathsf{MonotoneCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, R, D, O, T, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for the inspected sequence windows.*
2. *W is the stream-window ledger recording the finite index intervals and tail requests visible to the packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that ties each scheduled window to a regular rational approximation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row used to compare rational approximants inside common dyadic windows.*
5. *O is the monotonicity ledger, listing the finite comparison rows by which later inspected approximants do not move below earlier inspected approximants.*
6. *T is the tail-threshold row, assigning each requested precision a displayed tail window and dyadic comparison bound.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row that exposes the regular Cauchy source to real-name consumers without choosing a host limit.*
8. *H records componentwise* hsame *transport for the stream schedule, window ledger, regular readback, dyadic comparisons, monotonicity ledger, tail threshold, real seal, replay route, provenance, and local naming row.*
9. *C records displayed* Cont *replay from a convergence request through $S, W, R, D, O, T, A, H, P, N$.*

10. *P is the* Pkg *provenance over* MonotoneCauchySequence$^\uparrow$, Real$^\uparrow$, RegSeqRat$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RealSequenceLimit$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MonotoneCauchySequence}^\uparrow}$ *row.*

The classifier compares accepted packets only through

$$S, W, R, D, O, T, A, H, C, P, N.$$

It has no coordinate for quotient streams, selected host limits, host real equality, countable choice, trichotomy, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchySequenceUp.MonotoneCauchySequenceTasteGate single carrier alignment`

Theorem 3421.2 (MonotoneCauchySequence NameCert obligations)**.** *For every accepted* MonotoneCauchySequenc *carrier*

$$M = (S, W, R, D, O, T, A, H, C, P, N),$$

the local NameCert$_{\text{MonotoneCauchySequence}^\uparrow}$ *surface consists exactly of the finite rows for carrier admission, stream-window scheduling, regular rational readback, dyadic comparison, monotonicity-ledger admission, tail-threshold admission,* Real$^\uparrow$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each obligation reads the same* StreamName$^\uparrow$ *windows and the same* RegSeqRat$^\uparrow$ *source rows before the tail threshold or real seal is exposed. No obligation may appeal to quotient streams, selected host limits, host real equality, countable choice, trichotomy, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the carrier in Definition 3421.1. Carrier admission exposes precisely the rows $S, W, R, D, O, T, A, H, C, P, N$, so every inspected approximation is read as a scheduled regular rational row through the displayed stream windows.

The order-facing and Cauchy-facing rows use the same source schedule. The monotonicity ledger O compares only approximants exposed by S, W, R, D, while the tail-threshold row T assigns each requested precision a finite window and dyadic comparison bound over the same readback surface.

The real-facing seal comes after those rows. The Real$^\uparrow$ seal A may be exposed only from the regular rational source, dyadic comparison row, monotonicity ledger, and tail threshold already present in the carrier, and a RealSequenceLimit$^\uparrow$ consumer can read only that displayed seal.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, host-limit, host-equality, choice, trichotomy, and ambient-completeness coordinates do not occur in the carrier, they cannot occur among the local NameCert obligations. □

Theorem 3421.3 (MonotoneCauchySequence tail-window boundary)**.** *Let*

$$M = (S, W, R, D, O, T, A, H, C, P, N)$$

be an accepted MonotoneCauchySequence$^\uparrow$ *carrier. Every* RealSequenceLimit$^\uparrow$ *or completion-facing consumer of M must first read the* StreamName$^\uparrow$ *windows,* RegSeqRat$^\uparrow$ *readback,* DyadicRatCore$^\uparrow$ *comparisons, monotonicity ledger, tail-threshold row,* Real$^\uparrow$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *through the same finite surface. The consumer receives a monotone Cauchy tail-window route, not a quotient stream, selected host limit, host real equality, countable-choice tail, trichotomy decision, or ambient* Real$^\uparrow$ *completeness theorem.*

Proof. Start with a limit-facing or completion-facing consumer. By Definition 3421.1, the only available sequence data are the scheduled $\mathsf{StreamName}^{\uparrow}$ windows S, W, the regular rational readback row R, and the dyadic comparison row D.

Apply Definition 3421.2. The monotonicity ledger O and tail-threshold row T are attached to that same scheduled source surface, and the $\mathsf{Real}^{\uparrow}$ seal A is available only after both the dyadic comparison and tail threshold have been displayed.

The $\mathsf{RealSequenceLimit}^{\uparrow}$ route may therefore inspect a finite monotone Cauchy boundary before reaching any real-name consumer. A completion-facing route receives the same boundary through the real seal, not a completed quotient stream or selected host limit.

The structural rows H, C, P, N preserve this boundary under transport, replay, provenance, and naming. The absent quotient-stream, host-limit, host-equality, choice, trichotomy, and ambient-completeness coordinates remain absent for every downstream consumer. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneCauchySequence}^{\uparrow}$ consumer that claims access to a completion or limit boundary must display the StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, monotonicity ledger, tail threshold, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert before any RealSequenceLimit or completion read is inspected.

Independence witness: real, regseqrat, streamname, dyadicratcore, realsequencelimit: MonotoneCauchySequence is not bijective to any listed sibling because it jointly carries monotonicity, regular rational readback, dyadic tail thresholds, and a real seal before any limit-facing consumer may read the source sequence.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3421.1, Definition 3421.2, and Definition 3421.3 seed the finite source carrier, local obligation surface, and tail-window boundary for monotone Cauchy sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected host limits, host real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, stream-window scheduling, regular readback, dyadic comparison, monotonicity-ledger admission, tail-threshold admission, real sealing, transport replay, provenance, and local naming for BEDC.Derived.MonotoneCauchySequenceUp.

Constructive story: A $\mathsf{MonotoneCauchySequence}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore comparisons, monotonicity and tail-threshold ledgers, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3422

A Concrete Naming Certificate for MetaCICCandidateSetSNHandoff$^{\uparrow}$

The MetaCICCandidateSetSNHandoff$^{\uparrow}$ packet realizes the candidate-set strong-normalization handoff of chapter 9383 as finite NameCert data. It is narrower than the candidate-set normalization surface of chapter 2320: this packet records exactly the route from Candidate and CandidateSet membership evidence to a closed typed term strong-normalization export, with beta-closure and no-infinite-reduction kept visible. It sits beside the MetaCIC critical path of chapter 765, the closed substitution boundary of chapter 527, the finite trace evaluator of chapter 818, the kernel category surface of chapter 488, and the halting distinction packet of chapter 491.

Definition 3422.1 (MetaCIC candidate-set SN handoff carrier)**.** *A* MetaCICCandidateSetSNHandoff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, S, T, B, E, N, R, C, P, L)$$

with the following displayed rows:

1. *A is the Candidate row, naming the reducibility-candidate evidence admitted by the normalization audit map.*
2. *S is the CandidateSet membership row, recording finite membership, soundness, and the selected candidate-set witness consumed by the handoff.*
3. *T is the closed typed term evidence row, recording the empty-context term, its type, and the closedness evidence to which the candidate-set row is attached.*
4. *B is the beta-closure row, exposing the one-step beta stability supplied by the same CandidateSet route.*
5. *E is the strong-normalization export row, read only after A, S, T, B have all been displayed.*
6. *N is the no-infinite-reduction projection row obtained from the exported strong-normalization evidence.*
7. *R records componentwise* hsame *transport for candidate, candidate-set, closed-term, beta-closure, export, no-infinite, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a closed-term request through CandidateSet membership, beta closure, strong-normalization export, and no-infinite-reduction projection.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$*, Candidate, CandidateSet, closed typed term evidence, beta-closure evidence, strong-normalization export,* BHist, hsame, Cont, Pkg*, and* NameCert *rows.*

10. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, T, B, E, N, R, C, P, L$. It has no coordinate for candidate-free strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, or a proof-producing oracle outside the displayed handoff.

Theorem 3422.2 (MetaCIC candidate-set SN handoff obligations)**.** *For every accepted* MetaCICCandidateSetSNHa *carrier $H = (A, S, T, B, E, N, R, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission; Candidate evidence through A; CandidateSet membership, soundness, and witness exposure through S; closed typed term evidence through T; beta-closure stability through B; strong-normalization export through E; no-infinite-reduction projection through N; componentwise transport through R; displayed replay through C; provenance through P; and local naming through L.*

Proof. Project the accepted carrier of Definition 3422.1. The visible rows are exactly $A, S, T, B, E, N, R, C, P, L$.

The positive route begins with A and S. Candidate evidence and CandidateSet membership are displayed before the closed typed term row can be read as a normalization input.

The closed-term and beta rows are T and B. They keep the empty-context typing, closedness, and one-step beta stability attached to the same CandidateSet witness.

The export rows are E and N. Strong normalization is exported only after A, S, T, B are present, and no-infinite-reduction is only a projection from that export. The transport, replay, provenance, and naming rows preserve the same handoff, so no excluded host route can appear in the NameCert surface. □

Theorem 3422.3 (MetaCIC candidate-set SN export nonescape)**.** *Every public strong-normalization or no-infinite-reduction read exported by an accepted* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *packet factors through*

$$A, S \longmapsto T, B \longmapsto E, N \longmapsto R, C, P, L.$$

Thus a closed typed term consumer receives the route only as CandidateSet membership, closed-term, beta-closure, and SN export NameCert data, not as an unconditional theorem over all closed MetaCIC terms.

Proof. Read the source rows A and S. They name the candidate evidence and CandidateSet membership that the normalization audit map requires for the public closed-term route.

Next read T and B. The closed typed term row and beta-closure row attach the term and its local reduction stability to the same finite evidence route.

Finally read E, N, R, C, P, L. The strong-normalization export and no-infinite projection are downstream of the candidate-set source, while transport, replay, provenance, and local naming preserve that order. Since the carrier has no coordinate for candidate-free strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, or a proof-producing oracle, no consumer can escape the displayed route. □

Theorem 3422.4 (MetaCIC candidate-set SN export factorization)**.** *For every accepted* MetaCICCandidateSetSNH *carrier*

$$H = (A, S, T, B, E, N, R, C, P, L),$$

each strong-normalization export row factors through the displayed order

$$A \longmapsto S \longmapsto T \longmapsto B \longmapsto E \longmapsto N \longmapsto R \longmapsto C \longmapsto P \longmapsto L.$$

Thus the export is mediated first by Candidate evidence, then CandidateSet membership, closed typed term evidence, beta-closure stability, strong-normalization export, no-infinite projection, transport, replay, provenance, and local NameCert rows.

Proof. Project the accepted carrier of Definition 3422.1. The rows are displayed as $A, S, T, B, E, N, R, C, P, L$, and the local classifier compares accepted packets only through those coordinates.

The candidate-mediated source is A followed by S. The closed typed term row T and beta-closure stability row B can be read only as part of the same finite CandidateSet handoff.

The export itself is then E, with N only as the no-infinite-reduction projection from that export. The nonescape theorem Definition 3422.3 rules out a candidate-free or host-oracle route for these public reads.

Finally the structural rows R, C, P, L transport, replay, source, and name the same route. This gives the stated factorization and keeps the strong-normalization export inside the local NameCert surface. □

Theorem 3422.5 (MetaCIC candidate-set SN TasteGate route)**.** *Every strong-normalization export from an accepted* $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *carrier*

$$H = (A, S, T, B, E, N, R, C, P, L)$$

passes through Candidate evidence A, CandidateSet membership S, closed typed term evidence T, beta-closure stability B, no-infinite-reduction projection N, componentwise transport R, replay C, provenance P, and the local NameCert row L. These rows are exactly the field projections that a future TasteGate field-faithful instance for $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ *must preserve when it exposes the strong-normalization export E.*

```
Lean4 ✓  BEDC.Derived.MetaCICCandidateSetSNHandoffUp.MetaCICCandidateSetSNHandoffTasteGate
single carrier alignment
```

Proof. Project the accepted carrier using Definition 3422.1. The field inventory is $A, S, T, B, E, N, R, C, P, L$, and the classifier compares accepted packets only through those displayed coordinates.

Apply Definition 3422.4. It orders the SN export through Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure, export, no-infinite projection, transport, replay, provenance, and local naming.

The TasteGate field-faithful requirement therefore has no additional field to preserve and no license to merge fields. Preserving the handoff means preserving each of $A, S, T, B, E, N, R, C, P, L$ as a displayed projection.

Since the carrier has no coordinate for candidate-free strong normalization, host evaluator normalization, quotient conversion equality, or an external proof oracle, the same field-faithful route is the only route available to any SN export consumer. □

Falsifiable prediction: A closed typed term read that reaches $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$ strong-normalization export without displaying CandidateSet membership, closed typed term evidence, beta-closure stability, and the SN export row fails the carrier surface.

Independence witness: metaciccriticalpath, closedtermsubstitutionboundary, finitetraceevaluator, kernelcategory, haltingdistinction: none is bijective because $\mathsf{MetaCICCandidateSetSNHandoff}^{\uparrow}$

binds Candidate evidence, CandidateSet membership, closed typed term evidence, beta closure, SN export, no-infinite projection, transport, replay, provenance, and local NameCert rows in one finite handoff packet.

The Lean follow-up is `BEDC.Derived.MetaCICCandidateSetSNHandoffUp`. Its TasteGate surface must provide Nontrivial MetaCICCandidateSetSNHandoff$^{\uparrow}$ and FieldFaithful MetaCICCandidateSetSNHandoff$^{\uparrow}$ instances that keep $A, S, T, B, E, N, R, C, P, L$ as separate displayed fields.

Closure status for NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$.

Constructive story: A MetaCICCandidateSetSNHandoff$^{\uparrow}$ packet is generated from Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure evidence, strong-normalization export, no-infinite-reduction projection, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 3422.1, Definition 3422.2, Definition 3422.3, and Definition 3422.4 seed the finite candidate-set strong-normalization handoff surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateSetSNHandoff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unconditional MetaCIC strong normalization, closed subject reduction, global confluence, host evaluator normalization, quotient conversion equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Candidate evidence, CandidateSet membership, closed typed term evidence, beta-closure stability, SN export, no-infinite projection, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MetaCICCandidateSetSNHandoffUp.

Chapter 3423

A Concrete Naming Certificate for $\mathsf{FilterConvergence}^{\uparrow}$

Following Kelley and Bourbaki on filter convergence, the $\mathsf{FilterConvergence}^{\uparrow}$ packet records the finite convergence interface needed by the L10 completion route. It sits between the finite $\mathsf{CauchyFilter}^{\uparrow}$ surface of chapter 333, the regular directed-tail surface of chapter 1445, the window discipline of chapter 126, the ambient metric interface of chapter 57, and the terminal real-name seal of chapter 38. The packet names convergence toward a displayed target point through visible tail events and tolerance rows; it does not quotient filters, select host limits, invoke countable choice, hide cofinal tails, or assume ambient real completeness.

Definition 3423.1 (FilterConvergence carrier)**.** *A* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, X, T, V, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the visible* $\mathsf{CauchyFilter}^{\uparrow}$ *row. It stores the finite filterbase requests and refinement evidence from which convergence may be read.*
2. *X is the target point row over* $\mathsf{Met}^{\uparrow}$*. It records the displayed point classifier, distance observations, and metric-law ledger used by every neighborhood request.*
3. *T is the finite tail-event schedule. Each row names a filter event requested around the target point and the finite stage at which that event is inspected.*
4. *V is the Metric/Real tolerance row. It matches the metric ball or entourage demand around X with the rational and real-facing tolerance data read by the convergence request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is available only after the filter row, target point row, tail schedule, and tolerance row have all been displayed.*
6. *H records componentwise* hsame *transport for B, X, T, V, E and for the structural rows below.*
7. *C records displayed* Cont *replay from filterbase refinement to tail events, from target-point neighborhoods to tolerance windows, and from tolerance windows to the Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{FilterConvergence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, X, T, V, E, H, C, P, N$. There is no coordinate for quotient filters, selected host limits, hidden cofinal tails, countable-choice schedules, host equality of target points, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.FilterConvergenceUp.FilterConvergenceTasteGate single carrier alignment`

Theorem 3423.2 (FilterConvergence NameCert obligations)**.** *For every accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier $F = (B, X, T, V, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, visible filter exposure through B, target-point exposure through X, finite tail-event exposure through T, Metric/Real tolerance readback through V, terminal Real seal handoff through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the convergence route

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

without adding quotient filters, selected limits, hidden cofinal tails, countable choice, host target equality, or ambient real completeness.

Proof. Project the carrier of Definition 3423.1. The packet exposes exactly the visible filter row, target point row, tail-event schedule, tolerance row, Real seal, and structural rows.

The filter and target rows are read first. The row B supplies only finite filterbase requests, while X supplies the metric target point and its displayed neighborhood observations.

The tail schedule and tolerance row then bind convergence to finite data. The schedule T lists the filter events inspected around X, and V ties each such event to the Metric and Real tolerance windows consumed by the request.

Finally E, H, C, P, N seal, transport, replay, source, and name the same ordered route. Since the excluded objects are absent from the carrier, no further local NameCert obligation can be exported. □

Theorem 3423.3 (FilterConvergence tail stability)**.** *Let $F = (B, X, T, V, E, H, C, P, N)$ be an accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *carrier. Any two displayed tail events in T that are read around the same target point row X admit a later displayed filterbase refinement inside B whose Metric/Real tolerance data is still the row V and whose terminal seal is still E.*

Consequently tail refinement is stable under componentwise hsame *transport, displayed* Cont *replay, package provenance, and local naming through H, C, P, N. The stability claim does not use quotient filters, selected host limits, countable choice, hidden cofinal tails, host target equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with two tail events in the finite schedule T. Because both events are carrier rows, they are already tied to the same visible filterbase row B and the same target point row X.

Use the directed refinement surface read from B. The later filterbase row is displayed inside the packet, so the proof never passes to an external cofinal family or quotient filter.

Read the tolerance row next. The row V is the only Metric/Real tolerance interface attached to the scheduled tail events, and H, C transport and replay that same interface to the later refinement.

The Real seal is therefore unchanged. Since E is downstream from B, X, T, V, and P, N only source and name those coordinates, the excluded choice, limit, hidden-tail, equality, and completeness principles cannot enter the tail-stability route. □

Theorem 3423.4 (FilterConvergence Real seal handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{FilterConvergence}^{\uparrow}$ *packet must read the displayed convergence route*

$$B \longmapsto X \longmapsto T \longmapsto V \longmapsto E$$

before the Real seal is available. In BEDC terms, the consumer passes through the visible $\mathsf{CauchyFilter}^{\uparrow}$ *row, the target* $\mathsf{Met}^{\uparrow}$ *point row, the* $\mathsf{StreamName}^{\uparrow}$ *tail-event schedule, and the Metric/Real tolerance row before it reaches* $\mathsf{Real}^{\uparrow}$*.*

The handoff supplies a finite convergence interface for $\mathsf{RegularCauchyFilter}^{\uparrow}$ *and downstream completion-limit uniqueness arguments. It supplies no quotient filter, selected host limit, countable-choice schedule, hidden cofinal tail, host equality of target points, or ambient real-completeness principle.*

Proof. Start with the visible filter row B. A consumer can inspect only the finite filterbase requests and refinements recorded by the packet.

Next read the target and schedule rows. The row X supplies the metric target point and neighborhood observations, while T records the finite tail events that answer those neighborhood requests.

The tolerance and seal rows are downstream. The row V matches those tail events with the Metric and Real tolerance windows, and E seals the compatible readback only after that route has been replayed.

Thus every Real-facing consumer receives the same ordered route through $B, X, T, V, E, H, C, P, N$. Since no excluded principle is a carrier coordinate, the handoff is a finite convergence certificate rather than an ambient completeness assertion. □

Theorem 3423.5 (Filter convergence directed-net index)**.** *Any* $\mathsf{FilterConvergence}^{\uparrow}$ *packet that consumes* $\mathsf{MooreSmithNet}^{\uparrow}$ *data must expose the* $\mathsf{DirectedSet}^{\uparrow}$ *index row and the* $\mathsf{FilterBase}^{\uparrow}$ *handoff before any* $\mathsf{NetConvergence}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer reads convergence. The convergence route therefore passes through displayed directed indices, directed tails, and filterbase refinement data, not through host cofinal selectors, quotient net indices, hidden tail predicates, or ambient convergence relations.*

Proof. Project the filter-convergence carrier of Definition 3423.1. The visible filter row B, target row X, tail schedule T, tolerance row V, and seal row E are the only convergence-facing coordinates.

Read the Moore–Smith net carrier through chapter 3403. Its displayed route exposes a $\mathsf{DirectedSet}^{\uparrow}$ index row before the $\mathsf{FilterBase}^{\uparrow}$ handoff and before any $\mathsf{NetConvergence}^{\uparrow}$ or $\mathsf{UniformCompletion}^{\uparrow}$ consumer is named.

Apply the DirectedSet filter-net handoff of Definition 1607.4. It forces the finite index row, preorder comparisons, finite request windows, and upper-window witnesses to be visible before filter or net convergence data is consumed.

Since neither the Moore–Smith route nor the filter-convergence carrier contains host cofinal selectors, quotient net indices, hidden tail predicates, or ambient convergence relations, those objects cannot enter through filter notation. The convergence read remains the displayed route through $B, X, T, V, E, H, C, P, N$ after the directed-index and filterbase rows have been exposed. □

Falsifiable prediction: Every accepted $\mathsf{FilterConvergence}^{\uparrow}$ read factors through the visible $\mathsf{CauchyFilter}^{\uparrow}$ row, target $\mathsf{Met}^{\uparrow}$ point row, finite tail-event schedule, Metric/Real tolerance row, $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any completion-facing consumer reads convergence data.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, metric, real: $\mathsf{FilterConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a visible filter row, target point row, finite tail-event schedule, Metric/Real tolerance row, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterConvergence}^{\uparrow}$ packet is generated from a visible CauchyFilter row, a target Metric point row, a finite StreamName tail-event schedule, a Metric/Real tolerance row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3423.1, Definition 3423.2, Definition 3423.3, and Definition 3423.4 seed the finite convergence interface from visible filter rows and target metric neighborhoods to Real-facing seal data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected host limits, countable choice, hidden cofinal tails, host target equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible filter exposure, target-point exposure, finite tail-event exposure, Metric/Real tolerance readback, Real seal handoff, transport stability, replay stability, and ledger non-escape for BEDC.Derived.FilterConvergenceUp.

Chapter 3424

A Concrete Naming Certificate for $\mathsf{BishopCompleteMetric}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley completion language, the $\mathsf{BishopCompleteMetric}^{\uparrow}$ packet records the complete metric object reached when displayed Cauchy-filter data, regular rational readback, separated metric identity, and terminal real sealing are assembled as one finite L10 NameCert route. It sits after the metric-completion carrier of chapter 1506, the complete-metric service of chapter 128, the Cauchy-filter evidence of chapter 333, the regular rational readback of chapter 301, the Real classifier seal of chapter 38, and the separated metric classifier of chapter 1411. The packet exposes a named constructive complete metric surface; it does not assert host quotient completion equality, selected representatives, countable choice, trichotomy, or ambient real completeness outside the displayed rows.

Definition 3424.1 (Bishop complete metric carrier)**.** *A* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, F, R, S, C, E, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, including the displayed metric carrier, completion handoff, and finite tolerance readback surface.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *evidence row, recording the Cauchy-filter route from which completion-facing point names are admitted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational windows used by Real-facing consumers of the same completion data.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *evidence row, recording the zero-distance classifier boundary available for completed point equality.*
5. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording the displayed limit, modulus, and convergence-distance ledger for accepted regular Cauchy point names.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after M, F, R, S, C have been displayed.*
7. *H records componentwise* hsame *transport for the metric-completion, Cauchy-filter, regular-readback, separatedness, complete-metric, Real-seal, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from a complete-metric consumer through $M, F, R, S, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, S, C, E, H, K, P, N$. It has no coordinate for quotient completion equality, selected representatives, host metric equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompleteMetricUp.BishopCompleteMetricTasteGate single carrier alignment`

Theorem 3424.2 (Bishop complete metric NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier $B = (M, F, R, S, C, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion source, Cauchy-filter evidence, regular rational readback, separated metric classifier, complete-metric service, terminal Real seal, componentwise transport, displayed continuation replay, provenance, and local naming rows. Every complete-metric read is therefore routed through $M, F, R, S, C, E, H, K, P, N$, and no Bishop complete metric obligation is stored outside those rows.*

Proof. Project the carrier of Definition 3424.1. The visible packet consists exactly of $M, F, R, S, C, E, H, K, P, N$, so carrier admission, transport, continuation replay, provenance, and local naming are exhausted by those displayed rows.

Read the analytic dependencies in dependency order. The metric-completion source M admits the completion route, F records the Cauchy-filter evidence feeding that route, R supplies regular rational readback for Real-facing observations, S supplies the separated zero-distance classifier, and C supplies the complete-metric limit service.

The terminal seal E is downstream of those rows, while H, K, P, N preserve the same route by transport, replay, provenance, and naming. Since quotient completion equality, representatives, host equality, countable choice, trichotomy, and ambient real completeness have no carrier coordinate, they cannot appear as local obligations. □

Theorem 3424.3 (Bishop complete metric completion handoff)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *carrier must pass through the ordered route*

$$M, F, R, S, C, E, H, K, P, N$$

before reading the terminal complete metric seal. The handoff exposes metric-completion data, Cauchy-filter evidence, regular rational readback, separated metric identity, complete-metric limit service, Real sealing, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, but it exports no quotient completion equality, selected representative, countable-choice selector, host metric equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from a completion-facing consumer of the accepted carrier. The only completion source row is M, and the only Cauchy evidence row feeding it is F.

Move next through the readback and classifier boundary. The row R exposes the regular rational windows used by Real-facing reads, while S is the separated metric route by which zero-distance comparisons are admitted.

Finally read C and E, then replay the structural rows H, K, P, N. The absence of quotient, representative, choice, host-equality, trichotomy, and ambient-completeness coordinates in Definition 3424.1 forces every accepted handoff to remain on the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompleteMetric}^{\uparrow}$ completion or Real read factors through $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metriccompletion, completemetric, cauchyfilter, regseqrat, real, separatedmetric: $\mathsf{BishopCompleteMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the metric-completion source, Cauchy-filter evidence, regular rational readback, separated metric classifier, complete-metric service, terminal Real seal, transport, continuation, provenance, and local naming as one finite complete-metric packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3424.1, Definition 3424.2, and Definition 3424.3 seed the complete-metric surface over $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host metric equality, countable choice, trichotomy, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter exposure, regular rational readback, separated classifier exactness, complete-metric service compatibility, and completion-handoff nonescape over the same displayed packet.

Constructive story: The construction starts from a metric-completion source, records Cauchy-filter evidence and regular rational readback, passes through separated metric identity and complete-metric limit service, and exposes the terminal Real seal only along the same finite NameCert route.

Chapter 3425

A Concrete Naming Certificate for FredholmAlternative$^{\uparrow}$

Following Fredholm, Riesz, and compact-operator treatments of linear equations, the FredholmAlternative$^{\uparrow}$ packet records the finite BEDC surface for the kernel–range obstruction alternative attached to a displayed compact perturbation. It is read after the compact-operator certificate of chapter 1952, the Banach and Hilbert carrier rows of chapter 90 and chapter 91, the bounded-operator action surface of chapter 1644, the separation witness of chapter 3401, and the variational handoff of chapter 3419. The packet names only this displayed finite alternative route; it does not import a host Fredholm theorem, a quotient index equality, a choice-selected inverse, or an ambient spectral decomposition.

Definition 3425.1 (Fredholm alternative carrier)**.** *A* FredholmAlternative$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

with the following visible rows:

1. *S is the source endpoint row, read through the* Banach$^{\uparrow}$ *and, when an orthogonal witness is present,* Hilbert$^{\uparrow}$ *certificate surfaces;*
2. *T is the target endpoint row, read through the same Banach or Hilbert certificate discipline;*
3. *B is the* BoundedLinearOperator$^{\uparrow}$ *row naming the displayed linear action and norm-bound ledger between S and T;*
4. *K is the* CompactOperator$^{\uparrow}$ *row naming the compact perturbation witness by finite image-net schedules and tail moduli;*
5. *L is the* LaxMilgram$^{\uparrow}$ *or bounded-action handoff row used by variational consumers that read the same operator boundary;*
6. *V is the* RieszLemma$^{\uparrow}$ *separation row available when a range-obstruction witness is exposed;*
7. *R is the kernel witness row, recording the finite nonzero-kernel or kernel-refusal evidence attached to the displayed operator equation;*
8. *A is the range-obstruction witness row, recording the finite cokernel-facing obstruction, separation functional, or refusal of surjectivity over the same endpoints;*
9. *H stores componentwise* hsame *transport for the endpoint, bounded-action, compactness, variational, separation, kernel, and obstruction rows;*

10. *C stores displayed* Cont *replay for the same rows;*
11. *P is the* Pkg *provenance over* FredholmAlternative$^{\uparrow}$, CompactOperator$^{\uparrow}$, Banach$^{\uparrow}$, Hilbert$^{\uparrow}$, BoundedLinearOperato RieszLemma$^{\uparrow}$, LaxMilgram$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert*;*
12. *N is the local* NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host inverse, quotient-space equality, index formula, selected basis of the kernel, selected complement of the range, or external spectral theorem.

Lean4 ✓ `BEDC.Derived.FredholmAlternativeUp.FredholmAlternativeTasteGate single carrier alignment`

Theorem 3425.2 (Fredholm alternative NameCert obligations)**.** *For the carrier of Definition 3425.1, the local* NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ *surface consists of exactly six finite obligations:*

1. *endpoint admission through the displayed* Banach$^{\uparrow}$ *or* Hilbert$^{\uparrow}$ *source and target rows;*
2. *bounded-action admission through the* BoundedLinearOperator$^{\uparrow}$ *row B;*
3. *compact-perturbation admission through the* CompactOperator$^{\uparrow}$ *row K;*
4. *alternative-witness exposure, requiring the kernel witness R or the range-obstruction witness A to be read as displayed finite rows over the same endpoints;*
5. *separation and variational compatibility, requiring V and L to support only the displayed obstruction or handoff route already present in the carrier;*
6. *transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier rows. The endpoint, bounded-action, compactness, variational, separation, kernel, obstruction, transport, replay, provenance, and local naming entries are the full visible inventory of F.

Read the analytic dependencies next. The Banach or Hilbert rows determine the endpoints, the bounded-operator row determines the action, and the compact-operator row supplies the finite compactness witness. These rows name BEDC certificate data rather than a host operator theorem.

The alternative witness is then local to the displayed packet. If the kernel side is exposed, it is exposed through R; if the range-obstruction side is exposed, it is exposed through A together with the separation row V. The variational row L is only a handoff through the same bounded-action surface.

Finally H, C, P, N preserve this finite route. Since the carrier contains no additional inverse, quotient, index, complement, or spectral coordinate, the six obligations exhaust the local NameCert$_{\text{FredholmAlternative}^{\uparrow}}$ surface. □

Theorem 3425.3 (Fredholm alternative kernel–cokernel handoff)**.** *Let*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

be an accepted FredholmAlternative$^{\uparrow}$ *packet. Every consumer that reads a kernel-facing conclusion or a range-obstruction conclusion must pass through the same endpoint rows S, T, bounded-action row B, compact-operator row K, separation or variational handoff rows V, L, transport row H, replay row C, provenance row P, and local naming row N. The handoff exposes the displayed kernel witness R or the displayed range-obstruction witness A; it exposes no host inverse, quotient equality, Fredholm index, selected kernel basis, selected range complement, or ambient spectral theorem.*

Proof. Begin with the NameCert obligations of Definition 3425.2. They force endpoint admission, bounded-action admission, compact-perturbation admission, and witness exposure to be read from one finite packet.

If a consumer follows the kernel side, the only available kernel-facing row is R, and it is transported by the same H, C, P, N rows as the operator and compactness data. If a consumer follows the range-obstruction side, the obstruction row is A, with the separation support V and the variational handoff L read through the same endpoints.

Thus the two readings are not detached alternatives living over different carriers. They are the two witness exits of one finite route over $S, T, B, K, L, V, H, C, P, N$. The excluded inverse, quotient, index, basis, complement, and spectral objects are absent from the carrier and therefore cannot be exported by the handoff. □

Theorem 3425.4 (Fredholm alternative compact obstruction scope)**.** *Let*

$$F = (S, T, B, K, L, V, R, A, H, C, P, N)$$

be an accepted $\mathsf{FredholmAlternative}^{\uparrow}$ *packet. Every compact-obstruction read exposed by F factors through the endpoint rows S, T, the bounded-action row B, the compact row K, the Riesz separation row V, the Lax–Milgram handoff row L, and exactly one displayed witness exit, either R or A. The transport, replay, provenance, and local naming rows H, C, P, N preserve that route. The read exports no host inverse, quotient index, selected complement, or ambient spectral theorem.*

Proof. Project the finite carrier rows of F. The only analytic rows available to a compact-obstruction consumer are the endpoints S, T, the bounded action B, the compact witness K, the separation row V, and the variational handoff L.

Apply the local NameCert obligations of Definition 3425.2. They force any obstruction witness to be read from the displayed alternative-witness clause, so the consumer must use one of the two witness exits already present in the packet.

Use the kernel–cokernel handoff of Definition 3425.3. The kernel-facing route exits through R, while the range-obstruction route exits through A together with the same compact, separation, and variational rows. In both cases H, C, P, N transport and replay only the displayed finite route.

Since the carrier contains no row for a host inverse, quotient index, selected complement, or ambient spectral theorem, none of these objects can be exported by the compact-obstruction scope. □

Closure status for $\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}$.

Constructive story: A $\mathsf{FredholmAlternative}^{\uparrow}$ packet is generated from Banach or Hilbert endpoint rows, a bounded-operator row, a compact-operator row, variational and separation handoff rows, kernel and range-obstruction witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3425.1, Definition 3425.2, and Definition 3425.3 seed the finite Fredholm-alternative certificate surface consumed by compact-operator, Banach, Hilbert, bounded-operator, Riesz-separation, and Lax-Milgram routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FredholmAlternative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Fredholm theorem, quotient-space equality, index formula, selected inverse, selected kernel basis, selected range complement, or standard bridge.

Upgrade path: Obligation closure requires checked endpoint admission, bounded-action admission, compact-perturbation admission, kernel witness exposure, range-obstruction exposure, separation and variational compatibility, transport, replay, provenance, and local naming for the displayed packet.

Chapter 3426

A Concrete Naming Certificate for $\mathsf{CousinCover}^{\uparrow}$

Following Cousin and Bishop-style gauge integration, the $\mathsf{CousinCover}^{\uparrow}$ packet names the reusable finite gauge-cover carrier below the tagged-subdivision conclusion of chapter 1861. It separates the cover data consumed by $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, and compact interval finite-cover arguments from the lemma-level subdivision packet. The carrier reads located interval rows from chapter 1711, finite cover and radius boundaries from chapter 1596 and chapter 452, and Real-facing names through chapter 126, chapter 301, and chapter 38.

Definition 3426.1 (Cousin cover carrier)**.** *A* $\mathsf{CousinCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, G, T, M, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval row, including named endpoints, endpoint order, containment tests, and the interval boundary inspected by finite bisection and cover consumers.*
2. *G is the positive gauge row. Each displayed tag request carries a positive local radius read through rational and Real-facing witnesses before any cover cell is admitted.*
3. *T is the tagged cover-cell row. Each entry records a tag, a located interval cell, containment in I, local admission by the radius in G, and the finite handoff shape read by* $\mathsf{CousinLemma}^{\uparrow}$*.*
4. *M is the mesh and refinement ledger recording endpoint comparisons, adjacency, overlap permissions, and the finite refinement relation between a cover row and any smaller admitted cover row.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule exposed for endpoint, tag, gauge-radius, cell, and mesh observations in I, G, T, M.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback attached to the displayed windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite windows and regular rational readbacks in W, R have been displayed.*
8. *H records componentwise* hsame *transport for interval, gauge, tagged cover, mesh, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a gauge-cover request through $I, G, T, M, W, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{IntervalFiniteCoverCertificate}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, G, T, M, W, R, E, H, C, P, N$. It has no coordinate for general open-cover compactness, a choice-selected subcover, a host real completeness theorem, quotient stream equality, or an external interval object.

Lean4 ✓ `BEDC.Derived.CousinCoverUp.CousinCoverTasteGate single carrier alignment`

Theorem 3426.2 (Cousin cover NameCert obligations)**.** *For every accepted* $\mathsf{CousinCover}^{\uparrow}$ *carrier*

$$Q = (I, G, T, M, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, positive gauge, tagged cover cells, mesh and refinement ledger, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *gauge locality, requiring each tagged cover cell in T to read its radius from G and to prove containment in that local gauge window;*
3. *refinement coverage, requiring M to record that the finite cell family covers the displayed interval row and that every accepted refinement remains inside the same interval and gauge rows;*
4. *finite tagged-cover handoff, requiring downstream tagged-subdivision consumers to read the cover cells through T, M, W, R, E before reaching H, C, P, N;*
5. *ledger non-escape, excluding general open-cover compactness, selected subcovers, host interval equality, quotient stream equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3426.1. The tuple displays the located interval, gauge, tagged cover cells, refinement ledger, finite window schedule, readback, terminal Real seal, and structural rows.

Gauge locality is read from G and T. A cell cannot be admitted unless its tag has a displayed positive radius and the cell containment test is recorded inside that local gauge window.

Refinement coverage is read from M. The ledger states both that the finite cell family covers the interval row I and that any smaller cover cell remains attached to the same interval and gauge source.

The tagged-cover handoff is the ordered route through T, M, W, R, E. The cover cells and their mesh ledger are exposed before the StreamName windows, regular rational readbacks, and terminal Real seal can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs in the carrier, quotient stream equality, selected subcovers, open-cover compactness, countable choice, host interval equality, and ambient Real completeness cannot be projected from an accepted packet. □

Theorem 3426.3 (Cousin cover gauge refinement handoff)**.** *Every* $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, *or interval finite-cover consumer of an accepted* $\mathsf{CousinCover}^{\uparrow}$ *packet reads the cover through the ordered carrier*

$$I, G, T, M, W, R, E, H, C, P, N.$$

Thus the reusable cover surface is the finite gauge-cover packet over located intervals, tagged cells, mesh refinement, StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal. No consumer receives a detached open cover, a choice-selected refinement, host interval equality, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a downstream consumer read. Carrier admission exposes the located interval row I and the positive gauge row G before any cover cell can be inspected.

The tagged cover row T and the mesh ledger M then provide the finite refinement surface. This is the part consumed by $\mathsf{CousinLemma}^{\uparrow}$ when it forms a tagged subdivision, and by $\mathsf{GaugeIntegral}^{\uparrow}$, Darboux, Riemann, and interval finite-cover consumers when they need a gauge-local finite cover.

The Real-facing portion of the read passes through W, R, E. These rows expose only finite StreamName windows, RegSeqRat readback, and the terminal Real seal attached to the interval, tag, gauge-radius, cell, and mesh observations already displayed.

The remaining rows H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the carrier has no detached open-cover or host-completeness coordinate, the consumer handoff is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CousinCover}^{\uparrow}$ packet is generated from a located interval row, a positive gauge row, finite tagged cover cells, mesh and refinement ledgers, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3426.1, Definition 3426.2, and Definition 3426.3 seed the reusable finite gauge-cover route from located intervals and positive gauges through tagged cover cells, mesh refinement, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CousinCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, selected subcovers, host interval equality, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge locality, refinement coverage, finite tagged-cover handoff, and ledger non-escape for the Cousin cover packet.

Chapter 3427

A Concrete Naming Certificate for $\mathsf{RealUniformTailComparison}^{\uparrow}$

Following Bishop and Bridges constructive real sequence/tail-modulus discipline, the $\mathsf{RealUniformTailComparison}^{\uparrow}$ packet records a finite uniform comparison surface for two inspected real-sequence tails. It sits between the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window route of chapter 126, the dyadic tolerance ledger of chapter 291, the sequence-limit consumer of chapter 1413, and the limsup-facing envelope of chapter 768. The packet supplies a named L10 boundary for consumers that compare two tails under one displayed tolerance budget; it does not choose a limit, quotient streams, decide trichotomy, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3427.1 (Real uniform tail comparison carrier)**.** *A* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S_A, S_B, W_A, W_B, Q_A, Q_B, D, T, L, \Lambda, H, C, P, N)$$

with the following displayed rows:

1. *S_A and S_B are the two real-sequence source rows whose tails are compared.*
2. *W_A and W_B are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows exposed for the two inspected tails.*
3. *Q_A and Q_B are the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback rows attached to those windows.*
4. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every comparison request.*
5. *T is the finite tail-budget row naming the common index floor and precision request.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *consumer row that may read the comparison only after the windows, readbacks, ledger, and tail budget have been displayed.*
7. *Λ is the* $\mathsf{Limsup}^{\uparrow}$ *consumer row that may read the same comparison as an upper-tail envelope without selecting a subsequential limit.*
8. *H records componentwise* hsame *transport for the two source routes, the shared ledger, the tail budget, and the two consumer rows.*
9. *C records displayed* Cont *replay from a tail-comparison request through the windows, regular readbacks, dyadic ledger, tail budget, sequence-limit row, and limsup row.*
10. *P is the* Pkg *provenance over* $\mathsf{RealUniformTailComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$ *row.*
The carrier contains no coordinate for quotient stream equality, a selected limiting representative, countable choice, trichotomy of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealUniformTailComparisonUp.RealUniformTailComparisonTasteGate single carrier alignment`

Theorem 3427.2 (Real uniform tail comparison NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$ *surface for Definition 3427.1 consists exactly of source exposure for S_A, S_B, finite-window exposure for W_A, W_B, regular readback exposure for Q_A, Q_B, shared dyadic tolerance exposure for D, common tail-budget exposure for T, sequence-limit handoff through L, limsup handoff through Λ, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the displayed packet $U = (S_A, S_B, W_A, W_B, Q_A, Q_B, D, T, L, \Lambda, H, C, P, N)$. The comparison route is finite: a request reads the two source rows, the two StreamName windows, the two regular rational readbacks, the shared dyadic tolerance ledger, and the common tail budget before it reaches either consumer row.

The structural rows preserve that same route. Componentwise hsame transport keeps the two tail presentations and the shared ledger in their displayed sibling surfaces, while Cont replay records the order in which the request reaches the sequence-limit and limsup consumers.

No excluded object is a row of the carrier. Therefore the listed rows exhaust the seed NameCert obligations for the uniform tail-comparison packet. □

Theorem 3427.3 (Real uniform tail comparison RegSeqRat handoff)**.** *Every* $\mathsf{RealSequenceLimit}^{\uparrow}$ *or* $\mathsf{Limsup}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *packet must read the two* $\mathsf{RegSeqRat}^{\uparrow}$ *rows Q_A, Q_B through the displayed windows W_A, W_B and the shared dyadic ledger D before using the common tail budget T. The handoff exports a finite comparison of regular rational tails; it does not export quotient stream equality, a selected limit, countable choice, trichotomy, or ambient real completeness.*

Proof. Start with the carrier definition. The two regular rational readbacks Q_A, Q_B are attached to the two finite window rows W_A, W_B, and both sides share the single dyadic tolerance ledger D.

The common tail-budget row T is displayed after those rows. Hence a sequence-limit or limsup consumer cannot inspect the comparison budget before the two regular rational tail presentations and their shared tolerance surface are available.

The remaining rows H, C, P, N transport, replay, package, and name this same finite route. They add no quotient stream, selected limit, choice operator, trichotomy decision, or ambient completeness coordinate. □

Theorem 3427.4 (Real uniform tail comparison real-seal boundary)**.** *Every real-facing read exported by an accepted* $\mathsf{RealUniformTailComparison}^{\uparrow}$ *packet factors through the displayed source rows, StreamName windows, RegSeqRat readbacks, shared DyadicRatCore ledger, common tail budget, and the two consumer rows L, Λ before any* $\mathsf{Real}^{\uparrow}$ *seal can be used downstream. The packet therefore supplies a boundary for tail comparison under a common tolerance budget, not a proof of selected convergence, quotient equality of streams, real trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The only real-facing rows named by the packet are reached through the displayed sequence-limit and limsup consumers. Those consumers are positioned after $S_A, S_B, W_A, W_B, Q_A, Q_B, D, T$ in the carrier.

Transport and continuation replay preserve that order componentwise. Package provenance and local naming can expose the route, but they cannot insert a hidden real seal or a selected representative before the finite comparison rows.

Thus every downstream $\mathsf{Real}^{\uparrow}$ read is bounded by the displayed tail-comparison route, and the stated nonclaims follow from the absence of the excluded coordinates in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformTailComparison}^{\uparrow}$ packet is generated from two real-sequence source rows, two StreamName finite-window rows, two RegSeqRat readbacks, one shared DyadicRatCore tolerance ledger, one common tail budget, RealSequenceLimit and Limsup consumer rows, componentwise transport, displayed continuation replay, package provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3427.1, Definition 3427.2, Definition 3427.3, and Definition 3427.4 seed the finite L10 tail-comparison route over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Limsup}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformTailComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient streams, real trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, uniform tail-budget exposure, RegSeqRat handoff, real-seal boundary, and consumer nonescape for BEDC.Derived.RealUniformTailComparisonUp.

Chapter 3428

A Concrete Naming Certificate for $\mathsf{CauchyChainCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyChainCriterion}^{\uparrow}$ packet records the finite criterion surface by which a Cauchy-space consumer reads a chain of adjacent rational approximations before reaching a Real endpoint. It sits between the $\mathsf{CauchySpace}^{\uparrow}$ interface of chapter 1847, the selected-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite chain criterion, not a quotient completion, selected limit, host equality, countable-choice schedule, or ambient completeness principle.

Definition 3428.1 (CauchyChainCriterion carrier)**.** *A* $\mathsf{CauchyChainCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchySpace}^{\uparrow}$ *route that admits the finite filter-uniform source read.*
2. *L is the finite chain-link ledger, listing adjacent approximation links and their requested tolerances.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family from which each link endpoint is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and adjacent links.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the finite sums and refinements needed along the chain.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, L, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the source, links, windows, readbacks, tolerance ledger, endpoint seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the Cauchy-space source through the chain links, selected windows, regular readbacks, dyadic tolerance ledger, and endpoint seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyChainCriterion}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, W, R, D, E, H, C, P, N$. It has no coordinate for quotient completion, selected representatives, host real equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyChainCriterionUp.CauchyChainCriterionUp`
Lean4 ✓ `BEDC.Derived.CauchyChainCriterionUp.CauchyChainCriterionTasteGate single carrier alignment`

Theorem 3428.2 (CauchyChainCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyChainCriterion}^{\uparrow}$ *carrier* $K = (S, L, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, adjacent-link coverage through* L*, tail-chain transitivity through the displayed dyadic ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through* R*, and non-escape for quotient completion, selected limits, host equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3428.1. Carrier admission exposes the Cauchy-space source S, the link ledger L, the selected windows W, the regular readback R, the tolerance ledger D, and the endpoint seal E as named finite rows.

Adjacent-link coverage is read from L and W: each link endpoint must be present in the selected window family before the corresponding regular rational readback may be inspected.

Tail-chain transitivity is the displayed use of D. The dyadic ledger records the finite tolerance sums and refinements connecting successive links, so a tail-chain consumer receives a finite ledger rather than an external convergence theorem.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is forced through R before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be read. The structural rows H, C, P, N transport, replay, source, and name the same route, and the omitted host-side objects have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyChainCriterion}^{\uparrow}$ packet is generated from a $\mathsf{CauchySpace}^{\uparrow}$ source route, finite adjacent-link ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3428.1 and Definition 3428.2 seed the finite Cauchy-space chain criterion from $\mathsf{CauchySpace}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyChainCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host real equality, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjacent-link coverage, tail-chain transitivity, regular-name handoff, endpoint-seal routing, and non-escape proofs for BEDC.Derived.CauchyChainCriterionUp.

Chapter 3429

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$

Following Riemann, Darboux, and Bishop–Bridges constructive analysis, the ClosedBoundedIntervalDarbouxPartition packet records the finite partition surface needed to run Darboux and Riemann integration over one displayed closed bounded interval. It binds the base interval of chapter 1591, the located partition row of chapter 2248, the finite Darboux-sum boundary of chapter 1880, the Darboux integration packet of chapter 1460, and the tagged-mesh consumer of chapter 1526. The packet supplies a closed-interval-specific Darboux partition certificate; it is not an open-cover compactness theorem, a selected tag family, a quotient function-equality principle, a measure-completion object, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 3429.1 (Closed bounded interval Darboux partition carrier)**.** *A* ClosedBoundedIntervalDarbouxPartition *carrier is a finite* BHist *packet*

$$K = (I, M, L, U, S, D, J, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row, carrying lower and upper located endpoints, endpoint order, rational and dyadic containment ledgers, finite windows, readbacks, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
2. *M is the* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row over the same interval endpoints, exposing the finite cell mesh, cell adjacency, dyadic refinement, rational-cell witnesses, and located endpoint readbacks;*
3. *L and U are lower and upper Darboux bracket rows read through* $\mathsf{DarbouxSum}^{\uparrow}$ *over the cells of M;*
4. *S is the shared Darboux upper-lower sum surface, recording the finite lower sum, finite upper sum, and the visible gap ledger;*
5. *D is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *handoff row attached to the same interval, mesh, lower bracket, upper bracket, and gap ledger;*
6. *J is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer handoff row, read only after the Darboux row D and the partition row M have been displayed;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback and seal row reached from the finite gap ledger;*

8. *H records componentwise* hsame *transport of the interval, partition, Darboux brackets, sum gap, Darboux handoff, Riemann handoff, and real-seal rows;*
9. *C records displayed* Cont *replay from interval endpoints to partition cells, from cells to Darboux sums, and from Darboux gap data to the integration handoff rows;*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected tag families, quotient function equality, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Its public object is the finite closed-interval partition route from I through $M, L, U, S, D, J, R, H, C, P, N$.*

```
Lean4 ✓  BEDC.Derived.ClosedBoundedIntervalDarbouxPartitionUp.TasteGate.ClosedBoundedIntervalDarbouxPart
single carrier alignment
```

Theorem 3429.2 (Closed bounded interval Darboux partition NameCert obligations)**.** *For the carrier of Definition 3429.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, M, L, U, S, D, J, R, H, C, P, N)$; source-interval exactness through I; located partition exactness through M; lower and upper Darboux bracket exposure through L, U; finite Darboux-sum exactness and gap visibility through S; Darboux handoff through D; Riemann handoff through J;* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *sealing through R; componentwise transport through H; continuation replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier. The interval row I fixes the lower and upper located endpoints, endpoint order, containment ledgers, finite readbacks, and terminal real seal before any partition row can be consumed.

Next read the located partition row M. Since M is attached to the same endpoint and containment data, every Darboux cell and refinement read remains a finite closed-interval read rather than a host interval partition.

The lower and upper bracket rows L, U and the sum row S are then inspected through the Darboux-sum surface. They record finite lower and upper sums over displayed cells, and the gap ledger is visible before either integration handoff row can be used.

Finally the handoff rows D, J, R and the structural rows H, C, P, N preserve and name exactly this route. None of these rows contains open-cover compactness, a selected tag family, quotient function equality, measure completion, or ambient real completeness, so the listed obligations exhaust the NameCert surface. □

Theorem 3429.3 (Closed bounded interval Darboux partition refinement stability)**.** *Let $K = (I, M, L, U, S, D, J, R, H, C, P, N)$ and $K' = (I', M', L', U', S', D', J', R', H', C', P', N')$ be accepted* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *carriers classified together by componentwise* hsame *transport. If M' is read as a displayed common refinement of M over the transported closed bounded interval row, then the transported lower bracket, upper bracket, finite sum gap, Darboux handoff, Riemann handoff, and* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal are all read through the same finite refinement route.*

The refinement therefore preserves the Darboux partition certificate without adding selected tags, quotient equality of integrands, measure completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source interval row. Componentwise transport carries I to I', including endpoint order, containment ledgers, finite windows, readbacks, and the real seal, before any refined cell can be read.

Read the common refinement through M'. Its cells are located relative to the transported endpoints and containment ledgers, so each refined cell points back to the same closed bounded interval route.

The Darboux bracket and sum rows transport next. The rows L', U', S' compute lower and upper sums only from the displayed refined cells and carried values, hence the visible gap is preserved as a finite Darboux comparison.

The integration handoff rows D', J', R' are downstream of that comparison. Since H' and C' replay the same interval-to-partition-to-gap route and P', N' only source and name it, refinement stability cannot introduce the excluded host-side objects. □

Theorem 3429.4 (Closed bounded interval Darboux partition Riemann handoff)**.** *Every accepted* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of a* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *packet receives the closed interval only through I, the located partition only through M, the Darboux lower and upper bracket rows only through L, U, S, D, and the real-facing seal only through R. The handoff row J may place a tagged-mesh consumer over that displayed partition route, but it does not supply an independent tag-selection family, quotient equality of integrands, measure completion, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 3429.2. It makes the interval, partition, bracket, sum, gap, Darboux handoff, Riemann handoff, seal, transport, replay, provenance, and local naming rows the entire public surface.

Apply Definition 3429.3. Any consumer-facing refinement of the partition must preserve the same endpoint, containment, cell, bracket, sum, and gap route before the handoff rows are read.

Now compare with the Riemann handoff boundary of Definition 1526.4. The Riemann consumer may add its own tag-containment and finite-sum rows, but those rows sit downstream of the displayed closed-interval Darboux partition route.

Thus J is a consumer handoff for the finite route already named by $I, M, L, U, S, D, R, H, C, P, N$. It cannot export an independent selected tag family, quotient integrand equality, measure-completion object, open-cover compactness proof, or ambient real-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ packet is generated from a closed bounded interval row, a located real interval partition row, lower and upper Darboux bracket rows, a finite Darboux-sum gap ledger, Darboux and Riemann integration handoff rows, a RegSeqRat/Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3429.1, Definition 3429.2, Definition 3429.3, and Definition 3429.4 seed the closed bounded interval Darboux partition boundary over $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected tag families, quotient function equality, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located partition exactness, Darboux bracket exposure, finite gap preservation, Riemann handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.ClosedBoundedInterv

Chapter 3430

A Concrete Naming Certificate for $\mathsf{FiniteCoverRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet records the constructive finite refinement step used by compact metric and completion-facing consumers when two displayed finite covers must be compared. It sits over the compact metric finite-net surface of chapter 129, the dyadic common-refinement ledger of chapter 1877, and the finite real ball-cover surface of chapter 2377. The packet names only a finite refinement route through displayed cover rows; it is not open-cover compactness, not a selected Lebesgue number, and not countable choice.

Definition 3430.1 (Finite cover refinement carrier)**.** *A* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, D, F, Q, U, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite-net and metric-distance data bound the cover comparison.*
2. *D is the* $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *common-refinement ledger comparing the two dyadic cover presentations seen by the consumer.*
3. *F is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row naming the finite real-ball cover, its coverage witnesses, and its refinement ledger.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *tolerance row used to read radii and overlap bounds.*
5. *U is the finite* $\mathsf{ProbeBundle}$ *row listing the inspected centers, witnesses, and retained members.*
6. *W is the refinement witness row, pairing each retained probe with a member of the common finite refinement and recording the displayed inclusion or radius comparison.*
7. *H is the componentwise* hsame *transport row for the compact metric, dyadic refinement, finite real cover, rational tolerance, probe bundle, witness, provenance, and local naming coordinates.*
8. *C is the displayed* Cont *replay route for compact-metric reads, dyadic cover reads, finite real-cover reads, rational-radius reads, and refinement-witness reads.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicCoverRefinement}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$ *evidence row.*

The carrier has no coordinate for an arbitrary open cover, a host finite-subcover selector, a selected Lebesgue number, a countable-choice sequence, quotient real equality, or ambient compactness.

Theorem 3430.2 (Finite cover refinement NameCert obligations)**.** *For the carrier of Definition 3430.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (K, D, F, Q, U, W, H, C, P, N)$*, compact-metric source exposure through* K*, dyadic common-refinement exposure through* D*, finite real-cover exposure through* F*, rational tolerance exposure through* Q*, finite probe-bundle exposure through* U*, refinement-witness exposure through* W*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every compact-metric, finite-real-cover, or completion-facing consumer of this packet must therefore read the displayed route

$$K \longmapsto D \longmapsto F \longmapsto Q \longmapsto U \longmapsto W \longmapsto (H, C, P, N)$$

before it can compare the two finite covers. The route exports no open-cover compactness theorem, no selected Lebesgue number, no countable-choice selector, no quotient representative, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier tuple. The rows K, D, F, Q, U, W are the only analytic coordinates, and the rows H, C, P, N transport, replay, source, and name that same finite packet.

Read the dependency chain in order. The compact-metric row supplies the finite source data, the dyadic refinement row supplies the common-refinement ledger, the finite real-cover row supplies coverage witnesses, and the rational and probe rows expose the finite tolerance window in which W is meaningful.

The structural rows create no additional compactness principle. Since the carrier contains no arbitrary open cover, selected Lebesgue number, countable-choice sequence, quotient representative, or ambient completeness row, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.FiniteCoverRefinementUp.FiniteCoverRefinementTasteGate single carrier alignment`

Theorem 3430.3 (Finite cover refinement compact handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{FiniteRealCover}^{\uparrow}$*, or regular-completion consumer citing an accepted* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *packet receives only the finite comparison route displayed in Definition 3430.2. In particular, the consumer may compare retained cover members and dyadic radius rows through* D, F, Q, U, W*, but it cannot replace that comparison by open-cover compactness, a selected Lebesgue number, countable choice, a host subcover selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligation theorem. It places every cover comparison on the displayed route from the compact metric source through the dyadic and finite real-cover rows.

The witness row W is meaningful only after Q and U have exposed the finite tolerance and probe rows. Componentwise transport and continuation replay preserve those coordinates without adding a second comparison route.

The excluded objects are absent from the carrier, provenance, and local naming rows. Hence the compact-facing handoff is exactly the finite refinement packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a DyadicCoverRefinementUp common-refinement ledger, a FiniteRealCoverUp coverage row, RatUp tolerance rows, finite ProbeBundle rows, a refinement witness, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 3430.1, Definition 3430.2, and Definition 3430.3 define the finite refinement packet and its compact-facing handoff over displayed finite cover rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteCoverRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected Lebesgue numbers, countable choice, host subcover selection, quotient real equality, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement witness exactness, finite-cover transport, compact-facing handoff, and ledger refusal for BEDC.Derived.FiniteCoverRefinementUp.

Chapter 3431

A Concrete Naming Certificate for $\mathsf{LocatedIntervalCompactness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ packet records the finite-cover compactness step available for a displayed located real interval when the cover has already been reduced to explicit dyadic mesh data. It sits over the located endpoint surface of chapter 394, the dyadic rational arithmetic of chapter 291, the terminal real-name rows of chapter 38, the finite-net compact metric surface of chapter 129, and the finite refinement route of chapter 3430. The packet names only endpoint rows, mesh cells, finite cover rows, refinement witnesses, compact-metric handoff, transport, replay, provenance, and local naming evidence; it is not a general Heine–Borel theorem and does not import arbitrary open-cover compactness.

Definition 3431.1 (Located interval compactness carrier)**.** *A* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, E, M, F, W, K, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing the lower and upper endpoint names, rational cell ledger, dyadic endpoint refinements, finite endpoint windows, and the downstream* $\mathsf{Real}^{\uparrow}$ *seals.*
2. *E is the endpoint exposure row that records the two endpoint rows as the boundary read consumed by the finite mesh.*
3. *M is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, listing a finite dyadic subdivision with the endpoint cells and mesh-step witnesses used by the interval packet.*
4. *F is the finite cover row, containing only displayed finite cover members indexed by the mesh cells.*
5. *W is the refinement witness row, pairing each mesh cell with a retained finite cover member and the displayed inclusion or overlap comparison.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row read after the mesh and finite cover rows have been displayed.*
7. *H records componentwise* hsame *transport for the located interval, endpoint exposure, mesh, finite cover, refinement witness, compact-metric handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from endpoint inspection through dyadic mesh construction, finite cover lookup, refinement witness replay, compact-metric handoff, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary open cover, selected subcover functional, sequential compactness principle, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalCompactnessUp.LocatedIntervalCompactnessTasteGate single carrier alignment`

Theorem 3431.2 (Located interval compactness carrier exposure)**.** *For every accepted* LocatedIntervalCompactness *packet*

$$I = (L, E, M, F, W, K, H, C, P, N),$$

the local certificate exposes exactly the located endpoint rows, the dyadic mesh row, the finite cover row, the refinement witness, the compact-metric handoff, componentwise transport, displayed replay, provenance, and local NameCert rows. Every finite-cover compactness read factors through

$$L \longmapsto E \longmapsto M \longmapsto F \longmapsto W \longmapsto K \longmapsto (H, C, P, N)$$

before any compact-metric or completion-facing consumer can use the packet. The exported route contains no general open-cover compactness theorem, no sequential compactness principle, no choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness field.*

Proof. Project the carrier of Definition 3431.1. The interval source is the row L, and the only endpoint-facing data available to the compactness route are the displayed endpoint rows named by E.

The finite construction then proceeds by dyadic mesh induction over M. The base cell reads the endpoint exposure row and the corresponding finite cover entry in F. The induction step appends the next displayed dyadic cell, reads its cover member from the same finite cover row, and records the retained overlap or inclusion comparison in W.

Apply the finite refinement route of chapter 3430. The witness row W is the only refinement surface exported to the compact-metric handoff K, so a compact-facing consumer receives a finite mesh-and-cover comparison rather than an arbitrary open cover.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for a selected subcover functional, sequential compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be exported by the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ packet is generated from a LocatedIntervalUp endpoint row, an endpoint exposure row, a finite DyadicRatCoreUp mesh, finite cover rows, refinement witnesses, a CompactMetricUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 3431.1 and Definition 3431.2 seed the finite located-interval compactness surface over endpoint rows, dyadic mesh rows, finite cover rows, refinement witnesses, compact-metric handoff, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, arbitrary open-cover compactness, sequential compactness, choice, quotient real equality, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh induction, finite cover refinement, compact-metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalCompactnessUp.

Chapter 3432

A Concrete Naming Certificate for $\mathsf{LimitComparisonTest}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Hardy's classical comparison-test presentation, the $\mathsf{LimitComparisonTest}^{\uparrow}$ packet records the finite route by which a source sequence is compared to a positive comparator sequence before a real-facing conclusion is exposed. It is the sequence-level sibling of chapter 2023: $\mathsf{SeriesLimitComparison}^{\uparrow}$ consumes the resulting comparison surface for series tails, while $\mathsf{LimitComparisonTest}^{\uparrow}$ names the underlying ratio-test carrier over $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet exposes finite ratio-control data; it does not import a host quotient limit, quotient stream equality, real trichotomy, countable choice, or ambient completeness.

Definition 3432.1 (Limit comparison test carrier)**.** *A* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the inspected finite windows of the sequence being tested.*
2. *T is the comparator* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the matched finite windows of the positive comparator sequence.*
3. *W_S and W_T are the source and comparator observation schedules used by downstream sequence and series consumers.*
4. *Q_S and Q_T are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the displayed source and comparator windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the ratio slack, positivity margin, comparison radius, and endpoint tolerance consumed by the test.*
6. *R is the eventual positive ratio ledger. It displays a finite threshold after which comparator observations are positive and source-to-comparator comparisons remain inside the dyadic band recorded in D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source windows, comparator windows, regular readbacks, dyadic ledger, and ratio ledger have been displayed.*

8. *H stores componentwise* hsame *transport for the source windows, comparator windows, readback rows, dyadic ledger, ratio ledger, real seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from a limit-comparison request through $S, T, W_S, W_T, Q_S, Q_T, D, R$ and then to the real-facing seal E.*
10. *P is the* Pkg *provenance over* $\mathsf{LimitComparisonTest}^{\uparrow}$, $\mathsf{SeriesLimitComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N$. It has no coordinate for a host quotient limit, quotient stream equality, a selected infinite tail, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 3432.2 (Limit comparison test NameCert obligations)**.** *For every accepted* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source windows, comparator windows, readback rows, dyadic tolerance ledger, eventual positive ratio ledger, real seal, transport, replay, provenance, and local naming rows;*
2. *source-window exactness, requiring every inspected source observation to pass through S, W_S, Q_S, D before the ratio ledger is read;*
3. *comparator-window exactness, requiring every inspected comparator observation to pass through T, W_T, Q_T, D before positivity or ratio control is used;*
4. *eventual positivity, requiring R to display the finite threshold and positive comparator margin used by all later comparisons;*
5. *ratio-band exactness, requiring R and D to display the dyadic lower and upper comparison bounds before any real-facing seal is consumed;*
6. *real-seal nonescape, requiring E to be downstream of the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and ratio-ledger rows and to export no host quotient limit, quotient stream equality, selected tail, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge datum;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same finite route through H, C, P, N.*

No limit-comparison-test obligation is stored outside these rows.

Proof. Project the accepted packet $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$. The carrier definition exposes the source and comparator windows, their regular readbacks, the shared dyadic ledger, the eventual positive ratio ledger, and the structural rows.

The source side and comparator side are read separately. Source observations must pass through S, W_S, Q_S, D, while comparator observations must pass through T, W_T, Q_T, D.

The ratio ledger R then supplies the finite positivity threshold and dyadic comparison band. Since these are displayed rows, a consumer cannot replace them by a host quotient limit or by an undisplayed infinite tail.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Because no excluded analytic or choice object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.LimitComparisonTestUp.LimitComparisonTestTasteGate single carrier alignment`

Theorem 3432.3 (Limit comparison test Cauchy handoff)**.** *Let* $L = (S, T, W_S, W_T, Q_S, Q_T, D, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LimitComparisonTest}^{\uparrow}$ *carrier. Every Cauchy-facing or series-facing read of the source route factors through*

$$T, W_T, Q_T, D \longmapsto R \longmapsto S, W_S, Q_S, D \longmapsto E, H, C, P, N.$$

Thus comparator control is handed to the source route only through the displayed eventual positive ratio ledger and dyadic tolerance budget. The handoff exports finite BEDC comparison data, not a host quotient limit, quotient stream equality, selected tail, countable choice, real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Cauchy-facing or series-facing request for the source route. The comparator data available to the carrier are exactly T, W_T, Q_T, D, so the request first reads the displayed comparator windows, readback row, and tolerance ledger.

The ledger R supplies the finite threshold, positivity margin, and dyadic ratio band that connect later comparator observations to later source observations. This is a packet row rather than a host convergence assertion.

The request then returns to the source rows S, W_S, Q_S, D. These rows expose the finite source windows and regular rational readbacks that a downstream $\mathsf{SeriesLimitComparison}^{\uparrow}$ or Cauchy consumer may inspect.

The real seal and structural rows E, H, C, P, N preserve that finite route. Since the carrier has no coordinate for quotient limits, quotient streams, choice, trichotomy, or ambient completeness, every such handoff is exhausted by the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$.

Constructive story: A $\mathsf{LimitComparisonTest}^{\uparrow}$ packet is generated from source and comparator StreamName windows, RegSeqRat readback rows, a DyadicRatCore tolerance ledger, an eventual positive ratio ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$, seed)

Scope closed: Definition 3432.1, Definition 3432.2, and Definition 3432.3 seed the finite constructive limit-comparison route from positive comparator windows through ratio budgeting to source-route RealUp sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LimitComparisonTest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient limits, quotient stream equality, selected tails, countable choice, real trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and comparator window exactness, eventual positive ratio-ledger stability, DyadicRatCore and RegSeqRat budget handoff, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.LimitComparisonTestUp.

Chapter 3433

A Concrete Naming Certificate for $\mathsf{DenjoyIntegral}^{\uparrow}$

Following Denjoy, Henstock, Kurzweil, and Bishop–Bridges, the $\mathsf{DenjoyIntegral}^{\uparrow}$ packet records the constructive integration neighbour of $\mathsf{GaugeIntegral}^{\uparrow}$ that keeps exceptional variation data visible before any real-valued integral read is exported. It sits beside the gauge-partition route of chapter 1999, the public integral row of chapter 123, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Denjoy-style handoff for one displayed integrand window; it is not measure completion, a quotient of functions, a host almost-everywhere equality theorem, a selected primitive, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3433.1 (Denjoy integral carrier)**.** *A* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (G, I, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *source row, read through the finite gauge-partition route of chapter 1999.*
2. *I is the public* $\mathsf{Integral}^{\uparrow}$ *row for the same displayed integrand and interval window.*
3. *V is the exceptional-variation ledger, recording the finite variation exclusions, controlled subintervals, and refusal rows that prevent the handoff from becoming a hidden measure-completion argument.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational integral approximants and tolerance requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, I, V, R have been displayed.*
6. *H records componentwise* hsame *transport for the gauge source, integral row, exceptional-variation ledger, regular readback, real seal, continuation replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the same route.*
8. *P is the* Pkg *provenance over* $\mathsf{DenjoyIntegral}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$ *row.*

The carrier has no coordinate for measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.DenjoyIntegralUp.DenjoyIntegralUp`

Theorem 3433.2 (Denjoy integral NameCert obligations)**.** *For every accepted* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier* $D = (G, I, V, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed gauge-integral source, public integral row, exceptional-variation ledger, RegSeqRat readback, Real seal, hsame transport, Cont replay, provenance, and local naming;*
2. *gauge compatibility, requiring every partition-facing request to pass through* G *and the sibling route of chapter 1999;*
3. *public integral visibility, requiring the exported integral read to pass through* I *before any regular-rational or Real-facing row is consumed;*
4. *exceptional-variation accountability, requiring every exceptional subwindow or variation refusal to be displayed in* V *rather than hidden in a measure-completion or almost-everywhere premise;*
5. *readback exactness, requiring rational approximants and tolerance requests to pass through* R *before the* $\mathsf{Real}^{\uparrow}$ *seal* E *is available;*
6. *structural stability, requiring* H, C, P, N *to transport, replay, source, and name only the rows already listed.*

No Denjoy-integral obligation is stored outside D*.*

Proof. Project the carrier of Definition 3433.1. The displayed packet exposes exactly the gauge source, public integral row, exceptional-variation ledger, regular readback, real seal, and structural rows.

Gauge compatibility is the row G. It routes every partition-facing read through the same finite gauge data used by chapter 1999, so a consumer cannot replace the packet by a selected family of gauges or by an ambient compactness theorem.

Integral visibility and exceptional-variation accountability are the rows I and V. The public integral row records the exported integral surface, while V keeps exceptional variation and refusal data as displayed finite ledger entries.

The regular readback and real seal are downstream of those rows. The row R supplies rational approximants and tolerance requests before E is exposed, and H, C, P, N preserve only that route. Since the carrier contains no measure-completion, quotient-function, host almost-everywhere, selected-primitive, choice, or completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3433.3 (Denjoy integral gauge handoff)**.** *Every* $\mathsf{Integral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyIntegral}^{\uparrow}$ *carrier* $D = (G, I, V, R, E, H, C, P, N)$ *must read the finite route*

$$G \longmapsto I \longmapsto V \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The handoff therefore exposes the gauge-integral source, public integral row, exceptional-variation ledger, regular rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming before any Denjoy integral value is consumed.

Proof. Start with an integral-facing, regular-rational-facing, or Real-facing read of D. By Definition 3433.1, the only gauge source is G and the only public integral coordinate is I.

The exceptional side is forced through V. A consumer must inspect the displayed variation ledger and refusal rows before the handoff reaches rational approximants or a real-name seal.

The route into the real-facing value is $R \longmapsto E$. The regular rational readback records the finite approximants and tolerance requests, and the terminal seal can be replayed only through H and C with provenance P and local name N.

Consequently the Denjoy packet cannot export a value by bypassing $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$, and it cannot import measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, or ambient real completeness as hidden data. □

Falsifiable prediction: Every $\mathsf{DenjoyIntegral}^{\uparrow}$ integral-facing or Real-facing read factors through displayed GaugeIntegral, Integral, exceptional-variation ledger, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: gauge-integral, integral, regseqrat, real: $\mathsf{DenjoyIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a gauge source, public integral row, exceptional-variation ledger, regular rational approximants, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3433.1, Definition 3433.2, and Definition 3433.3 seed the finite Denjoy-integration route through $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected primitives, host almost-everywhere equality, countable choice, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge compatibility, public integral visibility, exceptional-variation accountability, regular rational readback exactness, Real seal handoff, transport stability, and replay stability for BEDC.Derived.DenjoyIntegralUp.

Constructive story: A $\mathsf{DenjoyIntegral}^{\uparrow}$ packet is generated from a displayed GaugeIntegral source, public Integral row, exceptional-variation ledger, RegSeqRat readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3434

A Concrete Naming Certificate for DecidableBetaFrontier$^{\uparrow}$

The DecidableBetaFrontier$^{\uparrow}$ packet realizes the bounded MetaCIC beta-audit window named in chapter 9386 as finite NameCert data. It sits beside the literal lambda-calculus substrate of chapter 200, the beta-redex frontier of chapter 1008, the decidability audit packet of chapter 746, and the GroundCompiler audit map of chapter 836. The packet records a bounded audit frontier for beta checking; it does not claim full normalization, arbitrary closed beta-conversion decidability, closed subject reduction, or global Church–Rosser.

Definition 3434.1 (Decidable beta frontier carrier)**.** *A* DecidableBetaFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, L, W, Q, A, G, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the MetaCIC audit-source row, reading the open-problem and decidability boundaries of chapter 9386.*
2. *L is the* LambdaCalc$^{\uparrow}$ *syntax and beta-step row, anchored in chapter 200.*
3. *W is the bounded beta-frontier window: it fixes the finite depth, fuel, closedness observations, and normal-output comparisons that the packet may inspect.*
4. *Q is the bounded checker row, read through the finite decidability surface of chapter 746.*
5. *A is the audit refusal row, recording why the bounded window is not a proof of full normalization or unrestricted beta-conversion decidability.*
6. *G is the GroundCompiler provenance row, read through chapter 836 only as compiler-layer audit data.*
7. *R is the redex-frontier handoff row, read through chapter 1008.*
8. *H records componentwise* hsame *transport for the audit source, lambda syntax row, bounded window, checker row, refusal row, compiler provenance, frontier handoff, continuation replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the bounded audit route.*
10. *P is the* Pkg *provenance over* DecidableBetaFrontier$^{\uparrow}$*, MetaCIC audit rows,* LambdaCalc$^{\uparrow}$*, bounded checker rows, GroundCompiler audit rows, beta-frontier rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DecidableBetaFrontier}^{\uparrow}}$ *row.*

The carrier has no coordinate for full normalization, arbitrary closed conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, host evaluator equality, or hidden compiler evidence.

Theorem 3434.2 (Decidable beta frontier NameCert obligations)**.** *For every accepted* $\mathsf{DecidableBetaFrontier}^{\uparrow}$ *carrier* $B = (M, L, W, Q, A, G, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DecidableBetaFrontier}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the MetaCIC audit source, LambdaCalc syntax row, bounded beta window, checker row, refusal row, GroundCompiler provenance, redex-frontier handoff, transport, replay, provenance, and local naming;*
2. *bounded-window exactness, requiring every beta-check request to name the finite depth, fuel, closedness observations, and normal-output comparisons in W;*
3. *checker accountability, requiring Q to expose only bounded checker facts already visible at the decidability audit surface;*
4. *frontier refusal, requiring A to record the non-claim against full normalization, unrestricted conversion decidability, subject reduction, and global confluence;*
5. *compiler-layer provenance, requiring G to be read only as a GroundCompiler audit row and not as hidden semantic evidence;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No decidable beta frontier obligation is stored outside B.

Proof. Project the carrier of Definition 3434.1. The displayed packet exposes the MetaCIC audit source, lambda syntax row, bounded window, checker row, refusal row, compiler provenance, redex-frontier handoff, and structural rows.

Bounded-window exactness is the row W. A beta-check request cannot enter the packet unless its finite depth, fuel, closedness observations, and normal-output comparisons have been fixed as displayed data.

Checker accountability is the row Q, and frontier refusal is the row A. The checker row may report bounded facts, while the refusal row keeps full normalization, unrestricted conversion decidability, subject reduction, and global confluence outside the carrier.

The compiler and redex rows are G and R. They provide audit provenance and frontier routing, not semantic shortcuts. Since H, C, P, N only transport, replay, source, and name displayed rows, the local obligations are exhausted by the finite packet. □

Theorem 3434.3 (Decidable beta frontier MetaCIC handoff)**.** *Every MetaCIC,* $\mathsf{LambdaCalc}^{\uparrow}$*, or GroundCompiler-facing consumer of an accepted* $\mathsf{DecidableBetaFrontier}^{\uparrow}$ *carrier* $B = (M, L, W, Q, A, G, R, H, C, P,$ *must read the bounded route*

$$M, L \longmapsto W \longmapsto Q \longmapsto A \longmapsto G, R \longmapsto H, C, P, N.$$

The handoff exposes the MetaCIC audit source, LambdaCalc syntax, bounded beta window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, transport, replay, provenance, and local naming before any beta-frontier conclusion is consumed.

Proof. Start with a MetaCIC, LambdaCalc, or GroundCompiler-facing read of B. The carrier gives the audit source M and lambda syntax row L as the only source coordinates.

The finite audit boundary is $W \longmapsto Q$. The bounded window fixes the observed beta frontier, and the checker row reads only that finite window.

The refusal and provenance rows are then forced into the route. The row A records the non-claims, while G and R expose GroundCompiler audit provenance and redex-frontier handoff without adding hidden semantic evidence.

Finally H, C, P, N replay and name the same packet. Therefore no consumer can turn the bounded frontier into full normalization, unrestricted beta-conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, host evaluator equality, or a compiler fact absent from the displayed rows. □

Theorem 3434.4 (Decidable beta frontier TasteGate boundary)**.** *The Lean-facing* TasteGate.lean *site for* DecidableBetaFrontier$^{\uparrow}$ *must provide explicit* Nontrivial DecidableBetaFrontier$^{\uparrow}$ *and* FieldFaithful DecidableBet *instances for the displayed rows* $M, L, W, Q, A, G, R, H, C, P, N$*. These instances preserve the distinction between bounded checker evidence, redex-frontier handoff, refusal ledger, and compiler-layer provenance before any checked carrier-alignment target may cite* BEDC.Derived.DecidableBetaFrontierUp*.*

Lean4 ✓ `BEDC.Derived.DecidableBetaFrontierUp.DecidableBetaFrontierTasteGate single carrier alignment`

Proof. Read the carrier rows of Definition 3434.1. The bounded window, checker row, refusal ledger, compiler provenance, and redex-frontier handoff have different consumer roles and cannot be collapsed without changing the NameCert surface.

Nontriviality rules out the degenerate packet in which the frontier has no distinguishable bounded audit row. Field faithfulness requires each displayed coordinate $M, L, W, Q, A, G, R, H, C, P, N$ to survive transport as the same named row.

Thus the later Lean TasteGate site must keep those rows separate before it records a carrier-alignment theorem for the packet. □

Falsifiable prediction: Every DecidableBetaFrontier$^{\uparrow}$ beta-frontier read factors through displayed MetaCIC audit source, LambdaCalc syntax, bounded window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: MetaCIC, LambdaCalc, GroundCompiler: DecidableBetaFrontier$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded beta-window data, checker accountability, refusal rows, compiler-layer provenance, redex-frontier handoff, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 3434.1, Definition 3434.2, Definition 3434.3, and Definition 3434.4 seed the finite bounded beta-frontier route over MetaCIC, LambdaCalc, GroundCompiler, and beta-audit rows.

Formal verification: FormalStatus(NameCert$_{\text{DecidableBetaFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full normalization, unrestricted closed beta-conversion decidability, closed subject reduction, residual-substitution compatibility, global confluence, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window exactness, checker accountability, frontier refusal, compiler-layer provenance, redex-frontier handoff, TasteGate nontriviality, and field faithfulness for BEDC.Derived.DecidableBetaFrontierUp.

Constructive story: A DecidableBetaFrontier$^{\uparrow}$ packet is generated from a MetaCIC audit source, LambdaCalc syntax row, bounded beta window, checker row, refusal ledger, GroundCompiler provenance, redex-frontier handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3435

A Concrete Naming Certificate for $\mathsf{HardyInequality}^{\uparrow}$

Following Hardy, Littlewood, and Polya on Hardy's classical inequality, the $\mathsf{HardyInequality}^{\uparrow}$ packet records the finite weighted-average surface needed by BEDC real-series consumers. It sits between the $\mathsf{RealSeries}^{\uparrow}$ summation interface of chapter 1581, the positive rational weight surface of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names Hardy-style partial-sum control as finite RealSeries data; it does not import host analytic inequalities, choose infinite sums, assume monotone convergence, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3435.1 (Hardy inequality carrier)**.** *A* $\mathsf{HardyInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, W, S, Q, D, U, E, T, J, K, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row. It stores the finite real-series prefix whose Hardy averages are inspected.*
2. *W is the* $\mathsf{Rat}^{\uparrow}$ *positive-weight row. Each inspected denominator and coefficient is displayed as a rational-positive entry, not as a host real scalar.*
3. *S is the partial-sum ledger. Its rows name the inspected prefix sums and weighted averages over the same finite windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected averages.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every comparison between a weighted average and its controlling series prefix.*
6. *U is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule. It records which finite prefix, denominator, and tolerance request are open at each read.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the RealSeries source, rational weights, partial sums, readback, dyadic tolerance, and window schedule have all been displayed.*
8. *T records componentwise* hsame *transport for A, W, S, Q, D, U, E.*
9. *J records displayed* Cont *replay from a Hardy comparison request through the window schedule, partial-sum ledger, readback row, dyadic tolerance ledger, and Real seal.*

10. *K is the* Pkg *provenance over* $\mathsf{HardyInequality}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *P records the finite boundary predicate saying that each displayed average reads only entries already present in the source prefix and positive-weight row.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an infinite summation operator, a choice of limiting representative, a host analytic norm, countable choice, quotient equality of streams, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.HardyInequalityUp.TasteGate.HardyInequalityTasteGate single carrier alignment`

Theorem 3435.2 (Hardy inequality NameCert obligations)**.** *For every accepted* $\mathsf{HardyInequality}^{\uparrow}$ *carrier* $H = (A, W, S, Q, D, U, E, T, J, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure through A, positive* $\mathsf{Rat}^{\uparrow}$ *weight exposure through W, finite partial-sum exposure through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure through D,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through U, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through T, displayed replay through J, package provenance through K, boundary-predicate exposure through P, and local naming through N.*

These obligations bind the finite Hardy route

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

without adding host analytic inequalities, infinite sums, selected limits, countable choice, quotient stream equality, or ambient real completeness.

Proof. Project the carrier of Definition 3435.1. Every accepted read begins with the displayed RealSeries source row and the positive rational weight row, so the weighted averages are finite BEDC rows rather than host-level analytic expressions.

The partial-sum ledger and window schedule are read next. The row S lists only prefix sums and weighted averages whose indices are opened by U, and the boundary row P requires each such average to use entries already present in A and weights already present in W.

Readback and tolerance data then enter through Q and D. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies regular rational approximants for the inspected averages, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the finite tolerance ledger for each comparison.

Finally E, T, J, K, N seal, transport, replay, source, and name the same ordered route. Since the excluded objects are absent from the carrier, no additional Hardy obligation can be exported by the local NameCert surface. □

Theorem 3435.3 (Hardy inequality partial-sum boundary)**.** *Let $H = (A, W, S, Q, D, U, E, T, J, K, P, N)$ be an accepted* $\mathsf{HardyInequality}^{\uparrow}$ *packet. Every Hardy average or partial-sum comparison exported by H factors through a displayed* $\mathsf{StreamName}^{\uparrow}$ *window in U, a finite* $\mathsf{RealSeries}^{\uparrow}$ *prefix in A, a positive* $\mathsf{Rat}^{\uparrow}$ *weight row in W, the partial-sum ledger S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before any* $\mathsf{Real}^{\uparrow}$ *seal E is available downstream.*

Consequently the packet records Hardy's classical inequality only as finite weighted-prefix control. It does not assert convergence of the whole series, import an external L^p inequality, choose a limiting representative, quotient streams, invoke countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the window row U. A Hardy request is accepted only after it names a finite prefix, denominator, and tolerance request, so no read can inspect a term outside the displayed window.

The source and weight rows supply the numeric data for that request. The $\mathsf{RealSeries}^{\uparrow}$ row A gives the inspected prefix, while the $\mathsf{Rat}^{\uparrow}$ row W gives the positive rational weights used by the displayed average.

The partial-sum ledger S and readback row Q then expose the finite average as regular rational data. The dyadic ledger D is the only tolerance surface available for comparing that readback with the controlling prefix.

The Real seal E is downstream from all of these rows, and T, J, K, N only transport, replay, source, and name the same route. Thus every exported comparison is bounded by the finite prefix data, and the excluded analytic and completeness principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{HardyInequality}^{\uparrow}$ read exposes a $\mathsf{RealSeries}^{\uparrow}$ prefix, positive $\mathsf{Rat}^{\uparrow}$ weights, a finite partial-sum ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise transport, displayed continuation replay, package provenance, boundary-predicate exposure, and local $\mathsf{NameCert}$ before any Hardy comparison is exported.

Independence witness: realseries, rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{HardyInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a weighted partial-sum ledger and a boundary predicate tying Hardy averages to finite RealSeries prefixes and positive rational weights.

Closure status for $\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{HardyInequality}^{\uparrow}$ packet is generated from a RealSeries source prefix, positive Rat weights, a finite partial-sum ledger, RegSeqRat readback, a DyadicRatCore tolerance ledger, StreamName windows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, a finite boundary predicate, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3435.1, Definition 3435.2, and Definition 3435.3 seed the finite Hardy weighted-prefix route over displayed $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HardyInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic Hardy inequalities, convergence of infinite sums, selected limits, quotient stream equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, positive-weight exposure, partial-sum boundary exposure, readback handoff, tolerance ledger nonescape, Real seal handoff, and transport stability for BEDC.Derived.HardyInequalityUp.

Chapter 3436

A Concrete Naming Certificate for $\mathsf{CommutativeCStar}^{\uparrow}$

Following Gelfand–Naimark and the Gelfand duality theorem, the $\mathsf{CommutativeCStar}^{\uparrow}$ packet records the finite algebraic source boundary by which a unital commutative C*-algebra can be handed to $\mathsf{GelfandDuality}^{\uparrow}$ before any completed spectrum, maximal ideal space, or host Gelfand transform is public. It refines the root $\mathsf{CStarAlg}^{\uparrow}$ surface of chapter 211 by adding unit and commutativity rows over the same $\mathsf{Banach}^{\uparrow}$ and $\mathsf{Ring}^{\uparrow}$ dependency rows, and it keeps the spectrum route through $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as displayed downstream rows rather than as an imported duality theorem.

Definition 3436.1 (Commutative CStar carrier)**.** *A* $\mathsf{CommutativeCStar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, B, R, I, M, E, Q, L, X, T, Y, H, C, P, N)$$

with the following displayed rows:

1. *A is the root* $\mathsf{CStarAlg}^{\uparrow}$ *row, carrying the Banach source, RingUp multiplication source, involution row, norm-square row, transport, replay, provenance, and local naming already required by Definition 211.3.*
2. *B is the consumed* $\mathsf{Banach}^{\uparrow}$ *row read from the same source history as A.*
3. *R is the consumed* $\mathsf{Ring}^{\uparrow}$ *row read from the same source history as A.*
4. *I is the displayed unit row for the multiplication surface of R.*
5. *M is the commutativity classifier row, recording that every displayed multiplication comparison used by the packet is symmetric over the same finite carrier.*
6. *E is the involution compatibility row, requiring the displayed involution of A to preserve the unit and commute with the displayed multiplication rows.*
7. *Q is the norm-square row, preserving the C^* identity $\|a^*a\| = \|a\|^2$ on the displayed algebra rows.*
8. *L is the character-facing ledger. It lists the finite algebraic rows that a later character or evaluation consumer may inspect, but it does not supply characters, a spectrum, or a transform.*
9. *X is the spectrum-facing* $\mathsf{GelfandDuality}^{\uparrow}$ *request row, recording that characters and evaluations must be supplied by the downstream spectrum packet.*

10. *T is the* $\mathsf{Compact}^{\uparrow}$ *and* $\mathsf{Topology}^{\uparrow}$ *boundary row for the displayed spectrum carrier.*
11. *Y is the* $\mathsf{Real}^{\uparrow}$ *scalar-readback row through which norm and evaluation values are sealed before topology-facing reads consume them.*
12. *H records componentwise* hsame *transport for $A, B, R, I, M, E, Q, L, X, T, Y$ and the structural rows.*
13. *C records displayed* Cont *replay of the source route*

$$\mathsf{CStarAlg}^{\uparrow}, \mathsf{Banach}^{\uparrow}, \mathsf{Ring}^{\uparrow} \longmapsto I, M, E, Q, L \longmapsto X, T, Y \longmapsto \mathsf{CommutativeCStar}^{\uparrow}.$$

14. *P is the* Pkg *provenance over* $\mathsf{CommutativeCStar}^{\uparrow}$, $\mathsf{CStarAlg}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{GelfandDuality}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through the root C-algebra row, the shared Banach and RingUp dependency rows, the unit row, the commutativity classifier, involution compatibility, norm-square compatibility, the character-facing ledger, the downstream spectrum request, the compact-topology boundary, the Real scalar seal, transport, replay, provenance, and local naming. It has no coordinate for a completed spectrum, maximal ideal space, host Gelfand transform, host algebra equality, quotient completion, selected character, or choice.*

Lean4 ✓ `BEDC.Derived.CommutativeCStarUp.TasteGate.CommutativeCStarTasteGate single carrier alignment`

Theorem 3436.2 (Commutative CStar NameCert obligations)**.** *For every accepted* $\mathsf{CommutativeCStar}^{\uparrow}$ *carrier*

$$U = (A, B, R, I, M, E, Q, L, X, T, Y, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$ *surface consists exactly of carrier admission, root CStarAlg source compatibility, Banach and RingUp source agreement, unit visibility, commutativity classifier stability, involution compatibility, norm-square compatibility, character-facing ledger visibility, downstream GelfandDuality request visibility, Compact and Topology boundary visibility, Real scalar sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

A downstream consumer may inspect these algebraic rows as the unital commutative C-source boundary. It receives no completed spectrum, no host Gelfand transform, no maximal ideal space, no host equality of algebras, no quotient completion, no selected character, and no choice principle.*

Proof. Project the carrier of Definition 3436.1. The projection exposes the root CStarAlg row A, the shared Banach and RingUp rows B, R, the unit and commutativity rows I, M, the involution and norm-square rows E, Q, the character-facing ledger L, the spectrum request X, the compact-topology boundary T, the Real scalar seal Y, and the structural rows H, C, P, N.

Source compatibility is inherited from the root certificate of Definition 211.3 and the Banach–Ring boundary of Definition 211.7. The present packet adds only the displayed unit and commutativity rows over that same finite source.

Stability is componentwise. Transport through H moves the root C*-algebra row, unit row, commutativity classifier, involution compatibility, norm-square row, ledger entries, spectrum request, compact-topology boundary, and Real scalar seal together; replay through C returns the same displayed algebraic source before any character-facing read is public.

Finally P and N name only this finite surface. Since a completed spectrum, host Gelfand transform, maximal ideal space, host equality, quotient completion, selected characters, and choice have no coordinate in the carrier, they cannot be recovered from the local NameCert surface. □

Theorem 3436.3 (Commutative CStar Gelfand spectrum boundary)**.** *Every* $\mathsf{GelfandDuality}^{\uparrow}$ *algebra-source or spectrum-facing read of an accepted* $\mathsf{CommutativeCStar}^{\uparrow}$ *packet factors through the same root CStarAlg row A, shared* $\mathsf{Banach}^{\uparrow}$ *row B, shared* $\mathsf{Ring}^{\uparrow}$ *row R, unit row I, commutativity classifier M, involution row E, norm-square row Q, character-facing ledger L, spectrum request row X, compact-topology boundary row T, Real scalar seal row Y, and structural rows H, C, P, N.*

Consequently $\mathsf{GelfandDuality}^{\uparrow}$ *can cite* $\mathsf{CommutativeCStar}^{\uparrow}$ *as its unital commutative C*-source boundary while the spectrum side remains a downstream* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *read. The packet does not treat the root* $\mathsf{CStarAlg}^{\uparrow}$ *chapter as already commutative and does not import a completed spectrum, host transform, maximal ideal space, quotient, host equality, selected character, or choice data.*

Proof. Begin with a $\mathsf{GelfandDuality}^{\uparrow}$ algebra-source read. The root C*-algebra dependency is the row A, and Definition 211.5 requires unit and commutativity rows to be displayed before that dependency can be consumed as a unital commutative source.

The rows I and M supply exactly that added algebraic surface. The rows E and Q keep the involution and norm-square laws attached to the same Banach and RingUp dependencies B, R already carried by A.

The character-facing ledger L is only a ledger of available algebraic rows. The downstream row X records that character and evaluation data belong to the $\mathsf{GelfandDuality}^{\uparrow}$ packet, while T and Y record the compact-topological spectrum boundary and Real scalar seal that such a packet may consume.

Transport, replay, provenance, and naming are then handled by H, C, P, N. Since these structural rows preserve only the displayed algebraic packet and its downstream boundary rows, the $\mathsf{GelfandDuality}^{\uparrow}$ source read remains the finite $\mathsf{CommutativeCStar}^{\uparrow}$ boundary and cannot collapse into host-level duality data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CommutativeCStar}^{\uparrow}$ packet is generated from a root CStarAlgUp row, shared BanachUp and RingUp dependency rows, a unit row, a commutativity classifier, involution compatibility, norm-square compatibility, a character-facing algebraic ledger, downstream GelfandDuality request visibility, Compact and Topology boundary rows, a Real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3436.1, Definition 3436.2, and Definition 3436.3 seed the finite unital commutative C*-source boundary over CStarAlgUp, BanachUp, RingUp, GelfandDualityUp, CompactUp, TopologyUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CommutativeCStar}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed spectrum, maximal ideal space, host Gelfand transform, host algebra equality, quotient completion, selected character, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, unit visibility, commutativity stability, involution and norm-square compatibility, character-ledger visibility, compact-topology boundary routing, Real scalar sealing, and Gelfand spectrum non-escape for BEDC.Derived.CommutativeCStarUp.

Chapter 3437

A Concrete Naming Certificate for MetaCICCriticalPairLedger$^{\uparrow}$

The MetaCICCriticalPairLedger$^{\uparrow}$ packet realizes a finite audit-map ledger for MetaCIC critical pairs. It reads the confluence critical-pair boundary of chapter 2212, the frontier critical-pair route of chapter 2382, the frontier dependency matrix of chapter 2307, the bounded normal-equality checker of chapter 752, and the subject-reduction route classifier of chapter 785. The packet stores critical-pair entries as a ledger of finite rows rather than as a global confluence theorem.

Falsifiable prediction: Every accepted MetaCICCriticalPairLedger$^{\uparrow}$ packet exposes a finite critical-pair table, local join ledger, bounded-checker handoff, route-separation row, audit-map replay row, provenance, and local naming before any confluence-facing consumer can read it.

Independence witness: metacic-confluence-critical-pair-boundary, metacic-frontier-critical-pair, metacic-frontier-dependency-matrix, bounded-normal-equality-checker, subject-reduction-route-classifier: these siblings respectively expose common-reduct boundary rows, frontier local-join rows, dependency accounting, bounded normal comparison, and typing-route separation; none is bijective with the packet that jointly records the ledger index, entry table, local joins, checker handoff, route refusal, audit replay, provenance, and local naming.

Definition 3437.1 (MetaCIC critical-pair ledger carrier)**.** *A* MetaCICCriticalPairLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ *source row.*
2. *F is the* MetaCICFrontierCriticalPair$^{\uparrow}$ *frontier route row.*
3. *D is the* MetaCICFrontierDependencyMatrix$^{\uparrow}$ *dependency-accounting row.*
4. *K is the ledger key row, indexing the finite critical-pair entries accepted by the packet.*
5. *J is the local-join ledger row, assigning displayed join candidates to the accepted ledger entries.*
6. *E is the* BoundedNormalEqualityChecker$^{\uparrow}$ *handoff row used only after K and J have been displayed.*
7. *S is the* SubjectReductionRouteClassifier$^{\uparrow}$ *separation row, keeping typing discharge outside the critical-pair ledger.*

8. *A is the audit-map replay row, recording the finite readback path from an audit request through B, F, D, K, J, E, S.*
9. *H records componentwise* hsame *transport for all source, ledger, handoff, route-separation, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the ledger route.*
11. *P is the* Pkg *provenance over* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICFrontierCriticalPair}^{\uparrow}$, $\mathsf{MetaCICFrontierDependencyMatrix}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, or an audit-map read outside the finite ledger.

Lean4 ✓ `BEDC.Derived.MetaCICCriticalPairLedgerUp.MetaCICCriticalPairLedgerTasteGate single carrier alignment`

Theorem 3437.2 (MetaCIC critical-pair ledger NameCert obligations)**.** *For every accepted* MetaCICCriticalPairLed *carrier*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$ *surface consists exactly of carrier admission, critical-pair boundary compatibility, frontier route compatibility, dependency-accounting visibility, finite ledger-key display, local-join ledger display, bounded-checker handoff, subject-reduction route separation, audit-map replay, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

No obligation is stored outside those rows, and no consumer receives global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, or a hidden audit-map credential.

Proof. Project the carrier of Definition 3437.1. The projection exposes exactly $B, F, D, K, J, E, S, A, H, C, P, N$.

Boundary and frontier compatibility are read first from B and F. The dependency-accounting row D then records which frontier and checker dependencies are visible before any ledger entry can be consumed.

The ledger rows K and J are finite. A bounded-checker handoff through E is available only after the entry key and local join have been displayed, and the route-separation row S keeps subject-reduction discharge outside the ledger.

The audit replay A and structural rows H, C, P, N transport, replay, source, and name only the displayed ledger route. Since the excluded confluence and typing claims have no carrier coordinate, the NameCert surface is exhausted by the stated rows. □

Theorem 3437.3 (MetaCIC critical-pair ledger audit route)**.** *Every MetaCIC confluence audit-map read of an accepted* $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ *packet factors through the finite route*

$$B, F, D \longmapsto K, J \longmapsto E, S \longmapsto A, H, C, P, N.$$

Thus the packet concretizes the audit-map critical-pair surface as a ledger of displayed rows, not as a proof of global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or subject-reduction discharge.

Proof. An audit-map consumer first reads the critical-pair boundary B, frontier route F, and dependency-accounting row D. These rows determine which finite critical-pair entries are in scope.

The ledger key K and local-join ledger J are then read together. The bounded normal-equality checker E receives only these displayed entries and join candidates.

The route-separation row S remains visible before the audit replay A and structural rows package the read. Therefore the consumer receives a finite ledger route, while the excluded global confluence and subject-reduction claims remain outside the packet. □

Theorem 3437.4 (MetaCIC critical-pair ledger local-join induction)**.** *For every accepted* MetaCICCriticalPairLedge *carrier*

$$L = (B, F, D, K, J, E, S, A, H, C, P, N),$$

induction over the finite ledger-key row K preserves the route

$$B, F, D \longmapsto K, J \longmapsto E, S \longmapsto A, H, C, P, N$$

for each displayed critical-pair entry. Each induction step reads the local join from J before the bounded-checker handoff E and the subject-reduction separation row S, and no step upgrades the ledger to global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or unchecked subject-reduction discharge.

Proof. The base key is read after the source rows B, F, D. By Definition 3437.1, the first ledger entry has no join coordinate except the displayed local-join row J.

For the induction step, assume the route has been preserved for the earlier ledger keys. The next key is another finite entry of K, so the packet again exposes its join candidate through J before any bounded-checker handoff through E is available.

The separation row S remains downstream of the local join and checker rows. Applying Definition 3437.3, the audit replay can therefore continue only through A, H, C, P, N after the route $B, F, D \longmapsto K, J \longmapsto E, S$ has been displayed.

No excluded confluence principle is present in the carrier. Componentwise transport, continuation replay, provenance, and local naming preserve the finite ledger route without adding global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, or unchecked subject-reduction discharge. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICCriticalPairLedger}^{\uparrow}$ packet is generated from confluence critical-pair boundary rows, frontier critical-pair rows, dependency-accounting rows, finite ledger keys, local-join rows, bounded-checker handoffs, subject-reduction route separation, audit-map replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3437.1, Definition 3437.2, Definition 3437.3, the falsifiable prediction, and the independence witness seed the finite MetaCIC critical-pair ledger over boundary, frontier, dependency, checker, route-separation, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCriticalPairLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global confluence, arbitrary closed-term Church–Rosser, full parallel diamond, residual-substitution compatibility, unchecked subject-reduction discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary compatibility, frontier compatibility, dependency visibility, ledger display, checker handoff, route separation, audit replay, and nonescape for BEDC.Derived.MetaCICCriticalPairLedg

Chapter 3438

A Concrete Naming Certificate for $\mathsf{RankNullity}^{\uparrow}$

Following the standard finite-dimensional linear-algebra theorem over a field, the $\mathsf{RankNullity}^{\uparrow}$ packet records the BEDC surface at which a linear map decomposes its source ledger into kernel and image dimensions. It sits after the field carrier of chapter 45, the vector-space ledger of chapter 47, the linear-map packet of chapter 48, and the matrix presentation of chapter 49. The chapter names the rank-nullity certificate as finite NameCert data; it does not add basis choice, quotient-vector-space construction, or spectral structure.

Definition 3438.1 (Rank-nullity carrier)**.** *A* $\mathsf{RankNullity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, S, T, K, I, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LinMap}^{\uparrow}$ *code row, recording the displayed map whose kernel and image ledgers are being read.*
2. *S is the finite* $\mathsf{VecSpace}^{\uparrow}$ *source ledger over the displayed* $\mathsf{Field}^{\uparrow}$ *row, including its finite dimension token.*
3. *T is the finite* $\mathsf{VecSpace}^{\uparrow}$ *target ledger over the same field row, including the codomain dimension context used by the image row.*
4. *K is the kernel row, exposing exactly the source vectors whose L-readback reaches the zero row of T.*
5. *I is the image row, exposing exactly the target vectors reached by an L-readback from an accepted source row.*
6. *A is the rank row, naming the finite dimension token attached to I.*
7. *U is the nullity row, naming the finite dimension token attached to K.*
8. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same map and the same finite source and target ledgers.*
9. *H stores componentwise* hsame *transport for the map, source, target, kernel, image, rank, nullity, matrix presentation, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from a rank-nullity request through $L, S, T, K, I, A, U, M, H$ and back to the local certificate row.*

11. *P is the* Pkg *provenance over* $\mathsf{RankNullity}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary non-singleton function extensionality, quotient dimensions, basis selection, spectral theory, Lean verification, or bridge checking.

Theorem 3438.2 (Rank-nullity NameCert obligations)**.** *For every accepted* $\mathsf{RankNullity}^{\uparrow}$ *carrier*

$$R = (L, S, T, K, I, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the linear-map code, finite source ledger, finite target ledger, kernel row, image row, rank row, nullity row, matrix presentation, transport, replay, provenance, and local naming;*
2. *map-source-target coherence, requiring L to be read only against the finite* $\mathsf{VecSpace}^{\uparrow}$ *ledgers S, T over the displayed* $\mathsf{Field}^{\uparrow}$ *row;*
3. *kernel exactness, requiring K to contain precisely the source rows mapped by L to the zero row of T;*
4. *image exactness, requiring I to contain precisely the target rows reached by L from an admitted source row in S;*
5. *finite dimension accounting, requiring A and U to read the displayed dimension tokens of I and K and to satisfy the source-dimension ledger encoded by the packet;*
6. *matrix compatibility, requiring M to present the same L, S, T route without adding an external basis-choice row;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rank-nullity rows.*

No rank-nullity obligation is stored outside R.

Proof. Project the carrier of Definition 3438.1. The tuple lists the linear-map code, source and target vector-space ledgers, kernel, image, rank, nullity, matrix presentation, and the structural rows.

The coherence, kernel, image, and matrix obligations are exactly the rows L, S, T, K, I, M. The source and target ledgers supply the finite vector-space context; the kernel and image rows are read only through the displayed map; and the matrix row is a presentation of the same route rather than an additional choice principle.

The dimension obligation is stored in A, U together with the finite source ledger S. Since H, C, P, N only transport, replay, source, and name rows already present in the carrier, there is no remaining location for a quotient dimension, basis choice, spectral datum, or external function-extensionality witness. □

Lean4 ✓ `BEDC.Derived.RankNullityUp.RankNullityTasteGate single carrier alignment`

Theorem 3438.3 (Rank-nullity kernel-image ledger)**.** *Every accepted* $\mathsf{RankNullity}^{\uparrow}$ *packet factors a finite-dimensional linear-map read through the displayed route*

$$\mathsf{LinMap}^{\uparrow} \longrightarrow \mathsf{VecSpace}^{\uparrow}_{\mathrm{src}} \times \mathsf{VecSpace}^{\uparrow}_{\mathrm{tgt}} \longrightarrow (\ker, \mathrm{im}) \longrightarrow (\mathrm{rank}, \mathrm{nullity}) \longrightarrow \mathsf{Mat}^{\uparrow}.$$

Along this route the rank row A and nullity row U account for the finite source-dimension token of S by the kernel-image ledger, and every exported rank-nullity read preserves the same rows $L, S, T, K, I, A, U, M, H, C, P, N$.

Proof. Start with an accepted carrier $R = (L, S, T, K, I, A, U, M, H, C, P, N)$. Projection exposes the map code, source and target ledgers, kernel and image rows, rank and nullity rows, matrix presentation, and structural rows.

The rows K and I are the only kernel and image coordinates in the packet. Thus any consumer reading a kernel-image decomposition must use the L-route from S to T rather than a detached quotient or ambient function graph.

The rows A and U then record the finite dimension accounting for I and K. The matrix row M presents the same map over the same finite ledgers, while H, C, P, N preserve the row order under transport, replay, provenance, and local naming.

Therefore the exported read is exactly the displayed kernel-image ledger and cannot export arbitrary non-singleton function extensionality, quotient dimensions, basis selection, spectral theory, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3438.1, Definition 3438.2, and Definition 3438.3 seed the finite rank-nullity carrier over LinearMap, VecSpace, Matrix, and Field rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RankNullity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary non-singleton function extensionality, quotient dimensions, choice of bases, spectral theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, map-source-target coherence, kernel exactness, image exactness, finite dimension accounting, matrix compatibility, transport stability, and replay stability for BEDC.Derived.RankNullityUp.

Constructive story: A $\mathsf{RankNullity}^{\uparrow}$ packet is generated from a LinearMap code row, finite source and target VecSpace ledgers, kernel and image rows, rank and nullity rows, a Matrix presentation row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3439

A Concrete Naming Certificate for $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$

The $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ packet realizes the endpoint bridge isolated by the MetaCIC decidability frontier in chapter 9386. It sits between the bounded comparison boundary of chapter 749, the candidate-mediated normalisation frontier of chapter 892, and the closed-normal endpoint seal of chapter 740. The packet records how a bounded normal-output comparison may be consumed only after finished-normal evidence is displayed; it does not claim unrestricted closed beta-conversion decidability.

Definition 3439.1 (Bounded normal conversion bridge carrier)**.** *A* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, O, E, S, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$ *row, exposing the bounded normal-equality checker and its refusal boundary.*
2. *F is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row, exposing the candidate-mediated route by which normal computation is allowed to reach a finished endpoint.*
3. *O is the bounded normal-output row, naming the finite fuel, closedness window, and two normalizer outputs inspected by the checker.*
4. *E is the finished-normal evidence row, requiring the compared outputs to be closed normal forms before conversion evidence is exported.*
5. *S is the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *endpoint row, recording that the finished-normal endpoint can be consumed only at the normal-form boundary.*
6. *R is the refusal row, recording that a read bypassing either O or E is not a conversion bridge read.*
7. *H stores componentwise* hsame *transport for the decidability boundary, normalisation frontier, bounded output, finished evidence, normal-form seal, refusal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a conversion request through D, F, O, E, S, R, H and back to the local certificate row.*

9. *P is the* Pkg *provenance over* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global conversion decider, fuel-erased normalization, quotient equality over represented terms, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.BoundedNormalConversionBridgeUp.BoundedNormalConversionBridgeUp`

Theorem 3439.2 (Bounded normal conversion bridge NameCert obligations)**.** *For every accepted* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *carrier*

$$B = (D, F, O, E, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the decidability boundary, normalisation frontier, bounded normal-output row, finished-normal evidence row, endpoint seal, refusal row, transport, replay, provenance, and local naming;*
2. *bounded-output visibility, requiring every conversion read to cite the finite output row O before checker equality is consumed;*
3. *finished-normal endpoint visibility, requiring E to certify that the compared outputs are finished closed normal forms before the normal-form seal S is read;*
4. *frontier compatibility, requiring D and F to be read together so bounded decidability is not separated from the normalisation route that supplies its endpoint evidence;*
5. *refusal stability, requiring R to reject any read that bypasses bounded normal output or finished-normal evidence;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the displayed rows.*

No bounded normal conversion bridge obligation is stored outside B.

Proof. Project the carrier of Definition 3439.1. The tuple exposes the decidability boundary, normalisation frontier, bounded output, finished evidence, normal-form seal, refusal, and structural rows.

The bounded-output and finished-endpoint obligations are the rows O and E. A conversion request first fixes the finite normalizer outputs and then supplies evidence that those outputs are finished closed normal forms.

The frontier compatibility obligation is stored in D, F, S. The bounded checker is read through the decidability boundary, the endpoint evidence is routed through the normalisation frontier, and the normal-form seal records the finished endpoint without creating a global conversion decision procedure.

The refusal and structural rows R, H, C, P, N preserve that order. Since the carrier contains no coordinate for fuel-erased normalization, quotient equality, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 3439.3 (Bounded normal conversion endpoint handoff)**.** *Every accepted* BoundedNormalConversionBridge *conversion read factors through the finite route*

$$\mathsf{MetaCICDecidableBoundary}^{\uparrow} \longrightarrow \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longrightarrow \mathsf{NormalFormConsistencySeal}^{\uparrow}$$

with bounded normal outputs and finished closed-normal evidence displayed before any conversion conclusion is exported.

Proof. Let $B = (D, F, O, E, S, R, H, C, P, N)$ be an accepted carrier. Projection exposes $D, F, O, E, S, R, H, C, P, N$ as the only rows available to a conversion read.

The row O fixes the bounded normalizer outputs inspected by the decidability boundary D. The row E then supplies the finished-normal evidence required before those outputs can be consumed as closed normal forms.

The row F supplies the normalisation-frontier route, and S places the endpoint at the normal-form consistency seal. The refusal row R rejects any path that omits O or E, while H, C, P, N preserve the same row order under transport, replay, provenance, and local naming.

Thus the packet supplies an endpoint handoff from bounded normal comparison to finished normal forms, not a global conversion decider or a fuel-erased normalization theorem. □

Falsifiable prediction: Any conversion read that bypasses either bounded normal output or finished-normal evidence refutes the $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ carrier.

Independence witness: metacicdecidableboundary, metacicnormalizationfrontier, normalformconsistencyseal, subjectreductiondischargeledger, metacicconfluenceauditpacket: the bridge carrier is not bijective to any listed sibling because it binds bounded normal-output comparison, finished-normal evidence, endpoint sealing, refusal, transport, replay, provenance, and local naming in one conversion-facing packet.

The Lean TasteGate follow-up for this chapter must provide explicit $\mathsf{Nontrivial\ BoundedNormalConversionBridge}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundedNormalConversionBridge}^{\uparrow}$ instances preserving $D, F, O, E, S, R, H, C, P, N$ as separate displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3439.1, Definition 3439.2, and Definition 3439.3 seed the bounded normal conversion bridge from decidability rows to finished normal-form endpoint evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalConversionBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted closed beta-conversion decidability, fuel-erased normalization, quotient equality over represented terms, host evaluator equality, unrestricted closedness decision, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-output visibility, finished-normal endpoint visibility, frontier compatibility, refusal stability, TasteGate nontriviality, and field faithfulness for BEDC.Derived.BoundedNormalConversio

Constructive story: A $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ packet is generated from MetaCICDecidableBoundary, MetaCICNormalizationFrontier, bounded normal-output, finished-normal evidence, NormalFormConsistencySeal, refusal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3440

A Concrete Naming Certificate for $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ packet records the finite comparison step used when two displayed dyadic or regular-rational windows must be compared before the terminal real seal is consumed. It sits over the dyadic tolerance rows of chapter 291, the finite observation windows of chapter 126, the regular-rational readback route of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the Bishop-style real comparison surface of chapter 1381. The packet names only a finite dyadic comparison surface; it is not real trichotomy, quotient-real equality, a selected limit, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 3440.1 (DyadicComparisonPrinciple carrier)**.** *A* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D_0, D_1, W, R, O, H, C, P, N)$$

with the following displayed rows:

1. *D_0 and D_1 are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows whose dyadic numerators, scales, tolerance ledgers, and descent witnesses are being compared.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule recording which finite observation windows of the two names are inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the scheduled windows into regular rational observations without selecting quotient representatives.*
4. *O is the finite comparison outcome row. It records one displayed dyadic relation among the inspected rows, or a displayed overlap certificate when the current window is not yet separating.*
5. *H records componentwise* hsame *transports for D_0, D_1, W, R, and O.*
6. *C records the displayed* Cont *replay routes by which downstream consumers inspect the two dyadic rows, the shared window, the regular-rational readback, and the finite comparison outcome.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for global real trichotomy, quotient-real equality, a selected limit, countable choice, a host order decision procedure, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicComparisonPrincipleUp.DyadicComparisonPrincipleUpTasteGate single carrier alignment`

Theorem 3440.2 (DyadicComparisonPrinciple NameCert obligations)**.** *For every accepted* DyadicComparisonPrinc *carrier* $K = (D_0, D_1, W, R, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the two dyadic rows, shared finite window schedule, regular-rational readback, finite comparison outcome, componentwise transport, displayed continuation replay, provenance, and local naming rows.*

Every $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{BishopRealComparison}^{\uparrow}$*-facing consumer of this packet must therefore pass through the displayed route*

$$D_0, D_1 \longmapsto W \longmapsto R \longmapsto O \longmapsto H, C, P, N$$

before it can read a dyadic comparison. The route exports no real trichotomy, no quotient equality of real names, no selected limit, no countable-choice schedule, and no ambient completeness coordinate.

Proof. Project the carrier of Definition 3440.1. The analytic rows are exactly the two dyadic endpoints, the shared finite window schedule, the regular-rational readback, and the finite outcome row.

Read the comparison in route order. The dyadic endpoint rows supply the tolerance ledgers, the window schedule names the finite observations, and the regular-rational row supplies the readback surface on which the outcome row is meaningful.

The transport and replay rows preserve those displayed coordinates componentwise. Provenance and local naming then source the same finite packet for downstream $\mathsf{Real}^{\uparrow}$ and $\mathsf{BishopRealComparison}^{\uparrow}$ consumers.

The excluded principles are absent from the carrier. Since no coordinate stores quotient equality, trichotomy, a selected limit, countable choice, a host decision procedure, or ambient completeness, none of them can be exported by the local certificate. □

Theorem 3440.3 (DyadicComparisonPrinciple trichotomy refusal)**.** *An accepted* $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ *packet exports finite dyadic comparison data only through* $D_0, D_1, W, R, O, H, C, P, N$*. In particular, the outcome row* O *may name a displayed separating relation or an overlap certificate for the inspected window, but it does not decide a global trichotomy for arbitrary* $\mathsf{Real}^{\uparrow}$ *names, identify quotient real equality, choose a limit representative, provide countable choice, or close ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the finite outcome row O. Its inputs are the two dyadic rows and the shared finite window route named by W and R, so it is a statement about the inspected window rather than about all real-name extensions.

Use Definition 3440.2. Every consumer read of the outcome must pass through the displayed carrier rows before reaching any $\mathsf{Real}^{\uparrow}$ or $\mathsf{BishopRealComparison}^{\uparrow}$-facing surface.

The packet therefore separates finite dyadic comparison from global order structure. Quotient real equality, selected limits, countable choice, host trichotomy, and real completeness have no row in the carrier and cannot be recovered from the provenance or local naming data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicComparisonPrinciple}^{\uparrow}$ packet is generated from two displayed DyadicRatCoreUp rows, a shared StreamName finite-window schedule, a RegSeqRat readback row, a finite comparison outcome or overlap certificate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 3440.1, Definition 3440.2, and Definition 3440.3 define the finite dyadic comparison packet and its trichotomy refusal boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicComparisonPrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, global real trichotomy, quotient-real equality, selected limits, countable choice, host order decision procedures, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite outcome transport, Real-facing handoff, BishopRealComparison-facing handoff, and non-escape theorems over BEDC.Derived.DyadicComparisonPrincipleUp.

Chapter 3441

A Concrete Naming Certificate for $\mathsf{CauchyQuotientCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, this packet records the finite quotient-read criterion for regular Cauchy real names. It sits over the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. Its algebra-facing dependencies are the displayed product carrier of chapter 461 and the synchronized sufficiency criterion of chapter 1586; the packet names quotient reads only through these finite rows and never through host division of completed real objects.

Definition 3441.1 (Cauchy quotient criterion carrier)**.** *A* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, Z, W, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A is the numerator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose quotient read is requested.*
2. *B is the denominator* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, accompanied only by displayed finite nonzero-window evidence in the dyadic ledger.*
3. *Z is the proposed quotient* $\mathsf{RegSeqRat}^{\uparrow}$ *row, read as a candidate regular rational name rather than as a selected representative of a quotient stream.*
4. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for A, B, Z, and the product check.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the requested quotient tolerance, the finite denominator-separation budget, and the residual comparison budget for $B \cdot Z$ against A.*
6. *P is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route for the displayed product of B and Z.*
7. *K is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row that accepts the product route before any regular-rational or real-facing quotient read is exposed.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the accepted quotient candidate after the product residual has been checked.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, P, K, R have been displayed.*
10. *H records componentwise* hsame *transport for the source, candidate, window, dyadic, product, criterion, handoff, seal, continuation, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a quotient tolerance request through W, D, P, K, R, E and then through the structural rows.*
12. *L is the* Pkg *provenance over* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, Z, W, D, P, K, R, E, H, C, L, N$. It has no coordinate for quotient streams, selected representatives, host real division, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 3441.2 (Cauchy quotient criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier $Q = (A, B, Z, W, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the numerator, denominator, candidate quotient, synchronized windows, dyadic quotient ledger, product route, product criterion, regular-rational handoff, real seal, transport, continuation, provenance, and local name; finite denominator-window coverage through W and D; product-residual exactness through P and K;* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R;* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E; and refusal of quotient streams, selected representatives, host real division, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

Proof. Carrier admission is projection from Definition 3441.1. The accepted packet displays the numerator row A, denominator row B, candidate quotient row Z, synchronized finite windows, dyadic quotient ledger, product route, criterion row, regular handoff, real seal, and the structural rows.

The denominator and residual checks are finite. The row D stores the denominator-separation budget and the product residual budget for the same windows named by W, while P and K route the displayed product $B \cdot Z$ through the regular Cauchy product surface and its criterion.

The consumer-facing rows are downstream. The handoff R is available only after the product residual has passed through W, D, P, K, and the seal row E is available only after R has been displayed. The rows H, C, L, N transport, replay, source, and name this same route.

Since the carrier has no coordinate for quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyQuotientCriterionUp.CauchyQuotientCriterionTasteGate single carrier alignment`

Theorem 3441.3 (Cauchy quotient criterion Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing quotient read of an accepted* $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ *carrier factors through the displayed finite route*

$$A, B, Z \longmapsto W \longmapsto D \longmapsto P \longmapsto K \longmapsto R \longmapsto E$$

together with the stored hsame *transport H,* Cont *replay C,* Pkg *provenance L, and local* NameCert *row N. The route exposes a quotient-read criterion for regular Cauchy real names, not quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing quotient consumer. By the carrier definition, the only $\mathsf{Real}^{\uparrow}$ seal row is E, and it is downstream of the regular-rational handoff R.

The handoff R is not primitive quotient data. It is admitted only after the synchronized window row W and dyadic ledger D have displayed the denominator-separation and residual

budgets, and only after P and K have routed the product check through $\mathsf{RegularCauchyProduct}^{\uparrow}$ and $\mathsf{CauchyProductCriterion}^{\uparrow}$.

The structural rows preserve the same route. Componentwise transport H, replay C, provenance L, and local naming N can only re-expose the finite packet already displayed.

Thus every real-facing quotient read reaches E through the named finite chain and cannot import quotient streams, selected representatives, host real division, trichotomy, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyQuotientCriterion}^{\uparrow}$ packet is generated from numerator, denominator, and candidate quotient RegSeqRat rows, synchronized StreamName windows, a DyadicRatCore quotient ledger, a RegularCauchyProduct route, a CauchyProductCriterion sufficiency row, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3441.1, Definition 3441.2, and Definition 3441.3 seed the finite quotient-read route for regular Cauchy real names through synchronized windows, dyadic denominator and residual budgets, product checking, RegSeqRat handoff, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyQuotientCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyQuotientCriterionUp.CauchyQuotientCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, host real division, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite denominator-window coverage, product-residual exactness, RegSeqRat handoff, Real-seal nonescape, transport stability, continuation replay, and ledger non-escape for BEDC.Derived.CauchyQuotientCriterionUp.

Chapter 3442

A Concrete Naming Certificate for PicardFixedPointModulus$^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive analysis, the PicardFixedPointModulus$^{\uparrow}$ packet records the finite tail modulus that a fixed-point consumer reads before any endpoint is selected. It sits after the contraction-map surface of chapter 1418, the one-step Picard route of chapter 395, the separate complete-metric consumer row of chapter 128, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a constructive modulus handoff for Picard tails; it is not a selected fixed point, not a completed orbit endpoint, not host fixed-point equality, and not ambient Real$^{\uparrow}$ completeness.

Definition 3442.1 (PicardFixedPointModulus carrier)**.** *A* PicardFixedPointModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, I, L, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* ContractionMapping$^{\uparrow}$ *source packet, including its metric carrier, self-map graph, rational contraction-bound ledger, and one-step Picard route.*
2. *I is the* PicardContraction$^{\uparrow}$ *iterate ledger that records the inspected Picard approximants and adjacent contraction comparisons.*
3. *L is the strict-ratio tail ledger, listing the finite bound powers and tolerance requests used to answer tail estimates.*
4. *T is the modulus threshold row. Each entry names a displayed stage after which the Picard tail comparison is admitted for the requested tolerance.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row that exposes the Picard orbit as regular rational sequence data only after K, I, L, T have been displayed.*
6. *E is the* CompleteMetric$^{\uparrow}$/Real$^{\uparrow}$ *seal boundary reached by completion and Real-facing consumers after the regular readback route has been exposed.*
7. *H records componentwise* hsame *transport for the contraction source, Picard ledger, tail ledger, threshold row, regular readback, seal boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from contraction-map admission through Picard iteration, tail-threshold extraction, regular readback, and completion-facing seal.*

9. *P is the* Pkg *provenance over* PicardFixedPointModulus$^{\uparrow}$, ContractionMapping$^{\uparrow}$, PicardContraction$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{PicardFixedPointModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, I, L, T, R, E, H, C, P, N$. It has no coordinate for endpoint selection, completed orbit equality, quotient stream equality, host function extensionality, countable choice, trichotomy, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardFixedPointModulusUp.PicardFixedPointModulusTasteGate single carrier alignment`

Theorem 3442.2 (PicardFixedPointModulus NameCert obligations)**.** *For every accepted* PicardFixedPointModulus *carrier*

$$M = (K, I, L, T, R, E, H, C, P, N),$$

the local NameCert$_{\text{PicardFixedPointModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, contraction-source visibility through K, Picard-iterate visibility through I, strict-ratio tail accounting through L, threshold extraction through T,* RegSeqRat$^{\uparrow}$ *readback through R, completion and Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the fixed-point modulus route without adding endpoint selection, completed orbit equality, quotient stream equality, host function extensionality, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3442.1. The displayed rows $K, I, L, T, R, E, H, C, P, N$ are the entire local surface, so carrier admission is exactly the admission of those rows.

The contraction and Picard data are read first. The row K supplies the contraction-map packet named by chapter 1418, while I supplies the Picard iterate route named by chapter 395.

The modulus row is extracted from finite tail accounting. The strict-ratio ledger L lists the bound powers and tolerance requests, and T records the displayed threshold entries consumed by a fixed-point modulus request.

Only then can regular readback and completion-facing seals be consumed. The row R exposes the RegSeqRat$^{\uparrow}$ route, while E marks the later CompleteMetric$^{\uparrow}$ and Real$^{\uparrow}$ boundary. Since endpoint selection and ambient completeness are not carrier coordinates, the local NameCert obligations stop at this displayed modulus handoff. □

Theorem 3442.3 (PicardFixedPointModulus tail handoff)**.** *Every fixed-point or completion-facing consumer of an accepted* PicardFixedPointModulus$^{\uparrow}$ *packet must read*

$$\text{ContractionMapping}^{\uparrow},\ \text{PicardContraction}^{\uparrow},\ L,\ T,\ \text{RegSeqRat}^{\uparrow},\ \text{CompleteMetric}^{\uparrow},\ \text{Real}^{\uparrow}$$

in the order displayed by Definition 3442.1. The handoff supplies a finite Picard-tail modulus for downstream fixed-point consumers; it does not select a fixed point, identify a completed orbit endpoint, assert host fixed-point equality, or import ambient completeness.

Proof. Begin with the contraction source and Picard ledger. A consumer first reads K and I, so each iterate and adjacent contraction comparison remains anchored in the same displayed contraction packet.

Next read the finite tail accounting. The rows L and T expose the strict-ratio budget and the requested threshold; no infinite tail selector or completed endpoint is available at this stage.

The regular rational and completion-facing rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R receives only the displayed modulus route, and the $\mathsf{CompleteMetric}^{\uparrow}/\mathsf{Real}^{\uparrow}$ seal boundary E is visible only after that route has been replayed.

Thus the packet supplies a fixed-point modulus handoff rather than an endpoint selection principle. The omitted host-side principles have no carrier row and cannot be exported by H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{PicardFixedPointModulus}^{\uparrow}$ packet is generated from a ContractionMapping source packet, a PicardContraction iterate ledger, a strict-ratio tail ledger, a displayed threshold row, a RegSeqRat readback, a CompleteMetric/Real seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3442.1, Definition 3442.2, and Definition 3442.3 seed the finite Picard tail-modulus handoff from contraction and Picard data to regular readback and completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PicardFixedPointModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim endpoint selection, completed orbit equality, quotient stream equality, host fixed-point equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-source visibility, Picard-iterate visibility, strict-ratio tail accounting, threshold extraction, regular readback, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PicardFixedPointModulusUp.

Chapter 3443

A Concrete Naming Certificate for $\mathsf{RectifiableCurve}^{\uparrow}$

Following Burago–Burago–Ivanov metric geometry and Bishop–Bridges constructive analysis, the $\mathsf{RectifiableCurve}^{\uparrow}$ packet records rectifiability by finite partition-length ledgers over a displayed metric path. It consumes the metric carrier of chapter 57, the continuous-path graph surface of chapter 130, the terminal real-name seal of chapter 38, the dyadic length rows of chapter 291, and the finite variation boundary of chapter 398. The packet names only the partition-level length certificate; it is not a completed path space, not an arclength supremum, and not quotient equality of curves.

Definition 3443.1 (Rectifiable curve carrier)**.** *A* $\mathsf{RectifiableCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, d, \gamma, I, \Pi, L, R, H, C, P, N)$$

with the following displayed rows:

1. X, d *is the accepted* $\mathsf{Met}^{\uparrow}$ *carrier and distance row.*
2. γ *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for a path from the interval row* I *into* X*.*
3. I *is the finite interval row supplying the ordered endpoint and subdivision domain.*
4. Π *is the ordered finite partition ledger over* I*, including adjacency and refinement witnesses.*
5. L *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-length sum row obtained by reading* $d(\gamma(t_i), \gamma(t_{i+1}))$ *on the displayed partition edges.*
6. R *is the common-refinement row comparing two displayed partition ledgers and their induced dyadic length sums.*
7. H *records componentwise* hsame *transport of the metric carrier, path graph, interval row, partition ledger, dyadic length sum, common-refinement row, provenance, and local naming row.*
8. C *records displayed* Cont *replay for metric reads, path evaluation, interval subdivision, edge-distance readback, length summation, and refinement comparison.*
9. P *is the* Pkg *provenance row over* $\mathsf{RectifiableCurve}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^{\uparrow}}$ *row.*

The carrier has no coordinate for a supremum over all partitions, quotient equality of paths, completed path spaces, host arclength measures, or bridge evidence. Its public object is the finite partition-length route from $X, d, \gamma, I, \Pi, L, R$ through H, C, P, N.

Lean4 ✓ `BEDC.Derived.RectifiableCurveUp.TasteGate.RectifiableCurveTasteGate single carrier alignment`

Theorem 3443.2 (Rectifiable curve NameCert obligations)**.** *For the carrier of Definition 3443.1, the local* $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$ *surface consists of exactly five obligation rows:*

1. *carrier admission for the finite packet $K = (X, d, \gamma, I, \Pi, L, R, H, C, P, N)$;*
2. *common-refinement classification for displayed partition ledgers through R;*
3. *dyadic edge-length exactness for the sum row L over the adjacent edges of Π;*
4. *componentwise* hsame *and* Cont *transport for metric reads, path graph reads, partition reads, edge-distance readbacks, and length-sum replay;*
5. *exclusion of arclength suprema, quotient path equality, completed path spaces, and bridge evidence from the local certificate.*

Every metric, variation, or path-length consumer must therefore read rectifiability through the displayed finite partition row and its dyadic length ledger before any completion-facing object can be considered.

Proof. Project the carrier. The rows $X, d, \gamma, I, \Pi, L, R$ are the only geometric coordinates, while H, C, P, N only transport, replay, source, and name that same finite packet.

Read the partition route in order. The interval row I supplies the ordered domain, the path graph γ supplies endpoint reads in the metric carrier, and the partition ledger Π lists the adjacent edge reads consumed by L.

Common refinement is finite. The row R compares two displayed partition ledgers by refinement witnesses and compares only their induced dyadic edge-length sums, so the classifier never ranges over all partitions.

The structural rows preserve this boundary. Since no carrier coordinate names an arclength supremum, quotient path equality, completed path space, or bridge datum, the five listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$**.**

Constructive story: A $\mathsf{RectifiableCurve}^\uparrow$ packet is generated from a MetricUp carrier and distance row, a ContinuousMapUp path graph, a finite interval row, an ordered partition ledger, a DyadicRatCoreUp edge-length sum, a common-refinement row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$, seed)

Scope closed: Definition 3443.1 and Definition 3443.2 name the finite partition-length carrier and its five-row NameCert obligation surface over $\mathsf{Met}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, $\mathsf{Real}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, and $\mathsf{BoundedVariation}^\uparrow$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RectifiableCurve}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an arclength supremum, quotient equality of paths, completed path spaces, host measure construction, ambient metric completion, or bridge evidence.

Upgrade path: Obligation closure requires checked carrier admission, common-refinement classification, dyadic edge-length exactness, transport, replay, provenance, and ledger non-escape for BEDC.Derived.RectifiableCurveUp.

Chapter 3444

A Concrete Naming Certificate for $\mathsf{HeineCantor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{HeineCantor}^{\uparrow}$ packet records compact-metric uniform-modulus extraction as finite NameCert data. It reads the finite-net surface of chapter 129, the endpoint mesh and refinement rows of chapter 3431, the pointwise modulus rows of chapter 130, and the target consumer interface of chapter 369. The interval-specialized consumer also factors compact finite-net rows through the interval mesh and refinement rows of chapter 3868 before it exports the uniform-modulus consumer route. The chapter names the constructive route from compact finite data and local continuity rows to one displayed uniform modulus, without importing open-cover compactness, a selected subcover functional, trichotomy on host reals, or ambient completeness.

Definition 3444.1 (Heine-Cantor carrier)**.** *A* $\mathsf{HeineCantor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite* $\mathsf{ProbeBundle}$ *net rows at the source precision scales consumed by the modulus extraction.*
2. *Y is the target metric row whose distance ledger receives the final uniform estimate.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from X to Y, including the pointwise modulus rows attached to listed net centers.*
4. *E is the positive target precision row, read as a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint history when the consumer presents a real tolerance.*
5. *N is the compact finite-net row selected from X for the scale determined by E.*
6. *R lists the local pointwise radius rows for the centers of N, with rational lower-bound readbacks through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the local rational classifier.*
7. *L is the* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *mesh and refinement handoff row when the finite-net data is produced from located interval endpoints.*
8. *M is the exported source modulus row, obtained by the finite fold over the radius rows R.*
9. *T is the target estimate row, combining compact-net coverage, two pointwise estimates around a selected center, and the target metric triangle route.*

10. *H records componentwise* hsame *transport for the compact source, target metric, continuous-map, precision, finite-net, radius, located-interval, modulus, and target-estimate rows.*
11. *C records the displayed* Cont *replay route from compact data and pointwise continuity data to the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer.*
12. *P records* Pkg *provenance over* $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, *and* Cont *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}$ *row.*

The carrier exposes only compact finite-net rows, continuous-map modulus rows, located mesh/refinement rows, endpoint readbacks, componentwise transport, replay, provenance, and local naming. It has no coordinate for arbitrary open covers, selected center choice, host real equality, quotient completion, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineCantorUp.HeineCantorTasteGate single carrier alignment`

Theorem 3444.2 (Heine-Cantor compact uniform modulus)**.** *For every accepted* $\mathsf{HeineCantor}^{\uparrow}$ *carrier*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A),$$

the exported source row M is a $\mathsf{UniformModulus}^{\uparrow}$ *witness for the continuous map row F at target precision E. Any consumer of the uniform-modulus witness must factor through the finite route*

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

Thus compact-metric finite nets, $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus rows,* $\mathsf{LocatedIntervalCompactness}^{\uparrow}$ *mesh and refinement data,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint readbacks, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *ledger suffice to export the compact uniform modulus.*

Proof. Project the carrier of Definition 3444.1. The target precision row E determines the compact source scale, and the $\mathsf{CompactMetric}^{\uparrow}$ row X supplies the finite net row N at that scale.

Read the continuous-map row F at each center of N. The pointwise radius rows are stored in R, and their rational lower-bound readbacks are attached to the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ endpoint rows already present in the packet.

When the source net is produced from an interval presentation, the row L supplies the located mesh and finite refinement data of chapter 3431. This row is a handoff into the compact finite-net route, not a general open-cover compactness principle.

Fold the finite family of radius rows R into the single source row M. The fold is finite because it ranges only over the centers listed by N, and componentwise transport in H preserves the compact source, continuous-map, endpoint, radius, and folded-modulus rows.

Finally use the target estimate row T. Compact-net coverage places two source points near a listed center, the pointwise rows in R estimate their images at that center, and the target metric ledger combines the estimates at precision E. Replay by C then presents M to the $\mathsf{UniformModulus}^{\uparrow}$ consumer, while P and A record the provenance and local naming ledger. □

Theorem 3444.3 (Heine-Cantor finite-net radius fold)**.** *Let*

$$K = (X, Y, F, E, N, R, L, M, T, H, C, P, A)$$

be an accepted $\mathsf{HeineCantor}^{\uparrow}$ *carrier. The row M is obtained only by a finite fold over the pointwise radius rows R indexed by the finite net N. When located interval data is present, the row L supplies*

the mesh and refinement handoff before the fold is read. Consequently every $\mathsf{UniformModulus}^{\uparrow}$ *consumer of* K *must read* N, R, L *before it reads the folded modulus row* M *and the target estimate row* T*.*

Proof. Project the carrier of Definition 3444.1. The finite-net row is N, the pointwise radius inventory is R, the located mesh and refinement handoff is L, and the exported modulus row is M.

Apply Definition 3444.2. Its route forces every compact uniform-modulus export through

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

The fold is finite because its indices are exactly the centers listed by the $\mathsf{ProbeBundle}$ net row N. The located interval row L refines the mesh when such data is used, but it does not replace the net-indexed radius list by open-cover compactness or selected subcovers. Thus any consumer must read N, R, L before the folded row M and the estimate row T are visible. □

Theorem 3444.4 (Heine-Cantor Dini window handoff)**.** *A* $\mathsf{DiniUniformConvergence}^{\uparrow}$ *consumer may use the* $\mathsf{HeineCantor}^{\uparrow}$ *compact uniform modulus only through the finite route*

$$X, F, E \longmapsto N, R, L \longmapsto M, T \longmapsto H, C, P, A.$$

Along this route the compact-window modulus rows supplied by the Heine–Cantor packet feed the Dini finite-window schedule of chapter 1436 without open-cover compactness, selected subcovers, countable choice, quotient real equality, or host real trichotomy.

Proof. Start with the Heine–Cantor carrier of Definition 3444.1. The compact source X, continuous-map row F, and target precision row E determine the finite-net and pointwise-radius rows N, R, L.

Fold those rows into the exported modulus and target estimate rows M, T. By Definition 3444.2, this is the only route by which a compact uniform modulus is presented to a $\mathsf{UniformModulus}^{\uparrow}$ consumer.

Now read the Dini packet through chapter 1436. Its finite-window schedule consumes compact-domain rows, continuous-map family rows, windows, readbacks, and Real seals; the Heine–Cantor route supplies the compact-window modulus part of that inventory while preserving the same finite-net discipline.

The structural rows H, C, P, A then transport, replay, source, and name the handoff. Since neither carrier contains an open-cover, selected-subcover, choice, quotient-equality, or trichotomy coordinate, the Dini consumer receives only the finite compact-window modulus route. □

Theorem 3444.5 (Heine-Cantor compact-modulus selector handoff)**.** *Let* $K = (X, Y, F, E, N, R, L, M, T, H, C, P,$ *be an accepted* $\mathsf{HeineCantor}^{\uparrow}$ *packet. Suppose the compact source row* X*, the total-bounded finite-net row exposed by* N*, the continuous-map row* F*, the dyadic tolerance row read from* E*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal all match an accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *packet. Then every* $\mathsf{UniformModulus}^{\uparrow}$ *export from* K *factors through Definition 4105.3 before the row* M *is visible to a consumer. The handoff exports no open-cover compactness, countable choice, host function equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the Heine–Cantor carrier of Definition 3444.1. Its compact source, continuous-map, tolerance, finite-net, readback, seal, transport, replay, provenance, and local naming rows are exactly the rows available to a uniform-modulus consumer.

Use the assumed row match with the $\mathsf{CompactModulusSelector}^{\uparrow}$ packet. The compact source row, total-bounded finite-net row, continuous-map row, dyadic tolerance ledger, finite windows,

regular readbacks, and Real seal are then the rows named by the selector carrier before its uniform-modulus export is visible.

Apply Definition 4105.3. It forces the selector export to pass through the finite route from compact metric data, total-bounded nets, pointwise continuity, dyadic comparison, finite windows, regular readback, and the Real seal before the $\mathsf{UniformModulus}^{\uparrow}$ row can be consumed.

Transport the resulting route back along the Heine–Cantor rows H, C, P, A. The exported row M is therefore a Heine–Cantor presentation of the selector's finite-net modulus route, and no excluded open-cover, choice, host-equality, or ambient-completeness coordinate is introduced. □

Theorem 3444.6 (Heine-Cantor compact uniform route scope)**.** *Every accepted* $\mathsf{HeineCantor}^{\uparrow}$ *consumer that exports a uniform modulus must first read* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus rows,* $\mathsf{CompactUniformModulusExtractor}^{\uparrow}$ *radius-extraction rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, and terminal* $\mathsf{Real}^{\uparrow}$ *seals. The export route is therefore an L10 finite dependency route through the displayed rows of Definition 3444.1, Definition 3444.2, Definition 3444.3, Definition 3444.4, and Definition 3444.5; it refuses open-cover compactness, selected moduli, quotient function equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 3444.1. Its compact source row X, continuous-map row F, target precision row E, finite-net row N, radius rows R, located handoff row L, exported modulus row M, and target estimate row T expose the compact finite route rather than an open-cover surface.

Apply the compact uniform modulus theorem and the finite-net radius fold, Definition 3444.2 and Definition 3444.3. They force every export through compact finite nets, pointwise radius rows, finite radius folding, and the target estimate ledger before the structural rows can replay the modulus.

Next use the Dini window handoff and compact-modulus selector handoff, Definition 3444.4 and Definition 3444.5. These handoffs identify the downstream L10 reads as finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, and Real seals.

The extractor row is therefore read only as finite radius extraction attached to those displayed dependencies. Since none of the cited rows contains an open cover, selected modulus, quotient function equality, or ambient completeness coordinate, the consumer receives exactly the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HeineCantor}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise modulus rows, LocatedIntervalCompactness mesh/refinement rows, RealUp and RegSeqRat endpoint readbacks, a finite radius fold, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3444.1 and Definition 3444.2 seed the compact finite-net route from pointwise continuity data to a UniformModulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineCantor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, choice, host real equality, trichotomy, quotient completion, ambient Real completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net extraction, finite radius fold, target estimate handoff, transport, replay, provenance, and local naming for BEDC.Derived.HeineCantorUp.

Chapter 3445

A Concrete Naming Certificate for $\mathsf{RealUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces, the $\mathsf{RealUniformSpace}^{\uparrow}$ packet records the finite L10 interface at which real endpoints, regular rational readback, stream windows, dyadic tolerance ledgers, and metric distance witnesses are assembled into a local uniformity surface. It sits beside the $\mathsf{Real}^{\uparrow}$ endpoint certificate of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{Met}^{\uparrow}$ distance witness route of chapter 57, and the regular Cauchy uniformity route of chapter 1628. The packet supplies a finite uniformity interface for Real and Completion consumers; it does not import host uniform spaces, quotient completions, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3445.1 (RealUniformSpace carrier)**.** *A* $\mathsf{RealUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, Q, W, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row, listing the endpoint names whose uniform comparison is allowed to be read.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the endpoint names.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected for the displayed endpoint and readback rows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the inspected radii, refinements, and finite comparison budgets.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *distance-witness row for the same displayed windows and dyadic tolerances.*
6. *E is the* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *entourage route reached only after R, Q, W, D, M have been displayed.*
7. *H records componentwise* hsame *transport for the endpoint, readback, window, tolerance, metric, entourage, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniformity request through $R, Q, W, D, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by componentwise displayed transport through these rows. It contains no host uniform-space carrier, quotient-completion equality, selected representative, open-cover compactness datum, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Theorem 3445.2 (RealUniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformSpace}^{\uparrow}$ *carrier* $U = (R, Q, W, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, endpoint access through* R*, regular rational readback through* Q*, finite window selection through* W*, dyadic tolerance accounting through* D*, metric distance witnessing through* M*, regular Cauchy entourage routing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3445.1. Each public obligation is attached to one displayed row, so carrier admission supplies precisely the endpoint, readback, window, tolerance, metric, entourage, transport, replay, provenance, and naming data.

The transport row is componentwise. Moving a packet along H changes only the displayed representatives inside their hsame classes and cannot create a host uniform-space carrier or a quotient-completion equality.

Replay by C follows the same route, and P, N source and name that route locally. Since no accepted row stores a selected representative, countable-choice schedule, open-cover compactness datum, or ambient completeness witness, the NameCert surface is exhausted by the displayed packet. □

Lean4 ✓ `BEDC.Derived.RealUniformSpaceUp.RealUniformSpaceTasteGate single carrier alignment`

Theorem 3445.3 (RealUniformSpace Cauchy-entourage route)**.** *Every* $\mathsf{RealUniformSpace}^{\uparrow}$ *consumer that asks for a Cauchy entourage comparison must traverse*

$$R \longmapsto Q \longmapsto W \longmapsto D \longmapsto M \longmapsto E$$

before using H, C, P, N*. Thus the real-uniform handoff reaches* $\mathsf{RegularCauchyUniformity}^{\uparrow}$ *only after the* $\mathsf{Real}^{\uparrow}$ *endpoint rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{Met}^{\uparrow}$ *distance witnesses have been displayed.*

Proof. Begin with a Cauchy-entourage request. The carrier admits endpoint rows only through R, so the request first exposes the real endpoints and then reads their regular rational names through Q.

The finite windows are then selected by W. Tolerance comparisons are recorded by D, and the metric-facing distance witnesses in M are read against the same windows and dyadic ledgers.

Only after those rows are visible may the route reach the entourage row E. The structural rows H, C, P, N transport, replay, source, and name this finite route, so no host uniformity, quotient completion, selected representative, or ambient completeness principle enters the handoff. □

Falsifiable prediction: Every $\mathsf{RealUniformSpace}^{\uparrow}$ Cauchy-entourage read factors through RealUp endpoint rows, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, MetricUp distance witnesses, a RegularCauchyUniformity entourage route, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore, metric, regular-cauchy-uniformity: $\mathsf{RealUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, regular readback, finite stream windows, dyadic tolerance ledgers, metric distance witnesses, regular Cauchy entourage routing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformSpace}^{\uparrow}$ packet is generated from RealUp endpoint rows, RegSeqRat readback, StreamName finite windows, a DyadicRatCore tolerance ledger, MetricUp distance witnesses, a RegularCauchyUniformity entourage route, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3445.1, Definition 3445.2, and Definition 3445.3 bind the finite real-uniform certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealUniformSpaceUp.RealUniformSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host uniform spaces, quotient completions, selected representatives, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Cauchy-entourage routing, metric tolerance accounting, and TasteGate field faithfulness for BEDC.Derived.RealUniformSpaceUp.

Chapter 3446

A Concrete Naming Certificate for $\mathsf{BishopRealApartnessTopology}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ packet records the finite neighbourhood surface obtained when Bishop real names are read through apartness-positive and located comparison evidence before any topology-facing consumer sees an open-neighbourhood ledger. It sits beside the terminal real-name seal of chapter 38, the positive separation surface of chapter 304, the located real packet of **??**, the topology interface of chapter 88, the finite stream windows of chapter 126, and the regular rational readback of chapter 301. The packet exports a constructive topology route for one displayed Bishop real neighbourhood; it exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.

Definition 3446.1 (Bishop real apartness topology carrier)**.** *A* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, L, T, W, Q, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *name row, read only through the terminal real-name seal selected for the packet.*
2. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-evidence row, recording the displayed apartness radius, endpoint readbacks, and positive separation ledger.*
3. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, recording the located left/right comparison windows used by the same neighbourhood request.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *consumer row, admitting only the topology-facing neighbourhood ledger generated from A, L, W, Q, O.*
5. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the real name, apartness evidence, located comparison, and neighbourhood ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations inspected by R, A, L, W.*
7. *O is the open-neighbourhood ledger generated from the apartness-positive evidence and located comparison rows before T is read.*
8. *H records componentwise* hsame *transport for the real name, apartness evidence, located comparison, topology row, stream schedule, regular readback, open-neighbourhood ledger, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Bishop real neighbourhood request through R, A, L, W, Q, O and then to the topology consumer row T.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealApartnessTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, A, L, T, W, Q, O, H, C, P, N$. It has no coordinate for trichotomy, host open sets, quotient real equality, countable choice, ambient completeness, or a topology object detached from the displayed open-neighbourhood ledger.

Lean4 ✓ `BEDC.Derived.BishopRealApartnessTopologyUp.BishopRealApartnessTopologyTasteGate single carrier alignment`

Theorem 3446.2 (Bishop real apartness topology NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealApartnessTopology}^{\uparrow}$ *carrier $B = (R, A, L, T, W, Q, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealApartnessTopol}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the* $\mathsf{Real}^{\uparrow}$ *name,* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-evidence row,* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row,* $\mathsf{Topology}^{\uparrow}$ *consumer row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, open-neighbourhood ledger, transport, replay, provenance, and local naming;*

2. *apartness-positive visibility, requiring the positive radius, endpoint readbacks, and separation ledger to be displayed in A before any neighbourhood is generated;*

3. *located comparison visibility, requiring every comparison used by the neighbourhood request to pass through the located row L and the shared W, Q windows;*

4. *topology admission, requiring the topology-facing consumer to read only the generated ledger O through T;*

5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No Bishop real apartness topology obligation is stored outside B, and the packet exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 3446.1. The displayed packet exposes exactly the real name, apartness-positive evidence, located comparison row, topology consumer row, stream schedule, regular readback, open-neighbourhood ledger, and structural rows.

Apartness-positive visibility is the row A. It carries the positive radius, endpoint readbacks, and separation ledger before the open-neighbourhood ledger is generated, so no trichotomy or host inequality row is needed.

Located comparison visibility is the row L together with W and Q. These rows align the comparison request with the finite stream window and regular-rational readback used by the same real name.

Topology admission is forced through O and T. The ledger O is generated only after the apartness and located rows have been displayed, and T admits that ledger as the topology-facing surface.

Finally H, C, P, N preserve only the displayed rows by transport, replay, provenance, and local naming. Since the carrier has no coordinate for host open sets, quotient equality, choice, completeness, or detached topology data, the listed rows exhaust the local NameCert surface. □

Theorem 3446.3 (Bishop real apartness topology Real seal)**.** *Every* Real$^{\uparrow}$*-facing or* Topology$^{\uparrow}$*-facing consumer of an accepted* BishopRealApartnessTopology$^{\uparrow}$ *carrier must pass through the route*

$$R, A, L, W, Q, O, H, C, P, N$$

before reading the topology row T. The handoff exposes the real name, apartness-positive evidence, located comparison rows, finite stream schedule, regular rational readback, open-neighbourhood ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, but it exports no trichotomy, host open set, quotient real equality, countable choice, or ambient completeness.*

Proof. Start with a Real-facing or topology-facing read of the accepted carrier. The only real-name source is R, and its finite observations are read through W and Q.

The neighbourhood evidence is forced through A and L. The apartness-positive row supplies the displayed separation ledger, while the located comparison row supplies the comparison windows used by the same request.

The topology consumer can read only after O has been generated. Thus T receives a ledger assembled from the displayed real, apartness, locatedness, stream, and regular-readback rows, not a host open set or quotient-open representative.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier contains no trichotomy, quotient equality, countable-choice, host-open, or completeness coordinate, every accepted handoff remains on the displayed finite route. □

Falsifiable prediction: Every accepted BishopRealApartnessTopology$^{\uparrow}$ Real-facing or topology-facing read factors through Real$^{\uparrow}$, ApartnessReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, Topology$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, open-neighbourhood ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: real, apartnessreal, locatedreal, topology, streamname, regseqrat: BishopRealApartnessTo is not bijective to any listed sibling because it jointly carries a real-name seal, apartness-positive evidence, located comparison rows, a topology-facing neighbourhood ledger, finite stream windows, regular rational readback, transport, replay, provenance, and local naming as one constructive topology packet.

Closure status for NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$, seed)

Scope closed: Definition 3446.1, Definition 3446.2, and Definition 3446.3 seed the finite Bishop real apartness topology route through Real$^{\uparrow}$, ApartnessReal$^{\uparrow}$, LocatedReal$^{\uparrow}$, Topology$^{\uparrow}$, StreamName$^{\uparrow}$, and RegSeqRat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopRealApartnessTopology}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopRealApartnessTopologyUp.BishopRealApartnessTopologyTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close trichotomy, host open sets, quotient real equality, countable choice, Lean theorem checking, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, apartness-positive visibility, located comparison visibility, topology admission, Real seal handoff, transport stability, and replay stability for BEDC.Derived.BishopRealApartnessTopologyUp.

Constructive story: The construction starts from a Bishop real name, records apartness-positive and located comparison evidence over finite StreamName and RegSeqRat windows, generates an open-neighbourhood ledger, and exposes the topology-facing consumer only along the same finite NameCert route.

Chapter 3447

A Concrete Naming Certificate for $\mathsf{FatouLemma}^{\uparrow}$

Following Fatou and Lebesgue's convergence theory, the $\mathsf{FatouLemma}^{\uparrow}$ packet records the finite lower-semicontinuity route from nonnegative measurable families to an integral lower-bound seal. It depends on the measure carrier of chapter 92, the integral readback surface of chapter 123, the regular rational tolerance interface of chapter 301, the Real seal of chapter 38, and the dominated-convergence boundary of chapter 2340. The packet is a lower-bound certificate only: it stores the rows needed to name Fatou's inequality inside BEDC, but it does not construct ambient measure completion, almost-everywhere quotient equality, quotient function equality, countable choice, or a full monotone convergence theorem.

Definition 3447.1 (FatouLemma carrier)**.** *A* $\mathsf{FatouLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, I, A, L, Q, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row naming the finite measurable-domain data and the measurable-set ledger read by the functions below.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row used for every lower-bound comparison in the packet.*
3. *A is the nonnegative family row: each displayed function in the sequence is admitted as a measurable nonnegative integrand over the same M.*
4. *L is the liminf window row, storing the finite tail windows whose lower envelope is consumed by the Fatou comparison.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback row for the tail-window lower bounds.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *lower-bound seal asserting that the named liminf integral is bounded above by the displayed lower limit of integral readbacks.*
7. *T records the transport from the family rows through the liminf windows to the integral and Real readbacks.*
8. *H records componentwise* hsame *transport for M, I, A, L, Q, R, T and the structural rows below.*
9. *C records the displayed* Cont *replay from Measure source rows to integral reads, from nonnegative family rows to liminf windows, and from regular rational lower bounds to the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FatouLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for an ambient completed measure algebra, host almost-everywhere equality, quotient function equality, a countable-choice selector for tails, or a monotone-convergence theorem.

Lean4 ✓ `BEDC.Derived.FatouLemmaUp.FatouLemmaTasteGate single carrier alignment`

Theorem 3447.2 (FatouLemma NameCert obligations)**.** *For every accepted* $\mathsf{FatouLemma}^{\uparrow}$ *carrier* $F = (M, I, A, L, Q, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FatouLemma}^{\uparrow}}$ *surface consists exactly of Measure source exposure through* M*, Integral readback through* I*, nonnegative family admission through* A*, liminf window exposure through* L*, RegSeqRat tolerance readback through* Q*, Real lower-bound sealing through* R*, transport through* T, H*, continuation replay through* C*, package provenance through* P*, and local naming through* N*.*

Thus every accepted Fatou comparison is a finite lower-bound packet over the displayed rows. No obligation may be discharged by ambient measure completion, host almost-everywhere equality, quotient function equality, countable choice, or a monotone convergence theorem.

Proof. Project the carrier of Definition 3447.1. The rows $M, I, A, L, Q, R, T, H, C, P, N$ are the only coordinates visible to the local classifier.

The analytic source rows are read first. The measure row M supplies the measurable-domain data, the integral row I supplies the readback surface, and the family row A records nonnegativity for each displayed function.

The lower-bound route is finite. The liminf window row L lists the tail windows, the regular-rational row Q records the tolerance readbacks for those windows, and the Real row R seals the comparison after the same data has been transported and replayed through T, H, C.

The package and local naming rows P, N source exactly this route. Since the excluded completion, equality, quotient, choice, and monotone-convergence coordinates are absent from the carrier, they cannot discharge a NameCert obligation. □

Theorem 3447.3 (FatouLemma liminf handoff)**.** *Let* $F = (M, I, A, L, Q, R, T, H, C, P, N)$ *be an accepted* $\mathsf{FatouLemma}^{\uparrow}$ *packet. Every liminf consumer factors through the finite route*

$$\mathsf{Measure}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto L \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads the Measure source row M*, the Integral readback row* I*, the nonnegative family row* A*, the liminf windows in* L*, and the RegSeqRat tolerance row* Q *before the lower-bound seal* R *is visible.*

Proof. Begin with the displayed liminf window row. Because L is part of the accepted carrier, each requested tail window is tied to the same Measure source M and the same Integral readback I.

Use the nonnegative family row A to keep every tail comparison inside the displayed measurable family. The row Q then records the regular rational tolerances used by those finite windows.

Finally read the Real seal. The rows T, H, C transport and replay the same lower-bound route into R, while P, N package and name it. No external tail selector, quotient equality, or completion principle is available to replace this handoff. □

Theorem 3447.4 (FatouLemma dominated-convergence boundary)**.** *The* FatouLemma$^{\uparrow}$ *packet supplies the lower-bound side needed by the dominated-convergence route of chapter 2340: it exposes the nonnegative family row, liminf window row, RegSeqRat tolerance readback, Integral readback, and Real lower-bound seal as finite NameCert data over the same Measure source.*

It does not export the domination majorant, almost-everywhere convergence quotient, two-sided limit equality, or global convergence conclusion of DominatedConvergence$^{\uparrow}$*. Those remain obligations of the dominated-convergence packet, whose boundary also cites the human-known comparison tradition around* TanneryTheorem$^{\uparrow}$*.*

Proof. Project the Fatou carrier and retain only the rows consumed by a dominated-convergence lower-bound reader: $M, I, A, L, Q, R, T, H, C, P, N$.

Apply Definition 3447.3. The lower-bound read reaches the Real seal only after the Measure source, Integral readback, nonnegative family row, liminf windows, and regular rational tolerance row have all been displayed.

The dominated-convergence packet has additional surface: domination data, convergence equality data, and a two-sided limit conclusion. Since none of those coordinates appears in Definition 3447.1, this chapter supplies only the Fatou lower-bound boundary and not the full dominated-convergence theorem. □

Closure status for NameCert$_{\text{FatouLemma}^{\uparrow}}$**.**

Constructive story: A FatouLemma$^{\uparrow}$ packet is generated from a Measure source row, an Integral readback row, a nonnegative family row, a liminf window row, RegSeqRat tolerance readback, a Real lower-bound seal, transport, continuation replay, package provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{FatouLemma}^{\uparrow}}$, seed)

Scope closed: Definition 3447.1, Definition 3447.2, Definition 3447.3, and Definition 3447.4 bind the finite lower-bound packet and its dominated-convergence boundary.

Formal verification: FormalStatus(NameCert$_{\text{FatouLemma}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FatouLemmaUp.FatouLemmaTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim ambient measure completion, host almost-everywhere equality, quotient function equality, countable choice, a full monotone convergence theorem, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Measure source exposure, Integral readback, nonnegative family admission, liminf window exposure, RegSeqRat tolerance readback, and Real lower-bound sealing for the same packet.

Chapter 3448

A Concrete Naming Certificate for $\mathsf{RadonNikodym}^{\uparrow}$

Following Radon's derivative of measures, Nikodym's theorem, and the standard measure-theoretic presentation of absolute continuity, the $\mathsf{RadonNikodym}^{\uparrow}$ packet records a finite density route from one displayed measure row to another. It sits after the measurable-event carrier of chapter 92, the integral readback surface of chapter 123, the regular rational tolerance interface of chapter 301, the Real seal of chapter 38, and the lower-bound comparison boundary of chapter 3447. The packet names the rows needed to consume a density witness inside BEDC; it is not ambient measure completion, quotient almost-everywhere equality, countable choice, or an imported Radon–Nikodym theorem.

Definition 3448.1 (RadonNikodym carrier)**.** *A* $\mathsf{RadonNikodym}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *M_μ is the base* $\mathsf{Measure}^{\uparrow}$ *row, exposing the measurable-event family, finite-additivity rows, and Real-valued measure readbacks.*
2. *M_ν is the target* $\mathsf{Measure}^{\uparrow}$ *row on the same displayed event family.*
3. *A is the absolute-continuity ledger: every event read as μ-null carries a displayed route to a ν-null comparison before any density row is consumed.*
4. *D is the candidate density row, stored as a nonnegative measurable integrand over the shared event family.*
5. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row assigning each displayed event E the integral of D over E.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance ledger for the finite simple-function and event-window approximants used by the density comparison.*
7. *S is the* $\mathsf{Real}^{\uparrow}$ *seal comparing the target-measure readback $\nu(E)$ with the integral readback of D on each requested event window.*
8. *H records componentwise* hsame *transport for $M_\mu, M_\nu, A, D, I, Q, S$ and the structural rows below.*
9. *C records displayed* Cont *replay from measure rows through absolute-continuity ledgers, density windows, integral readback, regular rational tolerance, and Real sealing.*

10. *P is the* Pkg *provenance over* RadonNikodym$^\uparrow$, Measure$^\uparrow$, Integral$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, FatouLemma$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{RadonNikodym}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed measure algebras, quotient almost-everywhere equality, countable-choice selectors for representatives, or a host Radon–Nikodym theorem.

Lean4 ✓ `BEDC.Derived.RadonNikodymUp.RadonNikodymTasteGate single carrier alignment`

Theorem 3448.2 (RadonNikodym NameCert obligations)**.** *For every accepted* RadonNikodym$^\uparrow$ *carrier $R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$, the local* NameCert$_{\text{RadonNikodym}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *shared-event carrier admission for the two* Measure$^\uparrow$ *rows M_μ and M_ν;*
2. *absolute-continuity exposure through A, requiring null-event requests to remain visible in the measure ledger;*
3. *density admission through D, requiring the candidate density to be a displayed nonnegative measurable integrand;*
4. *integral readback through I, requiring each event-facing density read to pass through the public* Integral$^\uparrow$ *row;*
5. *regular-rational tolerance accounting through Q, requiring finite simple-function and event-window approximants before Real-facing comparison;*
6. *Real sealing through S, requiring each displayed event comparison to be named only after M_ν, D, I, Q have been read;*
7. *structural stability through H, C, P, N, requiring transport, replay, provenance, and local naming to preserve exactly the displayed route.*

No obligation may be discharged by ambient measure completion, quotient almost-everywhere equality, countable choice, or an imported Radon–Nikodym theorem.

Proof. Project the carrier of Definition 3448.1. The two measure rows, the absolute-continuity ledger, the candidate density, the integral readback, the tolerance ledger, the Real seal, and the structural rows are the entire local surface.

The source side is read first. The rows M_μ and M_ν expose the common measurable-event family, while A records the null-event implication that licenses density search without hiding a completed measure algebra.

The density side is finite. The row D admits the displayed nonnegative measurable integrand, I reads its eventwise integrals, and Q records the finite tolerance windows used before a Real comparison is available.

Finally S seals the eventwise comparison and H, C, P, N transport, replay, source, and name the same route. Since the excluded completion, quotient, choice, and host theorem coordinates are absent from the carrier, they cannot discharge any NameCert obligation. □

Theorem 3448.3 (RadonNikodym density route)**.** *Every* Measure$^\uparrow$, Integral$^\uparrow$, RegSeqRat$^\uparrow$, *or* Real$^\uparrow$ *consumer of an accepted* RadonNikodym$^\uparrow$ *carrier $R = (M_\mu, M_\nu, A, D, I, Q, S, H, C, P, N)$ must read the finite route*

$$M_\mu, M_\nu \longmapsto A \longmapsto D \longmapsto I \longmapsto Q \longmapsto S \longmapsto H, C, P, N.$$

Thus the target measure is consumed through displayed absolute-continuity data, a density row, Integral readback, RegSeqRat tolerance ledgers, and the Real seal before any density conclusion is available.

Proof. Begin with the two measure rows. A consumer first reads M_μ and M_ν, so every event request remains attached to the same displayed measurable-event family.

Next read the absolute-continuity and density rows. The ledger A keeps null-event transfer visible, and the row D supplies the candidate density as a displayed nonnegative measurable integrand rather than an almost-everywhere quotient class.

The integral and Real-facing route is downstream. The row I reads the density on each requested event, Q records the regular rational approximants and tolerances, and S seals the comparison in $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N preserve only this route. Therefore no consumer can bypass $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or the Fatou-style lower-bound boundary of chapter 3447 by importing measure completion, quotient almost-everywhere equality, countable choice, or the classical Radon–Nikodym theorem as hidden data. □

Falsifiable prediction: Every $\mathsf{RadonNikodym}^{\uparrow}$ density-facing or Real-facing read factors through displayed Measure source rows, an absolute-continuity ledger, a density integrand row, Integral readback, RegSeqRat tolerance windows, Real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: measure, integral, real, regseqrat, fatou-lemma: $\mathsf{RadonNikodym}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two measure rows, an absolute-continuity ledger, a density integrand, eventwise integral readback, tolerance accounting, Real comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}$.

Constructive story: A $\mathsf{RadonNikodym}^{\uparrow}$ packet is generated from two shared-event Measure rows, an absolute-continuity ledger, a displayed density integrand, Integral readback, RegSeqRat tolerance windows, Real comparison seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3448.1, Definition 3448.2, and Definition 3448.3 seed the finite density route from shared Measure rows to Integral, RegSeqRat, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RadonNikodym}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RadonNikodymUp.RadonNikodymTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim ambient measure completion, quotient almost-everywhere equality, countable choice, imported Radon–Nikodym theorem use, Lean theorem checking, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-event measure exposure, absolute-continuity ledger exactness, density admission, integral readback, regular-rational tolerance accounting, Real sealing, transport stability, and replay stability for BEDC.Derived.RadonNikodymUp.

Chapter 3449

A Concrete Naming Certificate for GaussBonnetBoundary$^{\uparrow}$

Following Chern, Gauss, Bonnet, and Stokes, the GaussBonnetBoundary$^{\uparrow}$ packet records the finite boundary-sensitive route from an oriented compact surface window to an Euler-characteristic ledger. It consumes the chart and orientation discipline of chapter 96, the metric rows of chapter 133, the curvature rows of chapter 135, the differential-form and Stokes surface of chapter 136, the cohomology-class readback of chapter 137, and the characteristic-class handoff of chapter 272. The packet names only the displayed surface, boundary, curvature, form, cohomology, transport, replay, provenance, and local NameCert rows; it does not claim host smooth integration, unrestricted compactness, full topological classification, Lean verification, or bridge checking.

Definition 3449.1 (Gauss-Bonnet boundary carrier)**.** *A* GaussBonnetBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, M, K, B, F, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the oriented compact surface window row, read through the local* Manifold$^{\uparrow}$ *chart, atlas, boundary, and orientation data.*
2. *M is the* RiemannianMetric$^{\uparrow}$ *row assigning the displayed metric data on the same surface window.*
3. *K is the* Curvature$^{\uparrow}$ *row naming the Gaussian-curvature display produced from M on the interior chart windows.*
4. *B is the boundary geodesic-curvature row, carrying the finite boundary arcs, endpoint turns, and tangent-readback data used by the boundary term.*
5. *F is the* DiffForm$^{\uparrow}$ *and Stokes row, recording the two-dimensional curvature form, the boundary one-form readback, and the Stokes-compatible orientation route.*
6. *D is the* DeRham$^{\uparrow}$ *class row that reads the curvature form as a cohomology-class witness over the displayed surface window.*
7. *E is the Euler-characteristic ledger row, storing the finite cell, triangulation, or admissible decomposition count consumed by the comparison.*
8. *H records componentwise* hsame *transport for S, M, K, B, F, D, E and the structural rows below.*
9. *C records displayed* Cont *replay from the surface window through metric, curvature, boundary, form, cohomology, and Euler-ledger reads in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{ChernWeil}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, K, B, F, D, E, H, C, P, N$. It has no coordinate for host smooth integration, arbitrary compact surface selection, quotient topological classification, external triangulation equality, or a full Chern-Weil theory.

Theorem 3449.2 (Gauss-Bonnet boundary NameCert obligations)**.** *For every accepted* $\mathsf{GaussBonnetBoundary}^{\uparrow}$ *carrier $G = (S, M, K, B, F, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *surface consists exactly of oriented compact surface admission through S, Riemannian metric admission through M, curvature display through K, boundary geodesic-curvature exposure through B, differential-form and Stokes routing through F, DeRham class readback through D, Euler-characteristic ledger exposure through E, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N.*

No Gauss-Bonnet boundary obligation may be discharged by a host integration operator, an unrestricted compactness principle, a hidden smooth structure, a quotient classification theorem, or a Chern-Weil characteristic-class result not displayed in the packet provenance.

Proof. Project the carrier of Definition 3449.1. The rows $S, M, K, B, F, D, E, H, C, P, N$ are the only coordinates visible to the local classifier.

The geometric source is read first. The oriented surface window S fixes the chart, atlas, orientation, and boundary data; the metric row M and curvature row K then expose only the curvature display available over that same window.

The boundary and form route is separate. The row B supplies the boundary geodesic-curvature ledger, while F supplies the differential-form and Stokes readback tying the interior and boundary terms to the displayed orientation.

Finally D and E provide the cohomology and Euler-characteristic ledgers consumed by the comparison. Transport, replay, provenance, and naming preserve this row order through H, C, P, N, so no excluded host or classification coordinate can discharge a local obligation. □

Lean4 ✓ `BEDC.Derived.GaussBonnetBoundaryUp.GaussBonnetBoundaryTasteGate single carrier alignment`

Theorem 3449.3 (Gauss-Bonnet boundary Stokes-curvature handoff)**.** *Let $G = (S, M, K, B, F, D, E, H, C, P, N)$ be an accepted* $\mathsf{GaussBonnetBoundary}^{\uparrow}$ *packet. Every accepted boundary-sensitive curvature read factors through the finite route*

$$\mathsf{Manifold}^{\uparrow} \longmapsto \mathsf{RiemannianMetric}^{\uparrow} \longmapsto \mathsf{Curvature}^{\uparrow} \longmapsto \mathsf{DiffForm}^{\uparrow} \longmapsto \mathsf{DeRham}^{\uparrow} \longmapsto \mathsf{ChernWeil}^{\uparrow}.$$

Concretely, the consumer must read the oriented compact surface row S, the metric row M, the curvature row K, the boundary geodesic-curvature row B, the Stokes form row F, the DeRham class row D, and the Euler-characteristic ledger E before the local $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$ *row is visible.*

Proof. Begin with the surface window. The row S supplies the only accepted manifold, orientation, and boundary source for the packet.

Metric and curvature data are then read from M and K. These rows are tied to S by the classifier and cannot be replaced by an ambient smooth manifold or host integration surface.

The boundary term must pass through B, and the interior-plus-boundary form route must pass through F. The DeRham row D and Euler ledger E then supply the cohomology and characteristic-count surfaces consumed by the Chern-Weil-facing handoff.

The structural rows H, C, P, N transport, replay, source, and name exactly this finite route. Hence the packet exports a boundary Stokes-curvature handoff, not a full smooth integration theorem or topological classification theorem. □

Falsifiable prediction: Every accepted $\mathsf{GaussBonnetBoundary}^{\uparrow}$ read that exposes an Euler-characteristic ledger must also expose the oriented compact surface row, metric row, curvature row, boundary geodesic-curvature row, differential-form/Stokes row, and DeRham class row.

Independence witness: manifold, riemannianmetric, curvature, diffform, derham, chernweil: none is bijective to $\mathsf{GaussBonnetBoundary}^{\uparrow}$ because this packet jointly carries an oriented compact surface window, metric and curvature displays, boundary geodesic-curvature data, Stokes form routing, DeRham readback, Euler-characteristic ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{GaussBonnetBoundary}^{\uparrow}$ packet is generated from an oriented compact surface window, a RiemannianMetric row, a Curvature row, a boundary geodesic-curvature row, a DiffForm/Stokes row, a DeRham class row, an Euler-characteristic ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3449.1, Definition 3449.2, and Definition 3449.3 seed the finite boundary-sensitive Gauss-Bonnet NameCert route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GaussBonnetBoundary}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.GaussBonnetBoundaryUp.GaussBonnetBoundaryTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host smooth integration, unrestricted compactness, full topological classification, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric and curvature row exactness, boundary geodesic-curvature exposure, Stokes form routing, DeRham readback, Euler-ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.GaussBonnetBoundaryUp.

Chapter 3450

A Concrete Naming Certificate for $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$

The $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ packet realizes the Force–MetaCIC relation discipline of section 9416.5 as finite NameCert data. It consumes the same-schema carrier of chapter 1090, the family-level audit rows of chapter 807, the confluence-audit packet of chapter 735, the quotient-refusal boundary of chapter 634, and the authorized-generator route of chapter 525. The packet names a two-chain relation schema only; it does not claim empirical force content, identity between the domains, object-level MetaCIC closure, or bridge-checked verification.

Definition 3450.1 (Force-MetaCIC same-schema carrier)**.** *A* $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, B, L, P, Q, R, V, H, C, K, N)$$

with the following displayed rows:

1. *A is the force-cognition source-chain row, recording the displayed primitive-token, obstruction, classifier, ledger, and boundary stages.*
2. *B is the MetaCIC self-recognition source-chain row, recording the displayed token, audit, carrier, classifier, ledger, and boundary stages.*
3. *L is the finite two-chain layer-map row from the force chain to the MetaCIC chain.*
4. *P is the preserved-structure row: primitive-token replacement by relation, classifier, invariant, ledger, certificate, and boundary.*
5. *Q is the not-preserved row: empirical physical content, numerical constants, machine-checked proofs, phenomenal essence, object-level* T*, and global universe structure.*
6. *R is the failure-boundary row, refusing any read that treats same-schema exactness as object identity, empirical transfer, quotient equality, or a checked proof.*
7. *V is the same-schema verdict row, assigning paper-certificate strength to the displayed relation.*
8. *H records componentwise* hsame *transport for the two source chains, layer map, preserved row, not-preserved row, boundary row, and verdict row.*
9. *C records displayed* Cont *replay through A, B, L, P, Q, R, V in that order.*
10. *K is the* Pkg *provenance over* $\mathsf{LayeredRelationCert}^{\uparrow}$, $\mathsf{AuditMapFamily}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, B, L, P, Q, R, V, H, C, K, N$. It has no coordinate for empirical-force semantics, object identity, quotient transport, hidden proof transfer, global universe structure, or unchecked promotion of paper-certificate strength.

Lean4 ✓ `BEDC.Derived.ForceMetaCICSameSchemaUp.ForceMetaCICSameSchemaTasteGate single carrier alignment`

Theorem 3450.2 (Force-MetaCIC same-schema NameCert obligations)**.** *For every accepted* ForceMetaCICSameSc*carrier $F = (A, B, L, P, Q, R, V, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$ *surface consists exactly of force-chain admission through A, MetaCIC-chain admission through B, finite layer-map exposure through L, preserved-structure exposure through P, explicit not-preserved exposure through Q, failure-boundary exposure through R, same-schema verdict exposure through V, componentwise transport through H, continuation replay through C, package provenance through K, and local naming through N.*

Any downstream use omitting the not-preserved row Q or the failure-boundary row R cannot consume the same-schema relation, because the packet would then lack the rows that prevent identity collapse, empirical transfer, quotient transport, and proof-strength promotion.

Proof. Project the carrier of Definition 3450.1. The two displayed chains are A and B, and the comparison surface is the layer map L together with the preserved, not-preserved, boundary, and verdict rows P, Q, R, V.

The same-schema read first consumes A, B, L. This fixes the two chains and the relation map without identifying their objects or importing empirical force content into MetaCIC.

The rows P and Q are read together. The first records what the relation preserves, and the second records what it explicitly does not preserve.

The row R then refuses reads that bypass Q or promote the verdict V beyond paper-certificate strength. Transport, replay, provenance, and local naming through H, C, K, N preserve this finite order. □

Theorem 3450.3 (Force-MetaCIC same-schema refusal boundary)**.** *Every accepted* $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ *read that exposes the same-schema verdict must also expose the not-preserved row and the failure-boundary row. Consequently the packet refuses object identity between force cognition and MetaCIC self-recognition, empirical transfer from force theory into BEDC, quotient equality, unchecked proof transfer, and any reading of paper-certificate strength as checked or bridged evidence.*

Proof. Begin with an accepted verdict read. By Definition 3450.1, the verdict coordinate is V and is classified only together with $A, B, L, P, Q, R, H, C, K, N$.

The not-preserved row Q is therefore present before the verdict can be consumed. It records the empirical, numerical, proof-theoretic, phenomenal, object-level, and global-structure content that the relation does not preserve.

The failure-boundary row R is also present. It blocks identity collapse, empirical transfer, quotient equality, hidden proof transfer, and strength promotion before the local name row N can export the packet.

Thus the same-schema relation is available only as the finite carrier named by this chapter. Later Lean work in `TasteGate.lean` must provide Nontrivial and FieldFaithful instances for the displayed carrier rows and classifier projections. □

Falsifiable prediction: Any downstream use of $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ that omits the not-preserved row or the failure-boundary row cannot consume the same-schema relation.

Independence witness: layeredrelationcert, auditmapfamily, metacicconfluenceauditpacket, quotientsoundnessboundary, authorizedgeneratorrecursor: none is bijective to $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ because this packet jointly carries two source chains, a layer map, preserved and not-preserved rows, a failure-boundary row, same-schema verdict, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}$.

Constructive story: A $\mathsf{ForceMetaCICSameSchema}^{\uparrow}$ packet is generated from a force-cognition source-chain row, a MetaCIC self-recognition source-chain row, a finite two-chain layer map, preserved-structure rows, not-preserved rows, failure-boundary rows, a same-schema verdict row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3450.1, Definition 3450.2, and Definition 3450.3 seed the finite Force-MetaCIC same-schema relation certificate.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ForceMetaCICSameSchema}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim empirical force content, identity between the domains, object-level MetaCIC closure, checked Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-chain layer-map exactness, preserved-row exposure, not-preserved-row exposure, failure-boundary nonescape, same-schema verdict routing, transport, replay, provenance, and local naming for BEDC.Derived.ForceMetaCICSameSchemaUp.

Chapter 3451

A Concrete Naming Certificate for $\mathsf{RealBoundedSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealBoundedSequence}^{\uparrow}$ packet records the finite bounded-sequence neighbour needed by the L10 real route. It reads terminal real names from chapter 38, finite observation windows from chapter 126, regular rational readback from chapter 301, and dyadic tolerance ledgers from chapter 291; its downstream consumer is the subsequence compactness surface of chapter 1254. The packet names bounded real sequence inspection as displayed NameCert data, not as ambient sequential compactness, trichotomy, quotient equality of streams, a selected subsequence, or host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3451.1 (RealBoundedSequence carrier)**.** *A* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row, listing the admitted real-name positions whose values are inspected by the packet.*
2. *B is the bound witness row, recording the displayed lower and upper bounding data and the finite ledger that each inspected value stays inside those bounds.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite observation windows of the source sequence.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational approximants used by the inspected windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed value and for the bound endpoints consumed by the packet.*
6. *H records componentwise* hsame *transport for S, B, W, R, E and the structural rows below.*
7. *C records the displayed* Cont *replay from source sequence windows through regular rational readback to the terminal Real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RealBoundedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$ *row.*

The carrier exposes only the source sequence, displayed bound witness, finite windows, regular rational readbacks, terminal Real seals, transport, replay, provenance, and local naming. It has no coordinate for arbitrary subsequence choice, host sequence equality, quotient stream equality, trichotomy on reals, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealBoundedSequenceUp.RealBoundedSequenceTasteGate single carrier alignment`

Theorem 3451.2 (RealBoundedSequence NameCert obligations)**.** *For every accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier* $K = (S, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed sequence source* S*;*
2. *bound-ledger exactness for the row* B*, including the displayed endpoint reads and the finite evidence that inspected values stay within the bound;*
3. *sequence-window stability for every* $\mathsf{StreamName}^{\uparrow}$ *window in* W *and its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in* R*;*
4. *Real-seal nonescape, requiring every value and bound endpoint read to pass through the terminal* $\mathsf{Real}^{\uparrow}$ *row* E*;*
5. *downstream handoff through* H, C, P, N *before any* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer may inspect the bounded source.*

No accepted obligation is discharged by host sequence convergence, quotient equality of streams, a selected infinite subsequence, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 3451.1. The visible rows are exactly $S, B, W, R, E, H, C, P, N$, so carrier admission, bound evidence, windows, readback, seal, transport, replay, provenance, and naming exhaust the local certificate surface.

The bound row is read only through displayed windows. A request for an inspected sequence value first enters the source row S, then the finite $\mathsf{StreamName}^{\uparrow}$ window W, and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback R before the bound witness B can compare the value with its endpoints.

The terminal Real row is downstream of that finite route. Componentwise transport in H and displayed replay in C preserve the same rows, while P, N package and name the packet. Since the excluded host and compactness coordinates are absent, they cannot discharge any local NameCert obligation. □

Theorem 3451.3 (RealBoundedSequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier* $K = (S, B, W, R, E, H, C, P, N)$ *factors through the finite route*

$$S, B \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus sequence values and bounding endpoints reach the terminal Real seal only after the displayed source row, bound witness, finite StreamName window, and RegSeqRat readback have been exposed.

Proof. Start with a Real-facing read of a sequence value or bound endpoint. The carrier has no Real coordinate outside E, so the read must use the terminal seal row of Definition 3451.1.

Before E is visible, the requested value is displayed through S and the requested inspection window is displayed through W. The regular rational readback R then supplies the rational approximants against which the bound witness B is checked.

Transport and replay do not introduce a second Real seal. The rows H and C preserve the same finite route, and P, N source and name it for consumers. Hence a Real-facing consumer cannot bypass the sequence source, bound ledger, StreamName window, or RegSeqRat readback. □

Theorem 3451.4 (RealBoundedSequence monotone-subsequence boundary)**.** *The* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *consumer of chapter 1254 may read a bounded source from an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier only through the displayed route of Definition 3451.3. In particular, any monotone-subsequence or cluster-point boundary downstream receives the sequence source S, bound witness B, finite windows W, regular readback R, terminal Real seal E, transport H, replay C, provenance P, and local name N, but it receives no arbitrary subsequence selector, quotient stream equality, trichotomy principle, open-cover compactness proof, or ambient Real-completeness row.*

Proof. Read the bounded source demanded by the Bolzano–Weierstrass carrier. By Definition 3451.2, the only bounded-sequence surface available to that consumer is the finite package $S, B, W, R, E, H, C, P, N$.

Apply Definition 3451.3. The downstream route reaches $\mathsf{Real}^{\uparrow}$ only after the source sequence, bound witness, finite windows, and regular readback have been displayed, so the consumer cannot replace the packet with a host sequence or quotient stream.

The monotone-subsequence and cluster-point boundaries can therefore inherit only this bounded-source surface. Since the carrier contains no row for arbitrary subsequence selection, trichotomy, open-cover compactness, or ambient completeness, those principles remain outside the handoff. □

Theorem 3451.5 (RealBoundedSequence uniform-Cauchy subsequence boundary)**.** *Any uniform-Cauchy or subsequence-compactness consumer of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, E, H, C, P, N)$$

must read the finite bounded-source route through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows before it can export a subsequence or Cauchy-window conclusion. Thus the consumer receives $S, B, W, R, E, H, C, P, N$ as displayed packet data, but it receives no selected subsequence, quotient stream, trichotomy row, open-cover compactness proof, or ambient Real-completeness coordinate.*

Proof. Project the carrier of Definition 3451.1. The rows $S, B, W, R, E, H, C, P, N$ are the only displayed source, bound, window, readback, seal, transport, replay, provenance, and local naming coordinates.

Apply the NameCert obligations of Definition 3451.2. They force each bounded-sequence read to pass through the source row, bound witness, finite StreamName window, RegSeqRat readback, terminal Real seal, and structural support.

Use the Real-seal handoff of Definition 3451.3 and then the monotone-subsequence boundary of Definition 3451.4. The same route is the only input available to a uniform-Cauchy or subsequence-compactness consumer, so any exported subsequence or Cauchy-window conclusion is read through that finite packet.

Since no selected subsequence, quotient stream, trichotomy, open-cover compactness, or ambient Real-completeness row occurs in the carrier, none of those principles is exported by the boundary. □

Theorem 3451.6 (RealBoundedSequence compact-window handoff)**.** *Every compact-window, Bolzano-Weierstrass, or monotone-subsequence consumer of an accepted* $\mathsf{RealBoundedSequence}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, E, H, C, P, N)$$

must first read the displayed rows $S, B, W, R, E, H, C, P, N$ *through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *Only after that finite packet is exposed may the consumer pass the bounded-source data to* $\mathsf{BolzanoWeierstrass}^{\uparrow}$. *The handoff refuses selected subsequences, quotient stream equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 3451.1. The rows $S, B, W, R, E, H, C, P, N$ are the only sequence source, bound witness, finite window, regular readback, terminal Real seal, transport, replay, provenance, and local naming coordinates.

Apply the NameCert obligations of Definition 3451.2. They require a downstream bounded-source read to expose the StreamName windows, RegSeqRat readback, dyadic tolerance ledger, and terminal Real seal before the packet is available to any consumer.

Use the Real-seal handoff of Definition 3451.3. It prevents a compact-window route from bypassing W, R, E or replacing them by host sequence data.

Finally apply the monotone-subsequence boundary of Definition 3451.4. The same displayed packet is the only input available to $\mathsf{BolzanoWeierstrass}^{\uparrow}$ and to monotone-subsequence consumers, while the refused subsequence, quotient, trichotomy, open-cover, and completeness principles have no carrier row. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{RealBoundedSequence}^{\uparrow}$ packet is generated from a displayed sequence source, a bound witness row, finite StreamName windows, RegSeqRat readback, terminal Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3451.1, Definition 3451.2, Definition 3451.3, and Definition 3451.4 seed the finite bounded-sequence source consumed by Bolzano-Weierstrass and monotone-subsequence routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBoundedSequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealBoundedSequenceUp.RealBoundedSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim host sequence convergence, arbitrary subsequence choice, quotient stream equality, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bound-ledger exactness, sequence-window stability, Real-seal nonescape, and downstream handoff for BEDC.Derived.RealBoundedSequenceUp.

Chapter 3452

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ packet records the finite criterion by which a regular rational name is admitted to the completion-facing Real seal. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ rows of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus schedule of chapter 1438, the sequence-limit handoff of chapter 1413, the completion-admission boundary of chapter 1476, the functorial completion surface of chapter 3639, the recursor route of chapter 3628, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a displayed NameCert criterion for L10 completion consumers, not a quotient completion, selected representative, countable-choice schedule, host sequence theorem, trichotomy split, or ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Theorem 3452.1 (RegularCauchyCompletionCriterion carrier)**.** *A* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, M, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row naming the regular rational readback inspected by the completion criterion.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the source name is observed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested dyadic radius, slack cells, and comparison budget used by the regular readback.*
4. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule row naming the displayed tail threshold.*
5. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff row reached only after R, W, D, M have been displayed.*
6. *Q is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary row that admits the regular Cauchy route to the completion-facing* $\mathsf{Real}^{\uparrow}$ *seal.*
7. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, modulus schedule, sequence-limit handoff, completion boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from window requests through dyadic tolerance, regular readback, modulus schedule, sequence-limit handoff, and completion boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row naming the displayed criterion and its permitted completion-facing reads.*

The classifier compares accepted packets only through $R, W, D, M, L, Q, H, C, P, N$. It has no coordinate for quotient stream equality, a selected completion representative, countable choice over tail indices, host convergence, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier is the displayed L10 completion-neighbour packet. The first six rows record the analytic route from regular rational readback through finite StreamName windows, dyadic tolerance, modulus data, sequence-limit handoff, and completion admission.

The remaining rows are structural. They transport, replay, source, and name exactly that route, with the Bishop completion functor and completion recursor recorded only as provenance-facing consumers. No field remains for quotient completion, selected representatives, countable choice, host convergence, trichotomy, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionTasteGate`
`single carrier alignment`

Theorem 3452.2 (RegularCauchyCompletionCriterion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier $K = (R, W, D, M, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary, transport, replay, provenance, and local naming row;*
2. *finite-window exactness, requiring every regular Cauchy observation to read W before R or M is available;*
3. *dyadic tolerance exactness, requiring the comparison budget D to be read before the modulus schedule M, sequence-limit handoff L, or completion boundary Q is exposed;*
4. *completion-boundary handoff, requiring Q to remain downstream of R, W, D, M, L and upstream only of the structural rows H, C, P, N;*
5. *ledger refusal, excluding quotient completion, selected representatives, countable choice over thresholds, host sequence convergence, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is tuple projection from Definition 3452.1. The accepted packet displays exactly the regular readback R, finite windows W, dyadic ledger D, modulus schedule M, sequence-limit handoff L, completion boundary Q, and the structural rows H, C, P, N.

The analytic route first reads the finite windows W and the tolerance ledger D. The regular rational readback R and the convergence-modulus schedule M are therefore exposed only as displayed finite data.

The handoff rows L and Q are downstream of that finite route. The $\mathsf{RealSequenceLimit}^{\uparrow}$ row receives the displayed sequence-limit comparison, and the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ row records the completion-admission boundary before any $\mathsf{Real}^{\uparrow}$-facing consumer can read the seal.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no row for quotient completion, selected representatives, countable choice over thresholds, host convergence, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionNameCertObligat
↪ variant BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionCarrier
namecert obligations

Theorem 3452.3 (RegularCauchyCompletionCriterion Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *carrier* $K = (R, W, D, M, L, Q, H, C, P, N)$ *must first read the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{CauchyConvergenceModulus}^{\uparrow} \longrightarrow \mathsf{RealSequenceLimit}^{\uparrow} \longrightarrow \mathsf{C}$$

The handoff exposes only finite windows, dyadic tolerances, regular readback, modulus schedule, sequence-limit comparison, completion-admission boundary, componentwise transport, replay, provenance, and local naming.

Proof. Start with the carrier of Definition 3452.1. Its $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are the only source of inspected windows and tolerance data.

The regular readback and modulus rows consume those displayed data. Thus the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ row M cannot be replaced by a host sequence predicate or by an unbounded tail search.

Next the $\mathsf{RealSequenceLimit}^{\uparrow}$ handoff L and $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ boundary Q are read in order. The $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ and $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ rows can consume this boundary only as displayed completion data, so the completion-facing consumer reaches the $\mathsf{Real}^{\uparrow}$ seal only after the finite route has been displayed.

The structural rows H, C, P, N preserve and name the same handoff. Hence the exported surface is exactly the finite route in the statement. □

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionRealSealHandoff

Theorem 3452.4 (RegularCauchyCompletionCriterion ledger refusal)**.** *For every accepted* RegularCauchyCompletio *packet, the completion-facing ledger refuses quotient stream equality, quotient completion, selected representatives, countable choice over tail thresholds, host sequence convergence, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer requesting one of those objects must leave the displayed carrier route* $R, W, D, M, L, Q, H, C, P, N$*.*

Lean4 ✓ BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionLedgerRefusal

Proof. Inspect the carrier fields in Definition 3452.1. They record only the regular readback, finite windows, dyadic tolerance ledger, modulus schedule, sequence-limit handoff, completion boundary, completion functor and recursor provenance, transport, replay, provenance, and local naming rows.

The NameCert obligations of Definition 3452.2 make those rows exhaustive. No additional row stores a quotient, selected representative, choice schedule, host convergence predicate, trichotomy split, host equality proof, or ambient completeness witness.

Therefore any such request cannot be answered by the packet. It is refused at the ledger boundary before a $\mathsf{Real}^{\uparrow}$-facing consumer can treat the displayed completion criterion as a quotient completion theorem. □

Theorem 3452.5 (RegularCauchyCompletionCriterion transport stability)**.** *Let $K = (R, W, D, M, L, Q, H, C, P, N$ and $K' = (R', W', D', M', L', Q', H', C', P', N')$ be accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *packets. If componentwise* hsame *transport identifies the rows $R, W, D, M, L, Q, H, C, P, N$ with $R', W', D', M', L', Q', H$ then the finite-window route, dyadic tolerance ledger, modulus schedule,* $\mathsf{RealSequenceLimit}^{\uparrow}$ *handoff,* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *boundary, replay, provenance, and local* NameCert *row are preserved. The transported packet still refuses quotient completion, selected representatives, countable choice over thresholds, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project both carriers through Definition 3452.1. The assumed componentwise hsame transport aligns exactly the displayed rows $R, W, D, M, L, Q, H, C, P, N$ and their primed counterparts.

Apply the NameCert obligations of Definition 3452.2. Finite-window exactness, dyadic tolerance exactness, and completion-boundary handoff are all phrased in those rows, so the transport cannot move the route outside the displayed packet.

Finally use the Real-seal handoff and ledger refusal theorems, Definition 3452.3 and Definition 3452.4. They identify the terminal read and the refused coordinates, so the transported packet preserves both the completion-facing route and the same boundary exclusions. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionTransportStabil`

Theorem 3452.6 (RegularCauchyCompletionCriterion obligation closure package)**.** *The carrier, NameCert obligations, Real-seal handoff, ledger refusal, and transport stability results for* RegularCauchyCompletionC *assemble the obligation-closure surface over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, CauchyConvergenceModulu $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The surface closes exactly the finite criterion rows $R, W, D, M, L, Q, H, C, P, N$ and does not claim quotient completion, selected representatives, countable choice, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionObligationClosu`

Proof. Start with the carrier and obligation rows in Definition 3452.1 and Definition 3452.2. They expose every analytic and structural coordinate of the packet and make the local NameCert surface exhaustive.

Next apply the Real-seal handoff and ledger refusal theorems. The handoff theorem routes any completion-facing read through the displayed StreamName, DyadicRatCore, RegSeqRat, modulus, sequence-limit, and completion-boundary rows, while the refusal theorem removes quotient, choice, host, and ambient-completeness exits from the package.

Finally apply transport stability from Definition 3452.5. Since componentwise hsame preserves the displayed rows and their replay and provenance data, the package is stable as an obligation-closure surface over the named BEDC primitives. □

Theorem 3452.7 (RegularCauchyCompletionCriterion scoped factorization)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *packet factors through the displayed rows of Definition 3452.1 before it may export a local* NameCert *surface. Concretely, the read must pass through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionC}}$ *row.*

The factorization is scoped to those rows. It does not supply quotient completion, selected representatives, countable choice over thresholds, host sequence convergence, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier and NameCert obligations of Definition 3452.1 and Definition 3452.2. They make the rows $R, W, D, M, L, Q, H, C, P, N$ exhaustive for any accepted packet.

Apply the Real-seal handoff of Definition 3452.3. It orders the completion-facing route through StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, convergence modulus, sequence-limit handoff, completion boundary, completion functor, completion recursor, and the terminal Real seal before a consumer can read completion data.

The ledger refusal of Definition 3452.4 removes quotient, choice, host, trichotomy, and ambient-completeness exits from that route. Transport stability in Definition 3452.5 then preserves the same finite rows under componentwise hsame and displayed Cont replay.

Finally use the obligation-closure package of Definition 3452.6. It repacks the displayed route with Pkg provenance and the local NameCert row, so every completion-facing read factors through exactly the named scope. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionScopeFactorizat`

Theorem 3452.8 (RegularCauchyCompletionCriterion public export)**.** *Every accepted* RegularCauchyCompletionC *packet exports exactly the public completion-criterion surface*

$$R, W, D, M, L, Q, H, C, P, N$$

to Real-facing consumers: regular rational readback, finite StreamName windows, dyadic tolerance ledger, convergence-modulus schedule, RealSequenceLimit handoff, CauchyCompletionCriterion boundary, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row. It exports no quotient completion, selected representative, countable choice, trichotomy, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge certificate.*

Proof. Start with the carrier of Definition 3452.1. It exposes exactly the rows $R, W, D, M, L, Q, H, C, P, N$ and has no additional public coordinate for a quotient, representative selector, choice schedule, trichotomy split, host equality proof, ambient completeness witness, or bridge certificate.

Apply the NameCert obligations of Definition 3452.2. They force every accepted read to pass through the finite-window exactness row, dyadic tolerance exactness row, completion-boundary handoff, structural transport, replay, provenance, and local naming row.

The Real-seal handoff of Definition 3452.3 identifies the Real-facing order of the exported rows. The ledger refusal and transport stability theorems, Definition 3452.4 and Definition 3452.5, remove the forbidden exits and preserve the same public surface under componentwise hsame.

Finally use the obligation package and scoped factorization of Definition 3452.6 and Definition 3452.7. They repack the carrier, obligations, Real-seal handoff, ledger refusal, and transport stability into the displayed completion-criterion route, so the public export is exactly $R, W, D, M, L, Q, H, C, P, N$. □

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionPublicExport`

Theorem 3452.9 (RegularCauchyCompletionCriterion public export checked route)**.** *The public export of Definition 3452.8 is exactly the already-scoped finite route through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BishopCauchyCom $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}}$ *row. In particular, every public Real-facing read factors through the same rows* $R, W, D, M, L, Q, H, C, P, N$ *that support the carrier, NameCert obligations, Real-seal handoff, ledger refusal, transport stability, obligation package, and scoped factorization theorems.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCriterionUp.RegularCauchyCompletionCriterionPublicExportChe`

Proof. Start with the carrier and NameCert obligations of Definition 3452.1 and Definition 3452.2. They identify the finite rows $R, W, D, M, L, Q, H, C, P, N$ and make the local certificate surface exhaustive.

Apply the Real-seal handoff, ledger refusal, and transport stability theorems, Definition 3452.3, Definition 3452.4, and Definition 3452.5. These results order the public read through the RegSeqRat, StreamName, DyadicRatCore, convergence-modulus, sequence-limit, completion-boundary, completion-functor, completion-recursor, and Real rows while excluding quotient, choice, host, trichotomy, and ambient-completeness exits.

Finally use the obligation package and scoped factorization of Definition 3452.6 and Definition 3452.7. They repack the same route with componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row, so the public export theorem is precisely the checked finite route already isolated by the scoped surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ packet is generated from a RegSeqRat readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a CauchyConvergenceModulus schedule, a RealSequenceLimit handoff, a CauchyCompletionCriterion boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}, \mathsf{scopedClosed})$

Scope closed: Definition 3452.1, Definition 3452.2, Definition 3452.3, Definition 3452.4, Definition 3452.5, Definition 3452.6, and Definition 3452.7 bind the finite regular-Cauchy completion criterion route through RegSeqRat, StreamName, DyadicRatCore, CauchyConvergenceModulus, RealSequenceLimit, CauchyCompletionCriterion, BishopCauchyCompletionFunctor, CauchyCompletionRecursor, Real rows, hsame transport, Cont replay, Pkg provenance, and local NameCert transport before any completion-facing consumer reads the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCriterion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, host sequence convergence, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Public closure requires a checked export theorem for the same RegSeqRat, StreamName, DyadicRatCore, modulus, sequence-limit, completion-boundary, recursor, Real, transport, replay, provenance, and local NameCert primitives.

Chapter 3453

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}$ packet records the finite modulus realizer that turns regular rational Cauchy readback into a completion-facing $\mathsf{Real}^{\uparrow}$ seal. It sits between the $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ route of chapter 3452, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{BishopCompletionCriterion}^{\uparrow}$ handoff of chapter 3404, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is finite modulus-realizer data, not a quotient completion, representative choice, omniscient tail search, host convergence theorem, order trichotomy, or ambient completeness claim.

Definition 3453.1 (RegularCauchyCompletionModulusRealizer carrier)**.** *A* $\mathsf{RegularCauchyCompletionModulusReal}$ *carrier is a finite* BHist *packet*

$$R = (Q, W, D, M, K, B, E, H, C, P, N)$$

with the following rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the regular rational sequence inspected by the realizer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which observations of Q are requested.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row storing dyadic radii, slack cells, and comparison budget.*
4. *M is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *schedule row that chooses finite tail thresholds from D and W.*
5. *K is the* $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ *admission row certifying that Q, W, D, M form the completion-facing regular Cauchy route.*
6. *B is the* $\mathsf{BishopCompletionCriterion}^{\uparrow}$ *handoff row exposing the accepted criterion to Bishop completion consumers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, W, D, M, K, B have been displayed.*
8. *H records componentwise* hsame *transport for the readback, window, tolerance, modulus, criterion, handoff, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from window requests through dyadic tolerance, modulus scheduling, criterion admission, Bishop handoff, and the terminal real seal.*

10. *P is the* Pkg *provenance over* RegularCauchyCompletionModulusRealizer$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, RegularCauchyCompletionCriterion$^{\uparrow}$, BishopCompletionCriterion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyCompletionModulusRealizer}^{\uparrow}}$ *row naming the finite realizer and its permitted completion-facing reads.*

The classifier compares accepted packets only through $Q, W, D, M, K, B, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice over tail thresholds, host sequence convergence, trichotomy, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyCompletionModulusRealizerUp`

Theorem 3453.2 (RegularCauchyCompletionModulusRealizer NameCert obligations)**.** *For every accepted* RegularCauchyCompletionModulusRealizer$^{\uparrow}$ *carrier $R = (Q, W, D, M, K, B, E, H, C, P, N)$, the local* NameCert$_{\text{RegularCauchyCompletionModulusRealizer}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* RegSeqRat$^{\uparrow}$ *readback,* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* CauchyConvergenceModulus$^{\uparrow}$ *schedule,* RegularCauchyCompletionCriterion$^{\uparrow}$ *admission row,* BishopCompletionCriterion$^{\uparrow}$ *handoff, terminal* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*

2. *window-to-tolerance exactness, requiring W and D to be read before any modulus threshold in M is exposed;*

3. *modulus-to-criterion exactness, requiring the finite schedule M to remain upstream of the* RegularCauchyCompleti *admission row K;*

4. *criterion-to-seal handoff, requiring K and B to be read before the terminal* Real$^{\uparrow}$ *row E is available to a completion consumer;*

5. *ledger refusal, excluding quotient completion, representative selection, countable choice over tail indices, host convergence, trichotomy, host real equality, and ambient* Real$^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is tuple projection from Definition 3453.1. The accepted packet displays the regular rational readback Q, finite windows W, dyadic tolerance D, modulus schedule M, criterion row K, Bishop handoff B, terminal real seal E, and the structural rows H, C, P, N.

The exactness clauses follow the displayed dependency order. A threshold in M is meaningful only after the finite window and dyadic tolerance rows have been read. The criterion row K is then downstream of Q, W, D, M, and the Bishop handoff B cannot expose a terminal Real$^{\uparrow}$ seal before K is available.

The structural rows transport, replay, source, and name the same finite route. Since the carrier contains no row for quotient completion, selected representatives, countable choice, host convergence, trichotomy, host real equality, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 3453.3 (RegularCauchyCompletionModulusRealizer Real-seal handoff)**.** *Every* Real$^{\uparrow}$*-facing consumer of an accepted* RegularCauchyCompletionModulusRealizer$^{\uparrow}$ *carrier $R = (Q, W, D, M, K, B, E, H, C,$ must first read the route*

StreamName$^{\uparrow}$ $\longrightarrow$ DyadicRatCore$^{\uparrow}$ $\longrightarrow$ RegSeqRat$^{\uparrow}$ $\longrightarrow$ CauchyConvergenceModulus$^{\uparrow}$ $\longrightarrow$ RegularCauchyCompletion

The handoff exposes only finite windows, dyadic tolerances, regular readback, modulus scheduling, criterion admission, Bishop completion handoff, componentwise transport, displayed replay, provenance, and local naming.

Proof. Start with the carrier of Definition 3453.1. Its $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are the only source of inspected windows and tolerance data, and the $\mathsf{RegSeqRat}^{\uparrow}$ row is the only regular rational readback exposed to the schedule.

The modulus row M consumes those displayed data before any completion criterion can be read. Thus the $\mathsf{RegularCauchyCompletionCriterion}^{\uparrow}$ row K records admission for a finite route, not for an unbounded tail search or a host sequence theorem.

The Bishop handoff B receives only that admitted criterion. A $\mathsf{Real}^{\uparrow}$-facing consumer therefore reaches the terminal row E only after the finite modulus-realizer route has been displayed, transported, replayed, sourced, and named by H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}$ packet is generated from a RegSeqRat readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a CauchyConvergenceModulus schedule, a RegularCauchyCompletionCriterion admission row, a BishopCompletionCriterion handoff, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3453.1, Definition 3453.2, and Definition 3453.3 seed the finite regular-Cauchy completion modulus-realizer route through RegSeqRat, StreamName, DyadicRatCore, CauchyConvergenceModulus, RegularCauchyCompletionCriterion, BishopCompletionCriterion, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionModulusRealizer}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyCompletionModulusRealizerUp`

Bridge status: none

Not claimed: This chapter does not claim quotient completion, representative choice, countable choice, host sequence convergence, trichotomy, host real equality, ambient RealUp completeness, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-to-tolerance exactness, modulus-to-criterion exactness, criterion-to-seal handoff, transport stability, replay stability, and ledger refusal for BEDC.Derived.RegularCauchyCompletion

Chapter 3454

A Concrete Naming Certificate for $\mathsf{ConstructiveDiniModulus}^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ packet records the finite chain by which a monotone sequence of displayed real-valued functions on a compact metric source receives one uniform tolerance modulus. It sits after the compact finite-net route of chapter 129, the pointwise continuity surface of chapter 130, the compact uniform-modulus handoff of chapter 3444, the real-name seal of chapter 38, the finite-window readbacks of chapter 301 and chapter 126, and the dyadic tolerance ledger of chapter 291. The chapter names the finite BEDC route from pointwise monotone convergence data to a displayed Dini modulus without importing open-cover compactness, selected subcovers, choice, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3454.1 (Constructive Dini modulus carrier)**.** *A* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, L, N, Q, W, S, R, E, G, M, H, C, P, A)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the finite-net rows used by the modulus extraction.*
2. *F is the monotone real-function sequence row, listing the inspected function names, their pointwise order comparisons, and their target* $\mathsf{Real}^{\uparrow}$ *value rows.*
3. *L is the pointwise-limit row, naming the displayed limit function and the finite pointwise tail requests that compare $F_n(x)$ with $L(x)$.*
4. *N is the compact finite-net row selected from K for the current tolerance.*
5. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *continuity ledger for the finitely many functions and the limit row inspected at the net centers.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for function values, limit values, and net-center observations.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W.*
8. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested tolerance, the smaller comparison tolerances, and the finite dyadic descent used by the chain.*

9. *E is the finite chain comparison row. It lists the net-center tail index, monotonicity step, continuity transfer, and target estimate used for each displayed center.*
10. *G is the global modulus row obtained by taking the finite maximum of the center indices recorded in E.*
11. *M is the exported Dini-modulus witness for the monotone sequence and limit row.*
12. *H records componentwise* hsame *transport for $K, F, L, N, Q, W, S, R, E, G, M$ and the administrative rows.*
13. *C records displayed* Cont *replay from compact finite-net data through pointwise-limit windows, dyadic tolerances, finite chain comparisons, and the exported modulus.*
14. *P is the* Pkg *provenance over* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *A is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$ *row.*

The carrier exposes only monotone real-function sequence rows, pointwise-limit rows, compact finite-net rows, finite chain comparison rows, dyadic tolerance ledgers, regular rational and stream-name finite windows, componentwise transport, replay, provenance, and local naming. It has no coordinate for open-cover compactness, selected subcovers, choice, quotient equality, trichotomy, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusCarrier`
Lean4 ✓ `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusTasteGate single carrier alignment`

Theorem 3454.2 (Constructive Dini finite-chain modulus)**.** *For every accepted* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier*

$$D = (K, F, L, N, Q, W, S, R, E, G, M, H, C, P, A),$$

the exported row M is a finite Dini-modulus witness for the monotone sequence row F and the pointwise-limit row L at the tolerance recorded by R. Every consumer of M must factor through the displayed route

$$K, F, L, R \longmapsto N, Q, W, S, E \longmapsto G, M \longmapsto H, C, P, A.$$

Thus compact finite nets, pointwise limit windows, $\mathsf{ContinuousMap}^{\uparrow}$ *continuity ledgers,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance descent,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readbacks, finite chain comparisons, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *ledger suffice to export the constructive Dini modulus.*

Proof. Project the carrier of Definition 3454.1. The tolerance ledger R determines the comparison scale, while K supplies the finite net row N at the source scale consumed by the chain comparison.

At each center of N, read the pointwise-limit row L through the finite windows W and the regular rational readback S. The monotone function sequence row F supplies the ordered tail comparisons, and Q supplies the continuity rows needed to move from centers to neighboring source points.

The chain row E records the only comparison route used by the packet. It combines the center tail estimate, monotonicity, continuity transfer, and the dyadic tolerance descent of R into a displayed estimate at the requested tolerance.

Take the finite maximum of the center indices in E to obtain G, and read M as the exported modulus row. Componentwise transport in H preserves the compact source, function sequence, limit, window, tolerance, comparison, and modulus rows, while C replays the same route for downstream consumers.

Finally P and A record provenance and local naming. Since no carrier coordinate contains an open cover, selected subcover, choice functional, quotient equality, or ambient completeness principle, no consumer can extract any of those stronger readings from the modulus witness. □

Theorem 3454.3 (Constructive Dini modulus obligation lattice)**.** *Every exported modulus row of an accepted* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *carrier is read in the dependency order*

$$K, N \longmapsto L, W \longmapsto S \longmapsto R \longmapsto E \longmapsto G \longmapsto A.$$

In this order, K, N *expose the compact finite-net source,* L, W *expose pointwise limit windows,* S *exposes the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* R *exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance descent,* E *records the finite comparison chain,* G *records the finite maximum row, and* A *supplies the local* $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$ *surface. No consumer may read the exported modulus* M *before these rows have been displayed.*

Proof. Project the carrier of Definition 3454.1. The rows K, N provide the compact finite-net source before any pointwise or tolerance comparison is available.

Apply Definition 3454.2. The pointwise limit row L is inspected through the finite windows W, and those windows are read back through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ row S.

The dyadic tolerance ledger R then fixes the descent used by the finite comparison chain E. Only after the centerwise indices in E have been displayed can the finite maximum row G be read.

Finally H, C, P, A transport, replay, source, and name the same route. The exported row M is therefore downstream of the compact finite net, pointwise windows, stream and regular-rational readback, dyadic descent, finite chain, finite maximum, and local NameCert rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, monotone real-function sequence rows, pointwise-limit rows, ContinuousMap continuity ledgers, StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, finite chain comparison rows, a finite maximum row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3454.1 and Definition 3454.2 seed the finite BEDC route from monotone pointwise convergence on compact finite-net data to a displayed Dini modulus.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstructiveDiniModulus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveDiniModulusUp.ConstructiveDiniModulusTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, choice, quotient equality, host function extensionality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone sequence visibility, pointwise-limit windows, compact finite-net extraction, finite chain comparison, dyadic tolerance descent, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveDiniModulusUp.

Chapter 3455

A Concrete Naming Certificate for $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$

The $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ packet realizes the Poincare disk reading of chapter 9463 as finite BEDC data. It reads visible phase coordinates through chapter 39, disk coordinates and boundary rows through chapter 377, reversible transport through chapter 219, flow-style evolution through chapter 195, and boundary-circle consumers through chapter 58. The packet is a representation carrier over displayed histories, transports, flows, and provenance; it is not a host hyperbolic plane, not a global observer clock, not a physical cosmology, and not an internal point named T.

Definition 3455.1 (Poincare disk phase-transport carrier)**.** *A* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N)$$

with the following displayed rows:

1. *L is the focused finite multi-*Hist *ledger whose visible records are being represented.*
2. *Z is the* $\mathsf{Complex}^{\uparrow}$ *phase-coordinate row used for displayed amplitude, direction, or phase readback.*
3. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row, including the finite product-metric radius ledger and the optional* $\mathsf{S}^{1\uparrow}$ *boundary consumer row.*
4. *B is the boundary-far-end row, read only as a representation boundary over the disk packet and never as an interior object.*
5. *Γ is the* $\mathsf{UnitaryGroup}^{\uparrow}$ *reversible-transport row, recording classifier-preserving movement of the visible phase or disk surface.*
6. *V is the* $\mathsf{DynSystem}^{\uparrow}$ *flow row for finite evolution of the displayed disk-coordinate records.*
7. *F is the fiber and provenance row, retaining the hidden-side source distinctions behind each visible disk or phase readout.*
8. *O is the observer-focus row tying the disk surface back to the local ledger L rather than to a global observer or global clock.*
9. *A is the audit-refusal row excluding internal* T*, host hyperbolic geometry, physical light-speed derivation, global universe digest, and subject-substance readings.*

10. *H records componentwise* hsame *transport for the ledger, phase, disk, boundary, reversible-transport, flow, fiber, focus, and audit rows.*
11. *C records displayed* Cont *replay through finite disk coordinates, reversible transport, flow read-back, fiber provenance, and refusal rows.*
12. *R is the* Pkg *provenance over* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}^{\uparrow}}$ *row.*

The carrier compares packets only through these rows. It contains no coordinate for a completed Poincare disk, no host hyperbolic metric, no quotient boundary, no global physical evolution law, no subject-substance carrier, and no object-level T *inside the disk.*

Theorem 3455.2 (Poincare disk phase-transport ledger)**.** *For every accepted* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *carrier*

$$P = (L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N),$$

every disk-phase transport read factors through the finite route

$$L, Z, U \longmapsto B, \Gamma, V, F, O, A \longmapsto H, C, R, N.$$

The route exports visible disk-coordinate, reversible-transport, flow, fiber-provenance, observer-focus, and refusal data, while refusing host hyperbolic geometry, quotient boundary equality, global observer time, physical light-speed derivation, internal T*, and subject-substance readings.*

Lean4 ✓ `BEDC.Derived.PoincareDiskPhaseTransportUp.PoincareDiskPhaseTransportTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3455.1. The visible surface begins with the local ledger L, the complex phase row Z, and the disk-coordinate row U.

The representation rows are then read in order. The boundary row B records a far-end representation boundary, Γ supplies reversible classifier-preserving transport, V supplies finite flow-style evolution, F retains the fiber provenance behind the readout, O localizes the surface to the focused ledger, and A records the refused readings.

Transport and replay do not add new geometry. The rows H and C preserve and replay the same finite coordinates, while R and N record provenance and local naming.

Since the packet has no coordinate for a completed disk geometry, quotient boundary, global clock, physical constant, subject-substance, or internal T, no consumer of the ledger can export those stronger readings. □

Falsifiable prediction: Removing the fiber-provenance row or the audit-refusal row lets a disk-phase consumer identify visible disk-coordinate equality with source equality or with an internal T reading, so the packet is no longer accepted.

Independence witness: complex, unitdisk, unitarygroup, dynsystem, s1: none is bijective to $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ because the phase packet jointly carries visible complex coordinates, disk coordinates, optional boundary consumption, reversible transport, finite flow, fiber provenance, observer focus, refusal ledger, transport, replay, provenance, and local naming.

The Lean follow-up for this packet must provide explicit Nontrivial $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ and FieldFaithful $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ TasteGate instances. Those instances must keep

$L, Z, U, B, \Gamma, V, F, O, A, H, C, R, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}$.

Constructive story: A $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ packet is generated from a focused finite multi-Hist ledger, Complex phase coordinates, UnitDisk disk coordinates with optional SOne boundary consumption, UnitaryGroup reversible transport, DynSystem finite flow rows, fiber provenance, observer focus, refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3455.1 and Definition 3455.2 seed the finite disk-coordinate, reversible-transport, flow, fiber-provenance, and refusal route for the hyperbolic phase representation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskPhaseTransport}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Poincare disk geometry, host hyperbolic metric, quotient boundary equality, global observer time, physical light-speed derivation, internal T, subject-substance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-coordinate visibility, reversible transport, flow replay, fiber-provenance preservation, refusal exactness, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.PoincareDiskPhaseTransportUp.

Chapter 3456

A Concrete Naming Certificate for $\mathsf{MaschkeTheorem}^{\uparrow}$

Following Maschke's theorem for finite-group representations, the $\mathsf{MaschkeTheorem}^{\uparrow}$ packet records a finite averaging-projector and complement-decomposition ledger. It reads the group source of chapter 41, the vector-space source of chapter 47, the character-theory and representation-ring rows of chapter 245 and chapter 246, and the orthogonality surface of chapter 1603. The packet names only the displayed finite semisimplicity route; it is not a host representation category, not a quotient-subrepresentation construction, and not a global semisimplicity theorem.

Definition 3456.1 (Maschke theorem carrier)**.** *A* $\mathsf{MaschkeTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, V, R, X, A, Q, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Group}^{\uparrow}$ *source row supplying the group operation, unit, inverse, and finite averaging support;*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row supplying the represented vector carrier and its addition, scalar, and zero rows;*
3. *R is the* $\mathsf{RepresentationRing}^{\uparrow}$ *row recording the representation input and the direct-sum handoff surface;*
4. *X is the invariant-subspace row, displayed as a carried subobject of the represented vector row rather than as a quotient representation;*
5. *A is the averaging-projector row, storing the finite group average used to make a raw complement map invariant;*
6. *Q is the complement row, recording the kernel-image decomposition exposed by the averaged projector;*
7. *H is the componentwise* hsame *transport for G, V, R, X, A, Q;*
8. *C is the* Cont *replay row for the averaging and complement ledgers;*
9. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$ *evidence.*

The carrier has no coordinate for a host representation category, a quotient subrepresentation carrier, a selected basis, or a global semisimplicity theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.MaschkeTheoremUp.MaschkeTheoremTasteGate single carrier alignment`

Theorem 3456.2 (Maschke theorem NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$ *surface for the carrier of Definition 3456.1 consists of the finite obligations that admit the group source* G*, the vector source* V*, the representation row* R*, the invariant-subspace row* X*, the averaging projector* A*, the complement row* Q*, and the transport, replay, provenance, and local naming rows* H, C, P, N*. These obligations export no host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem outside* $G, V, R, X, A, Q, H, C, P, N$*.*

Proof. Project the carrier rows of Definition 3456.1. The group row is read through chapter 41, while the vector row is read through chapter 47.

Next read the representation surface through chapter 246 and the character-theory support through chapter 245. These rows account for R and for the finite representation-theoretic evidence consumed by the averaging ledger.

Finally inspect the remaining rows. The invariant subspace X, averaging projector A, and complement row Q are preserved only by the displayed hsame, Cont, Pkg, and NameCert rows H, C, P, N. Since the carrier contains no further coordinate, the excluded host category, quotient carrier, basis selection, and global semisimplicity data cannot be exported. □

Theorem 3456.3 (Maschke averaging-projector route)**.** *Every complement-facing consumer of an accepted* $\mathsf{MaschkeTheorem}^{\uparrow}$ *packet must pass through*

$$G \longmapsto V \longmapsto R \longmapsto X \longmapsto A \longmapsto Q \longmapsto H, C, P, N.$$

The averaging row A *is read as a finite* $\mathsf{Group}^{\uparrow}$ *average over the represented* $\mathsf{VecSpace}^{\uparrow}$ *row and the representation row* R*, and the complement row* Q *is read only after that projector row has been displayed. The route exports no host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem.*

Proof. Begin with the group and vector sources. The finite averaging support is supplied by chapter 41, and the linear target of the average is supplied by chapter 47.

Then use the representation rows. The representation-ring certificate of chapter 246 identifies the action row R, while the character-theory and Schur-orthogonality surfaces of chapter 245 and chapter 1603 supply the finite representation-theoretic ledger read by the projector route.

The complement row is downstream of the averaging row. Once A is displayed, Q records only the finite complement decomposition and the rows H, C, P, N replay and name that route. No listed sibling chapter supplies a hidden host category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem, so the consumer cannot receive those objects from this packet. □

Theorem 3456.4 (Maschke representation-ring handoff)**.** *Every* $\mathsf{RepresentationRing}^{\uparrow}$*-facing read of an accepted* $\mathsf{MaschkeTheorem}^{\uparrow}$ *packet receives only the finite handoff from the group and vector rows to the representation row* R*, the invariant row* X*, the averaging row* A*, the complement row* Q*, and the structural rows* H, C, P, N*. The handoff may record a displayed direct-sum decomposition row, but it does not create a host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem outside the finite packet.*

Proof. First expose the representation-ring dependency. The handoff begins at chapter 246, whose rows name representation addition and direct-sum bookkeeping over the underlying group and vector surfaces.

Next attach the finite Maschke rows. The invariant row X and the averaging projector row A produce the complement row Q only through the displayed packet route, with chapter 1603 available as a finite orthogonality-side ledger rather than as a basis selector.

Finally repack the handoff through H, C, P, N. These rows transport, replay, source, and name the same finite route for $\mathsf{RepresentationRing}^{\uparrow}$ consumers. Since no quotient carrier, host category, selected basis, or global semisimplicity coordinate is present, the handoff remains confined to the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MaschkeTheorem}^{\uparrow}$ packet is generated from a finite $\mathsf{Group}^{\uparrow}$ source row, a $\mathsf{VecSpace}^{\uparrow}$ source row, a $\mathsf{RepresentationRing}^{\uparrow}$ row, an invariant-subspace row, an averaging-projector row, a complement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3456.1, Definition 3456.2, Definition 3456.3, and Definition 3456.4 seed the finite averaging-projector and complement-decomposition route for finite-group representation consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaschkeTheorem}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MaschkeTheoremUp.MaschkeTheoremTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close a host representation category, quotient subrepresentation carrier, selected basis, or global semisimplicity theorem.

Upgrade path: Obligation closure requires checked group, vector, representation-ring, invariant-subspace, averaging-projector, complement, transport, replay, provenance, and local naming rows for the displayed packet.

Chapter 3457

A Concrete Naming Certificate for $\mathsf{MaschkeProjector}^{\uparrow}$

Following Maschke's theorem for finite-group representations, the $\mathsf{MaschkeProjector}^{\uparrow}$ packet isolates the finite averaging-projector construction from the semisimplicity packet of chapter 3456. It reads the finite group source of chapter 41, the represented vector source of chapter 47, the representation-row bookkeeping of chapter 246, and the orthogonality-side ledger of chapter 1603. The packet names the projector and invariant-complement route only; it does not close the full $\mathsf{MaschkeTheorem}^{\uparrow}$ semisimplicity certificate.

Definition 3457.1 (Maschke projector carrier)**.** *A* $\mathsf{MaschkeProjector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, V, R, S, A, I, Q, H, C, K, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{Group}^{\uparrow}$ *row supplying the finite averaging support, unit, inverse, and multiplication checks;*
2. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row supplying the represented vector carrier, zero, addition, and scalar-action rows;*
3. *R is the* $\mathsf{RepresentationRing}^{\uparrow}$ *row naming the representation action consumed by the average;*
4. *S is the invariant-subspace row whose inclusion into V is the target of the complement construction;*
5. *A is the averaging row that stores the finite group average of a raw projection along the representation action;*
6. *I is the invariance row recording that the averaged map commutes with the displayed group action;*
7. *Q is the complement row exposing the kernel-image decomposition read by downstream invariant-complement consumers;*
8. *H records componentwise* hsame *transport for G, V, R, S, A, I, Q;*
9. *C records displayed* Cont *replay for the averaging, invariance, and complement routes;*
10. *K is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, BHist, hsame, Cont, *and local naming;*

11. N is the local $\mathsf{NameCert}_{\mathsf{MaschkeProjector}^\uparrow}$ *row.*

The carrier has no coordinate for a host representation category, a quotient-subrepresentation carrier, a selected basis, a global semisimplicity theorem, or the full $\mathsf{MaschkeTheorem}^\uparrow$ *packet.*

Lean4 ✓ `BEDC.Derived.MaschkeProjectorUp.MaschkeProjectorTasteGate single carrier alignment`

Theorem 3457.2 (Maschke projector NameCert obligations)**.** *For every accepted* $\mathsf{MaschkeProjector}^\uparrow$ *carrier*

$$P = (G, V, R, S, A, I, Q, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{MaschkeProjector}^\uparrow}$ *surface consists exactly of finite group admission through G, vector-source admission through V, representation-row admission through R, invariant-subspace admission through S, averaging-row exactness through A, invariance exposure through I, complement exposure through Q, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through K, and local naming through N.*

Proof. Project the carrier of Definition 3457.1. The group, vector, representation, invariant-subspace, averaging, invariance, and complement rows are exactly the analytic coordinates of the packet.

The structural rows H, C, K, N preserve the same finite surface. They transport, replay, source, and name the projector data without adding a host category or quotient representation.

Since the full $\mathsf{MaschkeTheorem}^\uparrow$ packet is not a coordinate of the carrier, the local NameCert obligations stop at the averaging projector and its invariant-complement route. □

Theorem 3457.3 (Maschke projector invariant-complement route)**.** *Every invariant-complement consumer of an accepted* $\mathsf{MaschkeProjector}^\uparrow$ *packet must read the route*

$$\mathsf{Group}^\uparrow \longmapsto \mathsf{VecSpace}^\uparrow \longmapsto \mathsf{RepresentationRing}^\uparrow \longmapsto A \longmapsto I \longmapsto Q.$$

The averaging row A is the only row that may turn a raw projection into an invariant projector, the invariance row I is the only row that may expose equivariance of that projector, and the complement row Q is the only row that may export the resulting invariant complement.

Proof. Begin with the finite group and vector rows. They supply the finite averaging support and the represented target on which the average is computed.

Next read the representation row. The action data from $\mathsf{RepresentationRing}^\uparrow$ is what lets the averaging row A replay the group average as an internal row rather than as an external operator.

The invariance row I is downstream of A, and the complement row Q is downstream of I. The Schur-orthogonality sibling may support the finite representation ledger, but it does not add a basis selector or a hidden semisimplicity theorem.

Finally transport the route through H, C, K, N. These rows keep the invariant-complement consumer inside the displayed $\mathsf{MaschkeProjector}^\uparrow$ packet and prevent a read of the larger $\mathsf{MaschkeTheorem}^\uparrow$ certificate. □

Falsifiable prediction: A consumer that exports an invariant complement from finite-group representation data without reading the averaging row A, invariance row I, complement row Q, hsame, Cont, Pkg, and NameCert rows refutes the $\mathsf{MaschkeProjector}^\uparrow$ carrier.

Independence witness: maschke-theorem, group, vecspace, representationring, schurorthogonality: $\mathsf{MaschkeProjector}^\uparrow$ binds the averaging projector, invariance exposure, and invariant-complement route in one finite NameCert packet; each named sibling lacks at least one of these rows as a local carrier obligation or carries a strictly larger semisimplicity surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MaschkeProjector}^{\uparrow}}$.

Constructive story: A $\mathsf{MaschkeProjector}^{\uparrow}$ packet is generated from a finite $\mathsf{Group}^{\uparrow}$ source row, a $\mathsf{VecSpace}^{\uparrow}$ source row, a $\mathsf{RepresentationRing}^{\uparrow}$ action row, an invariant-subspace row, an averaging row, an invariance row, a complement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaschkeProjector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3457.1, Definition 3457.2, and Definition 3457.3 seed the reusable finite averaging-projector certificate for downstream representation-theory packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaschkeProjector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full $\mathsf{MaschkeTheorem}^{\uparrow}$ packet, a host representation category, quotient-subrepresentation carrier, selected basis, global semisimplicity theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite group, vector, representation, invariant-subspace, averaging, invariance, complement, transport, replay, provenance, and local naming rows for BEDC.Derived.MaschkeProjectorUp.

Chapter 3458

A Concrete Naming Certificate for $\mathsf{NestedCauchyRealSection}^{\uparrow}$

Bishop–Bridges constructive analysis treats nested Cauchy sections as finite approximation data before any completed real object is read. The $\mathsf{NestedCauchyRealSection}^{\uparrow}$ packet names that Real and Completion phase surface over the nested-interval carrier of chapter 320, the dyadic tolerance ledger of chapter 291, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It packages nested Cauchy section data for downstream completion consumers without asserting completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3458.1 (Nested Cauchy real section carrier)**.** *A* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval row, containing the finite interval chain, endpoint order data, and the diameter bounds consumed from chapter 320.*
2. *T is the cofinal tail-window row, assigning each requested tolerance a finite interval index and the later windows through which the section may be inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing only finite stream windows before any regular rational readback is available.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reached from the stream windows and the nested interval endpoints.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic comparison budgets for endpoint width, tail inclusion, and section readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after I, T, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for nested intervals, tail windows, stream observations, regular readback, dyadic tolerance ledgers, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Real or Completion request through T, D, W, R, I and then into the displayed* $\mathsf{Real}^{\uparrow}$ *seal.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedCauchyRealSection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, W, R, D, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.NestedCauchyRealSectionUp.NestedCauchyRealSectionUp`
Lean4 ✓ `BEDC.Derived.NestedCauchyRealSectionUp.NestedCauchyRealSectionTasteGate single carrier alignment`

Theorem 3458.2 (Nested Cauchy real section NameCert obligations)**.** *For every accepted* $\mathsf{NestedCauchyRealSectio}$ *carrier $S = (I, T, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the nested interval row, cofinal tail windows, finite stream observations, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *tail-window exactness, requiring every Real or Completion tolerance request to expose T before reading W, R, D;*
3. *nested-interval compatibility, requiring the interval chain I and dyadic ledger D to control endpoint width, tail inclusion, and section readback;*
4. *Real-seal handoff, requiring the terminal* $\mathsf{Real}^{\uparrow}$ *row E to be read only after I, T, W, R, D have been displayed;*
5. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no quotient stream, selected limit, countable choice, host real equality, or ambient Real completeness.*

Proof. Carrier admission is projection from Definition 3458.1. The tuple displays the nested intervals, tail windows, stream windows, regular readback, dyadic ledger, Real seal, and structural rows.

Tail-window exactness follows from the placement of T. A tolerance request reaches finite observations and regular readback only through the displayed cofinal window, so no off-packet schedule can be used.

Nested-interval compatibility is the joint route through I and D. The interval chain supplies the nested endpoint data, while the dyadic ledger supplies the width and comparison budgets.

The Real seal is downstream of those rows. The replay row C reaches E only after I, T, W, R, D are visible, and H, P, N preserve and name that same route.

The excluded objects are absent from the carrier. Hence quotient streams, selected limits, countable choice, host real equality, and ambient Real completeness cannot appear as local NameCert obligations. □

Theorem 3458.3 (Nested Cauchy real section Real-completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *carrier factors through the* $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it exposes the local section certificate. The handoff exports finite interval, tolerance, observation, regular-readback, and Real-seal data, and exports no quotient stream, selected limit, countable choice, host real equality, or ambient Real completeness theorem.*

Proof. Start from an accepted packet $S = (I, T, W, R, D, E, H, C, P, N)$. Projection gives the nested interval row I, cofinal tail row T, finite stream observations W, regular rational readback R, dyadic tolerance ledger D, terminal Real seal E, and structural rows.

A completion-facing tolerance request is replayed through C. The replay fixes the tail window T, reads dyadic budgets from D, opens the stream window W, obtains regular rational readback from R, and checks the nested interval row I before exposing the Real seal E.

The structural rows add no external completion principle. They transport, replay, source, and name the same finite rows, so the read cannot export a quotient stream, a selected limit, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Theorem 3458.4 (Nested Cauchy real section choice-free Real seal)**.** *Every accepted* $\mathsf{NestedCauchyRealSection}^{\uparrow}$ *packet exposes its terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed nested interval, cofinal tail-window row, finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows. The exported Real-facing boundary contains no quotient stream, selected limit, countable-choice row, host real equality, or ambient Real-completeness coordinate.*

Proof. Project the carrier tuple $S = (I, T, W, R, D, E, H, C, P, N)$ of Definition 3458.1. The terminal Real seal is the row E, and the rows available before it are exactly the nested interval row I, cofinal tail-window row T, finite observation row W, regular rational readback row R, and dyadic tolerance ledger D.

Apply the Real-completion handoff theorem. It forces every completion-facing read to pass through I, T, W, R, D before exposing the local section certificate and the terminal Real seal.

The refused objects are absent from the carrier definition. Since no coordinate names a quotient stream, selected limit, countable-choice row, host real equality, or ambient Real-completeness theorem, the seal exported by E is choice-free in exactly the displayed packet sense. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedCauchyRealSection}^{\uparrow}$ packet is generated from nested interval rows, cofinal tail windows, finite $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3458.1, Definition 3458.2, and Definition 3458.3 seed the finite nested Cauchy section handoff surface for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedCauchyRealSection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exactness, nested-interval compatibility, dyadic tolerance stability, regular rational readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedCauchyRealSectionUp.

Chapter 3459

A Concrete Naming Certificate for $\mathsf{RealUniformCauchyName}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealUniformCauchyName}^{\uparrow}$ packet records a regular Cauchy real name together with the finite modulus route that makes its requested tolerances inspectable. It sits between the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RealCauchyModulus}^{\uparrow}$ schedule of chapter 1423, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives completion and rate-extraction consumers a named real-name surface without importing host limits, quotient streams, selected representatives, or ambient completeness.

Definition 3459.1 (RealUniformCauchyName carrier)**.** *A* $\mathsf{RealUniformCauchyName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, M, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family selected for the requested precision levels.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those windows, exposing the rational approximants and their regularity checks.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, recording the finite threshold chosen for each visible dyadic tolerance request.*
4. *Q is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *route that attaches the schedule M to the real-name readback rather than to a host modulus function.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, listing the inspected dyadic radii, comparison budgets, and refinement rows used by S, R, M, Q.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream windows, regular readbacks, modulus schedules, and dyadic ledger have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, modulus, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through $S, R, M, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealUniformCauchyName}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by componentwise displayed transport through these rows. It has no coordinate for quotient stream equality, host sequence limits, countable-choice schedules, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealUniformCauchyNameUp.RealUniformCauchyNameUp`
Lean4 ✓ `BEDC.Derived.RealUniformCauchyNameUp.RealUniformCauchyNameTasteGate single carrier alignment`

Theorem 3459.2 (RealUniformCauchyName NameCert obligations)**.** *For every accepted* RealUniformCauchyName *carrier* $U = (S, R, M, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window access through* S*, regular rational readback through* R*,* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule exposure through* M*,* $\mathsf{RealCauchyModulus}^{\uparrow}$ *schedule-to-real-name routing through* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through* D*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3459.1. The only rows available to the local naming surface are $S, R, M, Q, D, E, H, C, P, N$, so carrier admission supplies precisely the window, readback, modulus, tolerance, seal, transport, replay, provenance, and naming data.

The modulus route is split into M and Q. The row M records the finite threshold schedule for visible dyadic requests, while Q attaches that schedule to the real-name readback. Thus the route names a displayed Cauchy real name rather than a host modulus function.

Transport by H is componentwise, and replay by C follows the same displayed tolerance route. Since P, N source and name only this packet, no quotient stream, selected representative, countable-choice schedule, or ambient completeness datum is available to the NameCert surface. □

Theorem 3459.3 (RealUniformCauchyName seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{RealUniformCauchyName}^{\uparrow}$ *carrier factors through the ordered route*

$$S \longmapsto R \longmapsto M \longmapsto Q \longmapsto D \longmapsto E$$

together with H, C, P, N*. Consequently a completion or rate-extraction consumer may read the named real only through displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{RealCauchyModulus}^{\uparrow}$ *schedules, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the local structural rows.*

Proof. Begin with a requested terminal seal. By Definition 3459.1, the request first exposes the selected finite windows S and their regular rational readbacks R.

The tolerance request is then governed by the schedule row M and attached to the real-name route by Q. The dyadic ledger D records the inspected radii and comparison budgets before the terminal seal E is visible.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the same finite route, so the consumer receives no host limit, quotient stream equality, countable-choice selector, selected representative, trichotomy, independent real value, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Falsifiable prediction: Every $\mathsf{RealUniformCauchyName}^{\uparrow}$ seal read factors through StreamName finite windows, RegSeqRat readbacks, CauchyModulus and RealCauchyModulus schedules, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, realcauchymodulus, real: RealUniformCauchyNam is not bijective to any listed sibling because it jointly carries the finite window family, regular rational readback, explicit Cauchy schedule, Real-facing modulus route, dyadic tolerance ledger, terminal Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformCauchyName}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, CauchyModulus and RealCauchyModulus schedules, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3459.1, Definition 3459.2, and Definition 3459.3 bind the finite regular Cauchy real-name certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformCauchyName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host sequence limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-route exactness, tolerance-ledger accounting, componentwise classifier transport, and Real seal nonescape for BEDC.Derived.RealUniformCauchyNameUp.

Chapter 3460

A Concrete Naming Certificate for $\mathsf{ProofObligationGap}^{\uparrow}$

The $\mathsf{ProofObligationGap}^{\uparrow}$ packet realizes the MetaCIC subject-reduction obstruction of chapter 9386 as finite NameCert data. Its carrier is narrower than a confluence, normalization, or conversion endpoint: it packages named obligation rows, obstruction rows, refusal rows, and downstream handoff rows so that a subject-reduction consumer must see the gap surface before it reads any obstruction as usable evidence.

Falsifiable prediction: Every accepted $\mathsf{ProofObligationGap}^{\uparrow}$ route exposes the named obligation row before any downstream subject-reduction or consistency consumer can read the obstruction.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, normalformconsistencyseal, bounded-normal-conversion-bridge: the gap packet is not bijective to these siblings because it jointly carries named obligations, obstruction/refusal rows, and downstream handoff routes rather than a positive confluence, normalization, or conversion endpoint.

The $\mathsf{ProofObligationGap}^{\uparrow}$ carrier is read after the MetaCIC critical-path packet of chapter 765 and before the bounded conversion boundary of chapter 3439 can consume an obstruction row.

Definition 3460.1 (Proof-obligation gap carrier)**.** *A* $\mathsf{ProofObligationGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (O, B, R, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the named obligation row, carrying one of the four MetaCIC subject-reduction obligations isolated in section 9386.2.*
2. *B is the obstruction row, recording that the named obligation cannot be replaced by a blanket closed subject-reduction theorem at this boundary.*
3. *R is the refusal row, rejecting any route that tries to read the obstruction before the obligation name has been exposed.*
4. *M is the MetaCIC handoff row, routing accepted packets to downstream subject-reduction, consistency, critical-path, or audit consumers only after O, B, R have been displayed together.*
5. *H records componentwise* hsame *transport for obligation, obstruction, refusal, handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a downstream MetaCIC request through O, B, R, M.*

7. *P is the* Pkg *provenance over* $\mathsf{ProofObligationGap}^{\uparrow}$*, the MetaCIC open-problem frontier,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

8. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$ *row.*

The classifier accepts exactly packets where a named obligation row is paired with its obstruction and refusal rows before any MetaCIC handoff is available. It has no coordinate for positive confluence, normalization, bounded conversion, closed subject reduction, or a consistency seal.

Theorem 3460.2 (Proof-obligation gap NameCert obligations)**.** *For every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *carrier $G = (O, B, R, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the named obligation, obstruction, refusal, MetaCIC handoff, transport, replay, provenance, and local naming rows;*

2. *obligation-first exposure, requiring O to be visible before any downstream consumer reads B;*

3. *refusal exactness, requiring R to reject obstruction-only reads and three-row subject-reduction shortcuts;*

4. *handoff confinement, requiring M to route only the displayed O, B, R surface to MetaCIC consumers;*

5. *structural non-escape, requiring H, C, P, N to preserve and name the same packet without exporting confluence, normalization, conversion, closed subject reduction, or consistency-seal evidence.*

Lean `TasteGate.lean` *will include explicit* `Nontrivial ProofObligationGapUp` *and* `FieldFaithful ProofObligationGapUp` *instances so the accepted and refused surfaces remain distinct and the row order O, B, R, M, H, C, P, N is preserved by faithful reads.*

Lean4 ✓ `BEDC.Derived.ProofObligationGapUp.ProofObligationGapTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3460.1. The accepted packet displays one named obligation row, one obstruction row, one refusal row, one handoff row, and the structural rows.

Obligation-first exposure follows from the classifier. The obstruction row B is accepted only together with the named obligation row O, so a downstream subject-reduction or consistency consumer cannot read B as a standalone discharge.

Refusal exactness is the role of R. It rejects routes that omit O, routes that treat the obstruction as a positive theorem, and routes that offer fewer than the named obligation surface required by section 9386.2.

The handoff row M then confines downstream reads to the displayed surface O, B, R. The structural rows H, C, P, N transport, replay, source, and name that packet, and they add no coordinate for confluence, normalization, conversion, closed subject reduction, or a consistency seal. □

Theorem 3460.3 (Proof-obligation gap MetaCIC handoff)**.** *Every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *handoff to a MetaCIC subject-reduction, consistency, critical-path, or audit consumer first exposes the named obligation row O, then the obstruction row B, then the refusal row R, and only then the handoff row M. Consequently the packet records the obstruction surface as a finite named gap and exports no positive confluence theorem, normalization endpoint, bounded conversion bridge, closed subject-reduction theorem, or consistency seal.*

Proof. Project an accepted carrier $G = (O, B, R, M, H, C, P, N)$. The only consumer-facing handoff coordinate is M, and it is displayed after the named obligation, obstruction, and refusal rows.

Replay through C preserves that order. A downstream MetaCIC request must pass through O, B, R before it can read M, so the obstruction is always seen together with the obligation name and the refusal boundary.

The packet has no endpoint row for confluence, normalization, bounded conversion, closed subject reduction, or consistency sealing. Provenance and local naming therefore authenticate only the finite gap surface, not any positive theorem at those endpoints. □

Theorem 3460.4 (Proof-obligation gap MetaCIC sibling dependency)**.** *Every accepted* $\mathsf{ProofObligationGap}^{\uparrow}$ *packet consumed by a MetaCIC critical-path or bounded-normal-conversion consumer must cite the named obligation row before it can hand off to chapter 765 or chapter 3439. This sibling dependency is substantive: those consumers can read the finite gap only after obligation-first exposure and refusal exactness, not as positive confluence evidence, normalization evidence, bounded-conversion evidence, or a closed subject-reduction endpoint.*

Proof. Project an accepted carrier $G = (O, B, R, M, H, C, P, N)$. The named obligation row O is displayed before the obstruction row B, the refusal row R, and the MetaCIC handoff row M.

A critical-path consumer may therefore cite chapter 765 only after reading O together with the obstruction and refusal boundary. A bounded-normal-conversion consumer may likewise cite chapter 3439 only after the same obligation-first exposure.

The dependency is not a positive endpoint. The rows B and R record obstruction and refusal exactness, while M, H, C, P, N preserve and name that finite gap surface. No row supplies confluence, normalization, bounded conversion, or closed subject reduction. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofObligationGap}^{\uparrow}$ packet is generated from named MetaCIC obligation rows, obstruction rows, refusal rows, downstream MetaCIC handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3460.1, Definition 3460.2, Definition 3460.3, and Definition 3460.4 seed the finite proof-obligation gap surface for MetaCIC subject-reduction obstruction consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProofObligationGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close positive confluence, normalization, bounded conversion, closed subject reduction, consistency sealing, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, obligation-first exposure, refusal exactness, MetaCIC handoff confinement, transport, replay, provenance, local naming, Nontrivial ProofObligationGapUp, and Field-Faithful ProofObligationGapUp.

Chapter 3461

A Concrete Naming Certificate for $\mathsf{MeasurableFunction}^{\uparrow}$

Following Caratheodory and Bourbaki measurable maps, the $\mathsf{MeasurableFunction}^{\uparrow}$ packet records the finite preimage surface by which a source measurable-event family is routed into a target measurable-event family. It sits after the event rows of chapter 92, the generated-set boundary of chapter 3384, the $\mathsf{SetLike}^{\uparrow}$ set-membership carrier discipline, the terminal real-name seal of chapter 38, and the finite observation windows of chapter 126. The packet names measurable-map behavior as finite preimage ledgers, not as a host function space, quotient equality of functions, selected representatives, or ambient completed measure theory.

Definition 3461.1 (MeasurableFunction carrier)**.** *A* $\mathsf{MeasurableFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, A, B, G, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source measurable carrier row, read through* $\mathsf{Measure}^{\uparrow}$ *or through a* $\mathsf{BorelSet}^{\uparrow}$ *event source when the map is real-facing.*
2. *T is the target measurable carrier row, again read through* $\mathsf{Measure}^{\uparrow}$ *or the* $\mathsf{BorelSet}^{\uparrow}$ *generated-set boundary.*
3. *A is the source event-family row, with membership and subset readback routed through* $\mathsf{SetLike}^{\uparrow}$*.*
4. *B is the target event-family row, with target membership readback routed through* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and finite* $\mathsf{StreamName}^{\uparrow}$ *observations when the target is a real or Borel source.*
5. *G is the graph-read row naming the finite source-to-target observation pairs exposed to the packet.*
6. *R is the preimage ledger assigning each displayed target event row in B to a displayed source event row in A and recording the finite membership readback for the graph row G.*
7. *H stores componentwise* hsame *transport for the source carrier, target carrier, event-family rows, graph row, preimage ledger, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay by which a consumer reads target events, preimage events, graph observations, and source memberships.*
9. *P is the* Pkg *provenance over* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, T, A, B, G, R, H, C, P, N$. It has no coordinate for a host function type, quotient equality of functions, almost-everywhere equality, completion of a measure, or a selected representative.

Lean4 (definition) `BEDC.Derived.MeasurableFunctionUp.MeasurableFunctionUp`

Theorem 3461.2 (MeasurableFunction NameCert obligations)**.** *For the carrier of Definition 3461.1, the local* $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, source-event admission, target-event admission, graph-read admission, preimage-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted measurable-map consumer must read a target event through B, its displayed preimage through R, and the source-event result through A before any* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *sibling row is exposed.*

Proof. Project the carrier $F = (S, T, A, B, G, R, H, C, P, N)$. The source and target carrier rows S, T supply the measurable contexts, while A and B supply the only event-family rows available to the packet.

Read a target event request through B. The preimage ledger R is the unique row that assigns that target event to a source event in A, and the graph row G supplies the finite membership readback tying source observations to target observations.

The structural rows add no host function object. The componentwise transport H, replay row C, provenance row P, and local naming row N preserve the same displayed route through the sibling certificates, so the listed rows exhaust the NameCert obligation surface. □

Theorem 3461.3 (MeasurableFunction simple preimage ledger)**.** *Every simple target-event read of an accepted* $\mathsf{MeasurableFunction}^{\uparrow}$ *packet factors through the finite ledger route*

$$B \longmapsto R \longmapsto A \longmapsto G \longmapsto H, C, P, N.$$

For real-facing Borel targets, the target membership observation reaches $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *only through the displayed* $\mathsf{BorelSet}^{\uparrow}$ *row in B and the graph-read row G. Thus preimage formation exports a named source event row, not a host inverse-image function, almost-everywhere quotient, completed measure, selected representative, or ambient function equality.*

Proof. Start with an accepted target-event request. The request is not a free subset of a host codomain; it is a displayed target event row in B, transported through the target carrier row T.

The preimage ledger R is then inspected. It names the corresponding source event row in A and records the graph-read membership comparison supplied by G. For real-facing Borel targets, this graph read is the only place where $\mathsf{Real}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ observations enter the route.

Finally replay the structural rows. The rows H, C, P, N transport, replay, source, and name the same finite route over $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, and $\mathsf{SetLike}^{\uparrow}$. Since no further coordinate is present, the excluded host inverse-image, quotient, completion, representative-selection, and ambient equality objects cannot be exported. □

Falsifiable prediction: A $\mathsf{MeasurableFunction}^{\uparrow}$ consumer that obtains a host function equality, an almost-everywhere quotient class, a completed measure, or a selected representative without first reading the displayed preimage ledger R refutes the carrier surface.

Independence witness: measure, borelset, setlike, real, streamname: no listed sibling supplies the whole ten-row packet $F = (S, T, A, B, G, R, H, C, P, N)$, because each sibling contributes only

event-family, generated-set, membership, real-seal, or finite-window data while this chapter binds the measurable-map preimage ledger with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3461.1, Definition 3461.2, and Definition 3461.3 seed the finite measurable-map surface over source events, target events, graph reads, preimage ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeasurableFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close random variables, distributions, integrals, almost-everywhere equality, completed measures, quotient function spaces, selected representatives, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, preimage-ledger exactness, graph-read preservation, componentwise transport, and ledger-refusal proofs for BEDC.Derived.MeasurableFunctionUp.

Constructive story: A $\mathsf{MeasurableFunction}^{\uparrow}$ packet is generated from source and target measurable carrier rows, source and target event-family rows, a graph-read row, a preimage ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3462

A Concrete Naming Certificate for $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ packet records the finite comparison surface between directed Cauchy nets and Cauchy filter bases before any completion-facing consumer reads a Real seal. It sits between the Cauchy-net carrier of chapter 589, the Cauchy-filter carrier of chapter 333, the Moore–Smith completion packet of chapter 2190, the uniform completion route of chapter 1473, and the Real readback boundary of chapter 38. The packet is finite BEDC data: directed index rows, tail rows, filter-base rows, cofinal comparison rows, completion handoff rows, readback rows, transport, replay, provenance, and a local NameCert. It does not import quotient completions, selected limits, countable choice, or an ambient completed space.

Definition 3462.1 (Moore–Smith Cauchy-net/filter bridge carrier)**.** *A* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, T, F, K, U, M, S, R, Y, E, H, C, P, N)$$

with the following displayed rows:

1. D *is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row naming the finite directed comparisons used by the net side;*
2. T *is the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row, recording eventual containment tests over the directed index surface* D*;*
3. F *is the* $\mathsf{CauchyFilter}^{\uparrow}$ *base row, recording finite base refinements and Cauchy diameter tests;*
4. K *is the cofinal net-filter comparison row, displaying the finite route from directed tails to filter bases and back to cofinal tail data;*
5. U *is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row, read only after* D, T, F, K *have been displayed;*
6. M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row for metric-facing completion consumers;*
7. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by completion-facing readback;*
8. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same window and comparison data;*
9. Y *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic comparison ledger used to bound the displayed tails, bases, and readbacks;*

10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the net, filter, cofinal, completion, and readback rows have been displayed;*
11. *H records componentwise* hsame *transport for all displayed rows;*
12. *C records displayed* Cont *replay from directed tails through filter bases, completion handoff, readback windows, and the Real seal;*
13. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$ *row.*

The public carrier exports the finite packet B and no quotient completion, selected subnet, chosen filter limit, countable-choice schedule, or ambient completeness datum.

Lean4 ✓ `BEDC.Derived.MooreSmithCauchyNetFilterBridgeUp.MooreSmithCauchyNetFilterBridgeTasteGate`
`single carrier alignment`

Theorem 3462.2 (Moore–Smith Cauchy-net/filter bridge NameCert obligations)**.** *For the carrier of Definition 3462.1, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for $B = (D, T, F, K, U, M, S, R, Y, E, H, C, P, N)$; directed-index admission through D; Cauchy-net tail admission through T; Cauchy-filter base admission through F; cofinal comparison through K; uniform and metric completion handoff through U and M; StreamName, RegSeqRat, DyadicRatCore, and Real readback through S, R, Y, E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The rows D, T, F, K expose the directed index surface, the Cauchy-net tails, the Cauchy-filter bases, and the cofinal comparison between them as visible coordinates of one finite packet.

The completion and readback rows are downstream. The uniform and metric completion handoffs U, M are read only after the net/filter comparison row is displayed, and the finite readback rows S, R, Y, E attach the StreamName, RegSeqRat, dyadic, and Real-facing reads to the same displayed route.

The structural rows exhaust the NameCert surface. Componentwise transport, displayed replay, provenance, and local naming are exactly H, C, P, N, so a consumer cannot obtain a quotient completion, selected subnet, chosen filter limit, countable-choice schedule, or ambient completion principle from the packet. □

Theorem 3462.3 (Moore–Smith Cauchy-net/filter bridge handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ *carrier factors through the displayed route*

$$D, T, F, K, U, M, S, R, Y, E, H, C, P, N$$

before reading any $\mathsf{Real}^{\uparrow}$ *seal. In particular, a consumer moving between chapter 589, chapter 333, chapter 2190, and chapter 1473 reads the cofinal net/filter rows before it can use the Real-facing readback of chapter 38.*

Proof. Begin with a completion-facing request. Carrier admission forces the directed index row D, Cauchy-net tail row T, Cauchy-filter base row F, and cofinal comparison row K to be visible before any completion handoff row appears.

The handoff rows U and M then carry only the displayed comparison route. They do not identify completion classes, choose a subnet, or select a filter limit; they pass the finite net/filter comparison to uniform and metric completion consumers.

Finally the readback rows S, R, Y, E and the structural rows H, C, P, N keep the Real seal downstream of the same displayed route. Since the packet has no coordinate for quotient completion, countable choice, selected limits, or ambient completeness, every completion-facing read factors through the named rows before it reaches $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}$ packet is generated from directed index data, Cauchy-net tail rows, Cauchy-filter base rows, a cofinal net-filter comparison, uniform and metric completion handoff rows, StreamName windows, RegSeqRat and DyadicRatCore readbacks, a RealUp seal, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3462.1, Definition 3462.2, and Definition 3462.3 bind the finite net/filter/cofinal route before uniform, metric, and Real readback consumers inspect the packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchyNetFilterBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completions, selected limits, countable choice, or ambient completed-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal comparison, completion handoff, transport stability, replay, provenance, and local naming for BEDC.Derived.MooreSmithCauchyNetFilterBridgeUp.

Chapter 3463

A Concrete Naming Certificate for $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$

The $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ packet realizes the punctured WNA cube, CUN square, bridge-edge, and articulation-gate support graph of Definition 9462.1293 as finite NameCert data. Its carrier records the cube vertex rows, the deleted ACA row, the CUN square, the UAG and AGG bridge rows, adjacency ledgers, biconnected-component rows, transport, replay, provenance, and local naming. It sits beside the codon source-window carrier of chapter 2234, the DNA window boundary tile of chapter 3393, and the codon-window spectral inverse of chapter 2161. The packet is a finite combinatorial support certificate; it does not assert a biochemical mechanism, a selected evolutionary path, or a host-level graph quotient.

Definition 3463.1 (Punctured cube skeleton carrier)**.** *A* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (W, A, C, B, G, E, H, T, Q, R, N)$$

with the following displayed rows:

1. *W is the WNA cube row listing the eight codon vertices with first coordinate in* $\{\mathrm{U}, \mathrm{A}\}$*, middle coordinate in* $\{\mathrm{U}, \mathrm{C}, \mathrm{A}, \mathrm{G}\}$*, and terminal coordinate* A*;*
2. *A is the puncture row deleting* ACA *and retaining the seven displayed WNA vertices;*
3. *C is the CUN square row over* $\{\mathrm{CUU}, \mathrm{CUC}, \mathrm{CUA}, \mathrm{CUG}\}$*;*
4. *B is the bridge-edge row carrying* $\{\mathrm{CUA}, \mathrm{UUA}\}$*,* $\{\mathrm{UAA}, \mathrm{UAG}\}$*, and* $\{\mathrm{AGA}, \mathrm{AGG}\}$*;*
5. *G is the articulation-gate row identifying the finite support graph induced by A, C, B;*
6. *E is the edge-decomposition ledger comparing the cube face, CUN square, bridge edges, and biconnected components;*
7. *H records componentwise* hsame *transport for the vertex, puncture, square, bridge, gate, and edge-decomposition rows;*
8. *T records displayed* Cont *replay from the support formula to the induced graph and component rows;*
9. *Q is the* Pkg *provenance over* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*
10. *R is the TasteGate requirement row for later* Nontrivial $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *and* FieldFaithful PuncturedCubeS *instances;*

11. N is the local $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$ *row.*

Theorem 3463.2 (Punctured cube skeleton NameCert obligations)**.** *For the carrier of Definition 3463.1, the local* $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$ *surface consists exactly of WNA cube admission, puncture admission, CUN square admission, bridge-edge admission, articulation-gate admission, edge-decomposition admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, TasteGate nontriviality and field-faithfulness requirements, and local naming. The future Lean-facing TasteGate surface must distinguish the puncture row A from the unpunctured cube row W and must preserve every field $W, A, C, B, G, E, H, T, Q, R, N$.*

Lean4 ✓ `BEDC.Derived.PuncturedCubeSkeletonUp.PuncturedCubeSkeletonTasteGate single carrier alignment`

Proof. Project the carrier. The rows W, A, C, B, G, E display the finite support formula, the puncture, the CUN square, the bridge edges, the articulation gate, and the induced edge-decomposition ledger.

Transport and replay are structural. The rows H, T, Q, N preserve and name the same finite graph support data without adding a host quotient or a biological mechanism.

The TasteGate requirement row R records the exact Lean obligations: nontriviality must separate the punctured packet from a collapsed graph, and field faithfulness must preserve every displayed vertex, puncture, bridge, gate, and decomposition row. These obligations are part of the carrier surface and do not assert that the Lean instances already exist. □

Theorem 3463.3 (Punctured cube skeleton boundary nonescape)**.** *Every consumer of an accepted* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier factors through*

$$W, A, C, B, G, E, H, T, Q, R, N$$

before reading the punctured cube skeleton as finite NameCert data. Consequently the consumer cannot erase the ACA *puncture, collapse the CUN square into the WNA cube, drop the bridge-edge rows, replace the articulation gate by a host graph quotient, or bypass the later* Nontrivial *and* FieldFaithful *TasteGate requirements.*

Proof. Start from the support rows. The WNA cube, puncture, CUN square, bridge edges, articulation gate, and edge-decomposition ledger are separate coordinates of the carrier.

Now use componentwise transport and replay. The rows H and T preserve those coordinates as finite BHist data, while Q and N record provenance and local naming for the same route.

Finally read the TasteGate row. Because R is displayed before a downstream carrier can be admitted, any later Lean realization must keep nontriviality and field faithfulness visible. Thus no consumer can erase the puncture, collapse the support graph, or bypass the stated gate requirements. □

Theorem 3463.4 (Punctured cube skeleton philosophy route)**.** *Every accepted* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *carrier is the finite support-graph consumer of Definition 9462.1293: the rows $W, A, C, B, G, E, H, T, Q, R, N$ record the displayed punctured WNA cube, CUN square, bridge edges, articulation gate, decomposition ledger, transport, replay, provenance, TasteGate requirements, and local name. Its concrete sibling route is through the codon source-window packet of chapter 2234, the DNA window boundary tile of chapter 3393, and the codon-window spectral inverse packet of chapter 2161; those chapters supply finite window, boundary-tile, and spectral-inverse rows, not biochemical mechanism claims.*

Proof. The philosophy theorem supplies the displayed support formula. The carrier rows W, A, C, B, G, E are exactly the finite graph-support data used to read that formula as a punctured cube skeleton.

The sibling chapters named in the statement provide only finite BEDC routes: codon windows, DNA boundary tiles, and codon-window spectral inverse rows. Reading them alongside H, T, Q, R, N records a dependency route for the support graph, not an empirical biochemical assertion.

Thus the packet leaves kernel-only isolation by naming its concrete finite consumers while retaining the same boundary as Definition 3463.3. □

Falsifiable prediction: A packet that omits the ACA puncture row, the CUN square row, or one of the bridge-edge rows cannot be accepted as a $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ carrier.

Independence witness: codon-window-six-bridge, dna-window-six-dynamics-bridge, codon-window-spectral-inverse: none is bijective to $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ because this carrier binds the punctured WNA cube, CUN square, bridge edges, articulation gate, edge decomposition, TasteGate requirement row, transport, replay, provenance, and local naming as one finite support graph packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}$.

Constructive story: A $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ packet is generated from WNA cube vertices, the deleted ACA puncture row, CUN square rows, bridge-edge rows, an articulation-gate support graph, an edge-decomposition ledger, hsame transports, Cont replay, Pkg provenance, a TasteGate requirement row, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3463.1, Definition 3463.2, and Definition 3463.3 bind the finite punctured cube skeleton support graph and its TasteGate requirements.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PuncturedCubeSkeleton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biochemical mechanism, evolutionary pathway selection, host graph quotients, or completed TasteGate instances.

Upgrade path: Obligation closure requires checked carrier admission, boundary nonescape, transport stability, replay, provenance, local naming, and explicit Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.PuncturedCubeSkeletonUp.

Chapter 3464

A Constructive Closed Metric Ball Certificate for $\mathsf{ClosedMetricBall}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ClosedMetricBall}^{\uparrow}$ packet records the center-radius containment object of a constructive metric space as finite BEDC NameCert data. It sits after the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the separated comparison boundary of chapter 1411, the complete-metric limit surface of chapter 128, and the regular-Cauchy equality route of chapter 1412. The packet names non-strict metric containment for one displayed center and radius; it does not import host subsets, open-cover compactness, quotient point equality, or ambient completeness.

Definition 3464.1 (Closed metric ball carrier)**.** *A* $\mathsf{ClosedMetricBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, X, c, r, \rho, D, R, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the distance-read interface.*
2. *X is the inspected point row.*
3. *c is the displayed center row.*
4. *r is the* $\mathsf{Real}^{\uparrow}$ *or rational radius row.*
5. *ρ is the nonnegative-radius ledger, read before any containment classifier is consumed.*
6. *D is the metric-distance witness row for the finite comparison $d(c, X)$.*
7. *R is the ordered-real comparison row recording $d(c, X) \leq r$ through the* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *compatibility row, used only to transport zero-distance agreement through the same metric source.*
9. *Q is the optional* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *handoff row for consumers that arrive from completed or regular-Cauchy names.*
10. *H stores componentwise* hsame *transport for $M, X, c, r, \rho, D, R, S, Q$ and the structural rows.*
11. *C stores displayed* Cont *replay from the metric source and radius ledger to the distance witness, ordered comparison, separated route, and completion-facing handoff.*

12. *P is the* Pkg *provenance over* $\mathsf{ClosedMetricBall}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$ *row.*

The carrier compares accepted containment packets only through $M, X, c, r, \rho, D, R, S, Q, H, C, P, N$. It has no coordinate for arbitrary subsets, topology closure, selected limits, open-cover compactness, quotient point equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedMetricBallUp.ClosedMetricBallTasteGate single carrier alignment`

Theorem 3464.2 (Closed metric ball NameCert obligations)**.** *For an accepted* $\mathsf{ClosedMetricBall}^{\uparrow}$ *carrier $B = (M, X, c, r, \rho, D, R, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, center and point exposure, radius nonnegativity, distance-witness visibility, ordered-real containment, separated-metric compatibility, completion-facing handoff when Q is present, componentwise transport, continuation replay, provenance, and local naming. Every containment read therefore follows the displayed route*

$$M, X, c, r, \rho \longmapsto D \longmapsto R \longmapsto S, Q \longmapsto H, C, P, N$$

and no host subset or compactness principle supplies an extra membership coordinate.

Proof. Project the carrier rows. The metric source, inspected point, center, radius, nonnegative-radius ledger, distance witness, ordered-real comparison, separated route, completion-facing row, transport, replay, provenance, and local name are the only public coordinates.

The containment classifier reads the distance witness D only after M, X, c, r, ρ have been displayed. The row R then records the non-strict comparison through the $\mathsf{Real}^{\uparrow}$ seal rather than through an ambient subset predicate.

Separated and completed consumers are downstream. The row S can transport only zero-distance agreement already visible in the same metric source, and the row Q can hand a completed or regular-Cauchy read to the same comparison route only after the closed-ball rows are present.

Finally H, C, P, N preserve, replay, source, and name the route. Since the carrier contains no host subset, topological closure, open-cover, selected-limit, quotient, choice, trichotomy, or ambient-completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3464.3 (Closed metric ball containment handoff)**.** *Every separated, bounded, complete, or compact metric consumer of an accepted* $\mathsf{ClosedMetricBall}^{\uparrow}$ *packet must read the finite containment route $M, X, c, r, \rho, D, R, S, Q, H, C, P, N$ before it may use the point as lying in the closed ball. The exported handoff is only the center-radius comparison $d(c, X) \leq r$ with its displayed transport, replay, provenance, and local name; it does not provide a compactness theorem, a host-level closed subset, quotient equality of points, a selected completion limit, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from the containment request. The rows M, X, c, r, ρ fix the metric source, point, center, radius, and nonnegative-radius ledger before a distance comparison can be consumed.

The distance and ordered-real rows are next. The row D supplies the displayed metric-distance read, and R supplies the non-strict $\mathsf{Real}^{\uparrow}$ comparison. This is the only route by which the packet exposes membership.

Downstream metric consumers then read S or Q only as compatibility rows. They preserve the same center-radius comparison for separated or completed contexts, but they cannot replace it by topology closure, host set equality, compactness, or selected limits.

The structural rows H, C, P, N carry the route forward without adding mathematical content. Therefore every exported handoff remains the finite closed-ball containment certificate described above. □

Falsifiable prediction: A downstream metric consumer that omits the metric source row, the center-radius ledger, or the ordered-real containment row cannot read a $\mathsf{ClosedMetricBall}^{\uparrow}$ packet as a closed ball.

Independence witness: metric, real, separatedmetric, completemetric, cauchysequenceequivalence, metricclosedball: $\mathsf{ClosedMetricBall}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, inspected point, center, radius, nonnegative-radius ledger, distance witness, ordered-real containment, separated compatibility, completion-facing handoff, transport, replay, provenance, and local naming as one center-radius containment packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedMetricBall}^{\uparrow}$ packet is generated from a MetricUp source row, an inspected point row, a center row, a RealUp or rational radius row, a nonnegative-radius ledger, a metric-distance witness, an ordered-real containment comparison, separated-metric compatibility, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3464.1, Definition 3464.2, and Definition 3464.3 seed the finite center-radius containment packet for constructive metric spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedMetricBall}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, topology closure, open-cover compactness, quotient point equality, selected completion limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, radius nonnegativity, distance-witness visibility, ordered-real containment, separated compatibility, completion-facing handoff, transport and replay stability, provenance non-escape, and local naming.

Chapter 3465

A Concrete Naming Certificate for $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen, the $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ packet records finite-prefix dependence of Baire and choice-sequence functionals as NameCert data. It sits beside the Baire-space surface of chapter 334, the choice-sequence rows of chapter 1482, the spread-law route of chapter 1532, and the finite stream-window surface of chapter 126. The packet also routes rational and real readbacks through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, keeping the continuity claim as a finite-prefix certificate rather than as a host functional, countable-choice selector, quotient stream equality, or ambient real-completeness theorem.

Definition 3465.1 (Brouwer continuity principle carrier)**.** *A* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, Q, S, W, R, E, M, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *source row, exposing only finite inspected prefixes of the input sequence;*
2. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row, recording the displayed choice-sequence presentation used by the functional read;*
3. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row, naming the spread law that admits the finite prefixes consumed by the packet;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, storing the requested prefix length, inspected stream cells, and pullback route to the source rows;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations made from the accepted finite prefix;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same finite-window and rational-readback data;*
7. *M is the finite modulus row, stating which prefix length is sufficient for the displayed output read;*
8. *H records componentwise* hsame *transports for B, Q, S, W, R, E, M and the local naming row;*
9. *C records the displayed* Cont *replay by which consumers read $B, Q, S, W, R, E, M, H, P, N$ in that order;*
10. *P is the* Pkg *provenance over* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through the displayed finite prefix, modulus, readback, seal, transport, continuation, package, and local NameCert rows.

Theorem 3465.2 (Brouwer continuity principle NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$ *surface for Definition 3465.1 consists exactly of carrier admission for $K = (B, Q, S, W, R, E, M, H, C, P, N)$, Baire-source exposure through B, choice-sequence exposure through Q, spread-law admission through S, finite stream-window inspection through W, rational readback through R, real-seal routing through E, finite-modulus display through M, componentwise transport through H, continuation replay through C, package provenance through P, and local naming through N. Every accepted continuity read is classified only by these displayed rows.*

Lean4 ✓ `BEDC.Derived.BrouwerContinuityPrincipleUp.BrouwerContinuityPrincipleTasteGate single carrier alignment`

Proof. Project the carrier tuple K. The Baire source, choice-sequence presentation, spread law, finite stream window, rational readback, real seal, finite modulus, transports, continuations, package row, and local name are all stored coordinates.

The source and admissibility obligations project B, Q, S, W. The output-facing obligations project R, E, M, so the real read is reached only after the finite prefix and rational readback have been displayed.

The structural obligations project H, C, P, N. Since no coordinate names a host functional, global continuity theorem, countable-choice selector, quotient stream equality, selected real representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the local certificate exports only the finite-prefix route. □

Theorem 3465.3 (Brouwer continuity principle finite-prefix modulus)**.** *For every accepted* BrouwerContinuityPrinci *carrier, each output read from a Baire or choice-sequence input factors through the finite route*

$$B, Q, S \longmapsto W \longmapsto M \longmapsto R \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Thus the consumer receives a displayed prefix length and the corresponding inspected $\mathsf{StreamName}^{\uparrow}$ *window before it reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The route exports no host functional continuity theorem, no choice-selected modulus, no quotient stream equality, no arbitrary bar induction principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Read B, Q, S first. These rows display the Baire-space source, the choice-sequence presentation, and the spread-law admission surface on which the functional read is allowed to depend.

Next read the finite-window and modulus rows W, M. They name the inspected prefix and the prefix length sufficient for the displayed output read, so the rational and real observations cannot be consumed before the finite prefix is exposed.

Finally route the output through R, E, H, C, P, N. Componentwise transport and continuation replay preserve the same finite-prefix packet for downstream consumers, while provenance records the sibling Baire-space, choice-sequence, spread-law, stream-name, regular-rational, and real rows. Since no refused object is a carrier coordinate, the output read remains a finite-prefix NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerContinuityPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ packet is generated from $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite-modulus, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{BrouwerContinuityPrinciple}^{\uparrow}}$, seed)

Scope closed: Definition 3465.1, Definition 3465.2, and Definition 3465.3 seed the finite-prefix continuity route over the displayed Baire-space, choice-sequence, spread-law, stream-name, regular-rational, real-seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{BrouwerContinuityPrinciple}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, host functional continuity theorem, choice-selected modulus, quotient stream equality, arbitrary bar induction principle, selected real representative, ambient Real$^{\uparrow}$ completeness theorem, or bridge-checked standard model.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same BaireSpace$^{\uparrow}$, ChoiceSequence$^{\uparrow}$, SpreadLaw$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 3466

A Concrete Naming Certificate for $\mathsf{CauchyHadamard}^{\uparrow}$

Following Cauchy and Hadamard's power-series radius formula, the $\mathsf{CauchyHadamard}^{\uparrow}$ packet records the constructive radius route for a real power series as finite BEDC certificate data. It sits over the coefficient windows of chapter 1583, the radius seal of chapter 66, the regular root-bound route of chapter 1481, and the sibling ratio-test comparison route of chapter 1461. The chapter packages the traditional formula as a NameCert route through displayed coefficient windows, dyadic tolerance rows, and regular-Cauchy readbacks rather than through a host limsup, quotient stream, choice-selected radius, or ambient completeness theorem.

Definition 3466.1 (Cauchy-Hadamard carrier)**.** *A* $\mathsf{CauchyHadamard}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, R, G, D, H, C, P, N)$$

with the following visible rows:

1. *A is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *source row, exposing only the displayed coefficient family and its finite inspection windows;*
2. *W is the coefficient-window ledger, recording the inspected unary indices, absolute-coefficient reads, and root-bound requests used by the packet;*
3. *R is the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound route, read from the same coefficient windows and tolerance requests;*
4. *G is the* $\mathsf{ConvRad}^{\uparrow}$ *seal row reached from the root-bound route and the power-series source;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the dyadic slack and comparison budgets used by the root-bound and radius rows;*
6. *H records componentwise* hsame *transports for the source row, coefficient windows, root-bound route, radius seal, tolerance ledger, and local naming row;*
7. *C records the displayed* Cont *replay by which consumers read A, W, R, G, D, H, P, N in that order;*
8. *P is the* Pkg *provenance over* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and their recorded transport, continuation, package, and local NameCert provenance.

Lean4 ✓ `BEDC.Derived.CauchyHadamardUp.CauchyHadamardTasteGate single carrier alignment`

Theorem 3466.2 (Cauchy-Hadamard NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$ *surface for Definition 3466.1 consists exactly of carrier admission for* $K = (A, W, R, G, D, H, C, P, N)$*, coefficient-window exposure through* A, W*, root-bound exposure through* R*, radius-seal exposure through* G*, dyadic tolerance exactness through* D*, componentwise transport through* H*, displayed continuation replay through* C*, package provenance through* P*, and local naming through* N*. Every accepted radius read is classified only by the displayed power-series row, finite coefficient windows, root-bound route, tolerance ledger, radius seal, and structural rows.*

Proof. Project the carrier tuple K. The source row, coefficient-window ledger, root-bound route, radius seal, dyadic tolerance ledger, transport rows, continuation routes, package row, and local name are all stored coordinates.

Carrier admission and coefficient exposure are therefore finite projections from A and W. The root-bound and radius obligations project R and G, with D recording the tolerance comparisons used by both rows.

The structural obligations project H, C, P, N. Since no coordinate names a host limsup, quotient stream, choice-selected radius, or ambient completeness theorem, the local certificate exports only the displayed Cauchy-Hadamard route. □

Theorem 3466.3 (Cauchy-Hadamard root-bound handoff)**.** *For every accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *carrier, each root-bound request consumed by the radius row factors through the finite route*

$$A \longmapsto W \longmapsto D \longmapsto R \longmapsto G \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The handoff uses the $\mathsf{RealPowerSeries}^{\uparrow}$ *coefficient source, the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{ConvRad}^{\uparrow}$ *seal. The* $\mathsf{RegularCauchyRatioTest}^{\uparrow}$ *sibling is available only as a comparison provenance row in* P *and does not replace the displayed root-bound route.*

Proof. Read A, W, D, R, G from the carrier. A radius request first opens the power-series coefficient source and the finite coefficient-window ledger, so the root-bound row sees only displayed coefficient data.

Use the dyadic tolerance ledger D to align the slack and comparison budget consumed by R. The radius seal G is then read after the root-bound route, not from a host supremum or selected limsup witness.

The continuation and provenance rows preserve the same order for downstream consumers. The ratio-test sibling remains a recorded dependency anchor in P, while the accepted handoff itself is the root-bound route through $A, W, D, R, G, H, C, P, N$. □

Theorem 3466.4 (Cauchy-Hadamard root-bound NameCert route)**.** *Any accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *radius read whose* $\mathsf{ConvRad}^{\uparrow}$ *row consumes a Cauchy-Hadamard exactness witness factors through the displayed* $\mathsf{RealPowerSeries}^{\uparrow}$ *coefficient source, finite coefficient windows,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{ConvRad}^{\uparrow}$ *seal. The exactness row is a local witness inside the displayed packet, not a host limsup, quotient stream, selected radius representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier $K = (A, W, R, G, D, H, C, P, N)$ of Definition 3466.1. The coefficient source is A, the finite windows are W, the root-bound route is R, the tolerance ledger is D, and the radius seal is G.

Apply the root-bound handoff theorem. It orders every radius request through A, W, D, R, G before the structural rows H, C, P, N expose the local certificate to a consumer.

The Cauchy-Hadamard exactness witness is therefore read as part of the local $\mathsf{ConvRad}^{\uparrow}$ seal row. Since no carrier coordinate stores a host limsup, quotient coefficient stream, selected radius, or ambient completeness theorem, the NameCert route remains the displayed finite root-bound packet. □

Theorem 3466.5 (Cauchy-Hadamard radius nonescape)**.** *Every* $\mathsf{ConvRad}^{\uparrow}$ *read exported by an accepted* $\mathsf{CauchyHadamard}^{\uparrow}$ *packet factors through the displayed* $\mathsf{RealPowerSeries}^{\uparrow}$ *source row, finite coefficient windows,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *root-bound route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* BHist *histories,* hsame *transports,* Cont *replay,* Pkg *provenance, and local* NameCert *row. It exports no host limsup, real supremum row, quotient coefficient stream, countable-choice selector, selected radius representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the radius seal row G. By the carrier definition, G is reached only after the source row A, coefficient windows W, tolerance ledger D, and root-bound route R have been displayed.

Apply the NameCert obligations. They identify the structural support of the read as H, C, P, N and keep componentwise transport and continuation replay attached to the same finite packet.

No refused object has a row in the carrier. Therefore a downstream $\mathsf{ConvRad}^{\uparrow}$ consumer cannot escape the displayed Cauchy-Hadamard route by invoking a host limsup, real supremum, quotient stream, choice selector, selected radius, or ambient real-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyHadamard}^{\uparrow}$ packet is generated from a $\mathsf{RealPowerSeries}^{\uparrow}$ coefficient source row, finite coefficient windows, a $\mathsf{RegularCauchyRootTest}^{\uparrow}$ root-bound route, a $\mathsf{ConvRad}^{\uparrow}$ seal row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3466.1, Definition 3466.2, Definition 3466.3, and Definition 3466.5 bind the constructive power-series radius route over $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyHadamard}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close a Lean implementation, host limsup, real supremum row, quotient coefficient stream, countable-choice selector, selected radius representative, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, or bridge-checked standard model.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{ConvRad}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 3467

A Concrete Naming Certificate for $\mathsf{DyadicSubsequence}^{\uparrow}$

Following Bishop and Bridges, constructive subsequence extraction is read as a finite choice of inspected windows and monotone index evidence before any compactness conclusion is invoked. The $\mathsf{DyadicSubsequence}^{\uparrow}$ packet records that L10 selection surface over the bounded real sequence rows of chapter 3451, the finite observation route of chapter 126, the regular rational readback of chapter 301, and the compactness-facing route of chapter 1254. It packages dyadic tolerance windows, monotone unary reindexing, and Real-seal readback as displayed NameCert data rather than as an ambient subsequence selector or a completed compactness theorem.

Definition 3467.1 (Dyadic subsequence carrier)**.** *A* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, D, I, O, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the bounded real sequence source row, exposing only the displayed sequence name, bound certificate, and finite request interface.*
2. *W is the finite window ledger, recording the sequence windows inspected before each subsequence output is admitted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, storing dyadic comparison budgets for window selection, bound reuse, and later readback.*
4. *I is the monotone unary reindexing row, recording a finite increasing index map from output positions to inspected source positions.*
5. *O is the* $\mathsf{StreamName}^{\uparrow}$ *output-window row, exposing the selected subsequence only by finite windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from O, W, D, I.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each accepted readback request, exposed only after the source, window, tolerance, index, stream, and regular-readback rows are displayed.*
8. *H records componentwise* hsame *transport for the bounded sequence source, windows, dyadic ledger, monotone index row, stream windows, regular readback, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a subsequence or compactness-facing request through* $A, W, D, I, O, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{RealBoundedSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, D, I, O, R, E, H, C, P, N$. It has no coordinate for a global choice function, quotient stream, selected limit, ambient compactness theorem, or host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3467.2 (Dyadic subsequence classifier)**.** *Two accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carriers*

$$S = (A, W, D, I, O, R, E, H, C, P, N) \quad \text{and} \quad S' = (A', W', D', I', O', R', E', H', C', P', N')$$

are classified as the same local subsequence certificate exactly when the displayed bounded-source rows agree, the finite window ledgers present the same inspected source windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows agree on every recorded comparison budget, the monotone unary reindexing rows choose the same finite output-to-source positions, the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback rows agree after the recorded* hsame *transports, and the* Cont, Pkg, *and* NameCert *rows replay and name the same packet.*

Theorem 3467.3 (Dyadic subsequence NameCert obligations)**.** *For every accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *carrier $S = (A, W, D, I, O, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded sequence source, finite windows, dyadic tolerance rows, monotone index row, finite stream output, regular rational readback, terminal Real seals, componentwise transport, continuation replay, package provenance, and local naming. Every compactness-facing read of the packet is classified only by these displayed rows and exports no global selector, quotient stream, selected limit, ambient compactness theorem, or host Real-completeness coordinate.*

Proof. Carrier admission is projection from Definition 3467.1. The tuple displays the bounded source, inspected windows, dyadic budgets, monotone reindexing row, output stream windows, regular readback, Real seals, and structural rows.

The classifier of Definition 3467.2 fixes equality of accepted packets by those same rows. Thus a NameCert consumer cannot compare subsequence data by an off-packet selector or by an ambient limit object.

The structural obligations project H, C, P, N. They transport, replay, source, and name only the displayed packet, so the excluded global choice, quotient-stream, selected-limit, compactness, and Real-completeness coordinates remain absent. □

Lean4 ✓ `BEDC.Derived.DyadicSubsequenceUp.TasteGate.DyadicSubsequenceTasteGate single carrier alignment`

Theorem 3467.4 (Dyadic subsequence window extraction)**.** *Every finite output window exported by an accepted* $\mathsf{DyadicSubsequence}^{\uparrow}$ *packet factors through the route*

$$A \longmapsto W \longmapsto D \longmapsto I \longmapsto O \longmapsto R \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

The route first opens a bounded real sequence source, then records the inspected source windows, dyadic tolerances, and monotone unary reindexing before exposing the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback rows. No output window may be read directly from an ambient subsequence choice function or from a Bolzano-Weierstrass compactness conclusion.*

Proof. Start from an accepted carrier $S = (A, W, D, I, O, R, E, H, C, P, N)$. A finite output-window request is replayed through C, so it must expose the bounded source A, the source windows W, the dyadic tolerance ledger D, and the monotone index row I before it reads the output stream row O.

Regular rational readback and Real sealing occur after those rows. The packet reaches R and E only through the displayed stream window and tolerance route, with H preserving componentwise transport.

Finally P and N source and name the same finite route. Since no carrier coordinate stores an ambient selector or compactness theorem, every exported output window remains a finite window-extraction certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSubsequence}^{\uparrow}$ packet is generated from a bounded real sequence source, finite source windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, monotone unary reindexing, finite $\mathsf{StreamName}^{\uparrow}$ output windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3467.1, Definition 3467.2, Definition 3467.3, and Definition 3467.4 seed the finite L10 subsequence-selection surface between bounded real sequence windows and compactness-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean implementation, Bolzano-Weierstrass compactness, selected limits, quotient streams, countable choice, host Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance exactness, monotone index-window extraction, StreamName output, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicSubsequenceUp.

Chapter 3468

A Concrete Naming Certificate for $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names, the $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ packet records when a finite tail shift preserves the named regular Cauchy object as displayed BEDC data. It sits over the finite schedule rows of chapter 126, the regular rational classifier of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-shift packet of chapter 1578, and the terminal real-name seal of chapter 38. The packet is a NameCert surface for tail-shift equivalence of regular Cauchy names, not a quotient of streams, not selected host equality, and not an ambient completeness theorem.

Definition 3468.1 (Regular Cauchy shift equivalence carrier)**.** *A* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, k, S, R, D, H, C, P, N)$$

with the following visible rows:

1. *Q is the source regular Cauchy name row, presented through the same regular rational readback used by the original* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier.*
2. *k is the finite tail-shift index, displayed as a unary prefix length and carrying no unbounded selector.*
3. *S is the shifted* $\mathsf{StreamName}^{\uparrow}$ *row whose window request at n reads the source name at the displayed index $n + k$.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row for the shifted readback, attached to Q through the displayed schedule S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic slack preserved by the shifted readback.*
6. *H records componentwise* hsame *transport for the source name, shift index, shifted stream row, classifier row, dyadic ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an equivalence request through k, S, R, D, and the source row Q, and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected representatives, host sequence equality, countable-choice tail selectors, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyShiftEquivalenceUp.RegularCauchyShiftEquivalenceTasteGate`
`single carrier alignment`

Theorem 3468.2 (Regular Cauchy shift equivalence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier* $E = (Q, k, S, R, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source regular Cauchy name, finite shift index, shifted* $\mathsf{StreamName}^{\uparrow}$ *row, shifted* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, transports, continuation, provenance, and local naming row;*
2. *source-name retention, requiring the shifted classifier to read the same source regular Cauchy name Q after the finite index shift;*
3. *shifted-window exposure, requiring every request to pass through the displayed shift index k before it can inspect S;*
4. *classifier stability, requiring the shifted* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to remain attached to the displayed windows in S and the tolerance ledger D;*
5. *real-seal compatibility, requiring every* $\mathsf{Real}^{\uparrow}$*-facing read to pass through Q, k, S, R, D before any seal-facing consumer can inspect the structural rows;*
6. *ledger refusal, excluding quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3468.1. The displayed coordinates are exactly the source regular Cauchy name, finite shift index, shifted stream row, shifted regular-rational classifier, dyadic tolerance ledger, and the structural rows H, C, P, N.

Replay the continuation row C. An equivalence request first consumes the finite shift index k, then reads the shifted schedule S and the shifted classifier row R attached to the source row Q.

The dyadic ledger D is read on the same route. It supplies the tolerance data needed before a $\mathsf{Real}^{\uparrow}$-facing consumer can inspect the shifted readback as a real-name seal.

The remaining rows only transport, replay, package, and locally name that finite route. Since no coordinate stores quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 3468.3 (Regular Cauchy shift equivalence tail stability)**.** *Let* $E = (Q, k, S, R, D, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ *carrier. Every tail-equivalence consumer and every* $\mathsf{Real}^{\uparrow}$*-facing consumer factors through the finite route*

$$Q \longmapsto k \longmapsto S \longmapsto R \longmapsto D \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Thus the shifted regular Cauchy name remains stable under the displayed finite tail shift before quotient streams, host sequence equality, selected representatives, countable-choice tail selectors, or ambient completion data are available.

Proof. Begin with a consumer request for the shifted regular Cauchy name. Carrier admission supplies only the source row Q, the finite shift index k, the shifted schedule S, the classifier row R, the dyadic ledger D, and the structural rows.

The shifted schedule is read after k. Hence every displayed window of the shifted name is still a window of the source name at the translated finite index, and the classifier row R remains attached to that same readback.

The tolerance ledger D travels with the shifted read. Transport by H and replay by C preserve the route through Q, k, S, R, D, while P and N only record provenance and local naming for that route.

No refused object has a carrier coordinate. A tail-equivalence or $\mathsf{Real}^{\uparrow}$-facing consumer therefore receives the displayed finite shift route and no quotient stream, host sequence equality row, selected representative, choice selector, or ambient completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}$ packet is generated from a source regular Cauchy name row, finite tail-shift index, shifted StreamName row, RegSeqRat classifier row, DyadicRatCore tolerance ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3468.1, Definition 3468.2, and Definition 3468.3 seed the finite tail-shift equivalence route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyShiftEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected representatives, host sequence equality, countable-choice tail selection, ambient Real completeness, Lean verification, or a checked standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, shifted-window exposure, classifier stability, dyadic tolerance preservation, real-seal compatibility, and ledger refusal for BEDC.Derived.RegularCauchyShiftEquivalenceUp.

Chapter 3469

A Concrete Naming Certificate for $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ packet records the constructive diagonal extraction of a regular Cauchy subsequence as finite BEDC NameCert data. It sits beside the stream-window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the subsequence and interleaving routes of chapter 1854 and chapter 3405. The packet names a diagonal finite-window selector for one displayed family of tails; it is not host subsequence equality, a chosen limit, quotient equality of real names, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3469.1 (Diagonal Cauchy subsequence carrier)**.** *A* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, T, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *schedules whose rows display the source sequence windows and the tail windows available to the diagonal construction;*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback ledger attached to the same source and tail schedules;*
3. *T is the Cauchy subsequence extraction row, recording the finite increasing-index and tail-request data consumed from the* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *and* $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$ *sibling routes;*
4. *Q is the diagonal selector row, pairing each requested precision window with the subsequence tail whose regular rational readback is admitted for that window;*
5. *H records componentwise* hsame *transport for schedules, regular readbacks, extraction rows, diagonal selector rows, replay, provenance, and local naming;*
6. *C records displayed* Cont *replay from a diagonal request through S, R, T, Q, H before any* $\mathsf{Real}^{\uparrow}$ *seal is readable;*
7. *P is the* Pkg *provenance over* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySubsequenceExtra}$ $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$ *row.*

The carrier exports a finite diagonal extraction surface. It has no coordinate for host subsequence equality, selected limits, quotient equality of real names, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiagonalCauchySubsequenceUp.DiagonalCauchySubsequenceTasteGate single carrier alignment`

Theorem 3469.2 (Diagonal Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier* $D = (S, R, T, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$ *surface consists exactly of schedule admission for the* $\mathsf{StreamName}^{\uparrow}$ *rows in* S*; regular rational readback admission through* R*; subsequence-extraction admission through* T*; diagonal selector exactness through* Q*; preservation under the componentwise* hsame *rows of* H*; displayed* Cont *replay through* C*; provenance through* P*; and local naming through* N*. These obligations keep the diagonal selector inside the finite schedule, readback, extraction, transport, replay, provenance, and naming rows.*

Proof. Project the carrier. The schedules, regular readbacks, extraction row, selector row, transports, replay, provenance, and local name are the only visible coordinates of D.

Read the analytic rows in order. The $\mathsf{StreamName}^{\uparrow}$ schedules in S display the finite source and tail windows, R supplies the regular rational readbacks for those windows, and T records the subsequence extraction data supplied by the sibling subsequence and interleaving criteria.

The diagonal row Q is downstream of those reads. It may choose among displayed windows already admitted by S, R, T, but it has no socket for a host subsequence, a selected real limit, quotient equality, trichotomy, or ambient completeness.

Finally use H, C, P, N. Transport and replay preserve the same finite rows, and the package and local certificate name that route without adding any external diagonal principle. □

Theorem 3469.3 (Diagonal Cauchy subsequence regularity)**.** *For an accepted* $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ *carrier* $D = (S, R, T, Q, H, C, P, N)$*, every diagonal read exported to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer preserves the regular Cauchy windows displayed by* S *and* R*. The consumer route factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchySubsequenceExtraction}^{\uparrow},\ \mathsf{CauchyInterleavingCriterion}^{\uparrow},\ Q, H, C, P, N.$$

Consequently the packet cannot expose host subsequence equality, selected limits, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted diagonal read. The route must first inspect the finite schedules S and the attached regular rational readbacks R, so the read is already tied to displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows.

Next inspect the subsequence and interleaving row T. It records only finite increasing-index and tail-window evidence from the sibling Cauchy routes, and the diagonal selector Q can select only from those displayed rows.

The structural rows preserve the same regular windows. Componentwise hsame transport and displayed Cont replay carry S, R, T, Q together, while P, N source and name the accepted packet. Since no excluded object is a coordinate of the carrier, no consumer can obtain host subsequence equality, a selected limit, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the diagonal packet. □

Falsifiable prediction: Every accepted $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ read factors through displayed StreamName schedules, RegSeqRat readbacks, a Cauchy subsequence extraction row, a diagonal selector row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any RealUp seal is consumed.

Independence witness: streamname, regseqrat, real, cauchysubsequencecriterion, cauchy-interleaving-criterion: $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source schedules, regular readbacks, extraction data, a diagonal selector, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{DiagonalCauchySubsequence}^{\uparrow}$ packet is generated from StreamName schedules, RegSeqRat readbacks, a Cauchy subsequence extraction row, a diagonal selector row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3469.1, Definition 3469.2, and Definition 3469.3 seed the finite diagonal subsequence route through stream schedules, regular readbacks, and RealUp seal boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiagonalCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsequence equality, selected limits, quotient equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal selector exactness, regular-window preservation, non-escape, and bridge-facing consumer rows for BEDC.Derived.DiagonalCauchySubsequenceUp.

Chapter 3470

A Concrete Naming Certificate for $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$

The $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ packet realizes the trigger-support and coverage-rank rows of subsection 9462.28.21 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the boundary-tile classifier of chapter 3393, the punctured support skeleton of chapter 3463, and the inverse spectral route of chapter 2161. The packet records coverage witnesses for boundary-trigger rows; it is not a biological code table, a stochastic evolutionary model, a quotient codon space, or a hidden global semantics.

Definition 3470.1 (Six boundary tile coverage carrier)**.** *A* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, T, G, R, A, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source-window row exposing the six bit coordinates on which trigger support is read;*
2. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers;*
3. *G is the* $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$ *support graph row listing the puncture, bridge, CUN-square, and articulation rows used by coverage reads;*
4. *R is the coverage-rank row, recording which displayed boundary triggers cover which support faces and which rows remain uncovered;*
5. *A is the* $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$ *inverse-spectral handoff row, used only after W, T, G, R are displayed;*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, support graph rows, coverage-rank rows, inverse-spectral handoff, replay, provenance, and local naming;*
7. *C records displayed* Cont *replay from a trigger-coverage request through W, T, G, R, A, H;*
8. *P is the* Pkg *provenance over* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{PuncturedCubeSkeleton}^{\uparrow}$, $\mathsf{CodonWindowSpectralInverse}^{\uparrow}$, *the source vision of subsection 9462.28.21,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$ *row.*

The carrier accepts coverage only when every trigger read is exhausted by $W, T, G, R, A, H, C, P, N$. It has no coordinate for a hidden biological table, quotient codon equality, selected stochastic dynamics, or global semantic equivalence.

Theorem 3470.2 (Six boundary tile coverage NameCert obligations)**.** *For every accepted* SixBoundaryTileCoverage *carrier $K = (W, T, G, R, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$ *surface consists exactly of six-bit source-window admission through W, boundary-tile admission through T, punctured-skeleton support admission through G, coverage-rank exactness through R, inverse-spectral handoff through A, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.SixBoundaryTileCoverageUp.SixBoundaryTileCoverageTasteGate single carrier alignment`

Proof. Project the accepted carrier. The rows $W, T, G, R, A, H, C, P, N$ list all source-window, boundary-tile, support-graph, coverage-rank, inverse-spectral, transport, replay, provenance, and naming data available to the packet.

The trigger-support surface is finite. A coverage request first reads the six-coordinate source window W, then the boundary tile row T, and then the punctured support row G before the rank ledger R can be inspected.

The inverse-spectral handoff A is downstream of that support route. It may compare coverage rows with spectral inverse data, but it cannot replace them by a hidden code table or by stochastic semantics.

Finally H, C, P, N preserve and name the same finite route. Since every accepted coverage read is exhausted by those rows, no quotient codon space, selected dynamics, or global biological semantics is exported. □

Theorem 3470.3 (Six boundary tile trigger nonescape)**.** *Every accepted trigger-coverage read of a* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *packet factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{PuncturedCubeSkeleton}^{\uparrow},\ R,\ \mathsf{CodonWindowSpectralInverse}^{\uparrow},\ H$$

In particular, a trigger coverage claim cannot escape the displayed six-coordinate source, boundary-tile classifier, punctured support graph, coverage-rank row, inverse-spectral handoff, componentwise transport, replay, provenance, and local naming rows.

Proof. Start with the source rows. The six-bit window W exposes the coordinates on which a trigger is read, while T restricts the trigger to the displayed boundary-tile classifier.

Next read the support and rank rows. The punctured skeleton G supplies the support faces, and R records exactly which displayed triggers cover which faces. The spectral inverse row A is only a handoff from this route, not an alternate coverage source.

The structural rows close the route. Componentwise transport and replay preserve W, T, G, R, A together, while provenance and local naming source the accepted packet. Thus a consumer cannot obtain a hidden biological table, quotient codon equality, selected stochastic dynamics, or global semantic equivalence from a trigger-coverage read. □

Falsifiable prediction: A concrete trigger-coverage nonescape claim for $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ is that every accepted boundary-trigger read preserving $W, T, G, R, A, H, C, P, N$ remains inside the displayed six-coordinate source, boundary tile, punctured support graph, coverage-rank, and inverse-spectral route.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, punctured-cube-skeleton, codon-window-spectral-inverse: $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries trigger support, coverage rank, inverse-spectral handoff, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.SixBoundaryTileCoverageUp.TasteGate` must provide explicit Nontrivial $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ and FieldFaithful $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ instances preserving $W, T, G, R, A, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}$.

Constructive story: A $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ packet is generated from codon six-bit windows, boundary-tile classifiers, punctured support skeleton rows, coverage-rank rows, inverse-spectral handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3470.1, Definition 3470.2, and Definition 3470.3 seed the finite trigger-coverage packet over six-bit windows, boundary tiles, support skeletons, and spectral inverse rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SixBoundaryTileCoverage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological table identity, stochastic evolutionary modeling, quotient codon equality, global semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-support exactness, coverage-rank nonescape, inverse-spectral handoff, Nontrivial SixBoundaryTileCoverageUp, and FieldFaithful SixBoundaryTileCoverageUp for BEDC.Derived.SixBoundaryTileCoverageUp.

Chapter 3471

A Concrete Naming Certificate for $\mathsf{HeineBorelFiniteNet}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of compact intervals by finite approximation data, the $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ packet records the interval-cover certificate obtained from a located closed interval, a displayed finite net, a positive Lebesgue-number ledger, and the endpoint real-name boundary. It consumes the finite-net interval surface of chapter 1808, the compact-cover radius route of chapter 2301, the compact uniform-modulus route of chapter 3444, the terminal real-name seal of chapter 38, and the dyadic tolerance rows of chapter 291. The packet is a finite-net interval-cover certificate; it is not a general open-cover compactness theorem, an unrestricted Heine-Borel principle, a selected subcover functional, or an ambient completeness assertion.

Definition 3471.1 (Heine-Borel finite-net carrier)**.** *A* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, F, L, C, R, D, E, T, Q, P, N)$$

with the following displayed rows:

1. *I is the located closed interval source row, exposing the lower endpoint, upper endpoint, and endpoint order data as* $\mathsf{Real}^{\uparrow}$ *histories.*
2. *F is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *finite-net row, listing the displayed mesh scale and finite interval net consumed by the cover request.*
3. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *positive-radius row attached to the displayed finite cover or pointwise radius family.*
4. *C is the* $\mathsf{HeineCantor}^{\uparrow}$ *compact-uniformity row that may be read only as a sibling route through the same finite-net and radius data.*
5. *R is the cover-read row, assigning each listed net point to its displayed cover witness and recording the finite neighborhood test used for nearby interval points.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording mesh width, endpoint containment, radius comparison, and finite cover slack checks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and cover-boundary reads.*
8. *T records componentwise* hsame *transport for the interval source, finite-net row, Lebesgue-number row, Heine-Cantor row, cover-read row, dyadic ledger, Real seal, replay row, provenance row, and local naming row.*

9. *Q records displayed* Cont *replay from an interval-cover request through I, F, L, R, D, E and, when needed by a uniform-continuity consumer, through the sibling row C.*

10. *P is the* Pkg *provenance over* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, C, R, D, E, T, Q, P, N$. It has no coordinate for arbitrary open covers, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineBorelFiniteNetUp.HeineBorelFiniteNetTasteGate single carrier alignment`

Theorem 3471.2 (Heine-Borel finite-net NameCert obligations)**.** *For every accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the located interval source, finite-net row, Lebesgue-number row, Heine-Cantor sibling row, cover-read row, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*

2. *interval finite-net exposure, requiring the cover request to display I and F before any cover witness is read;*

3. *radius compatibility, requiring the positive-radius row L and dyadic ledger D to control mesh width, endpoint containment, cover slack, and finite neighborhood tests;*

4. *sibling uniformity compatibility, requiring the* $\mathsf{HeineCantor}^{\uparrow}$ *row C to be read only as a sibling route over the same finite-net and radius rows;*

5. *non-escape, requiring T, Q, P, N to transport, replay, source, and name only the displayed finite packet and to export no arbitrary open-cover compactness, unrestricted Heine-Borel principle, selected subcover, countable-choice row, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is projection from Definition 3471.1. The tuple displays exactly the interval source, finite net, positive-radius row, sibling uniformity route, cover reads, dyadic tolerance ledger, Real seal, and structural rows.

The interval-cover computation first reads I and F. Thus every point-facing cover read is tied to the displayed located interval and to a listed finite mesh row before any neighborhood witness can be inspected.

The radius and cover checks are finite. The row L supplies the positive radius ledger, while D records the mesh-width and containment comparisons that let R transfer a cover witness from a listed net point to the interval point under inspection.

The sibling uniformity row C is only a compatibility route. It can be consumed by a Heine-Cantor-style downstream reader, but it does not replace the finite-net cover row or introduce an open-cover compactness principle.

Finally, the refused objects have no carrier coordinate. Since T, Q, P, N transport, replay, source, and name only the displayed packet, no accepted $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ certificate exports unrestricted Heine-Borel compactness, a selected subcover, countable choice, host equality, or ambient Real completeness. □

Theorem 3471.3 (Heine-Borel finite-net interval-cover route)**.** *Every interval-cover read from an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{HeineCantor}^{\uparrow}$ *row is available as the sibling compact-uniformity route over the same finite-net and radius data, but the exported certificate remains the displayed finite-net interval-cover route.*

Proof. Project the accepted carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$. The interval source and finite-net row are I and F, the positive-radius row is L, the dyadic comparison ledger is D, and the terminal endpoint seal is E.

Replay through Q keeps that order. A cover request opens the finite net, reads the positive-radius ledger, checks the dyadic slack against the mesh and endpoint rows, and only then exposes the Real-facing endpoint seal.

The row C records the compatible Heine-Cantor route for consumers that also need compact uniformity. It remains tied to the same finite rows, so it cannot introduce a selected subcover, an unrestricted open-cover theorem, or an ambient completeness coordinate. □

Definition 3471.4 (Heine-Borel finite-net grid refinement)**.** *Let* $H = (I, F, L, C, R, D, E, T, Q, P, N)$ *be an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *carrier and let a displayed dyadic submesh refine the finite net row* F*. The grid refinement is the packet obtained by transporting the positive-radius row* L *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger* D *while keeping the located interval source* I*, cover-read row* R*, endpoint seal* E*, transport row* T*, replay row* Q*, provenance row* P*, and local name row* N *unchanged. The refinement preserves the same finite mesh route and does not add any selected subcover coordinate.*

Theorem 3471.5 (Heine-Borel finite-net grid refinement stability)**.** *Refining the finite net row* F *of an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet by a displayed dyadic submesh and transporting the positive-radius row* L *through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger yields another accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet with the same* $\mathsf{Real}^{\uparrow}$ *endpoint seal and the same nonescape boundary. The refined packet still factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

and it exports no selected subcover, quotient interval equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3471.1. The finite net row F and the positive-radius row L are the only rows changed by the displayed dyadic submesh.

Replay Definition 3471.3. It forces the refined mesh to be read through the located interval source, the positive-radius ledger, the dyadic comparison row, and the Real seal in the same order as the unrefined packet.

Now use the grid-refinement packet itself. Because the refinement keeps I, R, E, T, Q, P, N unchanged and only retunes the mesh through L and D, the consumer still sees the same finite-net interval-cover route and no selected subcover or ambient completeness coordinate. □

Theorem 3471.6 (Heine-Borel finite-net obligation lattice)**.** *Any accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet must be read in the obligation order*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{HeineCantor}^{\uparrow}$ *row is only a sibling uniformity consumer over the same finite interval, positive-radius, dyadic, and Real-seal rows. The packet does not export general open-cover compactness, unrestricted Heine-Borel, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (I, F, L, C, R, D, E, T, Q, P, N)$. The finite interval source and net rows are I and F, the positive-radius row is L, the dyadic tolerance ledger is D, and the terminal Real-facing seal is E.

The cover-read row R can be consumed only after the interval and finite-net rows have been displayed, and the replay row Q keeps the Lebesgue-number row ahead of every dyadic slack check. Thus the finite-net obligation surface is ordered by interval net, positive radius, dyadic comparison, and Real seal.

The row C is read differently. It is available to a $\mathsf{HeineCantor}^{\uparrow}$ consumer that needs compact uniformity over the same finite rows, but it does not precede the finite-net route and does not replace F, L, D, E.

Finally the structural rows T, P, N transport, source, and name only the displayed packet. Since the refused objects have no coordinate in H, the obligation lattice cannot escape to unrestricted compactness, selected subcovers, quotient representatives, countable choice, host equality, or ambient completeness. □

Theorem 3471.7 (Heine-Borel finite-net scope witness)**.** *Every accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet reads only the displayed located-closed-interval finite-net row, the Lebesgue-number positive-radius row, the Heine-Cantor sibling row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. Thus the chapter scope is closed at the finite-net interval-cover certificate boundary and does not extend to general open-cover compactness, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition Definition 3471.1 lists exactly the rows $I, F, L, C, R, D, E, T, Q, P, N$ and states that the classifier compares accepted packets only through those rows.

Apply Definition 3471.2 to identify the local NameCert obligations, Definition 3471.3 to force every interval-cover read through the located interval, finite net, positive radius, dyadic ledger, and Real seal, and Definition 3471.6 to keep the Heine-Cantor row as a sibling uniformity route over the same finite data.

The refused objects have no coordinate in the carrier and are excluded by all three displayed theorem blocks. Therefore the accepted packet reads exactly the stated finite-net scope. □

Theorem 3471.8 (Heine-Borel finite-net subcover extraction)**.** *The finite subcover row consumes the finite-net interval carrier of chapter 1808 and the positive-radius witness of chapter 2301 before the extracted witness list reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Let $H = (I, F, L, C, R, D, E, T, Q, P, N)$ *be an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet. If the cover-read row* R *reaches the displayed finite net* F*, then the packet extracts a finite subcover row only through the route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow},\quad \mathsf{LebesgueNumberLemma}^{\uparrow},\quad \mathsf{HeineCantor}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{hsame},$$

The extracted row is the finite list of cover witnesses already recorded by R *for the net points and transported to interval points through* L, D, E, T, Q, P, N*. It is not an arbitrary open-cover*

compactness theorem, a selected subcover functional, a quotient-interval construction, a countable-choice principle, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

The same extracted finite subcover row is the cover input consumed by $\mathsf{ContinuousIntervalImage}^{\uparrow}$ *through chapter 1897 when interval-image connectedness or compact interval-image routes need finite cover data.*

Proof. Project the carrier of Definition 3471.1. The only cover-facing rows are the located interval source I, the finite net F, the positive-radius ledger L, the sibling uniformity row C, the cover-read row R, the dyadic ledger D, and the Real seal E.

Use Definition 3471.2 to keep every accepted cover read inside the local NameCert obligations. In particular, the cover request must display the interval and finite-net rows before a witness from R can be inspected, and the radius compatibility check must pass through L and D.

Apply Definition 3471.3 to order the extraction. The finite list of witnesses is read from R along the located finite net, then transported across the Lebesgue-number and dyadic slack rows to the endpoint Real seal.

Finally apply Definition 3471.7. It keeps the $\mathsf{HeineCantor}^{\uparrow}$ row as a sibling compact-uniformity route and confines transport, replay, provenance, and naming to $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$. Since the refused objects have no carrier coordinate, the extracted finite subcover row cannot be read as arbitrary compactness, selected choice, quotient interval data, countable choice, or ambient Real completeness. □

Theorem 3471.9 (Heine-Borel finite-net obligation-closure package)**.** *The exact obligation-closure surface of a* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet is the package consisting of the carrier of Definition 3471.1, the NameCert obligations of Definition 3471.2, the interval-cover route of Definition 3471.3, the obligation lattice of Definition 3471.6, the scope witness of Definition 3471.7, and the subcover extraction row of Definition 3471.8. Any downstream interval compactness consumer must read those rows as one finite packet over* $I, F, L, C, R, D, E, T, Q, P, N$ *and cannot replace them by arbitrary open-cover compactness, a selected subcover functional, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First project the carrier. It fixes the displayed rows $I, F, L, C, R, D, E, T, Q, P, N$ and leaves no coordinate outside the located interval, finite-net, positive-radius, sibling uniformity, cover-read, dyadic, Real-seal, transport, replay, provenance, and local naming data.

Next apply the NameCert obligations, interval-cover route, and obligation lattice. These rows force the cover request to pass through $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with $\mathsf{HeineCantor}^{\uparrow}$ available only as a sibling uniformity route over the same finite data.

Finally apply the scope witness and subcover extraction theorem. They identify the exportable finite subcover row as the witnessed cover reads already recorded by R and transported through L, D, E, T, Q, P, N. Since no refused object appears in the carrier, the displayed package is exactly the obligation-closure surface. □

Theorem 3471.10 (Heine-Borel finite-subcover extraction scope)**.** *Any finite subcover row exported by an accepted* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *packet remains scoped to* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$. *It may be read only through the displayed rows* $I, F, L, C, R, D, E, T, Q, P, N$*, so it is not arbitrary open-cover compactness, a choice-selected subcover, quotient interval equality, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 3471.8. The extracted row is already the finite list of witnesses recorded by R for the displayed finite net and transported to interval points through the positive-radius and dyadic slack rows.

Use Definition 3471.9 to keep that row inside the whole packet. The admissible dependencies are exactly the located finite-net source, Lebesgue-number radius, Heine-Cantor sibling route, dyadic ledger, Real seal, and structural $\mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$ rows.

The refused readings have no carrier coordinate and no replay route. Therefore the exported finite subcover row cannot be detached from the finite-net interval certificate or promoted into unrestricted compactness or choice-selected subcover data. □

Theorem 3471.11 (Heine-Borel finite-net primitive induction route)**.** *Let $H = (I, F, L, C, R, D, E, T, Q, P, N)$ be an accepted $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ packet. The finite-net cell list displayed by F supports primitive finite induction from the first accepted cell through each subsequent Cont replay row in Q. At every step the induction reads only the current net cell, the corresponding radius check in L, the dyadic slack row in D, the retained cover read in R, and the endpoint seal in E. The resulting induction route is stable under componentwise hsame transport in T and remains inside the provenance and local naming ledger P, N.*

Proof. Project the carrier of Definition 3471.1. The finite list is the displayed row F, and the replay row Q records how a cover request advances through the listed cells.

The base step reads the first accepted cell together with its positive-radius row from L, dyadic slack row from D, cover witness from R, and endpoint seal from E. The successor step repeats the same displayed read for the next cell and adjoins the resulting witness to the finite subcover row.

Componentwise transport through T preserves each cell, radius, slack, cover-read, and endpoint-seal row. Since P and N name only the same displayed packet, the induction route cannot leave the local NameCert ledger or import an arbitrary open-cover compactness principle. □

Theorem 3471.12 (Heine-Borel choice-free located net)**.** *For every accepted $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ carrier*

$$H = (I, F, L, C, R, D, E, T, Q, P, N),$$

the extracted located-net cover row is read only from the displayed located interval source I, finite-net row F, Lebesgue-number row L, Heine-Cantor sibling row C, DyadicRatCore ledger D, Real seal E, componentwise hsame transport T, displayed Cont replay Q, Pkg provenance P, and local $\mathsf{NameCert}$ row N. The route packages a finite located-net handoff and does not claim general open-cover compactness, unrestricted Heine-Borel compactness, selected subcover functionals, quotient interval representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. Begin with the finite subcover extraction theorem Definition 3471.8. It identifies the exported row as the finite list of cover witnesses already recorded for the displayed finite net and transported across the positive-radius, dyadic, endpoint, structural, provenance, and naming rows.

The finite-subcover extraction scope of Definition 3471.10 keeps that row inside the located interval, Lebesgue-number, Heine-Cantor sibling, DyadicRatCore, Real, hsame, Cont, Pkg, and $\mathsf{NameCert}$ surface. Thus no external subcover selector or open-cover compactness object is available.

Finally apply the primitive induction route of Definition 3471.11. The witness list is assembled by finite induction over the displayed cells of F, with each step reading only the current radius check, dyadic slack row, retained cover read, and endpoint seal. Therefore the located-net handoff is choice-free and remains confined to the accepted packet rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ packet is generated from a located closed interval source, a LocatedClosedIntervalFiniteNet row, a LebesgueNumberLemma positive-radius ledger, a HeineCantor sibling route, finite cover reads, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3471.1, Definition 3471.2, and Definition 3471.3 seed the constructive finite-net interval-cover certificate over located interval, positive-radius, dyadic tolerance, Heine-Cantor sibling, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, unrestricted Heine-Borel compactness, selected subcovers, quotient interval representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net interval exposure, radius compatibility, sibling uniformity compatibility, cover-read exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelFiniteNetUp.

Chapter 3472

A Concrete Naming Certificate for $\mathsf{HilbertBasis}^{\uparrow}$

Following Hilbert's basis theorem and constructive commutative algebra, the $\mathsf{HilbertBasis}^{\uparrow}$ packet records the finite generating rows needed to read a polynomial ideal as a named basis surface. It sits after the polynomial constructors of chapter 50, the ring and commutative-ring ledgers of chapter 43 and chapter 44, the ideal carrier of chapter 81, and the finite reduction route of chapter 2197.

Definition 3472.1 (HilbertBasis carrier)**.** *A* $\mathsf{HilbertBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{HB}(P, R, C, I, G, B, H, K, Q, N)$$

with the following rows.

- P *is the* $\mathsf{Poly}^{\uparrow}$ *source row for the ambient polynomial syntax and monomial ledger.*
- R *is the* $\mathsf{Ring}^{\uparrow}$ *source row for coefficient addition, multiplication, zero, one, and additive inverse.*
- C *is the* $\mathsf{CommRing}^{\uparrow}$ *row that makes coefficient multiplication commute for the displayed polynomial operations.*
- I *is the* $\mathsf{Ideal}^{\uparrow}$ *row naming the finite ideal-membership and generator surface.*
- G *is the* $\mathsf{GroebnerBasis}^{\uparrow}$ *reduction row used only as a finite normal-form witness.*
- B *is the displayed Hilbert-basis list of polynomial generators, together with the bounded reduction traces that place each ideal row under that list.*
- H, K, Q, N *are the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$ *naming rows.*

The packet has no coordinate for quotient ideals, host noetherianity, excluded middle over ideals, selected infinite generator search, or choice-produced basis extraction.

Lean4 ✓ `BEDC.Derived.HilbertBasisUp.HilbertBasisTasteGate single carrier alignment`

Theorem 3472.2 (HilbertBasis NameCert obligations)**.** *For an accepted* $\mathsf{HilbertBasis}^{\uparrow}$ *carrier* $\mathsf{HB}(P, R, C, I, G, B, H, K, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, polynomial-source compatibility through* P*, coefficient-ring compatibility through* R *and* C*, ideal generator access through* I*, finite reduction replay through* G*, basis-list exactness through* B*, componentwise transport through* H*, continuation replay through* K*, package provenance through* Q*, and local naming through* N*.*

Proof. Project the carrier of Definition 3472.1. The only rows available to the local certificate are $P, R, C, I, G, B, H, K, Q, N$.

The polynomial, ring, commutative-ring, and ideal rows supply the algebraic source surface. The Groebner row supplies finite reduction traces only after those source rows have been displayed.

The basis-list row B records the finitely many generators and the reductions that witness their coverage. Transport, continuation, provenance, and local naming then repack the same finite rows, so quotient ideals, host noetherianity, and choice data cannot enter the certificate. □

Theorem 3472.3 (HilbertBasis polynomial-ideal route)**.** *Every downstream consumer of an accepted* $\mathsf{HilbertBasis}^{\uparrow}$ *packet that asks for a polynomial-ideal basis must read the route*

$$\mathsf{Poly}^{\uparrow} \longrightarrow \mathsf{Ring}^{\uparrow} \longrightarrow \mathsf{CommRing}^{\uparrow} \longrightarrow \mathsf{Ideal}^{\uparrow} \longrightarrow \mathsf{GroebnerBasis}^{\uparrow} \longrightarrow \mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}.$$

The exported object is the finite generator row B together with its displayed reduction traces and structural rows H, K, Q, N; it is not a quotient ideal, not an infinite ascending-chain argument, not a host noetherian theorem, and not a selected basis obtained by choice.

Proof. Start with the polynomial row P and coefficient rows R, C. These determine the syntax and coefficient operations that an ideal consumer is allowed to inspect.

Next read the ideal row I and the finite Groebner row G. The ideal surface provides the finite membership and generator coordinates, while the Groebner route provides terminating reduction traces for the displayed basis list.

Finally apply the NameCert obligations of Definition 3472.2. They force the exported basis to remain inside B, H, K, Q, N, so no quotient ideal, no ambient noetherian principle, and no choice-produced generator search is exported. □

Falsifiable prediction: Every $\mathsf{HilbertBasis}^{\uparrow}$ consumer that claims a polynomial-ideal basis must exhibit the polynomial row, ring row, commutative-ring row, ideal row, Groebner reduction row, basis-list row, transport, continuation, provenance, and local naming row.

Independence witness: polynomial, ring, commring, ideal, groebner-basis: $\mathsf{HilbertBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries polynomial syntax, coefficient algebra, ideal membership, finite reduction traces, and a named basis-list surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{HilbertBasis}^{\uparrow}$ packet is generated from PolynomialUp syntax rows, RingUp and CommRingUp coefficient rows, an IdealUp generator surface, a GroebnerBasisUp reduction row, a finite basis-list row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$, seed)

Scope closed: Definition 3472.1, Definition 3472.2, and Definition 3472.3 seed the finite polynomial-ideal basis interface over PolynomialUp, RingUp, CommRingUp, IdealUp, GroebnerBasisUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HilbertBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient ideals, host noetherianity, excluded middle over ideals, selected infinite generator search, choice-produced basis extraction, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, polynomial and coefficient compatibility, ideal generator access, finite reduction replay, basis-list exactness, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.HilbertBasisUp.

Chapter 3473

A Concrete Naming Certificate for $\mathsf{CantorBernstein}^{\uparrow}$

Following Cantor, Bernstein, and Schroeder, the $\mathsf{CantorBernstein}^{\uparrow}$ packet records the finite BEDC route by which two displayed injection classifiers determine one mutual-injection equivalence boundary. It sits beside the equality and comparison surfaces of chapter 1404, the stream-window classifier surface of chapter 126, and the governance naming discipline of chapter 9467. The packet exposes classifier rows, finite zig-zag decomposition data, BHist inspection, hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming; it is not a host quotient equality theorem, a selected bijection from ambient set theory, or a hidden choice principle.

Definition 3473.1 (Cantor–Bernstein carrier)**.** *A* $\mathsf{CantorBernstein}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, I, J, X, Y, Z, H, C, P, N)$$

with the following displayed rows:

1. *A is the first classifier surface, naming the observed source-side rows and their equality test before any target-side identification is read.*
2. *B is the second classifier surface, naming the observed target-side rows and their equality test before any source-side identification is read.*
3. *I is the displayed injection row from A into B, including source classifier preservation and target classifier reflection for every inspected finite row.*
4. *J is the displayed injection row from B into A, including target classifier preservation and source classifier reflection for every inspected finite row.*
5. *X is the finite zig-zag decomposition ledger. It separates forward chains, backward chains, anchored components, and unresolved boundary components using only the two displayed injection rows.*
6. *Y is the classifier-equivalence route read from X. It records which side supplies the representative for each finite zig-zag component without identifying host quotient classes.*
7. *Z is the mutual-injection boundary row exported to later classifier consumers.*
8. *H records componentwise* hsame *transport for the two classifier surfaces, the two injection rows, the zig-zag decomposition ledger, the classifier-equivalence route, and the exported boundary row.*

9. *C records displayed* Cont *replay from classifier inspection through injection preservation, zig-zag decomposition, and boundary export.*
10. *P is the* Pkg *provenance over* $\mathsf{CantorBernstein}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the sibling classifier chapters cited above.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$ *row.*

The carrier compares the two classifier surfaces only through the rows listed here. It exports no host quotient of classifiers, no ambient set equality, no selected global bijection, no arbitrary chain choice, and no completed StreamName or Real equivalence theorem.

Lean4 ✓ `BEDC.Derived.CantorBernsteinUp.CantorBernsteinTasteGate single carrier alignment`

Theorem 3473.2 (Cantor–Bernstein NameCert obligations)**.** *For every accepted* $\mathsf{CantorBernstein}^{\uparrow}$ *carrier*

$$K = (A, B, I, J, X, Y, Z, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$ *surface consists exactly of carrier admission, two displayed classifier surfaces, two injection rows, finite zig-zag decomposition, classifier-equivalence routing, exported mutual-injection boundary data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the boundary row Z must factor through*

$$A, B, I, J \longmapsto X, Y, Z \longmapsto H, C, P, N.$$

Thus the mutual-injection theorem surface is available as finite BEDC certificate data while refusing host quotient equality, ambient set extensionality, arbitrary chain choice, and a selected global bijection.

Proof. Project the carrier of Definition 3473.1. The two classifier surfaces A and B are visible before either injection row is consumed, so no side is treated as a hidden quotient of the other.

Read the injections next. The row I records classifier preservation and reflection from A to B, while J records the corresponding route from B to A. These two rows are the only admissible source of mutual reachability data.

Build the finite zig-zag ledger X from the displayed reachability rows. The ledger separates chain components by finite inspection and hands the representative-side decision to Y without invoking an ambient chain selector.

The exported row Z is then a boundary for downstream classifier consumers. Transport by H and replay by C preserve the same displayed rows, while P and N record provenance and local naming. No additional quotient, extensionality, or choice row is present in the packet. □

Falsifiable prediction: Removing either displayed injection row or the finite zig-zag decomposition ledger prevents a consumer from distinguishing mutual-injection classifier equivalence from a host quotient or selected bijection.

Independence witness: regularcauchyequivalence, streamname, real: none is bijective to $\mathsf{CantorBernstein}^{\uparrow}$ because the packet jointly carries two classifier surfaces, two displayed injection rows, a finite zig-zag decomposition ledger, classifier-equivalence routing, boundary export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorBernstein}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorBernstein}^{\uparrow}$ packet is generated from two displayed classifier surfaces, two injection rows, finite zig-zag decomposition data, classifier-equivalence routing, a mutual-injection boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CantorBernstein}\uparrow}$, seed)

Scope closed: Definition 3473.1 and Definition 3473.2 seed the finite mutual-injection classifier-equivalence boundary over displayed BEDC rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CantorBernstein}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient equality, ambient set extensionality, arbitrary chain choice, a selected global bijection, StreamName equivalence, Real equivalence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two classifier surfaces, two injection rows, finite zig-zag decomposition, classifier-equivalence routing, boundary export, transport, replay, provenance, and local naming for BEDC.Derived.CantorBernsteinUp.

Chapter 3474

A Concrete Naming Certificate for $\mathsf{HyperbolicMetric}^{\uparrow}$

Following Gromov's four-point metric viewpoint and Kelley's topology-side use of metric balls, the $\mathsf{HyperbolicMetric}^{\uparrow}$ packet realizes hyperbolic metric geometry as finite BEDC certificate data. It sits after the metric readback of chapter 57, the center-radius surface of chapter 1450, the ordered-real comparison rows of chapter 38, the topology handoff of chapter 88, and the bounded-set ledger of chapter 1757. The chapter names a finite four-point metric surface without turning the host disk, smooth hyperbolic manifold, ideal boundary, or ambient metric completion into a primitive model.

Definition 3474.1 (Hyperbolic metric carrier)**.** *A* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, b, t, \beta, q, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite* $\mathsf{Met}^{\uparrow}$ *carrier row whose displayed distance comparisons are the only metric data read by the packet.*
2. *d is the* $\mathsf{Real}^{\uparrow}$ *valued distance row, naming the finite comparisons used by the four-point ledger.*
3. *b is the* $\mathsf{MetricBall}^{\uparrow}$ *neighborhood row, exposing center-radius packets that can be transported without an ambient powerset.*
4. *t is the* $\mathsf{Topology}^{\uparrow}$ *handoff row, reading the ball-neighborhood surface as an indexed-open request.*
5. *β is the* $\mathsf{BoundedSet}^{\uparrow}$ *boundary ledger, recording bounded containment data for the displayed ball rows.*
6. *q is the four-point inequality row, comparing the three paired sums determined by four displayed metric observations.*
7. *F is the refusal row excluding smooth hyperbolic manifolds, quotient ideal boundaries, host geodesic completions, ambient Real completeness, physical cosmology, and internal observer-substance readings.*
8. *H records componentwise* hsame *transport for the metric carrier, distance, ball, topology, bounded-set, four-point, and refusal rows.*
9. *C records displayed* Cont *replay through finite metric observations, ball-neighborhood transport, topology handoff, bounded containment, four-point comparison, provenance, and refusal.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$ *row.*

The carrier exports a hyperbolic metric-read packet only through these rows. It contains no primitive smooth hyperbolic space, no quotient boundary, no completed geodesic model, no host powerset of balls, no ambient Real completeness principle, and no object-level observer substance.

Theorem 3474.2 (Hyperbolic metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier*

$$M = (X, d, b, t, \beta, q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$ *surface consists exactly of metric-carrier admission, real-valued distance readback, metric-ball neighborhood transport, topology handoff, bounded-set boundary ledger, four-point inequality exposure, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every metric consumer must factor through*

$$X, d, b \longmapsto t, \beta, q, F \longmapsto H, C, P, N.$$

Thus hyperbolic geometry is read as a finite four-point metric certificate, not as a primitive ambient model.

Proof. Project the carrier of Definition 3474.1. The metric carrier row X is visible before any distance is read, and the distance row d is tied to displayed $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ observations.

Read the ball, topology, and bounded rows next. The row b records center-radius neighborhoods, t exposes the topology-side indexed-open request, and β records the bounded-set boundary ledger without creating a host subset.

The four-point row q is the only hyperbolic comparison row. The refusal row F keeps smooth hyperbolic geometry, quotient boundary equality, completed geodesic space, physical cosmology, and ambient Real completeness outside the packet.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier has no coordinate for a primitive smooth hyperbolic space, quotient boundary, completed metric model, physical cosmology, or observer substance, no consumer can extract those readings from the NameCert surface. □

Lean4 ✓ `BEDC.Derived.HyperbolicMetricUp.HyperbolicMetricCarrier namecert obligations`

Theorem 3474.3 (Hyperbolic metric four-point obligation surface)**.** *For every accepted* $\mathsf{HyperbolicMetric}^{\uparrow}$ *carrier* $M = (X, d, b, t, \beta, q, F, H, C, P, N)$*, carrier admission, four-point inequality exposure, ball-neighborhood transport, bounded-set boundary reading, topology handoff, and non-escape from smooth hyperbolic geometry or host metric completion are available only through the displayed rows*

$$X, d, b, t, \beta, q, F, H, C, P, N.$$

The row q may compare four displayed points by the usual hyperbolic four-point ledger, but it may not import a smooth manifold, an ideal-boundary compactification, a selected geodesic model, a host powerset of balls, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the projection of Definition 3474.1. It exposes X as the metric carrier and d as the real-valued distance row before any hyperbolic comparison is inspected.

The four-point surface is row q. It reads only the finite distance comparisons supplied by d, so the hyperbolic inequality is a ledger entry over displayed observations rather than a theorem about an ambient smooth model.

Ball-neighborhood transport uses b together with H and C. The same transported center-radius rows are handed to t for the topology request and to β for bounded-set boundary accounting.

The refusal row F is then preserved by the structural rows H, C, P, N. Hence every consumer receives a finite NameCert surface over metric, ball, real, topology, bounded-set, transport, replay, provenance, and naming rows, with no coordinate from which smooth hyperbolic geometry or host metric completion can be recovered. □

Falsifiable prediction: Omitting the metric carrier row, distance row, metric-ball row, or four-point row prevents carrier acceptance because the packet can no longer distinguish finite hyperbolic metric reading from a primitive host geometry.

Independence witness: metric, metricball, real, topology, boundedset: none is bijective to $\mathsf{HyperbolicMetric}^{\uparrow}$ because the packet jointly carries metric carrier data, real-valued distance reads, ball-neighborhood rows, topology handoff, bounded-set boundary accounting, four-point comparison, refusal exactness, transport, replay, package sourcing, and local naming.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial\ HyperbolicMetric}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicMetric}^{\uparrow}$ TasteGate instances. Those instances must keep $X, d, b, t, \beta, q, F, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicMetric}^{\uparrow}$ packet is generated from Metric carrier observations, Real distance reads, MetricBall neighborhoods, Topology handoff, BoundedSet boundary accounting, a four-point comparison ledger, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3474.1, Definition 3474.2, and Definition 3474.3 seed the finite four-point metric surface through metric, ball, real, topology, bounded-set, refusal, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close smooth hyperbolic manifold geometry, quotient boundary equality, host metric completion, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric rows, ball rows, real comparison rows, topology handoff, bounded-set boundary accounting, four-point ledger exactness, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicMetricUp.

Chapter 3475

A Concrete Naming Certificate for $\mathsf{DenseIsometryExtension}^{\uparrow}$

Following Bishop and Bridges on constructive completions and isometric extension, the $\mathsf{DenseIsometryExtension}^{\uparrow}$ packet records the finite surface by which an isometry on a dense displayed source is extended through the completion interface. It sits after the entry surface of chapter 1512, the extension-and-uniqueness surface of chapter 397, the separated comparison boundary of chapter 1411, the regular rational readback of chapter 301, the finite windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names only the displayed dense-source extension route; it does not construct quotient completions, select limit representatives, assert host function equality, import ambient completeness, or add bridge checking.

Definition 3475.1 (Dense isometry extension carrier)**.** *A* $\mathsf{DenseIsometryExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, J, U, S, F, W, T, R, H, C, P, N)$$

with the following visible rows:

1. *D is the dense-source row, exposing the displayed source observations and their finite density witnesses.*
2. *J is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *entry row that sends dense-source observations into the completion-facing surface.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-and-uniqueness row attached to the same dense source and target route.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row that makes zero-distance agreement a displayed classifier fact.*
5. *F is the source isometry row, recording finite distance preservation on the dense source through* $\mathsf{Real}^{\uparrow}$ *readback rows.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect dense-source names.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling approximation requests.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row whose observations feed the terminal real seal.*
9. *H records componentwise* hsame *transport for dense source, completion entry, universal-property, separatedness, isometry, finite-window, tolerance, regular-rational, and real-seal rows.*

10. *C records displayed* Cont *replay through the same rows.*

11. *P is the* Pkg *provenance over* DenseIsometryExtension$^{\uparrow}$, CompletionEmbedding$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{DenseIsometryExtension}^{\uparrow}}$ *row.*

The carrier exports an extension service only through these rows. It contains no quotient completion object, no selected limit, no host function equality, no unrecorded completeness theorem, and no hidden choice schedule.

Lean4 ✓ `BEDC.Derived.DenseIsometryExtensionUp.TasteGate.DenseIsometryExtensionTasteGate single carrier alignment`

Theorem 3475.2 (Dense isometry extension NameCert obligations)**.** *For every accepted* DenseIsometryExtension$^{\uparrow}$ *carrier*

$$E = (D, J, U, S, F, W, T, R, H, C, P, N),$$

the local NameCert$_{\text{DenseIsometryExtension}^{\uparrow}}$ *surface consists exactly of dense-source admission, completion-entry exposure, universal-property extension, separated comparison, source isometry preservation, finite-window exposure, dyadic tolerance accounting, regular-rational handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every extension consumer must factor through*

$$D, J, F, W, T, R \longmapsto U, S \longmapsto H, C, P, N.$$

Thus the extension is read as finite NameCert data over the dense source and its completion route, not as a host-level function on a quotient completion.

Proof. Project the carrier of Definition 3475.1. The dense-source row D, completion-entry row J, source isometry row F, finite-window row W, dyadic ledger T, and regular-rational row R are the only rows from which a source observation can be read.

Next use the universal-property row U. It repacks those displayed observations as an extension route, while the separated comparison row S supplies the zero-distance classifier boundary needed for uniqueness of the extended isometry.

Finally transport and replay are confined to H and C, with P and N recording provenance and local naming. Since the carrier has no coordinate for a quotient completion, selected representative, host function equality, ambient completeness theorem, or choice schedule, every accepted extension consumer is forced through the displayed finite route. □

Theorem 3475.3 (Dense isometry extension completion handoff)**.** *Let $E = (D, J, U, S, F, W, T, R, H, C, P, N)$ be an accepted* DenseIsometryExtension$^{\uparrow}$ *carrier. Any completion-facing consumer of the extended isometry first reads the dense-source observations through D, W, T, R, then reads the completion-entry row J, then reads the universal-property and separatedness rows U, S, and only then receives transport, replay, provenance, and naming through H, C, P, N.*

The handoff is therefore a finite completion-extension service. It exports no quotient completion equality, selected host limit, host function equality, countable-choice schedule, ambient Real$^{\uparrow}$ *completeness theorem, or bridge-checking credential.*

Proof. Begin with the source side. The dense-source row D is inspected only through the finite windows W, dyadic tolerances T, and regular-rational handoff R, so a completion-facing request cannot skip directly to a completed target.

The completion-entry row J is the first completion-facing coordinate. After J has exposed the source observation as completion-entry data, U supplies the extension-and-uniqueness ledger and S supplies the separated classifier boundary.

The suffix H, C, P, N does not add mathematical content beyond transport, replay, provenance, and local naming of the displayed rows. Therefore the completion handoff is exactly the stated finite route, with no coordinate from which a quotient completion, selected limit, host function, choice schedule, ambient completeness theorem, or bridge credential could be read. □

Falsifiable prediction: Every accepted $\mathsf{DenseIsometryExtension}^{\uparrow}$ consumer reads the dense-source row, completion-entry row, universal-property row, separated comparison row, source isometry row, finite-window schedule, dyadic tolerance ledger, regular-rational handoff, transport, replay, provenance, and local naming before any completion-facing extension service is exposed.

Independence witness: completionembedding, completionuniversalproperty, separatedmetric, regseqrat, streamname, dyadicratcore, real: none is bijective to $\mathsf{DenseIsometryExtension}^{\uparrow}$ because the packet jointly carries dense-source admission, completion entry, universal-property extension, separated uniqueness, isometry preservation, finite windows, tolerance accounting, regular-rational readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseIsometryExtension}^{\uparrow}$ packet is generated from dense-source observations, a CompletionEmbedding entry row, a CompletionUniversalProperty extension-and-uniqueness row, a SeparatedMetric comparison row, source isometry preservation, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, Real seal access, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3475.1, Definition 3475.2, and Definition 3475.3 seed the finite dense-isometry extension route through dense source, completion entry, universal property, separated comparison, source isometry, finite observation, tolerance, readback, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseIsometryExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected limit extraction, host function equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exposure, completion-entry handoff, universal-property extension, separated uniqueness, source isometry preservation, finite-window and tolerance accounting, regular-rational readback, and local naming for BEDC.Derived.DenseIsometryExtensionUp.

Chapter 3476

A Concrete Naming Certificate for $\mathsf{IsometricCompletionExtension}^{\uparrow}$

Following Bourbaki and Kelley on metric completion, the $\mathsf{IsometricCompletionExtension}^{\uparrow}$ packet records the finite dense-isometry extension route that begins with the completion certificate of **??**, passes through the dense-source extension surface of chapter 3475, and reads uniqueness only through the complete-metric comparison boundary of chapter 5983. The packet names the universal extension theorem as finite BEDC completion data; it does not construct quotient completions, select completion points, invoke countable choice, or import an ambient complete metric space.

Definition 3476.1 (Isometric completion extension carrier)**.** *An* $\mathsf{IsometricCompletionExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (M, D, K, G, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric carrier row, read through the same public metric surface used by* $\mathsf{IsometricCompletion}^{\uparrow}$*.*
2. *D is the dense isometry row, recording the displayed dense-source isometry and its finite approximation witnesses through* $\mathsf{DenseIsometryExtension}^{\uparrow}$*.*
3. *K is the completion target row, exposing the completion handoff named by* $\mathsf{IsometricCompletion}^{\uparrow}$ *before any extension value is public.*
4. *G is the unique extension graph row, recording source agreement, completion-facing extension reads, and uniqueness only as finite graph data.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *tolerance readback row used for distance preservation, dense approximation, and zero-distance comparison.*
6. *H records componentwise* hsame *transport for M, D, K, G, R and the local naming row.*
7. *C records displayed* Cont *replay of metric admission, dense-isometry handoff, completion handoff, extension-graph readback, and Real tolerance sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{IsometricCompletionExtension}^{\uparrow}$*,* $\mathsf{IsometricCompletion}^{\uparrow}$*,* $\mathsf{DenseIsometryExtension}^{\uparrow}$*,* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{IsometricCompletionExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a selected completion point, quotient metric equality, countable choice, host function equality, or an ambient complete metric space.

Lean4 (statement) `BEDC.Derived.IsometricCompletionExtensionUp`

Theorem 3476.2 (Isometric completion extension NameCert obligations)**.** *For every accepted* $\mathsf{IsometricCompletionExtension}^{\uparrow}$ *carrier* $X = (M, D, K, G, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IsometricCompletionExtension}^{\uparrow}}$ *surface consists exactly of five obligation rows before structural replay is consumed:*

1. *source metric carrier exposure through* M*;*
2. *dense isometry exposure through* D*;*
3. *completion target handoff through* K*;*
4. *unique extension graph exposure through* G*;*
5. $\mathsf{Real}^{\uparrow}$ *tolerance readback through* R*.*

The remaining rows H, C, P, N *transport, replay, source, and locally name that five-row surface, and no accepted carrier exports selected completion points, quotient metric equality, countable choice, host function equality, or ambient completeness.*

Proof. Project the carrier of Definition 3476.1. The rows M, D, K, G, R are displayed before a completion-facing extension value can be read.

The source side is M, D. The metric carrier fixes the distance surface, while the dense isometry row supplies only finite dense-source observations and their Real tolerance readback.

The completion side is K, G, R. The completion target row is read before the graph row, and uniqueness of graph reads is routed through the same Real tolerance row and the complete-metric limit uniqueness boundary.

Finally H, C, P, N preserve the displayed route under transport, continuation replay, provenance, and local naming. Since the carrier contains no further coordinate, the excluded completion point, quotient, choice, host equality, and ambient completeness services cannot be exported. □

Theorem 3476.3 (Isometric completion extension unique factorization)**.** *Every completion-facing extension read from an accepted* $\mathsf{IsometricCompletionExtension}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DenseIsometryExtension}^{\uparrow},\ \mathsf{IsometricCompletion}^{\uparrow},\ \mathsf{CompleteMetricLimitUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

In carrier notation, a consumer must first expose the dense isometry row D*, the completion handoff row* K*, the complete-metric uniqueness comparison attached to* G*, and the Real tolerance row* R *before it reads any extension value. The factorization exports no selected completion point, quotient metric equality, countable-choice schedule, host function equality, or ambient complete metric space.*

Proof. Start with an accepted packet $X = (M, D, K, G, R, H, C, P, N)$. The dense-source portion is read through D over the metric source M, so the consumer begins with the finite dense-isometry surface rather than a host map on a completion.

Next read the completion handoff K. This is the only coordinate that names a completion target, and it is supplied by the finite $\mathsf{IsometricCompletion}^{\uparrow}$ route.

The graph row G is then interpreted through complete-metric limit uniqueness and the terminal Real tolerance row R. This gives the unique factorization route for extension reads while keeping equality-like comparison inside displayed zero-distance and Real readback data.

The structural suffix H, C, P, N transports, replays, sources, and names the same route. It adds no selected point, quotient equality, choice schedule, host function equality, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{IsometricCompletionExtension}^{\uparrow}$ extension read exposes source metric, dense isometry, completion handoff, unique extension graph, Real tolerance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any completion-facing value is public.

Independence witness: isometriccompletion, denseisometryextension, completemetriclimituniqueness, real: none is bijective to $\mathsf{IsometricCompletionExtension}^{\uparrow}$ because the packet jointly carries the source metric row, dense isometry row, completion handoff, unique extension graph, Real tolerance readback, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{IsometricCompletionExtension}^{\uparrow}}$.

Constructive story: An $\mathsf{IsometricCompletionExtension}^{\uparrow}$ packet is generated from a source MetricUp row, a DenseIsometryExtensionUp row, an IsometricCompletionUp completion target row, a CompleteMetricLimitUniquenessUp uniqueness comparison, a RealUp tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IsometricCompletionExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3476.1, Definition 3476.2, and Definition 3476.3 seed the finite dense-isometry completion-extension route over metric, completion, uniqueness, and Real tolerance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IsometricCompletionExtension}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.IsometricCompletionExtensionUp`

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected completion points, quotient metric equality, countable choice, host function equality, ambient completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-isometry exposure, completion handoff, unique extension graph exactness, Real tolerance preservation, transport and replay stability, provenance non-escape, and field faithfulness for BEDC.Derived.IsometricCompletionExtensionUp.

Chapter 3477

A Concrete Naming Certificate for $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$

Following Bishop and Bridges' constructive completion theory and Mac Lane's finite-product interface, the $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ packet records the finite BEDC surface at which completion reads commute with binary product reads. It consumes the Cauchy completion monad route of chapter 378, the metric-completion surface of chapter 1506, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the product naming envelope of chapter 34. The packet names a finite product-preservation handoff; it does not assert quotient completion equality, selected limits, countable choice, or an ambient completed product.

Definition 3477.1 (Cauchy completion product-preservation carrier)**.** *A* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{CPP}(L, R, M, \Pi, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the left source-completion route, read through the Cauchy completion monad and the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *source.*
2. *R is the right source-completion route, read through the same completion discipline for the second source.*
3. *M is the product metric row, recording the finite product carrier, projection rows, and coordinatewise tolerance ledger.*
4. *Π is the product projection row, exposing the two completed-coordinate projections and their recombination read.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the paired finite windows and coordinatewise regular rational data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the coordinate tolerances and the product distance read.*
7. *H stores componentwise* hsame *transport for the two completion routes, product row, projection row, readback, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a product-completion request through L, R, M, Π, S, E.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $L, R, M, \Pi, S, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable-choice schedules, ambient product completion, or host equality of completed points.

```
Lean4 ✓  BEDC.Derived.CauchyCompletionProductPreservationUp.CauchyCompletionProductPreservationTasteGate
single carrier alignment
```

Theorem 3477.2 (Cauchy completion product-preservation NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ *carrier* $\mathsf{CPP}(L, R, M, \Pi, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *surface consists exactly of carrier admission for the two source completion routes, product metric row, projection row, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows; coordinate compatibility through L, R, M, Π; readback compatibility through S, E; and non-escape from quotient completions, selected limits, countable choice, ambient completed products, and host equality.*

Proof. Carrier admission is projection from Definition 3477.1. The displayed tuple exposes exactly the two completion routes, product row, projection row, readback row, Real seal, and structural rows.

Coordinate compatibility is forced by L, R, M, Π. The left and right completion routes are read before the product projection row, and the product row records only the coordinatewise product carrier and finite tolerance ledger.

Readback compatibility is then carried by S and E. The regular rational row reads paired finite windows, and the terminal Real seal presents only the coordinate tolerances and the product distance read produced from those windows.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since the carrier has no coordinate for quotient completion equality, selected representatives, countable choice, ambient product completion, or host equality, the listed obligations exhaust the local certificate surface. □

Theorem 3477.3 (Cauchy completion finite-product handoff)**.** *Every downstream consumer that reads completion after a finite product, or a finite product after two completion reads, must factor through the same displayed route*

$$\mathsf{CauchyCompletionMonad}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow} \longrightarrow \mathsf{Prod}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The exported object is the finite product-preservation handoff carried by $L, R, M, \Pi, S, E, H, C, P, N$, not a quotient-level isomorphism of completions and not a selected-limit construction.

Proof. Start with the accepted carrier of Definition 3477.1. The only completion-facing rows are L and R, and both are routed through the Cauchy completion monad and the metric-completion source surface.

The product read is M, Π. These rows expose the finite product carrier, coordinate projections, recombination read, and coordinatewise tolerance ledger without adding an ambient completed product.

The numerical handoff is S, E. Paired finite windows are read through $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal records the coordinate tolerances and product distance read.

Apply Definition 3477.2. The transport, replay, provenance, and local naming rows preserve exactly that route, so every finite-product completion consumer factors through the displayed handoff and cannot export quotient equality or selected limits. □

Theorem 3477.4 (Cauchy completion product-preservation scoped ledger)**.** *Every accepted* CauchyCompletionProd *packet factors its visible product-preservation read through exactly the following scoped ledger:*

1. *the two completion routes L and R, each read through* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$*;*
2. *the product metric row M and projection row Π, including the displayed finite product carrier, coordinate projections, recombination read, and coordinatewise tolerance ledger;*
3. *the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row S and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E;*
4. *componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$ *row N.*

No accepted packet exports quotient completion equality, selected-limit data, countable-choice schedules, ambient completed products, or host equality of completed points outside this ledger.

Proof. Start from Definition 3477.3. The handoff route already forces every completion-product read through the displayed completion, metric, product, regular-rational, and Real rows.

The carrier of Definition 3477.1 names the ledger entries. The rows L and R are the two source completion routes, while M and Π are the only product metric and projection rows.

The numerical readback is confined to S and E. These rows pair the regular rational windows and then seal the coordinate tolerances and product distance read at the $\mathsf{Real}^{\uparrow}$ boundary.

Finally H, C, P, N preserve the same ledger under transport, replay, provenance, and local naming. Since no other carrier coordinate exists, the scoped ledger is exact and the excluded quotient, selection, choice, ambient-product, and host-equality data cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionProductPreservation}^{\uparrow}$ packet is generated from two CauchyCompletionMonad and MetricCompletion source routes, a product metric row, projection rows, RegSeqRat paired-window readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$, seed)

Scope closed: Definition 3477.1, Definition 3477.2, Definition 3477.3, and Definition 3477.4 seed the finite completion-product handoff over CauchyCompletionMonadUp, MetricCompletionUp, ProdUp, RegSeqRatUp, RealUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionProductPreservation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, ambient completed products, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coordinate compatibility, paired RegSeqRat readback, Real seal exactness, projection non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionPro

Chapter 3478

A Concrete Naming Certificate for $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$

The $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ packet realizes the finite support-closure algebra of Definition 9462.1609 and Definition 9462.1618 as BEDC NameCert data. It consumes the boundary-tile row of chapter 3393, the coverage-rank row of chapter 3470, the finite observation interface of chapter 1007, the observer support lattice of chapter 1122, and the finite residue boundary of chapter 1167. The packet names closure of visible trigger supports under finite windows; it is not a hidden global semantics, stochastic evolution model, quotient codon space, or infinite coverage oracle.

Definition 3478.1 (Window coverage-trigger algebra carrier)**.** *A* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, C, F, L, R, A, H, Q, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *trigger-boundary row exposing the finite window classifiers whose supports are read.*
2. *C is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *coverage row, recording which visible triggers cover which displayed support faces.*
3. *F is the* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *row that bounds every trigger read by a selected finite window and an omitted-tail ledger.*
4. *L is the* $\mathsf{ObserverSupportLattice}^{\uparrow}$ *support-lattice row, used only as a finite inclusion and coverage-rank ledger.*
5. *R is the* $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$ *residue row, recording which support faces remain outside the current closure.*
6. *A is the trigger-algebra row, carrying the finite support set, closure set, blocker row, coverage threshold row, and handoff to the $(4, 2)$ codon instance when that instance is requested.*
7. *H records componentwise* hsame *transport for the trigger-boundary, coverage, finite-window, support-lattice, residue, algebra, replay, provenance, and naming rows.*
8. *Q records displayed* Cont *replay from a coverage-trigger request through T, C, F, L, R, A.*

9. *P is the* Pkg *provenance over* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{FiniteObservationInterface}^{\uparrow}$, $\mathsf{ObserverSupportLattice}^{\uparrow}$, FiniteObservationResidueBounda *the source vision of Definition 9462.1609,* BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, C, F, L, R, A, H, Q, P, N$. It has no coordinate for hidden biological semantics, stochastic transition laws, quotient codon equality, global support completion, infinite search over triggers, or a final coverage oracle.

Theorem 3478.2 (Window coverage-trigger algebra NameCert obligations)**.** *For every accepted* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *carrier $W = (T, C, F, L, R, A, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlg}}$ *surface consists exactly of boundary-trigger admission through T, coverage-rank admission through C, finite-window exposure through F, support-lattice inclusion through L, residue-boundary accounting through R, algebra-row exactness through A, componentwise transport through H, displayed replay through Q, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.WindowCoverageTriggerAlgebraUp.WindowCoverageTriggerAlgebraTasteGate`
`single carrier alignment`

Proof. Carrier admission is projection from Definition 3478.1. The accepted packet exposes only the boundary trigger, coverage, finite-window, support-lattice, residue, trigger-algebra, transport, replay, provenance, and naming rows.

The trigger read starts with T, C, F. A consumer first sees the displayed boundary-tile classifiers, then the coverage-rank row, and then the finite observation interface that records the selected window and omitted-tail ledger.

The support and residue stages are L, R. The support lattice provides finite inclusion and coverage-rank comparison, while the residue row records which faces remain outside the displayed closure.

The algebra row A may then package the support set, closure set, blocker row, threshold row, and $(4, 2)$ handoff. The structural rows H, Q, P, N preserve only this displayed route, so hidden semantics, stochastic laws, quotient codon equality, global support completion, infinite trigger search, and final coverage oracles have no carrier coordinate. □

Theorem 3478.3 (Window coverage-trigger vision handoff)**.** *Every concrete read of the universal coverage-trigger system through* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *factors through the finite route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longrightarrow \mathsf{SixBoundaryTileCoverage}^{\uparrow} \longrightarrow \mathsf{FiniteObservationInterface}^{\uparrow} \longrightarrow \mathsf{ObserverSupportLa}$$

The exported object is the finite support-closure trigger algebra carried by $T, C, F, L, R, A, H, Q, P, N$, with the source vision used only as provenance for the displayed closure, blocker, threshold, and codon-instance rows.

Proof. Start with the coverage-trigger definitions in Definition 9462.1609. The concrete packet does not import the whole vision as an oracle; it reads only the finite trigger-boundary and coverage rows T, C.

Next bind finite access through F. The selected window and omitted-tail ledger determine which trigger supports are visible before any closure row can be exported.

Then read L and R. The support lattice records finite inclusion and coverage rank, while the residue boundary records the uncovered faces that remain outside the current closure.

Finally package the closure, blocker, threshold, and codon-instance handoff through A, H, Q, P, N. Since these rows preserve the displayed finite route, every concrete read of the universal coverage-trigger algebra factors through the NameCert packet rather than through hidden semantics, quotient codon equality, stochastic dynamics, or infinite trigger search. □

Theorem 3478.4 (Window coverage-trigger algebra Markov source route)**.** *Every Markov-trigger-facing consumer of* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *must read the boundary-trigger row of chapter 3393, the coverage-rank row of chapter 3470, the finite-observation window row of chapter 1007, the support-lattice row of chapter 1122, and the residue-boundary row of chapter 1167 before any Markov state transition row can be formed.*

Thus the rank-sensitive transition surface of Definition 9462.1541 is reached only after the finite support-closure route $T, C, F, L, R, A, H, Q, P, N$ *has exposed boundary triggers, coverage rank, finite windows, support inclusion, and residue accounting. The route exports no stochastic evolution law, hidden biological semantics, quotient codon equality, global support completion, or infinite trigger search.*

Proof. Start with a consumer that requests a Markov-trigger-facing transition row. The carrier of Definition 3478.1 exposes no transition coordinate before the boundary-trigger, coverage, finite-window, support-lattice, and residue rows have been read.

The source route is therefore T, C, F, L, R. These rows are supplied by the five sibling concrete chapters named in the statement and fix the displayed boundary triggers, rank data, finite observation window, support inclusion, and residue boundary before any transition-facing handoff is admissible.

Only after that route is displayed may A, H, Q, P, N package the trigger algebra for the rank-sensitive surface of Definition 9462.1541. Since the carrier contains no stochastic evolution row, hidden biological semantics, quotient codon equality, global support completion, or infinite trigger-search coordinate, none of those objects can be introduced by the Markov source route. □

Falsifiable prediction: Every accepted $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ consumer that claims coverage-trigger closure must exhibit the boundary-trigger row, coverage-rank row, finite-observation row, support-lattice row, residue-boundary row, algebra row, componentwise transport, replay, provenance, and local naming row. A consumer omitting any of probe-bundle, finite-observation, support-lattice, namecert, or trigger-boundary sibling slugs refutes the packet.

Independence witness: probe-bundle, finite-observation, support-lattice, namecert, trigger-boundary, six-boundary-tile-coverage: $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boundary triggers, coverage rank, finite-window access, support closure, residue accounting, algebraic blocker and threshold rows, transport, replay, provenance, and local naming.

The later Lean route for `BEDC.Derived.WindowCoverageTriggerAlgebraUp` must provide explicit $\mathsf{Nontrivial}\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ instances preserving $T, C, F, L, R, A, H, Q, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ packet is generated from boundary-trigger rows, coverage-rank rows, finite observation windows, support-lattice inclusion rows, residue-boundary rows, a trigger-algebra row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3478.1, Definition 3478.2, and Definition 3478.3 seed the finite support-closure trigger algebra over boundary tiles, coverage rows, finite observation windows, support lattices, residue boundaries, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{WindowCoverageTriggerAlgebra}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close hidden biological semantics, stochastic transition laws, quotient codon equality, global support completion, infinite trigger search, final coverage oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-trigger exactness, coverage-rank exactness, finite-window non-escape, support-lattice inclusion, residue accounting, Nontrivial WindowCoverageTriggerAlgebraUp, and FieldFaithful WindowCoverageTriggerAlgebraUp for BEDC.Derived.WindowCoverageTriggerAlgebraUp.

Chapter 3479

A Concrete Naming Certificate for $\mathsf{BorelFunction}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on Borel maps between topological and real-facing spaces, the $\mathsf{BorelFunction}^{\uparrow}$ packet records the finite pullback route by which a displayed source topology reads target Borel rows. It sits between the generated-set boundary of chapter 3384, the measurable-map preimage surface of chapter 3461, the terminal real-name seal of chapter 38, and the topology interface of chapter 88. The packet names Borel pullback stability as a local BEDC certificate, not as a host inverse-image operation, quotient equality of functions, countable-choice enumeration of Borel codes, or ambient measure completion.

Definition 3479.1 (BorelFunction carrier)**.** *A* $\mathsf{BorelFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, B, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the source topology row read through* $\mathsf{Topology}^{\uparrow}$*, including the finite open-neighbourhood data needed by the current source-side query.*
2. *B is the target* $\mathsf{BorelSet}^{\uparrow}$ *row whose generated-set boundary supplies the target event being pulled back.*
3. *R is the pullback ledger assigning the displayed target Borel row in B to a displayed source Borel row over X.*
4. *Q is the* $\mathsf{Real}^{\uparrow}$ *readback row used when the target Borel query is real-facing; it is consumed only through the displayed Borel row and the pullback ledger.*
5. *H stores componentwise* hsame *transport for the source topology, target Borel row, pullback ledger, real readback, continuation, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a source query through the target Borel row, the pullback ledger, and the real-facing readback row.*
7. *P is the* Pkg *provenance over* $\mathsf{Topology}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, R, Q, H, C, P, N. It has no coordinate for host set preimages, quotient equality of functions, countable-choice selection of Borel codes, completed measure spaces, almost-everywhere equality, or an ambient function space.

Lean4 ✓ `BEDC.Derived.BorelFunctionUp.BorelFunctionTasteGate single carrier alignment`

Theorem 3479.2 (BorelFunction NameCert obligations)**.** *For the carrier of Definition 3479.1, the local* $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$ *surface consists exactly of source-topology admission, target Borel-row admission, pullback-ledger exactness, real-facing readback through* $\mathsf{Real}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every accepted Borel-function consumer must read the source row X, the target Borel row B, and the pullback ledger R before any* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{MeasurableFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or topology-facing sibling row is exposed.*

Proof. Project the carrier $F = (X, B, R, Q, H, C, P, N)$. The rows X and B are the only source-topology and target-Borel coordinates, and the row R is the only coordinate that relates them.

Read a real-facing target query through B and Q. The readback row Q is downstream of the displayed Borel row, while the source-side result is named only by the pullback ledger R over X.

The structural rows add no host inverse-image operation. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route, so the excluded quotient, choice, completion, almost-everywhere, and ambient function-space objects remain outside the packet. □

Theorem 3479.3 (BorelFunction pullback stability)**.** *Let $F = (X, B, R, Q, H, C, P, N)$ and*

$$F' = (X', B', R', Q', H', C', P', N')$$

be accepted $\mathsf{BorelFunction}^{\uparrow}$ *carriers. If the source topology rows, target* $\mathsf{BorelSet}^{\uparrow}$ *rows, pullback ledgers, and real-facing readback rows agree componentwise through the displayed* hsame *transports, then every accepted source-side Borel read produced from F is transported to the corresponding read produced from F'. The comparison uses only*

$$X, B, R, Q, H, C, P, N$$

and

$$X', B', R', Q', H', C', P', N'.$$

It exports pullback stability as a finite certificate, not as host set preimage equality, quotient equality of functions, countable-choice enumeration, completed measure semantics, almost-everywhere equality, or ambient function equality.

Proof. Apply Definition 3479.2 to both carriers. Any accepted target Borel read first passes through the displayed target rows B and B', then through the pullback ledgers R and R'.

Use the componentwise transports. The hypotheses identify the source topology rows, target Borel rows, pullback ledgers, and real readback rows, while H, C, P, N and H', C', P', N' preserve, replay, source, and name those same routes.

The transported consumer sees only the displayed coordinates. Since neither carrier contains a host inverse-image coordinate, quotient-function equality, choice-selected code sequence, completed measure, almost-everywhere class, or ambient function object, the stability claim is exactly the preservation of the finite pullback ledger. □

Falsifiable prediction: A $\mathsf{BorelFunction}^{\uparrow}$ consumer that obtains a host set preimage, a quotient equality of functions, a countable-choice enumeration of Borel codes, a completed measure, or an almost-everywhere representative without first reading the displayed pullback ledger R refutes the carrier surface.

Independence witness: borelset, measurablefunction, real, topology: no listed sibling supplies the whole eight-row packet $F = (X, B, R, Q, H, C, P, N)$, because each sibling contributes only generated-set, measurable-map, real-readback, or source-topology data while this chapter binds the Borel pullback ledger with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3479.1, Definition 3479.2, and Definition 3479.3 seed the finite Borel-function pullback boundary over source topology, target Borel rows, real readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, almost-everywhere equality, quotient function spaces, countable-choice Borel codes, host set preimages, ambient function equality, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, pullback-ledger exactness, real-readback preservation, componentwise transport, and ledger-refusal proofs for BEDC.Derived.BorelFunctionUp.

Constructive story: A $\mathsf{BorelFunction}^{\uparrow}$ packet is generated from a source topology row, a target BorelSet row, a pullback ledger, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3480

A Concrete Naming Certificate for $\mathsf{ChebyshevPolynomial}^{\uparrow}$

Following Chebyshev and the classical approximation-theory presentation of Chebyshev polynomials, the $\mathsf{ChebyshevPolynomial}^{\uparrow}$ packet names the finite recurrence surface for the displayed family T_n without importing an ambient real-function algebra. It sits after the terminal real-name seal of chapter 38, the polynomial carrier of chapter 50, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the stream-window surface of chapter 126. The packet supplies a traditional approximation-theory object as finite NameCert data for later minimax and constructive approximation routes.

Definition 3480.1 (Chebyshev polynomial carrier)**.** *A* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(d, T_0, T_1, R, W, Q, H, C, P, N).$$

Here d is the unary degree row, T_0 and T_1 are the displayed recurrence seeds, R is the recurrence-step ledger for $T_{n+1} = 2xT_n - T_{n-1}$, W is the dyadic evaluation-window schedule, Q is the $\mathsf{Real}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *readback row, H records componentwise* hsame *transport, C records* Cont *replay of the recurrence route, P records* Pkg *provenance from the polynomial and real-name sources, and N is the local* $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$ *row. A consumer of the packet may inspect a Chebyshev read only through these displayed rows.*

Lean4 ✓ `BEDC.Derived.ChebyshevPolynomialUp.ChebyshevPolynomialTasteGate single carrier alignment`

Theorem 3480.2 (Chebyshev polynomial NameCert obligations)**.** *For every accepted* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *carrier $(d, T_0, T_1, R, W, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 3480.1. The degree row chooses the recurrence depth, the seed rows display T_0 and T_1, the recurrence ledger displays each finite step, the dyadic window and regular readback rows mediate evaluation, and the transport, replay, provenance, and local naming rows export the resulting polynomial certificate. The packet imports no host polynomial equality, selected infinite family, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet rows in order. The degree row d bounds the visible recurrence segment, so the seed rows and recurrence ledger already determine every displayed polynomial row consumed by the packet.

The evaluation side factors through W and Q. The dyadic windows give the finite tolerance requests, while the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal keep coefficient and value reads inside the existing real-name route.

Finally use H, C, P, N. These rows transport equal displayed histories, replay the recurrence computation, remember the polynomial and real-name provenance, and attach the local certificate name. No additional host equality or infinite polynomial family is available to a consumer outside the listed rows. □

Theorem 3480.3 (Chebyshev recurrence handoff)**.** *Every accepted* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *recurrence read factors through*

$$d \longmapsto T_0, T_1 \longmapsto R \longmapsto W \longmapsto Q \longmapsto H, C, P, N$$

before any approximation-facing consumer receives a polynomial value or coefficient row. Thus the Chebyshev recurrence is handed to later constructive approximation packets as finite degree, seed, recurrence, dyadic-window, real-readback, transport, replay, provenance, and naming data, not as an external analytic theorem about an infinite sequence of polynomials.

Proof. Start with the carrier of Definition 3480.1. The only recurrence coordinates exposed by the packet are d, T_0, T_1, R, so any accepted recurrence read is bounded by the displayed unary degree and the finite seed-and-step ledger.

Next read the evaluation interface. The route through W and Q forces every value-facing request to pass through dyadic windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal before it can be consumed as approximation data.

The remaining rows provide structural export. Componentwise hsame transport, Cont replay, Pkg provenance, and the local NameCert row preserve the same finite recurrence route for downstream minimax, interpolation, and approximation packets. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChebyshevPolynomial}^{\uparrow}$ packet is generated from a unary degree row, recurrence seed rows, a finite recurrence-step ledger, dyadic evaluation windows, regular rational and Real readback, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$, obligation)

Scope closed: Definition 3480.1, Definition 3480.2, and Definition 3480.3 declare the finite recurrence certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChebyshevPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, minimax optimality, Markov or Bernstein inequalities, Jackson approximation, host polynomial equality, selected infinite families, or ambient RealUp completeness.

Upgrade path: Checked closure requires Lean carrier admission, recurrence ledger stability, evaluation-window readback, and recurrence handoff for the same finite packet.

Chapter 3481

A Concrete Naming Certificate for $\mathsf{CellularWindowTrace}^{\uparrow}$

The $\mathsf{CellularWindowTrace}^{\uparrow}$ packet realizes the cellular-carrier alternative of chapter 9391 as finite orbit-window NameCert rows while leaving the BEDC kernel carrier as BHist. It consumes the cellular automaton and substrate readings of chapter 934 and chapter 942, the finite BHist encoding discipline of chapter 943, the stream-window surface of chapter 126, and the bounded universal trace boundary of chapter 944. The packet names a finite window of a cellular orbit, not a replacement foundation for BEDC histories.

Definition 3481.1 (Cellular window trace carrier)**.** *A* $\mathsf{CellularWindowTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(w, r, i, o, B, H, C, P, N).$$

Here w is the displayed width row, r is the local-rule row, i is the initial cell-row encoding, o is the finite orbit-window ledger, B is the BHist *encoding of the selected rows, H records componentwise* hsame *transport, C records* Cont *replay of the local update route, P records* Pkg *provenance from the cellular-substrate sources, and N is the local* $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$ *row.*

Theorem 3481.2 (Cellular window trace NameCert obligations)**.** *For every accepted* $\mathsf{CellularWindowTrace}^{\uparrow}$ *carrier $(w, r, i, o, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$ *surface consists exactly of width, local-rule, initial-row, finite orbit-window,* BHist *encoding, transport, replay, provenance, and local naming rows. A consumer may read the cellular window only by first consuming these rows; it receives no global orbit, no infinite cellular history, no substrate replacement of* BHist*, and no bijection between cellular traces and all BEDC histories.*

Lean4 ✓ `BEDC.Derived.CellularWindowTraceUp.CellularWindowTraceTasteGate single carrier alignment`

Proof. Project the displayed carrier rows. The width and local-rule rows determine which finite cell row can be updated, and the initial row gives the only starting configuration available to the packet.

The orbit-window ledger o records the finite segment that is actually visible. Its BHist encoding row B translates that segment into the BEDC kernel carrier without changing the carrier used by the surrounding certificate.

Finally H, C, P, N export the route. Transport preserves the displayed rows, replay records the local update computation, provenance ties the route to the cellular-substrate sources, and the local certificate names this finite window only. □

Theorem 3481.3 (Cellular window trace finite-read factorization)**.** *Every accepted read of a* $\mathsf{CellularWindowTrace}^{\uparrow}$ *packet factors through*

$$w \longmapsto r \longmapsto i \longmapsto o \longmapsto B \longmapsto H, C, P, N.$$

Thus the cellular-carrier alternative is exposed as finite orbit-window data before any downstream consumer can treat it as a cellular trace.

Proof. Read the packet in dependency order. Width and local rule are needed before the initial row can be updated, and the finite orbit window is the only displayed result of those updates.

The BHist encoding row then returns the trace to the BEDC carrier. Since transport, replay, provenance, and naming are all downstream of that encoding, the public consumer sees a finite cellular-window certificate rather than a direct cellular replacement for BHist. □

Falsifiable prediction: Every accepted cellular-window read of $\mathsf{CellularWindowTrace}^{\uparrow}$ factors through width, local-rule, initial-row, finite orbit-window, BHist encoding, hsame transport, Cont replay, Pkg provenance, and local naming rows before any downstream cellular-trace consumer can inspect it.

Independence witness: cellularsubstrate, cellularautomaton, streamname, bhist-arity16-eventflow, boundeduniversaltrace: the cellular-window packet is not bijective with the BEDC carrier. It records one finite orbit window with a displayed encoding, while BHist supplies the kernel carrier independently of any chosen cellular width, rule, initial row, or orbit segment.

On the Lean side, the formal carrier should supply ‘Nontrivial CellularWindowTraceUp‘ and ‘FieldFaithful CellularWindowTraceUp‘ instances in ‘lean4/BEDC/Derived/CellularWindowTraceUp/TasteGate.l

Closure status for $\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CellularWindowTrace}^{\uparrow}$ packet is generated from a width row, local-rule row, initial-row encoding, finite orbit-window ledger, BHist encoding, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$, obligation)

Scope closed: Definition 3481.1, Definition 3481.2, and Definition 3481.3 declare the finite orbit-window certificate surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CellularWindowTrace}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CellularWindowTraceUp.CellularWindowTraceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim a global cellular orbit, an infinite cellular history, a substrate replacement for BHist, a bijection between cellular traces and all BEDC histories, or bridge checking.

Upgrade path: Checked closure requires Lean carrier admission, finite orbit-window factorization, BHist encoding transport, and TasteGate instance rows for the same finite packet.

Chapter 3482

A Concrete Naming Certificate for $\mathsf{WeakKonigBoundary}^{\uparrow}$

Following Kleene and Bishop–Bridges on finite binary trees, Cantor-space paths, and constructive real completion, the $\mathsf{WeakKonigBoundary}^{\uparrow}$ packet records the boundary at which finite tree inspection remains available while weak König path selection is refused as a BEDC primitive. It sits after the finite obstruction surface of chapter 1622, the Cantor and FanTheorem interfaces of chapter 1475 and chapter 400, the finite window and regular readback route of chapter 126 and chapter 301, and the completion-facing seal rows of chapter 38 and chapter 1714. The packet names the finite-tree/path-selection boundary used by Real and Completion consumers without turning that boundary into countable choice, quotient stream equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3482.1 (WeakKonigBoundary carrier)**.** *A* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, A, F, S, Q, R, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{KleeneTree}^{\uparrow}$ *finite-prefix tree packet, including the decidable node ledger and obstruction row of Definition 1622.1.*
2. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix observation row offered to consumers that inspect binary path fragments.*
3. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar test row available only as displayed finite cover data, not as open-cover compactness.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which every path-like request must pass.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from displayed finite windows before any Real-facing seal is inspected.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *and completion-facing seal row reached only after K, A, F, S, Q have been displayed.*
7. *O is the boundary-refusal row: no infinite branch selector, no countable-choice functional, no quotient stream equality, no open-cover compactness principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of the packet.*

8. *H records componentwise* hsame *transport for the tree packet, Cantor prefix row, fan row, stream windows, regular readback, Real seal, refusal row, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite tree or path request through $K, A, F, S, Q, R, O, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{WeakKonigBoundary}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite tree/path boundary packet. It has no coordinate for an infinite branch, a path-selection functional, countable choice, quotient stream equality, open-cover compactness, selected completion limits, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.WeakKonigBoundaryUp.TasteGate.WeakKonigBoundaryTasteGate single carrier alignment`

Theorem 3482.2 (WeakKonigBoundary NameCert obligations)**.** *For every accepted* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier $W = (K, A, F, S, Q, R, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{KleeneTree}^{\uparrow}$ *packet,* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, refusal row, transport, replay, provenance, and local naming;*

2. *tree-window exactness, requiring every path-like read to pass through the finite tree row K and finite window row S before any Cantor, FanTheorem, RegSeqRat, Real, or Completion-facing row is exposed;*

3. *regular-readback discipline, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q to be obtained from displayed finite windows rather than from quotient stream equality;*

4. *seal nonescape, requiring the Real and completion-facing row R to remain downstream of K, A, F, S, Q and the refusal row O;*

5. *boundary refusal, excluding infinite branch selection, countable choice, quotient stream equality, open-cover compactness, and ambient Real completeness from the packet.*

No WeakKonigBoundary obligation is stored outside these rows.

Proof. Project the carrier of Definition 3482.1. The projection exposes exactly the Kleene tree packet K, Cantor prefix row A, FanTheorem finite-bar row F, stream windows S, regular readback Q, Real seal R, refusal row O, and structural rows H, C, P, N.

Tree-window exactness is inherited from the finite route displayed by Definition 1622.3. A path-like request first reads the finite tree and the finite stream window; it cannot jump directly to an infinite branch or completed Cantor point.

The regular-readback and seal rows are downstream. The row Q is admitted only after the finite windows are fixed, and the row R is reached only after K, A, F, S, Q and the refusal row have been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since none of the carrier rows stores an infinite branch selector, countable-choice functional, quotient stream equality, open-cover compactness object, or ambient Real-completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 3482.3 (WeakKonigBoundary Real-completion refusal)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{WeakKonigBoundary}^{\uparrow}$ *carrier factors through the finite route*

$$K, A, F, S, Q, R, O, H, C, P, N.$$

Consequently the consumer may reuse finite Kleene-tree prefixes, Cantor-space observations, Fan-Theorem finite-bar rows, StreamName windows, and RegSeqRat readbacks, but it receives no infinite branch selection, no countable-choice schedule, no quotient stream equality, no open-cover compactness principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with a Real or Completion-facing read of $W = (K, A, F, S, Q, R, O, H, C, P, N)$. Carrier admission gives the consumer the tree, Cantor, fan, stream-window, regular-readback, and seal rows only as the displayed finite packet.

Use the Kleene-tree boundary from Definition 1622.10. Any Real-facing interpretation of tree data must already pass through finite $\mathsf{StreamName}^{\uparrow}$ windows and the obstruction row before a real seal is inspected.

The $\mathsf{WeakKonigBoundary}^{\uparrow}$ carrier adds the refusal row O at the same boundary. Thus the completion-facing row R can be read only after the finite tree/path route and cannot supply a path selector, quotient stream equality, open-cover compactness, or ambient Real completeness.

Transport and replay through H and C, with provenance and naming through P and N, preserve the displayed route. Therefore every Real or Completion consumer receives the finite boundary package and none of the excluded principles. □

Falsifiable prediction: A $\mathsf{WeakKonigBoundary}^{\uparrow}$ consumer that obtains an infinite branch selector, countable-choice schedule, quotient stream equality, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem without first reading the displayed route $K, A, F, S, Q, R, O, H, C, P, N$ refutes the carrier surface.

Independence witness: kleene-tree, cantorspace, fantheorem, streamname, regseqrat, real, separable-completion: $\mathsf{WeakKonigBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite tree packet, Cantor prefix row, fan row, stream windows, regular readback, Real seal, refusal row, transport, replay, provenance, and local NameCert row in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3482.1, Definition 3482.2, and Definition 3482.3 seed the finite tree/path-selection boundary separating Real and Completion-facing reads from weak König path selection.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakKonigBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak König's lemma, infinite branch selection, countable choice, quotient stream equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree-window exactness, regular-readback discipline, seal nonescape, and boundary-refusal proofs for BEDC.Derived.WeakKonigBoundaryUp.

Constructive story: A $\mathsf{WeakKonigBoundary}^{\uparrow}$ packet is generated from a KleeneTreeUp finite tree packet, CantorSpace finite-prefix row, FanTheorem finite-bar row, StreamName finite windows, RegSeqRat readback, RealUp seal, explicit refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3483

A Concrete Naming Certificate for $\mathsf{CauchyCutBridge}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCutBridge}^{\uparrow}$ packet records the finite handoff by which a regular Cauchy real name supplies located-cut observations without identifying either presentation by quotient equality. It sits between the regular-rational route of chapter 301, the located-cut surface of chapter 404, the dyadic arithmetic ledger of chapter 291, the finite stream windows of chapter 126, and the terminal real-name seal of chapter 38. The packet is a bridge for displayed approximation data, not a Dedekind-completeness theorem or an equality principle for completed reals.

Definition 3483.1 (Cauchy-cut bridge carrier)**.** *A* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, L, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the regular Cauchy rational source.*
2. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *row recording the lower and upper located observations obtained from the same regular name.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare regular readbacks with located lower–upper brackets.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule listing the finite precision rows at which the regular name and located cut are jointly inspected.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after Q, L, D, W have been displayed.*
6. *H records componentwise* hsame *transport for the regular readback, located-cut row, dyadic ledger, stream windows, Real seal, replay, provenance, and local naming row.*
7. *C records displayed* Cont *replay of the bridge route*

$$Q, W, D \longmapsto L \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyCutBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient reals, Dedekind completeness, host equality of real names, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCutBridgeUp.TasteGate.CauchyCutBridgeTasteGate single carrier alignment`

Theorem 3483.2 (Cauchy-cut bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier* $B = (Q, L, D, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed regular readback, located-cut row, dyadic tolerance ledger, stream-window schedule, Real seal, componentwise transport, continuation replay, package provenance, and local naming row; regular-to-located exposure through* Q, W, D, L*; Real-seal exposure through* E*; and refusal of quotient reals, Dedekind-completeness claims, host equality, trichotomy, choice, and ambient completeness.*

Proof. Project the carrier of Definition 3483.1. The visible rows are exactly $Q, L, D, W, E, H, C, P, N$, so carrier admission exposes the regular readback, located-cut observation, dyadic ledger, stream schedule, terminal seal, and structural rows.

The bridge exposure is supplied by the replay row C. It first reads the regular name through Q and the finite windows W, then checks the matching dyadic tolerances in D, and only then exposes the located-cut row L.

The Real seal and structural rows are downstream of that route. Transport through H, provenance through P, and local naming through N record the same finite bridge, while no refused object appears as a carrier coordinate. □

Theorem 3483.3 (Cauchy-cut bridge regular-located handoff)**.** *Let* $B = (Q, L, D, W, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCutBridge}^{\uparrow}$ *carrier. Every located-cut observation exported by* B *factors through the finite handoff*

$$Q, W, D \longmapsto L \longmapsto E, H, C, P, N.$$

Thus the bridge reads the regular Cauchy name through $\mathsf{RegSeqRat}^{\uparrow}$*, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the* $\mathsf{LocatedCut}^{\uparrow}$ *data or the terminal* $\mathsf{Real}^{\uparrow}$ *seal can be consumed. It does not export quotient reals, Dedekind completeness, host equality, trichotomy, countable choice, or ambient completeness.*

Proof. Begin with a consumer of the located-cut row. The carrier definition gives L only as a displayed row of B, and the continuation row records that it is reached from Q, W, D.

Use the sibling routes cited in the chapter introduction. $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance comparison that turns the readback into lower–upper located observations.

Finally read the terminal seal row E and the structural rows H, C, P, N. They transport, replay, source, and name the same handoff; none of them stores quotient equality, a Dedekind-completeness witness, a host equality test, trichotomy, choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCutBridge}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{LocatedCut}^{\uparrow}$ observation row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3483.1, Definition 3483.2, and Definition 3483.3 seed the finite bridge from regular Cauchy readbacks to located-cut observations over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCutBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient reals, Dedekind-completeness claims, host equality of real names, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, regular-to-located exposure, tolerance exactness, Real-seal nonescape, and ledger-refusal proofs for the same finite packet.

Chapter 3484

A Concrete Naming Certificate for $\mathsf{BishopCompletionTheorem}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompletionTheorem}^{\uparrow}$ packet records the theorem-level completion handoff as finite BEDC certificate data. It sits after the explicit modulus surface of chapter 1438, the finite-window naming surface of chapter 126, the regular rational readback of chapter 301, the terminal Real surface of chapter 38, and the universal regular-Cauchy handoff of chapter 1425. The packet names the constructive theorem route without quotient limits, selected representatives, countable choice, or ambient completeness.

Definition 3484.1 (Bishop completion theorem carrier)**.** *A* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row recording the explicit tolerance-to-tail schedule.*
2. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window source row inspected by the theorem route.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the displayed windows and modulus schedule.*
4. E *is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after* M, S, R *have been displayed.*
5. U *is the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *handoff row for extension and uniqueness consumers.*
6. H *records componentwise* hsame *transport for the modulus, stream, readback, Real seal, universal handoff, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a theorem consumer through* M, S, R, E, U.
8. P *is the* Pkg *provenance over* $\mathsf{BishopCompletionTheorem}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$ *row.*

The carrier exports only this theorem route. It contains no quotient limit, selected representative, host convergence predicate, host real equality test, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BishopCompletionTheoremUp.BishopCompletionTheoremTasteGate single carrier alignment`

Theorem 3484.2 (Bishop completion theorem NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionTheore}$ *carrier* $B = (M, S, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission, explicit Cauchy-modulus exposure, finite StreamName source exposure, RegSeqRat readback, Real seal handoff, regular-Cauchy universal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every theorem consumer must factor through*

$$M \longmapsto S \longmapsto R \longmapsto E \longmapsto U \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3484.1. The accepted packet displays exactly $M, S, R, E, U, H, C, P, N$.

Read the analytic route in order. The explicit modulus row M licenses the finite StreamName windows S, the windows feed the regular rational readback R, and only then can the Real seal E be consumed.

The universal handoff U is downstream of the same displayed route. The structural rows H, C, P, N transport, replay, source, and name the route, while the refused quotient, selected, host, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 3484.3 (Bishop completion theorem regular Cauchy existence route)**.** *Every accepted* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *carrier* $B = (M, S, R, E, U, H, C, P, N)$ *exposes the regular Cauchy existence route as finite certificate composition:*

$$\mathsf{CauchyConvergenceModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$$

The route witnesses existence only as a displayed regular-Cauchy handoff through $M, S, R, E, U, H, C, P, N$*; it does not create a quotient limit, select a representative, invoke countable choice, compare host real values, or assert ambient completeness.*

Proof. Start with the modulus-controlled source. The row M supplies the displayed tolerance schedule, and S supplies the finite windows inspected at those requests.

Compose the readback and seal rows. The row R repacks the finite windows as regular rational data, and E records the terminal Real-facing seal reached by that regular route.

Finally consume the universal handoff U. By projection and finite route composition, any theorem consumer reaches the regular-Cauchy existence surface only after M, S, R, E and before H, C, P, N close transport, replay, provenance, and naming. The excluded quotient, selected, choice, host, and ambient-completeness objects are absent from the packet. □

Falsifiable prediction: A claimed $\mathsf{BishopCompletionTheorem}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ without the displayed CauchyConvergenceModulus, StreamName, and RegSeqRat rows refutes the carrier surface.

Independence witness: cauchyconvergencemodulus, streamname, regseqrat, real, regularcauchycompletionuniversal: no listed sibling is bijective to $\mathsf{BishopCompletionTheorem}^{\uparrow}$ because the theorem packet binds the modulus-controlled source, finite windows, regular readback, Real seal, universal handoff, transport, replay, provenance, and local NameCert rows in one theorem-level carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionTheorem}^{\uparrow}}$.

Constructive story: A BishopCompletionTheorem$^{\uparrow}$ packet is generated from a CauchyConvergenceModulus row, StreamName finite windows, RegSeqRat readback, RealUp seal, RegularCauchyCompletionUniversal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BishopCompletionTheorem}\uparrow}$, seed)

Scope closed: Definition 3484.1, Definition 3484.2, and Definition 3484.3 seed the finite theorem-level route from modulus-controlled Cauchy data through regular rational readback into Real and regular-completion handoff rows.

Formal verification: FormalStatus(NameCert$_{\text{BishopCompletionTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion, selected representatives, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, modulus exposure, finite StreamName windows, RegSeqRat readback, Real seal handoff, regular-Cauchy universal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionTheoremUp.

Chapter 3485

A Concrete Naming Certificate for $\mathsf{BaireUltrametric}^{\uparrow}$

Following Brouwer's spread semantics and the prefix-space practice of Troelstra–van Dalen and Bishop–Bridges, the $\mathsf{BaireUltrametric}^{\uparrow}$ packet names the common-prefix ultrametric surface on Baire streams. It sits between the finite prefix carrier of chapter 334, the prefix metric route of chapter 1602, the general ultrametric discipline of chapter 1535, the finite-window readback of chapter 126, and the completion-facing packet of chapter 1884. The chapter names the finite common-prefix distance interface; it is not host stream equality, quotient equality of streams, countable choice, a selected branch principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The finite prefix route consumes the $\mathsf{StreamName}^{\uparrow}$ window discipline of chapter 126 and the strong-triangle comparison surface of chapter 1535; the present packet only binds those sibling rows to the Baire prefix classifier.

Definition 3485.1 (Baire ultrametric carrier)**.** *A* $\mathsf{BaireUltrametric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, M, V, S, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row, naming the two inspected Baire prefixes and their common-prefix length.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *row, recording the dyadic distance value attached to the first displayed disagreement or to full agreement on the inspected window.*
3. *V is the* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison row, exposing the strong-triangle comparison only through finite common-prefix witnesses.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the stream coordinates read before the prefix distance is declared.*
5. *H records componentwise* hsame *transport for prefix rows, metric rows, ultrametric comparisons, finite windows, replay rows, provenance, and local naming.*
6. *C records displayed* Cont *replay of the same finite prefix windows and distance rows.*
7. *P is the* Pkg *provenance over* $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{UltrametricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, M, V, S, H, C, P, N. It has no coordinate for host stream equality, quotient stream equality, selected branches, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier`

Theorem 3485.2 (Baire ultrametric NameCert obligations)**.** *For every accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *carrier* $U = (B, M, V, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed Baire prefix row, BaireMetric distance row, UltrametricSpace comparison row, StreamName window row, transport, replay, provenance, and local naming;*
2. *common-prefix distance exactness, requiring each distance read to be the dyadic value named by the first displayed disagreement, or the zero-distance window when the inspected prefixes agree throughout the displayed window;*
3. *finite-cylinder classifier stability, requiring every cylinder comparison to transport only the displayed prefix rows through componentwise* hsame *and displayed* Cont *replay;*
4. *StreamName readback, requiring every coordinate read to pass through the finite window row S before the metric row M or ultrametric row V is consulted;*
5. *nonescape, excluding selected branches, quotient stream equality, countable choice, ambient Real completeness, host stream equality, Lean verification, and bridge checking.*

No BaireUltrametric obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 3485.1. The accepted packet displays exactly the Baire prefix row, the prefix metric row, the ultrametric comparison row, the finite StreamName window, and the structural rows.

Common-prefix distance exactness is read from B and M. The prefix row records the inspected coordinates and the first displayed disagreement if one exists; the metric row records the corresponding dyadic distance, with the zero-distance window available only for agreement across the displayed finite window.

Finite-cylinder classifier stability is supplied by H and C. Componentwise hsame transport identifies the displayed prefix rows, and displayed Cont replay rereads the same finite windows before any classifier output is compared.

StreamName readback is the row S. Since every coordinate request must pass through S, the ultrametric row V can only witness the strong-triangle comparison through finite common-prefix evidence already displayed by the packet.

The listed rows contain no selector for an infinite branch, no quotient equality of streams, no countable-choice operation, no host stream equality predicate, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. Thus the five rows exhaust the local naming surface. □

Theorem 3485.3 (Baire ultrametric prefix stability)**.** *Let* $U = (B, M, V, S, H, C, P, N)$ *and* $U' = (B', M', V', S', H', C', P', N')$ *be accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *carriers. If the displayed* $\mathsf{BaireSpace}^{\uparrow}$ *prefix rows,* $\mathsf{BaireMetric}^{\uparrow}$ *common-prefix distance rows,* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison rows, and* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows transport componentwise under* hsame *and replay through displayed* Cont *routes, then U and U' determine the same finite-cylinder classifier value and the same common-prefix ultrametric route.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier prefix stability`

Proof. Project both carriers through Definition 3485.1. The only analytic rows available for comparison are the prefix rows B, B', the metric rows M, M', the ultrametric rows V, V', and the window rows S, S'.

Use the assumed componentwise hsame transports to identify the displayed prefix and distance rows. The displayed Cont replays then force the same finite StreamName coordinates to be reread before either carrier can expose a classifier value.

Apply Definition 3485.2. Common-prefix distance exactness fixes the dyadic value from the first displayed disagreement or from the displayed agreement window, and finite-cylinder classifier stability keeps the classifier on those same rows.

The structural rows P, N and P', N' provide provenance and local naming only for the already transported route. Since no excluded branch, quotient, choice, host equality, or ambient Real-completeness coordinate is present, both carriers determine the same finite common-prefix ultrametric surface. □

Theorem 3485.4 (Baire ultrametric finite-cylinder separation)**.** *Let $U = (B, M, V, S, H, C, P, N)$ and $U' = (B', M', V', S', H', C', P', N')$ be accepted* BaireUltrametric$^{\uparrow}$ *packets. Suppose their displayed* BaireSpace$^{\uparrow}$ *prefix rows expose a first-disagreement window before length k, and suppose the* StreamName$^{\uparrow}$ *rows S, S' list every coordinate inspected up to that window. Then the* BaireMetric$^{\uparrow}$ *rows M, M' display a positive dyadic common-prefix distance for the two inspected names, and the packets determine disjoint finite cylinders before any* CompleteBaireSpace$^{\uparrow}$ *consumer reads a limit row.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier finite cylinder separation`

Proof. Project both carriers through Definition 3485.1. The only rows capable of naming the disagreement are the BaireSpace$^{\uparrow}$ prefix rows B, B', the BaireMetric$^{\uparrow}$ distance rows M, M', and the finite StreamName$^{\uparrow}$ windows S, S'.

Use the displayed first-disagreement window in B, B'. It identifies a least inspected coordinate at which the two finite prefixes differ, while S, S' certify that the coordinate is part of the finite window actually read by the packets.

The positive distance is then read from M, M'. By Definition 3485.2, common-prefix distance exactness assigns the dyadic value named by that first displayed disagreement, so the two packets cannot be assigned the zero-distance window for the inspected prefixes.

The finite cylinders are the two prefix cylinders ending at the displayed disagreement window. Any stream extending the first prefix must keep the first value at the disagreement coordinate, and any stream extending the second prefix must keep the second value there; the two finite-cylinder rows are therefore disjoint without consulting a completed limit.

Finally, V, V' are used only for the strong-triangle boundary. They preserve the finite-cylinder separation when a later consumer compares a third prefix, but they do not introduce an infinite branch, selected limit, quotient equality, or host stream equality. Thus a CompleteBaireSpace$^{\uparrow}$ consumer receives the positive dyadic separation before any completion-facing limit row is read. □

Theorem 3485.5 (Baire ultrametric cylinder refinement)**.** *Let $U = (B, M, V, S, H, C, P, N)$ and $U' = (B', M', V', S', H', C', P', N')$ be accepted* BaireUltrametric$^{\uparrow}$ *packets. If U displays a finite prefix cylinder and U' extends the same prefix through a longer* StreamName$^{\uparrow}$ *window without changing the already inspected coordinates, then the later cylinder refines the earlier one and the common-prefix ultrametric distance cannot increase on the displayed window. The refinement route*

uses only finite prefix rows, StreamName readback, componentwise hsame *comparison, and the finite strong-triangle boundary; it exports no selected branch, quotient stream equality, countable choice, host stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier cylinder refinement`

Proof. Project both packets through Definition 3485.1. The rows B, B' name the displayed finite prefixes, S, S' name the inspected StreamName windows, M, M' name the common-prefix distances, and V, V' supply only the finite ultrametric comparison boundary.

Use the StreamName readback and common-prefix exactness clauses of Definition 3485.2. The longer window in S' rereads every coordinate already present in S, and the hypothesis says that those displayed coordinates are unchanged.

Compare the two finite prefix rows componentwise through H and H'. The shared coordinates are identified by hsame, while the additional coordinates in the later packet can only shrink the cylinder by imposing more displayed prefix data; they cannot move a stream outside the earlier cylinder.

The distance comparison is then read from M, M' with V, V' used only as a finite strong-triangle boundary. Extending a common displayed prefix cannot create an earlier disagreement on the already inspected window, so the common-prefix ultrametric distance on that window cannot increase.

The remaining rows C, C', P, P', N, N' replay, source, and name this finite route. Since no row contains a selected infinite branch, quotient stream equality, countable choice, host stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the refinement stays inside finite prefix evidence before any $\mathsf{CompleteBaireSpace}^{\uparrow}$ completion row is read. □

Theorem 3485.6 (Baire ultrametric finite-cylinder completion handoff)**.** *Every* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *consumer reading a limit boundary from an accepted* $\mathsf{BaireUltrametric}^{\uparrow}$ *packet* $U = (B, M, V, S, H, C, P, N)$ *must first expose the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row* B*, the* $\mathsf{BaireMetric}^{\uparrow}$ *distance row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, the* $\mathsf{UltrametricSpace}^{\uparrow}$ *comparison row* V*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*. The completion handoff is therefore a finite-cylinder route, not a selected branch, quotient stream equality, countable choice, host stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier through Definition 3485.1. The only rows visible before a completion-facing boundary are the prefix row, distance row, ultrametric comparison row, finite StreamName window, and the structural support rows.

Apply Definition 3485.2. Carrier admission and StreamName readback order the read through B, M, V, S, H, C, P, N, while common-prefix distance exactness keeps the distance row attached to displayed finite prefix witnesses.

Use finite-cylinder separation and refinement, Definition 3485.4 and Definition 3485.5. Separation gives the positive finite-cylinder handoff before a completion row can read a limit boundary, and refinement keeps later windows inside the same displayed prefix route.

Finally repeat the nonescape clauses from the local NameCert surface. Since the packet has no coordinate for a selected branch, quotient stream equality, countable choice, host stream equality, or ambient completeness, a $\mathsf{CompleteBaireSpace}^{\uparrow}$ consumer receives only the finite-cylinder route before the limit boundary. □

Lean4 ✓ `BEDC.Derived.BaireUltrametricUp.BaireUltrametricCarrier finite cylinder completion handoff`

Falsifiable prediction: Every $\mathsf{BaireUltrametric}^{\uparrow}$ read factors through a displayed $\mathsf{BaireSpace}^{\uparrow}$ prefix row, a $\mathsf{BaireMetric}^{\uparrow}$ common-prefix distance row, a finite $\mathsf{StreamName}^{\uparrow}$ window row, an $\mathsf{UltrametricSpace}^{\uparrow}$ strong-triangle comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bairespace, bairemetric, ultrametricspace, streamname, complete-bairespace: $\mathsf{BaireUltrametric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite Baire prefixes, common-prefix dyadic distances, ultrametric comparisons, finite stream windows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3485.1, Definition 3485.2, Definition 3485.3, and Definition 3485.4 seed the common-prefix ultrametric route from Baire prefixes through prefix distances, finite StreamName windows, finite-cylinder separation, ultrametric comparison rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireUltrametric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host stream equality, quotient stream equality, selected branches, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-prefix distance exactness, finite-cylinder classifier stability, finite-cylinder separation, StreamName readback, prefix stability, and nonescape for BEDC.Derived.BaireUltrametric

Constructive story: A $\mathsf{BaireUltrametric}^{\uparrow}$ packet is generated from finite Baire prefix rows, BaireMetric common-prefix distance rows, UltrametricSpace comparison rows, StreamName finite windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3486

A Concrete Naming Certificate for $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$

The $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ packet realizes the observer-expansion reading of section 9463.4 as finite BEDC certificate data. It sits after the disk-coordinate surface of chapter 377, the complex phase readout of chapter 39, the finite flow route of chapter 195, the Poincare disk transport packet of chapter 3455, and the metric-read packet of chapter 3474. The chapter names expansion as a records-side ledger reading, not as global time, physical cosmology, or object-level boundary fixedness.

Definition 3486.1 (Hyperbolic expansion ledger carrier)**.** *A* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (F, D, G, M, B, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the focused-ledger row recording the visible record family whose growth is being read.*
2. *D is the disk-readout row over* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*, and* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *coordinates.*
3. *G is the* Cont*-growth row comparing local continuation extension with successive ledger readouts.*
4. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row consumed only after F, D, G have been displayed.*
5. *B is the boundary-refusal row excluding global clock, physical cosmology, completed boundary object, and object-level boundary fixedness.*
6. *A is the provenance row tying F, D, G, M, B to the hyperbolic observer-expansion reading.*
7. *H records componentwise* hsame *transport for focused-ledger, disk-readout, growth, metric-read, boundary-refusal, provenance, replay, and local naming rows.*
8. *C records displayed* Cont *replay through the focused ledger, disk readout, growth route, metric read, and refusal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$*,* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{Complex}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicExpansionLedger}^{\uparrow}}$ *row.*

The carrier exports expansion only as a finite focused-ledger and metric-read route. It contains no global clock, no empirical cosmology, no completed boundary model, and no object-level fixed boundary.

Theorem 3486.2 (Hyperbolic expansion ledger NameCert obligations)**.** *For every accepted* HyperbolicExpansionLe *carrier* $X = (F, D, G, M, B, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicExpansionLedger}^{\uparrow}}$ *surface consists exactly of focused-ledger admission, disk-readout admission,* Cont*-growth routing, metric-read admission, boundary-refusal exactness, provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted expansion read factors through*

$$F \longmapsto D \longmapsto G \longmapsto M \longmapsto B \longmapsto A \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Lean4 ✓ `BEDC.Derived.HyperbolicExpansionLedgerUp.HyperbolicExpansionLedgerTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3486.1. The focused ledger F, disk readout D, growth row G, metric read M, and boundary-refusal row B are displayed before provenance and naming are consumed.

Read the route in the stated order. The disk metric is available only after the focused ledger and Cont-growth route have been displayed, so expansion is a records-side reading rather than a global clock.

The rows A, H, C, P, N then record provenance, transport, replay, package sourcing, and local naming. Since no global clock, physical cosmology, completed boundary model, or object-level fixed boundary has a carrier coordinate, none can be exported by the NameCert surface. □

Theorem 3486.3 (Hyperbolic expansion ledger Cont-growth route)**.** *Every accepted expansion read of a* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *packet factors through the focused ledger and* Cont*-growth route before disk metric consumption:*

$$F, D, G \longmapsto M.$$

The hyperbolic metric row M may compare displayed disk readouts only after the local continuation-growth row G has been opened, and the boundary-refusal row B prevents the comparison from becoming a global clock, physical cosmology, completed boundary, or object-level fixedness claim.

Proof. Start with a metric consumer of M. Carrier projection shows that M is downstream of the focused ledger F, disk readout D, and continuation-growth row G.

Use the displayed Cont replay row C. It can replay metric consumption only along the focused-ledger and growth route, so the disk metric cannot be read as an independent time parameter.

Finally apply boundary-refusal exactness through B. The packet refuses global clock, physical cosmology, completed boundary, and object-level fixedness readings, leaving only the finite records-side expansion route. □

Falsifiable prediction: Every accepted expansion read factors through the focused ledger and Cont-growth route before disk metric consumption; a consumer that reads the hyperbolic metric without F, D, G refutes the $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ carrier.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, unitdisk, dynsystem, complex: no listed sibling is bijective to $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ because the packet jointly binds focused-ledger growth, disk readout, Cont-growth routing, metric consumption, boundary refusal, provenance, transport, replay, package sourcing, and local naming.

The Lean follow-up for this packet must provide explicit Nontrivial HyperbolicExpansionLedger$^{\uparrow}$ and FieldFaithful HyperbolicExpansionLedger$^{\uparrow}$ TasteGate instances. Those instances must keep $F, D, G, M, B, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$.

Constructive story: A HyperbolicExpansionLedger$^{\uparrow}$ packet is generated from a focused-ledger row, disk-readout row, Cont-growth route, HyperbolicMetric read, boundary-refusal row, provenance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$, seed)

Scope closed: Definition 3486.1, Definition 3486.2, and Definition 3486.3 seed the finite observer-expansion route through focused-ledger, disk-readout, Cont-growth, metric-read, boundary-refusal, provenance, transport, replay, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicExpansionLedger}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HyperbolicExpansionLedgerUp.HyperbolicExpansionLedgerTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close global clock semantics, physical cosmology, completed boundary geometry, object-level boundary fixedness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, focused-ledger rows, disk-readout rows, Cont-growth routing, metric-read rows, boundary-refusal exactness, TasteGate nontriviality, field faithfulness, provenance, and local naming for BEDC.Derived.HyperbolicExpansionLedgerUp.

Chapter 3487

A Concrete Naming Certificate for $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$

The $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ packet realizes the reversible phase-transport conservation row of Definition 9463.4 as finite NameCert data. It sits between the complex-history surface of chapter 39, the phase-transport carrier of chapter 3455, the metric readback of chapter 3474, and the expansion ledger of chapter 3486. The packet records conservation only as auditable preservation of source, transport, classifier, and GAP provenance rows; it does not assert a completed symplectic manifold, a global Hamiltonian flow, a host Liouville measure, or an ambient unitary group.

Falsifiable prediction: Any accepted $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ route preserves source, transport, classifier, and GAP provenance rows before a conserved phase read is exported.

Independence witness: complex, unitarygroup, dynsystem, poincare-disk-phase-transport, hyperbolic-metric: the ledger-conservation packet is not bijective to these siblings because it records reversible preservation of finite source, route, classifier, and GAP provenance rows rather than a complex carrier, ambient unitary group, dynamical system, disk-transport chart, or metric tensor alone.

Definition 3487.1 (Liouville ledger-conservation carrier)**.** *A* $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, T, K, G, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source phase-surface row, carrying the finite visible histories whose distinctions must remain readable.*
2. *T is the reversible transport row, read through the local phase-transport surface and recording the forward and return continuations as displayed data.*
3. *K is the classifier-preservation row, requiring the transported surface to preserve the classifier readout attached to S.*
4. *G is the* GAP *provenance row, recording that residues, exits, and unresolved boundary data remain ledger-visible after transport.*
5. *Q is the conserved phase-read row. It is exposed only after S, T, K, G have all been displayed.*
6. *H records componentwise* hsame *transport for source, reversible transport, classifier preservation,* GAP *provenance, conserved read, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a conservation request through S, T, K, G, Q, H, P, N.*

8. *P is the* Pkg *provenance over* LiouvilleLedgerConservation$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicExpansionLedger$^{\uparrow}$, Complex$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose conserved phase read Q is downstream of source exposure, reversible transport, classifier preservation, and GAP *provenance. It has no coordinate for host measure theory, a symplectic manifold, an ambient Hilbert space, a hidden unitary operator, or a global dynamical-system solution.*

Theorem 3487.2 (Liouville ledger-conservation NameCert obligations)**.** *For every accepted* LiouvilleLedgerConserva *carrier*

$$L = (S, T, K, G, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the source phase surface, reversible transport, classifier preservation,* GAP *provenance, conserved phase read, transport, replay, package provenance, and local naming rows;*

2. *source preservation, requiring every conserved read to keep the displayed source distinctions of S visible after transport;*

3. *route reversibility, requiring T to expose both the forward and return transport rows before conservation is read;*

4. *classifier preservation, requiring K to certify the transported classifier readout before Q is exported;*

5. GAP *ledger preservation, requiring G to keep residue and boundary provenance visible rather than erasing it into the conserved read;*

6. *structural non-escape, requiring H, C, P, N to preserve and name the same packet without adding an ambient unitary group, host Hamiltonian flow, completed phase volume, or hidden global state.*

Future Lean `TasteGate.lean` *will provide explicit* `Nontrivial LiouvilleLedgerConservationUp` *and* `FieldFaithful LiouvilleLedgerConservationUp` *instances so the accepted conservation packet remains distinct from refused routes and faithful reads preserve all displayed rows.*

Proof. Carrier exposure is projection from Definition 3487.1. The displayed packet contains exactly the source, reversible transport, classifier, GAP provenance, conserved read, and structural rows.

Source preservation and route reversibility are enforced before any conserved read is available. The row Q is downstream of S and T, so a consumer cannot export conservation after forgetting either the source distinctions or the return route.

Classifier and GAP preservation are separate rows. The classifier row K records preservation of the readout, while G keeps residues and boundary provenance visible as provenance rather than absorbing them into the phase read.

The structural rows then confine the packet. Componentwise hsame transport, displayed Cont replay, package provenance, and local naming preserve only S, T, K, G, Q and add no coordinate for the excluded host structures. □

Lean4 ✓ `BEDC.Derived.LiouvilleLedgerConservationUp.LiouvilleLedgerConservation namecert obligations`

Theorem 3487.3 (Liouville ledger-conservation export discipline)**.** *Every accepted* $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ *route exports a conserved phase read only along*

$$S, T, K, G \longmapsto Q \longmapsto H, C, P, N.$$

Consequently a downstream consumer receives source exposure, reversible transport, classifier preservation, GAP *provenance, the conserved read, transport, replay, package provenance, and local naming, and receives no host Liouville measure, ambient unitary operator, global dynamical system, completed symplectic phase space, or hidden state.*

Proof. Project an accepted carrier $L = (S, T, K, G, Q, H, C, P, N)$. The only conservation-facing row is Q, and it is displayed after the source, transport, classifier, and GAP rows.

Replay through C preserves that order. A downstream conservation request must pass through S, T, K, G before it can read Q, so the conserved phase read remains tied to visible source distinctions, route data, classifier preservation, and boundary provenance.

The structural rows authenticate only that finite route. Since the carrier has no host-measure, unitary-operator, global-dynamics, symplectic-manifold, or hidden-state coordinate, none of those structures can be exported from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3487.1, Definition 3487.2, and Definition 3487.3 seed the finite reversible ledger-conservation surface over source, transport, classifier, GAP provenance, conserved-read, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LiouvilleLedgerConservation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Liouville measure theory, ambient unitary dynamics, completed symplectic geometry, global Hamiltonian flow, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source preservation, reversible transport, classifier preservation, GAP ledger preservation, export discipline, Nontrivial LiouvilleLedgerConservationUp, and FieldFaithful LiouvilleLedgerConservationUp.

Constructive story: A $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ packet is generated from finite source phase-surface rows, reversible transport rows, classifier-preservation rows, GAP provenance rows, a conserved phase-read row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3488

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet records the located-real completion boundary as finite NameCert data. It consumes the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the located-real observation row of chapter 1933, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a human-origin completion route for displayed located Cauchy data; it is not a quotient-real construction, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3488.1 (Bishop located-real completion carrier)**.** *A* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the requested radii and comparison budgets.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family inspected at those dyadic tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the displayed finite windows.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locatedness row recording lower–upper separation evidence for the same readback windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R, L have been displayed.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, locatedness, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each completion-facing request through D, S, R, L, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected representatives, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedRealCompletionUp.BishopLocatedRealCompletionTasteGate single carrier alignment`

Theorem 3488.2 (Bishop located-real completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier* $K = (D, S, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, finite StreamName window exposure, RegSeqRat readback, LocatedReal locatedness exposure, Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every completion-facing consumer reads only the displayed route*

$$D \longmapsto S \longmapsto R \longmapsto L \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3488.1. Its analytic rows are the dyadic tolerance ledger, finite stream windows, regular rational readback, locatedness row, and Real seal.

Read the route in order. The tolerance row D fixes the finite request, S opens the corresponding window, R repacks that window as regular rational data, and L supplies the located lower–upper evidence attached to the same displayed readback.

The seal row E is downstream of those four rows. The structural rows H, C, P, N transport, replay, source, and name the route, while quotient streams, selected limits, host equality, choice, trichotomy, compactness, and ambient completeness have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.BishopLocatedRealCompletionUp.BishopLocatedRealCompletion namecert obligations`

Theorem 3488.3 (Bishop located-real completion tolerance-window exactness)**.** *For an accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *carrier* $K = (D, S, R, L, E, H, C, P, N)$*, every located completion request fixes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before selecting the* $\mathsf{StreamName}^{\uparrow}$ *window* S*. The* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and* $\mathsf{LocatedReal}^{\uparrow}$ *row* L *are then read over that same window before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is exposed.*

Proof. Project the carrier of Definition 3488.1. The tolerance ledger D is the first analytic row of the packet, and the finite window row S is admitted only as the stream window inspected at those dyadic tolerances.

Apply the NameCert obligations of Definition 3488.2. They force the completion route to read D before S, then R, then L, and only then the terminal seal E.

The located Cauchy handoff of Definition 3488.4 keeps the regular readback and locatedness evidence on the same displayed window. Since quotient streams, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, no completion request can replace this tolerance-window route with a broader Real construction. □

Theorem 3488.4 (Bishop located-real completion located Cauchy handoff)**.** *Every* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *consumer that treats a displayed regular Cauchy name as a located real must pass through* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{LocatedReal}^{\uparrow}$ *row before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exposes located Cauchy evidence only as the finite packet* $D, S, R, L, E, H, C, P, N$*.*

Proof. Start with the NameCert obligations. They place the dyadic tolerance ledger before the window family and the regular rational readback.

The locatedness row is then forced to share the same finite windows. Since L is read after R, a consumer cannot substitute a host located real, a quotient stream class, or a selected representative for the displayed regular readback.

Finally the Real seal is reached through E and preserved by H, C, P, N. The handoff is therefore the finite BEDC route through chapter 291, chapter 126, chapter 301, chapter 1933, and chapter 38. □

Theorem 3488.5 (Located-real completion Cauchy representation)**.** *Every accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *packet exposes its regular Cauchy representation only through the shared rows* D, S, R, L *before the terminal* E *seal. Thus* $\mathsf{Real}^{\uparrow}$ *consumers can read DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, and LocatedReal locatedness evidence, but they cannot read quotient-stream representatives, selected limits, host equality, countable choice, trichotomy, or ambient Real completeness from the packet.*

Proof. Project the carrier of Definition 3488.1. The rows D, S, R, L are the only analytic coordinates available before the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Apply Definition 3488.2 and Definition 3488.3. These results force the representation route to pass through the displayed tolerance ledger, finite window, regular rational readback, and locatedness row on the same window.

Now apply Definition 3488.4 and Definition 3488.6. The handoff identifies the regular Cauchy representation consumed by located-real clients, and the seal nonescape theorem excludes quotient-stream representatives, selected limits, host equality, choice, trichotomy, and ambient completeness from the exported $\mathsf{Real}^{\uparrow}$ surface. □

Theorem 3488.6 (Bishop located-real completion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *packet factors through the displayed rows* $D, S, R, L, E, H, C, P, N$*. The seal exports no quotient-real equality, selected host limit, host real equality test, countable-choice selector, trichotomy split, open-cover compactness witness, or ambient completeness theorem.*

Proof. Read the carrier definition. The only route to E is through the tolerance ledger, finite windows, regular readback, and locatedness row.

Apply the located Cauchy handoff. It identifies D, S, R, L as the exact finite route consumed before the terminal seal can be inspected.

The structural rows do not add analytic content. They preserve and name the displayed route, while the refused quotient, selected, host, choice, order, compactness, and ambient-completeness objects are absent from the carrier and from the local NameCert provenance. □

Falsifiable prediction: A claimed $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the displayed DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, and LocatedReal locatedness row refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, locatedreal, real: no listed sibling supplies the whole $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet, because each sibling contributes only tolerance, window, readback, locatedness, or terminal seal data while this chapter binds all five rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, LocatedReal locatedness row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3488.1, Definition 3488.2, Definition 3488.4, and Definition 3488.6 seed the finite located-real completion route through dyadic tolerances, stream windows, regular rational readback, locatedness rows, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real construction, selected host limits, countable choice, trichotomy, open-cover compactness, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, regular readback, locatedness handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealCompletion

Chapter 3489

A Concrete Naming Certificate for $\mathsf{LocatedDyadicCut}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedDyadicCut}^{\uparrow}$ packet records a two-sided dyadic located-cut interface for Real and Completion consumers. It sits after the dyadic tolerance arithmetic of chapter 291, the finite window discipline of chapter 126, the regular rational readback route of chapter 301, the located-cut surface of chapter 404, and the regular Cauchy located-cut handoff of chapter 7018. The packet names lower and upper dyadic rows with a located gap request before the terminal $\mathsf{Real}^{\uparrow}$ seal; it is not a quotient Dedekind real, a selected limit, countable choice, trichotomy, or ambient completeness principle.

Definition 3489.1 (Located dyadic cut carrier)**.** *A* $\mathsf{LocatedDyadicCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic cut row, recording rounded lower observations at displayed dyadic scales.*
2. *U is the upper dyadic cut row, recording rounded upper observations over the same displayed dyadic scales.*
3. *G is the located gap request row, exposing the lower–upper separation request before a consumer treats the cut as located.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic row used to compare dyadic endpoints and request radii.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row naming the finite windows used by the dyadic observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational data attached to those windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after L, U, G, D, S, R have stayed visible.*
8. *H records componentwise* hsame *transport for lower, upper, gap, arithmetic, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each Real-facing or Completion-facing request through $L, U, G, D, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedDyadicCut}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicCut}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $L, U, G, D, S, R, E, H, C, P, N$. It has no coordinate for quotient Dedekind equality, selected Cauchy limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.LocatedDyadicCutUp`

Theorem 3489.2 (Located dyadic cut NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDyadicCut}^{\uparrow}$ *carrier $K = (L, U, G, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicCut}^{\uparrow}}$ *surface consists exactly of carrier admission for those eleven rows; lower roundedness through L; upper roundedness through U; located gap exposure through G; dyadic tolerance handoff through D; finite window inspection through S; regular rational readback through R; Real-seal nonescape through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3489.1. Its analytic rows are exactly the lower cut, upper cut, located gap request, dyadic tolerance arithmetic, stream inspection, regular rational readback, and terminal Real seal.

Read the lower and upper sides separately. Roundedness for the lower row is recorded only in L, roundedness for the upper row only in U, and locatedness is exposed only after the gap request row G is displayed with the same dyadic tolerance row D.

Now route the numeric data through S and R. The finite StreamName windows and RegSeqRat readback bind the dyadic observations to displayed regular rational data before the seal row E can be consumed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no refused quotient, selected limit, choice, trichotomy, or ambient-completeness object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3489.3 (Located dyadic cut Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{LocatedDyadicCut}^{\uparrow}$ *carrier $K = (L, U, G, D, S, R, E, H, C, P, N)$ must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{LocatedCut}^{\uparrow}$ *lower–upper row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before using the cut as a Real-facing boundary. The consumer route is the finite handoff*

$$D, S, R, L, U, G \longmapsto E, H, C, P, N$$

refined by the regular Cauchy located-cut sibling of chapter 7018 and the dyadic arithmetic core of chapter 291.

Proof. Start with a Real-facing or Completion-facing consumer. Carrier projection exposes D, S, R, L, U, G before E, so the consumer cannot read the terminal seal without first opening tolerance arithmetic, finite windows, regular rational readback, and the two located-cut sides.

Use the sibling route through chapter 7018. It identifies the regular Cauchy readback discipline consumed by the lower and upper located rows, while chapter 291 supplies the dyadic tolerance handoff used by the gap request.

The structural rows then preserve the same boundary. Componentwise transport H, replay C, provenance P, and local naming N keep the seal attached to the displayed rows and provide no coordinate for quotient Dedekind completeness, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: A claimed $\mathsf{LocatedDyadicCut}^{\uparrow}$ consumer that reaches the $\mathsf{Real}^{\uparrow}$ seal without first reading DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, LocatedCut lower–upper rows, and the located gap request refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, locatedcut, real: no listed sibling supplies the whole $\mathsf{LocatedDyadicCut}^{\uparrow}$ packet, because each sibling contributes only arithmetic, window, readback, lower–upper, or terminal seal data while this chapter binds all rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDyadicCut}^{\uparrow}$ packet is generated from lower and upper dyadic cut rows, a located gap request, DyadicRatCore tolerance arithmetic, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDyadicCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3489.1, Definition 3489.2, and Definition 3489.3 seed the finite two-sided dyadic located-cut route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDyadicCut}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.LocatedDyadicCutUp`

Bridge status: none

Not claimed: This chapter does not close quotient Dedekind completeness, selected limits, countable choice, trichotomy, ambient RealUp completeness, checked obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower and upper roundedness, located gap exposure, dyadic tolerance handoff, finite window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDyadicCutUp.

Chapter 3490

A Concrete Naming Certificate for $\mathsf{KirszbraunExtension}^{\uparrow}$

Following Kirszbraun and the Banach-space tradition around nonexpansive and Lipschitz extension theorems, the $\mathsf{KirszbraunExtension}^{\uparrow}$ packet records the finite certificate surface needed to read a Lipschitz map on a metric subspace as a convex inner-product-valued extension request. It sits beside the metric subspace packet of chapter 1519, the Lipschitz map packet of chapter 410, the reusable extension interface of chapter 3650, the convex target surface of chapter 208, the inner-product target source of chapter 86, and the metric source of chapter 57. The chapter names the finite Kirszbraun-facing route without claiming Hilbert-space completeness, a host extension operator, pointwise equality of functions, or a selected global formula.

Definition 3490.1 (Kirszbraun extension carrier)**.** *A* $\mathsf{KirszbraunExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, A, L, Y, V, B, F, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *row for the source observations.*
2. *A is the* $\mathsf{MetricSubspace}^{\uparrow}$ *row, including the membership, inclusion, and inherited-distance route from A into X.*
3. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row on A, carrying the finite graph window and displayed Lipschitz constant.*
4. *Y is the target* $\mathsf{InnerProduct}^{\uparrow}$ *row, read through its vector, scalar-form, norm-seed, and metric-seed observations.*
5. *V is the target* $\mathsf{ConvexSet}^{\uparrow}$ *row inside the inner-product carrier, recording the admitted convex codomain surface for extension values.*
6. *B is the Lipschitz-constant ledger, recording the rational bound, the source-distance reads through A, X, and the target-distance reads through Y, V.*
7. *F is the extension graph row over the ambient metric window X with values admitted by V.*
8. *R is the restriction and bound row, requiring source points admitted by A to replay the graph of L and requiring every displayed ambient pair of F to respect the same ledger B.*

9. *H records componentwise* hsame *transport for the ambient metric, subspace, source Lipschitz map, target inner-product, convex codomain, Lipschitz ledger, extension graph, restriction, replay, provenance, and local name rows.*
10. *C records displayed* Cont *replay from a Kirszbraun extension request through $X, A, L, Y, V, B, F, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{KirszbraunExtension}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers componentwise through these rows. It contains no coordinate for a host Hilbert space, a global extension operator, quotient equality of functions, countable choice, ambient compactness, or unbounded real completion.

Lean4 ✓ `BEDC.Derived.KirszbraunExtensionUp.KirszbraunExtensionTasteGate single carrier alignment`

Theorem 3490.2 (Kirszbraun extension NameCert obligations)**.** *For every accepted* $\mathsf{KirszbraunExtension}^{\uparrow}$ *carrier $K = (X, A, L, Y, V, B, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, metric-subspace inclusion through A, source Lipschitz-map admission through L, inner-product target admission through Y, convex codomain admission through V, Lipschitz-constant ledger readback through B, extension-graph admission through F, restriction and bound exactness through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The finite route is

$$X, A \longmapsto L \longmapsto Y, V \longmapsto B, F, R \longmapsto H, C, P, N.$$

No obligation is discharged by a host extension formula, a quotient of maps, or a hidden Hilbert-space theorem.

Proof. Carrier admission is projection from Definition 3490.1. The displayed rows are exactly $X, A, L, Y, V, B, F, R, H, C, P, N$.

The source side is read through X, A, L. The subspace row admits source points and inherited distances, and the Lipschitz-map row exposes the finite graph window and bound that the extension must preserve.

The target side is read through Y, V. The inner-product row supplies the target metric seed, while the convex-set row supplies the admitted value surface for extension graph reads.

The ledger and graph clauses are B, F, R. The ledger stores the single Lipschitz constant and both source and target distance reads, the graph row stores ambient observations, and the restriction row forces source points to replay the original Lipschitz map.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no host extension operator, quotient-function coordinate, choice row, compactness row, or completion theorem coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3490.3 (Kirszbraun extension non-escape)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, *or* $\mathsf{InnerProduct}^{\uparrow}$ *consumer of an accepted* $\mathsf{KirszbraunExtension}^{\uparrow}$ *packet factors through*

$$X, A, L, Y, V, B, F, R, H, C, P, N.$$

It may read the packet as a finite extension candidate whose graph restricts to the source Lipschitz map and whose target values remain in the displayed convex inner-product row, but it cannot export a host Kirszbraun theorem, a chosen extension operator, a quotient-map equality, countable choice, compactness, or ambient Hilbert completeness.

Proof. Start from the obligation surface of Definition 3490.2. Every accepted consumer route must read the ambient metric, subspace, source Lipschitz map, target inner-product row, convex codomain row, ledger, graph, restriction, transport, replay, provenance, and name rows.

A source-facing consumer reaches the extension only through A, L, R. Thus an admitted source point is compared with the original Lipschitz graph before any ambient graph read is treated as an extension value.

A target-facing consumer reaches values only through Y, V, B, F. The inner-product row supplies the metric seed, the convex row admits values, and the ledger checks the same Lipschitz constant against displayed distances.

The remaining rows H, C, P, N repack that route. They preserve and name the carrier but do not add a host theorem, operator selector, quotient equality, choice schedule, compactness witness, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3490.1, Definition 3490.2, and Definition 3490.3 seed the finite Kirszbraun extension carrier over metric subspace, Lipschitz map, convex target, inner-product target, extension graph, and Lipschitz-ledger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KirszbraunExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host Kirszbraun theorem, global Hilbert completeness, quotient equality of functions, selected extension operators, countable choice, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subspace inclusion, source Lipschitz admission, convex inner-product target admission, Lipschitz-ledger readback, graph admission, restriction exactness, transport, replay, provenance, and non-escape for BEDC.Derived.KirszbraunExtensionUp.

Constructive story: A $\mathsf{KirszbraunExtension}^{\uparrow}$ packet is generated from an ambient Metric row, a MetricSubspace source row, a source LipschitzMap row, an InnerProduct target row, a ConvexSet target row, a Lipschitz-constant ledger, an extension graph, a restriction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3491

A Concrete Naming Certificate for HyperbolicGeodesicBoundary$^{\uparrow}$

The HyperbolicGeodesicBoundary$^{\uparrow}$ packet realizes the boundary-geodesic route of chapter 9463 as finite Poincare-disk, hyperbolic-metric, phase-transport, and provenance rows. It sits after the disk-coordinate surface of chapter 377, reversible transport in chapter 219, finite flow rows in chapter 195, the disk-phase carrier of chapter 3455, and the metric-read surface of chapter 3474. The packet names geodesic-boundary data without treating the boundary as a completed host circle, quotient ideal point, physical horizon, or observer substance.

Definition 3491.1 (HyperbolicGeodesicBoundary carrier)**.** *A* HyperbolicGeodesicBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, T, B, J, F, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-coordinate row, including the finite radius and boundary-approach observations used by the packet.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *T is the* PoincareDiskPhaseTransport$^{\uparrow}$ *transport row, retaining reversible phase movement, finite flow data, and fiber provenance.*
4. *B is the boundary-geodesic incidence row, recording which finite disk observations are read as approaching the same geodesic far-end.*
5. *J is the classifier-stability row, preserving disk-coordinate, hyperbolic-metric, phase-transport, and boundary-geodesic classifiers under the displayed route.*
6. *F is the refusal row excluding quotient boundary equality, completed geodesic compactification, host horocycle selection, physical cosmology, and observer-substance readings.*
7. *A is the audit row tying D, M, T, B, J, F to the hyperbolic phase vision and to the Poincare-disk metric and transport packets.*
8. *H records componentwise* hsame *transport for disk-coordinate, metric-read, phase-transport, boundary-geodesic, classifier-stability, refusal, and audit rows.*
9. *C records displayed* Cont *replay through finite disk observations, metric reads, phase transport, boundary-geodesic checks, classifier stability, provenance, and refusal.*

10. *P is the* Pkg *provenance over* HyperbolicGeodesicBoundary$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskPhaseTra UnitaryGroup$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicGeodesicBoundary}^{\uparrow}}$ *row.*

The carrier exports boundary-geodesic data only through these rows. It has no coordinate for a completed hyperbolic plane, quotient ideal boundary, selected horocycle, global physical evolution law, or object-level observer substance.

Theorem 3491.2 (HyperbolicGeodesicBoundary NameCert obligations)**.** *For every accepted* HyperbolicGeodesicBo *carrier*

$$G = (D, M, T, B, J, F, A, H, C, P, N),$$

the local NameCert$_{\text{HyperbolicGeodesicBoundary}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric read admission, Poincare-disk phase-transport admission, boundary-geodesic incidence, classifier stability, refusal exactness, audit provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every boundary-geodesic consumer must factor through

$$D, M, T \longmapsto B, J, F, A \longmapsto H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicBoundaryUp.HyperbolicGeodesicBoundaryTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3491.1. The visible surface begins with disk-coordinate row D, metric-read row M, and phase-transport row T.

Read the incidence and stability rows next. The row B records boundary-geodesic incidence as finite far-end evidence, while J preserves the disk, metric, transport, and boundary classifiers across the displayed data.

The refusal and audit rows keep the packet aligned with the hyperbolic phase reading without exporting host geometry. Finally H, C, P, N transport, replay, source, and name only those displayed rows, so no quotient boundary, selected horocycle, physical horizon, or observer-substance reading can be extracted. □

Theorem 3491.3 (HyperbolicGeodesicBoundary phase handoff)**.** *Every phase-transport consumer of an accepted* HyperbolicGeodesicBoundary$^{\uparrow}$ *carrier receives the geodesic far-end only after the route*

UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, UnitaryGroup$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, Pkg

has been displayed. The consumer may use finite disk coordinates, hyperbolic metric reads, reversible phase transport, finite flow, boundary incidence, classifier stability, provenance, and local naming. It may not read quotient boundary equality, completed geodesic compactification, selected horocycles, physical cosmology, or observer substance.

Proof. Start from the NameCert obligation surface of Definition 3491.2. It places the disk-coordinate row, hyperbolic metric read, and phase-transport route before boundary incidence is exposed.

Use the phase-transport ledger of Definition 3455.2. The reversible transport and finite flow rows keep the source ledger visible while the boundary-geodesic row is read.

The hyperbolic metric obligations of Definition 3474.2 supply the metric-read discipline. Together with H, C, P, N, these rows give a finite phase handoff and no stronger host-side boundary object. □

Falsifiable prediction: A $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ consumer that obtains quotient boundary equality, selected horocycle data, or a completed geodesic compactification without first reading the route $D, M, T, B, J, F, A, H, C, P, N$ refutes the carrier surface.

Independence witness: hyperbolic_metric, unitdisk, unitarygroup, dynsystem, poincare_disk_phase_transpor $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk-coordinate observations, hyperbolic metric reads, phase transport, boundary-geodesic incidence, classifier stability, refusal, audit provenance, transport, replay, package sourcing, and local NameCert rows.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial\ HyperbolicGeodesicBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicGeodesicBoundary}^{\uparrow}$ TasteGate instances. Those instances must keep $D, M, T, B, J, F, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ packet is generated from UnitDisk disk coordinates, HyperbolicMetric metric reads, PoincareDiskPhaseTransport rows, boundary-geodesic incidence, classifier stability, refusal exactness, audit provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3491.1, Definition 3491.2, and Definition 3491.3 seed the finite boundary-geodesic route over disk-coordinate, hyperbolic-metric, phase-transport, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HyperbolicGeodesicBoundaryUp.HyperbolicGeodesicBoundaryTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient boundary equality, completed geodesic compactification, selected horocycles, physical cosmology, observer-substance readings, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-geodesic incidence, metric-read stability, phase-transport handoff, refusal exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicGeodesicBoundaryUp.

Chapter 3492

A Concrete Naming Certificate for $\mathsf{UniformClosure}^{\uparrow}$

Following Bishop-style metric and uniform-space practice, the $\mathsf{UniformClosure}^{\uparrow}$ packet records the closure interface whose tests are uniform-neighbourhood rows rather than host open covers. It sits between the metric closure surface of chapter 3342, the completion handoff of chapter 1506, the separation boundary of chapter 1411, the terminal real seal of chapter 38, and the regular-rational readback route of chapter 301. The packet names closure by finite entourage, window, and seal data; it is not open-cover compactness, quotient closure, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3492.1 (UniformClosure carrier)**.** *A* $\mathsf{UniformClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricClosure}^{\uparrow}$ *row naming the metric closure surface whose point, source, and candidate-limit data are being read.*
2. *Q is the uniform-neighbourhood ledger, recording the finite entourage requests admitted for the same carrier.*
3. *S is the source subset or located-source row being closed under those uniform requests.*
4. *L is the candidate closure point, read only through the displayed uniform and metric rows.*
5. *E is the entourage-to-metric comparison row that routes uniform requests into the metric tests already visible in M.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window surface used to inspect the approximating observations and the candidate point.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows, tolerances, and comparison requests.*
8. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, Q, S, L, E, W, R have been displayed.*
9. *H records componentwise* hsame *transport for the metric-closure, uniform-ledger, source, candidate, comparison, window, readback, and seal rows.*
10. *C records displayed* Cont *replay from an entourage request through the metric-closure test, finite windows, regular-rational readback, and real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{UniformClosure}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose uniform requests, metric closure route, candidate point, finite windows, regular readbacks, real seal, transport, replay, provenance, and local naming are displayed. It has no coordinate for open-cover compactness, quotient closure, selected limits, a quotient completion object, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformClosureUp.UniformClosureTasteGate single carrier alignment`

Theorem 3492.2 (UniformClosure NameCert obligations)**.** *For every accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier* $U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{MetricClosure}^{\uparrow}$ *row, uniform-neighbourhood ledger, source row, candidate closure point, entourage-to-metric comparison row, finite window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every public uniform-closure read must factor through

$$M, Q, S, L, E, W, R, Z, H, C, P, N$$

before the candidate closure point can be consumed.

Proof. Project the carrier of Definition 3492.1. The only admissible source data are the metric-closure row M, uniform ledger Q, source row S, and candidate closure point L.

The comparison row E then binds each uniform request to the metric tests already exposed by M. The finite window row W, regular-rational readback R, and real seal Z give the displayed numerical route for those tests.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row stores open-cover compactness, quotient closure, selected limits, a quotient completion object, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3492.3 (UniformClosure completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier reads the closure candidate only through the finite route*

$$\mathsf{MetricClosure}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The consumer receives the displayed metric-closure row, uniform-neighbourhood ledger, entourage-to-metric comparison, regular-rational readback, real seal, separation-facing identity boundary, transport, replay, provenance, and local naming. It receives no selected completion limit, quotient closure object, open-cover compactness principle, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the public route of Definition 3492.2. It places the uniform ledger and entourage-to-metric comparison before any completion-facing consumer can read the candidate point.

Use the metric-closure handoff of Definition 3342.3. The same finite windows and regular-rational readback rows feed the completion-facing side, so the consumer cannot detach the candidate from the displayed metric closure surface.

The separated-metric boundary of Definition 1411.48 supplies the identity discipline for zero-distance reads. Transport, replay, provenance, and naming through H, C, P, N preserve that route without adding selected limits, quotient closure, compactness, or ambient real completeness. □

Theorem 3492.4 (UniformClosure separated stability)**.** *Let $U = (M, Q, S, L, E, W, R, Z, H, C, P, N)$ be an accepted* $\mathsf{UniformClosure}^{\uparrow}$ *carrier. If a downstream separated-metric consumer compares two transported closure candidates from U, the comparison must pass through the displayed metric-closure row M, uniform ledger Q, comparison row E, finite windows W, regular-rational readback R, real seal Z, and the structural rows H, C, P, N before the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier can be read.*

Thus separated stability is a finite transport-and-seal obligation over the displayed rows, not a quotient closure equality or host identity theorem.

Proof. Project the two transported candidates through the componentwise hsame rows in H. The transport preserves the metric-closure row, uniform ledger, comparison row, windows, readback, and real seal as displayed rows of the same packet.

Replay the comparison through C. The route reaches the $\mathsf{Real}^{\uparrow}$ seal only after M, Q, E, W, R are fixed, and then uses the separated-metric identity boundary rather than host equality.

The provenance and local naming rows P, N expose this route as the public certificate. Since no quotient closure object, selected limit, open-cover compactness row, or ambient real-completeness coordinate occurs in U, the separated-stability conclusion remains confined to the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformClosure}^{\uparrow}$ packet is generated from a MetricClosureUp row, a uniform-neighbourhood ledger, source and candidate rows, an entourage-to-metric comparison row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$, seed)

Scope closed: Definition 3492.1, Definition 3492.2, Definition 3492.3, and Definition 3492.4 seed the finite uniform-closure interface for completion and separated-metric consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient closure, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-neighbourhood ledger stability, entourage-to-metric comparison, RegSeqRat readback, Real seal nonescape, completion handoff, separated stability, transport, replay, provenance, and local naming for BEDC.Derived.UniformClosureUp.

Chapter 3493

A Concrete Naming Certificate for CauchyFilterLocatedRealExtraction$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the CauchyFilterLocatedRealExtraction$^{\uparrow}$ packet records the finite route by which a Cauchy-filter surface is read as located-real window data. It consumes the Cauchy filter interface of chapter 3416, the located-real comparison surface of chapter 3395, the stream-name observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite extraction route only; it is not ambient Real completeness, a quotient filter construction, a selected limit, or a countable-choice principle.

Definition 3493.1 (Cauchy-filter located-real extraction carrier)**.** *A* CauchyFilterLocatedRealExtraction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, L, S, Q, D, R, T, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyFilter$^{\uparrow}$ *base row, exposing the filter-facing Cauchy surface and its finite request windows.*
2. *L is the* LocatedReal$^{\uparrow}$ *window row, recording the lower and upper located cuts read from the accepted filter request.*
3. *S is the* StreamName$^{\uparrow}$ *observation row, listing the finite observation windows through which the filter data are sampled.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, translating the sampled filter windows into regular rational approximants.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, recording the dyadic mesh, modulus, and separation checks used by the located window.*
6. *R is the* Real$^{\uparrow}$ *seal row for the extracted endpoint and readback data.*
7. *T records componentwise* hsame *transport for $F, L, S, Q, D, R, C, P, N$.*
8. *C records displayed* Cont *replay from a filter request through the observation, rational readback, dyadic tolerance, located-window, and Real-seal rows.*
9. *P is the* Pkg *provenance over* CauchyFilterLocatedRealExtraction$^{\uparrow}$*,* CauchyFilter$^{\uparrow}$*,* LocatedReal$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, L, S, Q, D, R, T, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient filters, selected limits, countable choice, host real equality, or an unrestricted completion principle.*

Lean4 ✓ `BEDC.Derived.CauchyFilterLocatedRealExtractionUp.CauchyFilterLocatedRealExtractionTasteGate`
`single carrier alignment`

Theorem 3493.2 (Cauchy-filter located-real extraction)**.** *Every accepted* $\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}$ *packet extracts a located-real finite-window route from the Cauchy-filter surface:*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported route is the displayed finite packet $E = (F, L, S, Q, D, R, T, C, P, N)$ and no stronger completeness, quotient-filter, selected-limit, or countable-choice object is produced.

Proof. Project the accepted carrier. The Cauchy-filter surface is F, the finite observation windows are S, the regular rational readback is Q, the dyadic tolerance ledger is D, the located-real window is L, and the terminal Real-facing seal is R.

Replay through C keeps these rows in extraction order. A filter request is sampled only through the displayed stream-name windows, those observations are read back as regular rational data, and the dyadic ledger checks the tolerance needed before the located window is exposed.

Transport and provenance do not add new mathematical objects. The rows T, P, N move and name only the displayed packet, so the extraction cannot escape to quotient filters, selected limits, countable choice, ambient Real completeness, or host real equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}$ packet is generated from a CauchyFilter base row, LocatedReal finite window, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 3493.1 and Definition 3493.2 seed the finite extraction route from Cauchy-filter windows to located-real readback data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyFilterLocatedRealExtraction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient filters, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observation-window readback, dyadic tolerance exactness, located-window extraction, Real-seal compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterLocatedRealExtractionUp.

Chapter 3494

A Concrete Naming Certificate for $\mathsf{SignedDyadicExpansion}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers presented by finite approximation data, the $\mathsf{SignedDyadicExpansion}^{\uparrow}$ packet records the finite signed dyadic prefix route before a prefix may be consumed as a regular rational name or sealed as a $\mathsf{Real}^{\uparrow}$ object. It sits between the binary expansion surface of chapter 418, the signed-digit stream interface of chapter 316, the signed dyadic normal form row of $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite signed dyadic readback data; it is not a quotient stream, a selected infinite expansion, a uniqueness theorem for dual binary tails, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3494.1 (Signed dyadic expansion carrier)**.** *A* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, W, E, T, F, A, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the sign row, recording whether the displayed prefix is read with positive, negative, or zero orientation before any numeric readback is opened.*
2. *W is the finite dyadic digit-prefix window, exposing the inspected binary cells and their request index as finite data.*
3. *E is the exponent and tail-radius ledger, recording the dyadic scale of the prefix and the displayed dyadic radius bounding the unseen tail.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the tail-radius ledger and by every readback request.*
5. *F is the* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *row, normalizing the sign, exponent, and visible digit window without selecting an infinite expansion.*
6. *A is the* $\mathsf{StreamName}^{\uparrow}$ *window row that identifies which finite prefix cells are inspected at each requested tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, W, E, T, F, A.*
8. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the signed prefix, dyadic tolerance, normal-form, stream-window, and regular-rational rows have been displayed.*

9. *H records componentwise* hsame *transport for the sign, digit window, exponent ledger, tail-radius ledger, normal form, stream window, regular readback, real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$S, W, E, T, F, A \longmapsto R \longmapsto L.$$

11. *P is the* Pkg *provenance over* $\mathsf{SignedDyadicExpansion}^{\uparrow}$, $\mathsf{BinaryExpansion}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, E, T, F, A, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite signed dyadic expansion, uniqueness at dual binary tails, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SignedDyadicExpansionUp.SignedDyadicExpansionTasteGate single carrier alignment`

Theorem 3494.2 (Signed dyadic expansion NameCert obligations)**.** *For every accepted* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier*

$$X = (S, W, E, T, F, A, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the sign row, finite dyadic digit-prefix window, exponent and tail-radius ledger, dyadic tolerance row, signed dyadic normal form, stream window, regular rational handoff, real seal, transport, replay, provenance, and local name;*

2. *sign-prefix coherence, requiring S and W to expose the same finite signed prefix before any rational readback is admitted;*

3. *exponent and tail-radius visibility, requiring E and T to bound the unseen dyadic tail by a displayed dyadic tolerance;*

4. *normal-form locality, requiring F to normalize only the displayed sign, exponent, and digit-window rows;*

5. *window exactness, requiring A to select exactly the finite prefix cells consumed by the requested tolerance;*

6. *handoff order, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume S, W, E, T, F, A before the terminal* $\mathsf{Real}^{\uparrow}$ *seal L becomes available;*

7. *componentwise transport, displayed replay, package provenance, and local naming for the same rows through H, C, P, N;*

8. *refusal of quotient stream equality, selected infinite expansions, dual-tail uniqueness, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No signed dyadic expansion obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.SignedDyadicExpansionUp.SignedDyadicExpansionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 3494.1. The accepted packet displays exactly the sign row, finite digit window, exponent and tail-radius ledger, dyadic tolerance row, signed normal form, stream window, regular readback, real seal, and structural rows.

The prefix information is read first. The sign row S and digit window W determine the visible signed dyadic prefix, while E and T expose the dyadic scale and the tail-radius budget that bounds all unseen cells relevant to the request.

The normal-form and window rows keep the readback local. The row F normalizes only the displayed finite prefix data, and A selects the finite cells consumed at the requested tolerance.

The handoff is ordered through R and L. A regular rational name may be read only after S, W, E, T, F, A are displayed, and the $\mathsf{Real}^{\uparrow}$ seal is terminal over that regular rational row.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores quotient stream equality, an infinite expansion selector, a dual-tail uniqueness decision, trichotomy, countable choice, or ambient completeness, none of those objects becomes a local obligation. □

Theorem 3494.3 (Signed dyadic expansion Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{SignedDyadicExpansion}^{\uparrow}$ *carrier factors through the finite route*

$$S, W, E, T, F, A \longmapsto R \longmapsto L \longmapsto H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives the sign row, finite dyadic prefix window, exponent and tail-radius ledger,* $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *window, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row before the regular rational handoff and terminal real seal are available. The route exports no quotient stream, selected infinite expansion, dual-tail uniqueness theorem, trichotomy, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a consumer request for the terminal real-facing row. The accepted carrier exposes L only after the signed prefix and tolerance route has been displayed.

The request first opens S, W, E, T, F, A. These rows determine the visible sign, dyadic digit prefix, exponent, tail radius, normal form, and inspected window as finite BEDC data.

The regular rational handoff R is then read from that same finite route. Because the tail-radius bound is already displayed in E and T, the handoff does not require a quotient stream or an infinite representative.

The seal L and structural rows H, C, P, N package the readback for downstream consumers. The excluded coordinates are absent from the carrier, so every exported real read remains a finite signed dyadic expansion certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3494.1, Definition 3494.2, and Definition 3494.3 seed the finite signed dyadic prefix route from sign and digit-window rows through DyadicRatCore tolerances, SignedDyadicNormalForm rows, StreamName windows, RegSeqRat handoff, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDyadicExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite expansions, dual-tail uniqueness, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, sign-prefix coherence, exponent and tail-radius visibility, normal-form locality, window exactness, handoff order, transport, replay, provenance, and local naming for BEDC.Derived.SignedDyadicExpansionUp.

Constructive story: A $\mathsf{SignedDyadicExpansion}^{\uparrow}$ packet is generated from a sign row, finite dyadic digit-prefix window, exponent and tail-radius ledger, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{SignedDyadicNormalForm}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{FrechetSpace}^{\uparrow}$ hub routes complete metrizable locally convex structure through chapter 1506, chapter 3386, and chapter 205.

Chapter 3495

A Concrete Naming Certificate for $\mathsf{FrechetSpace}^{\uparrow}$

Following Frechet and Bourbaki on complete metrizable locally convex spaces, the $\mathsf{FrechetSpace}^{\uparrow}$ packet records the finite BEDC surface where a topological vector space, a locally convex neighbourhood ledger, a displayed compatible metric, and a completion-facing row are read as one carrier. It sits beside the metric-completion surface of chapter 1506, the locally convex source of chapter 3386, and the topological-vector-space source of chapter 205. The chapter names only the carrier rows needed for a Frechet-space read; it does not import a host complete locally convex space, a selected metric, quotient completion equality, countable choice, or an infinite seminorm enumeration.

Every complete-metrizable $\mathsf{FrechetSpace}^{\uparrow}$ read consumes the $\mathsf{TopVecSpace}^{\uparrow}$ operation-continuity surface of chapter 205, the $\mathsf{LocallyConvexSpace}^{\uparrow}$ seminorm and neighbourhood ledger of chapter 3386, and the $\mathsf{MetricCompletion}^{\uparrow}$ completion handoff of chapter 1506 through the displayed rows $V, L, M, C, Q, H, R, P, N$ before a functional-analysis consumer can replay completeness or local convexity.

Definition 3495.1 (Frechet space carrier)**.** *A* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, C, Q, H, R, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{TopVecSpace}^{\uparrow}$ *source row, exposing the vector carrier, topology, addition-continuity ledger, and scalar-continuity ledger.*
2. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *row, recording the finite seminorm and convex balanced neighbourhood evidence consumed by local convexity reads.*
3. *M is the compatible* $\mathsf{Met}^{\uparrow}$ *row, recording the finite distance witnesses and the topology-compatibility comparison used by metrizable consumers.*
4. *C is the completeness ledger, read through Cauchy windows, Cauchy-filter or Cauchy-net evidence, and displayed convergence rows without selecting a completed point.*
5. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, used only to expose the completion-facing boundary of the compatible metric.*
6. *H records componentwise* hsame *transport for V, L, M, C, Q and for the replay, provenance, and local naming rows.*

7. *R records displayed* Cont *replay through the topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion handoff, transport, provenance, and local naming rows.*

8. *P is the* Pkg *provenance over* $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, L, M, C, Q, H, R, P, N$. It has no coordinate for a host Frechet space, an external complete metric, a selected translation-invariant metric, quotient completion equality, countable choice, or an infinite family of seminorms.

Theorem 3495.2 (Frechet space NameCert obligations)**.** *For every accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier $F = (V, L, M, C, Q, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ *surface consists exactly of topological-vector source exposure through V; locally convex source exposure through L; compatible metric exposure through M; completeness evidence through C; metric-completion handoff through Q; componentwise transport through H; displayed replay through R; provenance through P; and local naming through N.*

Proof. Project the accepted carrier of Definition 3495.1. Its mathematical rows are the topological-vector source V, the locally convex row L, the compatible metric row M, the completeness ledger C, and the metric-completion handoff Q.

The topological and locally convex rows are consumed before the metric row. Thus a Frechet-space consumer first reads vector operations, topology, continuity, seminorm probes, and convex balanced neighbourhood evidence through the already named $\mathsf{TopVecSpace}^{\uparrow}$ and $\mathsf{LocallyConvexSpace}^{\uparrow}$ surfaces.

The compatible metric and completeness rows then give the metrizable complete handoff. The row M supplies only displayed distance and topology-compatibility evidence, while C records Cauchy and convergence checks without adding a selected completion point.

Finally Q, H, R, P, N transport, replay, source, and name the same finite route. Since the carrier contains no host Frechet object, selected metric, quotient-completion equality, countable-choice schedule, or infinite seminorm enumeration, the listed rows exhaust the local NameCert surface. □

Theorem 3495.3 (Frechet space metrizable complete handoff)**.** *Let $F = (V, L, M, C, Q, H, R, P, N)$ be an accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *carrier. Every downstream functional-analysis, completion, or locally convex consumer that asks for metrizability and completeness must factor through*

$$V, L, M, C, Q \longmapsto H, R, P, N.$$

The handoff exposes the topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion boundary, transport, replay, provenance, and local naming, and it exports no host complete locally convex space, selected compatible metric, quotient completion, countable-choice row, or infinite seminorm-family object.

Proof. Start with a downstream request for a Frechet-space read. The vector and topology data are not independent host structure; they are obtained from the $\mathsf{TopVecSpace}^{\uparrow}$ row V.

The locally convex evidence is read next through L. This forces finite seminorm probes and convex balanced neighbourhood rows to remain visible before any metric or completion claim is consumed.

The metric and completeness portion then passes through M, C, Q. The compatible metric row M supplies the displayed distance and topology comparison, C supplies the finite completeness ledger, and Q exposes only the $\mathsf{MetricCompletion}^{\uparrow}$ handoff boundary.

The structural rows H, R, P, N preserve that same route under transport, replay, provenance, and local naming. Therefore a consumer receives a metrizable complete locally convex carrier only as the displayed finite packet above, not as a host Frechet space, selected metric, quotient completion, choice principle, or infinite seminorm family. □

Falsifiable prediction: Any accepted $\mathsf{FrechetSpace}^{\uparrow}$ consumer that cites complete metrizability without first reading $V, L, M, C, Q, H, R, P, N$ refutes the carrier surface.

Independence witness: `metriccompletion`, `locally_convex_space`, `topvecspace`: no listed sibling is bijective with $\mathsf{FrechetSpace}^{\uparrow}$, because this carrier alone binds topological-vector data, locally convex seminorm and neighbourhood rows, a compatible metric, completeness evidence, metric-completion handoff, transport, replay, provenance, and local NameCert rows into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetSpace}^{\uparrow}$ packet is generated from a topological-vector source, a locally convex source, a compatible metric row, a completeness ledger, a metric-completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3495.1, Definition 3495.2, and Definition 3495.3 seed the finite Frechet-space carrier over $V, L, M, C, Q, H, R, P, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, barrelledness, Baire category, nuclearity, selected compatible metrics, quotient completions, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topological-vector source, locally convex source, compatible metric, completeness ledger, metric-completion handoff, transport, replay, provenance, and local naming rows for BEDC.Derived.FrechetSpaceUp.

After the complete metric row of Definition 3495.1 has been displayed, any $\mathsf{FrechetSpace}^{\uparrow}$ consumer that needs a Baire-category conclusion must leave the Frechet carrier through the theorem route of chapter 3570 and the inspected complete-metric surface of chapter 3535. The Frechet packet supplies local convexity, topological-vector continuity, compatible metric data, and completion-facing rows; the dense-open theorem route is a separate sibling surface.

Theorem 3495.4 (Frechet Baire dependency route)**.** *Every accepted* $\mathsf{FrechetSpace}^{\uparrow}$ *read that asks for Baire-category evidence factors first through the complete metric row* C *and the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff* Q *of Definition 3495.1, and then through the* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *and* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *chapters cited above. The route adds no Hahn–Banach separation, selected generic point, countable-choice schedule, host intersection object, or ambient complete locally convex space to the Frechet carrier.*

Proof. Project the Frechet carrier as $F = (V, L, M, C, Q, H, R, P, N)$. The rows V, L, M, C, Q expose topological-vector structure, local convexity, compatible metric data, completeness evidence, and the metric-completion boundary before any theorem consumer reads the packet.

A Baire-category request is not one of those Frechet rows. Route it through chapter 3535 for the inspected complete-metric Baire surface and through chapter 3570 for the theorem-level dense-open package.

The remaining rows H, R, P, N only transport, replay, source, and name the displayed route. Since no row stores Hahn–Banach separation, a selected generic point, a countable-choice schedule,

a host intersection object, or an ambient complete locally convex space, none can be exported by the dependency route. □

Theorem 3495.5 (Frechet Baire-category completeness route)**.** *Every* $\mathsf{FrechetSpace}^{\uparrow}$ *consumer that needs dense-open completeness must expose the full carrier*

$$F = (V, L, M, C, Q, H, R, P, N)$$

and then pass through chapter 3570 and chapter 3535. The Frechet chapter contributes the displayed vector, local-convex, compatible-metric, complete-metric, completion-handoff, transport, replay, provenance, and local naming rows; the dense-open conclusion remains a theorem-level sibling route rather than a hidden field of F*.*

Proof. Project the accepted carrier as $F = (V, L, M, C, Q, H, R, P, N)$. The rows V, L, M, C, Q expose the topological-vector, local-convex, compatible-metric, complete-metric, and completion-handoff data needed before a dense-open consumer can even form its request.

Now apply the theorem route. The complete-metric surface is supplied by chapter 3535, while the dense-open theorem statement is supplied by chapter 3570. These two sibling chapters are therefore the only exits from the Frechet carrier toward Baire-category completeness.

The remaining rows H, R, P, N preserve and name the route. Since they contain only componentwise transport, replay, provenance, and the local $\mathsf{NameCert}_{\mathsf{FrechetSpace}^{\uparrow}}$ row, they cannot introduce a selected generic point, a countable-choice schedule, or an ambient complete locally convex space outside the displayed dependency route. □

Every $\mathsf{FrechetSpace}^{\uparrow}$ consumer reads the locally convex linear-topology surface of chapter 3386 before it treats the compatible metric as complete. The route then passes through the metric-completion handoff of chapter 1506, so Baire-category and metric-completion readers receive the displayed locally convex, complete-metric, transport, replay, provenance, and local naming rows rather than a host Frechet space or an unbounded seminorm schedule. Every $\mathsf{FrechetSpace}^{\uparrow}$ read first consumes the topological vector-space carrier of chapter 205 and the locally convex carrier of chapter 3386. It then reads the compatible metric row M, the completeness ledger C, and the $\mathsf{MetricCompletion}^{\uparrow}$ handoff row Q supplied by the metric-completion route of chapter 1506. The handoff is only the displayed carrier route needed to replay a complete locally convex packet through the local NameCert surface; it exports no selected compatible metric, quotient completion, countable-choice schedule, or infinite seminorm family.

Chapter 3496

A Concrete Naming Certificate for $\mathsf{FinitePartitionMesh}^{\uparrow}$

Following Bishop and Bridges on constructive partitions of real intervals, the $\mathsf{FinitePartitionMesh}^{\uparrow}$ packet records the finite dyadic mesh surface used before interval estimates may pass to Darboux or regular-completion consumers. It sits between the real endpoint seal of chapter 38, the dyadic tolerance rows of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the closed bounded interval Darboux partition route of chapter 3429, and the finite Darboux-sum boundary of chapter 1880. The packet names only displayed endpoint rows, dyadic breakpoints, mesh adjacency, refinement witnesses, and downstream handoff rows; it is not a selected tagged partition, an open-cover compactness theorem, a quotient equality of functions, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 3496.1 (Finite partition mesh carrier)**.** *A* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, Q, M, R, B, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *closed-interval endpoint row, exposing the lower endpoint, upper endpoint, endpoint order, and terminal real seal used by the mesh.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *breakpoint ledger naming the dyadic cut points and the dyadic mesh bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family in which endpoint and breakpoint reads are inspected.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed endpoints and dyadic breakpoints.*
5. *M is the finite mesh row listing the cells, endpoint adjacency, left-to-right order, and boundary inclusion witnesses.*
6. *R is the displayed common-refinement row comparing M with any accepted finite dyadic submesh over the same interval.*
7. *B is the* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *handoff row reached only after I, D, W, Q, M, R have been displayed.*
8. *S is the* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer row for finite lower and upper sum reads over the cells of M or its displayed refinement.*

9. *H records componentwise* hsame *transport of interval endpoints, dyadic breakpoints, windows, readbacks, mesh cells, refinement witnesses, Darboux handoff, sum rows, provenance, and local naming.*

10. *C records displayed* Cont *replay from endpoint rows to dyadic breakpoints, from breakpoints to mesh cells, and from mesh refinement to Darboux partition and sum handoffs.*

11. *P is the* Pkg *provenance over* $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite partition, a selected tag family, quotient function equality, measure completion, open-cover compactness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the displayed endpoint seal.*

Lean4 ✓ `BEDC.Derived.FinitePartitionMeshUp.FinitePartitionMeshTasteGate single carrier alignment decode encode`

Theorem 3496.2 (Finite partition mesh refinement obligations)**.** *For the carrier of Definition 3496.1, the local* $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (I, D, W, Q, M, R, B, S, H,$ interval endpoint exposure through I; dyadic breakpoint and mesh-bound exposure through D; finite observation windows through W; regular rational readback through Q; mesh-cell exposure through M; common-refinement evidence through R; closed bounded interval Darboux partition handoff through B; finite Darboux-sum handoff through S; componentwise transport through H; displayed continuation replay through C; provenance through P; and local naming through N.*

Every accepted refinement request must therefore route a $\mathsf{Real}^{\uparrow}$ *interval consumer, a* ClosedBoundedIntervalDarbo *consumer, or a* $\mathsf{DarbouxSum}^{\uparrow}$ *consumer through the finite path*

$$I \longmapsto D \longmapsto W \longmapsto Q \longmapsto M \longmapsto R \longmapsto B \longmapsto S \longmapsto (H, C, P, N)$$

before any lower-sum, upper-sum, or interval estimate can be read. The route exports no selected tags, quotient stream, host partition predicate, countable-choice schedule, open-cover compactness row, measure-completion object, or ambient real-completeness theorem.

Proof. Project the displayed packet. The interval row I, dyadic ledger D, finite windows W, regular readback Q, and mesh row M are all finite certificate rows, so the carrier exposes a bounded list of cells before a refinement witness can be inspected.

Read the refinement row next. Since R compares mesh cells only through the already displayed endpoint, breakpoint, window, and readback rows, every common refinement remains a finite dyadic submesh over the same interval boundary.

Finally inspect the Darboux handoff rows. The rows B and S are downstream of the finite mesh and its refinement witness, while H, C, P, N only transport, replay, source, and name that route. The excluded tag, quotient, choice, compactness, measure, and ambient-completeness objects have no carrier coordinate, so the listed rows exhaust the refinement obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$.

Constructive story: A $\mathsf{FinitePartitionMesh}^{\uparrow}$ packet is generated from RealUp interval endpoint rows, DyadicRatCoreUp breakpoint rows, StreamNameUp finite windows, RegSeqRatUp readbacks, finite mesh cells, a displayed common-refinement row, ClosedBoundedIntervalDarbouxPartitionUp and DarbouxSumUp handoff rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}$, seed)

Scope closed: Definition 3496.1 and Definition 3496.2 seed the finite dyadic partition-mesh route over $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, and $\mathsf{DarbouxSum}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FinitePartitionMesh}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected tags, quotient stream equality, host partition predicates, countable choice, open-cover compactness, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, breakpoint exactness, finite mesh refinement, Darboux partition handoff, Darboux-sum handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.FinitePartitionMeshU

Chapter 3497

A Concrete Naming Certificate for ChoiceWindowSequence$^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on choice sequences, the ChoiceWindowSequence$^{\uparrow}$ packet isolates the finite inspected windows by which a choice sequence becomes available to an L10 StreamName$^{\uparrow}$ consumer. It refines the finite-prefix surface of chapter 1482, keeps the lawless-sequence boundary of chapter 1495 visible, and reads Baire-style prefix data from chapter 334. The packet uses Nat$^{\uparrow}$ positions and answer rows, Bool$^{\uparrow}$ presence rows, and the StreamName$^{\uparrow}$ hand-off only as displayed finite data; it does not import countable choice, a completed host sequence, quotient stream equality, or a global law selecting all future answers.

Definition 3497.1 (ChoiceWindowSequence finite-window carrier)**.** *A* ChoiceWindowSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (Q, L, B, S, I, A, O, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* ChoiceSequence$^{\uparrow}$ *prefix packet whose currently available answers may be inspected;*
2. *L is the sibling* LawlessSequence$^{\uparrow}$ *boundary row, recording that the packet reads only displayed answers and no governing law for future positions;*
3. *B is the* BaireSpace$^{\uparrow}$ *finite-prefix context in which the window is interpreted;*
4. *S is the* StreamName$^{\uparrow}$ *schedule naming the finite positions requested by the consumer;*
5. *I is the* Nat$^{\uparrow}$ *index ledger listing the positions of the inspected window;*
6. *A is the* Nat$^{\uparrow}$ *answer ledger attached to the positions in I;*
7. *O is the* Bool$^{\uparrow}$ *presence classifier for each requested position, distinguishing displayed answers from absent future data;*
8. *H records componentwise* hsame *transport for Q, L, B, S, I, A, O and the local naming row;*
9. *C records displayed* Cont *replay of the same finite-window read;*
10. *P is the* Pkg *provenance over* ChoiceWindowSequence$^{\uparrow}$*,* ChoiceSequence$^{\uparrow}$*,* LawlessSequence$^{\uparrow}$*,* BaireSpace$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* Nat$^{\uparrow}$*,* Bool$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^\uparrow}$ *row.*
The carrier accepts only the finite window displayed by S, I, A, O. It exports no completed function $\mathsf{Nat}^\uparrow \to \mathsf{Nat}^\uparrow$*, no countable-choice selector, no quotient infinite stream, no hidden tail oracle, and no law determining positions outside the displayed window.*

Lean4 ✓ `BEDC.Derived.ChoiceWindowSequenceUp.ChoiceWindowSequenceTasteGate single carrier alignment`

Theorem 3497.2 (ChoiceWindowSequence carrier admission obligations)**.** *For every accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *carrier $W = (Q, L, B, S, I, A, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^\uparrow}$ *surface consists exactly of carrier admission for the source* $\mathsf{ChoiceSequence}^\uparrow$ *row, the lawless-sequence boundary row, the* $\mathsf{BaireSpace}^\uparrow$ *prefix context, the finite* $\mathsf{StreamName}^\uparrow$ *schedule, the* $\mathsf{Nat}^\uparrow$ *index and answer ledgers, the* $\mathsf{Bool}^\uparrow$ *presence classifier, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Project the carrier of Definition 3497.1. The visible coordinates are precisely $Q, L, B, S, I, A, O, H, C, P, N$, so carrier admission reads no object outside the displayed finite packet.

The source rows Q, L, B identify the traditional choice-sequence, lawless-sequence, and Baire-prefix context. The inspection rows S, I, A, O then name the finite schedule, the requested positions, the displayed answers, and the presence decisions.

The structural rows H, C, P, N transport, replay, source, and name the same finite window. Since the carrier has no coordinate for a total host sequence, countable-choice selector, quotient stream equality, hidden tail oracle, or global law, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3497.3 (ChoiceWindowSequence window classifier stability)**.** *Let $W = (Q, L, B, S, I, A, O, H, C, P, N)$ and $W' = (Q', L', B', S', I', A', O', H', C', P', N')$ be accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *carriers. If the displayed* $\mathsf{StreamName}^\uparrow$ *schedules,* $\mathsf{Nat}^\uparrow$ *index ledgers,* $\mathsf{Nat}^\uparrow$ *answer ledgers, and* $\mathsf{Bool}^\uparrow$ *presence classifiers agree componentwise under the available* hsame *rows, then every public finite-window read classifies W and W' the same way. The classifier is insensitive to tail values or laws outside the displayed window.*

Proof. Begin with the displayed schedules and ledgers. Agreement of S, I, A, O and S', I', A', O' gives the same requested positions, the same visible answer rows, and the same present-or-absent decisions for every finite request a consumer may replay.

The source context is read only through those windows. The $\mathsf{ChoiceSequence}^\uparrow$ and $\mathsf{BaireSpace}^\uparrow$ rows supply finite-prefix data, while the $\mathsf{LawlessSequence}^\uparrow$ row records that no law for undisplayed positions is part of the packet.

Finally transport the read through H, H' and replay it through C, C'. Since public reads factor through the displayed window and the carrier contains no tail oracle, completed function, quotient stream, or countable-choice selector, no classifier distinction can be obtained outside the agreed finite data. □

Theorem 3497.4 (ChoiceWindowSequence StreamName handoff)**.** *Every* $\mathsf{StreamName}^\uparrow$ *consumer of an accepted* $\mathsf{ChoiceWindowSequence}^\uparrow$ *packet factors through the finite route*

$$\mathsf{ChoiceSequence}^\uparrow,\ \mathsf{LawlessSequence}^\uparrow,\ \mathsf{BaireSpace}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow,\ \mathsf{Nat}^\uparrow,\ \mathsf{Bool}^\uparrow \longmapsto \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Na}$$

The handoff supplies a finite inspected choice-sequence window to the L10 StreamName-facing interface and supplies no countable choice, total host sequence, quotient stream equality, selected future value, or law for undisplayed positions.

Proof. Read the source side first. The consumer receives a choice-sequence prefix row, a lawless-sequence boundary row, and a Baire-prefix context, all as finite displayed packets.

Next open the StreamName-facing window. The schedule row S selects the inspected positions, the $\mathsf{Nat}^{\uparrow}$ rows I, A list the requested indices and displayed answers, and the $\mathsf{Bool}^{\uparrow}$ row O records which requested positions are currently present.

Finally pass through H, C, P, N. Componentwise transport and continuation replay preserve the same finite window, while provenance and local naming attach the packet to the BEDC kernel rows. No excluded global choice object is introduced by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ChoiceWindowSequence}^{\uparrow}$ packet is generated from a ChoiceSequence prefix row, LawlessSequence boundary row, BaireSpace finite-prefix context, StreamName finite schedule, NatUp index and answer ledgers, BoolUp presence classifiers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3497.1, Definition 3497.2, Definition 3497.3, and Definition 3497.4 seed the finite inspected-window route for choice-sequence data consumed by StreamName-facing L10 packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceWindowSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, a total host sequence, quotient stream equality, selected future values, a global law for undisplayed positions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window classifier stability, StreamName handoff, transport, continuation, provenance, and non-escape proofs for BEDC.Derived.ChoiceWindowSequenceUp.

Chapter 3498

A Concrete Naming Certificate for $\mathsf{ProximinalSet}^{\uparrow}$

Following Bishop and Bridges on located sets in metric spaces, a proximinal set is read here as a located subset whose displayed distance data may be accompanied by finite distance-attainment witnesses. The $\mathsf{ProximinalSet}^{\uparrow}$ packet sits between the located-subset interface of chapter 1541, the located-metric handoff of chapter 1533, the real distance seal of chapter 38, and the Hilbert or convex projection specialization of chapter 2356. It records displayed distance-attainment witness rows and deliberately does not export a host nearest-point selector, quotient point equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3498.1 (Proximinal set carrier)**.** *A* $\mathsf{ProximinalSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, L, D, W, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the parent* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance witness, metric-law ledger, and finite observation histories.*
2. *L is the* $\mathsf{LocatedSet}^{\uparrow}$ *row. It supplies the displayed subspace admission, located-distance ledger, rational comparison windows,* $\mathsf{Real}^{\uparrow}$ *distance seal, and* $\mathsf{LocatedMetric}^{\uparrow}$ *handoff already exposed by the located-set packet.*
3. *D is the located-metric distance-read row. It records that the ambient point, admitted set row, and requested tolerance are read through the* $\mathsf{LocatedMetric}^{\uparrow}$ *and* $\mathsf{LocatedSet}^{\uparrow}$ *interfaces before any attainment evidence is inspected.*
4. *W is the distance-attainment witness row. For each displayed request it stores a finite admitted point row and a finite equality-of-distance certificate tying that point to the real distance name supplied by L and D.*
5. *E is the real attainment seal. It records the* $\mathsf{Real}^{\uparrow}$ *comparison between the displayed distance name and the displayed witness point distance, without adding trichotomy, quotient point equality, or ambient real completeness.*
6. *H stores componentwise* hsame *transport for the parent metric row, located-set row, located-metric distance read, attainment witness row, real seal, continuation replay, provenance, and local naming row.*

7. *C stores displayed* Cont *replay from ambient point observations into located-set distance data, from located-set distance data into D, from D into W, and from the witness row into E.*
8. *P is the* Pkg *provenance over* $\mathsf{ProximinalSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$ *row.*

The classifier compares packets only through $X, L, D, W, E, H, C, P, N$. It has no coordinate for a host nearest-point function, quotient equality of ambient points, trichotomy, open-cover compactness, a Hilbert projection theorem, or full ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ProximinalSetUp.ProximinalSetUp`

Theorem 3498.2 (Proximinal set NameCert obligations)**.** *For an accepted* $\mathsf{ProximinalSet}^{\uparrow}$ *carrier $Z = (X, L, D, W, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$ *obligations are exactly carrier admission, parent metric compatibility, located-set distance compatibility, located-metric read ordering, displayed distance-attainment witness stability, real attainment seal readback, and structural non-escape. These obligations are discharged by the finite route*

$$X \longmapsto L \longmapsto D \longmapsto W \longmapsto E$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.ProximinalSetUp.ProximinalSetTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3498.1. The tuple lists the parent metric row, located-set row, located-metric distance read, attainment witness row, real seal, and structural rows.

Parent metric compatibility and located-set distance compatibility are read first. The consumer sees ambient points through X and sees set distance data through L, so the packet inherits the nearest-point refusal boundary of the located-set interface before any attainment row can be consumed.

The row D fixes the located-metric read order. It prevents an attainment witness from being read as a primitive selector by forcing the request through the located-set and located-metric distance route.

The witness row W then stores only displayed request-specific evidence: an admitted point row and a finite equality-of-distance certificate. The real seal E compares that displayed point distance with the real distance name already read from L and D.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for a host nearest-point function, quotient point equality, trichotomy, open-cover compactness, Hilbert projection, or ambient real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 3498.3 (Proximinal set distance-witness route)**.** *Every accepted* $\mathsf{ProximinalSet}^{\uparrow}$ *packet exposes distance-attainment data only through the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{LocatedSet}^{\uparrow} \longmapsto \mathsf{LocatedMetric}^{\uparrow} \longmapsto W \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a $\mathsf{MetricProjection}^{\uparrow}$ *or compactness consumer may read a displayed witness row, but it must supply any host nearest-point selector, quotient point equality principle, trichotomy branch, open-cover compactness argument, or ambient real-completeness principle outside the packet.*

Proof. Start with Definition 3498.1. The only analytic rows available before the witness are the parent metric row X, the located-set row L, and the located-metric distance read D.

Apply Definition 1541.3 and Definition 1541.4. These force located-set distance data through parent metric observations, subspace admission, located-distance ledgers, rational windows, and a $\mathsf{Real}^{\uparrow}$ seal before a consumer can read the distance as real data.

The $\mathsf{ProximinalSet}^{\uparrow}$ carrier then adds W only as a displayed witness row for that already-read distance. It does not turn the row into a host function from ambient points to nearest points, because the witness is indexed by the finite request already present in the packet.

The terminal seal E records the equality-of-distance readback for the displayed witness. The structural rows H, C, P, N preserve the same route, so downstream projection or compactness consumers receive a finite distance-attainment witness boundary rather than quotient point equality, trichotomy, open-cover compactness, Hilbert projection, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{ProximinalSet}^{\uparrow}$ read from an accepted packet factors through the parent $\mathsf{Met}^{\uparrow}$ row, $\mathsf{LocatedSet}^{\uparrow}$ distance interface, $\mathsf{LocatedMetric}^{\uparrow}$ handoff, displayed distance-attainment witness row, $\mathsf{Real}^{\uparrow}$ attainment seal, componentwise transport, continuation replay, provenance, and local NameCert before any projection, compactness, or completion consumer reads the route.

Independence witness: locatedset, locatedmetric, metricprojection, real: $\mathsf{ProximinalSet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries displayed located-set distance data and request-indexed distance-attainment witness rows while refusing host nearest-point selection, quotient point equality, trichotomy, open-cover compactness, and ambient real completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$.

Constructive story: A $\mathsf{ProximinalSet}^{\uparrow}$ packet starts from a parent Metric row, reads a LocatedSet distance interface, routes the distance through a LocatedMetric handoff, records displayed request-indexed distance-attainment witnesses, seals the equality of distance in a RealUp row, and preserves the same route through componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$, seed)

Scope closed: Definition 3498.1, Definition 3498.2, and Definition 3498.3 seed the finite proximinal-set interface through located-set distance data, located-metric reads, displayed attainment witnesses, and real distance seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProximinalSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host nearest-point selection, quotient point equality, trichotomy, open-cover compactness, Hilbert projection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parent metric compatibility, located-set distance compatibility, located-metric read ordering, displayed distance-attainment witness stability, RealUp attainment seal readback, structural non-escape, Nontrivial ProximinalSetUp, and FieldFaithful ProximinalSetUp.

Chapter 3499

A Concrete Naming Certificate for $\mathsf{CompletionUnitComparison}^{\uparrow}$

Following Bourbaki and Bishop-style completion theory, the $\mathsf{CompletionUnitComparison}^{\uparrow}$ packet records the finite route by which the Cauchy-completion unit is compared with the naturality and triangle-identity certificates that consume it. It sits after the unit packet of chapter 1527, the triangle surface of chapter 2256, and the naturality comparison of chapter 2102. The chapter names a displayed comparison object for unit-facing completion reads; it does not import quotient completions, selected representatives, host equality of completed objects, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3499.1 (Completion unit comparison carrier)**.** *A* $\mathsf{CompletionUnitComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, N, T, S, R, H, C, P, L)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row exposing the source-to-completion unit used by the comparison.*
2. *N is the* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *row recording naturality compatibility for the same displayed unit.*
3. *T is the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *row recording triangle-identity compatibility for the same unit-facing route.*
4. *S is the source-and-target ledger over the finite stream, regular-rational, dyadic-tolerance, and terminal Real rows read by the unit comparison.*
5. *R is the refusal row excluding quotient completion equality, selected representatives, host equality of completed carriers, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
6. *H records componentwise* hsame *transport for unit, naturality, triangle, source-target, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the unit row through naturality compatibility and triangle compatibility before any consumer reads the comparison.*
8. *P is the* Pkg *provenance over* $\mathsf{CompletionUnitComparison}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}$ $\mathsf{CompletionUnitTriangle}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$ *row.*

The carrier compares only displayed completion-unit routes. It has no coordinate for host quotient completions, representative selection, equality of completed objects, choice principles, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionUnitComparisonUp.CompletionUnitComparisonTasteGate single carrier alignment`

Theorem 3499.2 (Completion unit comparison NameCert obligations)**.** *For every accepted* CompletionUnitCompar *carrier*

$$U = (Q, N, T, S, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *unit-comparison exposure, requiring every comparison consumer to cite the displayed* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row Q;*
2. *naturality compatibility, requiring the comparison to pass through the* $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$ *row N for the same source and target data;*
3. *triangle compatibility, requiring the comparison to pass through the* $\mathsf{CompletionUnitTriangle}^{\uparrow}$ *row T before any completion-facing return is exported;*
4. *transport and provenance stability, requiring H, C, P, L to transport, replay, source, and name only the rows Q, N, T, S, R;*
5. *refusal of host quotient completion, requiring the row R to exclude quotient equality, selected representatives, host equality of completed carriers, countable-choice schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No unit-comparison obligation is stored outside these rows.

Proof. Project the accepted carrier. The comparison-facing rows are $Q, N, T, S, R, H, C, P, L$.

The unit row Q supplies the only source-to-completion entry. Naturality compatibility must therefore read the same source and target data through N, and triangle compatibility must read the same unit-facing route through T.

The transport, replay, provenance, and local naming rows H, C, P, L preserve exactly this displayed route. Since the refusal row R contains the only host-side exclusions and no quotient-completion coordinate appears in the carrier, the five listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CompletionUnitComparison}^{\uparrow}$ consumer factors through displayed $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionComparisonNaturality}^{\uparrow}$, $\mathsf{CompletionUnitTriangle}^{\uparrow}$, source-target ledger, refusal, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: cauchycompletionunit, cauchy-completion-comparison-naturality, completion-unit-triangle: the unit packet records source admission without the comparison route, the naturality packet records functorial compatibility without the triangle return, and the triangle packet records return compatibility without the naturality comparison. None is bijective to $\mathsf{CompletionUnitComparison}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompletionUnitComparison}^{\uparrow}$ packet is generated from a CauchyCompletionUnit row, CauchyCompletionComparisonNaturality row, CompletionUnitTriangle row, source-target ledger, refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompletionUnitComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3499.1 and Definition 3499.2 seed the finite completion-unit comparison route over unit, naturality, triangle, transport, provenance, and refusal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompletionUnitComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host equality of completed carriers, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked unit-comparison exposure, naturality compatibility, triangle compatibility, transport and provenance stability, and refusal exactness for BEDC.Derived.CompletionUnitComparisonUp.

Chapter 3500

A Concrete Naming Certificate for PoincareBoundaryFixedTransport$^{\uparrow}$

The PoincareBoundaryFixedTransport$^{\uparrow}$ packet realizes the boundary-fixed disk transport reading of Definition 9463.10 as finite disk-map, boundary, hyperbolic-metric, reversible-transport, provenance, refusal, and local naming rows. It sits after the disk-coordinate surface of chapter 377, the disk-phase transport packet of chapter 3455, the metric-read surface of chapter 3474, and the geodesic far-end boundary of chapter 3491. The chapter names a finite boundary-preserving transport object; it does not turn the Poincare disk into a host plane, identify ideal boundary points by quotient equality, choose a global geodesic representative, or assert physical observer motion.

Definition 3500.1 (Poincare boundary-fixed transport carrier)**.** *A* PoincareBoundaryFixedTransport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, M, T, P, R, H, C, A, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-map row, exposing the finite source and target disk-coordinate observations.*
2. *F is the boundary far-end row, recording the displayed* HyperbolicGeodesicBoundary$^{\uparrow}$ *far-end that must remain fixed by the transport.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same disk-map data.*
4. *T is the reversible* PoincareDiskPhaseTransport$^{\uparrow}$ *row carrying the phase-transport route between the displayed disk observations.*
5. *P is the provenance row tying D, F, M, T to the hyperbolic phase vision and to the sibling UnitDisk, HyperbolicMetric, PoincareDiskPhaseTransport, and HyperbolicGeodesicBoundary packets.*
6. *R is the refusal row excluding host disk automorphisms, quotient ideal-boundary equality, selected geodesic representatives, physical observer motion, and global cosmological dynamics.*
7. *H records componentwise* hsame *transport for disk-map, boundary far-end, metric-read, reversible-transport, provenance, refusal, replay, and local naming rows.*
8. *C records displayed* Cont *replay through disk-map admission, boundary far-end preservation, metric-read compatibility, reversible transport, provenance, and refusal.*

9. *A records the local transport audit row: every accepted replay must preserve the displayed boundary far-end while reading the same disk-map and metric rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$ *row.*

The carrier exports boundary-fixed disk transport only through $D, F, M, T, P, R, H, C, A, N$. It has no coordinate for a host-level disk automorphism, quotient boundary point, selected infinite geodesic, physical observer trajectory, or completed hyperbolic plane.

Lean4 (definition) `BEDC.Derived.PoincareBoundaryFixedTransportUp.PoincareBoundaryFixedTransportUp`
Lean4 ✓ `BEDC.Derived.PoincareBoundaryFixedTransportUp.PoincareBoundaryFixedTransportTasteGate`
`single carrier alignment`

Theorem 3500.2 (Poincare boundary-fixed transport NameCert obligations)**.** *For every accepted* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *carrier*

$$B = (D, F, M, T, P, R, H, C, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$ *surface consists exactly of disk-map admission, boundary far-end preservation, hyperbolic-metric read compatibility, reversible Poincare-disk transport, provenance, refusal exactness, componentwise* hsame *transport, displayed* Cont *replay, local transport audit, and local naming.*

Every accepted transport preserving the boundary far-end must factor through

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, BHist, hsame, Cont, Pk

Proof. Project the accepted carrier. Disk-map admission is row D, the fixed far-end is row F, the metric read is row M, and the reversible disk transport is row T.

The provenance and refusal rows P, R bind these four rows to the sibling packets and exclude host-level automorphisms, quotient boundary equality, selected geodesics, physical observer motion, and global cosmology.

Transport through H and replay through C preserve the same displayed route. The audit row A checks that the far-end row F is the same before and after the transport, and N names the resulting packet. No further row remains, so the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ transport preserving the boundary far-end factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, provenance, refusal, hsame, Cont, and local NameCert rows.

Independence witness: unitdisk, poincare-disk-phase-transport, hyperbolic-metric, hyperbolic-geodesic-boundary: the UnitDisk packet records disk coordinates without reversible transport or fixed far-end data, the Poincare disk phase-transport packet records reversible transport without fixed-boundary audit, the HyperbolicMetric packet records metric reads without transport preservation, and the HyperbolicGeodesicBoundary packet records far-end incidence without a disk-map transport replay. None is bijective to $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ packet is generated from UnitDisk disk-map rows, HyperbolicGeodesicBoundary far-end rows, HyperbolicMetric metric reads, PoincareDiskPhaseTransport reversible transport, provenance, refusal exactness, componentwise hsame transport, displayed Cont replay, local transport audit, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}}$, seed)

Scope closed: Definition 3500.1 and Definition 3500.2 seed the finite boundary-fixed disk transport route over disk-map, boundary, metric, transport, provenance, refusal, replay, audit, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareBoundaryFixedTransport}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host disk automorphisms, quotient boundary equality, selected geodesic representatives, physical observer motion, global cosmological dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, boundary far-end preservation, metric-read compatibility, reversible transport, provenance, refusal exactness, transport, replay, audit, and local naming for BEDC.Derived.PoincareBoundaryFixedTransportUp.

Chapter 3501

A Concrete Naming Certificate for $\mathsf{MetricCompletionNaturality}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion and uniform continuity, the $\mathsf{MetricCompletionNaturality}^{\uparrow}$ packet records the finite L10 square by which a uniformly continuous source map is read by completion-facing consumers. It sits over the metric-completion surface of chapter 1506, the map carrier of chapter 130, the extension route of chapter 769, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names only the displayed naturality packet; it is not a quotient completion construction, not a selected representative principle, not host function equality, not countable choice, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3501.1 (Metric completion naturality carrier)**.** *A* $\mathsf{MetricCompletionNaturality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L)$$

with the following displayed rows:

1. M_X *is the source* $\mathsf{MetricCompletion}^{\uparrow}$ *row, exposing the finite Cauchy-window requests inspected before the map is applied.*
2. M_Y *is the target* $\mathsf{MetricCompletion}^{\uparrow}$ *row, exposing the completion-facing consumer reached after the map and extension rows have been displayed.*
3. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the displayed source map and the finite continuity data it exposes to the packet.*
4. E *is the* $\mathsf{UniformContinuousExtension}^{\uparrow}$ *row that transports the same displayed map through the completion interface.*
5. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the source completion, target completion, and extension reads.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which numerical tolerance requests are answered.*
7. A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream window and regular readback rows have been fixed.*
8. H *records componentwise* hsame *transport for the source completion, target completion, map, extension, stream-window, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the naturality route*

$$M_X \longmapsto F, E \longmapsto S, R, A \longmapsto M_Y.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionNaturality}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M_X, M_Y, F, E, S, R, A, H, C, P, L$. It has no coordinate for quotient completion equality, selected representatives, host function equality, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.MetricCompletionNaturalityUp.MetricCompletionNaturalityTasteGate single carrier alignment`

Theorem 3501.2 (Metric completion naturality NameCert obligations)**.** *For every accepted* MetricCompletionNatu *carrier*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source completion, target completion, continuous-map, uniform-extension, stream-window, regular-readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *source-before-extension exposure, requiring any completion-facing read to inspect M_X and F before the extension row E can be consumed;*
3. *target-consumer routing, requiring M_Y to be reached only after the displayed extension route has passed through S, R, A;*
4. *finite observation exactness, requiring the shared* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows to be the only numerical readback route between the extension row and the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *classifier stability, requiring componentwise* hsame *transport through H and displayed replay through C to preserve the same finite naturality route;*
6. *ledger non-escape, excluding quotient completions, selected representatives, host function equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking from the packet.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3501.1. The accepted packet displays exactly the two completion rows, the map row, the extension row, the finite observation rows, and the structural rows.

Source-before-extension exposure is read from M_X, F, E. The map row F is available only with the source completion observations in M_X, and the extension row E records the completion-facing transport of that same displayed map.

Target-consumer routing is read from M_Y together with S, R, A. The target completion row can be offered to a consumer only after the extension has exposed the finite stream windows, regular readback, and real seal used by the route.

Classifier stability is componentwise transport through H followed by replay through C. Since P and L only source and name the same displayed route, there is no separate row from which quotient completion equality, selected representatives, host function equality, choice, ambient completeness, Lean verification, or bridge credentials could be projected. □

Theorem 3501.3 (Metric completion naturality consumer nonescape)**.** *Every completion-facing or real-seal-facing consumer of an accepted* $\mathsf{MetricCompletionNaturality}^{\uparrow}$ *carrier*

$$N = (M_X, M_Y, F, E, S, R, A, H, C, P, L)$$

must factor through the finite route

$$M_X, F \longmapsto E \longmapsto S, R, A \longmapsto M_Y \longmapsto H, C, P, L.$$

Consequently the consumer receives only the displayed source completion row, continuous-map row, uniform-extension row, shared finite window, regular readback, terminal real seal, target completion row, transport, replay, provenance, and local name. It exports no quotient completion, no selected representative, no host function equality, no countable-choice schedule, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no bridge-checking credential.*

Proof. Start with a completion-facing or real-seal-facing read of the carrier. Projection first exposes the source completion row M_X and the continuous-map row F, so the consumer cannot begin from a hidden completed representative or a host function.

The extension row E is then the only row carrying the displayed map through the completion interface. It does not add a quotient completion object; it only names the finite extension route attached to the same source and map rows.

The read next passes through S, R, A. These rows fix the finite observation windows, the regular rational readback, and the terminal real seal before the target completion row M_Y is available.

Finally H, C, P, L transport, replay, source, and name the same displayed route. Because the carrier has no further coordinates, the stated excluded principles cannot be exported by any consumer of the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionNaturality}^{\uparrow}$ packet is generated from source and target MetricCompletion rows, a ContinuousMap row, a UniformContinuousExtension row, shared StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$, seed)

Scope closed: Definition 3501.1, Definition 3501.2, and Definition 3501.3 seed the finite naturality route from source metric completion through displayed continuous-map extension to target completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionNaturality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-before-extension exposure, target-consumer routing, finite observation exactness, classifier stability, consumer nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.MetricCompletionNaturalityUp.

Chapter 3502

A Concrete Naming Certificate for AsymptoticSequence$^{\uparrow}$

Following Hardy and Littlewood on asymptotic comparison of sequences, the AsymptoticSequence$^{\uparrow}$ packet records finite tail-comparison data for displayed sequence names rather than importing host sequence equality or completed real analysis. It sits over the StreamName$^{\uparrow}$ window surface of chapter 126, the RegSeqRat$^{\uparrow}$ rational readback of chapter 301, the limit-facing handoff of chapter 1413, the symmetric tail comparison of chapter 1738, and the cofinal route comparison of chapter 1255. The packet is a named consumer for Real and Completion tail comparisons: it exposes finite windows, thresholds, readbacks, transport, replay, provenance, and local naming, while refusing quotient streams, arbitrary subsequence choice, and ambient Real$^{\uparrow}$ completeness.

Definition 3502.1 (AsymptoticSequence carrier)**.** *An* AsymptoticSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, W, M, Q, L, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S and T are the two source sequence-name rows, read only through finite* StreamName$^{\uparrow}$ *windows.*
2. *W is the paired* StreamName$^{\uparrow}$ *window ledger naming the inspected source cells and the corresponding comparison windows.*
3. *M is the tail-threshold row recording which displayed unary tail must be reached before the comparison classifier is read.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the rational observations attached to W and M.*
5. *L is the* RealSequenceLimit$^{\uparrow}$ *handoff row available only after the scheduled windows and rational readbacks are displayed.*
6. *E is the* CauchyTailEquivalence$^{\uparrow}$ *provenance row for symmetric regular-Cauchy tail comparison.*
7. *F is the* CofinalTailEquivalence$^{\uparrow}$ *provenance row for cofinal reindexing and tail-route comparison.*
8. *H records componentwise* hsame *transport for the source rows, windows, threshold, readback, limit handoff, tail-equivalence rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each asymptotic-comparison request through $S, T, W, M, Q, L, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* AsymptoticSequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealSequenceLimit$^{\uparrow}$, CauchyTailEquivalence$^{\uparrow}$, CofinalTailEquivalence$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^\uparrow}$ *row.*
The carrier exports no quotient equality of streams, no host asymptotic relation, no selected subsequence, no trichotomy row, and no ambient $\mathsf{Real}^\uparrow$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.AsymptoticSequenceUp.AsymptoticSequenceTasteGate single carrier alignment`

Theorem 3502.2 (AsymptoticSequence NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticSequence}^\uparrow$ *carrier*

$$A = (S, T, W, M, Q, L, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^\uparrow}$ *surface consists exactly of carrier admission for the two source sequence rows, paired* $\mathsf{StreamName}^\uparrow$ *windows, the displayed tail threshold,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{RealSequenceLimit}^\uparrow$ *handoff,* $\mathsf{CauchyTailEquivalence}^\uparrow$ *provenance,* $\mathsf{CofinalTailEquivalence}^\uparrow$ *provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted comparison reads W and M before Q, reads Q before L, E, F, and reaches H, C, P, N only as structural support for the same finite route.*

Proof. Project the carrier of Definition 3502.1. The source rows S, T expose only the finite $\mathsf{StreamName}^\uparrow$ windows stored in W, and the comparison cannot inspect a tail before the threshold row M has been displayed.

The readback row Q is then forced by the $\mathsf{RegSeqRat}^\uparrow$ surface of chapter 301. It consumes the same windows and threshold data rather than a host sequence value.

The analytic rows are downstream of that readback. The limit-facing handoff L is the finite route supplied by chapter 1413, while E and F record the symmetric and cofinal tail provenance of chapter 1738 and chapter 1255.

Finally H, C, P, N transport, replay, source, and name only the displayed route. Since no carrier coordinate contains quotient stream equality, host asymptotic comparison, a selected subsequence, trichotomy, or ambient completeness, none can appear as a local NameCert obligation. □

Theorem 3502.3 (AsymptoticSequence tail classifier)**.** *Let $A = (S, T, W, M, Q, L, E, F, H, C, P, N)$ and $A' = (S', T', W', M', Q', L', E', F', H', C', P', N')$ be accepted* $\mathsf{AsymptoticSequence}^\uparrow$ *carriers. Their accepted tail classifiers agree exactly when the paired source rows, displayed* $\mathsf{StreamName}^\uparrow$ *windows, tail-threshold rows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks,* $\mathsf{RealSequenceLimit}^\uparrow$ *handoff rows,* $\mathsf{CauchyTailEquivalence}^\uparrow$ *provenance rows,* $\mathsf{CofinalTailEquivalence}^\uparrow$ *provenance rows, componentwise* hsame *transports, displayed* Cont *replays, package provenance rows, and local naming rows agree under the displayed transport. The classifier has no further coordinate for host sequence equality, quotient tails, arbitrary subsequence choice, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. For the forward direction, read an accepted classifier agreement through the NameCert obligations of Definition 3502.2. The agreement must pass through $S, T, W, M, Q, L, E, F, H, C, P, N$, so each corresponding displayed row is preserved under the componentwise transport.

For the reverse direction, assume the listed rows agree under the displayed transport. Carrier admission reconstructs the same paired windows and tail threshold, the same $\mathsf{RegSeqRat}^\uparrow$ readback, and the same limit, Cauchy-tail, and cofinal-tail provenance rows.

Displayed replay through C and C' then gives the same consumer route, while P, N and P', N' record the same source and local naming surface. The absence of quotient, choice, host equality, and completeness coordinates shows that no hidden classifier field remains. □

Falsifiable prediction: Every $\mathsf{AsymptoticSequence}^\uparrow$ comparison read factors through paired StreamName windows, one displayed tail threshold, RegSeqRat readback, RealSequenceLimit

handoff, CauchyTailEquivalence provenance, CofinalTailEquivalence provenance, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, realsequencelimit, cauchytailequivalence, cofinaltailequivalence: $\mathsf{AsymptoticSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired source windows, a tail threshold, rational readback, limit handoff, symmetric tail comparison, cofinal route comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticSequence}^{\uparrow}$ packet is generated from paired source sequence rows, finite StreamName windows, a tail-threshold row, RegSeqRat readback, RealSequenceLimit handoff, CauchyTailEquivalence and CofinalTailEquivalence provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3502.1, Definition 3502.2, and Definition 3502.3 seed the finite tail-comparison certificate for sequence-limit and Cauchy-criterion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient stream equality, host asymptotic comparison, arbitrary subsequence choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Formal closure requires a Lean carrier and obligation package for the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{CofinalTailEquivalence}^{\uparrow}$, hsame, Cont, Pkg, and NameCert route.

Chapter 3503

A Concrete Naming Certificate for FiniteSubcoverCriterion$^{\uparrow}$

Following the Heine–Borel and Bishop finite-subcover discipline, the FiniteSubcoverCriterion$^{\uparrow}$ packet records the finite-net criterion that lets a compactness consumer use displayed metric cover data without importing arbitrary open-cover compactness. It sits next to the compact metric carrier of chapter 129, the finite center family of chapter 1767, the located interval boundary of chapter 394, and the terminal real-name seal of chapter 38. The packet names a finite criterion over already displayed rows; it does not select subcovers from arbitrary covers, choose representatives, assert countable choice, or use ambient Real$^{\uparrow}$ completeness.

Definition 3503.1 (FiniteSubcoverCriterion carrier)**.** *A* FiniteSubcoverCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, E, L, R, Q, D, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposed only through its displayed finite-net and metric coverage fields.*
2. *E is the* FiniteEpsilonNet$^{\uparrow}$ *center-bundle row for the requested positive scale.*
3. *L is the* LocatedInterval$^{\uparrow}$ *or located source-window row when the compact source is an interval-facing carrier.*
4. *R is the* Real$^{\uparrow}$ *seal row attached to endpoint, radius, and distance comparisons already displayed by the source.*
5. *Q is the cover-read ledger, recording for each bundled center the finite witness row that covers the corresponding metric ball.*
6. *D is the dyadic or rational comparison ledger certifying that the center radius and mesh slack fit the requested coverage scale.*
7. *H records componentwise* hsame *transport for the compact source, finite-net row, located window, Real seal, cover-read ledger, comparison ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a finite cover request through $K, E, L, R, Q, D, H, P, N$.*
9. *P is the* Pkg *provenance over* FiniteSubcoverCriterion$^{\uparrow}$, CompactMetric$^{\uparrow}$, FiniteEpsilonNet$^{\uparrow}$, LocatedInterval$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, E, L, R, Q, D, H, C, P, N$. It has no coordinate for arbitrary open covers, countable choice, selected representatives, a host finite-subcover operation, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteSubcoverCriterionUp.FiniteSubcoverCriterionTasteGate single carrier alignment`

Theorem 3503.2 (FiniteSubcoverCriterion NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *carrier $S = (K, E, L, R, Q, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the compact metric source, finite epsilon-net row, located source-window row, Real seal, cover-read ledger, rational comparison ledger, transport, replay, provenance, and local naming rows;*
2. *finite-net exposure, requiring every cover read to display the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *center bundle E before any covering witness is inspected;*
3. *source compatibility, requiring K, L, R to be read as the compact metric, located-window, and Real-seal rows already carried by sibling chapters;*
4. *radius exactness, requiring Q and D to record the finite center-to-cover and radius-slack checks that make the criterion usable by downstream compactness consumers;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no open-cover compactness, countable choice, selected representatives, host finite-subcover operation, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3503.1. Its coordinates are exactly the compact source, finite net, located source window, Real seal, cover-read ledger, comparison ledger, and structural rows.

Finite-net exposure is immediate from the order of the displayed packet. A cover consumer must first read E, so each later covering witness is tied to a named center bundle rather than to an arbitrary family of open sets.

Source compatibility is supplied by K, L, R. These rows identify the compact metric, located-window, and Real-seal surfaces already present in the sibling chapters, while Q and D record only finite coverage and comparison checks.

Finally, the refused objects have no coordinate in S. Transport, replay, provenance, and local naming preserve the same finite rows, so no accepted packet exports open-cover compactness, countable choice, selected representatives, host finite-subcover selection, quotient real equality, or ambient Real completeness. □

Theorem 3503.3 (FiniteSubcoverCriterion net extraction)**.** *Every compactness-facing read from an accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported object is the displayed finite-net subcover criterion over $K, E, L, R, Q, D, H, C, P, N$, not an open-cover compactness theorem.

Proof. Begin with an accepted carrier $S = (K, E, L, R, Q, D, H, C, P, N)$. The compactness-facing source is K, and the finite centers are exposed by E before the located and Real-facing rows L and R are read.

The cover ledger Q assigns the displayed cover witnesses only to the bundled centers, while D records the rational slack needed to pass from center coverage to the requested metric scale. Thus the extraction is a finite read from the listed packet rows.

Replay through C, with transport through H and provenance through P, preserves that route. The local naming row N therefore exports the finite-net criterion and cannot add open-cover compactness, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3503.4 (FiniteSubcoverCriterion compact-metric consumer)**.** *Let $S = (K, E, L, R, Q, D, H, C, P, N)$ be an accepted* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *carrier. Any compact-metric consumer that reads the finite subcover criterion must first display the* $\mathsf{CompactMetric}^{\uparrow}$ *source data K, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *center family E, the* $\mathsf{LocatedInterval}^{\uparrow}$ *or located-radius boundary row L, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows carried by D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R before the cover-read ledger Q is consumed. The exported criterion is therefore a finite compact-metric route through $K, E, L, D, R, Q, H, C, P, N$, with no arbitrary open-cover compactness, selected representative, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3503.1. The only compact-metric source coordinate is K, and the only finite center-family coordinate is E.

By Definition 3503.2, source compatibility and finite-net exposure require K and E to be visible before any cover witness is inspected. The located boundary row L and the terminal seal row R are part of the same source-compatibility boundary.

By Definition 3503.3, every compactness-facing read factors through the finite route from compact metric data to finite epsilon nets, located boundary rows, and the Real seal. The dyadic tolerance ledger D records the finite radius slack that makes the cover-read ledger Q meaningful to that route.

The remaining coordinates H, C, P, N only transport, replay, source, and name the same finite packet. Since the carrier has no row for arbitrary open-cover compactness, selected representatives, countable choice, quotient real equality, or ambient Real completeness, no compact-metric consumer can obtain one from the finite subcover criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a FiniteEpsilonNetUp center bundle, a LocatedIntervalUp source-window row, a RealUp seal, finite cover reads, rational comparison ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3503.1, Definition 3503.2, and Definition 3503.3 seed the finite-net compactness criterion consumed by later compact-continuity and Heine–Borel routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSubcoverCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected representatives, a host finite-subcover operation, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, source compatibility, radius exactness, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubcoverCriterionUp.

Chapter 3504

A Concrete Naming Certificate for $\mathsf{RieszMarkov}^{\uparrow}$

Following Riesz and Markov, the $\mathsf{RieszMarkov}^{\uparrow}$ packet isolates the compact positive functional representation surface: a positive linear functional on continuous real-valued observables over a displayed compact source is read only through a representing Radon row and an integral-readback ledger. It consumes the real scalar boundary of chapter 38, the compact source of chapter 60, the continuous observable rows of chapter 130, the regular measure packet of chapter 2263, and the broader representation boundary of chapter 1643. The chapter names the finite BEDC obligation surface needed by compact-analysis consumers without choosing a representing measure from a host theorem.

3504.1 Compact positive functional carrier

Definition 3504.1 (RieszMarkov compact positive functional carrier)**.** *A* $\mathsf{RieszMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{RieszMarkov}(K, F, L, M, I, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, read through* $\mathsf{Compact}^{\uparrow}$ *over the inherited* $\mathsf{Real}^{\uparrow}$ *scalar boundary;*
2. *F is the continuous real-valued observable row, read through* $\mathsf{ContinuousMap}^{\uparrow}$ *with codomain* $\mathsf{Real}^{\uparrow}$*;*
3. *L is the positive functional ledger, recording linear scalar reads and positivity reads on the displayed observable rows;*
4. *M is the representing* $\mathsf{RadonMeasure}^{\uparrow}$ *row over the same compact source;*
5. *I is the integral-readback row comparing each displayed functional value with the corresponding integral against M;*
6. *H is the componentwise* hsame *transport ledger for K, F, L, M, I, C, P, N;*
7. *C is the displayed* Cont *replay ledger for compact source, observable, functional, representing measure, and integral-readback routes;*
8. *P is the* Pkg *provenance over* $\mathsf{RieszMarkov}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{RieszRepresentation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*

9. N is the local $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, M, I, H, C, P, N$. It does not identify host functions, host measures, arbitrary sigma-algebras, quotient measure classes, selected representing measures, or ambient functional extensionality.

3504.2 Certificate obligations

Theorem 3504.2 (RieszMarkov NameCert obligations)**.** *For every accepted* $\mathsf{RieszMarkov}^{\uparrow}$ *carrier*

$$R = \mathsf{RieszMarkov}(K, F, L, M, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, real-valued continuity admission through F, positive-functional ledger visibility through L, representing Radon row visibility through M, integral-readback exactness through I, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

The certificate refuses arbitrary choice of representing measures, host sigma-algebras beyond the displayed $\mathsf{RadonMeasure}^{\uparrow}$ *rows, quotient equality of measures, host equality of functionals, and ambient functional extensionality. A consumer may use the representation only by reading the finite rows of R.*

Lean4 ✓ `BEDC.Derived.RieszMarkovUp.RieszMarkovCarrier namecert obligations`

Proof. Project the carrier of Definition 3504.1. The compact source and continuous observable are fixed by K and F, so the functional ledger L is read on displayed observables rather than on a host function space.

The representing object is the displayed Radon row M. The readback row I is the only route from a functional value to an integral value, and it is attached to the same compact source and observable rows.

Transport and replay are componentwise through H and C, while P and N source and name the route. Since no carrier coordinate stores a selector for representing measures, a larger host sigma-algebra, a quotient measure equality, host functional equality, or functional extensionality, the stated rows exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$, seed)

Scope closed: Definition 3504.1 and Definition 3504.2 seed the compact positive functional representation packet over Real, Compact, ContinuousMap, RadonMeasure, RieszRepresentation, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RieszMarkov}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary representing-measure choice, host sigma-algebras beyond displayed Radon rows, quotient measure equality, ambient functional extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-functional ledger visibility, representing Radon row visibility, integral-readback exactness, transport replay, and consumer refusal for BEDC.Derived.RieszMarkovUp.

Constructive story: A $\mathsf{RieszMarkov}^{\uparrow}$ packet is generated from compact source, continuous real-valued observable, positive functional ledger, representing Radon row, integral-readback row, componentwise transport, replay, provenance, and local NameCert rows.

Chapter 3505

A Constructive Apartness Certificate for $\mathsf{MetricApartness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{MetricApartness}^{\uparrow}$ packet records the positive-distance separation surface of a constructive metric space as finite BEDC NameCert data. It sits after the metric carrier of chapter 57, the terminal real-name seal of chapter 38, the separated zero-distance classifier of chapter 1411, and the completion boundary of chapter 1506. The packet supplies a finite apartness interface for metric and completion consumers; it does not import host inequality, real trichotomy, quotient classes, selected representatives, or ambient completeness.

Definition 3505.1 (MetricApartness carrier)**.** *A* $\mathsf{MetricApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, x, y, D, R, Z, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the displayed distance-read interface.*
2. *x and y are the two inspected point rows over the same metric source.*
3. *D is the metric-distance witness row for the finite read $d(x, y)$.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *positive-distance row, recording that the displayed distance has a positive real witness.*
5. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance refusal row, recording that the accepted apartness read rejects the zero-distance branch by displayed real data.*
6. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier-comparison row, used only to compare the zero-distance boundary with the displayed point classifier.*
7. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *finite-window transport row for completion-facing consumers.*
8. *H records componentwise* hsame *transport for the metric source, point rows, distance witness, real positivity row, zero-distance refusal row, separated comparison, finite-window transport, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an apartness request through $M, x, y, D, R, Z, S, Q, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$ *row.*

The classifier compares accepted apartness packets only through these displayed rows. It contains no host inequality predicate, no trichotomy of real numbers, no quotient-completion class, no selected representative, and no ambient completeness witness.

Lean4 ✓ `BEDC.Derived.MetricApartnessUp.TasteGate.MetricApartnessTasteGate single carrier alignment`

Theorem 3505.2 (MetricApartness NameCert obligations)**.** *For every accepted* $\mathsf{MetricApartness}^{\uparrow}$ *carrier* $A = (M, x, y, D, R, Z, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source and point rows, distance-witness visibility through* D*, positive-distance readback through* R*, zero-distance refusal through* Z*, separated classifier comparison through* S*, completion-facing finite-window transport through* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every apartness read therefore follows the route*

$$M, x, y \longmapsto D \longmapsto R, Z \longmapsto S, Q \longmapsto H, C, P, N.$$

No additional coordinate can decide apartness by host inequality, real trichotomy, quotient class comparison, selected representative equality, or ambient completeness.

Proof. Project the carrier of Definition 3505.1. The rows M, x, y expose the metric source and the two inspected points, and D is the only row from which their displayed distance can be read.

The real-facing branch is then forced through R and Z. The row R supplies positive-distance evidence, while Z records refusal of the zero-distance branch by the same $\mathsf{Real}^{\uparrow}$ readback surface.

Separated and completion-facing consumers are downstream. The row S compares the zero-distance boundary with the separated classifier, and Q transports finite-window completion reads through $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ without creating quotient classes or selected representatives.

Finally H, C, P, N preserve, replay, source, and name the displayed route. Since no row stores host inequality, real trichotomy, quotient comparison, selected representative equality, or ambient completeness, the listed coordinates exhaust the NameCert surface. □

Theorem 3505.3 (MetricApartness completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchySequenceEquivalence}$ *consumer of an accepted* $\mathsf{MetricApartness}^{\uparrow}$ *packet must read the finite route* $M, x, y, D, R, Z, S, Q, H, C, P, N$ *before treating the two displayed names as apart. The exported handoff is only the positive-distance certificate together with zero-distance refusal, separated-classifier comparison, finite-window transport, componentwise transport, replay, provenance, and local naming. It does not export host inequality, real trichotomy, quotient-completion equality, selected representatives, or an ambient completeness principle.*

Proof. Begin with a completion-facing apartness request. The metric source and inspected point rows M, x, y must be displayed before the distance witness D can be consumed.

The rows R and Z bind the constructive apartness decision to real readback: one row supplies positive-distance evidence, and the other refuses the zero-distance branch. This gives the separated row S a displayed classifier boundary to compare.

The finite-window row Q is then the only completion-facing handoff. It transports the same distance and refusal data through $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ consumers, while H, C, P, N record transport, replay, provenance, and local naming.

Thus the completion consumer receives exactly the finite apartness route described above, with no hidden route through host inequality, trichotomy, quotient classes, selected representatives, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{MetricApartness}^{\uparrow}$ read factors through a MetricUp source, two displayed point rows, a distance witness, RealUp positive-distance evidence, RealUp zero-distance refusal, a SeparatedMetric classifier comparison, MetricCompletion and CauchySequenceEquivalence finite-window transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, real, separatedmetric, metriccompletion, cauchysequenceequivalence: $\mathsf{MetricApartness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, two displayed point rows, positive-distance evidence, zero-distance refusal, separated classifier comparison, completion-facing finite-window transport, transport, replay, provenance, and local naming as one apartness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricApartness}^{\uparrow}$ packet is generated from a MetricUp source row, two displayed point rows, a metric-distance witness, RealUp positive-distance evidence, RealUp zero-distance refusal, a SeparatedMetric classifier-comparison row, MetricCompletion and CauchySequenceEquivalence finite-window transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3505.1, Definition 3505.2, and Definition 3505.3 seed the finite constructive apartness surface for metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inequality, real trichotomy, quotient-completion equality, selected representatives, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-witness visibility, positive-distance evidence, zero-distance refusal, separated classifier comparison, completion-facing finite-window transport, transport and replay stability, provenance non-escape, and local naming for BEDC.Derived.MetricApartnessUp.

Chapter 3506

A Concrete Naming Certificate for $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ packet records the finite orbit-to-Cauchy-completion handoff produced by a contraction before any fixed-point endpoint is sealed. It sits after the contraction-map surface of chapter 1418, the Picard-step ledger of chapter 395, the complete-metric service of chapter 128, the convergence-modulus boundary of chapter 1438, and the fixed-point endpoint discipline of chapter 396. The packet exposes the finite carrier and residual-modulus rows that a completion consumer must read before speaking about a completed orbit point.

Definition 3506.1 (ContractiveCauchyOrbit carrier)**.** *A* $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, T, I, W, R, M, K, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the carried* $\mathsf{CompleteMetric}^{\uparrow}$ *row over which distances between displayed orbit entries are read.*
2. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row, including the displayed contraction ratio and the self-map application route.*
3. *I is the* $\mathsf{PicardContraction}^{\uparrow}$ *finite iterate ledger recording the visible orbit entries $x_0, \ldots, x_n$ and the application rows $x_{k+1} = T(x_k)$.*
4. *W is the finite iterate-window exactness row, recording which adjacent orbit windows are available to every tolerance request.*
5. *R is the residual-distance row, storing the adjacent distance comparisons and ratio propagation read from X, T, I, W.*
6. *M is the extracted* $\mathsf{CauchyConvergenceModulus}^{\uparrow}$ *row, obtained from the residual ledger without selecting a limit.*
7. *K is the completion handoff row, naming the* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer that may read the Cauchy orbit after M is displayed.*
8. *E is the endpoint-seal boundary available only after the completion handoff has been read by a downstream fixed-point or real-name consumer.*

9. *H records componentwise* hsame *transport for the metric, contraction, iterate, window, residual, modulus, completion handoff, endpoint boundary, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from contraction admission through finite orbit windows, residual comparisons, modulus extraction, completion handoff, and endpoint separation.*
11. *P is the* Pkg *provenance over* ContractiveCauchyOrbit$^{\uparrow}$, ContractionMapping$^{\uparrow}$, PicardContraction$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyConvergenceModulus$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{ContractiveCauchyOrbit}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected fixed point, host limit equality, quotient completion representative, function extensionality, countable choice, or ambient completeness outside the displayed CompleteMetric$^{\uparrow}$ *row.*

Lean4 (definition) `BEDC.Derived.ContractiveCauchyOrbitUp.ContractiveCauchyOrbitUp`
Lean4 ✓ `BEDC.Derived.ContractiveCauchyOrbitUp.ContractiveCauchyOrbitTasteGate single carrier alignment`

Theorem 3506.2 (ContractiveCauchyOrbit NameCert obligations)**.** *For every accepted* ContractiveCauchyOrbit$^{\uparrow}$ *carrier*

$$O = (X, T, I, W, R, M, K, E, H, C, P, N),$$

the local NameCert$_{\text{ContractiveCauchyOrbit}^{\uparrow}}$ *surface consists exactly of carrier admission, contraction-row visibility, finite iterate-window exactness, residual Cauchy-modulus extraction, completion handoff, endpoint-boundary separation, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger non-escape for the rows of Definition 3506.1.*

Proof. Carrier admission projects the displayed tuple. The rows X, T, I, W expose the complete metric, contraction map, finite Picard ledger, and exact finite windows before any convergence-facing row is read.

The residual row R is then forced through the same finite data. It records adjacent distances and contraction-ratio propagation, so the modulus row M is extracted from displayed orbit windows rather than from a host convergence theorem.

The completion handoff K and endpoint boundary E remain downstream. They may be inspected only after the residual-to-modulus route has been displayed, and they do not add a selected fixed point or quotient representative.

Finally H, C, P, N transport, replay, source, and name exactly these rows. Thus every NameCert obligation is visible in the carrier, and the ledger has no escape route outside the twelve displayed coordinates. □

Theorem 3506.3 (ContractiveCauchyOrbit completion handoff)**.** *Every completion-facing consumer of an accepted* ContractiveCauchyOrbit$^{\uparrow}$ *carrier must read the ordered route*

$$\text{ContractionMapping}^{\uparrow},\ \text{PicardContraction}^{\uparrow},\ \text{CompleteMetric}^{\uparrow} \longmapsto W, R, M \longmapsto K, E, H, C, P, N.$$

The consumer receives the finite iterate windows, residual-distance comparisons, and extracted Cauchy modulus before it may read the completion handoff. It receives no Banach fixed point, selected limit, host function equality, quotient completion representative, or ambient completeness beyond the named CompleteMetric$^{\uparrow}$ *row.*

Proof. Project the contraction and iteration rows. The $\mathsf{ContractionMapping}^{\uparrow}$ row supplies the self-map and ratio data, while the $\mathsf{PicardContraction}^{\uparrow}$ row supplies the finite orbit ledger.

Read the complete-metric row next. All residual comparisons in R are distance reads through X, and the exact windows W determine which finite orbit entries are visible to a tolerance request.

The row M is therefore a Cauchy-modulus extraction from finite residual data. The completion handoff K may consume this route, and the endpoint boundary E may be read only after K.

The structural rows preserve the same dependency order. Since no row stores a selected endpoint or host equality, every completion-facing consumer remains confined to the displayed handoff. □

Theorem 3506.4 (ContractiveCauchyOrbit ledger non-escape)**.** *Every public read of an accepted* $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ *packet factors through*

$$X, T, I, W, R, M, K, E, H, C, P, N.$$

In particular, the residual Cauchy-modulus extraction and completion handoff are visible as finite BHist *rows, and no consumer can replace them by an external convergence theorem, a chosen completion point, a quotient representative, countable choice, or host fixed-point equality.*

Proof. Read the carrier definition together with the NameCert obligation theorem. They expose exactly the twelve rows listed in the statement, with finite orbit windows and residual comparisons preceding the modulus row.

Apply Definition 3506.3. Any completion-facing read must pass through the displayed contraction, Picard, complete-metric, window, residual, and modulus route before K and E are available.

The transport, replay, provenance, and naming rows add no analytic coordinate. Therefore a public consumer can only inspect the displayed carrier route, and the excluded host, choice, quotient, and endpoint objects remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ completion read factors through one CompleteMetric row, one ContractionMapping row, one PicardContraction finite orbit ledger, one finite iterate-window exactness row, one residual-distance row, one CauchyConvergenceModulus extraction, one completion handoff, one endpoint boundary, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: contractionmapping, picardcontraction, completemetric, cauchy-convergence-modulus, banachfixedpoint: none is bijective with $\mathsf{ContractiveCauchyOrbit}^{\uparrow}$ because the packet jointly stores the contraction row, Picard orbit ledger, complete-metric distance reads, finite window exactness, residual modulus extraction, completion handoff, endpoint boundary, transport, replay, provenance, and local naming as one finite completion-facing orbit.

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3506.1, Definition 3506.2, Definition 3506.3, and Definition 3506.4 seed the finite contraction-orbit route from Picard rows to Cauchy completion handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyOrbit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Banach fixed points, selected limits, quotient completion representatives, host function equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-row visibility, finite iterate-window exactness, residual Cauchy-modulus extraction, completion handoff, endpoint-boundary separation, transport, replay, provenance, local naming, and ledger non-escape for BEDC.Derived.ContractiveCauchyOrbitUp.

Constructive story: A ContractiveCauchyOrbit$^{\uparrow}$ packet is generated from a complete-metric row, contraction-map row, PicardContraction finite iterate ledger, finite window exactness row, residual-distance row, extracted CauchyConvergenceModulus row, completion handoff, endpoint boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3507

A Concrete Naming Certificate for $\mathsf{CompletionSubsetClosure}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{CompletionSubsetClosure}^{\uparrow}$ packet records the traditional fact that a closed subset of a complete metric completion remains complete, but only as finite BEDC NameCert data. It sits over the complete separable metric surface of chapter 1735, the dense embedding route of chapter 3351, the dense Cauchy embedding packet of chapter 2250, the Polish-space readback of chapter 1490, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a closed-subset completion surface for downstream completion consumers; it is not open-cover compactness, a quotient equality of completed points, countable choice, selected limit extraction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3507.1 (Completion subset closure carrier)**.** *A* $\mathsf{CompletionSubsetClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, E, D, P, S, W, Q, R, L, H, C, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *source row exposing the complete separable metric packet from which the closed subset is read.*
2. *E is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *dense-embedding row naming the completion-facing source image.*
3. *D is the* $\mathsf{DenseCauchyEmbedding}^{\uparrow}$ *handoff row, recording the Cauchy-name route by which subset points are inspected.*
4. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *consumer row, used only after the complete separable and dense embedding rows have been displayed.*
5. *S is the closed-subset ledger, listing subset membership, closedness witnesses, and boundary refusal rows over the completion source.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family used to inspect subset membership and Cauchy approximation rows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed Cauchy and distance-budget data.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the finite distance and radius comparisons read by the subset closure ledger.*

9. *L is the limit-handoff row, recording that every accepted Cauchy read remaining in S reaches a displayed subset limit row before any downstream completion consumer sees it.*
10. *H records componentwise* hsame *transport for the complete metric source, dense embeddings, closed-subset ledger, stream windows, readback rows, Real seals, limit handoff, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from subset Cauchy reads through $M, E, D, P, S, W, Q, R, L, H$.*
12. *G is the* Pkg *provenance over* $\mathsf{CompletionSubsetClosure}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, MetricCompletionDenseEmbe DenseCauchyEmbedding$^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, D, P, S, W, Q, R, L, H, C, G, N$. It has no coordinate for open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3507.2 (CompletionSubsetClosure NameCert obligations)**.** *For every accepted* CompletionSubsetClosure$^{\uparrow}$ *carrier $K = (M, E, D, P, S, W, Q, R, L, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete separable metric source, metric-completion dense embedding, dense Cauchy embedding, Polish-space consumer row, closed-subset ledger, finite stream windows, regular rational readback, Real seal, limit handoff, componentwise transport, displayed replay, provenance, and local naming.*

Every completion-facing read must therefore pass through

$$M \longmapsto E \longmapsto D \longmapsto P \longmapsto S \longmapsto W \longmapsto Q \longmapsto R \longmapsto L \longmapsto (H, C, G, N)$$

before it can treat a Cauchy read in the subset as closed under completion. The route exports no open-cover compactness theorem, no quotient equality of completion classes, no selected limit, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3507.1. The rows M, E, D, P expose the complete separable source, dense completion image, dense Cauchy handoff, and Polish consumer boundary before the subset ledger can be read.

Next inspect the closed-subset and finite observation rows. The ledger S is read only through finite windows W, regular rational data Q, and Real seals R, so a subset Cauchy request has a displayed membership and boundary route before a limit row is exposed.

Finally read the structural rows. The limit handoff L is transported by H, replayed by C, sourced by G, and named by N. Since the carrier has no row for open covers, quotient completion equality, selected limits, countable choice, trichotomy, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CompletionSubsetClosureUp.CompletionSubsetClosure namecert obligations`

Theorem 3507.3 (CompletionSubsetClosure dense embedding handoff)**.** *Every accepted* CompletionSubsetClosure$^{\uparrow}$ *packet hands a subset-completion consumer the displayed route*

$\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseCauchyEmbedding}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, StreamNa

The handoff is the finite closed-subset completion route carried by K; it is not a host theorem about all closed subsets of all complete spaces.

Proof. Begin with an accepted packet K. The complete separable source M and dense embedding row E identify the completion surface, while the dense Cauchy row D fixes how subset Cauchy names enter the packet.

The Polish row P is downstream of those completion rows, and the observation rows W, Q, R display the finite reads used by the closed-subset ledger S. Thus the subset-completion consumer sees only the named route through the listed sibling chapters.

Transport, replay, provenance, and local naming preserve that route. No host closed-subset theorem or ambient completion object is introduced by the handoff. □

Theorem 3507.4 (CompletionSubsetClosure nonescape)**.** *Every accepted* $\mathsf{CompletionSubsetClosure}^{\uparrow}$ *completion read remains inside the displayed rows* $M, E, D, P, S, W, Q, R, L, H, C, G, N$*. In particular, the packet cannot export open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet coordinates in order. The completion source, dense embedding, dense Cauchy handoff, and Polish consumer rows are exactly M, E, D, P; the subset boundary is exactly S.

The finite observation data is exactly W, Q, R, and the only completion-facing exit is the displayed limit handoff L. Componentwise transport, replay, provenance, and naming through H, C, G, N do not add new coordinates.

Thus every accepted read is exhausted by the displayed rows, and the refused principles have no carrier position from which they could be exported. □

Falsifiable prediction: Every accepted $\mathsf{CompletionSubsetClosure}^{\uparrow}$ completion read must display complete-separable source data, dense completion embedding data, dense Cauchy embedding data, Polish consumer data, a closed-subset ledger, StreamName windows, RegSeqRat readback, a RealUp seal, a limit handoff, transport, replay, provenance, and local naming before any closed-subset completion claim is public.

Independence witness: complete-separable-metric, metric-completion-dense-embedding, dense-cauchy-embedding, polishspace, streamname, regseqrat, real: $\mathsf{CompletionSubsetClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a closed-subset ledger, finite observation windows, regular readback, Real seals, a limit handoff, transport, replay, provenance, and local naming over the completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionSubsetClosure}^{\uparrow}$ packet is generated from CompleteSeparableMetricUp source rows, MetricCompletionDenseEmbeddingUp and DenseCauchyEmbeddingUp completion rows, a PolishSpaceUp consumer row, a closed-subset ledger, StreamName windows, RegSeqRat readback, RealUp seals, a limit handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3507.1, Definition 3507.2, Definition 3507.3, and Definition 3507.4 seed the closed-subset completion route over the L10 completion siblings.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionSubsetClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of completed points, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-subset ledger exactness, dense-embedding handoff, limit-handoff non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionSubsetClosureUp.

Chapter 3508

A Concrete Naming Certificate for DnaBoundaryTriggerClosure$^{\uparrow}$

The DnaBoundaryTriggerClosure$^{\uparrow}$ packet realizes the boundary-closure structure of Definition 9462.1306 as finite BEDC NameCert data. It sits over the six boundary-tile coverage packet of chapter 3470, the finite trigger algebra of chapter 3478, the cellular trace window of chapter 3481, and the weak Konig boundary refusal surface of chapter 3482. The packet names boundary-trigger closure for six-tile DNA window rows; it is not a biological code table, a stochastic evolutionary model, an infinite branch selector, a quotient codon semantics, or a host closure oracle.

Definition 3508.1 (DnaBoundaryTriggerClosure carrier)**.** *A* DnaBoundaryTriggerClosure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, A, T, W, F, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-tile coverage row, exposing the six-tile trigger and coverage-rank data.*
2. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *finite trigger-algebra row, recording closure of displayed support rows under boundary-trigger operations.*
3. *T is the* CellularWindowTrace$^{\uparrow}$ *trace row, used only to display finite local evolution windows attached to accepted boundary triggers.*
4. *W is the* WeakKonigBoundary$^{\uparrow}$ *refusal row, recording that the closure packet does not export an infinite branch selector or countable-choice schedule.*
5. *F is the finite frontier ledger listing the boundary codons, Q_2 completion faces, stage rows, and pressure rows read from the source vision.*
6. *L is the closure ledger, recording that each accepted boundary-trigger closure step is obtained from displayed six-tile rows, trigger algebra, trace windows, and weak-branch refusals.*
7. *H records componentwise* hsame *transport for boundary-tile coverage, trigger algebra, trace windows, weak-boundary refusals, frontier rows, closure rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a boundary-trigger closure request through B, A, T, W, F, L, H.*
9. *P is the* Pkg *provenance over* DnaBoundaryTriggerClosure$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAl CellularWindowTrace$^{\uparrow}$, WeakKonigBoundary$^{\uparrow}$, *the boundary-closure source vision,* BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, A, T, W, F, L, H, C, P, N$. It has no coordinate for hidden genetic-code semantics, stochastic dynamics, infinite branch choice, quotient codon equality, or a host closure oracle.

Lean4 ✓ `BEDC.Derived.DnaBoundaryTriggerClosureUp.DnaBoundaryTriggerClosureTasteGate single carrier alignment`

Theorem 3508.2 (DnaBoundaryTriggerClosure NameCert obligations)**.** *For every accepted* DnaBoundaryTriggerCl *carrier $K = (B, A, T, W, F, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for six-boundary-tile coverage, window-coverage trigger algebra, cellular window trace, weak Konig boundary refusal, finite frontier rows, closure ledger rows, componentwise transport, displayed replay, provenance, and local naming.*

Every boundary-trigger closure read must therefore route through

$$B \longmapsto A \longmapsto T \longmapsto W \longmapsto F \longmapsto L \longmapsto (H, C, P, N)$$

before it can be consumed as a finite closure packet for DNA window rows. The route exports no biological code table, stochastic dynamics, infinite branch selector, quotient codon equality, or host closure oracle.

Proof. Project the carrier of Definition 3508.1. The rows B, A, T, W expose the four sibling surfaces: six-boundary coverage, trigger algebra, cellular trace, and weak-branch refusal.

The frontier and closure ledgers are finite. The row F records only displayed boundary codons, Q_2 completion faces, stage rows, and pressure rows, while L records closure steps obtained from those rows and the sibling packets.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no hidden semantics, stochastic model, infinite branch selector, quotient equality, or host oracle is present in the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 3508.3 (DnaBoundaryTriggerClosure nonescape)**.** *Every accepted* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *boundary-trigger closure read passes through displayed six-tile boundary rows, finite trigger algebra rows, cellular trace rows, weak-branch refusal rows, frontier rows, closure rows, transport, replay, provenance, and local naming. The packet cannot escape to a biological code table, stochastic dynamics, infinite branch choice, quotient codon equality, or host closure oracle.*

Proof. Read B first. It supplies the six-boundary tile coverage row, so the trigger source is a displayed finite boundary packet rather than an implicit biological table.

Read A, T, W next. The trigger algebra closes only finite support rows, the cellular trace row supplies only finite local windows, and the weak-boundary row records the refusal of infinite branch selection.

The rows F, L, H, C, P, N then bind the frontier, closure, transport, replay, provenance, and naming surface. No further coordinate remains, so an accepted read cannot export any of the refused objects. □

Falsifiable prediction: Every boundary-trigger closure read for $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ must pass through displayed six-tile boundary rows, finite trigger-algebra rows, cellular trace windows, weak-branch refusal rows, frontier rows, closure rows, transport, replay, provenance, and local naming; omitting any displayed six-tile boundary row refutes the packet.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, cellular-window-trace, weak-konig-boundary: $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six-tile coverage, finite trigger algebra, cellular trace windows, weak-branch refusal, frontier ledgers, closure ledgers, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.DnaBoundaryTriggerClosureUp.TasteGate` supplies explicit $\mathsf{Nontrivial\ DnaBoundaryTriggerClosure}^{\uparrow}$ and $\mathsf{FieldFaithful\ DnaBoundaryTriggerClosure}^{\uparrow}$ instances preserving $B, A, T, W, F, L, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ packet is generated from SixBoundaryTileCoverageUp boundary rows, WindowCoverageTriggerAlgebraUp finite trigger rows, CellularWindowTraceUp finite trace windows, WeakKonigBoundaryUp refusal rows, finite frontier ledgers, closure ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3508.1, Definition 3508.2, and Definition 3508.3 seed the finite boundary-trigger closure packet for six-tile DNA window rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological code-table identity, stochastic evolutionary dynamics, infinite branch choice, quotient codon equality, host closure oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, six-tile boundary coverage, trigger-algebra exactness, trace-window handoff, weak-branch refusal, Nontrivial DnaBoundaryTriggerClosureUp, and FieldFaithful DnaBoundaryTriggerClosureUp for BEDC.Derived.DnaBoundaryTriggerClosureUp.

Chapter 3509

A Concrete Naming Certificate for $\mathsf{Ceva}^{\uparrow}$

Following Ceva, the $\mathsf{Ceva}^{\uparrow}$ packet records the finite incidence data for three cevians in a triangle and the concurrence witness that makes the side-intersection rows compatible. It sits beside the affine carrier of chapter 206, the projective incidence boundary of chapter 207, and the bilinear pairing row of chapter 146. The chapter names only triangle vertices, side-incidence rows, cevian rows, concurrence, transport, replay, provenance, and local naming; it does not export a coordinate determinant proof, a host projective plane, or an analytic area computation.

Definition 3509.1 (Ceva carrier)**.** *A* $\mathsf{Ceva}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (T, S, L, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the triangle-vertex row, recording three carried point histories read through the* $\mathsf{AffineSpace}^{\uparrow}$ *and* $\mathsf{ProjectiveSpace}^{\uparrow}$ *source certificates;*
2. *S is the side-incidence row, recording the three side lines and the finite incidence witnesses that place each side between the corresponding vertex pair;*
3. *L is the cevian row, recording one displayed cevian from each triangle vertex to the opposite side-incidence row;*
4. *X is the concurrence witness, recording that the three cevian rows meet at a common projective point;*
5. *H records componentwise* hsame *transport for vertices, side incidences, cevian rows, concurrence, and the bilinear incidence tests used by the local classifier;*
6. *C records displayed* Cont *replay for vertex admission, side construction, cevian attachment, concurrence inspection, and incidence readback;*
7. *P is the* Pkg *provenance over* $\mathsf{Ceva}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{ProjectiveSpace}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Ceva}^{\uparrow}}$ *row.*

The carrier contains no coordinate chart, no determinant numerator, no host quotient of points and lines, and no external proof that bypasses the displayed incidence rows.

Lean4 ✓ `BEDC.Derived.CevaUp.TasteGate.CevaTasteGate single carrier alignment`

Theorem 3509.2 (Ceva NameCert obligations)**.** *For the carrier of Definition 3509.1, the local* $\mathsf{NameCert}_{\mathsf{Ceva}^\uparrow}$ *surface consists exactly of carrier admission for* $V = (T, S, L, X, H, C, P, N)$*, triangle-vertex coverage through* T*, side-incidence coverage through* S*, cevian coverage through* L*, concurrence exactness through* X*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. The* $\mathsf{AffineSpace}^\uparrow$*,* $\mathsf{ProjectiveSpace}^\uparrow$*, and* $\mathsf{BilinForm}^\uparrow$ *rows are consumed only as dependency certificates for the displayed incidence packet.*

Lean4 ✓ `BEDC.Derived.CevaUp.CevaCarrier namecert obligations`

Proof. Project the carrier of Definition 3509.1. The vertex row T supplies the three carried point histories, while S records the side incidences they determine inside the affine and projective dependency surfaces.

Read the cevian row L next. Each cevian is attached to one vertex and the opposite side, so the line data cannot be consumed without the side-incidence witnesses already present in S.

The concurrence row X then binds the three cevian rows to one displayed projective point. The bilinear row is used only as the incidence test named by the packet, and the transport and replay rows H and C preserve the same vertices, sides, cevians, and concurrence witness.

Finally P and N source and name the resulting packet. Since the carrier has no further coordinate, these rows exhaust the local $\mathsf{NameCert}_{\mathsf{Ceva}^\uparrow}$ obligations. □

Theorem 3509.3 (Ceva incidence non-escape)**.** *Let*

$$V = (T, S, L, X, H, C, P, N)$$

be an accepted $\mathsf{Ceva}^\uparrow$ *packet. Any consumer that reads a Ceva concurrence conclusion must pass through the triangle-vertex row* T*, the side-incidence row* S*, the cevian row* L*, the concurrence witness* X*, the transport row* H*, the replay row* C*, the provenance row* P*, and the local naming row* N*. No conclusion can be recovered from a coordinate determinant, an unnamed projective-plane quotient, or a bilinear incidence test detached from the displayed side and cevian rows.*

Lean4 ✓ `BEDC.Derived.CevaUp.CevaCarrier incidence nonescape`

Proof. Use the obligation surface of Definition 3509.2. It places triangle vertices, side incidences, cevian rows, and concurrence in one finite packet, with the affine, projective, and bilinear sources appearing only through displayed dependency rows.

If a route omits S, it no longer knows which opposite side each cevian meets. If it omits L, it no longer has the three cevians whose concurrence is being read. If it omits X, it has only side and line data, not a concurrence witness.

The rows H, C, P, N keep the same incidence data under transport, replay, provenance, and local naming. Thus any route bypassing the listed rows exits the $\mathsf{Ceva}^\uparrow$ carrier rather than exporting a BEDC Ceva incidence conclusion. □

Falsifiable prediction: If the concurrence witness row is absent, no accepted $\mathsf{Ceva}^\uparrow$ packet exports a three-cevian concurrence reading.

Independence witness: affinespace, projectivespace, bilinform: the affine-space packet names carried point and translation data without the three side and cevian rows, the projective-space packet names scaling-orbit representatives without the Ceva concurrence witness, and the bilinear-form packet names pairing data without the triangle incidence carrier. None is bijective to $\mathsf{Ceva}^\uparrow$.

Closure status for $\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}$**.**

Constructive story: A $\mathsf{Ceva}^{\uparrow}$ packet is generated from triangle-vertex, side-incidence, cevian, concurrence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}, \text{obligation})$

Scope closed: Definition 3509.1, Definition 3509.2, and Definition 3509.3 bind the finite incidence packet for triangle vertices, side rows, cevian rows, concurrence, transport, replay, provenance, local naming, and the displayed $\mathsf{Ceva}^{\uparrow}$ carrier exclusions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Ceva}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, coordinate determinant proof, a host projective plane, host projective-plane quotient, quotient equality of point-line incidences, analytic area computation, or bridge-checked standard model.

Upgrade path: Scoped closure requires a Lean carrier and obligation package for the same triangle-vertex, side-incidence, cevian, concurrence, hsame, Cont, Pkg, and NameCert route, including carrier admission, incidence-row stability, concurrence exactness, and nonescape for BEDC.Derived.CevaUp.

Chapter 3510

A Concrete Naming Certificate for $\mathsf{BishopCauchyApproximation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyApproximation}^{\uparrow}$ packet names the finite approximation surface by which a regular Cauchy rational source answers a Real-facing precision request. It sits between the regular rational source of chapter 301, the finite observation interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the finite approximation service of chapter 601. The packet is the L10 human-origin approximation row used by Real and Completion consumers before any quotient stream, selected limit, countable choice, trichotomy, host equality, or ambient completeness principle is available.

Definition 3510.1 (Bishop Cauchy approximation source)**.** *A* $\mathsf{BishopCauchyApproximation}^{\uparrow}$ *source is a finite* BHist *packet*

$$A = (R, n, W, D, V, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational name supplies the Cauchy data.*
2. *n is the unary precision request row. It records the requested approximation depth before any finite window is opened.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at precision n.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same request and window.*
5. *V is the displayed approximation value row read from R through W and checked against D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It can be consumed only after R, n, W, D, V have all been displayed.*
7. *H stores componentwise* hsame *transports for the source, request, window, tolerance, value, seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a Real-facing approximation read through R, n, W, D, V and into the seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyApproximation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}$ *row.*

The local classifier compares two sources only through the displayed rows $R, n, W, D, V, E, H, C, P, N$. It has no coordinate for a quotient stream, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyApproximationUp.BishopCauchyApproximationTasteGate single carrier alignment`

Theorem 3510.2 (Bishop Cauchy approximation window handoff)**.** *Let*

$$A = (R, n, W, D, V, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyApproximation}^{\uparrow}$ *source. Every* $\mathsf{Real}^{\uparrow}$*-facing approximation read of A passes through, in order, the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row R, the unary precision request n, the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the displayed approximation value V, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff E, with componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support supplied by H, C, P, N. The read exports no quotient stream, selected limit, host equality, countable-choice selector, trichotomy row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the source through Definition 3510.1. The only analytic rows visible to a Real-facing consumer are R, n, W, D, V, E, while H, C, P, N supply transport, replay, provenance, and local naming.

The regular source row is read first. It presents the $\mathsf{RegSeqRat}^{\uparrow}$ Cauchy data as finite packet data, so the consumer has not yet received a Real value, quotient stream, or selected limit.

The precision request and finite window then determine the inspected observation. The row n fixes the requested depth, W opens the corresponding finite $\mathsf{StreamName}^{\uparrow}$ segment, and D records the dyadic tolerance ledger attached to that same finite request.

The displayed value row V is therefore read only after the source, request, window, and tolerance rows are all present. The terminal row E seals the result as Real-facing data, and the support rows H, C, P, N replay exactly that route.

Finally compare with the sibling service of chapter 601. Both packets use the same regular-source, finite-window, dyadic-ledger, and Real-seal order. Since neither route contains a coordinate for quotient streams, selected limits, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no such datum can be exported by the Bishop approximation handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3510.1 and Definition 3510.2 seed the L10 Bishop approximation route from $\mathsf{RegSeqRat}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteCauchyApproximation}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, host equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, precision-window handoff, dyadic tolerance exactness, RealUp seal nonescape, and finite approximation compatibility for BEDC.Derived.BishopCauchyApproximationUp.

Constructive story: A $\mathsf{BishopCauchyApproximation}^{\uparrow}$ packet is generated from a RegSeqRat source row, unary precision request, StreamName finite window, DyadicRatCore tolerance ledger, displayed approximation value, RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3511

A Concrete Naming Certificate for $\mathsf{KakutaniFixedPoint}^{\uparrow}$

Following Kakutani's fixed point theorem and constructive fixed-point readings, the $\mathsf{KakutaniFixedPoint}^{\uparrow}$ packet records only a finite certificate surface for compact convex domains, convex-valued set maps, approximate selector windows, and fixed-point witness ledgers. It sits beside $\mathsf{Compact}^{\uparrow}$ in chapter 60, $\mathsf{ConvexSet}^{\uparrow}$ in chapter 208, $\mathsf{SchauderFixedPoint}^{\uparrow}$ in chapter 1923, $\mathsf{FixedPointIteration}^{\uparrow}$ in chapter 2383, and $\mathsf{Met}^{\uparrow}$ in chapter 57. The packet is a NameCert surface for the fixed-point obligation; it does not import a host choice selector, a full classical compactness theorem, or ambient real completeness.

Definition 3511.1 (Kakutani fixed point carrier)**.** *A* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, G, V, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the named* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *surfaces.*
2. *G is the set-valued graph row, recording only finite graph windows and their source ledger.*
3. *V is the nonempty compact convex value row for the displayed graph windows.*
4. *S is the approximate selector-window row, listing finite selector approximants and their domain restrictions.*
5. *F is the fixed-point witness ledger, recording approximate self-contact rows and the accepted witness boundary.*
6. *H stores componentwise* hsame *transport for the domain, graph, value, selector, witness, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from domain reads to graph windows, from graph windows to selector windows, and from selector windows to the fixed-point witness ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{KakutaniFixedPoint}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{SchauderFixedPoint}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, G, V, S, F, H, C, P, N$. It has no coordinate for a host choice selector, a full classical compactness principle, an ambient real-completeness oracle, or an unrestricted multivalued map.

Lean4 ✓ `BEDC.Derived.KakutaniFixedPointUp.KakutaniFixedPointTasteGate single carrier alignment`

Theorem 3511.2 (Kakutani fixed point NameCert obligations)**.** *For every accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier $K = (D, G, V, S, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, compact-convex domain admission through D, finite set-valued graph access through G, nonempty compact convex value admission through V, approximate selector-window admission through S, fixed-point witness ledger admission through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3511.1. The domain, graph, value, selector, and witness coordinates are exactly D, G, V, S, F, so every fixed-point observation enters through those displayed rows.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3511.3 (Kakutani graph-selector nonescape)**.** *Every selector-facing read from an accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier factors through*

$$D, G, V \longmapsto S \longmapsto F$$

together with H, C, P, N. Consequently a consumer may use the displayed $\mathsf{Compact}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, *and* $\mathsf{Met}^{\uparrow}$ *interfaces named by the packet, but it receives no host selection function, no unbounded graph oracle, no full classical compactness theorem, and no ambient real-completeness principle.*

Proof. Begin with a selector-facing request. By Definition 3511.1, the request can inspect the compact convex domain row D, the finite set-valued graph row G, and the nonempty compact convex value row V before it reads a selector window.

The selector row S exposes only the displayed approximate selectors and their domain restrictions. The witness ledger F repacks those selector reads as approximate self-contact and fixed-point boundary data.

Finally read H, C, P, N. These rows preserve, replay, source, and name the same displayed data, so none of the excluded host principles can escape through the accepted classifier. □

Theorem 3511.4 (Kakutani convex compact value scope)**.** *The value coordinate V of an accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *carrier is scoped only as a displayed nonempty compact convex value row relative to D and G. It may feed the selector-window route S and the witness ledger F, but it exports no ambient hyperspace compactness theorem, no quotient set-of-points object, no arbitrary selection principle, and no host real-completeness assumption.*

Proof. Use the carrier rows D, G, V from Definition 3511.1. The value row is admitted relative to the displayed domain and graph windows, so its compactness and convexity evidence remains a finite row in the packet.

The only downstream consumers are the selector windows S and the fixed-point witness ledger F. The structural rows H, C, P, N then keep the value evidence tied to the same finite NameCert route rather than an ambient hyperspace or choice construction. □

Theorem 3511.5 (Kakutani Brouwer Schauder handoff)**.** *Every accepted* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *fixed-point read that is later consumed by* $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, *or* $\mathsf{NashEquilibrium}^{\uparrow}$ *factors first through the compact-convex domain row D, the finite set-valued graph row G, the nonempty compact convex value row V, approximate selector windows S, and the fixed-point witness ledger F of Definition 3511.1. The handoff reaches the Brouwer and Schauder fixed-point surfaces only through those displayed rows together with H, C, P, N; it exports no host choice selector, unrestricted hyperspace compactness theorem, selected fixed point, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or equilibrium/common-fixed-point shortcut.*

Proof. Begin with the carrier and NameCert obligations of Definition 3511.1 and Definition 3511.2. These fix the only admissible rows for the domain, graph, values, selectors, witness ledger, transport, replay, provenance, and local naming.

Next apply Definition 3511.3. A Brouwer or Schauder consumer can read selector data only after D, G, V have been displayed, and the selector window can be repacked into F only as finite approximate self-contact data.

Finally use Definition 3511.4. The value row remains a displayed nonempty compact convex row relative to the graph and domain. Therefore downstream equilibrium or common-fixed-point consumers receive the Brouwer/Schauder handoff through the finite Kakutani route, with no host selector, hyperspace compactness oracle, chosen fixed point, or ambient real-completeness principle. □

Falsifiable prediction: Every $\mathsf{KakutaniFixedPoint}^{\uparrow}$ fixed-point read factors through a compact convex domain row, finite set-valued graph row, nonempty compact convex value row, approximate selector windows, fixed-point witness ledger, transport, continuation, provenance, and local naming.

Independence witness: compact, convexset, schauder-fixed-point, fixed-point-iteration, metric-space: $\mathsf{KakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact convex domain, set-valued graph, nonempty compact convex values, approximate selector windows, fixed-point witness ledger, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{KakutaniFixedPoint}^{\uparrow}$ packet is generated from finite rows for a compact convex domain, set-valued graph, nonempty compact convex values, approximate selector windows, a fixed-point witness ledger, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3511.1, Definition 3511.2, Definition 3511.3, Definition 3511.4, and Definition 3511.5 bind the finite Kakutani fixed-point certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, full classical compactness, ambient real completeness, unrestricted hyperspace compactness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, graph-selector nonescape, convex-compact value scope, and TasteGate field faithfulness for BEDC.Derived.KakutaniFixedPointUp.

Chapter 3512

A Concrete Naming Certificate for $\mathsf{RatCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of the real line as regular Cauchy data over the rationals, the $\mathsf{RatCauchyCompletion}^{\uparrow}$ packet records the finite BEDC interface by which rational rows are read as regular rational Cauchy names and then presented to the real-name seal. It consumes the rational source surface of chapter 37, the Cauchy-modulus tail budget of chapter 292, the finite schedule surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet gives Phase Real and Completion consumers a named rational-Cauchy handoff without quotienting Cauchy names, selecting limits, invoking countable choice, or asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3512.1 (RatCauchyCompletion carrier)**.** *A* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, M, S, Q, D, E, H, K, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Rat}^{\uparrow}$ *source row, recording the rational entries that seed the completion-facing stream.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *tail row, recording the displayed budget by which two sufficiently late rational observations are forced inside the requested tolerance.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite windows through which later consumers request rational approximants.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence visible to the completion corridor.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic budgets, endpoint comparisons, and finite error rows used to check the regularity bounds.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording only the handoff boundary at which the regular rational name may be read as a real-facing name.*
7. *H records componentwise* hsame *transport for the rational source, tail budget, schedule, regular readback, dyadic ledger, seal, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *routes from rational source rows through tail budgets, schedules, regular readback, dyadic tolerance checks, and the real-seal boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RatCauchyCompletion}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, S, Q, D, E, H, K, P, N$. It has no coordinate for quotient-real equality, selected limits, arbitrary Cauchy filters, countable-choice schedules, host completeness, or a theorem that every $\mathsf{Real}^{\uparrow}$ *request is complete.*

Theorem 3512.2 (RatCauchyCompletion NameCert obligations)**.** *For every accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier*

$$C = (R, M, S, Q, D, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier exposure for the rational source row R, Cauchy-modulus tail row M, stream schedule row S, regular rational readback row Q, dyadic tolerance ledger D, real-seal boundary E, componentwise transport row H, continuation row K, package provenance row P, and local naming row N.*

Lean4 ✓ `BEDC.Derived.RatCauchyCompletionUp.TasteGate.RatCauchyCompletionCarrier namecert obligations`

Proof. Carrier exposure is projection from Definition 3512.1. Each row named in the statement is one coordinate of the finite packet, and the definition lists no hidden coordinate for completion beyond those rows.

The rational source and Cauchy-modulus obligations are R and M. The stream and regular-readback obligations are S and Q. The dyadic ledger and real-facing seal obligations are D and E.

Transport, replay, provenance, and local naming are supplied by H, K, P, N. Repacking these coordinates yields the displayed $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$ surface, while any claim about quotient equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness remains outside the packet. □

Theorem 3512.3 (RatCauchyCompletion Real seal boundary)**.** *An accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier may hand its regular rational readback row Q and dyadic tolerance ledger D to the* $\mathsf{Real}^{\uparrow}$ *seal row E only through the displayed continuation row K. The handoff preserves the rational-Cauchy interface for Phase Real and Completion consumers, but it does not assert that* $\mathsf{Real}^{\uparrow}$ *is complete, does not choose a quotient representative, and does not manufacture a limit outside the rows $R, M, S, Q, D, E, H, K, P, N$.*

Proof. The only real-facing coordinate of the carrier is E, and the only route into that coordinate is the displayed continuation row K. Thus a consumer reaches the seal boundary by reading the rational source, tail budget, stream schedule, regular rational readback, and dyadic ledger through the packet's finite route.

The carrier definition also lists the negative boundary: no coordinate stores a selected limit, a quotient representative, a countable-choice schedule, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. Therefore the seal row is a handoff boundary for existing regular rational data, not a proof of completeness of the target interface. □

Theorem 3512.4 (RatCauchyCompletion seal nonescape)**.** *Every accepted* $\mathsf{RatCauchyCompletion}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed rational source row, Cauchy-modulus tail row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise transport, displayed replay, provenance, and local NameCert rows. The packet exports no quotient Cauchy name, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3512.1. Its only Real-facing coordinate is E, and every coordinate before E is one of R, M, S, Q, D together with the structural rows H, K, P, N.

Apply the obligation theorem Definition 3512.2. It exposes the rational source, Cauchy modulus, stream schedule, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming as the whole local NameCert surface.

Now use the handoff theorem Definition 3512.3. The seal row is reachable only through the displayed continuation row after the rational-Cauchy and dyadic rows have been presented.

Therefore a seal read cannot escape to quotient Cauchy names, selected limits, countable-choice schedules, or ambient Real completeness. None of those objects is a carrier coordinate or a local NameCert obligation. □

Closure status for $\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RatCauchyCompletion}^{\uparrow}$ packet is generated from $\mathsf{Rat}^{\uparrow}$ source rows, $\mathsf{CauchyModulus}^{\uparrow}$ tail data, $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3512.1, Definition 3512.2, Definition 3512.3, and Definition 3512.4 bind the carrier, local NameCert obligations, Real-facing seal boundary, and seal nonescape theorem for the rational-Cauchy completion interface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RatCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove $\mathsf{Real}^{\uparrow}$ completeness, quotient Cauchy equality, selected limits, arbitrary Cauchy-filter completion, countable choice, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the NameCert obligation surface, the Real seal boundary, and the rejection of completeness claims outside the displayed rows.

Chapter 3513

A Concrete Naming Certificate for DyadicDensityRealEmbeddingComparison$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicDensityRealEmbeddingComparison$^{\uparrow}$ packet compares the dyadic-density route of chapter 1484 with the constant dyadic real-name route of chapter 467. It keeps the comparison over the finite rows already exposed by DyadicRatCore$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, DyadicEmbedding$^{\uparrow}$, RealDyadicEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$. The packet is a naming certificate for route agreement at displayed windows, not a construction of host real equality, selected dense points, trichotomy, or completeness of Real$^{\uparrow}$.

Definition 3513.1 (Dyadic-density real embedding comparison carrier)**.** *A* DyadicDensityRealEmbeddingCompariso *carrier is a finite* BHist *packet*

$$K = (D, E, E', S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicDensityReal$^{\uparrow}$ *density-witness row whose dyadic source is read through the* DyadicRatCore$^{\uparrow}$ *tolerance ledger.*
2. *E is the* DyadicEmbedding$^{\uparrow}$ *constant-name route from the same dyadic source into a stream, a regular rational readback, and a real seal.*
3. *E' is the* RealDyadicEmbedding$^{\uparrow}$ *comparison row that records the Real-facing read of the embedded dyadic name at the displayed window.*
4. *S is the shared* StreamName$^{\uparrow}$ *finite-window row opened by both the density witness and the embedding route.*
5. *R is the shared* RegSeqRat$^{\uparrow}$ *readback row attached to that window.*
6. *Q is the* Real$^{\uparrow}$ *seal row consumed by the comparison.*
7. *H stores componentwise* hsame *transport for the density row, embedding rows, stream window, regular readback, real seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from the dyadic source to the density witness, from the same source through the embedding route, and from the common window-readback pair to the* Real$^{\uparrow}$ *seal.*
9. *P stores* Pkg *provenance for* DyadicRatCore$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, DyadicEmbedding$^{\uparrow}$, RealDyadicEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$ *row.*
The carrier contains no coordinate for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ BEDC.Derived.DyadicDensityRealEmbeddingComparisonUp.DyadicDensityRealEmbeddingComparisonTasteGa
single carrier alignment

Theorem 3513.2 (Dyadic-density real embedding comparison NameCert obligations)**.** *For every accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *carrier* $K = (D, E, E', S, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$ *surface consists of six finite obligation rows:*

1. *carrier admission for the density, embedding, comparison, window, readback, real-seal, transport, continuation, provenance, and local naming rows;*
2. *dyadic-source agreement, stating that the density row D and the embedding row E read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row;*
3. *window agreement, stating that the comparison opens the same* $\mathsf{StreamName}^{\uparrow}$ *row S for the density read and for the embedded constant name;*
4. *readback agreement, stating that the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *row R is the only rational readback passed to the* $\mathsf{Real}^{\uparrow}$ *seal Q;*
5. *Real-facing comparison, stating that the* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *row E' compares only the displayed density route with the displayed embedding route at the common finite window;*
6. *non-escape, stating that host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are absent from the carrier.*

Lean4 ✓ BEDC.Derived.DyadicDensityRealEmbeddingComparisonUp.DyadicDensityRealEmbeddingComparisonCarrier
namecert obligations

Proof. Project the accepted carrier. The tuple lists exactly the density row, embedding rows, shared stream window, shared regular readback, real seal, transport, continuation, provenance, and local naming rows available to the classifier.

Dyadic-source agreement is read from the common $\mathsf{DyadicRatCore}^{\uparrow}$ source named by the density and embedding routes. The carrier has one source ledger for the comparison, so the embedded constant name cannot use a separate dyadic point.

Window agreement and readback agreement follow from the shared rows S and R. Both routes must open the same finite $\mathsf{StreamName}^{\uparrow}$ window and pass the same $\mathsf{RegSeqRat}^{\uparrow}$ readback before the $\mathsf{Real}^{\uparrow}$ seal is consumed.

The Real-facing comparison is therefore only the row E'. It records agreement of the two displayed routes at the common finite read, while the transport and continuation rows replay the same packet under componentwise hsame.

Finally inspect the carrier inventory. No row names host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, so the six rows exhaust the local NameCert surface. □

Theorem 3513.3 (Dyadic-density real embedding comparison route)**.** *Every accepted* DyadicDensityRealEmbedding
packet compares the density witness route and the constant dyadic embedding route only through the finite spine

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow (\mathsf{DyadicDensityReal}^{\uparrow}, \mathsf{DyadicEmbedding}^{\uparrow}, \mathsf{RealDyadicEmbedding}^{\uparrow}) \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSe}$$

The comparison is a route equality of displayed certificate rows at a common finite window; it does not import host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Start with an accepted packet $K = (D, E, E', S, R, Q, H, C, P, N)$. The density route begins with the $\mathsf{DyadicDensityReal}^{\uparrow}$ row D, and the constant-name route begins with the $\mathsf{DyadicEmbedding}^{\uparrow}$ row E over the same dyadic source.

The comparison row E' is downstream of those two rows. It does not add a new source; it only records the Real-facing comparison after the shared $\mathsf{StreamName}^{\uparrow}$ window S and the shared $\mathsf{RegSeqRat}^{\uparrow}$ readback R have been displayed.

The real seal Q is therefore reached through the same finite read on both sides. Componentwise transport H, continuation replay C, provenance P, and the local name N preserve the packet as a finite BEDC certificate.

Since the carrier provides no row for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, no consumer can replace this finite comparison by an external real-equality or completeness argument. □

Theorem 3513.4 (Dyadic-density real embedding comparison RegSeqRat modulus use)**.** *Every accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *packet reads the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *row through the same finite modulus window before comparing the density witness route with the constant dyadic embedding route. In particular, for* $K = (D, E, E', S, R, Q, H, C, P, N)$*, the window row* S *and readback row* R *are consumed before the comparison row* E' *can export agreement to the* $\mathsf{Real}^{\uparrow}$ *seal* Q*.*

Proof. Project the accepted carrier as $K = (D, E, E', S, R, Q, H, C, P, N)$. The carrier has one shared $\mathsf{StreamName}^{\uparrow}$ row S and one shared $\mathsf{RegSeqRat}^{\uparrow}$ readback row R for the density witness route and the constant dyadic embedding route.

Use the shared-window and readback agreement rows from Definition 3513.2. They force both sides of the comparison to open the same finite window and to pass the same regular rational readback before the Real-facing comparison row is consumed.

Apply Definition 3513.3 to identify the comparison spine ending in the $\mathsf{Real}^{\uparrow}$ seal Q. Since the carrier contains no row for host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, or ambient completeness, the shared $\mathsf{RegSeqRat}^{\uparrow}$ window cannot be replaced by any external Real-side principle. □

Theorem 3513.5 (Dyadic-density real embedding comparison non-escape)**.** *The dependency ledger of an accepted* $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ *carrier is exhausted by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicDensityReal}^{\uparrow}$*,* $\mathsf{DyadicEmbedding}^{\uparrow}$*,* $\mathsf{RealDyadicEmbedding}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*. No accepted comparison consumer may add host real equality, a selected dense point, trichotomy, quotient stream equality, arbitrary limit choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as an implicit dependency.*

Proof. Project the package row P and the local naming row N. They list exactly the finite dependency rows displayed in Definition 3513.1.

Apply Definition 3513.2 for carrier admission, source agreement, window agreement, readback agreement, Real-facing comparison, and non-escape. Then use Definition 3513.3 to identify the only comparison spine consumed by downstream Real-facing readers.

No coordinate remains that could carry an external equality test, a chosen dense point, a trichotomy decision, a quotient-stream equality proof, an arbitrary limit choice, or a completeness

principle. Each such object would therefore be an off-packet dependency, not a row of the accepted comparison certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}$ packet is generated from a DyadicDensityReal density row, a DyadicEmbedding constant-name route, a RealDyadicEmbedding comparison row, shared StreamName and RegSeqRat reads, a Real seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3513.1, Definition 3513.2, Definition 3513.3, and Definition 3513.5 seed the finite comparison packet over dyadic density and dyadic real embedding rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicDensityRealEmbeddingComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, selected dense points, trichotomy, quotient stream equality, arbitrary limit choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, shared-window agreement, shared-readback agreement, Real-facing comparison, and non-escape for the same finite comparison packet.

Chapter 3514

A Concrete Naming Certificate for $\mathsf{CauchyInterleavingComparison}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ packet records the finite comparison surface for two interleaved regular-Cauchy names before a completion-facing consumer reads either name as a real-facing object. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the convergence-modulus route of chapter 1438, the sequence-limit handoff of chapter 1413, and the terminal real seal of chapter 38. The packet supplies a finite comparison certificate for completion consumers; it is not a quotient of Cauchy names, not a selected representative, and not an ambient completeness principle.

Definition 3514.1 (Cauchy interleaving comparison carrier)**.** *A* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, E, O, D, T, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the paired regular-Cauchy source row, storing the two regular rational names whose observations are compared.*
2. *E is the even interleaving window row, exposing the finite observations assigned to the first source.*
3. *O is the odd interleaving window row, exposing the finite observations assigned to the second source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic comparison budget and endpoint error rows.*
5. *T is the shared tail threshold, read before either interleaving window may be used as a comparison witness.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family that displays the finite window requests routed through T.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback comparison row, reconstructing the regular rational comparison from E, O, D, T, W.*

8. *H records componentwise* hsame *transport for the source, window, tolerance, threshold, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each completion-facing comparison request through $S, E, O, D, T, W, Q, H, P$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of Cauchy names, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterleavingComparisonUp.CauchyInterleavingComparisonTasteGate`
`single carrier alignment`

Theorem 3514.2 (Cauchy interleaving comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *carrier $K = (S, E, O, D, T, W, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComp}}$ *surface consists exactly of paired regular-Cauchy source admission, even-window exposure, odd-window exposure, dyadic tolerance exposure, shared tail-threshold exposure, StreamName finite-window exposure, RegSeqRat readback comparison, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 3514.1. Its analytic rows are the paired source S, the even and odd windows E, O, the dyadic ledger D, the shared threshold T, the finite window family W, and the readback comparison row Q.

The NameCert obligations read those rows in the displayed order. The even and odd windows are not independent representatives; they are the two finite faces of the same paired source, controlled by the same dyadic ledger and the same threshold row.

The structural rows H, C, P, N preserve, replay, source, and name the packet. Since the carrier lists no quotient equality, chosen representative, choice selector, trichotomy row, or ambient completeness row, the local certificate surface is exhausted by the displayed finite comparison data. □

Theorem 3514.3 (Cauchy interleaving comparison tail lock)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ *packet must read the shared threshold row T before inspecting either the even window E or the odd window O. The comparison route is*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto E, O \longmapsto Q \longmapsto H, C, P, N,$$

so a downstream $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer receives a single finite tail lock rather than two selected tail representatives.*

Proof. Begin with the NameCert obligations. They place D and T before the window family and before the even–odd observations may be read as comparison evidence.

The $\mathsf{StreamName}^{\uparrow}$ family W then displays the finite requests controlled by the shared threshold. Since both interleaving windows are read through the same T, W pair, a consumer cannot replace the comparison by separate chosen tails for the two sources.

Finally Q repacks the displayed windows as a $\mathsf{RegSeqRat}^{\uparrow}$ readback comparison, and the structural rows preserve that route for $\mathsf{CauchyConvergenceModulus}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. Quotient equality, countable choice, trichotomy, selected representatives, and ambient completeness are absent from the carrier and from the local provenance. □

Falsifiable prediction: A $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ consumer that reads an even or odd interleaving window without first reading the shared threshold T refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, cauchyconvergencemodulus, realsequencelimit, real: no listed sibling supplies the whole packet $K = (S, E, O, D, T, W, Q, H, C, P, N)$, because each sibling contributes only tolerance, finite-window, readback, convergence-modulus, limit-handoff, or seal data while this chapter binds the two interleaved windows to one shared comparison threshold.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterleavingComparison}^{\uparrow}$ packet is generated from a paired regular-Cauchy source, even and odd interleaving windows, a DyadicRatCore tolerance ledger, a shared tail threshold, StreamName finite windows, RegSeqRat readback comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3514.1, Definition 3514.2, and Definition 3514.3 seed the finite comparison route for interleaved regular-Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterleavingComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality, selected representatives, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, even–odd window exposure, shared threshold locking, RegSeqRat readback comparison, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterleavingComparisonUp.

Chapter 3515

A Concrete Naming Certificate for $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ packet records the finite certificate surface for the standard fact that a monotone cofinal subsequence of a regular Cauchy rational name remains regular and presents the same real-name boundary. It consumes the subsequence interface of chapter 381, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal seal surface of chapter 38. The packet is a finite BEDC handoff for Phase Real and Completion consumers; it does not choose a subsequence by countable choice, quotient streams, or invoke ambient completeness.

Definition 3515.1 (RegularCauchySubsequenceStability carrier)**.** *A* $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy packet, carrying the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows and the source finite stream observations.*
2. *T is a monotone-cofinal schedule row, recording the unary reindexing route by which each selected subsequence window is reached from an inspected source window.*
3. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *source and subsequence windows compared by the packet.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying dyadic radii, endpoint comparisons, and the transported regularity budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the subsequence window after transport along T.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary exposed only after the source, schedule, window, dyadic, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the source packet, schedule, windows, tolerance ledger, readback row, seal boundary, continuation routes, provenance, and local certificate.*
8. *C records displayed* Cont *replay of the same finite route.*
9. *P records the* Pkg *provenance over* $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only by $S, T, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a choice-selected infinite subsequence, host sequence equality, host real equality, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityTasteGate`
`single carrier alignment`
Lean4 (definition) `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarri`

Theorem 3515.2 (RegularCauchySubsequenceStability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ *carrier*

$$G = (S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ *surface consists exactly of the source regular Cauchy rows S, the monotone-cofinal schedule T, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows W, the transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *subsequence readback R, the* $\mathsf{Real}^{\uparrow}$ *seal boundary E, the componentwise transport row H, the continuation replay row C, the package provenance row P, and the local naming row N.*

Proof. Carrier exposure is projection from Definition 3515.1. Each row in the statement is one displayed coordinate of the finite packet, and the definition lists no hidden coordinate for an externally selected subsequence or completion object.

The source and schedule obligations are S and T. The finite-window obligation is W, while the dyadic tolerance and regular rational readback obligations are D and R. These are the rows that express the usual cofinal-subsequence regularity argument inside the BEDC packet.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections yields the displayed $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$ surface, with no quotient stream, host equality, or ambient completeness datum. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarrier`
`namecert obligations`

Theorem 3515.3 (RegularCauchySubsequenceStability Real seal boundary)**.** *An accepted* RegularCauchySubseque
carrier may expose its $\mathsf{Real}^{\uparrow}$ *seal row E only after the source regular Cauchy data S, the monotone-cofinal schedule T, the finite stream windows W, the transported dyadic tolerance ledger D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R have been displayed through H and C. Thus cofinal subsequence transport preserves the real-name seal surface as finite BEDC data and does not export quotient streams, a choice-selected subsequence, host sequence equality, or ambient real completeness.*

Proof. Start with an accepted carrier. The seal row E is not a source coordinate; it is reached after the source, schedule, window, dyadic, readback, transport, continuation, provenance, and naming rows have been projected.

Apply the NameCert obligation surface of Definition 3515.2. It identifies the monotone-cofinal schedule, the inspected and selected finite windows, the transported dyadic tolerance ledger, and the regular rational readback as the only data available before the seal boundary.

The ordinary constructive subsequence argument is therefore recorded as carrier projection followed by cofinal schedule transport and finite-window replay. Since the classifier has no coordinate for quotient streams, choice-selected subsequences, host sequence equality, host real equality, or a completeness principle, none of those data can escape through the $\mathsf{Real}^{\uparrow}$ seal row. □

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceStabilityUp.RegularCauchySubsequenceStabilityCarrier`
`real seal boundary`

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$ packet is generated from source regular Cauchy rows, a monotone-cofinal schedule, finite $\mathsf{StreamName}^{\uparrow}$ windows, transported $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3515.1, Definition 3515.2, and Definition 3515.3 bind the carrier, local NameCert obligations, and Real-facing seal boundary for cofinal regular Cauchy subsequence stability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove a public standard bridge, quotient stream equality, host sequence equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the NameCert obligation surface, the Real seal boundary, and the nonescape boundary for data outside the displayed rows.

DNA hash: `aggregate=sha256:5f957223fdebb99cd0a4300c035bf3311aafd9888c8f99810e0304aa1e677f18;statement=sha256:a7`

Chapter 3516

A Concrete Naming Certificate for $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$

Following Specker's constructive counterexample tradition and Bishop–Bridges constructive analysis, the $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ packet isolates the reusable nonlocated-supremum obstruction behind monotone bounded rational streams. It sits after $\mathsf{SpeckerSequence}^{\uparrow}$ in chapter 1406 and reads only the finite windows supplied by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{LocatedCut}^{\uparrow}$ in chapter 126, chapter 301, chapter 291, chapter 38, and chapter 404. The packet names the obstruction surface; it does not supply Real completeness, a located supremum, trichotomy, quotient stream equality, countable choice, or selected limits.

Definition 3516.1 (Specker sequence boundary carrier)**.** *A* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, M, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SpeckerSequence}^{\uparrow}$ *source row naming the monotone bounded rational stream and its nonlocated-supremum boundary.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite source windows inspected by the boundary packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound ledger read from the same selected windows.*
4. *M is the monotone and bounded obstruction ledger recording the displayed comparison and bound data that can be read before any real-facing seal.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational windows certified by S, W, D, M.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{LocatedCut}^{\uparrow}$ *refusal row; it records that the displayed windows may be sealed as finite real readback data, while a located-cut or supremum witness remains external.*
7. *H stores componentwise* hsame *transport for the source, schedule, dyadic ledger, obstruction ledger, regular readback, seal-or-refusal row, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from S, W into D, from D into M, from M into R, and from R into the seal-or-refusal row L.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, D, M, R, L, H, C, P, N$. It has no coordinate for a located supremum, a selected limiting real, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SpeckerSequenceBoundaryUp.SpeckerSequenceBoundaryTasteGate single carrier alignment`

Theorem 3516.2 (Specker sequence boundary NameCert obligations)**.** *For every accepted* SpeckerSequenceBounda *carrier $B = (S, W, D, M, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$ *surface consists exactly of source admission through S, finite schedule selection through W, dyadic tolerance and bound admission through D, monotone bounded obstruction admission through M, regular rational readback through R, the* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{LocatedCut}^{\uparrow}$ *refusal row L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3516.1. The analytic rows are exactly the Specker source, stream schedule, dyadic ledger, monotone bounded obstruction ledger, regular readback, and seal-or-refusal row.

The structural rows H, C, P, N transport, replay, source, and name those same coordinates. Since the carrier contains no further analytic field, the listed rows exhaust the local NameCert obligations.

The missing fields are also part of the boundary. A located supremum, quotient stream equality, countable-choice selector, trichotomy branch, or selected limit would require an additional coordinate, and no such coordinate is present. □

Theorem 3516.3 (Specker sequence boundary Cauchy readback)**.** *Every Cauchy-facing read from an accepted* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier factors through*

$$S, W \longmapsto D \longmapsto M \longmapsto R \longmapsto L$$

together with H, C, P, N. The row R exports only the $\mathsf{RegSeqRat}^{\uparrow}$ *window readback certified by the displayed schedule and dyadic ledger, and the row L exports only a finite* $\mathsf{Real}^{\uparrow}$ *seal or a* $\mathsf{LocatedCut}^{\uparrow}$ *refusal boundary.*

Proof. Start with the source route. The $\mathsf{SpeckerSequence}^{\uparrow}$ row S and the $\mathsf{StreamName}^{\uparrow}$ schedule W determine which finite rational windows are inspected.

Read the dyadic and obstruction ledgers next. The row D supplies the tolerance and bound data for those windows, while M records the monotone bounded obstruction information available at the same finite stages.

The regular readback row R is downstream of those data, and the seal-or-refusal row L is downstream of R. Transport, replay, provenance, and local naming through H, C, P, N preserve this route, so no located supremum or selected limiting real is produced by the Cauchy readback. □

Theorem 3516.4 (Specker sequence boundary nonescape)**.** *Every consumer of an accepted* SpeckerSequenceBounda *carrier receives only the finite rows $S, W, D, M, R, L, H, C, P, N$. Consequently the packet may reuse the* $\mathsf{SpeckerSequence}^{\uparrow}$ *nonlocated-supremum obstruction and its* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{LocatedCut}^{\uparrow}$ *interfaces, but it exports no* $\mathsf{Real}^{\uparrow}$ *completeness principle, located supremum, trichotomy, quotient stream equality, countable-choice selector, or selected limit.*

Proof. Use Definition 3516.2 to identify the complete public surface of the accepted packet. All analytic reads pass through the six displayed rows S, W, D, M, R, L.

Use Definition 3516.3 to order those rows. The regular readback and seal-or-refusal row appear only after finite windows, dyadic data, and monotone bounded obstruction data have been displayed.

Finally read the structural rows. Componentwise transport, displayed replay, provenance, and local naming do not add analytic coordinates, so none of the excluded completeness, choice, equality, order-decision, or selected-limit principles can escape from the packet. □

Theorem 3516.5 (Specker sequence boundary LPO separation)**.** *Every consumer reading an accepted* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *carrier* $B = (S, W, D, M, R, L, H, C, P, N)$ *through the displayed rows* $S, W, D, M, R, L, H, C, P, N$ *receives only the Specker source, finite schedule, dyadic tolerance ledger, monotone bounded obstruction ledger, regular rational readback, seal-or-refusal boundary, transport, continuation, provenance, and local naming data. Such a consumer cannot extract* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$, $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, *countable choice, trichotomy, or a located supremum unless an external row supplies that principle.*

Proof. Project the carrier of Definition 3516.1. The only analytic rows available to the consumer are S, W, D, M, R, L, and the only structural rows are H, C, P, N.

Apply the Cauchy readback theorem Definition 3516.3. It orders the read through the Specker source, finite schedule, dyadic ledger, monotone bounded obstruction ledger, regular readback, and seal-or-refusal row before any Real-facing conclusion is public.

Now apply nonescape from Definition 3516.4. The carrier has no coordinate for located supremum construction, trichotomy, quotient stream equality, countable choice, selected limits, or ambient Real completeness.

The neighbouring omniscience packets of chapter 5945 and chapter 1977 are therefore only comparison siblings. Without an external row naming one of those principles, the Specker boundary exports the obstruction route and not LPO, LLPO, countable choice, trichotomy, or a located supremum. □

Falsifiable prediction: Every $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ Cauchy-facing read factors through a SpeckerSequence source row, StreamName schedule, DyadicRatCore tolerance ledger, monotone bounded obstruction ledger, RegSeqRat readback, Real seal or LocatedCut refusal row, transport, continuation, provenance, and local naming.

Independence witness: speckersequence, streamname, regseqrat, dyadicratcore, real, located-cut: $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nonlocated-supremum obstruction route, regular readback, seal-or-refusal boundary, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ packet is generated from a $\mathsf{SpeckerSequence}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and bound rows, a monotone bounded obstruction ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal or $\mathsf{LocatedCut}^{\uparrow}$ refusal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3516.1, Definition 3516.2, Definition 3516.3, Definition 3516.4, and Definition 3516.5 bind the finite nonlocated-supremum obstruction certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpeckerSequenceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, located supremum construction, trichotomy, quotient stream equality, countable choice, selected limits, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, Cauchy readback, nonescape, and TasteGate field faithfulness for BEDC.Derived.SpeckerSequenceBoundaryUp.

Chapter 3517

A Concrete Naming Certificate for $\mathsf{BishopOrderCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet records the order-completion face of the real-number construction as finite BEDC NameCert data. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the comparison boundary of chapter 1730, the lower and upper cut rows of chapter 1446 and chapter 2087, and the terminal real-name seal of chapter 38. The packet names a constructive order-completion interface; it is not a quotient-cut construction, selected supremum principle, trichotomy oracle, host real equality test, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3517.1 (Bishop order-completion carrier)**.** *A* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, E, B, L, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison-budget ledger used by all lower, upper, and real-seal requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at those dyadic requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R have been displayed.*
5. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison row, read through the located-real comparison surface and tied to the same windows and dyadic budgets.*
6. *L is the LowerReal-style located lower-cut ledger, recording only the displayed inhabited lower observations and downward-use rows available to the packet.*
7. *U is the UpperReal-style located upper-cut ledger, recording only the displayed upper observations and upward-use rows available to the packet.*
8. *H records componentwise* hsame *transport for the tolerance, window, readback, Real seal, comparison, lower-cut, upper-cut, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each order-completion request through $D, S, R, E, B, L, U, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopOrderCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $D, S, R, E, B, L, U, H, C, P, N$. It has no coordinate for quotient cuts, host supremum objects, selected representatives, trichotomy, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopOrderCompletionUp.TasteGate.BishopOrderCompletionTasteGate single carrier alignment`

Theorem 3517.2 (Bishop order-completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier $O = (D, S, R, E, B, L, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, finite StreamName window exposure, RegSeqRat readback, Real seal handoff, BishopRealComparison exposure, LowerReal-style lower-cut exposure, UpperReal-style upper-cut exposure, componentwise transport, displayed continuation replay, package provenance, and local naming. Every order-completion consumer factors through*

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto B \longmapsto (L, U) \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3517.1. Its real-facing rows are the dyadic tolerance ledger, finite stream windows, regular rational readback, and terminal Real seal.

Read the order rows after the same real-facing route. The comparison row B uses the displayed windows and dyadic budgets, while L and U record the lower and upper located-cut observations available to the packet.

The structural rows then preserve this ordered route. The hsame, Cont, Pkg, and local NameCert rows transport, replay, source, and name the route, and the refused quotient, host-supremum, selected, trichotomy, host-equality, choice, and ambient-completeness objects have no carrier coordinate. □

Theorem 3517.3 (Bishop order-completion located-cut handoff)**.** *Every located lower-or-upper cut read exported by an accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier is downstream of the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{BishopRealComparison}^{\uparrow}$ *comparison row. The handoff exposes the LowerReal-style and UpperReal-style ledgers only as the finite packet $D, S, R, E, B, L, U, H, C, P, N$.*

Proof. Start with the NameCert obligations. They place D, S, R, E before any order-facing row can be consumed.

Next read the comparison boundary. The row B ties lower and upper observations to the same finite windows and dyadic budgets, so the lower and upper ledgers cannot be inspected as independent host cuts.

Finally project L and U. These rows are named only inside the carrier and are preserved by H, C, P, N, so the handoff contains no quotient-cut equality, selected representative, trichotomy branch, host supremum, or ambient completeness coordinate. □

Theorem 3517.4 (Bishop order-completion Cauchy-cut comparison)**.** *For every accepted* $\mathsf{BishopOrderCompletion}^{\uparrow}$ *carrier, the comparison between regular-Cauchy real data and located lower–upper cut data is a single finite route through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BishopRealComparison}^{\uparrow} \longmapsto (\mathsf{LowerReal}^{\uparrow}, \mathsf{UpperR}$$

Thus a Cauchy-facing consumer and a cut-facing consumer read the same packet-local rows $D, S, R, E, B, L, U, H, C$ rather than two unrelated completion principles.

Proof. Read the Cauchy-facing side first. The rows D, S, R, E display the tolerance, window, readback, and Real seal route used by regular-Cauchy consumers.

Read the cut-facing side through the comparison row. The row B is downstream of the same route and supplies the finite comparison request before L and U can expose lower and upper located observations.

The two faces are therefore synchronized by the structural rows. Transport, continuation, provenance, and local naming preserve one packet-local route, while quotient-cut completion, selected supremum, host real equality, trichotomy, countable choice, and ambient Real completeness remain outside the carrier. □

Falsifiable prediction: A claimed $\mathsf{BishopOrderCompletion}^{\uparrow}$ consumer that reaches a lower-cut or upper-cut row without the displayed DyadicRatCore, StreamName, RegSeqRat, Real, and BishopRealComparison rows refutes the carrier surface.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishoprealcomparison, lowerreal, upperreal: no listed sibling supplies the whole $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet, because each sibling contributes only tolerance, window, readback, seal, comparison, or one-sided cut data while this chapter binds all of them with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopOrderCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RealUp seal, BishopRealComparison row, LowerReal-style and UpperReal-style located cut rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3517.1, Definition 3517.2, Definition 3517.3, and Definition 3517.4 seed the finite order-completion interface joining regular-Cauchy real data to located lower and upper cut rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopOrderCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient cut completion, selected suprema, trichotomy, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, regular readback, RealUp seal handoff, comparison-row stability, located cut handoff, Cauchy-cut comparison, transport, replay, provenance, and local naming for BEDC.Derived.BishopOrderCompletionUp.

Chapter 3518

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ packet records the finite precision-to-tail schedule that lets a Cauchy name serve L10 Real and Completion consumers without importing ambient completeness. It sits over the generic modulus schedule of chapter 292, the fast threshold discipline of chapter 1781, the request-to-tail realizer of chapter 1918, the finite-window interface of chapter 126, the regular rational route of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet supplies one displayed schedule row before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads a seal.

Definition 3518.1 (Effective Cauchy modulus carrier)**.** *An* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, M, T, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose observations are opened only through finite windows.*
2. *D is the dyadic precision ledger consumed by the tail estimates and regular rational readbacks.*
3. *M is the effective precision-to-tail modulus row, assigning each displayed precision request a finite tail index.*
4. *T is the tail-index comparison row showing that the selected index is stable under stronger displayed precision requests.*
5. *W is the finite tail-window row opened from S at the index selected by M.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W under the dyadic ledger D.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row, exposed only after S, D, M, T, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for the stream, dyadic ledger, effective modulus, tail comparison, window, readback, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a precision request through $S, D, M, T, W, Q, E_H, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyModulusRealizer}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, M, T, W, Q, E, H, C, P, N$. It has no coordinate for quotient limits, countable choice, host convergence, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3518.2 (Effective Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *carrier*

$$X = (S, D, M, T, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for the stream source, dyadic precision ledger, effective modulus row, tail comparison row, finite window, regular readback, Real handoff, transport, replay, provenance, and local naming rows;*
2. *schedule classification, requiring M to be the only precision-to-tail row read by a downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer;*
3. *monotone tail-index stability, requiring T to certify that stronger displayed precision requests never require a weaker tail boundary than the row already exposed by M;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal compatibility, requiring W and Q to be read before the handoff row E exposes a terminal Real seal;*
5. *ledger refusal for quotient limits, countable choice, host convergence, trichotomy, selected representatives, and ambient completeness as local certificate data.*

No effective Cauchy modulus obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are exactly the stream source, dyadic ledger, effective modulus, tail comparison, finite window, regular readback, and Real handoff rows.

The schedule classifier is the row M. Since every consumer reaches a tail window only after reading M, no hidden convergence selector or external completeness principle can choose the boundary.

The monotonicity information is stored in T. It relates stronger displayed precision requests to tail indices already chosen by M, so transport through H and replay through C preserve the same schedule.

Finally, the readback and seal are downstream of the finite window. The rows Q and E can expose regular rational and Real-facing data only after S, D, M, T, W have been displayed, and the refused objects have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.EffectiveCauchyModulusUp.EffectiveCauchyModulusCarrier namecert obligations`

Theorem 3518.3 (Effective Cauchy modulus seal route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *packet factors through*

$$S, D, M, T \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

The route exports a displayed finite tail schedule and does not export quotient limits, countable choice, host convergence, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier coordinates in order. A precision request first sees the stream and dyadic rows S, D, then the effective modulus and tail-stability rows M, T.

Only after those rows are displayed can the finite window W be opened. The regular rational readback Q and terminal handoff E_H are downstream of that finite window.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the refused principles, the exported object remains the finite schedule-to-window packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ packet is generated from a StreamName source row, a dyadic precision ledger, an effective precision-to-tail modulus row, a monotone tail-index comparison row, a finite tail-window row, a RegSeqRat readback, a RealUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3518.1, Definition 3518.2, and Definition 3518.3 seed the finite schedule object consumed by L10 StreamName, RegSeqRat, and Real routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close RealUp completeness, quotient limits, countable choice, host convergence, trichotomy, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, schedule classification, monotone tail-index stability, RegSeqRat and RealUp seal compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchyModulusUp.

Chapter 3519

A Concrete Naming Certificate for $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$

The $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ packet realizes the MetaCIC audit-map frontier of section 9376.7 as finite BEDC rows. It sits next to the redex frontier, normalization frontier, confluence-audit packet, and decidable-boundary anchors already used by the MetaCIC vision, but it names the smaller object needed by a candidate-mediated reducer: a displayed finite choice surface for which redex rows may be inspected, which normalization and confluence-audit rows are available, and which boundaries remain undecided. The packet consumes the redex-row surface of chapter 1008, the candidate-normalization boundary of chapter 892, the local critical-pair fork surface of chapter 2382, and the beta critical-pair row of chapter 881. The normal-form endpoint row is consumed through chapter 740, while the redex-choice packet keeps subject reduction and global consistency outside the carrier. The packet does not assert global normalization, Church–Rosser, subject reduction for all terms, or a complete MetaCIC semantics.

Definition 3519.1 (MetaCIC redex-choice frontier carrier)**.** *A* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, N, F, D, B, H, K, L, C, P, Q)$$

with the following displayed rows:

1. *A is the MetaCIC redex row, listing the candidate redexs exposed by the local term-and-context window.*
2. *N is the normalization-frontier row, recording which candidates have a displayed bounded reduction trace.*
3. *F is the confluence-audit row, recording local fork and join checks for the same candidate set.*
4. *D is the decidable-boundary row, recording which redex choices are classified by finite inspection and which remain outside the packet.*
5. *B is the* BHist *replay ledger for the term window, candidate row, trace row, fork row, boundary row, and local naming row.*
6. *H records componentwise* hsame *transport for the redex, normalization, confluence-audit, decidability, replay, continuation, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a term-window request through A, N, F, D, B, H, P, Q.*

8. *L is the* Pkg *provenance over* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$*, MetaCIC redex rows, normalization-frontier rows, confluence-audit rows, decidable-boundary rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *C records the candidate-choice classifier: it exposes only the redex choices whose A, N, F, D rows are present in the packet.*
10. *P records the local package boundary for Nontrivial and FieldFaithful TasteGate instances to be checked by a later Lean development over the displayed carrier.*
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, N, F, D, B, H, K, L, C, P, Q$. It has no coordinate for global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, or a completed MetaCIC model.

Lean4 ✓ `BEDC.Derived.MetacicRedexChoiceFrontierUp.MetacicRedexChoiceFrontierTasteGate single carrier alignment`

Theorem 3519.2 (MetaCIC redex-choice frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier*

$$R = (A, N, F, D, B, H, K, L, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier exposure for MetaCIC redex rows, normalization-frontier traces, confluence-audit fork rows, decidable-boundary classifiers, replay, transport, continuation, provenance, candidate-choice, TasteGate boundary, and local naming rows;*
2. *redex-choice classification, requiring the candidate-choice row C to inspect only redexs displayed in A and only boundary outcomes displayed in D;*
3. *normalization-frontier compatibility, requiring every candidate exported by C to carry either the bounded trace row in N or the explicit boundary refusal in D;*
4. *confluence-audit compatibility, requiring every fork exported by C to use the same candidate set as A and the same local fork rows as F;*
5. *TasteGate preparation, requiring the packet to expose the Nontrivial and FieldFaithful instance boundary in P without claiming those Lean instances as paper-side evidence;*
6. *ledger refusal for global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, and a completed MetaCIC model.*

No redex-choice frontier obligation is stored outside these rows.

Proof. Project the accepted carrier. The term-facing data are the redex row A, the normalization-frontier row N, the confluence-audit row F, the decidable-boundary row D, and the candidate-choice row C.

The candidate classifier cannot inspect a redex outside A, because C is downstream of the displayed redex row. It cannot treat an undecided boundary as decided, because every exported choice must pass through D.

The normalization and confluence rows are tied to the same candidate set. A selected candidate therefore carries the bounded trace information from N and the local fork information from F exactly when those rows are displayed by the packet.

The TasteGate boundary row P records the intended Nontrivial and FieldFaithful instance surface for Lean without adding a marker or treating the instances as verified. The refused global principles have no carrier coordinate, so the NameCert surface remains finite. □

Theorem 3519.3 (MetaCIC redex-choice frontier nonescape)**.** *Every reducer-facing read from an accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *packet factors through*

$$A, N, F, D \longmapsto C \longmapsto B, H, K, L, P, Q.$$

The exported object is a finite candidate-mediated frontier for local redex choice, not global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, or a completed MetaCIC model.

Proof. Start with the displayed redex, normalization, confluence-audit, and decidable-boundary rows. These rows are the only inputs to the candidate-choice classifier.

The classifier C exports only the candidates that survive those displayed rows. Replay through B, transport through H, continuation through K, provenance through L, TasteGate boundary through P, and naming through Q preserve the same finite route.

Since no global normalization theorem, global confluence theorem, unrestricted redex selector, quotient syntax object, host evaluator soundness proof, or completed MetaCIC model appears in the carrier, the reducer-facing read cannot escape the finite frontier. □

Theorem 3519.4 (MetaCIC redex-choice frontier candidate-boundary exhaustion)**.** *For every accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *carrier*

$$R = (A, N, F, D, B, H, K, L, C, P, Q),$$

each exported candidate choice is exhausted by the displayed A, N, F, D, C *rows before replay, provenance, transport, TasteGate preparation, and local naming are read through* B, H, K, L, P, Q*. No accepted route can import global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, or completed MetaCIC semantics.*

Proof. Begin with the candidate-choice classifier C. By Definition 3519.1, it exports only choices whose redex row A, normalization-frontier row N, confluence-audit row F, and decidable-boundary row D are already displayed.

Apply the nonescape theorem Definition 3519.3. After A, N, F, D and C have been consumed, replay and provenance can only preserve the same finite route through B, H, K, L, P, Q.

The refused global objects have no carrier coordinate. Thus the boundary is exhausted at the candidate-mediated finite rows and remains separate from global normalization, global confluence, unrestricted redex selection, quotient syntax, host evaluator soundness, and completed MetaCIC semantics. □

Theorem 3519.5 (MetaCIC redex-choice candidate-normalization handoff)**.** *Every exported redex choice from an accepted* $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ *packet consumed by a normalization-window or confluence-audit reader must pass through*

$$A, N, F, D, C \longmapsto B, H, K, L, P, Q.$$

The row N *supplies only displayed candidate-normalization evidence, the row* F *supplies only local fork information, and the row* D *preserves undecided boundary rows. The packet refuses global normalization, Church–Rosser, subject reduction for all terms, quotient syntax, host evaluator soundness, and a completed MetaCIC semantics.*

Proof. Start with the NameCert obligations of Definition 3519.2. Redex-choice classification forces C to inspect only redexs in A and boundary outcomes in D, while normalization-frontier and confluence-audit compatibility tie exported candidates to the displayed N and F rows.

The normalization-window reader therefore receives only candidate-normalization evidence from N. The confluence-audit reader receives only local fork information from F, and undecided rows remain recorded by D rather than being converted into global decisions.

Apply Definition 3519.4. Each exported candidate is already exhausted by A, N, F, D, C before replay, transport, continuation, provenance, TasteGate preparation, and local naming are read through B, H, K, L, P, Q.

Finally use Definition 3519.3. The refused global principles and host-side objects have no carrier coordinate, so the handoff cannot assert global normalization, Church–Rosser, subject reduction for all terms, quotient syntax, host evaluator soundness, or a completed MetaCIC semantics. □

Falsifiable prediction: A $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ reducer-facing read factors through displayed MetaCIC redex rows, normalization-frontier traces, confluence-audit fork rows, decidable-boundary classifiers, candidate-choice rows, BHist replay, hsame transport, Cont replay, Pkg provenance, TasteGate boundary rows, and local NameCert data.

Independence witness: metacic-redex, metacic-normalization-frontier, metacic-confluence-audit, metacic-decidable-boundary: $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ is not bijective to any listed anchor because it jointly carries candidate-choice classification, shared redex-boundary routing, Nontrivial and FieldFaithful TasteGate boundaries, replay, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}$ packet is generated from MetaCIC redex rows, normalization-frontier rows, confluence-audit rows, decidable-boundary rows, BHist replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, a candidate-choice classifier, Nontrivial and FieldFaithful TasteGate boundary rows, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3519.1, Definition 3519.2, Definition 3519.3, and Definition 3519.4 seed the finite candidate-mediated redex-choice frontier for MetaCIC audit maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicRedexChoiceFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, global confluence, unrestricted redex selection, subject reduction for all terms, quotient syntax, host evaluator soundness, a completed MetaCIC model, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, redex-choice classification, normalization-frontier compatibility, confluence-audit compatibility, Nontrivial and FieldFaithful TasteGate instances, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MetacicRedexChoiceFrontierUp.

Chapter 3520

A Concrete Naming Certificate for $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ packet records the finite route by which a displayed finite coproduct of metric sources is read before any completion-facing or real-facing consumer may inspect the completed object. It sits after the metric-completion surface of chapter 1506, the finite sum carrier of chapter 35, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite coproduct completion handoff, not a quotient completion equality, not a selected limit, not a countable-choice selector, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3520.1 (Metric completion finite-coproduct carrier)**.** *A* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{MCFC}(I, S, M, W, R, A, J, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row naming the admitted summands.*
2. *S is the* $\mathsf{Sum}^{\uparrow}$ *coproduct row, recording injections, finite case analysis, and the branch classifier.*
3. *M is the family of* $\mathsf{MetricCompletion}^{\uparrow}$ *source rows, one for each admitted branch.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row used when a branch request is inspected by a completion consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the branch-local tolerance and distance data.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W and R have fixed the requested branch data.*
7. *J is the coproduct-completion join row, stating that every accepted consumer first selects a displayed finite branch and then reads the corresponding completion route.*
8. *H records componentwise* hsame *transport for index, sum, completion, window, readback, seal, join, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a coproduct-completion request through I, S, M, W, R, A, J.*
10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{Sum}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, S, M, W, R, A, J, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, infinite coproducts, countable choice, host case analysis outside S, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionFiniteCoproductUp.MetricCompletionFiniteCoproductTasteGate`
`single carrier alignment`

Theorem 3520.2 (Metric completion finite-coproduct NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *carrier* $\mathsf{MCFC}(I, S, M, W, R, A, J, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index row,* $\mathsf{Sum}^{\uparrow}$ *coproduct row, branchwise* $\mathsf{MetricCompletion}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, coproduct-completion join, componentwise transport, displayed replay, provenance, and local naming.*

Every obligation is branch-local after the finite case split in S, and every public read must pass through W, R, A before it may expose a completed coproduct branch. No obligation is stored in quotient completion equality, selected representatives, countable choice, infinite coproduct data, host case analysis, or ambient completeness.

Proof. Carrier admission is projection from Definition 3520.1. The tuple displays the finite index row, sum row, branchwise completion rows, numerical readback rows, join row, and structural support.

The finite case split is the row S. Once a branch is selected, the corresponding completion route in M is the only completion-facing source available to a consumer.

The readback rows W, R, A then bind finite observation windows, regular rational tolerance data, and the terminal Real seal to that same branch. The join row J records that this branch-local route is the public coproduct-completion handoff.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded quotient, choice, host, or ambient-completeness objects, the stated rows exhaust the NameCert obligations. □

Theorem 3520.3 (Metric completion finite-coproduct universal route)**.** *Every consumer that reads a finite coproduct after metric completion, or metric completion after a finite coproduct, factors through the displayed route*

$$\mathsf{Sum}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The exported object is the branch classifier named by $I, S, M, W, R, A, J, H, C, P, N$, not a quotient-level universal property and not a selected completed representative.

Proof. Start with the finite index and coproduct rows I, S. They are the only rows that select a branch, so any universal read begins with the displayed finite case split.

For the chosen branch, project the corresponding $\mathsf{MetricCompletion}^{\uparrow}$ row from M. The carrier offers no completed branch outside this family.

Now pass through W, R, A. These rows provide the finite stream window, regular rational readback, and real seal used by the branch consumer.

Apply Definition 3520.2. The join, transport, replay, provenance, and naming rows keep the same route visible, so the consumer receives only the finite universal route and cannot import a quotient completion isomorphism or selected representative. □

Theorem 3520.4 (Metric completion finite-coproduct nonescape)**.** *No accepted* $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ *consumer can escape from the finite branch route into quotient completion equality, selected limits, countable choice, an infinite coproduct, host-side case analysis, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exposed claim is read through the same finite packet rows*

$$I, S, M, W, R, A, J, H, C, P, N.$$

Proof. Use Definition 3520.3. It places every completion-coproduct consumer behind the finite branch classifier and the branch-local completion rows.

The numerical boundary is fixed by W, R, A. These rows provide only finite windows, regular rational readback, and a Real seal for the displayed branch.

The support rows J, H, C, P, N join, transport, replay, source, and name the same packet. Because no excluded object occurs as a coordinate of Definition 3520.1, no consumer can export it. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}$ packet is generated from a finite index row, a SumUp coproduct row, branchwise MetricCompletionUp rows, StreamName windows, RegSeqRat readback, a RealUp seal, a coproduct-completion join row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3520.1, Definition 3520.2, Definition 3520.3, and Definition 3520.4 seed the finite coproduct completion route over MetricCompletionUp, SumUp, StreamNameUp, RegSeqRatUp, RealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFiniteCoproduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, infinite coproducts, host case analysis, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite branch classification, branchwise completion routing, StreamName and RegSeqRat readback, Real seal exactness, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionFiniteCoproductUp.

Chapter 3521

A Concrete Naming Certificate for $\mathsf{LatentCompletion}^{\uparrow}$

The $\mathsf{LatentCompletion}^{\uparrow}$ packet realizes the finite monotone-path completion witness of Definition 9462.1836 as BEDC NameCert data. It records observed module states, latent prefix states, target-monotone edges, exception rows, and a completion-choice ledger before any consumer may claim coverage of the module-state exception geometry. The observed module-state ledger is consumed through chapter 3470, chapter 3508, and chapter 3478 before the latent-prefix exception rows are read. The packet is a finite completion surface for the two missing monotone squares of Definition 9462.1837; it is not an empirical biological transition model, not a host graph-completion oracle, and not a global state-space completeness claim.

Definition 3521.1 (Latent completion carrier)**.** *A* $\mathsf{LatentCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{LatentCompletion}(O, E, L, D, M, K, H, C, P, N)$$

with the following displayed rows:

1. *O is the observed module-state ledger containing the sampled endpoints and their module-value rows.*
2. *E is the exception ledger for the target-monotone failures represented by table identifiers* 9*,* 14*, and* 28*.*
3. *L is the latent-state row containing one deep-branch prefix and one partial-boundary prefix from Definition 9462.1836.*
4. *D is the directed rewrite-edge row for target-monotone one-module changes.*
5. *M is the monotone-path coverage row verifying that every observed endpoint is reachable after the latent rows are admitted.*
6. *K is the exception-kill row recording that the two latent rows cover the two independent missing squares and no third exception branch is created.*
7. *H records componentwise* hsame *transport for observed states, latent states, rewrite edges, coverage rows, exception rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of a coverage request through O, E, L, D, M, K.*
9. *P is the* Pkg *provenance over* $\mathsf{LatentCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows, with the module-state source tied to Definition 9462.1836.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, E, L, D, M, K, H, C, P, N$. It has no coordinate for a continuous biological transition, a complete empirical state space, a graph oracle, unbounded latent search, or a module-state equality principle outside the displayed rows.

Theorem 3521.2 (Latent completion NameCert obligations)**.** *For every accepted* $\mathsf{LatentCompletion}^{\uparrow}$ *carrier* $\mathsf{LatentCompletion}(O, E, L, D, M, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for observed module states, exception rows, latent-prefix rows, directed rewrite edges, monotone-path coverage, exception-kill accounting, componentwise transport, displayed replay, provenance, and local naming.*

Every accepted coverage read must identify the two independent missing squares from E, consume the two latent rows in L, rebuild the directed rewrite graph through D, and then expose coverage through M, K, H, C, P, N. No obligation is stored in a host graph-completion oracle, a biological-transition claim, an unbounded latent search, or a complete empirical state space.

Lean4 ✓ `BEDC.Derived.LatentCompletionUp.LatentCompletionTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3521.1. The tuple displays the observed ledger, exception ledger, latent rows, rewrite edges, coverage row, exception-kill row, and structural support.

The exception analysis is read from E and L. The deep-branch exception rows require one deep prefix, and the partial-boundary exception row requires one partial-boundary prefix.

The coverage route is D, M, K. The directed rewrite row rebuilds target-monotone one-module steps, the coverage row records reachability of the observed endpoints, and the exception-kill row records that no third missing square remains.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no graph oracle, biological transition row, unbounded search row, or complete empirical state-space row, the stated surface exhausts the NameCert obligations. □

Theorem 3521.3 (Latent completion monotone-path coverage)**.** *Every accepted* $\mathsf{LatentCompletion}^{\uparrow}$ *packet covers the finite monotone-path exception geometry by routing each observed endpoint through the displayed rewrite graph after adjoining exactly one deep-branch latent prefix and exactly one partial-boundary latent prefix. The public coverage claim is the finite route*

$$O, E, L, D, M, K, H, C, P, N$$

and it exports no stronger temporal, mechanistic, or empirical completeness claim.

Proof. Start with the observed ledger O and exception ledger E. They identify the endpoints and the three target-monotone failures that must be covered.

Use the latent row L. It supplies one deep-branch prefix for the table-9 and table-14 square and one partial-boundary prefix for the table-28 square.

Now apply the rewrite-edge and coverage rows D, M. These rows rebuild the finite directed graph and record reachability of every observed endpoint from the standard state.

The exception-kill and support rows K, H, C, P, N preserve and name the same coverage route. Since no temporal mechanism or empirical-completeness coordinate is present, the exported conclusion is exactly monotone-path coverage of the finite module-state packet. □

Falsifiable prediction: If either the deep-branch latent prefix or the partial-boundary latent prefix is removed from an accepted $\mathsf{LatentCompletion}^{\uparrow}$ packet, at least one of the table-9, table-14, or table-28 monotone-path exceptions remains uncovered.

Independence witness: six-boundary-tile-coverage, dna-boundary-trigger-closure, window-coverage-trigger-algebra, choice-window-sequence, metacic-redex-choice-frontier: none is bijective to $\mathsf{LatentCompletion}^{\uparrow}$ because this packet binds observed module states, two latent prefix choices, directed rewrite edges, monotone-path coverage, and exception-kill accounting in one NameCert surface.

The Lean follow-up supplies Nontrivial and FieldFaithful TasteGate instances for the accepted latent-completion carrier, so the formal route can distinguish nonempty coverage from faithful preservation of the module-state fields without adding new paper-side rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LatentCompletion}^{\uparrow}$ packet is generated from observed module-state rows, exception rows, two latent-prefix rows, directed rewrite-edge rows, monotone-path coverage, exception-kill accounting, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3521.1, Definition 3521.2, and Definition 3521.3 seed the finite monotone-path completion witness over observed module states, latent prefixes, rewrite edges, coverage rows, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LatentCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close empirical biological transition mechanisms, complete state-space sampling, host graph-completion oracles, unbounded latent search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exception splitting, latent-prefix coverage, directed rewrite reachability, exception-kill accounting, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.LatentCompletionUp.

Chapter 3522

A Concrete Naming Certificate for $\mathsf{BoundedFunctionFamily}^{\uparrow}$

Following the bounded-family usage standard in Kelley, *General Topology*, and Bourbaki-style analysis, the $\mathsf{BoundedFunctionFamily}^{\uparrow}$ packet records the finite BEDC surface for a family of continuous maps whose displayed values are controlled by pointwise real bounds and a sup-norm ledger. It sits between the compact-continuity route of chapter 129, the equicontinuity family surface of chapter 413, the Arzela–Ascoli packet of chapter 412, and the real-name boundary of chapter 38. The carrier is a finite BHist packet over $\mathsf{ContinuousMap}^{\uparrow}$ rows, a $\mathsf{CompactMetric}^{\uparrow}$ source row, a $\mathsf{CompleteMetric}^{\uparrow}$ target row, pointwise $\mathsf{Real}^{\uparrow}$ bound rows, a displayed sup-norm ledger, hsame, Cont, Pkg, and a local $\mathsf{NameCert}$ row.

Definition 3522.1 (Bounded function-family carrier)**.** *A* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, I, M, B, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *surface of chapter 129;*
2. *Y is the complete metric target row, supplying only the displayed target metric and completion-facing name data;*
3. *I is the finite family-index row naming the inspected family members;*
4. *M is the finite list of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, one for each displayed family member from X into Y;*
5. *B is the pointwise* $\mathsf{Real}^{\uparrow}$ *bound ledger, recording for each displayed index and inspected source window the real-name bound read from the value row;*
6. *U is the displayed sup-norm ledger, obtained by comparing the value-distance rows against the finite pointwise bound surface and not by selecting an ambient supremum;*
7. *H stores componentwise* hsame *transport for the source, target, index, map, pointwise-bound, sup-norm, provenance, and name rows;*
8. *C stores the displayed* Cont *replay from compact source inspection through continuous-map values, pointwise real bounds, and the sup-norm ledger;*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$ *row.*
The carrier has no row for an open-cover compactness principle, an ambient function-space topology, a selected supremum, quotient real equality, or a choice-selected subsequence.

Lean4 (definition) `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier`
Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyUpTasteGate single carrier alignment`

Theorem 3522.2 (Bounded function-family NameCert obligations)**.** *For the carrier of Definition 3522.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$ *surface consists exactly of carrier admission for* $F = (X, Y, I, M, B, U, H, C, P, N)$*, compact-source readback through* X*, complete-target readback through* Y*, finite index admission through* I*, continuous-map readback through* M*, pointwise* $\mathsf{Real}^{\uparrow}$ *bound readback through* B*, displayed sup-norm ledger readback through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and terminal naming through* N*.*

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier namecert obligations`

Proof. Project the carrier row by row. The source row X is read through the compact metric certificate, while Y supplies only the complete metric target surface needed by the displayed value distances.

The finite index row I then bounds the family members that can be inspected. Each inspected member is read through a corresponding $\mathsf{ContinuousMap}^{\uparrow}$ row in M, so no ambient function-space topology is introduced.

The pointwise bound row B and the sup-norm ledger U are downstream of the map rows. They compare displayed real-name value distances and record the bound ledger as finite data; they do not choose a supremum in the host real line and do not identify quotient real representatives.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no other coordinate occurs in F, these rows exhaust the local NameCert obligations. □

Theorem 3522.3 (Bounded function-family sup-norm route)**.** *Every accepted* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *packet exports its sup-norm ledger only through*

$$X, Y, I, M, B, U, H, C, P, N.$$

A consumer first reads the compact source row, complete target row, finite family-index row, and displayed continuous-map rows; only then may it read the pointwise $\mathsf{Real}^{\uparrow}$ *bound rows and the finite sup-norm ledger. The route exports no open-cover compactness theorem, ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence.*

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilyCarrier sup norm route`

Proof. Start with the source and target rows. The $\mathsf{CompactMetric}^{\uparrow}$ row X supplies the inspected source windows, and the $\mathsf{CompleteMetric}^{\uparrow}$ row Y supplies the target metric surface without importing a function-space object.

Next read the finite family data. The index row I lists the family members available to the packet, and M attaches each listed member to a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row.

The pointwise real bounds in B are then read against those map values. The sup-norm ledger U is a finite comparison ledger over the displayed values and bounds, so it can be replayed by C and transported by H without selecting an ambient supremum.

The provenance and local name rows P, N close the packet. Because the carrier contains no open-cover, ambient topology, quotient equality, or subsequence-selector coordinate, the listed rows exhaust every sup-norm read available to downstream Arzela–Ascoli and equicontinuity consumers. □

Theorem 3522.4 (Bounded function-family sup-norm non-escape)**.** *Any Arzela–Ascoli, equicontinuity, or compact-continuity consumer using the* $\mathsf{BoundedFunctionFamily}^{\uparrow}$ *sup-norm ledger must read exactly*

$$X, Y, I, M, B, U, H, C, P, N.$$

Consequently it cannot replace U by an ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence. The only available sup-norm data are the displayed pointwise $\mathsf{Real}^{\uparrow}$ *bound comparisons, the finite family-index and continuous-map rows, structural transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3522.1. The rows available to a sup-norm consumer are precisely $X, Y, I, M, B, U, H, C, P, N$, with U positioned after the compact source, complete target, finite index, continuous-map, and pointwise-bound rows.

Apply Definition 3522.2 to identify the local certificate obligations and Definition 3522.3 to force every sup-norm read through the displayed route.

No additional carrier coordinate remains after those rows are exhausted. Thus an Arzela–Ascoli, equicontinuity, or compact-continuity consumer cannot import an ambient function-space topology, selected supremum, quotient real equality, or choice-selected subsequence in place of U. □

Lean4 ✓ `BEDC.Derived.BoundedFunctionFamilyUp.BoundedFunctionFamilySupNormNonescape`

Falsifiable prediction: An accepted $\mathsf{BoundedFunctionFamily}^{\uparrow}$ consumer that reads a sup-norm ledger without the displayed finite family-index row, continuous-map rows, pointwise $\mathsf{Real}^{\uparrow}$ bound rows, and structural hsame, Cont, Pkg, and NameCert rows refutes the carrier surface.

Independence witness: compactmetric, equicontinuityfamily, arzelaascoli, real: no listed sibling jointly carries the compact source, complete target, finite continuous-map family, pointwise real-bound ledger, displayed sup-norm route, transport, replay, provenance, and local name rows of $F = (X, Y, I, M, B, U, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedFunctionFamily}^{\uparrow}$ packet is generated from a compact metric source row, complete metric target row, finite family-index row, continuous-map rows, pointwise RealUp bound rows, displayed sup-norm ledger, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$, seed)

Scope closed: Definition 3522.1, Definition 3522.2, Definition 3522.3, and Definition 3522.4 bind the finite bounded-family carrier and its displayed sup-norm ledger route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedFunctionFamily}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient function-space topology, selected suprema, quotient real equality, choice-selected subsequences, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-map readback, pointwise bound exactness, displayed sup-norm replay, structural transport, provenance, and local naming for BEDC.Derived.BoundedFunctionFamilyUp.

Chapter 3523

A Concrete Naming Certificate for CauchyCompletionApartness$^{\uparrow}$

Following Bishop and Bridges on constructive metric completion and apartness, the CauchyCompletionApartness$^{\uparrow}$ packet records the positive separation boundary for completed Cauchy names without passing through quotient representatives. It is a Real-and-Completion neighbour of the universal completion route in chapter 2319, the apartness completion surface of chapter 1567, the separated metric classifier in chapter 1411, the regular rational readback of chapter 301, and the finite stream window interface of chapter 126.

Definition 3523.1 (Cauchy completion apartness carrier)**.** *A* CauchyCompletionApartness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (U, R, S, W, E, G, H, C, P, N)$$

with the following displayed rows:

1. *U is the* CauchyCompletionUniversalProperty$^{\uparrow}$ *completion source row, exposing the completed Cauchy-name request before any apartness read is attempted.*
2. *R is the* RegSeqRat$^{\uparrow}$ *regular rational readback row attached to the Cauchy name and to the requested tolerance budget.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row through which the compared Cauchy names are inspected.*
4. *W is the* Real$^{\uparrow}$ *witness-seal row reached only after the regular readback and stream-window rows have been displayed.*
5. *E is the* SeparatedMetric$^{\uparrow}$ *zero-distance classifier handoff that identifies when a completed comparison must collapse to the separated metric boundary.*
6. *G is the* RealApartnessCompletion$^{\uparrow}$ *positive-gap row, recording apartness by a displayed completed-real gap rather than by an external representative choice.*
7. *H records componentwise* hsame *transport for the completion source, readback, windows, real seal, separatedness handoff, apartness gap, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an apartness request through U, R, S, W, E, G.*
9. *P is the* Pkg *provenance over* CauchyCompletionApartness$^{\uparrow}$, CauchyCompletionUniversalProperty$^{\uparrow}$, RealApartnessCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, R, S, W, E, G, H, C, P, N$. It has no coordinate for quotient-stream equality, selected representatives, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Theorem 3523.2 (Cauchy completion apartness NameCert obligations)**.** *For every accepted* CauchyCompletionApa *carrier*

$$A = (U, R, S, W, E, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the completion source, regular readback, stream-window, real-seal, separatedness, apartness-gap, transport, replay, provenance, and local naming rows;*
2. *completion-source exposure, requiring an apartness request to pass through the displayed* CauchyCompletionUniver *row U before any completed comparison is read;*
3. *regular-window readback, requiring R and S to be the only numerical and stream observation route before the* $\mathsf{Real}^{\uparrow}$ *seal row W is reached;*
4. *separatedness handoff, requiring zero-distance collapse to be read only through the displayed* $\mathsf{SeparatedMetric}^{\uparrow}$ *row E;*
5. *apartness-gap exposure, requiring positive separation to be read only through the* $\mathsf{RealApartnessCompletion}^{\uparrow}$ *row G after U, R, S, W, E have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport through H and displayed replay through C to preserve the same route;*
7. *ledger non-escape, excluding quotient representatives, selected limits, host equality, trichotomy, countable choice, ambient completeness, Lean verification, and bridge checking from the packet.*

No NameCert obligation is stored outside these rows.

```
Lean4✓  BEDC.Derived.CauchyCompletionApartnessUp.TasteGate.CauchyCompletionApartnessCarrier
namecert obligations
```

Proof. Carrier admission is projection from Definition 3523.1. The accepted packet exposes exactly the completion source, regular readback, finite windows, real seal, separatedness row, apartness-gap row, and structural rows.

The completion-facing route begins at U. The regular rational row R and stream-window row S then supply the only finite observation data available before the real seal W is read.

Separatedness and apartness are two downstream readings of the same displayed route. The row E records the separated metric handoff for zero-distance collapse, while G records the positive-gap apartness completion row.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since the carrier has no coordinate for quotient representatives, host equality, trichotomy, choice, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3523.3 (Cauchy completion apartness separatedness handoff)**.** *Every separatedness or apartness read of an accepted* $\mathsf{CauchyCompletionApartness}^{\uparrow}$ *carrier*

$$A = (U, R, S, W, E, G, H, C, P, N)$$

must factor through the finite route

$$U \longmapsto R, S, W \longmapsto E, G \longmapsto H, C, P, N.$$

Consequently a consumer receives only the displayed completion source, regular rational readback, finite stream windows, real seal, separated-metric handoff, apartness-completion gap, componentwise transport, continuation replay, provenance, and local name. It receives no quotient representative, selected limit, host real equality, trichotomy split, countable-choice schedule, or ambient completeness theorem.

Proof. Start with a separatedness or apartness read of the carrier. Projection exposes U first, so the read begins from the displayed universal completion row rather than from a quotient representative of the completed Cauchy name.

The numerical read then passes through R, S, W. The regular rational row and finite stream-window row fix the inspected data before the real seal is available.

The comparison boundary is exactly E, G. Zero-distance collapse is handled by the separated metric row E, and positive separation is handled by the apartness-completion gap row G.

The structural rows H, C, P, N preserve and name only that finite route. Because no excluded coordinate appears in the carrier, the handoff cannot export quotient representatives, selected limits, host real equality, trichotomy, choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionApartness}^{\uparrow}$ packet is generated from a CauchyCompletionUniversalProperty source row, RegSeqRat readback, StreamName windows, a RealUp seal row, a SeparatedMetric handoff, a RealApartnessCompletion positive-gap row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3523.1, Definition 3523.2, and Definition 3523.3 seed the finite apartness route for completed Cauchy names over displayed completion, readback, window, separatedness, gap, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host real equality, real trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-source exposure, regular-window readback, separatedness handoff, apartness-gap exposure, classifier stability, consumer nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionApartnessUp.

Chapter 3524

A Concrete Naming Certificate for $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ packet records the finite comparison certificate by which located Dedekind cut language and regular Cauchy real names meet inside the existing $\mathsf{LocatedReal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ routes. It is a NameCert reconstruction over the concrete rows of chapter 1933, chapter 436, chapter 301, chapter 291, and chapter 38.

The packet is deliberately finite. It records a displayed cut row, a located-real enclosure row, a regular rational readback, a dyadic bracket ledger, and a terminal real seal, but it does not add quotient reals, trichotomy, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 3524.1 (Bishop regular-cut equivalence carrier)**.** *A* $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, L, R, D, Q, H, T, P, N)$$

with the following visible rows:

1. *C is a displayed Dedekind or located cut row whose lower and upper sides are read as positive and negative rational evidence rather than as quotient-real data;*
2. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row that records the finite located interval information associated to the cut row C;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the located enclosures, giving the regular Cauchy name visible to completion consumers;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger that records the dyadic lower and upper rational brackets used to compare the cut and regular-name presentations;*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from R through the existing real-name route;*
6. *H stores the displayed* hsame *transports for C, L, R, D, Q;*
7. *T stores the displayed* Cont *routes that replay the cut-to-enclosure, enclosure-to-readback, bracket-compatibility, and real-seal handoffs;*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$ *row for the finite comparison packet.*

The carrier predicate is $\mathsf{BishopRegularCutEquivalencePacket}(E)$*. Its public rows are exactly the displayed cut, located enclosure, regular-rational readback, dyadic bracket ledger, real seal, transports, provenance, and local naming row.*

Theorem 3524.2 (Bishop regular-cut equivalence NameCert obligations)**.** *For the carrier*

$$E = (C, L, R, D, Q, H, T, P, N)$$

of Definition 3524.1, the local $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$ *surface has five obligations:*

1. *carrier admission for the displayed finite packet;*
2. *bracket and readback compatibility, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *brackets in* D *to support both the located cut row* C *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*;*
3. *located-real handoff, requiring the enclosure row* L *to be the only route from* C *to* R*;*
4. *classifier transport, requiring accepted carriers to compare only through the displayed* hsame *and* Cont *routes in* H, T *together with the package row* P*;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring every real-facing read through* Q *to factor through* C, L, R, D, H, T, P, N *and no outside quotient, trichotomy, choice, or completeness datum.*

Lean4 ✓ `BEDC.Derived.BishopRegularCutEquivalenceUp.TasteGate.BishopRegularCutEquivalenceCarrier namecert obligations`

Proof. Carrier admission is projection from the finite packet E. The rows list the cut display, located enclosure, regular readback, dyadic bracket ledger, real seal, transports, provenance, and local naming certificate as one visible history.

Bracket and readback compatibility is the joint routing condition on D and R. The same dyadic bracket ledger that locates the cut side also supplies the rational bounds used by the regular Cauchy readback, so the comparison has no hidden quotient-real coordinate.

The located-real handoff is the row L. Any passage from the cut row C to the regular name R must pass through the displayed located enclosure, and any transported comparison must replay through H, T with provenance P.

Finally the real-facing row Q is a terminal seal, not a new completeness principle. Since Q is reached only after the displayed cut, enclosure, readback, bracket, transport, package, and local certificate rows have been consumed, the packet exports the finite comparison surface and no stronger real-analysis principle. □

Theorem 3524.3 (Bishop regular-cut equivalence located-real handoff)**.** *For every accepted* $\mathsf{BishopRegularCutEquiv}$ *carrier*

$$E = (C, L, R, D, Q, H, T, P, N),$$

the public comparison between the located cut row C *and the regular Cauchy row* R *factors through the* $\mathsf{LocatedReal}^{\uparrow}$ *enclosure row* L *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger* D*. The* $\mathsf{Real}^{\uparrow}$ *seal row* Q *may be consumed only after that finite route has been replayed through* H, T, P, N*.*

Consequently the packet gives a finite L10 comparison certificate between Bishop located cuts and regular Cauchy real names without asserting Dedekind completeness, quotient-real extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3524.1. The row C is the displayed cut surface, L is the located enclosure row attached to it, and R is the regular rational readback.

Apply Definition 3524.2. The bracket and readback obligation forces D to be the shared finite dyadic ledger for both sides of the comparison, while the located-real handoff obligation prevents a direct jump from C to R.

The classifier transport obligation then carries the same route through H, T, P, N. The real-seal nonescape obligation leaves Q as the terminal consumer of that route, so no unlisted quotient, trichotomy, choice, or completeness datum can be read from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularCutEquivalence}^{\uparrow}$ packet is generated from a displayed located cut row, a $\mathsf{LocatedReal}^{\uparrow}$ enclosure row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{DyadicRatCore}^{\uparrow}$ bracket ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, displayed hsame and Cont transports, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3524.1, Definition 3524.2, and Definition 3524.3 bind the finite comparison surface between Bishop located cuts and regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularCutEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient reals, trichotomy, choice principles, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: The next paper grade is a checked finite carrier theorem for the cut, located-enclosure, regular-readback, dyadic-bracket, and real-seal rows.

Chapter 3525

A Concrete Naming Certificate for $\mathsf{DyadicToleranceLadder}^{\uparrow}$

Following Bishop and Bridges' constructive use of rational tolerances, the $\mathsf{DyadicToleranceLadder}^{\uparrow}$ packet records the finite ladder of dyadic requests used by Real and Completion consumers before any completed real object is exposed. It sits on the dyadic arithmetic core of chapter 291, the finite request-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the tail-budget packet of chapter 379. The packet gives downstream Real and Completion chapters one reusable source object for the tolerances 2^{-n}, their halving route, and their finite handoff to regular-rational windows.

Definition 3525.1 (DyadicToleranceLadder carrier)**.** *A* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, Q, E, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording dyadic numerators, denominators, comparison rows, and the visible tolerance term* 2^{-n}*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *request-window row, recording the finite list of precision indices at which a consumer asks for tolerances.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational window row, recording the rational readback surface that consumes the requested dyadic tolerances.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary row, recording only the terminal handoff at which the tolerance ladder can be read by a real-facing consumer.*
5. *T is the* $\mathsf{DyadicTailBound}^{\uparrow}$ *row, recording the finite tail-budget comparison that treats successive ladder entries as shrinking tolerance requests.*
6. *H records componentwise* hsame *transport for the dyadic endpoint, window, regular-rational, real-seal, tail-bound, continuation, provenance, and naming rows.*
7. *C records displayed* Cont *routes from dyadic endpoints through finite windows, regular-rational readback, tail-bound comparisons, and the real-seal boundary.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicToleranceLadder}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicTailBound}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, Q, E, T, H, C, P, N$. It has no coordinate for quotient stream equality, trichotomy on real names, selected limits, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DyadicToleranceLadderUp.DyadicToleranceLadderTasteGate single carrier alignment`

Theorem 3525.2 (DyadicToleranceLadder NameCert obligations)**.** *For every accepted* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *carrier*

$$L = (D, S, Q, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ *surface consists exactly of carrier exposure for the dyadic endpoint row D, finite request-window row S, regular-rational readback row Q, real-seal boundary E, tail-bound comparison row T, componentwise transport row H, continuation row C, package provenance row P, and local naming row N.*

Proof. Carrier exposure is projection from Definition 3525.1. Each row named in the statement is one coordinate of the finite packet, and the definition lists no hidden coordinate for a completed real object.

The dyadic endpoint and request-window obligations are D and S. The regular-rational readback and real-facing boundary obligations are Q and E, while the shrinking tolerance budget is the row T.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these rows gives the displayed $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$ surface, with quotient stream equality, trichotomy, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the packet. □

Theorem 3525.3 (DyadicToleranceLadder monotone halving)**.** *Let $L = (D, S, Q, E, T, H, C, P, N)$ be an accepted* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *carrier. If two requested indices m, n are both displayed in the finite* $\mathsf{StreamName}^{\uparrow}$ *window S and m is the successor request after n, then the tolerance read from D at m is routed through the same dyadic endpoint row as one halving of the tolerance read at n, and the comparison is consumed by T before Q or E can export the request.*

Proof. The only dyadic endpoint row in the carrier is D, so both displayed requests are read from the same dyadic arithmetic surface. The successor relation is visible only as part of the finite window S, not as an external sequence choice.

The halving comparison is the role of the tail-bound row T. It consumes the two displayed endpoint reads before the regular-rational window Q is opened, so the comparison cannot be replaced by an unrecorded real-order argument.

After T records the comparison, the readback row Q and the seal boundary E may consume the tolerance request. The route therefore remains a finite ladder through $D, S, T, Q, E, H, C, P, N$, and it exports no quotient stream equality, trichotomy, selected limit, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicToleranceLadder}^{\uparrow}$ packet is generated from dyadic endpoint rows, finite StreamName request windows, RegSeqRat readback windows, Real seal boundaries, DyadicTailBound comparison rows, componentwise transports, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$, seed)

Scope closed: Definition 3525.1, Definition 3525.2, and Definition 3525.3 bind the finite dyadic tolerance ladder, its local NameCert surface, and its monotone halving route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DyadicToleranceLadder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove Real$^{\uparrow}$ completeness, quotient stream equality, trichotomy, selected limits, countable choice, or a public standard bridge.

Upgrade path: Scoped closure requires Lean encoding of the carrier, the local NameCert obligation surface, and the monotone halving route over the displayed dyadic and regular-rational rows.

Chapter 3526

A Concrete Naming Certificate for $\mathsf{NestedClosedBall}^{\uparrow}$

Following Bishop and Bridges and the Cantor-style nested closed ball criterion, the $\mathsf{NestedClosedBall}^{\uparrow}$ packet records explicit nested closed ball completion data as finite NameCert rows. It sits beside the metric-completion carrier of chapter 1506, the complete-metric service of chapter 128, the Cauchy-filter evidence of chapter 333, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a displayed descending ball route rather than a host completeness theorem, a quotient completion, or a choice-selected center.

Definition 3526.1 (Nested closed ball carrier)**.** *A* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, K, F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row naming the ambient completion request before any ball center is read.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row available only as a displayed limit-service dependency, not as an independent selector.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *filter row generated by the descending family of closed balls.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspecting the indexed balls and their center approximations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the inspected center and radius data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger, recording the dyadic shrink schedule, containment witnesses, and late-radius comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the completion source, complete-metric service, filter, windows, readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a nested-ball request through M, K, F, S, R, D, E.*
10. *P is the* Pkg *provenance over* $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected limits, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.TasteGate.NestedClosedBallTasteGate single carrier alignment`

Theorem 3526.2 (Nested closed ball NameCert obligations)**.** *For every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric-completion source, displayed complete-metric service, Cauchy-filter row, finite stream windows, regular rational readback, dyadic diameter ledger, terminal real seal, componentwise transport, displayed continuation replay, package provenance, and local naming. Every nested-ball read is therefore routed through $M, K, F, S, R, D, E, H, C, P, N$, and no obligation is stored outside those rows.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallNameCertObligations`

Proof. Carrier admission is projection from Definition 3526.1. The packet displays the completion source, the complete-metric service, the Cauchy filter generated by the nested balls, the finite windows, the rational readback, and the dyadic diameter ledger before the real seal can be consumed.

The constructive nested-ball content is the route through F, S, R, D. The filter row records the descending family, the stream row exposes only finite indexed windows, the regular rational row reads center and radius data, and the dyadic row records the shrink and containment comparisons.

The structural rows H, C, P, N transport, replay, source, and name exactly that route. Since the carrier contains no open-cover compactness row, selected limit, quotient completion, countable-choice schedule, ambient real-completeness row, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3526.3 (Nested closed ball Cauchy-center route)**.** *Every center read exported by an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

Equivalently, a consumer may inspect the descending closed-ball family only through the displayed Cauchy-filter row, finite stream windows, regular rational readback, dyadic shrink ledger, and terminal real seal; it receives no open-cover compactness theorem, selected limit, quotient completion, countable-choice schedule, independent real-completeness witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier and project the rows M, K, F, S, R, D, E. The metric-completion and complete-metric rows provide the surrounding completion service, while the Cauchy-filter row is the only entry point for the descending closed-ball family.

The center data are then inspected through finite windows. The stream row S exposes finite ball indices, the regular rational row R reads the center and radius approximants, and the dyadic ledger D records the shrink and containment comparisons needed for a Cauchy-center route.

Finally the real seal E and the structural rows H, C, P, N preserve the same route for public consumers. Because no excluded coordinate occurs in the carrier, the handoff cannot export open-cover compactness, selected limits, quotient completions, countable choice, or ambient real completeness. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallCauchyCenterRoute`

Theorem 3526.4 (Nested closed ball RegSeqRat center extraction)**.** *Every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *packet whose displayed diameters refine through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows yields a* $\mathsf{RegSeqRat}^{\uparrow}$*-facing center readback through the* $\mathsf{StreamName}^{\uparrow}$ *finite windows before the* $\mathsf{Real}^{\uparrow}$ *seal is available. The extracted readback uses the nested closed-ball carrier row, the local NameCert obligations, and the Cauchy-center route to expose only the finite center schedule, dyadic shrink comparisons, and regular-rational handoff.*

Proof. Begin with the carrier of Definition 3526.1. Its rows S, R, D are the only finite-window, regular-rational, and dyadic tolerance coordinates available before the terminal real seal.

Apply the local obligations of Definition 3526.2. The NameCert surface requires the Cauchy-filter row, finite windows, regular rational readback, and dyadic diameter ledger to be admitted together, so a displayed diameter refinement through D cannot bypass S or R.

Use the Cauchy-center route of Definition 3526.3. The nested finite-ball chain is inspected through S, its center and radius approximants are read through R, and the dyadic shrink comparisons in D provide the tolerance control needed for a center readback.

The structural rows H, C, P, N preserve this route under transport and replay. Since E is downstream of S, R, D, the packet yields the $\mathsf{RegSeqRat}^{\uparrow}$-facing center readback before any $\mathsf{Real}^{\uparrow}$ seal is consumed. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallRegSeqRatCenterExtraction`

Theorem 3526.5 (Nested closed ball diameter-tail stability)**.** *For every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N),$$

the dyadic diameter ledger D is stable under later $\mathsf{StreamName}^{\uparrow}$ *windows S and under the componentwise* hsame *and* Cont *transport rows H, C. Every later ball read therefore keeps the displayed containment witness and shrink comparison before the* $\mathsf{Real}^{\uparrow}$ *seal E is consumed.*

The packet exports no selected center, quotient completion, choice witness, or ambient completeness object through this stability route.

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallCarrier diameter tail stability`

Proof. Project the diameter, stream, transport, and replay rows D, S, H, C from the accepted carrier. The dyadic ledger records the shrink comparisons and containment witnesses before the terminal seal is available.

Use the Cauchy-filter row F to keep tail membership in the descending closed-ball family. Later stream windows in S inspect only finite indexed balls already covered by this filter row, so no later read can replace the displayed containment ledger.

The regular rational row R supplies the scheduled center and radius reads attached to those windows. Its center readback is downstream of the dyadic ledger, so the same shrink comparison remains attached to the later window.

Finally, H transports the completion source, filter, stream windows, readback, dyadic ledger, and seal componentwise, while C replays the same nested-ball route. Thus every later ball read preserves containment and diameter control before E is consumed. Since the carrier has no selected-center, quotient-completion, choice, or ambient-completeness coordinate, the stability theorem exports none. □

Theorem 3526.6 (Nested closed ball obligation package)**.** *The carrier of Definition 3526.1, the local obligations of Definition 3526.2, and the Cauchy-center route of Definition 3526.3 jointly form the exact obligation package consumed by later metric-completion and complete-metric chapters. The package consists of the metric-completion source row, complete-metric service row, Cauchy-filter row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row.*

It supplies no open-cover compactness, selected limit, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallObligationPackage`

Proof. Start with the carrier of Definition 3526.1. Its displayed tuple is precisely the list of metric-completion, complete-metric, Cauchy-filter, stream-window, regular-rational, dyadic, real-seal, transport, replay, provenance, and naming rows.

Apply Definition 3526.2. This identifies the local NameCert surface with admission and replay through exactly those rows, with no obligation stored outside the packet.

Finally apply Definition 3526.3. Every center read factors through the same finite route, so later metric-completion and complete-metric consumers receive one obligation package and no open-cover compactness, selected limit, quotient completion, countable-choice schedule, or ambient real-completeness witness. □

Theorem 3526.7 (Nested closed ball consumer nonescape)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$*-,* $\mathsf{CompleteMetric}^{\uparrow}$*-, or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *packet must read exactly the displayed route*

$$M, K, F, S, R, D, E, H, C, P, N$$

before using any nested-ball endpoint. The consumer may use the selected center only through the finite windows, regular-rational readback, dyadic diameter ledger, Cauchy-filter route, terminal real seal, componentwise transport, displayed replay, provenance, and local NameCert rows of that packet.

Consequently selected centers, quotient completions, countable choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ *completeness cannot escape the packet boundary.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallConsumerNonescape`

Proof. Start with the carrier of Definition 3526.1. Its displayed tuple is the only accepted source for a nested-ball endpoint, and the local obligations of Definition 3526.2 identify the NameCert surface with the rows $M, K, F, S, R, D, E, H, C, P, N$.

Apply the Cauchy-center route of Definition 3526.3. Any center read used by a metric-completion, complete-metric, or real-facing consumer must pass through the Cauchy-filter row F, the stream windows S, the regular-rational readback row R, and the dyadic ledger D before the terminal seal E is available.

Use the RegSeqRat extraction of Definition 3526.4 and the diameter-tail stability of Definition 3526.5. They keep later endpoint reads attached to the same finite windows, regular readback, dyadic shrink comparisons, and transport and replay rows.

Finally apply the obligation package of Definition 3526.6. The package is exactly the displayed route consumed by downstream completion chapters, so no consumer obtains a separate selected center, quotient completion, countable-choice schedule, open-cover compactness theorem, or ambient real-completeness witness outside the packet. □

Theorem 3526.8 (Nested closed ball kernel scope binding)**.** *Every scoped read of an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier*

$$B = (M, K, F, S, R, D, E, H, C, P, N)$$

binds M *to* $\mathsf{MetricCompletion}^{\uparrow}$*,* K *to* $\mathsf{CompleteMetric}^{\uparrow}$*,* F *to* $\mathsf{CauchyFilter}^{\uparrow}$*,* S *to* $\mathsf{StreamName}^{\uparrow}$*,* R *to* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *to* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *to* $\mathsf{Real}^{\uparrow}$*, and* H, C, P, N *respectively to* BHist *transport by* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *row. No scoped consumer can reinterpret one of these rows as an open-cover compactness object, a selected endpoint, a quotient completion, a countable-choice schedule, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallKernelScopeBinding`

Proof. Project the carrier of Definition 3526.1. Its displayed tuple already assigns the completion, complete-metric, filter, stream, regular-rational, dyadic, real-seal, transport, replay, provenance, and naming roles to the rows $M, K, F, S, R, D, E, H, C, P, N$.

Apply the obligation package of Definition 3526.6 and the consumer nonescape theorem Definition 3526.7. These results force every downstream completion, complete-metric, or real-facing read to consume the same displayed route before any endpoint is used.

Use the RegSeqRat extraction of Definition 3526.4 together with the diameter-tail stability of Definition 3526.5. They keep finite windows, center readback, dyadic shrink comparisons, and later tail reads attached to their original rows under hsame transport and Cont replay. Thus the scoped read binds the rows to the named kernel objects and has no coordinate for the excluded host-level objects. □

Theorem 3526.9 (Nested closed ball public NameCert export)**.** *The carrier of Definition 3526.1, the obligation package of Definition 3526.6, the consumer nonescape theorem Definition 3526.7, the RegSeqRat extraction of Definition 3526.4, the diameter-tail stability theorem Definition 3526.5, and the kernel scope binding of Definition 3526.8 assemble the public* $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$ *export for the nested-ball completion route.*

The exported surface is exactly the finite route through $M, K, F, S, R, D, E, H, C, P, N$*. It refuses open-cover compactness, selected limits, quotient completions, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallPublicNameCertExport`

Proof. Start with the carrier and the obligation package. They identify the displayed rows $M, K, F, S, R, D, E, H, C$ as the metric-completion source, complete-metric service, Cauchy-filter row, stream windows, regular rational readback, dyadic diameter ledger, real seal, transport, replay, provenance, and local naming route.

Use consumer nonescape together with RegSeqRat extraction. Any completion-facing or real-facing read must pass through the finite windows, regular readback, dyadic shrink ledger, Cauchy-filter route, terminal seal, transport, replay, provenance, and local NameCert rows before it can use an endpoint.

Apply diameter-tail stability and kernel scope binding. Later ball reads keep the same containment and shrink comparisons under hsame transport and Cont replay, and each row remains bound to its displayed kernel role.

Thus the public export is the finite nested-ball completion route named by the local certificate. Since the carrier has no coordinate for open-cover compactness, selected limits, quotient completions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those objects is exported. □

Theorem 3526.10 (Nested closed ball public completion surface)**.** *Every accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier exposes the finite nested-ball completion route as a public consumer surface through the rows*

$$M, K, F, S, R, D, E, H, C, P, N.$$

The surface is public exactly as a displayed completion interface: M names the metric-completion source, K the complete-metric service, F the nested Cauchy-filter row, S finite ball windows, R regular rational center readback, D dyadic diameter control, E the terminal real seal, and H, C, P, N the transport, replay, provenance, and local naming rows. It exposes no selected limit, no ambient completeness theorem, no quotient completion, no countable-choice schedule, and no open-cover compactness object.

Proof. Start with the public NameCert export of Definition 3526.9. It already packages the carrier, obligation package, nonescape theorem, center extraction, diameter-tail stability, and kernel scope binding as one public route.

Read the exported rows objectwise. The completion source, complete-metric service, nested filter, finite windows, regular readback, dyadic ledger, real seal, transport, replay, provenance, and local naming entries are exactly $M, K, F, S, R, D, E, H, C, P, N$.

Finally apply consumer nonescape and kernel scope binding. They keep every downstream completion or real-facing read on the displayed route, so the public surface exports only the finite nested-ball completion interface and none of the excluded host-level objects. □

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallPublicCompletionSurface`

Theorem 3526.11 (Nested closed ball separated uniqueness handoff)**.** *Let $B = (M, K, F, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{NestedClosedBall}^{\uparrow}$ *carrier whose public completion surface is read through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *service row K. Any two endpoint reads obtained from the same accepted packet can be compared only through a* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *separated comparison route as in chapter 5983. The handoff uses the same nested-filter row F, stream windows S, regular readback R, dyadic diameter ledger D, terminal seal E, transport H, replay C, provenance P, and local name N; it exports no selected center, quotient completion point, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.NestedClosedBallUp.NestedClosedBallSeparatedUniquenessHandoff`

Proof. Start with the carrier and obligation package of Definition 3526.1 and Definition 3526.6. They identify $M, K, F, S, R, D, E, H, C, P, N$ as the only rows available before a nested-ball endpoint can be read.

Use consumer nonescape and the public NameCert export, Definition 3526.7 and Definition 3526.9. These keep both endpoint reads inside the same displayed completion route and prevent either read from becoming a selected host center or quotient representative.

Now apply the public completion surface of Definition 3526.10. Since the surface exposes K only as the displayed $\mathsf{CompleteMetric}^{\uparrow}$ service row, any comparison between two endpoint reads must be delegated to the separated uniqueness route of chapter 5983. The structural rows H, C, P, N preserve the same packet identity, so the comparison cannot import ambient $\mathsf{Real}^{\uparrow}$ completeness or host equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedClosedBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NestedClosedBall}^{\uparrow}$ packet is generated from a MetricCompletion source row, a CompleteMetric service row, a CauchyFilter nested-ball row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore diameter

ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{NestedClosedBall}^{\uparrow}}$, publicClosed)

Scope closed: Definition 3526.9 and Definition 3526.10 bind the public nested-ball completion route over the displayed rows $M, K, F, S, R, D, E, H, C, P, N$.

Formal verification: FormalStatus(NameCert$_{\mathsf{NestedClosedBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient completions, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification and bridge checking require the public completion surface to be mirrored by a checked target and a standard bridge.

Chapter 3527

A Concrete Naming Certificate for $\mathsf{RealBoundedInterval}^{\uparrow}$

Bishop–Bridges constructive real analysis treats a bounded interval by endpoint evidence and finite approximation data rather than by an ambient compact set. The $\mathsf{RealBoundedInterval}^{\uparrow}$ packet records that endpoint interval as finite BEDC NameCert data over the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic endpoint-order ledger of chapter 291, the located endpoint route of chapter 1709, and the bracketing surface of chapter 1807. It names a closed finite endpoint packet for downstream Real and Completion consumers; it is not open-cover compactness, a Heine–Borel theorem, a selected point of the interval, a quotient interval, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3527.1 (Real bounded interval carrier)**.** *A* $\mathsf{RealBoundedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, S, T, R, Q, B, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint* $\mathsf{Real}^{\uparrow}$ *seal rows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order ledger recording the displayed finite evidence that the lower endpoint precedes the upper endpoint at the requested tolerance.*
3. *S and T are finite* $\mathsf{StreamName}^{\uparrow}$ *windows through which the endpoint names are observed before any interval read is exported.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the paired endpoint windows.*
5. *Q is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row tying the endpoint seals, dyadic order ledger, and readback route to a located interval surface.*
6. *B is the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *row giving the finite bracket consumed by Real and Completion-facing readers.*
7. *H records componentwise* hsame *transport for endpoint seals, endpoint-order evidence, windows, readback, located interval, bracket, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded-interval request through $L, U, D, S, T, R, Q, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{FiniteBracketingInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, Heine–Borel compactness, selected points, quotient intervals, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealBoundedIntervalUp.RealBoundedIntervalTasteGate single carrier alignment`

Theorem 3527.2 (Real bounded interval NameCert obligations)**.** *For every accepted* $\mathsf{RealBoundedInterval}^{\uparrow}$ *carrier*

$$I = (L, U, D, S, T, R, Q, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the two endpoint* $\mathsf{Real}^{\uparrow}$ *seals,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order evidence, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{LocatedRealInterval}^{\uparrow}$ *located endpoint row,* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Every bounded-interval consumer must therefore read $L, U, D, S, T, R, Q, B, H, C, P, N$ in the displayed route, and no obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3527.1. The projection exposes exactly the two endpoint seals, the endpoint-order ledger, the two finite windows, readback, located interval, bracket, and structural rows.

The endpoint-order obligation is read from D. The finite-window obligation is read from S, T, and the regular rational readback obligation is read from R before either the located interval row Q or the bracket row B is available.

Componentwise transport is supplied by H, and continuation replay is supplied by C. The provenance and local naming rows P, N attach the same finite route to the packet, so there is no separate host interval, quotient interval, selected point, compactness, or completeness coordinate from which a consumer could read additional data. □

Theorem 3527.3 (Real bounded interval endpoint seal route)**.** *Every endpoint read of an accepted* $\mathsf{RealBoundedInterval}^{\uparrow}$ *packet factors through the paired* $\mathsf{Real}^{\uparrow}$ *seals L, U, the endpoint-order ledger D, the finite windows S, T, and the regular rational readback R before the located interval row Q and bracket row B are consumed. The route exports no host real equality, quotient interval equality, selected endpoint representative, trichotomy row, open-cover compactness proof, Heine–Borel theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an endpoint-facing consumer and project the route C. The only endpoint rows are L, U, and the only order witness attached to them is the dyadic ledger D.

The consumer then reads the finite windows S, T and the regular rational readback R. Those rows provide the displayed approximation surface needed by the located interval and bracketing rows.

Finally Q and B package the endpoint surface for interval readers. Since the carrier contains no host equality, quotient interval, selected representative, compactness theorem, or completeness coordinate, the endpoint seal route is exhaustive. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealBoundedInterval}^{\uparrow}$ packet is generated from two RealUp endpoint seals, a DyadicRatCore endpoint-order ledger, paired StreamName finite windows, RegSeqRat readback, LocatedRealInterval and FiniteBracketingInterval rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBoundedInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3527.1, Definition 3527.2, and Definition 3527.3 seed the finite endpoint interval route over real seals, dyadic order evidence, finite windows, regular readback, located interval rows, and bracketing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBoundedInterval}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Heine–Borel compactness, selected interval points, quotient intervals, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order evidence, finite-window readback, located interval handoff, bracketing handoff, endpoint seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealBoundedIntervalUp.

Chapter 3528

A Concrete Naming Certificate for $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$

Following Stieltjes, Jordan, and the Bishop–Bridges constructive reading of bounded variation, this packet records only the finite boundary by which a $\mathsf{BoundedVariation}^{\uparrow}$ integrator supplies interval-jump rows to $\mathsf{RiemannStieltjes}^{\uparrow}$ and $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ consumers. It sits after the partition-variation ledger of chapter 398, the tagged-sum handoff of chapter 1637, the regulated Cauchy readback of chapter 2351, the regular rational surface of chapter 301, the dyadic endpoint core of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite interval-jump boundary; it is not a variation measure, a Lebesgue integration object, a quotient of host functions, a selected limit, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3528.1 (Stieltjes measure boundary carrier)**.** *A* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, \Pi, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row naming the integrator only through displayed finite partition-variation data.*
2. *Π is the displayed interval partition, including ordered endpoints, interval cells, and the refinement witnesses on which boundary reads are requested.*
3. *J is the endpoint-jump* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, recording the finite left and right endpoint differences attached to the cells of Π.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes those jump rows at the requested rational windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, Π, J, R have been displayed.*
6. *H records componentwise* hsame *transport for the bounded-variation source, partition, jump ledger, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from bounded-variation partition reads to interval jumps, from jumps to regular rational readback, and from readback to the* $\mathsf{Real}^{\uparrow}$ *seal.*
8. *P is the* Pkg *provenance over* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, \Pi, J, R, E, H, C, P, N$. It has no coordinate for measure completion, Lebesgue integration, a host function quotient, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.StieltjesMeasureBoundaryUp.StieltjesMeasureBoundaryTasteGate single carrier alignment`

Theorem 3528.2 (Stieltjes measure boundary NameCert obligations)**.** *For every accepted* StieltjesMeasureBoundar *carrier*

$$S = (B, \Pi, J, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{BoundedVariation}^{\uparrow}$ *source, displayed interval-partition exposure through Π, endpoint-jump exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger J,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3528.1. The source row B is already the finite partition-variation object of chapter 398, so the boundary packet begins from displayed partition data rather than from a completed variation measure.

The interval and jump obligations are the rows Π and J. The partition cells provide the only endpoint pairs that may be inspected, and the dyadic ledger records the corresponding finite endpoint differences through chapter 291.

The readback and seal obligations are downstream. The regular rational route of chapter 301 consumes J through R, and the real-name route of chapter 38 consumes R through E. Transport, replay, provenance, and local naming are exactly H, C, P, N, leaving no hidden measure, quotient, choice, selected-limit, or completeness row. □

Theorem 3528.3 (Stieltjes boundary bounded-variation handoff)**.** *Every* $\mathsf{RiemannStieltjes}^{\uparrow}$ *or* $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *packet receives the integrator boundary only through the displayed route*

$$\mathsf{BoundedVariation}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}.$$

Concretely, the consumer must read the bounded-variation source B, the interval partition Π, the endpoint-jump ledger J, and the regular rational readback R before any $\mathsf{Real}^{\uparrow}$ *seal E is visible.*

Proof. Start with the bounded-variation source. By chapter 398, that source exposes partition data and dyadic variation ledgers, not a variation measure or a host function quotient.

The boundary carrier refines the source to endpoint jumps on the displayed partition. The cells of Π determine the finite list of endpoint pairs, and the ledger J records their dyadic jump reads.

Now pass to consumers. The $\mathsf{RiemannStieltjes}^{\uparrow}$ route of chapter 1637 and the $\mathsf{RegulatedCauchyIntegral}^{\uparrow}$ route of chapter 2351 may consume only the displayed regular readback and seal rows R, E together with H, C, P, N. Thus the handoff is the listed finite route, with no channel for measure completion, Lebesgue integration, selected limits, countable choice, or ambient real completeness. □

Theorem 3528.4 (Stieltjes boundary Real seal non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* StieltjesMeasureBoundar *carrier is available only after the bounded-variation source row, displayed interval partition, endpoint-jump* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been read. The seal exposes the same*

finite interval-jump boundary and exports no measure completion, Lebesgue integration object, host function quotient, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the seal coordinate of Definition 3528.1. The row E is downstream from B, Π, J, R and is transported or replayed only by the structural rows H and C.

Apply Definition 3528.2. It identifies the bounded-variation source, partition exposure, dyadic jump ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows as the complete local certificate surface.

Apply Definition 3528.3. Every integration-facing read must pass through the same finite jump route before the seal is visible. Since none of the excluded objects is a carrier coordinate, the $\mathsf{Real}^{\uparrow}$ seal cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ packet is generated from a BoundedVariation source row, a displayed interval partition, endpoint-jump DyadicRatCore ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3528.1, Definition 3528.2, Definition 3528.3, and Definition 3528.4 seed the finite Stieltjes boundary by which bounded-variation interval jumps are read by Riemann–Stieltjes and regulated Cauchy integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StieltjesMeasureBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, Lebesgue integration, host function quotients, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-jump exactness, regular readback, bounded-variation handoff, Real-seal non-escape, transport and replay stability, provenance non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.StieltjesMeasureBoundaryUp.

Chapter 3529

A Concrete Naming Certificate for $\mathsf{BanachContraction}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{BanachContraction}^{\uparrow}$ packet records the fixed-point route as finite displayed data from the contraction map and Picard orbit surfaces to the complete-metric endpoint boundary. It consumes $\mathsf{ContractionMapping}^{\uparrow}$ from chapter 1418, $\mathsf{PicardIteration}^{\uparrow}$ from chapter 2359, $\mathsf{MetricCompletion}^{\uparrow}$ from chapter 1506, and $\mathsf{SeparatedMetric}^{\uparrow}$ from chapter 1411. The packet also reads the complete-metric row of chapter 128, regular rational readback through chapter 301, and the terminal real-name boundary of chapter 38; it does not claim quotient completions, selected representatives, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3529.1 (BanachContraction carrier)**.** *A* $\mathsf{BanachContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, I, M, Q, S, E, H, C, P, N)$$

whose coordinates are exactly:

1. K*, the accepted* $\mathsf{ContractionMapping}^{\uparrow}$ *row displaying the self-map graph, contraction bound, one-step Picard comparison, and fixed-point readiness surface;*
2. I*, the* $\mathsf{PicardIteration}^{\uparrow}$ *row displaying the finite orbit, residual-distance ledger, Cauchy-modulus extraction, and regular-sequence handoff;*
3. M*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, read only as the endpoint-admission surface for the Picard Cauchy data;*
4. Q*, the regular-Cauchy and readback route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal used by endpoint-facing consumers;*
5. S*, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness row, consumed only after the two endpoint reads expose zero distance through the displayed metric row;*
6. E*, the endpoint seal row naming the fixed-point candidate as the result of the completion route rather than as a selected host point;*
7. H*, componentwise* hsame *transport for contraction data, Picard iterates, complete-metric endpoint rows, readback rows, separatedness rows, endpoint seals, replay rows, provenance, and local naming;*

8. *C, displayed* Cont *replay from contraction admission through Picard iteration, Cauchy readback, complete-metric endpoint admission, separated uniqueness, and endpoint sealing;*
9. *P, the* Pkg *provenance row over* $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N, the local* $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$ *row.*

The carrier names the constructive Banach contraction route. It has no coordinate for quotient completion equality, selected representatives, host fixed-point equality, host function extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BanachContractionUp.BanachContractionTasteGate single carrier alignment`

Theorem 3529.2 (BanachContraction NameCert obligations)**.** *For an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier $B = (K, I, M, Q, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$ *surface consists exactly of contraction-map admission, Picard-iteration admission, complete-metric endpoint visibility, regular-Cauchy and Real readback visibility, separated uniqueness visibility, endpoint-seal stability, componentwise transport, displayed replay, package provenance, and local naming for the finite packet of Definition 3529.1.*

Proof. Project the carrier. The rows K, I, M, Q, S, E are the only mathematical coordinates, and the rows H, C, P, N are the only structural coordinates available to the local certificate.

The contraction and Picard obligations come from chapter 1418 and chapter 2359. They expose the self-map graph, contraction bound, finite orbit ledger, residual comparisons, and Cauchy-modulus handoff before any endpoint row is inspected.

The completion and readback obligations come next. The complete-metric row M admits the Cauchy data to an endpoint-facing surface, while Q records the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback path used by consumers of that endpoint.

Finally S, E, H, C, P, N bind uniqueness, endpoint sealing, transport, replay, provenance, and naming to the same displayed rows. Since no excluded object is a coordinate of B, the NameCert surface exports only the constructive Banach contraction packet. □

Theorem 3529.3 (BanachContraction Cauchy orbit route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier first consumes the contraction row K and the Picard row I, then the complete-metric row M, then the readback route Q through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and only afterward the endpoint seal E. Thus the Cauchy iteration route is the displayed finite chain*

$$\mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto (\mathsf{RegSeqRat}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow}) \longmapsto E.$$

No read may replace this chain by quotient completion equality, a selected limit representative, countable choice, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with K. The contraction row supplies only the finite self-map graph, rational contraction bound, one-step Picard comparison, and fixed-point readiness boundary.

Next consume I. The Picard row generates the finite orbit ledger, residual-distance comparisons, Cauchy-modulus extraction, and regular handoff, so the read has Cauchy data before it asks for an endpoint.

The complete-metric row M is then the only endpoint-admission coordinate. The readback row Q records how the admitted endpoint is inspected through regular rational windows, the metric-completion surface, and the terminal real seal.

The endpoint seal E is downstream of those rows. Because the carrier includes no quotient completion, representative selector, choice schedule, host function equality, or ambient real-completeness coordinate, the displayed route is the whole Cauchy iteration route. □

Theorem 3529.4 (BanachContraction Picard tail modulus)**.** *Every accepted* $\mathsf{BanachContraction}^{\uparrow}$ *orbit read must pass through the* $\mathsf{PicardIteration}^{\uparrow}$ *finite orbit row, residual-distance row,* $\mathsf{CauchyModulus}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{CompleteMetric}^{\uparrow}$ *row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the unique fixed-point seal can be read. Equivalently, the fixed-point consumer may inspect the separated uniqueness row only after the Picard tail modulus has been routed through the displayed regular rational completion boundary.*

Proof. Apply Definition 3529.3. It forces every endpoint-facing read through the contraction row, Picard row, complete-metric row, readback route, and endpoint seal before any fixed-point endpoint is consumed.

Now read the Picard row through Definition 2359.5. The finite orbit ledger and residual-distance row determine the $\mathsf{CauchyModulus}^{\uparrow}$ row, and that row is the source of the $\mathsf{RegSeqRat}^{\uparrow}$ handoff exposed to completion-facing consumers.

Finally apply Definition 3529.5. The separated uniqueness seal is downstream of the endpoint route, so it cannot be read before the Picard finite orbit, residual-distance ledger, modulus handoff, regular rational boundary, complete-metric row, and terminal real seal have all been displayed. □

Theorem 3529.5 (BanachContraction unique fixed-point seal)**.** *For an accepted* $\mathsf{BanachContraction}^{\uparrow}$ *carrier* $B = (K, I, M, Q, S, E, H, C, P, N)$*, uniqueness of the fixed-point endpoint is exported only through the separated zero-distance route* S *after both compared endpoint reads have passed through* K, I, M, Q, E *and the structural rows* H, C, P, N*. The exported uniqueness is classifier equality inside the displayed* $\mathsf{SeparatedMetric}^{\uparrow}$ *surface, not host equality of points, quotient equality of completion names, a selected representative, trichotomy, or ambient completeness.*

Proof. Take two endpoint-facing reads of the same carrier. By Definition 3529.3, each read must expose the contraction row, Picard row, complete-metric row, readback route, and endpoint seal before any uniqueness comparison is available.

Apply the separatedness row S. It compares the two endpoint reads only after the displayed zero-distance witness is routed through the $\mathsf{SeparatedMetric}^{\uparrow}$ surface; this is the same kind of classifier route isolated in chapter 1411.

Transport and replay are componentwise through H and C, while P and N record provenance and local naming for the same finite packet. These rows do not add a host equality, quotient equality, representative selector, trichotomy principle, or ambient completeness theorem.

Therefore the fixed-point seal is unique only in the displayed local classifier sense. The carrier records the constructive endpoint route and its separated uniqueness boundary, not a host-level fixed-point object. □

Falsifiable prediction: Every accepted $\mathsf{BanachContraction}^{\uparrow}$ read factors through one ContractionMapping row, one PicardIteration row, one CompleteMetric row, one RegSeqRat/MetricCompletion/Real readback route, one SeparatedMetric uniqueness row, one endpoint seal, hsame transport, Cont replay, Pkg provenance, and NameCert row before any fixed-point endpoint is consumed.

Independence witness: contractionmapping, picard-iteration, completemetric, metriccompletion, separatedmetric, regseqrat, real: none is bijective with $\mathsf{BanachContraction}^{\uparrow}$ because the packet jointly stores the contraction route, Picard Cauchy iteration, completion endpoint admission, readback route, separated uniqueness seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachContraction}^{\uparrow}$ packet is generated from a ContractionMapping row, a PicardIteration row, a CompleteMetric endpoint row, a RegSeqRat/MetricCompletion/Real readback route, a SeparatedMetric uniqueness row, an endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3529.1, Definition 3529.2, Definition 3529.3, and Definition 3529.5 seed the finite constructive Banach contraction fixed-point route over the existing contraction, Picard, completion, readback, and separatedness packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host fixed-point equality, host function extensionality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, contraction-map visibility, Picard-iteration visibility, complete-metric endpoint visibility, readback exactness, separated uniqueness routing, endpoint-seal stability, transport, replay, provenance, and local naming.

Chapter 3530

A Concrete Naming Certificate for $\mathsf{PeanoExistence}^{\uparrow}$

Following Peano and Bishop and Bridges, the $\mathsf{PeanoExistence}^{\uparrow}$ packet records constructive local ODE existence data without uniqueness as finite NameCert rows. It consumes the real endpoint discipline of chapter 38, the continuous vector-field surface of chapter 130, and the displayed ODE source of chapter 193. The packet names an interval-window, vector-field, Euler-polygon, and compact-containment route; it does not name a Lipschitz modulus, a unique flow, a maximal solution, host compactness, or a choice-selected path.

Definition 3530.1 (Peano existence carrier)**.** *A* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (I, V, W, E, Q, M, H, C, G, N)$$

with the following displayed rows:

1. *I is the interval source row, binding the initial time, initial state, local radius, and finite rational time window through the* $\mathsf{Ode}^{\uparrow}$ *source surface.*
2. *V is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row, read on the displayed time-state window and classified only through its continuous-map graph and modulus ledgers.*
3. *W is the closed* $\mathsf{Real}^{\uparrow}$ *interval and state-window row that bounds the Euler-polygon construction before any solution read is exported.*
4. *E is the finite Euler polygon approximation ledger, including step windows, endpoint approximants, and local slope reads from V.*
5. *Q is the compact-containment row, recording that every displayed polygon segment remains in the certified window W.*
6. *M is the modulus row used to compare nearby polygon windows and to pass from finite polygon data to a Cauchy-style local solution row.*
7. *H records componentwise* hsame *transport for interval sources, vector-field rows, real windows, polygon ledgers, containment rows, modulus rows, replay rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from the ODE request through I, V, W, E, Q, M.*
9. *G is the* Pkg *provenance over* $\mathsf{PeanoExistence}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for uniqueness, Lipschitz dependence, a selected maximal solution, ambient host compactness, or a choice principle.

Lean4 ✓ `BEDC.Derived.PeanoExistenceUp.TasteGate.PeanoExistenceTasteGate single carrier alignment`

Theorem 3530.2 (Peano existence NameCert obligations)**.** *For every accepted* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier*

$$P = (I, V, W, E, Q, M, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval source row, continuous vector-field row, real closed window row, Euler polygon approximation ledger, compact-containment row, modulus row, componentwise transport, displayed continuation replay, package provenance, and local naming. Every local existence read is therefore routed through* $I, V, W, E, Q, M, H, C, G, N$.

Proof. Project the accepted carrier of Definition 3530.1. The interval source I fixes the ODE request, V supplies the displayed continuous vector-field read, and W supplies the real closed window in which the construction is inspected.

The approximation content is carried by E, Q, M. Euler polygons in E provide finite stepwise approximants, Q keeps every inspected polygon segment inside the recorded window, and M gives the finite comparison ledger needed for a Cauchy-style local solution read.

The structural rows H, C, G, N transport, replay, package, and name exactly that finite route. Since the carrier has no coordinate for uniqueness, Lipschitz dependence, maximal-extension selection, host compactness, or choice, the NameCert surface cannot export any of those objects. □

Definition 3530.3 (Peano interval trap carrier)**.** *The Peano interval trap inside a* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier is the displayed subpacket*

$$(I, V, W, E, Q, M)$$

consisting of the closed $\mathsf{Real}^{\uparrow}$ *time-state window named by I and W, the continuous vector-field row V, the Euler polygon ledger E, the compact-containment row Q, and the modulus comparison row M. It records only the finite data used to keep polygon approximants inside the source packet before a local solution row is exported.*

Theorem 3530.4 (Peano interval trap obligations)**.** *Every accepted local-existence read of a* $\mathsf{PeanoExistence}^{\uparrow}$ *packet must pass through the interval source I, the continuous vector-field row V, the closed* $\mathsf{Real}^{\uparrow}$ *time-state window W, the Euler polygon ledger E, the compact-containment row Q, and the modulus handoff M before it can export a solution row. Thus the interval trap obligations are exactly the finite rows displayed in Definition 3530.3, together with the transport, replay, provenance, and local naming rows of Definition 3530.1.*

Proof. Begin with the carrier of Definition 3530.1. The interval source I and vector-field row V are the only rows that provide the ODE request and continuous slope reads, while W is the only closed Real window in which the finite construction is inspected.

The Euler polygon ledger E gives the finite approximants. The compact-containment row Q checks each displayed segment against the same source window, and the modulus row M supplies the comparison handoff needed before a local solution read can be exported.

Rows H, C, G, N do not add another analytic source. They transport, replay, record provenance, and name the same interval trap. Therefore every accepted local-existence read must expose I, V, W, E, Q, M before the solution row is public. □

Theorem 3530.5 (Peano existence scoped primitive route)**.** *Every accepted local-existence read of a* $\mathsf{PeanoExistence}^{\uparrow}$ *carrier factors through the exact primitive route*

$\mathsf{Ode}^{\uparrow}$ *interval source* $\to$ $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row* $\to$ $\mathsf{Real}^{\uparrow}$ *closed window* $\to$ *Euler polygon ledger* $\to$ *compac*

Thus the scoped read exposes precisely the interval source, vector-field, closed-window, polygon, containment, modulus, transport, replay, provenance, and local naming rows of Definition 3530.1. It exports no Lipschitz uniqueness, selected maximal solution, host compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the accepted carrier $P = (I, V, W, E, Q, M, H, C, G, N)$ of Definition 3530.1. The only analytic entrance rows are the $\mathsf{Ode}^{\uparrow}$ interval source I, the $\mathsf{ContinuousMap}^{\uparrow}$ vector-field row V, and the closed $\mathsf{Real}^{\uparrow}$ window W.

Apply Definition 3530.2. It identifies the local $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$ surface with the displayed carrier rows and records that every local-existence read is routed through $I, V, W, E, Q, M, H, C, G, N$.

Apply Definition 3530.4. The finite solution-producing part of the route is exactly the Euler polygon ledger E, compact-containment row Q, and modulus handoff M before transport, replay, provenance, and local naming are consumed.

Since the carrier definition has no coordinate for Lipschitz uniqueness, a selected maximal solution, host compactness, countable choice, or ambient Real completeness, none of those objects can be exported by the scoped primitive route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PeanoExistence}^{\uparrow}$ packet is generated from an $\mathsf{Ode}^{\uparrow}$ interval source row, a $\mathsf{ContinuousMap}^{\uparrow}$ vector-field row, a closed $\mathsf{Real}^{\uparrow}$ time-state window, Euler polygon approximation ledgers, compact-containment and modulus rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$, obligation)

Scope closed: Definition 3530.1, Definition 3530.2, Definition 3530.3, and Definition 3530.4 expose the constructive local existence obligations over displayed ODE, continuous-map, real-window, Euler-polygon, containment, modulus, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PeanoExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim uniqueness, Lipschitz dependence, selected maximal solutions, host compactness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding these obligations to their exact primitive routes for interval sources, continuous vector fields, closed Real windows, Euler polygon ledgers, compact-containment rows, modulus handoffs, transport, replay, provenance, and local naming.

Chapter 3531

A Concrete Naming Certificate for $\mathsf{LocatedCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges on located constructive metric completions and Kelley on functorial completion discipline, the $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet records the finite route by which a located metric-space presentation is sent to its completion-facing regular-name surface. It sits after the located real boundary of chapter 1933, the metric-completion carrier of chapter 1506, the universal-property witness of chapter 2319, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the displayed functorial handoff used by L10 completion consumers; it is not a quotient completion theorem, not a selected host limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3531.1 (Located completion functor carrier)**.** *A* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{LocatedCompletionFunctor}(L, M, U, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *or located metric-source row exposing the positive-apartness and approximation data that make Cauchy requests decidable as finite located reads.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row naming the completion-facing object surface, without quotienting Cauchy names.*
3. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row recording the displayed universal-property handoff for uniformly continuous maps already routed through finite data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when the completion object is presented through regular rational names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located, completion, universal-property, and regular rational rows have been displayed.*
6. *F is the functorial action row, naming how source maps are transported to completion-facing maps through L, M, U, R, E.*
7. *H records componentwise* hsame *transport for the located source, completion carrier, universal-property handoff, regular rational readback, Real seal, functorial action, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a located source request through the completion, universal-property, regular rational, Real-seal, and functorial-action rows.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, M, U, R, E, F, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, countable-choice schedules, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge data.

Lean4 ✓ `BEDC.Derived.LocatedCompletionFunctorUp.LocatedCompletionFunctorTasteGate single carrier alignment`

Theorem 3531.2 (Located completion functor NameCert obligations)**.** *For every accepted* LocatedCompletionFunc *carrier* $\mathsf{LocatedCompletionFunctor}(L, M, U, R, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the located source, metric-completion carrier, Cauchy-completion universal-property handoff, regular rational readback, Real seal, functorial action, componentwise transport, displayed replay, provenance, and local naming rows.*

Every completion-facing consumer must read the route

$$L, M, U, R, E, F, H, C, P, N$$

before treating a located map as a completion map. The route exports no quotient completion theorem, selected host limit, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge datum.*

Proof. Carrier admission is projection from Definition 3531.1. The accepted tuple displays the located source row, completion carrier, universal-property row, regular rational readback, Real seal, functorial action, and structural support.

The completion-facing read begins with L and M. These rows expose located input data and the completion object surface before any map action is available.

Next read U, R, E, F. The universal-property row supplies the finite handoff discipline, the regular rational and Real rows name the target readback boundary, and the functorial action row records the transported map only after those rows are present.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since no quotient equality, selected limit, choice schedule, trichotomy row, host real equality, Real completeness theorem, Lean verification row, or bridge datum is present in the carrier, none can become a local NameCert obligation. □

Theorem 3531.3 (Located completion functor Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ *packet reaches the Real-facing handoff only through the displayed located source row, metric-completion row, Cauchy-completion universal-property row, regular rational readback row, Real seal row, and functorial-action row of Definition 3531.1. In particular, the handoff is a finite NameCert route for located completion data, not a quotient completion, selected host limit, countable-choice construction, trichotomy principle, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. The consumer-facing evidence is the displayed carrier, so the located source, completion object, universal-property handoff, readback, seal, and functorial action are the only rows it may inspect.

The regular rational and Real rows are downstream. The consumer reaches R and E only after L, M, U have fixed the located completion route and before F packages the transported map.

The support rows H, C, P, N carry the same route through transport, replay, provenance, and local naming. No coordinate supplies quotient completion equality, a selected limit, countable choice, trichotomy, host equality, ambient Real completeness, Lean verification, or bridge checking. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet factors a Real-facing completion read through one located source row, one MetricCompletion row, one CauchyCompletionUniversalProperty handoff, one RegSeqRat readback, one RealUp seal, one functorial-action row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real, metric-completion, cauchy-completion-universal-property, regseqrat, real: $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located source, completion carrier, universal-property handoff, regular rational readback, Real seal, and functorial action as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3531.1, Definition 3531.2, and Definition 3531.3 seed the finite located-completion functor route over located source rows, metric-completion carriers, universal-property handoffs, regular rational readback, Real seals, functorial action, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-source exactness, completion-carrier handoff, universal-property consumption, RegSeqRat and Real handoff, functorial-action exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionFunctorUp.

Constructive story: A $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ packet is generated from a located source row, a MetricCompletion row, a CauchyCompletionUniversalProperty handoff, a RegSeqRat readback row, a RealUp seal row, a functorial-action row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3532

A Concrete Naming Certificate for $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ packet records the finite surface at which a located regular rational sequence exposes the modulus windows consumed by Phase Real and Completion classifiers. It sits after the dyadic tolerance core of chapter 291, the finite observation surface of chapter 126, the regular rational readback of chapter 301, the located Cauchy representation route of chapter 4070, and the terminal real-name seal of chapter 38. The packet is a carrier for displayed located-modulus data, not a quotient completion, a selected limit, a countable-choice schedule, a trichotomy oracle, or an ambient completeness principle.

Definition 3532.1 (RegSeqRat located-modulus carrier)**.** *A* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, L, \mu, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance query row.*
2. *L is the* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *located regular-sequence row.*
3. *μ is the displayed modulus row assigning finite stream windows to the tolerance queries admitted by Q.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed by μ.*
5. *D is the dyadic decision ledger binding the tolerance query to the displayed window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row at the window named by W.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached after the located regular-sequence readback.*
8. *H stores componentwise* hsame *transport for $Q, L, \mu, W, D, R, E, C, P, N$.*
9. *C stores displayed* Cont *replay from tolerance request to modulus window, regular readback, locatedness row, and real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $Q, L, \mu, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit representative, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegSeqRatLocatedModulusUp.TasteGate.RegSeqRatLocatedModulusTasteGate single carrier alignment`

Theorem 3532.2 (RegSeqRat located-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegSeqRatLocatedMod}$ *carrier $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, located-modulus exposure, finite-window stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise transport, displayed continuation replay, package provenance, local naming, and ledger refusal for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3532.1. The accepted packet first exposes the tolerance query Q, the located regular-sequence row L, and the displayed modulus row μ.

The modulus row opens only the finite $\mathsf{StreamName}^{\uparrow}$ window W and the dyadic decision ledger D attached to that request. The rational readback row R is then read at the displayed window, and the terminal real-name seal E is reached only after this finite route has been replayed.

Transport and continuation are exhausted by H and C, while P and N record provenance and local naming. Since the carrier has no further coordinate, no accepted packet can import quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3532.3 (RegSeqRat located-modulus finite-window stability)**.** *Every* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *packet factors through $Q, L, \mu, W, D, R, E, H, C, P, N$ before any consumer-visible located read is exported. The handoff therefore exposes the located modulus as a finite window discipline over regular rational readback and terminal real-name sealing, not as quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier as $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$. A $\mathsf{LocatedRegSeqRat}^{\uparrow}$ consumer must read the located row L together with the displayed modulus row μ before it can export a locatedness response.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches rational data only through the finite window W named by μ and the readback row R bound to that window. The dyadic ledger D records which tolerance request was served, so the read cannot be replaced by an unbounded stream equality or by an arbitrary limit choice.

A $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal E only after the same window and readback route has been replayed by C and transported by H. The remaining rows P and N certify provenance and local naming, so the handoff contains no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness. □

Theorem 3532.4 (RegSeqRat located-modulus located row exhaustion)**.** *For an accepted* $\mathsf{RegSeqRatLocatedModul}$ *carrier $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$, every located read exported to* $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *is exhausted by exactly the located row L, the displayed modulus row μ, the finite windows W, the dyadic decision ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{Real}^{\uparrow}$ *seal E, the componentwise transport row H, the displayed replay row C, the provenance row P, and the local naming row N. No exported located read factors through quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 3532.1. The located export has no coordinates other than $Q, L, \mu, W, D, R, E, H, C, P, N$, and its consumer-visible located row is L together with the displayed modulus row μ.

Use the finite-window stability theorem Definition 3532.3. A $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ consumer reads L and μ before exporting locatedness; a $\mathsf{RegSeqRat}^{\uparrow}$ consumer reaches data through W, D, R; and a $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal E only after transport and replay through H and C.

The remaining rows P and N source and name this same route. Since the carrier definition contains no quotient stream equality, selected-limit representative, countable-choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness witness, the listed rows exhaust every exported located read. □

Theorem 3532.5 (RegSeqRat located-modulus Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *carrier* $M = (Q, L, \mu, W, D, R, E, H, C, P, N)$ *passes in order through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance query* Q*, the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *row* L*, the displayed modulus row* μ*, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*, the dyadic decision ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}$ *row* N*. No* $\mathsf{Real}^{\uparrow}$*-facing read bypasses this route by quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness.*

Proof. Project the carrier from Definition 3532.1. The only Real-facing rows in the packet are the displayed tolerance, locatedness, modulus, window, ledger, readback, seal, transport, replay, provenance, and naming rows.

Apply the finite-window stability theorem Definition 3532.3 to force every $\mathsf{Real}^{\uparrow}$ consumer through $Q, L, \mu, W, D, R, E, H, C, P, N$ before a consumer-visible seal is exported.

Apply the located-row exhaustion theorem Definition 3532.4. It rules out additional coordinates for quotient stream equality, selected limits, countable choice, trichotomy, host equality, or ambient completeness, so the displayed order is the full Real route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance query, a LocatedRegSeqRat row, a displayed modulus row, StreamName finite windows, a dyadic decision ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3532.1, Definition 3532.2, Definition 3532.3, and Definition 3532.5 seed the finite located-modulus route from regular rational sequence data to Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegSeqRatLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-modulus exposure, finite-window stability, RegSeqRat readback, RealUp seal routing, and ledger refusal for BEDC.Derived.RegSeqRatLocatedModulusUp.

Chapter 3533

A Concrete Naming Certificate for $\mathsf{MorseTheory}^{\uparrow}$

Following Milnor on Morse theory, the $\mathsf{MorseTheory}^{\uparrow}$ packet records the finite certificate surface by which critical-point data, Hessian signatures, local chart rows, gradient-flow incidence, and filtration boundary rows are read as one displayed Morse-theory name. It sits after the charted carrier of chapter 96, the derivative observation rows of chapter 122, the scalar comparison seal of chapter 38, the tangent-fibre bookkeeping of chapter 210, and the fibrewise metric rows of chapter 133. The packet names a local finite Morse certificate; it is not a host smooth-function space, not a Sard or transversality theorem, not a handle-decomposition completeness theorem, and not a cobordism classification principle.

Definition 3533.1 (Morse theory carrier)**.** *A* $\mathsf{MorseTheory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, D, R, T, G, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Manifold}^{\uparrow}$ *charted source row naming the local carrier on which the Morse data are read.*
2. *F is the displayed function row, exposing only the finite function-name observations used by the certificate.*
3. *D is the* $\mathsf{Derivative}^{\uparrow}$ *first- and second-derivative observation row attached to F over the charted source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar comparison row used for critical values, local height comparisons, and finite ordering tests.*
5. *T is the* $\mathsf{VectorBundle}^{\uparrow}$ *tangent-fibre row over X, carrying the finite source for Hessian and flow-incidence reads.*
6. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *metric row used to read gradient-flow incidence only after X, T, D have been displayed.*
7. *L is the local Morse ledger row: critical-point admission, nondegenerate Hessian signature, index tag, finite flow-incidence boundary, and filtration boundary are recorded here as finite observations.*

8. *H records componentwise* hsame *transport for the chart, function, derivative, scalar, tangent-fibre, metric, ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through chart admission, derivative readback, real comparison, tangent-fibre transport, metric readback, and Morse ledger inspection.*
10. *P is the* Pkg *provenance over* $\mathsf{MorseTheory}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$ *row.*

The carrier exports Morse-theory reads only through $X, F, D, R, T, G, L, H, C, P, N$. It has no coordinate for a host function space, global transversality, Sard regular-value selection, complete handle decomposition, cobordism classification, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.MorseTheoryUp.MorseTheoryTasteGate single carrier alignment`

Theorem 3533.2 (Morse theory NameCert obligations)**.** *For every accepted* $\mathsf{MorseTheory}^{\uparrow}$ *carrier*

$$M = (X, F, D, R, T, G, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$ *surface consists exactly of charted manifold admission, displayed function rows, derivative readback, real scalar comparison, tangent-fibre bookkeeping, Riemannian metric readback, local Morse ledger inspection, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every critical-point, index, or flow-incidence read exported by the packet must factor through

$\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Proof. Project the accepted carrier. The charted source is row X, the displayed function observations are row F, derivative readback is row D, real comparison is row R, tangent-fibre bookkeeping is row T, and metric readback is row G.

The Morse-specific content is confined to row L. It records critical-point admission, nondegenerate Hessian signature, index tag, finite flow-incidence boundary, and filtration boundary only after the chart, derivative, scalar, tangent, and metric rows are available.

The structural rows H, C, P, N transport, replay, source, and name exactly that displayed route. Since no row stores Sard regular values, transversality, complete handle decompositions, cobordism classification, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{MorseTheory}^{\uparrow}$ critical-point, index, or flow-incidence read factors through displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VectorBundle}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, provenance, transport, replay, and local NameCert rows.

Independence witness: manifold, derivative, real, vectorbundle, riemannianmetric: the Manifold packet records charted carriers without Morse functions, the Derivative packet records derivative observations without critical-index ledgers, the Real packet records scalar comparison without tangent flow incidence, the VectorBundle packet records fibres without a fibrewise metric read, and the RiemannianMetric packet records metric rows without Morse ledger data. None is bijective to $\mathsf{MorseTheory}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{MorseTheory}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MorseTheory}^{\uparrow}$ packet is generated from Manifold chart rows, displayed function observations, Derivative first- and second-derivative reads, Real scalar comparisons, VectorBundle tangent-fibre rows, RiemannianMetric

metric reads, a local Morse ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MorseTheory}\uparrow}$, seed)

Scope closed: Definition 3533.1 and Definition 3533.2 seed the finite Morse-theory certificate over chart, function, derivative, scalar, tangent, metric, ledger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MorseTheory}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Sard theory, transversality, handle decomposition completeness, cobordism classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, critical-point ledger exposure, Hessian signature exactness, index tagging, finite flow-incidence routing, filtration boundary routing, and a Lean carrier/obligation formalization for BEDC.Derived.MorseTheoryUp.

Chapter 3534

A Concrete Naming Certificate for $\mathsf{HyperbolicHorocycle}^{\uparrow}$

Following classical hyperbolic geometry in the Poincare disk, the $\mathsf{HyperbolicHorocycle}^{\uparrow}$ packet realizes the boundary-tangent horocycle rows requested by chapter 9463 as finite NameCert data. It sits after the disk-coordinate carrier of chapter 377, the metric-read surface of chapter 3474, the boundary-fixed transport route of chapter 3500, and the geodesic metric carrier of chapter 2213. The packet names a displayed horocycle certificate rather than a host circle model, a completed ideal-boundary quotient, a global geodesic foliation, or a physical observer trajectory.

Definition 3534.1 (Hyperbolic horocycle carrier)**.** *A* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, M, B, G, T, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source observations.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same disk data.*
3. *B is the boundary-anchor row, recording the displayed ideal boundary point only as a finite boundary-facing observation.*
4. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *incidence row used to compare tangent geodesic directions near the boundary anchor.*
5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *transport row preserving the boundary anchor during admissible disk transport.*
6. *L is the horocycle ledger row: boundary tangency, constant hyperbolic-distance readback relative to the anchored geodesic family, and finite incidence compatibility are recorded here.*
7. *H records componentwise* hsame *transport for disk, metric, boundary, geodesic, transport, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through disk admission, metric readback, boundary anchoring, geodesic incidence, boundary-fixed transport, and horocycle ledger inspection.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTranspo}$ $\mathsf{GeodesicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$ *row.*

The carrier exports horocycle reads only through $D, M, B, G, T, L, H, C, P, N$. It has no coordinate for a host Euclidean circle, quotient ideal boundary, global horocycle foliation, selected geodesic representative, physical observer motion, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicHorocycleUp.HyperbolicHorocycleTasteGate single carrier alignment`

Theorem 3534.2 (Hyperbolic horocycle NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *carrier*

$$Q = (D, M, B, G, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric readback, boundary-anchor admission, geodesic-incidence readback, boundary-fixed transport compatibility, horocycle ledger inspection, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every boundary-tangent horocycle read exported by the packet must factor through

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{PoincareBoundaryFixedTransport}^{\uparrow},\ \mathsf{GeodesicMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Name}$$

Proof. Project the accepted carrier. The disk surface is row D, the hyperbolic metric read is row M, the boundary anchor is row B, the geodesic incidence read is row G, and the boundary-fixed transport route is row T.

The horocycle-specific content is row L. It records boundary tangency, constant hyperbolic-distance readback relative to the anchored geodesic family, and finite incidence compatibility only after the disk, metric, boundary, geodesic, and transport rows have been displayed.

The structural rows H, C, P, N preserve and name that route. Since the carrier contains no host Euclidean circle, quotient ideal boundary, global horocycle foliation, selected geodesic representative, physical observer motion, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHorocycle}^{\uparrow}$ boundary-tangent read factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, provenance, transport, replay, and local NameCert rows.

Independence witness: unitdisk, hyperbolic-metric, poincare-boundary-fixed-transport, geodesic-metric: the UnitDisk packet records disk coordinates without hyperbolic tangency, the HyperbolicMetric packet records metric reads without a boundary anchor ledger, the PoincareBoundaryFixedTransport packet records fixed-boundary transport without horocycle incidence, and the GeodesicMetric packet records geodesic metric data without a boundary-tangent horocycle row. None is bijective to $\mathsf{HyperbolicHorocycle}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicHorocycle}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric metric reads, a boundary-anchor row, GeodesicMetric incidence, PoincareBoundaryFixedTransport compatibility, a horocycle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$, seed)

Scope closed: Definition 3534.1 and Definition 3534.2 seed the finite horocycle certificate over disk, metric, boundary, geodesic, transport, ledger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicHorocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Euclidean circle models, ideal-boundary quotient equality, global horocycle foliations, selected geodesic representatives, physical observer motion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk admission, metric readback, boundary anchoring, geodesic incidence, boundary-fixed transport compatibility, horocycle ledger exactness, and a Lean carrier/obligation formalization for BEDC.Derived.HyperbolicHorocycleUp.

Chapter 3535

A Concrete Naming Certificate for $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ packet records the textbook complete-metric Baire argument as finite BEDC NameCert rows rather than as an ambient host theorem. It sits over the dense-open schedule of chapter 1542, the Cauchy-thread and limit-service row of chapter 128, the metric carrier rows of chapter 57, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the surface on which a Baire dense-open construction can be inspected before any countable-choice selector, host intersection, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim is available.

Definition 3535.1 (CompleteMetricBaireSurface carrier)**.** *A* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (B, K, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, listing the finite dense-open requests and the predecessor-ball refinements made visible to the packet.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, exposing only the displayed center sequence, radius shrinkage, and limit-service handoff used by the dense-open route.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, naming the distance, ball classifier, and metric comparison data consumed by B and K.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule, centers, radii, and requested tolerances are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational center and radius data visible before the terminal seal.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the schedule, Cauchy thread, metric carrier, finite windows, and rational readback have been displayed.*
7. *H records componentwise* hsame *transport for the dense-open schedule, complete-metric thread, metric carrier, finite windows, rational readback, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay of the same dense-open, Cauchy-thread, metric, window, readback, and seal route.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for countable choice, host intersections, open-cover compactness, selected limit points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CompleteMetricBaireSurfaceUp.CompleteMetricBaireSurfaceTasteGate single carrier alignment`

Theorem 3535.2 (Baire-category complete-metric NameCert surface)**.** *For every accepted* CompleteMetricBaireSur *carrier*

$$U = (B, K, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$ *surface exposes exactly the Baire dense-open schedule, complete-metric Cauchy thread, metric carrier rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every complete-metric Baire read is routed through $B, K, M, S, R, E, H, C, P, N$, and no obligation is stored outside those rows.*

Proof. Project the accepted carrier of Definition 3535.1. The rows B, K, M, S, R, E are the only analytic rows available to a consumer: dense-open schedule, Cauchy-thread service, metric carrier, finite windows, rational readback, and terminal real seal.

The Baire-category content is read first through B. Each dense-open request and predecessor-ball refinement is displayed as finite schedule data, then the complete-metric row K carries the Cauchy thread built from those refinements.

The numerical and metric observations are constrained by M, S, R, E. The metric carrier supplies distance and ball comparisons, the stream row supplies finite inspection windows, the regular rational row reads center and radius data, and the real seal is reached only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for countable choice, host intersections, open-cover compactness, selected limit points, ambient real completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3535.3 (Complete metric Baire surface nested-ball route)**.** *Let $U = (B, K, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier. A finite dense-open schedule displayed by B together with the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy service displayed by K determines a nested closed ball route whose radii are readable through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and whose limit request is sealed through* $\mathsf{Real}^{\uparrow}$ *only after the corresponding* $\mathsf{StreamName}^{\uparrow}$ *windows S have been displayed. The route is transported, replayed, sourced, and named only by H, C, P, N.*

Proof. Consume the dense-open schedule in B. Each finite request exposes the predecessor-ball refinement data that the Baire surface records before any host intersection or choice selector is available.

Choose the displayed ball rows from that finite schedule and read their metric comparisons through M. The complete-metric row K then supplies the Cauchy thread and limit-service handoff for the same displayed nested ball route.

The numerical observations pass through S and R. The stream windows display the center and radius observations, and the regular rational readback makes the radii available without adding a selected limit point.

Finally use the Real seal nonescape already built into Definition 3535.2. The seal row E is downstream from the dense-open schedule, complete-metric thread, metric carrier, stream windows, and rational readback, while H, C, P, N preserve that route without adding countable choice, host intersections, open-cover compactness, selected limit points, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3535.4 (CompleteMetricBaireSurface open-density thread)**.** *Let*

$$U = (B, K, M, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *carrier. Every dense-open schedule read from B must expose the complete-metric Cauchy thread K, metric row M,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and* $\mathsf{Real}^{\uparrow}$ *seal E before a Baire consumer may inspect the terminal route. The exposed thread is transported, replayed, sourced, and named only by H, C, P, N, and it exports no selected generic point, countable-choice schedule, host intersection, open-cover compactness, selected limit point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3535.1. The dense-open schedule B and Cauchy thread K are coordinates of the same packet, so a consumer cannot read the schedule as an external countable-choice object.

Use the NameCert surface of Definition 3535.2. It makes $B, K, M, S, R, E, H, C, P, N$ the complete visible route for every complete-metric Baire read, with no obligation stored outside those rows.

Compose with the nested-ball route of Definition 3535.3. The metric row M, finite windows S, rational readback R, and seal row E must be exposed before the terminal route can be inspected.

Finally close by carrier exhaustion. The structural rows H, C, P, N preserve the same route, while the excluded generic point, choice schedule, host intersection, compactness, selected limit, and ambient completeness endpoints have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3535.1 and Definition 3535.2 seed the complete-metric Baire surface over dense-open schedules, Cauchy-thread rows, metric carriers, finite windows, rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricBaireSurface}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, host intersections, open-cover compactness, selected limit points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric Cauchy-thread exactness, finite-window readback, rational center readback, real sealing, and nonescape for BEDC.Derived.CompleteMetricBair

Constructive story: A $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteMetric Cauchy thread, MetricSpace carrier rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3536

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Oxtoby's Baire-space presentation, the $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ packet records the usual complete ultrametric structure on Baire space as finite NameCert rows. It reads finite prefix-cylinder data from $\mathsf{BaireSpace}^{\uparrow}$ in chapter 334, routes Cauchy-thread convergence through $\mathsf{CompleteMetric}^{\uparrow}$, exposes windows through $\mathsf{StreamName}^{\uparrow}$, reads rational tolerances through $\mathsf{RegSeqRat}^{\uparrow}$, and seals endpoint distances only through $\mathsf{Real}^{\uparrow}$. The downstream complete-metric Baire surface of chapter 3535 can therefore consume a direct Baire-space completeness object instead of an implicit host completeness assertion.

Definition 3536.1 (BaireSpaceCompleteness carrier)**.** *A* $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, K, S, R, E, H, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, including the displayed stream schedule, finite prefix windows, prefix classifier, and local cylinder transport.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row for the usual ultrametric in which the distance between two streams is bounded by the first displayed prefix disagreement.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite observations of the Cauchy thread and its limit stream are inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for dyadic tolerances and prefix-length bounds used by the ultrametric estimates.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint distances, reached only after B, K, S, R have been displayed.*
6. *H records componentwise* hsame *transport for prefix cylinders, Cauchy-thread rows, stream windows, rational readback, and real endpoint seals.*
7. *P is the* Pkg *provenance over* $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompleteness}^{\uparrow}}$ *row naming the displayed completeness packet.*

The classifier compares accepted packets only through these rows. It has no coordinate for a choice-selected branch, host sequence equality, quotient completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3536.2 (BaireSpaceCompleteness NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ *carrier* $C = (B, K, S, R, E, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompleteness}^{\uparrow}}$ *surface consists exactly of the prefix-cylinder admission row, the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread route, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *dyadic tolerance readback, the* $\mathsf{Real}^{\uparrow}$ *endpoint-distance seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from Definition 3536.1. The Baire-space row B supplies only finite prefix-cylinder data, while K records the Cauchy-thread service for the ultrametric route.

The observation rows are then forced by S and R. A consumer reads finite stream windows through $\mathsf{StreamName}^{\uparrow}$ and reads dyadic tolerance bounds through $\mathsf{RegSeqRat}^{\uparrow}$ before it can inspect any endpoint distance.

Finally the endpoint seal is the row E, and the structural rows H, P, N only transport, source, and name the displayed route. Since none of these coordinates contains a branch selector, host equality, quotient completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no such obligation is exported. □

Theorem 3536.3 (BaireSpaceCompleteness Cauchy-thread route)**.** *Let* $C = (B, K, S, R, E, H, P, N)$ *be an accepted* $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ *packet. Every Cauchy thread in the usual Baire-space ultrametric is read through the finite prefix-cylinder row of* B*, the* $\mathsf{CompleteMetric}^{\uparrow}$ *route* K*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* S*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback* R *before its endpoint distance is sealed through* E*. The route is transported, replayed, sourced, and named only by* H, P, N*.*

Proof. Start with the prefix-cylinder surface of chapter 334. It exposes finite stream prefixes and their cylinder classifier before any completed stream or host topology is available.

Apply the complete-metric route through K. A Cauchy-thread request is therefore routed through the displayed $\mathsf{CompleteMetric}^{\uparrow}$ service rather than through a quotient completion or selected limit operation.

The finite observations are read through S and the dyadic tolerance bounds through R. Only after those rows are displayed does the endpoint-distance row E seal the resulting real-valued estimate. The transport, replay, provenance, and local naming rows preserve this order, matching the downstream complete-metric Baire surface of chapter 3535. □

Theorem 3536.4 (Baire finite-prefix cylinder induction)**.** *Let* $C = (B, K, S, R, E, H, P, N)$ *be an accepted* $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ *packet, and let* Λ *be a finite replay ledger whose entries refine the displayed* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row by one finite prefix-cylinder observation at a time. Induction on* Λ *preserves the invariant that the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread route is read through the same* $\mathsf{BaireSpace}^{\uparrow}$ *prefix surface* B*,* $\mathsf{StreamName}^{\uparrow}$ *window row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal* E*. Thus every finite prefix-cylinder refinement ledger exports the same Baire-space completeness route and no selected branch, quotient completion, host stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness premise.*

Proof. For the empty ledger, use Definition 3536.3. The accepted carrier already routes the Cauchy thread through B, K, S, R, E, H, P, N, so the invariant holds with no additional prefix-cylinder observation.

For the induction step, assume the invariant for a finite replay ledger Λ and extend it by one displayed prefix-cylinder observation. The new observation only refines the finite prefix row inside B; it does not create a new Cauchy-thread row, stream-window row, rational-tolerance row, or endpoint seal.

Reuse the same finite observation and tolerance rows. The $\mathsf{StreamName}^{\uparrow}$ row S reads the extended prefix window, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the same dyadic tolerance scale, and K keeps the Cauchy-thread route inside $\mathsf{CompleteMetric}^{\uparrow}$ before E seals the endpoint distance.

The structural rows H, P, N transport, source, and name the extended replay ledger without adding any carrier coordinate. Therefore each finite ledger prefix preserves the displayed route, and induction over Λ gives the stated cylinder-refinement invariant. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompleteness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3536.1, Definition 3536.2, and Definition 3536.3 seed the finite NameCert route for the usual complete ultrametric on Baire space.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sequence equality, quotient completion, choice-selected branches, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-cylinder projection, Cauchy-thread convergence routing, finite observation readback, rational tolerance readback, endpoint-distance sealing, and ledger nonescape for BEDC.Derived.BaireSpaceCompletenessUp.

Constructive story: A $\mathsf{BaireSpaceCompleteness}^{\uparrow}$ packet is generated from BaireSpace prefix cylinders, a CompleteMetric Cauchy-thread row, StreamName finite windows, RegSeqRat tolerance readback, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3537

A Concrete Naming Certificate for $\mathsf{MeasurableSpace}^{\uparrow}$

Following Caratheodory and Bourbaki measure theory, the $\mathsf{MeasurableSpace}^{\uparrow}$ packet records the carrier-level interface that pairs an underlying point row with a displayed measurable-event family before any measure, probability, Borel, or measurable-map consumer reads it. It sits between the sigma-algebra closure surface of chapter 5186, the generated-event boundary of chapter 3384, the measure rows of chapter 92, the probability rows of chapter 184, and the preimage ledger of chapter 3461. The packet names measurable-space data as finite BEDC rows; it is not a host powerset, completed measurable universe, quotient event algebra, countable-choice selector, or ambient measure object.

Definition 3537.1 (MeasurableSpace carrier)**.** *A* $\mathsf{MeasurableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, A, B, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-carrier row whose events are being named.*
2. *A is the* $\mathsf{SigmaAlgebra}^{\uparrow}$ *event-family row over X, including empty-event, complement, and countable-union closure readback.*
3. *B is the optional* $\mathsf{BorelSet}^{\uparrow}$ *generated-event row used when the measurable space is read from a topological or real-facing source.*
4. *E is the public event classifier, reading event membership and event comparison only through A and, when present, B.*
5. *H records componentwise* hsame *transport for the carrier, sigma-algebra, Borel, event-classifier, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a measurable-event request through X, A, B, E and then to the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{MeasurableSpace}^{\uparrow}$, $\mathsf{SigmaAlgebra}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MeasurableSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host powerset, completed sigma-algebra, quotient event equality, countable-choice enumeration of events, ambient measure, or external measurable-space instance.

Lean4 ✓ `BEDC.Derived.MeasurableSpaceUp.MeasurableSpaceTasteGate single carrier alignment`

Theorem 3537.2 (MeasurableSpace NameCert obligations)**.** *For every accepted* $\mathsf{MeasurableSpace}^{\uparrow}$ *carrier* $M = (X, A, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MeasurableSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for* X*, sigma-algebra admission for* A*, generated-event compatibility for* B *when a Borel source is present, event-classifier visibility for* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Every obligation routes through the sibling rows of $\mathsf{SigmaAlgebra}^{\uparrow}$*,* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, and* $\mathsf{MeasurableFunction}^{\uparrow}$ *before any downstream measurable-object read is public.*

Proof. Project the carrier of Definition 3537.1. The point row X supplies the underlying carrier, while A supplies the sigma-algebra closure rows that make event reads measurable.

When the source is Borel-facing, read the generated-event row B through chapter 3384. The event classifier E is then constrained to the same displayed event family, so it cannot leave the packet for a host powerset or completed measurable universe.

The structural rows H, C, P, N transport, replay, source, and name exactly this packet. Since no coordinate names quotient event equality, countable-choice enumeration, ambient measure data, or an external measurable-space instance, the listed rows exhaust the local NameCert obligations. □

Theorem 3537.3 (MeasurableSpace sigma-algebra handoff)**.** *Every* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*, or* $\mathsf{MeasurableFunction}^{\uparrow}$ *consumer that cites an accepted* $\mathsf{MeasurableSpace}^{\uparrow}$ *carrier must first read the* $\mathsf{SigmaAlgebra}^{\uparrow}$ *row* A*, the generated* $\mathsf{BorelSet}^{\uparrow}$ *row* B *when present, and the event classifier* E *over the same point row* X*. Thus measures receive measurable events before assigning values, probability spaces receive the same event family before normalization, and measurable functions receive source and target event families before reading preimage ledgers.*

The handoff exports no host powerset, completed sigma-algebra, quotient event algebra, countable-choice event selector, ambient measure completion, or hidden measurable-map carrier.

Proof. Start with the sigma-algebra row. By chapter 5186, event admission, complement, and countable-union closure are displayed inside the $\mathsf{SigmaAlgebra}^{\uparrow}$ packet before measure-facing rows consume them.

Next route generated events through chapter 3384 when the measurable space is Borel-sourced. The same event classifier E is used by chapter 92 and chapter 184, so value and normalization rows cannot precede measurable-event exposure.

Finally apply the preimage discipline of chapter 3461. A measurable-map consumer reads source and target event families before the preimage ledger, so the measurable-space handoff is a source interface for measurable functions rather than a hidden host function space or completed measure object. □

Falsifiable prediction: A $\mathsf{MeasurableSpace}^{\uparrow}$ consumer that assigns a measure value, probability normalization, or measurable-function preimage without first reading the displayed $\mathsf{SigmaAlgebra}^{\uparrow}$ row A and event classifier E refutes the carrier surface.

Independence witness: sigma-algebra, borelset, measure, probspace, measurable-function: no listed sibling supplies the full eight-row packet $M = (X, A, B, E, H, C, P, N)$, because each sibling contributes only closure, generation, value, normalization, or preimage data while this chapter binds the measurable-space carrier with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MeasurableSpace}^{\uparrow}}$**.**

Constructive story: A MeasurableSpace$^{\uparrow}$ packet is generated from an underlying carrier row, a sigma-algebra event-family row, an optional Borel generated-event row, an event classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MeasurableSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3537.1, Definition 3537.2, and Definition 3537.3 seed the finite measurable-space interface consumed by measure, probability, and measurable-function rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MeasurableSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure construction, probability normalization, random variables, completed sigma-algebras, quotient event algebras, countable-choice event selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sigma-algebra admission, Borel compatibility, event-classifier stability, transport, replay, provenance, and local naming for BEDC.Derived.MeasurableSpaceUp.

Chapter 3538

A Concrete Naming Certificate for $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$

Following Bishop and Bridges' constructive finite-subcover treatment for compact intervals, the $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ packet records the BEDC route that turns a located closed interval cover request into a displayed finite subcover through finite nets and positive-radius data. It sits over the located interval finite-net surface of chapter 1808, the compact-cover radius route of chapter 2301, the finite-net interval-cover packet of chapter 3471, the terminal real-name seal of chapter 38, and the dyadic tolerance rows of chapter 291. The packet names a located finite-subcover certificate; it does not claim unrestricted open-cover Heine–Borel compactness, selected subcovers from arbitrary covers, countable choice, quotient interval equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Its subcover read uses chapter 3471 and the located interval finite-net sibling as mandatory predecessors before any retained witness can reach the $\mathsf{Real}^{\uparrow}$ seal.

Definition 3538.1 (Located Heine–Borel subcover carrier)**.** *A* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, L, H, R, D, E, T, C, P, N)$$

with the following displayed rows:

1. *I is the located closed interval source row, exposing endpoint names, endpoint order, and locatedness only as finite* $\mathsf{Real}^{\uparrow}$ *histories.*
2. *F is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *finite-net row, listing the mesh scale and finite interval net used by the cover request.*
3. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *positive-radius row attached to the displayed finite cover or pointwise radius family.*
4. *H is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *sibling row that records the finite-net interval-cover route over the same interval, net, radius, dyadic, and real-seal data.*
5. *R is the retained-subcover row, listing exactly the finitely many cover witnesses selected by the displayed net and radius checks.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording mesh width, radius slack, endpoint containment, and finite neighborhood comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint, radius, and coverage reads.*

8. *T records componentwise* hsame *transport for the interval source, finite-net row, radius row, finite-net sibling row, retained-subcover row, dyadic ledger, real seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a located cover request through $I, F, L, H, R, D, E, T, P, N$.*

10. *P is the* Pkg *provenance over* LocatedHeineBorelSubcover$^{\uparrow}$, LocatedClosedIntervalFiniteNet$^{\uparrow}$, LebesgueNumberLem HeineBorelFiniteNet$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{LocatedHeineBorelSubcover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, L, H, R, D, E, T, C, P, N$. It has no coordinate for arbitrary open covers, unrestricted Heine–Borel compactness, selected subcover functionals, quotient interval representatives, countable choice, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedHeineBorelSubcoverUp.LocatedHeineBorelSubcoverTasteGate single carrier alignment`

Theorem 3538.2 (Located Heine–Borel subcover NameCert obligations)**.** *For every accepted* LocatedHeineBorelSubcover$^{\uparrow}$ *carrier*

$$S = (I, F, L, H, R, D, E, T, C, P, N),$$

the local NameCert$_{\text{LocatedHeineBorelSubcover}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the located interval source, finite-net row, positive-radius row, finite-net sibling row, retained-subcover row, dyadic ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *located interval exposure, requiring every cover request to display I and the finite-net row F before any retained cover witness is inspected;*

3. *radius compatibility, requiring L and D to record the positive-radius and dyadic slack checks that allow listed net points to control nearby interval points;*

4. *finite-subcover exactness, requiring R to list only cover witnesses reached through the finite net, radius ledger, and* HeineBorelFiniteNet$^{\uparrow}$ *sibling route;*

5. *real-seal non-escape, requiring E to be downstream of the located interval, finite-net, radius, retained-subcover, and dyadic rows;*

6. *structural preservation through T, C, P, N and refusal of arbitrary open-cover compactness, unrestricted Heine–Borel, selected subcover functionals, quotient interval representatives, countable choice, host real equality, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No located finite-subcover obligation is stored outside these rows.

Proof. Project the carrier of Definition 3538.1. The analytic rows are exactly the located interval source I, finite-net row F, positive-radius row L, finite-net sibling row H, retained-subcover row R, dyadic ledger D, and terminal real seal E.

The cover request first exposes I and F. The packet therefore reads a located interval and a displayed finite net before it can inspect any cover witness, so the route cannot start from an arbitrary open-cover selector.

The radius step is finite. The row L supplies the positive radius ledger, while D records the mesh-width and endpoint-slack comparisons that transfer a witness at a listed net point to nearby interval points.

The retained row R is downstream of F, L, H, D. Hence every retained cover witness is tied to the same finite-net route already named by chapter 3471, and the terminal seal E is reached only after those finite checks have been displayed.

Transport, replay, provenance, and local naming preserve this finite surface. Since the refused objects have no carrier coordinate, the local certificate exports only the located finite-subcover route. □

Theorem 3538.3 (Located Heine–Borel finite-net route)**.** *Every located cover read from an accepted* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{LebesgueNumberLemma}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The exported object is the retained finite subcover row R over $I, F, L, H, D, E, T, C, P, N$, not an unrestricted open-cover Heine–Borel principle.

Proof. Begin with an accepted carrier $S = (I, F, L, H, R, D, E, T, C, P, N)$. The located interval and finite net are exposed by I and F, the positive-radius ledger is L, and the already displayed finite-net cover route is H.

The retained-subcover row R may be read only after those rows. Its witnesses are checked by the dyadic ledger D and then sealed by E, so the exported finite subcover is a displayed BEDC packet row rather than a host-level choice operation.

Replay through C, transport through T, provenance through P, and naming through N keep the same route. No arbitrary open cover, unrestricted compactness theorem, quotient interval representative, countable-choice row, or ambient completeness coordinate is available to bypass it. □

Theorem 3538.4 (Located Heine–Borel subcover sibling route)**.** *Any accepted read of a* $\mathsf{LocatedHeineBorelSubcover}$ *retained subcover row must pass through chapter 3471 and the located interval finite-net sibling before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Equivalently, in the carrier $S = (I, F, L, H, R, D, E, T, C, P, N)$, the row R is readable only after the displayed finite-net row F and the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *sibling row H have exposed the same interval, radius, dyadic, and cover-witness surface.*

Proof. Project the carrier of Definition 3538.1. The located interval finite-net row is F, the finite-net sibling row is H, and the retained-subcover row is R.

Apply Definition 3538.2. Its located interval exposure and finite-subcover exactness clauses require every retained witness to be downstream of F, L, H, D before the real-seal row E is available.

Finally apply Definition 3538.3. The displayed route places $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ between the positive-radius row and the dyadic/Real seal, so a located subcover consumer cannot read R by bypassing either sibling. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$ packet is generated from a located closed interval source, a LocatedClosedIntervalFiniteNet row, a LebesgueNumberLemma positive-radius ledger, a HeineBorelFiniteNet sibling row, retained finite subcover witnesses, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3538.1, Definition 3538.2, and Definition 3538.3 seed the located finite-subcover route over interval nets, positive-radius ledgers, finite-net cover rows, dyadic tolerances, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedHeineBorelSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted open-cover Heine–Borel compactness, arbitrary subcover selection, countable choice, quotient interval equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval exposure, radius compatibility, finite-subcover exactness, real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedHeineBorelSubcover

Chapter 3539

A Concrete Naming Certificate for MetaCICTypedNormalizationAudit$^{\uparrow}$

The MetaCICTypedNormalizationAudit$^{\uparrow}$ packet realizes the candidate-mediated typed-normalization route of chapter 9383 as finite BEDC rows. It sits beside the normalization audit packet of chapter 750, the candidate-evidence strong-normalization route of chapter 758, the Pi-application adequacy route of chapter 907, and the normalization frontier of chapter 892. The packet names the closed typed term, candidate evidence, Pi and application adequacy, no-infinite-reduction projection, frontier refusal, transport, replay, provenance, and local naming rows needed for an audit citation. It does not claim evidence-free strong normalization for every closed typed MetaCIC term, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Definition 3539.1 (MetaCIC typed-normalization audit carrier)**.** *A* MetaCICTypedNormalizationAudit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, K, S, P, R, Q, F, H, C, G, N)$$

with the following displayed rows:

1. *T is the closed typed term row, exposing the empty-context term, type, and closedness checks read by the normalization audit map.*
2. *K is the candidate-evidence row, carrying the reducibility candidate, closed-candidate, or candidate-set membership evidence consumed by the closed typed term projection.*
3. *S is the strong-normalization projection row, recording the candidate-mediated route from K to the displayed strong-normalization witness for T.*
4. *P is the Pi adequacy row, exposing Pi-candidate and neutral-normal evidence only when the inspected term route requires it.*
5. *R is the application adequacy row, exposing function-side Pi evidence, argument candidate evidence, and the application-candidate handoff.*
6. *Q is the no-infinite-reduction row, recording the projection from strong normalization to absence of infinite beta chains along the same candidate-mediated route.*
7. *F is the frontier row, retaining the boundary named by* MetaCICNormalizationFrontier$^{\uparrow}$ *when the requested read would erase candidate evidence or subject-reduction obligations.*

8. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, SN projection, Pi route, application route, no-infinite-reduction row, frontier row, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a typed-normalization audit request through $T, K, S, P, R, Q, F, H, G, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$*, chapter 9383, chapter 750, chapter 758, chapter 907, chapter 892,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, S, P, R, Q, F, H, C, G, N$. It has no coordinate for evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, erased candidate evidence, or completed closed consistency.

Theorem 3539.2 (MetaCIC typed-normalization audit NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ *carrier*

$$A = (T, K, S, P, R, Q, F, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the closed typed term row, candidate-evidence row, SN projection row, Pi adequacy row, application adequacy row, no-infinite-reduction row, frontier row, transport, replay, provenance, and local naming rows;*
2. *candidate gate exactness, requiring every SN projection in S to read candidate, closed-candidate, or candidate-set evidence from K before it can read the closed typed term row T as normalized;*
3. *Pi and application adequacy compatibility, requiring P and R to be consumed only as typed route data for the same candidate-mediated term window;*
4. *no-infinite-reduction compatibility, requiring Q to be downstream of S and to reuse the same candidate evidence rather than introducing a separate termination source;*
5. *frontier retention, requiring F to remain visible whenever a requested read would erase candidate evidence, subject-reduction obligations, confluence boundaries, or the normalization audit map's conditional interface;*
6. *structural preservation through H, C, G, N and refusal of evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, erased candidate evidence, completed closed consistency, Lean verification, and bridge checking.*

No typed-normalization audit obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.MetaCICTypedNormalizationAuditUp.MetaCICTypedNormalizationAuditNameCertObligations`

Proof. Project the carrier of Definition 3539.1. Its typed-normalization data are exactly the closed typed term row T, candidate-evidence row K, SN projection row S, Pi row P, application row R, no-infinite-reduction row Q, and frontier row F.

The candidate gate is first. A closed typed term can be read as strongly normalizing only after K supplies the candidate, closed-candidate, or candidate-set membership evidence consumed by the audit map.

The Pi and application rows refine that same route. The row P supplies Pi-candidate and neutral-normal data when the term route requires it, while R supplies function, argument, and application-candidate evidence for application cases. Neither row can replace the candidate gate.

The no-infinite-reduction row Q is downstream of S. It inherits the same candidate-mediated strong-normalization witness and therefore cannot introduce a separate global termination theorem.

The frontier row F remains visible for rejected readings. Transport, replay, provenance, and local naming preserve the listed rows, so the packet refuses evidence-free normalization, unrestricted subject reduction, global confluence, host evaluation soundness, quotient syntax equality, and closed consistency outside the displayed carrier. □

Theorem 3539.3 (MetaCIC typed-normalization substitution route)**.** *Every closed typed normalization read from an accepted* $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ *packet factors through the finite route*

$$T \longmapsto K \longmapsto S \longmapsto (P, R) \longmapsto Q \longmapsto F.$$

The exported route is candidate-mediated typed-normalization audit data over $A = (T, K, S, P, R, Q, F, H, C, G, N)$*, not a theorem that every closed typed MetaCIC term normalizes without supplied candidate evidence.*

Proof. Start with a closed typed term read. The row T exposes the typed and closed term window, but it does not by itself carry a normalization theorem.

The next row is K. It supplies the candidate evidence used by S to project strong normalization for that same closed typed term. The Pi and application rows P, R are read only as adequacy data for this candidate-mediated route.

The row Q then projects no-infinite-reduction along the same strong-normalization path. If a consumer asks for a reading that drops the candidate evidence or erases subject-reduction and confluence boundaries, the frontier row F records that refusal.

Thus every accepted read remains inside the displayed finite carrier and cannot be reread as evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency. □

Falsifiable prediction: Every $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ closed typed normalization read factors through displayed closed typed term data, candidate evidence, candidate-mediated strong-normalization projection, Pi adequacy rows, application adequacy rows, no-infinite-reduction projection, frontier retention, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacicnormalizationaudit, candidate-evidence-sn-route, pi-application-adequacy-route, metacicnormalizationfrontier: $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed typed term row, candidate gate, SN projection, Pi route, application route, no-infinite-reduction route, frontier retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}$ packet is generated from a closed typed term row, candidate-evidence row, candidate-mediated strong-normalization projection, Pi adequacy row, application adequacy row, no-infinite-reduction row, normalization-frontier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedNormalizationAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3539.1, Definition 3539.2, and Definition 3539.3 seed the finite candidate-mediated typed-normalization audit route over closed typed term, candidate, Pi, application, no-infinite-reduction, and frontier rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICTypedNormalizationAudit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization for every closed typed MetaCIC term, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate gate exactness, Pi and application adequacy compatibility, no-infinite-reduction compatibility, frontier retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedNormalizationAuditUp.

Chapter 3540

A Concrete Naming Certificate for $\mathsf{LocatedCompletionCriterion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ packet records the traditional located criterion by which completion-facing consumers recognize a displayed Cauchy name as admitted by finite tolerance windows. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric-completion carrier of chapter 1506, and the located complete-metric service of chapter 3349. The packet names the criterion route used by L10 Real and Completion consumers; it is not a quotient completion, selected limit, countable-choice schedule, trichotomy principle, host equality test, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 3540.1 (Located completion criterion carrier)**.** *A* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, E, M, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, endpoint comparisons, and error-budget rows inspected by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the requested observations of the Cauchy name and the local locatedness tests attached to those observations.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence obtained from the displayed windows and dyadic budgets.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been exposed.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row that records the completion-facing object surface for the same displayed Cauchy name.*
6. *L is the* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row, recording the located complete-metric limit service available for accepted regular Cauchy point names.*
7. *H records componentwise* hsame *transport for the dyadic tolerance ledger, stream windows, regular readback, Real seal, metric-completion carrier, located complete-metric service, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a criterion request through D, S, R, E, M, L.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{LocatedCompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, E, M, L, H, C, P, N$. It has no coordinate for quotient Cauchy equality, selected representatives, countable choice, host real equality, trichotomy, or completeness outside the displayed located complete-metric service.

Lean4 ✓ `BEDC.Derived.LocatedCompletionCriterionUp.LocatedCompletionCriterionTasteGate single carrier alignment`

Theorem 3540.2 (Located completion criterion NameCert obligations)**.** *For every accepted* LocatedCompletionCrit *carrier $K = (D, S, R, E, M, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance ledger, finite stream windows, regular rational readback, terminal Real seal, metric-completion carrier, located complete-metric service, componentwise transport, displayed continuation replay, package provenance, and local naming rows.*

Every completion-facing consumer must read the finite route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto M \longmapsto L \longmapsto H \longmapsto C \longmapsto P \longmapsto N$$

before treating the Cauchy name as admitted by the located completion criterion.

Proof. Project the carrier of Definition 3540.1. The accepted packet displays exactly the dyadic tolerance, stream-window, regular readback, Real seal, metric-completion, located complete-metric, transport, replay, provenance, and naming rows.

The analytic criterion begins with D and S. The dyadic ledger supplies the requested tolerance budget, and the stream row supplies the finite observations and locatedness tests made at that budget.

The readback and completion handoff are downstream. The row R repacks the inspected windows as regular rational data, the row E seals the Real-facing name, and the rows M and L expose only the displayed metric-completion and located complete-metric service attached to the same route.

Finally H, C, P, N preserve, replay, source, and name that route. Since the carrier contains no quotient equality, selected representative, countable-choice schedule, host equality test, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the local NameCert obligations. □

Theorem 3540.3 (Located completion criterion Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *packet reaches the terminal seal and completion-facing service only through the dyadic tolerance ledger D, finite* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E,* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier M,* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row L, componentwise transport H, displayed replay C, provenance P, and local name N.*

The route exports a located criterion for displayed completion data, not a quotient completion, selected host limit, countable-choice construction, host equality test, trichotomy principle, or ambient completeness theorem for all $\mathsf{Real}^{\uparrow}$ *names.*

Proof. Start with a completion-facing or Real-facing consumer. The only terminal Real coordinate in the carrier is E, and the only completion-facing rows are M and L.

The consumer must first inspect D, S, R. The dyadic tolerances, finite windows, and regular rational readback are the displayed data that make the criterion located before the seal is exposed.

After the seal, M and L record the metric-completion surface and located complete-metric service attached to the same displayed Cauchy name. The support rows H, C, P, N can transport and replay this path, but they cannot add quotient completion equality, selected limits, countable choice, host equality, trichotomy, or ambient completeness because none of those coordinates is present in the accepted carrier. □

Theorem 3540.4 (Located completion criterion sequential closure)**.** *Let $K = (D, S, R, E, M, L, H, C, P, N)$ be an accepted* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier. Any Real-facing sequential-closure read of K must pass through the dyadic tolerance windows in D, the finite* $\mathsf{StreamName}^{\uparrow}$ *observations in S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in R, the* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier row M, the* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row L, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E, with transport, replay, provenance, and local naming supplied only by H, C, P, N.*

Consequently the accepted carrier admits a Real-facing read only as a finite tolerance-window route. It does not export quotient completion equality, a selected limit, countable choice, trichotomy, host real equality, or an ambient completeness theorem for all displayed names.

Proof. Read the carrier through Definition 3540.1. The only rows available for sequential closure are $D, S, R, E, M, L, H, C, P, N$, and the finite observation route begins with the dyadic tolerance ledger and stream windows.

Apply the Real seal route of Definition 3540.3. That theorem forces every Real-facing or completion-facing consumer to reach E, M, L only after D, S, R have been exposed.

The closure read is therefore finite at every stage. The dyadic ledger supplies the tolerance windows, the stream row supplies the inspected observations, the regular rational row supplies readback, and the metric-completion and located complete-metric rows supply the completion-facing service before the terminal Real seal is consumed.

Finally H, C, P, N preserve the same route without adding coordinates. Since the carrier definition contains no quotient completion, selected limit, countable-choice schedule, trichotomy row, host equality test, or ambient completeness field, the sequential-closure read cannot export any of those non-claims. □

Theorem 3540.5 (Located completion criterion terminal seal exhaustion)**.** *Let $K = (D, S, R, E, M, L, H, C, P, N)$ be an accepted* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *carrier. Any Real-facing completion read from K exhausts exactly the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D,* $\mathsf{StreamName}^{\uparrow}$ *observations S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal E,* $\mathsf{MetricCompletion}^{\uparrow}$ *carrier M,* $\mathsf{LocatedCompleteMetric}^{\uparrow}$ *service row L, componentwise transport H, displayed replay C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}$ *row N. It has no remaining coordinate for selected limits, quotient equality, host equality, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the NameCert obligations of Definition 3540.2. They identify the displayed rows $D, S, R, E, M, L, H, C, P, N$ as the exact local certificate surface for an accepted carrier.

Use the Real seal route of Definition 3540.3. It forces every Real-facing or completion-facing consumer to pass through the dyadic ledger, stream windows, regular readback, terminal Real seal, metric-completion carrier, located complete-metric service, transport, replay, provenance, and local naming rows.

Finally apply sequential closure, Definition 3540.4. The completion read remains finite at each stage and cannot acquire coordinates absent from the carrier. Therefore exhausting the listed rows exhausts the accepted terminal-seal route. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ consumer factors a completion-facing read through one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal, one $\mathsf{MetricCompletion}^{\uparrow}$ carrier, one $\mathsf{LocatedCompleteMetric}^{\uparrow}$ service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, metriccompletion, located-complete-metric: $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries tolerance inspection, finite windows, regular readback, Real sealing, metric-completion admission, and located complete-metric service as one criterion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3540.1, Definition 3540.2, and Definition 3540.3 seed the finite located completion criterion route over dyadic tolerance ledgers, stream windows, regular rational readback, Real seals, metric-completion carriers, located complete-metric service, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletionCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window inspection, RegSeqRat readback, Real seal handoff, MetricCompletion carrier admission, LocatedCompleteMetric service, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompletionCriterionUp.

Constructive story: A $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal, a MetricCompletion carrier, a LocatedCompleteMetric service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3541

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$

Following Bishop–Bridges and Bourbaki-style constructive Cauchy-space grounding, the RegularCauchySequenceSpa packet records rate-controlled families of regular rational Cauchy names before any completion or equality quotient is read. It sits after the broader Cauchy-name surface of chapter 406, the finite schedule rows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite interface for regular-rate sequence-space consumers; it is not a proof that $\mathsf{Real}^{\uparrow}$ is complete, not a quotient of Cauchy streams, and not a selected limiting representative.

Definition 3541.1 (RegularCauchySequenceSpace carrier)**.** *A* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is a finite family of* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence rows.*
2. *ρ is the regular-rate ledger assigning dyadic Cauchy budgets to the rows of F.*
3. *σ is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that chooses the inspected sequence and finite window.*
4. *W records the finite common-window data used by the scheduled family inspection.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the requested regularity budget.*
6. *Q is the surrounding* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *row through which the same family can be read as a Cauchy-name-space object.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, ρ, σ, W, D, Q have been displayed.*
8. *H stores componentwise* hsame *transport for the family, rate ledger, schedule, window, tolerance, Cauchy-space, and real-seal rows.*
9. *C stores displayed* Cont *replay from finite rate requests to scheduled windows, regular rational readback, the CauchySequenceSpace handoff, and the terminal Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, \rho, \sigma, W, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceSpaceUp.RegularCauchySequenceSpaceTasteGate single carrier alignment`

Theorem 3541.2 (RegularCauchySequenceSpace NameCert obligations)**.** *For every accepted* RegularCauchySequen *carrier $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, regular-rate ledger exposure, scheduled finite-window stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of quotient streams, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Carrier admission is projection from Definition 3541.1. The accepted packet exposes the regular rational family F, the regular-rate ledger ρ, the schedule σ, the finite window row W, and the dyadic tolerance row D before any Cauchy-space or Real-facing consumer can read the packet.

The Cauchy-name-space handoff is then forced through Q. The surrounding $\mathsf{CauchySequenceSpace}^{\uparrow}$ row can consume the same family, schedule, and tolerance data, but it receives them only as displayed rows of this finite carrier.

Finally the terminal seal E is downstream of F, ρ, σ, W, D, Q and is transported only by H and replayed only by C. The provenance row P and local naming row N close the certificate surface, while the absence of any additional carrier coordinate excludes quotient streams, selected limits, countable choice, host real equality, ambient completeness, Lean verification, and bridge checking. □

Theorem 3541.3 (RegularCauchySequenceSpace real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

Thus the consumer first reads regular rational sequence data, then a regular-rate dyadic budget, then a scheduled finite window, then the $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff, and only then the terminal real-name seal. The route exports no quotient of Cauchy names, no selected limiting representative, no host real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceSpaceUp.RegularCauchySequenceSpaceRealHandoff`

Proof. Begin with an accepted carrier $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$. The only regular sequence data available to a consumer are the rows F and ρ, and the only finite observation channel is the schedule and window pair σ, W together with the dyadic tolerance row D.

Use Definition 3541.2. It binds regular-rate exposure, scheduled window stability, readback, and tolerance exactness to the same finite packet.

The Cauchy-space read reaches Q only after those rows have been displayed, and the Real-facing read reaches E only after Q has been exposed. Since the carrier definition lists no quotient, selected limit, host real equality, or ambient completeness coordinate, every $\mathsf{Real}^{\uparrow}$ consumer route is exhausted by the displayed rows. □

Theorem 3541.4 (RegularCauchySequenceSpace rate-ledger stability)**.** *Let $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier. Any classifier-preserving replacement of the family row F, regular-rate ledger ρ, schedule σ, and finite window row W preserves the same dyadic tolerance row D, the same* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff row Q, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal row E whenever the replacement remains inside the local classifier determined by H, C, P, N.*

Thus a consumer may not replace the displayed rate ledger by quotient stream equality, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. The stable route remains the finite row sequence*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

Proof. Project the carrier rows from Definition 3541.1. The rows F, ρ, σ, W are the only family, rate, schedule, and finite-window data available to the local classifier.

Apply Definition 3541.2. It identifies regular-rate ledger exposure, scheduled finite-window stability, tolerance exactness, CauchySequenceSpace handoff, and Real seal routing as obligations over the same displayed carrier.

Now apply Definition 3541.3. Real-facing consumers see only the displayed route through D, Q, E after the replacement has preserved the classifier on F, ρ, σ, W. Since quotient stream equality, selected limits, host real equality, and ambient completeness are not carrier coordinates, the replacement preserves the same tolerance, handoff, and terminal seal rows rather than substituting any of those refused objects. □

Theorem 3541.5 (RegularCauchySequenceSpace tail-zero route)**.** *Let $R = (F, \rho, \sigma, W, D, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *carrier. If the regular-rate ledger ρ carries an explicit dyadic tail-to-zero modulus, then every* $\mathsf{Real}^{\uparrow}$*-facing consumer still factors through*

$$F, \rho, \sigma, W, D, Q, E, H, C, P, N.$$

The tail-zero conclusion is a displayed rate-ledger read over ρ, D, Q, E; it is not a selected limit, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier rows from Definition 3541.1. The family F, rate ledger ρ, schedule σ, window W, dyadic row D, CauchySequenceSpace handoff Q, and Real seal E are the only data available before the structural rows transport and name the packet.

Apply Definition 3541.2. It binds regular-rate exposure, scheduled finite-window stability, regular rational readback, dyadic tolerance exactness, the CauchySequenceSpace handoff, and the Real seal to the same displayed packet.

Now apply Definition 3541.3 and Definition 3541.4. The Real-facing route remains the finite row sequence above, and the tail-to-zero modulus remains a rate-ledger entry rather than a selected limit or quotient-stream coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ packet is generated from a finite regular rational sequence family, a regular-rate dyadic ledger, scheduled finite stream windows, dyadic tolerance rows, a CauchySequenceSpace handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3541.1, Definition 3541.2, Definition 3541.3, and Definition 3541.4 seed the finite regular-rate sequence-space route from regular rational families to Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient Cauchy-name equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, regular-rate ledger exposure, scheduled finite-window stability, tolerance exactness, CauchySequenceSpace handoff, RealUp seal routing, and non-escape for BEDC.Derived.RegularCauchySe

Chapter 3542

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive completions and dense embeddings, the $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ packet records the finite dense-source face used by the Cauchy completion universal-property route. It sits between the universal-property surface of chapter 2319, the metric dense-image packet of chapter 3351, the regular-Cauchy dense route of chapter 1453, the Real admission route of chapter 3389, the regular rational source of chapter 301, the stream-window source of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet lets universal-property, completion-functor, and Real-completion consumers cite a dense source without importing quotient completion, selected limits, countable choice, or ambient real completeness.

Definition 3542.1 (CauchyCompletionDensitySource carrier)**.** *A* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, M, J, R, W, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *consumer row that reads the dense source before any extension or uniqueness row is available.*
2. *M is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row, exposing the source metric, completion carrier, dense image, separated comparison, uniform handoff, and regular rational readback used by metric completion consumers.*
3. *J is the* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *row, recording the regular Cauchy rational source-to-completion route.*
4. *R is the* $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$ *row, recording the admission of the same regular Cauchy source into the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *W is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name and its displayed modulus discipline.*
6. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source only at requested stages.*
7. *S is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording precision budgets, slack rows, and finite comparison data before the* $\mathsf{Real}^{\uparrow}$ *seal is read.*

8. *H records componentwise* hsame *transport for $U, M, J, R, W, Q, S, C, P, N$.*
9. *C records displayed* Cont *replay from dense-source requests through the metric dense image, regular Cauchy route, Real admission route, source windows, dyadic tolerance ledger, transport, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay paths. It has no coordinate for quotient representatives, selected completion limits, selected real representatives, countable choice, trichotomy, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDensitySourceUp.CauchyCompletionDensitySourceTasteGate`
`single carrier alignment`

Theorem 3542.2 (CauchyCompletionDensitySource NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *carrier $D = (U, M, J, R, W, Q, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDen}}$ *surface consists exactly of carrier admission for the universal-property consumer row, metric dense-image row, regular Cauchy dense route, Real admission route, RegSeqRat source, StreamName finite windows, DyadicRatCore tolerance ledger, componentwise transport, displayed replay, provenance, and local naming. A completion universal-property consumer must read M, J, R, W, Q, S as dense-source evidence before it can consume the extension and uniqueness rows of U.*

Proof. Project the carrier of Definition 3542.1. The tuple displays the universal-property consumer row, metric dense-image row, regular Cauchy dense route, Real admission route, regular rational source, stream windows, dyadic tolerance ledger, transport, replay, provenance, and local naming row.

The dense-source evidence is finite and source-facing. The metric dense-image row supplies the completion-side dense embedding, the regular Cauchy route supplies the rational source discipline, the Real admission route records the terminal seal, and the RegSeqRat, StreamName, and DyadicRatCore rows expose only requested source observations and tolerance budgets.

The universal-property row U is downstream of those rows. Its extension and uniqueness reads therefore consume displayed dense-source evidence rather than quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient completeness. □

Theorem 3542.3 (CauchyCompletionDensitySource universal route)**.** *Every universal-property, completion-functor, or* $\mathsf{Real}^{\uparrow}$*-completion consumer of an accepted* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *packet factors through the finite route*

$\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{RealCauchyDenseEmbedding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, S

The exported object is a dense-source packet. It is not a quotient completion, a selected limit operator, a countable-choice principle, a host equality comparison, or an ambient real-completeness theorem.

Proof. Begin with the local obligations of Definition 3542.2. Any consumer reaching the universal-property row must first read the metric dense image, regular Cauchy dense route, Real admission route, regular rational source, finite stream windows, and dyadic tolerance ledger.

The sibling packets determine the route. The metric dense-image row gives completion-facing density, the regular Cauchy dense route gives the source discipline, the Real admission route gives

the seal handoff, and the RegSeqRat, StreamName, and DyadicRatCore rows give finite observation and tolerance data.

Transport, replay, provenance, and local naming preserve the same displayed route. Since the carrier contains no row for quotient representatives, selected limits, countable choice, host equality, or ambient real completeness, the universal route exports only the dense-source packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ consumer that reaches a universal-property extension or uniqueness read while omitting the metric dense image, regular Cauchy dense route, Real admission route, RegSeqRat source, StreamName windows, DyadicRatCore tolerance ledger, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchy-completion-universal-property, metric-completion-dense-embedding, regular-cauchy-dense-embedding, real-cauchy-dense-embedding, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective with $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ because the packet jointly carries dense-source evidence, universal-property consumption, completion-facing dense image data, regular rational source discipline, finite windows, dyadic tolerance accounting, Real-seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ packet is generated from a universal-property consumer row, a metric dense-image row, a regular Cauchy dense route, a Real admission route, a RegSeqRat source row, finite StreamName windows, a DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3542.1, Definition 3542.2, and Definition 3542.3 seed the dense-source face consumed by constructive Cauchy completion universal-property routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDensitySource}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, selected limits, countable choice, trichotomy, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exactness, metric dense-image handoff, regular Cauchy source readback, Real-seal handoff, finite-window and tolerance accounting, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyCompletionDensitySourceUp.

Chapter 3543

A Concrete Naming Certificate for $\mathsf{SeparatedCompletionReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ packet records the finite reflection boundary where separated zero-distance evidence is allowed to feed completion uniqueness. It sits between the separated metric classifier of chapter 1411, the complete-limit rows of chapter 128, the metric distance surface of chapter 57, the terminal real-name seal of chapter 38, and the regular-Cauchy comparison route of chapter 1412. The packet names a reflection route for completion consumers without importing quotient completion, selected representatives, trichotomy, countable choice, or a claim that $\mathsf{Real}^{\uparrow}$ is complete.

Definition 3543.1 (SeparatedCompletionReflection carrier)**.** *A* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (M, S, C, E, Z, H, K, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier and the metric-distance witness surface that completion reads may inspect.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier row, read only after the metric distance witness has been displayed.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *pair of completion-limit rows for a single regular Cauchy point name, including the displayed limit modulus and convergence-distance ledgers.*
4. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route binding the two completion-limit reads to the same finite regular-Cauchy window data.*
5. *Z is the* $\mathsf{Real}^{\uparrow}$ *zero-distance read of the metric-distance witness between the two displayed completion limits.*
6. *H records componentwise* hsame *transport for M, S, C, E, Z, K, P, N.*
7. *K records displayed* Cont *replay from finite Cauchy windows to completion limits, from those limits to a metric-distance witness, and from the real zero-distance row to the separated classifier.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$ *row.*

The carrier is a reflection surface: it records how already displayed separated zero-distance evidence is consumed by completion uniqueness. It has no quotient-completion object, no representative selector, no trichotomy row, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.SeparatedCompletionReflectionUp.SeparatedCompletionReflectionTasteGate`
`single carrier alignment`

Theorem 3543.2 (SeparatedCompletionReflection NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *carrier* $R = (M, S, C, E, Z, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the* $\mathsf{Met}^{\uparrow}$ *source row, the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance classifier, the paired* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *comparison route, the* $\mathsf{Real}^{\uparrow}$ *zero-distance read, componentwise transport, continuation replay, provenance, and local naming;*
2. *completion-window exactness, requiring the two completion-limit reads in C to be compared only through the finite-window route E before any endpoint classifier is inspected;*
3. *zero-distance exactness, requiring the metric-distance witness from M to pass through the* $\mathsf{Real}^{\uparrow}$ *row Z before the separated classifier S is consumed;*
4. *transport and replay stability, requiring* hsame *transport through H and* Cont *replay through K to preserve the same order E, M, Z, S under local naming;*
5. *ledger visibility, requiring P and N to expose only the displayed rows and to refuse quotient completion, selected representatives, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3543.1. Its fields give exactly the metric source, separated classifier, complete-limit pair, Cauchy comparison route, real zero-distance read, and structural support rows named in the first obligation.

For completion-window exactness, read C through E. The two limit rows are thereby tied to the same regular-Cauchy window data before the metric-distance witness is formed, so no quotient completion or selected representative can supply an alternate comparison.

For zero-distance exactness, the metric row M supplies the distance witness, Z reads that witness at zero through the $\mathsf{Real}^{\uparrow}$ seal, and only then may S export the separated classifier. The transport and replay rows H, K preserve this same order componentwise.

The final obligation is the provenance inventory. Since P and N name only the rows listed in the carrier, the local certificate cannot expose trichotomy, countable choice, a quotient completion carrier, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3543.3 (SeparatedCompletionReflection route)**.** *Every completion-uniqueness consumer of an accepted* $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchySequenceEquivalence}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Concretely, the consumer must first read the paired complete-limit rows C, then the finite-window comparison route E, then the metric-distance witness from M, then the $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, and only afterward the separated classifier row S with support from H, K, P, N. The exported conclusion is completion uniqueness as a local separated-classifier route, not quotient completion equality, a selected limit, host equality, trichotomy, countable choice, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the paired $\mathsf{CompleteMetric}^{\uparrow}$ limit rows in C. The comparison route E binds them to the same regular Cauchy point name through finite windows before any endpoint comparison is available.

Next form the displayed metric-distance witness using M. The real-valued zero-distance read Z is then the only row that turns that witness into data consumable by a separated classifier.

Finally apply the $\mathsf{SeparatedMetric}^{\uparrow}$ row S and carry the result through H, K, P, N. Transport, replay, provenance, and local naming do not add analytic coordinates; they only preserve and name the displayed route. Since the carrier contains no quotient-completion, selected-representative, trichotomy, countable-choice, or ambient-completeness field, the consumer route has no escape from the stated boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ packet is generated from a MetricUp source row, a SeparatedMetricUp zero-distance classifier, paired CompleteMetricUp limit rows, a CauchySequenceEquivalenceUp comparison route, a RealUp zero-distance read, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3543.1, Definition 3543.2, and Definition 3543.3 seed the completion-reflection boundary over separated zero-distance evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionReflection}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SeparatedCompletionReflectionUp.SeparatedCompletionReflectionTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, quotient completion, selected representatives, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-window exactness, zero-distance exactness, transport replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionReflectionUp.

Chapter 3544

A Concrete Naming Certificate for $\mathsf{ConvergentNet}^{\uparrow}$

Following Moore–Smith and Kelley, the $\mathsf{ConvergentNet}^{\uparrow}$ packet records ordinary convergent-net evidence as finite BEDC naming data. It sits over the directed-index substrate of chapter 1607, beside the Cauchy-tail surface of chapter 589, and before the convergence boundary of chapter 1521 and chapter 1561. The same finite route is readable by the completion consumers in chapter 1473 and chapter 1506, without turning convergence into a quotient of nets, a selected host limit, countable choice, or an ambient completeness assertion.

Definition 3544.1 (ConvergentNet carrier)**.** *A* $\mathsf{ConvergentNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, V, T, L, M, C, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index-window row, naming the inspected indices, tail windows, and displayed upper-window witnesses.*
2. *V is the net-value row over the indices displayed by D.*
3. *T is the convergence-test row, recording the finite neighborhoods, entourages, or metric tolerances that the packet actually inspects.*
4. *L is the displayed boundary row toward which the packet tests convergence, read only as finite naming data.*
5. *M is the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *boundary row that packages the same directed-window, value, test, and boundary evidence.*
6. *C is the optional* $\mathsf{CauchyNet}^{\uparrow}$ *tail row consumed by completion-facing readers that need Cauchy evidence.*
7. *H records componentwise* hsame *transport for the directed-window, value, test, boundary, convergence, Cauchy-tail, continuation, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from directed windows through net values, tests, boundary reads, Moore–Smith convergence, optional Cauchy-tail evidence, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{ConvergentNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithConvergence}^{\uparrow}$, $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient nets, selected host limits, countable-choice selectors, cofinal choice functions, host equality of boundary points, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ConvergentNetUp.ConvergentNetTasteGate single carrier alignment`

Theorem 3544.2 (ConvergentNet NameCert obligations)**.** *For every accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *carrier* $G = (D, V, T, L, M, C, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed directed index windows, net values, convergence tests, boundary row, Moore–Smith row, Cauchy-tail row, transport, continuation, provenance, and local naming;*
2. *directed-window coverage, requiring every convergence test in T to be attached to a tail window displayed by D;*
3. *value exposure, requiring each accepted test to inspect only net values displayed by V;*
4. *boundary exposure, requiring L to be read only after D, V, T have been displayed;*
5. *Moore–Smith packaging, requiring M to preserve the same finite route before any completion-facing consumer reads it;*
6. *completion handoff readiness, requiring the optional Cauchy-tail row C before* $\mathsf{CauchyNetLimit}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumes the packet;*
7. *ledger refusal, excluding quotient nets, selected host limits, countable choice, cofinal choice, host equality of limits, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3544.1. The tuple exposes directed windows, net values, convergence tests, a boundary row, the Moore–Smith convergence row, the optional Cauchy-tail row, and the structural rows H, K, P, N.

Directed-window coverage and value exposure are forced by the continuation row K. A convergence test is replayed first through D, then through V, and only then reaches T and the boundary row L.

The Moore–Smith row M packages the same displayed route. If a completion-facing reader also needs Cauchy evidence, it must consume the row C before the route can be handed to $\mathsf{CauchyNetLimit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, or $\mathsf{MetricCompletion}^{\uparrow}$.

Since the carrier has no coordinate for quotient nets, selected limits, choice operators, host equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3544.3 (ConvergentNet Cauchy handoff)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Any completion-facing consumer that reads G as Cauchy-directed evidence must first display the* $\mathsf{DirectedSet}^{\uparrow}$ *index substrate D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, and then the* $\mathsf{CauchyNet}^{\uparrow}$ *tail row C. The handoff preserves the finite directed-net route toward chapter 1561, chapter 1473, and chapter 1506, and exports no quotient net, selected limit, countable-choice selector, cofinal-choice operator, host equality, or ambient completeness principle.*

Proof. Start from a completion-facing read of the packet. Carrier admission gives the directed-window substrate D and the values V, but no completion consumer is visible before the convergence tests and boundary row have been replayed.

Use the continuation row K. It routes the read through D, V, T, L and then into the Moore–Smith row M, so the convergence evidence remains a finite directed-net display.

For a Cauchy-directed completion consumer, the row C is the required tail-evidence handoff. After C is displayed, the same finite route can be consumed by the directed-net limit, uniform-completion, or metric-completion surface.

The omitted coordinates are exactly the forbidden exits: quotient nets, selected limits, countable choice, cofinal choice, host equality, and ambient completeness. Hence the handoff preserves only the displayed finite route. □

Theorem 3544.4 (ConvergentNet completion-consumer boundary)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of G must read the* $\mathsf{DirectedSet}^{\uparrow}$ *row D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, and the optional* $\mathsf{CauchyNet}^{\uparrow}$ *tail-evidence row C in that order before any completion-facing handoff is available. The consumer receives only the finite route named by $D, V, T, L, M, C, H, K, P, N$ and receives no selected host limit, quotient net, countable-choice selector, cofinal-choice operator, host equality of boundary points, or ambient completeness principle.*

Proof. Project the carrier of Definition 3544.1. The only rows visible to a completion consumer are the directed-window row D, the value row V, the test row T, the boundary row L, the Moore–Smith row M, the optional Cauchy-tail row C, and the structural rows H, K, P, N.

Use Definition 3544.2. Its directed-window coverage, value exposure, boundary exposure, and Moore–Smith packaging clauses force the read order D, V, T, L, M before any completion-facing consumer can see convergence evidence.

Then apply Definition 3544.3. A metric-completion or uniform-completion consumer that needs Cauchy-directed evidence must next read C, so the completion-facing route is the displayed finite route through D, V, T, L, M, C.

Since the carrier omits selected host limits, quotient nets, choice operators, host equality, and ambient completeness, no completion consumer can bypass this order or turn the convergence packet into a selected limit object. □

Theorem 3544.5 (ConvergentNet limit handoff)**.** *Let $G = (D, V, T, L, M, C, H, K, P, N)$ be an accepted* $\mathsf{ConvergentNet}^{\uparrow}$ *packet. Every accepted limit read of G must expose the* $\mathsf{DirectedSet}^{\uparrow}$ *row D, the net-value row V, the convergence-test row T, the boundary row L, the* $\mathsf{MooreSmithConvergence}^{\uparrow}$ *row M, the optional* $\mathsf{CauchyNet}^{\uparrow}$ *row C, componentwise transport H, continuation replay K, provenance P, and the local* $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$ *row N before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer can read the packet as completion-facing data. The read exports the Moore–Smith boundary only through this finite row order and exports no quotient net, selected host limit, countable-choice selector, cofinal-choice operator, host equality of boundary points, or ambient completeness principle.*

Proof. Apply Definition 3544.2. Its carrier admission and exposure clauses display exactly $D, V, T, L, M, C, H, K, P,$ while its ledger refusal excludes quotient nets, selected limits, choice operators, host equality, and ambient completeness.

The limit-facing part is the Moore–Smith boundary row. Boundary exposure and Moore–Smith packaging force a read through D, V, T, L, M before the packet can be treated as a convergent-net boundary.

For completion consumers, apply Definition 3544.3. The optional Cauchy-tail row C is the only displayed Cauchy-directed handoff, and it is read after the Moore–Smith route rather than as a separate completion object.

Finally, use Definition 3544.4. It states that $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{UniformCompletion}^{\uparrow}$ consumers receive only the finite route named by the carrier rows, so the accepted limit read is exactly the Moore–Smith boundary row plus the listed structural support before completion consumption. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvergentNet}^{\uparrow}$ packet is generated from a DirectedSet index-window row, net values, convergence tests, a displayed boundary row, Moore–Smith convergence data, optional CauchyNet tail evidence, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3544.1, Definition 3544.2, Definition 3544.3, Definition 3544.4, and Definition 3544.5 seed the finite convergent-net surface between directed index data, Moore–Smith convergence, limit handoff, and completion-consumer handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvergentNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected host limits, countable choice, cofinal choice, host equality of boundary points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window coverage, value exposure, boundary exposure, Moore–Smith packaging, Cauchy handoff, provenance, and ledger refusal for the same finite packet.

Chapter 3545

A Concrete Naming Certificate for $\mathsf{MedianClosure}^{\uparrow}$

The $\mathsf{MedianClosure}^{\uparrow}$ packet realizes the coordinatewise-majority closure surface of Definition 9462.1319 and Definition 9462.1320 as finite NameCert data. It records the reassigned support, the majority-saturation row, inactive-corner readback, f-vector ledger, and maximal-cell ledger for the 20-codon WNR∪CUN carrier. It sits beside the sibling finite-structure packets in chapter 2309, chapter 349, and chapter 112, but is not bijective with them because it carries the coordinatewise-majority saturation row.

Definition 3545.1 (MedianClosure carrier)**.** *A* $\mathsf{MedianClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, M, I, F, X, H, C, P, N)$$

with the following displayed rows:

1. *S is the reassigned support row, naming the* 13 *codons of the source support $\mathcal{R}$.*
2. *M is the coordinatewise-majority saturation row, displaying exactly the* 20 *codons of* $\mathrm{WNR} \cup \mathrm{CUN}$ *as the median closure of S.*
3. *I is the inactive-corner readback row, displaying the seven codons in* $\mathrm{Med}(\mathcal{R}) \setminus \mathcal{R}$ *and the four codons in* $\mathrm{Med}(\mathcal{R}) \setminus \overline{\mathcal{R}}$.
4. *F is the f-vector ledger for $K(\mathrm{Med}(\mathcal{R}))$, displaying* $(20, 38, 26, 8, 1, 0, \ldots)$.
5. *X is the maximal-cell ledger, displaying the* WNR 4*-cube, the* CUN 2*-cube, and the listed bridge cells from the source vision.*
6. *H records componentwise* hsame *transport for support, median saturation, inactive-corner, f-vector, maximal-cell, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from support through coordinatewise-majority saturation, inactive-corner readback, f-vector ledger, maximal-cell ledger, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{MedianClosure}^{\uparrow}$, $\mathsf{HellyTheorem}^{\uparrow}$, $\mathsf{Trie}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$ *row.*

The carrier contains no coordinate for the full Q_6 convex hull, no Helly theorem, no cubical homology package, no trie dictionary order, and no abstract finite-set equality outside the displayed codon rows.

Theorem 3545.2 (MedianClosure NameCert obligations)**.** *For every accepted* $\mathsf{MedianClosure}^{\uparrow}$ *carrier* $Q = (S, M, I, F, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$ *surface consists exactly of support admission through* S*, coordinatewise-majority saturation through* M*, inactive-corner readback through* I*, f-vector exposure through* F*, maximal-cell exposure through* X*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MedianClosureUp.MedianClosureTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3545.1. The support row, median-saturation row, inactive-corner row, f-vector ledger, maximal-cell ledger, and structural rows are explicit coordinates of the same finite BHist packet.

The continuation row C fixes the order of reads. A consumer first sees the reassigned support S, then the coordinatewise-majority saturation row M, and only then the inactive-corner, f-vector, and maximal-cell rows.

Transport, provenance, and local naming preserve that same route. Since the carrier omits convex-hull completion, Helly theorem data, cubical homology, trie order, and abstract finite-set equality, the listed rows exhaust the local NameCert surface. □

Theorem 3545.3 (MedianClosure vision realization)**.** *Every accepted* $\mathsf{MedianClosure}^{\uparrow}$ *packet realizes the median-closure claims of Definition 9462.1319, Definition 9462.1320, Definition 9462.1321, Definition 9462.1323, and Definition 9462.1324 as one finite NameCert surface. In particular, the displayed closure-size row rejects any carrier whose median-saturation row is not the* 20*-codon set* $\mathrm{WNR} \cup \mathrm{CUN}$*.*

Proof. Read the support row S and the median-saturation row M. By carrier admission, M is the displayed coordinatewise-majority closure of the reassigned support and has the 20 codons $\mathrm{WNR} \cup \mathrm{CUN}$.

The inactive-corner row I and f-vector row F are downstream of M, so their readback is tied to the same displayed closure rather than to an independent finite set.

The maximal-cell row X uses that same closure to expose the WNR and CUN cells and their bridge cells. Therefore any carrier with a different median closure cannot satisfy the local NameCert route. □

Falsifiable prediction: The $\mathsf{MedianClosure}^{\uparrow}$ closure-size row rejects any carrier whose displayed median closure is not the 20-codon $\mathrm{WNR} \cup \mathrm{CUN}$ set.

Independence witness: helly theorem, cubical finite-support chapters, trie, finset: none is bijective with $\mathsf{MedianClosure}^{\uparrow}$ because this carrier alone binds the coordinatewise-majority saturation row to the inactive-corner readback, f-vector ledger, and maximal-cell ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MedianClosure}^{\uparrow}$ packet is generated from the reassigned support row, coordinatewise-majority saturation, inactive-corner readback, f-vector ledger, maximal-cell ledger, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$, seed)

Scope closed: Definition 3545.1, Definition 3545.2, and Definition 3545.3 seed the finite median-closure packet for the 20-codon $\mathrm{WNR} \cup \mathrm{CUN}$ carrier.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MedianClosure}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the full Q_6 convex hull, Helly theorem content, cubical homology, trie ordering, abstract finite-set equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support admission, majority saturation, inactive-corner readback, f-vector exposure, maximal-cell exposure, provenance, and ledger refusal for the same finite packet.

Chapter 3546

A Concrete Naming Certificate for $\mathsf{RegularCauchyReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyReal}^{\uparrow}$ packet presents a real number by regular Cauchy rational data with an explicit finite tolerance route before any quotient, selected limit, or ambient completeness principle is read. It sits on the finite observation surface of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the Cauchy-convergence handoff of chapter 1509, and the terminal real-name seal of chapter 38. The packet names the non-quotient regular Cauchy route used by L10 Real and Completion consumers; it does not assert that every Cauchy object has a chosen limit.

Definition 3546.1 (RegularCauchyReal carrier)**.** *A* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule listing the rational observations opened for each displayed tolerance request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic error rows, comparison bounds, and late-window inequalities.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled stream windows and dyadic tolerance ledger.*
4. *K is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff row, checking that the displayed regular readback satisfies the finite convergence criterion exposed to completion consumers.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, Q, K have all been displayed.*
6. *H records componentwise* hsame *transport for the stream schedule, dyadic ledger, regular readback, convergence handoff, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a regular-Cauchy-real request through S, D, Q, K, and E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, host real equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyRealUp.RegularCauchyRealTasteGate single carrier alignment`

Theorem 3546.2 (RegularCauchyReal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier*

$$R = (S, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$ *surface consists exactly of carrier admission, finite stream-window exposure, dyadic tolerance exactness, regular rational readback, Cauchy-convergence handoff, Real-seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3546.1. The accepted packet exposes the stream schedule S, dyadic tolerance ledger D, regular readback Q, convergence handoff K, terminal Real seal E, and structural rows H, C, P, N.

The obligation rows are then read in order. The finite windows of chapter 126 supply the observations, the dyadic comparisons of chapter 291 supply the tolerance ledger, and the readback surface of chapter 301 packages them as regular rational data.

The convergence and seal rows close the route without adding an ambient limit. The handoff through chapter 1509 exposes only the displayed criterion row K, and the Real seal of chapter 38 is reached only through E.

Finally H and C transport and replay the same coordinates, while P and N record provenance and local naming. Since no additional coordinate is present, the certificate surface exports none of the excluded quotient, choice, completeness, host-equality, verification, or bridge data. □

Theorem 3546.3 (RegularCauchyReal Real seal route)**.** *Every Real-facing read exported by an accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, Q, K \longmapsto E$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Thus a consumer receives scheduled stream windows, dyadic tolerance rows, regular rational readback, and the Cauchy-convergence handoff before any Real seal is visible.*

Proof. Start with an accepted carrier $R = (S, D, Q, K, E, H, C, P, N)$. By Definition 3546.2, the only Real-facing row is E, and it is downstream of the displayed stream, dyadic, regular-readback, and convergence rows.

The route cannot skip the finite observation stage because S is the only coordinate containing opened stream windows. It cannot skip the tolerance stage because D is the only coordinate containing dyadic comparison and late-window information.

The regular readback and convergence handoff are also forced. The row Q is the sole $\mathsf{RegSeqRat}^{\uparrow}$ readback, while K is the sole $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff available to a completion consumer.

Transport, replay, provenance, and local naming preserve the same route. Since H, C, P, N have no quotient, selected-limit, countable-choice, or ambient-completeness coordinate, every Real-facing read reaches E only through the displayed route. □

Theorem 3546.4 (RegularCauchyReal seal factorization)**.** *Every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *Real seal factors through the L10 four-object route*

$$\mathsf{StreamName}^{\uparrow}(S) \longmapsto \mathsf{DyadicRatCore}^{\uparrow}(D) \longmapsto \mathsf{RegSeqRat}^{\uparrow}(Q) \longmapsto \mathsf{CauchyConvergenceCriterion}^{\uparrow}(K) \longmapsto \mathsf{Real}^{\uparrow}(E)$$

before any consumer can expose the terminal $\mathsf{Real}^{\uparrow}$ *row. The factorization is carried by the same* H, C, P, N *support rows of the carrier, and it supplies no quotient equality of Cauchy names, no selected limit, no host real equality, and no ambient completeness principle.*

Proof. Project the accepted carrier $R = (S, D, Q, K, E, H, C, P, N)$ through Definition 3546.1. The rows S, D, Q, K, E are the only analytic rows available to a Real-facing consumer.

The sibling route is ordered by the carrier itself. The finite windows of chapter 126 are read from S, the tolerance comparisons of chapter 291 are read from D, and the regular rational readback of chapter 301 is read from Q.

Apply Definition 3546.3 and Definition 3546.5. The first theorem forces every Real-facing read through S, D, Q, K before E, while the second identifies K as the displayed Cauchy-convergence handoff rather than an ambient completeness theorem.

Finally use the support rows H, C, P, N. They transport, replay, source, and name exactly the displayed route, so the terminal $\mathsf{Real}^{\uparrow}(E)$ seal has no coordinate from which quotient equality, selected limits, host real equality, or ambient completeness could be recovered. □

Theorem 3546.5 (RegularCauchyReal Cauchy-criterion handoff)**.** *Every completion-facing request for an accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier reads the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row* K *only after the stream schedule* S*, dyadic tolerance ledger* D*, and regular rational readback* Q *have been displayed. The handoff therefore supplies a finite criterion witness, not an ambient proof that an arbitrary Cauchy object has a chosen Real limit.*

Proof. Project the carrier rows of Definition 3546.1. The row K is listed after S, D, Q and is defined as the convergence handoff for that displayed regular readback.

The sibling routes determine the inputs to K. The schedule row S gives the finite inspected windows, the dyadic ledger D supplies the tolerance comparisons, and the regular rational row Q packages those reads into the source expected by the convergence criterion.

Thus a completion-facing consumer cannot read K as a standalone completeness principle. The only accepted handoff is the displayed finite route through S, D, Q, K, transported by H, replayed by C, and named by P, N. □

Theorem 3546.6 (RegularCauchyReal Cauchy-modulus source lock)**.** *Every accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *packet must read its displayed Cauchy modulus or convergence budget before the* $\mathsf{RegSeqRat}^{\uparrow}$ *row can be sealed as* $\mathsf{Real}^{\uparrow}$*. Concretely, any Real-facing consumer of*

$$R = (S, D, Q, K, E, H, C, P, N)$$

must expose the $\mathsf{StreamName}^{\uparrow}$ *schedule* S*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *budget row* K*, with the same source discipline as the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *interface, before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed.*

Proof. Project the rows $S, D, Q, K, E, H, C, P, N$ from Definition 3546.1. The stream schedule S is the only row exposing finite observations, and D is the only row exposing dyadic tolerance and late-window comparisons.

The readback row Q is therefore source-locked to S and D. The convergence row K records the displayed Cauchy budget for that readback, matching the modulus-source role of chapter 292 while staying inside the carrier rows already present in the packet.

Only after S, D, Q, K have been read can the terminal row E serve as the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The seal is not an independent limit selector; it is downstream of the StreamName, DyadicRatCore, RegSeqRat, and Cauchy-budget route.

Finally H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no selected limit, quotient stream equality, countable-choice row, host equality, or ambient completeness coordinate, no such datum can enter the Real seal. □

Theorem 3546.7 (RegularCauchyReal non-escape)**.** *An accepted* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier exports no quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, host real equality, Lean verification, or bridge credential. Every exported read is confined to* $S, D, Q, K, E, H, C, P, N$.

Proof. Use the carrier definition to enumerate the available coordinates. They are exactly the stream schedule, dyadic ledger, regular readback, convergence handoff, Real seal, transport, replay, provenance, and local naming rows.

The obligation theorem identifies the NameCert surface with those same rows. Any proposed exported read must therefore be one of the displayed coordinates or a structural transport of them through H and C.

None of the displayed coordinates stores quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, host real equality, Lean verification, or bridge data. Hence no such datum can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3546.1, Definition 3546.2, Definition 3546.3, Definition 3546.5, Definition 3546.6, and Definition 3546.7 seed the non-quotient regular Cauchy real-name packet over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyRealUp.RegularCauchyRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance exactness, regular rational readback, convergence handoff, Real-seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchyRealUp.

Constructive story: A $\mathsf{RegularCauchyReal}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3547

A Concrete Naming Certificate for $\mathsf{GDeltaSet}^{\uparrow}$

Following Kechris and Engelking on Polish spaces and descriptive set theory, the $\mathsf{GDeltaSet}^{\uparrow}$ packet records the traditional countable-intersection-of-open-sets interface as finite BEDC NameCert data. It sits over the complete separable metric source of chapter 1490, the generated-set boundary of chapter 3384, the countable-base discipline of chapter 1901, and the indexed density surface of chapter 1566. The packet names the route by which Polish and Borel consumers inspect a displayed open-row schedule without importing a host subset, selected enumeration, classical complement closure, ambient countable choice, or a standard descriptive-set theorem.

Definition 3547.1 (GDeltaSet carrier)**.** *A* $\mathsf{GDeltaSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, B, S, W, O, H, C, L, N)$$

with the following displayed rows:

1. *P is the Polish and second-countable source row. It exposes the* $\mathsf{PolishSpace}^{\uparrow}$ *complete-separable metric packet, the* $\mathsf{SecondCountable}^{\uparrow}$ *base row, and the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *indexed-density handoff visible to the current consumer.*

2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *row that receives the generated-set handoff after the displayed open schedule has been inspected.*

3. *S is the countable open-row schedule. Each finite prefix names the open rows whose intersection is being offered, together with the schedule position and the source row over which the open row is read.*

4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row. It records exactly the schedule positions, base-cell indices, density indices, and finite query windows visible before the consumer reads the Borel row.*

5. *O is the open-set readback row, carrying the displayed open classifier, local basis witness, and source compatibility proof for every open row inspected through S and W.*

6. *H records componentwise* hsame *transport for the Polish source, Borel row, open schedule, stream windows, open readback, continuation replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a Polish or Borel query through P, S, W, O before the generated-set row B is consumed.*

8. *L is the* Pkg *provenance over* $\mathsf{GDeltaSet}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, B, S, W, O, H, C, L, N$. It has no coordinate for host subsets, chosen countable enumerations, ambient countable choice, quotient set equality, classical complement closure, or standard descriptive-set transfer theorems.

Lean4 ✓ `BEDC.Derived.GDeltaSetUp.GDeltaSetTasteGate single carrier alignment`

Theorem 3547.2 (GDeltaSet NameCert obligations)**.** *For every accepted* $\mathsf{GDeltaSet}^{\uparrow}$ *carrier* $G = (P, B, S, W, O, H, C, L, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Polish and second-countable source row, Borel row, open-row schedule, finite windows, open readback, transport, replay, provenance, and local naming;*
2. *countable schedule exposure, requiring every inspected intersection stage to appear as a displayed finite prefix of S before it reaches B;*
3. *finite-window stability, requiring each schedule position, base-cell index, density index, and query window to be read through W;*
4. *open-row readback, requiring each open component to pass through the classifier and local-basis data stored in O over the same source row P;*
5. *Borel and Polish handoff, requiring the generated-set row B to consume only the displayed route through P, S, W, O;*
6. *componentwise transport, requiring H to preserve the source, schedule, windows, open readback, Borel row, replay, provenance, and local name together;*
7. *continuation replay, requiring C to replay the same ordered route before any Polish or Borel consumer receives the packet;*
8. *provenance, requiring L to name exactly the displayed sibling packets and BEDC kernel rows used by the carrier;*
9. *local naming, requiring N to certify only the rows P, B, S, W, O, H, C, L;*
10. *ledger refusal, excluding host countable choice, quotient set equality, selected enumerations, host subsets, classical complement closure, and ambient descriptive-set theorems.*

No $\mathsf{GDeltaSet}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 3547.1. The tuple displays exactly the Polish and second-countable source row, Borel row, countable open-row schedule, finite stream windows, open readback row, transport, replay, provenance, and local naming.

The schedule exposure and finite-window obligations are read through S and W. A consumer sees only a finite prefix of the countable open schedule, and each schedule position, base-cell index, density index, and query window is inspected through the same displayed stream-window row.

Open-row readback is forced by O over the source row P. Thus each open component is tied to the Polish, second-countable, and indexed-density data before the generated-set row B can be consumed.

Finally H, C, L, N transport, replay, source, and name the same route. Since no listed row stores a host subset, chosen enumeration, countable-choice selector, quotient set equality, classical

complement closure, or standard descriptive-set theorem, the enumerated obligations exhaust the local NameCert surface. □

Theorem 3547.3 (GDeltaSet Polish-Borel handoff)**.** *Every Polish or Borel consumer of an accepted* $\mathsf{GDeltaSet}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{PolishSpace}^{\uparrow},\ \mathsf{SecondCountable}^{\uparrow},\ \mathsf{CountableDenseSubset}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{BorelSet}^{\uparrow}.$$

More precisely, the consumer first reads the source row P, then the schedule and finite windows S, W, then the open-row readback O, and only then the generated-set row B. The exported object is a finite NameCert packet for the displayed countable open-row route; it is not a host subset, a chosen enumeration, a classical complement-closure proof, an ambient standard descriptive-set theorem, or a quotient set equality.

Proof. Begin with a Polish or Borel consumer of the accepted packet. Carrier admission provides the source row P, but no generated-set read is available until the open schedule and window rows have been displayed.

Replay the continuation row C. The read moves through S and W, so every countable-intersection stage is consumed only as a finite visible prefix with finite schedule, base-cell, density, and query observations.

The open readback row O then ties the visible open components back to the Polish and second-countable source. After that readback, the Borel row B receives the generated-set handoff.

The remaining rows H, L, N preserve, source, and name the same route. Therefore every consumer factors through the displayed schedule, window, and open-row route, with no host subset, chosen enumeration, classical complement closure, standard descriptive-set theorem, or quotient set equality exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3547.1, Definition 3547.2, and Definition 3547.3 seed the finite GDeltaSet route over Polish, second-countable, countable-dense, StreamName, open-readback, Borel, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GDeltaSet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, selected countable enumerations, countable choice, quotient set equality, classical complement closure, standard descriptive-set theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable schedule exposure, finite-window stability, open-row readback, Polish-Borel handoff, transport, replay, provenance, local naming, and ledger refusal for BEDC.Derived.GDeltaSetUp.

Constructive story: A $\mathsf{GDeltaSet}^{\uparrow}$ packet is generated from a Polish and second-countable source row, a BorelSet row, a countable open-row schedule, finite StreamName windows, open-set readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3548

A Concrete Naming Certificate for $\mathsf{ClosedBallCompactness}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ClosedBallCompactness}^{\uparrow}$ packet records the theorem-facing finite certificate that every displayed closed ball in a proper metric carrier is compact. It sits between the closed-ball membership surface of chapter 1451, the proper-metric witness of **??**, the nested-ball completion route of chapter 3526, the compact-metric certificate of chapter 129, and the total-bounded plus complete-metric route through chapter 127 and chapter 128. The packet names the finite closed-ball compactness handoff only; it is not an open-cover compactness theorem, not a host Heine–Borel proof, not an arbitrary properness oracle, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3548.1 (Closed-ball compactness carrier)**.** *A* $\mathsf{ClosedBallCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, P, N, K, T, M, R, H, D, Q, A)$$

with the following displayed rows:

1. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row naming the center, radius, metric source, and closed membership comparison.*
2. *P is the* $\mathsf{ProperMetric}^{\uparrow}$ *row whose displayed closed-ball family contains the same center-radius window.*
3. *N is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *route available for Cauchy-center and descending-ball handoff reads.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness row attached to the displayed closed ball.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row exposed by the compact metric witness.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row exposed by the compact metric witness and by the proper-metric completion handoff.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for center, radius, distance, and compactness-window comparisons.*
8. *H records componentwise* hsame *transport for the closed-ball, proper-metric, nested-ball, compact-metric, totally-bounded, complete-metric, real-readback, replay, provenance, and naming rows.*
9. *D records displayed* Cont *replay from a closed-ball compactness request through B, P, N, K, T, M, R.*

10. *Q is the* Pkg *provenance over* ClosedBallCompactness$^{\uparrow}$, MetricClosedBall$^{\uparrow}$, ProperMetric$^{\uparrow}$, NestedClosedBall$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *A is the local* NameCert$_{\text{ClosedBallCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their displayed transports, their continuation routes, provenance, and local naming. It has no coordinate for open covers, selected subcovers, quotient metric points, a host compactness theorem, arbitrary properness beyond the shown ball, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.ClosedBallCompactnessUp.ClosedBallCompactnessTasteGate single carrier alignment`

Theorem 3548.2 (Closed-ball compactness NameCert obligations)**.** *For every accepted* ClosedBallCompactness$^{\uparrow}$ *carrier*

$$C = (B, P, N, K, T, M, R, H, D, Q, A),$$

the local NameCert$_{\text{ClosedBallCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric closed-ball row, proper-metric row, nested-ball route, compact-metric witness, totally-bounded row, complete-metric row, real-readback row, componentwise transport, displayed continuation replay, package provenance, and local naming; center-radius compatibility between B and the closed-ball family displayed by P; compact witness exactness through K; finite-net and complete-limit exposure through T and M; and ledger refusal for open-cover compactness, selected subcovers, quotient metric points, host compactness, arbitrary properness outside the displayed ball, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3548.1. The accepted packet displays the closed-ball row, the proper-metric source, the nested-ball route, the compact witness, the total-bounded and complete-metric witnesses, the real readback, and the structural rows.

Center-radius compatibility is forced before compactness is read. The row B names the closed-ball membership comparison, and P exposes the same center-radius window in its displayed proper-metric family, so a consumer cannot switch to another ball.

Compact witness exactness is the route through K. The finite-net face T and the limit-service face M are read only after K has named the compact metric witness attached to B.

The structural rows H, D, Q, A transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, open-cover compactness, selected subcovers, quotient point equality, host compactness, arbitrary properness outside the displayed ball, and ambient real completeness cannot enter the local NameCert surface. □

Theorem 3548.3 (Closed-ball compactness nested-ball handoff)**.** *Every nested-ball or completion-facing consumer of an accepted* ClosedBallCompactness$^{\uparrow}$ *carrier reads the closed ball through*

MetricClosedBall$^{\uparrow}$, ProperMetric$^{\uparrow}$, NestedClosedBall$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, CompleteMetric$^{\uparrow}$

The handoff therefore exposes the displayed closed-ball membership row, the proper-metric compact witness, the nested-ball Cauchy-center route, the finite-net face, and the complete-metric face, and it exports no open-cover compactness proof or choice-selected center.

Proof. Begin with the accepted carrier $C = (B, P, N, K, T, M, R, H, D, Q, A)$. The closed-ball request first reads B and the matching proper-metric row P.

The nested-ball route N is then available only as a displayed handoff. It supplies the descending-ball and Cauchy-center route used by completion consumers, while K supplies the compact metric witness attached to the same closed ball.

Finally T, M, R, H, D, Q, A expose the finite-net, limit-service, real-readback, transport, replay, provenance, and naming rows. Because the carrier has no selector coordinate, the consumer receives no open-cover compactness proof and no choice-selected center. □

Theorem 3548.4 (Closed-ball compactness nonescape)**.** *Every public read of an accepted* $\mathsf{ClosedBallCompactness}^{\uparrow}$ *packet factors through the carrier of Definition 3548.1 and the obligations of Definition 3548.2. In particular, the packet can export compactness for the displayed closed ball only as the route*

$$B, P, N, K, T, M, R, H, D, Q, A,$$

and cannot export open-cover compactness, a selected finite subcover, quotient equality of metric points, a host Heine–Borel theorem, arbitrary properness beyond the displayed center-radius window, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows B, P, N, K, T, M, R are the only mathematical rows available before structural replay, and the rows H, D, Q, A are the only transport, continuation, provenance, and naming rows.

Apply Definition 3548.2. The local NameCert surface requires the closed-ball row, proper-metric row, nested-ball route, compact witness, total-bounded face, complete-metric face, and real-readback face to be admitted together.

Any public compactness read must therefore pass through the same finite tuple. Since the tuple has no coordinate for the excluded objects, those objects cannot escape through the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBallCompactness}^{\uparrow}$ packet is generated from a MetricClosedBall row, ProperMetric closed-ball family row, NestedClosedBall route, CompactMetric witness, TotallyBounded finite-net row, CompleteMetric limit-service row, RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3548.1, Definition 3548.2, Definition 3548.3, and Definition 3548.4 seed the theorem-facing finite compactness certificate for displayed proper-metric closed balls.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBallCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient metric points, host Heine–Borel compactness, arbitrary properness beyond displayed closed balls, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center-radius compatibility, compact witness exactness, nested-ball handoff, finite-net and complete-limit exposure, nonescape, and TasteGate field faithfulness for BEDC.Derived.ClosedBallCompactnessUp.

Chapter 3549

A Concrete Naming Certificate for $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$

Following the finite-net proof of compact uniform continuity in chapter 9237, the $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ packet records the residual triangle-estimate surface that remains after a compact source, continuous-map pointwise modulus, and uniform-modulus candidate have already been displayed. It sits over the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the metric triangle estimate packet of chapter 1159, and the uniform-modulus consumer of chapter 369. The packet names the final source-and-target triangle ledger only; it is not a compactness theorem, not a pointwise-modulus selector, not a finite-net constructor, and not a host proof of Heine–Cantor.

Definition 3549.1 (Compact-source triangle estimate carrier)**.** *A* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, U, E, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the finite-net coverage and located-distance comparisons for the compact source metric.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose pointwise modulus and graph evaluations are inspected on the same compact source.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *candidate row produced by the finite-net lower-bound route.*
4. *E is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *residual row, recording the source triangle estimate around a selected net center and the target triangle estimate for the two image points.*
5. *S is the finite center-radius ledger that ties the selected compact-source probe to the pointwise modulus radius and to the uniform-modulus precision.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *comparison ledger for the epsilon-third, half-radius, and lower-bound inequalities used by E.*
7. *L is the local nonescape ledger saying that the estimate consumes only the displayed compact source, continuous-map graph, uniform-modulus candidate, triangle-estimate row, and rational comparisons.*
8. *H records componentwise* hsame *transport for the compact source, continuous map, uniform-modulus candidate, triangle estimate, center-radius ledger, rational comparison ledger, nonescape ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-uniform-continuity request through* K, F, U, E, S, R, L.
10. *P is the* Pkg *provenance over* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open cover, an external finite subset, a selected point outside the displayed probe row, a host supremum norm, or an independent Heine–Cantor theorem.

Lean4 ✓ `BEDC.Derived.CompactSourceTriangleEstimateUp.CompactSourceTriangleEstimateTasteGate`
`single carrier alignment`

Theorem 3549.2 (Compact-source triangle estimate NameCert obligations)**.** *For every accepted* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *carrier*

$$T = (K, F, U, E, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-map row, uniform-modulus candidate, metric triangle estimate, finite center-radius ledger, rational comparison ledger, local nonescape ledger, componentwise transport, displayed continuation replay, package provenance, and local naming; compact-source compatibility between K and the source row of F; triangle-estimate exactness through E; rational-bound exactness through R; and refusal of open covers, external finite subsets, hidden point selectors, host norms, and independent Heine–Cantor reasoning.*

Proof. Carrier admission is projection from Definition 3549.1. The accepted packet displays the compact source, continuous map, uniform-modulus candidate, triangle estimate, center-radius ledger, rational comparisons, and structural rows.

Compact-source compatibility is read first. The compact row K and continuous-map row F name the same source metric before any center-radius estimate in S can be consumed.

The residual estimate is the row E. It combines the source triangle comparison around the displayed compact-net center with the target triangle comparison for the two image points, while R records the epsilon-third and radius inequalities used by that comparison.

The rows L, H, C, P, N record nonescape, transport, replay, provenance, and naming. Since no excluded coordinate appears in the carrier, the packet cannot export an open cover, external finite subset, hidden selector, host norm, or independent Heine–Cantor proof. □

Theorem 3549.3 (Compact-source triangle estimate UniformModulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ *packet must read the compact source, continuous-map graph, uniform-modulus candidate, metric triangle estimate, finite center-radius ledger, rational comparison ledger, and nonescape ledger before it reads the structural rows*

$$H, C, P, N.$$

Consequently the consumer receives a final compact-source triangle residual over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *and* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, *rather than a new compactness or continuity principle.*

Proof. Start with the accepted carrier $T = (K, F, U, E, S, R, L, H, C, P, N)$. The uniform-modulus consumer can read U only after K and F have identified the compact source and graph surface.

The estimate then passes through S, R, E. The center-radius ledger ties the selected compact-source probe to the pointwise-modulus radius and uniform precision, the rational ledger records the required inequalities, and the metric triangle estimate row supplies the source and target triangle steps.

The nonescape ledger L and structural rows H, C, P, N preserve the same route. Thus the handoff is a residual triangle-estimate packet for an already displayed compact-uniform route, not an additional source of compactness or continuity data. □

Falsifiable prediction: Every accepted $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ read must expose a $\mathsf{CompactMetric}^{\uparrow}$ source row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph row over the same source, a $\mathsf{UniformModulus}^{\uparrow}$ candidate row, a $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ residual row, finite center-radius and rational comparison ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric-triangle-uniform-estimate: $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ is not bijective to any listed sibling because it binds the final compact-source triangle residual after the finite-net and pointwise-modulus data have already been displayed.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactSourceTriangleEstimate}^{\uparrow}$ packet is generated from a CompactMetric source row, ContinuousMap graph row, UniformModulus candidate row, MetricTriangleUniformEstimate residual row, finite center-radius ledger, RatUp comparison ledger, local nonescape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3549.1, Definition 3549.2, and Definition 3549.3 seed the final compact-source triangle residual consumed by the compact-uniform-continuity route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactSourceTriangleEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compactness, pointwise continuity, finite-net construction, selected centers outside displayed probes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source compatibility, triangle-estimate exactness, rational-bound exactness, UniformModulus handoff, nonescape, and TasteGate field faithfulness for BEDC.Derived.CompactSourceT

Chapter 3550

A Concrete Naming Certificate for $\mathsf{RealLineConnected}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLineConnected}^{\uparrow}$ packet records the finite connectedness surface of the real line as interval-chain and intermediate-value data visible before any host topological connectedness theorem is invoked. It sits over the real-name boundary of chapter 38, the interval connectedness route of chapter 1462, the intermediate-value bisection route of chapter 1402, the graph and modulus rows of chapter 130, and the dyadic tolerance ledger of chapter 291. The chapter names a constructive real-line connectedness packet for later continuous-image and completion consumers; it does not claim open-cover compactness, real trichotomy, quotient stream equality, selected paths, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3550.1 (Real line connected carrier)**.** *A* $\mathsf{RealLineConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, J, V, F, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *line-source row, exposing only terminal real-name reads through displayed finite windows;*
2. *J is the* $\mathsf{ConnectedInterval}^{\uparrow}$ *interval-chain row, recording the finite chain of located interval segments used by a connectedness consumer;*
3. *V is the* $\mathsf{IntermediateValue}^{\uparrow}$ *sign-route row attached to the interval-chain handoff;*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus row inspected by continuous-image consumers of the line route;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the interval-chain, bisection, and finite-window requests;*
6. *W is the finite observation-window row opened by D before any real-line connectedness read is exported;*
7. *Q is the regular rational readback row attached to the same windows;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after J, V, F, D, W, Q have been displayed;*
9. *H stores componentwise* hsame *transport for the real-source, interval-chain, intermediate-value, continuous-map, dyadic, window, readback, seal, replay, provenance, and local naming rows;*

10. *C stores displayed* Cont *replay through the same rows;*
11. *P is the* Pkg *provenance over* $\mathsf{RealLineConnected}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ConnectedInterval}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open-cover connectedness, trichotomy, quotient stream equality, a selected path object, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealLineConnectedUp.RealLineConnectedTasteGate single carrier alignment`

Theorem 3550.2 (Real line connected NameCert obligations)**.** *For every accepted* $\mathsf{RealLineConnected}^{\uparrow}$ *carrier*

$$L = (R, J, V, F, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed real-source, interval-chain, intermediate-value, continuous-map, dyadic, finite-window, readback, terminal-seal, transport, replay, provenance, and local naming rows;*
2. *interval-chain visibility, requiring every connectedness read to pass through J before a continuous-image or completion consumer can inspect the terminal seal;*
3. *intermediate-value routing, requiring each separation-refusal read along the chain to cite V and its finite bisection route;*
4. *continuous-map handoff, requiring downstream image consumers to use F only after the same interval-chain and intermediate-value rows have been displayed;*
5. *dyadic-window exactness, requiring the tolerance row D, window row W, and readback row Q to name the same finite request before E is exported;*
6. *nonescape from host open-cover connectedness, trichotomy, quotient stream equality, selected path objects, host real equality, and ambient completeness.*

No $\mathsf{RealLineConnected}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 3550.1. The tuple lists the real-source, interval-chain, intermediate-value, continuous-map, dyadic, finite-window, readback, terminal-seal, transport, replay, provenance, and local naming rows.

Interval-chain visibility is read from J. A connectedness consumer must inspect the finite $\mathsf{ConnectedInterval}^{\uparrow}$ chain before it can treat the real line as connected data.

Intermediate-value routing is read from V. The only separation-refusal information available to the packet is the finite intermediate-value sign route attached to the displayed interval chain.

Continuous-map handoff is read from F. A continuous-image consumer therefore receives graph and modulus data only after the line-source, interval-chain, and intermediate-value rows have already been displayed.

Dyadic-window exactness is read from D, W, Q, E. The dyadic tolerance opens the finite window, the readback row names the regular rational data in that window, and the terminal seal exports the same request as $\mathsf{Real}^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name only the displayed packet. Since the carrier contains no host open-cover connectedness, trichotomy, quotient stream equality, selected path object, host real equality, or ambient completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3550.3 (Real line connected interval-chain handoff)**.** *Every continuous-image or completion-facing read exported by an accepted* $\mathsf{RealLineConnected}^{\uparrow}$ *packet factors through*

$$\mathsf{ConnectedInterval}^{\uparrow} \longmapsto \mathsf{IntermediateValue}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route uses the displayed rows $J, V, F, D, W, Q, E, H, C, P, N$ *of Definition 3550.1. It exports no host open-cover connectedness theorem, trichotomy principle, quotient stream equality, selected path object, host real equality, or ambient completeness claim.*

Proof. Start with an accepted carrier $L = (R, J, V, F, D, W, Q, E, H, C, P, N)$. The source connectedness row is J, so any real-line connectedness read begins with the displayed $\mathsf{ConnectedInterval}^{\uparrow}$ chain rather than with a host connected-subset assertion.

Read the separation-refusal surface through V. This intermediate-value row records the finite bisection and sign-route data used by the chain; no trichotomy or selected separator is available.

Read the downstream continuous-map surface through F. The graph and modulus rows are attached to the same chain and sign-route data, so continuous-image consumers cannot bypass the interval-chain handoff.

Finally read the dyadic and real-name boundary through D, W, Q, E. The dyadic tolerance opens the finite window, the readback row presents the regular rational data, and the terminal seal exports the Real-facing datum.

The remaining rows H, C, P, N preserve the same displayed route. Since no row names open-cover compactness, quotient stream equality, selected paths, host real equality, or ambient completeness, every continuous-image or completion-facing read remains inside the finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3550.1, Definition 3550.2, and Definition 3550.3 seed the finite real-line connectedness route through interval chains, intermediate-value rows, continuous-map rows, dyadic windows, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineConnected}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLineConnectedUp.RealLineConnectedTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host open-cover connectedness, trichotomy, quotient stream equality, selected path objects, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, interval-chain visibility, intermediate-value routing, continuous-map handoff, dyadic-window exactness, and Real seal nonescape for BEDC.Derived.RealLineConnectedUp.

Constructive story: A $\mathsf{RealLineConnected}^{\uparrow}$ packet is generated from a Real source row, ConnectedInterval chain row, IntermediateValue sign route, ContinuousMap graph and modulus row, DyadicRatCore tolerance ledger, finite observation windows, regular readback, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3551

A Concrete Naming Certificate for $\mathsf{CauchyProductMetric}^{\uparrow}$

Following Bishop–Bridges on metric products and regular Cauchy names, the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet records the finite product-metric surface used by two displayed CauchyProduct rows before any host product metric or Real completeness principle is available. It sits beside the product route of chapter 464, the metric carrier surface of chapter 57, the regular rational readback of chapter 301, the scheduled window layer of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is the regular-Cauchy specialization consumed by chapter 424 when the product-distance ledger is read as a metric packet, and by chapter 1506 when paired Cauchy windows are passed to completion-facing consumers. The packet gives downstream Real, metric, and Cauchy-completion consumers a non-quotient product-distance certificate over displayed rows only.

Definition 3551.1 (CauchyProductMetric carrier)**.** *A* $\mathsf{CauchyProductMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two accepted* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows. Each row supplies a finite product-name route rather than host product equality.*
2. *W_L and W_R are the paired* $\mathsf{StreamName}^{\uparrow}$ *window rows inspected on the left and right product names.*
3. *Q_L and Q_R are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to the two displayed window pairs.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger. It records coordinate tolerances, the product-distance radius, the comparison budget, and the displayed error allowance used by the packet.*
5. *Δ is the* $\mathsf{Met}^{\uparrow}$ *distance row, read only through the displayed product windows, regular rational readbacks, and dyadic ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product-distance read.*
7. *H records componentwise* hsame *transport for the CauchyProduct rows, stream windows, regular readbacks, dyadic ledger, metric row, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a product-metric request through $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E$.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductMetric}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N$. It has no coordinate for host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyProductMetricUp.CauchyProductMetricTasteGate single carrier alignment`

Theorem 3551.2 (CauchyProductMetric NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *carrier $M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{CauchyProduct}^{\uparrow}$ *rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger,* $\mathsf{Met}^{\uparrow}$ *distance row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, local naming, and refusal of host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 3551.1. The accepted packet exposes exactly the two CauchyProduct rows, paired stream windows, regular rational readbacks, dyadic product-distance ledger, metric row, Real seal, and structural rows.

The product-distance ledger is downstream of the displayed windows and readbacks. Thus D cannot be read before W_L, W_R, Q_L, Q_R have fixed the finite CauchyProduct observations.

The metric and Real rows are then forced through Δ and E. The distance row reads only the displayed dyadic ledger, and the seal row records the resulting product-distance value without adding a second carrier.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host product metrics, quotient stream equality, selected limits, trichotomy, detached real carriers, or ambient completeness, the listed obligations exhaust the local certificate surface. □

Theorem 3551.3 (CauchyProductMetric projection stability)**.** *Let $M = (L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P,$ be an accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet. Any projection-facing metric consumer that reads the left or right coordinate distance must factor through the same* $\mathsf{CauchyProduct}^{\uparrow}$ *source row, the same* $\mathsf{StreamName}^{\uparrow}$ *window row, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger, and the same structural rows H, C, P, N. Projection reads may forget the unused coordinate, but they cannot introduce a host product metric, a quotient stream, a selected limit, or an independent real carrier.*

Proof. Start with a projection-facing read of the accepted packet. The only coordinate data exposed by the carrier are L, R together with their displayed window and regular readback rows.

For the left projection, the read passes through L, W_L, Q_L and then through the shared dyadic ledger D. For the right projection, the same argument uses R, W_R, Q_R and the same ledger.

The metric row Δ and Real seal E are downstream of this ledger. Forgetting one coordinate therefore removes data from the displayed packet, but it does not create a new metric source or a new Real-facing endpoint.

The structural rows H, C, P, N preserve exactly this projected route. The omitted coordinates in Definition 3551.1 rule out host product metrics, quotient streams, selected limits, and detached real carriers. □

Lean4 ✓ `BEDC.Derived.CauchyProductMetricUp.CauchyProductMetricProjectionStability`

Theorem 3551.4 (CauchyProductMetric Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet must first display the finite route*

$$\mathsf{CauchyProduct}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

through the rows $L, R, W_L, W_R, Q_L, Q_R, D, \Delta, E, H, C, P, N$*. The Real seal is a product-distance seal for this displayed route only; it exports no host product metric, no quotient product of Cauchy streams, no selected product limit, no trichotomy row, no detached real carrier, and no ambient completeness witness.*

Proof. Begin with a Real-facing read. Carrier admission makes the two CauchyProduct rows visible, but no Real seal is available before the paired stream windows and regular readbacks have been displayed.

The dyadic ledger D records the product-distance budget attached to those windows. The metric row Δ can only read this ledger, so it cannot import a host product metric or a separate distance object.

The seal E is reached after Δ, and the continuation row C replays the same route for downstream Real, metric, or completion-facing consumers. The transport and provenance rows keep the endpoint tied to $L, R, W_L, W_R, Q_L, Q_R, D, \Delta$.

Thus the consumer receives exactly the finite product-distance route named by the local certificate. All quotient, selection, trichotomy, detached-carrier, host-metric, and ambient-completeness exits are outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductMetric}^{\uparrow}$ packet is generated from two CauchyProduct source rows, paired StreamName windows, RegSeqRat readbacks, a DyadicRatCore product-distance ledger, a MetricUp distance row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3551.1, Definition 3551.2, Definition 3551.3, and Definition 3551.4 seed the finite product-metric route from CauchyProduct rows through displayed windows, regular readbacks, dyadic product-distance ledgers, metric rows, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host product metrics, quotient stream equality, selected product limits, trichotomy, detached real carriers, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-distance ledger exactness, projection stability, Real seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductMetricUp.

Chapter 3552

A Concrete Naming Certificate for $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$

Following Bishop and Bridges on uniformly continuous maps between Cauchy presentations, the $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ packet records the finite morphism surface between regular Cauchy sequence spaces before any completion equality or limiting representative is read. It sits after the source-space surface of chapter 3541, the finite window source of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the Cauchy-name-space packet of chapter 406, and the completion-functor route of chapter 423. The packet is the L10 finite morphism interface consumed by completion-functor and Real-facing handoff chapters; it is not a proof of ambient completeness, quotient equality of streams, selected limits, or host equality of real names.

Definition 3552.1 (RegularCauchySpaceMorphism carrier)**.** *A* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, T, A, W, D, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, including the source regular rational family, rate ledger, finite schedule, dyadic tolerance route, CauchySequenceSpace handoff, and Real seal.*
2. *T is the target* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, carrying the same visible sequence-space structure for the codomain presentation.*
3. *A is the finite-window action row assigning each accepted source window request to a displayed target window request.*
4. *W is the dyadic budget ledger recording the input precision, output precision, and finite modulus rows used by A.*
5. *D is the classifier-transport row, binding source acceptance to target acceptance through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling routes.*
6. *H records componentwise* hsame *transport for source space, target space, action, budget ledger, classifier transport, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from each morphism request through the source finite window, action row, dyadic budget ledger, target finite window, target CauchySequenceSpace handoff, and target Real seal.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$, $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, A, W, D, H, C, P, N$. It has no coordinate for quotient stream equality, selected source or target limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchySpaceMorphismUp.RegularCauchySpaceMorphismTasteGate single carrier alignment`

Theorem 3552.2 (RegularCauchySpaceMorphism NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ *carrier $M = (S, T, A, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$ *surface consists exactly of carrier admission, source-target compatibility, finite-window exactness, rate-ledger preservation, classifier transport, non-escape, displayed replay, provenance, and local naming.*

Source-target compatibility requires both S and T to be accepted $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *packets. Finite-window exactness requires A to transform only displayed source windows into displayed target windows. Rate-ledger preservation requires W to expose the dyadic input and output budget rows before any target seal is read. Classifier transport requires D to preserve the visible source and target classifiers through the same sibling routes named in the carrier. Non-escape refuses quotient stream equality, selected limits, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as certificate data.*

Proof. Project the carrier of Definition 3552.1. The rows S and T supply the source and target regular Cauchy sequence spaces, while A supplies only a displayed finite-window action between them.

Next read the dyadic budget and classifier transport rows. The budget ledger W records the precision rows consumed by the action, and D binds acceptance of the source and target windows through the named StreamName, RegSeqRat, DyadicRatCore, CauchySequenceSpace, and Real sibling routes.

Finally attach H, C, P, N. These rows preserve and replay the same finite morphism packet, so every local naming obligation is exhausted by the displayed tuple. Since the tuple has no quotient, selected-limit, countable-choice, host-equality, completeness, verification, or bridge coordinate, the refused objects cannot be exported by the NameCert surface. □

Theorem 3552.3 (RegularCauchySpaceMorphism finite-window exactness)**.** *Every accepted* RegularCauchySpaceM *read is exhausted by the displayed source and target regular Cauchy sequence-space rows, the finite-window action row, the dyadic budget ledger, classifier transport, componentwise transport, replay, provenance, and local naming:*

$$S, T, A, W, D, H, C, P, N.$$

Thus the morphism exports a finite L10 action between regular Cauchy sequence spaces. It exports no quotient stream equality, selected source limit, selected target limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted carrier $M = (S, T, A, W, D, H, C, P, N)$. The only source-space and target-space data available to the consumer are the displayed rows S and T, each read as a $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ packet.

Use Definition 3552.2. It places the action A behind finite windows and places the precision behavior behind the dyadic ledger W and classifier transport row D.

The structural rows H, C, P, N then transport, replay, source, and name only this route. No remaining coordinate can supply quotient stream equality, selected limits, countable choice, host real equality, or ambient real completeness, so every accepted morphism read is finite-window exact in the displayed sense. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySpaceMorphism}^{\uparrow}$ packet is generated from source and target RegularCauchySequenceSpaceUp rows, a finite-window action, a dyadic budget ledger, classifier transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3552.1, Definition 3552.2, and Definition 3552.3 seed the finite morphism surface between regular Cauchy sequence spaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySpaceMorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-target compatibility, finite-window exactness, rate-ledger preservation, classifier transport, non-escape, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.RegularCauchySpaceMorphismUp.

Chapter 3553

A Concrete Naming Certificate for $\mathsf{MetricCompletionComparison}^{\uparrow}$

Following Bishop and Bridges and Kelley on constructive metric completions, the $\mathsf{MetricCompletionComparison}^{\uparrow}$ packet records the finite comparison route by which two dense completions of the same metric source are read as the same completion surface. It sits after the universal-property surface of chapter 2319, the dense-source row of chapter 3542, the metric dense-image packet of chapter 3351, the regular Cauchy sequence-space packet of chapter 3541, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names constructive uniqueness of dense metric completions through displayed comparison rows; it is not a quotient representative theorem, not a selected-limit operator, not countable choice, and not an ambient completeness assertion for $\mathsf{Real}^{\uparrow}$.

Definition 3553.1 (MetricCompletionComparison carrier)**.** *A* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row used twice, once to extend the first dense embedding into the second completion and once in the opposite direction.*
2. *D is the* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *row that fixes the common dense source before either comparison map is read.*
3. *E_0 and E_1 are* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *rows for the two displayed dense completions of the same metric source.*
4. *S is the* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row recording the common regular Cauchy sequence-space interface inspected by both completions.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite regular rational windows of S.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the source, the two dense embeddings, and the comparison maps.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the precision budgets used to compare the two completion reads.*

8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* D, E_0, E_1, S, R, W, Q *have been displayed.*
9. *H records componentwise* hsame *transport for the dense-source, dense-embedding, sequence-space, readback, window, tolerance, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the two comparison directions and of the round-trip reads through the same finite rows.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionComparison}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N$ and their displayed replay paths. It has no coordinate for quotient representatives, selected Cauchy limits, host equality of completion classes, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricCompletionComparisonUp.MetricCompletionComparisonTasteGate single carrier alignment`

Theorem 3553.2 (MetricCompletionComparison NameCert obligations)**.** *For every accepted* MetricCompletionCo*
carrier $K = (U, D, E_0, E_1, S, R, W, Q, A, H, C, P, N)$, the local $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the universal-property row, common dense-source row, paired dense embeddings, regular Cauchy sequence-space row, regular rational readback, stream-window schedule, dyadic tolerance ledger, terminal real seal, componentwise transport, displayed replay, provenance, and local naming. The two comparison directions must read the same D, S, R, W, Q, A rows before either round-trip read is accepted.*

Proof. Project the carrier of Definition 3553.1. The rows $D, E_0, E_1, S, R, W, Q, A$ display the common source, the paired dense embeddings, the regular Cauchy sequence-space interface, the finite readback windows, the dyadic tolerances, and the terminal real-name seal.

Apply the universal-property row U in the two displayed directions. The first use extends E_0 across the dense source into the second completion surface, and the second use extends E_1 back across the same dense source. Both uses are forced to read the same dense-source, sequence-space, regular-readback, stream-window, tolerance, and seal rows.

The structural rows H, C, P, N then preserve, replay, source, and name that finite route. Since the carrier contains no row for quotient representatives, selected limits, host equality, countable choice, trichotomy, or ambient real completeness, no additional NameCert obligation can be projected from the packet. □

Theorem 3553.3 (MetricCompletionComparison universal route)**.** *Every constructive comparison between the two dense completions named by an accepted* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetricCompletionDenseEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionDensitySource}^{\uparrow},\ \mathsf{RegularCauchySequenceSpace}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

The exported result is the displayed pair of comparison maps with finite round-trip evidence on the common dense source. It is not a quotient-completion equality proof, a selected representative, a selected limiting operator, a countable-choice principle, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the obligation surface of Definition 3553.2. A completion-comparison consumer must first read the paired dense-embedding rows E_0, E_1 and the common dense-source row D.

The finite observation route is then fixed by S, R, W, Q, A. The regular Cauchy sequence-space row supplies the shared source discipline, the regular rational and stream-window rows supply the inspected finite windows, the dyadic row supplies the precision ledger, and the real seal is reached only after those rows are displayed.

Finally use the two universal-property reads stored in U. They produce comparison maps in both directions and check the round-trip evidence only on the displayed dense source. Transport and replay through H and C, with provenance and local naming through P, N, preserve the same route and export no excluded host-side principle. □

Theorem 3553.4 (MetricCompletionComparison triangle route)**.** *Let two accepted* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *packets compare a first dense completion surface to a middle dense completion surface and the same middle surface to a third dense completion surface over one displayed dense metric source. Then every completion-facing read from the first surface to the third surface factors through the two comparison packets in sequence: the common* $\mathsf{CauchyCompletionDensitySource}^{\uparrow}$ *row, the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *row, the relevant* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *rows, the shared* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and the two local* $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$ *rows.*

The composite route exports only the finite comparison read obtained by passing through the middle surface. It exports no quotient representative, selected limit, host equality of completions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 3553.2 to the first-to-middle packet and again to the middle-to-third packet. The two projections expose the same obligation surface: dense source, universal-property row, dense embeddings, regular Cauchy sequence-space interface, regular rational readback, stream windows, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming.

Next apply Definition 3553.3 to each comparison packet. The first application gives the route from the first completion surface to the middle surface, while the second gives the route from the middle surface to the third surface. Both routes are constrained to read the displayed dense source and the same universal-property discipline before any completion-facing read is accepted.

Compose the displayed routes along the middle completion surface, using the shared dense-source and universal-property rows as the overlap. The componentwise hsame row transports the common finite rows across the overlap, and the Cont row replays the two comparison reads in sequence without adding a new observation coordinate.

The carrier of Definition 3553.1 has no field for quotient representatives, selected limits, host equality of completion classes, countable choice, or ambient real completeness. Therefore the composite route can export only the finite first-to-third comparison read assembled from the two displayed packets. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionComparison}^{\uparrow}$ consumer that reaches a comparison map or round-trip read while omitting the common dense-source row, either dense-embedding row, the regular Cauchy sequence-space row, the RegSeqRat readback, the StreamName windows, the DyadicRatCore tolerance ledger, the Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchy-completion-universal-property, cauchy-completion-density-source, metric-completion-dense-embedding, regular-cauchy-sequence-space, regseqrat, streamname, dyadicratcore, real: no listed sibling is bijective with $\mathsf{MetricCompletionComparison}^{\uparrow}$ because the packet

jointly carries paired dense embeddings, a common dense source, two universal-property comparison directions, shared finite sequence-space readback, tolerance accounting, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionComparison}^{\uparrow}$ packet is generated from a CauchyCompletionUniversalProperty row, a CauchyCompletionDensitySource row, two MetricCompletionDenseEmbedding rows, a RegularCauchySequenceSpace row, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3553.1, Definition 3553.2, Definition 3553.3, and Definition 3553.4 seed the finite comparison route for constructive uniqueness of dense metric completions over a common source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, selected limits, countable choice, trichotomy, host equality of completion classes, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired dense-embedding exposure, common dense-source exactness, two universal-property comparison directions, round-trip evidence on the displayed source, finite readback and tolerance accounting, Real-seal routing, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.MetricCompletionComparisonUp.

Chapter 3554

A Concrete Naming Certificate for $\mathsf{BaireTwoFunction}^{\uparrow}$

Following Baire and classical descriptive set theory, the $\mathsf{BaireTwoFunction}^{\uparrow}$ packet records the successor step from Baire-one real-valued maps to pointwise limits of Baire-one approximants as finite BEDC NameCert data. It sits over the $\mathsf{BaireOneFunction}^{\uparrow}$ surface of chapter 1604, the finite schedules of chapter 126, the regular rational readback of chapter 301, and the terminal real-value seal of chapter 38. The packet names only the displayed approximation ladder and its finite read route; it is not a host function-space quotient, a selected limiting function, a measurability theorem, or an ambient completeness assertion.

Definition 3554.1 (BaireTwoFunction carrier)**.** *A* $\mathsf{BaireTwoFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, A, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BaireOneFunction}^{\uparrow}$ *approximation-ladder row, listing the Baire-one packets whose pointwise observations are being inspected.*
2. *A is the finite ladder-admission row, naming the displayed source points, approximation indices, and precision requests visible to the successor packet.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite approximation windows opened for each displayed request.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational windows obtained from the selected Baire-one observations.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Baire-one ladder, displayed schedule, and regular rational readback have been replayed.*
6. *H records componentwise* hsame *transport for the Baire-one ladder, finite admission row, schedule windows, rational readback, real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a pointwise successor request through the Baire-one ladder, the stream schedule, the rational readback, the real seal, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{BaireTwoFunction}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^\uparrow}$ *row.*

The carrier has no coordinate for a host space of Baire-two functions, an arbitrary selected approximating ladder, quotient equality of real names, full Borel hierarchy closure, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireTwoFunctionUp.TasteGate.BaireTwoFunctionTasteGate single carrier alignment`

Theorem 3554.2 (BaireTwoFunction NameCert obligations)**.** *For every accepted* $\mathsf{BaireTwoFunction}^\uparrow$ *carrier* $T = (B, A, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Baire-one approximation ladder, ladder-admission row, schedule windows, rational readback, real seal, componentwise transport, displayed continuation, provenance, and local naming rows;*
2. *Baire-one ladder exposure, requiring every successor observation to pass first through a displayed* $\mathsf{BaireOneFunction}^\uparrow$ *row in B;*
3. *schedule exactness, requiring every approximation request to use only the finite* $\mathsf{StreamName}^\uparrow$ *windows named by S after the ladder-admission row A is displayed;*
4. *readback exactness, requiring the value read to pass through the* $\mathsf{RegSeqRat}^\uparrow$ *row Q before the* $\mathsf{Real}^\uparrow$ *seal R is available;*
5. *transport and replay stability, requiring H and C to preserve the same route B, A, S, Q, R componentwise;*
6. *ledger refusal, excluding host function-space quotients, selected limiting functions, arbitrary ladder choice, quotient equality of real names, full Borel hierarchy closure, and ambient* $\mathsf{Real}^\uparrow$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3554.1. The rows B, A, S, Q, R are the only analytic rows available to a consumer, and H, C, P, N are the only structural rows attached to them.

The successor step begins with the Baire-one ladder B. The admission row A fixes the displayed source points, approximation indices, and precision requests before any schedule window can be opened.

The schedule and readback rows then force the finite route. A request reads the $\mathsf{StreamName}^\uparrow$ windows in S, checks the selected observations through the $\mathsf{RegSeqRat}^\uparrow$ row Q, and reaches the $\mathsf{Real}^\uparrow$ seal R only after those rows have been replayed.

Transport, continuation, provenance, and local naming preserve exactly this route. Since no carrier coordinate stores a host function space, selected limiting function, arbitrary ladder choice, quotient real equality, full Borel hierarchy closure, or ambient real completeness, the listed rows exhaust the NameCert surface. □

Theorem 3554.3 (BaireTwoFunction Baire-one handoff)**.** *Every* $\mathsf{BaireOneFunction}^\uparrow$ *approximation consumed by an accepted* $\mathsf{BaireTwoFunction}^\uparrow$ *packet is read before the* $\mathsf{StreamName}^\uparrow$ *schedule,* $\mathsf{RegSeqRat}^\uparrow$ *readback, and* $\mathsf{Real}^\uparrow$ *seal. Thus every successor value exported by the packet factors through*

$$B, A, S, Q \longmapsto R.$$

The handoff exposes a finite pointwise successor route from Baire-one data to real seals and exports no host function-space quotient, selected limiting function, arbitrary ladder selector, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer of the value seal R. By carrier admission, R is downstream of the same packet that contains the Baire-one ladder B, the admission row A, the schedule S, and the readback row Q.

Replay the continuation row C. The consumer first reads a displayed $\mathsf{BaireOneFunction}^{\uparrow}$ approximation from B, then uses A to identify the pointwise request and the finite approximation window available to the packet.

The schedule row S selects only those displayed windows, and the regular rational row Q reads the selected observations before R is exposed. Componentwise transport by H preserves this order for every accepted carrier.

The provenance and local naming rows contain no host function-space quotient, selected limiting function, arbitrary ladder selector, quotient real equality, or ambient completeness coordinate. Hence the exported successor value is only the displayed finite Baire-one handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3554.1, Definition 3554.2, and Definition 3554.3 seed the finite Baire-two successor packet over Baire-one ladders, finite schedules, rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireTwoFunction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BaireTwoFunctionUp.TasteGate.BaireTwoFunctionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host function-space quotients, selected limiting functions, arbitrary ladder choice, quotient equality of real names, full Borel hierarchy closure, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-one ladder exposure, schedule exactness, rational readback exactness, real-seal nonescape, transport stability, provenance, and ledger refusal for BEDC.Derived.BaireTwoFunctionUp.

Constructive story: A $\mathsf{BaireTwoFunction}^{\uparrow}$ packet is generated from a BaireOneFunction approximation ladder, finite request admission, StreamName schedule windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3555

A Concrete Naming Certificate for $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$

The $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ packet realizes the Alexander-dual trigger and blocker algebra of subsection 9462.28.21 as finite BEDC NameCert data. It sits over the finite $\mathsf{Bool}^{\uparrow}$ tag discipline of chapter 32, the finite observation interface of chapter 1007, the DNA boundary-tile carrier of chapter 3393, the six-tile coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the finite coverage-trigger algebra of chapter 3478. The packet names the blocker-dual surface as finite rows; it is not a polynomial-ring oracle, stochastic biological model, quotient codon semantics, infinite trigger search, or hidden Alexander-duality principle.

Definition 3555.1 (AlexanderDualTriggerAlgebra carrier)**.** *An* $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, T, I, U, L, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the boundary-tile carrier row read through* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *and* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$.
2. *T is the trigger row recording the finite coordinate supports whose full-trigger behavior is visible.*
3. *I is the inactive-face row recording the complements that remain blocked by a missed coordinate.*
4. *U is the blocker-dual cover row, carrying the four displayed dual generator faces and their finite lcm-degree ledger.*
5. *L is the* $\mathsf{Bool}^{\uparrow}$ *and finite-observation ledger that tags present, absent, trigger, blocker, active, and inactive faces without importing host truth values.*
6. *R is the transport row from boundary tiles through* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *and* WindowCoverageTriggerAlgeb *to the displayed blocker-dual cover.*
7. *H records componentwise* hsame *transport for boundary tiles, trigger rows, inactive faces, blocker-dual cover rows, finite tags, algebra transport, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each dual-trigger read through B, T, I, U, L, R, H.*
9. *P is the* Pkg *provenance over* $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{FiniteObservationInterface}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, WindowCoverageTriggerA *the Alexander-dual trigger source vision,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, T, I, U, L, R, H, C, P, N$. It has no coordinate for host polynomial equality, ambient monomial ideals, stochastic evolutionary dynamics, quotient codon equality, infinite trigger search, or a hidden duality oracle.

Theorem 3555.2 (AlexanderDualTriggerAlgebra NameCert obligations)**.** *For every accepted* AlexanderDualTrigger *carrier $A = (B, T, I, U, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}}$ *surface consists exactly of boundary-tile admission, trigger-row exposure, inactive-face exposure, blocker-dual cover exactness, finite Bool and observation tagging, algebra transport, componentwise transport, displayed replay, provenance, and local naming.*

The blocker-dual cover row U is accepted only when its trigger and blocker faces are read through the displayed B, T, I, L, R route. Any consumer that attempts to read Alexander-dual data without those rows, or that replaces them by host polynomial equality, an ambient ideal, quotient codon semantics, stochastic dynamics, infinite search, or a hidden duality oracle, leaves the accepted packet surface.

Proof. Project the carrier of Definition 3555.1. The rows B, T, I, U, L, R expose the boundary-tile source, trigger supports, inactive faces, blocker-dual cover, finite tag ledger, and transport from the boundary-trigger algebra.

The cover row is finite. It records only the four displayed dual generator faces and their lcm-degree ledger, with $\mathsf{Bool}^{\uparrow}$ and finite-observation rows marking which faces are present, absent, active, inactive, triggering, or blocking.

Finally H, C, P, N preserve the same displayed route. Since no host polynomial equality, ambient ideal, stochastic model, quotient codon equality, infinite search, or hidden duality oracle is a carrier coordinate, the local NameCert obligations are exactly the listed finite rows. □

Theorem 3555.3 (AlexanderDualTriggerAlgebra blocker-duality nonescape)**.** *Every accepted* AlexanderDualTrigge *read factors through displayed trigger, blocker, inactive-face, dual-cover, transport, replay, provenance, and local* NameCert *rows:*

$$B, T, I, U, L, R, H, C, P, N.$$

In particular, a dual-trigger consumer cannot replace the blocker-duality row by the $\mathsf{Bool}^{\uparrow}$ *tag discipline alone, by finite observation alone, by boundary closure alone, by trigger-hypergraph reliability alone, by universal coverage-trigger algebra alone, or by any host polynomial-ring equality.*

Proof. Read the boundary source first. The row B supplies the boundary tiles and coverage rows, while T and I separate visible trigger supports from inactive faces.

Next read the blocker-dual data. The row U carries the dual-cover faces, and L records the finite Bool and observation tags needed to distinguish trigger, blocker, active, and inactive positions.

The algebra transport R then ties those rows to the boundary-trigger closure and coverage-trigger algebra siblings. The structural rows H, C, P, N preserve the same route, so no sibling alone can absorb the blocker-duality coordinate and no host polynomial equality can be exported. □

Falsifiable prediction: Every accepted $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ read factors through displayed trigger, blocker, inactive-face, dual-cover, transport, replay, provenance, and local NameCert rows; omitting any of those rows refutes the packet.

Independence witness: bool, finite-observation-budget-selector, dna-boundary-trigger-closure, trigger-hypergraph-reliability, universal-coverage-trigger-algebra: no listed sibling is bijective with $\mathsf{AlexanderDualTriggerAlgebra}^{\uparrow}$ because the packet jointly carries trigger rows, blocker rows, inactive-face rows, dual-cover rows, finite tag ledgers, algebra transport, replay, provenance, and local

naming. A bijection preserving only finite tags, observation budgets, boundary closure, reliability, or coverage-trigger closure cannot preserve the blocker-duality row U together with the inactive-face row I.

The Lean follow-up `BEDC.Derived.AlexanderDualTriggerAlgebraUp.TasteGate` supplies explicit Nontrivial AlexanderDualTriggerAlgebra$^{\uparrow}$ and FieldFaithful AlexanderDualTriggerAlgebra$^{\uparrow}$ instances preserving $B, T, I, U, L, R, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$.

Constructive story: An AlexanderDualTriggerAlgebra$^{\uparrow}$ packet is generated from boundary-tile rows, trigger rows, inactive-face rows, blocker-dual cover rows, finite Bool and observation ledgers, algebra transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3555.1, Definition 3555.2, and Definition 3555.3 seed the finite Alexander-dual trigger and blocker NameCert packet.

Formal verification: FormalStatus(NameCert$_{\text{AlexanderDualTriggerAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host polynomial equality, ambient monomial ideals, stochastic biological dynamics, quotient codon equality, infinite trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-row exposure, blocker-dual cover exactness, inactive-face accounting, finite tag and observation ledgers, algebra transport, Nontrivial AlexanderDualTriggerAlgebraUp, and FieldFaithful AlexanderDualTriggerAlgebraUp for BEDC.Derived.AlexanderDualTriggerAlgebraUp.

Chapter 3556

A Concrete Naming Certificate for $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$

Following Moore–Smith and Kelley, the $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ packet records the finite criterion by which a directed Cauchy net exposes a convergence-facing subnet route without adding a selected subnet object or a choice principle. It sits over the directed-index surface of chapter 1607, the Moore–Smith carrier of chapter 3403, the Cauchy-net surface of chapter 589, the convergence interface of chapter 1667, the regular-rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a NameCert boundary for the L10 completion route: it records directed tails, Cauchy windows, subnet witnesses, and Real-facing readback rows, not a quotient of nets or an ambient compactness theorem.

Definition 3556.1 (Moore–Smith Cauchy subnet criterion carrier)**.** *A* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, M, A, W, B, N, S, G, Y, R, H, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row naming finite vertices, preorder comparisons, and upper-window witnesses.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *row exposing directed net values and the tail ledger consumed by the criterion.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *row recording the common-window Cauchy evidence read on the same directed surface.*
4. *W is the cofinal subnet-window row, listing the finite tail requests and their displayed upper refinements.*
5. *B is the subnet bound row, recording how each requested late window in M is met by a later displayed witness in W.*
6. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *consumer row that reads the subnet route only after D, M, A, W, B have been displayed.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for finite observation windows along the accepted subnet route.*
8. *G is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the scheduled observations.*

9. *Y is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those stream windows and dyadic tolerances.*
10. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, G, Y are displayed.*
11. *H records componentwise* hsame *transport for all criterion rows.*
12. *C records displayed* Cont *replay through the same directed-tail and readback route.*
13. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
14. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a globally chosen subnet, quotient-net equality, countable choice, an ambient compactness theorem, or host-level real completeness.

Lean4 ✓ `BEDC.Derived.MooreSmithCauchySubnetCriterionUp.MooreSmithCauchySubnetCriterionTasteGate`
`single carrier alignment`

Theorem 3556.2 (Moore–Smith Cauchy subnet criterion NameCert obligations)**.** *For an accepted carrier $Q = (D, M, A, W, B, N, S, G, Y, R, H, C, P, L)$ of Definition 3556.1, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCr}}$ *surface consists exactly of carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-window exposure, cofinal subnet-window evidence, subnet-bound compatibility, NetConvergence handoff, StreamName scheduling, DyadicRatCore tolerance comparison, RegSeqRat readback, Real seal exposure, componentwise transport, displayed route replay, provenance, and local naming.*

Proof. Carrier admission projects the finite packet of Definition 3556.1. The directed evidence is stored in D, M, A, W, B, so the criterion first reads the index row, Moore–Smith net row, Cauchy-net row, cofinal subnet windows, and subnet bounds.

The convergence-facing handoff is then N. By chapter 1667, a convergence consumer reads only displayed directed-tail evidence before it reaches filter or Real-facing rows.

The Real-facing part is S, G, Y, R. The StreamName, DyadicRatCore, RegSeqRat, and Real chapters supply finite observation windows, dyadic tolerances, regular-rational readback, and the terminal seal, so the packet does not need a hidden completed limit.

Finally H, C, P, L transport, replay, name, and certify exactly those rows. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Theorem 3556.3 (Moore–Smith directed-tail handoff)**.** *Every* $\mathsf{NetConvergence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ \mathsf{MooreSmithNet}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow} \longmapsto W, B \longmapsto \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{R}$$

The route supplies directed tails, Cauchy common-window evidence, subnet-window refinements, and finite readback rows only through the displayed carrier. It does not export a selected subnet function, quotient net, countable-choice selector, ambient compactness statement, or host-level completeness witness.

Proof. Start with the directed rows D, M, A. The $\mathsf{DirectedSet}^{\uparrow}$ row names upper-window witnesses, the $\mathsf{MooreSmithNet}^{\uparrow}$ row names the net-value and tail ledger, and the $\mathsf{CauchyNet}^{\uparrow}$ row names common-window evidence on that same directed surface.

The subnet criterion is the displayed pair W, B. The row W records the finite cofinal windows, while B records that each requested late Moore–Smith window is met by a later displayed witness.

The handoff to $\mathsf{NetConvergence}^{\uparrow}$ reads only after these rows are present. The downstream Real-facing rows S, G, Y, R are then the finite StreamName, DyadicRatCore, RegSeqRat, and Real seal route.

Transport, replay, provenance, and naming through H, C, P, L preserve the same carrier. Thus every accepted consumer factors through the displayed directed-tail handoff, and the excluded subnet-selection, quotient, choice, compactness, and host-completeness routes have no export row. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}$ packet is generated from directed indices, a Moore–Smith net row, a CauchyNet row, cofinal subnet windows, subnet-bound compatibility, a NetConvergence handoff, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3556.1, Definition 3556.2, and Definition 3556.3 seed the finite directed-tail subnet criterion used by L10 completion and compactness-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCauchySubnetCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subnets, quotient nets, countable choice, compactness, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-window exposure, cofinal-window compatibility, convergence handoff, Real-facing readback, transport preservation, and local naming for the displayed criterion packet.

Chapter 3557

A Concrete Naming Certificate for $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$

Following Moore–Smith and Kelley-style net convergence, the $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$ packet records the standard completeness criterion that every displayed Cauchy net on the metric surface has a displayed convergence endpoint. It sits beside the Moore–Smith net carrier of chapter 3403, the Cauchy-net surface of chapter 589, the convergence interface of chapter 1667, the completeness row of chapter 128, and the Cauchy-subnet criterion of chapter 3556. The packet is a finite completion-facing NameCert route: it names directed tails, Cauchy evidence, endpoint convergence, regular-rational readback, and the terminal real seal without asserting countable choice, quotient equality, or an ambient host-completeness oracle.

Definition 3557.1 (Moore–Smith completeness criterion carrier)**.** *A* $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, M, A, N, E, K, S, G, R, Y, H, C, P, L)$$

with the following displayed rows:

1. *D is the directed Moore–Smith index row, exposing finite comparison windows and common upper-window witnesses before any Cauchy request is read.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *row over D, recording net values, tail windows, and the directed route consumed by the criterion.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *evidence row, recording the common-window Cauchy estimates on the same Moore–Smith directed surface.*
4. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *endpoint row, exposing the displayed candidate endpoint and the convergence windows that the Cauchy evidence hands to the consumer.*
5. *E is the endpoint-admission row, recording that the endpoint inspected by N is the same completion-facing object read by the net consumer.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completeness row, used only as the displayed service that accepts the Cauchy-net evidence and returns the endpoint convergence row.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule for finite observation windows along the Cauchy and convergence reads.*

8. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream windows and the displayed convergence estimates.*
9. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the scheduled Cauchy and endpoint estimates.*
10. *Y is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream, regular-rational, and dyadic rows have been displayed.*
11. *H records componentwise* hsame *transport for the directed surface, Moore–Smith net row, Cauchy evidence, convergence endpoint, completeness row, readback rows, routes, provenance, and local naming.*
12. *C records displayed* Cont *replay through the same Cauchy-net-to-convergence route.*
13. *P is the* Pkg *provenance over* $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
14. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, quotient equality of nets, a selected family of limits, an ambient host-completeness oracle, or completeness of $\mathsf{Real}^{\uparrow}$ *outside the displayed readback route.*

Theorem 3557.2 (Moore–Smith completeness criterion NameCert obligations)**.** *For an accepted carrier $Q = (D, M, A, N, E, K, S, G, R, Y, H, C, P, L)$ of Definition 3557.1, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithCompletenessCri}}$ *surface consists exactly of carrier admission, directed Moore–Smith index exposure, net-value and tail exposure, CauchyNet evidence, NetConvergence endpoint admission, endpoint compatibility, CompleteMetric service exposure, StreamName scheduling, RegSeqRat readback, DyadicRatCore tolerance comparison, Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Carrier admission projects the finite packet of Definition 3557.1. The directed and net rows D, M must be visible before the Cauchy row A can be inspected.

The endpoint-facing rows are N, E, K. The $\mathsf{CompleteMetric}^{\uparrow}$ service row K accepts the displayed CauchyNet evidence and exposes the $\mathsf{NetConvergence}^{\uparrow}$ endpoint row N, while E records that the same endpoint is read by the convergence consumer.

The numerical readback rows are S, G, R, Y. They provide finite stream windows, regular-rational readback, dyadic tolerance comparisons, and the terminal real-name seal for the displayed convergence estimates.

Finally H, C, P, L transport, replay, source, and name the same rows. Because no other coordinate occurs in the carrier, no NameCert obligation can appeal to countable choice, quotient equality, a selected family of limits, an ambient host-completeness oracle, or a separate $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 3557.3 (Moore–Smith completeness criterion Cauchy-net handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$ *packet that starts from a displayed* $\mathsf{CauchyNet}^{\uparrow}$ *row must factor through*

$$\mathsf{MooreSmithNet}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicR}$$

The handoff supplies endpoint convergence as finite NameCert data and exports no countable-choice selector, quotient equality of nets, selected family of limits, ambient host-completeness oracle, or standalone $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the Moore–Smith and Cauchy rows. The row M records the directed net windows, and A records the common-window Cauchy estimates on the same directed surface.

The completeness service is the displayed row K. It can be read only after M and A are present, and it returns the endpoint-facing convergence row N together with the endpoint compatibility row E.

The readback route is finite. Rows S, G, R, Y record the observation windows, regular-rational estimates, dyadic tolerance comparisons, and terminal real-name seal attached to the endpoint convergence data.

Transport, replay, provenance, and local naming through H, C, P, L preserve this route. The excluded choice, quotient, selected-limit-family, host-completeness, and standalone real-completeness principles have no export row in the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}$ packet is generated from directed Moore–Smith net rows, CauchyNet evidence, NetConvergence endpoint rows, a CompleteMetric service row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance comparisons, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3557.1, Definition 3557.2, and Definition 3557.3 seed the finite Moore–Smith criterion by which displayed Cauchy-net evidence hands a convergence endpoint to completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithCompletenessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient equality of nets, selected families of limits, ambient host-completeness, standalone $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, directed-net exposure, CauchyNet evidence, CompleteMetric service exposure, NetConvergence endpoint compatibility, finite readback routing, transport preservation, and local naming for the displayed criterion packet.

Chapter 3558

A Concrete Naming Certificate for $\mathsf{ReassignmentSupportCompression}^{\uparrow}$

The $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ packet realizes the codon reassignment support compression branch of the DNA six-bit boundary tile analysis as finite NameCert data. The support row is read through the six-bit codon window of chapter 2234 and the boundary-tile coverage surface of chapter 3470; the compression packet therefore consumes those displayed coordinate and coverage rows before it names motif and prime-cover data. It packages the displayed 13-codon support, five-motif formula, seven prime implicants, and unique five-motif prime cover from Definition 9462.1274 through finite transport and replay rows, rather than treating the support as a host-side Boolean simplification oracle.

Falsifiable prediction: Every accepted $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ read factors through displayed codon support, motif-expansion, prime-implicant, minimal-cover, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, stopcodonzeckendorfsuffix, codon-transition-matrix, bitvector: none is bijective to $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ because this packet simultaneously binds the reassigned 13-codon support, five-motif expansion, seven prime implicants, unique minimum prime cover, transport, replay, provenance, and local naming rows. The six-bit window carrier supplies coordinates without the support theorem; the Zeckendorf suffix row names a stop-codon boundary; the transition matrix records table movement; and a raw bitvector lacks motif and cover obligations.

Definition 3558.1 (ReassignmentSupportCompression carrier)**.** *A* $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ *carrier is a finite packet*

$$K = (S, M, I, U, H, C, P, N)$$

with the following visible rows:

1. *S is the displayed reassigned support row containing exactly the* 13 *codons of Definition 9462.1275;*
2. *M is the five-motif expansion row recording* CUN, AGR, UNA, WWA, *and* UAR *as the support formula;*
3. *I is the prime-implicant row recording exactly the seven maximal exact subcubes of Definition 9462.1280;*
4. *U is the minimal-cover row recording the unique five-prime cover of Definition 9462.1281;*

5. *H stores componentwise* hsame *transport for the support, motif, prime-implicant, minimal-cover, provenance, and local naming rows;*
6. *C stores the displayed* Cont *replay route from codon support to motif expansion, from motif expansion to prime implicants, and from prime implicants to the minimum-cover row;*
7. *P is the* Pkg *provenance over the DNA six-bit boundary tile source rows,* BHist, hsame, Cont, *and* NameCert*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}^{\uparrow}}$ *row.*

The carrier contains no host Boolean minimizer, no unbounded codon-table oracle, no genetic-code semantics beyond the displayed reassignment support, and no quotient identifying support lists with motif certificates.

Theorem 3558.2 (ReassignmentSupportCompression NameCert obligations)**.** *Every accepted* ReassignmentSuppo *packet satisfies the following finite obligations:*

1. *carrier admission exposes the support, motif, prime-implicant, minimal-cover, transport, replay, provenance, and NameCert rows of Definition 3558.1;*
2. *support exactness identifies S with the displayed* 13*-codon reassigned support and rejects any added or omitted codon;*
3. *motif expansion identifies M with the five IUPAC motifs whose union is the same support;*
4. *prime-cover compression identifies I with the seven prime implicants and U with the unique five-prime cover;*
5. *nonescape prevents a consumer from replacing the displayed rows by a host minimizer, a raw bitvector equality, a transition-matrix equivalence, a stop-codon suffix rule, or an unbounded codon-table oracle.*

Lean4 ✓ `BEDC.Derived.ReassignmentSupportCompressionUp.ReassignmentSupportCompressionNameCertObligations`

Proof. Project the carrier rows of Definition 3558.1. The support row S is the only entry point for codon membership, and the continuation row C replays it first into the five-motif row M.

Next replay M through the finite prime-implicant enumeration row I. This preserves exactly the displayed maximal exact subcubes because every admitted implicant is checked against the already fixed support row.

Finally project the minimal-cover row U from I. The package, transport, replay, and local naming rows carry no additional semantic oracle, so a consumer cannot bypass the finite support, motif, prime-implicant, and cover rows. The Lean follow-up will provide ‘Nontrivial ReassignmentSupportCompressionUp‘ and ‘FieldFaithful ReassignmentSupportCompressionUp‘ in ‘lean4/BEDC/Derived/ReassignmentSupportCompressionUp/TasteGate.lean‘. □

Theorem 3558.3 (ReassignmentSupportCompression sibling lattice)**.** *Every accepted* ReassignmentSupportCompre *consumer must read the codon six-bit window of chapter 2234, the stop-codon Zeckendorf suffix surface of chapter 1374, the codon transition matrix of chapter 2245, and the bitvector carrier of chapter 346 before treating the five-motif row M and the prime-cover rows I, U as support-compression data.*

Thus the packet is read as a horizontal lattice over displayed codon coordinates, stop-boundary suffixes, transition rows, and finite bitvector surfaces. It cannot be consumed as an isolated Boolean minimization certificate, a raw support bitvector, a stop-codon suffix rule, or a transition-matrix equivalence.

Proof. Project the carrier of Definition 3558.1. The support row S is already tied to displayed codon membership, and the packet's provenance row names the DNA six-bit boundary tile source rows before the motif row is replayed.

Read the sibling surfaces in order. The codon six-bit window supplies the coordinate surface, the stop-codon Zeckendorf suffix supplies the stop-boundary comparison row, the codon transition matrix supplies table-movement visibility, and the bitvector carrier supplies the finite support encoding boundary.

Only after those sibling rows are visible can the consumer interpret M, I, U as support-compression rows. Since the carrier contains no host Boolean minimizer, raw bitvector equality, isolated suffix rule, or transition-matrix quotient, the five-motif and prime-cover rows remain downstream of the sibling lattice. □

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}$.

Constructive story: A $\mathsf{ReassignmentSupportCompression}^{\uparrow}$ packet is generated from the reassigned codon support, five-motif expansion, seven-prime-implicant surface, unique minimum prime cover, componentwise transport, displayed replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3558.1 and Definition 3558.2 seed the finite support-compression route for the reassignment support branch.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentSupportCompression}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a checked Lean certificate, genetic-code semantics outside the displayed support, host Boolean minimization, table-transition completeness, or biological causality.

Upgrade path: Obligation closure requires checked carrier admission, support exactness, motif expansion, prime-implicant enumeration, minimum-cover uniqueness, and nonescape from host minimizers.

Chapter 3559

A Concrete Naming Certificate for $\mathsf{TopologicalClosure}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{TopologicalClosure}^{\uparrow}$ packet records closure of a displayed subset through finite neighbourhood evidence. It sits after the metric closure certificate of chapter 3342, the metric source of chapter 57, the topology surface of chapter 88, the finite ball discipline of $\mathsf{MetricBall}^{\uparrow}$, and the terminal real-name seal of chapter 38. The packet names a topological closure interface for BEDC neighbourhood reads; it is not a host closure operator, not a powerset completion, not quotient equality, not open-cover compactness, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3559.1 (TopologicalClosure carrier)**.** *A* $\mathsf{TopologicalClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, x, U, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row naming the finite neighbourhood request surface.*
2. *S is the displayed subset row whose closure is being tested.*
3. *x is the candidate point row, readable only through the same neighbourhood requests.*
4. *U is the neighbourhood-hit ledger, pairing each displayed neighbourhood of x with a retained source witness from S.*
5. *B is the optional* $\mathsf{MetricBall}^{\uparrow}$ *refinement row when the topology is read through metric balls.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained neighbourhood and witness observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for metric radii and witness distances when a metric refinement is present.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the neighbourhood, ball, window, and readback rows have been displayed.*
9. *H records componentwise* hsame *transport for the topology, subset, candidate, neighbourhood ledger, metric-ball refinement, windows, readback, and Real seal rows.*
10. *C records displayed* Cont *replay from neighbourhood requests to source witnesses, through optional metric balls, finite windows, regular readback, and the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{TopologicalClosure}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$ *row.*

The classifier accepts only packets whose topology row, subset row, candidate point, neighbourhood-hit ledger, optional metric-ball refinement, finite windows, rational readbacks, Real seal, transport, replay, provenance, and local naming are all displayed. It has no coordinate for a host closure operator, arbitrary open-set quantification, powersets, quotient equality, countable choice, or ambient completion of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.TopologicalClosureUp.TopologicalClosureTasteGate single carrier alignment`

Theorem 3559.2 (TopologicalClosure NameCert obligations)**.** *For every accepted* $\mathsf{TopologicalClosure}^{\uparrow}$ *carrier* $K = (T, S, x, U, B, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient topology row, displayed subset row, candidate point row, neighbourhood-hit ledger, optional metric-ball refinement, StreamName windows, RegSeqRat readback, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public closure read must pass through those rows before it can consume the candidate point as closure evidence.*

Proof. Project the accepted carrier to the displayed rows $T, S, x, U, B, W, R, E, H, C, P, N$. The topology, subset, and candidate rows give the only admissible source data for a closure read.

The ledger row U binds each neighbourhood request around x to a retained source witness from S. When the topology is read metrically, the row B refines that same request through finite $\mathsf{MetricBall}^{\uparrow}$ data before the window row W and readback row R are exposed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same displayed packet. Since no row stores a host closure operator, a powerset, quotient equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the NameCert obligations are exhausted by the listed rows. □

Theorem 3559.3 (TopologicalClosure metric neighbourhood route)**.** *Every* $\mathsf{MetricClosure}^{\uparrow}$ *or metric-topology consumer of an accepted* $\mathsf{TopologicalClosure}^{\uparrow}$ *packet reads a candidate closure point through the finite route*

$$T, S, x, U, B, W, R, E, H, C, P, N.$$

The consumer receives a topology row, displayed subset row, candidate point, neighbourhood-hit ledger, metric-ball refinement, finite StreamName windows, RegSeqRat readbacks, RealUp seal, transport, replay, provenance, and local naming. It receives no ambient closure operator, arbitrary open-set family, quotient-set equality, selected convergent net, or RealUp completeness theorem.

Proof. Start with a metric-neighbourhood consumer. The carrier exposes the candidate point x only together with the common topology row T and source subset row S, so the consumer cannot detach the candidate from the displayed neighbourhood interface.

Use the ledger U to align each neighbourhood read with a source witness. The metric-ball row B, finite window row W, regular rational row R, and Real seal E then give the same finite evidence surface used by the $\mathsf{MetricClosure}^{\uparrow}$ packet of chapter 3342.

The rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Thus the handoff is the finite topological closure surface itself, not a stronger host closure, selected net, powerset, quotient, or completeness principle. □

Falsifiable prediction: Every $\mathsf{TopologicalClosure}^{\uparrow}$ public read factors through a Topology row, a displayed subset row, a candidate point row, a neighbourhood-hit ledger, optional MetricBall refinement, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: topology, metric, metric-closure, streamname, regseqrat, real: $\mathsf{TopologicalClosure}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a neighbourhood-hit ledger, optional metric-ball refinement, finite windows, rational readbacks, Real sealing, transport, replay, provenance, and local naming for closure evidence.

Closure status for $\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{TopologicalClosure}^{\uparrow}$ packet is generated from a Topology row, displayed subset row, candidate point, neighbourhood-hit ledger, optional MetricBall refinement, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3559.1, Definition 3559.2, and Definition 3559.3 seed the finite topological closure interface for displayed neighbourhood evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TopologicalClosure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.TopologicalClosureUp.TopologicalClosureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host topological closure operators, arbitrary open-set families, selected convergent nets, quotient equality, $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood-hit stability, metric-ball refinement, StreamName and RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TopologicalClosureUp.

Chapter 3560

A Concrete Naming Certificate for $\mathsf{UniformSpaceCompletion}^{\uparrow}$

Following Bourbaki–Kelley uniform-space completion practice, the $\mathsf{UniformSpaceCompletion}^{\uparrow}$ packet records the finite route by which a uniform entourage source admits Cauchy-filter or Cauchy-net data, passes through dense-source and separated-reflection rows, and reaches a completion seal without turning the seal into quotient classes. It sits after the uniform entourage surface of chapter 339, the Cauchy filter surface of chapter 333, the generic completion packet of chapter 95, the metric comparison route of chapter 3553, the finite stream-window discipline of chapter 126, the regular rational source of chapter 301, and the terminal real-name seal of chapter 38. The packet names the L10 completion handoff as displayed NameCert data; it is not a quotient completion theorem, not a selected-limit operator, not countable choice, not trichotomy, not host equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3560.1 (UniformSpaceCompletion carrier)**.** *A* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, F, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform entourage source row, read from the* $\mathsf{UniformSpace}^{\uparrow}$ *interface before any Cauchy or completion demand is accepted.*
2. *F is the Cauchy filter or Cauchy net row, exposing exactly the finite entourage tests used by the completion consumer.*
3. *D is the dense-source embedding row, recording the source-to-completion handoff that lets original points enter the completion surface.*
4. *S is the separated reflection row, forcing indistinguishable Cauchy data to pass through a displayed separated interface before the completion seal is read.*
5. *E is the completion seal row, recording the accepted completion name after U, F, D, S have all been displayed.*
6. *H records componentwise* hsame *transport for the uniform source, Cauchy row, dense-source row, separated-reflection row, completion seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from Cauchy demands through the uniform source, dense-source handoff, separated reflection, and completion seal.*

8. *P is the* Pkg *provenance over* $\mathsf{UniformSpaceCompletion}^{\uparrow}$*, the seven sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, D, S, E, H, C, P, N$ and their displayed replay paths. It has no coordinate for quotient representatives, selected limits, countable choice, trichotomy, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformSpaceCompletionUp.UniformSpaceCompletionTasteGate single carrier alignment`

Theorem 3560.2 (UniformSpaceCompletion NameCert obligations)**.** *For every accepted* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *carrier $K = (U, F, D, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$ *surface consists exactly of five completion-facing obligations: carrier admission for the displayed finite packet; uniform-Cauchy exactness of the F row against the entourage source U; dense-source handoff through D; separated-reflection non-escape through S before the seal is consumed; and completion-seal stability through E, H, C, P, N. Every completion consumer must read those five rows in that order before it can export a completion name.*

Proof. Project the carrier of Definition 3560.1. The packet first displays the uniform entourage source U and the Cauchy row F, so the Cauchy demand is tested only against named finite entourages rather than against an ambient host uniformity.

The dense-source and separated-reflection rows then control the handoff. The row D records how original source data enters the completion surface, and S records that indistinguishable Cauchy data cannot escape the displayed separated interface before the completion seal is read.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Because the carrier has no row for quotient completion classes, selected limits, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles can be projected as a NameCert obligation. □

Falsifiable prediction: Every accepted $\mathsf{UniformSpaceCompletion}^{\uparrow}$ consumer that reaches a completion seal while omitting the uniform entourage source, Cauchy filter or net row, dense-source embedding row, separated-reflection row, completion-seal row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: uniformspace, cauchyfilter, completion, metric-completion-comparison, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{UniformSpaceCompletion}^{\uparrow}$ because the packet jointly carries a uniform entourage source, Cauchy filter or net exactness, dense-source handoff, separated-reflection non-escape, completion seal stability, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformSpaceCompletion}^{\uparrow}$ packet is generated from a uniform entourage source row, a Cauchy filter or Cauchy net row, a dense-source embedding row, a separated-reflection row, a completion seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3560.1 and Definition 3560.2 seed the finite L10 uniform completion route over uniform source, Cauchy exactness, dense-source handoff, separated reflection, seal stability, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformSpaceCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion classes, selected limits, countable choice, trichotomy, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-Cauchy exactness, dense-source handoff, separated-reflection non-escape, completion-seal stability, transport replay, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.UniformSpaceCompletionUp.

Chapter 3561

A Concrete Naming Certificate for $\mathsf{RegularCauchySubspace}^{\uparrow}$

Following Bishop and Bridges on subspaces of constructive metric spaces, the $\mathsf{RegularCauchySubspace}^{\uparrow}$ packet names regular Cauchy rational data whose accepted reads remain inside a displayed subspace predicate before any quotient equality, selected limit, or ambient completeness service is exposed. It sits on the regular rational readback of chapter 301, the finite-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real seal of chapter 38, and the classifier discipline of chapter 1820. The packet is a finite NameCert surface for subspace-restricted regular Cauchy names; it is not a quotient real, not a selected limit, and not an assertion that the ambient space is complete.

Definition 3561.1 (RegularCauchySubspace carrier)**.** *A* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (Q, S, D, A, K, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row carrying the regular rational readback inspected by the subspace consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule that opens only the windows needed to test the displayed subspace route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bounds, window radii, and comparison rows consumed by Q and S.*
4. *A is the displayed subspace predicate row; it records the finite acceptance evidence that the inspected regular Cauchy name is read inside the named subspace.*
5. *K is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row, used only to compare accepted displayed names through their regular readback, tolerance ledger, and subspace predicate evidence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after Q, S, D, A, K have all been displayed.*
7. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, subspace predicate, classifier, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a subspace-restricted regular Cauchy request through S, D, Q, A, K and then to the Real seal E.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubspace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, S, D, A, K, E, H, C, P, N$. The carrier has no coordinate for quotient real equality, a selected limit, an ambient completeness theorem, a host real equality predicate, countable choice, or an external subspace oracle.

Lean4 ✓ `BEDC.Derived.RegularCauchySubspaceUp.RegularCauchySubspaceTasteGate single carrier alignment`

Theorem 3561.2 (RegularCauchySubspace NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier*

$$U = (Q, S, D, A, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$ *surface consists exactly of carrier admission, regular rational readback, finite StreamName window exposure, DyadicRatCore tolerance exactness, displayed subspace-predicate evidence, CauchyEquivalenceSetoid classifier transport, Real-seal routing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3561.1. The accepted packet exposes the regular readback Q, finite windows S, dyadic ledger D, subspace predicate evidence A, classifier row K, terminal Real seal E, and structural rows H, C, P, N.

The finite observation route is forced before the subspace predicate can be consumed. The $\mathsf{StreamName}^{\uparrow}$ schedule supplies the inspected windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger supplies the tolerance comparisons, and the $\mathsf{RegSeqRat}^{\uparrow}$ row packages those observations as the regular rational name to which A applies.

Classifier transport is then read only through K and H. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row compares accepted displayed names through the same regular readback, tolerance ledger, and subspace predicate evidence; componentwise hsame preserves those coordinates without adding a quotient real equality row.

Finally, the Real seal E is downstream of Q, S, D, A, K, while C, P, N only replay, source, and name the same route. Since the carrier has no coordinate for selected limits, ambient completeness, host real equality, countable choice, or an external subspace oracle, the local NameCert exports none of those objects. □

Theorem 3561.3 (RegularCauchySubspace inherited modulus route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchySubspace}^{\uparrow}$ *carrier inherits its regular Cauchy modulus only through the finite route*

$$S, D, Q, A, K, E, H, C, P, N.$$

Thus the consumer reads StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, displayed subspace evidence, the CauchyEquivalenceSetoid classifier, and the terminal Real seal in that order. It receives no quotient equality for Cauchy names, no selected representative, no ambient completeness service, and no detached subspace-membership oracle.

Proof. Project the accepted carrier $U = (Q, S, D, A, K, E, H, C, P, N)$. The only windows and tolerance data available to a consumer are S and D, and the only regular Cauchy readback attached to them is Q.

The subspace restriction is not a separate oracle. It is the displayed predicate row A applied to the same regular readback route, so every accepted subspace-facing read passes through the same finite windows and dyadic tolerance ledger.

The classifier row K transports accepted names only after Q, S, D, A have been displayed. The terminal seal E is therefore a Real-facing endpoint of this packet, not a selected ambient limit.

The structural rows H, C, P, N preserve and name the route. Because no quotient equality, selected representative, ambient completeness service, or detached membership oracle is present among the displayed coordinates, none can be exported by composing the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3561.1, Definition 3561.2, and Definition 3561.3 seed the finite subspace-restricted regular Cauchy NameCert packet over displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular readback, subspace-predicate stability, classifier transport, Real-seal routing, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchySubspaceUp.

Constructive story: A $\mathsf{RegularCauchySubspace}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, displayed subspace-predicate evidence, a $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3562

A Concrete Naming Certificate for $\mathsf{MetricReflectionSandwich}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{MetricReflectionSandwich}^{\uparrow}$ packet records the standard constructive route that reads a separated metric surface through a metric embedding, lands it in a completion surface, and checks that the dense embedding row preserves the same finite distance evidence. It sits on the metric carrier of chapter 57, the separation witness of chapter 1411, the embedding rows of chapter 3339, the completion surface of chapter 1506, the dense completion discipline of chapter 3351, and the terminal scalar seal of chapter 38. The packet is a finite NameCert consumer for the metric-reflection route; it is not a quotient completion, not a selected-representative theorem, not a choice principle, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 3562.1 (MetricReflectionSandwich carrier)**.** *A* $\mathsf{MetricReflectionSandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, M, Q, E, C, D, R, H, T, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose distances are inspected.*
2. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row, exposing the finite zero-distance separation evidence for the same source.*
3. *Q is the* $\mathsf{MetricEmbedding}^{\uparrow}$ *row, read as a distance-preserving route out of X rather than as a host function equality.*
4. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row naming the completion-facing target surface.*
5. *C is the* $\mathsf{DenseEmbedding}^{\uparrow}$ *row showing that the embedded source rows are dense in the displayed completion target.*
6. *D is the dyadic distance ledger used by the metric, embedding, completion, and density rows.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal reached only after X, M, Q, E, C, D have all been displayed.*
8. *H records componentwise* hsame *transport for metric distances, separation evidence, embedding rows, completion rows, dense embedding rows, dyadic comparisons, Real sealing, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from a source-distance read through the embedding and completion rows to the dense target read.*

10. *P is the* Pkg *provenance over* $\mathsf{MetricReflectionSandwich}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$ *row.*

The classifier accepts only packets whose separated metric, metric embedding, completion, dense embedding, dyadic distance ledger, Real seal, transport, replay, provenance, and local naming rows are displayed together. It has no coordinate for quotient completion, selected representatives, countable choice, host metric equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.MetricReflectionSandwichUp.MetricReflectionSandwichTasteGate single carrier alignment`

Theorem 3562.2 (MetricReflectionSandwich NameCert obligations)**.** *For every accepted* MetricReflectionSandwich *carrier $S = (X, M, Q, E, C, D, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$ *surface consists exactly of carrier admission, separated-metric exposure, metric-embedding distance preservation, completion-row routing, dense-embedding readback, dyadic distance-ledger exactness, Real distance sealing, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 3562.1. The accepted packet exposes the source metric, separation row, embedding row, completion target, dense embedding row, dyadic ledger, Real seal, and structural rows as finite coordinates.

The route through the sandwich is forced by those coordinates. A source-distance read starts at X, is separated by M, passes through the distance-preserving row Q, lands in the completion row E, and is reread through the dense embedding row C.

The remaining rows keep the route finite. The ledger D supplies the distance comparisons, R seals only those displayed comparisons as Real-facing data, and H, T, P, N transport, replay, source, and name the same packet. Since no row stores quotient completion, selected representatives, countable choice, host equality, or ambient Real completeness, the NameCert surface exports none of those objects. □

Theorem 3562.3 (MetricReflectionSandwich completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricReflectionSandwich}^{\uparrow}$ *carrier reads the source separated metric through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{MetricEmbedding}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{DenseEmbedding}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},$$

The consumer receives a displayed metric source, separation evidence, distance-preserving embedding, completion target, dense embedding readback, dyadic distance ledger, Real distance seal, transport, replay, provenance, and local naming. It receives no quotient completion, selected representative, countable-choice schedule, detached host metric equality, or ambient Real-completeness theorem.

Proof. Begin with the carrier and apply Definition 3562.2. The completion-facing reader can inspect only the displayed rows of the packet.

The source metric is not read directly by the completion target. It first passes through the separated metric row and the metric embedding row, so the completion consumer sees the same finite distance data under the embedding route.

The dense embedding row then rereads the embedded source inside the completion surface. Dyadic comparisons and the Real seal are downstream of the displayed metric and embedding

rows, while transport, replay, provenance, and local naming preserve that order. Therefore the completion route is exactly the finite sandwich route, with no quotient completion, selected representative, choice, host equality, or ambient Real completeness exported. □

Falsifiable prediction: An accepted $\mathsf{MetricReflectionSandwich}^{\uparrow}$ completion consumer that reaches $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{DenseEmbedding}^{\uparrow}$ without first reading the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricEmbedding}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows refutes the carrier surface.

Independence witness: metric, separatedmetric, metricembedding, metriccompletion, denseembedding, dyadicratcore, real: $\mathsf{MetricReflectionSandwich}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the separated source metric, embedding, completion target, dense readback, dyadic distance ledger, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3562.1, Definition 3562.2, and Definition 3562.3 seed the finite metric-reflection route between separated metric, metric embedding, completion, dense embedding, dyadic distance, Real seal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricReflectionSandwich}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host metric equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-metric exposure, metric-embedding preservation, completion routing, dense-embedding readback, dyadic distance exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricReflectionSandwichUp.

Constructive story: A $\mathsf{MetricReflectionSandwich}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ source row, a $\mathsf{SeparatedMetric}^{\uparrow}$ row, a $\mathsf{MetricEmbedding}^{\uparrow}$ route, a $\mathsf{MetricCompletion}^{\uparrow}$ target, a $\mathsf{DenseEmbedding}^{\uparrow}$ readback row, a $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3563

A Concrete Naming Certificate for $\mathsf{ConvexConjugate}^{\uparrow}$

Following Rockafellar and Boyd–Vandenberghe, the $\mathsf{ConvexConjugate}^{\uparrow}$ packet names the Fenchel–Legendre conjugate surface by finite primal rows, dual rows, pairing evaluations, epigraph rows, and displayed supremum budgets. It sits over the convex-set, vector-space, real-name, and real-order interfaces of chapter 208, chapter 47, chapter 38, and section 38.3, and it supplies the conjugacy handoff consumed by chapter 321. The packet records conjugate data as BEDC carrier rows; it does not import host convex analysis, selected suprema, quotient function spaces, choice selectors, or ambient completeness.

Definition 3563.1 (Convex conjugate carrier)**.** *A* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, V, P, D, E, S, H, C, Q, N)$$

with the following displayed rows:

1. *F is the primal convex functional row, read through a displayed* $\mathsf{ConvexSet}^{\uparrow}$ *epigraph boundary.*
2. *V is the dual-vector row over the visible* $\mathsf{VecSpace}^{\uparrow}$ *dependency surface.*
3. *P is the pairing-evaluation row that records the finite bilinear read used by the conjugate comparison.*
4. *D is the domain and epigraph admission row tying the primal functional to the visible convex-set carrier.*
5. *E is the extended-value comparison row, read through* $\mathsf{Real}^{\uparrow}$ *and the real-order interface.*
6. *S is the supremum-budget row listing the finite inspected witnesses and the budget that prevents escape to a host supremum.*
7. *H stores componentwise* hsame *transport for the primal row, dual vector, pairing row, epigraph row, comparison row, supremum budget, replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from primal and dual reads to pairing evaluations, from pairing evaluations to epigraph comparisons, and from those comparisons to the budgeted conjugate row.*
9. *Q is the* Pkg *provenance over* $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, P, D, E, S, H, C, Q, N$. It has no coordinate for a host supremum operator, quotient function space, choice-selected maximizer, ambient dual-space completion, or external conjugacy theorem.

Lean4 ✓ `BEDC.Derived.ConvexConjugateUp.ConvexConjugateTasteGate single carrier alignment`

Theorem 3563.2 (Convex conjugate NameCert obligations)**.** *For every accepted* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier* $K = (F, V, P, D, E, S, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, classifier stability under componentwise* hsame *transport, epigraph transport through the visible* $\mathsf{ConvexSet}^{\uparrow}$ *dependency, supremum-budget nonescape through* S*, and ledger visibility through the replay and provenance rows* C, Q, N*. Every conjugate read must pass through the primal functional row, dual-vector row, pairing evaluation row, epigraph row, extended-value comparison row, and displayed budget before a Fenchel-facing consumer can inspect it.*

Proof. Project the carrier of Definition 3563.1. Its visible coordinates are exactly $F, V, P, D, E, S, H, C, Q, N$, so carrier admission and classifier comparison are componentwise on those rows.

The epigraph and comparison clauses route every primal functional read through $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the real-order interface before it can be paired with the dual row. Transport by H and replay by C preserve that displayed route.

The row S is the only supremum-facing coordinate. Since the carrier has no host supremum, quotient function space, or selected maximizer field, the budgeted ledger S, C, Q, N is the entire local NameCert surface available to downstream consumers. □

Theorem 3563.3 (Convex conjugate Fenchel handoff)**.** *Every* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer that reads conjugacy through an accepted* $\mathsf{ConvexConjugate}^{\uparrow}$ *carrier receives only the displayed packet route*

$$F, V, P, D, E, S \longmapsto H, C, Q, N.$$

The consumer reads primal functionals, dual vectors, pairings, epigraphs, extended-value comparisons, and supremum budgets through the $\mathsf{ConvexConjugate}^{\uparrow}$ *packet, and receives no host supremum, quotient function space, choice selector, ambient completeness theorem, or external Fenchel conjugacy principle.*

Proof. Use Definition 3563.2. It places every conjugate-facing read on the carrier rows F, V, P, D, E, S before the structural rows H, C, Q, N can replay or source the answer.

Now pass to the Fenchel-facing consumer. The consumer may cite the $\mathsf{FenchelDuality}^{\uparrow}$ packet only after the displayed conjugate rows have been read, so conjugacy enters as a finite BEDC row rather than as an analytic operation in the host language.

The excluded host objects are absent from Definition 3563.1. Therefore the handoff supplies exactly the packet route and no selected supremum, quotient-function value, choice witness, completeness theorem, or external duality theorem. □

Falsifiable prediction: Every $\mathsf{ConvexConjugate}^{\uparrow}$ consumer reads conjugacy through a primal functional row, dual-vector row, pairing-evaluation row, epigraph row, extended-value comparison row, supremum-budget row, transport, continuation replay, provenance, and local naming.

Independence witness: convexset, vecspace, real, realalgorder, fenchelduality: $\mathsf{ConvexConjugate}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries primal functional data, dual-vector data, pairing evaluation, epigraph comparison, a displayed supremum budget, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}$.

Constructive story: A $\mathsf{ConvexConjugate}^{\uparrow}$ packet is generated from a primal convex functional row, a dual-vector row, a pairing-evaluation row, epigraph and extended-value comparison rows, a displayed supremum budget, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3563.1, Definition 3563.2, and Definition 3563.3 bind the finite Fenchel–Legendre conjugate certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConvexConjugate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConvexConjugateUp.ConvexConjugateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host convex analysis, selected suprema, quotient function spaces, choice selectors, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, supremum-budget nonescape, Fenchel handoff, and TasteGate field faithfulness for BEDC.Derived.ConvexConjugateUp.

$\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ is routed through chapter 38, chapter 301, and the apartness sibling surface before the L10 completion-facing dependency rows are consumed.

Chapter 3564

A Concrete Naming Certificate for LocatedApartnessRealBridge$^{\uparrow}$

Following Bishop and Bridges, the LocatedApartnessRealBridge$^{\uparrow}$ packet records the finite comparison route by which located-radius readback and positive apartness evidence can be consumed together without promoting the result to classical real order. It sits between the located-real surface of **??**, the Cauchy real apartness surface of chapter 1881, the real apartness order bridge of chapter 1905, and the cotransitivity route of chapter 1788. The packet is a constructive comparison bridge for Real and Completion consumers; it is not a trichotomy theorem, quotient-real equality principle, host real equality test, selected-limit operator, or completeness claim.

Definition 3564.1 (Located apartness real bridge carrier)**.** *A* LocatedApartnessRealBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$$

with the following displayed rows:

1. *L_0 and L_1 are accepted* LocatedReal$^{\uparrow}$ *endpoint packets whose finite located-radius windows are being compared.*
2. *R_0 and R_1 are the located-radius readback rows, each exposing a* StreamName$^{\uparrow}$ *window, a* RegSeqRat$^{\uparrow}$ *readback, and a* DyadicRatCore$^{\uparrow}$ *radius ledger before any terminal* Real$^{\uparrow}$ *seal is inspected.*
3. *A is the positive apartness witness row, read through the* CauchyRealApartness$^{\uparrow}$ *surface and tied to the same endpoint windows.*
4. *O is the order-facing bridge row supplied by* RealApartnessOrderBridge$^{\uparrow}$*, used only after A has exposed positive apartness.*
5. *T is the cotransitivity route row, pointing to the existing* ApartnessCotransitivity$^{\uparrow}$ *surface when a third displayed real name is inspected.*
6. *H records componentwise* hsame *transport for the endpoint packets, readbacks, apartness witness, order bridge, cotransitivity route, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located-apartness bridge request through $L_0, L_1, R_0, R_1, A, O, T, H, P, N$.*
8. *P is the* Pkg *provenance over* LocatedApartnessRealBridge$^{\uparrow}$, LocatedReal$^{\uparrow}$, CauchyRealApartness$^{\uparrow}$, RealApartnessOrderBridge$^{\uparrow}$, ApartnessCotransitivity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$ *row.*
The classifier admits exactly the finite packet in which located-radius readback and apartness evidence share the displayed endpoint windows. It has no coordinate for real trichotomy, quotient equality, host real equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Theorem 3564.2 (Located real apartness cotransitive bridge)**.** *For every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N),$$

any comparison read that uses the located-radius readbacks R_0, R_1 together with the positive apartness witness A reaches a Real-facing consumer only through the existing apartness-cotransitivity surface. More precisely, the read first opens the two $\mathsf{LocatedReal}^{\uparrow}$ *endpoint packets, then the shared finite readback route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

then the $\mathsf{CauchyRealApartness}^{\uparrow}$ *witness, then the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *order-facing handoff, and finally the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row T when a third real name is present. The export is finite constructive comparison evidence only. It exports no trichotomy, quotient real equality, host real equality, selected limit, countable-choice branch selection, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or completion principle.*

Proof. Project the carrier of Definition 3564.1. The endpoint information is exactly the two located real packets L_0, L_1 and the two finite located-radius readbacks R_0, R_1.

The apartness row is opened only after those finite readbacks are visible. Thus A is tied to the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ route as the endpoint data.

The order-facing handoff is downstream of apartness. The row O may expose the order bridge supplied by $\mathsf{RealApartnessOrderBridge}^{\uparrow}$, but it receives positive separation from A and cannot introduce an equality test or a trichotomy branch.

When a third real name is inspected, the route must pass through T. That row is precisely the cotransitivity surface of $\mathsf{ApartnessCotransitivity}^{\uparrow}$, so the consumer receives a displayed cotransitive apartness route rather than a classical comparison principle.

The remaining rows H, C, P, N transport, replay, package, and name the same finite route. Since the carrier lists no quotient equality, host equality, selected limit, choice, or completeness row, none can be exported by the local certificate. □

Theorem 3564.3 (Located apartness real bridge NameCert obligations)**.** *For every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *endpoint admission for the two* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint packets L_0, L_1;*
2. *located-radius readback for R_0, R_1, each displaying the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *route attached to its endpoint;*
3. *positive apartness witness admission for the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A;*

4. *order-facing handoff for the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *row* O*, read only after* A *has exposed positive apartness;*
5. *cotransitivity routing for the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row* T *when a third displayed real name is inspected;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* H, C, P, N*.*

These rows export only the displayed constructive comparison route. They do not assert real trichotomy, quotient real equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Proof. Project the carrier of Definition 3564.1. The endpoint and readback obligations are exactly the rows L_0, L_1, R_0, R_1, so the local certificate admits only displayed located endpoints and finite radius-readback data.

The apartness and order rows are A and O. The carrier orders them so that the $\mathsf{CauchyRealApartness}^{\uparrow}$ witness is exposed before the $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ handoff can be consumed.

Apply Definition 3564.2. It reads the same endpoint, readback, apartness, and order rows as the consumer-facing route, and it sends any third-name comparison through the cotransitivity row T.

The remaining rows H, C, P, N transport, replay, package, and name that route. Since the carrier has no coordinate for trichotomy, quotient equality, host equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 3564.4 (Located apartness real two-way separation)**.** *An accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier exposes* $\mathsf{Real}^{\uparrow}$ *separation reads only through the displayed located-apartness row* A *and its endpoint readbacks* L_0, L_1, R_0, R_1*. Conversely, it exposes located-apartness reads only through the terminal* $\mathsf{Real}^{\uparrow}$ *seal route carried by* R_0, R_1 *together with the bridge transport, replay, provenance, and naming rows* H, C, P, N*.*

Proof. Project the carrier $J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$ from Definition 3564.1. The Real-facing data are the two located endpoints and their radius-readback rows, while the positive separation data are confined to the apartness witness row A.

Apply the NameCert obligations of Definition 3564.3. They require endpoint admission, located-radius readback, positive apartness admission, order-facing handoff, cotransitivity routing, and structural replay to be read as the same finite packet.

In the Real-facing direction, a separation read reaches the endpoint seal only after the displayed apartness row has tied the endpoints to positive evidence. In the apartness-facing direction, a consumer of A receives only the endpoint $\mathsf{Real}^{\uparrow}$ seal route plus H, C, P, N; the carrier exports no trichotomy, quotient equality, host equality, selected limit, countable-choice branch, or ambient completeness coordinate. □

Theorem 3564.5 (Located apartness real nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *read that exports an order-facing handoff still factors through the positive apartness row* A *and the cotransitivity route* T*. Consequently the handoff cannot export a three-way comparison branch, quotient-real equality, host real equality, or a detached order comparison outside*

$$L_0, L_1, R_0, R_1, A, O, T, H, C, P, N.$$

Proof. Project the carrier $J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$ from Definition 3564.1. The order-facing row is O, and it is listed downstream of the positive apartness witness A rather than as an independent trichotomy or equality coordinate.

Apply Definition 3564.2. It forces any third-name comparison to pass through the cotransitivity row T after the endpoint readbacks and the positive apartness row have been displayed.

Apply Definition 3564.3 and Definition 3564.4. These obligations keep O downstream of A and keep all Real-facing separation reads attached to the endpoint readbacks R_0, R_1.

The remaining rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate records a three-way comparison branch, quotient-real equality, host real equality, or detached order comparison, none can be exported by an accepted read. □

Theorem 3564.6 (Located apartness real bridge completion forward route)**.** *Any L10 completion consumer, in particular the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *consumer of chapter 1639, that reads located apartness from an accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *carrier*

$$J = (L_0, L_1, R_0, R_1, A, O, T, H, C, P, N)$$

must pass through the located-radius readbacks R_0, R_1*, the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *surface* A*, the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *compatibility row* O*, and the* $\mathsf{ApartnessCotransitivity}^{\uparrow}$ *row* T *before it consumes any completion-facing comparison. The route exports constructive comparison evidence only; it exports no trichotomy, quotient-real equality, selected limit, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or completion principle outside the displayed rows.*

Proof. Project the endpoint and readback rows from Definition 3564.1. The completion-facing consumer first sees the two located endpoints and their finite radius readbacks, so the comparison begins with visible $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ data.

Apply Definition 3564.2. It opens the Cauchy real apartness witness after the readbacks, then routes the positive evidence through the order-facing bridge and the cotransitivity surface whenever a third name is inspected.

Use Definition 3564.3 and Definition 3564.4. These keep located-apartness evidence and Real-facing separation tied to the same finite packet before any completion-facing comparison is consumed.

Since the carrier has no coordinate for trichotomy, quotient equality, selected limits, host equality, choice, completeness, or an independent completion principle, the forward route is exactly the displayed located-radius, apartness, order-bridge, and cotransitivity handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3564.1 and Definition 3564.2 seed the finite constructive comparison route over located-real endpoints, located-radius readbacks, Cauchy apartness, the real-apartness order bridge, and apartness cotransitivity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedApartnessRealBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close real trichotomy, quotient real equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-radius readback stability, Cauchy apartness handoff, order-bridge compatibility, cotransitivity routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedApartnessRealBridgeUp.

Constructive story: A $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ packet is generated from two located-real endpoints, two located-radius readbacks, a positive Cauchy real apartness witness, a real-apartness order bridge row, an apartness-cotransitivity route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Lemma 3564.7 (Located apartness real bridge sibling dependency route)**.** *Every accepted* LocatedApartnessRealBri *read consumes the located endpoint rows of* **??***, the positive Cauchy apartness evidence of chapter 1881, the order-facing handoff rows of chapter 1905, and the cotransitivity routing of chapter 1788 before the local bridge name is inspected.*

Proof. Project the carrier of Definition 3564.1. The endpoint rows L_0, L_1 are located-real packets, so their finite located-radius windows are read through the sibling $\mathsf{LocatedReal}^{\uparrow}$ surface.

The positive separation row A is then read through $\mathsf{CauchyRealApartness}^{\uparrow}$, and the order-facing row O is downstream of that apartness evidence. If a third displayed real name is present, the row T routes the comparison through $\mathsf{ApartnessCotransitivity}^{\uparrow}$.

The structural rows H, C, P, N transport, replay, package, and name this same route. Since the carrier has no additional dependency coordinate, every local bridge read factors through exactly the listed sibling chapters. □

Lemma 3564.8 (Located apartness real bridge forward lattice)**.** *Every downstream Real and Completion consumer of a* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *packet reads the local bridge only after the located-real, Cauchy-apartness, order-bridge, and cotransitivity sibling surfaces of Definition 3564.7 have been exposed. In particular, a handoff to the separated-completion route of chapter 1612 may consume the bridge as finite apartness-routing data, but not as quotient-real equality, trichotomy, countable choice, or a selected completion limit.*

Proof. Project the dependency route already recorded in Definition 3564.7. The local bridge has no independent downstream coordinate before the located endpoint rows, positive Cauchy apartness evidence, order-facing handoff rows, and cotransitivity routing have been read.

A separated-completion consumer may therefore receive only the finite bridge packet produced by those sibling surfaces. The consumer obtains a displayed Real and Completion handoff, while the excluded host principles remain outside the carrier. □

Lemma 3564.9 (Located apartness real bridge Cauchy setoid route)**.** *Every located-apartness read exported by an accepted* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *packet to a completion or separated-setoid consumer must pass through* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier route before it can be used as apartness evidence. The exported evidence is the finite L10 readback chain through chapter 38, chapter 301, chapter 126, chapter 291, and chapter 1820; it is not quotient-real equality, a selected completion representative, host order data, or ambient completeness.*

Proof. Project the carrier of Definition 3564.1. The located-radius readbacks are R_0, R_1, and each exposes the stream-name window, regular-rational readback, dyadic-radius ledger, and terminal $\mathsf{Real}^{\uparrow}$ seal before any apartness row is consumed.

Use Definition 3564.3. The positive apartness row A is tied to those same endpoint windows, so a completion or separated-setoid consumer cannot detach the apartness witness from the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Now route the completion-facing read through the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ classifier. The setoid classifier is the sibling surface that records equality-compatible Cauchy comparison before a completion consumer may reuse the apartness evidence.

The structural rows H, C, P, N only transport, replay, source, and name this route. Since no row stores quotient-real equality, selected completion representatives, host order data, or ambient completeness, the L10 chain is nonescaping. □

Chapter 3565

A Concrete Naming Certificate for $\mathsf{LocatedApartnessCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedApartnessCompletion}^{\uparrow}$ packet records a completion-facing carrier in which located real windows and apartness evidence are displayed before any completion read is allowed to leave the finite BEDC ledger. It sits after the generic apartness-completion boundary of chapter 1787, the located-apartness real bridge of chapter 3564, the located real carrier of **??**, the positive-separation surface of chapter 304, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The carrier names a finite located-apartness completion route only; it does not assert Real completeness, trichotomy, quotient equality, countable choice, open-cover compactness, or host-selected limits.

Definition 3565.1 (Located apartness completion carrier)**.** *A* $\mathsf{LocatedApartnessCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, B, U, V, G, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ApartnessCompletion}^{\uparrow}$ *boundary row exposing the separated completion route.*
2. *B is the* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *row binding the completion boundary to located real apartness data.*
3. *U is the source* $\mathsf{LocatedReal}^{\uparrow}$ *row whose finite windows are inspected by the completion read.*
4. *V is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-separation row used to keep equality information apartness-facing.*
5. *G is the located gap row recording that the inspected endpoint windows have a displayed positive dyadic separation whenever apartness is consumed.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the finite completion and apartness windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for positive gaps, radii, and comparison cells.*
9. *E is the optional* $\mathsf{Real}^{\uparrow}$ *seal row, present only after the located, apartness, stream, regular-rational, and dyadic rows have been displayed.*

10. *H records componentwise* hsame *transport for all source, bridge, window, readback, tolerance, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay of the same finite route.*
12. *P is the* Pkg *provenance over* LocatedApartnessCompletion$^{\uparrow}$, ApartnessCompletion$^{\uparrow}$, LocatedApartnessRealBridge$^{\uparrow}$, LocatedReal$^{\uparrow}$, ApartnessReal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{LocatedApartnessCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, U, V, G, S, R, D, E, H, C, P, N$. It has no coordinate for Real completeness, trichotomy, quotient equality, countable choice, open-cover compactness, or host-selected limits.

Lean4 ✓ `BEDC.Derived.LocatedApartnessCompletionUp.LocatedApartnessCompletionTasteGate single carrier alignment`

Theorem 3565.2 (Located apartness completion NameCert obligations)**.** *For every accepted* LocatedApartnessCompletion$^{\uparrow}$ *carrier $L = (A, B, U, V, G, S, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{LocatedApartnessCor}}$ *surface consists exactly of apartness-completion boundary exposure through A; located-apartness real bridge exposure through B; located real source exposure through U; positive-separation exposure through V; located gap exposure through G; finite window, regular readback, and dyadic tolerance exposure through S, R, D; optional Real sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The displayed tuple contains precisely the apartness-completion boundary, located-apartness bridge, located real source, positive-separation row, located gap row, finite windows, regular readback, dyadic tolerance ledger, optional seal, transport, replay, provenance, and local name.

The NameCert obligations are read in the same order. Rows A and B expose the completion boundary and bridge before U, V, G make the located real and apartness evidence available. Rows S, R, D then bind every consumer observation to finite windows, regular rational readback, and dyadic tolerance data before the optional Real seal E can be inspected.

The structural rows H, C, P, N transport, replay, source, and name only those displayed rows. Since the carrier has no coordinate for Real completeness, trichotomy, quotient equality, countable choice, open-cover compactness, or host-selected limits, the listed rows exhaust the local obligation surface. □

Theorem 3565.3 (Located apartness completion apartness route)**.** *Every apartness-facing completion read of an accepted* LocatedApartnessCompletion$^{\uparrow}$ *carrier factors through the route*

$$A, B, U, V, G, S, R, D \longmapsto H, C, P, N.$$

Thus positive separation is consumed only as displayed ApartnessReal$^{\uparrow}$ *evidence attached to located* LocatedReal$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, and* DyadicRatCore$^{\uparrow}$ *tolerance rows, not as a host trichotomy or a quotient equality test.*

Proof. Read the apartness source of the completion route first. The generic boundary row A and the bridge row B identify the completion-facing apartness route before any located real endpoint is inspected.

Rows U, V, G then display the located real window, the positive-separation evidence, and the dyadic gap row. Rows S, R, D keep the same observation finite by naming the stream schedule, regular rational readback, and dyadic tolerance ledger.

The transport, replay, provenance, and naming rows H, C, P, N preserve that route without adding a host comparison coordinate. Hence the consumer receives apartness only through the displayed rows and cannot read trichotomy or quotient equality from the packet. □

Theorem 3565.4 (Located apartness completion nonescape)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{LocatedApartnessCompletion}^{\uparrow}$ *carrier must pass through* A, B, U, V, G, S, R, D *before it may inspect the optional seal row* E *or the structural rows* H, C, P, N*. Consequently the packet exports no Real completeness principle, host-selected limit, open-cover compactness argument, countable-choice schedule, trichotomy decision, or quotient equality of completed names.*

Proof. Start from any downstream consumer. Carrier projection gives only the ordered rows $A, B, U, V, G, S, R, D, E, H, C, P, N$.

The optional Real seal is downstream of the located-apartness data. A read must first display the apartness-completion boundary, located-apartness bridge, located real source, apartness evidence, located gap, stream window, regular rational readback, and dyadic tolerance ledger.

After that finite route is displayed, E, H, C, P, N can seal, transport, replay, package, and name the same data. None of these rows stores completeness, selected limits, open-cover compactness, countable choice, trichotomy, or quotient equality, so those objects cannot escape through the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedApartnessCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedApartnessCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3565.1, Definition 3565.2, Definition 3565.3, and Definition 3565.4 seed the finite located-apartness completion route over ApartnessCompletion, LocatedApartnessRealBridge, LocatedReal, ApartnessReal, StreamName, RegSeqRat, DyadicRatCore, optional Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedApartnessCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, trichotomy, quotient equality, countable choice, open-cover compactness, host-selected limits, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, bridge exposure, located real and apartness route stability, finite readback discipline, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedApartnessCompletion

Constructive story: A $\mathsf{LocatedApartnessCompletion}^{\uparrow}$ packet is generated from an ApartnessCompletion boundary row, a LocatedApartnessRealBridge row, a LocatedReal source row, an ApartnessReal positive-separation row, a located gap row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, an optional RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ hub reads its bounded candidate row through chapter 719 and the MetaCIC normalization frontier, so the socket is a visible sibling dependency rather than an isolated kernel float.

Chapter 3566

A Concrete Naming Certificate for NormalizationCandidateSocket$^{\uparrow}$

The NormalizationCandidateSocket$^{\uparrow}$ packet realizes the candidate-mediated normalization evidence of chapter 9383 as a finite socket object. It sits beside the critical-path packet of chapter 765, the normalization frontier of chapter 892, the confluence audit packet of chapter 735, and the normal-form consistency seal of chapter 740. The packet separates candidate rows from closed-candidate rows and candidate-set rows, records the Pi and App adequacy surfaces, and keeps no-infinite-reduction exports downstream of supplied candidate evidence. It is not an unconditional closed-term strong-normalization theorem.

The socket consumes the MetaCIC candidate-normalization handoff through the candidate-set surface of chapter 2320 and the closed candidate route of chapter 5351. Its subject-reduction boundary is the four-row discharge socket of chapter 719; the local packet may hand candidate-mediated normalization data toward that boundary, but it does not replace the beta, application, lambda, or Pi preservation rows.

Definition 3566.1 (Normalization candidate socket carrier)**.** *A* NormalizationCandidateSocket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the closed typed MetaCIC term window, exposing the empty-context term, type, and closedness checks available to the audit map.*
2. *K is the reducibility candidate row, carrying the candidate predicate and its strong-normalization consequence.*
3. *D is the closed-candidate row, carrying closedness together with candidate evidence.*
4. *E is the candidate-set membership row, carrying membership, soundness, and beta-closure data.*
5. *A is the application adequacy row, recording the function-side Pi evidence, argument candidate evidence, and application candidate handoff.*
6. *B is the Pi adequacy row, recording Pi-candidate and neutral-normal evidence consumed by typed application routes.*
7. *I is the no-infinite-reduction row, downstream of the candidate-mediated strong-normalization projection.*

8. *F is the frontier row, retaining the refusal of any read that asks for closed-term strong normalization after deleting K, D, or E.*
9. *H records componentwise* hsame *transport for the term window, candidate row, closed-candidate row, candidate-set row, App adequacy row, Pi adequacy row, no-infinite-reduction row, frontier row, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a normalization-candidate socket request through $M, K, D, E, A, B, I, F, H,$*
11. *P is the* Pkg *provenance over* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$*, chapter 9383, chapter 765, chapter 892, chapter 735, chapter 740,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}$ *row.*

The classifier compares accepted sockets only through $M, K, D, E, A, B, I, F, H, C, P, N$. It has no coordinate for global closed-term strong normalization, evidence-erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Theorem 3566.2 (Normalization candidate socket boundary)**.** *For every accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N),$$

the local socket exports closed typed strong-normalization and no-infinite-reduction evidence only through at least one of the candidate rows K, D, E. Application adequacy A and Pi adequacy B are admissible only as route data for that same candidate-mediated socket. Any consumer missing the candidate row, the closed-candidate row, or the candidate-set row reaches F and cannot export closed-term strong normalization from this packet.

Proof. Read the carrier of Definition 3566.1. The closed term window M is present, but the socket does not attach strong normalization directly to M.

The candidate rows are the gate. A read may use K as reducibility candidate evidence, D as closed-candidate evidence, or E as candidate-set membership evidence. The strong-normalization export is downstream of one of these rows.

The adequacy rows refine the same gate. The application row A and Pi row B supply route data for function, argument, Pi, and neutral cases, but neither row replaces candidate evidence.

The no-infinite-reduction row I is downstream of the candidate-mediated strong-normalization projection. If a consumer omits K, D, E or asks to erase them, the frontier row F records the refusal.

Transport, replay, provenance, and local naming preserve this row order through H, C, P, N. Because no global normalization, unrestricted subject-reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed consistency row occurs in the carrier, the socket cannot export one. □

Theorem 3566.3 (Normalization candidate socket candidate-erasure refusal)**.** *Let $S = (M, K, D, E, A, B, I, F, H,$ be an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier. Any consumer that keeps the closed typed term window M but deletes all candidate evidence rows K, D, E cannot export the no-infinite-reduction row I from this packet. Such a consumer must reach the frontier refusal row F. The App adequacy row A and Pi adequacy row B remain route data for candidate-mediated reads; they are not substitutes for reducibility candidate, closed-candidate, or candidate-set evidence.*

Proof. Project the carrier of Definition 3566.1. The closed typed term window M is a visible socket input, but the carrier does not attach no-infinite-reduction evidence directly to M.

Apply Definition 3566.2. That boundary places closed typed strong-normalization and no-infinite-reduction exports downstream of at least one of K, D, E.

Now delete all three candidate evidence rows. With K, D, E absent, the consumer has no displayed reducibility candidate, closed-candidate, or candidate-set membership route from which I can be replayed.

The adequacy rows do not close the gap. App adequacy A and Pi adequacy B organize function, argument, Pi, and neutral routes once candidate evidence is present, but neither row supplies candidate evidence on its own. Therefore the only available response is the frontier row F, and the packet refuses the evidence-erased no-infinite-reduction export. □

Theorem 3566.4 (Normalization candidate socket MetaCIC diamond route)**.** *Every diamond-residual consumer of an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

must first read the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, and the frontier row F before citing chapter 4061. The MetaCIC diamond residual budget may consume the socket only after those candidate-mediated rows are visible; it cannot read a closed typed term window M as a premise-free normalization endpoint.

Proof. Project the socket carrier of Definition 3566.1. The rows relevant to a diamond-residual consumer are displayed as K, D, E, A, B, I, F inside the same finite packet.

Use Definition 3566.2. It places closed typed strong-normalization and no-infinite-reduction exports downstream of at least one candidate evidence row among K, D, E.

Use Definition 3566.3. It shows that deleting K, D, E while retaining M forces the frontier refusal row F, so the diamond-residual consumer cannot treat the closed term window alone as normalization evidence.

The adequacy rows A and B remain route data for candidate-mediated App and Pi reads, and I remains downstream of the same candidate evidence. Therefore the sibling route to chapter 4061 is available only after K, D, E, A, B, I, F are visible in the socket. □

Theorem 3566.5 (Normalization candidate socket forward dependency)**.** *Every forward discharge consumer of an accepted* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *carrier*

$$S = (M, K, D, E, A, B, I, F, H, C, P, N)$$

must expose the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, the frontier row F, and then the forward-socket ledger of chapter 3387 before exporting a downstream socket read. The forward ledger is therefore downstream of visible candidate evidence, not a replacement for reducibility, closed-candidate, candidate-set, App adequacy, Pi adequacy, or frontier rows.

Proof. Project the carrier of Definition 3566.1. The forward discharge consumer can inspect only the displayed rows $M, K, D, E, A, B, I, F, H, C, P, N$.

Use Definition 3566.2. It places any strong-normalization or no-infinite-reduction export downstream of the candidate rows K, D, E and keeps App and Pi adequacy in the rows A, B.

Use Definition 3566.3. If the consumer deletes K, D, E while keeping the closed term window M, the route reaches the frontier refusal F and cannot export I.

Only after K, D, E, A, B, I, F are visible may the consumer cite chapter 3387. The forward-socket ledger records the downstream socket read, while H, C, P, N transport, replay, source, and name the same candidate-mediated packet. □

The socket is a consumer-side sibling of chapter 2320, chapter 5351, and chapter 719. It may route candidate-mediated normalization toward residual or discharge consumers only while keeping the candidate rows K, D, E, the adequacy rows A, B, the no-infinite-reduction row I, and the frontier row F visible.

Falsifiable prediction: Any consumer missing candidate, closed-candidate, or candidate-set rows in a $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet cannot export closed-term strong normalization from that packet.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacicconfluenceauditpacket, normalformconsistencyseal: $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, frontier, transport, replay, provenance, and local naming rows.

Lean TasteGate coverage for this packet must provide Nontrivial and FieldFaithful instances for the candidate row family, the closed-candidate row family, the candidate-set row family, and the socket classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3566.1 and Definition 3566.2 seed the finite candidate-mediated normalization socket over candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, and frontier rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationCandidateSocket}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global closed-term strong normalization, evidence-erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate gate exactness, App and Pi adequacy compatibility, no-infinite-reduction routing, frontier refusal, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.NormalizationCandidateSocketUp.

Constructive story: A $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet is generated from one closed typed MetaCIC term window, candidate evidence, closed-candidate evidence, candidate-set evidence, App adequacy, Pi adequacy, no-infinite-reduction, frontier refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 3566.6 (Normalization candidate socket fuel boundary)**.** *Every* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *consumer that asks for bounded normalization receives only the displayed finite socket rows: the candidate term window M, a fuel row U, a bounded trace row T, a closedness-preservation row Q, and the structural rows H, C, P, N that transport and name that finite route. The trace row records that the fuel-indexed computation reaches the displayed bounded result within U steps and that each recorded step preserves the closedness row Q. The socket does not export a strong-normalization certificate, a no-infinite-reduction theorem, an erased candidate projection, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency from this bounded trace.*

Proof. Project the socket carrier of Definition 3566.1. The closed typed term window is available as M, while candidate, closed-candidate, and candidate-set rows remain the only rows that can support any strong-normalization export.

For bounded normalization, insert the fuel row U and bounded trace row T between the term window and the replay rows. The trace is finite because each step is indexed by the displayed fuel ledger rather than by an unbounded reduction argument.

Closedness preservation is carried separately by Q. Each trace edge may be read only together with the source and target closedness rows named by Q, so the consumer receives a closed finite computation rather than a global normalization theorem.

Finally, H, C, P, N transport, replay, package, and name exactly these displayed rows. Since no strong-normalization certificate is a coordinate of the bounded route, the socket boundary refuses any consumer that tries to read strong normalization from fuel alone. □

The bounded trace row U, T, Q is the local socket-side consumer of the MetaCIC normalization fuel boundary in chapter 5936. The socket fuel boundary reuses the MetaCIC fuel ledger as bounded trace evidence while refusing to turn fuel alone into strong normalization or no-infinite-reduction.

Theorem 3566.7 (Normalization candidate socket fuel-erasure refusal)**.** *Every* NormalizationCandidateSocket$^{\uparrow}$ *no-infinite-reduction read or closed strong-normalization read that crosses the socket fuel boundary must preserve the candidate rows K, D, E and the candidate-set fuel route before reading the no-infinite-reduction row I or frontier refusal row F. Erasing K, D, E from such a read turns the socket into an unconditional normalization endpoint over the closed typed term window M and fuel row U, but the carrier has no coordinate for that endpoint. The erased route is therefore refused at F rather than exported as evidence-free normalization.*

Proof. Project the carrier rows and use the fuel boundary of Definition 3566.6. The fuel row U and trace row T expose only bounded computation data, while the candidate rows K, D, E remain the visible evidence rows for strong-normalization and no-infinite-reduction reads.

Now apply the candidate-set fuel route of Definition 3566.8. A read that reaches candidate-set evidence must pass through E, the App and Pi adequacy rows, the bounded trace, the closedness-preservation row, and the structural rows before any public normalization-facing consequence is named.

Use the fuel-boundary candidate gate, Definition 3566.9. If K, D, E are erased, the fuel boundary cannot be lowered into a strong-normalization or no-infinite-reduction export; the only admissible output is the frontier refusal row F.

Finally specialize the candidate-set readback theorem, Definition 3566.10. It places the no-infinite-reduction row I downstream of M, E, A, B and the structural route. Thus reading I or F after deleting K, D, E would assert an endpoint that the carrier does not store, so the socket refuses the erased route. □

Theorem 3566.8 (Normalization candidate socket candidate-set fuel route)**.** *If an accepted* NormalizationCandidate *packet exposes the candidate-set row E and bounded fuel row U, then every bounded normalization read factors through candidate evidence, the closedness-preservation trace, App and Pi adequacy rows, the bounded trace, and the structural rows H, C, P, N. The route uses the candidate-set readback row E, the App adequacy row A, the Pi adequacy row B, the bounded trace row T, the closedness-preservation row Q, and the fuel row U; it does not export global strong normalization or completed closed consistency.*

Proof. Project the socket rows from Definition 3566.1. The candidate-set row E supplies the candidate evidence available to the bounded read, while the adequacy rows A and B keep application and Pi cases attached to that evidence.

Use Definition 3566.6. The bounded fuel row U and trace row T display the finite computation, and the closedness-preservation row Q accompanies each recorded step.

The transport and replay rows preserve the order of the read. Thus H, C, P, N can move and name only the route that has already passed through candidate evidence, adequacy rows, bounded fuel, bounded trace, and closedness preservation.

Since neither the candidate-set route nor the bounded-fuel boundary contains a coordinate for global strong normalization or completed closed consistency, those conclusions are not exported by the packet. □

Theorem 3566.9 (Normalization candidate socket fuel-boundary candidate gate)**.** *A fuel-indexed normalization read of an accepted* NormalizationCandidateSocket$^{\uparrow}$ *packet may lower or inspect the finite fuel boundary only while at least one of the candidate rows K, D, E remains visible. If K, D, E are erased, the read is forced to the frontier refusal row F and cannot export no-infinite-reduction, evidence-erased strong normalization, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.*

Proof. Project the carrier of Definition 3566.1. The strong-normalization and no-infinite-reduction exports are mediated by the candidate rows K, D, E, while the fuel row is only route data for bounded inspection.

Apply the boundary theorem Definition 3566.2. It places normalization-facing reads downstream of visible candidate evidence rather than downstream of the closed term window alone.

Use candidate-erasure refusal, Definition 3566.3. When K, D, E are removed, the packet has no candidate evidence to support a no-infinite-reduction read, so the frontier refusal row F is the only admissible boundary.

Thus fuel inspection can refine a displayed bounded trace but cannot replace reducibility-candidate evidence. The excluded strong-normalization, confluence, quotient, evaluator, and consistency conclusions require coordinates absent from the fuel boundary. □

Theorem 3566.10 (Normalization candidate socket candidate-set readback)**.** *If an accepted* NormalizationCandidat *packet exposes the candidate-set row E, then closed typed strong-normalization and no-infinite-reduction reads factor through the finite route*

$$M, E, A, B, I, H, C, P, N.$$

Bounded fuel rows may refine the displayed trace, but they cannot replace E as the candidate-set readback row. The packet therefore refuses global normalization, erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, and completed consistency.

Proof. Project the carrier of Definition 3566.1. The closed typed term window M is available to the socket, but Definition 3566.2 places strong-normalization and no-infinite-reduction exports downstream of candidate evidence.

Now specialize that boundary to the candidate-set row E. The application adequacy row A and Pi adequacy row B can refine the route through function and dependent-function cases, while I records the no-infinite-reduction read downstream of the candidate-mediated strong-normalization projection.

Use Definition 3566.6 for bounded fuel traces. A fuel row may display a finite computation and preserve closedness along that trace, but it supplies no candidate-set membership, no candidate soundness, and no beta-closure data. It therefore cannot replace E.

Finally H, C, P, N transport, replay, source, and name the candidate-set route. Since neither the base carrier nor the bounded-fuel extension contains rows for global normalization, erased candidate projection, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed consistency, those reads remain refused. □

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ dependency route reads the candidate rows K, D, E, App and Pi adequacy rows A, B, and frontier refusal row F beside the MetaCIC candidate normalization witness of chapter 6577. The socket therefore records witness-facing dependency as a horizontal route through displayed candidate evidence, not as a theorem-free shortcut from a closed term window to global normalization.

Theorem 3566.11 (Normalization candidate socket MetaCIC candidate dependency route)**.** *The* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *packet consumes candidate-mediated strong-normalization evidence and normalization-frontier rows through chapter 892 and chapter 765. Its accepted route therefore reads candidate, closed-candidate, candidate-set, App adequacy, Pi adequacy, no-infinite-reduction, and frontier-refusal rows before any kernel-facing consumer may use the socket. The socket is not a kernel-only packet and cannot export strong normalization or confluence without the displayed MetaCIC candidate dependency edge.*

Proof. Project the carrier of Definition 3566.1. The candidate rows K, D, E, adequacy rows A, B, no-infinite-reduction row I, and frontier refusal row F are all visible in the same finite socket.

Use the boundary theorem Definition 3566.2 and the diamond route Definition 3566.4. They keep normalization-facing reads downstream of the candidate evidence rows and attach the socket to the MetaCIC critical-path consumer.

Finally read the normalization frontier and critical path chapters as the candidate-mediated parents. The accepted socket route therefore depends on those MetaCIC rows rather than floating as a closed-term or kernel-only surface. □

The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ packet reads the subject-reduction discharge surface through chapter 765 and the normalization-frontier sibling of chapter 892. A candidate-normalization consumer must therefore expose the critical-path subject-reduction budget, the displayed normalization frontier rows, and the socket-local candidate evidence before treating the socket as a closed proof. The route is a forward dependency edge for MetaCIC normalization discharge, not an independent strong-normalization theorem or a replacement for the subject-reduction proof surface. The $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ route is consumed by chapter 892 and chapter 6577 only after the candidate evidence rows, endpoint obstruction row, and socket rows remain visible in the same finite packet. The frontier read therefore passes through displayed candidate evidence rather than treating the socket as a theorem-free normalization endpoint.

Theorem 3566.12 (NormalizationCandidateSocket frontier cross-reference)**.** *Every* MetaCICNormalizationFrontier *consumer of a* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *packet must first pass through the candidate evidence rows, endpoint obstruction rows, socket rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any normalization endpoint is read. The route reaches the frontier consumer of chapter 892 and the candidate witness surface of chapter 6577 only after those rows remain visible in the same finite packet.*

Proof. Project the accepted socket packet. Its public coordinates already contain the candidate evidence rows, endpoint obstruction rows, socket rows, transport, replay, provenance, and local naming.

The frontier route then consumes those displayed rows through the sibling dependency named in chapter 892. The witness route of chapter 6577 is available only as part of the same displayed candidate evidence surface.

Since no carrier coordinate stores an already-read endpoint or an external normalization result, any endpoint-facing read must traverse the listed rows before it reaches the frontier consumer. □

Theorem 3566.13 (NormalizationCandidateSocket frontier dependency)**.** *Every endpoint-facing* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ *read must expose candidate evidence rows, frontier refusal rows,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*, and candidate-normalization witness rows before it exports a normalization endpoint. The conclusion is conditional on the displayed socket packet and claims no global strong normalization, subject reduction, confluence, evaluator equality, or kernel consistency.*

Proof. Start with the frontier cross-reference of Definition 3566.12. It already routes a frontier consumer through candidate evidence, endpoint obstruction, socket, transport, replay, provenance, and local naming rows before any normalization endpoint is read.

Use the socket boundary theorem from the parent chapter. The endpoint-facing surface of $\mathsf{NormalizationCandidateSocket}^{\uparrow}$ is therefore not available until the finite packet has displayed the candidate evidence and refusal rows that keep the endpoint conditional.

The $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ row supplies the frontier side of that refusal, while the $\mathsf{MetaCICCriticalPath}^{\uparrow}$ row records the critical-path dependency that must remain visible to endpoint consumers. The candidate-normalization witness rows are read as part of the same finite packet, not as an external evaluator equality or kernel theorem.

Thus an endpoint-facing read can export only the conditional endpoint licensed by those displayed rows. It cannot turn the socket into global strong normalization, subject reduction, confluence, evaluator equality, or kernel consistency. □

Chapter 3567

A Concrete Naming Certificate for MetaCICSnCandidateInterface$^{\uparrow}$

The MetaCICSnCandidateInterface$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization frontier of chapter 9386 as finite MetaCIC NameCert data. It sits beside the normalization frontier of chapter 892, the closure trace of chapter 795, the normalization certificate router of chapter 1108, and the closed-term substitution boundary of chapter 527. The packet names candidate evidence, closed-candidate routing, bounded-normal endpoint data, and subject-reduction frontier retention without claiming evidence-free closed-term normalization.

Definition 3567.1 (MetaCIC SN candidate interface carrier)**.** *A* MetaCICSnCandidateInterface$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, K, L, B, E, F, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term window in scope for the normalization read.*
2. *K is the candidate evidence row, allowing candidate, closed-candidate, or candidate-set evidence to be displayed explicitly.*
3. *L is the closed-candidate routing row that binds the evidence in K to the closed-term boundary consumed by downstream readers.*
4. *B is the bounded normalizer evidence row, recording the finite endpoint data available at the inspected fuel or budget.*
5. *E is the endpoint readback row that exposes the bounded normal form only after T, K, L, B have been displayed.*
6. *F is the subject-reduction frontier retention row, keeping the required subject-reduction discharge boundary visible instead of erasing it into an unconditional theorem.*
7. *H stores componentwise* hsame *transport for closed typed terms, candidate evidence, closed-candidate routing, bounded endpoint data, frontier retention, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay for candidate-mediated normalization requests.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICClosureTrace}^{\uparrow}$, $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$ *row.*

The classifier accepts only packets whose strong-normalization-facing read traverses $T, K, L, B, E, F, H, C, P, N$. It has no coordinate for unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Lean4 ✓ `BEDC.Derived.MetaCICSnCandidateInterfaceUp.MetaCICSnCandidateInterfaceTasteGate single carrier alignment`

Theorem 3567.2 (MetaCIC SN candidate interface NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ *carrier $S = (T, K, L, B, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$ *obligations are exactly carrier admission, closed typed term scoping, candidate evidence admission, closed-candidate routing, bounded normalizer endpoint readback, subject-reduction frontier retention, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3567.1. Carrier admission exposes exactly the closed term window, candidate evidence, closed-candidate route, bounded endpoint data, frontier retention, and structural rows.

Read the candidate side first. The rows T, K, L ensure that a closed typed term is paired with explicit candidate evidence and that the evidence is routed through the closed-term boundary before any endpoint readback is consumed.

The bounded endpoint row is then inspected through B and E. This read records only finite normalizer evidence at the displayed budget, not a global normalization theorem.

Finally, F, H, C, P, N keep the subject-reduction frontier, transport, replay, provenance, and local name visible. Since the carrier lacks evidence-free normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, and completed consistency rows, the listed obligations exhaust the packet. □

Theorem 3567.3 (MetaCIC SN candidate interface frontier refusal)**.** *Any closed-term strong-normalization consumer of an accepted* $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ *packet is refused unless the route exposes candidate evidence, bounded normalizer evidence, subject-reduction frontier retention, transport, replay, provenance, and local naming rows. The packet cannot be read as arbitrary closed-term normalization or as a completed MetaCIC consistency theorem.*

Proof. Start from the closed typed term window T. The carrier makes T visible only together with the candidate evidence row K and closed-candidate routing row L.

Inspect the endpoint data. Rows B and E provide bounded normalizer evidence and endpoint readback at the displayed budget, so a consumer cannot skip directly from a closed term to a strong-normalization conclusion.

Now read the frontier row F. It retains the subject-reduction boundary as part of the packet rather than discharging it as an unconditional theorem.

The structural rows H, C, P, N preserve and name the same candidate-mediated route. Therefore any consumer missing one of these rows is refused, and the packet cannot export evidence-free closed-term normalization or completed consistency. □

Falsifiable prediction: If a MetaCIC strong-normalization consumer exports a closed-term SN row without candidate evidence, bounded normalizer evidence, subject-reduction frontier retention, transport, replay, provenance, and local naming rows, then the packet is refuted.

Independence witness: metacicnormalizationfrontier, metacicclosuretrace, metacicnormalizationcertificaterouter, closedtermsubstitutionboundary: $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate evidence, closed-candidate routing, bounded-normal endpoint data, and subject-reduction frontier retention.

The Lean TasteGate site for $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ provides explicit Nontrivial MetaCICSnCandidateInter and FieldFaithful $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ instances in `TasteGate.lean`. These instances preserve candidate evidence, closed-candidate routing, bounded endpoint readback, subject-reduction frontier retention, transport, replay, provenance, and local naming as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSnCandidateInterface}^{\uparrow}$ packet is generated from a closed typed MetaCIC term window, candidate evidence, closed-candidate routing, bounded normalizer endpoint data, subject-reduction frontier retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3567.1, Definition 3567.2, and Definition 3567.3 seed the finite candidate-mediated strong-normalization interface for MetaCIC readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICSnCandidateInterface}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetaCICSnCandidateInterfaceUp.MetaCICSnCandidateInterfaceTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close arbitrary closed-term normalization, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, completed closed consistency, full MetaCIC strong normalization, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence routing, bounded endpoint readback, subject-reduction frontier retention, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSnCandidateInterfaceUp.

$\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ reads its finite family-modulus packet as a consumer of chapter 388, chapter 369, and chapter 413, while the routed child files below keep the compact-net, equicontinuity, and completion handoffs separate.

Chapter 3568

A Concrete Naming Certificate for $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet records the finite-modulus subsequence route for an equicontinuous family on a compact metric source. It consumes the uniform-continuity witness pattern of chapter 3444, the compact source surface of chapter 129, the displayed map rows of chapter 130, the window schedules of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the dyadic tolerance arithmetic of chapter 291. The chapter names a finite extraction certificate; it does not assert general open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Concretely, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ consumes the equicontinuity-family rows of chapter 4076 and the compact metric finite-net rows of chapter 129 before exporting a finite modulus packet.

It also consumes the compact finite-net and uniform-modulus route of chapter 7330 together with the equicontinuity obligation surface of chapter 4076 before exposing the Arzela–Ascoli finite-modulus carrier.

The same finite-modulus route is consumed by the downstream selector lattice of chapter 4076, chapter 388, and chapter 914. Before selector rows are exported, the route also consumes the finite ProbeBundle realization route of chapter 3957, which supplies compact metric coverage, continuous pointwise moduli, RatUp lower-bound folds, and UniformModulusUp handoff rows for $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$. It also reads the finite compactness certificate of chapter 3586, whose compact-metric, finite-net-equicontinuity, compact-uniform-continuity-modulus, and finite-epsilon-net rows are available before the selector row is exported. The downstream equicontinuity and compact-uniform-continuity chapters consume the shared finite-modulus row M only through this displayed lattice, so their dependency on $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ is a row-level handoff rather than a hub-level citation. Those siblings may read the packet only after the compact metric, continuous-map, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows are displayed. In row form, the selector lattice receives exactly $K, F, M, W, R, E, H, C, P, N$: K supplies the compact finite-net source, F the inspected continuous-map family, M the shared finite modulus, W the windows, R the rational readback, E the terminal real seal, and H, C, P, N the transport, replay, provenance, and local naming rows.

The finite-modulus extraction route also consumes the compact-uniform-continuity surface of chapter 388 and the uniform modulus ledger of chapter 369 before exporting any Arzela–Ascoli selection row. Compact finite nets provide the domain-side finite cover, while $\mathsf{UniformModulus}^{\uparrow}$ supplies the image-side modulus ledger that keeps the extracted windows tied to one displayed

tolerance schedule.

Before any selector-facing row is exported, the same finite-modulus extraction also consumes compact oscillation and compact uniform modulus rows, for example through chapter 2299. The dependency is row-level: compact finite-net data, oscillation or modulus ledgers, $\mathsf{UniformModulus}^{\uparrow}$ handoff, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows are exposed before selection data, without selected subsequence choice or host function-space compactness.

Definition 3568.1 (Arzela–Ascoli finite-modulus carrier)**.** *An* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, M, W, R, E, H, C, P, N)$$

with the following rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source certificate, read only through finite net and dyadic-radius data;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family surface, inspected by finite* $\mathsf{StreamName}^{\uparrow}$ *windows rather than by a host function-space object;*
3. *M is the finite common modulus ledger, routed through* $\mathsf{HeineCantor}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;*
4. *W is the finite net-window extraction surface for the chosen inspected depths;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the extracted finite values;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed pointwise limit windows;*
7. *H records componentwise* hsame *transport of K, F, M, W, R, E;*
8. *C records displayed* Cont *replay of the same finite extraction route;*
9. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a choice-selected subsequence, function-space compactness, quotient equality of real names, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusCarrier`
Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusTasteGate single carrier alignment`

Theorem 3568.2 (Arzela–Ascoli finite-modulus NameCert obligations)**.** *For the carrier of Definition 3568.1, the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-metric source admission through K, continuous-map family admission through F, finite common modulus admission through M, finite net-window extraction through W, regular rational readback through R, pointwise* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteModulusUp.ArzelaAscoliFiniteModulusNamecertObligations`

Proof. Project the carrier. The compact source row K, continuous-map row F, and finite common modulus row M are visible before any limit seal is read.

Read the extraction rows next. The finite windows W are selected from displayed net data and inspected depths, while R reads those windows as regular rational schedules through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$.

Finally the seal row E and the structural rows H, C, P, N transport, replay, source, and name exactly that finite route. Since no other coordinate appears in the carrier, no open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness obligation is introduced. □

Theorem 3568.3 (Arzela–Ascoli equicontinuous finite-net extraction)**.** *Let $A = (K, F, M, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier. Every equicontinuous extraction row factors through the compact metric finite-net data in K, the common finite modulus in M, and the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows in W before the regular rational readback R is consumed. Thus the extraction is a dyadic finite-net construction over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and not a host compactness or function-space compactness principle.*

Proof. Use the compact source row K to obtain only the displayed finite-net and radius data. The row does not expose an arbitrary open-cover compactness theorem.

Apply the finite common modulus row M. It routes the inspected family values through the same Heine–Cantor-style modulus ledger at every selected depth.

The row W then records the finite net windows, and R reads them as $\mathsf{RegSeqRat}^{\uparrow}$ schedules. Transport and replay by H and C preserve the same dyadic route, so no selected subsequence or host function-space compactness object is available. □

Theorem 3568.4 (Arzela–Ascoli subsequence nonescape)**.** *Every subsequence or limit-facing consumer of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier must pass through*

$$K, F, M, W, R, E, H, C, P, N.$$

The consumer receives finite compact-source nets, continuous-map inspection rows, a shared modulus ledger, finite extraction windows, regular rational readback, and a $\mathsf{Real}^{\uparrow}$ *seal. It receives no selected subsequence choice, no host function-space compactness theorem, no quotient equality of real names, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with Definition 3568.3. It places the compact-source nets, common modulus, finite windows, and regular rational readback before the limit seal.

Then read the seal row E. It is downstream of K, F, M, W, R and therefore names only the displayed pointwise limit windows.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier inventory in Definition 3568.1 has no field for the excluded host principles, those principles cannot escape through the subsequence consumer. □

Theorem 3568.5 (Arzela–Ascoli finite-modulus forward lattice)**.** *Every downstream read of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet by chapter 4076, chapter 388, or chapter 914 factors through the displayed rows*

$$K, F, M, W, R, E, H, C, P, N.$$

The consuming selector receives compact metric finite-net data, continuous-map inspection data, a finite common modulus ledger, finite windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ *seal, and dyadic tolerance arithmetic before it can export any finite selector row. It receives no selected subsequence choice, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3568.1. The compact metric row K, continuous-map family row F, finite common modulus row M, and window row W are visible before any selector-facing consumer can read a limit row.

Apply Definition 3568.3. It places the finite-net and modulus route ahead of the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R and keeps the dyadic tolerance arithmetic in the displayed dependency surface.

Before the selector row is exported, read chapter 3586. Its compact-metric, finite-net-equicontinuity, compact-uniform-continuity-modulus, and finite-epsilon-net rows provide the finite compactness certificate that feeds the same selector route.

Finally apply Definition 3568.4. The real seal E and structural rows H, C, P, N preserve the same finite route for the equicontinuity, compact-uniform-continuity, and compact-net-modulus selector siblings. Since no excluded host principle is a carrier coordinate, no downstream selector can consume one through this packet. □

Theorem 3568.6 (Arzela–Ascoli Heine–Cantor compactness route)**.** *Every selector-facing finite-modulus read of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier*

$$A = (K, F, M, W, R, E, H, C, P, N)$$

first consumes the compact uniform modulus route supplied by $\mathsf{HeineCantor}^{\uparrow}$*, the finite-net source data of* $\mathsf{CompactMetric}^{\uparrow}$*, the pointwise rows of* $\mathsf{ContinuousMap}^{\uparrow}$*, the tolerance ledger of* $\mathsf{UniformModulus}^{\uparrow}$*, the finite windows of* $\mathsf{StreamName}^{\uparrow}$*, the regular readback of* $\mathsf{RegSeqRat}^{\uparrow}$*, the tolerance arithmetic of* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Only after these rows are visible may the selector export a finite-modulus handoff to chapter 4076, chapter 388, or chapter 914.*

Proof. Project the carrier. The displayed rows $K, F, M, W, R, E, H, C, P, N$ expose the compact source, inspected continuous-map family, finite modulus ledger, finite windows, rational readback, terminal real seal, transport, replay, provenance, and local naming.

Apply Definition 3568.5. It places the selector-facing read behind the compact metric, continuous-map, finite-modulus, window, readback, real-seal, and dyadic tolerance rows.

Now read Definition 3568.9. The compact-uniform finite-selector route forces the same consumer to pass through the $\mathsf{HeineCantor}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ sibling rows before it reaches the compact-uniform-continuity surface.

Finally apply Definition 3568.4. Since the packet has no selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness coordinate, the Heine–Cantor compactness route remains a finite row-level dependency for every selector-facing export. □

Theorem 3568.7 (Arzela–Ascoli compact-open dependency)**.** *Every finite-modulus extraction read that compares a family of maps in an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first pass through the* $\mathsf{CompactOpenMetric}^{\uparrow}$ *finite source-net, graph-sampling, target-distance, and sup-distance rows of chapter 1903 before exporting the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector row. In the notation of Definition 3568.1, the dependency is read through*

$$K, F, M, W, R, E, H, C, P, N.$$

Proof. Project the accepted carrier. The compact source row K supplies the finite source-net surface, while the continuous-map family row F supplies the maps whose graph samples are compared.

Read the modulus and window rows M, W. They restrict the comparison to displayed finite source windows and keep the sampled values tied to one tolerance schedule.

Now apply the compact-open metric route of chapter 1903. Its target-distance and finite sup-distance rows are consumed before the regular rational readback R and terminal seal E are available to the selector.

Finally H, C, P, N transport, replay, source, and name the same route. Thus the Arzela–Ascoli finite-modulus selector receives a compact-open finite sup-distance dependency, not an unrecorded host function-space comparison. □

Theorem 3568.8 (Arzela–Ascoli finite modulus compactmetric link)**.** *Every selector-facing read through an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source,* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows,* $\mathsf{Equicontinuity}^{\uparrow}$ *family-modulus rows,* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, and* $\mathsf{Real}^{\uparrow}$ *seal before exporting selector data. In the notation of Definition 3568.1, this read is carried by the finite rows*

$$K, F, M, W, R, E, H, C, P, N.$$

Proof. Begin with the compactmetric sibling readback in Definition 3568.3. It places the compact metric finite-net source K and the common finite modulus M before the regular rational readback.

Apply Definition 3568.5. The selector-facing lattice then reads the continuous-map family rows, equicontinuity family-modulus rows, finite windows, dyadic tolerance arithmetic, and terminal $\mathsf{Real}^{\uparrow}$ seal in the same displayed route.

Use Definition 3568.7 for the compact-open comparison. It keeps graph-sampling and finite sup-distance data downstream of the compact metric and continuous-map rows.

Finally apply Definition 3568.4. The seal and structural rows E, H, C, P, N prevent selector data from escaping the finite row surface, so no selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness is exported. □

Theorem 3568.9 (Arzela–Ascoli compact-uniform finite-selector route)**.** *Any* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-selector consumer of an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first read*

$$K, F, M, W, R, E, H, C, P, N$$

together with the compact metric, continuous map, Heine–Cantor, $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *sibling rows. The route then lands in chapter 388 only through the finite selector surface also consumed by chapter 914.*

Proof. Project the carrier of Definition 3568.1. The rows $K, F, M, W, R, E, H, C, P, N$ expose the compact metric source, continuous-map family, finite common modulus, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming.

Apply Definition 3568.5. It already forces every downstream finite-selector read to pass through the compact metric, continuous-map, Heine–Cantor, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before any selector-facing handoff is available.

Therefore the compact-uniform-continuity consumer reaches chapter 388 by the same finite route that feeds chapter 914, with no selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle inserted. □

Falsifiable prediction: Every consumer of an $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet must read the compact metric finite nets, continuous-map family windows, finite common modulus, regular rational readback, and $\mathsf{Real}^{\uparrow}$ seal before consuming the extracted limit row.

Independence witness: compactmetric, continuousmap, heine-cantor, streamname, regseqrat, real, dyadicratcore: $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ is not bijective to any listed sibling because it

jointly carries compact-source nets, a continuous-map family surface, a finite common modulus ledger, finite extraction windows, regular rational readback, Real sealing, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet is generated from compact metric finite nets, continuous-map family windows, a finite common modulus ledger, finite extraction windows, regular rational readback, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3568.1, Definition 3568.2, Definition 3568.3, and Definition 3568.4 bind the finite-modulus extraction surface for one displayed equicontinuous family.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Scoped closure requires the finite-net extraction rows, Real seal row, and NameCert dependency surface to be formalized for BEDC.Derived.ArzelaAscoliFiniteModulusUp.

Lemma 3568.10 (Arzela–Ascoli compact-metric forward lattice)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net carrier of chapter 129 and the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *modulus route before a finite selector row can be read. The dependency remains the displayed finite lattice through compact metric nets, compact-uniform-continuity moduli, inspected stream windows, regular rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal; it does not expose open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3568.11 (Arzela–Ascoli compact metric forward route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows of chapter 129 and the* $\mathsf{Equicontinuity}^{\uparrow}$ *pointwise modulus rows of chapter 4076 before exporting its finite subfamily modulus route. In the carrier of chapter 3568, the compact metric source row is read before the equicontinuity family row, the shared finite modulus row, the finite window ledger, the regular rational readback, and the terminal real seal.*

Lean4 ✓ `BEDC.Derived.Arzela ascoli finite modulusUp.ArzelaAscoliFiniteModulusCompactmetricForwardRoute`

Proof. Project the accepted $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ carrier through the local NameCert chapter chapter 3568. The compact metric coordinate supplies only finite-net centers, radius rows, and source coverage windows.

Read the equicontinuity coordinate next. Its pointwise modulus rows are attached to the displayed family and to the compact metric source windows, so the shared finite modulus cannot be exported before those two dependency surfaces are visible.

The finite subfamily modulus route then reads the window ledger, regular rational readback, real seal, transport, replay, provenance, and local naming rows. Since neither dependency chapter exports a selected subsequence or host function-space compactness row, the Arzela–Ascoli packet remains a finite forward route from $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{Equicontinuity}^{\uparrow}$ into the local selector surface. □

Theorem 3568.12 (Arzela–Ascoli compact metric forward reference)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *extraction first reads the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row* K *of chapter 129 before any selector output, finite window, regular rational readback, or terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route then passes through the continuous-map row* F*, finite modulus row* M*, window row* W*, readback row*

R, seal row E, and structural rows H, C, P, N; it exports no open-cover compactness theorem, selected subsequence choice, host function-space compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3568.1. The compact source row K is the only coordinate that names compact metric data, and it is read as finite-net and dyadic-radius information.

Apply the finite-net extraction theorem Definition 3568.3. It places K before the shared modulus row M, the inspected windows W, and the regular rational readback R.

Finally use the nonescape theorem Definition 3568.4. The terminal seal E and the structural rows H, C, P, N preserve the same finite route, so no selector-facing consumer can bypass the $\mathsf{CompactMetric}^{\uparrow}$ source row or replace it by an excluded host compactness principle. □

Theorem 3568.13 (Arzela–Ascoli finite modulus compact-metric sibling readback)**.** *Any finite-modulus selector read through an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first consume the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows of chapter 129 and the* $\mathsf{ContinuousMap}^{\uparrow}$ *family rows before the selector row can be exported. The route passes through the compact-uniform-continuity finite-modulus handoff of chapter 7330; it therefore exposes finite covers, continuity witnesses, and the shared modulus ledger before a downstream consumer may read the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector.*

Proof. Read the compact-metric sibling first. The finite-net rows named by chapter 129 provide the bounded finite cover surface on which a modulus selector can be evaluated.

Next read the $\mathsf{ContinuousMap}^{\uparrow}$ family rows. These rows keep the maps in the family visible as continuous data rather than as an external collection of functions.

Finally pass through chapter 7330. Its finite-modulus handoff is the point where compactness and continuity are combined, so the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ selector row can be exported only after those sibling rows have been consumed. □

Theorem 3568.14 (Arzela–Ascoli compact-equicontinuity finite-modulus dependency)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector route consumes three sibling surfaces before the Arzela–Ascoli readback is public: the compact finite-net source of chapter 129, the shared family-modulus discipline of chapter 4076, and the compact-uniform finite-modulus handoff of chapter 7330. In carrier rows, the compact metric dependency supplies K, equicontinuity supplies the family-level constraint read by F and M, and compact-uniform continuity supplies the finite modulus handoff consumed before W, R, E, H, C, P, N are exported.*

Proof. Start with the compact metric sibling. Its finite-net source gives the only compact-domain row available to the selector route, so the readback cannot begin from an unrecorded host compactness theorem.

Next consume the equicontinuity sibling. The shared family-modulus rows constrain the continuous-map family before the finite-modulus row M can be read as a common selector budget.

Finally use the compact-uniform finite-modulus handoff. It combines compact finite nets with the displayed continuity rows before the stream windows, regular rational readback, and terminal real seal are exposed. Therefore the Arzela–Ascoli readback is a consumer of those three sibling surfaces, not a kernel-only packet detached from the dependency lattice. □

Theorem 3568.15 (Arzela–Ascoli compact-metric sibling route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector row consumes the compact-metric finite-net source of chapter 129, the equicontinuity family rows, the compact-uniform-continuity finite-modulus handoff, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$

seal, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the local* $\mathsf{NameCert}$ *rows before the selector is public. In displayed carrier order the route is*

$$K,\ F,\ U,\ R,\ E,\ Q,\ H,\ C,\ P,\ N,$$

where K is the compact-metric finite-net source, F is the family-level equicontinuity surface, U is the compact-uniform finite-modulus handoff, R is regular-rational readback, E is the terminal Real seal, Q is the dyadic tolerance ledger, and H, C, P, N are componentwise transport, replay, provenance, and local naming.

Proof. Read the displayed row order. The compact-metric source K is first because the selector cannot choose a finite modulus over an unnamed compact domain; its substantive sibling source is chapter 129.

Next consume the equicontinuity family row F and the compact-uniform handoff U. These rows expose the family constraint and the finite-modulus transfer before any rational or Real-facing readback occurs.

Finally read R, E, Q, H, C, P, N in the stated order. The regular-rational row, Real seal, dyadic tolerance ledger, transport, replay, provenance, and local naming rows together make the selector public without adding an external compactness theorem, hidden choice of subsequence, or host-level equality. □

Theorem 3568.16 (Arzela–Ascoli compact-image bounded dependency route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *finite-selector read that exports selector-facing modulus data must first consume the compact source nets of chapter 129, the displayed map rows of chapter 130, the finite tolerance ledger of chapter 369, and the bounded image radius route of chapter 3924. In carrier notation, the read stays inside*

$$K, F, M, W, R, E, H, C, P, N.$$

Thus the selector-facing finite modulus is exported only after compact source nets, uniform continuity data, bounded image radius data, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal have all been displayed.*

Proof. Project the accepted carrier of Definition 3568.1. The compact source row K, continuous-map row F, finite modulus row M, window row W, regular readback row R, terminal seal row E, and structural rows H, C, P, N are the only rows available to a selector-facing consumer.

Apply the forward lattice theorem Definition 3568.5. It places compact metric finite-net data, continuous-map inspection, finite common modulus data, finite windows, dyadic tolerance arithmetic, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal before any downstream selector row is public.

Read the compact-image boundedness sibling next. The total-bounded image certificate of chapter 3924 supplies the bounded radius route consumed after the compact source nets, continuous-map rows, and uniform modulus ledger have been displayed.

Finally use subsequence nonescape from Definition 3568.4. Since the carrier has no coordinate for open-cover compactness, selected subsequence choice, host function-space compactness, quotient real equality, or ambient completeness, the selector-facing finite modulus cannot bypass the compact-image boundedness data while remaining inside the accepted packet. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ packet consumes the finite Cauchy modulus surface of chapter 1486 together with the equicontinuity schedule rows already routed through the compact finite-net lattice. A compactness consumer therefore reads the Cauchy-modulus threshold, the

equicontinuity schedule, and the compact finite-net rows before it may inspect the Arzela–Ascoli finite-modulus packet. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ selector-facing route also consumes the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet of chapter 5549. This child route records the finite compact-net, pointwise modulus, rational lower-bound fold, and $\mathsf{UniformModulus}^{\uparrow}$ handoff rows as a dependency before any finite extraction data is public.

Theorem 3568.17 (Arzela–Ascoli compact-uniform-continuity ledger route)**.** *Every accepted* ArzelaAscoliFiniteMo *selector read that exports finite-modulus data must first consume the* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *rows through finite compact-net data, pointwise modulus rows, rational lower-bound folds, and the terminal* $\mathsf{UniformModulus}^{\uparrow}$ *handoff. In the notation of Definition 3568.1, the read remains inside*

$$K, F, M, W, R, E, H, C, P, N.$$

It therefore exports no open-cover compactness theorem, selected subcover, selected subsequence, host function-space compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The compact source row K, continuous-map family row F, finite modulus row M, finite windows W, regular readback R, terminal seal E, and structural rows H, C, P, N are the only rows available to a selector-facing consumer.

Apply Definition 3568.5. It places compact finite-net data, inspected pointwise rows, finite tolerance ledgers, dyadic arithmetic, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal before any selector row is exported.

Read the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ route through Definition 5549.3. Its lower-bound fold and $\mathsf{UniformModulus}^{\uparrow}$ handoff supply the compact-uniform-continuity dependency consumed by the finite modulus row M.

Finally apply Definition 3568.4. Since the carrier contains no field for open-cover compactness, selected subcovers, selected subsequences, quotient real equality, countable choice, or ambient completeness, the selector-facing export cannot bypass the compact-uniform-continuity ledger. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ finite modulus is consumed by compact-uniform-continuity and equicontinuity-family routes: chapter 388 uses the compact window budget as the continuity handoff, while chapter 413 reads the same finite family modulus before exporting an equicontinuity surface. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ hub consumes the compact-uniform-continuity route of chapter 388, the shared-radius discipline of chapter 4076, and the tolerance-only ledger of chapter 369 before its finite-modulus selector is public. Thus the hub records the actual sibling lattice used by the carrier: compact finite-net rows, equicontinuity rows, uniform-modulus rows, finite windows, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal stay visible as separate dependencies rather than collapsing into a host compactness or function-space compactness theorem. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ route consumes the equicontinuity surface of chapter 4076, the compact-uniform-continuity handoff of chapter 388, and the compact metric finite-net rows before any selector-facing modulus export is public. Thus a downstream selector reads compact source data, equicontinuity bounds, compact-uniform transfer, finite windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the structural H, C, P, N rows as a displayed dependency lattice rather than as host compactness, selected subsequences, quotient real equality, or ambient completeness. The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ route consumes the compact-uniform-continuity finite modulus surface of chapter 388 together with the equicontinuity-family rows of chapter 4076 before it extracts the finite-modulus window. The same dependency route may also read the finite-net minimum fold of chapter 619 when the compact source has already exposed its finite inspection schedule, but it receives only the displayed compact source, equicontinuity, modulus-window, BHist,

hsame, Cont, Pkg, and local NameCert rows. The ArzelaAscoliFiniteModulus$^{\uparrow}$ route consumes the CompactMetric$^{\uparrow}$ finite-net data of chapter 129, the Equicontinuity$^{\uparrow}$ family-modulus rows of chapter 4076, and the CompactUniformContinuity$^{\uparrow}$ uniform-modulus handoff of chapter 388 before it exports an Arzela–Ascoli finite-modulus packet. Thus the selector-facing surface is a dependency lattice over compact source nets, shared equicontinuity bounds, compact-uniform continuity transfer, finite windows, regular rational readback, and terminal Real$^{\uparrow}$ sealing, not an isolated compactness principle, host function-space compactness theorem, selected subsequence choice, quotient real equality, or ambient completeness claim. The ArzelaAscoliFiniteModulus$^{\uparrow}$ finite-modulus rows consume the compact finite-net uniform-modulus ledger before exporting equicontinuous family bounds: chapter 388 supplies the compact-uniform-continuity handoff, while chapter 369 supplies the tolerance-to-window modulus ledger that keeps the selector-facing rows finite. The finite Ascoli modulus route consumes the compact uniform modulus row of chapter 2299 and the equicontinuous family row of chapter 413 before finite-subfamily extraction is visible. Thus an accepted ArzelaAscoliFiniteModulus$^{\uparrow}$ packet reads compact-source uniform control and shared family moduli first, then exposes the local selector, window, readback, and seal rows; it exports no host compact-open topology, selected subsequence, or detached function-space compactness principle.

Theorem 3568.18 (Arzela–Ascoli compact uniform modulus dependency route)**.** *Every accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *selector-facing read consumes the* CompactUniformModulus$^{\uparrow}$ *finite-net fold of chapter 2299 as a substantive dependency before the finite-modulus row is public. The same read passes through the compact-uniform-continuity ledger surface of chapter 5549, so compact source coverage, pointwise continuity data, rational lower-bound folding,* UniformModulus$^{\uparrow}$ *output, finite windows,* RegSeqRat$^{\uparrow}$ *readback, and the terminal* Real$^{\uparrow}$ *seal remain one displayed route rather than a local Arzela–Ascoli row alone.*

Proof. Project the accepted carrier of Definition 3568.1. The selector-facing route exposes compact source data, continuous-family data, the finite modulus row, finite windows, regular rational readback, the Real seal, and the structural transport, replay, provenance, and naming rows.

Now read chapter 2299. Its CompactUniformModulus$^{\uparrow}$ finite-net fold supplies the compact-domain coverage and tolerance-only modulus handoff consumed before the Arzela–Ascoli finite modulus can be exported.

Finally read the compact-uniform-continuity ledger route through chapter 5549. That ledger keeps the compact finite-net, pointwise modulus, rational lower-bound fold, and UniformModulus$^{\uparrow}$ handoff visible to the same selector-facing consumer. Since no carrier row supplies host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness, the dependency is a displayed finite-modulus route rather than a background compactness assumption. □

The ArzelaAscoliFiniteModulus$^{\uparrow}$ dependency lattice consumes the finite-net rows of CompactMetric$^{\uparrow}$ through chapter 129, the pointwise-to-uniform modulus rows of CompactUniformContinuity$^{\uparrow}$ through chapter 388, and the tolerance schedules of CauchyModulus$^{\uparrow}$ through chapter 292.

The route is a sibling ledger for compactness and uniform-modulus consumers; it does not add a theorem body to the hub, choose a subcover, select a modulus from an ambient proof, or assert compact-open completeness. The hub route for ArzelaAscoliFiniteModulus$^{\uparrow}$ records the selector-facing dependency already present in the carrier: a finite-modulus read must pass through the compact-open metric rows of chapter 1903 and the compact-uniform handoff rows of chapter 388 before the rows $K, F, M, W, R, E, H, C, P, N$ can be exported to a downstream selector. Thus the hub links the compact-open comparison surface and compact-uniform continuity surface through

the existing finite-modulus selector, not through a host function-space compactness theorem, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 3568.19 (Compact-open equicontinuity dependency route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *finite-modulus read that compares a family of maps consumes the compact-open metric surface of chapter 1903 and the equicontinuity family surface of chapter 4076 before exporting selector-facing rows. The exported selector data therefore remains downstream of compact-open source-window comparison, equicontinuity radius discipline, finite modulus rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the structural* H, C, P, N *rows, rather than a host function-space compactness theorem or a selected subsequence principle.*

Proof. Project the carrier of Definition 3568.1. A family-comparison read already exposes compact source data, continuous-map rows, finite modulus rows, window ledgers, regular readback, the terminal Real seal, and structural replay rows.

Use the compact-open dependency route of Definition 3568.7. It places the compact-open metric comparison surface before any selector-facing finite-modulus export can be read.

Use the equicontinuity anchors Definition 3568.23 and Definition 3568.25. They force the same family comparison to consume the equicontinuity family-window and shared-radius rows before finite subfamily modulus data is public.

Thus a selector read that compares maps must pass through both sibling surfaces. Since neither sibling supplies host function-space compactness, selected subsequences, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the Arzela–Ascoli export remains a finite displayed dependency route. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ sibling lattice is read through the compact source rows of chapter 129 and the compact-uniform modulus rows of chapter 388 before either the forward finite-modulus lattice of Definition 3568.5 or the compact-uniform finite-selector route of Definition 3568.9 can export selector-facing data. The equicontinuity children Definition 3568.10 and Definition 3568.25 then consume the same displayed rows $K, F, M, W, R, E, H, C, P, N$, so the hub records a genuine lattice dependency rather than an isolated kernel float.

Theorem 3568.20 (Arzela–Ascoli sibling lattice repair)**.** *Every selector-facing export from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the sibling rows of chapter 129, chapter 388, chapter 4076, chapter 126, chapter 301, chapter 38, and chapter 291, namely*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{CompactUniformContinuity}^{\uparrow},\quad \mathsf{Equicontinuity}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Dyadi}$$

before any finite-modulus selector row is public. The export therefore reads compact source data, compact-uniform continuity data, equicontinuity discipline, finite stream windows, regular-rational readback, the terminal real-name seal, and dyadic tolerance arithmetic as separate certificate rows.

Proof. Project the carrier of Definition 3568.1. Its selector-facing surface already exposes the compact source row, finite window row, regular rational readback, real seal, transport, replay, provenance, and local naming rows.

Apply the sibling lattice routes recorded in Definition 3568.5, Definition 3568.12, Definition 3568.9, and Definition 3568.26. These routes place $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, and $\mathsf{Equicontinuity}^{\uparrow}$ before the exported selector row.

The same carrier supplies $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic. Since the export is obtained only after those displayed rows are read, no selector-facing consumer can bypass the horizontal dependency lattice. □

Theorem 3568.21 (Arzela–Ascoli compact-metric source link)**.** *Every selector-facing* ArzelaAscoliFiniteModulus$^{\uparrow}$ *read first consumes the compact source row through chapter 129 and the compact-uniform route before exporting finite-modulus selector data. Thus the selector export is downstream of the* CompactMetric$^{\uparrow}$ *source,* CompactUniformContinuity$^{\uparrow}$ *transport, and the finite carrier rows of* ArzelaAscoliFiniteModulus$^{\uparrow}$*, not a host compactness principle, selected-subsequence principle, quotient-real equality, countable-choice principle, or ambient* Real$^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier of Definition 3568.1. Its selector surface has a compact source coordinate together with finite window, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Apply Definition 3568.20. The theorem places the compact source row, compact-uniform route, equicontinuity discipline, stream windows, regular-rational readback, real seal, and dyadic arithmetic before the selector-facing export.

The compact source is therefore a named certificate dependency rather than a host compactness assumption. Because the carrier and sibling route add no coordinate for selected subsequences, quotient real equality, countable choice, or ambient Real$^{\uparrow}$ completeness, every selector-facing export must pass through the displayed compact-metric source link first. □

Theorem 3568.22 (Arzela–Ascoli finite-modulus Equicontinuity anchor)**.** *Every* ArzelaAscoliFiniteModulus$^{\uparrow}$ *finite-modulus read first consumes the shared-family modulus and compact-source route of chapter 4076 before any finite subfamily, diagonal, compactness-facing, or selector-facing handoff is public. The anchor route exposes the compact source, continuous-family ledger, tolerance row, shared radius row, uniform-modulus handoff, transport, replay, provenance, and local naming rows supplied by* Equicontinuity$^{\uparrow}$*, and only then may the finite-modulus packet read the compact-uniform-continuity, finite-window, regular-rational, dyadic-tolerance, and terminal Real rows of the Arzela–Ascoli lattice.*

Proof. Read the Equicontinuity$^{\uparrow}$ carrier first through Definition 4076.1. Its rows expose the compact metric source, continuous-family ledger, tolerance, shared radius, uniform-modulus handoff, transport, replay, provenance, and local naming before any family selector is visible.

Apply Definition 4076.3 and Definition 4076.7. These force compact-source and shared-radius consumption before a compact-uniform or finite-net consumer can use the family.

Now pass to the finite-modulus lattice through Definition 3568.24 and Definition 3568.25. Those routes identify the downstream Arzela–Ascoli rows as consumers of the equicontinuity family ledger, so finite subfamilies, diagonal data, compactness-facing handoffs, and selector outputs remain downstream of the Equicontinuity$^{\uparrow}$ anchor. □

Theorem 3568.23 (Arzela–Ascoli finite-modulus equicontinuity forward anchor)**.** *Every* ArzelaAscoliFiniteModulus *finite-modulus read first consumes the* Equicontinuity$^{\uparrow}$ *family-window and shared-radius rows before it exports compact finite-subfamily data. In carrier terms, the read must expose the equicontinuity compact source, continuous-family ledger, tolerance row, shared radius row, uniform-modulus handoff, transport, replay, provenance, and local naming supplied by chapter 4076; only then may the Arzela–Ascoli finite-modulus packet read its compact finite-subfamily, diagonal, modulus, and selector-facing rows.*

Proof. Project the existing finite-modulus rows through the equicontinuity anchor of Definition 3568.22. That route already places the Equicontinuity$^{\uparrow}$ compact source, family ledger, tolerance, shared radius, uniform-modulus handoff, transport, replay, provenance, and local naming before any finite-modulus consumer is public.

Use chapter 4076 to identify the family-window and shared-radius rows as substantive sibling data rather than optional annotations. The finite-modulus packet can therefore export compact finite-subfamily data only after those rows have been consumed.

Finally apply the downstream dependency routes of this chapter, which read finite subfamilies, diagonal data, compactness-facing handoffs, and selector rows as Arzela–Ascoli consumers of the equicontinuity family surface. This proves the forward anchor without adding a second source for compactness or equicontinuity. □

Theorem 3568.24 (Arzela–Ascoli finite-modulus equicontinuity dependency route)**.** *Every finite-modulus extraction read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier factors through the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *ledger, the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-modulus handoff, the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any selector output is public. The route exports no selected subsequence, host function-space compactness, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $A = (K, F, M, W, R, E, H, C, P, N)$. The compact metric row K, continuous-map family row F, finite modulus row M, window row W, regular-rational readback row R, and seal row E are the whole analytic surface available before a selector can read output.

Apply Definition 3568.5. It places every downstream selector read through the compact metric, continuous-map, finite common modulus, window, readback, Real seal, and dyadic tolerance rows.

Use the sibling routes of chapter 4076 and chapter 388. The equicontinuity family supplies the shared finite family ledger, while compact uniform continuity supplies the finite-modulus handoff consumed by the selector surface.

Finally use Definition 3568.9. It identifies the compact-uniform-continuity consumer as the same finite selector route, so selector output cannot become public before the listed dependency rows are displayed. □

Theorem 3568.25 (Arzela–Ascoli EquicontinuityFamily dependency route)**.** *Every finite-modulus extraction read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *carrier consumes the family-level ledger of chapter 413 before any selector row is exported. The route then passes through the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *finite-modulus handoff, the compact metric source row, the continuous-map family row, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no selected subsequence, host function-space compactness, quotient equality of real names, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or selector row detached from the family ledger.*

Proof. Project the accepted carrier $A = (K, F, M, W, R, E, H, C, P, N)$. The continuous-map family row F is not a bare list of maps: it is read as the family surface whose shared modulus and common finite-window obligations are supplied by $\mathsf{EquicontinuityFamily}^{\uparrow}$.

Apply Definition 3568.24. It already forces the finite-modulus extraction through the equicontinuity ledger, compact-uniform-continuity handoff, compact source, continuous-map rows, windows, readback, and terminal Real seal before selector output becomes public.

It remains to name the family-level source. The sibling chapter 413 supplies the displayed family ledger consumed by F and M, while chapter 388 supplies the finite-modulus handoff consumed after that ledger is visible.

Therefore a selector-facing consumer cannot bypass $\mathsf{EquicontinuityFamily}^{\uparrow}$. The rows H, C, P, N preserve only the same displayed carrier, so no selected subsequence, host function-space compactness, quotient real equality, ambient completeness, or detached selector coordinate is exported. □

Theorem 3568.26 (Arzela–Ascoli finite-modulus sibling dependency route)**.** *Every accepted* ArzelaAscoliFiniteMod *read must pass through the* CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, CompactUniformContinuityLedger$^{\uparrow}$ *of chapter 5549,* UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *sibling rows before it exports the finite-modulus handoff. The route exposes compact metric source data, equicontinuity-family discipline, compact-uniform-continuity ledger data, uniform tolerance-to-precision data, finite stream windows, regular-rational readback, and the terminal Real seal as separate rows, and it does not replace them by host compactness, selected subsequences, quotient Real equality, or ambient completeness.*

Proof. Read the finite carrier of Definition 3568.1. Its selector-facing handoff is available only after the compact source, finite-window, regular readback, Real seal, transport, replay, provenance, and local naming rows have been displayed.

Now use the sibling lattice and forward routes, including Definition 3568.5, Definition 3568.25, and Definition 3568.30. These rows insert the CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, and UniformModulus$^{\uparrow}$ dependencies before the exported finite modulus.

The compact-uniform-continuity ledger supplies the finite compact-net row and the pointwise modulus lower-bound fold consumed before the exported finite modulus.

The remaining rows are the L10 source rows StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ already carried by the same finite packet. Since the handoff is obtained by exhausting those displayed rows, no host compactness principle, selected subsequence, quotient equality, or ambient completeness row can substitute for the sibling dependency route. □

Theorem 3568.27 (Arzela–Ascoli sibling dependency export)**.** *Every selector-facing read exported from an accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *packet factors through the displayed horizontal dependency lattice*

CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, Dyadi

The export may consume compact finite-net data, equicontinuity radii, compact-uniform transfer, finite stream windows, regular rational schedules, terminal real seals, and dyadic tolerance arithmetic, but it has no coordinate for host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness.

Proof. Read the carrier rows of Definition 3568.1. The compact source row, finite-window row, regular-rational readback, terminal real seal, transport, replay, provenance, and local naming row are all displayed before the selector-facing finite-modulus handoff is public.

The sibling route theorems refine those rows. The compact-metric, equicontinuity, and compact-uniform-continuity routes place CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, and CompactUniformContinuity$^{\uparrow}$ before the exported selector row.

The L10 readback rows then supply StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ as the finite window, rational schedule, terminal seal, and tolerance-arithmetic part of the same displayed packet. Since the export is obtained only after these rows are visible, every selector-facing read factors through the stated lattice. □

Theorem 3568.28 (Arzela–Ascoli sibling route export)**.** *Every selector-facing export from an accepted* ArzelaAscoliFiniteModulus$^{\uparrow}$ *packet factors through the displayed sibling route*

CompactMetric$^{\uparrow}$, Equicontinuity$^{\uparrow}$, CompactUniformContinuity$^{\uparrow}$, UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqR

The export may read compact finite nets, equicontinuity bounds, compact-uniform-continuity transfer, tolerance-to-precision ledgers, finite stream windows, regular rational readback, the terminal

real seal, and dyadic tolerance arithmetic; it does not obtain a selector row from host compactness, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the forward lattice of Definition 3568.5. It places compact metric finite-net data, continuous-map inspection, finite common modulus rows, stream windows, regular rational readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and dyadic tolerance arithmetic before any selector-facing handoff is public.

The sibling routes refine that lattice. The compact-metric, equicontinuity, compact-uniform-continuity, and uniform-modulus dependency theorems force $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{CompactUniformCon}$ and $\mathsf{UniformModulus}^{\uparrow}$ rows to be displayed before the finite-modulus export can be consumed.

The L10 readback rows are then fixed by the same carrier. $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational schedules, $\mathsf{Real}^{\uparrow}$ seals the terminal values, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the tolerance arithmetic. Since the carrier has no coordinate for host compactness, selected subsequence choice, quotient real equality, or ambient completeness, every selector-facing export factors through the stated sibling route. □

Theorem 3568.29 (Arzela–Ascoli finite modulus proper-map consumer)**.** *Every* $\mathsf{ProperMap}^{\uparrow}$ *compact-preimage consumer that reads a finite-modulus selector from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet must first display the dependency route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{CompactUniformContinuity}^{\uparrow},\quad \mathsf{Equicontinuity}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{Dyadi}$$

The consumer may then use the finite-modulus selector as compact-preimage evidence for the proper-map boundary; it does not obtain host compactness, selected subsequences, quotient real equality, ambient Real completeness, or a ProperMap compact-preimage row detached from the displayed selector route.

Proof. Project the $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ carrier through the sibling route export. Definition 3568.28 places compact metric data, equicontinuity bounds, compact-uniform-continuity transfer, finite stream windows, regular rational readback, the terminal Real seal, and dyadic tolerance arithmetic before any selector-facing export.

Next use the existing sibling lattice repair. The lattice paragraph preceding Definition 3568.28 ties the compact source rows of chapter 129 and the compact-uniform rows of chapter 388 to the equicontinuity children, so the selector cannot be read before the displayed compact and modulus rows.

Finally attach the proper-map compact-preimage boundary. **??** requires every compact-preimage read to remain behind the source, target, continuous-map, compact-preimage, transport, replay, provenance, and local naming rows of $\mathsf{ProperMap}^{\uparrow}$. Hence a proper-map consumer may use the finite-modulus selector only after all rows in the displayed route have been shown. □

Theorem 3568.30 (Arzela–Ascoli UniformModulus forward route)**.** *Every accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *selector read must first consume the* $\mathsf{UniformModulus}^{\uparrow}$ *finite tolerance-to-precision ledger, together with the* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, before any selector-facing finite-modulus data is public.*

The read remains inside the displayed carrier route

$$K, F, M, W, R, E, H, C, P, N.$$

It does not export host function-space compactness, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The compact metric row K, continuous-map family row F, finite modulus row M, finite windows W, regular readback R, Real seal E, and structural rows H, C, P, N are the only visible selector coordinates.

Apply the existing forward lattice, Definition 3568.5. It already forces downstream selector reads to pass through the compact metric, continuous-map, finite-window, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before the selector-facing handoff.

Now read the uniform-modulus chapter, chapter 369. Its finite ProbeBundle fold supplies the tolerance-to-precision ledger that keeps the image-side windows tied to one displayed compact-continuous modulus route.

Repack the route through H, C, P, N. Since no carrier coordinate supplies host function-space compactness, selected subsequence choice, quotient real equality, or ambient completeness, the UniformModulus handoff is a dependency edge inside the finite packet rather than an added analytic principle. □

The dependence on chapter 369 is substantive: $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ consumes $\mathsf{UniformModulus}^{\uparrow}$ through the finite ProbeBundle lower-bound fold and the continuous-map modulus rows before any selector-facing modulus data is visible. The route keeps compact-domain coverage, continuous-map modulus readback, and finite image-window selection in one displayed ledger instead of treating uniform continuity as background analysis.

Theorem 3568.31 (Arzela–Ascoli Cauchy-filter completeness forward route)**.** *Every completion-facing read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet may consume Cauchy-filter completion data only after the compact-metric source, equicontinuity family, uniform-modulus handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal have already been displayed. The dependency route is read through chapter 4076, chapter 369, and chapter 3885; it exports no selected subsequence, quotient filter, selected limit, countable-choice schedule, host function-space compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3568.1. The rows $K, F, M, W, R, E, H, C, P, N$ expose the compact metric source, continuous-map family, finite common modulus, finite windows, regular readback, terminal real seal, transport, replay, provenance, and local naming before any downstream completion read is public.

Use the equicontinuity and uniform-modulus sibling routes, including Definition 3568.24 and Definition 3568.30. They force the equicontinuity family and $\mathsf{UniformModulus}^{\uparrow}$ tolerance ledger to be visible before selector-facing finite-modulus data is exported.

Now read the Cauchy-filter completion criterion and Cauchy-filter completeness surfaces. The completion consumer may inspect those rows only as a continuation of the displayed compact-source, family, modulus, window, readback, and Real-seal route, so the Cauchy-filter dependency is a forward row-level handoff rather than a hidden completeness theorem.

The refused endpoints are absent from the carrier and from the cited completion-facing route. Thus the Arzela–Ascoli finite-modulus packet can pass Cauchy-filter completeness data forward without exporting selected subsequences, quotient filters, selected limits, countable choice, host function-space compactness, or ambient Real completeness. □

Theorem 3568.32 (Arzela–Ascoli Real-and-completion dependency route)**.** *Every Real-and-completion-facing read from an accepted* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *packet consumes the route through finite modulus data, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The route therefore reads chapter 38, chapter 301, chapter 126, and chapter 291 as row-level dependencies, not as a selected limit, quotient*

stream equality, countable-choice schedule, host function-space compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier of Definition 3568.1. The finite modulus row M, window row W, regular readback row R, and terminal seal row E are part of the displayed packet.

Read the Real-and-completion route in order. The finite modulus controls which family values may be compared, the $\mathsf{StreamName}^{\uparrow}$ row bounds the inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational handoff, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells keep each read tied to the same finite precision ledger.

Only after those rows have been displayed may the consumer see the $\mathsf{Real}^{\uparrow}$ seal. Since none of the carrier rows contains a selected limit, quotient stream equality, countable-choice schedule, host function-space compactness, or ambient completeness coordinate, the dependency remains a finite row-level route. □

The $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ dependency lattice consumes the Cauchy sequential criterion of chapter 5139 only after its compact finite-modulus, equicontinuity, finite-window, regular readback, and terminal real-seal rows have been displayed. Thus a sequence-facing compactness consumer reads the same compact metric row K, continuous-family row F, finite modulus row M, window row W, readback row R, seal row E, transport row H, replay row C, provenance row P, and local naming row N before the Cauchy sequential criterion can be used as a completion-facing dependency.

Every $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ read that reaches the Cauchy sequential criterion first exposes the compact finite-modulus and equicontinuity rows named by Definition 3568.5 and Definition 3568.9. The dependency is a displayed row handoff to chapter 5139, not a host compactness theorem, selected subsequence choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Chapter 3569

A Concrete Naming Certificate for $\mathsf{CompactTotallyBounded}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactTotallyBounded}^{\uparrow}$ packet records the constructive compact-metric surface obtained from a complete metric carrier with explicit finite nets at visible positive radii. It sits between the finite-net interface of chapter 127, the limit-service interface of chapter 128, and the compact-metric interface of chapter 129. The packet names the finite bridge used by compact and continuous arguments; it does not turn metric compactness into open-cover compactness, selected limits, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3569.1 (Compact totally bounded carrier)**.** *A* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, C, T, N, E, H, K, P, L)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric row, including the point carrier, point classifier, distance witness relation, and rational comparison ledger used by every compactness read.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row. It records only displayed regular Cauchy streams, moduli, limit histories, and late-distance ledgers available inside the metric carrier.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row. For each visible positive radius it exposes a closed* $\mathsf{ProbeBundle}$ *of center histories, membership constructors, and coverage witnesses.*
4. *N is the net-radius schedule. It records the positive* $\mathsf{Rat}^{\uparrow}$ *radii at which the finite-net rows are read and the refinement relation between requested tolerances and displayed net radii.*
5. *E is the exactness row tying M, C, T, N together. It says that each distance comparison used by T and each convergence comparison used by C is a row of the same metric source M.*
6. *H records componentwise* hsame *transport for the metric, complete-metric, totally-bounded, radius, exactness, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from a compactness request through M, C, T, N, E, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactTotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $M, C, T, N, E, H, K, P, L$. It has no coordinate for open covers, selected subcovers, quotient metric points, host equality of Cauchy streams, a choice-selected finite net, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.CompactTotallyBoundedUp.compactTotallyBoundedEncodeBHist`

Theorem 3569.2 (Compact totally bounded carrier admission)**.** *For every accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier $J = (M, C, T, N, E, H, K, P, L)$, the local carrier admits exactly the metric source, complete-metric limit service, totally-bounded finite-net family, radius schedule, exactness row, transport row, replay row, provenance row, and local naming row displayed in Definition 3569.1.*

Proof. Project the accepted packet. The mathematical rows are M, C, T, N, E, and the structural rows are H, K, P, L.

No other coordinate is present in the carrier. Hence any accepted packet is admitted through exactly those rows, with the metric, completion, finite-net, radius, exactness, transport, replay, provenance, and naming data listed in the definition. □

Theorem 3569.3 (Compact totally bounded classifier stability)**.** *The classifier of an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier is stable under componentwise* hsame *transport of the metric row, complete-metric row, totally-bounded finite-net row, radius schedule, and exactness row. The transported packet has the same displayed* Cont *replay,* Pkg *provenance, and local* NameCert *surface.*

Proof. Use the row H. It records componentwise transport for M, C, T, N, E and for the structural rows.

The replay row K reads only the transported coordinates, and P, L record only the same provenance and local naming after transport. Thus the classifier cannot change by moving to an unrecorded metric, net family, limit service, or host compactness principle. □

Theorem 3569.4 (Compact totally bounded finite-net exhaustion)**.** *Every compactness read from an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier factors its finite coverage demand through the displayed* $\mathsf{TotallyBounded}^{\uparrow}$ *row T and the radius schedule N. For each requested positive tolerance, the read obtains only the closed* ProbeBundle *centers, membership rows, metric coverage rows, and rational comparison ledgers exposed by those two rows.*

Proof. Begin with a compactness read and inspect the finite-net coordinate. The only available finite-net family is T.

The radius at which the finite net is consumed is not chosen externally. It is read from N, and the exactness row E ties the resulting coverage distances back to the metric row M.

Therefore finite coverage is exhausted by T, N, E over the metric source. A host enumeration, a selected finite subcover, or an ambient compactness theorem has no row in the packet. □

Theorem 3569.5 (Compact totally bounded completion handoff)**.** *Every Cauchy or completion-facing read from an accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *carrier reaches the complete-metric service only through the displayed row C, after the same metric row M and exactness row E have been fixed. The handoff exports no selected limit outside the carrier and no quotient equality of Cauchy names.*

Proof. The complete-metric coordinate is C. It names the displayed streams, moduli, limit histories, and late-distance ledgers available in the carrier.

The row E ties those convergence comparisons to M, so a completion-facing consumer cannot replace the metric source or import a host equality of Cauchy names.

Since the carrier contains no selector outside C and no quotient coordinate, the completion handoff remains the displayed finite BEDC route. □

Theorem 3569.6 (Compact totally bounded nonescape)**.** *No accepted* $\mathsf{CompactTotallyBounded}^{\uparrow}$ *packet exports open-cover compactness, selected subcovers, selected limits, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its compact-metric content is exactly the finite route*

$$M, C, T, N, E, H, K, P, L.$$

Proof. Apply the carrier admission theorem. Every public read of the packet factors through the displayed tuple $M, C, T, N, E, H, K, P, L$.

Finite coverage is confined to T, N, E, and completion service is confined to C, E. The transport, replay, provenance, and local naming rows preserve those routes.

The excluded principles do not occur among the carrier coordinates. They therefore cannot be exported by an accepted packet without leaving the local $\mathsf{CompactTotallyBounded}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactTotallyBounded}^{\uparrow}$ packet is generated from a MetricSpace row, CompleteMetric limit-service row, TotallyBounded finite-net row, net-radius schedule, exactness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3569.1, Definition 3569.2, Definition 3569.3, Definition 3569.4, Definition 3569.5, and Definition 3569.6 seed the constructive compact-metric bridge from complete metric service and finite total-bounded nets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactTotallyBounded}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected limits, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite-net exhaustion, completion handoff, nonescape, and local NameCert registration for BEDC.Derived.CompactTotallyBoundedUp.

Chapter 3570

A Concrete Naming Certificate for $\mathsf{BaireCategoryTheorem}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireCategoryTheorem}^{\uparrow}$ packet records the theorem-level complete-metric Baire route as finite NameCert data. It sits over the dense-open schedule of $\mathsf{BaireCategory}^{\uparrow}$ in chapter 1542, the inspected complete-metric route of $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ in chapter 3535, the Cauchy service of $\mathsf{CompleteMetric}^{\uparrow}$ in chapter 128, the carrier comparisons of $\mathsf{Met}^{\uparrow}$ in chapter 57, the finite observation windows of $\mathsf{StreamName}^{\uparrow}$ in chapter 126, the regular-rational readback of $\mathsf{RegSeqRat}^{\uparrow}$ in chapter 301, and the terminal seal of $\mathsf{Real}^{\uparrow}$ in chapter 38. The packet names the finite route by which a complete-metric dense-open construction is exposed to downstream theorem consumers; it does not assert a host intersection object, a countable-choice selector, an open-cover compactness theorem, or ambient completeness outside the displayed route.

Definition 3570.1 (Baire category theorem carrier)**.** *A* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, U, K, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, including the finite predecessor-ball and refinement data offered by the Baire-category packet.*
2. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection row, exposing the complete-metric Baire surface through dense-open, Cauchy-thread, metric, stream, regular-rational, and real-seal reads.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row naming the Cauchy-thread and limit-service data used by the theorem-level route.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row naming the metric carrier, ball classifier, distance comparisons, and shrinkage checks consumed by B, U, K.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the schedule, centers, radii, and requested tolerances.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular-rational center, radius, and tolerance data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after B, U, K, M, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the theorem route and all structural rows.*
9. *C records displayed* Cont *replay of the dense-open, complete-metric, metric, finite-window, readback, and real-seal route.*

10. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host countable intersection, countable choice, a selected point outside the displayed readback, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BaireCategoryTheoremUp.BaireCategoryTheoremTasteGate single carrier alignment`

Theorem 3570.2 (Baire category theorem NameCert obligations)**.** *For an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier*

$$T = (B, U, K, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed theorem route, dense-open schedule exposure through B, complete-metric Baire inspection through U, Cauchy-thread service through K, metric comparison exposure through M, finite-window and regular-rational readback through S, R, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3570.1. Its analytic-facing rows are exactly B, U, K, M, S, R, E, followed by the structural rows H, C, P, N.

The dense-open schedule is exposed first through B and then inspected through U. The complete-metric Baire surface row binds that schedule to a displayed Cauchy thread, so the theorem-level route cannot begin from an unrecorded host intersection.

The rows K and M provide the complete-metric service and metric comparisons consumed by the route. The rows S and R then record the finite windows and regular-rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal E is available.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, these rows exhaust the local NameCert obligation surface. □

Theorem 3570.3 (Complete-metric Baire theorem nonescape)**.** *Every theorem-level consumer of an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

and through the structural rows BHist, hsame, Cont, Pkg, NameCert. *The consumer receives the finite complete-metric Baire route carried by $T = (B, U, K, M, S, R, E, H, C, P, N)$, not a host countable intersection, countable-choice selector, ambient complete-metric theorem, open-cover compactness theorem, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$.

Proof. Start with the NameCert obligations of Definition 3570.2. They force the consumer to read the dense-open row B, the complete-metric Baire inspection row U, the complete-metric Cauchy service K, the metric comparison row M, the finite-window row S, the regular-rational readback R, and the terminal real seal E in the displayed order.

The surface row U imports only the inspected route already named by $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$. Thus the theorem-level packet packages that route for downstream use without creating a second intersection or selector coordinate.

The structural rows H, C, P, N preserve the same route under accepted transport, replay the same observations, source the same package, and name the same local certificate. Coordinate exhaustion then blocks every excluded host object in the statement. □

Definition 3570.4 (BaireCategoryTheorem finite certificate route)**.** *For an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier* $T = (B, U, K, M, S, R, E, H, C, P, N)$*, the finite certificate route is the ordered row*

$$\mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

The source coordinate B *supplies the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule,* U *supplies the* $\mathsf{CompleteMetricBaireSurface}$ *inspection row,* K *and* M *supply the* $\mathsf{CompleteMetric}^{\uparrow}$ *and* $\mathsf{Met}^{\uparrow}$ *service rows,* S *and* R *supply the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* E *supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The structural rows* H, C, P, N *transport, replay, source, and name only this route.*

Theorem 3570.5 (BaireCategoryTheorem schedule induction)**.** *Let* $T = (B, U, K, M, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier. For every finite dense-open prefix listed in the* $\mathsf{BaireCategory}^{\uparrow}$ *schedule row* B*, the prefix is matched by a predecessor ball, a successor refinement ball, a complete-metric Cauchy-thread row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal in the same order as the route of Definition 3570.4. The structural rows* H, C, P, N *preserve the constructed prefix route under componentwise transport, displayed replay, provenance, and local naming.*

Proof. Use primitive induction over the displayed finite schedule prefix in B. The base case reads the initial ball data from the carrier of Definition 3570.1 and the metric comparison row made available by M. The NameCert obligations of Definition 3570.2 attach that initial read to the complete-metric inspection route before any theorem-level consumer can inspect it.

For the successor step, assume the predecessor prefix already has its predecessor ball, Cauchy-thread row, finite windows, regular-rational readback, and real seal. Consume the next dense-open request from B, refine inside the predecessor ball through the $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ rows U, K, and record the successor refinement ball through the same metric row M.

Append the finite observation data through S and R. The $\mathsf{StreamName}^{\uparrow}$ windows record the schedule, center, radius, and tolerance reads for the successor ball, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback converts those finite windows into the regular-rational row used before the terminal seal E.

Finally transport the extended prefix through H, C, P, N. These rows do not introduce a second schedule or an unrecorded point; they preserve the route named by Definition 3570.4. Thus every finite dense-open prefix is matched by the displayed predecessor, successor, Cauchy-thread, finite-window, readback, and real-seal rows in order. □

Theorem 3570.6 (BaireCategoryTheorem finite-prefix nested ball route)**.** *Let* $T = (B, U, K, M, S, R, E, H, C, P, N$ *be an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier. Every finite dense-open prefix recorded in the schedule row* B *determines a nested predecessor/successor ball chain in which each successor ball is read through the same metric row* M*, its radius and requested tolerance are displayed through the finite-window and regular-rational rows* S, R*, containment in the predecessor ball is preserved by the transport and replay rows* H, C*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is reachable only after that finite nested chain has been displayed. The chain is part of the finite certificate route of Definition 3570.4 and carries no host intersection object, countable-choice selector, or ambient completeness assertion outside the accepted packet.*

Proof. Use induction over the finite prefix, as in Definition 3570.5. The base case reads the first predecessor ball from B and the metric comparison row from M; the NameCert obligations of Definition 3570.2 attach those reads to the displayed theorem route before any terminal seal can be inspected.

For the successor step, assume the previous prefix has already produced a nested chain whose last ball is the current predecessor. The next dense-open request in B is consumed through the finite certificate route of Definition 3570.4, and the successor ball is read through the same M row rather than through a new metric carrier.

The radius and tolerance attached to the successor ball are then exposed through S and R. The row S supplies the finite schedule window, center, radius, and requested tolerance reads, while R records the regular-rational readback used by the route before any $\mathsf{Real}^{\uparrow}$ seal is available.

It remains to preserve containment. The transport row H carries the predecessor/successor comparison componentwise, and the replay row C displays the same containment relation inside the accepted packet. Thus the terminal seal E can be reached only after the finite nested chain has been built, transported, and replayed.

No step introduces a coordinate for a host intersection, a countable-choice selector, or ambient completeness. Those objects are absent from the carrier of Definition 3570.1, so the induction isolates exactly the finite nested-ball invariant consumed by downstream complete-metric readers. □

Theorem 3570.7 (BaireCategoryTheorem route nonescape)**.** *Every theorem-level read admitted by an accepted* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *carrier is exhausted by the finite certificate route of Definition 3570.4. In particular, any downstream complete-metric, compactness, or Baire-surface consumer receives the displayed chain through* $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *together with* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *support, and no additional host intersection, choice selector, open-cover compactness row, or ambient real-completeness row.*

Proof. Start with the carrier definition Definition 3570.1. It lists exactly the analytic rows B, U, K, M, S, R, E and the structural rows H, C, P, N.

Apply the local NameCert obligations in Definition 3570.2. Those obligations identify the dense-open schedule, complete-metric Baire inspection, Cauchy-thread service, metric comparison, finite-window readback, regular-rational readback, and terminal real seal as the only theorem-facing analytic rows.

Finally use Definition 3570.3. It already blocks host intersections, countable-choice selectors, ambient complete-metric theorems, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness outside the displayed chain. Therefore every theorem-level read is exhausted by the route named in Definition 3570.4. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3570.1, Definition 3570.2, and Definition 3570.3 seed the theorem-level complete-metric Baire route over dense-open schedules, complete-metric Baire inspection, Cauchy-thread service, metric comparison, finite windows, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host intersections, countable choice, selected points outside the displayed readback, open-cover compactness, ambient complete-metric theorems, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric Baire inspection, Cauchy-thread service, metric comparison exposure, finite-window readback, regular-rational readback, real sealing, and nonescape for BEDC.Derived.BaireCategoryTheoremUp.

Constructive story: A $\mathsf{BaireCategoryTheorem}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteMetricBaireSurface inspection row, a CompleteMetric Cauchy-thread service row, MetricSpace comparison rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3571

A Concrete Naming Certificate for $\mathsf{BaireCategoryArgument}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireCategoryArgument}^{\uparrow}$ packet records the finite nested-ball proof route that sits below the theorem-level $\mathsf{BaireCategoryTheorem}^{\uparrow}$ certificate of chapter 3570. It reads the dense-open schedule of chapter 1542, the Bishop complete metric service of chapter 3424, the metric shrinking-ball rows of chapter 57, the nested-ball and Cantor-intersection route of chapter 2318, the finite windows of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite proof argument before any host countable intersection, countable-choice schedule, or ambient completeness reading can enter.

Definition 3571.1 (Baire category argument carrier)**.** *A* $\mathsf{BaireCategoryArgument}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (B, K, M, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the dense-open schedule row, listing the finite prefix of open dense requests, predecessor balls, and requested shrinkage obligations consumed by the argument.*
2. *K is the* $\mathsf{BishopCompleteMetric}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, exposing exactly the Cauchy-thread and limit-service data used by the finite proof route.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *shrinking-ball row, carrying the metric carrier, ball classifier, distance comparisons, radius bounds, and containment checks for predecessor and successor balls.*
4. *I is the Cantor or nested-ball induction row, read through* $\mathsf{CantorIntersectionTheorem}^{\uparrow}$ *or the corresponding nested-ball surface, and recording the finite induction spine for the displayed prefix.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for centers, radii, dense-open requests, and tolerances.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the displayed center, radius, and tolerance windows as regular rational data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the nested-ball prefix, complete-metric service, metric shrinkage checks, finite windows, and regular readback have all been displayed.*
8. *H records componentwise* hsame *transport for B, K, M, I, W, R, E and for every predecessor, successor, radius, tolerance, and seal row exposed by them.*

9. *C records displayed* Cont *replay from each dense-open request through the complete-metric service, shrinking-ball construction, nested-ball induction row, finite windows, regular readback, and terminal seal.*

10. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryArgument}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{BishopCompleteMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CantorIntersectionTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryArgument}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for a host intersection point, countable-choice selector, quotient real equality, open-cover compactness theorem, or ambient completeness witness outside the displayed route.

Theorem 3571.2 (Baire category argument NameCert obligations)**.** *For every accepted* $\mathsf{BaireCategoryArgument}^{\uparrow}$ *carrier $A = (B, K, M, I, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryArgument}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows; dense-open prefix exposure through B; Bishop complete metric service through K; shrinking-ball metric comparison through M; nested-ball induction through I; finite-window and regular-rational readback through W, R; terminal Real sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 3571.1. Its analytic rows are exactly B, K, M, I, W, R, E, and its structural rows are exactly H, C, P, N.

The dense-open schedule row B supplies only finite prefixes. The complete-metric service row K and metric shrinking row M turn each request in such a prefix into a successor ball inside the predecessor ball, with all radii and distance comparisons displayed as carrier data.

The nested-ball row I records the induction spine for the same prefix. The finite-window row W and regular-rational readback row R expose the centers, radii, and tolerances before the terminal Real seal E can be inspected.

Finally, H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains no further coordinate, these rows exhaust the local NameCert obligations and block host intersections, countable-choice selectors, quotient real equality, open-cover compactness, and ambient completeness. □

Theorem 3571.3 (Baire category argument nested-ball induction)**.** *Let $A = (B, K, M, I, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BaireCategoryArgument}^{\uparrow}$ *carrier. Every finite dense-open prefix in B determines a nested predecessor/successor ball chain whose successor at each stage is obtained through the complete-metric service K, checked in the metric shrinking row M, recorded in the induction row I, displayed through the finite-window and regular-rational rows W, R, and sealed by E only after the finite chain has been built. The structural rows H, C, P, N preserve the same chain under transport, replay, provenance, and local naming.*

Proof. Induct on the displayed finite dense-open prefix in B.

For the empty prefix, the carrier provides the initial metric context through M and the complete-metric service boundary through K. The induction row I records the empty nested-ball spine, while H, C, P, N transport, replay, source, and name that base route.

For the successor step, assume the predecessor prefix has already produced a nested chain whose last ball is displayed in M and whose window data have been read through W, R. Consume the next dense-open request from B and use the complete-metric service K to obtain the successor-ball obligation inside the predecessor ball.

Record the metric part of the step in M: center, radius, containment, and shrinkage comparisons are all displayed there. Append the step to the nested-ball induction row I, then expose the corresponding finite windows and regular rational readback through W and R.

The terminal seal E is read only after those rows are visible. Transport and replay through H and C, and provenance and local naming through P and N, preserve the same finite chain without adding a host intersection point, countable-choice selector, quotient real equality, or ambient completeness witness. □

Theorem 3571.4 (Baire category argument theorem-route handoff)**.** *Every* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *consumer that reads a* $\mathsf{BaireCategoryArgument}^{\uparrow}$ *packet receives the finite nested-ball route*

$$B \longrightarrow K \longrightarrow M \longrightarrow I \longrightarrow W, R \longrightarrow E$$

together with H, C, P, N*. Thus the theorem-level Baire route may consume the argument as finite dense-open prefix data, complete-metric service, metric shrinking checks, nested-ball induction, finite-window readback, regular-rational readback, and terminal Real sealing, but not as a host countable intersection object or countable-choice construction.*

Proof. Apply the NameCert obligations of Definition 3571.2. They list the readable rows of an accepted argument packet and place the dense-open schedule, complete-metric service, shrinking-ball metric row, nested-ball induction row, finite-window row, regular readback row, and Real seal in the displayed route.

Use the nested-ball induction theorem Definition 3571.3. It shows that each finite dense-open prefix has already been converted into a predecessor/successor ball chain before the terminal seal is available.

A theorem-level $\mathsf{BaireCategoryTheorem}^{\uparrow}$ consumer therefore receives precisely that finite route and the structural rows H, C, P, N. Since the argument carrier has no coordinate for a host countable intersection, choice selector, open-cover compactness, quotient real equality, or ambient completeness, none of those objects can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryArgument}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryArgument}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3571.1, Definition 3571.2, Definition 3571.3, and Definition 3571.4 seed the finite nested-ball proof route behind the complete-metric Baire theorem packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryArgument}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host countable intersections, countable choice, selected generic points, quotient real equality, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open prefix exposure, complete-metric service, shrinking-ball exactness, nested-ball induction, finite-window readback, regular-rational readback, Real sealing, and nonescape for BEDC.Derived.BaireCategoryArgumentUp.

Constructive story: A $\mathsf{BaireCategoryArgument}^{\uparrow}$ packet is generated from dense-open schedule rows, a Bishop complete metric service row, MetricSpace shrinking-ball rows, a Cantor or nested-ball induction row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 3572

A Concrete Naming Certificate for RationalBracketRefinement$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RationalBracketRefinement$^{\uparrow}$ packet records the finite rational operation that tightens a displayed bracket after a dyadic tolerance request and before the terminal Real$^{\uparrow}$ seal is read. It sits over the dyadic arithmetic core of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the real-name seal of chapter 38, the finite bracketing surface of chapter 1807, and the Archimedean approximation boundary of chapter 1507. The packet names refinement as finite rational bracket data; it is not a host-order trichotomy principle, a quotient-real equality test, a selected limiting interval, or a completeness theorem.

Definition 3572.1 (RationalBracketRefinement carrier)**.** *A* RationalBracketRefinement$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N)$$

with the following rows:

- *B is the incoming* FiniteBracketingInterval$^{\uparrow}$ *bracket row.*
- *ϵ is the* DyadicRatCore$^{\uparrow}$ *tolerance row that controls the requested rational refinement scale.*
- *L, U are the incoming rational endpoint rows and L', U' are the displayed refined rational endpoint rows.*
- *W is the* StreamName$^{\uparrow}$ *finite observation window used to inspect the named object.*
- *G is the* RegSeqRat$^{\uparrow}$ *readback row extracted from W.*
- *A is the* ArchimedeanApproximation$^{\uparrow}$ *row that supplies the finite bound and scale comparison used by the refinement.*
- *E is the terminal* Real$^{\uparrow}$ *seal reached only after the refined bracket has been displayed.*
- *H records componentwise* hsame *transport for the bracket, tolerance, endpoint, window, readback, approximation, real seal, replay, provenance, and local name.*
- *C is the displayed* Cont *replay row for the same finite route.*
- *P is the* Pkg *provenance over* RationalBracketRefinement$^{\uparrow}$, FiniteBracketingInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, ArchimedeanApproximation$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

- *N is the local* $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$ *row.*

The carrier accepts only packets in which the refined endpoints are read after the tolerance row and before the real seal, using the same finite window, regular rational readback, and Archimedean approximation rows.

Lean4 ✓ `BEDC.Derived.RationalBracketRefinementUp.RationalBracketRefinementTasteGate single carrier alignment`

Theorem 3572.2 (RationalBracketRefinement NameCert obligations)**.** *For every accepted* RationalBracketRefinem *carrier*

$$R = (B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission for the incoming* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, incoming and refined rational endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{ArchimedeanApproximation}^{\uparrow}$ *scale row,* $\mathsf{Real}^{\uparrow}$ *terminal seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation may read host real order, quotient equality of real names, trichotomy, selected infinite refinement, or ambient completeness.*

Proof. Project the accepted carrier of Definition 3572.1. The rows $B, \epsilon, L, U, L', U', W, G, A, E, H, C, P, N$ are the entire inventory available to the local certificate.

The bracket row B supplies the incoming endpoint surface, and the dyadic tolerance row ϵ fixes the finite refinement scale. The refined endpoint rows L', U' are therefore checked against displayed rational and dyadic data, not against a host real-order oracle.

The window row W and the readback row G keep the inspected object on the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route. The Archimedean approximation row A supplies the finite scale handoff, and the real seal E is available only after those rows have been displayed.

Transport, replay, provenance, and local naming are the remaining structural rows H, C, P, N. Since the carrier has no coordinate for trichotomy, quotient equality, selected infinite refinement, or completeness, none of those principles can enter the local NameCert obligations. □

Theorem 3572.3 (RationalBracketRefinement L10 handoff)**.** *Every Real-facing or Completion-facing consumer of an accepted* $\mathsf{RationalBracketRefinement}^{\uparrow}$ *packet reaches the refined rational bracket only through the finite route*

$$\mathsf{FiniteBracketingInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{ArchimedeanApproximation}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff displays the incoming bracket, the dyadic tolerance, the regular rational readback, and the Archimedean scale row before the terminal $\mathsf{Real}^{\uparrow}$ *seal is exported.*

Proof. Begin with a consumer requesting a refined bracket for a displayed real name. The incoming bracket row B and the dyadic tolerance row ϵ are read first, so the refinement request has a finite endpoint surface and a finite scale.

Next open the finite window W and its regular rational readback G. These rows keep the computation on the StreamName and RegSeqRat route before any Real-facing consumer can inspect the terminal seal.

The Archimedean approximation row A binds the bound and scale data to the same finite packet, and the refined endpoint rows L', U' are then available as rational bracket data. The real seal E is exported only after this displayed route has been traversed.

Because the route never introduces host order, quotient equality, selected limiting intervals, or completeness, the handoff exports only the finite rational bracket refinement surface named by the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBracketRefinement}^{\uparrow}$ packet is generated from an incoming FiniteBracketingInterval bracket row, a DyadicRatCore tolerance row, incoming and refined rational endpoint rows, StreamName finite observation windows, RegSeqRat readback, ArchimedeanApproximation scale data, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3572.1, Definition 3572.2, and Definition 3572.3 seed the finite rational bracket refinement route through FiniteBracketingInterval brackets, DyadicRatCore tolerance, incoming and refined rational endpoint rows, StreamName windows, RegSeqRat readback, ArchimedeanApproximation scale data, terminal RealUp seal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBracketRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real order, quotient equality of real names, trichotomy, selected infinite refinement, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket endpoint visibility, dyadic tolerance routing and exactness, endpoint refinement exactness, StreamName window exposure, RegSeqRat readback, Archimedean scale handoff, terminal seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RationalBracketRefinementUp.

Chapter 3573

A Concrete Naming Certificate for LocatedCompactIntervalOscillationBound$^{\uparrow}$

Following Bishop and Bridges on constructive compact intervals and finite moduli, the LocatedCompactIntervalOscill packet records the finite route by which a continuous map on a located compact interval receives a displayed oscillation bound from interval nets and dyadic tolerance ledgers. It sits after the finite-net source of chapter 1808, the compact interval surface of chapter 3431, the continuous-map graph rows of chapter 130, and the tolerance-window route of chapter 2224. The packet names only the L10 finite boundedness handoff through displayed rows; it does not assert Heine–Borel compactness, choice-selected extrema, quotient-real equality, host supremum existence, or ambient Real$^{\uparrow}$ completeness.

Definition 3573.1 (Located compact interval oscillation-bound carrier)**.** *A* LocatedCompactIntervalOscillationBoun *carrier is a finite* BHist *packet*

$$B = (I, K, F, N, M, D, W, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the located compact interval source row, read through the located endpoint and interval-compactness interfaces before any bound request is accepted.*
2. *K is the* LocatedIntervalCompactness$^{\uparrow}$ *row recording the compact-interval admission used by the packet.*
3. *F is the* ContinuousMap$^{\uparrow}$ *graph row for the map whose interval oscillation is bounded.*
4. *N is the* LocatedClosedIntervalFiniteNet$^{\uparrow}$ *finite-net row, listing the inspected interval net and mesh scale.*
5. *M is the* OscillationModulus$^{\uparrow}$ *row converting the displayed oscillation request into a finite tolerance-window schedule.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger carrying requested radii, mesh slack, and comparison budgets for N and M.*
7. *W is the* StreamName$^{\uparrow}$ *finite-window row used to inspect interval endpoints, net points, graph reads, and oscillation witnesses.*
8. *R is the* RegSeqRat$^{\uparrow}$ *readback row exposing the rational observations consumed by the oscillation bound.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed bound, readable only after I, K, F, N, M, D, W, R have been displayed.*

10. *H records componentwise* hsame *transport for the interval, compactness, map, finite-net, modulus, tolerance, window, readback, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the located interval and map rows through the finite net and oscillation modulus into the Real-facing seal.*

12. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$, $\mathsf{LocatedIntervalCompactness}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, K, F, N, M, D, W, R, E, H, C, P, L$. It has no coordinate for open-cover compactness, Heine–Borel subcovers, choice-selected extrema, quotient stream equality, host supremum data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompactIntervalOscillationBoundUp.LocatedCompactIntervalOscillationBoundTas`
`single carrier alignment`

Theorem 3573.2 (Located compact interval oscillation-bound NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ *carrier $B = (I, K, F, N, M, D, W, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located compact interval source, compactness row, continuous-map row, finite-net row, oscillation-modulus row, dyadic tolerance ledger, finite stream windows, rational readback, terminal Real seal, transport, replay, provenance, and local naming;*

2. *source compatibility, requiring K and F to read the same located interval source I before the finite net or modulus is consumed;*

3. *finite-net exactness, requiring N to expose the mesh and inspected net through D and W without importing an open-cover compactness theorem;*

4. *oscillation-modulus exactness, requiring M to consume the displayed continuous-map row F together with the finite net N and the same dyadic tolerance ledger D;*

5. *Real seal nonescape, requiring the bound seal E to be downstream of I, K, F, N, M, D, W, R and forbidding choice-selected extrema, quotient-real equality, host supremum data, Lean verification, and bridge checking.*

No located compact interval oscillation-bound obligation is stored outside these rows.

Proof. Project the accepted carrier. The source rows are I, K, F, so the interval, compactness admission, and map graph are fixed before any finite net or oscillation modulus can be read.

The finite-net row N then supplies the inspected mesh through the dyadic ledger D and stream windows W. Because the carrier has no open-cover row or subcover selector, the net is only the displayed finite row inherited from the located interval route.

The oscillation-modulus row M consumes the same map, net, tolerance, and window data. The readback row R exposes the rational observations used by that modulus route before the terminal seal E is available.

The structural rows H, C, P, L transport, replay, source, and name this route. Since no coordinate stores Heine–Borel compactness, selected extrema, quotient equality, host supremum data, Lean verification, or bridge evidence, those objects cannot become local NameCert obligations. □

Theorem 3573.3 (Located compact interval oscillation-bound route)**.** *Every accepted* LocatedCompactIntervalOscil *packet routes a Real-facing oscillation-bound read through*

$$\mathsf{LocatedIntervalCompactness}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longrightarrow \mathsf{OscillationModulus}^{\uparrow} \longrightarrow \mathsf{Dy}$$

The route exposes the located compact interval source, continuous-map graph row, finite-net row, oscillation-modulus row, dyadic tolerance ledger, finite observation windows, rational readback, and terminal Real seal before any bound is consumed. A consumer that omits one of those rows is outside the accepted carrier.

Proof. Read the carrier of Definition 3573.1. The first analytic coordinates bind the located interval source, compactness admission, and continuous-map graph.

The finite-net coordinate N supplies the inspected interval mesh. The oscillation-modulus coordinate M then converts the displayed continuity and net data into a finite tolerance-window schedule using D and W.

The rational readback R and terminal seal E are downstream of that schedule. Transport and replay preserve the same components, and provenance and local naming add no analytic row.

Thus every accepted bound follows the displayed route. If any row in the route is absent, the packet has no coordinate from which a Real-facing oscillation-bound read can be projected. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ consumer read factors through one located compact interval source, one ContinuousMap graph row, one LocatedClosedIntervalFiniteNet row, one OscillationModulus row, one DyadicRatCore tolerance ledger, one StreamName window, one RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert, and the terminal RealUp seal.

Independence witness: located-closed-interval-finite-net, located-interval-compactness, continuousmap, oscillation-modulus: no listed sibling is bijective with $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ because the packet jointly carries the compact interval source, continuous-map graph, finite-net mesh, oscillation-modulus schedule, dyadic tolerance ledger, finite windows, rational readback, Real-facing bound seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}$ packet is generated from a located compact interval source row, a ContinuousMap graph row, a LocatedClosedIntervalFiniteNet mesh row, an OscillationModulus tolerance-window row, DyadicRatCore budgets, StreamName windows, RegSeqRat readback, a terminal RealUp bound seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3573.1, Definition 3573.2, and Definition 3573.3 seed the finite L10 oscillation-bound packet over located compact interval, continuous-map, finite-net, modulus, dyadic, window, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalOscillationBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Heine–Borel compactness, selected extrema, quotient-real equality, host supremum data, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite-net exactness, oscillation-modulus exactness, Real seal nonescape, transport replay, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocatedCompactIntervalOscillationBoundUp.

Chapter 3574

A Concrete Naming Certificate for $\mathsf{CharacterTable}^{\uparrow}$

Following Serre and Fulton–Harris on finite-group characters, the $\mathsf{CharacterTable}^{\uparrow}$ packet records a finite character table as displayed NameCert data rather than as a host matrix. It consumes the finite group source of chapter 41, the trace and class-function rows of chapter 245, the Grothendieck ledger of chapter 246, and the Schur row and column ledger of chapter 1603.

Definition 3574.1 (CharacterTable carrier sketch)**.** *A* $\mathsf{CharacterTable}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, C, I, V, \Omega, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row.*
2. *C is the finite class-row ledger, listing the displayed conjugacy-class readbacks only through the group packet and not through a quotient carrier.*
3. *I is the irreducible-character row ledger, read through* $\mathsf{NameCert}_{\mathsf{CharacterTheory}^{\uparrow}}$ *and the representation-ring classifier rows.*
4. *V is the value-table row ledger, whose entries are displayed trace readbacks indexed by the accepted C and I rows.*
5. *Ω is the Schur-orthogonality ledger imported as finite row-normalization, column-normalization, and off-diagonal-zero rows.*
6. *H records componentwise* hsame *transport for the group, class, irreducible-character, value-table, orthogonality, provenance, and naming rows.*
7. *P is the* Pkg *provenance over the four sibling certificates named above,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}$ *row.*

The carrier contains no host character matrix, no quotient conjugacy-class object, no ambient complex inner product, and no selected list of representatives outside these rows.

Lean4 ✓ `BEDC.Derived.CharacterTableUp.CharacterTableTasteGate single carrier alignment`

Theorem 3574.2 (CharacterTable NameCert obligations)**.** *Every accepted* $\mathsf{CharacterTable}^{\uparrow}$ *packet satisfies the following five local NameCert obligations:*

1. *carrier admission exposes exactly the group, class-row, irreducible-character-row, value-table, Schur ledger, transport, provenance, and local naming rows of Definition 3574.1;*
2. *class-row exactness requires each class read to be sourced by the displayed finite group packet of chapter 41;*
3. *irreducible-row exactness requires each row name to pass through the trace and class-function surface of chapter 245 and the representation-ring classifier of chapter 246;*
4. *value-table stability requires* hsame *transport to preserve the displayed value rows together with the Schur row and column constraints of chapter 1603;*
5. *nonescape prevents a consumer from replacing the finite rows by a host character table, quotient class carrier, selected representative list, ambient matrix equality, or external representation category.*

Proof. Project the carrier rows of Definition 3574.1. The class ledger is available only through the consumed group row, and the irreducible-character ledger is available only through the character-theory and representation-ring source rows.

The value-table entries are then read as displayed trace values indexed by those two ledgers. The Schur ledger supplies the row and column normalization and off-diagonal rows that make the table stable under accepted transport.

Finally repack the displayed rows with H, P, and N. Since the carrier has no coordinate for a host matrix, quotient class carrier, representative selector, or external representation category, every accepted table read remains inside the finite NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3574.1 and Definition 3574.2 seed the finite character-table route over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, and $\mathsf{SchurOrthogonality}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CharacterTable}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close character-table computation, quotient conjugacy classes, host matrix equality, complete representation categories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, class-row exactness, irreducible-row exactness, value-table stability, and no-host-matrix boundary for the same finite packet.

Constructive story: A $\mathsf{CharacterTable}^{\uparrow}$ packet is generated from a finite group source, displayed class rows, displayed irreducible-character rows, value-table trace rows, Schur orthogonality rows, componentwise transport, package provenance, and a local NameCert row.

Chapter 3575

A Concrete Naming Certificate for $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$

The $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ packet realizes the Denjoy–Wolff-style fixed-end reading of section 9463.5 in chapter 9463 as finite NameCert data. It consumes the disk-coordinate surface of chapter 377, the hyperbolic metric-read rows of chapter 3474, the disk phase-transport route of chapter 3455, and the boundary-tangent comparison rows of chapter 3534. The packet names boundary dynamics without importing a completed ideal boundary, a classical holomorphic self-map theorem, or an observer substance.

Definition 3575.1 (DenjoyWolffBoundaryDynamics carrier)**.** *A* $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, M, T, B, R, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-map row, including the finite source observations and target observations accepted by the packet.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row used to compare the accepted disk observations.*
3. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *route row, recording how the disk observations move through displayed phase transport.*
4. *B is the boundary-read row, recording the finite boundary-facing observation used as the fixed far-end.*
5. *R is the nonexpansive-route row, recording that every accepted step is read through the metric and transport rows before the boundary far-end is inspected.*
6. *F is the fixed-boundary row, recording stability of the same boundary read under the displayed route.*
7. *H records componentwise* hsame *transport for disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, provenance, and naming rows.*
8. *C records displayed* Cont *replay from D, M, T into B, R, F and then into provenance and naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$*, the four sibling chapters named above,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host ideal-boundary quotient, a selected fixed point, a global contraction theorem, a physical horizon, or a completed disk compactification.

Lean4 (definition) `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsUp`
Lean4 ✓ `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsTasteGate single carrier alignment`

Theorem 3575.2 (DenjoyWolffBoundaryDynamics NameCert obligations)**.** *Every accepted* DenjoyWolffBoundaryD *packet satisfies the following finite obligations:*

1. *carrier admission exposes exactly the disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, transport, replay, provenance, and local naming rows of Definition 3575.1;*
2. *boundary-read exactness requires B to be sourced through the displayed* $\mathsf{UnitDisk}^{\uparrow}$ *and* $\mathsf{HyperbolicMetric}^{\uparrow}$ *rows;*
3. *nonexpansive-route exactness requires R to pass through the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *row before a boundary read is exported;*
4. *fixed-boundary stability requires F, H, C to preserve the same boundary-facing read under accepted route transport;*
5. *nonescape prevents any consumer from replacing these finite rows by a host fixed-point theorem, selected ideal point, quotient boundary equality, completed compactification, or physical observer trajectory.*

Proof. Project the carrier rows of Definition 3575.1. The disk-map, metric-read, and phase-transport rows are the only route by which the boundary read can be formed.

The nonexpansive-route row then records the displayed metric and transport discipline before the fixed-boundary row is inspected. Transport and replay preserve the same boundary-facing read because H and C have no endpoint outside the displayed rows.

Finally the provenance and naming rows repack the route as a local NameCert surface. Since the packet contains no host fixed-point theorem, selected ideal point, quotient boundary equality, completed compactification, or physical observer trajectory, those readings cannot be exported. □

Falsifiable prediction: Every accepted $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ read factors through displayed disk-map, boundary-read, nonexpansive-route, fixed-boundary, provenance, and local NameCert rows.

Independence witness: unitdisk, hyperbolic-metric, poincare-disk-phase-transport, hyperbolic-horocycle: none is bijective to $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ because this packet jointly carries disk-map observations, metric-read discipline, phase-transport routing, boundary-read data, nonexpansive-route rows, fixed-boundary stability, transport, replay, provenance, and local naming. The UnitDisk packet supplies coordinates without boundary dynamics; the HyperbolicMetric packet supplies metric reads without a fixed-boundary row; the Poincare disk phase-transport packet supplies movement without a Denjoy–Wolff far-end commitment; and the horocycle packet supplies boundary-tangent rows without nonexpansive disk-map dynamics.

The Lean follow-up must provide `Nontrivial DenjoyWolffBoundaryDynamicsUp` and `FieldFaithful DenjoyWolffBoundaryDynamicsUp` in `lean4/BEDC/Derived/DenjoyWolffBoundaryDynamicsUp/TasteGate.lean`

keeping $D, M, T, B, R, F, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3575.1 and Definition 3575.2 seed the finite boundary-dynamics route over displayed disk-map, metric-read, phase-transport, boundary-read, nonexpansive-route, fixed-boundary, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.DenjoyWolffBoundaryDynamicsTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close the classical Denjoy–Wolff theorem, holomorphic self-map classification, ideal-boundary quotient equality, selected fixed points, physical observer trajectories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-read exactness, nonexpansive-route exactness, fixed-boundary stability, nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.DenjoyWolffBoundaryDynamicsUp.

Constructive story: A $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$ packet is generated from UnitDisk disk-map rows, HyperbolicMetric metric reads, PoincareDiskPhaseTransport rows, a boundary-read row, a nonexpansive-route row, a fixed-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3576

A Concrete Naming Certificate for $\mathsf{BishopCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyCriterion}^{\uparrow}$ packet records the finite criterion by which a real-facing sequence displays Cauchy control before any completion seal is consumed. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion criterion route of chapter 1476. The packet names the criterion as finite BEDC NameCert data; it is not a proof of ambient $\mathsf{Real}^{\uparrow}$ completeness, a quotient-stream equality theorem, a selected limit, or a host Cauchy completion object.

Definition 3576.1 (Bishop Cauchy criterion carrier)**.** *A* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radii, comparison budgets, and slack entries used by the criterion.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite source windows inspected at the displayed tolerance requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity handoff row, assembled from the stream windows and dyadic tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *K is the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *dependency row, recording that the criterion can be consumed by completion-facing packets without exporting completion itself.*
6. *H records componentwise* hsame *transport for the tolerance, window, regularity, seal, and dependency rows.*
7. *C records displayed* Cont *replay through the same finite criterion route.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $D, S, R, E, K, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3576.2 (Bishop Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier $B = (D, S, R, E, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission, DyadicRatCore tolerance exposure, StreamName finite-window exposure, RegSeqRat regularity handoff, Real seal exposure, CauchyCompletionCriterion dependency exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3576.1. The analytic route is exactly the tolerance row D, the stream-window row S, the regularity handoff R, the terminal seal E, and the completion-criterion dependency K.

The dyadic and stream rows supply the finite Cauchy evidence. The regularity handoff then repacks those finite observations through $\mathsf{RegSeqRat}^{\uparrow}$ before a $\mathsf{Real}^{\uparrow}$ seal is available.

The dependency row K records the only completion-facing use of the criterion. It points to the displayed $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ surface without adding a completion object or an ambient completeness theorem.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no other coordinate is present, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCauchyCriterionUp.TasteGate.BishopCauchyCriterionTasteGate single carrier alignment`

Theorem 3576.3 (Bishop Cauchy criterion completion route)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionCriterion}^{\uparrow}$$

together with H, C, P, N. Consequently the consumer receives dyadic tolerances, finite stream windows, regular rational readback, a terminal real seal, and the named completion criterion dependency, but it receives no selected limit, quotient stream, countable-choice schedule, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a completion-facing read of B. The only tolerance and observation rows are D and S, so the read must expose the dyadic budgets and stream windows before any regularity handoff can occur.

The row R supplies the regular rational readback, and E supplies the terminal real seal. Thus the real-facing part is downstream of the displayed finite Cauchy observations.

The row K is the sole completion criterion dependency. It permits the packet to be consumed by the Cauchy-completion criterion route, while the absence of quotient, selected-limit, and ambient-completeness coordinates prevents any stronger export. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyCriterion}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite windows, RegSeqRat regularity readback, a Real seal, a CauchyCompletionCriterion dependency row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyCriterion}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 3576.1, Definition 3576.2, and Definition 3576.3 close the finite Bishop Cauchy criterion route for the L10 Real and Completion surface.

Formal verification: FormalStatus(NameCert$_{\text{BishopCauchyCriterion}\uparrow}$, theoremChecked)
Lean target: `BEDC.Derived.BishopCauchyCriterionUp.TasteGate.BishopCauchyCriterionTasteGate\_single\_carrier\_alig`
Bridge status: none
Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient Real completeness, or bridge checking.
Upgrade path: Public export or bridge work requires a standard interface for the checked criterion route.

Chapter 3577

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ packet records the finite equivalence route between a displayed Bishop Cauchy criterion and a displayed dyadic modulus schedule before any completion-facing consumer can read a real seal. It sits after the criterion packet of chapter 3576, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion criterion route of chapter 1476. Its downstream completion consumer is the witness route of chapter 3579; the packet does not export quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3577.1 (Bishop Cauchy modulus equivalence carrier)**.** *A* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, D, S, R, E, K, U, H, T, P, N)$$

with displayed rows Q for the $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *criterion, D for the dyadic modulus schedule, S for the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule, R for the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E for the terminal* $\mathsf{Real}^{\uparrow}$ *seal, K for the* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *dependency, U for the criterion-to-modulus transport row, H for componentwise* hsame *transport, T for displayed* Cont *replay, P for* Pkg *provenance, and N for the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$ *row. Accepted packets compare criterion and modulus data only through these rows.*

Lean4 ✓ `BEDC.Derived.BishopCauchyModulusEquivalenceUp.BishopCauchyModulusEquivalenceTasteGate`
`single carrier alignment`

Theorem 3577.2 (Bishop Cauchy modulus equivalence NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier $M = (Q, D, S, R, E, K, U, H, T, P, N)$, the local* $\mathsf{NameCert}$ *surface consists exactly of criterion-row admission, dyadic modulus exposure, stream-window exposure, regular rational readback, Real seal exposure, completion-criterion dependency exposure, criterion-to-modulus transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3577.1. The analytic part is Q, D, S, R, E, K, U: the criterion row is read first, the dyadic modulus schedule and finite stream windows display the same Cauchy demand, and the regular rational readback reaches the Real seal only through that displayed route.

The completion dependency is K, so completion-facing use remains a dependency row rather than a selected completion object. The structural rows H, T, P, N transport, replay, source, and name exactly the same finite data, leaving no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 3577.3 (Bishop Cauchy modulus equivalence consumer route)**.** *Every consumer of an accepted* $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ *carrier that claims a criterion/modulus handoff for L10 completion must factor through* $\mathsf{BishopCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionCriterion}^{\uparrow}$, *and the downstream* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *route before the local* NameCert *row is exported.*

Proof. Begin with the criterion row Q and the dyadic modulus row D. The stream windows S and regular readback R are the only rows that connect those two presentations to the terminal seal E, so a completion consumer cannot jump from a criterion statement to a Real-facing witness.

The dependency row K points to the completion criterion, and the witness consumer reads that dependency through the displayed L10 route. The rows U, H, T, P, N then record the handoff, transport, replay, provenance, and name; no additional carrier coordinate supplies quotient equality, selected limits, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}$ packet is generated from a BishopCauchyCriterion row, a dyadic modulus row, StreamName finite windows, RegSeqRat readback, a Real seal, a CauchyCompletionCriterion dependency, criterion-to-modulus transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 3577.1, Definition 3577.2, and Definition 3577.3 seed the finite criterion/modulus equivalence route for the L10 Real and Completion surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyModulusEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exposure, dyadic modulus exposure, stream-window exposure, regular readback, Real seal non-escape, completion-criterion dependency preservation, transport stability, replay stability, provenance, and local naming.

Chapter 3578

A Concrete Naming Certificate for $\mathsf{BurnsideOrbitCount}^{\uparrow}$

Following Burnside and Pólya on finite group actions and orbit enumeration, the $\mathsf{BurnsideOrbitCount}^{\uparrow}$ packet records the finite ledger by which a displayed $\mathsf{Group}^{\uparrow}$ action on a displayed $\mathsf{FinSet}^{\uparrow}$ carrier yields an orbit-count read from fixed-point rows. It sits between the group certificate of chapter 41, the finite-set certificate of chapter 112, the character ledger of chapter 245, and the representation-ring surface of chapter 246. The packet names only the finite action and fixed-row accounting surface; it does not import quotient orbit sets, host cardinal arithmetic, complete character tables, or a representation category.

Definition 3578.1 (Burnside orbit-count carrier)**.** *A* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, X, A, F, O, M, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row for the finite group endpoint.*
2. *X is the consumed* $\mathsf{NameCert}_{\mathsf{FinSet}^{\uparrow}}$ *row for the acted-on finite carrier.*
3. *A is the displayed action ledger, read as finite* Cont *rows from G and X to the endpoint rows of X.*
4. *F is the fixed-point table whose row for g lists exactly the displayed x-rows satisfying the action read $g \cdot x = x$ through the classifier of X.*
5. *O is the orbit ledger, listing carried orbit-name rows and the finite membership witnesses by which each displayed x-row is assigned to one orbit row.*
6. *M is the finite averaging ledger displaying the numerator rows $\sum_{g \in G} \#F_g$, the divisor row $\#G$, and the resulting orbit-count read.*
7. *H records componentwise* hsame *transport for the group, finite-set, action, fixed-point, orbit, averaging, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the group and finite-set rows through the action ledger into the fixed-point table, orbit ledger, and finite averaging row.*
9. *P is the* Pkg *provenance over* $\mathsf{BurnsideOrbitCount}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, X, A, F, O, M, H, C, P, N$. It has no coordinate for a quotient orbit carrier, a host set cardinal, a complex character table, an ambient representation ring, or a choice-selected orbit representative.

Lean4 ✓ `BEDC.Derived.BurnsideOrbitCountUp.BurnsideOrbitCountTasteGate single carrier alignment`

Theorem 3578.2 (Burnside orbit-count NameCert obligations)**.** *For every accepted* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier $B = (G, X, A, F, O, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the consumed group row, consumed finite-set row, action ledger, fixed-point table, orbit ledger, averaging row, transport, replay, provenance, and local naming;*
2. *action visibility, requiring every action read in A to expose its group source in G and its input and output rows in X;*
3. *fixed-point visibility, requiring every row of F to be witnessed by an action read in A and a displayed classifier comparison in X;*
4. *orbit-ledger stability, requiring O to transport along the same* hsame *rows that transport G, X, A, and F;*
5. *finite averaging exactness, requiring M to consume only the displayed fixed-point rows and the displayed group-size divisor before exposing the orbit-count read.*

No Burnside orbit-count obligation is stored outside these rows.

Proof. Project the accepted carrier. The rows G and X supply the finite group source and finite acted-on carrier, and A is the only row in which action endpoints are displayed.

The fixed-point table F is downstream of A and the classifier of X. Thus a fixed-point read must show both the action result and the displayed comparison with the original x-row.

The orbit ledger O is then read as finite assignment data over the same X rows. Its stability is exactly componentwise transport by H, because no separate quotient-orbit carrier is present.

The averaging row M consumes the displayed fixed-point rows and the displayed group-size divisor. The replay row C, provenance row P, and local naming row N repack this data as the local NameCert surface, so no host cardinal, character table, representation ring, or selected representative becomes an additional obligation. □

Theorem 3578.3 (Burnside fixed-row handoff)**.** *Every orbit-count read exported by an accepted* $\mathsf{BurnsideOrbitCount}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Group}^{\uparrow},\ \mathsf{FinSet}^{\uparrow} \longrightarrow \mathsf{ActionLedger} \longrightarrow \mathsf{FixedRowTable} \longrightarrow \mathsf{OrbitLedger} \longrightarrow \mathsf{FiniteAverage}.$$

The handoff exposes the finite group row, finite-set row, action ledger, fixed-point table, orbit-ledger stability data, and finite averaging row before any orbit count is consumed. A consumer that omits one of those rows is outside the accepted carrier.

Proof. Read the carrier of Definition 3578.1. The first two coordinates bind the consumed $\mathsf{Group}^{\uparrow}$ and $\mathsf{FinSet}^{\uparrow}$ rows, and the action coordinate A records all displayed action reads.

The fixed-row table F can only be projected after those action reads are visible. Each fixed row is therefore a finite ledger row over G, X, and A, rather than a host fixed-point subset.

The orbit ledger O records the finite orbit-name assignments transported by H. The averaging ledger M then reads the fixed rows and group-size divisor to expose the orbit-count row.

Since C, P, and N only replay, package, and name the same displayed route, every exported orbit count factors through the stated handoff. If any row in the route is absent, the packet has no coordinate from which the orbit-count read can be projected. □

Falsifiable prediction: Every accepted $\mathsf{BurnsideOrbitCount}^{\uparrow}$ consumer read factors through one $\mathsf{Group}^{\uparrow}$ row, one $\mathsf{FinSet}^{\uparrow}$ row, one displayed action ledger, one fixed-point table, one orbit ledger, one finite averaging row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Independence witness: group, finset, character-theory, representation-ring: no listed sibling is bijective with $\mathsf{BurnsideOrbitCount}^{\uparrow}$ because the packet jointly carries action reads, fixed-point rows, orbit assignment rows, finite averaging data, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}$.

Constructive story: A $\mathsf{BurnsideOrbitCount}^{\uparrow}$ packet is generated from a consumed $\mathsf{Group}^{\uparrow}$ row, a consumed $\mathsf{FinSet}^{\uparrow}$ row, a displayed finite action ledger, a fixed-point table, an orbit assignment ledger, a finite averaging row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3578.1, Definition 3578.2, and Definition 3578.3 seed the finite Burnside orbit-count packet over displayed group, finite-set, action, fixed-row, orbit-ledger, averaging, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BurnsideOrbitCount}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient orbit carriers, host cardinal arithmetic, complete character theory, representation-ring computation, selected orbit representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action visibility, fixed-point visibility, orbit-ledger stability, finite averaging exactness, transport replay, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.BurnsideOrbitCountUp.

Chapter 3579

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessWitness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ packet records the finite witness by which regular Cauchy data, finite stream windows, dyadic tolerance rows, a completeness modulus, and a terminal Real seal are read as one completion-facing surface. It sits after the regular rational sequence surface of chapter 301, the finite stream-name discipline of chapter 126, the dyadic tolerance core of chapter 291, the terminal Real-name seal of chapter 38, and the Cauchy completeness modulus packet of chapter 2051. The packet names the L10 completion witness between regular Cauchy data and the $\mathsf{Real}^{\uparrow}$ seal; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient Real completeness.

Definition 3579.1 (Cauchy completeness witness carrier)**.** *A* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, S, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row whose Cauchy data is being witnessed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the inspected stream observations used by the witness.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic budgets attached to those windows.*
4. *M is the* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *row that bounds the tail-coherence demands accepted by the witness.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D, M have all been displayed.*
6. *H records componentwise* hsame *transport for the regular readback, finite windows, dyadic ledger, completeness modulus, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-witness request through R, S, D, M, E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, D, M, E, H, C, P, N$ and their displayed replay path. It has no coordinate for quotient stream equality, selected representatives, host-limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessWitnessUp.CauchyCompletenessWitnessTasteGate single carrier alignment`

Theorem 3579.2 (Cauchy completeness witness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletenessW}$ *carrier $W = (R, S, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, classifier stability under componentwise* hsame*, witness-to-seal routing through the completeness modulus and terminal Real row, ledger nonescape for dyadic tolerance data, and sibling dependency routing through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Every completion-facing consumer must read the route

$$R \longmapsto S \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N$$

before treating the regular Cauchy data as sealed by a completion witness.

Proof. Project the carrier of Definition 3579.1. The accepted packet exposes only the regular readback row, finite stream windows, dyadic tolerance ledger, completeness modulus, terminal Real seal, transport, replay, provenance, and local naming rows.

Carrier admission and classifier stability are componentwise. The rows R, S, D, M, E are compared only by the displayed hsame transports in H, and the replay row C preserves the same order of inspection for any accepted completion-witness request.

The witness-to-seal route is forced by M and E. The completeness modulus bounds the tail-coherence demand before the Real seal is consumed, while the dyadic ledger D has already exhausted the tolerance data that could otherwise be mistaken for a detached limit choice.

Finally P and N record sibling dependency routing and the local certificate name. Since the carrier contains no quotient stream equality, selected representative, host-limit operator, countable-choice schedule, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the NameCert obligation surface. □

Theorem 3579.3 (Cauchy completeness witness Real seal route)**.** *Every completion-seal read of an accepted* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *packet factors through the displayed finite witness rows $R, S, D, M, E, H, C, P, N$. In particular, a* $\mathsf{Real}^{\uparrow}$ *consumer must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational data, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window evidence, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$ *tail-control row before it reaches the terminal seal E.*

The route exports a completion witness for displayed regular Cauchy data. It does not export quotient stream equality, a selected host limit, countable choice, trichotomy, a detached Real seal, or ambient Real completeness.

Proof. Start with a consumer that reads a completion seal. The only Real-facing terminal coordinate in the carrier is E, and E appears after the readback, finite-window, dyadic-ledger, and completeness-modulus rows.

The consumer must therefore pass through R, S, D, M. These rows provide the regular Cauchy data, the finite observations, the tolerance budget, and the tail-control witness needed before the Real seal can be named.

The support rows H, C, P, N transport, replay, source, and locally certify that same finite route. They can preserve the displayed witness, but they cannot introduce quotient equality, selected

limits, countable choice, trichotomy, detached Real seals, or ambient completeness because no such coordinate is present in the accepted packet. □

Theorem 3579.4 (Cauchy completeness witness obligation package)**.** *The carrier definition Definition 3579.1, the NameCert-obligations theorem Definition 3579.2, and the Real-seal-route theorem Definition 3579.3 together supply the obligation surface for* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*. This surface consists of carrier admission, classifier stability through componentwise* hsame*, witness-to-seal routing through the completeness modulus,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger nonescape, sibling routing through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyCompletenessModulus}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*, and local* NameCert *provenance.*

Proof. Project the finite tuple $W = (R, S, D, M, E, H, C, P, N)$ from Definition 3579.1. Carrier admission is exactly the admission of these displayed rows, with no extra coordinate for quotient stream equality, selected limits, choice, trichotomy, detached Real seals, or ambient completeness.

Apply Definition 3579.2. It identifies classifier stability, componentwise hsame transport, dyadic ledger discipline, sibling dependency routing, provenance, and the local NameCert row as the NameCert surface of the packet.

Apply Definition 3579.3. It forces every completion-seal read through $R, S, D, M, E, H, C, P, N$, so the witness-to-seal obligation passes through the completeness modulus and terminal Real seal rather than through an unstated host limit. The three cited surfaces therefore exhaust the obligation package for the finite completion witness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting the regular rational readback, finite stream windows, dyadic tolerance ledger, completeness modulus, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: regseqrat, streamname, dyadicratcore, cauchy-completeness-modulus, real: no listed sibling supplies the whole witness packet $W = (R, S, D, M, E, H, C, P, N)$, because each sibling contributes only readback, windows, tolerances, tail control, or the terminal seal while this chapter binds all five analytic faces as one finite completion witness.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ packet is generated from a RegSeqRat regular readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a CauchyCompletenessModulus tail-control row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3579.1, Definition 3579.2, and Definition 3579.3 seed the finite completion-witness route over regular rational readback, stream windows, dyadic tolerances, completeness modulus data, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, detached Real seals, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, witness-to-seal routing, ledger nonescape, sibling dependency routing, and Real-seal factorization for BEDC.Derived.CauchyCompletenessWitnessUp.

Chapter 3580

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ packet records the finite obligation package that a completion-facing witness must discharge before downstream Real-completion consumers may use the $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ route. It sits over the regular rational readback of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the witness carrier of chapter 3579. The chapter names only the obligation surface for that textbook completion witness; it does not assert quotient stream equality, selected host limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3580.1 (Cauchy completeness witness obligation carrier)**.** *A* $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (W, A, K, R, D, S, E, H, C, P, N)$$

whose rows record a displayed $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *carrier* W*, carrier-admission evidence* A*, classifier-stability evidence* K*, sibling readback rows* R, D, S*, a terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance over the witness and obligation surfaces, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}$ *row* N*. The classifier compares such carriers only by these rows and their displayed route through* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessWitnessObligationUp.CauchyCompletenessWitnessObligationTasteGate`
`single carrier alignment`

Theorem 3580.2 (Cauchy completeness witness obligation closure package)**.** *For every accepted* $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ *carrier* $O = (W, A, K, R, D, S, E, H, C, P, N)$*, carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling dependency routing, and Real-seal factorization exhaust the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *obligation surface.*

More precisely, A *admits the displayed witness carrier,* K *compares the accepted rows only through the componentwise* hsame *transports,* W *routes every completion-witness request to the terminal seal* E*,* D *accounts for the dyadic tolerance ledger before the seal is visible,* R, S, D *route through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* P, N *attach the same finite route to the local package and naming certificate. No quotient stream, selected representative, host-limit operator, countable-choice schedule, trichotomy row, detached Real seal, or ambient completeness coordinate is exported.*

Proof. Project the carrier of Definition 3580.1. The displayed rows are exactly the witness carrier, admission row, classifier row, sibling readbacks, Real seal, transport, replay, provenance, and local naming certificate.

Carrier admission is the row A, while classifier stability is the row K transported through H. The replay row C can only read the same finite route, so accepted comparisons cannot introduce a second classifier surface.

The witness-to-seal route is the displayed $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ row W ending at E. The dyadic row D is already part of that route, so every tolerance ledger read is consumed before the terminal seal is available.

The sibling dependencies are the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows, together with the witness chapter of chapter 3579. The Real-seal factorization is therefore the finite route from those sibling rows through W to E, packaged by P and named by N.

Since the carrier has no coordinate for quotient streams, selected host limits, choice, trichotomy, detached Real seals, or ambient completeness, those objects cannot become obligations of the local certificate. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ consumer that reaches a $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ seal while omitting carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling dependency routing, Real-seal factorization, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cauchy-completeness-witness, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole obligation packet $O = (W, A, K, R, D, S, E, H, C, P, N)$, because each sibling contributes only witness structure, regular readback, finite windows, dyadic tolerances, or the terminal seal while this chapter binds the obligation surface itself.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}$ packet is generated from a CauchyCompletenessWitness carrier, carrier admission, classifier stability, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance data, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3580.1 and Definition 3580.2 close the finite obligation surface for CauchyCompletenessWitness carrier admission, classifier stability, witness-to-seal routing, dyadic ledger nonescape, sibling routing, Real-seal factorization, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessWitnessObligation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, detached Real seals, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A checked theorem for the same $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert obligation route would raise the verification axis.

Chapter 3581

A Concrete Naming Certificate for $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$

Following Hardy–Littlewood maximal estimates and the Vitali-cover selection lemma, the $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ packet records the finite maximal-function surface for metric balls, positive radius windows, and local average rows. It sits beside the finite covering ledger of chapter 2166 and the terminal real-name boundary of chapter 38; the packet names one displayed maximal domination route without importing measure completion, countable choice, selected suprema, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3581.1 (Hardy-Littlewood maximal carrier)**.** *A* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, B, R, A, V, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric-source row. It records the carried points, the displayed distance reads, and the finite ball names inspected by the maximal request.*
2. *B is the metric-ball row. Each ball record stores a center, a positive radius witness, the membership probes read by the packet, and the carrier-local inclusion comparisons used downstream.*
3. *R is the positive-radius window row. It stores only finite radius candidates and rational comparison budgets; it has no coordinate for a selected optimizing radius.*
4. *A is the local-average row. For every displayed ball window it records the finite readback route for the corresponding average value and the rational tolerance by which that value is compared.*
5. *V is the finite Vitali-cover ledger, read through* $\mathsf{FiniteVitaliCover}^{\uparrow}$*. It records the selected disjoint subfamily, the overlap budget, and the cover cells used to dominate the requested window.*
6. *D is the domination witness row. It records the finite comparison that bounds each displayed local average by the exported maximal value read through the same radius and cover windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the finite real-name windows reached from A and D.*
8. *H records componentwise* hsame *transport for the metric source, ball windows, radius budgets, local averages, Vitali ledger, domination witness, real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay of the route*

$$\mathsf{FiniteVitaliCover}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist} \longmapsto \mathsf{HardyLittlewoodMaximal}^{\uparrow}.$$

10. *P is the* Pkg *provenance over* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$, $\mathsf{FiniteVitaliCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these eleven rows. Its classifier compares accepted packets only through $X, B, R, A, V, D, E, H, C, P, N$ and has no coordinate for a selected supremum, a countable ball family, a measure-completion object, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HardyLittlewoodMaximalUp.HardyLittlewoodMaximalTasteGate single carrier alignment`

Theorem 3581.2 (Hardy-Littlewood maximal NameCert obligations)**.** *For an accepted* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *carrier*

$$M = (X, B, R, A, V, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, metric-ball windows, positive-radius budgets, local average rows, finite Vitali-cover ledger, domination witness, real seal, transport, replay, provenance, and local naming row;*
2. *positive-radius visibility, requiring every local average request to read a displayed radius witness from R and the matching ball data from B before any average row in A is inspected;*
3. *finite Vitali-cover discipline, requiring domination to pass through the ledger V and its overlap budget before the witness row D is exported;*
4. *real-seal discipline, requiring every numerical output to pass through E after the local average and domination rows have been displayed;*
5. *ledger non-escape, requiring H, C, P, N to preserve exactly the rows X, B, R, A, V, D, E and to export no selected supremum, no countable-choice ball selection, no measure completion, no quotient function equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3581.1. The accepted packet exposes exactly the metric source, ball windows, positive-radius budgets, local average rows, finite Vitali ledger, domination witness, real seal, transport, replay, provenance, and local naming rows.

The radius and average obligations are ordered by the carrier itself. A local average read from A is meaningful only after the matching ball data from B and positive radius witness from R have been displayed, so the packet cannot replace finite windows by an optimizing host radius.

The covering step is the row V. Its $\mathsf{FiniteVitaliCover}^{\uparrow}$ ledger supplies the finite selected subfamily and overlap budget used by D, and the domination witness can only cite those displayed cells.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since there is no carrier coordinate for a selected supremum, countable-choice ball family, measure completion, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, none of those principles appears in the local $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$ surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ packet is generated from a metric source row, metric-ball windows, positive-radius budgets, local average rows, a finite Vitali-cover ledger, a domination witness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardyLittlewoodMaximal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3581.1 and Definition 3581.2 seed the finite maximal-function carrier and obligation surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{HardyLittlewoodMaximal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, measure completion, countable-choice ball selection, selected suprema, quotient function equality, ambient Real$^{\uparrow}$ completeness, Lebesgue differentiation, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius visibility, finite Vitali-cover discipline, real-seal discipline, and ledger non-escape for BEDC.Derived.HardyLittlewoodMaximalUp.

Chapter 3582

A Concrete Naming Certificate for $\mathsf{RationalRootInterval}^{\uparrow}$

Following Sturm and Bishop–Bridges constructive interval methods, the $\mathsf{RationalRootInterval}^{\uparrow}$ packet records a finite rational interval certificate for a displayed polynomial sign change and its bisection route. It sits after the dyadic rational carrier of chapter 291, the located interval carrier of chapter 1709, the bisection route of chapter 1711, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite interval-certificate route used by L10 Real and Completion consumers; it is not a host real equality test, quotient root set, global root selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 3582.1 (RationalRootInterval carrier)**.** *A* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite coefficient row for the inspected rational polynomial, including the displayed degree bound and the rational coefficient reads used by the certificate.*
2. *L and U are the left and right endpoint rows, carried through* $\mathsf{DyadicRatCore}^{\uparrow}$ *or rational endpoint data and then embedded into the located interval surface.*
3. *G is the sign-read ledger, recording the endpoint sign observations and every comparison row used to treat the interval as bracketing a rational root candidate.*
4. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *provenance row, recording the displayed bisection path from the original endpoint pair to the certified subinterval.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for coefficient reads, endpoint reads, sign reads, and bisection-stage observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the dyadic or rational endpoint ledgers, so every exported approximation is sourced by finite observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the coefficient row, endpoint rows, sign-read ledger, bisection provenance, stream windows, and regular rational readback have been replayed.*
8. *H records componentwise* hsame *transport for coefficient data, endpoints, sign reads, bisection provenance, stream windows, regular readback, terminal seal, replay, provenance, and local naming data.*

9. *C records displayed* Cont *replay for consumers that inspect the polynomial row, endpoint interval, sign ledger, bisection route, regular readback, or terminal seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RationalRootInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisecti}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, U, G, B, W, R, S, H, C, P, N$. It has no coordinate for host real equality, quotient root sets, global root selectors, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RationalRootIntervalUp.RationalRootIntervalTasteGate single carrier alignment`

Theorem 3582.2 (RationalRootInterval bisection seal)**.** *For every accepted* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$*-facing interval seal is exported only by the finite route*

$$A, L, U, G \longrightarrow B \longrightarrow W \longrightarrow R \longrightarrow S.$$

The coefficient row fixes the inspected rational polynomial, the dyadic or rational endpoint rows fix the displayed interval, the sign-read ledger records the endpoint comparisons, the bisection provenance supplies the certified narrowing route, the $\mathsf{StreamName}^{\uparrow}$ *windows expose finite observations, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row packages the regular rational readback, and* $\mathsf{Real}^{\uparrow}$ *consumes exactly that seal.*

Proof. Project the accepted carrier of Definition 3582.1. The rows A, L, U, G display the polynomial coefficients, the two endpoint carriers, and the sign-read ledger; there is no separate coordinate for a host-level polynomial root or for equality of ambient reals.

The bisection data are then forced through B. The route records the finite interval refinements and their provenance, so a consumer cannot replace the displayed path by a global root selector or by a quotient root set.

Regular readback starts only after the finite observations have been opened. The windows in W expose coefficient, endpoint, sign, and bisection-stage reads; the row R packages those reads as regular rational data; and the terminal row S is the only $\mathsf{Real}^{\uparrow}$ seal available to the packet.

Finally, H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no host real equality, quotient root set, global root selector, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, none of those principles is claimed by the bisection seal. □

Theorem 3582.3 (RationalRootInterval sign-read stability)**.** *For every accepted* $\mathsf{RationalRootInterval}^{\uparrow}$ *carrier*

$$Q = (A, L, U, G, B, W, R, S, H, C, P, N),$$

endpoint sign observations are transported only through the displayed coefficient row A, endpoint rows L, U, sign-read ledger G, bisection provenance B, and componentwise hsame *row H. The rows C, P, N replay, source, and name that same route. No host root selector, quotient root set, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can alter the sign-read path.*

Proof. Project the carrier of Definition 3582.1. The polynomial data read by a sign observation are fixed by A, and the two endpoint observations are fixed by L and U.

Read the sign ledger through G and the finite bisection provenance through B. These rows record which endpoint comparisons are visible before the interval is narrowed, so the sign route is determined before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Transport is componentwise through H. The replay row C, provenance row P, and local naming row N preserve the same displayed observations. Since the carrier contains no host root selector, quotient root set, or ambient completeness coordinate, those objects cannot change the sign-read route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3582.1 and Definition 3582.2 seed the finite rational root interval route over coefficient rows, dyadic or rational endpoint rows, sign-read ledgers, bisection provenance, StreamName windows, RegSeqRat readback, Real seals, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalRootInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient root sets, global root selectors, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-row exactness, endpoint routing, sign-read stability, bisection provenance, StreamName window exposure, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalRootIntervalUp.

Constructive story: A $\mathsf{RationalRootInterval}^{\uparrow}$ packet is generated from finite polynomial coefficient rows, dyadic or rational endpoint rows, a sign-read ledger, located-interval bisection provenance, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3583

A Concrete Naming Certificate for EnergyBiasedTriggerReliability$^{\uparrow}$

The EnergyBiasedTriggerReliability$^{\uparrow}$ packet realizes the energy-weighted trigger reliability formula of Definition 9462.1593 as finite BEDC NameCert rows. It consumes the six-coordinate codon window of chapter 2234, the boundary-tile row of chapter 3393, the coverage row of chapter 3470, and the trigger-algebra surface of chapter 3478. The packet records weight rows, partition-factor rows, blocker rows, and reliability rows; it is not a stochastic evolution model, an empirical biological likelihood claim, quotient codon equality, or an infinite trigger search.

Definition 3583.1 (Energy-biased trigger reliability carrier)**.** *An* EnergyBiasedTriggerReliability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, B, V, A, G, F, L, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the finite codon-window coordinates used by the trigger counts.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary row supplying the stop/Trp and Ile/Met trigger faces.*
3. *V is the* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row recording which boundary faces are covered by each displayed trigger family.*
4. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *trigger-algebra row carrying support closure, blocker, and threshold data.*
5. *G is the energy-weight row, recording the displayed powers of t and y assigned to finite trigger cardinality and trigger energy.*
6. *F is the partition-factor row, recording the finite $A_k(t, y)$ partition contributions that feed the reliability quotient.*
7. *L is the blocker row, recording the finite inclusion-exclusion blocker expression used in the numerator of the reliability formula.*
8. *R is the reliability row, recording the displayed value of $R(t, y)$ only after G, F, L and the trigger-algebra dependencies have been displayed.*

9. *H records componentwise* hsame *transport for source, boundary, coverage, algebra, weight, partition, blocker, and reliability rows.*
10. *C records displayed* Cont *replay through the same finite trigger-partition route.*
11. *P is the* Pkg *provenance over* EnergyBiasedTriggerReliability$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, *the source vision of Definition 9462.1593,* BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* NameCert$_{\text{EnergyBiasedTriggerReliability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, V, A, G, F, L, R, H, C, P, N$. It has no coordinate for hidden biological semantics, empirical likelihood calibration, stochastic dynamics, quotient codon equality, global support completion, or infinite trigger enumeration.

Lean4 (definition) `BEDC.Derived.EnergyBiasedTriggerReliabilityUp.EnergyBiasedTriggerReliabilityUp`
Lean4 ✓ `BEDC.Derived.EnergyBiasedTriggerReliabilityUp.EnergyBiasedTriggerReliabilityTasteGate`
`single carrier alignment`

Theorem 3583.2 (Energy-biased trigger reliability NameCert obligations)**.** *For every accepted* EnergyBiasedTriggerReliability$^{\uparrow}$ *carrier $E = (W, B, V, A, G, F, L, R, H, C, P, N)$, the local* NameCert$_{\text{EnergyBiasedTriggerRe}}$ *surface consists exactly of carrier admission, codon-window exposure, boundary-tile exposure, six-boundary coverage exposure, trigger-algebra exposure, energy-weight exposure, partition-factor exposure, blocker exposure, reliability-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3583.1. The trigger source rows are exactly W, B, V, A, so a reliability consumer first receives the finite codon window, boundary tile, coverage, and trigger-algebra surfaces.

The quantitative rows are G, F, L, R. The weight row records the powers assigned to trigger cardinality and energy, the partition row records the displayed $A_k(t, y)$ factors, the blocker row records the finite inclusion-exclusion expression, and the reliability row is available only after those rows have been exposed.

The structural rows H, C, P, N transport, replay, source, and name this same packet. Since no hidden semantics, empirical calibration, stochastic dynamics, quotient codon equality, global completion, or infinite enumeration coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 3583.3 (Energy-biased trigger partition route)**.** *Every consumer of an accepted* EnergyBiasedTriggerRelia *packet reads the reliability formula through the finite route*

$$\text{CodonSixBitWindow}^{\uparrow} \longmapsto \text{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \text{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \text{WindowCoverageTriggerA}$$

Thus the exported reliability value is a finite trigger-partition NameCert row over displayed weights, partition factors, blockers, and reliability data. The packet does not export stochastic evolution, empirical likelihood calibration, quotient codon equality, global support completion, or an infinite trigger search.

Proof. Start with the source route. The codon window row W supplies finite coordinates, the boundary row B supplies the boundary triggers, and the coverage row V records the displayed support faces.

The trigger algebra row A then binds those supports to blocker and threshold data. It is the only algebraic route into the energy-biased reliability packet.

Finally the rows G, F, L, R record weights, partition factors, blockers, and the reliability quotient. Transport, replay, provenance, and naming through H, C, P, N preserve this finite route and leave the refused external readings without export rows. □

Falsifiable prediction: Every accepted $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumer that reads $R(t, y)$ must exhibit CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WindowCoverageTriggerAlgebra, energy-weight, partition-factor, blocker, reliability, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra: $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite codon windows, boundary triggers, coverage rows, trigger algebra, energy weights, partition factors, blocker expression, reliability row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}$.

Constructive story: An $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ packet is generated from codon six-bit windows, DNA boundary tiles, six-boundary coverage rows, trigger-algebra rows, energy weights, partition factors, blockers, reliability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3583.1, Definition 3583.2, and Definition 3583.3 seed the finite trigger-partition route for energy-biased reliability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close stochastic evolution, empirical likelihood calibration, quotient codon equality, global support completion, infinite trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window exposure, boundary-tile exposure, coverage exposure, trigger-algebra exposure, energy-weight exactness, partition-factor exactness, blocker exactness, reliability non-escape, transport stability, replay stability, provenance, and local naming.

Chapter 3584

A Concrete Naming Certificate for $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ packet records the theorem-level product route saying that two displayed source series may be multiplied by the Cauchy product once the summand windows, convolution windows, absolute-convergence or majorant tail ledgers, and terminal real-name seal are all visible. It sits over the local criterion of chapter 1586, the convolution surface of chapter 409, the series-product packet of chapter 2214, the fast product route of chapter 368, the majorant discipline of chapter 399, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names the finite L10 Real seal route for Mertens' product theorem; it does not assert a quotient series product, a selected infinite sum, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3584.1 (Mertens Cauchy product theorem carrier)**.** *A* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, K, T, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the finite source-series rows exposing the inspected summand windows for the two input series.*
2. *W is the paired* $\mathsf{CauchyConvolution}^{\uparrow}$ *window row, recording the finite diagonal product cells used at each requested product index.*
3. *K is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *handoff row that binds the convolution windows to the Cauchy-product acceptance test.*
4. *T is either an* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *row for one source series or a finite majorant-tail ledger of the kind consumed by* $\mathsf{WeierstrassMTest}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchySeriesProduct}^{\uparrow}$ *or* $\mathsf{FastSeriesProduct}^{\uparrow}$ *product packet read from A, B, W, K, T.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the accepted product packet.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the product packet and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for the source rows, convolution windows, tail ledger, criterion handoff, product packet, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a product-theorem request through A, B, W, K, T, Q, R, E.*

10. *P is the* Pkg *provenance over* MertensCauchyProductTheorem$^{\uparrow}$, CauchyConvolution$^{\uparrow}$, CauchyProductCriterion$^{\uparrow}$, CauchySeriesProduct$^{\uparrow}$, FastSeriesProduct$^{\uparrow}$, AbsoluteConvergence$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{MertensCauchyProductTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, host multiplication of completed sums, selected limiting sums, countable-choice schedules, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MertensCauchyProductTheoremUp.MertensCauchyProductTheoremTasteGate single carrier alignment`

Theorem 3584.2 (Mertens Cauchy product theorem NameCert obligations)**.** *For every accepted* MertensCauchyProductTheorem$^{\uparrow}$ *carrier $M = (A, B, W, K, T, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{MertensCauchyPro}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the two finite source-series rows, paired convolution windows, tail ledger, criterion handoff, product packet, regular readback, Real seal, transport, replay, provenance, and local naming data;*

2. *convolution-window exactness, requiring W to be read through chapter 409 before any product packet Q is exposed;*

3. *tail-to-criterion handoff, requiring the absolute-convergence row or majorant-tail ledger T to feed the* CauchyProductCriterion$^{\uparrow}$ *row K through chapter 1586;*

4. *product readback routing, requiring Q to be consumed through the* CauchySeriesProduct$^{\uparrow}$ *and* FastSeriesProduct$^{\uparrow}$ *sibling surfaces before the* RegSeqRat$^{\uparrow}$ *row R is read;*

5. *L10 Real seal factorization, requiring every terminal seal read E to pass through $A, B, W, K, T, Q, R, H, C, P, N$.*

Thus a product-theorem consumer receives a finite Mertens certificate only after the source-series rows, CauchyConvolution windows, AbsoluteConvergence or majorant tail ledgers, CauchyProductCriterion handoff, RegSeqRat readback, Real seal, transport, provenance, and local NameCert rows have all been displayed.

Proof. Project the carrier of Definition 3584.1. The accepted packet exposes exactly the two source-series rows, the paired convolution window, the tail ledger, the criterion handoff, the product packet, the regular readback, the terminal Real seal, and the structural support rows.

For the convolution obligation, read W through the sibling convolution certificate. This fixes the finite diagonal product cells before any series-product packet is available.

For the tail-to-criterion obligation, read T either as an absolute-convergence witness or as a finite majorant ledger. In both cases the displayed tail control is consumed by the CauchyProductCriterion$^{\uparrow}$ handoff K, so the criterion route is not a hidden appeal to host completeness.

For the product readback obligation, route Q through the CauchySeriesProduct and FastSeriesProduct surfaces and then through the regular rational readback row R. The terminal seal E is reached only after this readback, and H, C, P, N transport, replay, source, and locally name that same route.

Since the carrier contains no quotient stream equality, host product of completed sums, selected limiting sum, countable-choice schedule, trichotomy row, or ambient completeness coordinate, none of those objects belongs to the local NameCert surface. □

Theorem 3584.3 (Mertens Cauchy product theorem Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *packet factors through the displayed route*

$$A, B \longmapsto W \longmapsto T \longmapsto K \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The route exports the Mertens product theorem as a finite product certificate over the L10 Real seal corridor. It does not export quotient series equality, a selected product sum, countable choice, trichotomy, host real multiplication, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer that asks for the terminal product seal. The only Real-facing coordinate in the carrier is E, and E is stored after the source rows, convolution window, tail ledger, criterion handoff, product packet, and regular readback.

The consumer must therefore pass through A, B, W, T, K, Q, R. These rows display the finite source observations, the CauchyConvolution cells, the absolute-convergence or majorant tail control, the CauchyProductCriterion acceptance row, and the regular rational readback used to expose the product.

The remaining rows H, C, P, N preserve the same route under componentwise transport, replay it for accepted requests, record its package provenance, and name it locally. Because no excluded object is a carrier coordinate, the Real seal cannot bypass the finite product route. □

Theorem 3584.4 (Mertens absolute-convergence product route)**.** *Let* $M = (A, B, W, K, T, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *carrier. If the displayed tail row* T *is read as an* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *witness for one source series, then every downstream product-convergence consumer must read the route through*

$$\mathsf{AbsoluteConvergence}^{\uparrow} \longmapsto \mathsf{CauchyProductCriterion}^{\uparrow} \longmapsto (\mathsf{CauchySeriesProduct}^{\uparrow} \textit{ or } \mathsf{FastSeriesProduct}^{\uparrow}) \longmapsto \mathsf{RegSeqR}$$

before invoking $\mathsf{CauchyProductConvergence}^{\uparrow}$*. The branch therefore exports the Mertens theorem to product-convergence consumers only through the absolute-convergence handoff, the finite product criterion, the product packet, the regular rational readback, and the terminal Real seal.*

Proof. Project the carrier of Definition 3584.1. Under the stated branch hypothesis, the tail row T is not merely an unnamed tail ledger: it is displayed as an $\mathsf{AbsoluteConvergence}^{\uparrow}$ row and is therefore subject to the handoff and Real-seal nonescape routes of Definition 1700.3 and Definition 1700.4.

Feed this displayed tail row into the local route of Definition 3584.3. The route forces the consumer through W, T, K, Q, R, E; hence the absolute-convergence branch reaches the product packet only after the $\mathsf{CauchyProductCriterion}^{\uparrow}$ handoff and reaches the terminal seal only after the $\mathsf{CauchySeriesProduct}^{\uparrow}$ or $\mathsf{FastSeriesProduct}^{\uparrow}$ packet and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Finally compose this finite Mertens route with the downstream fusion theorem Definition 3608.4. A $\mathsf{CauchyProductConvergence}^{\uparrow}$ consumer that consumes the Mertens surface reads the source-window and tail-control stage before any regular rational, dyadic, stream-name, or Real-seal product data. Thus the absolute-convergence branch cannot enter product convergence except through the displayed criterion, product, readback, and Real-seal corridor. □

Falsifiable prediction: Every accepted $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source-series row, CauchyConvolution window, tail ledger,

CauchyProductCriterion handoff, product packet, RegSeqRat readback, transport row, replay row, provenance row, or local naming row refutes the packet.

Independence witness: cauchyconvolution, cauchy-product-criterion, cauchy-series-product, fast-seriesproduct, regseqrat, real: no listed sibling supplies the whole theorem packet $M = (A, B, W, K, T, Q, R, E, H, C$ because each sibling contributes only convolution cells, a criterion handoff, a product packet, readback, or terminal seal while this chapter binds them as one finite Mertens certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ packet is generated from two finite source-series rows, CauchyConvolution windows, AbsoluteConvergence or majorant tail ledgers, a CauchyProductCriterion handoff, a CauchySeriesProduct or FastSeriesProduct packet, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3584.1, Definition 3584.2, and Definition 3584.3 seed the finite product-theorem route over source-series rows, convolution windows, tail ledgers, criterion handoff, product readback, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MertensCauchyProductTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient series equality, selected product sums, countable choice, trichotomy, host real multiplication, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convolution-window exactness, tail-to-criterion handoff, product readback routing, Real-seal factorization, transport, replay, provenance, and local naming for BEDC.Derived.MertensCauchyP

Chapter 3585

A Concrete Naming Certificate for BishopCompletionUniversalComposition$^{\uparrow}$

Following Bishop and Bridges' constructive treatment of completions by regular Cauchy data, the BishopCompletionUniversalComposition$^{\uparrow}$ packet records the finite composition surface for two Bishop completion universal-property extensions. It consumes the boundary route of chapter 1838 and the extension-plus-uniqueness ledger of chapter 397. The shared source rows are the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readback of chapter 301, the DyadicRatCore$^{\uparrow}$ tolerance ledger of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

The packet composes two already displayed extension ledgers. It does not select representatives of quotient streams, choose host limits, use countable choice, or appeal to ambient completeness of Real$^{\uparrow}$.

Definition 3585.1 (Bishop completion universal composition carrier)**.** *A* BishopCompletionUniversalComposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (U, V, E, F, R, S, D, H, T, P, N)$$

with the following displayed rows:

1. *U is the first* CompletionUniversalProperty$^{\uparrow}$ *ledger, restricted to the shared Bishop regular Cauchy completion source;*
2. *V is the second* CompletionUniversalProperty$^{\uparrow}$ *ledger, whose dense-source read begins at the extension surface exported by U;*
3. *E is the first extension row exposed by U through the Bishop completion universal boundary;*
4. *F is the second extension row exposed by V after E has fixed the intermediate completion-facing map;*
5. *R is the shared* RegSeqRat$^{\uparrow}$ *readback row used by both extension ledgers;*
6. *S is the shared* StreamName$^{\uparrow}$ *finite-window row for the source regular Cauchy name;*
7. *D is the shared* DyadicRatCore$^{\uparrow}$ *tolerance ledger used to align the two precision schedules;*
8. *H records the terminal* Real$^{\uparrow}$ *seal reached by the source route and preserved by both extensions;*
9. *T records componentwise* hsame *transport and displayed* Cont *replay for U, V, E, F, R, S, D, H;*

10. *P is the* Pkg *provenance over* BishopCompletionUniversalBoundary$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{BishopCompletionUniversalComposition}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed extension ledgers, shared source rows, transport, replay, provenance, and local naming data. It contains no quotient representative, selected limit, countable-choice schedule, host real equality, or ambient Real$^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.BishopCompletionUniversalCompositionUp.BishopCompletionUniversalCompositionTasteGa`
`single carrier alignment`

Theorem 3585.2 (Bishop completion universal composition obligations)**.** *For every accepted* BishopCompletionUniversalComposition$^{\uparrow}$ *carrier $C = (U, V, E, F, R, S, D, H, T, P, N)$, the local* NameCert$_{\text{BishopComplet}}$ *surface consists exactly of carrier admission, first-extension exactness through U, E, second-extension exactness through V, F, shared-source exactness through S, R, D, H, componentwise transport and replay through T, provenance through P, and local naming through N.*

The two extension reads must share the same StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows before the composed extension row is exported. No local obligation can be discharged by quotient representatives, selected limits, countable choice, host equality of extensions, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3585.1. The visible coordinates are exactly the two universal-property ledgers, their two extension rows, the shared source and seal rows, and the structural rows T, P, N.

Use the Bishop completion universal boundary route for the first extension. It requires the source windows, regular readback, dyadic tolerance ledger, and terminal Real seal to be displayed before the extension row E can be read.

Use the CompletionUniversalProperty obligations for both ledgers. The first ledger exports E from U over the shared source surface; the second ledger exports F from V only after the same source and seal rows have been transported and replayed through T.

The package and local naming rows then record the composed route as a single NameCert surface. Since the carrier has no quotient representative, selected-limit coordinate, choice schedule, host equality row, or ambient-completeness field, none of these can discharge an obligation. □

Theorem 3585.3 (Bishop completion universal composition route)**.** *Let $C = (U, V, E, F, R, S, D, H, T, P, N)$ be an accepted* BishopCompletionUniversalComposition$^{\uparrow}$ *carrier. Every composed completion-universal read factors through the finite route*

$$S, R, D, H \longmapsto U, E \longmapsto T \longmapsto V, F \longmapsto P, N.$$

The first leg is the Bishop completion universal boundary extension route, the second leg is the CompletionUniversalProperty extension-and-uniqueness route over the transported intermediate extension, and the handoff between the legs is only the displayed hsame *and* Cont *replay stored in T.*

Consequently the composed read exports a finite NameCert packet over the shared source rows. It does not export quotient representatives, selected limits, countable choice, host equality of maps, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the shared source rows. The StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ coordinates S, R, D, H are the only source and seal data carried by C.

Apply the Bishop completion universal boundary extension route to U and E. This gives the first extension only after the shared finite windows, regular rational readback, dyadic tolerance ledger, and terminal Real seal have been displayed.

Next apply the CompletionUniversalProperty extension-uniqueness scope to V and F. The intermediate extension is available to the second ledger only through the transported and replayed row T, so uniqueness is read as a displayed ledger comparison rather than as host function equality.

Finally package the composed route with P and N. The carrier inventory has no quotient stream, selected limit, countable-choice schedule, host equality row, or ambient completeness theorem, so the route cannot leave the finite NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3585.1, Definition 3585.2, and Definition 3585.3 seed the finite composition surface for two Bishop completion universal-property extension ledgers over shared StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionUniversalComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, selected limits, countable choice, host equality of extensions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-source exactness, first-extension exactness, second-extension exactness, transport-replay, provenance, and local naming for BEDC.Derived.BishopCompletionUniversalCompositionUp

Constructive story: A $\mathsf{BishopCompletionUniversalComposition}^{\uparrow}$ packet is generated from two CompletionUniversalProperty ledgers, two extension rows, shared StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3586

A Concrete Naming Certificate for FiniteModulusCompactness$^{\uparrow}$

Following Arzela–Ascoli finite-modulus compactness in constructive metric analysis, the FiniteModulusCompactness$^{\uparrow}$ packet records the finite BEDC surface that turns compact metric inspection, finite-net equicontinuity, and compact uniform-continuity modulus rows into one displayed compactness certificate. It sits between the compact source of chapter 129, the finite-net family route of chapter 2362, the modulus extraction route of chapter 2292, the finite epsilon-net interface of chapter 1767, and the downstream extraction surface of chapter 3568. The packet names a finite-modulus compactness carrier; it does not assert open-cover compactness, host function-space compactness, selected subsequences, quotient equality of real names, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 3586.1 (Finite modulus compactness carrier)**.** *A* FiniteModulusCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, E, U, A, W, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row. It exposes only the finite compact-metric inspection data, finite-net coverage rows, metric-distance ledgers, and source* NameCert *provenance consumed by the certificate.*
2. *E is the* FiniteNetEquicontinuity$^{\uparrow}$ *row. It records the finite family windows, shared center rows, radius ledgers, and equicontinuity comparison rows available at the inspected compact-net scale.*
3. *U is the* CompactUniformContinuityModulus$^{\uparrow}$ *row. It displays the pointwise modulus rows, compact-net fold, dyadic tolerance schedule, and public modulus handoff used by the compact source.*
4. *A is the* ArzelaAscoliFiniteModulus$^{\uparrow}$ *handoff row. It records the downstream extraction interface that may consume the finite-modulus compactness certificate after K, E, U have been displayed.*
5. *W is the finite modulus-window schedule. It records the requested tolerances, the selected compact-net radii, the finite family indices inspected at those radii, and the dyadic support row that links E and U.*
6. *H records componentwise* hsame *transport for the compact source, equicontinuity, compact-uniform-modulus, Ascoli handoff, window schedule, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite compactness request through K, E, U, A, W, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$, $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, E, U, A, W, H, C, P, N$. It has no coordinate for open-cover compactness, a host compactness theorem, a selected subsequence, a quotient stream, a host function-space point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusCompactnessUp.FiniteModulusCompactnessTasteGate single carrier alignment`

Theorem 3586.2 (Finite modulus compactness NameCert obligations)**.** *For an accepted* FiniteModulusCompactnes *carrier $F = (K, E, U, A, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$ *surface consists exactly of compact source admission through K, finite-net equicontinuity admission through E, compact uniform-continuity modulus admission through U, Ascoli finite-modulus handoff through A, finite modulus-window scheduling through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every public finite-modulus compactness read must therefore pass through the displayed compact metric, finite-net equicontinuity, compact uniform-continuity modulus, and Ascoli handoff rows before it can export a downstream compactness certificate.

Proof. Project the accepted carrier. The mathematical rows are K, E, U, A, W, and the structural rows are H, C, P, N.

The row W is the only place where requested tolerances, compact-net radii, family indices, and dyadic support are tied together. Rows K, E, U, A supply the four named chapter interfaces read by that schedule, while H, C, P, N preserve transport, replay, provenance, and local naming.

No further coordinate appears in the carrier. Hence the local NameCert obligations are exhausted by exactly the displayed rows, and every public read must traverse those rows before it can be consumed by a compactness or Ascoli-facing packet. □

Theorem 3586.3 (Finite modulus compactness Ascoli handoff)**.** *Every* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteModulusCompactness}^{\uparrow}$ *carrier receives a finite handoff through*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{FiniteNetEquicontinuity}^{\uparrow},\quad \mathsf{CompactUniformContinuityModulus}^{\uparrow},\quad \mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}.$$

The handoff consists of the compact source row K, the finite-net equicontinuity row E, the compact uniform-continuity modulus row U, the downstream Ascoli row A, the window schedule W, and the structural rows H, C, P, N. It contains no open-cover compactness route, no selected subsequence, no host function-space compactness point, no quotient real equality, no countable-choice row, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from the obligation theorem. It identifies $K, E, U, A, W, H, C, P, N$ as the complete public surface of the carrier.

The Ascoli-facing row is A, but A is not read alone. Its finite-modulus input is supplied by the compact source row K, the finite-net equicontinuity row E, the compact uniform-continuity modulus row U, and the shared window schedule W.

The transport and replay rows keep the handoff inside that finite packet. Since the excluded objects do not occur among $K, E, U, A, W, H, C, P, N$, an Ascoli consumer cannot obtain them from the $\mathsf{FiniteModulusCompactness}^{\uparrow}$ certificate. □

Theorem 3586.4 (Finite modulus compactness nonescape)**.** *No accepted* $\mathsf{FiniteModulusCompactness}^{\uparrow}$ *packet exports open-cover compactness, selected subcovers, selected subsequences, host function-space compactness, quotient stream equality, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Its compactness content is exactly the finite row package*

$$K, E, U, A, W, H, C, P, N.$$

Proof. Apply Definition 3586.2. Every public read of the packet factors through the displayed tuple $K, E, U, A, W, H, C, P, N$.

The compact metric input is confined to K, the equicontinuity input to E, the compact uniform-continuity modulus input to U, the Ascoli-facing output to A, and the finite scheduling relation to W. Rows H, C, P, N only transport, replay, package, and name those displayed rows.

None of the excluded principles is a carrier coordinate or a structural row. Exporting any of them would leave the local $\mathsf{FiniteModulusCompactness}^{\uparrow}$ NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteModulusCompactness}^{\uparrow}$ packet is generated from a CompactMetricUp source row, FiniteNetEquicontinuityUp row, CompactUniformContinuityModulusUp row, ArzelaAscoliFiniteModulusUp handoff row, finite modulus-window schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 3586.1, Definition 3586.2, Definition 3586.3, and Definition 3586.4 seed the finite BEDC compactness certificate consumed by downstream Ascoli and compact-open routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected subsequences, host function-space compactness, quotient equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window scheduling, Ascoli handoff, nonescape, and local NameCert registration for BEDC.Derived.FiniteModulusCompactnessUp.

Chapter 3587

A Concrete Naming Certificate for $\mathsf{LocatedCauchyNet}^{\uparrow}$

Following Moore–Smith, Kelley, and constructive net convergence, the $\mathsf{LocatedCauchyNet}^{\uparrow}$ packet records the located directed-tail Cauchy evidence needed by uniform-completion arguments. It sits between the Moore–Smith carrier of chapter 3403, the convergence interface of chapter 1667, the Cauchy-net surface of chapter 589, the filter-base handoff of chapter 1601, the uniform-space entourage rows of chapter 339, and the completion consumer of chapter 1473. The packet supplies finite located-Cauchy windows for the L10 directed-net completion corridor; it does not introduce quotient nets, selected limits, cofinal choice, or host completeness.

Definition 3587.1 (Located Cauchy net carrier)**.** *A* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, M, A, N, U, F, S, W, H, C, P, K)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *row naming finite directed vertices, preorder comparisons, and displayed upper-window witnesses.*
2. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *row exposing net-value windows and the finite tail ledger over D.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *row recording common-tail Cauchy evidence on the same directed surface.*
4. *N is the* $\mathsf{NetConvergence}^{\uparrow}$ *row naming the convergence-facing route that may inspect D, M, A before any Real or filter handoff is read.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer row that reads only the located Cauchy data displayed by D, M, A, N, F, S.*
6. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained from the displayed tail filters of M and the Cauchy windows of A.*
7. *S is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row used to compare Cauchy windows, located neighbourhood requests, and completion-facing refinements.*
8. *W is the located-window ledger, listing the finite neighbourhood requests, positive tolerance names, and displayed tail witnesses used to locate each Cauchy read.*
9. *H records componentwise* hsame *transport for D, M, A, N, U, F, S, W and for the structural rows.*

10. *C records displayed* Cont *replay through the same directed-tail, Cauchy, filter, uniform-space, convergence, and completion route.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *K is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $D, M, A, N, U, F, S, W, H, C, P, K$. It has no coordinate for quotienting nets, selecting a limit, selecting a cofinal subnet, invoking countable choice, or importing ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyNetUp.LocatedCauchyNetTasteGate single carrier alignment`

Theorem 3587.2 (Located Cauchy net NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *carrier*

$$L = (D, M, A, N, U, F, S, W, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$ *surface consists exactly of carrier admission, directed-index exposure, Moore–Smith tail exposure, Cauchy-net common-window exposure, NetConvergence handoff, UniformCompletion consumption, FilterBase handoff, UniformSpace entourage comparison, located-window evidence, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the accepted carrier of Definition 3587.1. The directed and Cauchy-facing rows are D, M, A, W, so every located read first fixes an index window, a Moore–Smith tail, common-tail Cauchy evidence, and a finite located-window witness.

The handoff rows are then N, F, S, U. The convergence route can inspect only the displayed tail evidence, the filter-base route is obtained from the same tails, the uniform-space row supplies the entourage comparisons, and the completion row consumes these data after they have been displayed.

Finally H, C, P, K transport, replay, source, and name the same packet. Since the carrier contains no additional coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3587.3 (Located Cauchy net uniform completion route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedCauchyNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DirectedSet}^{\uparrow},\ \mathsf{MooreSmithNet}^{\uparrow},\ \mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow},\ \mathsf{UniformSpace}^{\uparrow},\ \mathsf{NetConvergence}^{\uparrow} \longmapsto \mathsf{UniformComplet}$$

The route supplies directed tails, located Cauchy windows, filter-base handoff, entourage comparison, and convergence-facing evidence only through the displayed carrier. It exports no quotient net, selected limit, selected cofinal subnet, countable-choice operator, or ambient completeness theorem.

Proof. Start with the directed rows D, M, A, W. The index row supplies finite upper-window witnesses, the Moore–Smith row supplies net values and tail ledgers, the Cauchy-net row supplies common-window evidence, and the located-window row binds each neighbourhood request to a displayed tail witness.

The filter-base and uniform-space rows F, S are read next. They translate the same displayed tails into filter and entourage language without creating a quotient of nets.

The convergence handoff N and completion consumer U then read only this finite route. Transport, replay, provenance, and naming through H, C, P, K preserve the same rows, so the excluded selection, quotient, choice, and ambient-completeness routes have no export coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyNet}^{\uparrow}$ packet is generated from DirectedSet, MooreSmithNet, CauchyNet, NetConvergence, UniformCompletion, FilterBase, UniformSpace, located-window, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3587.1, Definition 3587.2, and Definition 3587.3 close the located directed-tail route used by uniform-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyNet}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedCauchyNetUp.LocatedCauchyNetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, selected cofinal subnets, countable choice, ambient completeness, or a standard bridge.

Upgrade path: Public export or bridge work requires a standard interface for the checked located-net route.

Chapter 3588

A Concrete Naming Certificate for $\mathsf{BishopUniformCauchyNet}^{\uparrow}$

Following Moore–Smith, Kelley, and Bishop–Bridges, the $\mathsf{BishopUniformCauchyNet}^{\uparrow}$ packet records the uniform-Cauchy net corridor used by constructive completion arguments before any quotient net, selected limit, cofinal-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available. It sits after the located Cauchy net surface of chapter 3587, the uniform entourage discipline of chapter 339, the completion consumer of chapter 1473, the filter-base handoff of chapter 1601, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38.

Definition 3588.1 (Bishop uniform Cauchy net carrier)**.** *A* $\mathsf{BishopUniformCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, M, A, E, W, F, S, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row naming finite Moore–Smith indices, preorder comparisons, and displayed upper-window witnesses.*
2. *M is the Moore–Smith net row exposing finite value windows over D.*
3. *A is the* $\mathsf{CauchyNet}^{\uparrow}$ *row recording common-tail Cauchy evidence for the same net.*
4. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage schedule used to compare the displayed tail windows.*
5. *W is the located-window row, listing the requested entourages, tail bounds, and displayed witnesses that the net enters each requested uniform neighbourhood.*
6. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row obtained from the displayed tails of D, M, A, W.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row consumed by sequence-facing completion readers.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from S and the dyadic tolerance ledger.*
9. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row recording dyadic radii, comparison slack, and tolerance names used by the located uniform reads.*
10. *H records componentwise* hsame *transport for the directed, net, Cauchy, entourage, located-window, filter-base, stream, rational-readback, dyadic, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a uniform precision request through the same rows.*
12. *P is the* Pkg *provenance over* $\mathsf{BishopUniformCauchyNet}^{\uparrow}$, $\mathsf{LocatedCauchyNet}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, A, E, W, F, S, R, Q, H, C, P, N$. It has no coordinate for quotient nets, selected limits, cofinal-choice selectors, hidden host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3588.2 (Bishop uniform Cauchy net NameCert obligations)**.** *For every accepted* BishopUniformCauchyNe *carrier $U = (D, M, A, E, W, F, S, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyNet}^{\uparrow}}$ *surface consists exactly of carrier admission, directed-index exposure, uniform entourage schedule exactness, located-window evidence, filter-base handoff,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3588.1. The directed and Cauchy-facing rows are D, M, A, E, W, so every uniform read first fixes an index window, a Moore–Smith value window, common-tail evidence, an entourage request, and the located witness for that request.

The filter and sequence-facing rows are then F, S, R, Q. The filter-base handoff is obtained from the displayed tails, while the stream, regular-rational, and dyadic rows give finite readback data for downstream Real and Completion consumers.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no additional coordinate, quotient nets, selected limits, cofinal-choice selectors, host equality, and ambient completeness cannot be exported by the local certificate. □

Theorem 3588.3 (Bishop uniform Cauchy net completion route)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopUniformCauchyNet}^{\uparrow}$ *carrier must pass through the displayed net row, uniform entourage schedule, located-window evidence, filter-base handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row. The consumer receives no quotient net, selected limit, cofinal-choice selector, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with a completion-facing read. The route exposes D, M, A before the uniform entourage row E and located-window row W can be interpreted.

Read the filter-base and finite-window handoff next. The rows F, S, R, Q repackage the same displayed tails as filter data, stream windows, regular rational readback, and dyadic tolerance evidence without selecting a subnet or a limit.

The Real-facing seal is downstream from this finite route. Transport, replay, provenance, and local naming through H, C, P, N preserve the displayed rows, so the completion consumer cannot bypass them into quotient nets, host equality, selected limits, cofinal choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformCauchyNet}^{\uparrow}$ packet is generated from a directed index row, Moore–Smith net row, Cauchy-net row, uniform entourage schedule, located-window evidence, filter-base handoff, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopUniformCauchyNet}^{\uparrow}}$, seed)

Scope closed: Definition 3588.1, Definition 3588.2, and Definition 3588.3 name the finite uniform-Cauchy net corridor for UniformCompletion, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformCauchyNet}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, cofinal choice, host equality, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Formal work requires carrier admission, entourage exactness, located-window evidence, filter-base handoff, finite readback, Real seal non-escape, transport, replay, provenance, and local naming for the Bishop uniform-Cauchy net route.

Chapter 3589

A Concrete Naming Certificate for $\mathsf{AlternativeCodeReassignment}^{\uparrow}$

The $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ packet realizes the comparative genetic-code reassignment spectrum of Definition 9462.1208 as finite NameCert data. It packages the standard-code classifier, curated alternative table row, reassignment set, module ledger, sparse support-count ledger, transport, replay, provenance, and local naming rows; it does not treat the observed comparative spectrum as a kernel fact about biological mechanism or as an oracle for all translation systems.

The standard-code classifier row and codon carrier are read through chapter 3393, where the finite boundary-tile classifier rows, transport, replay, provenance, and NameCert rows are already displayed. The sparse support-count ledger and compression boundary are consumed through chapter 3558, so the reassignment packet depends only on finite support-count ledgers, transport, replay, provenance, and local NameCert rows. Its concrete sibling lattice also passes through the four-phase alphabet bridge of chapter 2260, the six-bit codon window of chapter 2234, and the support-compression surface of chapter 3558.

Definition 3589.1 (Alternative-code reassignment carrier)**.** *An* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, D, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the standard-code classifier row, naming the displayed Q_6 codon carrier and the standard assignment classifier used as the comparison base.*
2. *T is the alternative-table row, recording one curated non-standard table on the same codon carrier together with its displayed assignment labels.*
3. *D is the reassignment-set row, listing exactly those codons whose row in T differs from the standard classifier row S.*
4. *M is the module ledger over the Stop/Trp, Ile/Met, Phe/Leu, and AGA/AGG local modules of Definition 9462.1209.*
5. *Q is the support-count ledger, recording sparse codon support and multiplicity information as displayed in Definition 9462.1212.*
6. *H records componentwise* hsame *transport for the standard classifier, alternative table, reassignment set, module ledger, support-count ledger, provenance, replay, and local naming rows.*

7. *C records displayed* Cont *replay from the standard classifier and alternative-table comparison to the reassignment set, then to the module ledger and support-count ledger.*
8. *P is the* Pkg *provenance over* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$*, the DNA six-bit boundary tile source rows,* $\mathsf{ReassignmentSupportCompression}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$ *row.*

The carrier contains no biochemical mechanism coordinate, no exhaustive table-family oracle, no unbounded codon-table search, no quotient identifying biological labels with BEDC constructors, and no claim that reassignment frequency is a cause.

Lean4 ✓ `BEDC.Derived.AlternativeCodeReassignmentUp.AlternativeCodeReassignmentTasteGate single carrier alignment`

Theorem 3589.2 (Alternative-code reassignment NameCert obligations)**.** *For an accepted* AlternativeCodeReassignm *carrier $R = (S, T, D, M, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed comparison packet, standard-code classifier exposure through S, alternative-table exposure through T, reassignment-set exactness through D, module-ledger grouping through M, sparse support-count exposure through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 3589.1. The comparison starts from the standard classifier row S and the alternative-table row T, both over the same displayed codon carrier.

The reassignment row D is then forced by finite row comparison. The module ledger M groups those displayed changes by the four local modules, and the support-count row Q records the sparse support and multiplicity data available to downstream consumers.

The structural rows H, C, P, N transport, replay, source, and name exactly this finite packet. No other coordinate exists, so the listed rows exhaust the local NameCert obligation surface. □

Theorem 3589.3 (Alternative-code reassignment spectrum nonescape)**.** *Every consumer of an accepted* $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ *packet factors through the finite route*

$$S, T \longmapsto D \longmapsto M, Q \longmapsto H, C, P, N.$$

It receives a displayed standard-code comparison, alternative-table row, reassignment set, module ledger, sparse support-count ledger, transport, replay, provenance, and local NameCert surface. It does not receive biological causality, exhaustive coverage of possible genetic codes, a mechanism for translation dynamics, an unbounded codon-table oracle, or a quotient from external labels into BEDC kernel facts.

Proof. Use the obligation surface of Definition 3589.2. The only comparison inputs are S and T, and the only reassignment output is the finite row D obtained from them.

The module and support-count reads pass through M, Q. These rows preserve the comparative spectrum as displayed finite data and do not add a mechanistic or exhaustive biological premise.

Finally H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Since the excluded claims have no carrier coordinate, they cannot be exported by an accepted $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ read. □

Falsifiable prediction: An accepted $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ read is refuted if the standard-code comparison, curated alternative table, reassignment set, module ledger, sparse support-count

ledger, and local NameCert rows can no longer be routed through dna-four-phase-alphabet-bridge, codon-six-bit-window, reassignment-support-compression, and dna-window-six-boundary-tile sibling surfaces.

Independence witness: dna-four-phase-alphabet-bridge, codon-six-bit-window, reassignment-support-compression, dna-window-six-boundary-tile: none is bijective with $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ because the siblings respectively expose alphabet phase rows, codon-window coordinates, support-compression counts, or boundary-tile classifier rows, while the reassignment packet jointly binds standard and alternative table comparison, reassignment exactness, module grouping, sparse support accounting, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}$.

Constructive story: An $\mathsf{AlternativeCodeReassignment}^{\uparrow}$ packet is generated from a standard-code classifier, alternative table row, reassignment set, module ledger, sparse support-count ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3589.1, Definition 3589.2, and Definition 3589.3 seed the finite comparative-code reassignment spectrum surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlternativeCodeReassignment}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological causality, exhaustive genetic-code coverage, translation mechanism, unbounded table search, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, standard-code comparison, alternative-table exposure, reassignment-set exactness, module-ledger grouping, sparse support-count exposure, transport preservation, and local naming for BEDC.Derived.AlternativeCodeReassignmentUp.

Chapter 3590

A Concrete Naming Certificate for $\mathsf{UniformReflectionKernel}^{\uparrow}$

Following Bourbaki's uniform structures and Kelley's separated uniform reflection, the $\mathsf{UniformReflectionKernel}^{\uparrow}$ packet records the finite indistinguishability kernel through which a uniform space becomes visible to separated and completion-facing consumers. It sits after the entourage rows of chapter 339, the displayed separated-uniform boundary of chapter 1876, the Hausdorff reflection route of **??**, and the completion-facing routes of chapter 2332 and chapter 2345. The packet names the kernel boundary itself; it is not a quotient representative set, not host equality, not a selected completion limit, not Classical choice, and not ambient Real completeness.

Definition 3590.1 (Uniform reflection kernel carrier)**.** *A* $\mathsf{UniformReflectionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, K_0, S, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, exposing finite entourage membership, refinement, inverse, and composition data before any separated consumer is reached.*
2. *K_0 is the indistinguishability-kernel row over the same entourage surface:*

$$x \equiv_{K_0} y \quad \Longleftrightarrow \quad \textit{every displayed entourage row of } U \textit{ admits } (x, y).$$

3. *S is the separated handoff row, read as either the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *boundary or the* $\mathsf{HausdorffReflection}^{\uparrow}$ *boundary according to the downstream consumer.*
4. *H records componentwise* hsame *transport for the uniform row, kernel row, separated or Hausdorff handoff, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an entourage request through U, then K_0, then the separated or Hausdorff handoff S.*
6. *P is the* Pkg *provenance over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient representatives, host equality, selected completion limits, Classical choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformReflectionKernelUp.UniformReflectionKernelTasteGate single carrier alignment`

Theorem 3590.2 (Uniform reflection separated handoff)**.** *Every separated or completion-facing consumer of an accepted* $\mathsf{UniformReflectionKernel}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto K_0 \longmapsto \mathsf{SeparatedUniformSpace}^{\uparrow}, \mathsf{HausdorffReflection}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow}.$$

The consumer first reads the displayed entourage rows, then the indistinguishability-kernel row, then the separated-uniform or Hausdorff-reflection boundary, and only then any completion-facing row. It cannot bypass the kernel through quotient representatives, host equality, selected limits, Classical choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3590.1. The uniform source is row U, and the kernel row K_0 is attached to the same displayed entourage data.

The separated handoff is row S. By construction, S is available only after U and K_0 have been replayed, so a $\mathsf{SeparatedUniformSpace}^{\uparrow}$ or $\mathsf{HausdorffReflection}^{\uparrow}$ consumer receives the finite kernel boundary rather than a host quotient.

A completion-facing consumer can then read the $\mathsf{UniformCompletion}^{\uparrow}$ route only through that separated or Hausdorff boundary. The transport, replay, provenance, and local NameCert rows preserve this order, while the excluded quotient, representative, selector, choice, host-equality, and ambient-completeness objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3590.1 and Definition 3590.2 seed the finite route from displayed uniform entourages through the indistinguishability kernel into separated, Hausdorff-reflection, and uniform-completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformReflectionKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host equality, selected limits, Classical choice, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exposure, kernel exactness, separated handoff, Hausdorff reflection handoff, completion nonescape, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{UniformReflectionKernel}^{\uparrow}$ packet is generated from UniformSpace entourage rows, an indistinguishability-kernel row, SeparatedUniformSpace or HausdorffReflection handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3591

A Concrete Naming Certificate for $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$

The $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ packet realizes the focused phase-ledger and boundary-transport reading of chapter 9463 as finite NameCert rows. It consumes the horocycle carrier of chapter 3534, the metric-read surface of chapter 3474, the boundary-fixed transport route of chapter 3500, the disk-coordinate carrier of chapter 377, the flow-style route of chapter 195, and the Poincare disk representation principles of section 9463.3 and section 9463.4. The packet is a BEDC-native phase-ledger boundary row; it is not a host hyperbolic surface, not an unlogged observer trajectory, not a global clock, and not a physical cosmology.

Falsifiable prediction: Every accepted $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ read factors through displayed phase-ledger, disk-coordinate, hyperbolic-metric, horocycle, boundary-fixed transport, flow, provenance, replay, and local NameCert rows before it exports a boundary-transport reading.

Independence witness: hyperbolic-horocycle, hyperbolic-metric, poincare-boundary-fixed-transport, unitdisk, dynsystem: the horocycle packet records boundary-tangent incidence without focused phase-ledger provenance; the metric packet records distance readback without ledger phase; the boundary-fixed transport packet records a preserved boundary anchor without horocycle phase accounting; the UnitDisk packet supplies coordinates without phase-ledger replay; and the DynSystem packet records flow-style evolution without hyperbolic boundary tangency. None is bijective to $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$.

Definition 3591.1 (Phase-ledger horocycle carrier)**.** *A* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, D, M, Q, T, Y, H, C, P, N)$$

with the following displayed rows:

1. *F is the focused phase-ledger row, carrying the source* Hist *chain, visible phase readout, and provenance cells of the observer-expansion reading.*
2. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row through which the ledger readout is placed on a finite disk surface.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row over the same displayed disk observations.*
4. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *boundary-tangent row, read only after D and M have been displayed.*

5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row preserving the boundary far-end reading along admissible disk transport.*
6. *Y is the* $\mathsf{DynSystem}^{\uparrow}$ *flow-style ledger row that replays local record expansion without introducing a global clock.*
7. *H records componentwise* hsame *transport for the phase ledger, disk, metric, horocycle, boundary transport, flow, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the focused ledger through disk placement, metric readback, horocycle incidence, boundary-fixed transport, and flow-style expansion.*
9. *P is the* Pkg *provenance over* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$ *row.*

The carrier exports no host disk model, global universe digest, synchronized radial clock, physical observer path, selected boundary representative, quotient ideal-boundary equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PhaseLedgerHorocycleUp.PhaseLedgerHorocycleTasteGate single carrier alignment`

Theorem 3591.2 (Phase-ledger horocycle NameCert obligations)**.** *For every accepted* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *carrier $L = (F, D, M, Q, T, Y, H, C, P, N)$, the local NameCert surface consists exactly of focused phase-ledger admission, disk-coordinate placement, hyperbolic-metric readback, horocycle boundary-tangency exposure, boundary-fixed transport, flow-style replay, componentwise transport, displayed continuation, provenance, and local naming.*

Every boundary-transport read from the packet factors through

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicHorocycle}^{\uparrow} \longmapsto \mathsf{PoincareBoundaryFixedTransport}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow}$$

together with the focused phase-ledger row and the structural BHist, hsame, Cont, Pkg, *and* NameCert *rows. The Lean TasteGate follow-up must include 'Nontrivial PhaseLedgerHorocycleUp' and 'Field-Faithful PhaseLedgerHorocycleUp' instances for the same carrier surface.*

Proof. Project the accepted carrier. The focused phase-ledger row F supplies the source history chain and phase readout, while D places that readout on a displayed disk surface.

The metric and horocycle rows are then forced in order. Row M reads the hyperbolic metric over the displayed disk data, and row Q exposes the horocycle boundary-tangency surface only after D and M have been replayed.

Boundary transport and flow-style replay are the last public geometric rows. Row T preserves the boundary far-end reading, and row Y records local expansion as a finite flow-style ledger rather than as a global clock.

The structural rows H, C, P, N preserve and name that route. Since the carrier contains no host disk model, universe digest, synchronized radial clock, physical observer path, selected boundary representative, or quotient ideal-boundary equality, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$, seed)

Scope closed: Definition 3591.1 and Definition 3591.2 seed the focused phase-ledger route through disk coordinates, hyperbolic metric readback, horocycle incidence, boundary-fixed transport, flow-style replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PhaseLedgerHorocycle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperbolic geometry, global clocks, physical cosmology, quotient ideal-boundary equality, selected boundary representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, phase-ledger exactness, disk placement, metric readback, horocycle incidence, boundary-fixed transport, flow replay, TasteGate nontriviality, FieldFaithful coverage, and local naming.

Constructive story: A PhaseLedgerHorocycle$^{\uparrow}$ packet is generated from a focused phase-ledger row, UnitDisk coordinates, HyperbolicMetric readback, HyperbolicHorocycle incidence, PoincareBoundaryFixedTransport, DynSystem flow-style replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3592

A Concrete Naming Certificate for $\mathsf{MooreSpace}^{\uparrow}$

Following R. L. Moore and the standard developable-space formulation, the $\mathsf{MooreSpace}^{\uparrow}$ packet records a topology equipped with a countable development as finite BEDC carrier data. It sits beside the topology carrier of chapter 88, the metric-space distance surface of chapter 57, and the entourage-style neighbour rows of chapter 2334. The point of the packet is not to infer a metric or a cover-selection theorem; it is to expose the neighbourhood classifiers, development rows, star-refinement witnesses, transport, replay, provenance, and local naming rows that make the Moore-space name usable by downstream topology consumers.

Definition 3592.1 (Moore space carrier)**.** *A* $\mathsf{MooreSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, N, D, S, H, C, P, L)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *carrier row, exposing the point carrier, finite open-row classifier, finite-meet row, arbitrary-union interface as recorded BEDC cover data, and membership transport surface.*
2. *N is the neighbourhood classifier row. It records when a displayed open row surrounds a point and when one neighbourhood row refines another around the same point.*
3. *D is the countable development row, storing a finite name for each development level together with the pointwise neighbourhood request that the level is meant to answer.*
4. *S is the star-refinement witness row. For each accepted point-neighbourhood request, it records a development level whose displayed star around the point refines the requested neighbourhood row.*
5. *H records componentwise* hsame *transport for the topology, neighbourhood, development, star-refinement, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a point-neighbourhood request through T, then N, then the development row D, then the star-refinement row S.*
7. *P is the* Pkg *provenance over* $\mathsf{MooreSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a metric selector, paracompact refinement, choice-selected open cover, open-cover compactness theorem, or host topological equality.

Theorem 3592.2 (Moore space NameCert obligations)**.** *For every accepted* $\mathsf{MooreSpace}^{\uparrow}$ *carrier* $M = (T, N, D, S, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, topology exposure, neighbourhood classification, countable-development rows, star-refinement witnesses, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every Moore-space consumer reads neighbourhood information through the finite route

$$\mathsf{Topology}^{\uparrow} \longmapsto N \longmapsto D \longmapsto S \longmapsto \mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}.$$

The route may compare with metric and quasi-uniform neighbours only through the displayed $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *provenance rows; it exports no metrization theorem, no paracompactness theorem, no choice-selected cover, and no open-cover compactness conclusion.*

Proof. Project the accepted packet of Definition 3592.1. The topology row T supplies the open-row carrier and membership surface, while N is the only row that may classify neighbourhoods around a displayed point.

The development data is then forced through D and S. Row D lists the countable development levels, and row S records the local star-refinement witness that answers a point-neighbourhood request.

The structural rows H, C, P, L preserve, replay, cite, and name that route. Since the carrier contains no metric selector, paracompact refinement, choice-selected cover, open-cover compactness theorem, or host equality row, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.MooreSpaceUp.TasteGate.MooreSpaceTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3592.1 and Definition 3592.2 seed the finite developable-topology packet over topology rows, neighbourhood classifiers, countable development rows, star-refinement witnesses, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MooreSpaceUp.TasteGate.MooreSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close metrization, paracompactness, choice-selected covers, open-cover compactness, host topological equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology exposure, neighbourhood classification, development-level indexing, star-refinement exactness, componentwise transport, displayed replay, provenance, and local naming.

Constructive story: A $\mathsf{MooreSpace}^{\uparrow}$ packet is generated from a topology carrier, neighbourhood classifier, countable development rows, star-refinement witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3593

A Concrete Naming Certificate for $\mathsf{VariationMeasure}^{\uparrow}$

Following the finite-partition side of Jordan bounded variation as treated by Rudin and Folland, the $\mathsf{VariationMeasure}^{\uparrow}$ packet records the ledger that a bounded-variation consumer may read before any passage to classical measure theory. It sits over the bounded-variation surface of chapter 398, the Jordan signed-variation split of chapter 2069, the real-name seal of chapter 38, the regular rational stream surface of chapter 301, and the dyadic observation rows of chapter 291. The packet names only displayed finite partition ledgers, finite edge sums, common refinements, and their visible transport rows.

Definition 3593.1 (VariationMeasure finite-partition carrier)**.** *A* $\mathsf{VariationMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, J, \Pi, E, X, D, V, A, R, H, C, P, N)$$

with the following visible coordinates:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet and its finite partition-variation ledger.*
2. *J is the optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed ledger read on the same displayed partition surface.*
3. *Π is a displayed ordered finite partition with endpoint and edge rows.*
4. *E is the finite edge ledger induced by adjacent endpoints of Π.*
5. *X records endpoint* $\mathsf{Real}^{\uparrow}$ *rows or their* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name representatives.*
6. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations used to compare adjacent endpoint values.*
7. *V is the finite dyadic variation-sum ledger on Π.*
8. *A records finite additivity rows for displayed concatenations and subpartition blocks.*
9. *R records common-refinement rows comparing transported partitions, edge ledgers, and finite sums.*
10. *H records componentwise* hsame *transport for the bounded-variation source, Jordan split, partition, endpoint, dyadic, finite-sum, additivity, refinement, package, and naming rows.*
11. *C records displayed* Cont *replay from a bounded-variation request to endpoints, edge observations, finite sums, refinement comparison, and consumer exposure.*

12. *P is the* Pkg *provenance over* $\mathsf{VariationMeasure}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, \Pi, E, X, D, V, A, R, H, C, P, N$. It has no coordinate for a sigma-additive measure, a host supremum over all partitions, quotient function equality, selected limits, ambient Real completeness, hidden partition families, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationMeasureUp.VariationMeasureTasteGate single carrier alignment`

Theorem 3593.2 (VariationMeasure finite partition additivity)**.** *Let $M = (B, J, \Pi, E, X, D, V, A, R, H, C, P, N)$ be an accepted* $\mathsf{VariationMeasure}^{\uparrow}$ *carrier. If Π is displayed as the concatenation of finite subpartition blocks $\Pi_0, \ldots, \Pi_k$ whose shared endpoints are transported by the visible* hsame *and* Cont *rows, then the public variation-measure read on Π is the finite sum of the block reads recorded in A. Each block read uses only the source* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger, the optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed ledger, endpoint rows from* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, and the local* Pkg *and* $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$ *provenance.*

The additivity statement is restricted to the displayed finite partition. It exports no sigma-additive measure, host supremum over all partitions, quotient function equality, selected limit, or ambient Real-completeness principle.

Proof. Project the carrier of Definition 3593.1. The rows available to a consumer are the source bounded-variation packet, the optional Jordan signed ledger, the displayed partition, endpoint names, dyadic edge observations, finite variation sum, additivity rows, refinement rows, and visible provenance.

For each displayed block Π_i, read its endpoint rows through $\mathsf{Real}^{\uparrow}$ or $\mathsf{RegSeqRat}^{\uparrow}$ and its adjacent-edge comparisons through $\mathsf{DyadicRatCore}^{\uparrow}$. The source bounded-variation ledger supplies the finite edge-sum surface, while the Jordan ledger, when present, supplies the signed split on the same finite edge list.

The additivity row A concatenates those finite block sums inside the same partition ledger. Shared endpoints are identified only by the displayed hsame and Cont rows, so no unrecorded endpoint or selected limit is introduced.

Finally apply the no-escape clause built into the carrier. Since every summand is a displayed finite dyadic row and the packet has no coordinate for sigma-additivity, host suprema, quotient function equality, selected limits, or ambient Real completeness, the result is a finite-partition additivity certificate rather than a measure-theoretic extension. □

Theorem 3593.3 (VariationMeasure common-refinement classifier stability)**.** *Let M and M' be accepted* $\mathsf{VariationMeasure}^{\uparrow}$ *packets. Suppose their source* $\mathsf{BoundedVariation}^{\uparrow}$ *rows, optional* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *rows, endpoint* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *names,* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, finite variation-sum ledgers, additivity rows,* hsame *rows,* Cont *rows,* Pkg *provenance, and local* NameCert *rows agree after transport to a displayed common finite refinement. Then the local* $\mathsf{VariationMeasure}^{\uparrow}$ *classifier identifies M and M' on that refinement, and every public finite-partition read factors through the same transported refinement ledger.*

This classifier stability uses no sigma-additive measure, host supremum over all partitions, quotient function equality, selected limit, ambient Real completeness, hidden partition family, or bridge row.

Proof. Start from the displayed common finite refinement. Its rows provide the only comparison surface for the two packets: transported source bounded-variation data, optional Jordan signed ledgers, endpoint names, dyadic observations, finite sums, additivity rows, and provenance.

Use Definition 3593.2 on each packet after transport to the common refinement. Both public reads are finite sums over the same displayed block and edge rows, and the shared endpoints are handled by the same hsame and Cont routes.

Repack these componentwise comparisons into the local classifier data. Because the carrier has no additional coordinate beyond $B, J, \Pi, E, X, D, V, A, R, H, C, P, N$, the common-refinement comparison exhausts the classifier.

The excluded objects cannot distinguish the packets. None is present among the carrier rows, and none is generated by the finite additivity or transport steps. □

Theorem 3593.4 (VariationMeasure bounded-variation dependency route)**.** *Every* $\mathsf{VariationMeasure}^{\uparrow}$ *public read factors through a displayed* $\mathsf{BoundedVariation}^{\uparrow}$ *partition ledger and, when a signed split is requested, through the displayed* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *finite signed ledger. Endpoint values are read only through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, and adjacent-edge variation rows are read only through* $\mathsf{DyadicRatCore}^{\uparrow}$*. The packet exposes those sibling rows as finite additivity and common-refinement data with visible* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *provenance.*

Proof. Project the source row B from Definition 3593.1. By the bounded-variation chapter, the available variation data are finite partition ledgers, endpoint rows, dyadic edge observations, finite sums, refinement rows, and visible provenance.

If the signed row J is present, read it through the Jordan variation decomposition chapter. That row only splits the same displayed edge ledger into positive and negative finite ledgers and reconstructs the displayed endpoint-difference row.

Endpoint rows remain on the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces, while edge comparisons remain on $\mathsf{DyadicRatCore}^{\uparrow}$. The $\mathsf{VariationMeasure}^{\uparrow}$ carrier packages these rows into finite additivity and common-refinement certificates without adding a host measure object. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationMeasure}^{\uparrow}$ packet is generated from a bounded-variation source row, an optional Jordan signed ledger, a displayed finite partition, endpoint real or regular-rational names, dyadic edge observations, finite variation sums, additivity rows, common-refinement transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3593.1, Definition 3593.2, Definition 3593.3, and Definition 3593.4 seed the finite-partition variation-measure packet over $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-additive measure, host supremum over all partitions, quotient function equality, selected limits, ambient Real completeness, hidden partition families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition additivity, common-refinement classifier stability, bounded-variation dependency routing, transport, provenance, and no-escape for BEDC.Derived.VariationMeasureUp.

Chapter 3594

A Concrete Naming Certificate for $\mathsf{CauchyContinuityCalibration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ packet records the finite comparison by which Cauchy-name continuity data is calibrated against the tolerance-only uniform-modulus interface. It sits between the finite transformer surface of chapter 1401, the pointwise Cauchy-modulus schedule of chapter 1486, the explicit uniform-modulus packet of chapter 2228, the stream-name and regular-rational readback layers of chapter 126 and chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite calibration surface, not a host function on reals, quotient stream equality, compactness, or an ambient completeness theorem.

Definition 3594.1 (CauchyContinuityCalibration carrier)**.** *A* $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, U, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{CauchyContinuousMap}^{\uparrow}$ *transformer row that reads source Cauchy-name windows and produces the target Cauchy-name windows visible to the consumer.*
2. *M is the* $\mathsf{CauchyModulusOfContinuity}^{\uparrow}$ *row that sends each requested target tolerance to the finite source-window threshold used by S.*
3. *U is the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *comparison row, read only after the Cauchy tolerance route has displayed the same target tolerance and source radius as finite data.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger containing the source windows inspected by M and the target windows produced by S.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source and target regular rational names exposed on W.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested target tolerance, the source threshold supplied by M, and the comparison tolerance consumed by U.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the calibrated target readback after the stream, regular-rational, and dyadic rows have been replayed.*
8. *H records componentwise* hsame *transport for S, M, U, W, R, D, E and for the support rows.*

9. *C records displayed* Cont *replay from a target tolerance request through M, the source windows W, the transformer row S, the readbacks R, the dyadic ledger D, the uniform comparison U, and the seal row E.*

10. *P is the* Pkg *provenance over* CauchyContinuityCalibration$^{\uparrow}$, CauchyContinuousMap$^{\uparrow}$, CauchyModulusOfContinuity UniformContinuityModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyContinuityCalibration}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying these rows to a common Cauchy-name transformer, Cauchy-modulus schedule, uniform-modulus comparison, stream-window, regular-readback, dyadic-tolerance, and real-seal surface.

Lean4 ✓ `BEDC.Derived.CauchyContinuityCalibrationUp.CauchyContinuityCalibrationTasteGate single carrier alignment`

Theorem 3594.2 (CauchyContinuityCalibration NameCert obligations)**.** *For every accepted* CauchyContinuityCali *carrier $K = (S, M, U, W, R, D, E, H, C, P, N)$, the local* NameCert$_{\text{CauchyContinuityCalibration}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the displayed Cauchy transformer, Cauchy-modulus schedule, uniform-modulus comparison, stream windows, regular-rational readbacks, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *Cauchy-route exactness, requiring each target tolerance to pass through M before any source window in W is consumed by S;*

3. *readback accountability, requiring the source-window and target-window reads to factor through W, R, D before the seal row E or the comparison row U is available;*

4. *uniform-comparison compatibility, requiring U to read only the tolerance and radius already displayed by M and D;*

5. *componentwise transport and replay stability under H and C for the same displayed rows;*

6. *ledger refusal, stating that host real functions, quotient streams, compactness selectors, selected limits, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 3594.1. The projection exposes the transformer row S, the Cauchy-modulus row M, the uniform comparison U, the stream windows W, the readbacks R, the dyadic ledger D, the seal E, and the support rows H, C, P, N.

The Cauchy route starts with a target tolerance. The schedule M turns that request into a source-window threshold, and only then may the transformer row S inspect the windows in W and expose the target readback through R and D.

The uniform comparison is downstream of the same route. Since U has no independent tolerance coordinate in the carrier, it can read only the tolerance and source radius already displayed by M and D.

Transport and replay preserve the row order. The hsame and Cont coordinates move and replay the same finite packet, while P and N only source and name it. The listed rows exhaust the carrier, so no host function, quotient stream, compactness selector, selected limit, or ambient completeness datum can be exported. □

Theorem 3594.3 (Cauchy continuity to uniform modulus bridge)**.** *Every accepted* $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ *carrier* $K = (S, M, U, W, R, D, E, H, C, P, N)$ *routes a target tolerance first through the Cauchy-modulus row* M*, reads the required source windows through* S, W, R, D*, and then compares the induced tolerance-only source-radius row with the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *row* U*. The resulting bridge exports a finite uniform-modulus consumer boundary while retaining the Cauchy-name transformer and real-seal rows as the only source of target readback evidence.*

Proof. Begin with the target tolerance stored in the dyadic ledger D. The row M supplies the source-window threshold demanded by the Cauchy-name route, so the source observations are read through W before S can produce target windows.

Readback then passes through R and returns to D for the tolerance comparison. This isolates the tolerance-only datum induced by the Cauchy route: a requested target tolerance together with the displayed source threshold that was sufficient for the transformer row.

Compare that induced row with U. The uniform-modulus consumer sees the same target tolerance and source radius, but it does not receive the Cauchy-name windows except through the already displayed provenance of M, S, W, R, D.

Finally replay C and transport through H. The seal row E remains downstream of the Cauchy readback, while P, N bind the bridge as a finite NameCert surface. Thus the bridge connects Cauchy-name continuity to tolerance-only uniform-modulus consumers without adding a host function, quotient stream, compactness selector, selected limit, or completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyContinuityCalibration}^{\uparrow}$ packet is generated from a CauchyContinuousMap transformer row, a CauchyModulusOfContinuity schedule, a UniformContinuityModulus comparison row, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3594.1, Definition 3594.2, and Definition 3594.3 seed the finite calibration route from Cauchy-name continuity data to tolerance-only uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContinuityCalibration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real functions, quotient stream equality, compactness, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-route exactness, readback accountability, uniform-comparison compatibility, transport and replay stability, and field faithfulness for BEDC.Derived.CauchyContinuityCalibrationUp.

Chapter 3595

A Concrete Naming Certificate for MetaCICLocalDiamondWitness$^{\uparrow}$

The MetaCICLocalDiamondWitness$^{\uparrow}$ packet realizes the local-peak witness surface of chapter 9378 as finite BEDC confluence data. It sits beside the confluence audit packet of chapter 735, the critical-path carrier of chapter 765, the subject-reduction discharge socket of chapter 719, and the typed reduction trace of chapter 2330. The packet names local peaks and candidate joins; it does not promote the conditional substitution boundary into a closed MetaCIC confluence theorem.

Falsifiable prediction: Every accepted local-diamond witness has peak, left-step, right-step, candidate-join, closedness, transport, continuation, provenance, and naming rows.

Independence witness: metacic-confluence-audit-packet, metacic-critical-path, subject-reduction-discharge-socket, metacic-subject-reduction-trace: the confluence audit packet records the global boundary without a per-peak witness carrier; the critical-path packet records priority and dependency pressure without candidate joins; the subject-reduction discharge socket records typing-side hypotheses without confluence joins; and the subject-reduction trace records reduction evidence without a local-diamond peak certificate. None is bijective to MetaCICLocalDiamondWitness$^{\uparrow}$.

Definition 3595.1 (MetaCIC local-diamond witness carrier)**.** *A* MetaCICLocalDiamondWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, L, R, J, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the peak row, naming the source term and the two outgoing MetaCIC reduction requests that form the local peak.*
2. *L is the left-step row, recording the displayed one-step or parallel-step evidence for the left branch.*
3. *R is the right-step row, recording the displayed one-step or parallel-step evidence for the right branch.*
4. *J is the candidate-join row, listing the proposed common reduct together with the two displayed route requests into it.*
5. *B is the closedness and boundary row, recording whether the witness is closed, normal-source, atom-source, or substitution-boundary conditional according to the confluence audit map.*

6. *H records componentwise* hsame *transport for the peak, branch steps, candidate join, closedness boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the peak through the two branch rows, then the candidate join, then the closedness boundary.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$*, the MetaCIC confluence audit map,* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{TypedExampleReductionTrace}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$ *row.*

The carrier exports no arbitrary closed Church–Rosser theorem, no proof of BetaParallelSubstitutionCompatibility*, no proof of* BetaParallelDiamondClosed*, no subject-reduction theorem, and no host quotient equality of terms.*

Lean4 ✓ `BEDC.Derived.MetaCICLocalDiamondWitnessUp.MetaCICLocalDiamondWitnessTasteGate single carrier alignment`

Theorem 3595.2 (MetaCIC local-diamond witness NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ *carrier $W = (K, L, R, J, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$ *surface consists exactly of peak admission, left-step evidence, right-step evidence, candidate-join evidence, closedness and boundary classification, componentwise transport, displayed continuation replay, provenance, and local naming.*

The Lean TasteGate follow-up must provide Nontrivial *and* FieldFaithful *instances for the same carrier surface, so the witness cannot be collapsed to an empty marker and cannot hide a missing peak, branch, join, boundary, transport, replay, provenance, or naming coordinate.*

Proof. Project the carrier of Definition 3595.1. The peak row K names the source and the two outgoing reduction requests, while L and R record the displayed branch evidence.

The join and boundary rows are then read in order. Row J supplies the candidate common reduct and the route requests into it; row B classifies the witness as closed, normal-source, atom-source, or substitution-boundary conditional.

The structural rows H, C, P, N preserve and name that local peak route. Since the carrier contains no arbitrary closed Church–Rosser theorem, no substitution-compatibility proof, no closed parallel-diamond proof, no subject-reduction theorem, and no quotient equality of terms, the listed rows exhaust the NameCert obligations. □

Theorem 3595.3 (MetaCIC local-diamond candidate-join exactness)**.** *For every accepted* MetaCICLocalDiamondW *carrier $W = (K, L, R, J, B, H, C, P, N)$, every candidate-join row J is read only after the peak row K, left-step row L, right-step row R, and boundary row B have been displayed through the same local-diamond witness packet. The row J exports a candidate common reduct and the two route requests into it, but it exports no closed Church–Rosser theorem, no substitution-compatibility proof, no closed parallel-diamond proof, no subject-reduction theorem, and no quotient equality of terms.*

Proof. Project the carrier of Definition 3595.1. The NameCert obligations theorem Definition 3595.2 lists peak admission, left-step evidence, right-step evidence, candidate-join evidence, and closedness and boundary classification as the visible local-diamond rows.

Reading J therefore occurs inside that listed route. The peak row K fixes the source and outgoing requests; the branch rows L and R fix the displayed left and right steps; and the boundary row B records whether the witness is closed, normal-source, atom-source, or substitution-boundary conditional before the join row is consumed as local evidence.

The structural rows H, C, P, N only transport, replay, record provenance for, and locally name the same packet. Since the carrier has no coordinate for a closed Church–Rosser theorem, substitution compatibility, a closed parallel diamond, subject reduction, or quotient equality, the candidate join cannot export any of those objects. □

Theorem 3595.4 (MetaCIC local-diamond closedness-boundary classification)**.** *For every accepted* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ *carrier* $W = (K, L, R, J, B, H, C, P, N)$*, the boundary row* B *must be displayed before any consumer reads the candidate-join row* J*. The displayed row classifies the witness as closed, normal-source, atom-source, or substitution-boundary conditional, and the later join read may only be interpreted inside that classification.*

Consequently a local-diamond witness cannot turn a candidate join into arbitrary closed Church–Rosser, BetaParallelSubstitutionCompatibility*,* BetaParallelDiamondClosed*, subject reduction, or quotient equality of terms.*

Proof. Project the carrier of Definition 3595.1. The carrier lists B as the closedness and boundary row, while Definition 3595.3 already requires J to be read only after the peak, branch, and boundary rows have been displayed.

Read B before J. Its alternatives are closed, normal-source, atom-source, and substitution-boundary conditional, so the candidate common reduct in J is consumed only under the exposed boundary case.

The structural rows H, C, P, N transport, replay, source, and name the same classification. Since no row contains a closed Church–Rosser theorem, substitution compatibility, a closed parallel diamond, subject reduction, or quotient equality of terms, the candidate join remains a classified local witness rather than a global confluence claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3595.1 and Definition 3595.2 seed the finite local-diamond witness packet over peaks, branch steps, candidate joins, closedness boundaries, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary closed Church–Rosser, BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, subject reduction, quotient equality of terms, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, peak exposure, left-step and right-step evidence, candidate-join exactness, boundary classification, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming.

Constructive story: A $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ packet is generated from a local peak row, left-step row, right-step row, candidate-join row, closedness and boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3596

A Concrete Naming Certificate for $\mathsf{TangentCone}^{\uparrow}$

Following Bouligand and Rockafellar, the $\mathsf{TangentCone}^{\uparrow}$ packet names the finite feasible-direction surface of a convex set at a displayed base point. It sits after the convex carrier of chapter 208, the affine ambient row of chapter 206, the vector-space operations of chapter 47, the metric window route of chapter 57, and the terminal real-name interface of chapter 38. The packet records finite secant windows and scale ledgers for a proposed direction; it does not build a host limit, a quotient tangent space, a differentiability theorem, or a separation theorem.

Definition 3596.1 (TangentCone carrier)**.** *A* $\mathsf{TangentCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, L, V, H, C, P, N)$$

with the following displayed rows:

1. *B is the base-point row, naming the displayed point in the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier together with its* $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *ambient readbacks.*
2. *S is the feasible secant-window row. It lists finite windows in which the point obtained by moving from B along the candidate direction remains inside the convex feasible row at the displayed scale.*
3. *L is the positive scale and radius ledger. It records the inspected positive scale rows, metric radii, comparison budgets, and the refusal of zero-scale or unbounded-host reads.*
4. *V is the direction row, read through the* $\mathsf{VecSpace}^{\uparrow}$ *addition and scalar-action surface and compared only by the displayed metric windows.*
5. *H records componentwise* hsame *transport for the base point, secant windows, scale ledger, direction row, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a feasible-direction request through B, S, L, V before any structural row is consumed.*
7. *P is the* Pkg *provenance over* $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{AffineSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through B, S, L, V, H, C, P, N. It has no coordinate for an ambient host limit, quotient tangent space, differentiability theorem, normal cone, separation theorem, or closed convex analysis theorem outside the displayed finite rows.

Theorem 3596.2 (TangentCone secant-window exposure)**.** *Let $T = (B, S, L, V, H, C, P, N)$ be an accepted* $\mathsf{TangentCone}^{\uparrow}$ *carrier. Every feasible-direction read must first expose the base-point row B, then the secant-window row S, then the positive scale and radius ledger L, and only then the direction row V. Consequently a consumer sees a finite Bouligand-style secant surface, not a host limiting process or a selected tangent representative.*

Proof. Project the carrier of Definition 3596.1. The only rows available before the structural data are B, S, L, V, and the secant-window row is explicitly indexed by the base point exposed in B.

Replay the continuation row C. A feasible-direction request is routed through the displayed base-point read, then through the finite feasible secant windows, and then through the positive scale and radius ledger before the direction row can be inspected.

Transport by H preserves the same route, while P and N package only those rows. Thus no host limit, quotient tangent representative, differentiability theorem, normal cone, or separation theorem can be read from the packet. □

Theorem 3596.3 (TangentCone cone-scaling stability)**.** *For an accepted* $\mathsf{TangentCone}^{\uparrow}$ *carrier $T = (B, S, L, V, H, C, P, N)$, scaling an accepted displayed direction by a positive scale already present in L stays on the same finite tangent-cone surface. The resulting read uses the same base-point row B, the scaled secant windows recorded in S, the positive comparison ledger L, and the vector-space scalar-action row inside V, with transport and replay supplied by H and C.*

Proof. Start with a direction read accepted by the packet and a positive scale row displayed in L. The vector-space row V contains the scalar-action data, while the scale ledger records the positive comparison budget needed to reuse the feasible secant windows.

The secant-window row S is then read with the scaled direction and the same base point B. Convex feasibility is consumed only through the displayed $\mathsf{ConvexSet}^{\uparrow}$ row, and metric comparison is consumed only through the windows and radii already listed in L.

Finally replay C and transport through H. Provenance P and the local name row N certify that the scaled direction remains inside the same finite packet surface, with no appeal to a host closure theorem, quotient tangent space, differentiability result, or separation principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3596.1, Definition 3596.2, and Definition 3596.3 seed the finite convex feasible-direction packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TangentCone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host limit, quotient tangent space, differentiability theorem, separation theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, secant-window exposure, cone-scaling stability, transport, replay, provenance, and local naming for BEDC.Derived.TangentConeUp.

Chapter 3597

A Concrete Naming Certificate for $\mathsf{WeakTopology}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{WeakTopology}^{\uparrow}$ packet records a weak topology only through finite windows of scalar linear-functional observations. It sits over the source carrier of chapter 1950, the functional-analysis surface of chapter 94, the continuous-map rows of chapter 130, the open-neighbourhood classifier of chapter 88, and the scalar seal of chapter 38. The chapter names the finite observation-window certificate used by weak-convergence consumers; it does not choose dual elements, assert Banach–Alaoglu, compactness, quotient function spaces, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3597.1 (WeakTopology carrier)**.** *A* $\mathsf{WeakTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (V, F, U, T, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the source vector or normed-space carrier, read through the* $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *sibling rows.*
2. *F is the finite family of linear-functional observation rows. Each row names a scalar-valued functional already admitted by the functional-analysis surface, not a selected element of a completed dual space.*
3. *U is the weak-neighbourhood classifier row. It records finite scalar constraints around a displayed source point and sends them to the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood surface.*
4. *T is the finite test-window row. It lists the scalar probes, tolerances, source points, and comparison budgets inspected by a consumer.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for the observed functional values and their finite tolerance comparisons.*
6. *H records componentwise* hsame *transport for the source carrier, functional rows, neighbourhood classifier, test windows, scalar readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source reads through finite functional observations, from observations to weak-neighbourhood rows, and from those rows to scalar readbacks.*
8. *P is the* Pkg *provenance over* $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, F, U, T, R, H, C, P, N$. It has no coordinate for an unrestricted dual object, a selected family of all functionals, compactness, quotient function spaces, choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyTasteGate single carrier alignment`

Theorem 3597.2 (WeakTopology NameCert obligations)**.** *For every accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier $W = (V, F, U, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the source vector or normed-space row V, finite functional-observation family F, weak-neighbourhood classifier U, finite test windows T, scalar readback row R, componentwise transport H, displayed replay C, provenance P, and local naming N. Every weak-neighbourhood read must cite the sibling routes named above before it reaches a public topology row.*

Proof. Project the carrier of Definition 3597.1. The only source-facing coordinate is V, the only functional-facing coordinate is the finite row F, and the only topology-facing coordinate is the neighbourhood classifier U.

The test-window row T records which finitely many probes and tolerances are inspected. Scalar comparison then passes through R, so the public read receives finite $\mathsf{Real}^{\uparrow}$ observations rather than a selected dual object.

Finally transport, replay, provenance, and naming are supplied by H, C, P, N. Since the carrier contains no further coordinates, the listed rows exhaust the local certificate obligations. □

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyNamecertObligations`

Theorem 3597.3 (WeakTopology functional-window stability)**.** *Let $W = (V, F, U, T, R, H, C, P, N)$ be an accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier. Replacing a test request by a finite subwindow already recorded in T preserves the same source carrier V, the same displayed functional family F, the same weak-neighbourhood classifier U, and the same scalar readback route R. The transport and replay rows H, C carry the subwindow through the same provenance and local naming rows P, N.*

Proof. Read the finite test-window clause of Definition 3597.1. A subwindow of T still names only scalar probes from F, source-point reads from V, and finite tolerance comparisons.

Replay through C. The subwindow reaches the weak-neighbourhood classifier U and scalar row R along the same displayed path, so it cannot add a new functional, a hidden dual element, or a host function-space quotient.

Transport through H and repack through P, N. The packet therefore preserves the weak-topology observation surface while narrowing the inspected finite window. □

Lean4 ✓ `BEDC.Derived.WeakTopologyUp.WeakTopologyFunctionalWindowStability`

Theorem 3597.4 (WeakTopology scalar observation nonescape)**.** *Every scalar observation exported by an accepted* $\mathsf{WeakTopology}^{\uparrow}$ *carrier factors through*

$$V, F, T, U, R, H, C, P, N.$$

Thus weak-convergence, neighbourhood, or continuity consumers may inspect only finite functional windows and their $\mathsf{Real}^{\uparrow}$ *scalar comparisons. They receive no Banach–Alaoglu compactness claim, no selected dual element, no quotient function space, no countable-choice selector, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with an exported scalar observation. By Definition 3597.2, the observation must first expose the source row V, the finite functional family F, the test windows T, and the weak-neighbourhood classifier U.

The scalar row R is then the only route to $\mathsf{Real}^{\uparrow}$ readback. It records finite observed values and tolerance comparisons, not a completed dual-space object or a host function-space equality.

The structural rows H, C, P, N preserve and name that same route. Since the excluded analytic principles have no carrier coordinate, they cannot be exported by the accepted packet. □

Falsifiable prediction: Every $\mathsf{WeakTopology}^{\uparrow}$ neighbourhood or convergence read factors through a source vector or normed-space carrier, a finite family of scalar functional-observation rows, a weak-neighbourhood classifier, finite test windows, scalar Real readbacks, transport, continuation, provenance, and local naming.

Independence witness: normedspace, functionalanalysis, continuousmap, topology, real: $\mathsf{WeakTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source carrier rows, finite linear-functional observations, weak-neighbourhood classifiers, finite scalar test windows, RealUp readbacks, transport, replay, provenance, and local naming without selecting a completed dual or importing compactness.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{WeakTopology}^{\uparrow}$ packet is generated from a source vector or normed-space row, finite scalar linear-functional observation rows, a weak-neighbourhood classifier, finite test windows, scalar $\mathsf{Real}^{\uparrow}$ readbacks, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3597.1, Definition 3597.2, Definition 3597.3, and Definition 3597.4 seed the finite weak-topology observation-window certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Banach–Alaoglu, compactness, quotient function spaces, selected dual elements, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite functional-window stability, scalar observation nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeakTopologyUp.

Chapter 3598

A Concrete Naming Certificate for $\mathsf{CondensationTailSelector}^{\uparrow}$

Following Cauchy's condensation test and the Bishop–Bridges constructive series criterion, the $\mathsf{CondensationTailSelector}^{\uparrow}$ packet records the finite tail-selector route used by constructive series-test consumers before a regular rational handoff or real seal is read. It sits after the dyadic block comparison surface of chapter 1478, the alternating route of chapter 1483, the Dirichlet route of chapter 1497, the summable-tail modulus of chapter 1799, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names a displayed selector for finite tail windows; it is not an infinite tail choice, a selected sum, a quotient series equality, a host real equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3598.1 (Condensation tail selector carrier)**.** *A* $\mathsf{CondensationTailSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, A, D, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CauchyCondensationTest}^{\uparrow}$ *dyadic-block row, exposing the monotone nonnegative source tail, block-index ledger, condensed block terms, and finite comparison budget.*
2. *A is the* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *face row, exposing the alternating sign ledger, monotone magnitude row, and finite tail-bound schedule when the downstream consumer uses the alternating route.*
3. *D is the* $\mathsf{DirichletTest}^{\uparrow}$ *face row, exposing bounded partial sums, coefficient windows, null-tail data, and the finite Dirichlet tail estimate schedule.*
4. *M is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row, naming the displayed modulus request that bounds the inspected tail window without choosing an infinite tail.*
5. *S is the selector row. It records which of the condensation, alternating, or Dirichlet faces supplies the next finite tail window and which modulus row certifies the requested tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row read only after K, A, D, M, S have displayed the selected finite tail route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff and the selected finite tail route have been replayed.*

8. *H records componentwise* hsame *transport for the condensation, alternating, Dirichlet, modulus, selector, handoff, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a series-test request through K, A, D, M, S, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{CondensationTailSelector}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay route. It has no coordinate for selected infinite sums, countable choice, quotient series equality, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CondensationTailSelectorUp.TasteGate.CondensationTailSelectorTasteGate single carrier alignment`

Theorem 3598.2 (Condensation tail selector NameCert obligations)**.** *For every accepted* $\mathsf{CondensationTailSelector}^{\uparrow}$ *carrier $T = (K, A, D, M, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the condensation, alternating, Dirichlet, summable-tail modulus, selector, regular handoff, Real seal, transport, replay, provenance, and local naming rows;*
2. *selector coverage, requiring S to name one of the displayed* $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *faces before a tail window is exported;*
3. *modulus handoff, requiring the selected face to pass through the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row M before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is available;*
4. *non-escape, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to pass through $K, A, D, M, S, R, H, C, P, N$ and excluding selected infinite sums, countable choice, quotient series equality, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3598.1. The rows K, A, D are the only convergence-test faces, and S is the only row allowed to choose among those displayed finite faces for a requested tail window.

The selected face then passes through M. Thus the requested tail window is bounded by a displayed $\mathsf{SummableTailModulus}^{\uparrow}$ row before the $\mathsf{RegSeqRat}^{\uparrow}$ handoff R can be read.

Finally E is downstream of R, while H, C, P, N transport, replay, source, and locally name the same finite route. Since the carrier contains no coordinate for selected infinite sums, countable choice, quotient series equality, host real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3598.3 (Condensation tail selector test handoff)**.** *Every series-test consumer of an accepted* $\mathsf{CondensationTailSelector}^{\uparrow}$ *packet factors its tail read through the finite route*

$$K, A, D \longmapsto S \longmapsto M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer may cite the condensation, alternating, or Dirichlet test surface through one displayed selector row and one displayed modulus row, but it receives no selected infinite sum, countable-choice operator, quotient series equality, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer asking for a tail window usable by a Real-facing readback. The only available test data are the three displayed faces K, A, D, and the only row that records the chosen face is S.

The route then reads M before R. This forces the tail request to become a finite modulus-bound request rather than a hidden choice of an unbounded tail.

The seal row E is reached only after this handoff, and H, C, P, N preserve the same route under transport, replay it as displayed continuation data, record provenance, and name it locally. The excluded objects are not coordinates of the packet, so they cannot be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{CondensationTailSelector}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ handoff or $\mathsf{Real}^{\uparrow}$ seal while omitting the selected convergence-test face, the selector row, the summable-tail modulus row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cauchy-condensation-test, alternating-series-test, dirichlet-test, summable-tail-modulus, regseqrat, real: no listed sibling supplies the whole selector packet $T = (K, A, D, M, S, R, E, H, C, P, N$ because each sibling contributes a test face, a modulus row, a regular handoff, or a terminal seal while this chapter binds them as one finite tail-selector certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CondensationTailSelector}^{\uparrow}$ packet is generated from CauchyCondensationTest, AlternatingSeriesTest, DirichletTest, SummableTailModulus, RegSeqRat, and RealUp rows, together with componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3598.1, Definition 3598.2, and Definition 3598.3 seed the finite tail-selector route over condensation, alternating, Dirichlet, summable-tail modulus, regular handoff, Real seal, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CondensationTailSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite sums, countable choice, quotient series equality, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector coverage, modulus handoff, test handoff, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CondensationTailSelectorUp.

Chapter 3599

A Concrete Naming Certificate for NestedIntervalCauchy$^{\uparrow}$

Following Bishop and Bridges on nested interval and Cauchy real constructions, the NestedIntervalCauchy$^{\uparrow}$ packet records the finite route by which a displayed nested interval chain becomes a regular rational Cauchy name before the terminal real seal is available. It consumes the dyadic endpoint and comparison rows of chapter 291, the finite observation schedules of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name boundary of chapter 38, and the dyadic finite-tail handoff of chapter 1811. The packet is a textbook-grounded L10 completion route for interval evidence; it is not a completed intersection point, quotient real, host real equality, countable-choice selector, or ambient completeness principle.

Definition 3599.1 (Nested interval Cauchy carrier)**.** *A* NestedIntervalCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested interval window. It records the inspected interval prefix, endpoint rows, containment rows, and nonempty-window evidence visible to the nested-interval consumer.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *endpoint and comparison ledger used to read endpoint order, diameter, and tolerance comparisons at the requested dyadic scales.*
3. *W is the shrinking-width witness row. It assigns each displayed precision request to a prefix index whose interval diameter is bounded by the requested tolerance.*
4. *S is the* StreamName$^{\uparrow}$ *schedule row. It names exactly the interval-prefix entries, endpoint observations, and diameter comparisons that may be opened before regular rational readback.*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row formed from the scheduled endpoint or midpoint observations and the shrinking-width witness.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after I, D, W, S, R have been displayed.*
7. *H records componentwise* hsame *transport for interval prefixes, dyadic ledgers, shrinking-width witnesses, stream schedules, regular rational readback, real seals, replay rows, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite nested-interval request through D, W, S, R, E and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalCauchy}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NestedDyadicCauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for a completed intersection point, quotient real equality, host real equality, countable-choice selector, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.NestedIntervalCauchyUp.NestedIntervalCauchyTasteGate single carrier alignment`

Theorem 3599.2 (Nested interval Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *carrier*

$$K = (I, D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval prefix, dyadic endpoint ledger, shrinking-width witness, stream schedule, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*
2. *prefix exposure, requiring each interval read to be one of the displayed rows of I;*
3. *endpoint and diameter exactness, requiring every endpoint-order and diameter comparison to be read through D;*
4. *precision discharge, requiring every Cauchy tolerance request to be answered by a prefix index and comparison stored in W;*
5. *schedule compatibility, requiring S to expose only the interval-prefix, endpoint, and diameter observations used by W and R;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be formed only after the prefix, dyadic, width, and schedule rows have been projected;*
7. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to be consumed only after R and the structural rows have fixed the finite route.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3599.1. The projection exposes one finite nested interval prefix, one dyadic endpoint ledger, one shrinking-width witness, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Prefix exposure is forced by I. A consumer may inspect only interval rows already present in the finite prefix, and each such row carries the endpoint observations that the downstream dyadic ledger may compare.

Endpoint and diameter exactness are forced by D. Every order or diameter comparison used in the Cauchy route is a displayed dyadic comparison, so no external host interval relation is introduced.

Precision discharge is then forced by W. Each tolerance request names a displayed prefix index and a displayed comparison certifying that the selected interval is small enough for the requested regular rational read.

The schedule and handoff rows are downstream. The row S records the finite prefix, endpoint, and diameter observations that may be opened, and R repacks exactly those observations as regular rational Cauchy data.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection point, quotient real equality, host real equality, countable-choice selector, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 3599.3 (Nested interval Cauchy RegSeqRat handoff)**.** *Let $K = (I, D, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of K factors through the finite route*

$$I \longmapsto D, W \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite nested interval prefix, displayed dyadic endpoint and diameter comparisons, a shrinking-width witness, and scheduled endpoint observations before it exports the regular rational Cauchy row.

Proof. Start with the displayed interval prefix I. Its rows give the only endpoint data available to a regular rational consumer, and the containment rows keep later prefix rows inside earlier rows.

Use the dyadic ledger D and shrinking-width witness W to answer a displayed Cauchy tolerance request. The selected prefix index, endpoint observations, and diameter comparison are part of the same finite packet, so no external modulus or choice of tail is needed.

The schedule row S then fixes which interval endpoints or midpoint observations are opened. The handoff row R is obtained by reading those scheduled observations under the width discipline and repacking them as the regular rational Cauchy surface.

The rows H, C, P, N transport, replay, source, and name that handoff. Since the carrier has no additional tail selector, quotient-real coordinate, or completed intersection coordinate, every RegSeqRat read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedIntervalCauchy}^{\uparrow}$ packet is generated from finite nested interval windows, dyadic endpoint and diameter ledgers, shrinking-width witnesses, StreamName schedules, RegSeqRat handoffs, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3599.1, Definition 3599.2, and Definition 3599.3 seed the finite nested interval Cauchy route from interval-prefix evidence to RegSeqRat readback and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed intersections, quotient real equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix exposure, endpoint and diameter exactness, precision discharge, schedule compatibility, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.NestedIntervalCauchyUp.

Chapter 3600

A Concrete Naming Certificate for $\mathsf{TypeLike}^{\uparrow}$

Following Martin-Lof type theory, categorical families, and displayed fibrations, the $\mathsf{TypeLike}^{\uparrow}$ packet records the carrier-plus-classifier-plus-transport interface used when BEDC data are read as typed families rather than as an unindexed host universe. It sits beside the membership discipline of chapter 3668, the syntactic carrier of chapter 197, and the satisfaction carrier of chapter 198. The packet names only displayed classifier families, fiber rows, classifier transport, componentwise history transport, continuation replay, package provenance, and local naming; it does not introduce a universe hierarchy, quotient type equality, impredicative comprehension, host isomorphism of types, or an external category of families.

Definition 3600.1 (TypeLike carrier)**.** *A* $\mathsf{TypeLike}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the displayed base classifier family row, read as BEDC classifier data rather than as a host universe;*
2. *F is the fiber-carrier row assigning to each accepted base entry a finite displayed fiber packet;*
3. *Q is the classifier row for fiber membership, endpoint equality, and family-compatible reads;*
4. *E is the extensionality and endpoint-exactness ledger, recording which displayed fiber endpoints may be identified under the classifier;*
5. *H records componentwise* hsame *transport for B, F, Q, E together with replay, provenance, and local naming rows;*
6. *C records displayed* Cont *replay through which a consumer reads a base classifier, a fiber carrier, a classifier row, and an endpoint-exactness ledger;*
7. *P is the* Pkg *provenance over* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{ModelTheory}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient type universe, quotient identity, selected representative, host-level equivalence of categories, or a comprehension rule not already displayed by B, F, Q, E, H, C, P, N.

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeTasteGate single carrier alignment`

Theorem 3600.2 (TypeLike NameCert obligations)**.** *For every accepted* $\mathsf{TypeLike}^{\uparrow}$ *carrier*

$$T = (B, F, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *surface consists exactly of carrier admission for* T*, base-family classifier admission through* B*, fiber-carrier admission through* F*, classifier exposure through* Q*, endpoint-exactness exposure through* E*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 3600.1. Its base side is the displayed classifier family B, while its fiber side is the finite carrier row F indexed by accepted base entries.

The classifier row Q supplies the only family-compatible membership and endpoint reads. The exactness row E records when two displayed endpoints may be identified by that classifier, so endpoint equality is a carried ledger row rather than host type equality.

The structural rows H, C, P, N preserve and expose the same packet. Since no further coordinate is present, the NameCert obligations cannot ask for an ambient universe, quotient identity, selected representative, categorical equivalence, or external comprehension rule. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeNameCert obligations`
↪ variant `BEDC.Derived.TypeLikeUp.TypeLikeNameCertObligations`

Theorem 3600.3 (TypeLike classifier transport)**.** *Every downstream consumer of an accepted* $\mathsf{TypeLike}^{\uparrow}$ *packet that transports a fiber endpoint must factor through the finite route*

$$B, F, Q, E \longmapsto H, C, P, N.$$

Consequently the transported endpoint remains inside the displayed classifier family and cannot be replaced by host equality of types, an untracked equivalence of fibers, or a SetLike, FirstOrder, or ModelTheory read that bypasses the carried classifier row.

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeClassifierTransport`

Proof. Begin with an accepted carrier $T = (B, F, Q, E, H, C, P, N)$. A consumer can reach a fiber endpoint only after reading the base classifier B and the indexed fiber carrier F.

The classifier row Q fixes the admissible endpoint comparison, and E records the exact endpoint ledger for that comparison. Transport therefore has to preserve both the base-family row and the fiber classifier row.

The rows H, C, P, N then move, replay, source, and name the same finite route. Reads inherited from $\mathsf{SetLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, or $\mathsf{ModelTheory}^{\uparrow}$ may be cited only through the package provenance P and the displayed continuation C, so no sibling can supply a bypass around B, F, Q, E. □

Theorem 3600.4 (TypeLike displayed-family consumer exhaustion)**.** *Every downstream* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{TypeLike}^{\uparrow}$ *endpoint factors through the displayed base classifier* B*, fiber carrier* F*, classifier row* Q*, endpoint-exactness ledger* E*, componentwise transport* H*, displayed replay* C*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$ *row* N*. No such consumer may read the endpoint as unindexed membership, bare syntax, or satisfaction data before the displayed family route has been exposed.*

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeDisplayedFamilyConsumerExhaustion`

Proof. Begin with the accepted carrier $T = (B, F, Q, E, H, C, P, N)$ of Definition 3600.1. Its first four rows are the only displayed family data: base classifier, indexed fiber carrier, family-compatible classifier, and endpoint-exactness ledger.

Apply Definition 3600.3. Any endpoint transport used by a sibling consumer must preserve those four rows before the structural rows are available.

Finally inspect the sibling reads. $\mathsf{SetLike}^{\uparrow}$ can supply membership only through Q, $\mathsf{FirstOrder}^{\uparrow}$ can supply syntax only through the provenance row P and replay row C, and $\mathsf{ModelTheory}^{\uparrow}$ can supply satisfaction only after the same displayed classifier route has been named locally by N. Thus the three consumer families are exhausted by B, F, Q, E, H, C, P, N and cannot add an ambient universe, quotient type equality, or external category of families. □

Theorem 3600.5 (TypeLike SetLike FirstOrder ModelTheory carrier boundary)**.** *Every accepted* $\mathsf{TypeLike}^{\uparrow}$ *endpoint consumed by chapter 3668, chapter 197, or chapter 198 must expose the finite displayed-family route*

$$B, F, Q, E \longmapsto H, C, P, N$$

before the sibling read. The consumer may then interpret the exposed row as membership, syntax, or satisfaction-facing data, but it cannot replace the TypeLike carrier by unindexed SetLike membership, bare FirstOrder syntax, external ModelTheory satisfaction, quotient type equality, or an ambient host family.

Proof. Start with classifier transport, Definition 3600.3. It forces endpoint movement to preserve the displayed base classifier, fiber carrier, classifier row, and endpoint-exactness ledger.

Then apply consumer exhaustion, Definition 3600.4. The SetLike, FirstOrder, and ModelTheory consumers enter only after the displayed route has reached the transport, replay, provenance, and local naming rows.

Thus the sibling chapters can consume the endpoint only through B, F, Q, E, H, C, P, N. Since the carrier contains no unindexed membership source, bare syntax row, external satisfaction predicate, quotient equality, or ambient family coordinate, none of those can serve as a replacement boundary. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeSetLikeFirstOrderModelTheoryCarrierBoundary`

Theorem 3600.6 (TypeLike displayed-family inversion)**.** *Let* $T = (B, F, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{TypeLike}^{\uparrow}$ *carrier. Any endpoint read classified as* $\mathsf{TypeLike}^{\uparrow}$ *data determines the displayed base classifier row* B*, the fiber-carrier row* F*, the classifier row* Q*, and the endpoint-exactness ledger* E *before any structural row* H, C, P, N *is used. Thus endpoint readback is inverted through the concrete displayed-family spine*

$$B, F, Q, E \longmapsto H, C, P, N$$

and cannot be read first as host type equality, unindexed $\mathsf{SetLike}^{\uparrow}$ *membership, bare* $\mathsf{FirstOrder}^{\uparrow}$ *syntax, or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction.*

Proof. Project the carrier of Definition 3600.1. The endpoint-facing data appear in the first four rows: base classifier, indexed fiber carrier, family-compatible classifier, and endpoint-exactness ledger.

Use classifier transport, Definition 3600.3. It keeps any transported endpoint inside the displayed classifier family and prevents a downstream consumer from replacing the route by host type equality or an untracked fiber equivalence.

Finally apply consumer exhaustion, Definition 3600.4. The SetLike, FirstOrder, and ModelTheory consumers may enter only after B, F, Q, E have been exposed and carried through H, C, P, N. Hence an accepted endpoint read determines those four displayed rows before the structural rows are consumed. □

Lean4 ✓ `BEDC.Derived.TypeLikeUp.TypeLikeDisplayedFamilyInversion`

Falsifiable prediction: A consumer that claims a $\mathsf{TypeLike}^{\uparrow}$ endpoint transport while omitting the base classifier row, fiber-carrier row, classifier row, endpoint-exactness ledger, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 3600.1.

Independence witness: setlike, firstorder, modeltheory: $\mathsf{TypeLike}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a displayed base classifier family, indexed fiber carrier, classifier row, endpoint-exactness ledger, componentwise transport, continuation replay, provenance, and local NameCert row. SetLikeUp has membership without indexed fiber transport; FirstOrderUp has syntax and deduction rows without the carrier family; ModelTheoryUp has satisfaction rows over FirstOrderUp without a general classifier-family transport interface.

Closure status for $\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeLike}^{\uparrow}$ packet is generated from a displayed base classifier family, indexed fiber carrier rows, a family-compatible classifier row, endpoint-exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3600.1, Definition 3600.2, Definition 3600.3, and Definition 3600.4 bind the displayed carrier-plus-classifier-plus-transport scope and exhaust the SetLikeUp, FirstOrderUp, and ModelTheoryUp sibling consumers over BHist, hsame, Cont, Pkg, NameCert, and the local family rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TypeLike}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universe hierarchy, quotient type equality, impredicative comprehension, host isomorphism of types, Lean verification, bridge checking, or an external category of families.

Upgrade path: No theory-axis upgrade remains for this displayed classifier-family transport surface; Lean verification and bridge status are recorded separately.

Chapter 3601

A Concrete Naming Certificate for $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$

Following Bishop and Bridges on constructive apartness and completion, the $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ packet records a displayed apartness space whose Cauchy data are complete by finite tolerance inspection. It sits after the located completion criterion of chapter 3540, the apartness completion boundary of chapter 1787, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion object for L10 Real and Completion consumers; it is not a quotient completion, a selected limit, a trichotomy principle, a countable-choice schedule, or a host real equality test.

Definition 3601.1 (Cauchy-complete apartness space carrier)**.** *A* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, G, T, S, R, D, E, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed apartness-space carrier, recording the points, positive separation evidence, and local apartness classifier inspected by the packet.*
2. *G is the positive-gap row, carrying the finite apartness witnesses used to separate accepted Cauchy windows.*
3. *T is the Cauchy tolerance route, recording the requested dyadic budgets and the finite window discipline that makes the Cauchy request inspectable.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for the Cauchy name.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including radius, endpoint, and comparison rows used by the Cauchy and apartness checks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, R, D have been exposed.*
8. *L is the* $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ *row, recording the finite located-completion admission route for the same Cauchy data.*
9. *Q is the* $\mathsf{ApartnessCompletion}^{\uparrow}$ *handoff row, recording the separated apartness-completion boundary read by completion-facing consumers.*

10. *H records componentwise* hsame *transport for the apartness source, positive gap, Cauchy tolerance route, stream window, regular readback, dyadic ledger, Real seal, located-completion criterion, apartness-completion handoff, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a Cauchy-complete apartness-space request through $X, G, T, S, R, D, E, L, Q$*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$, $\mathsf{LocatedCompletionCriterion}^{\uparrow}$, $\mathsf{ApartnessCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, G, T, S, R, D, E, L, Q, H, C, P, N$. It has no coordinate for quotient Cauchy classes, chosen representatives, selected limits, countable choice, trichotomy, host real equality, or a host-level complete space.

Lean4 ✓ `BEDC.Derived.CauchyCompleteApartnessSpaceUp.CauchyCompleteApartnessSpaceTasteGate`
`single carrier alignment`

Theorem 3601.2 (Cauchy-complete apartness space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *carrier $A = (X, G, T, S, R, D, E, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyComplete}}$ *surface consists exactly of carrier admission for the apartness source, positive-gap witnesses, Cauchy tolerance route, finite stream windows, regular rational readback, dyadic tolerance ledger, terminal Real seal, located-completion criterion, apartness-completion handoff, componentwise transport, displayed continuation replay, package provenance, and local naming rows.*

Every completion-facing consumer must read the finite route

$$X, G, T \longmapsto S, R, D \longmapsto E \longmapsto L, Q \longmapsto H, C, P, N$$

before treating the space as Cauchy complete for displayed apartness data.

Proof. Project the carrier of Definition 3601.1. The rows X, G, T expose the apartness source, positive separation witnesses, and the Cauchy tolerance route before any Real-facing or completion-facing row is available.

The finite readback then passes through S, R, D. Stream windows supply the observed name data, regular rational readback repacks those windows, and the dyadic ledger records the tolerance and comparison budgets used by both the Cauchy and apartness checks.

The terminal and completion rows are downstream. Row E seals the Real-facing read, row L records the located completion criterion route of chapter 3540, and row Q records the apartness completion boundary of chapter 1787.

Finally H, C, P, N transport, replay, source, and name only that displayed route. Since the carrier contains no quotient class, chosen representative, selected limit, countable-choice schedule, trichotomy row, host equality test, or host complete-space coordinate, those objects cannot become local NameCert obligations. □

Theorem 3601.3 (Cauchy-complete apartness space located-completion handoff)**.** *Every* $\mathsf{LocatedCompletionCriteri}$ $\mathsf{ApartnessCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ *packet reaches the completion-facing surface only through the displayed apartness carrier X, positive-gap row G, Cauchy tolerance route T, finite* $\mathsf{StreamName}^{\uparrow}$ *window S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, located-completion row L, apartness-completion handoff Q, componentwise transport H, displayed replay C, provenance P, and local name N.*

The handoff exports complete-apartness data only as finite NameCert rows. It does not export quotient completions, selected limits, countable choice, trichotomy, host real equality, or an ambient completeness theorem for all $\mathsf{Real}^{\uparrow}$ *names.*

Proof. Begin with a completion-facing or Real-facing consumer. Projection of the carrier gives the ordered route from the apartness source and positive gap rows through the Cauchy tolerance row, then through the finite stream and dyadic readback rows.

The located-completion handoff is forced through the existing siblings. The row L is read as the located criterion route named in Definition 3540.2, while Q is read as the apartness-completion boundary named in Definition 1787.2. Both are downstream of the same S, R, D, E readback route.

Transport and replay cannot add coordinates. Rows H, C, P, N preserve and name the same finite route, so a consumer receives complete-apartness evidence only through displayed BEDC rows and receives no quotient equality, selected representative, choice schedule, trichotomy decision, host equality test, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ consumer factors a completion-facing read through one apartness source row, one positive-gap row, one Cauchy tolerance route, one $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{Real}^{\uparrow}$ seal, one $\mathsf{LocatedCompletionCriterion}^{\uparrow}$ row, one $\mathsf{ApartnessCompletion}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-completion-criterion, apartness-completion, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries apartness-source evidence, positive gaps, Cauchy tolerance inspection, finite readback, Real sealing, located-completion admission, apartness-completion handoff, transport, replay, provenance, and local naming as one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3601.1, Definition 3601.2, and Definition 3601.3 seed the finite Cauchy-complete apartness-space route over apartness sources, positive gaps, Cauchy tolerance windows, regular readback, Real seals, located-completion criteria, apartness-completion handoffs, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-source exactness, positive-gap visibility, Cauchy tolerance inspection, finite readback, Real seal handoff, located-completion criterion admission, apartness-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompleteApartnessSpaceUp.

Constructive story: A $\mathsf{CauchyCompleteApartnessSpace}^{\uparrow}$ packet is generated from an apartness source row, positive-gap evidence, a Cauchy tolerance route, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a LocatedCompletionCriterion row, an ApartnessCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3602

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ packet records the finite functorial surface of tail shift on regular Cauchy rational names. It sits over the finite window service of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the one-step tail-shift packet of chapter 1578, and the terminal real-name seal of chapter 38. The packet names the identity and composition behaviour of displayed tail shifts; it does not identify quotient streams, select host limits, import countable choice, or claim ambient completeness.

Definition 3602.1 (Regular Cauchy tail-functor carrier)**.** *A* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, W, D, I, K, J, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational name is inspected only through displayed finite windows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the source windows, shifted windows, and comparison windows used by the functorial read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the precision requests and endpoint comparisons preserved by the shifts.*
4. *I is the identity-tail row, recording that the zero shift rereads the same displayed windows and tolerance rows.*
5. *K is the composable-tail row, recording two displayed tail shifts k and ℓ together with the single displayed shift $k + \ell$.*
6. *J is the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *handoff row that supplies each one-step or finite-tail shift certificate used by I and K.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after R, W, D, I, K, J have been displayed.*
8. *H records componentwise* hsame *transport for the source, window, tolerance, identity, composition, shift-handoff, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each tail-shift request through $R, W, D, I, K, J, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyShift}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, D, I, K, J, E, H, C, P, N$. It has no coordinate for quotienting streams by tail agreement, selecting a limit, countable-choice tail selectors, host equality of sequences, or ambient completeness.

Theorem 3602.2 (Regular Cauchy tail-functor NameCert obligations)**.** *For every accepted* RegularCauchyTailFunct *carrier $F = (R, W, D, I, K, J, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the source, window, dyadic tolerance, identity-tail, composable-tail, shift-handoff, Real seal, transport, replay, provenance, and local naming rows; identity stability through I; composition stability through K and J; Real-seal non-escape through E; and refusal of quotient stream equality, selected limits, countable choice, host sequence equality, and ambient completeness as certificate data.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFunctorUp.RegularCauchyTailFunctorTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3602.1. The analytic rows are R, W, D, I, K, J, E, and the structural rows H, C, P, N transport, replay, source, and name the same finite data.

The identity obligation is read from I. It records the zero-tail request against the same displayed $\mathsf{StreamName}^{\uparrow}$ windows and the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, so the accepted read has no second source row.

The composition obligation is read from K and J. The two displayed shifts k and ℓ are admitted only through $\mathsf{RegularCauchyShift}^{\uparrow}$ handoff rows, and the same handoff surface records the single displayed shift $k + \ell$.

Finally E is downstream of the source, window, tolerance, identity, composition, and shift-handoff rows. Since the excluded quotient, selection, choice, host-equality, and completeness objects have no carrier coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3602.3 (Regular Cauchy tail-functor identity and composition)**.** *Let $F = (R, W, D, I, K, J, E, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of the identity tail read and every* $\mathsf{Real}^{\uparrow}$*-facing consumer of a composed tail read factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegularCauchyShift}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The zero shift rereads the displayed R, W, D rows, and the composite of shifts k and ℓ is exported only through the displayed $k + \ell$ shift handoff. No consumer receives a quotient-stream equality, selected host limit, countable-choice selector, or ambient completeness theorem from the packet.

Proof. Use Definition 3602.2. The identity-tail row I is tied to the displayed source R, window schedule W, and tolerance ledger D, so an identity consumer cannot bypass those rows.

For composition, the row K records the two shifts and their displayed sum. The handoff row J supplies the required $\mathsf{RegularCauchyShift}^{\uparrow}$ certificates before any Real-facing seal is read.

The terminal row E seals only this displayed route. The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming, and they do not add quotient, selection, choice, host-equality, or completeness coordinates. □

Theorem 3602.4 (Regular Cauchy tail-functor window composition)**.** *Let $F = (R, W, D, I, K, J, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ *packet. For any two displayed tail shifts k and ℓ recorded by K, their composed tail read is the same finite-window route obtained by rereading the* $\mathsf{StreamName}^{\uparrow}$ *schedule W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, and the* $\mathsf{RegularCauchyShift}^{\uparrow}$ *handoff row J for the displayed shift $k + \ell$. The* $\mathsf{Real}^{\uparrow}$ *seal E is reached only after that $k + \ell$ handoff, so the packet exports no quotient-stream equality, host sequence equality, selected limit, countable-choice selector, or ambient completeness row.*

Proof. Project the carrier in Definition 3602.1. The rows $R, W, D, K, J, E, H, C, P, N$ are the only rows visible to a composed-tail request, with K naming the two displayed shifts and their displayed sum.

Use Definition 3602.2 to read the local certificate obligations. The window data are reread from W, the tolerance checks from D, and the finite-tail handoff from J; the structural rows only transport and replay these displayed rows.

Apply Definition 3602.3 to route every Real-facing consumer through

$$\mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegularCauchyShift}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Because E is downstream of the displayed $k+\ell$ handoff, none of the refused quotient, host-equality, selection, choice, or completeness objects can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailFunctor}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName schedule rows, a DyadicRatCore tolerance ledger, identity-tail and composable-tail rows, RegularCauchyShift handoff rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 3602.1, Definition 3602.2, Definition 3602.3, and Definition 3602.4 seed the finite tail-functor route over StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RegularCauchyShiftUp, RealUp, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable-choice tail selectors, host sequence equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, identity-tail stability, composition stability, RegularCauchyShift handoff exactness, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailFunctorUp.

Chapter 3603

A Concrete Naming Certificate for $\mathsf{OperatorNorm}^{\uparrow}$

Following Bishop's constructive normed-space practice and the bounded-operator presentations in Conway and Kreyszig, the $\mathsf{OperatorNorm}^{\uparrow}$ packet records the public BEDC carrier by which a bounded linear operator exposes a nonnegative scalar bound, a norm-bound ledger, and the transport rows needed by downstream spectral, dual, and uniform-boundedness certificates. It is the named operator-norm surface over the bounded-linear-operator route of chapter 1644, the scalar endpoint discipline of chapter 38, and the functional-analysis stability obligation of Definition 94.3; it is not a host supremum construction, quotient operator equality, a Banach-completeness assertion, or a selected norm-attaining vector.

Definition 3603.1 (OperatorNorm carrier)**.** *An* $\mathsf{OperatorNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (T, K, L, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the carried source endpoint, target endpoint, linear map code, boundedness witness, and source-target norm reads.*
2. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint exposed as the operator-norm scalar boundary.*
3. *L is the norm-bound ledger, recording the input norm row, output norm row, scalar comparison row, and the local inequality route saying that the output norm is controlled by K on carried source inputs.*
4. *H records componentwise* hsame *transport for the bounded-linear-operator row, the* $\mathsf{Real}^{\uparrow}$ *bound endpoint, the norm-bound ledger, the replay row, provenance, and local naming.*
5. *C records displayed* Cont *replay from a bounded-operator request through T, K, L, H back to the same operator-norm surface.*
6. *P is the* Pkg *provenance over* $\mathsf{OperatorNorm}^{\uparrow}$*,* $\mathsf{BoundedLinearOperator}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, K, L, H, C, P, N. It has no coordinate for a host supremum over unit vectors, quotient equality of operators, norm-attainment choice, hidden Banach completeness, spectral-radius limits, or uniform-boundedness families.

Lean4 ✓ `BEDC.Derived.OperatorNormUp.OperatorNormTasteGate single carrier alignment`

Theorem 3603.2 (OperatorNorm NameCert obligations)**.** *For every accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *carrier* $O = (T, K, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-linear-operator source compatibility, nonnegative* $\mathsf{Real}^{\uparrow}$ *bound exposure, norm-bound ledger exactness, componentwise transport, continuation replay, provenance, local naming, and non-escape from the displayed scalar-bound surface. A downstream consumer may read the carried operator row, the displayed scalar bound, and the ledger showing where that bound is used, but it receives no host supremum, no quotient operator equality, no selected norm-attaining vector, no hidden completeness, and no spectral or uniform-boundedness conclusion.*

Proof. Carrier admission is projection from Definition 3603.1. The packet displays exactly the bounded-linear-operator row T, the nonnegative scalar endpoint K, the norm-bound ledger L, and the structural rows H, C, P, N.

The bounded-operator source compatibility is inherited from the public bounded-linear-operator row. Thus a consumer sees the carried source and target endpoints, the map code, and the boundedness witness before it can inspect the scalar bound.

The norm-bound surface is finite. The row L records the input norm, output norm, scalar comparison, and local inequality route, while K supplies the only Real-facing endpoint to which those comparisons may point.

Finally H, C, P, N transport, replay, source, and name only the displayed rows. Since the refused objects have no coordinate in O, the certificate cannot export a host supremum, quotient equality, selected norm-attainment data, hidden completeness, spectral-radius limits, or uniform-boundedness families. □

Theorem 3603.3 (OperatorNorm bound stability)**.** *If two accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *packets are related by the classifier, then they expose the same* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *source row, the same* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint, and the same norm-bound ledger up to the componentwise* hsame *transport row. Consequently every downstream use reads a transported scalar-bound certificate rather than a host-level equality of operators or norms.*

Proof. Unfold the classifier of Definition 3603.1. Its comparison coordinates are exactly the bounded-linear-operator row, Real bound endpoint, norm-bound ledger, replay row, provenance row, and local naming row.

The operator component transports through the inherited bounded-linear-operator classifier, while the scalar endpoint transports through the $\mathsf{Real}^{\uparrow}$ comparison component. The ledger component then moves the input-norm row, output-norm row, scalar comparison row, and inequality route together.

Thus any downstream consumer receives the same public scalar-bound surface after transport. No step introduces a host equality of operators, a supremum equality of norms, or a quotient representative. □

Theorem 3603.4 (OperatorNorm non-escape boundary)**.** *Every read of an accepted* $\mathsf{OperatorNorm}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto T, K, L, H, C, P, N \longmapsto \mathsf{OperatorNorm}^{\uparrow}.$$

In particular, the packet cannot be used to recover a host supremum over all unit vectors, a norm-attainment selector, quotient operator equality, a Banach-completeness assertion, a spectral-radius theorem, or a uniform-boundedness principle.

Proof. Start with an accepted packet $O = (T, K, L, H, C, P, N)$. Any read first exposes the bounded-linear-operator row T and the scalar endpoint K, then reads the norm-bound ledger L and structural rows.

Replay through C returns to the same finite route, and transport through H is componentwise. The provenance and naming rows record only this displayed surface.

The excluded analytic objects require coordinates not present in the carrier. Since no such coordinate can be manufactured by transport or replay, every accepted read remains inside the public operator-norm bound certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}$.

Constructive story: An $\mathsf{OperatorNorm}^{\uparrow}$ packet is generated from a bounded-linear-operator row, a nonnegative $\mathsf{Real}^{\uparrow}$ bound endpoint, a norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3603.1, Definition 3603.2, Definition 3603.3, and Definition 3603.4 seed the public scalar-bound certificate over BoundedLinearOperatorUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OperatorNorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OperatorNormUp.OperatorNormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host supremum construction, quotient operator equality, norm-attainment choice, hidden Banach completeness, spectral-radius limits, uniform-boundedness families, or bridge checking.

Upgrade path: Further checked bound-stability and non-escape theorems for BEDC.Derived.OperatorNormUp would extend the verification axis while preserving the displayed scalar-bound surface.

Chapter 3604

A Concrete Naming Certificate for $\mathsf{ContinuousImageCompact}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet records the finite-net reading of the image of a compact metric source under a displayed continuous map. It sits over the compact source of chapter 129, the graph and pointwise-continuity row of chapter 130, and the compact-uniform-continuity route of chapter 388. The packet names the finite image-net certificate needed by downstream compactness and extreme-value consumers; it is not an open-cover compactness theorem, not a host image-set construction, and not a quotient equality principle for target names.

Definition 3604.1 (Continuous image compact carrier)**.** *A* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, U, E, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes only the compact metric inspection data, finite source-net rows, source radii, and source* $\mathsf{NameCert}$ *provenance consumed by the image certificate.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from the source surface displayed by K into the target metric surface. It records the graph, pointwise continuity data, and target readback boundary available to the packet.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *row for F over K. It turns each target tolerance request into a finite source-net radius before any image-net row is exported.*
4. *E is the finite source-net ledger. It records the displayed source centers, the radius attached to the requested target tolerance, and the finite coverage row supplied by K.*
5. *T is the finite target-image ledger. It records the target names $F(e)$ for each displayed source center e in E and the target tolerance justified through U.*
6. *H records componentwise* hsame *transport for the compact source, continuous-map row, compact-uniform-continuity row, source-net ledger, target-image ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an image-compactness request through K, F, U, E, T, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{ContinuousImageCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, F, U, E, T, H, C, P, N$. It has no coordinate for open-cover compactness, a host image subset, a selected inverse image, quotient equality of target names, or ambient completeness.

Theorem 3604.2 (Continuous image compact NameCert obligations)**.** *For every accepted* ContinuousImageCompac *carrier $I = (K, F, U, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the compact source, continuous-map row, compact-uniform-continuity row, source-net ledger, target-image ledger, transport, replay, provenance, and local naming rows;*
2. *source-net accountability, requiring every displayed source center and radius in E to be read through the compact metric row K;*
3. *continuity routing, requiring every target tolerance request to pass through the compact-uniform-continuity row U for the continuous-map row F before T is exposed;*
4. *image-net exactness, requiring every target center in T to be the displayed F-readback of a source center from E;*
5. *componentwise transport and replay stability under H and C for the same displayed rows;*
6. *ledger refusal, stating that open-cover compactness, host image sets, selected inverse images, quotient equality, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.ContinuousImageCompactUp.ContinuousImageCompactTasteGate single carrier alignment`

Proof. Project the accepted carrier. The mathematical rows are K, F, U, E, T, and the structural rows are H, C, P, N.

The source-net obligation is read first. Row E may display only centers and radii whose coverage data has already been supplied by the compact metric row K.

The continuity route is next. A target tolerance is sent through U, which supplies the source radius for the same displayed map row F over the same compact source K.

The target-image ledger is therefore forced. Each target center in T is obtained by reading F at one of the displayed source centers in E, and the radius attached to that target center is the tolerance justified by U.

Transport, replay, provenance, and local naming preserve the row order. Since the excluded objects do not occur among $K, F, U, E, T, H, C, P, N$, no local obligation can export them. □

Theorem 3604.3 (Finite-net image compactness)**.** *Every accepted* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *carrier $I = (K, F, U, E, T, H, C, P, N)$ exports compactness of the displayed image only as the finite target-net row T obtained from the source-net row E along the continuous-map row F and the compact-uniform-continuity row U. A downstream compactness consumer receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{CompactUniformContinuity}^{\uparrow}, \quad \mathsf{ContinuousImageCompact}^{\uparrow}$$

and receives no open-cover compactness datum, host image subset, selected preimage, quotient equality of target names, or ambient completeness coordinate.

Proof. Begin with an image-compactness request at a target tolerance. By the obligation theorem, the request must pass through U before the target-image ledger T can be read.

The row U supplies a source radius for the same map row F over the same compact source K. The compact source row then exposes a finite source-net ledger E at that radius.

Map the displayed source centers through F. The finite list of target readbacks is exactly the target-image ledger T, and the tolerance attached to it is the one already routed through U.

Rows H, C, P, N transport, replay, package, and name this finite route without adding another coordinate. Thus the exported compactness content is the finite image-net row T and not an open-cover, host-image, selected-preimage, quotient-equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}$.

Constructive story: A $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a ContinuousMapUp row, a CompactUniformContinuityUp row, a finite source-net ledger, a finite target-image ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3604.1, Definition 3604.2, and Definition 3604.3 seed the finite-net image-compactness route from compact metric sources to downstream compactness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousImageCompact}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host image-set construction, selected preimages, quotient equality of target names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-net accountability, continuity routing, image-net exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.ContinuousImageCompactUp.

Chapter 3605

A Concrete Naming Certificate for $\mathsf{CompactMetricContinuousImage}^{\uparrow}$

Bishop-style compact metric reasoning treats the image of a compact metric source under a continuous map through finite approximations: choose a source net, route the requested target tolerance through the displayed continuity modulus, and read the image centers by evaluating the displayed map on the source centers. The $\mathsf{CompactMetricContinuousImage}^{\uparrow}$ packet records exactly that finite route. It sits over the compact metric source of chapter 129, the continuous-map graph row of chapter 130, the target metric-space observations of chapter 57, the uniform-modulus consumer of chapter 369, and the total-bounded image handoff of chapter 3924. The packet names finite image-net witnesses without claiming open-cover compactness, a host image subset, or a selected inverse image.

Definition 3605.1 (Compact metric continuous image carrier)**.** *A* $\mathsf{CompactMetricContinuousImage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, M, U, E, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the displayed finite source-net data, source radii, and compact-metric provenance.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from the source carrier to the target carrier.*
3. *M is the target* $\mathsf{Met}^{\uparrow}$ *observation row used to compare the displayed target names.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row routing each target tolerance to a source radius for the same displayed map row.*
5. *E is the finite source-net ledger read from K at the source radius supplied by U.*
6. *T is the finite target-image ledger whose centers are the displayed F-readbacks of the centers in E.*
7. *H records componentwise* hsame *transport for the compact source, map row, metric observation row, modulus row, source-net ledger, target-image ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a target-image request through $K, F, M, U, E, T, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricContinuousImage}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImage}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, F, M, U, E, T, H, C, P, N$. It has no coordinate for open-cover compactness, an ambient image subset, selected preimages, quotient equality of target names, choice-selected centers, or target completeness.

Lean4 ✓ `BEDC.Derived.CompactMetricContinuousImageUp.CompactMetricContinuousImageTasteGate single carrier alignment`

Theorem 3605.2 (Compact metric continuous image obligations)**.** *For every accepted* CompactMetricContinuousIm *carrier $I = (K, F, M, U, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImage}^{\uparrow}}$ *surface consists exactly of carrier admission, compact source-net accountability, continuous-map graph routing, target metric observation, uniform-modulus compatibility, finite image-net exactness, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3605.1. The mathematical rows are K, F, M, U, E, T, and the structural rows are H, C, P, N.

The source-net read is governed by K and U. A target tolerance is first sent through the uniform-modulus row U, and only then may the source-net ledger E be read from the compact metric source K.

The target-image ledger is governed by F, M, E, T. Each center in T is the displayed value of the continuous-map graph row F at a center already listed in E, with target comparisons read through the metric observation row M.

Transport and replay preserve that same finite route. Since the excluded open-cover, host-image, selected-preimage, quotient-equality, choice, and completeness objects do not occur among the rows, no local obligation can export them. □

Theorem 3605.3 (Finite image-net handoff)**.** *Every compactness-facing or total-boundedness-facing consumer of an accepted* $\mathsf{CompactMetricContinuousImage}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{UniformModulus}^{\uparrow},\quad \mathsf{CompactContinuousImageTotalBounded}^{\uparrow}.$$

The exported witness is the finite target-image ledger T obtained from E along F and U; it is not an open-cover compactness theorem, not a host image-set construction, and not a choice-selected family of target centers.

Proof. Start with a target-image request. The carrier forces the request through U, which supplies a source radius for the same displayed continuous-map row F.

Read the compact metric source row K at that radius to obtain the finite source-net ledger E. No open cover or arbitrary subcover selector is inspected.

Apply the displayed graph row F to the centers in E and compare the resulting target names through M. The resulting finite list is exactly the target-image ledger T.

Rows H, C, P, N transport, replay, package, and name this route without adding another coordinate. Thus the handoff can be consumed by $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ as finite image-net data, but not as ambient image compactness or a selected inverse-image construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImage}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricContinuousImage}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap graph rows, MetricSpace target observations, UniformModulus tolerance routing, finite source-net ledgers, finite target-image ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3605.1, Definition 3605.2, and Definition 3605.3 seed the compact-metric continuous-image route from finite source nets to finite target-image witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, host image-set construction, selected preimages, quotient equality of target names, choice-selected centers, target completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-net accountability, graph routing, metric observation, uniform-modulus compatibility, image-net exactness, transport, replay, provenance, and local NameCert registration for BEDC.Derived.CompactMetricContinuousImageUp.

Chapter 3606

A Concrete Naming Certificate for $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive compact metric analysis, the $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ packet records the finite BEDC surface by which a displayed finite cover over a compact metric source yields a positive-radius ledger. It sits between the compact metric source of chapter 129, the metric ball rows of chapter 57, the finite radius ledger of chapter 452, the theorem route of chapter 2301, and the uniform-modulus consumer of chapter 369. The packet names finite-cover-to-positive-radius data only; it is not an open-cover compactness theorem, a subcover selector, a host minimum principle, countable choice, or an ambient completeness claim.

Definition 3606.1 (Finite-cover Lebesgue-number carrier)**.** *A* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$F = (K, M, V, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row supplying the compact finite-net and coverage data admitted by the cover request.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row through which distances, rational balls, and containment checks are read.*
3. *V is the displayed finite cover row, listing the named cover cells and their metric ball witnesses.*
4. *R is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive radius ledger attached to the displayed finite cover.*
5. *L is the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *route row certifying that the compact metric source and finite cover have been routed to the positive ledger R.*
6. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that may consume the positive radius after K, M, V, R, L have been exposed.*
7. *H records componentwise* hsame *transport for $K, M, V, R, L, U, C, P, N$.*
8. *C records displayed* Cont *replay from compact metric finite nets and metric ball containment rows to the finite cover, positive radius ledger, lemma route, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{LebesgueNumberLemma}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $K, M, V, R, L, U, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteCoverLebesgueNumberUp.TasteGate.FiniteCoverLebesgueNumberTasteGate single carrier alignment`

Theorem 3606.2 (Finite-cover Lebesgue-number NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier* $F = (K, M, V, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$ *surface consists exactly of five rows: carrier admission for the displayed tuple; cover-to-radius exactness binding the finite cover row V to the positive radius ledger R; lemma-row compatibility requiring the* $\mathsf{LebesgueNumberLemma}^{\uparrow}$ *row L to read the same compact metric source, metric ball observations, finite cover, and positive ledger; uniform handoff through U; and non-escape from open-cover compactness, selected subcovers, host minima, countable choice, and ambient completeness.*

Proof. Project the accepted carrier of Definition 3606.1. The displayed tuple contains exactly the compact metric source, metric observation row, finite cover, positive radius ledger, lemma route, uniform handoff, and structural rows.

Cover-to-radius exactness is read through V and R. The finite cover is not replaced by an arbitrary open family, and the radius is not selected by a host minimum; it is the displayed positive ledger attached to the cover row.

Lemma-row compatibility is read through L. It records that the compact metric source K, metric ball row M, finite cover row V, and positive ledger R are the same rows consumed by the $\mathsf{LebesgueNumberLemma}^{\uparrow}$ route.

The uniform handoff U can be reached only after this finite route has been transported by H and replayed by C, with provenance and local naming stored in P and N. Since the carrier has no further coordinate, no accepted packet imports open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness. □

Theorem 3606.3 (Finite-cover Lebesgue-number uniform-radius handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *carrier reaches the positive radius only through*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{FiniteLebesgueNumber}^{\uparrow}, \quad \mathsf{LebesgueNumberLemma}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The consumer-facing route is the finite compact-cover-to-positive-radius handoff $K, M, V, R, L, U, H, C, P, N$, not open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness.

Proof. Read K, M, V first. These rows display the compact metric source, the metric ball observations, and the finite cover whose cover cells generate the positive-radius request.

Then read R and L. The radius ledger is the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ row for the displayed finite cover, and the lemma route records the compact-cover-to-ledger compatibility needed before a uniform-modulus consumer can inspect the radius.

Finally read U, H, C, P, N. These rows transport, replay, source, and name the same finite route. The carrier therefore exports a uniform-radius handoff without adding any coordinate for open-cover compactness, selected subcovers, host minima, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ packet is generated from a CompactMetric source row, Metric ball observations, a displayed finite cover, a FiniteLebesgueNumber positive radius ledger, a LebesgueNumberLemma route row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3606.1, Definition 3606.2, and Definition 3606.3 seed the finite compact-cover-to-positive-radius data route consumed by uniform-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteCoverLebesgueNumberUp.TasteGate.FiniteCoverLebesgueNumberTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, host minima, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-to-radius exactness, lemma-row compatibility, uniform handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.FiniteCoverLebesgueNumberUp.

Chapter 3607

A Concrete Naming Certificate for $\mathsf{ContractionPrinciple}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{ContractionPrinciple}^{\uparrow}$ packet records the theorem-level completion-neighbour object that connects the contraction-map surface of chapter 1418, the finite iteration ledger of chapter 2383, the complete-metric handoff of chapter 128, the metric source of chapter 57, and the endpoint-facing consumer discipline of chapter 396. The packet names the finite route through which contraction data becomes completion-facing evidence; it does not itself choose a fixed point or assert ambient completeness.

Definition 3607.1 (ContractionPrinciple carrier)**.** *A* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, d, T, q, I, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *carrier row whose point histories and classifier are read by the contraction route.*
2. *d is the metric-distance row used for all finite source, target, and iterate-distance comparisons.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *graph row, exposing the self-map application surface without host function extensionality.*
4. *q is the rational contraction-bound ledger, including the displayed nonnegative row and strict unit-bound row consumed by finite estimates.*
5. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *route, displaying the initial row, finite iterate window, and residual-budget rows generated from T and q.*
6. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, naming the completion-facing service that may consume the finite Cauchy evidence without being stored as ambient completeness in the contraction packet.*
7. *H records componentwise* hsame *transport for the metric source, distance row, contraction graph, rational bound, finite iteration route, completion handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for metric reads, contraction graph inspection, bound comparison, finite iteration extension, completion handoff, and consumer readback.*
9. *P is the* Pkg *provenance row over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$ *row.*

The carrier is a theorem-level route packet. It contains no selected fixed point, no completed endpoint row, no host equality of maps, and no completeness assertion outside the displayed handoff row.

Lean4 ✓ `BEDC.Derived.ContractionPrincipleUp.ContractionPrincipleTasteGate single carrier alignment`

Theorem 3607.2 (ContractionPrinciple NameCert obligations)**.** *For an accepted* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier* $R = (X, d, T, q, I, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$ *surface consists of carrier admission, contraction-bound stability, iterate visibility, complete-metric handoff visibility, componentwise transport, continuation replay, package provenance stability, and local naming for the packet of Definition 3607.1. Every accepted read of the principle is routed through these rows.*

Proof. Carrier admission projects the displayed tuple $X, d, T, q, I, L, H, C, P, N$. The contraction graph row and rational bound ledger supply the map-level comparison surface, while the metric row is the only source of distance reads.

Iterate visibility is supplied by I. It records only finite Picard windows and residual budgets generated from T and q, so it cannot introduce a completed orbit or a selected endpoint.

The completion handoff is supplied by L, transported by H, and replayed by C. Provenance and local naming are then read from P and N, giving exactly the listed NameCert obligations and no additional analytic coordinate. □

Theorem 3607.3 (ContractionPrinciple completion handoff)**.** *Every* $\mathsf{BanachFixedPoint}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{ContractionPrinciple}^{\uparrow}$ *carrier must pass through the displayed route*

$$\mathsf{ContractionMapping}^{\uparrow}, \quad \mathsf{FixedPointIteration}^{\uparrow}, \quad \mathsf{CompleteMetric}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}.$$

Equivalently, the consumer reads the rows $X, d, T, q, I, L, H, C, P, N$ before any endpoint-facing fixed-point seal can be consumed. The handoff exports contraction-map evidence, finite iterate evidence, metric-distance reads, complete-metric service visibility, transport, replay, provenance, and local naming; it does not export a selected fixed point, a completed orbit endpoint, host fixed-point equality, quotient equality of maps, or ambient completeness.

Proof. Start with the contraction surface. The rows T and q are read as the $\mathsf{ContractionMapping}^{\uparrow}$ graph and rational bound ledger over the $\mathsf{Met}^{\uparrow}$ source X, d.

Next read the finite iteration route I. It exposes the $\mathsf{FixedPointIteration}^{\uparrow}$ orbit window and residual budget that a completion-facing consumer may inspect before any endpoint row is available.

Finally pass through L, H, C, P, N. The row L is only a $\mathsf{CompleteMetric}^{\uparrow}$ handoff, while H and C preserve the same componentwise route under transport and replay. Therefore a $\mathsf{BanachFixedPoint}^{\uparrow}$ or $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives the displayed theorem-level route but must provide its own endpoint or completeness discipline downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{ContractionPrinciple}^{\uparrow}$ packet is generated from a MetricUp source row, a distance row, a ContractionMapping graph row, a rational contraction-bound ledger, a FixedPointIteration route, a CompleteMetric handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3607.1, Definition 3607.2, and Definition 3607.3 seed the theorem-level contraction principle route over metric, contraction-map, finite iteration, complete-metric, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractionPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, completed endpoints, host fixed-point equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-bound stability, iterate visibility, complete-metric handoff visibility, transport stability, replay stability, provenance stability, and local naming for BEDC.Derived.ContractionPrincipleUp.

Chapter 3608

A Concrete Naming Certificate for $\mathsf{CauchyProductConvergence}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{CauchyProductConvergence}^{\uparrow}$ packet records the finite Real-and-completion route by which two displayed regular Cauchy or summable rows expose a Cauchy product name. It sits over the regular rational readback of chapter 301, the tolerance ledger of chapter 291, the completion witness route of chapter 3579, and the product theorem surface of chapter 3584. The chapter names only the finite certificate route for product convergence; it does not assert quotient stream equality, selected infinite limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3608.1 (Cauchy product convergence carrier)**.** *A* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, P, T, R, D, W, E, H, C, Q, N)$$

with the following displayed rows:

1. *A and B are the two source rows, each presented either as a regular Cauchy row or as a summable row with the finite windows needed by the product route.*
2. *P is the Cauchy product window. It records the finite convolution band, the summand-pair reads, and the displayed partial product windows used by the convergence claim.*
3. *T is the tail-control row. It records the finite majorant, Mertens, or Bishop–Bridges tail ledger that bounds the product tail after the window P has been fixed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the product windows determined by A, B, P, T.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the source reads, product-window reads, and tail-control inequalities.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window data obtained from the same product windows and tolerance requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback, stream windows, and dyadic tolerance ledger have been exposed.*
8. *H records componentwise* hsame *transport for the source rows, Cauchy product window, tail-control row, readback, tolerance ledger, stream windows, Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a product-convergence request through $A, B, P, T, R, D, W, E, H, Q, N$.*
10. *Q is the* Pkg *provenance over* $\mathsf{CauchyProductConvergence}^{\uparrow}$, $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite product-convergence history. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductConvergenceUp.CauchyProductConvergenceTasteGate single carrier alignment`

Theorem 3608.2 (Cauchy product convergence NameCert obligations)**.** *For every accepted* CauchyProductConverg *carrier $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two source rows, the product window, the tail-control row, the regular readback, the dyadic tolerance ledger, stream windows, the Real seal, transport, replay, provenance, and local naming;*
2. *product-window stability under componentwise* hsame *transport of A, B, P, T;*
3. *tail-control compatibility between the Cauchy product window P, the tail row T, and the dyadic tolerance ledger D;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback for the product windows exposed by P, T, D;*
5. $\mathsf{StreamName}^{\uparrow}$ *window exposure using the same finite product windows and tolerance rows;*
6. $\mathsf{Real}^{\uparrow}$ *seal nonescape through R, D, W, E;*
7. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
8. *ledger refusal for any product-convergence consumer that omits a source row, product window, tail-control row, tolerance ledger, readback row, stream window, Real seal, transport row, replay row, provenance row, or local naming row.*

Proof. Project the carrier of Definition 3608.1. The projection exposes exactly the two source rows, the Cauchy product window, tail-control ledger, regular readback, dyadic tolerance ledger, stream windows, Real seal, and structural support rows.

The product-window and tail obligations are bound by P, T, D. A consumer can read a product window only together with the corresponding source-window reads and dyadic tolerance request, and it can claim tail control only through the displayed row T.

The readback and seal obligations are bound by R, W, E. These rows repack the same finite product windows as regular rational and stream-name data before the terminal Real seal is available.

Finally H, C, Q, N transport, replay, package, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, or ambient completeness coordinate, the obligation list is exhaustive. □

Theorem 3608.3 (Cauchy product convergence tail handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *packet factors through the finite route*

$$A, B \longmapsto P, T, D \longmapsto R, W \longmapsto E, H, C, Q, N.$$

Thus the read first consumes the two source rows, then the displayed Cauchy product window, tail-control row, and dyadic tolerance ledger, then the regular rational and stream-name readbacks, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient stream semantics, selected product limit, countable-choice schedule, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$. The only source coordinates are A and B, so all source regularity or summability reads begin there.

The next stage is P, T, D. The product window P fixes the finite convolution band, the tail-control row T supplies the displayed convergence bound for the omitted band, and the dyadic ledger D records the tolerance requests used by both.

The readback stage is R, W. The regular rational row and stream-name row use the same product windows fixed by P, T, D, so there is no separate selected-limit coordinate.

Finally the seal stage is E, H, C, Q, N. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the readback rows, and the structural rows transport, replay, package, and name the same finite route. □

Theorem 3608.4 (Cauchy product convergence tail-budget fusion)**.** *Every accepted* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *tail-control row that consumes a* $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ *surface factors through chapter 3584 before the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows are read. In the carrier* $K = (A, B, P, T, R, D, W, E, H, C, Q, N)$, *the theorem packet is consumed by the source-window and tail-control stage* A, B, P, T, *while* R, D, W, E *only read the resulting finite product route.*

Proof. Start with the carrier route of Definition 3608.3. Its first stage consumes A, B, P, T, and the provenance row Q already names $\mathsf{MertensCauchyProductTheorem}^{\uparrow}$ among the package sources.

The readback rows R, D, W, E occur only after this stage. Therefore a consumer cannot read regular rational, dyadic, stream-name, or Real-seal data for the product tail until the Mertens theorem packet has fixed the finite product-window surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductConvergence}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source row, product window, tail-control row, DyadicRatCore tolerance ledger, RegSeqRat readback, StreamName window, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductConvergence}^{\uparrow}$ packet is generated from two source regular Cauchy or summable rows, a Cauchy product window, a tail-control row, RegSeqRat readback, DyadicRatCore tolerance ledger, StreamName windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 3608.1, Definition 3608.2, and Definition 3608.3 seed the finite product-convergence route from source rows and Cauchy product windows to tail control, readback, tolerance, StreamName, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, product-window stability, tail-control compatibility, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductConvergenceUp.

Chapter 3609

A Concrete Naming Certificate for $\mathsf{CauchyProductMertens}^{\uparrow}$

Following Mertens and Bishop–Bridges constructive analysis, the $\mathsf{CauchyProductMertens}^{\uparrow}$ packet records the finite theorem row that mediates between the displayed Mertens Cauchy product theorem of chapter 3584 and the product-convergence seal of chapter 3608. It sits over the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the stream-window route of chapter 126, the completion witness of chapter 3579, and the terminal Real naming surface of chapter 38.

Definition 3609.1 (Cauchy product Mertens carrier)**.** *A* $\mathsf{CauchyProductMertens}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, W, T, R, D, S, E, H, C, Q, N)$$

whose rows record two displayed regular Cauchy or summable source names A, B*, a finite convolution-window row* W*, a Mertens tail-budget row* T*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, a* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, a* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *handoff row* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* Q*, and the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$ *row* N*. The carrier predicate asserts that all rows belong to one finite product-theorem history from the Mertens theorem packet to the later convergence packet; it contains no coordinate for quotient streams, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductMertensUp.CauchyProductMertensTasteGate single carrier alignment`

Theorem 3609.2 (Cauchy product Mertens NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductMertens}^{\uparrow}$ *carrier* $M = (A, B, W, T, R, D, S, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$ *surface consists exactly of carrier admission for those twelve rows, Mertens-window compatibility between* A, B, W, T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance routing through* D*,* $\mathsf{StreamName}^{\uparrow}$ *exposure through* S*, completion handoff through* E*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and refusal of any consumer that omits one of these rows.*

Proof. Project the carrier in Definition 3609.1. The source, window, and tail rows A, B, W, T expose exactly the Mertens theorem surface needed before product convergence can be read.

The readback rows R, D, S, E then carry the same finite window through regular rational, dyadic, stream-name, and completion-witness routes. The structural rows H, C, Q, N transport, replay, source, and name that route.

Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, or ambient completeness coordinate, the displayed list exhausts the local NameCert obligations. □

Theorem 3609.3 (Cauchy product Mertens tail-budget handoff)**.** *Every accepted* $\mathsf{CauchyProductMertens}^{\uparrow}$ *tail-budget row that can be consumed by* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *first factors through the finite route*

$$A, B \longmapsto W, T \longmapsto R, D, S, E \longmapsto H, C, Q, N.$$

Thus the Mertens product theorem packet is read before the $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows of the convergence route are consumed.*

Proof. Start from an accepted carrier $M = (A, B, W, T, R, D, S, E, H, C, Q, N)$. The only rows that expose the product theorem surface are A, B, W, T, so a convergence-facing tail read begins with those coordinates.

The readback stage is then forced through R, D, S, E. These rows encode the same finite window as regular rational, dyadic tolerance, stream-window, and completion-witness data.

Finally H, C, Q, N provide transport, replay, provenance, and local naming. No other coordinate can supply a tail-budget handoff to the convergence packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchyProductMertens}^{\uparrow}$ consumer that reaches a product-convergence tail handoff while omitting a source row, convolution window, Mertens tail row, RegSeqRat readback, DyadicRatCore tolerance ledger, StreamName window, CauchyCompletenessWitness handoff, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductMertens}^{\uparrow}$ packet is generated from two source regular Cauchy or summable rows, a finite convolution window, a Mertens tail-budget row, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, CauchyCompletenessWitness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3609.1, Definition 3609.2, and Definition 3609.3 seed the finite Mertens product-theorem route from source rows and tail budgets to readback, stream, completion, and convergence-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductMertens}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyProductMertensUp.CauchyProductMertensTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Mertens-window compatibility, RegSeqRat readback, DyadicRatCore tolerance routing, StreamName exposure, CauchyCompletenessWitness handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductMertensUp.

Chapter 3610

A Concrete Naming Certificate for $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$

Following Krein and Milman, the $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ packet records the finite extreme-point witness route that a convex-analysis consumer may inspect before any ambient locally convex compactness theorem is available. It sits over the compact metric carrier of chapter 129, the convex-set interface of chapter 208, the scalar $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the weak functional-window surface of chapter 3597, the operator-norm row of chapter 3603, and the finite weak-star window of chapter 3420. The chapter names only a finite exposed-face and candidate-extreme certificate; it does not assert full locally convex duality, selected supporting functionals, unrestricted compactness, quotient convex bodies, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3610.1 (Krein–Milman finite extreme-point carrier)**.** *A* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, V, X, F, E, S, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *row carrying the finite compact-window data through which the candidate convex body is inspected.*
2. *V is the ambient vector or functional-analysis row, read through the weak-topology and operator-norm sibling packets.*
3. *X is the* $\mathsf{ConvexSet}^{\uparrow}$ *row for the displayed convex subset, including the affine-combination and transport data available to the packet.*
4. *F is the finite exposed-face ledger, listing the exposed faces, scalar thresholds, comparison budgets, and source probes that the certificate actually opens.*
5. *E is the candidate extreme-point witness row, recording the displayed points and finite refusal rows for nontrivial convex decompositions inside the opened face ledger.*
6. *S is the separating-functional ledger. Each functional is read only through finite weak-topology, operator-norm, and scalar* $\mathsf{Real}^{\uparrow}$ *observation rows.*
7. *W records the scalar and metric windows used by F, E, S, including finite* $\mathsf{Real}^{\uparrow}$ *readbacks for values, gaps, and tolerance comparisons.*

8. *H records componentwise* hsame *transport for the compact metric row, ambient row, convex-set row, exposed-face ledger, candidate extreme witnesses, separating-functional ledger, scalar windows, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from compact and convex reads through exposed-face and separating-functional ledgers to the candidate extreme witness rows.*
10. *P is the* Pkg *provenance over* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, V, X, F, E, S, W, H, C, P, N$. It has no coordinate for a host choice of all extreme points, a completed dual space, full locally convex separation, unrestricted compactness, quotient convex-body equality, selected supporting functionals outside S, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.KreinMilmanFiniteExtremeUp.KreinMilmanFiniteExtremeUpTasteGate single carrier alignment`

Theorem 3610.2 (Krein–Milman finite extreme-point NameCert obligations)**.** *For every accepted* $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ *carrier*

$$K = (M, V, X, F, E, S, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$ *surface consists exactly of compact-window admission for M, ambient weak functional admission for V, convex-set admission for X, exposed-face ledger visibility for F, candidate extreme-point witness admission for E, separating-functional ledger visibility for S, scalar and metric window readback for W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every consumer of the finite extreme-point certificate must read the route

$$M, V, X \longmapsto F, S, W \longmapsto E \longmapsto H, C, P, N$$

before treating a candidate as an extreme witness. The route exports finite certificate rows only; it exports no selected set of all extreme points, no full separation theorem, no host compact convex hull equality, no quotient convex-body object, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3610.1. The compact and convex source rows are exactly M and X, while the ambient functional-analysis reads are confined to V.

The exposed-face and separating-functional data are then forced through F and S. Their scalar values, thresholds, and gap comparisons are opened through the finite window row W, so a consumer sees only displayed weak and normed scalar observations.

The candidate extreme witness row E is downstream from those ledgers. It records the displayed point and the finite refusal of nontrivial convex decompositions inside the opened face route, not a global selector for all extreme points.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row contains a completed dual space, unrestricted compactness object, quotient convex body, host separation theorem, or ambient completeness principle, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KreinMilmanFiniteExtreme}^{\uparrow}$ packet is generated from compact metric rows, convex-set rows, weak functional-analysis windows, finite exposed-face ledgers, candidate extreme-point witnesses, separating-functional

ledgers, scalar Real readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{KreinMilmanFiniteExtreme}\uparrow}$, seed)

Scope closed: Definition 3610.1 and Definition 3610.2 seed the finite exposed-face route for a Krein–Milman extreme-point witness over compact metric, convex-set, real scalar, weak-topology, operator-norm, and Banach–Alaoglu window rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KreinMilmanFiniteExtreme}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full locally convex duality, unrestricted compactness, host separation theorems, selected supporting functionals, quotient convex-body equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window routing, convex-set admission, finite exposed-face exactness, candidate extreme witness admission, separating-functional ledger visibility, scalar readback, transport, replay, provenance, and local naming for BEDC.Derived.KreinMilmanFiniteExtremeUp.

Chapter 3611

A Concrete Naming Certificate for $\mathsf{CFunCarrier}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley continuous-function carriers, the $\mathsf{CFunCarrier}^{\uparrow}$ packet records a continuous function as finite source, map, target, modulus, and certificate rows rather than as an ambient function-space quotient. It sits between the terminal real-name seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the graph-and-modulus surface of chapter 130. The chapter isolates the non-quotient carrier object consumed by $\mathsf{RealPowerSeries}^{\uparrow}$ and later $\mathsf{ContinuousMap}^{\uparrow}$ readers: a consumer sees only displayed BEDC rows, not a host function, function extensionality principle, selected modulus function, or completed mapping space.

Definition 3611.1 (Continuous function carrier)**.** *A* $\mathsf{CFunCarrier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, F, T, M, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-domain row, including the finite input histories and any* $\mathsf{StreamName}^{\uparrow}$ *windows used to read source points.*
2. *F is the graph or map row, recording only the displayed relation between inspected source histories and target histories.*
3. *T is the target-domain row, including the terminal* $\mathsf{Real}^{\uparrow}$ *or sibling endpoint rows that the carrier may expose.*
4. *M is the modulus row, recording the finite tolerance conversion used by the continuity request.*
5. *Q is the certificate row that binds the source read, graph read, target read, and modulus read into one continuity witness.*
6. *W is the shared finite-window ledger over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and any real-facing tolerance readbacks needed by the source and target rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row used when the carrier is consumed by real-valued maps or power-series endpoints.*
8. *H records componentwise* hsame *transport for the source, map, target, modulus, certificate, window, readback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an input tolerance request through S, F, T, M, Q, W, R to the named endpoint rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CFunCarrier}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, M, Q, W, R, H, C, P, N$. It has no coordinate for host functions, quotient mapping spaces, function extensionality, selected moduli, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ContinuousFunctionCarrierUp.ContinuousFunctionCarrierTasteGate single carrier alignment`

Theorem 3611.2 (Continuous function carrier NameCert obligations)**.** *For every accepted* $\mathsf{CFunCarrier}^{\uparrow}$ *carrier $K = (S, F, T, M, Q, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, source access through S, graph or map access through F, target access through T, modulus access through M, continuity-certificate binding through Q, shared* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows through W, real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3611.1. The only rows available to a consumer are $S, F, T, M, Q, W, R, H, C,$ so source, graph, target, modulus, certificate, window, and readback obligations must be read there.

The continuity witness is not a host theorem. It is the finite row Q together with the shared windows W and the real-facing readback row R, so every endpoint read stays inside the displayed packet.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no other coordinate exists, the listed obligations exhaust the local NameCert surface. □

Theorem 3611.3 (Continuous function carrier L10 readback)**.** *Every L10 consumer read of an accepted* $\mathsf{CFunCarrier}^{\uparrow}$ *packet factors through*

$$S, F, T, M, Q \longmapsto W, R \longmapsto H, C, P, N.$$

Thus $\mathsf{RealPowerSeries}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *consumers may use the displayed source, graph, target, modulus, certificate, stream-window, regular-rational, real-readback, transport, replay, provenance, and local naming rows, but they receive no host function object, quotient mapping space, selected modulus function, function extensionality principle, countable-choice selector, or ambient real-completeness witness.*

Proof. Start with the displayed carrier. The source, graph, target, modulus, and certificate rows form the only continuity-facing entry point.

The finite windows and readback rows come next. They are the rows through which $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ observations are exposed to real-valued and continuous-map consumers.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. The field-exclusion clause of the carrier then blocks every host-function or quotient-function-space escape route. □

Falsifiable prediction: Every accepted $\mathsf{CFunCarrier}^{\uparrow}$ read must expose source, graph or map, target, modulus, certificate, finite StreamName windows, RegSeqRat readback, Real-facing readback when present, componentwise transport, displayed replay, provenance, and local naming rows.

Independence witness: continuousmap, real, streamname, regseqrat: $\mathsf{CFunCarrier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source, graph, target, modulus, certificate, window, readback, transport, replay, provenance, and local naming rows without quotienting maps or selecting host functions.

Closure status for $\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}$.

Constructive story: A $\mathsf{CFunCarrier}^{\uparrow}$ packet is generated from source-domain rows, graph or map rows, target-domain rows, a modulus row, a continuity-certificate row, shared finite StreamName and RegSeqRat windows, real-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3611.1, Definition 3611.2, and Definition 3611.3 seed the finite non-quotient continuous-function carrier route over source, graph, target, modulus, certificate, window, readback, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CFunCarrier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions, function extensionality, quotient mapping spaces, selected modulus functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source access, graph or map access, target access, modulus access, continuity-certificate binding, shared window routing and readback, real-facing readback, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousFunctionCarrierUp.

Chapter 3612

A Concrete Naming Certificate for $\mathsf{SturmRootCount}^{\uparrow}$

Following Sturm's theorem and Bishop–Bridges constructive analysis, the $\mathsf{SturmRootCount}^{\uparrow}$ packet records the finite route by which a displayed polynomial and two endpoint windows yield a root-count ledger without selecting any root in a host field. It consumes the sign-variation surface of chapter 2079 and exposes the retained count to the bracket consumers of chapter 2269. The packet names the count-and-handoff interface between a Sturm-chain variation ledger and a Real-facing isolating-window consumer; it does not name a chosen algebraic root.

Definition 3612.1 (Sturm root count carrier)**.** *A* $\mathsf{SturmRootCount}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, S, E, V, D, I, A, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Poly}^{\uparrow}$ *source row, carrying the coefficient ledger, degree bound, derivative row, and polynomial identity whose interval counts are inspected.*
2. *S is the* $\mathsf{SturmSequence}^{\uparrow}$ *sign-variation window row, recording the accepted Sturm chain, endpoint sign table, zero-deletion witnesses, and the two visible variation counts.*
3. *E is the endpoint dyadic-window row. It stores the left and right dyadic endpoint cells, requested precisions, comparison rows, and endpoint containment checks.*
4. *V is the variation-count difference row. It records the displayed subtraction of the right endpoint variation count from the left endpoint variation count.*
5. *D is the* $\mathsf{RealRootIsolation}^{\uparrow}$ *bracket handoff row, exposing only the polynomial source, endpoint windows, and certified count accepted by the downstream isolating-window consumer.*
6. *I is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, retaining the same two endpoint windows and the same count at unary precision before a real seal is inspected.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row for the two endpoint names and the bracket-facing interval read.*
8. *H records componentwise* hsame *transport for polynomial source, Sturm-sequence window, endpoint dyadic windows, variation-count difference, isolation handoff, regular-rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a root-count request through polynomial inspection, Sturm-chain variation, endpoint dyadic comparison, count subtraction, isolation handoff, regular-rational readback, and real-seal access.*

10. *K is the* Pkg *provenance over* $\mathsf{Poly}^{\uparrow}$, $\mathsf{SturmSequence}^{\uparrow}$, $\mathsf{RealRootIsolation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$ *registration row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected root, a host-field factorization, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SturmRootCountUp.TasteGate.SturmRootCountTasteGate single carrier alignment`

Theorem 3612.2 (Sturm root count NameCert obligations)**.** *For every accepted* $\mathsf{SturmRootCount}^{\uparrow}$ *carrier* $R = (P, S, E, V, D, I, A, H, C, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the rows listed in Definition 3612.1;*
2. *polynomial-source exactness through the coefficient, degree, derivative, and identity cells of P;*
3. *Sturm-window exactness through the finite chain, endpoint sign table, zero-deletion witnesses, and variation counts of S;*
4. *endpoint-window exactness through the dyadic cells, precision requests, comparison rows, and containment checks of E;*
5. *count-difference exactness through the displayed subtraction row V;*
6. *bracket-consumer discipline through the* $\mathsf{RealRootIsolation}^{\uparrow}$ *handoff row D;*
7. *regular-rational and Real-facing readback through I and A;*
8. *componentwise transport, continuation replay, provenance, and local naming through H, C, K, N.*

There is no additional NameCert obligation for selected roots, factorization over a host field, quotient real equality, real trichotomy, or ambient completeness.

Proof. Project the carrier of Definition 3612.1. The polynomial source P and the Sturm-window row S fix the displayed chain, endpoint sign table, zero-deletion witnesses, and variation counts, so a count read cannot inspect roots or factors outside the finite ledger.

The dyadic endpoint row E supplies the only endpoint cells admitted to the count. The difference row V then records the visible subtraction of endpoint variation counts, with no separate root-selection coordinate.

The handoff row D gives the downstream $\mathsf{RealRootIsolation}^{\uparrow}$ consumer exactly the source polynomial, endpoint windows, and accepted count. Rows I and A expose the same endpoint data through $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal.

Finally, H, C, K, N transport, replay, source, and locally name the same route. Since none of those structural rows adds a selected root, factorization theorem, quotient equality proof, trichotomy decision, or completeness witness, the local certificate is exhausted by the displayed rows. □

Theorem 3612.3 (Sturm root count isolation handoff)**.** *Every accepted* $\mathsf{SturmRootCount}^{\uparrow}$ *read handed to a* $\mathsf{RealRootIsolation}^{\uparrow}$ *bracket consumer factors through*

$$P, S, E, V \longmapsto D \longmapsto I, A, H, C, K, N.$$

The consumer receives the polynomial source, Sturm-chain sign-variation window, endpoint dyadic windows, variation-count difference, regular-rational endpoint readback, Real-facing seal, componentwise transport, replay, provenance, and local naming. It receives no selected host root, no factorization over a host field, no quotient real equality principle, no trichotomy decision, and no ambient completeness principle.

Proof. Start with the downstream read. The source polynomial must pass through P, and the variation data must pass through the $\mathsf{SturmSequence}^{\uparrow}$ row S from chapter 2079.

The endpoint cells are then read from E, while V records the displayed count difference. Thus the bracket-facing row can see only a finite source, two endpoint windows, and one accepted integer count.

The handoff row D is the only row available to the $\mathsf{RealRootIsolation}^{\uparrow}$ consumer described in chapter 2269. The regular-rational readback I and real seal A preserve the same endpoint information for Real-facing consumers.

The remaining rows H, C, K, N compare, replay, source, and name the route componentwise. They add no hidden analytic principle, so the isolation handoff remains a finite Sturm-count certificate. □

Falsifiable prediction: Every $\mathsf{SturmRootCount}^{\uparrow}$ read handed to a $\mathsf{RealRootIsolation}^{\uparrow}$ bracket consumer factors through one $\mathsf{Poly}^{\uparrow}$ source row, one $\mathsf{SturmSequence}^{\uparrow}$ sign-variation window row, one dyadic endpoint-window row, one variation-count difference row, one bracket handoff row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, sturmsequence, realrootisolation, dyadicratcore, regseqrat, real: $\mathsf{SturmRootCount}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the polynomial source, Sturm-chain sign-variation windows, endpoint dyadic windows, count-difference row, bracket handoff, readback, real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3612.1, Definition 3612.2, and Definition 3612.3 seed the finite bridge from Sturm-chain variation ledgers to Real-root-isolation bracket consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SturmRootCount}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected roots, factorization over a host field, quotient real equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sturm-window exactness, endpoint-window exactness, count-difference exactness, isolation handoff, readback discipline, and transport-replay coverage for BEDC.Derived.SturmRootCountUp

Constructive story: A $\mathsf{SturmRootCount}^{\uparrow}$ packet is generated from a $\mathsf{Poly}^{\uparrow}$ source row, a $\mathsf{SturmSequence}^{\uparrow}$ sign-variation window row, dyadic endpoint windows, a variation-count difference row, a $\mathsf{RealRootIsolation}^{\uparrow}$ bracket handoff row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3613

A Concrete Naming Certificate for $\mathsf{UniformCauchyBicompletion}^{\uparrow}$

Following Bourbaki on uniform structures and Kelley on completions of uniform spaces, the $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ packet records the finite two-sided Cauchy-filter corridor used to approach a separated uniform-completion boundary. It sits over the uniform-space surface of chapter 339, the Cauchy-filter route of chapter 333, the separated comparison row of chapter 3365, the uniform-completion consumer of chapter 1473, and the real readback boundary of chapter 38. The chapter names only finite bicompletion data for a displayed left and right Cauchy route; it does not construct a quotient completion, select a limit representative, use choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3613.1 (Uniform Cauchy bicompletion carrier)**.** *A* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (U, L, R, J, S, E, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, recording finite entourage membership, diagonal, inverse, refinement, and composition data available to both Cauchy sides.*
2. *L is the left* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing the finite left filter-base observations and their entourage refinements.*
3. *R is the right* $\mathsf{CauchyFilter}^{\uparrow}$ *row, exposing the finite right filter-base observations and their entourage refinements.*
4. *J is the joint Cauchy row, recording that requested left and right tails share a displayed common entourage refinement before any completion-facing read.*
5. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row, used only to identify two displayed boundary reads through separated zero-distance data.*
6. *E is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row, reached after U, L, R, J, S have been displayed.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *readback row for real-valued observers of the separated boundary, available only downstream of the uniform-completion handoff.*
8. *H records componentwise* hsame *transport for the uniform, left-filter, right-filter, joint-Cauchy, separated, completion, Real, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a bicompletion request through $U, L, R, J, S, E, A, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite two-sided Cauchy corridor. It has no coordinate for quotient filters, selected completion points, ambient host equality, choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchyBicompletionUp.UniformCauchyBicompletionTasteGate single carrier alignment`

Theorem 3613.2 (Uniform Cauchy bicompletion NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *carrier $B = (U, L, R, J, S, E, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform-space row, two Cauchy-filter rows, joint-Cauchy row, separated comparison row, uniform-completion handoff, Real readback, transport, replay, provenance, and local naming;*
2. *left and right Cauchy-filter stability under componentwise* hsame *transport over the same* $\mathsf{UniformSpace}^{\uparrow}$ *row U;*
3. *joint-refinement exactness through J, requiring both Cauchy sides to reach every displayed entourage request through finite filter-base refinements;*
4. *separated-boundary exactness through S, requiring any identification of two displayed boundary reads to pass through the separated comparison row;*
5. *completion and Real nonescape through E and A, requiring every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to stay downstream of U, L, R, J, S;*
6. *displayed* Cont *replay and* Pkg *provenance for the whole two-sided corridor;*
7. *refusal of quotient filters, selected completion points, ambient host equality, choice, and ambient Real completeness as certificate data.*

Proof. Project the carrier of Definition 3613.1. The source rows are $U, L, R, J, S, E, A, H, C, P, N$, so every obligation must be read from those coordinates.

The two Cauchy sides are bound by L and R over the same uniform row U. The joint row J records the common finite entourage refinement requested by a bicompletion read, so neither side can be replaced by an independent host filter or a selected representative.

The separated and completion-facing rows are S, E, A. The row S is the only comparison route for displayed boundary reads, E is available only after the joint two-sided Cauchy data have been exposed, and A is downstream of that handoff.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same corridor. Since the carrier has no quotient-filter, selected-point, host-equality, choice, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3613.3 (Uniform Cauchy bicompletion separated route)**.** *Every separated* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ *packet factors through the finite route*

$$U \longmapsto L, R \longmapsto J \longmapsto S \longmapsto E, A, H, C, P, N.$$

Thus the consumer first reads the uniform entourage row, then both Cauchy-filter sides, then the joint-refinement row, then the separated comparison row, and only then the completion handoff and Real readback rows. The route exports no quotient completion, selected limit representative, choice schedule, ambient host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $B = (U, L, R, J, S, E, A, H, C, P, N)$. The only uniform source is U, so every completion-facing read begins with the displayed entourage data.

The two Cauchy inputs are L and R. Their common-refinement witness is J, which records the finite two-sided response to each requested entourage before a boundary read is available.

The separated boundary is S. Any comparison of two displayed completion reads must pass through this row, so host equality or a selected quotient representative cannot enter the route.

The endpoint rows are E, A, H, C, P, N. They hand off to the uniform-completion and Real-facing consumers, transport and replay the same data, and locally name the corridor without adding quotient, selection, choice, host-equality, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ consumer that reaches a separated $\mathsf{UniformCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ boundary while omitting the UniformSpace row, either CauchyFilter side, the joint-refinement row, the SeparatedMetric comparison row, transport, replay, provenance, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyBicompletion}^{\uparrow}$ packet is generated from a UniformSpace entourage row, two CauchyFilter rows, a joint-refinement row, a SeparatedMetric comparison row, a UniformCompletion handoff, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3613.1, Definition 3613.2, and Definition 3613.3 seed the finite two-sided Cauchy corridor from uniform entourage rows to separated completion and Real readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyBicompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limit representatives, choice, ambient host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left and right Cauchy-filter stability, joint-refinement exactness, separated-boundary exactness, UniformCompletion handoff nonescape, Real readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyBicompletionUp.

Chapter 3614

A Concrete Naming Certificate for $\mathsf{DyadicCauchyApproximation}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ packet records the finite dyadic approximation surface used to pass from tolerance requests to regular rational and real-name consumers. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the completed real-name boundary of chapter 38, and the locator-facing route of chapter 3410. The chapter names only finite dyadic approximation data; it does not introduce quotient streams, selected limits, trichotomy, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3614.1 (Dyadic Cauchy approximation carrier)**.** *A* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius, comparison, and refinement data requested by the approximation.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the inspected stream prefixes associated with the dyadic tolerance request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same dyadic tolerances and stream windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after D, W, Q have been displayed.*
5. *L is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *handoff row, recording the locator-facing use of the same finite approximation data.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, Real seal, locator handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an approximation request through D, W, Q, E, L to the named consumer rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite approximation history. It has no coordinate for quotient stream equality, selected limits, trichotomy, open-cover compactness, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.DyadicCauchyApproximationUp.DyadicCauchyApproximationTasteGate single carrier alignment`

Theorem 3614.2 (Dyadic Cauchy approximation NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *carrier* $K = (D, W, Q, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, dyadic tolerance admission through* D*, finite stream-window exposure through* W*, regular rational readback through* Q*, RealUp sealing through* E*, CauchyRealLocator handoff through* L*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3614.1. The only rows available to an approximation consumer are $D, W, Q, E, L, H, C, P, N$, so tolerance, window, readback, seal, locator, transport, replay, provenance, and naming obligations must be read there.

The dyadic and stream obligations are bound by D and W. The dyadic radius, comparison, and refinement data come from the $\mathsf{DyadicRatCore}^{\uparrow}$ row, while the inspected prefixes are the finite $\mathsf{StreamName}^{\uparrow}$ windows attached to the same request.

The readback obligations are bound by Q, E, L. The regular rational row reads exactly the dyadic-window data, the Real seal is downstream of that row, and the locator handoff reuses the same finite approximation surface.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier has no quotient-stream, selected-limit, trichotomy, open-cover, choice, or ambient completeness coordinate, the listed obligations exhaust the certificate surface. □

Theorem 3614.3 (Dyadic Cauchy approximation RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of an accepted* $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ *packet factors through*

$$D, W \longmapsto Q \longmapsto E, L, H, C, P, N.$$

Thus the consumer first receives the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row and finite* $\mathsf{StreamName}^{\uparrow}$ *windows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal or* $\mathsf{CauchyRealLocator}^{\uparrow}$ *handoff. The route exports no quotient stream equality, selected limit, trichotomy, open-cover compactness, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $K = (D, W, Q, E, L, H, C, P, N)$. The only approximation-facing entry point is the pair D, W, so every tolerance request is read as a dyadic core row together with a finite stream-window row.

The regular rational stage is Q. It reads precisely the windows and dyadic tolerances already displayed by D and W, so no separate stream quotient or selected limit can enter the handoff.

The endpoint stage is E, L, H, C, P, N. The $\mathsf{Real}^{\uparrow}$ seal and $\mathsf{CauchyRealLocator}^{\uparrow}$ handoff are downstream of the regular readback, while transport, replay, provenance, and local naming preserve the same finite route. □

Theorem 3614.4 (Dyadic Cauchy approximation window composition)**.** *Every accepted* DyadicCauchyApproximati *packet composes a tolerance request through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$*-facing seal* E *before any*

locator-facing or consumer-facing output is exported. The composed route is preserved by hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *naming N, and it provides no quotient-stream equality, selected limit, trichotomy, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 3614.1. The tolerance request is recorded first by D, and the only stream data available to it is the finite window row W.

Apply the NameCert obligations theorem Definition 3614.2. It identifies the regular rational readback as Q, the Real-facing endpoint as E, and the locator handoff as L, all sourced by the same finite approximation history.

Apply the RegSeqRat handoff theorem Definition 3614.3. It forces every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer through D, W, Q, E before transport, replay, provenance, and naming rows export the route.

Since the carrier has no quotient-stream, selected-limit, trichotomy, choice, or ambient-completeness coordinate, the displayed window composition is the full consumer-visible path. □

Falsifiable prediction: Every accepted $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or $\mathsf{CauchyRealLocator}^{\uparrow}$ while omitting the DyadicRatCore tolerance row, StreamName window row, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyApproximation}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, finite StreamName windows, RegSeqRat readback, a RealUp seal, a CauchyRealLocator handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3614.1, Definition 3614.2, and Definition 3614.3 seed the finite dyadic approximation route from tolerance rows and stream windows to regular rational readback, Real seal, and locator handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, trichotomy, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance routing, StreamName window exposure, RegSeqRat readback, Real seal nonescape, CauchyRealLocator handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCauchyApproximationUp.

Chapter 3615

A Concrete Naming Certificate for $\mathsf{SequentialCauchyRegularization}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ packet records the finite route by which ordinary Cauchy stream data is regularized into a displayed regular Cauchy name before the terminal Real seal is read. It sits over the displayed threshold discipline of chapter 376, the finite window service of chapter 126, the rational readback of chapter 301, the tolerance ledger of chapter 291, and the Real seal of chapter 38. The packet names a sequential regularization certificate only; it does not identify quotient streams, choose a limit, use countable choice, invoke trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3615.1 (Sequential Cauchy regularization carrier)**.** *A* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source Cauchy stream row. It records the ordinary sequential Cauchy data through finite requests, not through a quotient stream or selected limit.*
2. *T is the displayed regularization threshold row imported from the* $\mathsf{CauchyRegularization}^{\uparrow}$ *route. It records, for each requested precision, the finite stage after which the source windows are regular enough to serve the requested name.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite source windows and the regularized output windows read at the threshold supplied by T.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regularized rational name exposed by T and W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the source-window comparisons, threshold requests, regularized readbacks, and Real-facing output tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the source row, regularization threshold, stream windows, readback, and dyadic tolerance ledger have been displayed.*
7. *H records componentwise* hsame *transport for the source stream, threshold row, window row, readback row, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a sequential regularization request through $X, T, W, R, D, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$, $\mathsf{CauchyRegularization}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite sequential regularization history. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host sequence equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.SequentialCauchyRegularizationUp.SequentialCauchyRegularizationTasteGate single carrier alignment`

Theorem 3615.2 (Sequential Cauchy regularization NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *carrier* $S = (X, T, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegulariza}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the source Cauchy stream, displayed regularization threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming;*
2. *threshold stability under componentwise* hsame *transport of* X, T, W, D*;*
3. *regularized readback compatibility between the threshold row* T*, the window row* W*, the rational readback row* R*, and the tolerance ledger* D*;*
4. $\mathsf{Real}^{\uparrow}$ *seal nonescape through* E*, excluding any read that bypasses* X, T, W, R, D*;*
5. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
6. *refusal of quotient streams, selected limits, countable choice, trichotomy, host sequence equality, and ambient Real completeness as certificate data.*

Proof. Project the carrier of Definition 3615.1. The analytic rows are the source stream X, the regularization threshold T, the window row W, the readback row R, the tolerance ledger D, and the terminal seal E.

The threshold obligation is read from T together with W and D. The row T names the finite stage supplied by the $\mathsf{CauchyRegularization}^{\uparrow}$ route, while W and D record the finite windows and tolerance comparisons checked at that stage.

The readback and seal obligations are read from R and E. The regularized rational name is exposed only through R, and the Real-facing seal is downstream of the same threshold, window, and tolerance data.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, trichotomy, host-equality, or completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 3615.3 (Sequential Cauchy regularization Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ *packet factors through the finite route*

$$X \longmapsto T \longmapsto W, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the consumer first reads the source Cauchy stream, then the displayed regularization threshold from $\mathsf{CauchyRegularization}^{\uparrow}$*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route exports no quotient stream, selected limit, countable-choice selector, trichotomy row, host sequence equality, or ambient completeness theorem.*

Proof. Start with an accepted packet $S = (X, T, W, R, D, E, H, C, P, N)$. The only source coordinate is X, so any sequential Cauchy read begins with the displayed source stream row.

The regularization stage is T, W, D. The threshold row T supplies the finite stage used by the regularization route, and the window and tolerance rows record the exact finite observations made at that stage.

The readback stage is R. It packages the regularized windows as a regular rational name before any Real-facing seal is available.

The terminal stage is E, H, C, P, N. These rows seal, transport, replay, record provenance for, and locally name the same route, and none of them adds quotient, selection, choice, trichotomy, host-equality, or completeness data. □

Falsifiable prediction: Every accepted $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source Cauchy stream row, displayed CauchyRegularization threshold, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCauchyRegularization}^{\uparrow}$ packet is generated from a source Cauchy stream, a displayed CauchyRegularization threshold, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3615.1, Definition 3615.2, and Definition 3615.3 seed the finite sequential regularization route from ordinary Cauchy stream data to regular readback and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCauchyRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host sequence equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold stability, regularized readback compatibility, DyadicRatCore tolerance routing, StreamName exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialCauchyRegularizationUp.

Chapter 3616

A Concrete Naming Certificate for MetaCICDecidableTypingFrontier$^{\uparrow}$

The MetaCICDecidableTypingFrontier$^{\uparrow}$ packet realizes the checker frontier of chapter 9379 as finite NameCert data. It sits beside the decidability witness of chapter 751, the decidability boundary packet of chapter 749, the local diamond witness of chapter 3595, and the self-compilation substrate of section 9235.3. The packet names the boundary between checked typing inference and bounded normal equality; it does not claim unrestricted beta-conversion, global normalization, subject reduction, confluence, or a closedness decider.

Falsifiable prediction: Every accepted MetaCICDecidableTypingFrontier$^{\uparrow}$ packet displays checker rows for `inferTypeCtx`, `checkSameTerm`, `Term.eq`, bounded normal equality, frontier refusal, transport, continuation, provenance, and local naming.

Independence witness: metacicdecidabilitywitness, metacicdecidableboundary, bounded-normal-equality-checker, metacic-local-diamond-witness, metacic-confluence-audit-packet: the decidability witness packages positive checker exactness, the decidability boundary records a broader refusal surface, the bounded-normal-equality checker names the normal-form comparison row, the local-diamond witness names confluence peaks, and the confluence audit packet records join obligations. None is bijective to MetaCICDecidableTypingFrontier$^{\uparrow}$.

Definition 3616.1 (MetaCIC decidable typing frontier carrier)**.** *A* MetaCICDecidableTypingFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the `inferTypeCtx` row. It records the context, represented term, optional inferred type, and the checked soundness and completeness surface available for that option-valued procedure.*
2. *S is the `checkSameTerm` row. It records successful same-term checks as structural equality evidence and refuses to treat failure as negative beta-conversion data.*
3. *Q is the `Term.eq` row. It records Boolean structural equality, reflexivity, true-to-equality, equality-to-true, and the exported correctness theorem for represented terms.*
4. *B is the bounded normal-equality row. It records the fuel, bounded normalizer outputs, and finished-normalization hypotheses required before `decideClosedBetaConv` can be read as a sound comparison.*

5. *R is the frontier refusal row. It records that unrestricted beta-conversion, global normalization, arbitrary closedness decidability, subject reduction, and confluence are not exported by this packet.*
6. *H records componentwise* hsame *transport for the typing checker, same-term checker, structural equality, bounded normal equality, frontier refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decidable-typing-frontier request through I, S, Q, B, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$*, the MetaCIC decidability audit map,* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unrestricted beta-conversion oracle, a global normalization theorem, a subject-reduction theorem, a confluence theorem, host quotient equality of terms, or an ambient completeness principle for all represented syntax.

Lean4 ✓ `BEDC.Derived.MetaCICDecidableTypingFrontierUp.MetaCICDecidableTypingFrontierTasteGate single carrier alignment`

Theorem 3616.2 (MetaCIC decidable typing frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$ *carrier $F = (I, S, Q, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDecidableTypingFront}}$ *surface consists exactly of typing-checker admission through I, same-term checker admission through S, structural equality readback through Q, bounded normal-equality admission through B, frontier refusal through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Later Lean TasteGate work must provide Nontrivial *and* FieldFaithful *instances for this carrier surface. Those instances must witness that the packet cannot collapse to an empty frontier marker and cannot hide a missing typing, same-term, structural-equality, bounded-normal, refusal, transport, replay, provenance, or naming row.*

Proof. Project the carrier of Definition 3616.1. The positive checker rows are I, S, Q: typing inference, same-term checking, and structural equality.

The bounded comparison row is B. It is read only as bounded normal equality with the finished-normalization hypotheses named by the audit map, so it cannot become unrestricted beta-conversion data.

The refusal row R records the open boundary for global normalization, closedness decidability, subject reduction, confluence, and host quotient equality. This row is part of the certificate rather than prose outside the packet.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same frontier route. Since the carrier has no coordinate for the stronger theorems listed above, the displayed rows exhaust the NameCert obligations. □

Theorem 3616.3 (MetaCIC decidable typing frontier nonescape)**.** *Every consumer of an accepted* $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$ *packet factors through the finite route*

$$\texttt{inferTypeCtx},\ \texttt{checkSameTerm},\ \texttt{Term.eq},\ \texttt{decideClosedBetaConv} \longmapsto R, H, C, P, N.$$

The route exports checked typing-inference and structural same-term data, plus bounded normal-equality data under its displayed hypotheses. It exports no unrestricted beta-conversion oracle, no global normalization theorem, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, and no host quotient equality of terms.

Proof. Start with an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$. Rows I, S, Q expose the typing, same-term, and structural equality procedures named by the audit map.

The bounded normal-equality request then passes through B. This row contains the fuel and finished-normalization boundary, so the consumer receives only the checked bounded comparison surface.

The refusal row R is read before the structural rows finish the packet. It prevents the consumer from replacing the bounded route by global beta-conversion, normalization, subject reduction, confluence, closedness decidability, or quotient equality.

The final rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. □

Theorem 3616.4 (MetaCIC decidable typing audit split)**.** *Every audit-map consumer of an accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet first reads* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq` *as the positive typing and structural syntax-equality side. It reads* `decideClosedBetaConv` *only through the bounded normal-equality row and its displayed finished-normalisation hypotheses. The refusal row then blocks any promotion to global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or host quotient equality of terms.*

Proof. Project an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$. The positive checker side is exactly I, S, Q: typing inference through `inferTypeCtx`, same-term checking through `checkSameTerm`, and structural equality through `Term.eq`.

The beta-conversion side is narrower. A consumer reaches `decideClosedBetaConv` only through B, whose stored data includes bounded normalizer outputs, fuel, and the finished-normalisation hypotheses required by the audit map.

The refusal row R is read before the packet is exported through H, C, P, N. It prevents the bounded comparison row from being reinterpreted as global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality.

Thus the packet preserves the audit-map split: positive typing and syntax equality are read directly from checker rows, while beta conversion remains bounded and hypothesis-indexed. □

Theorem 3616.5 (MetaCIC decidable typing coverage frontier)**.** *Every accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet exports exactly the positive checker rows* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq`*. It exports* `decideClosedBetaConv` *only through the bounded normal-equality row carrying displayed fuel, bounded normalizer outputs, and finished-normalization hypotheses. Therefore any consumer claiming global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality must leave this packet.*

Proof. Project an accepted carrier $F = (I, S, Q, B, R, H, C, P, N)$ from Definition 3616.1. The projected positive checker rows are exactly I, S, Q, corresponding to `inferTypeCtx`, `checkSameTerm`, and `Term.eq`.

Use Definition 3616.2. It identifies the local NameCert obligations with the displayed checker rows, bounded normal-equality row, refusal row, and structural rows, with no hidden coordinate for stronger metatheory.

Apply Definition 3616.4. The only route to `decideClosedBetaConv` passes through B, where the fuel, bounded outputs, and finished-normalization hypotheses are part of the exported row.

Finally apply Definition 3616.3. The refusal row R blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and quotient equality from being read as packet exports. □

Theorem 3616.6 (MetaCIC decidable typing frontier positive checker exactness)**.** *An accepted* MetaCICDecidableTypingFrontier$^{\uparrow}$ *packet exports positive checker exactness only for* `inferTypeCtx`*,* `checkSameTerm`*, and* `Term.eq`*. Any read of* `decideClosedBetaConv` *must pass through the bounded normal-equality row with displayed fuel, bounded normalizer outputs, and finished-normalization hypotheses, and the refusal row blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and quotient equality.*

Proof. Project the carrier $F = (I, S, Q, B, R, H, C, P, N)$ of Definition 3616.1. The positive checker rows are exactly I, S, Q, so exactness for typing inference, same-term checking, and structural equality is confined to those rows.

Compose the NameCert obligations of Definition 3616.2 with the audit split of Definition 3616.4. They keep the bounded beta-conversion read in B, where fuel and finished-normalization hypotheses are displayed before any comparison result is exported.

Finally use the coverage frontier of Definition 3616.5. It separates the positive checker rows from the bounded normal-equality row and reads the refusal row R before transport, replay, provenance, and local naming finish the packet.

Because the carrier has no row for global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality, those claims cannot be exported as positive checker exactness. □

Closure status for NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$.

Constructive story: A MetaCICDecidableTypingFrontier$^{\uparrow}$ packet is generated from inferTypeCtx typing rows, checkSameTerm rows, Term.eq structural equality rows, bounded normal-equality rows, frontier refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 3616.1, Definition 3616.2, Definition 3616.3, and Definition 3616.4 seed the finite decidable-typing frontier over checked typing inference, structural equality, bounded normal equality, and explicit refusal rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICDecidableTypingFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted beta-conversion, global normalization, subject reduction, confluence, arbitrary closedness decidability, host quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typing-checker exactness, same-term exactness, structural equality readback, bounded-normal comparison exactness, frontier refusal, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDecidableTypingFrontierUp.

Chapter 3617

A Concrete Naming Certificate for $\mathsf{DenseEmbedding}^{\uparrow}$

Following Kelley and Bourbaki on dense embeddings, the $\mathsf{DenseEmbedding}^{\uparrow}$ packet records the finite BEDC surface by which a source metric carrier is read inside a target metric completion through a dense image-window and an approximation ledger. It sits before the specialized metric-completion embedding packet of chapter 3351, and it supplies a canonical source route for $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ consumers in chapter 38, chapter 301, chapter 126, chapter 291, and chapter 3579. The packet names dense-image approximation as finite certificate data; it does not import a host embedding theorem, quotient representatives, selected limits, countable choice, or ambient completion completeness.

Definition 3617.1 (Dense embedding carrier)**.** *A* $\mathsf{DenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, T, C, I, W, A, E, H, K, P, N)$$

with the following displayed rows:

1. *S is the source metric row, including the finite carrier, distance readings, and source-side comparison windows.*
2. *T is the target metric row, read only through displayed target-distance comparisons.*
3. *C is the target completion row, carrying the completion-facing names used by the embedding read.*
4. *I is the source-to-target image row. It records the embedded image of inspected source points without choosing quotient representatives or host limits.*
5. *W is the dense image-window row. Each density request opens a finite target window, a source witness window, and the target-distance tolerance being tested.*
6. *A is the approximation ledger. It records that the witness in S reaches the target request through I, W, T, C at the displayed tolerance.*
7. *E is the completion handoff row exposed to completion-facing consumers after S, T, C, I, W, A have already been read.*
8. *H stores componentwise* hsame *transport for the source metric, target metric, completion, image, dense window, approximation ledger, handoff, continuation, provenance, and local naming rows.*

9. *K stores displayed* Cont *replay from source observations to image observations, from image observations to dense windows, and from dense windows to the completion handoff row.*
10. *P is the* Pkg *provenance over* $\mathsf{DenseEmbedding}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, C, I, W, A, E, H, K, P, N$. It has no coordinate for host topological density, quotient completion equality, a selected limit, a global inverse, countable choice, or ambient completeness of the target completion.

Lean4 ✓ `BEDC.Derived.DenseEmbeddingUp.DenseEmbeddingTasteGate single carrier alignment`

Theorem 3617.2 (Dense embedding NameCert obligations)**.** *For every accepted* $\mathsf{DenseEmbedding}^{\uparrow}$ *carrier $D = (S, T, C, I, W, A, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the source metric, target metric, target completion, source-to-target image, dense image-window, approximation ledger, completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. Every density read must first expose S, T, C, I, W, A before the handoff row E can be consumed.*

Proof. Project the carrier of Definition 3617.1. The source metric, target metric, completion row, image row, dense window, and approximation ledger are exactly S, T, C, I, W, A, so a density request has no other analytic coordinates to inspect.

The handoff row E is downstream of those six rows. Hence completion-facing consumers first see the displayed source witness, its image, the finite target window, and the approximation ledger before receiving any completion data.

The structural rows H, K, P, N only transport, replay, source, and name that same finite route. Since the carrier excludes host density, quotient representatives, selected limits, countable choice, global inverses, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3617.3 (Dense embedding completion handoff)**.** *Every* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenseEmbedding}^{\uparrow}$ *packet factors through the finite route*

$$S, T, C, I \longmapsto W \longmapsto A \longmapsto E$$

together with H, K, P, N. The route may be read through the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *windows named by the packet, but it exports no selected completion limit, quotient representative, ambient density theorem, countable-choice selector, or host completeness principle.*

Proof. Start with the NameCert obligations of Definition 3617.2. They place the source metric row, target metric row, target completion row, and image row before any dense-window or approximation-ledger read is available.

The dense image-window W opens only the finite source and target observations requested by the consumer. The approximation ledger A then records that this displayed source witness reaches the requested target window through the image row I at the stated tolerance.

Only after that ledger is read does the completion handoff row E become available. The rows H, K, P, N preserve and name the same route, so specialized completion, real, and Cauchy-completeness consumers receive finite dense-embedding data rather than host completion structure. □

Falsifiable prediction: Every accepted $\mathsf{DenseEmbedding}^{\uparrow}$ completion-facing read factors through a source metric row, target metric row, target completion row, source-to-target image row, dense image-window, approximation ledger, completion handoff row, transport, continuation replay, provenance, and local naming.

Independence witness: metric-completion-dense-embedding, real, regseqrat, streamname, dyadicratcore, cauchy-completeness-witness: $\mathsf{DenseEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it carries the canonical dense image-window and approximation ledger before those siblings consume the completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{DenseEmbedding}^{\uparrow}$ packet is generated from a source metric row, target metric row, target completion row, source-to-target image row, dense image-window, approximation ledger, completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3617.1, Definition 3617.2, and Definition 3617.3 seed the canonical dense-embedding carrier and completion handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topological density, quotient completion equality, selected limits, global inverses, countable choice, ambient completion completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, dense image-window exactness, approximation-ledger handoff, completion nonescape, and TasteGate field faithfulness for BEDC.Derived.DenseEmbeddingUp.

Chapter 3618

A Concrete Naming Certificate for $\mathsf{UniformCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completions, the $\mathsf{UniformCauchyCompletion}^{\uparrow}$ packet records the finite uniform-Cauchy handoff used by Phase Real and Completion consumers before any completed object is exposed. It sits over the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy-sequence space surface of chapter 406, the terminal Real seal of chapter 38, and the Bishop completion route of chapter 3484. The packet is a finite NameCert surface for uniform Cauchy families; it is not a quotient completion, selected limit, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3618.1 (Uniform Cauchy completion carrier)**.** *A* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the uniform Cauchy family row, recording the inspected finite family, common-window requests, and uniform Cauchy tolerance responses.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing only the windows used by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on those same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that discharges every displayed uniform Cauchy request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *and completion seal row reached only after F, S, R, D have been displayed, with the sibling handoff to* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *and* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *recorded as part of the same finite route.*
6. *H records componentwise* hsame *transport for the family, stream-window, regular-readback, dyadic-ledger, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through F, S, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{BishopCompletionTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite uniform-Cauchy completion handoff. It has no coordinate for quotient completions, selected limits, countable choice, host real equality, or ambient Real completeness.

Theorem 3618.2 (Uniform Cauchy completion NameCert obligations)**.** *For every accepted* UniformCauchyComplet *carrier* $U = (F, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform Cauchy family, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real and completion seal, transport, replay, provenance, and local naming rows;*
2. *uniform Cauchy family stability, requiring every requested comparison to stay on the displayed family row* F *and the finite windows exposed by* S*;*
3. *regular rational readback exactness, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to read only the same windows and budgets already named by* F *and* S*;*
4. *dyadic tolerance exactness, requiring the row* D *to discharge each uniform Cauchy request before the seal row is visible;*
5. *Real and completion seal nonescape, requiring the row* E *to be reached only through* F, S, R, D *and the sibling* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *and* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *routes;*
6. *componentwise* hsame *transport and displayed* Cont *replay through* H *and* C*;*
7. Pkg *provenance and local naming through* P *and* N*;*
8. *refusal of quotient completions, selected limits, countable choice, host real equality, and ambient Real completeness as certificate data.*

Lean4 ✓ `BEDC.Derived.UniformCauchyCompletionUp.UniformCauchyCompletionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3618.1. The visible coordinates are $F, S, R, D, E, H, C, P, N$, so every local obligation must be read from those rows.

The uniform Cauchy family is fixed by F and its finite observations are fixed by S. The regular rational readback R and tolerance ledger D are attached to those same windows, so a completion-facing request cannot exchange them for an external sequence, host equality, or selected limit representative.

The seal row is E. It is downstream of the family, StreamName, RegSeqRat, and DyadicRatCore rows, and its sibling route names the $\mathsf{CauchySequenceSpace}^{\uparrow}$ and $\mathsf{BishopCompletionTheorem}^{\uparrow}$ handoffs without adding quotient or choice data.

Finally H, C, P, N transport, replay, record provenance for, and locally name the same handoff. Since no quotient completion, selected limit, countable choice, host real equality, or ambient Real completeness coordinate appears in the carrier, the listed rows exhaust the certificate surface. □

Theorem 3618.3 (Uniform Cauchy completion finite handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *packet factors through the finite route*

$$F \longmapsto S \longmapsto R \longmapsto D \longmapsto E, H, C, P, N.$$

Thus a consumer first reads the uniform Cauchy family, then the finite StreamName windows, then the RegSeqRat readback, then the DyadicRatCore tolerance ledger, and only then the Real and

completion seal. The route exports no quotient completion, selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $U = (F, S, R, D, E, H, C, P, N)$. The only family coordinate is F, and the only finite observation coordinate is S.

The readback and tolerance rows are R and D. They keep the completion-facing request on the same regular rational windows and dyadic tolerance ledger before any terminal seal can be read.

The endpoint row is E, with structural support from H, C, P, N. These rows hand the same finite packet to $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, and $\mathsf{BishopCompletionTheorem}^{\uparrow}$ consumers without adding quotient, selection, choice, host-equality, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing boundary while omitting the uniform Cauchy family row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, seal row, transport, replay, provenance, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCompletion}^{\uparrow}$ packet is generated from a uniform Cauchy family row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real and completion seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3618.1, Definition 3618.2, and Definition 3618.3 seed the finite uniform-Cauchy route from stream windows and regular rational readback to Real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform Cauchy family stability, finite-window readback, dyadic tolerance exactness, Real and completion seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyCompletionUp.

Chapter 3619

A Concrete Naming Certificate for $\mathsf{ArchimedeanCompletion}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of Archimedean ordered fields and regular Cauchy completions, the $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet records the finite naming surface that carries a dyadic ordered-field source into the Real and Completion route. It sits after the dyadic ordered core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the ordered cut comparison of chapter 3517, and the completion witness of chapter 3579. The packet names an Archimedean completion interface; it is not quotient equality of Cauchy streams, a selected host limit, a trichotomy oracle, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3619.1 (Archimedean completion carrier)**.** *An* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, K, R, D, O, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the dyadic ordered-field source row, including the finite addition, multiplication, order, unit, and sign-comparison data read from the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger.*
2. *K is the Archimedean cofinality row: every displayed field element is bounded by a named dyadic scale already present in the same packet.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row generated from the source and cofinality data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare finite approximants and dyadic scales.*
5. *O is the order-density comparison row linking the regular readback to the order-completion surface of chapter 3517.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, K, R, D, O have been displayed.*
7. *H records componentwise* hsame *transport for the ordered-field source, cofinality witness, regular readback, dyadic ledger, order comparison, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an Archimedean-completion request through F, K, R, D, O, E.*
9. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopOrderCompletion}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, K, R, D, O, E, H, C, P, N$ and the displayed replay path. It contains no coordinate for quotient stream equality, selected representatives, host-limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArchimedeanCompletionUp.TasteGate.ArchimedeanCompletionTasteGate single carrier alignment`

Theorem 3619.2 (Archimedean completion NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier $A = (F, K, R, D, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$ *surface consists exactly of the following five rows:*

1. *carrier admission for the finite dyadic ordered-field packet F together with the Archimedean cofinality witness K;*
2. *regular-Cauchy readback through R and dyadic tolerance accounting through D;*
3. *order-density comparison through O, with no independent located-cut or trichotomy oracle;*
4. *terminal* $\mathsf{Real}^{\uparrow}$ *handoff through E, reached only after the source, cofinality, readback, ledger, and order-comparison rows have been displayed;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No local obligation can be discharged by quotient Cauchy equality, selected host limits, countable choice, a detached order-completion theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3619.1. Its analytic rows are exactly the dyadic ordered-field source, the Archimedean cofinality witness, the regular-Cauchy readback, the dyadic tolerance ledger, the order-density comparison, and the terminal Real seal.

Carrier admission is the first row: F supplies the finite ordered-field operations and comparisons, while K supplies only the displayed cofinal dyadic scales. The readback row R and ledger row D then account for every regular-Cauchy and tolerance datum consumed by the packet.

The order-facing route is forced through O. It compares the same finite readback with the Bishop order-completion surface, but it cannot introduce a located-cut selector, trichotomy oracle, or host order equality because no such row belongs to the carrier.

Finally E, H, C, P, N route the terminal Real handoff and structural support. Since the carrier has no quotient stream equality, selected limit, countable-choice schedule, detached completion theorem, or ambient-completeness coordinate, none of those objects belongs to the NameCert obligation surface. □

Theorem 3619.3 (Archimedean completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{ArchimedeanCompletion}^{\uparrow}$ *carrier factors through the displayed dyadic, regular, and order-completion rows*

$$F, K \longmapsto R, D \longmapsto O \longmapsto E \longmapsto H, C, P, N.$$

The dyadic ordered-field source and Archimedean cofinality rows determine the regular-Cauchy readback; the dyadic ledger and order-density row compare that readback with the Bishop order-completion interface; the terminal Real seal is available only after those rows have been replayed.

Consequently a Real-facing consumer receives a finite NameCert route from dyadic ordered-field data to regular Cauchy and order-completion data. It receives no quotient stream equality, selected representative, host-limit operator, trichotomy oracle, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a consumer that reaches the terminal Real row. In the carrier of Definition 3619.1, the only Real-facing coordinate is E, and E is placed after F, K, R, D, O.

The consumer must therefore read the dyadic ordered-field source and cofinality rows before any regular-Cauchy data is available. The readback row R and ledger row D expose the finite approximation route, while the order-density row O aligns that route with the Bishop order-completion packet.

The structural rows H, C, P, N then transport, replay, source, and locally certify the same route. They preserve the displayed handoff, but they cannot add quotient equality, selected limits, trichotomy, countable choice, detached completion, or ambient Real completeness because the accepted packet has no such coordinate. □

Falsifiable prediction: Every accepted $\mathsf{ArchimedeanCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ handoff while omitting the dyadic ordered-field source, Archimedean cofinality row, regular-Cauchy readback, dyadic tolerance ledger, order-density comparison, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: dyadicratcore, regseqrat, real, bishop-order-completion, cauchy-completeness-witness: no listed sibling supplies the whole $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet, because each sibling contributes only dyadic arithmetic, regular readback, Real sealing, order comparison, or completion witnessing while this chapter binds all of them with cofinality, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCompletion}^{\uparrow}$ packet is generated from a dyadic ordered-field source row, an Archimedean cofinality row, RegSeqRat readback, a DyadicRatCore tolerance ledger, a BishopOrderCompletion order-density comparison row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3619.1, Definition 3619.2, and Definition 3619.3 seed the finite Archimedean completion route from dyadic ordered-field source data through regular-Cauchy readback, order-density comparison, Real sealing, transport, replay, provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ArchimedeanCompletionUp.TasteGate.ArchimedeanCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy equality, selected host limits, trichotomy, countable choice, detached order completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinality routing, regular-Cauchy readback, dyadic ledger exactness, order-density comparison, Real handoff factorization, transport, replay, provenance, and local naming for BEDC.Derived.ArchimedeanCompletionUp.

Chapter 3620

A Concrete Naming Certificate for $\mathsf{RieszSpace}^{\uparrow}$

Following Riesz and Kantorovich, the $\mathsf{RieszSpace}^{\uparrow}$ packet records the finite BEDC naming surface for a real vector lattice with a displayed positive cone and norm-facing compatibility rows. It sits over the algebraic vector-space carrier of chapter 47, the join-meet order rows of chapter 55, the norm comparison surface of chapter 89, the positive scalar endpoint discipline of chapter 1587, and the real scalar seal of chapter 38. The packet names the BEDC-facing ordered-functional-analysis object; it is not a host order primitive, a supremum completion theorem, a Banach-lattice completeness claim, or a choice of lattice decompositions outside the displayed finite ledger.

Definition 3620.1 (RieszSpace carrier)**.** *A* $\mathsf{RieszSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (V, L, N, P, H, C, S)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming the carried vector endpoint, real scalar endpoint, zero vector, addition, additive inverse, scalar action, and finite vector-space law ledger.*
2. *L is the lattice row, recording join and meet operations on the same vector endpoint, the induced order-comparison rows, and the finite absorption, commutativity, associativity, and distributivity witnesses used by the local classifier.*
3. *N is the norm or seminorm compatibility row, read through* $\mathsf{Norm}^{\uparrow}$*, saying that the displayed norm endpoint respects the vector rows and the lattice decomposition rows already carried by V and L.*
4. *P is the positive-cone decomposition row, read through* $\mathsf{PositiveReal}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording positive and negative part formation, cone membership, scalar positivity, and the local identity route tying a vector to the difference of its displayed positive pieces.*
5. *H records componentwise* hsame *transport for the vector-space rows, lattice join and meet rows, norm or seminorm rows, positive-cone rows, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from a Riesz-space request through V, L, N, P, H back to the same finite ordered vector-lattice surface.*
7. *S is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$ *row over* $\mathsf{RieszSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

The classifier compares accepted packets only through V, L, N, P, H, C, S. It has no coordinate for an ambient order relation, a host supremum construction, a selected order completion, Banach-lattice completeness, pointwise function models, or a quotient by order ideals.

Lean4 ✓ `BEDC.Derived.RieszSpaceUp.RieszSpaceUpTasteGate single carrier alignment`

Theorem 3620.2 (RieszSpace NameCert obligations)**.** *For every accepted* $\mathsf{RieszSpace}^{\uparrow}$ *carrier* $R = (V, L, N, P, H, C, S)$*, the local* $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, vector-space row compatibility, lattice join and meet exactness, norm or seminorm compatibility, positive-cone decomposition, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A downstream consumer may read the common vector endpoint, the real scalar endpoint, the join and meet operations, the norm-facing comparison row, and the positive-cone ledger, but it receives no host order primitive, no chosen order completion, no Banach-lattice completeness theorem, and no pointwise function representation.*

Proof. Carrier admission is projection from Definition 3620.1. The packet displays exactly the vector-space row V, the lattice row L, the norm or seminorm compatibility row N, the positive-cone decomposition row P, and the structural rows H, C, S.

The vector-space obligations are inherited from the carried $\mathsf{VecSpace}^{\uparrow}$ row, while the lattice obligations are read only through the displayed join and meet ledger. Thus the same vector endpoint is the domain of addition, scalar multiplication, join, meet, and order comparison.

The norm-facing and positive-cone obligations are finite rows. The row N records compatibility with the vector and lattice decompositions, and P records positive and negative part formation using the $\mathsf{PositiveReal}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar seals.

Finally H, C, S transport, replay, source, and locally certify only the displayed coordinates. Since the carrier has no ambient order coordinate, no completion selector, no Banach-lattice completeness row, and no pointwise function model, none of those objects can be exported by the local certificate. □

Theorem 3620.3 (RieszSpace lattice-norm compatibility)**.** *If two accepted* $\mathsf{RieszSpace}^{\uparrow}$ *packets are related by the classifier, then their vector-space rows, lattice join and meet rows, norm or seminorm compatibility rows, and positive-cone decomposition rows are related by the componentwise* hsame *transport recorded in H. Consequently every downstream norm-facing or order-facing read factors through the displayed vector-lattice ledger and never through a host order equality or a hidden normed-lattice quotient.*

Proof. Unfold the classifier of Definition 3620.1. Its comparison coordinates are exactly the vector-space row, lattice join and meet row, norm or seminorm compatibility row, positive-cone row, replay row, provenance row, and local naming row.

The vector component transports through the $\mathsf{VecSpace}^{\uparrow}$ comparison, and the lattice component transports through the displayed join and meet comparisons. The norm or seminorm row then moves along the same transported vector and lattice data.

The positive-cone row is carried componentwise as well: positive part, negative part, cone membership, scalar positivity, and reconstruction of the original vector all remain inside the visible ledger. Hence every norm-facing or order-facing consumer reads a transported finite certificate, not a host order equality or a hidden quotient construction. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RieszSpace}^{\uparrow}$ packet is generated from vector-space rows, lattice join and meet rows, norm or seminorm compatibility rows, positive-cone decomposition data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RieszSpace}\uparrow}$, seed)

Scope closed: Definition 3620.1, Definition 3620.2, and Definition 3620.3 seed the finite ordered vector-lattice certificate over VecSpaceUp, LatticeUp, NormUp, PositiveRealUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RieszSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host order primitives, supremum completion, Banach-lattice completeness, pointwise function representation, quotient by order ideals, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vector-space compatibility, lattice join and meet exactness, norm or seminorm compatibility, positive-cone decomposition, transport, replay, provenance, and local naming for BEDC.Derived.RieszSpaceUp.

Chapter 3621

A Concrete Naming Certificate for ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$

Following Arzela and Ascoli, the ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ packet names the finite-window bridge between the diagonal extraction surface of chapter 2219 and the compact family interface of chapter 412. It consumes the equicontinuity ledger of chapter 413, the compact source rows of chapter 129, the target completion rows of chapter 128, the finite window names of chapter 126, the rational readback rows of chapter 301, and the pointwise seal of chapter 38. The chapter records a finite BEDC NameCert bridge; it does not assert an ambient compact-open topology, a host sequence-space compactness theorem, or a choice-selected subsequence.

Definition 3621.1 (Arzela-Ascoli finite-window diagonal bridge)**.** *An* ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, E, Y, D, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *K records the compact source net rows read from* CompactMetric$^{\uparrow}$*, including the finite probe windows and coverage witnesses consumed by the diagonal route;*
2. *E records the* EquicontinuityFamily$^{\uparrow}$ *ledger that binds every requested target tolerance to a displayed source-radius row uniformly over the family ledger;*
3. *Y records the* CompleteMetric$^{\uparrow}$ *target surface through which finite-window target reads are sealed;*
4. *D records the accepted* AscoliDiagonal$^{\uparrow}$ *carrier whose diagonal extraction rows are being exposed to the Arzela-Ascoli consumer;*
5. *W records the finite diagonal* StreamName$^{\uparrow}$ *windows, with each window naming the source-net points, family indices, and target reads currently inspected;*
6. *R records* RegSeqRat$^{\uparrow}$ *readback rows for the rational tolerances and target-distance witnesses visible on W;*
7. *S records the* Real$^{\uparrow}$ *seal that repacks each finite readback into the target-side pointwise boundary;*
8. *H records componentwise* hsame *transport for K, E, Y, D, W, R, S and the structural rows;*
9. *C records displayed* Cont *replay through AscoliDiagonal, EquicontinuityFamily, CompactMetric, CompleteMetric, StreamName, RegSeqRat, Real, and local NameCert reads;*

10. *P is the* Pkg *provenance over* ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$, AscoliDiagonal$^{\uparrow}$, ArzelaAscoli$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^{\uparrow}}$ *row.*

The carrier exports only the displayed finite-window diagonal bridge data. It has no coordinate for an unbounded subsequence selector, an ambient compact-open topology, or an open-cover compactness principle.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteWindowDiagonalBridgeUp.ArzelaAscoliFiniteWindowDiagonalBridgeTas`
`single carrier alignment`

Theorem 3621.2 (Finite-window diagonal bridge route)**.** *Every bridge read from an accepted* ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ *carrier factors through the finite route*

$$\text{AscoliDiagonal}^{\uparrow} \longrightarrow (\text{EquicontinuityFamily}^{\uparrow}, \text{CompactMetric}^{\uparrow}) \longrightarrow \text{CompleteMetric}^{\uparrow} \longrightarrow \text{StreamName}^{\uparrow} \longrightarrow \text{RegSeqRa}$$

The route first reads the accepted AscoliDiagonal carrier, then the compact source nets and equicontinuity ledger, then the complete target surface, then the finite diagonal StreamName windows, then RegSeqRat readback and the Real seal. The exported surface is a finite NameCert bridge and no choice-selected subsequence or host function-space compactness object.

Proof. Let $B = (K, E, Y, D, W, R, S, H, C, P, N)$ be an accepted carrier. Project D first. Its accepted AscoliDiagonal$^{\uparrow}$ row fixes the diagonal extraction surface before any Arzela-Ascoli consumer can inspect a target read.

Next project K and E. The compact source row supplies finite net windows, and the equicontinuity-family row supplies the shared tolerance-to-radius ledger used by those windows.

Project Y to read the complete-metric target boundary. The bridge can then inspect only the finite StreamName$^{\uparrow}$ windows recorded in W, so each target read is tied to displayed source-net points and displayed family indices.

The readback row R records the corresponding RegSeqRat$^{\uparrow}$ tolerance and distance data, and the seal row S repacks the finite readback as the visible Real$^{\uparrow}$ boundary. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve exactly those rows.

Since the carrier contains no unbounded subsequence selector, no ambient compact-open topology, and no host function-space compactness coordinate, every bridge read factors through the displayed finite route and exports nothing beyond it. □

Closure status for NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^{\uparrow}}$ **.**

Constructive story: An ArzelaAscoliFiniteWindowDiagonalBridge$^{\uparrow}$ packet is generated from an accepted AscoliDiagonal carrier, compact source-net rows, equicontinuity-family rows, complete-metric target rows, finite StreamName diagonal windows, RegSeqRat readback rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^{\uparrow}}$, seed)

Scope closed: Definition 3621.1 and Definition 3621.2 seed the finite-window diagonal bridge from AscoliDiagonal data to the Arzela-Ascoli compact family interface.

Formal verification: FormalStatus(NameCert$_{\text{ArzelaAscoliFiniteWindowDiagonalBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a choice-selected subsequence, ambient compact-open topology, open-cover compactness, host sequence-space compactness, or a standard bridge.

Upgrade path: Obligation closure requires Lean carrier admission, classifier transport, finite-window route extraction, readback sealing, and non-escape rows for BEDC.Derived.ArzelaAscoliFiniteWindowDiagonalBridgeUp.

Chapter 3622

A Concrete Naming Certificate for $\mathsf{BishopCauchySequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchySequence}^{\uparrow}$ packet records the textbook sequence-level Cauchy surface that mediates between regular rational names and the sealed Real object. It sits after the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational name packet of chapter 301, and the uniqueness route of chapter 3390, while feeding the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite carrier for Cauchy sequence evidence; it does not assert quotient streams, selected limits, countable choice, or ambient Real completeness.

Definition 3622.1 (Bishop Cauchy sequence carrier)**.** *A* $\mathsf{BishopCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, D, X, U, R, H, C, P, N)$$

with the following visible rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposing the requested rational sequence entries;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, comparison budget, and rational-distance rows used by the Cauchy test;*
3. *X is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row that certifies the sequence entries respect the displayed tolerance discipline;*
4. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *comparison row used only to compare accepted regular names at the same displayed tolerance surface;*
5. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, D, X, U have been displayed;*
6. *H stores componentwise* hsame *transport for the stream window, dyadic ledger, regularity row, uniqueness comparison, seal row, continuation route, package row, and local naming row;*
7. *C records displayed* Cont *replay of the route*

$$S, D \longmapsto X \longmapsto U \longmapsto R;$$

8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $S, D, X, U, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, ambient Real completeness, or a host equality oracle for Real values.

Lean4 ✓ `BEDC.Derived.BishopCauchySequenceUp.BishopCauchySequenceUpTasteGate single carrier alignment`

Theorem 3622.2 (Bishop Cauchy sequence NameCert obligations)**.** *For an accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *carrier*

$$B = (S, D, X, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row,* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *comparison row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under the same displayed rows; tolerance exactness through D; regularity handoff through X; uniqueness comparison through U; and Real-seal boundary control through R.*

Proof. Carrier admission is projection from Definition 3622.1. The projection exposes exactly the stream window, dyadic tolerance ledger, regular rational sequence evidence, uniqueness comparison, Real seal, and structural rows.

Classifier stability follows row by row because the classifier has no coordinate beyond $S, D, X, U, R, H, C, P, N$. Tolerance exactness is the projection of D, and regularity handoff is the projection of X with the same displayed tolerance row.

The uniqueness row U compares only accepted regular names at the displayed surface. The Real seal row R is downstream from S, D, X, U, so no consumer can read a sealed Real object without first passing through the finite Bishop Cauchy sequence packet. □

Theorem 3622.3 (Bishop Cauchy sequence regularity handoff)**.** *Every accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *packet hands its stream-window and tolerance data to a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row before the uniqueness comparison or Real seal is available. The handoff route is the finite chain*

$$S, D \longmapsto X \longmapsto U \longmapsto R,$$

preserved by H, C, P, N and not by quotient stream equality, a selected limiting object, countable choice, or an ambient completeness theorem.

Proof. Project the carrier and read the continuation route C. The route opens the finite stream window S and dyadic tolerance ledger D before the regularity row X.

The comparison row U is downstream from the regularity row. Since U is a $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ row, it compares accepted regular names through the displayed sequence and tolerance data rather than through a quotient stream.

The Real seal row R occurs at the end of the displayed route. The transport, provenance, and naming rows preserve that order, while no carrier coordinate supplies a selected limit, countable-choice principle, or ambient completeness argument. □

Theorem 3622.4 (Bishop Cauchy sequence Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{BishopCauchySequence}^{\uparrow}$ *packet factors through the finite rows S, D, X, U before it reaches the seal row R. The packet therefore exposes a sequence-level route from rational windows and dyadic tolerances to the Real seal, while withholding quotient streams, selected limits, countable choice, and ambient Real completeness.*

Proof. Start from a Real-facing read and project the carrier. The only seal row is R, and the continuation route places it after the stream window, tolerance ledger, regularity row, and uniqueness comparison.

Use Definition 3622.3 to bind the regularity row to the same stream and dyadic tolerance data. Then use the uniqueness row U only as a displayed comparison between accepted regular names.

The coordinate list of Definition 3622.1 exhausts the packet. Since it contains no quotient stream, selected limit, countable choice, or ambient-completeness coordinate, no consumer can derive such data from the Real seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchySequence}^{\uparrow}$ packet is generated from finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat regularity evidence, a RegularCauchyUniqueness comparison row, a Real seal, transport data, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3622.1, Definition 3622.2, Definition 3622.3, and Definition 3622.4 name the L10 sequence-level carrier between regular rational streams and the Real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, tolerance exactness, regularity handoff, uniqueness comparison, and Real-seal boundary rows for the displayed Bishop Cauchy sequence packet.

Chapter 3623

A Concrete Naming Certificate for $\mathsf{MeirKeelerContraction}^{\uparrow}$

Following Meir and Keeler's fixed point theorem and Bishop–Bridges constructive metric analysis, the $\mathsf{MeirKeelerContraction}^{\uparrow}$ packet records the finite epsilon-to-delta contraction ledger that sits between ordinary contraction data and completion-facing fixed-point consumers. It reads the metric carrier of chapter 57, the fixed-point route of chapter 3607, the iteration ledger of chapter 2383, the endpoint-admission surface of chapter 128, and the Banach fixed-point packet of chapter 396. It also sits beside the Lipschitz and nonexpansive surfaces of chapter 410, chapter 2328, and chapter 2105, while refusing to replace the Meir–Keeler ledger by a single host Lipschitz constant.

Definition 3623.1 (Meir–Keeler contraction carrier)**.** *A* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, d, T, E, I, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{Met}^{\uparrow}$ *carrier row for the source points.*
2. *d is the metric-distance readback row used by all comparison requests.*
3. *T is the self-map graph row for the candidate contraction.*
4. *E is the Meir–Keeler epsilon-to-delta contraction ledger: for each displayed positive tolerance request it stores a displayed response window forcing a strictly smaller image distance whenever the source distance lies in the named annulus.*
5. *I is the fixed-point iteration row, recording the finite Picard-style orbit segments and adjacent comparison reads used by downstream consumers.*
6. *L is the Cauchy handoff ledger, listing the finite tail requests that the Meir–Keeler row transfers to the complete-metric endpoint surface.*
7. *H records componentwise* hsame *transport for metric rows, distance reads, self-map graph rows, epsilon-to-delta ledger rows, iteration rows, Cauchy handoff rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a contraction request through $X, d, T, E, I, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MeirKeelerContraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContractionPrinciple}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, d, T, E, I, L, H, C, P, N$. It has no coordinate for a global Lipschitz constant, a selected endpoint, quotient completion equality, host function extensionality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MeirKeelerContractionUp.MeirKeelerContractionTasteGate single carrier alignment`

Theorem 3623.2 (Meir–Keeler contraction NameCert obligations)**.** *For an accepted* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier $M = (X, d, T, E, I, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$ *surface consists exactly of metric-carrier admission, distance-readback visibility, self-map graph admission, epsilon-to-delta ledger exactness, fixed-point iteration visibility, Cauchy handoff visibility, componentwise transport, displayed replay, provenance, and local naming. These obligations bind the Meir–Keeler route without adding a global contraction constant, endpoint selection, quotient completion equality, host fixed-point equality, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 3623.1. The displayed rows $X, d, T, E, I, L, H, C, P, N$ are the whole local surface, so every obligation must be one of their admissions, transports, or replay checks.

The contraction content is carried by T and E. The self-map row fixes the map being iterated, while the epsilon-to-delta row supplies the strict Meir–Keeler comparison at each displayed tolerance window. No single Lipschitz bound is extracted from this row.

The iteration and completion-facing content is carried by I and L. These rows expose finite orbit segments and finite tail requests before any complete-metric endpoint can be consumed by another chapter.

Finally H, C, P, N keep the same rows stable under classifier transport, replay, provenance, and local naming. Since the excluded host objects are not carrier coordinates, the NameCert surface cannot export them. □

Theorem 3623.3 (Meir–Keeler contraction Cauchy handoff)**.** *Every fixed-point or completion-facing read of an accepted* $\mathsf{MeirKeelerContraction}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow},\ d,\ T,\ E,\ \mathsf{FixedPointIteration}^{\uparrow},\ L,\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The read uses the Meir–Keeler epsilon-to-delta ledger to feed finite iteration-tail requests into the complete-metric endpoint surface. It never exposes a selected limit, a quotient representative, a host global Lipschitz constant, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the metric and map rows. A consumer first reads X, d, T, so the source carrier, comparison relation, and self-map graph are fixed before the Meir–Keeler ledger is inspected.

Next consume E. The epsilon-to-delta ledger answers each displayed tolerance request by a finite annulus response, which is the contraction data available to the iteration route. This is weaker than a global Lipschitz constant but strong enough for the displayed tail comparisons.

Then read I and L. The iteration row supplies the finite orbit comparisons, and the Cauchy handoff row packages the resulting tail requests for the $\mathsf{CompleteMetric}^{\uparrow}$ endpoint surface.

The structural rows H, C, P, N transport and replay the same route. Because no endpoint selector, quotient equality, host Lipschitz constant, or ambient completeness row occurs in the carrier, every downstream read remains inside the finite handoff just described. □

Falsifiable prediction: Every accepted $\mathsf{MeirKeelerContraction}^{\uparrow}$ read factors through one MetricUp carrier row, one distance readback row, one self-map row, one Meir–Keeler epsilon-to-delta ledger, one fixed-point iteration row, one Cauchy handoff ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any completion-facing fixed-point consumer can read it.

Independence witness: metric, contraction-principle, fixed-point-iteration, completemetric, banach-fixed-point, lipschitzmap, regular-cauchy-nonexpansive-map: none is bijective with $\mathsf{MeirKeelerContraction}^{\uparrow}$ because the packet jointly stores the metric carrier, self-map graph, epsilon-to-delta contraction ledger, finite iteration route, Cauchy handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{MeirKeelerContraction}^{\uparrow}$ packet is generated from a MetricUp carrier, distance readback, self-map graph, Meir–Keeler epsilon-to-delta contraction ledger, fixed-point iteration row, Cauchy handoff ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3623.1, Definition 3623.2, and Definition 3623.3 seed the finite Meir–Keeler contraction packet and its completion-facing Cauchy handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MeirKeelerContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global Lipschitz constant, selected endpoint, quotient completion equality, host fixed-point equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, epsilon-to-delta ledger exactness, fixed-point iteration visibility, Cauchy handoff exactness, transport, replay, provenance, and local naming for BEDC.Derived.MeirKeelerContractionUp.

Chapter 3624

A Concrete Naming Certificate for $\mathsf{IntervalHalvingModulus}^{\uparrow}$

Beginning from Bishop–Bridges constructive interval bisection, the $\mathsf{IntervalHalvingModulus}^{\uparrow}$ packet records the finite halving schedule used by L10 interval and Cauchy constructions before any completed point or ambient real-completeness principle is exposed. It consumes the located interval surface of chapter 1709, the dyadic tolerance ladder of chapter 3525, the finite observation windows of chapter 126, the regular located-rational modulus route of chapter 3532, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet gives bisection, nested-interval, and regular-Cauchy extraction chapters a shared finite modulus object rather than a compactness or choice principle.

Definition 3624.1 (Interval halving modulus carrier)**.** *An* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, L, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row. It records the inspected interval window, endpoint location data, inclusion rows, and nonempty finite evidence visible to a halving consumer.*
2. *L is the* $\mathsf{DyadicToleranceLadder}^{\uparrow}$ *row. It records the finite tolerance requests and the displayed halving relation between successive dyadic scales.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names the finite interval, endpoint, midpoint, and tolerance observations that may be opened by the route.*
4. *R is the* $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$ *handoff row. It records the located-rational regular Cauchy modulus extracted from scheduled interval halves and tolerance requests.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for midpoint formation, endpoint comparison, diameter comparison, and dyadic tolerance readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after I, L, S, R, D have displayed the finite halving route.*
7. *H records componentwise* hsame *transport for interval windows, tolerance ladders, stream schedules, located-modulus handoffs, dyadic ledgers, real seals, replay rows, provenance, and local naming.*

8. *C records displayed* Cont *replay from an interval-halving request through I, L, S, R, D, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{IntervalHalvingModulus}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicToleranceLadder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRatLocatedModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten displayed rows. It has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit, countable choice, trichotomy, quotient interval equality, host interval equality, or bridge checking.*

Lean4 ✓ `BEDC.Derived.IntervalHalvingModulusUp.TasteGate.IntervalHalvingModulusTasteGate single carrier alignment`

Theorem 3624.2 (Interval halving modulus NameCert obligations)**.** *For every accepted* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier*

$$M = (I, L, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, dyadic tolerance ladder L, finite stream schedule S, regular located-rational modulus handoff R, dyadic arithmetic ledger D, real-seal boundary E, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3624.1. The projection exposes one finite interval row, one dyadic tolerance ladder, one stream schedule, one located-rational modulus handoff, one dyadic arithmetic ledger, one real-seal boundary, and the structural rows.

The interval and tolerance obligations are I and L. The finite schedule S states which interval endpoints, midpoints, and tolerance requests are opened, while D supplies the arithmetic comparisons needed by those observations.

The handoff row R is downstream from the displayed interval, ladder, schedule, and dyadic rows. It exports only the regular located-rational modulus read from those finite observations.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no extra coordinate for completeness, selected limits, countable choice, trichotomy, quotient interval equality, or host interval equality, the displayed rows exhaust the NameCert obligations. □

Theorem 3624.3 (Interval halving modulus ladder handoff)**.** *Let $M = (I, L, S, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{IntervalHalvingModulus}^{\uparrow}$ *carrier. Every bisection, nested-interval, or regular-Cauchy consumer of M factors through the finite route*

$$I, L \longmapsto S, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the consumer receives a displayed halving modulus and located-rational Cauchy handoff before any $\mathsf{Real}^{\uparrow}$ *seal can be read.*

Proof. Start with the located interval row I and the dyadic tolerance ladder L. They provide the only interval window and halving-scale data available to the consumer.

The schedule row S selects the finite observations to open, and the dyadic ledger D checks midpoint formation, endpoint order, diameter bounds, and tolerance comparisons for those observations.

The row R repacks exactly this scheduled data as the regular located-rational modulus. The real-facing row E is therefore downstream from the finite halving route rather than an external source of completeness.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Hence bisection, nested-interval, and regular-Cauchy consumers cannot leave the finite ladder handoff without leaving the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{IntervalHalvingModulus}^{\uparrow}$ packet is generated from located interval rows, dyadic tolerance ladders, StreamName schedules, regular located-rational modulus handoffs, dyadic arithmetic ledgers, Real seal boundaries, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3624.1, Definition 3624.2, and Definition 3624.3 seed the finite halving schedule used by interval bisection, nested interval, and regular-Cauchy extraction routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalHalvingModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntervalHalvingModulusUp.TasteGate.IntervalHalvingModulusTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter refuses $\mathsf{Real}^{\uparrow}$ completeness, selected limits, countable choice, trichotomy, quotient interval equality, host interval equality, and bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval exposure, tolerance-ladder exactness, schedule compatibility, dyadic midpoint and diameter exactness, regular located-rational modulus handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.IntervalHalvingModulusUp.

Chapter 3625

A Concrete Naming Certificate for $\mathsf{PostMedianGateShell}^{\uparrow}$

The $\mathsf{PostMedianGateShell}^{\uparrow}$ packet realizes the post-median gate-shell object of Definition 9462.1490 as finite BEDC NameCert data. It reads the six-boundary-tile vision through codon-shell rows, Boolean-coordinate rows, spectral-optimum rows, first-exit rows, transport, replay, provenance, and local naming. The packet is a science-facing finite certificate: it does not treat the observed shell as a universal genetic-code law, an unbounded evolutionary optimizer, or an oracle for unseen codon tables.

Falsifiable prediction: Every accepted $\mathsf{PostMedianGateShell}^{\uparrow}$ read must display the six shell rows CAR, CCR, GUR, the median gate M, the first-exit target layer, and the retained CUR exit-closing row before any spectral-optimum claim is exported.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, punctured-cube-skeleton, wrr-state-phase-space, median-closure-structure: six-boundary-tile coverage names the codon-window partition, window-coverage trigger algebra names activation gates, the punctured-cube skeleton names cube geometry, WRR state phase space names replay states, and median closure structure names the median set. None is bijective to $\mathsf{PostMedianGateShell}^{\uparrow}$ because this packet binds post-median shell rows, Boolean coordinates, first-exit rows, spectral optimality, transport, replay, provenance, and local naming in one accepted surface.

The shell route consumes the finite boundary-tile coverage row of chapter 3470 and the trigger-algebra row of chapter 3478 before the median gate and first-exit rows are read. These sibling rows provide the boundary support and activation surface; the present packet adds the post-median shell ledger, spectral row, first-exit row, transport, replay, provenance, and local naming.

Definition 3625.1 (Post-median gate shell carrier)**.** *A* $\mathsf{PostMedianGateShell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, B, S, F, T, C, P, N)$$

with the following displayed rows:

1. *K is the codon-shell row. It records the median gate M, the shell rows* CAR*,* CCR*,* GUR*, the retained* CUR *exit-closing row, and the finite read request.*
2. *B is the Boolean-coordinate row. It records the six-boundary tile coordinates used to read the shell as a finite yes-or-no window rather than as a host-level biological assertion.*
3. *S is the spectral-optimum row. It records the killed-walk Perron-root comparison for the displayed shell rows and refuses to export an optimizer outside the finite comparison table.*

4. *F is the first-exit row. It records that* CAR, CCR, *and* GUR *are read as the top first-exit target layer after the median gate.*
5. *T records componentwise* hsame *transport for codon-shell rows, Boolean coordinates, spectral rows, first-exit rows, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from the post-median gate-shell request through K, B, S, F, T, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{PostMedianGateShell}^{\uparrow}$*, the six-boundary-tile coverage, the window-coverage trigger algebra, the punctured-cube skeleton, the WRR state phase space, the median closure structure,* BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eight rows. It has no coordinate for arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, host biochemical causality, or bridge checking.

Lean4 ✓ `BEDC.Derived.PostMedianGateShellUp.PostMedianGateShellTasteGate single carrier alignment`

Theorem 3625.2 (Post-median gate shell NameCert obligations)**.** *For every accepted* $\mathsf{PostMedianGateShell}^{\uparrow}$ *carrier*

$$G = (K, B, S, F, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$ *surface consists exactly of carrier admission for the codon-shell row K, Boolean-coordinate row B, spectral-optimum row S, first-exit row F, componentwise transport T, continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3625.1. The projection exposes one finite codon-shell row, one Boolean-coordinate row, one spectral row, one first-exit row, and the structural rows.

The codon and coordinate obligations are K and B. They force every read to display the six shell rows and the finite boundary-tile coordinates before the shell can be compared spectrally.

The spectral and first-exit obligations are S and F. The spectral row is tied to the displayed finite comparison table, while the first-exit row ties CAR, CCR, and GUR to the post-median target layer.

Finally T, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, or host biochemical causality, the displayed rows exhaust the NameCert obligations. □

Theorem 3625.3 (Post-median gate shell accepted-read frontier)**.** *Every accepted read of a* $\mathsf{PostMedianGateShell}^{\uparrow}$ *packet factors through*

$$K, B \longmapsto S, F \longmapsto T, C, P, N.$$

Thus a consumer may export the post-median shell only after the six shell rows, Boolean-coordinate window, finite spectral comparison, and first-exit target layer have all been displayed.

Proof. Begin with the shell row K and Boolean-coordinate row B. These rows display the median gate, CAR, CCR, GUR, CUR, and the finite six-boundary coordinates available to the read.

The spectral row S then records the finite Perron-root comparison for that displayed shell. The first-exit row F records the route from the median gate to the top target layer, so the consumer cannot replace the shell by an unrelated codon orbit.

The rows T, C, P, N preserve the same finite surface under transport, replay, provenance, and local naming. Any read that claims an unbounded optimizer, unseen-window extrapolation, global evolutionary selection, or host biochemical causality leaves the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}$.

Constructive story: A $\mathsf{PostMedianGateShell}^{\uparrow}$ packet is generated from finite codon-shell rows, Boolean-coordinate rows, spectral-optimum rows, first-exit rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3625.1, Definition 3625.2, and Definition 3625.3 seed the finite post-median shell certificate over displayed codon, Boolean-coordinate, spectral, and first-exit rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PostMedianGateShell}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary codon-table optimality, global evolutionary selection, unseen-window extrapolation, host biochemical causality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, six-row shell exposure, Boolean-coordinate exactness, spectral-optimum exactness, first-exit exactness, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.PostMedianGateShellUp.TasteGate.

Chapter 3626

A Concrete Naming Certificate for MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$

The MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ packet realizes the dependent-codomain barrier of subsection 9386.7.1 as finite NameCert data. It reads the inversion boundary of chapter 781, the closure-preservation packet of chapter 804, the subject-reduction boundary of chapter 721, and the closed-normal consistency boundary of chapter 733. The packet names the stability row required before a dependent-codomain application branch can be routed to subject reduction; it does not assert unconditional subject reduction.

Definition 3626.1 (MetaCIC dependent-codomain stability boundary carrier)**.** *A* MetaCICDependentCodomainSta *carrier is a finite* BHist *packet*

$$S = (A, D, R, I, V, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the application-argument reduction row whose codomain may observe the reducing argument.*
2. *D is the dependent-codomain readback row, displaying the pre-step and post-step codomain observations.*
3. *R is the subject-reduction discharge request row that asks for the application branch of the MetaCIC interface.*
4. *I is the dependent-codomain inversion-boundary row.*
5. *V is the closure-preservation transport row available before typing preservation is requested.*
6. *Q is the obstruction row recording that subject reduction is unavailable unless the dependent-codomain stability row is supplied.*
7. *L is the stability ledger: it records the exact row that a later bundle witness must fill to make the dependent codomain readback stable under the application-argument step.*
8. *H records componentwise* hsame *transport for application, codomain readback, discharge request, inversion, closure-preservation, obstruction, stability-ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the MetaCIC open-problem boundary through $A, D, R, I, V, Q, L, H, P, N$.*

10. *P is the* Pkg *provenance over* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$, DependentCodomainInversionBoun DependentCodomainClosurePreservation$^{\uparrow}$, MetaCICSubjectReductionBoundaryUp, ClosedNormalConsistencyBound SubjectReductionDischargeBundle, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, D, R, I, V, Q, L, H, C, P, N$. It has no coordinate for unconditional subject reduction, conversion completeness, closed consistency, ambient normalization, or a hidden discharge witness.

Theorem 3626.2 (MetaCIC dependent-codomain stability boundary obligations)**.** *For an accepted* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ *carrier $S = (A, D, R, I, V, Q, L, H, C, P, N)$, the local* NameCert$_{\text{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$ *surface consists exactly of application-argument reduction admission, dependent-codomain readback visibility, subject-reduction discharge-request visibility, inversion-boundary routing, closure-preservation transport, obstruction recording, stability-ledger exactness, componentwise transport, displayed replay, provenance, and local naming. A downstream Lean follow-up must supply the corresponding 'Nontrivial' and 'FieldFaithful' instances before this boundary can be consumed as a checked discharge row.*

Lean4 ✓ `BEDC.Derived.MetaCICDependentCodomainStabilityBoundaryUp.TasteGate.MetaCICDependentCodomainStab`
`single carrier alignment`

Proof. Project the carrier. The rows $A, D, R, I, V, Q, L, H, C, P, N$ are the only displayed coordinates, so the local NameCert obligations are their admission, compatibility, and replay checks.

The codomain-sensitive part is A, D, R. These rows record that the reduction branch is an application-argument branch and that the codomain readback can observe the reducing argument before the subject-reduction request is made.

The boundary part is I, V, Q, L. The inversion row and closure-preservation row provide the available finite data, the obstruction row records why they are insufficient by themselves, and the stability ledger names the missing row rather than discharging it.

Finally H, C, P, N transport, replay, source, and name the same finite packet. The required Lean 'Nontrivial' and 'FieldFaithful' instances belong to the later bundle witness because no hidden instance coordinate occurs in this carrier. □

Theorem 3626.3 (MetaCIC dependent-codomain stability handoff)**.** *Every subject-reduction-facing read of an accepted* MetaCICDependentCodomainStabilityBoundary$^{\uparrow}$ *carrier factors through the finite route*

DependentCodomainInversionBoundary$^{\uparrow}$, DependentCodomainClosurePreservation$^{\uparrow}$, MetaCICSubjectReductionBound

The handoff exposes the stability ledger that a later bundle witness must fill. It does not turn the dependent-codomain application case into unconditional subject reduction and does not promote closed-normal consistency to full closed consistency.

Proof. Begin with A, D, R. The read is subject-reduction-facing only because it has first displayed the application-argument step, the dependent-codomain readback, and the discharge request.

Next consume I, V, Q. The inversion boundary and closure-preservation packet give the available codomain and closedness data, while the obstruction row records that this data does not yet type the post-step codomain.

The ledger L is then the only exported positive handoff. It names the dependent-codomain stability row that a later bundle witness must supply, including the instance obligations recorded in the carrier theorem.

Finally H, C, P, N replay the route without adding theorem strength. Thus a consumer receives a precise boundary packet and a setup-facing row, not an unconditional subject-reduction theorem. □

Theorem 3626.4 (AppArgTypeStable boundary row)**.** *For an accepted* MetaCICDependentCodomainStabilityBoun *carrier* $S = (A, D, R, I, V, Q, L, H, C, P, N)$*, the ledger row* L *is the AppArgTypeStable-facing boundary: any application-argument reduction read must display the pre-step and post-step codomain observations in* D *before it can feed the subject-reduction discharge request* R*.*

Proof. Project the rows A, D, R, L from the accepted carrier. The row A identifies the read as an application-argument reduction, while D displays the dependent codomain before and after that step.

Apply Definition 3626.3. The only positive row exported toward the subject-reduction request is L, and the handoff reaches it only after A, D, R have been displayed.

Thus L names the AppArgTypeStable-facing boundary for the later discharge witness. It records the required codomain-stability row without claiming unconditional subject reduction. □

Falsifiable prediction: Any $\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}$ packet lacking the dependent-codomain stability row L cannot discharge the application-argument branch of subject reduction through the displayed MetaCIC discharge interface.

Independence witness: dependentcodomaininversionboundary, dependentcodomainclosurepreservation, metacicsubjectreductionboundary, closed-normal-consistency-boundary, metacic-open-problems: none is bijective with $\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}$ because this packet jointly stores application-argument reduction, dependent codomain readback, discharge request, inversion boundary, closure-preservation transport, obstruction row, stability ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}$ packet is generated from application-argument reduction, dependent-codomain readback, subject-reduction discharge request, inversion-boundary, closure-preservation, obstruction, stability-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3626.1, Definition 3626.2, and Definition 3626.3 seed the finite dependent-codomain stability boundary packet for MetaCIC subject-reduction consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDependentCodomainStabilityBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unconditional subject reduction, conversion completeness, full closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked application-argument admission, dependent-codomain readback visibility, inversion-boundary routing, closure-preservation transport, obstruction recording, stability-ledger exactness, and the required witness instances for BEDC.Derived.MetaCICDependentCodomainStabilityBoundaryUp.

Chapter 3627

A Concrete Naming Certificate for $\mathsf{DyadicSandwich}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{DyadicSandwich}^{\uparrow}$ packet records the finite squeeze surface by which lower and upper dyadic rows trap a regular rational tail before a Real-facing consumer can read the seal. It sits over the dyadic tolerance core of chapter 291, the finite schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tail ball boundary of chapter 1633, and the terminal Real packet of chapter 38. The packet names the squeeze route as finite certificate data; it is not a quotient-real equality theorem, a selected limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 3627.1 (DyadicSandwich carrier)**.** *A* $\mathsf{DyadicSandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic row, listing the inspected lower endpoints and their finite-window order witnesses.*
2. *U is the upper dyadic row, listing the inspected upper endpoints and the matching containment witnesses.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing only the windows on which L and U are compared.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the dyadic radii, order comparisons, and residual budgets for the sandwich.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-tail row whose displayed values remain between the lower and upper dyadic rows at the scheduled windows.*
6. *H records componentwise* hsame *transport for lower rows, upper rows, schedules, dyadic ledgers, regular tails, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from the dyadic lower and upper rows through the schedule, tolerance ledger, regular tail, and Real-facing handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicSandwich}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicTailBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $L, U, S, D, R, H, C, P, N$. It has no coordinate for quotient equality of regular names, selected limits, trichotomy, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 3627.2 (DyadicSandwich NameCert obligations)**.** *For every accepted* $\mathsf{DyadicSandwich}^{\uparrow}$ *carrier $Q = (L, U, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the lower dyadic row, upper dyadic row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat regular-tail row, transport, replay, provenance, and local naming;*
2. *lower/upper order, requiring each displayed lower row to lie below the matching upper row through the dyadic comparison data in D;*
3. *tail containment, requiring the regular-tail values in R to remain inside the displayed dyadic interval at every window named by S;*
4. *classifier stability, requiring any accepted comparison of packets to transport $L, U, S, D, R, H, C, P, N$ componentwise and to preserve the displayed containment route;*
5. *squeeze handoff, requiring every* $\mathsf{DyadicTailBall}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read the schedule, dyadic ledger, and regular-tail row before the terminal seal is available;*
6. *provenance exhaustion, requiring all accepted reads to be sourced by P and locally named by N, with no hidden quotient, selected-limit, trichotomy, countable-choice, or ambient-completeness coordinate.*

Lean4 ✓ `BEDC.Derived.DyadicSandwichUp.DyadicSandwichTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3627.1. The visible coordinates are exactly $L, U, S, D, R, H, C, P, N$, so carrier admission and provenance exhaustion are read from those rows.

The order and containment obligations are the two analytic faces of the same displayed dyadic ledger. The lower and upper rows are compared through D, while the regular-tail row R is inspected only at windows named by S.

Classifier stability follows componentwise through H and by replay through C. Since the replay route names the same lower row, upper row, schedule, dyadic ledger, and regular tail after transport, a classifier comparison cannot replace the sandwich with a quotient class or a selected limiting object.

Finally the consumer handoff is downstream of S, D, R. A $\mathsf{DyadicTailBall}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read therefore reaches the terminal boundary only after the finite squeeze rows have been displayed, and the carrier list supplies no trichotomy, countable-choice, or ambient-completeness datum. □

Theorem 3627.3 (DyadicSandwich regular Cauchy squeeze route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{DyadicSandwich}^{\uparrow}$ *packet factors through*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicTailBall}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The route first reads the dyadic lower and upper endpoints with their tolerance ledger, then the finite schedule, then the regular-tail row, then the dyadic tail-ball boundary, and only then the Real seal. It exports no quotient equality, selected limit, trichotomy, countable-choice principle, or ambient completeness theorem.

Proof. Start from an accepted packet $Q = (L, U, S, D, R, H, C, P, N)$. The dyadic information is exhausted by L, U, D, so every lower/upper comparison and tolerance request is first reduced to the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger.

The schedule row S fixes the finite windows on which the squeeze is inspected. The regular-tail row R is attached to those windows, so a consumer cannot read a tail value outside the displayed schedule.

The dyadic tail-ball boundary is then obtained by replaying the same lower, upper, schedule, dyadic-ledger, and regular-tail rows through C and transporting them through H. The provenance row P and local naming row N package that route for the terminal $\mathsf{Real}^{\uparrow}$ read.

No remaining carrier coordinate can supply quotient equality, a selected limit, trichotomy, countable choice, or ambient completeness. Hence every Real-facing read factors through the displayed constructive squeeze route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicSandwich}^{\uparrow}$ packet is generated from a lower dyadic row, an upper dyadic row, a finite StreamName schedule, a DyadicRatCore tolerance ledger, a RegSeqRat regular-tail row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$, seed)

Scope closed: Definition 3627.1, Definition 3627.2, and Definition 3627.3 seed the constructive finite squeeze route from dyadic tail windows to Real-facing reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicSandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower/upper order, tail containment, classifier stability, squeeze handoff, and provenance exhaustion for BEDC.Derived.DyadicSandwichUp.

Chapter 3628

A Concrete Naming Certificate for $\mathsf{CauchyCompletionRecursor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and uniformly continuous extension, the $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ packet records the elimination principle by which a completion-facing consumer reads a completed object only through a displayed dense source, a modulus-controlled map, an extension handoff, and a uniqueness row. It sits after the regular completion carrier of chapter 1639, the source admission row of chapter 1527, the uniformly continuous extension packet of chapter 769, and the terminal Real seal of chapter 38. The recursor is an elimination route for displayed completion data; it is not a selected limit, quotient stream, countable-choice principle, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3628.1 (CauchyCompletionRecursor carrier)**.** *A* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, T, f, m, E, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the dense source row, read through the Bishop regular Cauchy completion carrier and the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-admission route.*
2. *T is the completion target row, exposing only the completion-facing boundary consumed by the recursor.*
3. *f is the source map row, visible only on the displayed dense source S.*
4. *m is the uniform modulus row controlling the source map before any completion-facing read is admitted.*
5. *E is the extension handoff row supplied by the uniformly continuous extension route.*
6. *U is the uniqueness row, requiring any two admitted extension reads with the same source restriction and modulus ledger to agree through the displayed completion and Real seal rows.*
7. *H records componentwise* hsame *transport for the dense source, target, source map, modulus, extension handoff, uniqueness row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing request through S, f, m, E, U and then to the terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{UniformContinuousExtension}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for selected limits, quotient stream representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionRecursorUp.CauchyCompletionRecursorTasteGate single carrier alignment`

Theorem 3628.2 (CauchyCompletionRecursor NameCert obligations)**.** *For every accepted* CauchyCompletionRecu *carrier $R = (S, T, f, m, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dense source, completion target, source map, modulus ledger, extension handoff, uniqueness row, transport, replay, provenance, and local naming rows;*
2. *source restriction, requiring f to be read only on the displayed dense source S admitted by the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *route;*
3. *modulus preservation, requiring every extension read to preserve the same uniform modulus row m before the target row T is consumed;*
4. *extension handoff, requiring E to be supplied through the uniformly continuous extension route and not by a selected limit or host completion theorem;*
5. *uniqueness, requiring U to identify any two admitted extension reads with the same source restriction, modulus ledger, and completion target route;*
6. *ledger non-escape, requiring H, C, P, N to replay and name only the displayed rows and to export no selected limit, quotient stream, countable-choice selector, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3628.1. The visible packet consists exactly of the dense source row S, target row T, source map row f, modulus row m, extension handoff E, uniqueness row U, and structural rows H, C, P, N.

Source restriction follows from the placement of S before f. The source map can be inspected only after the dense source has been displayed through the Bishop regular completion and $\mathsf{CauchyCompletionUnit}^{\uparrow}$ routes.

Modulus preservation and extension handoff are read through m and E. The uniformly continuous extension route consumes the same modulus ledger before it exposes an extension row, so no selected limit or ambient completion theorem can replace the displayed handoff.

Uniqueness is exactly the role of U. Once the source restriction, modulus ledger, target row, and extension handoff agree, the local uniqueness row identifies the two recursor reads through the displayed completion-facing route.

Finally, H, C, P, N exhaust the ledger. They transport, replay, source, and name the displayed rows only, so there is no carrier coordinate from which quotient streams, selected limits, countable choice, host real equality, or ambient Real completeness can be exported. □

Theorem 3628.3 (CauchyCompletionRecursor universal route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{CauchyCompletionUnit}^{\uparrow} \longmapsto \mathsf{UniformContinuousExtension}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer first reads the displayed dense source S, then the source map f and modulus ledger m, then the extension handoff E, then the uniqueness row U, and only then the terminal Real seal. The route contains no selected limit, countable choice, quotient stream, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an accepted packet $R = (S, T, f, m, E, U, H, C, P, N)$. The dense source row S is the only source coordinate available to the consumer, and it is admitted through the Bishop regular completion and $\mathsf{CauchyCompletionUnit}^{\uparrow}$ routes.

Replay the continuation row C. A completion-facing request reaches f only after S is displayed, and it reaches E only after the modulus ledger m has been read. Thus the extension handoff is modulus-controlled rather than limit-selected.

The uniqueness row U is read before the terminal Real seal. It forces any two admitted extension reads with the same source restriction, modulus ledger, and target route to share the same completion-facing output row.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and naming. Since the carrier has no coordinate for selected limits, countable choice, quotient streams, host real equality, or ambient Real completeness, no completion-facing consumer can export any of those data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRecursor}^{\uparrow}$ packet is generated from a dense source row, completion target row, source map, uniform modulus ledger, extension handoff, uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3628.1, Definition 3628.2, and Definition 3628.3 seed the finite elimination route from displayed dense source data through modulus-controlled extension and uniqueness into the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRecursor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient streams, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source restriction, modulus preservation, extension handoff, uniqueness, replay stability, provenance exhaustion, and Real seal nonescape for BEDC.Derived.CauchyCompletionRecursorUp.

Chapter 3629

A Concrete Naming Certificate for $\mathsf{DedekindReal}^{\uparrow}$

Following Dedekind and Bishop–Bridges constructive analysis, the $\mathsf{DedekindReal}^{\uparrow}$ packet records a real name by a displayed located cut together with the finite Cauchy readback surface that lets completion-facing consumers compare cut data with regular rational Cauchy names. It sits beside the located-cut carrier of chapter 404, the tight cut boundary of **??**, the regular rational sequence route of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the cut-versus-Cauchy comparison surface; it is not a quotient of cuts, a selected supremum, trichotomy on rationals, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 3629.1 (DedekindReal carrier)**.** *A* $\mathsf{DedekindReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (L, C, T, R, S, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *row exposing the lower and upper rational probes and the locatedness witness available to the cut.*
2. *C is the* $\mathsf{CauchyCut}^{\uparrow}$ *row tying finite Cauchy windows to the same cut boundary.*
3. *T is the* $\mathsf{TightLocatedCut}^{\uparrow}$ *row recording the no-gap compatibility between lower and upper probes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row used as the Cauchy name readback.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the inspected rational approximants.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare cut probes and Cauchy windows at one displayed budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, C, T, R, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for cut, Cauchy-cut, tightness, sequence, stream, tolerance, Real-seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a cut-facing or Cauchy-facing request through L, C, T, R, S, Q and then to E.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindReal}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{TightLocatedCut}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient cut equality, a selected supremum, a detached Cauchy limit, rational trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DedekindRealUp.DedekindRealTasteGate single carrier alignment`

Theorem 3629.2 (DedekindReal NameCert obligations)**.** *For every accepted* $\mathsf{DedekindReal}^{\uparrow}$ *carrier* $D = (L, C, T, R, S, Q, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, located-cut exposure through L, Cauchy-cut compatibility through C, tight no-gap checking through T, regular rational readback through R, finite stream-window exposure through S, dyadic tolerance comparison through Q, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Every Real and Completion consumer that treats a Dedekind-style cut as a real name must read the finite route

$$\mathsf{LocatedCut}^{\uparrow} \longrightarrow \mathsf{CauchyCut}^{\uparrow} \longrightarrow \mathsf{TightLocatedCut}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Re}$$

before exporting any real-facing row.

Proof. Project the carrier of Definition 3629.1. The accepted packet exposes only the located cut, Cauchy-cut, tightness, regular rational sequence, stream window, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

The cut side is forced by L, C, T. Locatedness supplies rational probes on both sides of the boundary, the Cauchy-cut row ties those probes to finite Cauchy windows, and the tight row records that no displayed gap remains between the lower and upper faces.

The Cauchy side is forced by R, S, Q, E. The regular rational sequence supplies the finite approximants, the stream row selects the inspected windows, the dyadic ledger fixes the comparison budget, and the Real seal can be reached only after those rows have been consumed.

Finally H, K, P, N exhaust the structural surface. They transport, replay, source, and name the displayed rows only, so quotient cut equality, selected suprema, detached limits, rational trichotomy, countable choice, and ambient Real completeness have no export coordinate in the local certificate. □

Theorem 3629.3 (DedekindReal cut–Cauchy comparison)**.** *Let $D = (L, C, T, R, S, Q, E, H, K, P, N)$ be an accepted* $\mathsf{DedekindReal}^{\uparrow}$ *carrier. A cut-facing read and a Cauchy-facing read compare only when both are routed through the same tolerance ledger Q and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal E. In particular, every admitted comparison between the Dedekind cut surface and the regular Cauchy real-name surface factors through*

$$(L, C, T) \longmapsto Q \longleftarrow (R, S)$$

and then through E, H, K, P, N.

The comparison contains no quotient of cuts, no selected supremum, no selected Cauchy limit, no trichotomy branch, no countable-choice selector, and no appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted packet. A cut-facing read can inspect only L, C, T, while a Cauchy-facing read can inspect only R, S before the common dyadic tolerance row is reached.

The row Q is the only displayed comparison budget. Therefore a lower-or-upper rational probe from L and a regular rational approximant from R, S can be compared only after both sides have entered the same dyadic ledger.

The terminal seal E is downstream of that shared ledger. Transport and replay through H and K preserve the same order, and P, N package only the resulting finite route.

Since no carrier coordinate stores quotient cut equality, a supremum selector, a detached limit, trichotomy, countable choice, or ambient completeness, the comparison is exactly the finite cut-versus-Cauchy route displayed above. □

Falsifiable prediction: Every accepted $\mathsf{DedekindReal}^{\uparrow}$ consumer that claims a cut-versus-Cauchy real-name comparison must exhibit locatedcut, cauchycut, tightlocatedcut, regseqrat, streamname, dyadicratcore, real, transport, replay, provenance, and local namecert rows. A consumer omitting any one of those rows refutes the packet.

Independence witness: locatedcut, cauchycut, tightlocatedcut, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole packet $D = (L, C, T, R, S, Q, E, H, K, P, N)$, because each sibling contributes only cut probes, Cauchy compatibility, tightness, rational sequence data, finite windows, tolerance budgets, or terminal real sealing while this chapter binds them as one finite comparison certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindReal}^{\uparrow}$ packet is generated from a LocatedCut row, CauchyCut compatibility row, TightLocatedCut no-gap row, RegSeqRat readback, StreamName finite windows, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3629.1, Definition 3629.2, and Definition 3629.3 seed the finite cut-versus-Cauchy real-name route from located cut probes through Cauchy compatibility, tightness, regular rational readback, finite stream windows, dyadic tolerance, and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient cut equality, selected suprema, selected Cauchy limits, rational trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cut exposure, Cauchy-cut compatibility, tight no-gap checking, regular rational readback, finite-window exposure, dyadic tolerance comparison, terminal Real seal routing, and ledger nonescape for BEDC.Derived.DedekindRealUp.

Chapter 3630

A Concrete Naming Certificate for $\mathsf{ObservableDiameterDecay}^{\uparrow}$

The $\mathsf{ObservableDiameterDecay}^{\uparrow}$ packet realizes the compact finite-net and uniform-modulus route of chapter 9237 as a finite diameter ledger. It consumes the compact source of chapter 129, the pointwise graph and radius data of chapter 130, the finite radius ledger of chapter 452, the uniform-modulus consumer of chapter 369, and the source-to-target triangle estimate of chapter 1159. The packet exposes observable diameter decay as NameCert data rather than as a narrative compactness bridge.

Definition 3630.1 (Observable diameter decay carrier)**.** *An* $\mathsf{ObservableDiameterDecay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, L, T, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the inspected centers and source-radius windows.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise graph and local-modulus row for the displayed map on those centers.*
3. *L is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius row binding the finite compact cover to one positive radius ledger.*
4. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *triangle row that turns source-radius bounds and pointwise target estimates into a target-diameter bound.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, read only after K, F, L, T have been displayed.*
6. *H records componentwise* hsame *transport for compact-net, continuous-map, radius-ledger, triangle-estimate, uniform-modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from finite-net source windows through pointwise estimates, the finite Lebesgue-number row, the triangle estimate, and the uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{ObservableDiameterDecay}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, L, T, M, H, C, P, N$. It has no coordinate for open-cover compactness, host semantic continuity, selected centers outside the displayed finite net, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayTasteGate` single carrier alignment

Theorem 3630.2 (Observable diameter decay NameCert obligations)**.** *For every accepted* ObservableDiameterDeca *carrier $O = (K, F, L, T, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$ *surface consists exactly of carrier admission; compact finite-net coverage through K; pointwise-modulus compatibility through F; finite Lebesgue-number radius exactness through L; triangle-estimate diameter control through T; uniform-modulus handoff through M; componentwise transport and displayed replay through H and C; and provenance exhaustion through P and N.*

Proof. Project the carrier of Definition 3630.1. The visible coordinates are exactly the compact finite-net row, continuous-map row, finite radius ledger, triangle estimate, uniform-modulus handoff, and structural rows.

Coverage is read from K, and pointwise-modulus compatibility is read from F. The finite Lebesgue-number row L binds those local source-radius windows to a single positive radius ledger.

The triangle row T converts the shared source-radius bound and pointwise target estimates into the displayed target-diameter control. The uniform-modulus row M is downstream from K, F, L, T, so the handoff cannot be inspected before the finite diameter-decay route is exposed.

Transport and replay through H and C preserve exactly those rows, while P and N source and name them. Since the carrier contains no open-cover compactness, host semantic continuity, external center selector, or ambient completeness coordinate, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayNamecertObligations`

Theorem 3630.3 (Observable diameter decay compact-uniform route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read from an accepted* $\mathsf{ObservableDiameterDecay}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{FiniteLebesgueNumber}^{\uparrow} \longrightarrow \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longrightarrow \mathsf{Uniform}$$

The route measures diameter decay on displayed compact-net cells, transports pointwise target estimates along the finite Lebesgue-number radius row, and seals the result as a uniform-modulus handoff. It exports no open-cover compactness, host semantic continuity, selected center outside the packet, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ObservableDiameterDecayUp.ObservableDiameterDecayCompactUniformRoute`

Proof. Begin with an accepted packet $O = (K, F, L, T, M, H, C, P, N)$. The compact source data are exactly the finite-net centers and source-radius windows recorded in K.

Read the continuous-map row F at those displayed centers. These pointwise target estimates are tied to the same finite source windows that the compact-net row exposes.

The radius row L supplies the finite Lebesgue-number ledger needed to make the source-radius request uniform over the inspected compact-net cells. The triangle row T then turns the source bounds and pointwise target estimates into a target-diameter bound.

Only after that route is replayed through C and transported through H can the uniform-modulus handoff M be read. The provenance and naming rows P, N package the same route, and no carrier coordinate supplies open-cover compactness, host semantic continuity, external center selection, or ambient completeness. □

Falsifiable prediction: Removing the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ radius row from an accepted $\mathsf{ObservableDiameterDec}$ carrier breaks the $\mathsf{UniformModulus}^{\uparrow}$ handoff, because the compact finite-net source row and pointwise $\mathsf{ContinuousMap}^{\uparrow}$ estimates no longer share one displayed positive-radius ledger.

Independence witness: compactuniformcontinuity, finitelebesguenumber, metrictriangleuniformestimate, uniformmodulus, compactmetric: none is bijective to $\mathsf{ObservableDiameterDecay}^{\uparrow}$ because this packet jointly carries compact-net coverage, pointwise continuous-map estimates, the finite Lebesgue-number radius ledger, triangle-estimate diameter control, uniform-modulus handoff, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservableDiameterDecay}^{\uparrow}$ packet is generated from a CompactMetric finite-net row, a ContinuousMap pointwise-modulus row, a FiniteLebesgueNumber radius ledger, a MetricTriangleUniformEstimate diameter-control row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3630.1, Definition 3630.2, and Definition 3630.3 seed the finite compact-net diameter-decay route consumed by uniform-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservableDiameterDecay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host semantic continuity, selected centers outside the displayed finite net, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net coverage, pointwise-modulus compatibility, finite radius exactness, triangle-estimate diameter control, uniform-modulus handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.ObservableDiameterDecayUp.

Chapter 3631

A Concrete Naming Certificate for $\mathsf{FastCauchyNormalForm}^{\uparrow}$

Bishop–Bridges constructive Cauchy completion treats a real number as regular approximation data rather than as a quotient object selected by an ambient limit operation. The $\mathsf{FastCauchyNormalForm}^{\uparrow}$ packet records the finite normal-form surface by which dyadic rational cores, finite stream names, regular rational sequences, terminal Real rows, and Cauchy-rate comparisons are read as one L10 completion-facing NameCert object. It is not quotient real equality, selected representative choice, countable choice, trichotomy, or an ambient completeness oracle.

The sibling dependency is part of the carrier: D imports the dyadic tolerance carrier of chapter 291, S imports the finite-window stream surface of chapter 126, R imports the regular rational route of chapter 301, E imports the terminal seal of chapter 38, and C imports the rate comparison surface of chapter 1671. The normal-form packet binds those sibling rows before any completion consumer can treat a regular name as fast normalized.

Definition 3631.1 (Real fast Cauchy normal-form carrier)**.** *A* $\mathsf{FastCauchyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, S, R, E, C, H, Q, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row, carrying the finite endpoint, radius, distance, and budget data used by the normal form.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected stream observations and the omitted-tail ledger.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row whose modulus is being normalized.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *C is the* $\mathsf{CauchyRateComparison}^{\uparrow}$ *row comparing the displayed regular sequence against the fast dyadic rate demanded by the packet.*
6. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular-sequence, Real-seal, Cauchy-rate, replay, provenance, and local naming rows.*
7. *Q records displayed* Cont *replay from a fast-normal-form request through D, S, R, E, C.*
8. *P is the* Pkg *provenance over* $\mathsf{FastCauchyNormalForm}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRateComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, R, E, C, H, Q, P, N$ and their displayed replay route. It has no coordinate for quotient real equality, selected representatives, detached limit choice, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FastCauchyNormalFormUp.FastCauchyNormalFormTasteGate single carrier alignment`

Theorem 3631.2 (Real fast Cauchy normal-form NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyNormalForm}^{\uparrow}$ *carrier $F = (D, S, R, E, C, H, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet, classifier stability under componentwise* hsame*, dyadic tolerance nonescape through D, finite-window exposure through S, regular-sequence admission through R, terminal seal routing through E, fast-rate comparison through C, displayed replay through Q, provenance through P, and local naming through N.*

Every completion-facing consumer that treats a regular rational sequence as fast-normalized must read the finite route

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{CauchyRateComparison}^{\uparrow}$$

before exporting any normal-form row.

Proof. Project the carrier of Definition 3631.1. The accepted packet exposes only the dyadic tolerance row, finite stream window, regular rational sequence, terminal Real seal, Cauchy-rate comparison, transport, replay, provenance, and local naming rows.

The analytic route is forced by D, S, R, E, C. The dyadic ledger supplies the displayed tolerance budgets, the stream row records the finite observations, the regular sequence row supplies the Cauchy data, the Real row seals only that displayed route, and the comparison row checks the fast rate against those same finite rows.

The structural rows H, Q, P, N preserve this route under componentwise transport and replay. Since no accepted coordinate stores quotient real equality, a selected representative, detached limit choice, countable choice, trichotomy, or ambient completeness, none of those objects belongs to the NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{FastCauchyNormalForm}^{\uparrow}$ consumer that claims fast Cauchy normal form must exhibit the dyadic tolerance row, finite stream window, regular rational sequence row, terminal Real seal row, Cauchy-rate comparison row, componentwise transport, displayed replay, provenance, and local naming row. A consumer omitting dyadicratcore, streamname, regseqrat, real, cauchyratecomparison, or namecert sibling slugs refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchyratecomparison: no listed sibling supplies the whole packet $F = (D, S, R, E, C, H, Q, P, N)$, because each sibling contributes only tolerances, finite observations, regular sequence data, terminal sealing, or rate comparison while this chapter binds all five analytic faces as one finite normal-form certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FastCauchyNormalForm}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular-sequence data, a terminal RealUp seal, a CauchyRateComparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 3631.1 and Definition 3631.2 seed the finite fast Cauchy normal-form route over dyadic tolerance data, finite stream names, regular rational sequences, terminal Real sealing, Cauchy-rate comparison, transport, replay, provenance, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastCauchyNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, detached limit choice, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance nonescape, finite-window exposure, regular-sequence admission, terminal-seal routing, fast-rate comparison, and FieldFaithful FastCauchyNormalFormUp for BEDC.Derived.FastCauchyNormalFormUp.

The $\mathsf{Calculus}^{\uparrow}$ dossier consumes the finite $\mathsf{Limit}^{\uparrow}$ route of chapter 1678 and the regular-Cauchy diagonal meet of chapter 909 before derivative and integral rows are read.

Chapter 3632

A Concrete Naming Certificate for $\mathsf{Calculus}^{\uparrow}$

Following Cartan's local differential calculus and the Bishop–Bridges constructive treatment of limits, derivatives, and integrals, the $\mathsf{Calculus}^{\uparrow}$ packet records the finite analysis root used by later derivative and integration chapters. Its derivative/integral local-linearization carrier consumes the $\mathsf{ContinuousMap}^{\uparrow}$ modulus surface of chapter 130, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ error ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It sits beside the real-name boundary of chapter 38, the generic limit route of chapter 1678, the continuous-map surface used throughout the analysis region, the derivative carrier of chapter 122, the integral carrier, and the regular rational readback of chapter 301. The packet names the common calculus substrate without importing a host function space, a global limit operator, quotient equality of functions, or an ambient completeness theorem.

Definition 3632.1 (Calculus local-linearization carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier is a finite* BHist *packet*

$$\Lambda = (E, R, D, L, J, H, C, P, N)$$

whose rows are the terminal $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the* $\mathsf{Derivative}^{\uparrow}$ *finite difference-quotient row* L*, the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite tagged-sum row* J*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. The classifier compares accepted packets only through these rows. It has no coordinate for a host differentiability theorem, selected antiderivative, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusLocalLinearizationCarrier`
Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocalLinearizationCarrier admission`

Theorem 3632.2 (Calculus derivative continuation obligation)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes a derivative-facing continuation only after the* $\mathsf{Real}^{\uparrow}$ *seal* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*,* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient row* L*,* hsame *transport* H*,* Cont *replay* C*, provenance* P*, and local NameCert row* N *have been displayed. The read cannot export a host differentiability theorem, quotient real equality, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeContinuationObligation`

Proof. Project the carrier of Definition 3632.1. The derivative-facing row is L, and its tolerance and value data are visible only together with the real seal E, regular-rational readback R, and dyadic ledger D.

The structural rows H, C, P, N transport, replay, source, and name that same finite packet. Since none of those rows contains a host differentiability theorem or quotient real equality, the continuation remains a displayed finite difference-quotient route.

The Riemann row J may coexist in the packet, but it does not supply derivative data. Thus every derivative consumer must read through E, R, D, L, H, C, P, N before any derivative-facing conclusion is public. □

Theorem 3632.3 (Calculus integral finite-sum obligation)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes an integral-facing read only through the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite tagged-sum row* J *together with the* $\mathsf{Real}^{\uparrow}$ *seal* E*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, componentwise transport, replay, provenance, and local naming. The read does not select an antiderivative and does not import quotient real equality or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusIntegralFiniteSumObligation`

Proof. Read J from the carrier. It records the partition, tags, finite sum, and endpoint handoff as displayed calculus data, so an integral consumer cannot bypass the finite-sum route.

The rows E, R, D provide the real-facing seal, regular-rational comparison, and dyadic tolerance ledger used to interpret that sum. The structural rows H, C, P, N preserve the same finite route under accepted transport and replay.

The derivative row L remains separate. Because no coordinate of Λ contains a selected antiderivative, host integration functional, quotient real equality, or ambient completeness theorem, the integral obligation is exactly the finite-sum route through J and the displayed support rows. □

Theorem 3632.4 (Calculus limit real-seal obligation)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing limit read exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier factors through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, derivative row* L *or Riemann finite-sum row* J*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*,* hsame *transport,* Cont *replay, provenance, and local NameCert rows. No such read may use a host limit theorem, quotient real equality, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local data.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealSealObligation`

Proof. Start from a Real-facing calculus read. The carrier exposes E as the only terminal real seal, so every such read must arrive at E through the displayed regular-rational and dyadic rows R, D.

There are two calculus sources for that seal. A derivative read uses L and the finite difference-quotient route; an integral read uses J and the finite tagged-sum route. Both sources are transported and replayed only by H and C, with provenance and local naming recorded by P and N.

Since the carrier has no coordinate for host limits, quotient real equality, selected antiderivatives, or ambient completeness, the real-seal obligation is confined to the displayed BEDC rows. □

Definition 3632.5 (Calculus root public package)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root public package of an accepted carrier is the finite packet*

$$\Pi_{\mathsf{Calculus}^{\uparrow}} = (E, R, D, L, J, H, C, P, N; C_{\mathrm{root}}, D_{\mathrm{root}}, I_{\mathrm{root}}, L_{\mathrm{root}}, R_{\mathrm{root}}, Q_{\mathrm{root}}, Y_{\mathrm{root}}, H_{\mathrm{root}}, T_{\mathrm{root}}, P_{\mathrm{root}}, N_{\mathrm{root}})$$

obtained by reading the local-linearization rows $E, R, D, L, J, H, C, P, N$ *together with the root rows* $C_{\mathrm{root}}, D_{\mathrm{root}}, I_{\mathrm{root}}, L_{\mathrm{root}}, R_{\mathrm{root}}, Q_{\mathrm{root}}, Y_{\mathrm{root}}, H_{\mathrm{root}}, T_{\mathrm{root}}, P_{\mathrm{root}}, N_{\mathrm{root}}$ *of Definition 3632.43. Its public surface is the derivative row, Riemann finite-sum row,* $\mathsf{Limit}^{\uparrow}$ *row,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming.*

The package compares accepted calculus reads only through those rows. It has no coordinate for a host function space, selected antiderivative, selected derivative operator, quotient equality of functions, host limit operator, host integral functional, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootPublicPackage`

Definition 3632.6 (Calculus root obligation surface)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root obligation surface is the ordered read of* $\Pi_{\mathsf{Calculus}^{\uparrow}}$ *through the derivative row, integral finite-sum row,* $\mathsf{Limit}^{\uparrow}$ *row,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal row, and the transport, replay, provenance, and local naming rows. It is the same finite packet as Definition 3632.5, read as downstream obligations rather than as a public package.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootObligationSurface`

Theorem 3632.7 (Calculus root limit derivative integral route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *root read for limits, derivatives, integrals, continuous-map replay,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{Real}^{\uparrow}$ *sealing passes through the obligation surface of Definition 3632.6. The consumer may choose the relevant operation row, but the shared route remains* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Proof. Read the obligation surface as the public package with its rows ordered by consumer. Derivative, integral, and limit consumers select D_{root}, I_{root}, or L_{root}, respectively, while continuous-map replay, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming remain common rows.

Apply the existing root package readiness theorem. It already forces each listed consumer through $\Pi_{\mathsf{Calculus}^{\uparrow}}$, so reading the same packet as an obligation surface gives the stated L10 route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitDerivativeIntegralRoute`

Theorem 3632.8 (Calculus root L10 nonescape)**.** *The obligation surface of Definition 3632.6 exports no host function space, quotient function equality, selected antiderivative, host limit operator, host integral functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every L10 calculus consumer stays inside the displayed limit, derivative, integral, continuous-map, dyadic-rational, regular-rational, Real-seal, transport, replay, provenance, and local naming rows.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootL10Nonescape`

Proof. Compare the refused objects with the row inventory of Definition 3632.6. None is a row of the displayed obligation surface.

Use Definition 3632.7 for the positive route and Definition 3632.10 for the boundary. Together they exhaust the public calculus read through $\Pi_{\mathsf{Calculus}^{\uparrow}}$ and leave no coordinate for the refused host objects. □

Theorem 3632.9 (Calculus root public package readiness)**.** *Every downstream* $\mathsf{Calculus}^{\uparrow}$ *consumer of derivative, integral, limit, continuous-map, or Real-facing data factors through the root public*

package of Definition 3632.5. A derivative consumer reads the derivative rows and the shared graph, dyadic, regular-rational, Real, transport, replay, provenance, and naming rows; an integral consumer reads the Riemann finite-sum rows and the same support; a limit consumer reads the $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows and the same support. No downstream calculus consumer obtains a public conclusion from a row outside* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootPublicPackageReadiness`

Proof. Start with the local-linearization obligations. The derivative, integral, and limit real-seal statements above force every local read through $E, R, D, L, J, H, C, P, N$ before a public conclusion is visible.

Now project the root carrier of Definition 3632.43. Its public operation rows are $C_{\mathrm{root}}, D_{\mathrm{root}}, I_{\mathrm{root}}, L_{\mathrm{root}}, R_{\mathrm{root}}, Q$

Use Definition 3632.44, Definition 3632.45, and Definition 3632.48. They place derivative, integral, and limit consumers behind the graph row, the relevant operation row, and the common regular-rational, dyadic, Real, transport, replay, provenance, and naming support.

The package $\Pi_{\mathsf{Calculus}^{\uparrow}}$ is exactly the union of those local and root rows. Since the cited obligations exhaust the operation ledger, a downstream consumer has no remaining coordinate from which to read calculus data outside the package. □

Theorem 3632.10 (Calculus root package nonescape)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root public package exports no host function space, selected antiderivative, selected derivative operator, quotient real equality of functions, host limit operator, host integral functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public derivative, integral, limit, continuous-map, or Real-facing read remains inside the derivative, Riemann finite-sum,* $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local naming rows of* $\Pi_{\mathsf{Calculus}^{\uparrow}}$.

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootPackageNonescape`

Proof. Read the excluded objects against Definition 3632.5. None is a row of $\Pi_{\mathsf{Calculus}^{\uparrow}}$, and the package comparison only inspects the displayed local and root rows.

For derivative and integral reads, apply Definition 3632.13 and Definition 3632.45. They keep derivative and Riemann finite-sum data in separate source rows and refuse a host fundamental theorem or selected antiderivative.

For limit, continuous-map, and Real-facing reads, apply Definition 3632.4, Definition 3632.42, and Definition 3632.46. These place the read behind the continuous-map, limit, regular-rational, dyadic, and Real-seal rows before anything is public.

Thus every exported calculus conclusion is obtained through the package rows. The excluded host structures remain absent, so the package cannot supply them by projection. □

Theorem 3632.11 (Calculus RegSeqRat dyadic error route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *Real-facing error estimate exposed by an accepted local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *passes through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is consumed. The route is finite packet readback and not a host analytic estimate, quotient real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegSeqRatDyadicErrorRoute`

Proof. Project the local-linearization carrier. The rows R and D are the only coordinates that record rational readback and dyadic tolerance data for a Real-facing calculus estimate.

The derivative source row L and the Riemann finite-sum source row J both sit before the terminal seal E. Their estimates can be transported and replayed only by the displayed structural rows H, C, P, N.

Thus any accepted Real-facing error estimate must read R, D before E. Since the carrier has no coordinate for host analysis, quotient real equality, or ambient completeness, no other route can supply the estimate. □

Theorem 3632.12 (Calculus ContinuousMap limit route)**.** *Every continuous-map limit read exposed by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier must display the graph and pointwise replay data of* $\mathsf{ContinuousMap}^{\uparrow}$ *before any* $\mathsf{Limit}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is consumed. In the local-linearization packet this means the derivative or Riemann source row is read only after its graph data has been routed through* R, D, H, C, P, N *toward* E*.*

Proof. Read the carrier rows in order. The local-linearization packet has no separate host function-space coordinate; graph and pointwise replay data can appear only through the displayed calculus source rows and their replay row C.

The real-seal obligation already confines limit-facing reads to the regular-rational row R, dyadic tolerance row D, one of the calculus source rows L, J, and the seal E. Therefore a continuous-map consumer cannot reach the Limit or Real seal without first displaying the graph data attached to that source row.

The transport, provenance, and local naming rows preserve the same dependency order. Hence the route is a finite ContinuousMap-to-Limit-to-Real read, not an ambient continuity theorem or a hidden limit operator. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuousMapLimitRoute`

Theorem 3632.13 (Calculus Riemann derivative separation)**.** *In every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$*, the* $\mathsf{Derivative}^{\uparrow}$ *row* L *and the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite-sum row* J *remain separate consumers of the common displayed support* E, R, D, H, C, P, N*. The shared support does not identify derivative and integral conclusions and does not supply a host fundamental theorem between them.*

Proof. Project the carrier. The derivative data is stored in L, and the finite tagged-sum data is stored in J; these are distinct coordinates of the finite packet.

Both rows may use the same real seal, rational readback, dyadic tolerance, transport, replay, provenance, and local naming support. Sharing those support rows only records common BEDC infrastructure; it does not turn one source row into the other.

Since the carrier contains no coordinate for a host fundamental theorem, selected antiderivative, quotient function equality, or ambient real completeness, derivative and Riemann finite-sum consumers remain separated after projection. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannDerivativeSeparation`

Theorem 3632.14 (Calculus Real-seal error budget)**.** *Every derivative, integral, or limit read exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier* $\Lambda = (E, R, D, L, J, H, C, P, N)$ *exposes a source graph or finite-sum row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any calculus conclusion is public. The exposed route is one of*

$$C, L, R, D, E, H, P, N \qquad \text{or} \qquad C, J, R, D, E, H, P, N,$$

according to whether the consumer asks for a derivative/limit read or a Riemann finite-sum read, and it has no coordinate for host limits, quotient function equality, selected antiderivatives, or ambient real completeness.

Proof. Project the local-linearization carrier. Its calculus-facing source rows are the derivative row L, the Riemann finite-sum row J, the continuous-map graph row implicit in C, the regular-rational readback row R, the dyadic tolerance row D, and the real seal E.

Apply Definition 3632.2 to derivative-shaped reads and Definition 3632.3 to integral-shaped reads. Apply Definition 3632.30 and Definition 3632.12 for limit and continuous-map reads.

The real-seal route is then forced by Definition 3632.4. The separating theorem Definition 3632.13 prevents the finite-sum row from supplying derivative data and prevents the derivative row from supplying integral data.

The structural rows H, C, P, N transport, replay, source, and name the same exposed inventory. Since the carrier lacks the excluded host coordinates, the listed rows form the full public error budget before any calculus conclusion is read. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealSealErrorBudget`

Theorem 3632.15 (Calculus source-window obligation)**.** *For an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier, every derivative, integral, and limit consumer must display the finite source window on which the read depends before it reaches a public conclusion. A derivative read displays a difference-quotient window, an integral read displays a partition and tagged finite-sum window, and a limit read displays the* $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *window rows, all with* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming.*

Proof. Start from the carrier rows of Definition 3632.21 and the local-linearization rows of Definition 3632.1. They display the finite source rows D, I, L, J together with the regular-rational, dyadic, Real, transport, replay, provenance, and naming support.

For derivative reads, use Definition 3632.2 and Definition 3632.33. The conclusion remains behind the displayed difference-quotient source, the continuous-map route, the regular-rational readback, and the dyadic tolerance data.

For integral reads, use Definition 3632.3 and Definition 3632.41. The finite partition, tagged-sum, endpoint, and readback rows are visible before the Real boundary is consumed.

For limit reads, use Definition 3632.30, Definition 3632.12, and Definition 3632.42. These force the continuous-map and limit windows to remain inside the displayed BEDC route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusSourceWindowObligation`

Theorem 3632.16 (Calculus root-unblock limit window scope)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *limit consumer that exports a* $\mathsf{Real}^{\uparrow}$*-facing conclusion must expose the* $\mathsf{Limit}^{\uparrow}$ *window row, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-replay row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming before the conclusion is public. The consumer cannot replace those rows by a host limit operator, quotient real equality, selected function-space theorem, or ambient completeness principle.*

Proof. Project the carrier rows of Definition 3632.21. The limit route is L, the continuous-map route is C, the regular-rational readback is Q, the terminal real boundary is R, and the structural rows are H, T, P, N.

Use Definition 3632.15 to require the finite source window before any public read. For a limit-shaped consumer, Definition 3632.30 and Definition 3632.12 identify the required $\mathsf{Limit}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows.

Finally apply Definition 3632.42 and Definition 3632.46. They keep the read behind the regular-rational, dyadic, transport, replay, provenance, naming, and Real-seal rows. Since the excluded host objects are absent from the carrier, the listed rows form the full limit-window scope. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockLimitWindowScope`

Theorem 3632.17 (Calculus root-unblock finite-sum handoff)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *integral consumer that exports a* $\mathsf{Real}^{\uparrow}$*-facing conclusion must pass through the* $\mathsf{RiemannIntegral}^{\uparrow}$ *finite-sum row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the conclusion is public. The consumer cannot bypass the finite-sum row by selecting an antiderivative, a host integration functional, quotient equality of functions, or ambient real completeness.*

Proof. Project Definition 3632.21. The integral source is I, the common regular-rational readback is Q, the terminal Real boundary is R, and the transport, replay, provenance, and naming rows are H, T, P, N.

Apply Definition 3632.15 and Definition 3632.3. Together they force the partition, tagged finite sum, endpoint handoff, regular-rational readback, dyadic tolerance data, and structural rows to be displayed before any Real-facing integral read is exported.

The handoff is separated from derivative data by Definition 3632.45. The Real seal is then controlled by Definition 3632.41 and Definition 3632.46, so the public conclusion remains a finite-sum-to-regular-rational-to-Real route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockFiniteSumHandoff`

Theorem 3632.18 (Calculus root-unblock continuous derivative window)**.** *Every derivative-facing* $\mathsf{Calculus}^{\uparrow}$ *root read that reaches a public* $\mathsf{Real}^{\uparrow}$*-facing conclusion must expose the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{Derivative}^{\uparrow}$ *difference row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming before the derivative conclusion is public. It cannot replace those rows by a host function space, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3632.21. The continuous-map route is C, the derivative source is D, the regular-rational readback is Q, the terminal Real boundary is R, and the structural rows are H, T, P, N.

Use Definition 3632.15 to force the finite derivative source window before any public read. Then apply Definition 3632.2, Definition 3632.33, and Definition 3632.12 to keep the derivative read behind the displayed graph and difference rows.

The dyadic and Real boundary is supplied by Definition 3632.11, Definition 3632.4, and Definition 3632.46. Since Definition 3632.52 excludes the host coordinates, the listed rows form the full continuous derivative window. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockContinuousDerivativeWindow`

Theorem 3632.19 (Calculus root-unblock Riemann endpoint readback)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *endpoint read consumed by a* $\mathsf{Calculus}^{\uparrow}$ *root integral conclusion must expose the finite tagged-sum row, the endpoint* $\mathsf{StreamName}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any integral conclusion is public. The endpoint read is a finite sum and readback route, not a host integration functional or selected antiderivative.*

Proof. Project the integral route of Definition 3632.21. The integral source is I, the shared regular-rational readback is Q, the Real boundary is R, and the transport, replay, provenance, and naming rows are H, T, P, N.

Apply Definition 3632.3, Definition 3632.41, and Definition 3632.17. These theorems place partition, tag, endpoint, finite-sum, regular-rational, dyadic, and Real-seal data before the public integral read.

Finally use Definition 3632.13 and Definition 3632.10. They prevent the endpoint read from being supplied by derivative data, a host integral functional, a selected antiderivative, quotient equality, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRiemannEndpointReadback`

Theorem 3632.20 (Calculus root-unblock RegSeqRat Real-seal factorization)**.** *Every derivative, integral, and limit consumer exported by the* $\mathsf{Calculus}^{\uparrow}$ *root package factors first through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No such consumer may use a host function space, quotient equality of functions, selected antiderivative, selected derivative operator, host limit operator, host integration functional, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness in place of the* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *factorization.*

Proof. Read the common carrier rows from Definition 3632.21 and the root package rows from Definition 3632.5. In both inventories, derivative, integral, and limit data reach a Real-facing conclusion only through the regular-rational row and the Real boundary row.

For derivative reads, apply Definition 3632.18. For integral reads, apply Definition 3632.17 and Definition 3632.19. For limit reads, apply Definition 3632.16.

The common factorization is then exactly the one recorded by Definition 3632.11, Definition 3632.4, and Definition 3632.46. The nonescape statements Definition 3632.52 and Definition 3632.10 exclude every listed host replacement. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRegSeqRatRealSealFactorization`

Definition 3632.21 (Calculus carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, L, C, D, I, Q, H, T, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar boundary used by all endpoint values, tolerances, finite sums, and error estimates.*
2. *L is the* $\mathsf{Limit}^{\uparrow}$ *route for displayed convergence windows and downstream real seals.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the finite graph, pointwise continuity, and replay windows available to calculus consumers.*
4. *D is the* $\mathsf{Derivative}^{\uparrow}$ *local difference-quotient row, including the source point, target read, and remainder ledger.*

5. *I is the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row, exposing the partition, tagged-read, and endpoint handoff data used by integration consumers.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row shared by the limit, derivative, and integral ledgers.*
7. *H records componentwise* hsame *transport for real, limit, continuous-map, derivative, integral, regular-rational, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay from real reads to limits, from limits to continuous maps, from finite difference quotients to derivative reads, and from finite sums to integral handoffs.*
9. *P is the* Pkg *provenance over* $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row.*

The classifier reads only $R, L, C, D, I, Q, H, T, P, N$. It has no coordinate for a host function space, a selected global derivative operator, an unbounded integration functional, quotient equality of functions, an open-cover compactness principle, or ambient real completeness.

Theorem 3632.22 (Calculus NameCert obligations)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *surface consists exactly of carrier admission, real-boundary exposure, limit-window readback, continuous-map replay, local-linear approximation routing, finite difference quotient readback, integral finite-sum handoff, regular-rational readback, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3632.21. The rows $R, L, C, D, I, Q, H, T, P, N$ are the full finite inventory available to a calculus consumer.

The local-linear approximation route is carried by D and is read only after the real boundary R, limit route L, continuous-map row C, and regular-rational readback Q have supplied the displayed point, tolerance, and remainder data. Thus a derivative-facing consumer sees a finite difference quotient route rather than a hidden global derivative operator.

The integral handoff is carried by I. Its partition, finite-sum, and endpoint rows are exposed before any real-facing value is consumed, and the same Q row supplies the rational readback needed to compare finite sums with limit and derivative windows.

Finally H, T, P, N transport, replay, source, and name exactly the displayed rows. Since no coordinate contains host function-space equality, ambient completeness, a selected derivative operator, or an unbounded integral functional, every accepted read remains inside the finite calculus packet. □

Definition 3632.23 (Calculus completion-source envelope)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *completion-source envelope is the finite row inventory read jointly from Definition 3632.21 and Definition 3632.1. It consists of the displayed BEDC rows*

$$\mathsf{Real}^{\uparrow},\ \mathsf{Limit}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Derivative}^{\uparrow},\ \mathsf{Integral}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The $\mathsf{Real}^{\uparrow}$ *row is the terminal seal for endpoint values and tolerances. The* $\mathsf{Limit}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows expose convergence windows and graph replay. The* $\mathsf{Derivative}^{\uparrow}$ *and* $\mathsf{Integral}^{\uparrow}$ *rows expose finite difference-quotient, remainder, partition, tagged-sum, and endpoint handoff data. The* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows carry rational readback and dyadic tolerance data before any Real-facing seal is consumed. The remaining* hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows transport, replay, source, and name exactly this finite packet.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusCompletionSourceEnvelope`

Theorem 3632.24 (Calculus completion-source envelope obligations)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier and every accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier, any derivative, integral, or limit consumer must expose the completion-source envelope rows of Definition 3632.23 before it reaches a public calculus conclusion. Thus a calculus conclusion may read only through displayed Real, Limit, ContinuousMap, Derivative, Integral, RegSeqRat, DyadicRatCore,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and not through a host limit operator, quotient function equality, selected global derivative operator, unbounded integration functional, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCompletionSourceEnvelopeObligations`

Proof. Project the finite rows of Definition 3632.21 and Definition 3632.1. Together they expose the Real boundary, limit route, continuous-map replay, derivative source, integral source, regular-rational readback, dyadic tolerance data, transport, replay, provenance, and local naming listed in Definition 3632.23.

Apply Definition 3632.22. It identifies carrier admission, real-boundary exposure, limit-window readback, continuous-map replay, local-linear approximation routing, finite difference-quotient readback, integral finite-sum handoff, regular-rational readback, componentwise transport, displayed replay, provenance, and naming as the local NameCert surface.

Now apply Definition 3632.15. Derivative, integral, and limit consumers must display their finite source windows with the regular-rational, dyadic, transport, replay, provenance, and naming rows before any public conclusion is read. The excluded host objects have no row in either carrier, so the envelope is the full public route. □

Theorem 3632.25 (Calculus completion-source real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing calculus read from an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier or accepted* $\mathsf{Calculus}^{\uparrow}$ *local-linearization carrier factors through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows of Definition 3632.23 before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. In particular, derivative, integral, and limit reads may hand a value to the Real boundary only after the displayed regular-rational readback and dyadic tolerance route have been exposed inside the finite BEDC packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCompletionSourceRealReadback`

Proof. Use Definition 3632.11. It forces every Real-facing error estimate in the local-linearization carrier through the regular-rational readback row and dyadic tolerance ledger before the terminal Real seal is read.

Use Definition 3632.14. The derivative, integral, and limit routes expose a source graph or finite-sum row together with regular-rational readback, dyadic tolerance data, and the Real seal before any calculus conclusion is public.

Finally apply Definition 3632.52. Public local operation reads cannot replace the displayed packet by a host function space, host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient Real completeness. Therefore every Real-facing calculus read factors through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ before reaching $\mathsf{Real}^{\uparrow}$. □

Theorem 3632.26 (Calculus continuous-map graph replay obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a continuous-map read is public only after the consumer has displayed the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, the* $\mathsf{Limit}^{\uparrow}$ *window route* L*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*, provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. A derivative or integral consumer using that continuous-map read must therefore factor through* R, L, C, Q, H, T, P, N *before reading* D *or* I*.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuousMapGraphReplayObligation`

Proof. Project the carrier of Definition 3632.21. The only row that names a continuous-map graph is C, and its endpoint values are typed by the real boundary R and compared through the regular-rational readback Q.

The limit route L supplies the finite convergence windows through which the graph row is replayed. The rows H, T, P, N transport, replay, source, and name the same displayed graph data.

Now apply Definition 3632.22. Since the derivative row D and integral row I are later consumers of the continuous-map graph, neither can use C until the route through R, L, C, Q, H, T, P, N is visible. □

Theorem 3632.27 (Calculus derivative-integral separation obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, the derivative row* D *and the integral row* I *are separate finite rows. The row* D *may expose local difference-quotient and remainder data only through* R, L, C, Q, H, T, P, N*, while the row* I *may expose finite-sum and endpoint handoff data only through those same support rows. Neither row supplies the other's conclusion, and the carrier contains no host fundamental theorem connecting them.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeIntegralSeparationObligation`

Proof. Project Definition 3632.21. The derivative data is stored in D, and the integral data is stored in I; they are distinct coordinates of the finite packet.

Use Definition 3632.2 for the derivative-facing route and Definition 3632.3 for the integral-facing route. Each theorem reads its own calculus row together with the real, limit, continuous-map, regular-rational, transport, replay, provenance, and naming support.

Definition 3632.22 lists the full local surface and includes no coordinate for a host fundamental theorem. Hence a consumer cannot derive an integral conclusion from D, cannot derive a derivative conclusion from I, and cannot import a global theorem to identify the two reads. □

Theorem 3632.28 (Calculus RegSeqRat readback root obligation)**.** *Every real-facing value exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must display the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *together with the* $\mathsf{Real}^{\uparrow}$ *boundary* R *and the dyadic tolerance route inherited by the regular-rational readback. This readback is required before any limit, derivative, or integral read is exported from* L*,* D*, or* I*.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegSeqRatReadbackRootObligation`

Proof. Begin with the carrier of Definition 3632.21. The terminal real boundary is R, and the common rational readback row is Q; no other coordinate converts limit, derivative, or integral data into a real-facing value.

The earlier real-seal theorem Definition 3632.4 already forces Real-facing limit reads through regular-rational and dyadic tolerance data. The derivative and integral obligations similarly keep their finite quotient and finite-sum reads attached to the displayed real and rational support rows.

Finally use Definition 3632.22. The local NameCert surface lists regular-rational readback as a root obligation, so every export from L, D, I must first pass through Q with R and the underlying dyadic tolerance route visible. □

Theorem 3632.29 (Calculus root regular readback obligation)**.** *Every derivative, integral, or limit export from an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the inherited* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route before the terminal*

$\mathsf{Real}^{\uparrow}$ *seal R is consumed. The export is a displayed calculus packet read and not a host analytic estimate, quotient real equality, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRegularReadbackObligation`

Proof. Project the carrier in Definition 3632.21. The limit row is L, the derivative row is D, the integral row is I, the common regular-rational readback row is Q, and the terminal Real boundary is R.

Apply Definition 3632.28. It already requires every real-facing value exported from L, D, I to display Q together with R and the dyadic tolerance route inherited by the regular-rational readback.

Apply Definition 3632.30. It forces a continuous-map graph read and a limit-window read to remain visible before any Real seal is exported, with Q supplying the readback used by R.

Thus no derivative, integral, or limit consumer can replace the displayed readback by a host analytic estimate, quotient real equality, or ambient Real completeness. The root obligation is exactly the route through Q and its dyadic tolerance data before R is consumed. □

Theorem 3632.30 (Calculus continuous-limit carrier budget)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, a continuous-map graph read and a limit-window read must both be displayed before any* $\mathsf{Real}^{\uparrow}$ *seal is exported. The graph row C supplies the named finite graph and pointwise replay windows, the limit row L supplies the convergence windows, and the regular-rational row Q supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the terminal real boundary R. Thus a real-facing calculus export cannot bypass C, L, Q or treat Real completeness as an ambient carrier.*

Proof. Project the carrier of Definition 3632.21. The only continuous-map graph coordinate is C, the only limit-window coordinate is L, and the only regular-rational readback coordinate is Q.

Use Definition 3632.26. It requires R, L, C, Q, H, T, P, N before a derivative or integral consumer can read the continuous-map graph.

Use Definition 3632.28. Every real-facing export from L, D, I must expose Q together with R and the dyadic tolerance route. Hence the Real seal is downstream of the displayed graph, limit, and readback rows rather than an ambient completeness principle. □

Theorem 3632.31 (Calculus derivative-integral source separation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, the derivative row D and integral row I remain separated even when they share the source rows R, L, C, Q, H, T, P, N. A derivative consumer must read local difference-quotient and remainder data from D, while an integral consumer must read partition, tagged-sum, and endpoint handoff data from I; no shared support row identifies the two sources or exports a host fundamental theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeIntegralSourceSeparation`

Proof. Project Definition 3632.21. The derivative source is the coordinate D, and the integral source is the coordinate I.

Apply Definition 3632.27. That theorem already records that the two rows use the same finite support but do not supply each other's conclusions.

The carrier list contains no coordinate for quotient function equality, a selected antiderivative, or a host fundamental theorem. Therefore shared support through R, L, C, Q, H, T, P, N preserves source separation rather than collapsing D and I. □

Theorem 3632.32 (Calculus real RegSeqRat error handoff)**.** *Every real-facing error estimate exported by an accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$ *must first display the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the dyadic rational tolerance route inherited by that readback. Limit, derivative, and integral consumers may then hand the estimate to the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, but only after the regular-rational and dyadic data have been read inside the finite packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealRegseqratErrorHandoff`

Proof. Begin with Definition 3632.28. It makes Q mandatory before any real-facing value is exported from L, D, I.

The derivative and integral routes use the same readback discipline. Their finite quotient, remainder, partition, tagged-sum, and endpoint rows are compared through regular-rational data before a Real-facing estimate is named.

Finally apply Definition 3632.22. The local NameCert surface contains $R, L, C, D, I, Q, H, T, P, N$ and no ambient Real-completeness or host error functional. Thus the handoff to R is downstream of $\mathsf{RegSeqRat}^{\uparrow}$ and dyadic readback data in Q. □

Theorem 3632.33 (Calculus derivative-limit root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a derivative-limit read is public only after the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*, the* $\mathsf{Derivative}^{\uparrow}$ *quotient row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the dyadic tolerance ledger inherited by* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *have all been displayed. The read exports a finite BEDC derivative-limit route and not a host function space, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusDerivativeLimitObligation`

Proof. Project the carrier of Definition 3632.21. The derivative source is the row D, while the limit-facing route is L and the graph replay available to a derivative consumer is C.

Use Definition 3632.26. It places every derivative consumer behind R, L, C, Q, H, T, P, N, so the graph and pointwise replay are visible before the derivative row is consumed.

Use Definition 3632.28. The regular-rational readback Q carries the dyadic tolerance route required before any Real-facing value reaches R.

The remaining rows H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host function space, quotient function equality, selected derivative operator, selected antiderivative, or ambient completeness, the derivative-limit obligation is exactly this displayed route. □

Theorem 3632.34 (Calculus integral-sum root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, an integral read may consume the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row* I *only after the* $\mathsf{Real}^{\uparrow}$ *boundary* R*,* $\mathsf{Limit}^{\uparrow}$ *route* L*,* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, dyadic tolerance ledger, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *are visible. The finite-sum row records partition, tag, sum, and endpoint handoff data; it does not export a host integration functional, quotient equality of functions, a selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusIntegralSumObligation`

Proof. Read the carrier K and isolate the integral source row I. The statement of Definition 3632.22 lists integral finite-sum handoff as one local NameCert obligation, so I is a displayed row rather than a host integral object.

Apply Definition 3632.3. It keeps the partition, tag, finite-sum, and endpoint data attached to the real boundary, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows.

Apply Definition 3632.26. If the integral consumer reads graph data, it reaches that graph through R, L, C, Q, H, T, P, N before the finite-sum handoff is public.

The derivative row D remains separate by Definition 3632.31. Thus the integral-sum obligation is the finite route through I and the displayed support rows, with no selected antiderivative, host function space, quotient equality, or ambient Real-completeness coordinate. □

Theorem 3632.35 (Calculus continuity real-readback root obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, a continuity-shaped Real read must display the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row* C*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the dyadic tolerance ledger inherited by* Q*, and the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *before derivative or integral data is consumed. The structural rows* H, T, P, N *preserve this same route through* hsame*,* Cont*,* Pkg*, and* NameCert*, and no host continuity theorem, function-space equality, quotient real equality, selected antiderivative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness is exported.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusContinuityRealReadbackObligation`

Proof. Start with Definition 3632.30. It identifies C, L, Q as mandatory displayed rows before any $\mathsf{Real}^{\uparrow}$ seal is exported from the calculus carrier.

Use Definition 3632.32. The real-facing estimate must pass through Q and the dyadic rational tolerance route before it reaches R.

Derivative and integral reads remain downstream consumers. The row D supplies only finite difference-quotient data, and the row I supplies only finite-sum data, so neither can replace the continuity-to-limit-to-readback route.

Finally read H, T, P, N. These rows give componentwise transport, displayed continuation replay, package provenance, and local naming for the same packet. The carrier therefore exports a finite continuity real-readback route and no host continuity theorem, function-space equality, quotient real equality, selected antiderivative, or ambient completeness principle. □

Theorem 3632.36 (Calculus root chain-rule window obligation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a chain-rule-shaped read for composed local maps must display the outer and inner* $\mathsf{ContinuousMap}^{\uparrow}$ *graph windows in* C*, the two* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient windows in* D*, the* $\mathsf{Limit}^{\uparrow}$ *route* L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The read is a finite window-composition obligation, not a host chain-rule theorem over an ambient function space.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootChainRuleWindowObligation`

Proof. Project the root carrier of Definition 3632.43. The graph data for a composed read can only be displayed through C, while derivative data can only be read through D.

Use Definition 3632.44 to place every derivative, integral, or limit consumer behind the $\mathsf{ContinuousMap}^{\uparrow}$ graph row. Use Definition 3632.33 to keep the derivative-limit read behind $R, L, C, D, Q, H, T, P, N$.

The regular-rational and dyadic rows Q, Y supply the comparison window before R is consumed. The structural rows H, T, P, N transport, replay, source, and name the same finite packet.

Thus the chain-rule-shaped obligation is a displayed composition of two local derivative windows. Since the carrier has no coordinate for host function-space equality or a global derivative operator, no ambient chain-rule theorem is exported. □

Theorem 3632.37 (Calculus root fundamental-theorem refusal boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a consumer that reads both the* $\mathsf{Derivative}^{\uparrow}$ *row* D *and the* $\mathsf{Integral}^{\uparrow}$ *row* I *must keep them as separate boundary rows over the common* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *support. The packet may expose compatible finite source windows, but it does not identify* D *with* I*, select an antiderivative, or claim the full fundamental theorem of calculus.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootFtcBoundaryRefusal`

Proof. Project D and I from Definition 3632.43. The derivative source and integral source are distinct coordinates of the same finite BHist packet.

Apply Definition 3632.45. It states that D and I remain separate over $C, L, Q, Y, R, H, T, P, N$ and that neither row supplies the other's conclusion.

The real-facing support is controlled by Definition 3632.46. Any value read from either source must pass through Q, Y before reaching R.

Therefore the carrier records a refusal boundary: derivative and integral rows can share finite support without becoming a fundamental-theorem certificate. Selected antiderivatives, quotient equality of functions, and a host theorem relating all derivatives and integrals are absent from the packet. □

Theorem 3632.38 (Calculus root local-linearity composition)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a local-linearity composition read must route the source* $\mathsf{ContinuousMap}^{\uparrow}$ *graph windows through* C*, the two local* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient rows through* D*, the comparison windows through* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows* L, Q, Y*, and the terminal value through the* $\mathsf{Real}^{\uparrow}$ *boundary* R*, with componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*. The conclusion is only that the finite local-linear windows compose inside the displayed BEDC packet.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLocalLinearityComposition`

Proof. Begin with the graph route. By Definition 3632.44, every local operation consumer must display C before reading derivative, integral, or limit data.

For the local-linear source, use Definition 3632.36. It gives the two derivative windows and their limit/readback support through D, L, Q, Y before R is consumed.

The rows H, T, P, N then transport, replay, source, and name this composed finite window. No row of G supplies a host function space, quotient equality of functions, selected global derivative operator, or ambient Real completeness theorem.

Thus local linearity composes only as a finite BEDC dependency route: graph windows, derivative windows, regular-rational and dyadic comparison windows, and the terminal Real boundary remain visible in the same packet. □

Definition 3632.39 (Calculus root derivative-integral carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier is the carrier*

$$K = (R, L, C, D, I, Q, H, T, P, N)$$

of Definition 3632.21 read with the following root dependency rows singled out:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *boundary for endpoint, value, error, derivative, integral, and limit readback.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row carrying the rational comparison and error schedule used before R is consumed.*
3. *the dyadic tolerance data inherited by Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dependency row for denominator, radius, and comparison reads;*
4. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise replay row used before derivative, integral, or limit consumers read function data;*
5. *D, I, L, H, T, P, N are respectively the derivative source row, integral finite-sum row, limit route, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row.*

The root classifier compares accepted packets only through these displayed BHist, hsame, Cont, NameCert, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows. It has no coordinate for a host function space, quotient equality of functions, a selected derivative operator, a selected antiderivative, or ambient real completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootDerivativeIntegralCarrier`

Theorem 3632.40 (Calculus root derivative real handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, a derivative-facing real read must expose the* $\mathsf{ContinuousMap}^{\uparrow}$ *row C, the* $\mathsf{Derivative}^{\uparrow}$ *row D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data inherited by Q, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row N before the* $\mathsf{Real}^{\uparrow}$ *boundary R is consumed.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootDerivativeRealHandoff`

Proof. Project the root carrier of Definition 3632.39. The derivative source is D, but D is public only as part of the same BHist packet as the graph row C, readback row Q, transport row H, replay row T, provenance row P, and local naming row N.

Use Definition 3632.33. It already places every derivative-limit read behind the $\mathsf{ContinuousMap}^{\uparrow}$ graph row, the regular-rational readback, the dyadic tolerance route, and the structural rows.

The terminal real boundary is R. Since no coordinate of the root carrier contains quotient function equality, a selected derivative operator, or ambient real completeness, the derivative-facing handoff reaches R only through the displayed finite route. □

Theorem 3632.41 (Calculus root integral RegSeqRat handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier $K = (R, L, C, D, I, Q, H, T, P, N)$, an integral-facing read through the* $\mathsf{Integral}^{\uparrow}$ *row I must first display the* $\mathsf{ContinuousMap}^{\uparrow}$ *row C, the finite-sum and endpoint handoff data in I, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route inherited by Q, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row N. Only after this readback may the consumer reach the* $\mathsf{Real}^{\uparrow}$ *boundary R.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootIntegralRegSeqRatHandoff`

Proof. Read the integral source row I from Definition 3632.39. Its partition, tag, finite-sum, and endpoint data are displayed inside the same finite BHist packet as C, Q, H, T, P, N.

Apply Definition 3632.34. That obligation forces the integral row through the real boundary, limit route, continuous-map replay, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is therefore not optional. The consumer cannot replace Q by a host integration functional, selected antiderivative, quotient function equality, or ambient completeness theorem before reading R. □

Theorem 3632.42 (Calculus root limit nonescape)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root derivative-integral carrier* $K = (R, L, C, D, I, Q, H, T, P, N)$*, each limit-shaped read exported by* L *must remain inside the route through the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route inherited by* Q*, the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. It cannot escape to a host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitNonescape`

Proof. Project the limit row L from the root carrier. By construction, L is not a standalone host limit object; it is stored together with C, Q, R, H, T, P, N in the displayed calculus packet.

Use Definition 3632.30. A continuous-map graph read and a limit-window read must both be displayed before any real-facing seal is exported, and the regular-rational row Q supplies the readback used by R.

The structural rows preserve the same route through hsame, Cont, Pkg, and $\mathsf{NameCert}$. Since the carrier has no coordinate for the excluded host principles, every limit-shaped read remains confined to the listed BEDC dependency rows. □

Definition 3632.43 (Calculus root carrier)**.** *A* $\mathsf{Calculus}^{\uparrow}$ *root carrier is a finite* BHist *packet*

$$G = (C, D, I, L, R, Q, Y, H, T, P, N)$$

whose rows are the $\mathsf{ContinuousMap}^{\uparrow}$ *graph and replay surface* C*, the* $\mathsf{Derivative}^{\uparrow}$ *difference-quotient row* D*, the* $\mathsf{Integral}^{\uparrow}$ *finite-sum row* I*, the* $\mathsf{Limit}^{\uparrow}$ *window row* L*, the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and denominator ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N*. The root classifier compares accepted packets only through these finite rows. It has no coordinate for a host limit operator, quotient equality of functions, a selected derivative operator, a selected antiderivative, or ambient Real completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootCarrier`
Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootCarrier admission`

Theorem 3632.44 (Calculus root ContinuousMap handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, any derivative, integral, or limit consumer must display the* $\mathsf{ContinuousMap}^{\uparrow}$ *row* C *before it reads* D*,* I*, or* L *and before it consumes the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *rows. The handoff is a finite graph-and-replay read through* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows, not an ambient function space or host continuity theorem.*

Proof. Project the root carrier of Definition 3632.43. The only coordinate that names graph and pointwise replay data is the row C, so calculus consumers cannot read derivative, integral, or limit data as graph data unless C is visible.

Use Definition 3632.26. It already places derivative and integral consumers behind the continuous-map graph route of the calculus packet.

The rows H, T, P, N preserve that same graph route under transport, replay, provenance, and local naming. Since G has no coordinate for a host function space or host continuity theorem, the handoff remains the displayed $\mathsf{ContinuousMap}^{\uparrow}$ row. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootContinuousMapHandoff`

Theorem 3632.45 (Calculus root derivative-integral boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, the* $\mathsf{Derivative}^{\uparrow}$ *row* D *and the* $\mathsf{Integral}^{\uparrow}$ *row* I *are separate boundary rows over the common support* $C, L, Q, Y, R, H, T, P, N$*. A derivative read may use finite difference-quotient data from* D*, and an integral read may use finite-sum data from* I*, but neither row supplies the other's conclusion or a host fundamental theorem.*

Proof. Read the two source rows from Definition 3632.43. The derivative source is D, and the integral source is I; they are distinct coordinates of the finite packet.

Apply Definition 3632.31. It states that derivative and integral sources remain separated even when they share the real, limit, continuous-map, regular-rational, transport, replay, provenance, and local naming support.

The dyadic tolerance row Y and readback row Q only provide comparison data before the terminal Real boundary R is consumed. They do not identify D with I and do not contain a selected antiderivative, quotient function equality, or host fundamental theorem. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootDerivativeIntegralBoundary`

Theorem 3632.46 (Calculus root Real-seal nonescape)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, every Real-facing export from derivative, integral, or limit data must pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The export remains inside the displayed finite packet and cannot escape to host limit operators, quotient function equality, or ambient Real completeness.*

Proof. Start with the root carrier. The terminal Real boundary is R, while Q and Y are the only regular-rational and dyadic tolerance rows available before that boundary is read.

Use Definition 3632.28. It requires every real-facing value exported from the calculus limit, derivative, and integral rows to display regular-rational readback before the Real boundary is consumed.

Use Definition 3632.42 for limit-shaped reads and Definition 3632.45 for derivative and integral source separation. Together they keep the route inside $C, D, I, L, Q, Y, R, H, T, P, N$.

Because the root carrier has no coordinate for host limit operators, quotient function equality, selected derivative or antiderivative operators, or ambient Real completeness, no Real-facing export can bypass Q, Y on the way to R. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRealSealNonescape`

Definition 3632.47 (Calculus root operation ledger)**.** *The* $\mathsf{Calculus}^{\uparrow}$ *root operation ledger attached to an accepted carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$ *is the displayed row inventory*

$$\mathsf{OpLedger}_{\mathsf{Calculus}^{\uparrow}}(G) = (C, D, I, L, R, Q, Y, H, T, P, N).$$

Here C *is the continuous-map graph and replay row,* D *is the derivative difference-quotient row,* I *is the integral finite-sum row,* L *is the limit-window row,* R *is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary,* Q *is*

the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* Y *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* H *is componentwise* hsame *transport,* T *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row. The ledger has no row for a host function space, quotient equality of functions, a selected antiderivative, a selected derivative operator, or ambient completeness.*

Lean4 (definition) `BEDC.Derived.CalculusUp.CalculusRootOperationLedger`

Theorem 3632.48 (Calculus root operation ledger exhaustion)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, every derivative, integral, limit, or Real-facing operation read is exhausted by the operation ledger of Definition 3632.47. A consumer may read derivative data only through* $C, D, Q, Y, R, H, T, P, N$*, integral data only through* $C, I, Q, Y, R, H, T, P, N$*, and limit data only through* $C, L, Q, Y, R, H, T, P, N$*. No operation read may replace those rows by a host function space, quotient equality of functions, selected antiderivatives, selected derivative operators, or ambient completeness.*

Proof. Begin with the root carrier of Definition 3632.43. It displays exactly the rows $C, D, I, L, R, Q, Y, H, T, P, N$, and Definition 3632.47 only repackages that same finite inventory as the operation ledger.

Apply Definition 3632.44. Every derivative, integral, and limit consumer must display C before reading D, I, or L.

For derivative and integral reads, apply Definition 3632.45. It keeps D and I as separate source rows over the common support and refuses a host fundamental theorem or selected antiderivative.

For limit and Real-facing reads, apply Definition 3632.42 and Definition 3632.46. They force the read through Q, Y before R is consumed and leave no coordinate for host limits, quotient function equality, selected derivative or antiderivative operators, or ambient Real completeness. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationLedgerExhaustion`

Theorem 3632.49 (Calculus root unblock Real operation ledger)**.** *The root* $\mathsf{Calculus}^{\uparrow}$ *operation ledger unblocks downstream derivative, integral, limit, and completion consumers only as the finite BEDC route*

$$C, D, I, L, R, Q, Y, H, T, P, N.$$

Any downstream consumer that asks for a Real-facing calculus value must display the relevant operation row D*,* I*, or* L*, the graph row* C*, the regular-rational readback* Q*, the dyadic tolerance ledger* Y*, the terminal Real boundary* R*, and the structural rows* H, T, P, N*. The ledger does not export ambient function spaces, quotient equality, selected antiderivatives, selected derivative operators, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Read the operation ledger from Definition 3632.47. The only public rows available to a downstream consumer are the displayed finite rows $C, D, I, L, R, Q, Y, H, T, P, N$.

Use the derivative-integral boundary of Definition 3632.45 for operation separation. It prevents the derivative row from supplying integral conclusions and prevents the integral row from supplying derivative conclusions.

Use the Real-seal nonescape theorem Definition 3632.46. It requires every Real-facing export from derivative, integral, or limit data to pass through Q, Y before R is consumed.

Finally apply Definition 3632.48. The exhaustion statement identifies the ledger as the full consumer-facing route, so the root unblock surface is a finite operation ledger and not a host analytic structure or ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootUnblockRealOperationLedger`

Theorem 3632.50 (Calculus root Real-completion operation handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, any consumer that combines derivative, integral, or limit rows with a Real-facing value must route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *before reading the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R*. The handoff is the Phase Real and Completion route of the finite root carrier, not an ambient completion or host analytic-value principle.*

Proof. Apply Definition 3632.45 to keep the derivative and integral rows separate over the common support.

Use Definition 3632.46 and Definition 3632.48. These force every derivative, integral, limit, or Real-facing read through Q, Y before R is public.

Finally apply Definition 3632.51 for the completion-facing derivative window. The structural rows H, T, P, N preserve the same finite route, so the consumer reaches Phase Real and Completion only through the displayed root ledger. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRealCompletionOperationHandoff`

Theorem 3632.51 (Calculus root local linearization Cauchy handoff)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a local-linearization consumer that asks for a Cauchy-facing derivative window must display the* BHist *derivative row* D*, the continuous graph row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *row* N *before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *is consumed. The handoff is a finite derivative-to-regular-readback route and not a host differentiability theorem, quotient equality of functions, or ambient Cauchy completeness claim.*

Proof. Project the root operation ledger. The derivative source for local linearization is the displayed row D, and any graph data used by that derivative source is routed through C.

Apply Definition 3632.40. It places every derivative-facing real read behind C, D, Q, H, T, P, N before R is consumed. Apply Definition 3632.46 to keep the same read behind the dyadic tolerance ledger Y and the regular-rational row Q.

The structural rows H, T, P, N preserve the same finite packet under hsame, Cont, Pkg, and local naming. Because the carrier has no coordinate for host differentiability, quotient function equality, or ambient completion, the Cauchy-facing handoff is exactly the displayed derivative, readback, tolerance, and Real-seal route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLocalLinearizationCauchyHandoff`

Theorem 3632.52 (Calculus local operation nonescape)**.** *Every public* $\mathsf{Calculus}^{\uparrow}$ *local operation read is confined to the displayed* BHist *rows for source graph data or finite sums,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming. No derivative, integral, or limit conclusion may replace those rows by a host function space, host limit operator, quotient equality of functions, selected derivative operator, selected antiderivative, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocalOperationNonescape`

Proof. Read the carrier inventory from Definition 3632.21. The local operations are stored only in the derivative, integral, and limit rows, and all Real-facing reads pass through the regular-rational and dyadic rows before the terminal seal is consumed.

Apply Definition 3632.44 to keep every operation behind its graph-and-replay row. Apply Definition 3632.45 and Definition 3632.13 to keep derivative and integral sources distinct.

The Real-facing boundary is controlled by Definition 3632.46 together with Definition 3632.14. The source-window requirement is exactly Definition 3632.15.

Thus each operation is exported only through the displayed BEDC packet. The excluded host and quotient objects are absent from the carrier, so there is no accepted escape route outside the finite local operation surface. □

Theorem 3632.53 (Calculus root continuity composition ledger)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a continuity-composition read for derivative, integral, or limit consumers must first display the* BHist *continuous-map row* C*, the operation row among* D, I, L*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, the terminal* $\mathsf{Real}^{\uparrow}$ *row* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*. The composition ledger exports no host continuity theorem, function-space equality, selected operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the continuous-map row C from the root carrier. By Definition 3632.44, every derivative, integral, or limit consumer must expose C before it reads D, I, or L.

For the operation side, apply Definition 3632.48. It separates derivative reads through D, integral reads through I, and limit reads through L, while keeping all three behind the common regular-rational, dyadic, Real, transport, replay, provenance, and naming rows.

The Real-facing portion is controlled by Definition 3632.46. Hence the continuity-composition ledger is a finite route through $C, D, I, L, Q, Y, R, H, T, P, N$, with no extra coordinate from which a host continuity theorem or ambient function-space equality could be read. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootContinuityCompositionLedger`

Theorem 3632.54 (Calculus root Riemann-sum Real seal)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, R, Q, Y, H, T, P, N)$*, a Riemann-sum-facing integral read must display the* BHist *finite-sum row* I*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *row* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, componentwise* hsame *transport* H*, displayed* Cont *replay* T*,* Pkg *provenance* P*, local* $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$ *naming* N*, and the terminal* $\mathsf{Real}^{\uparrow}$ *boundary* R *before any integral value is public. The seal is a finite Riemann row over L10 readback data and not a selected antiderivative, host integral functional, quotient equality of functions, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootRiemannSumRealSeal`

Proof. Project the integral source row I from the root carrier. It is the only row carrying finite-sum and endpoint handoff data for an integral-shaped read.

Apply Definition 3632.41. It forces the integral row through C, I, Q, H, T, P, N and then through the terminal Real boundary R. Apply Definition 3632.46 to keep the same route behind the dyadic tolerance ledger Y before R is consumed.

The derivative row D and limit row L remain separate by Definition 3632.45 and Definition 3632.48. Thus the Riemann-sum Real seal is exactly the finite route through $I, C, Q, Y, H, T, P, N, R$, with no selected antiderivative, host integration functional, quotient equality, or ambient completeness row. □

Theorem 3632.55 (Calculus root limit-derivative-integral triad)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, the derivative row* D*, integral row* I*, and limit row* L *form one finite root triad over the continuous-map source* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* Y*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R*, and the structural rows* H, T, P, N*. A consumer of any one of* D, I, L *must expose the whole route* $C, D, I, L, Q, Y, R, H, T, P, N$ *before exporting a Real-facing calculus value.*

Proof. Apply Definition 3632.44. It places every derivative, integral, and limit read behind the shared continuous-map row C.

Apply Definition 3632.45 to keep the derivative and integral coordinates distinct, and use Definition 3632.48 for the limit coordinate and the common ledger inventory.

Finally apply Definition 3632.54 and Definition 3632.46. The integral and Real-seal routes force the same Q, Y, R, H, T, P, N readback boundary, so the triad is the displayed finite carrier surface and not an ambient analytic operator family. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootLimitDerivativeIntegralTriad`

Theorem 3632.56 (Calculus root common Real readback)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, derivative, integral, and limit consumers share the same Real readback route: the chosen operation row among* D, I, L *is read only together with* C, Q, Y, R, H, T, P, N*. No operation row exports a Real value through a separate quotient representative, selected limiting object, or host completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootCommonRealReadback`

Proof. Start from Definition 3632.46. It requires every Real-facing calculus export to pass through Q, Y before the terminal row R is consumed.

Use Definition 3632.44 to attach that readback to C, and use Definition 3632.48 to identify the only operation rows as D, I, L.

The Riemann-sum boundary of Definition 3632.54 supplies the integral case, while Definition 3632.45 prevents that case from being reused as derivative data. The shared readback is therefore $C, D, I, L, Q, Y, R, H, T, P, N$ with the selected operation coordinate named by the consumer. □

This common readback consumes the Real and completion chain through chapter 38, chapter 301, and chapter 291. Those sibling rows provide the terminal seal, regular rational readback, and tolerance ledger for the calculus root route, while importing no ambient Real completeness theorem or host function space.

Theorem 3632.57 (Calculus root operation separation)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, the derivative, integral, and limit obligations are separated by the operation rows* D, I, L *while remaining tied to the same continuous-map, regular-rational, dyadic, Real, transport, replay, provenance, and naming rows* C, Q, Y, R, H, T, P, N*. Thus no accepted calculus consumer may infer a derivative claim from* I *or* L*, an integral claim from* D *or* L*, or a limit claim from* D *or* I *without reopening the displayed finite route.*

Proof. Apply Definition 3632.45. It already separates derivative reads through D from integral reads through I.

Apply Definition 3632.48. The exhaustion ledger adds the limit row L to the same separated inventory and gives no hidden coordinate from which one operation can supply another.

Use Definition 3632.44 and Definition 3632.46 to keep all three operation rows attached to C, Q, Y, R, H, T, P, N. The carrier route is therefore common at the readback boundary and separated at the operation boundary. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationSeparation`

Theorem 3632.58 (Calculus root chain-rule ContinuousMap RegSeqRat scope)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, a composable pair of local-linearization ledgers whose graph replay is accepted by* $\mathsf{ContinuousMap}^{\uparrow}$ *through* C *and whose dyadic error budgets descend through* $\mathsf{RegSeqRat}^{\uparrow}$ *via* Q, Y *produces a* $\mathsf{Real}^{\uparrow}$ *readback scope for the chain-rule window. The consumer may read only the finite route* $C, D, Q, Y, R, H, T, P, N$ *together with the two composable local-linearization rows; it may not infer the full fundamental theorem of calculus, a global derivative operator, a selected antiderivative, quotient equality of functions, or ambient Real completeness.*

Proof. Begin with the calculus root ContinuousMap handoff of Definition 3632.44. It forces the graph replay for each local-linearization row through the displayed continuous-map coordinate C before the derivative row D is consumed.

Use the derivative-integral boundary of Definition 3632.45. The chain-rule window remains on the derivative side of the root carrier and does not borrow an integral row or a host antiderivative claim.

Next apply the regular-rational and dyadic readback route supplied by Definition 3632.53 and Definition 3632.56. These keep the error budgets in Q, Y before any terminal $\mathsf{Real}^{\uparrow}$ readback row R is exposed.

Finally use Real-seal nonescape, Definition 3632.46. The resulting scope is the displayed finite route through $C, D, Q, Y, R, H, T, P, N$ for the composable local-linearization pair, not the full fundamental theorem of calculus or any ambient analytic completeness principle. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootChainRuleContinuousMapRegSeqRatScope`

Theorem 3632.59 (Calculus root operation readback tail chain-rule boundary)**.** *For every accepted* $\mathsf{Calculus}^{\uparrow}$ *root carrier* $G = (C, D, I, L, Q, Y, R, H, T, P, N)$*, any chain-rule readback that uses the derivative row* D *must retain the common route* $C, D, Q, Y, R, H, T, P, N$*. The derivative evidence cannot be supplied by the integral row* I *or the limit row* L*; those rows remain available only through their own integral and limit obligations.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRootOperationReadbackTailChainRuleBoundary`

Proof. Apply Definition 3632.58. It fixes the chain-rule window on the displayed derivative route through $C, D, Q, Y, R, H, T, P, N$ together with the composable local-linearization rows.

Next use Definition 3632.57. The separated operation ledger prevents an integral row I or a limit row L from being consumed as derivative evidence without reopening the corresponding operation obligation.

Finally apply Definition 3632.56. The Real-facing readback is the common finite BEDC route through the continuous-map, derivative, regular-rational, dyadic, Real, transport, replay, provenance, and naming rows. By Real-seal nonescape this route is not a host analytic chain-rule operator, selected integral, selected limit, quotient function equality, or ambient completeness principle. □

Lemma 3632.60 (Calculus derivative and Riemann-integral dependency route)**.** *Every substantive* $\mathsf{Calculus}^{\uparrow}$ *consumer of a derivative or integral conclusion reads the derivative branch through chapter 122, the Riemann finite-sum branch through chapter 1526, and the shared endpoint readback through the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rows. The route is therefore a displayed BEDC dependency through derivative windows, tagged Riemann sums, regular rational readback, and real-name sealing; it is not a host function space, a host measure-space integral, a selected antiderivative, or quotient equality of functions.*

Proof. Project the $\mathsf{Calculus}^{\uparrow}$ local-linearization and root carrier rows already named in Definition 3632.1 and Definition 3632.5. Their derivative-facing coordinates are the $\mathsf{Derivative}^{\uparrow}$ rows, and their integral-facing coordinates are the $\mathsf{RiemannIntegral}^{\uparrow}$ or finite-sum rows.

The shared numerical readback is not supplied by a host analytic object. It passes through the regular rational and real-name rows before a public endpoint value is exposed.

The existing nonescape theorems in the chapter exclude host function spaces, host integration functionals, selected antiderivatives, quotient equality of functions, and ambient completeness. Hence the dependency route for calculus consumers is the finite $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ route stated above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannIntegralDependencyRoute`

Theorem 3632.61 (Calculus derivative-integral dependency factorization)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer that combines a derivative conclusion with a Riemann-integral conclusion factors its public read through the* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before exposing endpoint equality or an estimate. In the notation of Definition 3632.1 and Definition 3632.5, the derivative branch consumes the displayed derivative rows, the integral branch consumes the displayed finite-sum rows, and the common endpoint readback consumes the regular rational and real-name rows before any equality or estimate is public.*

Proof. Start with the dependency route of Definition 3632.60. It already separates derivative-facing reads through chapter 122 from Riemann finite-sum reads through chapter 1526.

The local-linearization carrier Definition 3632.1 supplies the derivative-side window, while the root public package Definition 3632.5 supplies the shared public rows. Thus a consumer combining the two conclusions must first read the derivative and integral branches as displayed BEDC rows.

The endpoint or estimate then passes through the regular rational readback and terminal Real seal. This is the same nonescape boundary used by Definition 3632.52, Definition 3632.10, and Definition 3632.46.

Those nonescape statements exclude host function spaces, selected antiderivatives, quotient function equality, host integration functionals, and ambient completeness. Hence no combined derivative-integral conclusion can expose an endpoint equality or estimate before the finite $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ factorization has been consumed. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRiemannIntegralDependencyFactorization`

Theorem 3632.62 (Calculus finite-limit seal dependency)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *consumer that turns derivative or integral endpoint data into a limit-facing read must first pass through the finite* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *route*

$$M, W, R, D, E$$

of Definition 5722.5. The calculus consumer may read derivative windows through $\mathsf{Derivative}^{\uparrow}$ *and finite tagged sums through* $\mathsf{RiemannIntegral}^{\uparrow}$*, but its public endpoint handoff is limit-facing only*

after the displayed modulus, window, readback, tolerance, and terminal real-seal rows have been consumed.

Proof. Begin with Definition 3632.60. It separates the derivative branch, the Riemann finite-sum branch, and the shared regular-rational and real-name readback before any calculus endpoint is public.

Use Definition 3632.61 to combine the derivative and integral branches. The factorization leaves the endpoint read behind the same $\mathsf{Derivative}^{\uparrow}$–$\mathsf{RiemannIntegral}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ support, not behind a host function space or selected antiderivative.

Finally apply Definition 5722.5. Its finite route M, W, R, D, E is the Real-facing limit seal. Therefore a calculus consumer that turns the endpoint into a limit-facing read must expose that seal before the endpoint can be consumed as public limit data. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealDependency`

Theorem 3632.63 (Calculus finite-limit seal public handoff)**.** *Any public* $\mathsf{Calculus}^{\uparrow}$ *endpoint equality or estimate that combines derivative and Riemann-integral branches may be exported only after the* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *route*

$$M, W, R, D, E$$

of Definition 5722.5, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal have all been exposed. The exported handoff is therefore the displayed finite route through* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FiniteLimitSeal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *and not a host function space, selected antiderivative, host integration functional, quotient function equality, or ambient completeness assertion.*

Proof. Start from the dependency route of Definition 3632.60. It already places derivative reads, Riemann finite-sum reads, regular-rational readback, and real-name sealing on displayed BEDC rows.

Apply Definition 3632.61. Any combined endpoint equality or estimate must first factor through the derivative branch, the Riemann-integral branch, and the shared $\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ readback before it is public.

Now use Definition 3632.62. When that endpoint read becomes limit-facing, the finite route M, W, R, D, E of the $\mathsf{FiniteLimitSeal}^{\uparrow}$ seal is exposed before export.

The non-escape boundary is unchanged: the handoff supplies no host function space, selected antiderivative, host integration functional, quotient function equality, or ambient completeness principle. Hence later calculus statements can consume only the public finite-limit-seal handoff displayed above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealPublicHandoff`

The $\mathsf{Calculus}^{\uparrow}$ plane-boundary branch reads vector-calculus integration through the Green theorem certificate before treating the result as a calculus-facing Riemann-integral dependency. This keeps the planar region, boundary curve, differential-form row, scalar curl or divergence row, and tagged integration handoff visible as a finite route.

Thus any $\mathsf{Calculus}^{\uparrow}$ consumer of plane boundary integration passes through chapter 7971 and chapter 5720 before reading the $\mathsf{RiemannIntegral}^{\uparrow}$ packet of chapter 1526.

Lemma 3632.64 (Calculus finite-limit seal route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *endpoint consumer that turns derivative, integral, or root limit data into a limit-facing read must pass through chapter 5722 and the finite* M, W, R, D, E *route of Definition 5722.5 before the public* $\mathsf{Real}^{\uparrow}$ *endpoint is exposed.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealDependencyRoute`

Proof. Start with the derivative and integral dependency route of Definition 3632.60. It already forces calculus endpoint reads through the displayed $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Now read the finite-limit seal route. Theorem Definition 5722.5 supplies the M, W, R, D, E handoff for real-facing limit reads, so the calculus endpoint consumer must expose that seal before treating the endpoint as public limit-facing data. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusFiniteLimitSealRoute`

The $\mathsf{Calculus}^{\uparrow}$ packet supplies the local difference-quotient and derivative-window route consumed by chapter 5135, chapter 6674, and chapter 6763. These sibling certificates read the same finite calculus window before exporting finite-difference tables, Taylor-polynomial coefficient rows, or mean-value comparison data. The $\mathsf{Calculus}^{\uparrow}$ derivative-facing route reads finite computations through chapter 5135 before using the derivative carrier of chapter 122. Its Real-facing rows remain tied to the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Consequently the finite-difference derivative route is a downstream consumer of the L10 real-completion spine before any derivative endpoint is read: the dyadic tolerance rows of chapter 291 feed the regular rational sequence rows of chapter 301, and those rows feed the real-completion seal of chapter 38. The derivative carrier may inspect finite differences only after that spine has supplied the tolerance-visible $\mathsf{Real}^{\uparrow}$ endpoint. The $\mathsf{Calculus}^{\uparrow}$ packet supplies derivative and limit local-linearization rows to the located-compact boundary only after the finite-net and real-readback routes are visible. A $\mathsf{LocatedCompact}^{\uparrow}$ consumer of a calculus window must first read the local-linearization carrier of Definition 3632.1 and the Cauchy-facing derivative handoff of Definition 3632.51; it then receives the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, the $\mathsf{ContinuousMap}^{\uparrow}$ graph replay row, and the terminal $\mathsf{Real}^{\uparrow}$ seal before any compactness-facing finite-net row may be consumed.

Theorem 3632.65 (Calculus located-compact dependency route)**.** *Every* $\mathsf{LocatedCompact}^{\uparrow}$ *consumer of a* $\mathsf{Calculus}^{\uparrow}$ *derivative, limit, or local-linearization row factors through the finite calculus route displayed by Definition 3632.1, Definition 3632.51,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it reaches the located-compact packet of chapter 1488. The handoff exports a finite graph-and-readback window for compactness use, not open-cover compactness, a selected finite subcover, quotient function equality, or ambient Real completeness.*

Proof. Project the calculus local-linearization carrier. Its derivative and limit rows expose graph data through $\mathsf{ContinuousMap}^{\uparrow}$, regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, and endpoint values through the $\mathsf{Real}^{\uparrow}$ boundary.

Use Definition 3632.51 for the Cauchy-facing derivative window. That route displays the derivative row, graph row, readback row, dyadic tolerance ledger, transport, replay, provenance, and local naming before the Real boundary is consumed.

Only after those rows are visible may the consumer enter chapter 1488. The located-compact packet therefore receives finite calculus data suitable for its finite-net and locatedness rows, with no host compactness theorem or selected subcover coordinate added by the calculus route. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLocatedCompactDependencyRoute`

The $\mathsf{Calculus}^{\uparrow}$ packet consumes the Cauchy and real-completion route before derivative, integral, or limit rows can expose a real-valued endpoint. Its root carrier therefore sits downstream of the regular rational readback of chapter 301, the terminal real boundary of chapter 38, and the generic limit packet of chapter 1678.

Theorem 3632.66 (Calculus Cauchy dependency route)**.** *Every* Calculus$^{\uparrow}$ *derivative, integral, or limit read that exposes a real-valued limit factors through the finite Cauchy route displayed by the local-linearization carrier of Definition 3632.1 and the root local-linearization Cauchy handoff of Definition 3632.51. In particular, the read must display the* RegSeqRat$^{\uparrow}$ *readback row and the terminal* Real$^{\uparrow}$ *seal before any calculus operation can export a real-valued endpoint. The route supplies finite Cauchy-window data for the* Limit$^{\uparrow}$ *consumer, not an ambient completeness theorem, a host limit operator, quotient equality of functions, or a selected real representative.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCauchyDependencyRoute`

Proof. Project the calculus local-linearization carrier. Its derivative, integral, and limit rows are interpreted only together with the regular rational readback row and the terminal real seal.

Apply the root local-linearization Cauchy handoff. That theorem keeps the derivative-facing calculus window behind the displayed Cauchy rows, dyadic tolerance ledger, transport, replay, provenance, and local naming before any Real$^{\uparrow}$ endpoint is public.

Finally read the sibling route through chapter 301, chapter 38, and chapter 1678. These chapters supply the finite readback, real boundary, and limit consumer rows consumed by calculus; none supplies an ambient host completeness or quotient-function equality coordinate. □

Theorem 3632.67 (Calculus Cauchy dependency real-completion readback)**.** *Every accepted* Calculus$^{\uparrow}$ *real-valued derivative, integral, or limit endpoint read exposes the* RegSeqRat$^{\uparrow}$ *readback row, the terminal* Real$^{\uparrow}$ *seal, and the* Limit$^{\uparrow}$ *consumer row in that order. The route exports only finite Cauchy-window data from the local-linearization carrier and never exports an ambient Real-completion theorem, a selected limit, quotient equality of functions, or a host endpoint object.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusCauchyDependencyRealCompletionReadback`

Proof. Project the local-linearization carrier of Definition 3632.1. Its derivative, integral, and limit endpoints are visible only through the displayed Cauchy-window rows, so no endpoint can bypass the finite route.

Apply Definition 3632.66. It already places every real-valued calculus read behind the regular rational readback and the terminal Real seal before any endpoint is public.

Now read the sibling routes in order. The RegSeqRat$^{\uparrow}$ chapter supplies the regular rational readback row, the Real$^{\uparrow}$ chapter supplies the terminal seal, and the Limit$^{\uparrow}$ chapter consumes the resulting finite Cauchy-window data as a limit-facing row. Since those are the only exposed rows, the readback cannot import host completeness, selected representatives, quotient-function equality, or an ambient endpoint. □

The Calculus$^{\uparrow}$ packet consumes the shared finite limit route whenever a derivative or integral read exposes a limiting value. This route is the same public route recorded by chapter 1678, not a host limit operator or a selected real representative.

Theorem 3632.68 (Calculus Limit dependency route)**.** *Every* Calculus$^{\uparrow}$ *finite derivative or integral read that consumes a limiting value must pass through the shared* Limit$^{\uparrow}$ *route of chapter 1678, together with the* RegSeqRat$^{\uparrow}$ *readback rows of chapter 301, the* StreamName$^{\uparrow}$ *schedule rows of chapter 126, the* DyadicRatCore$^{\uparrow}$ *tolerance ledgers of chapter 291, and the downstream* Real$^{\uparrow}$ *seal rows of chapter 38. The calculus carrier cannot replace that route by host limit operators, quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitDependencyRoute`

Proof. Project a calculus read that asks for a derivative or integral endpoint. The endpoint is not read as a host real value; it is routed through the finite limit consumer surface.

Apply the public $\mathsf{Limit}^{\uparrow}$ handoff of Definition 1678.3, Definition 1678.4, and Definition 1678.8. These theorems force the same StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, transport, replay, provenance, and local naming route before a limiting value can be consumed.

The surrounding calculus rows can then read derivative and integral data through that shared route. Since the route contains no host limit operator, quotient stream equality, selected representative, host real equality, countable-choice selector, or ambient-completeness coordinate, no calculus consumer can export one by bypassing $\mathsf{Limit}^{\uparrow}$. □

Theorem 3632.69 (Calculus Limit dependency refusal boundary)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative or integral endpoint read that omits one of the* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *seal rows is outside the BEDC calculus limit-consumer surface. Such a read cannot be retyped as an accepted calculus limit consumer by replacing the missing row with a host limit operator, quotient stream equality, a selected representative, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitDependencyRefusalBoundary`

Proof. Apply Definition 3632.68. It requires every finite derivative or integral endpoint read that consumes a limiting value to pass through the public $\mathsf{Limit}^{\uparrow}$ route and its RegSeqRat, StreamName, DyadicRatCore, and Real handoffs.

If one of those seal rows is absent, the route supplied by Definition 1678.3, Definition 1678.4, and Definition 1678.8 is not available. The attempted endpoint read therefore lacks the displayed finite limit source that the calculus carrier consumes.

The omitted row cannot be supplied by the host side. The cited handoffs exclude host limit operators, quotient stream equality, selected representatives, countable-choice selectors, and ambient completeness rows from the accepted $\mathsf{Limit}^{\uparrow}$ route, so the calculus consumer has no alternate BEDC row through which to retype the missing dependency. □

The $\mathsf{Calculus}^{\uparrow}$ limit route is read only after the real-completion display has fixed its readback rows. The child route below records the sibling order used by derivative, integral, and endpoint consumers: $\mathsf{Limit}^{\uparrow}$ supplies the limit-facing consumer surface, while $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ supply the terminal seal, regular rational readback, and schedule rows that make the limit read finite.

Theorem 3632.70 (Calculus Limit real handoff)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or endpoint consumer that cites the* $\mathsf{Limit}^{\uparrow}$ *route must first display the* $\mathsf{Real}^{\uparrow}$ *terminal seal, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{StreamName}^{\uparrow}$ *schedule row used by that limit route. The handoff is the finite dependency lattice through chapter 1678, chapter 38, chapter 301, and chapter 126; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealHandoff`

Proof. Start with the existing calculus limit dependency route of Definition 3632.68. It already places every derivative or integral endpoint read behind the shared $\mathsf{Limit}^{\uparrow}$ consumer surface.

Now read the sibling rows that make this consumer surface finite. The $\mathsf{StreamName}^{\uparrow}$ schedule names the displayed stream positions, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal consumed by the limit-facing row.

The calculus packet can therefore cite $\mathsf{Limit}^{\uparrow}$ only after those readback rows are visible. Since the route contains no coordinate for a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the handoff remains a finite BEDC dependency route. □

This terminal-export theorem is the canonical public handoff consumed by the $\mathsf{Calculus}^{\uparrow}$ closure ledger, so the ledger points to this route without changing the independent verification grade recorded for the chapter.

Theorem 3632.71 (Calculus Limit real derivative readback)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative endpoint consumer that exports a Real-facing derivative value must pass through* $\mathsf{Limit}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *before the endpoint may be read as a* $\mathsf{Real}^{\uparrow}$ *value. The derivative readback is the finite BEDC route through limit consumption, stream scheduling, regular rational readback, dyadic rational coordinates, and terminal Real sealing; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealDerivativeReadback`

Proof. Start with the real handoff of Definition 3632.70. It already places derivative endpoint reads behind the $\mathsf{Limit}^{\uparrow}$ consumer surface together with $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Now include the existing calculus limit dependency route of Definition 3632.68. Its endpoint schedule is read through rational coordinates, so the $\mathsf{DyadicRatCore}^{\uparrow}$ row must be visible before the regular readback can be sealed as a Real-facing derivative value.

The route therefore exposes only the displayed limit, stream, dyadic rational, regular readback, and Real seal rows. Since no coordinate supplies a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the derivative endpoint remains a finite BEDC readback. □

Theorem 3632.72 (Calculus Limit real terminal export)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or endpoint theorem that exports a Real-facing value through the* $\mathsf{Limit}^{\uparrow}$ *route must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any downstream calculus consumer may read that value. The export composes the real handoff of Definition 3632.70, the derivative readback of Definition 3632.71, and the limit dependency route of Definition 3632.68; it is not a host limit, quotient stream equality, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLimitRealTerminalExport`

Proof. Begin with Definition 3632.70. It places every derivative, integral, or endpoint consumer behind the $\mathsf{Limit}^{\uparrow}$ surface together with the StreamName schedule, RegSeqRat readback, and terminal Real seal.

Apply Definition 3632.71 to the Real-facing endpoint case. That theorem inserts the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row before the regular readback is sealed as a Real-facing value.

Finally use Definition 3632.68 to identify the common calculus limit route consumed by derivative, integral, and endpoint theorems. The same displayed limit surface is the only route by which those consumers can reach a Real-facing value.

The terminal export is therefore the finite row order $\mathsf{Limit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. Because none of the composed routes contains a host limit operator, quotient stream equality, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, no downstream calculus consumer can bypass the displayed rows. □

The $\mathsf{Calculus}^{\uparrow}$ product-distance route is read through the Cauchy product metric packet before derivative-product or endpoint consumers expose any paired Real data. This child route records that the calculus hub uses the finite product-metric surface of chapter 3551 rather than a host product metric.

Theorem 3632.73 (Calculus product metric forward link)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative-product, endpoint, or paired-distance consumer that uses product-distance data must first display the* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet through* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before reading either component as a* $\mathsf{Real}^{\uparrow}$ *value. The route supplies the finite product-distance ledger, paired stream windows, regular rational readbacks, dyadic tolerance coordinates, metric distance row, and componentwise terminal Real seals. It supplies no host product metric, quotient stream equality, selected product limit, detached Real carrier, detached Real pair, or ambient Real completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardLink`

Proof. Begin with the real handoff of Definition 3632.70. It already forces endpoint reads through $\mathsf{Limit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a calculus consumer may expose a Real-facing value.

Now insert the product-distance sibling. The $\mathsf{CauchyProductMetric}^{\uparrow}$ carrier of chapter 3551 reads two $\mathsf{CauchyProduct}^{\uparrow}$ rows, the $\mathsf{Met}^{\uparrow}$ distance row, paired stream windows, regular rational readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The terminal Real seal is componentwise. A paired-distance read may expose the left component and the right component only after the product-metric sibling has displayed the two carried streams, their dyadic comparison ledger, and the Real-facing seals for those same coordinates. It cannot replace that route by a detached host product metric or a selected product limit whose components are later read as independent Real values.

Therefore a derivative-product, endpoint, or paired-distance consumer can use product-distance data only after that finite route is visible. Since no row in the calculus packet or the product-metric sibling carries a host product metric, quotient stream equality, selected product limit, detached Real carrier, detached Real pair, or ambient completeness theorem, the forward link remains a BEDC dependency route. □

Theorem 3632.74 (Calculus product metric seal pairing)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative-product, endpoint, or paired-distance consumer that exposes two component* $\mathsf{Real}^{\uparrow}$ *seals must read both seals through the same* $\mathsf{CauchyProductMetric}^{\uparrow}$ *packet and the shared route*

$$\mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The paired seals are therefore componentwise but not detached: the consumer may inspect the left and right $\mathsf{Real}^{\uparrow}$ *rows only as terminal rows of one displayed product-metric packet. It has no route through detached Real pairs, host product metrics, selected product limits, quotient stream equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricSealPairing`

Proof. Start with the forward link of Definition 3632.73. It places every derivative-product, endpoint, or paired-distance consumer behind the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet before either component Real value is read.

Next apply the finite modulus route of Definition 3632.75. This identifies the common route through the two $\mathsf{CauchyProduct}^{\uparrow}$ source rows, the $\mathsf{Met}^{\uparrow}$ distance row, paired $\mathsf{StreamName}^{\uparrow}$ windows,

paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Finally use the handoff theorem Definition 3632.76. The handoff keeps both component seals downstream of the same product-metric packet, so a calculus consumer cannot split the pair into detached Real data and later attach a host product metric, selected product limit, quotient stream equality, or ambient completeness witness. □

Theorem 3632.75 (Calculus product metric finite modulus route)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *paired-distance read that reaches component* $\mathsf{Real}^{\uparrow}$ *seals must expose the finite product-metric modulus rows before those component seals are read. Concretely, the read factors through the* $\mathsf{CauchyProductMetric}^{\uparrow}$ *route of chapter 3551, the two* $\mathsf{CauchyProduct}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *windows, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-distance ledger, the* $\mathsf{Met}^{\uparrow}$ *distance row, and only then the terminal component* $\mathsf{Real}^{\uparrow}$ *seals. It does not factor through a host product metric, quotient stream equality, ambient completeness, or a detached product Real.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardModulusRoute`

Proof. Start with the forward link of Definition 3632.73. It already makes the product-distance sibling visible before any calculus derivative-product, endpoint, or paired-distance consumer can inspect component Real data.

Next read the product-metric sibling through Definition 3551.4. That boundary orders the route by the two product sources, paired stream windows, regular rational readbacks, dyadic product-distance ledger, metric row, and terminal Real seal. Thus the finite modulus rows are not optional decoration: they are the rows through which the paired distance is displayed before the component seals are available.

The excluded coordinates are the same on both sides of the handoff. Neither the calculus route nor the $\mathsf{CauchyProductMetric}^{\uparrow}$ route contains a host product metric, quotient stream equality, selected product limit, detached product Real, or ambient completeness witness. Therefore every accepted paired-distance read keeps the finite modulus route visible before it reads the component $\mathsf{Real}^{\uparrow}$ seals. □

Theorem 3632.76 (Calculus product metric modulus handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer that reads a derivative-product, endpoint, or paired-distance conclusion must read the product-metric modulus handoff through*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

before either component $\mathsf{Real}^{\uparrow}$ *seal is exposed. The handoff supplies only the displayed product-distance packet, paired source rows, metric row, finite stream windows, regular rational readbacks, dyadic tolerance ledger, and terminal component seals. It supplies no host product metric, quotient stream equality, detached Real pair, selected product limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricModulusHandoff`

Proof. Start with Definition 3632.73. It places every derivative-product, endpoint, or paired-distance consumer after the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet and before any component Real-facing value can be inspected.

Refine the route by Definition 3632.75. That theorem identifies the finite modulus rows: the $\mathsf{CauchyProduct}^{\uparrow}$ sources, $\mathsf{Met}^{\uparrow}$ distance row, paired $\mathsf{StreamName}^{\uparrow}$ windows, paired $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ product-distance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Finally use the $\mathsf{CauchyProductMetric}^{\uparrow}$ real-seal boundary of Definition 3551.4. The boundary keeps the component seals downstream of the product-metric route, so a calculus consumer cannot detach the two Real rows and later pretend that their product distance came from a host product metric.

The excluded principles have no row in the composed display. Quotient stream equality, detached Real pairs, selected product limits, host product metrics, and ambient $\mathsf{Real}^{\uparrow}$ completeness are not generated by the calculus packet or by the product-metric sibling, so the modulus handoff is the only accepted route to the paired-distance read. □

Theorem 3632.77 (Calculus product metric regulated-integral handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *paired-distance or derivative-product consumer that feeds a* $\mathsf{RegulatedIntegral}^{\uparrow}$ *endpoint must first expose the shared route*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and then the $\mathsf{RegulatedIntegral}^{\uparrow}$ *finite ledger*

$$I,\ G,\ A,\ S,\ D,\ R,\ E,\ H,\ C,\ P,\ N$$

before any integral endpoint is public. The route supplies product-metric Real endpoints and the regulated finite-sum handoff through the same L10 Real-completion path. It supplies no host product metric, selected product limit, measure completion, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRegulatedIntegralHandoff`

Proof. Start with Definition 3632.76. It forces any derivative-product, endpoint, or paired-distance conclusion to expose the $\mathsf{CauchyProductMetric}^{\uparrow}$ packet, the $\mathsf{CauchyProduct}^{\uparrow}$ source rows, the $\mathsf{Met}^{\uparrow}$ distance row, the finite $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seals.

Next feed the terminal Real-facing data into the regulated-integral route of Definition 1557.7. That route keeps the integral endpoint behind the same L10 Real-completion interface rather than behind a host integral or a detached completed Real object.

Finally apply the public interface theorem Definition 1557.10. The public regulated-integral endpoint is readable only through the finite ledger $I, G, A, S, D, R, E, H, C, P, N$, so the calculus product-metric route and the regulated-integral finite-sum handoff meet at the displayed Real-completion boundary.

The excluded objects have no coordinate in the composed route. Host product metrics, selected product limits, measure completion, quotient function equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are therefore not available to the consumer before the regulated endpoint is exposed. □

Theorem 3632.78 (Calculus product metric regulated-endpoint route)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *consumer exporting regulated integral endpoints from paired-distance or derivative-product data must factor through* $\mathsf{CauchyProductMetric}^{\uparrow}$*,* $\mathsf{CauchyProduct}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and then the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *endpoint ledger. The route first exposes the shared product-metric rows*

$$\mathsf{CauchyProductMetric}^{\uparrow},\quad \mathsf{CauchyProduct}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

and then reads the regulated endpoint ledger

$$I,\ G,\ A,\ S,\ D,\ R,\ E,\ H,\ C,\ P,\ N.$$

It supplies no host product metric, measure completion, quotient function equality, selected product limit, detached endpoint pair, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Begin with the product metric handoff of Definition 3632.76. It places every derivative-product, endpoint, or paired-distance conclusion behind the CauchyProductMetric$^{\uparrow}$ packet, the CauchyProduct$^{\uparrow}$ source rows, the Met$^{\uparrow}$ distance row, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readbacks, the DyadicRatCore$^{\uparrow}$ tolerance ledger, and the terminal Real$^{\uparrow}$ seals.

Refine the route by the finite modulus theorem Definition 3632.75. That theorem keeps the paired-distance data attached to the product-metric packet before either component Real seal can be read, so the endpoint source is not a detached product limit.

Now apply the regulated-integral handoff of Definition 3632.77. It feeds the terminal product-metric Real data into the regulated endpoint ledger $I, G, A, S, D, R, E, H, C, P, N$ rather than into a host integral or a measure-completion object.

Finally use the public interface theorem Definition 1557.10. The public endpoint is readable only through the displayed regulated-integral rows, and the preceding product-metric rows supply its Real-facing inputs. Since neither route contains a host product metric, measure completion, quotient function equality, selected product limit, detached endpoint pair, or ambient completeness principle, every exported endpoint factors through the finite route stated above. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRegulatedEndpointRoute`

Theorem 3632.79 (Calculus product metric Real route)**.** *Any* Calculus$^{\uparrow}$ *paired-distance consumer that exports a* Real$^{\uparrow}$*-facing derivative or endpoint read factors through the same finite route: the* CauchyProductMetric$^{\uparrow}$ *packet, two* CauchyProduct$^{\uparrow}$ *rows, paired* StreamName$^{\uparrow}$ *windows, paired* RegSeqRat$^{\uparrow}$ *readbacks, the* DyadicRatCore$^{\uparrow}$ *product-distance ledger, the* Met$^{\uparrow}$ *distance row, the modulus handoff row, and then the terminal* Real$^{\uparrow}$ *seals. The route exports no host product metric, quotient stream equality, detached Real pair, selected product limit, or ambient* Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricRealRoute`

Proof. Start with the forward link of Definition 3632.73. It forces every derivative-product, endpoint, or paired-distance consumer that uses product-distance data to display the CauchyProductMetric$^{\uparrow}$ packet before reading component Real values.

Use the seal pairing theorem of Definition 3632.74. It keeps the two terminal Real$^{\uparrow}$ seals componentwise but attached to one shared product-metric packet rather than to a detached Real pair.

Refine the shared packet by Definition 3632.75 and Definition 3632.76. These rows expose the two CauchyProduct$^{\uparrow}$ sources, paired StreamName$^{\uparrow}$ windows, paired RegSeqRat$^{\uparrow}$ readbacks, the DyadicRatCore$^{\uparrow}$ product-distance ledger, the Met$^{\uparrow}$ distance row, and the handoff before the terminal seals are read.

Since each cited route excludes host product metrics, quotient stream equality, selected product limits, detached Real pairs, and ambient completeness, their composition gives exactly the finite Real-facing route stated above. □

Theorem 3632.80 (Calculus product metric forward-link export)**.** *Any public* Calculus$^{\uparrow}$ *paired-distance or derivative-product export factors through the product-metric route before it may feed regulated-integral or endpoint consumers. The export must display the existing* CauchyProductMetric$^{\uparrow}$ *route,* CauchyProduct$^{\uparrow}$ *rows, the* Met$^{\uparrow}$ *distance row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readbacks, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and terminal* Real$^{\uparrow}$ *seals as one finite Real-completion path.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusProductMetricForwardLinkExport`

Proof. Start with Definition 3632.73. It places every paired-distance or derivative-product consumer behind the product-metric route before component Real-facing data can be read.

Use Definition 3632.74, Definition 3632.75, and Definition 3632.76. These rows keep the paired seals attached to one $\mathsf{CauchyProductMetric}^{\uparrow}$ packet, expose the product rows, metric row, stream windows, regular readbacks, and dyadic tolerance ledger, and prevent a detached Real pair or host product metric from entering the export.

Finally apply Definition 3632.79. The Real route is already terminal for the component seals, so the public export can feed regulated-integral or endpoint consumers only through that same displayed Real-completion path. □

Theorem 3632.81 (Calculus Peano dependency route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *read that is used as an ODE-facing derivative row or local-flow row must consume the Peano and initial-value surfaces after the Real and continuous-map rows have already been displayed. Concretely, such a read is routed through the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and modulus rows, chapter 3530, and chapter 3979 before it may be exported as an ODE existence consumer.*

Lean4 ✓ `BEDC.Derived.CalculusUp.PeanoDependencyRoute.CalculusPeanoDependencyRoute`

Proof. The hub spine fixes $\mathsf{Calculus}^{\uparrow}$ as a packet over Real-facing limits, local operations, and continuous-map readback. The existing Real handoff and finite-difference routes make any derivative read visible only after the terminal $\mathsf{Real}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ rows have been named.

An ODE-facing consumer needs more than that derivative row. It must also expose the local existence packet of chapter 3530 and the initial-data packet of chapter 3979, since those chapters bind the interval source, vector-field row, local window, and solution-read surface used by the existence request.

Thus the calculus derivative or local-flow read is not a detached host derivative. It is an additive child route from the already displayed Real and continuous-map rows through Peano existence and initial-value data before any ODE existence consumer is public. □

The $\mathsf{Calculus}^{\uparrow}$ real-completion route is read as a child of the existing limit handoff. The endpoint surface of Definition 3632.70 and Definition 3632.72 already fixes the finite readback order through $\mathsf{Limit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any calculus consumer can export a Real-facing value.

Theorem 3632.82 (Calculus Real-completion forward link)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *Real-facing endpoint export that consumes a* $\mathsf{Limit}^{\uparrow}$ *row must pass through the* $\mathsf{Real}^{\uparrow}$ *terminal seal, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row displayed by Definition 3632.70 and Definition 3632.72. The forward link is the finite L10 route from calculus limits into the Real-completion spine of chapter 38; it is not a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionForwardLink`

Proof. Read the endpoint export through Definition 3632.70. That theorem already places every derivative, integral, or endpoint consumer behind the $\mathsf{Limit}^{\uparrow}$ surface together with the $\mathsf{Real}^{\uparrow}$ terminal seal, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{StreamName}^{\uparrow}$ schedule rows.

Now apply Definition 3632.72. It inserts the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row before the regular readback is sealed as a Real-facing value, so the downstream calculus export cannot skip the displayed real-completion readback order.

Since the route has no coordinate for a host limit operator, quotient stream equality, selected representative, or ambient completeness theorem, the child link records only the finite L10 dependency lattice already visible in the sibling handoff theorems. □

Theorem 3632.83 (Calculus real-completion sibling route)**.** *Every accepted* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or limit consumer that reaches Real-facing completion data must route through the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *completion stack, with the* $\mathsf{StreamName}^{\uparrow}$ *naming surface available when the consumer reads a displayed sequence. The route is witnessed by chapter 38, chapter 301, and chapter 126. It is not an appeal to ambient Real completeness, host quotient completion, or selected limits outside the accepted completion stack.*

Proof. Read the accepted $\mathsf{Calculus}^{\uparrow}$ consumer through its derivative, integral, or limit interface. A Real-facing completion request first has to expose the $\mathsf{Real}^{\uparrow}$ certificate surface rather than a host completeness object.

The completion data then factors through the regular rational naming route supplied by $\mathsf{RegSeqRat}^{\uparrow}$. When the consumer carries a sequential name, the visible stream component is the $\mathsf{StreamName}^{\uparrow}$ naming surface, so the read remains inside the displayed completion stack.

Thus the horizontal dependency is exactly the sibling route through Real, regular rational sequences, and StreamName names. Since no carrier row in the local route names ambient Real completeness, host quotient completion, or selected limits, those endpoints are not exported by the calculus consumer. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionSiblingRoute`

Theorem 3632.84 (Calculus regulated-integral forward link)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *integration-facing consumer that reads a regulated integral first exposes the regulated-integral finite step route of chapter 1557 before it may use* $\mathsf{Real}^{\uparrow}$ *endpoint seals. In the root package of Definition 3632.5, the integral branch reads the calculus finite-sum row, then the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *interval, regulated-function, step approximation, finite step-sum, Darboux/Riemann compatibility, error, transport, replay, provenance, and local NameCert rows, and only then consumes the Real-facing endpoint.*

The forward link is therefore a sibling dependency from the accepted $\mathsf{Calculus}^{\uparrow}$ *root package to the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *route. It does not supply a host integral functional, selected antiderivative, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRegulatedIntegralForwardLink`

Proof. Start from the $\mathsf{Calculus}^{\uparrow}$ root public package of Definition 3632.5. Its integration-facing branch contains the finite-sum row and the structural rows that transport, replay, source, and locally name the accepted calculus read.

Route that branch through chapter 1557. The regulated-integral packet exposes its interval, regulated-function, dyadic step approximation, finite step-sum, Darboux/Riemann compatibility, error, transport, replay, provenance, and local NameCert rows before a Real-facing handoff is public.

The terminal endpoint seal is then read through $\mathsf{Real}^{\uparrow}$ and the existing calculus Real-completion sibling route. Since neither the calculus root package nor the regulated-integral sibling carries a coordinate for host integration, selected antiderivatives, quotient function equality, measure completion, or ambient completeness, the forward link remains the displayed finite route. □

Theorem 3632.85 (Calculus real and continuous-map dependency route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or limit-facing consumer that reads a scalar endpoint or function-value row must*

expose the Real$^{\uparrow}$ *seal of chapter 38 and the* ContinuousMap$^{\uparrow}$ *modulus-and-evaluation surface of chapter 130 before the calculus packet can replay the read through its local* NameCert *row. Thus the calculus hub records a substantive L10-to-analysis edge through Real names and continuous-map data rather than a detached kernel-only route.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealContinuousDependencyRoute`

Theorem 3632.86 (Calculus Real-completion orphan lattice route)**.** *Every* Calculus$^{\uparrow}$ *regulated-integral or limit-facing consumer that reads an endpoint seal must first pass through the L10 Real-completion source lattice of chapter 38, chapter 301, and chapter 126. In particular, the consumer may use the* RegulatedIntegral$^{\uparrow}$ *handoff of Definition 3632.84 only after the source stream window, regular rational readback, and terminal Real seal are visible as the displayed completion route. It exports no host completion theorem, selected representative of a Cauchy stream, quotient real equality, detached endpoint seal, or ambient integration functional.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionOrphanLatticeRoute`

Proof. Start with the Calculus$^{\uparrow}$ root package. Its regulated-integral and limit-facing rows are accepted only as finite BEDC reads through the calculus carrier, transport, replay, provenance, and local naming rows.

Use the regulated-integral forward link. It routes endpoint reads through the displayed RegulatedIntegral$^{\uparrow}$ ledger rather than through a host integral or detached endpoint object.

Finally compose this handoff with the Real-completion source lattice. The StreamName$^{\uparrow}$ window supplies the finite source, RegSeqRat$^{\uparrow}$ supplies regular rational readback, and Real$^{\uparrow}$ supplies the terminal seal, so every such endpoint read remains inside the L10 completion lattice. □

Theorem 3632.87 (Calculus real-completion CauchySpace dependency link)**.** *Every accepted* Calculus$^{\uparrow}$ *derivative, integral, or limit consumer that reaches Real-facing completion data must pass through the* CauchySpace$^{\uparrow}$ *filter-uniform and regular-sequence boundary before the terminal* Real$^{\uparrow}$ *seal is read. Concretely, the existing route through Definition 3632.83 factors through the Cauchy-space consumer surface of chapter 1847: the* CauchyFilter$^{\uparrow}$ *and* UniformSpace$^{\uparrow}$ *rows expose the completion request, the* RegSeqRat$^{\uparrow}$ *row supplies the regular rational readback, the* StreamName$^{\uparrow}$ *row supplies any displayed sequential name, and only then may the* Real$^{\uparrow}$ *seal be consumed.*

Proof. Start with a Real-facing Calculus$^{\uparrow}$ consumer and read it through Definition 3632.83. That route already forces the endpoint through the Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and StreamName$^{\uparrow}$ completion stack rather than through ambient completeness.

Now insert the Cauchy-space boundary supplied by chapter 1847. Its Real-consumer rows require the displayed filter, uniform, regular-name, precompletion, transport, replay, provenance, and local naming data before a Real-facing consumer can inspect completion data.

Thus the calculus endpoint cannot jump directly from a local derivative, integral, or limit row to a Real seal. It first traverses the Cauchy-space filter-uniform and regular-sequence interface, and the route exports no host quotient completion, selected limit, countable-choice selector, or ambient Real$^{\uparrow}$ completeness theorem. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusRealCompletionDependencyLink`

Theorem 3632.88 (Calculus pointwise stream metric dependency route)**.** *Every accepted* Calculus$^{\uparrow}$ *derivative, integral, or local-error consumer that compares real-valued stream data must first read the pointwise metric-stream packet of chapter 4139. The comparison route consumes the displayed*

StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* RegSeqRat$^{\uparrow}$ *readback, terminal* Real$^{\uparrow}$ *seal, and metric row before derivative or integral estimates may treat the local error as a real-valued bound. Interval-local derivative or integral consumers also read the compact interval endpoint and finite-net route of chapter 5553 before treating a local error bound as a calculus estimate on a compact real interval.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusPointwiseStreamMetricDependencyRoute`

Proof. Project the calculus consumer through its derivative, integral, or local limit row. A local error comparison needs a pointwise distance before it can be read as a Real-facing estimate.

Use chapter 4139. Its carrier displays the stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, terminal Real seal, and metric row before any downstream completion or calculus consumer receives the distance packet.

Use chapter 5553 for interval-local estimates. The CompactRealInterval$^{\uparrow}$ packet supplies endpoint Real seals and finite interval-net rows, so the pointwise metric stream comparison is anchored to a displayed compact interval packet rather than to an ambient host interval.

Thus the calculus route reads pointwise stream metric control before derivative and integral consumers compare real-valued local errors. The dependency exports no host function equality, quotient stream equality, selected representative, countable-choice schedule, or ambient Real$^{\uparrow}$ completeness principle. □

Theorem 3632.89 (Calculus L10 Real and Completion dependency)**.** *Every* Calculus$^{\uparrow}$ *certificate that exports derivative, integral, or limit-facing clauses to the Real and Completion route must first read the L10 substrate through chapter 38, chapter 301, chapter 126, and chapter 130. The calculus row therefore consumes terminal real seals, regular rational readbacks, finite stream windows, and displayed continuous-map evidence before derivative or integral clauses may be used as public BEDC obligations.*

Proof. Read the finite observation schedule through the StreamName certificate. This fixes the inspected windows before any regular rational or real-facing calculus row is exported.

Use the RegSeqRat and Real certificates next. They attach regular rational readback and terminal real sealing to the same windows.

Finally read the ContinuousMap certificate for displayed function evidence. With those four sibling certificates present, the derivative, integral, and limit-facing calculus clauses have a visible L10 dependency route rather than a kernel-only carrier island. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusL10RealCompletionDependency`

Theorem 3632.90 (Calculus L10 dependency route)**.** *Every accepted* Calculus$^{\uparrow}$ *consumer reads derivative and integral data only after the displayed* Real$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* Limit$^{\uparrow}$*, and* ContinuousMap$^{\uparrow}$ *style rows have been exposed. The route is therefore horizontal through chapter 38, chapter 301, chapter 126, chapter 1678, and chapter 130; it is not a host differentiability oracle, a selected antiderivative, an ambient completeness principle, or a quotient of function germs.*

Proof. Project an accepted Calculus$^{\uparrow}$ packet through the existing dependency routes in this hub. The Real-facing row supplies the scalar seal, the regular rational and stream-name rows supply finite approximation schedules, the limit row supplies the local convergence boundary, and the continuous-map row supplies the function readback surface.

Derivative and integral reads occur only after those rows are displayed. The packet may replay them through its structural BHist, hsame, Cont, Pkg, and NameCert rows, but those rows preserve the displayed dependencies rather than adding a host-level analytic oracle.

Thus an L10 consumer sees calculus data as a finite dependency route through the sibling chapters named above. No row exports selected antiderivatives, ambient completeness, quotient germs, or an untyped host computation. □

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusL10DependencyRoute`

The $\mathsf{Calculus}^{\uparrow}$ limit and endpoint consumers that reuse a Real-completion diagonal estimate read the shared-threshold packet of chapter 909 before exporting a Real-facing calculus row. That sibling packet exposes the route through the shared modulus, regular-Cauchy tail meet, witness envelope, diagonal-window coverage, and comparison row; calculus therefore receives a displayed finite diagonal-meet boundary rather than a detached host limit, quotient stream equality, or selected completion witness. The $\mathsf{Calculus}^{\uparrow}$ hub consumes its Real-facing dependency route through the existing BEDC name certificates for chapter 38, chapter 301, chapter 126, and chapter 291. Its paired-distance consumers enter the product-metric child route of Definition 3632.73 before exposing any derivative-product, endpoint, or component $\mathsf{Real}^{\uparrow}$ seal, so the hub records a horizontal dependency on those finite route packets rather than a bare list of local children.

The $\mathsf{Calculus}^{\uparrow}$ endpoint and derivative-product reads also pass through the finite Cauchy real completion interface of chapter 1844 before any completion-facing consumer may inspect a terminal $\mathsf{Real}^{\uparrow}$ seal.

Lemma 3632.91 (Calculus lower-envelope Real-facing route)**.** *Every* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or variation consumer that reads a lower-envelope regularization as a Real-facing value must route first through chapter 1536, then through chapter 6224, then through the regular rational readback of chapter 301, and finally through the terminal seal of chapter 38. Thus the calculus packet sees the lower-envelope value only after the finite lower-semicontinuity, epigraph, regular-rational, and Real rows are displayed, with no route through host suprema, quotient functions, selected minimizers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLowerEnvelopeRealFacingRoute`

Proof. Read the lower-envelope request as a calculus-facing value. The lower-semicontinuity row supplies the admissible lower-envelope boundary, and the epigraph row supplies the displayed value/lower-bound comparison used by that boundary.

Next pass to regular rational data. The $\mathsf{RegSeqRat}^{\uparrow}$ route records the finite readback through visible windows, so the calculus consumer has a concrete value surface before it can inspect any Real seal.

Finally read the $\mathsf{Real}^{\uparrow}$ seal. Since the route has already passed through the displayed lower-semicontinuity, epigraph, and regular-rational rows, no derivative or integral consumer can replace it by a host supremum, quotient-function equality, selected minimizer, or ambient completeness principle. □

Theorem 3632.92 (Calculus lower-envelope regularization handoff)**.** *Any* $\mathsf{Calculus}^{\uparrow}$ *derivative, integral, or variation consumer that imports lower-envelope regularization must preserve the finite route*

$$\mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{Epigraph}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

before any calculus operation reads the regularized value. The consumer first displays the lower-semicontinuity obligation and epigraph comparison rows, then reads the regular rational approximation together with the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and only then consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no host supremum, quotient function, selected minimizer, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CalculusUp.CalculusLowerEnvelopeRegularizationHandoff`

Proof. Begin with the route lemma above. It already forces a calculus-facing lower-envelope value through $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the value is visible to a derivative, integral, or variation consumer.

Next use the lower-envelope regularization packet of chapter 7792. Its NameCert obligations keep the finite lower-envelope ledger between the epigraph row and the regular-rational readback, while its L10 handoff keeps the regularized value inside the displayed envelope route.

Finally insert the shared dyadic tolerance discipline. The regularization carrier records the finite comparison ledger read by the regular-rational row, and downstream calculus consumers must retain that $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance surface before they read the $\mathsf{Real}^{\uparrow}$ seal.

The resulting handoff is the listed finite route. Since the cited rows contain no coordinate for host suprema, quotient functions, selected minimizers, or ambient completeness, no calculus operation can recover those objects from the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Calculus}^{\uparrow}$ packet is generated from Real, Limit, ContinuousMap, Derivative, Integral, RegSeqRat, componentwise hsame, displayed Cont, Pkg, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$, maturePackageClosed)

Scope closed: The bridge-facing export surface is the finite source-window calculus route over $\mathsf{Real}^{\uparrow}$, $\mathsf{Limit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Calculus}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CalculusUp.CalculusLimitRealTerminalExport`

Bridge status: none

Not claimed: This chapter does not close bridge checking, a host function space, global differentiability, an ambient integration operator, quotient function equality, or real completeness beyond the displayed Real row.

Upgrade path: Further work is standard-bridge verification over the checked limit-real terminal export without adding host function spaces, global differentiability, ambient integration, quotient function equality, or extra real completeness.

Chapter 3633

A Concrete Naming Certificate for $\mathsf{ChainRule}^{\uparrow}$

Following Bishop and Bridges and standard calculus treatments, the $\mathsf{ChainRule}^{\uparrow}$ packet records a finite derivative-composition read as NameCert data inside the L10 calculus boundary. It sits beside the calculus root surface of chapter 3632, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names the displayed composition window through derivative and continuity rows; it is not a host theorem over an ambient function space, quotient equality of maps, a selected representative, countable choice, or an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3633.1 (Chain-rule carrier)**.** *A* $\mathsf{ChainRule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D_0, C_i, D_o, W, Q, E, H, T, P, N)$$

with the following displayed rows:

1. D_0 *is the source* $\mathsf{Derivative}^{\uparrow}$ *row naming the derivative observation for the inner displayed map at the inspected source window.*
2. C_i *is the inner* $\mathsf{ContinuousMap}^{\uparrow}$ *row that carries the graph window needed to feed the outer derivative row.*
3. D_o *is the outer* $\mathsf{Derivative}^{\uparrow}$ *row naming the derivative observation at the displayed image window.*
4. W *is the composition-window row recording the finite source, image, tolerance, and endpoint requests common to* D_0, C_i, D_o*.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational derivative and error values inspected through* W*.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* D_0, C_i, D_o, W, Q *have been displayed.*
7. H *records componentwise* hsame *transport for the derivative, continuous-map, composition-window, readback, real-seal, replay, provenance, and local naming rows.*
8. T *records displayed* Cont *replay from a chain-rule request through* $D_0, C_i, D_o, W, Q, E, H, P, N$*.*
9. P *is the* Pkg *provenance over* $\mathsf{ChainRule}^{\uparrow}$*,* $\mathsf{Calculus}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{ChainRule}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D_0, C_i, D_o, W, Q, E, H, T, P, N$. It has no coordinate for host function equality, quotient equality of maps, a selected representative, countable choice, a global derivative operator, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 3633.2 (Chain-rule NameCert obligations)**.** *For every accepted* $\mathsf{ChainRule}^{\uparrow}$ *carrier* $G = (D_0, C_i, D_o, W, Q, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChainRule}^{\uparrow}}$ *surface consists exactly of carrier admission for the source derivative row, inner continuous-map row, outer derivative row, composition-window row, regular rational readback, real-name seal, componentwise transport, displayed replay, package provenance, and local naming. Every derivative-composition read must pass through the* $\mathsf{Calculus}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *sibling rows before a consumer reads the resulting Real-facing value.*

Proof. Project the carrier of Definition 3633.1. The only analytic rows available to the classifier are D_0, C_i, D_o, W, Q, E: the inner derivative observation, inner graph window, outer derivative observation, shared composition window, regular rational readback, and terminal Real seal.

The calculus root surface of chapter 3632 supplies the L10 boundary in which those rows are inspected. The source and outer derivative rows remain displayed $\mathsf{Derivative}^{\uparrow}$ observations, and the inner graph row remains a $\mathsf{ContinuousMap}^{\uparrow}$ observation; none of them is replaced by a host function-space equality or a global derivative operator.

Readback and sealing are downstream of the same composition window. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q records the finite rational values and tolerances exposed by W, and the $\mathsf{Real}^{\uparrow}$ row E is consumed only after those values have been displayed.

Finally, H, T, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for quotient map equality, selected representatives, countable choice, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3633.3 (Chain-rule L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{ChainRule}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{Calculus}^{\uparrow},\ \mathsf{Derivative}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Thus a derivative-composition read can be cited by later calculus, completion, and real-readback chapters as a named finite packet. The handoff exports no host function equality, quotient function space, selected representative, countable-choice schedule, global derivative operator, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the obligation surface of Definition 3633.2. It fixes the derivative-composition request inside the $\mathsf{Calculus}^{\uparrow}$ boundary and exposes the two derivative rows together with the inner continuous-map row before any Real-facing read is allowed.

The regular rational and real-name siblings then determine the endpoint of the read. By chapter 301, the values visible to the consumer are rational-window readbacks, and by chapter 38, the terminal row is a sealed Real-facing name rather than an ambient completeness principle.

Transport and replay preserve the same finite route through H and T, while P and N record provenance and local naming. Therefore the L10 handoff is exactly the displayed BEDC route, with the excluded host equalities, representatives, choice principles, and ambient completeness objects absent from the packet. □

Falsifiable prediction: Every accepted $\mathsf{ChainRule}^{\uparrow}$ read must expose source derivative data, inner ContinuousMap graph data, outer Derivative data, a finite composition window, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance,

and local naming before any calculus or completion consumer may read the derivative-composition value.

Independence witness: calculus, derivative, continuousmap, regseqrat, real: $\mathsf{ChainRule}^{\uparrow}$ is not bijective to any listed sibling because it binds two derivative observations and one inner continuousmap graph row through one composition-window read before the rational and Real-facing rows are consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{ChainRule}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChainRule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3633.1, Definition 3633.2, and Definition 3633.3 seed the finite derivative-composition carrier and its L10 Real-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChainRule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host analytic chain-rule theorem, quotient equality of functions, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exactness, inner continuousmap exposure, composition-window exactness, regular-readback boundary, Real-seal boundary, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.ChainRuleUp.

Constructive story: A $\mathsf{ChainRule}^{\uparrow}$ packet is generated from a source Derivative row, inner ContinuousMap row, outer Derivative row, finite composition-window row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3634

A Concrete Naming Certificate for $\mathsf{FrechetDerivative}^{\uparrow}$

Following Cartan-style differential calculus on normed spaces, the $\mathsf{FrechetDerivative}^{\uparrow}$ packet records the local linear approximation surface as finite NameCert data. It sits beside the norm carrier of chapter 89, the linear-map obligations of chapter 48, the calculus route named by $\mathsf{Calculus}^{\uparrow}$, and the terminal real-name seal of chapter 38. The packet names a displayed derivative read through norm, linearity, and dyadic remainder control; it is not a host function-space derivative, not a selected tangent map outside the carrier, and not an ambient theorem about differentiability.

Definition 3634.1 (Frechet derivative carrier)**.** *A* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the normed source row, carrying the domain and codomain norm data through which difference vectors and error sizes are read.*
2. *A is the point-and-window row, recording the base point, admissible neighbourhood window, and finite comparison requests visible to the derivative packet.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the linear approximation admitted at the displayed point and over the same source and target norm rows.*
4. *R is the dyadic remainder-modulus row. It records a finite tolerance schedule that bounds the residual after the linear read L has been subtracted from the local step read.*
5. *H records componentwise* hsame *transport for X, A, L, R and the structural rows.*
6. *C records displayed* Cont *replay from a derivative request through the normed source, point window, linear map, and dyadic remainder-modulus rows.*
7. *P is the* Pkg *provenance over* $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host function equality, a global function-space topology, an unrecorded derivative operator, quotient tangent germs, or classical differentiability primitives outside the displayed finite route.

Lean4 ✓ `BEDC.Derived.FrechetDerivativeUp.TasteGate.FrechetDerivativeTasteGate single carrier alignment`

Theorem 3634.2 (Frechet derivative NameCert obligations)**.** *For every accepted* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier* $D = (X, A, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the normed source row, point-and-window row, linear-map row, dyadic remainder-modulus row, componentwise transport, displayed replay, provenance, and local naming; linear approximation of the local step through* L*; remainder-modulus control through* R *over the same window* A *and norm row* X*; classifier stability under* hsame *transport of all displayed rows; and ledger non-escape from host function equality, global function-space topology, unrecorded derivative operators, quotient tangent germs, and classical differentiability primitives outside the packet.*

Proof. Project the carrier of Definition 3634.1. The tuple X, A, L, R, H, C, P, N is the whole accepted packet, so carrier admission for the named rows is a direct read from the finite history.

The linear approximation clause is the read of L over the source row X and the point window A. Because L is a $\mathsf{LinMap}^{\uparrow}$ row in the same packet, the approximation is not detached into a host tangent map.

The remainder-modulus clause is the read of R. Its dyadic tolerance schedule is attached to the same X and A, so the residual bound is compared through the displayed norm and window rows rather than through an external limit predicate.

Classifier stability is the componentwise transport row H, replay is the continuation row C, and source accounting is the package row P. Since no coordinate names any excluded host object, the local certificate exports only the displayed Frechet-derivative route. □

Theorem 3634.3 (Frechet derivative linear-remainder handoff)**.** *Every derivative read of an accepted* $\mathsf{FrechetDerivative}^{\uparrow}$ *carrier factors through the ordered finite route*

$$\mathsf{Norm}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{Calculus}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The linear approximation row and dyadic remainder-modulus row are exposed before any consumer may read the derivative as public BEDC data. Thus the packet supplies a traditional analysis neighbour for later derivative and implicit-function certificates without importing a host derivative operator.

Proof. Read a derivative request through the continuation row C. The replay first exposes the normed source row X and point window A, so the request is already tied to displayed source, target, and local neighbourhood data.

Next read the $\mathsf{LinMap}^{\uparrow}$ row L and the dyadic remainder-modulus row R. The first row supplies the linear approximation, while the second supplies the finite tolerance ledger that keeps the residual under the displayed norm comparison.

The package row P records the dependency route through $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$. The local naming row N exports only that route, with no host function-space derivative, selected tangent germ, quotient equality, or ambient differentiability theorem. □

Falsifiable prediction: Every accepted $\mathsf{FrechetDerivative}^{\uparrow}$ packet must expose $\mathsf{Norm}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before any derivative read is public.

Independence witness: norm, linearmap, calculus, real: $\mathsf{FrechetDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a normed source row, point-and-window row, linear approximation row, dyadic remainder-modulus row, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3634.1, Definition 3634.2, and Definition 3634.3 seed the finite Frechet derivative route over Norm, LinearMap, Calculus, Real, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, global function-space topology, quotient tangent germs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, linear approximation, remainder-modulus control, classifier stability, derivative handoff stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.FrechetDerivative

Constructive story: A $\mathsf{FrechetDerivative}^{\uparrow}$ packet is generated from a normed source row, a point-and-window row, a LinearMap approximation row, a dyadic remainder-modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 3635

A Concrete Naming Certificate for $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Riemann–Darboux treatment of bounded intervals, the $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ packet records a finite subdivision surface for a displayed bounded $\mathsf{Real}^{\uparrow}$ interval. It sits after the dyadic rational core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the located partition surface of chapter 2248. The packet supplies a reusable finite subdivision carrier for $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$ consumers; it does not supply open-cover compactness, selected limits, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3635.1 (Bounded real interval partition carrier)**.** *A* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, S, M, T, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the bounded interval endpoint row, carrying lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seals together with the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks used to observe them.*
2. *S is the ordered subdivision row, listing finitely many subdivision nodes between the displayed endpoints and the adjacent cell ledger they generate.*
3. *M is the dyadic mesh-bound row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording a rational precision budget, finite cell-width bounds, and the endpoint-to-cell comparison rows.*
4. *T is the tag-window row, assigning each cell a finite tag window and recording only the displayed containment evidence for that tag.*
5. *Q is the regular rational readback row for the subdivision nodes, tag windows, mesh bounds, and endpoint comparisons.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and subdivision observations, reached only after E, S, M, T, Q have been displayed.*
7. *L is the integration-consumer ledger, recording that* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* $\mathsf{RiemannStieltjes}^{\uparrow}$*, and* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumers may read the finite partition only through the rows E, S, M, T, Q, R.*

8. *H records componentwise* hsame *transport for endpoints, subdivision nodes, dyadic mesh bounds, tag windows, regular readbacks, real seals, consumer ledgers, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from endpoint observations to ordered cells, from cells to tag windows, and from mesh bounds to the consumer ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed endpoint, subdivision, mesh-bound, tag-window, readback, seal, consumer-ledger, transport, replay, provenance, and local naming rows. It has no coordinate for an open-cover compactness proof, selected limiting partition, quotient real equality, host interval partition, countable choice, or ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.BoundedRealIntervalPartitionUp.TasteGate.BoundedRealIntervalPartitionTasteGate single carrier alignment`

Theorem 3635.2 (Bounded real interval partition NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier* $B = (E, S, M, T, Q, R, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartitio}}$ *surface consists exactly of carrier admission; bounded endpoint exposure through E; ordered subdivision exposure through S; dyadic mesh-bound exposure through M; tag-window containment through T;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through R; integration-consumer routing through L; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Thus a Riemann, Darboux, Riemann–Stieltjes, or regulated-integral consumer receives a finite subdivision surface only after the endpoint, cell, mesh-bound, tag-window, readback, and seal rows have all been displayed.

Proof. Project the carrier of Definition 3635.1. The tuple lists the only available rows: endpoints, ordered subdivision nodes, dyadic mesh bounds, tag windows, regular readbacks, real seals, consumer routing, transport, replay, provenance, and local naming.

The endpoint stage is E. Its $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and $\mathsf{Real}^{\uparrow}$ seals supply bounded endpoint observations before a cell can be inspected.

The finite subdivision stage is S, M, T, Q. The ordered row S supplies the cells, the dyadic row M records the mesh budget and cell-width bounds, the tag row T confines each tag to a displayed window, and the regular readback row Q reads those finite observations.

The consumer stage is R, L, H, C, P, N. The terminal seal R is reached only through the displayed rows, while L records the permitted $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$ reads. The structural rows transport, replay, package, and name exactly this route, so the listed obligations exhaust the local NameCert surface. □

Theorem 3635.3 (Bounded real interval partition mesh consumer readiness)**.** *Let* $B = (E, S, M, T, Q, R, L, H, C, P$ *be an accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carrier. The ordered subdivision row S and dyadic mesh row M are read as a bounded refinement of the located partition surface of chapter 2248 before any* $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer may inspect the integration-consumer ledger L. The consumer therefore receives a located finite partition with bounded dyadic mesh evidence, not an isolated mesh code or host interval partition.*

Proof. Project the carrier of Definition 3635.1. The endpoint row E supplies the bounded Real interval, and the subdivision row S lists ordered cells over that same endpoint surface.

Apply Definition 3635.2. The dyadic mesh-bound row M is not a separate object; it is read after S and compares adjacent cells by the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ budget.

The located partition dependency is therefore substantive. The rows S and M refine the located interval partition of chapter 2248 by adding bounded endpoint access and dyadic mesh bounds before the regular readback Q and real seal R are exposed.

Only then may the ledger L route the packet to Riemann, Darboux, Riemann–Stieltjes, or regulated-integral consumers. Since the carrier contains no host partition coordinate, every such consumer reads the bounded refinement route rather than a free-standing subdivision code. □

Theorem 3635.4 (Bounded real interval partition transport stability)**.** *Let $B = (E, S, M, T, Q, R, L, H, C, P, N)$ and $B' = (E', S', M', T', Q', R', L', H', C', P', N')$ be accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *carriers classified together by componentwise* hsame *transport. If S' is read as the transported ordered subdivision over the transported bounded endpoint row E', then the mesh-bound row, tag-window row, regular readback row, real-seal row, and integration-consumer ledger are all read through the same finite transported subdivision route.*

The transported carrier therefore preserves the finite partition surface without adding open-cover compactness, selected limits, quotient real equality, a host interval partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint row. Componentwise transport carries E to E', including the finite StreamName windows, RegSeqRat readbacks, and Real seals attached to the lower and upper endpoints.

Read the transported subdivision through S'. Since its cells are displayed over E', every later mesh or tag read remains attached to the same bounded interval route.

The rows M', T', Q', R' are downstream of that route. The dyadic mesh bounds compare transported adjacent cells, the tag windows remain cell-local, the regular readbacks observe only the displayed finite windows, and the Real seal is reached after those observations.

Finally L', H', C', P', N' carry the same route to the integration consumers and name it. No additional coordinate exists in the carrier, so the transported packet cannot export any of the excluded compactness, selection, quotient, host-partition, choice, or completeness data. □

Theorem 3635.5 (Bounded real interval partition consumer nonescape)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* $\mathsf{RiemannStieltjes}^{\uparrow}$*, or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BoundedRealIntervalPartition}^{\uparrow}.$$

The route exposes endpoint rows, ordered subdivision rows, dyadic mesh bounds, tag windows, readbacks, terminal seals, componentwise transport, displayed replay, provenance, and local naming. It exports no open-cover compactness theorem, selected limiting partition, quotient real equality, host interval partition, measure-completion object, or ambient real-completeness principle.

Proof. Apply Definition 3635.2. It makes $E, S, M, T, Q, R, L, H, C, P, N$ the entire public surface of the packet.

Apply Definition 3635.4. Any transported or refined consumer read must preserve the same endpoint-to-subdivision-to-mesh route before the consumer ledger L is available.

Now inspect the consumers named in L. A Riemann or Darboux read may use the displayed cells and mesh bounds; a Riemann–Stieltjes read may additionally pair those cells with its own

integrator row; and a regulated-integral read may use the same finite subdivision as a step surface. In each case the consumer receives only the listed finite rows and their structural provenance.

Since the carrier contains no row for compactness, selected limits, quotient equality, host partitions, measure completion, or ambient real completeness, none of those objects can escape through the consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealIntervalPartition}^{\uparrow}$ packet is generated from bounded Real endpoint seals, ordered subdivision rows, dyadic mesh bounds, tag windows, RegSeqRat readbacks, integration-consumer routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3635.1, Definition 3635.2, Definition 3635.4, and Definition 3635.5 seed the bounded real interval partition boundary over $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, and $\mathsf{RegulatedIntegral}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealIntervalPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient real equality, host interval partitions, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, ordered subdivision exactness, dyadic mesh-bound stability, tag-window containment, RegSeqRat readback, Real seal nonescape, integration-consumer routing, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRealIntervalPartitionUp.

Chapter 3636

A Concrete Naming Certificate for $\mathsf{CauchySequenceOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceOrder}^{\uparrow}$ packet records the finite constructive-order route by which two regular rational Cauchy source rows are compared before any $\mathsf{Real}^{\uparrow}$ consumer reads an ordered boundary. It sits over the binary algebra packet of chapter 3708, the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic comparison ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives completion and ordered-Real consumers one finite comparison surface for Cauchy sequence algebra; it does not import trichotomy, quotient Cauchy classes, selected limits, countable choice, host real order, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3636.1 (Cauchy sequence order carrier)**.** *A* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular rational observations supply the left and right comparison inputs.*
2. *A is the upstream* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row that fixes the shared binary Cauchy-algebra source from which the comparison is read.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite pairs of observations where X and Y are inspected at the same requested tolerance scale.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row. It stores the displayed dyadic radius, rational endpoint differences, slack conversions, strict-gap tests, and non-strict comparison checks used by the accepted order request.*
5. *L is the apartness row. It records only displayed positive-gap evidence between the two regular names, with the gap budget read through D and the windows read through W.*
6. *Q is the non-strict order row. It records only displayed tolerance domination evidence, again through the same W and D rows.*
7. *R is the comparison replay row, exposing whether a downstream consumer requested apartness, non-strict order, or the boundary case where no displayed comparison row is accepted.*

8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the requested comparison row has displayed its regular-source and dyadic-budget dependencies.*

9. *H records componentwise* hsame *transport for the two source rows, algebra row, synchronized windows, dyadic budget, apartness row, non-strict order row, replay row, Real seal, continuation route, provenance, and local naming row.*

10. *C records displayed* Cont *replay from a constructive order request through $X, Y, A, W, D, L, Q, R, S, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceOrder}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, A, W, D, L, Q, R, S, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient-stream equality, a selected limit, host real order, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceOrderUp.CauchySequenceOrderTasteGate single carrier alignment`

Theorem 3636.2 (Cauchy sequence order NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier*

$$O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceOrder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, upstream* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budget, apartness row, non-strict order row, comparison replay row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, continuation, provenance, and local naming;*

2. *source-window synchronization, requiring every comparison request to inspect X and Y through the same displayed window row W before any dyadic comparison is read;*

3. *dyadic-budget exactness, requiring every apartness or non-strict order row to consume only the displayed radius, difference, slack, strict-gap, and domination entries of D;*

4. *comparison-row separation, requiring L to carry only positive-gap apartness evidence and Q to carry only non-strict domination evidence, while both rows consume the same source, algebra, window, and dyadic-budget rows;*

5. *Real seal handoff, requiring S to be exposed only after the requested comparison row has displayed its regular-source and dyadic-budget dependencies;*

6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, A, W, D, L, Q, R, S, P, N$ without adding trichotomy, quotient equality, selected limits, host real order, countable choice, or ambient completeness.*

No Cauchy sequence order NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3636.1. The displayed coordinates are exactly the two source rows X, Y, algebra row A, synchronized windows W, dyadic budget D, comparison rows L, Q, R, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the comparison budget is available, so the carrier has no second observation service for either input.

Dyadic-budget exactness and comparison-row separation are read from D, L, Q. The row D supplies the finite arithmetic comparisons, the row L consumes those comparisons for positive-gap apartness, and the row Q consumes them for non-strict domination. Since L and Q are distinct rows with the same upstream inputs, neither row can import a hidden order oracle.

Finally, the replay, seal, and structural rows close the finite route. The row R records which comparison request is being replayed, the seal S is downstream of the displayed comparison dependency, and H, C, P, N only transport, replay, package, and name the listed rows. The excluded trichotomy, quotient, selected-limit, host-order, choice, and ambient-completeness data have no carrier coordinate. □

Theorem 3636.3 (Cauchy sequence order dyadic budget exactness)**.** *Every apartness or non-strict order fact exported from an accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier reads its comparison budget through the displayed route*

$$X, Y, A \longmapsto W \longmapsto D \longmapsto L, Q, R \longmapsto S \longmapsto H, C, P, N.$$

Consequently the exported fact obtains synchronized regular-source observations, dyadic radius and slack data, strict-gap or domination evidence, the terminal Real seal handoff, transport, continuation, provenance, and local naming, and obtains no trichotomy datum, quotient Cauchy class, selected limit, host real order, countable-choice datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Begin with a downstream comparison request. By Definition 3636.1, the only source coordinates are X and Y, and the upstream algebra row is A.

The finite observation surface is then forced through W. The dyadic comparison budget is forced through D, whose rows record the displayed radius, endpoint differences, slack conversions, strict-gap tests, and domination checks.

Apartness reads the route row L, non-strict order reads the route row Q, and replay reads the route row R. Each of these rows is downstream of the same sources, algebra row, windows, and dyadic budget named in Definition 3636.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ consumer can read S only after the requested comparison row has displayed its dependencies, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which trichotomy, quotient equality, selected limits, host order, choice, or ambient completeness could be exported. □

Theorem 3636.4 (Cauchy sequence order non-escape)**.** *Let* $O = (X, Y, A, W, D, L, Q, R, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceOrder}^{\uparrow}$ *carrier. Any order-facing fact exported from* O *is a rowwise consequence of the displayed regular source rows, algebra row, synchronized windows, dyadic comparison budget, apartness row, non-strict order row, replay row, Real seal, transport, continuation, provenance, and local naming rows. It is not a trichotomy principle, quotient Cauchy class, selected limit, host real-order fact, countable-choice datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly $X, Y, A, W, D, L, Q, R, S$, and the structural coordinates are exactly H, C, P, N.

An exported apartness fact must use L after the source, algebra, window, and dyadic-budget rows have been displayed. An exported non-strict order fact must use Q after the same upstream rows have been displayed. A replayed comparison must pass through R, and any Real-facing boundary then passes through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create trichotomy, quotient equality, selected limits, host order, choice, or

ambient completeness, while provenance and local naming record only the finite packet. Thus every exported order-facing fact is confined to the listed rows, and each excluded principle has no entry point. □

Closure status for NameCert$_{CauchySequenceOrderUp}$.

Theory closure: TheoryGate(NameCert$_{CauchySequenceOrderUp}$, seed)

Scope closed: Definition 3636.1, Definition 3636.2, Definition 3636.3, and Definition 3636.4 seed the finite constructive-order route over CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{CauchySequenceOrderUp}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient Cauchy classes, selected limits, host real order, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, comparison-row separation, Real seal handoff, transport, continuation, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceOrderUp.

Constructive story: A CauchySequenceOrder$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, an upstream CauchySequenceAlgebra$^{\uparrow}$ row, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ comparison-budget ledger, apartness and non-strict order rows, a comparison replay row, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3637

A Concrete Naming Certificate for $\mathsf{NoetherianRing}^{\uparrow}$

Following Emmy Noether's ascending-chain formulation, the $\mathsf{NoetherianRing}^{\uparrow}$ packet records finite ideal-chain stabilization data over a commutative-ring source and an ideal source. In BEDC form it sits after $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44) and $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81), supplies a reusable finitary algebra row for the reduction surface of chapter 2197, and exposes the Noetherian component later consumed by chapter 172.

3637.1 Carrier definition

Definition 3637.1 (Noetherian ring carrier)**.** *A* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{NoethRing}(C, I, A, G, H, Q, P, N).$$

The row C is the $\mathsf{CommRing}^{\uparrow}$ *source surface. The row I is the* $\mathsf{Ideal}^{\uparrow}$ *source surface over the same classifier. The row A is the ascending-chain stabilization witness: every displayed chain of ideal rows over C and I carries a finite stage after which all later ideal endpoints classify with that stage. The row G is the finite-generator witness used by algebraic consumers that read Noetherianity through finitely generated ideal data. The row H is componentwise* hsame *transport for the commutative-ring, ideal, stabilization, and generator rows. The row Q is displayed* Cont *replay for the chain and generator certificates. The row P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ *naming row.*

Lean4 ✓ `BEDC.Derived.NoetherianRingUp.NoetherianRingTasteGate single carrier alignment`

Theorem 3637.2 (Noetherian ring NameCert obligations)**.** *For an accepted* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier* $\mathsf{NoethRing}(C, I, A, G, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ *surface consists exactly of carrier admission for the eight displayed rows, source compatibility with* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$*, finite stabilization access through A, finite-generator access through G, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoetherianRingUp.NoetherianRingCarrier namecert obligations`

Proof. Project the eight rows of Definition 3637.1. The source rows C and I fix the shared commutative-ring and ideal classifier before any chain is inspected.

The stabilization row A and generator row G are then read over that classifier. Componentwise transport H carries each displayed row through hsame, while Q replays the corresponding continuations without replacing the source surface.

Finally, P and N repack the same finite inventory as the local naming certificate. No quotient ideal space, maximal ideal selection, or Hilbert basis proof is introduced by this packet. □

Theorem 3637.3 (Noetherian ascending-chain stability)**.** *Let* $\mathsf{NoethRing}(C, I, A, G, H, Q, P, N)$ *be an accepted* $\mathsf{NoetherianRing}^{\uparrow}$ *carrier. For every displayed ascending chain of ideal rows over the shared* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ideal}^{\uparrow}$ *classifier, the witness A supplies a finite stabilization stage, and H transports that stabilization along the same classifier. Thus any downstream consumer may use* $\mathsf{NoetherianRing}^{\uparrow}$ *as a finite ideal-chain certificate without re-proving stabilization inside its own packet.*

Proof. Begin with the source compatibility rows from Definition 3637.2. They force the chain to be read over the commutative-ring and ideal classifier already present in C and I.

Apply the stabilization witness A to obtain a finite stage for the displayed chain. If the accepted packet is transported through a classified replacement, use the componentwise transport row H to move both the chain entries and the stabilization witness through the shared classifier.

The transported stage remains a stage for the same finite chain surface because Q replays only the displayed continuation data and P prevents provenance escape. The conclusion is therefore a reusable finite ideal-chain certificate for later algebra chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$.

Constructive story: A $\mathsf{NoetherianRing}^{\uparrow}$ packet is generated from a $\mathsf{CommRing}^{\uparrow}$ source, an $\mathsf{Ideal}^{\uparrow}$ source over the same classifier, an ascending-chain stabilization witness, a finite-generator witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3637.1, Definition 3637.2, and Definition 3637.3 seed the finite ideal-chain stabilization interface over CommRingUp, IdealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient ideal spaces, maximal ideal choice, Hilbert basis proof, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite stabilization access, finite-generator access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.NoetherianRingUp.

Chapter 3638

A Concrete Naming Certificate for $\mathsf{LeviCivitaConnection}^{\uparrow}$

Following Kobayashi and Nomizu, the $\mathsf{LeviCivitaConnection}^{\uparrow}$ packet records the finite NameCert surface that ties a Riemannian metric to the unique torsion-free metric-compatible connection visible to curvature consumers. It is grounded in the source-fibre metric packet of chapter 133, the connection packet of chapter 134, and the curvature boundary surface of chapter 135. The packet exports only displayed rows and certificate ledgers; it does not import an ambient smooth connection operator, coordinate Christoffel symbols, or a host-level uniqueness theorem.

Definition 3638.1 (Levi-Civita connection carrier)**.** *A* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, C, T, M, R, H, K, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row whose source-fibre surface supplies the metric data on the displayed tangent fibres.*
2. *C is the* $\mathsf{Connection}^{\uparrow}$ *row carrying the base, fibre, section, tangent-direction, derivative endpoint, and connection provenance visible to the packet.*
3. *T is the torsion-free boundary row, recording that the two displayed tangent-direction orders have no residual antisymmetric endpoint outside the connection ledger.*
4. *M is the metric-compatibility row, recording that the connection derivative preserves the displayed* $\mathsf{RiemannianMetric}^{\uparrow}$ *inner-product row along the inspected tangent window.*
5. *R is the* $\mathsf{Curvature}^{\uparrow}$ *compatibility row, exposing the curvature boundary obtained from the connection row after the torsion-free and metric-compatibility rows have been displayed.*
6. *H records componentwise* hsame *transport for G, C, T, M, R and their dependency rows.*
7. *K records the displayed* Cont *replay from the metric row and connection row through torsion-free verification, metric compatibility, and curvature compatibility.*
8. *P is the* Pkg *provenance over* $\mathsf{LeviCivitaConnection}^{\uparrow}$, $\mathsf{RiemannianMetric}^{\uparrow}$, $\mathsf{Connection}^{\uparrow}$, $\mathsf{Curvature}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$ *row.*

The carrier therefore packages a displayed Riemannian metric together with a displayed connection whose torsion-free, metric-compatible, and curvature-facing rows are all visible in the same finite certificate. It does not contain a coordinate formula for Christoffel symbols, a selected smooth atlas, a global geodesic theorem, or a host-level equality between possible connections.

Lean4 ✓ `BEDC.Derived.LeviCivitaConnectionUp.LeviCivitaConnectionTasteGate single carrier alignment decode encode`

Theorem 3638.2 (Levi-Civita connection NameCert obligations)**.** *For an accepted* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier* $L = (G, C, T, M, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *carrier admission for the displayed metric and connection rows* G *and* C*;*
2. *torsion-free exposure for the boundary row* T*;*
3. *metric-compatibility exposure for the row* M*;*
4. *curvature-facing compatibility for the row* R*;*
5. *componentwise transport for* G, C, T, M, R *through* H*;*
6. *replay exactness for the route recorded by* K*;*
7. *provenance and local naming through* P *and* N*.*

No obligation asks for an unlogged covariant derivative, a coordinate chart calculation, a selected geodesic flow, or a host-level uniqueness principle.

Proof. Project the carrier of Definition 3638.1. The rows G and C are the only metric and connection sources available to the local certificate, and their dependency surfaces are already the public rows of $\mathsf{RiemannianMetric}^{\uparrow}$ and $\mathsf{Connection}^{\uparrow}$.

The torsion-free and metric-compatibility clauses are the two displayed boundary rows T and M. They are read after the connection row has exposed tangent-direction and derivative-endpoint data and after the metric row has exposed the fibrewise inner-product surface.

The curvature-facing clause is exactly R. It can be consumed only after the connection and compatibility rows have been replayed by K, so the downstream $\mathsf{Curvature}^{\uparrow}$ read receives a finite boundary route rather than a detached tensor object.

Transport and replay exactness are then componentwise projections from H and K. The package row P and local name row N repack these rows as the whole local NameCert surface, leaving no coordinate for the excluded ambient objects. □

Theorem 3638.3 (Levi-Civita connection curvature compatibility)**.** *Every* $\mathsf{Curvature}^{\uparrow}$ *consumer of an accepted* $\mathsf{LeviCivitaConnection}^{\uparrow}$ *carrier reads curvature through the finite route*

$$G \longmapsto C \longmapsto T \longmapsto M \longmapsto R.$$

Thus the curvature row is downstream of the displayed $\mathsf{RiemannianMetric}^{\uparrow}$ *metric packet, the displayed* $\mathsf{Connection}^{\uparrow}$ *derivative packet, the torsion-free boundary, and the metric-compatibility row. It cannot be read as an independent coordinate curvature tensor, a host connection form, or a curvature object detached from the metric-compatible connection certificate.*

Proof. Start with a curvature-facing request. The carrier has only one curvature coordinate, namely R, and R is declared as a compatibility row over the displayed connection row C.

The connection row C is not detached from the metric. The torsion-free row T and metric-compatibility row M sit between C and R in the replay row K, while G supplies the metric source that M preserves.

Apply the connection curvature-boundary surface from Definition 134.5 and the curvature source obligations of chapter 135. These existing rows explain how a connection boundary becomes consumable by $\mathsf{Curvature}^{\uparrow}$.

Repacking the projected rows gives the displayed route $G \mapsto C \mapsto T \mapsto M \mapsto R$. Since no coordinate curvature tensor, host connection form, or detached curvature object appears in the carrier, no such object is exported to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3638.1, Definition 3638.2, and Definition 3638.3 seed the displayed metric-compatible torsion-free connection route from $\mathsf{RiemannianMetric}^{\uparrow}$ and $\mathsf{Connection}^{\uparrow}$ rows through torsion-free, metric-compatible, curvature-facing, transport, replay, provenance, and local naming boundaries over $\mathsf{Curvature}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeviCivitaConnection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close coordinate Christoffel symbols or formulae, selected smooth atlases, global geodesic theorems or flows, host-level connection uniqueness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, torsion-free exposure, metric-compatibility exposure, curvature-facing compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LeviCivitaConnectionUp.

Constructive story: A $\mathsf{LeviCivitaConnection}^{\uparrow}$ packet is generated from displayed Riemannian metric and connection rows, torsion-free and metric-compatibility boundary rows, a curvature-facing compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3639

A Concrete Naming Certificate for $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ packet records completion as a finite transformation of metric names rather than as an ambient quotient or selected limit. It sits on the dyadic endpoint discipline of chapter 291, the finite observation route of chapter 126, the regular rational sequence row of chapter 301, the terminal real-name seal of chapter 38, the regular-Cauchy completion packet of chapter 1639, the functor rows of chapter 423, and the extension-uniqueness surface of chapter 397.

Definition 3639.1 (Bishop Cauchy completion functor carrier)**.** *A* $\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, R, E, F, U, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, containing the named points, displayed distance observations, and finite tolerance ledger read by the source classifier.*
2. *T is the target metric row, containing the metric surface into which transported Cauchy data are read.*
3. *R is the regular-Cauchy family row, exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules and stream windows used by the completion handoff.*
4. *E is the completion seal row, recording the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *boundary reached from S and R.*
5. *F is the transported map row, storing the displayed map action on source rows and the finite replay by which regular-Cauchy families are sent to target completion rows.*
6. *U is the extension-uniqueness row, read through the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *surface and tied to the same source, target, regular-Cauchy, and seal rows.*
7. *H records componentwise* hsame *transport for source, target, regular-Cauchy, completion seal, transported map, extension, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for a completion-facing consumer from S, T, R through E, F, U and back to the local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BishopRegularCauchyCompletio CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$ *evidence.*

The carrier contains no quotient completion, no selected limit, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyCompletionFunctorUp.BishopCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 3639.2 (Bishop Cauchy completion functor NameCert obligations)**.** *For every accepted carrier $Q = (S, T, R, E, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$ *surface consists of six finite obligation rows: carrier admission for $S, T, R, E, F, U, H, C, P, N$; classifier stability by componentwise* hsame *transport; functorial map transport through the displayed source, target, regular-Cauchy, and seal rows; unit handoff from source metric points into the completion seal; extension uniqueness through the row U; and non-escape from quotient completions, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is coordinate projection from Definition 3639.1. The packet exposes the source and target metric rows, regular-Cauchy family row, completion seal, transported map row, extension row, and the structural rows H, C, P, N as one finite history.

Classifier stability is componentwise. The row H supplies the only equality-like transport, and C replays the same completion-facing route through S, T, R, E, F, U whenever a consumer replaces one accepted packet by a classified mate.

Functorial map transport and unit handoff are sibling-row compositions. The source row S and regular-Cauchy row R feed the completion seal E; the transported map row F carries that route to the target row T; and the unit read is the special case in which a displayed source point enters the same seal route without naming a host limit.

Extension uniqueness is read only from U after E and F have been fixed. Since no carrier coordinate stores a quotient completion, selected representative, choice schedule, or ambient Real-completeness theorem, coordinate exhaustion gives the non-escape clauses. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$.

Constructive story: A BishopCauchyCompletionFunctor$^{\uparrow}$ packet is generated from source and target metric rows, a regular-Cauchy family row, a completion seal row, a transported map row, an extension-uniqueness row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 3639.1 and Definition 3639.2 bind the seed carrier and its finite obligation rows over the displayed completion-functor surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not assert quotient completion construction, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, functorial map transport, unit handoff, extension uniqueness, and non-escape rows for the displayed Bishop completion-functor packet.

Chapter 3640

A Concrete Naming Certificate for $\mathsf{RealLocatedOrder}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealLocatedOrder}^{\uparrow}$ packet names located comparison for two displayed real names by reading only regular Cauchy data, finite stream windows, dyadic tolerance budgets, a located comparison row, and an apartness handoff before any completion-facing consumer sees an order claim. It consumes the terminal $\mathsf{Real}^{\uparrow}$ seals of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the positive-separation boundary of chapter 304, and the finite located-order test of chapter 2112. The packet is a public constructive located-order certificate over $\mathsf{Real}^{\uparrow}$ names; it is not host trichotomy, host real equality, quotient stream equality, selected limits, countable choice, or ambient real completeness.

Definition 3640.1 (Real located-order carrier)**.** *A* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, R, S, D, O, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Real}^{\uparrow}$ *seal rows whose order relation is being read.*
2. *R stores the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks for the two real names.*
3. *S stores the shared* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to inspect the paired approximants.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that fixes the visible precision budget.*
5. *O is the located comparison row imported from the* $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$ *surface.*
6. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *handoff row, present only when the comparison exposes positive separation.*
7. *H stores componentwise* hsame *transport for the two real seals, readbacks, windows, dyadic tolerances, located comparison, apartness handoff, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from Real-name reads through regular Cauchy readbacks, stream windows, dyadic tolerances, located comparison, and apartness handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RealLocatedOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyRealLocatedOrderTest}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, R, S, D, O, A, H, C, P, N$. It has no coordinate for host trichotomy, host real equality, quotient stream equality, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderTasteGate single carrier alignment`

Theorem 3640.2 (Real located-order NameCert obligations)**.** *For every accepted* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier* $L = (X, Y, R, S, D, O, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{Real}^{\uparrow}$ *seals, paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget, located comparison row, optional* $\mathsf{ApartnessReal}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderCarrier namecert obligations`

Proof. Project the carrier of Definition 3640.1. The two real names are visible only as the seal rows X, Y, and their numerical reads pass through the paired regular rational row R.

The finite windows S and dyadic tolerance ledger D are then the only inspection surface available to the located comparison row O. Thus the local NameCert reads order evidence as the displayed CauchyRealLocatedOrderTest route rather than as a host order theorem.

Finally read A, H, C, P, N. The apartness row is a handoff when positive separation is displayed, while the structural rows transport, replay, source, and name the same finite route. Since no remaining coordinate stores trichotomy, host equality, quotient stream equality, selected limits, choice, or ambient completeness, the listed rows exhaust the obligation surface. □

Theorem 3640.3 (Real located-order apartness handoff)**.** *For every accepted* $\mathsf{RealLocatedOrder}^{\uparrow}$ *carrier* $L = (X, Y, R, S, D, O, A, H, C, P, N)$*, an apartness-facing consumer may read positive separation only through the route*

$$X, Y \longmapsto R \longmapsto S \longmapsto D \longmapsto O \longmapsto A$$

together with H, C, P, N*. The handoff exports an* $\mathsf{ApartnessReal}^{\uparrow}$ *row when the located comparison displays a positive gap, but it exports no host real equality test, quotient-stream equality, trichotomy, selected limit, countable-choice selector, or ambient completeness witness.*

Lean4 ✓ `BEDC.Derived.RealLocatedOrderUp.RealLocatedOrderCarrier apartness handoff`

Proof. Start with the located comparison row in Definition 3640.1. The row O is downstream of the two Real seals, paired regular readbacks, finite windows, and dyadic tolerance ledger.

Use the apartness clause of the carrier. The row A is not an independent equality or order oracle; it is the ApartnessReal handoff available only when the displayed located comparison has exposed positive separation.

The structural rows H, C, P, N preserve this same packet route. Therefore a consumer can export the apartness handoff, but cannot bypass the displayed rows to obtain host equality, quotient stream equality, trichotomy, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealLocatedOrder}^{\uparrow}$ consumer must exhibit two RealUp seals, paired RegSeqRat readbacks, StreamName finite windows, a DyadicRatCore tolerance budget, a CauchyRealLocatedOrderTest row, an ApartnessReal handoff when positive separation is claimed, componentwise transport, displayed replay, provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore, apartnessreal, cauchy-real-located-order-test: no listed sibling carries the whole packet $L = (X, Y, R, S, D, O, A, H, C, P, N)$

because each sibling supplies only real sealing, rational readback, finite windows, tolerance budgets, apartness handoff, or located comparison while this chapter binds all rows as one Real-facing located-order certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}$.

Constructive story: A $\mathsf{RealLocatedOrder}^{\uparrow}$ packet is generated from two RealUp seals, paired RegSeqRat readbacks, StreamName finite windows, a DyadicRatCore tolerance budget, a CauchyRealLocatedOrderTest located comparison row, an ApartnessReal handoff when positive separation is displayed, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3640.1, Definition 3640.2, and Definition 3640.3 seed the finite constructive located-order certificate over two displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocatedOrder}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, host real equality, quotient stream equality, selected limits, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, apartness handoff, and TasteGate field faithfulness for BEDC.Derived.RealLocatedOrderUp.

Chapter 3641

A Concrete Naming Certificate for $\mathsf{RealSequenceSpace}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of sequences of real numbers, the $\mathsf{RealSequenceSpace}^{\uparrow}$ packet records a real-valued sequence carrier as finite BEDC route data rather than as a quotient of streams or a completed function space. It sits between the finite stream windows of chapter 126, the regular rational names of chapter 301, the dyadic tolerance core of chapter 291, the finite real-name boundary of chapter 38, and the Cauchy convergence handoff of chapter 1509.

Definition 3641.1 (RealSequenceSpace carrier)**.** *A* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, E, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite sequence windows visible to the carrier.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row supplying regular rational names for the inspected real-valued coordinates.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those inspected coordinates.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the scheduled regular rows and dyadic tolerances have been displayed.*
5. *K is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff row for consumers that ask for Cauchy behavior of the displayed real-valued sequence.*
6. *H records componentwise* hsame *transport for the stream schedule, regular names, dyadic ledger, Real seal, Cauchy handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each finite window request through S, R, D, E, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealSequenceSpaceUp.RealSequenceSpaceTasteGate single carrier alignment`

Theorem 3641.2 (RealSequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier* $Q = (S, R, D, E, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$ *surface consists exactly of finite* BHist *packet rows over* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$*:*

1. *carrier admission for the displayed stream schedule, regular rational coordinate names, dyadic tolerance ledger, Real seal, Cauchy-criterion handoff, transport, replay, provenance, and local name;*
2. *finite-window classification, requiring every coordinate read to pass through the* $\mathsf{StreamName}^{\uparrow}$ *row* S*;*
3. *regular-coordinate admission, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *to be read only at windows named by* S*;*
4. *tolerance admission, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *to accompany every displayed regular coordinate request;*
5. *Real-seal ordering, requiring the* $\mathsf{Real}^{\uparrow}$ *row* E *to be downstream of* S, R, D*;*
6. *Cauchy-handoff ordering, requiring the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *row* K *to consume only the same displayed* S, R, D, E *route;*
7. *transport and replay stability through* H, C, P, N*.*

No obligation supplies quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.

Proof. Project the carrier of Definition 3641.1. The packet exposes only the stream schedule, regular rational coordinate rows, dyadic tolerance ledger, Real seal, Cauchy-criterion handoff, and structural rows.

The finite-window obligations are then forced. A coordinate read has no source except S, the regular name row R is read only at those windows, and the dyadic ledger D records the tolerance data attached to the same request.

The Real and Cauchy rows are downstream. The seal E is reached only after the displayed S, R, D route, and the handoff K reuses that same route when a Cauchy-facing consumer reads the sequence.

Finally the rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier definition contains none of the excluded host-side data, the listed rows exhaust the local NameCert surface. □

Theorem 3641.3 (RealSequenceSpace Cauchy criterion handoff)**.** *Every* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier* $Q = (S, R, D, E, K, H, C, P, N)$ *reads a finite* BHist *packet route over*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{CauchyConvergenceCriterion}^{\uparrow}$$

in the ordered form

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K.$$

Thus a convergence-facing read may inspect only scheduled finite windows, regular rational coordinate rows, dyadic tolerance rows, the finite Real seal, and the displayed Cauchy handoff. It cannot extract quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness from the sequence-space packet.

Proof. Start from an accepted convergence-facing read. By Definition 3641.2, the read first enters the finite $\mathsf{StreamName}^{\uparrow}$ schedule and then the corresponding $\mathsf{RegSeqRat}^{\uparrow}$ coordinate rows.

The tolerance and seal rows are fixed by the same packet. The $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D records the precision budget for the inspected coordinates, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of S, R, D.

The $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ row K consumes only that displayed route. The structural rows H, C, P, N preserve and name the same finite packet, so no quotient-stream representative, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient completeness theorem is available to the consumer. □

Theorem 3641.4 (RealSequenceSpace finite-window classifier determinacy)**.** *Let*

$$Q = (S, R, D, E, K, H, C, P, N) \qquad \textit{and} \qquad Q' = (S', R', D', E', K', H', C', P', N')$$

be accepted $\mathsf{RealSequenceSpace}^{\uparrow}$ *carriers. If Q and Q' display the same* $\mathsf{StreamName}^{\uparrow}$ *schedule, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate rows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{Real}^{\uparrow}$ *seal, the same* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff, and componentwise matching* hsame *and* Cont *rows, then their public classifier reads are the same. Conversely, any public classifier disagreement between Q and Q' must occur in one of those displayed finite rows; it cannot be hidden in quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness.*

Proof. Project both carriers through Definition 3641.1. The only classifier coordinates exposed by either packet are the stream schedule, regular rational coordinate rows, dyadic tolerance ledger, Real seal, Cauchy handoff, transport, replay, provenance, and local naming rows.

Apply Definition 3641.2 to both packets. The theorem fixes the ordering of finite-window admission, regular-coordinate admission, tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability, so matching displayed rows force matching public reads at every permitted coordinate.

It remains to exclude hidden disagreement. By Definition 3641.3, every convergence-facing read factors through the same ordered route

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K.$$

Thus no quotient stream, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient completeness theorem can supply an additional classifier coordinate. Any disagreement must therefore be visible in one of the displayed finite rows. □

Theorem 3641.5 (RealSequenceSpace window-seal obligation closure)**.** *For every accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier $Q = (S, R, D, E, K, H, C, P, N)$, the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *handoff,* hsame *transport,* Cont *replay, provenance, and local* $\mathsf{NameCert}$ *jointly close the local obligation surface. Every sequence-space consumer must read*

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$$

before any public sequence-space classifier is exported, and any classifier disagreement must appear in one displayed finite row rather than in a quotient stream, selected limit, countable-choice family, trichotomy branch, host real equality, or ambient Real completeness theorem.

Proof. Project the carrier of Definition 3641.1. The projection exposes exactly the window schedule S, coordinate row R, tolerance ledger D, Real seal E, Cauchy handoff K, and structural rows H, C, P, N.

Apply Definition 3641.2. This fixes the row ordering from finite-window admission through regular-coordinate admission, dyadic tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability.

Apply Definition 3641.3 for convergence-facing reads. Such a consumer can inspect only the ordered route $S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$, so the Cauchy side receives no additional host equality or completeness coordinate.

Finally apply Definition 3641.4. Matching displayed rows force matching public reads, and any remaining classifier disagreement is therefore visible in one displayed finite row rather than hidden in a quotient-stream or host-equality layer. □

Theorem 3641.6 (RealSequenceSpace tail-zero subspace route)**.** *Let $Q = (S, R, D, E, K, H, C, P, N)$ be an accepted* $\mathsf{RealSequenceSpace}^{\uparrow}$ *carrier equipped with a displayed dyadic tail modulus T whose entries are read through the same* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *coordinate rows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger as S, R, D. Then every tail-zero subspace consumer factors through the route*

$$S \longrightarrow R \longrightarrow D \longrightarrow E \longrightarrow K$$

before it exports the tail-zero surface. The extra tail claim is visible only as the displayed D, K-facing ledger T; it does not add quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 3641.1. The sequence source, regular coordinate readback, dyadic tolerance rows, Real seal, and Cauchy handoff are already the rows S, R, D, E, K, and the tail modulus T is displayed over the same finite windows.

Apply Definition 3641.2 and Definition 3641.3. These results force every Real-facing or Cauchy-facing read to reuse the ordered route through S, R, D, E, K before any public sequence-space conclusion is exported.

The tail-zero subspace surface therefore adds no new equality coordinate. It records that the dyadic tail requests in T are ledger entries facing D and K, while the structural rows H, C, P, N transport and name the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSequenceSpace}^{\uparrow}$ packet is generated from a finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ coordinate rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{Real}^{\uparrow}$ seal row, a $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3641.1, Definition 3641.2, and Definition 3641.3 seed the real-valued sequence-space carrier over finite StreamName, RegSeqRat, DyadicRatCore, Real, and Cauchy convergence rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window classification, regular-coordinate admission, dyadic tolerance admission, Real-seal ordering, Cauchy-handoff ordering, and transport stability for the same displayed packet.

Chapter 3642

A Concrete Naming Certificate for $\mathsf{FrechetUrysohn}^{\uparrow}$

Following Frechet and Urysohn's sequential analysis of closure in topology, the $\mathsf{FrechetUrysohn}^{\uparrow}$ packet records the finite data by which a point is certified as sequentially reachable from a displayed subset rather than assuming an ambient topological closure operator. It sits between the finite real-valued sequence surface of chapter 3641, the sequential compactness packet of chapter 2372, and the finite stream windows of chapter 126.

Definition 3642.1 (Frechet-Urysohn carrier)**.** *A* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, S, W, R, A, X, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology carrier row, containing the named point, the displayed subset, and the finite neighborhood probes used by the closure read.*
2. *S is the sequence-space row over* $\mathsf{RealSequenceSpace}^{\uparrow}$*, recording the accepted real-valued or chart-valued sequence route used to approach the named point.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule for the finite initial segments and tail windows visible to the classifier.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the coordinate reads that witness convergence through S and W.*
5. *A is the closure-witness row, displaying that every inspected tail window lies in the named subset and that the limit read is the named point.*
6. *X is the transport row for the topology probes, sequence-space rows, StreamName windows, Real seal, closure witness, replay, provenance, and local naming data.*
7. *H records componentwise* hsame *transport along the same displayed rows.*
8. *C records displayed* Cont *replay from a closure request through T, S, W, R, A, X.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetUrysohn}^{\uparrow}$, $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$ *row.*

The classifier has no coordinate for arbitrary nets, choice-selected sequences, host closure equality, excluded middle over neighborhoods, or an ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.FrechetUrysohnUp.FrechetUrysohnTasteGate single carrier alignment`

Theorem 3642.2 (Frechet-Urysohn NameCert obligations)**.** *For every accepted* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier* $Q = (T, S, W, R, A, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$ *surface consists exactly of finite* BHist *rows for topology admission, sequence-space admission, StreamName windowing, Real-seal ordering, closure-witness display, transport, replay, provenance, and local naming. Every sequential-closure classifier read must pass through* $T, S, W, R, A, X, H, C, P, N$ *in that order, and any disagreement between two accepted packets must be visible in one of those displayed rows.*

Proof. Project the carrier of Definition 3642.1. The packet exposes the topology row, the real sequence-space row, the finite StreamName windows, the Real seal, and the closure-witness row before any structural replay is available.

The NameCert obligations are therefore row-local. Topology admission is read from T, the sequence route from S, the finite windows from W, the coordinate seal from R, and the sequential closure witness from A.

The structural rows X, H, C, P, N then transport, compare, replay, source, and name the same finite packet. Since the carrier contains none of the excluded host-side coordinates, a classifier disagreement cannot be hidden outside the displayed rows. □

Theorem 3642.3 (Frechet-Urysohn sequential closure handoff)**.** *Every consumer that reads a point as belonging to the sequential closure of a displayed subset through an accepted* $\mathsf{FrechetUrysohn}^{\uparrow}$ *carrier factors through*

$$\mathsf{RealSequenceSpace}^{\uparrow}, \qquad \mathsf{SequentialCompact}^{\uparrow}, \qquad \mathsf{StreamName}^{\uparrow}, \qquad \mathsf{Real}^{\uparrow}$$

and then through the closure-witness row A*. The handoff exports only the finite sequence windows, the accepted sequence-space route, the Real seal, and the displayed tail-in-subset witness; it does not export arbitrary nets, countable choice, host closure equality, or ambient topological compactness.*

Proof. Start with an accepted sequential-closure read. By Definition 3642.2, the read first enters the topology row T and then the sequence-space row S with its displayed StreamName window schedule W.

The Real seal R and the closure-witness row A are downstream of the same sequence route. The witness can therefore certify only the inspected tail windows and the displayed limit read named by the packet.

The sequential compactness reference is consumed as a finite comparison boundary rather than as an ambient compactness theorem. The structural rows X, H, C, P, N preserve this same route, so the consumer receives a sequential handoff but no hidden net, selected sequence, host equality, or external closure principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetUrysohn}^{\uparrow}$ packet is generated from a topology carrier row, a real sequence-space row, StreamName windows, a Real seal, a closure-witness row, transport, componentwise hsame rows, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3642.1, Definition 3642.2, and Definition 3642.3 seed the sequential-closure carrier over finite topology, sequence-space, window, Real, and witness rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetUrysohn}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FrechetUrysohnUp.FrechetUrysohnTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim arbitrary nets, countable choice, host closure equality, excluded middle over neighborhoods, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-space admission, finite-window ordering, Real-seal ordering, closure-witness display, and transport stability for the same packet.

Chapter 3643

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ packet realizes the boundary-phase audit face of chapter 9463 as finite NameCert data. It sits after the observer-expansion ledger of chapter 3486, the boundary-geodesic route of chapter 3491, the horocycle packet of chapter 3534, the phase-ledger horocycle packet of chapter 3591, and the boundary-fixed disk transport route of chapter 3500. The packet names an audit row for checking that a visible boundary-phase read has passed through the displayed hyperbolic ledger, boundary, horocycle, and transport surfaces; it is not a second horocycle carrier, not a host ideal-boundary quotient, and not a physical observer trajectory.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ read factors through displayed $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows before it exports a boundary-phase audit verdict.

Independence witness: hyperbolic-expansion-ledger, hyperbolic-geodesic-boundary, hyperbolic-horocycle, phase-ledger-horocycle, poincare-boundary-fixed-transport: the expansion ledger records records-side growth without a boundary-phase audit verdict; the geodesic-boundary packet records far-end incidence without horocycle phase accounting; the horocycle packet records boundary tangency without focused phase-ledger provenance; the phase-ledger horocycle packet records the boundary-transport route without a separate audit verdict; and the boundary-fixed transport packet records preserved disk transport without ledger-phase coverage. None is bijective to $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$.

Definition 3643.1 (Hyperbolic boundary-phase audit carrier)**.** *A* $\mathsf{HyperbolicBoundaryPhaseAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, G, Q, L, T, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ *observer-expansion ledger row.*
2. *G is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary-geodesic incidence row.*
3. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *boundary-tangent row.*
4. *L is the* $\mathsf{PhaseLedgerHorocycle}^{\uparrow}$ *focused phase-ledger route.*

5. *T is the* PoincareBoundaryFixedTransport$^{\uparrow}$ *boundary-preserving disk-transport row.*
6. *V is the audit verdict row, recording that the visible boundary-phase read has consumed X, G, Q, L, T before any public verdict is exported.*
7. *H records componentwise* hsame *transport for the expansion ledger, boundary-geodesic, horocycle, phase-ledger, transport, audit, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the same ordered audit route.*
9. *P is the* Pkg *provenance over* HyperbolicBoundaryPhaseAudit$^{\uparrow}$, HyperbolicExpansionLedger$^{\uparrow}$, HyperbolicGeodesicB HyperbolicHorocycle$^{\uparrow}$, PhaseLedgerHorocycle$^{\uparrow}$, PoincareBoundaryFixedTransport$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$ *row.*

The carrier exports no host hyperbolic plane, quotient ideal-boundary equality, selected boundary representative, global clock, physical observer path, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryPhaseAuditUp.HyperbolicBoundaryPhaseAuditTasteGate`
`single carrier alignment`

Theorem 3643.2 (Hyperbolic boundary-phase audit NameCert obligations)**.** *For every accepted* HyperbolicBoundaryPhaseAudit$^{\uparrow}$ *carrier $A = (X, G, Q, L, T, V, H, C, P, N)$, the local* NameCert$_{\text{HyperbolicBoundaryPhaseAu}}$ *surface consists exactly of observer-expansion ledger admission, boundary-geodesic incidence, horocycle boundary tangency, focused phase-ledger route admission, boundary-fixed transport, audit verdict exposure, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the accepted carrier. The analytic route begins with the expansion ledger X, then reads the boundary-geodesic row G, the horocycle row Q, the phase-ledger horocycle row L, and the boundary-fixed transport row T.

The audit verdict V is downstream of those five rows. It therefore cannot certify a boundary-phase read unless the expansion, boundary, horocycle, phase-ledger, and transport surfaces have already been displayed.

The structural rows H, C, P, N preserve and name that route. Since the carrier has no coordinate for a host hyperbolic plane, quotient boundary equality, selected boundary representative, global clock, physical observer path, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 3643.3 (Hyperbolic boundary-phase audit route nonescape)**.** *Every boundary-phase audit verdict exported by an accepted* HyperbolicBoundaryPhaseAudit$^{\uparrow}$ *packet factors through*

$$\text{HyperbolicExpansionLedger}^{\uparrow} \longmapsto \text{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \text{HyperbolicHorocycle}^{\uparrow} \longmapsto \text{PhaseLedgerHorocycle}$$

before the audit verdict row is visible. The verdict exports only the finite audit route carried by $A = (X, G, Q, L, T, V, H, C, P, N)$, not a host ideal-boundary quotient, a selected boundary representative, a physical observer trajectory, a global clock, or a completed hyperbolic surface.

Proof. Start with the NameCert obligations of Definition 3643.2. They place the expansion ledger, boundary-geodesic row, horocycle row, phase-ledger horocycle route, and boundary-fixed transport before the audit verdict.

Read the sibling packets in order. Definition 3486.3 supplies the records-side expansion route, Definition 3491.3 supplies the boundary-geodesic handoff, Definition 3534.2 supplies the horocycle boundary-tangency route, and Definition 3591.2 supplies the focused phase-ledger route.

Finally consume the boundary-fixed transport row and the structural rows H, C, P, N. Because no other carrier coordinate exists, the audit verdict cannot escape to host boundary equality, selected representatives, physical observer motion, global clocks, or completed hyperbolic geometry. □

The Lean follow-up for this packet must provide explicit Nontrivial HyperbolicBoundaryPhaseAudit$^{\uparrow}$ and FieldFaithful HyperbolicBoundaryPhaseAudit$^{\uparrow}$ TasteGate instances. Those instances must keep $X, G, Q, L, T, V, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$, seed)

Scope closed: Definition 3643.1, Definition 3643.2, and Definition 3643.3 seed the finite boundary-phase audit route over expansion-ledger, boundary-geodesic, horocycle, phase-ledger, boundary-fixed transport, audit, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicBoundaryPhaseAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperbolic geometry, quotient ideal-boundary equality, selected boundary representatives, physical observer trajectories, global clocks, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, expansion-ledger routing, boundary-geodesic handoff, horocycle incidence, phase-ledger coverage, boundary-fixed transport, audit verdict exactness, TasteGate nontriviality, FieldFaithful coverage, and local naming for BEDC.Derived.HyperbolicBoundaryPhaseAuditUp.

Constructive story: A HyperbolicBoundaryPhaseAudit$^{\uparrow}$ packet is generated from a HyperbolicExpansionLedger row, HyperbolicGeodesicBoundary row, HyperbolicHorocycle row, PhaseLedgerHorocycle route, PoincareBoundaryFixedTransport row, audit verdict row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3644

A Concrete Naming Certificate for $\mathsf{SardTheoremFiniteJet}^{\uparrow}$

Following Sard and the finite-jet presentation of differential topology, the $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet records the displayed local data by which a smooth map is inspected through finitely many derivative, chart, rank, and critical-value rows. It sits after the Frechet derivative surface of chapter 3634, the local inverse-function route of chapter 2055, the comparison discipline of chapter 1912, and the charted carrier of chapter 96. The packet names a finite jet certificate for critical-value exclusion windows; it is not the classical full measure-zero Sard theorem, not a transversality theorem, not a choice-selected regular-value principle, and not a standard bridge.

Definition 3644.1 (Sard theorem finite-jet carrier)**.** *A* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (M, F, D, R, V, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Manifold}^{\uparrow}$ *chart row naming the source and target chart windows in which the finite jet is read.*
2. *F is the displayed map row, containing only the finite map observations inspected by the packet.*
3. *D is the* $\mathsf{FrechetDerivative}^{\uparrow}$ *jet row, recording the derivative orders and finite remainder windows consumed by the critical-value test.*
4. *R is the rank-defect row, displaying which derivative minors are inspected and where the finite packet records criticality.*
5. *V is the candidate value-window row, naming the finite output windows whose regular-value status is being tested.*
6. *L is the local inverse or mean-value ledger supplied through* $\mathsf{InverseFunction}^{\uparrow}$ *and* $\mathsf{MeanValueTheorem}^{\uparrow}$ *when a displayed noncritical branch is certified.*
7. *H records componentwise* hsame *transport for the chart, map, derivative, rank, value-window, ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite value-window request through the chart, map, derivative, rank, and local ledger rows.*
9. *P is the* Pkg *provenance over* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{InverseFunction}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for ambient measure zero, global transversality, choice-selected regular values, host smooth-function spaces, or a standard bridge.

Lean4 ✓ `BEDC.Derived.SardTheoremFiniteJetUp.TasteGate.SardTheoremFiniteJetTasteGate single carrier alignment`

Theorem 3644.2 (Sard theorem finite-jet NameCert obligations)**.** *For every accepted* $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ *carrier* $J = (M, F, D, R, V, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *charted carrier admission through* $\mathsf{Manifold}^{\uparrow}$ *for the source and target windows;*
2. *finite derivative admission through* $\mathsf{FrechetDerivative}^{\uparrow}$ *for the displayed jet orders and remainders;*
3. *rank-defect and value-window inspection through R and V;*
4. *noncritical local handoff through* $\mathsf{InverseFunction}^{\uparrow}$ *and* $\mathsf{MeanValueTheorem}^{\uparrow}$ *when the packet records a regular finite window;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Every accepted finite-jet regularity read factors through these five rows, and any disagreement between two accepted packets is visible in one of them.

Proof. Project the carrier of Definition 3644.1. The rows $M, F, D, R, V, L, H, C, P, N$ are the whole accepted packet, so the chart, map, derivative, rank, value-window, and local ledger reads are finite BHist data.

The critical-value boundary is read only from D, R, V. The derivative row supplies the inspected jet windows, the rank-defect row records the displayed minors, and the value row records the finite output windows under consideration.

When a noncritical branch is exposed, the local ledger L records the inverse-function and mean-value handoff. That handoff is local to the same chart and derivative rows, so it does not choose a global regular value or import a host transversality principle.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores ambient measure-zero data, global transversality, selected regular values, host smooth-function spaces, or bridge data, the five obligation rows exhaust the NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet must expose $\mathsf{Manifold}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{InverseFunction}^{\uparrow}$, $\mathsf{MeanValueTheorem}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows before any finite-jet regularity read is public.

Independence witness: frechet-derivative, inverse-function, mean-value-theorem, manifold: no listed sibling is bijective to $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ because the Sard finite-jet packet jointly binds chart windows, map observations, derivative jets, rank-defect rows, value windows, local regularity ledger, transport, replay, provenance, and local naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SardTheoremFiniteJet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SardTheoremFiniteJet}^{\uparrow}$ packet is generated from Manifold chart rows, displayed map observations, Frechet derivative jet rows, rank-defect rows, finite value-window rows, an inverse-function and mean-value local ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{SardTheoremFiniteJet}\uparrow}$, seed)

Scope closed: Definition 3644.1 and Definition 3644.2 seed the finite Sard jet certificate over chart, derivative, rank, value-window, local regularity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{SardTheoremFiniteJet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the classical full measure-zero Sard theorem, transversality, choice-selected regular values, host smooth-function spaces, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite derivative admission, rank-defect inspection, value-window inspection, local inverse-function and mean-value handoff, and transport stability for BEDC.Derived.SardTheoremFiniteJetU

Chapter 3645

A Concrete Naming Certificate for $\mathsf{BusemannBoundary}^{\uparrow}$

Following Busemann and Bridson–Haefliger, the $\mathsf{BusemannBoundary}^{\uparrow}$ packet records boundary reads by finite distance-difference rows rather than by a completed host ideal-boundary quotient. It sits after the metric-read surface of chapter 3474, the geodesic far-end carrier of chapter 3491, the boundary-fixed transport route of chapter 3500, and the disk-coordinate surface of chapter 377.

Definition 3645.1 (Busemann boundary carrier)**.** *A* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, o, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *source row, exposing the metric carrier from which all distance-difference reads are taken.*
2. *o is the basepoint row, recording the named origin point and its admission through the same metric source.*
3. *R is the geodesic-ray window row, displaying the finite ray segments and endpoint probes supplied by* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*.*
4. *D is the dyadic distance-difference ledger for the inspected values $d(x, \gamma(t)) - d(o, \gamma(t))$ over the displayed windows.*
5. *L is the boundary-read row, classifying the finite Busemann read against the visible ideal-boundary data of* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*, and* $\mathsf{UnitDisk}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for the metric source, basepoint, ray windows, dyadic ledger, boundary-read row, replay, provenance, and local naming data.*
7. *C records displayed* Cont *replay from a boundary request through X, o, R, D, L.*
8. *P is the* Pkg *provenance over* $\mathsf{BusemannBoundary}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$ *row.*

The classifier has no coordinate for an arbitrary compactification, a host quotient of ideal points, a globally chosen geodesic representative, classical sequential compactness at infinity, or an ambient visual boundary not displayed by the packet.

Lean4 ✓ `BEDC.Derived.BusemannBoundaryUp.BusemannBoundaryTasteGate single carrier alignment`

Theorem 3645.2 (Busemann boundary NameCert obligations)**.** *For every accepted* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier* $B = (X, o, R, D, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$ *surface consists exactly of finite* BHist *rows for carrier admission, ray-window exactness, metric-read stability, boundary classifier laws, and nonescape from the host ideal-boundary quotients. Every boundary consumer must read* X, o, R, D, L *before transport, replay, provenance, and local naming are available, and any disagreement between accepted Busemann reads must occur in one of those displayed rows.*

Proof. Project the carrier of Definition 3645.1. The packet exposes a hyperbolic metric source, a basepoint, geodesic-ray windows, a dyadic distance-difference ledger, and a boundary-read row before any structural replay is present.

Carrier admission is therefore row-local. Ray-window exactness is read from R, metric-read stability from the shared X, o source, and boundary classifier laws from L against the displayed geodesic, transport, and disk rows.

The structural rows H, C, P, N then preserve, replay, source, and name the same finite surface. Since the carrier contains no coordinate for a completed host quotient or an unrecorded compactification, the read cannot escape the displayed ideal-boundary data. □

Theorem 3645.3 (Busemann boundary nonescape)**.** *Every accepted boundary read exported by a* $\mathsf{BusemannBoundary}^{\uparrow}$ *carrier factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \quad \mathsf{HyperbolicGeodesicBoundary}^{\uparrow}, \quad \mathsf{PoincareBoundaryFixedTransport}^{\uparrow}, \quad \mathsf{UnitDisk}^{\uparrow}$$

and then through the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of the same packet. The exported read is therefore a finite distance-difference boundary certificate; it is not a host compactification point, a quotient class of rays, or an ambient visual-boundary object.*

Proof. Begin with an accepted boundary read. By Definition 3645.2, the read first enters the $\mathsf{HyperbolicMetric}^{\uparrow}$ row X and the basepoint row o, so each distance-difference entry is tied to the same displayed metric source.

The ray-window and boundary rows then force the read through $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, and $\mathsf{UnitDisk}^{\uparrow}$. The ledger D records only the finite dyadic distance-difference values attached to those windows, and L classifies only against the displayed boundary data.

Finally H, C, P, N transport, replay, source, and name this same route. No additional coordinate remains for a host ideal-boundary quotient, a global ray representative, or an ambient compactification, so every exported Busemann boundary read stays inside the named packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BusemannBoundary}^{\uparrow}$ packet is generated from a HyperbolicMetric source row, a basepoint row, geodesic-ray windows, a dyadic distance-difference ledger, a boundary-read row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3645.1, Definition 3645.2, and Definition 3645.3 seed the Busemann boundary carrier over finite metric, basepoint, ray-window, distance-difference, and boundary-read rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BusemannBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary compactifications, host ideal-boundary quotients, global geodesic representatives, classical compactness at infinity, ambient visual boundaries, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ray-window exactness, metric-read stability, boundary classifier laws, nonescape from host ideal-boundary quotients, transport, replay, provenance, and local naming for the same displayed packet.

Chapter 3646

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ packet records the unit route from displayed regular Cauchy source data into the completion-facing seal. It sits between the regular-name source of chapter 371, the unit map of chapter 1527, the monadic completion surface of chapter 378, and the real completion boundary of chapter 360. Its finite reads pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as exposed in chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the source-to-completion unit as finite BEDC data; it is not quotient completion equality, a selected Cauchy representative, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3646.1 (RegularCauchyCompletionUnit carrier)**.** *A* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (G, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the source row, read either as a* $\mathsf{RegularCauchyName}^{\uparrow}$ *packet or as the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy row already visible to a completion consumer.*
2. *R is the unit-admission row stating that the displayed source row is admitted as the unit image before any completion-facing seal is consumed.*
3. *S is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows through which the source row is inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those source windows and precision requests.*
5. *E is the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after G, R, S, D have been displayed.*
6. *H stores componentwise* hsame *transport for the source, unit-admission row, stream windows, dyadic tolerance ledger, seal handoff, continuation, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from source inspection through unit admission, finite windows, tolerance readback, and seal handoff.*
8. *P records* Pkg *provenance over* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$, $\mathsf{RegularCauchyName}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, a selected regular-Cauchy representative, countable-choice schedules, host sequence equality, an ambient completion theorem, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionUnitUp.RegularCauchyCompletionUnitTasteGate single carrier alignment`

Theorem 3646.2 (RegularCauchyCompletionUnit NameCert obligations)**.** *For an accepted* RegularCauchyComplet *carrier $U = (G, R, S, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$ *surface consists exactly of source admission through G, unit-route admission through R, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance readback through D,* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3646.1. Its displayed source row is G, and the unit-admission row is R.

The finite observation surface is S, D. The stream windows expose only scheduled source reads, and the dyadic ledger records only the tolerance data needed before completion-facing consumption.

The seal handoff E is downstream of those rows. The structural rows H, C, P, N transport, replay, source, and name the same route, so the local certificate is exhausted by the listed fields. □

Theorem 3646.3 (RegularCauchyCompletionUnit seal handoff)**.** *Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* RegularCauchyCompletionUnit *packet reaches the seal row E only through the route*

$$G \longmapsto R \longmapsto S \longmapsto D \longmapsto E$$

together with H, C, P, N. Thus the unit image is consumed as displayed regular Cauchy source data with finite windows and dyadic tolerance readback, not as a quotient class or selected completion representative.

Proof. Begin with a completion-facing consumer. By the carrier definition, the only source row available to the consumer is G, and the only unit-admission row is R.

The consumer must next pass through the finite windows S and the dyadic tolerance ledger D. These rows are the displayed readback route inherited from the regular-name and regular-rational source surfaces.

Only after that route is available can E be read. The rows H, C, P, N attach transport, replay, provenance, and local naming to the same route, so the seal handoff is the displayed unit route and not a hidden completion object. □

Theorem 3646.4 (RegularCauchyCompletionUnit non-escape)**.** *No accepted* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *packet exports data outside $G, R, S, D, E, H, C, P, N$. In particular, it cannot be used to recover quotient completion equality, a selected regular-Cauchy representative, countable-choice schedules, host sequence equality, an ambient completion theorem, analytic closure of the monad, or completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Inspect the carrier. The source, unit-admission, window, tolerance, seal, transport, replay, provenance, and local-name rows are the only fields.

Apply Definition 3646.2. Every accepted obligation is a projection of those rows or a structural transport, replay, provenance, or naming row for them.

Finally use Definition 3646.3. All completion-facing consumers pass through $G, R, S, D, E, H, C, P, N$, so no excluded quotient, choice, host, monadic closure, or completeness object can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ packet is generated from a RegularCauchyName or RegSeqRat source row, unit admission, finite StreamName windows, a DyadicRatCore tolerance ledger, a RealCauchyCompletion or RealUp seal handoff, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3646.1, Definition 3646.2, Definition 3646.3, and Definition 3646.4 seed the finite source-to-completion unit route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, selected representatives, countable choice, host sequence equality, monadic analytic closure, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, unit-route admission, finite window exposure, dyadic tolerance readback, seal handoff, transport stability, provenance, and non-escape for the displayed packet.

Chapter 3647

A Concrete Naming Certificate for $\mathsf{SobolevTrace}^{\uparrow}$

Following Lions–Magenes and Adams–Fournier, the $\mathsf{SobolevTrace}^{\uparrow}$ packet records finite BEDC boundary trace data for Sobolev carriers. It sits next to the weak-derivative packet of chapter 440, consumes the boundary-control route of chapter 2329, uses the compact-window handoff of chapter 2218, and reads scalar and metric data through chapter 38, chapter 57, and chapter 128. The packet names only the displayed boundary restriction surface; it does not claim a host trace operator on a completed Sobolev quotient.

Definition 3647.1 (Sobolev trace carrier)**.** *A* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, D, B, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, exposing the weak-derivative, integrability, codomain, transport, replay, provenance, and local naming rows already admitted by the Sobolev packet.*
2. *D is the boundary-domain row. It records the finite boundary chart, collar, normal-window, or boundary-read ledger on which the trace is inspected.*
3. *B is the Poincare-facing boundary-control row from* $\mathsf{PoincareInequality}^{\uparrow}$*, used only to keep constant or zero-boundary escape from entering the trace read.*
4. *R is the Rellich–Kondrachov compact-window row, recording the finite compactness or subsequence-window handoff consumed by boundary convergence tests.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *row that supplies the finite distance and Cauchy-readback surface for the boundary values.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for boundary magnitudes, radii, and norm estimates.*
7. *H stores componentwise* hsame *transport for S, D, B, R, M, E and the structural rows.*
8. *C stores the displayed* Cont *replay from a trace request through the Sobolev source, boundary row, Poincare row, compact-window row, metric row, and real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SobolevTrace}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*,* $\mathsf{PoincareInequality}^{\uparrow}$*,* $\mathsf{RellichKondrachov}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$ *row.*

The classifier has no coordinate for an ambient trace operator, quotient equality of boundary functions, selected representative, distributional boundary value, open-cover compactness theorem, or global completeness theorem for the real line.

Lean4 ✓ `BEDC.Derived.SobolevTraceUp.sobolevTraceChapterTasteGate`

Theorem 3647.2 (Sobolev trace NameCert obligations)**.** *For every accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier* $T = (S, D, B, R, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Sobolev source, boundary-domain row, Poincare control row, Rellich–Kondrachov compact-window row, metric or complete-metric row, real seal, transport, replay, provenance, and local naming data;*
2. *source compatibility, requiring every trace request to enter through the* $\mathsf{Sobolev}^{\uparrow}$ *weak-derivative ledger before any boundary row is read;*
3. *boundary-control exactness, requiring* D *and* B *to supply the boundary restriction and normalization data consumed by the trace;*
4. *compact and complete-metric handoff, requiring* R *and* M *to expose only finite convergence, compact-window, distance, and Cauchy-readback rows;*
5. *scalar readback and transport, requiring* E, H, C, P, N *to preserve, replay, source, and name the same finite trace route.*

Proof. Project the carrier. The accepted packet exposes only the rows $S, D, B, R, M, E, H, C, P, N$, so carrier admission is row-local.

The source compatibility row is forced by S. A trace request first reads the $\mathsf{Sobolev}^{\uparrow}$ weak-derivative and magnitude ledger, then reaches the boundary-domain row D and the Poincare control row B.

The compact and complete-metric handoff is similarly row-local. The Rellich–Kondrachov row R supplies only finite compact-window data, and M supplies only the displayed metric or complete-metric readback surface.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier contains no ambient trace operator, host quotient, selected representative, or distributional boundary coordinate, the local NameCert obligations are exhausted by the five displayed rows. □

Lean4 ✓ `BEDC.Derived.SobolevTraceUp.SobolevTraceNamecertObligationSurface`

Theorem 3647.3 (Sobolev trace boundary readback)**.** *Every boundary value read exported by an accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *packet factors through*

$$\mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{PoincareInequality}^{\uparrow} \longmapsto \mathsf{RellichKondrachov}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

and then through the displayed BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows of the same packet. The exported object is a finite boundary readback certificate; it is not a host boundary function, quotient trace class, distributional trace, or selected representative.*

Proof. Start with an accepted trace read. By Definition 3647.2, the read first enters the Sobolev source row S, so it inherits only the weak-derivative and magnitude data visible in the source packet.

The boundary rows D and B then restrict the request to the displayed boundary ledger and Poincare-facing control surface. These rows determine which boundary observations are available before any compactness or completion-facing datum is consumed.

The remaining analytic read passes through R, M, E. Thus compact-window information, metric or complete-metric readback, and scalar magnitude are all finite rows of the same packet.

The structural rows H, C, P, N merely preserve, replay, source, and name this route. No carrier coordinate remains from which a host boundary function, quotient class, distributional trace, or selected representative could be read. □

Theorem 3647.4 (Sobolev trace complete-metric handoff)**.** *Any* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *PDE, variational, or boundary-analysis consumer citing an accepted* $\mathsf{SobolevTrace}^{\uparrow}$ *carrier receives only the complete-metric handoff displayed by* M*, the real scalar seal displayed by* E*, and the upstream rows* S, D, B, R*. In particular, a complete-metric consumer may request Cauchy-readback data for the boundary trace only after the Sobolev source, boundary-control, Poincare, and Rellich–Kondrachov rows have been read.*

Proof. Project a downstream consumer request. The carrier places M after S, D, B, R, so the complete-metric surface cannot be reached before the Sobolev source, boundary row, boundary-control row, and compact-window row are fixed.

The real scalar seal is read through E. It records the displayed boundary magnitudes and radii used by M, not a host completeness theorem for all real-valued boundary functions.

The transport, replay, provenance, and local naming rows keep this handoff inside the same BHist packet. Hence the complete-metric consumer receives only the finite Cauchy-readback route displayed by the trace carrier. □

Falsifiable prediction: Every $\mathsf{SobolevTrace}^{\uparrow}$ boundary read factors through displayed SobolevUp, PoincareInequalityUp, RellichKondrachovUp, MetricUp, CompleteMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any downstream trace consumer receives it.

Independence witness: sobolev, poincare-inequality, rellich-kondrachov, metric, completemetric, real: $\mathsf{SobolevTrace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Sobolev source, boundary-domain row, boundary-control row, compact-window row, complete-metric boundary readback, scalar seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SobolevTrace}^{\uparrow}$ packet is generated from a Sobolev source row, boundary-domain row, Poincare control row, Rellich–Kondrachov compact-window row, metric or complete-metric readback row, real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$, seed)

Scope closed: Definition 3647.1, Definition 3647.2, Definition 3647.3, and Definition 3647.4 seed the finite boundary trace route over Sobolev, Poincare, Rellich–Kondrachov, metric, complete-metric, and real neighbours.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SobolevTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host trace operator, quotient equality of boundary functions, selected representatives, distributional boundary values, open-cover compactness, real-line completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, boundary-control exactness, compact and complete-metric handoff, scalar readback, componentwise transport, replay, provenance, and local naming.

Chapter 3648

A Concrete Naming Certificate for $\mathsf{NoetherianModule}^{\uparrow}$

Following Emmy Noether and standard module theory, the $\mathsf{NoetherianModule}^{\uparrow}$ packet records finite submodule-chain stabilization data over a displayed module source. In BEDC form it sits after $\mathsf{NameCert}_{\mathsf{Module}^{\uparrow}}$ (chapter 46), the commutative scalar source of $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44), the ideal-side Noetherian surface of $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ (chapter 3637), and the ideal carrier of $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81). The packet supplies the module-side finite-chain row consumed by algebraic reduction and kernel-image arguments; it does not assert quotient-module classification, selected bases, maximal-submodule choice, or a host Noetherian theorem.

3648.1 Carrier definition

Definition 3648.1 (Noetherian module carrier)**.** *A* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N).$$

The row M is the $\mathsf{Module}^{\uparrow}$ *source surface. The row R is the scalar* $\mathsf{CommRing}^{\uparrow}$ *source read from the module certificate. The row I is the* $\mathsf{Ideal}^{\uparrow}$ *comparison surface used when scalar-generated submodule rows are inspected. The row A is the ascending-chain stabilization witness: every displayed chain of submodule rows over M carries a finite stage after which all later submodule endpoints classify with that stage. The row G is the finite-generator witness used by algebraic consumers that read the module through finitely generated submodule data. The row H is componentwise* hsame *transport for the module, scalar, ideal-comparison, stabilization, and generator rows. The row Q is displayed* Cont *replay for the submodule-chain and generator certificates. The row P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$ *naming row.*

Lean4 ✓ `BEDC.Derived.NoetherianModuleUp.TasteGate.NoetherianModuleTasteGate single carrier alignment`

Theorem 3648.2 (Noetherian module NameCert obligations)**.** *For an accepted* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier* $\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, source compatibility with* $\mathsf{Module}^{\uparrow}$*,* $\mathsf{CommRing}^{\uparrow}$*,* $\mathsf{NoetherianRing}^{\uparrow}$*, and* $\mathsf{Ideal}^{\uparrow}$*, finite submodule-chain stabilization access through A, finite-generator access through G, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.NoetherianModuleUp.NoetherianModuleCarrier namecert obligations`

Proof. Project the nine rows of Definition 3648.1. The module row M fixes the additive carrier, scalar action, and module classifier before any submodule chain is inspected.

The scalar and ideal comparison rows R and I bind the chain to the same commutative-ring and ideal surfaces used by chapter 3637. Thus the $\mathsf{NoetherianRing}^{\uparrow}$ dependency supplies only the scalar-side finite-chain context; it does not replace the module chain by an ideal chain.

The stabilization row A and generator row G are then read over the displayed module classifier. Componentwise transport H carries the same rows through hsame, while Q replays only the displayed continuations.

Finally, P and N repack this inventory as the local naming certificate. Since no quotient-module classifier, basis selector, maximal-submodule choice, or host Noetherian principle occurs in the carrier, none can occur among the local obligations. □

Theorem 3648.3 (Noetherian submodule-chain stability)**.** *Let* $\mathsf{NoethMod}(M, R, I, A, G, H, Q, P, N)$ *be an accepted* $\mathsf{NoetherianModule}^{\uparrow}$ *carrier. For every displayed ascending chain of submodule rows over the* $\mathsf{Module}^{\uparrow}$ *classifier of* M*, the witness* A *supplies a finite stabilization stage, and* H *transports that stabilization along the same classifier. Thus any downstream algebraic consumer may use* $\mathsf{NoetherianModule}^{\uparrow}$ *as a finite submodule-chain certificate without importing quotient-module classification, selected bases, maximal-submodule choice, or a host Noetherian theorem.*

Proof. Begin with the source compatibility rows from Definition 3648.2. They force the chain to be read over the module classifier already present in M, while R and I keep the scalar and ideal comparison surfaces visible.

Apply the stabilization witness A to the displayed submodule chain. The result is a finite stage inside the same module classifier, not an external quotient or selected basis.

If the accepted packet is transported through a classified replacement, use the componentwise transport row H to move the chain entries and the stabilization witness together. The replay row Q preserves only the displayed continuation data, and the provenance row P prevents an ambient Noetherian principle from entering the route.

The transported stage is therefore a reusable finite submodule-chain certificate for later module consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}$.

Constructive story: A $\mathsf{NoetherianModule}^{\uparrow}$ packet is generated from a $\mathsf{Module}^{\uparrow}$ source, a scalar $\mathsf{CommRing}^{\uparrow}$ source, a $\mathsf{NoetherianRing}^{\uparrow}$ scalar-side chain context, an $\mathsf{Ideal}^{\uparrow}$ comparison surface, a submodule-chain stabilization witness, a finite-generator witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3648.1, Definition 3648.2, and Definition 3648.3 seed the finite submodule-chain stabilization interface over ModuleUp, NoetherianRingUp, CommRingUp, IdealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NoetherianModule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-module classification, selected bases, maximal-submodule choice, host Noetherian module theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite submodule-chain stabilization access, finite-generator access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.NoetherianModuleUp.

Chapter 3649

A Concrete Naming Certificate for $\mathsf{LaplaceTransform}^{\uparrow}$

Following classical Laplace transform theory in analysis, the $\mathsf{LaplaceTransform}^{\uparrow}$ packet records only the finite constructive transform interface visible to BEDC consumers. It sits after the coefficient and convergence rows of chapter 1583, the integral carrier of chapter 123, the graph discipline of chapter 130, the real seal of chapter 38, the regular rational readback of chapter 301, the finite observation discipline of chapter 126, and the dyadic tolerance core of chapter 291. The chapter records finite exponential-kernel, integral-window, transform-output, transport, replay, provenance, and local NameCert rows; it does not assert host analytic continuation, a global function-space quotient, selected improper integrals, or ambient real completeness.

Definition 3649.1 (Laplace transform carrier)**.** *A* $\mathsf{LaplaceTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, I, F, W, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the exponential-kernel row, read through* $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *data for the finitely inspected values of e^{-st}.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *window row naming the displayed interval, mesh or tagged integration data, and tolerance budget used by the transform read.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *input-function row restricted to the finite source observations consumed by I.*
4. *W is the finite product-read window combining the kernel row K with the input row F before the integral row I is consumed.*
5. *O is the transform-output row, a* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff reached only through K, F, W, I.*
6. *H records componentwise* hsame *transport for the kernel, integral window, input function, product reads, output handoff, provenance, and local name.*
7. *C records displayed* Cont *replay from kernel and function reads through the integral window into the output row.*
8. *P is the* Pkg *provenance over* $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^\uparrow}$ *row.*
The classifier compares accepted carriers only through $K, I, F, W, O, H, C, P, N$. It has no coordinate for host analytic continuation, quotient equality of functions, selected improper limits, unbounded integration outside the displayed window, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.LaplaceTransformUp.LaplaceTransformTasteGate single carrier alignment`

Theorem 3649.2 (Laplace transform NameCert obligations)**.** *For an accepted* $\mathsf{LaplaceTransform}^\uparrow$ *carrier $L = (K, I, F, W, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^\uparrow}$ *surface consists exactly of carrier admission, exponential-kernel admission, integral-window exactness, input-function read compatibility, product-window replay, transform-output handoff, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged along the finite route*

$$K, F \longmapsto W \longmapsto I \longmapsto O$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 3649.1. The kernel row K, input-function row F, product-read window W, integral row I, and output row O are stored coordinates before any downstream consumer reads the transform.

The exponential-kernel obligation is confined to the displayed $\mathsf{RealPowerSeries}^\uparrow$ and real-read rows inside K. The input-function obligation is confined to the finite $\mathsf{ContinuousMap}^\uparrow$ observations in F, and the product-window obligation records exactly which paired reads enter the integration window.

Integral-window exactness is the row I. It consumes only the paired reads in W and the displayed tolerance and mesh data carried by the integral interface, so the calculation is a finite BHist route rather than an improper-integral assertion.

The rows H and C transport and replay the same route componentwise. The provenance row P and local name N source the packet, while O is downstream of K, F, W, I. Since none of the excluded host objects appears as a carrier coordinate, the listed obligations exhaust the certificate surface. □

Theorem 3649.3 (Laplace transform exponential-kernel window)**.** *Every accepted* $\mathsf{LaplaceTransform}^\uparrow$ *read of the exponential factor passes through the finite kernel row K before it can enter the product window W and the integral row I. The kernel read is therefore a displayed* $\mathsf{RealPowerSeries}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{StreamName}^\uparrow$, *and* $\mathsf{DyadicRatCore}^\uparrow$ *window over the inspected s, t values, not a host exponential-function oracle and not an analytic-continuation principle.*

Proof. Project the kernel row K from the carrier of Definition 3649.1. The row records only the finitely inspected real and regular-rational reads needed for the exponential kernel.

The product window W consumes K together with the input-function row F. Thus an integral read cannot reach the kernel through any route except the displayed finite kernel row and its stored transport and replay data.

Finally the integral row I consumes W rather than a host exponential function. The transform output O is downstream of this route, so no consumer can replace the kernel window by analytic continuation or an unrecorded exponential oracle. □

Theorem 3649.4 (Laplace transform real seal nonescape)**.** *The output row O of an accepted* $\mathsf{LaplaceTransform}^{\uparrow}$ *packet exports to* $\mathsf{Real}^{\uparrow}$ *only after the displayed route*

$$K, F \longmapsto W \longmapsto I$$

has been transported and replayed through H and C. The real seal records the finite transform-output handoff already present in the packet, and it does not export host analytic continuation, global function-space equality, selected improper limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The output row O is a carrier coordinate, but Definition 3649.1 places it downstream of the kernel row, input-function row, product-read window, and integral row. A $\mathsf{Real}^{\uparrow}$ consumer therefore receives only the output handoff obtained from the displayed finite route.

Componentwise transport through H preserves the same coordinates, and continuation replay through C rebuilds the same dependency route. The package row P and local name N record provenance without adding analytic structure.

The excluded objects are absent from the carrier. There is no host analytic-continuation row, no quotient function-space coordinate, no selected improper-integral limit, no countable-choice selector, and no ambient completeness witness. Hence the real seal is nonescaping. □

Closure status for $\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3649.1, Definition 3649.2, Definition 3649.3, and Definition 3649.4 seed the finite exponential-kernel and integral-window transform boundary over $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LaplaceTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, global function-space equality, selected improper limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponential-kernel admission, integral-window exactness, product-window replay, transform-output handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.LaplaceTransformUp.

Constructive story: A $\mathsf{LaplaceTransform}^{\uparrow}$ packet is generated from a finite exponential-kernel row, an input-function row, a product-read window, an Integral window, a RegSeqRat and Real output handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 3650

A Concrete Naming Certificate for $\mathsf{LipschitzExtension}^{\uparrow}$

Following McShane, Whitney, and Bishop–Bridges constructive metric analysis, the $\mathsf{LipschitzExtension}^{\uparrow}$ packet records the reusable finite certificate by which a displayed Lipschitz map on a displayed subcarrier is read as an extension candidate on an ambient metric carrier. It sits above the map-level surface of chapter 410, the metric carrier of chapter 57, the continuity handoff of chapter 130, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the formula-specific McShane–Whitney certificate of chapter 1941. The chapter isolates the extension interface itself: source inclusion, bound ledger, candidate graph, restriction agreement, continuity handoff, transport, replay, provenance, and local naming.

Definition 3650.1 (Lipschitz extension carrier)**.** *A* $\mathsf{LipschitzExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, A, L, B, F, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *row naming the source metric observations available to the extension request.*
2. *A is the displayed subcarrier row, including membership evidence and inclusion routes into X.*
3. *L is the source* $\mathsf{LipschitzMap}^{\uparrow}$ *row on A, carrying the finite graph window and displayed Lipschitz constant.*
4. *B is the* $\mathsf{RegSeqRat}^{\uparrow}$ *bound row that records the rational tolerance and constant reads used by the extension certificate.*
5. *F is the candidate extension graph row, exposed only as finite ambient graph observations.*
6. *R is the restriction-agreement row, requiring each candidate read over an admitted source point to replay the source* $\mathsf{LipschitzMap}^{\uparrow}$ *read through A, L, B.*
7. *Q is the* $\mathsf{ContinuousMap}^{\uparrow}$ *handoff row for the candidate extension on the displayed ambient window.*
8. *H records componentwise* hsame *transport for the metric, subcarrier, source map, bound, candidate graph, restriction, continuity handoff, provenance, and local name rows.*
9. *C records displayed* Cont *replay from an ambient extension request through $X, A, L, B, F, R, Q, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LipschitzExtension}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{McShaneExtension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function equality, quotient function spaces, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LipschitzExtensionUp.LipschitzExtensionTasteGate single carrier alignment`

Theorem 3650.2 (Lipschitz extension NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzExtension}^{\uparrow}$ *carrier* $E = (X, A, L, B, F, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$ *surface consists exactly of carrier admission, subcarrier inclusion, source Lipschitz-map admission, rational bound readback, candidate graph admission, restriction-agreement exactness, continuity handoff, componentwise transport, displayed replay, provenance, and ledger non-escape. The finite route is*

$$X, A \longmapsto L \longmapsto B \longmapsto F \longmapsto R, Q \longmapsto H, C, P, N.$$

No obligation is discharged by a host extension formula or by equality of ambient functions.

Proof. Carrier admission is projection from Definition 3650.1. The rows $X, A, L, B, F, R, Q, H, C, P, N$ are the complete visible inventory of an accepted packet.

The subcarrier and source-map clauses read A and L. A source value can enter the extension certificate only after the membership and inclusion row A has been displayed and the $\mathsf{LipschitzMap}^{\uparrow}$ row L has exposed the finite graph and bound data.

The rational bound and candidate graph clauses read B and F. The bound is a regular-rational ledger, and the candidate graph is a finite ambient observation row, so neither row selects a global function object or an unrecorded real limit.

Restriction exactness is the row R, while continuity handoff is the row Q. The transport and replay rows H and C preserve the same route componentwise, and P, N source and name it. Since no excluded host object occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3650.3 (Lipschitz extension restriction handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{McShaneExtension}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzExtension}^{\uparrow}$ *packet must read the restriction route*

$$A, L, B \longmapsto R$$

before it can treat the candidate graph row F as an extension of the source Lipschitz row. The handoff exports the displayed agreement between source and ambient rows, not a quotient function equality, not a selected representative, and not an ambient completeness theorem.

Proof. Start with a consumer asking whether the candidate extends the source map. The subcarrier row A is read first, because the question is meaningful only for points admitted by the displayed inclusion route.

Next read the source Lipschitz row L and the rational bound row B. These rows supply the finite source graph and tolerance data against which the candidate graph will be checked.

The restriction-agreement row R then compares the candidate graph row F with the source route over the admitted point. The continuity row Q and structural rows H, C, P, N replay and source the same comparison without adding a host function-space coordinate.

Thus a McShane–Whitney certificate may instantiate the interface, but the interface itself exports only the finite restriction handoff displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3650.1, Definition 3650.2, and Definition 3650.3 seed the finite reusable Lipschitz extension surface over displayed metric, Lipschitz-map, continuity, regular-rational, real, and McShane-extension rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subcarrier inclusion, source Lipschitz admission, rational bound readback, candidate graph admission, restriction exactness, continuity handoff, transport, replay, provenance, and non-escape for BEDC.Derived.LipschitzExtensionUp.

Constructive story: A $\mathsf{LipschitzExtension}^{\uparrow}$ packet is generated from an ambient Metric row, a displayed subcarrier row, a source LipschitzMap row, a regular rational bound ledger, a candidate extension graph, a restriction-agreement row, a ContinuousMap handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3651

A Concrete Naming Certificate for $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$

The $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ packet realizes the anchor-subcube trigger theorem of Definition 9462.1519 as finite NameCert data. It records the displayed anchor, boundary-tile span, coverage-rank, trigger-algebra, blocker, transport, replay, provenance, and local naming rows that let a DNA-window consumer read the median closure route without importing a global codon geometry oracle or a host powerset calculation. The packet consumes the boundary-tile carrier of chapter 3393 and the trigger-algebra route of chapter 3478 before exposing the median-closure read.

Definition 3651.1 (Anchor subcube trigger carrier)**.** *An* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, S, R, G, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the anchor row naming the* CUY *boundary anchor and the admitted coordinate window.*
2. *B is the boundary-tile row listing the outside Y-halfspace observations inspected by the packet.*
3. *S is the subcube-span row recording the axis-aligned span generated by the anchor and the projected outside observations.*
4. *R is the coverage-rank row, including the displayed value of the span dimension used by the trigger theorem.*
5. *G is the trigger-algebra row recording the finite median-closure operations allowed by the packet.*
6. *K is the blocker row excluding closure reads outside the generated anchor subcube.*
7. *H records componentwise* hsame *transport for anchor, boundary-tile, span, rank, trigger-algebra, blocker, provenance, and local name rows.*
8. *C records displayed* Cont *replay from a trigger request through $A, B, S, R, G, K, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, anchored in Definition 9462.1519.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for a host codon powerset, a global median algebra, an unbounded closure engine, or equality of biological interpretations outside the displayed trigger window.

Lean4 ✓ `BEDC.Derived.AnchorSubcubeTriggerUp.AnchorSubcubeTriggerTasteGate single carrier alignment`

Theorem 3651.2 (Anchor subcube trigger NameCert obligations)**.** *For every accepted* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *carrier* $T = (A, B, S, R, G, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$ *surface consists exactly of anchor admission, boundary-tile admission, subcube-span calculation, coverage-rank readback, trigger-algebra replay, blocker non-escape, componentwise transport, displayed continuation replay, provenance, and local naming. The finite route is*

$$A, B \longmapsto S \longmapsto R \longmapsto G, K \longmapsto H, C, P, N.$$

No obligation is discharged by a host median-closure oracle or by an unrecorded codon-space equality.

Proof. Project the accepted carrier. The rows $A, B, S, R, G, K, H, C, P, N$ are the complete packet, so anchor admission and boundary-tile admission are direct finite reads.

The span row S records the axis-aligned subcube generated by the anchor and the outside boundary observations. The rank row R then reads the displayed span dimension, so the trigger threshold is a certificate row rather than a host calculation over all codon subsets.

The trigger-algebra row G replays the finite median operations that generate the subcube, while the blocker row K prevents reads outside the displayed span. Transport through H and replay through C preserve this same route componentwise.

The provenance and local naming rows P, N source the packet in the anchor-subcube trigger theorem. Since no carrier coordinate names a global codon powerset or an ambient closure engine, the listed rows exhaust the local NameCert obligations. □

Theorem 3651.3 (Anchor subcube trigger replay boundary)**.** *Every consumer of an accepted* $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$ *packet that reads a trigger conclusion must first traverse the displayed route*

$$A, B, S, R, G, K.$$

The replay exposes the anchor-relative subcube span and its coverage rank before the trigger is public. It exports no closure claim for boundary tiles outside B*, no radial trigger not carried by* S*, and no codon equality outside the finite rows named by the packet.*

Proof. Begin with a trigger read. The anchor row A and boundary-tile row B fix the finite observation window before any span or rank claim is available.

Read the span and rank rows S and R. These rows are the only source of the generated subcube and its dimension, so a consumer cannot substitute an external radial or global closure calculation.

The trigger-algebra row G then replays the median operations licensed by the span, and the blocker row K records the boundary beyond which the packet refuses closure reads. The structural rows H, C, P, N transport, replay, source, and name the same route without adding further codon geometry. □

Closure status for $\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3651.1, Definition 3651.2, and Definition 3651.3 seed the finite anchor, boundary-tile, span, rank, trigger-algebra, blocker, transport, replay, provenance, and NameCert surface for anchor-subcube triggers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AnchorSubcubeTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host codon powersets, global median algebra, radial trigger claims outside the displayed span, biological equality outside the packet, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked anchor admission, boundary-tile admission, span calculation, rank readback, trigger-algebra replay, blocker non-escape, transport, replay, provenance, and local naming for BEDC.Derived.AnchorSubcubeTrigger

Constructive story: An AnchorSubcubeTrigger$^{\uparrow}$ packet is generated from a displayed anchor row, boundary-tile row, anchor-relative span row, coverage-rank row, trigger-algebra row, blocker row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Chapter 3652

A Concrete Naming Certificate for $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$

Following Fourier's heat-equation analysis and the Courant–Friedrichs–Lewy finite-difference tradition, the $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ packet records only the finite constructive stencil interface visible to BEDC consumers. It sits after the partial-differential-equation row of chapter 194, the derivative row of chapter 122, the integral row of chapter 123, the transform handoff of chapter 3649, the real seal of chapter 38, the regular rational readback of chapter 301, the stream-window discipline of chapter 126, and the dyadic tolerance core of chapter 291. The chapter records heat-equation finite-stencil data as a local NameCert surface; it does not assert analytic solution existence, global PDE well-posedness, host function spaces, measure theory, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3652.1 (Heat-equation finite-stencil carrier)**.** *A* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, T, X, B, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite source-function window row, read through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *observations of the finitely inspected source values.*
2. *T is the time-step row containing the displayed dyadic time width, time-index schedule, and local tolerance budget.*
3. *X is the spatial stencil row containing the displayed dyadic mesh width, nearest-neighbour or finite-radius stencil weights, and inspected spatial index schedule.*
4. *B is the boundary row containing only the boundary values and boundary classifier reads requested by the finite stencil.*
5. *Q is the finite replay row that records how source, time-step, spatial-stencil, and boundary reads produce the next displayed heat row.*
6. *R is the real-seal row reached through* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readbacks for the finitely inspected stencil values.*
7. *H records componentwise* hsame *transport for F, T, X, B, Q, R, P, N and for the local certificate rows.*

8. *C records displayed* Cont *replay from the source-function windows through time-step, spatial-stencil, boundary, and real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, T, X, B, Q, R, H, C, P, N$. It has no coordinate for analytic solution existence, global parabolic well-posedness, host function-space equality, measure-theory integration, infinite mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeatEquationFiniteStencilUp.HeatEquationFiniteStencilTasteGate single carrier alignment`

Theorem 3652.2 (Heat-equation finite-stencil NameCert obligations)**.** *For an accepted* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *carrier $E = (F, T, X, B, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window admission, time-step exactness, spatial-stencil exactness, boundary-row exactness, finite replay, real-seal handoff, componentwise transport, displayed continuation replay, provenance, and local naming. These obligations are discharged along the finite route*

$$F, T, X, B \longmapsto Q \longmapsto R$$

together with H, C, P, N.

Proof. Carrier admission is projection from Definition 3652.1. The projection exposes the source-function windows, time-step row, spatial-stencil row, boundary row, finite replay row, real-seal row, transport rows, replay rows, provenance row, and local name.

The source-window, time-step, spatial-stencil, and boundary obligations are the rows F, T, X, B. They record only finite stream positions, regular rational readbacks, dyadic mesh and time budgets, and boundary reads already displayed in the packet.

The finite replay obligation is Q. It consumes F, T, X, B before the real seal R is read, so the heat row is a displayed stencil calculation rather than a host PDE solution.

Finally H transports the same rows componentwise, C replays the same route, and P, N source and name the packet. Since the carrier contains no row for analytic existence, global well-posedness, host function spaces, measure theory, infinite mesh limits, or ambient real completeness, the listed rows exhaust the local certificate surface. □

Theorem 3652.3 (Heat-equation finite-stencil Laplace-PDE handoff)**.** *Every accepted* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *handoff to a* $\mathsf{Pde}^{\uparrow}$ *or* $\mathsf{LaplaceTransform}^{\uparrow}$ *consumer passes first through the displayed finite stencil route*

$$F, T, X, B \longmapsto Q \longmapsto R.$$

The $\mathsf{Pde}^{\uparrow}$ *read receives only the derivative-relation and boundary rows visible through the stencil packet, while the* $\mathsf{LaplaceTransform}^{\uparrow}$ *read receives only the finite transform-window handoff after R has been displayed. Neither consumer obtains analytic solution existence, global heat-kernel semantics, selected improper limits, host function-space equality, measure theory, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $E = (F, T, X, B, Q, R, H, C, P, N)$. The PDE-facing data are not independent host equations; they are the derivative-relation, time-step, spatial-stencil, and boundary reads already present in T, X, B, Q.

For a $\mathsf{Pde}^{\uparrow}$ consumer, replay C from the source windows through the time and spatial rows into Q. The consumer can inspect only the finite relation and boundary rows that the replay exposes, transported by H and sourced by P, N.

For a $\mathsf{LaplaceTransform}^{\uparrow}$ consumer, the transform-window handoff is available only after the real-seal row R has been reached through the same route. The read therefore consumes finite stream, regular rational, and dyadic rows before it reaches any transform output.

The excluded analytic objects are not fields of the carrier. The handoff is therefore a finite NameCert route from stencil data to PDE and Laplace interfaces, not a claim of global heat-equation solvability or host analytic semantics. □

Falsifiable prediction: An accepted $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ consumer that reaches a $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed $\mathsf{StreamName}^{\uparrow}$ source windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ mesh and tolerance rows, finite stencil replay, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: pde, derivative, integral, laplacetransform, real, regseqrat, streamname, dyadicratcore: no listed sibling supplies the whole heat finite-stencil packet $E = (F, T, X, B, Q, R, H, C, P, N)$, because each sibling contributes only a PDE relation, derivative read, integral window, transform handoff, real seal, regular rational readback, stream schedule, or dyadic tolerance ledger while this chapter binds them into one finite replay and local naming route.

Closure status for $\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}$.

Constructive story: A $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ packet is generated from finite source-function windows, a displayed time-step row, a displayed spatial-stencil row, boundary rows, finite replay rows, a Real-seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3652.1, Definition 3652.2, and Definition 3652.3 seed the finite constructive heat-equation stencil boundary over $\mathsf{Pde}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{LaplaceTransform}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeatEquationFiniteStencil}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim analytic solution existence, global PDE well-posedness, host function spaces, measure theory, infinite mesh convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and Laplace-PDE handoff.

Chapter 3653

A Concrete Naming Certificate for $\mathsf{HausdorffDistance}^{\uparrow}$

Following the Hausdorff–Kuratowski compact-set metric reading, the $\mathsf{HausdorffDistance}^{\uparrow}$ packet records the scalar finite-distance certificate between two displayed nonempty compact subsets of one compact metric source. It sits next to the host hyperspace route of chapter 2086, consumes the compact-metric finite-net surface of chapter 129, and seals the resulting value through the real-name interface of chapter 38. The packet names the distance value as a finite BEDC carrier rather than treating $\mathsf{HausdorffMetric}^{\uparrow}$ itself as the only available consumer object.

Definition 3653.1 (Hausdorff distance finite carrier)**.** *A* $\mathsf{HausdorffDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, H, N_X, N_Y, D, E, R, T, C, P, L)$$

with the following displayed rows:

1. *A and B are the two nonempty compact-subset rows over the same compact metric source.*
2. *H is the* $\mathsf{HausdorffMetric}^{\uparrow}$ *ledger row that records the finite directed comparisons already admitted by the neighbouring Hausdorff-metric certificate.*
3. *N_X and N_Y are finite net rows for the two compact-subset names, read from the* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *sources.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row controlling the displayed mesh, directed bound, and symmetric slack.*
5. *E is the symmetric bound-exactness row, recording that the two directed finite-net estimates determine the same scalar tolerance window.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *distance seal reached only after the ledger, nets, and dyadic tolerance rows have been displayed.*
7. *T records componentwise* hsame *transport for $A, B, H, N_X, N_Y, D, E, R, C, P, L$.*
8. *C records displayed* Cont *replay of the route from compact-subset rows through finite nets, symmetric bounds, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, H, N_X, N_Y, D, E, R, T, C, P, L$. It has no coordinate for a host hyperspace point, quotient subset equality, selected nearest point, open-cover compactness, or ambient supremum formation.

Lean4 ✓ `BEDC.Derived.HausdorffDistanceUp.HausdorffDistanceTasteGate single carrier alignment`

Theorem 3653.2 (Hausdorff distance NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffDistance}^{\uparrow}$ *carrier*

$$Q = (A, B, H, N_X, N_Y, D, E, R, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the two compact-subset rows, HausdorffMetric ledger, finite nets, dyadic tolerance row, symmetric bound row, Real seal, transport, replay, provenance, and local naming row;*
2. *symmetric bound exactness, requiring the directed finite-net estimates in H and E to share the dyadic tolerance budget recorded by D;*
3. *finite-net witness stability, requiring N_X and N_Y to remain attached to the same* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{TotallyBounded}^{\uparrow}$ *sources under componentwise transport;*
4. *Real seal nonescape, requiring every scalar distance read to pass through R after H, N_X, N_Y, D, E have been displayed;*
5. *ledger nonescape, requiring C, P, L to replay, source, and name only the displayed finite route and to export no host hyperspace point, quotient subset equality, selected nearest point, open-cover compactness, or ambient supremum row.*

No scalar Hausdorff distance obligation is stored outside these rows.

Theorem 3653.3 (Hausdorff distance scalar handoff)**.** *Every Gromov–Hausdorff-facing, Vietoris-facing, or compact-subset consumer of an accepted* $\mathsf{HausdorffDistance}^{\uparrow}$ *packet reads the scalar value only through the finite route*

$$A, B, H, N_X, N_Y, D, E \longmapsto R \longmapsto T, C, P, L.$$

Thus the consumer receives a $\mathsf{Real}^{\uparrow}$ *distance seal for the two displayed compact-subset rows without receiving a host hyperspace metric object, quotient subset equality, selected nearest point, open-cover compactness theorem, or ambient supremum principle.*

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffDistance}^{\uparrow}$ packet is generated from two nonempty compact-subset rows, a HausdorffMetricUp ledger, finite CompactMetricUp and TotallyBoundedUp net rows, a DyadicRatCore tolerance row, symmetric bound exactness, a RealUp distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$, seed)

Scope closed: Definition 3653.1, Definition 3653.2, and Definition 3653.3 seed the finite scalar Hausdorff distance certificate consumed by compact-metric downstream routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HausdorffDistance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperspaces, quotient subset equality, selected nearest points, open-cover compactness, ambient supremum formation, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, symmetric bound exactness, finite-net witness stability, RealUp seal nonescape, and ledger nonescape for BEDC.Derived.HausdorffDistanceUp.

Chapter 3654

A Concrete Naming Certificate for $\mathsf{BoundedVariationRealFunction}^{\uparrow}$

Following Jordan's bounded variation theorem and Bishop–Bridges constructive analysis, the $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ packet records a real-valued function on a displayed real interval through finite partition windows and variation budgets. It sits between the real-valued sequence interface of chapter 3641, the finite partition carrier of chapter 1863, the stream-name observation route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the finite BEDC surface needed by later regulated-integration and Jordan-decomposition consumers without selecting a total variation supremum or importing ambient real completeness.

Definition 3654.1 (BoundedVariationRealFunction carrier)**.** *A* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, F, W, R, D, B, J, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval and partition window row supplied by* $\mathsf{DyadicPartition}^{\uparrow}$*, listing the finite dyadic subdivision on which variation is inspected.*
2. *F is the displayed real-valued function row, exposed only through* $\mathsf{RealSequenceSpace}^{\uparrow}$ *coordinate requests over the partition endpoints.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row scheduling the finite endpoint windows and adjacent-pair probes.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the values of F at the endpoints named by W.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *variation-budget ledger, recording absolute-difference budgets, partial variation sums, and the displayed bound attached to the current partition window.*
6. *B is the bounded-variation acceptance row, asserting that the inspected partition window is admitted only when its dyadic variation sum is witnessed below the displayed budget in D.*
7. *J is the Jordan-route row, naming the finite positive and negative variation ledgers available to a later Jordan-decomposition consumer without selecting a global decomposition.*
8. *H records componentwise* hsame *transport for the partition, function, observation, readback, variation-budget, bounded-variation, Jordan-route, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a bounded-variation request through I, F, W, R, D, B, J.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RealSequenceSpace}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedVariationRealFunctionUp.TasteGate.BoundedVariationRealFunctionTasteGate single carrier alignment`

Theorem 3654.2 (BoundedVariationRealFunction NameCert obligations)**.** *For every accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier $V = (I, F, W, R, D, B, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFun}}$ *surface consists exactly of finite* BHist *rows for carrier admission, partition-window exposure, real-value readback, variation-budget exactness, bounded-variation acceptance, Jordan-route availability, componentwise transport, replay, provenance, and local naming:*

1. *carrier admission requires the displayed* $\mathsf{DyadicPartition}^{\uparrow}$ *row,* $\mathsf{RealSequenceSpace}^{\uparrow}$ *function row,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budgets, bounded-variation row, Jordan-route row, transport, replay, provenance, and local name;*

2. *partition exposure requires every variation request to pass through the finite dyadic row I before any endpoint value is read;*

3. *value readback requires the endpoint values of F to be inspected only through W and R, and then sealed as* $\mathsf{Real}^{\uparrow}$ *data through the provenance row P;*

4. *variation-budget exactness requires the partial sums and displayed bound in D to mediate every acceptance read by B;*

5. *Jordan-route availability requires J to expose only finite positive and negative variation ledgers downstream of the same I, F, W, R, D, B route;*

6. *transport and replay stability require H, C, P, N to preserve and name the same finite surface.*

No obligation supplies total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3654.1. The packet exposes a dyadic partition window, a displayed real-valued function row, finite observation windows, regular rational readbacks, a dyadic variation ledger, a bounded-variation acceptance row, and a Jordan-route row before any structural replay is available.

The local obligations are therefore row-local. A variation request first enters the $\mathsf{DyadicPartition}^{\uparrow}$ row I; endpoint values of F are then read only through the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows W and R; and every absolute-difference or partial-sum comparison is mediated by the $\mathsf{DyadicRatCore}^{\uparrow}$ row D.

The bounded-variation and Jordan-facing rows are downstream of the same route. The acceptance row B can certify only the displayed partition window and budget, while J can expose only finite positive and negative variation ledgers built from that accepted data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host supremum, quotient function equality, selected decomposition, measure integration, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3654.3 (BoundedVariationRealFunction Jordan route)**.** *Every Jordan-facing consumer of an accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier* $V = (I, F, W, R, D, B, J, H, C, P, N)$ *reads the ordered finite route*

$$\mathsf{DyadicPartition}^{\uparrow} \longrightarrow \mathsf{RealSequenceSpace}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through $I, F, W, R, D, B, J, H, C, P, N$*. The consumer may use the finite positive and negative variation ledgers exposed by* J*, but it cannot extract total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Start with a Jordan-facing read of an accepted carrier. By Definition 3654.2, the read first passes through the finite $\mathsf{DyadicPartition}^{\uparrow}$ row and the displayed $\mathsf{RealSequenceSpace}^{\uparrow}$ function row, so every inspected endpoint value is tied to the same partition window.

The observation and readback rows then fix the value interface. The $\mathsf{StreamName}^{\uparrow}$ row schedules the endpoint probes, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readbacks, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row records the variation-budget comparisons used by the bounded-variation acceptance row.

The Jordan route J is downstream of that accepted data. It may expose finite positive and negative variation ledgers for the displayed window, and the structural rows H, C, P, N transport, replay, source, and name the route.

No additional carrier coordinate remains for a host supremum of all partitions, quotient equality of real-valued functions, a selected Jordan decomposition, measure integration, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence every Jordan-facing read factors through the displayed finite route. □

Theorem 3654.4 (BoundedVariationRealFunction partition-refinement induction)**.** *Let* $V = (I, F, W, R, D, B, J, H$ *be an accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier. Suppose that* I *displays a finite dyadic partition window and that a finite refinement spine is obtained by repeatedly inserting one dyadic endpoint into an already displayed adjacent interval. Then primitive induction over that spine preserves the variation-budget row: at every stage each inserted endpoint is read through the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *route* W, R*, the two adjacent absolute-difference budgets replacing the old interval budget are recorded in the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*, and the refined partial variation ledger is accepted by the same bounded-variation row* B*.*

Proof. Use induction on the finite refinement spine displayed over I. In the base case the spine has no inserted endpoint, so the current ledger is exactly the partition row already accepted by B through the budget comparisons recorded in D.

For the successor step, assume that the current displayed ledger is accepted by B and insert one dyadic endpoint inside a single adjacent interval. The endpoint observation is scheduled by the $\mathsf{StreamName}^{\uparrow}$ row W and read back by the $\mathsf{RegSeqRat}^{\uparrow}$ row R, so no value of F is inspected outside the carrier route.

The $\mathsf{DyadicRatCore}^{\uparrow}$ row D then replaces the old interval budget by the two adjacent absolute-difference budgets and updates the displayed partial variation sum. Since the induction hypothesis already identifies every unchanged interval with its accepted budget row, this replacement is the only local comparison required for the refined window.

Finally H, C, P, N transport, replay, source, and name the refined packet without changing the carrier rows. The acceptance row B therefore reads the refined ledger through the same finite packet, and the induction preserves bounded-variation acceptance along the whole refinement spine. □

Theorem 3654.5 (BoundedVariationRealFunction ledger handoff)**.** *Every accepted* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *carrier whose partition window is read as a* $\mathsf{BoundedVariation}^{\uparrow}$ *packet must expose*

$$I, F, W, R, D, B, J, H, C, P, N$$

first. The finite partition, endpoint, and dyadic-variation ledger recorded by I, W, R, D, B *is precisely the input consumed by chapter 398; the Jordan row* J *only routes the same accepted ledger to later positive and negative variation consumers.*

Proof. Start with the NameCert obligations of Definition 3654.2. They require carrier admission, partition-window exposure, value readback, variation-budget exactness, bounded-variation acceptance, and Jordan-route availability to be read through the same tuple $I, F, W, R, D, B, J, H, C, P, N$.

The Jordan route theorem Definition 3654.3 keeps J downstream of the accepted partition and variation-budget rows. Thus the positive and negative variation ledgers do not replace the bounded-variation input; they are read only after I, F, W, R, D, B have been displayed.

Finally apply the refinement induction of Definition 3654.4. It shows that endpoint insertions preserve the dyadic variation ledger through the same W, R, D, B route, so the reusable $\mathsf{BoundedVariation}^{\uparrow}$ chapter consumes exactly this finite ledger and no host supremum, quotient function equality, selected Jordan decomposition, or ambient completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ packet is generated from finite $\mathsf{DyadicPartition}^{\uparrow}$ windows, a displayed $\mathsf{RealSequenceSpace}^{\uparrow}$ function row, $\mathsf{StreamName}^{\uparrow}$ observations, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{DyadicRatCore}^{\uparrow}$ variation-budget ledger, a bounded-variation acceptance row, a finite Jordan-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3654.1, Definition 3654.2, Definition 3654.3, Definition 3654.4, and Definition 3654.5 seed the finite bounded-variation real-function surface over partition windows, real-valued sequence rows, finite observations, regular readbacks, dyadic variation budgets, refinement ledgers, bounded-variation handoff, and local Jordan-route ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedVariationRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim total variation as a host supremum, quotient equality of functions, a selected Jordan decomposition, measure integration, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-window exposure, real-value readback, variation-budget exactness, bounded-variation acceptance, finite Jordan-route availability, componentwise transport, replay, provenance, and local naming for the same displayed packet.

Chapter 3655

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ packet records the finite modulus evidence that turns regular Cauchy rational windows into located real-name reads. It sits between the dyadic arithmetic substrate of chapter 291, the stream observation surface of chapter 126, the regular rational sequence packet of chapter 301, the located comparison rows of chapter 1730, and the sealed real-name handoff of chapter 38.

3655.1 Located modulus carrier

Definition 3655.1 (Regular Cauchy located modulus carrier)**.** *A* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *carrier is a finite packet* $(D, S, Q, L, U, M, H, C, P, N)$ *with the following displayed rows:*

1. D*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger whose positive rows name the dyadic error budgets used by the modulus;*
2. S*, a* $\mathsf{StreamName}^{\uparrow}$ *observation window for the rational name under inspection;*
3. Q*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular Cauchy rational values visible on* S*;*
4. L *and* U*, lower and upper located comparison rows tied to the same finite window and the same dyadic precision ledger;*
5. M*, a finite modulus row assigning to each displayed dyadic precision a window depth after which the* Q*-values stay inside the* L, U *located bracket;*
6. H*, the componentwise* hsame *transport rows for the dyadic ledger, stream window, regular readback, located comparison rows, and modulus row;*
7. C*, the displayed* Cont *replay routes preserving the finite window and modulus reads under transport;*
8. P*, the* Pkg *provenance rows for the dyadic ledger, stream window, regular readback, located comparison evidence, and modulus ledger;*
9. N*, the local* $\mathsf{NameCert}$ *row binding these nine rows as one located modulus certificate.*

The carrier contains no quotient equality of Cauchy names, no selected limit point, no host real order predicate, and no ambient completeness principle.

Lean4 ✓ BEDC.Derived.RegularCauchyLocatedModulusUp.RegularCauchyLocatedModulusTasteGate single carrier alignment

Theorem 3655.2 (Regular Cauchy located modulus NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedM}}$ *surface for Definition 3655.1 consists of six finite obligations:*

1. *carrier admission: the rows* $(D, S, Q, L, U, M, H, C, P, N)$ *occur in the same finite* BHist *packet;*
2. *window alignment:* S *is the only observation window read by* Q*, by the located comparison rows* L, U*, and by the modulus row* M*;*
3. *dyadic precision exactness: every modulus query in* M *consumes a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget displayed in* D*;*
4. *regular readback confinement: all rational values inspected by* M *come through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *and not through an external sequence selector;*
5. *located bracket handoff: the lower and upper comparison rows* L, U *are the only located evidence exported toward* $\mathsf{LocatedRealComparison}^{\uparrow}$*;*
6. *transport stability:* hsame, Cont, Pkg*, and* NameCert *move the dyadic ledger, window, readback, located brackets, and modulus row componentwise.*

These obligations certify a finite located-modulus route for regular Cauchy names. They do not certify quotient equality of names, chosen limits, total order, or real completeness.

Proof. Project the carrier of Definition 3655.1. The projection exposes the dyadic precision ledger D, the stream window S, the regular readback Q, the located comparison rows L, U, and the modulus row M.

Window alignment and dyadic precision exactness are read directly from those rows. Since M has no coordinate except the displayed D, S, Q, L, U data, each precision query is evaluated against the same finite observation window and the same regular Cauchy readback.

The located bracket handoff is componentwise. The L, U rows are already the displayed $\mathsf{LocatedRealComparison}^{\uparrow}$ evidence, and the transport rows H, C replay them together with D, S, Q, M.

Finally P and N record the provenance and local certificate for exactly these rows. No extra coordinate can supply quotient stream equality, a selected limit, a host real order predicate, or ambient completeness. □

Theorem 3655.3 (Regular Cauchy located modulus Real-seal handoff)**.** *Every accepted* RegularCauchyLocatedMod *packet exports toward* $\mathsf{Real}^{\uparrow}$ *only the sealed read obtained from its* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q*, finite* $\mathsf{StreamName}^{\uparrow}$ *window* S*, positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger* D*, located comparison rows* L, U*, and modulus row* M*. The handoff preserves the same* $D, S, Q, L, U, M, H, C, P, N$ *carrier rows and supplies no quotient equality of Cauchy names, selected limit point, trichotomy, host real order predicate, or ambient completeness principle.*

Proof. Start with an accepted carrier and read the regular Cauchy name through Q on the displayed window S. The dyadic budgets used to test locatedness are the positive rows of D, and the lower and upper comparison evidence is precisely L, U.

The modulus row M turns each displayed precision into a finite window-depth claim over those same rows. Thus the Real-facing read has the data needed for a sealed $\mathsf{Real}^{\uparrow}$ handoff without selecting a representative limit.

Transport is confined to H and replayed by C. These rows preserve the dyadic ledger, stream window, regular readback, located brackets, and modulus ledger componentwise, while P, N record provenance and local naming.

The carrier has no quotient, trichotomy, host order, selected limit, or completeness coordinate. Hence the Real-seal consumer receives exactly the finite located-modulus evidence stated above. □

Theorem 3655.4 (Regular Cauchy located modulus finite-window nonescape)**.** *Every* $\mathsf{LocatedRealComparison}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *packet must pass through the displayed dyadic ledger* D, $\mathsf{StreamName}^{\uparrow}$ *window* S, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q, *lower and upper located comparison rows* L, U, *modulus row* M, *transport rows* H, *replay rows* C, *provenance rows* P, *and local* $\mathsf{NameCert}$ *row* N. *A strengthened finite-window read can refine these rows only by reading the same carrier coordinates; it cannot bypass them by using quotient Cauchy equality, selected limits, trichotomy, host order, or ambient completeness.*

Proof. Project the carrier of Definition 3655.1. The projection exposes exactly $D, S, Q, L, U, M, H, C, P, N$ as the finite rows available to a consumer.

Apply Definition 3655.2. Window alignment, dyadic precision exactness, regular readback confinement, located bracket handoff, and transport stability force each accepted read through the displayed ledger, window, readback, brackets, modulus, transport, replay, provenance, and local naming rows.

Apply Definition 3655.3 for the Real-facing case. The same $D, S, Q, L, U, M, H, C, P, N$ route is preserved at the seal, so a strengthened finite-window read remains a refinement of the displayed packet route.

The refused objects are absent carrier coordinates. Therefore quotient Cauchy equality, selected limits, trichotomy, host order, and ambient completeness cannot be used to bypass the finite-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ packet is generated from a dyadic precision ledger, a StreamName observation window, a RegSeqRat readback, located comparison brackets, a finite modulus row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3655.1, Definition 3655.2, and Definition 3655.3 name the seed carrier, finite NameCert obligation surface, and Real-seal handoff for located moduli of regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy names, selected limits, trichotomy, host real order, ambient completeness, or a standard bridge.

Upgrade path: To reach obligationClosure, the carrier, finite modulus obligations, and Real-seal handoff must be checked as BEDC packet obligations.

Chapter 3656

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ packet records the L10 completion reflector as one finite NameCert surface. It reads the reflector rows of chapter 1585, the functor rows of chapter 423, the extension-and-uniqueness rows of chapter 397, and the reflective-inclusion rows of chapter 1793. The chapter names the reflector boundary consumed by completion arguments; it does not assert quotient completions, choose limit representatives, identify host completions, import countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3656.1 (CauchyCompletionUniversalReflector carrier)**.** *A* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, R, F, E, J, W, G, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and name-data row, including the displayed metric classifier, finite source observations,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *reads, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger available before completion is consumed.*
2. *R is the reflector row read through* $\mathsf{CauchyCompletionReflector}^{\uparrow}$*, naming the source-to-completion admission and completed-object surface without exposing a quotient carrier or selected host limit.*
3. *F is the functor row read through* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$*, recording the finite action on displayed completion-facing source maps and continuation schedules.*
4. *E is the universal-extension row read through* $\mathsf{CompletionUniversalProperty}^{\uparrow}$*, carrying extension existence, extension uniqueness, and finite-window consumer factorization after R and F have been displayed.*
5. *J is the reflective-inclusion row read through* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$*, exposing completed objects only through the public reflective boundary.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the source and completed-name reads.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row used by the source and completion-facing sequences.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row used by S, W, G, R, F, E, J.*

9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *name row reached only after the stream, regular-rational, dyadic, reflector, functor, universal-extension, and reflective-inclusion rows have been displayed.*

10. *H records componentwise* hsame *transport for the source metric/name data, reflector row, functor row, universal-extension row, reflective-inclusion row, StreamName row, RegSeqRat row, DyadicRatCore row, Real row, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from source observations through $W, G, D, R, F, E, J, Q, H$.*

12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$ *row.*

The carrier exposes only this finite reflector surface. It has no coordinate for a quotient completion, chosen limit, host-level completion equality, countable-choice schedule, ambient completeness theorem, or full completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.CauchyCompletionUniversalReflectorUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorTasteGate`
`single carrier alignment`

Theorem 3656.2 (CauchyCompletionUniversalReflector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier $U = (S, R, F, E, J, W, G, D, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, reflector, functor, universal-extension, reflective-inclusion, stream-window, regular-rational, dyadic-tolerance, real-name, transport, replay, provenance, and local naming rows;*

2. *reflector admission, requiring each completion-facing object read to pass through R as in chapter 1585;*

3. *functorial action, requiring each source-map continuation to pass through F as in chapter 423;*

4. *universal-extension factorization, requiring each finite-window consumer to pass through E as in chapter 397;*

5. *reflective-inclusion exposure, requiring completed-object access to pass through J as in chapter 1793;*

6. *local StreamName, RegSeqRat, DyadicRatCore, and Real discipline, requiring W, G, D, Q to be read as the only name and tolerance rows;*

7. *ledger nonescape, requiring P and N to export no quotient completion, selected limit, host completion equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3656.1. The packet displays the source row, reflector row, functor row, universal-extension row, reflective-inclusion row, local StreamName-RegSeqRat-DyadicRatCore-Real rows, transport, replay, provenance, and local name.

The reflector and functor obligations are the first completion-facing projections. The row R is read through chapter 1585, and the row F is read through chapter 423; no consumer reaches completion data before those rows are visible.

The universal-extension and reflective-inclusion obligations follow the same displayed route. The row E cites chapter 397, while J cites chapter 1793; both are projected from the same finite packet.

Finally W, G, D, Q, H, C, P, N transport, replay, package, and name only the displayed rows. Since quotient completions, selected limits, host completion equality, countable choice, ambient completeness, and full $\mathsf{Real}^{\uparrow}$ completeness are absent as coordinates, no NameCert obligation can export them. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorNamecertOb`

Theorem 3656.3 (CauchyCompletionUniversalReflector extension factorization)**.** *Let $U = (S, R, F, E, J, W, G, D,$ be an accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *carrier. Every accepted completion-extension read factors through the finite route*

$$S, W, G, D \longmapsto R \longmapsto F \longmapsto E \longmapsto J \longmapsto Q \longmapsto H, C, P, N.$$

Thus a finite-window source consumer can reach an extension row only after the source metric/name data, reflector row, functor row, universal-property row, reflective-inclusion row, and local StreamName-RegSeqRat-DyadicRatCore-Real rows have all been projected from the same carrier.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalReflectorUp.CauchyCompletionUniversalReflectorCarrier extension factorization`

Proof. Begin with an accepted completion-extension read and project the carrier of Definition 3656.1. The source metric/name data and the local W, G, D rows supply the only finite observations and tolerance accounting available to the read.

Apply the reflector and functor projections. The reflector row R is the public entry into completion-facing data, and the functor row F is the only displayed action row for source-map continuations.

Apply the universal-property projection from E. By chapter 397, the extension and uniqueness read is downstream of the displayed completion rows rather than an independent host function.

The reflective-inclusion row J and real-name row Q close the route. The structural rows H, C, P, N preserve exactly the same route, so the extension factorization has no coordinate from which to read quotient completions, selected limits, host completion equality, choice, or ambient completeness. □

Theorem 3656.4 (CauchyCompletionUniversalReflector nonescape)**.** *Every public L10 completion read exported by an accepted* $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ *packet is exhausted by the rows $S, R, F, E, J, W, G, D, Q, H, C, P, N$. In particular, the read cannot replace the reflector by* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *alone, the functor by* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *alone, the extension by* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *alone, the public completed-object boundary by* $\mathsf{CauchyCompletionReflectiveSubcateg}$ *alone, or the local name route by* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *alone.*

Proof. Project P first. It names the available sibling packets: $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctc}$ $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Apply Definition 3656.2. Each sibling is assigned to one row of the carrier, so a consumer that keeps only one sibling loses the displayed relation between source data, reflector entry, functor action, universal extension, reflective inclusion, and local name data.

Apply Definition 3656.3. The extension route is finite and ordered, and the structural rows H, C, P, N preserve that order. Therefore no public L10 completion read can leave the packet surface to obtain a quotient completion, selected limit, host equality, choice schedule, ambient completeness theorem, or full $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}$ packet is generated from source metric/name data, reflector rows, functor rows, universal-extension rows, reflective-inclusion rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Real name rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3656.1, Definition 3656.2, Definition 3656.3, and Definition 3656.4 seed the finite universal-reflector boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host completion equality, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector admission, functorial action, universal-extension factorization, reflective-inclusion exposure, local name-route discipline, and ledger nonescape for BEDC.Derived.CauchyCompletionUn

Chapter 3657

A Concrete Naming Certificate for $\mathsf{TriggerHypergraphReliability}^{\uparrow}$

The $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ packet realizes the trigger-hypergraph reliability surface of Definition 9462.1556 as finite BEDC NameCert data. It sits over the six-boundary-tile coverage route of chapter 3470, the window-coverage trigger algebra of chapter 3478, the cellular window trace of chapter 3481, the dyadic tolerance ladder of chapter 3525, and the trigger-hypergraph reliability theorem of Definition 9462.1564. The packet names accepted reliability reads as finite trigger-edge and budget rows; it is not a stochastic oracle, polynomial-ring equality, quotient codon semantics, infinite trigger search, or hidden probability model.

Definition 3657.1 (TriggerHypergraphReliability carrier)**.** *A* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, E, R, L, W, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the six-boundary-tile coverage row naming the finite boundary positions whose trigger behavior is read.*
2. *E is the finite trigger-edge list row, recording the accepted hyperedges and their coordinate supports.*
3. *R is the reliability-budget row, carrying the finite acceptance budget against which trigger-edge reads are evaluated.*
4. *L is the local tag ledger for present, absent, covered, uncovered, triggering, nontriggering, accepted, and rejected edge reads.*
5. *W is the cellular window-trace row assigning trigger-edge reads to the visible window route.*
6. *D is the dyadic-tolerance row used to compare reliability budgets without importing real-valued probability or host polynomial equality.*
7. *A is the algebra transport row through the window-coverage trigger algebra and the trigger-hypergraph source vision.*
8. *H records componentwise* hsame *transport for boundary tiles, trigger-edge lists, reliability budgets, local tags, window traces, dyadic tolerances, algebra transport, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from each accepted trigger-edge read through B, E, R, L, W, D, A, H.*

10. *P is the* Pkg *provenance over* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$*, the six-boundary-tile coverage route, the window-coverage trigger algebra, the cellular window trace, the dyadic tolerance ladder, the trigger-hypergraph source vision,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, E, R, L, W, D, A, H, C, P, N$. It has no coordinate for host probability spaces, ambient polynomial equality, quotient codon equality, infinite trigger search, stochastic evolutionary dynamics, or a hidden reliability oracle.

Lean4 (statement) `BEDC.Derived.TriggerHypergraphReliabilityUp`

Theorem 3657.2 (TriggerHypergraphReliability NameCert obligations)**.** *For every accepted* TriggerHypergraphRel *carrier $T = (B, E, R, L, W, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$ *surface consists exactly of boundary-coverage admission, trigger-edge list exposure, reliability-budget exposure, finite tag accounting, window-trace assignment, dyadic-tolerance comparison, algebra transport, componentwise transport, displayed replay, provenance, and local naming.*

The reliability row R is accepted only when its read is paired with one finite trigger-edge list E and one finite budget row over the displayed B, L, W, D, A route. Any consumer that reads reliability without those rows, or replaces them by host probability spaces, ambient polynomial equality, quotient codon semantics, stochastic dynamics, infinite search, or a hidden reliability oracle, leaves the accepted packet surface.

Proof. Project the carrier of Definition 3657.1. The rows B, E, R, L, W, D, A expose the boundary coverage, finite edge list, reliability budget, finite tag ledger, window trace, dyadic tolerance row, and algebra transport.

The reliability-budget row is finite. It is read only together with the trigger-edge list E, while L, W, D record finite tags, cellular placement, and dyadic comparison data before any reliability decision is accepted.

Finally H, C, P, N preserve the same displayed route. Since no host probability space, polynomial equality, quotient codon equality, stochastic model, infinite search, or hidden reliability oracle is a carrier coordinate, the local NameCert obligations are exactly the listed finite rows. □

Theorem 3657.3 (TriggerHypergraphReliability factorization)**.** *Every accepted reliability read for* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *factors through one finite trigger-edge list and one reliability-budget row:*

$$B \longmapsto E \longmapsto R \longmapsto L, W, D, A \longmapsto H, C, P, N.$$

Thus accepted reliability is a finite NameCert route through boundary coverage, trigger-edge exposure, budget accounting, window tracing, dyadic tolerance comparison, algebra transport, replay, provenance, and local naming.

Proof. Start from an accepted reliability read and project the carrier rows. The boundary row B supplies the only visible coverage data, and the trigger-edge list E supplies the only finite hypergraph edge surface.

Read the budget row R next. The row is paired with E, so reliability is not evaluated against an ambient probability space or an unbounded family of triggers.

The rows L, W, D, A then assign finite tags, cellular window placement, dyadic budget comparison, and algebra transport. The structural rows H, C, P, N preserve exactly this route, so the factorization has no coordinate outside the finite packet. □

Lean4 ✓ `BEDC.Derived.TriggerHypergraphReliabilityUp.TriggerHypergraphReliabilityFactorization`

Theorem 3657.4 (TriggerHypergraphReliability nonescape)**.** *Every public reliability read exported by an accepted* $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ *packet is exhausted by* $B, E, R, L, W, D, A, H, C, P, N$*. In particular, a consumer cannot replace the trigger-edge list by the six-boundary-tile coverage route alone, the reliability budget by the cellular window trace alone, the dyadic comparison by a host real-probability model, or the algebra transport by polynomial-ring equality.*

Proof. Project the provenance row P. It names the six-boundary-tile coverage route, the window-coverage trigger algebra, the cellular window trace, the dyadic tolerance ladder, and the source trigger-hypergraph vision.

Apply Definition 3657.2. Each dependency is assigned to one displayed carrier row, so no sibling route alone contains the trigger-edge list, reliability budget, tag ledger, window trace, dyadic comparison, algebra transport, replay, provenance, and local name.

Apply Definition 3657.3. The reliability read is ordered through the finite edge list and budget row before any downstream tag, trace, tolerance, or algebra transport is consumed. Therefore the packet cannot export host probability spaces, polynomial equality, quotient codon equality, stochastic dynamics, infinite trigger search, or a hidden reliability oracle. □

Falsifiable prediction: Every accepted reliability read factors through one finite trigger-edge list and one reliability-budget row.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, cellular-window-trace, dyadic-tolerance-ladder, trigger-hypergraph vision source: none is bijective with $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ because the packet jointly carries boundary coverage, trigger-edge lists, reliability budgets, finite tag accounting, cellular windows, dyadic tolerance comparison, algebra transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.TriggerHypergraphReliabilityUp.TasteGate` supplies explicit $\mathsf{Nontrivial}\ \mathsf{TriggerHypergraphReliability}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{TriggerHypergraphReliability}^{\uparrow}$ instances preserving $B, E, R, L, W, D, A, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TriggerHypergraphReliability}^{\uparrow}$ packet is generated from boundary-coverage rows, trigger-edge list rows, reliability-budget rows, finite tag ledgers, cellular window traces, dyadic tolerance rows, algebra transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3657.1, Definition 3657.2, Definition 3657.3, and Definition 3657.4 seed the finite trigger-edge reliability NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriggerHypergraphReliability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host probability spaces, ambient polynomial equality, quotient codon equality, stochastic biological dynamics, infinite trigger search, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-edge exposure, reliability-budget exposure, finite tag accounting, window-trace assignment, dyadic-tolerance comparison, algebra transport, Nontrivial TriggerHypergraphReliabilityUp, and FieldFaithful TriggerHypergraphReliabilityUp for BEDC.Derived.TriggerHypergraphReliabilityUp.

Chapter 3658

A Concrete Naming Certificate for $\mathsf{LocatedCompleteInterval}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompleteInterval}^{\uparrow}$ packet records a located complete closed interval as finite BEDC NameCert data. It refines the closed bounded interval carrier of chapter 1591, the located real interval surface of chapter 1709, and the nested interval Cauchy route of chapter 3599. Its completion-facing reads pass through the regular rational sequence packet of chapter 301, the terminal real seal of chapter 38, and the dyadic tolerance substrate of chapter 291. The packet names finite interval completeness evidence; it is not a host completeness axiom, quotient construction, trichotomy oracle, selected limit, or choice principle.

3658.1 Located Complete Interval Carrier

Definition 3658.1 (Located complete interval carrier)**.** *A* $\mathsf{LocatedCompleteInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, L, N, Q, D, S, R, E, K, H, C, P, M)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row that answers finite membership, containment, separation, and refinement requests through endpoint-radius locatedness data.*
3. *N is the* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *row carrying a descending bracket family, diameter budget, and finite Cauchy extraction route.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational centers and endpoint witnesses inspected by the bracket family.*
5. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to measure bracket diameter, endpoint slack, and refinement depth.*
6. *S is the finite observation schedule for bracket requests and Cauchy reads.*
7. *R is the completion-read row that sends each nested bracket request to a sealed* $\mathsf{Real}^{\uparrow}$ *output through B, L, N, Q, D, S.*

8. *E records the interval-membership evidence for the sealed output, again read only through the closed bounded and located interval rows.*
9. *K is the Cauchy-bracket induction ledger, recording the base bracket, successor refinement, diameter decrease, and endpoint containment rows for the displayed finite family.*
10. *H records componentwise* hsame *transport for all rows above.*
11. *C records displayed* Cont *replay from bracket refinement requests to the same completion-read and membership rows.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedCompleteInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{NestedIntervalCauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *M is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$ *row binding the packet.*

The carrier contains no host closed interval object, no quotient equality of real names, no selected representative stream, no trichotomy row, no countable choice schedule, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCompleteIntervalUp.LocatedCompleteIntervalTasteGate single carrier alignment`

Theorem 3658.2 (Located complete interval NameCert obligations)**.** *For the carrier of Definition 3658.1, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for $I = (B, L, N, Q, D, S, R, E, K, H, C, P, M)$ as one* BHist *packet;*
2. *source-interval exactness, requiring endpoint, containment, and Real-seal reads to pass through the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row B;*
3. *located-boundary exactness, requiring membership and separation reads to pass through the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row L;*
4. *nested-bracket exactness, requiring each Cauchy bracket read to pass through the* $\mathsf{NestedIntervalCauchy}^{\uparrow}$ *row N and the induction ledger K;*
5. *rational and dyadic budget exactness, requiring all centers, endpoint witnesses, diameters, and tolerances to be read through Q and D;*
6. *completion-read confinement, requiring the sealed* $\mathsf{Real}^{\uparrow}$ *output and its interval-membership evidence to be exported only through R and E;*
7. *transport and replay stability through H, C, P, M.*

These obligations name located interval completeness as a finite certificate surface. They do not assert host completeness, quotient construction, trichotomy, selected limits, countable choice, compactness beyond the displayed bracket route, or a public standard bridge.

Proof. Project the carrier of Definition 3658.1. The projection exposes exactly the closed bounded source row, located interval row, nested Cauchy bracket row, regular rational readback, dyadic tolerance ledger, schedule, completion read, membership evidence, induction ledger, and structural rows.

Source-interval and located-boundary exactness are immediate from the rows B and L. Every endpoint, containment, membership, separation, and refinement read is already routed through those two interval surfaces before a completion-facing consumer can inspect R or E.

Nested-bracket exactness is supplied by N and K. The bracket family, diameter decrease, and endpoint containment facts are displayed as finite rows, and the rational and dyadic budgets used by those rows are exactly Q and D.

Finally the structural rows close the obligation surface. The hsame transports, Cont replay routes, provenance, and local naming row preserve the listed packet without adding a host completeness coordinate, quotient, trichotomy decision, selected limit, or choice schedule. □

Theorem 3658.3 (Located complete interval Cauchy bracket induction)**.** *Let $I = (B, L, N, Q, D, S, R, E, K, H, C,$ be an accepted* $\mathsf{LocatedCompleteInterval}^{\uparrow}$ *carrier. Every nested-bracket Cauchy read exported by I is obtained by induction over the displayed bracket ledger K: the base step reads the source interval through B and L; the successor step refines the current bracket through N, Q, D, and S; and the terminal read exports a sealed* $\mathsf{Real}^{\uparrow}$ *row through R together with interval-membership evidence through E. The induction is finite-window BEDC data, not an appeal to ambient completeness or choice.*

Proof. Begin with the base bracket. The closed bounded source row B supplies the endpoint and containment packet, while the located interval row L supplies the finite membership and separation service for the same endpoints.

For the successor step, read the nested interval row N. Each refinement consumes a rational center or endpoint witness from Q, a positive dyadic tolerance from D, and an observation window from S. The ledger K records that the next bracket remains inside the previous bracket and has the displayed smaller diameter budget.

Iterating this finite ledger yields the Cauchy route named by the packet. The completion-read row R seals the resulting read as a $\mathsf{Real}^{\uparrow}$ output only after the bracket data have passed through B, L, N, Q, D, S, K.

The membership row E then reads the sealed output back into the same interval carrier. Since the construction has used only the displayed bracket induction rows and the structural transports H, C, P, M, no host completeness principle, quotient equality, selected representative, trichotomy, or choice schedule enters the export. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompleteInterval}^{\uparrow}$ packet is generated from a closed bounded interval source row, a located real interval row, a nested interval Cauchy row, regular rational readback, dyadic tolerance data, finite schedules, completion-read and membership rows, a bracket induction ledger, componentwise transport, replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3658.1, Definition 3658.2, and Definition 3658.3 name the seed carrier, finite NameCert obligation surface, and nested-bracket induction route for located complete closed intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompleteInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, quotient construction, trichotomy, selected limits, countable choice, compactness beyond the displayed nested-bracket route, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-interval exactness, located-boundary exactness, nested-bracket exactness, rational and dyadic budget exactness, completion-read confinement, and bracket induction for the same finite packet.

Chapter 3659

A Concrete Naming Certificate for $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ packet records the closed-interval Cauchy completeness route as finite BEDC NameCert data. It reads the endpoint and containment rows of chapter 1591, the regular rational sequence route of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the located interval boundary of chapter 3658. The packet names the interval-contained Cauchy readback surface; it does not assert open-cover compactness, selected subsequences, quotient equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

3659.1 Interval Cauchy Completeness Carrier

Definition 3659.1 (Interval Cauchy completeness carrier)**.** *An* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *interval row, exposing lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, and the local closed-interval NameCert surface.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy row for the inspected rational approximants, including the finite modulus windows used by the interval read.*
3. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare successive approximants, endpoint slack, and interval containment at the displayed scale.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite window row selecting exactly the stream observations consumed by the regular-Cauchy and interval-containment checks.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, Q, D, W have been replayed.*
6. *E records that the sealed read remains inside the same closed interval source row B through the located complete interval boundary.*
7. *H records componentwise* hsame *transport for endpoint rows, regular-Cauchy windows, dyadic tolerances, StreamName windows, the Real seal, interval-membership evidence, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite Cauchy-read request through W, Q, D, B and only then to R and E.*
9. *P is the* Pkg *provenance over* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCompleteInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$ *row.*

The accepted carrier contains no open-cover compactness coordinate, no selected subsequence, no quotient equality of real names, no trichotomy row, no countable-choice schedule, and no ambient completeness field for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.IntervalCauchyCompletenessUp.IntervalCauchyCompletenessTasteGate single carrier alignment`

Theorem 3659.2 (Interval Cauchy completeness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *carrier $I = (B, Q, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint-row exactness, regular-Cauchy window consumption, dyadic tolerance exactness, StreamName window confinement, Real seal handoff, interval-membership readback, componentwise transport, continuation replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 3659.1. The projection exposes the closed interval row B, the regular rational row Q, the dyadic ledger D, the finite stream window row W, the Real seal R, the interval-membership evidence E, and the structural rows H, C, P, N.

Endpoint-row exactness is supplied by B. Every endpoint, order, rational containment, and dyadic containment read is already a row of the closed bounded interval packet before the Cauchy consumer can inspect Q, D, W, R, or E.

Regular-Cauchy and tolerance exactness are supplied by Q, D, W. The finite stream window names the observations, the regular rational row names the Cauchy approximants and modulus windows, and the dyadic ledger names the positive tolerances used in the endpoint and successor comparisons.

Finally R and E close the seal route, while H, C, P, N preserve and name only the displayed packet. Since no row stores open-cover compactness, a selected subsequence, quotient equality, trichotomy, countable choice, or ambient Real completeness, none of those objects can enter the local NameCert surface. □

Theorem 3659.3 (Interval Cauchy completeness regular seal)**.** *Let $I = (B, Q, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *packet. Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by I first consumes the finite route*

$$B, W, Q, D \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the sealed real output is visible only after the closed-interval endpoint rows, StreamName observation windows, regular-Cauchy readback, and dyadic tolerance ledger have all been displayed.

Proof. Begin with a Real-facing consumer of I. Carrier admission exposes R, but only as the seal row downstream of B, W, Q, D.

Replay the continuation row C. The request first reads the closed interval row B, then the StreamName window W, the regular rational Cauchy row Q, and the dyadic tolerance ledger D.

Only after this finite replay can R be read as a $\mathsf{Real}^{\uparrow}$ seal. The interval-membership row E then reads the same sealed output back through the source interval, and the structural rows H, P, N transport, package, and name that route without adding any separate completeness principle. □

Theorem 3659.4 (Interval Cauchy completeness no open-cover route)**.** *Every consumer of an* $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ *packet receives a finite regular-Cauchy interval route through* $B, Q, D, W, R, E, H, C, P$ *The packet has no row from which an open-cover compactness proof, a selected subsequence, quotient equality for real names, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness can be recovered.*

Proof. Inspect the carrier of Definition 3659.1. Its non-structural rows are exactly the closed interval source, regular rational Cauchy readback, dyadic tolerance ledger, StreamName window, Real seal, and interval-membership readback.

Open-cover compactness and selected subsequences would require separate rows for cover indices, finite subcover choice, or subsequence selection. Quotient equality, trichotomy, countable choice, and ambient Real completeness would likewise require decision, selection, or host-completeness coordinates not present in the packet.

The only exported route is therefore the displayed finite Cauchy interval route. Componentwise transport, continuation replay, provenance, and local NameCert evidence preserve that route; they do not create any of the excluded coordinates. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntervalCauchyCompleteness}^{\uparrow}$ packet is generated from a closed bounded interval row, a RegSeqRat regular-Cauchy row, a DyadicRatCore tolerance ledger, StreamName finite windows, a RealUp seal, interval-membership readback through the located complete interval boundary, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3659.1, Definition 3659.2, Definition 3659.3, and Definition 3659.4 name the seed carrier, finite NameCert obligation surface, regular-Cauchy seal route, and excluded compactness route for closed-interval Cauchy completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalCauchyCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subsequences, quotient equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-row exactness, regular-Cauchy window consumption, dyadic tolerance exactness, StreamName window confinement, Real seal handoff, interval-membership readback, and structural stability for the same finite packet.

Chapter 3660

A Concrete Naming Certificate for $\mathsf{LimitPointCompact}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LimitPointCompact}^{\uparrow}$ packet records the sequence-facing compactness boundary: an infinite sequence in a displayed compactly bounded carrier has a named cluster-point route through a subsequence and a real-valued convergence witness. The packet sits beside the $\mathsf{SequentialCompact}^{\uparrow}$ chapter of chapter 2372, the $\mathsf{BolzanoWeierstrass}^{\uparrow}$ chapter of chapter 1254, the $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ chapter of chapter 1580, and the $\mathsf{Real}^{\uparrow}$ chapter of chapter 38. It deliberately names a limit-point route for sequences, not open-cover compactness, arbitrary choice, or ambient real-completeness.

Definition 3660.1 (LimitPointCompact carrier)**.** *A* $\mathsf{LimitPointCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, S, B, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compactly bounded carrier row whose points are read by the sequence packet;*
2. *S is the input sequence row, displaying a finite code for the sequence-facing object under consideration;*
3. *B is the boundedness and membership ledger, keeping every visible sequence value inside the carrier row K;*
4. *W is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *witness row that extracts a subsequence candidate from the bounded sequence data;*
5. *Q is the* $\mathsf{SequentialCompact}^{\uparrow}$ *subsequence route row, naming the selected subsequence only through displayed finite evidence;*
6. *E is the* $\mathsf{RealSequenceClusterPoint}^{\uparrow}$ *endpoint row, reading the subsequence as converging to a real cluster point through* $\mathsf{Real}^{\uparrow}$ *data;*
7. *H records componentwise* hsame *transport for K, S, B, W, Q, E and the structural rows;*
8. *C records displayed* Cont *replay from the sequence row through boundedness, subsequence extraction, endpoint reading, and local naming;*
9. *P is the* Pkg *provenance over the displayed compactness, subsequence, cluster-point, real, transport, replay, and naming rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^\uparrow}$ *row.*

The classifier accepts exactly packets whose visible coordinates are $K, S, B, W, Q, E, H, C, P, N$. It contains no open-cover subcover coordinate, no countable-choice selector, no ambient $\mathsf{Real}^\uparrow$ *completeness theorem, and no host-level selection of cluster points.*

Lean4 ✓ `BEDC.Derived.LimitPointCompactUp.LimitPointCompactTasteGate single carrier alignment`

Theorem 3660.2 (LimitPointCompact NameCert obligations)**.** *For the carrier of Definition 3660.1, the local* $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^\uparrow}$ *surface consists exactly of five finite obligations:*

1. *carrier admission: every accepted object exposes the displayed packet $L = (K, S, B, W, Q, E, H, C, P, N)$;*
2. *bounded sequence exposure: every sequence value consumed by the packet is read through S and certified inside K by the ledger row B;*
3. *subsequence extraction: every cluster-point route passes through the* $\mathsf{BolzanoWeierstrass}^\uparrow$ *witness row W and the* $\mathsf{SequentialCompact}^\uparrow$ *subsequence row Q;*
4. *endpoint exactness: the cluster endpoint is read only through the* $\mathsf{RealSequenceClusterPoint}^\uparrow$ *row E and its* $\mathsf{Real}^\uparrow$ *coordinate data;*
5. *ledger exhaustion: H, C, P, N preserve, replay, source, and name the displayed route without adding open-cover compactness, countable choice, a selected representative, or ambient real completeness.*

Consequently the certificate names the sequence-facing limit-point compactness route and no stronger compactness principle.

Proof. Carrier admission is projection from the displayed tuple $L = (K, S, B, W, Q, E, H, C, P, N)$. The only point carrier is K, the only input sequence row is S, and the only boundedness ledger is B.

The subsequence part is forced by the witness inventory. The row W is the Bolzano–Weierstrass extraction witness and Q is the sequential-compactness subsequence route; a consumer reaches endpoint data only after those rows have been read.

The endpoint part is then supplied by E. It is the cluster-point row and reads the relevant real-valued convergence data through the $\mathsf{Real}^\uparrow$ packet, so no separate ambient completeness theorem is imported.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no open-cover, choice, selected-representative, or ambient-completeness coordinate, those principles cannot be exported by the local certificate. □

Theorem 3660.3 (LimitPointCompact Bolzano–Weierstrass factorization)**.** *Every accepted* $\mathsf{LimitPointCompact}^\uparrow$ *limit-point consumer factors through*

$$K, S \longmapsto B \longmapsto W \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Here W is read from the $\mathsf{BolzanoWeierstrass}^\uparrow$ *surface of chapter 1254, Q is read from the* $\mathsf{SequentialCompact}^\uparrow$ *surface of chapter 2372, and E is read from the* $\mathsf{RealSequenceClusterPoint}^\uparrow$ *and* $\mathsf{Real}^\uparrow$ *surfaces of chapter 1580 and chapter 38. The factorization exports no open-cover compactness, no countable choice, no selected representative, and no ambient real-completeness claim.*

Proof. Start with Definition 3660.2. Its bounded sequence exposure clause places every sequence-facing read behind K, S, B.

Next read the extraction rows. The Bolzano–Weierstrass chapter supplies the finite witness row W, and the sequential compactness chapter supplies the subsequence route Q; neither row carries an open-cover subcover object or an unrestricted choice function.

The endpoint is then read through the real sequence cluster-point packet and the real packet. This is exactly the row E, so the conclusion is a displayed sequence endpoint rather than ambient completeness for all real Cauchy data.

The structural rows H, C, P, N preserve and name the same route. Because the carrier inventory contains no stronger compactness coordinate, every consumer remains inside the stated factorization. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3660.1, Definition 3660.2, and Definition 3660.3 seed the sequence-facing compactness route through bounded sequence data, Bolzano–Weierstrass extraction, sequential compactness, real cluster endpoints, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LimitPointCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary choice, selected cluster-point representatives, quotient completion equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded sequence exposure, Bolzano–Weierstrass factorization, sequential-subsequence exactness, real cluster-point endpoint exactness, nonescape ledgers, classifier transport, and provenance/name exhaustion for BEDC.Derived.LimitPointCompactUp.

Constructive story: A $\mathsf{LimitPointCompact}^{\uparrow}$ packet is generated from a compactly bounded carrier row, an input sequence row, a boundedness ledger, a Bolzano–Weierstrass witness row, a sequential-compactness subsequence row, a real sequence cluster-point endpoint row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3661

A Concrete Naming Certificate for $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$

The $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ packet follows the proof-theoretic normal-form consistency tradition for typed calculi: a closed term displayed at the internal false encoding cannot simultaneously be a beta-strong normal endpoint. In BEDC form, this is not a full closed-consistency claim. It is a finite witness package for the exact closed-normal boundary of chapter 9376, sitting beside the closed-normal canonicity packet of chapter 799, the endpoint ledger of chapter 837, and the Z-normal predicate packet of chapter 936.

Definition 3661.1 (ClosedNormalConsistencyWitness carrier)**.** *A* $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, F, L, B, E, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the MetaCIC term row, displaying the closed proof candidate under audit;*
2. *F is the internal false-encoding row, naming the target type at which the term is displayed;*
3. *L is the empty-context typing ledger for the judgement that T has type F;*
4. *B is the beta-strong endpoint row, recording that the displayed term is already at the normal-form boundary;*
5. *E is the closedness row, recording that the term has no external context dependency;*
6. *R is the false-exclusion witness row read from the closed-normal consistency theorem;*
7. *H records componentwise* hsame *transport for T, F, L, B, E, R and the structural rows;*
8. *C records displayed* Cont *replay from typing and closedness through beta-strong endpoint evidence to false exclusion;*
9. *P is the* Pkg *provenance over MetaCIC typing, closedness, beta-strong endpoint, false encoding, transport, replay, and naming rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible coordinates are $T, F, L, B, E, R, H, C, P, N$. It contains no subject-reduction transport, no full closed-consistency theorem, no Church–Rosser quotient, no normalization theorem for all closed terms, and no host proof of arbitrary false exclusion.

Lean4 ✓ BEDC.Derived.ClosedNormalConsistencyWitnessUp.ClosedNormalConsistencyWitnessTasteGate
single carrier alignment

Theorem 3661.2 (ClosedNormalConsistencyWitness NameCert obligations)**.** *For the carrier of Definition 3661.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission: every accepted object exposes the displayed packet* $W = (T, F, L, B, E, R, H, C, P, N)$*;*
2. *false typing exposure: the typing ledger* L *displays the empty-context judgement that the term row* T *has the false-encoding type row* F*;*
3. *normal endpoint exposure: the rows* B *and* E *display beta-strong endpoint evidence and closedness before false exclusion is read;*
4. *false-exclusion exactness: the row* R *is available only after* T, F, L, B, E *have been displayed together;*
5. *ledger exhaustion:* H, C, P, N *preserve, replay, source, and name that route without adding full closed consistency, subject reduction, Church–Rosser transport, or global normalization.*

Consequently the certificate names the closed-normal false-exclusion witness and no stronger consistency principle.

Proof. Carrier admission is projection from the displayed tuple. The only term row is T, the only false-encoding row is F, and the only typing ledger is L.

The normal endpoint is part of the input, not a theorem extracted from arbitrary closed typing. Rows B and E display beta-strong normality and closedness before the false-exclusion witness can be read.

The false-exclusion row R is therefore exactly the closed-normal consistency route. It consumes the typing, closedness, and beta-strong evidence together, matching the boundary stated in chapter 9376.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the packet has no subject-reduction, Church–Rosser, global-normalization, or full-consistency coordinate, none of those stronger principles can be exported by the certificate. □

Theorem 3661.3 (ClosedNormalConsistencyWitness false-exclusion factorization)**.** *Every accepted* $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ *consumer of false exclusion factors through*

$$T, F \longmapsto L \longmapsto E \longmapsto B \longmapsto R \longmapsto H, C, P, N.$$

The route reads the term and false-encoding rows, exposes empty-context typing, checks closedness, checks beta-strong endpoint evidence, and only then exports the false-exclusion witness. It does not infer full closed consistency, arbitrary subject-reduction transport, Church–Rosser conversion, quotient equality of proofs, or normalization for all closed terms.

Proof. Use Definition 3661.2. The false typing exposure clause places the term and false encoding behind T, F, L.

Next the normal endpoint exposure clause places the closedness and beta-strong rows before any contradiction is read. This mirrors the main-result boundary: closed and typed alone are not enough for the present packet.

The row R is then the only false-exclusion coordinate. It is reached after the typing, closedness, and beta-strong rows have all been displayed.

The structural rows H, C, P, N preserve and name this route. The missing stronger principles have no carrier coordinate, so the exported witness remains closed-normal rather than full closed consistency. □

Lean4 ✓ `BEDC.Derived.ClosedNormalConsistencyWitnessUp.ClosedNormalConsistencyWitnessFalseExclusionfactorization`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3661.1, Definition 3661.2, and Definition 3661.3 seed the closed-normal false-exclusion route through MetaCIC term display, false typing, closedness, beta-strong endpoint evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full closed consistency, subject reduction, Church–Rosser conversion, quotient equality of proofs, global normalization for all closed terms, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, false typing exposure, closedness and beta-strong endpoint exactness, false-exclusion factorization, nonescape ledgers, classifier transport, and provenance/name exhaustion for BEDC.Derived.ClosedNormalConsistencyWitnessUp.

Constructive story: A $\mathsf{ClosedNormalConsistencyWitness}^{\uparrow}$ packet is generated from a MetaCIC term row, an internal false-encoding row, an empty-context typing ledger, a closedness row, a beta-strong endpoint row, a false-exclusion witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3662

A Concrete Naming Certificate for $\mathsf{FiniteGroupRepresentation}^{\uparrow}$

Following Frobenius on finite-group characters and the representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ packet records a finite group action on a finite displayed vector-space row as NameCert data. It consumes the group source of chapter 41, the vector-space source of chapter 47, the linear-map and matrix rows of chapter 48 and chapter 49, and the downstream trace and ledger consumers of chapter 245, chapter 246, and chapter 3574. The packet names only the displayed finite action, transport, and consumer boundary rows; it does not assert Maschke semisimplicity, Schur orthogonality, complete reducibility, irreducible decomposition, or a host category of representations.

Definition 3662.1 (FiniteGroupRepresentation carrier)**.** *A* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, V, L, M, A, U, C, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, including the multiplication, unit, inverse, and finite support rows needed to index the action.*
2. *V is the consumed* $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ *row for the represented vector-space endpoint.*
3. *L is the consumed* $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ *row naming the linear endomorphism surface over V.*
4. *M is the consumed* $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ *row used only when the finite action is displayed by matrix rows over a chosen presentation already recorded in the packet.*
5. *A is the finite action ledger assigning to each displayed group row a displayed linear endomorphism row and recording the multiplication and unit compatibility rows.*
6. *U is the finite action-unit row, recording that the unit row of G acts by the identity linear-map row on V.*
7. *C is the downstream consumer ledger, exposing only the rows read by* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *and* $\mathsf{CharacterTable}^{\uparrow}$ *consumers.*
8. *H is the componentwise* hsame *transport for G, V, L, M, A, U, C.*
9. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *row.*

No carrier coordinate stores a host representation category, a quotient representation, an irreducible decomposition, a character table, a Maschke complement, or a Schur orthogonality relation.

Lean4 (definition) `BEDC.Derived.FiniteGroupRepresentationUp.FiniteGroupRepresentationUp`

Theorem 3662.2 (FiniteGroupRepresentation NameCert bridge)**.** *For every accepted* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *carrier*

$$R = (G, V, L, M, A, U, C, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission exposes the displayed group row G, vector-space row V, linear-map row L, optional matrix presentation row M, action ledger A, unit row U, consumer ledger C, transport row H, provenance row P, and local naming row N;*
2. *action exactness requires each action entry in A to be sourced by a group row of G and to land in a linear endomorphism row read through L over the represented vector-space row V;*
3. *unit and multiplication stability require the unit row U and the product rows of G to replay through finite* Cont *rows inside A and remain preserved by the componentwise* hsame *transport H;*
4. *matrix presentation compatibility allows M to display action entries only as rows already tied to L and V, so matrix reads cannot create an additional carrier outside the packet;*
5. *consumer boundary requires every* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTable}^{\uparrow}$ *read to pass through C after the action ledger has been displayed, and it excludes Maschke semisimplicity, Schur orthogonality, irreducible decomposition, quotient representation carriers, and host representation categories.*

These five obligations give the finite NameCert bridge from the consumed group and vector-space certificates to the downstream character and representation-ring consumers.

Proof. Project the carrier rows of Definition 3662.1. Carrier admission is exactly the displayed tuple $G, V, L, M, A, U, C, H, P, N$, so no additional source row is available.

For action exactness, read each entry of A as a finite row from the group source G into the endomorphism surface supplied by L over V. The unit row U and the product rows of G are then replayed through the same finite continuation ledger.

Matrix presentation compatibility is only a display route. Any matrix row in M must already be tied to the linear-map and vector-space rows L and V, so it cannot supply a second representation carrier.

Finally expose the consumer ledger C. Character-theory, representation-ring, and character-table consumers read only after the displayed action rows have been named, while Maschke semisimplicity and Schur orthogonality remain in their own chapters. Repacking the projected rows with H, P, and N gives the stated local NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteGroupRepresentationUp.FiniteGroupRepresentationNameCertBridge`

Theorem 3662.3 (FiniteGroupRepresentation Schur consumer boundary)**.** *Any* $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, *or* $\mathsf{SchurOrthogonality}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *packet*

$$R = (G, V, L, M, A, U, C, H, P, N)$$

must first expose the action ledger A, linear-map row L, vector-space row V, optional matrix row M, and consumer ledger C. A $\mathsf{SchurOrthogonality}^{\uparrow}$ *read may inspect finite character-pairing rows only through those displayed rows and cannot import a host representation category, quotient carrier, irreducible decomposition, or complete character table.*

Proof. Project the accepted carrier tuple of Definition 3662.1. The action ledger A, vector-space row V, linear-map row L, optional matrix row M, and consumer ledger C are the only rows through which downstream character-facing consumers can see the finite representation packet.

Route a Schur-facing request through C. The request may read character-pairing information only after the displayed action entries in A have landed in the endomorphism surface supplied by L over V, with any matrix presentation in M already tied to the same rows.

The exclusions follow from the carrier boundary. Since R contains no host representation category, quotient carrier, irreducible decomposition, or complete character-table coordinate, a $\mathsf{SchurOrthogonality}^{\uparrow}$ consumer cannot obtain any of those endpoints from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3662.1, Definition 3662.2, and Definition 3662.3 seed the finite action carrier, its NameCert obligation bridge, and the Schur-facing consumer boundary over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, and $\mathsf{SchurOrthogonality}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteGroupRepresentation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Maschke semisimplicity, Schur orthogonality, irreducible decomposition, complete reducibility, character-table computation, quotient representation carriers, host representation categories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, action exactness, unit and multiplication stability, matrix presentation compatibility, and downstream consumer boundary for the same finite packet.

Constructive story: A $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ packet is generated from a displayed finite group source, represented vector-space source, linear-map and matrix presentation rows, a finite action ledger, unit and multiplication replay rows, componentwise transport, package provenance, and local NameCert evidence.

Chapter 3663

A Concrete Naming Certificate for $\mathsf{KSpace}^{\uparrow}$

Following Kelley's compactly generated spaces, the $\mathsf{KSpace}^{\uparrow}$ packet records the topology whose public open rows are tested by compact probes. It sits after the topology carrier of chapter 88, the compact-space certificate surface, and the compact-open function-space consumer of chapter 2221. The packet names compact-test topology rows for downstream continuity and mapping-space consumers; it does not import arbitrary open-cover choice, a host category of spaces, quotient equality of opens, or an ambient set-theoretic coreflection.

Definition 3663.1 (K-space carrier)**.** *A* $\mathsf{KSpace}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$K = (X, T, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-name carrier whose observations are read only through* BHist *rows.*
2. *T is the ambient* $\mathsf{Topology}^{\uparrow}$ *row listing the candidate open-name classifier.*
3. *Q is the compact-probe ledger: each accepted row records a compact source certificate, a probe map into X, and its compact-domain provenance.*
4. *A is the compact-test topology ledger, saying that an open-name row U is admitted precisely when every compact probe in Q pulls U back to an open row on its compact source.*
5. *E records exactness between T and A for the displayed open names, so the packet exports the compactly generated topology rather than a second unrelated topology.*
6. *H records componentwise* hsame *transport for X, T, Q, A, E, C, P, N.*
7. *C records displayed* Cont *replay from topology requests and compact-probe requests through the same rows.*
8. *P is the* Pkg *provenance over* $\mathsf{KSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*, compact probes,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$ *row.*

The carrier exposes compact-test topology data only through $X, T, Q, A, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.KSpaceUp.KSpaceTasteGate single carrier alignment`

Definition 3663.2 (K-space classifier)**.** *The* $\mathsf{KSpace}^{\uparrow}$ *classifier accepts two carrier rows as the same public packet exactly when their point carrier, ambient topology row, compact-probe ledger, compact-test topology ledger, exactness row, transport row, replay row, provenance row, and local naming row are componentwise related by the displayed* hsame *evidence. It has no coordinate for selected subcovers, host open families, quotient-open representatives, choice of compact probes outside* Q*, or an external coreflection functor.*

Theorem 3663.3 (K-space NameCert obligations)**.** *For every accepted* $\mathsf{KSpace}^{\uparrow}$ *carrier* $K = (X, T, Q, A, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed point carrier, topology row, compact-probe ledger, compact-test topology ledger, exactness row, transport, replay, provenance, and local naming rows;*
2. *compact-probe accountability, requiring every probe inspected by* A *to be named by the compact-probe ledger* Q*;*
3. *compact-test exactness, requiring the admitted open rows to be exactly those whose pullbacks along all displayed compact probes are open on their compact sources;*
4. *componentwise transport and replay stability under* H *and* C *for the same displayed rows;*
5. *ledger refusal, stating that arbitrary open-cover choice, host compact-object selection, quotient equality of opens, and ambient set-theoretic coreflection are not carrier coordinates.*

No local obligation is stored outside these five rows.

Proof. Project the accepted carrier. The only mathematical rows are X, T, Q, A, E, and the structural rows are H, C, P, N.

The compact-probe accountability row is read through Q. Each pullback test used by A must name its compact source and probe map before the topology request can inspect it.

The exactness row E binds the topology row T to the compact-test ledger A. Hence the packet exports the compactly generated topology determined by the displayed probes, not a freely chosen open-family classifier.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since the excluded objects are absent from $X, T, Q, A, E, H, C, P, N$, no accepted packet can export them. □

Theorem 3663.4 (Compact-test continuity handoff)**.** *A downstream continuity or compact-open function-space consumer of an accepted* $\mathsf{KSpace}^{\uparrow}$ *carrier can inspect openness only through*

$$\mathsf{Topology}^{\uparrow}, \quad \mathsf{KSpace}^{\uparrow}, \quad Q, \quad A.$$

Thus a continuity check against a $\mathsf{KSpace}^{\uparrow}$ *source is routed through displayed compact probes and compact-test topology rows, while open-cover choice, external compact-object selection, and host coreflection data remain outside the packet.*

Proof. Read the topology request through T and the compact-test row through A. The row A can certify an open-name request only after all displayed compact probes in Q have supplied the corresponding pullback-open rows.

The exactness row E then identifies the exported open-name classifier with that compact-test ledger. A compact-open function-space consumer therefore receives the same probe route rather than a host-space construction.

Rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. No extra coordinate is available for open-cover choice, selected compact probes outside Q, or an ambient coreflection principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{KSpace}^{\uparrow}$ packet is generated from a point-name carrier, a TopologyUp row, compact-probe rows, compact-test topology rows, exactness between displayed opens and compact tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3663.1, Definition 3663.2, Definition 3663.3, and Definition 3663.4 seed the compactly generated topology packet for topology and compact-open consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.KSpaceUp.KSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected compact probes outside the displayed ledger, quotient equality of opens, host coreflections, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-probe accountability, compact-test exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.KSpaceUp.

Chapter 3664

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$

Following Heine–Cantor and Lebesgue finite-cover analysis, the $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ packet records the finite selector route used by compact-uniform-continuity proofs. It consumes the compact finite-net source of chapter 3606, the pointwise modulus rows of the continuous-function carrier, the rational lower-bound fold, and the uniform-modulus handoff. The packet names the finite cover selector as BEDC data; it is not an appeal to countable choice, a global modulus oracle, an open-cover compactness theorem, or an ambient minimum over a host set of radii.

Definition 3664.1 (Compact-uniform finite-cover selector carrier)**.** *A* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *carrier is a finite* BHist *tuple*

$$S = (K, F, E, M, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric or compact finite-net source row that supplies a finite* ProbeBundle *cover for the requested tolerance.*
2. *F is the continuous-function row whose graph and pointwise modulus are being uniformized.*
3. *E is the finite cover selector ledger, listing the selected compact centers, half-radius coverage rows, and probe provenance.*
4. *M is the pointwise modulus ledger at the selected centers.*
5. *R is the rational lower-bound fold over the displayed positive radius rows.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row exported after E, M, R have been read in order.*
7. *H records componentwise* hsame *transport for $K, F, E, M, R, U, C, P, N$.*
8. *C records displayed* Cont *replay from compact-uniform requests through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$*, compact finite covers, continuous-function pointwise moduli, rational lower-bound folds,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}}$ *row.*

The carrier exposes only the finite cover selector, pointwise modulus, rational fold, and uniform-modulus handoff rows displayed above.

Lean4 ✓ `BEDC.Derived.CompactUniformFiniteCoverSelectorUp.CompactUniformFiniteCoverSelectorTasteGate`
`single carrier alignment`

Definition 3664.2 (Compact-uniform selector classifier)**.** *The* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *classifier compares accepted packets componentwise through* $K, F, E, M, R, U, H, C, P, N$*. Two packets are classifier-equal exactly when the same compact source, continuous graph, finite cover selector, pointwise modulus rows, rational lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local naming rows are related by the displayed* hsame *evidence. It has no coordinate for an external selector, a host finite minimum, countable choice, a global modulus oracle, or open-cover compactness outside the displayed compact row.*

Theorem 3664.3 (Compact-uniform finite-cover selector obligations)**.** *For every accepted* CompactUniformFiniteCo *carrier* $S = (K, F, E, M, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the compact source, continuous-function row, finite cover selector, pointwise modulus ledger, rational lower-bound fold, uniform handoff, transport, replay, provenance, and local naming rows;*
2. *finite-cover accountability, requiring the selected centers and half-radius coverage rows in* E *to be read from the displayed compact source* K*;*
3. *pointwise-modulus compatibility, requiring every selected center in* E *to carry the matching pointwise modulus row in* M *for the same continuous-function row* F*;*
4. *rational-fold exactness and uniform handoff, requiring* R *to be a certified lower bound for the displayed radii before* U *exports the tolerance-only precision row;*
5. *ledger refusal, stating that countable choice, external finite minima, global modulus oracles, host open-cover compactness, and ambient completeness are not carrier coordinates.*

No accepted selector packet carries any additional NameCert obligation.

Proof. Project the accepted carrier. The compact source, continuous-function row, finite cover selector, pointwise modulus ledger, rational fold, and uniform handoff are precisely K, F, E, M, R, U.

The finite-cover accountability row is read first. Row E may list only centers and half-radius coverage witnesses supplied by the compact source row K.

The pointwise modulus row is then checked at those centers. Compatibility requires the ledger M to use the same continuous-function row F and the same selected centers exposed by E.

The rational fold R is read before the uniform handoff U. It certifies one positive precision history below the displayed half-radii, so U exports a tolerance-only uniform-modulus row without selecting a radius outside the finite ledger.

Rows H, C, P, N transport, replay, source, and name this route. Since no further coordinate occurs in S, the refused external principles cannot be exported by the local NameCert surface. □

Theorem 3664.4 (Finite selector to uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ *carrier reaches its precision history only through the finite route*

$$\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}, \quad E, \quad M, \quad R, \quad U.$$

The consumer sees a displayed compact finite-net selector, pointwise modulus rows at its centers, a certified rational lower-bound fold, and the uniform-modulus handoff; it receives no countable-choice selector, host minimum, or open-cover compactness datum.

Proof. Start with the compact finite-net selector E. It exposes a finite ProbeBundle of centers and half-radius coverage rows already tied to the compact source K.

At each displayed center, read the pointwise modulus row in M for the same continuous-function row F. The row order prevents a modulus from being imported for a different graph or an unlisted center.

Fold the positive radius rows through R. The fold certificate supplies one lower-bound precision history, and only then can U export the uniform-modulus handoff.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Therefore the consumer-facing datum is the finite BEDC selector route, not an external choice or host minimum. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}$.

Constructive story: A $\mathsf{CompactUniformFiniteCoverSelector}^{\uparrow}$ packet is generated from a compact source row, a continuous-function row, compact finite-net selector rows, pointwise modulus rows, rational lower-bound fold rows, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3664.1, Definition 3664.2, Definition 3664.3, and Definition 3664.4 seed the finite cover selector packet for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteCoverSelector}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, external finite minima, global modulus oracles, host open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover accountability, pointwise-modulus compatibility, rational-fold exactness, uniform handoff, transport and replay stability, and local NameCert registration for BEDC.Derived.CompactUniformFiniteCoverSelectorUp.

Chapter 3665

A Concrete Naming Certificate for $\mathsf{BishopRealCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealCompletion}^{\uparrow}$ packet records real completion by regular rational names with located finite observation data. It sits on the dyadic arithmetic substrate of chapter 291, the finite stream-window surface of chapter 126, the regular rational sequence packet of chapter 301, the terminal real-name seal of chapter 38, and the located completion route of chapter 3488.

Definition 3665.1 (Bishop real completion carrier)**.** *A* $\mathsf{BishopRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger whose positive dyadic rows name the tolerances inspected by the completion route;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the finite rational observations read from the candidate Cauchy name;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular rational values visible on the windows in S;*
4. *L is the* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *located-completion row, recording the located brackets available for the same dyadic precision ledger and stream windows;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached from D, S, Q, L;*
6. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readbacks, located-completion row, Real-seal row, replay row, provenance row, and local naming row;*
7. *C records displayed* Cont *replay from dyadic precision queries through stream readback, located completion, and Real sealing;*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *evidence.*

The carrier contains no quotient real object, no selected limiting representative, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRealCompletionUp.TasteGate.BishopRealCompletionTasteGate single carrier alignment`

Theorem 3665.2 (Bishop real completion NameCert obligations)**.** *For every accepted carrier* $R = (D, S, Q, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; dyadic precision exactness through* D*; finite stream-window readback through* S*; regular rational Cauchy readback through* Q*; located completion handoff through* L*; Real-seal exposure through* E*; componentwise* hsame *stability through* H*; displayed* Cont *replay through* C*;* Pkg *provenance through* P*; and local naming through* N*.*

Proof. Carrier admission is coordinate projection from Definition 3665.1. The packet displays the dyadic precision ledger, finite stream windows, regular rational readbacks, located-completion row, Real-seal row, transport, replay, provenance, and local name as one BHist surface.

The dyadic and stream obligations are read before the located-completion row. Thus every precision request uses a positive row of D, every rational observation enters through the finite window ledger S, and every rational value is exposed by the regular readback row Q.

The located-completion and Real-seal obligations are then forced through L and E. Since H and C are the only transport and replay rows, any classified mate of the packet preserves the same finite route.

Finally P and N record provenance and local naming for exactly these rows. The carrier inventory has no quotient real, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, so the listed obligations exhaust the certificate surface. □

Theorem 3665.3 (Bishop real completion Real-seal route)**.** *Every Real-facing consumer of an accepted* $\mathsf{BishopRealCompletion}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopLocatedRealCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $D, S, Q, L, E, H, C, P, N$*. The route supplies finite precision, window, readback, located-completion, and seal rows, and it supplies no quotient real equality, no selected limit, no countable-choice schedule, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $R = (D, S, Q, L, E, H, C, P, N)$. The only Real-facing row is E, and the only rows feeding it are the dyadic ledger D, stream windows S, regular rational readbacks Q, and located-completion row L.

Use the NameCert obligations of Definition 3665.2. They force precision queries, observations, regular readbacks, located completion, and Real sealing to pass through the same displayed carrier rows.

Transport and replay do not add a second route. The rows H and C preserve and replay the displayed path componentwise, while P and N record its provenance and local name.

The carrier excludes quotient real equality, selected limits, countable-choice schedules, and ambient completeness. Hence every Real-facing consumer receives exactly the finite Real-seal route stated above. □

Theorem 3665.4 (Bishop real completion local NameCert exhaustion)**.** *For any accepted* $\mathsf{BishopRealCompletion}^{\uparrow}$ *carrier*

$$R = (D, S, Q, L, E, H, C, P, N),$$

every local $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$ *consumer factors through exactly the five dependency rows*

$$D, S, Q, L, E$$

followed by the structural rows H, C, P, N*. The first five rows are respectively the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger,* $\mathsf{StreamName}^{\uparrow}$ *window ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ *located-completion row, and* $\mathsf{Real}^{\uparrow}$ *seal; no consumer-visible branch is supplied by quotient real equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier using Definition 3665.1. The displayed dependency coordinates are exactly D, S, Q, L, E, and the only structural coordinates are H, C, P, N.

Split over the finite row inventory. Precision requests land in D, finite observations in S, regular rational values in Q, located completion data in L, and Real-facing seals in E.

Apply Definition 3665.2. Its obligation list identifies the same dependency rows and then the same transport, replay, provenance, and local naming rows as the full local certificate surface.

Finally use Definition 3665.3 for consumer-visible reads that reach $\mathsf{Real}^{\uparrow}$. That route ends at E after D, S, Q, L, and the remaining rows H, C, P, N preserve and name it without adding a second branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealCompletion}^{\uparrow}$ packet is generated from a dyadic precision ledger, finite StreamName windows, RegSeqRat readbacks, a BishopLocatedRealCompletion row, a Real seal row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3665.1, Definition 3665.2, and Definition 3665.3 name the seed carrier, finite NameCert obligation surface, and Real-seal route for Bishop real completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert quotient reals, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic precision exactness, finite stream-window readback, regular rational readback, located-completion handoff, Real-seal exposure, transport stability, replay, provenance, and local naming for the displayed Bishop real completion packet.

Chapter 3666

A Concrete Naming Certificate for UniformCompletionUniversalProperty$^{\uparrow}$

Following Bishop and Bridges constructive completion and Kelley uniform-space terminology, the UniformCompletionUniversalProperty$^{\uparrow}$ packet records the uniform analogue of the L10 completion universal property. It sits after the uniform-completion carrier of chapter 1473, the generic completion surface of chapter 397, and the regular-rational readbacks of chapter 126, chapter 301, and chapter 38. The packet names extension and uniqueness only through finite uniform rows, not through quotient completions, selected limits, ambient complete spaces, or choice.

Definition 3666.1 (Uniform completion universal-property carrier)**.** *A* UniformCompletionUniversalProperty$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, D, T, E, L, M, W, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform source row, exposing only the displayed* UniformSpace$^{\uparrow}$ *entourages and their finite source windows.*
2. *D is the dense unit row from the source into the completion carrier, aligned with the* UniformCompletion$^{\uparrow}$ *unit surface.*
3. *T is the uniformly continuous target row, naming the complete separated target interface through finite entourage/modulus data.*
4. *E is the extension row for uniformly continuous source consumers, read only after U, D, T have been displayed.*
5. *L is the uniqueness row, comparing two displayed extensions by* hsame *over the dense unit and the same target row.*
6. *M is the entourage/modulus ledger for source entourages, target entourages, and the finite modulus witnesses used by E and L.*
7. *W is the readback row for* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *observations downstream of the same uniform route.*
8. *H stores componentwise* hsame *transport for the source, dense unit, target, extension, uniqueness, modulus, readback, continuation, provenance, and naming rows.*

9. *C stores displayed* Cont *replay from source windows through the dense unit, target modulus, extension row, uniqueness row, and readback row.*

10. *P is the* Pkg *provenance over* UniformCompletionUniversalProperty$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CompletionUniversalPro UniformSpace$^{\uparrow}$, UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* NameCert$_{\text{UniformCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, D, T, E, L, M, W, H, C, P, N$. It has no coordinate for quotient completion points, selected representatives, ambient target completeness, host function extensionality, or countable choice.

Theorem 3666.2 (Uniform completion universal-property NameCert obligations)**.** *For every accepted* UniformCompletionUniversalProperty$^{\uparrow}$ *carrier $K = (U, D, T, E, L, M, W, H, C, P, N)$, the local* NameCert$_{\text{UniformCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the eleven displayed rows, uniform-source exposure through U, dense-unit exposure through D, complete separated target exposure through T, extension existence through E, uniqueness through L, entourage/modulus accounting through M,* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *readbacks through W, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.UniformCompletionUniversalPropertyUp.UniformCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 3666.1. The source, dense unit, target, extension, uniqueness, modulus ledger, readback row, transport, replay, provenance, and local name are exactly the displayed coordinates of one finite BHist packet.

The extension row E is available only after the source row U, dense unit row D, target row T, and entourage/modulus ledger M have been displayed. The uniqueness row L compares extensions over the same rows by the recorded hsame ledger.

The readback row W is downstream of that uniform route. Since the classifier has no remaining coordinate for selected limits, quotient completions, host function equality, countable choice, or ambient target completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3666.3 (Uniform completion universal-property extension uniqueness)**.** *Let $K = (U, D, T, E, L, M, W, H$ be an accepted* UniformCompletionUniversalProperty$^{\uparrow}$ *carrier. Every extension-and-uniqueness consumer factors through*

$$U, D, T, M \longmapsto E \longmapsto L \longmapsto W$$

together with H, C, P, N. Thus extension existence, uniqueness over the dense unit, and any StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *or* Real$^{\uparrow}$ *readback are all read from the same finite uniform packet, and no consumer obtains a quotient completion, selected extension, ambient complete target theorem, or choice principle.*

Proof. Read the carrier definition. The extension coordinate is E, and its stated inputs are the uniform source U, dense unit D, uniformly continuous target T, and entourage/modulus ledger M.

The uniqueness row L is not an independent equality principle. It compares displayed extensions only along the same dense unit and target row, using the componentwise hsame route recorded by H and replayed by C.

Finally inspect the downstream rows. The readback coordinate W is attached to the same route, while P and N record provenance and local naming. Row exhaustion excludes every omitted host object, so the extension-and-uniqueness surface is exactly the finite route displayed above. □

Falsifiable prediction: Every accepted $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence or uniqueness while omitting the uniform source, dense unit, uniformly continuous target, extension row, uniqueness row, entourage/modulus ledger, readback row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: uniformcompletion, completionuniversalproperty, uniformspace, uniformmodulus, streamname, regseqrat, real: $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries uniform source, dense unit, target continuity, extension, uniqueness, modulus ledger, readbacks, transport, replay, provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionUniversalProperty}^{\uparrow}$ packet is generated from a uniform source row, dense unit row, uniformly continuous target row, extension row, uniqueness row, entourage/modulus ledger, StreamName/RegSeqRat/Real readbacks, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3666.1, Definition 3666.2, and Definition 3666.3 seed the finite uniform universal-property certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected extensions, ambient complete target theorems, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, extension existence, uniqueness, readback nonescape, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.UniformCompletionUniversalPropertyUp.

Chapter 3667

A Concrete Naming Certificate for CauchyCompletionStableReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completion, the CauchyCompletionStableReflection$^{\uparrow}$ packet records the stable reflection surface by which a displayed completion functor, its universal property, separatedness, regular rational sequence data, and terminal real-name rows remain attached to one finite NameCert route. It sits after the completion functor of chapter 423, the universal handoff of chapter 397, the separated limit classifier of chapter 1411, the regular rational sequence rows of chapter 301, and the Real seal of chapter 38. The packet names a stable reflection certificate for the displayed route; it does not assert quotient completion construction, selected limits, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 3667.1 (Cauchy completion stable reflection carrier)**.** *A* CauchyCompletionStableReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, U, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyCompletionFunctor$^{\uparrow}$ *row, containing the named source metric, target completion surface, and functorial map route.*
2. *U is the* CompletionUniversalProperty$^{\uparrow}$ *row, storing the extension and uniqueness handoff for maps out of the displayed dense source.*
3. *S is the* SeparatedMetric$^{\uparrow}$ *row, recording the zero-distance classifier used for completion-limit uniqueness.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *row, carrying the regular rational sequence schedules and finite tolerance windows read by the completion route.*
5. *E is the* Real$^{\uparrow}$ *seal row reached from the regular sequence and zero-distance data.*
6. *H records componentwise* hsame *transport for functor, universal-property, separatedness, regular-sequence, Real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from F, Q through U, S, E and back to the local naming row.*
8. *P is the* Pkg *provenance over* CauchyCompletionStableReflection$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}$ *row.*
The carrier contains no quotient completion object, no chosen representative of a limit, no external completeness theorem for $\mathsf{Real}^{\uparrow}$*, and no route outside the displayed BEDC rows.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionStableReflectionUp.CauchyCompletionStableReflectionTasteGate`
`single carrier alignment`

Theorem 3667.2 (Cauchy completion stable reflection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ *carrier* $R = (F, U, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableRefl}}$ *surface consists exactly of carrier admission for* $F, U, S, Q, E, H, C, P, N$*; functor stability through* F*; extension uniqueness through* U*; separated zero-distance classification through* S*; regular rational window control through* Q*; Real seal confinement through* E*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Carrier admission is projection from Definition 3667.1. The accepted packet displays exactly the completion functor row, the universal-property row, the separatedness row, the regular rational sequence row, the Real seal, and the structural rows.

Functor stability and extension uniqueness are read in that order. The row F supplies the completion-facing map route, while U allows an extension consumer to compare two displayed routes only after they share the same source, completion surface, and regular sequence data.

Separatedness and Real sealing are likewise internal to the packet. The row S turns zero-distance evidence into classifier equivalence, the row Q supplies the finite regular rational windows that feed the completion comparison, and E seals the resulting value inside the displayed $\mathsf{Real}^{\uparrow}$ boundary.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since the carrier has no quotient object, selected limit, countable-choice schedule, trichotomy row, or ambient Real-completeness coordinate, every local NameCert obligation is confined to the listed rows. □

Theorem 3667.3 (Cauchy completion stable reflection extension route)**.** *Every extension-facing consumer of an accepted* $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}^{\uparrow} \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Equivalently, the consumer receives the rows $F, U, S, Q, E, H, C, P, N$ *and no quotient completion carrier, selected representative, hidden completion theorem, countable-choice schedule, trichotomy row, or off-packet certificate.*

Proof. Begin with the functor row F. It is the only coordinate that exposes the completion-facing map route and the source-to-completion handoff.

Next use the universal-property row U. It compares extension candidates only after the route through F is fixed, so extension uniqueness cannot be read as a host equality or as a selected representative of a completed object.

The separatedness and regular-sequence rows then bind the uniqueness read to finite BEDC data. The row S supplies the zero-distance classifier, Q supplies the regular rational windows that make the completion comparison visible, and E confines the resulting value to the Real seal.

Finally H, C, P, N transport, replay, source, and name that same route. Coordinate exhaustion rules out any hidden quotient completion, ambient Real-completeness theorem, choice schedule, trichotomy row, or off-packet certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionStableReflection}^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a CompletionUniversalProperty row, a SeparatedMetric row, a RegSeqRat row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3667.1, Definition 3667.2, and Definition 3667.3 seed the finite stable-reflection certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStableReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, functor stability, extension uniqueness, separated zero-distance classification, regular rational window control, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionStableReflectionUp.

Chapter 3668

A Concrete Naming Certificate for $\mathsf{SetLike}^{\uparrow}$

Following Russell and Bourbaki-style set-membership grounding, the $\mathsf{SetLike}^{\uparrow}$ packet records the finite BEDC carrier by which membership, subset implication, restricted comprehension, and extensional boundary reads are exposed without importing an ambient powerset, host quotient, or impredicative universe. It sits before the displayed family discipline of chapter 3600, the syntactic carrier of chapter 197, and the satisfaction carrier of chapter 198. The packet names only finite membership source, classifier, implication, bounded comprehension, extensionality boundary, transport, replay, provenance, and local naming rows.

Definition 3668.1 (SetLike carrier)**.** *A* $\mathsf{SetLike}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, Q, I, R, E, H, C, P, N)$$

with the following rows:

1. *M is the membership-source row, naming the finite source from which element and set-read markers are admitted;*
2. *Q is the classifier row for displayed membership decisions and membership-compatible reads;*
3. *I is the subset-implication row, recording finite implications from one membership classifier to another;*
4. *R is the restricted-comprehension row, admitting only classifiers already routed through M, Q, I;*
5. *E is the extensionality-boundary ledger, recording when two displayed set rows may be compared by the carried membership classifier;*
6. *H records componentwise* hsame *transport for M, Q, I, R, E together with the structural rows;*
7. *C records* Cont *replay through which a consumer reads membership source, classifier, subset implication, restricted comprehension, and extensionality boundary data;*
8. *P is the* Pkg *provenance over* $\mathsf{SetLike}^{\uparrow}$*,* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{ModelTheory}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, global choice of elements, or a host universe of all sets.

Lean4 ✓ BEDC.Derived.SetlikeUp.SetlikeTasteGate single carrier alignment

Definition 3668.2 (SetLike membership carrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the membership carrier is the finite BHist *surface*

$$S_{\in} = (M, Q, H, C, P, N).$$

Here M is the membership-source row, Q is the classifier-based membership row, H is the componentwise hsame *transport for the displayed rows, C is the* Cont *replay route, P is the* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The membership carrier has no coordinate for host elementhood, powersets, quotient equality of sets, unrestricted comprehension, or an external universe of sets.*

Lean4 (definition) BEDC.Derived.SetlikeUp.SetlikeMembershipCarrier

Theorem 3668.3 (SetLike membership stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P,$ every classifier-based membership read through the membership carrier $S_{\in} = (M, Q, H, C, P, N)$ remains the same finite* BHist *read after componentwise* hsame *transport and* Cont *replay. In particular, transport may expose only the membership source M, classifier row Q, structural route H, C, provenance P, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row N.*

Lean4 ✓ BEDC.Derived.SetlikeUp.SetlikeMembershipStability

Proof. Project the membership carrier of Definition 3668.2. Its only membership data are the source row M and the classifier row Q.

Transport is componentwise through H, and replay is displayed through C. These rows preserve and expose the same classifier-based membership surface rather than creating a second element relation.

The rows P and N source and name the same finite packet. Hence a transported membership read remains a BHist read through M, Q, H, C, P, N and cannot become host elementhood, a powerset coordinate, quotient equality, or unrestricted comprehension. □

Theorem 3668.4 (SetLike membership classifier public boundary)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, every* $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *or root consumer must read membership through M, Q, H, C, P, N, subset and restricted-comprehension data through I, R, and the extensional boundary through E. The public boundary exports no host membership relation, powerset, unrestricted separation rule, quotient set equality, selected representative, or external satisfaction predicate.*

Proof. Project Definition 3668.1 and Definition 3668.2. The membership carrier exposes only the source row M, classifier row Q, transport row H, replay row C, provenance row P, and local naming row N.

Apply Definition 3668.3. A membership read transported through the public boundary remains the same finite classifier read, so a downstream root or typed consumer cannot replace it by host elementhood or an external satisfaction predicate.

For subset and comprehension consumers, use Definition 3668.5 and Definition 3668.6. These place subset implication and restricted comprehension behind I, R over the displayed membership classifier.

Finally read the extensional boundary through E and close by the existing consumer-boundary rows. Since no coordinate of S stores a powerset, unrestricted separation rule, quotient set equality, selected representative, or external satisfaction predicate, the public boundary exports exactly the displayed rows. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierPublicBoundary`

Theorem 3668.5 (SetLike subset ledger exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the subset ledger is exact: every accepted subset read is the implication row* I *over the classifier-based membership carrier* $S_{\in}$*, together with the restricted-comprehension row* R *only as a bounded consumer of* M, Q, I*. No subset consumer receives host-set extensionality, a global inclusion predicate, quotient equality of sets, or a powerset row outside* M, Q, I, R, H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetLedger exactness`

Proof. Start with the membership carrier $S_{\in}$ and the subset row I. The classifier Q is the only membership classifier available to the implication row, and M is the only membership source.

The restricted-comprehension row R is bounded by the same displayed route. It can consume M, Q, I but does not provide an ambient subset object.

Transport and replay through H, C preserve the same ledger, while P, N record provenance and local naming. Since no extra coordinate exists, the subset ledger is exactly the finite route M, Q, I, R, H, C, P, N. □

Theorem 3668.6 (SetLike comprehension scope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P,$ *restricted comprehension through* R *is scoped by classifier-based membership over* M, Q *and subset implication through* I*. The resulting read remains a finite* BHist *row transported by* H*, replayed by* C*, sourced by* P*, and named by* N*; it does not assert unrestricted separation, host-set extensionality, quotient equality, or a universe of all subsets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeComprehensionScope`

Proof. Read R from the carrier of Definition 3668.1. The row is restricted comprehension, not a powerset constructor.

The scope is supplied by the membership carrier of Definition 3668.2 and the implication row I. Thus a comprehension read is accepted only when its classifier is already routed through M, Q, I.

Finally apply the structural rows. Componentwise hsame transport, displayed Cont replay, provenance, and local naming keep the same finite row visible, so the comprehension scope does not become unrestricted separation, host-set extensionality, quotient equality, or a universe of all subsets. □

Theorem 3668.7 (SetLike bounded-comprehension induction)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every accepted restricted-comprehension read through* R *is generated by finite induction over the displayed membership-source row* M *and subset-implication row* I*. The generated row is transported through* H *and replayed through* C *before provenance* P *and local naming* N *are consumed, and the route cannot introduce unrestricted separation, powerset data, quotient set equality, selected representatives, or an external universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionInduction`

Proof. Project the carrier of Definition 3668.1. The only rows available before restricted comprehension is read are the membership-source row M, the classifier row Q, and the subset-implication row I.

Apply Definition 3668.6. It fixes R as a bounded consumer of M, Q, I, so the induction has no case for a host powerset, unrestricted separator, quotient equality witness, or external universe.

Induct over the displayed BHist packet. The base case reads the empty or initial membership-source row and carries no unrestricted subset data. In the finite step case, a new displayed membership row from M may be admitted only when the implication row I keeps it inside the already displayed classifier route.

Transport through H and replay through C preserve the same generated route, while P and N only source and name the finite packet. Thus every accepted restricted-comprehension read is produced by the finite row induction and cannot export any refused comprehension principle. □

Theorem 3668.8 (SetLike finite membership induction)**.** *Every accepted finite* $\mathsf{SetLike}^{\uparrow}$ *membership ledger whose rows are generated from displayed* BHist *membership observations and transported by* hsame *can be consumed by induction over the displayed ledger rows. The induction exports only the carrier row, membership row, classifier row, transport row,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row; it exports no host powerset membership, quotient-set equality, unrestricted comprehension, selected representatives, or ambient extensional equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFiniteMembershipInduction`

Proof. Start from the membership carrier of Definition 3668.2. Its displayed ledger exposes only M, Q, H, C, P, N, so the base case is the empty or initial membership observation together with the same carrier, replay, provenance, and naming rows.

For the finite step case, append one displayed BHist membership observation. The row is admitted only through the classifier row Q and then transported componentwise by hsame through H.

Finally read the structural rows. The Cont replay C, provenance P, and local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ row N repack the same finite ledger for downstream consumers, while Definition 3668.3 blocks replacement by host powerset membership or ambient extensional equality. □

Theorem 3668.9 (SetLike membership-family BHist induction)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any finite membership-family consumer generated from* $M, Q, I, R, E, H, C, P, N$ *is obtained by induction on the displayed* BHist *membership source of Definition 3668.2. The step case first routes a displayed membership request through the classifier row* Q*, then through the subset-implication row* I*, and only then may the restricted-comprehension row* R *consume it. The induction has no branch for powersets, unrestricted separation, quotient set equality, host universe rows, selected representatives, or ambient elementhood.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipFamilyBHistInduction`

Proof. Start with Definition 3668.1 and Definition 3668.2. The displayed membership-family source is $S_{\in} = (M, Q, H, C, P, N)$, so the base case exposes only the membership source, classifier, transport, replay, provenance, and local naming rows.

For the finite step case, read the classifier row Q before any subset or comprehension consumer is allowed. The implication row I records the displayed subset route over that classifier, and Definition 3668.7 then permits R to consume only the already routed M, Q, I data.

The extensional boundary remains separate. By Definition 3668.82, E is a displayed comparison boundary and not a quotient set equality row. Hence the induction never opens powerset data,

unrestricted separation, quotient equality, host universe rows, selected representatives, or ambient elementhood. □

Theorem 3668.10 (SetLike NameCert obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *surface consists exactly of carrier admission for* S*, membership-source admission through* M*, classifier exposure through* Q*, subset-implication exposure through* I*, restricted-comprehension exposure through* R*, extensionality-boundary exposure through* E*, componentwise* hsame *stability through* H*,* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeNameCertObligations`

Proof. Project the accepted carrier of Definition 3668.1. The membership source M and classifier Q provide the only entry points for element-in-set reads.

The subset row I records implication between displayed classifiers, while the restricted-comprehension row R may bind only predicates already exposed through the same membership source and classifier route. The extensionality row E records the comparison boundary for displayed set rows; it does not identify host sets outside the packet.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert obligations cannot require unrestricted separation, ambient powersets, quotient equality of sets, selected representatives, global element choice, or a universe of all sets. □

Theorem 3668.11 (SetLike membership source admission)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every membership read must expose the membership-source row* M*, displayed classifier row* Q*, componentwise transport row* H*, continuation replay row* C*, package provenance row* P*, and local naming row* N *before the read is accepted. In particular, a membership consumer cannot replace* M, Q, H, C, P, N *by an ambient element relation, a host powerset, selected representatives, or an external universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipSourceAdmission`

Proof. Project the carrier of Definition 3668.1. The only rows that open an element-in-set assertion are the membership source M and displayed classifier Q.

Use the obligation surface of Definition 3668.10. It requires membership-source admission, classifier exposure, componentwise hsame stability, Cont replay, provenance, and local naming as visible NameCert rows.

The existing route theorem Definition 3668.27 then blocks every bypass. Since no carrier coordinate stores an ambient element relation, powerset, representative selector, or host universe, every accepted membership read must expose M, Q, H, C, P, N. □

Theorem 3668.12 (SetLike classifier transport route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, classifier transport for subset implication, restricted comprehension, and extensional boundary rows stays inside the finite route*

$$Q, I, R, E \longmapsto H, C, P, N.$$

The transported classifier remains the displayed membership classifier over M *and cannot be reinterpreted as proof-irrelevant host truth, unrestricted separation, quotient equality of sets, or an external inclusion relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierTransportRoute`

Proof. Start from the classifier row Q in Definition 3668.1. The subset implication row I, restricted-comprehension row R, and extensionality-boundary ledger E are all defined as displayed rows that consume the same membership classifier.

Apply Definition 3668.58, Definition 3668.59, and Definition 3668.60. These obligations keep subset, comprehension, and extensional comparison reads attached to M, Q and to the structural rows H, C, P, N.

Finally use Definition 3668.68. Transport and replay preserve the displayed classifier route rather than replacing it by host truth, unrestricted separation, quotient equality, or external inclusion. □

Theorem 3668.13 (SetLike membership carrier admission)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any membership-facing consumer must first admit the membership carrier*

$$S_{\in} = (M, Q, H, C, P, N)$$

from Definition 3668.2. The admissible read exposes M, Q, H, C, P, N *as source, classifier, transport, replay, provenance, and local naming rows, and it cannot inspect host elementhood, a powerset coordinate, unrestricted comprehension, quotient equality, selected representatives, or an ambient universe before those rows are visible.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierAdmission`

Proof. Project the carrier of Definition 3668.1 and the membership carrier of Definition 3668.2. The only membership rows available at this stage are the source row M and classifier row Q.

Use Definition 3668.10. The NameCert surface requires componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N for the same displayed rows.

Finally apply Definition 3668.12. Classifier transport keeps the read inside the displayed route, so no consumer can replace M, Q, H, C, P, N by host elementhood, powerset data, unrestricted comprehension, quotient equality, selected representatives, or an ambient universe. □

Theorem 3668.14 (SetLike membership kernel obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every consumer that reads membership or subset data from* S *must first expose the membership source* M*, displayed classifier* Q*, subset implication* I*, restricted comprehension* R*, extensional boundary* E*, componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. The admissible read is therefore the concrete kernel route*

$$M, Q, I, R, E \longmapsto H, C, P, N,$$

and it cannot be replaced by an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a host universe.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipKernel obligations`

Proof. Project the carrier of Definition 3668.1. Its membership-facing coordinates are exactly the source row M and the classifier row Q, while subset and bounded family reads are carried only by I and R.

Use Definition 3668.13. A membership-facing consumer must admit $S_{\in} = (M, Q, H, C, P, N)$ before it can inspect element-in-set data, so the source and classifier are visible before the structural rows are replayed.

Next apply Definition 3668.12. The subset implication, restricted comprehension, and extensional boundary rows stay attached to the displayed classifier route Q, I, R, E under H, C, P, N.

It remains to exclude host-level replacements. The membership route theorem Definition 3668.27 and the bounded-comprehension and extensional-boundary obligations keep every membership, subset, comprehension, and comparison read inside $M, Q, I, R, E, H, C, P, N$. Since Definition 3668.1 has no powerset, unrestricted-separation, quotient-equality, representative-selection, or host-universe coordinate, every accepted consumer must expose the displayed kernel route before reading membership or subset data. □

Theorem 3668.15 (SetLike subset classifier exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, subset implication and restricted comprehension are exact over the classifier route*

$$Q, I, R, E \longmapsto H, C, P, N.$$

The implication row I *and restricted-comprehension row* R *may use only the displayed classifier* Q *and extensional-boundary ledger* E *before transport, replay, provenance, and local naming are read. They export no powerset row, host-universe coordinate, unrestricted separation rule, quotient set equality, or external inclusion predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetClassifierExactness`

Proof. Start from Definition 3668.1. The rows I and R are displayed subset-implication and restricted-comprehension rows over the classifier row Q, while E records the comparison boundary.

Apply Definition 3668.10. Its obligation surface keeps subset, comprehension, and boundary reads attached to the accepted packet before structural replay.

Use Definition 3668.12. Since classifier transport stays inside Q, I, R, E, H, C, P, N, exactness is a property of the displayed rows rather than an ambient powerset, host universe, unrestricted separation rule, quotient equality, or external inclusion relation. □

Theorem 3668.16 (SetLike TypeLike and model-theory handoff)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer that reads family, satisfaction, or model data from an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must first pass through the membership carrier* M, Q, H, C, P, N *and the classifier route* Q, I, R, E, H, C, P, N*. Satisfaction-facing and family-facing reads therefore remain downstream of the SetLike membership and classifier surface, not of an ambient set-theoretic assumption.*

Proof. Begin with the accepted carrier of Definition 3668.1. The membership source, classifier, subset implication, restricted comprehension, and extensional boundary rows are M, Q, I, R, E.

Use Definition 3668.2 and Definition 3668.10. These fix the membership carrier and require the structural rows H, C, P, N before a sibling consumer receives any membership data.

Apply Definition 3668.12. The classifier route remains displayed through Q, I, R, E, H, C, P, N, so $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ consumers may build family, syntax, and satisfaction reads only after that SetLike surface is visible.

Thus no downstream consumer obtains satisfaction, family membership, or model-theory data from an ambient powerset, host universe, external satisfaction relation, quotient set equality, or unrestricted comprehension principle. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypelikeModeltheoryHandoff`

Theorem 3668.17 (SetLike membership classifier transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, membership transport is accepted only as the displayed classifier route*

$$M, Q \longmapsto H, C, P, N.$$

Here M *is the finite* BHist *membership source,* Q *is the membership classifier,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The transported read cannot become host elementhood, an ambient powerset, quotient equality of sets, unrestricted comprehension, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierTransport`

Proof. Project the accepted carrier and its membership subcarrier. The only element-facing rows before the structural tail are M and Q.

Apply the SetLike membership route and classifier transport route. These force every transported membership read to remain a classifier-bounded BHist read before H and C replay it.

Finally use the local NameCert obligations. The rows P and N source and name the same displayed route, so no host elementhood, powerset, quotient equality, unrestricted comprehension, or external satisfaction coordinate is available. □

Theorem 3668.18 (SetLike NameCert ledger nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *ledger records membership, subset implication, restricted comprehension, and extensional-boundary obligations only through*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The ledger is a finite BHist *route stable under* hsame *and* Cont*; it cannot export an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a host universe outside the displayed packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeNameCertLedgerNonescape`

Proof. Read the NameCert obligations of Definition 3668.10. They expose exactly the carrier rows $M, Q, I, R, E, H, C, P, N$.

Use subset classifier exactness, restricted-comprehension membership factorization, and extensional-subset transport. These keep subset, comprehension, and extensional comparison on the same membership classifier before structural replay.

The structural tail is exhausted by hsame transport, Cont replay, provenance, and local naming. Therefore the ledger cannot escape into powersets, unrestricted separation, quotient equality, selected representatives, or a host universe. □

Theorem 3668.19 (SetLike type-model satisfaction boundary)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must first expose* $M, Q, I, R, E, H, C, P, N$ *as a finite* BHist *route with componentwise* hsame*, displayed* Cont*, provenance, and local* NameCert *rows. Type, syntax, and satisfaction reads remain downstream of this displayed SetLike route and cannot replace it by host membership, quotient equality of sets, unrestricted comprehension, an ambient universe, or an external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeModelSatisfactionBoundary`

Proof. Start with the SetLike TypeLike and model-theory handoff. It already requires family, syntax, and satisfaction consumers to pass through the membership carrier and classifier route.

Apply membership classifier transport and NameCert ledger nonescape. Together they expose $M, Q, I, R, E, H, C, P, N$ and block any replacement by host membership or a powerset-style carrier.

Finally use the FirstOrder signature and ModelTheory satisfaction root-unblock theorems. A typed or model-theoretic consumer that stays on the displayed route receives only the finite BEDC packet, while a consumer that leaves it lacks the required SetLike source. □

Theorem 3668.20 (SetLike family handoff obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{TypeLike}^{\uparrow}$ *family consumer that reads indexed fibers from* S *must expose the membership source* M*, classifier row* Q*, subset implication row* I*, restricted-comprehension row* R*, extensional boundary row* E*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N*. The family handoff cannot replace these rows by an ambient powerset, unrestricted separation principle, host element relation, quotient equality of sets, or external universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyHandoffObligation`

Proof. Project the accepted carrier of Definition 3668.1. The family consumer sees only the displayed rows $M, Q, I, R, E, H, C, P, N$ before it can build a TypeLike fiber read.

Use Definition 3668.16. It forces sibling family reads through the membership carrier and the classifier route, so M, Q and Q, I, R, E remain visible before the consumer proceeds.

Finally apply Definition 3668.26 and Definition 3668.52. These keep the restricted family row bounded and the comparison row tied to the displayed classifier; hence no ambient powerset, unrestricted separation, quotient set equality, host element relation, or external universe is exported. □

Theorem 3668.21 (SetLike satisfaction handoff obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer must obtain its membership and comprehension base through the concrete route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The satisfaction handoff exports no host satisfaction predicate, ambient powerset, unrestricted separation rule, quotient equality of sets, selected representative, or external universe outside those nine rows.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSatisfactionHandoffObligation`

Proof. Start with Definition 3668.16. Satisfaction-facing reads are downstream of the SetLike membership and classifier surface, not independent model-theoretic coordinates.

Apply Definition 3668.54. First-order syntax may consume membership, subset implication, restricted comprehension, and extensional comparison only after the carrier route has been displayed.

Then use Definition 3668.89. The ModelTheory satisfaction consumer inherits the same classifier-bounded envelope through FirstOrder syntax, so host satisfaction, ambient powersets, unrestricted separation, quotient equality, selected representatives, and external universes remain outside the route. □

Theorem 3668.22 (SetLike family fiber obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer accepted from* S *must expose the membership source* M*, classifier row* Q*, subset implication row* I*, restricted-comprehension row* R*, extensional boundary row* E*, componentwise transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *before any* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer reads satisfaction or family membership. The fiber route cannot replace those rows by a host powerset, unrestricted separation principle, quotient equality of sets, selected representative, external satisfaction predicate, or ambient universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberObligations`

Proof. Project the accepted carrier of Definition 3668.1. The family-fiber consumer has access only to $M, Q, I, R, E, H, C, P, N$ before it can form a typed fiber read.

Use the SetLike family handoff obligation. It forces indexed fibers through the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows.

Finally compose with the satisfaction handoff obligation and the type-model satisfaction boundary. FirstOrder and ModelTheory consumers inherit the same visible rows from the TypeLike fiber route, so satisfaction and family membership cannot be read from an external powerset, host element relation, quotient set equality, selected representative, unrestricted separation rule, or ambient universe. □

Theorem 3668.23 (SetLike family-fiber model-satisfaction route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer that feeds a* $\mathsf{FirstOrder}^{\uparrow}$ *or* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction read must expose the rows* $M, Q, I, R, E, H, C, P, N$ *before satisfaction or family membership is inspected. The route cannot be replaced by host membership, quotient equality of sets, unrestricted comprehension, selected representatives, or an external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberModelSatisfactionRoute`

Proof. Start with the family handoff obligation. It opens the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows before a TypeLike family fiber is available.

Next compose with the satisfaction handoff obligation and the type-model satisfaction boundary. These force FirstOrder and ModelTheory satisfaction reads to inherit the same displayed SetLike route rather than introducing a separate model-theoretic carrier.

Finally apply the family-fiber obligations above. Since the fiber route already blocks host powersets, unrestricted separation, quotient equality, selected representatives, and external universes, the satisfaction-facing consumer has no coordinate from which host membership or an external satisfaction predicate can be read. □

Theorem 3668.24 (SetLike family-fiber FirstOrder ModelTheory lock)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every* $\mathsf{TypeLike}^{\uparrow}$ *family-fiber consumer that feeds a* $\mathsf{FirstOrder}^{\uparrow}$ *syntax read or a* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction read must expose exactly the rows* $M, Q, I, R, E, H, C, P, N$ *from the accepted packet. The lock cannot replace those rows by host membership, unrestricted comprehension, quotient set equality, selected representatives, external satisfaction, or ambient universe data.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFamilyFiberFirstorderModeltheoryLock`

Proof. Start with Definition 3668.23. It already routes every family-fiber satisfaction read through the displayed SetLike packet before any FirstOrder or ModelTheory consumer inspects membership or satisfaction.

Apply Definition 3668.20. A TypeLike family-fiber consumer receives the membership source, classifier, subset implication, restricted-comprehension row, extensional boundary, transport, replay, provenance, and local naming rows from S, not from a host membership relation or ambient family carrier.

Next apply Definition 3668.21. FirstOrder syntax and ModelTheory satisfaction inherit the same route, so no external satisfaction predicate, unrestricted comprehension rule, quotient set equality, selected representative, or universe datum is available as a replacement.

Finally use Definition 3668.18. The local ledger names only $M, Q, I, R, E, H, C, P, N$ and exports no coordinate outside the accepted packet, so the cross-consumer route is locked to those nine rows. □

Theorem 3668.25 (SetLike extensionality refusal obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensional row* E *is a classifier-boundary row over the displayed membership route* M, Q, I, R *and the structural route* H, C, P, N*. A downstream TypeLike, FirstOrder, or ModelTheory consumer may cite* E *only as that boundary row, not as host set extensionality, quotient equality, a powerset equality principle, unrestricted separation, or an external universe equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalityRefusalObligation`

Proof. Project the row E from Definition 3668.1. It is the carrier's extensionality-boundary coordinate, not a host equality object.

Use Definition 3668.62 and Definition 3668.71. Together they route every extensional read through $M, Q, I, R, E, H, C, P, N$ and rule out quotient equality, powersets, unrestricted separation, host set equality, and external inclusion predicates.

Finally apply Definition 3668.16. Since TypeLike, FirstOrder, and ModelTheory consumers must receive SetLike data through the same membership and classifier surface, they cannot reinterpret E as ambient extensionality or as an equality principle outside the concrete carrier rows. □

Theorem 3668.26 (SetLike bounded comprehension nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a restricted-comprehension read through* R *cannot export a powerset, unrestricted separation principle, host subset object, or universe of all subsets. It remains a bounded finite* BHist *row over* M, Q, I *and is accepted only together with* H, C, P, N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionNonescape`

Proof. Project the restricted-comprehension coordinate R from Definition 3668.1. The carrier describes R as a row admitting only classifiers already routed through M, Q, I.

Use Definition 3668.59 and Definition 3668.66. Both results keep the comprehension read inside the shared membership classifier and the subset implication route.

The structural rows H, C, P, N preserve and name that finite row. Since the packet has no coordinate for a powerset, unrestricted separation rule, host subset object, or universe of all subsets, restricted comprehension cannot export any of them. □

Theorem 3668.27 (SetLike membership route)**.** *Every downstream consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that reads membership, subset implication, restricted comprehension, or extensional comparison must factor through the finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Consequently sibling reads by $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *may use* $\mathsf{SetLike}^{\uparrow}$ *only through the carried membership route, and cannot replace it by an ambient powerset, unrestricted comprehension principle, host quotient of sets, or external satisfaction relation.*

Proof. Begin with an accepted carrier $S = (M, Q, I, R, E, H, C, P, N)$. A membership consumer can reach an element-in-set assertion only by reading the source row M and the classifier row Q.

Subset and comprehension reads then remain inside the same boundary. The row I supplies finite implication between displayed membership classifiers, and R admits restricted comprehension only when the classifier has already passed through M, Q, I.

Extensional reads are likewise bounded. The row E permits comparison only through the displayed membership classifier, so it does not introduce quotient equality or an ambient host set identity.

Finally replay the structural rows. The rows H, C, P, N preserve, replay, source, and name the same finite route, while $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ can cite the packet only through P and C. No sibling read therefore bypasses M, Q, I, R, E. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipRoute`

Theorem 3668.28 (SetLike kernel membership obligation)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *that reads membership must expose* M *as the* BHist *membership source and* Q *as the displayed membership classifier before any TypeLike, FirstOrder, or ModelTheory row is consumed. The read remains valid only through componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeKernelMembershipObligation`

Proof. Project the carrier and the membership carrier. Their membership-facing coordinates are exactly M, Q, H, C, P, N.

Apply the membership route theorem. It forces every downstream membership read through the displayed BHist source and classifier before sibling consumers can use the packet.

The structural tail is then read by projection. The rows H, C, P, N transport, replay, source, and locally name the same membership surface, so no host element relation, powerset row, or unrecorded classifier is introduced. □

Definition 3668.29 (SetLike root membership carrier)**.** *The root membership carrier of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet* $S = (M, Q, I, R, E, H, C, P, N)$ *is the finite* BHist *route*

$$M \longrightarrow Q \longrightarrow I \longrightarrow R \longrightarrow E,$$
$$H \longrightarrow C \longrightarrow P \longrightarrow N.$$

Here M *is the displayed membership source,* Q *is the classifier-based membership row,* I *records subset as implication of membership over* Q*,* R *records restricted comprehension only after* M, Q, I *are visible,* E *records the extensional boundary for the restricted result,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeRootMembershipCarrier`
Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipCarrier route readback`

Theorem 3668.30 (SetLike root membership transport)**.** *Every root membership read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet transports only the classifier-based membership surface of Definition 3668.29. A consumer that moves the read along* hsame *must replay* M, Q, I, R, E *through the displayed* Cont *row, preserve the* Pkg *provenance, and expose the same local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows before any TypeLike, FirstOrder, ModelTheory, or root-unblock consumer may use the membership judgment.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipTransport`

Proof. Project the carrier of Definition 3668.29. The only membership row is Q over the displayed BHist source M, while subset and comprehension rows are downstream of that classifier.

Use the existing membership, subset, and comprehension obligations in this surface. They keep membership reads on M, Q, subset reads on the implication row I, and restricted-comprehension reads on R, E.

The structural rows then provide the transport discipline. Componentwise hsame transport, Cont replay, Pkg provenance, and the local NameCert row preserve the same route, so no host powerset, quotient set equality, unrestricted separation principle, or carrier-equivalence replacement is introduced. □

Theorem 3668.31 (SetLike root subset ledger non-escape)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet, every subset ledger exported to a root consumer is the implication-of-membership row* I *from Definition 3668.29. The export must display classifier-based membership* M, Q*, restricted-comprehension boundary rows* R, E *when comprehension is read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows; it cannot escape to host inclusion, powerset equality, quotient set equality, or an unrestricted subset object.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootSubsetLedgerNonescape`

Proof. Read the subset coordinate I. Its source and target are classifier memberships over M, Q, so the ledger records implication of membership rather than an ambient inclusion relation.

When a comprehension consumer is present, read R only through the same M, Q, I route and then attach E as the displayed extensional boundary. This keeps restricted comprehension subordinate to the classifier ledger.

Finally project H, C, P, N. These rows transport, replay, source, and locally name the finite subset route, leaving no carrier coordinate for host inclusion, powerset equality, quotient equality, or unrestricted separation. □

Theorem 3668.32 (SetLike root comprehension scope)**.** *Every root comprehension consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must treat comprehension as the restricted row* R *scoped by classifier-based membership* M, Q *and subset implication* I*. The consumer may read the extensional boundary* E*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*, but it receives no unbounded predicate former, no host powerset, no quotient equality of sets, and no carrier-equivalence transport outside the displayed root route.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootComprehensionScope`

Proof. Start with the restricted-comprehension obligation already present in the child surface. It places R after the membership source, classifier, and subset implication rows.

Attach the extensional boundary E. The boundary describes the displayed restricted result only, and it remains tied to the same classifier-based membership route.

The structural tail H, C, P, N is then read without adding a new classifier or carrier. Thus the comprehension scope is exactly the finite SetLike root route and not an unrestricted separation, host powerset, quotient-set equality, or external subset object. □

Theorem 3668.33 (SetLike subset classifier obligation)**.** *Every accepted subset-classifier read of a* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must expose the* BHist *membership source* M*, membership classifier* Q*, subset implication row* I*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. It cannot replace the implication row by host inclusion, carrier-equivalence transport, quotient equality of sets, or an ambient powerset predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetClassifierObligation`

Proof. Start with the subset row I in the carrier. It is an implication over the displayed classifier Q and membership source M, not an external inclusion relation.

Use subset ledger exactness and classifier transport. These keep subset reads on the same M, Q, I route while H and C replay the displayed classifier rather than transporting across a different carrier.

Finally read P and N. Provenance and local naming expose the same finite packet to downstream consumers, so the subset classifier has no coordinate for powersets, quotient equality, host inclusion, or carrier-equivalence replacement. □

Theorem 3668.34 (SetLike comprehension ledger obligation)**.** *Every restricted-comprehension ledger read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must expose* $M, Q, I, R, E, H, C, P, N$ *as one finite* BHist *route. The row* R *is admitted only after the membership source, classifier, and subset implication rows are visible; the row* E *records the resulting extensional boundary;* H, C, P, N *preserve, replay, source, and locally name that ledger.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeComprehensionLedgerObligation`

Proof. Project the carrier to the comprehension row R. By the restricted-comprehension factorization theorem, R is bounded by M, Q, I before the structural rows are read.

Attach the extensional boundary row E. The extensional-boundary route keeps comparison of the displayed comprehension result inside the same classifier ledger.

The final projection reads H, C, P, N. Componentwise transport, continuation replay, provenance, and local NameCert expose the ledger as one finite packet, with no powerset, unrestricted separation principle, host subset object, quotient set equality, or carrier-equivalence transport. □

Theorem 3668.35 (SetLike root extensional boundary)**.** *Every extensionality read exported by the* $\mathsf{SetLike}^{\uparrow}$ *root route is bounded by the displayed membership source* M*, classifier* Q*, subset row* I*, restricted-comprehension row* R*, extensional boundary* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N*. The export gives no host powerset equality, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootExtensionalBoundary`

Proof. Project the root membership carrier of Definition 3668.29. Its extensional-facing row is E, and E is available only after the membership source, classifier, subset implication, and restricted-comprehension rows have been displayed.

Use the subset and comprehension obligations. The subset ledger keeps comparison on I over M, Q, while the comprehension ledger places R behind the same membership route and then attaches E as the boundary for the displayed restricted result.

Finally read H, C, P, N. These rows transport, replay, source, and locally name the same finite route, so the exported extensionality is the bounded root boundary rather than host powerset equality, quotient equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of carriers. □

Theorem 3668.36 (SetLike root finite route package)**.** *Every FirstOrder, ModelTheory, TypeLike, or root logic consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must read*

$$M, Q, I, R, E, H, C, P, N$$

as one finite BHist *route. The package exposes classifier-based membership, subset-as-implication, restricted comprehension, and the extensional boundary only through the displayed carrier of Definition 3668.29; it gives no host powerset, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootFiniteRoutePackage`

Proof. Start with Definition 3668.30. It places every root membership read on M, Q and keeps downstream consumers inside the displayed transport, replay, provenance, and local naming rows.

Next apply Definition 3668.31 and Definition 3668.32. The subset ledger is the implication row I over the classifier membership surface, and the comprehension row R is available only after M, Q, I have been displayed.

Finally apply Definition 3668.35. The extensional row E remains tied to the same restricted-comprehension route, while H, C, P, N preserve the finite packet for logic consumers. Since the existing root obligations already exclude host powersets, quotient set equality, unrestricted separation, carrier-equivalence replacement, and ambient carrier equality, the packaged consumer boundary excludes them as well. □

Theorem 3668.37 (SetLike root type-model membership factorization)**.** *Every TypeLike, FirstOrder, and ModelTheory consumer of the* $\mathsf{SetLike}^{\uparrow}$ *root route factors membership through* M, Q*, subset through* I*, restricted comprehension through* R, E*, and structural preservation through* H, C, P, N *before any logic-facing judgment is exported. The factorization gives no host powerset, quotient set equality, unrestricted separation, carrier-equivalence replacement, or ambient equality of set-like carriers.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootTypeModelMembershipFactorization`

Proof. Start with the membership transport theorem. It forces each membership read to expose the displayed source M and classifier Q before a TypeLike, FirstOrder, or ModelTheory consumer can use the judgment.

Next apply subset non-escape and comprehension scope. The subset read is the implication row I over classifier membership, while restricted comprehension is admitted only through R together with the extensional boundary E.

Use the root extensional boundary and finite-route package. They keep the resulting logic-facing judgment inside M, Q, I, R, E and preserve it only through componentwise hsame transport, displayed Cont replay, provenance, and local naming rows H, C, P, N.

The excluded host powerset, quotient equality, unrestricted separation, carrier-equivalence replacement, and ambient carrier equality are not coordinates of this route. Hence no logic or model-theory consumer can export them from the root packet. □

Theorem 3668.38 (SetLike membership classifier non-escape)**.** *Every TypeLike, FirstOrder, and ModelTheory consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *root packet must keep membership on the displayed source and classifier rows* M, Q *and must preserve the structural rows* H, C, P, N *before exporting any logic-facing judgment. The consumer cannot replace classifier membership by host elementhood, powerset membership, quotient equality, carrier-equivalence transport, or an ambient set equality outside the root route.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierNonescape`

Proof. First apply the root membership transport theorem. It keeps every membership read on the displayed BHist source M and classifier row Q before any logic consumer may use the judgment.

Next use subset ledger non-escape and comprehension scope. The subset row remains the implication ledger I over classifier membership, while restricted comprehension is admitted only through R, E after the same M, Q, I route has been displayed.

Finally apply the type-model membership factorization theorem. It packages TypeLike, FirstOrder, and ModelTheory reads through $M, Q, I, R, E, H, C, P, N$, so the structural rows preserve the classifier route rather than transporting it to host elementhood, powerset membership, quotient equality, carrier-equivalence replacement, or ambient set equality. □

Theorem 3668.39 (SetLike membership classifier transport boundary)**.** *Every TypeLike, FirstOrder, or ModelTheory consumer that transports membership from an accepted* $\mathsf{SetLike}^{\uparrow}$ *root packet* $S = (M, Q, I, R, E, H, C, P, N)$ *must preserve the displayed route*

$$M, Q, I, R, E, H, C, P, N.$$

The transported read cannot replace Q *by host elementhood, powerset membership, quotient set equality, carrier-equivalence transport, or ambient set equality.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierTransportBoundary`

Proof. Start with the membership transport and membership classifier non-escape theorems. They keep the membership source on the displayed BHist row M and the classifier on Q before any TypeLike, FirstOrder, or ModelTheory consumer can read the judgment.

Next apply subset non-escape and restricted comprehension scope. The subset row is still the displayed implication row I, and the comprehension route remains confined to R, E over the same M, Q, I prefix.

Finally use the root extensional boundary and type-model membership factorization theorem. These force transport to proceed through H, C, P, N after the displayed classifier route has been exposed, so none of the excluded host elementhood, powerset, quotient, carrier-equivalence, or ambient set-equality routes is available as a replacement for Q. □

Theorem 3668.40 (SetLike root membership obligation)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that attempts to read membership must display the route*

$$M, Q, H, C, P, N$$

before the membership judgment is available. The source row M supplies the carried membership surface, Q supplies the displayed membership classifier, and H, C, P, N transport, replay, source, and name the same route. The read exports no ambient powerset, unrestricted subset object, host equality of sets, quotient set equality, or universe-level membership relation.

Proof. Project the accepted carrier of Definition 3668.1. The only membership-facing source row is M, and the only classifier row attached to that source is Q.

Use Definition 3668.27. Any downstream membership read must factor through the displayed finite route before sibling consumers may cite the packet.

Finally read the structural rows. The rows H, C, P, N preserve, replay, source, and locally name the same membership surface, so the root consumer cannot replace it by a host powerset, quotient set equality, or universe-level membership relation. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipObligation`

Theorem 3668.41 (SetLike bounded comprehension family handoff)**.** *Every restricted-comprehension consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet reads a bounded family only through*

$$M, Q, I, R.$$

The membership source M, displayed classifier Q, implication ledger I, and restricted-comprehension row R together provide the bounded family handoff. The handoff does not supply unrestricted separation, a powerset carrier, quotient equality of subsets, selected representatives, or an external comprehension principle.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionFamilyHandoff`

Proof. Start from the restricted-comprehension row R in Definition 3668.1. It is a displayed packet coordinate and not a host comprehension operator.

Apply Definition 3668.53. The bounded family consumed by R is visible only after the membership source, classifier, and implication rows M, Q, I have been read.

The structural preservation clauses then keep the same bounded route under transport and replay. Since no row stores unrestricted separation, powersets, quotient subset equality, selected representatives, or an external comprehension principle, the consumer receives only the bounded family handoff. □

Theorem 3668.42 (SetLike TypeLike family nonescape)**.** *A* $\mathsf{TypeLike}^{\uparrow}$ *family consumer that cites a* $\mathsf{SetLike}^{\uparrow}$ *packet may use only the displayed rows*

$$M, Q, I, R, E, H, C, P, N$$

and cannot reinterpret the SetLike membership or family rows as a host universe, an ambient type of all subsets, quotient set equality, or external equality of family fibers.

Proof. Begin with the sibling read through the package row P. The $\mathsf{TypeLike}^{\uparrow}$ consumer receives a cited $\mathsf{SetLike}^{\uparrow}$ packet, not a replacement carrier for the host universe.

Apply Definition 3668.27 and Definition 3668.53. Membership, implication, bounded comprehension, and extensional comparison all remain inside M, Q, I, R, E before the structural rows are replayed.

Finally use H, C, P, N to transport, replay, source, and name that same route. These rows do not add a universe classifier, a powerset object, quotient set equality, or external equality of family fibers, so the TypeLike family read cannot escape the SetLike certificate surface. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeFamilyNonescape`

Theorem 3668.43 (SetLike root membership family exhaustion)**.** *Every root consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet that combines membership reads, bounded-comprehension family reads, and* $\mathsf{TypeLike}^{\uparrow}$ *family reads is exhausted by the displayed rows*

$$M, Q, I, R, E, H, C, P, N.$$

The combined route supplies membership, the bounded implication and comprehension handoff, extensional comparison, componentwise transport, continuation replay, package provenance, and local naming. It supplies no powerset, unrestricted separation, host universe, quotient set equality, selected representatives, or external family equality.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipFamilyExhaustion`

Proof. First apply Definition 3668.40. A root membership consumer must expose M, Q, H, C, P, N before it can read membership, and that route has no ambient powerset or host equality field.

Next apply Definition 3668.41. A restricted-comprehension consumer may add only the implication and bounded-comprehension rows I, R to the same displayed membership route.

Finally apply Definition 3668.42. A $\mathsf{TypeLike}^{\uparrow}$ family read may cite the SetLike packet only through $M, Q, I, R, E, H, C, P, N$ and cannot reinterpret those rows as a host universe, quotient set equality, selected representatives, or external equality of family fibers.

The three consumer routes therefore share the same finite row inventory. Their union is exactly $M, Q, I, R, E, H, C, P, N$, so no further family or set-forming coordinate is available to the root consumer. □

Definition 3668.44 (SetLike root carrier membership)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet* $S = (M, Q, I, R, E, H, C, P, N)$*, the root carrier membership read is the certified family route*

$$\begin{gathered} M \longrightarrow Q \longrightarrow I \longrightarrow R \longrightarrow E, \\ H \longrightarrow C \longrightarrow P \longrightarrow N. \end{gathered}$$

The row M is the finite BHist *membership source, Q is the classifier row over that source, I is subset implication between certified membership rows, R is bounded comprehension after the membership and subset rows are visible, E is the displayed extensional boundary, and H, C, P, N transport, replay, source, and locally name the same certified family packet.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeRootCarrierMembership`
Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootCarrierMembership route closed`

Theorem 3668.45 (SetLike root classifier transport)**.** *Every* hsame *or* Cont *transport of the root carrier membership read of Definition 3668.44 preserves the displayed membership classifier. A TypeLike, FirstOrder, ModelTheory, or root consumer that reads the transported packet receives* $M, Q, I, R, E, H, C, P, N$ *as one certified family route, and receives no host membership relation, ambient powerset, quotient equality of sets, selected representative, or external classifier.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootClassifierTransport`

Proof. Project the route of Definition 3668.44. The membership read is based on M and Q, while I, R, E are downstream ledger rows over the same classifier.

Use componentwise hsame transport and displayed Cont replay. These rows move the finite packet without replacing Q by a host relation or by a classifier over a different source.

Finally read P and N. Provenance and local naming expose the same certified family packet to downstream consumers, so TypeLike, FirstOrder, ModelTheory, and root consumers see only the transported SetLike route. □

Theorem 3668.46 (SetLike root NameCert obligations)**.** *The root* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *obligations for membership consumers are exactly carrier admission for Definition 3668.44, classifier transport for* M, Q*, bounded-comprehension ledger exposure for* I, R, E*, componentwise* hsame *and* Cont *preservation through* H, C*, provenance and local naming through* P, N*, and the boundary excluding host extensionality. Any root consumer that imports host set equality, ambient powersets, unrestricted separation, selected representatives, or an external membership classifier is outside the accepted packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootNameCertObligations`

Proof. Start with carrier admission. The definition presents the only membership carrier available to a root consumer: the finite route $M, Q, I, R, E, H, C, P, N$.

Apply classifier transport. It keeps membership reads on M, Q and makes H, C preserve the displayed classifier rather than a host membership relation.

Use the bounded-comprehension ledger rows. The subset and comprehension coordinates I, R, E are read only after the membership classifier is visible, so comprehension remains a finite BEDC ledger.

Finally attach provenance and local naming. Since P, N name only the displayed route, the accepted NameCert surface has no coordinate for host extensionality, powersets, unrestricted separation, selected representatives, or external classifiers. □

Theorem 3668.47 (SetLike membership classifier root unblock)**.** *Every downstream membership read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet exposes the membership source* M*, the classifier row* Q*, the transport row* H*, the replay row* C*, the provenance row* P*, and the local naming row* N *before the read may be consumed by a root dependency. The read therefore reaches TypeLike, FirstOrder, ModelTheory, or another root consumer only as the displayed finite membership classifier route, not as host membership, ambient powerset inclusion, unrestricted separation, quotient equality, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierRootUnblock`

Proof. Project Definition 3668.44. The membership source and classifier are exactly M and Q, and no other row supplies a membership predicate.

Apply Definition 3668.27 and Definition 3668.45. They force every membership-facing read through the displayed carrier route and preserve the classifier under hsame transport and Cont replay.

Finally read the structural rows H, C, P, N. Provenance and local naming expose the same finite route to downstream consumers, so the root dependency cannot replace it by host membership, powersets, unrestricted separation, quotient equality, or external satisfaction. □

Theorem 3668.48 (SetLike subset comprehension root unblock)**.** *Every root read of subset implication or restricted comprehension for an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet must pass through the finite route* M, Q, I, R, H, C, P, N*:* M, Q *expose the displayed membership classifier,* I *exposes subset implication between those classified rows,* R *exposes restricted comprehension only after* M, Q, I *are visible, and* H, C, P, N *transport, replay, source, and locally name the same packet. The route exports no ambient powerset, unrestricted comprehension principle, host subset relation, selected representative, or quotient equality of sets.*

Proof. Start with the membership route of Definition 3668.27. Any subset or comprehension read is downstream of the displayed membership classifier M, Q.

Use the rows I and R from Definition 3668.44. The implication row compares certified membership rows, and the restricted-comprehension row is available only after that comparison has been displayed.

Finally apply Definition 3668.46. The structural rows H, C, P, N keep the same finite route available to root consumers and leave no coordinate for ambient powersets, unrestricted comprehension, host subset relations, selected representatives, or quotient equality. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetComprehensionRootUnblock`

Theorem 3668.49 (SetLike extensionality root unblock)**.** *Every extensional comparison exported by an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet is only the displayed classifier boundary* E *over the membership rows* M, Q *with support from* H, C, P, N*. A downstream root consumer may read that boundary as the packet's extensional comparison route, but it may not identify it with host set equality, quotient equality of sets, selected representatives, an ambient inclusion predicate, or an external extensionality principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalityRootUnblock`

Proof. Project the extensional boundary of Definition 3668.44. The row E is the only comparison coordinate, and it is tied to the same displayed membership source M and classifier Q.

Use the SetLike membership route and the root NameCert obligations. They make E available only after the carrier rows have exposed the membership classifier and the structural preservation rows.

Thus a root consumer receives an extensional boundary for the accepted packet, not host set equality. The rows H, C, P, N preserve and name that boundary without adding quotient equality, selected representatives, ambient inclusion, or external extensionality. □

Theorem 3668.50 (SetLike root membership provenance exhaustion)**.** *Every TypeLike, FirstOrder, ModelTheory, or root consumer of accepted* $\mathsf{SetLike}^{\uparrow}$ *membership must exhaust the finite route* $M, Q, I, R, E, H, C, P, N$ *before exporting membership, subset, comprehension, or extensionality rows. In particular, the provenance row* P *and the local naming row* N *are required before any of the displayed membership classifier, subset implication, restricted-comprehension, or extensional boundary rows may leave the root packet.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipProvenanceExhaustion`

Proof. Project Definition 3668.44. The only accepted membership carrier is the displayed route $M, Q, I, R, E, H, C, P, N$, with P and N attached to the same packet as the membership and comparison rows.

Apply Definition 3668.45 and Definition 3668.46. These results keep the membership classifier on M, Q under hsame transport and Cont replay, while requiring provenance and local naming before downstream reads can treat the route as a NameCert surface.

Use Definition 3668.47, Definition 3668.48, and Definition 3668.49. The membership, subset, comprehension, and extensionality exports are exactly the rows M, Q, I, R, E supported by H, C, P, N.

Thus any consumer that omits P or N has not exhausted the root route. The omitted coordinate would be precisely where host membership, powersets, unrestricted separation, selected representatives, quotient set equality, or external satisfaction could enter, so the accepted packet refuses that export. □

Theorem 3668.51 (SetLike root membership nonescape ledger)**.** *Every TypeLike, FirstOrder, ModelTheory, or root consumer that has exhausted the accepted* $\mathsf{SetLike}^{\uparrow}$ *route* $M, Q, I, R, E, H, C, P, N$ *can export only the membership, subset, restricted-comprehension, and extensional-boundary rows already displayed in the SetLike packet. Such a consumer cannot introduce host membership, ambient powersets, unrestricted separation, quotient equality, selected representatives, or external satisfaction.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipNonescapeLedger`

Proof. Start from Definition 3668.44. The carrier route exposes exactly the membership source M, classifier Q, subset row I, restricted-comprehension row R, extensional boundary E, transport H, replay C, provenance P, and local naming N.

Apply the root classifier and NameCert rows, Definition 3668.45 and Definition 3668.46. They keep the membership classifier on M, Q and confine the accepted NameCert surface to the displayed finite route.

Use the three root-unblock exports: Definition 3668.47, Definition 3668.48, and Definition 3668.49. These are exactly the membership, subset, restricted-comprehension, and extensional-boundary rows available to downstream root consumers.

Finally use provenance exhaustion from Definition 3668.50. Once P, N have been read, no remaining carrier coordinate can supply host membership, ambient powersets, unrestricted separation, quotient equality, selected representatives, or external satisfaction. Hence the ledger is exhausted by the displayed SetLike packet. □

Theorem 3668.52 (SetLike extensional boundary classifier lock)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an extensional comparison read must expose the membership source* M*, the displayed classifier* Q*, the extensionality-boundary ledger* E*, and the structural rows* H, C, P, N *before any downstream consumer treats the comparison as available. The read is locked to the displayed classifier and cannot be replaced by host set equality, quotient equality, selected representatives, or an external inclusion predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryClassifierLock`

Proof. Project the carrier of Definition 3668.1. The extensional read is the row E, and its only membership data are the source row M and classifier row Q.

Use Definition 3668.27. The route forces any membership-facing consumer to expose M, Q, I, R, E before the structural rows are replayed.

Finally apply the local carrier rows H, C, P, N. Transport, replay, provenance, and naming keep E attached to the same displayed classifier, so the comparison is not host set equality, quotient equality, selected representatives, or external inclusion. □

Theorem 3668.53 (SetLike bounded comprehension transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, restricted-comprehension transport remains inside the finite route*

$$M, Q, I, R, H, C, P, N.$$

It may read a bounded family only after the membership source, classifier, and subset implication rows are visible, and componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming preserve that bounded route without importing powersets, unrestricted separation, quotient equality, or an external comprehension principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionTransport`

Proof. Project the restricted-comprehension row R from Definition 3668.1. The row is part of the accepted finite packet and is not an ambient comprehension operator.

Apply Definition 3668.27. The bounded family consumed by R must already pass through M, Q, I before comprehension data are visible.

Transport is then componentwise. The rows H, C, P, N preserve, replay, source, and name the same bounded route, so the transported comprehension remains inside M, Q, I, R, H, C, P, N and cannot introduce the excluded host-level surfaces. □

Theorem 3668.54 (SetLike first-order membership handoff)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *consumer that receives membership data from an accepted* $\mathsf{SetLike}^{\uparrow}$ *packet receives it only through the displayed finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Thus first-order syntax may consume SetLike membership, subset implication, restricted comprehension, and extensional comparison only after the carrier route has been displayed, and it cannot import an ambient powerset, unrestricted comprehension principle, quotient equality of sets, selected representatives, or external satisfaction relation.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFirstOrderMembershipHandoff`

Proof. Start with Definition 3668.27. It states the exact finite route available to downstream $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, and $\mathsf{ModelTheory}^{\uparrow}$ consumers.

Project the carrier of Definition 3668.1. The rows M, Q, I, R, E are the only membership, subset, comprehension, and extensional comparison rows available before structural replay.

Finally replay H, C, P, N. These rows transport, replay, source, and name the same finite membership route for the $\mathsf{FirstOrder}^{\uparrow}$ consumer, so first-order syntax receives no independent powerset, unrestricted comprehension, quotient equality, selected representative, or external satisfaction surface. □

Theorem 3668.55 (SetLike finite-family membership route)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a finite family of membership reads accepted by* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumers is routed componentwise through the same displayed packet*

$$M, Q, I, R, E, H, C, P, N.$$

Each family member first reads the BHist *membership source M and classifier Q, then consumes subset implication I, bounded comprehension R, and the extensional boundary E before the shared transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^\uparrow}$ *rows may be used. The family route exports no ambient powerset, unrestricted separation, quotient equality of sets, selected representative, host elementhood relation, or external satisfaction predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFiniteFamilyMembershipRoute`

Proof. Project the accepted carrier and read the family one displayed member at a time. Carrier admission supplies the same finite rows $M, Q, I, R, E, H, C, P, N$ for every member, with no additional row indexed by the family.

Apply the membership route and the bounded-comprehension transport theorem. They force every member read to enter through M, Q and to use I, R only as bounded rows over that same classifier.

Finally use the extensional boundary classifier lock. The row E compares only displayed SetLike data, while H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no powerset, unrestricted separation, quotient-set equality, selector, host elementhood, or satisfaction row, a finite family cannot export one. □

Theorem 3668.56 (SetLike TypeLike consumer handoff)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{TypeLike}^\uparrow$ *consumer receives SetLike membership as the classifier-bounded route M, Q together with the bounded-comprehension rows I, R, E and the structural tail H, C, P, N. The handoff preserves both the displayed membership classifier and the bounded-comprehension ledger; it cannot replace them by a host type predicate, an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, or a global elementhood oracle.*

Proof. Start with the membership route for downstream TypeLike, FirstOrder, and ModelTheory consumers. It gives the TypeLike consumer access to SetLike data only after the membership source, classifier, subset, comprehension, extensional, transport, replay, provenance, and local naming rows have been displayed.

Use the extensional boundary classifier lock to keep Q attached to the same source row M. Then use bounded-comprehension transport to keep I, R, E attached to that classifier route rather than an ambient type predicate or powerset.

The structural rows H, C, P, N complete the handoff by transporting, replaying, sourcing, and naming the same finite packet. Because no host type predicate, powerset, unrestricted separation, quotient equality, selector, or global elementhood coordinate occurs in the accepted carrier, the TypeLike consumer receives only the preserved SetLike route. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeConsumerHandoff`

Theorem 3668.57 (SetLike ModelTheory satisfaction boundary)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{ModelTheory}^\uparrow$ *satisfaction consumer may read SetLike data only through the FirstOrder-facing finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The boundary supplies displayed membership, classifier-bounded subset and comprehension rows, extensional comparison, componentwise transport, continuation replay, provenance, and local naming. It refuses external satisfaction predicates, powerset semantics, unrestricted separation, quotient equality of sets, selected representatives, host universes, and completed model-theoretic semantics.

Proof. First route the consumer through the FirstOrder membership handoff. The syntax-facing read exposes $M, Q, I, R, E, H, C, P, N$ before satisfaction-facing data can be consumed.

Next apply the root unblock for ModelTheory satisfaction. It keeps satisfaction downstream of the FirstOrder syntactic carrier and therefore downstream of the displayed SetLike membership classifier and bounded-comprehension ledger.

Finally use the finite-family route for any displayed family of satisfaction reads. Each member still reads the same rows, and the carrier has no external satisfaction predicate, powerset semantics, unrestricted separation, quotient equality, selector, host universe, or completed semantic closure coordinate. The ModelTheory boundary is therefore the finite BEDC packet itself. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeModelTheorySatisfactionBoundary`

Theorem 3668.58 (SetLike subset implication obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an accepted subset read is an implication row* I *over the displayed membership classifier* Q *and the membership-source row* M*. The read is admitted as finite* BHist *data only when the componentwise* hsame *row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row* N *all keep the same implication surface visible; it cannot import host subset equality or a global inclusion relation outside those rows.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetImplicationObligation`

Proof. Project the carrier and isolate the subset row I. The only membership facts available to I are the source row M and the displayed classifier row Q.

Route the implication through the existing membership classifier. A consumer may read that an accepted membership assertion under one displayed row implies an accepted membership assertion under another, but the route is still the finite M, Q, I ledger.

The structural rows close the boundary. Transport by H, replay by C, provenance by P, and local naming by N preserve the same finite implication row, so no host subset equality or external inclusion predicate enters the packet. □

Theorem 3668.59 (SetLike bounded comprehension obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a comprehension read is admitted only through the restricted-comprehension row* R *bounded by a displayed certified family over* M, Q, I*. The resulting* BHist *row remains tied to componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming* N*, and therefore does not create a powerset or unrestricted separation principle.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionObligation`

Proof. Begin with the restricted-comprehension coordinate R. By Definition 3668.1, R is not an ambient class former; it is one row of the accepted finite packet.

The bound is supplied by the existing membership route. The source row M, classifier row Q, and implication row I determine which displayed family may be read before R is exposed.

Finally replay the structural rows. The rows H, C, P, N transport, replay, source, and name the same bounded family row, so the comprehension read remains finite BEDC data and never becomes a powerset, unrestricted separation rule, or host universe of subsets. □

Theorem 3668.60 (SetLike extensional boundary obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, extensional comparison is a boundary ledger* E *over displayed membership rows rather than quotient equality of sets. The comparison is accepted as* BHist *data*

only when its membership evidence is routed through M, Q, I, R and its stability is preserved by componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row N.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryObligation`

Proof. Project the extensionality row E. It is a boundary ledger inside the carrier, not a quotient constructor and not a host equality of sets.

Use the existing membership classifier to supply the comparison evidence. A displayed extensional read compares membership rows already exposed through M, Q and routed through the subset and bounded-comprehension rows I, R.

The structural rows keep the boundary exact. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed membership ledger, so no quotient equality, representative selection, or ambient set identity is exported. □

Definition 3668.61 (SetLike membership boundary subcarrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the membership-boundary subcarrier is the finite prefix

$$S_{\partial} = (M, Q, I, R, E).$$

It cuts out the membership source, classifier, subset implication, restricted-comprehension row, and extensionality-boundary ledger before transport, replay, provenance, and local naming are consumed. The subcarrier has no coordinate for powersets, unrestricted separation, quotient equality of sets, selected representatives, or host universe membership.

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeMembershipBoundarySubcarrier`

Theorem 3668.62 (SetLike extensional boundary route)**.** *Every extensional read of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$ must first factor through the membership-boundary subcarrier $S_{\partial} = (M, Q, I, R, E)$ and then through H, C, P, N. Thus an extensional consumer may compare displayed set rows only after the membership source, classifier, subset implication, restricted-comprehension boundary, extensionality ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row have all remained visible.*

Proof. Project the membership-boundary subcarrier of Definition 3668.61. It exposes exactly M, Q, I, R, E before any structural replay is used.

The extensional comparison is the row E. By Definition 3668.60, the comparison evidence must already be routed through the membership source, classifier, subset implication, and restricted-comprehension rows.

Finally replay the structural rows H, C, P, N. They transport, replay, source, and name the same finite boundary, so an extensional consumer cannot bypass M, Q, I, R, E or replace the displayed boundary by host set equality. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryRoute`

Theorem 3668.63 (SetLike subset implication witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, the implication row I is the only subset witness available over the displayed membership classifier Q and membership source M. Any accepted subset read must keep I attached to M, Q, H, C, P, N and cannot replace it by powersets, unrestricted separation, quotient equality of sets, host subset equality, selected representatives, or an external inclusion predicate.*

Proof. Project the carrier of Definition 3668.1. The subset coordinate is exactly the row I, and the only membership data available to it are the source row M and displayed classifier row Q.

Use the membership route and subset obligation, Definition 3668.27 and Definition 3668.58. These force every subset consumer to read implication as finite carrier data over M, Q, I before the structural rows are replayed.

Finally replay through H, C, P, N. Since the carrier has no powerset, unrestricted separation, quotient-equality, selected-representative, host-subset, or external-inclusion coordinate, I is the only subset witness exposed by the packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetImplicationWitness`

Theorem 3668.64 (SetLike bounded comprehension witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the restricted-comprehension row* R *is bounded by the displayed membership source* M*, classifier* Q*, and subset-implication row* I*. The row* R *witnesses only bounded comprehension inside that carrier route and cannot export powersets, unrestricted separation, quotient equality of sets, selected representatives, host subset objects, or an external comprehension principle.*

Proof. Project the carrier of Definition 3668.1. The comprehension coordinate is the row R, and its definition admits only classifiers already routed through M, Q, I.

Apply bounded-comprehension obligation and restricted-comprehension factorization, Definition 3668.59 and Definition 3668.75. They identify M, Q, I as the required bound before R may be read.

The structural rows H, C, P, N preserve and name that same bounded route. Because no other carrier row stores powersets, unrestricted separation, quotient equality, selected representatives, host subset objects, or external comprehension, R remains only the displayed bounded-comprehension witness. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionWitness`

Theorem 3668.65 (SetLike extensional boundary witness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensionality row* E *is only a classifier boundary over the displayed membership route* M, Q, I, R*. It may witness that two displayed set rows have the same carried classifier behavior, but it does not export host set equality, powersets, unrestricted separation, quotient equality of sets, selected representatives, or an equality oracle outside* $M, Q, I, R, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 3668.1. The row E is an extensionality-boundary ledger, not a quotient constructor or host equality coordinate.

Use the extensional boundary obligation and route, Definition 3668.60 and Definition 3668.62. They force every extensional read through the membership source, classifier, subset implication, and restricted-comprehension rows before H, C, P, N are replayed.

Finally cite classifier extensional nonescape, Definition 3668.71. The displayed classifier boundary remains inside $M, Q, I, R, E, H, C, P, N$, so the refused powerset, unrestricted-separation, quotient-equality, selected-representative, host-equality, and equality-oracle readings cannot be exported. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryWitness`

Theorem 3668.66 (SetLike subset-comprehension stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, subset implication through* I *and restricted comprehension through* R *share the same membership classifier* Q *over the membership source* M*. Any consumer that reads both rows must keep the common classifier visible through* H, C, P, N*, and the read cannot introduce a powerset row, unrestricted separation row, host subset equality, or an external comprehension principle.*

Proof. Start from the accepted carrier and its subcarrier $S_{\partial} = (M, Q, I, R, E)$. The rows I and R both sit inside the same finite boundary and have access only to the membership source M and classifier Q.

The subset implication row I reads implication between displayed membership classifiers. The restricted-comprehension row R may bind only a classifier already routed through M, Q, I; it therefore cannot choose a second classifier outside the packet.

The structural rows close the stability claim. Componentwise transport H, replay C, provenance P, and local naming N preserve the same classifier route for both I and R, so the combined read remains finite BHist data and does not become a powerset or unrestricted separation surface. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetComprehensionStability`

Theorem 3668.67 (SetLike membership carrier stability)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a membership read remains on the same finite carrier when transported by the componentwise* hsame *row* H *and replayed by* Cont *row* C*. More precisely, if a consumer reads membership through* M, Q *before transport, then every transported read is again a membership-source and classifier read through* M, Q, H, C, P, N*, not a new host element relation, powerset coordinate, or external equality predicate.*

Proof. Project the carrier of Definition 3668.1. The membership source M and classifier Q are the only rows from which an element-in-set read can be opened.

Apply the componentwise transport row H. Since H is recorded for M, Q, I, R, E together with the structural rows, it preserves the displayed membership source and classifier rather than replacing them by an ambient relation.

Finally replay through C and name the route through P, N. The continuation can expose only the transported M, Q rows, so the transported consumer remains inside the same finite BHist carrier and no host powerset or external equality predicate enters. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierStability`

Theorem 3668.68 (SetLike classifier transport exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, classifier transport is exact on the displayed membership surface: subset implication through* I*, restricted comprehension through* R*, and extensional comparison through* E *may use only the classifier* Q *transported by* H *and replayed by* C*. The transported classifier cannot be weakened to a proof-irrelevant host proposition or strengthened to an unrestricted comprehension classifier.*

Proof. Start with the classifier row Q. The subset, comprehension, and extensionality rows I, R, E are all described in Definition 3668.1 as rows over the same membership source and classifier.

Transport by H is componentwise. It therefore moves Q together with I, R, E and leaves no independent classifier coordinate from which a consumer could read a stronger or weaker predicate.

Replay by C exposes the same classifier route to downstream packets. Because P, N source and name only this finite packet, the transported classifier is exact: it remains the displayed membership classifier and does not become proof irrelevance, unrestricted separation, or a host proposition. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierTransportExactness`

Theorem 3668.69 (SetLike ledger scope closure)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the visible ledger scope for membership, subset implication, restricted comprehension, and extensional boundary reads is exactly the finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Every $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *consumer that leaves this route lacks the carrier rows needed for an accepted read, and every consumer that stays on the route has the full ledger scope required by the chapter.*

Proof. The forward direction is carrier projection. The rows M, Q, I, R, E are the complete membership-boundary prefix, and the rows H, C, P, N are the complete transport, replay, provenance, and local naming tail.

For nonescape, suppose a consumer leaves this route. It then omits either the membership source, the classifier, one of the subset/comprehension/extensional boundary rows, or one of the structural rows that preserves and names them. By Definition 3668.10, such a consumer lacks a required local obligation.

For closure inside the route, keep all nine rows visible. The earlier membership route, subset implication, bounded comprehension, extensional boundary, and subset ledger theorems each consume only these rows, so the displayed $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ ledger scope is exhausted by the route and no additional host coordinate is required. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeLedgerScopeClosure`

Falsifiable prediction: A consumer that claims a $\mathsf{SetLike}^{\uparrow}$ membership, subset, restricted-comprehension, or extensionality read while omitting the membership source, classifier, subset implication, restricted comprehension, extensionality boundary, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 3668.1.

Independence witness: typelike, firstorder, modeltheory: $\mathsf{SetLike}^{\uparrow}$ is not bijective to any listed sibling because it carries membership source, classifier, subset implication, restricted comprehension, extensionality boundary, componentwise transport, continuation replay, provenance, and local NameCert rows. TypeLikeUp adds displayed indexed fibers beyond membership; FirstOrderUp adds syntax and deduction rows without membership-source extensionality; ModelTheoryUp adds satisfaction over FirstOrderUp without a general finite membership carrier.

Theorem 3668.70 (SetLike subset ledger obligation)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the subset ledger exposed to downstream consumers consists exactly of membership-carrier rows* M, Q*, subset-as-membership-implication rows* I*, restricted-comprehension provenance rows* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, and the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *rows named through* P, N*. The ledger exports no host set extensionality, unrestricted comprehension, powerset row, quotient equality of sets, or external inclusion predicate.*

Proof. Project the accepted carrier in Definition 3668.1. The membership source M and displayed classifier Q are the only rows from which membership facts can be read.

Use the subset obligation of Definition 3668.58. The subset row I is an implication between displayed membership classifiers over M, Q, not a host inclusion relation.

Add the restricted-comprehension and extensional boundary controls from Definition 3668.59 and Definition 3668.60. The row R records only bounded comprehension provenance, while E remains a boundary ledger and does not export host set extensionality.

Finally replay Definition 3668.27. The structural rows H, C, P, N preserve and name the same finite route, so the subset ledger seen by downstream TypeLike, FirstOrder, and ModelTheory consumers contains no powerset, unrestricted comprehension, quotient equality, or external inclusion coordinate. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeSubsetLedgerObligation`

Theorem 3668.71 (SetLike classifier extensional nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the extensional row* E *can compare only displayed membership rows over* M, Q, I, R*. It cannot export quotient equality of sets, powersets, unrestricted separation, host set equality, or an external inclusion predicate; any accepted extensional read must keep* E *inside the structural route* H, C, P, N*.*

Proof. Project E from the carrier of Definition 3668.1. The carrier assigns E to the extensionality-boundary ledger, not to a quotient or host equality coordinate.

Use the extensional boundary route of Definition 3668.62. That theorem forces every extensional read through the membership-boundary subcarrier $S_{\partial} = (M, Q, I, R, E)$ before the structural rows are replayed.

Finally apply Definition 3668.68 and Definition 3668.70. Classifier transport stays on the displayed Q surface, and the downstream ledger exposes no powerset, unrestricted comprehension, quotient equality, or host inclusion coordinate. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeClassifierExtensionalNonescape`

Definition 3668.72 (SetLike extensional-subset carrier)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier*

$$S = (M, Q, I, R, E, H, C, P, N),$$

the extensional-subset subcarrier is the finite displayed comparison packet

$$S_{\subseteq\partial} = (M, Q, I, E, H, C, P, N).$$

Here M, Q *expose the membership source and classifier,* I *records subset-as-classifier implication,* E *records the extensional comparison boundary, and* H, C, P, N *transport, replay, source, and locally name the same displayed comparison. The subcarrier has no coordinate for host set equality, powersets, unrestricted separation, quotient equality of sets, or an external inclusion predicate.*

Lean4 (definition) `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetCarrier`

Theorem 3668.73 (SetLike extensional-subset NameCert obligations)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, an accepted extensional-subset read through* $S_{\subseteq\partial}$ *requires exactly membership-source admission through* M*, classifier exposure through* Q*, subset implication through* I*, extensional-boundary exposure through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *naming through* N*. It cannot require or export host set equality, powersets, unrestricted separation, quotient equality of sets, or an external inclusion predicate.*

Proof. Project the subcarrier of Definition 3668.72. The only comparison rows available before the structural tail are the membership source, classifier, subset implication, and extensional boundary.

Use the membership route and subset ledger obligation, Definition 3668.27 and Definition 3668.70. These force subset reads to remain implications over the displayed classifier rather than external inclusion predicates.

Apply classifier extensional nonescape and ledger scope closure, Definition 3668.71 and Definition 3668.69. The row E stays a boundary ledger over displayed membership rows, while H, C, P, N preserve and name the same route. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetNameCertObligations`

Theorem 3668.74 (SetLike extensional-subset transport)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, componentwise* hsame *transport and displayed* Cont *replay preserve the extensional-subset route*

$$M, Q, I, E \longmapsto H, C, P, N.$$

Thus a transported extensional-subset read remains the same classifier route and cannot become host set equality, a powerset read, unrestricted separation, quotient equality of sets, or an external inclusion predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalSubsetTransport`

Proof. Start with the extensional-subset subcarrier of Definition 3668.72. It keeps only M, Q, I, E before structural replay, so the displayed classifier and implication rows determine the comparison.

Apply classifier transport exactness. The row H transports the displayed classifier surface componentwise, and C replays the same subset and extensional-boundary route rather than introducing a second inclusion relation.

Finally use the extensional-subset obligations of Definition 3668.73. Provenance and local naming through P, N keep the transported read inside the finite BHist packet, excluding host equality, powersets, unrestricted separation, quotient equality, and external inclusion. □

Theorem 3668.75 (SetLike restricted comprehension membership factorization)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, every restricted-comprehension read through* R *first factors through the membership source* M*, displayed classifier* Q*, and subset implication row* I*. Only after this factorization may the read use* H, C, P, N *for transport, replay, provenance, and local naming; it cannot create a powerset row, unrestricted separation principle, host subset object, or external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRestrictedComprehensionMembershipFactorization`

Proof. Start with Definition 3668.59. It admits R only as a restricted-comprehension row bounded by a displayed certified family over M, Q, I.

The common classifier is fixed by Definition 3668.66. Subset implication through I and restricted comprehension through R share the membership classifier Q over the source M, so a comprehension read cannot skip the membership route.

Apply bounded comprehension nonescape and ledger scope closure, Definition 3668.26 and Definition 3668.69. The structural rows H, C, P, N preserve the displayed factorization and add no powerset, unrestricted separation, host subset object, or external comprehension coordinate. □

Theorem 3668.76 (SetLike TypeLike FirstOrder ModelTheory consumer boundary)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, *or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must cite the SetLike route through*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Such a consumer cannot replace the route by external satisfaction, host membership, quotient equality of sets, a powerset row, unrestricted comprehension, or a host universe of sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeLikeFirstOrderModelTheoryConsumerBoundary`

Proof. Use the membership route theorem Definition 3668.27. It already states that TypeLike, FirstOrder, and ModelTheory sibling reads may use $\mathsf{SetLike}^{\uparrow}$ only through the carried membership route.

Next apply the subset ledger obligation and classifier extensional nonescape, Definition 3668.70 and Definition 3668.71. These results keep subset, comprehension, and extensionality reads on M, Q, I, R, E before H, C, P, N are used.

Finally invoke Definition 3668.69. A consumer leaving the displayed route lacks the required carrier rows, while a consumer staying on it receives the complete local ledger. Therefore external satisfaction, host membership, quotient equality, powersets, unrestricted comprehension, and host universes are outside the accepted SetLike consumer boundary. □

Theorem 3668.77 (SetLike root unblock for FirstOrder signatures)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *signature consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must factor membership, subset, restricted-comprehension, and extensional-boundary reads through the displayed route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The signature consumer may use this route as the SetLike membership base for syntactic symbols, arities, relation slots, and bounded classifier reads, but it cannot replace the route by host set membership, an ambient universe of sets, a powerset row, unrestricted comprehension, or quotient equality of sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockFirstOrderSignature`

Proof. Project the accepted carrier of Definition 3668.1. The only SetLike data available to a signature consumer are the membership source, classifier, subset implication, restricted comprehension, extensional boundary, transport, replay, provenance, and local naming rows.

Use the membership route theorem Definition 3668.27. It forces FirstOrder sibling reads through the carried membership route before syntax can consume SetLike membership, subset, comprehension, or extensional comparison data.

Apply classifier transport and extensional boundary routing, Definition 3668.12 and Definition 3668.62. The syntactic consumer receives transported displayed classifiers and displayed boundary ledgers, not host elementhood, ambient universes, powersets, unrestricted comprehension, or quotient equality. □

Theorem 3668.78 (SetLike root unblock for ModelTheory satisfaction)**.** *Every* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *may read SetLike data only through the classifier-based membership route and the* $\mathsf{FirstOrder}^{\uparrow}$ *syntactic carrier that consumes that route. The satisfaction-facing read must keep* $M, Q, I, R, E, H, C, P, N$ *visible, and cannot import an external satisfaction relation, quotient equality of sets, host powerset, unrestricted comprehension, or ambient host universe.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockModelTheorySatisfaction`

Proof. Start from Definition 3668.1. The carrier has no satisfaction row of its own; it exposes only the SetLike membership source, classifier, subset, restricted-comprehension, extensional-boundary, transport, replay, provenance, and local-name rows.

Use Definition 3668.27. A ModelTheory consumer can cite SetLike only through the same classifier-based route already available to TypeLike and FirstOrder siblings, so satisfaction cannot read an independent element relation.

Now apply Definition 3668.12 and Definition 3668.62. Transport preserves the displayed classifier route, and extensional reads remain boundary ledgers over M, Q, I, R, E before H, C, P, N are replayed. The satisfaction consumer therefore obtains SetLike data only through the FirstOrder syntactic carrier built on that route, with no external satisfaction relation, quotient equality, host powerset, unrestricted comprehension, or ambient universe. □

Theorem 3668.79 (SetLike kernel membership source totality)**.** *Every accepted* $\mathsf{SetLike}^{\uparrow}$ *membership consumer must expose the full kernel membership source before any TypeLike, FirstOrder, or ModelTheory read is available. Concretely, the consumer must display M as the finite* BHist *membership source, Q as the membership classifier row, H as the componentwise* hsame *transport ledger, C as the displayed* Cont *replay, P as the* Pkg *provenance row, and N as the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row.*

Proof. Start with the carrier of Definition 3668.1. The carrier lists $M, Q, I, R, E, H, C, P, N$, so any accepted membership read has M, Q, H, C, P, N available as displayed rows.

Use the membership route theorem Definition 3668.27. It forces TypeLike, FirstOrder, and ModelTheory consumers through the carried membership route before they may read typed, syntactic, or satisfaction-facing data.

Finally apply the consumer boundary of Definition 3668.76. A consumer lacking M, Q, H, C, P, N has left the displayed route, while a consumer staying inside the route exposes exactly the kernel membership source and structural rows. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeKernelMembershipSourceTotality`

Theorem 3668.80 (SetLike restricted comprehension kernel scope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, the restricted-comprehension row R is admitted only over the displayed membership source M, classifier Q, and subset implication row I. The row R cannot export a powerset, unrestricted separation, a host subset family, or an external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRestrictedComprehensionKernelScope`

Proof. Apply the restricted-comprehension membership factorization theorem Definition 3668.75. It already factors every R read through M, Q, I before structural replay.

Use bounded comprehension nonescape from Definition 3668.26. This blocks powerset, unrestricted separation, host subset objects, and external comprehension predicates from the comprehension row.

The ledger scope closure theorem Definition 3668.69 then repacks the result through H, C, P, N. Thus R remains a kernel-scoped row over M, Q, I and not an escaping separation principle. □

Theorem 3668.81 (SetLike extensional ledger kernel nonescape)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier $S = (M, Q, I, R, E, H, C, P, N)$, extensional comparison through E is a visible ledger over*

the membership, classifier, subset, and restricted-comprehension rows. It is not quotient equality of sets, host extensionality, selected representatives, or an equality oracle outside $M, Q, I, R, E, H, C, P, N$.

Proof. Begin with the extensional boundary route Definition 3668.62. It routes extensional reads through the membership-boundary subcarrier before structural replay.

Apply classifier extensional nonescape from Definition 3668.71. The extensional comparison remains attached to the displayed classifier and cannot become quotient equality or host extensionality.

Finally use the TypeLike FirstOrder ModelTheory consumer boundary Definition 3668.76. Downstream consumers receive the visible ledger $M, Q, I, R, E, H, C, P, N$ and no equality oracle outside it. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalLedgerKernelNonescape`

Theorem 3668.82 (SetLike extensional-boundary consumer exactness)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, any* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of a SetLike extensional comparison must read exactly*

$$M, Q, I, R, E, H, C, P, N.$$

The row E *remains the displayed extensional-boundary ledger over the membership, classifier, subset, and restricted-comprehension route. It cannot be replaced by host set equality, quotient equality, powerset equality, unrestricted separation, or any equality oracle outside the accepted packet.*

Proof. Begin with the extensional boundary route Definition 3668.62. It makes E a boundary row over M, Q, I, R before the structural tail H, C, P, N is replayed.

Use the classifier transport route Definition 3668.12. Transport by H preserves the displayed classifier route and does not substitute a host equality predicate, quotient equality, powerset equality, or unrestricted separation row for E.

Apply ledger scope closure Definition 3668.69. The membership, classifier, subset, comprehension, and extensional rows are repacked with H, C, P, N as one finite BEDC ledger.

Finally use the TypeLike FirstOrder ModelTheory consumer boundary Definition 3668.76. A downstream consumer that reads the extensional comparison therefore receives exactly $M, Q, I, R, E, H, C, P, N$, while any attempted replacement for E leaves the accepted $\mathsf{SetLike}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeExtensionalBoundaryConsumerExactness`

Theorem 3668.83 (SetLike membership carrier root surface)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the membership-facing root surface seen by* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, and* $\mathsf{ModelTheory}^{\uparrow}$ *is exactly the finite* BHist *carrier*

$$S_{\in} = (M, Q, H, C, P, N).$$

Here M *is the displayed membership source,* Q *is the membership classifier,* H *is componentwise* hsame *transport,* C *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ *row. The surface exports no powerset, host elementhood, quotient equality of sets, unrestricted comprehension, selected representatives, or external satisfaction predicate.*

Proof. Project the accepted carrier and the membership carrier of Definition 3668.2. The only rows retained in $S_\in$ are M, Q, H, C, P, N.

Use the membership route theorem Definition 3668.27. It forces TypeLike, FirstOrder, and ModelTheory consumers to enter through the carried membership source and classifier before any sibling read is available.

Finally apply membership carrier stability and the consumer boundary, Definition 3668.67 and Definition 3668.76. Componentwise transport and continuation replay keep the read inside the same BHist route, while provenance and local naming certify that no host element relation, powerset, quotient equality, unrestricted comprehension, selected representative, or external satisfaction row has been inserted. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipCarrierRootSurface`

Theorem 3668.84 (SetLike membership classifier stability)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the membership classifier* Q *remains stable under the complete displayed route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Subset implication through I*, bounded comprehension through* R*, and extensional comparison through* E *may consume* Q *only as the same classifier over the same* BHist *source* M*. A* $\mathsf{FirstOrder}^\uparrow$*,* $\mathsf{TypeLike}^\uparrow$*, or* $\mathsf{ModelTheory}^\uparrow$ *consumer therefore receives a stable SetLike classifier, not a proof-irrelevant host predicate, ambient powerset classifier, unrestricted separation rule, or external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeMembershipClassifierStability`

Proof. Start with classifier transport exactness, Definition 3668.68. It states that I, R, E consume the displayed membership classifier Q and that transport by H does not replace it with a second predicate.

Next use subset-comprehension stability and extensional ledger nonescape, Definition 3668.66 and Definition 3668.81. These keep the subset, comprehension, and comparison rows attached to the same M, Q route.

The structural tail is supplied by C, P, N. By the TypeLike, FirstOrder, and ModelTheory consumer boundary, any downstream read that omits this tail is not an accepted SetLike read, and any downstream read that keeps it receives only the stable displayed classifier. □

Theorem 3668.85 (SetLike bounded comprehension ledger)**.** *For every accepted* $\mathsf{SetLike}^\uparrow$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the bounded-comprehension ledger is the finite route*

$$M, Q, I, R \longmapsto E, H, C, P, N.$$

The row R *records only comprehension over classifiers already exposed by the membership source* M*, membership classifier* Q*, and subset-implication row* I*; the extensional ledger* E *can compare only the displayed result. Thus FirstOrder syntax, TypeLike family data, and ModelTheory satisfaction may consume the ledger only through* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, with no powerset, unrestricted separation, host subset object, quotient set equality, or external comprehension predicate.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeBoundedComprehensionLedger`

Proof. Apply restricted-comprehension membership factorization, Definition 3668.75. It factors every R read through M, Q, I before the structural rows are available.

Use bounded comprehension nonescape and the extensional boundary route, Definition 3668.26 and Definition 3668.62. The first keeps R from becoming a powerset or unrestricted separation principle; the second keeps any comparison of the displayed result inside E over the same classifier route.

Finally replay ledger scope closure and the sibling consumer boundary, Definition 3668.69 and Definition 3668.76. The FirstOrder, TypeLike, and ModelTheory consumers therefore receive only the finite BEDC ledger $M, Q, I, R, E, H, C, P, N$ and no host comprehension or quotient-set coordinate. □

Theorem 3668.86 (SetLike TypeLike signature scope)**.** *Let $S = (M, Q, I, R, E, H, C, P, N)$ be an accepted $\mathsf{SetLike}^{\uparrow}$ packet. Any TypeLike, FirstOrder, or ModelTheory consumer that reads S as a set-like signature must first read the membership source row M, membership classifier row Q, subset implication row I, restricted-comprehension row R, extensional boundary row E, componentwise transport row H, continuation row C, provenance row P, and local $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$ row N. The signature read exports no ambient powerset, unrestricted separation, quotient equality of sets, selected representative, or host equality oracle outside $M, Q, I, R, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypelikeSignatureScope`

Proof. Start with the membership route of Definition 3668.27. It makes M and Q the visible source of every membership read before a typed, syntactic, or satisfaction-facing consumer can inspect the packet.

Next apply restricted comprehension factorization from Definition 3668.75 and the kernel scope theorem Definition 3668.80. These force R to remain a row over M, Q, I rather than a powerset or unrestricted separation principle.

Finally use the consumer boundary of Definition 3668.76 together with the extensional ledger nonescape theorem Definition 3668.81. The downstream TypeLike, FirstOrder, and ModelTheory reads receive exactly $M, Q, I, R, E, H, C, P, N$, and no external set-equality or comprehension coordinate is available. □

Theorem 3668.87 (SetLike typed-comprehension kernel envelope)**.** *For every accepted $\mathsf{SetLike}^{\uparrow}$ carrier $S = (M, Q, I, R, E, H, C, P, N)$, a typed-comprehension read is accepted only as the finite BEDC envelope*

$$M, Q, I, R \longmapsto H, C, P, N.$$

Here M is the BHist membership source, Q is the classifier-bounded membership row, I is subset implication over that classifier, and R is restricted comprehension over the same displayed rows. The envelope cannot be read as an ambient powerset, a host universe, unrestricted separation, quotient equality of sets, or an external comprehension predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypedComprehensionKernelEnvelope`

Proof. Project the carrier. The rows M, Q, I, R are the only membership, classifier, subset, and restricted-comprehension data before the structural tail.

Use restricted-comprehension membership factorization and restricted-comprehension kernel scope. These results force R to remain bounded by M, Q, I rather than becoming an unrestricted separation or powerset row.

Finally replay the structural route through H, C, P, N. Since no carrier coordinate stores an ambient powerset, host universe, quotient equality, or external comprehension predicate, the accepted typed-comprehension read is exactly the finite BEDC envelope. □

Theorem 3668.88 (SetLike FirstOrder membership substitution boundary)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{FirstOrder}^{\uparrow}$ *substitution consumer may substitute membership only through the displayed classifier route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

The substitution boundary is classifier-bounded over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, and it cannot import host membership, an ambient universe of sets, unrestricted comprehension, quotient equality, or an external satisfaction relation.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeFirstOrderMembershipSubstitutionBoundary`

Proof. Start with SetLike TypeLike signature scope. It exposes $M, Q, I, R, E, H, C, P, N$ before any syntactic consumer may read the packet as a signature.

Apply the typed-comprehension kernel envelope. The comprehension side of the substitution remains bounded by M, Q, I, R, and the extensional row E stays a boundary ledger over the same classifier route.

Use the FirstOrder signature root unblock theorem. A substitution consumer that leaves $M, Q, I, R, E, H, C, P, N$ lacks the displayed SetLike base, so host membership, ambient universes, unrestricted comprehension, quotient equality, and external satisfaction remain outside the accepted boundary. □

Theorem 3668.89 (SetLike ModelTheory satisfaction envelope)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, a* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction consumer may read SetLike data only after the* $\mathsf{FirstOrder}^{\uparrow}$ *syntax has consumed the classifier-bounded membership and typed-comprehension route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Thus satisfaction sees a BEDC envelope over displayed membership, restricted comprehension, extensional boundary, transport, replay, provenance, and local naming rows, not an ambient powerset, host universe, quotient set equality, unrestricted separation, or external satisfaction predicate.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeModelTheorySatisfactionEnvelope`

Proof. Read the ModelTheory satisfaction root unblock theorem. It already routes satisfaction-facing use through the classifier-based membership route and the FirstOrder syntactic carrier.

Next use FirstOrder membership substitution boundary. The syntactic carrier receives only $M, Q, I, R, E, H, C, P, N$, so the satisfaction consumer inherits the same classifier-bounded envelope.

Finally apply typed-comprehension kernel envelope and extensional ledger kernel nonescape. The comprehension and extensional rows remain finite BEDC ledgers, and no ambient powerset, host universe, quotient set equality, unrestricted separation, or external satisfaction predicate is available. □

Theorem 3668.90 (SetLike root-unblock kernel membership envelope)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *or* $\mathsf{TypeLike}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *reads membership only through the finite kernel envelope*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

Here M *is the displayed* BHist *membership carrier,* Q *is the classifier-bounded membership row,* I *is the subset-implication ledger,* R *is the bounded-comprehension row,* E *is the extensional-boundary row, and* H, C, P, N *provide componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming. The envelope does not expose host set extensionality, quotient membership, unbounded comprehension, or an ambient universe of sets.*

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootUnblockKernelMembershipEnvelope`

Proof. Start with the SetLike root-unblock theorem and the TypeLike signature scope. These routes expose the membership carrier M, classifier row Q, subset implication ledger I, bounded-comprehension row R, extensional-boundary row E, and the structural tail H, C, P, N before any typed consumer can read the packet.

For FirstOrder consumers, apply the FirstOrder membership substitution boundary. For ModelTheory consumers, apply the ModelTheory satisfaction envelope. Both routes pass through the same classifier-bounded membership and typed-comprehension rows before syntax or satisfaction can use SetLike data.

Finally use typed-comprehension kernel envelope and extensional ledger kernel nonescape. The bounded-comprehension and extensional-boundary rows remain displayed BEDC rows, while H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host set extensionality, quotient membership, unbounded comprehension, or an ambient universe of sets, every listed consumer receives exactly the finite kernel membership envelope. □

Theorem 3668.91 (SetLike root membership triple unblock)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, *and* $\mathsf{TypeLike}^{\uparrow}$ *consumer that reads membership from an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must enter through the shared BEDC membership source* M *and classifier row* Q *before it can consume syntax, satisfaction, or carrier-transport data. The common read is the finite route*

$$M, Q \longmapsto I, R, E, H, C, P, N.$$

It cannot be replaced by host elementhood, an ambient powerset, unrestricted separation, quotient equality, an external satisfaction predicate, or a selected element.

Proof. Start with the membership carrier root surface and classifier stability. They make M, Q the only membership source and classifier available to the three downstream consumers.

For syntactic use, apply the FirstOrder membership substitution boundary. For satisfaction use, apply the ModelTheory satisfaction envelope. For carrier transport, apply the TypeLike signature scope.

Finally use the root-unblock kernel membership envelope. All three reads are repacked through the same $M, Q, I, R, E, H, C, P, N$ ledger, so none of them has a coordinate for host elementhood, powersets, unrestricted separation, quotient equality, external satisfaction, or selected elements. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipTripleUnblock`

Theorem 3668.92 (SetLike root membership source triple)**.** *For an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, the row* M *is the single source-admission row for* $\mathsf{FirstOrder}^{\uparrow}$ *formulas,* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction rows, and* $\mathsf{TypeLike}^{\uparrow}$ *carrier transport. Each consumer may read* M *only together with the classifier* Q *and the structural rows* H, C, P, N*, and no consumer may import a second membership source, host universe, quotient-set source, unrestricted separation domain, or selected representative.*

Proof. Project the membership carrier $S_{\in} = (M, Q, H, C, P, N)$ from Definition 3668.83. It exposes one source row M and one classifier row Q.

Use classifier stability and the TypeLike signature scope. They force typed transport and syntactic reads to keep the same source while H, C, P, N replay and name the packet.

Apply the ModelTheory satisfaction envelope for satisfaction-facing reads. Since satisfaction is downstream of the FirstOrder syntax carrier, it inherits the same M, Q, H, C, P, N source admission and cannot add a host universe, quotient source, unrestricted domain, or selected representative. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootMembershipSourceTriple`

Theorem 3668.93 (SetLike root comprehension ledger triple)**.** *For every accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$*, bounded comprehension remains a ledger-filtered membership route for* $\mathsf{FirstOrder}^{\uparrow}$*,* $\mathsf{ModelTheory}^{\uparrow}$*, and* $\mathsf{TypeLike}^{\uparrow}$ *consumers:*

$$M, Q, I, R \longmapsto E, H, C, P, N.$$

The row R *may be read only after* M, Q, I *are visible, and its result may be compared only through* E *before transport, replay, provenance, and local naming. The route exports no ambient powerset, unrestricted separation, quotient equality, host subset object, external comprehension predicate, or choice of elements.*

Proof. Begin with the bounded-comprehension ledger and the typed-comprehension kernel envelope. They place R after the membership source, membership classifier, and subset-implication rows.

Next apply the FirstOrder membership substitution boundary and ModelTheory satisfaction envelope. Syntax and satisfaction can consume comprehension only through the same classifier-bounded ledger.

Finally use TypeLike signature scope and the root-unblock kernel membership envelope. Typed carrier transport receives the same $M, Q, I, R, E, H, C, P, N$ route, while powersets, unrestricted separation, quotient equality, host subset objects, external comprehension predicates, and element choices remain outside the accepted packet. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeRootComprehensionLedgerTriple`

Theorem 3668.94 (SetLike type-model membership envelope)**.** *Every* $\mathsf{TypeLike}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, and* $\mathsf{ModelTheory}^{\uparrow}$ *consumer of an accepted* $\mathsf{SetLike}^{\uparrow}$ *carrier* $S = (M, Q, I, R, E, H, C, P, N)$ *must read membership, bounded comprehension, extensional boundary, transport, replay, provenance, and local naming as the single finite route*

$$M, Q, I, R, E \longmapsto H, C, P, N.$$

No such consumer may replace this route by host membership, an ambient powerset, unrestricted separation, quotient equality, or external satisfaction.

Proof. Begin with the root membership triple and the root comprehension ledger triple, Definition 3668.91 and Definition 3668.93. They expose M, Q, I, R, E before any typed, syntactic, or satisfaction-facing consumer can read the packet.

For syntactic consumers, apply the FirstOrder membership substitution boundary, Definition 3668.88. For satisfaction consumers, apply the ModelTheory satisfaction envelope, Definition 3668.89. For typed-family consumers, use TypeLike signature scope, Definition 3668.86.

The three routes share the same structural tail H, C, P, N. Since the accepted carrier stores no coordinate for host membership, powersets, unrestricted separation, quotient equality, or external satisfaction, the displayed route is the only membership envelope available to all three consumers. □

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikeTypeModelMembershipEnvelope`

Theorem 3668.95 (SetLike public NameCert export)**.** *The public* $\mathsf{SetLike}^{\uparrow}$ *export is exactly the finite route*

$$M, Q, I, R, E, H, C, P, N.$$

Membership reads expose M, Q, subset reads expose I, restricted-comprehension reads expose R, extensional-boundary reads expose E, and all public consumers pass through the structural rows H, C, P, N. The export contains no powerset coordinate, unrestricted separation principle, host quotient equality of sets, selected representative row, global choice coordinate, or external universe of all sets.

Lean4 ✓ `BEDC.Derived.SetlikeUp.SetlikePublicNameCertExport`

Proof. Start from the carrier of Definition 3668.1. Its displayed rows are precisely $M, Q, I, R, E, H, C, P, N$, so every public read must be assembled from those rows before any downstream consumer can name a $\mathsf{SetLike}^{\uparrow}$ packet.

The membership route theorem Definition 3668.27 supplies the public read for M, Q, H, C, P, N. The subset ledger obligation Definition 3668.70 and the bounded comprehension obligation Definition 3668.59 add I and R without adding a powerset or unrestricted separation coordinate.

Finally, the extensional-boundary consumer exactness theorem Definition 3668.82 fixes E as a displayed boundary row rather than host quotient equality. The TypeLike, FirstOrder, and ModelTheory consumer boundary keeps sibling consumers on the same nine-row route, so the public export is exactly $M, Q, I, R, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SetLike}^{\uparrow}$ packet is generated from a finite membership source, classifier row, subset implication row, restricted-comprehension row, extensionality-boundary ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3668.1, Definition 3668.10, the membership, subset, bounded-comprehension, extensional-boundary, TypeLike, FirstOrder, and ModelTheory consumer-boundary rows, and Definition 3668.95 bind the mature paper surface from the finite SetLike membership carrier to its public consumer export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SetLike}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Not claimed: This chapter does not claim an ambient powerset, unrestricted separation, quotient equality of sets, selected representatives, global choice of elements, Lean verification, bridge checking, or a host universe of all sets.

Upgrade path: Bridge checking requires Lean verification of the same carrier, consumer-boundary rows, and public export for BEDC.Derived.SetLikeUp.

Chapter 3669

A Concrete Naming Certificate for $\mathsf{SchauderBasis}^{\uparrow}$

Following Schauder and the Banach-space basis tradition, the $\mathsf{SchauderBasis}^{\uparrow}$ packet records a sequence of named basis vectors together with finite coordinate projections and partial-sum controls. It sits over the linear carrier of chapter 47, the norm row of chapter 89, the Banach-space completion surface of chapter 90, the finite-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives basis expansions a finite NameCert surface before any coordinate functional or projection budget is consumed.

Definition 3669.1 (Schauder basis carrier)**.** *A* $\mathsf{SchauderBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, Nrm, B, C, S, E, R, H, T, P, L)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *row naming the vector-space carrier, addition, scalar action, zero vector, and finite linear-combination syntax.*
2. *Nrm is the* $\mathsf{Norm}^{\uparrow}$ *row naming the norm, triangle budget, scalar-compatibility budget, and zero-detection surface.*
3. *B is the basis-vector stream row, a* $\mathsf{StreamName}^{\uparrow}$ *finite-window record whose inspected entries name the vectors $b_0, b_1, \ldots$ available to the packet.*
4. *C is the coordinate-readback row, recording finite coordinate requests and their* $\mathsf{RegSeqRat}^{\uparrow}$ *rational names.*
5. *S is the partial-sum row, recording finite sums $\sum_{i<n} c_i b_i$ together with the requested truncation index.*
6. *E is the error-control row, a* $\mathsf{Real}^{\uparrow}$*-facing norm estimate for the tail between the named vector and the displayed partial sum.*
7. *R is the reconstruction row, recording that each accepted vector read is recovered through the displayed partial sums and tail estimates.*
8. *H records componentwise* hsame *transport for the vector, norm, basis stream, coordinate, partial-sum, error, reconstruction, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay from a coordinate request through B, C, S, E, R.*

10. *P records* Pkg *provenance over* $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, Nrm, B, C, S, E, R, H, T, P, L$. It has no coordinate for a Hamel basis choice, an arbitrary linear functional, a detached coefficient oracle, ambient compactness, countable choice, or a global unconditional-basis assertion.

Lean4 ✓ `BEDC.Derived.SchauderBasisUp.SchauderBasisTasteGate single carrier alignment`

Theorem 3669.2 (Schauder basis NameCert obligations)**.** *For every accepted* $\mathsf{SchauderBasis}^{\uparrow}$ *carrier $K = (V, Nrm, B, C, S, E, R, H, T, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, vector-space exposure through V, norm exposure through Nrm, basis-stream exposure through B, coordinate readback through C, partial-sum exposure through S, tail-error control through E, reconstruction through R, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through P, and local naming through L.*

Proof. Carrier admission is coordinate projection from Definition 3669.1. The accepted packet exposes the vector-space carrier, norm row, basis-vector stream, coordinate readback, partial sums, tail-error estimates, reconstruction row, transport, replay, provenance, and local naming.

The analytic route is fixed by V, Nrm, B, C, S, E, R. The vector row supplies the finite linear-combination syntax, the norm row supplies the tail metric, the basis stream supplies inspected vectors, the coordinate row supplies rational names, the partial-sum row supplies the finite approximants, the error row supplies the Real-facing tail estimate, and the reconstruction row binds those finite approximants to the named vector.

The structural rows H, T, P, L preserve that same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for a Hamel basis choice, detached coefficient oracle, arbitrary functional, compactness substitute, countable-choice schedule, or unconditional-basis assertion, those objects cannot belong to the local NameCert surface. □

Theorem 3669.3 (Schauder basis coordinate-projection handoff)**.** *Every coordinate-projection consumer of an accepted* $\mathsf{SchauderBasis}^{\uparrow}$ *packet must read the projection through the basis stream B, coordinate row C, partial-sum row S, tail-error row E, reconstruction row R, replay row T, provenance row P, and local name L. The handoff exports finite coordinate projections and norm-controlled partial sums, but it exports no standalone bounded linear functional, no choice of representatives outside the displayed stream, and no unconditional convergence claim.*

Proof. Start with a coordinate-projection request. The carrier definition places the visible projection data in B, C, S, E, R: the basis vectors are inspected through the stream row, coordinates are read through regular rational names, partial sums are formed in the vector-space row, and the tail is measured by the norm and Real-facing estimate.

Next use the replay and provenance rows. The replay row T requires the consumer to traverse the displayed coordinate and partial-sum route, while P records that the route belongs to the $\mathsf{SchauderBasis}^{\uparrow}$ packet and its Banach-space dependencies.

Finally the local name L certifies only that displayed route. Because the packet has no coordinate for arbitrary functionals, detached coefficient selection, ambient compactness, or unconditional convergence, a coordinate-projection consumer receives only the finite projection handoff governed by B, C, S, E, R, T, P, L. □

Falsifiable prediction: Every accepted $\mathsf{SchauderBasis}^{\uparrow}$ coordinate-projection export factors through a VecSpace carrier row, Norm row, StreamName basis-vector window, RegSeqRat coordinate readback, finite partial-sum row, Real tail estimate, reconstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: banach, vecspace, norm, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (V, Nrm, B, C, S, E, R, H, T, P, L)$, because the Schauder basis carrier binds linear syntax, norm control, basis windows, rational coordinate readback, finite partial sums, tail estimates, reconstruction, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{SchauderBasis}^{\uparrow}$ packet is generated from a vector-space row, norm row, basis-vector StreamName window, RegSeqRat coordinate readback, finite partial-sum row, Real tail estimate, reconstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3669.1, Definition 3669.2, and Definition 3669.3 seed the Schauder basis route from Banach-space carrier data to finite coordinate projections and norm-controlled partial sums.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchauderBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Hamel bases, arbitrary linear functionals, detached coefficient oracles, ambient compactness, countable choice, unconditional convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, basis-window exposure, coordinate-readback exactness, partial-sum control, tail-error control, reconstruction, projection handoff, and non-escape rows for the displayed Schauder basis packet.

Chapter 3670

A Concrete Naming Certificate for $\mathsf{RationalCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalCauchyCompletion}^{\uparrow}$ packet records the rational Cauchy completion route as finite regular-name data rather than as a quotient of rational sequences or a selected ambient limit. It sits between the rational source of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 completion route a concrete carrier before later formalization reads it as a derived Lean surface.

Definition 3670.1 (Rational Cauchy completion carrier)**.** *A* $\mathsf{RationalCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, S, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row, recording the rational endpoints and rational difference witnesses supplied to the completion route.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording finite dyadic radii, slack comparisons, and tolerance budgets.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording inspected requests and omitted-tail boundaries for the source rational stream.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational name produced by the displayed windows and tolerances.*
5. *M is the explicit Cauchy-modulus row, a finite schedule assigning every requested tolerance a displayed tail threshold.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, D, S, R, M have been displayed.*
7. *H records componentwise* hsame *transport for the rational source, dyadic tolerance, stream window, regular readback, modulus, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through Q, D, S, R, M, E.*
9. *P records* Pkg *provenance over* $\mathsf{RationalCauchyCompletion}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, S, R, M, E, H, C, P, N$. It has no coordinate for quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalCauchyCompletionUp.RationalCauchyCompletionTasteGate single carrier alignment`

Theorem 3670.2 (Rational Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{RationalCauchyComplet}$ *carrier $K = (Q, D, S, R, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, rational-source exposure through Q, dyadic tolerance exposure through D, finite-window exposure through S, regular readback through R, explicit Cauchy-modulus control through M, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is coordinate projection from Definition 3670.1. The accepted packet exposes the rational source, dyadic tolerance ledger, finite stream window, regular rational readback, explicit modulus, terminal Real seal, transport, replay, provenance, and local naming rows.

The analytic route is fixed by Q, D, S, R, M, E. The source row Q supplies rational endpoints, D supplies dyadic tolerance comparisons, S exposes only finite observations, R records the regular rational readback, M binds each requested tolerance to a displayed tail threshold, and E seals the resulting Real-facing read.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 3670.3 (Rational Cauchy completion diagonal seal handoff)**.** *Every diagonal completion read from a* $\mathsf{RationalCauchyCompletion}^{\uparrow}$ *packet to a* $\mathsf{Real}^{\uparrow}$ *consumer factors through the same finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The explicit modulus row M is the only row that may choose tail thresholds for the diagonal, and the terminal seal E exports only the regular rational readback named by R.

Proof. Start with a Real-facing diagonal read. The carrier definition places the terminal seal in coordinate E, and E is downstream of the source, tolerance, stream, readback, and modulus rows.

Trace the read backward. A diagonal tolerance request is served by M, interpreted through the dyadic row D, inspected through the finite-window row S, and returned as the regular rational readback R from the rational source Q.

Finally H, C, P, N transport, replay, package, and name the same displayed route. There is no separate coordinate for a selected limit, quotient stream equality, countable choice schedule, or host real equality, so every accepted diagonal handoff reaches $\mathsf{Real}^{\uparrow}$ only through Q, D, S, R, M, E. □

Falsifiable prediction: Every accepted $\mathsf{RationalCauchyCompletion}^{\uparrow}$ Real-facing read factors through a Rat source row, DyadicRatCore tolerance row, StreamName finite-window row, RegSeqRat readback row, explicit Cauchy-modulus row, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: no listed sibling supplies the whole packet $K = (Q, D, S, R, M, E, H, C, P, N)$, because the completion carrier binds

source rationals, dyadic tolerances, finite windows, regular readback, explicit modulus, terminal sealing, transport, replay, provenance, and local naming in one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}$.

Constructive story: A $\mathsf{RationalCauchyCompletion}^{\uparrow}$ packet is generated from a Rat source row, DyadicRatCore tolerance row, StreamName finite-window row, RegSeqRat readback row, explicit Cauchy modulus, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3670.1, Definition 3670.2, and Definition 3670.3 seed the rational Cauchy completion route from rational source data to the Real seal through finite tolerance, window, readback, modulus, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalCauchyCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient sequence equality, selected representatives, countable choice, detached limits, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic tolerance exposure, finite-window exposure, regular readback, explicit modulus control, Real-seal handoff, and non-escape rows for the displayed rational Cauchy completion packet.

Chapter 3671

A Concrete Naming Certificate for $\mathsf{CompactMetricClosedBall}^{\uparrow}$

Following Bishop and Bridges constructive compact metric practice, the $\mathsf{CompactMetricClosedBall}^{\uparrow}$ packet consumes chapter 3464, chapter 129, chapter 57, chapter 127, chapter 128, and chapter 38. It names the finite-net surface of a closed ball inside a displayed compact metric carrier; it does not turn the ball into a host subset, choose a finite subcover, quotient points, or import ambient real completeness.

Definition 3671.1 (Compact metric closed ball carrier)**.** *A* $\mathsf{CompactMetricClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, c, r, M, F, T, C, H, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the shared metric carrier, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net family, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *complete-limit family over one point classifier.*
2. *c is the displayed center row carried by the same metric source.*
3. *r is the* $\mathsf{Real}^{\uparrow}$ *radius row together with its nonnegative-radius ledger.*
4. *M is the* $\mathsf{ClosedMetricBall}^{\uparrow}$ *membership row, recording the comparison $d(c, x) \leq r$ for each displayed point row x that enters the ball surface.*
5. *F is the finite-net row obtained by restricting a* $\mathsf{TotallyBounded}^{\uparrow}$ *ProbeBundle-net request to centers whose coverage route remains inside the closed-ball membership row M.*
6. *T is the total-bounded handoff row that records the tolerance request, bundled centers, membership constructors, coverage distances, and rational comparison ledgers inherited from the compact metric source.*
7. *C is the complete-limit inheritance row: every carried Cauchy name whose observations remain inside M is routed through the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit witness of X and returns a limit row still admitted by M.*
8. *H records componentwise* hsame *transport for the compact metric source, center, radius, closed-ball membership, finite-net row, total-bounded handoff, complete-limit inheritance, and structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricClosedBall}^{\uparrow}$, $\mathsf{ClosedMetricBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected finite subcover, quotient balls, host subsets, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactMetricClosedBallUp.CompactMetricClosedBallTasteGate single carrier alignment`

Theorem 3671.2 (Compact metric closed ball finite-net obligations)**.** *For an accepted* $\mathsf{CompactMetricClosedBall}^{\uparrow}$ *carrier* $K = (X, c, r, M, F, T, C, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the compact metric source X, center c, radius r, closed-ball membership row M, finite-net row F, total-bounded handoff T, complete-limit inheritance row C, transport row H, provenance row P, and local naming row N;*
2. *closed-ball membership stability, requiring the comparison $d(c, x) \leq r$ to transport through the metric classifier, the* $\mathsf{Real}^{\uparrow}$ *radius ledger, and the* $\mathsf{ClosedMetricBall}^{\uparrow}$ *row M;*
3. *finite-net extraction inside the closed ball, requiring each displayed tolerance request to read bundled centers, membership constructors, coverage distances, and rational comparison rows through F and T without leaving M;*
4. *completion inheritance for carried Cauchy names, requiring every Cauchy thread whose observations are admitted by M to use the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness family of X and return a limit row still admitted by the same closed-ball comparison;*
5. *ledger nonescape, stating that no open-cover compactness theorem, selected subcover, quotient ball, host subset, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row is exported by the packet.*

Proof. Project the carrier of Definition 3671.1. The tuple lists exactly the compact metric source, center, radius, closed-ball membership row, finite-net row, total-bounded handoff, complete-limit inheritance, transport, provenance, and local naming row.

Membership stability is inherited from the displayed metric and closed-ball surfaces. The row M supplies the center-radius comparison, X supplies the metric classifier and distance rows, and the radius row r supplies the $\mathsf{Real}^{\uparrow}$ comparison ledger. Transport through H preserves the same comparison rather than replacing it by a host subset predicate.

For a positive tolerance request, read the $\mathsf{TotallyBounded}^{\uparrow}$ family inside X and then restrict the resulting ProbeBundle route through F. The handoff row T records the bundled centers, membership constructors, coverage distances, and rational comparison ledgers that remain inside the closed-ball membership row M.

For a carried Cauchy name, use only the complete-limit family already present in X. The inheritance row C states that observations admitted by M are handed to the complete-metric limit witness and that the returned limit row is checked again by the same center-radius comparison.

The remaining structural rows H, P, N transport, source, and name this finite route. Since the carrier inventory contains no open-cover object, subcover selector, quotient-ball constructor, host subset, countable-choice row, or ambient real-completeness row, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactMetricClosedBall}^{\uparrow}$ packet is generated from a $\mathsf{CompactMetric}^{\uparrow}$ source row, center row, $\mathsf{Real}^{\uparrow}$ radius row, $\mathsf{ClosedMetricBall}^{\uparrow}$ membership row, restricted finite-net row, total-bounded handoff, complete-limit inheritance, componentwise hsame transport, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$, seed)

Scope closed: Definition 3671.1 and Definition 3671.2 seed the finite closed-ball compact-metric NameCert surface for displayed center-radius packets.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactMetricClosedBall}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice of subcover, quotient balls, host subsets, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, membership stability, finite-net extraction inside the closed ball, completion inheritance for carried Cauchy names, and ledger nonescape.

Chapter 3672

A Concrete Naming Certificate for RegularCauchyProductCriterion$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the RegularCauchyProductCriterion$^{\uparrow}$ packet isolates the finite sufficiency criterion by which two regular rational Cauchy source rows produce a product read suitable for the CauchyProductCriterion$^{\uparrow}$ surface. It sits between the pointwise regular product carrier of chapter 461 and the synchronized product criterion of chapter 1586. The chapter records the L10 Real and Completion handoff as finite NameCert data, not as host multiplication of completed reals, quotient equality of streams, a selected product limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 3672.1 (Regular Cauchy product criterion carrier)**.** *A* RegularCauchyProductCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A and B are the two* RegSeqRat$^{\uparrow}$ *source rows whose scheduled rational observations supply the regular Cauchy inputs.*
2. *W_A and W_B are synchronized* StreamName$^{\uparrow}$ *window rows for the same precision request and product observation index.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *product-budget row, recording input tolerances, visible endpoint magnitude bounds, cross-error terms, and the output tolerance assigned to the product read.*
4. *P is the* RegularCauchyProduct$^{\uparrow}$ *route row, read only after A, B, W_A, W_B, D have been displayed.*
5. *K is the* CauchyProductCriterion$^{\uparrow}$ *sufficiency row, certifying that the displayed product route is adequate for the downstream criterion.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced by the route P under the budget D and criterion row K.*
7. *E is the* Real$^{\uparrow}$ *seal row, consumed only after the regular-rational readback R has been displayed.*
8. *H records componentwise* hsame *transport for $A, B, W_A, W_B, D, P, K, R, E, C, L, N$ and the local packet shape.*
9. *C records displayed* Cont *replay from the two source rows through the synchronized windows, dyadic product budget, regular product route, criterion row, readback row, and real seal.*

10. *L is the* Pkg *provenance over* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W_A, W_B, D, P, K, R, E, H, C, L, N$. It has no coordinate for host multiplication of completed reals, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductCriterionUp.RegularCauchyProductCriterionTasteGate single carrier alignment`

Theorem 3672.2 (Regular Cauchy product criterion NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier $Q = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *obligations are exactly carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic product-budget exactness, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, component-wise transport through H, displayed continuation replay through C, provenance through L, and local naming through N.*

Proof. Carrier admission is projection from Definition 3672.1. The displayed packet lists the two source rows, synchronized windows, dyadic budget, product route, criterion row, readback row, real seal, and structural rows.

The source and window obligations are read from A, B, W_A, W_B. These rows display the regular rational inputs and the shared product window before any product observation reaches the criterion.

The dyadic product-budget obligation is the row D. It records finite tolerance and magnitude data, so the route P is a budgeted regular Cauchy product route rather than host multiplication of completed real names.

The rows K, R, E are the downstream sufficiency, regular-rational, and real-seal boundaries. Transport through H, replay through C, provenance through L, and local naming through N preserve the same route and introduce no additional coordinate.

Thus the listed rows exhaust the local certificate surface. □

Theorem 3672.3 (Regular Cauchy product criterion window-budget exactness)**.** *For an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier, the synchronized source windows W_A, W_B and the dyadic row D determine the product-budget data consumed by the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route P and the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *row K. Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of the product factors through*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto P \longmapsto K \longmapsto R.$$

No additional budget rule is read from a quotient stream, a selected product limit, countable choice, trichotomy, host real multiplication, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyProductCriterionUp.RegularCauchyProductCriterionCarrier window budget exactness`

Proof. Read the two regular rational source rows through the synchronized windows. The rows W_A and W_B expose the finite observations used by the product request at the same displayed precision.

The row D is the only dyadic product-budget row in the packet. It supplies the tolerance and endpoint-bound data consumed by the regular product route P.

The sufficiency row K then certifies that the displayed route is acceptable for the Cauchy product criterion. Since the readback row R occurs only after P and K, every regular-rational product read factors through the displayed chain.

The carrier has no coordinate for any excluded budget source, so the budget exactness is internal to the finite packet. □

Theorem 3672.4 (Regular Cauchy product criterion Real seal non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *carrier factors through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *sufficiency row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the displayed* hsame *transports,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$ *row. Hence the terminal seal is a finite product-sufficiency object, not host multiplication, quotient stream equality, a selected limit, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The real-facing consumer can read only E as the terminal $\mathsf{Real}^{\uparrow}$ seal row. In the carrier, E is downstream of R, and R is downstream of the criterion route through P and K.

Apply the window-budget exactness theorem. It places the readback route on the finite chain from the two source rows through the synchronized windows, dyadic product budget, product route, criterion row, and regular-rational readback.

The structural rows H, C, L, N preserve that same chain under transport, replay, provenance, and local naming. They do not add a second source for the real seal.

Therefore every terminal read remains inside the displayed finite criterion, and the excluded host, quotient, selector, choice, order, and completeness routes cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore product budget, a RegularCauchyProduct route, a CauchyProductCriterion sufficiency row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3672.1, Definition 3672.2, Definition 3672.3, and Definition 3672.4 seed the finite L10 product-sufficiency object between RegularCauchyProduct and CauchyProductCriterion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyProductCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient streams, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic product-budget exactness, RegularCauchyProduct routing, CauchyProductCriterion sufficiency, RegSeqRat readback, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.RegularCauchyProductCriterionUp.

Chapter 3673

A Concrete Naming Certificate for $\mathsf{BishopMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopMetricSpace}^{\uparrow}$ packet names a metric presentation whose distances are read through displayed regular rational approximations, finite stream windows, and real-name seals. It sits between the abstract metric certificate of chapter 57, the positive-separation interface of chapter 3505, the finite observation surface of chapter 126, the regular rational route of chapter 301, and the sealed real-name consumer of chapter 38. The packet is a constructive metric object for Real and completion-facing consumers; it is not a host metric-space structure, quotient completion, choice-selected limit, or ambient classical distance oracle.

Definition 3673.1 (Bishop metric-space carrier)**.** *A* $\mathsf{BishopMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, D, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the displayed point-name row and local carrier admission surface.*
2. *D is the metric-distance row, read as a* $\mathsf{Real}^{\uparrow}$ *name rather than as a host real-valued function.*
3. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to observe the distance name.*
4. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of each displayed distance.*
5. *A is the* $\mathsf{MetricApartness}^{\uparrow}$ *handoff row, present when a positive distance window exports apartness.*
6. *H stores componentwise* hsame *transport for point names, distance rows, stream windows, regular rational readbacks, apartness handoff, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from point-name reads through the distance, stream, regular rational, apartness, and Real-seal rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricApartness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, D, S, R, A, H, C, P, N$. It has no coordinate for host equality of points, host-valued distance functions, quotient completion, choice-selected limits, total boundedness, compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BishopMetricSpaceUp.BishopMetricSpaceTasteGate single carrier alignment`

Theorem 3673.2 (Bishop metric-space NameCert obligations)**.** *For every accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *carrier*

$$M = (X, D, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed point names, Real-valued distance rows, finite stream windows, regular rational readbacks, optional positive-distance apartness handoff, componentwise transport, displayed replay, package provenance, and local naming row.*

Proof. Project the carrier of Definition 3673.1. The projection exposes point-name rows X, distance rows D, stream windows S, regular readbacks R, and the apartness handoff A.

The distance row is Real-facing because every inspected distance is routed through S and R before it reaches the $\mathsf{Real}^{\uparrow}$ seal. Thus no consumer can replace the displayed constructive distance with a host-valued function.

The apartness handoff is confined to A. It can be exported only when the displayed distance window supplies positive separation, and the transport rows H, C replay that same finite route.

Finally P and N record provenance and local naming for exactly these rows. Since the carrier has no coordinate for host point equality, quotient completion, selected limits, compactness, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3673.3 (Bishop metric-space located Cauchy route)**.** *Every completion-facing read of an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet factors through*

$$X, D \longmapsto S, R \longmapsto A, H, C, P, N.$$

Thus a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer may inspect a Cauchy or locatedness claim only through displayed distance windows, regular rational readbacks, optional apartness handoff, transport, replay, provenance, and local naming.*

Proof. Begin with the point-name and distance rows X, D. They name the metric input but do not by themselves expose a Real-valued oracle.

The stream and regular rational rows S, R provide the finite observation route for the distance name. Any located Cauchy read must therefore pass through those rows before it reaches a Real-facing consumer.

The rows A, H, C, P, N preserve the same route under apartness handoff, componentwise transport, replay, provenance, and local naming. A claim using quotient completion, selected limits, host equality of points, compactness, total boundedness, or ambient completeness has no carrier coordinate and leaves the packet. □

Theorem 3673.4 (Bishop metric-space distance-window exactness)**.** *Let* $M = (X, D, S, R, A, H, C, P, N)$ *be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Every accepted read of a distance row factors through the displayed route*

$$D \longmapsto S \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the distance consumed by a $\mathsf{Real}^{\uparrow}$*-facing or completion-facing reader is the finite StreamName window and RegSeqRat readback of* D*, not a host-valued metric function or an ambient distance oracle.*

Proof. Project the carrier of Definition 3673.1. The row D is present only as the displayed metric-distance row, while S and R are the StreamName and RegSeqRat rows attached to that distance.

Apply Definition 3673.2. Its local certificate surface lists distance rows, finite stream windows, regular rational readbacks, transport, replay, provenance, and local naming as the exact obligations for accepted packets.

Finally use Definition 3673.3. Any Real-facing or completion-facing read must pass through the same S, R windows before it reaches a Real consumer, so no host-valued metric function can replace the displayed route. □

Theorem 3673.5 (Bishop metric-space apartness handoff)**.** *Let $M = (X, D, S, R, A, H, C, P, N)$ be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Apartness can be exported from M only through the displayed positive-distance row A, and only after the same distance-window route*

$$D \longmapsto S \longmapsto R$$

has supplied the positive separation read. The handoff cannot be replaced by host inequality of points or by an apartness predicate detached from the displayed distance data.

Proof. Read the carrier in Definition 3673.1. The row A is the only MetricApartness handoff coordinate, and the carrier contains no separate host point-equality or host inequality field.

Apply Definition 3673.2. The certificate surface makes the apartness row optional and ties it to positive-distance evidence transported with the same componentwise rows H and C.

Use Definition 3673.3 to identify the upstream read. Since every completion-facing distance claim goes through D, S, R before the structural rows, the exported apartness handoff is exactly the displayed row A downstream of those windows. □

Theorem 3673.6 (Bishop metric-space Real-seal nonescape)**.** *Let $M = (X, D, S, R, A, H, C, P, N)$ be an accepted* $\mathsf{BishopMetricSpace}^{\uparrow}$ *packet. Every Real-facing consumer of a distance row receives only the finite route*

$$D \longmapsto S \longmapsto R \longmapsto \mathsf{Real}^{\uparrow}$$

together with the structural rows A, H, C, P, N when they are displayed. The packet exports no quotient completion, selected limit, host point equality, total boundedness, compactness, or ambient completeness principle.

Proof. Start from Definition 3673.1. The accepted packet exposes point names, a distance row, finite windows, regular readbacks, optional apartness, transport, replay, provenance, and local naming, and it has no coordinate for the excluded objects.

Apply Definition 3673.2. The local NameCert admits exactly those displayed rows and refuses host-valued distance functions, quotient completion data, selected limits, compactness, total boundedness, and ambient completeness.

Apply Definition 3673.3. The Real-facing read stays on the finite StreamName and RegSeqRat path before it reaches the seal, so the consumer cannot escape into any object outside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMetricSpace}^{\uparrow}$ packet is generated from displayed point names, Real-valued distance rows, StreamName finite windows, RegSeqRat regular rational readbacks, optional MetricApartness handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3673.1, Definition 3673.2, and Definition 3673.3 seed the constructive metric carrier, finite NameCert obligations, and located Cauchy route for Real and completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{BishopMetricSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected limits, compactness, total boundedness, host equality of points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-window exactness, regular rational readback, apartness handoff, located Cauchy routing, and TasteGate field faithfulness for BEDC.Derived.BishopMetricSpaceUp.

Chapter 3674

A Concrete Naming Certificate for $\mathsf{BishopSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, a Bishop space is read through finite observation windows, explicit metric evidence, and constructive continuity data rather than through a host set with ambient point equality. The $\mathsf{BishopSpace}^{\uparrow}$ packet records that interface as a NameCert surface on the L10 Phase Real and Completion route: it starts from the metric witness of chapter 57, hands distance reads to the located metric-space route of chapter 3685, and keeps the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows visible before any completion-facing consumer reads the packet.

Definition 3674.1 (Bishop-space carrier)**.** *A* $\mathsf{BishopSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, M, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the local point-observation row. It contains only finite observation windows and the classifier used by the packet; it is not a host equality relation on an ambient set.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, recording the metric witness consumed by each accepted observation pair.*
3. *L is the* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *handoff row. It routes each metric read through located-distance evidence before the read can be used by a completion-facing consumer.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the located metric read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the displayed stream-name windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular readback.*
7. *H records componentwise* hsame *transport for X, M, L, W, R, E and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from point observations to metric witnesses, then through located reads, stream windows, regular readbacks, and real seals.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopSpace}^\uparrow}$ *row.*
The carrier has no coordinate for host point equality, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopSpaceUp.BishopSpaceTasteGate single carrier alignment`

Theorem 3674.2 (Bishop-space NameCert obligations)**.** *For every accepted* $\mathsf{BishopSpace}^\uparrow$ *carrier*

$$B = (X, M, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSpace}^\uparrow}$ *surface consists exactly of carrier admission for the displayed observation row, metric-source readback through M, located metric-space handoff through L, finite stream-window exposure through W, regular rational readback through R, real-seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3674.1. The point-observation row X supplies the finite observation windows and the local classifier, while the metric source row M supplies the distance witness inspected by each accepted observation pair.

The located handoff is the role of L. It forces each distance read onto the $\mathsf{LocatedMetricSpace}^\uparrow$ route before the read can open completion-facing data.

The L10 readback obligations are then W, R, E. Row W displays the finite $\mathsf{StreamName}^\uparrow$ windows, row R packages those windows as $\mathsf{RegSeqRat}^\uparrow$ data, and row E exposes the $\mathsf{Real}^\uparrow$ seal only after the regular readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name exactly the same rows. Since the carrier contains no host point equality, quotient completion, selected limit, choice, or ambient real-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3674.3 (Bishop-space located-metric handoff)**.** *Every L10 completion consumer that reads an accepted* $\mathsf{BishopSpace}^\uparrow$ *packet factors through*

$$X \longmapsto M \longmapsto L \longmapsto W \longmapsto R \longmapsto E$$

together with H, C, P, N. Thus the consumer reads the $\mathsf{Met}^\uparrow$ *source, then the* $\mathsf{LocatedMetricSpace}^\uparrow$ *handoff, then the* $\mathsf{StreamName}^\uparrow$ *finite windows,* $\mathsf{RegSeqRat}^\uparrow$ *readback, and* $\mathsf{Real}^\uparrow$ *seal before any completion-facing endpoint is exposed.*

Proof. Begin with a completion-facing request against the accepted packet. The observation row X contains only finite point windows, so the request must use the metric source row M to obtain a displayed distance witness.

Next use the located handoff row L. This is where the metric read becomes a located metric-space read, so the route cannot jump from M directly to a real or completion endpoint.

The readback then passes through W, R, E. The finite windows in W are read as stream-name data, the regular rational row R packages those windows, and the real seal E is visible only after that regular readback.

Rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. Because the packet has no coordinate for host equality, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^\uparrow$ completeness, every completion-facing read remains on the displayed L10 route. □

Falsifiable prediction: Every L10 completion-facing read from a $\mathsf{BishopSpace}^{\uparrow}$ packet factors through the displayed MetricUp source, LocatedMetricSpaceUp handoff, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local naming row before completion data is consumed.

Independence witness: metric, locatedmetricspace, streamname, regseqrat, real: $\mathsf{BishopSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries point-observation windows, a MetricUp source, a LocatedMetricSpaceUp handoff, StreamName windows, RegSeqRat readback, RealUp seal, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSpace}^{\uparrow}$ packet is generated from finite point observations, a MetricUp source, a LocatedMetricSpaceUp handoff, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3674.1, Definition 3674.2, and Definition 3674.3 seed the Bishop-space carrier and its L10 real-and-completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host point equality, quotient completion, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric source readback, located metric-space handoff, stream-window exposure, RegSeqRat readback, RealUp seal handoff, transport replay, and ledger refusal for BEDC.Derived.BishopSpaceUp.

Chapter 3675

A Concrete Naming Certificate for $\mathsf{RealClosure}^{\uparrow}$

Following Kelley and Bourbaki topology, the $\mathsf{RealClosure}^{\uparrow}$ packet names a closure operation by displayed finite sequential data rather than by ambient completeness of a host real line. It sits over the terminal real-name seal of chapter 38, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the topological carrier of chapter 88, and the sequential closure packet of chapter 1571. The chapter uses these rows to expose an L10 closure object for Real-facing and completion-facing consumers without asserting quotient reals, host equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3675.1 (Real closure carrier)**.** *A* $\mathsf{RealClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, T, S, Q, A, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *point-name row whose accepted entries are inspected only through terminal real-name seals.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *neighbourhood row for the displayed Real-facing carrier.*
3. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *finite windows used to observe candidate closure witnesses.*
4. *Q stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of the same windows.*
5. *A is the* $\mathsf{SequentialClosure}^{\uparrow}$ *approach row, displaying a sequence or net-free sequential route from the named source rows toward the closure point.*
6. *E is the exactness and refusal row stating that the closure verdict is read from X, T, S, Q, A and not from quotient representatives, host real equality, choice-selected subsequences, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*
7. *H stores componentwise* hsame *transport for Real names, topological neighbourhood rows, stream windows, regular rational readbacks, sequential approach rows, exactness rows, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay from finite Real-name observations through topology, stream, regular rational, sequential closure, and refusal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RealClosure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SequentialClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealClosure}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, T, S, Q, A, E, H, C, P, N$. It has no coordinate for completed subsets of a host real line, quotient real equality, selected limits, countable-choice schedules, trichotomy, open-cover compactness, or ambient Real completeness.

Theorem 3675.2 (Real closure NameCert obligations)**.** *For every accepted* $\mathsf{RealClosure}^\uparrow$ *carrier*

$$R = (X, T, S, Q, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealClosure}^\uparrow}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Real}^\uparrow$ *point names,* $\mathsf{Topology}^\uparrow$ *neighbourhood rows,* $\mathsf{StreamName}^\uparrow$ *windows,* $\mathsf{RegSeqRat}^\uparrow$ *readbacks,* $\mathsf{SequentialClosure}^\uparrow$ *approach row, exactness and refusal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.RealClosureUp.TasteGate.RealClosureTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3675.1. The projection exposes Real names X, neighbourhood rows T, stream windows S, regular rational readbacks Q, and the sequential approach row A.

The closure verdict is then forced through E. That row records exactness of the displayed sequential route and refuses any replacement by quotient real representatives, host equality, countable-choice selection, trichotomy, open-cover compactness, or an ambient completeness theorem.

Finally transport and replay are only the componentwise rows H and C, while P and N record provenance and local naming for the same finite surface. Since no other coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 3675.3 (Real closure stream-regular handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{RealClosure}^\uparrow$ *packet reaches the closure verdict only through the ordered handoff*

$$\mathsf{Real}^\uparrow \longrightarrow \mathsf{StreamName}^\uparrow \longrightarrow \mathsf{RegSeqRat}^\uparrow \longrightarrow \mathsf{Topology}^\uparrow \longrightarrow \mathsf{SequentialClosure}^\uparrow$$

displayed by $X, S, Q, T, A, E, H, C, P, N$. The handoff supplies finite observation windows, regular rational readback, neighbourhood membership, sequential approach evidence, transport, replay, provenance, and local naming; it supplies no quotient real, host equality, countable-choice schedule, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^\uparrow$ *completeness principle.*

Proof. Begin with an accepted carrier $R = (X, T, S, Q, A, E, H, C, P, N)$. A consumer may inspect a Real name only through X, and every finite observation of that name is recorded by the stream-window and regular-rational rows S, Q.

The topological and sequential closure rows then consume the same readback. The neighbourhood row T records the local closure test, while A records the displayed sequential route that meets the test.

Use the exactness row E and the transport-replay rows H, C. They preserve the ordered handoff under classifier comparison and prevent a consumer from replacing the finite route by quotient equality, selected representatives, countable choice, trichotomy, compactness, or ambient completeness.

The provenance and local name rows P, N package exactly this route. Hence every downstream closure read receives the stated L10 handoff and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{RealClosure}^{\uparrow}$ packet is generated from Real point names, topology neighbourhood rows, finite StreamName windows, RegSeqRat readbacks, a SequentialClosure approach row, exactness refusal, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3675.1, Definition 3675.2, and Definition 3675.3 seed the constructive L10 closure carrier, finite NameCert obligation surface, and stream-to-regular-rational handoff for Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient reals, host real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-neighbourhood exactness, finite stream-window readback, regular rational readback, sequential closure handoff, refusal, transport stability, replay, provenance, and local naming for BEDC.Derived.RealClosureUp.

Chapter 3676

A Concrete Naming Certificate for $\mathsf{IntermediateValueTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{IntermediateValueTheorem}^{\uparrow}$ packet records the constructive intermediate-value theorem as a finite theorem-object certificate rather than as host real analysis. It reads the real-name seal of chapter 38 for $\mathsf{Real}^{\uparrow}$, the regular rational readback route of chapter 301 for $\mathsf{RegSeqRat}^{\uparrow}$, the finite observation windows of chapter 126 for $\mathsf{StreamName}^{\uparrow}$, the endpoint interval packet of chapter 394 for $\mathsf{LocatedInterval}^{\uparrow}$, the graph-and-modulus surface of chapter 130 for $\mathsf{ContinuousMap}^{\uparrow}$, and the closed interval carrier of chapter 1591 for $\mathsf{ClosedBoundedInterval}^{\uparrow}$. The chapter packages the theorem route as BEDC rows: it does not decide classical trichotomy, use Heine–Borel or open-cover compactness, select a zero, identify quotient real names, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3676.1 (Intermediate value theorem carrier)**.** *An* $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, J, F, A, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source interval row, exposing lower and upper endpoint reads, endpoint order, containment ledgers, and interval windows.*
2. *J is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint-location row used by the sign route, carrying the same endpoint surface as displayed interval data.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row for the displayed map on the interval.*
4. *A and B are the endpoint sign rows, recording the inspected negative-side and positive-side evidence as finite real-name reads rather than as a trichotomy decision.*
5. *S stores the* $\mathsf{StreamName}^{\uparrow}$ *windows used to inspect endpoint, midpoint, and branch values along the theorem route.*
6. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readbacks attached to those windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the located sign-transition output, reached only after I, J, F, A, B, S, R have been displayed.*
8. *H stores componentwise* hsame *transport for the interval, location, map, sign, window, readback, seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from an intermediate-value request through I, J, F, A, B, S, R, E.*
10. *P is the* Pkg *provenance over* $\mathsf{IntermediateValueTheorem}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, J, F, A, B, S, R, E, H, C, P, N$. It has no coordinate for classical trichotomy, Heine–Borel compactness, open-cover compactness, a selected zero, quotient real equality, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremTasteGate single carrier alignment`

Theorem 3676.2 (Intermediate value theorem NameCert obligations)**.** *For every accepted* IntermediateValueTheore *carrier*

$$T = (I, J, F, A, B, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed interval and endpoint-location rows, continuous-map admission through F, endpoint sign exposure through A and B, finite-window exposure through S, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremCarrier namecert obligations`

Proof. Project the carrier of Definition 3676.1. The accepted packet exposes the closed bounded interval row I, the located endpoint row J, the continuous-map row F, the sign rows A, B, the finite windows S, the regular readbacks R, and the terminal seal row E.

The theorem route is constructive because every sign and value read is a displayed window-readback pair. The $\mathsf{StreamName}^{\uparrow}$ row supplies the finite observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational readback, and the $\mathsf{Real}^{\uparrow}$ row receives only the sealed sign-transition output.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for classical trichotomy, Heine–Borel or open-cover compactness, a selected zero, quotient real equality, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 3676.3 (Intermediate value theorem finite sign-route handoff)**.** *Every Real-facing consumer of an accepted* $\mathsf{IntermediateValueTheorem}^{\uparrow}$ *packet factors through*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longrightarrow \mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only the displayed interval row, endpoint-location row, continuous-map row, endpoint sign rows, finite windows, regular readbacks, Real seal, componentwise transport, replay, provenance, and local naming.

Proof. Start with a Real-facing theorem read. The carrier definition places the terminal seal in row E, and E is downstream of the interval, location, map, sign, window, and readback rows.

Trace the read backward through the finite route. The closed interval row I and located endpoint row J supply the domain surface, the continuous-map row F supplies the displayed modulus and graph data, and the endpoint sign rows A, B are inspected only through S and R before the seal is reached.

Finally H, C, P, N transport, replay, package, and name the same displayed route. There is no second coordinate for a selected zero, quotient real equality, classical trichotomy, Heine–Borel compactness, open-cover compactness, host semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so every accepted handoff reaches $\mathsf{Real}^{\uparrow}$ only through the displayed rows. □

Falsifiable prediction: Every accepted $\mathsf{IntermediateValueTheorem}^{\uparrow}$ Real-facing read factors through a ClosedBoundedInterval source row, a LocatedInterval endpoint-location row, a ContinuousMap graph-and-modulus row, endpoint sign rows, StreamName finite windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: closedboundedinterval, locatedinterval, continuousmap, streamname, regseqrat, real: $\mathsf{IntermediateValueTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the theorem-level interval source, endpoint-location surface, continuous-map row, endpoint sign ledger, finite window route, regular rational readback, terminal seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{IntermediateValueTheorem}^{\uparrow}$ packet is generated from a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ source row, a $\mathsf{LocatedInterval}^{\uparrow}$ endpoint-location row, a $\mathsf{ContinuousMap}^{\uparrow}$ graph-and-modulus row, endpoint sign rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3676.1, Definition 3676.2, and Definition 3676.3 seed the constructive intermediate-value theorem route as a finite NameCert packet over interval, map, sign, window, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntermediateValueTheorem}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntermediateValueTheoremUp.IntermediateValueTheoremTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close classical trichotomy, Heine–Borel compactness, open-cover compactness, selected zeros, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign visibility, continuous-map routing, finite-window readback, Real-seal handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.IntermediateValueTheor

Chapter 3677

A Concrete Naming Certificate for $\mathsf{FirstExitChannel}^{\uparrow}$

The $\mathsf{FirstExitChannel}^{\uparrow}$ packet realizes the first-exit channel geometry of Definition 9462.1480 as finite BEDC NameCert data. It reads a killed bit-flip walk state through origin state, flipped bit, outside target, target class, transition ledger, transport, replay, provenance, and local naming before any first-exit geometry claim is exported. The packet is a finite science-facing certificate; it is not a bijection with the six-boundary-tile coverage, not a host biological law, not an optimizer for unseen codon tables, and not an unrestricted stochastic-process oracle.

The transition ledger is consumed beside chapter 3470, chapter 3478, chapter 3463, and chapter 3625 because those chapters supply the codon-window partition, trigger activation, cube adjacency, and post-median shell consumer rows.

Falsifiable prediction: Every accepted $\mathsf{FirstExitChannel}^{\uparrow}$ read must display origin state, flipped bit, outside target, target class, transition ledger, transport, replay, provenance, and local naming before any first-exit geometry claim is exported.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, post-median-gate-shell, punctured-cube-skeleton: six-boundary-tile coverage names the codon-window partition, window-coverage trigger algebra names gate activation, post-median gate shell names the downstream shell consumer, and the punctured-cube skeleton names cube adjacency. None is bijective to $\mathsf{FirstExitChannel}^{\uparrow}$ because this packet binds origin row, flipped-bit row, outside-target row, target-class row, transition ledger, transport, replay, provenance, and local naming as one accepted first-exit surface.

Definition 3677.1 (First-exit channel carrier)**.** *A* $\mathsf{FirstExitChannel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, B, T, K, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the origin-state row inside the killed median-closure domain.*
2. *B is the flipped-bit row naming the bit coordinate through which the walk exits.*
3. *T is the outside-target row naming the codon reached after the bit flip.*
4. *K is the target-class row recording the standard genetic-code class of T.*
5. *L is the transition ledger binding O, B, T, and K to the displayed killed-walk boundary half-edge.*

6. *H stores componentwise* hsame *transport for origin, bit, outside target, target class, transition ledger, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the first-exit request through O, B, T, K, L, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FirstExitChannel}^{\uparrow}$*, the six-boundary-tile coverage, the window-coverage trigger algebra, the post-median gate shell, the punctured-cube skeleton,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, B, T, K, L, H, C, P, N$. It has no coordinate for global codon-table optimality, unseen-window extrapolation, host biochemical causality, an unrestricted Markov-chain oracle, or bridge checking.

Lean4 (definition) `BEDC.Derived.FirstExitChannelUp.FirstExitChannelUp`

Lean4 ✓ `BEDC.Derived.FirstExitChannelUp.FirstExitChannelTasteGate single carrier alignment`

Theorem 3677.2 (First-exit channel NameCert obligations)**.** *For every accepted* $\mathsf{FirstExitChannel}^{\uparrow}$ *carrier*

$$F = (O, B, T, K, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$ *surface consists exactly of carrier admission for the origin-state row O, flipped-bit row B, outside-target row T, target-class row K, transition ledger L, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

Lean4 ✓ `BEDC.Derived.FirstExitChannelUp.FirstExitChannelNamecertObligations`

Proof. Project the carrier of Definition 3677.1. The projection exposes origin, bit, outside target, target class, and transition rows before it exposes the structural rows.

The transition ledger L is the only row that binds the four observational rows into a first-exit boundary half-edge. Thus a consumer cannot replace the displayed exit by an unrelated codon adjacency or by an unrestricted stochastic-process read.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for global codon-table optimality, unseen-window extrapolation, or host biochemical causality, the displayed rows exhaust the NameCert obligations. □

Theorem 3677.3 (First-exit channel geometry readback)**.** *Every accepted first-exit geometry read through* $\mathsf{FirstExitChannel}^{\uparrow}$ *factors through*

$$O, B, T, K \longmapsto L \longmapsto H, C, P, N.$$

Consequently the read may export origin-state distribution rows, flipped-bit rows, outside-target rows, and target-class rows only after the transition ledger and structural NameCert rows are displayed.

Proof. Start with the four observational rows O, B, T, K. They name the origin state, flipped bit, outside target, and target class, but they do not by themselves assert first-exit geometry.

The transition ledger L binds those rows to the killed-walk boundary half-edge. This is the point where the packet distinguishes a first exit from an arbitrary bit flip or a free codon-class lookup.

Finally H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Any geometry claim that bypasses one of these rows, or that asserts an unseen-window optimizer or host biochemical cause, is outside the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}$.

Constructive story: A $\mathsf{FirstExitChannel}^{\uparrow}$ packet is generated from origin-state rows, flipped-bit rows, outside-target rows, target-class rows, a transition ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3677.1, Definition 3677.2, and Definition 3677.3 seed the finite first-exit channel carrier, NameCert obligations, and geometry readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FirstExitChannel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global codon-table optimality, unseen-window extrapolation, host biochemical causality, unrestricted Markov-chain semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, origin-bit-target-class ledger exactness, transition replay, provenance, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.FirstExitChannelUp.TasteGate.

Chapter 3678

A Concrete Naming Certificate for $\mathsf{BoundaryShadowTrigger}^{\uparrow}$

Following finite Markov-chain first-exit arguments and the stopped-walk boundary convention, the $\mathsf{BoundaryShadowTrigger}^{\uparrow}$ packet records the narrow trigger surface by which a killed bit-flip walk exposes a boundary half-edge before any first-exit conclusion is read. It sits below the first-exit channel packet of chapter 3677, reads the first-exit channel geometry of Definition 9462.1480, and consumes the six-boundary-tile coverage of chapter 3470, the punctured-cube skeleton of chapter 3463, and the window-coverage trigger algebra of chapter 3478. The packet names only the finite origin row, exit bit, outside target, target class, boundary-shadow row, trigger ledger, transport, replay, provenance, and local naming rows; it does not assert a global stochastic-process oracle, codon-table optimality, unseen-window extrapolation, host biochemical causality, or bridge checking.

Definition 3678.1 (BoundaryShadowTrigger carrier)**.** *A* $\mathsf{BoundaryShadowTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (O, B, X, K, W, L, H, C, P, N),$$

with the following displayed rows:

1. *O is the origin-state row inside the killed median-closure domain.*
2. *B is the exit-bit row naming the coordinate through which the stopped walk leaves the displayed window.*
3. *X is the outside-target row reached by the boundary half-edge.*
4. *K is the target-class row read from the standard genetic-code class of X.*
5. *W is the boundary-shadow row recording that O, B, X, K are inspected only through the displayed half-edge.*
6. *L is the trigger ledger binding O, B, X, K, W into a first-exit trigger event.*
7. *H stores componentwise* hsame *transport for the observational, shadow, ledger, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from the boundary-shadow trigger request through $O, B, X, K, W, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowTrigger}^{\uparrow}$, $\mathsf{FirstExitChannel}^{\uparrow}$, *the six-boundary-tile coverage, the punctured-cube skeleton, the window-coverage trigger algebra,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowTrigger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, B, X, K, W, L, H, C, P, N$. It has no coordinate for a free codon-class lookup, unrestricted Markov-chain semantics, global codon-table optimality, unseen-window extrapolation, host biochemical causality, or bridge checking.

Theorem 3678.2 (BoundaryShadowTrigger NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowTrigger}^{\uparrow}$ *carrier*

$$T = (O, B, X, K, W, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowTrigger}^{\uparrow}}$ *surface consists exactly of carrier admission for the origin-state row O, exit-bit row B, outside-target row X, target-class row K, boundary-shadow row W, trigger ledger L, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

Every first-exit-channel consumer must read W and L before using O, B, X, K as a boundary trigger. No consumer receives a free codon-class lookup, unrestricted Markov-chain semantics, global codon-table optimality, unseen-window extrapolation, host biochemical causality, or bridge checking.

Proof. Project the carrier of Definition 3678.1. The rows expose the origin, exit bit, outside target, target class, boundary shadow, trigger ledger, transport, replay, provenance, and local naming data in the order available to the classifier.

The boundary-shadow row W is the only row that marks the observational tuple O, B, X, K as a half-edge read rather than as a free lookup. The trigger ledger L then binds that half-edge to the stopped-walk first-exit event before any downstream first-exit-channel consumer can inspect it.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for unrestricted Markov-chain semantics, global codon-table optimality, unseen-window extrapolation, host biochemical causality, or bridge checking, the local NameCert obligations are exhausted by the stated finite rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowTrigger}^{\uparrow}$ packet is generated from origin-state rows, exit-bit rows, outside-target rows, target-class rows, boundary-shadow rows, trigger ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundaryShadowTrigger}^{\uparrow}}$, seed)

Scope closed: Definition 3678.1 and Definition 3678.2 seed the finite boundary-shadow trigger carrier and NameCert obligation surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundaryShadowTrigger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted Markov-chain semantics, global codon-table optimality, unseen-window extrapolation, host biochemical causality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, origin-bit-target-class shadow exactness, trigger-ledger replay, provenance, local naming, and TasteGate instances for BEDC.Derived.BoundaryShadowTriggerUp.

Chapter 3679

A Concrete Naming Certificate for $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$

Following Whitney's finite-jet extension theorem and the constructive finite-row presentation of differentiability data, the $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ packet records a finite jet family over a closed carrier together with the compatibility estimates needed by extension consumers. It sits over the real and order-facing rows of chapter 38 and chapter 766, the metric and ball rows of chapter 57 and chapter 1450, the derivative-facing surface of chapter 122, and the Sobolev trace boundary of chapter 3647. The packet is a finite NameCert carrier for jet compatibility and consumer routing; it is not a global smooth extension operator, a selected extension function, host compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3679.1 (Whitney finite-jet extension carrier)**.** *A* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (X, F, J, A, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *or Euclidean metric carrier whose distance rows supply the only spatial comparisons used by the packet.*
2. *F is the closed-set classifier row naming the finite observation domain on which jets are read.*
3. *J is the finite jet table, containing the displayed value and derivative rows indexed by the admitted multiindex budget.*
4. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *comparison ledger for the Whitney difference quotients, radius bounds, and finite error budgets.*
5. *E is the local extension-admissibility ledger, recording that every requested jet comparison is routed through F, J, A and the displayed metric-ball scales.*
6. *R is the readback row for derivative-facing and Sobolev-trace-facing consumers.*
7. *H records componentwise* hsame *transport for the carrier, closed-set classifier, jet table, order ledger, extension-admissibility ledger, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite jet request through $X, F, J, A, E, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, F, J, A, E, R, H, C, P, N$. It has no coordinate for a host smooth function, a selected extension, compactness, arbitrary choice, quotient equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3679.2 (Whitney finite-jet NameCert obligations)**.** *For every accepted* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier $W = (X, F, J, A, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$ *surface consists exactly of the finite rows named in Definition 3679.1: carrier admission for $X, F, J, A, E, R, H, C, P, N$, closed-set classifier stability, finite jet-table stability, order-ledger comparison for the Whitney estimates, extension-admissibility replay through metric-ball scales, derivative and Sobolev trace readback, transport, provenance, and local naming. No obligation row stores a host extension function, a global smoothness theorem, a selected representative, compactness, choice, quotient equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.WhitneyExtensionFiniteJetUp.TasteGate.WhitneyExtensionFiniteJetTasteGate`
`single carrier alignment`

Proof. Project the accepted carrier of Definition 3679.1. The only geometric data are X and F, and the only analytic data are the finite jet table J, the comparison ledger A, the extension-admissibility row E, and the readback row R.

The NameCert obligations are therefore rowwise. A finite jet request first names the metric carrier and closed-set classifier, then opens the jet table and the order-facing comparison ledger before any admissibility claim is available.

The admissibility row E is the only coordinate that relates two jet reads at a displayed scale. It forces every difference-quotient and remainder comparison through F, J, A and the relevant metric-ball radius rather than through a host extension function.

Finally, H, C, P, N transport, replay, package, and name exactly the same route. Since the carrier contains no coordinate for a selected smooth extension, compactness, arbitrary choice, quotient equality, trichotomy, or ambient Real completeness, none of those objects can become a local obligation. □

Theorem 3679.3 (Whitney finite-jet consumer boundary)**.** *Every* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{SobolevTrace}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, *or* $\mathsf{RealAlgOrder}^{\uparrow}$ *consumer reading an accepted* $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ *carrier must factor through the displayed route*

$$X, F \longmapsto J, A \longmapsto E, R \longmapsto H, C, P, N.$$

The consumer receives finite jet compatibility, order-facing estimates, and derivative or trace readback only along this route. It receives no global extension operator, no selected smooth function, no host compactness theorem, no arbitrary choice principle, no quotient equality, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a consumer read. By Definition 3679.1, the read can inspect only the metric carrier X, the closed-set classifier F, the jet table J, the comparison ledger A, and the admissibility row E before it reaches derivative-facing or trace-facing output.

The row R supplies the public readback boundary. It exposes the finite derivative and trace-facing records already justified by E, but it does not introduce a host smooth function or select an extension outside the displayed rows.

The structural rows preserve the same route. Componentwise transport and continuation replay move the packet through H and C, while P and N record provenance and local naming. Thus every downstream consumer remains inside the finite Whitney jet surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3679.1, Definition 3679.2, and Definition 3679.3 seed the finite Whitney jet packet over metric carriers, closed-set classifier rows, jet tables, order-facing estimates, and derivative or Sobolev trace readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the classical global Whitney theorem, a selected smooth extension operator, compactness, arbitrary choice, quotient equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, jet-table stability, order-ledger comparison, extension-admissibility replay, derivative or trace readback, transport, provenance, and local naming for BEDC.Derived.WhitneyExtensionFinite.

Constructive story: A $\mathsf{WhitneyExtensionFiniteJet}^{\uparrow}$ packet is generated from a MetricUp carrier, a closed-set classifier row, a finite jet table, a RealAlgOrderUp comparison ledger, a local extension-admissibility row, derivative or Sobolev trace readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3680

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionKernel}^{\uparrow}$

Following Hausdorff and Bourbaki-style completion language, the $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ packet records the separated Cauchy-completion boundary as finite BEDC data. It sits between the separated metric classifier of chapter 1411, the complete metric service of chapter 128, the Cauchy-filter completion route of chapter 1569, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Hausdorff completion boundary of chapter 402. The packet names the zero-distance kernel that separates completion reads before any quotient representative, host equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem may be consumed.

Definition 3680.1 (Hausdorff completion kernel carrier)**.** *A* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, Z, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the Cauchy-completion thread row, recording the inspected Cauchy filter, Cauchy sequence, or regular Cauchy name requests that enter the completion boundary.*
2. *Z is the separated zero-distance row supplied by* $\mathsf{SeparatedMetric}^{\uparrow}$*, recording when two displayed completion reads have zero carried distance.*
3. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row, keeping filter-completion requests tied to the same finite Cauchy thread.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing only regular rational windows read from Q and F.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, Z, F, R have been displayed.*
6. *H records componentwise* hsame *transport for the Cauchy thread, zero-distance row, filter-completion handoff, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a separated completion request through Q, Z, F, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{HausdorffCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, Z, F, R, E, H, C, P, N$. It has no coordinate for quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionKernelUp.HausdorffCompletionKernelTasteGate single carrier alignment`

Theorem 3680.2 (Hausdorff completion kernel NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionKe}$ *carrier $K = (Q, Z, F, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, Cauchy-completion thread exposure through Q, separated zero-distance exposure through Z, Cauchy-filter completion handoff through F, regular rational readback through R, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionKernelUp.TasteGate.HausdorffCompletionKernelTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3680.1. The accepted packet exposes the Cauchy thread, separated zero-distance boundary, filter-completion handoff, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows.

The separated completion route is fixed by Q, Z, F, R, E. The Cauchy thread Q supplies the inspected completion source, Z supplies the zero-distance classifier, F binds Cauchy-filter completion reads to that source, R gives the regular rational readback, and E seals the resulting Real-facing boundary.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since the carrier has no coordinate for quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot belong to the local NameCert surface. □

Theorem 3680.3 (Hausdorff completion kernel nonescape)**.** *Every separated completion consumer of an accepted* $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ *packet factors through*

$$\mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HausdorffCompletion}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

In particular, equality of completed reads is exposed only by the displayed zero-distance row Z and its transported replay through H, C, P, N.

Proof. Start with a consumer that requests equality or reuse of a completed read. The carrier exposes no equality coordinate before the Cauchy thread Q and the separated zero-distance row Z have been read.

Next read the completion and real-facing rows. The handoff F ties the request to the Cauchy-filter completion route, R gives the regular rational readback used by the real-name boundary, and E is downstream of those rows.

Finally replay the structural data. The rows H, C, P, N preserve and name the same finite route, while the $\mathsf{HausdorffCompletion}^{\uparrow}$ provenance identifies the separated completion boundary. No quotient representative, host equality, selected limit, countable-choice schedule, detached Real equality, or ambient completeness coordinate is available to the consumer. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ separated completion read factors through a Cauchy-completion thread row, a separated zero-distance row, a CauchyFilterCompletion handoff, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: separatedmetric, completemetric, cauchyfiltercompletion, regseqrat, real, hausdorffcompletion: $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cauchy thread, zero-distance kernel, filter-completion handoff, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionKernel}^{\uparrow}$ packet is generated from a Cauchy-completion thread row, a separated zero-distance row, a CauchyFilterCompletion handoff, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3680.1, Definition 3680.2, and Definition 3680.3 seed the separated Cauchy-completion kernel over displayed completion, zero-distance, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient representatives, host completion equality, selected limits, countable choice, detached Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated zero-distance exactness, Cauchy-filter handoff, regular readback, Real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.HausdorffCompletionKernelUp.

Chapter 3681

A Concrete Naming Certificate for RadialWobbleSpectrum$^{\uparrow}$

The RadialWobbleSpectrum$^{\uparrow}$ packet realizes the radial spectral surface of Definition 9462.1448 as finite BEDC NameCert data. It consumes the boundary-tile coverage packet of chapter 3470, the support-closure algebra of chapter 3478, the hyperbolic metric reading of chapter 3474, and the substrate trace invariant of chapter 2360. The packet names radial-shell, spectral-hinge, wobble-edge, and trigger-ledger rows; it is not a biological code table, stochastic evolutionary model, quotient codon semantics, or hidden spectral oracle.

Definition 3681.1 (Radial wobble spectrum carrier)**.** *A* RadialWobbleSpectrum$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, S, H, W, L, M, A, Q, P, N)$$

with the following displayed rows:

1. *T is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-tile coverage row, exposing the finite trigger support read by the radial packet.*
2. *S is the radial-shell row, recording the six displayed states $X_3, X_2, X_1, X_0, Y_R, Y_Y$ and their finite adjacency counts.*
3. *H is the spectral-hinge row, recording the cubic hinge $r^3 - 12r^2 + 26r - 9$ and the Perron read attached to the radial matrix.*
4. *W is the wobble-edge row, recording the f_3 self-loop contribution and its normalized contribution to the killed bit-flip spectrum.*
5. *L is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *trigger ledger, binding support closure, residue accounting, blocker rows, and coverage thresholds to the same radial read.*
6. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row used only as a boundary geometry witness for the radial shells.*
7. *A is the* SubstrateTraceInvariant$^{\uparrow}$ *substrate trace row, preserving the source trace across shell, hinge, wobble, and trigger reads.*
8. *Q records componentwise* hsame *transport and displayed* Cont *replay for T, S, H, W, L, M, A.*
9. *P records* Pkg *provenance over* RadialWobbleSpectrum$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgeb HyperbolicMetric$^{\uparrow}$, SubstrateTraceInvariant$^{\uparrow}$, *the source vision of Definition 9462.1448,* BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, S, H, W, L, M, A, Q, P, N$. It has no coordinate for hidden biological semantics, stochastic transition laws, quotient codon equality, infinite spectral computation, or a detached Perron oracle.

Lean4 ✓ `BEDC.Derived.RadialWobbleSpectrumUp.RadialWobbleSpectrumTasteGate single carrier alignment`

Theorem 3681.2 (Radial wobble spectrum NameCert obligations)**.** *For every accepted* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *carrier $R = (T, S, H, W, L, M, A, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}$ *surface consists exactly of boundary-tile coverage admission through T, radial-shell exposure through S, spectral-hinge exactness through H, wobble-edge accounting through W, trigger-ledger exposure through L, hyperbolic metric boundary reading through M, substrate trace preservation through A, componentwise transport and replay through Q, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3681.1. The accepted packet exposes only the boundary-tile coverage, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, metric, substrate-trace, transport, replay, provenance, and naming rows.

The radial spectrum is carried by S, H, W. The shell row records the six states and adjacency counts, the hinge row records the cubic spectral constraint, and the wobble row records the f_3 contribution before any spectral read may be exported.

The support and geometry rows are T, L, M, A. Boundary-tile coverage and trigger algebra bind the finite support surface, while the hyperbolic metric and substrate trace rows preserve the geometric and trace witnesses without adding a hidden spectral oracle.

Finally Q, P, N transport, replay, source, and name the same finite route. Since no coordinate records biological semantics, stochastic transition laws, quotient codon equality, infinite spectral computation, or detached Perron data, none of those objects belongs to the local NameCert surface. □

Theorem 3681.3 (Radial wobble spectrum trigger nonescape)**.** *Every concrete radial wobble stability read of an accepted* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *packet factors through*

$$\mathsf{SixBoundaryTileCoverage}^{\uparrow} \longrightarrow \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow} \longrightarrow \mathsf{HyperbolicMetric}^{\uparrow} \longrightarrow \mathsf{SubstrateTraceInvariant}^{\uparrow}$$

The radial-shell row S, spectral-hinge row H, wobble-edge row W, and trigger ledger L must be displayed before the packet may claim radial wobble stability.

Proof. Start with the boundary-tile support. The row T supplies finite trigger coverage, and L records the support closure and residue ledger from the trigger algebra.

Next read the radial spectral data. The shell row S gives the six-state adjacency surface, the hinge row H gives the spectral cubic, and W records the wobble self-loop contribution. Removing any one of these rows leaves no carrier coordinate for the claimed radial wobble stability.

Finally read the boundary geometry and trace rows. The hyperbolic metric row M and substrate trace row A preserve the geometric and source-trace witnesses, while Q, P, N transport, replay, source, and name the same route. Thus the packet cannot export a hidden biological table, stochastic transition law, quotient codon equality, infinite spectral computation, or detached Perron oracle. □

Falsifiable prediction: The packet fails if the spectral-hinge row is removed while the same carrier still claims radial wobble stability.

Independence witness: `six_boundary_tile_coverage`, `window_coverage_trigger_algebra`, `hyperbolic_metric`, `substrate_trace_invariant`: the carrier has a radial spectral row and trigger ledger not bijective to any single sibling packet.

The later Lean route for `BEDC.Derived.RadialWobbleSpectrumUp` must provide explicit Nontrivial RadialWobbl and FieldFaithful $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ instances preserving $T, S, H, W, L, M, A, Q, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ packet is generated from boundary-tile coverage, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, hyperbolic metric, substrate trace, componentwise transport, displayed replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3681.1, Definition 3681.2, and Definition 3681.3 seed the finite radial wobble spectrum packet over boundary-tile, radial-shell, spectral-hinge, wobble-edge, trigger-ledger, geometry, and trace rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RadialWobbleSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological table identity, stochastic evolutionary modeling, quotient codon equality, infinite spectral computation, detached Perron oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radial-shell exactness, spectral-hinge preservation, wobble-edge accounting, trigger-ledger nonescape, Nontrivial RadialWobbleSpectrumUp, and FieldFaithful RadialWobbleSpectrumUp for BEDC.Derived.RadialWobbleSpectrumUp.

Chapter 3682

A Concrete Naming Certificate for $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$

Following Schwarz–Pick and Ahlfors' hyperbolic metric reading of holomorphic disk maps, the $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ packet names disk-map contraction as finite BEDC data. It is grounded in the unit disk carrier of chapter 377, the Poincare disk transport route of chapter 3455, the hyperbolic metric certificate of chapter 3474, the boundary-dynamics route of chapter 3575, and the observer-expansion vision of chapter 9463. The packet certifies finite disk-map, metric comparison, boundary refusal, transport, provenance, and NameCert rows; it is not an ambient theorem about all analytic self-maps of a completed complex disk.

Definition 3682.1 (Schwarz-Pick disk-contraction carrier)**.** *A* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, F, M, K, B, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for displayed source and target disk points.*
2. *F is the finite disk-map row, recording how the displayed points are transported by the named map.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *readout row for source and target point pairs.*
4. *K is the contraction-comparison row stating that the target hyperbolic readout is no larger than the source readout on the displayed pair.*
5. *B is the boundary-refusal row, separating interior disk-map comparison from Denjoy–Wolff or far-end boundary dynamics.*
6. *R is the route row linking finite disk coordinates, map transport, metric readout, comparison, and boundary refusal.*
7. *H stores componentwise* hsame *transport for disk coordinates, map rows, metric readouts, comparison rows, boundary refusals, route rows, replay rows, provenance rows, and local naming rows.*
8. *C stores displayed* Cont *replay of the same disk-map and metric-comparison route.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, F, M, K, B, R, H, C, P, N$. It has no coordinate for a host holomorphic function space, completed complex disk equality, boundary fixed-point selection, open-cover compactness, or unlogged analytic continuation.

Lean4 ✓ `BEDC.Derived.SchwarzPickDiskContractionUp.SchwarzPickDiskContractionTasteGate single carrier alignment`

Theorem 3682.2 (Schwarz-Pick disk-contraction NameCert obligations)**.** *For every accepted* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *carrier*

$$S = (D, F, M, K, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$ *surface consists exactly of carrier admission for displayed disk coordinates, finite disk-map rows, hyperbolic metric readouts, contraction-comparison rows, boundary-refusal rows, route rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3682.1. The projection exposes disk coordinates D, the finite map row F, hyperbolic metric readouts M, the contraction comparison K, and the boundary-refusal row B.

The comparison is read only through R. Thus a consumer first displays the source and target disk coordinates, applies the named finite disk-map row, reads the hyperbolic metric on both displayed pairs, and then consumes the no-expansion comparison.

Transport and replay are confined to H and C. The provenance and local naming rows P, N record exactly the same route, and the carrier has no coordinate for host analytic function spaces, completed disk equality, boundary fixed-point selection, compactness, or unlogged continuation. These rows therefore exhaust the NameCert obligations. □

Theorem 3682.3 (Schwarz-Pick hyperbolic-metric handoff)**.** *Every hyperbolic-metric consumer of an accepted* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *packet reaches the contraction verdict only through the ordered handoff*

$$\mathsf{UnitDisk}^{\uparrow} \longrightarrow \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longrightarrow \mathsf{HyperbolicMetric}^{\uparrow} \longrightarrow \mathsf{SchwarzPickDiskContraction}^{\uparrow}$$

displayed by $D, F, M, K, B, R, H, C, P, N$. The handoff supplies finite disk-map transport, source and target metric readouts, contraction comparison, boundary refusal, provenance, and NameCert data, and it supplies no ambient analytic self-map theorem or boundary fixed-point conclusion.

Proof. Start with the disk-coordinate row D and the finite map row F. These rows display the only source and target points available to the contraction comparison.

The hyperbolic-metric row M then reads the source and target distances on those displayed pairs. The comparison row K consumes those two readouts, while B keeps Denjoy–Wolff boundary dynamics and far-end fixedness outside the interior disk-contraction verdict.

Use the route, transport, and replay rows R, H, C to preserve the same handoff under classifier comparison. The provenance and local naming rows P, N package the finite route, so no host analytic function space, completed disk equality, selected boundary point, compactness argument, or unlogged continuation can enter the metric handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ packet is generated from UnitDisk coordinates, a finite disk-map row, HyperbolicMetric readouts, a contraction-comparison row, boundary refusal, a route row, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3682.1, Definition 3682.2, and Definition 3682.3 seed the finite disk-contraction carrier, NameCert obligations, and hyperbolic-metric handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwarzPickDiskContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert a host holomorphic-function theorem, completed disk equality, selected boundary fixed point, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-map exactness, hyperbolic metric readback, contraction comparison, boundary refusal, route preservation, transport stability, replay, provenance, and local naming for BEDC.Derived.SchwarzPickDiskContractionUp.

Chapter 3683

A Concrete Naming Certificate for $\mathsf{PoincareTriangleDefect}^{\uparrow}$

The $\mathsf{PoincareTriangleDefect}^{\uparrow}$ packet realizes the Poincare disk representation of chapter 9463 as finite NameCert data. It reads disk-point rows, geodesic-side rows, hyperbolic metric rows, cross-ratio comparison rows, and terminal $\mathsf{Real}^{\uparrow}$ seals through the existing Poincare disk phase-transport, hyperbolic metric, geodesic boundary, hyperbolic cross-ratio, and real-name certificates. The packet names a finite triangle-defect surface, not a host model of the hyperbolic plane, an ambient angle measure, a selected ideal boundary representative, a global field instance, or a completed real-geometric quotient.

Definition 3683.1 (Poincare triangle defect carrier)**.** *A* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, G, M, X, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-point row, displaying three accepted Poincare disk readouts and the local incidence data needed to form a triangle.*
2. *G is the geodesic-side row, read through the boundary-geodesic route and recording the three displayed side carriers.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row used to read side and angle-facing metric comparisons.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the finite cross-ratio comparison data used to orient the defect reading.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed defect value or defect-bound read.*
6. *H records componentwise* hsame *transport for D, G, M, X, R, C, P, N and the local triangle-defect packet shape.*
7. *C records displayed* Cont *replay from disk-point admission through geodesic-side admission, metric reading, cross-ratio reading, and the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{PoincareTriangleDefect}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, M, X, R, H, C, P, N$. It has no coordinate for a host hyperbolic plane, global angle measure, selected boundary representative, field-completion quotient, Lean TasteGate instance, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PoincareTriangleDefectUp.PoincareTriangleDefectTasteGate single carrier alignment`

Theorem 3683.2 (Poincare triangle defect NameCert obligations)**.** *For an accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier* $T = (D, G, M, X, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$ *obligations are exactly disk-point carrier admission, geodesic-side admission, hyperbolic-metric readout, cross-ratio comparison admission,* $\mathsf{Real}^{\uparrow}$ *defect sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3683.1. The tuple displays disk points, geodesic sides, metric rows, cross-ratio rows, a real seal, and the structural rows.

The disk and geodesic rows determine the finite triangle surface. The metric and cross-ratio rows provide the only comparison data from which the defect read can be formed.

The real seal R is downstream of those displayed geometric rows. The transport, replay, provenance, and local name preserve the same packet and introduce no ambient model.

Thus the listed rows exhaust the local certificate obligations. □

Theorem 3683.3 (Poincare triangle defect transport stability)**.** *Let* $T = (D, G, M, X, R, H, C, P, N)$ *be an accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier. Replacing disk-point rows, geodesic-side rows, hyperbolic-metric rows, cross-ratio rows, and the* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible triangle-defect route*

$$D \longmapsto G \longmapsto M \longmapsto X \longmapsto R.$$

The transported packet exports no host hyperbolic plane, ambient angle measure, selected boundary representative, global field quotient, or unrecorded geometric equality.

Proof. The accepted carrier has a single real-facing route: disk points are displayed first, geodesic sides are read from those points, the metric and cross-ratio rows compare the displayed sides, and the real seal records the defect read.

The row H transports each displayed coordinate componentwise. The row C then replays the same continuation from the transported disk and side rows through the transported metric, cross-ratio, and real-seal rows.

The provenance row P and local name N only source and name this route. Since the carrier contains no coordinate for an ambient model or selected boundary representative, the transported defect remains the same finite BEDC packet. □

Theorem 3683.4 (Poincare triangle defect sibling route)**.** *Every accepted* $\mathsf{PoincareTriangleDefect}^{\uparrow}$ *carrier reads disk rows through the Poincare disk phase-transport surface of chapter 3455, side rows through the hyperbolic geodesic boundary surface of chapter 3491, comparison rows through the hyperbolic metric surface of chapter 3474 and the hyperbolic cross-ratio surface of chapter 3948, and terminal value rows through the real-name seal of chapter 38. The local packet exports the resulting route only through* $D, G, M, X, R, H, C, P, N$ *and exports no host hyperbolic plane, ambient angle measure, selected boundary representative, or unrecorded quotient geometry.*

Proof. Project the carrier of Definition 3683.1. The disk row D is the only row from which point data may be read, so it is routed through the Poincare disk phase-transport sibling before any side or comparison consumer sees the packet.

Next read the side and comparison rows. The side row G is routed through the hyperbolic geodesic boundary sibling, while M and X are routed through the hyperbolic metric and hyperbolic cross-ratio siblings. These four sibling surfaces provide the displayed geometric input rows; none adds a host hyperbolic plane or ambient angle measure.

Finally read R, H, C, P, N. The terminal value is sealed through the RealUp sibling, and the transport, replay, provenance, and local name rows keep the same sibling route attached to the finite triangle-defect packet. Since no accepted coordinate names a selected boundary representative or quotient geometry, those objects cannot be exported by the sibling route. □

Falsifiable prediction: A candidate implementation of $\mathsf{PoincareTriangleDefect}^{\uparrow}$ that omits any one of the disk-point, geodesic-side, hyperbolic-metric, cross-ratio, or $\mathsf{Real}^{\uparrow}$ seal rows will fail to export a stable defect read under componentwise hsame transport.

Independence witness: The packet depends on the visible rows named poincare-disk-phase-transport, hyperbolic-metric, hyperbolic-geodesic-boundary, hyperbolic-cross-ratio, and real; it does not depend on a host hyperbolic plane, ambient angle measure, selected boundary representative, or global quotient geometry. Lean TasteGate support will require explicit Nontrivial and FieldFaithful instances before the geometric rows can be checked as a typed finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareTriangleDefect}^{\uparrow}$ packet is generated from disk-point rows, geodesic-side rows, HyperbolicMetric rows, HyperbolicCrossRatio rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3683.1, Definition 3683.2, Definition 3683.3, and Definition 3683.4 seed the finite Poincare disk triangle-defect packet and its sibling dependency route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareTriangleDefect}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host hyperbolic plane, ambient angle measure, selected ideal boundary representative, global field quotient, Lean TasteGate instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-point admission, geodesic-side admission, hyperbolic-metric readout, cross-ratio comparison, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.PoincareTriangleDefectUp.

Chapter 3684

A Concrete Naming Certificate for $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ packet records the finite criterion by which a displayed regular Cauchy source carries a boundedness ledger before it is sealed as a Real-completion object. It sits beside the regular rational name surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the bounded regular Cauchy route of chapter 1697, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet supplies an L10 completion-facing boundedness criterion; it is not a quotient stream, not a selected limit, not countable choice, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3684.1 (Cauchy boundedness criterion carrier)**.** *A* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the Cauchy source row, naming the regular rational sequence whose finite observations are admitted by the packet.*
2. *B is the boundedness ledger, recording the displayed initial bound, tail bound, and comparison rows that make the source a* $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *object.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source at the requested precision and tail threshold.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite window and boundedness ledger.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the precision budget, bound comparison, and tail estimate consumed by the readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, B, W, R, D have been displayed.*
7. *H stores componentwise* hsame *transport for the Cauchy source, boundedness ledger, window row, readback row, dyadic tolerance row, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a bounded Cauchy request through $S, B, W, R, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$, $\mathsf{BoundedRegularCauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, B, W, R, D, E, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected Real limit, countable choice, trichotomy, a host supremum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyBoundednessCriterionUp.CauchyBoundednessCriterionTasteGate single carrier alignment`

Theorem 3684.2 (Cauchy boundedness criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *carrier*

$$K = (S, B, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row S, boundedness ledger B,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3684.1. The analytic rows are the source S, boundedness ledger B, window row W, readback row R, dyadic tolerance row D, and Real seal E.

The boundedness obligation is read only from B together with the dyadic row D. The ledger records the displayed bound comparisons and tail estimates, while D records the precision budget consumed by those comparisons.

The regular readback is downstream of the same finite window. Thus R is a $\mathsf{RegSeqRat}^{\uparrow}$ row obtained through W, B, D, not a host sequence, quotient stream, selected limit, or completeness argument.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for countable choice, trichotomy, host suprema, quotient-stream equality, selected Real limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3684.3 (Cauchy boundedness criterion Real-completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing completion read exported by an accepted* $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ *packet factors through*

$$S \longmapsto B \longmapsto W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently a completion consumer may use the displayed Cauchy source, boundedness ledger, finite $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local structural rows, but it receives no quotient-stream equality, selected limit, countable choice, trichotomy, host supremum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the displayed source S. The source is admissible for the packet only as a Cauchy row whose boundedness information is supplied by the ledger B.

The finite observation then passes through W and D. The row W exposes the inspected stream window, while D supplies the dyadic tolerance and bound-comparison budget for the same request.

Only after those rows have been displayed does the packet expose R and then E. The row R is the regular rational readback, and E is the terminal Real seal served by that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Since none of the excluded objects occurs in the carrier, no completion-facing read can bypass the bounded finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyBoundednessCriterion}^{\uparrow}$ packet is generated from a Cauchy source row, a boundedness ledger, a StreamName finite-window row, a RegSeqRat readback row, a DyadicRatCore tolerance row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3684.1, Definition 3684.2, and Definition 3684.3 seed the finite boundedness criterion for regular Cauchy sources and Real-completion handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyBoundednessCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected Real limits, countable choice, trichotomy, host suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundedness-ledger exactness, StreamName window admission, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.CauchyBoundednessCriterionUp.

Chapter 3685

A Concrete Naming Certificate for $\mathsf{LocatedMetricSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet records a constructive metric space whose distance reads are located before they are consumed by the L10 Phase Real and Completion route. It refines the metric source of chapter 57 by forcing every distance observation through the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet supplies a finite NameCert surface for completion-facing consumers without importing host equality of points, quotient completion, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3685.1 (Located metric-space carrier)**.** *A* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric-source carrier row, including the local point classifier and the finite point-observation windows on which distance requests are opened.*
2. *M is the metric source row inherited from chapter 57; it displays the distance witness inspected by each accepted pair of point observations.*
3. *D is the located-distance witness and dyadic tolerance ledger. For each displayed distance request and positive tolerance, it records the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison data that locates the requested distance at that scale.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It packages the dyadic comparison windows in D as regular rational distance names before any real-facing consumer can read them.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *readback and seal boundary reached only after the metric source, located-distance witness, dyadic tolerance ledger, and regular rational readback have all been displayed.*
6. *H stores componentwise* hsame *transport for point observations, metric source rows, located-distance witnesses, dyadic tolerance ledgers, regular rational readbacks, real seals, replay rows, provenance rows, and local naming rows.*
7. *C stores displayed* Cont *replay from point observations to metric witnesses, from metric witnesses to located dyadic comparisons, from those comparisons to* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and from the readbacks to the* $\mathsf{Real}^{\uparrow}$ *seal.*

8. *P is the* Pkg *provenance over* $\mathsf{LocatedMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted located metric-space packets only through $X, M, D, R, E, H, C, P, N$. It has no coordinate for host equality of points, quotient real equality, quotient completion, selected limits, total trichotomy of distances, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedMetricSpaceUp.LocatedMetricSpaceTasteGate single carrier alignment`

Theorem 3685.2 (Located metric-space NameCert obligations)**.** *For every accepted* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *carrier*

$$L = (X, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric source, distance-witness readback through M, located-distance and dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3685.1. The point observations and their local classifier are read from X, and every distance request is forced through the metric source row M before locatedness data can be inspected.

The located-distance obligation is the role of D. A positive tolerance request reads the dyadic comparison window stored in the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, so the packet gives a finite locatedness witness rather than a semantic real-order decision.

The readback obligation then passes through R and E. Row R packages the same dyadic windows as regular rational distance names, and row E exposes the real seal only after that regular readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name exactly the same rows. Since the carrier contains no coordinate for host point equality, quotient equality, quotient completion, selected limits, trichotomy, choice, or ambient real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3685.3 (L10 located metric-space completion route)**.** *Every L10 Phase Real and Completion consumer that reads an accepted* $\mathsf{LocatedMetricSpace}^{\uparrow}$ *packet factors through*

$$X \longmapsto M \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N. Thus completion-facing distance reads pass through the metric source, located-distance witness, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before any completion data is consumed.*

Proof. Begin with a completion-facing distance request. The source row X supplies only displayed point observations, so the request must use the metric source row M to obtain the finite distance witness certified by the metric sibling.

Locatedness is supplied by D. The dyadic tolerance ledger binds the requested radius to the same metric witness, and it exposes only finite $\mathsf{DyadicRatCore}^{\uparrow}$ comparison data at the requested scale.

The L10 real-facing route is then determined by R and E. The regular rational readback packages the dyadic windows as distance-name data, and the real seal is reached only after that package is visible.

Rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Because no excluded quotient, selection, host equality, choice, trichotomy, or ambient-completeness coordinate appears in the carrier, every completion-facing read remains on the displayed L10 route. □

Falsifiable prediction: Every L10 completion-facing read from a $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet factors through the displayed metric source, located-distance witness, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local naming row before completion data is consumed.

Independence witness: metric, dyadicratcore, regseqrat, real: $\mathsf{LocatedMetricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, located-distance witness, dyadic tolerance ledger, RegSeqRat readback, RealUp seal, componentwise transport, continuation replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedMetricSpace}^{\uparrow}$ packet is generated from a metric source, located-distance witness, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3685.1, Definition 3685.2, and Definition 3685.3 seed the located metric-space carrier and its L10 real-and-completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host equality of points, quotient real equality, quotient completion, selected limits, total trichotomy of distances, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-distance witness exactness, dyadic tolerance ledger accounting, RegSeqRat readback, RealUp seal handoff, transport replay, and ledger refusal for BEDC.Derived.LocatedMetricSpaceU

Chapter 3686

A Concrete Naming Certificate for $\mathsf{ProperMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{ProperMetricSpace}^{\uparrow}$ packet records the constructive metric-space interface in which displayed closed metric balls carry compactness-style readback for completion consumers. It sits after the metric carrier of chapter 57, the open-ball boundary of chapter 1450, the closed-ball containment surface of chapter 3464, the compact-metric certificate of chapter 129, the bounded-family completion route of chapter 1896, and the terminal real-name seal of chapter 38. The packet is a finite closed-ball compactness ledger, not an open-cover compactness theorem, a selected finite subcover, quotient point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3686.1 (Proper metric space carrier)**.** *A* $\mathsf{ProperMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P_{\mathrm{pms}} = (X, B, R, K, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *row carrying the point classifier, distance witnesses, and metric transport surface.*
2. *B is the displayed family of* $\mathsf{ClosedMetricBall}^{\uparrow}$ *rows indexed by the accepted center-radius windows of X.*
3. *R is the radius ledger, recording nonnegative radius rows, strict-boundary handoffs to* $\mathsf{MetricBall}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *comparison readback.*
4. *K assigns compactness-style* $\mathsf{CompactMetric}^{\uparrow}$ *readback to each displayed closed ball in B.*
5. *O records compatibility between the closed-ball family and the* $\mathsf{CauchyBornology}^{\uparrow}$ *bounded-family route consumed by completion-facing readers.*
6. *H records componentwise* hsame *transport for the metric row, closed-ball family, radius ledger, compactness readback, bornology compatibility, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a proper-metric request through X, B, R, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ProperMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{ClosedMetricBall}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{CauchyBornology}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows and exposes no host subset of points, open-cover compactness object, selected compactness witness outside the displayed closed ball, quotient metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.ProperMetricSpaceUp.TasteGate.ProperMetricSpaceTasteGate single carrier alignment`

Theorem 3686.2 (Proper metric space NameCert obligations)**.** *For every accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *carrier*

$$P_{\mathrm{pms}} = (X, B, R, K, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric row* X*, closed-ball family row* B*, radius ledger* R*, compactness readback* K*, bornology compatibility row* O*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*; center-radius compatibility between* X, B, R*; compact readback exactness for each displayed ball through* K*; completion-facing bounded-family compatibility through* O*; and non-escape for host subsets, open-cover compactness, selected subcovers, quotient point equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3686.1. The metric row, closed-ball family, radius ledger, compact readback, bornology compatibility, and structural rows are all displayed before a consumer reads the local naming row.

Center-radius compatibility is forced by the order of the rows. The row X supplies the metric source, B supplies only closed balls over that source, and R supplies the radius comparisons and strict-boundary handoffs used by $\mathsf{MetricBall}^{\uparrow}$ consumers.

Compactness readback is attached ball by ball. The row K is indexed by the displayed closed-ball family B, so a consumer cannot replace it by an open-cover compactness proof or by a compactness witness for an undisplayed ball.

Finally O, H, C, P, N preserve the same route for completion-facing and bounded-family consumers. Since the carrier has no coordinate for the excluded host principles, the listed rows exhaust the NameCert obligations. □

Theorem 3686.3 (Proper metric space closed-ball compactness readback)**.** *Every compactness-style read of a displayed closed ball in an accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{ClosedMetricBall}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The $\mathsf{ClosedMetricBall}^{\uparrow}$ *row supplies the center-radius containment, the radius ledger supplies the ordered-real comparison, and the* $\mathsf{CompactMetric}^{\uparrow}$ *row supplies the compactness readback attached to that same displayed ball. The readback exports no open-cover compactness theorem, selected finite subcover, quotient point equality, host closed subset, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the closed-ball family row B of Definition 3686.1. Each accepted ball read is indexed by a center-radius window over the metric row X and by a radius comparison in R.

Read the compactness row K only after that ball is fixed. Its index is the displayed closed-ball row, so compactness-style data are attached to the same metric source, center, radius, and containment comparison.

The structural rows H, C, P, N transport, replay, source, and name this finite route. No row names an open cover, a chosen subcover, a quotient point equality, or an ambient completeness theorem. □

Theorem 3686.4 (Proper metric space ball-bornology boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{ProperMetricSpace}^{\uparrow}$ *packet reads bounded-family data through the displayed boundary*

$$\mathsf{MetricBall}^{\uparrow},\quad \mathsf{ClosedMetricBall}^{\uparrow},\quad \mathsf{CauchyBornology}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

Open-ball handoffs use the strict-radius side of R, closed-ball handoffs use B, bounded-family requests use O, and compactness-style requests use K. Thus the consumer receives a finite ball and bornology boundary rather than a host bounded set, selected bound, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Inspect the radius ledger R. Its strict-radius entries are the only route to $\mathsf{MetricBall}^{\uparrow}$ consumers, while its non-strict entries are the radius comparisons used by the $\mathsf{ClosedMetricBall}^{\uparrow}$ family B.

The bornology compatibility row O is downstream of those ball rows. A completion-facing consumer therefore reads bounded-family data only after the metric row, radius ledger, and ball family have been displayed.

Compactness-style information remains attached through K. The same transport, replay, provenance, and local naming rows preserve the boundary, so the route cannot turn into a host bounded set, selected bound, open-cover compactness proof, or ambient Real-completeness datum. □

Theorem 3686.5 (Proper metric space compact-ball route)**.** *Every completion-facing proper-metric consumer must expose the metric row X, the displayed* $\mathsf{ClosedMetricBall}^{\uparrow}$ *family B, the* $\mathsf{Real}^{\uparrow}$ *radius ledger R, the* $\mathsf{CompactMetric}^{\uparrow}$ *readback row K, the* $\mathsf{CauchyBornology}^{\uparrow}$ *bounded-family route O, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row N before reading compact-ball data.*

The compact-ball route is therefore the displayed finite packet

$$X, B, R, K, O, H, C, P, N$$

read in the order forced by Definition 3686.1. It refuses open-cover compactness, selected finite subcovers, quotient point equality, host bounded sets, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3686.1. The only available completion-facing rows are the metric source, closed-ball family, radius ledger, compact readback, bornology route, transport, replay, provenance, and local naming rows.

Use the closed-ball compactness readback theorem, Definition 3686.3, to attach compactness-style information to a displayed closed ball over X, B, R. Then compose the ball-bornology boundary of Definition 3686.4, which routes bounded-family consumers through O while keeping the compact readback attached through K.

The structural rows H, C, P, N preserve the same finite route. Since no carrier coordinate names open covers, selected finite subcovers, quotient point equality, host bounded sets, or ambient Real completeness, a compact-ball consumer cannot read any of those principles from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProperMetricSpace}^{\uparrow}$ packet is generated from a MetricUp row, a displayed ClosedMetricBallUp family, a RealUp radius ledger, CompactMetricUp readback for each displayed closed ball, CauchyBornologyUp compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3686.1, Definition 3686.2, Definition 3686.3, Definition 3686.4, and Definition 3686.5 seed the constructive closed-ball compactness ledger for completion-facing proper metric space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProperMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, selected subcovers, quotient metric equality, host bounded sets, arbitrary compactness outside displayed closed balls, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, center-radius compatibility, compact readback exactness, ball-bornology boundary routing, transport, replay, provenance, and local NameCert exactness for BEDC.Derived.ProperMetricSpace

Chapter 3687

A Concrete Naming Certificate for $\mathsf{CompleteMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive metric completion, the $\mathsf{CompleteMetricSpace}^{\uparrow}$ packet records a metric carrier together with finite Cauchy-limit service data visible to BEDC consumers. It sits beside the constructive metric surface of chapter 3673, the separated metric route of chapter 2355, the regular Cauchy completion unit of chapter 3646, and the Real-facing finite-window route of chapter 38. The packet supplies an L10 completion-facing neighbour for separated metric, Baire, and completion chapters; it is not a quotient completion, not a selected-limit principle, not countable choice, not host equality, not trichotomy, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3687.1 (Complete metric-space carrier)**.** *A* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, C, E, L, U, H, T, P, N)$$

with the following displayed rows:

1. *M is the underlying* $\mathsf{Met}^{\uparrow}$ *row, exposing only the point-name and distance-read surfaces consumed by the packet.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, recording that accepted Cauchy requests are answered through displayed limit rows.*
3. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *boundary row, used only to identify regular Cauchy names that have the same displayed tail behaviour.*
4. *L is the limit-row ledger. It names the proposed limit point, the convergence-distance rows, and the precision requests consumed by the limit service.*
5. *U is the uniform Cauchy-service route through* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; it exposes finite windows, regular rational readbacks, and the terminal Real seal in that order.*
6. *H stores componentwise* hsame *transport for the metric row, completeness row, Cauchy-equivalence boundary, limit ledger, uniform service route, replay, provenance, and local naming rows.*
7. *T stores displayed* Cont *replay from Cauchy-request inspection through M, C, E, L, U, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, C, E, L, U, H, T, P, N$. It has no coordinate for quotient completions, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteMetricSpaceUp.CompleteMetricSpaceTasteGate single carrier alignment`

Theorem 3687.2 (Complete metric-space NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *carrier*

$$K = (M, C, E, L, U, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed metric row M, the complete-metric service row C, the Cauchy-sequence equivalence boundary E, the limit-row ledger L, the uniform Cauchy-service route U, componentwise transport H, displayed replay T, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3687.1. The projection exposes exactly the metric surface M, completeness service C, Cauchy-equivalence boundary E, limit ledger L, uniform service route U, and structural rows H, T, P, N.

The completeness service is read only through C and L. A consumer receives a displayed limit row and convergence-distance ledger, not a quotient completion object or an externally selected representative.

The Cauchy boundary is confined to E, while the finite observation route is confined to U. Thus tail comparison passes through $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any limit read is exported.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, host equality, trichotomy, selected limits, quotient completions, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3687.3 (Complete metric-space Cauchy-limit route)**.** *Every Cauchy-limit read exported by an accepted* $\mathsf{CompleteMetricSpace}^{\uparrow}$ *packet factors through*

$$M \longmapsto E \longmapsto U \longmapsto C, L \longmapsto H, T, P, N.$$

Consequently separated metric, Baire, and completion-facing consumers may use only the displayed metric row, Cauchy-equivalence boundary, finite-window regular readback route, complete-metric service row, limit ledger, transport, replay, provenance, and local naming.

Proof. Begin with the metric row M. It supplies the point-name and distance-read surface, but it does not by itself export a limit object.

The Cauchy request then passes through E and U. The row E fixes the tail-equivalence boundary for the displayed Cauchy name, while U exposes the finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, and terminal $\mathsf{Real}^{\uparrow}$ seal needed for the convergence-distance route.

Only after those rows are displayed may the packet read C and L. The row C supplies the complete-metric service, and L records the proposed limit point and convergence ledger for the same request.

The rows H, T, P, N preserve that route under transport, replay, provenance, and local naming. Any route that appeals to quotient completions, selected limits, countable choice, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness has no displayed coordinate in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricSpace}^{\uparrow}$ packet is generated from a Metric row, a CompleteMetric service row, a CauchySequenceEquivalence boundary, a limit ledger, a RegSeqRat–StreamName–Real uniform service route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3687.1, Definition 3687.2, and Definition 3687.3 seed the finite complete metric-space carrier, NameCert obligation surface, and Cauchy-limit route for separated metric, Baire, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-equivalence exactness, finite-window regular readback, complete-metric service confinement, limit-ledger exactness, and non-escape for BEDC.Derived.CompleteMetricSpaceUp.

The $\mathsf{Cauchy}^{\uparrow}$ hub routes finite completion reads through the sibling lattice of chapter 126, chapter 301, chapter 38, chapter 3794, and chapter 1569 before any completed-object consumer is reached.

Chapter 3688

A Concrete Naming Certificate for $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packet records the constructive complete ordered-field interface by finite regular-Cauchy and order-algebra rows rather than by an ambient completed real line. It sits beside the terminal Real naming seal of chapter 38, the regular rational sequence packet of chapter 301, the finite stream-window surface of chapter 126, the dyadic arithmetic substrate of chapter 291, the constructive algebraic order packet of chapter 766, and the Bishop real completion route of chapter 3665. The chapter exposes the common interface downstream completion and metric chapters can cite when they need Real scalars, regular Cauchy limit rows, and ordered-field compatibility in one NameCert packet.

Definition 3688.1 (Cauchy-complete ordered-field carrier)**.** *A* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar row, read only through terminal Real name seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-precision ledger used by the regular Cauchy and order rows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row for rational windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational sequence row.*
5. *B is the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *completion route from D, S, Q into the Real seal R.*
6. *A is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *algebra-order row, recording addition, multiplication, zero, one, positivity, and order compatibility only as displayed finite ledger entries.*
7. *L is the regular-Cauchy limit ledger naming accepted Cauchy requests, precision windows, candidate limit rows, and convergence readbacks through D, S, Q, B, R.*
8. *O is the order-algebra compatibility ledger, recording that limit rows preserve the displayed ordered-field operations through A and not through host equality.*
9. *H records componentwise* hsame *transport for scalar, dyadic, stream, regular-Cauchy, completion, algebra-order, limit, compatibility, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from precision requests through regular Cauchy readback, Bishop completion, Real sealing, algebra-order compatibility, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, S, Q, B, A, L, O, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, trichotomy, open-cover compactness, selected representatives, or choice.*

Theorem 3688.2 (Cauchy-complete ordered-field NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Real scalar row R, dyadic precision ledger D, stream-window row S, regular rational sequence row Q, Bishop completion route B, algebra-order row A, regular-Cauchy limit ledger L, order-algebra compatibility ledger O, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3688.1. The projection exposes exactly the Real scalar, dyadic precision, stream-window, regular rational, Bishop completion, algebra-order, limit, compatibility, transport, replay, provenance, and naming rows.

The completion-facing rows are read only through D, S, Q, B, R, L. A consumer receives finite precision requests, finite rational windows, regular readbacks, the Bishop completion route, the terminal Real seal, and a displayed limit ledger.

The ordered-field rows are read only through A and O. They record operation and order compatibility for the same displayed limit rows, without adding host equality, quotient representatives, trichotomy, open-cover compactness, selected limits, or choice.

Finally H, C, P, N transport, replay, source, and name that finite surface. Since the carrier has no further coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 3688.3 (Regular-Cauchy limit route)**.** *Every limit read exported by an accepted* CauchyCompleteOrderedF *packet factors through the ordered route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopRealCompletion}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

and then through the displayed limit ledger L and structural rows H, C, P, N. Thus downstream completion and metric consumers may cite one finite interface for regular-Cauchy limit service, but they receive no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient equality, selected representative, trichotomy, open-cover compactness, or choice principle.*

Proof. Begin with the precision ledger D. It supplies only the positive dyadic rows inspected by the regular-Cauchy request.

The request then passes through S and Q. The row S names the finite rational windows, and Q supplies the regular rational readback for those windows.

Only after that readback may the packet use B and R. The Bishop completion row B routes the same data to the terminal Real seal R, while L records the candidate limit and convergence readbacks for the accepted request.

The rows H, C, P, N preserve, replay, source, and name the route. No excluded ambient completeness or choice coordinate appears in the carrier, so every exported limit read is confined to the stated finite path. □

Theorem 3688.4 (Order-algebra compatibility route)**.** *Every algebraic or order read exported by an accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *packet factors through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *row A together with the regular-Cauchy limit ledger L and the compatibility ledger O. Addition, multiplication, zero, one, positivity, and order comparison are available only as displayed rows compatible with the route through D, S, Q, B, R, and not as ambient host-field operations.*

Proof. Read the algebra-order row A. It is the only coordinate naming field operations and order comparison, so a consumer cannot obtain algebraic structure from the host Real carrier.

Next read the limit ledger L. The ledger ties the operation and order requests to the same regular-Cauchy route through D, S, Q, B, R used for limit service.

Use the compatibility ledger O to bind the two surfaces. It records preservation of addition, multiplication, zero, one, positivity, and order comparison for the displayed limit rows.

Transport and replay are again confined to H and C, with P and N supplying provenance and local naming. Thus the compatibility surface remains finite and supplies no quotient equality, trichotomy, open-cover compactness, selected representative, ambient completeness, or choice. □

Theorem 3688.5 (Cauchy-complete ordered-field regular-Cauchy existence)**.** *For every accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

every regular-Cauchy limit request admitted by K is presented only by a finite route through

$$D \longrightarrow S \longrightarrow Q \longrightarrow B \longrightarrow R$$

and by the limit ledger L. The existence surface therefore names limit service as displayed precision, stream-window, regular-sequence, Bishop-completion, Real-seal, and ledger rows, with no quotient equality, selected representative, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with an admitted limit request. The carrier projection gives the dyadic precision row D, stream-window row S, regular rational sequence row Q, Bishop completion route B, Real seal R, and limit ledger L as the only completion-facing coordinates.

The request is first read against D, then against the finite observation row S, then against the regular-sequence row Q. The route can enter the Real naming surface only through B and then R.

The ledger L records the accepted request, the precision windows, the candidate limit row, and the convergence readbacks for that same route. Since the carrier has no quotient, representative-selection, choice, trichotomy, or ambient-completeness coordinate, the displayed rows are the whole existence surface. □

Theorem 3688.6 (Cauchy-complete ordered-field order-limit transport)**.** *For every accepted* $\mathsf{CauchyCompleteOrder}$ *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

every algebraic and order read on a limit row of L factors through the algebra-order row A and the compatibility ledger O. The transport rows H, replay rows C, provenance row P, and local naming row N preserve that same finite route through D, S, Q, B, R, L, A, O and add no host equality, selected representative, choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read a limit row from L. Its completion data has already been confined to the route through D, S, Q, B, R.

Algebraic and order observations can then use only A and O. The row A names addition, multiplication, zero, one, positivity, and comparison, while O records that those reads are compatible with the same displayed limit rows.

The structural rows do not create a second route. The transport row H preserves the componentwise names, C replays the displayed dependencies, P records provenance, and N supplies the local certificate name. Hence order-limit transport stays on the finite rows and does not expose host-field equality, representatives, choice, trichotomy, or ambient completeness. □

Theorem 3688.7 (CauchyCompleteOrderedField scoped kernel route)**.** *For every accepted* CauchyCompleteOrdere *carrier*

$$K = (R, D, S, Q, B, A, L, O, H, C, P, N),$$

the consumer-visible kernel route consists exactly of Real scalar exposure through R, dyadic precision through D, finite $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, Bishop completion handoff through B,* $\mathsf{RealAlgOrder}^{\uparrow}$ *operations through A, regular-Cauchy limit ledger reads through L, order-algebra compatibility through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$ *row N. No accepted carrier exports ambient Real completeness, quotient streams, host field operations, trichotomy, or selected limits beyond those displayed rows.*

Proof. Project the carrier of Definition 3688.1. The projection exposes precisely the rows $R, D, S, Q, B, A, L, O, H, C$ so any consumer route must begin inside the displayed finite packet.

Apply the NameCert obligations of Definition 3688.2. They identify the Real scalar, dyadic precision, stream-window, regular-rational, Bishop completion, algebra-order, limit, compatibility, transport, replay, provenance, and local naming rows as the full obligation surface.

The completion side is then fixed by Definition 3688.3. The route to a Real-facing limit read passes through D, S, Q, B, R, L and the structural rows, with no quotient stream or selected-limit coordinate.

The algebra and order side is fixed by Definition 3688.4. Operations and order comparisons are read only through A and O against the same displayed limit ledger.

Since no carrier coordinate remains after H, C, P, N, the route refuses ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, host field operations, trichotomy, and selected limits. The scoped dependency surface is exactly the finite row list stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packet is generated from Real scalar seals, dyadic precision rows, StreamName windows, RegSeqRat readbacks, a BishopRealCompletion route, RealAlgOrder operation and order rows, regular-Cauchy limit ledgers, compatibility ledgers, componentwise transport, replay, provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 3688.1, Definition 3688.2, Definition 3688.3, Definition 3688.4, Definition 3688.5, Definition 3688.6, and Definition 3688.7 close the explicit finite scoped surface for Real, regular-Cauchy existence, order-limit transport, completion, and metric consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedField}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, trichotomy, open-cover compactness, selected representatives, choice, Lean verification, or bridge checking.

Upgrade path: Public closure requires checked consumer exports and downstream bridge rows for BEDC.Derived.CauchyCompleteOrdered

The $\mathsf{Cauchy}^{\uparrow}$ hub exposes a displayed Real and Completion route only after finite stream windows and regular rational readback rows have been made visible. This child route records the dependency on chapter 126, chapter 301, and chapter 38 without adding substantive prose to the hub.

Theorem 3688.8 (Cauchy displayed RegSeqRat route)**.** *Any* $\mathsf{Cauchy}^{\uparrow}$ *route used by the L10* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *surface must pass through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows before a* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The route carries only displayed* BHist*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows, and it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyDisplayedRegseqratRoute`

Proof. Begin with a Real- or Completion-facing Cauchy read. The finite window surface of chapter 126 is the first visible observation boundary, so the consumer cannot treat the Cauchy source as an uninspected host sequence.

Read those windows through chapter 301. The regular rational readback row supplies the displayed tail data required before any terminal seal can be used.

Finally consume the terminal row of chapter 38. The structural rows transport, replay, source, and name the same finite route, so the Cauchy handoff reaches the L10 Real and Completion surface only after StreamName windows and RegSeqRat readback have been displayed. □

Theorem 3688.9 (Cauchy L10 child dependency route)**.** *Every finite-window read exported by the* $\mathsf{Cauchy}^{\uparrow}$ *hub is consumed by sibling L10 chapters only after it passes through the child routes displayed in this chapter. The regular-rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-facing route of chapter 1569 read the* $\mathsf{Cauchy}^{\uparrow}$ *packet through finite stream windows, dyadic tolerance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. No child route exports quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the hub through its child files. The core carrier in chapter 3413 displays the stream-window, dyadic tolerance, regular readback, transport, replay, provenance, and naming rows before any downstream consumer can inspect a Cauchy object.

The sibling dependency spine then sends those rows to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The cited regular-rational and Real chapters consume only the displayed finite route; they do not receive a quotient stream, selected limit, choice schedule, trichotomy oracle, host equality coordinate, or ambient completeness proof.

The completion-facing consumer follows the same rows through the Cauchy-filter completion surface. Because the hub contains only child inputs and each child route exposes finite BHist and NameCert data, the L10 dependency path is connected through sibling chapters rather than through an unstated top-level object. □

Theorem 3688.10 (Cauchy RegSeqRat and completion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *finite-window packet consumed by* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *factors through* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route exports no quotient streams, selected limits, choice schedules, host equality, or ambient completeness.*

Proof. Start with the child dependency route above. It already displays the finite stream-window, dyadic tolerance, regular readback, Real-seal, transport, replay, provenance, and naming rows before any L10 consumer reads a Cauchy packet.

The regular rational and Real consumers are the first forward users. The chapters chapter 301 and chapter 38 read only the displayed finite windows, readback rows, tolerance rows, and terminal seal.

The completion consumer reads the same route through chapter 1506. A metric-completion request can inspect Cauchy data only after the finite windows, tolerances, readbacks, Real seal, transport, replay, provenance, and local naming rows have been displayed.

The excluded objects have no row in this finite route. Thus quotient streams, selected limits, choice schedules, host equality, and ambient completeness cannot be exported by the forward handoff. □

The L10 route consumes $\mathsf{Cauchy}^{\uparrow}$ through the finite-window surface of chapter 126, the regular readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal seal rows of chapter 38. These four child dependencies keep the Cauchy packet attached to finite stream windows, regular rational evidence, dyadic budgets, and the Real seal before any L10 consumer may read it.

Every $\mathsf{Cauchy}^{\uparrow}$ read used by Real and completion consumers factors through the same horizontal lattice: finite observation windows from chapter 126, regular-rational readback from chapter 301, dyadic tolerance rows from chapter 291, and the terminal seal of chapter 38. Downstream consumers therefore receive a displayed finite-window route before they may treat the packet as Real-facing Cauchy data.

Theorem 3688.11 (Cauchy regular-source completion dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *read consumed by an L10 completion-facing packet is public only through the displayed dependency route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first exposes the regular-rational tail budget of chapter 301, then the metric-distance comparison used by the $\mathsf{MetricCompletion}^{\uparrow}$ *boundary of chapter 1506, and only then the terminal real seal of chapter 38. It exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child carriers. The hub contributes only displayed finite rows, so a completion-facing consumer must first inspect the regular-rational tail budget that names the tail data and tolerance requests.

Next compare distances at the metric-completion boundary. The $\mathsf{MetricCompletion}^{\uparrow}$ consumer receives the same displayed regular-source route and metric-distance comparison; it does not receive a quotient representative or a selected completed point.

Finally consume the terminal $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the regular tail budget and the metric-completion comparison have been displayed, the public route is a dependency edge through sibling chapters rather than an ambient completeness principle. □

Theorem 3688.12 (Cauchy filter-base net-convergence route)**.** *Every completion-facing consumer of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet that reads Cauchy finite-window evidence through filter or net interfaces must first pass through the displayed routes of chapter 1601, chapter 1915, and chapter 1667. The route exposes the same finite windows, tolerance ledger, tail comparisons, componentwise transport, replay, provenance, and local naming rows before any filter-base or net-convergence*

completion consumer inspects the packet. It exports no selected limit, quotient completion equality, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or detached net convergence theorem.*

Proof. Project the accepted $\mathsf{Cauchy}^{\uparrow}$ packet to its finite-window and tolerance rows. These are the only rows available to a downstream filter or net consumer.

The filter-base read is routed through the finite basis surface named by chapter 1601 and the Cauchy-specific basis of chapter 1915. The net read then passes through chapter 1667, which consumes the same displayed rows rather than a separate completion theorem.

Since the route never adds a coordinate for selected limits, quotient completion equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or detached convergence, those objects remain outside the exported Cauchy packet. □

Theorem 3688.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Every completion-facing $\mathsf{Cauchy}^{\uparrow}$ read first exposes the regular rational row of chapter 301, then passes the same finite-window and tolerance data to the $\mathsf{MetricCompletion}^{\uparrow}$ boundary of chapter 1506, and only afterward reaches the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Thus the visible dependency route is $\mathsf{Cauchy}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow} \to \mathsf{Real}^{\uparrow}$, not a self-contained hub read, quotient stream, selected limit, or ambient completeness principle.

Theorem 3688.14 (Cauchy regular-fusion forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that uses a fused regular-Cauchy tail passes through the sibling* $\mathsf{RegularCauchyFusion}^{\uparrow}$ *packet before the Cauchy hub exports the route. The displayed route consumes the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* BHist *evidence,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any completion consumer may inspect the fused tail.*

Proof. Read the fused tail as a sibling packet rather than as a hidden coordinate of the Cauchy hub. The $\mathsf{RegularCauchyFusion}^{\uparrow}$ row supplies the displayed fusion boundary, while $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ expose the finite observation windows, regular-rational readback, and dyadic tolerance rows used by the tail.

The completion-facing consumer then reaches the $\mathsf{Real}^{\uparrow}$ seal only after the finite route has been replayed through BHist, hsame, Cont, Pkg, and NameCert. Thus the Cauchy hub exports a sibling dependency route for fused regular-Cauchy tails, not a quotient stream, selected limit, choice schedule, trichotomy oracle, or ambient completeness row. □

The $\mathsf{Cauchy}^{\uparrow}$ carrier reads regular rational tails through chapter 301 and finite observation schedules through chapter 126. The tolerance rows supplied by chapter 291 and the terminal sealing of chapter 38 remain downstream of that lattice, so the hub exposes the RegSeqRat and StreamName dependencies before completion-facing consumers inspect the Real boundary.

Theorem 3688.15 (Cauchy RegSeqRat and Real sibling lattice repair)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *route used by this hub depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling for regular rational windows and on the* $\mathsf{Real}^{\uparrow}$ *sibling for the terminal seal boundary. Thus every completion-facing consumer first reads the regular rational dependency of chapter 301 and then the Real seal of chapter 38; the hub exports no quotient stream, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row of its own.*

Proof. Use the displayed RegSeqRat dependency files already routed through the hub. They force the Cauchy read through regular rational windows before any completion-facing row can be interpreted.

Then apply the terminal Real sibling route named by chapter 38. The seal boundary is therefore read as a sibling obligation after RegSeqRat evidence, not as hidden content of the top-level hub. □

The $\mathsf{Cauchy}^{\uparrow}$ hub also exposes the forward lattice used by regular-real consumers: the displayed Cauchy source is first read as $\mathsf{RegSeqRat}^{\uparrow}$ data and then handed to the $\mathsf{RegularCauchyReal}^{\uparrow}$ packet before any terminal real-facing consumer treats it as a completed object.

Theorem 3688.16 (Cauchy RegSeqRat to RegularCauchyReal sibling route)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *route consumed by a regular-real boundary must pass through the regular rational readback of chapter 301 before it reaches the* $\mathsf{RegularCauchyReal}^{\uparrow}$ *carrier of chapter 3546. The route exports a displayed L10 handoff through finite windows, regular readback, transport, replay, provenance, and local naming; it exports no detached Cauchy representative, quotient stream equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the hub through its existing regular-source children. Those children expose the Cauchy data as finite observation windows and regular rational rows before any real-facing seal can be consumed.

The $\mathsf{RegSeqRat}^{\uparrow}$ row is then the source read by the $\mathsf{RegularCauchyReal}^{\uparrow}$ carrier. Its sibling chapter adds the StreamName, dyadic tolerance, convergence handoff, Real seal, transport, replay, provenance, and local naming rows around the same regular readback.

Since neither route contains a detached representative, quotient stream equality, selected limit, countable-choice schedule, or ambient completeness coordinate, the handoff is exactly the displayed finite L10 lattice. □

The public $\mathsf{Cauchy}^{\uparrow}$ route consumed by completion and real-readback arguments is routed through the same finite lattice: $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, $\mathsf{RegSeqRat}^{\uparrow}$ rational

readback from chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the net-facing completion surface of chapter 3794, and the filter-facing completion surface of chapter 1569. The hub exports this dependency route only as finite BHist, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ data; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Theorem 3688.17 (Cauchy RegSeqRat Real dependency top link)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *consumer must expose the regular-source row of chapter 301, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before it can read a completion endpoint. The route remains inside componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

Proof. Read the top-level $\mathsf{Cauchy}^{\uparrow}$ hub through its displayed children. The hub exports only ordered sibling routes, so a completion-facing consumer first encounters finite observation windows and dyadic tolerance data rather than a hidden completion endpoint.

The regular-source handoff is then forced through the $\mathsf{RegSeqRat}^{\uparrow}$ row named by chapter 301. Only after that handoff is visible can the consumer read the terminal $\mathsf{Real}^{\uparrow}$ seal named by chapter 38.

The structural rows preserve the same path. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row keep the completion read inside the finite packet, with no quotient stream, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 3688.18 (Cauchy forward RegSeqRat and Real route)**.** *Every completion-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *packet must pass through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301 and then the terminal* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38 before any completion consumer may use it. The route carries only the finite stream-window, dyadic tolerance, regular-rational readback, real-seal,* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows exposed by the Cauchy sibling files; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with the carrier of Definition 3413.1. Its public readback is organized by finite stream windows, dyadic tolerances, regular-rational readback, terminal real-name sealing, componentwise transport, replay, provenance, and local naming.

Apply Definition 3688.8 and Definition 3413.4. These routes place every Real-facing or completion-facing Cauchy read behind displayed finite windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row before any terminal seal is consumed.

Finally apply Definition 3413.9 together with Definition 5888.13. The Real seal is reached only after the regular-rational row has been displayed, and the completion-facing consumer receives the same finite route. Since the cited sibling routes provide no coordinate for quotient equality, selected limits, choice, host equality, trichotomy, or ambient completeness, none can enter the forward route. □

Theorem 3688.19 (Cauchy StreamName–RegSeqRat forward route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that is later consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *must pass through* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* BHist*-indexed observations, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. The route is the displayed L10 handoff through chapter 126 and chapter 301; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.*

Proof. Begin with the displayed RegSeqRat route of Definition 3688.8. It already places every Real- or Completion-facing Cauchy read behind finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the terminal seal can be consumed.

Next use the dependency lattice of the Cauchy hub. The lattice records the sibling order in which the $\mathsf{RegSeqRat}^{\uparrow}$ tail rows and $\mathsf{StreamName}^{\uparrow}$ observation schedules are read, so the consumer cannot bypass the finite windows by appealing to a host Cauchy sequence.

Finally apply the regular-fusion route of Definition 3688.14. Its fused-tail handoff preserves the same BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows, so the Cauchy hub reaches any later $\mathsf{RegSeqRat}^{\uparrow}$ consumer only through the finite StreamName and regular rational surface. □

The $\mathsf{Cauchy}^{\uparrow}$ hub routes completion-facing consumers through the regular rational source of chapter 301, the complete-metric boundary of chapter 128, and the Cauchy-completion functor surface of chapter 423. Thus a completion read remains a displayed finite dependency route before any regular-name, metric-completion, or functorial completion sibling is consumed.

Chapter 3689

A Concrete Naming Certificate for $\mathsf{CauchyTailDominance}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailDominance}^{\uparrow}$ packet records the finite tail-dominance surface for Cauchy names at a displayed tolerance. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the index cofinality discipline of chapter 1594. The packet gives completion-facing Real arguments a named route from tolerance requests to tail windows; it does not select a limit, choose a subsequence, identify quotient real names, or import ambient completeness.

Definition 3689.1 (Cauchy tail-dominance carrier)**.** *A* $\mathsf{CauchyTailDominance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, I, W, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected Cauchy-name segment.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from that window.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the requested radius, slack budget, and comparison cell used for the tail test.*
4. *I is the* $\mathsf{CofinalSequence}^{\uparrow}$ *index row witnessing that the displayed tail index dominates the requested tolerance window.*
5. *W is the tail-window ledger, pairing the tolerance row D with the cofinal index row I and the inspected stream segment S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D, I, W have been displayed.*
7. *H records componentwise* hsame *transport for the stream window, regular readback, tolerance row, cofinal index row, tail ledger, Real seal, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a tolerance request through $D, I, S, R, W, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailDominance}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a chosen limit, quotient stream equality, selected subsequence, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3689.2 (Cauchy tail-dominance NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *carrier*

$$T = (S, R, D, I, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite stream window* S*, regular rational readback* R*, dyadic tolerance row* D*, cofinal index row* I*, tail-window ledger* W*, terminal Real seal* E*, componentwise transport* H*, displayed continuation replay* C*, package provenance* P*, and local naming row* N*.*

Proof. Project the carrier of Definition 3689.1. The projection exposes exactly $S, R, D, I, W, E, H, C, P, N$.

The tolerance request is first read through D. The cofinal row I then supplies the displayed index dominance needed to open the tail window, while S and R supply only the finite stream segment and regular rational readback attached to that request.

The ledger W binds D, I, S, R before the terminal seal E is visible. Thus the Real-facing consumer receives a tolerance-directed tail route rather than a chosen limit or a quotient equality of completed names.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no row for selected subsequences, countable choice, trichotomy, quotient equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3689.3 (Cauchy tail-dominance RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{CofinalSequence}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The consumer therefore reads tail dominance only as a displayed tolerance request, cofinal index witness, finite stream window, regular rational readback, and terminal Real seal. It receives no chosen Cauchy limit, selected subsequence, quotient stream equality, countable choice, trichotomy, or ambient completeness row.

Proof. Start with an admitted consumer read. By Definition 3689.1, the only analytic rows available before the Real seal are the tolerance row D, cofinal index row I, stream window S, regular readback R, and tail-window ledger W.

Read the requested precision through D. The row I then certifies the displayed dominance of the chosen tail index over that request, and W records that the same request controls the window opened in S.

The regular rational handoff is exactly R. It is downstream of D, I, S, W and upstream of the terminal seal E, so the Real-facing boundary cannot bypass the finite tail route.

The structural rows H, C, P, N replay and name this route without adding analytic coordinates. Hence every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read factors through the displayed tail-dominance packet and exports none of the excluded limit, subsequence, quotient, choice, trichotomy, or completeness data. □

Theorem 3689.4 (Cauchy tail-dominance Real seal consumption)**.** *Every* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer of an accepted* $\mathsf{CauchyTailDominance}^{\uparrow}$ *packet must read the tolerance row, cofinal index*

witness, $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and tail-window ledger before the Real seal is visible. Thus the completion boundary consumes the finite tail route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CofinalSequence}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ W \longrightarrow \mathsf{Real}^{\uparrow}$$

and cannot introduce selected limits, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 3689.1. The terminal Real row E is downstream of the tolerance row D, cofinal index row I, stream window S, regular rational readback R, and tail-window ledger W.

Apply Definition 3689.2. It makes those rows part of the local certificate surface, so a completion-facing consumer cannot read E before it has exposed the same displayed tolerance request, dominance witness, finite window, readback, and ledger.

Now use the handoff of Definition 3689.3. It identifies the route from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{CofinalSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ to the terminal Real seal.

The structural rows H, C, P, N preserve and name that route without adding analytic coordinates. Since the accepted carrier has no selected-limit row, no quotient stream equality row, no countable-choice row, no trichotomy row, and no ambient completeness row, the Real seal can only be consumed through the displayed finite tail boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3689.1, Definition 3689.2, Definition 3689.3, and Definition 3689.4 seed the finite tolerance-directed Cauchy tail-dominance route through StreamName, RegSeqRat, DyadicRatCore, CofinalSequence, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailDominance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, selected subsequences, quotient stream equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-row exactness, cofinal-index dominance, tail-window confinement, RegSeqRat handoff, and RealUp seal non-escape for BEDC.Derived.CauchyTailDominanceUp.

Constructive story: A $\mathsf{CauchyTailDominance}^{\uparrow}$ packet is generated from a StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance row, CofinalSequence index row, tail-window ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{Cauchy}^{\uparrow}$ completion-facing tail arguments consume chapter 3689 as the finite route that opens the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{CofinalSequence}^{\uparrow}$ index witness, and terminal $\mathsf{Real}^{\uparrow}$ seal before any Cauchy boundary is visible. This sibling route records that the Cauchy lattice reaches tail dominance through displayed L10 rows rather than through selected limits, quotient stream equality, countable choice, trichotomy, or ambient completeness.

Theorem 3689.5 (Cauchy metastability-filter sibling route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing read that uses finite tail control factors through* $\mathsf{CauchyMetastability}^{\uparrow}$ *and* $\mathsf{CauchyFilter}^{\uparrow}$ *before reaching* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. Concretely, the consumer first reads the finite-tail stability packet of chapter 2093 and the finite-filterbase carrier of chapter 333; only then may it inspect stream windows, regular rational readback, dyadic tolerance ledgers, terminal real seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows.*

Proof. Start with the Cauchy hub dependency surface. Its completion-facing children already route through finite stream windows and regular rational readback before the terminal real seal is visible.

For finite tail control, use chapter 2093. The metastability packet exposes the precision request, control row, finite tail-window answer, dyadic ledger, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local name.

For filter-facing Cauchy data, use chapter 333. The CauchyFilter packet exposes the finite filterbase carrier, directed refinement row, Cauchy modulus row, stream schedules, regular observations, dyadic neighbourhood witnesses, transport, replay, provenance, and local name.

Combining the two sibling packets gives the stated route. No completion-facing $\mathsf{Cauchy}^{\uparrow}$ read can reach the StreamName, RegSeqRat, DyadicRatCore, or RealUp rows by bypassing both finite-tail stability and finite-filterbase data. □

Theorem 3689.6 (Cauchy top-level sibling route)**.** *The top-level* $\mathsf{Cauchy}^{\uparrow}$ *hub routes ordinary Cauchy evidence,* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *sequence-surface data, completion algebra, and tail-dominance readback through its sibling chapters before any Real or Completion consumer reads a terminal seal. The route consumes the finite observation surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Bishop regular completion consumer of chapter 1639. It thereby keeps Cauchy, sequence-space, completion-algebra, and tail-dominance surfaces on sibling NameCert routes rather than on a hidden hub coordinate.*

Proof. Read the hub as an ordered sibling route. Its ordinary Cauchy and tail-dominance files supply local Cauchy evidence, while the displayed RegSeqRat and metric-completion dependency files point outward to the Real and Completion lattice.

The finite-window side is supplied by chapter 126 and chapter 291. Those siblings provide observation windows and tolerance rows before a regular rational or real-seal consumer can inspect the Cauchy surface.

The completion-facing side then passes through chapter 301, chapter 38, and chapter 1639. Thus the Cauchy hub exports only a horizontal dependency route among displayed sibling chapters, with no quotient stream, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness row. □

Theorem 3689.7 (Cauchy Real-completion forward route)**.** *Every Real-completion consumer of the top-level* $\mathsf{Cauchy}^{\uparrow}$ *hub first passes through the sibling route displayed in Definition 3689.6: finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the Bishop regular completion handoff. No hub coordinate supplies quotient streams, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness for such a consumer.*

Proof. Apply Definition 3689.6 to read the hub as an ordered dependency route rather than as a terminal completion object. Its first two displayed siblings provide the finite-window and dyadic-tolerance data that a completion-facing consumer must expose before any rational or real readback can be inspected.

The remaining siblings then force the read through RegSeqRat, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the Bishop regular completion handoff. Since the route already accounts for every completion-facing coordinate named by the hub, there is no remaining coordinate from which a quotient stream, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle could be exported. □

Theorem 3689.8 (Cauchy top-level completion lattice route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of the top-level hub factors through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can use any completion*

boundary. The route is read through chapter 126, chapter 291, chapter 301, chapter 38, and the Bishop regular completion surface of chapter 1639; it exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a completion-facing read of the hub. The hub is only an ordered list of child routes, so the consumer first meets the finite StreamName window rows and the DyadicRatCore tolerance rows named by the displayed siblings, not a hidden completion object.

The regular rational readback is then routed through $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Structural replay through hsame, Cont, Pkg, and NameCert preserves the same ordered rows rather than adding a quotient stream or selected endpoint.

Finally the Bishop regular completion surface may consume the already displayed route. Since every completion-facing coordinate has passed through the sibling lattice, no remaining hub coordinate can supply choice, host equality, or an ambient completeness theorem. □

Chapter 3690

A Concrete Naming Certificate for MonotoneCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the MonotoneCauchySubsequence$^{\uparrow}$ packet records the finite surface by which a monotone source sequence exposes a Cauchy subsequence route without exporting a selected limit or quotient stream. It sits after the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal Real$^{\uparrow}$ seal of chapter 38, the monotone bounded source packet of chapter 3407, and the subsequence-stability handoff of chapter 3515. The packet names the extraction surface available to Real and Completion consumers; it has no coordinate for a quotient stream, selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 3690.1 (MonotoneCauchySubsequence carrier)**.** *A* MonotoneCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, I, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source monotone sequence row, read through the* MonotoneBoundedSequence$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *surfaces before any subsequence request is consumed.*
2. *I is the monotone index row. It records the finite increasing index schedule used to inspect subsequence windows.*
3. *M is the modulus and tail row. It records the finite tail threshold requested by a Cauchy read and the matching source windows opened by I.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *budget row carrying the dyadic tolerance and refinement data for the source and subsequence windows.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the selected windows after the index and dyadic budget rows have been displayed.*
6. *E is the* Real$^{\uparrow}$ *seal row reached only after S, I, M, D, R have been read.*
7. *H records componentwise* hsame *transport for the source sequence, index schedule, modulus and tail row, dyadic budget, readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a subsequence Cauchy request through $S, I, M, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* MonotoneCauchySubsequence$^{\uparrow}$*,* MonotoneBoundedSequence$^{\uparrow}$*,* RegularCauchySubseq StreamName$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$ *row.*
The carrier exports only the finite subsequence extraction packet. It exports no quotient stream, selected limit, countable choice, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchySubsequenceUp.MonotoneCauchySubsequenceTasteGate single carrier alignment`

Definition 3690.2 (MonotoneCauchySubsequence classifier)**.** *Two accepted* $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ *carriers*

$$Q = (S, I, M, D, R, E, H, C, P, N)$$

and

$$Q' = (S', I', M', D', R', E', H', C', P', N')$$

are classified together exactly when the displayed hsame *rows transport S to S', I to I', M to M', D to D', R to R', and E to E', and the displayed* Cont *rows replay the same route from the monotone source through index schedule, tail modulus, dyadic budget, regular readback, and Real seal. The classifier has no coordinate for host sequence equality, quotient stream equality, a selected infinite subsequence, a selected limit, countable choice, trichotomy, or ambient completeness.*

Theorem 3690.3 (MonotoneCauchySubsequence NameCert obligations)**.** *For every accepted* MonotoneCauchySub *carrier*

$$Q = (S, I, M, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the source monotone sequence row, monotone index row, modulus and tail row, dyadic budget row, regular rational readback, Real seal, componentwise transport, replay, provenance, and local naming. Every subsequence Cauchy read must display S, I, M, D, R before E is available, and H, C, P, N only preserve that route.*

Proof. Project the carrier of Definition 3690.1. The displayed analytic rows are S, I, M, D, R, E, and the structural rows are H, C, P, N.

The source row S is read through the monotone bounded sequence surface, while I records the increasing index schedule. A Cauchy request can therefore inspect a subsequence window only after the source schedule and the selected index row have both been displayed.

The modulus and budget rows come next. The row M supplies the finite tail threshold, and D supplies the dyadic tolerance and refinement budget for the same source and subsequence windows.

The readback and seal rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the scheduled windows after the index and dyadic budget have been displayed, and the $\mathsf{Real}^{\uparrow}$ row E seals only that finite route. Transport, replay, provenance, and local naming preserve the same coordinates, so the listed rows exhaust the local NameCert obligations. □

Theorem 3690.4 (MonotoneCauchySubsequence RegSeqRat handoff)**.** *Every accepted* MonotoneCauchySubsequen *carrier $Q = (S, I, M, D, R, E, H, C, P, N)$ hands a subsequence Cauchy read to* $\mathsf{RegSeqRat}^{\uparrow}$ *only through the finite route*

$$S, I \longmapsto M, D \longmapsto R \longmapsto E, H, C, P, N.$$

The handoff composes the source monotone bounded sequence route with the stability surface of chapter 3515; it does not export a quotient stream, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier and classifier. They force every subsequence read to preserve the source sequence row, monotone index schedule, modulus and tail row, dyadic budget, readback, and seal by componentwise transport and displayed replay.

Apply Definition 3690.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ row R after S, I, M, D, so the regular rational readback cannot be detached from the selected source windows or the dyadic tolerance ledger.

Finally use the sibling stability route of chapter 3515. That route preserves regular Cauchy readback under monotone cofinal subsequence schedules. The present packet supplies the corresponding $S, I, M, D, R, E, H, C, P, N$ rows, and no excluded quotient, selector, choice, trichotomy, or completeness coordinate is available. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ must display the monotone source row, increasing index row, modulus and tail row, dyadic budget, regular readback, Real seal, transport, replay, provenance, and local NameCert row first.

Independence witness: monotoneboundedsequence, regularcauchysubsequencestability, regseqrat, streamname, dyadicratcore, real: $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source monotone sequence, increasing index schedule, modulus and tail row, dyadic budget, readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3690.1, Definition 3690.2, Definition 3690.3, and Definition 3690.4 seed the finite extraction surface for monotone Cauchy subsequence reads over source sequence, index, modulus, dyadic budget, readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, classifier preservation, monotone index exactness, dyadic budget transport, RegSeqRat handoff, Real seal nonescape, provenance, and local naming for BEDC.Derived.MonotoneCauchySubseque

Constructive story: A $\mathsf{MonotoneCauchySubsequence}^{\uparrow}$ packet is generated from a monotone source sequence row, increasing index row, modulus and tail row, dyadic budget, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3691

A Concrete Naming Certificate for $\mathsf{CauchySequenceCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchySequenceCompletion}^{\uparrow}$ packet records Cauchy completion data by finite regular-sequence rows rather than by ambient quotient language. It sits after the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the completion handoff of chapter 1506 and chapter 128, the terminal real-name seal of chapter 38, and the separated uniqueness boundary of chapter 1411. The packet gives Real and Completion consumers one displayed route from finite Cauchy sequence observations to a named completion point.

Definition 3691.1 (CauchySequenceCompletion carrier)**.** *A* $\mathsf{CauchySequenceCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, S, R, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric or regular-Cauchy source row whose distance observations are the only metric data available to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected indices and the request schedule used by the Cauchy read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row on exactly those windows, including the rational approximants and their displayed tail budgets.*
4. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row exposing the completion-facing carrier and entry discipline.*
5. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-service row reached only after S, R, M have been displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the completed readback, paired with the* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness route when two completed names are compared.*
7. *H records componentwise* hsame *transport for the source, window, readback, completion handoff, limit-service, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through $X, S, R, M, L, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, S, R, M, L, E, H, C, P, N$. It has no coordinate for quotient completion classes, selected representatives, countable choice, host equality of sequences, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCompletionUp.CauchySequenceCompletionTasteGate single carrier alignment`

Theorem 3691.2 (CauchySequenceCompletion NameCert obligations)**.** *For every accepted* CauchySequenceCompl*carrier $K = (X, S, R, M, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the source metric row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff,* $\mathsf{CompleteMetric}^{\uparrow}$ *limit row,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{SeparatedMetric}^{\uparrow}$ *uniqueness route, componentwise transport, displayed replay, provenance, and local naming.*

Every completion-facing consumer must read these rows in the order

$$X, S \longmapsto R \longmapsto M, L \longmapsto E \longmapsto H, C, P, N.$$

No obligation is discharged by a quotient-completion equality, a selected limit, countable choice, host sequence equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3691.1. The only source observations are the metric row X, the finite windows S, and the regular rational readback R.

The completion data is downstream of those observations. The row M exposes the $\mathsf{MetricCompletion}^{\uparrow}$ handoff, while L records the $\mathsf{CompleteMetric}^{\uparrow}$ limit service and cannot be read before the displayed windows and readback have been accepted.

The terminal row E is the Real-facing seal. When two completed names are compared, the separated uniqueness boundary is reached through the same displayed rows before any classifier identification is exported.

Finally H, C, P, N transport, replay, source, and name the route. Since the carrier contains no quotient, selected representative, countable-choice schedule, host equality, trichotomy, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3691.3 (CauchySequenceCompletion finite-window route)**.** *Every public read of an accepted* $\mathsf{CauchySequenceCompletion}^{\uparrow}$ *packet that reaches* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback used by the Cauchy sequence. If a uniqueness consumer compares two completed reads, it must then pass through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *zero-distance route before reading equality of the exposed completion names.*

Thus the route exports completion data only as a finite window-and-readback handoff:

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow}.$$

It exports no quotient completion class, selected representative, unrestricted limit selector, host sequence equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a completion-facing request. By Definition 3691.1, the request must open S and R before it can reach either completion row.

The $\mathsf{MetricCompletion}^{\uparrow}$ handoff M and $\mathsf{CompleteMetric}^{\uparrow}$ row L then carry the displayed Cauchy read into the completion surface. The Real seal E is downstream of the same handoff, so a Real-facing read cannot bypass finite windows or regular rational readback.

For uniqueness reads, apply the separated boundary recorded in the carrier. The comparison passes through the zero-distance route of $\mathsf{SeparatedMetric}^{\uparrow}$ before any completed-name identification is public.

The structural rows H, C, P, N preserve this dependency order under transport and replay. No excluded completion, choice, host-equality, trichotomy, or Real-completeness principle is present as a carrier row, so none can be exported by the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3691.1, Definition 3691.2, and Definition 3691.3 seed the finite Cauchy sequence completion route over StreamName windows, RegSeqRat readback, MetricCompletion and CompleteMetric handoffs, Real seals, separated uniqueness, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host sequence equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window readback, completion handoff, limit-service exposure, Real seal nonescape, separated uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCompletionUp.

Constructive story: A $\mathsf{CauchySequenceCompletion}^{\uparrow}$ packet is generated from a source metric row, finite StreamName windows, RegSeqRat readback, MetricCompletion handoff, CompleteMetric limit service, Real seal, SeparatedMetric uniqueness route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3692

A Concrete Naming Certificate for $\mathsf{BoundedIntervalProjection}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{BoundedIntervalProjection}^{\uparrow}$ packet records the finite data needed to project a displayed real-name input into a displayed closed bounded interval. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream window surface of chapter 126, the dyadic tolerance core of chapter 291, the endpoint interval packet of chapter 3527, the clamp route of Definition 1810.4, and the located interval surface of chapter 1709. The packet gives bounded-subsequence, interval-trapping, and completion-facing consumers a named projection object; it is not host trichotomy, not quotient equality, not a selected limit, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3692.1 (Bounded interval projection carrier)**.** *A* $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Real}^{\uparrow}$ *row whose projected value is being named.*
2. *L and U are the lower and upper endpoint* $\mathsf{Real}^{\uparrow}$ *rows of the closed bounded interval.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-order and tolerance ledger used to compare finite reads of X, L, U.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window surface for the source and endpoint names.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
6. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row tying the endpoint seals, dyadic order ledger, and locatedness data to the interval surface.*
7. *C is the* $\mathsf{RealClamping}^{\uparrow}$ *row naming the constructive clamp of X against the displayed interval.*
8. *H stores componentwise* hsame *transport for the source, endpoints, tolerance ledger, windows, readback, interval, clamp, replay, provenance, and naming rows.*
9. *T stores displayed* Cont *replay of the projection route through $X, L, U, D, W, R, I, C, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedIntervalProjection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, L, U, D, W, R, I, C, H, T, P, N$. It has no coordinate for host trichotomy, quotient equality of real names, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3692.2 (Bounded interval projection NameCert obligations)**.** *For every accepted* BoundedIntervalProjecti *carrier*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *row X, endpoint rows L, U, dyadic endpoint-order and tolerance ledger D, finite stream windows W, regular rational readback R, located interval row I, constructive clamping row C, componentwise transport H, displayed replay T, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3692.1. The projection exposes exactly the source real row, endpoint rows, dyadic tolerance ledger, finite windows, regular readback, located interval row, clamp row, and structural rows.

The interval boundary is read through the $\mathsf{RealBoundedInterval}^{\uparrow}$ and $\mathsf{LocatedRealInterval}^{\uparrow}$ surfaces. Thus endpoint order, locatedness, and finite bracket information are visible before any projection output is named.

The projection value is read through C. By the $\mathsf{RealClamping}^{\uparrow}$ route cited above, the clamp consumes the same source, endpoints, dyadic tolerance ledger, stream windows, regular readback, and located interval row, so it introduces no host comparison oracle or hidden limit selector.

Finally H, T, P, N transport, replay, source, and name that same route. Since the carrier has no coordinate for host trichotomy, quotient equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.BoundedIntervalProjectionUp.BoundedIntervalProjectionTasteGate single carrier alignment`

Theorem 3692.3 (Bounded interval projection clamping stability)**.** *Let*

$$B = (X, L, U, D, W, R, I, C, H, T, P, N)$$

be an accepted $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *carrier. Any componentwise* hsame *transport of the source row, endpoint rows, dyadic ledger, stream windows, regular readback, and located interval row transports the clamp row C to the same* $\mathsf{BoundedIntervalProjection}^{\uparrow}$ *output surface. Consequently bounded-subsequence and interval-trapping consumers read a stable closed-interval projection through the displayed rows, not through quotient equality, selected limits, host trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the accepted carrier and its finite interval boundary. The rows L, U, D, W, R, I display the endpoint seals, dyadic tolerance comparisons, finite windows, regular rational readback, and located interval data used by the clamp.

Apply the clamp stability route of Definition 1810.4. It states that the clamped value is stable under the displayed finite Cauchy readback and transport rows, so the projection output is determined by the same rows already present in B.

The transport row H and replay row T preserve that route for the local packet. Provenance and naming through P, N then repack the result as a $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$ surface.

There is no further coordinate in Definition 3692.1 from which a host comparison, quotient real equality, selected limit, or ambient completeness principle could be read. Thus clamping stability is exactly stability of the displayed finite projection packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedIntervalProjection}^{\uparrow}$ packet is generated from a source RealUp row, two endpoint RealUp rows, a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a LocatedRealInterval row, a RealClamping row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3692.1, Definition 3692.2, and Definition 3692.3 seed the finite closed-interval projection route over $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealBoundedInterval}^{\uparrow}$, $\mathsf{RealClamping}^{\uparrow}$, and $\mathsf{LocatedRealInterval}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedIntervalProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host trichotomy, quotient equality of real names, selected limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order evidence, finite-window readback, located interval handoff, clamping stability, transport, replay, provenance, and local naming for BEDC.Derived.BoundedIntervalProjection

Chapter 3693

A Concrete Naming Certificate for $\mathsf{MontelTheorem}^{\uparrow}$

Following Montel's theorem in the normal-family tradition, the $\mathsf{MontelTheorem}^{\uparrow}$ packet records only the BEDC finite-window content of constructive normal-family compactness. It binds compact-window rows, local boundedness and equicontinuity ledgers, uniform Cauchy extraction, and holomorphic readback rows over displayed finite charts. The chapter does not claim classical open-cover compactness, countable choice, quotient real or complex equality, selected subsequences, or full analytic compactness. Its sibling route passes through the finite-modulus Arzela–Ascoli surface of chapter 412, the compact metric and complete metric rows already available in the concrete-instance family, the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, and the holomorphic and complex surfaces when those rows are present.

Definition 3693.1 (Montel theorem carrier)**.** *A* $\mathsf{MontelTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-window row, naming the finite subwindow, metric inspection data, and compact-source certificate used by the family read.*
2. *L is the local boundedness ledger for the displayed family on the same compact window.*
3. *E is the equicontinuity ledger, expressed as finite modulus rows over the compact-window inspection schedule.*
4. *U is the uniform Cauchy extraction row, recording a finite tail window, a subsequence or subnet schedule when already displayed by the source packet, and a dyadic tolerance budget.*
5. *Q is the uniform-limit readback row over the same compact window.*
6. *G is the holomorphic or complex-read row, restricted to the displayed chart and consumed only after K, L, E, U, Q have been exposed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback family for the finite observation windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the local boundedness, equicontinuity, and uniform Cauchy rows.*
9. *Z is the terminal complex or real seal row read by the holomorphic surface when such a row is part of the source family.*

10. *H records componentwise* hsame *transport for all displayed rows.*
11. *C records displayed* Cont *replay from compact-window request through local boundedness, equicontinuity, uniform Cauchy extraction, uniform-limit readback, and holomorphic readback.*
12. *P is the* Pkg *provenance over* $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, *compact-window and complete-metric rows,* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *complex or holomorphic surfaces,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their structural transport, replay, provenance, and local naming data.

Lean4 (definition) `BEDC.Derived.MontelTheoremUp.TasteGate.MontelTheoremUp`
Lean4 ✓ `BEDC.Derived.MontelTheoremUp.TasteGate.MontelTheoremTasteGate single carrier alignment`

Theorem 3693.2 (Montel theorem NameCert obligations)**.** *For every accepted* $\mathsf{MontelTheorem}^{\uparrow}$ *carrier* $M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *compact-window admission for K and agreement of every later row with the same displayed window;*
2. *local boundedness admission for L over the compact-window schedule;*
3. *equicontinuity admission for E with dyadic tolerances read from D;*
4. *uniform Cauchy extraction through U, with every late-tail comparison served by R and D;*
5. *uniform-limit readback through Q before any holomorphic or complex read in G is exposed;*
6. *holomorphic readback compatibility, requiring G and Z to consume only the displayed uniform Cauchy and uniform-limit rows;*
7. *transport, replay, provenance, and naming stability through H, C, P, N.*

No obligation introduces open-cover compactness, countable choice, quotient real equality, quotient complex equality, a selected analytic limit, or ambient analytic compactness.

Proof. Project the carrier in Definition 3693.1. The compact-window row K is the only source of domain inspection data, so every later read is scoped to the same finite compact window.

Local boundedness and equicontinuity are the finite normal-family inputs. The rows L and E expose pointwise bound and finite-modulus data over K, while D supplies the dyadic tolerance budget used by both ledgers.

The extraction step is the finite Arzela–Ascoli route. Apply the sibling surface chapter 412 through the displayed rows K, L, E, U, R, D to obtain a uniform Cauchy extraction over the compact window rather than a choice-selected analytic subsequence.

The readback step consumes Q and then G. The uniform-limit row is displayed before the holomorphic or complex-read row, so holomorphic readback can use only the finite uniform Cauchy and tolerance surface already named by the carrier.

Finally, H, C, P, N preserve the same row inventory under transport, continuation, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, countable choice, quotient equality, selected analytic limits, or ambient analytic compactness, the NameCert surface cannot export any such datum. □

Theorem 3693.3 (Montel theorem finite normal-family criterion)**.** *Let $M = (K, L, E, U, Q, G, R, D, Z, H, C, P, N)$ be an accepted* $\mathsf{MontelTheorem}^{\uparrow}$ *packet. Any consumer claiming a normal-family compactness read from M factors through the finite route*

$$\textit{compact window} \longmapsto \textit{local boundedness and equicontinuity} \longmapsto \textit{uniform Cauchy extraction} \longmapsto \textit{uniform-limit readl}$$

Thus the consumer receives compact-window, boundedness, equicontinuity, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *uniform Cauchy, uniform-limit, and holomorphic rows as finite BEDC data. It receives no classical open-cover compactness theorem, countable-choice selector, quotient real or complex equality, selected analytic limit, or full analytic compactness assertion.*

Proof. Start from the accepted carrier. The only admissible source route begins with K, so every compactness-facing consumer first reads the displayed compact-window packet.

Next consume L and E. These rows give local boundedness and equicontinuity over that window, with the tolerance budget synchronized by D and the regular readbacks synchronized by R.

Now replay C through U and Q. The route supplies uniform Cauchy extraction and uniform-limit readback on the same finite window, matching the ContinuousMap and UniformLimit style chapters without selecting an external analytic limit.

Finally consume G and Z only after the uniform-limit row is present. Componentwise transport through H and provenance through P, N preserve the same route, so no forbidden open-cover, choice, quotient-equality, selected-limit, or ambient-compactness coordinate can enter the public read. □

Closure status for $\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3693.1, Definition 3693.2, and Definition 3693.3 seed the finite normal-family route through compact windows, local boundedness, equicontinuity, uniform Cauchy extraction, uniform-limit readback, and holomorphic readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MontelTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical open-cover compactness, countable choice, quotient real equality, quotient complex equality, selected analytic limits, full analytic compactness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exactness, local-boundedness and equicontinuity ledgers, uniform Cauchy extraction, holomorphic readback compatibility, and provenance exhaustion for BEDC.Derived.MontelTheoremUp.

Constructive story: A $\mathsf{MontelTheorem}^{\uparrow}$ packet is generated from compact-window rows, local boundedness and equicontinuity ledgers, uniform Cauchy extraction, uniform-limit readback, holomorphic or complex readback, RegSeqRat and DyadicRatCore tolerance rows, transport, replay, provenance, and local NameCert rows.

Chapter 3694

A Concrete Naming Certificate for $\mathsf{AntipodalTriggerLedger}^{\uparrow}$

The $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ packet realizes the antipodal AGY trigger of Definition 9462.1506 as finite NameCert data. It records the median-closure source, the R-halfspace with its CUY anchor, the AGY antipode row, the trigger ledger, transport, replay, provenance, and local naming surface. The packet is adjacent to the six-boundary and trigger-algebra family of chapter 3470, chapter 3478, chapter 3651, and chapter 3681, but it is not bijective to any of them because it binds the one-pass antipodal median trigger as its own carrier row.

Definition 3694.1 (Antipodal trigger ledger carrier)**.** *An* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, Y, G, T, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row for $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$, including the saturated R halfspace and the* CUY *anchor row.*
2. *Y is the third-position Y halfspace shell ledger, displaying the shell decomposition H_0, H_1, H_2, H_3, H_4.*
3. *G is the antipodal* AGY *row, displaying either* AGU *or* AGC *as the single trigger.*
4. *T is the trigger ledger that pairs the* CUY *anchor, the* AGY *antipode, and the already present R-halfspace witness t_b for each target Y-side codon b.*
5. *M is the median-closure replay row, recording that the coordinatewise median of those three displayed witnesses reaches b and hence fills the whole codon cube in one pass.*
6. *H records componentwise* hsame *transport for the source, shell, antipode, trigger, median-closure, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from source presentation through the antipode trigger ledger into the median-closure row.*
8. *P is the* Pkg *provenance over* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{AnchorSubcubeTrigger}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^{\uparrow}}$ *row.*

The classifier admits only packets carrying these finite rows and future implementations must provide the corresponding Nontrivial *and* FieldFaithful *instances before a Lean-side field interpretation can be claimed.*

Lean4 (definition) `BEDC.Derived.AntipodalTriggerLedgerUp.AntipodalTriggerLedgerUp`

Falsifiable prediction: Removing the displayed AGY antipode row prevents an accepted $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ packet from claiming one-pass full-cube median closure.

Independence witness: `six_boundary_tile_coverage`, `window_coverage_trigger_algebra`, `anchor_subcube_trigger`, and `radial_wobble_spectrum`: none is bijective to $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ because the packet jointly stores the $M^{\star}$ source, the CUY anchor, the AGY antipode, the one-pass target witness ledger, median replay, transport, continuation, provenance, and local naming rows.

Theorem 3694.2 (Antipodal trigger ledger NameCert obligations)**.** *For every accepted* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *carrier* $A = (S, Y, G, T, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *source admission for* $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$ *and its median-closed halfspace anchor;*
2. *shell admission for the displayed* Y*-halfspace ledger and its antipodal* $H_4 = \mathrm{AGY}$ *row;*
3. *trigger exactness, requiring the same* CUY *anchor,* AGY *antipode, and* R*-halfspace witness row to feed the median calculation;*
4. *one-pass median replay, requiring the row* M *to expose every* Y*-side target before the full-cube claim is exported;*
5. *transport, replay, provenance, and naming stability through* H, C, P, N*;*
6. *field-read discipline, requiring future* Nontrivial *and* FieldFaithful *instances before any field-facing interpretation of the trigger ledger is exposed.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier in Definition 3694.1. The source row S supplies the R halfspace and the CUY anchor, while Y supplies the finite shell ledger.

The trigger row G is indispensable. Without a displayed AGY antipode, the packet contains no finite row pairing the anchor with the antipodal point required by Definition 9462.1506.

The ledger T then names the witness triple for each target Y-side codon b: the CUY anchor, the AGY antipode, and the already present R-halfspace point t_b. The replay row M records the coordinatewise median calculation and exposes the one-pass full-cube result.

Finally H, C, P, N preserve this route under transport, continuation, provenance, and local naming. The field-read clause is only a discipline on later interpretation: until Nontrivial and FieldFaithful instances are displayed, the packet exports a finite median-trigger ledger rather than a field theorem. □

Theorem 3694.3 (Antipodal trigger ledger median nonescape)**.** *Every accepted* $\mathsf{AntipodalTriggerLedger}^{\uparrow}$ *full-cube read factors through the finite route*

$$M^{\star} \longmapsto \mathrm{CUY}\ \textit{anchor} \longmapsto \mathrm{AGY}\ \textit{antipode} \longmapsto \textit{target witness ledger} \longmapsto \textit{one-pass median replay.}$$

Thus the one-pass median-closure claim is available only when $S, Y, G, T, M, H, C, P, N$ *are all displayed. A packet missing the* AGY *row, the target witness ledger, or the median replay row cannot export full-cube closure.*

Proof. Begin with an accepted carrier. The public full-cube read must first expose S, so the median source is exactly $M^\star$ with the CUY anchor and the already present R halfspace.

Next the read consumes Y and G. These rows place the AGY point in the antipodal shell H_4 and identify it as the single trigger rather than as an undifferentiated radial-shell element.

Now apply the trigger ledger T and median replay M. For each target Y-side codon, the route supplies the anchor, antipode, and matching R-halfspace witness, so the coordinatewise median calculation remains inside displayed finite data.

The structural rows H, C, P, N do not add another closure principle; they preserve the same finite route. Hence a consumer that omits the antipode, witness ledger, or median replay row cannot claim one-pass full-cube median closure from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 3694.1, Definition 3694.2, and Definition 3694.3 seed the finite median-trigger ledger from the $M^\star$ source through the AGY antipode into one-pass full-cube median replay.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AntipodalTriggerLedger}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, field-facing interpretation without Nontrivial and FieldFaithful instances, radial optimality, spectral optimality, or a public standard bridge.

Upgrade path: Obligation closure requires checked source admission, shell admission, antipode exactness, target-witness ledger replay, median nonescape, field-read discipline, and provenance exhaustion for BEDC.Derived.AntipodalTriggerLedgerUp.

Constructive story: An $\mathsf{AntipodalTriggerLedger}^\uparrow$ packet is generated from the $M^\star$ source row, Y-halfspace shell ledger, AGY antipode row, target witness ledger, one-pass median replay, transport, continuation, provenance, and local NameCert rows.

Chapter 3695

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ packet records the displayed criterion by which a Cauchy source, an effective precision schedule, and a finite tail-window readback become one completion-facing BEDC object. It sits after the criterion surface of chapter 392, the modulus schedule of chapter 292, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the effective modulus packet of chapter 3518, and the terminal real-name seal of chapter 38. The packet supplies a finite L10 criterion/modulus handoff; it does not construct quotient streams, select a host limit, import countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3695.1 (Effective Cauchy criterion carrier)**.** *An* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, M, S, T, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *source row naming the finite Cauchy request and its accepted comparison surface.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, refined through the* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *packet to a displayed precision-to-tail request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite source window opened by the criterion request.*
4. *T is the finite tail-window row selected by the effective modulus schedule for the same request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested radius, tail comparison, and endpoint budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after Q, M, S, T, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only to consumers that have already read the criterion, schedule, windows, tolerance ledger, and regular-rational readback.*
8. *H stores componentwise* hsame *transport for the criterion, modulus, windows, tolerance ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from an effective Cauchy request through $Q, M, S, T, D, R, L, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{EffectiveCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, S, T, D, R, L, H, C, P, N$. It has no coordinate for quotient-stream equality, a selected limit, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchyCriterionUp.TasteGate.EffectiveCauchyCriterionTasteGate single carrier alignment`

Theorem 3695.2 (Effective Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *carrier*

$$E = (Q, M, S, T, D, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the criterion row Q, effective modulus schedule M, source window S, tail window T, dyadic tolerance ledger D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Real}^{\uparrow}$ *seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3695.1. The analytic rows are exactly the criterion request Q, the effective modulus schedule M, the source and tail windows S, T, the dyadic tolerance ledger D, the regular-rational readback R, and the terminal real seal L.

The schedule obligation is read through M and T. The modulus row turns a displayed precision request into a finite tail threshold, and the tail-window row records the finite observations permitted at that threshold.

The readback obligation is downstream of Q, M, S, T, D. The regular-rational row R is therefore a finite readback from the same criterion and tolerance ledger, not a quotient stream, selected limit, or ambient completeness argument.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded objects, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3695.3 (Effective Cauchy criterion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *packet factors through*

$$Q, M \longmapsto S, T, D \longmapsto R \longmapsto L \longmapsto H, C, P, N.$$

Consequently the consumer receives the displayed criterion request, effective modulus schedule, finite source and tail windows, dyadic tolerance ledger, regular-rational readback, terminal real seal, and local structural rows; it receives no quotient-stream equality, selected host limit, countable choice, host sequence equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the criterion and modulus rows. The row Q states the Cauchy comparison request, while M supplies the effective precision-to-tail schedule consumed by that request.

Next open the finite windows and tolerance ledger. The source window S and tail window T provide the only observations exposed to the request, and D records the dyadic budget against which those observations are compared.

Only after those rows are displayed does the packet expose R and then L. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback produced from the finite windows and tolerance ledger, and L is the terminal $\mathsf{Real}^{\uparrow}$ seal available to completion-facing consumers.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Thus every consumer route factors through the displayed finite handoff and cannot bypass it through any excluded host object. □

Closure status for NameCert$_{EffectiveCauchyCriterionUp}$.

Constructive story: An EffectiveCauchyCriterion$^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus effective schedule, StreamName source and tail windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{EffectiveCauchyCriterionUp}$, seed)

Scope closed: Definition 3695.1, Definition 3695.2, and Definition 3695.3 seed the finite effective Cauchy criterion handoff for RegSeqRat and Real-completion consumers.

Formal verification: FormalStatus(NameCert$_{EffectiveCauchyCriterionUp}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient-stream equality, selected limits, host sequence equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective schedule exactness, StreamName source and tail window admission, DyadicRatCore tolerance readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.EffectiveCauchyCriterionUp.

Chapter 3696

A Concrete Naming Certificate for $\mathsf{UniformlyCauchySequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the packet records a uniformly Cauchy sequence as displayed finite tail schedules rather than host convergence. It sits beside the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the displayed modulus interface of chapter 1438, and the terminal seal of chapter 38. The packet supplies an L10 completion-facing finite schedule object; it is not a quotient stream, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3696.1 (Uniformly Cauchy sequence carrier)**.** *A* $\mathsf{UniformlyCauchySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row, naming the displayed stream whose finite observations are admitted by the packet.*
2. *T is the uniform tail-threshold schedule. For each requested dyadic tolerance it records the single tail index after which all later source windows are compared.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source at the threshold and at later comparison windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the precision budget and dyadic comparison rows consumed by the uniform tail schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same source, window, tolerance, and tail-threshold data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, T, W, D, R have been displayed.*
7. *H stores componentwise* hsame *transport for the source, threshold schedule, window row, tolerance row, readback row, Real seal, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a uniform Cauchy request through $S, T, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyCauchySequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, W, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, hidden cofinal subsequences, countable choice, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformlyCauchySequenceUp.UniformlyCauchySequenceTasteGate single carrier alignment`

Theorem 3696.2 (Uniformly Cauchy sequence NameCert obligations)**.** *For every accepted* UniformlyCauchySequen *carrier*

$$K = (S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the sequence source S, uniform tail-threshold schedule T,* $\mathsf{StreamName}^{\uparrow}$ *window row W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{Real}^{\uparrow}$ *seal row E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3696.1. The analytic rows are the source S, threshold schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback R, and Real seal E.

The uniformity obligation is read only from T, W, D. For a requested tolerance, T gives one displayed threshold and W supplies the inspected windows beyond it, while D records the dyadic comparison budget used by those windows.

The regular readback is downstream of the same schedule. Thus R is a $\mathsf{RegSeqRat}^{\uparrow}$ row obtained through S, T, W, D, not a host sequence, quotient stream, selected limit, hidden cofinal family, or completeness argument.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for countable choice, host equality, quotient streams, selected limits, hidden cofinal subsequences, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Theorem 3696.3 (Uniformly Cauchy sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{UniformlyCauchySequence}^{\uparrow}$ *packet factors through*

$$S \longmapsto T \longmapsto W, D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently completion and filter consumers may use the displayed source, uniform tail threshold, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and local structural rows, but they receive no quotient stream, selected limit, hidden cofinal tail, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the displayed source S. The source is admissible for the packet only as a sequence row whose uniform Cauchy information is supplied by the threshold schedule T.

The finite observation then passes through W and D. The row W exposes the inspected stream windows at and beyond the chosen tail threshold, while D supplies the dyadic tolerance and comparison budget for the same request.

Only after those rows have been displayed does the packet expose R and then E. The row R is the regular rational readback, and E is the terminal Real seal served by that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Since none of the excluded objects occurs in the carrier, no completion-facing or filter-facing consumer can bypass the finite uniform schedule. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyCauchySequence}^{\uparrow}$ packet is generated from a sequence source row, a uniform tail-threshold schedule, a StreamName finite-window row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3696.1, Definition 3696.2, and Definition 3696.3 seed the finite uniform Cauchy schedule object and its regular-rational handoff for completion and filter consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, hidden cofinal tails, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-threshold exactness, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.UniformlyCauchySequenceUp.

Chapter 3697

A Concrete Naming Certificate for DyadicMeshScaleNormalization$^{\uparrow}$

Following Bishop and Bridges constructive analysis and the interval-normalization practice of Moore–Kearfott–Cloud interval computation, the DyadicMeshScaleNormalization$^{\uparrow}$ packet records the common-scale conversion needed before finite nets, dyadic approximations, and real-name seal windows compare mesh cells. It sits after the finite mesh carrier of chapter 358, the dyadic arithmetic core of chapter 291, the precision-scale ledger of chapter 1768, and the real embedding boundary of chapter 1942. The packet exposes a displayed common scale and its readback route; it is not host real equality, quotient stream equality, countable choice, or an appeal to ambient completeness.

Definition 3697.1 (Dyadic mesh scale-normalization carrier)**.** *A* DyadicMeshScaleNormalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, A, B, S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* DyadicMesh$^{\uparrow}$ *mesh row whose cells are to be compared or consumed by a finite-net route.*
2. *A and B are the source mesh-scale rows exposed before normalization.*
3. *S is the common-scale row, recording one dyadic scale that refines both A and B for the requested comparison window.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *arithmetic ledger for rescaling endpoints, widths, and cell representatives from A and B into S.*
5. *T is the* DyadicToleranceScale$^{\uparrow}$ *precision row that binds the normalization scale to the requested tolerance.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window row opened only after S, D, T have been displayed.*
7. *R is the* RegSeqRat$^{\uparrow}$ *handoff row obtained from the normalized dyadic windows.*
8. *E is the* RealDyadicEmbedding$^{\uparrow}$ *seal row reached after the normalized dyadic readback.*
9. *H records componentwise* hsame *transport for the mesh row, source scales, common scale, arithmetic ledger, tolerance row, window row, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a mesh comparison request through* $G, A, B, S, D, T, W, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealDyadicEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, A, B, S, D, T, W, R, E, H, C, P, N$. *It has no coordinate for host real equality, quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicMeshScaleNormalizationUp.DyadicMeshScaleNormalizationTasteGate`
`single carrier alignment`

Theorem 3697.2 (Dyadic mesh scale-normalization NameCert obligations)**.** *For every accepted* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ *carrier*

$$M = (G, A, B, S, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic mesh row G, source scale rows A, B, common-scale row S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *rescaling ledger D,* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *precision row T,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R,* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *seal row E, componentwise transport H, displayed continuation replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3697.1. The analytic rows are exactly the mesh row G, the two source scales A, B, the common scale S, the dyadic arithmetic ledger D, the tolerance-scale row T, the finite-window row W, the regular rational readback R, and the real-dyadic seal E.

The normalization obligation is read only from S, D, T. The row S names the common dyadic scale, D records the arithmetic that rewrites both source scales into that scale, and T binds the scale choice to the requested tolerance.

The stream and real-facing rows are downstream of the same normalization. Thus W opens only normalized finite windows, R reads the corresponding regular rational data, and E seals the route through $\mathsf{RealDyadicEmbedding}^{\uparrow}$ rather than through host equality of real numbers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3697.3 (Dyadic mesh common-scale exhaustion)**.** *Every finite-net, dyadic-approximation, or* $\mathsf{RealDyadicEmbedding}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{DyadicMesh}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{DyadicToleranceScale}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealDyadicEm}$$

Consequently such a consumer may read the original mesh, the two source scales, the common scale, the dyadic rescaling ledger, the tolerance row, normalized stream windows, regular rational readback, and real-dyadic seal, but it receives no host real equality, quotient stream, selected representative, hidden cofinal refinement, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an admitted consumer read. By Definition 3697.1, the only mesh-facing data before readback are G, A, B, S, D, T, W.

The common scale is forced through S. The rows A and B expose the two source scales, while D supplies the dyadic rescaling ledger that rewrites both into S under the tolerance request recorded by T.

Only after that scale has been displayed may the packet expose W, R, and E. The finite windows are therefore normalized before regular-rational readback, and the real-facing seal is the $\mathsf{RealDyadicEmbedding}^{\uparrow}$ row downstream of that readback.

The structural rows H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming. Hence every exported finite-net, approximation, or real-dyadic consumer factors through the displayed common-scale route and cannot bypass it by using any excluded host, quotient, choice, or completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMeshScaleNormalization}^{\uparrow}$ packet is generated from a DyadicMesh row, two source scale rows, one displayed common scale, a DyadicRatCore rescaling ledger, a DyadicToleranceScale precision row, StreamName finite windows, a RegSeqRat readback row, a RealDyadicEmbedding seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3697.1, Definition 3697.2, and Definition 3697.3 seed the common-scale normalization packet used by finite nets, dyadic approximation, and real-dyadic seal windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMeshScaleNormalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected representatives, hidden cofinal refinements, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-scale exactness, dyadic rescaling confinement, tolerance-scale binding, normalized StreamName coverage, RegSeqRat handoff, RealDyadicEmbedding seal non-escape, transport stability, continuation replay, and provenance confinement for BEDC.Derived.DyadicMeshScaleNormalizationUp.

Chapter 3698

A Concrete Naming Certificate for $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$

Following the Heine–Cantor and Bishop finite-net route, the $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ packet records the compact-domain uniform modulus data that completion-extension consumers need before a completed map can be read. It refines the compact-uniform-continuity vision of chapter 9237 into concrete rows over the compact metric packet of chapter 129, the continuous-map surface of chapter 130, the uniform modulus consumer of chapter 369, the Cauchy lift route of chapter 2037, and the regular completion readback rows of chapter 301, chapter 126, chapter 291, and chapter 38. The packet names the finite modulus consumer; it does not choose a pointwise selector, quotient a completion, or import ambient compactness or completeness.

Definition 3698.1 (Compact completion extension modulus carrier)**.** *A* $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, G, B, M, L, R, S, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the compact metric finite-net source row, carrying the finite net bundle, located radius rows, and coverage witnesses used by the uniform-modulus fold.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise modulus and graph row, read at bundled centers of the compact net.*
3. *B is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, recording the finite rational radius comparison that produces a tolerance-only precision history.*
4. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row opened by Q, G, B.*
5. *L is the* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *route row, exposing the modulus handoff before any completion-extension consumer reads lifted map data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular Cauchy approximants opened by the completion consumer.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to compare the compact-net source data with the completion readback.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying dyadic radii, error budgets, and transported comparison rows.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, visible only after Q, G, B, M, L, R, S, D have been displayed.*
10. *H records componentwise* hsame *transport for the compact source, graph, lower-bound fold, uniform modulus, lift route, readback, window, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of the route*

$$Q, G, B \longmapsto M, L \longmapsto R, S, D \longmapsto E.$$

12. *P is the* Pkg *provenance over* $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, G, B, M, L, R, S, D, E, H, C, P, N$. It has no coordinate for a global point selector, quotient completion equality, selected completed representatives, host function equality, open-cover compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3698.2 (Compact modulus preservation for completion-extension consumers)**.** *Let*

$$K = (Q, G, B, M, L, R, S, D, E, H, C, P, N)$$

be an accepted $\mathsf{CompactCompletionExtensionModulus}^{\uparrow}$ *carrier. Any completion-extension consumer that reads the compact-domain map must preserve the uniform modulus row M produced from the finite-net source Q, pointwise graph row G, and rational lower-bound fold B before it can inspect the* $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows.*

Lean4 ✓ `BEDC.Derived.CompactCompletionExtensionModulusUp.TasteGate.CompactCompletionExtensionModulus modulus preservation route`

Proof. Project the carrier. The compact finite-net source row Q, continuous graph row G, and rational lower-bound fold row B are displayed before the uniform modulus row M.

The completion-extension consumer then enters through L. Since L is downstream of M, the completion-facing map is read as data carrying the compactly produced uniform modulus rather than as a host function on completed representatives.

The readback rows R, S, D, E preserve this dependency order. Regular approximants, finite stream windows, dyadic tolerance budgets, and the terminal Real seal are all opened after the modulus handoff, so none can replace it with quotient completion equality or a selected limit.

The structural rows H, C, P, N only transport, replay, source, and name the same route. Therefore point selectors, host equality, open-cover compactness, choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the accepted consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3698.1 and Definition 3698.2 seed the compact finite-net uniform-modulus route consumed by completion-extension packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCompletionExtensionModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close point selector choice, quotient completion equality, selected completed representatives, host function equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net visibility, continuous-map pointwise modulus readback, rational lower-bound fold stability, uniform-modulus preservation, completion-lift routing, regular readback, finite-window stability, dyadic tolerance stability, Real seal non-escape, and ledger exhaustion for BEDC.Derived.CompactCompletionExtensionModulusUp.

Constructive story: A CompactCompletionExtensionModulus$^{\uparrow}$ packet is generated from CompactMetric finite-net source rows, a ContinuousMap pointwise graph row, a RatUp lower-bound fold, a UniformModulus consumer row, a CauchyCompletionLift route, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3699

A Concrete Naming Certificate for UniformlyContinuousCompletionExtension$^{\uparrow}$

Following Bishop and Bridges on uniformly continuous maps between completions, the UniformlyContinuousComplet packet records the finite NameCert surface by which a uniformly continuous source map extends across displayed BEDC completion data. It sits after the Cauchy lift route of chapter 2037, the uniform completion consumer of chapter 1892, the regular readback of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the real-name seal of chapter 38. The packet names the extension route through finite rows; it does not introduce quotient completion classes, selected representatives, host function extensionality, choice, or ambient Real$^{\uparrow}$ completeness.

Definition 3699.1 (Uniformly continuous completion extension carrier)**.** *A* UniformlyContinuousCompletionExtensi *carrier is a finite* BHist *packet*

$$X = (D, T, M, L, U, R, S, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source row, recording the displayed source observations, density witnesses, Cauchy windows, and tolerance requests made before any completed target is inspected.*
2. *T is the target completion-consumer row, exposing the target-facing completion object only through an accepted consumer interface.*
3. *M is the uniform modulus and graph row for the source map, carrying finite graph observations and the tolerance-to-precision ledger consumed by the extension.*
4. *L is the* CauchyCompletionLift$^{\uparrow}$ *handoff row, requiring the source map and its modulus to be read before a lifted completion-facing map is named.*
5. *U is the* UniformCompletionFunctor$^{\uparrow}$ *route row, binding the same source, target, graph, and modulus data to the uniform completion surface.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row, recording the regular rational approximants opened by the completion consumer.*
7. *S is the* StreamName$^{\uparrow}$ *finite-window row, recording the finite stream observations used to compare source Cauchy data with target readback data.*

8. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic radii and comparison budgets through the modulus handoff.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, visible only after the dense source, target consumer, modulus, lift, uniform completion, regular readback, stream-window, and dyadic tolerance rows have been displayed.*
10. *H records componentwise* hsame *transport for the dense source, target consumer, modulus graph, lift, uniform completion route, readback, stream windows, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of the route*

$$D, T, M \longmapsto L, U \longmapsto R, S, Y \longmapsto E.$$

12. *P is the* Pkg *provenance over* $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$, $\mathsf{CauchyCompletionLift}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, T, M, L, U, R, S, Y, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, host function extensionality, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3699.2 (Uniformly continuous completion extension modulus handoff)**.** *Let*

$$X = (D, T, M, L, U, R, S, Y, E, H, C, P, N)$$

be an accepted $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$ *carrier. Every extension consumer of the completed map factors through the uniform modulus and graph row M, the* $\mathsf{CauchyCompletionLift}^{\uparrow}$ *handoff row L, the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *route row U, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Y, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E before the structural rows H, C, P, N name the result.*

```
Lean4 ✓  BEDC.Derived.UniformlyContinuousCompletionExtensionUp.TasteGate.UniformlyContinuousCompletionEx
modulus handoff route
```

Proof. Carrier projection gives the dense source row D, the target completion-consumer row T, and the modulus-bearing graph row M. The carrier orders these rows before any lift, uniform completion, readback, finite-window, dyadic, or real-seal row.

The completion-facing handoff is then read through L and U. The row L records the same uniform-modulus handoff required by the Cauchy completion lift route, while U records the uniform completion consumer surface; neither row supplies a selected completed representative.

Regular readback is downstream of the handoff. The rows R, S, Y expose regular rational approximants, finite stream windows, and dyadic tolerance budgets before E may seal a real-facing read.

Finally H, C, P, N transport, replay, source, and name only the displayed route. Since the packet has no coordinate for quotient completion, selected representatives, host function extensionality, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no extension consumer can factor through those alternate surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3699.1 and Definition 3699.2 seed the finite modulus-handoff route for uniformly continuous maps extending across BEDC completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, host function extensionality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source visibility, uniform-modulus handoff, completion-lift routing, uniform-completion routing, regular readback, finite-window stability, dyadic tolerance stability, Real seal non-escape, and ledger exhaustion for BEDC.Derived.UniformlyContinuousCompletionExtensionUp.

Constructive story: A $\mathsf{UniformlyContinuousCompletionExtension}^{\uparrow}$ packet is generated from dense source rows, a target completion-consumer row, a uniform modulus and graph row, CauchyCompletionLift and UniformCompletionFunctor rows, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3700

A Concrete Naming Certificate for $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$

Following Bishop and Bridges' constructive monotone convergence argument, the $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ packet records the finite route from a bounded monotone real sequence to a located-supremum handoff and a terminal $\mathsf{Real}^{\uparrow}$ seal. It sits beside the located monotone convergence route of chapter 3373, the Bishop monotone limit packet of chapter 1914, the monotone sequence convergence surface of chapter 1435, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the bounded monotone case only; it does not assert arbitrary Real completeness, trichotomy, selected suprema outside the displayed located row, countable choice, or quotient-stream equality.

Definition 3700.1 (BishopMonotoneRealConvergence carrier)**.** *A* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, B, O, L, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded monotone* $\mathsf{Real}^{\uparrow}$ *sequence row, recording the source sequence, monotonicity direction, and the displayed upper or lower bound.*
2. *B is the boundedness ledger that exposes the finite bound witnesses used by the monotone route.*
3. *O is the order-comparison ledger that records the finite monotonicity reads consumed before any limit row is available.*
4. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row attached to the bounded monotone sequence data.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which source entries, bound witnesses, and located-supremum approximants are inspected.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite windows in W.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the requested precision and the dyadic comparison budget.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after L, W, Q, D have been displayed.*
9. *H records componentwise* hsame *transport for the sequence, bound, order, located-supremum, window, readback, tolerance, and real-seal rows.*

10. *C records displayed* Cont *replay from bounded monotone data through the located-supremum handoff and the regular rational readback to the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$, $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$, $\mathsf{BishopMonotoneRealLimit}^{\uparrow}$, $\mathsf{MonotoneSequenceConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, O, L, W, Q, D, E, H, C, P, N$. It has no coordinate for arbitrary Cauchy completeness, open-cover compactness, trichotomy of real order, quotient stream equality, selected host suprema, or countable choice.

Lean4 ✓ `BEDC.Derived.BishopMonotoneRealConvergenceUp.BishopMonotoneRealConvergenceTasteGate`
`single carrier alignment`

Theorem 3700.2 (BishopMonotoneRealConvergence NameCert obligations)**.** *For every accepted* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ *carrier $M = (S, B, O, L, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopMonotone}}$ *surface consists exactly of bounded monotone sequence exposure, bound-ledger exposure, monotonicity-read exposure, located-supremum handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

A consumer may read the convergence row only after S, B, O, L, W, Q, D have been displayed. The exported real seal is therefore the finite bounded-monotone route through located supremum and regular rational readback, not an ambient completeness theorem for arbitrary $\mathsf{Real}^{\uparrow}$ *data.*

Proof. Project the carrier of Definition 3700.1. The rows S, B, O expose the bounded monotone source data before any limit-facing row is available.

The row L supplies the located-supremum handoff. The finite windows W, regular readback Q, and dyadic tolerance ledger D then turn the located handoff into the displayed sequence of rational observations consumed by the terminal seal.

Finally E, H, C, P, N preserve the same route. Since the carrier has no coordinate for arbitrary Cauchy completeness, order trichotomy, quotient streams, selected host suprema, or countable choice, the listed rows exhaust the local NameCert surface. □

Theorem 3700.3 (BishopMonotoneRealConvergence located supremum route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *limit read exported by an accepted* $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ *packet factors through the displayed located-supremum row L and the finite readback route*

$$S, B, O \longmapsto L \longmapsto W, Q, D \longmapsto E, H, C, P, N.$$

Thus the bounded monotone convergence claim is a finite NameCert route from monotone source data to a located supremum and then to a $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Read the route rows in order. The source sequence S, boundedness row B, and order ledger O provide the only monotone data available to the packet.

The located-supremum handoff L is therefore the unique route by which a limit candidate enters the certificate. The window, readback, and tolerance rows W, Q, D inspect that handoff at finite precision before the real seal E can be consumed.

Transport, replay, provenance, and local naming are supplied by H, C, P, N over the same displayed rows. Hence an exported limit read cannot bypass L or import an ambient completeness principle outside the finite packet. □

Falsifiable prediction: Every accepted bounded monotone convergence read factors through a displayed located-supremum row before any RealUp seal is exported.

Independence witness: located-monotone-convergence, bishop-monotone-real-limit, monotone-sequence-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded monotone source data, bound and order ledgers, a located-supremum handoff, finite windows, regular rational readback, dyadic tolerance accounting, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMonotoneRealConvergence}^{\uparrow}$ packet is generated from bounded monotone RealUp sequence data, a boundedness ledger, monotonicity reads, a LocatedSupremum handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3700.1, Definition 3700.2, and Definition 3700.3 seed the bounded monotone Real convergence NameCert route through located supremum and finite regular-rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMonotoneRealConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary RealUp completeness, order trichotomy, open-cover compactness, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone source exposure, located-supremum handoff exactness, finite-window readback, dyadic tolerance accounting, RealUp seal routing, and TasteGate nontriviality with field faithfulness.

Chapter 3701

A Concrete Naming Certificate for $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy completion, the $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ packet records the finite classifier lock by which a diagonal limit candidate and its uniqueness witness are read through one completion-seal route. It sits after the Cauchy limit seal of chapter 386, the completion-seal classifier of chapter 900, the finite limit-seal envelope of chapter 1276, the regular Cauchy comparison surface of chapter 411, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the uniqueness classifier for the diagonal regular Cauchy limit, not an arbitrary choice of representatives or an ambient Real completeness principle.

Definition 3701.1 (CauchyLimitClassifierLock carrier)**.** *A* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, \mu, L, U, S, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the regular Cauchy input row over* $\mathsf{RegSeqRat}^{\uparrow}$*.*
2. *μ is the finite modulus-budget row used to select diagonal and comparison windows.*
3. *L is the diagonal limit-candidate row.*
4. *U is the uniqueness classifier row comparing any second candidate with L through the displayed RealUp classifier.*
5. *S is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *row carrying the common seal interface for L and U.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened for the input, diagonal candidate, and comparison candidate.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the diagonal and uniqueness comparisons.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, Q, D are displayed.*
10. *H records componentwise* hsame *transport for the input, modulus, diagonal candidate, uniqueness classifier, seal, windows, readback, tolerance, and real seal.*

11. *C records displayed* Cont *replay from regular Cauchy data through the diagonal candidate and classifier lock to the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, RealCompletionSealClassifierUp, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \mu, L, U, S, W, Q, D, E, H, C, P, N$. It has no coordinate for selected quotient representatives, host real equality, countable choice, arbitrary Cauchy completeness, or a second seal route outside S.

Lean4 ✓ `BEDC.Derived.CauchyLimitClassifierLockUp.CauchyLimitClassifierLockTasteGate single carrier alignment`

Theorem 3701.2 (CauchyLimitClassifierLock NameCert obligations)**.** *For every accepted* CauchyLimitClassifierLoc *carrier $K = (X, \mu, L, U, S, W, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$ *surface consists exactly of regular Cauchy input exposure, modulus-budget exposure, diagonal candidate exposure, uniqueness-classifier exposure,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

The uniqueness row U is accepted only when it is replayed through the same S, W, Q, D, E route as the diagonal candidate L. A consumer that compares candidates through a distinct host equality, quotient representative, or hidden completion seal leaves the packet.

Proof. Project the carrier of Definition 3701.1. The rows X, μ, L, U expose the regular input, modulus budget, diagonal candidate, and uniqueness classifier.

The seal row S is shared by the candidate and the uniqueness comparison. The finite windows W, regular readback Q, and dyadic tolerance ledger D are the only rows through which the comparison may reach the terminal real seal E.

Finally H, C, P, N preserve and replay this same route. Since no host equality, quotient representative, countable choice, arbitrary completeness row, or second seal route is a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3701.3 (CauchyLimitClassifierLock uniqueness route)**.** *Every uniqueness read exported by an accepted* $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ *packet factors through the common completion-seal route*

$$X, \mu, L, U \longmapsto S \longmapsto W, Q, D \longmapsto E, H, C, P, N.$$

Thus the diagonal limit candidate and the RealUp uniqueness classifier are locked to the same finite CauchyLimitSeal, RegSeqRat, StreamName, DyadicRatCore, and RealUp NameCert packet.

Proof. Start from an accepted uniqueness read. The carrier exposes only one diagonal candidate row L and one uniqueness classifier row U, both attached to the same regular input X and modulus budget μ.

The common seal row S is then read before any window, readback, tolerance, or real-seal row is exported. Consequently W, Q, D, E inspect the same completion-seal route for the candidate and for the uniqueness comparison.

Transport, replay, provenance, and local naming are supplied by H, C, P, N over this single route. A second classifier route would require an additional carrier coordinate, which Definition 3701.1 excludes. □

Falsifiable prediction: Every accepted Cauchy limit uniqueness read uses the same CauchyLimitSeal route as the displayed diagonal candidate before reaching RealUp.

Independence witness: cauchy-limit-seal, real-completion-seal-classifier, finite-limit-seal-envelope, regular-cauchy-comparison, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a regular Cauchy input, modulus budget, diagonal candidate, uniqueness classifier, common seal, finite windows, regular readback, dyadic tolerance ledger, RealUp seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyLimitClassifierLock}^{\uparrow}$ packet is generated from a regular Cauchy input, modulus budget, diagonal limit candidate, uniqueness classifier, CauchyLimitSeal handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3701.1, Definition 3701.2, and Definition 3701.3 seed the finite classifier-lock route for the Real completeness limit theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLimitClassifierLock}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary RealUp completeness, host equality of completed reals, selected quotient representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal candidate exposure, uniqueness classifier exactness, common CauchyLimitSeal routing, finite-window readback, dyadic tolerance accounting, RealUp seal routing, and TasteGate nontriviality with field faithfulness.

Chapter 3702

A Concrete Naming Certificate for $\mathsf{DarbouxTheorem}^{\uparrow}$

Following Darboux and Bishop–Bridges constructive analysis, the $\mathsf{DarbouxTheorem}^{\uparrow}$ packet records the theorem route saying that a displayed continuous real map on a displayed interval has the Darboux property. It sits over the graph-and-modulus surface of chapter 130, the constructive finite bisection route of chapter 1422, the target-value handoff of chapter 3348, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. The chapter packages Darboux's theorem as finite NameCert data rather than as host continuity semantics, a selected intermediate point, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3702.1 (Darboux theorem carrier)**.** *A* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row carrying the displayed graph, pointwise modulus, and continuity budget for the real-valued map.*
2. *I is the interval endpoint row, exposing the lower and upper endpoint reads and their orientation data.*
3. *A and B are the endpoint value rows obtained from F at the two endpoint rows of I.*
4. *Y is the target-value row displayed between the endpoint value rows.*
5. *S is the endpoint-side exposure row, recording which finite residual comparisons place Y between A and B.*
6. *D is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *schedule that retains branches whose residual comparisons still bracket Y.*
7. *R is the* $\mathsf{DarbouxProperty}^{\uparrow}$ *handoff row reached after the bisection schedule has produced the retained target-value route.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained windows and requested dyadic tolerances.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal available to theorem consumers only after D, R, Q have been read.*

10. *H stores componentwise* hsame *transport for the map, interval, endpoint values, target value, endpoint-side exposure, bisection schedule, property handoff, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C stores displayed* Cont *replay from a Darboux-theorem request through $F, I, A, B, Y, S, D, R, Q, E$.*
12. *P is the* Pkg *provenance over* $\mathsf{DarbouxTheorem}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted theorem packets only through $F, I, A, B, Y, S, D, R, Q, E, H, C, P, N$. It has no coordinate for host real equality, quotient real equality, a selected intermediate point, real trichotomy, open-cover compactness, semantic continuity outside F, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DarbouxTheoremUp.DarbouxTheoremTasteGate single carrier alignment`

Theorem 3702.2 (Darboux theorem NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier $T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission through the displayed packet of Definition 3702.1, endpoint-side exposure through I, A, B, Y, S, bisection exactness through D, Darboux-property handoff through R, Real-seal readback through Q, E, componentwise transport through H, displayed continuation replay through C, provenance through P, local naming through N, and non-escape from host real equality, quotient real equality, selected intermediate points, trichotomy, open-cover compactness, semantic continuity outside F, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3702.1. The continuous-map row F and interval row I expose the only theorem source data, while A, B, Y, S expose the finite endpoint-side comparisons for the requested target value.

The bisection schedule is the row D. It is the only branch-retention route in the packet, so every accepted theorem read must pass through the finite $\mathsf{IntermediateValueBisection}^{\uparrow}$ schedule before a target-value handoff is visible.

The property handoff is then the row R, and the Real-facing readback is Q, E. The row Q supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback against the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data named by provenance, and E is the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier has no coordinate for host equality, quotient equality, a selected intermediate point, trichotomy, open-cover compactness, semantic continuity outside F, or ambient Real completeness, none of those objects can appear in the local NameCert surface. □

Theorem 3702.3 (Darboux theorem bisection route)**.** *Every accepted theorem read of a* $\mathsf{DarbouxTheorem}^{\uparrow}$ *carrier factors through the finite route*

$\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{DarbouxProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsam

In particular, the read first consumes the displayed map, endpoint rows, target-value row, and endpoint-side exposure; then the finite bisection schedule; then the Darboux-property handoff; then the regular rational and dyadic readback; and only then the $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start from an accepted carrier $T = (F, I, A, B, Y, S, D, R, Q, E, H, C, P, N)$. The continuation row C exposes theorem reads only through the map, interval, endpoint value, target value, and endpoint-side rows before any Real-facing seal is available.

The finite bisection route is forced by D. The carrier contains no alternate theorem coordinate between the endpoint-side exposure S and the Darboux-property handoff R, so a consumer cannot bypass $\mathsf{IntermediateValueBisection}^{\uparrow}$ by appealing to host connectedness or semantic continuity.

After the bisection schedule, the theorem read reaches R as the $\mathsf{DarbouxProperty}^{\uparrow}$ handoff. The retained windows are then read through Q using the dyadic tolerance surface named in P, and the terminal read reaches E as the $\mathsf{Real}^{\uparrow}$ seal.

The rows H, C, P, N keep this route stable under transport, replay, provenance, and local naming. Thus every accepted theorem read factors through the displayed finite bisection schedule before the Real seal, with the excluded host and completeness objects remaining outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxTheorem}^{\uparrow}$ packet is generated from a ContinuousMap row, interval endpoint rows, endpoint value rows, a target-value row, endpoint-side exposure, an IntermediateValueBisection schedule, a Darboux-Property handoff, RegSeqRat readback, a DyadicRatCore tolerance surface, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3702.1, Definition 3702.2, and Definition 3702.3 seed Darboux's theorem as a finite bisection route from displayed endpoint-side exposure to DarbouxProperty handoff and Real-seal readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DarbouxTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient real equality, selected intermediate points, trichotomy, open-cover compactness, semantic continuity outside the displayed ContinuousMap row, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-side exposure, bisection exactness, DarbouxProperty handoff, Real-seal readback, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.DarbouxTheoremUp.

Chapter 3703

A Concrete Naming Certificate for LocatedCompletionUniversalProperty$^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the LocatedCompletionUniversalProperty$^{\uparrow}$ packet records the extension-and-uniqueness surface for located completion as finite BEDC NameCert data. It sits after the located completion seal of chapter 1805, the generic completion universal-property packet of chapter 397, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet names located extension and uniqueness only through displayed source, route, modulus, transport, replay, provenance, and local naming rows; it is not a quotient completion, a selected host limit, countable choice, ambient real completeness, or function extensionality.

Definition 3703.1 (Located completion universal-property carrier)**.** *A* LocatedCompletionUniversalProperty$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, E, Q, M, H, C, P, N)$$

with the following displayed rows:

1. *L is the located completion source row, exposing the source metric, locatedness request surface, dense unit, and completion seal read from chapter 1805.*
2. *U is the universal extension row, read against the target row and the generic completion-universal interface of chapter 397.*
3. *E is the extension uniqueness row, comparing two displayed extensions only over the same located source, dense unit, and target route.*
4. *Q is the request route through* RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$*, carrying finite windows, regular rational readback, and the observations used by located requests.*
5. *M is the uniform-modulus compatibility row, tying source moduli, target moduli, and located request tolerances to the same extension route.*
6. *H records componentwise* hsame *transport for the located source, universal extension, uniqueness row, request route, modulus row, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from located source requests through finite stream windows, regular rational readback, uniform-modulus compatibility, extension, and uniqueness.*

8. *P is the* Pkg *provenance over* LocatedCompletionUniversalProperty$^{\uparrow}$, LocatedCompletionSeal$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. *N is the local* NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, E, Q, M, H, C, P, N$. It has no coordinate for quotient completion representatives, selected limits, ambient complete-space theorems, function extensionality, countable choice, or unlocated real equality.

Lean4 ✓ `BEDC.Derived.LocatedCompletionUniversalPropertyUp.LocatedCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Theorem 3703.2 (Located completion universal-property NameCert obligations)**.** *For every accepted* LocatedCompletionUniversalProperty$^{\uparrow}$ *carrier $K = (L, U, E, Q, M, H, C, P, N)$, the local* NameCert$_{\text{LocatedComple}}$ *surface consists exactly of located-source admission through L, universal extension exposure through U, extension uniqueness through E,* RegSeqRat$^{\uparrow}$ *and* StreamName$^{\uparrow}$ *request routing through Q, uniform-modulus compatibility through M, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3703.1. The located completion source row L is visible before any extension row is consumed, and it reads the dense unit and completion seal through chapter 1805.

Read the universal-property rows next. The row U supplies the displayed extension route, while E compares two such routes only along the same located source and dense unit. The generic obligation surface is therefore the one already named by chapter 397, refined by located request data rather than by a new host completion object.

The request and modulus rows keep the route finite. The row Q exposes the StreamName$^{\uparrow}$ windows and RegSeqRat$^{\uparrow}$ readback consumed by located requests, and M ties those requests to the same uniform-modulus compatibility ledger before the Real$^{\uparrow}$ seal is read.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier has no quotient representative, selected host limit, ambient completeness theorem, function-extensionality row, countable-choice row, or unlocated real-equality coordinate, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted LocatedCompletionUniversalProperty$^{\uparrow}$ extension or uniqueness read factors through the displayed located source row, finite request route, uniform-modulus compatibility row, transport, replay, provenance, and local naming before reaching a Real$^{\uparrow}$ consumer.

Independence witness: located-completion-seal, completionuniversalproperty, streamname, regseqrat, real: LocatedCompletionUniversalProperty$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a located completion source, universal extension row, extension uniqueness row, finite request route, uniform-modulus compatibility, transport, replay, provenance, and local NameCert row.

Closure status for NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$ **·**

Constructive story: A LocatedCompletionUniversalProperty$^{\uparrow}$ packet is generated from a located completion source row, universal extension row, extension uniqueness row, RegSeqRat and StreamName request route, uniform-modulus compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedCompletionUniversalProperty}^{\uparrow}}$, seed)

Scope closed: Definition 3703.1 and Definition 3703.2 seed the finite located-completion universal-property certificate surface.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCompletionUniversalProperty}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, ambient complete-space theorems, unlocated real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located request routing, extension existence, extension uniqueness, uniform-modulus compatibility, transport replay, provenance, and TasteGate field faithfulness for BEDC.Derived.LocatedCompletionUniversalPropertyUp.

Chapter 3704

A Concrete Naming Certificate for $\mathsf{BooleanTriggerLattice}^{\uparrow}$

The $\mathsf{BooleanTriggerLattice}^{\uparrow}$ packet realizes the Boolean trigger closure lattice of Definition 9462.1520 and Definition 9462.1532 as finite BEDC NameCert data. It sits over the finite truth-tag discipline of chapter 32, the six-boundary-tile coverage route of chapter 3470, and the trigger-algebra sibling surface of chapter 3478. The packet names Boolean support, rank, meet, join, spectral table, transport, replay, provenance, and local naming rows; it is not a biological code oracle, stochastic evolution model, host powerset lattice, quotient codon semantics, or infinite trigger search.

Definition 3704.1 (BooleanTriggerLattice carrier)**.** *A* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, R, J, K, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite* $\mathsf{Bool}^{\uparrow}$ *tag row for present, absent, covered, uncovered, full-trigger, and non-trigger reads.*
2. *S is the six-boundary support row, read from* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *and carrying the displayed coordinates* $\{s_1, f_1, s_2, f_2\}$*.*
3. *R is the rank row, naming the finite support union $J(B)$ and the rank $r(B)$ of the source vision.*
4. *J is the join-trigger row, recording median closure of a union as support union.*
5. *K is the meet-trigger row, recording intersection of trigger closures as support intersection.*
6. *O is the observed table row, carrying the finite size, cubical, spectral, threshold, and singleton-trigger reads displayed in the source vision.*
7. *H records componentwise* hsame *transport for Boolean tags, support rows, rank rows, join rows, meet rows, observed table rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from each accepted trigger read through B, S, R, J, K, O, H.*
9. *P is the* Pkg *provenance over* $\mathsf{BooleanTriggerLattice}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* WindowCoverageTriggerA *the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, R, J, K, O, H, C, P, N$. It has no coordinate for host powersets, biological-code truth, stochastic dynamics, quotient codon equality, hidden spectral computation, or unbounded trigger enumeration.

Lean4 ✓ `BEDC.Derived.BooleanTriggerLatticeUp.BooleanTriggerLatticeTasteGate single carrier alignment`

Theorem 3704.2 (BooleanTriggerLattice NameCert obligations)**.** *For every accepted* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *carrier* $T = (B, S, R, J, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$ *surface consists exactly of Boolean tag admission, six-boundary support exposure, finite rank exposure, join-trigger exposure, meet-trigger exposure, observed table exposure, componentwise transport, displayed replay, provenance, and local naming.*

The future Lean TasteGate plan for this packet requires $\mathsf{Nontrivial\ BooleanTriggerLattice}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ BooleanTriggerLattice}^{\uparrow}$ *so that* $B, S, R, J, K, O, H, C, P, N$ *remain separate displayed fields rather than a collapsed tag.*

Proof. Project the carrier of Definition 3704.1. The rows B, S, R, J, K, O expose the Boolean tags, boundary support, rank, join, meet, and observed table before any consumer can read a trigger-lattice conclusion.

The join and meet rows are finite. The row J reads support union from the source vision, while K reads support intersection. Both rows depend on the same six-boundary support surface and therefore cannot be replaced by an ambient powerset lattice.

The observed table row is also displayed. It records the finite rank-size, cubical, spectral, threshold, and singleton-trigger data already present in the vision file, but it does not add a stochastic model or a hidden spectral oracle.

Finally H, C, P, N preserve the same displayed route. Since no host powerset, biological truth oracle, quotient codon equality, stochastic dynamics, or unbounded trigger enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Falsifiable prediction: An accepted $\mathsf{BooleanTriggerLattice}^{\uparrow}$ read is refuted if its full-trigger, rank, meet, or join decision can be produced without the displayed Boolean tags, six-boundary support row, finite rank row, observed table row, transport, replay, provenance, and local naming.

Independence witness: bool, six-boundary-tile-coverage, window-coverage-trigger-algebra, boolean-trigger source vision: none is bijective with $\mathsf{BooleanTriggerLattice}^{\uparrow}$ because the packet jointly carries Boolean tags, six-boundary support, trigger rank, join rows, meet rows, observed rank-spectral table data, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{BooleanTriggerLattice}^{\uparrow}$ packet is generated from BoolUp tag rows, six-boundary support rows, finite rank rows, join-trigger rows, meet-trigger rows, observed rank-spectral table rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3704.1 and Definition 3704.2 seed the finite Boolean trigger-lattice NameCert packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BooleanTriggerLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, host powerset lattices, quotient codon equality, hidden spectral computation, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Boolean tag exposure, support-rank exposure, meet and join rows, observed table exposure, Nontrivial BooleanTriggerLatticeUp, and FieldFaithful BooleanTriggerLatticeUp for BEDC.Derived.BooleanTriggerLatticeUp.

Chapter 3705

A Concrete Naming Certificate for $\mathsf{BishopCauchyComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyComparison}^{\uparrow}$ packet records the finite comparison route between Cauchy-criterion reads and convergence-modulus reads before a $\mathsf{Real}^{\uparrow}$ or completion consumer can inspect a terminal seal. It sits beside the criterion surface of chapter 392, the bidirectional criterion/modulus surface of chapter 3343, the convergence-modulus boundary of chapter 2074, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic arithmetic core of chapter 291, and the terminal real-name boundary of chapter 38. The packet names the finite comparison only; it does not assert quotient-stream equality, selected limits, countable choice over thresholds, host convergence, or ambient completeness.

Definition 3705.1 (Cauchy criterion comparison chain)**.** *A* $\mathsf{BishopCauchyComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *window row, displaying the finite sequence windows read by both presentations.*
2. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *late-pair request row, recording the criterion-facing tolerance read.*
3. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *schedule row, recording the modulus-facing threshold hand-off.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by the criterion request and the modulus schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which both presentations inspect the same rational windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the source windows, criterion request, modulus schedule, tolerance, readback, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from either criterion-facing or modulus-facing entry through the same finite route.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyComparison}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, M, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable-choice threshold selection, detached completion, host convergence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyComparisonUp.BishopCauchyComparisonTasteGate single carrier alignment`

Theorem 3705.2 (Cauchy criterion comparison NameCert obligations)**.** *For every accepted* BishopCauchyComparis *carrier $B = (S, Q, M, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$ *surface consists exactly of carrier admission, criterion-to-modulus handoff, modulus-to-criterion handoff, shared tolerance-readback exactness,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise transport, displayed replay, provenance exhaustion, and local naming.*

A criterion-facing consumer may read the modulus schedule M only through the same S, D, R, E, H, C, P, N rows that support the criterion request Q. Conversely, a modulus-facing consumer may read the criterion request Q only through those same rows. Thus the packet supplies a finite comparison chain rather than a host equality between presentations.

Proof. Project the carrier of Definition 3705.1. Carrier admission exposes exactly the tuple $S, Q, M, D, R, E, H, C, P, N$.

For the criterion-to-modulus direction, the late-pair request Q is read together with the source windows S, the shared dyadic tolerance row D, and the regular rational readback row R. The modulus schedule M is visible only as the threshold row attached to that same finite data.

For the modulus-to-criterion direction, the schedule M is read with the same source, tolerance, and readback rows. The criterion request Q is therefore reconstructed as a finite late-pair obligation over the displayed windows, not as an independent convergence predicate.

The real seal E is downstream of both reads. Transport, replay, provenance, and naming are supplied by H, C, P, N over the same rows, so no quotient stream, selected limit, countable-choice selector, host convergence theorem, or ambient completeness row can enter the local certificate. □

Theorem 3705.3 (Cauchy criterion comparison real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyComparison}^{\uparrow}$ *packet reaches the terminal seal only through the finite route*

$$S, Q, M, D, R \longmapsto E \longmapsto H, C, P, N.$$

In particular, neither the criterion presentation nor the modulus presentation can bypass the shared tolerance-readback route or replace it by quotient-stream equality, selected limits, host convergence, countable choice, or ambient completion.

Proof. Use Definition 3705.2. Both handoff directions force the criterion row Q and modulus row M to share the source windows S, tolerance row D, and readback row R.

The terminal row E is listed only after those shared rows. A $\mathsf{Real}^{\uparrow}$ or completion-facing consumer therefore cannot inspect the seal before the finite comparison route has been displayed.

Finally H, C, P, N preserve and name that same route. Since the carrier has no row for quotient streams, selected limits, countable-choice schedules, host convergence, or detached completion, the seal cannot escape through such data. □

Falsifiable prediction: Every accepted criterion/modulus comparison read exposes the shared StreamName, RegSeqRat, and DyadicRatCore rows before any RealUp or completion-facing seal is exported.

Independence witness: cauchycriterion, cauchy-criterion-equivalence, real-modulus-of-convergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopCauchyComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries criterion request, modulus schedule, shared tolerance-readback, RealUp seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3705.1, Definition 3705.2, and Definition 3705.3 seed the finite comparison route between Cauchy-criterion reads and convergence-modulus reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-stream equality, selected limits, countable-choice threshold selection, host convergence, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bidirectional handoff exactness, shared tolerance-readback exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyComparisonUp.

Constructive story: A $\mathsf{BishopCauchyComparison}^{\uparrow}$ packet is generated from source StreamName windows, a Cauchy-Criterion request, a RealModulusOfConvergence schedule, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3706

A Concrete Naming Certificate for $\mathsf{ClosedObservationTotalHost}^{\uparrow}$

The $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ packet realizes the total-host discipline of chapter 9370 as finite NameCert data. It sits beside the candidate-mediated normalization socket of chapter 3566, the MetaCIC normalization frontier of chapter 892, the total-host fuel handoff of chapter 962, and the reduction fuel boundary of chapter 970. The packet names a discipline on accepted host-observation records: a finite termination or observation row must be visible before substrate-step readback. It does not claim global normalization of arbitrary host calculi, substrate halting, or evaluator soundness outside the displayed rows.

Definition 3706.1 (ClosedObservationTotalHost carrier)**.** *A* $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, T, S, F, R, H, C, P, N)$$

with the following displayed rows:

1. *A is the accepted host-observation row, displaying the closed observation record and its host-side typing boundary.*
2. *T is the totality row, recording the finite termination or normal-form readback required before the record can be consumed.*
3. *S is the substrate-step row, displaying the finite step-indexed substrate observation that the host record describes.*
4. *F is the refusal row for any consumer that asks to read S before A, T have been displayed.*
5. *R is the readback row binding the host observation to the substrate-step record without asserting substrate halting.*
6. *H records componentwise* hsame *transport for accepted observation, totality, substrate-step, refusal, readback, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a closed-observation request through A, T, S, F, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedObservationTotalHost}^{\uparrow}$*, chapter 9370,* $\mathsf{NormalizationCandidateSocket}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{TotalHostFuelHandoff}^{\uparrow}$*,* $\mathsf{ReductionFuelBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, T, S, F, R, H, C, P, N$. It has no coordinate for unbounded substrate halting, non-total host terms, erased termination evidence, host evaluator soundness, completed consistency, global confluence, or quotient syntax equality.

Theorem 3706.2 (ClosedObservationTotalHost NameCert obligations)**.** *For every accepted* ClosedObservationTota *carrier $O = (A, T, S, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$ *surface consists exactly of accepted observation exposure, finite termination or normal-form exposure, substrate-step readback, pre-read refusal, readback binding, componentwise transport, displayed replay, provenance exhaustion, and local naming.*

Every accepted host-observation record exposes A and T before any consumer can read S. The substrate row is therefore a finite step-indexed observation named by a total host record, not a claim that the substrate computation halts or that arbitrary host computations are accepted.

Lean4 ✓ `BEDC.Derived.ClosedObservationTotalHostUp.ClosedObservationTotalHostNameCertObligations`

Proof. Project the carrier of Definition 3706.1. The accepted record A and totality row T are distinct rows, so accepting the host observation does not erase the termination or normal-form evidence.

The substrate row S is downstream of A, T. If a consumer asks for substrate readback first, the refusal row F is the only available route, so the packet records the total-host discipline as a finite ordering of rows.

The row R then binds the displayed host observation to the displayed substrate step. This binding is step-indexed and finite; it is not a substrate-halting theorem and does not import an unbounded evaluator.

Finally H, C, P, N transport, replay, source, and name exactly these rows. Since no coordinate carries non-total host terms, erased termination evidence, host evaluator soundness, global confluence, quotient syntax equality, or completed consistency, none of those principles can be exported from the packet. □

Falsifiable prediction: Every accepted host-observation record exposes a finite termination or observation row before any substrate-read row is available to a consumer.

Independence witness: normalization-candidate-socket, metacicnormalizationfrontier, continuationtrace, certificate-machine: $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries accepted observation, finite totality, substrate-step, refusal, readback, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3706.1 and Definition 3706.2 seed the finite total-host discipline for closed observation records before substrate-step readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedObservationTotalHost}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ClosedObservationTotalHostUp.ClosedObservationTotalHostNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close global host normalization, substrate halting, host evaluator soundness, global confluence, quotient syntax equality, completed consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, accepted-observation exposure, totality-row precedence, substrate-read refusal, readback binding, transport, replay, provenance, and local naming for BEDC.Derived.ClosedObservationTota

Constructive story: A $\mathsf{ClosedObservationTotalHost}^{\uparrow}$ packet is generated from an accepted host-observation row, a finite totality row, a substrate-step row, a pre-read refusal row, a readback binding row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3707

A Concrete Naming Certificate for $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ packet records the finite BEDC route by which regular rational Cauchy addition and product share one source surface, one window discipline, one dyadic arithmetic ledger, and one terminal real-name seal. It refines the binary route of chapter 3708 by isolating the coherence rows that make the addition and product certificates descend through the same $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ interfaces. The packet supplies later Real algebra consumers with a finite $\mathsf{NameCert}$ surface for shared operation descent, without importing quotient Cauchy classes, host real arithmetic, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3707.1 (RegularCauchyAlgebraCoherence carrier)**.** *A* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, R, W, D, A, M, S, T, C, P, N)$$

with the following rows:

1. *L and R are the left and right* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows.*
2. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect both sources at the requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger containing the displayed slack conversions, endpoint arithmetic, comparison checks, addition budgets, product budgets, and magnitude bounds.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *descent row. It consumes only L, R, W, D and returns the regular addition readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent row. It consumes only L, R, W, D and returns the regular product readback.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *operation seal. It exposes an addition or product real-name handoff only after the corresponding regular readback row has been displayed.*
7. *T records componentwise* hsame *transport for the sources, window, dyadic ledger, addition row, product row, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an operation request through $L, R, W, D, A, M, S, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, R, W, D, A, M, S, T, C, P, N$. It has no coordinate for quotient-stream equality, a host real operation, a selected limit, countable choice, trichotomy, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyAlgebraCoherenceUp.RegularCauchyAlgebraCoherenceTasteGate`
`single carrier alignment`

Theorem 3707.2 (RegularCauchyAlgebraCoherence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier*

$$K = (L, R, W, D, A, M, S, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, the shared* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *row, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *row, the* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source coherence, requiring addition and product to read the same ordered source pair L, R;*
3. *window coherence, requiring both operation rows to inspect the source pair only through W;*
4. *dyadic ledger coherence, requiring every addition and product estimate to consume only the displayed arithmetic and comparison entries of D;*
5. *descent separation, requiring A to carry the addition descent and M to carry the product descent while sharing the same upstream rows;*
6. *operation sealing, requiring S to expose a* $\mathsf{Real}^{\uparrow}$ *handoff only after the selected operation row has produced its regular readback;*
7. *stability and replay, requiring T and C to preserve the full route $L, R, W, D, A, M, S, P, N$ under* hsame *and* Cont *without adding any excluded analytic or quotient principle.*

No RegularCauchyAlgebraCoherence NameCert obligation is stored outside these rows.

Proof. Carrier exposure is the projection list in Definition 3707.1. The displayed data rows are exactly L, R, W, D, A, M, S, and the structural rows are exactly T, C, P, N.

Source and window coherence follow from the single source pair and the single window row. Since both A and M cite L, R, W as their upstream surface, there is no second source pair or hidden window service from which either operation can draw.

The dyadic ledger row gives the common arithmetic budget. Addition consumes it through A, product consumes it through M, and the theorem asserts only those two separated descent rows over the same ledger.

Finally, S, T, C, P, N close the finite certificate surface. The seal exposes only regular readback data as a $\mathsf{Real}^{\uparrow}$ handoff, while transport, replay, provenance, and local naming preserve the displayed route. None of the excluded principles has a carrier coordinate. □

Theorem 3707.3 (RegularCauchyAlgebraCoherence operation seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier reads regular Cauchy addition or product data through the finite route*

$$L, R \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto T, C, P, N.$$

Consequently the consumer obtains the two regular source rows, the shared stream window, the dyadic ledger, the addition descent row, the product descent row, the operation seal, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a $\mathsf{Real}^{\uparrow}$ operation request. By Definition 3707.1, the request has access to the source rows only as L, R, and it has access to their finite observations only through W.

The arithmetic step is then forced through D. Addition reads A, product reads M, and both descent rows are downstream of the same source, window, and dyadic ledger rows recorded in Definition 3707.2.

The seal S is the only Real-facing coordinate. It can expose an addition or product handoff only after the corresponding regular readback is displayed, and the rows T, C, P, N preserve that same finite route. Therefore a $\mathsf{Real}^{\uparrow}$ consumer has no coordinate from which to extract any excluded quotient, host, choice, order, or completeness datum. □

Theorem 3707.4 (RegularCauchyAlgebraCoherence non-escape)**.** *Let* $K = (L, R, W, D, A, M, S, T, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ *carrier. Any algebraic fact exported from* K *is a rowwise consequence of the displayed regular source rows, shared stream window, dyadic ledger, operation descent rows, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly L, R, W, D, A, M, S, and the structural coordinates are exactly T, C, P, N.

An exported addition fact must pass through A after the common source, window, and ledger rows are displayed. An exported product fact must pass through M after the same upstream rows are displayed. A Real-facing fact must then pass through S.

The remaining rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot introduce a quotient relation, host operation, selected limit, choice principle, trichotomy principle, or completeness principle, while Pkg and NameCert record only the finite packet. Hence every exported algebraic fact is confined to the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3707.1, Definition 3707.2, Definition 3707.3, and Definition 3707.4 seed the finite coherence route for addition and product descent over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source coherence, window coherence, dyadic ledger coherence, descent separation, operation sealing, stability, replay, provenance, local naming, and non-escape for BEDC.Derived.RegularCauchyAlgebraCoherenceUp.

Constructive story: A $\mathsf{RegularCauchyAlgebraCoherence}^{\uparrow}$ packet is generated from two $\mathsf{RegSeqRat}^{\uparrow}$ source rows, one shared $\mathsf{StreamName}^{\uparrow}$ window, one $\mathsf{DyadicRatCore}^{\uparrow}$ ledger, $\mathsf{RegularCauchyAddition}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$ descent rows, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3708

A Concrete Naming Certificate for $\mathsf{CauchySequenceAlgebra}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ packet records the finite algebra route by which two regular rational Cauchy source rows are added or multiplied before any $\mathsf{Real}^{\uparrow}$ consumer reads the sealed result. It sits between the regular source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the addition route of chapter 1695, the product route of chapter 461, and the terminal real-name seal of chapter 38. The packet gives later Real algebra and CauchyRealField consumers one BEDC-native finite dependency for binary Cauchy algebra; it does not import quotient Cauchy classes, host real arithmetic, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3708.1 (CauchySequenceAlgebra carrier)**.** *A* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations supply the left and right inputs.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite window pairs in which the two source rows are inspected at the same requested tolerance scale.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row. It stores the displayed endpoint arithmetic, slack conversions, tolerance comparisons, addition-budget entries, and product-budget entries used by the accepted binary operation request.*
4. *A is the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the summed endpoint row, finite addition ledger, and resulting regular readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row. It may be read only through X, Y, W, D and records the product endpoint row, product-error ledger, magnitude-budget entries, and resulting regular readback.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff. It is exposed only after the chosen operation row A or M has displayed its regular readback.*
7. *H records componentwise* hsame *transport for the two source rows, synchronized windows, dyadic budgets, operation rows, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a binary algebra request through* $X, Y, W, D, A, M, S, H, P, N$.
9. *P is the* Pkg *provenance over* CauchySequenceAlgebra$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, A, M, S, H, C, P, N$. The carrier has no coordinate for quotient-stream equality, host real addition or multiplication, a selected limit, countable choice, trichotomy, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 3708.2 (CauchySequenceAlgebra NameCert obligations)**.** *For every accepted* CauchySequenceAlgebra$^{\uparrow}$ *carrier*

$$G = (X, Y, W, D, A, M, S, H, C, P, N),$$

the local NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the two* RegSeqRat$^{\uparrow}$ *sources, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *operation budgets,* RegularCauchyAddition$^{\uparrow}$ *route,* RegularCauchyProduct$^{\uparrow}$ *route,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *source-window synchronization, requiring every operation request to inspect X and Y through the same displayed window row W before any endpoint arithmetic is read;*
3. *dyadic-budget exactness, requiring every addition or product endpoint row to consume only the displayed tolerance, slack, comparison, and magnitude-budget entries of D;*
4. *operation-route separation, requiring A to carry only the addition route and M to carry only the product route, while both routes consume the same source, window, and dyadic-budget rows;*
5. *Real seal handoff, requiring S to be exposed only after the selected operation route has produced its* RegSeqRat$^{\uparrow}$ *readback;*
6. *transport and replay stability, requiring H and C to preserve the full route $X, Y, W, D, A, M, S, P, N$ without adding a quotient stream, host operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.*

No CauchySequenceAlgebra NameCert obligation is stored outside these rows.

Proof. Carrier exposure is row projection from Definition 3708.1. The displayed coordinates are exactly the two source rows X, Y, synchronized windows W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Source-window synchronization is forced by the single window coordinate W. Both source reads must pass through that coordinate before the dyadic operation budget is available, so the carrier has no second window service for either input.

Dyadic-budget exactness and operation-route separation are read from D, A, M. The row D supplies the finite arithmetic budgets, the row A consumes those budgets for addition, and the row M consumes those budgets for product. Since A and M are distinct rows with the same upstream inputs, neither route can borrow an unrecorded host operation.

Finally, the Real seal and structural rows close the finite route. The seal S is downstream of the operation readback, while H, C, P, N only transport, replay, package, and name the displayed rows. The excluded quotient, host arithmetic, selected-limit, choice, trichotomy, and ambient-completeness data have no carrier coordinate. □

Theorem 3708.3 (CauchySequenceAlgebra Real-field handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *binary algebra read reaches field-facing data only through the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *operation-budget row, the* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *routes, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In particular the handoff exports regular Cauchy addition and multiplication readbacks before the final Real seal, and it exports no quotient stream, host real arithmetic, selected limit, trichotomy, countable-choice datum, or ambient completeness principle.*

Proof. Project the carrier of Definition 3708.1. The only field-facing source data are the two regular rational Cauchy rows X, Y, the synchronized window W, and the dyadic operation budget D.

Read the operation rows before the final Real seal. The addition route is A, the product route is M, and both routes consume the same source, window, and budget rows named in Definition 3708.2.

The terminal consumer then reads S only after one of those operation routes has produced its $\mathsf{RegSeqRat}^{\uparrow}$ readback. Transport, replay, provenance, and local naming in H, C, P, N preserve that route. Since the carrier has no coordinate for quotient streams, host arithmetic, selected limits, trichotomy, countable choice, or ambient completeness, no Real-field handoff can export any of those objects. □

Theorem 3708.4 (CauchySequenceAlgebra operation route)**.** *Every* $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{RegularCauchyProdu}$ $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{CauchyRealField}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier reads binary algebra data only through the displayed route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N.$$

Consequently the consumer obtains the two regular source rows, synchronized stream windows, dyadic operation budgets, the addition route, the product route, the terminal Real seal handoff, transport, replay, provenance, and local naming, and obtains no quotient Cauchy class, host real operation, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream algebra request. By Definition 3708.1, the only source coordinates are X and Y, and the only synchronized window coordinate available to both sources is W.

The operation budget is then forced through D. Addition reads the route row A, product reads the route row M, and both rows are displayed downstream of the same sources, windows, and dyadic budgets named in Definition 3708.2.

The Real-facing handoff is last. A $\mathsf{Real}^{\uparrow}$ or $\mathsf{CauchyRealField}^{\uparrow}$ consumer can read the seal S only after an operation route has produced its regular readback, and the rows H, C, P, N preserve that same finite route. No coordinate remains from which a quotient Cauchy class, host operation, selected limit, countable choice, trichotomy, or ambient completeness principle could be exported. □

Theorem 3708.5 (CauchySequenceAlgebra route non-escape)**.** *Let* $G = (X, Y, W, D, A, M, S, H, C, P, N)$ *be an accepted* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *carrier. Any algebraic fact exported from* G *is a rowwise consequence of the displayed regular source rows, synchronized windows, dyadic operation budgets, operation routes, Real seal, transport, replay, provenance, and local naming rows. It is not a quotient Cauchy class, host real addition, host real multiplication, selected limit, countable-choice datum, trichotomy datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Inspect the carrier rows. The data-bearing coordinates are exactly X, Y, W, D, A, M, S, and the structural coordinates are exactly H, C, P, N.

An exported addition fact must use A after the source, window, and dyadic-budget rows have been displayed. An exported product fact must use M after the same upstream rows have been displayed. A Real-facing algebra fact must then pass through the seal row S.

The structural rows preserve only that route. Componentwise hsame transport and displayed Cont replay cannot create a host operation or a quotient relation, while provenance and local naming record only the finite packet. Thus every exported algebraic fact is confined to the listed rows, and each excluded principle has no entry point. □

Theorem 3708.6 (CauchySequenceAlgebra remainder-estimate handoff)**.** *Let*

$$G = (X, Y, W, D, A, M, S, H, C, P, N)$$

be an accepted CauchySequenceAlgebra$^{\uparrow}$ *carrier. Any downstream* CauchyRemainderEstimate$^{\uparrow}$ *read for an addition or product result factors first through the same finite binary-algebra route*

$$X, Y \longmapsto W \longmapsto D \longmapsto A, M \longmapsto S \longmapsto H, C, P, N$$

and then through the finite tail-estimate rows of chapter 3991. The handoff exports the displayed source rows, synchronized windows, dyadic operation budgets, selected addition or product route, terminal Real seal, transport, replay, provenance, local naming, finite tail threshold, admitted tail windows, regular readback, and tail-diameter boundary rows. It exports no quotient Cauchy class, host real operation, selected limit, countable-choice datum, trichotomy datum, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier G. By Definition 3708.2, every binary algebra read exposes only the source rows X, Y, synchronized window W, dyadic budget D, operation rows A, M, Real seal S, and structural rows H, C, P, N.

Apply Definition 3708.4 to the downstream addition or product request. The result route must pass through A for addition or M for product, and in either case it uses the same upstream X, Y, W, D rows before reaching the seal S.

Now apply Definition 3708.5. No part of that binary-algebra route can introduce a quotient Cauchy class, host real operation, selected limit, countable-choice row, trichotomy row, or ambient completeness row.

The remainder-estimate consumer therefore begins from the regular readback already displayed by the operation route and then uses the sibling surface chapter 3991. That surface supplies the finite threshold, admitted tail windows, regular readback, tail-diameter boundary, transport, replay, provenance, and local naming rows for CauchyRemainderEstimate$^{\uparrow}$. Composing these two finite routes gives the stated handoff and leaves every excluded principle without a carrier coordinate. □

Closure status for NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 3708.1, Definition 3708.2, Definition 3708.3, Definition 3708.4, Definition 3708.5, and Definition 3708.6 seed the finite binary algebra route over RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAddition$^{\uparrow}$, RegularCauchyProduct$^{\uparrow}$, CauchyRemainderEstimate$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequenceAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient Cauchy classes, host real operations, selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, source-window synchronization, dyadic-budget exactness, operation-route separation, Real seal handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CauchySequenceAlgebraUp.

Constructive story: A CauchySequenceAlgebra$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, synchronized StreamName$^{\uparrow}$ windows, one DyadicRatCore$^{\uparrow}$ operation-budget ledger, RegularCauchyAddition$^{\uparrow}$ and RegularCauchyProduct$^{\uparrow}$ route rows, a Real$^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3709

A Concrete Naming Certificate for $\mathsf{CauchySequenceProductStability}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySequenceProductStability}^{\uparrow}$ packet records the finite stability route by which two regular rational Cauchy source rows keep their product inside the Real and Completion interface. It gathers the regular product route of chapter 461, the product criterion of chapter 1586, the synchronized window discipline of chapter 126, the dyadic budget rows of chapter 291, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet is a finite NameCert carrier for product stability, not host multiplication of completed reals, quotient equality of streams, selected limits, countable choice, trichotomy, or ambient completeness.

Definition 3709.1 (Cauchy sequence product stability carrier)**.** *A* $\mathsf{CauchySequenceProductStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular Cauchy schedules supply the two input sequences.*
2. *W_A and W_B are synchronized* $\mathsf{StreamName}^{\uparrow}$ *window rows for the same product request, precision demand, and observation index.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, recording input tolerances, finite magnitude bounds, cross-error terms, and the output tolerance assigned to the product window.*
4. *P is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route row, read only after A, B, W_A, W_B, D have been displayed.*
5. *K is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *acceptance row for the displayed product route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the product route under the dyadic budget and criterion row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after the regular-rational product readback R has been displayed.*
8. *H records componentwise* hsame *transport for $A, B, W_A, W_B, D, P, K, R, E, C, L, N$ and the local product-stability packet shape.*

9. *C records displayed* Cont *replay from the two source rows through the synchronized windows, dyadic budget, product route, criterion row, readback row, and real seal.*
10. *L is the* Pkg *provenance over* $\mathsf{CauchySequenceProductStability}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, W_A, W_B, D, P, K, R, E, H, C, L, N$. It has no coordinate for host multiplication of completed reals, quotient stream equality, a selected product limit, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchySequenceProductStabilityUp.CauchySequenceProductStabilityTasteGate`
`single carrier alignment`

Theorem 3709.2 (Cauchy sequence product stability NameCert obligations)**.** *For an accepted* $\mathsf{CauchySequenceProductStability}^{\uparrow}$ *carrier $S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$ *obligations are exactly carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic product-budget exactness, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *route, the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *acceptance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport through H, displayed continuation replay through C, provenance through L, and local naming through N.*

Proof. Carrier admission is projection from Definition 3709.1. The tuple displays the two source rows, synchronized windows, dyadic product budget, product route, criterion row, regular-rational readback, real seal, and structural rows.

The source and window obligations are read from A, B, W_A, W_B. These rows expose the two regular schedules and align them before the product route is available.

The budget and route obligations are read from D, P, K. The dyadic row records the finite tolerance and magnitude data, the regular product route consumes that data, and the criterion row accepts the displayed product route for downstream readback.

The rows R, E, H, C, L, N then provide the regular-rational boundary, real-seal boundary, transport, replay, provenance, and local name. Since no other coordinate occurs in the carrier, the listed rows exhaust the certificate surface. □

Theorem 3709.3 (Cauchy sequence product budget stability)**.** *Let $S = (A, B, W_A, W_B, D, P, K, R, E, H, C, L, N)$ be an accepted* $\mathsf{CauchySequenceProductStability}^{\uparrow}$ *carrier. Replacing the two source rows, synchronized window rows, dyadic product budget, product route, criterion row, readback row, and real seal by displayed componentwise* hsame *rows, and replaying the replacement through C, preserves the admissible product-stability route*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto P \longmapsto K \longmapsto R \longmapsto E.$$

Thus every Real and Completion consumer of the product reads the same finite stability packet and cannot introduce host multiplication, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only route from the two source sequences to the terminal real seal is the displayed chain through the synchronized windows, dyadic budget, regular product route, criterion row, regular-rational readback, and real seal.

The replacement hypothesis is internal to the structural rows. The row H gives componentwise hsame transport for every displayed product-stability coordinate, and C replays the continuation through the transported coordinates in the same order.

Therefore the transported packet still reaches D, then P, then K, then R, then E. The provenance row L and local name N preserve the same finite packet and do not add a second product operation.

Since the carrier has no coordinate for host multiplication, quotient streams, selected limits, choice, trichotomy, or ambient completeness, the budget-stable route is exactly the displayed BEDC product-stability route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceProductStability}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore product budget, a RegularCauchyProduct route, a CauchyProductCriterion acceptance row, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3709.1, Definition 3709.2, and Definition 3709.3 seed the finite product-stability packet for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceProductStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient stream equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window admission, dyadic product-budget exactness, RegularCauchyProduct routing, CauchyProductCriterion acceptance, RegSeqRat readback, Real seal handoff, transport stability, continuation replay, and provenance non-escape for BEDC.Derived.CauchySequenceProductStabilityUp.

Chapter 3710

A Concrete Naming Certificate for $\mathsf{BishopCauchyProductBound}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCauchyProductBound}^{\uparrow}$ packet records the finite boundedness estimate used before a Cauchy product is admitted as a regular real product. It sits between the product surface of chapter 464, the regular product route of chapter 461, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet supplies a finite majorant and cross-error budget for product windows; it is not host multiplication of completed reals, quotient stream equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3710.1 (Bishop Cauchy product bound carrier)**.** *A* $\mathsf{BishopCauchyProductBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, C, W_A, W_C, M_A, M_C, D, P, R, E, H, K, L, N)$$

with the following displayed rows:

1. *A and C are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular Cauchy observations will be multiplied.*
2. *W_A and W_C are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite windows for the requested product precision.*
3. *M_A and M_C are finite magnitude-majorant rows for the two source windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the output tolerance, the two source tolerances, and the cross-error terms bounded by M_A and M_C.*
5. *P is the* $\mathsf{CauchyProduct}^{\uparrow}$ *product-window row exposed only after the source windows and dyadic budget are displayed.*
6. *R is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *handoff row that repacks the product-window estimate as a regular Cauchy readback.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after R has been displayed.*
8. *H records componentwise* hsame *transport for all product-bound rows.*
9. *K records displayed* Cont *replay from the source windows through the bound ledger, product row, regular handoff, and real seal.*

10. *L is the* Pkg *provenance over* $\mathsf{BishopCauchyProductBound}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyProductBound}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, C, W_A, W_C, M_A, M_C, D, P, R, E, H, K, L, N$. It has no coordinate for host real multiplication, quotient stream equality, a selected product limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.BishopCauchyProductBoundUp`

Theorem 3710.2 (Bishop Cauchy product bound NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyProductBound}^{\uparrow}$ *carrier $B = (A, C, W_A, W_C, M_A, M_C, D, P, R, E, H, K, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyProductBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows, two finite magnitude-majorant rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-bound ledger, the* $\mathsf{CauchyProduct}^{\uparrow}$ *window row, the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *handoff, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the accepted carrier. The source rows A, C and windows W_A, W_C expose the only finite observations from which a product estimate can be read.

The rows M_A, M_C, D then bind the boundedness estimate. The two majorants control the source magnitudes, while the dyadic ledger records the source tolerances and the resulting cross-error budget before any product row is exposed.

The product-facing rows P, R, E are downstream of that ledger. They route the finite product estimate into the regular Cauchy product handoff and then into the real seal.

Finally H, K, L, N transport, replay, source, and name exactly these rows. Since the carrier contains no host multiplication, quotient stream, selected-limit, choice, trichotomy, or ambient-completeness coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3710.3 (Bishop Cauchy product bound regular handoff)**.** *Every accepted* $\mathsf{BishopCauchyProductBound}^{\uparrow}$ *carrier hands a bounded Cauchy-product estimate to* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *only through the displayed route*

$$A, C \longmapsto W_A, W_C \longmapsto M_A, M_C \longmapsto D \longmapsto P \longmapsto R \longmapsto E.$$

Thus a Real multiplication or Completion consumer must first inspect the two regular source rows, the synchronized finite windows, the magnitude bounds, the dyadic cross-error ledger, and the Cauchy-product window before it can read the regular-product handoff.

Proof. Start with the carrier projection. The only source data are A, C, W_A, W_C, so a product consumer begins with regular rational observations and finite stream windows rather than with completed real representatives.

The boundedness estimate is forced through M_A, M_C, D. These rows supply the finite majorants and the dyadic error ledger that make the product-window estimate admissible.

The downstream handoff is then P, R, E. The $\mathsf{CauchyProduct}^{\uparrow}$ row records the product window, the $\mathsf{RegularCauchyProduct}^{\uparrow}$ row repacks it as a regular readback, and the $\mathsf{Real}^{\uparrow}$ row seals the endpoint.

The structural rows preserve the same order under transport, replay, provenance, and naming. No other carrier coordinate can supply a host product, quotient stream equality, selected limit, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyProductBound}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyProductBound}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, two finite magnitude bounds, a DyadicRatCore product-bound ledger, a CauchyProduct window row, a RegularCauchyProduct handoff row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyProductBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3710.1, Definition 3710.2, and Definition 3710.3 seed the finite bounded Cauchy-product estimate route for Real multiplication consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyProductBound}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.BishopCauchyProductBoundUp`

Bridge status: none

Not claimed: This chapter does not close host multiplication of completed reals, quotient stream equality, selected product limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, magnitude-bound extraction, dyadic product-budget exactness, CauchyProduct routing, RegularCauchyProduct handoff, Real seal nonescape, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BishopCauchyProductBoundUp.

Chapter 3711

A Concrete Naming Certificate for $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$

Following Oresme and Bishop–Bridges constructive analysis, the $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ packet names the dyadic-block lower-bound route for the harmonic series as finite BEDC data. It sits beside the divergence refusal surface of chapter 2047, the positive condensation route of chapter 3345, the rational comparison source of chapter 37, the dyadic arithmetic ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback row of chapter 301, and the terminal real-name seal of chapter 38. The carrier records only displayed finite partial sums, dyadic block schedules, rational comparisons, tolerance ledgers, readback windows, transport, replay, provenance, and local naming.

Definition 3711.1 (OresmeHarmonicDivergence carrier)**.** *An* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, B, Q, D, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *S is the harmonic partial-sum row, exposing only finite sums of the displayed terms* $1, 1/2, 1/3, \ldots$ *through inspected windows.*
2. *B is the dyadic block schedule, listing the finite intervals* $[2^k, 2^{k+1})$ *requested by the packet and the endpoints used by each partial-sum comparison.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row, carrying the rational lower-bound entries used inside each displayed block.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording powers of two, reciprocal denominators, and the finite comparison budget used to show that each inspected block contributes at least* $1/2$.
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the harmonic terms, block endpoints, rational comparisons, and partial-sum reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed partial sums and block lower bounds.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* S, B, Q, D, W, R *have been displayed.*
8. *T records componentwise* hsame *transport for the partial-sum row, block schedule, rational comparison row, dyadic ledger, stream windows, readback row, Real seal, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a block request through the harmonic partial sums, dyadic schedule, rational comparison row, dyadic ledger, stream windows, readback row, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$, $\mathsf{SeriesDivergenceTest}^{\uparrow}$, $\mathsf{CauchyCondensationRealSeries}$ $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, Q, D, W, R, E, T, C, P, N$. It has no coordinate for a host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OresmeHarmonicDivergenceUp.OresmeHarmonicDivergenceTasteGate single carrier alignment`

Theorem 3711.2 (OresmeHarmonicDivergence NameCert obligations)**.** *For every accepted* OresmeHarmonicDiverg *carrier $H = (S, B, Q, D, W, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed harmonic partial-sum row, dyadic block schedule, rational comparison row, dyadic tolerance ledger, stream-window row, readback row, Real seal, transport, replay, provenance, and local naming rows;*

2. *block schedule exactness, requiring B to expose precisely the inspected finite intervals $[2^k, 2^{k+1})$ and their partial-sum endpoints before any lower-bound comparison is read;*

3. *rational lower-bound visibility, requiring Q, D, W, R to display the finite comparison that every term in the inspected block is at least $1/2^{k+1}$ and that the whole block contributes at least $1/2$;*

4. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring E to remain downstream of the displayed finite block route and to export no host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, or ambient completeness principle;*

5. *ledger exhaustion, requiring T, C, P, N to transport, replay, source, and name the same finite route without adding unlisted analytic or choice data.*

No Oresme harmonic-divergence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.OresmeHarmonicDivergenceUp.OresmeHarmonicDivergenceNameCertObligations`

Proof. Project the carrier of Definition 3711.1. The accepted packet exposes exactly the harmonic partial-sum row, the dyadic block schedule, the rational comparison row, the dyadic tolerance ledger, stream windows, readback, Real seal, and structural rows.

The block schedule is finite. For each inspected k, the row B gives the interval $[2^k, 2^{k+1})$ and the partial-sum endpoints that the consumer may read from S.

The lower-bound comparison is carried by Q, D, W, R. The schedule W selects the finitely many terms in the displayed block, Q supplies the rational comparison against $1/2^{k+1}$, D records the dyadic denominator and tolerance budget, and R reads back the block contribution.

The Real-facing route is downstream of those rows. The seal E can only be reached after the finite block comparison has passed through the displayed readback, while T, C, P, N preserve and name the same route.

Since a host infinite sum, quotient series equality, countable-choice schedule, selected endpoint, and ambient completeness are not coordinates of H, the listed rows exhaust the local NameCert surface. □

Theorem 3711.3 (Oresme dyadic block lower bound)**.** *Every accepted* $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ *packet exports the finite lower-bound route for each displayed block* $[2^k, 2^{k+1})$ *through*

$$S, B \longmapsto Q, D, W, R \longmapsto E, T, C, P, N.$$

Thus the packet records the Oresme block fact that each inspected dyadic block contributes at least $1/2$ *to the harmonic partial-sum row, while exporting no host infinite sum, quotient series equality, countable choice, selected endpoint, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted packet $H = (S, B, Q, D, W, R, E, T, C, P, N)$. A consumer requesting the block lower bound must first read the partial-sum row S together with the dyadic block schedule B.

For a displayed block $[2^k, 2^{k+1})$, the window row W exposes exactly the finitely many harmonic terms in that interval. The comparison row Q and dyadic ledger D show that every such term is bounded below by $1/2^{k+1}$.

There are 2^k displayed terms in the block, so the same finite ledger records the rational lower bound $2^k/2^{k+1} = 1/2$. The readback row R attaches this lower bound to the corresponding block contribution of the partial-sum row.

The seal row E and the structural rows T, C, P, N only carry this finite route forward. Because the carrier has no field for host infinite sums, quotient series equality, countable choice, selected endpoints, or ambient completeness, the exported statement is exactly the finite dyadic-block lower-bound route. □

Falsifiable prediction: Every accepted $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ block read factors through the harmonic partial-sum row, dyadic block schedule, RatUp comparison row, DyadicRatCore tolerance ledger, StreamName window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: series-divergence-test, cauchy-condensation-real-series, rat, dyadicratcore, streamname, regseqrat, real: $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the harmonic partial-sum row, dyadic block schedule, RatUp comparison row, DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3711.1, Definition 3711.2, and Definition 3711.3 seed the finite Oresme dyadic-block lower-bound carrier for harmonic partial sums, rational comparisons, dyadic ledgers, stream windows, readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OresmeHarmonicDivergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient series equality, countable choice, selected endpoints, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, block schedule exactness, rational lower-bound visibility, RealUp seal nonescape, ledger exhaustion, transport, replay, provenance, and local naming for BEDC.Derived.OresmeHarmonicDiverge

Constructive story: An $\mathsf{OresmeHarmonicDivergence}^{\uparrow}$ packet is generated from a harmonic partial-sum row, a dyadic block schedule, RatUp comparison rows, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3712

A Concrete Naming Certificate for $\mathsf{Subdifferential}^{\uparrow}$

Following Rockafellar and Boyd–Vandenberghe, the $\mathsf{Subdifferential}^{\uparrow}$ packet records the finite supporting-hyperplane data that makes a convex subgradient usable inside BEDC without importing a separation theorem. It sits beside the convex-set carrier of chapter 208, the conjugacy rows of chapter 3563, the duality handoff of chapter 321, the order boundary of chapter 2297, and the stationarity surface of chapter 300. The chapter names the local subgradient certificate only; it does not assert Hahn–Banach separation, quotient function-space duality, selected suprema, or choice-selected subgradients.

3712.1 Subdifferential Carrier

Definition 3712.1 (Subdifferential carrier)**.** *A* $\mathsf{Subdifferential}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, x, \xi, E, L, G, K, H, C, P, N)$$

with the following visible rows:

1. *F is the displayed convex functional row, including its finite convex-domain, convex-combination, value-comparison, and lower-support ledgers;*
2. *x is the displayed base point row inside the carrier domain of F;*
3. *ξ is the displayed dual covector row, read only through the finite pairing tests carried by the packet;*
4. *E is the pairing-evaluation ledger recording the finite values $\langle \xi, y-x \rangle$ that are actually inspected;*
5. *L is the supporting-inequality ledger asserting, for each displayed comparison point y, the row $F(y) \geq F(x) + \langle \xi, y - x \rangle$;*
6. *G is the epigraph row that reads the same support inequality as a displayed lower supporting face of the epigraph carrier;*
7. *K is the cone-boundary row connecting the finite support inequality to the* $\mathsf{PositiveCone}^{\uparrow}$ *order boundary and to the stationarity rows later consumed by* $\mathsf{KKT}^{\uparrow}$*;*
8. *H stores componentwise* hsame *transport for the functional, base point, covector, pairing ledger, support ledger, epigraph row, cone boundary, continuation row, provenance row, and local naming row;*

9. *C stores displayed* Cont *replay from a subgradient request through the convex functional, pairing, support, epigraph, cone-boundary, and downstream handoff rows;*
10. *P is the* Pkg *provenance over* $\mathsf{Subdifferential}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{Subdifferential}^{\uparrow}}$ *row.*

The carrier contains no ambient locally convex dual, no quotient function space, no selected separating hyperplane, no selected supremum in a conjugate formula, and no rule choosing a subgradient when the displayed support ledger is absent.

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialTasteGate single carrier alignment`

3712.2 Subdifferential Certificate

Theorem 3712.2 (Subdifferential NameCert obligations)**.** *For every accepted* $\mathsf{Subdifferential}^{\uparrow}$ *carrier* $S = (F, x, \xi, E, L, G, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Subdifferential}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the convex functional row, base point row, dual covector row, finite pairing-evaluation ledger, supporting-inequality ledger, epigraph row, cone boundary, transport, replay, provenance, and local naming row;*
2. *support-inequality exactness, requiring every exported subgradient read to pass through the displayed row $F(y) \geq F(x) + \langle \xi, y - x \rangle$ for the listed comparison point;*
3. *epigraph compatibility, requiring the support ledger and epigraph row to name the same lower supporting face and the same base point;*
4. *Fenchel handoff, requiring any conjugacy or duality consumer to pass through the displayed* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *rows before reading a dual certificate;*
5. *KKT and normal-cone consumer boundary, requiring stationarity consumers to read the cone-boundary row and refusing any normal-cone claim not carried by the finite support ledger;*
6. *non-escape from Hahn–Banach, quotient function spaces, selected suprema, and choice-selected subgradients.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier namecert obligations`

Proof. Project the accepted carrier. The rows F, x, ξ, E, L, G, K are the entire analytic surface, and the rows H, C, P, N are the entire structural surface.

Support exactness is a ledger condition. A consumer can call ξ a subgradient only at a displayed comparison point whose support inequality is present in L and whose pairing value is present in E.

Epigraph compatibility and the Fenchel handoff are then read through G and the provenance row P. The same lower support row is available to epigraph, conjugacy, and duality consumers, but no hidden supremum or separating functional is added by that handoff.

Finally the stationarity boundary passes through K. Since no coordinate of S names Hahn–Banach separation, a quotient dual space, a selected supremum, or a choice functional for subgradients, those principles cannot be exported by the local certificate. □

Theorem 3712.3 (Subdifferential support inequality exactness)**.** *For every accepted* Subdifferential$^{\uparrow}$ *carrier* $S = (F, x, \xi, E, L, G, K, H, C, P, N)$*, a public subgradient read is exact only when its displayed comparison point is named in the finite support ledger* L *and its pairing value is named in the finite pairing ledger* E*. The row* L *must expose the inequality* $F(y) \geq F(x) + \langle \xi, y - x \rangle$ *for that same* y*, and no consumer may replace this row by an ambient separation principle or by a chosen supporting functional.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier support inequality exactness`

Proof. Project the carrier coordinates E and L. The pairing ledger E is the only row that evaluates $\langle \xi, y - x \rangle$, and the support ledger L is the only row that compares this value with $F(y)$ and $F(x)$.

Since the carrier contains no Hahn–Banach coordinate and no selection rule for subgradients, a consumer can export only the support inequalities that are already displayed in L. Exactness is therefore the rowwise agreement of the finite comparison point, the pairing value, and the listed inequality. □

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialSupportInequalityExactness`

Theorem 3712.4 (Subdifferential epigraph and Fenchel handoff)**.** *Every* ConvexConjugate$^{\uparrow}$ *or* FenchelDuality$^{\uparrow}$ *read of an accepted* Subdifferential$^{\uparrow}$ *carrier factors through the displayed epigraph row* G*, the support ledger* L*, the pairing ledger* E*, the componentwise transport row* H*, and the provenance row* P*. The handoff may expose a conjugacy or duality consumer, but it does not add selected suprema, quotient dual spaces, or an unlisted separating hyperplane.*

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier epigraph fenchel handoff`

Proof. Read the epigraph coordinate G together with the ledgers E and L. These rows identify the lower supporting face that the conjugacy and duality consumers may inspect.

The provenance row P names the ConvexConjugate$^{\uparrow}$ and FenchelDuality$^{\uparrow}$ dependencies, while H carries the same finite rows across permitted comparisons. Because no coordinate stores a selected supremum or quotient dual object, the Fenchel-facing handoff is exactly the displayed epigraph support route. □

Theorem 3712.5 (Subdifferential KKT boundary)**.** *Every* PositiveCone$^{\uparrow}$ *or* KKT$^{\uparrow}$ *stationarity read of an accepted* Subdifferential$^{\uparrow}$ *carrier must pass through the cone-boundary row* K*, the support ledger* L*, the pairing ledger* E*, transport* H*, replay* C*, provenance* P*, and the local naming row* N*. The boundary records only finite stationarity data and refuses Hahn–Banach separation, normal-cone completion, and choice-selected subgradients not present in the support ledger.*

Proof. Start from the cone-boundary coordinate K. It is the only carrier row that connects the support inequality to the positive-cone and KKT stationarity surface.

The ledgers E and L supply the finite pairing and support data used by that boundary, while H, C, P, N transport, replay, source, and name the same packet. With no coordinate for Hahn–Banach separation or an ambient normal-cone completion, the stationarity consumer receives exactly the finite boundary listed above. □

Lean4 ✓ `BEDC.Derived.SubdifferentialUp.SubdifferentialCarrier kkt boundary`

Closure status for NameCert$_{\text{Subdifferential}^{\uparrow}}$**.**

Constructive story: A Subdifferential$^{\uparrow}$ packet is generated from a convex functional row, base point row, dual covector row, finite pairing-evaluation ledger, supporting-inequality ledger, epigraph row, cone-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{Subdifferential}\uparrow}$, scopedClosed)

Scope closed: Definition 3712.1, Definition 3712.2, Definition 3712.3, Definition 3712.4, and Definition 3712.5 bind the finite subgradient obligation surface between convex sets, convex conjugacy, Fenchel duality, positive cones, and KKT stationarity.

Formal verification: FormalStatus(NameCert$_{\mathsf{Subdifferential}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, quotient function-space duality, selected suprema, choice-selected subgradients, Lean verification, or bridge checking.

Upgrade path: Public closure requires a public subgradient consumer theorem and later Lean verification for BEDC.Derived.SubdifferentialUp.

Chapter 3713

A Concrete Naming Certificate for $\mathsf{ProximalMap}^{\uparrow}$

Following Rockafellar's convex-analysis treatment of subdifferentials and resolvents, and Boyd–Vandenberghe's finite optimization presentation of proximal operators, the $\mathsf{ProximalMap}^{\uparrow}$ packet records the displayed proximal minimization surface as BEDC NameCert data. It sits after the convex-set carrier of chapter 208, the conjugacy rows of chapter 3563, the Fenchel-duality handoff of chapter 321, the positive-cone order boundary of chapter 2297, the stationarity ledger of chapter 300, the real seal of chapter 38, and the vector-space source of chapter 47. The packet gives subdifferential and monotone-operator consumers a finite resolvent-facing route without asserting a selected global minimizer, Hilbert projection theorem, host lower semicontinuity, or ambient completeness.

3713.1 ProximalMap Carrier

Definition 3713.1 (ProximalMap carrier)**.** *A* $\mathsf{ProximalMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, F, u, z, R, D, K, G, O, H, C, P, N)$$

with the following visible rows:

1. *X is the displayed vector-space source row, read through* $\mathsf{VecSpace}^{\uparrow}$ *and the real scalar seal of* $\mathsf{Real}^{\uparrow}$*;*
2. *F is the convex functional and domain row, including the visible* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{ConvexConjugate}^{\uparrow}$ *surfaces used by the request;*
3. *u is the input point row in the carried vector-space source;*
4. *z is the displayed candidate output row;*
5. *R is the real penalty and distance-evaluation row used for the displayed expression $F(y)+\frac{1}{2\lambda}\|y-u\|^2$ at the finitely inspected comparison points;*
6. *D is the finite dominance ledger recording that the candidate row is no worse than each displayed comparison row;*
7. *K is the KKT stationarity row tying the candidate, penalty, cone boundary, and finite residual rows to the* $\mathsf{KKT}^{\uparrow}$ *packet;*

8. *G is the Fenchel and convex-conjugate handoff row, read only after F, R, D, K have been displayed;*
9. *O is the convex-resolvent consumer row for later subdifferential or monotone-operator packets;*
10. *H records componentwise* hsame *transport for the vector-space source, convex functional, input, candidate, real penalty, dominance ledger, KKT stationarity, Fenchel handoff, resolvent consumer, replay, provenance, and local naming rows;*
11. *C records displayed* Cont *replay from a proximal request through convex admission, real penalty evaluation, stationarity, duality, dominance, and resolvent-facing readback;*
12. *P is the* Pkg *provenance over* $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient Hilbert projection theorem, a selected minimizer outside the displayed candidate row, host lower semicontinuity, global coercivity, quotient vector equality, or an existence theorem for every input.

Lean4 (definition) `BEDC.Derived.ProximalMapUp.ProximalMapUp`
Lean4 ✓ `BEDC.Derived.ProximalMapUp.ProximalMapTasteGate single carrier alignment`

3713.2 ProximalMap Certificate

Theorem 3713.2 (ProximalMap NameCert obligations)**.** *For every accepted* $\mathsf{ProximalMap}^{\uparrow}$ *carrier* $Q = (X, F, u, z, R, D, K, G, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the vector-space source, convex functional, input point, candidate output, real penalty row, dominance ledger, KKT row, Fenchel handoff, resolvent consumer row, transport, replay, provenance, and local naming;*
2. *penalty evaluation exactness, requiring every proximal comparison to read the same* $\mathsf{Real}^{\uparrow}$ *penalty row R and the same* $\mathsf{VecSpace}^{\uparrow}$ *source row X before any dominance claim is exported;*
3. *convex-domain compatibility, requiring the candidate and comparison points to be admitted through the* $\mathsf{ConvexSet}^{\uparrow}$ *row carried by F;*
4. *stationarity compatibility, requiring the displayed residual, cone boundary, and multiplier data to pass through the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{PositiveCone}^{\uparrow}$ *rows before any optimality-facing read is public;*
5. *Fenchel and conjugacy handoff, requiring dual or subgradient consumers to pass through* $\mathsf{ConvexConjugate}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *only after the dominance and stationarity rows have been displayed;*
6. *non-escape from global minimizer selection, host projection theorems, quotient vector equality, lower-semicontinuity assumptions, and ambient completeness.*

Proof. Project the accepted carrier. The rows $X, F, u, z, R, D, K, G, O$ are the complete proximal surface, while H, C, P, N are the structural route that transports, replays, sources, and names the same packet.

Penalty exactness is a finite ledger condition. A comparison with a point y can be read only when the relevant vector-space rows in X, convex-domain rows in F, and real penalty value in R are all displayed, and the dominance row in D names that comparison.

Stationarity and order compatibility are then read through K and the provenance row P. The KKT and positive-cone rows may inspect only the displayed residual, cone, multiplier, real, and vector entries already present in the packet.

The Fenchel and conjugacy handoff is downstream of the same displayed data. Since no coordinate of Q stores a selected global minimizer, a Hilbert projection theorem, quotient vector equality, lower-semicontinuity evidence, or ambient completeness, those principles cannot be exported by the local certificate. □

Theorem 3713.3 (ProximalMap convex-resolvent handoff)**.** *Every subdifferential, monotone-operator, or splitting-method consumer of an accepted* $\mathsf{ProximalMap}^{\uparrow}$ *packet must factor through the finite route*

$$X, F, u, z, R, D, K, G, O \longmapsto H, C, P, N.$$

The consumer may read the candidate z as a resolvent-facing output only after the convex-domain row, real penalty row, dominance ledger, KKT stationarity row, positive-cone boundary, and Fenchel-duality handoff have all been displayed. It receives no selected minimizer for unlisted inputs, no maximal-monotone theorem, no quotient vector equality, no host projection semantics, and no ambient completeness principle.

Proof. Begin with the candidate row. By Definition 3713.1, the row z is only one displayed output candidate in the finite packet, and it is tied to the input row u and vector-space source row X.

The dominance ledger D and penalty row R supply the proximal comparison surface. They are read through the same convex-domain row F and the same real scalar seal, so a consumer cannot replace the displayed comparisons by an external minimization predicate.

Next read the stationarity and duality boundary. The row K routes optimality-facing material through the KKT and positive-cone packets, while G routes conjugacy-facing material through the convex-conjugate and Fenchel-duality packets. The resolvent consumer row O is exposed only after those rows are present.

Transport, continuation replay, package provenance, and local naming preserve exactly this route through H, C, P, N. Since the carrier lacks coordinates for unlisted input selection, maximal-monotone closure, quotient vector equality, host projection semantics, or completeness, the handoff stops at the finite convex-resolvent boundary. □

Falsifiable prediction: Every accepted $\mathsf{ProximalMap}^{\uparrow}$ consumer read factors through a VecSpaceUp source, ConvexSetUp and ConvexConjugateUp functional rows, one displayed input, one displayed candidate output, a RealUp penalty row, a finite dominance ledger, KKTUp stationarity rows, PositiveConeUp order rows, FenchelDualityUp handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, convex-conjugate, fenchelduality, positive-cone, kkt, real, vecspace: $\mathsf{ProximalMap}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an input row, candidate output row, real penalty ledger, dominance comparisons, stationarity boundary, Fenchel handoff, and resolvent-facing consumer row in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProximalMap}^{\uparrow}$ packet is generated from a vector-space source, convex functional and domain rows, one input point, one candidate output, RealUp penalty evaluations, finite dominance comparisons, KKT stationarity, PositiveCone order, FenchelDuality and ConvexConjugate handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProximalMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3713.1, Definition 3713.2, and Definition 3713.3 seed the finite proximal-resolvent certificate surface over convex, real, vector-space, KKT, cone, and Fenchel rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProximalMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a global proximal existence theorem, selected minimizers for all inputs, Hilbert projection semantics, maximal-monotone closure, quotient vector equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, penalty evaluation exactness, convex-domain compatibility, stationarity compatibility, Fenchel handoff, resolvent consumer boundary, transport, replay, provenance, and local naming for BEDC.Derived.ProximalMapUp.

Chapter 3714

A Concrete Naming Certificate for TypedNormalFormDecider$^{\uparrow}$

The TypedNormalFormDecider$^{\uparrow}$ packet realizes the normal-equality section of chapter 9379 as a finite BEDC certificate tying `inferTypeCtx` to bounded normal-form comparison. It sits beside the bounded normal-equality checker of chapter 752, the decidability witness of chapter 751, the closure-preservation audit trace of chapter 795, and the normalization audit map of chapter 9383. The packet names the typed checker and bounded comparison route; it refuses to turn that route into unrestricted beta-conversion decidability.

Falsifiable prediction: Every accepted TypedNormalFormDecider$^{\uparrow}$ packet displays `inferTypeCtx` rows, typed input rows, bounded normalizer output rows, finished-normalization witnesses when supplied, bounded normal-form comparison rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Independence witness: bounded-normal-equality-checker, metacicdecidabilitywitness, closure-preservation audit, normalization audit: TypedNormalFormDecider$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries typed-checker admission, bounded normal-form comparison, finished-normalization side conditions, refusal, transport, replay, provenance, and local naming rows.

Definition 3714.1 (Typed normal-form decider carrier)**.** *A* TypedNormalFormDecider$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (I, L, R, T_L, T_R, F, N_L, N_R, E, W, Q, H, C, P, A)$$

with the following displayed rows:

1. *I is the `inferTypeCtx` checker row for the shared context and the represented input terms;*
2. *L and R are the two closed represented MetaCIC input term rows;*
3. *T_L and T_R are the optional inferred type rows produced by the checker, together with the displayed success or refusal status of each check;*
4. *F is the bounded fuel row used by the normalizer;*
5. *N_L and N_R are the bounded normal-form output rows obtained from L and R at fuel F;*
6. *E is the structural normal-form comparison row, read through `Term.eq` on N_L and N_R;*
7. *W stores the finished-normalization witnesses required before the comparison can be consumed as a typed normal-form equality result;*

8. *Q is the refusal row blocking unrestricted beta-conversion decidability, global normalization, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, and fuel selection;*
9. *H records componentwise* hsame *transport for checker, term, type, fuel, normal-form output, comparison, finished-witness, refusal, replay, provenance, and local naming rows;*
10. *C records displayed* Cont *replay from a typed normal-form decision request through $I, L, R, T_L, T_R, F, N_L, N_R, E,$*
11. *P is the* Pkg *provenance over* $\mathsf{TypedNormalFormDecider}^{\uparrow}$*, the MetaCIC decidability audit map,* $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICDecidableTypingFrontier}^{\uparrow}$*, closure-preservation audit rows, normalization audit rows,* BHist, hsame, Cont, *and* NameCert*;*
12. *A is the local* $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a fuel-erased beta-conversion oracle, no proof that all closed typed terms normalize, no hidden subject-reduction discharge, no confluence theorem, and no host quotient over terms.

Lean4 ✓ `BEDC.Derived.TypedNormalFormDeciderUp.TypedNormalFormDeciderTasteGate single carrier alignment`

Theorem 3714.2 (Typed normal-form decider NameCert obligations)**.** *For the carrier of Definition 3714.1, the local* $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$ *surface consists exactly of typing-checker admission through I, T_L, T_R, bounded-normal comparison through F, N_L, N_R, E, finished-normalization side-condition exposure through W, explicit refusal through Q, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through A.*

Proof. Project the carrier coordinates. The checker side is exhausted by I, L, R, T_L, T_R, so a consumer obtains typed input only through the displayed `inferTypeCtx` rows.

The normal-form side is exhausted by F, N_L, N_R, E, W. The comparison row E can be consumed only with the bounded fuel and finished-normalization witnesses displayed in the same packet.

The refusal row Q is part of the certificate surface. It prevents the bounded route from being reread as global normalization, unrestricted beta-conversion decidability, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, or fuel selection.

Finally H, C, P, A preserve and name the same finite route. Future Nontrivial and FieldFaithful TasteGate instances must witness that none of these rows can be collapsed away or replaced by an unlisted oracle. □

Theorem 3714.3 (Typed normal-form decider bounded comparison boundary)**.** *Every consumer of an accepted* $\mathsf{TypedNormalFormDecider}^{\uparrow}$ *packet reaches a typed normal-form comparison only through the finite route*

$$\mathit{inferTypeCtx},\ L, R, T_L, T_R, F, N_L, N_R, E, W, Q, H, C, P, A.$$

The route exports typing-checker data and bounded finished-normal comparison data; it exports no unrestricted closed beta-conversion decider, no global normalizer, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, and no host quotient equality for MetaCIC terms.

Proof. Read the checker rows first. A typed comparison request must pass through I and the displayed optional type rows T_L, T_R before it can inspect normal forms.

Next read the bounded normal-form rows. The fuel F, the outputs N_L, N_R, the structural comparison E, and the finished-normalization witnesses W form the complete comparison surface.

The refusal row Q is then read before structural replay. It records that the packet has no coordinate for a stronger conversion, normalization, subject-reduction, confluence, closedness, quotient, or fuel-selection theorem.

Thus transport, replay, provenance, and local naming in H, C, P, A preserve only the displayed bounded route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}$.

Constructive story: A $\mathsf{TypedNormalFormDecider}^{\uparrow}$ packet is generated from inferTypeCtx checker rows, closed input term rows, optional inferred type rows, bounded fuel, bounded normalizer outputs, structural normal-form comparison, finished-normalization witnesses, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3714.1, Definition 3714.2, and Definition 3714.3 seed the finite typed normal-form decider surface over typing-checker and bounded normal-form comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypedNormalFormDecider}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted beta-conversion decidability, global normalization, subject reduction, confluence, arbitrary closedness decidability, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typing-checker exactness, bounded-normal comparison exactness, finished-normalization side-condition exposure, refusal, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.TypedNormalFormDeciderUp.

Chapter 3715

A Concrete Naming Certificate for $\mathsf{ResolventIdentity}^{\uparrow}$

Following Dunford–Schwartz and Conway on the resolvent calculus for bounded operators, the $\mathsf{ResolventIdentity}^{\uparrow}$ packet records the finite BEDC surface on which the algebraic resolvent identity is read before any completed spectrum or host operator quotient is available. It sits at the operator-theory frontier beside the bounded-operator carrier of chapter 1644, the scalar operator-norm surface of chapter 3603, the resolvent-facing spectral-radius route of chapter 2352, and the compact-operator schedule boundary of chapter 1952.

Definition 3715.1 (Resolvent identity carrier)**.** *A* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the Banach source surface on which the operator rows are read.*
2. *T is the accepted* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the displayed operator action, linearity ledger, norm-bound endpoint, and transport data.*
3. *λ, μ are the two scalar-window rows whose separation witness is recorded before any inverse row is consumed.*
4. *A_λ, A_μ are the shifted-operator rows for $\lambda I - T$ and $\mu I - T$, each read through the same bounded-operator and operator-norm surface.*
5. *U_λ, U_μ are the displayed two-sided inverse-check rows for the two shifts.*
6. *I is the identity and composition ledger needed to compare $U_\lambda - U_\mu$ with $(\mu - \lambda)U_\lambda U_\mu$.*
7. *D is the domain and denominator ledger recording the scalar separation, multiplication order, and refusal of zero-denominator or detached inverse reads.*
8. *H records componentwise* hsame *transport for the operator, scalar, shifted-operator, inverse-check, identity, and denominator rows.*
9. *C records displayed* Cont *replay of the route from bounded operator and operator norm rows to the two resolvent checks and the identity ledger.*
10. *P is the* Pkg *provenance over* $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. N is the local $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed spectrum, a spectral measure, quotient operator equality, selected inverse outside the displayed two-sided checks, compact-operator eigenvalue lists, or countable choice.

Lean4 ✓ `BEDC.Derived.ResolventIdentityUp.ResolventIdentityTasteGate single carrier alignment`

Theorem 3715.2 (Resolvent identity NameCert obligations)**.** *For an accepted* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier* $R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-operator source admission through* T*, operator-norm scalar-bound admission for the two shifts, scalar-separation admission for* λ, μ*, two-sided inverse-check admission for* U_λ, U_μ*, identity and composition ledger exactness through* I*, denominator and refusal ledger exactness through* D*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Every downstream resolvent consumer must read the same rows before using the identity. It receives no completed spectrum, no spectral-radius limit theorem beyond the finite resolvent-facing boundary of chapter 2352, no compact-operator spectral decomposition, and no quotient equality of operators.

Proof. Project the carrier of Definition 3715.1. The rows $X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N$ are the whole accepted surface.

The bounded-operator and operator-norm rows expose the two shifted operators. The scalar rows record λ, μ and their separation before any inverse-check row is read, so no inverse can be selected outside the displayed denominator ledger.

The rows U_λ, U_μ supply only two-sided inverse checks for the displayed shifts. The identity ledger I then compares the two compositions and the scalar multiple on the same bounded-operator route.

Finally, H, C, P, N transport, replay, source, and name that finite route. Since the carrier has no coordinate for a completed spectrum, compact spectral decomposition, quotient equality, or choice, those objects cannot be part of the NameCert surface. □

Theorem 3715.3 (Resolvent identity two-sided law)**.** *Let* $R = (X, T, \lambda, \mu, A_\lambda, A_\mu, U_\lambda, U_\mu, I, D, H, C, P, N)$ *be an accepted* $\mathsf{ResolventIdentity}^{\uparrow}$ *carrier. If the two displayed inverse-check rows certify*

$$U_\lambda A_\lambda = A_\lambda U_\lambda = I_X \quad \textit{and} \quad U_\mu A_\mu = A_\mu U_\mu = I_X,$$

then every public resolvent read factors through the finite identity ledger

$$U_\lambda - U_\mu = (\mu - \lambda) U_\lambda U_\mu$$

with the same scalar-separation row, bounded-operator source row, operator-norm bound rows, and structural rows H, C, P, N*.*

The law exports only this two-sided displayed algebraic comparison. It does not export a completed resolvent set, a spectral theorem, quotient operator equality, selected approximate eigenvectors, compact-operator eigenvalue data, or countable choice.

Lean4 ✓ `BEDC.Derived.ResolventIdentityUp.ResolventIdentityCarrier two sided law`

Proof. Start with the two shifted-operator rows A_λ and A_μ. They are read from the same bounded operator T and scalar-window rows λ, μ, so their comparison is already inside the accepted packet.

Use the inverse-check rows U_λ and U_μ. The hypotheses certify both left and right inverse equations for the displayed shifts, and the denominator ledger D records the scalar separation needed to form $\mu - \lambda$.

Apply the identity and composition ledger I. It records the finite algebraic rearrangement of the two inverse equations and the composition $U_\lambda U_\mu$, producing the displayed equality without invoking a host quotient of operators.

The structural rows H, C, P, N preserve and name the same route for every public consumer. Since no carrier coordinate stores a completed spectrum, selected inverse, compact eigenvalue list, or choice schedule, the public law is exactly the finite two-sided resolvent identity. □

Falsifiable prediction: Every accepted $\mathsf{ResolventIdentity}^{\uparrow}$ consumer exposes the bounded operator row, both scalar-window rows, both shifted-operator rows, both two-sided inverse-check rows, the identity ledger, the denominator ledger, componentwise transport, replay, provenance, and local naming before reading the resolvent identity.

Independence witness: boundedlinearoperator, operatornorm, spectralradius, compactoperator: BoundedLinearOperatorUp supplies the action and norm-bound carrier but not the two scalar shifted-inverse ledger; OperatorNormUp supplies scalar norm bounds without the two-sided inverse checks; SpectralRadiusUp supplies a resolvent-facing boundary but not the algebraic two-shift identity packet; CompactOperatorUp supplies image-net schedules without the bounded-operator resolvent algebra.

Closure status for $\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}$.

Constructive story: A $\mathsf{ResolventIdentity}^{\uparrow}$ packet is generated from a Banach source surface, a bounded-operator row, two scalar-window rows, two shifted-operator rows, two two-sided inverse-check rows, identity and denominator ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3715.1, Definition 3715.2, Definition 3715.3, the falsifiable prediction, and the independence witness seed the finite two-shift resolvent identity route over BoundedLinearOperatorUp, OperatorNormUp, SpectralRadiusUp, CompactOperatorUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ResolventIdentity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a completed resolvent set, a spectral theorem, quotient operator equality, selected approximate eigenvectors, compact-operator eigenvalue data, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, shifted-operator stability, inverse-check exactness, denominator-ledger exactness, two-sided identity replay, classifier transport, and consumer non-escape for BEDC.Derived.ResolventIdentit

Chapter 3716

A Concrete Naming Certificate for BoundedIntervalCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the BoundedIntervalCauchySubsequence$^{\uparrow}$ packet records a finite-window substitute for the bounded-interval subsequence argument used by completion-facing real analysis. It sits after the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the interval projection route of chapter 3692, and the cofinal regular-Cauchy subsequence route of chapter 3515. The packet names the data needed by Real$^{\uparrow}$ and completion consumers to inspect a bounded interval source through explicit finite windows; it does not assert full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, or a choice-selected infinite subsequence.

Definition 3716.1 (Bounded interval Cauchy subsequence carrier)**.** *A* BoundedIntervalCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row, including endpoint* Real$^{\uparrow}$ *seals and the finite interval projection surface.*
2. *S is the source bounded sequence row, read only through displayed* StreamName$^{\uparrow}$ *observation windows.*
3. *W is the finite window ledger listing the inspected source positions, endpoint windows, and selected comparison windows.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger containing the mesh radius, endpoint comparisons, pairwise distance bounds, and late-window budgets used by the extraction route.*
5. *T is the monotone cofinal schedule row naming how each selected window is reached from an inspected finite source window.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *handoff row obtained from the displayed windows, dyadic ledger, and schedule.*
7. *E is the* RegularCauchySubsequenceStability$^{\uparrow}$ *row recording that the selected finite windows remain a regular Cauchy readback surface.*

8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the interval, window, dyadic, schedule, regular-readback, and subsequence-stability rows have been displayed.*
9. *H stores componentwise* hsame *transport for $I, S, W, D, T, Q, E, R, C, P, N$.*
10. *C stores displayed* Cont *replay of the extraction route*

$$I, S \longmapsto W, D, T \longmapsto Q, E \longmapsto R.$$

11. *P is the* Pkg *provenance over* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedIntervalProjection}^{\uparrow}$, $\mathsf{RegularCauchySubsequenceStability}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, T, Q, E, R, H, C, P, N$. It has no coordinate for an ambient compactness theorem, open-cover subcover, host trichotomy, quotient stream equality, selected representatives, countable choice, or an unrecorded $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BoundedIntervalCauchySubsequenceUp.TasteGate.BoundedIntervalCauchySubsequenceTaste`
`single carrier alignment`

Theorem 3716.2 (Bounded interval Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *carrier*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded interval row I, source sequence row S, finite stream-window ledger W, dyadic tolerance ledger D, monotone cofinal schedule T, regular rational handoff Q, subsequence-stability row E, terminal real seal R, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

The obligation surface also includes classifier stability under the same displayed rows, dyadic tolerance exactness through D, finite-window visibility through W, cofinal schedule replay through T, $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through Q,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through R, and provenance visibility through P, N.*

Proof. Carrier admission is projection from Definition 3716.1. The projection exposes exactly the interval, source, window, dyadic, schedule, readback, subsequence-stability, seal, transport, replay, provenance, and naming rows listed in the statement.

The classifier is componentwise on those rows. Hence any accepted comparison must preserve the same bounded interval row, the same finite source-window surface, the same dyadic ledger, and the same schedule route before it reaches the regular-readback and real-seal rows.

The nonescape clauses follow from the absence of further coordinates in the carrier. There is no slot for open-cover compactness, host trichotomy, quotient stream equality, countable choice, selected representatives, or an ambient completeness principle, so no local NameCert consumer can read such data from the packet. □

Theorem 3716.3 (Bounded interval dyadic window extraction)**.** *Let*

$$K = (I, S, W, D, T, Q, E, R, H, C, P, N)$$

be an accepted $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *carrier. Every finite extraction read from* K *factors through the displayed dyadic ledger* D *and the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger* W *before it may consume the monotone cofinal schedule* T *or the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* Q*.*

Consequently each selected comparison window is certified by endpoint interval data from I*, inspected source windows from* S*, dyadic radii and pairwise bounds from* D*, and replay data from* C*, rather than by a hidden infinite subsequence, trichotomy decision, open cover, quotient stream, or choice principle.*

Proof. Start with an accepted extraction read and project the carrier. The only rows that mention finite observations are W and D: W lists the inspected source and endpoint windows, while D records the mesh radius, endpoint comparisons, pairwise distance bounds, and late-window budgets attached to those observations.

The schedule row T is downstream from those finite ledgers. It records which displayed windows are reused along a monotone cofinal route; it does not add a new source of observations or a selection functional outside the packet.

The regular handoff row Q consumes W, D, T together with the interval source I, S. Replaying the route through C therefore forces every extraction read to pass through the finite dyadic-window data before any subsequence or regular Cauchy surface is available.

The forbidden principles have no carrier coordinate. Since the read has already exhausted the displayed rows, no infinite choice-selected subsequence, host trichotomy, open-cover compactness, quotient stream equality, or ambient completeness proof can be exported by the extraction route. □

Theorem 3716.4 (Bounded interval Cauchy subsequence Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ *packet must read the route*

$$I, S \longmapsto W, D, T \longmapsto Q, E \longmapsto R$$

before consuming the terminal seal row R*. Thus the handoff to* $\mathsf{Real}^{\uparrow}$ *and completion consumers is a finite BEDC route through bounded-interval endpoint data, stream windows, dyadic tolerances, a cofinal schedule, regular rational readback, subsequence stability,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The handoff does not close full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, quotient stream equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The terminal seal R is downstream from the interval and source rows I, S, the finite ledgers W, D, the schedule row T, the regular handoff Q, and the subsequence-stability row E.

Use Definition 3716.3 to bind the extraction stage to W and D. This ensures that the cofinal schedule and regular handoff are read only after the finite window and dyadic tolerance rows have been displayed.

The transport and replay rows H, C preserve the same ordered route, while P, N record provenance and the local naming surface. A $\mathsf{Real}^{\uparrow}$ consumer can therefore consume R only as the final row of that route.

The carrier contains no coordinate for the excluded compactness or choice principles. The theorem is therefore a finite-window handoff to Real and completion work, not a claim of unrestricted subsequence compactness or open-cover compactness for closed intervals. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}$ packet is generated from a bounded interval row, a source sequence row, finite StreamName windows, a DyadicRatCore tolerance ledger, a monotone cofinal schedule, RegSeqRat readback, RegularCauchySubsequenceStability handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3716.1, Definition 3716.2, Definition 3716.3, and Definition 3716.4 seed the finite-window bounded-interval subsequence packet over displayed Real, RegSeqRat, StreamName, DyadicRatCore, BoundedIntervalProjection, and RegularCauchySubsequenceStability rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedIntervalCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Bolzano–Weierstrass, Heine–Borel, open-cover compactness, trichotomy, quotient stream equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic window extraction, cofinal schedule replay, RegSeqRat handoff, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BoundedIntervalCauchySubsequenceUp.

Chapter 3717

A Concrete Naming Certificate for $\mathsf{BishopCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCompletion}^{\uparrow}$ packet names the finite boundary at which regular Cauchy rational names, stream windows, dyadic tolerance ledgers, filter-completion handoffs, and the terminal Real seal can be read as one completion-facing NameCert object. It sits on the L10 route supplied by the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic arithmetic ledger of chapter 291, the terminal real-name seal of chapter 38, the filter completion handoff of chapter 1569, and the universal extension boundary of chapter 397. The packet records a Bishop-style completion boundary for displayed names; it is not a quotient of streams, not a selected limit, not countable choice, not host real equality, and not an ambient completeness principle.

Definition 3717.1 (Bishop completion carrier)**.** *A* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, S, D, E, F, U, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name row, carrying the rational readback and regularity budget visible to completion-facing consumers.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row from which the displayed rational observations in R are read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and precision requests.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D have been displayed.*
5. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff row generated from the same finite tail windows and dyadic budgets.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row available only after the regular-name route and filter-completion handoff have both been exposed.*
7. *H records componentwise* hsame *transport for the regular name, stream window, dyadic ledger, Real seal, filter-completion handoff, universal-property row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion request through S, R, D, E, F, U.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows and their displayed replay paths. It has no coordinate for quotient stream equality, a selected representative, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionTasteGate single carrier alignment`

Theorem 3717.2 (Bishop completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier $B = (R, S, D, E, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed regular rational name, stream window, dyadic tolerance ledger, Real seal, filter-completion handoff, universal-property row, transport, replay, provenance, and local naming row;*
2. *finite-window exposure, requiring every completion-facing observation to pass through S before R or D can be read;*
3. *regular rational boundary exposure, requiring R and D to be displayed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E becomes available;*
4. *filter-compatibility exposure, requiring the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row F to be generated from the same S, R, D route;*
5. *universal-property exposure, requiring U to read only the already displayed regular-name, Real-seal, and filter-completion rows;*
6. *ledger non-escape, requiring H, C, P, N to export no quotient stream equality, selected representative, countable choice, host real equality, or ambient Real completeness coordinate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the accepted carrier through Definition 3717.1. The visible packet consists exactly of $R, S, D, E, F, U, H, C, P, N$.

The finite-window and regular-name obligations are ordered by the carrier. A completion-facing observation first exposes the window row S, then reads the regular rational row R and the dyadic ledger D before reaching the terminal Real seal E.

The filter and universal-property rows are downstream of the same route. The row F uses the displayed windows and dyadic budgets as its filter-completion handoff, while U is read only after the regular-name, Real-seal, and filter-completion rows have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since quotient stream equality, selected representatives, countable choice, host real equality, and ambient completeness have no carrier row, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionCarrier namecert obligations`

Theorem 3717.3 (Bishop completion regular real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier reads the completion boundary only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

In the notation of Definition 3717.1, the consumer reads S, then R and D, and only then the seal row E together with the structural rows H, C, P, N. The route exports no quotient stream, selected host limit, countable-choice schedule, host real equality, or ambient Real completeness datum.

Proof. Begin with a Real-facing read of $B = (R, S, D, E, F, U, H, C, P, N)$. The carrier has a single Real-seal coordinate, namely E, and E is placed after the window, regular-readback, and dyadic-ledger rows.

The window row S supplies the finite observations available to the read. The row R records the regular rational name extracted from those observations, and D records the dyadic tolerance ledger for the same precision requests.

The structural rows preserve this order. The hsame row H transports the displayed coordinates componentwise, the Cont row C replays the same route, and P, N record provenance and local naming without adding analytic data. Thus every Real-facing read remains inside the listed finite route. □

Theorem 3717.4 (Bishop completion filter compatibility)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *consumer of an accepted* $\mathsf{BishopCompletion}^{\uparrow}$ *carrier must read the same displayed regular-name boundary S, R, D, E before it can read the filter handoff F or the universal-property row U. Consequently sequence-style and filter-style completion reads agree on the finite BEDC support*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The compatibility exports no quotient filter, selected limit, countable choice, host equality proof, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.BishopCompletionUp.BishopCompletionCarrier filter compatibility`

Proof. Project the filter and universal rows from Definition 3717.1. The row F is downstream of the same stream, regular-readback, dyadic-ledger, and Real-seal route used by the regular-name boundary; the row U is downstream of both that boundary and F.

A filter-style read therefore cannot start from an independent filter representative. It first sees the displayed finite windows S, the regular rational readback R, and the dyadic tolerance ledger D, then reaches the Real seal E and only afterward the filter-completion handoff F.

The universal-property read is even later in the route. It is exposed through U after S, R, D, E, F have been displayed and after H, C, P, N have fixed the componentwise transport, replay, provenance, and local naming rows.

Since the carrier includes no quotient filter, selected limit, countable-choice selector, host equality proof, or ambient completeness coordinate, compatibility between the sequence-style and filter-style reads is exactly the shared finite support listed in the statement. □

Falsifiable prediction: Every $\mathsf{BishopCompletion}^{\uparrow}$ Real-facing or filter-facing read factors through displayed StreamName, RegSeqRat, DyadicRatCore, RealUp, CauchyFilterCompletion, CompletionUniversalProperty, transport, replay, provenance, and local naming rows.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchyfiltercompletion, completionuniversalproperty: $\mathsf{BishopCompletion}^{\uparrow}$ is not bijective to any listed sibling because it carries the regular-name route, Real seal, filter-completion handoff, and universal-property row in one finite completion boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\mathsf{BishopCompletion}\uparrow}$, seed)

Scope closed: Definition 3717.1, Definition 3717.2, Definition 3717.3, and Definition 3717.4 seed the finite Bishop completion boundary over regular Cauchy names, stream windows, dyadic ledgers, Real seals, filter completion, and universal-property rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopCompletion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, quotient filters, selected limits, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular Real boundary exposure, filter compatibility, universal-property exposure, and ledger non-escape for BEDC.Derived.BishopCompletionUp.

Constructive story: A BishopCompletion$^{\uparrow}$ packet is generated from a StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, CauchyFilterCompletion handoff, CompletionUniversalProperty row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3718

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ packet records the finite route by which a regular Cauchy rational stream exposes a modulus schedule before a $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the name. It sits over the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ tail-bound route of chapter 1659, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate makes the modulus extraction visible as finite NameCert data rather than as a prose appeal to a completed Cauchy object.

Definition 3718.1 (Regular Cauchy modulus extraction carrier)**.** *A* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the regular Cauchy source and by each downstream precision request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding displayed stream windows to rational observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii, comparison cells, half-radius splits, and endpoint observations.*
4. *T is the* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *row, displaying the tail bound that makes the requested dyadic scale stable after a finite stage.*
5. *E is the extracted modulus schedule, assigning each displayed tolerance request to the finite stage justified by S, R, D, T.*
6. *H records componentwise* hsame *transport for the window, readback, tolerance, tail-envelope, schedule, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, R, D, T \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, T, E, H, C, P, N$. It has no coordinate for selected limits, quotient equality of streams, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a modulus schedule chosen outside the displayed tail-envelope route.*

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusExtractionUp.RegularCauchyModulusExtractionTasteGate` `single carrier alignment`

Theorem 3718.2 (Regular Cauchy modulus extraction NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier*

$$M = (S, R, D, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, DyadicCauchyTailEnvelope bound, extracted schedule, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *readback exactness, requiring R to read only rational observations displayed in S and tolerance data displayed in D;*
3. *tail-bound exposure, requiring T to be the only row that certifies a stable tail after a finite stage for the requested dyadic scale;*
4. *modulus extraction exactness, requiring E to be computed from the displayed route S, R, D, T and from no detached selector;*
5. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the same displayed route.*

No $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 3718.1. The accepted packet displays exactly the window, readback, tolerance, tail-envelope, extracted schedule, transport, replay, provenance, and local naming rows.

Readback exactness is controlled by S, R, D. The regular rational observations are available only through displayed stream windows and displayed dyadic tolerance cells.

Tail-bound exposure is the role of T. It records the finite tail envelope attached to the same dyadic scale, so the packet has no independent convergence oracle.

The extracted schedule E is downstream of S, R, D, T. A consumer can therefore read a modulus only as the displayed finite schedule justified by the tail envelope.

Finally, H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no remaining coordinate stores a selected limit, quotient equality, outside schedule, trichotomy, or ambient Real-completeness principle, the listed rows exhaust the local certificate surface. □

Theorem 3718.3 (Regular Cauchy modulus extraction tail-bound route)**.** *Let*

$$M = (S, R, D, T, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer that reads a modulus from M must factor through the finite tail-bound route*

$$S, R, D \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives a displayed StreamName window row, RegSeqRat rational readback, DyadicRatCore tolerance ledger, DyadicCauchyTailEnvelope bound, extracted schedule, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *data, and receives no quotient stream equality, selected limit, trichotomy, countable choice, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credential.*

Proof. Begin with an accepted carrier and project the rows $S, R, D, T, E, H, C, P, N$. The only observation surface before a modulus is read is the displayed window, readback, and dyadic tolerance route.

The tail-bound row T is then read at the same dyadic scale. It supplies the finite stage after which the displayed rational tail is stable for the request.

The extracted schedule E is downstream of that tail-bound row. Thus any modulus read by a Real or completion consumer is the schedule justified by S, R, D, T.

The structural rows H, C, P, N transport, replay, source, and name the same route. Because the carrier has no coordinate for the excluded equality, selection, order, choice, completeness, or bridge objects, every such consumer factors through the displayed tail-bound route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusExtraction}^{\uparrow}$ packet is generated from a StreamName window row, RegSeqRat readback row, DyadicRatCore tolerance row, DyadicCauchyTailEnvelope bound row, extracted modulus schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3718.1, Definition 3718.2, and Definition 3718.3 seed the finite modulus-extraction route from regular Cauchy readbacks through dyadic tail envelopes and displayed schedules.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of Cauchy streams, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, readback exactness, tail-bound exposure, modulus extraction exactness, and structural stability for BEDC.Derived.RegularCauchyModulusExtractionUp.

Chapter 3719

A Concrete Naming Certificate for $\mathsf{CauchySelectionFunction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySelectionFunction}^{\uparrow}$ packet records the finite selector interface by which a Real and Completion consumer asks for a displayed regular Cauchy window without importing countable choice. It sits after the Cauchy modulus surface of chapter 292, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the displayed modulus-selection route of chapter 1986, and the selector-budget packet of chapter 678. It provides the finite selector row used by the Real completeness agenda of chapter 9238; it is not a countable-choice principle, quotient stream object, selected global limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3719.1 (Cauchy selection-function carrier)**.** *A* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the request row, naming the precision request, source regular Cauchy name, and completion-facing consumer boundary.*
2. *M is the modulus schedule row, read through the* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{CauchyModulusSelection}^{\uparrow}$ *surfaces for the displayed request.*
3. *W is the selected finite* $\mathsf{StreamName}^{\uparrow}$ *window opened by M for this request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row assembled from the selected window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger containing the dyadic radius, slack, comparison, and scale data used by the readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, M, W, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the request, schedule, selected window, regular readback, dyadic tolerance ledger, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the selector route*

$$Q, M \longmapsto W \longmapsto R, D \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchySelectionFunction}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusSelection}^{\uparrow}$, $\mathsf{RegularCauchySelectorBudget}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, M, W, R, D, E, H, C, P, N$. It has no coordinate for countable choice, quotient streams, selected global limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySelectionFunctionUp.CauchySelectionFunctionTasteGate single carrier alignment`

Theorem 3719.2 (Cauchy selection-function NameCert obligations)**.** *For every accepted* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier*

$$S = (Q, M, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the request row Q, modulus schedule M, selected finite window W, regular readback R, dyadic tolerance ledger D, terminal real seal E, componentwise transport H, continuation replay C, package provenance P, and local naming row N.*

The obligation surface also includes request-schedule compatibility, finite-window visibility, regular-readback exactness, dyadic tolerance compatibility, real-seal nonescape, and stability under the displayed hsame *and* Cont *rows. No obligation is stored outside $Q, M, W, R, D, E, H, C, P, N$.*

Proof. Carrier admission is projection from Definition 3719.1. The projection exposes exactly the request, schedule, selected-window, readback, dyadic, seal, transport, replay, provenance, and naming rows listed in the statement.

Request-schedule compatibility is the role of Q and M. The schedule is read as a displayed modulus-selection row for the request, and the selected finite window W is downstream of that schedule rather than a separate choice function.

Regular readback and dyadic tolerance exactness are the roles of R and D. The carrier places both rows after W, so the readback is assembled only from displayed observations and the displayed dyadic ledger.

The terminal real seal E is downstream of Q, M, W, R, D, while H, C, P, N transport, replay, source, and name the same finite route. Since no further coordinates occur in the packet, countable choice, quotient streams, selected global limits, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot be exported by the local certificate. □

Theorem 3719.3 (Cauchy selection-function regular readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySelectionFunction}^{\uparrow}$ *carrier*

$$S = (Q, M, W, R, D, E, H, C, P, N)$$

must first read the displayed request row Q, the modulus schedule M, the selected finite window W, and the dyadic tolerance ledger D. The regular readback row R and real seal row E then expose only the finite selector route

$$Q, M \longmapsto W, D \longmapsto R \longmapsto E$$

together with the structural rows H, C, P, N.

Thus the selector interface supplies a finite BEDC service for Real and Completion consumers without providing countable choice, quotient stream equality, a selected global limit, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read. By the carrier definition, the readback rows available to that consumer are R and E, and both are downstream of the request row, schedule row, selected window, and dyadic ledger.

The selected window is finite because it is the displayed $\mathsf{StreamName}^{\uparrow}$ row W. The schedule M only names the request-facing modulus surface that opens W; it does not add a second stream, a quotient stream, or an infinite family of choices.

The dyadic ledger D records the tolerance data used by the same finite window, and R assembles the regular rational readback from those displayed rows. The real seal E can therefore be consumed only as the terminal row of the route through Q, M, W, D, R.

Finally, H, C, P, N preserve and name the route under componentwise transport and continuation replay. The excluded objects have no carrier coordinate, so a consumer cannot read countable choice, quotient stream equality, selected global limits, host real equality, or ambient completeness from this packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySelectionFunction}^{\uparrow}$ packet is generated from a request row, a displayed modulus schedule, a selected finite StreamName window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3719.1, Definition 3719.2, and Definition 3719.3 seed the finite selector interface for Real and Completion consumers over displayed CauchyModulus, StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySelectionFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient streams, selected global limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-schedule compatibility, selected-window visibility, regular readback, dyadic tolerance compatibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySelectionFunctionUp.

Chapter 3720

A Concrete Naming Certificate for $\mathsf{CauchySumLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySumLimit}^{\uparrow}$ packet records the finite route by which two displayed regular Cauchy summands produce a regular Cauchy sum and then a terminal real-name seal. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name boundary of chapter 38. The chapter packages the familiar algebraic limit route for sums as NameCert data rather than as quotient streams, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3720.1 (CauchySumLimit carrier)**.** *A* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy summand rows, each exposing only its displayed finite regularity surface.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite read windows opened for both summands and for the sum-read request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the rational readbacks of the two summands and the pointwise sum readback obtained from the same windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, splitting every requested sum tolerance into displayed summand tolerances and recording the dyadic addition comparisons used by the readback row.*
5. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the sum, available only after S, Q, D have been displayed.*
6. *H records componentwise* hsame *transport for the summand rows, stream windows, regular readbacks, dyadic tolerance ledger, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from any Cauchy-sum-limit request through X, Y, S, Q, D, A.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySumLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, Q, D, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchySumLimitUp.CauchySumLimitTasteGate single carrier alignment toEventFlow`

Theorem 3720.2 (CauchySumLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier*

$$L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy summands, finite stream-window display, regular rational sum readback, dyadic tolerance splitting, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from quotient streams, selected limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3720.1. The only summand data are X and Y, and the only observation windows available to consumers are the finite windows recorded by S.

The readback stage is finite. The row Q records the rational observations of both summands and the pointwise sum readback obtained from those same windows, while D records the dyadic tolerance split and the addition comparisons that justify the requested sum tolerance.

The seal stage is terminal. The row A can be consumed only after the summand windows, readbacks, and tolerance ledger have been displayed, so a Real-facing consumer cannot bypass S, Q, D.

Finally H, C, P, N transport, replay, source, and name exactly the displayed route. Since the packet has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Lean4 ✓ `BEDC.Derived.CauchySumLimitUp.CauchySumLimitNamecertObligations`

Theorem 3720.3 (CauchySumLimit Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchySumLimit}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y \longmapsto S \longmapsto Q, D \longmapsto A, H, C, P, N.$$

Thus the sum limit is presented as a displayed regular-Cauchy algebra route to a Real seal, not as a quotient stream, selected limit, countable-choice schedule, trichotomy principle, or ambient completeness theorem.

Proof. Begin with an accepted packet $L_{\Sigma} = (X, Y, S, Q, D, A, H, C, P, N)$. A consumer first reads the displayed summand rows X and Y; no host sequence equality or quotient representative is available at this stage.

Next the stream windows in S determine the finite observations used for the request. The row Q reads both summands on those windows and records the pointwise rational sum, while D supplies the dyadic tolerance split and the corresponding addition comparisons.

The terminal Real seal is A. Since the continuation row C replays only the route through X, Y, S, Q, D, A, every Real-facing read reaches A after the finite regular readback and tolerance ledger have been displayed.

The transport and provenance rows preserve the same route under componentwise hsame. The local naming row N therefore exports the sum-limit seal without adding quotient streams, selected limits, countable choice, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3720.1, Definition 3720.2, and Definition 3720.3 seed the finite route from two regular Cauchy summands to StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, and a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySumLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, summand-window display, regular rational sum readback, dyadic tolerance splitting, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchySumLimitUp.

Constructive story: A $\mathsf{CauchySumLimit}^{\uparrow}$ packet is generated from two regular Cauchy summand rows, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3721

A Concrete Naming Certificate for $\mathsf{CauchyProductLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyProductLimit}^{\uparrow}$ packet records the finite route by which two displayed regular Cauchy factor rows produce a regular Cauchy product and then a terminal real-name seal. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic product-tolerance ledger of chapter 291, the product descent surface of chapter 461, and the real-name boundary of chapter 38. The chapter is the product analogue of chapter 3720: it packages a finite regular-Cauchy algebra route to a Real seal rather than quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3721.1 (CauchyProductLimit carrier)**.** *A* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two regular Cauchy factor rows, each exposing only its displayed finite regularity surface.*
2. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite read windows opened for both factors and for the product-read request.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the rational readbacks of both factors and the pointwise product readback obtained from the same shared window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product tolerance ledger, recording factor bounds, product-error splitting, dyadic multiplication comparisons, and the tolerance entries consumed by the product readback.*
5. *M is the* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent row. It can be read only after X, Y, S, Q, D have been displayed.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the product, available only after the regular product descent has reached the displayed readback boundary.*
7. *H records componentwise* hsame *transport for the factor rows, shared windows, regular readbacks, dyadic product ledger, product descent row, Real seal, replay row, provenance row, and local naming row.*

8. *C records displayed* Cont *replay from any product-limit request through X, Y, S, Q, D, M, A.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, Q, D, M, A, H, C, P, N$. It has no coordinate for quotient stream equality, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyProductLimitUp.CauchyProductLimitTasteGate single carrier alignment`

Theorem 3721.2 (CauchyProductLimit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier*

$$L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy factors, shared finite stream-window display, regular rational product readback, dyadic product-tolerance accounting,* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *descent, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3721.1. The only factor data are X and Y, and the only observation windows available to consumers are the finite shared windows recorded by S.

The readback stage is finite. The row Q records the rational observations of both factors and the pointwise product readback obtained from those same windows, while D records the dyadic factor bounds, product-error split, and multiplication comparisons that justify the requested product tolerance.

The descent stage is forced by row order. The row M is downstream of X, Y, S, Q, D, so a product-limit consumer cannot read $\mathsf{RegularCauchyProduct}^{\uparrow}$ evidence before the factor windows, product readback, and dyadic ledger have been displayed.

Finally A, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since the packet has no coordinate for quotient stream equality, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Theorem 3721.3 (CauchyProductLimit Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{CauchyProductLimit}^{\uparrow}$ *carrier factors through the finite route*

$$X, Y \longmapsto S \longmapsto Q, D \longmapsto M \longmapsto A, H, C, P, N.$$

Thus the product limit is presented as a displayed regular-Cauchy algebra route to a Real seal, not as a quotient stream, selected product limit, countable-choice schedule, trichotomy principle, host multiplication of completed reals, or ambient completeness theorem.

Proof. Begin with an accepted packet $L_{\Pi} = (X, Y, S, Q, D, M, A, H, C, P, N)$. A consumer first reads the displayed factor rows X and Y; no host sequence equality, quotient representative, or completed-real multiplication primitive is available at this stage.

Next the shared stream windows in S determine the finite observations used for the request. The row Q reads both factors on those windows and records the pointwise rational product, while D supplies the dyadic factor bounds and product-tolerance ledger.

The regular product descent is M. Since M is downstream of the factor windows, readbacks, and tolerance ledger, the terminal Real seal A is reached only after the finite product route has already been displayed.

The transport and provenance rows preserve the same route under componentwise hsame. The local naming row N therefore exports the product-limit seal without adding quotient streams, selected limits, countable choice, trichotomy, host multiplication of completed reals, or ambient Real completeness. □

Theorem 3721.4 (CauchyProductLimit shared-window stability)**.** *Let*

$$L_\Pi = (X, Y, S, Q, D, M, A, H, C, P, N)$$

and

$$L'_\Pi = (X', Y', S', Q', D', M', A', H', C', P', N')$$

be accepted $\mathsf{CauchyProductLimit}^\uparrow$ *carriers. If the two packets preserve the two factor rows, the shared* $\mathsf{StreamName}^\uparrow$ *window, the* $\mathsf{RegSeqRat}^\uparrow$ *readback, the* $\mathsf{DyadicRatCore}^\uparrow$ *product ledger, the* $\mathsf{RegularCauchyProduct}^\uparrow$ *descent row, and the terminal* $\mathsf{Real}^\uparrow$ *seal componentwise under* hsame*, then every product-limit consumer reads the same finite route through* $S, Q, D, M, A, H, C, P, N$*. In particular the read is stable under the displayed product-limit route and cannot be replaced by quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the two carrier displays from Definition 3721.1. The admissible coordinates of a product-limit read are exactly the factor rows, the shared window row, the regular rational readback, the dyadic product ledger, the product descent row, the Real seal, the componentwise transport, replay, provenance, and local naming rows.

Apply Definition 3721.2 to both packets. Carrier admission binds every accepted read to the displayed rows, while the shared-window, readback, dyadic-ledger, descent, and seal clauses rule out any extra route not present in L_Π or L'_Π.

Use the componentwise hsame hypotheses next. They identify the two factor rows, S with S', Q with Q', D with D', M with M', and A with A' at the exact sites where a consumer can inspect the product-limit packet.

Finally apply the Real-seal route theorem Definition 3721.3. Since the terminal read must pass through the finite route from X, Y to $S, Q, D, M, A, H, C, P, N$, the preserved components force the same product-limit route and keep quotient streams, selected product limits, countable choice, trichotomy, host multiplication, and ambient Real completeness outside the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductLimit}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductLimit}\uparrow}$, seed)

Scope closed: Definition 3721.1, Definition 3721.2, and Definition 3721.3 seed the finite route from two regular Cauchy factor rows to shared StreamName windows, RegSeqRat readbacks, a DyadicRatCore product ledger, RegularCauchyProduct descent, and a RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductLimit}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected product limits, countable choice, trichotomy, host multiplication of completed reals, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-window display, regular rational product readback, dyadic product-tolerance accounting, RegularCauchyProduct descent, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyProductLimitUp.

Constructive story: A $\mathsf{CauchyProductLimit}^\uparrow$ packet is generated from two regular Cauchy factor rows, shared finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore product-tolerance ledger, a RegularCauchyProduct descent row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3722

A Concrete Naming Certificate for $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ packet records the finite diagonal selector route used by the real-completeness corridor of chapter 9238. It concretizes the transported stationary diagonal theorem of Definition 9238.17 as a NameCert packet over the no-choice selector boundary of chapter 695, the cofinal tail route of chapter 697, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The selector is a displayed Bishop-style regular-Cauchy diagonal construction, not a host choice function, quotient stream, selected completed limit, trichotomy branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness primitive.

Definition 3722.1 (RealCompletenessDiagonalSelector carrier)**.** *A* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, E, T, S, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the regular Cauchy source-family row whose finite observations are available through the displayed real-completeness route.*
2. *E is the* $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$ *row, recording the explicit precision request, selected finite window, and refusal of a host choice function.*
3. *T is the* $\mathsf{DiagonalTailSelector}^{\uparrow}$ *row, recording the cofinal tail index and the finite tail-window handoff consumed by the diagonal route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row shared by the selector, tail, and readback requests.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected diagonal windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the precision budget, selected-window comparisons, and tail-cofinality tolerance entries.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the selected diagonal representative, available only after E, T, S, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for the source-family, selector, tail, window, readback, dyadic ledger, Real seal, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from any diagonal-selector request through R, E, T, S, Q, D, A.*
10. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$, $\mathsf{ChoiceFreeDiagonalSelector}^{\uparrow}$, $\mathsf{DiagonalTailSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, E, T, S, Q, D, A, H, C, P, N$. It has no coordinate for host choice, quotient stream equality, selected completed limits, hidden tail oracles, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCompletenessDiagonalSelectorUp.RealCompletenessDiagonalSelectorTasteGate`
`single carrier alignment`

Theorem 3722.2 (RealCompletenessDiagonalSelector NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier*

$$G = (R, E, T, S, Q, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular Cauchy source family, choice-free finite selector admission, cofinal diagonal-tail routing, shared StreamName window display, RegSeqRat selected-window readback, DyadicRatCore precision and tail-cofinality ledgers, terminal RealUp seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3722.1. The source-family row R is only readable through the finite selector and tail rows E and T, so no host family choice or completed-limit selector is available before the displayed rows have been opened.

The finite selector stage is explicit. The row E records the precision request and selected window without a host choice function, while T records the cofinal tail handoff used by the same request.

The observation stage is also finite. The shared window row S, readback row Q, and dyadic ledger D name the StreamName, RegSeqRat, and DyadicRatCore data consumed by the diagonal representative before the Real seal A can be read.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier has no coordinate for host choice, quotient stream equality, selected completed limits, hidden tail oracles, trichotomy, or ambient Real completeness, none of those objects can be exported by the local certificate. □

Theorem 3722.3 (RealCompletenessDiagonalSelector non-escape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or real-completeness-facing consumer of an accepted* $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto E, T \longmapsto S, Q, D \longmapsto A, H, C, P, N.$$

The transported stationary diagonal row of Definition 9238.17 is therefore consumed as finite selector, tail, window, readback, dyadic, seal, transport, replay, provenance, and local naming data, not as host choice, quotient stream equality, selected completed limit, hidden tail oracle, trichotomy, or ambient completeness.

Proof. Start with an accepted packet $G = (R, E, T, S, Q, D, A, H, C, P, N)$. The source-family row R is not a completed real carrier; it becomes visible to a consumer only through the displayed selector and tail rows.

The selector rows E and T fix the finite observation request. The former records the no-choice selection boundary, and the latter records the cofinal tail route that serves the same diagonal window.

The rows S, Q, D then supply the finite StreamName window, RegSeqRat readback, and DyadicRatCore tolerance ledger. The Real seal A is downstream of these rows, so a consumer cannot bypass them to obtain an independent RealUp limit.

Transport and replay preserve the row order under componentwise hsame. The local naming row N therefore exports only the displayed diagonal-selector route and cannot export host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RealCompletenessDiagonalSelectorUp.RealCompletenessDiagonalSelectorNonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3722.1, Definition 3722.2, and Definition 3722.3 seed the finite Bishop-style diagonal selector route from regular Cauchy source rows through ChoiceFreeDiagonalSelector and DiagonalTailSelector rows to StreamName windows, RegSeqRat readbacks, a DyadicRatCore ledger, and a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host choice, quotient streams, selected completed limits, hidden tail oracles, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, choice-free selector admission, cofinal tail routing, selected-window readback, dyadic precision and tail-cofinality ledgers, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.RealCompletenessDiagonalSelectorUp.

Constructive story: A $\mathsf{RealCompletenessDiagonalSelector}^{\uparrow}$ packet is generated from a regular Cauchy source-family row, a ChoiceFreeDiagonalSelector row, a DiagonalTailSelector row, shared StreamName windows, RegSeqRat selected-window readbacks, a DyadicRatCore precision ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3723

A Concrete Naming Certificate for $\mathsf{LeibnizSeriesTest}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of alternating series, the $\mathsf{LeibnizSeriesTest}^{\uparrow}$ packet records the classical alternating-series tail-bound route as finite BEDC data before any completion-facing consumer may read a regular rational handoff or a real seal. It refines the convergence-test surface of chapter 1483, uses the regular rational readback of chapter 301, opens only the finite observation windows of chapter 126, measures tail tolerances through the dyadic core of chapter 291, and terminates at the real-name boundary of chapter 38. The packet names the finite Leibniz tail-bound route; it does not import selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3723.1 (LeibnizSeriesTest carrier)**.** *A* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, listing the finite partial-sum and summand windows opened by the request.*
2. *Σ is the finite alternating-sign row, recording the parity and sign entries visible on the displayed windows.*
3. *M is the monotone magnitude row, recording nonnegative dyadic magnitude reads and their finite monotonicity comparisons on the inspected tail.*
4. *T is the tail-bound schedule, assigning each requested dyadic tolerance to a displayed tail index whose next visible magnitude is within that tolerance.*
5. *Q is the* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *handoff row, accepting the source, alternating-sign, monotone-magnitude, and tail-bound rows as the local convergence-test face.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed partial-sum stream after the handoff row has been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after S, Σ, M, T, Q, R have been read.*
8. *H records componentwise* hsame *transport for the source, sign, magnitude, tail-bound, handoff, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any Leibniz-series-test request through* S, Σ, M, T, Q, R, E.
10. *P is the* Pkg *provenance over* $\mathsf{LeibnizSeriesTest}^{\uparrow}$, $\mathsf{AlternatingSeriesTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, \Sigma, M, T, Q, R, E, H, C, P, N$. It has no coordinate for selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LeibnizSeriesTestUp.LeibnizSeriesTestTasteGate single carrier alignment`

Theorem 3723.2 (LeibnizSeriesTest NameCert obligations)**.** *For every accepted* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier*

$$L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$ *surface consists exactly of finite source-window admission, alternating-sign display, monotone magnitude display, dyadic tail-bound scheduling,* $\mathsf{AlternatingSeriesTest}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay, provenance, local naming, and non-escape from selected sums, quotient series equality, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LeibnizSeriesTestUp.LeibnizSeriesTestNameCertObligations`

Proof. Project the carrier of Definition 3723.1. The only source observations are the finite windows in S, and the only sign information is the displayed parity ledger Σ.

The tail estimate is finite. The row M gives the visible monotone magnitude comparisons, while T records the tail index chosen for each requested dyadic tolerance. The handoff row Q then accepts exactly those source, sign, magnitude, and tail-bound rows through the $\mathsf{AlternatingSeriesTest}^{\uparrow}$ surface.

The completion-facing route is also finite. The regular readback R may be read only after Q, and the terminal seal E receives only R together with the displayed source, sign, magnitude, and tail-bound route.

Finally H, C, P, N transport, replay, source, and name that same route. Since no carrier coordinate stores a selected sum, quotient series equality, countable-choice schedule, trichotomy premise, or ambient completeness theorem, none of those objects can be exported by the local certificate. □

Theorem 3723.3 (LeibnizSeriesTest Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal-facing read exported by an accepted* $\mathsf{LeibnizSeriesTest}^{\uparrow}$ *carrier factors through the finite route*

$$S, \Sigma, M, T \longmapsto Q \longmapsto R \longmapsto E, H, C, P, N.$$

Thus the Leibniz tail-bound route reaches the terminal Real seal only through displayed alternating signs, monotone magnitudes, dyadic tail bounds, the alternating-series-test handoff, and regular rational readback; it does not expose selected sums, quotient series equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted packet $L_\lambda = (S, \Sigma, M, T, Q, R, E, H, C, P, N)$. A Real-facing consumer must first open the displayed source windows S, because no partial-sum observation is available outside those windows.

The sign and magnitude checks are then forced by Σ and M. The tail-bound schedule T is the only row that converts a requested dyadic tolerance into a displayed tail index, and the handoff row Q records that those finite rows enter the alternating-series-test surface.

The regular rational handoff is R, and the Real seal is E. Since E is available only after R and the finite handoff route have been displayed, any seal-facing read stays inside the named route.

The transport, replay, provenance, and local naming rows preserve that route componentwise. Hence the exported boundary cannot contain selected sums, quotient series equality, countable choice, trichotomy, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3723.1, Definition 3723.2, and Definition 3723.3 seed the finite Leibniz tail-bound route from displayed alternating signs and monotone magnitudes through RegSeqRat readback to a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeibnizSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign and magnitude display, tail-bound scheduling, alternating-series-test handoff, regular rational readback, Real-seal routing, transport and replay stability, provenance exhaustion, and non-escape for BEDC.Derived.LeibnizSeriesTestUp.

Constructive story: A $\mathsf{LeibnizSeriesTest}^{\uparrow}$ packet is generated from finite StreamName windows, an alternating-sign row, monotone magnitude row, dyadic tail-bound schedule, AlternatingSeriesTestUp handoff, RegSeqRatUp readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3724

A Concrete Naming Certificate for $\mathsf{LipschitzModulus}^{\uparrow}$

Following Bishop–Bridges constructive metric analysis and Kelley-style uniform-space bookkeeping, the $\mathsf{LipschitzModulus}^{\uparrow}$ packet records the finite tolerance object carried by a displayed Lipschitz bound. It sits over the source and target metric rows of chapter 57, the real-bound seal of chapter 38, the Lipschitz map and constant surfaces of chapter 410 and chapter 2328, the division and rounding ledger of chapter 291, and the exported modulus and continuity consumers of chapter 369 and chapter 391. The packet names a finite tolerance-only route; it is not a host proof of continuity, a quotient of maps, a selected inverse for a real bound, or an ambient completeness theorem.

Definition 3724.1 (LipschitzModulus carrier)**.** *A* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, F, K, E, D, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *F is the* $\mathsf{LipschitzMap}^{\uparrow}$ *row whose graph and source-distance reads use X and Y.*
3. *K is the* $\mathsf{LipschitzConstant}^{\uparrow}$ *row for the same source, target, and graph data.*
4. *E is the positive target-tolerance row, read through* $\mathsf{Real}^{\uparrow}$ *rather than through a host scalar.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division and rounding ledger that turns E and the displayed bound in K into a source-tolerance request.*
6. *U is the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row consumed by* $\mathsf{UniformContinuity}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *routes.*
7. *H records componentwise* hsame *transport for metric rows, map rows, constant rows, tolerance rows, dyadic division and rounding rows, the exported modulus, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from a target-tolerance request through E, K, D to the source-tolerance row exported by U.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for an unbounded host Lipschitz proof, a quotient-function equality, a hidden choice of reciprocal, a completed function space, or an ambient real-completeness argument.

Lean4 ✓ `BEDC.Derived.LipschitzModulusUp.LipschitzModulusTasteGate single carrier alignment`

Theorem 3724.2 (LipschitzModulus NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier* $M = (X, Y, F, K, E, D, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the source metric, target metric, Lipschitz map, Lipschitz constant, positive tolerance, dyadic division and rounding, exported modulus, transport, replay, provenance, and local naming rows;*
2. *bound alignment, requiring F and K to read the same source, target, and graph rows before a tolerance conversion is exported;*
3. *tolerance positivity, requiring E to be displayed before the dyadic ledger is read;*
4. *modulus export, requiring every* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer to pass through E, K, D, U in that order;*
5. *ledger non-escape, excluding host Lipschitz proofs, quotient map equality, hidden reciprocal choice, completed function spaces, selected limits, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 3724.1. The displayed packet exposes exactly the metric, Lipschitz-map, Lipschitz-constant, positive-tolerance, dyadic-ledger, modulus, transport, replay, provenance, and naming rows.

The bound-alignment and positivity obligations are row-order checks. The map row F and constant row K share the source and target metric rows, and the tolerance row E is the only positive target request available to the dyadic ledger D.

The export obligation follows because U is the only $\mathsf{UniformModulus}^{\uparrow}$ row in the carrier. A continuity consumer therefore reaches U only after reading the displayed tolerance, constant, and dyadic division and rounding route.

The refusal clause is an inventory check on the carrier. None of the displayed rows supplies host continuity, quotient map equality, hidden reciprocal selection, completed function-space data, selected limits, choice, or ambient completeness. □

Theorem 3724.3 (LipschitzModulus uniform-continuity handoff)**.** *Let* $M = (X, Y, F, K, E, D, U, H, C, P, N)$ *be an accepted* $\mathsf{LipschitzModulus}^{\uparrow}$ *carrier. Every* $\mathsf{UniformContinuity}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer that requests a target tolerance receives only the source-tolerance route exported by U after the packet has displayed the target tolerance E, the aligned Lipschitz constant K, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *division and rounding ledger D. Thus the handoff is a finite modulus route, not a rereading of a host Lipschitz proof.*

Proof. Start from the carrier of Definition 3724.1. The source and target metrics are X and Y, the graph and bound rows are F and K, and the only target-tolerance request is E.

The dyadic ledger D performs the finite division and rounding step from the displayed tolerance and bound data. Since D is the only arithmetic route in the packet, no consumer can replace it by a host reciprocal or by a selected real approximation.

The exported row U is then read by the $\mathsf{UniformModulus}^{\uparrow}$ surface and by downstream $\mathsf{UniformContinuity}^{\uparrow}$ and $\mathsf{ContinuousMap}^{\uparrow}$ consumers. Transport through H and replay through C preserve this order, so the consumer receives only the displayed finite modulus route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3724.1, Definition 3724.2, and Definition 3724.3 seed the finite Lipschitz tolerance route from displayed metric, Lipschitz-map, Lipschitz-constant, positive-tolerance, dyadic division and rounding, and uniform-modulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient function spaces, host continuity proofs, reciprocal choice, completed Sobolev or function-space carriers, selected limits, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, bound alignment, tolerance positivity, dyadic division and rounding exactness, uniform-modulus export, transport stability, and replay stability for BEDC.Derived.LipschitzModulusUp.

Constructive story: A $\mathsf{LipschitzModulus}^{\uparrow}$ packet is generated from source and target metric rows, Lipschitz map and constant rows, a positive tolerance row, a DyadicRatCore division and rounding ledger, an exported UniformModulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 3725

A Concrete Naming Certificate for $\mathsf{ModulusCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ packet records the finite completion endpoint surface obtained when a displayed Cauchy modulus is bound to regular rational readback and a terminal real seal. It sits over the modulus interface of chapter 292, the regular rational surface of chapter 301, the finite stream-window route of chapter 126, the dyadic tolerance ledger of chapter 291, the Bishop regular Cauchy completion packet of chapter 1639, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a modulus-driven endpoint handoff; it does not choose a quotient representative, assert ambient completeness, or select limits by countable choice.

Definition 3725.1 (Modulus Cauchy completion carrier)**.** *A* $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, carrying the finite precision-to-window schedule used by the completion endpoint request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the finite source windows admitted by the modulus schedule.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the displayed stream windows as regular rational Cauchy data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic thresholds, comparison rows, and endpoint error budgets used by M, S, R.*
5. *E is the endpoint seal row, read through the Bishop regular completion boundary and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
6. *H stores componentwise* hsame *transport for the modulus, stream, readback, dyadic ledger, endpoint seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from the modulus schedule to source windows, from source windows to regular readback, from the dyadic ledger to endpoint tolerances, and from the regular completion handoff to the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *Bishop regular Cauchy completion,* $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, S, R, D, E, H, C, P, N$. It contains no coordinate for quotient completion equality, selected host limits, countable choice, ambient real completeness, or a bridge-checked standard endpoint.

Lean4 ✓ `BEDC.Derived.ModulusCauchyCompletionUp.ModulusCauchyCompletionTasteGate single carrier alignment`

Theorem 3725.2 (Modulus Cauchy completion NameCert obligations)**.** *For every accepted* ModulusCauchyComple *carrier $Q = (M, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed modulus, stream-window, regular readback, dyadic tolerance, endpoint seal, transport, replay, provenance, and local naming rows;*
2. *modulus compatibility, requiring every endpoint request to read its threshold and observation window through the same* $\mathsf{CauchyModulus}^{\uparrow}$ *row M and* $\mathsf{StreamName}^{\uparrow}$ *row S;*
3. *regular-readback stability, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be transported componentwise from the displayed stream windows and dyadic tolerance rows;*
4. *endpoint handoff, requiring the endpoint row E to be exposed only after the Bishop regular completion boundary and terminal* $\mathsf{Real}^{\uparrow}$ *seal have read M, S, R, D;*
5. *ledger nonescape, excluding quotient completion equality, host real equality, selected limits, countable choice, ambient completeness, and bridge checking from the local carrier.*

Proof. Project the carrier of Definition 3725.1. The rows $M, S, R, D, E, H, C, P, N$ are the full accepted inventory, so carrier admission and classifier comparison are finite.

The modulus and stream rows are read together. A precision request reaches an observation window only through M and S, and the dyadic row D records the tolerance ledger attached to that same read.

The regular readback and endpoint handoff are downstream. The row R is obtained from the displayed windows and tolerance data, while E is exposed only after the Bishop regular completion boundary and the $\mathsf{Real}^{\uparrow}$ seal consume the same rows.

Finally inspect the missing coordinates. Since the carrier has no quotient equality, host real equality, selected limit, countable-choice, ambient-completeness, or bridge-checking row, the local NameCert surface cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusCauchyCompletion}^{\uparrow}$ packet is generated from a CauchyModulus row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an endpoint Real/completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3725.1 and Definition 3725.2 seed the finite modulus-to-endpoint packet over completion-facing regular rational Cauchy data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulusCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient completion equality, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus compatibility, regular-readback stability, endpoint handoff, and ledger nonescape for the same finite packet.

Chapter 3726

A Concrete Naming Certificate for $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ packet records the finite-cover substrate used by located compact-uniform-continuity arguments on closed bounded interval data. It refines the closed interval source of chapter 1591, the finite ball-cover carrier of chapter 2377, the positive-radius ledger of chapter 3606, the compact uniform-continuity route of chapter 2292, and the metric-ball observation surface of chapter 1450. The packet names only finite located cover rows, local membership witnesses, Lebesgue-number handoff data, transport, replay, provenance, and local naming evidence.

Definition 3726.1 (Uniformly located finite cover carrier)**.** *A* $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including endpoint order, rational and dyadic containment ledgers,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *F is the* $\mathsf{FiniteRealCover}^{\uparrow}$ *row whose ball cells, centers, dyadic radii, coverage witnesses, and refinement ledger cover the inspected interval windows.*
3. *L is the* $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$ *ledger row that records the positive-radius handoff for the displayed finite cover.*
4. *M is the metric-ball and local locatedness membership row: each inspected interval window is paired with an admitted* $\mathsf{MetricBall}^{\uparrow}$ *cell and a finite locatedness witness for membership in that cell.*
5. *H records componentwise* hsame *transport for I, F, L, M, C, P, N and for the endpoint, cover, radius, ball, membership, and local naming coordinates exposed by those rows.*
6. *C records displayed* Cont *replay from interval containment rows to finite cover coverage checks, from coverage checks to local locatedness membership, and from the cover ledger to the Lebesgue-number handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{FiniteRealCover}^{\uparrow}$, $\mathsf{FiniteCoverLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformContinuityModulus}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, a choice-selected subcover, a host-set membership oracle, quotient real equality, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.UniformlyLocatedFiniteCoverUp.UniformlyLocatedFiniteCoverTasteGate single carrier alignment`

Theorem 3726.2 (Uniformly located finite cover NameCert obligations)**.** *For an accepted* UniformlyLocatedFiniteC *carrier* $U = (I, F, L, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple; finite cover coverage through* F *over the interval source* I*; local locatedness preservation through the metric-ball membership row* M*; Lebesgue-number handoff through* L*; componentwise transport stability through* H*; displayed replay through* C*; provenance through* P*; local naming through* N*; and ledger nonescape from open-cover compactness, selected subcovers, host membership oracles, quotient real equality, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 3726.1. The displayed tuple exposes the closed bounded interval source, finite cover row, positive-radius ledger, local metric-ball membership row, and the structural rows H, C, P, N.

Coverage is read only through F after I has supplied its endpoint, containment, window, readback, and Real-seal data. Locatedness preservation is read through M, where each finite cell membership remains attached to an admitted $\mathsf{MetricBall}^{\uparrow}$ row and to the same interval window.

The Lebesgue-number handoff is downstream of the displayed finite cover: L may be consumed only after the cover row and local membership ledger have been replayed. The transport and replay rows then preserve the same interval-cover-membership route componentwise.

No excluded object is a coordinate of U. The local certificate therefore exports only the displayed finite located cover route, with P and N recording provenance and naming for that route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}$ packet is generated from a closed bounded interval source, a finite real cover row, a finite-cover Lebesgue-number ledger, metric-ball located membership rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3726.1 and Definition 3726.2 seed the finite located cover substrate consumed by compact uniform-continuity routes over closed bounded interval data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyLocatedFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, choice-selected subcovers, host-set membership oracles, quotient real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coverage exactness, locatedness preservation, Lebesgue-number handoff, transport and replay stability, and provenance exhaustion for BEDC.Derived.UniformlyLocatedFiniteCoverUp.

Chapter 3727

Cauchy Condensation Theorem NameCert Construction

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ packet records the theorem-level NameCert surface that turns a monotone nonnegative $\mathsf{RealSeries}^{\uparrow}$ tail into the dyadic block series and back through Cauchy criteria. It sits after the dyadic-block packet of chapter 1464, the comparison-test surface of chapter 1478, the Real-series carrier of chapter 3345, and the finite dyadic partition ledger of chapter 1863. The packet is a theorem certificate over displayed tails, block ledgers, and Cauchy handoff rows; it is not a host summability principle, a quotient of infinite series, a selected sum, countable choice, arbitrary trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3727.1 (Cauchy condensation theorem carrier)**.** *A* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the monotone nonnegative* $\mathsf{RealSeries}^{\uparrow}$ *source row, read only through finite term windows and partial-sum windows.*
2. *M is the monotone-tail ledger, recording the nonnegativity row, decreasing transport, and the finite tail cut from which the theorem is applied.*
3. *D is the* $\mathsf{DyadicPartition}^{\uparrow}$ *block schedule, listing the powers of two, block endpoints, and finite block intervals used by the theorem.*
4. *B is the original block-sum ledger, storing the finite sums of A over the dyadic intervals named by D.*
5. *G is the condensed-series row, displaying the dyadic terms $2^n a_{2^n}$ on the same block schedule.*
6. *Q is the blockwise comparison ledger, recording the lower and upper estimates that monotonicity supplies between B and G.*
7. *U is the original-series Cauchy criterion row, expressed through displayed partial-sum windows of A.*
8. *V is the condensed-series Cauchy criterion row, expressed through displayed partial-sum windows of G.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the original and condensed partial-sum windows.*
10. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-facing row reached only after the Cauchy criterion handoff has passed through U, V, R.*
11. *H records componentwise* hsame *transport for the source, monotone ledger, dyadic schedule, block sums, condensed terms, comparison ledger, Cauchy rows, readback row, seal row, continuation, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from each finite tail request through the dyadic blocks, comparison ledger, Cauchy criteria, regular readback, and seal-facing row.*
13. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{CauchyCondensationTest}^{\uparrow}$, $\mathsf{CauchyCondensationRealSeries}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, M, D, B, G, Q, U, V, R, E, H, C, P, N$. It has no coordinate for a host infinite sum, quotient equality of series, selected endpoint, countable choice, an arbitrary trichotomy split, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyCondensationTheoremUp.CauchyCondensationTheoremTasteGate single carrier alignment`

Definition 3727.2 (Blockwise Cauchy comparison classifier)**.** *Two accepted* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carriers are blockwise Cauchy equivalent when their displayed* $\mathsf{RealSeries}^{\uparrow}$ *source rows, monotone-tail ledgers, dyadic schedules, original block sums, condensed terms, comparison rows, original and condensed Cauchy criterion rows, regular readbacks, seal-facing rows, transports, continuations, provenance, and local names are pairwise* hsame *through the rows named in Definition 3727.1. Equivalence is not allowed to inspect an unlisted host limit, a quotient of streams, or a chosen infinite sum.*

Theorem 3727.3 (Cauchy condensation theorem NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *carrier $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyConden}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed fourteen-row packet;*
2. *monotone nonnegative source exposure through A and M;*
3. *dyadic block scheduling through D and original block-sum formation through B;*
4. *condensed dyadic-term formation through G;*
5. *blockwise tail comparison through Q, using only monotone transport on displayed finite windows;*
6. *original-to-condensed Cauchy transport through U, V, R;*
7. $\mathsf{Real}^{\uparrow}$ *seal exposure through E only after the Cauchy handoff has reached the regular readback row;*
8. *transport stability, continuation replay, provenance, and local naming through H, C, P, N.*

No obligation row asserts host summability, a selected infinite sum, quotient equality of real streams, countable choice, arbitrary trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$. The carrier row exposes exactly the monotone Real-series source, monotone-tail ledger, dyadic block schedule, original block sums, condensed terms, comparison ledger, two Cauchy criterion rows, regular readback, seal-facing row, and structural rows.

The source and classifier obligations are finite. The rows A and M provide nonnegativity and decreasing behavior only on displayed term windows, while Definition 3727.2 permits comparison only through the carrier coordinates named in Definition 3727.1.

The dyadic block ledger is then read. The schedule D fixes the inspected powers of two and finite intervals, B records the original block sums on those intervals, and G records the condensed terms $2^n a_{2^n}$ on the same indices.

Monotone transport enters only through Q. Each block comparison is a finite estimate between rows already displayed by A, M, D, B, G, so the theorem-level packet does not appeal to an infinite selector of blocks or to host equality of series.

Finally the Cauchy-convergence handoff is read through U, V, R, E. The original and condensed Cauchy criterion rows route the two partial-sum tails into the same regular readback boundary before the $\mathsf{Real}^{\uparrow}$ seal row is visible. The rows H, C, P, N preserve, replay, source, and name that route, so the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ consumer that claims theorem-level condensation without reading the monotone source row, dyadic block schedule, original block sums, condensed terms, blockwise comparison ledger, Cauchy criterion rows, and $\mathsf{RegSeqRat}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: cauchycondensation, cauchy-condensation-test, cauchy-condensation-real-series, dyadic-partition, realseries, regseqrat, real: no listed sibling supplies the whole fourteen-row theorem packet $K = (A, M, D, B, G, Q, U, V, R, E, H, C, P, N)$, because each sibling contributes only a dyadic block, comparison-test, Real-series, partition, readback, or seal face while this chapter binds the theorem-level equivalence through blockwise comparison and Cauchy handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ packet is generated from a monotone nonnegative RealSeries source row, a monotone-tail ledger, a DyadicPartition block schedule, original block sums, condensed dyadic terms, a blockwise comparison ledger, original and condensed Cauchy criterion rows, RegSeqRat readback, a RealUp seal-facing row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3727.1, Definition 3727.2, and Definition 3727.3 seed the theorem-level dyadic block carrier, classifier, monotone transport ledger, and Cauchy-convergence handoff for monotone nonnegative real series.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient equality of series, selected sums, countable choice, arbitrary trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode BEDC.Derived.CauchyCondensationTheoremUp and prove carrier admission, blockwise classifier stability, monotone dyadic transport, the block ledger, and the Cauchy criterion handoff.

Chapter 3728

A Concrete Naming Certificate for $\mathsf{Sylow}^{\uparrow}$

Following Sylow's finite group theorem, the $\mathsf{Sylow}^{\uparrow}$ packet records a prime-power subgroup witness inside a displayed finite $\mathsf{Group}^{\uparrow}$ carrier. It consumes the group and subgroup rows of chapter 41 and chapter 80, the prime and unary counting rows of chapter 64 and chapter 28, and the finite action surface of chapter 3662. The packet names only a displayed p-subgroup carrier, maximal prime-power coverage, conjugacy transport, and representation-facing consumer boundary; it does not assert a host subgroup lattice, quotient group classification, character table, arbitrary normalizer computation, or complete finite-group structure theorem.

Definition 3728.1 (Sylow carrier)**.** *A* $\mathsf{Sylow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (G, S, p, n, W, A, T, C, H, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row, including multiplication, unit, inverse, finite support, and componentwise group transport.*
2. *S is the consumed* $\mathsf{NameCert}_{\mathsf{Subgroup}^{\uparrow}}$ *row naming the candidate subgroup inside G.*
3. *p is the* $\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$ *row naming the prime whose power controls the subgroup order.*
4. *n is the* $\mathsf{Nat}^{\uparrow}$ *unary exponent row recording the displayed prime-power size p^n.*
5. *W is the membership and order witness row saying that the subgroup carrier has exactly the displayed p^n support and that no larger displayed p-subgroup row is present in the packet.*
6. *A is the finite conjugation-action row, read through the group operation of G and through the action discipline of* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *when a downstream representation consumer asks for transport of a Sylow row.*
7. *T is the conjugacy-transport row sending any displayed Sylow witness with the same prime-power coverage to a conjugate subgroup row inside G.*
8. *C is the consumer ledger for representation, character, and subgroup-normalizer reads; it exposes only the rows G, S, p, n, W, A, T before a consumer may name a Sylow conclusion.*
9. *H records componentwise* hsame *transport for group, subgroup, prime, exponent, coverage, action, conjugacy, consumer, replay, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{Sylow}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, S, p, n, W, A, T, C, H, P, N$. It has no coordinate for an ambient subgroup lattice, quotient classification, selected normalizer, character-table computation, or host finite-group theorem.

Lean4 (definition) `BEDC.Derived.SylowUp.SylowCarrier`

Lean4 ✓ `BEDC.Derived.SylowUp.SylowTasteGate single carrier alignment`

Theorem 3728.2 (Sylow NameCert obligations)**.** *For every accepted* $\mathsf{Sylow}^{\uparrow}$ *carrier* $Y = (G, S, p, n, W, A, T, C, H,$ *the local* $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$ *surface consists exactly of finite group admission through G, subgroup admission through S, prime admission through p, unary exponent exposure through n, prime-power coverage through W, conjugation-action exposure through A, conjugacy transport through T, consumer-boundary exposure through C, componentwise transport through H, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3728.1. The rows G, S, p, n, W expose the finite group, candidate subgroup, prime, unary exponent, and prime-power coverage before any Sylow read can be consumed.

The conjugation row A is finite because it is read from the displayed group operation and, when representation-facing consumers are present, through the finite action surface of chapter 3662. The transport row T is therefore a row of the packet, not a host orbit theorem.

Finally C, H, P, N preserve, source, and name the same route. Since no ambient subgroup lattice, quotient classification, selected normalizer, character-table computation, or complete finite-group structure theorem is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier namecert obligations`

Theorem 3728.3 (Sylow conjugacy transport)**.** *Let $Y = (G, S, p, n, W, A, T, C, H, P, N)$ and $Y' = (G, S', p, n, W', A', T', C', H', P', N')$ be accepted* $\mathsf{Sylow}^{\uparrow}$ *packets over the same displayed finite* $\mathsf{Group}^{\uparrow}$ *row and the same* $\mathsf{Prime}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *rows. If both coverage ledgers display maximal p^n support inside G, then every consumer that identifies S with S' must pass through the conjugation-action rows A, A' and the transport rows T, T'. The exported comparison is a displayed conjugacy transport inside G, not equality of host subgroup sets or a quotient of all subgroups.*

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier conjugacy transport`

Proof. Read the two carriers and their shared rows G, p, n. The only rows that connect subgroup witnesses with the ambient group operation are the coverage rows W, W' and the conjugation-action rows A, A'.

The transport rows T, T' record the allowed movement between displayed maximal prime-power subgroup rows. Thus a consumer comparing S and S' must use those rows before it may name a conjugacy result.

The consumer and structural rows $C, C', H, H', P, P', N, N'$ preserve the same displayed route. Because neither carrier contains a host subgroup lattice or quotient over all subgroup rows, the comparison remains a finite conjugacy-transport packet inside G. □

Theorem 3728.4 (Sylow prime-power coverage)**.** *For an accepted* $\mathsf{Sylow}^{\uparrow}$ *carrier* $Y = (G, S, p, n, W, A, T, C, H, P, N$ *every prime-power coverage read of* S *in* G *factors through the prime row* p*, the unary exponent row* n*, and the coverage ledger* W*. The packet may expose that the displayed subgroup has maximal* p^n *support among the subgroup rows named by the packet, but it does not compute the full host order of* G*, enumerate all subgroups of* G*, or choose a canonical Sylow subgroup.*

Proof. Project p, n, W from the carrier. The prime row supplies the arithmetic base, the unary exponent row supplies the finite counting index, and the coverage ledger records the displayed p^n support for the subgroup row S.

Apply Definition 3728.2. No coverage conclusion is available before G, S, p, n, W have been read, and no later row can replace W with an ambient order computation.

The action, transport, consumer, and structural rows A, T, C, H, P, N only replay the same coverage route. Therefore prime-power coverage is a packet-level finite witness, with full subgroup enumeration and canonical selection outside the chapter. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier prime power coverage`

Theorem 3728.5 (Sylow obligation closure surface)**.** *The* $\mathsf{Sylow}^{\uparrow}$ *obligation surface is exactly the carrier admission, prime-power coverage, conjugacy transport, and consumer-boundary route exposed by Definition 3728.2, Definition 3728.3, and Definition 3728.4.*

Proof. Apply Definition 3728.2 for the carrier-admission and consumer-boundary rows. Apply Definition 3728.4 for the prime-power ledger, and apply Definition 3728.3 for the displayed conjugacy route.

Together these three blocks expose the finite group, subgroup, prime, exponent, coverage, action, transport, consumer, structural, provenance, and local naming rows before any Sylow consumer can read the packet. No further host subgroup lattice, canonical selector, quotient classification, or character-table computation is a coordinate of the carrier. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier obligation closure surface`

Theorem 3728.6 (Sylow scope dependency lock)**.** *Every* $\mathsf{Sylow}^{\uparrow}$ *consumer must first read the* $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows of Definition 3728.1. Only after those rows are exposed may the consumer name any prime-power coverage, conjugacy transport, or representation-facing consumer-boundary conclusion.*

Proof. Project the carrier and apply Definition 3728.2. The rows $G, S, p, n, W, A, T, C, H, P, N$ are the only admission, arithmetic, transport, replay, provenance, and local naming coordinates available to a consumer.

Apply Definition 3728.4 for the coverage read and Definition 3728.3 for the transport read. Both conclusions depend on the displayed rows already named by the carrier.

The consumer-boundary row C and the structural rows H, P, N then prevent any conclusion from escaping to an ambient subgroup lattice, quotient classification, character table, normalizer computation, or canonical Sylow selector. Thus the dependency scope is locked before the chapter exposes any Sylow conclusion. □

Lean4 ✓ `BEDC.Derived.SylowUp.SylowCarrier scope dependency lock`

Closure status for $\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Sylow}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 3728.1, Definition 3728.2, Definition 3728.3, and Definition 3728.4 seed the finite prime-power subgroup packet over displayed $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, and $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{Sylow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subgroup lattices, quotient group classification, character-table computation, arbitrary normalizer calculation, canonical Sylow selection, Lean verification, or bridge checking.

Upgrade path: Carrier admission, prime-power coverage, and conjugacy transport already have checked targets; the remaining next step is checked representation-facing consumer-boundary and public-scope coverage, followed by bridge discipline for BEDC.Derived.SylowUp.

Constructive story: A Sylow$^{\uparrow}$ packet is generated from a finite GroupUp row, a SubgroupUp row, a PrimeUp row, a NatUp exponent row, a prime-power coverage ledger, conjugation-action and transport rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3729

A Concrete Naming Certificate for $\mathsf{SylowTheorem}^{\uparrow}$

Following Sylow and the finite-group presentation in Rotman, the $\mathsf{SylowTheorem}^{\uparrow}$ packet names the theorem-facing finite-action route over the existing $\mathsf{Sylow}^{\uparrow}$ witness packet. It consumes the displayed finite $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Sylow}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, and $\mathsf{ClassEquation}^{\uparrow}$ rows of chapter 41, chapter 80, chapter 64, chapter 28, chapter 3728, chapter 6559, and chapter 6631. The packet records only the finite action and counting route that a theorem consumer may read; it is not a host subgroup-lattice oracle, quotient classification, canonical Sylow selector, normalizer calculator, or complete finite-group structure theorem.

Definition 3729.1 (Sylow theorem carrier)**.** *A* $\mathsf{SylowTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (G, p, n, Y, A, O, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *source row whose multiplication, unit, inverse, and finite support are already displayed.*
2. *p is the consumed* $\mathsf{NameCert}_{\mathsf{Prime}^{\uparrow}}$ *row and n is the* $\mathsf{Nat}^{\uparrow}$ *exponent row for the displayed p^n divisor route.*
3. *Y is the consumed* $\mathsf{Sylow}^{\uparrow}$ *packet naming the prime-power subgroup witness and its coverage ledger inside G.*
4. *A is the finite conjugation-action row read through* $\mathsf{OrbitStabilizer}^{\uparrow}$*.*
5. *O is the orbit-stabilizer counting row for the displayed conjugation orbit of Sylow witnesses.*
6. *Q is the class-equation remainder row recording the finite congruence and divisibility boundary used by the theorem-facing route.*
7. *R is the theorem-read row assembling existence, conjugacy, and counting conclusions only as finite BEDC rows over G, p, n, Y, A, O, Q.*
8. *H records componentwise* hsame *transport for the group, prime, exponent, Sylow witness, action, counting, theorem-read, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a theorem request through the same finite action and counting route.*

10. *P is the* Pkg *provenance over* $\mathsf{SylowTheorem}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Sylow}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, $\mathsf{ClassEquation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SylowTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient subgroup lattice, quotient group classification, selected canonical Sylow subgroup, arbitrary normalizer computation, character table, or complete finite-group structure theorem.

Theorem 3729.2 (Sylow theorem NameCert obligations)**.** *For every accepted* $\mathsf{SylowTheorem}^{\uparrow}$ *carrier* $T = (G, p, n, Y, A, O, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SylowTheorem}^{\uparrow}}$ *surface consists exactly of finite group admission through G, prime and exponent admission through p, n, Sylow witness consumption through Y, conjugation-action exposure through A, orbit-stabilizer counting through O, class-equation remainder exposure through Q, theorem-read assembly through R, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3729.1. The group, prime, exponent, and Sylow rows G, p, n, Y are displayed before any theorem-facing read may inspect existence, conjugacy, or counting conclusions.

The finite action rows A, O are read through the orbit-stabilizer packet, and the remainder row Q is read through the class-equation packet. Thus the theorem-read row R can assemble only the displayed finite action and counting route.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since no carrier coordinate stores a host subgroup lattice, quotient classification, canonical Sylow selector, normalizer calculator, character table, or complete finite-group structure theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 3729.3 (Sylow theorem finite-action route)**.** *Every theorem-facing read from an accepted* $\mathsf{SylowTheorem}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{Sylow}^{\uparrow} \longmapsto \mathsf{OrbitStabilizer}^{\uparrow} \longmapsto \mathsf{ClassEquation}^{\uparrow}.$$

The route may expose the existence, conjugacy, and counting rows assembled in R, but only after the group row G, prime row p, exponent row n, Sylow witness row Y, conjugation-action row A, orbit-stabilizer row O, and class-equation row Q have all been displayed.

Proof. Start with Definition 3729.2. It requires a theorem request to read G, p, n, Y before the finite action and counting rows become available.

Next consume the $\mathsf{Sylow}^{\uparrow}$ witness row Y. Its own carrier already exposes the prime-power subgroup witness inside the displayed group row, so the theorem-facing packet does not select a new subgroup.

Finally route the conjugation action through A, O and the counting remainder through Q. The replay and provenance rows C, P, N can name only this finite route, so R is a theorem-facing readback of displayed action and counting data rather than a host finite-group theorem. □

Theorem 3729.4 (Sylow theorem nonescape)**.** *An accepted* $\mathsf{SylowTheorem}^{\uparrow}$ *carrier cannot be used to recover data outside the displayed tuple $G, p, n, Y, A, O, Q, R, H, C, P, N$. In particular, it exports no host subgroup lattice, quotient classification of subgroups, canonical Sylow selector, arbitrary normalizer computation, character table, or complete finite-group structure theorem.*

Proof. Read the carrier in Definition 3729.1. The only theorem-facing mathematical rows are the displayed group, prime, exponent, Sylow witness, finite action, orbit-stabilizer, class-equation, and theorem-read rows.

Apply Definition 3729.3. Every theorem read must pass through $\mathsf{Sylow}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, and $\mathsf{ClassEquation}^{\uparrow}$ before R may be consumed.

The structural rows H, C, P, N preserve that same finite packet. Because no row contains a subgroup lattice, quotient classification, canonical selector, normalizer calculator, character table, or complete structure theorem, no such object can be recovered from the carrier. □

Falsifiable prediction: An accepted $\mathsf{SylowTheorem}^{\uparrow}$ read is refuted if it preserves the theorem-read row R while removing the consumed $\mathsf{Sylow}^{\uparrow}$, $\mathsf{OrbitStabilizer}^{\uparrow}$, or $\mathsf{ClassEquation}^{\uparrow}$ rows.

Independence witness: sylow, orbit-stabilizer, class-equation: none of the sibling packets is bijective with $\mathsf{SylowTheorem}^{\uparrow}$ because this packet jointly carries a Sylow witness row, a conjugation-action route, an orbit-stabilizer count, a class-equation remainder, theorem-read assembly, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SylowTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{SylowTheorem}^{\uparrow}$ packet is generated from a finite GroupUp row, a PrimeUp row, a NatUp exponent row, a SylowUp witness row, OrbitStabilizerUp action-counting rows, ClassEquationUp remainder rows, theorem-read assembly, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SylowTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3729.1, Definition 3729.2, Definition 3729.3, and Definition 3729.4 seed the theorem-facing finite-action Sylow route over displayed group, subgroup, prime, natural-number, Sylow, orbit-stabilizer, and class-equation rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SylowTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subgroup lattices, quotient classification, canonical Sylow selection, arbitrary normalizer calculation, character-table computation, complete finite-group structure, Lean verification, or bridge checking.

Upgrade path: Formal closure requires checked carrier admission, finite-action routing, orbit-stabilizer counting, class-equation remainder exposure, theorem-read assembly, transport, replay, provenance, and nonescape for BEDC.Derived.SylowTheoremUp.

Chapter 3730

A Concrete Naming Certificate for $\mathsf{BernsteinPolynomial}^{\uparrow}$

Following Bernstein and Bishop–Bridges constructive analysis, the $\mathsf{BernsteinPolynomial}^{\uparrow}$ packet records constructive uniform approximation over displayed $\mathsf{Real}^{\uparrow}$ interval data as finite BEDC rows. It sits after the polynomial carrier of chapter 50, the real-name surface of chapter 38, the compact-uniform continuity packet of chapter 388, the uniform-modulus carrier of chapter 369, and the rational observation rows of chapter 37. The chapter names the finite Bernstein basis, coefficient, mesh, and error-ledger surface; it does not import a host Weierstrass theorem, a selected infinite approximation sequence, quotient equality of real functions, or ambient completeness beyond the cited rows.

Definition 3730.1 (Bernstein polynomial carrier)**.** *A* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, U, M, S, K, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *closed interval row, exposing the endpoint names and the interval membership boundary used by the approximation request.*
2. *F is the sampled* $\mathsf{Real}^{\uparrow}$ *function-read row over I, read only through the compact-uniform continuity surface.*
3. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}/\mathsf{UniformModulus}^{\uparrow}$ *row that supplies a finite modulus request for the interval.*
4. *M is the dyadic mesh row, recording the chosen mesh scale, mesh points, and interval subdivision ledger.*
5. *S is the* $\mathsf{Rat}^{\uparrow}$ *coefficient row, listing the rational sample and binomial-weight reads used by the displayed Bernstein polynomial.*
6. *K is the* $\mathsf{Poly}^{\uparrow}$ *Bernstein-basis row, recording the finite basis terms and the polynomial assembly route.*
7. *E is the uniform-error ledger, comparing each mesh and interval cell read against the requested tolerance by rational bounds.*
8. *H records componentwise* hsame *transport for interval, function-read, modulus, mesh, coefficient, basis, error, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from interval admission through compact-uniform modulus selection, dyadic mesh construction, rational coefficient readback, polynomial assembly, and uniform-error certification.*

10. *P records* Pkg *provenance over* $\mathsf{BernsteinPolynomial}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$ *row.*

The classifier compares accepted Bernstein packets only through $I, F, U, M, S, K, E, H, C, P, N$. It has no coordinate for host function equality, a nonconstructive supremum, a choice-selected polynomial sequence, or an external real-analysis theorem.

Lean4 ✓ `BEDC.Derived.BernsteinPolynomialUp.BernsteinPolynomialTasteGate single carrier alignment`

Theorem 3730.2 (Bernstein polynomial NameCert obligations)**.** *For every accepted* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *carrier*

$$B = (I, F, U, M, S, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$ *surface consists exactly of interval admission, compact-uniform modulus exposure, dyadic mesh construction, rational coefficient readback, Bernstein-basis polynomial assembly, uniform-error ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Every approximation-facing consumer must read

$\mathsf{Real}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert

through these rows before it receives a Bernstein approximation certificate.

Theorem 3730.3 (Bernstein uniform approximation boundary)**.** *Given an accepted* $\mathsf{BernsteinPolynomial}^{\uparrow}$ *packet and a displayed tolerance request, the exported approximation boundary is the finite route*

$$I, F, U \longmapsto M \longmapsto S, K \longmapsto E \longmapsto H, C, P, N.$$

The packet may certify only that the displayed Bernstein polynomial row meets the displayed uniform-error ledger over the displayed interval. It does not claim a free-standing Weierstrass theorem, an ambient complete function space, a quotient equality class of continuous functions, or a selected infinite chain of approximants.

Closure status for $\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BernsteinPolynomial}^{\uparrow}$ packet is generated from Real interval rows, compact-uniform modulus rows, dyadic mesh rows, Rat coefficient rows, Polynomial Bernstein-basis rows, a uniform-error ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 3730.1, Definition 3730.2, and Definition 3730.3 seed the finite constructive approximation certificate over interval, modulus, mesh, coefficient, basis, error, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BernsteinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the Weierstrass approximation theorem, host function equality, selected approximation sequences, ambient complete function spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval admission, modulus exposure, dyadic mesh construction, rational coefficient readback, polynomial assembly, uniform-error exactness, transport, replay, provenance, and local naming for BEDC.Derived.BernsteinPolynomialUp.

Chapter 3731

A Concrete Naming Certificate for $\mathsf{SequentialCauchySelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentialCauchySelector}^{\uparrow}$ packet records the finite selector rows needed when a displayed finite family of Cauchy requests is answered one request at a time, without turning that finite ledger into countable choice or ambient real completeness. It sits after the single-request selection surface of chapter 3719, the choice-sequence boundary of chapter 1482, the located choice-sequence boundary of chapter 2088, the stream-window carrier of chapter 126, the regular rational readback row of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The packet names a finite sequential handoff; it does not choose an infinite sequence of witnesses, quotient-identify completed reals, or assert a general completeness principle.

Definition 3731.1 (Sequential Cauchy selector carrier)**.** *A* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, I, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request ledger, listing the Cauchy-selection queries whose answers are demanded by the packet.*
2. *M is the modulus schedule, assigning each listed request its displayed precision and finite window budget.*
3. *I is the finite index ledger, ordering the listed requests and recording which rows have already been answered.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite stream windows admitted by Q, M, I.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and their regular rational comparison data.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, recording dyadic thresholds, comparison rows, and error budgets for every listed request.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing a terminal real-facing name only after the selected stream window, regular readback, and dyadic tolerance rows have been read.*

8. *H records componentwise* hsame *transport for the request, modulus, index, window, readback, dyadic, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each finite request through its modulus schedule, index row, stream window, regular readback, dyadic ledger, and real seal.*

10. *P records* Pkg *provenance over* $\mathsf{SequentialCauchySelector}^{\uparrow}$, $\mathsf{CauchySelectionFunction}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LocatedChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$ *row.*

The classifier compares accepted selector packets only through $Q, M, I, W, R, D, E, H, C, P, N$. It has no coordinate for countable choice, an infinite choice sequence, quotient real equality, ambient real completeness, or a choice-selected family beyond the listed requests.

Lean4 ✓ `BEDC.Derived.SequentialCauchySelectorUp.SequentialCauchySelectorTasteGate single carrier alignment`

Theorem 3731.2 (Sequential Cauchy selector NameCert obligations)**.** *For every accepted* SequentialCauchySelector *carrier $S = (Q, M, I, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the request ledger, modulus schedule, index ledger, selected stream windows, regular rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*

2. *request-index exactness, requiring every selected row to be indexed by I and sourced from a displayed query in Q;*

3. *modulus-window compatibility, requiring every precision demand to pass through M before it reads a finite* $\mathsf{StreamName}^{\uparrow}$ *window in W;*

4. *readback and seal stability, requiring R, D, E to be transported componentwise from the same request, modulus, index, and window rows;*

5. *boundary refusal, excluding countable choice, infinite choice sequences, quotient real equality, ambient real completeness, and unlisted selected witnesses from the local carrier.*

Proof. Project the carrier of Definition 3731.1. The tuple $Q, M, I, W, R, D, E, H, C, P, N$ is the full accepted inventory, so each NameCert obligation must be read from one of these finite rows.

The request-index and modulus-window clauses are internal to Q, M, I, W. A selected window is exposed only after the query appears in Q, its finite index appears in I, and its precision demand passes through M.

The readback and seal clauses are downstream. The rows R and D read the displayed windows as regular rational and dyadic data, while E records the terminal real-facing seal obtained from the same finite route.

Finally H and C preserve the route under transport and replay, and P, N record provenance and local naming. Since no row stores countable choice, an infinite sequence, quotient real equality, ambient completeness, or an unlisted witness, the NameCert surface cannot export any of those objects. □

Theorem 3731.3 (Sequential Cauchy selector finite-window determinacy)**.** *Every consumer of an accepted* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *packet reads each selected Cauchy answer through the finite*

route

$$Q, I \longmapsto M \longmapsto W \longmapsto R, D \longmapsto E \longmapsto H, C, P, N.$$

Consequently two accepted reads agreeing on $Q, M, I, W, R, D, E, H, C, P, N$ determine the same finite selected windows, the same regular rational readback rows, the same dyadic tolerance ledgers, and the same terminal real-seal rows for the listed requests.

Proof. Fix an accepted carrier and a listed request. The index ledger I identifies the request row in Q, and the modulus schedule M fixes the precision and finite window budget attached to that row.

The selected window row W is read only after those rows have been fixed. Thus any second accepted read agreeing on Q, M, I, W reaches the same finite stream windows.

The rows R and D are readbacks from those windows, and the seal row E is exposed only after the same readback and tolerance data. Agreement on R, D, E therefore fixes the regular rational, dyadic, and real-facing coordinates for the listed request.

Transport, replay, provenance, and naming do not add fresh selector data. The rows H, C, P, N preserve and certify the same finite route, so determinacy is exactly the equality of the displayed packet rows. □

Theorem 3731.4 (Sequential Cauchy selector choice-boundary nonescape)**.** *Every read from an accepted* $\mathsf{SequentialCauchySelector}^{\uparrow}$ *packet into* $\mathsf{CauchySelectionFunction}^{\uparrow}$*,* $\mathsf{ChoiceSequence}^{\uparrow}$*,* $\mathsf{LocatedChoiceSequence}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *factors through the finite rows $Q, M, I, W, R, D, E, H, C, P, N$. The read may reuse the listed finite selections, but it cannot produce a countable-choice functional, an infinite choice sequence, quotient real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or selector rows for requests absent from Q.*

Proof. Start with the factorization route in Definition 3731.3. Every consumer sees the request, index, modulus, stream-window, readback, tolerance, and seal rows before it can inspect the structural rows.

The single-request surface of $\mathsf{CauchySelectionFunction}^{\uparrow}$ is therefore available only for the finitely listed requests. The $\mathsf{ChoiceSequence}^{\uparrow}$ and $\mathsf{LocatedChoiceSequence}^{\uparrow}$ chapters can read those displayed rows as finite sequence data, but no row supplies a law assigning witnesses to all future requests.

The regular rational, dyadic, and real-facing consumers likewise receive only R, D, E as finite downstream rows. These rows certify the listed readbacks and seals, not quotient equality of completed reals or ambient completeness.

Thus the packet remains inside its displayed BHist carrier. Any claimed countable-choice functional, infinite choice sequence, unlisted selector row, quotient real equality, or real-completeness theorem would require a coordinate absent from $Q, M, I, W, R, D, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCauchySelector}^{\uparrow}$ packet is generated from a finite request ledger, modulus schedule, index ledger, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3731.1, Definition 3731.2, Definition 3731.3, and Definition 3731.4 seed the finite sequential selector boundary over request, modulus, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCauchySelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, an infinite choice sequence, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, selector rows outside the finite request ledger, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-index exactness, modulus-window compatibility, readback and seal stability, and boundary-refusal proofs for BEDC.Derived.SequentialCauchySelectorUp.

Chapter 3732

A Concrete Naming Certificate for HyperbolicBusemannBoundary$^{\uparrow}$

The HyperbolicBusemannBoundary$^{\uparrow}$ packet realizes the boundary-fixed disk transport and horocycle reading of Definition 9463.10 as finite Busemann-level boundary data. It sits after the metric-read surface of chapter 3474, the horocycle packet of chapter 3534, the boundary-fixed transport route of chapter 3500, the disk phase-transport packet of chapter 3455, and the Denjoy-Wolff boundary-dynamics packet of chapter 3575. The chapter names a finite boundary metric and horocycle ledger; it does not turn the ideal boundary into a quotient compactification, select a global geodesic representative, or identify boundary dynamics with a host analytic fixed-point theorem.

Definition 3732.1 (Hyperbolic Busemann boundary carrier)**.** *A* HyperbolicBusemannBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, O, T, R, B, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* PoincareDiskPhaseTransport$^{\uparrow}$ *disk-phase row exposing the finite source, target, and reversible disk-coordinate route.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row over the same displayed disk observations.*
3. *O is the* HyperbolicHorocycle$^{\uparrow}$ *horocycle row, recording the boundary-tangent horocycle read used by the packet.*
4. *T is the* PoincareBoundaryFixedTransport$^{\uparrow}$ *row, requiring the displayed boundary end to remain fixed during the disk transport.*
5. *R is the* DenjoyWolffBoundaryDynamics$^{\uparrow}$ *boundary-dynamics row consumed only as displayed finite boundary-dynamics data.*
6. *B is the Busemann-level row, recording the finite distance-difference read against the fixed boundary end.*
7. *L is the boundary-horocycle ledger, pairing the Busemann-level row with the horocycle row and the boundary-fixed transport audit.*
8. *H records componentwise* hsame *transport for disk phase, metric, horocycle, boundary-fixed transport, boundary dynamics, Busemann-level, ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through disk phase, hyperbolic metric, horocycle read, boundary-fixed transport, boundary dynamics, Busemann-level comparison, and boundary-horocycle ledger exposure.*

10. *P records* Pkg *provenance over* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{DenjoyWolffBoundaryDynamics}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted boundary packets only through $D, M, O, T, R, B, L, H, C, P, N$. It has no coordinate for quotient ideal-boundary equality, a selected geodesic representative, a host compactification, or an analytic fixed-point theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.HyperbolicBusemannBoundaryUp.HyperbolicBusemannBoundaryTasteGate single carrier alignment`

Theorem 3732.2 (Hyperbolic Busemann boundary NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *carrier*

$$G = (D, M, O, T, R, B, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$ *surface consists exactly of disk-phase admission, hyperbolic-metric read compatibility, horocycle read exposure, boundary-fixed transport, boundary-dynamics provenance, Busemann-level distance-difference reading, boundary-horocycle ledger exactness, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

The Lean follow-up for this packet must provide `Nontrivial` *and* `FieldFaithful` *instances in* `lean4/BEDC/Derived/HyperbolicBusemannBoundaryUp/TasteGate.lean`*; the paper packet itself records only the finite NameCert obligation surface.*

Theorem 3732.3 (Hyperbolic Busemann horocycle route)**.** *Every accepted* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary read factors through*

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, DenjoyW

before the Busemann-level row and boundary-horocycle ledger are exposed. A consumer that bypasses the metric read, horocycle read, boundary-fixed transport, provenance, or local NameCert *row is outside the accepted packet.*

Falsifiable prediction: Any accepted boundary consumer omitting $\mathsf{HyperbolicMetric}^{\uparrow}$, horocycle, boundary-fixed transport, provenance, or local NameCert rows refutes the $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ packet.

Independence witness: `hyperbolic_metric`, `hyperbolic_horocycle`, `poincare_boundary_fixed_transport` `poincare_disk_phase_transport`, `denjoy_wolff_boundary_dynamics`: none is bijective to HyperbolicBusemannB because the packet adds the Busemann-level row and the boundary-horocycle ledger tying metric, horocycle, fixed-boundary transport, boundary dynamics, provenance, replay, and local naming into one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ packet is generated from disk-phase transport rows, HyperbolicMetric reads, HyperbolicHorocycle rows, boundary-fixed transport, Denjoy-Wolff boundary-dynamics rows, a Busemann-level distance-difference row, a boundary-horocycle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3732.1, Definition 3732.2, and Definition 3732.3 seed the finite boundary metric, horocycle, fixed-transport, Busemann-level, ledger, provenance, replay, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{HyperbolicBusemannBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient ideal-boundary equality, selected geodesic representatives, host compactifications, analytic fixed-point theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-phase admission, metric-read compatibility, horocycle exposure, boundary-fixed transport, Busemann-level reading, boundary-horocycle ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicBusemannBoundaryUp.

The BooleanAlgebra$^{\uparrow}$ hub routes its finite join, meet, complement, endpoint, order, transport, replay, provenance, and local naming rows to the lattice and Stone-duality consumers of chapter 55 and chapter 261.

Chapter 3733

A Concrete Naming Certificate for $\mathsf{BooleanAlgebra}^{\uparrow}$

Following Stone and Birkhoff, the $\mathsf{BooleanAlgebra}^{\uparrow}$ packet records the finite algebraic endpoint where complemented distributive lattice operations meet two-element truth rows. It sits beside the order carrier of chapter 53, the meet–join interface of chapter 55, and the endpoint discriminator of chapter 32; it supplies a finite order-algebra neighbour for later Stone-duality and topology bridges without assuming a host Boolean algebra. The packet consumes the finite meet/join order rows of chapter 55 and the endpoint discriminator of chapter 32 before the Stone-duality handoff of chapter 261. Its complemented distributive-lattice rows require the meet and join witnesses, the order comparison, and the complement-to-zero and complement-to-one endpoint checks before they can serve as the algebraic input exported to chapter 261. Before any local Boolean-algebra NameCert row is accepted, the packet must display the $\mathsf{Bool}^{\uparrow}$ truth-value rows of chapter 32 and the $\mathsf{Lattice}^{\uparrow}$ meet, join, and complement-stability rows of chapter 55; the Boolean classifier then reads endpoints and operations only through those sibling certificates. The finite complemented distributive-lattice rows are the algebraic input later consumed by chapter 261, while this chapter itself does not construct the Stone space, ultrafilters, or topology. The same rows also serve topology-facing consumers: when a displayed open-set regularization route is read as Boolean meet, join, and complement, the consumer must first expose the finite complemented distributive-lattice packet here, then pass through the indexed-open surface of chapter 88 and the clopen ledger of chapter 261. Thus regular-open and clopen uses receive the same algebraic rows, not a second host Boolean algebra or a powerset representation.

Definition 3733.1 (BooleanAlgebra BHist carrier)**.** *A* $\mathsf{BooleanAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (J, M, K, Z, O, R, H, C, P, N)$$

with the following displayed rows:

1. *J is the binary join row, read as the finite least-upper-bound operation inherited from the lattice-facing surface.*
2. *M is the binary meet row, read as the finite greatest-lower-bound operation inherited from the lattice-facing surface.*
3. *K is the complement row, assigning to each displayed carrier element its finite complement witness.*
4. *Z is the zero row, the finite bottom endpoint of the order surface.*

5. *O is the one row, the finite top endpoint of the order surface.*
6. *R is the partial-order row tying J, M, Z, O to the displayed* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *obligations.*
7. *H records componentwise* hsame *transport for join, meet, complement, endpoint, order, provenance, and local naming rows.*
8. *C records displayed* Cont *replay routes by which a consumer reads the operation, endpoint, order, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{Bool}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BooleanAlgebra}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host powerset algebra, excluded-middle oracle, ultrafilter space, locale of opens, quotient lattice, or ambient topological representation.

Lean4 (definition) `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier`

Theorem 3733.2 (BooleanAlgebra NameCert obligation surface)**.** *For the carrier of Definition 3733.1, the local* $\mathsf{NameCert}_{\mathsf{BooleanAlgebra}^{\uparrow}}$ *surface consists of exactly five finite obligation rows: carrier exposure for $J, M, K, Z, O, R, H, C, P, N$; lattice-order compatibility with the* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *rows; complement endpoint exactness, requiring $x \wedge Kx$ to read through Z and $x \vee Kx$ to read through O inside the displayed route; endpoint transport under componentwise* hsame *and displayed* Cont *replay; and non-escape from host Boolean algebras, powersets, ultrafilter spaces, locale carriers, quotient lattices, and topological representation theorems.*

Proof. Project the accepted carrier. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete visible inventory, with R carrying the order comparison and P naming the Bool, Poset, and Lattice sources.

Apply the sibling order surfaces. The poset row supplies the comparison route, while the lattice row supplies the meet and join readings; the Bool row supplies only finite endpoint discrimination, not an external truth oracle.

The complement row is checked against the displayed endpoints. The meet-with-complement route lands at Z, the join-with-complement route lands at O, and both reads are transported only by H and replayed only by C.

Since no row names a powerset algebra, ultrafilter space, locale carrier, quotient lattice, or ambient topology, coordinate exhaustion gives the stated non-escape boundary for the local naming certificate. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier namecert obligation surface`

Theorem 3733.3 (BooleanAlgebra Stone-duality forward route)**.** *Every* $\mathsf{StoneDuality}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet must read the finite join row J, meet row M, complement row K, zero row Z, one row O, order row R, componentwise* hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N before any Stone-space, ultrafilter, clopen-ledger, or topology bridge is public. Thus the Stone-duality source of chapter 261 starts from the displayed Boolean-algebra rows rather than from a host Boolean algebra or ambient Stone representation theorem.*

Proof. Project the accepted Boolean-algebra carrier. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete operation, endpoint, order, transport, replay, provenance, and naming inventory.

Apply the NameCert obligation surface. It requires the join, meet, complement, zero, one, and order rows to be displayed and transported before any downstream consumer can treat the packet as a Boolean-algebra source.

The Stone-duality chapter consumes exactly such a source row before it constructs clopen or topology-facing ledgers. Since the Boolean-algebra carrier contains no powerset algebra, ultrafilter space, locale carrier, quotient lattice, or ambient topology coordinate, no Stone-space or topology bridge can be public before these displayed rows are read. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone duality forward route`

Theorem 3733.4 (BooleanAlgebra Stone-source lattice determinacy)**.** *Any* StoneDuality$^{\uparrow}$ *consumer reading an accepted* BooleanAlgebra$^{\uparrow}$ *source receives the same finite join row J, meet row M, complement row K, zero row Z, one row O, order row R, componentwise* hsame *row H, displayed* Cont *replay row C,* Pkg *provenance row P, and local* NameCert *row N from the carrier of Definition 3733.1. The endpoint discriminator is the* Bool$^{\uparrow}$ *row named by P, while the meet/join order is read through the* Poset$^{\uparrow}$ *and* Lattice$^{\uparrow}$ *rows also named by P.*

Thus two Stone-duality consumers that agree on the accepted Boolean-algebra packet cannot diverge by choosing different host Boolean algebras, powerset representations, ultrafilter spaces, locale carriers, quotient lattices, or topology bridges before those displayed rows have been read.

Proof. Project the accepted carrier. The inventory $J, M, K, Z, O, R, H, C, P, N$ is exactly the finite operation, endpoint, order, transport, replay, provenance, and naming surface exposed by Definition 3733.1.

Apply Definition 3733.2. It forces the complement endpoint reads to pass through Z and O, and it forces the order comparison to be the displayed Poset$^{\uparrow}$/Lattice$^{\uparrow}$ route recorded by R and named by P.

Now use Definition 3733.3. Every StoneDuality$^{\uparrow}$-facing read starts from those Boolean-algebra rows before any Stone-space or topology-facing ledger is public. Since the carrier has no coordinate for a host Boolean algebra or alternate representation, the source boundary is deterministic. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone source lattice determinacy`

Closure status for NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$.

Constructive story: A BooleanAlgebra$^{\uparrow}$ packet is generated from finite join, meet, complement, zero, one, order, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert rows, with Bool, Poset, and Lattice sources read only through their displayed NameCert surfaces.

Theory closure: TheoryGate(NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 3733.2, Definition 3733.3, Definition 3733.4, Definition 3733.7, Definition 3733.8, and Definition 3733.10 bind the mature finite Bool, Poset, Lattice, Stone-source, and RegularOpenAlgebra route for Boolean algebra packets through the displayed finite order and algebra rows.

Formal verification: FormalStatus(NameCert$_{\text{BooleanAlgebra}^{\uparrow}}$, unformalized)

Bridge status: paperBridge

Origin: human

Not claimed: This chapter does not close Stone duality, Heyting algebra, locale theory, frame theory, ultrafilter construction, quotient Boolean algebra equality, host powerset semantics, or a Lean theorem.

Upgrade path: Lean verification requires checked carrier habitation, order-lattice compatibility, complement endpoint exactness, transport replay, and non-escape for the same finite packet.

Theorem 3733.5 (BooleanAlgebra dossier lattice route)**.** *Every top-level* BooleanAlgebra$^{\uparrow}$ *dossier read first consumes the* Bool$^{\uparrow}$ *endpoint rows and the* Lattice$^{\uparrow}$ *meet, join, and complement rows of the local carrier before it is available as a Boolean source for* StoneDuality$^{\uparrow}$*. The route adds no Stone space, ultrafilter, powerset representation, quotient lattice, or host Boolean-algebra equality row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraDossierLatticeRoute`

Proof. Project the carrier surface of chapter 3733. The endpoint coordinates are the displayed zero and one rows inherited from $\mathsf{Bool}^{\uparrow}$, while the algebraic coordinates are the meet, join, complement, and order rows inherited through the $\mathsf{Lattice}^{\uparrow}$ dependency route.

The Stone-duality handoff can read a Boolean source only after those local Boolean-algebra rows have been exposed. Since the dossier has no coordinate for Stone spaces, ultrafilters, powerset representatives, quotient lattices, or host equality, the outgoing route remains the existing Bool–Lattice–Stone dependency surface. □

The $\mathsf{BooleanAlgebra}^{\uparrow}$ dependency route consumes the $\mathsf{Bool}^{\uparrow}$ endpoint rows of chapter 32 together with the $\mathsf{Lattice}^{\uparrow}$ meet, join, and complement-facing lattice rows of chapter 55 before any $\mathsf{StoneDuality}^{\uparrow}$ handoff in chapter 261 is public.

Theorem 3733.6 (Boolean algebra lattice dependency route)**.** *Every accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *read exposes the finite meet, join, complement, endpoint, order, transport, replay, provenance, and local naming rows before any* $\mathsf{Lattice}^{\uparrow}$ *or* $\mathsf{StoneDuality}^{\uparrow}$ *consumer can use the packet. The lattice consumer of chapter 55 reads the meet and join rows together with the complement and endpoint rows as displayed Boolean data, while the Stone-duality consumer of chapter 261 receives the same rows only after that finite lattice surface has been made visible.*

Proof. Project the $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier rows used by the existing lattice and Stone routes. The meet and join rows are already displayed before the complement and endpoint rows can be consumed, and the structural rows only transport and replay that finite packet.

Thus a $\mathsf{Lattice}^{\uparrow}$ consumer reads the lattice part directly from the Boolean carrier. A $\mathsf{StoneDuality}^{\uparrow}$ consumer then uses the same Boolean-source rows as its input side, but no host Boolean algebra, powerset representation, or ambient topology is introduced between the two reads. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraLatticeDependencyRoute`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier consumes the finite meet, join, order, complement, and endpoint rows of chapter 55 before the Boolean source is read by chapter 261. Thus a $\mathsf{StoneDuality}^{\uparrow}$ handoff sees the same displayed $\mathsf{Lattice}^{\uparrow}$ dependency surface already present in the Boolean packet, rather than a host powerset algebra, quotient lattice, ultrafilter selection, or ambient topology.

Theorem 3733.7 (BooleanAlgebra obligation closure package)**.** *The accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet has an obligation-closure package over the displayed rows*

$$J, M, K, Z, O, R, H, C, P, N.$$

Carrier admission is the finite BHist *surface of Definition 3733.1; complement stability is the pair of endpoint reads* $x \wedge Kx$ *through* Z *and* $x \vee Kx$ *through* O *from Definition 3733.2; finite meet and join laws are consumed through the* $\mathsf{Lattice}^{\uparrow}$ *route of Definition 3733.6; and the Stone-source dependency is exported through Definition 3733.3 and Definition 3733.4. Thus the obligation package contains exactly carrier admission, complement stability, finite meet/join laws, Stone-source route, lattice dependency rows, transport, replay, provenance, and local naming, and no host powerset algebra, ultrafilter selector, quotient lattice, locale carrier, or ambient topology row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier obligation closure package`

Proof. Project the carrier from Definition 3733.1. The rows $J, M, K, Z, O, R, H, C, P, N$ are the complete finite inventory available to a Boolean-algebra consumer.

Use Definition 3733.2 for carrier admission and complement stability. It forces the complement row K to meet the zero endpoint Z and join the one endpoint O inside the displayed route, with transport and replay still attached.

Use Definition 3733.6 for the finite meet and join laws. The lattice dependency reads J and M together with complement, endpoint, order, transport, replay, provenance, and local naming rows before any lattice or Stone-duality consumer can use the packet.

Finally apply Definition 3733.3 and Definition 3733.4. They identify the Stone-source route with the same Boolean rows and prevent divergence through host Boolean algebras, powerset representations, ultrafilter spaces, quotient lattices, locale carriers, or topology bridges. These cited rows are exactly the obligation-closure package. □

Theorem 3733.8 (BooleanAlgebra public NameCert export)**.** *The finite* $\mathsf{BooleanAlgebra}^{\uparrow}$ *NameCert surface exported from Definition 3733.1 is public for* $\mathsf{StoneDuality}^{\uparrow}$ *and* $\mathsf{RegularOpenAlgebra}^{\uparrow}$ *consumers. A public read keeps the* $\mathsf{Bool}^{\uparrow}$ *endpoint rows,* $\mathsf{Poset}^{\uparrow}$ *order row,* $\mathsf{Lattice}^{\uparrow}$ *join and meet rows, complement row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row visible before either downstream consumer may treat the packet as a Boolean source.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier public namecert export`

Proof. Project the accepted carrier $B = (J, M, K, Z, O, R, H, C, P, N)$. The rows Z and O are the visible $\mathsf{Bool}^{\uparrow}$ endpoints, R is the $\mathsf{Poset}^{\uparrow}$ order route, and J, M, K are the $\mathsf{Lattice}^{\uparrow}$ join, meet, and complement-facing rows.

Use the local obligation surface and the obligation package. They bind complement endpoint exactness, order-lattice compatibility, transport, replay, provenance, and local naming to the same finite packet, so the public export cannot drop H, C, P, N while retaining a Boolean source.

The outgoing consumers read this same surface in two directions. $\mathsf{StoneDuality}^{\uparrow}$ consumes it as the Boolean-source side of Stone duality, while $\mathsf{RegularOpenAlgebra}^{\uparrow}$ consumes it as the algebraic surface behind regular-open meet, join, and complement. No host Boolean algebra, powerset representation, selected ultrafilter, quotient lattice, locale carrier, or ambient topology row is exported. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraPublicNameCertExport`

The $\mathsf{RegularOpenAlgebra}^{\uparrow}$ consumer of chapter 6835 reads the $\mathsf{BooleanAlgebra}^{\uparrow}$ join, meet, complement, endpoint, order, transport, replay, provenance, and local NameCert rows through topology-facing regularization. The read starts from the finite Boolean-algebra carrier of Definition 3733.1, passes through the regular-open closure and interior rows, and exports no second host Boolean algebra, powerset representation, quotient lattice, or detached topology carrier.

Theorem 3733.9 (BooleanAlgebra Stone-duality handoff)**.** *Every* $\mathsf{StoneDuality}^{\uparrow}$*-facing read of an accepted* $\mathsf{BooleanAlgebra}^{\uparrow}$ *packet first consumes the finite join row* J*, meet row* M*, complement row* K*, endpoint rows* Z, O*, order row* R*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local* NameCert *row* N *of chapter 3733. Only after those rows are exposed may the consumer reach the Boolean-source side of chapter 261.*

The handoff is additive to the local Boolean-algebra carrier: it exports a concrete outgoing dependency edge to $\mathsf{StoneDuality}^{\uparrow}$ *while adding no Stone space, ultrafilter, clopen topology, powerset representation, quotient lattice, or host Boolean-algebra equality row.*

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneDualityHandoff`

Proof. Project the accepted Boolean-algebra carrier from Definition 3733.1. Its visible operation and endpoint surface is exactly J, M, K, Z, O, R, with H, C, P, N carrying transport, replay, provenance, and local naming.

Apply Definition 3733.2. The local certificate requires the operation rows, endpoint exactness rows, and order compatibility rows to be displayed before any downstream packet can treat the carrier as a Boolean source.

The Stone-duality source of chapter 261 consumes a Boolean-algebra source before clopen or topology-facing rows appear. Therefore the Stone-duality-facing read must pass through the BooleanAlgebra rows first, and no absent Stone-space, ultrafilter, clopen-topology, powerset, quotient-lattice, or host-equality coordinate can be exported. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraCarrier stone duality handoff`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier exposes the join, meet, complement, endpoint, order, transport, replay, provenance, and local $\mathsf{NameCert}$ rows that a $\mathsf{StoneDuality}^{\uparrow}$ consumer later reads as its Boolean-source side. Thus the dependency into chapter 261 and the lattice-source handoff of chapter 55 is a finite classifier route through $J, M, K, Z, O, R, H, C, P, N$, not a hidden import of a host Boolean algebra, ultrafilter space, powerset representation, or ambient topology.

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneSourceLatticeForwardConsumer`

The same outgoing route also records the handoff row: after the Stone-source row is read from order and local naming, the provenance row transports it to the concrete downstream handoff row while the forward Stone-source certificate remains available to the same consumer.

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraStoneHandoffForwardConsumer`

The $\mathsf{BooleanAlgebra}^{\uparrow}$ carrier supplies the complemented distributive-lattice source read by chapter 261 and the Boolean endpoint discipline read by chapter 5153. The horizontal route is the same finite join, meet, complement, endpoint, order, transport, replay, provenance, and local $\mathsf{NameCert}$ surface; it does not add a host powerset algebra, selected ultrafilter, ambient locale frame, quotient lattice equality, or topology object outside the displayed Boolean rows.

Theorem 3733.10 (Boolean algebra mature export)**.** *The finite* $\mathsf{BooleanAlgebra}^{\uparrow}$ *rows routed through this hub provide the mature export consumed by the* $\mathsf{Lattice}^{\uparrow}$ *and* $\mathsf{StoneDuality}^{\uparrow}$ *sibling routes. The export consists exactly of the join row* J*, meet row* M*, complement row* K*, endpoint rows* Z, O*, order row* R*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}$ *row* N *of Definition 3733.1. It is the same packet exposed by the lattice dependency route, the Stone-duality handoff, the regular-open forward lattice route, the obligation closure package, and the public NameCert export.*

Proof. Project the carrier of Definition 3733.1. The rows $J, M, K, Z, O, R, H, C, P, N$ are the full finite Boolean-algebra inventory, so every export to a lattice or Stone-facing consumer must use those rows.

Apply Definition 3733.6. It makes the meet and join rows visible together with complement, endpoint, order, transport, replay, provenance, and local naming before any $\mathsf{Lattice}^{\uparrow}$ consumer uses the packet.

Apply Definition 3733.9 and Definition 3733.8. They send the same finite packet to $\mathsf{StoneDuality}^{\uparrow}$ and regular-open consumers after the public obligation surface has been exposed.

Finally use Definition 3733.7. It identifies carrier admission, complement endpoint exactness, finite meet and join laws, Stone-source routing, transport, replay, provenance, and naming as one package. Thus the mature export has no coordinate for a second host Boolean algebra, powerset

representation, selected ultrafilter, quotient lattice equality, detached locale frame, or ambient topology. □

Lean4 ✓ `BEDC.Derived.BooleanalgebraUp.BooleanAlgebraMatureExport`

Chapter 3734

A Concrete Naming Certificate for $\mathsf{CoverageRankAlgebra}^{\uparrow}$

The $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet realizes the coverage-rank algebra of Definition 9462.1576 as finite BEDC NameCert data. It consumes the six-boundary tile coverage route of chapter 3470, the Boolean trigger lattice of chapter 3704, the window coverage-trigger algebra of chapter 3478, and the finite observation fold of chapter 2236. The packet binds boundary-tile coverage, rank, trigger support, classifier stability, replay, provenance, and local naming rows; it is not a biological code oracle, stochastic transition model, quotient codon space, hidden global support lattice, or infinite trigger search.

Definition 3734.1 (Coverage-rank algebra carrier)**.** *A* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, R, T, L, F, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the consumed* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row exposing the six boundary-tile coverage coordinates and their visible support faces.*
2. *R is the rank row, recording the finite support-union rank read from the coverage-rank algebra vision.*
3. *T is the boundary-trigger support row, carrying the visible trigger support whose equality or inequality is tested by the classifier.*
4. *L is the consumed* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *row naming Boolean meet, join, and support-rank comparisons over the same visible boundary supports.*
5. *F is the consumed* $\mathsf{FiniteObservationFold}^{\uparrow}$ *row that bounds every algebra read by a displayed finite observation fold and its omitted-tail ledger.*
6. *Q is the consumed* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *row, carrying the finite closure, blocker, threshold, and residue-facing algebra rows used by downstream trigger consumers.*
7. *H records componentwise* hsame *transport for coverage, rank, trigger support, Boolean lattice, finite fold, trigger algebra, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-trigger coverage request through B, R, T, L, F, Q, H.*

9. *P is the* Pkg *provenance over* $\mathsf{CoverageRankAlgebra}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{FiniteObservationFold}^{\uparrow}$, *the source vision of Definition 9462.1576,* BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, T, L, F, Q, H, C, P, N$. It has no coordinate for hidden biological semantics, stochastic dynamics, quotient codon equality, global support completion, host rank lattices, or infinite trigger enumeration.

Theorem 3734.2 (Coverage-rank algebra NameCert obligations)**.** *For every accepted* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *carrier $K = (B, R, T, L, F, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$ *surface consists exactly of boundary-tile coverage through B, rank exposure through R, boundary-trigger support through T, Boolean trigger-lattice comparison through L, finite observation folding through F, coverage-trigger algebra handoff through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The proposed Lean follow-up for `BEDC.Derived.CoverageRankAlgebraUp` *must provide explicit* Nontrivial $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *and* FieldFaithful $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *instances preserving $B, R, T, L, F, Q, H, C, P, N$ as separate displayed fields.*

Proof. Project the carrier of Definition 3734.1. The rows B, R, T, L, F, Q expose boundary-tile coverage, rank, trigger support, Boolean comparison, finite observation folding, and trigger-algebra handoff before any algebra consumer can read a coverage-rank conclusion.

The classifier is rank-sensitive but support-visible. Equal-rank reads are admitted only through the displayed trigger support row T and the Boolean comparison row L, so a consumer cannot collapse unequal boundary-trigger supports into one stable classifier row.

The finite observation fold F bounds every read, and Q supplies only the already displayed coverage-trigger algebra route. The rows H, C, P, N preserve, replay, source, and name this finite route, leaving hidden biological semantics, stochastic dynamics, quotient codon equality, global support completion, host rank lattices, and infinite trigger enumeration outside the carrier. □

Theorem 3734.3 (Coverage-rank boundary-trigger separation)**.** *Let $K = (B, R, T, L, F, Q, H, C, P, N)$ and $K' = (B', R', T', L', F', Q', H', C', P', N')$ be accepted* $\mathsf{CoverageRankAlgebra}^{\uparrow}$ *packets. If the rank rows R and R' display the same finite rank but the boundary-trigger support rows T and T' expose different support, then a classifier-stable identification of K and K' is refused unless it also supplies a displayed transport row through L, L', F, F', Q, Q' preserving the unequal support evidence. Thus equal rank alone never determines equality of the coverage-rank algebra packet.*

Lean4 ✓ `BEDC.Derived.CoverageRankAlgebraUp.CoverageRankAlgebraBoundaryTriggerSeparation`

Proof. Read the two carriers. The rank rows R, R' and trigger-support rows T, T' are separate coordinates, so equality of rank does not erase the displayed support data.

The Boolean lattice and finite-fold rows L, L', F, F' are the only rows that compare support and bound the observation. A classifier-stable identification must therefore preserve these rows before the trigger-algebra handoff can be read.

Finally $Q, Q', H, H', C, C', P, P', N, N'$ replay and name the same finite route. Since no host rank lattice or hidden global support completion is a carrier coordinate, equal rank with unequal boundary-trigger support remains separated at the packet boundary. □

Falsifiable prediction: Any accepted $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet that identifies equal rank with unequal boundary-trigger support without displaying the Boolean trigger-lattice, finite observation fold, coverage-trigger algebra, transport, replay, provenance, and local naming rows violates classifier stability.

Independence witness: six-boundary-tile-coverage, boolean-trigger-lattice, window-coverage-trigger-algebra, finite-observation-fold: $\mathsf{CoverageRankAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries boundary-tile coverage, rank, boundary-trigger support, Boolean comparison, finite observation fold, coverage-trigger algebra handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{CoverageRankAlgebra}^{\uparrow}$ packet is generated from six-boundary tile coverage rows, rank rows, boundary-trigger support rows, Boolean trigger-lattice comparison rows, finite observation fold rows, coverage-trigger algebra rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3734.1, Definition 3734.2, and Definition 3734.3 seed the finite coverage-rank algebra packet over boundary-tile coverage, rank, trigger support, classifier, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoverageRankAlgebra}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CoverageRankAlgebraUp.CoverageRankAlgebraBoundaryTriggerSeparation`

Bridge status: none

Not claimed: This chapter does not close biological-code truth, stochastic transition laws, quotient codon equality, hidden global support lattices, host rank lattices, infinite trigger search, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rank exposure, boundary-trigger separation, classifier stability, Nontrivial CoverageRankAlgebraUp, and FieldFaithful CoverageRankAlgebraUp for BEDC.Derived.CoverageRankAlgebraUp.

Chapter 3735

A Concrete Naming Certificate for $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ packet records the finite boundary where an ordered Cauchy sequence can be read as a completion-facing real-order object without quotient equality, selected limits, or an ambient completeness primitive. It sits over the ordered sequence route of chapter 3636, the algebra route of chapter 3708, the regular rational handoff of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the complete-metric boundary of chapter 128. The packet names the ordered Cauchy completion boundary as displayed BEDC data; it is not a quotient completion, not a selected representative, and not a host real-order completeness theorem.

Definition 3735.1 (Cauchy sequence order completeness carrier)**.** *A* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, A, R, S, D, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the* ***??*** *row, carrying the finite order-comparison requests and monotone-tail witnesses available to the completion boundary.*
2. *A is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, carrying the finite algebraic closure operations used by the same tail windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the ordered algebraic Cauchy sequence data.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the inspected sequence entries and comparison windows are read.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, split tolerances, and comparison budgets used by O, A, R, S.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback and dyadic ledger have been displayed.*
7. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row, recording the finite completion-facing metric handoff visible to consumers after O, A, R, S, D, E.*

8. *H stores componentwise* hsame *transport for the order, algebra, regular readback, stream window, dyadic ledger, real seal, complete-metric boundary, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from ordered Cauchy requests through algebraic closure, stream windows, regular rational readback, dyadic comparison, real sealing, and the complete-metric handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$, *??*, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $O, A, R, S, D, E, M, H, C, P, N$. It contains no coordinate for quotient completion equality, selected representatives, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a bridge-checked standard completion object.*

Theorem 3735.2 (Cauchy sequence order completeness NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier $K = (O, A, R, S, D, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed order, algebra, regular readback, stream-window, dyadic tolerance, real-seal, complete-metric boundary, transport, replay, provenance, and local naming rows;*
2. *order-algebra compatibility, requiring every completion-facing order comparison to read the same ?? row O and* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row A before any regular rational readback is exposed;*
3. *regular-name handoff, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to be generated from the displayed stream windows S and dyadic tolerance ledger D;*
4. *real-seal and metric-boundary exposure, requiring the* $\mathsf{Real}^{\uparrow}$ *row E and* $\mathsf{CompleteMetric}^{\uparrow}$ *row M to appear only after O, A, R, S, D have been transported through H and replayed through C;*
5. *ledger nonescape, excluding quotient completion equality, selected representatives, countable choice, host real equality, ambient real completeness, and bridge checking from the local carrier.*

Proof. Project the carrier of Definition 3735.1. The accepted inventory is exactly $O, A, R, S, D, E, M, H, C, P, N$, so carrier admission and classifier comparison are finite.

The order and algebra rows are read together. A completion-facing order request cannot inspect the regular rational row until the displayed order row O, algebra row A, stream windows S, and dyadic comparison ledger D have been exposed.

The real seal and complete-metric boundary are downstream. The row E seals the regular rational readback through $\mathsf{Real}^{\uparrow}$, while M records only the finite completion-facing metric handoff after the same route has been transported and replayed.

Finally inspect the excluded coordinates. Since the carrier has no quotient equality, selected representative, countable-choice, host equality, ambient-completeness, or bridge-checking row, the local NameCert cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.CauchySequenceOrderCompletenessUp.CauchySequenceOrderCompletenessNamecertObligatio`

Theorem 3735.3 (Cauchy sequence order completeness nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ *carrier must traverse the displayed route*

??, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame,

before it can expose an ordered completion boundary. The route exports no quotient completion, no selected limit representative, no countable-choice schedule, no host real equality, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no bridge claim.*

Proof. Begin with an accepted carrier $K = (O, A, R, S, D, E, M, H, C, P, N)$. The only order source is O, and the only algebraic closure source is A; both are upstream of the finite stream windows and regular rational readback.

Read S, R, D next. These rows expose finite observations, regular rational data, and dyadic tolerance budgets before any real or metric boundary can be inspected.

The terminal rows E and M are reached only through the componentwise transports H and displayed replay C. The provenance row P and local naming row N source exactly that route.

Since the carrier inventory has no quotient, selector, choice, host equality, ambient completeness, or bridge coordinate, every ordered completion read remains inside the displayed finite BEDC route. □

Falsifiable prediction: Every $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ completion-facing read factors through displayed CauchySequenceOrder, CauchySequenceAlgebra, StreamName, RegSeqRat, DyadicRatCore, RealUp, CompleteMetric, transport, replay, provenance, and local naming rows before exposing an ordered completion boundary.

Independence witness: cauchy-sequence-order, cauchy-sequence-algebra, regseqrat, streamname, dyadicratcore, real, completemetric: $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ordered tail comparisons, algebraic Cauchy closure, regular rational readback, finite stream windows, dyadic tolerance rows, a Real seal, and a complete-metric handoff in one finite completion-boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}$ packet is generated from CauchySequenceOrder and CauchySequenceAlgebra rows, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a CompleteMetric boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3735.1, Definition 3735.2, and Definition 3735.3 seed the finite ordered Cauchy completion boundary over order, algebra, regular rational, stream, dyadic, real, and complete-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceOrderCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order-algebra compatibility, regular-name handoff, real-seal exposure, complete-metric boundary exposure, and ledger nonescape for BEDC.Derived.CauchySequenceOrderCompletenessUp.

Chapter 3736

A Concrete Naming Certificate for $\mathsf{HilleYosida}^{\uparrow}$

Following Hille and Yosida, the $\mathsf{HilleYosida}^{\uparrow}$ packet records the finite BEDC surface by which a densely read linear generator graph is accepted as strongly continuous contraction-semigroup generator data. It consumes the functional-analytic source of chapter 94, the Banach carrier of chapter 90, the linear graph rows of chapter 48, the norm ledgers of chapter 89, and the complete-metric convergence surface of chapter 128. The packet names the generator and resolvent route used by later evolution equations; it is not a spectral theorem, a PDE solver, a selected semigroup object outside the displayed rows, or a global regularity principle.

Definition 3736.1 (Hille–Yosida carrier)**.** *A* $\mathsf{HilleYosida}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (B, A, R, S, N, H, C, P, L)$$

with the following visible rows:

1. *B is the* $\mathsf{Banach}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *source row, recording the ambient vector space, normed structure, and complete state classifier in which generator reads are interpreted.*
2. *A is the linear generator graph row, a* $\mathsf{LinMap}^{\uparrow}$*-facing finite graph with its displayed domain row and graph-compatibility observations.*
3. *R is the positive-parameter resolvent ledger, recording the displayed $\lambda > 0$ requests, resolvent graph rows, domain-return rows, and resolvent identity comparisons used by the generator test.*
4. *S is the contraction semigroup row, recording the finite time-parameter observations, identity-at-zero row, composition replay row, and contraction comparisons visible to a strongly continuous semigroup consumer.*
5. *N is the norm-bound ledger, joining the* $\mathsf{Norm}^{\uparrow}$ *contraction bounds for S and the positive-parameter resolvent bounds for R.*
6. *H records componentwise* hsame *transport for the Banach source, generator graph, resolvent ledger, semigroup row, norm bounds, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from finite positive-parameter and finite time requests through*

$$B, A, R, N \longmapsto S, H, P, L.$$

8. *P is the* Pkg *provenance over* HilleYosida$^{\uparrow}$, FunctionalAnalysis$^{\uparrow}$, Banach$^{\uparrow}$, LinMap$^{\uparrow}$, Norm$^{\uparrow}$, CompleteMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* NameCert$_{\text{HilleYosida}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a chosen global semigroup outside S, a spectral calculus, differentiability beyond the displayed graph, maximal PDE regularity, excluded middle, quotient equality, or choice-selected limits.

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaTasteGate single carrier alignment`

Theorem 3736.2 (Hille–Yosida NameCert obligations)**.** *For an accepted* HilleYosida$^{\uparrow}$ *carrier*

$$Y = (B, A, R, S, N, H, C, P, L),$$

the local NameCert$_{\text{HilleYosida}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Banach and functional-analysis source compatibility for B, linear generator graph exposure for A, positive-parameter resolvent accounting for R, contraction semigroup visibility for S, norm-bound accounting for N, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through L.*

Proof. Carrier admission is projection from Definition 3736.1. The accepted packet displays the Banach source, generator graph, resolvent ledger, semigroup row, norm ledger, transport row, replay row, provenance row, and local name.

The analytic obligations are read in the displayed order. The source row B fixes the complete normed state space, the generator row A supplies the linear graph, and the resolvent row R records the positive-parameter reads whose estimates are accounted for by N.

The semigroup-facing obligation is then restricted to S. A consumer may read finite time observations, the identity-at-zero row, composition replay, and contraction comparisons, but it cannot import an additional selected semigroup object.

Finally H, C, P, L transport, replay, source, and name exactly the same route. Since no row stores spectral calculus, maximal PDE regularity, quotient equality, excluded middle, or choice-selected limits, those principles are outside the local certificate surface. □

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaCarrier namecert obligations`

Theorem 3736.3 (Hille–Yosida semigroup generator handoff)**.** *Every strongly continuous semigroup generator read accepted by a* HilleYosida$^{\uparrow}$ *carrier factors through*

FunctionalAnalysis$^{\uparrow}$, Banach$^{\uparrow}$, LinMap$^{\uparrow}$, Norm$^{\uparrow}$, CompleteMetric$^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

Equivalently, an evolution-equation consumer may inspect the Banach source B, generator graph A, positive-parameter resolvent ledger R, contraction semigroup row S, norm-bound ledger N, and structural rows H, C, P, L, but it cannot bypass them by selecting a global semigroup, invoking spectral calculus, or reading a host completion principle not displayed by the packet.

Lean4 ✓ `BEDC.Derived.HilleYosidaUp.HilleYosidaSemigroupGeneratorHandoff`

Proof. Use Definition 3736.2. It exposes a generator-facing read only after B, A, R, S, N have been projected from the finite carrier.

The resolvent side is controlled by R and N. Positive-parameter requests, domain-return rows, identity comparisons, and norm bounds are visible only as finite ledger entries in those coordinates.

The semigroup side is controlled by S. Strong continuity and contraction information are read as the displayed time-row observations and replay rows, not as an external choice of a completed semigroup.

The structural rows H, C, P, L then preserve the same data under transport, replay, provenance, and local naming. Thus a later ODE or PDE evolution packet receives precisely the finite generator handoff and no stronger global regularity or spectral conclusion. □

Falsifiable prediction: Every $\mathsf{HilleYosida}^{\uparrow}$ generator handoff read factors through one Banach and FunctionalAnalysis source row, one linear generator graph row, one positive-parameter resolvent ledger, one contraction semigroup row, one norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: functionalanalysis, banach, linearmap, norm, completemetric: $\mathsf{HilleYosida}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a generator graph, positive-parameter resolvent ledger, contraction semigroup row, norm-bound ledger, and the transport/replay/provenance rows tying them into a semigroup-generator certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3736.1, Definition 3736.2, and Definition 3736.3 seed the finite generator certificate from Banach and functional-analysis source rows through linear graph, resolvent, semigroup, norm-bound, transport, replay, provenance, and local naming data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilleYosida}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral calculus, maximal PDE regularity, a selected global semigroup outside the displayed row, quotient equality, choice principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, generator graph exposure, resolvent positive-parameter accounting, contraction semigroup visibility, norm-bound stability, complete-metric convergence visibility, transport stability, replay stability, and ledger non-escape for BEDC.Derived.HilleYosidaUp.

Constructive story: A $\mathsf{HilleYosida}^{\uparrow}$ packet is generated from one Banach and FunctionalAnalysis source row, one LinearMap generator graph, one positive-parameter resolvent ledger, one contraction semigroup row, one Norm ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3737

A Concrete Naming Certificate for $\mathsf{CompactUniformOscillation}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor finite-net route, the CompactUniformOscillatio packet records the finite data by which compact metric source rows and uniform-modulus rows produce bounded target-distance oscillation witnesses. It consumes the compact metric source of chapter 129, the continuous-map graph of chapter 130, the uniform-modulus ledger of chapter 369, and the target distance surface of chapter 57. The packet is a finite oscillation-bound consumer for compact-uniform-continuity routes; it is not an open-cover compactness principle, a selected supremum, host extensionality of functions, or a global image-compactness theorem.

Definition 3737.1 (Compact uniform oscillation carrier)**.** *A* $\mathsf{CompactUniformOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, M, D, W, H, C, P, L)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, recording the compact finite-net observations and metric comparison rows on the source space.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the finite graph evaluations and local continuity reads used by the compact-uniform route.*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger, recording tolerance requests, source-radius responses, and the modulus rows obtained from compact finite nets.*
4. *D is the target-distance row, a* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{RealMetric}^{\uparrow}$ *readback ledger for the distances between displayed output observations.*
5. *W is the finite* ProbeBundle *oscillation-witness row, recording source net points, paired source observations within the modulus radius, and the resulting target-distance bounds.*
6. *H records componentwise* hsame *transport for the compact source, continuous graph, uniform modulus, target-distance ledger, oscillation witnesses, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from each finite target-oscillation request through*

$$K, F, M, D, W \longmapsto H, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformOscillation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.

9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected supremum of image diameters, open-cover compactness, quotient equality of maps, choice-selected net refinements, trichotomy, or host function extensionality.

Lean4 ✓ `BEDC.Derived.CompactUniformOscillationUp.CompactUniformOscillationTasteGate single carrier alignment`

Theorem 3737.2 (Compact uniform oscillation NameCert obligations)**.** *For an accepted* CompactUniformOscillatio *carrier*

$$O = (K, F, M, D, W, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, compact finite-net exposure for K, continuous graph exposure for F, uniform-modulus accounting for M, target-distance accounting for D, finite oscillation-witness exposure for W, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Carrier admission is projection from Definition 3737.1. The accepted packet displays the compact metric source, continuous-map graph, uniform-modulus ledger, target-distance ledger, finite witness row, transport row, replay row, provenance row, and local name.

The compact-uniform route is read from K, F, M. A consumer first receives finite compact-net observations, then the displayed continuous graph rows, and then the tolerance-only modulus ledger built over those observations.

The oscillation estimate is read from D and W. Target-distance comparisons are visible only through D, while W stores the finite witness pairs whose source separation is small enough for the modulus row.

Finally H, C, P, L transport, replay, source, and name the same route. Because no coordinate stores a selected supremum, open-cover compactness, quotient equality, choice-selected refinements, trichotomy, or host function extensionality, those principles are outside the local certificate surface. □

Theorem 3737.3 (Compact uniform oscillation modulus handoff)**.** *Every bounded image-diameter or equicontinuity-facing read accepted by a* $\mathsf{CompactUniformOscillation}^{\uparrow}$ *carrier factors through*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, NameCert.

Equivalently, a downstream consumer may inspect compact finite-net data K, continuous graph data F, the uniform-modulus ledger M, target-distance rows D, finite oscillation witnesses W, and structural rows H, C, P, L, but it cannot bypass them by selecting a supremum, importing open-cover compactness, or comparing maps by host extensionality.

Proof. Apply Definition 3737.2. It forces every oscillation-facing request to open K, F, M, D, W before the structural rows are available.

The finite-net part is controlled by K and M. Compactness is consumed as displayed net data and a tolerance-only modulus ledger, not as an external open-cover principle.

The target estimate is controlled by D and W. The witness row stores the paired observations whose target distances are bounded, and the distance ledger records the corresponding metric readback.

Transport, replay, provenance, and naming through H, C, P, L preserve exactly that route. Hence later bounded-image or equicontinuity packets receive a finite oscillation-bound handoff and no selected supremum or host-level function equality. □

Falsifiable prediction: Every $\mathsf{CompactUniformOscillation}^{\uparrow}$ oscillation read factors through one CompactMetric finite-net source row, one ContinuousMap graph row, one UniformModulus ledger, one target Metric or RealMetric distance row, one finite ProbeBundle oscillation-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, metric, realmetric: $\mathsf{CompactUniformOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net data, continuous graph data, a uniform-modulus ledger, target-distance readback, finite oscillation witnesses, and the transport/replay/provenance rows tying them into an oscillation-bound packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3737.1, Definition 3737.2, and Definition 3737.3 seed the finite compact-uniform oscillation packet over compact metric source rows, continuous graph rows, uniform-modulus ledgers, target-distance readbacks, finite witnesses, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected suprema, open-cover compactness, image compactness, host function extensionality, choice-selected net refinements, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous graph exposure, uniform-modulus accounting, target-distance accounting, oscillation-witness stability, transport stability, replay stability, and ledger non-escape for BEDC.Derived.CompactUniformOscillationUp.

Constructive story: A $\mathsf{CompactUniformOscillation}^{\uparrow}$ packet is generated from one CompactMetric finite-net source row, one ContinuousMap graph row, one UniformModulus ledger, one target-distance row, one finite ProbeBundle oscillation-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3738

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$

Following Bishop and Bridges on diagonal arguments in constructive analysis, the $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ packet names the finite window data that a Real or Completion route may inspect before it reads a diagonal regular Cauchy seal. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the downstream diagonal extraction route of chapter 1773. The packet records the named diagonal-window boundary; it is not a quotient stream, not a selected limit, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3738.1 (Regular Cauchy diagonal window carrier)**.** *A* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row containing the requested dyadic radius, split-tolerance ledger, and comparison budget for the diagonal window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite stream-window row that displays the source entries available to the diagonal read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to the same window and tolerance budget.*
4. *I is the finite diagonal index schedule, recording only the named indices whose windows are inspected by the consumer.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after D, S, R, I have been displayed.*
6. *H records componentwise* hsame *transport for the dyadic tolerance, stream window, regular readback, diagonal index schedule, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal-window request through the tolerance row D, stream window S, regular readback R, index schedule I, and Real seal E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, I, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalWindowUp.RegularCauchyDiagonalWindowTasteGate single carrier alignment`

Theorem 3738.2 (Regular Cauchy diagonal window obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalWindo}$ *carrier $W = (D, S, R, I, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance row, stream window, regular readback, diagonal index schedule, Real seal, transport, replay, provenance, and local naming row;*
2. *diagonal-window visibility, requiring each consumer-facing diagonal request to read a displayed finite window before any Real seal is exposed;*
3. *dyadic tolerance exactness, requiring the requested tolerance and every split tolerance to be read through D;*
4. *regular readback compatibility, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R to consume the same finite window S and diagonal schedule I named by the carrier;*
5. *Real-seal nonescape, requiring E to appear only after D, S, R, I have been transported through H and replayed through C;*
6. *transport, replay, provenance, and local naming for exactly the displayed rows $D, S, R, I, E, H, C, P, N$.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3738.1. The accepted inventory is exactly the dyadic tolerance row D, stream window S, regular readback R, diagonal index schedule I, Real seal E, and the structural rows H, C, P, N.

The diagonal-window obligation is therefore visible before the seal. A consumer request first reads the dyadic tolerance and split budget in D, then the finite stream window S, and then the regular rational readback R attached to the displayed schedule I.

The Real seal is downstream of that finite route. Componentwise hsame transport in H preserves the displayed rows, and Cont replay in C repeats the same ordered route before E can be inspected.

Finally, provenance and naming are exhausted by P and N. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness, no such obligation can be exported by the local certificate. □

Theorem 3738.3 (Regular Cauchy diagonal window nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

together with the finite diagonal index schedule. The packet exports no quotient stream, selected limit, countable-choice schedule, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted carrier $W = (D, S, R, I, E, H, C, P, N)$. The only tolerance source is D, the only finite source-window row is S, and the only regular rational readback row is R.

The diagonal schedule I is finite and displayed. It selects only the named window entries already present in S and the tolerance budgets already present in D before the readback reaches R.

The terminal Real seal E is reached only through the componentwise transport H and displayed replay C. The provenance row P and local naming row N source exactly that route.

Since the carrier inventory has no quotient-stream, selected-limit, countable-choice, trichotomy, host-equality, or ambient-completeness coordinate, every Real or Completion read remains inside the displayed finite diagonal-window packet. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ Real or Completion read factors through displayed DyadicRatCore, StreamName, RegSeqRat, RealUp, transport, replay, provenance, local naming, and finite diagonal-index rows before exposing a diagonal seal.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-diagonal-extraction: $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance exactness, finite stream-window exposure, regular rational readback, a diagonal index schedule, and a Real seal in one finite window packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite window, RegSeqRat readback, a finite diagonal index schedule, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3738.1, Definition 3738.2, and Definition 3738.3 seed the finite diagonal-window packet used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyDiagonalWindowUp`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-window visibility, dyadic tolerance exactness, regular readback compatibility, Real-seal exposure, and ledger nonescape for BEDC.Derived.RegularCauchyDiagonalWindowUp.

Chapter 3739

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy real-completion route may select a diagonal modulus only after the source regular sequence, dyadic error ledger, observation windows, and terminal real handoff are all visible. The $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ packet records that finite L10 selection surface. It sits after the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic ledger of chapter 291, the finite window rows of chapter 126, and the minimum-modulus packet of chapter 5544, before a $\mathsf{Real}^{\uparrow}$ consumer reads the terminal handoff of chapter 38. The packet is a displayed diagonal schedule and selected-modulus ledger, not quotient equality of Cauchy names, a hidden limit selector, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3739.1 (Regular Cauchy diagonal modulus selection carrier)**.** *A* RegularCauchyDiagonalModulusSelectio *carrier is a finite* BHist *packet*

$$D = (S, E, W, \Delta, M, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational sequence whose diagonal requests are being inspected.*
2. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error row recording the displayed dyadic tolerances attached to those requests.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row listing the finite source windows, target precisions, and inspected coordinates.*
4. *Δ is the diagonal schedule row, naming the finite diagonal order in which source windows and error bounds are paired.*
5. *M is the selected modulus row. It records the finite modulus chosen from the displayed diagonal schedule and the minimum-modulus surface of chapter 5544.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after S, E, W, Δ, M have been displayed.*
7. *H records componentwise* hsame *transport for source, error, observation, diagonal schedule, selected modulus, Real handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a real-completion request through $S, E, W, \Delta, M, L, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyMinModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, W, \Delta, M, L, H, C, P, N$. It has no coordinate for quotient equality of Cauchy names, a hidden diagonal selector, countable choice over infinitely many windows, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3739.2 (Regular Cauchy diagonal modulus selection obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ *carrier*

$$D = (S, E, W, \Delta, M, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *error row,* $\mathsf{StreamName}^{\uparrow}$ *observation row, diagonal schedule row, selected modulus row,* $\mathsf{Real}^{\uparrow}$ *handoff row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from quotient equality, hidden selectors, countable choice, host real equality, and ambient completion.*

Proof. Project the carrier of Definition 3739.1. The source, error, and observation rows are S, E, W, so every diagonal request is attached to a displayed regular-rational source, dyadic tolerance ledger, and finite observation window.

The diagonal schedule is Δ, and the selected modulus is M. The selected row cites the minimum-modulus packet of chapter 5544, so it records a finite visible budget rather than an off-packet selector.

The Real-facing row is L. It is reachable only after the source, dyadic, observation, schedule, and selected-modulus rows have been exposed, matching the L10 route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded object is a carrier coordinate, the displayed rows exhaust the NameCert obligations. □

Theorem 3739.3 (Regular Cauchy diagonal modulus Real handoff)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *consumer read of a* $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegularCauchyMinModulus}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

on the displayed rows $S, E, W, \Delta, M, L, H, C, P, N$. The handoff exports no quotient-name equality, no selected representative, no host real equality, no countable-choice row, and no ambient completeness row.

Proof. Start with an accepted carrier $D = (S, E, W, \Delta, M, L, H, C, P, N)$. The route to a Real consumer begins with the visible regular-rational source S, dyadic error row E, and stream observation row W.

Apply the obligation theorem above. It forces the diagonal schedule Δ and selected modulus M to remain attached to the same displayed source, error, and observation rows before L is readable.

The terminal handoff row L is therefore a Real-facing seal over the same finite packet. Transport through H, replay through C, provenance through P, and local naming through N preserve that route and do not add any quotient, selected-representative, choice, host-equality, or ambient-completeness coordinate. □

Falsifiable prediction: A real-completion read that requests a diagonal modulus without displaying a $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{DyadicRatCore}^{\uparrow}$ error row, $\mathsf{StreamName}^{\uparrow}$ observation row, diagonal schedule row, selected modulus row, $\mathsf{Real}^{\uparrow}$ handoff row, hsame transport, Cont replay, Pkg provenance, and local NameCert row is rejected by the $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ classifier.

Independence witness: regseqrat, dyadicratcore, streamname, regularcauchyminmodulus, real: no listed sibling is bijective to $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ because this packet jointly carries a regular-rational source, dyadic error ledger, finite observation windows, diagonal schedule, selected modulus, Real handoff, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}$ packet is generated from a RegSeqRat source row, DyadicRatCore error row, StreamName observation row, diagonal schedule row, selected modulus row, RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3739.1, Definition 3739.2, and Definition 3739.3 seed the finite diagonal modulus selection route consumed by L10 real-completion readers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of Cauchy names, hidden limit selection, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, dyadic error exactness, finite observation windows, diagonal schedule exposure, selected-modulus exactness, Real handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularCauchyDiagonalModulusSelectionUp.

Chapter 3740

A Concrete Naming Certificate for DunfordFunctionalCalculus$^{\uparrow}$

Following Dunford and Schwartz on contour functional calculus for bounded operators, the DunfordFunctionalCalculu packet records the finite NameCert route by which a holomorphic scalar window is read through resolvent rows before any spectral theorem or projection-valued measure is public. It sits beside the Banach-algebra source of chapter 2189, the bounded-operator carrier of chapter 1644, the resolvent identity packet of chapter 3715, the spectral-radius boundary of chapter 2352, and the compact-operator schedule of chapter 1952.

Definition 3740.1 (Dunford functional calculus carrier)**.** *A* DunfordFunctionalCalculus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the* BanachAlgebra$^{\uparrow}$ *source row in which products, scalar multiplication, and norm estimates are read.*
2. *T is the* BoundedLinearOperator$^{\uparrow}$ *row naming the operator whose calculus is being inspected.*
3. *Γ is the finite contour-window row. It records the oriented contour pieces, mesh endpoints, denominator separation witnesses, and enclosure data used by the accepted request.*
4. *Ω is the holomorphic scalar-window row. It records the finite function values and local compatibility checks visible on the contour windows.*
5. *R is the* ResolventIdentity$^{\uparrow}$ *row. It supplies the displayed shifted-operator inverse checks and two-shift identity ledger used along Γ.*
6. *I is the contour-integral ledger. It records only finite Riemann-sum windows, dyadic tolerance budgets, and composition order through A, T, Γ, Ω, R.*
7. *Q is the spectral-radius and compact-operator boundary row. It allows later spectral consumers to read enclosure and compact-image schedules, but only as rows already displayed by chapter 2352 and chapter 1952.*
8. *S is the terminal calculus handoff row naming the value of the accepted contour read as a bounded-operator-facing packet.*

9. *H records componentwise* hsame *transport for the Banach-algebra, operator, contour, scalar-window, resolvent, integral-ledger, spectral-boundary, handoff, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from contour requests through $A, T, \Gamma, \Omega, R, I, Q, S, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{DunfordFunctionalCalculus}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a spectral projection, a projection-valued measure, quotient operator equality, a selected spectrum, countable choice, or a host holomorphic functional calculus outside the displayed contour packet.

Lean4 ✓ `BEDC.Derived.DunfordFunctionalCalculusUp.DunfordFunctionalCalculusTasteGate single carrier alignment`

Theorem 3740.2 (Dunford functional calculus NameCert obligations)**.** *For every accepted* DunfordFunctionalCalcu *carrier*

$$F = (A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Banach-algebra source compatibility through A, bounded-operator source compatibility through T, contour-window admission through Γ, holomorphic scalar-window admission through Ω, resolvent identity exactness through R, finite contour-ledger exactness through I, spectral-radius and compact-operator boundary visibility through Q, terminal calculus handoff through S, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every calculus-facing consumer must read the same rows before using the contour value. It receives no spectral projection, no spectral measure, no selected spectrum, no quotient equality of operators, no countable-choice schedule, and no ambient host functional calculus.

Proof. Carrier admission is projection from Definition 3740.1. The accepted packet displays exactly $A, T, \Gamma, \Omega, R, I, Q, S, H, C, P, N$.

The source rows A and T fix the Banach-algebra and bounded-operator surfaces. The contour and scalar-window rows Γ, Ω then provide the finite requests on which the calculus read is allowed to inspect function values.

The resolvent row R supplies only the displayed inverse checks and two-shift identity ledger. The contour ledger I reads those rows through finite Riemann-sum and dyadic-budget windows, so it cannot import a host integral or an unrecorded inverse.

The spectral-boundary row Q and handoff row S are downstream of the same route. Transport, replay, provenance, and local naming in H, C, P, N preserve that finite packet. Since the carrier has no coordinate for spectral projections, spectral measures, selected spectra, quotient equality, countable choice, or a host calculus, the listed rows exhaust the local obligations. □

Theorem 3740.3 (Dunford functional calculus resolvent-contour handoff)**.** *Every bounded-operator or spectral-analysis consumer of an accepted* $\mathsf{DunfordFunctionalCalculus}^{\uparrow}$ *carrier reaches the calculus value only through the displayed route*

$$A, T \longmapsto \Gamma, \Omega \longmapsto R \longmapsto I \longmapsto Q, S \longmapsto H, C, P, N.$$

Thus the consumer may read Banach-algebra operations, the bounded operator, contour windows, holomorphic scalar windows, resolvent identity rows, finite contour-integration budgets, spectral-radius and compact-operator boundary rows, the terminal handoff, transport, replay, provenance, and local naming, and it may not read spectral projections, spectral measures, quotient operator equality, a selected spectrum, countable choice, or a host functional calculus.

Proof. Begin with a calculus-facing consumer. The only operator sources in the carrier are A and T, and the only contour-facing source rows are Γ and Ω.

The resolvent dependency is forced through R. It supplies the two-sided inverse checks and identity ledger already isolated by the $\mathsf{ResolventIdentity}^{\uparrow}$ packet, so no separate selected inverse or quotient operator comparison is available.

The finite contour read is then the row I. It records the Riemann-sum windows, dyadic tolerance budgets, and composition order needed to pass from the contour and resolvent rows to the handoff.

Finally Q, S, H, C, P, N expose the spectral boundary, terminal bounded-operator-facing result, transport, replay, provenance, and local naming. The excluded projection, spectral-measure, selected-spectrum, choice, quotient, and host-calculus objects have no carrier coordinate and therefore cannot be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3740.1, Definition 3740.2, and Definition 3740.3 seed the contour functional-calculus route over Banach-algebra, bounded-operator, contour-window, holomorphic scalar-window, resolvent, spectral-boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordFunctionalCalculus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral projections, spectral measures, selected spectra, quotient operator equality, countable choice, host holomorphic functional calculus, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contour-window admission, scalar-window admission, resolvent exactness, contour-ledger exactness, spectral-boundary visibility, handoff non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DunfordFunctionalCalculusUp.

Constructive story: A $\mathsf{DunfordFunctionalCalculus}^{\uparrow}$ packet is generated from a BanachAlgebra source row, a BoundedLinearOperator row, finite contour windows, holomorphic scalar windows, a ResolventIdentity route, a finite contour ledger, SpectralRadius and CompactOperator boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3741

A Concrete Naming Certificate for $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$

Following Kelley on uniform spaces and Bishop–Bridges on completion by explicit regular data, the $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ packet records the finite common-window route by which uniform-completion requests share one observation window before any $\mathsf{Real}^{\uparrow}$ seal is read. It concretizes the Real and Completion vision of chapter 9238 by tying the $\mathsf{UniformCompletionFunctor}^{\uparrow}$ surface to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows without quotient completion equality. Its concrete dependency route reads the functor source of chapter 1892, the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic budget of chapter 291, and the terminal real-name seal of chapter 38.

Definition 3741.1 (Uniform completion common-window carrier)**.** *A* $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (U, E, S, R, D, L, F, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *source row naming the completion-facing request, entourage scale, and functorial handoff boundary.*
2. *E is the finite entourage-window row. It records the common uniform scale used by all source observations in the accepted request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *common-window row. It synchronizes the finite observation windows before any regular rational readback is exposed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source-readback row generated from the synchronized window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row, recording the dyadic tolerance, slack, comparison, and endpoint-conversion entries used by the readback.*
6. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is exposed only after S, R, D have displayed the shared window and regular rational readback.*
7. *F is the common-window functoriality row. It states that completion-facing consumers of U and Real-facing consumers of L read the same E, S, R, D route.*
8. *H records componentwise* hsame *transport for the uniform-completion source, common entourage window, StreamName window, RegSeqRat readback, DyadicRatCore budget, Real seal, functoriality, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a completion request through $U, E, S, R, D, L, F, H, P, N$.*
10. *P is the* Pkg *provenance over* UniformCompletionCommonWindow$^{\uparrow}$, UniformCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformCompletionCommonWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected completion points, host metric completeness, countable choice, or a second private observation window.

Lean4 ✓ `BEDC.Derived.UniformCompletionCommonWindowUp.UniformCompletionCommonWindowTasteGate`
`single carrier alignment`

Theorem 3741.2 (Uniform completion common-window NameCert obligations)**.** *For every accepted* UniformCompletionCommonWindow$^{\uparrow}$ *carrier*

$$W = (U, E, S, R, D, L, F, H, C, P, N),$$

the local NameCert$_{\text{UniformCompletionCommonWindow}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-completion source compatibility through U, common entourage-window admission through E, synchronized* StreamName$^{\uparrow}$ *admission through S,* RegSeqRat$^{\uparrow}$ *readback exactness through R,* DyadicRatCore$^{\uparrow}$ *budget exactness through D,* Real$^{\uparrow}$ *seal handoff through L, common-window functoriality through F, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Every completion-facing or Real-facing consumer must use the same E, S, R, D route. It receives no quotient completion equality, no selected completion point, no host metric-completeness principle, no countable-choice schedule, and no private window not displayed by the packet.

Lean4 ✓ `BEDC.Derived.UniformCompletionCommonWindowUp.UniformCompletionCommonWindowNameCertObligations`

Proof. Carrier admission is projection from Definition 3741.1. The displayed rows are exactly $U, E, S, R, D, L, F, H, C, P, N$.

The uniform-completion source row U is read first, and the common entourage window E fixes the finite scale for the request. The synchronized StreamName row S then supplies the only observation window available to the packet.

The regular rational and dyadic rows R, D are downstream of that shared window. The Real seal L can be exposed only after those rows display the readback and budget entries, so no consumer can substitute a quotient completion class or selected limit.

The row F ties the completion-facing and Real-facing reads to the same route. Transport, replay, provenance, and local naming in H, C, P, N preserve the route without adding host completeness, choice, or a second private window. □

Theorem 3741.3 (Uniform completion common-window Real seal handoff)**.** *Every* UniformCompletionFunctor$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* UniformCompletionCommonWindow$^{\uparrow}$ *carrier reaches the terminal Real seal only through*

$$U \longmapsto E \longmapsto S \longmapsto R, D \longmapsto L \longmapsto F, H, C, P, N.$$

Thus the completion request, common entourage window, synchronized StreamName window, RegSeqRat readback, DyadicRatCore budget, RealUp seal, common-window functoriality, transport, replay, provenance, and local naming are jointly visible, and quotient completion equality, selected completion points, host metric completeness, countable choice, and private observation windows are not exported.

Proof. Start with a completion-facing or Real-facing read. The source request is U, and the only uniform scale that may be consumed is the common row E.

The observation side is forced through S. Once that StreamName window is displayed, the regular rational readback R and dyadic budget D provide the finite data needed by the Real seal.

The seal row L is therefore downstream of the shared completion window, not of a quotient completion object. The functoriality row F records that both the completion consumer and the Real consumer read the same route.

Finally H, C, P, N transport, replay, source, and name the packet. Since no coordinate stores selected completion points, host completeness, choice, or private windows, the handoff exports only the displayed common-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3741.1, Definition 3741.2, and Definition 3741.3 seed the common-window route from UniformCompletionFunctor through StreamName, RegSeqRat, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionCommonWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completion points, host metric completeness, countable choice, private observation windows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common entourage-window admission, synchronized StreamName admission, RegSeqRat readback exactness, DyadicRatCore budget exactness, Real seal handoff, common-window functoriality, transport, replay, provenance, and local naming for BEDC.Derived.UniformCompletionCommonWindowUp.

Constructive story: A $\mathsf{UniformCompletionCommonWindow}^{\uparrow}$ packet is generated from one UniformCompletionFunctor request row, one common entourage-window row, one StreamName synchronized window, one RegSeqRat readback, one DyadicRatCore budget, one RealUp seal, a common-window functoriality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3742

A Concrete Naming Certificate for $\mathsf{FiniteMetricSpace}^{\uparrow}$

Following Burago–Burago–Ivanov and Gromov on finite metric spaces as completely displayed distance data, the $\mathsf{FiniteMetricSpace}^{\uparrow}$ packet records a metric carrier whose public observations are a finite point row and a finite distance table. It sits over the $\mathsf{Met}^{\uparrow}$ surface of chapter 57, the finite-net and total-bounded handoff of chapter 1767 and chapter 127, and the compact-metric consumer boundary of chapter 129. The packet names finite metric data without importing host finite sets, quotient point equality, selected centers, open-cover compactness, or ambient completion.

Definition 3742.1 (Finite metric space carrier)**.** *A* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, D, T, B, H, C, P, N)$$

with the following displayed rows:

1. *X is a finite point row, recorded as a closed* ProbeBundle *with point histories and membership witnesses for every public point read.*
2. *D is the finite distance-table row, assigning each displayed ordered pair of point histories a* $\mathsf{Real}^{\uparrow}$ *distance witness.*
3. *T is the metric-law ledger for the displayed table: zero diagonal, symmetry, triangle-budget rows, and positive-separation rows are read only as finite table entries.*
4. *B is the finite-net and boundedness handoff row. For each displayed positive tolerance it reuses the point row X as the center bundle and exposes the coverage route consumed by* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, and* $\mathsf{CompactMetric}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the point row, distance table, metric-law ledger, finite-net handoff, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from point membership reads to distance-table reads, from distance-table reads to metric-law checks, and from the finite point row to finite-net and compact-metric consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host finite set, powerset membership, quotient point equality, selected nearest-point search, open-cover compactness, or ambient completion.

Lean4 ✓ `BEDC.Derived.FiniteMetricSpaceUp.FiniteMetricSpaceTasteGate single carrier alignment`

Theorem 3742.2 (Finite metric space NameCert obligation surface)**.** *For every accepted* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *carrier* $F = (X, D, T, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite point row, distance table, metric-law ledger, finite-net handoff, componentwise transport, displayed replay, provenance, and local naming rows; distance readback through* D*; metric compatibility through* T*; finite-net and compact-metric handoff through* B*; and nonescape from host finite sets, powerset membership, quotient point equality, selected nearest-point search, open-cover compactness, and ambient completion.*

Proof. Project the carrier of Definition 3742.1. The displayed tuple exposes the point bundle, distance table, metric-law ledger, finite-net handoff, and the structural rows H, C, P, N.

The distance readback is table-local. Every distance observation names two point histories from X and then reads the corresponding entry of D, while T supplies the finite metric-law rows attached to those entries.

The finite-net handoff is the row B. It reuses the same point row as the center bundle and sends the resulting coverage data to finite-net, total-bounded, and compact-metric consumers through the displayed replay route.

No excluded object is a coordinate of the carrier. Thus the local certificate exports only the displayed finite metric route and its structural transport, replay, provenance, and naming rows. □

Theorem 3742.3 (Finite metric space distance-table boundary)**.** *Every* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*, or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteMetricSpace}^{\uparrow}$ *packet reads metric information only through*

$$X \longmapsto D \longmapsto T \longmapsto B.$$

The consumer receives a finite point bundle, a displayed distance table, the metric-law ledger for that table, and the finite-net handoff generated from the same point bundle. It receives no host finite-set equality, ambient enumeration principle, selected nearest point, external compactness theorem, quotient completion, or hidden completion carrier.

Proof. Begin with a consumer request for metric or compact-metric data. The only public points are the histories displayed in X, and every distance read is routed through the table row D.

The metric-space laws are then read from T. Since T is attached to the same table entries, a consumer cannot replace the displayed table by a host metric or by a quotient comparison on points.

For finite-net and compact-metric use, the handoff row B supplies the center bundle and coverage route from X. The replay row C preserves the order from point membership to distance-table read to metric-law check to finite-net export.

The remaining rows transport, source, and name this same route. Because no carrier coordinate stores a host finite set, selected nearest point, external compactness theorem, quotient completion, or hidden completion carrier, every consumer boundary is exhausted by X, D, T, B together with H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteMetricSpace}^{\uparrow}$ packet is generated from a finite point row, a displayed Real-valued distance table, finite metric-law rows, a finite-net and compact-metric handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteMetricSpace}\uparrow}$, seed)

Scope closed: Definition 3742.1, Definition 3742.2, and Definition 3742.3 seed the finite distance-table packet and its finite-net, total-bounded, and compact-metric consumer boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteMetricSpace}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteMetricSpaceUp.FiniteMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host finite sets, quotient point equality, nearest-point selection, open-cover compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, table-local metric-law exactness, finite-net handoff stability, compact-metric boundary nonescape, and provenance exhaustion for BEDC.Derived.FiniteMetricSpaceUp.

Chapter 3743

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailBound}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailBound}^{\uparrow}$ packet records the finite tail estimate used when a regular rational Cauchy name is read as a completion-facing real datum. It consumes the observation windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic radius ledger of chapter 291, the bounded Cauchy sequence route of chapter 1522, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a finite tail-bound witness for completion and Real sealing; it is not a proof of ambient Real completeness, quotient equality, countable choice, or trichotomy.

Definition 3743.1 (Regular Cauchy tail-bound carrier)**.** *A* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, D, N, T_0, H, C, P, L)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, displaying the finite stream windows from which the tail request is read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking those windows as regular rational Cauchy data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording the dyadic tolerance and comparison rows used by the tail estimate.*
4. *N is the unary tail-index row, naming the finite stage after which the displayed radius is to be enforced.*
5. *T_0 is the later-index comparison row, recording the finite comparisons that put every displayed later observation beyond N and inside the dyadic radius from D.*
6. *H records componentwise* hsame *transport for the stream, regular readback, dyadic radius, tail index, later-index comparison, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through*

$$S, R, D, N, T_0 \longmapsto H, P, L.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, R, D, N, T_0, H, C, P, L$. It contains no coordinate for a quotient completion equality, selected limit, countable-choice sequence, trichotomy decision, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailBoundUp.RegularCauchyTailBoundTasteGate single carrier alignment`

Definition 3743.2 (Regular Cauchy tail-bound classifier)**.** *Two accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *carriers*

$$T = (S, R, D, N, T_0, H, C, P, L)$$

and

$$T' = (S', R', D', N', T_0', H', C', P', L')$$

are classified together exactly when the displayed hsame *rows transport S to S', R to R', D to D', N to N', and T_0 to T_0', and when the displayed* Cont *rows replay the same stream-to-readback-to-radius-to-tail-index route on both packets. The classifier has no coordinate for host real equality, completion quotient equality, selected representatives, or order trichotomy.*

Theorem 3743.3 (Regular Cauchy tail-bound NameCert obligations)**.** *For every accepted* RegularCauchyTailBound *carrier*

$$T = (S, R, D, N, T_0, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, StreamName observation exposure through S, RegSeqRat readback exposure through R, dyadic radius accounting through D, unary tail-index exposure through N, later-index comparison through T_0, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 3743.1. The accepted packet displays the stream observation row, regular rational readback row, dyadic radius ledger, unary tail index, later-index comparison row, transport row, replay row, provenance row, and local name.

The tail estimate is read in order. A consumer first opens S and R, then reads the dyadic tolerance in D, and only then consumes the finite index row N and the later-index comparisons in T_0.

The structural rows preserve that route. The hsame and Cont rows transport and replay the same finite observations, while P and L source and name the packet. Since no coordinate stores ambient Real completeness, quotient equality, countable choice, selected limits, or trichotomy, those principles are outside the local certificate surface. □

Theorem 3743.4 (Regular Cauchy tail-bound completion handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*, regular-completion, or bounded-Cauchy consumer of an accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CauchySequenceBounded}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff supplies a displayed tail-bound packet to completion-facing chapters, but it does not export ambient completeness, countable choice, quotient equality of Real names, selected representatives, or trichotomy.

Proof. Apply Definition 3743.3. It forces every tail-bound read through S, R, D, N, T_0 before the structural rows can be consumed.

The completion-facing route is finite. Stream windows are repacked as regular rational readback, the dyadic row supplies the radius, and the unary index plus later-index comparisons form the displayed tail witness.

Finally inspect the missing coordinates of the carrier and classifier. They preserve the finite route under transport and replay, but they contain no selected limit, quotient equality, countable-choice selector, trichotomy row, or ambient completeness principle. Thus downstream Real and completion packets receive only the finite tail-bound handoff. □

Theorem 3743.5 (Regular Cauchy tail-bound RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *reader of an accepted* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *packet must read the* $\mathsf{StreamName}^{\uparrow}$ *observation row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger D, the unary tail-index row N, and the later-index comparison row T_0 before any* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal is consumed. The handoff keeps the Real seal downstream and exports no quotient streams, selected limits, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Specialize Definition 3743.3 to a RegSeqRat-facing reader. The obligations expose S, R, D, N, T_0 before the transport, replay, provenance, and local naming rows can be used.

Use the completion handoff of Definition 3743.4 only at the boundary where a Real or completion-facing consumer appears. Before that boundary, the reader sees the stream observation, regular rational readback, radius ledger, tail index, and later-index comparisons as finite rows.

Thus the narrower handoff is the RegSeqRat portion of the same packet. The downstream Real seal cannot be used to supply quotient streams, selected limits, countable choice, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3743.1, Definition 3743.2, Definition 3743.3, Definition 3743.4, and Definition 3743.5 seed the finite regular-Cauchy tail-bound packet over stream observations, regular rational readback, dyadic radii, unary tail indices, later-index comparisons, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, countable choice, quotient equality, trichotomy, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, tail-index exposure, later-index comparison, dyadic-radius accounting, replay stability, and ledger nonescape for BEDC.Derived.RegularCauchyTailBoundUp.

Constructive story: A $\mathsf{RegularCauchyTailBound}^{\uparrow}$ packet is generated from one StreamName observation row, one RegSeqRat readback row, one DyadicRatCore radius ledger, one unary tail-index row, one later-index comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3744

A Concrete Naming Certificate for HochsterNerveBettiReduction$^{\uparrow}$

The HochsterNerveBettiReduction$^{\uparrow}$ packet realizes the Hochster–nerve reduction of subsection 9462.28.21 as a finite NameCert surface. It turns the vision theorem Definition 9462.1639 into displayed support rows, nerve-face rows, homology-range rows, Betti-strand rows, and ledger rows. The packet is a sibling of the homology carrier of chapter 98, the finite simplicial-chain carrier of chapter 3414, and the simplicial approximation surface of chapter 3402. It names a finite algebraic-topology reduction; it does not assert field faithfulness, nontriviality of coefficients, unrestricted minimal-resolution construction, or a Lean-checked homology computation.

Definition 3744.1 (Hochster nerve Betti reduction carrier)**.** *A* HochsterNerveBettiReduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (V, F, N, H_0, B_0, G, T, P, L)$$

with the following displayed rows:

1. *V is the finite support row, recording the displayed vertex subset and the local support window read by Hochster's formula.*
2. *F is the nerve-face row, recording the finite face and nonface tests that produce the nerve complex used by the reduction.*
3. *N is the homology-range row, recording the displayed dimensions in which reduced homology can be read.*
4. *H_0 is the finite homology row, read through the* Homology$^{\uparrow}$ *and* FiniteSimplicialChain$^{\uparrow}$ *carriers rather than through an ambient chain complex.*
5. *B_0 is the Betti-strand row, recording the diagonal and top-strand entries exposed by the support and homology rows.*
6. *G records componentwise* hsame *transport for the support, nerve-face, homology-range, finite homology, Betti-strand, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from finite support and nerve faces to homology ranges and then to Betti-strand ledgers.*
8. *P is the* Pkg *provenance over* HochsterNerveBettiReduction$^{\uparrow}$, Homology$^{\uparrow}$, FiniteSimplicialChain$^{\uparrow}$, SimplicialApproximation$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. L is the local $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $V, F, N, H_0, B_0, G, T, P, L$. It has no coordinate for a field-faithfulness theorem, coefficient nontriviality instance, unrestricted free resolution, or host-level equality of Betti tables.

Theorem 3744.2 (Hochster nerve Betti reduction obligations)**.** *For every accepted* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *carrier*

$$B = (V, F, N, H_0, B_0, G, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$ *surface consists exactly of finite support exposure through V, nerve-face exposure through F, homology-range exposure through N, finite homology readback through H_0, Betti-strand ledger exposure through B_0, componentwise transport through G, displayed replay through T, package provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.HochsterNerveBettiReductionUp.HochsterNerveBettiReductionNameCertObligations`

Proof. Project the carrier of Definition 3744.1. The finite support, nerve-face, homology-range, finite homology, Betti-strand, transport, replay, provenance, and local name rows exhaust the accepted packet.

The reduction is read along the finite route. The support row V fixes the displayed vertex window, the face row F builds the nerve complex, and the range row N restricts the homology degrees that can be queried.

The Betti ledger is downstream of finite homology. The row H_0 is read through homology and finite simplicial-chain carriers, and B_0 records only the strand entries obtained from that range. The structural rows G, T, P, L transport, replay, source, and name the same finite route, so no field-faithfulness theorem, nontrivial coefficient instance, unrestricted resolution, or host Betti-table equality is exported. □

Theorem 3744.3 (Hochster nerve Betti reduction TasteGate boundary)**.** *A downstream consumer may promote a* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *packet to a checked Betti computation only after future* Nontrivial *and* FieldFaithful *TasteGate instances are displayed for the coefficient surface. Before those instances are present, the packet exports only the finite support, nerve-face, homology-range, Betti-strand, transport, replay, provenance, and local naming rows of Definition 3744.2.*

Proof. Read the NameCert obligations first. They expose $V, F, N, H_0, B_0, G, T, P, L$ and no coefficient-faithfulness or nontriviality coordinate.

The TasteGate instances are therefore separate inputs. A consumer that wants a field-faithful Betti computation must supply those coefficient gates before interpreting the finite Betti ledger as a checked algebraic-topology calculation.

Without those gates, the accepted reading remains the finite reduction surface. It may cite the Hochster–nerve vision route and the sibling homology carriers, but it cannot claim unrestricted free resolutions, host-level Betti-table equality, or Lean-checked homology arithmetic. □

Falsifiable prediction: Every $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ read factors through one finite support row, one nerve-face row, one homology-range row, one finite homology row, one Betti-strand ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: homology, finite_simplicial_chain, simplicial_approximation, dna_window_six_bound alexander_dual_trigger_algebra: $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries support windows, nerve-face tests, homology-range restriction, Betti-strand ledgers, and the coefficient TasteGate boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3744.1, Definition 3744.2, and Definition 3744.3 seed the finite Hochster–nerve Betti-reduction packet over support rows, nerve-face rows, homology-range rows, finite homology rows, Betti-strand ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HochsterNerveBettiReduction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HochsterNerveBettiReductionUp.HochsterNerveBettiReductionNameCertObligations`

Bridge status: none

Not claimed: This chapter does not claim unrestricted free resolutions, host Betti-table equality, or bridge checking.

Upgrade path: Obligation closure requires checked finite support exposure, nerve-face exposure, homology-range exposure, finite homology readback, Betti-strand ledger stability, coefficient TasteGate instances, replay stability, and ledger nonescape for BEDC.Derived.HochsterNerveBettiReductionUp.

Constructive story: A $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ packet is generated from a finite support row, a nerve-face row, a homology-range row, a finite homology row, a Betti-strand row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3745

A Concrete Naming Certificate for $\mathsf{BrouwerDegree}^{\uparrow}$

Following Brouwer and standard constructive degree-theory presentations, the $\mathsf{BrouwerDegree}^{\uparrow}$ packet records the finite boundary ledger by which a displayed continuous map on a compact domain is read as an integer degree certificate. It sits between the circle boundary surface of chapter 58, the compact-domain ledger of chapter 60, the graph and modulus rows of chapter 130, the chain-boundary route of chapter 98, and the terminal real-name seal of chapter 38. The packet names degree data only as finite BHist rows; it does not claim the classical Brouwer fixed point theorem, host topological degree, selected preimages, quotient homology classes, or ambient compactness.

Definition 3745.1 (Brouwer degree carrier)**.** *A* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, B, F, O, I, H, T, C, P, N)$$

with the following displayed rows:

1. *K is the compact-domain row, read through the finite compact or compact-metric certificate surface.*
2. *B is the boundary row, with its $\mathsf{S}^{1\uparrow}$ orientation circle ledger and the displayed boundary inclusion into K.*
3. *F is the $\mathsf{ContinuousMap}^{\uparrow}$ row whose graph, modulus, and boundary restriction are visible to the packet.*
4. *O is the orientation ledger, recording the finite sign, face-order, or winding-window data used by the boundary read.*
5. *I is the integer-degree readback row, storing the displayed $\mathbb{Z}$-valued degree output and its finite comparison classifier.*
6. *H is the $\mathsf{Homology}^{\uparrow}$ boundary row, exposing the finite chain-boundary or cycle-class route consumed by the degree readback.*
7. *T is the componentwise hsame transport row for the compact-domain, boundary, continuous-map, orientation, integer-readback, and homology-boundary rows.*
8. *C is the displayed Cont replay from K, B, F, O, H to the integer readback row I and then to the local naming row.*

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, F, O, I, H, T, C, P, N$. It has no coordinate for a host degree function, selected zeros or preimages, quotient homology classes, open-cover compactness, or a Brouwer fixed-point conclusion.

Lean4 ✓ `BEDC.Derived.BrouwerDegreeUp.BrouwerDegreeTasteGate single carrier alignment`

Theorem 3745.2 (Brouwer degree NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier*

$$D = (K, B, F, O, I, H, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-domain row K, boundary and* $\mathsf{S}^{1\uparrow}$ *row B, continuous-map row F, orientation ledger O, integer-degree readback I, homology boundary row H, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and local naming row N. Every degree consumer must read the boundary, orientation, and homology rows before the integer row is exported.*

Proof. Project the accepted carrier. The compact-domain row K, boundary row B, and continuous-map row F are the only geometric inputs, and each is already a displayed finite BEDC row.

The degree-specific data are O, I, H. The orientation ledger O supplies the finite boundary orientation, the homology boundary row H supplies the finite chain-boundary route, and the integer row I is read only after those two rows and the continuous-map boundary restriction have been displayed.

The structural rows T, C, P, N transport, replay, source, and name that same route. Since no carrier coordinate stores host topological degree, selected preimages, quotient homology classes, open-cover compactness, or fixed-point existence, the listed rows exhaust the local NameCert obligations. □

Theorem 3745.3 (Brouwer degree boundary handoff)**.** *Every boundary-facing degree read from an accepted* $\mathsf{BrouwerDegree}^{\uparrow}$ *carrier factors through*

$$K, B, F, O, H \longmapsto I \longmapsto T, C, P, N.$$

Thus the compact domain, boundary $\mathsf{S}^{1\uparrow}$ *row, continuous-map restriction, orientation ledger, and homology boundary row are displayed before the integer-degree readback can be transported or named.*

The handoff exports only this finite boundary route. It does not assert the Brouwer fixed point theorem, a host topological-degree functor, selected zeros, selected preimages, quotient homology classes, or ambient compactness.

Proof. Start with the NameCert obligations of Definition 3745.2. They place the compact-domain, boundary, continuous-map, orientation, and homology rows before the integer readback row.

The boundary handoff is therefore forced through K, B, F, O, H. The row B supplies the $\mathsf{S}^{1\uparrow}$ boundary ledger, F supplies the displayed map restriction, O records the orientation data, and H records the homology boundary route consumed by the integer row I.

Finally replay the structural rows T, C, P, N. These rows preserve and name the same finite route, so the exported degree surface has no additional coordinate for host degree theory, fixed-point existence, selected preimages, quotient homology, or ambient compactness. □

Falsifiable prediction: A $\mathsf{BrouwerDegree}^{\uparrow}$ consumer that exports an integer degree without reading the boundary row B, orientation ledger O, and homology boundary row H refutes the carrier surface.

Independence witness: s1, compact, continuousmap, homology, real: the sibling chapters supply boundary circles, compact rows, graph and modulus rows, chain-boundary routes, and real seals separately, while this chapter binds those rows into the finite degree readback packet $D = (K, B, F, O, I, H, T, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerDegree}^{\uparrow}$ packet is generated from a compact-domain row, boundary and SOne row, ContinuousMap row, orientation ledger, integer-degree readback, Homology boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$, seed)

Scope closed: Definition 3745.1, Definition 3745.2, and Definition 3745.3 seed the finite compact-domain, boundary, continuous-map, orientation, homology-boundary, integer-readback, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BrouwerDegree}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the Brouwer fixed point theorem, host topological degree, selected zeros, selected preimages, quotient homology classes, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary and orientation exactness, continuous-map restriction stability, homology boundary handoff, integer readback stability, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerDegreeUp.

Chapter 3746

A Concrete Naming Certificate for $\mathsf{Metacompact}^{\uparrow}$

Following Morita and Kelley general topology, the $\mathsf{Metacompact}^{\uparrow}$ packet records the point-finite refinement surface between paracompact locally finite refinement, Moore-space developments, and star-refinement cover ledgers. It sits after the paracompact cover surface of chapter 1963, beside the developable-neighbourhood rows of chapter 3592, and before the uniform star-refinement handoff of chapter 2167. The packet names only a displayed open-cover request, a point-finite open refinement, and the ledgers that certify coverage and point incidence; it does not choose arbitrary subcovers, infer local finiteness, or import compactness.

Definition 3746.1 (Metacompact open-cover carrier)**.** *A* $\mathsf{Metacompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, O, R, Q, I, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row containing the open predicates, point observations, and neighbourhood classifier used by the packet.*
2. *O is the open-cover request row, listing the displayed cover cells and the pointwise coverage ledger for T.*
3. *R is the open-refinement row, listing the displayed refinement cells and their containment witnesses back into cells of O.*
4. *Q is the point-finite incidence row, recording for each displayed point observation the finite list of refinement cells that contain it.*
5. *I is the coverage-and-incidence exactness row, requiring every point covered by O to be covered by a refinement cell and requiring every incidence read to be one of the finite entries in Q.*
6. *S is the sibling handoff row to the* $\mathsf{Paracompact}^{\uparrow}$*,* $\mathsf{MooreSpace}^{\uparrow}$*, and* $\mathsf{StarRefinementUniformCover}^{\uparrow}$ *cover ledgers.*
7. *H records componentwise* hsame *transport for $T, O, R, Q, I, S, C, P, N$.*
8. *C records displayed* Cont *replay from an open-cover request through refinement, point-finite incidence, sibling handoff, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{Metacompact}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Paracompact}^{\uparrow}$*,* $\mathsf{MooreSpace}^{\uparrow}$*,* $\mathsf{StarRefinementUniformCov}$ BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}$ *row.*
The carrier contains no coordinate for selected subcovers, locally finite partitions, metric development extraction, uniform-star generation, or ambient compactness.

Lean4 ✓ `BEDC.Derived.MetacompactUp.MetacompactTasteGate single carrier alignment`

Definition 3746.2 (Metacompact classifier)**.** *The* $\mathsf{Metacompact}^{\uparrow}$ *classifier accepts exactly those displayed packets $M = (T, O, R, Q, I, S, H, C, P, N)$ whose refinement row R is open in T, whose containment ledger maps each refinement cell into a displayed cell of O, whose coverage ledger in I covers the same point observations as O, and whose incidence row Q displays a finite list of refinement cells at each point observation. A classifier read is rejected if it asks for a subcover selection, a locally finite refinement, a Moore development, a uniform star refinement, or a compactness principle not present among $T, O, R, Q, I, S, H, C, P, N$.*

Theorem 3746.3 (Metacompact carrier admission)**.** *For every accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet $M = (T, O, R, Q, I, S, H, C, P, N)$, carrier admission exposes exactly the topology source, open-cover request, open-refinement row, point-finite incidence row, coverage-and-incidence exactness row, sibling handoff row, transport, replay, provenance, and local naming rows of Definition 3746.1.*

Proof. Project the carrier definition. The tuple lists $T, O, R, Q, I, S, H, C, P, N$ and no further coordinate.

The classifier of Definition 3746.2 accepts only packets that read refinement, containment, coverage, point-finite incidence, sibling handoff, transport, replay, provenance, and local naming through those coordinates. Therefore carrier admission is the displayed finite packet itself. □

Theorem 3746.4 (Metacompact point-finite refinement exactness)**.** *For every accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet $M = (T, O, R, Q, I, S, H, C, P, N)$, the row R is an open refinement of the request row O, and the row Q is a point-finite incidence ledger for R. Every displayed point covered by O is covered by at least one row of R, and every refinement cell containing that point appears in the finite list recorded by Q.*

Proof. Use the refinement part of the classifier. It requires the cells of R to be open in T and to carry containment witnesses into the displayed cover cells of O.

Use the exactness row I next. It links coverage by O to coverage by R and restricts incidence reads to the finite lists displayed in Q.

Thus the point-finite refinement is not inferred from a choice principle. It is stored as the finite incidence ledger attached to the accepted packet. □

Theorem 3746.5 (Metacompact sibling cover handoff)**.** *Every cover-refinement consumer of an accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet factors through the sibling handoff row S before citing* $\mathsf{Paracompact}^{\uparrow}$, $\mathsf{MooreSpace}^{\uparrow}$, *or* $\mathsf{StarRefinementUniformCover}^{\uparrow}$. *The handoff may compare locally finite, development, and star-refinement ledgers with the point-finite row Q, but it cannot promote the metacompact packet to any of those stronger cover principles.*

Proof. Read the sibling coordinate S in the carrier. It is the only row that mentions the paracompact, Moore-space, and star-refinement neighbours.

The refinement and incidence rows remain R, Q, I. Comparing them with sibling cover ledgers does not add locally finite partitions, countable developments, or uniform-star generation to the packet.

Therefore any downstream citation of the neighbours passes through S as a handoff and returns to the same point-finite refinement surface. □

Theorem 3746.6 (Metacompact transport stability)**.** *Transporting an accepted* $\mathsf{Metacompact}^{\uparrow}$ *read along H preserves the topology source T, open-cover request O, refinement row R, point-finite incidence row Q, exactness row I, sibling handoff S, replay row C, provenance P, and local naming row N. The transported classifier still reads only the displayed point-finite refinement packet.*

Proof. The row H is componentwise by definition of the carrier. It transports the same open-cover request, refinement cells, point-finite incidence lists, exactness row, and sibling handoff.

The classifier has no alternate coordinate for cover selection or compactness. After transport, the accepted read is still checked by the same rows $T, O, R, Q, I, S, C, P, N$ and by no hidden cover operation. □

Theorem 3746.7 (Metacompact nonescape)**.** *No consumer of an accepted* $\mathsf{Metacompact}^{\uparrow}$ *packet can read selected subcovers, locally finite partitions of unity, Moore developments, uniform star-refinement generation, ambient compactness, or standard bridge data from the packet. All consumer reads are confined to $T, O, R, Q, I, S, H, C, P, N$.*

Proof. Inspect the carrier coordinates. The only cover-theoretic data are the open-cover request O, open refinement R, point-finite incidence row Q, exactness row I, and sibling handoff S.

Transport, replay, provenance, and naming do not add cover choices. They preserve and expose the same finite rows already present in the carrier.

The excluded stronger objects have no projection from $T, O, R, Q, I, S, H, C, P, N$, so a consumer cannot extract them from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3746.1, Definition 3746.2, and Definition 3746.4 seed the finite point-finite open-refinement packet over topology source, open-cover request, refinement, incidence, exactness, sibling handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Metacompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, local finiteness, partitions of unity, Moore-space development extraction, uniform star-refinement generation, Lean verification, or standard bridge closure.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, point-finite refinement exactness, sibling cover handoff, transport stability, and nonescape for BEDC.Derived.MetacompactUp.

Constructive story: A $\mathsf{Metacompact}^{\uparrow}$ packet is generated from a topology source row, open-cover request row, open-refinement row, point-finite incidence ledger, coverage-and-incidence exactness row, sibling cover handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3747

A Concrete Naming Certificate for FiniteRecognizerCarrierObstruction$^{\uparrow}$

The FiniteRecognizerCarrierObstruction$^{\uparrow}$ packet realizes the recognizer-to-carrier boundary in chapter 9379 as a finite seven-row NameCert obstruction. It sits beside the recognizer-flow packet of chapter 1048, the finite reflection tuple of chapter 993, the normalization certificate router of chapter 1108, and the inductive-acceptance packet of chapter 915. The packet names the finite landing surface that must be present before a recognizer-derived carrier can even be compared with a standard-bridge question.

Definition 3747.1 (FiniteRecognizerCarrierObstruction carrier)**.** *A* FiniteRecognizerCarrierObstruction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, G, Q, F, R, T, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the recognizer-acceptance row consumed from* GroundCompilerRecognizerFlow$^{\uparrow}$*.*
2. *G is the generated-recognizer mediation row, recording that the accepted recognizer is the generated object read by the compiler-facing flow.*
3. *Q is the certificate-gate row, recording that the recognizer output has passed the displayed gate before any carrier comparison is attempted.*
4. *F is the finite tuple extraction row consumed through* FiniteReflectionTuple$^{\uparrow}$*.*
5. *R is the component readback row, exposing the* BHist *components recovered from the finite tuple image.*
6. *T is the componentwise* hsame *and* NameCert *transport row for the recovered components.*
7. *K is the* ChapterTasteGate *row for the displayed carrier-facing surface.*
8. *H records componentwise* hsame *transport for $A, G, Q, F, R, T, K, C, P, N$.*
9. *C records displayed* Cont *replay from recognizer acceptance through generated-recognizer mediation, certificate gate, tuple extraction, readback, componentwise transport,* ChapterTasteGate*, provenance, and local naming.*
10. *P is the* Pkg *provenance over* FiniteRecognizerCarrierObstruction$^{\uparrow}$*,* GroundCompilerRecognizerFlow$^{\uparrow}$*,* FiniteReflectionTuple$^{\uparrow}$*,* MetaCICNormalizationCertificateRouter$^{\uparrow}$*,* KernelInductiveAcceptance$^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*, and* ChapterTasteGate *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}}$ *row.*
The carrier does not construct a recognizer-to-carrier map, does not assert unrestricted beta-conversion, and does not supply a StandardBridge *theorem. It only records the finite landing rows that a later construction must expose before those questions are reached.*

Lean4 (definition) `BEDC.Derived.FiniteRecognizerCarrierObstructionUp.FiniteRecognizerCarrierObstructionUp`

Theorem 3747.2 (FiniteRecognizerCarrierObstruction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *carrier* $O = (A, G, Q, F, R, T, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}}$ *surface consists exactly of recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise* hsame *and* NameCert *transport,* ChapterTasteGate *exposure, componentwise transport, replay, provenance, and local naming.*

Proof. Project the carrier. The first seven rows A, G, Q, F, R, T, K are the recognizer-to-carrier landing surface, while H, C, P, N are the structural rows that preserve and expose that surface.

Each obligation is read from its displayed row. Acceptance comes from A, generated-recognizer mediation from G, gate admission from Q, tuple extraction from F, readback from R, componentwise transport from T, and chapter-facing gate exposure from K.

No obligation row supplies a recognizer-to-carrier map or a standard bridge. The packet therefore isolates the obstruction boundary rather than resolving it. □

Theorem 3747.3 (FiniteRecognizerCarrierObstruction missing-row refusal)**.** *An attempted recognizer-to-carrier read that omits any one of A, G, Q, F, R, T, K fails inside the* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *packet before the* StandardBridge *question is reached. The failure is a missing landing row, not a failed bridge theorem.*

Proof. Use the obligation surface of Definition 3747.2. It lists the seven landing rows as jointly required before the carrier-facing surface is displayed.

If one row is absent, the replay route C cannot traverse the full path from recognizer acceptance to ChapterTasteGate. The packet then records the absence as an obstruction at the landing surface.

Since the packet has no StandardBridge theorem coordinate, such a failure occurs before bridge comparison and cannot be read as a bridge counterexample. □

Theorem 3747.4 (FiniteRecognizerCarrierObstruction seven-row landing)**.** *If a packet displays recognizer acceptance A, generated-recognizer mediation G, certificate-gate admission Q, finite tuple extraction F, component readback R, componentwise* hsame *and* NameCert *transport T, and* ChapterTasteGate *exposure K, together with the structural transport, replay, provenance, and local naming rows of Definition 3747.1, then the packet lands on the* $\mathsf{FiniteRecognizerCarrierObstruction}^{\uparrow}$ *surface before any* StandardBridge *question is evaluated.*

Proof. Project the carrier of Definition 3747.1. Its first seven displayed coordinates are exactly A, G, Q, F, R, T, K, and the remaining rows H, C, P, N preserve, replay, source, and name that same landing surface.

Apply Definition 3747.2. The obligation inventory recognizes the seven rows as the complete recognizer-to-carrier landing face, so a packet displaying all of them has reached the obstruction surface rather than a later bridge surface.

This is the positive side of Definition 3747.3. Omission of one landing row refuses the read, while display of all seven rows admits the finite obstruction surface. Neither direction constructs a recognizer-to-carrier map or closes a StandardBridge theorem. □

Theorem 3747.5 (FiniteRecognizerCarrierObstruction sibling nonbijection)**.** *The carrier of Definition 3747.1 is not bijective, row for row, with* GroundCompilerRecognizerFlow$^{\uparrow}$, FiniteReflectionTuple$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, *or* KernelInductiveAcceptance$^{\uparrow}$. *Any row-preserving bijection would have to preserve recognizer acceptance, generated-recognizer mediation, certificate-gate passage, tuple extraction, component readback, componentwise* hsame *and* NameCert *transport, and* ChapterTasteGate *exposure at once, and no listed sibling carries that whole surface.*

Proof. Compare first with GroundCompilerRecognizerFlow$^{\uparrow}$. It supplies recognizer acceptance, generated-recognizer mediation, and certificate-gate rows, but it does not carry finite tuple extraction, tuple readback, componentwise NameCert transport, and ChapterTasteGate exposure as one carrier.

Compare next with FiniteReflectionTuple$^{\uparrow}$. It supplies tuple image and readback rows, but it does not carry recognizer acceptance, generated-recognizer mediation, or certificate-gate passage.

The normalization router and inductive-acceptance packet have different endpoints. They record normalization evidence routing or inductive declaration acceptance, not the simultaneous seven-row recognizer-to-carrier landing surface. Hence no row-preserving bijection can identify them with this obstruction packet. □

Theorem 3747.6 (FiniteRecognizerCarrierObstruction TasteGate follow-up surface)**.** *Any Lean-side* Nontrivial *and* FieldFaithful *TasteGate instance for* FiniteRecognizerCarrierObstruction$^{\uparrow}$ *must preserve the displayed carrier rows* $A, G, Q, F, R, T, K, H, C, P, N$. *Nontriviality separates a complete seven-row landing from a missing-row obstruction, and field faithfulness requires every accepted read to retain each of the seven landing rows componentwise.*

Proof. The nontrivial boundary is visible in the carrier. A complete landing displays A, G, Q, F, R, T, K, while a failed landing omits at least one of those rows and is refused by Definition 3747.3.

Field faithfulness is componentwise. Preserving the recognizer-to-carrier obstruction requires preserving acceptance, generated recognizer, gate, tuple extraction, readback, componentwise transport, and ChapterTasteGate rows together.

Thus the later Lean instances must consume the same rows as the paper carrier and may not replace them by a vacuous or report-only encoding. □

Falsifiable prediction: Any attempted recognizer-to-carrier map missing recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, or ChapterTasteGate exposure fails before the StandardBridge question is reached.

Independence witness: groundcompilerrecognizerflow, finitereflectiontuple, metacicnormalizationcertificaterouter, kernelinductiveacceptance: GroundCompilerRecognizerFlow$^{\uparrow}$ records recognizer acceptance, generated-recognizer mediation, and certificate-gate passage without tuple extraction and readback; FiniteReflectionTuple$^{\uparrow}$ records tuple image and readback without recognizer-flow acceptance; MetaCICNormalizationCertificateRouter$^{\uparrow}$ routes normalization evidence without the seven-row recognizer landing surface; KernelInductiveAcceptance$^{\uparrow}$ records inductive declaration acceptance without recognizer-to-carrier landing. None is bijective to FiniteRecognizerCarrierObstruction$^{\uparrow}$.

Closure status for NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$, seed)

Scope closed: Definition 3747.1, Definition 3747.2, and Definition 3747.3 seed the seven-row recognizer-to-carrier obstruction packet over recognizer acceptance, generated recognizer, certificate gate, finite tuple extraction, component readback, componentwise transport, ChapterTasteGate, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteRecognizerCarrierObstruction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a recognizer-to-carrier construction, unrestricted beta-conversion, standard bridge closure, Lean verification, or global MetaCIC normalization.

Upgrade path: Obligation closure requires checked carrier admission, seven-row landing exactness, missing-row refusal, sibling nonbijection, TasteGate nontriviality, and field faithfulness for BEDC.Derived.FiniteRecognizerCarrierObstructionUp.

Constructive story: A FiniteRecognizerCarrierObstruction$^{\uparrow}$ packet is generated from recognizer acceptance, generated-recognizer mediation, certificate-gate admission, finite tuple extraction, component readback, componentwise hsame and NameCert transport, ChapterTasteGate exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 3748

A Concrete Naming Certificate for $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy reals, the $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ packet records the finite naming surface for the canonical modulus read from a regular rational Cauchy presentation. It sits after the dyadic tolerance rows of chapter 291, the finite window rows of chapter 126, the regular rational sequence carrier of chapter 301, and the real seal of chapter 38. The chapter names the displayed modulus object needed by the L10 RegSeqRat-to-Real route without importing an ambient Cauchy-search principle.

Definition 3748.1 (Regular Cauchy canonical modulus carrier)**.** *A* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic radii, denominator windows, and scale comparison data.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the observed rational stream windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row recording the Cauchy bound read from the same windows.*
4. *Q is the canonical modulus row, assigning to each displayed tolerance scale the finite window at which the regularity row certifies stability.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached by the regular rational presentation.*
6. *H records componentwise* hsame *transport for the dyadic, stream, regularity, modulus, endpoint, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a modulus request through D, S, R, Q, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for unbounded search over Cauchy witnesses, quotient equality of real presentations, countable choice, a selected limit outside E, or a host completeness theorem.

Lean4 (definition) `BEDC.Derived.RegularCauchyCanonicalModulusUp.RegularCauchyCanonicalModulusUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyCanonicalModulusUp.RegularCauchyCanonicalModulusTasteGate`
`single carrier alignment`

Theorem 3748.2 (Regular Cauchy canonical modulus NameCert obligations)**.** *For the carrier of Definition 3748.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance row D, stream-window row S, regular rational row R, canonical modulus row Q, real endpoint row E, transport row H, replay row C, provenance row P, and local naming row N;*
2. *canonical-modulus exactness, requiring each tolerance scale read from D to be assigned by Q to a finite stream window already present in S and certified by the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R;*
3. *classifier stability, requiring two accepted packets to compare only through componentwise* hsame *transport of D, S, R, Q, E, P, N;*
4. *endpoint nonescape, requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to reach E only after the displayed dyadic, stream, regularity, and modulus rows have been read;*
5. *provenance and naming exactness, requiring P and N to record the same finite RegSeqRat-to-Real route.*

Proof. Carrier admission is projection from the displayed packet $K = (D, S, R, Q, E, H, C, P, N)$. Each row is finite BHist data, and no row names an ambient search operator.

Canonical-modulus exactness reads Q against the tolerance row D and stream row S. The regularity row R supplies the bounded Cauchy witness for the displayed finite window, so the modulus is canonical only relative to those already named rows.

For classifier stability, compare accepted packets componentwise through H. The dyadic, stream, regularity, modulus, endpoint, provenance, and local naming rows are transported together, and replay through C preserves the same route.

Finally inspect a $\mathsf{Real}^{\uparrow}$ consumer. It can reach the endpoint row E only through D, S, R, Q and the structural rows H, C, P, N, so quotient equality, countable choice, unbounded search, and host completeness are outside the NameCert surface. □

Theorem 3748.3 (Regular Cauchy canonical modulus RegSeqRat handoff)**.** *Every accepted* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *read that cites a* $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyCanonicalModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff first reads dyadic tolerance scales, then finite stream windows, then the regular rational Cauchy row, then the canonical modulus row, and only then the real endpoint seal. It supplies no independent limit selector, no quotient equality of presentations, no unbounded Cauchy-search oracle, and no ambient real-completeness argument.

Proof. Start with a RegSeqRat-to-Real consumer. The carrier forces the request to read D and S before the regularity row R can be interpreted, because the Cauchy bound is attached to dyadic scales and finite stream windows.

Next read the canonical modulus row Q. Its entries are finite assignments from displayed tolerance scales to displayed windows, with R certifying regularity on those windows.

The endpoint row E is then reached only through the same finite route. The structural rows H, C, P, N transport, replay, source, and name that route, so no absent search, quotient, selector, or completion principle can be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ read factors through displayed DyadicRatCoreUp tolerance rows, StreamName finite windows, RegSeqRat regularity rows, a canonical modulus row, a RealUp endpoint seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance scales, finite stream windows, regular rational Cauchy bounds, a canonical modulus assignment, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}$ packet is generated from dyadic tolerance scales, finite StreamName windows, RegSeqRat regularity rows, a canonical modulus assignment, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3748.1, Definition 3748.2, and Definition 3748.3 seed the finite canonical-modulus route from RegSeqRat data to the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCanonicalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of real presentations, countable choice, unbounded Cauchy search, selected limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, canonical-modulus exactness, classifier stability, endpoint nonescape, and RegSeqRat handoff for the displayed packet.

Chapter 3749

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$

Following the Bishop–Bridges constructive-completion treatment, the $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ packet records the finite stability surface by which a regular rational Cauchy name, its finite stream windows, dyadic tolerance ledger, terminal real seal, and completion-witness row remain attached to one completion route. It sits after the regular rational sequence interface of chapter 301, the stream-window naming surface of chapter 126, the dyadic rational core of chapter 291, the real-name seal of chapter 38, and the completion witness boundary of chapter 3579. The packet names finite stability and idempotence-boundary data for the displayed route; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of streams, selected limits, countable choice, or an external completion theorem.

Definition 3749.1 (Regular Cauchy completion stability carrier)**.** *A* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational Cauchy name and its finite tolerance schedule.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that selects the finite source observations inspected by the completion route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison rows used to certify regularity on the selected windows.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular sequence, stream windows, and dyadic tolerance ledger.*
5. *E is the* $\mathsf{CauchyCompletenessWitness}^{\uparrow}$ *row, exposing only the displayed completion-witness surface attached to Q, W, D, R.*
6. *H records componentwise* hsame *transport for the regular sequence, stream windows, dyadic ledger, real seal, completion witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from Q through W, D, R, E and back to the local naming row.*

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, W, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a chosen limit representative, ambient completeness of $\mathsf{Real}^{\uparrow}$*, countable choice, trichotomy, or a second completion route.*

Theorem 3749.2 (Regular Cauchy completion stability NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier $S = (Q, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCom}}$ *surface consists exactly of carrier admission for the displayed rows; regular-source exposure through Q; finite window selection through W; dyadic tolerance accounting through D; real-seal confinement through R; completion-witness attachment through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionStabilityUp.RegularCauchyCompletionStabilityNameCertObligat`

Proof. Carrier admission is projection from Definition 3749.1. The accepted packet displays exactly the regular rational source, the finite window schedule, the dyadic tolerance ledger, the terminal real seal, and the completion-witness row.

The stability route is read in source order. The row Q supplies the regular Cauchy name, W selects the finite observations that a consumer may inspect, and D records the dyadic tolerance comparisons visible on those observations.

The rows R and E then bind the completion-facing read. The real seal R is reached only after Q, W, D have been displayed, and the completion witness E attaches to that same route rather than to an external completeness principle.

Finally H, C, P, N transport, replay, source, and name the route. Since no coordinate stores quotient stream equality, a chosen limit representative, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, trichotomy, or a second completion route, the listed rows exhaust the local NameCert obligations. □

Theorem 3749.3 (Regular Cauchy completion idempotence boundary)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletenessWitness}^{\uparrow}.$$

If the consumer asks to complete a route already sealed by R and witnessed by E, the packet can only replay the same displayed rows $Q, W, D, R, E, H, C, P, N$. It cannot introduce a quotient of streams, a selected representative of the limit, a second real seal, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionStabilityUp.RegularCauchyCompletionStabilityCarrier idempotence boundary`

Proof. Start with a completion-facing read of the carrier. By Definition 3749.1, the only source of regular Cauchy data is Q, and every finite observation passes through the stream window row W and dyadic tolerance ledger D.

The completion read reaches R and E only along this route. Thus the real seal and the completion witness are already attached to the finite regular-source path before any consumer receives the packet.

For a second completion request, replay C and transport along H. The replay has no coordinate that could manufacture a quotient stream, chosen limit, second real seal, or ambient completeness proof, so the request remains on $Q, W, D, R, E, H, C, P, N$. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ consumer that exports a completion witness without first displaying the RegSeqRat source, StreamName windows, DyadicRatCore tolerance ledger, and RealUp seal refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchy-completeness-witness: these sibling chapters supply regular schedules, finite windows, dyadic tolerance rows, real seals, and completion witnesses separately, while this packet binds them into the finite stability carrier $S = (Q, W, D, R, E, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionStability}^{\uparrow}$ packet is generated from a RegSeqRat source row, StreamName finite windows, a DyadicRatCore tolerance ledger, a RealUp seal, a CauchyCompletenessWitness row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3749.1, Definition 3749.2, and Definition 3749.3 seed the finite regular-Cauchy completion stability surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-source exposure, finite-window selection, dyadic tolerance accounting, real-seal confinement, completion-witness attachment, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionStabilityUp.

Chapter 3750

A Concrete Naming Certificate for $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ packet records the finite subsequence bridge from a bounded real sequence display toward regular-Cauchy extraction. It sits between the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational handoff of chapter 301, the finite stream schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the bounded regular Cauchy route of chapter 1698, the bounded interval route of chapter 3716, and the diagonal extraction route of chapter 3469. The packet names the monotone-window surface used by Bolzano-Weierstrass-style arguments; it does not select an infinite subsequence or assert classical compactness.

Definition 3750.1 (Bounded monotone subsequence carrier)**.** *A* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, D, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule exposing the finite sequence windows available to the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording the dyadic tolerances and rational comparisons used inside the windows.*
3. *I is the bounded interval row naming the finite lower and upper barriers consumed by the monotone-window read.*
4. *W is the monotone-window ledger, consisting of a finite subsequence index list, a direction tag, and the displayed comparisons proving monotonicity inside the named window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached from S, D, I, W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational handoff has been displayed.*
7. *H records componentwise* hsame *transport for the schedule, dyadic ledger, interval bounds, monotone window, regular handoff, Real seal, continuation, provenance, and local name rows.*
8. *C records displayed* Cont *replay from finite window requests through the dyadic ledger, interval bounds, monotone window, regular-rational handoff, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, I, W, R, E, H, C, P, N$. It has no coordinate for a chosen infinite subsequence, quotient real equality, trichotomy, countable choice, full Bolzano-Weierstrass compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMonotoneSubsequenceUp.BoundedMonotoneSubsequenceTasteGate single carrier alignment`

Theorem 3750.2 (Bounded monotone subsequence monotone-window preservation)**.** *For an accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier $X = (S, D, I, W, R, E, H, C, P, N)$, every subsequence read exposed by the packet preserves the displayed monotone-window ledger W under the componentwise transport H and replay row C. The read first consumes the finite stream schedule S, the dyadic comparison ledger D, and the interval bound row I; only then may it read the monotone direction and finite subsequence list stored in W.*

Proof. Project the carrier rows. The stream schedule S is the only source of sequence observations, while D and I are the only rows that record the comparison scale and bounded interval data used by the window.

The monotone-window row W is therefore downstream of S, D, I. A consumer that asks for an increasing or decreasing window must read the finite index list and direction tag from W, and the displayed comparisons in D certify the order relation inside the named interval row.

Transport and replay do not add another source. The row H preserves the same displayed coordinates componentwise, and C replays the request path through the same windows and comparisons. Since the carrier has no coordinate for a chosen infinite subsequence, trichotomy, countable choice, or ambient completeness, the monotone-window surface is exactly the finite one stored in W. □

Theorem 3750.3 (Bounded monotone subsequence regular-Cauchy handoff)**.** *Every regular-Cauchy consumer of an accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier factors through the displayed route*

$$S \longmapsto D \longmapsto I \longmapsto W \longmapsto R \longmapsto E$$

together with H, C, P, N. The handoff can be consumed by the bounded regular Cauchy, bounded interval, and diagonal Cauchy subsequence packets, but it does not identify quotient real representatives, choose an infinite subsequence, or assert $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a regular-Cauchy read. Carrier admission gives the finite schedule S, the dyadic ledger D, the bounded interval row I, and the monotone-window row W as the only pre-handoff data.

Apply the monotone-window preservation theorem. The consumer cannot read R until the finite monotone comparisons in W have been displayed over the same S, D, I rows.

The regular-rational row R is then the unique handoff available to the downstream Cauchy packets, and the Real seal row E is downstream of that handoff. The structural rows H, C, P, N preserve, replay, source, and name the same route. Hence bounded regular Cauchy, bounded interval, and diagonal Cauchy consumers may reuse the finite handoff, but no quotient equality, choice-selected infinite subsequence, or ambient completeness principle is exported. □

Theorem 3750.4 (Bounded monotone subsequence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ *carrier $X = (S, D, I, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubseque}}$

surface consists exactly of carrier admission, finite-window classification, dyadic comparison coverage, bounded-interval nonescape, monotone-window preservation, regular-Cauchy handoff, Real-seal nonescape, and provenance replay through H, C, P, N.

No obligation row is stored outside the displayed carrier, and no exported read depends on full Bolzano-Weierstrass, chosen infinite subsequences, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission projects the tuple $S, D, I, W, R, E, H, C, P, N$. The finite-window classifier reads S, the dyadic comparison coverage reads D, and bounded-interval nonescape reads I before any subsequence-facing row is available.

Monotone-window preservation is supplied by Definition 3750.2. It makes W the finite source of the direction tag, subsequence index list, and displayed monotone comparisons.

Regular-Cauchy and Real-facing obligations are supplied by Definition 3750.3. The regular-rational row R and Real seal row E remain downstream of the same finite window data.

The structural rows close the surface. The transport H, replay C, provenance P, and local name N preserve and source the displayed route, and the carrier contains no coordinate for the excluded classical or ambient-completeness principles. Therefore the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneSubsequence}^{\uparrow}$ packet is generated from a StreamName schedule, a DyadicRatCore comparison ledger, bounded interval rows, a finite monotone-window ledger, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3750.1, Definition 3750.2, Definition 3750.3, and Definition 3750.4 name the finite monotone-window bridge from bounded real sequence data to regular-Cauchy extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneSubsequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BoundedMonotoneSubsequenceUp.BoundedMonotoneSubsequenceTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close full Bolzano-Weierstrass, chosen infinite subsequences, quotient real equality, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier, monotone-window, regular-Cauchy handoff, Real-seal, transport, replay, provenance, and local NameCert rows for the same finite packet.

Chapter 3751

A Concrete Naming Certificate for $\mathsf{DistributionDerivative}^{\uparrow}$

Following Schwartz distribution theory and Evans' weak-derivative presentation for PDE, the $\mathsf{DistributionDerivative}^{\uparrow}$ packet names the finite action route by which a distribution is differentiated through test probes. It sits after the distribution endpoint interface of chapter 186, the weak-derivative handoff of chapter 2210, the Sobolev ledger of chapter 440, the real seal of chapter 38, the regular rational readback surface of chapter 301, and the finite stream windows of chapter 126. The chapter records only the finite carrier, its naming obligations, and the weak-action route needed by downstream PDE and variational consumers.

Definition 3751.1 (Distribution derivative carrier)**.** *A* $\mathsf{DistributionDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, D, I, E, Q, W, H, C, P, N)$$

with the following displayed rows:

1. *A is the test-action row, listing the finite test probes, support windows, sign convention, and coordinate direction.*
2. *D is the input* $\mathsf{Distribution}^{\uparrow}$ *row, read only by evaluating the tests named in A.*
3. *I is the integration-by-parts sign row, recording the displayed minus sign and coordinate handoff for the derivative action.*
4. *E is the candidate derivative action row on the same finite tests.*
5. *Q is the shared* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for input and candidate action values.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite tests, tolerances, and readback prefixes.*
7. *H records componentwise* hsame *transport for $A, D, I, E, Q, W, C, P, N$.*
8. *C records displayed* Cont *replay from test selection to input action, candidate action, readback, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{DistributionDerivative}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for a host distribution quotient, selected representative, completed test-function space, full PDE regularity theorem, or ambient Real completeness.

Theorem 3751.2 (Distribution derivative NameCert obligations)**.** *For every accepted* $\mathsf{DistributionDerivative}^{\uparrow}$ *carrier* $K = (A, D, I, E, Q, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, test-action coverage, input distribution access, integration-by-parts sign exposure, candidate derivative action exposure, shared* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite* $\mathsf{StreamName}^{\uparrow}$ *window selection, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DistributionDerivativeUp.DistributionDerivativeTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3751.1. The analytic rows visible to a consumer are A, D, I, E, Q, W: tests, input distribution action, sign handoff, candidate derivative action, readback, and finite windows.

The structural rows H, C, P, N preserve and name only that route. Componentwise transport may compare two accepted packets, and replay may re-read the same displayed action chain, but neither operation creates a host derivative object.

Since the carrier has no coordinate for distribution quotients, selected representatives, completed test-function spaces, full PDE regularity, or ambient Real completeness, the NameCert obligations are exactly the displayed finite rows. □

Theorem 3751.3 (Distribution derivative weak-action route)**.** *Every* $\mathsf{WeakDerivative}^{\uparrow}$*,* $\mathsf{Sobolev}^{\uparrow}$*, PDE, trace, Rellich, or variational consumer of an accepted* $\mathsf{DistributionDerivative}^{\uparrow}$ *packet must read the derivative through the finite route*

$$A \longmapsto D, I, E \longmapsto Q \longmapsto W \longmapsto H, C, P, N.$$

The route supplies a distribution action and a candidate derivative action on the same displayed tests, with shared readback and stream windows. It does not supply quotient equality of distributions, a chosen representative, Sobolev completion, density of smooth tests, elliptic regularity, trace compactness, or a completed Real carrier.

Proof. Begin with a derivative-facing consumer. The test row A fixes the probes and support windows before any action value can be inspected.

Read D, I, E next. The input distribution action and candidate derivative action are tied by the displayed sign handoff, and both actions are evaluated only on the tests already selected by A.

Finally read Q, W, H, C, P, N. These rows provide real and regular-rational readback, finite windows, transport, replay, provenance, and local naming for the same action route. No absent analytic completion or quotient surface can be recovered from that finite packet. □

Falsifiable prediction: Every accepted $\mathsf{DistributionDerivative}^{\uparrow}$ weak-action read factors through a displayed test-action row, input DistributionUp row, integration-by-parts sign row, candidate derivative action row, shared RealUp and RegSeqRatUp readback, StreamName windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: distribution, weak-derivative, sobolev, real, regseqrat, streamname: $\mathsf{DistributionDerivative}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries input distribution action, candidate derivative action, sign handoff, shared readback, finite windows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}$.

Constructive story: A $\mathsf{DistributionDerivative}^{\uparrow}$ packet is generated from finite test-action rows, an input DistributionUp row, an integration-by-parts sign row, a candidate derivative action row, shared RealUp and RegSeqRatUp readback, StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3751.1, Definition 3751.2, and Definition 3751.3 seed the finite distribution-action derivative packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistributionDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution quotients, selected representatives, completed test-function spaces, full PDE regularity, ambient Real completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-action exactness, sign-handoff stability, readback stability, weak-action nonescape, and TasteGate field faithfulness for BEDC.Derived.DistributionDerivativeUp.

Chapter 3752

A Concrete Naming Certificate for $\mathsf{NetClusterPoint}^{\uparrow}$

Following Moore–Smith and Kelley cluster points, the $\mathsf{NetClusterPoint}^{\uparrow}$ packet records cluster-point evidence for a directed net through displayed cofinal tails and neighbourhood hits rather than through a selected subnet or an ambient compactness theorem. It sits over the Moore–Smith net carrier of chapter 3403, the directed-index surface of chapter 1607, the cofinal subnet route of chapter 1600, the convergence interface of chapter 1667, and the compactness handoff of chapter 3660. The chapter names the finite BEDC landing for cluster-point evidence consumed by later net and compactness arguments, not a quotient of nets, a choice-selected subnet, or a host-level limit principle.

Definition 3752.1 (Net cluster-point carrier)**.** *A* $\mathsf{NetClusterPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, D, S, T, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *source row, naming the directed net, its displayed value windows, and the finite tail ledger consumed by every cluster-point read.*
2. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row, exposing the upper-window witnesses needed to compare tail requests.*
3. *S is the* $\mathsf{DirectedSubnet}^{\uparrow}$ *cofinal-tail row. It records the cofinal reindexing evidence through which a cluster-point consumer may follow a subnet-shaped tail without selecting an external subnet representative.*
4. *T is the* $\mathsf{NetConvergence}^{\uparrow}$ *neighbourhood-hit row. It records that each displayed neighbourhood request is met along the cofinal tail route before a convergence-facing consumer is exposed.*
5. *L is the* $\mathsf{LimitPointCompact}^{\uparrow}$ *handoff row, available only after the Moore–Smith source, directed index, cofinal-tail, and neighbourhood-hit rows have been displayed.*
6. *H records componentwise* hsame *transport for the Moore–Smith net row, directed-index row, cofinal-tail row, neighbourhood-hit row, compactness handoff, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a cluster-point request through M, D, S, T, L, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{NetClusterPoint}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{DirectedSubnet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{LimitPointCompact}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$ *row.*

The carrier contains no quotient-net coordinate, no selected cluster point, no countable-choice subnet selector, no host neighbourhood filter, and no ambient compactness or completeness theorem.

Lean4 ✓ `BEDC.Derived.NetClusterPointUp.NetClusterPointTasteGate single carrier alignment`

Theorem 3752.2 (Net cluster-point NameCert obligations)**.** *For every accepted* $\mathsf{NetClusterPoint}^{\uparrow}$ *carrier $K = (M, D, S, T, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Moore–Smith source, directed-index row, cofinal-tail row, neighbourhood-hit row, compactness handoff, transport, replay, provenance, and local naming rows;*
2. *source-tail exposure, requiring each cluster-point request to read the* $\mathsf{MooreSmithNet}^{\uparrow}$ *value window and* $\mathsf{DirectedSet}^{\uparrow}$ *upper-window witnesses before any neighbourhood-hit row is consumed;*
3. *cofinal-tail exactness, requiring the* $\mathsf{DirectedSubnet}^{\uparrow}$ *row S to expose only displayed cofinal reindexing data from the same directed source;*
4. *neighbourhood-hit routing, requiring T to route every cluster-point neighbourhood request through the* $\mathsf{NetConvergence}^{\uparrow}$ *interface before the compactness handoff is visible;*
5. *compactness handoff, requiring L to consume only the displayed cluster route supplied by M, D, S, T before any* $\mathsf{LimitPointCompact}^{\uparrow}$ *consumer receives the packet;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

Consequently the packet supplies finite Moore–Smith cluster-point evidence and exports no quotient of nets, selected subnet, host neighbourhood filter, countable choice, or ambient compactness theorem.

Proof. Project the accepted carrier of Definition 3752.1. Its analytic rows are M, D, S, T, L, and its structural rows are H, C, P, N.

The source-tail exposure is read from M and D. A cluster-point consumer first names the displayed Moore–Smith net window, then reads the directed upper-window witness that places the request inside the finite tail ledger.

The cofinal route is fixed by S. Since S is a $\mathsf{DirectedSubnet}^{\uparrow}$ row over the same directed source, it contributes only displayed cofinal-tail evidence and cannot introduce a selected subnet representative or a countable-choice selector.

The neighbourhood-hit row T is downstream of that cofinal route. It uses the $\mathsf{NetConvergence}^{\uparrow}$ interface to expose only the finite neighbourhood hits visible along the displayed tail, so no host neighbourhood filter or quotient-net equality is available.

Lean4 ✓ `BEDC.Derived.NetClusterPointUp.NetClusterPoint neighbourhood hit netconvergence dependency`

Finally L hands the route to $\mathsf{LimitPointCompact}^{\uparrow}$, while H, C, P, N transport, replay, source, and name the same finite packet. The carrier inventory has no stronger compactness or completeness coordinate, so the local certificate is exhausted by the stated obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NetClusterPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3752.1 and Definition 3752.2 seed the finite cluster-point route through MooreSmithNet, DirectedSet, DirectedSubnet, NetConvergence, LimitPointCompact, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{NetClusterPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected subnets, countable-choice selectors, host neighbourhood filters, ambient compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-tail exposure, cofinal-tail exactness, neighbourhood-hit routing, compactness handoff, transport, replay, provenance, and local naming for BEDC.Derived.NetClusterPointUp.

Constructive story: A NetClusterPoint$^{\uparrow}$ packet is generated from a MooreSmithNet source row, a DirectedSet index row, a DirectedSubnet cofinal-tail row, a NetConvergence neighbourhood-hit row, a LimitPointCompact handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3753

A Concrete Naming Certificate for $\mathsf{GreedySpectralClosure}^{\uparrow}$

The $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet realizes the block-optimal spectral closure route of Definition 9462.1419 as finite BEDC NameCert data. It reads the active codon support, fixed-budget spectral frontiers, anchor and median layers, and symmetric inactive completion from the DNA-window spectral closure vision, then records only the finite trigger-spectrum rows needed by downstream finite-window and trigger-algebra consumers.

The trigger-spectrum route consumes the finite observation discipline of chapter 689, the boolean trigger rows of chapter 3704, the window-coverage trigger algebra of chapter 3478, and the stabilization surface of chapter 2270. Thus the packet is not a kernel-only spectral table: its support, trigger, and layered-closure rows are read as sibling finite-window and trigger-algebra data before the local NameCert row can be consumed.

Definition 3753.1 (Greedy spectral closure carrier)**.** *A* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, A, M, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the active support row for $\mathcal{R}$ and the finite complement inside Q_6.*
2. *A is the anchor-block row naming the unique three-point spectral maximizer* $\{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$.
3. *M is the median-block row naming the next three-point spectral maximizer* $\{\mathrm{AUG}, \mathrm{UCG}, \mathrm{UUG}\}$.
4. *T is the terminal singleton row naming* ACG *as the next single-point spectral maximizer.*
5. *R is the radius table row carrying the displayed killed-walk spectral values and multiplicity counts.*
6. *L is the layered-closure row saying that the packet is block-optimal rather than a nested fixed-budget optimum from $\mathcal{R}$.*
7. *H records componentwise* hsame *transport for support, anchor, median, terminal, radius, layered-closure, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a trigger-spectrum request through S, A, M, T, R, L, H.*
9. *P is the* Pkg *provenance over* $\mathsf{GreedySpectralClosure}^{\uparrow}$, $\mathsf{FiniteWindowObserver}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{SpectralStabilization}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, A, M, T, R, L, H, C, P, N$. It has no coordinate for a host Perron-root oracle, biological-code truth, stochastic evolution model, quotient codon semantics, or unbounded spectral search.

Lean4 (definition) `BEDC.Derived.GreedySpectralClosureUp.GreedySpectralClosureUp`

Theorem 3753.2 (Greedy spectral closure NameCert obligations)**.** *For every accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier* $G = (S, A, M, T, R, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$ *surface consists exactly of finite support admission, anchor-block exposure, median-block exposure, terminal singleton exposure, spectral table exposure, layered-closure exposure, componentwise transport, displayed replay, provenance, and local naming.*

The Lean follow-up for this packet should provide Nontrivial and FieldFaithful instances for BEDC.Derived.GreedySpectralClosureUp so that $S, A, M, T, R, L, H, C, P, N$ remain separated carrier fields.

Proof. Project the carrier of Definition 3753.1. The rows S, A, M, T, R, L expose the finite support, the three chosen blocks, the spectral table, and the layered-closure reading before any consumer can cite the packet.

The spectral data are finite table rows. They record the displayed optima and multiplicities from the vision file, but they do not introduce a Perron-root oracle or an unbounded search procedure.

The structural rows H, C, P, N preserve that route. Since no host biological truth predicate, stochastic model, quotient codon equality, hidden spectral computation, or unbounded enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Theorem 3753.3 (Greedy spectral closure trigger nonescape)**.** *Every trigger-spectrum consumer of an accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *packet factors through*

$$S \longmapsto A, M, T \longmapsto R, L \longmapsto H, C, P, N.$$

The route may cite the finite anchor block, median block, terminal singleton, displayed spectral values, and layered-closure row. It cannot export a biological-code oracle, a stochastic evolutionary law, an ambient powerset closure, quotient codon equality, or an unbounded spectral optimizer.

Proof. Start with the finite support row S. It fixes the active support and complement inside the displayed codon window, so no later read can range over an unbounded search space.

Read the rows A, M, T next. They name the anchor block, median block, and terminal singleton as the finite closure path consumed by trigger-spectrum clients.

The table and layered rows R, L identify the route as block-optimal and budget-frustrated. They do not change the packet into a global optimizer, because the carrier stores only the displayed values and multiplicities.

Finally H, C, P, N transport, replay, source, and name the same route. The absent host objects listed in the statement are not coordinates of the carrier and cannot be exported by a consumer read. □

Theorem 3753.4 (Greedy spectral closure sibling trigger route)**.** *Let* $G = (S, A, M, T, R, L, H, C, P, N)$ *be an accepted* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *carrier. Every trigger-spectrum consumer of G first reads the finite-window observer, boolean trigger lattice, window-coverage trigger algebra, and spectral-stabilization sibling rows recorded in P before it consumes the displayed rows*

$$S, A, M, T, R, L, H, C, P, N.$$

Consequently the packet cannot be consumed as an isolated spectral kernel detached from finite observation, trigger algebra, coverage, or stabilization surfaces.

Proof. Project the provenance row P in Definition 3753.1. It names $\mathsf{FiniteWindowObserver}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, and $\mathsf{SpectralStabilization}^{\uparrow}$ as part of the same accepted packet source.

The finite-window observer fixes the public observation row from which the active support S is read. The boolean trigger lattice and window-coverage trigger algebra then supply the trigger rows against which the anchor, median, and terminal rows A, M, T are interpreted.

Only after those sibling rows are in place can the displayed spectral table and layered-closure rows R, L be consumed. The structural rows H, C, P, N transport, replay, source, and name that route, so no consumer can bypass the sibling finite-window and trigger-algebra surfaces while still reading G as an accepted $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet. □

Falsifiable prediction: An accepted $\mathsf{GreedySpectralClosure}^{\uparrow}$ trigger-spectrum read is refuted if the anchor block, median block, terminal singleton, displayed spectral table, layered-closure row, transport, replay, provenance, or local naming can be removed while preserving the same packet claim.

Independence witness: finite-window-observer, boolean-trigger-lattice, window-coverage-trigger-algebra, spectral-stabilization, dna-window spectral closure vision: none is bijective with $\mathsf{GreedySpectralClosure}^{\uparrow}$ because the packet jointly carries support, anchor, median, singleton, spectral table, layered closure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}$.

Constructive story: A $\mathsf{GreedySpectralClosure}^{\uparrow}$ packet is generated from active support rows, anchor-block rows, median-block rows, terminal singleton rows, displayed spectral table rows, layered-closure rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3753.1, Definition 3753.2, and Definition 3753.3 seed the finite trigger-spectrum closure packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GreedySpectralClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, ambient powerset closure, quotient codon equality, hidden spectral computation, unbounded optimization, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-spectrum exactness, displayed-table stability, Nontrivial and FieldFaithful instances, and trigger nonescape for BEDC.Derived.GreedySpectralClosureUp.

Chapter 3754

A Concrete Naming Certificate for $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ packet names the finite comparison surface between two displayed Cauchy-complete ordered-field presentations of a Real-style object. It reads the terminal $\mathsf{Real}^{\uparrow}$ seal, regular rational sequence rows, dyadic arithmetic ledger, algebra-order rows, the $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ interface, and the $\mathsf{CompleteArchimedeanField}^{\uparrow}$ capstone as input packets, then records only the BEDC route that compares their NameCert-visible rows. The chapter exports uniqueness as rowwise route agreement, not as quotient field equality, trichotomy, a selected limit, or a host complete ordered field.

Definition 3754.1 (Cauchy-complete ordered-field uniqueness carrier)**.** *A* $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K_0, K_1, R_0, R_1, Q_0, Q_1, D, A_0, A_1, F_0, F_1, G, H, C, P, N)$$

with the following displayed rows:

1. *K_0 and K_1 are two accepted* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *packets whose Real-facing reads are to be compared.*
2. *R_0 and R_1 are the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows exposed by K_0 and K_1.*
3. *Q_0 and Q_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows used before the terminal Real seals are consumed.*
4. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger for precision scales, rational windows, and tolerance transfer.*
5. *A_0 and A_1 are the* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation-and-order rows displayed by the two packets.*
6. *F_0 and F_1 are the* $\mathsf{CompleteArchimedeanField}^{\uparrow}$ *capstone rows reached from the two Cauchy-complete ordered-field presentations.*
7. *G is the BEDC comparison route, recording componentwise agreement of the regular-rational windows, dyadic tolerances, Real seals, algebra-order rows, Cauchy-complete ordered-field rows, and complete Archimedean capstone rows.*
8. *H records componentwise* hsame *transport for all displayed rows of U.*
9. *C records displayed* Cont *replay from a comparison request through $Q_0, Q_1, D, R_0, R_1, A_0, A_1, K_0, K_1, F_0, F_1, G, H$*

10. *P is the* Pkg *provenance over* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchimedeanField$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted uniqueness packets only through these rows. It has no coordinate for quotient field equality, trichotomy, a selected limit, a chosen isomorphism outside G, or an ambient host complete ordered field.

Lean4 ✓ `BEDC.Derived.CauchyCompleteOrderedFieldUniquenessUp.CauchyCompleteOrderedFieldUniquenessTasteGa`
`single carrier alignment`

Theorem 3754.2 (Cauchy-complete ordered-field uniqueness sibling dependency route)**.** *Every accepted* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$ *carrier is a dependency consumer of the Real and Completion lattice. The two terminal seal rows R_0, R_1 are read through chapter 38; the readback rows Q_0, Q_1 are read through chapter 301; the shared tolerance ledger D is read through chapter 291; the operation-and-order rows A_0, A_1 are read through chapter 766; the comparison packets K_0, K_1 are read through chapter 3688; and the capstone rows F_0, F_1 are read through chapter 2025. Hence the uniqueness packet is not self-standing: its rowwise comparison route is grounded in those sibling chapters before any consumer may cite the uniqueness surface.*

Proof. Project the carrier U of Definition 3754.1. Its visible comparison data begins with the two regular rational readbacks Q_0, Q_1 and the shared dyadic ledger D, so a consumer must first pass through the RegSeqRat and DyadicRatCore sibling rows that make the tolerance comparison meaningful.

The Real seals R_0, R_1 are then consumed as terminal name rows, not as quotient-field equality. The algebra-order rows A_0, A_1 supply the operation-order surface used by the two Cauchy-complete ordered-field packets K_0, K_1.

Finally the complete Archimedean capstone rows F_0, F_1 and the comparison route G record agreement along the displayed BEDC path. The structural rows H, C, P, N preserve and name that path, so no host complete ordered field, selected isomorphism, trichotomy, or quotient equality can bypass the listed sibling dependencies. □

Theorem 3754.3 (Cauchy-complete ordered-field uniqueness NameCert obligations)**.** *For every accepted* CauchyCompleteOrderedFieldUniqueness$^{\uparrow}$ *carrier*

$$U = (K_0, K_1, R_0, R_1, Q_0, Q_1, D, A_0, A_1, F_0, F_1, G, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$ *surface consists exactly of the two displayed* CauchyCompleteOrderedField$^{\uparrow}$ *packets, their* Real$^{\uparrow}$ *seal rows, their* RegSeqRat$^{\uparrow}$ *readback rows, the shared* DyadicRatCore$^{\uparrow}$ *comparison ledger, their* RealAlgOrder$^{\uparrow}$ *operation-and-order rows, their* CompleteArchimedean *capstone rows, the BEDC comparison route G, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3754.1. The projection exposes exactly the two Cauchy-complete ordered-field packets, the Real seals, regular-rational readbacks, dyadic comparison ledger, algebra-order rows, complete Archimedean capstone rows, comparison route, transport, replay, provenance, and local naming.

The comparison route first reads Q_0, Q_1 against the shared dyadic ledger D. This places every tolerance and rational-window comparison before the terminal Real seals R_0, R_1 are consumed.

It then compares the ordered-field surfaces through A_0, A_1 and the completion-facing surfaces through K_0, K_1, F_0, F_1. The row G records exactly this BEDC route and not an external field equality or selected isomorphism.

Finally H, C, P, N preserve, replay, source, and name the same displayed rows. Since quotient field equality, trichotomy, selected limits, and host complete ordered fields have no coordinate in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3754.4 (Cauchy-complete ordered-field uniqueness route)**.** *Every accepted* CauchyCompleteOrderedFieldU *comparison between two displayed Cauchy-complete ordered-field packets factors through*

$$Q_0, Q_1, D \longmapsto R_0, R_1 \longmapsto A_0, A_1, K_0, K_1 \longmapsto F_0, F_1 \longmapsto G, H, C, P, N.$$

Thus a consumer may cite a finite BEDC route from regular rational readback and dyadic tolerance comparison through Real seals, algebra-order rows, Cauchy-complete ordered-field rows, and complete Archimedean capstone rows. The consumer receives no quotient field equality, trichotomy principle, selected limit, chosen external isomorphism, or host complete ordered field.

Proof. Start with the regular-rational side. The rows Q_0, Q_1 are the only readbacks available before Real sealing, and the shared ledger D supplies the dyadic precision and comparison data for both packets.

The route next reaches R_0, R_1. Only after these Real seals are displayed may the comparison inspect the algebra-order rows A_0, A_1 and the two $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ packets K_0, K_1.

The capstone comparison is then made through F_0, F_1, the $\mathsf{CompleteArchimedeanField}^{\uparrow}$ rows attached to the two presentations. The row G records that the two presentations agree along this displayed BEDC path.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Because no excluded equality, trichotomy, selected-limit, external-isomorphism, or host-field coordinate appears anywhere in the carrier, every uniqueness read factors through the finite route stated above. □

Theorem 3754.5 (Cauchy-complete ordered-field uniqueness regular-Cauchy dependency)**.** *Every accepted* $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ *packet must route its completion witness through the regular-Cauchy completion surface of chapter 1513 and the terminal Real seal of chapter 38 before it compares ordered-field endpoints. The ordered-field comparison therefore reads the displayed completion and real-name rows first; it does not introduce a host complete ordered field, quotient equality, or a selected external isomorphism.*

Proof. Begin with an accepted carrier U. Its only completion-facing source rows are the regular-rational readbacks Q_0, Q_1, the dyadic comparison ledger D, the Real seals R_0, R_1, and the Cauchy-complete ordered-field rows K_0, K_1.

The separated regular-Cauchy completion sibling supplies the finite completion route consumed by those rows, while the Real chapter supplies the terminal seal that the ordered-field comparison is allowed to inspect. Thus the comparison cannot jump directly from K_0, K_1 to endpoint equality.

Only after that route is displayed may the algebra-order rows A_0, A_1 and capstone rows F_0, F_1 be read. The comparison row G records agreement along this finite path, so no host complete ordered field, quotient equality, or external isomorphism is available. □

Falsifiable prediction: Every $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ comparison read factors through two CauchyCompleteOrderedField packets, two Real seals, two RegSeqRat readbacks, one DyadicRatCore comparison ledger, two RealAlgOrder rows, two CompleteArchimedeanField rows, one BEDC comparison route, transport, continuation replay, provenance, and local naming.

Independence witness: real, regseqrat, dyadicratcore, real-alg-order, cauchy-complete-ordered-field, complete-archimedean-field: $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two displayed complete ordered-field presentations and a rowwise comparison route between them.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}$ packet is generated from two CauchyCompleteOrderedField packets, two Real seals, two RegSeqRat readbacks, a DyadicRatCore comparison ledger, two RealAlgOrder rows, two CompleteArchimedeanField rows, one BEDC comparison route, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3754.1, Definition 3754.3, and Definition 3754.4 seed the finite comparison surface for two displayed Cauchy-complete ordered-field presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompleteOrderedFieldUniqueness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert quotient field equality, trichotomy, selected limits, a chosen external isomorphism, a host complete ordered field, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, and the rowwise BEDC comparison route for BEDC.Derived.CauchyCompleteOrderedFieldUniquenessUp.

Chapter 3755

A Concrete Naming Certificate for $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$

Following Bishop and Bridges constructive metric completion theory, the $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ packet records the finite route by which a uniformly controlled metric map induces a map between displayed completion packets. It sits after the metric completion carrier of chapter 1506, the functor-facing Cauchy completion surface of chapter 423, the universal extension ledger of chapter 397, the dense-extension route of chapter 3475, the regular rational readback of chapter 301, and the finite stream-window interface of chapter 126. The packet names functoriality only as displayed source, target, map, Cauchy-control, completion-entry, universal-property, comparison, transport, replay, provenance, and local NameCert rows; it is not a quotient-completion equality, a selected limit map, function extensionality, countable choice, or ambient completeness.

Definition 3755.1 (MetricCompletionFunctoriality carrier)**.** *A* $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, G, A, E, U, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row whose completion-facing names are inspected.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose completion packet receives the induced read.*
3. *G is the finite graph row of the underlying map from source names to target names.*
4. *A is the Cauchy-control ledger, recording the modulus or window transfer by which* $\mathsf{StreamName}^{\uparrow}$ *observations of X yield* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks over Y.*
5. *E is the completion-entry row, naming the* $\mathsf{MetricCompletion}^{\uparrow}$ *source and target entries used by the induced map.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row through which the extension and uniqueness read is consumed.*
7. *S is the separated comparison row, requiring induced target reads to compare only through the displayed separated metric or dense-extension surface.*
8. *H records componentwise* hsame *transport for the source metric, target metric, graph, Cauchy-control, completion-entry, universal-property, separated-comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a source completion request through G, A, E, U, S, H, P, N.*
10. *P is the* Pkg *provenance over* MetricCompletion$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, DenseIsometryExtension$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, Met$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetricCompletionFunctoriality}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient completions, selected representatives of Cauchy classes, host function equality, countable-choice selection, an ambient completed metric space, or a second extension principle outside U.

```
Lean4 ✓  BEDC.Derived.MetricCompletionFunctorialityUp.MetricCompletionFunctorialityTasteGate
single carrier alignment
```

Theorem 3755.2 (MetricCompletionFunctoriality NameCert obligations)**.** *For every accepted* MetricCompletionFunctoriality$^{\uparrow}$ *carrier $F = (X, Y, G, A, E, U, S, H, C, P, N)$, the local* NameCert$_{\text{MetricCompletionFunctoria}}$ *surface consists exactly of carrier admission for the source metric, target metric, finite map graph, Cauchy-control ledger, completion-entry row, universal-property row, separated comparison row, componentwise transport, continuation replay, provenance, and local naming. Every induced completion read must pass through the* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *control recorded in A, the* MetricCompletion$^{\uparrow}$ *entry row E, the* CompletionUniversalProperty$^{\uparrow}$ *ledger U, and the separated comparison row S before the local name N is exposed.*

Proof. Project the carrier of Definition 3755.1. The rows $X, Y, G, A, E, U, S, H, C, P, N$ are the whole public packet, so source and target metric data, the map graph, the Cauchy-control ledger, completion entries, universal-property data, and separated comparison are all displayed before the induced map is consumed.

The Cauchy-control row supplies the constructive handoff. A source completion request first reads finite StreamName$^{\uparrow}$ windows, then the RegSeqRat$^{\uparrow}$ readback governed by A, and only then reaches the completion-entry row E.

The universal-property and comparison rows bind the extension. The row U records the extension-and-uniqueness ledger, while S keeps target comparison inside the displayed separated surface rather than in host equality or a quotient-completion principle.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for selected representatives, choice, ambient completeness, host function equality, or a detached extension principle, the listed rows exhaust the local NameCert obligations. □

Theorem 3755.3 (MetricCompletionFunctoriality composition)**.** *Let*

$$F = (X, Y, G, A, E, U, S, H, C, P, N)$$

and

$$F' = (Y, Z, G', A', E', U', S', H', C', P', N')$$

be accepted MetricCompletionFunctoriality$^{\uparrow}$ *carriers whose middle metric row Y and completion-entry row compare by the displayed* hsame *transport. The composed completion read is the finite replay obtained by first consuming G, A, E, U, S and then consuming G', A', E', U', S'. It is sourced by the same* CauchyCompletionFunctoriality$^{\uparrow}$ *and* MetricCompletion$^{\uparrow}$ *rows named in P and P', and it exports no host-level function composition, quotient-completion equality, selected limit representative, countable choice, or ambient completeness theorem.*

Proof. Start with the two carriers. The first packet exposes the route from X to Y through its finite graph, Cauchy-control ledger, completion-entry row, universal-property row, and separated comparison row; the second exposes the same kind of route from Y to Z.

Use the displayed hsame transport on the shared middle rows. This identifies the target metric and completion-entry read of the first packet with the source metric and completion-entry read of the second packet only through the finite transport rows already present in H and H'.

Replay the two continuation rows in order. The composed route consumes C and then C', so every source completion read still passes through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, and separated comparison rows before the final local name is available.

The nonescape boundary is inherited from the carrier inventory. Neither packet contains host function equality, quotient-completion equality, selected representatives, countable choice, or ambient completeness, and sequential replay of displayed rows cannot create such a coordinate. □

Falsifiable prediction: Any accepted $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ consumer that maps a source completion name to a target completion name without displaying the finite graph row, Cauchy-control ledger, completion-entry row, universal-property row, and separated comparison row refutes the carrier surface.

Independence witness: metriccompletion, cauchycompletionfunctor, completionuniversalproperty, dense-isometry-extension, regseqrat, streamname: these sibling chapters supply completion entries, functorial Cauchy routes, universal extension ledgers, dense separated comparison, regular readbacks, and finite windows separately, while this packet binds them into one functorial carrier $F = (X, Y, G, A, E, U, S, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionFunctoriality}^{\uparrow}$ packet is generated from source and target metric rows, a finite map graph, a Cauchy-control ledger, a completion-entry row, a universal-property row, a separated comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3755.1, Definition 3755.2, and Definition 3755.3 seed the finite functoriality surface for metric completion maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionFunctoriality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives of Cauchy classes, host function extensionality, countable-choice selection, ambient metric completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-control routing, completion-entry exactness, universal-property replay, separated comparison, transport, provenance, and local naming for BEDC.Derived.MetricCompletionFunctoriali

Chapter 3756

A Concrete Naming Certificate for NaturalDeduction$^{\uparrow}$

Following Gentzen and Prawitz, the NaturalDeduction$^{\uparrow}$ packet records a finite proof-tree calculus whose rows expose assumptions, rule applications, discharge, conclusion readback, and proof-term boundary data without importing host proof equality. It sits beside the syntax-and-deduction surface of chapter 197, the proof-term carrier of chapter 200, the bounded computation packet of chapter 199, and the run-trace surface of chapter 280. The packet is a finite NameCert neighbour: it names proof-tree evidence, but it does not identify proofs by normalization, beta equality, semantic validity, or machine execution.

Definition 3756.1 (NaturalDeduction carrier)**.** *A* NaturalDeduction$^{\uparrow}$ *carrier is a finite proof-tree* BHist *packet*

$$D = (A, R, S, Q, B, T, H, K, P, N)$$

with the following displayed rows:

1. *A is the assumption row, listing the open formula endpoints admitted at the leaves;*
2. *R is the rule row, recording introduction, elimination, weakening, and structural rule instances as finite local steps;*
3. *S is the discharge schedule, pairing discharged assumption occurrences with the rule rows that close them;*
4. *Q is the conclusion row, exposing the formula endpoint read at the root of the tree;*
5. *B is the proof-term boundary row, recording when a proof tree may be compared with a* LambdaCalc$^{\uparrow}$ *term without quotienting proof identity;*
6. *T is the transport row, comparing proof-tree nodes only by displayed* hsame *routes over formula, assumption, rule, discharge, and conclusion rows;*
7. *H records componentwise* hsame *stability for the assumption, rule, discharge, conclusion, proof-term boundary, transport, replay, provenance, and local name rows;*
8. *K records* Cont *replay from assumptions through rule rows and discharge rows to the conclusion and proof-term boundary;*
9. *P is the* Pkg *provenance over* FirstOrder$^{\uparrow}$, LambdaCalc$^{\uparrow}$, Computable$^{\uparrow}$, TuringMachine$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$ *row.*
The carrier has no coordinate for host proof equality, global proof normalization, semantic validity, executable machine trace, selected proof representative, or quotient proof identity.

Lean4 ✓ `BEDC.Derived.NaturalDeductionUp.TasteGate.NaturalDeductionTasteGate single carrier alignment`

Theorem 3756.2 (NaturalDeduction NameCert obligations)**.** *For the carrier of Definition 3756.1, the local* $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$ *surface consists exactly of carrier admission for* $D = (A, R, S, Q, B, T, H, K, P, N)$ *assumption exposure through* A*, rule-step exposure through* R*, discharge exposure through* S*, conclusion readback through* Q*, proof-term boundary exposure through* B*, transport through* T*, componentwise* hsame *stability through* H*,* Cont *replay through* K*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier of Definition 3756.1. The rows $A, R, S, Q, B, T, H, K, P, N$ are the only coordinates from which the local certificate may read proof-tree data.

The proof-tree content is exposed by the assumption, rule, discharge, and conclusion rows. A consumer can inspect open hypotheses, local rule instances, discharged occurrences, and the root conclusion, but those reads remain inside the finite tree packet.

The proof-term boundary row is separate from proof identity. It records the visible point at which a $\mathsf{LambdaCalc}^{\uparrow}$ term may be compared with a tree boundary, while forbidding any quotient that identifies proof trees by normalization or beta equality.

The structural rows then provide transport, replay, provenance, and naming. Since the carrier contains no further field, the listed rows exhaust the NameCert obligations. □

Theorem 3756.3 (NaturalDeduction FirstOrder consumer route)**.** *Every* $\mathsf{FirstOrder}^{\uparrow}$ *consumer of an accepted* $\mathsf{NaturalDeduction}^{\uparrow}$ *packet reads only the displayed formula endpoints, assumption rows, rule rows, discharge schedule, conclusion row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row of Definition 3756.1. It cannot replace the proof-tree route by host syntax equality, semantic validity, model-theoretic completeness, or an external proof checker.*

Proof. Start with a $\mathsf{FirstOrder}^{\uparrow}$ read of the accepted carrier D. The package row P names $\mathsf{FirstOrder}^{\uparrow}$ as the syntax and formula source, so the formula endpoints available to the consumer are exactly the endpoints carried by the assumption and conclusion rows.

Rule and discharge reads pass through R and S. These rows record finite local proof steps and the assumptions closed by those steps; they do not add a semantic model, a completeness theorem, or a host proof oracle.

Finally replay T, H, K, P, N. Transport and continuation preserve the same displayed proof-tree rows, and the local name row registers that route as the $\mathsf{NaturalDeduction}^{\uparrow}$ certificate. No external proof checker has a coordinate from which it could be projected. □

Theorem 3756.4 (NaturalDeduction LambdaCalc proof-term boundary)**.** *The* $\mathsf{LambdaCalc}^{\uparrow}$ *comparison surface of an accepted* $\mathsf{NaturalDeduction}^{\uparrow}$ *packet is exactly the proof-term boundary row* B *together with the assumption, rule, discharge, conclusion, transport, replay, provenance, and local naming rows of Definition 3756.1. It supplies a boundary for Curry–Howard-style comparison, but it does not identify natural-deduction proof trees with lambda terms, alpha classes, beta-normal forms, or machine traces.*

Proof. Read the boundary row B from the carrier. Its purpose is to name the visible point where a proof tree may be compared with the $\mathsf{LambdaCalc}^{\uparrow}$ term surface.

The comparison still depends on the proof-tree rows. The assumption row A, rule row R, discharge row S, and conclusion row Q determine which proof boundary is being compared, while T, H, K preserve and replay those rows componentwise.

The provenance row P may cite $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, and $\mathsf{TuringMachine}^{\uparrow}$ as nearby surfaces. This citation does not add alpha quotienting, beta-normal equality, host program equality, or a machine run trace to the natural-deduction packet.

Thus the chapter supplies a boundary for proof-term comparison only. Any stronger identification would require a separate carrier coordinate, and none is present in Definition 3756.1. □

Falsifiable prediction: If a proof-tree packet claims $\mathsf{NaturalDeduction}^{\uparrow}$ acceptance while omitting assumption rows, rule rows, discharge schedule, conclusion readback, proof-term boundary, transport, replay, provenance, or local NameCert rows, it refutes the carrier of Definition 3756.1.

Independence witness: firstorder, lambdacalc, computable, turingmachine: $\mathsf{NaturalDeduction}^{\uparrow}$ is not bijective to any listed sibling because it carries assumption rows, rule rows, discharge schedule, conclusion readback, and a proof-term boundary as a single proof-tree certificate. FirstOrderUp carries syntax and deduction endpoints without the separate proof-term boundary; LambdaCalcUp carries terms and beta-substitution ledgers without discharged proof assumptions; ComputableUp and TuringMachineUp carry bounded computation and run traces rather than finite proof-tree discharge.

Closure status for $\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}$.

Constructive story: A $\mathsf{NaturalDeduction}^{\uparrow}$ packet is generated from assumption rows, local rule rows, a discharge schedule, a conclusion row, a proof-term boundary row, transport, componentwise hsame stability, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3756.1, Definition 3756.2, Definition 3756.3, and Definition 3756.4 seed the finite proof-tree carrier and its consumer boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NaturalDeduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host proof equality, global proof normalization, semantic validity, Curry–Howard equivalence, alpha quotienting, beta-normal equality, Lean verification, bridge checking, or executable machine-trace identity for $\mathsf{NaturalDeduction}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, proof-tree rule routing, discharge exactness, FirstOrder consumer nonescape, and LambdaCalc proof-term boundary coverage for BEDC.Derived.NaturalDeductionUp.

Chapter 3757

A Concrete Naming Certificate for $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real arithmetic, the $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ packet records finite sums of displayed regular Cauchy rational names as folded addition routes. It sits after the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, the binary addition packet of chapter 1695, the terminal real-name seal of chapter 38, and the finite-product sibling of chapter 2009. The carrier records displayed folded addition data, not host finite sums, quotient streams, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3757.1 (Regular Cauchy finite sum carrier)**.** *A* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, S, W, D, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite unary index row. It lists exactly the summand slots opened by the request and the displayed fold order used by the addition route.*
2. *S is the component* $\mathsf{RegSeqRat}^{\uparrow}$ *source family, one regular rational Cauchy name for each slot of I.*
3. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for component reads, intermediate addition reads, and the final sum-facing read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *sum-budget ledger, storing component observations, intermediate sum observations, requested radii, and slack cells.*
5. *A is the folded* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route row. It records the adjacent binary additions named by I, W, D and exposes no host finite-sum operation.*
6. *R is the final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the source, window, dyadic, and folded-addition rows have been displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback of the folded sum has been displayed.*
8. *H records componentwise* hsame *transport for the index row, summand sources, synchronized windows, dyadic ledger, folded-addition route, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from any finite-sum request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyAddition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, W, D, A, R, E, H, C, P, N$. It has no coordinate for a host finite sum, quotient stream equality, selected limit, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFiniteSumUp.RegularCauchyFiniteSumTasteGate single carrier alignment`

Theorem 3757.2 (Regular Cauchy finite sum NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier*

$$K = (I, S, W, D, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the finite unary index row, component* $\mathsf{RegSeqRat}^{\uparrow}$ *summands, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger, folded* $\mathsf{RegularCauchyAddition}^{\uparrow}$ *route, final readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local name;*

2. *finite tuple-window coverage, requiring every summand slot in I to open the matching source row and all synchronized windows needed by the fold;*

3. *dyadic sum-budget exactness, requiring every intermediate addition observation to consume its assigned radius and slack cells from D;*

4. *RegSeqRat readback stability, requiring the final readback row R to depend only on the displayed sources, windows, dyadic ledger, and folded-addition route;*

5. *Real seal non-escape, requiring E, H, C, P, N to expose no host finite sum, quotient stream equality, selected limit, trichotomy, ambient completeness theorem, Lean verification, or bridge credential.*

No Regular Cauchy finite sum obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3757.1. The displayed tuple $I, S, W, D, A, R, E, H, C, P, N$ is the whole public surface, so carrier admission is exactly the admission of those rows.

Tuple-window coverage is forced by the index row. Each slot named by I has one component $\mathsf{RegSeqRat}^{\uparrow}$ source in S, and the synchronized window row W is the only place where component, intermediate, and final sum reads are opened.

The dyadic budget is read through D. Each intermediate binary addition request consumes a displayed radius and slack cell from that ledger before the corresponding folded $\mathsf{RegularCauchyAddition}^{\uparrow}$ row A can be inspected.

The final regular readback row R is downstream of S, W, D, A. Therefore readback stability follows from the carrier order: no sum-facing $\mathsf{RegSeqRat}^{\uparrow}$ data can be inspected until the synchronized windows, dyadic budget, and folded addition route have already been displayed.

Finally, the seal, transport, replay, provenance, and local name rows preserve only this route. Since none of the excluded objects is a coordinate of the carrier, the listed five rows exhaust the local NameCert obligations. □

Theorem 3757.3 (Regular Cauchy finite sum budget route)**.** *Let*

$$K = (I, S, W, D, A, R, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *carrier. Every finite sum read from* K *factors through*

$$I, S \longmapsto W \longmapsto D, A \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, regrouping displayed adjacent additions may change only the fold order named by I *and* A*; it preserves the same synchronized windows* W*, dyadic sum-budget ledger* D*, final* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E *when those rows are fixed by the accepted carrier.*

Proof. Begin with a finite sum request. The request first names the finite unary index row I and the component source family S, because these are the only carrier rows that identify which regular Cauchy names participate in the fold.

The requested component and intermediate observations then pass through W. This synchronized window row aligns component reads with the intermediate addition reads used by the displayed fold.

The dyadic part of the request is read next through D, and the adjacent binary additions are read through A. The proof uses the displayed $\mathsf{RegularCauchyAddition}^{\uparrow}$ rows rather than host associativity of sums.

Only after those rows are displayed can the final regular rational readback R be consumed. The downstream rows E, H, C, P, N seal, transport, replay, source, and name the same route, so the finite sum read cannot bypass the synchronized windows, the dyadic ledger, or the folded-addition route. □

Falsifiable prediction: Every $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ sum-facing read factors through a finite unary index row, component RegSeqRat sources, synchronized StreamName windows, DyadicRatCore sum-budget observations, a folded RegularCauchyAddition route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-addition, real, regular-cauchy-finite-product: $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite unary summand index, synchronized multi-source windows, dyadic sum-budget accounting, folded addition rows, final RegSeqRat readback, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3757.1, Definition 3757.2, and Definition 3757.3 seed the finite sum route from displayed regular Cauchy sources through synchronized windows, dyadic sum budgets, folded addition rows, final RegSeqRat readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host finite sums, quotient stream equality, selected limits, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tuple-window coverage, dyadic budget exactness, folded-addition routing, readback stability, seal non-escape, and local naming for BEDC.Derived.RegularCauchyFiniteSumUp.

Constructive story: A $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ packet is generated from a finite unary summand index row, component RegSeqRat source rows, synchronized StreamName windows, DyadicRatCore sum-budget observations, a folded RegularCauchyAddition route, final RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3758

A Concrete Naming Certificate for $\mathsf{FiniteSeriesPermutation}^{\uparrow}$

Following Hardy, Courant, and Bishop and Bridges on finite series and constructive finite sums, the $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ packet records the finite invariance route by which a displayed summand list may be reindexed without changing the BEDC sum read. It sits after the permutation carrier of chapter 116, the finite list spine of chapter 36, the addition row of chapter 29, the unary index discipline of chapter 28, and the folded regular Cauchy sum surface of chapter 3757. The packet is finite certificate data; it does not expose an infinite rearrangement principle or a host summation operator.

Definition 3758.1 (Finite series permutation carrier)**.** *A* $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, L, \Pi, A, W, F, T, H, C, P, N)$$

with the following displayed rows.

1. *I is the* $\mathsf{Nat}^{\uparrow}$ *index-window row naming the finite interval of summand slots.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *summand spine, with one displayed summand row for each slot named by I.*
3. *Π is the* $\mathsf{Permutation}^{\uparrow}$ *witness row on the same finite index window.*
4. *A is the* $\mathsf{Add}^{\uparrow}$ *finite-fold route, recording the adjacent binary additions used by the source order and by the permuted order.*
5. *W is the window ledger that pairs each source summand slot with its permuted slot and records the finite reads used by both folds.*
6. *F is the folded-sum comparison row, read either as a bare finite* $\mathsf{Add}^{\uparrow}$ *route or through a* $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ *consumer when the summands are regular Cauchy names.*
7. *T is the terminal transport row saying that the two displayed fold outputs are classified by the same downstream sum classifier.*
8. *H records componentwise* hsame *transport for the index window, list spine, permutation witness, addition rows, comparison row, terminal transport, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteSeriesPermutation}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Add}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{RegularCauchyFiniteSum}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSeriesPermutation}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. There is no coordinate for an infinite series, a conditionally convergent rearrangement, a host finite-sum function, a quotient sequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteSeriesPermutationUp.FiniteSeriesPermutationTasteGate single carrier alignment`

Theorem 3758.2 (Finite series permutation NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ *carrier*

$$S = (I, L, \Pi, A, W, F, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSeriesPermutation}^{\uparrow}}$ *surface consists exactly of carrier admission, finite index-window coverage, list-slot coverage, permutation-witness coverage, addition-fold compatibility, folded-sum comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations are exhausted by the finite rows of Definition 3758.1.*

Proof. Project the accepted carrier. The rows $I, L, \Pi, A, W, F, T, H, C, P, N$ are the whole visible packet, so carrier admission is just admission of those rows and of their local naming row.

The finite coverage obligations are read in order. The $\mathsf{Nat}^{\uparrow}$ row I fixes the inspected slots, the $\mathsf{List}^{\uparrow}$ row L supplies exactly one summand row at each slot, and the $\mathsf{Permutation}^{\uparrow}$ row Π supplies the source-to-target slot matching together with its inverse witness.

The compatibility obligation then reads A, W, F, T. The addition route A lists the two displayed folds, W records which summand reads are paired by Π, F compares the two folded outputs, and T transports that comparison to the downstream classifier that the consumer is allowed to inspect.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since none of the excluded objects is a carrier coordinate, no further NameCert obligation is available outside the listed rows. □

Theorem 3758.3 (Finite series permutation sum invariance)**.** *Let*

$$S = (I, L, \Pi, A, W, F, T, H, C, P, N)$$

be an accepted $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ *carrier. Every consumer that reads the source-order finite fold and the* Π*-permuted finite fold must factor through*

$$I, L, \Pi \longmapsto W, A \longmapsto F, T \longmapsto H, C, P, N.$$

Hence the two finite sums are classified together by the displayed folded-sum comparison row, and the conclusion is a BEDC certificate over finite data rather than a host equality of summation functions.

Proof. Begin with the finite source order. The requested slots are exactly the slots displayed by I, and the summands are exactly the entries of L at those slots. No consumer can name an additional summand because the list-slot coverage row is bounded by I.

Read the permutation witness Π next. Its forward and inverse rows pair the source slots with the permuted slots in the same finite window, so each permuted fold input is already a displayed entry of L.

The two folds are then read through A. The source-order fold and the permuted fold may use different adjacent binary addition traces, but both traces consume only the slot reads named by W and the addition rows named by A.

The comparison row F is therefore a finite folded-sum comparison over the same multiset of displayed summand rows. The terminal row T transports this comparison to the permitted downstream sum classifier, including the $\mathsf{RegularCauchyFiniteSum}^{\uparrow}$ classifier when the summands are regular Cauchy names.

The remaining rows H, C, P, N keep the route stable under componentwise transport, replay the same finite read, record provenance, and supply the local name. Thus the invariant exported by the packet is exactly finite permutation invariance for the displayed fold route, with no appeal to infinite rearrangement, quotient sequence equality, host finite sums, or $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ sum-invariance read exposes a NatUp index window, a ListUp summand spine, a PermutationUp witness, AddUp fold rows, a paired read window, folded-sum comparison, terminal classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before the permuted sum is consumed.

Independence witness: permutation, list, add, nat, regular-cauchy-finite-sum: $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite index window, summand spine, permutation witness, paired fold routes, folded-sum comparison, terminal transport, replay, provenance, and local naming required for sum-invariance reads.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSeriesPermutation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSeriesPermutation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3758.1, Definition 3758.2, and Definition 3758.3 seed the finite permutation-invariance route from displayed index windows and summand lists through permutation witnesses, addition folds, folded-sum comparison, terminal transport, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSeriesPermutation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite rearrangements, conditionally convergent series, host finite sums, quotient sequences, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite index coverage, list-slot coverage, permutation-witness coverage, fold compatibility, folded-sum comparison, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSeriesPermutationUp.

Constructive story: A $\mathsf{FiniteSeriesPermutation}^{\uparrow}$ packet is generated from a NatUp finite index window, a ListUp summand spine, a PermutationUp witness, AddUp finite-fold rows, paired fold windows, folded-sum comparison, terminal classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3759

A Concrete Naming Certificate for $\mathsf{EpsilonDeltaLimit}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet records the traditional epsilon-delta limit interface as finite BEDC data. It sits between the source and target metric carriers of chapter 57, the dyadic tolerance surface of chapter 291, the finite-window names of chapter 126, the regular rational readback rows of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a displayed point, a proposed target limit, positive dyadic epsilon requests, matching positive dyadic delta witnesses, finite window reads, regular-rational readback, transport, replay, provenance, and local naming; it does not identify host reals, select limits, form quotient streams, invoke trichotomy, use choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3759.1 (Epsilon-delta limit carrier)**.** *An* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, a, L, E, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including its accepted point classifier and distance-witness surface.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including its accepted point classifier and* $\mathsf{Real}^{\uparrow}$*-valued distance-witness surface.*
3. *a is the source point row at which the limiting read is taken.*
4. *L is the proposed target-limit row, read as a displayed point of Y before any Real-facing equality question is asked.*
5. *E is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *epsilon request row for the target distance to L.*
6. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *delta witness row for source distance to a.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source point, the candidate values, the proposed limit, and the two dyadic tolerance rows.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and for the* $\mathsf{Real}^{\uparrow}$ *distance rows reached through X and Y.*
9. *H records componentwise* hsame *transport for the metric rows, point row, proposed limit, epsilon request, delta witness, windows, readback, continuation, provenance, and local name.*

10. *C records displayed* Cont *replay from epsilon requests through delta witnesses, metric-distance reads, finite windows, regular-rational readback, and the target* $\mathsf{Real}^{\uparrow}$ *seal.*
11. *P is the* Pkg *provenance over* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$ *row.*

Lean4✓ `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitTasteGate single carrier alignment`

The classifier accepts the packet only when all rows name the same finite epsilon-delta route: every accepted epsilon request in E is met by the displayed delta row D, and every Real-facing target-distance read passes through $X, Y, a, L, E, D, W, R, H, C, P, N$.

Theorem 3759.2 (Epsilon-delta limit NameCert obligations)**.** *For the carrier of Definition 3759.1, the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (X, Y, a, L, E, D, W, R, H, C,$ source and target metric-space admission through X and Y, point admission through a, proposed-limit admission through L, positive dyadic epsilon and delta admission through E and D, finite-window admission through W, regular-rational and Real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The metric rows X and Y display the source and target point classifiers and the distance witnesses before the point row a or the proposed target limit L can be consumed by a limit read.

The tolerance part is finite. The epsilon request E and delta witness D are $\mathsf{DyadicRatCore}^{\uparrow}$ rows, so the smallness comparison is a dyadic ledger rather than a host real inequality or selected limit object.

The analytic readback is downstream of those rows. The $\mathsf{StreamName}^{\uparrow}$ window row W records the finite observations used by the source point, target values, proposed limit, and dyadic tolerances, while the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the displayed windows before any $\mathsf{Real}^{\uparrow}$ distance seal is exposed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host real equality, quotient streams, selected limits, trichotomy, choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3759.3 (Epsilon-delta limit Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ *packet reads the limit assertion through the finite route*

$$X, Y, a, L, E, D, W, R, H, C, P, N.$$

The consumer first sees the source and target $\mathsf{Met}^{\uparrow}$ *rows, the point row, the proposed limit row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *epsilon request and delta witness, the* $\mathsf{StreamName}^{\uparrow}$ *windows, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback. Only then may it read the target* $\mathsf{Real}^{\uparrow}$ *distance seal carried by the target metric row, and it receives no host real equality, quotient stream, selected limit, trichotomy, choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4✓ `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitRealSealNonescape`

Proof. Use Definition 3759.2. It orders every Real-facing read after X, Y, a, L, E, D, W, R, so the target distance to the proposed limit is visible only as a readback routed through the displayed metric, dyadic, window, and regular-rational rows.

The target metric row Y is the only source of the $\mathsf{Real}^{\uparrow}$-valued distance seal. Because E and D are displayed $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, the consumer cannot replace the certificate by a host epsilon quantifier, a hidden delta chooser, or a quotient representative of a target real.

The structural rows preserve the same route. Componentwise transport H, replay C, provenance P, and local naming N carry the finite packet forward without adding coordinates for host equality, selected limits, trichotomy, choice, quotient streams, or ambient completeness. □

Falsifiable prediction: Every epsilon-delta limit read of an $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet factors through displayed metric rows, dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, and the terminal RealUp distance seal.

Independence witness: metric, dyadicratcore, streamname, regseqrat, real: EpsilonDeltaLimitUp is not bijective to any listed sibling because it jointly carries source and target MetricSpace rows, a point row, a proposed limit row, dyadic epsilon and delta rows, finite windows, regular-rational readback, Real-facing distance seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}$.

Constructive story: An $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ packet is generated from source and target MetricSpace rows, a point row, a proposed limit row, dyadic epsilon requests, dyadic delta witnesses, finite StreamName windows, RegSeqRat readback, Real-facing distance seals, hsame transport, Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3759.1, Definition 3759.2, and Definition 3759.3 seed the finite epsilon-delta limit interface for Phase Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.EpsilonDeltaLimitUp.EpsilonDeltaLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, selected limits, trichotomy, choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance handling, finite-window readback, target Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EpsilonDeltaLimitUp.

Chapter 3760

A Concrete Naming Certificate for $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ packet records local epsilon-delta continuity as finite BEDC data. It sits beside the source and target metric rows of chapter 57, the displayed $\mathsf{Real}^{\uparrow}$ distance seals, the graph row of chapter 130, the modulus surface of chapter 1631, and the limit-facing packet of chapter 3759. The packet names only a displayed point, an epsilon request, a delta witness, finite metric-window reads, continuity transport, replay, provenance, and local naming; it does not assert classical open-cover compactness, host function equality, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3760.1 (Epsilon-delta continuity carrier)**.** *An* $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, F, a, E, D, W, R, M, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier and source-distance witness surface.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, including the target-distance witness surface and its terminal* $\mathsf{Real}^{\uparrow}$ *distance seal.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the named map, exposing only the finite graph and target-window reads used by the continuity certificate.*
4. *a is the displayed source point at which the continuity read is taken.*
5. *E is the positive target-tolerance row, read through the dyadic or regular rational tolerance ledger before any target-distance request is made.*
6. *D is the positive source-radius witness row attached to E and to the point a.*
7. *W is the finite window row that inspects source points within the displayed D-neighbourhood and the corresponding target graph cells of F.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source and target distance windows.*
9. *M is the* $\mathsf{ModulusContinuity}^{\uparrow}$ *handoff row when the local continuity read is consumed by a modulus-facing route.*
10. *L is the* $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ *boundary row witnessing that the local read is compatible with the limit packet at a without replacing continuity by a selected limit.*

11. *H records componentwise* hsame *transport for the metric rows, graph row, point row, tolerance rows, windows, readback, modulus handoff, limit boundary, replay, provenance, and local name.*
12. *C records displayed* Cont *replay from tolerance requests through source-radius witnesses, metric-distance windows, graph reads, regular-rational readback, the modulus handoff, and the limit boundary.*
13. *P is the* Pkg *provenance over* $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{EpsilonDeltaLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaContinuity}^{\uparrow}}$ *row.*

The classifier accepts the packet only when every tolerance read factors through the same finite route $X, Y, F, a, E, D, W, R, M, L, H, C, P, N$.

Theorem 3760.2 (Epsilon-delta continuity NameCert obligations)**.** *For the carrier of Definition 3760.1, the local* $\mathsf{NameCert}_{\mathsf{EpsilonDeltaContinuity}^{\uparrow}}$ *surface consists exactly of source and target metric admission through X and Y, Real-valued distance readback through the target metric seal, continuous graph admission through F, point admission through a, positive tolerance and radius admission through E and D, finite-window admission through W, regular readback through R, modulus-continuity handoff through M, epsilon-delta limit compatibility through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3760.1. The source metric row X, target metric row Y, displayed graph row F, and point row a are explicit coordinates, so a continuity consumer cannot begin from an ambient function or an unshown host point.

The tolerance part is local. The target request E and source-radius witness D are paired before the finite window row W is consumed. Thus the target-distance read reaches the $\mathsf{Real}^{\uparrow}$ seal of Y only after the source-neighbourhood window and the graph cells of F have been displayed.

The two sibling handoffs keep the packet connected to the surrounding metric route. The $\mathsf{ModulusContinuity}^{\uparrow}$ row M exposes the same tolerance-to-window data to modulus-facing consumers, while the $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ row L records compatibility with the limit packet at a without adding a selected limit or quotient real equality.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for classical open-cover compactness, host function equality, quotient real equality, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3760.3 (Epsilon-delta continuity local route)**.** *Every local continuity read of an accepted* $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{ModulusContinuity}^{\uparrow},\ \mathsf{EpsilonDeltaLimit}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},$$

The consumer receives a displayed tolerance request, a displayed source-radius witness, finite source and target windows, regular readback, a modulus handoff, and an epsilon-delta limit boundary. It receives no classical open-cover compactness, host function equality, quotient real equality, global uniform-continuity claim, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Use the NameCert obligations of Definition 3760.2. They force the local read to begin with X, Y, F, a, E, D, W, R before either handoff row is inspected.

Read the $\mathsf{ModulusContinuity}^{\uparrow}$ coordinate M only after the finite tolerance-to-window data have been displayed. This supplies the modulus-facing consumer with the same local budget, not a detached global uniform-continuity object.

Read the $\mathsf{EpsilonDeltaLimit}^{\uparrow}$ coordinate L only as a boundary row tied to the same point a and graph row F. It records limit compatibility for the local route and does not insert a chosen limit, quotient representative, or host real equality.

The structural rows H, C, P, N preserve this order under transport and replay. Since no excluded external coordinate appears in the carrier, every accepted continuity read is confined to the listed finite route. □

Theorem 3760.4 (Epsilon-delta sequential Cauchy handoff)**.** *Let K be an accepted* $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ *packet. If the source stream of K is read through a displayed* $\mathsf{StreamName}^{\uparrow}$ *row and the image windows are read through a displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *row, then the local radius ledger supplied by Definition 3760.3 may feed the* $\mathsf{CauchySequentialContinuity}^{\uparrow}$ *preservation surface before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is inspected.*

The handoff is a finite same-packet route through the source-window, radius-ledger, image-window, readback, transport, replay, provenance, and local-name rows. It does not assert host sequential continuity, selected subsequences, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the local route theorem Definition 3760.3. It exposes the displayed tolerance request, source-radius witness, finite source and target windows, regular readback, modulus handoff, and epsilon-delta limit boundary before any Real-facing consumer reads the packet.

Now read the sequential dependencies. The $\mathsf{StreamName}^{\uparrow}$ source row supplies the finite inspected windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular-rational readback consumed by the Cauchy image route of chapter 4045. The terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is therefore reached only after the preservation surface has seen the local radius ledger and image windows.

The construction uses only the displayed rows named in this chapter together with chapter 126, chapter 301, and chapter 38. Since the route has no coordinate for host sequential continuity, selected subsequences, quotient stream equality, countable choice, or ambient completeness, none of those endpoints is exported. □

Falsifiable prediction: Every epsilon-delta continuity read of an $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ packet factors through displayed MetricSpace rows, a ContinuousMap graph row, finite tolerance and radius rows, RegSeqRat readback, a ModulusContinuity handoff, and an EpsilonDeltaLimit boundary.

Independence witness: metricspace, real, continuousmap, moduluscontinuity, epsilondeltalimit: $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source and target metric rows, a graph row, a point row, paired tolerance and radius rows, finite windows, regular readback, modulus handoff, limit boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{EpsilonDeltaContinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{EpsilonDeltaContinuity}^{\uparrow}$ packet is generated from source and target MetricSpace rows, a Real-valued target-distance seal, a ContinuousMap graph row, a displayed source point, target-tolerance and source-radius rows, finite windows, RegSeqRat readback, ModulusContinuity handoff, EpsilonDeltaLimit boundary, hsame transport, Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3760.1, Definition 3760.2, Definition 3760.3, and Definition 3760.4 seed the local epsilon-delta continuity interface for metric, continuous-map, and sequential Cauchy consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EpsilonDeltaContinuity}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close classical open-cover compactness, host function equality, quotient real equality, global uniform continuity, countable choice, trichotomy, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-radius routing, finite-window readback, modulus handoff, limit-boundary compatibility, transport, replay, provenance, and local naming for BEDC.Derived.EpsilonDeltaContinuity

Chapter 3761

A Concrete Naming Certificate for $\mathsf{PackingNumber}^{\uparrow}$

Following Bishop and Bridges and Kelley metric-space practice, the $\mathsf{PackingNumber}^{\uparrow}$ packet records the finite separated-center count dual to a displayed metric covering tolerance. It sits beside the finite net route of chapter 1767, the upper-count packet of chapter 2187, the scale-control surface of chapter 2035, and the metric source of chapter 57. The packet names a lower-bound quantitative ledger as finite BEDC data, not as a maximal subset, a choice-selected packing, a compactness proof, or an ambient cardinality comparison.

Definition 3761.1 (Metric packing-number carrier)**.** *A* $\mathsf{PackingNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, \varepsilon, U, D, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier, distance classifier, and metric-law ledger used by all separation reads.*
2. *ε is the positive tolerance row, read through* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *together with the displayed positivity evidence needed by separated-center consumers.*
3. *U is the finite closed* $\mathsf{ProbeBundle}$ *of candidate centers admitted for the tolerance ε.*
4. *D is the pairwise distance ledger. For every two distinct displayed centers of U, it records the metric-distance row in X and the comparison witnessing ε-separation.*
5. *B is the unary packing-budget row naming the number of center entries exposed by U and certified by the pairwise ledger D.*
6. *H records componentwise* hsame *transport for the metric source, tolerance row, separated-center bundle, pairwise distance ledger, budget row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from any packing-number request through the same rows $X, \varepsilon, U, D, B, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{PackingNumber}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, \varepsilon, U, D, B, H, C, P, N$. It has no coordinate for an ambient finite subset, host maximality proof, selected saturated packing, powerset cardinality, compactness, total boundedness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PackingNumberUp.TasteGate.PackingNumberTasteGate single carrier alignment`
Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberTasteGate single carrier alignment`

Theorem 3761.2 (Metric packing-number NameCert obligations)**.** *For every accepted* $\mathsf{PackingNumber}^{\uparrow}$ *carrier* $K = (X, \varepsilon, U, D, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, positive tolerance, finite separated-center bundle, pairwise distance ledger, unary packing budget, transport, replay, provenance, and local naming rows;*
2. *center admission, requiring every entry counted by* B *to be visible in the same closed* $\mathsf{ProbeBundle}$ *row* U *and admitted by the metric source row* X*;*
3. *pairwise separation exactness, requiring every two distinct displayed centers to be compared by* D *through the same metric distance and the same tolerance* ε*;*
4. *budget exactness, requiring the exported packing number to count exactly the visible center entries of* U *certified by the pairwise ledger;*
5. *tolerance transport, requiring the* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row to remain attached to the separation comparisons under* H *before any consumer reads the count;*
6. *ledger nonescape, excluding maximal subset selection, host cardinality comparison, powerset enumeration, open-cover compactness, total-bounded completion, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No metric packing-number obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberCarrier namecert obligations`
Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberNamecertObligations`

Proof. Project the accepted carrier of Definition 3761.1. The analytic rows are exactly the metric source X, positive tolerance ε, separated-center bundle U, pairwise ledger D, and unary budget B.

Center admission is rowwise. A center can be counted only after it is visible in U and read as a point of the metric source X.

Pairwise separation is also rowwise. The ledger D supplies each displayed distance comparison between distinct entries of U, and every comparison uses the same positive tolerance row ε.

The budget row then names the quantitative boundary. Because B counts the same visible entries certified by U and D, a consumer cannot replace the packet by a host maximal packing, an external enumeration, or a powerset cardinality argument.

Finally H, C, P, N transport, replay, source, and name exactly this route. Since the carrier contains none of the excluded compactness, choice, completeness, or maximality coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 3761.3 (Metric packing covering-dual handoff)**.** *Every* $\mathsf{MetricCoveringNumber}^{\uparrow}$*,* $\mathsf{FiniteEpsilonNet}^{\uparrow}$*,* $\mathsf{DoublingMetric}^{\uparrow}$*, or compact-metric consumer reading an accepted* $\mathsf{PackingNumber}^{\uparrow}$ *carrier factors through the finite route*

$$X, \varepsilon \longmapsto U, D, B \longmapsto H, C, P, N.$$

The consumer receives a separated-center lower-bound ledger before comparing it with a covering-number or finite-net upper-count ledger. The handoff exports no maximal packing, no selected cover, no open-cover compactness proof, no powerset cardinality comparison, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PackingNumberUp.PackingNumberCoveringDualHandoff`

Proof. Begin with a covering, finite-net, doubling, or compact-metric consumer of K. By Definition 3761.1, the consumer can enter the packet only through the displayed metric source X and the displayed positive tolerance row ε.

The separated data are opened next. The bundle U names the candidate centers, the ledger D certifies pairwise separation through the metric distance row, and B records the resulting unary packing budget.

The covering-dual comparison is external to the packing packet. A covering-number or finite-net consumer may compare its own center budget with B, but only after both packets expose their metric source, tolerance, finite bundle, and distance ledger rows.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Applying Definition 3761.2, the consumer cannot replace the finite separated ledger by a selected maximal packing, compactness argument, powerset cardinality comparison, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3761.1, Definition 3761.2, and Definition 3761.3 seed the quantitative separated-center packet over displayed metric sources, positive rational or dyadic tolerances, finite ProbeBundle centers, pairwise distance ledgers, unary budgets, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PackingNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close maximal packing existence, minimal covering duality, total boundedness, compact metric completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, center admission, pairwise separation exactness, budget exactness, tolerance transport, ledger nonescape, and covering-dual handoff for BEDC.Derived.PackingNumberUp.

Constructive story: A $\mathsf{PackingNumber}^{\uparrow}$ packet is generated from a MetricUp source row, a positive RatUp or DyadicRatCoreUp tolerance, a finite ProbeBundle of separated centers, pairwise metric-distance comparisons, a unary packing budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3762

A Concrete Naming Certificate for $\mathsf{LipschitzFunction}^{\uparrow}$

Following Kelley metric-space practice and Bishop–Bridges constructive analysis, the $\mathsf{LipschitzFunction}^{\uparrow}$ packet records a function only through displayed metric rows, a graph ledger, and a carried bound. It sits between the source and target metric surface of chapter 57, the terminal real-bound seal of chapter 38, the graph discipline of chapter 130, and the tolerance export of chapter 369. The packet is a finite NameCert connector for uniform-continuity and completion-facing consumers; it does not import host function extensionality, quotient map equality, choice of representatives, or ambient completeness.

Definition 3762.1 (Lipschitz function carrier)**.** *A* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, G, K, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row carrying the function-read relation from X to Y.*
3. *K is the nonnegative* $\mathsf{Real}^{\uparrow}$ *Lipschitz-constant row.*
4. *D is the distance-inequality ledger recording every displayed comparison between a source-distance request and the corresponding target-distance response.*
5. *H records componentwise* hsame *transport for source points, target points, graph rows, bound rows, distance ledgers, continuation rows, provenance, and local naming.*
6. *C records displayed* Cont *replay from source metric requests through G, K, D to target metric responses and uniform-modulus reads.*
7. *P is the* Pkg *provenance over* $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$ *row.*

The classifier accepts only packets whose target-distance reads are sourced by G and bounded through the same K, D ledger. It has no coordinate for a host function object, quotient equality of maps, selected representatives, differentiability, compactness, or completeness of either metric space.

Lean4 ✓ `BEDC.Derived.LipschitzFunctionUp.LipschitzFunctionTasteGate single carrier alignment`

Theorem 3762.2 (Lipschitz function NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier* $F = (X, Y, G, K, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, componentwise classifier comparison, Lipschitz-bound exactness, graph-to-distance replay, uniform-modulus export, transport stability, provenance sourcing, and local naming through the displayed rows* $X, Y, G, K, D, H, C, P, N$*.*

No obligation row is stored outside the carrier, and no exported read depends on host function extensionality, quotient map equality, selected representatives, countable choice, differentiability, compactness, or ambient metric completeness.

Proof. Project the carrier. The only analytic rows are the source metric X, target metric Y, graph row G, Lipschitz constant K, and distance-inequality ledger D.

The classifier comparison is componentwise over those rows and over H, C, P, N. Bound exactness follows because a target-distance read has to pass through the graph row and then through the displayed comparison stored in D against the same constant K.

The uniform-modulus export reads no further object. Transport H and replay C preserve the same graph, constant, and inequality rows, while P and N only source and name the route. Since the carrier contains none of the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 3762.3 (Lipschitz function uniform-continuity handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformContinuity}^{\uparrow}$*, or* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier receives its source-tolerance route only after the packet has displayed the source metric* X*, target metric* Y*, graph row* G*, Lipschitz constant* K*, distance-inequality ledger* D*, and replay rows* H, C, P, N*. The handoff is a finite BEDC modulus route, not a rereading of a host Lipschitz proof.*

Proof. Start from the carrier and its NameCert obligations. The graph row G supplies the only function-read surface, while K and D supply the only bound and distance-comparison surface.

A uniform-modulus request therefore cannot bypass the displayed inequality ledger. It first fixes the target-distance request in Y, reads the bound row K, and then uses D to return the corresponding source-distance request in X.

The continuation and transport rows preserve this order. Hence uniform-continuity and continuous-map consumers receive a finite modulus handoff through $X, Y, G, K, D, H, C, P, N$, with no host function proof or selected representative added. □

Theorem 3762.4 (Lipschitz function completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{LipschitzFunction}^{\uparrow}$ *carrier remains confined to the displayed metric rows, graph row, bound row, distance-inequality ledger, transport, replay, provenance, and local name rows. Reading a complete source or target metric in another packet does not turn* F *into a completed function-space object, a quotient of uniformly continuous maps, a selected extension, or a statement about limits outside* $X, Y, G, K, D, H, C, P, N$*.*

Proof. The packet carries only two metric rows and a graph-bound ledger between them. Completion-facing reads may reuse those metric rows, but no completed function-space coordinate is present in the carrier.

The distance ledger D is also local. It records comparisons for the displayed graph row and constant, and replay through C keeps those comparisons attached to the same source and target metric requests.

Thus a completion consumer can transport the finite Lipschitz route but cannot manufacture quotient maps, selected extensions, host limits, or ambient completeness claims. The exported surface remains exactly $X, Y, G, K, D, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3762.1, Definition 3762.2, Definition 3762.3, and Definition 3762.4 seed the finite Lipschitz-function route from metric, graph, real-bound, distance-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function extensionality, quotient map equality, selected extensions, differentiability, compactness, completed function spaces, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier comparison, Lipschitz-bound exactness, graph-to-distance replay, uniform-modulus export, transport stability, provenance sourcing, and local NameCert rows for BEDC.Derived.LipschitzFunctionUp.

Constructive story: A $\mathsf{LipschitzFunction}^{\uparrow}$ packet is generated from source and target metric rows, a ContinuousMap graph row, a RealUp Lipschitz constant, a distance-inequality ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 3763

A Concrete Naming Certificate for $\mathsf{CantorTernarySet}^{\uparrow}$

Following Cantor's middle-third construction and Bishop–Bridges constructive analysis, the $\mathsf{CantorTernarySet}^{\uparrow}$ packet records the traditional ternary description of the Cantor set as finite BEDC data. It refines the retained-prefix and deleted-gap surface of chapter 1530 by making the inspected base-three windows explicit, and it routes every real-facing read through chapter 38, chapter 301, chapter 126, and chapter 291. The packet names only finite ternary windows, triadic endpoint ledgers, gap checks, regular-rational readback, and the terminal Real seal; it does not assert host powersets, infinite branch choice, quotient real equality, measure, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3763.1 (Cantor ternary set carrier)**.** *A* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, D, V, G, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite base-three digit positions currently inspected.*
2. *D is the ternary digit ledger for W, and every accepted digit row is tagged as a retained digit* 0 *or* 2*.*
3. *V is the triadic place-value row, evaluating the inspected prefix as a rational with denominator a displayed power of* 3*.*
4. *G is the deleted-middle gap ledger, listing the middle-third intervals excluded by every inspected non-retained alternative at the displayed depths.*
5. *I is the retained interval row inherited from* $\mathsf{CantorSet}^{\uparrow}$*, with endpoints computed from the same triadic prefix and checked against the gap ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from W, D, V, G, I.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular-rational readback has been displayed.*
8. *H records componentwise* hsame *transport for the window, digit, triadic value, gap, interval, readback, Real seal, continuation, provenance, and local name rows.*
9. *C records displayed* Cont *replay from window requests through the digit ledger, triadic value ledger, gap ledger, retained interval, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorTernarySet}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$ *row.*

The classifier accepts a packet only when $W, D, V, G, I, R, E, H, C, P, N$ are all displayed for the same finite triadic window, the digit ledger uses only retained digits, the gap ledger refuses the corresponding middle thirds, and every $\mathsf{Real}^{\uparrow}$ *consumer enters through R and E. It has no coordinate for a host subset of* $\mathsf{Real}^{\uparrow}$*, selected infinite ternary stream, quotient real equality, endpoint convention selector, measure, open-cover compactness, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CantorTernarySetUp.CantorTernarySetTasteGate single carrier alignment`

Theorem 3763.2 (Cantor ternary set NameCert obligations)**.** *For an accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier*

$$T = (W, D, V, G, I, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$ *surface consists exactly of carrier admission, finite ternary-window classification, retained-digit checking, triadic place-value evaluation, middle-third gap exclusion, retained-interval compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transport, continuation replay, provenance, and local naming through the displayed rows $W, D, V, G, I, R, E, H, C, P,$*

Proof. Carrier admission projects the finite tuple of Definition 3763.1. The window row W fixes the inspected positions, and the digit ledger D is the only source of ternary digit information for those positions.

The retained-digit and triadic-value checks are then internal to the same finite prefix. The row V evaluates exactly the inspected prefix, while the gap row G records the deleted middle thirds rejected by replacing a retained digit with the middle alternative.

The retained interval row I is inherited from the $\mathsf{CantorSet}^{\uparrow}$ surface and is therefore tied to the same prefix and gap ledger. The regular-rational row R and Real seal row E are downstream of that finite data.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate stores host subsets, infinite streams, quotient equality, endpoint selectors, measure, compactness, choice, or ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 3763.3 (Cantor ternary set triadic nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *carrier remains on the finite triadic route*

$$W, D, V, G, I \longmapsto R \longmapsto E$$

together with H, C, P, N. A consumer may inspect retained ternary windows, deleted middle thirds, triadic endpoint values, regular-rational readback, and the terminal Real seal, but it cannot extract a selected infinite ternary branch, a host real subset, quotient equality of real representatives, a measure row, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier. Any real-facing consumer must first enter through the displayed window W, because no digit row outside that finite window is part of the packet.

The digit ledger D, triadic value row V, and gap ledger G determine the retained middle-third decision at the inspected depths. The interval row I packages the same decision as a $\mathsf{CantorSet}^{\uparrow}$ retained-prefix row, so there is no separate branch or endpoint source.

The handoff to real data is forced through R and E. The regular-rational row R is reached only after the finite triadic rows have been displayed, and the Real seal E is downstream of R.

The structural rows H, C, P, N preserve and name this same route. Thus every exported read remains on the finite triadic packet, and the excluded host subset, infinite branch, quotient equality, measure, choice, compactness, and completeness data are not available to the consumer. □

Theorem 3763.4 (Cantor ternary set CantorSet dependency)**.** *Every accepted* $\mathsf{CantorTernarySet}^{\uparrow}$ *packet factors through the retained-prefix and deleted-gap surface of chapter 1530 before any* $\mathsf{Real}^{\uparrow}$*-facing seal is read. After that CantorSet row is displayed, the remaining route uses* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows to reach the terminal real-name boundary, without asserting compactness, measure, host powersets, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3763.1. The row I is the retained interval row inherited from the CantorSet sibling, and it is computed from the same finite triadic prefix and deleted-gap ledger as W, D, V, G.

By Definition 3763.3, every real-facing read remains on the route $W, D, V, G, I \longmapsto R \longmapsto E$. Thus the CantorSet retained-prefix and deleted-gap row is consumed before the regular-rational readback R and Real seal E are available.

The remaining rows R, E, H, C, P, N use the RegSeqRat, StreamName, and DyadicRatCore surfaces named in the carrier provenance. None of those rows supplies compactness, measure, host powersets, or ambient real completeness, so the dependency is exactly the finite sibling consumption stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CantorTernarySet}^{\uparrow}$ packet is generated from finite StreamName windows, retained ternary digit rows, triadic place-value rows, deleted-middle gap rows, a retained CantorSet interval row, RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3763.1, Definition 3763.2, and Definition 3763.3 seed the explicit finite ternary-window carrier for the middle-third Cantor set through retained digits, triadic values, deleted gaps, regular-rational readback, and the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorTernarySet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, selected infinite branches, quotient real equality, endpoint uniqueness, measure, compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained-digit classification, triadic place-value evaluation, gap exclusion, retained-interval compatibility, RegSeqRat readback, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.CantorTernarySetUp.

Chapter 3764

A Concrete Naming Certificate for $\mathsf{CyclicTagSystem}^{\uparrow}$

Following Cook 2004 and Neary–Woods 2007, the $\mathsf{CyclicTagSystem}^{\uparrow}$ packet records cyclic tag systems as finite step-indexed rewriting data between the machine surface of chapter 280, the finite Rule 110 glider-catalog surface of chapter 1045, and the total-language boundary of chapter 9370. The packet names the alphabet, cyclic production list, deletion number, inspected tape window, bounded fuel evaluator, and Rule 110 handoff as displayed BHist rows. It does not import a partial halting function, an unbounded convergence theorem, a host evaluator, or Cook universality as a hidden coordinate.

Definition 3764.1 (Cyclic tag system carrier)**.** *A* $\mathsf{CyclicTagSystem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, P, D, W, F, R, H, C, Q, N)$$

with the following displayed rows:

1. *A is the finite alphabet row, including the distinguished blank or empty-word convention used by the inspected packet.*
2. *P is the cyclic production-list row, exposing the finite list of appendants and the current production index as BEDC data.*
3. *D is the deletion-number row, recording the positive deletion budget used by each accepted step.*
4. *W is the tape-window state row, containing only the displayed prefix, suffix guard, current word window, and step-index ledger.*
5. *F is the bounded fuel evaluator row, naming the finite replay of at most the displayed fuel many cyclic-tag steps.*
6. *R is the Rule 110 handoff row, read through the glider-catalog and cellular rows when a bounded cyclic-tag trace is presented to a Rule 110 consumer.*
7. *H records componentwise* hsame *transport for alphabet, production-list, deletion, tape-window, fuel, handoff, provenance, and local naming rows.*
8. *C records the displayed* Cont *replay from alphabet and production reads to one-step tape updates, from tape updates to bounded fuel evaluation, and from the bounded trace to the Rule 110 handoff row.*

9. *Q is the* Pkg *provenance over* CyclicTagSystem$^{\uparrow}$, TuringMachine$^{\uparrow}$, RuleOneTenGliderCatalog$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the rows $A, P, D, W, F, R, H, C, Q, N$. It has no coordinate for a host partial function, an unbounded halting predicate, an infinite tape quotient, a complete Rule 110 orbit, or a detached universality theorem.

Lean4 ✓ `BEDC.Derived.CyclicTagSystemUp.CyclicTagSystemTasteGate single carrier alignment`

Theorem 3764.2 (Cyclic tag system step-indexed Rule 110 handoff)**.** *For every accepted* CyclicTagSystem$^{\uparrow}$ *carrier $T = (A, P, D, W, F, R, H, C, Q, N)$, every finite fuel read factors through the cyclic-tag evaluator rows A, P, D, W, F before any Rule 110 orbit claim is exported through R. The handoff may cite the* TuringMachine$^{\uparrow}$ *source surface, the* RuleOneTenGliderCatalog$^{\uparrow}$ *finite catalog surface, and the* CellularAutomaton$^{\uparrow}$ *closed-observation route only after the bounded cyclic-tag replay has been displayed. Thus a Rule 110-facing consumer receives a finite step-indexed handoff and receives no halting decision, unbounded convergence theorem, host partial evaluator, or full universality proof.*

Proof. Project the accepted carrier. The alphabet row A, production-list row P, deletion-number row D, and tape-window row W are the only inputs from which a cyclic-tag step can be read.

The bounded evaluator row F is downstream of those rows. A fuel-indexed request must therefore replay the displayed one-step cyclic-tag updates before it can ask for any substrate-facing handoff.

The Rule 110 handoff row R is read only after F. At that point the consumer may cite the Turing-machine source chapter, the finite Rule 110 glider-catalog chapter, and the cellular closed-observation route, but only as sibling certificate surfaces attached to the displayed bounded replay.

Finally inspect H, C, Q, N. These rows transport, replay, source, and name the same bounded route. Since the carrier has no coordinate for halting, unbounded convergence, host partiality, or a complete universality proof, none of those objects can be exported with the Rule 110-facing handoff. □

Falsifiable prediction: Every CyclicTagSystem$^{\uparrow}$ Rule 110-facing read first exposes a finite alphabet row, cyclic production-list row, deletion-number row, tape-window row, bounded fuel evaluator row, transport row, continuation route, provenance row, and local naming row.

Independence witness: turingmachine, rule110glidercatalog, cellularautomaton: CyclicTagSystem$^{\uparrow}$ is not bijective to any listed sibling because it carries cyclic production-list and deletion-number rows together with a bounded tape-window evaluator before the Rule 110 handoff.

Closure status for NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$**.**

Constructive story: A CyclicTagSystem$^{\uparrow}$ packet is generated from a finite alphabet row, cyclic production-list row, deletion-number row, tape-window state row, bounded fuel evaluator row, Rule 110 handoff row, componentwise hsame transports, displayed Cont routes, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$, seed)

Scope closed: Definition 3764.1 and Definition 3764.2 seed the finite cyclic-tag route between total BEDC step-indexing, TuringMachine$^{\uparrow}$, and the Rule 110 glider-catalog surface.

Formal verification: FormalStatus(NameCert$_{\text{CyclicTagSystem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close halting, unbounded convergence, host partiality, full universality, global Rule 110 orbit classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cyclic production-list replay, deletion-step exactness, bounded fuel evaluation, Rule 110 handoff nonescape, provenance, and local NameCert rows for the same packet.

Chapter 3765

A Concrete Naming Certificate for $\mathsf{HausdorffContent}^{\uparrow}$

Following Hausdorff and later constructive metric-measure treatments, the $\mathsf{HausdorffContent}^{\uparrow}$ packet records finite cover content data for metric subsets without importing a completed measure theory. It sits over the metric carrier of chapter 57, the finite measure-facing ledger of chapter 92, the retained-prefix surfaces of chapter 1530 and chapter 3763, and the terminal real-name seal of chapter 38. The packet names only displayed cover rows, rational radius and exponent budgets, refinement transport, content-bound replay, package provenance, and local NameCert evidence.

Definition 3765.1 (Hausdorff content carrier)**.** *A* $\mathsf{HausdorffContent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, A, R, S, U, B, H, C, P, N)$$

with the following displayed rows:

1. *M is the accepted* $\mathsf{Met}^{\uparrow}$ *row whose distance observations are sealed through* $\mathsf{Real}^{\uparrow}$*.*
2. *A is the inspected subset or source row, presented only by finite membership probes inherited from metric, Cantor, or ternary Cantor packets.*
3. *R is the rational radius-budget row, listing finitely many positive rational or dyadic radius bounds.*
4. *S is the rational exponent-budget row, recording the displayed content exponent and the arithmetic route used for each radius power.*
5. *U is the finite metric-cover row, pairing each listed cover center and radius with the source probes it covers.*
6. *B is the content-bound replay row, summing the displayed radius-power bounds into a* $\mathsf{Real}^{\uparrow}$*-sealed upper content value.*
7. *H is the componentwise* hsame *transport row for the metric source, subset probes, radius budgets, exponent budgets, cover rows, content bounds, provenance, and local name.*
8. *C is the displayed* Cont *replay route for cover refinement, radius-budget reuse, exponent-budget reuse, and content-bound replay.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffContent}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{CantorSet}^{\uparrow}$*,* $\mathsf{CantorTernarySet}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$ *row.*

The classifier accepts a packet only when $M, A, R, S, U, B, H, C, P, N$ are displayed for the same finite cover request, every cover row uses a listed radius budget, every content-bound replay uses the listed exponent budget, and every metric, measure-facing, Cantor-facing, or real-facing consumer enters through the displayed rows. It has no coordinate for a host infimum over all covers, a choice-selected cover family, quotient real equality, σ-additive Hausdorff measure, or completed measure theory.

Lean4 ✓ `BEDC.Derived.HausdorffContentUp.HausdorffContentTasteGate single carrier alignment`

Theorem 3765.2 (Hausdorff content NameCert obligations)**.** *For an accepted* $\mathsf{HausdorffContent}^{\uparrow}$ *carrier*

$$Q = (M, A, R, S, U, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source admission through M, finite source-probe admission through A, rational radius-budget admission through R, rational exponent-budget admission through S, finite cover admission through U, content-bound replay through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission projects the packet in Definition 3765.1. The only metric data are in M, and the only source observations are the finite probes in A.

The radius and exponent budgets are displayed before any content value is read. Thus R and S determine the finite arithmetic route for the cover row U and the bound row B.

Transport and replay add no analytic coordinate. The rows H and C preserve the same metric, source, budget, cover, and bound rows, while P and N source and name the packet.

Since no carrier coordinate stores a host infimum, selected cover family, quotient real equality, σ-additive measure, or completed measure theory, the listed rows exhaust the local NameCert obligations. □

Theorem 3765.3 (Hausdorff content finite-cover refinement)**.** *Every refinement-facing read of an accepted* $\mathsf{HausdorffContent}^{\uparrow}$ *packet factors through the displayed route*

$$M, A, R, S, U, B, H, C, P, N.$$

If a finer cover is cited, it must reuse or refine the listed radius-budget row R, exponent-budget row S, finite cover row U, and content-bound replay row B through hsame *transport and displayed continuation. The refinement read cannot cite an ambient cover lattice, a host infimum over covers, a choice-selected minimizing cover, or quotient equality of real content values.*

Proof. Start with the finite cover coordinate. By Definition 3765.1, the row U is the only place where cover centers, radii, and covered source probes are displayed.

The budget rows R and S are upstream of U and B. A refinement can therefore change a cover only by displaying the reused or refined radius and exponent ledgers before the content-bound replay is read.

Use H and C to transport and replay the refinement route. The provenance row P and local name N keep the refined packet attached to the same metric, measure-facing, Cantor-facing, and real-facing source chapters.

The excluded host cover lattice, infimum, selected minimizing cover, and quotient real equality are not packet coordinates, so no refinement-facing consumer can obtain them from $\mathsf{HausdorffContent}^{\uparrow}$. □

Theorem 3765.4 (Hausdorff content Cantor cover handoff)**.** *Every* $\mathsf{CantorSet}^{\uparrow}$ *or* $\mathsf{CantorTernarySet}^{\uparrow}$ *consumer citing* $\mathsf{HausdorffContent}^{\uparrow}$ *receives only a finite cover-content handoff through*

$$A \longmapsto R, S \longmapsto U \longmapsto B$$

together with M, H, C, P, N*. The handoff can record finite retained-prefix probes, ternary-window probes, rational radius and exponent budgets, finite cover rows, and replayed content bounds. It does not supply* σ*-additive Hausdorff measure, a host infimum, a choice-selected cover, quotient real equality, or completed measure theory.*

Proof. Read the Cantor-facing source through A. The source row admits only finite retained-prefix or ternary-window probes inherited from chapter 1530 and chapter 3763.

Next read the budget and cover rows. The rational radius and exponent budgets R, S are displayed before the finite cover row U, and the content-bound replay row B is downstream of that cover.

Finally attach the metric and structural rows. The metric source M supplies the distance observations, while H, C, P, N transport, replay, source, and name the same finite handoff.

The measure-facing provenance does not create a completed measure. Because σ-additivity, host infima, choice-selected covers, quotient real equality, and completed measure theory are absent from the carrier, the Cantor consumer receives only the finite cover-content route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffContent}^{\uparrow}$ packet is generated from metric source rows, finite source probes, rational radius and exponent budgets, finite cover rows, content-bound replay, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3765.1, Definition 3765.2, Definition 3765.3, and Definition 3765.4 seed the finite cover-content carrier for metric, measure-facing, Cantor-facing, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffContent}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close σ-additive Hausdorff measure, host infimum over all covers, choice-selected covers, quotient real equality, completed measure theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget admission, cover admission, refinement transport, content-bound replay, Cantor cover handoff, provenance, and local naming for BEDC.Derived.HausdorffContentUp.

Chapter 3766

A Concrete Naming Certificate for $\mathsf{CauchyTailFilterBornology}^{\uparrow}$

Following Bourbaki and Weil on uniform spaces and bounded structures, the $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ packet records bounded Cauchy-tail filters as finite BEDC rows. It sits between the finite stream schedule of chapter 126, the regular rational source discipline of chapter 301, the bounded-set carrier of chapter 1757, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names tail-bound surfaces for regular Cauchy data; it is not a quotient of streams, a choice-selected bound, trichotomy, or an ambient completeness theorem.

Definition 3766.1 (Cauchy tail filter bornology carrier)**.** *A* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, B, W, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are the only tail observations visible to the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row read through the same windows.*
3. *B is the* $\mathsf{BoundedSet}^{\uparrow}$ *tail-bound row, recording the displayed bound cell for the admitted tail window.*
4. *W is the tail-window row specifying the finite index after which the bound row is read.*
5. *M is the Cauchy-modulus compatibility row, matching the tail window to the tolerance requests carried by the regular sequence.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, B, W, M have been displayed.*
7. *H records componentwise* hsame *transport for schedule, regular source, bounded tail, window, modulus, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tail-bound request through S, R, B, W, M, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$ *row.*

The carrier exports no quotient stream, selected bound, selected tail representative, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.CauchyTailFilterBornologyUp.CauchyTailFilterBornologyTasteGate single carrier alignment`

Definition 3766.2 (Cauchy tail filter bornology classifier)**.** *Two accepted* $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ *carriers*

$$T = (S, R, B, W, M, E, H, C, P, N)$$

and

$$T' = (S', R', B', W', M', E', H', C', P', N')$$

are classified together exactly when the displayed hsame *rows transport* S *to* S'*,* R *to* R'*,* B *to* B'*,* W *to* W'*,* M *to* M'*, and* E *to* E'*, and the displayed* Cont *rows replay the same tail-window-to-bound-to-seal route on both packets. The classifier has no coordinate for equality of quotient streams, a least bound, a selected eventual index, trichotomy, or ambient completeness.*

Theorem 3766.3 (Cauchy tail filter bornology NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailFilterBornolog}$ *carrier* $T = (S, R, B, W, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *tail-window carrier admission, requiring* S, R, W *to display the finite observation window before any tail-bound row is consumed;*
2. *eventual boundedness classifier, requiring* B *to be preserved under the componentwise* hsame *transport of the accepted carrier and replayed by the same* Cont *route;*
3. *Cauchy-modulus compatibility, requiring* M *to attach each bound read to the same regular Cauchy source and finite window;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal handoff, requiring* R *and* E *to remain downstream of the displayed schedule, tail window, bound row, and modulus row;*
5. *ledger non-escape, requiring* H, C, P, N *to expose no quotient stream, selected bound, selected tail representative, trichotomy row, or ambient completeness theorem.*

No Cauchy tail filter bornology obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 3766.1. The displayed tuple $S, R, B, W, M, E, H, C, P, N$ is the whole public surface, so tail-window carrier admission is exactly admission of those rows.

The eventual boundedness classifier is forced by the bound row. A tail-bound consumer can read B only through the finite window W over the stream schedule S and the regular source R, and classifier preservation transports that same row componentwise.

The Cauchy-modulus row supplies the compatibility check. It attaches the displayed bound to the tolerance and tail-window requests already visible in S, R, W, so the read cannot replace the packet by a host-level eventual bound or by a chosen index.

Finally, the Real-facing rows are downstream. The seal row E is reached only after S, R, B, W, M are displayed, and H, C, P, N preserve and name that route. Since the carrier has no coordinate for quotient streams, selected bounds, trichotomy, or ambient completeness, the five listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyTailFilterBornologyUp.CauchyTailFilterBornologyNameCertObligations`

Falsifiable prediction: Every $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ read factors through a finite StreamName schedule, a RegSeqRat source row, a BoundedSet tail-bound row, a finite tail-window row, a Cauchy-modulus compatibility row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, boundedset, real: $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite tail-window schedule, regular Cauchy source, bounded tail row, modulus compatibility, RealUp sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3766.1, Definition 3766.2, and Definition 3766.3 seed the bounded tail-filter surface from finite StreamName windows through RegSeqRat readback, BoundedSet tail bounds, Cauchy-modulus compatibility, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailFilterBornology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected bounds, least bounds, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, eventual boundedness classification, Cauchy-modulus compatibility, RegSeqRat and Real seal handoff, ledger non-escape, and local naming for BEDC.Derived.CauchyTailFilterBornologyUp.

Constructive story: A $\mathsf{CauchyTailFilterBornology}^{\uparrow}$ packet is generated from finite StreamName windows, a RegSeqRat source row, a BoundedSet tail-bound row, a finite tail-window row, Cauchy-modulus compatibility, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3767

A Concrete Naming Certificate for SequentialCompletenessCriterion$^{\uparrow}$

Following Bishop–Bridges and Kelley, the SequentialCompletenessCriterion$^{\uparrow}$ packet records the finite criterion by which sequence completion evidence is read as real sequential completeness data. It sits after the sequence-completion surfaces of chapter 1683 and chapter 3691, reads the real-facing seal of chapter 1544, and keeps the metric criterion aligned with chapter 1515.

Definition 3767.1 (SequentialCompletenessCriterion carrier)**.** *A* SequentialCompletenessCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, C, R, M, W, G, E, H, T, P, N)$$

with the following displayed rows:

1. *S is the* SequentialCompletion$^{\uparrow}$ *row naming the displayed sequence-completion interface consumed by the criterion.*
2. *C is the* CauchySequenceCompletion$^{\uparrow}$ *row naming the regular Cauchy sequence completion route available to the same finite packet.*
3. *R is the* RealSequentialCompleteness$^{\uparrow}$ *row, exposed only after the sequence-completion and Cauchy-sequence completion rows have been displayed.*
4. *M is the* SequentiallyCompleteMetric$^{\uparrow}$ *row, recording the metric criterion surface through which later metric consumers read sequential completeness.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row for the sequence observations inspected by the criterion.*
6. *G is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed stream windows and Cauchy schedules.*
7. *E is the* Real$^{\uparrow}$ *seal row reached from the regular readback and real sequential completeness rows.*
8. *H records componentwise* hsame *transport for S, C, R, M, W, G, E.*
9. *T records displayed* Cont *replay from sequence windows through Cauchy completion, metric criterion data, regular readback, and the real seal.*
10. *P is the* Pkg *provenance over* SequentialCompletenessCriterion$^{\uparrow}$, SequentialCompletion$^{\uparrow}$, CauchySequenceCompleti RealSequentialCompleteness$^{\uparrow}$, SequentiallyCompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite criterion packet. It has no coordinate for quotient completion, countable choice, host equality of sequences or completion points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialCompletenessCriterionUp.SequentialCompletenessCriterionTasteGate single carrier alignment`

Theorem 3767.2 (SequentialCompletenessCriterion NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompletenessCriterion}^{\uparrow}$ *carrier* $Q = (S, C, R, M, W, G, E, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentialCompletenes}}$ *surface consists exactly of carrier admission for the sequence-completion row, Cauchy-sequence completion row, real sequential-completeness row, sequentially complete metric row, stream-window row, regular readback row, real seal row, componentwise transport, displayed replay, provenance, and local naming. Any real sequential-completeness consumer must read*

$$S, C, W, G, R, M, E, H, T, P, N$$

before it can export a criterion result, and the export contains no quotient completion, countable choice, host equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3767.1. The rows $S, C, R, M, W, G, E, H, T, P, N$ are the only rows available to a consumer of the local name.

The sequence-facing part of the criterion is supplied by S and C. The finite observation schedule is supplied by W, and the regular rational readback is supplied by G. Thus the criterion never begins from a host sequence or from a quotient completion object.

The real-facing part is then forced through R, M, E. The real sequential-completeness row and the sequentially complete metric row name the criterion surface, while the $\mathsf{Real}^{\uparrow}$ seal can be read only after the displayed regular readback route has been consumed.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the packet contains no row for countable choice, host equality, open-cover compactness, quotient completion, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.SequentialCompletenessCriterionUp.SequentialCompletenessCriterionCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompletenessCriterion}^{\uparrow}$ packet is generated from a $\mathsf{SequentialCompletion}^{\uparrow}$ row, a $\mathsf{CauchySequenceCompletion}^{\uparrow}$ row, a $\mathsf{RealSequentialCompleteness}^{\uparrow}$ row, a $\mathsf{SequentiallyCompleteMetric}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 3767.1 and Definition 3767.2 seed the finite criterion surface connecting sequence completion, Cauchy sequence completion, metric sequential completeness, regular readback, and real sequential completeness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialCompletenessCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient completion, countable choice, host equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-completion handoff, Cauchy-sequence completion readback, metric criterion exposure, real seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 3768

A Concrete Naming Certificate for $\mathsf{RegularModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularModulus}^{\uparrow}$ packet records the explicit modulus row carried by a regular Cauchy rational name before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the terminal seal. It sits between the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the regular Cauchy naming surface of chapter 371, and the real-name seal of chapter 38. The packet is the finite modulus object consumed by Real and Completion routes; it does not choose limits, quotient streams, or host real equality.

Definition 3768.1 (Regular modulus carrier)**.** *A* $\mathsf{RegularModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, \mu, D, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row whose rational windows are already admitted through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*.*
2. *μ is the unary modulus row assigning, to each displayed precision request, a finite stream-window bound for the source row.*
3. *D is the dyadic tolerance ledger translating precision requests into* $\mathsf{DyadicRatCore}^{\uparrow}$ *radii and comparison budgets.*
4. *W is the finite window route showing that the window selected by μ is sufficient for the requested tolerance.*
5. *H records componentwise* hsame *transport for the source row, modulus row, dyadic ledger, finite window route, continuation replay, package provenance, and local name.*
6. *C records displayed* Cont *replay from a precision request through S, μ, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularModulus}^{\uparrow}$*,* $\mathsf{RegularCauchyName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the modulus row, dyadic tolerance ledger, and finite window route are displayed for the same source regular Cauchy row. It has no coordinate for

a selected limit, quotient stream equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularModulusUp.RegularModulusTasteGate single carrier alignment`

Theorem 3768.2 (Regular modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularModulus}^{\uparrow}$ *carrier* $M = (S, \mu, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$ *surface consists exactly of source regular Cauchy admission, unary modulus display, dyadic tolerance translation, finite window sufficiency, componentwise transport, continuation replay, package provenance, and local naming through the rows* S, μ, D, W, H, C, P, N*.*

Proof. Project the carrier of Definition 3768.1. The source row S is the only regular Cauchy data exposed by the packet, and the row μ is the only modulus coordinate attached to that source.

The dyadic ledger D translates each precision request into the tolerance language used by chapter 291. The window route W then records that the bound supplied by μ is sufficient for the inspected $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Finally H, C, P, N preserve, replay, source, and name the same displayed rows. Since selected limits, quotient streams, host real equality, countable choice, and ambient completeness have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3768.3 (Regular modulus RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularModulus}^{\uparrow}$ *packet must read the source regular Cauchy row through*

$$S, \mu, D, W \longmapsto H, C, P, N$$

before the regular-rational readback or terminal real-name seal is consumed. Thus the modulus is a displayed finite BEDC row on the Real and Completion route, not a hidden choice of limit, quotient representative, or host equality witness.

Lean4 ✓ `BEDC.Derived.RegularModulusUp.RegularModulusRegSeqRatHandoff`

Proof. Start with a consumer request for regular-rational or real-name data. The packet exposes the source only as S, so the consumer must read the unary modulus row μ and the dyadic tolerance ledger D before it can identify a sufficient finite window.

The row W supplies that finite window route. It is already tied to the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ surfaces named in the provenance row, and it reaches the $\mathsf{Real}^{\uparrow}$ seal only after the displayed tolerance check has been replayed.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. No excluded limit, quotient, host-equality, choice, or ambient-completeness coordinate is available, so every handoff remains on the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularModulus}^{\uparrow}$ packet is generated from a source regular Cauchy row, a unary modulus row, a dyadic tolerance ledger, a finite window route, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$, seed)

Scope closed: Definition 3768.1, Definition 3768.2, and Definition 3768.3 seed the finite modulus surface for regular Cauchy rational names on the Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not assert completeness, choice of limits, quotient streams, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus sufficiency, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularModulusUp.

Chapter 3769

A Concrete Naming Certificate for $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ packet records the constructive finite-subcover surface obtained when a located real interval is paired with displayed finite-net and local-coverage data. It consumes the located endpoint and radius discipline of chapter 394, the finite-net replacement surface of chapter 3471, and the terminal real-name boundary of chapter 38. The packet names explicit extraction of a finite subcover from a finite mesh route; it is not general open-cover compactness, sequential compactness, countable choice, or a host Heine-Borel theorem.

Definition 3769.1 (Compact located interval subcover carrier)**.** *A* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, N, C, M, E, H, R, P, L)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row, carrying the displayed endpoint names, radius budget, and locatedness reads.*
2. *N is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *row, carrying a finite mesh, mesh-scale ledger, and finite-net coverage route for the same interval.*
3. *C is the local coverage row: each finite mesh cell is paired with an admitted cover token and the finite witness that the cell is covered at the displayed tolerance.*
4. *M is the extraction row listing the selected finite cover tokens in the order read from the finite net.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located interval, finite net, and local coverage rows have been displayed.*
6. *H records componentwise* hsame *transport for interval, net, local coverage, extracted token list, seal, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from an interval-subcover request through I, N, C, M, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open-cover compactness, sequential compactness without choice-like data, selected quotient representatives, trichotomy, countable choice, host interval equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactLocatedIntervalSubcoverUp.CompactLocatedIntervalSubcoverTasteGate single carrier alignment`

Theorem 3769.2 (Compact located interval subcover obligations)**.** *For every accepted* CompactLocatedIntervalSub *carrier* $S = (I, N, C, M, E, H, R, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$ *surface consists exactly of located interval admission through* I*, finite-net admission through* N*, local mesh-cell coverage through* C*, explicit finite token extraction through* M*, terminal real-name sealing through* E*, componentwise transport through* H*, displayed replay through* R*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 3769.1. The rows I, N, C, M, E expose the located interval, finite mesh, local coverage, extracted token list, and real-name seal before any consumer can read a subcover claim.

The obligation surface is finite. The only coverage evidence available to the packet is the row C, whose entries are indexed by the finite mesh row N and whose selected tokens are exactly the entries displayed in M.

Transport and replay do not add compactness principles. The rows H and R preserve the same finite route, while P and L record provenance and local naming for that route.

Because no arbitrary open-cover, sequential compactness, choice, quotient-representative, trichotomy, host-equality, or ambient-completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3769.3 (Compact located interval finite-net extraction)**.** *Every accepted* CompactLocatedIntervalSubcove *carrier exports a finite subcover list only by the route*

$$\mathsf{LocatedInterval}^{\uparrow} \longrightarrow \mathsf{HeineBorelFiniteNet}^{\uparrow} \longrightarrow C \longrightarrow M \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, the finite list M *is extracted from the displayed mesh cells and their local coverage witnesses; it is not chosen from an arbitrary open cover and is not obtained from sequential compactness.*

Proof. Start with the located interval row I. It fixes the endpoint and radius data whose finite-net request is read by N.

The row N supplies only a finite mesh route. The local coverage row C attaches cover tokens to that finite mesh, so the extraction row M can list exactly those displayed tokens.

The terminal seal E is downstream of the finite extraction. It records the $\mathsf{Real}^{\uparrow}$ boundary of the already displayed route and does not supply a separate compactness theorem.

Thus the exported subcover is the finite list read from M, transported by H and replayed by R. The excluded open-cover and sequential compactness principles have no coordinate from which a consumer could read them. □

Theorem 3769.4 (Compact located interval L10 scope)**.** *The* $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ *packet is an L10 real/completion consumer of located interval and finite-net data. It may be used where a downstream real or completion argument needs a displayed finite subcover extracted from a located interval mesh. It does not close general Heine-Borel compactness, arbitrary open-cover compactness, sequential compactness without choice-like data, or any bridge to host compact spaces.*

Proof. The dependency route is already displayed in the carrier. It starts at $\mathsf{LocatedInterval}^{\uparrow}$, passes through $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, and reaches $\mathsf{Real}^{\uparrow}$ only after the local coverage and extraction rows are present.

Any L10 consumer can therefore inspect the finite mesh, the selected coverage tokens, and the real-name seal. No consumer receives an open-cover family, a sequence-to-subsequence selector, or a host compact-space object.

The scope claim follows by ledger exhaustion: P records the listed dependencies, L names the local packet, and all excluded compactness principles are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}$ packet is generated from a located interval row, a finite-net row, local mesh-cell coverage witnesses, an extracted finite token list, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3769.1, Definition 3769.2, Definition 3769.3, and Definition 3769.4 seed explicit finite-subcover extraction for located real intervals from finite-net data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLocatedIntervalSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Heine-Borel compactness, arbitrary open-cover compactness, sequential compactness without choice-like data, countable choice, selected quotient representatives, host interval equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net extraction, local coverage exactness, L10 nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactLocatedIntervalSubcoverUp.

Chapter 3770

A Concrete Naming Certificate for $\mathsf{RealAbsoluteValue}^{\uparrow}$

Following Bishop and Bridges on constructive absolute value for real names, the $\mathsf{RealAbsoluteValue}^{\uparrow}$ packet records absolute value as a finite NameCert read over the current L10 $\mathsf{Real}^{\uparrow}$ construction. It seals the dyadic endpoint of chapter 293 into regular real-name data without importing host real equality, trichotomy, or an ambient ordered-field object.

Definition 3770.1 (Real absolute-value carrier)**.** *A* $\mathsf{RealAbsoluteValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, \delta, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming the finite read windows used to inspect the real name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name row for the input real, including the rational approximants and regularity ledger.*
3. *δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to choose dyadic approximants at each displayed window.*
4. *D is the* $\mathsf{DyadicAbsoluteValue}^{\uparrow}$ *endpoint readback row, applying dyadic absolute value to the inspected dyadic cores.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability budget that bounds the replay of the endpoint readback across later windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that packages the resulting regular absolute-value name.*
7. *H records componentwise* hsame *transport for the schedule, regular-name data, dyadic tolerance, dyadic endpoint readback, modulus budget, and real seal.*
8. *C records the* Cont *replay routes from schedule inspection through dyadic readback to the sealed regular real name.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicAbsoluteValue}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$ *row.*

The accepted carrier contains no host real equality test, no trichotomy branch, no order-completion shortcut, no RealAlgOrder completion claim, and no absolute-value operation detached from the displayed regular-name and dyadic endpoint rows.

Lean4 ✓ `BEDC.Derived.RealAbsoluteValueUp.RealAbsoluteValueTasteGate single carrier alignment`

Theorem 3770.2 (Real absolute-value NameCert obligations)**.** *For an accepted packet* $A = (S, R, \delta, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$ *obligation surface consists exactly of carrier admission, dyadic absolute-value stability, regular-sequence seal compatibility, triangle nonescape, and ledger exactness for the displayed rows* $S, R, \delta, D, M, E, H, C, P, N$*.*

Proof. Carrier admission reads the schedule, regular-name data, dyadic tolerance, dyadic endpoint readback, stability budget, and real seal from the finite packet. The support rows H, C, P, N transport, replay, package, and name exactly those displayed rows.

Dyadic absolute-value stability is local to D and δ. The endpoint readback uses the dyadic absolute-value packet, while the tolerance row fixes the finite accuracy at which the dyadic core is inspected.

Regular-sequence seal compatibility is supplied by M and E. The modulus budget records enough stability to replay the endpoint readback across later windows, and the $\mathsf{Real}^{\uparrow}$ seal packages the resulting regular name without adding a host real comparison.

Ledger exactness follows by projection from the carrier. Every exported read is one of $S, R, \delta, D, M, E, H, C, P, N$ so no trichotomy branch, order-completion shortcut, or RealAlgOrder completion field can be recovered from the packet. □

Theorem 3770.3 (Real absolute-value triangle nonescape)**.** *Every triangle-facing consumer of an accepted* $\mathsf{RealAbsoluteValue}^{\uparrow}$ *packet reads only the regular-name seal* E *together with its schedule, dyadic tolerance, dyadic endpoint readback, modulus budget, componentwise transport, continuation replay, package provenance, and local naming rows. The packet therefore exposes the triangle surface as a stability obligation over regular real names and dyadic absolute-value readback, not as host real order, trichotomy, or a completed* $\mathsf{RealAlgOrder}^{\uparrow}$ *structure.*

Proof. Start with the carrier rows of Definition 3770.1. The only endpoint operation is the dyadic readback D, and the only real-facing result is the $\mathsf{Real}^{\uparrow}$ seal E.

The triangle-facing route must use the schedule and tolerance rows to compare finite read windows, then use M to replay the resulting regular-name stability. Transport by H and replay by C keep the route tied to those same displayed rows.

Package provenance P and local naming N identify the output as the $\mathsf{RealAbsoluteValue}^{\uparrow}$ surface. Since the carrier has no host real equality test, no trichotomy coordinate, and no RealAlgOrder completion field, a triangle consumer cannot escape the regular-name and dyadic-readback ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealAbsoluteValue}^{\uparrow}$ packet is generated from a StreamName schedule, RegSeqRat regular-name data, DyadicRatCore tolerance, DyadicAbsoluteValue endpoint readback, CauchyModulus stability budget, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$, seed)

Scope closed: Definition 3770.1, Definition 3770.2, and Definition 3770.3 bind the seed carrier, obligation surface, and triangle-facing nonescape boundary for constructive real absolute value.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealAbsoluteValue}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealAbsoluteValueUp.RealAbsoluteValueTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close RealAlgOrderUp completion, host real equality, trichotomy, or bridge checking.

Upgrade path: Bridge checking requires a paper bridge from the checked carrier alignment to dyadic absolute-value stability, regular-sequence seal compatibility, triangle nonescape, and ledger exactness.

Chapter 3771

A Concrete Naming Certificate for CUNSpectralAntenna$^{\uparrow}$

The CUNSpectralAntenna$^{\uparrow}$ packet realizes the CUN tail spectral antenna theorem of Definition 9462.1457 as finite BEDC NameCert data. It reads the retention inequality, Perron response, positive self-energy row, and small-mass row from the six-boundary-tile spectral window, then exposes only the finite antenna rows needed by downstream trigger-spectrum consumers. The packet is not a biological-code oracle, stochastic evolution law, hidden Perron solver, quotient codon semantics, or unbounded spectral search.

The antenna packet consumes the spectral-closure and trigger-algebra sibling surfaces through the same separation row used below: chapter 3753 supplies the greedy spectral closure boundary, chapter 2270 supplies the stabilization read, and chapter 3478 supplies the finite trigger-algebra coverage surface. These sibling routes are the concrete sources for the S row and for the Pkg provenance row Q.

Definition 3771.1 (CUN spectral antenna carrier)**.** *A* CUNSpectralAntenna$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, R, P, E, M, S, H, C, Q, N)$$

with the following displayed rows:

1. *T is the CUN tail row naming the finite tail cells and their attachment to the same six-boundary window as Definition 9462.1457.*
2. *R is the retention row displaying $\rho(M) = 19/30$ and its comparison with the WNR baseline.*
3. *P is the Perron-response row displaying $\lambda(M) = 0.675247931\ldots$ and the comparison with the same WNR baseline.*
4. *E is the self-energy row recording that the tail contributes a positive perturbation to the radial Perron mode.*
5. *M is the small-mass row recording the quasi-stationary tail mass boundary used by the source theorem.*
6. *S is the sibling separation row over* GreedySpectralClosure$^{\uparrow}$, SpectralStabilization$^{\uparrow}$, RadialWobbleSpectrum$^{\uparrow}$, BooleanTriggerLattice$^{\uparrow}$, *and* WindowCoverageTriggerAlgebra$^{\uparrow}$.
7. *H records componentwise* hsame *transport for tail, retention, Perron-response, self-energy, small-mass, sibling-separation, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an antenna read through T, R, P, E, M, S, H.*
9. *Q is the* Pkg *provenance over* $\mathsf{CUNSpectralAntenna}^{\uparrow}$, $\mathsf{GreedySpectralClosure}^{\uparrow}$, $\mathsf{SpectralStabilization}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, $\mathsf{BooleanTriggerLattice}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem Definition 9462.1457,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, P, E, M, S, H, C, Q, N$. It has no coordinate for hidden spectral computation, biological-code truth, stochastic dynamics, quotient codon equality, ambient eigenvector choice, or unbounded enumeration.

Theorem 3771.2 (CUN spectral antenna NameCert obligations)**.** *For every accepted* $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *carrier $A = (T, R, P, E, M, S, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$ *surface consists exactly of CUN tail admission, retention-row exposure, Perron-response exposure, self-energy exposure, small-mass exposure, sibling-separation exposure, componentwise transport, displayed replay, provenance, and local naming.*

The Lean follow-up `lean4/BEDC/Derived/CUNSpectralAntennaUp/TasteGate.lean` *must provide* Nontrivial $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *and* FieldFaithful $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *instances so that T, R, P, E, M, S, H, C remain separate displayed fields.*

Lean4 ✓ `BEDC.Derived.CUNSpectralAntennaUp.CUNSpectralAntennaTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3771.1. The rows T, R, P, E, M, S expose the finite tail, retention comparison, Perron response, self-energy mechanism, small-mass boundary, and sibling separation before any consumer can read an antenna conclusion.

The spectral rows are displayed table rows from the source theorem. They record the stated numerical comparisons and the self-energy interpretation, but they do not introduce a hidden Perron solver or a stochastic model.

The structural rows H, C, Q, N preserve that route. Since no host biological truth predicate, quotient codon equality, eigenvector choice, hidden spectral computation, or unbounded enumeration is a carrier coordinate, the local NameCert obligations are exactly the listed rows. □

Theorem 3771.3 (CUN spectral antenna downstream separation)**.** *Every downstream antenna read of an accepted* $\mathsf{CUNSpectralAntenna}^{\uparrow}$ *packet factors through*

$$T \longmapsto R, P, E, M \longmapsto S, H, C, Q, N.$$

The read may cite the finite CUN tail, retention row, Perron-response row, self-energy row, small-mass row, and sibling separation row. It cannot export a biological-code oracle, stochastic evolutionary law, hidden Perron solver, quotient codon equality, or unbounded spectral optimizer.

Proof. Start with the tail row T. It fixes the finite CUN tail cells read by the source theorem, so later consumers cannot range over an unbounded codon space.

Read the four spectral rows next. The retention row R, Perron-response row P, self-energy row E, and small-mass row M are displayed finite data tied to the same tail.

The sibling row S separates the packet from trigger closure, stabilization, radial wobble, Boolean trigger, and coverage-trigger algebra packets. The remaining rows H, C, Q, N transport, replay, source, and name the same antenna route.

The absent host objects listed in the statement are not coordinates of the carrier and cannot be exported by a consumer read. □

Falsifiable prediction: An accepted $\mathsf{CUNSpectralAntenna}^{\uparrow}$ antenna read is refuted if removing the CUN tail self-energy row preserves the same retention, Perron-response, small-mass, sibling-separation, transport, replay, provenance, and local naming claim.

Independence witness: greedy-spectral-closure, spectral-stabilization, radial-wobble-spectrum, boolean-trigger-lattice, window-coverage-trigger-algebra: none is bijective with $\mathsf{CUNSpectralAntenna}^{\uparrow}$ because the packet jointly carries CUN tail, retention, Perron response, self-energy, small-mass, sibling-separation, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}$.

Constructive story: A $\mathsf{CUNSpectralAntenna}^{\uparrow}$ packet is generated from a finite CUN tail row, retention row, Perron-response row, self-energy row, small-mass row, sibling-separation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3771.1, Definition 3771.2, and Definition 3771.3 seed the finite CUN tail spectral antenna packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CUNSpectralAntenna}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological-code truth, stochastic evolutionary dynamics, quotient codon equality, hidden spectral computation, ambient eigenvector choice, unbounded optimization, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retention exactness, Perron-response exactness, self-energy preservation, small-mass exposure, Nontrivial CUNSpectralAntennaUp, and FieldFaithful CUNSpectralAntennaUp for BEDC.Derived.CUNSpectralAntennaUp.

Chapter 3772

A Concrete Naming Certificate for $\mathsf{FrechetCauchyCriterion}^{\uparrow}$

Following Frechet and Bishop–Bridges constructive analysis, the $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ packet records the finite criterion surface identifying convergence of a displayed real sequence with the Cauchy tail condition. It sits after the Real seal of chapter 38, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Real Cauchy criterion surface of chapter 1590, and the sequence-limit interface of chapter 1413. The packet exposes a choice-free equivalence route between a displayed limit row and a displayed Cauchy-tail row; it does not import quotient streams, countable choice, host real equality, or an ambient completeness theorem.

Definition 3772.1 (Frechet Cauchy criterion carrier)**.** *A* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, S, Q, D, K, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row naming the real surface on which the sequence and candidate limit are displayed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for source entries, candidate-limit reads, and late-tail comparisons.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the requested radii and the dyadic budgets used for both convergence and tail comparisons.*
5. *K is the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *row naming the accepted Cauchy-tail certificate for the displayed sequence.*
6. *L is the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row naming the accepted sequence-to-limit certificate over the same real source, windows, readbacks, and tolerances.*
7. *H records componentwise* hsame *transport for R, S, Q, D, K, L and the structural rows.*
8. *C records displayed* Cont *replay from a tolerance request through sequence windows, regular readback, dyadic budgets, Cauchy-tail admission, and sequence-limit admission.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, Q, D, K, L, H, C, P, N$. It has no coordinate for quotient stream equality, a selected subsequence, countable choice, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FrechetCauchyCriterionUp.FrechetCauchyCriterionTasteGate single carrier alignment`

Theorem 3772.2 (Frechet Cauchy criterion NameCert obligations)**.** *For an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier $F = (R, S, Q, D, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Real source, finite windows, regular rational readback, dyadic tolerance ledger, Cauchy-tail row, sequence-limit row, transport, replay, provenance, and local naming;*
2. *tail-to-limit compatibility, requiring the* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *row K and the* $\mathsf{RealSequenceLimit}^{\uparrow}$ *row L to use the same R, S, Q, D route before either certificate is consumed;*
3. *eventual-tail exactness, requiring each tolerance request to be represented by a displayed late window and dyadic budget in S, Q, D;*
4. *transport and replay stability, requiring H and C to preserve the same route under componentwise* hsame *comparison;*
5. *ledger non-escape, refusing quotient streams, selected representatives, countable-choice selectors, host real equality, and ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 3772.1. The tuple exposes exactly the Real source, finite windows, readbacks, dyadic tolerances, Cauchy-tail certificate, sequence-limit certificate, and structural rows.

The compatibility row reads K and L only after R, S, Q, D have been displayed. Thus the Cauchy-tail side and the convergence side compare the same sequence observations and the same dyadic tolerance ledger.

Eventual-tail exactness is finite. Each accepted tolerance request is represented by a late window in S, regular rational data in Q, and a dyadic budget in D, so no hidden selector over tail positions is introduced.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the excluded objects are not carrier coordinates, the local NameCert surface is exhausted by the displayed rows. □

Theorem 3772.3 (Frechet Cauchy criterion eventual-tail exactness)**.** *Let $F = (R, S, Q, D, K, L, H, C, P, N)$ be an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier. A convergence-facing read through L and a Cauchy-tail-facing read through K consume the same finite tail route*

$$R, S, Q, D \longmapsto K, L \longmapsto H, C, P, N.$$

For every displayed tolerance request, the route supplies a late finite window, a regular rational readback, and a dyadic comparison budget before either the limit certificate or the Cauchy-tail certificate can be exported.

Proof. Start with the carrier projection. The source and observation rows R, S, Q, D precede both certificate rows K and L.

Read the Cauchy-tail certificate K through chapter 1590. Its accepted route already requires finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance data before the Cauchy criterion is exposed.

Read the sequence-limit certificate L through chapter 1413. Its accepted route uses the same kind of finite tail-window schedule and dyadic ledger before any Real seal is consumed.

The Frechet packet identifies those routes at the displayed rows R, S, Q, D and then preserves them with H, C, P, N. Hence the eventual-tail data are exact inside the packet and cannot be replaced by a detached sequence observation. □

Theorem 3772.4 (Frechet Cauchy criterion Real seal nonescape)**.** *Every Real-facing consumer of an accepted* $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ *carrier* $F = (R, S, Q, D, K, L, H, C, P, N)$ *is confined to the displayed route from the Real source through the finite windows, regular readback, dyadic tolerance ledger, Cauchy-tail row, sequence-limit row, and structural rows. The consumer cannot escape into quotient streams, a selected limit representative, countable choice, host real equality, or an ambient Real completeness principle.*

Proof. Project the carrier. The only Real-facing rows are the source row R and the sequence-limit row L, and both are attached to the finite observation and tolerance rows S, Q, D.

Use Definition 3772.2. The ledger non-escape obligation refuses quotient streams, selected representatives, choice, host equality, and ambient completeness as exported coordinates.

Use Definition 3772.3 to fix the terminal route. Since K and L are consumed only through the same displayed tail data and the structural rows preserve that route, no Real-facing consumer can introduce an additional seal or completion principle outside $R, S, Q, D, K, L, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetCauchyCriterion}^{\uparrow}$ packet is generated from a RealUp source row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealCauchyCriterion row, a RealSequenceLimit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3772.1, Definition 3772.2, Definition 3772.3, and Definition 3772.4 seed the finite equivalence surface between convergence and Cauchy-tail data for real sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetCauchyCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FrechetCauchyCriterionUp.FrechetCauchyCriterionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, countable choice, host real equality, ambient Real completeness, or bridge checking.

Upgrade path: Bridge checking requires a paper bridge from the checked carrier alignment to tail-to-limit compatibility, eventual-tail exactness, transport stability, replay stability, provenance exhaustion, and Real seal nonescape.

Chapter 3773

A Concrete Naming Certificate for $\mathsf{LocatedCutReal}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedCutReal}^{\uparrow}$ packet records the constructive comparison surface where located lower and upper cut data are read as finite real-name evidence before any endpoint comparison is exported. It sits between the regular Cauchy real-name route of $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ in chapter 38, chapter 301, chapter 126, and chapter 291, and the cut-facing siblings $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, and $\mathsf{DyadicLocatedCut}^{\uparrow}$ in **??**, chapter 436, and chapter 2089. The packet exposes located comparison as a finite NameCert route; it does not claim Dedekind completeness, quotient-real equality, trichotomy, countable choice, selected suprema, or an ambient host real order.

Definition 3773.1 (LocatedCutReal carrier)**.** *A* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, G, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the located lower-and-upper cut ledger. It records finite lower cells, upper cells, gap requests, and comparison witnesses in the style of the located cut surface.*
2. *G is the dyadic gap row over* $\mathsf{DyadicRatCore}^{\uparrow}$*. It supplies the displayed dyadic endpoint and radius cells by which a finite gap request is read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the finite observation windows opened by the comparison request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It reads the endpoint-normalized windows as regular rational data before any real seal is consumed.*
5. *Q is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row. It records the finite locatedness evidence used by order, metric, and comparison consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It can be read only after D, G, W, R, Q have been displayed.*
7. *H records componentwise* hsame *transport for cut cells, dyadic endpoints, schedules, regular-rational handoff, located comparison, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for every consumer read through the same ordered route $D, G, W, R, Q, E, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCutReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicLocatedCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, G, W, R, Q, E, H, C, P, N$. It has no coordinate for quotient reals, selected cuts, Dedekind completeness, trichotomy, countable choice, or host analytic order.

Lean4 ✓ `BEDC.Derived.LocatedCutRealUp.LocatedCutRealTasteGate single carrier alignment`

Theorem 3773.2 (LocatedCutReal NameCert obligations)**.** *For an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier*

$$L = (D, G, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the located cut ledger, dyadic gap row, finite stream schedule, regular-rational handoff, located real comparison row, terminal real seal, componentwise transport, displayed replay, provenance, and local naming.*

The obligation surface may cite the $\mathsf{DedekindCut}^{\uparrow}$ *lower-and-upper cut discipline, the* $\mathsf{DyadicLocatedCut}^{\uparrow}$ *endpoint route, the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, and the* $\mathsf{Real}^{\uparrow}$ *seal route. It cannot add a selected supremum, quotient-real representative, ambient completeness proof, trichotomy branch, or external comparison oracle.*

Proof. Project the carrier of Definition 3773.1. The rows D, G, W, R, Q, E expose the located cut cells, dyadic endpoint budgets, finite schedules, regular-rational readback, located real comparison, and terminal real seal in order.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier stores no quotient real, selected cut, completeness witness, trichotomy branch, choice selector, or external order oracle, the local NameCert surface is exactly the displayed finite route. □

Theorem 3773.3 (LocatedCutReal Cauchy comparison)**.** *Every comparison consumer of an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *packet factors through the finite Cauchy-facing route*

$$D, G, W, R, Q, E, H, C, P, N.$$

The read may use $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff data, the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It may not bypass those rows by invoking a host Dedekind-completeness theorem, quotient-real equality, selected supremum, trichotomy, countable choice, or an ambient real-order decision procedure.*

Proof. Begin with the located cut ledger D and dyadic gap row G. They provide only finite lower, upper, endpoint, radius, and gap cells, so the comparison request starts as a displayed packet route rather than as an ambient real-order statement.

Next read the schedule W and handoff row R. These rows force endpoint data to pass through finite stream windows and regular-rational budgets before any real-facing comparison is available.

The located comparison row Q and terminal seal E are downstream of that Cauchy-facing route. The rows H, C, P, N preserve the same carrier under transport and replay, so no consumer can export the excluded host objects while still consuming the accepted $\mathsf{LocatedCutReal}^{\uparrow}$ packet. □

Theorem 3773.4 (LocatedCutReal nonescape)**.** *The terminal* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{LocatedCutReal}^{\uparrow}$ *carrier exports only the displayed located cut, dyadic endpoint, stream schedule, regular-rational handoff, and located comparison rows. In particular, it exports no Dedekind completeness principle, quotient real, selected cut representative, trichotomy branch, countable-choice selector, or host analytic order.*

Proof. Read the seal row E through Definition 3773.3. The seal is available only after D, G, W, R, Q have been displayed, and every transported or replayed use is still sourced and named by H, C, P, N.

The absent objects listed in the statement are not fields of Definition 3773.1. Therefore a terminal seal read can only expose the displayed finite comparison route and cannot be reinterpreted as an external completeness, quotient, choice, trichotomy, or host-order witness. □

Falsifiable prediction: An accepted $\mathsf{LocatedCutReal}^{\uparrow}$ comparison read is refuted if the located cut ledger, dyadic endpoint row, StreamName schedule, RegSeqRat handoff, LocatedReal comparison row, Real seal, transport, replay, provenance, or local naming can be removed while preserving the same packet claim.

Independence witness: real, regseqrat, streamname, dyadicratcore, locatedreal, dedekindcut, dyadic-located-cut: $\mathsf{LocatedCutReal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located cut cells, dyadic endpoint budgets, finite schedules, regular-rational handoff, located comparison, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCutReal}^{\uparrow}$ packet is generated from located lower-and-upper cut ledgers, dyadic endpoint and gap budgets, StreamName finite windows, RegSeqRat handoff rows, LocatedReal comparison evidence, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3773.1, Definition 3773.2, Definition 3773.3, and Definition 3773.4 seed the finite constructive located-cut comparison surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutReal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedCutRealUp.LocatedCutRealTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient reals, selected suprema, trichotomy, countable choice, ambient host real order, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located cut exactness, dyadic endpoint transport, regular-Cauchy handoff, located comparison preservation, and Real seal nonescape for BEDC.Derived.LocatedCutRealUp.

Chapter 3774

A Concrete Naming Certificate for DyadicStepApproximation$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicStepApproximation$^{\uparrow}$ packet records the finite step approximation of a located real interval by dyadic cells, displayed sample values, and an oscillation ledger. It sits after the dyadic rational carrier of chapter 291, the located interval source of chapter 1709, the tagged finite-sum boundary of chapter 1615, the upper/lower sum surface of chapter 1880, and the regulated integration consumer of chapter 1557. The packet gives L10 Real and Completion routes one reusable finite approximation carrier before any quotient of functions, selected mesh limit, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ completeness principle is available.

Definition 3774.1 (Dyadic step approximation carrier)**.** *A* DyadicStepApproximation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, M, L, V, O, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval source row, read through the endpoint-plus-radius surface of chapter 1709.*
2. *M is the dyadic mesh row from* DyadicRatCore$^{\uparrow}$*. It records the common exponent scale, ordered dyadic breakpoints, adjacent cell widths, endpoint containment, and common-scale distance windows for the interval cells.*
3. *L is the finite cell ledger. It lists the carried cells of M, their inclusion in I, the boundary orientation rows, and the adjacency data consumed by Riemann and Darboux readers.*
4. *V is the step value row. For each carried cell it displays one value row and the finite readback route by which that value is available to* RiemannSum$^{\uparrow}$ *and* DarbouxSum$^{\uparrow}$ *consumers.*
5. *O is the oscillation bound row. It records the displayed upper-minus-lower tolerance on each carried cell, using dyadic endpoint and radius comparisons from M.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after the located source, dyadic mesh, cell ledger, step values, and oscillation bounds have been displayed.*
7. *H records componentwise* hsame *transport for I, M, L, V, O, E and for the replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from an interval request through the dyadic mesh, finite cells, step values, oscillation bounds, terminal seal, transports, provenance, and local name.*
9. *P is the* Pkg *provenance over* DyadicStepApproximation$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RegulatedIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite step-approximation surface. It contains no quotient function equality, selected tag family outside L, limiting mesh process, open-cover compactness witness, countable-choice schedule, ambient Real$^{\uparrow}$ *completeness theorem, Lean verification, or bridge credential.*

Lean4 ✓ `BEDC.Derived.DyadicStepApproximationUp.DyadicStepApproximationTasteGate single carrier alignment`

Theorem 3774.2 (Dyadic step approximation obligation surface)**.** *For every accepted* DyadicStepApproximation$^{\uparrow}$ *carrier $A = (I, M, L, V, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, dyadic mesh, finite cell ledger, step value rows, oscillation bound rows, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *mesh exactness, requiring every carried breakpoint, width, and endpoint comparison to pass through the common-scale* DyadicRatCore$^{\uparrow}$ *row M;*
3. *interval coverage, requiring every cell in L to lie in the located interval source I through displayed endpoint and radius rows;*
4. *step-value readback, requiring every value row in V to be read through the same finite cells before a* RiemannSum$^{\uparrow}$ *or* DarbouxSum$^{\uparrow}$ *consumer can use it;*
5. *oscillation control, requiring O to compare the upper and lower cell values through the displayed dyadic distance-window ledger;*
6. *real-seal handoff, requiring the terminal* Real$^{\uparrow}$ *row E to be exposed only after I, M, L, V, O are displayed;*
7. *transport and replay stability, requiring H and C to preserve the same finite rows componentwise;*
8. *ledger non-escape, excluding quotient function equality, selected mesh limits, arbitrary tag choice, open-cover compactness, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No step-approximation obligation is stored outside these rows.

Proof. Project the carrier of Definition 3774.1. The analytic rows are precisely the located interval source I, dyadic mesh row M, finite cell ledger L, step value row V, oscillation bound row O, and terminal real seal E.

Mesh exactness is read from M. The dyadic source chapter 291 supplies the common exponent scale, breakpoint comparisons, distance-window rows, and radius ledgers used by every cell of the packet.

Interval coverage is read from I and L. The located interval chapter supplies the endpoint-plus-radius boundary, while the ledger lists exactly the cells whose endpoints are certified to lie inside that boundary.

Step-value readback and oscillation control are then finite. The row V gives one displayed value route for each carried cell, and O compares the upper and lower values on that same cell through the dyadic distance-window ledger. These are the rows consumed by the Riemann and Darboux sibling chapters.

Finally, the Real-facing route exposes E only after I, M, L, V, O are visible. The transport row H, continuation row C, provenance row P, and local name N repack the same finite surface. Since the carrier has no quotient-function coordinate, limiting mesh coordinate, arbitrary tag selector, compactness row, choice row, ambient completeness theorem, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Theorem 3774.3 (Dyadic step approximation integration handoff)**.** *Every* $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, *or* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicStepApproximation}^{\uparrow}$ *carrier receives the finite route*

$$I, M, L, V, O \longmapsto E \longmapsto H, C, P, N.$$

The Riemann consumer reads the displayed cells and values as a tagged finite-sum boundary, the Darboux consumer reads the same cells and oscillation rows as upper/lower finite sums, and the regulated-integral consumer reads the packet as a step approximation ledger before any terminal $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start from the obligation surface of Definition 3774.2. It confines every accepted packet to the located source, dyadic mesh, finite cell ledger, value rows, oscillation rows, terminal seal, and structural rows.

For a Riemann consumer, apply the sibling surface of chapter 1615. The cell ledger L supplies the carried mesh cells and the row V supplies the finite value reads, so the consumer obtains a tagged finite-sum boundary rather than a hidden tag family.

For a Darboux consumer, apply chapter 1880. The same cells and value rows are read through O as upper/lower oscillation data, so the upper-minus-lower comparison remains a displayed finite ledger.

For a regulated-integral consumer, apply chapter 1557. The packet is consumed as a step approximation ledger before the $\mathsf{Real}^{\uparrow}$ seal E is public. Thus all three consumers receive the same finite route, and the excluded quotient, limit, compactness, choice, and ambient-completeness objects have no row through which to enter. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicStepApproximation}^{\uparrow}$ packet is generated from a located interval source, a $\mathsf{DyadicRatCore}^{\uparrow}$ mesh row, a finite cell ledger, step value rows, oscillation bound rows, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3774.1, Definition 3774.2, and Definition 3774.3 seed the finite step-approximation carrier consumed by RiemannSum, DarbouxSum, and RegulatedIntegral routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicStepApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient function equality, selected mesh limits, arbitrary tag choice, open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh exactness, interval coverage, step-value readback, oscillation control, real-seal handoff, transport stability, replay stability, provenance exhaustion, and integration handoff for BEDC.Derived.DyadicStepApproximationUp.

Chapter 3775

A Concrete Naming Certificate for $\mathsf{HausdorffUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley uniform spaces, the $\mathsf{HausdorffUniformSpace}^{\uparrow}$ packet names the separated uniform interface that lies between the entourage carrier of chapter 339 and the separated boundary already exposed by chapter 1876. It also gives the Hausdorff-reflection route of **??** a finite NameCert surface before any completion-facing consumer reads quotient points, host equality, or selected representatives.

Definition 3775.1 (Hausdorff uniform space carrier)**.** *A* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (U, S, R, A, T, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, including the finite diagonal, inverse, refinement, and composition data exposed by the source uniformity.*
2. *S is the* $\mathsf{SeparatedUniformSpace}^{\uparrow}$ *classifier row, recording when two displayed points are separated by the available entourage observations.*
3. *R is the* $\mathsf{HausdorffReflection}^{\uparrow}$ *route row, naming the quotient-free reflection boundary as a consumer of U and S.*
4. *A is the antisymmetry ledger row, requiring indistinguishable point-pair reads to be identified only through the displayed separated classifier.*
5. *T records componentwise* hsame *transport for U, S, R, A, C, P, N.*
6. *C records displayed* Cont *replay from uniform, separated, and Hausdorff-reflection consumers through the same finite route.*
7. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one separated uniform-space packet. It contains no coordinate for quotient points, host equality of points, a completion object, selected representatives, countable choice, or ambient set-theoretic separation.

Lean4 ✓ `BEDC.Derived.HausdorffUniformSpaceUp.TasteGate.HausdorffUniformSpaceTasteGate single carrier alignment`

Theorem 3775.2 (Hausdorff uniform space NameCert obligations)**.** *For an accepted* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *carrier* $H = (U, S, R, A, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed uniform entourage row, separated classifier row, Hausdorff-reflection route row, antisymmetry ledger row, transport row, replay row, provenance row, and local naming row;*
2. *uniform-source exactness, requiring every separated or Hausdorff-reflection read to begin with the finite entourage data in* U*;*
3. *separated-classifier stability, requiring* S *and* A *to compare point-pair reads only through displayed uniform observations;*
4. *reflection non-escape, requiring* R *to expose the Hausdorff boundary without adding quotient points, host equality, selected representatives, or completion coordinates;*
5. *transport, replay, provenance, and local-naming stability for the same displayed row order.*

No Hausdorff uniform-space obligation is stored outside U, S, R, A, T, C, P, N*.*

Proof. Carrier admission is projection from Definition 3775.1. The projected coordinates are exactly the uniform row U, separated classifier row S, Hausdorff-reflection route R, antisymmetry ledger A, and structural rows T, C, P, N.

Uniform-source exactness follows by reading every separated and reflection-facing consumer through the same finite entourage surface used by chapter 339. The separated row S and antisymmetry ledger A add only classifier evidence on top of those entourage observations.

Reflection non-escape is the role of R. It may cite the separated packet of chapter 1876 and the reflection boundary of **??**, but it receives no quotient point, host equality, selected representative, completion object, or ambient separation principle.

The transport and replay rows T and C preserve that same route. Since P and N package and name only the displayed carrier rows, every excluded object remains outside the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffUniformSpace}^{\uparrow}$ consumer reads a $\mathsf{UniformSpace}^{\uparrow}$ entourage row, a $\mathsf{SeparatedUniformSpace}^{\uparrow}$ classifier row, and a $\mathsf{HausdorffReflection}^{\uparrow}$ route row before citing Hausdorff uniform separation.

Independence witness: uniformspace, separateduniformspace, hausdorffreflection: $\mathsf{HausdorffUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the entourage source, separated classifier, Hausdorff-reflection route, antisymmetry ledger, transport, replay, provenance, and local NameCert rows as one finite interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3775.1 and Definition 3775.2 seed the separated uniform-space packet over displayed entourage, separated classifier, Hausdorff-reflection route, antisymmetry, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffUniformSpaceUp.TasteGate.HausdorffUniformSpaceTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient point construction, host equality, selected representatives, completion existence, countable choice, or ambient set-theoretic separation.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exactness, separated-classifier stability, reflection non-escape, transport and replay stability, and ledger non-escape for BEDC.Derived.HausdorffUniformSpaceUp.

Constructive story: A HausdorffUniformSpace$^{\uparrow}$ packet is generated from UniformSpace entourage data, SeparatedUniformSpace classifier data, a HausdorffReflection route, an antisymmetry ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3776

A Concrete Naming Certificate for $\mathsf{SecantMethod}^{\uparrow}$

Following Traub and Ortega–Rheinboldt on iterative root-finding, the $\mathsf{SecantMethod}^{\uparrow}$ packet records the finite two-point root-finding surface used by the L10 completion-facing numerical-analysis route. It sits beside the Newton–Kantorovich packet of chapter 1785, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, and the terminal real-name seal of chapter 38. The chapter names only the displayed bracketing windows, two previous iterates, finite secant denominator checks, residual reads, next-iterate rows, Cauchy handoff, transport, replay, provenance, and local naming; it does not assert global convergence, a derivative bound, quotient equality of reals, a selected host limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3776.1 (Secant method carrier)**.** *A* $\mathsf{SecantMethod}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, X, Y, D, R, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-evaluation row, naming the finite* $\mathsf{Real}^{\uparrow}$ *input and residual observations used by the root-finding request.*
2. *X and Y are the two previous iterate rows, each exposed through* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the same* $\mathsf{StreamName}^{\uparrow}$ *window schedule.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *denominator-separation row, recording the finite nonzero secant-denominator comparison needed before the next iterate can be read.*
4. *R is the residual-control row, binding the two displayed residuals and the candidate next residual to finite dyadic tolerances.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the iterate, residual, denominator, and tolerance observations.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that reads every iterate and residual window as regular rational data.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, X, Y, D, R, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for function-evaluation, iterate, denominator, residual, window, readback, endpoint, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a secant request through $F, X, Y, D, R, W, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SecantMethod}^{\uparrow}$, $\mathsf{NewtonKantorovich}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$ *row.*

The classifier accepts the packet only through these finite rows. There is no coordinate for derivative availability, a completed limit point, quotient equality of real presentations, an unbounded convergence search, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SecantMethodUp.TasteGate.SecantMethodTasteGate single carrier alignment`

Theorem 3776.2 (Secant method NameCert obligations)**.** *For the carrier of Definition 3776.1, the local* $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$ *surface consists exactly of carrier admission for $S = (F, X, Y, D, R, W, Q, E, H, C, P,$ function-evaluation admission through F, two-iterate admission through X, Y, dyadic denominator-separation admission through D, residual-control admission through R, finite-window admission through W, regular-rational readback through Q, downstream* $\mathsf{Real}^{\uparrow}$ *sealing through E, component-wise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The function row F and iterate rows X, Y expose the only data from which the secant step may be read, and both iterate rows are tied to the same finite window surface.

The denominator and residual checks are displayed before the next-iterate read. The row D supplies a dyadic nonzero comparison for the secant denominator, while R records residual tolerances using the same finite observations.

Readback then passes through W and Q. The $\mathsf{StreamName}^{\uparrow}$ row supplies the requested windows, and $\mathsf{RegSeqRat}^{\uparrow}$ turns those windows into regular rational data before the downstream $\mathsf{Real}^{\uparrow}$ seal E is visible.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier contains no derivative row, limit selector, quotient equality, unbounded search, choice principle, or ambient completeness coordinate, the listed rows exhaust the local obligations. □

Theorem 3776.3 (Secant method Cauchy handoff)**.** *Every completion-facing read of an accepted* $\mathsf{SecantMethod}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{SecantMethod}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff first exposes finite iterate and residual windows, then regular rational readback, then dyadic denominator and residual budgets, and only then the terminal real-name seal. It supplies no independent convergence theorem, no derivative-based Newton certificate, no selected limit, and no ambient completeness principle.

Proof. Start with a completion-facing consumer. The carrier forces the request to read W and Q before a secant step can be interpreted, so the iterates and residuals are finite regular rational observations rather than host real representatives.

Next inspect the dyadic rows. The denominator-separation row D and residual-control row R provide the only permitted comparison data for the next iterate, and both are attached to the same displayed windows.

The $\mathsf{Real}^{\uparrow}$ seal E is reached after those rows have been replayed through C and transported through H. Thus the exported handoff is a finite Cauchy-facing route, not an assertion of global

convergence, derivative control, selected limits, quotient equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{SecantMethod}^{\uparrow}$ completion-facing read factors through finite StreamName windows, RegSeqRat readback, DyadicRatCore denominator and residual budgets, displayed secant-step rows, a downstream RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: newtonkantorovich, regseqrat, dyadicratcore, streamname, real: $\mathsf{SecantMethod}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two previous iterate rows, a finite secant denominator check, residual-control rows, regular rational readback, dyadic tolerance data, a downstream Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}$.

Constructive story: A $\mathsf{SecantMethod}^{\uparrow}$ packet is generated from function-evaluation rows, two previous iterate rows, dyadic denominator-separation and residual-control rows, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3776.1, Definition 3776.2, and Definition 3776.3 seed the finite secant root-finding route from regular rational windows to the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SecantMethod}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global convergence, derivative-control hypotheses, quotient equality of real presentations, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, two-iterate admission, denominator-separation exactness, residual-control routing, Cauchy handoff, transport, replay, provenance, and local naming for BEDC.Derived.SecantMethodUp.

Chapter 3777

A Concrete Naming Certificate for $\mathsf{CauchyCompletionBind}^{\uparrow}$

Following Bishop and Bridges and Mac Lane, the $\mathsf{CauchyCompletionBind}^{\uparrow}$ packet records the finite Kleisli-composition surface for Bishop-style completion: a completion-facing read may bind a source completion row to a displayed completion-valued route only through the already named $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ certificates. It is a local ledger for bind composition, not a quotient completion, not a selected representative, not countable choice, and not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

The bind packet consumes the completion monad of chapter 378, the unit admission of chapter 1527, the Kleisli consumer route of chapter 1845, the Bishop-regular completion boundary of chapter 1639, and the terminal Real seal of chapter 38 as concrete sibling rows rather than as background terminology.

Definition 3777.1 (CauchyCompletionBind carrier)**.** *A* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, U, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-completion row over* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, exposing the regular Cauchy input windows before any bind consumer is read.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row recording the unit admission for source points and terminal seal handoff.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row recording the single-bind consumer route and the reassociation schedule inherited from* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*.*
4. *E is the completion seal row that records which displayed* $\mathsf{Real}^{\uparrow}$ *or Bishop-regular completion boundary the bind route may consume.*
5. *H is the componentwise* hsame *transport row for the source, unit, Kleisli, seal, replay, provenance, and local naming data.*
6. *C is the displayed* Cont *replay row for the finite bind route.*
7. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionBind}^{\uparrow}$*,* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$*,* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$ *row naming the bind packet.*

The carrier has no coordinate for quotient Cauchy classes, selected representatives, countable-choice schedules, host equality of completed objects, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionBindUp.CauchyCompletionBindTasteGate single carrier alignment`

Theorem 3777.2 (CauchyCompletionBind NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *carrier* $B = (S, U, K, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$ *surface consists exactly of source-completion exposure, unit admission, Kleisli bind exposure, completion-seal compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every bind consumer must first read the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$*,* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows recorded in* P*.*

Proof. Project the accepted carrier. The rows S, U, K, E are the only analytic rows available to a bind consumer, and each is already sourced from the named completion, unit, Kleisli, Bishop-regular, and real-seal siblings.

Transport and replay are structural. The row H preserves the same source, unit, Kleisli, and seal coordinates under displayed hsame routes, while C replays the same finite bind path through the consumer surface.

Finally P and N source and name the local certificate. Since the carrier contains no quotient class, representative selector, choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 3777.3 (CauchyCompletionBind associativity ledger)**.** *Let* $B = (S, U, K, E, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *packet. Any two public reads of a two-step completion bind that use the same displayed source-completion row* S*, unit row* U*, Kleisli row* K*, completion seal* E*, transport row* H*, replay row* C*, provenance row* P*, and local name* N *have the same finite associativity ledger. The ledger is a displayed route equality of BEDC rows; it is not quotient completion equality, not equality of selected representatives, not countable choice, and not a proof that* $\mathsf{Real}^{\uparrow}$ *is complete in the ambient host.*

Proof. Read the two-step bind only through the carrier rows. The first route and the reassociated route both project the same S, U, K, E coordinates, so their source completion, unit admission, Kleisli schedule, and seal handoff are the displayed data of one packet.

Use H to transport the coordinates componentwise and C to replay the same finite consumer path. These structural rows compare the bind schedules inside the packet rather than by host equality of completed objects.

The provenance and naming rows attach the comparison to the same $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sources. No excluded quotient, representative, choice, or ambient-completeness object is available to alter the ledger. □

Theorem 3777.4 (CauchyCompletionBind real-completion forward link)**.** *Every accepted* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *consumer that exposes a completion seal must first route through chapter 378, chapter 1527, chapter 1845, chapter 1639, and chapter 38. The exposed seal is therefore read only after the source-completion row* S*, unit row* U*, Kleisli row* K*, completion seal row* E*, componentwise transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *have been displayed. It exports no quotient completion class, selected representative, countable-choice schedule, host equality of completed objects, or ambient completeness theorem.*

Proof. Project the accepted carrier $B = (S, U, K, E, H, C, P, N)$. The source row S and seal row E are completion-facing only because the provenance row P names the Cauchy monad, Bishop-regular completion, and Real seal siblings.

Read the bind route through U and K. The unit row supplies the source admission boundary, and the Kleisli row supplies the displayed consumer route before any completion seal can be exposed.

The structural rows H and C only transport and replay the same finite route. Together with P and N, they keep the consumer inside the cited sibling certificates, so the packet has no coordinate from which to recover quotient completion equality, selected representatives, countable choice, host completed-object equality, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{CauchyCompletionBind}^{\uparrow}$ read is refuted if a completion consumer obtains a two-step bind without first exposing the source completion row, unit row, Kleisli row, completion seal, transport, replay, provenance, and local naming rows.

Independence witness: cauchycompletionmonad, cauchycompletionunit, cauchycompletionkleisli, bishopregularcauchycompletion, real: none is bijective with $\mathsf{CauchyCompletionBind}^{\uparrow}$ because the bind packet jointly carries the source-completion row, unit admission, Kleisli consumer route, completion seal, componentwise transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionBind}^{\uparrow}$ packet is generated from source completion rows, unit rows, Kleisli rows, a completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3777.1, Definition 3777.2, and Definition 3777.3 bind completion-facing Kleisli composition to finite source, unit, seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionBind}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not assert quotient completions, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CauchyCompletionBind}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, source-completion exposure, unit compatibility, Kleisli bind exposure, associativity-ledger exactness, and non-escape for BEDC.Derived.CauchyCompletionBindUp.

Chapter 3778

A Concrete Naming Certificate for $\mathsf{GlobalFiberSpectra}^{\uparrow}$

The $\mathsf{GlobalFiberSpectra}^{\uparrow}$ packet realizes the global-fiber spectra branch of Definition 9462.1167 and Definition 9462.1169 as finite BEDC NameCert data. It sits beside the six-boundary tile coverage packet of chapter 3470, the window-coverage trigger algebra of chapter 3478, and the greedy spectral closure packet of chapter 3753; those siblings expose coverage, trigger, and closure faces, while this packet records the global tripleton, singleton, edge-spectrum, tile, trigger, transport, replay, provenance, and naming rows together.

Definition 3778.1 (GlobalFiberSpectra carrier)**.** *A* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, T, E, L, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the global codon-cube fiber row recording tripleton, singleton, doubleton, quartet, and sextupleton multiplicities for the standard-code classifier.*
2. *T is the complete global fiber tile row recording the displayed Ile/Met and Stop/Trp $3+1$ tiles.*
3. *E is the edge-spectrum row recording same-class and cross-class Hamming edge counts by bit direction.*
4. *L is the sibling tile-coverage row over* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *and finite standard-tile enumeration data.*
5. *R is the trigger handoff row over* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *and* $\mathsf{GreedySpectralClosure}^{\uparrow}$*.*
6. *H is the componentwise* hsame *transport row for the fiber, tile, edge, coverage, trigger, replay, provenance, and local naming data.*
7. *C is the displayed* Cont *replay row for standard-code reads.*
8. *P is the* Pkg *provenance over* $\mathsf{GlobalFiberSpectra}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{GreedySpectralClosure}^{\uparrow}$*, finite codon-cube tiles,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden genetic-code oracle, biochemical causation claim, infinite codon family, non-displayed trigger semantics, or host probability model.

Lean4 ✓ `BEDC.Derived.GlobalFiberSpectraUp.GlobalFiberSpectraCarrier event flow rows`

Theorem 3778.2 (GlobalFiberSpectra NameCert obligations)**.** *For every accepted* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *carrier* $G = (F, T, E, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$ *surface consists exactly of global fiber multiplicity, complete global fiber tile exposure, edge-spectrum exposure, six-boundary tile coverage, trigger handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A standard-code read cannot expose trigger interpretation before the global tripleton, singleton, edge-spectrum, tile, and sibling coverage rows have been displayed.*

Proof. Project the carrier rows. The fiber row F, tile row T, and edge row E are the only global-spectrum data available to the packet.

Route sibling dependence through L and R. The coverage row keeps the packet tied to six-boundary and standard-tile data, while the trigger handoff row records the window-coverage and greedy-closure interfaces without exposing trigger interpretation early.

Use H, C, P, N to transport, replay, source, and name the same finite route. Since no hidden oracle, biochemical causation coordinate, infinite family, non-displayed trigger semantics, or host probability model is present, the listed rows exhaust the NameCert surface. □

Theorem 3778.3 (GlobalFiberSpectra sibling separation)**.** *An accepted* $\mathsf{GlobalFiberSpectra}^{\uparrow}$ *packet is not determined by any one of* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*,* $\mathsf{GreedySpectralClosure}^{\uparrow}$ *or finite standard-tile enumeration alone. The packet jointly carries global fiber multiplicity* F*, complete* $3 + 1$ *tile rows* T*, Hamming edge-spectrum rows* E*, sibling coverage* L*, trigger handoff* R*, transport* H*, replay* C*, provenance* P*, and local naming* N*.*

Proof. Read each sibling as a proper face. Six-boundary coverage supplies tile coverage, window-coverage trigger algebra supplies trigger handoff, greedy spectral closure supplies closure rows, and standard-tile enumeration supplies finite local tiles.

The global fiber and edge rows are additional coordinates. They record global tripleton, singleton, and direction-sensitive edge spectra before trigger interpretation is exposed.

Therefore no sibling alone is bijective to the packet. The structural rows H, C, P, N preserve the joint route and prevent any one face from replacing the whole $\mathsf{GlobalFiberSpectra}^{\uparrow}$ carrier. □

Falsifiable prediction: Every accepted $\mathsf{GlobalFiberSpectra}^{\uparrow}$ standard-code read factors through displayed global tripleton, singleton, edge-spectrum, and tile rows before any trigger interpretation is exposed.

Independence witness: sixboundarytilecoverage, windowcoveragetriggeralgebra, greedyspectralclosure, standardtileenumeration: $\mathsf{GlobalFiberSpectra}^{\uparrow}$ is not bijective to these siblings because it jointly carries global fiber multiplicity, local tile spectrum, edge counts, trigger handoff, transport, replay, provenance, and local naming.

The Lean follow-up in `lean4/BEDC/Derived/GlobalFiberSpectraUp.lean` must provide explicit Nontrivial $\mathsf{GlobalFiberSpectra}^{\uparrow}$ and FieldFaithful $\mathsf{GlobalFiberSpectra}^{\uparrow}$ TasteGate instances preserving $F, T, E, L, R, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GlobalFiberSpectra}^{\uparrow}$ packet is generated from finite codon-cube fiber rows, complete tile rows, edge-spectrum rows, sibling coverage rows, trigger handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$, seed)

Scope closed: Definition 3778.1, Definition 3778.2, and Definition 3778.3 bind the global-fiber spectra vision to finite codon-cube fiber, tile, edge, trigger, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GlobalFiberSpectra}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not assert biochemical causation, hidden genetic-code optimality, host probability semantics, non-displayed trigger interpretation, Lean verification, or bridge checking for GlobalFiberSpectra$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, global-fiber exposure, edge-spectrum exactness, sibling-separation exactness, Nontrivial and FieldFaithful instances, and trigger nonescape for BEDC.Derived.GlobalFiberSpectraUp.

Chapter 3779

A Concrete Naming Certificate for MetaCICNormalizationAuditPacket$^{\uparrow}$

The MetaCICNormalizationAuditPacket$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization surface of chapter 9383 as finite NameCert data. It sits beside the normalization audit packet of chapter 750, the critical-path packet of chapter 765, the normalization frontier of chapter 892, the candidate-evidence strong-normalization route of chapter 758, and the typed-normalization audit of chapter 3539. The packet names the closed typed term row, candidate or candidate-set evidence, candidate-to-SN projection, no-infinite-reduction projection, application and Pi adequacy rows, refusal frontier, transport, replay, provenance, and local naming rows needed before a public SN read may be cited. It does not assert unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Definition 3779.1 (MetaCIC normalization audit packet carrier)**.** *A* MetaCICNormalizationAuditPacket$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, K, A, S, I, P, F, H, C, G, N)$$

with the following displayed rows:

1. *T is the closed typed term row, exposing the term, type, empty-context read, and closedness data.*
2. *K is the candidate-evidence row, carrying* Candidate*, closed-candidate, or candidate-set membership evidence for the displayed term.*
3. *A is the application adequacy row, recording the function-side Pi evidence, argument evidence, and application-candidate handoff used when the term route contains an application.*
4. *S is the strong-normalization projection row, reading SN only through the candidate or candidate-set evidence displayed in K.*
5. *I is the no-infinite-reduction row, projecting the same candidate-mediated SN evidence to the absence of infinite reduction chains.*
6. *P is the Pi-candidate adequacy row, recording the Pi route used by dependent function cases without replacing K.*
7. *F is the normalization-frontier refusal row, kept visible whenever a requested citation would erase candidate evidence, subject-reduction boundaries, or confluence boundaries.*

8. *H records componentwise* hsame *transport for the closed typed term, candidate evidence, application adequacy, SN projection, no-infinite-reduction projection, Pi adequacy, frontier, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a normalization-audit request through $T, K, A, S, I, P, F, H, G, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$*, chapter 9383,* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$*,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, A, S, I, P, F, H, C, G, N$. No row stores a theorem that every closed typed MetaCIC term normalizes without candidate or candidate-set evidence.

Theorem 3779.2 (MetaCIC normalization audit packet NameCert obligations)**.** *For the carrier of Definition 3779.1, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$ *surface consists exactly of carrier admission for $M = (T, K, A, S, I, P, F, H, C, G, N)$, closed typed term admission through T, candidate or candidate-set evidence admission through K, application adequacy through A, candidate-mediated SN projection through S, no-infinite-reduction projection through I, Pi-candidate adequacy through P, frontier retention through F, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through G, and local naming through N.*

```
Lean4 ✓  BEDC.Derived.MetaCICNormalizationAuditPacketUp.MetaCICNormalizationAuditPacketCarrier
namecert obligations
```

Proof. Project the carrier. The only term source is T, and the only evidence source for a public SN read is the candidate row K.

Read A and P as adequacy rows for application and Pi cases. They refine the same candidate-mediated route and do not create an evidence-free normalization source.

The SN row S is downstream of K. The no-infinite-reduction row I is downstream of S, so absence of infinite reduction chains is cited only through the same candidate or candidate-set evidence.

The frontier row F records requests that would erase the evidence boundary. Transport, replay, provenance, and local naming through H, C, G, N preserve these rows without adding subject reduction, confluence, quotient syntax equality, host evaluator soundness, or closed consistency. □

Theorem 3779.3 (MetaCIC normalization audit packet candidate handoff)**.** *Every public strong-normalization or no-infinite-reduction read from an accepted* $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ *carrier factors through the finite route*

$$T \longmapsto K \longmapsto (A, P) \longmapsto S \longmapsto I \longmapsto F.$$

In particular, a public SN read must pass through candidate or candidate-set evidence before the closed typed term can be read as strongly normalizing, and the exported packet is not an unconditional normalization theorem for all closed typed MetaCIC terms.

Proof. Start with a public SN request. The term row T displays the closed typed term, but the carrier does not allow T alone to be read as normalized.

The request must next pass through K. Candidate, closed-candidate, or candidate-set evidence is the only input accepted by the projection row S, while A and P supply only the application and Pi adequacy data required by that same route.

The no-infinite-reduction row I reuses the output of S. If a consumer asks for evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or closed consistency, the frontier row F keeps that request outside the packet.

Thus the handoff exports candidate-mediated SN and no-infinite-reduction evidence, not a theorem that every closed typed MetaCIC term normalizes without the displayed candidate or candidate-set rows. □

Theorem 3779.4 (MetaCIC normalization candidate interface composition)**.** *An accepted* MetaCICNormalizationA *carrier whose candidate row K is accepted exports the three-step route from candidate evidence to strong normalization and then to no-infinite-reduction evidence:*

$$K \longmapsto S \longmapsto I.$$

The route preserves the closed typed term row T, application and Pi adequacy rows A, P, frontier row F, and structural rows H, C, G, N. It does not erase candidate evidence into an unconditional closed-term strong-normalization claim.

Proof. Apply Definition 3779.2. The local NameCert surface exposes the candidate row K, the strong-normalization projection S, and the no-infinite-reduction projection I as distinct displayed rows.

Apply Definition 3779.3. Public SN and no-infinite-reduction reads must pass through the route $T \longmapsto K \longmapsto (A, P) \longmapsto S \longmapsto I \longmapsto F$, so the candidate evidence remains visible at the interface.

The Lean-facing normalization API follows the same discipline: `candidate_implies_sn` reads candidate evidence before SN, `closed_typed_term_sn_from_candidate` exports closed typed SN from that evidence, and `closed_typed_term_no_infinite_reduction_from_candidate` projects the resulting SN evidence to absence of infinite reductions. These names record the conditional route; they do not license an evidence-free theorem for every closed typed MetaCIC term. □

Theorem 3779.5 (MetaCIC normalization audit packet residual-candidate route)**.** *For an accepted* MetaCICNormalizationAuditPacket$^{\uparrow}$ *carrier $M = (T, K, A, S, I, P, F, H, C, G, N)$, every residual normalization or no-infinite-reduction read must expose candidate or candidate-set evidence in row K before application adequacy A, Pi adequacy P, strong-normalization projection S, and no-infinite-reduction projection I can be consumed. The structural rows H, C, G, N preserve the displayed route, and the frontier row F prevents evidence-free residual normalization from being exported.*

Proof. Project the carrier of Definition 3779.1. The rows $T, K, A, S, I, P, F, H, C, G, N$ are the full classifier-visible surface for residual normalization and no-infinite-reduction reads.

Use Definition 3779.2. It identifies K, A, S, I, P, F as the candidate-evidence row, application adequacy row, SN projection row, no-infinite-reduction projection row, Pi adequacy row, and frontier row.

Apply Definition 3779.3 and Definition 3779.4. Together they order candidate evidence before S and I, while A and P remain adequacy rows attached to the same candidate-mediated route.

Finally use row exhaustion for F, H, C, G, N. The frontier row records requests that would erase candidate evidence, and the structural rows transport, replay, source, and name the displayed route without creating evidence-free normalization. □

Falsifiable prediction: Every public strong-normalization or no-infinite-reduction read of a $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ carrier factors through closed typed term data, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated SN projection, no-infinite-reduction projection, frontier retention, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: metacicconfluenceauditpacket, metaciccriticalpath, metacicnormalizationfrontier, closedtermsubstitutionboundary: $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries closed typed term data, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated SN projection, no-infinite-reduction projection, frontier retention, transport, replay, provenance, and local naming as one audit carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}$ packet is generated from a closed typed term row, candidate or candidate-set evidence, application and Pi adequacy rows, candidate-mediated strong-normalization projection, no-infinite-reduction projection, frontier retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3779.1, Definition 3779.2, and Definition 3779.3 seed the finite candidate-mediated normalization audit packet for public SN and no-infinite-reduction reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationAuditPacket}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetaCICNormalizationAuditPacketUp.MetaCICNormalizationAuditPacketCarrier\_namecert\_obl`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Upgrade path: Obligation closure requires checked carrier admission, candidate-evidence exactness, application and Pi adequacy routing, SN projection, no-infinite-reduction projection, frontier retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalizationAuditPacketUp.

Chapter 3780

A Concrete Naming Certificate for $\mathsf{StepFunction}^{\uparrow}$

Following Bourbaki and Bishop–Bridges constructive analysis, the $\mathsf{StepFunction}^{\uparrow}$ packet records a finite partition-to-real approximation object as BEDC NameCert data. It sits after the partition surface of chapter 201, the dyadic step-function specialization of chapter 428, the tagged finite-sum route of chapter 1615, the integration boundary of chapter 123, and the regular rational readback of chapter 301. The packet supplies a textbook finite step-function interface without forcing every integration-facing consumer through dyadic endpoints alone.

Definition 3780.1 (Step function carrier)**.** *A* $\mathsf{StepFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, V, A, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is a* $\mathsf{Partition}^{\uparrow}$ *row listing finitely many ordered cells and their boundary incidence data.*
2. *V assigns a displayed* $\mathsf{Real}^{\uparrow}$ *value row to each cell of P.*
3. *A records the constant-on-cell admission relation, so every accepted domain read first chooses a cell of P and then reads the corresponding value in V.*
4. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *comparison row, present when the partition cells and values are supplied through dyadic subrows.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *finite-sum row obtained from the displayed cells, tags, and values.*
6. *I is the* $\mathsf{Integral}^{\uparrow}$ *consumer boundary that may read S only after P, V, A, D have been displayed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *approximation ledger used when a later real-facing route samples the finite row by regular rational names.*
8. *H records componentwise* hsame *transport for partition, value, cell-admission, dyadic comparison, finite-sum, integral-boundary, regular-readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a step-function read through P, V, A, D, S, I, R, H.*
10. *Q is the* Pkg *provenance over* $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{StepFunction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these rows. It has no coordinate for a measurable-function quotient, an ambient function space, a selected refinement limit, open-cover compactness, or an unbounded integration procedure.

Theorem 3780.2 (Step function NameCert obligations)**.** *For every accepted* $\mathsf{StepFunction}^{\uparrow}$ *carrier* $F = (P, V, A, D, S, I, R, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StepFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure, real-value exposure, constant-on-cell admission, dyadic comparison when present, finite-sum handoff, integral-boundary handoff, regular rational readback, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.StepFunctionUp.StepFunctionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3780.1. The rows P, V, A display the finite partition, the value row on each cell, and the constant-on-cell admission relation before any consumer can read a function value.

The analytic consumers are downstream of those rows. The dyadic comparison row D is only a displayed specialization, the finite-sum row S is computed from the visible cells and values, and the integral boundary I reads the same finite row before any integration claim is exported.

The regular readback and structural rows preserve that route. Since R, H, C, Q, N name only regular sampling, transport, replay, provenance, and the local certificate, and since the carrier has no coordinate for a host function quotient, selected refinement limit, compactness proof, or unbounded integration procedure, the listed rows exhaust the NameCert surface. □

Theorem 3780.3 (Step function partition-integral route)**.** *Every integration-facing read of an accepted* $\mathsf{StepFunction}^{\uparrow}$ *carrier factors through*

$$\mathsf{Partition}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

together with the $\mathsf{RegSeqRat}^{\uparrow}$ *approximation ledger and the structural rows* H, C, Q, N*. The route exports a finite partition-to-real approximation surface, not a quotient function object, a selected limiting mesh, measure-theoretic integration, open-cover compactness, or ambient completeness.*

Proof. Start with the displayed partition row. A read has no domain observation except the cells listed by P, and it has no value observation except the corresponding $\mathsf{Real}^{\uparrow}$ rows in V.

The step-function comparison and finite-sum handoff are then forced. The dyadic row D records only the dyadic specialization of the same cells and values, while S is the finite $\mathsf{RiemannSum}^{\uparrow}$ row computed from those displayed data.

Finally read the integral and approximation rows. The $\mathsf{Integral}^{\uparrow}$ boundary I is available only after P, V, A, D, S are displayed, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies regular rational sampling without changing the carrier. The structural rows H, C, Q, N transport and name the same route, so no external function quotient, mesh-limit choice, compactness principle, measure-theoretic integral, or ambient completeness coordinate can be exported. □

Falsifiable prediction: An accepted $\mathsf{StepFunction}^{\uparrow}$ integration-facing read is refuted if its $\mathsf{Integral}^{\uparrow}$ boundary is preserved while the displayed $\mathsf{Partition}^{\uparrow}$ row, $\mathsf{Real}^{\uparrow}$ value row, $\mathsf{RiemannSum}^{\uparrow}$ finite-sum row, and $\mathsf{RegSeqRat}^{\uparrow}$ readback are removed.

Independence witness: partition, real-value, dyadic-step-function, riemann-sum, integral, regseqrat: none is bijective with $\mathsf{StepFunction}^{\uparrow}$ because the packet jointly carries finite partition cells, displayed real values, constant-on-cell admission, dyadic comparison, finite-sum handoff, integral

boundary, regular rational readback, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}$.

Constructive story: A $\mathsf{StepFunction}^{\uparrow}$ packet is generated from a finite PartitionUp row, displayed RealUp values, constant-on-cell admission, optional DyadicStepFunctionUp comparison, finite RiemannSumUp handoff, IntegralUp boundary, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3780.1, Definition 3780.2, and Definition 3780.3 seed the finite partition-to-real step-function approximation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StepFunction}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.StepFunctionUp.StepFunctionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim measure-theoretic integration, host function quotients, selected refinement limits, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, real-value exposure, constant-on-cell admission, finite-sum handoff, IntegralUp boundary routing, RegSeqRatUp readback, and structural transport for BEDC.Derived.StepFunctionUp.

Chapter 3781

A Concrete Naming Certificate for $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$

Following Khinchin's continued fractions and Hardy–Wright's Stern–Brocot/Farey treatment, the $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ packet records the finite bridge by which Stern–Brocot interval refinement and continued-fraction convergent readback expose the same rational-approximation route before any $\mathsf{Real}^{\uparrow}$ completion-facing consumer reads it. It sits over the finite tree surface of chapter 1704, the bracketing approximation route of chapter 1431, the continued-fraction carrier of chapter 342, the convergent row surface of chapter 1777, the Farey-neighbor sequence route of chapter 1759, the rational source of chapter 37, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite NameCert handoff for the L10 rational-approximation lattice; it does not use quotient rational equality, selected representatives, host enumeration, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3781.1 (Stern–Brocot continued-fraction bridge carrier)**.** *A* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, A, F, Q, V, I, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *tree row, including the finite address, boundary, mediant, and Farey-neighbor ledger read by the bridge.*
2. *A is the* $\mathsf{SternBrocotApproximation}^{\uparrow}$ *bracket row, exposing the same inspected rational interval as a finite approximation surface.*
3. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *handoff row, recording the adjacent determinant-one boundary sequence used to compare tree intervals with continued-fraction cylinders.*
4. *Q is the* $\mathsf{ContinuedFraction}^{\uparrow}$ *partial-quotient spine row, restricted to the finite prefix inspected by the bridge.*
5. *V is the* $\mathsf{ContinuedFractionConvergents}^{\uparrow}$ *convergent row, including the numerator, denominator, normalized rational, dyadic error, and Real-facing handoff rows of the inspected prefix.*
6. *I is the rational-interval comparison ledger, read through* $\mathsf{Rat}^{\uparrow}$ *rows and the displayed determinant-one adjacency data rather than through host rational equality.*
7. *W is the finite window row scheduling the Stern–Brocot branch, Farey sequence, partial-quotient prefix, and convergent observations.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row built from the shared rational interval and convergent data.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, A, F, Q, V, I, W, R have been displayed.*
10. *H records componentwise* hsame *transport for the tree, approximation, Farey, continued-fraction, convergent, rational-comparison, window, readback, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a bridge request through Stern–Brocot boundary inspection, Farey adjacency, continued-fraction prefix inspection, convergent readback, regular rational handoff, and the terminal Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{SternBrocotApproximation}^{\uparrow}$, $\mathsf{ContinuedFraction}^{\uparrow}$, $\mathsf{ContinuedFractionConvergents}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$ *row.*

The classifier compares accepted bridge packets only through $S, A, F, Q, V, I, W, R, E, H, C, P, N$. The carrier has no coordinate for quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.SternBrocotContinuedFractionBridgeUp`
Lean4 ✓ `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.SternBrocotContinuedFractionBridgeTasteGate`
`single carrier alignment`
Lean4 ✓ `BEDC.Derived.SternBrocotContinuedFractionBridgeUp.TasteGate.SternBrocotContinuedFractionBridgeT`
`single carrier alignment`

Theorem 3781.2 (Stern–Brocot continued-fraction NameCert obligations)**.** *For every accepted* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *carrier*

$$B = (S, A, F, Q, V, I, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Stern–Brocot tree row, bracketing approximation row, Farey-sequence handoff, continued-fraction prefix row, convergent row, rational-interval ledger, finite window row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-prefix compatibility, requiring the Stern–Brocot branch in S and the continued-fraction prefix in Q to inspect the same displayed rational interval through F and I;*
3. *convergent handoff, requiring every rational convergent row in V to be read through the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *route before the terminal seal E is consumed;*
4. *approximation bridge discipline, requiring the bracketing row A and the Farey handoff F to mediate between Stern–Brocot intervals and continued-fraction convergents as finite* BHist *rows;*
5. *componentwise transport, displayed replay, package provenance, and local naming, requiring H, C, P, N to preserve the same bridge packet under* hsame *transport and* Cont *replay;*
6. *refusal of quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as bridge data.*

No local bridge obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3781.1. The accepted tuple exposes exactly the tree, approximation, Farey, continued-fraction, convergent, rational-comparison, window, readback, seal, transport, replay, provenance, and local naming rows listed in the statement.

Interval-prefix compatibility is read through S, F, Q, I. The Stern–Brocot side supplies a finite branch and boundary pair, the Farey side supplies determinant-one adjacency, the continued-fraction side supplies the inspected partial-quotient prefix, and the rational ledger I records that these rows refer to the same displayed rational interval.

The convergent handoff is downstream of V, W, R, E. A consumer first reads the scheduled finite convergent window, then the regular rational readback R, and only then reaches the Real seal E.

The approximation bridge is finite because A and F are displayed rows. They mediate from rational bracketing and Farey adjacency to the continued-fraction convergent route without importing a host enumeration of rationals or a selected representative.

Finally H, C, P, N transport, replay, package, and name the same packet. Since the carrier inventory contains no quotient rational equality, selected representative, host enumeration, choice, trichotomy, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3781.3 (Stern–Brocot continued-fraction Farey handoff)**.** *Let $B = (S, A, F, Q, V, I, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ *packet. Every bridge-facing read from a Stern–Brocot interval to a continued-fraction convergent factors through the finite route*

$$\mathsf{SternBrocot}^{\uparrow} \longmapsto \mathsf{SternBrocotApproximation}^{\uparrow} \longmapsto \mathsf{FareySequence}^{\uparrow} \longmapsto \mathsf{ContinuedFraction}^{\uparrow} \longmapsto \mathsf{ContinuedFractionCo}$$

The Farey handoff F is the only adjacency surface in the route, and the rational comparison ledger I is read as displayed $\mathsf{Rat}^{\uparrow}$ *data. Thus the bridge exports a Real-facing regular-rational handoff while refusing quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Stern–Brocot side. The row S exposes only a finite tree address, boundary rows, a mediant row, and a Farey-neighbor ledger, while A records the corresponding bracketing approximation surface.

Next read the Farey handoff F. It supplies determinant-one adjacency for the displayed interval boundary, so the bridge compares the tree interval with the continued-fraction cylinder through finite Farey data rather than through a host enumeration of rational numbers.

The continued-fraction side then reads Q and V. The prefix row Q names the finite partial quotients under inspection, and the convergent row V names the numerator, denominator, normalized rational, dyadic error, and regular handoff data for that prefix.

Finally read W, R, E, H, C, P, N. The finite window row schedules the common observation, R carries the $\mathsf{RegSeqRat}^{\uparrow}$ readback, E is terminal for the Real-facing route, and the structural rows preserve and name the same packet. The carrier has no coordinate for quotient rational equality, selected representatives, host enumeration, choice, trichotomy, or ambient completeness, so none is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}$ packet is generated from a Stern–Brocot tree row, a bracketing approximation row, a Farey adjacency handoff, a continued-fraction prefix row, a convergent row, a displayed rational-interval ledger, finite observation windows, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3781.1, Definition 3781.2, and Definition 3781.3 seed the finite bridge from Stern–Brocot interval refinement to continued-fraction convergent readback and Real-facing regular rational handoff.

Formal verification: FormalStatus(NameCert$_{\mathsf{SternBrocotContinuedFractionBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient rational equality, selected representatives, host enumeration, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-prefix compatibility, convergent handoff, Farey mediation, transport, replay, provenance, and local naming for the same bridge packet.

Chapter 3782

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ packet records the finite route by which a regular Cauchy family of regular rational names is read along a unary diagonal selector before any Real or Completion consumer treats the diagonal read as a completed real name. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the sibling diagonal extraction packet of chapter 1773. The packet gives the L10 Real and Completion route one displayed diagonalization carrier; it does not assert quotient stream equality, choose a host limit, invoke countable choice, use trichotomy, claim ambient $\mathsf{Real}^{\uparrow}$ completeness, provide Lean verification, or record a checked bridge.

Definition 3782.1 (Regular Cauchy diagonalization carrier)**.** *A* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, \delta, S, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the regular Cauchy family row. It displays the source family of rational names together with the finite precision requests inspected by the diagonal read.*
2. *δ is the unary diagonal selector row. It names the selected source index for each inspected precision and records no independent choice schedule beyond that displayed unary selector.*
3. *S is the aligned* $\mathsf{StreamName}^{\uparrow}$ *window row. It displays the finite windows of the selected source names and aligns them with the requests carried by F and δ.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the requested dyadic radius, split-tolerance budget, comparison windows, and common-scale data used by the diagonal read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row. It reads the selected stream windows through the tolerance ledger as one regular rational sequence.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, δ, S, T, R have been displayed.*
7. *H records componentwise* hsame *transport for the family, selector, stream windows, tolerance ledger, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a diagonalization request through the family row, unary selector, aligned windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, transports, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, \delta, S, T, R, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected host limits, countable-choice schedules, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Theorem 3782.2 (Regular Cauchy diagonalization NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier*

$$D = (F, \delta, S, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular Cauchy family, unary diagonal selector, aligned StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat diagonal readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *selector exactness, requiring every diagonal read to pass through the displayed unary selector δ and the family row F;*
3. *window alignment, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row S to expose exactly the selected finite windows named by F and δ;*
4. *tolerance exactness, requiring the requested radius, split budget, comparison windows, and common-scale data to be read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row T;*
5. *regular readback exactness, requiring R to consume the same selected windows and tolerance ledger before any Real-facing seal is visible;*
6. *Real-seal nonescape, requiring E to appear only after F, δ, S, T, R are transported through H and replayed through C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly the rows $F, \delta, S, T, R, E, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable-choice schedules, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No local diagonalization obligation is stored outside these rows.

Proof. Project the carrier of Definition 3782.1. The accepted inventory is exactly the family row F, unary selector δ, aligned StreamName windows S, dyadic tolerance ledger T, regular rational readback R, Real seal E, and structural rows H, C, P, N.

Selector exactness is read from F and δ. A consumer may inspect only the named source family entries selected by the displayed unary diagonal row, so no hidden choice schedule or host enumeration is available.

Window alignment and tolerance exactness are read from S and T. The stream row exposes the finite windows selected by the diagonal read, while the dyadic row supplies the requested radius, split budget, comparison windows, and common-scale ledger used to compare those windows.

Regular readback and the Real seal are downstream of the same route. The row R consumes S and T as one $\mathsf{RegSeqRat}^{\uparrow}$ diagonal readback, and E is visible only after that readback has been transported and replayed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since the carrier inventory contains no quotient-stream equality, selected host limit, countable-choice schedule, trichotomy, ambient completeness principle, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalizationUp.RegularCauchyDiagonalizationTasteGate single carrier alignment`

Theorem 3782.3 (Regular Cauchy diagonalization Real seal handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier factors through the finite route*

$$F, \delta \longmapsto S \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Equivalently, the consumer receives a regular Cauchy family row, one unary diagonal selector, aligned $\mathsf{StreamName}^{\uparrow}$ *windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback, and one terminal* $\mathsf{Real}^{\uparrow}$ *seal before any Real-facing output is named.*

Proof. Start with the obligation surface of Definition 3782.2. It confines every accepted packet to the displayed family, selector, stream-window, tolerance-ledger, readback, Real-seal, and structural rows.

The consumer first reads F and δ. These rows name the source family and the unary diagonal selector, so the diagonal route has a displayed source for every inspected precision request.

The consumer then reads S, T, R. The aligned stream windows expose the selected entries, the dyadic ledger supplies the tolerance and comparison budget, and the regular rational readback repacks those rows as one $\mathsf{RegSeqRat}^{\uparrow}$ diagonal sequence.

Only after that route is visible may the consumer read the terminal $\mathsf{Real}^{\uparrow}$ seal E and the structural rows H, C, P, N. Because the carrier has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking, none of those objects can enter the Real-facing handoff. □

Theorem 3782.4 (Regular Cauchy diagonalization selector-window exactness)**.** *For an accepted* $\mathsf{RegularCauchyDiagonalization}^{\uparrow}$ *carrier*

$$D = (F, \delta, S, T, R, E, H, C, P, N),$$

the unary diagonal selector row δ determines the aligned $\mathsf{StreamName}^{\uparrow}$ *window row S before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R are consumed. Any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of the diagonal seal must therefore use the same selected window with the tolerance ledger T before reading R or the terminal seal E.*

Proof. Apply Definition 3782.2. It states selector exactness through F, δ, window alignment through S, tolerance exactness through T, regular readback exactness through R, and Real-seal nonescape through E.

Then apply Definition 3782.3. It orders every Real or Completion consumer through

$$F, \delta \longmapsto S \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the selector fixes the window before the tolerance ledger and readback are visible, and the same selected window is preserved until the diagonal seal is read. □

Theorem 3782.5 (Regular Cauchy diagonalization common-tolerance lock)**.** *If two accepted* RegularCauchyDiagona *packets share the same regular Cauchy family row F, unary selector δ, aligned* StreamName$^{\uparrow}$ *window row S, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger T, then every* RegSeqRat$^{\uparrow}$ *readback and* Real$^{\uparrow}$ *seal handoff in both packets is locked to that common tolerance ledger. More explicitly, for*

$$D = (F, \delta, S, T, R, E, H, C, P, N)$$

and

$$D' = (F, \delta, S, T, R', E', H', C', P', N'),$$

every diagonal readback route through R or R' and every Real seal handoff through E or E' must consume the displayed ledger T after the same F, δ, S route.

Proof. Use the obligation theorem, Definition 3782.2, to project both carriers. Their shared rows F, δ, S, T are the only family, selector, window, and tolerance rows available before any readback or Real seal is exposed.

Apply selector-window exactness, Definition 3782.4. It fixes the route through F, δ, S before the tolerance ledger and readback rows are consumed.

Now apply the Real seal handoff theorem, Definition 3782.3. It keeps R, E and R', E' downstream of the same tolerance ledger T, with the structural rows only transporting, replaying, sourcing, and naming the displayed evidence.

Therefore both packets lock every RegSeqRat readback and Real seal handoff to the common ledger T. Since neither carrier contains quotient stream equality, selected host limits, countable choice, trichotomy, or ambient Real completeness, none of those excluded objects can loosen the common tolerance budget. □

Falsifiable prediction: Every RegularCauchyDiagonalization$^{\uparrow}$ Real or Completion read factors through a regular Cauchy family row, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-diagonal-extraction: RegularCauchyDiagonalization$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source regular Cauchy family, a unary diagonal selector, aligned finite windows, a dyadic tolerance ledger, a regular rational diagonal readback, and a terminal RealUp seal in one displayed packet.

Closure status for NameCert$_{\text{RegularCauchyDiagonalization}^{\uparrow}}$.

Constructive story: A RegularCauchyDiagonalization$^{\uparrow}$ packet is generated from a regular Cauchy family row, one unary diagonal selector, aligned StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyDiagonalization}^{\uparrow}}$, seed)

Scope closed: Definition 3782.1, Definition 3782.2, Definition 3782.3, Definition 3782.4, and Definition 3782.5 seed the finite diagonalization packet used by L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDiagonalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector exactness, window alignment, dyadic tolerance exactness, regular readback exactness, Real-seal exposure, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyDiagonalizationUp.

Chapter 3783

A Concrete Naming Certificate for $\mathsf{TotallyBoundedClosure}^{\uparrow}$

Following Bishop and Bridges on total boundedness and Kelley and Bourbaki on metric closure, the $\mathsf{TotallyBoundedClosure}^{\uparrow}$ packet records the finite BEDC route by which a displayed closure window of a totally bounded metric carrier remains available to finite-net consumers. It sits after the finite-net source of chapter 127, the metric-distance surface of chapter 57, the compact metric consumer of chapter 129, the closed-subspace completion route of chapter 1949, and the precompact bridge of chapter 1531. The packet names only displayed closure-window data and finite rational readbacks; it does not import host closure, selected limits, quotient completion, open-cover compactness, or ambient real completeness.

Definition 3783.1 (TotallyBoundedClosure carrier)**.** *A* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, M, F, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net source row, including the closed* $\mathsf{ProbeBundle}$ *center families,* $\mathsf{InBundle}$ *membership rows, center-carrier rows, distance witnesses, and rational bound ledgers of Definition 127.3;*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row for the same source, exposing endpoint classifiers, distance-witness reads, triangle ledgers, and the exact comparison rows used by closure requests;*
3. *F is the finite closure-window ledger, naming the displayed approximating source points, their tolerance windows, and the inclusion route from each window point into the closure carrier;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records only the finite observation prefixes needed by a closure consumer before a net center is inspected;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational tails attached to the closure windows and the finite-net tolerances;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint distance bounds and closure radii already exposed by M, F, W, R;*
7. *H records componentwise* hsame *transport for all rows T, M, F, W, R, E and for every center, window point, rational tail, real seal, and metric-distance history read from them;*

8. *C records the displayed* Cont *replay route from closure-window reads to the source finite-net row, then back through the metric distance ledger to the closure carrier;*
9. *P is the* Pkg *provenance over* $\mathsf{TotallyBoundedClosure}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$, $\mathsf{PrecompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$ *row.*

The classifier on such packets is componentwise transport through H together with replay of the finite routes in C. The carrier has no coordinate for a host closure operator, a choice-selected convergent sequence, quotient completion equality, an open-cover compactness principle, or an ambient completeness theorem.

Theorem 3783.2 (TotallyBoundedClosure NameCert obligations)**.** *For an accepted* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *carrier $K = (T, M, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$ *surface consists exactly of six obligations:*

1. *carrier admission: every row exposed to a consumer is one of the ten displayed rows or a finite subrow of the totally bounded source, metric distance ledger, closure-window ledger, stream window, rational readback, or real seal;*
2. *classifier transport: packet equality is componentwise* hsame *transport of centers, source points, closure-window points, stream prefixes, rational tails, real seals, metric distances, provenance rows, and local naming rows;*
3. *finite-net inheritance: each positive tolerance read reaches the closed* ProbeBundle*, membership constructors, center histories, distance witnesses, and rational comparison ledgers already named by T;*
4. *closure-window exactness: each closure point read first passes through a finite window in F, W, R, E before the metric row M compares it with a displayed source point;*
5. *replay discipline: every handoff from a closure-window point to a finite net factors through the displayed* Cont *route in C and the same source metric rows in M;*
6. *provenance and non-escape: P and N name only the listed source certificates, and the packet exports no host closure, selected limit, quotient completion, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

These obligations make total boundedness under displayed closure a finite-row interface rather than a semantic closure theorem in an ambient metric space.

Proof. Carrier admission is projection from Definition 3783.1. The ten rows list the finite-net source, metric distance row, closure-window ledger, stream windows, regular rational readbacks, real seals, transport ledger, replay ledger, provenance row, and local naming row.

Classifier transport is the role of H. It moves the source centers, closure-window points, rational tails, real seals, and metric-distance histories by hsame, while the point and distance classifiers in M keep the source and closure endpoint reads aligned.

Finite-net inheritance is read from T. A positive tolerance request reaches exactly the $\mathsf{TotallyBounded}^{\uparrow}$ ProbeBundle spine, its InBundle membership constructors, center-carrier rows, coverage-distance witnesses, and rational comparison ledgers.

Closure-window exactness is supplied by F, W, R, E. The packet never asks for a host closure membership proof; it displays the finite windows, the stream prefixes, the rational tail reads, and the real endpoint seals that justify each closure read before any finite net is consumed.

Finally, C, P, N package the handoff. The replay route sends a closure-window point to a displayed source approximation, reads the totally bounded center through T, compares the closure point through M, R, E, and returns only the named finite rows to compact, completion, or precompact consumers. Since no other coordinate is present, host closure, selected limits, quotient completion, open-cover compactness, countable choice, and ambient real completeness cannot be exported. □

Lean4 ✓ `BEDC.Derived.TotallyBoundedClosureUp.TotallyBoundedClosureTasteGate single carrier alignment`

Theorem 3783.3 (TotallyBoundedClosure finite-net closure handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedSubspaceCompletion}^{\uparrow}$, *or* $\mathsf{PrecompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{TotallyBoundedClosure}^{\uparrow}$ *packet* $K = (T, M, F, W, R, E, H, C, P, N)$ *reads total boundedness of the closure only through the finite route*

$$\mathsf{TotallyBounded}^{\uparrow}, \mathsf{Met}^{\uparrow}, F, W, R, E, H, C, P, N.$$

For each positive tolerance history, the consumer receives a displayed source net from T, a closure-window approximation from F, W, R, E, and the metric comparison row from M that transfers the source-net bound to the closure point. The route exports no host closure operator, no chosen dense sequence, no quotient completion, no selected subcover, and no ambient completeness principle.

Proof. Fix a positive tolerance request made by one of the listed consumers. The request is first routed by C to the finite-net source row T, so the available centers and coverage witnesses are exactly the closed ProbeBundle rows of chapter 127.

For a closure point, the packet then reads a finite window in F together with the prefix row in W, the regular rational tail row in R, and the real seal in E. These rows display a source approximation close enough for the requested tolerance; no closure predicate outside the packet is inspected.

The metric row M compares the closure point, the displayed source approximation, and the selected net center through the same endpoint classifiers and distance ledgers. Transport by H and replay by C make the comparison stable under the packet classifier.

Thus any compact-metric, closed-subspace-completion, or precompact consumer sees only the finite-net source, closure windows, rational readbacks, real seals, metric comparisons, transport, replay, provenance, and local NameCert rows. The excluded host closure, selected dense sequence, quotient completion, selected subcover, and ambient completeness data have no carrier coordinate from which they could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3783.1, Definition 3783.2, and Definition 3783.3 seed the finite L10 closure interface over totally bounded source nets, metric distance rows, finite closure windows, stream windows, rational readbacks, real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotallyBoundedClosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host closure operators, selected dense sequences, quotient completions, selected subcovers, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, finite-net inheritance, closure-window exactness, replay discipline, and provenance non-escape for the same packet.

Constructive story: A closure consumer reads a finite window around the requested closure point, routes that window back to the totally bounded source net, compares the displayed source approximation by the metric ledger, and exports only the resulting finite rows.

Chapter 3784

A Concrete Naming Certificate for $\mathsf{UniformModulusComposition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformModulusComposition}^{\uparrow}$ packet records the finite bookkeeping needed to compose displayed uniform moduli for map-level continuity before a continuous-map or completion-facing consumer reads the result. It sits beside the map carrier of chapter 130, the modulus surface of chapter 2228, the Cauchy budget composition of chapter 1617, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real boundary of chapter 38. The packet names only the displayed composition route for uniform-modulus rows; it is not a quotient stream, not a selected limit, not host real equality, not trichotomy, not ambient Real completeness, and not a bridge assertion.

Definition 3784.1 (Uniform modulus composition carrier)**.** *A* $\mathsf{UniformModulusComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *F and G are composable* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, with G read after the codomain row exported by F.*
2. *M_F and M_G are the displayed* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *rows for F and G.*
3. *B is the composed uniform-budget row. It first reads the precision request through M_G, then uses the resulting domain tolerance as the request for M_F, and stores the single source tolerance supplied to the composite map.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window route through which the composed budget may inspect only displayed finite observations.*
5. *D is the dyadic or rational tolerance ledger naming every radius comparison used by the two modulus lookups.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that receives the composed source tolerance after W and D have been replayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *route row that may be read only after the regular-rational handoff has consumed the same composed budget.*

8. *H records componentwise* hsame *transport for $F, G, M_F, M_G, B, W, D, R, E$, the continuation row, the provenance row, and the local naming row.*
9. *C records displayed* Cont *replay from a requested output tolerance through M_G, M_F, the composed budget B, stream windows W, dyadic or rational tolerances D, the RegSeqRat handoff R, and the Real route E.*
10. *P is the* Pkg *provenance over* ContinuousMap$^\uparrow$, UniformContinuityModulus$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformModulusComposition}^\uparrow}$ *row.*

The carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, ambient completeness of Real$^\uparrow$, *or standard bridge data.*

Lean4 ✓ BEDC.Derived.UniformModulusCompositionUp.TasteGate.UniformModulusCompositionTasteGate
single carrier alignment

Theorem 3784.2 (Uniform modulus composition NameCert obligations)**.** *For every accepted* UniformModulusComposition$^\uparrow$ *carrier $K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$, the local* NameCert$_{\text{UniformModu}}$ *surface consists exactly of carrier admission for the two composable map rows, admission of their displayed uniform-modulus rows, budget composition through M_G then M_F, window and tolerance exposure through W and D, RegSeqRat and Real handoff through R and E, componentwise transport stability through H, continuation replay through C, package provenance through P, and local naming through N. No obligation row stores a quotient stream, selected limit, host equality of reals or maps, trichotomy, ambient completeness principle, or standard bridge claim.*

Proof. Project the carrier of Definition 3784.1. The tuple displays exactly the two continuous-map rows, their uniform-modulus rows, the composed budget, finite-window route, tolerance ledger, RegSeqRat handoff, Real route, transport, continuation, provenance, and naming row.

The composition order is fixed by B. An output tolerance for the composite is first read through M_G because G controls the second map. The tolerance exposed by that read is then passed to M_F, and the resulting source tolerance is stored as one composed budget row.

The consumer route is ordered by C. Before any regular-rational or Real-facing read can consume the composed budget, it must replay the finite windows W and the tolerance ledger D.

Transport stability is componentwise. The row H replaces only displayed rows by hsame-classified rows and replays the same continuation order, so it cannot introduce a new equality principle.

The carrier contains no quotient-stream, selected-limit, host-equality, trichotomy, completeness, or bridge coordinate. The listed rows therefore exhaust the local NameCert obligation surface. □

Theorem 3784.3 (Uniform modulus composition Real route)**.** *Let $K = (F, G, M_F, M_G, B, W, D, R, E, H, C, P, N)$ be an accepted* UniformModulusComposition$^\uparrow$ *carrier. Every Real-facing read of the composite map reaches the* Real$^\uparrow$ *route only after the composed uniform budget has passed through the two map rows, the two displayed uniform-modulus rows, the StreamName windows, the tolerance ledger, and the RegSeqRat handoff. The exported surface contains no quotient stream, selected limit, host real equality, trichotomy, ambient Real completeness assertion, or standard bridge claim.*

Proof. The Real route row E is displayed in the carrier, but its accessibility is ordered by C. A Real-facing read must therefore replay $F, G, M_G, M_F, B, W, D, R$ before E is available.

The replay keeps the composed modulus finite. The two source map rows supply the composability context, M_G and M_F supply the only tolerance lookups, W supplies the only stream-window route, D supplies the only tolerance ledger, and R supplies the only regular rational handoff.

The support rows H, P, N name and transport that same finite packet. They cannot introduce quotient streams, selected limits, host equality, trichotomy, ambient completeness, or bridge data, because none of those data appear in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformModulusComposition}^{\uparrow}$ packet is generated from two composable ContinuousMap rows, two UniformContinuityModulus rows, a composed uniform-budget row, a StreamName window route, a tolerance ledger, a RegSeqRat handoff, a RealUp route, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3784.1, Definition 3784.2, and Definition 3784.3 seed the finite map-level route that composes displayed uniform moduli before RegSeqRat and Real consumers read them.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget composition, window and tolerance exposure, RegSeqRat handoff, Real route nonescape, and transport stability for BEDC.Derived.UniformModulusCompositionUp.

Chapter 3785

A Concrete Naming Certificate for $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet names a closed constructive interval by regular Cauchy data rather than by quotient real equality. It sits between the closed bounded interval rows of chapter 1591 and the completion-facing consumers of chapter 301, chapter 126, chapter 291, and chapter 38.

Definition 3785.1 (Constructive Cauchy interval carrier)**.** *A* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, O, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are lower and upper located endpoint rows inherited from a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source.*
2. *O is the endpoint-order row asserting that the lower endpoint is not above the upper endpoint in the located order used by the source interval.*
3. *D is the dyadic containment row over* $\mathsf{DyadicRatCore}^{\uparrow}$*; it names the finite dyadic windows that contain the interval at each displayed precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the finite stream of containment windows read by Cauchy consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row; it turns each accepted containment window into regular rational sequence data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row recording the real-completion consumer reached from R without quotient real equality, trichotomy, choice, or an ambient completeness axiom.*
7. *H records componentwise* hsame *transport for endpoint, order, dyadic, stream, regular-sequence, real-seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from any interval-completion read through L, U, O, D, S, R, E, H.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$*,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, O, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected interval points, quotient real equality, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ConstructiveCauchyIntervalUp.ConstructiveCauchyIntervalTasteGate single carrier alignment`

Theorem 3785.2 (Constructive Cauchy interval NameCert obligations)**.** *For every accepted* ConstructiveCauchyInt *carrier $I = (L, U, O, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$ *surface consists exactly of endpoint admission, endpoint-order preservation, dyadic containment exposure, StreamName window exposure, RegSeqRat handoff, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3785.1. The rows L, U, O are inherited from the closed bounded interval source and therefore supply the only endpoint data visible to the packet.

Next read the completion route. The dyadic row D exposes finite containment windows, S names their stream surface, R supplies the regular rational sequence handoff, and E seals the route as a Real-facing completion consumer.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Since the excluded host objects are not carrier coordinates, the local NameCert obligations are exactly the listed rows. □

Theorem 3785.3 (Constructive Cauchy interval RegSeqRat handoff)**.** *Every completion-facing read of an accepted* $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ *packet factors through*

$$L, U, O \longmapsto D \longmapsto S \longmapsto R \longmapsto E, H, C, P, N.$$

In particular, any RegSeqRat or Real consumer sees only the located endpoints, endpoint order, dyadic containment ledger, StreamName windows, RegSeqRat handoff row, Real seal, transport, replay, provenance, and local NameCert row.

Proof. Start with the endpoint rows. The closed bounded interval source fixes L, U, O, so a completion-facing consumer cannot introduce a new endpoint classifier or a selected point of the interval.

The containment route is finite. Dyadic containment D supplies the displayed windows, the stream row S names their readback order, and the handoff row R exposes exactly the regular rational sequence surface consumed by $\mathsf{RegSeqRat}^{\uparrow}$.

Finally the Real seal E and the structural rows H, C, P, N preserve the same route for downstream readers. Thus every RegSeqRat or Real-facing read factors through the displayed rows and does not claim open-cover compactness, quotient real equality, trichotomy, countable choice, or ambient completeness. □

Falsifiable prediction: Every completion-facing read of a $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet factors through located endpoints, endpoint order, dyadic containment, StreamName windows, RegSeqRat handoff, a RealUp seal, transport, replay, provenance, and local naming.

Independence witness: closedboundedinterval, regseqrat, streamname, dyadicratcore, real: ConstructiveCauchyI is not bijective to any listed sibling because it jointly carries interval endpoints, endpoint order, dyadic containment, stream windows, regular rational handoff, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCauchyInterval}^{\uparrow}$ packet is generated from located endpoint rows, an endpoint-order row, dyadic containment data, StreamName windows, RegSeqRat handoff data, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3785.1, Definition 3785.2, and Definition 3785.3 seed the constructive Cauchy interval completion packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCauchyInterval}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConstructiveCauchyIntervalUp.ConstructiveCauchyIntervalTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient real equality, trichotomy, selected interval points, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-order preservation, dyadic containment exposure, StreamName window exposure, RegSeqRat handoff, and Real seal nonescape for BEDC.Derived.ConstructiveCauchyIntervalUp.

Chapter 3786

A Concrete Naming Certificate for CauchyCompletionDenseReflector$^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley on dense embeddings, the CauchyCompletionDenseRef packet records the dense-reflector face of Cauchy completion as finite NameCert data. It reads the completion transformer of chapter 378, the dense image surface of chapter 3351, and the universal mapping row of chapter 1456. Its local name route is the finite sequence of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the classical dense-reflection argument without importing quotient completions, selected limits, countable choice, host equality, or ambient completeness.

Definition 3786.1 (CauchyCompletionDenseReflector carrier)**.** *A* CauchyCompletionDenseReflector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, I, A, W, T, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source completion schedule, read through the* CauchyCompletionMonad$^{\uparrow}$ *unit, bind, and public* Real$^{\uparrow}$ *seal route rather than through a quotient completion.*
2. *I is the dense-image row inherited from* MetricCompletionDenseEmbedding$^{\uparrow}$*, exposing the source metric image inside the completion seal as finite approximation data.*
3. *A is the adjunction row inherited from* CauchyCompletionAdjunction$^{\uparrow}$*, comparing completion-facing consumers with source-side finite-window consumers over the same dense image.*
4. *W is the finite* StreamName$^{\uparrow}$ *window row through which the source and completion observations are inspected.*
5. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used by the source schedule, dense-image row, adjunction row, and real-seal readback.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the displayed stream windows and dyadic tolerance ledger.*
7. *R is the terminal* Real$^{\uparrow}$ *seal row reached only after S, I, A, W, T, Q have been displayed.*
8. *H records componentwise* hsame *transport for the schedule, dense-image, adjunction, window, tolerance, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the source completion schedule through the dense image, adjunction row, finite windows, dyadic tolerance, regular readback, and real seal.*

10. *P is the* Pkg *provenance over* CauchyCompletionDenseReflector$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, MetricCompletionDe CauchyCompletionAdjunction$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyCompletionDenseReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion representatives, selected limits, countable-choice schedules, host equality of completed objects or real names, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseReflectorUp.CauchyCompletionDenseReflectorTasteGate`
`single carrier alignment`

Theorem 3786.2 (CauchyCompletionDenseReflector NameCert obligations)**.** *For every accepted* CauchyCompletionDenseReflector$^{\uparrow}$ *carrier $D = (S, I, A, W, T, Q, R, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionDen}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source completion schedule, dense-image row, adjunction row, finite windows, dyadic tolerance ledger, regular readback, real seal, transport, replay, provenance, and local naming;*

2. *dense-source exposure, requiring every completion-facing read to expose S and then I before the real-seal row R can be consumed;*

3. *reflector exactness, requiring the dense image I and source schedule S to be replayed through the same* BHist, hsame, *and* Cont *rows before any completion consumer is served;*

4. *universal-row handoff, requiring every extension or uniqueness consumer to pass through the adjunction row A after the dense image and before the terminal real seal;*

5. *classifier transport, requiring all accepted comparisons to use only the displayed componentwise transport H, replay C, provenance P, and local name N;*

6. *ledger refusal, excluding quotient completion representatives, selected limits, countable choice, host equality of completed objects or real names, and ambient completeness.*

No dense-reflector obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3786.1. The accepted packet exposes exactly the rows $S, I, A, W, T, Q, R, H, C, P, N$ and no detached completion object.

Dense-source exposure is read in order. The source completion schedule S supplies the finite completion transformer route, and the dense image row I places the displayed source inside the completion seal before any real-facing read reaches R.

Reflector exactness and universal-row handoff then use the same route. The rows H and C replay the schedule and dense-image coordinates componentwise, while A is the only adjunction row available to extension and uniqueness consumers.

Finally inspect the provenance and local naming rows. The row P names only the completion monad, dense embedding, adjunction, StreamName, RegSeqRat, DyadicRatCore, Real, and structural BEDC rows, and N packages exactly that surface. Hence the listed refusals cannot be exported by an accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ packet is generated from a CauchyCompletionMonad source schedule, a MetricCompletionDenseEmbedding dense-image row, a CauchyCompletionAdjunction universal row, finite StreamName windows, DyadicRatCore tolerance accounting, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3786.1 and Definition 3786.2 seed the finite dense-reflector surface connecting completion monad, dense embedding, adjunction, and the terminal StreamName-RegSeqRat-DyadicRatCore-Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of completed objects or real names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exposure, reflector exactness, universal-row handoff, classifier transport, and ledger refusal for BEDC.Derived.CauchyCompletionDenseReflectorUp.

Chapter 3787

A Concrete Naming Certificate for $\mathsf{LocallyCompactMetric}^{\uparrow}$

Following Bourbaki and Kelley metric topology together with Bishop-style constructive metric practice, the $\mathsf{LocallyCompactMetric}^{\uparrow}$ packet records local compact neighbourhood data as finite BEDC rows around a displayed metric point. It sits over the $\mathsf{Met}^{\uparrow}$ distance surface of chapter 57, the $\mathsf{CompactMetric}^{\uparrow}$ compact-neighbourhood witness of chapter 129, the $\mathsf{CompleteMetric}^{\uparrow}$ limit rows of chapter 128, the $\mathsf{TotallyBounded}^{\uparrow}$ finite-net rows of chapter 127, and the positive $\mathsf{Real}^{\uparrow}$ radius seal of chapter 38. The chapter names a local compactness certificate for later proper-metric and compactification consumers; it does not assert arbitrary open-cover compactness, choice of neighbourhood bases, quotient completion equality, or ambient local compactness outside the displayed packet.

Definition 3787.1 (Locally compact metric carrier)**.** *A* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, x, U, K, T, C_m, r, H, D, P, N)$$

with the following displayed rows:

1. *X is a* $\mathsf{Met}^{\uparrow}$ *carrier row whose point histories and distance witnesses are read from chapter 57.*
2. *x is the inspected point history carried by X.*
3. *U is a local neighbourhood row around x, displayed only by metric-ball membership and finite route data.*
4. *K is a* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the neighbourhood closure packet.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net component read by K.*
6. *C_m is the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit-witness component read by K.*
7. *r is a positive* $\mathsf{Real}^{\uparrow}$ *radius seal for the local ball.*
8. *H records componentwise* hsame *transport for X, x, U, K, T, C_m, r.*
9. *D records displayed* Cont *replay from a local query through the metric carrier, neighbourhood row, compact witness, finite-net row, completion row, radius seal, and transport row.*
10. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, a selected neighbourhood basis, global compactness of X, quotient completion equality, countable choice, or a host topological space.

Lean4 (definition) `BEDC.Derived.LocallyCompactMetricUp.LocallyCompactMetricUp`
Lean4 ✓ `BEDC.Derived.LocallyCompactMetricUp.LocallyCompactMetricTasteGate single carrier alignment`

Definition 3787.2 (Locally compact metric classifier)**.** *Two accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carriers*

$$L = (X, x, U, K, T, C_m, r, H, D, P, N) \quad \textit{and} \quad L' = (X', x', U', K', T', C'_m, r', H', D', P', N')$$

classify together exactly when the metric carrier rows, inspected point rows, local neighbourhood rows, compact-metric witnesses, totally-bounded rows, complete-metric rows, Real radius seals, transport rows, continuation replays, provenance rows, and local naming rows classify componentwise by the sibling classifiers and the displayed hsame *transports.*

Theorem 3787.3 (Locally compact metric NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier $L = (X, x, U, K, T, C_m, r, H, D, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$ *surface consists exactly of the metric carrier row X, the inspected point row x, the neighbourhood row U, the compact-metric witness K, the totally-bounded finite-net row T, the complete-metric limit row C_m, the positive Real radius seal r, componentwise transport H, displayed replay D, provenance P, and local naming row N.*

Proof. Project the carrier of Definition 3787.1. The projection exposes the metric row, inspected point, local neighbourhood, compact witness, finite-net witness, completion witness, radius seal, transport, replay, provenance, and naming row.

The compact witness K is read only through the sibling $\mathsf{CompactMetric}^{\uparrow}$ surface, whose own carrier consumes $\mathsf{TotallyBounded}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ rows. Thus the local compactness packet records compact neighbourhood evidence without adding an open-cover coordinate.

The radius row r is read through the Real seal surface and the route D replays all displayed dependencies. Since the carrier has no coordinate for selected bases, global compactness, quotient completion equality, countable choice, or an ambient host topology, the listed rows exhaust the NameCert obligations. □

Theorem 3787.4 (Locally compact metric neighbourhood non-escape)**.** *Every consumer of an accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *packet that preserves the classifier of Definition 3787.2 must preserve the same metric carrier, inspected point, neighbourhood row, compact-metric witness, total-boundedness row, completeness row, Real radius seal, transport, replay, provenance, and local naming row. In particular, it cannot replace the displayed local neighbourhood by an unrecorded open cover, a choice-selected compact set, or a host topology.*

Proof. Use the classifier of Definition 3787.2. It compares the accepted packets componentwise and has no residual coordinate where a different neighbourhood, compact witness, finite-net row, completion row, or radius seal could be hidden.

The continuation row preserves the same route through the metric, compact, total-boundedness, completeness, and Real-radius siblings. Therefore a consumer that keeps the classifier fixed must read the same finite packet and cannot escape to an unrecorded open cover, selected compact subset, or host topology. □

Theorem 3787.5 (Locally compact metric sibling dependency route)**.** *Every accepted* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *carrier routes its local compactness certificate through chapter 57, chapter 129, chapter 128, chapter 127, and chapter 38. The* $\mathsf{Met}^{\uparrow}$ *row supplies the point and distance surface; the* $\mathsf{CompactMetric}^{\uparrow}$ *row supplies the compact-neighbourhood witness; the* $\mathsf{TotallyBounded}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows supply the finite-net and limit components read by compact metric data; and the* $\mathsf{Real}^{\uparrow}$ *row supplies the positive radius seal.*

Proof. Read the rows of L in the order displayed by Definition 3787.1. The metric carrier X and point x must be available before a neighbourhood row U can be read as a metric-local object.

The compact witness K then exposes its two constructive compact-metric dependencies: total boundedness through T and complete-metric convergence through C_m. The radius seal r is a Real-facing row, so it is consumed through the terminal Real chapter rather than through a host real number.

Finally H, D, P, N transport, replay, source, and name the same dependency chain. Thus the local compactness packet is a finite L10 neighbour of the metric, compact metric, complete metric, totally bounded, and Real chapters, not a stand-alone topological assertion. □

Falsifiable prediction: Every $\mathsf{LocallyCompactMetric}^{\uparrow}$ read factors through one Metric carrier, one inspected point, one neighbourhood row, one CompactMetric witness, one TotallyBounded finite-net row, one CompleteMetric limit row, one Real radius seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: metric, compactmetric, completemetric, totallybounded, real: $\mathsf{LocallyCompactMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a point-local neighbourhood row, a compact-metric witness for that neighbourhood, the compact witness's total-boundedness and completeness components, and a positive Real radius seal in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactMetric}^{\uparrow}$ packet is generated from a Metric carrier, inspected point, local neighbourhood row, CompactMetric witness, TotallyBounded finite-net row, CompleteMetric limit row, positive Real radius seal, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3787.1, Definition 3787.2, Definition 3787.3, Definition 3787.4, and Definition 3787.5 seed the finite local-compactness surface over metric, compact-metric, complete-metric, total-boundedness, and Real-radius rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not assert arbitrary open-cover compactness, a selected neighbourhood basis, global compactness of the carrier, quotient completion equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, NameCert obligations, neighbourhood non-escape, and sibling route coverage for BEDC.Derived.LocallyCompactMetricUp.

Chapter 3788

A Concrete Naming Certificate for $\mathsf{LocallyCompactPolish}^{\uparrow}$

Following Bourbaki, Kelley, and Alexandrov–Urysohn, the $\mathsf{LocallyCompactPolish}^{\uparrow}$ packet records the finite BEDC surface where local compact neighbourhood data is read together with a Polish complete-separable metric presentation. It sits at the joint frontier of the local compact metric packet of chapter 3787, the Polish readback of chapter 1490, the proper closed-ball route of **??**, the sigma-compact exhaustion of chapter 2369, and the complete-separable metric presentation of chapter 1735. The certificate is finite data: neighbourhood rows, complete-separable readback, properness rows, sigma-compact exhaustion windows, StreamName and RegSeqRat observations, transport, replay, provenance, and local naming.

Definition 3788.1 (Locally compact Polish carrier)**.** *A* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, K, C, S, E, W, T, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the underlying metric-object row shared by the locally compact, Polish, proper, sigma-compact, and complete-separable presentations.*
2. *M is the* $\mathsf{LocallyCompactMetric}^{\uparrow}$ *neighbourhood row. It records, for every displayed point-window, a compact neighbourhood candidate, an interior witness, and the metric inspection data by which the candidate is consumed.*
3. *K is the local compactness compact-row ledger. It binds compact neighbourhood reads to the compact witnesses supplied through M and refuses any neighbourhood not named in the packet.*
4. *C is the* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *readback row. It records the complete metric presentation, the countable dense probe family, and the finite comparison windows through which completeness and separability are inspected.*
5. *S is the* $\mathsf{PolishSpace}^{\uparrow}$ *row obtained from the same complete-separable readback; it carries no independent host topology coordinate.*
6. *E is the* $\mathsf{ProperMetric}^{\uparrow}$ *closed-ball compactness row. It is read only after M, K, C, S have fixed the metric and neighbourhood surface.*
7. *W is the* $\mathsf{SigmaCompact}^{\uparrow}$ *exhaustion row. Its compact exhaustion windows are generated from the displayed local compactness and properness rows, not from a selected external exhaustion.*

8. *T is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row for points, compact neighbourhood candidates, complete-separable probes, closed balls, exhaustion windows, and endpoint-facing reads.*
9. *R is the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row for the same metric probes and exhaustion windows.*
10. *H records componentwise* hsame *transport for $X, M, K, C, S, E, W, T, R, Q, P, N$.*
11. *Q records displayed* Cont *replay of the route*
$$M, K \longmapsto C, S \longmapsto E, W \longmapsto T, R.$$
12. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactPolish}^{\uparrow}$, $\mathsf{LocallyCompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{SigmaCompact}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, K, C, S, E, W, T, R, H, Q, P, N$. It has no coordinate for arbitrary open-cover compactness, a selected exhaustion, host topological equality, quotient metric equality, countable choice, or a Polish presentation detached from the displayed complete-separable row.

Theorem 3788.2 (Locally compact Polish NameCert obligations)**.** *For every accepted* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier*
$$L = (X, M, K, C, S, E, W, T, R, H, Q, P, N),$$
the local $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the thirteen displayed rows;*
2. *local compact neighbourhood exactness, requiring M and K to bind every compact neighbourhood read to a displayed point-window and compact witness;*
3. *complete-separable readback compatibility, requiring C to carry the complete metric presentation and countable dense probe family used by the packet;*
4. *Polish readback compatibility, requiring S to be consumed through C and not through an independent host topology;*
5. *proper closed-ball compactness compatibility, requiring E to inspect the metric and compact neighbourhood rows already displayed;*
6. *sigma-compact exhaustion compatibility, requiring W to use only the compact windows generated from the local compactness and properness route;*
7. *StreamName and RegSeqRat finite-window compatibility, requiring T and R to observe the same metric probes, compact neighbourhoods, closed balls, and exhaustion windows;*
8. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, Q, P, N.*

No locally compact Polish obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocallyCompactPolishUp.LocallyCompactPolishNameCertObligations`

Proof. Project the accepted packet $L = (X, M, K, C, S, E, W, T, R, H, Q, P, N)$. The metric object, local compactness, compact-neighbourhood ledger, complete-separable readback, Polish row,

properness row, sigma-compact exhaustion, finite observations, and structural rows are precisely the displayed carrier coordinates.

The neighbourhood route is fixed first. Rows M and K name the compact neighbourhood candidates and bind every accepted local compactness read to the corresponding point-window and compact witness.

Next the complete-separable route is read through C, and the Polish row S is downstream of that readback. Since S has no separate host-topology coordinate, every Polish consumer receives the same complete metric and dense-probe data.

The rows E and W expose proper closed-ball compactness and sigma-compact exhaustion only after the metric, neighbourhood, and complete-separable rows have been displayed. The observation rows T and R then inspect the same probes and windows through finite StreamName and RegSeqRat data.

Finally H, Q, P, N transport, replay, source, and name the route without adding new coordinates. The excluded objects do not occur in the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 3788.3 (Locally compact Polish proper-sigma route)**.** *Every properness, sigma-compactness, completion, or compact-neighbourhood consumer of an accepted* $\mathsf{LocallyCompactPolish}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{LocallyCompactMetric}^{\uparrow}, \mathsf{CompleteSeparableMetric}^{\uparrow}, \mathsf{PolishSpace}^{\uparrow} \longmapsto M, K, C, S \longmapsto \mathsf{ProperMetric}^{\uparrow}, \mathsf{SigmaCompact}^{\uparrow}$$

The route exposes local compact neighbourhood rows and Polish complete-separable readback before proper closed-ball compactness or sigma-compact exhaustion is public. It exports no arbitrary open-cover compactness, selected exhaustion, host topological equality, quotient metric equality, countable-choice selector, or Polish presentation detached from the displayed complete-separable packet.

Proof. Start with the rows M, K, C, S. The first pair carries the local compact neighbourhood data, while the second pair carries the complete-separable presentation and the Polish readback consumed by the packet.

The properness and sigma-compactness rows are then downstream. Row E reads closed-ball compactness only through the displayed metric and local compactness surface, and row W records exhaustion windows generated from the same source.

The observation and structural rows T, R, H, Q, P, N preserve that route for consumers. Because the carrier has no coordinate for open-cover compactness, selected exhaustion, host equality, quotient equality, countable choice, or detached Polish data, every consumer remains inside the displayed proper-sigma route. □

Falsifiable prediction: An accepted $\mathsf{LocallyCompactPolish}^{\uparrow}$ read is refuted if a properness, sigma-compactness, completion, or compact-neighbourhood consumer exports a closed-ball compactness row or exhaustion row without displaying LocallyCompactMetric neighbourhood data, Polish complete-separable readback, StreamName and RegSeqRat finite windows, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: locally-compact-metric, polishspace, propermetric, sigma-compact, complete-separable-metric, streamname, regseqrat: none alone is bijective with $\mathsf{LocallyCompactPolish}^{\uparrow}$ because the packet jointly carries local compact neighbourhood rows, complete-separable Polish readback, proper closed-ball compactness, sigma-compact exhaustion, finite observations, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyCompactPolish}^{\uparrow}$ packet is generated from locally compact metric neighbourhood rows, compact-neighbourhood ledgers, complete-separable metric readback, PolishSpaceUp readback, ProperMetricUp closed-ball compactness, SigmaCompactUp exhaustion windows, StreamNameUp and RegSeqRatUp finite observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3788.1, Definition 3788.2, and Definition 3788.3 seed the locally compact Polish certificate surface from compact neighbourhood rows and complete-separable readback through proper closed-ball compactness and sigma-compact exhaustion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyCompactPolish}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected exhaustions, host topological equality, quotient metric equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked local compact neighbourhood exactness, complete-separable and Polish readback compatibility, proper closed-ball compatibility, sigma-compact exhaustion compatibility, StreamName and RegSeqRat finite-window compatibility, transport, replay, provenance, and local naming for the displayed packet.

Chapter 3789

A Concrete Naming Certificate for $\mathsf{SequentialRealCompactness}^{\uparrow}$

Following Bolzano and Weierstrass, the $\mathsf{SequentialRealCompactness}^{\uparrow}$ packet records the finite route by which a bounded real sequence exposes a regular cluster subsequence through displayed interval-tree data. It sits after the subsequence extraction certificate of chapter 1254, the bounded source surface of chapter 1415, the interval-net carrier of chapter 1250, the regular rational readback of chapter 301, the finite window interface of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a sequential compactness interface for bounded real sequences, not an open-cover compactness theorem, a host completeness principle, quotient equality of real names, or a choice-selected arbitrary subsequence.

Definition 3789.1 (Sequential real compactness carrier)**.** *A* $\mathsf{SequentialRealCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, I, E, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source sequence window schedule exposed through* $\mathsf{StreamName}^{\uparrow}$*.*
2. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row naming the bounded real sequence interface consumed by the extraction route.*
3. *I is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *interval-tree row carrying dyadic retained cells and finite mesh comparisons.*
4. *E is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *extraction row, including the monotone-cofinal subsequence schedule and retained finite windows.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the extracted windows and dyadic tolerances.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal reached only after S, B, I, E, Q have been displayed.*
7. *H records componentwise* hsame *transport for the source schedule, bounded source row, interval tree, extraction row, regular readback, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from bounded sequence inspection through interval-tree retention, Bolzano–Weierstrass extraction, RegSeqRat readback, and RealUp sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentialRealCompactness}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$ *row.*

The carrier stores no open-cover compactness object, no arbitrary subsequence selector, no quotient equality of real names, no host supremum, no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no branch outside the displayed interval-tree extraction route.*

Lean4 ✓ `BEDC.Derived.SequentialRealCompactnessUp.SequentialRealCompactnessTasteGate single carrier alignment`

Theorem 3789.2 (Sequential real compactness NameCert obligations)**.** *For every accepted* SequentialRealCompact *carrier $K = (S, B, I, E, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, bounded-source admission through B, finite interval-tree retention through I, Bolzano–Weierstrass extraction through E, finite stream-window and dyadic-tolerance readback through S and Q, terminal real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3789.1. The bounded sequence source is available only through S and B, while the interval-tree row I supplies the retained dyadic cells before extraction is read.

Use the bounded-source handoff of chapter 1415 and the interval-net surface of chapter 1250. These rows expose finite sequence windows, regular readback, real-name seals, and interval containment without adding an open-cover compactness object or a host completeness theorem.

Read the extraction row through chapter 1254. The Bolzano–Weierstrass packet supplies the monotone-cofinal subsequence schedule and finite retained windows, which are then read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

The structural rows H, C, P, N transport, replay, source, and name only this finite route. Since the carrier has no coordinate for arbitrary subsequence choice, quotient real equality, host supremum, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3789.3 (Sequential real compactness Bolzano–Weierstrass handoff)**.** *For every accepted* $\mathsf{SequentialRealCompactness}^{\uparrow}$ *carrier $K = (S, B, I, E, Q, R, H, C, P, N)$, the terminal compactness-facing read factors through the finite route*

$$\mathsf{BoundedRealSequence}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalNet}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatC}$$

The handoff composes the bounded-source row with Bolzano–Weierstrass finite interval-tree extraction and the RegSeqRat/Real seal readback. It exports a displayed sequential cluster route, not open-cover compactness, choice-like subsequence selection, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 3789.2. It places the bounded source row B, interval-tree row I, extraction row E, readback row Q, and terminal seal R in a single finite carrier route.

The bounded source is read through chapter 1415. Its finite windows and interval bound supply the source surface consumed by the Bolzano–Weierstrass certificate, while chapter 1250 supplies the dyadic retained-cell discipline.

Now compose with chapter 1254. The extraction row reads only the finite interval-tree schedule and then hands the selected windows to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ cluster seal can be consumed.

Transport and replay through H and C, then source and name the same route through P and N. The packet therefore records a finite sequential compactness handoff and refuses the excluded open-cover, choice, quotient, and ambient-completeness exits. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialRealCompactness}^{\uparrow}$ packet is generated from finite StreamName windows, a bounded real sequence source row, a closed bounded interval-net tree, a Bolzano–Weierstrass extraction row, a RegSeqRat readback row, a RealUp cluster seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3789.1, Definition 3789.2, and Definition 3789.3 seed the finite bounded-sequence compactness route through interval-tree extraction and regular real readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialRealCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary subsequence choice, quotient real equality, host supremum, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source handoff, interval-tree extraction, RegSeqRat readback, Real seal routing, and NameCert field faithfulness for BEDC.Derived.SequentialRealCompactnessUp.

Chapter 3790

A Concrete Naming Certificate for $\mathsf{LawlikeSequence}^{\uparrow}$

Following Brouwer and Bishop-style constructive sequence analysis, the $\mathsf{LawlikeSequence}^{\uparrow}$ packet records a sequence source whose future values are governed by a displayed finite rule rather than by a host infinite function. It sits between the finite-window interface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the unfolding prefix surface of chapter 1482, and the finite observation boundary of chapter 1495. The packet supplies a lawful finite-rule stream source for the L10 Real and Completion route while keeping quotient streams, countable choice, host equality, and selected Real limits outside the exported surface.

Definition 3790.1 (Lawlike sequence carrier)**.** *A* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, W, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite rule row: a program, recurrence, primitive recursion table, or other finite calculation ledger whose admitted executions produce the inspected sequence values.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request-window row listing the finite prefixes and positions currently asked of the rule.*
3. *O is the observation row produced by executing R on the requests in W; it records only the displayed output values and their finite calculation traces.*
4. *H records componentwise* hsame *transport for R, W, O and for the structural rows that preserve the same displayed rule executions.*
5. *C records displayed* Cont *replay from a prefix request in W through the finite rule row R to the observation row O, and then to any downstream regular-rational or Real-facing read.*
6. *P is the* Pkg *provenance over* $\mathsf{LawlikeSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{LawlessSequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through R, W, O, H, C, P, N. It has no coordinate for a host total function $\mathsf{Nat}^{\uparrow} \to \mathsf{Nat}^{\uparrow}$*, quotient stream equality, a countable-choice selector, host sequence equality, a selected Real limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LawlikeSequenceUp.TasteGate.LawlikeSequenceTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.LawlikeSequenceUp.LawlikeSequenceUp`

Definition 3790.2 (Lawlike sequence classifier)**.** *The local* $\mathsf{LawlikeSequence}^{\uparrow}$ *classifier accepts two packets*

$$L = (R, W, O, H, C, P, N) \quad \textit{and} \quad L' = (R', W', O', H', C', P', N')$$

as the same lawlike sequence read precisely when the displayed rule rows are related by the local hsame *transport, the request windows list the same finite positions, the observation rows agree on every requested position by replaying the displayed rule executions, and the continuation, provenance, and local NameCert rows preserve that same finite route. Agreement is therefore a finite-rule replay judgment over the current window, not equality of completed streams or equality of host functions.*

Theorem 3790.3 (Lawlike sequence NameCert obligations)**.** *For every accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier* $L = (R, W, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite rule row, request-window exposure through* W*, observation replay through* O*, classifier stability under componentwise* hsame *transport, displayed* Cont *replay of the same finite rule executions, package provenance through* P*, and local naming through* N*. No obligation row may be discharged by a host infinite function, a countable-choice selector, quotient stream equality, host sequence equality, a selected Real limit, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.LawlikeSequenceUp.LawlikeSequenceNameCertObligations`

Proof. Project the carrier of Definition 3790.1. The only sequence source displayed to a consumer is the finite rule row R, and the only inspected arguments are the finite requests listed in W.

For each request in W, replay the calculation trace stored in O. The replay is finite because O stores only the values produced by executing the displayed rule on the displayed window, not an ambient total function.

Transport stability is componentwise. The row H compares rule rows, request windows, and observation rows by hsame, while C replays the same dependency order from requests through rule execution to downstream reads.

Finally repack the route with P and N. Since none of the carrier coordinates stores a completed stream, a choice selector, quotient equality, host sequence equality, selected limit, or ambient completeness principle, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3790.4 (Lawlike sequence StreamName handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier receives the finite route*

$$R \longmapsto W \longmapsto O$$

before any regular-rational readback or terminal Real seal can be consumed. The route presents lawlike sequence data as a displayed finite-rule stream source: $\mathsf{StreamName}^{\uparrow}$ *reads the request window,* $\mathsf{RegSeqRat}^{\uparrow}$ *may read only the observations already produced on that window, and* $\mathsf{Real}^{\uparrow}$ *may be reached only after that regular-rational handoff has been replayed through* H, C, P, N*. The handoff exports no host infinite function, countable-choice selector, quotient stream equality, host equality, selected Real limit, or ambient Real completeness.*

Proof. Read the accepted carrier through Definition 3790.1. The first available sequence-facing row is the rule row R, and every downstream request must pass through the displayed finite window W.

The observation row O is then obtained by replaying the finite executions named by R on the requests in W. A $\mathsf{StreamName}^{\uparrow}$ consumer receives exactly that windowed read; it receives no tail values beyond the displayed requests.

If a regular-rational or Real-facing consumer is present, its read is downstream of the same observation row. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff can consume only the displayed observations, and the $\mathsf{Real}^{\uparrow}$ seal can be reached only after that handoff is replayed through H, C, P, N.

Thus the lawful source is the finite-rule packet itself. The absent coordinates are exactly the data needed for a host infinite function, countable-choice selector, quotient stream equality, host equality, selected Real limit, or ambient Real completeness claim, so none of those objects can be exported by the handoff. □

Theorem 3790.5 (Lawlike sequence regular-window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{LawlikeSequence}^{\uparrow}$ *carrier* $L = (R, W, O, H, C, P, N)$ *must read the finite rule row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *request-window row* W*, the observation replay row* O*, and the componentwise transport and replay rows* H, C *before treating the displayed observations as a regular rational sequence. The consumer obtains no host infinite function, quotient stream equality, countable-choice selector, selected* $\mathsf{Real}^{\uparrow}$ *limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

If the read continues to a $\mathsf{Real}^{\uparrow}$ *endpoint, it does so only after the same regular-window route has passed through the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff and the terminal* $\mathsf{Real}^{\uparrow}$ *seal recorded by* P, N*.*

Proof. Project the carrier through Definition 3790.1. The rule row R and the request-window row W determine exactly which finite calculations may be inspected.

Apply Definition 3790.4. It forces every stream-facing read to pass through

$$R \longmapsto W \longmapsto O$$

before any regular-rational consumer can see observations.

Now route the regularity readback through $\mathsf{RegSeqRat}^{\uparrow}$. Since the observations are the replayed outputs stored in O, regularity can be checked only against the displayed window and the componentwise H, C rows.

Finally, if a Real-facing endpoint is requested, the route is sealed through the terminal $\mathsf{Real}^{\uparrow}$ boundary named by P, N. The absent carrier coordinates are precisely the data needed for a host infinite function, quotient stream equality, countable choice, selected Real limit, or ambient completeness, so none of those objects is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LawlikeSequence}^{\uparrow}$ packet is generated from a finite rule row, StreamName request windows, observation rows produced by the rule on those windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3790.1, Definition 3790.2, Definition 3790.3, and Definition 3790.4 seed the finite-rule stream source feeding StreamName, RegSeqRat, and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawlikeSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, countable choice, host sequence equality, host function equality, selected Real limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-rule classifier stability, StreamName request-window exposure, RegSeqRat handoff exactness, Real seal nonescape, and refusal of host equality and choice for BEDC.Derived.LawlikeSequenceUp.

Chapter 3791

A Concrete Naming Certificate for MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$

The MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ packet realizes a finite conditional-boundary row from the MetaCIC confluence audit map of chapter 9378. It sits beside the residual-substitution boundary of chapter 2181, the critical-pair boundary of chapter 2212, the frontier critical-pair packet of chapter 2382, the local diamond witness of chapter 3595, and the confluence audit witness of chapter 745. The carrier isolates the substitution-stability gate that must remain visible before any conditional parallel-diamond route is read as a closed confluence theorem.

Definition 3791.1 (MetaCIC confluence substitution-boundary carrier)**.** *A* MetaCICConfluenceSubstitutionBounda *carrier is a finite* BHist *packet*

$$S = (T, R, U, D, J, O, G, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite typed-term window containing the source term, the two one-step parallel reducts, and the candidate common reduct.*
2. *R is the redex-frontier row, exposing the local redex shapes consumed by the confluence audit map.*
3. *U is the substitution-boundary row, recording the closed substitution support, binder-depth ledger, and free-variable non-escape checks needed by the branch.*
4. *D is the conditional parallel-diamond handoff row. It may be consumed only after T, R, U have been displayed.*
5. *J is the local join witness row, recording the candidate common reduct as a finite replay path rather than as a global Church–Rosser theorem.*
6. *O is the obstruction row naming the absent endpoints: unrestricted substitution compatibility, arbitrary subject reduction, and global closed confluence.*
7. *G is the* TasteGate *readiness row, reserving the later* Nontrivial *and* FieldFaithful *instance obligations for this finite carrier.*
8. *H records componentwise* hsame *transport for the typed-term window, redex frontier, substitution boundary, conditional-diamond handoff, local join, obstruction, TasteGate readiness, replay, provenance, and local name rows.*

9. *C records displayed* Cont *replay from a confluence-boundary query through $T, R, U, D, J, O, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$*, the confluence audit map, residual-substitution boundary rows, critical-pair rows, local-diamond rows,* BHist*,* hsame*,* Cont*,* Pkg*,* TasteGate*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for global confluence, unrestricted substitution compatibility, subject reduction, kernel consistency, evaluator equality, or hidden conversion.

Theorem 3791.2 (MetaCIC confluence substitution-boundary obligations)**.** *For every accepted* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$ *carrier $S = (T, R, U, D, J, O, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, redex-frontier exposure through R, substitution-boundary exposure through U, conditional parallel-diamond handoff through D, local join replay through J, obstruction retention through O, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The confluence-facing route is

$$T, R, U \longmapsto D, J, O \longmapsto G, H, C, P, N.$$

It is conditional at U and D: the packet records where substitution stability is available, not a closed confluence theorem.

Proof. Project the carrier of Definition 3791.1. The term-facing rows are T, R, U, D, J, O, and the structural rows are G, H, C, P, N.

The conditional branch begins with T, R, U. The typed-term window supplies the finite peak, the redex-frontier row supplies the local critical shape, and the substitution-boundary row supplies exactly the closed substitution support that must be checked before the diamond handoff is read.

The rows D, J, O then separate positive and negative content. The handoff and join rows record a local candidate route, while the obstruction row keeps unrestricted substitution compatibility, subject reduction, and global confluence outside the packet.

Finally G, H, C, P, N prepare the later TasteGate instance surface, transport the same rows, replay the same query, source it, and name it. Since no hidden conversion, evaluator equality, global confluence theorem, or kernel-consistency coordinate occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3791.3 (MetaCIC confluence substitution-boundary non-escape)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$ *carrier must factor through*

$$T, R, U, D, J, O, G, H, C, P, N.$$

If the substitution-boundary row U or the conditional handoff row D is absent, the packet no longer exports a confluence-boundary read. The remaining rows may still record typed terms, redex shapes, obstruction names, local join candidates, transport, replay, provenance, TasteGate readiness, and local naming, but they cannot supply global Church–Rosser, unrestricted substitution compatibility, arbitrary subject reduction, or kernel consistency.

Proof. Apply the obligation surface of Definition 3791.2. A confluence-audit read reaches the local join only through the typed window, redex frontier, substitution boundary, and conditional handoff.

If U is absent, the substitution-stability premise is not displayed. If D is absent, the conditional parallel-diamond handoff is not displayed. In either case the finite packet no longer contains the route needed to treat J as a confluence-boundary read.

The rows O, G, H, C, P, N do not repair the missing premise. They retain obstruction names, reserve the later TasteGate obligations, and preserve the visible packet, but none is a coordinate for global Church–Rosser, unrestricted substitution compatibility, subject reduction, or consistency. □

Theorem 3791.4 (MetaCIC confluence substitution-boundary TasteGate readiness)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ *at the displayed finite carrier surface*

$$T, R, U, D, J, O, G, H, C, P, N.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the typed-term window, redex-frontier row, substitution-boundary row, conditional handoff, local join row, obstruction row, transport, replay, provenance, and local NameCert data.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceSubstitutionBoundaryUp.MetaCICConfluenceSubstitutionBoundaryTaste`
`single carrier alignment`

Proof. The readiness row G is part of the carrier in Definition 3791.1. It is deliberately tied to the whole tuple rather than to the obstruction row alone.

Nontriviality must therefore exhibit an accepted packet with all finite rows present. FieldFaithfulness must show that deleting any of $T, R, U, D, J, O, H, C, P, N$ changes the readable carrier surface. The theorem records those Lean obligations at paper level while leaving their proofs to the later formalization namespace. □

Falsifiable prediction: A MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ confluence-audit read factors through displayed typed-term windows, redex-frontier rows, substitution-boundary rows, conditional parallel-diamond handoffs, local join witnesses, obstruction rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, metacic-frontier-critical-pair, metacic-local-diamond-witness, metacic-confluence-audit-witness: MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a typed peak, redex frontier, substitution-boundary row, conditional handoff, local join candidate, obstruction ledger, TasteGate readiness, transport, replay, provenance, and local naming as one boundary packet.

Closure status for NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 3791.1, Definition 3791.2, Definition 3791.3, and Definition 3791.4 seed the finite substitution-sensitive confluence-boundary packet for MetaCIC audit consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global Church–Rosser, unrestricted substitution compatibility, arbitrary subject reduction, kernel consistency, evaluator equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, redex-frontier exposure, substitution-boundary coverage, conditional handoff, local join replay, obstruction retention, TasteGate readiness, transport, replay, provenance, and non-escape for BEDC.Derived.MetaCICConfluenceSubstitutionBoundaryUp.

Constructive story: A MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ packet is generated from a typed-term peak window, redex-frontier rows, a substitution-boundary row, a conditional parallel-diamond handoff, a local join witness, an obstruction ledger, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3792

A Concrete Naming Certificate for $\mathsf{CauchyCompletionComonad}^{\uparrow}$

Following Mac Lane on comonads and Bishop–Bridges on constructive Cauchy completion, the $\mathsf{CauchyCompletionComonad}^{\uparrow}$ packet records the finite completion-read surface obtained by reading already displayed completion data back through counit and comparison rows. It is paired with the monad packet of chapter 378, the Kleisli consumer boundary of chapter 1845, the counit packet of chapter 1539, the comparison packet of chapter 1954, and the triangle-identity packet of chapter 1795. The analytic route is the same finite L10 exit through chapter 126, chapter 301, chapter 291, and chapter 38; the chapter names this dual read as finite NameCert data rather than as a quotient completion object.

Definition 3792.1 (CauchyCompletionComonad carrier)**.** *A* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *route whose source, unit, bind, finite-window, readback, and terminal-seal rows are being read in the opposite direction.*
2. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *consumer row, displaying how a completion-facing request reaches the same finite source and seal rows.*
3. *E is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *return row from displayed completion data to the real-facing boundary.*
4. *Q is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row comparing the monad-facing and counit-facing reads over the same finite completion route.*
5. *T is the* $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$ *row identifying the two displayed unit-counit composites with the direct completion read.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected before any rational readback or terminal seal is exposed.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row consumed by the completion read.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation ledger attached to the same finite windows.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, R, D have been displayed.*

10. *H records componentwise* hsame *transport for $M, K, E, Q, T, S, R, D, A$ and the structural rows.*
11. *C records displayed* Cont *replay for consumers that inspect the comultiplication side, the counit side, the comparison side, or the terminal real-facing route.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionComonad}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{CauchyCompletionTriangleIdentity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected Cauchy representatives, host equality of completed objects, countable choice, ambient completeness, $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-axis evidence.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionComonadUp.CauchyCompletionComonadTasteGate single carrier alignment`

Theorem 3792.2 (CauchyCompletionComonad NameCert obligations)**.** *For every accepted* CauchyCompletionCom *carrier $G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the monad, Kleisli, counit, comparison, triangle, finite-window, regular-readback, dyadic-ledger, terminal-seal, transport, replay, provenance, and local naming rows;*
2. *comultiplication exposure, requiring every doubled completion-read request to pass through M, K, Q, T before a terminal seal can be inspected;*
3. *counit return, requiring every already displayed completion read to pass through E before it reaches S, R, D, A;*
4. *finite triangle compatibility, requiring the comultiplication-counit composites to reach the same displayed S, R, D, A route as the direct completion read;*
5. *classifier stability, requiring transported packets with the same displayed rows to be compared only by the carried* hsame *and* Cont *data;*
6. *non-escape, requiring P, N to source and name no quotient completion, selected representative, countable-choice schedule, host equality, ambient completeness theorem,* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification credential, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3792.1. The tuple lists the monad, Kleisli, counit, comparison, triangle, finite-window, regular-readback, dyadic-ledger, terminal-seal, transport, replay, provenance, and local naming rows.

Comultiplication exposure is read from M, K, Q, T. The monad and Kleisli rows give the doubled completion consumer surface, the comparison row aligns it with the already displayed completion read, and the triangle row identifies the visible composites with the direct route.

Counit return is read from E. Once completion data have already been displayed, the packet forces their return through S, R, D, A before any real-facing consumer sees a terminal seal.

The structural obligations are componentwise. The rows H and C preserve and replay only the displayed packet, while P and N source and name that packet. Since no excluded coordinate occurs in the carrier, no stronger completion object can be exported. □

Theorem 3792.3 (CauchyCompletionComonad comultiplication and counit route)**.** *Let* $G = (M, K, E, Q, T, S, R, D, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier. Every consumer of the finite comultiplication-counit surface factors through the displayed route*

$$M, K \longmapsto Q, T \longmapsto E \longmapsto S, R, D \longmapsto A \longmapsto H, C, P, N.$$

The route exposes a completion read only after the monad-facing and Kleisli-facing rows have been compared with the counit-facing return and then passed through the finite $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. It exports no quotient completion, selected representative, host equality of completed objects, countable-choice selector, ambient completeness theorem, or bridge credential.*

Proof. Project the accepted carrier. The only doubled completion-read rows are M and K, so a comultiplication-facing request first reads the monad route and the Kleisli consumer boundary.

The middle comparison is forced by Q and T. The comparison row aligns the monad-facing read with the counit-facing return, and the triangle row identifies the two visible composites with the same direct completion route.

The counit row E then returns the displayed completion read to the terminal finite route. That route must pass through the finite window S, the regular readback R, the dyadic ledger D, and the real seal A.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no coordinate for quotient completions, selected representatives, host equality, countable choice, ambient completeness, or bridge evidence, the displayed route is the whole comonad surface. □

Theorem 3792.4 (CauchyCompletionComonad finite completion-read boundary)**.** *Every completion-facing,* $\mathsf{StreamName}^{\uparrow}$*-facing,* $\mathsf{RegSeqRat}^{\uparrow}$*-facing,* $\mathsf{DyadicRatCore}^{\uparrow}$*-facing, or* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{CauchyCompletionComonad}^{\uparrow}$ *carrier is generated by the rows of Definition 3792.1. In particular, the packet gives later completion algebra and reflector consumers a finite non-quotient read surface, not an ambient completed object.*

Proof. Read the completion-facing side first. The only completion control rows are M, K, E, Q, T, so every completion-facing consumer must factor through the monad route, Kleisli route, counit return, comparison row, and triangle row.

Read the analytic side next. The only finite observation and terminal seal rows are S, R, D, A, so every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal read through those rows.

The structural rows H, C, P, N preserve the same surface and add no new carrier coordinate. Therefore algebra and reflector consumers can reuse the finite read boundary, but they cannot obtain a quotient completion, selected limit, host completed object, ambient completeness theorem, or bridge credential from it. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCompletionComonad}^{\uparrow}$ packet is generated from CauchyCompletionMonad, CauchyCompletionKleisli, CauchyCompletionCounit, CauchyCompletionComparison, CauchyCompletionTriangleIdentity, StreamName, RegSeqRat, DyadicRatCore, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$, seed)

Scope closed: Definition 3792.1, Definition 3792.2, Definition 3792.3, and Definition 3792.4 seed the finite dual completion-read surface over the L10 Real and Completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionComonad}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected Cauchy representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comultiplication exposure, counit return, triangle compatibility, classifier stability, non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.CauchyCompletionComonadUp.

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ lattice is routed as a Cantor-to-Real diagonal boundary through chapter 1530, chapter 5879, chapter 301, and chapter 38.

Chapter 3793

A Concrete Naming Certificate for $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$

Following Cantor's diagonal argument, the $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet records the finite BEDC boundary that prevents a displayed countable $\mathsf{StreamName}^{\uparrow}$ enumeration from being read as exhaustive $\mathsf{Real}^{\uparrow}$ data. It sits over the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the dyadic bit ledger of chapter 291, the terminal real-name seal of chapter 38, and the comparison setoid boundary of chapter 1820. The packet names only the inspected enumeration rows, the diagonal bit-flip row, the regular readback route, transport, replay, provenance, and local naming data; it is not a host real-uncountability theorem, not a quotient-stream principle, and not a completed powerset of reals. The downstream Cantor diagonal real-uncountability packet of **??** consumes this finite non-surjection boundary as its Real-facing diagonal obstruction before adding CantorSpace and CantorSet rows.

The dependency direction is one-way at this lattice node: $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ consumes $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ to compare the diagonal witness with inspected rows, while CantorDiagonalRealUncountab consumes this chapter as the Real-facing obstruction before adding CantorSpace and CantorSet rows.

The diagonal-window side is read through chapter 4080. That sibling supplies the finite Cauchy diagonal-window selector used before the bit-flip row F and the terminal Real seal row S are consumed, so this chapter records the non-surjection witness as a finite selector-to-flip route rather than as only a kernel-to-Real route.

Definition 3793.1 (Real diagonal non-surjection carrier)**.** *A* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, W, B, F, R, S, H, C, P, N)$$

with the following displayed rows:

1. *E is the countable-enumeration row, displaying the finite slice of the proposed* $\mathsf{StreamName}^{\uparrow}$ *enumeration inspected by the packet.*
2. *W is the row-index schedule, listing the inspected diagonal rows and the finite windows used for each row.*
3. *B is the dyadic-bit observation row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any real-name seal is exposed.*
4. *F is the diagonal bit-flip row, recording for each inspected row the opposite dyadic bit at the matching diagonal window.*

5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed bit-flip data into regular rational real-name observations.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the diagonal witness, exposed only after E, W, B, F, R have been displayed.*
7. *H records componentwise* hsame *transport for E, W, B, F, R, S and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an enumeration request through*

$$E, W, B, F, R \longmapsto S \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *row.*

The carrier contains no host real equality test, quotient stream equality, completed powerset of real names, countable-choice schedule, selected global enumeration inverse, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge credential.*

Theorem 3793.2 (Real diagonal non-surjection NameCert obligations)**.** *For every accepted* RealDiagonalNonSurjec *carrier $D = (E, W, B, F, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, enumeration-slice exposure through E, finite diagonal-window accounting through W, dyadic-bit readback through B, bit-flip exactness through F, regular rational readback through R, terminal real-name sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.RealDiagonalNonSurjectionUp.RealDiagonalNonSurjectionNameCertObligations`

Proof. Project the carrier of Definition 3793.1. The analytic rows are the enumeration slice E, the finite diagonal schedule W, the dyadic observation row B, the flipped-bit row F, the regular rational readback R, and the terminal seal S.

The enumeration obligations are finite. A consumer sees only the rows named by W, and for each such row it sees only the dyadic window displayed in B before the corresponding flipped bit is recorded in F.

The real-name handoff is routed through R and then S. Thus the diagonal witness is read as regular rational data before it is sealed as $\mathsf{Real}^{\uparrow}$ data, and the Cauchy-equivalence boundary remains a displayed comparison surface rather than a quotient of streams.

Transport and replay preserve the same rows componentwise. Since the carrier has no coordinate for host real equality, quotient streams, completed powersets, countable choice, selected enumeration inverses, ambient real completeness, Lean verification, or bridge checking, the listed rows exhaust the local certificate. □

Theorem 3793.3 (Real diagonal non-surjection diagonal witness)**.** *Every accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet exports a displayed diagonal witness route*

$$\begin{gathered} \textit{inspected row } n, \quad \textit{dyadic bit } b_n, \\ b_n \longmapsto 1 - b_n \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}. \end{gathered}$$

For each inspected enumeration row, the witness differs from that row at the matching displayed dyadic window before the terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed. Therefore a countable* $\mathsf{StreamName}^{\uparrow}$ *enumeration cannot be used as an exhaustive Real-facing packet unless it supplies an additional displayed row that defeats the bit-flip witness through the same* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route.*

Proof. Start with an accepted carrier $D = (E, W, B, F, R, S, H, C, P, N)$. For a row listed in W, the enumeration slice E exposes the row and B exposes the dyadic bit observed at the matching diagonal window.

The row F records the opposite bit at that window. The displayed step is

$$b_n \longmapsto 1 - b_n.$$

This is a finite dyadic row, not an appeal to host real equality or to a completed set of all real names.

The readback row R repacks the flipped bits as regular rational observations, and S is the only terminal $\mathsf{Real}^{\uparrow}$ seal available to the witness. Any comparison with the inspected row must therefore pass through the displayed Cauchy-equivalence setoid route, not through quotient streams or ambient completeness.

Consequently every inspected row is separated from the diagonal witness at its own displayed dyadic window. The packet refuses host real equality, quotient streams, completed powersets, countable choice, and ambient Real completeness as sources of surjectivity data. □

Theorem 3793.4 (Real diagonal non-surjection diagonal sequence route)**.** *Every accepted* RealDiagonalNonSurjecti *diagonal witness route of Definition 3793.3 factors through the reusable* $\mathsf{DiagonalSequence}^{\uparrow}$ *schedule of chapter 3940 before the dyadic bit flip and terminal* $\mathsf{Real}^{\uparrow}$ *seal are read. The* $\mathsf{DiagonalSequence}^{\uparrow}$ *row supplies only the finite row-index and window schedule used by* $E, W, B, F, R, S, H, C, P, N$*; it supplies no host countability theorem, no completed enumeration, and no Real surjectivity principle.*

Proof. Project the accepted carrier $D = (E, W, B, F, R, S, H, C, P, N)$. The rows E and W expose the inspected enumeration slice and the finite row-index schedule before any dyadic bit is flipped.

Read this schedule through chapter 3940. The sibling packet contributes the reusable diagonal row and finite-window discipline, so the present carrier does not need a separate kernel-level scheduling surface.

Only after that schedule is displayed may B and F read the inspected dyadic bit and its opposite. The readback row R and seal row S then turn the flipped finite row into the displayed $\mathsf{Real}^{\uparrow}$ witness.

The structural rows H, C, P, N preserve the same route. Since $\mathsf{DiagonalSequence}^{\uparrow}$ supplies only finite schedule data, the factorization imports no host countability theorem, completed enumeration, quotient stream equality, or Real surjectivity principle. □

Theorem 3793.5 (Real diagonal non-surjection StreamName dependency)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *read that exposes a diagonal non-surjection witness must first expose the* $\mathsf{StreamName}^{\uparrow}$ *enumeration slice and its finite diagonal windows from chapter 126. Only after those rows are displayed can the carrier read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bit flip, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the diagonal witness route depends on the finite StreamName window surface before any dyadic flip or Real-facing seal is visible.*

Proof. Project the carrier of Definition 3793.1. The rows E and W are the enumeration slice and finite diagonal-window schedule, and both are read before the dyadic observation row B and flipped-bit row F.

The diagonal-sequence route of Definition 3793.4 preserves this order. It first exposes the finite schedule, then routes through the dyadic bit read, the regular rational readback R, and the Real seal S.

Therefore every downstream Cantor or Real consumer sees a StreamName dependency theorem: the diagonal obstruction is not a direct host uncountability claim, but a finite path through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Theorem 3793.6 (Real diagonal non-surjection Cantor-RegSeqRat dependency)**.** *Every Cantor-style consumer of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet must read the diagonal obstruction in the order*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{CantorSpace}^{\uparrow}/\mathsf{ConstructiveC}$$

Thus the packet consumed by the Cantor diagonal real-uncountability route is a finite Real-facing diagonal obstruction: it exposes the StreamName enumeration slice, the dyadic bit flip, the regular rational readback, the Cauchy-equivalence comparison boundary, the CantorSpace and constructive Cantor theorem route, and the terminal Real seal before any downstream Real consumer may cite non-surjection.

Proof. Start with the carrier of Definition 3793.1. It lists the StreamName enumeration slice E, W, the dyadic observation and flip rows B, F, the RegSeqRat readback R, the Real seal S, and the Cauchy-equivalence comparison route as displayed packet data.

Apply the diagonal witness theorem Definition 3793.3. The witness is separated from each inspected row at a displayed dyadic window before it is read back through $\mathsf{RegSeqRat}^{\uparrow}$ and sealed as $\mathsf{Real}^{\uparrow}$.

Apply Definition 3793.5 and the Cantor sibling route of Definition 3793.7. They force Cantor consumers to read the StreamName slice first and to pass through the CantorSpace and constructive Cantor theorem route only after the dyadic, regular-rational, and Cauchy-equivalence rows are visible.

Consequently the consumer receives a finite Real-facing diagonal obstruction. It does not receive a host uncountability theorem, a completed powerset of reals, quotient-stream equality, countable choice, or an ambient Real completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet is generated from a finite StreamName enumeration slice, a diagonal row-index schedule, a DyadicRatCore bit-observation row, a flipped-bit row, a RegSeqRat readback route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3793.1, Definition 3793.2, Definition 3793.3, Definition 3793.4, and Definition 3793.5 seed the finite diagonal boundary and reusable StreamName-to-Real schedule for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RealDiagonalNonSurjectionUp`

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient streams, completed powersets of real names, countable choice, selected enumeration inverses, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, enumeration-slice exposure, diagonal-window accounting, dyadic-bit readback, bit-flip exactness, RegSeqRat readback, RealUp sealing, Cauchy-equivalence comparison routing, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealDiagonalNonSurjectionUp.

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet is a Real-facing diagonal boundary rather than a kernel-only island. Its diagonal witness consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 for the displayed regular-rational readback, and the Cantor diagonal real-uncountability route of **??** consumes this chapter as its Real-facing non-surjection input before adding CantorSpace and CantorSet rows.

The non-surjection carrier first reads the reusable diagonal schedule of chapter 3940: finite stream windows select the inspected enumeration row, the diagonal bit disagreement is recorded at the selected coordinate, the $\mathsf{RegSeqRat}^{\uparrow}$ readback exposes the rational window that realizes the disagreement, and the terminal $\mathsf{Real}^{\uparrow}$ seal packages that readback as the Real-facing obstruction. Only after this route is visible may **??** and chapter 5879 consume the packet as a Cantor-facing diagonal obstruction.

Theorem 3793.7 (RealDiagonalNonSurjection Cantor sibling route)**.** *Every accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier consumed by a Cantor-style diagonal argument remains a Real-facing sibling route through* $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The consumer first reads the finite enumeration slice, diagonal row-index schedule, dyadic bit-observation row, flipped-bit row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison boundary, transport, replay, provenance, and local naming rows of Definition 3793.1; only then may the Cantor route of chapter 5879 interpret the same disagreement ledger as a Cantor-space non-enumeration witness.*

Proof. Start with the carrier and diagonal witness rows of Definition 3793.1 and Definition 3793.3. They expose the inspected enumeration row, finite diagonal window, dyadic bit read, flipped-bit row, regular-rational readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before any non-surjection conclusion is read.

Next route the displayed disagreement through the Cantor-space handoff of chapter 5879. That sibling consumes the same finite diagonal classifier over $\mathsf{CantorSpace}^{\uparrow}$ and carries it toward the Real-facing non-surjection boundary; it does not introduce a completed binary stream, host powerset, or ambient surjectivity principle.

Finally use the sibling dependency route already recorded for this chapter. The $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ comparison and $\mathsf{Real}^{\uparrow}$ seal remain visible on the Real side, so the packet is a Cantor/Real diagonal sibling route rather than a kernel-only packet. □

Theorem 3793.8 (Real diagonal non-surjection RegSeqRat dependency route)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *read that constructs the anti-diagonal* $\mathsf{Real}^{\uparrow}$ *row first exposes the finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and* $\mathsf{Real}^{\uparrow}$ *seal used by the source sequence. The consumer cannot bypass those rows with host stream equality, selected limits, quotient reals, ambient completeness, countable choice, or a hidden real-surjectivity principle.*

Proof. Start with the carrier of Definition 3793.1. The rows E and W expose only the inspected finite enumeration slice and diagonal windows, so the anti-diagonal read begins as finite $\mathsf{StreamName}^{\uparrow}$ data.

Next apply the diagonal witness route of Definition 3793.3. The dyadic bit read and bit flip pass through B and F, then the regular-rational readback R repacks the displayed windows before the terminal seal S is exposed.

Use the existing sibling dependency route of this chapter to keep the same boundary visible to downstream consumers. The sibling route already records that the diagonal packet is Real-facing, and chapter 301 and chapter 38 supply the named readback and seal surfaces.

Since the carrier has no coordinate for host stream equality, selected limits, quotient reals, ambient completeness, countable choice, or hidden surjectivity data, every anti-diagonal Real row must follow the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route. □

Theorem 3793.9 (Real diagonal non-surjection Cantor–RegSeqRat lock)**.** *Every accepted* RealDiagonalNonSurjecti *packet that claims a non-surjectivity read must factor through the Cantor diagonal row of chapter 5879, the regular rational readback row of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row of chapter 38, and the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalNonSurjection}^{\uparrow}}$ *ledger before the conclusion is exposed. In*

carrier coordinates this means that the enumeration slice E, diagonal schedule W, dyadic bit row B, flipped-bit row F, RegSeqRat row R, Real seal S, transport row H, replay row C, provenance row P, and naming row N of Definition 3793.1 remain visible in that order; no classical choice principle, host powerset, quotient stream equality, selected inverse, or ambient real-completeness row is introduced by the lock.

Proof. Start with the carrier and diagonal witness rows of Definition 3793.1 and Definition 3793.3. They expose $E, W, B, F, R, S, H, C, P, N$ before any non-surjectivity read is available.

Next use the Cantor sibling route of Definition 3793.7. The Cantor-facing consumer receives the same displayed disagreement row and not a host powerset, completed binary stream, or selected inverse.

Then apply the RegSeqRat dependency route of Definition 3793.8 and the Real source route of Definition 3793.11. The flipped bit row is read through R as regular rational data before S exposes the terminal Real-facing seal.

Finally the structural rows H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for classical choice, quotient stream equality, selected enumeration inverse, host powerset, or ambient real completeness, every accepted non-surjection claim remains locked to the Cantor, RegSeqRat, RealUp, and local NameCert rows. □

Theorem 3793.10 (Real diagonal non-surjection RegSeqRat–Real route)**.** *Every accepted* RealDiagonalNonSurjecti *read that exposes a diagonal non-surjection packet must consume the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row of chapter 301, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of chapter 38, and the Cantor-style diagonal sibling rows of Definition 3793.7 before the non-surjection surface is available.*

In carrier coordinates this route reads $E, W, B, F, R, S, H, C, P, N$ from Definition 3793.1: the enumeration slice and window schedule are exposed first, the flipped dyadic bit is repacked through R as regular rational data, the terminal seal S presents the Real-facing row, and the Cantor sibling consumes the same displayed disagreement ledger. No host stream equality, selected inverse, quotient real equality, completed powerset, or ambient Real-completeness row is introduced by the route.

Proof. Project the carrier of Definition 3793.1. The rows E and W expose only the inspected finite enumeration slice and diagonal windows, so the non-surjection read begins as $\mathsf{StreamName}^{\uparrow}$ window data.

Apply Definition 3793.8. The bit observation and flipped bit are read through B and F, then the row R repacks the disagreement as regular rational data before a Real-facing endpoint is visible.

Next apply Definition 3793.12. The seal row S is the only $\mathsf{Real}^{\uparrow}$ endpoint available to the diagonal witness, and it is exposed only after the regular readback has been displayed.

Finally use Definition 3793.7. The Cantor-style sibling consumes the same finite disagreement ledger after the StreamName, RegSeqRat, and Real seal rows are visible. The structural rows H, C, P, N transport, replay, source, and name that route, so no excluded host object can enter the non-surjection packet. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ completion-facing dependency route consumes the regular-rational readback of chapter 301 and the terminal real-name seal of chapter 38 before a diagonal non-surjection packet may refuse a surjective enumeration. This anchor keeps the L10 readback lattice visible: the inspected enumeration slice and finite diagonal windows are read first, the disagreement is repacked through $\mathsf{RegSeqRat}^{\uparrow}$, the resulting row is sealed by $\mathsf{Real}^{\uparrow}$, and only then can the packet deny that the displayed enumeration is exhaustive.

Theorem 3793.11 (Real diagonal non-surjection Real source route)**.** *Every diagonal non-surjection read of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet consumes the* $\mathsf{Real}^{\uparrow}$ *source row of chapter 38, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row of chapter 126, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301 before exposing a diagonal packet. The route exposes only the finite enumeration slice, diagonal window schedule, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows; it does not claim host uncountability, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the carrier of Definition 3793.1. The enumeration slice E and finite diagonal schedule W are read as $\mathsf{StreamName}^{\uparrow}$ source data before the bit observation row B or flipped-bit row F can be interpreted.

Apply the RegSeqRat dependency route of Definition 3793.8. The anti-diagonal readback passes through R as regular rational data and cannot bypass that row with host stream equality, selected limits, quotient reals, countable choice, or hidden surjectivity data.

Finally use the Real seal dependency of Definition 3793.12. The seal row S is the only $\mathsf{Real}^{\uparrow}$ endpoint available to the diagonal witness, and it is exposed after the stream and regular-rational source rows have already been read. The diagonal packet is therefore a downstream consumer of the $\mathsf{Real}^{\uparrow}$ terminal seal, the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback, so its non-surjection surface is unavailable unless those sibling NameCert chapters have supplied their finite rows.

Thus every diagonal packet is source-routed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the non-surjection surface is visible. The carrier has no coordinate for host uncountability, quotient stream equality, countable choice, or ambient Real completeness. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ packet consumes $\mathsf{RealBaireCylinder}^{\uparrow}$ as the finite prefix-cylinder readback surface for Real-facing diagonal rows. The sibling route through chapter 1685 first exposes the $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, then the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal; the diagonal row therefore reads the Baire cylinder as displayed Real-facing data rather than as a host stream, quotient real equality, selected branch, or ambient completeness principle.

Theorem 3793.12 (Real diagonal non-surjection Real seal dependency)**.** *Every* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *diagonal read reaches its contradiction surface only after the* $\mathsf{Real}^{\uparrow}$ *seal route of chapter 38 and the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-window rows are visible. The packet is not a host cardinality theorem, not a quotient-surjection argument, and not an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3793.1. The only analytic rows available before the terminal real-name handoff are the enumeration slice E, the finite diagonal schedule W, the dyadic bit row B, the bit-flip row F, and the regular rational readback R.

Apply Definition 3793.3. The contradiction surface is reached by comparing an inspected enumeration row with the flipped bit at the displayed diagonal window. That comparison is read through $\mathsf{DyadicRatCore}^{\uparrow}$ and then through $\mathsf{RegSeqRat}^{\uparrow}$ before the seal row S can expose the witness as $\mathsf{Real}^{\uparrow}$ data.

Now use the $\mathsf{Real}^{\uparrow}$ seal route of chapter 38. The seal does not add a separate cardinality principle; it only marks that the already displayed regular-rational finite windows have been admitted as the real-name row used by the diagonal witness.

Since the carrier has no coordinate for host cardinality, quotient surjection, selected enumeration inverse, completed powerset, countable choice, or ambient real completeness, every diagonal

contradiction remains inside the displayed $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal dependency route. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ dependency lattice consumes the Cantor-style diagonal split, the $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and the terminal $\mathsf{Real}^{\uparrow}$ seal through chapter 1530, chapter 301, and chapter 38. This route exposes only the finite diagonal disagreement, regular rational readback, and Real sealing rows; it does not claim cardinality, a powerset object, quotient equality, or host diagonal completeness.

Theorem 3793.13 (Real diagonal non-surjection metric dependency route)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer that compares outputs of a* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *packet must first read the diagonal witness through the* $\mathsf{RegSeqRat}^{\uparrow}$ *route of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. Only after those rows are exposed may the consumer use the metric comparison surface of chapter 57; the diagonal packet itself exports no direct metric distance row, quotient real equality, selected limit, countable-choice schedule, or ambient completeness witness.*

Proof. Start with the carrier of Definition 3793.1. The rows B and F produce the displayed dyadic bit disagreement, but this data is still a finite diagonal observation rather than a metric comparison.

The readback row R converts the displayed bit-flip data into the regular rational route named by Definition 3793.8. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ endpoint visible to a downstream consumer.

A $\mathsf{Met}^{\uparrow}$ consumer can compare diagonal outputs only after it has both the regular rational readback and the terminal real seal. Thus the dependency order is

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow}.$$

Transport, replay, provenance, and local naming preserve this order componentwise. Since no carrier row names a host metric distance directly, the metric consumer cannot bypass the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ sibling routes. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ dependency route combines the Cantor-style diagonal cylinder, the $\mathsf{RegSeqRat}^{\uparrow}$ finite readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and the $\mathsf{Met}^{\uparrow}$ separation surface in one visible order. A consumer first reads the finite binary disagreement through the Cantor route of chapter 5879 and the regular-rational readback of chapter 301; only then does the route expose the Real seal of chapter 38 and the metric comparison boundary of chapter 57. The packet therefore supplies a concrete dependency edge from diagonal cylinders to metric separation without adding host cardinality, quotient real equality, selected branches, countable choice, or ambient completeness.

Theorem 3793.14 (RealDiagonalNonSurjection forward Cantor consumer route)**.** *Every* CantorDiagonalRealUncou *consumer of an accepted* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ *carrier must first read the finite diagonal obstruction through the displayed* $\mathsf{StreamName}^{\uparrow}$ *enumeration slice,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *bit-observation and bit-flip rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the sibling Cantor route of Definition 3793.7. The consumer may then pass to* **??***, but only with the Real-facing seal and comparison boundary already constrained by chapter 38.*

Proof. Project the accepted carrier of Definition 3793.1. Its enumeration slice, row-index schedule, dyadic bit-observation row, flipped-bit row, regular rational readback, and Real seal are all displayed before the local diagonal witness is available.

Apply the StreamName dependency theorem and the Cantor sibling route. Theorems 3793.5 and 3793.7 force the consumer through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the disagreement ledger can be interpreted as Cantor-facing non-enumeration data.

The downstream chapter **??** therefore receives the diagonal obstruction as a finite Real-facing route, not as a free-standing host uncountability principle. The Real seal remains governed by chapter 38, so the consumer imports no quotient stream equality, completed powerset, countable choice, selected enumeration inverse, or ambient completeness. □

The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ route is the Real-facing obstruction consumed by **??** and by chapter 5879. The downstream Cantor read occurs only after $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ have already exposed the finite name, rational readback, dyadic tolerance, real seal, and equivalence-setoid rows. The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ sibling lattice consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular readback route of chapter 301, the CantorSpace chapter of chapter 1475, and the Cantor-style diagonal handoff of Definition 3793.7. These dependencies keep the non-surjection packet on the displayed Real, RegSeqRat, CantorSpace, and Cantor diagonal rows, with no host enumeration theorem, quotient stream equality, or ambient completeness coordinate. The $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$ completion-facing diagonal route is read through the $\mathsf{CantorSpace}^{\uparrow}$ finite-prefix surface of chapter 1475, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion-facing siblings chapter 360 and chapter 1506. The Cantor row supplies only the displayed binary-prefix disagreement, the regular readback row repacks that disagreement as rational observations, and the Real seal is exposed before any completion consumer may inspect the packet. The route does not claim host cardinality, quotient real equality, selected infinite branches, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 3794

A Concrete Naming Certificate for $\mathsf{CauchyNetCompletion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet records the directed Cauchy-net completion carrier that sits between the raw $\mathsf{CauchyNet}^{\uparrow}$ directed-window surface and the completion consumers already named by $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{UniformCompletion}^{\uparrow}$. It consumes the same separated comparison discipline as chapter 1411 and the same finite numerical readback route as chapter 301 and chapter 38. The chapter names the completion-facing carrier without selecting a limit, quotienting nets, invoking countable choice, asserting host equality of completed objects, importing an ambient completion, or claiming completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3794.1 (CauchyNetCompletion carrier)**.** *A* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, M, U, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index substrate read by the accepted* $\mathsf{CauchyNet}^{\uparrow}$ *row.*
2. *W is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window and common-upper-witness row over D.*
3. *Q is the completion-boundary request attached to the displayed Cauchy-net row.*
4. *M is the* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *handoff row, exposing the same directed-net completion boundary to Moore–Smith consumers.*
5. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row when the consumer reads entourages rather than metric balls.*
6. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *comparison row for comparing two displayed completion-boundary reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback row attached to the retained directed windows.*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *terminal seal reached only after D, W, Q, M, U, S, R have been displayed.*
9. *H records componentwise* hsame *transport for the directed substrate, Cauchy-net row, completion-boundary request, Moore–Smith handoff, uniform handoff, separated comparison, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from directed Cauchy-net evidence through completion-boundary reads, separated comparison, regular-rational readback, and the real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient nets, countable choice, host equality of completed objects, an ambient completion object, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionTasteGate single carrier alignment`

Theorem 3794.2 (CauchyNetCompletion NameCert obligations)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the directed substrate, Cauchy-net row, completion-boundary request, Moore–Smith handoff, uniform handoff, separated comparison, regular readback, real seal, transport, replay, provenance, and local naming row;*
2. *directed-net coverage, requiring every completion-facing request to consume D and W before Q, M, U, S, R, A can be inspected;*
3. *Moore–Smith handoff, requiring M to expose the same directed-net boundary that chapter 2190 consumes;*
4. *metric and uniform handoff, requiring* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *consumers to read the boundary through Q and U rather than through an ambient completed object;*
5. *separated comparison discipline, requiring any comparison of two displayed boundary reads to pass through S;*
6. *numerical readback discipline, requiring R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal A is inspected;*
7. *non-escape for selected limits, quotient nets, countable choice, host equality, ambient completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3794.1. The tuple displays exactly the directed substrate, the Cauchy-net window row, the completion-boundary request, the Moore–Smith and uniform handoffs, separated comparison, readback, seal, and structural rows.

Directed-net coverage is the first dependency. A completion-facing request cannot reach Q, M, U, S, R, A until the accepted $\mathsf{CauchyNet}^{\uparrow}$ row W over D has been displayed.

The handoff obligations use the sibling completion chapters. The row M feeds the Moore–Smith completion route, while Q and U feed metric and uniform completion consumers over the same displayed directed windows. The separated comparison row S supplies comparison data without host equality.

Finally R and A provide the finite numerical readback and terminal seal, and H, C, P, N preserve, replay, source, and name the same route. Since the excluded completion principles have no carrier coordinate, they cannot become NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier namecert obligations`

Theorem 3794.3 (CauchyNetCompletion Moore–Smith handoff)**.** *Every* $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *packet reads the directed-net completion boundary through*

$$D, W, Q \longmapsto M, U, S \longmapsto R, A \longmapsto H, C, P, N.$$

The route starts with the $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, passes through the Moore–Smith, metric, uniform, and separated comparison rows when those consumers are present, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the* $\mathsf{Real}^{\uparrow}$ *seal only afterward. No quotient net, selected limit, countable-choice schedule, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness row bypasses this route.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier moore smith handoff`

Proof. Project the accepted carrier of Definition 3794.1. The rows D and W are the only directed-net source rows, and Q is the first completion-boundary request attached to them.

Read the sibling handoff next. The Moore–Smith consumer receives M, metric completion consumers read the same boundary through Q, and uniform completion consumers receive U only as a handoff from the displayed Cauchy-net boundary. Comparisons of displayed boundary reads pass through S.

The numerical exit is downstream. The regular rational row R and real seal row A are available only after the directed-net and completion-boundary rows have been replayed, and the structural rows H, C, P, N preserve and name that route.

The carrier inventory contains no selected limit, quotient-net equality, countable-choice selector, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness theorem. Thus every accepted completion consumer factors through the displayed route. □

Theorem 3794.4 (CauchyNetCompletion choice-free limit boundary)**.** *Any accepted completion-limit read from a* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *packet factors through the displayed rows*

$$D, W, Q, M, U, S, R, A, H, C, P, N.$$

The separated comparison row S *mediates comparison of displayed boundary reads, and the rows* R, A *provide the finite* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal only after the directed windows and completion handoff rows have been exposed. The read exports no selected limit, quotient-net representative, countable-choice schedule, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionCarrier choice free limit boundary`

Proof. Project the carrier of Definition 3794.1. Its coordinates are exactly $D, W, Q, M, U, S, R, A, H, C, P, N$, with D, W supplying the directed substrate and Cauchy-net windows before any completion-boundary request is inspected.

Apply the Moore–Smith handoff theorem. Theorem 3794.3 forces every Moore–Smith, metric, or uniform completion consumer through the displayed route from D, W, Q to M, U, S, then to R, A, and finally to the transport, replay, provenance, and naming rows.

The comparison and numerical exits are therefore internal to the same packet. The row S is the only separated comparison coordinate, while R, A are the only regular-rational readback and real-seal coordinates reachable by the consumer.

It remains to exclude escape data. The carrier definition has no coordinate for a selected limit, quotient net, countable-choice selector, host equality, ambient completion object, or $\mathsf{Real}^{\uparrow}$ completeness theorem, so no accepted completion-limit read can export any of those objects. □

Theorem 3794.5 (CauchyNetCompletion directed-window terminality)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier*

$$K = (D, W, Q, M, U, S, R, A, H, C, P, N),$$

every completion-facing read of $Q, M, U, S, R, A, H, C, P, N$ *has terminal directed-window source* D, W*. In particular, no selected limit, quotient net, countable-choice selector, host equality of completion points, ambient completion object, or* $\mathsf{Real}^{\uparrow}$ *completeness row can supply the first directed witness for the packet.*

Proof. Project the carrier in Definition 3794.1. The only directed substrate row is D, and the only Cauchy-net directed-window row over it is W.

Apply the choice-free boundary theorem, Definition 3794.4. It places all completion-limit reads on the displayed route through $D, W, Q, M, U, S, R, A, H, C, P, N$.

The remaining rows are downstream witnesses. The request row Q, the Moore–Smith and uniform handoff rows M, U, the separated comparison row S, and the readback rows R, A can be inspected only after D, W have been displayed.

Since the excluded objects have no carrier coordinate, none of them can replace D, W as the directed source of an accepted read. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowTerminality`

Theorem 3794.6 (CauchyNetCompletion directed-window cofinal restriction)**.** *Let* $K = (D, W, Q, M, U, S, R, A, H$ *be an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier. Replace the directed substrate row* D *by a displayed cofinal restricted substrate*

$$D_{\mathrm{cof}} \hookrightarrow D$$

and replace W *by its restricted* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row* W_{cof}*, provided* W_{cof} *still carries the common-upper witnesses used by* W *for completion-boundary requests. Then the completion-facing route*

$$D, W, Q, M, U, S, R, A, H, C, P, N$$

is preserved as the restricted route

$$D_{\mathrm{cof}}, W_{\mathrm{cof}}, Q, M, U, S, R, A, H, C, P, N.$$

In particular, the restricted packet exports no selected limit, quotient net, countable-choice selector, host equality of completion points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier of Definition 3794.1. The directed substrate row D and directed-window row W are the only source rows before the completion-boundary request Q is read.

Use directed-window terminality, Definition 3794.5. It makes D, W terminal for every completion-facing read. A cofinal restricted substrate with the same common-upper witnesses supplies the same terminal source position through $D_{\mathrm{cof}}, W_{\mathrm{cof}}$.

Now apply the Moore–Smith handoff theorem, Definition 3794.3. The completion route still passes from the directed source through Q to the Moore–Smith, metric, uniform, and separated rows M, U, S, then to the regular readback and Real seal rows R, A.

The structural rows are unchanged. Componentwise transport H, displayed replay C, provenance P, and local naming N preserve the same restricted route, while the carrier has no coordinate for selected limits, quotient nets, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowCofinalRestriction`

Theorem 3794.7 (CauchyNetCompletion directed-window refinement functoriality)**.** *Let* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *be an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier. Suppose a first displayed cofinal restriction replaces* D, W *by* $D_{\mathrm{cof}}, W_{\mathrm{cof}}$*, and a second displayed cofinal restriction replaces those rows by*

$$D_{\mathrm{ref}}, W_{\mathrm{ref}}.$$

If each restricted window retains the common-upper witnesses needed for the same completion-boundary request Q*, then the composite restricted route preserves the same CauchyNetCompletion surface:*

$$D, W, Q, M, U, S, R, A, H, C, P, N \quad \rightsquigarrow \quad D_{\mathrm{ref}}, W_{\mathrm{ref}}, Q, M, U, S, R, A, H, C, P, N.$$

The composite route exports no selected limit, quotient net, countable-choice schedule, host equality of completion points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with directed-window terminality, Definition 3794.5. It makes the displayed directed-window rows the source of every completion-facing read, so the first cofinal restriction can replace D, W only by another displayed terminal source row for the same request.

Apply cofinal restriction, Definition 3794.6, to the first restriction and then to the second. The retained common-upper witnesses keep the completion-boundary request Q and the handoff rows $M, U, S, R, A, H, C, P, N$ on the same route.

Use the Moore–Smith handoff theorem to read the composite as one completion-facing package. The sibling Moore–Smith, metric, uniform, separated, regular-rational, and Real rows are not recomputed; they are transported and replayed through the final restricted window.

The carrier has no coordinate for selected limits, quotient nets, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness. Thus composing two displayed cofinal restrictions is functorial only for the visible directed-window route, and it cannot export any excluded completion principle. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionDirectedWindowRefinementFunctoriality`

Theorem 3794.8 (CauchyNetCompletion separated-comparison exhaustion)**.** *For an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$*, every comparison between two displayed completion-boundary reads is exhausted by the separated row* S*. The comparison may use* H, C, P, N *to preserve, replay, source, and name the row, but it exports no quotient-net equality, host equality, ambient completion equality, selected limit, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionSeparatedComparisonExhaustion`

Proof. Start from the NameCert obligations of Definition 3794.2. They identify S as the separated comparison row attached to the displayed completion-boundary request.

Use Definition 3794.4. Any completion-limit read must first pass through D, W, Q, M, U and reaches comparison only at S.

The structural rows do not add a new comparison source. They transport, replay, remember provenance, and name the same separated row, so the comparison inventory is exhausted by S.

The carrier has no quotient-net, host-equality, ambient-completion, selected-limit, countable-choice, or $\mathsf{Real}^{\uparrow}$-completeness coordinate. Those excluded objects therefore cannot appear as comparison evidence. □

Theorem 3794.9 (CauchyNetCompletion uniform handoff nonescape)**.** *Every* UniformCompletion$^{\uparrow}$ *consumer of an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *must read its completion boundary through* U *after the directed-window source* D, W *and the request row* Q*. The handoff cannot escape through a selected limit, quotient net, countable-choice selector, host equality, ambient completion object, or* Real$^{\uparrow}$ *completeness row.*

Proof. Project the carrier. The uniform handoff row is U, and it is listed only after the directed substrate D, the Cauchy-net window W, and the completion-boundary request Q.

Apply Definition 3794.3. It sends every Moore–Smith, metric, or uniform completion consumer through the displayed route from D, W, Q to M, U, S.

The choice-free boundary theorem, Definition 3794.4, keeps the later readback rows R, A, H, C, P, N downstream of the same route.

Thus a uniform consumer that bypasses U or replaces D, W, Q by any excluded completion object is outside the accepted CauchyNetCompletion$^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionUniformHandoffNonescape`

Theorem 3794.10 (CauchyNetCompletion root filter admission)**.** *Every root completion consumer of an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *must admit the directed substrate* D*, retained directed-window row* W*, and completion-boundary request* Q *before Moore–Smith, metric, uniform, separated, readback, or Real-seal rows are inspected. No root consumer may replace* D, W, Q *by a selected limit, quotient net, countable-choice selector, host completion equality, ambient completion object, or* Real$^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootFilterAdmission`

Proof. Project the carrier of Definition 3794.1. The first three root rows are exactly D, W, Q, and the Moore–Smith, uniform, separated, readback, and Real-seal rows occur only downstream.

Apply the NameCert obligations of Definition 3794.2. Directed-net coverage requires a completion-facing request to consume D and W before Q, M, U, S, R, A can be inspected.

The Moore–Smith handoff, Definition 3794.3, keeps metric and uniform consumers on the same route. Thus root admission starts with D, W, Q for every completion consumer.

The excluded completion objects have no carrier coordinate, so none can serve as the admitted root filter row. □

Theorem 3794.11 (CauchyNetCompletion root eventual stability)**.** *Suppose an accepted* CauchyNetCompletion$^{\uparrow}$ *carrier retains a directed-window row* W *over* D*, and a later common-upper-window witness replaces* W *by a retained row* W' *over the same directed substrate and completion-boundary request* Q*. Then the accepted packet is preserved through the separated comparison row* S*, regular readback row* R*, Real seal* A*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming row* N*.*

Proof. Start with the directed-window terminality theorem, Definition 3794.5. It makes D, W the terminal directed-window source for every completion-facing read.

The later common-upper-window witness is still a retained directed-window row over the same D and Q. Replacing W by W' therefore changes only the displayed window coordinate and not the completion-boundary request or the sibling handoff route.

Apply the choice-free boundary theorem, Definition 3794.4. The rows S, R, A are downstream of the directed-window and completion-boundary route, so their accepted reads are preserved by the later witness.

Finally the structural rows H, C, P, N transport, replay, source, and name the same retained packet. Since no selected limit, quotient net, countable choice, host equality, ambient completion, or $\mathsf{Real}^{\uparrow}$ completeness row participates, eventual stability is a finite retained-window preservation claim. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootEventualStability`

Theorem 3794.12 (CauchyNetCompletion root seal nonescape)**.** *In every accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$*, the Real seal* A *cannot be read before the directed substrate* D*, directed-window row* W*, completion-boundary request* Q*, Moore–Smith handoff* M*, uniform handoff* U*, separated comparison row* S*, and regular readback row* R *have been displayed. Reading* A *exports no selected limit, quotient net, countable choice, host completion equality, ambient completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier. The row A is listed after D, W, Q, M, U, S, R, and the carrier gives no alternate Real-seal source.

Apply Definition 3794.3. It routes Moore–Smith, metric, and uniform completion consumers from D, W, Q through M, U, S before the numerical readback rows are reached.

Use the choice-free limit boundary of Definition 3794.4. That theorem places R, A, H, C, P, N downstream of the displayed directed-net completion route and excludes hidden limit data.

The carrier contains no selected-limit, quotient-net, countable-choice, host-equality, ambient-completion, or $\mathsf{Real}^{\uparrow}$-completeness coordinate. Therefore the root Real seal cannot export any of those objects. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRootSealNonescape`

Falsifiable prediction: If a directed-net completion consumer reads a boundary without first displaying the $\mathsf{CauchyNet}^{\uparrow}$ directed-window row and the Moore–Smith, metric, or uniform handoff row it consumes, it is not a $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet.

Independence witness: cauchynet, moore-smith-completion, metriccompletion, uniformcompletion, separatedmetric, regseqrat, real: the packet is not bijective to these siblings because it binds the directed Cauchy-net carrier, completion-boundary request, sibling handoffs, separated comparison, finite readback, and terminal seal in one local NameCert route.

Theorem 3794.13 (CauchyNetCompletion public handoff)**.** *The public completion surface of an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *is the reusable handoff package formed by the directed substrate* D*, the retained* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row* W*, the completion-boundary request* Q*, the Moore–Smith, metric, and uniform handoff rows* M, U*, the separated comparison row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A*, and the structural rows* H, C, P, N*. Any downstream completion consumer may reuse this package only as the displayed directed-net completion route; it receives no selected limit, quotient net, countable-choice schedule, ambient completion object, host equality of completed points, or* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from the carrier and the NameCert obligations. They display the directed substrate, the Cauchy-net window, the completion-boundary request, the sibling completion handoffs, separated comparison, numerical readback, Real seal, transport, replay, provenance, and local naming rows.

Use the Moore–Smith handoff and choice-free limit boundary. These theorems force every Moore–Smith, metric, and uniform completion read through $D, W, Q, M, U, S, R, A, H, C, P, N$ and

exclude selected limits, quotient nets, countable choice, host equality, ambient completion, and Real completeness from the route.

Apply directed-window terminality, cofinal restriction, root eventual stability, and root seal nonescape. They make the directed-window source reusable under retained windows and cofinal restriction while keeping the terminal Real seal behind the displayed completion route.

Thus the named rows assemble a public handoff surface: a downstream consumer can cite the package as completion-facing data, but cannot replace it by selected limits, quotient nets, or an ambient Real-completeness principle. □

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionPublicHandoff`

Theorem 3794.14 (CauchyNetCompletion regular-sequence public bridge)**.** *Suppose an accepted* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *carrier* $K = (D, W, Q, M, U, S, R, A, H, C, P, N)$ *factors its directed Cauchy-net packet through a cofinal unary* $\mathsf{StreamName}^{\uparrow}$ *schedule. If the reads along that schedule satisfy the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity predicate and the separated comparison exhausted by* S *is the displayed* $\mathsf{Real}^{\uparrow}$ *seal* A*, then the* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *public handoff exports only the same regular-sequence completion data*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ S,\ A,\ H,\ C,\ P,\ N.$$

It exports no additional directed-net limit, quotient net, selected representative, countable-choice schedule, host equality of completed points, ambient completion object, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.CauchyNetCompletionUp.CauchyNetCompletionRegularSequencePublicBridge`

Proof. Start from the public handoff theorem. The public surface is the displayed route through $D, W, Q, M, U, S, R, A, H, C, P, N$, with no selected limit, quotient net, countable choice, host equality, ambient completion, or Real-completeness row.

Use directed-window terminality and cofinal restriction. The cofinal unary $\mathsf{StreamName}^{\uparrow}$ schedule may replace the displayed directed-window source only as a retained terminal source for the same completion-boundary request Q; it does not add a new limit coordinate.

Read the regularity assumption through the existing readback row R. Since the schedule reads satisfy the $\mathsf{RegSeqRat}^{\uparrow}$ regularity predicate, the numerical exit of the restricted route is the regular-sequence readback already named by R.

Apply separated-comparison exhaustion. The row S exhausts comparisons between displayed boundary reads, and by hypothesis its separated comparison is precisely the displayed $\mathsf{Real}^{\uparrow}$ seal A.

Finally use uniform-handoff nonescape and the Moore–Smith handoff. They keep every completion consumer on the same displayed route, so the public packet can export only the regular-sequence completion data carried by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, S, A, H, C, P, N and none of the excluded completion principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyNetCompletion}^{\uparrow}$ packet is generated from a directed substrate, a CauchyNet directed-window row, a completion-boundary request, MooreSmithCompletion, MetricCompletion, UniformCompletion, SeparatedMetric, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyNetCompletion}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 3794.1, Definition 3794.2, Definition 3794.5, Definition 3794.6, Definition 3794.11, Definition 3794.12, Definition 3794.13, and Definition 3794.14 expose the finite directed-net completion route as a public handoff and paper bridge through directed-window data, cofinal restriction, root eventual stability, root seal, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependencies.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyNetCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient nets, countable choice, host equality of completed objects, ambient completion, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Bridge checking requires a Lean standard-bridge target for the public directed-net completion handoff and regular-sequence public bridge.

Chapter 3795

A Concrete Naming Certificate for $\mathsf{RealTightness}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{RealTightness}^{\uparrow}$ packet records located dyadic trap data over a displayed $\mathsf{Real}^{\uparrow}$ source as finite NameCert evidence. It consumes the real-name seal of chapter 38, the dyadic rational grid of chapter 291, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, the located interval surface of chapter 1709, the bounded located-set row of chapter 2176, and the regular Cauchy located-order discipline of chapter 1681. The packet gives later Real and Completion consumers a finite located-tightness route; it is not Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3795.1 (Real tightness carrier)**.** *A* $\mathsf{RealTightness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$$

with the following visible rows:

1. *R is the source* $\mathsf{Real}^{\uparrow}$ *row whose name is being trapped.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid row supplying dyadic endpoints, midpoint refinements, and tolerance comparisons.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the requested real-name observations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
5. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row recording the current located interval trap.*
6. *B is the* $\mathsf{LocatedBoundedRealSet}^{\uparrow}$ *bounded-source row that bounds the admitted trap requests.*
7. *O is the* $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$ *row recording located-order comparisons between the real-name readback and the dyadic trap endpoints.*
8. *T is the located dyadic trap ledger: a finite list of intervals, dyadic tolerances, inside/outside locator reads, and retained-window witnesses.*
9. *H records componentwise* hsame *transport for R, D, S, Q, I, B, O, T and the structural rows.*
10. *C records displayed* Cont *replay of the real-name read, dyadic trap refinement, located-interval admission, bounded-set read, located-order comparison, and terminal trap ledger.*

11. *P is the* Pkg *provenance over* $\mathsf{RealTightness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedBoundedRealSet}^{\uparrow}$, $\mathsf{RegularCauchyLocatedOrder}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$ *row.*

The carrier has no coordinate for Heine–Borel, open-cover compactness, trichotomy of arbitrary reals, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealTightnessUp.RealTightnessTasteGate single carrier alignment`

Theorem 3795.2 (Real tightness NameCert obligations)**.** *For an accepted* $\mathsf{RealTightness}^{\uparrow}$ *carrier* $Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real source, dyadic grid, stream windows, regular rational readback, located interval, bounded located set, located-order row, trap ledger, transport, replay, provenance, and local naming row;*
2. *dyadic trap exposure, requiring every retained trap in T to display dyadic endpoints from D and the matching finite observation window from S;*
3. *located interval admission, requiring each trap read to pass through I and the bounded-source row B before it can be consumed as a located real interval;*
4. *regular Cauchy located-order comparison, requiring the readback row Q and the comparison row O to certify the displayed relation between R and the dyadic trap endpoints;*
5. *replay stability under componentwise* hsame *transport and displayed* Cont *routes for R, D, S, Q, I, B, O, T;*
6. *nonescape for Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3795.1. The tuple lists exactly the real source, dyadic grid, finite window, regular readback, located interval, bounded-set, located-order, trap, transport, replay, provenance, and naming rows.

Trap exposure is finite. Each trap row in T reads dyadic endpoints and tolerances from D, then matches them to the finite observation window named by S and the regular rational readback named by Q.

Located admission is downstream of those displayed rows. The located interval row I records the interval trap, while B records that the trap request stays inside the bounded located-set surface available to the consumer.

The comparison step is the located-order row O. It compares the regular Cauchy readback of Q with the dyadic endpoints retained in T, and it is transported and replayed only by H and C.

The carrier inventory contains no coordinate for Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient real completeness. Hence the six rows exhaust the local NameCert surface. □

Theorem 3795.3 (Real tightness located trap route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing located trap read of an accepted* $\mathsf{RealTightness}^{\uparrow}$ *carrier factors through the finite route*

$$R \longmapsto S, Q \longmapsto D, I, B, O \longmapsto T, H, C, P, N.$$

The route starts with the real-name source, reads finite stream windows and regular rational approximants, opens the dyadic grid and located interval rows, checks boundedness and located order, and only then exports the trap ledger as the named $\mathsf{RealTightness}^{\uparrow}$ *surface. It exports no Heine–Borel theorem, open-cover compactness principle, trichotomy classifier, countable-choice selector, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with an accepted carrier $Z = (R, D, S, Q, I, B, O, T, H, C, P, N)$. The source real-name row is R, and the first observable data are the finite windows S and the regular rational readback Q attached to that source.

The dyadic and located stages follow. The row D supplies endpoints and tolerances, I admits the current located interval, B bounds the trap request, and O compares the readback with the displayed endpoints through the regular Cauchy located-order surface.

The trap ledger T is therefore downstream of all numeric and located rows. The structural rows H, C, P, N transport, replay, record provenance, and name that same route without adding a host-level compactness or completeness object.

Since no carrier coordinate stores Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, or ambient real completeness, every accepted located trap read remains inside the displayed finite route. □

Falsifiable prediction: Every accepted RealTightnessUp consumer read of a located dyadic trap factors through RealUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, LocatedRealIntervalUp, LocatedBoundedRealSetUp, RegularCauchyLocatedOrderUp, and the local trap ledger before it can be exported.

Independence witness: real, dyadicratcore, regseqrat, streamname, locatedrealinterval, locatedboundedrealset, regularcauchylocatedorder: the packet is not bijective to these siblings because it binds the real source, finite windows, regular readback, dyadic trap grid, located interval, bounded located-set, located-order comparison, and local NameCert row as one located-tightness route.

Closure status for $\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTightness}^{\uparrow}$ packet is generated from a Real source row, DyadicRatCore grid row, StreamName finite-window row, RegSeqRat readback row, LocatedRealInterval row, LocatedBoundedRealSet row, RegularCauchyLocatedOrder row, located dyadic trap ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$, seed)

Scope closed: Definition 3795.1, Definition 3795.2, and Definition 3795.3 seed the finite located-tightness carrier and trap route for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealTightness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Heine–Borel, open-cover compactness, trichotomy, countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic trap exposure, located interval admission, bounded-source discipline, regular Cauchy located-order comparison, replay stability, and nonescape for BEDC.Derived.RealTightnessUp.

Chapter 3796

A Concrete Naming Certificate for $\mathsf{SchwarzLemma}^{\uparrow}$

Following Schwarz and Ahlfors, the $\mathsf{SchwarzLemma}^{\uparrow}$ packet names the unit-disk Schwarz lemma as finite BEDC certificate data. It consumes the holomorphic-disk source interface of chapter 68, the coordinate and boundary rows of chapter 377, the complex carrier of chapter 39, and the contraction handoff of chapter 3682. The packet records the pointed disk map, the zero-fixing row, the unit-bound row, the modulus comparison ledger, transport, provenance, and local NameCert rows; it is not a host theorem about arbitrary analytic continuation or a completed complex function space.

Definition 3796.1 (Schwarz lemma carrier)**.** *A* $\mathsf{SchwarzLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (H, D, Z, B, M, R, T, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Holomorphic}^{\uparrow}$ *row naming the disk-local complex map over the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
2. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *source and target disk-coordinate row for the displayed input and output points.*
3. *Z is the zero-fixing row recording that the displayed map sends the named disk origin to the named disk origin.*
4. *B is the unit-bound row recording that displayed disk inputs are routed to displayed disk outputs.*
5. *M is the modulus-comparison row recording the Schwarz-lemma read $|f(z)| \leq |z|$ on the displayed disk point and the boundary equality refusal attached to the same row.*
6. *R is the route row through* $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, *and the local modulus ledger.*
7. *T stores componentwise* hsame *and displayed* Cont *transport for the holomorphic map row, disk-coordinate row, zero-fixing row, unit-bound row, modulus-comparison row, and route row.*
8. *P is the* Pkg *provenance over* $\mathsf{SchwarzLemma}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, D, Z, B, M, R, T, P, N$. It has no coordinate for a host identity theorem, Riemann mapping theorem, unlogged analytic continuation, completed function-space equality, or selected boundary extremal map.

Lean4 (definition) `BEDC.Derived.SchwarzLemmaUp.SchwarzLemmaUp`
Lean4 ✓ `BEDC.Derived.SchwarzLemmaUp.TasteGate.SchwarzLemmaTasteGate single carrier alignment`

Theorem 3796.2 (Schwarz lemma NameCert obligations)**.** *For every accepted* $\mathsf{SchwarzLemma}^{\uparrow}$ *carrier*

$$L = (H, D, Z, B, M, R, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission for the displayed holomorphic disk map, disk coordinates, zero-fixing row, unit-bound row, modulus-comparison row, route row, transport rows, provenance, and local name;*
2. *classifier stability under componentwise* hsame *and displayed* Cont *transport of exactly those rows;*
3. *zero-fixing exactness, requiring the Schwarz-lemma read to pass through Z before any modulus comparison is exported;*
4. *disk-bound exactness, requiring every input-output read to pass through the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate and unit-bound rows before the local name is consumed;*
5. *nonescape for the identity theorem, Riemann mapping, host analytic continuation, completed complex function-space equality, and selected boundary extremal maps.*

Proof. Carrier admission is projection from Definition 3796.1. The tuple exposes the holomorphic source row, disk-coordinate row, zero-fixing row, unit-bound row, modulus-comparison ledger, route, transport, provenance, and local name as one finite packet.

Classifier stability is componentwise. A transported packet may compare the same displayed rows after hsame and Cont replay, but it cannot keep the modulus comparison while dropping the zero-fixing or unit-bound rows.

Zero-fixing and disk-bound exactness follow from the route row R. The modulus comparison row M is read only after H, D, Z, B have been displayed, so a consumer cannot use it as an arbitrary disk-map contraction theorem.

The nonescape clauses are read from the carrier inventory. Since no row stores a host identity theorem, Riemann mapping theorem, analytic-continuation principle, completed function-space equality, or selected boundary extremal map, the local NameCert cannot export one. □

Theorem 3796.3 (Schwarz lemma disk-contraction handoff)**.** *Every disk-contraction consumer of an accepted* $\mathsf{SchwarzLemma}^{\uparrow}$ *packet factors through the ordered route*

$$\mathsf{Holomorphic}^{\uparrow} \longrightarrow \mathsf{Complex}^{\uparrow} \longrightarrow \mathsf{UnitDisk}^{\uparrow} \longrightarrow \mathsf{SchwarzLemma}^{\uparrow} \longrightarrow \mathsf{SchwarzPickDiskContraction}^{\uparrow}.$$

The handoff supplies only the displayed holomorphic disk-map row, zero-fixing row, unit-bound row, modulus-comparison row, transport, provenance, and local NameCert data. It supplies no full complex analytic identity theorem, Riemann mapping theorem, host analytic continuation principle, or ambient completed function-space equality.

Proof. Start with the holomorphic source row H over the complex carrier and the disk-coordinate row D. These rows are the only source of the displayed map and input-output points available to the handoff.

The zero-fixing row Z and unit-bound row B then restrict the read to the pointed disk situation. The modulus-comparison row M records the Schwarz-lemma inequality only for that displayed route.

Finally pass the same rows through R, T, P, N and into the $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ consumer. The downstream contraction packet can read the displayed disk contraction, but no coordinate in L adds identity-theorem reasoning, Riemann mapping, host analytic continuation, or ambient function-space equality. □

Falsifiable prediction: If a purported SchwarzLemmaUp packet exports a modulus comparison without displaying the zero-fixing row and the UnitDisk route, it is not a $\mathsf{SchwarzLemma}^{\uparrow}$ carrier.

Independence witness: holomorphic, complex, unitdisk, schwarz-pick-disk-contraction: the packet is not bijective to any one sibling because it binds the zero-fixing and unit-bound rows to a local modulus-comparison NameCert before the disk-contraction handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwarzLemma}^{\uparrow}$ packet is generated from a holomorphic disk-map row, complex carrier row, UnitDisk coordinate row, zero-fixing row, unit-bound row, modulus-comparison row, route row, transport, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3796.1, Definition 3796.2, and Definition 3796.3 seed the pointed disk-map carrier, NameCert obligations, and disk-contraction handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwarzLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a full complex analytic identity theorem, Riemann mapping theorem, host analytic continuation principle, Lean verification, or bridge checking for $\mathsf{SchwarzLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, zero-fixing exactness, unit-bound exactness, modulus comparison, route preservation, transport stability, provenance, and local naming for BEDC.Derived.SchwarzLemmaUp.

Chapter 3797

A Concrete Naming Certificate for $\mathsf{CauchyProductCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyProductCompletion}^{\uparrow}$ packet records the constructive product of two completed real carriers by finite Cauchy windows, regular rational readback, and an explicit completion seal. It keeps the product-completion surface separate from ambient Cartesian powersets: each exported product read must still name the stream windows, dyadic observations, product budget, regular-sequence classifier, and real-completion seal that make the read finite.

Definition 3797.1 (CauchyProductCompletion carrier)**.** *A* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N)$$

with the following displayed rows:

1. S_0, S_1 *are paired* $\mathsf{StreamName}^{\uparrow}$ *windows for the two completion-facing coordinates.*
2. D_0, D_1 *are* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation rows for the finite rational approximants read from the two streams.*
3. B *is the* $\mathsf{CauchyProduct}^{\uparrow}$ *budget row, recording the common tolerance window used to multiply or pair the displayed approximants.*
4. R_0, R_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier rows, certifying that each coordinate read is regular enough to enter the real-completion route.*
5. L_0, L_1 *are* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility rows, witnessing that the coordinate windows are read only through the completion seal used by the L10 route.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *export seal for the product-completion read.*
7. H *records the* hsame *transports for stream names, dyadic observations, regular classifiers, product budget, limit seals, and real export.*
8. C *records the* Cont *replays that move from stream windows through dyadic observations, regular-sequence readback,* $\mathsf{CauchyProduct}^{\uparrow}$ *budget,* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *compatibility, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
9. P *is the* Pkg *provenance over* $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}$ *row.*

The accepted carrier contains no ambient product set, no selected representative of a completion quotient, no unrestricted sequence equality, and no completion principle beyond the displayed finite-window route.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionUp.CauchyProductCompletionTasteGate single carrier alignment`

Theorem 3797.2 (CauchyProductCompletion NameCert obligations)**.** *For an accepted packet* $Q = (S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^\uparrow}$ *obligation surface consists exactly of carrier admission, classifier stability, budget factorization, completion-seal nonescape, and ledger exactness for the displayed rows* $S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N$*.*

Proof. Carrier admission reads the two $\mathsf{StreamName}^\uparrow$ windows first, then admits the two $\mathsf{DyadicRatCore}^\uparrow$ observations only as finite approximants attached to those windows. The $\mathsf{RegSeqRat}^\uparrow$ rows certify the regular readback needed by the real-completion route.

Classifier stability is carried by H. Transporting stream names, dyadic observations, regular classifiers, product budget, limit seals, and the real export moves the whole packet together, so no coordinate can be reclassified by an external quotient equality.

Budget factorization is supplied by B and replayed by C. The product-facing read reaches $\mathsf{CauchyProduct}^\uparrow$ only after both coordinate windows and dyadic observations are displayed, and it reaches $\mathsf{Real}^\uparrow$ only after $\mathsf{CauchyLimitSeal}^\uparrow$ compatibility has been exposed.

Completion-seal nonescape follows from the L_0, L_1, E rows. They bind the coordinate reads to the L10 sibling routes of chapter 126, chapter 291, chapter 301, chapter 464, chapter 386, and chapter 38.

Ledger exactness is an inspection of the packet. Every exported read is one of the displayed rows, so the local NameCert has no coordinate from which to recover an ambient product object, a representative-choice operation, unrestricted sequence equality, or a completion principle outside the finite route. □

Theorem 3797.3 (CauchyProductCompletion budget factorization)**.** *Every completion-facing product read exported by* $\mathsf{CauchyProductCompletion}^\uparrow$ *factors through* $\mathsf{StreamName}^\uparrow$*,* $\mathsf{DyadicRatCore}^\uparrow$*,* $\mathsf{RegSeqRat}^\uparrow$*,* $\mathsf{CauchyProduct}^\uparrow$*,* $\mathsf{CauchyLimitSeal}^\uparrow$*, and* $\mathsf{Real}^\uparrow$ *before export. In particular, the product budget is not a free-standing multiplication oracle: it is a finite tolerance-window route whose coordinate streams, dyadic observations, regular classifiers, completion seals,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows remain visible at the export boundary.*

Proof. Start with a completion-facing product read. By Definition 3797.1, the only carrier rows available before export are $S_0, S_1, D_0, D_1, B, R_0, R_1, L_0, L_1, E, H, C, P, N$.

The first stage is the paired stream window. The rows S_0, S_1 force the read to pass through the $\mathsf{StreamName}^\uparrow$ route before any dyadic observation is available, and D_0, D_1 then expose the $\mathsf{DyadicRatCore}^\uparrow$ approximants.

The second stage is regularization and budget. The rows R_0, R_1 record the $\mathsf{RegSeqRat}^\uparrow$ classifiers, while B records the $\mathsf{CauchyProduct}^\uparrow$ tolerance budget that combines the coordinate approximants.

The final stage is sealing and export. The rows L_0, L_1 expose $\mathsf{CauchyLimitSeal}^\uparrow$ compatibility before E exports a $\mathsf{Real}^\uparrow$ product-completion read. Since H, C, P, N are the only structural rows, the export cannot bypass this route through an ambient product, selected completion representative, unrestricted sequence equality, or hidden real-completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletion}^{\uparrow}$ packet is generated from paired StreamName windows, DyadicRatCore observations, RegSeqRat classifiers, a CauchyProduct budget row, CauchyLimitSeal compatibility rows, a Real export seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3797.1, Definition 3797.2, and Definition 3797.3 bind the finite product-completion seed used by the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed product universal property, quotient equality of completed reals, arbitrary sequence multiplication, compactness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, classifier stability, budget factorization, completion-seal nonescape, and ledger exactness for BEDC.Derived.CauchyProductCompletionUp.

Chapter 3798

A Concrete Naming Certificate for $\mathsf{CantorSpaceCompactness}^{\uparrow}$

Following Brouwer and Bishop, the $\mathsf{CantorSpaceCompactness}^{\uparrow}$ packet records the constructive compactness route for Boolean streams as a finite NameCert surface adjacent to $\mathsf{CantorSpace}^{\uparrow}$ in chapter 1475. It reads finite prefix data through $\mathsf{FanTheorem}^{\uparrow}$, obstruction rows through $\mathsf{KleeneTree}^{\uparrow}$, and observation windows through $\mathsf{StreamName}^{\uparrow}$; it does not import open-cover compactness, countable choice, quotient equality of infinite streams, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3798.1 (CantorSpaceCompactness carrier)**.** *A* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, F, T, S, B, H, R, Q, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix source row, including the Boolean prefix observations and cylinder reads named in Definition 1475.1;*
2. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *row, recording the finite decidable bar, its inspected prefix depth, and the finite branch-window witness;*
3. *T is the* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row, used only to record that a displayed infinite-branch escape is absent from the finite fan route;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row that schedules the finite prefix requests consumed by C and F;*
5. *B is the prefix-cover row, listing the finitely many Boolean cylinders whose union covers the inspected prefix surface;*
6. *H records componentwise* hsame *transport for the Cantor-space, fan, obstruction, stream, prefix-cover, continuation, provenance, and local naming rows;*
7. *R records displayed* Cont *replay from a finite prefix request through the fan-depth row and the prefix-cover row;*
8. *Q is the refusal row excluding open-cover compactness, countable choice, quotient equality of infinite streams, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
9. *P is the* Pkg *provenance over* $\mathsf{CantorSpaceCompactness}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *prefix covers,* BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$ *row naming the displayed finite compactness packet and its permitted consumer reads.*

The classifier compares accepted packets only through the displayed Cantor prefix, fan-depth, obstruction, stream-window, and prefix-cover rows, transported by H and replayed by R. No classifier coordinate compares ambient infinite streams or completed function-space objects.

Lean4 ✓ `BEDC.Derived.CantorSpaceCompactnessUp.CantorSpaceCompactnessTasteGate single carrier alignment`

Theorem 3798.2 (CantorSpaceCompactness NameCert obligations)**.** *For every accepted* CantorSpaceCompactness *carrier $K = (C, F, T, S, B, H, R, Q, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed* $\mathsf{CantorSpace}^{\uparrow}$ *source,* $\mathsf{FanTheorem}^{\uparrow}$ *finite-bar row,* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row,* $\mathsf{StreamName}^{\uparrow}$ *schedule, prefix-cover ledger, transport, replay, refusal, provenance, and local naming row;*
2. *prefix-cover exactness, requiring every accepted compactness read to use the same finite Boolean cylinders listed in B and the same inspected prefix schedule in S;*
3. *fan-depth stability, requiring the finite bar depth in F to bound the cylinders consumed by a downstream compactness read;*
4. *obstruction nonescape, requiring T to remain a finite refusal row rather than a source of an infinite branch;*
5. *ledger nonescape, excluding open-cover compactness, countable choice, quotient equality of infinite streams, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection of Definition 3798.1. The packet exposes exactly $C, F, T, S, B, H, R, Q, P, N$ before any compactness-facing read is permitted.

Prefix-cover exactness reads B together with the $\mathsf{StreamName}^{\uparrow}$ schedule S. Fan-depth stability reads F and forces the listed cylinders to sit below the inspected finite bar depth.

The obstruction row T is then read only as a finite nonescape boundary. Transport by H and replay by R preserve the same displayed route, while Q, P, N record refusal, provenance, and local naming. Since none of these rows contains open-cover compactness, countable choice, quotient equality, or ambient real completeness, the listed obligations exhaust the local surface. □

Theorem 3798.3 (CantorSpaceCompactness finite prefix subcover)**.** *Let $K = (C, F, T, S, B, H, R, Q, P, N)$ be an accepted* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier. Every compactness-facing read that asks for a cover of a displayed* $\mathsf{CantorSpace}^{\uparrow}$ *prefix is answered by the finite prefix-cover row B after the fan-depth row F has fixed the inspected depth and the* $\mathsf{StreamName}^{\uparrow}$ *schedule S has fixed the visible positions. The subcover is therefore a finite list of Boolean cylinders over the displayed prefix surface, not an open-cover compactness principle and not a choice-selected family of cylinders.*

Proof. Start with the $\mathsf{CantorSpace}^{\uparrow}$ row C. By Definition 1475.5, every cylinder read of C factors through finite $\mathsf{StreamName}^{\uparrow}$ windows, finite Boolean observations, and local NameCert rows.

Read the fan row F. It supplies a finite decidable bar depth and the inspected branch-window witness, so the compactness-facing request is reduced to the cylinders visible below that depth.

Now use the prefix-cover ledger B. The row lists the finitely many Boolean cylinders that cover the inspected prefix surface, and the replay row R routes the consumer through exactly that

ledger. The refusal row Q excludes open-cover compactness, countable choice, quotient stream equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the exported subcover. □

Theorem 3798.4 (CantorSpaceCompactness nonescape)**.** *For every accepted* $\mathsf{CantorSpaceCompactness}^{\uparrow}$ *carrier* $K = (C, F, T, S, B, H, R, Q, P, N)$*, any downstream consumer of compactness data remains inside the route*

$$C, S \longmapsto F \longmapsto B \longmapsto T \longmapsto H, R, Q, P, N.$$

The route exports only finite Boolean prefix covers for the displayed Cantor-space packet. It exports no open-cover compactness theorem, no countable choice principle, no quotient equality of streams, no selected infinite branch through a $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 3798.3. A compactness-facing read first consumes the displayed Cantor prefix row C and the stream schedule S, then reads the fan-depth row F and the finite prefix-cover ledger B.

The obstruction row T is downstream of that finite route. It records that the displayed route has no accepted infinite-branch escape; it does not provide a selected branch or a host-level tree path.

Finally inspect H, R, Q, P, N. These rows transport, replay, refuse, source, and name only the displayed finite packet. Therefore a consumer has no row from which to recover open-cover compactness, countable choice, quotient stream equality, an infinite branch selector, or ambient real completeness. □

Falsifiable prediction: Every $\mathsf{CantorSpaceCompactness}^{\uparrow}$ compactness read factors through a displayed $\mathsf{CantorSpace}^{\uparrow}$ prefix, a finite $\mathsf{FanTheorem}^{\uparrow}$ bar-depth row, and a finite prefix-cover ledger before any consumer sees a compactness-shaped output.

Independence witness: cantorspace, fantheorem, kleenetree, streamname: $\mathsf{CantorSpaceCompactness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Cantor prefix data, fan-depth coverage, obstruction nonescape, stream scheduling, prefix-cover rows, transport, continuation replay, package provenance, refusal, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3798.1, Definition 3798.2, Definition 3798.3, and Definition 3798.4 seed the constructive finite-prefix compactness packet over $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, prefix covers, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSpaceCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, countable choice, quotient equality of streams, selected infinite branches, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix subcover exactness, fan-depth stability, obstruction nonescape, and local NameCert replay for BEDC.Derived.CantorSpaceCompactnessUp.

Constructive story: A $\mathsf{CantorSpaceCompactness}^{\uparrow}$ packet is generated from CantorSpace finite-prefix rows, FanTheorem finite-bar rows, KleeneTree obstruction rows, StreamName schedules, prefix-cover ledgers, componentwise hsame transport, displayed Cont replay, refusal, Pkg provenance, and a local NameCert row.

Chapter 3799

A Concrete Naming Certificate for $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$

Following Knopp and Hardy on condensation tests for positive series, the $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ packet records the finite comparison route between a monotone nonnegative series and its Schlomilch-block condensation. It sits beside the dyadic block discipline of chapter 3598, the integral comparison surface of chapter 2259, and the real series endpoint route of chapter 1581. The packet names block maps, block masses, monotonicity windows, tail comparisons, and endpoint seals as finite NameCert data; it does not assert an unrestricted host convergence theorem or choose an infinite sum.

Definition 3799.1 (CauchySchlomilchCondensation carrier)**.** *A* $\mathsf{CauchySchlomilchCondensation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, M, B, Q, T, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the nonnegative real-series row, including the inspected finite coefficient window and the displayed monotonicity direction.*
2. *M is the monotone tail row certifying that later blocks may be bounded by their boundary terms.*
3. *B is the Schlomilch block-index row, listing the increasing block cut points and their finite gap sizes.*
4. *Q is the block-mass comparison row, recording the finite inequalities between original tails and condensed block terms.*
5. *T is the tail-transfer row that routes convergence or divergence reads through the displayed block comparison.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealSeries}^{\uparrow}$ *endpoint seal reached only after A, M, B, Q, T have been displayed.*
7. *H records componentwise* hsame *transport for series rows, monotonicity rows, block ledgers, tail transfers, endpoint seals, provenance, and local naming.*
8. *C records displayed* Cont *replay from a tail request through the monotone series row, Schlomilch block ledger, comparison row, tail transfer, and endpoint seal.*

9. *P is the* Pkg *provenance over* CauchySchlomilchCondensation$^{\uparrow}$, RealSeries$^{\uparrow}$, CauchyCondensationTest$^{\uparrow}$, CauchyIntegralTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchySchlomilchCondensation}^{\uparrow}}$ *row.*

The carrier has no coordinate for a nonmonotone rearrangement, an unstated analytic integral, a selected infinite sum, quotient real equality, or a convergence claim outside the displayed Schlomilch blocks.

Lean4 ✓ `BEDC.Derived.CauchySchlomilchCondensationUp.CauchySchlomilchCondensationTasteGate`
`single carrier alignment`

Theorem 3799.2 (CauchySchlomilchCondensation NameCert obligations)**.** *For every accepted* CauchySchlomilchCondensation$^{\uparrow}$ *carrier $S = (A, M, B, Q, T, E, H, C, P, N)$, the local* NameCert$_{\text{CauchySchlomilchCondensa}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nonnegative series row, monotone tail row, block-index ledger, block-mass comparison, tail-transfer row, endpoint seal, transport, replay, provenance, and local naming;*
2. *monotone tail discipline, requiring M before any block comparison row can be consumed;*
3. *Schlomilch block exactness, requiring B and Q to name the same finite cut points and gap sizes;*
4. *condensation handoff, requiring every convergence-facing or divergence-facing tail read to pass through T before reaching E;*
5. *sibling compatibility, allowing dyadic Cauchy condensation and integral-test rows to be read only as named comparison siblings, not as hidden analytic premises;*
6. *ledger non-escape, excluding nonmonotone rearrangement, selected infinite sums, detached integral estimates, quotient real equality, and ambient completeness.*

No condensation obligation is stored outside these rows.

Proof. Project the carrier of Definition 3799.1. The rows $A, M, B, Q, T, E, H, C, P, N$ are the only displayed coordinates of the accepted packet.

Read the tail discipline first. The monotone row M governs the series row A before the Schlomilch cut points B and block comparisons Q are exposed, so the block ledger cannot be detached from the source tail.

Next apply the tail-transfer row T. It is the unique route from the displayed block comparison to the endpoint seal E, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic or quotient coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: A candidate packet with Schlomilch cut points but no monotone tail row is rejected by the CauchySchlomilchCondensation$^{\uparrow}$ carrier.

Independence witness: condensationtailselector, cauchy-integral-test, realseries, regseqrat, streamname: none is bijective to CauchySchlomilchCondensation$^{\uparrow}$ because this packet jointly carries monotone source-series rows, Schlomilch block cut points, block-mass comparisons, tail-transfer rows, endpoint seals, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{CauchySchlomilchCondensation}^{\uparrow}}$ **·**

Constructive story: A CauchySchlomilchCondensation$^{\uparrow}$ packet is generated from a monotone nonnegative series row, Schlomilch block-index ledger, block-mass comparison, tail-transfer row, endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchySchlomilchCondensation}\uparrow}$, seed)

Scope closed: Definition 3799.1 and Definition 3799.2 seed the finite Schlomilch condensation packet over monotone tails, block ledgers, comparison rows, and endpoint seals.

Formal verification: FormalStatus(NameCert$_{\text{CauchySchlomilchCondensation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, nonmonotone rearrangement, selected infinite sums, host integral estimates, quotient real equality, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, monotone tail discipline, Schlomilch block exactness, condensation handoff, sibling compatibility, and ledger non-escape for BEDC.Derived.CauchySchlomilchCondensationUp.

Chapter 3800

A Concrete Naming Certificate for $\mathsf{FiniteOscillationPartition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteOscillationPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed oscillation window for each cell. It sits after the closed bounded interval surface of chapter 1591, the located interval source of chapter 1709, the oscillation-modulus ledger of chapter 2224, the real-name seal of chapter 38, the regular rational sequence readback of chapter 301, and the stream source of chapter 126. The packet gives later uniform-continuity, Darboux, Riemann, and regulated-integral certificates a finite-means carrier for oscillation-controlled partitions without invoking open-cover compactness, quotient equality of real functions, selected suprema, or ambient real completeness.

Definition 3800.1 (Finite oscillation partition carrier)**.** *A* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, M, A, B, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded and located* $\mathsf{Real}^{\uparrow}$ *interval row, read through the endpoint, inclusion, and apartness surfaces of* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *and* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$*.*

2. *M is the finite mesh row, listing ordered partition points, adjacent cell rows, endpoint containment, and a common finite index ledger.*

3. *A is the oscillation-modulus row, recording the tolerance request and the finite cellwise radius or mesh constraint supplied by* $\mathsf{OscillationModulus}^{\uparrow}$*.*

4. *B is the cell-bound row, assigning to each cell a displayed upper and lower real-name readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is used.*

5. *R is the refinement-readback row, recording how a later common refinement or integration consumer reads I, M, A, B without changing the accepted classifier.*

6. *H records componentwise* hsame *transport for I, M, A, B, R and for the continuation, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from an oscillation-window request through*

$$I, M, A, B, R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{OscillationModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row.*

The carrier contains no open-cover compactness witness, no quotient equality of real-valued functions, no selected supremum or infimum, no unbounded partition search, and no ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.FiniteOscillationPartitionUp.FiniteOscillationPartitionTasteGate single carrier alignment`

Theorem 3800.2 (Finite oscillation partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *carrier $O = (I, M, A, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, interval-source exposure through I, mesh exactness through M, oscillation-window accounting through A, cell-bound readback through B, refinement-readback stability through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3800.1. The interval, mesh, oscillation, cell-bound, and refinement rows are the analytic part of the packet, while H, C, P, N are the structural rows that transport, replay, source, and name it.

The interval and mesh obligations are finite. The row I supplies the closed bounded located interval boundary, and M lists the partition cells and endpoint containments over that same boundary.

The oscillation and bound obligations are also finite. The row A records the tolerance and the cellwise oscillation constraint, while B exposes upper and lower readbacks through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Refinement stability is read from R and transported by H. A downstream consumer may refine the displayed cells only by replaying the same interval, mesh, oscillation, and bound rows through C with provenance P and local name N.

Since the carrier has no coordinate for open-cover compactness, quotient function equality, selected extrema, unbounded search, or ambient completeness, no accepted NameCert obligation can project one of those objects. □

Lean4 ✓ `BEDC.Derived.FiniteOscillationPartitionUp.FiniteOscillationPartition namecert obligations`

Theorem 3800.3 (Finite oscillation partition uniform window)**.** *Every uniform-continuity or integration-facing consumer of an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{LocatedRealIntervalPartition}^{\uparrow},\ \mathsf{OscillationModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto$$

In particular, the consumer reads a displayed interval source, finite mesh, oscillation tolerance ledger, cell-bound readback, and refinement route before any terminal real seal is used.

Proof. Apply the NameCert obligations of Definition 3800.2. They require every accepted consumer to open I, M, A, B, R before the structural rows are available.

The uniform window is determined by A over the cells of M. Thus a uniform-continuity consumer obtains a tolerance-indexed cell ledger rather than an open-cover argument.

An integration-facing consumer reads the same route through B and R. The cell-bound readbacks expose finite upper and lower rows through stream and regular rational names, while the refinement row records how common refinements preserve the displayed classifier.

Therefore the accepted packet exports a finite oscillation-controlled partition window, and no hidden supremum, quotient-function comparison, open-cover compactness witness, or ambient completeness theorem is available to the consumer. □

Theorem 3800.4 (Finite oscillation partition uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet before a* $\mathsf{UniformModulus}^{\uparrow}$ *handoff must pass through exactly the displayed interval source, finite mesh, oscillation-window row, cell-bound readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row. The handoff exports a finite oscillation route to a uniform-modulus consumer, not open-cover compactness, selected extrema, quotient function equality, unbounded partition search, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3800.1. Its analytic rows are the interval source I, finite mesh M, oscillation-window row A, cell-bound readback B, and refinement-readback row R; its structural rows are H, C, P, N.

Apply Definition 3800.2. The local certificate forces any compact-uniform-continuity consumer to expose carrier admission, interval-source exposure, mesh exactness, oscillation accounting, cell-bound readback, refinement stability, transport, replay, provenance, and local naming before treating the packet as usable modulus data.

Apply Definition 3800.3. The uniform window is read only after $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal have been displayed over the same finite partition rows.

Thus the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the displayed finite oscillation handoff. Since no carrier coordinate stores open-cover compactness, selected extrema, quotient function equality, unbounded partition search, or ambient completeness, none of those principles can be exported by the handoff. □

Theorem 3800.5 (Compact-uniform finite oscillation modulus)**.** *Every compact-uniform-continuity or integration-facing consumer that receives an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet obtains a finite oscillation modulus only through the displayed interval source, finite mesh, oscillation-window row, cell-bound readback,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$ *row. The received modulus is a finite packet read, not an open-cover compactness theorem, selected-extrema argument, quotient function equality, unbounded partition search, or ambient completeness principle.*

Proof. Project the carrier of Definition 3800.1. The only analytic rows available to a compact-uniform or integration-facing consumer are the interval source I, finite mesh M, oscillation-window row A, cell-bound readback B, and refinement route R.

Apply Definition 3800.2. It forces the consumer to expose interval-source data, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, componentwise transport, replay, provenance, and local naming before the packet can be consumed as a modulus.

Use Definition 3800.3 to identify the uniform window read. The window factors through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal, so the compact-uniform reader receives a finite tolerance-indexed route rather than a host-level compactness principle.

Finally inspect the absent coordinates of the carrier. It contains no open-cover compactness witness, selected extrema, quotient comparison of functions, unbounded partition search, or ambient completeness row; hence none of those principles can be projected from the finite oscillation modulus. □

Theorem 3800.6 (Finite oscillation partition step-function handoff)**.** *Let $O = (I, M, A, B, R, H, C, P, N)$ be an accepted* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *packet whose cell-bound row B chooses displayed lower and upper real-name reads for every cell of the finite mesh M. Then every* $\mathsf{StepFunction}^{\uparrow}$ *or* $\mathsf{RegulatedFunction}^{\uparrow}$ *consumer that reads the packet receives a finite approximation route through*

$$I, M, A, B, R,\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ H, C, P, N.$$

The route first exposes the interval source, finite mesh, oscillation modulus, stream windows, regular rational readbacks, dyadic tolerance rows, and terminal real seal; it then repacks the displayed cellwise constant lower and upper reads as $\mathsf{StepFunction}^{\uparrow}$ *rows and as the finite schedule visible to* $\mathsf{RegulatedFunction}^{\uparrow}$*. No open-cover compactness, selected extrema, quotient functions, unbounded partition search, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle is introduced by this handoff.*

Proof. Project the carrier of Definition 3800.1. The interval source I, mesh M, oscillation row A, cell-bound row B, and refinement row R are already finite rows of the accepted packet, while H, C, P, N provide componentwise transport, displayed replay, provenance, and the local naming certificate.

Apply Definition 3800.2. The NameCert obligations force every consumer to expose the interval source, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, transport, replay, provenance, and local name before the packet can be read as approximation data.

Use Definition 3800.5 to expose the uniform cell window. For each cell of M, the selected lower and upper reads in B pass through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, are calibrated by the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and reach the terminal $\mathsf{Real}^{\uparrow}$ seal only along that displayed finite route.

Now repack each displayed cellwise constant read as a $\mathsf{StepFunction}^{\uparrow}$ row over the same interval source and finite mesh. Reading the finitely many rows in the order recorded by R gives the regulated-function finite schedule: the $\mathsf{RegulatedFunction}^{\uparrow}$ consumer sees only the interval, mesh, step rows, oscillation window, readbacks, transport, replay, provenance, and local name.

Finally inspect the absent coordinates of the carrier. Since the packet has no coordinate for open-cover compactness, selected extrema, quotient functions, unbounded partition search, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the step-function and regulated-function consumers cannot acquire any of those principles through this handoff. □

Falsifiable prediction: Any accepted $\mathsf{FiniteOscillationPartition}^{\uparrow}$ consumer that claims a uniform oscillation window while omitting one of the interval source, finite mesh, oscillation-modulus, cell-bound, refinement, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: closedboundedinterval, located-real-interval-partition, oscillationmodulus, real, regseqrat, streamname: none is bijective with $\mathsf{FiniteOscillationPartition}^{\uparrow}$ because this packet jointly carries interval source, finite mesh, oscillation tolerance, cell-bound readback, refinement stability, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOscillationPartition}^{\uparrow}$ packet is generated from a closed bounded located interval row, a finite mesh row, an oscillation-modulus row, cell-bound readbacks through StreamName and RegSeqRat, a refinement-readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3800.1, Definition 3800.2, Definition 3800.3, and Definition 3800.6 seed the finite oscillation-controlled partition route for uniform-continuity, step-function, regulated-function, and integration-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient equality of real-valued functions, selected suprema, unbounded partition search, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, mesh exactness, oscillation-window accounting, cell-bound readback, refinement stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.FiniteOscillationPartitionUp.

Chapter 3801

A Concrete Naming Certificate for $\mathsf{TriggerBlockerDuality}^{\uparrow}$

The $\mathsf{TriggerBlockerDuality}^{\uparrow}$ packet realizes the blocker-complex reading of Definition 9462.1547, Definition 9462.1548, Definition 9462.1550, and Definition 9462.1551 as finite BEDC NameCert data. It consumes the six-boundary coverage packet of chapter 3470, the finite support algebra of chapter 3478, the Boolean trigger lattice of chapter 3704, the Alexander-dual trigger algebra of chapter 3555, and the DNA boundary-tile source of chapter 3393. The packet names trigger-blocker duality by finite blocker-complex, maximal-blocker, intersection, nerve, transport, provenance, and local NameCert rows; it is not a biological stochastic model, a hidden codon semantics, an unbounded trigger search, or an external homology oracle.

Definition 3801.1 (Trigger-blocker duality carrier)**.** *A* $\mathsf{TriggerBlockerDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, L, B, I, V, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row exposing the finite boundary-trigger coordinates and coverage ranks.*
2. *A is the* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *row carrying the finite trigger-support closure and blocker threshold data.*
3. *L is the* $\mathsf{BooleanTriggerLattice}^{\uparrow}$ *row recording meet, join, rank, and observed trigger-table reads.*
4. *B is the blocker-complex row, displaying the four maximal blockers obtained by omitting one trigger coordinate and the non-trigger face predicate of Definition 9462.1547.*
5. *I is the intersection row, recording the sizes* $14, 6, 2, 0$ *for one-, two-, three-, and four-fold omitted-coordinate intersections.*
6. *V is the nerve row, recording the four vertices, six edges, four triangular faces, and absent three-simplex of the blocker nerve.*
7. *H records componentwise* hsame *transport for S, A, L, B, I, V and for the continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from trigger coverage through blocker readback and nerve readback.*

9. *P is the* Pkg *provenance over* TriggerBlockerDuality$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$ BooleanTriggerLattice$^{\uparrow}$, AlexanderDualTriggerAlgebra$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, *the source blocker-complex vision,* BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{TriggerBlockerDuality}^{\uparrow}}$ *row.*

The carrier contains no hidden biological semantics, no stochastic evolutionary model, no quotient codon space, no unbounded trigger enumeration, and no external homology oracle.

Lean4 ✓ `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityTasteGate single carrier alignment`

Theorem 3801.2 (Trigger-blocker duality NameCert obligations)**.** *For every accepted* TriggerBlockerDuality$^{\uparrow}$ *carrier $D = (S, A, L, B, I, V, H, C, P, N)$, the local* NameCert$_{\text{TriggerBlockerDuality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, coverage exposure through S, trigger-algebra exposure through A, Boolean lattice exposure through L, blocker-complex exactness through B, maximal-blocker intersection accounting through I, blocker-nerve readback through V, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityNameCertObligations`

Proof. Project the carrier of Definition 3801.1. The rows S, A, L expose the trigger side, the rows B, I, V expose the blocker and nerve side, and H, C, P, N transport, replay, source, and name the packet.

The trigger side is finite because S displays the boundary-tile coverage coordinates, A displays the support-closure algebra, and L displays Boolean meet, join, and rank rows over the same finite support surface.

The blocker side is finite because B lists the four maximal omitted-coordinate blockers, I records the displayed intersection sizes, and V records the nerve readback. These are exactly the rows needed by Definition 9462.1551.

Transport and replay preserve the same rows componentwise. Since the carrier has no coordinate for hidden biological meaning, stochastic evolution, quotient codon semantics, unbounded search, or an external homology oracle, no accepted NameCert obligation can project one of those objects. □

Theorem 3801.3 (Trigger-blocker duality finite nerve readback)**.** *Every accepted* TriggerBlockerDuality$^{\uparrow}$ *read of the blocker nerve factors through the finite route*

SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, BooleanTriggerLattice$^{\uparrow}$, AlexanderDualTriggerAlgebra$^{\uparrow}$,

Removing one maximal-blocker row changes the blocker nerve readback because V is computed from the four displayed blocker rows and their intersection ledger I.

Proof. Apply the obligation surface of Definition 3801.2. It forces every accepted nerve read to expose the trigger-side coverage, algebra, and Boolean lattice rows before the blocker rows are available.

The nerve row V is tied to B and I. The four vertices are the four maximal-blocker rows, the six edges and four triangular faces are read from the nonempty two- and three-fold intersections, and the absent three-simplex is read from the empty four-fold intersection.

If one maximal-blocker row is removed, the vertex set of V changes and the intersection ledger no longer has the same four omitted-coordinate coordinates. Thus the accepted blocker nerve readback changes before any downstream trigger-spectrum consumer can use the packet. □

Falsifiable prediction: Removing one displayed maximal-blocker row from an accepted $\mathsf{TriggerBlockerDuality}^{\uparrow}$ carrier changes the blocker nerve readback through V before any downstream trigger-spectrum consumer can preserve the certificate.

Independence witness: six-boundary-tile-coverage, window-coverage-trigger-algebra, boolean-trigger-lattice, alexander-dual-trigger-algebra, dna-window-six-boundary-tile: none is bijective with $\mathsf{TriggerBlockerDuality}^{\uparrow}$ because this packet jointly carries blocker-complex rows, maximal-blocker rows, intersection rows, nerve rows, transport, replay, provenance, and local naming.

The Lean follow-up for this chapter must provide `Nontrivial TriggerBlockerDualityUp` and `FieldFaithful TriggerBlockerDualityUp` instances in `BEDC.Derived.TriggerBlockerDualityUp` so that $S, A, L, B, I, V, H, C, P, N$ remain separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}$.

Constructive story: A $\mathsf{TriggerBlockerDuality}^{\uparrow}$ packet is generated from six-boundary coverage rows, trigger-algebra rows, Boolean trigger-lattice rows, blocker-complex rows, maximal-blocker intersection rows, blocker-nerve rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3801.1, Definition 3801.2, and Definition 3801.3 seed the finite blocker-complex, maximal-blocker, intersection, and nerve packet consumed by downstream trigger-spectrum chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriggerBlockerDuality}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.TriggerBlockerDualityUp.TriggerBlockerDualityNameCertObligations`

Bridge status: none

Not claimed: This chapter does not close biological stochastic semantics, quotient codon semantics, unrestricted trigger search, external homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coverage exposure, trigger-algebra exposure, Boolean lattice exposure, blocker-complex exactness, intersection accounting, nerve readback, Nontrivial TriggerBlockerDualityUp, and FieldFaithful TriggerBlockerDualityUp for BEDC.Derived.TriggerBlockerDualityUp.

Chapter 3802

A Concrete Naming Certificate for $\mathsf{FiniteRelation}^{\uparrow}$

Following Tarski-style relation algebra and finite relational semantics, the $\mathsf{FiniteRelation}^{\uparrow}$ packet records a finite binary relation as displayed BEDC pair rows rather than as a host predicate, quotient graph, or finite map with discarded duplicate rows.

3802.1 Finite-relation source

Definition 3802.1 (FiniteRelationUp carrier)**.** *A* $\mathsf{FiniteRelation}^{\uparrow}$ *carrier from a displayed left surface to a displayed right surface is a finite* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *pair rows. Each row has first component a left* BHist *witness and second component a right* BHist *witness, with both components read through their local classifiers and package provenance.*

Membership in the relation is exact display membership in the spine: a pair (a, b) *is accepted precisely when some product-coded row in the spine has left component classified with* a *and right component classified with* b*. The domain support is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration obtained by projecting the left components of all rows, and the range support is the* $\mathsf{FinSet}^{\uparrow}$ *enumeration obtained by projecting the right components. Repeated rows and repeated component classes remain visible ledger entries; unlike* $\mathsf{FiniteMap}^{\uparrow}$ *lookup in chapter 352, no first-hit rule suppresses later rows and no functionality condition is imposed.*

Lean4 ✓ `BEDC.Derived.FiniteRelationUp.FiniteRelationTasteGate single carrier alignment`

Theorem 3802.2 (FiniteRelationUp NameCert obligations)**.** *The* $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$ *obligation surface consists of seven rows:*

1. *the finite* $\mathsf{List}^{\uparrow}$ *carrier of* $\mathsf{Prod}^{\uparrow}$ *pair rows from Definition 3802.1;*
2. *membership exactness for displayed pair membership;*
3. *domain and range support projections through* $\mathsf{FinSet}^{\uparrow}$*;*
4. *classifier transport for both components and for the pair-row membership classifier;*
5. *replay through the displayed* Cont *route;*
6. *source provenance through the recorded* Pkg *and* BHist *rows;*
7. *local naming of the finite relation packet as a* NameCert *row.*

Together these rows expose a non-functional finite relation surface whose consumers may cite chapter 36, chapter 34, chapter 32, chapter 33, chapter 112, and chapter 352 without importing host relation equality.

Proof. Project the displayed relation spine. The carrier row is the finite $\mathsf{List}^{\uparrow}$ object, while each member row is a $\mathsf{Prod}^{\uparrow}$ pair whose two components retain their component classifiers and provenance.

Membership exactness is immediate from spine membership: a consumer can accept a pair only by pointing to a displayed row and transporting both components along their classifiers. Domain and range support are obtained by component projection followed by the finite-set enumeration interface, so repeated component classes remain ledger-visible.

Replay and provenance are inherited rowwise. The Cont route replays the same inspected product row, the Pkg row names the source packet, and the BHist entries identify the displayed component histories. These rows give the local naming certificate without requiring functionality or quotienting duplicate rows. □

Theorem 3802.3 (FiniteRelationUp image and preimage route)**.** *Every public image or preimage read of an accepted* $\mathsf{FiniteRelation}^{\uparrow}$ *packet factors only through the displayed finite-data rows named in Definition 3802.2. An image read starts from a displayed left witness, scans the* $\mathsf{List}^{\uparrow}$ *spine of* $\mathsf{Prod}^{\uparrow}$ *rows, tests component membership through the left classifier, and exports the corresponding right witnesses through a* $\mathsf{FinSet}^{\uparrow}$ *range-support row. A preimage read is the symmetric route through the right classifier and left support row.*

The route may use $\mathsf{Bool}^{\uparrow}$ *as a finite discriminator for visible classifier outcomes and* $\mathsf{Option}^{\uparrow}$ *as an optional read witness for single-row consumers, but it never turns the relation into a* $\mathsf{FiniteMap}^{\uparrow}$ *first-hit lookup. All exported facts pass through* $\mathsf{List}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Option}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$ *rows.*

Proof. For an image read, inspect the relation spine in order. Each accepted row is a displayed product row, so the left component can be compared with the queried witness through the component classifier and transported along hsame when the classifier accepts.

The accepted right components are collected as a finite support enumeration. The same construction with left and right interchanged gives the preimage route. The boolean and optional witnesses are only visible finite-data control rows for the scan; they do not choose a hidden branch or collapse the relation to a function.

Thus every consumer-visible output is a projection of displayed pair rows, replayed by Cont and named by Pkg provenance under the relation certificate. No image or preimage read can bypass the finite spine or import host relational semantics. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$, seed)

Scope closed: The chapter fixes the finite relation carrier, membership exactness surface, and image/preimage dependency route as paper-side obligations.

Constructive story: A finite relation is built by displaying a finite list spine of product-coded rows and projecting membership, domain support, and range support from that spine.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteRelation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: No Lean target, relational algebra operations, transitive closure, graph reachability, or category hom-relation bridge is claimed here.

Upgrade path: Formalize the finite spine carrier and membership classifier as *BEDC.Derived.FiniteRelationUp*, then discharge the NameCert obligation surface.

Chapter 3803

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy sequences, the $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ packet names the finite tail-envelope data needed when a regular rational sequence is handed to the $\mathsf{Real}^{\uparrow}$ completion route. It sits between the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the explicit modulus surface of chapter 3768, and the terminal real-name seal of chapter 38. The packet is finite tail-control data for the L10 regular-Cauchy route; it is not a selected real limit, quotient sequence, countable-choice schedule, host equality comparison, or ambient completeness theorem.

Definition 3803.1 (Regular Cauchy tail envelope carrier)**.** *A* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, U, D, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row whose rational entries are read through the* $\mathsf{RegSeqRat}^{\uparrow}$ *route.*
2. *U is the unary tail-bound row. For each precision request it records a finite tail index beyond which the source row is inspected.*
3. *D is the dyadic tolerance ledger, translating the precision request and the chosen tail bound into* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison budgets.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window selected by the tail bound.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the inspected window as regular rational sequence data.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after S, U, D, W, R have been replayed.*
7. *H records componentwise* hsame *transport for the source row, tail bound, dyadic ledger, window, readback, seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a precision request through $S, U, D, W, R, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$, $\mathsf{RegularModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the source row, unary tail bound, dyadic ledger, StreamName window, RegSeqRat readback, and Real seal all describe the same displayed regular Cauchy route. No carrier coordinate stores a quotient representative, selected limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEnvelopeUp.RegularCauchyTailEnvelopeTasteGate single carrier alignment`

Theorem 3803.2 (Regular Cauchy tail envelope NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier* $E = (S, U, D, W, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$ *surface consists exactly of source regular Cauchy admission, unary tail-bound display, dyadic tolerance accounting, finite StreamName window selection, RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows* $S, U, D, W, R, Q, H, C, P, N$*.*

Proof. Project the carrier of Definition 3803.1. The source row S is the only regular Cauchy input, and U is the only tail-bound coordinate attached to it.

The ledger D translates the requested precision and the displayed tail bound into dyadic comparison data. The window row W and readback row R then expose exactly the finite observations consumed by the StreamName and RegSeqRat surfaces.

The seal row Q is downstream of those rows. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, and the excluded quotient, selected-limit, choice, host-equality, and completeness data have no carrier coordinate. □

Theorem 3803.3 (Regular Cauchy tail envelope handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ *carrier must read the finite route*

$$S, U, D, W \longmapsto R \longmapsto Q \longmapsto H, C, P, N$$

before consuming the regular rational row or terminal real-name seal. Thus the tail envelope supplies named finite tail control for the L10 route rather than an informal appeal to a hidden limit, quotient stream, countable-choice schedule, host equality, or ambient completeness coordinate.

Proof. Start with a consumer request for regular rational or real-name data. The packet exposes the source row only as S, so the request must first read the tail bound U and the dyadic ledger D.

The finite window W is selected from that bound and ledger. The readback row R repacks the same inspected data as $\mathsf{RegSeqRat}^{\uparrow}$ information, so the consumer reaches the regular rational surface only through displayed finite observations.

The terminal seal Q is reached after this readback. The rows H, C, P, N then transport, replay, source, and name the same route; no excluded analytic coordinate is present, so every handoff remains finite BEDC data. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailEnvelope}^{\uparrow}$ packet is generated from a source regular Cauchy row, a unary tail-bound row, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 3803.1, Definition 3803.2, and Definition 3803.3 seed the finite tail-envelope surface for the L10 RegSeqRat-to-Real route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient sequences, countable choice, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-bound exactness, dyadic tolerance accounting, StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailEnvelopeUp.

Chapter 3804

A Concrete Naming Certificate for $\mathsf{VariationNorm}^{\uparrow}$

Following Jordan and the bounded-variation treatment in Rudin, the $\mathsf{VariationNorm}^{\uparrow}$ packet names the finite norm row obtained from displayed partition variation data. It consumes the $\mathsf{BoundedVariation}^{\uparrow}$ partition ledger of chapter 398, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the regular completion route through chapter 301, chapter 126, and chapter 291. The packet records a finite variation-norm observation over displayed partitions; it does not introduce a completed Banach space of bounded-variation functions, a host supremum over all partitions, or quotient equality of functions.

Definition 3804.1 (VariationNorm carrier)**.** *A* $\mathsf{VariationNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, R, Q, S, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed interval row, finite partition ledger, endpoint-name rows, dyadic edge observations, finite variation sum, and common-refinement rows.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and value seal read from the same partition ledger.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-name row used when the endpoint values are consumed through rational readback.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row from which the regular-name reads are inspected.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge-difference and finite-sum ledger.*
6. *A is the displayed variation-norm row, recording the finite base-value read together with the finite partition-variation sum read from B.*
7. *H records componentwise* hsame *transport for the bounded-variation source, Real seal, regular-name row, stream window, dyadic ledger, norm row, continuation, provenance, and local name.*
8. *C records the displayed* Cont *replay route from a variation-norm request through $B, R, Q, S, D, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed bounded-variation packet, Real or regular endpoint reads, stream windows, dyadic variation ledger, variation-norm row, and structural transport and provenance. It has no coordinate for a completed function space, host function equality, variation measure, hidden partition family, or supremum over all partitions.

Lean4 ✓ `BEDC.Derived.VariationNormUp.VariationNormTasteGate single carrier alignment`

Theorem 3804.2 (VariationNorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationNorm}^{\uparrow}$ *carrier* $V = (B, R, Q, S, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$ *obligation surface consists exactly of bounded-variation source admission, Real endpoint sealing, RegSeqRat readback, finite StreamName window selection, DyadicRatCore edge accounting, displayed variation-norm row formation, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier. The only source of variation data is the $\mathsf{BoundedVariation}^{\uparrow}$ row B, and the only real-valued reads attached to it are the Real seal R, the regular-name row Q, the stream window S, and the dyadic ledger D.

The variation-norm row A is then formed from those displayed rows. It reads a finite base-value component and the displayed partition-variation sum, both already present in the bounded-variation ledger and its Real/Completion dependency route.

Transport and replay are handled by H and C. These rows move the same finite source, endpoint, stream, dyadic, and norm data componentwise, while P and N record provenance and local naming. Since the excluded analytic objects have no carrier coordinate, they cannot be part of the local obligation surface. □

Theorem 3804.3 (VariationNorm refinement stability)**.** *Let* $V = (B, R, Q, S, D, A, H, C, P, N)$ *and* $V' = (B', R', Q', S', D', A', H', C', P', N')$ *be accepted* $\mathsf{VariationNorm}^{\uparrow}$ *carriers whose bounded-variation source packets compare by a displayed common-refinement classifier. Then the variation-norm rows* A *and* A' *compare componentwise after transporting the Real endpoint seals, RegSeqRat readbacks, StreamName windows, DyadicRatCore edge ledgers,* Pkg *provenance, and local* NameCert *rows through the stored* hsame *and* Cont *routes.*

Proof. Begin with the common-refinement classifier for the source $\mathsf{BoundedVariation}^{\uparrow}$ packets. By chapter 398, the comparison is a displayed finite refinement with transported endpoint rows, dyadic edge observations, finite variation sums, and package and naming provenance.

Read the Real and completion-facing coordinates along that same refinement. The rows R, Q, S, D and R', Q', S', D' expose the Real seal, regular-name readback, stream windows, and dyadic edge ledgers without adding a hidden partition or completed function carrier.

This variation-norm row is a carrier and classifier consumer of the $\mathsf{BoundedVariation}^{\uparrow}$ partition ledger and of the L10 Real/Completion readback route through $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$; it is not a decorative citation detached from the finite source rows.

Finally replay the norm rows through C and C' and transport them through H and H'. The row A is built from the same finite base-value and partition-variation data as A', after the displayed common refinement, so the variation-norm comparison is stable under the classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{VariationNorm}^{\uparrow}$ packet is generated from a BoundedVariation source, Real endpoint seal, RegSeqRat readback, StreamName window, DyadicRatCore edge ledger, displayed variation-norm row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{VariationNorm}^{\uparrow}}$, obligation)

Scope closed: Definition 3804.1, Definition 3804.2, and Definition 3804.3 bind the finite variation-norm carrier, obligation surface, and common-refinement stability route to the displayed BoundedVariation and Real/Completion dependency rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{VariationNorm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed bounded-variation function spaces, host function equality, a variation measure, hidden partition families, a supremum over all partitions, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked finite partition-norm consumer theorem over the same bounded-variation source, Real seal, regular-name, stream-window, dyadic-ledger, transport, replay, provenance, and local naming rows.

Chapter 3805

A Concrete Naming Certificate for $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges on regular Cauchy names, the $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ packet records the finite basis rows that sit between regular-Cauchy closeness and the terminal real seal. It refines the dyadic-entourage surface of chapter 1628 by naming basis generators before they are consumed by the $\mathsf{Real}^{\uparrow}$ route. The packet reads the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the real endpoint seal of chapter 38; it is not an ambient uniformity parameter, a quotient equality of Cauchy presentations, a selected limit, countable choice, or a completeness theorem.

Definition 3805.1 (Regular Cauchy entourage-basis carrier)**.** *A* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, B, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational Cauchy presentation whose finite windows are being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row carrying the displayed observation windows for the same presentation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic radii, denominator scales, and comparison data available to those windows.*
4. *B is the basis-entourage row, assigning each displayed dyadic tolerance scale to the finite entourage generator read from R, W, D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the source, window, tolerance, and basis-entourage rows have been displayed.*
6. *H records componentwise* hsame *transport for the regular source, window, tolerance, basis, real seal, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a basis-entourage request through R, W, D, B, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$, $\mathsf{RegularCauchyUniformity}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient uniform-space object, an arbitrary entourage filter, quotient equality of real presentations, a selected limit, countable choice, or real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyEntourageBasisUp.RegularCauchyEntourageBasisTasteGate single carrier alignment`

Theorem 3805.2 (Regular Cauchy entourage-basis NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier* $K = (R, W, D, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, regular-source exposure through* R*, finite-window exposure through* W*, dyadic-tolerance exposure through* D*, basis-entourage exactness through* B*, real-seal nonescape through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3805.1. The source, window, and tolerance rows are the only inputs from which the basis-entourage row can be read.

Basis-entourage exactness is finite because B assigns displayed dyadic tolerance scales from D to displayed windows from W for the same regular source row R. This gives the basis-level row used by chapter 1628 without importing an ambient uniformity object.

The real seal is nonescaping because E is reached only after R, W, D, B have been displayed and transported through H. Replay through C preserves the same route, while P and N record provenance and local naming for that finite packet. Thus no quotient equality, selected limit, countable-choice schedule, or completeness theorem can be projected from the NameCert surface. □

Theorem 3805.3 (Regular Cauchy entourage-basis Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyEntourageBasis}^{\uparrow} \longmapsto \mathsf{RegularCauchyUniformit}$$

The route exposes a basis-entourage generator before the uniformity and real-seal rows are read, and exports no ambient uniformity parameter, no selected completion limit, and no host equality of real presentations.

Proof. Start with the accepted carrier $K = (R, W, D, B, E, H, C, P, N)$. A real-seal request first reads R as the regular rational source, W as its finite observation window, and D as the dyadic tolerance ledger for that window.

The basis row B is then read as the finite entourage generator for the displayed tolerance scale. The regular-Cauchy uniformity consumer can use this row as its basis-level input, but it receives no coordinate for a larger ambient uniformity filter.

Finally the route reaches E through componentwise transport, replay, provenance, and local naming. Since the only visible path to E runs through R, W, D, B, H, C, P, N, the handoff cannot manufacture quotient equality, countable choice, selected limits, or completeness outside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ read exposes RegSeqRat source rows, StreamName finite windows, DyadicRatCore tolerance rows, a basis-entourage row, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any Real-facing consumer can use the packet.

Independence witness: regular-cauchy-uniformity, regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source row, finite windows, tolerance ledger, basis-entourage generator, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyEntourageBasis}^{\uparrow}$ packet is generated from a RegSeqRat source row, a StreamName finite-window row, a DyadicRatCore tolerance row, a basis-entourage row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3805.1, Definition 3805.2, and Definition 3805.3 seed the finite basis-level route between regular-Cauchy windows and the Real endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEntourageBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, quotient equality of real presentations, selected limits, countable choice, ambient uniformity completeness, or Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, basis-entourage exactness, classifier stability, real-seal nonescape, and Real handoff for the displayed packet.

Chapter 3806

A Concrete Naming Certificate for $\mathsf{TriggerEnergyPartition}^{\uparrow}$

The $\mathsf{TriggerEnergyPartition}^{\uparrow}$ packet realizes the trigger-energy theorem of Definition 9462.1584 as finite BEDC NameCert data. It records support rows, an energy ledger, a partition row, codon-lift accounting, transport, replay, provenance, and local naming for the energy-minimal trigger surface. It is a sibling of the reliability packet in chapter 3583: $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumes energy-weighted reliability rows, while $\mathsf{TriggerEnergyPartition}^{\uparrow}$ isolates the partition certificate for equality in the energy lower bound.

Definition 3806.1 (Trigger-energy partition carrier)**.** *A* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, L, E, Q, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite support row over the four trigger coordinates $\Omega = \{s_1, f_1, s_2, f_2\}$.*
2. *L is the trigger-selection row naming the displayed quotient-level trigger and its codon-level lift choices.*
3. *E is the energy-ledger row recording the sum of the support sizes of the selected external codons.*
4. *Q is the partition row certifying that the selected supports are pairwise disjoint and cover Ω exactly in the energy-minimal case.*
5. *K is the codon-lift row recording the finite f_3-lift multiplicity for each selected support block.*
6. *H records componentwise* hsame *transport for support, selection, energy, partition, codon-lift, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an energy-minimal trigger request through S, L, E, Q, K, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{TriggerEnergyPartition}^{\uparrow}$*,* $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*, the source theorem Definition 9462.1584,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$ *row.*

The carrier contains no biological stochastic model, empirical likelihood claim, hidden codon semantics, unbounded trigger enumeration, or external partition oracle.

Lean4 ✓ `BEDC.Derived.TriggerEnergyPartitionUp.TriggerEnergyPartitionTasteGate single carrier alignment`

Theorem 3806.2 (Trigger-energy partition NameCert obligations)**.** *For every accepted* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *carrier* $T = (S, L, E, Q, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, finite support exposure through* S*, trigger-selection exposure through* L*, energy-ledger exactness through* E*, partition exactness through* Q*, codon-lift accounting through* K*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3806.1. The support row S and selection row L expose the finite trigger family, while E records the total support-size ledger.

Partition exactness reads the equality case of Definition 9462.1584. When the energy ledger has value 4, the partition row Q records that the selected supports cover the four coordinates exactly once; conversely, the displayed partition row forces the energy ledger to count four coordinate appearances.

The codon-lift row K is finite because it records only the displayed f_3 choices over the selected support blocks. Transport and replay preserve S, L, E, Q, K componentwise, and provenance plus local naming keep the certificate tied to the source theorem and the energy-biased reliability sibling. No stochastic, empirical, hidden-semantic, or unbounded-search row is exported. □

Theorem 3806.3 (Trigger-energy partition reliability handoff)**.** *Every energy-minimal trigger read consumed by* $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ *factors through the finite* $\mathsf{TriggerEnergyPartition}^{\uparrow}$ *route*

$$S, L, E, Q, K, H, C, P, N.$$

The reliability consumer receives the support family, energy ledger, partition certificate, and codon-lift count as finite rows, not as an external enumeration or biological probability model.

Proof. Start with an accepted trigger-energy partition carrier. The finite support row S names the coordinate supports, and L names the selected quotient trigger together with its codon-level lift surface.

The energy ledger E and partition row Q then expose the equality case in the lower bound: every coordinate of Ω is covered exactly once. The codon-lift row K records the finite lift multiplicities attached to those blocks.

An $\mathsf{EnergyBiasedTriggerReliability}^{\uparrow}$ consumer may read these rows as inputs for its weighted reliability formula, but the transport, replay, provenance, and local naming rows keep the handoff inside the displayed finite certificate. Thus the consumer obtains no external trigger enumeration, empirical likelihood, or hidden codon semantics. □

Falsifiable prediction: Every accepted $\mathsf{TriggerEnergyPartition}^{\uparrow}$ read exposes finite support, trigger-selection, energy-ledger, partition, codon-lift, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any energy-biased reliability consumer can use an energy-minimal trigger partition.

Independence witness: energy-biased-trigger-reliability, six-boundary-tile-coverage, window-coverage-trigger-algebra: $\mathsf{TriggerEnergyPartition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the energy equality partition row and codon-lift accounting rather than only coverage, algebra, or reliability rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TriggerEnergyPartition}^{\uparrow}}$**.**

Constructive story: A TriggerEnergyPartition$^{\uparrow}$ packet is generated from finite support rows, trigger-selection rows, an energy ledger, a partition certificate, codon-lift accounting, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{TriggerEnergyPartition}^{\uparrow}}$, seed)

Scope closed: Definition 3806.1, Definition 3806.2, and Definition 3806.3 seed the finite partition surface for equality in the trigger-energy lower bound.

Formal verification: FormalStatus(NameCert$_{\text{TriggerEnergyPartition}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biological stochastic semantics, empirical likelihood, hidden codon semantics, or unrestricted trigger enumeration.

Upgrade path: Obligation closure requires checked carrier admission, support exposure, energy-ledger exactness, partition exactness, codon-lift accounting, reliability handoff, Nontrivial TriggerEnergyPartitionUp, and FieldFaithful TriggerEnergyPartitionUp for BEDC.Derived.TriggerEnergyPartitionUp.

Chapter 3807

A Concrete Naming Certificate for $\mathsf{ContinuousSupNorm}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ContinuousSupNorm}^{\uparrow}$ packet records the compact-source uniform metric for bounded continuous real-valued functions as finite BEDC data. It reads the compact metric source of chapter 129, the continuous-map row of chapter 130, the real-valued uniform Cauchy route of chapter 1830, the uniform-limit metric handoff of chapter 1764, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the finite sup-norm surface needed by compact continuous-function consumers without importing a completed function space, host function equality, or a global supremum operator outside the displayed ledger.

Definition 3807.1 (Continuous sup-norm carrier)**.** *A* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, R, O, M, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *packet.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for a real-valued function on K.*
3. *R is the pointwise* $\mathsf{Real}^{\uparrow}$ *difference and absolute-value seal for inspected source windows.*
4. *O is the finite oscillation ledger, recording the inspected cover cells, source windows, and value-oscillation bounds.*
5. *M is the displayed modulus row that transports continuous-map control from source-window refinement to real-value tolerance.*
6. *B is the supremum-bound witness, recording a finite bound row for the displayed oscillation and pointwise-difference ledger.*
7. *H records componentwise* hsame *transport for the compact source, continuous-map row, Real difference row, oscillation ledger, modulus row, bound witness, replay row, provenance, and local name.*
8. *C records the displayed* Cont *replay route from a sup-norm request through $K, F, R, O, M, B, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{UniformLimitMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed compact source, continuous-map row, pointwise Real difference row, finite oscillation and modulus ledgers, supremum-bound witness, structural transport, replay, provenance, and local naming. It has no coordinate for a completed Banach space, quotient equality of functions, an ambient supremum over all host points, a choice-selected extremal point, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContinuousSupNormUp.ContinuousSupNormTasteGate single carrier alignment`

Theorem 3807.2 (ContinuousSupNorm NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *surface consists exactly of compact metric source admission through* K*, continuous-map graph and pointwise-continuity admission through* F*, pointwise Real difference sealing through* R*, finite oscillation exposure through* O*, modulus transport through* M*, finite sup-bound exposure through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3807.1. The tuple displays the compact source row, continuous-map row, Real difference row, finite oscillation ledger, modulus row, finite bound witness, transport, replay, provenance, and local naming.

The compact source and continuous-map rows determine the only function values that may be inspected. The Real difference row and finite oscillation ledger compare those displayed values, while the modulus row keeps the comparison tied to the same compact source windows.

The finite bound witness records the sup-norm-facing estimate without selecting an ambient supremum. Since the remaining structural rows only transport, replay, source, and name the same displayed data, the listed rows exhaust the local NameCert obligations. □

Theorem 3807.3 (ContinuousSupNorm compact-uniform handoff)**.** *Every compact-continuous norm consumer of an accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *packet reads the norm route through compact metric source rows, continuous-map rows, finite oscillation rows, modulus rows, Real difference seals, finite sup-bound witnesses,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data. The route may be consumed by compact-uniform and uniform-modulus surfaces such as chapter 388, chapter 369, and chapter 130, but it exports no host function equality, open-cover compactness, quotient real equality, selected supremum, or ambient completeness.*

Proof. Use Definition 3807.2. It orders the compact source row, continuous-map row, Real difference row, oscillation ledger, modulus row, and finite bound witness inside one accepted packet.

The compact and continuous rows expose only finite inspection windows and displayed graph reads. The modulus and oscillation rows transport those reads to a common Real tolerance, and the bound witness records the finite norm-facing estimate.

Thus every compact-uniform consumer receives the displayed BEDC route. The carrier has no coordinate for a host function object, open-cover compactness object, quotient Real equality, selected supremum, or ambient completeness theorem. □

Theorem 3807.4 (ContinuousSupNorm Real seal nonescape)**.** *Every Real-valued bound exported by an accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier passes through the pointwise* $\mathsf{Real}^{\uparrow}$ *difference seal* R *and the finite sup-bound witness* B*. Consequently the exported sup-norm data can depend only on compact source windows, continuous-map rows, finite oscillation and modulus ledgers, displayed Real comparison rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Begin with the accepted carrier. The Real-facing coordinate is R, and the norm-facing bound coordinate is B.

Any bound read first uses the compact source K and continuous-map row F, then the oscillation and modulus rows O, M, and then the Real difference seal R. Only after those displayed rows are fixed can the finite bound witness B export a sup-norm-facing value.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. Since no other Real-valued coordinate appears in the carrier, the exported bound cannot leave the displayed Real-seal surface. □

Theorem 3807.5 (Continuous sup-norm uniform-limit metric export)**.** *Every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$ *handed to* $\mathsf{UniformLimitMetric}^{\uparrow}$ *or* $\mathsf{UniformSequentialLimit}^{\uparrow}$ *exports only compact metric source rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, finite oscillation and modulus ledgers, pointwise* $\mathsf{Real}^{\uparrow}$ *difference seals, finite sup-bound witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}^{\uparrow}}$ *row. The export route has no coordinate for host function equality, a selected supremum point, a completed function-space quotient, a choice-selected limiting function, or ambient compactness outside the displayed compact metric source.*

Proof. Project the carrier and apply Definition 3807.3. The compact source, continuous-map row, oscillation ledger, modulus row, Real difference seal, and finite bound witness already form the only compact-uniform route available to a consumer.

Use Definition 3807.4 for the Real-facing boundary. It forces every exported bound through the displayed pointwise difference seal and finite sup-bound witness.

Finally compose this boundary with the Cauchy-control handoff of Definition 3807.6. The resulting uniform-limit metric export reads exactly $K, F, O, M, R, B, H, C, P, N$ and nothing else.

Since the carrier has no coordinate for host function equality, selected supremum data, completed function-space quotients, selected limiting functions, or ambient compactness beyond K, both $\mathsf{UniformLimitMetric}^{\uparrow}$ and $\mathsf{UniformSequentialLimit}^{\uparrow}$ receive only the finite BEDC route. □

Theorem 3807.6 (Continuous sup-norm Cauchy control)**.** *Every accepted* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *carrier* $S = (K, F, R, O, M, B, H, C, P, N)$ *that is read as a sup-norm Cauchy row supplies uniform finite-window Cauchy control over the compact source. For each displayed tolerance request, the accepted read factors through the finite route*

$$K, F \longmapsto O, M \longmapsto R, B \longmapsto H, C, P, N.$$

Thus every inspected pair of function rows is controlled by the same compact-source window ledger and the same RealUp difference-and-bound witness.

Proof. Project the carrier of Definition 3807.1. The compact source K and continuous-map row F determine which source windows and pointwise Real reads can be inspected.

The finite oscillation row O and modulus row M then give a single finite window ledger for the tolerance request. The Real difference row R and supremum-bound witness B read all inspected pointwise differences through that ledger.

Finally, H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores host function equality, a completed function space, or an ambient supremum operator, the Cauchy control available to a consumer is exactly the displayed uniform finite-window control. □

Theorem 3807.7 (Continuous sup-norm uniform-limit handoff)**.** *Every* ContinuousSupNorm$^{\uparrow}$ *Cauchy-control read accepted by* $S = (K, F, R, O, M, B, H, C, P, N)$ *may be handed to the* UniformLimitMetric$^{\uparrow}$ *packet only through the displayed compact source, continuous-map row, finite oscillation and modulus ledgers, RealUp pointwise-difference seal, supremum-bound witness, componentwise transport, replay, provenance, and local naming rows. The same sup-norm comparison row is consumed by* UniformSequentialLimit$^{\uparrow}$ *through chapter 3874 when a uniform tail schedule is read as one tolerance-controlled Real-limit route. The handoff exports no additional completeness, quotient-function equality, or selected limit object.*

Proof. Begin with the Cauchy-control route of Definition 3807.6. It exposes the compact source and continuous-map row before the oscillation, modulus, Real difference, and bound rows are consumed.

The UniformLimitMetric$^{\uparrow}$ consumer can therefore read only the finite route already present in $K, F, O, M, R, B, H, C, P, N$. The package provenance P records the connection with RealFunctionUniformCauchy$^{\uparrow}$ and UniformLimitMetric$^{\uparrow}$, while N names only the local sup-norm packet.

There is no remaining coordinate for a host completion theorem, a quotient of functions, or a choice-selected limiting function. Hence the handoff is exactly the finite BEDC route displayed by the carrier. □

Closure status for NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$**.**

Constructive story: A ContinuousSupNorm$^{\uparrow}$ packet is generated from a compact metric source, continuous-map row, pointwise RealUp difference seal, finite oscillation and modulus ledgers, supremum-bound witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$, seed)

Scope closed: Definition 3807.1, Definition 3807.6, Definition 3807.5, and Definition 3807.7 seed the compact-source sup-norm carrier, Cauchy-control route, uniform-limit metric export, and uniform-limit handoff for bounded continuous real-valued functions.

Formal verification: FormalStatus(NameCert$_{\text{ContinuousSupNorm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach-space completeness, quotient equality of functions, host supremum existence, selected limit functions, Lean verification, or bridge checking for ContinuousSupNorm$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite oscillation coverage, modulus transport, RealUp difference sealing, supremum-bound exactness, Cauchy-control routing, uniform-limit handoff, replay stability, and ledger non-escape for BEDC.Derived.ContinuousSupNormUp.

Chapter 3808

A Concrete Naming Certificate for $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$

The $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ packet realizes the bounded redex-frontier route needed by the MetaCIC confluence open problem of section 9386.3. It sits beside the residual-substitution boundary of chapter 2181, the closed parallel-diamond premise ledger of chapter 2325, the local diamond witness of chapter 3595, and the critical-pair boundary of chapter 2212. The packet records a finite bounded route for residual substitution without converting that route into full closed confluence.

Falsifiable prediction: Any claimed full parallel diamond for MetaCIC closed terms that bypasses the displayed $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ route through finite redex-frontier rows, residual-route rows, bounded checker rows, local joins, obstruction retention, transport, replay, provenance, and local NameCert data fails the packet boundary.

Independence witness: metacic-residual-substitution-boundary, closed-parallel-diamond-premise-ledger, metacic-local-diamond-witness, metacic-confluence-critical-pair-boundary: none is bijective to $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ because this packet jointly carries bounded redex-frontier data, residual-route accounting, bounded checking, local joins, obstruction retention, transport, replay, provenance, and local naming as one finite frontier.

Definition 3808.1 (Bounded residual substitution frontier carrier)**.** *A* $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, S, B, J, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the finite redex-frontier row, listing the inspected beta-redex positions and their bounded residual descendants.*
2. *S is the residual-route row, recording the displayed substitution route and binder-depth ledger for the bounded window.*
3. *B is the bounded checker row, reading only finite closedness, variable-escape, and branch-compatibility checks.*
4. *J is the local join row, storing the common-reduct candidate produced inside the bounded frontier.*

5. *O is the obstruction row, retaining the absent endpoints: full* BetaParallelSubstitutionCompatibility*, premise-free closed parallel diamond, arbitrary closed Church–Rosser, and unchecked subject-reduction discharge.*
6. *H records componentwise* hsame *transport for the redex frontier, residual route, bounded checker, local join, obstruction, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay from a confluence-frontier request through R, S, B, J, O, H, P, N.*
8. *P is the* Pkg *provenance over* BoundedResidualSubstitutionFrontier$^{\uparrow}$*,* MetaCICResidualSubstitutionBoundary$^{\uparrow}$*,* ClosedParallelDiamondPremiseLedger$^{\uparrow}$*,* MetaCICLocalDiamondWitness$^{\uparrow}$*,* MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$ BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* NameCert$_{\text{BoundedResidualSubstitutionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for full closed confluence, unrestricted residual-substitution compatibility, a hidden substitution theorem, evaluator equality, or kernel consistency.

Lean4 ✓ `BEDC.Derived.BoundedResidualSubstitutionFrontierUp.BoundedResidualSubstitutionFrontierTasteGate`
`single carrier alignment`

Theorem 3808.2 (Bounded residual substitution frontier obligations)**.** *For every accepted* BoundedResidualSubstitu *carrier $F = (R, S, B, J, O, H, C, P, N)$, the local* NameCert$_{\text{BoundedResidualSubstitutionFrontier}^{\uparrow}}$ *surface consists exactly of finite redex-frontier exposure through R, residual-route exposure through S, bounded checking through B, local join exposure through J, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

The confluence-facing read is finite:

$$R, S, B \longmapsto J, O \longmapsto H, C, P, N.$$

It does not escape to full closed confluence; any consumer still sees the residual-substitution and closed parallel-diamond premise boundaries as displayed inputs.

Proof. Project the carrier of Definition 3808.1. The rows $R, S, B, J, O, H, C, P, N$ are the whole packet, so every local obligation is read from one displayed coordinate.

The bounded route begins with R, S, B. The redex-frontier row lists the inspected positions, the residual-route row records the substitution accounting, and the bounded checker row verifies only the finite window available to the packet.

The local join row J is therefore downstream of bounded evidence. The obstruction row O records that full substitution compatibility, a premise-free closed parallel diamond, arbitrary closed Church–Rosser, and unchecked subject-reduction discharge are not exported.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for a global confluence theorem or hidden substitution theorem, the displayed obligations cannot escape their bounded frontier. □

Lean4 ✓ `BEDC.Derived.BoundedResidualSubstitutionFrontierUp.BoundedResidualSubstitutionFrontierNamecert(`

Theorem 3808.3 (MetaCIC residual-substitution candidate support)**.** *Any candidate-mediated strong-normalization or confluence consumer using an accepted* BoundedResidualSubstitutionFrontier$^{\uparrow}$ *carrier $F = (R, S, B, J, O, H, C, P, N)$ must expose $R, S, B, J, O, H, C, P, N$ before projecting a common-reduct candidate. In particular, it must cite the residual-substitution boundary, the closed*

parallel-diamond premise ledger, the local diamond witness, and the critical-pair boundary before treating J as a confluence-facing candidate.

The read presents only bounded candidate support. The obstruction row O remains visible as the refusal row for full confluence, unrestricted substitution compatibility, premise-free closed parallel diamond, and unchecked subject-reduction discharge.

Proof. Project the carrier of Definition 3808.1. The rows $R, S, B, J, O, H, C, P, N$ are the complete support surface available to the consumer.

Apply Definition 3808.2. It orders the finite redex-frontier row R, residual-route row S, and bounded checker row B before the local join row J can be read.

The candidate projection therefore remains mediated by the sibling boundaries named in the provenance row: residual substitution supplies the boundary of the route, the closed parallel-diamond premise ledger supplies the premise account, the local diamond witness supplies the bounded join, and the critical-pair boundary supplies the finite obstruction interface.

Keep O in the exported packet. Since O records exactly the missing full-confluence and unrestricted-substitution data, the common-reduct candidate is a bounded local candidate rather than a proof of the unrestricted MetaCIC confluence claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$ packet is generated from finite redex-frontier rows, residual-route accounting, bounded checker rows, local joins, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3808.1 and Definition 3808.2 seed the finite bounded residual-substitution frontier for MetaCIC confluence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full BetaParallelSubstitutionCompatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, redex-frontier coverage, residual-route exactness, bounded-checker soundness, local-join exposure, obstruction retention, transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.BoundedResidualSubstitutionFrontierUp.

Chapter 3809

A Concrete Naming Certificate for $\mathsf{LocatedRealContinuity}^{\uparrow}$

Following Bishop and Bridges, located continuity is recorded as finite precision-response data over located real names, not as a host function on completed reals. The $\mathsf{LocatedRealContinuity}^{\uparrow}$ packet sits over the located-real carrier of **??**, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located modulus surface of chapter 3406. It gives the L10 Real and Completion route a named continuity consumer whose public data are precision requests and finite responses rather than quotient-real functions, trichotomy, countable choice, or ambient completeness.

Definition 3809.1 (Located real continuity carrier)**.** *A* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, D, W, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the located input row, read through the* $\mathsf{LocatedReal}^{\uparrow}$ *surface.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for requested input and output tolerances.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window used to inspect the located input at the requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the inspected window as regular rational data.*
5. *M is the modulus-response row, returning a finite precision response for the displayed input tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the output name, visible only after X, D, W, R, M have been displayed.*
7. *H records componentwise* hsame *transport for the located input, dyadic ledger, stream window, regular readback, modulus response, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located precision request through $X, D, W, R, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealContinuity}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRealModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host real function, quotient-real equality test, trichotomy oracle, countable-choice schedule, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.LocatedRealContinuityUp.LocatedRealContinuityTasteGate single carrier alignment`

Theorem 3809.2 (Located real continuity modulus stability)**.** *Let* $K = (X, D, W, R, M, E, H, C, P, N)$ *and* $K' = (X', D', W', R', M', E', H', C', P', N')$ *be accepted* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carriers. If the located input rows, dyadic tolerance ledgers, finite windows, and regular-rational readbacks agree componentwise under* hsame*, then the modulus-response rows* M *and* M' *answer the same displayed dyadic tolerance requests, and every* $\mathsf{Real}^{\uparrow}$ *seal consumer reaches the same response route*

$$X, D, W, R \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

Thus modulus-response stability is a finite packet property: it is preserved by hsame *transport and by the displayed dyadic tolerance rows, without reading quotient-real equality, trichotomy, countable choice, selected representatives, host function extensionality, or ambient completeness.*

Proof. Project the two accepted carriers from Definition 3809.1. The only precision source rows are X, D, W, R and their transported counterparts.

Use the assumed componentwise hsame agreement on those rows. The dyadic ledger D fixes the tolerance request, the window W fixes the finite observations, and R fixes the regular-rational readback. Therefore the accepted modulus rows M and M' respond to the same displayed precision request.

The seal row is downstream of the response row. Since E and E' can be consumed only after the displayed route through the input, tolerance, window, readback, and response rows, any $\mathsf{Real}^{\uparrow}$ consumer reads the same finite route before seeing a real-name seal.

The excluded analytic principles have no carrier coordinate in either packet. They cannot alter the transported modulus response or add a second continuity channel. □

Theorem 3809.3 (Located real continuity NameCert obligations)**.** *For an accepted* $\mathsf{LocatedRealContinuity}^{\uparrow}$ *carrier* $K = (X, D, W, R, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission, classifier stability for the located input and precision-response rows, componentwise transport, the continuity-modulus ledger, terminal real-seal routing, package provenance, and local naming. The obligation surface does not include quotient reals, trichotomy, countable choice, selected representatives, host function extensionality, or ambient completeness.*

Proof. Carrier admission is projection from Definition 3809.1. The accepted packet displays exactly the located input, dyadic tolerance ledger, finite window, regular readback, modulus response, real seal, transport, replay, provenance, and local naming rows.

Classifier stability is supplied by the same inventory. An accepted continuity read must keep the rows X, D, W, R, M attached to one displayed precision route, so changing any of these rows changes the carrier rather than producing a hidden function on completed reals.

Transport and replay are the rows H and C. They preserve the displayed route through the dyadic tolerance and modulus-response ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally P and N source and name the same finite packet. Since quotient reals, trichotomy, countable choice, selected representatives, host function extensionality, and ambient completeness have no carrier rows, they are not local NameCert obligations for $\mathsf{LocatedRealContinuity}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealContinuity}^{\uparrow}$ packet is generated from a located input row, a dyadic precision ledger, finite StreamName windows, RegSeqRat readback, a modulus-response row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3809.1, Definition 3809.2, and Definition 3809.3 seed the finite located-continuity route from precision requests to RealUp seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions on completed reals, quotient-real equality, trichotomy, countable choice, selected representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, transport, continuity-modulus ledger rows, real-seal routing, provenance, and local naming for BEDC.Derived.LocatedRealContinuityUp.

Chapter 3810

A Concrete Naming Certificate for CellularSubstrateClassifierBoundary$^{\uparrow}$

The CellularSubstrateClassifierBoundary$^{\uparrow}$ packet realizes the cellular-automaton substrate reading of chapter 9390 as a finite classifier-boundary carrier. It binds the local transition table, orbit replay, pattern-classifier tag row, closed-observation manifest, and refusal boundary into one NameCert surface. The packet is designed to accept cellular substrate observations only when the classifier that names the observed pattern is displayed as data, so no untagged orbit replay can masquerade as a closed observation.

Before its manifest and closed-observation rows are exported, the packet consumes the rule-table and finite-orbit rows of chapter 934, the substrate carrier of chapter 942, and the classifier comparison surface of chapter 945. These sibling routes determine which local transition, orbit replay, and pattern tag rows may enter the boundary carrier.

The classifier-boundary packet also consumes the observation-window discipline of chapter 2371 and the classifier-row surface of chapter 945 before its Rule 110 handoff row is exposed. Thus the closed-observation boundary and the classifier lattice edge are both visible before a universal-substrate read can use the packet.

Definition 3810.1 (CellularSubstrateClassifierBoundary carrier)**.** *A* CellularSubstrateClassifierBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, O, G, M, A, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the local cellular transition row, including the finite alphabet, neighborhood window, and rule table.*
2. *O is the orbit replay row generated by iterating L over the stated initial row and finite time window.*
3. *G is the pattern-classifier tag row that assigns each inspected local pattern to a finite tag family.*
4. *M is the manifest row specifying the accepted observation windows, expected replay checks, and public audit fields.*
5. *A is the closed-observation admission row, requiring the replayed orbit and the classifier tags to agree on every observed window.*

6. *R is the Rule 110 substrate handoff row when the local rule is read through the cellular universal-substrate exhibit.*
7. *H records componentwise* hsame *transport for the local rule, orbit replay, classifier tags, manifest, admission row, Rule 110 handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from local transition data through orbit replay, pattern classification, closed-observation admission, and any Rule 110 handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$*, cellular-automaton substrate reading, Rule 110 substrate, classifier tags, closed observation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unclassified orbit, a detached semantic interpretation of a pattern, an induction principle over all time steps, or an observation that bypasses the manifest and classifier rows.

```
Lean4 ✓  BEDC.Derived.CellularSubstrateClassifierBoundaryUp.CellularSubstrateClassifierBoundaryTasteGate
single carrier alignment
```

Theorem 3810.2 (CellularSubstrateClassifierBoundary NameCert obligations)**.** *For every accepted* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *carrier* $B = (L, O, G, M, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the local transition table, orbit replay, classifier tag row, manifest, closed-observation admission row, optional Rule 110 handoff, transport, replay, provenance, and local naming;*
2. *local-rule replay, requiring every accepted observation to display L and O before any classifier-boundary claim is consumed;*
3. *classifier-boundary exposure, requiring the tag row G to be present for each observed pattern window;*
4. *manifest closure, requiring the row M and admission row A to bind replayed orbits and classifier tags into one finite observation record;*
5. *Rule 110 handoff discipline, requiring R when the packet is read through the universal cellular substrate route;*
6. *classifier transport, requiring all accepted comparisons to use only H, C, P, and N;*
7. *ledger refusal, excluding untagged orbit replay, detached semantic interpretation, all-time induction, and observations outside the manifest and classifier rows.*

No classifier-boundary obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3810.1. The accepted packet displays exactly the local rule, orbit replay, classifier tag row, manifest, admission row, Rule 110 handoff, and structural rows.

Local-rule replay comes first. The orbit row O is generated from the transition row L, so no accepted observation can be read without displaying the finite rule table and the replay window.

Classifier-boundary exposure and manifest closure then bind the orbit to a tag discipline. The tag row G names the inspected patterns, while M and A require those tags to agree with the manifest's closed-observation checks.

Finally the Rule 110 handoff and structural rows preserve the same route. The optional row R names the universal cellular substrate path when present, and H, C, P, N transport, replay, source, and name the local carrier. Since untagged or detached observations have no row in the packet, they cannot become NameCert obligations. □

Falsifiable prediction: A packet with a local transition row and orbit replay but no pattern-classifier tag row is rejected by the classifier-boundary carrier.

Independence witness: cellular-automaton-substrate-reading, metacicredexfrontier, rule110substrate, binaire, apophaticname: the packet is not bijective to these siblings because it binds local cellular transition, orbit replay, classifier tags, closed-observation admission, Rule 110 handoff, and classifier-boundary refusal in one finite NameCert route.

Theorem 3810.3 (CellularSubstrateClassifierBoundary closed-observation handoff)**.** *Every closed cellular observation accepted by a* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *packet reads the route*

$$L, O \longmapsto G \longmapsto M, A \longmapsto R \longmapsto H, C, P, N.$$

The route starts with a finite local transition rule and orbit replay, passes through the pattern-classifier tag row, binds the result to a manifest and closed-observation admission row, and reaches the Rule 110 handoff only as displayed substrate data. A Lean follow-up for this carrier must supply Nontrivial *and* FieldFaithful *instances in* `lean4/BEDC/Derived/CellularSubstrateClassifierBoundaryUp/Taste`

Proof. Project the accepted carrier of Definition 3810.1. The rows L and O are the only local-rule and orbit-replay rows, and G is the only classifier row attached to the inspected patterns.

Read manifest admission next. The rows M and A bind the displayed classifier tags to the closed-observation record, so an orbit replay without G cannot pass the admission surface.

The remaining rows preserve the same finite dependency chain. The Rule 110 handoff R is downstream of rule, orbit, classifier, and manifest data, while H, C, P, N transport, replay, source, and name the accepted route. Therefore every accepted observation factors through the classifier boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ packet is generated from a local cellular transition table, orbit replay, pattern-classifier tags, a closed-observation manifest, an admission row, an optional Rule110Substrate handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3810.1, Definition 3810.2, and Definition 3810.3 seed the finite classifier-boundary surface for cellular substrate observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close all-time induction, unclassified orbit semantics, a complete Rule 110 bridge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-rule replay, classifier-boundary exposure, manifest closure, Rule 110 handoff discipline, classifier transport, and ledger refusal for BEDC.Derived.CellularSubstrateClassifierBoundaryUp.

Chapter 3811

A Concrete Naming Certificate for HadamardProductSeries$^{\uparrow}$

Following Hadamard's coefficientwise product of power series and Hardy's series estimates, the HadamardProductSeries$^{\uparrow}$ packet records the finite route by which two displayed real coefficient streams are multiplied coordinate by coordinate before they are handed to the ordinary CauchyProduct$^{\uparrow}$ and RealSeries$^{\uparrow}$ convergence surfaces. It sits beside the real-series carrier of chapter 1581, the Cauchy-product certificate of chapter 464, the majorant route of chapter 399, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic coefficient ledger of chapter 291, and the terminal real seal of chapter 38. The packet names coefficientwise multiplication as finite BEDC data; it is not a host analytic-function identity, not a quotient of series, not a selected infinite sum, and not an ambient completeness theorem.

Definition 3811.1 (Hadamard product series carrier)**.** *A* HadamardProductSeries$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, R, S, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two source* RealSeries$^{\uparrow}$ *coefficient-stream rows.*
2. *W is the synchronized* StreamName$^{\uparrow}$ *window row selecting matching coefficient positions from A and B.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *coefficient-product ledger; for every displayed position in W it records the dyadic product of the two read coefficients and its tolerance budget.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row repacking the coefficientwise products as regular rational finite-window data.*
5. *S is the derived* RealSeries$^{\uparrow}$ *coefficient stream assembled from R and the synchronized windows.*
6. *M is the convergence-control row, either a displayed* WeierstrassMTest$^{\uparrow}$ *majorant for the product coefficients or a* CauchyProduct$^{\uparrow}$ *handoff consuming the same finite product ledger.*
7. *E is the terminal* Real$^{\uparrow}$ *seal exposed only after A, B, W, D, R, S, M have been replayed.*
8. *H records componentwise* hsame *transport for the two source series, the synchronized windows, dyadic products, regular readback, derived series, convergence-control row, endpoint seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from paired coefficient requests through synchronized windows, dyadic products, regular readback, the derived series, convergence control, and the endpoint seal.*
10. *P is the* Pkg *provenance over* HadamardProductSeries$^{\uparrow}$, RealSeries$^{\uparrow}$, CauchyProduct$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HadamardProductSeries}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, W, D, R, S, M, E, H, C, P, N$. It has no coordinate for host analytic continuation, quotient series equality, selected infinite sums, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 3811.2 (Hadamard product series NameCert obligations)**.** *For every accepted* HadamardProductSeries$^{\uparrow}$ *carrier $K = (A, B, W, D, R, S, M, E, H, C, P, N)$, the local* NameCert$_{\text{HadamardProductSeries}^{\uparrow}}$ *surface consists exactly of the two source* RealSeries$^{\uparrow}$ *rows, synchronized* StreamName$^{\uparrow}$ *windows,* DyadicRatCore$^{\uparrow}$ *coefficient-product ledger,* RegSeqRat$^{\uparrow}$ *readback, derived* RealSeries$^{\uparrow}$ *row, convergence-control row, terminal* Real$^{\uparrow}$ *seal, componentwise transport, displayed continuation replay, package provenance, and local naming through $A, B, W, D, R, S, M, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.HadamardProductSeriesUp.HadamardProductSeriesTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3811.1. The only source series coordinates are A and B, and coefficient requests can reach them only through the synchronized finite window row W.

The coefficient-product data are then forced through D. Row D records the dyadic product and tolerance budget for each displayed pair of coefficients, and row R repacks exactly those reads as RegSeqRat$^{\uparrow}$ data.

The derived series row S and convergence-control row M are downstream of that readback. The structural rows H, C, P, N transport, replay, source, and name the same route, while the carrier excludes host analytic identities, quotient series equality, selected infinite sums, countable choice, and ambient real completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 3811.3 (Hadamard product Cauchy-product handoff)**.** *Every* CauchyProduct$^{\uparrow}$, WeierstrassMTest$^{\uparrow}$, RealSeries$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer of an accepted* HadamardProductSeries$^{\uparrow}$ *carrier must read the finite route*

$$A, B \longmapsto W \longmapsto D \longmapsto R \longmapsto S \longmapsto M \longmapsto E$$

before consuming the derived coefficient stream or endpoint seal. The handoff therefore supplies coefficientwise-product data as synchronized finite windows and dyadic products, not as a host multiplication theorem for completed series.

Proof. Begin with a consumer request for product-series data. The two source streams are readable only as A and B, and the matching of coefficient positions is stored only in W.

The product coefficient cannot be read before the dyadic ledger D is displayed. Its regular-rational readback is R, and the derived real-series row S is assembled only from those finite product windows.

The convergence-control row M is the unique handoff to the named CauchyProduct$^{\uparrow}$ or WeierstrassMTest$^{\uparrow}$ surface, and E is downstream of that handoff. Since H, C, P, N preserve the same route and no additional analytic coordinate is present, every public consumer factors through the displayed finite coefficientwise-product path. □

Falsifiable prediction: Every accepted $\mathsf{HadamardProductSeries}^{\uparrow}$ endpoint read factors through paired $\mathsf{RealSeries}^{\uparrow}$ coefficient rows, synchronized $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ product ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyProduct}^{\uparrow}$ or $\mathsf{WeierstrassMTest}^{\uparrow}$ convergence control, provenance, replay, and local NameCert rows.

Independence witness: realseries, cauchyproduct, weierstrassmtest, regseqrat, streamname, dyadicratcore, real: RealSeriesUp carries single-stream series data without paired coefficient products; CauchyProductUp records convolution-style handoff without the synchronized coefficientwise ledger; WeierstrassMTestUp records majorants without paired coefficient multiplication; RegSeqRatUp, StreamNameUp, and DyadicRatCoreUp supply readback, windows, and dyadic products separately; RealUp supplies the endpoint seal without the product-series route. None is bijective to $\mathsf{HadamardProductSeries}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardProductSeries}^{\uparrow}$ packet is generated from two RealSeries coefficient streams, synchronized StreamName windows, DyadicRatCore coefficient-product ledgers, RegSeqRat readback, a derived RealSeries row, CauchyProduct or WeierstrassMTest convergence control, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3811.1, Definition 3811.2, and Definition 3811.3 seed the finite coefficientwise-product route from paired real-series coefficients to convergence control and terminal real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardProductSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host analytic continuation, quotient series equality, selected infinite sums, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized window exactness, dyadic coefficient-product accounting, RegSeqRat readback, RealSeries assembly, CauchyProduct or WeierstrassMTest handoff, RealUp seal routing, and local naming for BEDC.Derived.HadamardProductSeriesUp.

Chapter 3812

A Concrete Naming Certificate for $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley metric-space terminology, the $\mathsf{LipschitzContinuityHandoff}$ packet records the textbook handoff from a displayed Lipschitz bound to a uniform continuity modulus before regular-Cauchy or Real-facing consumers read the result. It is grounded in the bound-carrying $\mathsf{LipschitzMap}^{\uparrow}$ surface of chapter 410, the metric rows of chapter 57, the graph row of chapter 130, the induced modulus row of chapter 369, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the continuity handoff consumed by L10 Real and Completion routes; it does not duplicate the upstream Lipschitz map carrier or turn that carrier into a complete downstream theorem.

Definition 3812.1 (Lipschitz continuity handoff carrier)**.** *A* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, X, Y, G, B, U, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *L is the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, read through the carrier package of chapter 410.*
2. *X and Y are the source and target* $\mathsf{Met}^{\uparrow}$ *rows used by the displayed distance comparison.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row carried by the same Lipschitz packet.*
4. *B is the displayed rational bound ledger used to compare source and target distances.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row induced by the bound ledger.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row available to regular-Cauchy consumers of the mapped readings.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, consumed only after L, X, Y, G, B, U, Q have fixed the handoff.*
8. *H records componentwise* hsame *transport for the Lipschitz row, metric rows, graph, bound ledger, modulus, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a continuity request through $L, X, Y, G, B, U, Q, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted handoff packets only through $L, X, Y, G, B, U, Q, R, H, C, P, N$. It has no coordinate for host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LipschitzContinuityHandoffUp.LipschitzContinuityHandoffTasteGate single carrier alignment`

Theorem 3812.2 (Lipschitz continuity handoff NameCert obligations)**.** *For every accepted* LipschitzContinuityHand *carrier*

$$K = (L, X, Y, G, B, U, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$ *surface consists exactly of carrier exposure for the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, source and target* $\mathsf{Met}^{\uparrow}$ *rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, rational Lipschitz-bound ledger, induced* $\mathsf{UniformModulus}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3812.1. The displayed tuple contains one upstream Lipschitz row, two metric rows, one graph row, one bound ledger, one induced modulus row, one regular-rational handoff row, one Real seal row, and the structural rows H, C, P, N.

The analytic route is inherited from the upstream $\mathsf{LipschitzMap}^{\uparrow}$ carrier rather than duplicated. The source and target metric rows, graph row, and bound ledger are read from L, and the modulus row U is the displayed uniform-modulus handoff induced by that same ledger.

The downstream rows Q and R are sealed only after the finite handoff through L, X, Y, G, B, U is visible. Since the carrier has no coordinate for host function extensionality, quotient map equality, representative choice, selected limits, countable choice, differentiability, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3812.3 (Lipschitz bound to uniform modulus handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ *carrier must first read the upstream* $\mathsf{LipschitzMap}^{\uparrow}$ *row, source and target* $\mathsf{Met}^{\uparrow}$ *rows,* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, rational bound ledger, and induced* $\mathsf{UniformModulus}^{\uparrow}$ *row. Only then may it consume the regular-rational handoff row or terminal Real seal row.*

Proof. Start with an accepted handoff packet and project the route in Definition 3812.1. The only entrance to the continuity handoff is the upstream Lipschitz row L, and its metric, graph, and bound coordinates are displayed as X, Y, G, B.

The uniform modulus row U is downstream of that same bound ledger. Thus a regular-Cauchy consumer receives the modulus as a finite handoff from the Lipschitz bound rather than as a selected host-function modulus.

The rows Q and R are downstream seals. A consumer that reads either row has already exposed L, X, Y, G, B, U and the structural transport and replay rows, so it cannot bypass the route by host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LipschitzContinuityHandoff}^{\uparrow}$ packet is generated from an upstream LipschitzMap row, source and target MetricSpace rows, a ContinuousMap graph row, a rational Lipschitz-bound ledger, an induced UniformModulus row, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzContinuityHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3812.1, Definition 3812.2, and Definition 3812.3 seed the finite continuity handoff from displayed Lipschitz bounds to uniform moduli and regular-Cauchy or Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{LipschitzContinuityHandoff}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.LipschitzContinuityHandoffUp.LipschitzContinuityHandoffTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient map equality, choice of representatives, selected limits, countable choice, differentiability, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upstream Lipschitz row projection, metric and graph exposure, bound-ledger to uniform-modulus handoff, regular-rational handoff, Real-seal discipline, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.LipschitzContinuityHandoffUp.

Chapter 3813

A Concrete Naming Certificate for $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$

The $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ packet realizes the boundary-fixed disk transport reading of chapter 9463 and Definition 9463.10 as finite NameCert rows. It consumes the disk-coordinate carrier of chapter 377, the metric-read surface of chapter 3474, the fixed-boundary transport route of chapter 3500, the boundary-tangent horocycle row of chapter 3534, and the flow-style replay surface of chapter 195. The packet names a boundary shadow of transport on a displayed disk surface; it is not a host ideal-boundary quotient, not a physical cosmology, not a global flow, and not a completed hyperbolic surface.

Falsifiable prediction: Every accepted $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ read factors through displayed $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, audit/refusal, provenance, replay, and local NameCert rows before it exports a boundary-shadow reading.

Independence witness: unitdisk, hyperbolic-metric, poincare-boundary-fixed-transport, hyperbolic-horocycle, dynsystem: the UnitDisk packet records disk coordinates without metric shadowing; the HyperbolicMetric packet records metric reads without a fixed-boundary transport ledger; the PoincareBoundaryFixedTransport packet preserves a boundary anchor without a horocycle shadow; the HyperbolicHorocycle packet records boundary tangency without flow-style replay; and the DynSystem packet records routed evolution without a hyperbolic boundary anchor. None is bijective to $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$.

Definition 3813.1 (Hyperbolic boundary shadow carrier)**.** *A* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, M, B, T, Q, Y, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source and transported observations.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the boundary row naming the preserved far-end boundary reading as finite certificate data.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row that transports disk points while preserving the boundary row B.*
5. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *shadow row recording the boundary-tangent trace left by the transported disk points.*

6. *Y is the* $\mathsf{DynSystem}^{\uparrow}$ *flow-style replay row, used only to order the finite transport observations.*
7. *F is the audit/refusal row: it records that boundary-shadow reads pass through D, M, B, T, Q, Y and do not expose a host ideal-boundary quotient, physical trajectory, or global hyperbolic surface.*
8. *H records componentwise* hsame *transport for disk, metric, boundary, transport, horocycle shadow, flow replay, audit/refusal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from disk observations through metric readback, boundary preservation, boundary-fixed transport, horocycle shadowing, flow-style ordering, and audit/refusal.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, PoincareBoundaryFixed
$\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B, T, Q, Y, F, H, C, P, N$. It has no coordinate for host ideal-boundary equality, unlogged observer motion, a global flow parameter, physical spacetime, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicBoundaryShadowUp.HyperbolicBoundaryShadowTasteGate single carrier alignment`

Theorem 3813.2 (Hyperbolic boundary shadow NameCert obligations)**.** *For every accepted* HyperbolicBoundarySh
carrier $S = (D, M, B, T, Q, Y, F, H, C, P, N)$, the local $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric readback, boundary preservation, boundary-fixed transport, horocycle-shadow exposure, flow-style replay, audit/refusal exactness, componentwise transport, displayed continuation, package provenance, and local naming through $D, M, B, T, Q, Y, F, H, C, P, N$.*

Every boundary-shadow read from the packet factors through

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{PoincareBoundaryFixedTransport}^{\uparrow} \longmapsto \mathsf{HyperbolicHorocycle}^{\uparrow} \longmapsto \mathsf{DynSystem}^{\uparrow}$$

together with the boundary, audit/refusal, and structural BHist, hsame, Cont, Pkg, NameCert *rows.*

Proof. Project the accepted carrier. The disk surface is row D, and the hyperbolic metric readback over that surface is row M.

The boundary information is then forced through B and T. Row B names the preserved far-end boundary reading, while row T records transport that keeps that boundary row fixed in the sense of Definition 9463.10.

The shadow-specific content is row Q, read only after the disk, metric, boundary, and fixed-transport rows have been displayed. Row Y orders the finite observations as a flow-style replay without adding a global clock.

The audit/refusal row F and the structural rows H, C, P, N exhaust the public surface. Since no carrier coordinate stores host ideal-boundary equality, unlogged observer motion, a global flow parameter, or physical spacetime, the listed rows are exactly the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ packet is generated from UnitDisk coordinates, HyperbolicMetric readback, a preserved boundary row, PoincareBoundaryFixedTransport, a HyperbolicHorocycle shadow row, DynSystem flow-style replay, audit/refusal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicBoundaryShadow}^{\uparrow}}$, seed)

Scope closed: Definition 3813.1 and Definition 3813.2 seed the finite boundary-shadow route through disk coordinates, metric readback, preserved boundary transport, horocycle shadowing, flow-style replay, audit/refusal, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicBoundaryShadow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host ideal-boundary quotient equality, physical spacetime, unlogged observer motion, global hyperbolic flows, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk placement, metric readback, boundary preservation, boundary-fixed transport, horocycle-shadow exactness, flow replay, audit/refusal coverage, and local naming for BEDC.Derived.HyperbolicBoundaryShadowUp.

Chapter 3814

A Concrete Naming Certificate for $\mathsf{LocatedUniformSpace}^{\uparrow}$

Following Nachbin, Kelley, and Bishop-style located uniform structures, the $\mathsf{LocatedUniformSpace}^{\uparrow}$ packet records directed entourage data together with a finite locatedness ledger for deciding when a displayed point, filter, or net read has entered a named uniform neighbourhood. It sits between the directed-uniform carrier of chapter 2334, the symmetric uniform interface of chapter 339, the filter route of chapter 333, and the net route of chapter 589. Its constructive readback uses the finite window and rational ledgers of chapter 126, chapter 291, chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 3814.1 (LocatedUniformSpace carrier)**.** *A* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (U, D, A, W, T, F, N, H, C, P, Q)$$

with the following displayed rows:

1. *U is the directed entourage row, read through the* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *naming surfaces.*
2. *D is the finite directedness ledger, recording refinement, composition, and symmetry requests only as displayed entourage rows.*
3. *A is the located apartness row, recording when a point or finite window is apart from the complement of a displayed entourage.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row for finite point, net, and filter reads.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to measure the displayed locatedness request.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *and* $\mathsf{CauchyNet}^{\uparrow}$ *handoff row.*
7. *N is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rationally named tolerance witnesses.*
8. *H is the componentwise* hsame *transport row.*
9. *C is the displayed* Cont *replay row.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformSpace}^{\uparrow}$, $\mathsf{QuasiUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$ *row.*

No carrier coordinate stores a completed uniform space, a quotient filter, a selected limit, or a host locatedness decision.

Lean4 ✓ `BEDC.Derived.LocatedUniformSpaceUp.LocatedUniformSpaceTasteGate single carrier alignment`

Theorem 3814.2 (LocatedUniformSpace NameCert obligations)**.** *For every accepted* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier* $L = (U, D, A, W, T, F, N, H, C, P, Q)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, directed entourage access through* U, D*, located apartness access through* A*, finite observation windows through* W*, dyadic tolerance access through* T*, Cauchy filter and Cauchy net handoff through* F*, regular rational readback through* N*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* Q*.*

Proof. Project the carrier of Definition 3814.1. The entourage, directedness, locatedness, window, tolerance, Cauchy, and rational-readback rows are exactly U, D, A, W, T, F, N.

The structural rows H, C, P, Q transport, replay, source, and name the same finite route. Since the carrier has no further coordinate, a consumer cannot replace this route by a host uniform space, a quotient filter, a selected limit, or an undecorated locatedness predicate. □

Theorem 3814.3 (LocatedUniformSpace Cauchy-filter route)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *consumer read accepted by a* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier factors through*

$$U, D, A \longmapsto W, T \longmapsto F \longmapsto N, H, C, P, Q.$$

The Cauchy handoff may inspect only the displayed directed entourage ledger, located apartness row, finite observation windows, dyadic tolerance ledger, and regular rational readback.

Proof. Start with a Cauchy-facing consumer. The carrier exposes Cauchy data only at row F, so the consumer must first read the entourage and locatedness rows that determine which finite windows are admissible.

The rows W and T then display the finite observations and dyadic tolerances used by the Cauchy request. The regular readback row N records the rational witness visible to the consumer.

Transport, replay, provenance, and local naming are supplied only by H, C, P, Q. Thus the route is finite and excludes quotient filters, selected subnet limits, and ambient completeness. □

Theorem 3814.4 (LocatedUniformSpace entourage-location obligation)**.** *For every accepted* $\mathsf{LocatedUniformSpace}$ *carrier*

$$L = (U, D, A, W, T, F, N, H, C, P, Q),$$

each located entourage request must expose the directed entourage row U*, directedness ledger* D*, located apartness row* A*, finite observation-window row* W*, and dyadic tolerance row* T *before any* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *handoff is available. The request may not enter* F *from an unnamed entourage, a host locatedness predicate, a quotient-filter class, a selected limit, or an ambient completeness principle.*

Proof. Project the carrier rows of Definition 3814.1. The only coordinates that can answer an entourage-location request before Cauchy handoff are U, D, A, W, T.

The directedness ledger D refines requests only as displayed entourage rows over U, while A records located apartness only for the displayed point or window. The rows W and T then provide the finite observation and dyadic tolerance data needed to state the request.

Since F is downstream of these five rows, a Cauchy consumer can be admitted only after the located entourage request has passed through them. No excluded host object is a coordinate of L, so none can supply a second route into F. □

Theorem 3814.5 (LocatedUniformSpace filter-net handoff exactness)**.** *For every accepted* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *carrier*

$$L = (U, D, A, W, T, F, N, H, C, P, Q),$$

every $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyNet}^{\uparrow}$ *handoff through* F *is exact only along the finite route*

$$U, D, A, W, T \longmapsto F \longmapsto N, H, C, P, Q.$$

The packet therefore exports Cauchy-facing data as the displayed located-uniform handoff, not as quotient-filter equality, a selected subnet, a completed uniform-space theorem, separated reflection, or host locatedness.

Proof. Begin with the handoff row F. By Definition 3814.3, a Cauchy-facing read reaches F only after the directed entourage, locatedness, finite-window, and tolerance rows have been read.

Apply the entourage-location obligation. It prevents F from being entered by an unnamed entourage or external locatedness decision, so the handoff is tied to the displayed U, D, A, W, T rows.

Finally read N, H, C, P, Q. These rows supply rational readback, componentwise transport, displayed replay, provenance, and local naming for the same route. Because the carrier has no coordinate for quotient filters, selected subnet limits, completion, separated reflection, or host locatedness, the displayed route is the exact handoff surface. □

Theorem 3814.6 (LocatedUniformSpace real-completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *readback exported by a* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *packet is downstream of the displayed route through* U, D, A, W, T, F, N *and the structural rows* H, C, P, Q*. The packet can feed completion-facing chapters only by exposing the finite located uniform data; it does not assert that the surrounding uniform space is complete, separated, precompact, or represented by chosen Cauchy names.*

Proof. Read the completion-facing clause of the carrier. A real-facing consumer reaches rational witnesses only through N, and N is downstream of the displayed window and tolerance route.

The package row P records the relationship with $\mathsf{Real}^{\uparrow}$ and the uniform siblings, while Q names only this local packet. No row remains for a host completion theorem, a selected Cauchy representative, a quotient construction, or a global precompactness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformSpace}^{\uparrow}$ packet is generated from directed entourage data, located apartness, finite observation windows, dyadic tolerance rows, Cauchy filter and net handoff, regular rational readback, componentwise transport, displayed continuation, package provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$, obligation)

Scope closed: Definition 3814.1, Definition 3814.2, Definition 3814.3, Definition 3814.4, Definition 3814.5, and Definition 3814.6 bind the finite located directed-uniform certificate surface and its Cauchy filter/net obligation route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close completion, precompactness, quotient-filter equality, selected limits, separated reflection, or bridge checking for $\mathsf{LocatedUniformSpace}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, directed-entourage routing, locatedness exactness, Cauchy handoff, Real readback boundary, transport stability, replay stability, and ledger non-escape for BEDC.Derived.LocatedUniformSpa

Chapter 3815

A Concrete Naming Certificate for $\mathsf{ClosedIntervalBisection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedIntervalBisection}^{\uparrow}$ packet records constructive interval bisection over a located closed bounded interval by finite endpoint, midpoint, branch, and readback rows. It refines the closed bounded interval carrier of chapter 1591, the located bisection route of chapter 1711, the explicit interval split of chapter 1465, and the nested-route consumer of chapter 1713. Its completion-facing rows pass through chapter 126, chapter 301, and chapter 38. The packet supplies a closed-interval bisection surface for L10 Real and Completion arguments without importing open-cover compactness, real trichotomy, quotient real equality, selected infinite branches, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3815.1 (Closed interval bisection carrier)**.** *A* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, including lower and upper endpoint reads, endpoint order, rational containment, dyadic containment, finite windows, regular readback, and the terminal real-name seal of the source interval.*
2. *L is the located endpoint row. It records the endpoint-location evidence needed to decide which child interval is retained for the current finite request, without adding a trichotomy or host real comparison coordinate.*
3. *M is the midpoint row. It is computed from the displayed endpoint cells of I and is read before the branch row is inspected.*
4. *D is the dyadic branch row. It records the left or right retained child, the parent-to-child containment witness, and the dyadic diameter budget for the current bisection request.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the parent endpoints, midpoint read, retained child endpoints, and branch ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and dyadic branch data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, M, D, S, R have been displayed.*

8. *H records componentwise* hsame *transport for the closed interval, located endpoint, midpoint, dyadic branch, finite windows, regular rational readback, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a closed-interval bisection request through I, L, M, D, S, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, M, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, hidden endpoint choice, selected infinite branches, real trichotomy, quotient stream equality, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3815.2 (Closed interval bisection NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier $B = (I, L, M, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, closed-bounded interval exposure through I, located endpoint exposure through L, midpoint formation through M, dyadic branch and child-containment exposure through D, finite-window exposure through S, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 3815.1. The row I supplies the only closed bounded interval source, while L supplies the only located endpoint evidence available to the branch request.

The midpoint and branch obligations are consecutive. The row M is read from the displayed endpoint cells of I, and the dyadic branch row D records the retained child interval only after that midpoint row is visible.

The completion-facing obligations are downstream of the branch. The finite windows S and regular rational readback R consume the parent endpoints, midpoint, and retained child rows fixed by I, L, M, D, and the real seal E is reached only after that readback route is displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since none of the excluded host objects occurs as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3815.3 (Closed interval bisection branch nonescape)**.** *Every retained-branch read of an accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto L, M, D \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the retained child interval is read from displayed closed-interval endpoints, located endpoint evidence, the midpoint row, and the dyadic branch ledger; it is not supplied by a hidden branch selector, a real trichotomy oracle, open-cover compactness, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalBisectionUp.ClosedIntervalBisection branch nonescape route`

Proof. Start with the source row I. It fixes the closed bounded interval whose lower and upper endpoint data can be inspected by the branch request.

Next read L and M. The located endpoint row supplies only the displayed endpoint-location evidence for the current finite request, and the midpoint row supplies the child boundary computed from the same endpoint cells.

The dyadic branch row D is the only retained-child coordinate. It records the left or right child, the containment relation to the parent interval, and the dyadic diameter budget before any StreamName window is exposed.

Finally S, R, E, H, C, P, N read the retained child through finite windows, regular rational readback, real sealing, transport, replay, provenance, and local naming. No hidden selector, trichotomy oracle, compactness principle, quotient equality, choice schedule, or ambient completeness row is available in the carrier. □

Theorem 3815.4 (Closed interval bisection completion handoff)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, nested-interval, or compactness-facing consumer of an accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *packet must first read the closed source interval, located endpoint row, midpoint row, and dyadic branch row. The packet hands downstream consumers one displayed retained child route with transport, replay, provenance, and local naming; it does not hand them open-cover compactness, a completed branch, a selected real point, quotient equality of real names, or an ambient completeness theorem.*

Proof. Project the carrier $B = (I, L, M, D, S, R, E, H, C, P, N)$. The first four rows I, L, M, D are the complete interval-bisection route: source interval, located endpoint evidence, midpoint, and retained dyadic branch.

The readback rows are ordered after that route. A $\mathsf{StreamName}^{\uparrow}$ consumer reads only the finite windows in S, a $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads only the regular rational row R, and a $\mathsf{Real}^{\uparrow}$ consumer reaches the seal E only after the same branch data has been displayed.

Nested-interval and compactness-facing consumers receive the retained child through H, C, P, N. These structural rows preserve and replay the same packet; they do not introduce completed branches, selected points, quotient equality, open-cover compactness, or ambient completeness. □

Theorem 3815.5 (Closed interval bisection interval-modulus termination)**.** *An accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *carrier with a displayed unary history depth row n exports a finite dyadic interval window at that depth. Its endpoint route is inherited from the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row I, its branch row is inherited from the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *and* $\mathsf{IntervalBisection}^{\uparrow}$ *rows L, M, D, and its* $\mathsf{StreamName}^{\uparrow}$–$\mathsf{RegSeqRat}^{\uparrow}$–$\mathsf{Real}^{\uparrow}$ *seal handoff remains inside the displayed rows S, R, E, H, C, P, N. This is a finite modulus and termination row for the displayed depth, not open-cover compactness, trichotomy, a selected infinite branch, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed unary depth row. At depth zero, read the original closed bounded interval from I; its endpoint order, containment, finite window, regular readback, and real-name seal are exactly the source data already exposed by Definition 3815.1.

For the successor depth, apply the midpoint and retained-branch rows M, D. The located endpoint data in L chooses only the displayed finite child, while D transports endpoint order, containment, and the dyadic diameter budget from the parent to that child.

Componentwise hsame transport in H and displayed replay in C keep the induction inside the same carrier. The completion handoff of Definition 3815.4 then composes S, R, E after the current branch row, so the depth-n window terminates as a finite row rather than as a hidden infinite branch or completeness principle. □

Theorem 3815.6 (Closed interval bisection RegSeqRat route)**.** *Every accepted retained finite bisection window reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row only after reading the closed source interval, located*

endpoint evidence, midpoint row, dyadic retained-branch ledger, and $\mathsf{StreamName}^{\uparrow}$ *finite windows. For* $B = (I, L, M, D, S, R, E, H, C, P, N)$*, the regular rational readback factors through*

$$I, L, M, D, S \longmapsto R \longmapsto E, H, C, P, N.$$

It does not use open-cover compactness, a selected infinite branch, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3815.1. The row I supplies the closed source interval, L supplies the located endpoint evidence, M supplies the midpoint row, and D supplies the dyadic retained-branch ledger.

Apply Definition 3815.2 to identify S as the finite-window exposure and R as the regular rational readback row. The retained-branch route of Definition 3815.3 orders the source, located endpoint, midpoint, branch, stream, regular rational, and real rows in that sequence.

The depth window of Definition 3815.5 keeps this read finite at the displayed unary depth. Therefore a $\mathsf{RegSeqRat}^{\uparrow}$ consumer receives only the stated finite route.

No carrier coordinate names open-cover compactness, a selected infinite branch, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so none of those principles can mediate the readback. □

Falsifiable prediction: Every $\mathsf{ClosedIntervalBisection}^{\uparrow}$ consumer that omits the dyadic branch row loses the retained child interval before any RegSeqRat or Real seal can be read.

Independence witness: closedboundedinterval, located-interval-bisection, intervalbisection, located-real-nested-intervals, streamname, regseqrat, real: $\mathsf{ClosedIntervalBisection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a closed bounded interval source, located endpoint evidence, midpoint formation, a dyadic retained-branch ledger, finite windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalBisection}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, located endpoint evidence, a midpoint row, a dyadic retained-branch row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3815.1, Definition 3815.2, Definition 3815.3, and Definition 3815.4 seed the constructive closed-interval bisection route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, real trichotomy, quotient real equality, selected infinite branches, selected real points, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, closed-interval exposure, located endpoint exposure, midpoint formation, dyadic branch exactness, StreamName window exposure, RegSeqRat readback, and Real seal nonescape for BEDC.Derived.ClosedIntervalBisectionUp.

Chapter 3816

A Concrete Naming Certificate for $\mathsf{RealSpectrum}^{\uparrow}$

Following the classical Banach-algebra spectrum tradition of Gelfand theory, the $\mathsf{RealSpectrum}^{\uparrow}$ packet records the finite BEDC surface by which a real Banach-algebra element is tested against displayed real scalar windows, resolvent-boundary rows, and spectral-radius estimates without importing a host spectrum. It is a spectral neighbour of the L10 real and completion frontier: it reads the complete normed algebra surface of chapter 2189, the power-growth boundary of chapter 2352, the bounded-operator route of chapter 1644, the real seal of chapter 38, the regular-rational classifier of chapter 301, and the finite stream windows of chapter 126.

Definition 3816.1 (Real spectrum carrier)**.** *A* $\mathsf{RealSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, a, U, R, B, W, Q, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row carrying multiplication, norm, completion, product control, transport, replay, provenance, and local naming.*
2. *a is the carried algebra element whose real spectral behaviour is inspected.*
3. *U is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *or algebra-action row used when the element is read through a bounded operator surface.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal row. It admits only displayed real scalars, not a completed host scalar field.*
5. *B is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for real scalar windows. It records the regular rational classifier data used to compare a scalar probe with the element.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *spectral-window observation row. It names the finite list of scalar probes, operator-power requests, and resolvent-boundary tests that are actually displayed.*
7. *Q is the resolvent-boundary test row. For each displayed scalar window it records the local multiplication and inverse-check surface, together with a refusal row when the inverse evidence is not displayed.*
8. *E is the* $\mathsf{SpectralRadius}^{\uparrow}$ *estimate row. It reads only the power-window and norm-growth estimates supplied by the spectral-radius packet.*

9. *G is the real-spectrum readback row. It classifies a displayed scalar as admitted, refused, or boundary-facing only through R, B, W, Q, E.*

10. *H records componentwise* hsame *transport for the Banach-algebra source, carried element, bounded-operator row, real seal, regular-rational readback, stream windows, resolvent tests, spectral-radius estimates, readback, and structural rows.*

11. *C records displayed* Cont *replay of the route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SpectralRadius}^{\uparrow} \longmapsto \mathsf{RealSpectrum}^{\uparrow}.$$

12. *P is the* Pkg *provenance over* $\mathsf{RealSpectrum}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, a, U, R, B, W, Q, E, G, H, C, P, N$ and has no coordinate for a host spectrum, maximal ideals, Gelfand transform, spectral theorem, quotient completion, selected approximate eigenvectors, trichotomy over real scalars, or countable choice.

Lean4 ✓ `BEDC.Derived.RealSpectrumUp.RealSpectrumTasteGate single carrier alignment`

Theorem 3816.2 (Real spectrum NameCert obligations)**.** *For an accepted* $\mathsf{RealSpectrum}^{\uparrow}$ *carrier*

$$S = (A, a, U, R, B, W, Q, E, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Banach-algebra source, carried element, bounded-operator read, real seal, regular-rational readback, stream-window observations, resolvent-boundary tests, spectral-radius estimates, readback, transport, replay, provenance, and local naming row;*

2. *scalar-window compatibility, requiring every scalar probe in W to read its* $\mathsf{Real}^{\uparrow}$ *seal through R and its* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier through B before the probe can enter a resolvent test;*

3. *resolvent-boundary locality, requiring Q to display the multiplication and inverse-check rows used by each accepted scalar test and to record refusal when those rows are absent;*

4. *spectral-radius estimate discipline, requiring E to use only the power-window and norm-growth rows already certified by* $\mathsf{SpectralRadius}^{\uparrow}$*;*

5. *classifier transport, requiring H and C to preserve the scalar windows, resolvent tests, radius estimates, real seal, regular-rational readback, bounded-operator row, provenance, and local naming componentwise.*

No NameCert obligation is stored outside these rows. In particular, the packet exports no completed host spectrum, no maximal ideal space, no Gelfand-transform equality, no spectral theorem, no quotient completion, no selected approximate eigenvector, no trichotomy row, and no countable-choice schedule.

Proof. Carrier admission is projection from Definition 3816.1. The accepted packet exposes exactly the Banach-algebra source, carried element, bounded-operator read, real seal, regular-rational classifier, stream-window observations, resolvent tests, spectral-radius estimates, readback, transport, replay, provenance, and local naming rows.

Scalar-window compatibility uses R, B, W. Each scalar probe is first a displayed stream-window request, then a regular-rational readback, and only then a real-seal read. Thus a scalar cannot enter the spectrum packet as an ambient real number detached from the displayed L10 rows.

Resolvent locality is the row Q. It names the local multiplication and inverse-check evidence used by each displayed test, and it records refusal when the evidence is absent. The row therefore supplies a boundary test rather than a host inverse or a completed spectrum.

The radius estimate discipline is the row E. It imports only the finite power-window and norm-growth surface already exposed by $\mathsf{SpectralRadius}^{\uparrow}$, so no limit over all powers or selected approximate eigenvector is introduced.

Finally H, C, P, N transport, replay, source, and name only the listed coordinates. Since the excluded spectral and choice-theoretic objects have no coordinate in the carrier, they cannot appear as local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealSpectrumUp.RealSpectrumCarrier namecert obligations`

Closure status for $\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}$.

Constructive story: A $\mathsf{RealSpectrum}^{\uparrow}$ packet is generated from a BanachAlgebraUp source, a carried element, a BoundedLinearOperatorUp read, RealUp scalar seals, RegSeqRatUp readback rows, StreamNameUp spectral windows, resolvent-boundary tests, SpectralRadiusUp estimates, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3816.1 and Definition 3816.2 seed the finite real-spectrum route over BanachAlgebraUp, SpectralRadiusUp, BoundedLinearOperatorUp, RealUp, RegSeqRatUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealSpectrum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host spectrum, maximal ideal space, Gelfand representation, spectral theorem, quotient completion, selected approximate eigenvectors, real trichotomy, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, scalar-window compatibility, resolvent-boundary locality, spectral-radius estimate discipline, classifier transport, and consumer non-escape for BEDC.Derived.RealSpectrumUp.

Chapter 3817

A Concrete Naming Certificate for $\mathsf{BanachLattice}^{\uparrow}$

Following Kantorovich and Schaefer on Banach lattices, the $\mathsf{BanachLattice}^{\uparrow}$ packet records the finite BEDC naming surface for a vector lattice whose normed carrier is Cauchy-complete and whose positive cone remains visible to norm-facing consumers. It strengthens the ordered vector-lattice surface of chapter 3620 by adding the Banach-space completeness rows of chapter 90, while still reading the norm row of chapter 89, the join-meet lattice row of chapter 55, and the vector-space carrier of chapter 47. The packet supplies finite vector-lattice, norm, positive-cone, and Cauchy-completeness rows; it does not supply Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, or a host order-completion principle.

Definition 3817.1 (BanachLattice carrier)**.** *A* $\mathsf{BanachLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, A, N, P, M, H, C, S)$$

with the following displayed rows:

1. *R is the* $\mathsf{RieszSpace}^{\uparrow}$ *row, carrying the common vector endpoint, real scalar endpoint, lattice join and meet operations, positive and negative part ledgers, and vector-lattice classifier.*
2. *A is the* $\mathsf{Banach}^{\uparrow}$ *row over the same vector endpoint and normed metric surface, exposing the Cauchy-completeness witness through finite regular-Cauchy and limit-seal rows.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *compatibility row tying the norm used by A to the lattice decomposition and positive-cone rows visible in R.*
4. *P is the positive-cone monotonicity row. It records that positive part, negative part, cone membership, and order comparison are read before norm or completeness consumers inspect the packet.*
5. *M is the Cauchy-completeness ledger. It records the finite regular-Cauchy windows, modulus or metric-control rows, and terminal Banach seal used by downstream complete normed-lattice consumers.*
6. *H records componentwise* hsame *transport for the Riesz-space row, Banach row, norm compatibility row, positive-cone row, completeness ledger, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Banach-lattice request through R, A, N, P, M, H and back to the same finite ordered complete-normed surface.*

8. *S is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{BanachLattice}^\uparrow}$ *row over* $\mathsf{BanachLattice}^\uparrow$, $\mathsf{RieszSpace}^\uparrow$, $\mathsf{Banach}^\uparrow$, $\mathsf{Norm}^\uparrow$, $\mathsf{Lattice}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, BHist, hsame, Cont, *and* NameCert.

The classifier compares accepted packets only through these displayed rows. It has no coordinate for Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, pointwise function models, countable choice, or ambient host order equality.

Lean4 ✓ `BEDC.Derived.BanachLatticeUp.BanachLatticeTasteGate single carrier alignment`

Theorem 3817.2 (BanachLattice NameCert obligations)**.** *For every accepted* $\mathsf{BanachLattice}^\uparrow$ *carrier* $B = (R, A, N, P, M, H, C, S)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachLattice}^\uparrow}$ *surface consists exactly of carrier admission, vector-lattice row compatibility, norm compatibility, positive-cone exposure, Cauchy-completeness ledger exposure, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A downstream consumer may read the common vector endpoint, the lattice join and meet rows, the norm row, the positive-cone ledger, and the finite Banach completeness route, but it receives no Hahn-Banach functional, no Dedekind-complete lattice envelope, no selected supremum, and no quotient function-space representation.*

Proof. Carrier admission is projection from Definition 3817.1. The tuple displays exactly the Riesz-space row R, Banach row A, norm row N, positive-cone row P, completeness ledger M, and structural rows H, C, S.

The vector-lattice obligations are read through R. This fixes the vector endpoint, scalar endpoint, join and meet operations, order comparisons, and positive and negative part ledgers before any Banach-facing row is consumed.

The norm and completeness obligations are finite rows. The norm row N ties the Banach metric used by A to the lattice decomposition data in R, and the ledger M records the regular-Cauchy windows and terminal completeness seal exposed by the Banach row.

Finally H, C, S transport, replay, source, and name only those displayed coordinates. Since the carrier contains no coordinate for Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, or host order equality, none of those objects can be exported by the local certificate. □

Theorem 3817.3 (BanachLattice classifier stability)**.** *If two accepted* $\mathsf{BanachLattice}^\uparrow$ *packets are related by the classifier, then their Riesz-space rows, Banach completeness rows, norm compatibility rows, positive-cone rows, and Cauchy-completeness ledgers are related by the componentwise* hsame *transport recorded in H. Consequently every norm-facing, order-facing, or completeness-facing read factors through the displayed finite ledger and never through a host order equality, selected supremum, or hidden quotient carrier.*

Proof. Unfold the classifier of Definition 3817.1. Its comparison coordinates are exactly R, A, N, P, M, H, C, S.

The Riesz-space component transports the vector, lattice, and positive-part rows. The Banach component transports the normed metric and Cauchy-completeness rows over the same carried endpoint.

The norm compatibility and positive-cone rows then move along those transported coordinates. The completeness ledger M remains attached to the transported Banach row, so no accepted read can replace it by a selected supremum, quotient function-space equality, or host order construction. □

Theorem 3817.4 (BanachLattice Riesz-Banach handoff)**.** *Every accepted* $\mathsf{BanachLattice}^\uparrow$ *packet supplies a finite handoff*

$$\mathsf{RieszSpace}^\uparrow, \quad \mathsf{Banach}^\uparrow, \quad \mathsf{Norm}^\uparrow, \quad \mathsf{Lattice}^\uparrow, \quad \mathsf{VecSpace}^\uparrow$$

in which vector-lattice and positive-cone data are read before the Banach completeness ledger is consumed. The handoff exports the ordered complete normed carrier as NameCert data only through R, A, N, P, M, H, C, S.

Proof. Read the handoff from the carrier. The row R supplies the $\mathsf{RieszSpace}^{\uparrow}$ surface over the vector-space and lattice rows, while N repeats the norm compatibility needed by the Banach-facing route.

Next read A and M. These rows expose the Cauchy-complete normed carrier and the finite regular-Cauchy ledger without adding a supremum selector, Hahn-Banach functional, or quotient function model.

The positive-cone row P binds the order-facing part of the route to the same vector endpoint before H, C, S transport, replay, source, and name it. Thus the Riesz-space and Banach-space surfaces meet only through the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachLattice}^{\uparrow}$ packet is generated from a Riesz-space row, a Banach completeness row, norm compatibility data, positive-cone data, a Cauchy-completeness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3817.1, Definition 3817.2, Definition 3817.3, and Definition 3817.4 seed the finite ordered complete normed-lattice certificate over RieszSpaceUp, BanachUp, NormUp, LatticeUp, VecSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hahn-Banach separation, Dedekind completeness, selected suprema, quotient function spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vector-lattice compatibility, norm compatibility, positive-cone exposure, Cauchy-completeness ledger exposure, transport, replay, provenance, and local naming for BEDC.Derived.BanachLatticeUp.

The $\mathsf{HeronSquareRoot}^{\uparrow}$ hub routes Newton, dyadic bisection, reciprocal, and regular-Cauchy square-root rows through chapter 1756, chapter 291, and chapter 301 before its child files expose the Heron packet.

Chapter 3818

A Concrete Naming Certificate for $\mathsf{HeronSquareRoot}^{\uparrow}$

From Heron's classical square-root iteration and Bishop-style constructive convergence, the $\mathsf{HeronSquareRoot}^{\uparrow}$ packet records the finite dyadic evidence needed to run the Babylonian recurrence inside the L10 Real and Completion route. It refines the square-root service of chapter 1756 by exposing seed positivity, reciprocal accounting, contraction estimates, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal as named rows. The dependency route passes through chapter 1756, chapter 291, chapter 223, chapter 301, and chapter 38; it is not a host square-root function, quotient-real equality, a selected completed limit, or an ambient completeness theorem.

The carrier consumes $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows from chapter 291 before every reciprocal, halving, and squared-error comparison, and it consumes $\mathsf{RegSeqRat}^{\uparrow}$ readbacks from chapter 301 before exposing the terminal $\mathsf{Real}^{\uparrow}$ square-root seal. The Heron recurrence also consumes the guarded dyadic Newton square-root packet of chapter 7196 before exporting the RegSeqRat and Real seal rows. Thus the Heron route is a dependency-bearing finite lattice: dyadic tolerance rows govern the visible recurrence windows, regular rational readbacks certify the Cauchy handoff, and only then may a Real-facing consumer inspect the seal row.

Definition 3818.1 (Heron square-root carrier)**.** *A* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *A is the positive dyadic radicand row, including lower-bound and nonzero-denominator witnesses inside* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *S is the finite positive dyadic seed row from which the Heron recurrence starts.*
3. *R is the reciprocal and division ledger row, recording the displayed divisions A/x_i and every denominator witness used by the recurrence.*
4. *L is the finite Heron ledger whose rows have the form $x_{i+1} = (x_i + A/x_i)/2$ and whose arithmetic is read only through A, S, R and* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *W is the finite iteration-window row, exposing exactly the initial segment of L requested by a precision consumer.*
6. *D is the dyadic contraction-estimate row, recording the squared-error, halving, and tolerance comparisons that bound the visible window.*

7. *K is the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row showing that the displayed recurrence is the specialized Newton step for $x^2 - A$ without importing a host root.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from A, S, R, L, W, D, K.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular readback row Q has been displayed.*
10. *T records componentwise* hsame *transport for the radicand, seed, reciprocal ledger, Heron ledger, window, contraction estimate, Newton provenance, readback, Real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay of each precision request through $A, S, R, L, W, D, K, Q, E, T, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{HeronSquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host square-root operation, quotient-real equality, selected limits, arbitrary real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3818.2 (Heron square-root regular-Cauchy handoff)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite dyadic error ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal Real seal displayed in*

$$A, S \longmapsto R, L, W, D, K \longmapsto Q \longmapsto E.$$

Thus a Real or Completion consumer first sees the positive radicand and seed rows, then the reciprocal and division ledger, Heron recurrence window, dyadic contraction estimates, and $\mathsf{NewtonIteration}^{\uparrow}$ *provenance, and only then may read the regular rational name and* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports no host square-root function, quotient-real equality, selected completed limit, arbitrary real trichotomy, or ambient completeness object.*

Proof. Start with an accepted carrier $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$. The carrier inventory exposes the positive dyadic radicand A and seed S before any recurrence row can be inspected.

The numerical ledger is finite. The reciprocal row R stores the displayed divisions A/x_i, the Heron ledger L stores the visible recurrence rows, the window row W selects the requested finite segment, and the contraction row D stores the dyadic error comparisons for that segment.

The route then records provenance through K. This row identifies the same displayed recurrence as the Newton step for $x^2 - A$ through $\mathsf{NewtonIteration}^{\uparrow}$, but it contributes only finite ledger provenance and no external square-root value.

The consumer-facing read is downstream of those rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q is produced by replaying the finite dyadic ledger, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after Q has been displayed.

The remaining rows T, C, P, N transport, replay, record provenance, and name the same packet. Since the carrier contains no host square-root coordinate, quotient-real equality, selected limit, trichotomy classifier, or ambient completeness row, every accepted Real-facing read factors through the displayed route. □

Theorem 3818.3 (Heron square-root finite-window contraction induction)**.** *Let $H = (A, S, R, L, W, D, K, Q, E, T,$ be an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier. Every finite window W of the Heron ledger L preserves the dyadic contraction estimate recorded by D by induction on the displayed window length. In the successor step the proof reads the reciprocal row R, the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row K,*

and the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *are inspected. The induction exports no host square-root function, selected limit, trichotomy, quotient-real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The finite rows A, S, R, L, W, D, K are available before any regular rational readback or Real seal, and the handoff theorem places Q and E downstream of those rows.

Induct on the displayed length of the window row W. The empty window has no Heron step to check, so the contraction estimate is exactly the empty restriction of D.

For the successor window, read the previous visible window from L and use the induction hypothesis for its restriction of D. The new row of L is the displayed recurrence $x_{i+1} = (x_i + A/x_i)/2$; its division data is supplied by R, its recurrence provenance is supplied by K, and each tolerance comparison is read from the $\mathsf{DyadicRatCore}^{\uparrow}$ rows already recorded in D.

After the successor estimate is recorded, apply Definition 3818.2. The $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and $\mathsf{Real}^{\uparrow}$ seal E are therefore downstream of the finite induction and cannot change the displayed contraction window.

Rows T, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate contains a host square-root function, selected completed limit, trichotomy, quotient-real equality, or ambient completeness theorem, the finite-window induction remains an internal Heron-ledger statement. □

Theorem 3818.4 (Heron square-root regular-Cauchy RegSeqRat lattice)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing square-root consumer of an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

must first read the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *sibling route, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radicand and arithmetic rows, the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then the* $\mathsf{Real}^{\uparrow}$ *seal. In carrier notation, the lattice route projects* $A, S, R, L, W, D, K, Q, E, T, C, P, N$ *in that order and exposes no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the accepted carrier. The rows A and S are the positive dyadic radicand and seed rows, while R, L, W, D are the reciprocal ledger, Heron recurrence ledger, finite window, and dyadic contraction estimate.

Apply Definition 3818.2. It places every Real or Completion read downstream of the finite dyadic ledger and before the regular rational readback Q and terminal seal E.

Read the sibling lattice through provenance. The row K records the $\mathsf{NewtonIteration}^{\uparrow}$ specialization, P records the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ dependency route, and Q is the regular rational handoff generated from the displayed rows.

Finally transport and name the route through T, C, P, N. Since all consumer-facing data is already exhausted by $A, S, R, L, W, D, K, Q, E, T, C, P, N$, no Real-facing consumer can replace the lattice by a host square-root operation, quotient equality, selected limit, trichotomy, or ambient completeness. □

Theorem 3818.5 (Heron square-root NameCert obligations)**.** *For every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *surface consists exactly of the displayed carrier admission row, positive dyadic radicand and seed exactness, reciprocal and division ledger exactness, finite Heron-window stability, dyadic contraction accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The coordinates A and S give the positive dyadic radicand and seed rows, while R stores the reciprocal and division witnesses used by the recurrence.

The rows L, W, D, K form the finite numerical spine. The Heron ledger L records the displayed recurrence, W selects the finite window, D records the dyadic contraction estimates on that window, and K identifies the same row as the specialized $\mathsf{NewtonIteration}^{\uparrow}$ step for $x^2 - A$.

Apply Definition 3818.2 to the consumer-facing side. The regular readback Q is available only after the finite dyadic ledger has been replayed, and the $\mathsf{Real}^{\uparrow}$ seal E is read only after Q.

The rows T, C, P, N then transport, replay, source, and locally name the same packet. Since the carrier has no row for a host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 3818.6 (Heron square-root contraction-window stability)**.** *Let* $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N$ *be an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier. Any componentwise* hsame *transport recorded by* T *preserves the finite Heron window* W *and the same dyadic contraction estimate recorded by* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *is inspected.*

Proof. Read the source window first. The window row W is a finite initial segment of the displayed Heron ledger L, and the contraction row D records the dyadic comparisons for exactly that segment.

Transport acts componentwise through T. Hence a transported carrier must carry corresponding A, S, R, L, W, D, K rows before any consumer can inspect Q or E.

The handoff theorem separates the finite dyadic ledger from the regular-Cauchy readback. Therefore the contraction estimate seen on the transported window is the same estimate certified by D, and no later $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row can alter the finite Heron-window bound. □

Theorem 3818.7 (Heron square-root regular-Cauchy sibling route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read factors through the sibling chain from chapter 1756 to chapter 291, chapter 223, chapter 301, and chapter 38 before any* $\mathsf{Real}^{\uparrow}$*-facing square-root consumer receives the result. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the route first projects the positive dyadic radicand and seed rows A, S*, then the reciprocal ledger, Heron recurrence, window, contraction estimate, and Newton provenance rows* R, L, W, D, K*, then the regular readback row* Q*, and finally the Real seal* E *together with transport, replay, provenance, and local naming through* T, C, P, N*.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The first seven rows are exactly the finite dyadic and Newton-style data named by the sibling chapters: radicand and seed in $\mathsf{DyadicRatCore}^{\uparrow}$, recurrence provenance in $\mathsf{NewtonIteration}^{\uparrow}$, and a visible Heron window with contraction accounting.

Apply Definition 3818.2. It forces every Real or Completion consumer to pass from A, S through R, L, W, D, K before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and the terminal $\mathsf{Real}^{\uparrow}$ seal E are visible.

Use Definition 3818.6. The finite window and contraction estimate survive componentwise transport, so the sibling route cannot be replaced by a host square-root operation, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.

Thus the accepted read is a regular-Cauchy square-root route: it moves through the displayed dyadic, Newton, regular rational, and Real siblings before any Real-facing consumer can use the square-root result. □

Theorem 3818.8 (Heron square-root dyadic-Real forward link)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer whose output is read as a* $\mathsf{Real}^{\uparrow}$ *result must first consume the dyadic radicand and contraction rows from chapter 291, the Newton provenance row from chapter 223, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal from chapter 38. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the forward dependency order is A, S, R, L, W, D *before* K*,* K *before* Q*, and* Q *before* E*; the transport, replay, provenance, and local naming rows* T, C, P, N *preserve that order for every accepted Real-facing read.*

Proof. Project the carrier. The dyadic radicand and seed rows A, S, reciprocal ledger R, Heron ledger L, finite window W, and contraction row D are visible before any row with a Real-facing type is inspected.

Use the sibling route of Definition 3818.7. It places the Newton provenance row K downstream of the dyadic Heron rows, then places $\mathsf{RegSeqRat}^{\uparrow}$ readback Q downstream of K, and finally exposes the $\mathsf{Real}^{\uparrow}$ seal E only after Q.

The structural rows T, C, P, N cannot permute this order: they transport componentwise, replay the displayed route, record provenance, and name the same carrier. Hence a Real-facing consumer must follow the DyadicRatCore, NewtonIteration, RegSeqRat, and Real sibling route rather than reading a host square-root function, quotient-real equality, selected limit, or ambient completeness witness. □

Theorem 3818.9 (Heron square-root dyadic refinement route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$$

first refines the positive dyadic radicand and visible recurrence window through the $\mathsf{DyadicRatCore}^{\uparrow}$ *and interval-refinement tolerance rows before Newton provenance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are visible. The route reads the dyadic interval-refinement surface of chapter 1831 as part of the* A, S, R, L, W, D *window discipline, then exposes* K, Q, E, T, C, P, N *only as downstream provenance, readback, seal, transport, replay, provenance, and local naming rows.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The positive radicand row A, seed row S, reciprocal ledger R, Heron ledger L, finite window W, and contraction estimate D are the complete dyadic surface visible before Newton provenance or Real-facing readback.

Apply the dyadic-Real forward link of Definition 3818.8. It orders the DyadicRatCore radicand and contraction rows before K, places K before the regular rational readback Q, and places Q before the terminal Real seal E.

Use finite-window contraction induction from Definition 3818.3. Each successor window reads the reciprocal row R, the displayed recurrence row L, and the dyadic tolerance comparisons in D before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ row is inspected.

The interval-refinement dependency is therefore internal to the dyadic window. The refinement surface of chapter 1831 is consumed while A, S, R, L, W, D are being read, and the remaining rows K, Q, E, T, C, P, N only preserve, replay, source, name, or seal that already-refined route. □

Theorem 3818.10 (Heron square-root Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read from an accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet first exposes the positive dyadic radicand, seed, reciprocal ledger, Heron ledger, finite window, contraction estimate, Newton provenance, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows*

$$A, S, R, L, W, D, K, Q$$

and only then exposes the $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows*

$$E, T, C, P, N.$$

In particular, the read cannot bypass $\mathsf{RegSeqRat}^{\uparrow}$ *with a host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness.*

Proof. Project the accepted carrier $H = (A, S, R, L, W, D, K, Q, E, T, C, P, N)$. The first eight rows are the dyadic Heron and regular-rational route, while E, T, C, P, N are the terminal seal and structural rows.

Invoke the regular-Cauchy handoff of Definition 3818.2. It places Q downstream of A, S, R, L, W, D, K and places E downstream of Q.

Use contraction-window stability from Definition 3818.6. Transport preserves the finite window W and contraction estimate D before any regular readback or Real seal can be inspected.

Apply the sibling route of Definition 3818.7. The terminal E row is therefore downstream of $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, so no host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness coordinate can replace the displayed L10 route. □

Falsifiable prediction: Every accepted HeronSquareRootUp Real-facing read factors through positive DyadicRatCoreUp radicand and seed rows, a reciprocal and division ledger, a finite Heron window, dyadic contraction estimates, NewtonIterationUp provenance, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, newtoniteration, regseqrat, real, regularcauchysquare-root: $\mathsf{HeronSquareRoot}^{\uparrow}$ is not bijective to any listed sibling because it binds positive dyadic seed data, reciprocal/division rows, Heron recurrence windows, contraction estimates, Newton provenance, regular readback, Real seal handoff, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{HeronSquareRoot}^{\uparrow}$ packet is generated from positive DyadicRatCore radicand and seed rows, reciprocal and division ledger rows, finite Heron recurrence windows, dyadic contraction estimates, NewtonIteration provenance, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3818.1 and Definition 3818.2 seed the finite Heron square-root carrier and its regular-Cauchy handoff route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host square-root function, quotient-real equality, selected completed limits, arbitrary real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic seed exactness, reciprocal and division ledger exactness, Heron-window stability, dyadic contraction accounting, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HeronSquareRootUp.

The $\mathsf{HeronSquareRoot}^{\uparrow}$ iteration route consumes $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{Real}^{\uparrow}$ sealing through the same finite tail-filter discipline used by chapter 1605 and chapter 38. Its positive radicand,

reciprocal ledger, Heron window, contraction estimate, and Newton provenance are therefore visible before the regular rational readback and Real seal can be used, and no host square-root operation, quotient real equality, selected limit, countable choice, or ambient completeness row replaces that tail-filter handoff.

Theorem 3818.11 (Heron square-root DyadicRatCore anchor)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet that exposes a dyadic square-root iteration must read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, denominator, reciprocal, halving, and squared-error rows of chapter 291 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 is consumed. In carrier notation the Heron recurrence route remains*

$$A, S, R, L, W, D, K, Q, E, T, C, P, N,$$

where A, S, R, L, W, D *are the dyadic radicand, seed, reciprocal, recurrence, finite-window, and contraction rows;* Q *is the regular rational readback; and* E *is the Real seal. The route exports no host square-root function, quotient real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the Heron carrier from the chapter spine. The radicand and seed rows are dyadic rows, and the reciprocal, halving, and squared-error comparisons in the visible recurrence are also read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next use the regular-Cauchy handoff and finite-window contraction induction already present in this chapter. They place the $\mathsf{RegSeqRat}^{\uparrow}$ readback after the dyadic iteration window, so a Real-facing consumer cannot reach the seal before the dyadic tolerance ledger is displayed.

Finally read the terminal $\mathsf{Real}^{\uparrow}$ route. Since the seal is downstream of the dyadic and regular-rational rows, the Heron packet contributes a dependency edge through chapter 291 and chapter 38, not a host square-root operator or an ambient completeness principle. □

Every accepted $\mathsf{HeronSquareRoot}^{\uparrow}$ consumer that reads a square-root endpoint through a bisection or interval-shrink route first consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, the dyadic interval cells of chapter 306, and the located bisection or nested-interval rows of chapter 1711 and chapter 1713. Only after those interval rows have been displayed may the route pass through the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38; no host square-root endpoint, selected limit, quotient-real equality, trichotomy, or ambient completeness row is exported.

Theorem 3818.12 (Heron square-root Newton dependency route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that seals a square-root approximation consumes the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row of chapter 223, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic rows before the square-root read is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route projects A, S, R, L, W, D *as the finite dyadic Heron ledger, reads* K *as the specialized Newton step, then exposes* Q *and* E *before transport, replay, provenance, and local naming through* T, C, P, N*. It exports no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D contain the positive radicand, seed, reciprocal ledger, recurrence window, and dyadic contraction data needed before any square-root approximation can be consumed.

Apply Definition 3818.2. It orders every Real-facing read through the finite dyadic ledger and the $\mathsf{NewtonIteration}^{\uparrow}$ provenance row before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal seal E are visible.

Use the sibling route in Definition 3818.7. It identifies the same route through chapter 223, chapter 301, and the Real-facing endpoint without adding a host square-root value.

Finally read the regular-Cauchy and Real dependency children already present in this chapter. They keep Q and E attached to the displayed Heron ledger, so the square-root read is sealed only after the Newton, regular rational, Real, and dyadic rows have all been displayed. □

Theorem 3818.13 (Heron square-root Newton interval dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that exports a* $\mathsf{Real}^{\uparrow}$*-facing square-root approximation consumes either the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row or the nested-interval shrink route before the regular-rational and Real rows are public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the route first exposes the positive dyadic radicand and seed rows A, S, the reciprocal ledger R, the Heron recurrence window L, W, and the contraction estimate D. It then reads the Newton provenance row K through chapter 223, or the interval-shrink evidence through the nested route of chapter 1713, before reaching the $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D are the finite radicand, seed, reciprocal, recurrence-window, and contraction rows; none of them is a completed square-root value.

The iteration witness is consumed next. If the packet follows the Newton route, the row K identifies the Heron recurrence as the displayed $\mathsf{NewtonIteration}^{\uparrow}$ step for $x^2 - A$. If it follows the interval route, the same contraction ledger is read as retained nested interval shrink data through the located nested-interval chapter.

Only after one of those finite iteration routes is visible may the Real-facing rows be inspected. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q records the regular rational readback of the displayed approximation window, and the $\mathsf{Real}^{\uparrow}$ row E seals the terminal read. Transport, replay, provenance, and local naming through T, C, P, N preserve that dependency order and add no host square-root operation, selected limit, quotient-real equality, trichotomy, or ambient completeness theorem. □

Theorem 3818.14 (Heron square-root Newton regular-Cauchy bridge)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read factors its Newton update route through chapter 223 and its convergence and seal route through chapter 1756 before reaching the Heron square-root* $\mathsf{NameCert}$ *surface. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the rows A, S, R, L, W, D, K expose the displayed Heron recurrence as a finite $\mathsf{NewtonIteration}^{\uparrow}$ *update, while Q, E, T, C, P, N expose the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming. No accepted consumer may read the Heron square-root packet from the Newton row alone or from the regular-Cauchy seal alone.*

Proof. Project the accepted carrier. The rows A, S, R, L, W, D give the positive radicand, seed, reciprocal ledger, recurrence ledger, visible window, and contraction estimate, while K records that this finite recurrence is the displayed $\mathsf{NewtonIteration}^{\uparrow}$ route.

Apply Definition 3818.12. It places every square-root-facing read downstream of the finite Newton update ledger and before the regular rational readback and terminal seal are exposed.

Apply Definition 3818.18. It routes the same packet through the $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ convergence service, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the $\mathsf{Real}^{\uparrow}$ seal row E before export.

Finally use the NameCert obligations of Definition 3818.5. The rows T, C, P, N preserve transport, replay, provenance, and local naming for the joint route, so neither the Newton update nor the regular-Cauchy seal can be consumed as an isolated square-root source. □

Theorem 3818.15 (Heron square-root Newton sibling lattice)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that exposes a square-root-facing value first consumes the* $\mathsf{NewtonIteration}^{\uparrow}$ *route of chapter 223, the Newton interval route already attached to the Heron window, the regular Cauchy dependency through chapter 301, and the reciprocal Real route of chapter 6251. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the read projects the positive radicand and seed rows, the reciprocal ledger, the finite Heron window, the contraction estimate, the Newton provenance row K, the regular-rational readback Q, and the terminal $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any square-root read is public.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The finite Heron ledger is visible before any consumer-facing read, so the Newton provenance row K is read while the positive radicand, reciprocal ledger, window, and contraction estimate are still part of the displayed packet.

Apply Definition 3818.12 and Definition 3818.13. These sibling routes attach the Newton iteration surface and the interval-shrink evidence to the same Heron window rather than to a separate square-root object.

Next apply Definition 3818.17 and Definition 3818.19. The regular Cauchy route supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, while the reciprocal route forces the reciprocal ledger through the existing Real reciprocal packet before Q and E are public.

Finally apply Definition 3818.10. The terminal $\mathsf{Real}^{\uparrow}$ row E is available only after the Newton, regular Cauchy, and reciprocal rows have all been displayed, so no host square-root function, quotient-real equality, selected limit, trichotomy, or ambient completeness coordinate can bypass the sibling lattice. □

Theorem 3818.16 (Heron square-root nonnegative regular-Cauchy reciprocal-window route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed by either chapter 6251 or chapter 1756 must display the same nonnegative* $\mathsf{Real}^{\uparrow}$ *radicand row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and reciprocal-window budget before the square-root-facing seal is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route first exposes the positive radicand and seed rows A, S, then the reciprocal ledger R through the reciprocal-window rows of chapter 703, then the finite Heron window L, W, D, K, and only then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and terminal* $\mathsf{Real}^{\uparrow}$ *seal E used by the regular-Cauchy square-root route.*

Proof. Project the Heron carrier. The rows A, S display the positive radicand and seed, while R is the reciprocal ledger consumed by the Heron recurrence.

Read the reciprocal-window dependency through Definition 3818.19. The Real reciprocal chapter can consume the Heron packet only after the reciprocal ledger has passed through the finite inverse-budget and guarded-division rows named there.

Next read the regular-Cauchy square-root dependency through Definition 3818.18. The same nonnegative radicand route must expose L, W, D, K before the regular-rational readback Q and terminal Real seal E are visible.

The structural rows T, C, P, N preserve the shared route. Thus the reciprocal and regular-Cauchy square-root consumers meet at the displayed nonnegative Real row, RegSeqRat readback, and reciprocal-window rows, with no host square-root function, quotient real equality, selected completed limit, trichotomy, or ambient completeness coordinate. □

Theorem 3818.17 (Heron square-root regular-Cauchy dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read first consumes the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *route of chapter 1756. The visible dependency then proceeds through the positive radicand and reciprocal rows of* $\mathsf{DyadicRatCore}^{\uparrow}$*, the recurrence provenance row of* $\mathsf{NewtonIteration}^{\uparrow}$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any square-root-facing consumer may read the packet. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the dependency route is exactly

$$\mathsf{RegularCauchySquareRoot}^{\uparrow} \longmapsto A, S, R, L, W, D, K \longmapsto Q \longmapsto E \longmapsto T, C, P, N.$$

It has no coordinate for a host square-root function, quotient-real equality, selected completed limit, arbitrary real trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$. The positive radicand, seed, reciprocal, division, Heron-window, and contraction rows are the finite data consumed by the regular-Cauchy square-root route.

Apply Definition 3818.2. It orders every Real-facing read through A, S, R, L, W, D, K before the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and the $\mathsf{Real}^{\uparrow}$ seal E are visible.

Apply Definition 3818.7. This identifies the same route with the sibling chain through chapter 1756, chapter 291, chapter 223, chapter 301, and chapter 38.

The structural rows T, C, P, N then transport, replay, source, and name the dependency without adding any excluded host square-root, quotient equality, selected limit, trichotomy, or completeness row. □

Theorem 3818.18 (Heron square-root regular-Cauchy square-root forward route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed as a* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *route must display the finite path*

$$\mathsf{NewtonIteration}^{\uparrow}, \mathsf{NestedInterval}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

before the square-root packet is exported. For a carrier

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the consumer first reads the positive radicand and seed rows A, S*, the reciprocal ledger* R*, the Heron recurrence and window rows* L, W*, and the contraction estimate* D*; then it reads the* $\mathsf{NewtonIteration}^{\uparrow}$ *provenance row* K*, the nested-interval shrink route attached to the same window data, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and finally the structural rows* T, C, P, N*. The route exports no host square-root function, quotient-real equality, selected completed limit, arbitrary trichotomy, or ambient completeness theorem.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D are the finite numerical ledger for the radicand, seed, reciprocal accounting, recurrence window, and dyadic contraction estimate.

Read the iteration and interval support next. The row K records the Heron recurrence as the displayed $\mathsf{NewtonIteration}^{\uparrow}$ step, while the nested-interval dependency route of Definition 3818.13 attaches the same contraction ledger to a finite interval-shrink path.

Apply Definition 3818.17. It forces every $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ consumer through the sibling route before the square-root-facing read is public.

Apply Definition 3818.4. The regular rational readback row Q and terminal $\mathsf{Real}^{\uparrow}$ seal E are available only after the displayed Heron ledger and Newton route have been opened.

Finally transport, replay, source, and name the packet through T, C, P, N. These are exactly the hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows needed before export, and they add no host square-root operation, quotient equality, selected limit, trichotomy, or ambient completeness coordinate. □

Theorem 3818.19 (Heron square-root real reciprocal forward link)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet reads its reciprocal row through chapter 703, chapter 1943, and chapter 6251 before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal are public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the reciprocal ledger R is the displayed nonzero-apart route feeding the Heron recurrence, and the visible consumer path is

$$R \longmapsto \mathsf{CauchyInverseBudget}^{\uparrow} \longmapsto \mathsf{RegularCauchyDivision}^{\uparrow} \longmapsto \mathsf{RealReciprocal}^{\uparrow} \longmapsto Q \longmapsto E.$$

No Heron read exports a host reciprocal function, total division, quotient real equality, arbitrary trichotomy, selected completed limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted Heron carrier. The reciprocal ledger row R is already a coordinate of the finite packet, and it appears before the Heron recurrence window, contraction estimate, regular readback, and Real seal.

Apply Definition 3818.2. It forces every Real-facing Heron read to pass through the reciprocal and division ledger before Q and E can be inspected.

Use the sibling route from Definition 3818.7. The row R is therefore not a local host inverse; it is a displayed reciprocal ledger over the dyadic and regular-Cauchy route.

Finally read the reciprocal lattice through the named chapters. chapter 703 supplies the finite inverse budget, chapter 1943 supplies the guarded division handoff, and chapter 6251 supplies the nonzero-apart Real reciprocal packet. Only after those rows are visible may the Heron readback Q and terminal seal E be public. □

Theorem 3818.20 (HeronSquareRoot real and RegSeqRat forward lattice)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *square-root approximation read consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint first passes through the finite lattice*

$$\mathsf{NewtonIteration}^{\uparrow},\ \mathsf{RegularCauchySquareRoot}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

already displayed by the sibling routes of chapter 223, chapter 1756, chapter 301, chapter 291, and chapter 38. For an accepted carrier

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the read exposes the Newton provenance row K, the regular-Cauchy square-root route over A, S, R, L, W, D, the regular-rational readback Q, the dyadic arithmetic rows A, S, R, D, and the terminal Real seal E before T, C, P, N may transport, replay, source, or name the result. It exports no host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows A, S, R, L, W, D display the positive dyadic radicand, seed, reciprocal ledger, Heron recurrence, finite window, and contraction estimate, while K records the same finite recurrence as a $\mathsf{NewtonIteration}^{\uparrow}$ route.

Apply Definition 3818.18. The square-root-facing consumer must read the regular-Cauchy route before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ endpoint is public.

Next apply Definition 3818.4 and Definition 3818.10. These place Q before E and keep both rows downstream of the finite dyadic and Newton lattice.

Finally attach T, C, P, N. Transport, replay, provenance, and local naming preserve the displayed route and add no coordinate for a host square-root function, quotient-real equality, selected completed limit, trichotomy, or ambient Real completeness. □

Theorem 3818.21 (Heron square-root Real route)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer that reaches a* $\mathsf{Real}^{\uparrow}$ *seal must first display the Newton–Heron recurrence route, the dyadic bisection or Newton-interval dependency route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the reciprocal and nonnegative regular-Cauchy rows, componentwise transport, replay, provenance, and local naming. The route exports no host square-root function, selected limit, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the routed siblings for the carrier. The Newton and Heron recurrence rows expose the iterative rational window, while the dyadic bisection or Newton-interval rows identify the finite interval dependency used by that window.

The completion-facing read then passes through the $\mathsf{RegSeqRat}^{\uparrow}$ row and the reciprocal and nonnegative regular-Cauchy rows before the $\mathsf{Real}^{\uparrow}$ seal is available. Componentwise transport, displayed replay, provenance, and the local $\mathsf{NameCert}$ row preserve the same route without changing its endpoints.

Since none of these routed rows contains a host square-root function, a selected limit, quotient real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$, every Real-facing Heron square-root consumer remains inside the displayed finite dependency path. □

Every $\mathsf{HeronSquareRoot}^{\uparrow}$ read consumes the dyadic-bisection dependency rows of chapter 361 and the Newton interval rows of Definition 3818.13 before the regular Cauchy square-root row is handed to chapter 301 and the terminal seal is read through chapter 38. This route keeps square-root dependency data inside the displayed L10 lattice and does not export selected limits or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 3818.22 (Heron square-root RegSeqRat Real dependency route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *iteration budget consumed by the reciprocal forward link first reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular windows and the* $\mathsf{Real}^{\uparrow}$ *seal rows already displayed by the Heron carrier. For*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the reciprocal route of Definition 3818.19 exposes the square-root approximation only after the regular readback Q and terminal seal E are attached to the same positive radicand, seed, reciprocal ledger, Heron recurrence window, dyadic contraction estimate, and Newton provenance rows. Thus the visible dependency passes through chapter 6251, chapter 301, and chapter 38 without adding a host square-root function, quotient real equality, selected completed limit, arbitrary trichotomy, or ambient completeness coordinate.

Proof. Project the Heron carrier. The rows A, S, R, L, W, D, K are the finite radicand, seed, reciprocal, recurrence, window, contraction, and Newton-provenance rows consumed before any square-root-facing read is public.

Apply Definition 3818.19. Its reciprocal forward path reads the existing reciprocal chapter through the same row R and then returns to the Heron readback boundary rather than fabricating a separate inverse or square-root object.

Now read the regular-rational and real rows. The row Q is the $\mathsf{RegSeqRat}^{\uparrow}$ regular-window readback produced from the displayed Heron iteration budget, and E is the terminal $\mathsf{Real}^{\uparrow}$ seal visible only after that readback.

Finally transport and replay through T, C, P, N. These rows keep the reciprocal, RegSeqRat, and Real dependencies on the same finite Heron route, so the square-root approximation is exposed as a displayed NameCert dependency rather than a host operation, quotient equality, selected limit, trichotomy, or completeness principle. □

Theorem 3818.23 (Heron square-root RegSeqRat forward dependency)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *endpoint must first pass through chapter 1756. For*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the square-root-facing dependency displays the regular-Cauchy source route before exposing the Heron rows $A, S, R, L, W, D, K, Q, E, T, C, P, N$*. Thus the RegSeqRat or Real consumer reads the positive radicand, seed, reciprocal ledger, recurrence window, contraction estimate, Newton provenance, regular readback, Real seal, transport, replay, provenance, and local naming only after the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *handoff is visible.*

Proof. Project the Heron carrier. The rows A, S, R, L, W, D, K display the radicand, seed, reciprocal ledger, recurrence window, contraction estimate, and Newton provenance before the regular readback row Q or terminal Real seal row E can be consumed.

Invoke the regular-Cauchy handoff through Definition 3818.18 and Definition 3818.17. These routes force the square-root-facing read through chapter 1756 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ endpoint receives the Heron packet.

Now use the RegSeqRat Real dependency route of Definition 3818.22. It keeps Q and E downstream of the same A, S, R, L, W, D, K rows and therefore prevents a detached square-root approximation from entering the regular-rational or Real boundary.

Finally close through T, C, P, N. Transport, replay, provenance, and local naming preserve the same dependency order, so the read exports a finite Heron route rather than a host square-root function, quotient Real equality, selected completed limit, arbitrary trichotomy, or ambient completeness coordinate. □

Theorem 3818.24 (Heron square-root sibling consumer lattice)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer route used by chapter 6251, chapter 6793, and chapter 7196 first reads the Heron square-root packet's dyadic approximation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* $\mathsf{NameCert}_{\mathsf{HeronSquareRoot}^{\uparrow}}$ *rows.*

Proof. Project the accepted Heron carrier from the chapter spine. The rows A, S, R, L, W, D expose the dyadic radicand, seed, reciprocal ledger, Heron recurrence window, and contraction estimate before any downstream consumer can inspect the regular-rational readback Q or terminal $\mathsf{Real}^{\uparrow}$ seal E.

Apply the Heron reciprocal, continuity, and Newton sibling routes already recorded in this chapter. The real reciprocal forward link routes the reciprocal row through chapter 6251; the nonnegative regular-Cauchy reciprocal-window route keeps the same Heron window visible for square-root continuity consumers; and the Newton sibling lattice keeps the dyadic Newton route attached to the displayed approximation rows.

Now read the downstream chapters. chapter 6251, chapter 6793, and chapter 7196 therefore receive the Heron packet only after Q, E, T, C, P, N have preserved readback, Real sealing, transport, replay, provenance, and local naming. No consumer route obtains a host square-root value, a hidden reciprocal, quotient-real equality, selected completion, or ambient completeness row. □

Theorem 3818.25 (Heron square-root sibling dependency lattice)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *read that is exposed to a Real or Completion consumer must pass through the displayed Newton, regular-Cauchy, RegSeqRat, and Real siblings before the terminal square-root value is public. For a carrier*

$$H = (A, S, R, L, W, D, K, Q, E, T, C, P, N),$$

the lattice order first reads the positive radicand and seed rows, then the Heron recurrence and Newton provenance rows of chapter 223, then the regular-Cauchy square-root route of chapter 1756, then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, and finally the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route exports no host square-root operation, quotient Real equality, selected completed limit, arbitrary trichotomy, countable choice, or ambient completeness theorem.*

Proof. Project the accepted Heron carrier. The rows A, S, R, L, W, D, K expose the positive source data, reciprocal ledger, visible Heron window, dyadic contraction estimate, and Newton provenance before any Real-facing row is available.

Apply the existing Heron handoff and sibling-route theorems. They place the $\mathsf{NewtonIteration}^{\uparrow}$ row on the same finite recurrence window, route the convergence read through $\mathsf{RegularCauchySquareRoot}^{\uparrow}$, and force the $\mathsf{RegSeqRat}^{\uparrow}$ row Q to be inspected before the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The square-root boundary is therefore a consumer of the same displayed seal, not an independent oracle. Transport, replay, provenance, and local naming through T, C, P, N preserve the order of the sibling rows, so the public Heron read remains inside the horizontal dependency lattice and cannot export any excluded host or quotient object. □

Theorem 3818.26 (Heron square-root sibling dependency route)**.** *Every* $\mathsf{HeronSquareRoot}^{\uparrow}$ *consumer that exports a square-root approximation must first read the initial positive* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *rows, the* $\mathsf{NewtonIteration}^{\uparrow}$ *update route, the* $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ *convergence readback, and the* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *interval discipline. These sibling rows are a dependency route for the Heron packet, not a new square-root existence theorem, not a host square-root function, and not a selected completed limit.*

Proof. Project the Heron carrier through the already displayed radicand, seed, reciprocal ledger, recurrence window, dyadic contraction, readback, terminal seal, transport, replay, provenance, and local naming rows.

Read the initial numerical rows first. The positive $\mathsf{Real}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ data supply the radicand and rational seed interface before any iterative square-root approximation can be consumed.

Next follow the sibling routes. The $\mathsf{NewtonIteration}^{\uparrow}$ row records the Heron update, $\mathsf{RegularCauchySquareRoot}^{\uparrow}$ records the constructive convergence readback, and $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ records the interval discipline that keeps the approximation inside finite dyadic windows.

The recurrence also reads the reciprocal-accounting route before the regular readback and Real seal become public. Its reciprocal ledger passes through chapter 703, chapter 1943, and chapter 6251, matching the forward link of Definition 3818.19.

Thus the public Heron read is a horizontal dependency route through the existing Newton, regular-Cauchy, Real, rational, and interval siblings. It exports only the displayed dependency order and adds no square-root theorem, selected limit, quotient-real equality, trichotomy, or ambient completeness coordinate. □

Theorem 3818.27 (Heron square-root sibling route)**.** *Every accepted* $\mathsf{HeronSquareRoot}^{\uparrow}$ *packet must read* $\mathsf{DyadicRatCore}^{\uparrow}$ *radicand and tolerance data from chapter 291,* $\mathsf{NewtonIteration}^{\uparrow}$ *recurrence rows from chapter 223, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 together with the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows used by chapter 1756 before it exports the square-root route. The Heron carrier therefore sits on the displayed sibling lattice rather than on a host square-root function, selected completed limit, quotient-real equality, trichotomy row, or ambient completeness coordinate.*

Chapter 3819

A Concrete Naming Certificate for $\mathsf{ContractiveMap}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{ContractiveMap}^{\uparrow}$ packet records a displayed contraction between metric carriers as finite NameCert data. It sits over the source and target metric rows of chapter 57, the complete-metric consumer boundary of chapter 128, the tolerance export surface of chapter 369, the endpoint discipline of chapter 396, and the finite orbit route of chapter 2359. The packet names the map-level shrink surface that Banach and Picard consumers can inspect before any fixed point, completed limit, quotient equality, or host function extensionality is available.

Definition 3819.1 (Contractive map carrier)**.** *A* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, f, k, D, G, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row and Y is the target* $\mathsf{Met}^{\uparrow}$ *row.*
2. *f is the finite map-code row from carried source histories to carried target histories.*
3. *k is the contraction-constant row, with displayed nonnegativity and strict unit-bound evidence.*
4. *D is the metric-distance witness row recording source-distance and target-distance requests for the accepted map reads.*
5. *G is the graph row witnessing that every accepted application of f is a finite BEDC graph read.*
6. *S is the shrink ledger recording each displayed comparison*

   $$d_Y(f(x), f(y)) \leq k\, d_X(x, y)$$

   for source histories carried by X and graph reads carried by G.
7. *H records componentwise* hsame *transport for the source and target metric rows, map-code row, contraction constant, distance witnesses, graph row, shrink ledger, continuation rows, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from source metric requests through f, k, D, G, S to target metric responses, uniform-modulus handoff, Picard iteration reads, and fixed-point consumers.*

9. *P is the* Pkg *provenance over* ContractiveMap$^\uparrow$, Met$^\uparrow$, CompleteMetric$^\uparrow$, UniformModulus$^\uparrow$, BanachFixedPoint$^\uparrow$, PicardIteration$^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{ContractiveMap}^\uparrow}$ *row.*

The carrier has no coordinate for a selected fixed point, a completed Picard limit, quotient equality of maps, host function extensionality, ambient completeness, or bridge checking.

Lean4 ✓ `BEDC.Derived.ContractiveMapUp.ContractiveMapTasteGate single carrier alignment`

Theorem 3819.2 (Contractive map NameCert obligations)**.** *For every accepted* ContractiveMap$^\uparrow$ *carrier $K = (X, Y, f, k, D, G, S, H, C, P, N)$, the local* NameCert$_{\text{ContractiveMap}^\uparrow}$ *surface consists exactly of source and target metric exposure, map-code admission, contraction-constant exactness, distance-witness exposure, graph replay, shrink-ledger replay, componentwise transport, displayed continuation, package provenance, and local naming through the rows $X, Y, f, k, D, G, S, H, C, P, N$.*

Proof. Project the carrier. The analytic rows are precisely the source metric X, target metric Y, map code f, contraction constant k, distance witnesses D, graph row G, and shrink ledger S.

The contraction obligation is replayed from the same rows. A target-distance read must pass through G, use the distance witnesses in D, and cite the comparison stored in S against the same constant row k.

The structural rows preserve that packet and add no analytic coordinate. Transport through H, replay through C, provenance through P, and local naming through N keep the displayed route intact, so the listed rows exhaust the NameCert surface. □

Theorem 3819.3 (Contractive map Picard consumer handoff)**.** *Every* PicardIteration$^\uparrow$ *or* BanachFixedPoint$^\uparrow$ *consumer of an accepted* ContractiveMap$^\uparrow$ *carrier receives contraction evidence only through*

$$X, Y, f, k, D, G, S, H, C, P, N.$$

The handoff exports metric-distance shrink replay and the strict unit-bound row for k; it does not export fixed-point existence, complete-metric convergence, quotient equality of maps, a selected endpoint, or host function extensionality.

Proof. Start with the NameCert obligations of Definition 3819.2. They make the shrink ledger S the only row that can justify a Picard or Banach contraction read.

A Picard consumer therefore reads the graph row G, the distance witnesses D, and the contraction constant k before it can replay a one-step or finite-orbit shrink comparison. The continuation row C transports this order to uniform-modulus, iteration, and fixed-point consumers without adding a limit point.

Completeness and endpoint selection remain outside the packet. A Banach consumer must attach its own complete-metric and endpoint rows, while the ContractiveMap$^\uparrow$ handoff supplies only the finite map-level shrink data. □

Theorem 3819.4 (Contractive map constant-row exactness)**.** *For every accepted* ContractiveMap$^\uparrow$ *carrier $K = (X, Y, f, k, D, G, S, H, C, P, N)$, every contraction read exposed by* NameCert$_{\text{ContractiveMap}^\uparrow}$ *uses the displayed constant row k together with its nonnegativity and strict unit-bound evidence. No Picard, Banach, or uniform-modulus consumer may replace k by an unstored scalar or by a scalar recovered from ambient host data.*

Proof. Project the carrier and apply the NameCert obligations of Definition 3819.2. The obligation list has a single contraction-constant coordinate, namely the row k, and it records the strict unit-bound evidence as part of that row.

The Picard handoff of Definition 3819.3 exports the same row to downstream consumers. Since the handoff excludes host function extensionality, quotient equality of maps, completed limits, and selected endpoints, there is no second route from which a different contraction scalar can be read. □

Theorem 3819.5 (Contractive map distance-witness stability)**.** *For every accepted* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier* $K = (X, Y, f, k, D, G, S, H, C, P, N)$*, each source-distance or target-distance request used in a contraction comparison factors through the distance-witness row* D*. If the request involves a displayed application of* f*, the same read is tied to the graph row* G *before it can be used by the shrink ledger* S*.*

Proof. Project the carrier. The obligations in Definition 3819.2 list distance-witness exposure and graph replay as separate rows, and the proof of that theorem identifies D as the row carrying source-distance and target-distance requests.

A contraction comparison in S is therefore not a free metric assertion. It first reads the relevant source and target distance witnesses through D, and when the comparison evaluates f it reads the corresponding finite graph entry through G. The continuation row C can replay this order, but it cannot bypass either displayed row. □

Theorem 3819.6 (Contractive map shrink-ledger replay)**.** *For every accepted* $\mathsf{ContractiveMap}^{\uparrow}$ *carrier* $K = (X, Y, f, k, D, G, S, H, C, P, N)$*, every shrink comparison available to* $\mathsf{PicardIteration}^{\uparrow}$*,* $\mathsf{BanachFixedPoint}^{\uparrow}$*, or* $\mathsf{UniformModulus}^{\uparrow}$ *consumers is one of the comparisons stored in* S *and transported, replayed, provenanced, and locally named through* H, C, P, N*. The carrier exposes no additional contraction inequality outside those rows.*

Proof. Project the carrier and read the shrink-ledger clause of Definition 3819.2. That clause makes S the row recording the displayed comparisons

$$d_Y(f(x), f(y)) \leq k\, d_X(x, y)$$

for carried source histories and graph reads.

The consumer handoff in Definition 3819.3 sends the same finite shrink data to Picard and Banach consumers, and the carrier definition routes uniform-modulus handoff through the same continuation surface. Componentwise transport through H, displayed replay through C, provenance through P, and local naming through N preserve the stored ledger entries rather than generating fresh analytic inequalities. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3819.1, Definition 3819.2, Definition 3819.3, Definition 3819.4, Definition 3819.5, and Definition 3819.6 seed the finite contraction-map surface for metric, graph, constant, shrink-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not assert fixed-point existence, complete-metric convergence, quotient equality of maps, selected endpoints, Lean verification, axiom-clean audit, or bridge checking for $\mathsf{ContractiveMap}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, contraction-constant exactness, graph replay, distance-witness exposure, shrink-ledger replay, Picard handoff, and field faithfulness for $X, Y, f, k, D, G, S, H, C, P, N$.

Constructive story: A ContractiveMap$^{\uparrow}$ packet is generated from source and target metric rows, a finite map-code row, a contraction constant below one, metric-distance witnesses, a graph row, a shrink ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3820

A Concrete Naming Certificate for $\mathsf{ClosedBoundedSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet records a bounded real sequence as finite BEDC source rows rather than as a completed sequence space or a compactness theorem. It is the bounded-sequence neighbour of the closed interval carrier in chapter 1591 and the Bolzano–Weierstrass route of chapter 1254. The carrier exposes only the finite bound interval row, the sequence windows read through chapter 126, the regular rational readbacks of chapter 301, the dyadic bound ledger of chapter 291, the terminal real seal of chapter 38, transport, replay, provenance, and local naming.

Definition 3820.1 (Closed bounded sequence carrier)**.** *A* $\mathsf{ClosedBoundedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the bound interval row, read through the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *certificate of chapter 1591;*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows inspecting the requested sequence coordinates;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same coordinate windows;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger recording dyadic lower and upper bound comparisons against the interval row I;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for each inspected coordinate, reached only after I, W, R, D have been displayed;*
6. *H records componentwise* hsame *transport of the interval, window, readback, dyadic, and seal rows;*
7. *C records displayed* Cont *replay from interval bounds through finite sequence windows and coordinate seals;*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedSequence}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$ *row.*

The carrier exports no general sequential compactness principle, no countable choice selector, no selected convergent subsequence, and no ambient space of all bounded sequences. Its public object is the finite bounded-coordinate packet B.

Lean4 ✓ `BEDC.Derived.ClosedBoundedSequenceUp.TasteGate.ClosedBoundedSequenceTasteGate single carrier alignment`

Theorem 3820.2 (Closed bounded sequence NameCert obligations)**.** *For the carrier of Definition 3820.1, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for* $B = (I, W, R, D, E, H, C, P, N)$*; bound-interval admission through* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*; coordinate-window admission through* $\mathsf{StreamName}^{\uparrow}$*; regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*; dyadic containment of every inspected coordinate through* $\mathsf{DyadicRatCore}^{\uparrow}$*; terminal coordinate sealing through* $\mathsf{Real}^{\uparrow}$*; classifier preservation under* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier. Its interval row, coordinate-window rows, readback rows, dyadic bound ledger, Real seals, transport, replay, provenance, and local name are visible coordinates of one finite packet.

Read the bound before the coordinate seals. The $\mathsf{ClosedBoundedInterval}^{\uparrow}$ row supplies the lower endpoint, upper endpoint, endpoint order, and containment ledgers, while W, R, D inspect only finitely many sequence coordinates against that same interval.

The structural rows close the naming surface. Componentwise hsame transport and displayed Cont replay preserve exactly the listed rows, and the provenance and local naming rows source that finite route without adding sequential compactness, a choice-selected subsequence, or an ambient sequence-space object. □

Theorem 3820.3 (Closed bounded sequence Cauchy-subsequence window)**.** *Every* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *or Cauchy-subsequence consumer of an accepted* $\mathsf{ClosedBoundedSequence}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows $I, W, R, D, E, H, C, P, N$*. The route supplies bounded coordinate windows and dyadic containment readbacks, but it does not supply general sequential compactness, countable choice, a selected convergent subsequence, or an ambient completeness theorem.*

Proof. Start from the carrier inventory. The only sequence-facing rows are the finite windows W, regular rational readbacks R, dyadic bound ledger D, and coordinate seals E, all attached to the bound interval row I.

Next route the downstream consumer through the same rows. A $\mathsf{BolzanoWeierstrass}^{\uparrow}$ or Cauchy-subsequence consumer can request a bounded-coordinate window, but the request is replayed by C only after I, W, R, D, E have been transported by H and sourced by P, N.

The excluded principles are not carrier coordinates. Since the packet contains no selector for an infinite subsequence, no global compactness row, no countable choice row, and no ambient complete sequence space, the consumer receives only the displayed finite bounded-window route. □

Falsifiable prediction: Every $\mathsf{BolzanoWeierstrass}^{\uparrow}$ or Cauchy-subsequence read of a $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet factors through a bound interval row, finite StreamName coordinate windows, RegSeqRat readbacks, a DyadicRatCore bound ledger, RealUp coordinate seals, transport, replay, provenance, and local naming.

Independence witness: closedboundedinterval, bolzanoweierstrass, streamname, regseqrat, dyadicratcore, real: $\mathsf{ClosedBoundedSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly

carries a bounded sequence source, finite coordinate windows, dyadic bound readbacks, coordinate seals, and a consumer-facing bounded-window route.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedSequence}^{\uparrow}$ packet is generated from a closed bounded interval row, finite sequence windows, regular rational readbacks, dyadic bound comparisons, RealUp coordinate seals, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3820.1, Definition 3820.2, and Definition 3820.3 bind the finite bounded-coordinate carrier before Bolzano–Weierstrass and Cauchy-subsequence consumers add their own extraction rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, general sequential compactness, countable choice, selected convergent subsequences, or ambient sequence-space completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, bound-readback, and Cauchy-subsequence handoff for BEDC.Derived.ClosedBoundedSequenceUp.

Chapter 3821

A Concrete Naming Certificate for $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet records the finite radius-choice ledger used in the Heine–Cantor route. It reads compact finite-net data from chapter 129, pointwise continuity rows from chapter 130, finite lower-bound folds from chapter 619, dyadic tolerance bookkeeping from chapter 291, and the uniform-modulus consumer surface of chapter 369. The chapter is a finite NameCert carrier for the radius handoff; it is not an open-cover compactness proof, a semantic continuity theorem, or a choice function over all points.

Definition 3821.1 (Compact continuity radius-choice carrier)**.** *A* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, M, Q, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric finite-net row, read through* $\mathsf{CompactMetric}^{\uparrow}$*;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row at the displayed finite centers;*
3. *M is the finite center-to-radius matching ledger;*
4. *Q is the positive rational radius ledger attached to the selected center rows;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger;*
6. *L is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound fold over the displayed radius rows;*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row reached from the finite lower-bound ledger;*
8. *H records componentwise* hsame *transport for compact-net, continuous-map, radius, dyadic, fold, and handoff rows;*
9. *C records displayed* Cont *replay from finite-net centers and pointwise radii to the uniform-modulus handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{HeineCantor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$ *row.*

The carrier exposes the finite radius ledger and its consumer handoff. It has no coordinate for an external radius-selection function, open-cover compactness, semantic function equality, or a point chosen outside the finite compact-net bundle.

Lean4 (definition) `BEDC.Derived.CompactContinuityRadiusChoiceUp.CompactContinuityRadiusChoiceUp`

Theorem 3821.2 (Compact continuity radius-choice NameCert obligations)**.** *For the carrier of Definition 3821.1, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (K, F, M, Q, D, L, U, H, C, P, N)$*; compact finite-net admission through* $\mathsf{CompactMetric}^{\uparrow}$*; pointwise-radius admission through* $\mathsf{ContinuousMap}^{\uparrow}$*; radius matching through* M, Q, D*; finite lower-bound folding through* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$*; the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through* U*; classifier preservation under* H*; displayed replay through* C*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier. The compact finite-net row, continuous-map pointwise rows, radius ledgers, dyadic comparisons, lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local name are all visible coordinates.

The radius choice is finite. Each radius row is attached to a displayed compact-net center and a displayed pointwise-modulus row; the lower-bound fold consumes only those finitely many positive rational and dyadic comparisons.

The consumer handoff is therefore confined to the displayed route. Componentwise transport and replay preserve the same rows, while provenance and local naming record the route without adding a global selector, open-cover compactness object, semantic continuity theorem, or point outside the finite bundle. □

Theorem 3821.3 (Compact continuity radius-choice uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{FiniteNetMinimumFold}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

Equivalently, the handoff receives the compact finite-net centers, pointwise radius rows, dyadic tolerance comparisons, the finite lower-bound ledger, and the displayed transport, replay, provenance, and local naming rows. It receives no open-cover compactness object, no external radius-selection function, no semantic continuity theorem, and no point selected outside the finite net.

Proof. Start with the compact-net and pointwise-modulus rows. The carrier exposes K and F before any radius row can be folded, so every radius is tied to a displayed finite center and a displayed continuous-map modulus row.

Next read the finite lower-bound ledger. The rows M, Q, D, L package the radius matching, positive rational radius data, dyadic comparisons, and finite minimum fold. This is the only source of a tolerance-only handoff to $\mathsf{UniformModulus}^{\uparrow}$.

Finally apply the structural rows. The route H, C, P, N transports, replays, sources, and names exactly the finite radius-choice ledger, so a uniform-modulus consumer cannot add open-cover compactness, a global radius-selection function, semantic continuity, or an off-net selected point. □

Falsifiable prediction: Every $\mathsf{UniformModulus}^{\uparrow}$ read of a $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet factors through CompactMetric finite-net centers, ContinuousMap pointwise radii, DyadicRatCore tolerance comparisons, a FiniteNetMinimumFold lower-bound ledger, transport, replay, provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, finitenetminimumfold, dyadicratcore, heinecantor: $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite compact-net radius ledger and the UniformModulus-facing lower-bound handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactContinuityRadiusChoice}^{\uparrow}$ packet is generated from compact finite-net centers, ContinuousMap pointwise radius rows, radius matching ledgers, dyadic tolerance comparisons, a finite lower-bound fold, a UniformModulus handoff row, hsame transports, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3821.1, Definition 3821.2, and Definition 3821.3 bind the finite radius-choice carrier feeding the Heine–Cantor uniform-modulus route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContinuityRadiusChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, semantic continuity, global radius selection, or off-net point choice.

Upgrade path: Obligation closure requires checked carrier admission, classifier preservation, finite radius fold, and UniformModulus handoff for BEDC.Derived.CompactContinuityRadiusChoiceUp.

Chapter 3822

A Concrete Naming Certificate for $\mathsf{DuhamelPrinciple}^{\uparrow}$

Following Pazy's semigroup treatment of evolution equations and Evans' variation-of-constants discussion for forced linear problems, the $\mathsf{DuhamelPrinciple}^{\uparrow}$ packet records the finite BEDC route by which a homogeneous evolution row, a forcing row, and an integral convolution row are read as one solution handoff. It sits after the real-time surface of chapter 38, the integration interface of chapter 123, the ODE local-flow source of chapter 193, the dynamical-system flow packet of chapter 195, and the semigroup associativity surface of chapter 79. The carrier names the forcing route; it does not assert a host solution operator, classical existence theorem, quotient trajectory equality, or choice-selected mild solution.

Definition 3822.1 (DuhamelPrinciple carrier)**.** *A* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, E, F, G, I, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Real}^{\uparrow}$ *time-window row, including the initial time, endpoint time, and inspected finite subintervals.*
2. *E is the homogeneous evolution row, read as an* $\mathsf{Ode}^{\uparrow}$ *local-flow or* $\mathsf{DynSystem}^{\uparrow}$ *flow packet and constrained by the* $\mathsf{Semigroup}^{\uparrow}$ *composition ledger.*
3. *F is the forcing-data row, recording the displayed inhomogeneous source samples and their* $\mathsf{Real}^{\uparrow}$ *readbacks.*
4. *G is the propagator action row, pairing each displayed forcing sample with the elapsed-time evolution row that carries it toward the endpoint.*
5. *I is the* $\mathsf{Integral}^{\uparrow}$ *convolution row over the finite time window, with its partition, approximation, and endpoint ledgers visible as integral data.*
6. *S is the solution readback row, exposing the homogeneous contribution and the integral forcing contribution as the endpoint history inspected by downstream consumers.*
7. *H records componentwise* hsame *transport for time windows, evolution rows, forcing rows, propagator rows, integral rows, solution readbacks, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Duhamel request through $T, E, F, G, I, S, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{DuhamelPrinciple}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Semigroup}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, F, G, I, S, H, C, P, N$. It has no coordinate for a host semigroup on a completed function space, an unobserved Bochner integral, a selected solution operator, classical choice, quotient trajectory equality, or global well-posedness.

Lean4 ✓ `BEDC.Derived.DuhamelPrincipleUp.DuhamelPrincipleTasteGate single carrier alignment`

Theorem 3822.2 (DuhamelPrinciple NameCert obligations)**.** *For every accepted* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *carrier $D = (T, E, F, G, I, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}^{\uparrow}}$ *obligation surface consists exactly of real-time window admission, homogeneous evolution readback, forcing-data admission, propagator action exposure,* $\mathsf{Integral}^{\uparrow}$ *convolution readback, endpoint solution readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier. The time row T fixes the inspected interval before the evolution, forcing, or convolution rows are read. The homogeneous row E is already a named ODE or dynamical-system flow route, and its semigroup composition behavior is visible only through the displayed $\mathsf{Semigroup}^{\uparrow}$ ledger.

Read the forcing and propagator rows next. The row F supplies the inhomogeneous samples and real readbacks, while G records how the displayed evolution row carries each sample toward the endpoint. No host trajectory space is introduced at this stage.

The integral row I is then the only source of convolution data. Its partition, approximation, and endpoint ledgers are inherited from $\mathsf{Integral}^{\uparrow}$, and the solution row S may inspect only the homogeneous contribution together with that displayed convolution endpoint.

Finally use H and C to transport and replay the same finite rows, and use P and N for provenance and local naming. Since all excluded analytic objects lack carrier coordinates, they cannot enter the local NameCert obligation surface. □

Theorem 3822.3 (DuhamelPrinciple forcing route)**.** *Every downstream* $\mathsf{Ode}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, *stability, or PDE consumer of an accepted* $\mathsf{DuhamelPrinciple}^{\uparrow}$ *packet reads the inhomogeneous contribution through the finite route*

$$T, E, F, G \longmapsto I \longmapsto S \longmapsto H, C, P, N.$$

Thus the forced endpoint is available only as homogeneous evolution plus the displayed $\mathsf{Integral}^{\uparrow}$ *convolution of forcing data against the propagator row. The route exports no host solution operator, unbounded semigroup theorem, chosen mild solution, quotient trajectory equality, or ambient completeness principle.*

Proof. Begin with a forcing-facing consumer. The real-time row T and the homogeneous evolution row E determine the displayed elapsed-time cells and the semigroup-style composition ledger available to the packet.

Read F and G after that boundary is fixed. The forcing samples are paired with visible propagator action rows, so the consumer sees only finitely displayed source observations and their transported endpoint contributions.

The convolution is then read from I. This invokes the $\mathsf{Integral}^{\uparrow}$ carrier and ledger discipline over the same time window, so partitions and approximations remain integral rows rather than hidden choices.

The endpoint row S repacks the homogeneous part and the forcing convolution as the public readback. Transport, replay, provenance, and naming are supplied by H, C, P, N, which preserve the same route and leave every absent analytic object outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}\uparrow}$.

Constructive story: A $\mathsf{DuhamelPrinciple}^{\uparrow}$ packet is generated from a RealUp time window, homogeneous ODE or DynSystem evolution row, Semigroup composition ledger, forcing-data row, propagator action row, IntegralUp convolution row, solution readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3822.1, Definition 3822.2, and Definition 3822.3 seed the finite variation-of-constants carrier, obligation surface, and forcing route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DuhamelPrinciple}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host solution operators, unbounded semigroup theorems, chosen mild solutions, quotient trajectory equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, forcing-route exactness, IntegralUp convolution readback, evolution transport stability, solution nonescape, and TasteGate field faithfulness for BEDC.Derived.DuhamelPrincipleUp.

Chapter 3823

A Concrete Naming Certificate for $\mathsf{LocatedSequenceLimit}^{\uparrow}$

Following Bishop and Bridges on located real limits of Cauchy sequences, the $\mathsf{LocatedSequenceLimit}^{\uparrow}$ packet records the finite evidence that a displayed stream name, regular rational sequence, and real-name seal describe the same located limit route. It sits between the stream observation surface of chapter 126, the regular rational sequence surface of chapter 301, the terminal real-name seal of chapter 38, the Cauchy criterion route of chapter 1509, and the located Cauchy surface of chapter 1595. The packet is finite L10 NameCert data for a located sequence limit; it is not a selected quotient representative, an ambient completeness theorem, a trichotomy decision, or a countable-choice schedule.

Definition 3823.1 (Located sequence limit carrier)**.** *A* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing only finite windows of the displayed sequence name.*
2. *W is the located-window ledger. For each displayed precision request it records the finite inspected window, the located side of the comparison, and the tolerance budget used by the read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the inspected windows as regular rational sequence data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the stream window and regular rational readback agree on the same located limit route.*
5. *H records componentwise* hsame *transport for the stream source, located window, regular readback, real seal, replay, provenance, and local name.*
6. *C records displayed* Cont *replay from a precision request through*

$$S, W \longmapsto R \longmapsto E.$$

7. *P is the* Pkg *provenance over* $\mathsf{LocatedSequenceLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCrite}$ $\mathsf{LocatedCauchySequence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the finite stream windows, located comparison ledger, regular rational readback, and real-name seal describe one displayed limit route. No carrier coordinate stores a quotient sequence, selected representative of the limit, host equality on reals, trichotomy oracle, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedSequenceLimitUp.TasteGate.LocatedSequenceLimitTasteGate single carrier alignment`

Theorem 3823.2 (Located sequence limit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier* $L = (S, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}^{\uparrow}}$ *surface consists exactly of stream-window admission, located-window tolerance accounting, regular rational readback, real-name seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows* S, W, R, E, H, C, P, N*. The* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *and* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *siblings may consume the packet only through those displayed rows.*

Proof. Project the carrier of Definition 3823.1. The sequence is visible only through the finite stream row S, and the located comparison evidence is visible only through the window ledger W.

The readback row R repacks the same finite observations as $\mathsf{RegSeqRat}^{\uparrow}$ data. The real-name row E is downstream of S, W, R, so a real-facing consumer reaches the limit seal only after replaying the displayed located route.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier has no coordinate for quotient sequences, selected representatives, host equality, trichotomy, or ambient completeness, the listed NameCert rows are exhaustive. □

Theorem 3823.3 (Located sequence limit Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, *or* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *carrier must read*

$$S, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the terminal real-name seal cannot be replaced by host real equality, a quotient completion object, a selected subsequence, trichotomy on signs, or an ambient completeness proof.

Proof. Begin with a consumer request for real-limit or Cauchy-facing data. The packet exposes sequence observations only through S, and every located precision request is accounted for by W.

The regular rational readback row R is the only bridge from the inspected windows to a rational sequence surface. Thus the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ seal E only after the finite route has been replayed.

The rows H, C, P, N transport, replay, source, and name the same route. No excluded analytic object is present in the tuple, so no downstream consumer can extract one from the packet. □

Theorem 3823.4 (Located sequence limit window uniqueness)**.** *Let*

$$L = (S, W, R, E, H, C, P, N) \quad \textit{and} \quad L' = (S', W', R', E', H', C', P', N')$$

be accepted $\mathsf{LocatedSequenceLimit}^{\uparrow}$ *packets. If they expose the same source* $\mathsf{StreamName}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, located-window ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row, then they determine the same finite located-limit route. In particular, there are not two different accepted finite L10 windows for the same located request.*

Proof. Project both packets through the carrier definition Definition 3823.1. The hypotheses identify the only visible source row S, the only located-window ledger W, and the only regular readback row R available to either packet.

Apply the NameCert obligation theorem Definition 3823.2. It states that all accepted located-limit data are exhausted by the displayed rows S, W, R, E, H, C, P, N, so the two packets have no additional coordinate on which their finite routes could diverge.

Finally use the Real seal nonescape theorem Definition 3823.3. The terminal seal is reached only through the same stream, located-window, and readback route, and the carrier contains no selected quotient representative, ambient completeness proof, trichotomy oracle, or countable-choice schedule. Thus the accepted packets determine one finite located-limit route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}$.

Constructive story: A $\mathsf{LocatedSequenceLimit}^{\uparrow}$ packet is generated from StreamName observations, located finite-window tolerance data, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3823.1, Definition 3823.2, Definition 3823.3, and Definition 3823.4 seed the finite located sequence-limit surface for the L10 StreamName-to-RegSeqRat-to-Real route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSequenceLimit}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-window accounting, RegSeqRat readback, Real seal routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSequenceLimitUp.

Chapter 3824

A Concrete Naming Certificate for $\mathsf{DedekindGapLocatedCut}^{\uparrow}$

Following Bishop and Bridges constructive located cuts, the $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ packet records the finite lower-upper separation data that a completion or real-comparison consumer may inspect before any construction of $\mathsf{Real}^{\uparrow}$ is claimed. It is a located-cut boundary over the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational sequence route of chapter 301, the terminal real-name seal of chapter 38, and the located-supremum consumer surface of chapter 1427. The packet supplies a gap-located comparison boundary consumed by $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readers; it is not a Dedekind construction of $\mathsf{Real}^{\uparrow}$, not quotient equality of cuts, and not an ambient completeness principle.

Definition 3824.1 (Dedekind gap located cut carrier)**.** *A* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, D, W, R, S, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-cut ledger: finite rational cells certified to lie below the boundary.*
2. *U is the upper-cut ledger: finite rational cells certified to lie above the boundary.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap row. For every displayed tolerance request it names a dyadic bracket or a separating dyadic cell between the currently exposed lower and upper ledgers.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the finite lower, upper, and gap requests actually read by a consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the windowed lower-upper data as regular rational observations.*
6. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *boundary handoff row available to consumers that need an upper-bound or lower-approximation surface.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is exposed only after L, U, D, W, R, S have been displayed.*
8. *H records componentwise* hsame *transport for lower cells, upper cells, dyadic gaps, stream windows, regular readback, located-supremum handoff, real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay through the ordered route $L, U, D, W, R, S, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, W, R, S, E, H, C, P, N$. It has no coordinate for quotient equality of cuts, a selected real representative, countable choice, trichotomy, a host order-completeness theorem, or a Dedekind construction of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DedekindGapLocatedCutUp.DedekindGapLocatedCutTasteGate single carrier alignment`

Theorem 3824.2 (Dedekind gap located cut carrier boundary)**.** *Every accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier is read through the displayed route*

$$L, U, D, W, R, S, E, H, C, P, N.$$

Thus a consumer receives lower-cut cells, upper-cut cells, dyadic gap witnesses, finite stream windows, regular rational readback, a located-supremum boundary handoff, a terminal Real seal, transport, replay, provenance, and local naming. No accepted read exposes quotient equality of lower-upper cuts, a selected real representative, trichotomy, countable choice, ambient completeness, or a construction of $\mathsf{Real}^{\uparrow}$ *from Dedekind cuts.*

Proof. Project the carrier of Definition 3824.1. The rows L and U are the only lower and upper cut data, while D supplies the finite dyadic gap evidence inspected by the request.

The rows W and R place those gap observations inside a finite observation schedule and regular rational readback. The located-supremum row S is downstream of the same windowed data, and the real seal E is reached only after that route is displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not fields of the carrier, no consumer can recover them from an accepted $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ read. □

Theorem 3824.3 (Dedekind gap located cut obligations)**.** *For an accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *carrier $G = (L, U, D, W, R, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper ledgers, dyadic gap exposure, StreamName window exposure, RegSeqRat readback, LocatedSupremum boundary handoff, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming. The obligations do not include a quotient of cuts, a host real equality test, selected supremum data outside S, or any completeness assertion for* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission lists the rows $L, U, D, W, R, S, E, H, C, P, N$, so the local NameCert surface begins with a finite inventory rather than an external real construction.

The computational obligations are read in order. Dyadic gap exposure comes from D, window discipline from W, regular rational readback from R, and the upper-bound or lower-approximation consumer handoff from S before the terminal seal E can be consumed.

Transport and replay do not enlarge the inventory. The rows H and C preserve the same displayed data, while P and N record provenance and local naming, so the listed obligations exhaust the accepted packet. □

Theorem 3824.4 (Dedekind gap RegSeqRat boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ *packet factors through the finite route*

$$L, U \longmapsto D \longmapsto W \longmapsto R \longmapsto S \longmapsto E.$$

The route exposes a located lower-upper gap boundary that $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers may use for comparison work. It does not identify cuts by quotient equality, select a representing real, assert a Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, or import ambient order completeness.*

Proof. Start with the lower and upper ledgers L, U. A consumer sees only the displayed finite cells, so the request is a boundary read rather than an equality of cuts.

The dyadic row D supplies the gap witness requested at the displayed precision, and W records the finite observation window in which that witness is inspected. The readback row R then repacks the same window as regular rational sequence data.

The located-supremum handoff S and terminal seal E remain downstream of those rows. Consequently every RegSeqRat or Real-facing read is a routed boundary read, with no coordinate for the excluded quotient, selection, construction, or completeness claims. □

Falsifiable prediction: An accepted $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ read is refuted if a RegSeqRat or Real-facing consumer can remove the lower ledger, upper ledger, dyadic gap row, StreamName window, RegSeqRat readback, LocatedSupremum handoff, Real seal, transport, replay, provenance, or local naming while preserving the same packet claim.

Independence witness: dyadicratcore, streamname, regseqrat, real, locatedsupremum: $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower and upper cut ledgers, dyadic gap witnesses, finite observation windows, regular rational readback, located-supremum handoff, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindGapLocatedCut}^{\uparrow}$ packet is generated from lower-cut ledgers, upper-cut ledgers, dyadic gap witnesses, StreamName windows, RegSeqRat readback, a LocatedSupremum boundary handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3824.1, Definition 3824.2, Definition 3824.3, and Definition 3824.4 seed the L10 located-cut boundary consumed by RegSeqRat and Real-facing comparison packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindGapLocatedCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Dedekind construction of RealUp, quotient equality of cuts, selected real representatives, trichotomy, countable choice, ambient order completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper gap exposure, dyadic tolerance transport, StreamName window discipline, RegSeqRat readback, LocatedSupremum handoff, and Real seal nonescape for BEDC.Derived.DedekindGapLocatedCutUp.

Chapter 3825

A Concrete Naming Certificate for $\mathsf{FastCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FastCauchyCriterion}^{\uparrow}$ packet records the finite criterion data by which displayed fast Cauchy windows, monotone fast-modulus control, regular rational readback, and the terminal real-name seal may be inspected together before a Real or Completion consumer reads the result. It sits between the fast sequence surface of chapter 2185, the fast modulus discipline of chapter 1781, the regular rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is criterion data for a displayed regular-Cauchy route; it is not quotient real equality, a selected representative, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3825.1 (FastCauchyCriterion carrier)**.** *A* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row recording the inspected sequence values used by the criterion.*
2. *D is the dyadic tolerance scale row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, against which the fast-window comparisons are measured.*
3. *M is the monotone* $\mathsf{FastCauchyModulus}^{\uparrow}$ *row, displaying the threshold discipline that sends a requested tolerance to a later inspected window and preserves monotonicity of the request ledger.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility row, recording that the displayed windows and dyadic bounds form the regular rational readback consumed by Real and Completion routes.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, available only after S, D, M, R have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerance scale, monotone modulus, regular rational readback, and Real seal handoff.*
7. *C records displayed* Cont *replay for window inspection, tolerance comparison, monotone threshold use, RegSeqRat compatibility, and Real seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{FastCauchyCriterion}^{\uparrow}$*,* $\mathsf{FastCauchySequence}^{\uparrow}$*,* $\mathsf{FastCauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$ *row.*

The carrier exports no quotient equality of real names, no selected representative of a Cauchy class, no countable-choice schedule, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FastCauchyCriterionUp.FastCauchyCriterionTasteGate single carrier alignment`

Theorem 3825.2 (FastCauchyCriterion NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier* $K = (S, D, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$ *surface consists exactly of finite StreamName window admission, dyadic tolerance-scale admission, monotone fast-modulus threshold exposure, RegSeqRat compatibility, RealUp seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The rows S, D, M, R, E are the only analytic rows available to a criterion consumer, and they appear in the order needed by the fast Cauchy route: finite windows first, dyadic tolerances next, the monotone threshold ledger next, regular rational readback next, and the Real seal last.

Read the sibling routes through provenance. The row M is the visible threshold discipline of chapter 1781, while S, D, R are the finite window, dyadic tolerance, and regular rational surfaces already used by chapter 2185.

The structural rows do not add analytic content. The transport row H moves only the displayed coordinates componentwise, the replay row C repeats the same threshold and comparison route, and P, N source and name that finite surface. Thus no NameCert obligation remains outside $S, D, M, R, E, H, C, P, N$. □

Theorem 3825.3 (FastCauchyCriterion modulus readback)**.** *Let* $K = (S, D, M, R, E, H, C, P, N)$ *be an accepted* $\mathsf{FastCauchyCriterion}^{\uparrow}$ *carrier. Every requested precision read obtains its fast threshold from* M*, opens only the finite* $\mathsf{StreamName}^{\uparrow}$ *windows recorded in* S*, checks the dyadic bound through* D*, and hands the resulting regular schedule to* R *before any* $\mathsf{Real}^{\uparrow}$ *seal in* E *is consumed. The readback is therefore a finite criterion route over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows, not a selected representative or an ambient completeness argument.*

Proof. Start with the monotone modulus row. The request is sent to the displayed threshold in M, and monotonicity keeps later precision requests inside the same finite threshold ledger rather than invoking a choice sequence.

Open the finite windows through S and compare them through D. These rows provide exactly the inspected values and dyadic tolerance account needed for a fast Cauchy criterion, so the route remains inside the displayed BHist packet.

Finally read R and then E. The $\mathsf{RegSeqRat}^{\uparrow}$ compatibility row consumes the same windows and tolerances, and the $\mathsf{Real}^{\uparrow}$ handoff is downstream of that compatibility row. Componentwise transport, replay, provenance, and local naming preserve the same route and add no quotient equality, selected representative, countable choice, or ambient completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{FastCauchyCriterion}^{\uparrow}$ precision read exposes one finite StreamName window row, one dyadic tolerance-scale row, one monotone FastCauchyModulus row, one RegSeqRat compatibility row, one RealUp seal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer may read the criterion.

Independence witness: fastcauchysequence, fastcauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{FastCauchyCriterion}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries windows, tolerance scale, monotone threshold discipline, regular readback, Real seal handoff, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyCriterion}^{\uparrow}$ packet is generated from finite StreamName windows, a dyadic tolerance scale, a monotone FastCauchyModulus row, RegSeqRat compatibility, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3825.1, Definition 3825.2, and Definition 3825.3 seed the finite criterion surface connecting fast Cauchy sequence windows, fast modulus thresholds, RegSeqRat readback, and Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FastCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{FastCauchyCriterion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, monotone modulus exposure, dyadic tolerance readback, RegSeqRat compatibility, Real seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.FastCauchyCriterionUp

Chapter 3826

A Concrete Naming Certificate for CauchyCompletionKleisliCategory$^{\uparrow}$

Following Mac Lane's Kleisli-category formulation, the CauchyCompletionKleisliCategory$^{\uparrow}$ packet records the finite category-level surface of the L10 completion monad. It sits after the completion monad of chapter 378, the Kleisli-arrow packet of chapter 1845, the universal-property surface of chapter 397, and the finite real-name route through chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names a category of completion consumers through displayed finite rows; it is not a quotient completion, not a selected limit, not host equality of completed objects, and not ambient Real$^{\uparrow}$ completeness.

Definition 3826.1 (CauchyCompletionKleisliCategory carrier)**.** *A* CauchyCompletionKleisliCategory$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (O, U, A, B, E, R, H, C, P, N)$$

with the following displayed rows:

1. *O is the object row, reading each completion-facing object through* CauchyCompletionMonad$^{\uparrow}$ *and the regular rational name surface.*
2. *U is the unit-arrow row, recording the monad unit route from displayed source rows into the completion packet.*
3. *A is the Kleisli-arrow row, whose arrows are completion-valued maps exposed by* CauchyCompletionKleisli$^{\uparrow}$*.*
4. *B is the bind-composition row, recording composition only through the monad bind schedule and the displayed Kleisli route.*
5. *E is the identity and associativity ledger, comparing unit and bind composites as finite BEDC routes rather than as host function equalities.*
6. *R is the real-completion handoff row, routing completion consumers through* CompletionUniversalProperty$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*, and* Real$^{\uparrow}$ *seals.*
7. *H records componentwise* hsame *transport for object, unit, arrow, bind, ledger, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for unit-arrow reads, bind composition, completion consumers, and terminal* Real$^{\uparrow}$ *seal requests.*

9. *P is the* Pkg *provenance over* CauchyCompletionKleisliCategory$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionKl CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyCompletionKleisliCategory}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, selected limits, host equality of arrows, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CauchyCompletionKleisliCategoryUp.CauchyCompletionKleisliCategoryUp`

Theorem 3826.2 (CauchyCompletionKleisliCategory NameCert obligations)**.** *For every accepted* CauchyCompletionKleisliCategory$^{\uparrow}$ *carrier $K = (O, U, A, B, E, R, H, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionKleisli}}$ *surface consists exactly of object exposure, unit-arrow exposure, Kleisli-arrow exposure, bind-composition exposure, identity and associativity ledger exposure, real-completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3826.1. The object, unit, arrow, bind, ledger, and real-completion handoff rows are precisely O, U, A, B, E, R.

The unit-arrow row is read through CauchyCompletionMonad$^{\uparrow}$, while the arrow and bind rows are read through CauchyCompletionKleisli$^{\uparrow}$. The ledger E compares the category identities and associative composites only after those rows have been displayed.

The real-facing handoff R routes every completion consumer through CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ seals. Thus a consumer cannot read a terminal real-completion claim before the finite Cauchy-name and dyadic rows are visible.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since no quotient completion, selected representative, selected limit, host equality, or ambient completeness coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 3826.3 (CauchyCompletionKleisliCategory real-completion handoff)**.** *Every* Real$^{\uparrow}$ *or completion-facing consumer of an accepted* CauchyCompletionKleisliCategory$^{\uparrow}$ *packet factors through the ordered route*

CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionKleisli$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, D

The route may use unit arrows and bind composition inside the Kleisli category, but it exports no quotient completion, selected limit, host equality of arrows, selected representative, or ambient Real$^{\uparrow}$ *completeness principle.*

Proof. First read the monad and Kleisli rows. The unit arrows come from U, and the composition row B uses the bind schedule attached to the displayed Kleisli-arrow row A.

Next read the category ledger. The row E records the identity and associativity comparisons as finite route comparisons over O, U, A, B, so the categorical equations are consumed as BEDC ledger data rather than as host equality of functions.

Finally pass to real-completion consumers through R. The handoff row requires the universal-property surface and the finite name route through StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ before any terminal seal is available.

The structural rows H, C, P, N keep this route inside the displayed packet. There is no carrier coordinate that could supply a quotient completion, selected limit, host arrow equality, selected representative, or ambient completeness theorem. □

Falsifiable prediction: A $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ completion consumer that reads a $\mathsf{Real}^{\uparrow}$ seal without first exposing the monad, Kleisli-arrow, universal-property, stream-window, regular-rational, dyadic-tolerance, transport, replay, provenance, and local NameCert rows refutes the packet.

Independence witness: cauchycompletionmonad, cauchycompletionkleisli, completionuniversal-property, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ because the category packet jointly carries objects, unit arrows, Kleisli arrows, bind composition, categorical ledgers, completion handoff, transport, replay, provenance, and local naming in one finite carrier.

Lean follow-up requires a finite carrier namespace at `BEDC.Derived.CauchyCompletionKleisliCategoryUp` with the displayed object, unit, Kleisli-arrow, bind-composition, ledger, handoff, transport, replay, provenance, and naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}$ packet is generated from completion-object rows, unit-arrow rows, Kleisli-arrow rows, bind-composition rows, identity and associativity ledgers, real-completion handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3826.1, Definition 3826.2, and Definition 3826.3 seed the finite Kleisli-category completion packet for L10 consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionKleisliCategory}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality of arrows, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-arrow exposure, Kleisli-arrow exposure, bind-composition exposure, categorical ledger exactness, real-completion handoff, transport stability, replay stability, provenance exhaustion, and non-escape for BEDC.Derived.CauchyCompletionKleisliCategoryUp.

Chapter 3827

A Concrete Naming Certificate for MetaCICConfluenceResidualChecker$^{\uparrow}$

The MetaCICConfluenceResidualChecker$^{\uparrow}$ packet realizes the substitution-sensitive confluence frontier of chapter 9378 as finite checker data. It sits beside the substitution-boundary packet of chapter 3791, the bounded residual frontier of chapter 3808, the critical-pair boundary of chapter 2212, and the local diamond witness of chapter 3595. The packet records which typed windows, redex frontiers, local joins, obstruction rows, and replay routes have actually been checked before a confluence consumer may cite the conditional handoff.

Falsifiable prediction: A MetaCIC confluence read that treats a residual-substitution peak as checked while omitting the finite typed-window, redex-frontier, bounded-checker, local-join, obstruction, transport, replay, provenance, and local NameCert rows refutes MetaCICConfluenceResidualChecker$^{\uparrow}$.

Independence witness: metacic-confluence-substitution-boundary, bounded-residual-substitution-frontier, metacic-confluence-critical-pair-boundary, metacic-local-diamond-witness: none is bijective to MetaCICConfluenceResidualChecker$^{\uparrow}$ because the checker packet jointly carries typed windows, redex frontiers, bounded checker outcomes, local joins, obstruction retention, transport, replay, provenance, and local naming as one finite confluence frontier.

Definition 3827.1 (MetaCIC confluence residual-checker carrier)**.** *A* MetaCICConfluenceResidualChecker$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, R, B, J, O, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row containing the source term, the two parallel reducts, the residual substitution window, and the candidate common reduct.*
2. *R is the redex-frontier row, listing the inspected beta positions, binder depths, residual descendants, and free-variable exposure checks.*
3. *B is the bounded-checker row, recording the finite closedness, substitution-support, variable-escape, and branch-compatibility checks.*
4. *J is the local-join row, storing the finite replay path to the candidate common reduct when the checker accepts.*

5. *O is the obstruction row, naming the still-visible endpoints: unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, and kernel consistency.*
6. *S is the checker-status row, recording whether the inspected residual window is accepted, rejected, or left at the named obstruction boundary.*
7. *H records componentwise* hsame *transport for typed windows, redex frontiers, bounded checks, local joins, obstruction rows, checker status, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence-frontier query through $T, R, B, J, O, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$, $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceCriticalPairBoundary}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It contains no coordinate for full confluence, unrestricted residual-substitution compatibility, hidden conversion, evaluator equality, subject reduction, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceResidualCheckerUp.MetaCICConfluenceResidualCheckerTasteGate`
`single carrier alignment`

Theorem 3827.2 (MetaCIC confluence residual-checker obligations)**.** *For every accepted* MetaCICConfluenceResidu *carrier $Q = (T, R, B, J, O, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$ *surface consists exactly of typed-window exposure through T, redex-frontier exposure through R, bounded checking through B, local-join exposure through J, obstruction retention through O, checker-status exposure through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3827.1. The rows T, R, B, J, O, S are the checker-facing data, and the rows H, C, P, N are the structural certificate data.

The checker begins at T and R. These rows display the finite typed peak, the residual substitution window, the inspected redex positions, binder-depth ledger, and free-variable exposure checks.

The row B records the bounded decision surface, while J records only the local candidate join made available by that bounded surface. The checker-status row S separates accepted windows, rejected windows, and windows that remain at the named substitution obstruction.

The obstruction row O prevents the bounded read from being cited as global confluence. Transport, replay, provenance, and local naming preserve the same finite packet, and no hidden conversion, evaluator equality, subject reduction, or kernel-consistency coordinate is present. □

Theorem 3827.3 (MetaCIC confluence residual-checker handoff)**.** *Every confluence-audit consumer of an accepted* $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ *carrier factors through the finite route*

$$T, R, B \longmapsto J, S, O \longmapsto H, C, P, N.$$

The route may hand off a checked residual window to the substitution-boundary, bounded-frontier, critical-pair, and local-diamond packets named above, but it does not supply unrestricted substitution compatibility, a premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, or kernel consistency.

Proof. First read the finite window. The rows T and R expose the typed peak, residual substitution window, redex frontier, binder-depth ledger, and free-variable checks.

Next apply the bounded checker row B. If the finite checks accept, the local join row J and checker-status row S expose the handoff; if they reject or stop at the boundary, S and O keep the obstruction visible.

Finally route the handoff through H, C, P, N. These rows transport, replay, source, and name the same finite packet, and the provenance row points only to the substitution-boundary, bounded-frontier, critical-pair, and local-diamond packets.

The carrier therefore has no path to a global confluence theorem or hidden substitution theorem. Lean follow-up requires Nontrivial and FieldFaithful instances for `BEDC.Derived.MetaCICConfluenceResidualChec` at exactly this displayed carrier surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$ packet is generated from typed-window rows, redex-frontier rows, bounded-checker rows, local joins, checker-status rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3827.1, Definition 3827.2, and Definition 3827.3 seed the finite residual-substitution checker packet for MetaCIC confluence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject-reduction discharge, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window exposure, redex-frontier coverage, bounded-checker exactness, local-join exposure, checker-status determinacy, obstruction retention, transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MetaCICConfluenceResidualChec

Chapter 3828

A Concrete Naming Certificate for $\mathsf{DarbouxContinuity}^{\uparrow}$

Darboux/Bishop–Bridges constructive analysis treats interval continuity as a finite route from a closed bounded source interval, through an explicitly continuous graph, into bracketed intermediate-value data before any completed real image is read. The $\mathsf{DarbouxContinuity}^{\uparrow}$ packet records that route over chapter 1591, chapter 130, chapter 1402, chapter 291, chapter 301, and chapter 38. It is a local continuity certificate for L10 interval-image consumers, not an open-cover compactness principle, not quotient real equality, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3828.1 (DarbouxContinuity carrier)**.** *A* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, G, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval source row, carrying endpoint order, containment, finite-net source, and source-obligation data from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*.*
2. *G is the continuous-map graph row over the interval source, recording the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *modulus, graph, and classifier-compatible point route.*
3. *V is the intermediate-value crossing row, recording the admitted sign or order crossing data read through* $\mathsf{IntermediateValue}^{\uparrow}$*.*
4. *D is the dyadic bracket ledger from* $\mathsf{DyadicRatCore}^{\uparrow}$*, recording finite rational brackets for endpoints, graph samples, crossing windows, and tolerance budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the bracketed endpoint, sample, and crossing observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after I, G, V, D, R have been displayed.*
7. *H is the componentwise* hsame *row for the interval, graph, crossing, bracket, readback, seal, replay, provenance, and local naming data.*
8. *C records displayed* Cont *replay from any interval-continuity consumer through I, G, V, D, R, E and then through the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{DarbouxContinuity}^{\uparrow}$*,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{IntermediateValue}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for host open covers, selected subcovers, quotient real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.DarbouxContinuityUp.DarbouxContinuityTasteGate single carrier alignment`

Theorem 3828.2 (DarbouxContinuity NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$ *surface consists exactly of closed-interval source admission, continuous-map graph exposure, intermediate-value crossing exposure, dyadic bracket accounting, RegSeqRat readback, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier rows. The interval source I, graph row G, crossing row V, dyadic ledger D, readback row R, and seal row E are the only analytic rows available to a consumer.

Read the structural rows separately. The transport row H preserves those analytic rows componentwise, while C replays the same displayed route for interval-continuity consumers.

The provenance and naming rows P, N attach the packet to the cited closed-interval, continuous-map, intermediate-value, dyadic, regular-readback, and Real certificates. No excluded cover, quotient, trichotomy, choice, or completeness coordinate is present. □

Theorem 3828.3 (DarbouxContinuity source route)**.** *Every* $\mathsf{DarbouxContinuity}^{\uparrow}$ *consumer that reads an interval-image or crossing value must first enter through the closed bounded interval source of chapter 1591, then through the continuous graph of chapter 130, then through the crossing row of chapter 1402, and only then through the dyadic, RegSeqRat, and Real rows of chapter 291, chapter 301, and chapter 38.*

Proof. Start with the source interval row I. Its endpoint, containment, finite-net, and readback obligations supply the finite interval boundary before any graph or image row is available.

Next read the graph and crossing rows. The continuous-map row G supplies the displayed modulus and graph route, and V supplies the intermediate-value crossing data relative to that graph.

Finally expose D, R, E through the same route. The dyadic bracket ledger, regular readback, and Real seal are downstream of I, G, V, so a consumer cannot start from an ambient interval image or a completed real-value selection. □

Theorem 3828.4 (DarbouxContinuity graph exposure)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row* G *is exposed only after the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row* I *has been displayed. Any crossing read through* V *depends on that graph exposure and therefore cannot be read as an interval-image fact detached from the source interval and continuous-map graph.*

Proof. Project the carrier of Definition 3828.1. The row I is the source interval coordinate, and G is explicitly described as the continuous-map graph row over that source.

Now inspect the crossing row. The row V records intermediate-value crossing data read through the graph row G, so a consumer of V must have already displayed both the source interval row and the continuous graph row.

The structural rows H, C, P, N only transport, replay, source, and name this route. They add no detached image coordinate, so every accepted crossing read remains dependent on the I then G exposure order. □

Theorem 3828.5 (DarbouxContinuity Real-seal handoff)**.** *For every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *carrier* $T = (I, G, V, D, R, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *row* E *is terminal. It may be read only after the dyadic bracket ledger* D *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *expose the bracketed endpoint, sample, and crossing observations used by the intermediate-value route.*

Proof. Project the analytic rows. The dyadic ledger D stores the finite rational brackets, and the row R is the regular rational readback for those bracketed observations.

The seal row is downstream. In Definition 3828.1, E is exposed only after I, G, V, D, R have been displayed, so the completed real-name read cannot precede the bracket ledger or its regular readback.

Transport and replay preserve this terminal position. The rows H, C, P, N can reread the same sealed route for a consumer, but they cannot supply an ambient Real value without the displayed dyadic and regular-rational handoff. □

Theorem 3828.6 (DarbouxContinuity non-escape)**.** *Every accepted* $\mathsf{DarbouxContinuity}^{\uparrow}$ *packet exports only the finite rows*

$$I, G, V, D, R, E, H, C, P, N.$$

It exports no host interval image, open-cover compactness object, selected subcover, selected point, quotient real equality, trichotomy decision, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the packet and inspect the carrier inventory. The analytic rows are the closed interval source, continuous graph, intermediate-value crossing, dyadic bracket ledger, regular readback, and terminal Real seal.

The structural rows do not add analytic content. They only transport, replay, source, and name the same finite route through H, C, P, N.

Since no carrier coordinate stores host compactness, quotient equality, a selected point, trichotomy, choice, or ambient completeness, no accepted consumer can extract such data from the Darboux continuity certificate. □

Falsifiable prediction: An accepted $\mathsf{DarbouxContinuity}^{\uparrow}$ read is refuted if it exposes an interval-image or crossing value without first displaying the closed interval source, continuous graph, intermediate-value crossing, dyadic bracket, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: closedboundedinterval, continuousmap, intermediatevalue, dyadicratcore, regseqrat, real: none alone is bijective with $\mathsf{DarbouxContinuity}^{\uparrow}$ because the packet jointly records an interval source, continuous graph, crossing ledger, dyadic brackets, regular readback, Real seal, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DarbouxContinuity}^{\uparrow}$ packet is generated from a closed bounded interval source, a continuous-map graph, intermediate-value crossing data, a dyadic bracket ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 3828.1, Definition 3828.2, Definition 3828.3, and Definition 3828.6 bind the L10 interval-continuity carrier to closed-interval, continuous-map, intermediate-value, dyadic, regular-readback, and Real-seal source rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DarbouxContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, quotient real equality, trichotomy, selected interval-image points, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, graph exposure, crossing-route exactness, dyadic bracket readback, Real-seal handoff, and non-escape for BEDC.Derived.DarbouxContinuityUp.

Chapter 3829

A Concrete Naming Certificate for $\mathsf{ConcaveModulus}^{\uparrow}$

Following Bishop and Bridges on constructive moduli of continuity, the $\mathsf{ConcaveModulus}^{\uparrow}$ packet records the finite normalization surface at which a positive tolerance schedule is replaced by a monotone midpoint-concave modulus before Real or Completion consumers inspect it. It sits on the rational and dyadic tolerance rows of chapter 37 and chapter 291, the explicit uniform-modulus surface of chapter 2228, the located-uniform-continuity handoff of chapter 1859, and the terminal Real seal of chapter 38. The packet names a constructive normalization object; it does not choose a modulus from a host theorem, assert real completeness, or import quotient equality of functions.

Definition 3829.1 (Concave modulus carrier)**.** *A* $\mathsf{ConcaveModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, D, M, J, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row. It names the rational tolerance requests and records the finite order in which a consumer may inspect them.*
2. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same requests, giving the dyadic scale at which Real and Completion routes read the modulus.*
3. *M is the monotone modulus row. It records that weaker tolerance requests are served by no stronger source radius than the displayed stronger requests.*
4. *J is the midpoint-concavity row. It records the finite Jensen midpoint comparisons used by the normalization route before any subadditivity consumer may read the packet.*
5. *S is the subadditive handoff row. It exposes the normalized modulus as the public row consumed by* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *and* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *certificates.*
6. *H stores componentwise* hsame *transport for the rational tolerance, dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a modulus-normalization request through Q, D, M, J, S, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ConcaveModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformContinuityModulus}^{\uparrow}$, $\mathsf{LocatedUniformContinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected modulus, host continuity theorem, quotient function equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusTasteGate single carrier alignment`

Theorem 3829.2 (Concave modulus NameCert obligations)**.** *For every accepted* $\mathsf{ConcaveModulus}^{\uparrow}$ *carrier*

$$K = (Q, D, M, J, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the positive rational tolerance, positive dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, transport, replay, provenance, and local naming rows;*
2. *tolerance exposure, requiring every normalization request to read* Q *and* D *before it reads any modulus row;*
3. *monotonicity discipline, requiring* M *to preserve the displayed order of tolerance requests without choosing a new host-scale modulus;*
4. *midpoint-concavity discipline, requiring* J *to supply the finite midpoint comparisons used by the subadditive handoff;*
5. *non-escape, requiring* S, H, C, P, N *to export only the displayed normalized modulus route and no choice-selected modulus, quotient function equality, trichotomy, ambient completeness, Lean verification, or bridge checking.*

No $\mathsf{ConcaveModulus}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusCarrier namecert obligations`

Proof. Project the carrier of Definition 3829.1. The tuple lists exactly the positive rational tolerance, dyadic tolerance, monotone modulus, midpoint-concavity, subadditive handoff, and structural rows.

Tolerance exposure is forced by the order of Q and D. A consumer cannot inspect the normalized modulus before the positive rational request and dyadic scale have been displayed as finite packet data.

The monotone and midpoint-concavity obligations are the rows M and J. They are read before the public handoff row S, so the handoff is a displayed normalization route rather than a host theorem about all moduli.

Finally H, C, P, N transport, replay, source, and name the same finite rows. Since the carrier contains no coordinate for a selected modulus, quotient function equality, trichotomy, ambient completeness, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 3829.3 (Concave modulus subadditive handoff)**.** *Every* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *or* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *consumer of an accepted* $\mathsf{ConcaveModulus}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Rat}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{ConcaveModulus}^{\uparrow} \longrightarrow \mathsf{UniformContinuityModulus}^{\uparrow} \longrightarrow \mathsf{LocatedUniformContinuity}^{\uparrow}$$

through the displayed rows $Q, D, M, J, S, H, C, P, N$*. The subadditive handoff row* S *is available only after the positive tolerance rows, monotone modulus row, and midpoint-concavity row have been displayed. The route exports no choice-selected modulus, host continuity theorem, quotient equality of functions, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with an accepted carrier $K = (Q, D, M, J, S, H, C, P, N)$. Projection exposes the tolerance rows Q, D before any modulus-normalization row is consumed.

Read the monotone row M and midpoint-concavity row J. These rows are finite packet data, so the consumer receives the normalization discipline as displayed comparisons rather than as a global host theorem.

The handoff row S is downstream of those rows. A $\mathsf{UniformContinuityModulus}^{\uparrow}$ consumer reads S as its public tolerance-to-radius surface, and a $\mathsf{LocatedUniformContinuity}^{\uparrow}$ consumer reads the same handoff only after its located source and modulus routes have been displayed.

The remaining rows H, C, P, N preserve componentwise transport, replay, provenance, and naming for this route. Since no listed row contains a selected modulus, quotient function equality, trichotomy, or ambient completeness coordinate, the subadditive handoff exports none of them. □

Lean4 ✓ `BEDC.Derived.ConcaveModulusUp.ConcaveModulusCarrier subadditive handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3829.1, Definition 3829.2, and Definition 3829.3 seed the finite constructive normalization route from positive rational and dyadic tolerances through monotone midpoint-concave modulus rows into uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConcaveModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a choice-selected modulus, host continuity theorem, quotient function equality, real trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive tolerance exposure, monotone modulus discipline, midpoint-concavity rows, subadditive handoff, transport, replay, provenance, and local naming for BEDC.Derived.ConcaveModulusU

Constructive story: A $\mathsf{ConcaveModulus}^{\uparrow}$ packet is generated from positive RatUp tolerance rows, positive DyadicRatCoreUp tolerance rows, monotone modulus rows, midpoint-concavity rows, a subadditive handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3830

A Concrete Naming Certificate for $\mathsf{SignedDigitIntervalReal}^{\uparrow}$

Following Avizienis and Bishop–Bridges, the $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ packet records a closed real interval whose endpoints are supplied by signed-digit finite-window streams, normalized dyadic endpoint rows, StreamName schedules, RegSeqRat readbacks, and a terminal RealUp seal. It consumes the signed-digit stream surface of chapter 316, the signed-digit real endpoint route of chapter 363, the dyadic normalization ledger of chapter 291, the finite schedule interface of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the closed bounded interval carrier of chapter 1591. The packet gives Real and Completion consumers a finite-window interval source without exporting a quotient stream, a selected infinite expansion, or ambient real completeness.

Definition 3830.1 (Signed digit interval real carrier)**.** *A* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower signed-digit endpoint row, carrying the finite signed-digit windows admitted by the signed-digit stream and signed-digit real routes.*
2. *U is the upper signed-digit endpoint row, carrying the corresponding finite signed-digit windows for the upper endpoint.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *normalization ledger, recording the dyadic endpoint rows obtained from the inspected signed-digit windows.*
4. *O is the endpoint-order row, recording that the normalized lower endpoint is contained below the normalized upper endpoint through the displayed dyadic comparisons.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule, listing exactly the endpoint windows and precision requests inspected by the packet.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the normalized lower and upper endpoint observations.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the signed-digit endpoint rows, dyadic ledger, order row, schedule, and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint rows, dyadic ledger, order row, schedule, readback, seal, replay, provenance, and local naming row.*

9. *C records displayed* Cont *replay from signed-digit windows through dyadic normalization, endpoint ordering, regular readback, and the RealUp seal.*
10. *P is the* Pkg *provenance over* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, $\mathsf{SignedDigitReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, D, O, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected infinite signed-digit expansion, real trichotomy, a selected point of the interval, open-cover compactness, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SignedDigitIntervalRealUp.SignedDigitIntervalRealTasteGate single carrier alignment`

Theorem 3830.2 (Signed digit interval real NameCert obligations)**.** *For every accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier*

$$I = (L, U, D, O, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$ *obligations are exhausted by carrier admission for the eleven displayed rows; signed-digit endpoint normalization through L, U, D; dyadic containment through the normalized endpoint ledger D; endpoint-order preservation through O; regular readback through R along the finite schedule S; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

No obligation may call on quotient stream equality, a selected infinite expansion, trichotomy, a selected point of the interval, open-cover compactness, countable choice, or ambient RealUp completeness.

Proof. Project the carrier of Definition 3830.1. The endpoint rows L and U are the only signed-digit sources available to the packet, and the schedule row S records the finite windows through which those sources are inspected.

The dyadic and order obligations are then read through D and O. The ledger D normalizes the inspected endpoint windows into dyadic rows, while O records the displayed lower-before-upper comparison needed by the closed interval carrier.

The regular readback and seal obligations pass through R and E. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local certificate and leave the excluded principles outside the packet. □

Theorem 3830.3 (Signed digit interval real seal nonescape)**.** *An accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier exports its interval-facing RealUp seal only through*

$$L, U, D, O, S, R, E, H, C, P, N.$$

Consequently every consumer of the packet receives a finite signed-digit endpoint route, dyadic containment ledger, endpoint-order row, regular readback, and closed-interval seal; it receives no quotient stream equality, selected infinite expansion, trichotomy, selected point of the interval, open-cover compactness, countable choice, or ambient RealUp completeness.

Proof. Start with the endpoint rows. The lower and upper endpoints are visible only as the finite signed-digit windows displayed by L and U along the schedule S.

Normalize those endpoint windows through D and compare them through O. This is the only place where the carrier records containment of the lower endpoint below the upper endpoint, and it is expressed as a finite dyadic ledger rather than as trichotomy or quotient real equality.

The RealUp-facing read then passes through R, E, H, C, P, N. Since the carrier has no further public coordinates, any interval consumer is confined to the displayed finite route and none of the excluded selection, compactness, choice, or completeness principles can escape through the seal. □

Theorem 3830.4 (Signed digit interval real endpoint-order nonescape)**.** *Let*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier. Any consumer read that uses interval containment or the lower-before-upper comparison must expose the finite signed-digit endpoint rows* L, U*, the finite schedule* S*, the dyadic normalization ledger* D*, and the endpoint-order row* O *before it can pass to the readback and seal rows* R, E*.*

Consequently the order-facing read exports only finite dyadic endpoint comparison data. It exports no real trichotomy, quotient real equality, selected endpoint, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier displayed in Definition 3830.1. The rows L and U are the only endpoint-window sources, and S is the only row that records which finite windows and precision requests the packet has inspected.

The order-facing data then factors through D and O. The ledger D is the unique dyadic normalization row for the inspected endpoint windows, while O is the only coordinate that records the lower-before-upper comparison.

The rows R, E, H, C, P, N are downstream readback, seal, transport, replay, provenance, and naming rows. They add no further order coordinate, so every public order-facing read remains confined to the finite endpoint windows, their dyadic normalization ledger, and the displayed endpoint-order row. □

Theorem 3830.5 (Signed digit interval real refinement stability)**.** *Let*

$$I = (L, U, D, O, S, R, E, H, C, P, N)$$

and

$$I' = (L', U', D', O', S', R', E', H', C', P', N')$$

be accepted $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carriers whose displayed rows are related componentwise by the transport row* hsame *on* L, U, D, O, S*. For any smaller signed-digit endpoint window and any dyadic tolerance refinement admitted by the finite schedule, the refined read factors again through the rows*

$$L, U, D, O, S, R, E, H, C, P, N.$$

The factorization preserves the endpoint-order row and the terminal seal without using trichotomy, selected signed-digit expansions, quotient stream equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the two accepted carriers. The componentwise hsame data identifies the endpoint rows L, U, the normalization ledger D, the endpoint-order row O, and the schedule row S as the only coordinates available for comparing the original and refined reads.

Refine the schedule row S to the smaller endpoint windows and requested dyadic tolerance. The refined observations are still signed-digit windows from L and U, so their normalization is replayed through the same dyadic ledger shape D rather than through a quotient stream or a selected expansion.

The order row O is transported along the displayed hsame data. Thus the refined dyadic endpoint comparison is the same finite lower-before-upper certificate, only read at the smaller window and tolerance.

Finally repack the refined route through R, E, H, C, P, N. The readback, seal, transport, continuation, provenance, and naming rows carry the refined finite route and add no coordinate from trichotomy, choice, compactness, quotient equality, or ambient completeness. □

Theorem 3830.6 (Signed digit interval real RegSeqRat seal route)**.** *Every Real-facing endpoint or interval read of an accepted* $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ *carrier must pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *normalization ledger D, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E. The route exports no selected infinite signed-digit expansion and no completeness principle.*

Proof. Use the carrier of Definition 3830.1. Its only Real-facing coordinates are arranged after the endpoint rows, the schedule, the dyadic normalization ledger, and the regular rational readback row.

Apply Definition 3830.2. The local certificate lists regular readback along S and terminal RealUp sealing through E as obligations downstream of the signed-digit endpoint and dyadic normalization rows.

Apply Definition 3830.3. Since the seal exports only the displayed route $L, U, D, O, S, R, E, H, C, P, N$, any endpoint or interval read must pass through S, D, R before E and cannot expose an infinite expansion, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SignedDigitIntervalReal}^{\uparrow}$ packet is generated from lower and upper signed-digit endpoint rows, a DyadicRatCore normalization ledger, an endpoint-order row, StreamName finite schedules, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3830.1, Definition 3830.2, and Definition 3830.3 seed the finite signed-digit endpoint route for closed RealUp intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDigitIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite signed-digit expansions, trichotomy, selected interval points, open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint normalization, dyadic containment, endpoint-order preservation, regular readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SignedDigitIntervalRealUp.

Chapter 3831

A Concrete Naming Certificate for $\mathsf{CayleyTheorem}^{\uparrow}$

Following Cayley, the $\mathsf{CayleyTheorem}^{\uparrow}$ packet records the finite left-regular action of a displayed group row as a permutation-action row. It consumes the finite $\mathsf{Group}^{\uparrow}$ source of chapter 41 and the finite-position bijection surface of chapter 116. The packet names the carrier, action, image, faithfulness, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows needed to read the theorem as BEDC data; it does not assert a classification of finite groups, a presentation by generators and relations, or a representation-theoretic decomposition.

Definition 3831.1 (Cayley theorem carrier)**.** *A* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (G, L, I, F, H, T, P, N)$$

with the following displayed rows:

1. *G is the consumed finite* $\mathsf{NameCert}_{\mathsf{Group}^{\uparrow}}$ *row, including the carrier, multiplication, unit, inverse, and finite support surfaces;*
2. *L is the left-regular action row sending a displayed group element g and a displayed group endpoint x to the product endpoint gx;*
3. *I is the permutation-image row, reading each left-translation row as a certified self-bijection of the finite carrier of G through the* $\mathsf{Permutation}^{\uparrow}$ *interface;*
4. *F is the faithfulness row, recording that if two left-translation rows have the same action on every displayed endpoint, then their source group rows classify together by the* $\mathsf{Group}^{\uparrow}$ *classifier;*
5. *H is the componentwise* hsame *transport for G, L, I, F;*
6. *T stores the displayed* Cont *routes through which consumers read the left action, permutation image, and faithfulness rows;*
7. *P is the* Pkg *provenance over* $\mathsf{Group}^{\uparrow}$, $\mathsf{Permutation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host embedding into an ambient symmetric group, no quotient by an external equality of functions, and no representation category.

Lean4 ✓ `BEDC.Derived.CayleyTheoremUp.TasteGate.CayleyTheoremTasteGate single carrier alignment`

Theorem 3831.2 (Cayley theorem NameCert obligations)**.** *For every accepted* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier*

$$C = (G, L, I, F, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite group row, left-regular action row, permutation-image row, faithfulness row, componentwise transport row, continuation routes, provenance, and local naming row. The obligation route is finite group row, left-regular action row, permutation image row, faithfulness row, and then the* hsame*,* Cont*,* Pkg*, and* NameCert *packaging rows.*

Proof. Project the carrier of Definition 3831.1. Carrier admission is the displayed tuple G, L, I, F, H, T, P, N, so no host symmetric group, host function equality, or representation category is available as an additional coordinate.

Read L from the multiplication row of G. For each displayed source row g, left multiplication by g sends every displayed endpoint x to the endpoint gx and sends the inverse endpoint through the inverse row of G, so the action has the inverse-witness data needed by the permutation surface.

Read I through chapter 116. The finite support row of G supplies the finite spine, while the left-action and inverse rows supply the forward graph, inverse graph, inverse witnesses, and action ledger of the carried $\mathsf{Permutation}^{\uparrow}$ row.

Finally read F at the unit endpoint of G. If two left-translation rows act the same on every displayed endpoint, their actions on the unit endpoint classify the two source group rows together. Repacking this faithfulness row with H, T, P, N gives the stated local NameCert surface. □

Theorem 3831.3 (Cayley theorem permutation-action handoff)**.** *Every downstream consumer of a* $\mathsf{CayleyTheorem}^{\uparrow}$ *carrier that asks for a permutation action of a finite group reads exactly the rows*

$$G, L, I, F, H, T, P, N$$

of Definition 3831.1. The handoff exports the consumed $\mathsf{Group}^{\uparrow}$ *row, the left-regular action row, the* $\mathsf{Permutation}^{\uparrow}$ *image row, and the faithfulness row, and then transports them through the displayed* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It does not export a bare host function, a separate symmetric-group carrier, or any representation-theoretic decomposition.*

Proof. First apply Definition 3831.2. It identifies the public packet with the finite group row, left-regular action row, permutation-image row, faithfulness row, and the transport and provenance rows.

Next project the permutation-image coordinate through chapter 116. Each left-translation row is read as a certified finite-position self-bijection; the action row is the corresponding endpoint action, and inverse coverage is supplied by the inverse row of the consumed group packet.

Finally use the faithfulness row. The action on the unit endpoint recovers the source group classifier, so the consumer can read the group-to-permutation map as faithful without importing host function equality. Since every public field has already been listed in G, L, I, F, H, T, P, N, the handoff has exactly the displayed BEDC surface and no additional symmetric-group or representation coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3831.1, Definition 3831.2, and Definition 3831.3 seed the finite left-regular action carrier, permutation-image handoff, faithfulness row, and hsame/Cont/Pkg/NameCert packaging over displayed $\mathsf{Group}^{\uparrow}$ and $\mathsf{Permutation}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CayleyTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a classification of finite groups, generator-and-relation presentations, symmetric-group closure, representation decomposition, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, left-action exactness, permutation-image construction, faithfulness at the unit endpoint, and downstream permutation-action consumer boundary for the same finite packet.

Constructive story: A CayleyTheorem$^{\uparrow}$ packet is generated from a displayed finite group source, its left-regular action on its own finite carrier, the induced certified permutation row for each source element, a faithfulness read at the unit endpoint, componentwise transport, continuation routes, provenance, and local NameCert evidence.

Chapter 3832

A Concrete Naming Certificate for $\mathsf{CauchyMeshRefinement}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy data and regular rational approximations, the $\mathsf{CauchyMeshRefinement}^{\uparrow}$ packet records a finite common-refinement mesh for two displayed Cauchy requests before a terminal real-name seal is read. It sits between the finite-window gateway of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal seal of chapter 38, and the tail-budget surface of chapter 2071. The packet supplies later diagonal, tail-budget, and regular-Cauchy arguments with finite mesh data; it does not introduce quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3832.1 (Cauchy mesh refinement carrier)**.** *A* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *Q_0 and Q_1 are the two source Cauchy request rows. Each records the source name, requested precision, finite tail index, and the local tolerance budget visible to the request.*
2. *D_0 and D_1 are the two dyadic mesh rows. They display the finite dyadic partitions*

$$D_i = \{a_{i,0} < a_{i,1} < \cdots < a_{i,k_i}\}$$

   for $i = 0, 1$, together with the endpoint, radius, and comparison cells supplied by $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *J is the common refinement row. It lists the finite dyadic mesh containing every point of D_0 and D_1, the inclusion maps from both meshes, and the comparison ledger showing that the inherited tolerances are below the two source requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source schedules inspected along J.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same displayed windows and the common dyadic refinement.*
6. *T is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row recording that the inherited tail budget is read from J, not from an unlisted completion object.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $Q_0, Q_1, D_0, D_1, J, S, R, T$ have been displayed.*

8. *H records componentwise* hsame *transport for the source requests, dyadic meshes, common refinement, StreamName windows, RegSeqRat readback, tail-bound row, real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay from a precision request through*

$$Q_0, Q_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto S \longmapsto R \longmapsto T \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyMeshRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$ *row.*

The classifier accepts only packets in which both displayed meshes factor through the same finite row J and every downstream read of S, R, T, E, H, C, P, N preserves that common refinement. No coordinate stores a quotient stream, selected limit, countable-choice schedule, trichotomy oracle, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyMeshRefinementUp.CauchyMeshRefinementTasteGate single carrier alignment`

Theorem 3832.2 (Cauchy mesh refinement NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier*

$$M = (Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$ *surface consists exactly of source Cauchy request admission, two displayed dyadic mesh ledgers, common-refinement inclusion,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyTailBound}^{\uparrow}$ *budget routing, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows $Q_0, Q_1, D_0, D_1, J, S, R, T, E, H, C, P, N$.*

Proof. Project the carrier of Definition 3832.1. The two source requests are visible only as Q_0, Q_1, and their dyadic data are visible only as the finite mesh rows D_0, D_1.

The common row J is the only place where the two meshes are compared. Since the inclusion maps and inherited tolerance checks are stored there, every later StreamName, RegSeqRat, and tail-budget consumer must read the same finite refinement before it can inspect a Real-facing seal.

The rows H, C, P, N transport, replay, source, and name this route. The carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, so the listed rows exhaust the certificate surface. □

Theorem 3832.3 (Cauchy mesh refinement RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *carrier must factor through the displayed route*

$$Q_0, Q_1 \longmapsto D_0, D_1 \longmapsto J \longmapsto S \longmapsto R \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Consequently a regular rational readback cannot be replaced by a quotient stream, selected limit, countable-choice schedule, trichotomy decision, host real equality, detached tail witness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness argument.*

Proof. Begin with a consumer request for regular rational, tail-budget, or real-seal data. The source Cauchy requests first expose Q_0, Q_1, and the dyadic rows D_0, D_1 display the two finite meshes answering those requests.

The common refinement row J is then forced by carrier admission. It is the unique finite row through which both dyadic meshes pass, so the StreamName windows S and the RegSeqRat readback R are read from the same mesh surface.

The tail-bound row T and Real seal row E are downstream of that readback. Transport, replay, provenance, and local naming preserve the displayed route, and the excluded analytic objects are absent from the tuple. Thus every consumer handoff is the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyMeshRefinement}^{\uparrow}$ packet is generated from two source Cauchy requests, two displayed dyadic meshes, a common refinement row, StreamName windows, RegSeqRat readback, a CauchyTailBound budget row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3832.1, Definition 3832.2, and Definition 3832.3 seed the finite common-refinement mesh surface for L10 Cauchy requests over StreamName, RegSeqRat, DyadicRatCore, CauchyTailBound, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMeshRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-request admission, dyadic mesh ledgers, common-refinement inclusion, StreamName windows, RegSeqRat readback, CauchyTailBound routing, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMeshRefinementUp.

Chapter 3833

A Concrete Naming Certificate for $\mathsf{MonotoneConvergenceReal}^{\uparrow}$

Following Bishop and Bridges on bounded monotone real sequences, the $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ packet records the finite NameCert surface by which monotone inspection rows, bounded sequence rows, and located-supremum rows meet before any ambient completeness theorem for $\mathsf{Real}^{\uparrow}$ is available. It consumes the bounded sequence carrier of chapter 1415, the located upper-bound surface of chapter 1427, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a located-supremum route for bounded monotone RealUp sequences; it is not a proof of full RealUp completeness, a host supremum construction, a trichotomy decision, quotient real equality, countable choice, or a selected limit operator.

Definition 3833.1 (Monotone convergence real carrier)**.** *A* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, O, U, A, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row, exposing only the finite sequence schedule, stream windows, regular readback, RealUp seal, and closed bounded interval-net bound already named in Definition 1415.1.*
2. *O is the monotonicity ledger, recording for the inspected index pairs that the displayed sequence windows are nondecreasing through the same comparison route used by the bounded sequence carrier.*
3. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *upper-bound row attached to the bounded range displayed by B and to the comparison rows recorded in O.*
4. *A is the lower-approximation ledger, saying that each requested tolerance window receives a displayed sequence index whose regular-rational readback approaches the located-supremum row from below.*
5. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *precision-window row used to inspect the source sequence, monotonicity ledger, upper-bound row, and approximation ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected windows in W.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, O, U, A, W, R have been displayed.*

8. *H records componentwise* hsame *transport for the bounded source, monotonicity ledger, located-supremum row, approximation ledger, precision windows, regular readback, real seal, replay, provenance, and local naming row.*
9. *C records displayed* Cont *replay from bounded sequence inspection through monotonicity, located-supremum access, lower approximation, regular readback, and the RealUp seal.*
10. *P is the* Pkg *provenance over* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $B, O, U, A, W, R, E, H, C, P, N$. It has no coordinate for full $\mathsf{Real}^{\uparrow}$ *completeness, a host-level supremum, trichotomy, quotient real equality, countable choice, or a selected-limit functional.*

Lean4 ✓ `BEDC.Derived.MonotoneConvergenceRealUp.MonotoneConvergenceRealTasteGate single carrier alignment`

Theorem 3833.2 (Monotone convergence real NameCert obligations)**.** *For every accepted* MonotoneConvergenceRe *carrier $M = (B, O, U, A, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}^{\uparrow}}$ *obligations are exhausted by the following finite rows: carrier admission for the eleven displayed coordinates; bounded-source exposure through B; monotonicity inspection through O; located upper-bound access through U; lower-approximation cofinality through A; shared finite-window accounting through W; regular-rational readback through R; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

No obligation may call on a full completeness principle for $\mathsf{Real}^{\uparrow}$*, a host supremum object, trichotomy, quotient real equality, countable choice, or a selected limit.*

Proof. Project the carrier of Definition 3833.1. The source sequence is visible only through the bounded sequence row B, and the monotonicity information is visible only through the displayed comparison ledger O.

The located-supremum side is then read through U and A. The row U supplies the upper-bound surface, while A supplies the lower approximants requested by the finite window row W.

Finally the readback and seal rows R and E repack the same inspected data as regular-rational and RealUp observations. The structural rows H, C, P, N transport, replay, source, and name this route, so the listed rows exhaust the local certificate. □

Theorem 3833.3 (Monotone convergence real located-supremum route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ *carrier reaches the candidate monotone limit only through*

$$B, O, U, A, W, R, E, H, C, P, N.$$

The route first reads the bounded real sequence row, then the monotonicity ledger, then the located-supremum upper-bound and lower-approximation rows, then the shared finite windows and regular rational readback, and only then the terminal RealUp seal. Consequently the consumer receives a located-supremum route for the displayed bounded monotone sequence rather than full RealUp completeness, a host supremum, trichotomy, quotient real equality, countable choice, or a selected limit.

Proof. Start with the bounded sequence surface. By Definition 1415.1, a downstream consumer can inspect only the finite sequence schedule, stream windows, regular readback, real seal, closed bounded interval-net bound, and structural naming rows carried by B.

Add the monotonicity ledger O. It compares only the inspected windows of the same bounded source, so it cannot introduce an unbounded stream, a quotient sequence, or a host order principle.

Now apply the located-supremum surface of chapter 1427. The rows U and A provide upper-bound access and lower-approximation cofinality for the same windowed route.

The final handoff passes through W, R, E, H, C, P, N. Since these rows are the only remaining public coordinates, the RealUp consumer obtains only the stated located-supremum route and none of the excluded completeness or choice principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}\uparrow}$.

Constructive story: A $\mathsf{MonotoneConvergenceReal}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a monotonicity ledger, a LocatedSupremumUp upper-bound row, a lower-approximation ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3833.1, Definition 3833.2, and Definition 3833.3 seed the finite located-supremum route for bounded monotone RealUp sequences.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceReal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close full RealUp completeness, a host supremum construction, trichotomy, quotient real equality, countable choice, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source exposure, monotonicity ledger stability, located-supremum access, lower-approximation cofinality, readback sealing, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneConvergenceRealUp.

$\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ reads unit and multiplication rows through chapter 423, chapter 3836, and chapter 397.

Chapter 3834

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$

Following Mac Lane's monad-law ledger and Bishop–Bridges constructive completion, the CauchyCompletionMonadL packet records the finite BEDC rows by which the Cauchy-completion transformer exposes its unit and associativity laws. It reads the completion monad of chapter 378, the unit route of chapter 1527, the bind ledger of chapter 3777, the idempotence packet of chapter 2381, and the real-and-completion readbacks of chapter 126, chapter 291, chapter 301, and chapter 38. The unit and multiplication law rows consume the functorial completion surface of chapter 423 and the universal-property witness of chapter 397 before associativity is read as law data. The chapter names a law surface for finite completion packets only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

As a dependency relation, the monad-law packet consumes the CauchyCompletionAssociativity route in chapter 1635 and the regular completion route in chapter 1639 before exporting unit and bind compatibility.

Definition 3834.1 (CauchyCompletionMonadLaw carrier)**.** *A* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, U, I, B, S, D, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *finite completion transformer row.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *source-admission row for constant and displayed regular-rational inputs.*
3. *I is the* $\mathsf{CompletionIdempotence}^{\uparrow}$ *row recording right-unit reuse of already completed names.*
4. *B is the bind-association ledger data supplied by the* $\mathsf{CauchyCompletionBind}^{\uparrow}$ *schedule route.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window readback shared by the unit, bind, and idempotence rows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and observation ledger read by those windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
8. *H is the componentwise* hsame *transport row for the monad, unit, idempotence, bind, stream, dyadic, and regular-rational rows.*

9. *C is the displayed* Cont *replay row forcing every law-facing consumer through M, U, I, B, S, D, R, H in that order.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and never through quotient completions, selected representatives, host equality, or an ambient limit object.

Lean4 (definition) `BEDC.Derived.Cauchy completion monad lawUp.CauchyCompletionMonadLawCarrier`

Theorem 3834.2 (CauchyCompletionMonadLaw NameCert obligations)**.** *For every accepted* CauchyCompletionMo *carrier $L = (M, U, I, B, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *surface consists exactly of these rows:*

1. *carrier admission for the monad transformer, unit admission, idempotence reuse, bind-association ledger, stream windows, dyadic observations, regular-rational readback, componentwise transport, replay, provenance, and local name;*
2. *left-unit exactness, requiring a unit-fed bind read to return through the same displayed S, D, R route before the* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
3. *right-unit and idempotence exactness, requiring a completed read followed by the unit/idempotence route to reuse the same M, I, S, D, R rows;*
4. *associativity exactness, requiring the two two-step bind bracketings to expose the same B, S, D, R, H, C, P, N ledger;*
5. *classifier transport for all law rows by componentwise* hsame *and displayed* Cont *replay;*
6. *non-escape of quotient completion classes, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No monad-law obligation is stored outside this finite carrier.

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadLawUp.CauchyCompletionMonadLawCarrier namecert obligations`
↪variant `BEDC.Derived.Cauchy completion monad lawUp.CauchyCompletionMonadLawNamecertObligations`

Proof. Project the carrier of Definition 3834.1. Carrier admission and classifier transport are exactly the displayed rows $M, U, I, B, S, D, R, H, C, P, N$.

For the unit rows, read U before any bind or terminal seal. The left-unit route passes from U through the shared stream, dyadic, and regular-rational readback rows S, D, R, while the right-unit route reads I as the idempotence reuse of the same completion-facing data.

For associativity, use the bind ledger B together with H and C. Both bracketings must be replayed through the same displayed schedule, finite windows, dyadic observations, and regular-rational readback before the provenance and local naming rows are consumed.

The non-escape clause follows by exhaustion. The carrier has no coordinate for quotient completion classes, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient completeness, so none of those objects can be exported by the local certificate. □

Theorem 3834.3 (CauchyCompletionMonadLaw unit-associativity boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the unit rows and the associativity rows through the finite carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$*. In particular, left-unit, right-unit, idempotence, and associativity claims reach* $\mathsf{Real}^{\uparrow}$ *only after the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$*,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$*,* $\mathsf{CompletionIdempotence}^{\uparrow}$*,* $\mathsf{CauchyCompletionBind}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows have been displayed. No quotient completion, selected limit, host equality, hidden stream representative, or ambient completeness datum is available as a second route.*

Proof. Start from a $\mathsf{Real}^{\uparrow}$-facing read. By Definition 3834.2, the read must first pass through carrier admission and then through the law row relevant to the requested unit or associativity claim.

For left-unit and right-unit reads, the route is forced through U and I respectively, with S, D, R supplying the finite stream, dyadic, and regular-rational readbacks before any terminal real seal can be inspected.

For associativity reads, the route is forced through the bind-association ledger B. The replay row C and transport row H keep the two bracketings on the same displayed schedule, window, observation, and readback surface.

Thus every real-facing law consumer uses the finite packet rows $M, U, I, B, S, D, R, H, C, P, N$. Since the packet contains no second coordinate for quotient completion, selected limits, host equality, hidden representatives, or ambient completeness, those objects cannot serve as alternate witnesses for the monad laws. □

Theorem 3834.4 (CauchyCompletionMonadLaw completion-reflection route)**.** *Every unit, multiplication, idempotence, or associativity read exported by an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet first consumes the completion-reflection route through chapter 423, chapter 397, and chapter 1506. Only after those sibling rows have supplied functorial completion, universal-property reflection, and metric-completion admission may the finite law rows* $M, U, I, B, S, D, R, H, C, P, N$ *be read as monad-law data. The theorem is a finite dependency route; it does not identify quotient completions, select representatives, or assert equality of completed objects.*

Proof. Begin with the carrier of Definition 3834.1. Its law rows expose the monad transformer, unit, idempotence, bind ledger, stream windows, dyadic observations, regular-rational readback, transport, replay, provenance, and local name.

Apply the NameCert obligations of Definition 3834.2. A law-facing consumer can read left-unit, right-unit, idempotence, or associativity only through the displayed carrier and through the finite readback rows that precede the terminal $\mathsf{Real}^{\uparrow}$ seal.

The completion-reflection route supplies the required sibling boundary before those law rows are exported. The completion functor gives the functorial completion surface, the universal-property chapter gives the reflection witness, and the metric-completion chapter gives the completion admission surface consumed by both.

Apply Definition 3834.3. It rules out quotient completion, selected limits, host equality, hidden representatives, and ambient completeness as alternate witnesses, so the completion-reflection route remains a dependency theorem about finite rows only. □

Theorem 3834.5 (CauchyCompletionMonadLaw L10 forward consumer)**.** *Any* $\mathsf{BoundedCauchyInterval}^{\uparrow}$*,* $\mathsf{RegularCauchyKleisliExtension}^{\uparrow}$*, or* $\mathsf{CauchyCompletionIntersection}^{\uparrow}$ *consumer of the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must read the functorial completion, unit, bind, idempotence,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows before using unit or associativity law data. The downstream chapters*

chapter 6065, chapter 7362, and chapter 7527 therefore consume $L = (M, U, I, B, S, D, R, H, C, P, N)$ *as a finite L10 completion-law surface.*

This theorem is a consumer-facing dependency route. It does not import quotient completions, selected representatives, selected limits, host equality of completed objects, countable choice, hidden stream witnesses, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the monad transformer, unit, idempotence, bind ledger, finite windows, dyadic observations, regular-rational readback, transport, replay, provenance, and local name.

Apply Definition 3834.2. It supplies the carrier admission, unit exactness, idempotence exactness, associativity exactness, classifier transport, and nonescape clauses for the law surface.

Apply Definition 3834.3 and Definition 3834.4. Together they force every law-facing consumer through the completion reflection siblings and through the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readback rows before unit or associativity data are consumed.

The bounded interval, Kleisli extension, and completion-intersection chapters already read this chapter as a downstream completion-law dependency. Since the local carrier contains no quotient, selector, choice, hidden-stream, or ambient-completeness coordinate, those consumers receive only the finite L10 route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3834.1, Definition 3834.2, Definition 3834.3, and Definition 3834.4 seed the finite completion-monad law carrier over monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, local naming, and completion-reflection rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, quotient completion equality, selected representatives, selected limits, host equality of completed objects, countable choice, hidden stream witnesses, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, left-unit exactness, right-unit and idempotence exactness, associativity exactness, classifier transport, and non-escape for BEDC.Derived.CauchyCompletionMonadLawUp.

Constructive story: A $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet is generated from a CauchyCompletionMonad row, CauchyCompletionUnit row, CompletionIdempotence row, CauchyCompletionBind ledger, StreamName windows, DyadicRatCore observations, RegSeqRat readback, componentwise transport, displayed replay, Pkg provenance, and a local NameCert row.

Theorem 3834.6 (CauchyCompletionMonadLaw functor dependency route)**.** *Every accepted* CauchyCompletionMo *packet consumes the functorial completion surface of chapter 423 before its multiplication, unit, idempotence, or associativity rows can be read as law data. Concretely, the monad-law carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *first reads the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local NameCert rows, and only then replays* $M, U, I, B, S, D, R, H, C, P, N$ *as the law surface. Thus the law packet records coherence over the displayed functorial completion route rather than a kernel-only island or a host-level completion equality.*

Proof. Project the carrier of Definition 3834.1. Its monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming rows are law-facing rows, but they do not by themselves identify the completion transformer as a functorial public route.

Read chapter 423 together with the sibling universal-property certificate of chapter 397. The functor packet supplies the source-metric row, regular-Cauchy family row, completion seal, unit-and-bind schedule, universal-property row, classifier ledger, transport row, provenance row, and

local NameCert row consumed by completion-facing clients; the unit-and-bind law rows consume the universal-property witness before associativity is read as monad law data.

Now apply Definition 3834.2 and Definition 3834.3. They replay the unit, idempotence, bind, and associativity rows through the displayed stream, dyadic, regular-rational, transport, and provenance data. Since the functor route is the public completion surface being coordinated, the monad-law packet depends substantively on $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ before the law rows are read. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet also reads the completion universal-property and metric-completion siblings before its law rows are exposed to completion consumers. The extension-uniqueness side is supplied by chapter 397, while the metric-completion handoff is supplied by chapter 1506; hence the unit, multiplication, idempotence, and associativity rows remain attached to displayed completion data rather than to an isolated law ledger. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads the completion functor, universal-property, metric-completion, and Real-completion sibling rows before the monad multiplication and unit laws are available as law data. The completion functor route of chapter 423 supplies the displayed functorial transformer, the universal-property route of chapter 397 supplies the extension and uniqueness rows, the metric-completion packet of chapter 1506 supplies the completion carrier, and the Real-completion handoff of chapter 360 supplies the terminal completion-facing seal. Only after those rows are visible may the law packet expose multiplication, unit, idempotence, and associativity through its local carrier. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads the completion functor route of chapter 423, the completion universal-property surface of chapter 397, and the metric-completion dependency lattice of chapter 1506 before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer treats the monad boundary as available law data. Thus unit, multiplication, idempotence, and associativity rows are exposed only after the displayed completion-facing dependencies are visible, not as isolated quotient-completion equality, selected limits, or ambient real completeness. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ rows read the completion functor sibling of chapter 423 and the Cauchy-completion universal-property sibling of chapter 2319 before their unit, multiplication, and associativity coordinates are accepted as monad-law data. This dependency route is a displayed finite lattice over completion-facing NameCert rows; it does not add quotient completion equality, selected limits, host equality of completed objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet exports its multiplication-law rows toward chapter 1829 as a concrete completion-monad sibling route. The multiplication consumer reads the displayed unit, bind, idempotence, associativity, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming rows of the law packet before it flattens a completion of completions; the edge is finite NameCert data, not quotient-completion equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 3834.7 (CauchyCompletionMonadLaw Bishop completion forward route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *read that supplies the completion monad multiplication and unit route must expose the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *carrier of chapter 1639 before the monad law is consumed. The required forward route displays the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows as the Bishop completion boundary for the monad multiplication and unit.*

Proof. Read the monad-law carrier through the existing completion route. Its multiplication and unit rows are completion-facing, so they cannot be consumed as a free algebraic law before the completion boundary is displayed.

Apply the Bishop regular Cauchy completion packet of chapter 1639. That packet exposes the dyadic tolerance ledger, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows that a completion-facing consumer must inspect.

The monad law then receives its forward route through those visible rows. Because the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows remain on the path, the read is a finite completion route rather than quotient completion, selected limit, countable choice, host real equality, or detached ambient completeness. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads chapter 397 and chapter 423 before associativity law rows are public. This is a finite completion-law dependency through the universal-property and functor rows, not quotient completion equality, selected limits, or host equality.

Theorem 3834.8 (CauchyCompletionMonadLaw functor forward anchor)**.** *Every public monad-law read from an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet factors through the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *dependency route before unit, bind, idempotence, or associativity rows are exposed. The consumer must first read the functor dependency route of Definition 3834.6 and the local NameCert construction of Definition 3834.2; only then may it inspect* $M, U, I, B, S, D, R, H, C, P, N$ *as law rows. The route is a forward anchor into the public functorial completion surface, not a quotient-completion equality, selected representative, selected limit, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Read the functor dependency route first. The theorem Definition 3834.6 places the CauchyCompletionFunctor source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local naming rows before any monad-law consumer reads law data.

Then read the local construction through Definition 3834.2. It exposes $M, U, I, B, S, D, R, H, C, P, N$ as the finite monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming surface.

With both reads in place, the public law route may expose unit, bind, idempotence, and associativity as rows attached to the functorial completion dependency. Since the carrier has no quotient-completion, selected-representative, selected-limit, countable-choice, or ambient-completeness coordinate, the anchor is only the displayed forward dependency. □

Lemma 3834.9 (CauchyCompletionMonadLaw functorial law lattice route)**.** *Every accepted* CauchyCompletionMo *law read first consumes the functorial completion route of chapter 423 and the universal-property route of chapter 397. Only after those rows are displayed may the unit, bind, idempotence, and associativity rows of* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *be read as monad-law data. The packet therefore records a law lattice over the public functorial completion dependency, not a host-level completion equality or a quotient-completion shortcut.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ are the monad-law surface, while the functorial completion packet supplies the public source-metric, regular-Cauchy, completion-seal, unit-and-bind, universal-property, classifier, transport, provenance, and local naming route.

Apply Definition 3834.6. It forces the completion functor and universal-property rows to be read before the law rows are consumed by a completion-facing client.

Finally apply Definition 3834.2 and Definition 3834.3. These replay unit, bind, idempotence, and associativity through the same finite stream, dyadic, regular-rational, transport, replay, provenance, and local naming rows, so the law lattice remains attached to the displayed functorial dependency. □

Lemma 3834.10 (CauchyCompletionMonadLaw completion dependency route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law read consumes the completion-route carrier of chapter 1556, the criterion surface of chapter 1476, and the finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *route before its unit and multiplication laws can be read as monad-law data. The packet therefore binds the unit, idempotence, bind, and associativity rows to the displayed constructive completion route rather than to quotient completion equality, selected limits, countable choice, trichotomy, host equality, or an ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 3834.1. Its law rows $M, U, I, B, S, D, R, H, C, P, N$ are completion-facing rows, so the source completion route must be visible before a client can read them as unit or multiplication data.

Read the constructive metric completion carrier and the Cauchy-completion criterion surface through the cited sibling chapters. They supply the completion admissibility and criterion rows consumed by the monad transformer before the law schedule is replayed.

Finally apply Definition 3834.2 and Definition 3834.3. These force the law read through the finite StreamName, RegSeqRat, and DyadicRatCore rows, leaving no coordinate for quotient completion equality, selected limits, countable choice, trichotomy, host equality, or ambient real completeness. □

Lemma 3834.11 (CauchyCompletionMonadLaw idempotent metric-monad route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *consumer that reads the unit, bind, multiplication, or associativity rows must first expose the completion idempotent route of* **??** *and the metric completion monad route of chapter 2156. The idempotent sibling supplies the completion-after-completion stabilization row, while the metric completion monad sibling supplies the unit and bind law surface; the combined route exports no quotient-completion equality, selected representative, countable choice, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the displayed law carrier from Definition 3834.1. Its unit, bind, multiplication, and associativity rows are read as completion-law data only after the carrier, stream, dyadic, regular rational, transport, replay, provenance, and local naming rows have been displayed.

Now read the completion idempotent sibling. The chapter **??** supplies the stabilization surface for applying completion after completion, so a monad-law consumer cannot treat the multiplication row as an unrecorded host equality.

Finally read the metric completion monad sibling. The chapter chapter 2156 supplies the unit and bind law surface consumed by the same route, and Definition 3834.2 keeps those rows inside the finite packet. Hence the sibling dependency lattice is substantive and finite, with none of the excluded quotient, selection, choice, host-equality, or ambient-completeness endpoints. □

Theorem 3834.12 (CauchyCompletionMonadLaw Kleisli consumer route)**.** *Every* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must first read the monad row, unit row, idempotence row, bind row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before exporting a Kleisli-facing* $\mathsf{Real}^{\uparrow}$ *seal. The route is a consumer of chapter 1845; it does not export quotient completion equality, selected representatives, countable-choice data, host equality of completed names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the local NameCert construction, Definition 3834.2. It exposes $M, U, I, B, S, D, R, H, C, P, N$ as the finite monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming surface.

Now use the unit-associativity boundary, Definition 3834.3. It forces every law-facing $\mathsf{Real}^{\uparrow}$ read through the displayed stream, dyadic, and regular-rational rows before any terminal seal can be inspected.

Finally attach the Kleisli consumer chapter chapter 1845. A $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ consumer may read the monad-law packet only after the same finite rows have been displayed, so its Kleisli-facing seal remains a finite BEDC consumer route rather than a quotient, selection, choice, host equality, or ambient-completeness claim. □

$\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ consumes the completion functor unit and multiplication route of chapter 423 together with the finite readback rows of the Cauchy carrier in chapter 3413. The dependency is only the displayed unit, multiplication, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming surface; it is not a quotient completion or a host universal property.

Theorem 3834.13 (CauchyCompletionMonadLaw completion-adjunction route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *unit or multiplication read consumes the same completion-adjunction route used by chapter 3836. The unit row, bind row, idempotence row, stream window, dyadic tolerance row, regular-rational readback, transport, replay, provenance, and local naming rows of* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *therefore sit over the displayed completion-adjunction surface rather than over an isolated kernel-only packet.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the law surface, but a completion-facing consumer still needs the adjunction route that identifies how completion reads are admitted and returned.

Read chapter 3836. Its source, dense embedding, completion seal, extension handoff, uniqueness, transport, replay, provenance, and local naming rows are the completion-adjunction surface through which the monad-law unit and multiplication rows are consumed.

Now apply Definition 3834.2 and Definition 3834.3. They force unit, idempotence, bind, and associativity reads through the same finite stream, dyadic, regular-rational, transport, replay, provenance, and naming rows, so the monad-law packet is routed through the completion-adjunction sibling before it can serve a Real-facing law read. □

Theorem 3834.14 (CauchyCompletionMonadLaw dense embedding dependency)**.** *Every unit, multiplication, idempotence, bind, and associativity read for an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet must pass through the sibling dense embedding and completion route supplied by chapter 1651 and chapter 1512 before the law rows can be consumed as completion-facing data. The rows* $M, U, I, B, S, D, R, H, C, P, N$ *are therefore read as dense-image, completion-entry, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming data, not as a local hub-only law island.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose the monad-law surface, but the completion-facing consumer still needs the dense image and completion embedding routes that make those law rows readable by a completion packet.

Apply the monad-law obligations and the unit-associativity boundary, Definition 3834.2 and Definition 3834.3. They force the unit, multiplication, idempotence, bind, and associativity rows to remain attached to the displayed stream, dyadic, regular-rational, transport, replay, provenance, and naming rows.

Finally route those rows through the dense uniform embedding and completion embedding siblings. The dense image witness supplies the source-to-completion coverage, and the completion

embedding handoff supplies the consumer-facing entry, so the law rows are consumed only after the sibling route has been read. □

Theorem 3834.15 (CauchyCompletionMonadLaw completion-reflection consumer link)**.** *Every* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *unit, multiplication, idempotence, or associativity read factors through the completion-adjunction surface of chapter 3836, the completion-functor route of chapter 423, and the completion universal-property sibling of chapter 397 before it is consumed as* $\mathsf{Real}^{\uparrow}$*-facing law data. The route is finite completion reflection through displayed carrier rows, not quotient completion equality, selected representatives, selected limits, host equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 3834.1. The monad-law rows $M, U, I, B, S, D, R, H, C, P, N$ expose unit, multiplication, idempotence, bind, stream, dyadic, regular-readback, transport, replay, provenance, and local naming data.

Read the sibling dependency route of Definition 3834.13. It already places the unit and multiplication rows over the metric completion-adjunction surface rather than over an isolated kernel packet.

Finally pass the same finite rows through the completion functor and universal-property siblings. These siblings supply the completion-reflection consumer route needed before a $\mathsf{Real}^{\uparrow}$-facing law read can occur, while the refused quotient, representative, limit, host-equality, and completeness objects remain absent from the carrier. □

Theorem 3834.16 (CauchyCompletionMonadLaw downstream completion consumer)**.** *Any downstream completion consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the law surface through chapter 423, chapter 397, and the totally bounded completion consumer of chapter 1573 before the rows* $M, U, I, B, S, D, R, H, C, P, N$ *are exported as completion-law data. The route is a displayed consumer edge through finite carrier rows; it is not quotient completion equality, selected representatives, selected limits, open-cover compactness, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. The rows $M, U, I, B, S, D, R, H, C, P, N$ expose monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming data, but none of those rows by itself supplies a downstream completion consumer.

Use the completion-functor route of Definition 3834.13. It places the unit and multiplication reads over the same functorial completion surface that a completion-facing client must inspect before law rows are available.

Next apply Definition 3834.14 and Definition 3834.15. These route the same finite rows through the displayed completion embedding, completion functor, and completion universal-property siblings before any $\mathsf{Real}^{\uparrow}$-facing or completion-facing consumer reads them.

Finally read chapter 1573. The totally bounded completion packet is an existing downstream consumer of completion rows, so it may consume the monad-law surface only after the functorial and universal-property routes have already kept $M, U, I, B, S, D, R, H, C, P, N$ attached to finite carrier data. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet also sits horizontally over the completion-functor sibling of chapter 423. Its unit, multiplication, idempotence, and associativity rows consume the functorial completion route before they are read as monad-law data, so the completion monad/functor law lattice is visible as a sibling dependency rather than a kernel-only float. The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ route consumes the Real and Completion siblings before any unit or associativity row is exported: $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ provide the L10 real-name readback

through chapter 38, while CauchyCompletionFunctor$^{\uparrow}$ and CompletionUniversalProperty$^{\uparrow}$ provide the completion-facing handoff through chapter 397. Thus a monad-law consumer sees the finite real-name route and the completion universal-property route as displayed dependencies, not as hidden quotient completion data.

Theorem 3834.17 (CauchyCompletionMonadLaw real completion unit route)**.** *Every accepted* CauchyCompletionMonadLaw$^{\uparrow}$ *unit, multiplication, or associativity read that is consumed as completion-facing law data first passes through the* RealCompletionUnit$^{\uparrow}$ *route of chapter 7084. The read exposes the* StreamName$^{\uparrow}$ *window row,* DyadicRatCore$^{\uparrow}$ *tolerance row,* RegSeqRat$^{\uparrow}$ *readback row, terminal* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before the law packet* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *is exported.*

Thus the monad-law consumer receives the real-completion unit as a displayed finite route. It does not receive quotient-completion equality, a selected Cauchy representative, countable-choice schedule, host equality of completions, hidden real completeness, or a detached Real$^{\uparrow}$ *endpoint.*

Proof. Project the carrier of Definition 3834.1. The law rows M, U, I, B can be consumed only together with the finite stream, dyadic, regular-rational, transport, replay, provenance, and naming rows S, D, R, H, C, P, N.

Apply the real completion unit surface of chapter 7084. Its carrier and nonescape theorems place every real-completion unit read behind the dyadic source, stream-window, regular readback, terminal real seal, transport, replay, provenance, and local naming rows.

Finally use Definition 3834.2 and Definition 3834.3. They bind unit, multiplication, idempotence, and associativity reads to the same finite carrier before any completion-facing consumer sees law data, so the exported route cannot import quotient completions, selected representatives, countable choice, host equality, hidden completeness, or a detached real endpoint. □

The CauchyCompletionMonadLaw$^{\uparrow}$ packet consumes its sibling completion lattice through displayed finite rows. The unit, multiplication, and composition reads are routed through chapter 3413, chapter 1506, chapter 397, and chapter 301: Cauchy rows name the input streams, MetricCompletion rows expose the completion carrier, the universal-property rows supply the comparison route, and RegSeqRat rows give the regular rational windows used by the readbacks. No step appeals to a host completion monad outside the displayed Cauchy completion and regular-rational certificate rows.

Theorem 3834.18 (Cauchy completion monad law completion dependency lattice)**.** *Every accepted* CauchyCompletionMonadLaw$^{\uparrow}$ *packet consumes its completion-facing law rows through the displayed sibling lattice:* MetricCompletion$^{\uparrow}$ *supplies the completion carrier of chapter 1506,* Completion$^{\uparrow}$ *and the* Real$^{\uparrow}$ *completion seal rows supply the real-name boundary of chapter 38, and* CauchyCompletionFunctor$^{\uparrow}$ *supplies the functorial route before unit, bind, multiplication, or associativity data are exported. The route is a finite dependency projection; it is not a quotient-completion equality, selected representative, selected limit, host equality of completed objects, countable-choice principle, or ambient* Real$^{\uparrow}$ *completeness claim.*

Proof. Project the monad-law carrier and the existing sibling routes. The law rows are already carried through the Cauchy completion monad, unit, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming coordinates before any Real-facing read is available.

Read the metric-completion boundary next. The MetricCompletion$^{\uparrow}$ sibling fixes the completion carrier as a displayed route through chapter 1506, so the monad-law packet cannot obtain a hidden host completion object.

Finally compose with the Real-facing route and the functor dependency route. The $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is reached only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows, and the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ route is consumed before unit, bind, multiplication, or associativity rows are exported. Thus the dependency lattice is horizontal and finite, with none of the refused quotient, selector, choice, equality, or completeness endpoints. □

Lemma 3834.19 (CauchyCompletionMonadLaw regular Cauchy uniform-modulus route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *consumer that compares completion unit and bind windows must first read the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *budget of chapter 4158. The modulus row fixes the common regular-Cauchy tolerance budget before the monad-law consumer may identify two completion routes, and the idempotent metric-monad route of Definition 3834.11 keeps the resulting comparison inside the displayed unit, bind, multiplication, and associativity packet. The route exports no quotient-completion equality, selected representative, countable-choice schedule, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The regular-modulus sibling is not a decorative cross-reference: it supplies the uniform tolerance row that lets the local carrier of Definition 3834.1 compare unit and bind reads over one shared regular-Cauchy budget. Without that row, the monad-law packet would still display law coordinates, but it would not expose the finite modulus through which completion consumers synchronize their Cauchy windows.

Theorem 3834.20 (CauchyCompletionMonadLaw regular-modulus unit route)**.** *Every accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *carrier* $L = (M, U, I, B, S, D, R, H, C, P, N)$ *that exports a unit read to a completion-facing consumer first consumes the* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *sibling of chapter 4158. The sibling supplies the common regular-Cauchy modulus for the stream row* S*, dyadic tolerance row* D*, and regular-rational row* R *before the unit row* U *is replayed through* Cont *and recorded by the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadLaw}^{\uparrow}}$ *row. Hence the unit route is a finite modulus-controlled read of the local carrier, not a quotient-completion equality, selected representative, countable-choice schedule, host equality of completions, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier through Definition 3834.1. Its rows S, D, R are the finite stream, dyadic tolerance, and regular-rational readback data used by unit and bind comparisons.

Apply Definition 3834.19. It places the $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ budget before the monad-law consumer identifies unit and bind windows.

Finally apply Definition 3834.2. The unit row U, replay row C, provenance row P, and local naming row N are consumed only after the displayed carrier rows are available, so the route remains inside the finite monad-law packet and cannot import any excluded completion principle. □

Theorem 3834.21 (CauchyCompletionMonadLaw forward completion-category route)**.** *Every* $\mathsf{CauchyCompletionCategory}^{\uparrow}$ *or metric-completion Kleisli consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the monad-law surface through the displayed rows of chapter 378, the functorial completion rows of chapter 423, and the metric-completion admission rows of chapter 1506 before any categorical composition law is used. The visible route is*

$\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, $\mathsf{CompletionIdempotence}^{\uparrow}$, $\mathsf{CauchyCompletionBind}^{\uparrow}$, $\mathsf{StreamNam}$

It exports no quotient completion equality, selected representative, selected limit, host equality of completed objects, countable-choice schedule, hidden stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3834.1. A law-facing consumer receives only the rows $M, U, I, B, S, D, R, H, C, P, N$, so the monad, unit, idempotence, bind, stream, dyadic, regular-rational, transport, replay, provenance, and local naming data are the complete visible surface.

For a completion-category or Kleisli consumer, first read the monad and functorial completion siblings through chapter 378 and chapter 423. The unit row U, idempotence row I, and bind row B supply the law data only as displayed carrier rows.

Next pass through the metric-completion admission of chapter 1506. The stream, dyadic, regular-rational, and Real rows S, D, R keep the consumer on the finite readback route before any categorical composition law is visible.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no coordinate for quotient completion equality, selected representatives, selected limits, host equality, countable choice, hidden stream witnesses, or ambient completeness, the stated route is exhaustive. □

The $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ packet reads the multiplication sibling of chapter 1829 together with the monad sibling of chapter 378 before its associativity and unit-law rows are exposed. The multiplication row supplies the displayed bind-reassociation surface, while the monad row supplies the unit-and-bind schedule consumed by the law carrier; the route therefore remains a finite NameCert dependency through those sibling packets rather than a host-level equality of completed objects.

Theorem 3834.22 (CauchyCompletionMonadLaw finite-approximation consumer route)**.** *Every finite-approximation or totally bounded completion consumer of an accepted* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *packet reads the monad-law surface through the displayed completion rows before using it as approximation data. The route passes through the finite window, dyadic tolerance, regular-rational readback, unit, multiplication, idempotence, bind, transport, replay, provenance, and local naming rows of Definition 3834.1, then through the finite approximation surface of chapter 7574, and finally through the totally bounded completion consumer of chapter 1573. It exports a finite completion-law route for approximation consumers, not quotient completion equality, selected representatives, selected limits, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadLawUp.CauchyCompletionMonadLawFiniteApproximationConsumerRout`

Proof. Project the carrier of Definition 3834.1. The law rows $M, U, I, B, S, D, R, H, C, P, N$ expose unit, multiplication, idempotence, bind, stream-window, dyadic-tolerance, regular-readback, transport, replay, provenance, and local naming data.

Use the local NameCert obligations and the unit-associativity boundary, Definition 3834.2 and Definition 3834.3. They keep every law read attached to the same finite completion-facing rows before a consumer can treat the law as available approximation data.

Now route the read through the approximation and total-boundedness consumers. The finite approximation surface of chapter 7574 supplies the windowed approximation interface, while chapter 1573 supplies the totally bounded completion consumer edge.

Since the carrier and both consumer routes display only finite rows, the resulting read cannot introduce quotient completion equality, selected representatives, selected limits, open-cover compactness, countable choice, or ambient Real completeness. □

Theorem 3834.23 (CauchyCompletionMonadLaw forward consumer lattice)**.** *Finite-approximation and totally bounded completion consumers read the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *monad-law rows*

only through the displayed sibling routes

CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Reg

Thus a downstream approximation read may consume unit, multiplication, bind, idempotence, transport, replay, provenance, and local naming rows from the monad-law packet only after the functor, universal-property, metric-completion, finite stream-window, dyadic tolerance, regular-rational readback, and terminal Real rows have remained visible. The lattice exports no selected completion representative, quotient completion equality, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

The CauchyCompletionMonadLaw$^{\uparrow}$ route supplies a downstream completion consumer edge for finite-net packets: chapter 1573 reads monadic completion composition only after the Cauchy-filter and embedding rows have remained attached to the finite-net ledger, while chapter 1844 keeps the dyadic, stream, regular-rational, and Real-seal faces visible before any completion-facing read is consumed. Thus the monad-law chapter is a sibling lattice route for finite-net completion consumers, not a source of selected limits, quotient representatives, open-cover compactness, or ambient Real completeness.

Chapter 3835

A Concrete Naming Certificate for CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and filter convergence, and Bishop–Bridges on constructive completion, the CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$ packet records the finite filterbase-completion refinement of the complete uniform-space surface. It sits after the uniform entourage data of chapter 339, the saturated Cauchy-filter route of chapter 333, the finite basis generator of chapter 1501, the limit-facing boundary of chapter 1558, the window and readback surfaces of chapter 126 and chapter 301, the terminal seal of chapter 38, and the completion handoffs of chapter 1473 and chapter 1569. The packet names a displayed filterbase route to completion consumers; it is not a quotient filter, selected completion point, countable-choice selector, host equality, ambient completeness theorem, or public standard bridge.

Definition 3835.1 (Complete uniform-space filterbase-completion carrier)**.** *A* CompleteUniformSpaceFilterbaseCom *carrier is a finite* BHist *packet*

$$Q = (U, F, B, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UniformSpace$^{\uparrow}$ *entourage row, displaying the finite entourage filterbase, diagonal, refinement, symmetry, and composition data consumed by the completion request.*
2. *F is the* CauchyFilter$^{\uparrow}$ *row, recording the saturated Cauchy-filter request over the uniform surface of U.*
3. *B is the* CauchyFilterBasis$^{\uparrow}$ *generator row, exposing the finite basis elements and common-refinement data through which F is read.*
4. *L is the* CauchyFilterLimit$^{\uparrow}$ *boundary row, reached only after the basis and saturated filter rows have been replayed.*
5. *S is the* StreamName$^{\uparrow}$ *window row attached to the filterbase convergence request.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row extracted from the same windows.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row read only after U, F, B, L, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the entourage, filter, basis, limit boundary, window, readback, seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from the uniform entourage request through the filterbase generator, Cauchy saturation, limit boundary, finite windows, regular readback, and Real seal.*

10. *P is the* Pkg *provenance over* CompleteUniformSpaceFilterbaseCompletion$^{\uparrow}$, CompleteUniformSpace$^{\uparrow}$, UniformSpace$^{\uparrow}$, CauchyFilter$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, UniformCompletion$^{\uparrow}$, CauchyFilterCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, F, B, L, S, R, E, H, C, P, N$. The carrier has no coordinate for quotient filters, selected completion points, countable-choice schedules, host equality, ambient completeness, or a detached completion object.

Theorem 3835.2 (Filterbase convergence obligations)**.** *For every accepted* CompleteUniformSpaceFilterbaseComple *carrier $Q = (U, F, B, L, S, R, E, H, C, P, N)$, the local* NameCert$_{\text{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed uniform-space entourage data, Cauchy-filter row, CauchyFilterBasis generator, CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, Real seal, transport, replay, provenance, and local naming row;*

2. *filterbase convergence exposure, requiring every completion-facing request to pass through U, B, and F before the limit boundary L is inspected;*

3. *readback exactness, requiring every accepted boundary observation to pass from L through the matching S and R rows before the Real seal E is available;*

4. *completion-handoff stability, requiring* UniformCompletion$^{\uparrow}$ *and* CauchyFilterCompletion$^{\uparrow}$ *consumers named by P to read the same displayed route U, B, F, L, S, R, E;*

5. *ledger nonescape, requiring H, C, P, N to export no quotient filter, selected completion point, countable-choice selector, host equality, ambient completeness theorem, or detached completion object.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3835.1. Its public rows are exactly $U, F, B, L, S, R, E, H, C, P, N$, so carrier admission exposes only the displayed uniform, filterbase, convergence, seal, and structural data.

The filterbase convergence exposure is forced by the row order. A completion-facing request first reads the finite entourage data U, then the basis generator B, and only then the saturated Cauchy-filter row F before the limit boundary L is available.

Readback exactness is downstream of the boundary row. The row L can expose observations only through the StreamName windows S and the RegSeqRat readback R, and the terminal Real seal E is available only after those rows have been replayed.

The completion-handoff and ledger clauses are structural. Uniform-completion and Cauchy-filter-completion consumers are named in P as readers of the same displayed route, while H, C, N transport, replay, and name only that route. Hence the excluded quotient, selector, host-equality, choice, completeness, and detached-completion objects have no coordinate in the packet. □

Lean4 ✓ `BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.CompleteUniformSpaceFilterbaseCompletio`
`single carrier alignment`

Theorem 3835.3 (Cauchy-filter completion handoff)**.** *Let $Q = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ *packet. Every* $\mathsf{UniformCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer that reads Q receives the finite handoff*

$$U \longmapsto B \longmapsto F \longmapsto L \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion handoff is filterbase-first: the consumer cannot inspect a completion-facing boundary before the uniform entourage row, the CauchyFilterBasis generator, and the saturated CauchyFilter row have been displayed.

Proof. Begin with a completion-facing consumer. The provenance row P may mention $\mathsf{UniformCompletion}^{\uparrow}$ and $\mathsf{CauchyFilterCompletion}^{\uparrow}$, but it does so only as a row of the accepted carrier Q.

The source of the handoff is therefore the displayed route. The uniform-space row U supplies the entourage filterbase, B supplies the finite basis generator, and F supplies the saturated Cauchy-filter request over the same data.

The boundary and readback rows are downstream. The consumer reaches L only after U, B, F have been replayed, then reads S and R before the terminal seal E is available.

Finally H, C, P, N transport, replay, source, and name this route. Since the carrier has no separate completion object, selected point, quotient filter, choice schedule, host equality, or ambient completeness row, every accepted completion handoff is confined to the displayed path. □

Theorem 3835.4 (Filterbase Real-seal nonescape)**.** *Let $Q = (U, F, B, L, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ *carrier. Every consumer that reaches the displayed* $\mathsf{Real}^{\uparrow}$ *seal row E must first pass through the uniform entourage row U, the CauchyFilterBasis row B, the saturated CauchyFilter row F, the CauchyFilterLimit boundary row L, the StreamName windows S, and the RegSeqRat readback row R. The rows H, C, P, N then preserve and name the same route, so no accepted read can escape from E into a selected completion point, quotient filter, host equality, countable-choice selector, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or detached* $\mathsf{UniformCompletion}^{\uparrow}$ *object.*

Proof. Project the carrier definition. The terminal row E appears only as the Real seal of the tuple, and the preceding public coordinates are precisely U, F, B, L, S, R.

Apply the obligation surface of Definition 3835.2. It requires the filterbase convergence request to consume U, B, F, then the limit boundary L, then the window and readback rows S, R before E can be read.

The structural rows cannot introduce a second route. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed packet, while the excluded selector, quotient, host-equality, choice, ambient-completeness, and detached-completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}$ packet is generated from UniformSpace entourage data, a CauchyFilter row, a CauchyFilterBasis generator, a CauchyFilterLimit boundary, StreamName windows, RegSeqRat readbacks, a RealUp seal, UniformCompletionUp and CauchyFilterCompletionUp provenance, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 3835.1, Definition 3835.2, Definition 3835.3, and Definition 3835.4 seed the finite filterbase-completion route for complete uniform-space consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompleteUniformSpaceFilterbaseCompletion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.CompleteUniformSpaceFilterbaseCompletionTast`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, countable-choice selectors, host equality, ambient completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, filterbase convergence exposure, readback exactness, completion-handoff stability, and Real-seal nonescape for BEDC.Derived.CompleteUniformSpaceFilterbaseCompletionUp.

Chapter 3836

A Concrete Naming Certificate for $\mathsf{MetricCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ packet records the universal mapping boundary for a displayed metric source and its completion as finite BEDC data. It reads the metric-completion carrier of chapter 1506, the dense embedding route of chapter 3351, the Cauchy adjunction surface of chapter 1456, and the unit-triangle ledger of chapter 2256. The chapter names the metric-specific adjunction packet without quotient completion, selected representatives, countable choice, host equality of completed objects, or ambient completeness.

Definition 3836.1 (MetricCompletionAdjunction carrier)**.** *A* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, M, D, U, K, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, carrying the displayed carrier, distance classifier, finite balls, and the* $\mathsf{Met}^{\uparrow}$ *provenance needed before completion is read.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of chapter 1506, including the chosen Cauchy-filter or Cauchy-net branch, readback row, separated comparison, and uniform-completion handoff.*
3. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row of chapter 3351, displaying the source-to-completion dense image used by the universal mapping read.*
4. *U is the unit row sending each displayed source point into the completion packet through D and M.*
5. *K is the counit row for completion-facing consumers whose target is already read through the same metric completion surface.*
6. *T is the triangle ledger. It records that the unit followed by the counit, and the counit followed by the unit-facing extension, replay the same finite rows rather than invoking host equality of completed carriers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal row used by the readback side of the completion packet.*
8. *E is the* $\mathsf{UniformCompletion}^{\uparrow}$ *and extension-consumer row through which uniformly continuous finite-window maps read the same adjunction packet.*

9. *H records componentwise* hsame *transport for the source metric, completion row, dense embedding, unit, counit, triangle ledger, readback seal, extension row, replay, provenance, and local name.*

10. *C records displayed* Cont *replay from the source metric through dense embedding, unit, completion readback, extension, counit, and triangle comparison.*

11. *P is the* Pkg *provenance over* MetricCompletionAdjunction$^{\uparrow}$, MetricCompletion$^{\uparrow}$, MetricCompletionDenseEmbeddin CauchyCompletionAdjunction$^{\uparrow}$, CompletionUnitTriangle$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, UniformCompletion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$ *row.*

The classifier accepts only packets whose unit, counit, and triangle rows are read through the same displayed source metric, completion packet, dense embedding, regular readback, Real seal, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows.*

Theorem 3836.2 (MetricCompletionAdjunction NameCert obligations)**.** *For every accepted* MetricCompletionAdju *carrier*

$$A = (X, M, D, U, K, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of source metric admission, metric-completion admission, dense embedding exposure, unit visibility, counit visibility, triangle-ledger comparison, RegSeqRat/Real seal readback, uniform-extension handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of A. No obligation may be discharged by a quotient completion, selected Cauchy limit, countable-choice schedule, host equality of completion classes, or ambient complete metric space.*

Proof. Project the carrier of Definition 3836.1. The analytic rows are precisely the source metric X, completion packet M, dense embedding D, unit U, counit K, triangle ledger T, readback seal R, and extension row E.

The dense embedding and completion rows force the order of every accepted read. A consumer first opens the source metric and metric-completion packet, then reads the dense image route before any unit, counit, triangle, Real-seal, or uniform-extension coordinate is available.

Transport and replay are structural. The row H compares only the displayed components by hsame, while C replays the same route through source, dense image, completion, extension, counit, and triangle rows.

Finally P and N source and name the same finite packet. Since none of the listed rows stores a quotient completion, selected limit, countable-choice schedule, host equality, or ambient complete metric space, the displayed obligations exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetricCompletionAdjunctionUp.MetricCompletionAdjunctionTasteGate single carrier alignment`

Theorem 3836.3 (MetricCompletionAdjunction unit-counit triangle)**.** *Every unit-counit triangle read of an accepted* MetricCompletionAdjunction$^{\uparrow}$ *carrier factors through the finite route*

$$X \longmapsto D \longmapsto U \longmapsto M \longmapsto E \longmapsto K \longmapsto T \longmapsto R, H, C, P, N.$$

The route identifies the two triangle faces only by the displayed hsame *and* Cont *rows carried in H and C, and it consumes the RegSeqRat/Real seal row R only after the metric completion, dense embedding, unit, counit, and uniform-extension rows have been displayed.*

Proof. Start from the carrier inventory. The source metric row X and dense embedding row D are the only rows that introduce source points into the completion packet, so the unit face must pass through them before it can reach M.

The extension and counit faces are then downstream of the same completion packet. The row E records the uniform-extension consumer route, K records the counit-facing return, and T records the triangle comparison over those already displayed rows.

The terminal seal and structural rows cannot change the route. The row R is the RegSeqRat/Real readback attached to the completion packet, while H, C, P, N only transport, replay, source, and name the same finite data. Therefore the triangle read is exhausted by the displayed route and imports no quotient, selector, choice, host equality, or ambient-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ packet is generated from a source metric row, metric-completion row, dense embedding row, unit row, counit row, triangle ledger, RegSeqRat/Real seal readback, uniform-extension handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3836.1, Definition 3836.2, and Definition 3836.3 seed the metric-specific completion adjunction boundary over dense embedding, unit, counit, triangle, readback, and uniform-extension rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjunction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricCompletionAdjunctionUp.MetricCompletionAdjunctionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host equality of completed objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense embedding exposure, unit and counit visibility, triangle comparison, RegSeqRat/Real seal handoff, uniform-extension handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionAdjunctionUp.

The $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ hub routes first through chapter 486, chapter 782, and chapter 444 before the NameCertMorphism dependency child is read.

Chapter 3837

A Concrete Naming Certificate for $\mathsf{GeneratorClosureRobustness}^{\uparrow}$

The $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ packet records a finite hypercube and median-closure certificate for the zero-marker generator-closure robustness surface of Definition 9462.1779. It reads active median generators, median-essential codons, deletion-loss rows, Q_2 completion closure, and minimal percolating seeds as finite combinatorial data over Q_6. The finite table read is routed through the finite observation budget selector of chapter 782 before the active-generator, deletion-loss, and Q_2 completion rows are consumed. The collision-window consumer of chapter 444 reads the same finite moment-index and Hankel-shadow style rows, so the robustness packet's trigger and Q_2-completion ledgers feed a sibling finite-kernel consumer. The chapter is a NameCert target for finite closure robustness only; it does not claim biological necessity, probabilistic selection, host-level optimality, or an unbounded search principle.

The active-generator and Q_2-completion rows refine the generator-closure carrier of chapter 486. The robustness packet adds deletion-loss, minimal-seed, and trigger-comparison ledgers before any robustness read is exported, so every robustness consumer first sees the same finite generator-closure boundary and then the additional robustness tables.

The same dependency also controls replay: the robustness packet consumes the generator-closure carrier and replay rows of chapter 486 before it can state robustness under perturbation or audit transport.

Thus the active-generator and Q_2 completion rows consume three sibling surfaces before any trigger read is available: chapter 486 supplies the finite generator boundary, chapter 782 supplies the finite table-inspection budget, and chapter 444 supplies the downstream finite-kernel consumer that reads the same collision-facing row family.

Definition 3837.1 (GeneratorClosureRobustness carrier)**.** *A* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, E, L, Q, P, T, H, C, K, N)$$

with the following displayed rows:

1. *A is the active median-generator row, listing the two size-7 active generators and their common rigid skeleton.*
2. *E is the median-essential row, listing the five codons whose deletion changes the median closure.*
3. *L is the deletion-loss row, recording the finite lost-codon table for each median-essential deletion.*

4. *Q is the Q_2 completion row, recording the rule that a three-of-four occupied Q_2 face adjoins the fourth vertex.*
5. *P is the percolating-seed row, recording the 484 size-7 Q_2-completion seeds and the absence of smaller percolating seeds.*
6. *T is the trigger-closure row comparing rigid median generation with broad Q_2 percolation capacity.*
7. *H is the componentwise* hsame *transport row for active-generator, deletion-loss, completion, seed, and trigger rows.*
8. *C is the displayed* Cont *replay row forcing every robustness read through A, E, L, Q, P, T, H.*
9. *K is the* Pkg *provenance over* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$*, finite codon-window rows, Q_6, Q_2 faces,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by this finite table data and its displayed transports.

Theorem 3837.2 (GeneratorClosureRobustness NameCert obligations)**.** *For every accepted* GeneratorClosureRobu *carrier $G = (A, E, L, Q, P, T, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$ *surface consists exactly of these rows:*

1. *carrier admission for the active median generators, median-essential list, deletion-loss table, Q_2 completion rule, percolating-seed table, trigger comparison, transport, replay, provenance, and local name;*
2. *active-generator exactness, requiring the two displayed size-7 generators and their common six-codon skeleton to be read before a rigidity claim is accepted;*
3. *deletion-loss exactness, requiring every median-essential deletion to expose its finite lost-codon row;*
4. *Q_2 completion exactness, requiring every completion step to be witnessed by a displayed axis-aligned Q_2 face with at least three current vertices;*
5. *seed minimality, requiring the no-size-6 row and the 484 size-7 seed row before a percolation claim is accepted;*
6. *non-escape of biological necessity, probabilistic selection, unbounded search, host optimality, or hidden codon classifiers.*

No closure-robustness obligation is stored outside this finite packet.

Proof. Project the carrier of Definition 3837.1. The only rows available to the certificate are $A, E, L, Q, P, T, H, C, K, N$.

The active-generator and deletion-loss obligations read A, E, L as finite tables. The median-generator claim is accepted only after the two size-7 generators and their common skeleton are displayed, and each essential deletion is accepted only with its listed lost-codon row.

The Q_2 completion and seed obligations read Q and P. Each completion step is justified by a finite face row, and the percolation claim is accepted only after the no-smaller-seed row and the 484 minimal-seed row are present.

The trigger and non-escape clauses follow through T, H, C, K, N. These rows package the finite comparison between rigid median generation and broad Q_2 percolation; they contain no

biological-necessity coordinate, probability model, unbounded search oracle, host optimum, or hidden classifier. □

Theorem 3837.3 (GeneratorClosureRobustness Q2 completion exactness)**.** *Every accepted Q_2 completion read in a* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *packet must expose the displayed axis-aligned Q_2 face row Q, the active-generator row A, the deletion-loss row L, and the percolating-seed row P before a trigger-closure conclusion is read.*

Proof. Project the carrier of Definition 3837.1. The completion coordinate is the row Q, while the rows A, L, and P record the finite active-generator, deletion-loss, and percolating-seed data that determine whether the completion is part of the accepted robustness packet.

Apply Definition 3837.2. Its Q_2 completion exactness clause requires every completion step to be witnessed by a displayed axis-aligned Q_2 face with at least three current vertices, and its active-generator, deletion-loss, and seed clauses require the rows A, L, P before any trigger-facing read is accepted.

The rows T, H, C, K, N can only compare, transport, replay, source, and name those finite rows. Therefore the completion read exports no biological necessity, probability model, host optimum, hidden classifier, or unbounded search principle. □

Theorem 3837.4 (GeneratorClosureRobustness trigger-closure boundary)**.** *Every trigger-closure read of an accepted* $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ *packet factors through the displayed active-generator row, median-essential row, deletion-loss table, Q_2 completion rule, percolating-seed table, trigger comparison, componentwise transport, replay, provenance, and local NameCert row. A missing active-generator row, deletion-loss row, Q_2 face row, or minimal-seed row is a failure to land on the finite robustness surface, not a robustness counterexample.*

Proof. Begin with an accepted trigger-closure read. Carrier admission exposes the rows $A, E, L, Q, P, T, H, C, K, N$ before the trigger comparison can be consumed.

If the read concerns median rigidity, it must inspect A, E, L: the active generators, essential codons, and deletion losses are the displayed finite evidence for the rigid skeleton.

If the read concerns completion percolation, it must inspect Q and P: the Q_2 completion rule supplies the step witnesses, and the seed table supplies both the minimality lower bound and the 484 size-7 witnesses.

The replay row C and transport row H keep these two closure surfaces in one packet. Therefore a route missing one of the listed finite rows has not reached the accepted trigger-closure boundary, while a route that reaches the boundary exports no hidden classifier, biological-necessity assertion, probabilistic model, host optimum, or unbounded search principle. □

Falsifiable prediction: An accepted $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ read is refuted if a trigger-closure consumer reaches a rigidity or percolation conclusion without first displaying the active-generator row, deletion-loss table, Q_2 completion row, minimal-seed row, transport row, replay row, provenance row, and local naming row.

Independence witness: active median generation, median-essential deletion, Q_2 completion, and minimal percolating seeds: none is bijective with $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ because the packet jointly carries the rigid median skeleton, deletion losses, percolation rule, seed count, trigger comparison, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GeneratorClosureRobustness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3837.1, Definition 3837.2, and Definition 3837.4 seed the finite generator-closure robustness carrier over active-generator, essential-codon, deletion-loss, Q_2 completion, seed, trigger, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{GeneratorClosureRobustness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, standard bridge checking, biological necessity, probabilistic selection, unbounded search, host optimality, or hidden codon classifiers.

Upgrade path: Obligation closure requires checked carrier admission, active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, trigger-closure boundary, and non-escape for BEDC.Derived.GeneratorClosureRobustnessUp.

Constructive story: A GeneratorClosureRobustness$^{\uparrow}$ packet is generated from active median generators, median-essential deletion rows, deletion-loss tables, Q2 completion faces, minimal percolating seed rows, trigger comparison, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theorem 3837.5 (GeneratorClosureRobustness forward-kernel consumer)**.** *Let $G = (A, E, L, Q, P, T, H, C, K, N)$ be an accepted* GeneratorClosureRobustness$^{\uparrow}$ *carrier. Any forward socket or kernel-audit consumer citing the robustness packet must read the active-generator row A, deletion-loss rows L, Q_2 completion row Q, minimal-seed row P, trigger row T, componentwise transport row H, replay row C, provenance row K, and local naming row N before invoking the* ForwardSocketLedger$^{\uparrow}$ *route of chapter 3387. The forward-kernel route is a displayed finite dependency route; it does not export biological necessity, probabilistic selection, host optimality, hidden codon classifiers, or an unbounded search principle.*

Proof. Project the carrier of Definition 3837.1. The rows $A, E, L, Q, P, T, H, C, K, N$ are the only coordinates a robustness consumer can inspect.

Use the NameCert obligations of Definition 3837.2. They require active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, and the local non-escape clauses before any public robustness read is accepted.

Apply Definition 3837.3 and Definition 3837.4. These force the completion and trigger reads to pass through $A, L, Q, P, T, H, C, K, N$ as one transported packet.

Only after those rows are visible may the consumer cite chapter 3387. The ForwardSocketLedger$^{\uparrow}$ chapter records the downstream socket ledger, while the robustness rows remain the finite kernel evidence that the consumer is auditing. □

Theorem 3837.6 (GeneratorClosureRobustness NameCertMorphism dependency route)**.** *Let $G = (A, E, L, Q, P, T, H, C, K, N)$ be an accepted* GeneratorClosureRobustness$^{\uparrow}$ *carrier. Every morphism-facing consumer of this packet must preserve the active-generator row A, the deletion-loss rows L, the Q_2 completion row Q, the minimal-seed row P, the trigger row T, the componentwise transport row H, the replay row C, the provenance row K, and the local naming row N before the read is transported through chapter 481. The route is a horizontal NameCert dependency route; it exports no biological necessity, probabilistic selection principle, host optimum, hidden codon classifier, unbounded search oracle, or morphism detached from the displayed robustness packet.*

Proof. Project the carrier of Definition 3837.1. The finite rows $A, E, L, Q, P, T, H, C, K, N$ are the only data available to a morphism-facing robustness consumer.

Apply the NameCert obligations of Definition 3837.2. They force carrier admission, active-generator exactness, deletion-loss exactness, Q_2 completion exactness, seed minimality, trigger comparison, transport, replay, provenance, and local naming to remain visible before any robustness read is accepted.

Use Definition 3837.3 and Definition 3837.4. These keep the completion and trigger rows inside the same transported packet, so a consumer cannot replace the robustness evidence by a hidden classifier or an unbounded search principle.

Only after those rows are preserved may the consumer cite chapter 481. The NameCertMorphism route transports the displayed naming dependency, while $\mathsf{GeneratorClosureRobustness}^{\uparrow}$ still supplies the finite active-generator, deletion-loss, completion, seed, trigger, transport, replay, provenance, and local naming rows being transported. □

Chapter 3838

A Concrete Naming Certificate for $\mathsf{ComputableUniformContinuity}^{\uparrow}$

Following Bishop–Bridges and Weihrauch computable analysis, the $\mathsf{ComputableUniformContinuity}^{\uparrow}$ packet records the finite BEDC surface by which a represented-space map exposes one computable modulus route before any real-valued completion consumer reads it as an ambient host function. It sits over the CauchyRepresentation decoding of chapter 1680, the source and target represented-space rows of chapter 1494, the finite name-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, the modulus row of chapter 369, and the continuous-map route of chapter 130. The packet names a computable-analysis uniform-continuity boundary; it is not a host function, not a choice-selected modulus, not a quotient representation, not full $\mathsf{Real}^{\uparrow}$ completeness, and not a checked public bridge.

Definition 3838.1 (Computable uniform continuity carrier)**.** *A* $\mathsf{ComputableUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, K, R, M, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the represented-space source row, exposing only the source carrier, admissible names, and finite observation cells used by the packet.*
2. *Y is the represented-space target row, exposing the matching target carrier, target names, and target observation cells.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoding row, relating source and target names to regular Cauchy information before any real-facing seal is inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the decoded Cauchy data.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *realizer row, giving the finite rule that turns a requested target tolerance into a source-name window.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger for the source and target name segments inspected by M and K.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic budgets used by the source window, target window, and regular rational readback.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, Y, K, R, M, W, T have been displayed.*
9. *H records componentwise* hsame *transport for represented-space rows, decoding, readback, modulus, windows, tolerance ledger, seal, replay, provenance, and naming.*
10. *C records displayed* Cont *replay from a represented-space consumer through the decoding, modulus, finite-window, tolerance, readback, and seal rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ComputableUniformContinuity}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K, R, M, W, T, E, H, C, P, N$. The carrier has no coordinate for an external host function, a selected modulus outside M, quotient equality of representations, full real completeness, countable choice over names, or a detached continuity proof.

Lean4 ✓ `BEDC.Derived.ComputableUniformContinuityUp.ComputableUniformContinuityTasteGate single carrier alignment`

Theorem 3838.2 (Computable uniform continuity NameCert obligations)**.** *For every accepted* $\mathsf{ComputableUniformContinuity}^{\uparrow}$ *carrier $Q = (X, Y, K, R, M, W, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ComputableUniformC}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $X, Y, K, R, M, W, T, E, H, C, P, N$;*
2. *modulus visibility: every exported continuity read passes through the single* $\mathsf{UniformModulus}^{\uparrow}$ *row M before a target tolerance is accepted;*
3. *represented-space compatibility: the source row X, target row Y, and decoding row K are read together, so the modulus never acts on a detached host carrier;*
4. *window transport: the* $\mathsf{StreamName}^{\uparrow}$ *row W and* hsame *row H carry the same finite source and target name windows through every accepted replay;*
5. *tolerance stability: the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E use the same dyadic budgets exposed by M and W;*
6. *ledger nonescape: H, C, P, N export no host function, choice-selected modulus, quotient representation, full* $\mathsf{Real}^{\uparrow}$ *completeness theorem, countable choice over names, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier in Definition 3838.1. Its accepted rows are exactly $X, Y, K, R, M, W, T, E, H, C, P, N$, so carrier admission exposes represented-space data, decoding, readback, modulus, windows, tolerance ledger, real seal, and structural rows only.

The modulus visibility and represented-space compatibility clauses are forced by the row order. A continuity consumer must first read the source and target represented-space rows X, Y, then the decoding row K, and then the displayed modulus realizer M before it can request a target tolerance.

The window and tolerance clauses read the finite ledgers. The row W lists the inspected source and target name windows, while T records the dyadic budgets that make the $\mathsf{RegSeqRat}^{\uparrow}$ readback R and terminal $\mathsf{Real}^{\uparrow}$ seal E available.

Finally inspect the structural rows. Componentwise transport H, replay C, provenance P, and local naming N preserve the same displayed route and contain no coordinate for the excluded host function, selected modulus, quotient representation, full real completeness theorem, countable choice, or bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{ComputableUniformContinuity}^{\uparrow}$ read is refuted if it exports a uniform-continuity consumer without displaying represented-space source and target rows, CauchyRepresentation decoding, RegSeqRat readback, UniformModulus realizer, StreamName windows, DyadicRatCore tolerance ledger, RealUp seal, transport, replay, provenance, and local naming.

Independence witness: cauchy-representation, representedspace, streamname, regseqrat, dyadicratcore, real, uniformmodulus, continuousmap: none alone is bijective with $\mathsf{ComputableUniformContinuity}^{\uparrow}$ because the packet jointly records represented source and target rows, Cauchy decoding, a computable modulus realizer, finite name windows, rational readback, dyadic tolerance stability, terminal Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableUniformContinuity}^{\uparrow}$ packet is generated from represented-space source and target rows, CauchyRepresentation decoding, RegSeqRat readback, a UniformModulus realizer, StreamName finite windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3838.1 and Definition 3838.2 seed the finite computable-analysis route from represented-space names to a uniform-modulus Real consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host functions, choice-selected moduli, quotient representations, full RealUp completeness, countable choice over names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus visibility, represented-space compatibility, window transport, tolerance stability, ledger nonescape, and local naming for BEDC.Derived.ComputableUniformContinuityUp.

Chapter 3839

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy sequences, the $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ packet records the quotient-free separation certificate used when two completion names are compared before a terminal real seal is consumed. It sits on the finite observation route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the extension-and-uniqueness surface of chapter 2319. The packet names separation between displayed completion names; it is not quotient completion equality, selected representative choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3839.1 (Regular Cauchy completion separation carrier)**.** *A* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, D, W, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two completion-name source rows, each opened only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational route.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the two names at a displayed precision.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window exposing the observations of A and B requested by D.*
4. *R is the regular-rational comparison row recording the inspected rational entries, common-tail data, and separation witness before any Real-facing read.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, D, W, R have been displayed.*
6. *U is the* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *uniqueness row, read only as a finite extension-and-uniqueness boundary for the same displayed completion names.*
7. *H records componentwise* hsame *transport for $A, B, D, W, R, E, U, C, P, N$.*
8. *C records displayed* Cont *replay from a separation request through the two source rows, dyadic tolerance, stream window, regular rational comparison, terminal seal, and universal-property uniqueness row.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient representatives, host sequence equality, selected limits, countable choice over tails, trichotomy, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionSeparationUp.RegularCauchyCompletionSeparationTasteGate`
`single carrier alignment`

Theorem 3839.2 (Regular Cauchy completion separation NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *carrier* $S = (A, B, D, W, R, E, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$ *surface consists exactly of two-name source admission, shared dyadic tolerance exposure, finite StreamName window routing, regular rational comparison, terminal Real seal handoff, universal-property uniqueness routing, componentwise transport, continuation replay, package provenance, and local naming through the displayed rows* $A, B, D, W, R, E, U, H, C, P, N$.

Proof. Project the carrier of Definition 3839.1. The two completion names are visible only through A and B, and the comparison has a single displayed dyadic tolerance ledger D.

The finite window row W exposes the observations requested by D. The row R records the inspected regular-rational entries, common-tail data, and separation witness before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be read.

The universal-property row U supplies only the displayed uniqueness boundary already named by chapter 2319. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, and the carrier lacks any quotient representative, host equality, selected limit, countable-choice, trichotomy, or ambient completeness coordinate. □

Theorem 3839.3 (Regular Cauchy completion separation handoff)**.** *Every completion-facing consumer that compares two accepted* $\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}$ *names must read the finite route*

$$A, B, D, W \longmapsto R \longmapsto E, U \longmapsto H, C, P, N$$

before treating the names as separated or identified by the completion boundary. The handoff routes through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ *as displayed NameCert data, and it exports no quotient equality, selected representative, host real equality, countable-choice selector, trichotomy split, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier $S = (A, B, D, W, R, E, U, H, C, P, N)$. A comparison request first opens the two regular Cauchy source rows and the shared dyadic tolerance ledger.

The finite StreamName window W determines which observations of the two names are inspected. The regular-rational comparison row R then records the common-tail and separation data, so the request remains inside finite $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ evidence.

Only after that comparison can the terminal Real seal E and universal-property uniqueness row U be consumed. The structural rows H, C, P, N transport, replay, source, and name the same route, leaving no carrier coordinate for quotient equality, selected representatives, host equality, countable choice, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionSeparation}^{\uparrow}}$.

Constructive story: A RegularCauchyCompletionSeparation$^{\uparrow}$ packet is generated from two regular Cauchy completion-name rows, a shared DyadicRatCore tolerance ledger, finite StreamName windows, a RegSeqRat comparison row, a terminal RealUp seal, a CauchyCompletionUniversalProperty uniqueness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionSeparation}^{\uparrow}}$, seed)

Scope closed: Definition 3839.1, Definition 3839.2, and Definition 3839.3 seed the quotient-free separation surface for displayed regular Cauchy completion names.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionSeparation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, host sequence equality, countable choice over tails, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-tolerance exactness, finite-window routing, regular-rational comparison, terminal seal handoff, universal-property uniqueness routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionSeparationUp.

Chapter 3840

A Concrete Naming Certificate for WRRHotspotCube$^{\uparrow}$

The WRRHotspotCube$^{\uparrow}$ packet realizes the purine-suffix hotspot cube of Definition 9462.1780, the reassignment count of Definition 9462.1781, and the highest-scoring cube certificate of Definition 9462.1783 as finite BEDC NameCert data. It consumes the DNA boundary-tile source of chapter 3393, the six-boundary coverage surface of chapter 3470, the finite trigger algebra of chapter 3478, the trigger-energy partition packet of chapter 3806, and the trigger-blocker duality packet of chapter 3801. The packet names C_{WRR} as finite cube, layer, transfer-square, count, and refusal data; it is not a biological causal model, stochastic evolutionary law, exhaustive code table, or hidden codon semantics.

Definition 3840.1 (WRR hotspot cube carrier)**.** *A* WRRHotspotCube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (V, A, L, T, M, Q, H, C, P, N)$$

with the following displayed rows:

1. *V is the eight-vertex row for $C_{\mathrm{WRR}} = \{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}, \mathrm{AAA}, \mathrm{AAG}, \mathrm{AGA}, \mathrm{AGG}\}$.*
2. *A is the six-bit atlas row recording the free coordinates b_1^s, b_2^f, b_3^f and fixed coordinates $b_1^f = 0$, $b_2^s = 1$, and $b_3^s = 1$.*
3. *L is the layer-classifier row recording the $\mathrm{U} \times R \times R$ and $\mathrm{A} \times R \times R$ layers and the standard $3 + 2 + 2 + 1$ pattern.*
4. *T is the transfer-square row for $\{\mathrm{UGA}, \mathrm{UGG}, \mathrm{AGA}, \mathrm{AGG}\}$ and its Stop/Trp/Arg transition surface.*
5. *M is the reassignment-count ledger recording the codon counts $8, 10, 15, 0, 3, 0, 8, 8$ and total 52.*
6. *Q is the Q_3 score row recording that the displayed cube is the unique highest-scoring axis-aligned cube among the 160 displayed candidates.*
7. *H records componentwise* hsame *transport for vertex, atlas, layer, transfer-square, count, score, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the source hotspot-cube vision through $V, A, L, T, M, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* WRRHotspotCube$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, TriggerEnergyPartition$^{\uparrow}$, TriggerBlockerDuality$^{\uparrow}$, *the source hotspot-cube vision,* BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$ *row.*
The carrier has no coordinate for biochemical mechanism, evolutionary ancestry, exhaustive biological coverage, stochastic causality, host codon semantics, or an unlisted reassignment table.

Lean4 ✓ `BEDC.Derived.WRRHotspotCubeUp.WRRHotspotCubeTasteGate single carrier alignment`

Theorem 3840.2 (WRR hotspot cube NameCert obligations)**.** *For every accepted* $\mathsf{WRRHotspotCube}^{\uparrow}$ *carrier* $W = (V, A, L, T, M, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$ *surface consists exactly of vertex exposure, six-bit atlas exposure, layer-classifier exposure, transfer-square exposure, reassignment-count ledger exactness,* Q_3 *score-row exposure, componentwise transport, continuation replay, package provenance, and local naming through the displayed rows* $V, A, L, T, M, Q, H, C, P, N$*.*

Proof. Project the carrier of Definition 3840.1. The vertex row V and atlas row A name the finite cube, while L and T expose the two layer and transfer-square classifiers already displayed in the source vision.

The count and score rows are finite ledgers. The row M records the eight codon-level reassignment counts and their total, and Q records the comparison against the displayed family of axis-aligned Q_3 cubes.

Transport, replay, provenance, and local naming preserve exactly those rows. Since the carrier has no coordinate for biochemical mechanism, evolutionary ancestry, stochastic causality, exhaustive biological coverage, hidden codon semantics, or an unlisted table, the listed rows exhaust the NameCert obligations. □

Theorem 3840.3 (WRR hotspot cube dependency handoff)**.** *Every downstream trigger-spectrum consumer of an accepted* $\mathsf{WRRHotspotCube}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{SixBoundaryTileCoverage}^{\uparrow},\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow},\ \mathsf{TriggerEnergyPartition}^{\uparrow},$$

The handoff exposes the cube vertices, atlas coordinates, layer classifier, transfer square, reassignment count ledger, and highest-score row before any consumer treats the hotspot cube as a finite NameCert packet.

Proof. The provenance row P lists the sibling dependency surface. The DNA boundary-tile packet supplies the source codon-window rows, the six-boundary coverage packet supplies the finite coverage coordinates, and the trigger algebra supplies the finite support and coverage-trigger readback.

The trigger-energy partition and trigger-blocker duality packets then supply the partition and blocker rows used by trigger-spectrum consumers. Only after those sibling rows are visible can V, A, L, T, M, Q be read as the hotspot-cube NameCert surface.

The structural rows H, C, P, N preserve and name the same route. Therefore the handoff is a finite dependency chain, not a biological mechanism, stochastic law, or hidden codon table. □

Falsifiable prediction: Any accepted $\mathsf{WRRHotspotCube}^{\uparrow}$ consumer that changes one of the eight displayed vertex rows or one of the count entries $8, 10, 15, 0, 3, 0, 8, 8$ must change the total-count row M or reject the packet.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra, trigger-energy-partition, trigger-blocker-duality: none is bijective with $\mathsf{WRRHotspotCube}^{\uparrow}$ because this packet jointly carries the WRR vertex cube, six-bit atlas row, layer classifier, transfer square, reassignment-count ledger, highest-score row, transport, replay, provenance, and local naming.

The Lean follow-up for this chapter must provide `Nontrivial WRRHotspotCubeUp` and `FieldFaithful WRRHotspotCubeUp` TasteGate instances under `BEDC.Derived.WRRHotspotCubeUp`, preserving $V, A, L, T, M, Q, H,$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}$.

Constructive story: A $\mathsf{WRRHotspotCube}^{\uparrow}$ packet is generated from eight WRR cube vertices, six-bit atlas coordinates, layer-classifier rows, transfer-square rows, reassignment-count ledgers, highest-score rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3840.1, Definition 3840.2, Definition 3840.3, the falsifiable prediction, and the independence witness seed the finite WRR hotspot cube packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WRRHotspotCube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close biochemical causality, evolutionary ancestry, exhaustive biological coverage, stochastic semantics, hidden codon semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vertex exposure, atlas exposure, layer and transfer-square classifier exactness, count-ledger exactness, highest-score row exposure, Nontrivial WRRHotspotCubeUp, and FieldFaithful WRRHotspotCubeUp for BEDC.Derived.WRRHotspotCubeUp.

Chapter 3841

A Concrete Naming Certificate for $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$

Bishop–Bridges nested interval/regular Cauchy construction treats a decreasing located interval chain with a displayed diameter schedule as finite approximation data before any real point is read. The $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ packet records that handoff as a NameCert route from the nested-interval theorem surface of chapter 2308 and the located endpoint surface of chapter 394 through the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It names the L10 route from nested interval data to regular Cauchy real names; it is not an interval-intersection oracle, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3841.1 (Nested Cauchy interval-selection carrier)**.** *A* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{NestedIntervalTheorem}^{\uparrow}$ *source row, exposing the finite descending interval windows, containment rows, and diameter controls.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, recording located lower and upper endpoint reads for the inspected intervals.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare interval diameter bounds with the requested regular Cauchy precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which endpoint and midpoint observations are requested.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy handoff row extracted from the nested interval windows and tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, L, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for interval, located endpoint, dyadic tolerance, stream-window, regular readback, real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a nested interval request through located endpoint reads, dyadic tolerances, stream windows, regular readback, and real seal handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$, $\mathsf{NestedIntervalTheorem}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $I, L, D, W, R, E, H, C, P, N$. It has no coordinate for a completed host intersection point, a quotient real representative, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedCauchyIntervalSelectionUp.NestedCauchyIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 3841.2 (Nested Cauchy interval-selection NameCert obligations)**.** *For every accepted* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *carrier $S = (I, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}}$ *surface consists exactly of carrier admission for the displayed rows, nested-interval exposure through I, located endpoint exposure through L, dyadic tolerance exactness through D, finite-window exposure through W, regular Cauchy handoff through R, terminal real seal confinement through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3841.1. The rows $I, L, D, W, R, E, H, C, P, N$ are exactly the data visible to the local classifier.

The interval source and located endpoint rows are separate. The source row I carries the descending windows and diameter controls, while L carries the endpoint locatedness evidence needed to read those windows as rational approximation data.

The regular Cauchy handoff is downstream. The tolerance ledger D converts displayed diameter bounds into requested precision rows, the window row W supplies finite observations, and R packages the resulting observations as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff.

The real seal and structural rows do not add a hidden selector. The row E is reached only after I, L, D, W, R have been replayed, and H, C, P, N preserve and name the same route. Hence the excluded host intersection, quotient representative, choice, trichotomy, compactness, and ambient-completeness coordinates are absent from the NameCert surface. □

Theorem 3841.3 (Nested Cauchy interval-selection regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{NestedIntervalTheorem}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports a regular Cauchy name read from displayed interval windows and diameter controls, not a completed intersection point or a host equality of real presentations.

Proof. Begin with an accepted carrier $S = (I, L, D, W, R, E, H, C, P, N)$. A regular Cauchy request first reads the nested interval theorem row I and the located endpoint row L.

Next the request consumes the dyadic tolerance ledger D and finite-window row W. These rows identify which displayed interval prefix supplies the requested Cauchy precision.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is then read as the regular Cauchy handoff. A $\mathsf{Real}^{\uparrow}$ consumer may proceed to E, but only through the same route and the structural rows H, C, P, N.

Since no carrier coordinate names a completed host intersection point, quotient real representative, countable-choice schedule, trichotomy split, open-cover compactness proof, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the handoff remains choice-free and finite. □

Falsifiable prediction: Every accepted $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ read exposes NestedIntervalTheorem rows, LocatedInterval endpoint rows, DyadicRatCore tolerance rows, StreamName finite windows, a RegSeqRat handoff, a Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any Real-facing consumer can use the packet.

Independence witness: nested-interval-theorem, locatedinterval, regseqrat, dyadicratcore, streamname, real: $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the nested interval source, located endpoint evidence, dyadic tolerance ledger, finite-window route, regular Cauchy handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedCauchyIntervalSelection}^{\uparrow}$ packet is generated from a NestedIntervalTheorem source row, a LocatedInterval endpoint row, a DyadicRatCore tolerance ledger, StreamName finite windows, a RegSeqRat regular Cauchy handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3841.1, Definition 3841.2, and Definition 3841.3 seed the finite route from nested interval data to regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedCauchyIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, host interval intersection, quotient real equality, selected limit points, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, nested interval exposure, located endpoint exposure, dyadic tolerance exactness, regular Cauchy handoff, Real seal confinement, and field-faithful TasteGate evidence for BEDC.Derived.NestedCauchyIntervalSelectionUp.

Chapter 3842

A Concrete Naming Certificate for $\mathsf{BoundaryPressureLedger}^{\uparrow}$

The $\mathsf{BoundaryPressureLedger}^{\uparrow}$ packet realizes the outer-boundary degree and pressure rows of Definition 9462.1298, Definition 9462.1300, and Definition 9462.1305 as finite BEDC NameCert data. It sits after the median-closure carrier of chapter 3545, the six-boundary trigger coverage route of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the coverage algebra of chapter 3478. The packet records pressure rows for inactive boundary codons before any trigger or class-pressure consumer may use them; it is not a biological stochastic model, an empirical fitness claim, a hidden codon table, or an unrestricted boundary enumerator.

Definition 3842.1 (Boundary-pressure ledger carrier)**.** *A* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, O, D, P, A, K, H, C, Q, N)$$

with the following displayed rows:

1. *S is the reassigned support row naming the displayed* 13*-codon support $\mathcal{R}$.*
2. *O is the outer-boundary row, listing exactly the inactive codons at Hamming distance one from $\mathcal{R}$.*
3. *D is the boundary-degree ledger, recording $d_{\partial}(z)$ for each displayed boundary codon z.*
4. *P is the boundary-pressure ledger, recording $P_{\partial}(z)$ from the displayed support-neighbor weights.*
5. *A is the aggregate class-pressure row, grouping boundary codons by standard-code class while preserving degree and pressure totals.*
6. *K is the high-pressure corner row naming the inactive envelope corners and the first completion frontier consumed by boundary-closure readers.*
7. *H records componentwise* hsame *transport for support, outer-boundary, degree, pressure, aggregate, corner, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-pressure request through $S, O, D, P, A, K, H, Q, N$.*
9. *Q is the* Pkg *provenance over* $\mathsf{BoundaryPressureLedger}^{\uparrow}$*,* $\mathsf{MedianClosure}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$*,* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$ *row.*

The carrier accepts pressure reads only through $S, O, D, P, A, K, H, C, Q, N$. It has no coordinate for a pressure-row bypass, hidden biological semantics, stochastic selection dynamics, empirical likelihood, or a boundary row outside the displayed finite ledger.

Lean4 ✓ `BEDC.Derived.BoundaryPressureLedgerUp.BoundaryPressureLedgerTasteGate single carrier alignment`

Theorem 3842.2 (Boundary-pressure ledger NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *carrier $B = (S, O, D, P, A, K, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$ *surface consists exactly of support admission through S, outer-boundary exposure through O, degree-ledger exactness through D, pressure-ledger exactness through P, class-pressure aggregation through A, inactive-corner and completion-frontier readback through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Proof. Project the carrier of Definition 3842.1. Its visible pressure data are exactly the rows $S, O, D, P, A, K, H, C, Q, N$.

The source support and boundary rows are finite. The row S names the reassigned support, and O lists the inactive codons at Hamming distance one from that support.

The degree and pressure rows are downstream of the same source. The row D counts displayed support neighbors for each boundary codon, while P sums the displayed support-neighbor weights. The aggregate row A groups those same entries by class without changing their provenance.

The corner, transport, replay, provenance, and naming rows preserve the ledger. Because no coordinate bypasses D and P, a consumer cannot obtain pressure data from hidden biological semantics, stochastic selection dynamics, empirical likelihood, or an unrestricted boundary enumerator. □

Lean4 ✓ `BEDC.Derived.BoundaryPressureLedgerUp.BoundaryPressureLedgerNamecertObligations`

Theorem 3842.3 (Boundary-pressure trigger handoff)**.** *Every boundary-trigger or class-pressure consumer of an accepted* $\mathsf{BoundaryPressureLedger}^{\uparrow}$ *packet factors through*

$$S \longmapsto O \longmapsto D \longmapsto P \longmapsto A \longmapsto K \longmapsto H, C, Q, N.$$

In particular, any consumer preserving $\mathsf{MedianClosure}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, *and* $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$ *provenance must read the degree and pressure ledgers before it can use high-pressure inactive boundary codons.*

Proof. Begin with a boundary-trigger request. The provenance row Q names the median-closure and boundary-trigger siblings, but the request cannot read those siblings as pressure rows unless the local support row S and outer-boundary row O have first been displayed.

The pressure data then pass through D and P. Degree is counted from displayed support neighbors, and pressure is the displayed weighted sum over the same neighbor relation.

The aggregate and corner rows are read only after the degree and pressure ledger. The row A groups class totals, while K names the inactive envelope corners and completion-frontier entries consumed by boundary-closure readers.

Finally H, C, Q, N transport, replay, source, and name the same finite route. Thus a pressure-row bypass would refute the packet: it would use high-pressure boundary data without passing through the displayed degree and pressure ledgers. □

Falsifiable prediction: A pressure-row bypass refutes $\mathsf{BoundaryPressureLedger}^{\uparrow}$: any accepted boundary-trigger consumer that reaches high-pressure inactive codons without first reading $S, O, D, P, A, K, H, C, Q$ is outside the packet.

Independence witness: median-closure, six-boundary-tile-coverage, dna-boundary-trigger-closure, window-coverage-trigger-algebra: $\mathsf{BoundaryPressureLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the outer-boundary row, degree ledger, pressure ledger, class aggregation, inactive-corner frontier, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.BoundaryPressureLedgerUp.TasteGate` must provide explicit $\mathsf{Nontrivial\ BoundaryPressureLedger}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundaryPressureLedger}^{\uparrow}$ instances preserving $S, O, D, P, A, K, H, C, Q, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryPressureLedger}^{\uparrow}$ packet is generated from a reassigned support row, an outer-boundary row, boundary-degree and boundary-pressure ledgers, class-pressure aggregation, inactive-corner frontier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3842.1, Definition 3842.2, and Definition 3842.3 seed the finite pressure-ledger route for DNA boundary-trigger consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryPressureLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, biological stochastic semantics, empirical likelihood, hidden codon semantics, unrestricted boundary enumeration, or pressure-row bypasses.

Upgrade path: Obligation closure requires checked support admission, outer-boundary exposure, degree-ledger exactness, pressure-ledger exactness, class aggregation, trigger handoff, Nontrivial BoundaryPressureLedgerUp, and FieldFaithful BoundaryPressureLedgerUp for BEDC.Derived.BoundaryPressureLedgerUp.

Chapter 3843

A Concrete Naming Certificate for $\mathsf{CompletionAdjoint}^{\uparrow}$

Following Mac Lane and Bishop–Bridges, the $\mathsf{CompletionAdjoint}^{\uparrow}$ packet records the Cauchy completion reflector as a finite adjunction certificate. It sits after the reflector surface of chapter 1585, the functorial completion surface of chapter 423, the universal extension ledger of chapter 397, the reflective-subcategory boundary of chapter 1793, the regular rational and stream-window interfaces of chapter 126 and chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the displayed unit, counit, and universal-property rows; it is not a quotient completion, a selected limit object, an ambient reflective subcategory theorem, or an appeal to countable choice.

Definition 3843.1 (Completion adjoint carrier)**.** *A* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, F, U, V, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row from chapter 1585, recording the source-to-complete handoff through the Cauchy completion reflector surface.*
2. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row from chapter 423, carrying the functorial source, regular-Cauchy, seal, and classifier data.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row from chapter 397, recording extension existence, extension uniqueness, and unit compatibility for the same displayed source.*
4. *V is the* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$ *visibility row from chapter 1793: it records that complete objects are read through the reflected completion surface, not through a host subcategory object.*
5. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule for completion requests, approximating maps, unit reads, and counit reads.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular Cauchy names inspected by F and by the universal-property row U.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for all approximation and triangle reads used by the unit-counit ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after R, F, U, V, S, Q, D have been displayed.*

9. *H records componentwise* hsame *transport for the reflector, functor, universal-property, reflective-subcategory, stream, readback, dyadic, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from any completion-adjunction consumer through the rows R, F, U, V, S, Q, D, E and then through the structural rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CompletionAdjoint}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected limits, host reflective subcategories, functor extensionality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionAdjointUp.CompletionAdjointTasteGate single carrier alignment`

Theorem 3843.2 (Completion adjoint NameCert obligations)**.** *For every accepted* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier $A = (R, F, U, V, S, Q, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the twelve displayed rows; reflector exposure through R; functorial completion exposure through F; universal-property exposure through U; reflective-subcategory visibility through V; finite request scheduling through S; regular rational readback through Q; dyadic tolerance accounting through D; terminal RealUp sealing through E; componentwise transport through H; continuation replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier rows. The adjunction-facing data are precisely the reflector row R, the functor row F, the universal-property row U, the reflective-subcategory visibility row V, and the finite analytic rows S, Q, D, E.

Read the structural rows separately. The transport row H preserves those displayed rows componentwise, while C replays the same route for completion-adjunction consumers.

The provenance and naming rows P, N attach the packet to the cited completion, stream, regular-readback, dyadic, and Real certificates. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 3843.3 (Completion adjoint unit-counit ledger)**.** *Every unit or counit read exported by an accepted* $\mathsf{CompletionAdjoint}^{\uparrow}$ *carrier factors through the finite route*

$$R, F, U, V \longrightarrow S, Q, D \longrightarrow E, H, C, P, N.$$

The unit side reads the source-to-completion handoff through the reflector and functor rows, while the counit side reads the complete-object return through the universal-property and reflective-subcategory visibility rows. Both sides use the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the structural ledger is inspected.*

Proof. Start with the rows R, F, U, V. They are the only coordinates in the packet that can name the reflector, functor, extension, uniqueness, and reflected-object handoff needed by an adjunction consumer.

Next read S, Q, D, E. These rows supply finite request schedules, regular rational readback, dyadic tolerance accounting, and the terminal Real seal for both the unit-facing and counit-facing directions.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same displayed ledger, so the unit and counit routes cannot escape to a quotient completion, selected limit, host reflective subcategory, functor-extensional equality, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionAdjoint}^{\uparrow}$ packet is generated from CauchyCompletionReflector, CauchyCompletionFunctor, CompletionUniversalProperty, CauchyCompletionReflectiveSubcategory, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3843.1, Definition 3843.2, and Definition 3843.3 seed the finite adjunction certificate for Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected limits, host reflective subcategories, functor extensionality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, reflector exposure, functorial completion exposure, universal-property compatibility, unit-counit ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompletionAdjointUp.

Chapter 3844

A Concrete Naming Certificate for $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$

Following Arzela–Ascoli and Bishop–Bridges equicontinuity arguments, the $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ packet records the finite-net route by which a compact source and a finite family of continuous maps expose one shared modulus before any open-cover compactness or choice principle is read. It consumes compact-metric data from chapter 129, total-bounded finite nets from chapter 127, continuous-map graph and pointwise modulus rows from chapter 130, compact uniform-continuity handoff data from chapter 3444, and the downstream modulus surface of chapter 369. The certificate is a finite uniform-equicontinuity route for compact-window consumers, not a selected subcover, not a family-choice functional, and not ambient completeness of the codomain.

Definition 3844.1 (UniformEquicontinuityModulus carrier)**.** *A* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, T, F, W, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the compact metric carrier, distance ledger, and compact finite-net surface used by the family.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row over the same source, listing inspected centers and dyadic source-radius budgets.*
3. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose pointwise modulus and graph rows are inspected at the centers named by T.*
4. *W is the finite target-window row, recording the common tolerance window and the family indices admitted by the packet.*
5. *M is the exported* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained by a finite fold over the centerwise and familywise modulus witnesses displayed by F against T and W.*
6. *H records componentwise* hsame *transport for the compact source, finite-net, continuous-family, window, folded-modulus, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a compact-window consumer through K, T, F, W, M and then through the structural rows.*

8. *P is the* Pkg *provenance over* UniformEquicontinuityModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, ContinuousMap$^{\uparrow}$, HeineCantor$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{UniformEquicontinuityModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected finite subcovers, countable choice over the family, quotient equality of maps, host real trichotomy, or ambient codomain completeness.

Lean4 ✓ `BEDC.Derived.UniformEquicontinuityModulusUp.UniformEquicontinuityModulusTasteGate single carrier alignment`

Theorem 3844.2 (UniformEquicontinuityModulus NameCert obligations)**.** *For every accepted* UniformEquicontinuityModulus$^{\uparrow}$ *carrier $U = (K, T, F, W, M, H, C, P, N)$, the local* NameCert$_{\text{UniformEquicontinuityModulus}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the finite packet $K, T, F, W, M, H, C, P, N$;*
2. *finite-window classifier: every family and target-tolerance read is confined to the displayed window row W and the listed* ContinuousMap$^{\uparrow}$ *rows in F;*
3. *compact finite-net exactness: the only compactness data read by the classifier are the* CompactMetric$^{\uparrow}$ *source row K and the* TotallyBounded$^{\uparrow}$ *finite-net row T;*
4. *uniform-modulus export: the shared modulus row M is obtained by the finite fold over the centerwise pointwise-modulus rows in F and the compact finite-net route already exposed by K, T;*
5. *non-escape: no accepted consumer receives an open-cover compactness object, selected subcover, family-choice functional, quotient map equality, host real trichotomy, or ambient completeness theorem.*

Proof. Project the carrier row by row. The analytic data available before the exported modulus are exactly the compact source K, the finite-net inventory T, the finite continuous-family rows F, and the common target window W.

The compact finite-net exactness clause follows from the route through K and T. Since T lists the centers and radius budgets as finite data, the classifier never needs an open-cover subcover or a choice over covers.

The uniform-modulus export is then the finite fold stored in M. Each fold input is a pointwise modulus row from F at a center named by T and a tolerance named by W; the Heine–Cantor handoff supplies the compact-window reading of those rows, while the final row is presented to the UniformModulus$^{\uparrow}$ surface.

Finally read the structural rows. The transport row H, replay row C, provenance row P, and local naming row N preserve the same finite route and contain no coordinate for the excluded cover, choice, equality, trichotomy, or completeness data. □

Falsifiable prediction: An accepted UniformEquicontinuityModulus$^{\uparrow}$ read is refuted if it exports a shared family modulus without displaying the compact source row, total-bounded finite-net row, finite continuous-family rows, target-window row, folded modulus row, transport, replay, provenance, and local naming rows.

Independence witness: compactmetric, totallybounded, continuousmap, heine-cantor, uniformmodulus: none alone is bijective with UniformEquicontinuityModulus$^{\uparrow}$ because the packet jointly records a compact source, finite net, finite continuous family, common target window, folded uniform modulus, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ packet is generated from CompactMetric source data, TotallyBounded finite nets, ContinuousMap family rows, a common finite target window, a finite folded UniformModulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3844.1 and Definition 3844.2 seed the compact finite-net route from a finite continuous family to a shared UniformModulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformEquicontinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable family choice, quotient equality of maps, host real trichotomy, ambient codomain completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window classification, compact finite-net exactness, uniform-modulus export, transport, replay, provenance, and local naming for BEDC.Derived.UniformEquicontinuityModulusUp.

Chapter 3845

A Concrete Naming Certificate for $\mathsf{LimitSpace}^{\uparrow}$

Following Kelley and Bishop–Bridges convergence-space and limit-space terminology, the $\mathsf{LimitSpace}^{\uparrow}$ packet records the finite interface by which a displayed Cauchy-filter or filter-base row is read against an observation schedule, a dyadic tolerance ledger, a regular rational readback, and a terminal real limit seal. It consumes the Cauchy-filter surface of chapter 333, the finite observation schedule of chapter 126, the dyadic tolerance core of chapter 291, the regular rational route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a convergence-space interface, not a quotient filter, selected limit, host equality of convergent objects, or ambient completeness theorem.

Definition 3845.1 (LimitSpace carrier)**.** *A* $\mathsf{LimitSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, D, R, L_0, H, C, P, N)$$

with the following displayed rows:

1. *F is a* $\mathsf{CauchyFilter}^{\uparrow}$ *or filter-base row, carrying the finite filter requests and Cauchy-stability witnesses permitted at this limit-space site;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule that records which finite windows of F are inspected;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the scheduled and tolerated observations into regular rational data;*
5. *L_0 is the* $\mathsf{Real}^{\uparrow}$ *limit-seal row reached only after F, S, D, R have been displayed;*
6. *H records componentwise* hsame *transport for the filter, schedule, tolerance, readback, limit seal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a convergence request through F, S, D, R, L_0, H, P, N;*
8. *P is the* Pkg *provenance over* $\mathsf{LimitSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, host equality of convergent filters, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge evidence.*

Lean4 ✓ `BEDC.Derived.LimitSpaceUp.LimitSpaceTasteGate single carrier alignment`

Theorem 3845.2 (LimitSpace NameCert obligations)**.** *For every accepted* $\mathsf{LimitSpace}^{\uparrow}$ *carrier*

$$L = (F, S, D, R, L_0, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Cauchy-filter or filter-base row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, Real limit seal, transport, replay, provenance, and local naming row;*
2. *filter-source exposure, requiring every limit-space consumer to read F before any scheduled observation, tolerance comparison, readback, or limit seal is inspected;*
3. *schedule and tolerance discipline, requiring S and D to determine the finite windows and dyadic budgets used by the readback row;*
4. *regular readback discipline, requiring R to be downstream of F, S, D and upstream of the terminal seal;*
5. *limit-seal non-escape, requiring the* $\mathsf{Real}^{\uparrow}$ *row L_0 to be read only as a seal for the displayed readback route;*
6. *structural preservation through H, C, P, N;*
7. *refusal of quotient filters, selected limits, countable choice, host equality of convergent filters, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge evidence.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3845.1. The tuple displays exactly the Cauchy-filter or filter-base source, finite schedule, dyadic tolerance ledger, regular rational readback, real limit seal, and structural rows.

The source obligation is first. A convergence consumer cannot reach a scheduled observation or terminal seal without reading F, because no other row carries filter-source data.

The schedule and tolerance obligations are then forced by S and D. They determine the finite windows and dyadic budgets that the regular readback row may inspect, so R is downstream of the displayed observation discipline rather than a separate selected representative.

Finally L_0 is exposed only as the real seal for that readback route, while H, C, P, N preserve, replay, source, and name the same finite packet. Since the excluded principles have no carrier coordinate, they cannot become NameCert obligations. □

Theorem 3845.3 (LimitSpace Cauchy-filter consumer route)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LimitSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$F \longmapsto S, D \longmapsto R \longmapsto L_0 \longmapsto H, C, P, N.$$

The consumer first reads the Cauchy-filter or filter-base row, then the StreamName schedule and DyadicRatCore tolerance ledger, then the RegSeqRat readback, and only then the Real limit seal. The route exports no quotient filter, no selected limit, no host equality of convergent filters, no countable-choice schedule, and no ambient completeness theorem.

Proof. Project the accepted carrier. The first row F is the only source for Cauchy-filter or filter-base data, so every filter-facing read starts there.

The next step is the finite observation discipline. The schedule row S names the inspected windows and the tolerance row D names the dyadic budgets attached to them; without those two rows the regular readback row has no displayed finite request to answer.

The readback and seal rows are downstream. The row R converts the scheduled observations into regular rational data, and L_0 seals the resulting route as a $\mathsf{Real}^{\uparrow}$ limit row without selecting an external limit object.

The structural rows H, C, P, N preserve and name the same route. Because the carrier contains no quotient-filter coordinate, selected-limit coordinate, host equality row, choice schedule, or ambient completeness row, every accepted consumer factors through the displayed path. □

Falsifiable prediction: An accepted $\mathsf{LimitSpace}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ limit seal without first displaying the $\mathsf{CauchyFilter}^{\uparrow}$ or filter-base row, the $\mathsf{StreamName}^{\uparrow}$ schedule, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger refutes the carrier surface.

Independence witness: cauchyfilter, streamname, dyadicratcore, regseqrat, real: none is bijective to $\mathsf{LimitSpace}^{\uparrow}$ because this packet jointly carries a filter-source row, observation schedule, tolerance ledger, regular readback, terminal limit seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{LimitSpace}^{\uparrow}$ packet is generated from a CauchyFilter or filter-base row, StreamName observation schedule, DyadicRatCore tolerance ledger, RegSeqRat readback row, Real limit seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3845.1, Definition 3845.2, and Definition 3845.3 seed the finite convergence-space interface from CauchyFilter rows through StreamName, DyadicRatCore, RegSeqRat, and Real limit-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LimitSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, host equality of convergent filters, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-source exposure, schedule and tolerance discipline, regular readback discipline, Real limit-seal non-escape, and structural preservation for BEDC.Derived.LimitSpaceUp.

$\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ reads compactness-facing finite-window criteria through chapter 129, chapter 127, and chapter 3844 before its Arzela–Ascoli and completion consumers inspect the local rows.

Chapter 3846

A Concrete Naming Certificate for $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$

Following Kolmogorov–Riesz and Frechet compactness in constructive finite-window form, the $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet records the BEDC route by which compact metric control, total boundedness, uniform equicontinuity, completion readiness, and terminal Real readbacks are named as one compactness-facing criterion. It sits after the compact-metric surface of chapter 129, the total-bounded source of chapter 127, the uniform-equicontinuity modulus of chapter 3844, the complete-metric handoff of chapter 128, and the terminal real-name seal of chapter 38. Its compactness route also consumes bounded-set rows from chapter 1757, equicontinuous-sequence rows from chapter 4356, and finite-net compact-metric product rows from chapter 1822. The carrier names finite net and completion rows only; it does not assert open-cover compactness, choose subsequences, select subcovers, or import ambient completeness outside the displayed packet.

As a carrier dependency, the packet consumes the compactness route through chapter 129, the equicontinuity-family route through chapter 413, and the real and metric boundary routes through chapter 38 and chapter 57. These rows are part of the $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ admission surface: a compactness consumer must see the compact metric source, family-equicontinuity ledger, Real readback seal, and metric carrier before the Frechet–Kolmogorov criterion row is exported.

Definition 3846.1 (Frechet-Kolmogorov compactness carrier)**.** *A* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, E, Q, C, S, R, H, A, P, N)$$

with the following rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the displayed metric and compactness-facing finite-net interface.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row recording the finite ε-net budget consumed by the criterion.*
3. *E is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row binding a shared modulus to the same finite windows.*
4. *Q is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion boundary row, used only after the finite net and modulus rows are visible.*
5. *C is the compactness-criterion row relating the metric, total-bounded, equicontinuity, and completion rows as one finite BEDC route.*

6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the inspected net cells, modulus requests, and Cauchy-readback windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance comparisons.*
8. *H records componentwise* hsame *transport for all displayed source, net, modulus, completion, readback, and naming rows.*
9. *A records displayed* Cont *replay of the same finite route.*
10. *P is the* Pkg *provenance over* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, T, E, Q, C, S, R, H, A, P, N$. It has no coordinate for an open-cover compactness theorem, selected subsequence, selected subcover, quotient completion equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3846.2 (Frechet-Kolmogorov compactness NameCert obligations)**.** *For every accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier $K = (M, T, E, Q, C, S, R, H, A, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCor}}$ *surface consists exactly of compact-metric source exposure through M; total-bounded finite-net exposure through T; uniform-modulus exposure through E; completion-boundary exposure through Q; criterion compatibility through C; finite-window and rational readback exposure through S, R; componentwise transport through H; displayed replay through A; provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The coordinates are precisely the compact-metric source, total-bounded net, uniform-modulus, completion-boundary, criterion, finite-window, readback, transport, replay, provenance, and naming rows.

Read the source rows in order. The compact-metric row gives the metric-facing carrier; the total-bounded row supplies finite nets over that metric; the uniform-equicontinuity row uses the same windows; and the completion row is reached only through those displayed finite controls.

The structural rows H, A, P, N transport, replay, source, and name that same route. Since no carrier coordinate names a selected subsequence, selected subcover, open-cover compactness proof, quotient equality, or ambient completeness theorem, the listed rows exhaust the local certificate surface. □

Theorem 3846.3 (Frechet-Kolmogorov compactness total-bounded handoff)**.** *Every compactness-facing consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier must read the total-bounded row T through the compact-metric source M and the uniform-modulus row E before it inspects the StreamName and RegSeqRat readback rows S, R. Thus finite-net evidence is exported as a displayed BEDC handoff, not as a choice-selected dense set or an unrecorded cover refinement.*

Proof. Begin with the compact-metric source row. It fixes the metric and the carrier surface on which the total-bounded net is meaningful.

The total-bounded row then exposes the finite net budget, while the uniform-modulus row forces all later compactness requests to use the same finite tolerance windows. Only after those rows are visible can the StreamName and RegSeqRat rows read the inspected windows and rational tolerances.

Therefore the consumer receives a finite-net handoff through M, T, E, S, R and not a selected dense subset, a selected subcover, or an unrecorded refinement outside the packet. □

Theorem 3846.4 (Frechet-Kolmogorov compactness sibling route)**.** *Every compactness-facing export of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier consumes the compactness, total-boundedness, and equicontinuity sibling surfaces before it exports compactness of the function family. The route enters through the* $\mathsf{CompactMetric}^{\uparrow}$ *source of chapter 129, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row of chapter 127, and the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row of chapter 3844; it may then hand the assembled finite-window criterion to the Arzela–Ascoli finite-net surface of chapter 7576. The packet exports no open-cover compactness proof, selected subsequence, selected subcover, quotient completion equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier and NameCert obligations. The rows M, T, E are respectively the compact-metric source, total-bounded finite-net budget, and uniform-equicontinuity modulus, so no compactness-facing read can begin outside those sibling surfaces.

Use the total-bounded handoff. It forces finite-net evidence through the compact-metric source and the shared modulus before StreamName and RegSeqRat readbacks are inspected.

Finally use the completion boundary theorem. The row Q can be consumed only after the compactness, total-boundedness, and equicontinuity rows are visible, and the structural rows H, A, P, N preserve the same finite route for any Arzela–Ascoli or completion-facing consumer. □

Theorem 3846.5 (Frechet-Kolmogorov compactness Arzela-Ascoli dependency)**.** *Every Arzela–Ascoli-facing consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier first reads the* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already exposed by the carrier. The Arzela–Ascoli finite-net surface may consume this route, but it does not bypass the compact-metric source, total-bounded handoff, uniform-modulus row, completion boundary, finite windows, rational readback, or Real seal.*

Proof. Apply the sibling route of Definition 3846.4. It already directs compactness-facing exports through the CompactMetric, TotallyBounded, and UniformEquicontinuityModulus sibling surfaces before any Arzela–Ascoli finite-net consumer receives the assembled criterion.

Use the total-bounded handoff of Definition 3846.3. The finite-net budget is read through the compact-metric source and shared modulus before the StreamName and RegSeqRat rows expose inspected windows and rational tolerance comparisons.

Finally apply the completion boundary of Definition 3846.6. The CompleteMetric row and Real-facing seal remain downstream boundaries over the same finite data, so an Arzela–Ascoli-facing consumer cannot obtain open-cover compactness, a selected subsequence, a selected subcover, quotient completion equality, or ambient Real completeness outside the displayed route. □

Theorem 3846.6 (Frechet-Kolmogorov compactness completion boundary)**.** *The completion-facing part of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier is confined to the displayed row* Q *after the compact-metric source, total-bounded net, and uniform-modulus rows have been read. It may hand a downstream completion consumer a finite Cauchy-window boundary, but it does not claim quotient completion equality, a selected limit representative, sequential compactness, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the route through M, T, E first. These rows supply the metric source, finite-net budget, and shared modulus required before any completion boundary can be inspected.

The completion row Q is therefore a boundary row over already displayed finite data. The readback rows S, R and the structural rows H, A, P, N keep the same route visible under windowing, rational comparison, transport, replay, provenance, and naming.

Because the carrier contains no quotient-equality, selected-limit, sequential-compactness, open-cover, or ambient-completeness coordinate, a completion consumer cannot obtain any of those objects from the local packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet is generated from a CompactMetric source row, a TotallyBounded finite-net row, a UniformEquicontinuityModulus row, a CompleteMetric boundary row, StreamName finite windows, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3846.1, Definition 3846.2, Definition 3846.3, Definition 3846.4, Definition 3846.5, and Definition 3846.6 seed the finite compactness criterion over compact-metric, total-bounded, uniform-modulus, completion-boundary, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, sequential compactness, selected subsequences, selected subcovers, quotient completion equality, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, uniform-modulus compatibility, completion-boundary nonescape, readback stability, transport, replay, provenance, and local naming for BEDC.Derived.FrechetKolmogoro

Theorem 3846.7 (Frechet-Kolmogorov finite-window compactness handoff)**.** *Every accepted* FrechetKolmogorovCo *compactness read that is exported to a finite-net or compact-metric consumer must route through the displayed boundedness row, finite translation-modulus row, tail-cutoff row, finite-net extraction row,* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Met}^{\uparrow}$ *distance rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FrechetKolmogorovCompactness}^{\uparrow}}$ *row before compactness is public. In carrier notation the handoff refines the local route*

$$M, T, E, Q, C, S, R, H, A, P, N$$

by requiring boundedness, translation control, tail cutoff, finite nets, Real readback, rational readback, and metric-distance rows to be visible before the consumer can use the compactness criterion. The handoff exports no ambient compactness theorem, open-cover compactness, countable choice, selected subsequence, selected subcover, quotient completion equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the local NameCert obligations of Definition 3846.2. They force the compact-metric source M, total-bounded finite-net row T, uniform-modulus row E, completion boundary Q, finite-window row S, readback row R, and structural rows H, A, P, N to be displayed before a compactness-facing consumer receives the packet.

Read the additional finite-window controls through the same route. The boundedness row is supplied by the bounded-set dependency already named in the chapter introduction, the finite translation-modulus and tail-cutoff rows are consumed as uniform-window controls, and the finite-net extraction row is read through the total-bounded handoff of Definition 3846.3. The $\mathsf{Met}^{\uparrow}$ distance rows and the $\mathsf{Real}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}$ readbacks attach the tolerance comparisons to the displayed windows.

Finally apply the completion boundary of Definition 3846.6. The structural rows only transport, replay, source, and locally name the same finite handoff, so no coordinate remains for ambient compactness, open-cover compactness, countable choice, selected subsequences, selected subcovers, quotient completion equality, or ambient Real completeness. □

The finite compactness route of $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ feeds the Arzela–Ascoli finite-modulus criterion of chapter 3568 only after the compact metric source of chapter 129, the finite-net

discipline of chapter 127, and the uniform-equicontinuity modulus of chapter 3844 have been displayed. A compact-uniform-continuity consumer therefore receives compact metric, total-bounded, uniform-modulus, completion, transport, replay, provenance, and local NameCert rows, not an open-cover compactness theorem or a selected subsequence.

Theorem 3846.8 (Frechet–Kolmogorov Arzela–Ascoli dependency route)**.** *Every accepted* FrechetKolmogorovComp *compactness read consumes the equicontinuity and finite-subfamily surface of chapter 412 before it exports compactness for function families. The Frechet–Kolmogorov packet therefore routes compact metric data, total-bounded finite nets, bounded-family rows, Arzela–Ascoli equicontinuity rows, transport, replay, provenance, and local* NameCert *naming as one displayed dependency chain.*

Proof. Project the carrier through the local Frechet–Kolmogorov route. The compact source and finite-net rows are visible before any family compactness conclusion is offered to a downstream consumer.

Now read the Arzela–Ascoli sibling. Its equicontinuity and finite-subfamily surface supplies the function-family control that the Frechet–Kolmogorov packet consumes before exporting compactness.

The structural rows only transport, replay, source, and name this chain. Hence the dependency is horizontal and finite, not an ambient compactness theorem or a selected subsequence principle. □

Theorem 3846.9 (Frechet–Kolmogorov Arzela–Ascoli compact handoff)**.** *Every accepted* FrechetKolmogorovComp *handoff to an Arzela–Ascoli consumer must first display the compact-metric finite nets of chapter 129, the total-bounded finite-net rows of chapter 127, and the equicontinuity surface of chapter 412. The exported compactness packet is therefore a finite* BHist *route through compact metric, total-boundedness, Arzela–Ascoli finite-window selection, transport, replay, provenance, and local* NameCert *rows.*

Proof. Read the accepted Frechet–Kolmogorov carrier through its compact metric source. The compact-metric row fixes the metric-facing source, while the total-bounded row supplies the finite nets that any compactness-facing consumer can inspect.

Next read the Arzela–Ascoli dependency. Its equicontinuity and finite-window rows are consumed after those compact metric and total-bounded rows, so the function-family compactness conclusion is not detached from the finite net budget.

The remaining rows only preserve the displayed route. Componentwise hsame transport, Cont replay, provenance, and local naming keep the handoff inside the same finite packet, excluding selected subsequences, open-cover compactness, and ambient completeness. □

Theorem 3846.10 (Frechet–Kolmogorov downstream Arzela–Ascoli link)**.** *Every Arzela–Ascoli finite-net consumer of an accepted* FrechetKolmogorovCompactness$^{\uparrow}$ *packet must first read the* CompactMetric$^{\uparrow}$ *source row, the* TotallyBounded$^{\uparrow}$ *finite-net row, the* UniformEquicontinuityModulus$^{\uparrow}$ *row, the* CompleteMetric$^{\uparrow}$ *completion boundary, the* StreamName$^{\uparrow}$ *observation windows, the* RegSeqRat$^{\uparrow}$ *readback rows, and the* Real$^{\uparrow}$ *seal rows carried by the displayed dependency route. The handoff exports no selected subsequence, selected subcover, open-cover compactness theorem, quotient completion equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the existing Frechet–Kolmogorov carrier through the compactness dependency route. The compact metric source and total-bounded finite-net rows are visible before any function-family compactness read is offered downstream.

Read the Arzela–Ascoli-facing rows next. The dependency route of Definition 3846.8 and the compact handoff of Definition 3846.9 require the uniform-equicontinuity modulus before the finite-net consumer receives the family surface.

Finally inspect the completion and real-readback side. The compact metric total-bounded dependency of Definition 3846.11 keeps the $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows attached to the same finite route. Since no route theorem supplies a selected subsequence, selected subcover, open-cover compactness theorem, quotient completion equality, or ambient completeness coordinate, the downstream Arzela–Ascoli link exports only the displayed finite lattice. □

Theorem 3846.11 (Compact metric total-bounded dependency)**.** *The* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *criterion consumes the compact-metric source of chapter 129 and the total-bounded finite-net budget of chapter 127 before it reads the uniform-equicontinuity row of chapter 3844 and the completion boundary of chapter 128. Thus the later StreamName, RegSeqRat, and Real readbacks remain attached to displayed finite metric and net data, rather than to an unrecorded compactness principle.*

The $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet exposes a real-sequence compactness read only after its compact-metric finite-net rows, bounded-function-family rows, and Arzela–Ascoli finite-window rows have all been displayed. Thus every downstream compactness consumer first passes through the compact source of chapter 129, the bounded family surface of chapter 3522, and the Arzela–Ascoli route of chapter 412; before real-sequence compactness readback, it also consumes the finite epsilon-net chain of chapter 8049. The Frechet–Kolmogorov carrier then contributes only the finite BHist, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows that assemble those dependencies into the real-sequence compactness read.

Theorem 3846.12 (Frechet-Kolmogorov locally convex boundedness route)**.** *Every* $\mathsf{FrechetKolmogorovCompactnes}$ *consumer that reads compactness through locally convex bounded sets must factor through the finite route*

$$\mathsf{LocallyConvexSpace}^{\uparrow}, \mathsf{FrechetSpace}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer must expose the locally convex seminorm row of chapter 3386, the Frechet metric completion row of chapter 3495, the compact metric row of chapter 129, the total bounded row of chapter 127, and the terminal Real seal row of chapter 38 before it may treat a bounded family as Frechet-Kolmogorov compactness data.

Proof. Start with the locally convex seminorm row. It supplies the bounded-set requests that a functional-analysis consumer is allowed to ask of the packet.

The Frechet metric completion row then turns those seminorm requests into a displayed metric-completion boundary. The compact metric row and total bounded row refine that boundary into finite compactness data, so the consumer sees nets and metric windows rather than an ambient bounded-set compactness theorem.

Finally the Real seal row records the scalar readback used by the seminorm and metric estimates. Since each step is an exposed row in the displayed route, no consumer can bypass the seminorm, completion, compact metric, total bounded, or Real-seal surfaces by appealing to host boundedness or selected compactness data. □

Theorem 3846.13 (Frechet–Kolmogorov metric-completion functor route)**.** *Every* $\mathsf{MetricCompletionFunctor}^{\uparrow}$ *consumer of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *carrier must first read the compact-metric source of chapter 129, the total-bounded finite-net surface of chapter 127, the uniform-equicontinuity modulus of chapter 3844, the complete-metric boundary of chapter 128, the StreamName window rows of chapter 126, the RegSeqRat readback of chapter 301, and the Real seal of*

chapter 38. The route exports a displayed compactness-to-completion handoff only; it exports no quotient completion equality, selected compact limit, selected subsequence, selected subcover, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the carrier of Definition 3846.1. Its compactness-facing source rows are the compact metric row, the total-bounded finite-net row, the uniform-modulus row, and the completion boundary row.

Apply the sibling route of Definition 3846.16. It keeps the compact metric, total-bounded, uniform-equicontinuity, RegSeqRat, and Real-seal rows visible as one dependency lattice before any compactness conclusion is exported.

Finally apply the completion boundary of Definition 3846.6. A $\mathsf{MetricCompletionFunctor}^{\uparrow}$ consumer can only repackage that displayed boundary through StreamName and RegSeqRat readbacks, so quotient completion equality, selected compact limits, and ambient completeness are outside the exported surface. □

The $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ packet is read through a substantive compactness sibling lattice rather than as a detached compactness kernel: its equicontinuity branch consumes the finite-modulus Arzela–Ascoli route of chapter 412, its compact source consumes the $\mathsf{CompactMetric}^{\uparrow}$ total-bounded finite-net rows of chapter 129, and its completion handoff is routed through the MetricCompletion completion surface of chapter 1506. These references keep compact metric, total-bounded finite nets, completion, transport, replay, provenance, and local NameCert rows visible before any compactness-facing consumer reads the Frechet–Kolmogorov packet.

Theorem 3846.14 (Frechet-Kolmogorov Sobolev compact-embedding forward link)**.** *Every Sobolev-family consumer that reaches the* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *packet through a compact-embedding route must first display the* $\mathsf{Sobolev}^{\uparrow}$ *source rows, the* $\mathsf{CompactEmbedding}^{\uparrow}$ *handoff of chapter 7166, and the compact-metric and total-bounded rows of chapter 129 and chapter 127. The Frechet–Kolmogorov packet may then consume the finite StreamName, RegSeqRat, completion-boundary, transport, replay, provenance, and local NameCert rows exposed by those siblings, but it does not replace the compact-embedding handoff with selected subsequences, open-cover compactness, quotient function equality, or ambient Real completeness.*

Proof. Start from the compact-embedding side. The $\mathsf{CompactEmbedding}^{\uparrow}$ chapter exposes a finite handoff from Sobolev source windows into compact and complete target rows, so a Sobolev-family read arrives with displayed source, target, readback, transport, replay, provenance, and naming data.

Use the Frechet–Kolmogorov compactness carrier and its sibling route. The rows $M, T, E, Q, S, R, H, A, P, N$ already require compact-metric source exposure, total-bounded finite nets, completion boundary, finite-window readback, rational comparison, transport, replay, provenance, and local naming before compactness-facing data are exported.

The forward link is therefore a dependency route, not a new compactness principle. The Frechet–Kolmogorov consumer reads the Sobolev and compact-embedding siblings as displayed rows and then reuses the compact-metric and total-bounded surfaces, leaving selected subsequences, open-cover compactness, quotient function equality, and ambient Real completeness outside the packet. □

Theorem 3846.15 (Frechet–Kolmogorov hub dependency index)**.** *The hub route for* FrechetKolmogorovCompactne *reads the* $\mathsf{CompactMetric}^{\uparrow}$ *source rows, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net rows, the Arzela–Ascoli finite-modulus rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback rows, the* $\mathsf{Real}^{\uparrow}$ *terminal boundary rows, and the*

MetricCompletion$^{\uparrow}$ *sibling rows before any Frechet–Kolmogorov compactness consumer can inspect the local compactness packet. Thus the top-level input boundary records the same dependency lattice that the child certificates expose: compact metric source, total-bounded budget, finite-modulus equicontinuity, regular rational comparison, Real boundary, completion handoff, transport, replay, provenance, and local* NameCert *data.*

Proof. Read the hub as an ordered certificate boundary. Its child files first expose the name certificate spine and then the compact-metric, total-bounded, Arzela–Ascoli, locally convex, and metric-completion routes.

The rational and Real rows are read through the same sibling lattice. The RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ surfaces supply readback and terminal boundary data before the compactness consumer opens the Frechet–Kolmogorov packet.

Consequently a consumer cannot bypass the sibling rows at the hub boundary. The hub records the dependency lattice directly while leaving all theorem content in child files. □

The FrechetKolmogorovCompactness$^{\uparrow}$ dependency lattice reads compactness only after its sibling surfaces are visible: the compact metric source of chapter 129, the finite-family control supplied by chapter 412, the completion-facing transport of chapter 1506, the regular rational tolerance readback of chapter 301, the terminal real seal of chapter 38, and the tightness ledger of chapter 3795. This makes compactness a displayed route through compact metric, equicontinuity, completion, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, and tightness rows, rather than an ambient subsequence selector, open-cover oracle, or quotient completion equality.

Theorem 3846.16 (Frechet–Kolmogorov compactness sibling route)**.** *The* FrechetKolmogorovCompactness$^{\uparrow}$ *packet consumes the Arzela–Ascoli finite-modulus route of chapter 3568 as its equicontinuity-facing compactness control, and it consumes the local seminorm-to-completion route recorded in Definition 3846.12 as its locally convex boundedness control. Together with the compact metric surface of chapter 129, the total-bounded surface of chapter 127, the equicontinuity radius source of chapter 3844, the compact metric total-bounded source of chapter 4055, the regular rational readback of chapter 301, and the Real seal of chapter 38, these sibling reads keep the compact metric, total-bounded, uniform-equicontinuity, locally convex seminorm, Frechet metric, RegSeqRat, and Real-seal rows visible as one finite dependency lattice rather than as an ambient function-space compactness theorem.*

Proof. Read the compact metric and total-bounded rows first. The compactness-facing consumer must pass through the displayed compact metric packet and its total-bounded control before any finite-cover or subsequence conclusion is available.

Next read the equicontinuity and local boundedness rows. The Arzela–Ascoli finite-modulus route supplies the uniform-equicontinuity control, while the locally convex boundedness route supplies the seminorm-to-completion control.

Finally read the completion, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ seal rows. Since each consumer opens these sibling rows before the compactness conclusion, the packet exports a finite dependency lattice rather than an ambient function-space compactness principle. □

Theorem 3846.17 (Frechet–Kolmogorov bounded-tail compactness route)**.** *An accepted* FrechetKolmogorovCompa *consumer may read total-bounded compactness only after it has exposed both the Arzela–Ascoli finite-modulus route and the locally convex boundedness route. The read passes through* CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *and the displayed completion boundary of Definition 3846.6; it does not use open-cover compactness, selected subsequences, or an ambient choice of compact limit points.*

Proof. Start with the sibling route of Definition 3846.16. It orders the compact metric, total-bounded, Arzela–Ascoli finite-modulus, locally convex seminorm, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as one dependency lattice.

Next apply the locally convex boundedness route of Definition 3846.12. It supplies the bounded finite-tail control needed before the Frechet metric can be read through the completion surface.

Finally combine the total-bounded handoff of Definition 3846.3 with the completion boundary of Definition 3846.6. The handoff gives finite total-bounded data, and the boundary keeps the completion read in $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ form, so no open-cover compactness principle or selected subsequence is introduced. □

Theorem 3846.18 (Frechet–Kolmogorov sibling dependency witness)**.** *Every accepted* FrechetKolmogorovCompact *carrier witnesses a substantive dependency on the compact metric, total-bounded, and Arzela–Ascoli finite-modulus sibling chapters. The compactness criterion may be consumed only after the compact source of chapter 129, the finite-net surface of chapter 127, and the Arzela–Ascoli finite-modulus route of chapter 3568 have all been displayed together with the local carrier rows of Definition 3846.1.*

Proof. Read the compact source first. The row M in Definition 3846.1 is a $\mathsf{CompactMetric}^{\uparrow}$ source row, so the criterion has no compactness-facing read until the displayed compact metric packet is available.

Read the total-bounded surface next. The row T supplies the finite-net budget consumed by the criterion, and the theorem Definition 3846.3 forces that budget to pass through the same compact metric source before any readback rows are inspected.

Read the Arzela–Ascoli finite-modulus route as the equicontinuity-facing control. The sibling route of Definition 3846.16 binds the Arzela–Ascoli finite-modulus chapter to the same finite compactness route, so the equicontinuity row cannot be replaced by an ambient function-space compactness theorem.

Finally the local transport, replay, provenance, and naming rows H, A, P, N keep those sibling dependencies on the same packet. Thus the Frechet–Kolmogorov chapter records a dependency lattice over compact metric, total boundedness, and Arzela–Ascoli finite modulus rather than a kernel-only compactness assertion. □

Theorem 3846.19 (Frechet–Kolmogorov orphan compact metric lattice)**.** *The compactness-facing export of an accepted* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *packet is not a kernel-only assertion. Before it exposes the local compactness criterion, the consumer must read the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *control row, the Arzela–Ascoli finite-modulus route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row as sibling dependencies of the same carrier.*

Proof. Start with the dependency witness of Definition 3846.18. It already forces the compact metric, total-bounded, and Arzela–Ascoli finite-modulus rows to be displayed with the local carrier rows.

Now read the sibling route of Definition 3846.16. That route places the uniform-equicontinuity, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows in the same finite lattice rather than letting the criterion stand as an isolated kernel compactness claim.

Finally apply the total-bounded handoff of Definition 3846.3. The finite-net budget must pass through the compact metric source before readback rows are inspected, so the local compactness criterion is exposed only after the compact-metric dependency lattice is visible. □

The FrechetKolmogorovCompactness$^{\uparrow}$ packet consumes the compact-metric total-bounded route from chapter 129 and the finite extraction route of chapter 412 before it can export a compactness criterion. Thus the compact metric source, finite extraction surface, transport rows, replay rows, and local NameCert data are visible as horizontal dependencies rather than as an isolated compactness kernel. The FrechetKolmogorovCompactness$^{\uparrow}$ packet consumes the finite total-bounded and completion rows of chapter 129 before it exports compactness data to the equicontinuity and finite-subfamily surface of chapter 412. This route keeps compact metric source data, finite nets, completion readiness, Arzela–Ascoli family control, transport, replay, provenance, and local NameCert rows visible together, rather than treating Frechet–Kolmogorov compactness as an isolated compactness oracle.

Theorem 3846.20 (Frechet–Kolmogorov top sibling pointer)**.** *Every finite-window criterion exported by* FrechetKolmogorovCompactness$^{\uparrow}$ *must first display the compact metric source of chapter 129, the total-bounded finite-net surface of chapter 127, and the Arzela–Ascoli finite-modulus route of chapter 3568. The top-level route therefore exposes the sibling lattice before any Frechet–Kolmogorov compactness consumer reads the local carrier.*

Theorem 3846.21 (Frechet–Kolmogorov tightness dependency route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read must expose the compact metric source of chapter 129, the total-bounded finite-net surface of chapter 127, the Arzela–Ascoli finite-modulus route of chapter 3568, and the locally-convex boundedness and tightness rows before the local carrier is consumed. The dependency route therefore records Frechet–Kolmogorov compactness as a finite lattice read over compact metric, total boundedness, equicontinuity-modulus, and tightness data, not as an ambient compactness oracle.*

Theorem 3846.22 (Frechet–Kolmogorov tightness route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read must display the* CompactMetric$^{\uparrow}$ *source row, the* TotallyBounded$^{\uparrow}$ *finite-net row, the* UniformEquicontinuityModulus$^{\uparrow}$ *control row, the* CompleteMetric$^{\uparrow}$ *boundary row, the* RegSeqRat$^{\uparrow}$ *rational readback row, the* Real$^{\uparrow}$ *seal, and the local tightness and boundedness rows before it exports compactness of the family. The route is a finite* BHist/Cont/Pkg *ledger over compact metric, total-bounded, equicontinuity, completion, regular-rational, real, tightness, boundedness, transport, replay, provenance, and local* NameCert *data; it is not an ambient compactness oracle, a selected subsequence, a selected subcover, or quotient completion equality.*

Proof. Read the carrier of Definition 3846.1. Its source rows already expose CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, UniformEquicontinuityModulus$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ as the visible compactness route.

Now combine the total-bounded handoff of Definition 3846.3 with the sibling route of Definition 3846.4. Those results force the finite-net, equicontinuity, and completion-facing rows to remain visible before any compactness export is read.

The tightness and boundedness rows are therefore local dependency rows in the same ledger. They do not introduce a hidden subsequence choice, a hidden subcover, or an equality in a quotient completion, because the structural rows transport and replay only the displayed finite packet. □

The compactness export of FrechetKolmogorovCompactness$^{\uparrow}$ is read through a compact-metric finite-net source together with the Arzela–Ascoli equicontinuity and tightness rows. The compact metric source fixes the finite-net budget; the Arzela–Ascoli sibling supplies the function-family tightness surface; the local replay rows then expose compactness only after both pieces have been displayed.

Theorem 3846.23 (Frechet–Kolmogorov compact-metric tightness route)**.** *Every accepted* FrechetKolmogorovCom *consumer that exports compactness for function-family windows must first read the compact-metric finite-net data of chapter 129 and the equicontinuity/tightness rows of chapter 412. The export is therefore a finite dependency route through compact metric source rows, finite-net coverage rows, Arzela–Ascoli tightness rows, transport, replay, provenance, and local* NameCert *data, not a selected subsequence, open-cover compactness theorem, quotient function-family equality, or ambient completeness principle.*

Proof. Project the Frechet–Kolmogorov carrier through its compactness-facing rows. The compact metric source supplies the finite-net data that any downstream compactness read must inspect before it can speak about a family window.

Next read the Arzela–Ascoli sibling route. Its equicontinuity and tightness rows are consumed after the compact-metric finite nets and before the local compactness export, so the function-family conclusion remains tied to displayed finite data.

Finally project the structural rows. Componentwise hsame transport, displayed Cont replay, provenance, and local naming preserve exactly the compact-metric and tightness route, leaving no coordinate for a hidden subsequence, open-cover theorem, quotient equality of families, or ambient completeness principle. □

Theorem 3846.24 (Frechet–Kolmogorov Lp-tail dependency route)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *consumer that reads compactness through Lp tails must first pass through the displayed sibling route*

CompactMetric$^{\uparrow}$, TotallyBounded$^{\uparrow}$, MetricCompletion$^{\uparrow}$, LpSpace/LpTail, BHist, hsame, Cont, Pkg, N

The exported compactness row is therefore a finite compact-metric, total-bounded, completion-facing, and Lp-tail dependency route, not an ambient compactness oracle, a selected subsequence, a selected subcover, or quotient completion equality.

Proof. Project the accepted carrier of Definition 3846.1. Its compactness-facing read already begins with the compact metric source and the finite-net budget, so any Lp-tail consumer must inherit those rows before reading a tail estimate.

Apply the total-bounded handoff of Definition 3846.3. It keeps the finite-net row visible before the StreamName, RegSeqRat, and Real readbacks are inspected.

Now use the completion boundary and the existing sibling route of Definition 3846.4. The Lp-space and Lp-tail rows can only be attached after the displayed MetricCompletion$^{\uparrow}$ handoff has been routed through the same finite BHist/Cont/Pkg ledger.

Route exhaustion gives the stated dependency chain. No row in the carrier supplies an unrecorded compactness principle, selected subsequence, selected subcover, or quotient-completion equality, so the Lp-tail compactness export is exactly the displayed sibling route. □

Theorem 3846.25 (Frechet–Kolmogorov Lp-tail localization)**.** *Every* FrechetKolmogorovCompactness$^{\uparrow}$ *compactness read that localizes through an Lp-tail estimate consumes that estimate only after the compact-metric total-bounded route, tightness route, and Lp-tail dependency route have been displayed. Equivalently, the localized read factors through Definition 3846.11, Definition 3846.22, and Definition 3846.24; it does not introduce an ambient compactness oracle, selected subsequence, selected subcover, quotient completion equality, or hidden Lp-tail selector.*

Proof. Start with the compact-metric total-bounded route. It keeps the compact metric source and finite-net surface visible before a compactness-facing consumer may inspect downstream rows.

Next use the tightness route. Tightness and boundedness remain local dependency rows in the same finite ledger, so a tail estimate cannot bypass the displayed $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, completion, readback, transport, replay, provenance, and local naming route.

Finally apply the Lp-tail dependency route. It attaches the LpSpace/LpTail row only after the compact-metric, total-bounded, and completion-facing rows have been routed through the same finite BHist/Cont/Pkg ledger. Thus Lp-tail localization is a local factorization through the existing sibling routes, not a new compactness principle or choice-selected tail object. □

The operator-tail compactness route for $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ is read only after a bounded-linear-operator source from chapter 1644, an equicontinuous-sequence surface from chapter 4356, and the existing compact-metric, total-bounded, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route have been displayed. A consumer that starts from operator-tail compactness therefore reaches the Frechet–Kolmogorov packet through bounded operator rows, equicontinuity rows, compact-metric finite nets, total-bounded coverage, regular-rational readback, and the terminal Real seal, not through an ambient compactness theorem or a selected subsequence.

Theorem 3846.26 (Frechet–Kolmogorov upstream dependency lattice)**.** *Every* $\mathsf{FrechetKolmogorovCompactness}^{\uparrow}$ *compactness read must pass through the compact metric, totally bounded,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, Arzela–Ascoli finite-modulus, and Lp-tail localization rows before any compactness conclusion is exported. The exported route is therefore a finite dependency lattice over the displayed carrier rows, not an ambient compactness oracle, selected subsequence, selected subcover, quotient completion equality, or host completeness principle.*

Proof. Start with the local carrier and the compact metric total-bounded dependency of Definition 3846.11. It forces the compact metric source and total-bounded finite-net budget to appear before later StreamName, RegSeqRat, and Real readbacks are inspected.

Next read the Arzela–Ascoli side through Definition 3846.8 and Definition 3846.9. Those routes keep finite-modulus and finite-window function-family data downstream of the compact metric and total-bounded rows.

Now apply the completion and rational-readback routes, Definition 3846.13 and Definition 3846.16. They keep the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows attached to the same displayed finite lattice.

Finally use the Lp-tail routes, Definition 3846.24 and Definition 3846.25. These place tail localization after compact-metric, total-bounded, completion-facing, and rational-readback rows. The stated lattice follows because no cited carrier row supplies a hidden compactness principle or selected limiting object. □

Chapter 3847

A Concrete Naming Certificate for $\mathsf{RationalBracketing}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalBracketing}^{\uparrow}$ packet records the finite rational lower-and-upper bracket surface used before a real-name consumer reads an approximation. It sits after the rational density surface of chapter 1409, the Archimedean bound surface of chapter 3372, the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the displayed bracket endpoints, tolerance rows, finite observation windows, nesting ledger, and downstream regular-Cauchy handoff; it is not a trichotomy principle, quotient real equality theorem, limit selector, or ambient completeness assertion.

Definition 3847.1 (Rational bracketing carrier)**.** *A* $\mathsf{RationalBracketing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, D, S, R, E, N, H, C, P, A)$$

with the following displayed rows:

1. *L is the rational lower-endpoint row supplied through the* $\mathsf{RationalDensityReal}^{\uparrow}$ *surface.*
2. *U is the rational upper-endpoint row supplied through the same rational-density surface.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the displayed bracket width with the requested precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the observations used to inspect L, U, and D.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row exported by the bracket window.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, D, S, R have been displayed.*
7. *N is the nesting ledger recording that later bracket windows refine earlier ones by moving the lower row upward, moving the upper row downward, or preserving an endpoint already inside the requested tolerance.*
8. *H records componentwise* hsame *transport for lower endpoint, upper endpoint, dyadic tolerance, stream window, regular readback, real seal, nesting, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bracketing request through $L, U, D, S, R, E, N, H, P, A$.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalBracketing}^{\uparrow}$, $\mathsf{RationalDensityReal}^{\uparrow}$, $\mathsf{RealArchimedeanProperty}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *A is the local* $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $L, U, D, S, R, E, N, H, C, P, A$. It has no coordinate for trichotomy, quotient real equality, selected limits, hidden real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalBracketingUp.RationalBracketingTasteGate single carrier alignment`

Theorem 3847.2 (Rational bracketing NameCert obligations)**.** *For every accepted* $\mathsf{RationalBracketing}^{\uparrow}$ *carrier $B = (L, U, D, S, R, E, N, H, C, P, A)$, the local* $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$ *surface consists exactly of lower-endpoint exposure through L, upper-endpoint exposure through U, dyadic width and tolerance exactness through D, finite-window exposure through S, regular rational readback through R, terminal real-name sealing through E, nesting-ledger visibility through N, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through A.*

Proof. Project the carrier of Definition 3847.1. Its visible approximation data are exactly the endpoint rows L, U, the tolerance row D, the finite-window row S, the readback row R, and the real seal row E.

The endpoint and tolerance rows are separate. The rational-density surface supplies L and U, while the Archimedean and dyadic rows supply the bounded comparison budget used to certify that the bracket width is below the requested tolerance.

The nesting ledger is downstream of those rows. It records refinement between displayed windows, but it does not add a new real representative, a trichotomy split, or a selected limiting endpoint.

Finally H, C, P, A transport, replay, source, and name the same finite surface. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3847.3 (Rational bracketing window nesting)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalBracketing}^{\uparrow}$ *packet factors through a finite nested-bracket route*

$$\mathsf{RationalDensityReal}^{\uparrow} \longmapsto \mathsf{RealArchimedeanProperty}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Re}$$

For any later accepted window, the consumer must read the same lower endpoint, upper endpoint, dyadic tolerance, stream observation, regular readback, real seal, nesting, transport, replay, provenance, and local naming rows before using the narrowed bracket. The route exports nested rational brackets, not trichotomy, quotient real equality, selected limits, or ambient completeness.

Proof. Begin with an accepted carrier $B = (L, U, D, S, R, E, N, H, C, P, A)$. A bracketing request first reads the rational lower and upper endpoint rows L, U together with the Archimedean bound information named in the provenance row.

Next the request reads D and S. The dyadic ledger specifies the precision budget, while the stream row records the finite observations used to inspect the current bracket.

The regular rational readback R and real seal E are available only after those rows have been displayed. A later window can be consumed only through the nesting ledger N, which compares the later endpoints with the earlier displayed endpoints under the same dyadic budget discipline.

Thus every downstream read remains inside the finite route named in the theorem. No step introduces trichotomy, quotient real equality, a selected limiting endpoint, countable choice, hidden real representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{RationalBracketing}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without first reading rational lower and upper endpoint rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{StreamName}^{\uparrow}$ finite window, nesting evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence is outside the packet.

Independence witness: rational-density-real, real-archimedean-property, dyadicratcore, streamname, regseqrat, real: $\mathsf{RationalBracketing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries rational lower and upper endpoint rows, dyadic width budgets, finite observation windows, regular readback, terminal Real sealing, nesting, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalBracketing}^{\uparrow}$ packet is generated from rational lower and upper endpoint rows, an Archimedean boundedness context, a DyadicRatCore tolerance ledger, StreamName finite windows, a RegSeqRat readback row, a RealUp seal, a nesting ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3847.1, Definition 3847.2, and Definition 3847.3 seed the finite rational-bracket route from density and Archimedean rows to regular rational readback and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalBracketing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, trichotomy, quotient real equality, selected limits, hidden real representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked lower and upper endpoint exposure, dyadic width exactness, finite-window exposure, nesting stability, regular readback, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.RationalBracketingUp.

Chapter 3848

A Concrete Naming Certificate for $\mathsf{RealModulusExtraction}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy data, the $\mathsf{RealModulusExtraction}^{\uparrow}$ packet records the finite BEDC surface by which an explicit regular Cauchy modulus is exposed before a $\mathsf{Real}^{\uparrow}$ seal is read. It sits on the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names modulus extraction as displayed finite NameCert data; it does not import quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3848.1 (RealModulusExtraction carrier)**.** *A* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic radius, denominator budget, and comparison ledger for each requested precision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite windows inspected before a modulus answer is emitted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *witness row, recording regular rational Cauchy data exposed by the scheduled windows.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, R have been displayed.*
5. *T records componentwise* hsame *transport for the dyadic tolerance, schedule, regularity, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a precision request through D, then through S, R, and finally to the seal row E.*
7. *P is the* Pkg *provenance over* $\mathsf{RealModulusExtraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$ *row.*

The carrier has no coordinate for an implicit analytic modulus, quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealModulusExtractionUp.RealModulusExtractionTasteGate single carrier alignment`

Theorem 3848.2 (RealModulusExtraction NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier* $M = (D, S, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$ *surface consists exactly of dyadic tolerance exposure through* D*, finite schedule exposure through* S*, regular Cauchy witness exposure through* R*, terminal Real sealing through* E*, componentwise transport through* T*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier coordinates. The dyadic tolerance row D, schedule row S, regular witness row R, and seal row E are the analytic-facing rows of the packet.

The structural rows do not add analytic assumptions. Transport through T is componentwise, replay through C follows the displayed precision route, provenance through P records the sibling chapters named above, and N supplies the local NameCert row. Since no further coordinate is present, the listed rows exhaust the obligation surface. □

Theorem 3848.3 (RealModulusExtraction regular Cauchy handoff)**.** *Every regular Cauchy handoff exported by an accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The extracted modulus is therefore read as dyadic tolerance data, finite schedule data, and regular rational witness data before the Real seal is available.

Proof. Begin with a precision request. The carrier first exposes the dyadic tolerance row D, so the requested radius and denominator budget are displayed before any stream window is inspected.

The schedule row S then supplies the finite windows, and the regular witness row R reads the rational Cauchy data inside those windows. Only after this displayed route has been replayed by C does the seal row E become available to a $\mathsf{Real}^{\uparrow}$ consumer.

Thus the handoff is the finite sibling route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, not an implicit analytic assumption. □

Theorem 3848.4 (RealModulusExtraction Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealModulusExtraction}^{\uparrow}$ *carrier is exhausted by the rows* D, S, R, E, T, C, P, N*. In particular, the read exports no quotient stream equality, selected representative, countable-choice schedule, host real equality, implicit analytic modulus, or ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Use the obligation surface of Definition 3848.2. A Real-facing consumer can inspect only the dyadic tolerance, finite schedule, regular witness, seal, transport, replay, provenance, and local naming rows.

The nonescape claim follows from the carrier boundary. None of the excluded objects appears as a coordinate of M, and the replay row C only preserves the displayed precision route through D, S, R, E. Hence every Real seal read remains inside the finite NameCert packet. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusExtraction}^{\uparrow}$ read exposes dyadic tolerance, finite schedule, regular Cauchy witness, Real seal, transport, replay, provenance, and local NameCert rows before any modulus answer is exported.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RealModulusExtraction}^{\uparrow}$ is not bijective to any listed sibling because it carries the dyadic tolerance row, schedule row, regular

witness row, terminal seal row, componentwise transport, displayed replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusExtraction}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, StreamName schedule row, RegSeqRat regular witness row, RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3848.1, Definition 3848.2, Definition 3848.3, and Definition 3848.4 seed the finite modulus-extraction route from regular Cauchy data to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic schedule coverage, regular Cauchy handoff, Real seal nonescape, and a Lean carrier formalization for BEDC.Derived.RealModulusExtractionUp.

Chapter 3849

A Concrete Naming Certificate for $\mathsf{CauchyStability}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy data, the $\mathsf{CauchyStability}^{\uparrow}$ packet names the finite stability route used when displayed transport changes a Cauchy presentation but not the Real-facing read. It sits on the StreamName windows of chapter 126, the dyadic tolerance core of chapter 291, the regular rational sequence route of chapter 301, the Cauchy equivalence boundary of chapter 1820, and the terminal real-name seal of chapter 38. The packet records stability of displayed Cauchy data; it is not quotient equality, selected representatives, countable choice over tails, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3849.1 (CauchyStability carrier)**.** *A* $\mathsf{CauchyStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are two regular Cauchy source rows whose rational observations are opened only through* $\mathsf{RegSeqRat}^{\uparrow}$ *and finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *M is the shared tail-modulus comparison row relating the two source rows at each displayed precision request.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the transported source observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row recording the regular rational readback after A, B, M, D have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached through the displayed* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary.*
6. *H records componentwise* hsame *transport for the source, modulus, dyadic, regular-readback, equivalence, seal, replay, package, and naming rows.*
7. *C records displayed* Cont *replay from a Cauchy-stability request through $A, B, M, D, R, E, H, P, N$.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyStability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient sequence equality, selected limits, hidden representatives, countable-choice tail selection, host real equality, or an ambient completeness theorem.

Lean4 (definition) `BEDC.Derived.CauchyStabilityUp.CauchyStabilityUp`
Lean4 ✓ `BEDC.Derived.CauchyStabilityUp.CauchyStabilityTasteGate single carrier alignment`

Theorem 3849.2 (CauchyStability NameCert obligations)**.** *For every accepted* $\mathsf{CauchyStability}^{\uparrow}$ *carrier* $S = (A, B, M, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the two regular Cauchy source rows, shared tail-modulus comparison, dyadic tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *stability of the transported readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$*, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming through the rows* $A, B, M, D, R, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 3849.1. The only source data are the rows A and B, and the only comparison data between them are the shared tail-modulus row M and dyadic tolerance ledger D.

The regular-rational readback R is downstream from those rows, so stability is read as a finite $\mathsf{RegSeqRat}^{\uparrow}$ handoff rather than as equality of host sequences. The displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ boundary is consumed before the terminal seal E can be read.

Transport and replay use H and C componentwise. The package and naming rows P, N source the same finite route, and the excluded quotient, selected-limit, hidden-representative, choice, host-equality, and completeness data have no carrier coordinate. □

Theorem 3849.3 (CauchyStability Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyStability}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before the terminal seal E *is consumed. Thus displayed transport may change the finite Cauchy presentation only through the stored* $A, B, M, D, R, H, C, P, N$ *rows, and the Real-facing boundary receives no quotient equality, selected representative, hidden tail family, countable-choice selector, host real equality, or ambient completeness coordinate.*

Proof. Start with a Real-facing request. The carrier first exposes the finite StreamName windows attached to A and B, together with the shared modulus comparison M.

The dyadic ledger D translates those windows into displayed tolerance observations. The regular-rational handoff R then repacks the inspected observations as $\mathsf{RegSeqRat}^{\uparrow}$ data.

Only after this readback does the packet enter the displayed $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ boundary and then the seal E. The rows H, C, P, N transport, replay, source, and name that same route, leaving no hidden Real-facing coordinate beyond the listed sibling surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyStability}^{\uparrow}$ packet is generated from two regular Cauchy source rows, a shared tail-modulus comparison row, dyadic tolerance rows, a RegSeqRat handoff, a CauchyEquivalenceSetoid boundary, a RealUp seal, displayed hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$, seed)

Scope closed: Definition 3849.1, Definition 3849.2, and Definition 3849.3 seed the displayed stability route for Cauchy data under transport before the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyStability}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient equality of Cauchy presentations, selected representatives, hidden tails, countable choice over moduli, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-modulus comparison, RegSeqRat stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyStabilityUp.

Chapter 3850

A Concrete Naming Certificate for $\mathsf{CauchySequenceTailFilter}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki–Kelley filterbase terminology, the $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ packet records the finite route by which a Bishop Cauchy sequence supplies tail-filter data for completion-facing consumers. It sits between the sequence-side evidence of $\mathsf{BishopCauchySequence}^{\uparrow}$, the filter carrier of chapter 333, the finite stream schedule of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance ledger of chapter 291, and the $\mathsf{Real}^{\uparrow}$ seal boundary of chapter 38. The packet names tail neighborhoods generated by displayed sequence windows; it does not select a limit, quotient streams, choose subsequences, decide trichotomy, or assert ambient real completeness.

Definition 3850.1 (Cauchy sequence tail-filter carrier)**.** *A* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, B, F, C, D, Q, R, E, H, K, P, N)$$

with the following visible rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose finite windows are the only sequence observations visible to the carrier.*
2. *B is the* $\mathsf{BishopCauchySequence}^{\uparrow}$ *row certifying that the displayed sequence windows satisfy the Bishop Cauchy modulus requests read by the packet.*
3. *F is the Frechet-tail filterbase row generated by tails of S: a basic tail is named by a displayed index threshold together with the windows beyond that threshold.*
4. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row saying that every admitted tail-basic row is a Cauchy-filter neighborhood request over the same displayed windows and thresholds.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording dyadic scales, rational comparisons, and exhaustion data used to pass from sequence tails to filterbase requests.*
6. *Q is the finite tail-query ledger pairing each tolerance request with the stream windows and tail threshold used by B, F, C, D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for consumers that read the same tail evidence as regular rational names.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, B, F, C, D, Q, R have been displayed.*

9. *H stores componentwise* hsame *transport for the stream windows, Bishop Cauchy row, tail-filterbase row, Cauchy-filter compatibility row, dyadic tolerance ledger, tail-query ledger, regular handoff, Real seal, continuation replay, provenance, and local name.*
10. *K stores displayed* Cont *replay from each tolerance request through the stream window, tail threshold, filterbase row, Cauchy-filter compatibility row, regular handoff, and seal row.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{BishopCauchySequence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}$ *row.*

The classifier accepts a packet only when every tail-basic row in F is generated from the displayed stream schedule S, every Cauchy-filter compatibility read in C uses the Bishop Cauchy evidence B and dyadic ledger D, and every completion-facing read factors through Q, R, E, H, K, P, N. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterUp.CauchySequenceTailFilterTasteGate single carrier alignment`

Theorem 3850.2 (Cauchy sequence tail-filter NameCert obligations)**.** *For every accepted* CauchySequenceTailFilter *carrier $T = (S, B, F, C, D, Q, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}$ *obligations are exactly carrier admission, finite stream-window classification, Bishop Cauchy tail coverage, Frechet-tail filterbase generation,* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility, dyadic tolerance-ledger exhaustion,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal nonescape,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local naming row. These obligations are discharged by the displayed route*

$$S \longmapsto B \longmapsto F \longmapsto C \longmapsto D \longmapsto Q \longmapsto R \longmapsto E$$

together with H, K, P, N.

The obligation surface exports no quotient stream, selected limit, choice principle, trichotomy row, open-cover compactness object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterUp.CauchySequenceTailFilterCarrier namecert obligations`

Proof. Project the carrier of Definition 3850.1. The only sequence observations are the finite $\mathsf{StreamName}^{\uparrow}$ windows in S, and the only Cauchy evidence for those windows is the Bishop row B.

Read the tail-filter rows next. The Frechet-tail row F names basic tails by displayed thresholds, while the compatibility row C checks those basic tails against the $\mathsf{CauchyFilter}^{\uparrow}$ surface. The dyadic ledger D and tail-query ledger Q account for the tolerance requests used in that check.

The regular and real-facing rows are downstream. The $\mathsf{RegSeqRat}^{\uparrow}$ handoff R and $\mathsf{Real}^{\uparrow}$ seal E are available only after the stream, Bishop, tail-filterbase, Cauchy-filter, dyadic, and query rows have been displayed.

Finally apply the structural rows. Componentwise transport H, continuation replay K, provenance P, and the local name N preserve the same finite route, and the carrier definition supplies no coordinate for quotient streams, selected limits, choice, trichotomy, open-cover compactness, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ completion-facing read factors through the displayed $\mathsf{StreamName}^{\uparrow}$ windows, Bishop Cauchy row, Frechet-tail filterbase row,

$\mathsf{CauchyFilter}^{\uparrow}$ compatibility row, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, and $\mathsf{Real}^{\uparrow}$ seal before any tail consumer is allowed to read the packet.

Independence witness: cauchyfilter, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries sequence-window evidence, Bishop Cauchy tail coverage, tail-filterbase generation, filter compatibility, dyadic tolerance accounting, regular-rational handoff, real-seal routing, transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ packet is generated from finite stream windows, a Bishop Cauchy sequence row, a Frechet-tail filterbase row, a CauchyFilter compatibility row, a dyadic tolerance ledger, a tail-query ledger, a RegSeqRat handoff, a Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3850.1 and Definition 3850.2 seed the finite sequence-to-tail-filter carrier from StreamName windows through Bishop Cauchy evidence, tail filterbase rows, CauchyFilter compatibility, dyadic tolerance accounting, RegSeqRat handoff, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, choice principles, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, tail-filterbase generation, CauchyFilter compatibility, dyadic ledger exhaustion, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local NameCert rows for BEDC.Derived.CauchySequenceTailFilterUp.

Chapter 3851

A Concrete Naming Certificate for $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Bourbaki–Kelley filter terminology, the CauchySequenceTailFilte packet records the finite bridge by which sequence-tail filters are read as Cauchy-filter completion data before any quotient completion point or selected limit is available. It sits after the sequence-tail route of chapter 3850, the completion-facing carrier of chapter 1569, and the saturated filter surface of chapter 333. The chapter names only the displayed finite handoff from tail-filter evidence to completion readback; it does not assert ambient real completeness, quotient-filter equality, selected representatives, countable choice, trichotomy, or host equality of completed points.

Definition 3851.1 (Cauchy sequence tail-filter completion carrier)**.** *A* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, F, C, D, W, R, E, H, K, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ *row naming the sequence-generated tail-filter data from chapter 3850.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row that reads the same tail-filter neighborhoods as saturated Cauchy-filter requests.*
3. *C is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row whose completion-facing consumer is allowed to inspect only the displayed filter route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the tail thresholds with completion precision requests.*
5. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows exposed by the tail-filter row before the completion consumer reads R or E.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same windows, dyadic tolerances, and filter requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after T, F, C, D, W, R have been displayed.*
8. *H stores componentwise* hsame *transport for the tail-filter, Cauchy-filter, completion, dyadic, window, readback, seal, replay, provenance, and local naming rows.*

9. *K stores displayed* Cont *replay from a sequence-tail request through Cauchy-filter compatibility, completion handoff, finite windows, dyadic tolerance, regular readback, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$ *row.*

The classifier accepts X only when every completion-facing read first opens the sequence-tail row T, then the Cauchy-filter row F, then the completion row C, and only then the dyadic, window, readback, and Real-seal rows D, W, R, E. It has no coordinate for quotient filters, selected limits, countable-choice selectors, host real equality, trichotomy, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchySequenceTailFilterCompletionUp.CauchySequenceTailFilterCompletionTasteGate`
`single carrier alignment`

Theorem 3851.2 (Cauchy sequence tail-filter completion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *carrier $X = (T, F, C, D, W, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$ *obligations are exactly carrier admission, sequence-tail source admission,* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *handoff, dyadic tolerance exactness, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local naming row. These obligations are discharged by the displayed route*

$$T \longmapsto F \longmapsto C \longmapsto D, W \longmapsto R \longmapsto E$$

together with H, K, P, N.

The obligation surface exports no quotient filter, selected limit, host equality, countable-choice selector, trichotomy row, open-cover compactness object, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or detached completion coordinate.*

Proof. Project the carrier of Definition 3851.1. The public rows are exactly $T, F, C, D, W, R, E, H, K, P, N$, so carrier admission exposes a sequence-tail source, a Cauchy-filter compatibility row, a completion handoff, a finite tolerance and window route, a regular readback, a Real seal, and the structural rows.

Read the source and filter rows first. The row T supplies the finite tail-filter data generated from sequence windows, while F reads that data as the Cauchy-filter surface used by completion-facing consumers. The row C is downstream of those two rows and does not replace them by a quotient filter or selected limit.

Next inspect the completion readback route. The dyadic ledger D and window row W are the only precision and observation rows available before the regular readback R and the terminal seal E are consumed.

Finally apply the structural rows. Transport H, replay K, provenance P, and the local name N preserve the same route, and the carrier definition supplies no coordinate for quotient filters, selected limits, choice, trichotomy, open-cover compactness, host equality, or ambient completeness. □

Theorem 3851.3 (Cauchy sequence tail-filter completion Real handoff)**.** *Let $X = (T, F, C, D, W, R, E, H, K, P, N)$ be an accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *packet. Every Real-facing or completion-facing consumer that reads the terminal seal factors through*

$$T, F, C, D, W, R \longmapsto E \longmapsto H, K, P, N.$$

Thus the $\mathsf{Real}^{\uparrow}$ *seal is available only after the sequence-tail filter, Cauchy-filter compatibility, Cauchy-filter completion row, dyadic tolerance ledger, finite stream windows, and regular rational readback have been displayed.*

Proof. Start with a consumer of the terminal seal. Carrier projection gives access to E only as the seal row of the accepted packet X.

Replay the continuation row K. The request consumes the sequence-tail source T, the Cauchy-filter row F, the completion row C, the dyadic tolerance ledger D, the finite windows W, and the regular readback R before E is available.

The structural rows preserve this order. Transport through H, provenance through P, and naming through N record the same route, so the consumer has no available coordinate for a selected completion point, quotient-filter equality, host real equality, choice, or ambient completeness. □

Theorem 3851.4 (Cauchy sequence tail-filter completion filterbase nonescape)**.** *Let* $X = (T, F, C, D, W, R, E, H, K$ *be an accepted* $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ *packet. No accepted completion read can bypass the displayed sequence-tail filter row* T *or the displayed* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *when it reaches the completion row* C*, the regular readback* R*, or the seal row* E*.*

Consequently every filterbase-dependent completion read is confined to the finite route

$$T_{\mathsf{CauchySequenceTailFilter}^{\uparrow}}, F_{\mathsf{CauchyFilter}^{\uparrow}} \longmapsto C_{\mathsf{CauchyFilterCompletion}^{\uparrow}} \longmapsto D, W, R, E, H, K, P, N.$$

Detached filterbases, quotient filters, selected limits, free tolerance parameters, countable-choice selectors, and ambient completeness principles are outside the accepted packet.

Proof. Inspect the carrier of Definition 3851.1. The only source for filterbase-dependent data is T, and the only Cauchy-filter compatibility row is F.

Apply the obligation surface of Definition 3851.2. It requires the completion handoff C to read after T and F, then requires D, W, R, E to remain downstream of that handoff.

The Real handoff of Definition 3851.3 keeps terminal consumers on the same route. Since H, K, P, N only transport, replay, source, and name the displayed rows, no detached filterbase or excluded completion object can be introduced later. □

Falsifiable prediction: Every accepted $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ completion read opens the displayed $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ row, $\mathsf{CauchyFilter}^{\uparrow}$ row, $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row, dyadic tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal before any consumer may inspect the packet as completion data.

Independence witness: cauchy-sequence-tail-filter, cauchyfilter, cauchyfiltercompletion, dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries sequence-tail source data, Cauchy-filter compatibility, completion handoff, dyadic tolerance accounting, finite stream windows, regular readback, Real-seal routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}$ packet is generated from a CauchySequenceTailFilter row, a CauchyFilter row, a CauchyFilterCompletion row, dyadic tolerance data, finite StreamName windows, RegSeqRat readback, a Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3851.1, Definition 3851.2, Definition 3851.3, and Definition 3851.4 seed the finite sequence-tail-filter bridge into CauchyFilterCompletion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceTailFilterCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CauchySequenceTailFilterCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, selected completion representatives, countable choice, trichotomy, open-cover compactness, host equality, ambient Real$^{\uparrow}$ completeness, or theorem-level Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, sequence-tail source admission, CauchyFilter compatibility, CauchyFilterCompletion handoff, dyadic tolerance exactness, StreamName window exposure, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local NameCert rows for BEDC.Derived.CauchySequenceTailFilterCompletion

Chapter 3852

A Concrete Naming Certificate for $\mathsf{LocalizedCauchyRate}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocalizedCauchyRate}^{\uparrow}$ packet records localized convergence-rate evidence as finite BEDC rows before a Real or Completion consumer reads a regular Cauchy name. It sits over the tail-bound route of chapter 2071, the regular tail-bound route of chapter 3743, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the oscillation ledger of chapter 448. The packet is a localized rate ledger; it does not choose quotient stream representatives, selected limits, trichotomy decisions, ambient completeness, or global convergence-speed functions.

Definition 3852.1 (Localized Cauchy rate carrier)**.** *A* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, D, T, R, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected rational stream segment.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the local dyadic budgets used by the packet.*
3. *T is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row displaying the tail-bound witness for the requested local tolerance window.*
4. *R is the* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *row recording the regular tail-bound strengthening consumed by regular Cauchy readers.*
5. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, tying the window and tail-bound rows to regular rational sequence data.*
6. *Q is the* $\mathsf{CauchyOscillation}^{\uparrow}$ *row, recording the local oscillation bound checked against the same window and dyadic budget.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after W, D, T, R, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for windows, tolerances, tail bounds, regular readback, oscillation, real seal, replay, provenance, and naming.*
9. *C records displayed* Cont *replay from a localized rate request through $W, D, T, R, S, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{LocalizedCauchyRate}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyOscillation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, D, T, R, S, Q, E, H, C, P, N$. The carrier has no coordinate for a selected rate outside T and R, quotient stream equality, an external limit selector, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocalizedCauchyRateUp.LocalizedCauchyRateTasteGate single carrier alignment`

Theorem 3852.2 (Localized Cauchy rate NameCert obligations)**.** *For every accepted* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *carrier $L = (W, D, T, R, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $W, D, T, R, S, Q, E, H, C, P, N$;*
2. *window localization: every rate read starts with the finite* $\mathsf{StreamName}^{\uparrow}$ *row W and the dyadic tolerance ledger D;*
3. *tail-bound visibility: the* $\mathsf{CauchyTailBound}^{\uparrow}$ *row T and* $\mathsf{RegularCauchyTailBound}^{\uparrow}$ *row R are displayed before any regular rational readback is consumed;*
4. *regular readback: the* $\mathsf{RegSeqRat}^{\uparrow}$ *row S reads the same window, dyadic budget, and tail-bound rows;*
5. *oscillation compatibility: the* $\mathsf{CauchyOscillation}^{\uparrow}$ *row Q checks the local oscillation bound against the same budget and tail window;*
6. *real handoff: the* $\mathsf{Real}^{\uparrow}$ *seal E is reachable only after W, D, T, R, S, Q have been displayed;*
7. *ledger nonescape: H, C, P, N export no selected rate, quotient representative, host limit, trichotomy decision, countable-choice schedule, ambient completeness theorem, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 3852.1. The accepted rows are exactly $W, D, T, R, S, Q, E, H, C, P, N$, so carrier admission exposes only finite windows, dyadic budgets, tail bounds, regular readback, oscillation, a real seal, and structural rows.

The localized rate route begins with W and D. These rows name the finite stream segment and the tolerance budget, so the tail-bound rows cannot be read as global convergence-speed functions.

Next read T and R. The ordinary Cauchy tail-bound row supplies the requested local tail witness, while the regular tail-bound row records the strengthening needed by a regular Cauchy consumer. The readback row S consumes those same displayed rows, and Q checks the matching oscillation bound.

Finally E, H, C, P, N seal and transport the route. Since none of these rows adds a selector, quotient representative, trichotomy decision, countable-choice schedule, ambient completeness theorem, or bridge certificate, the listed obligations exhaust the local surface. □

Theorem 3852.3 (Localized Cauchy rate Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LocalizedCauchyRate}^{\uparrow}$ *packet factors through*

$$W \longmapsto D \longmapsto T \longmapsto R \longmapsto S \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

In particular, a consumer preserving the $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RegularCauchyTailBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *routes must read the localized tail-bound ledger before it can treat the stream as a real-facing regular Cauchy name.*

Proof. Begin with a Real-facing request. The finite stream row W and dyadic ledger D determine the local tolerance window that the request may inspect.

The tail-bound handoff then passes through T and R. The first row gives the Cauchy tail witness at the displayed tolerance, and the second row exposes the regular tail-bound strengthening required by regular rational readback.

The readback and oscillation rows are downstream of the same data. The row S reads the regular rational sequence through the displayed window and tail-bound rows, while Q confirms that the local oscillation bound uses the same budget.

Only after those rows have been displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so any Real or Completion consumer bypassing T and R lies outside the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{LocalizedCauchyRate}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ seal without displaying StreamName windows, DyadicRatCore budgets, CauchyTailBound and RegularCauchyTailBound rows, RegSeqRat readback, CauchyOscillation compatibility, transport, replay, provenance, and local naming.

Independence witness: cauchy-tail-bound, regular-cauchy-tail-bound, dyadicratcore, streamname, regseqrat, real, cauchy-oscillation: none alone is bijective with $\mathsf{LocalizedCauchyRate}^{\uparrow}$ because the packet jointly records localized windows, dyadic budgets, ordinary and regular tail-bound rows, regular rational readback, oscillation compatibility, terminal Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalizedCauchyRate}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore budgets, CauchyTailBound and RegularCauchyTailBound rows, RegSeqRat readback, CauchyOscillation compatibility, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3852.1, Definition 3852.2, and Definition 3852.3 seed the finite localized convergence-rate ledger consumed by Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalizedCauchyRate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected rates, quotient stream representatives, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked window localization, tail-bound visibility, regular readback, oscillation compatibility, Real handoff, ledger nonescape, and local naming for BEDC.Derived.LocalizedCauchyRateUp.

Chapter 3853

A Concrete Naming Certificate for $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$

Following Bishop–Bridges compact metric continuity, the $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ packet records the finite route from compact source data and pointwise continuity rows through finite oscillation partitions to a displayed uniform-modulus output. It sits over the compact-metric source of chapter 129, the metric carrier of chapter 57, the continuous-map surface of chapter 130, the finite oscillation partition row of chapter 3800, the oscillation modulus boundary of chapter 2224, and the uniform-modulus consumer of chapter 369. The packet names a compact-uniform handoff; it does not select an external modulus, identify host functions, choose points from a cover, assert quotient equality, or import ambient compactness.

Definition 3853.1 (Finite oscillation uniform-modulus carrier)**.** *A* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, A, B, O, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, carrying the compact-metric carrier, finite cover data, and metric distance rows inspected by the packet.*
2. *M is the metric compatibility row tying the compact source to the distance comparisons used by oscillation estimates.*
3. *A is the continuous-map pointwise row, exposing source, target, graph, and local continuity evidence for the displayed map.*
4. *B is the* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, listing the finite partition cells and their source-side cover witnesses.*
5. *O is the* $\mathsf{OscillationModulus}^{\uparrow}$ *row, recording the oscillation bound attached to each partition cell and requested tolerance.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row obtained by folding the finite oscillation bounds into one tolerance-only precision ledger.*
7. *H records componentwise* hsame *transport for compact, metric, continuous-map, partition, oscillation, uniform-modulus, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform-continuity request through $K, M, A, B, O, U, H, P, N$.*

9. *P is the* Pkg *provenance over* FiniteOscillationUniformModulus$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteOscillationPartition$^{\uparrow}$, OscillationModulus$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteOscillationUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, A, B, O, U, H, C, P, N$. The carrier has no coordinate for an external modulus selector, host function equality, point choice from a cover, quotient representation, ambient compactness theorem, or bridge certificate.

Lean4 (definition) `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusUp`
Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusTasteGate`
`single carrier alignment`

Theorem 3853.2 (Finite oscillation uniform-modulus NameCert obligations)**.** *For every accepted* FiniteOscillationUniformModulus$^{\uparrow}$ *carrier $F = (K, M, A, B, O, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteOscillationUniformM}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $K, M, A, B, O, U, H, C, P, N$;*
2. *compact source exposure: K and M display the compact-metric source and distance comparisons used by the packet;*
3. *continuity visibility: the pointwise continuous-map row A is read before any oscillation cell is accepted;*
4. *finite partition exactness: B lists the finite oscillation partition cells and their cover witnesses;*
5. *oscillation-bound compatibility: O attaches each partition cell to the requested tolerance through the same metric and continuity rows;*
6. *uniform-modulus output: U is the only exported tolerance-only precision ledger and is obtained after B and O are displayed;*
7. *ledger nonescape: H, C, P, N export no external modulus selector, host function equality, point choice from a cover, quotient representation, ambient compactness theorem, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusNameCertObligatio`

Proof. Project the carrier in Definition 3853.1. Its accepted rows are exactly $K, M, A, B, O, U, H, C, P, N$, so carrier admission exposes compact source data, metric comparisons, continuous-map evidence, finite partition rows, oscillation bounds, a uniform-modulus output, and structural rows only.

The compact and continuity clauses are read first. The compact source row K and metric row M supply the finite cover and distance comparisons, while A supplies the pointwise continuity evidence used on each displayed cell.

The partition route then reads B and O. The finite oscillation partition row lists the cells, and the oscillation-modulus row binds each cell to the requested tolerance through the same compact, metric, and continuity data.

Finally the output row U records the uniform-modulus ledger obtained from the finite oscillation bounds. The transport, replay, provenance, and naming rows preserve that displayed route and contain no coordinate for the excluded selector, host equality, point choice, quotient representation, ambient compactness theorem, or bridge certificate. □

Lean4 ✓ `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusCarrier`
`namecert obligations`

Theorem 3853.3 (Finite oscillation compact-uniform handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ *packet factors through*

$$K \longmapsto M \longmapsto A \longmapsto B \longmapsto O \longmapsto U \longmapsto H, C, P, N.$$

In particular, a consumer preserving compact-metric, continuous-map, finite oscillation partition, oscillation modulus, and uniform-modulus provenance must read the finite oscillation partition and its oscillation bounds before exporting the tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ *row.*

Proof. Begin with a compact-uniform request. The rows K and M display the compact source and metric comparisons, so the request is tied to finite source-side data rather than to an ambient compactness theorem.

The continuous-map row A supplies the pointwise continuity evidence used on partition cells. After that, B displays the finite oscillation partition, including the cells and cover witnesses read by the source-side request.

The oscillation row O binds those cells to the requested tolerance. Since O is downstream of K, M, A, B, it cannot be read as a detached modulus or host-level continuity theorem.

The uniform-modulus row U is then the only exported tolerance-only precision ledger. The structural rows H, C, P, N transport and name the same finite route, so any compact-uniform consumer reaching U without B and O bypasses the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying compact source rows, metric comparisons, continuous-map evidence, finite oscillation partitions, oscillation-modulus bounds, transport, replay, provenance, and local naming.

Independence witness: compactmetric, metricspace, continuousmap, finite-oscillation-partition, oscillation-modulus, uniformmodulus: none alone is bijective with $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ because the packet jointly carries compact source data, metric comparisons, pointwise continuity rows, finite partition cells, oscillation bounds, uniform-modulus output, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteOscillationUniformModulus}^{\uparrow}$ packet is generated from compact-metric source rows, metric comparison rows, ContinuousMap pointwise rows, finite oscillation partition rows, oscillation-modulus bounds, a UniformModulus output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3853.1, Definition 3853.2, and Definition 3853.3 seed the finite oscillation route from compact local control to a uniform-modulus output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationUniformModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteOscillationUniformModulusUp.FiniteOscillationUniformModulusCarrier\_namecert\_obl`

Bridge status: none

Not claimed: This chapter does not close external modulus selectors, host function equality, point choice from covers, quotient representations, ambient compactness, or bridge checking.

Upgrade path: Obligation closure requires checked compact source exposure, continuity visibility, finite partition exactness, oscillation-bound compatibility, uniform-modulus output, ledger nonescape, and local naming for BEDC.Derived.FiniteOscillationUnifo

Chapter 3854

A Concrete Naming Certificate for $\mathsf{FubiniFiniteRectangle}^{\uparrow}$

Following Fubini and Bishop-style finite integration, the $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ packet records a bounded finite-product rectangle computation as BEDC data before either iterated-integral route may be read. It sits over the measure row of chapter 92, the integral row of chapter 123, the product rows of chapter 2000, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the Real seal of chapter 38. The chapter names the finite rectangle surface only; it does not assert sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, or ambient measure completeness.

Definition 3854.1 (Fubini finite rectangle carrier)**.** *A* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, M, I, P, A, B, D, S, E, H, C, Q, N)$$

with the following displayed rows:

1. *R is the finite product-rectangle row, listing the source rectangles and their cell tags.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *row assigning the displayed finite measure data used by the cells of R.*
3. *I is the* $\mathsf{Integral}^{\uparrow}$ *row for cellwise finite integration over the same rectangle table.*
4. *P is the* ***??*** *row tying the two coordinate directions to the same finite rectangle cells.*
5. *A is the first-then-second iterated-integral route, reading the first coordinate cells before folding the second coordinate.*
6. *B is the second-then-first iterated-integral route, reading the second coordinate cells before folding the first coordinate.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the common finite error budget used by both iterated routes.*
8. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the displayed finite sums.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, M, I, P, A, B, D, S have been displayed.*
10. *H records componentwise* hsame *transport for rectangle cells, measure rows, integral rows, product rows, both iterated routes, tolerance ledgers, rational readback, Real seal, replay, provenance, and naming.*

11. *C records displayed* Cont *replay from a finite-rectangle integration request through* $R, M, I, P, A, B, D, S, E, H, Q,$
12. *Q is the* Pkg *provenance over* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, *??*, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, I, P, A, B, D, S, E, H, C, Q, N$. It has no coordinate for quotient function equality, selected representatives, countable choice, ambient completion of a measure space, or an infinite summability principle.

Lean4 ✓ `BEDC.Derived.FubiniFiniteRectangleUp.FubiniFiniteRectangleTasteGate single carrier alignment`

Theorem 3854.2 (Fubini finite rectangle NameCert obligations)**.** *For every accepted* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *carrier $F = (R, M, I, P, A, B, D, S, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet $R, M, I, P, A, B, D, S, E, H, C, Q, N$;*
2. *rectangle visibility: R and P display the finite product cells before either iterated route is read;*
3. *measure and integral compatibility: M and I attach measure and integral rows to the same finite cell table;*
4. *route symmetry: A and B read the same cells in opposite coordinate order and share the tolerance row D;*
5. *rational readback: S reads the finite sums produced by both routes through regular rational data;*
6. *Real sealing: E is reached only after both iterated routes, the dyadic ledger, and regular rational readback have been displayed;*
7. *ledger nonescape: H, C, Q, N export no sigma-finite theorem, Tonelli theorem, quotient function equality, selected representative, countable-choice schedule, ambient measure completeness, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FubiniFiniteRectangleUp.FubiniFiniteRectangleNamecertObligations`

Proof. Project the carrier of Definition 3854.1. The accepted rows are exactly $R, M, I, P, A, B, D, S, E, H, C, Q, N$, so carrier admission exposes a finite rectangle table, its measure and integral rows, two displayed iterated routes, a dyadic ledger, rational readback, a Real seal, and structural rows.

The finite rectangle route starts with R and P. These rows fix the cell table and the coordinate-product reading, so neither iterated route may introduce a new domain or a hidden product object.

Next read M, I, A, B, D, S. The measure and integral rows are attached to the same finite cells. The two iterated routes fold those cells in opposite coordinate order, while the dyadic ledger and regular rational readback force both routes to use the same finite tolerance and sum data.

Finally E, H, C, Q, N seal and transport the same displayed computation. Since none of these rows adds sigma-finite measure structure, Tonelli data, quotient function equality, selected representatives, countable choice, ambient measure completeness, or a bridge certificate, the listed obligations exhaust the local surface. □

Theorem 3854.3 (Fubini finite rectangle iterated-integral nonescape)**.** *Every iterated-integral consumer of an accepted* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *packet factors through*

$$R \longmapsto M \longmapsto I \longmapsto P \longmapsto A, B \longmapsto D \longmapsto S \longmapsto E \longmapsto H, C, Q, N.$$

Thus a consumer may cite equality of the two finite rectangle reads only through the displayed cells, measure rows, integral rows, product row, both iterated routes, dyadic tolerance ledger, regular rational readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows. It obtains no sigma-finite Fubini theorem, Tonelli theorem, quotient function equality, selected representative, countable-choice schedule, or ambient measure-completeness principle.*

Proof. Begin with an iterated-integral request. The finite rectangle row R and product row P identify the only cells and coordinate directions available to the request.

The measure and integral rows M and I then bind the computation to that same table. The first-then-second route A and second-then-first route B are therefore two readings of the displayed finite cell data rather than appeals to an ambient product-measure theorem.

The dyadic ledger D and regular rational readback S force both route totals through the same finite tolerance and sum surface. Only after those rows have been displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal E be consumed.

The structural rows H, C, Q, N transport and name the same finite route. Since the packet contains no row for sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, or ambient measure completeness, no iterated-integral consumer can obtain such data from the accepted carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}$.

Constructive story: A $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ packet is generated from finite product rectangle cells, displayed Measure and Integral rows, a Product row, first-then-second and second-then-first iterated routes, a DyadicRatCore tolerance ledger, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3854.1, Definition 3854.2, and Definition 3854.3 seed the finite rectangle iterated-integral packet over displayed product cells and common tolerance readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FubiniFiniteRectangle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-finite Fubini, Tonelli, quotient function equality, selected representatives, countable choice, ambient measure completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rectangle visibility, measure-integral compatibility, route symmetry, regular rational readback, Real sealing, ledger nonescape, and local naming for BEDC.Derived.FubiniFiniteRectangleUp.

Chapter 3855

A Concrete Naming Certificate for $\mathsf{TonelliFiniteRectangle}^{\uparrow}$

Following Tonelli and Fubini on nonnegative iterated integration, the $\mathsf{TonelliFiniteRectangle}^{\uparrow}$ packet records the finite nonnegative rectangle-sum surface that may be read before any sigma-additive or completed measure theorem is available. It is a conservative companion to chapter 3854; it also sits beside the finite summation surfaces of chapter 1615 and chapter 1880, the measure and integral rows of chapter 92 and chapter 123, and the terminal real-name seal of chapter 38. The packet names finite nonnegative rectangular data only; it does not assert countable Tonelli, sigma-additivity, countable choice, measure completion, or an ambient product-measure theorem.

Definition 3855.1 (Tonelli finite rectangle carrier)**.** *A* $\mathsf{TonelliFiniteRectangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, C, A, U, V, F, E, H, K, P, N)$$

with the following displayed rows:

1. *R is the row-index ledger, listing the finite outer rectangle coordinates and their cell tags.*
2. *C is the column-index ledger, listing the finite inner rectangle coordinates attached to the same cells.*
3. *A is the nonnegative rectangular entry ledger, recording each displayed cell value, its nonnegativity witness, and the finite measure/integral rows used to read it.*
4. *U is the row-first finite-sum readback, summing each row over C before folding the visible row ledger R.*
5. *V is the column-first finite-sum readback, summing each column over R before folding the visible column ledger C.*
6. *F is the* $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ *handoff row, requiring both readbacks to remain inside the finite rectangle packet of chapter 3854.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, C, A, U, V, F have been displayed.*
8. *H records componentwise* hsame *transport for the row ledger, column ledger, nonnegative entry ledger, both finite-sum readbacks, the Fubini handoff, Real seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a finite nonnegative rectangle request through $R, C, A, U, V, F, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{TonelliFiniteRectangle}^{\uparrow}$, $\mathsf{FubiniFiniteRectangle}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{TonelliFiniteRectangle}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $R, C, A, U, V, F, E, H, K, P, N$. No coordinate stores sigma-additivity, countable summation, countable choice, measure completion, quotient function equality, selected representatives, or an ambient product-measure theorem.

Theorem 3855.2 (Tonelli finite rectangle NameCert obligations)**.** *For every accepted* $\mathsf{TonelliFiniteRectangle}^{\uparrow}$ *carrier $T = (R, C, A, U, V, F, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TonelliFiniteRectangle}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; nonnegative ledger visibility through A; row-first and column-first finite-sum route equality through U, V; confinement of the Fubini handoff row F to the finite rectangle surface; Real sealing through E; componentwise transport, replay, provenance, and local naming through H, K, P, N; and refusal of sigma-additivity, countable choice, measure completion, quotient function equality, selected representatives, and ambient product-measure escape.*

Proof. Project the carrier of Definition 3855.1. The only index rows are R and C, and the only cell-value row is A, so carrier admission and nonnegative ledger visibility are read from those displayed coordinates.

The two finite-sum routes are the rows U and V. Since both routes consume the same finite rectangle cells from R, C, A, their equality is a finite ledger comparison before any completed measure or countable summation principle can be cited.

Now read the handoff row F. It points to the already finite $\mathsf{FubiniFiniteRectangle}^{\uparrow}$ packet and therefore confines the equality route to displayed rectangle cells, not to a sigma-finite product theorem.

Finally E, H, K, P, N seal, transport, replay, source, and name the same finite packet. Because no carrier coordinate stores sigma-additivity, countable choice, measure completion, quotient function equality, selected representatives, or ambient product-measure structure, those readings are refused by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{TonelliFiniteRectangle}^{\uparrow}}$.

Constructive story: A $\mathsf{TonelliFiniteRectangle}^{\uparrow}$ packet is generated from finite row and column ledgers, nonnegative rectangular entry rows, two iterated finite-sum readbacks, a FubiniFiniteRectangle handoff row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TonelliFiniteRectangle}^{\uparrow}}$, seed)

Scope closed: Definition 3855.1 and Definition 3855.2 seed the finite nonnegative rectangle-sum packet over displayed row, column, entry, iterated-sum, Fubini handoff, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TonelliFiniteRectangle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable Tonelli, sigma-additivity, countable choice, measure completion, quotient function equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nonnegative ledger visibility, finite route equality, Fubini handoff confinement, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.TonelliFiniteRectangleUp.

Chapter 3856

A Concrete Naming Certificate for $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$

The $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ packet realizes the disk-dynamics fixed-end reading of section 9463.5 in chapter 9463 as finite disk metric, boundary, flow, and provenance rows. It sits after the metric-read surface of chapter 3474, the boundary-geodesic packet of chapter 3491, the finite routed-flow carrier of chapter 195, the disk-coordinate surface of chapter 377, and the disk phase-transport route of chapter 3455. The packet names a finite geodesic-flow surface; it does not complete the hyperbolic disk, quotient ideal endpoints, choose a global geodesic representative, or identify observer expansion with a physical spacetime law.

Definition 3856.1 (Hyperbolic geodesic-flow carrier)**.** *A* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (U, M, B, T, D, F, J, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source and target observations inspected by the flow.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same displayed disk data.*
3. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary row, recording the finite far-end incidence used by the route.*
4. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *transport row carrying reversible disk-surface movement and fiber provenance.*
5. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *routed-flow row, displaying the finite step relation read as geodesic-flow evolution.*
6. *F is the fixed-end row stating that the boundary far-end commitment is preserved along the displayed route.*
7. *J is the classifier-stability row for disk coordinates, metric reads, boundary incidence, phase transport, routed flow, and fixed-end data.*
8. *A is the audit row tying U, M, B, T, D, F, J to the hyperbolic observer-expansion vision and to its sibling concrete packets.*
9. *H records componentwise* hsame *transport for disk-coordinate, metric, boundary, phase-transport, routed-flow, fixed-end, classifier-stability, audit, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a geodesic-flow request through* $U, M, B, T, D, F, J, A, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, B, T, D, F, J, A, H, C, P, N$. *The carrier has no coordinate for a completed hyperbolic disk, quotient ideal-boundary equality, a selected global geodesic, a host analytic flow, physical cosmology, or observer substance.*

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicFlowUp.HyperbolicGeodesicFlowTasteGate single carrier alignment`

Theorem 3856.2 (Hyperbolic geodesic-flow NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *carrier* $G = (U, M, B, T, D, F, J, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *carrier admission: the accepted object is exactly the displayed packet* $U, M, B, T, D, F, J, A, H, C, P, N$;
2. *disk and metric visibility:* U *and* M *display disk coordinates and hyperbolic metric reads before any flow row is exported;*
3. *boundary incidence:* B *supplies the finite far-end data consumed by the geodesic route;*
4. *phase and flow routing:* T *and* D *display the reversible phase-transport route and finite routed-flow relation;*
5. *fixed-end stability:* F *and* J *preserve boundary far-end and classifier readings along the displayed route;*
6. *audit provenance:* A, H, C, P, N *source, transport, replay, package, and name the same finite rows;*
7. *ledger nonescape: no row exports a completed disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, observer substance, or bridge certificate.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.HyperbolicGeodesicFlowUp.HyperbolicGeodesicFlowNamecertObligations`

Proof. Project the carrier of Definition 3856.1. The accepted rows are exactly $U, M, B, T, D, F, J, A, H, C, P, N$, so carrier admission exposes finite disk observations, metric reads, boundary incidence, transport, routed flow, fixed-end data, classifier stability, audit provenance, and structural rows only.

The route begins with U and M. These rows bind the computation to displayed disk and metric data, so the flow cannot import a completed hyperbolic plane or an ambient analytic model.

Next read B, T, D, F, J. Boundary incidence supplies the finite far-end, phase transport and routed flow display the movement, and fixed-end plus classifier-stability rows keep the same boundary commitment through the route.

Finally A, H, C, P, N audit, transport, replay, source, and name the finite packet. Since none of these rows adds quotient ideal-boundary equality, selected global geodesics, host analytic flow, physical cosmology, observer substance, or a bridge certificate, the listed obligations exhaust the local surface. □

Theorem 3856.3 (Hyperbolic geodesic-flow fixed-end nonescape)**.** *Every geodesic-flow consumer of an accepted* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicGeodesicBoundary}^{\uparrow} \longmapsto \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto \mathsf{DynSyster}$$

Thus a consumer may read disk coordinates, hyperbolic metric data, finite boundary incidence, phase transport, routed flow, fixed-end stability, classifier preservation, audit provenance, transport, replay, package sourcing, and local naming. It obtains no completed hyperbolic disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance.

Proof. Begin with a geodesic-flow request. The disk-coordinate row U and metric-read row M give the only geometric data available to the request.

The boundary row B supplies finite far-end incidence. The transport row T and routed-flow row D then display movement through certified phase transport and finite flow data, rather than through a host geodesic-flow theorem.

The fixed-end and classifier rows F, J preserve the boundary commitment and the sibling classifiers along the displayed route. The audit and structural rows A, H, C, P, N preserve that same finite packet.

No carrier coordinate names a completed hyperbolic disk, quotient ideal-boundary equality, selected global geodesic, host analytic flow, physical cosmology, or observer substance. Therefore a geodesic-flow consumer cannot escape the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ read is refuted if a geodesic-flow consumer obtains a completed disk, quotient ideal-boundary equality, a selected global geodesic, or a host analytic flow without displaying UnitDisk coordinates, HyperbolicMetric reads, HyperbolicGeodesicBoundary incidence, PoincareDiskPhaseTransport routing, DynSystem flow rows, fixed-end stability, transport, replay, provenance, and local naming.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-boundary, dynsystem, unitdisk, poincare-disk-phase-transport: none is bijective to $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ because the packet jointly carries disk observations, metric reads, boundary incidence, reversible phase transport, finite routed flow, fixed-end stability, classifier preservation, audit provenance, transport, replay, package sourcing, and local NameCert rows.

The Lean TasteGate for this packet will require explicit $\mathsf{Nontrivial\ HyperbolicGeodesicFlow}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicGeodesicFlow}^{\uparrow}$ instances. Those instances must keep $U, M, B, T, D, F, J, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric reads, HyperbolicGeodesicBoundary incidence, PoincareDiskPhaseTransport rows, DynSystem routed-flow rows, fixed-end stability, classifier preservation, audit provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3856.1, Definition 3856.2, and Definition 3856.3 seed the finite geodesic-flow packet over disk metric, boundary, flow, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic disks, quotient ideal-boundary equality, selected global geodesics, host analytic flows, physical cosmology, observer-substance readings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk visibility, metric-read visibility, boundary incidence, phase-flow routing, fixed-end stability, TasteGate nontriviality, field faithfulness, ledger nonescape, and local naming for BEDC.Derived.HyperbolicGeodesic

Chapter 3857

A Concrete Naming Certificate for $\mathsf{LocatedSequence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedSequence}^{\uparrow}$ packet records the finite surface by which a sequence is read through located windows before it is consumed as regular rational, located real, or real-facing data. It sits after the finite stream observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the located-real boundary of **??**, and the terminal real-name seal of chapter 38. The chapter isolates the generic located-sequence carrier used by later completion and compact-uniform-continuity routes; it does not choose a quotient stream, a selected limit, a trichotomy decision, countable choice, or ambient real completeness.

Definition 3857.1 (Located sequence carrier)**.** *A* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, D, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing exactly the finite sequence windows that may be inspected by a consumer.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the inspected stream windows as regular rational sequence data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used when located windows are compared, tightened, or handed to a modulus consumer.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *window row, recording the located interval or ball evidence available at the displayed sequence indices.*
5. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream, regular rational, dyadic, and located-real rows have been projected.*
6. *H records componentwise* hsame *transport for stream windows, regular rational readback, dyadic tolerances, located-real windows, real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite sequence request through*

$$S, D, L \longmapsto R \longmapsto Q.$$

8. *P is the* Pkg *provenance over* $\mathsf{LocatedSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets in which the displayed stream windows, dyadic tolerance budgets, located-real windows, regular rational readback, and real seal describe one finite route. It has no coordinate for quotient streams, selected representatives, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedSequenceUp.TasteGate.LocatedSequenceTasteGate single carrier alignment`

Theorem 3857.2 (Located sequence NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier $A = (S, R, D, L, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$ *surface consists exactly of stream-window admission, regular rational readback, dyadic tolerance accounting, located-real window coverage, real-seal routing, componentwise transport, continuation replay, package provenance, and local naming through the rows $S, R, D, L, Q, H, C, P, N$. Consumers from* $\mathsf{LocatedSequenceModulus}^{\uparrow}$, $\mathsf{LocatedSequenceLimit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *may read the packet only through those displayed rows.*

Proof. Project the carrier of Definition 3857.1. The sequence is visible only through S, and every tolerance-facing comparison is visible only through the dyadic row D and the located-real row L.

The readback row R repacks the same inspected windows as $\mathsf{RegSeqRat}^{\uparrow}$ data. The real seal Q is downstream of S, D, L, R, so a real-facing consumer reaches it only after replaying the displayed located route.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier has no coordinate for quotient streams, selected representatives, host equality, trichotomy, countable choice, or ambient completeness, the listed NameCert rows are exhaustive. □

Theorem 3857.3 (Located sequence regular rational readback)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *carrier must read*

$$S, D, L \longmapsto R \longmapsto Q \longmapsto H, C, P, N.$$

Thus regular rational readback is a displayed finite route from stream windows and located dyadic tolerances to the real seal, not a selected quotient stream, not host real equality, not a trichotomy principle, and not ambient real completeness.

Proof. Begin with a consumer request for regular rational or real-facing data. The packet exposes sequence observations only through S, and each located precision request is accounted for by D and L.

The row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback surface. Therefore a located-real or real-facing consumer reaches the terminal seal Q only after the finite route through S, D, L, R has been displayed.

The structural rows H, C, P, N preserve that route by transport, replay, source, and local naming. No excluded analytic object is present in the tuple, so no downstream consumer can extract one from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedSequence}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, LocatedReal windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocatedSequence}\uparrow}$, seed)

Scope closed: Definition 3857.1, Definition 3857.2, and Definition 3857.3 seed the finite located-sequence surface for the L10 StreamName-to-RegSeqRat-to-Real route.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedSequence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window coverage, dyadic tolerance accounting, located-real window stability, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSequenceUp.

Chapter 3858

A Concrete Naming Certificate for $\mathsf{CauchyModulusExtractor}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyModulusExtractor}^{\uparrow}$ packet records the finite operation that reads an accepted effective Cauchy sequence and exposes the explicit modulus row required by regular-Cauchy and real-completion consumers. It sits over the source certificate of chapter 1705, the fast-threshold selector of chapter 1781, the ordinary modulus schedule of chapter 292, the finite-window service of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback boundary of chapter 301, and the terminal real-name seal of chapter 38. The packet is a displayed extraction route from explicit Cauchy evidence to a Real-facing handoff; it is not quotient stream equality, a host-selected limit, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 3858.1 (Cauchy modulus extractor carrier)**.** *A* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (E, F, M, S, D, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the accepted* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *source row whose explicit-modulus coordinate is to be read.*
2. *F is the* $\mathsf{FastCauchyModulus}^{\uparrow}$ *threshold row used when the consumer requests a fast or doubled-window presentation of the same precision query.*
3. *M is the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row, displayed before any finite stream window or regular rational readback is consumed.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the stage selected from M and, when needed, refined by F.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the precision query, slack rows, comparison budgets, and tolerance conversions attached to the extracted stage.*
6. *Q is the precision-query row whose requested tolerance is routed through E, F, M, S, D.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff row exposed only after the source, extractor, finite-window, and dyadic rows have been displayed.*
8. *H records componentwise* hsame *transport for the source, fast selector, extracted modulus, finite window, dyadic ledger, query, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a regular-Cauchy or Real-facing request through $E, F, M, S, D, Q, R, H, P, N$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusExtractor}^{\uparrow}$, $\mathsf{EffectiveCauchySequence}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite explicit-modulus extraction packet. It has no coordinate for quotient stream equality, host-selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyModulusExtractorUp.CauchyModulusExtractorTasteGate single carrier alignment`

Theorem 3858.2 (Cauchy modulus extractor NameCert obligations)**.** *For every accepted* CauchyModulusExtractor *carrier $X = (E, F, M, S, D, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed effective Cauchy source, fast threshold selector, extracted modulus, finite stream window, dyadic tolerance ledger, precision query, regular and Real handoff, transport, replay, provenance, and local naming rows;*
2. *source exposure, requiring the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *row E to be read before M is exported;*
3. *modulus extraction, requiring M to be the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row exposed from E for the same precision query Q;*
4. *threshold compatibility, requiring F to refine only the same selected threshold row, either directly or through the doubled-window route;*
5. *finite-window and dyadic accounting, requiring S and D to be downstream of M and F before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read is consumed;*
6. *handoff nonescape, requiring R to expose only the regular rational readback or Real seal certified by E, F, M, S, D, Q;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same displayed rows through H, C, P, N.*

No extractor obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3858.1. The accepted packet displays exactly the source effective Cauchy row, the fast selector, the extracted modulus, finite windows, the dyadic ledger, the precision query, the regular and Real handoff, and the structural rows.

Source exposure uses Definition 1705.2. The explicit modulus row is part of the accepted $\mathsf{EffectiveCauchySequence}^{\uparrow}$ carrier surface, so an extractor cannot export M before it has read the source row E.

Threshold compatibility is supplied by Definition 1781.4 and Definition 1781.5. The row F selects the direct fast threshold or the doubled-window threshold for the same query Q, and it creates no third selector coordinate.

The finite-window and dyadic rows then follow chapter 126 and chapter 291. The handoff row R is downstream of that route, while H, C, P, N transport, replay, package, and name the same rows. Hence the listed obligations exhaust the local certificate surface. □

Theorem 3858.3 (Cauchy modulus extractor stage determinacy)**.** *Let $X = (E, F, M, S, D, Q, R, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier. For the displayed precision query Q, every inspected* $\mathsf{StreamName}^{\uparrow}$ *window row S is opened at the stage determined by the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row M together with the fast-threshold row F. Thus two accepted reads with the same transported E, F, M, Q, H, C, P, N rows cannot diverge on the stage used by S or on the dyadic tolerance ledger D.*

Proof. Project the accepted carrier. The only coordinates that mention stage selection are E, F, M, Q together with the downstream window and ledger rows S, D.

The source theorem Definition 1705.3 places every regular rational read behind the explicit modulus row carried by E. The extractor reads that row as M for the same query Q.

The fast-modulus theorems Definition 1781.4 and Definition 1781.5 then determine whether the direct threshold or doubled threshold is used, and identify the corresponding S and D rows.

Since the carrier has no additional selector coordinate, transported agreement of E, F, M, Q, H, C, P, N forces the same selected stage, the same finite window, and the same dyadic ledger. □

Theorem 3858.4 (Cauchy modulus extractor Real handoff nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusExtractor}^{\uparrow}$ *carrier $X = (E, F, M, S, D, Q, R, H, C, P, N)$ must first read the finite route*

$$E \longmapsto M, F, Q \longmapsto S, D \longmapsto R \longmapsto H, C, P, N.$$

Consequently the handoff exports no quotient stream equality, host-selected limit, countable-choice schedule, trichotomy principle, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, hidden stream read, Lean verification row, or bridge datum.*

Proof. Start from a regular-Cauchy or Real-facing consumer. The carrier definition exposes the handoff only as the row R, downstream of E, F, M, S, D, Q.

Apply Definition 1705.5. The effective source row E permits a Real-facing read only after its explicit modulus, finite windows, dyadic ledger, and regular rational handoff have been displayed.

Apply Definition 3858.3. The extractor route fixes the stage row used by S and the tolerance account stored in D before the handoff row is consumed.

The structural rows H, C, P, N preserve that route and contain no excluded analytic or verification coordinate. Therefore the regular-rational or Real handoff cannot escape the displayed extraction packet. □

Falsifiable prediction: Every $\mathsf{CauchyModulusExtractor}^{\uparrow}$ regular or Real-facing read factors through an EffectiveCauchySequence source row, an extracted CauchyModulus row, a FastCauchyModulus threshold row, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: effective-cauchy-sequence, fast-cauchy-modulus, cauchymodulus, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyModulusExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source effective-Cauchy row, extracted modulus, fast threshold selector, finite window, dyadic ledger, regular and Real handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3858.1, Definition 3858.2, Definition 3858.3, and Definition 3858.4 seed the finite explicit-modulus extraction route from EffectiveCauchySequence$^{\uparrow}$ through FastCauchyModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusExtractor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host-selected limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, modulus extraction, threshold compatibility, finite-window and dyadic accounting, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusExtractorUp.

Constructive story: A CauchyModulusExtractor$^{\uparrow}$ packet is generated from an EffectiveCauchySequence source row, a FastCauchyModulus threshold row, an extracted CauchyModulus row, finite StreamName windows, a DyadicRatCore tolerance ledger, a precision query, a RegSeqRat and RealUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3859

A Concrete Naming Certificate for $\mathsf{RealArithmeticProgression}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealArithmeticProgression}^{\uparrow}$ packet records the textbook arithmetic progression route $a, a+d, a+2d, \ldots$ as finite BEDC data before any completed real-analysis theorem is read. It sits over the terminal real-name seal of chapter 38, the regular rational readback surface of chapter 301, the finite stream-window discipline of chapter 126, and the finite dyadic arithmetic ledgers of chapter 291. The packet supplies a constructive L10 sequence object for later finite-window sequence and series arguments; it is not a host function, not an infinite completed sum, not a quotient-real equality, not trichotomy, and not an ambient completeness principle.

Definition 3859.1 (Real arithmetic progression carrier)**.** *A* $\mathsf{RealArithmeticProgression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (a, d, S, R, D, H, C, P, N)$$

with the following displayed rows:

1. *a is the initial* $\mathsf{Real}^{\uparrow}$ *row, read through its displayed regular-rational name rather than through a host real value;*
2. *d is the common-difference* $\mathsf{Real}^{\uparrow}$ *row, read through the same real-name boundary;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule listing the indices currently requested from the progression;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row whose nth window is obtained by the displayed finite arithmetic route for $a + nd$;*
5. *D is the finite dyadic arithmetic ledger recording the repeated addition budget, tolerance scale, and endpoint readbacks used by the requested windows;*
6. *H records componentwise* hsame *transport for the initial real row, common-difference row, schedule, regular-rational readback, dyadic ledger, continuation, provenance, and local naming rows;*
7. *C records displayed* Cont *replay from a finite index request through S, then through the dyadic ledger D, the regular-rational readback R, and the endpoint real rows a, d;*
8. *P is the* Pkg *provenance over* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a host function $\mathbb{N} \to \mathbb{R}$, quotient equality of real representatives, selected limits, infinite sums, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealArithmeticProgressionUp.RealArithmeticProgressionTasteGate single carrier alignment`

Theorem 3859.2 (Real arithmetic progression NameCert obligations)**.** *For every accepted* RealArithmeticProgressio *carrier $A = (a, d, S, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the initial real row a, common-difference row d, finite schedule S, regular-rational readback R, dyadic arithmetic ledger D, transport, continuation, provenance, and local naming row;*
2. *finite-window exactness, requiring each inspected index to be replayed through S before the arithmetic readback R is exposed;*
3. *arithmetic-ledger exactness, requiring the displayed row D to account for the finite repeated-addition route from a and d to every requested readback;*
4. *real-name routing, requiring every downstream* $\mathsf{Real}^{\uparrow}$ *or sequence-facing consumer to reach the progression through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *rows;*
5. *transport stability, requiring componentwise* hsame *transport to preserve the same initial row, common-difference row, schedule, readback, and dyadic ledger;*
6. *ledger refusal, excluding host functions, completed infinite sums, quotient-real equality, trichotomy, selected limits, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 3859.1. This exposes exactly the rows $a, d, S, R, D, H, C, P, N$ and no detached operation row.

Replay the continuation C at a finite index request. The request first passes through the $\mathsf{StreamName}^{\uparrow}$ schedule S, then reads the repeated-addition ledger D, and only then exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback R tied to the endpoint rows a and d.

Transport by H is componentwise, so changing representative histories preserves the same initial real row, common-difference row, finite schedule, arithmetic ledger, readback, replay route, provenance, and local naming row. The provenance row P cites only the displayed Real, regular-rational, stream, dyadic, and kernel surfaces.

Since the tuple has no coordinate for a host sequence function, quotient-real equality, infinite sum, trichotomy decision, selected limit, or ambient completeness theorem, those exits cannot be read as local $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$ obligations. □

Lean4 ✓ `BEDC.Derived.RealArithmeticProgressionUp.RealArithmeticProgressionNameCertObligations`

Closure status for $\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$.

Constructive story: A $\mathsf{RealArithmeticProgression}^{\uparrow}$ packet is generated from an initial Real row, a common-difference Real row, a finite StreamName schedule, a RegSeqRat readback, a DyadicRatCore repeated-addition ledger, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$, seed)

Scope closed: Definition 3859.1 and Definition 3859.2 seed the finite arithmetic-progression route from $\mathsf{Real}^{\uparrow}$ endpoints through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ ledgers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealArithmeticProgression}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completeness of $\mathsf{Real}^{\uparrow}$, trichotomy, quotient-real equality, host functions $\mathbb{N} \to \mathbb{R}$, selected limits, infinite sums, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exactness, arithmetic-ledger exactness, real-name routing, transport stability, and ledger refusal for the same packet.

Chapter 3860

A Concrete Naming Certificate for $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ packet records the finite readback ledger through which a regular Cauchy presentation is reflected into completion-facing consumers. It sits after the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the reflector surface of chapter 1585, and the extension boundary of chapter 397. The ledger exposes a BEDC-native reflection route for completion consumers without quotient streams, selected limits, host real equality, countable choice, or ambient completeness.

Definition 3860.1 (Regular Cauchy reflection ledger carrier)**.** *A* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, W, D, R, K, U, J, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational Cauchy readback.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which the source row is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the regular Cauchy read.*
4. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from Q, W, D.*
5. *K is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row recording the reflector-facing handoff from the displayed regular Cauchy route.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row consumed by completion-facing maps.*
7. *J is the reflection ledger row: it records that the same Q, W, D, R source route feeds both K and U.*
8. *H records componentwise* hsame *transport for Q, W, D, R, K, U, J and the local naming row.*
9. *C records displayed* Cont *replay of the route*

$$Q, W, D \longmapsto R \longmapsto K, U \longmapsto J, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, W, D, R, K, U, J, H, C, P, N$. It has no coordinate for quotient stream classes, selected completion limits, host equality of real names, countable-choice schedules, hidden cofinal tails, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyReflectionLedgerUp.RegularCauchyReflectionLedgerTasteGate`
`single carrier alignment`

Theorem 3860.2 (Regular Cauchy reflection ledger NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *carrier $L = (Q, W, D, R, K, U, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyReflection}}$ *surface consists exactly of carrier admission; regular rational source exposure through Q; finite stream exposure through W; dyadic tolerance exposure through D; terminal real sealing through R; reflector handoff through K; universal-property extension through U; reflection ledger binding through J; componentwise transport through H; displayed replay through C; provenance through P; local naming through N; and non-escape from quotient streams, selected limits, host real equality, countable choice, hidden cofinal tails, or ambient completeness.*

Proof. Carrier admission is projection from Definition 3860.1. The accepted packet exposes exactly the regular rational source, finite stream schedule, dyadic tolerance ledger, real seal, reflector handoff, universal-property extension row, reflection ledger, transport, replay, provenance, and local name.

The source route is finite. Rows Q, W, D determine the regular Cauchy read before R can be consumed, so the packet never begins from a quotient stream class or a selected limit representative.

The completion-facing route is read only after the real seal is displayed. Rows K and U expose the reflector and universal-property surfaces, while J records that both surfaces use the same Q, W, D, R source route.

Rows H, C, P, N transport, replay, source, and name the same ledger. Since no excluded host-side coordinate exists in the carrier, the listed rows exhaust the local $\mathsf{NameCert}$ surface. □

Theorem 3860.3 (Regular Cauchy reflection ledger universal handoff)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ *packet factors through*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyCompletionReflector}^{\uparrow}, \mathsf{CompletionUniversalProperty}$$

The handoff keeps the regular Cauchy source, finite observation schedule, dyadic tolerance ledger, terminal real seal, reflector row, universal-property row, transport, replay, provenance, and local naming inside one finite packet. It exports no quotient stream, selected limit, host real equality, countable-choice schedule, hidden cofinal tail, or ambient completeness witness.

Proof. Start with the displayed source rows. The regular rational readback Q is inspected through the finite windows W and the dyadic tolerance ledger D, and only then reaches the real seal R.

The reflector and universal-property rows are downstream of that same source route. The row K records the reflector-facing handoff, while U records the extension-facing handoff for completion consumers.

The ledger row J binds the two completion reads to the same Q, W, D, R route. The structural rows H, C, P, N transport, replay, source, and name that binding, so every completion-facing consumer remains inside the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ consumer that reaches a completion-facing reflector or universal-property row while omitting any of the regular rational source, finite stream schedule, dyadic tolerance ledger, terminal real seal, reflection ledger, transport, replay, provenance, or local naming rows refutes the packet.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchycompletionreflector, completionuniversalproperty: none is bijective because $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ jointly carries the source readback, finite observation schedule, tolerance ledger, real seal, reflector handoff, universal-property handoff, reflection binding, transport, replay, provenance, and local NameCert rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3860.1, Definition 3860.2, and Definition 3860.3 seed the finite regular-Cauchy reflection ledger over RegSeqRat, StreamName, DyadicRatCore, Real, CauchyCompletionReflector, CompletionUniversalProperty, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyReflectionLedger}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyReflectionLedgerUp.RegularCauchyReflectionLedgerTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, hidden cofinal tails, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular source exposure, tolerance exactness, reflector handoff, universal-property handoff, ledger binding, structural transport, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.RegularCauchyReflectionLedgerUp.

Constructive story: A $\mathsf{RegularCauchyReflectionLedger}^{\uparrow}$ packet is generated from a regular rational source row, finite stream windows, dyadic tolerance data, a terminal Real seal, a CauchyCompletionReflector row, a CompletionUniversalProperty row, a reflection ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 3861

A Concrete Naming Certificate for $\mathsf{TubeLemma}^{\uparrow}$

Following Munkres and Kelley on product topology and compact neighbourhoods, the $\mathsf{TubeLemma}^{\uparrow}$ packet records the finite BEDC surface for the tube lemma: when a slice around a compact factor lies inside an open product-neighbourhood row, a displayed neighbourhood in the other factor suffices for the whole compact slice. It sits over the product topology certificate of chapter 2000, the generated-open topology rows of chapter 88, the compact finite-cover surface of chapter 60, and the finite-cover siblings such as chapter 1596 and chapter 3430. The chapter names only the finite tube-neighbourhood carrier and its local obligations; it does not assert product compactness, quotient equality of product points, selected neighbourhood bases, or ambient open-cover compactness beyond the displayed compact row.

Definition 3861.1 (Tube lemma carrier)**.** *A* $\mathsf{TubeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, K, O, U, V, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Topology}^{\uparrow}$ *row naming the first factor.*
2. *Y is the source* $\mathsf{Topology}^{\uparrow}$ *row naming the second factor.*
3. *K is the* $\mathsf{Compact}^{\uparrow}$ *row for the compact slice inside the second factor.*
4. *O is the* ***??*** *open-neighbourhood row containing the displayed slice.*
5. *U is the finite family of product-open rows read from O along the compact slice.*
6. *V is the resulting neighbourhood row in the first factor.*
7. *F is the finite-cover row over K that justifies replacing the family U by one displayed tube neighbourhood.*
8. *H records componentwise* hsame *transport for the two topology rows, compact row, product-open row, finite family, tube neighbourhood, finite-cover row, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a slice-neighbourhood request through $X, Y, K, O, U, V, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{TubeLemma}^{\uparrow}$*,* ***??****,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Compact}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

11. N is the local $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, K, O, U, V, F, H, C, P, N$. It has no coordinate for global product compactness, a host topological theorem, quotient equality of product points, an arbitrary open-cover selector, or a detached neighbourhood basis.

Lean4 ✓ `BEDC.Derived.TubeLemmaUp.TubeLemmaTasteGate single carrier alignment`

Theorem 3861.2 (Tube lemma NameCert obligations)**.** *For every accepted* $\mathsf{TubeLemma}^{\uparrow}$ *carrier $T = (X, Y, K, O, U, V, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$ *surface is exhausted by carrier admission, topology-row exposure for both factors, compact-slice exposure through K, product-open exposure through O, finite product-neighbourhood family exposure through U, tube-neighbourhood exposure through V, finite-cover justification through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3861.1. The accepted tuple exposes the two topology rows, the compact slice row, the product-open row, the finite product-neighbourhood family, the resulting tube neighbourhood, and the finite-cover row.

The compactness use is local to K and F. The finite family U is read from the displayed product-open neighbourhood O along the compact slice, and F records the finite-cover handoff that permits one neighbourhood row V in the first factor to cover the whole slice.

Finally transport and replay preserve the same finite route. Since H, C, P, N contain only componentwise transport, continuation replay, provenance, and naming, they cannot add product compactness, quotient product equality, selected bases, or an arbitrary open-cover selector. The listed rows therefore exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{TubeLemma}^{\uparrow}$ packet is generated from two topology rows, a compact slice row, a product-open neighbourhood row, a finite product-neighbourhood family, a tube neighbourhood row, a finite-cover row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$, seed)

Scope closed: Definition 3861.1 and Definition 3861.2 seed the finite tube-neighbourhood packet over product topology, compact slice, generated-open, and finite-cover rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TubeLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close product compactness, arbitrary open-cover compactness, quotient equality of product points, selected neighbourhood bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked topology-row exposure, compact-slice exposure, product-open exposure, finite-cover justification, tube-neighbourhood exactness, transport replay, provenance exhaustion, and local naming for BEDC.Derived.TubeLemmaUp.

Chapter 3862

A Concrete Naming Certificate for $\mathsf{FilterLimitBasis}^{\uparrow}$

Following Kelley and Bourbaki on filters, convergence, and completion by directed neighbourhood data, the $\mathsf{FilterLimitBasis}^{\uparrow}$ packet records the finite basis interface used when a completion consumer asks for limit evidence before reading a terminal real-name seal. It sits after the Cauchy-filter completion boundary of chapter 1569, the Cauchy-filter surface of chapter 333, the limit boundary of chapter 1558, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a traditional finite filter-basis route for limit reads; it is not a quotient filter, selected limit, host real equality, ultrafilter argument, countable-choice selector, or ambient completeness theorem.

Definition 3862.1 (FilterLimitBasis carrier)**.** *A* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, F, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-boundary row whose accepted reads are still tied to finite Cauchy-filter data.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row exposing the saturated finite-filter surface consumed by the basis request.*
3. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *limit-boundary row, read only after the same filter and basis data have been displayed.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the basis request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback at the requested precision.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after Q, F, L, W, R, D have been replayed.*
8. *H records componentwise* hsame *transport for completion, filter, limit, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a limit-basis request through $Q, F, L, W, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, L, W, R, D, E, H, C, P, N$. There is no coordinate for quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier field projection round trip`

Theorem 3862.2 (FilterLimitBasis NameCert obligations)**.** *For every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for the displayed Cauchy-filter completion row, Cauchy-filter row, limit-boundary row, finite stream window, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming row;*
2. *completion-basis exposure, requiring every completion-facing read to consume Q and F before it consults L, W, R, D, or E;*
3. *limit compatibility, requiring the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row L to use the same displayed filter, window, readback, and dyadic tolerance route;*
4. *Real handoff, requiring the terminal seal E to remain downstream of Q, F, L, W, R, D and the structural rows H, C, P, N;*
5. *ledger refusal, excluding quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, and bridge certificates.*

No local NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisNameCertObligations`

Proof. Project the carrier of Definition 3862.1. The displayed rows are exactly $Q, F, L, W, R, D, E, H, C, P, N$, so carrier admission exposes only finite completion, filter, limit, window, readback, tolerance, seal, transport, replay, provenance, and naming data.

The completion-basis route starts with Q and F. These rows force the request to pass through the accepted Cauchy-filter completion boundary and the saturated Cauchy-filter surface before any limit-facing row can be inspected.

Next read L, W, R, D. The limit boundary consumes the same finite stream windows, regular rational readback, and dyadic tolerance ledger, so the packet has one displayed basis route rather than two unrelated filter presentations.

Finally the terminal seal E and the structural rows H, C, P, N preserve the same route. Since quotient filters, host equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, and bridge certificates are absent from the carrier, the listed obligations exhaust the local surface. □

Theorem 3862.3 (FilterLimitBasis CauchyFilterCompletion handoff)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier factors through the finite route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus a completion-facing read receives the displayed Cauchy-filter completion boundary, the saturated filter surface, the limit boundary, finite stream windows, regular rational readback, dyadic tolerance accounting, the terminal real-name seal, componentwise transport, replay, provenance, and local naming. It receives no quotient filter, ultrafilter, selected limit, host real equality, countable-choice selector, ambient completeness theorem, or bridge certificate.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCauchyFilterCompletionHandoff`

Proof. Begin with a completion-facing consumer of B. The carrier admits the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row Q only together with the finite filter and limit rows F, L.

Replay the displayed continuation row C. The request consumes Q and F, then passes through the limit row L, finite windows W, regular readback R, and dyadic tolerance ledger D before the terminal seal E is available.

The structural rows do not create an additional completion object. Transport through H, provenance through P, and local naming through N record the same finite route, so every $\mathsf{CauchyFilterCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives only that route. □

Theorem 3862.4 (FilterLimitBasis limit compatibility)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier. The limit-boundary row L, finite-window row W, regular readback row R, and dyadic tolerance row D are read from the same displayed filter-basis data exposed by Q and F. In particular, a limit-facing consumer cannot replace the filter of F with a second host filter, detach W from the displayed basis request, or read R, D as tolerance evidence for a different filter presentation.*

↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasis limit compatibility`
↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier limit compatibility`

Proof. Project the carrier of Definition 3862.1. Its only completion and filter coordinates are Q and F, and its only limit-facing coordinates are L, W, R, D.

Apply the NameCert obligations of Definition 3862.2. The limit-compatibility row requires L to use the same displayed filter, window, readback, and dyadic tolerance route rather than a detached filter presentation.

Finally read the handoff theorem Definition 3862.3. Every completion-facing consumer follows the finite route Q, F, L, W, R, D before the terminal seal appears, so L, W, R, D remain tied to the filter-basis data already exposed by Q and F. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisLimitCompatibility`

Theorem 3862.5 (FilterLimitBasis ledger refusal)**.** *For every accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the local ledger has no coordinate for quotient filters, host filter equality, ultrafilters, selected limits, countable-choice schedules, ambient completeness, or bridge certificates. Any request for one of those objects is therefore refused by the absence of a carrier row and cannot be recovered from $Q, F, L, W, R, D, E, H, C, P, N$.*

Proof. Inspect the carrier definition. The displayed tuple contains completion, filter, limit, window, readback, tolerance, seal, transport, replay, provenance, and local naming rows, and no row with the requested quotient, ultrafilter, selected-limit, choice, completeness, or bridge role.

Apply Definition 3862.2. Its ledger-refusal clause lists exactly those excluded objects and states that the local NameCert surface stores no obligation outside the displayed rows.

The handoff theorem Definition 3862.3 then identifies every completion-facing and Real-facing read with the route through $Q, F, L, W, R, D, E, H, C, P, N$. Since the route has no additional ledger coordinate, the excluded objects remain refused for every accepted packet. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier ledger refusal`

Theorem 3862.6 (FilterLimitBasis Real handoff stability)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Componentwise transport of the displayed rows $Q, F, L, W, R, D, E, H, C, P,$ preserves the terminal* $\mathsf{Real}^{\uparrow}$ *handoff of Definition 3862.3. In particular, any transported completion-facing or Real-facing read still factors through the same finite route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N$$

and cannot create quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.

Proof. Start with the carrier definition Definition 3862.1. Its visible rows are exactly $Q, F, L, W, R, D, E, H, C, P, N,$ and the transport row H is componentwise over those rows rather than over an ambient filter quotient or a selected limit.

Apply the NameCert obligations of Definition 3862.2. They require completion-basis exposure through Q, F, limit compatibility through L, W, R, D, Real handoff through E, and structural replay through H, C, P, N.

Finally use the handoff theorem Definition 3862.3. Since the transported packet has the same displayed coordinates and no additional carrier row, the terminal $\mathsf{Real}^{\uparrow}$ handoff remains downstream of Q, F, L, W, R, D and cannot acquire quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisRealHandoffStability`

Theorem 3862.7 (FilterLimitBasis scoped route)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet exposes completion-facing limit evidence only through the finite route*

$$\mathsf{CauchyFilterCompletion}^{\uparrow} \longrightarrow \mathsf{CauchyFilter}^{\uparrow} \longrightarrow \mathsf{CauchyFilterLimit}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{DyadicRa}$$

followed by the structural hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. Equivalently, in the carrier notation of Definition 3862.1, every completion-facing, limit-facing, or Real-facing read must pass through $Q, F, L, W, R, D, E, H, C, P, N$ in that order before it is exported as limit-basis evidence.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedRoute`

Proof. Start with the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the only local obligation rows.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. It forces every completion-facing and Real-facing consumer through Q, F, L, W, R, D before the terminal seal E and the structural rows are available.

Next use the limit compatibility theorem Definition 3862.4. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ row L, finite window row W, regular readback row R, and dyadic tolerance row D remain attached to the same displayed filter-basis data exposed by Q and F.

Finally apply Real handoff stability from Definition 3862.6. Componentwise transport preserves the same route through H, C, P, N, so the scoped route is exactly the displayed completion, filter, limit, stream, regular-rational, dyadic, Real, and structural sequence. □

Theorem 3862.8 (FilterLimitBasis window-basis exhaustion)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *request must expose the completion row* Q*, the Cauchy-filter row* F*, the limit-boundary row* L*, the finite-window row* W*, the regular readback row* R*, and the dyadic tolerance row* D *before it may read the terminal seal* E*. Only after that finite basis has been displayed may the request continue through* H, C, P, N*. Consequently the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*, and* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *rows are all visible before any* $\mathsf{Real}^{\uparrow}$ *consumer can inspect the seal.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisWindowBasisExhaustion`

Proof. Project the carrier of Definition 3862.1. Its visible order places Q, F, L, W, R, D before the terminal seal row E, and then records transport, replay, provenance, and local naming as H, C, P, N.

Apply the NameCert obligations of Definition 3862.2. The completion-basis row forces Q, F to be read first, the limit-compatibility row ties L, W, R, D to that same request, and the Real handoff row keeps E downstream of those finite rows.

Finally use Definition 3862.4 and Definition 3862.6. They preserve the same displayed basis under limit-facing and transported Real-facing reads, so no $\mathsf{Real}^{\uparrow}$ consumer can reach the seal before the StreamName, RegSeqRat, DyadicRatCore, CauchyFilter, and CauchyFilterLimit rows are visible. □

Theorem 3862.9 (FilterLimitBasis window induction)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier, and let* Λ *be a finite replay ledger whose entries are finite window reads or structural transports of the displayed rows. Induction on* Λ *preserves the following invariant: every* $\mathsf{Real}^{\uparrow}$*-facing consumer exposes* Q, F, L, W, R, D *before* E*, and every later replay through* H, C, P, N *preserves that same basis route. Thus no induction step can introduce a quotient filter, ultrafilter, selected limit, countable-choice schedule, host filter equality, host real equality, or ambient completeness premise.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisWindowInduction`

Proof. For the empty ledger, apply Definition 3862.8. The terminal seal is available only after Q, F, L, W, R, D have been displayed, and the remaining rows are exactly the structural route H, C, P, N.

For the induction step, assume the finite prefix of Λ preserves the displayed basis route. If the next entry is a finite window read, use Definition 3862.4: the rows L, W, R, D remain tied to the same filter-basis data exposed by Q and F.

If the next entry is a structural transport or replay, use Definition 3862.6. Componentwise transport of $Q, F, L, W, R, D, E, H, C, P, N$ keeps the $\mathsf{Real}^{\uparrow}$ handoff downstream of the same finite basis and does not create an additional carrier coordinate.

The carrier of Definition 3862.1 has no row for quotient filters, ultrafilters, selected limits, choice schedules, host equality, or ambient completeness. Therefore the invariant holds for every finite ledger Λ. □

Theorem 3862.10 (FilterLimitBasis finite-prefix StreamName consumption)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C,$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Every accepted request that reaches the terminal row* E *first consumes the finite prefix StreamName windows* W*, the RegSeqRat readback row* R*, and the DyadicRatCore tolerance row* D *through the same* Q, F, L *basis route. Consequently a Real or completion-facing read cannot replace the displayed finite-prefix route by a quotient filter, a selected limit, an ambient completeness row, or a detached stream representative.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisFinitePrefixStreamConsumption`

Proof. Project the carrier of Definition 3862.1. The only route to E is the displayed chain through Q, F, L, W, R, D followed by the structural rows H, C, P, N.

Apply Definition 3862.8. It forces every Real-facing or completion-facing request to expose Q, F, L, W, R, D before the terminal seal is read.

Then apply Definition 3862.9. Finite replay of the request preserves the same basis route through every prefix of the ledger, so W, R, D remain tied to the same Cauchy-filter completion, Cauchy-filter, and limit-boundary rows.

Since Definition 3862.1 has no coordinate for quotient filters, selected limits, detached stream representatives, or ambient completeness, none of those objects can be introduced before the E handoff. □

Theorem 3862.11 (FilterLimitBasis real-window exhaustion)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier exposing the completion-basis row, limit-compatibility row,* $\mathsf{Real}^{\uparrow}$ *handoff-stability row, and finite-window induction row. Then every limit-basis consumer must expose a finite window tuple through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *before it may read terminal* $\mathsf{Real}^{\uparrow}$ *evidence. In particular, the consumer has no route through quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisRealWindowExhaustion`

Proof. First apply Definition 3862.3. It sends every completion-facing or Real-facing request through the displayed route $Q, F, L, W, R, D, E, H, C, P, N$, so the terminal seal is not available before the completion and filter rows have been read.

Next apply Definition 3862.4. The limit boundary L remains tied to the same finite window row W, regular readback row R, and dyadic tolerance row D exposed by Q and F.

Then use Definition 3862.6 and Definition 3862.9. Transport and finite replay preserve the same route for every ledger prefix, so the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows exhaust the visible window before terminal Real evidence is read.

The carrier definition has no row for quotient filters, selected limits, ultrafilters, countable-choice schedules, host equalities, or ambient completeness. Therefore any consumer using one of those objects has left the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ packet rather than reading its real-window exhaustion surface. □

Theorem 3862.12 (FilterLimitBasis obligation-closure readiness)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. The carrier admission of Definition 3862.1, the local NameCert obligations of Definition 3862.2, the CauchyFilterCompletion handoff of Definition 3862.3, the limit-compatibility row of Definition 3862.4, the ledger refusal of Definition 3862.5, the Real handoff stability of Definition 3862.6, the window-basis exhaustion of Definition 3862.8, the window induction of Definition 3862.9, and the real-window exhaustion of Definition 3862.11 jointly provide the paper-side obligation surface needed to move the chapter from seed closure toward obligation closure.*

The readiness surface is still finite BEDC data: it records only the displayed completion row, Cauchy-filter row, limit-boundary row, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise transport, replay, provenance, and local NameCert rows. It introduces no quotient filter, selected limit, ultrafilter, countable-choice schedule, host equality, ambient Real completeness, or bridge certificate.

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisObligationClosureReadiness`

Proof. Project the carrier row of Definition 3862.1. The only available coordinates are $Q, F, L, W, R, D, E, H, C, P, N$ so any obligation-closure package must be assembled from those finite rows.

Apply Definition 3862.2, Definition 3862.3, and Definition 3862.4. These rows supply carrier admission, the completion-facing route, and the condition that the limit boundary uses the same finite filter-basis data.

Next apply Definition 3862.5, Definition 3862.6, and Definition 3862.8. They keep the terminal Real seal downstream of the displayed basis and exclude quotient, choice, ultrafilter, host-equality, ambient-completeness, and bridge coordinates.

Finally use Definition 3862.9 and Definition 3862.11. Finite replay preserves the same window route through every ledger prefix, so the chapter has the paper-side obligation surface required for the next closure step without adding any object outside the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier. □

Theorem 3862.13 (FilterLimitBasis scope exactness)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier. Every consumer read exported by the local limit-basis surface factors through the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row Q, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row L, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the* $\mathsf{Real}^{\uparrow}$ *seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *row N before any limit-basis export is read. The scope contains no selected limit, quotient filter, ultrafilter, countable-choice schedule, host filter equality, host real equality, or ambient real-completeness premise.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopeExactness`

Proof. Read the carrier rows in order. The completion row Q and Cauchy-filter row F determine the finite filter-basis request, and the limit-boundary row L attaches the requested limit evidence to that same basis.

The finite-prefix material then fixes the observation route. By Definition 3862.10, the windows W, regular readback R, and dyadic tolerance ledger D are consumed before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be inspected.

The structural rows preserve exactness rather than adding new objects. Componentwise hsame transport in H and displayed Cont replay in C keep every consumer read on the same route, while P and N record only provenance and local naming.

Therefore any exported limit-basis read has passed through $Q, F, L, W, R, D, E, H, C, P, N$. The refused objects have no carrier coordinate and no replay edge, so selected limits, quotient filters, countable choice, and ambient real completeness cannot be smuggled into the scope. □

Theorem 3862.14 (FilterLimitBasis scoped obligation)**.** *For an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier $B = (Q, F, L, W, R, D, E, H, C, P, N)$, the obligation surface is scoped exactly by the displayed route through* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. Every completion-facing, limit-facing, or Real-facing read must pass through those rows, and no quotient filter, selected limit, ultrafilter, countable-choice schedule, host filter equality, host real equality, ambient completeness, or bridge certificate is part of the scope.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedObligation`

Proof. Start from the carrier of Definition 3862.1. It displays precisely the rows $Q, F, L, W, R, D, E, H, C, P, N$, so the candidate scope is already bounded by the completion boundary, Cauchy-filter surface, limit

boundary, window, readback, tolerance, Real seal, transport, replay, provenance, and local naming data.

Apply the NameCert obligations and the CauchyFilterCompletion handoff, Definition 3862.2 and Definition 3862.3. They force every completion-facing consumer through Q, F, L, W, R, D before E and keep H, C, P, N attached to the same finite route.

Use limit compatibility, window induction, and real-window exhaustion from Definition 3862.4, Definition 3862.9, and Definition 3862.11. These theorems preserve the same route for limit-facing and Real-facing reads, so the scoped obligation is exactly the displayed L10 completion route and contains no additional host or bridge coordinate. □

Theorem 3862.15 (FilterLimitBasis kernel scope upgrade)**.** *Every accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet routes completion-facing limit evidence through the kernel sequence*

$$\mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow},\ \mathsf{CauchyFilterLimit}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

and only then through BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *rows. In carrier notation, every accepted read is scoped by* $Q, F, L, W, R, D, E, H, C, P, N$ *before it is exported as limit-basis evidence.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisKernelScopeUpgrade`

Proof. Use the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the local obligation rows.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. A completion-facing consumer must pass through Q, F, L, W, R, D before the terminal seal E and the structural rows can be read.

The limit compatibility theorem and window-basis exhaustion theorem, Definition 3862.4 and Definition 3862.8, keep the $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows attached to the same displayed basis.

Finally use finite-prefix StreamName consumption from Definition 3862.10. The structural rows BHist, hsame, Cont, Pkg, and $\mathsf{NameCert}$ replay and name that route without adding quotient filters, selected limits, countable choice, host equality, ambient completeness, or bridge data. □

Theorem 3862.16 (FilterLimitBasis cofinal subbasis scope)**.** *Let* $B = (Q, F, L, W, R, D, E, H, C, P, N)$ *be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *carrier whose filter row* F *is generated from a displayed* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *route. Before* B *exports limit-basis evidence, the consumer must expose the subbasis finite intersections, the generated* $\mathsf{FilterBase}^{\uparrow}$ *row, the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The export is therefore scoped by the cofinal subbasis route together with* $Q, F, L, W, R, D, E, H, C, P, N$ *and it contains no quotient filter, selected limit, ultrafilter, countable-choice schedule, host filter equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge certificate.*

Proof. Start with the generated-base cofinality and finite-intersection route of Definition 4232.4 and Definition 4232.5. They force a subbasis-generated completion read through finite subbasis tokens, finite intersections, generated events, generation ledger, and a generated $\mathsf{FilterBase}^{\uparrow}$ row before any downstream Cauchy-filter row can be consumed.

Compose this with the Cauchy filter subbasis completion handoff of Definition 4232.7. The handoff keeps the generated base before the $\mathsf{CauchyFilter}^{\uparrow}$ row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Now apply the local scoped route of Definition 3862.15. Once the generated $\mathsf{CauchyFilter}^{\uparrow}$ row is the filter row F of the accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ carrier, the limit-basis read must continue through $Q, F, L, W, R, D, E, H, C, P, N$ in the kernel order displayed there.

The structural rows transport and name the same finite route. Since neither the subbasis carrier nor the limit-basis carrier has a coordinate for quotient filters, selected limits, ultrafilters, countable choice, host filter equality, ambient completeness, or bridge certificates, the composed scope is exactly the cofinal subbasis surface stated above. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCofinalSubbasisScope`

Theorem 3862.17 (FilterLimitBasis cofinal Cauchy-entourage induction)**.** *Let $B = (Q, F, L, W, R, D, E, H, C, P, N$ be an accepted* $\mathsf{FilterLimitBasis}^{\uparrow}$ *packet whose finite basis rows refine the Cauchy-filter route of* $\mathsf{CauchyFilter}^{\uparrow}$*, the completion boundary of* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*, the Real-induced uniformity of* $\mathsf{RealUniformStructure}^{\uparrow}$*, the regular rational readback of* $\mathsf{RegSeqRat}^{\uparrow}$*, and the source-window surface of* $\mathsf{StreamName}^{\uparrow}$*. Induction over the finite basis-window ledger produces a cofinal entourage route*

$$Q, F, L, W, R, D \longmapsto E \longmapsto H, C, P, N$$

such that every Real-facing limit read must first pass through the displayed Cauchy-filter basis, the completion boundary, the Real-uniform entourage row, the StreamName windows, the RegSeqRat readback, and the dyadic tolerance ledger before any terminal $\mathsf{Real}^{\uparrow}$ *seal is exposed. The route exports no selected limit, quotient filter, host filter equality, ultrafilter, countable-choice schedule, ambient Real completeness, or bridge certificate.*

Proof. Start with the finite ledger induction of Definition 3862.9. Its base case reads the displayed rows Q, F, L, W, R, D before E, and its step case preserves the same route under every finite window read or structural replay.

Use the real-window exhaustion theorem Definition 3862.11. It identifies the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as the visible Real-facing window route, so no terminal Real evidence is read before the basis windows and regular rational handoff are exposed.

Finally apply the kernel scope upgrade of Definition 3862.15. The upgraded scope places $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in one displayed kernel route. Since the accepted carrier has no coordinate for selected limits, quotient filters, host filter equality, ultrafilters, countable-choice schedules, ambient Real completeness, or bridge certificates, the induced entourage route is cofinal only inside that finite BEDC surface. □

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCofinalCauchyEntourageInduction`

Theorem 3862.18 (FilterLimitBasis scoped completion dependency)**.** *Every completion-facing* $\mathsf{FilterLimitBasis}^{\uparrow}$ *read consumes* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before a filter-limit basis can be sealed. In carrier notation, the read must expose the completion row Q, finite windows W, regular readback R, dyadic tolerance ledger D, terminal real seal E, componentwise transport H, displayed replay C, provenance P, and local naming N before any* $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$ *consumer treats the basis as sealed. The route does not claim selected limits, ultrafilters, countable choice, quotient filters, host filter equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisScopedCompletionDependency`

Proof. Project the carrier of Definition 3862.1. The completion-facing rows include $Q, W, R, D, E, H, C, P, N$, with the Cauchy-filter and limit rows attached by the same finite packet.

Apply the CauchyFilterCompletion handoff of Definition 3862.3. It forces the completion row and the finite-prefix StreamName consumption to appear before the terminal Real seal is inspected.

Use Definition 3862.10 and Definition 3862.11. These results route the windows through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the structural rows replay and name the basis.

The accepted carrier has no coordinate for selected limits, ultrafilters, countable choice, quotient filters, host filter equality, or ambient completeness. Hence the listed rows are the complete scoped dependency for the L10 completion handoff. □

Theorem 3862.19 (FilterLimitBasis public export)**.** *Every public completion-facing read of* $\mathsf{FilterLimitBasis}^{\uparrow}$ *factors through the already scoped route from* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *In carrier notation, the public export must expose* $Q, F, L, W, R, D, E, H, C, P, N$ *before a limit-basis consumer may read the seal. The export contains no quotient filters, ultrafilters, selected limits, countable-choice schedules, host equality, ambient Real completeness, or bridge certificate.*

Lean4 ✓ `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisPublicExport`
↪ variant `BEDC.Derived.FilterLimitBasisUp.FilterLimitBasisCarrier public export`

Proof. Use the NameCert obligations of Definition 3862.2. They make carrier admission, completion-basis exposure, limit compatibility, Real handoff, and ledger refusal the only local public rows.

Apply the handoff and limit-compatibility theorems, Definition 3862.3 and Definition 3862.4. These force every completion-facing read through Q, F, L, W, R, D before the terminal seal and keep the limit boundary attached to the same displayed filter-basis data.

Use Real handoff stability from Definition 3862.6 together with the scoped route of Definition 3862.15. Componentwise transport, displayed replay, provenance, and local naming preserve the same route through E, H, C, P, N and do not add a second public object.

Finally apply the scoped completion dependency of Definition 3862.18. Since that dependency already excludes selected limits, ultrafilters, countable choice, quotient filters, host equality, and ambient completeness, the public export is exactly the displayed finite route and carries no bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{FilterLimitBasis}^{\uparrow}$ read is refuted if a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reaches the terminal seal without displaying the Cauchy-filter completion row, Cauchy-filter row, Cauchy-filter limit row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming.

Independence witness: cauchyfiltercompletion, cauchyfilter, cauchyfilterlimit, streamname, regseqrat, dyadicratcore, real: none alone is bijective with $\mathsf{FilterLimitBasis}^{\uparrow}$ because the packet jointly records completion-boundary, saturated-filter, limit-boundary, finite-window, readback, tolerance, seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterLimitBasis}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, a CauchyFilter row, a CauchyFilterLimit row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FilterLimitBasis}^{\uparrow}}$, publicClosed)

Scope closed: Definition 3862.19, Definition 3862.17, and Definition 3862.18 close the public completion-facing export through CauchyFilterCompletion, CauchyFilter, CauchyFilterLimit, StreamName, RegSeqRat, DyadicRatCore, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FilterLimitBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, host filter equality, ultrafilter arguments, countable-choice schedules, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Bridge closure requires a standard bridge surface for the same finite public export route.

Chapter 3863

A Concrete Naming Certificate for $\mathsf{PerfectSet}^{\uparrow}$

Following Cantor and Bendixson, the $\mathsf{PerfectSet}^{\uparrow}$ packet names the finite certificate surface for a closed set whose every displayed point remains a derived point. It sits over the open-neighborhood discipline of chapter 88, the compact metric inspection rows of chapter 129, the terminal real-name seal of chapter 38, and the derivative-stage route of chapter 2024. The chapter isolates the reusable perfect-kernel face consumed by Cantor-Bendixson analysis; it is not a transfinite decomposition, a selected point enumeration, quotient real equality, or an ambient completeness theorem.

Definition 3863.1 (PerfectSet carrier)**.** *A* $\mathsf{PerfectSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, K, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the open-neighborhood tests and closed-set complement reads through chapter 88.*
2. *K is the compact-metric source row, carrying the finite metric inspection and cover data supplied by chapter 129.*
3. *D is the derived-point ledger. For every displayed point read from the set and every finite neighborhood request, it records a second compatible point read through the same topology and metric rows.*
4. *W is the finite window schedule for neighborhood, metric, and derived-point requests.*
5. *R is the regular rational readback row used by the metric comparisons and the non-isolation witnesses.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the topology, compact-metric, derived-point, window, and readback rows have been displayed.*
7. *H records componentwise* hsame *transport for T, K, D, W, R, E and for the structural replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of every perfect-set consumer request through*

$$T, K, D, W, R \longmapsto E \longmapsto H, C, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{PerfectSet}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CantorBendixson}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{PerfectSet}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected countable dense subset, transfinite rank, isolated-point enumeration, quotient equality of real names, host closed-set equality, countable choice, or ambient compactness beyond the displayed rows.

Theorem 3863.2 (PerfectSet NameCert obligations)**.** *For every accepted* $\mathsf{PerfectSet}^{\uparrow}$ *carrier* $S = (T, K, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PerfectSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topology, compact-metric, derived-point, finite-window, regular readback, Real-seal, transport, replay, provenance, and local naming rows;*
2. *closed-set exposure, requiring complement and neighborhood requests to pass through the topology row T and compact-metric row K;*
3. *derived-point non-isolation, requiring each displayed point-and-neighborhood request to receive its second compatible point through D at a window listed in W;*
4. *real-seal compatibility, requiring every metric comparison and non-isolation witness to pass through R before the terminal seal E is consumed;*
5. *ledger nonescape, requiring H, C, P, N to export no isolated-point selector, transfinite derivative rank, quotient real equality, host closed-set equality, countable-choice schedule, or ambient completeness theorem.*

No perfect-set obligation is stored outside these rows.

Proof. Project the accepted carrier $S = (T, K, D, W, R, E, H, C, P, N)$. Carrier admission exposes exactly the topology row, compact-metric source, derived-point ledger, finite window schedule, regular readback row, Real seal, and the structural rows.

Closed-set exposure is read through T and K. The topology row supplies neighborhood and complement tests, while the compact-metric row supplies the finite metric inspection data used by the same requests.

The derived-point clause is the row D. For each displayed point-and-neighborhood request, the finite window row W names the inspected neighborhood and D records the compatible second point before any Real-facing seal is available.

Finally, R and E bind the metric and Real-facing parts of the route, while H, C, P, N transport, replay, source, and name the same packet. Since the excluded classical data are not carrier coordinates, the five listed rows exhaust the local certificate. □

Theorem 3863.3 (PerfectSet derived-point nonescape)**.** *Every* $\mathsf{CantorBendixson}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{PerfectSet}^{\uparrow}$ *packet must read the topology row, compact-metric row, derived-point ledger, finite window schedule, regular readback route, and terminal Real seal before treating the set as a perfect kernel. In particular, the consumer receives the finite route*

$$T, K, D, W, R \longmapsto E \longmapsto H, C, P, N$$

and receives no isolated-point selector, transfinite derivative rank, quotient real equality, host closed-set equality, countable-choice schedule, or ambient compactness theorem outside the displayed certificate.

Proof. Start with a consumer request for the perfect-kernel face. By Definition 3863.2, the request first passes through the topology and compact-metric rows T, K and then through the derived-point ledger D at a finite window listed in W.

The regular readback row R and terminal seal E are the only Real-facing route in the carrier. Thus a Cantor-Bendixson consumer can consume this packet only as the retained perfect-set face displayed by chapter 2024.

The structural rows H, C, P, N preserve and replay the same route. Because the carrier lacks coordinates for isolated-point selection, transfinite ranks, quotient equality, host closed-set equality, countable choice, or extra compactness principles, no such data can escape through the derived-point ledger. □

Lean4 ✓ `BEDC.Derived.PerfectSetUp.PerfectSetCarrier derived point nonescape`

Closure status for $\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$.

Constructive story: A $\mathsf{PerfectSet}^{\uparrow}$ packet is generated from topology rows, compact-metric inspection rows, a derived-point ledger, finite request windows, regular rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$, seed)

Scope closed: Definition 3863.1, Definition 3863.2, and Definition 3863.3 seed the finite perfect-kernel certificate consumed by Cantor-Bendixson and Real-facing routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PerfectSet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close transfinite Cantor-Bendixson decomposition, selected isolated-point enumeration, quotient real equality, host closed-set equality, countable choice, ambient compactness, Lean theorem coverage, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closed-set exposure, derived-point non-isolation, Real-seal compatibility, ledger nonescape, and Cantor-Bendixson consumer routing for BEDC.Derived.PerfectSetUp.

Chapter 3864

A Concrete Naming Certificate for $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ packet records the finite even–odd interleaving route by which two regular Cauchy rational names are read as one binary stream before any Real or Completion consumer treats the result as a sealed real-facing name. It sits over the finite stream surface of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion boundary of chapter 1569. The packet names a pairing-and-projection ledger; it does not identify host streams, select quotient representatives, or import ambient completeness.

Definition 3864.1 (Regular Cauchy binary interleaving carrier)**.** *A* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two finite* $\mathsf{StreamName}^{\uparrow}$ *source rows whose regular rational observations are to be paired.*
2. E *is the even* $\mathsf{StreamName}^{\uparrow}$ *projection row reading the* S_0 *observations from the interleaved stream.*
3. O *is the odd* $\mathsf{StreamName}^{\uparrow}$ *projection row reading the* S_1 *observations from the interleaved stream.*
4. T *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, assigning the same dyadic budget to the two projections and to the interleaved read.*
5. R_0 *and* R_1 *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the even and odd projections after the dyadic budget is fixed.*
6. L *is the* $\mathsf{Real}^{\uparrow}$ *seal compatibility row, requiring the two projected regular names and the interleaved name to expose the same terminal seal discipline to Real and Completion consumers.*
7. H *records componentwise* hsame *transport for the two source rows, projection rows, tolerance ledger, regular readbacks, Real seal row, replay, provenance, and naming rows.*
8. K *records displayed* Cont *replay from an interleaving request through* $S_0, S_1, E, O, T, R_0, R_1, L, H, P, N$.

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N$. It has no coordinate for host stream equality, quotient sequence equality, a selected completion representative, countable-choice data, or an ambient real-completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyBinaryInterleavingUp.RegularCauchyBinaryInterleavingTasteGate`
`single carrier alignment`

Theorem 3864.2 (Regular Cauchy binary interleaving NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *carrier $B = (S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$ *surface is exhausted by five rows:*

1. *carrier admission: the accepted object is exactly the displayed packet $S_0, S_1, E, O, T, R_0, R_1, L, H, K, P, N$;*
2. *projection stability: the even row E and odd row O read the two source* $\mathsf{StreamName}^{\uparrow}$ *rows without changing their finite windows;*
3. *dyadic tolerance preservation: the ledger T assigns one dyadic tolerance discipline to the two projections and to the interleaved stream;*
4. *Real seal compatibility: the row L is reachable only after E, O, T, R_0, R_1 have been displayed;*
5. *ledger refusal: H, K, P, N export no quotient sequences, choice selectors, host stream equality, hidden completion selector, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyBinaryInterleavingUp.RegularCauchyBinaryInterleavingNameCertObligatio`

Proof. Project the carrier of Definition 3864.1. Carrier admission exposes exactly the two sources, two projection rows, one dyadic tolerance ledger, two regular rational readbacks, one Real seal compatibility row, and the structural rows.

Read the projection rows next. The even row E is tied to S_0, and the odd row O is tied to S_1, so a consumer sees projection stability as a finite StreamName route rather than as host equality of streams.

The tolerance and readback rows then fix the regular Cauchy surface. The same dyadic ledger T is used before R_0 and R_1 are consumed, and the Real seal compatibility row L is available only after those rows have been displayed.

Finally inspect H, K, P, N. These rows transport, replay, source, and name the same finite packet, and none adds quotient sequences, choice selectors, host stream equality, hidden completion selectors, or ambient completeness data. Thus the five obligations exhaust the local NameCert surface. □

Theorem 3864.3 (Regular Cauchy binary interleaving seal stability)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ *packet factors through*

$$S_0, S_1 \longmapsto E, O \longmapsto T \longmapsto R_0, R_1 \longmapsto L \longmapsto H, K, P, N.$$

In particular, the consumer cannot read the terminal $\mathsf{Real}^{\uparrow}$ *seal compatibility row L without first displaying the even–odd projection rows, the shared dyadic tolerance ledger, and the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows.*

Proof. Begin with a Real or Completion request. The only source rows in the carrier are S_0 and S_1, and the only ways to read them through the interleaved stream are the even and odd projection rows E and O.

Next apply the tolerance ledger. The row T fixes the dyadic budget shared by the two projected reads, so the regular rational rows R_0 and R_1 cannot be consumed as independent selected sequences.

The seal row L is downstream of those displayed rows. Since the structural rows H, K, P, N only transport and name the same route, any consumer reaching a Real or Completion conclusion without the projection, tolerance, and readback rows would ask for a coordinate outside the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ read is refuted if a Real or Completion consumer reaches the terminal seal compatibility row without displaying both StreamName source rows, even and odd projection rows, the shared DyadicRatCore tolerance ledger, both RegSeqRat readbacks, transport, replay, provenance, and local naming.

Independence witness: streamname, dyadicratcore, regseqrat, real, cauchyfiltercompletion: none alone is bijective with $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ because the packet jointly records two source streams, even and odd projection rows, one dyadic tolerance ledger, two regular rational readbacks, a terminal Real seal compatibility row, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}$ packet is generated from two StreamName source rows, even and odd projection rows, a DyadicRatCore tolerance ledger, two RegSeqRat readback rows, a Real seal compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3864.1, Definition 3864.2, and Definition 3864.3 seed the finite binary interleaving route for regular Cauchy rational names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinaryInterleaving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient sequences, host stream equality, choice selectors, selected completion representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projection stability, dyadic tolerance preservation, Real seal compatibility, ledger refusal, seal stability, and local naming for BEDC.Derived.RegularCauchyBinaryInterleavingUp.

Chapter 3865

A Concrete Naming Certificate for ??

Following Bishop and Bridges on nested closed intervals, the **??** packet records the constructive point-name extracted from a displayed shrinking chain of closed bounded intervals. It sits after the endpoint carrier of chapter 1591, the retained-chain surface of chapter 320, the intersection route of chapter 389, and the Cantor-style singleton route of chapter 1421. The packet names the point by finite BHist rows over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it does not choose a host real, quotient representative, open-cover compactness proof, or countable-choice schedule.

Definition 3865.1 (Nested interval point-name carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$P = (I, J, K, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint-and-containment source row for each retained interval window;*
2. *J is the* $\mathsf{NestedInterval}^{\uparrow}$ *chain row, recording inclusion of the next retained interval inside the preceding one;*
3. *K is the* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *and* $\mathsf{CantorIntersection}^{\uparrow}$ *singleton-intersection route row;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule used to read endpoints and diameter bounds at requested precision;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from the retained endpoint windows;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the named point, reached only after I, J, K, S, R have been displayed;*
7. *H records componentwise* hsame *transport for the interval source, nested chain, intersection row, StreamName windows, RegSeqRat readback, and Real seal;*
8. *C records displayed* Cont *replay from closed bounded intervals through nested containment, intersection, finite windows, regular readback, and the Real seal;*
9. *Q records* Pkg *provenance over **??**,* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier has no coordinate for an ambient intersection set, a selected quotient representative, countable choice, open-cover compactness, or an unbounded completeness theorem.

Lean4 ✓ BEDC.Derived.NestedIntervalPointNameUp.NestedIntervalPointNameTasteGate single carrier alignment

Theorem 3865.2 (Nested interval point-name regular Cauchy seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted ?? carrier factors through the displayed route*

$\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}$

In particular, the regular rational readback row R and the terminal seal row E are downstream of the interval source I, nested-chain row J, singleton-intersection row K, and finite schedule row S.

Proof. Project the carrier of Definition 3865.1. The rows I, J, K are the source, nesting, and intersection rows; the rows S, R, E are the finite schedule, regular readback, and terminal Real seal.

Use the closed bounded interval source route from chapter 1591. That route exposes endpoint, containment, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows as finite BHist data before any point-facing consumer can read them.

Now apply the nested and Cantor intersection siblings. The $\mathsf{NestedInterval}^{\uparrow}$ and $\mathsf{NestedIntervalIntersection}^{\uparrow}$ rows retain the chain and intersection route, while $\mathsf{CantorIntersection}^{\uparrow}$ supplies the singleton-facing seal. Since H, C, Q, N only transport, replay, source, and name these displayed rows, the exported $\mathsf{Real}^{\uparrow}$ read factors through the stated route. □

Theorem 3865.3 (Nested interval point-name endpoint containment)**.** *For every accepted ?? carrier $P = (I, J, K, S, R, E, H, C, Q, N)$, every finite read of the named point remains contained in each displayed retained* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *window before the* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The containment read is transported componentwise by H and replayed by C through the same* $\mathsf{NestedInterval}^{\uparrow}$ *and* $\mathsf{NestedIntervalIntersection}^{\uparrow}$ *chain.*

Proof. Start with the interval source row I. By the carrier, I is a closed bounded interval packet with lower endpoint, upper endpoint, endpoint-order, rational containment, dyadic containment, StreamName, RegSeqRat, and Real-seal rows.

Next use the nested-chain row J. It records that each retained window is included in the preceding one, so a later finite read cannot leave an earlier displayed closed bounded interval without breaking the hsame transport recorded in H.

Finally use the intersection row K. The $\mathsf{NestedIntervalIntersection}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ route supplies the point-facing intersection witness, while C replays the same containment route into S, R, E. Hence endpoint containment is preserved before the named point reaches the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 3865.4 (Nested interval point-name uniqueness route)**.** *If two accepted ?? carriers are classified together by their displayed* hsame *transport rows, then their* $\mathsf{Real}^{\uparrow}$ *point-name seals agree only through the same closed-interval source, nested-chain, intersection, StreamName, and RegSeqRat rows. The classifier exposes no separate host equality of points and no quotient-real representative selection.*

Proof. Compare the two carriers componentwise. The classifier can transport the interval source, nested chain, intersection row, finite schedule, regular readback, and Real seal only along displayed hsame rows.

Apply the endpoint-containment theorem. Each finite read remains inside the same transported retained intervals before the Real seal is consumed, so the uniqueness route is pinned to the displayed closed bounded interval and nested-intersection data.

The terminal comparison is therefore a route comparison, not a host equality principle. Since the carrier has no coordinate for quotient real equality or representative choice, the only available uniqueness evidence is the transported $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{NestedIntervalIntersection}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route named by the packet. □

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from displayed ClosedBoundedIntervalUp interval rows, a NestedIntervalUp retained-chain row, NestedIntervalIntersectionUp and CantorIntersectionUp intersection rows, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{??}$, seed)

Scope closed: Definition 3865.1, Definition 3865.2, Definition 3865.3, and Definition 3865.4 seed the finite point-name route extracted from nested closed bounded intervals.

Formal verification: FormalStatus($\mathsf{NameCert}_{??}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host interval intersection, selected quotient representatives, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular Cauchy seal routing, endpoint containment, uniqueness routing, transport, replay, provenance, and local naming for the displayed point-name packet.

Chapter 3866

A Concrete Naming Certificate for $\mathsf{HyperbolicVisualMetric}^{\uparrow}$

The $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ packet realizes the disk readout branch of chapter 9463 as finite NameCert data. It sits after the Poincare disk metric surface of chapter 3474, the boundary-geodesic route of chapter 3491, the boundary-shadow packet of chapter 3813, and the boundary-fixed transport surface of chapter 3500. The packet names a simultaneous visual-metric read over displayed disk, metric, boundary, shadow, transport, and provenance rows; it does not turn the observer boundary into an object-level T or a completed global geometry.

Definition 3866.1 (Hyperbolic visual metric carrier)**.** *A* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (D, M, B, S, T, F, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk source row, read through the Poincare disk and unit-disk coordinate surface;*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row over the same disk source;*
3. *B is the* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ *boundary row for geodesic far-end incidence;*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *shadow row attached to the displayed boundary read;*
5. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *classifier-preserving boundary transport row;*
6. *F is the global-geometry refusal row, rejecting reads that replace displayed disk, metric, boundary, shadow, and transport rows by an object-level geometry;*
7. *A is the vision anchor row tying the packet to the observer-expansion reading of chapter 9463;*
8. *H records componentwise* hsame *transport for disk source, metric read, boundary row, shadow row, transport row, refusal row, and anchor row;*
9. *C records displayed* Cont *replay from disk source through metric read, boundary read, shadow, transport, refusal, and provenance;*
10. *P records* Pkg *provenance over* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed boundary compactification, object-level T*, physical horizon, global metric oracle, or quotient boundary equality.*

Lean4 (definition) `BEDC.Derived.HyperbolicVisualMetricUp.HyperbolicVisualMetricUp`
Lean4 ✓ `BEDC.Derived.HyperbolicVisualMetricUp.HyperbolicVisualMetricTasteGate single carrier alignment`

Theorem 3866.2 (Hyperbolic visual metric obligation surface)**.** *For the carrier of Definition 3866.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$ *surface consists exactly of disk source admission, hyperbolic metric read admission, geodesic-boundary read admission, boundary-shadow admission, classifier-preserving boundary transport, global-geometry refusal, vision-anchor provenance, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier rows. The disk source D, metric read M, boundary row B, shadow row S, and transport row T are the only geometric data available to the packet.

Use the sibling surfaces in order. $\mathsf{HyperbolicMetric}^{\uparrow}$ supplies the metric read, $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ supplies the boundary incidence route, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ supplies the shadow row, and $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ supplies classifier-preserving transport.

The refusal and structural rows finish the surface. The row F refuses any read that bypasses the displayed siblings, while A, H, C, P, N only anchor, transport, replay, source, and name the same finite route. Thus the listed rows exhaust the local certificate. □

Theorem 3866.3 (Hyperbolic visual metric global-geometry refusal)**.** *A visual metric read of an accepted* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *packet is refused unless the displayed route contains disk source, metric read, boundary row, boundary shadow, classifier-preserving transport, provenance, and the global-geometry refusal row. The packet cannot replace those rows by an object-level* T*, completed boundary compactification, physical horizon, or global metric oracle.*

Proof. Read the geometric rows first. The carrier exposes disk source D, hyperbolic metric row M, geodesic-boundary row B, boundary-shadow row S, and boundary-fixed transport row T as separate finite fields.

Apply the obligation surface. By Definition 3866.2, a local NameCert read can inspect only those rows plus refusal, anchor, transport, replay, provenance, and local naming.

The excluded objects have no carrier coordinate. Since F explicitly refuses global-geometry substitution and H, C, P, N only preserve the displayed route, no accepted read can produce object-level T, completed compactification, physical horizon, or a metric oracle outside the packet. □

Theorem 3866.4 (Hyperbolic visual metric sibling separation)**.** *The* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *carrier is not bijective, row for row, with any one of* $\mathsf{HyperbolicMetric}^{\uparrow}$*,* $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$*,* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$*,* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*, or* $\mathsf{DynSystem}^{\uparrow}$*. A bijection preserving local NameCert obligations would have to preserve simultaneous disk readout, metric read, boundary incidence, shadow, boundary-fixed transport, refusal, provenance, transport, replay, and local naming, and no listed sibling carries that whole surface.*

Proof. Compare first with $\mathsf{HyperbolicMetric}^{\uparrow}$. The metric sibling supplies a disk metric read, but it does not carry boundary-geodesic incidence, boundary shadow, boundary-fixed transport, and global-geometry refusal as simultaneous rows.

Compare next with $\mathsf{HyperbolicGeodesicBoundary}^{\uparrow}$ and $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$. These siblings supply boundary incidence and shadow reads, while neither also owns the metric read, boundary-fixed transport, and visual refusal row.

Finally compare with $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ and $\mathsf{DynSystem}^{\uparrow}$. Transport and flow rows can move or evolve displayed data, but they are not the full visual metric packet. Since each sibling omits at least one of D, M, B, S, T, F, A, no row-preserving bijection can preserve the $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ obligations. □

Falsifiable prediction: A visual metric read is refused whenever the packet lacks disk source, metric read, boundary shadow, transport, and provenance rows.

Independence witness: hyperbolic_metric, hyperbolic_geodesic_boundary, hyperbolic_boundary_shadow, poincare_boundary_fixed_transport, dynsystem: the visual metric packet carries the simultaneous readout route not bijective with any one sibling.

The Lean TasteGate site for $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ HyperbolicVisualMetric}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicVisualMetric}^{\uparrow}$ instances. These instances keep disk source, metric read, boundary row, shadow row, boundary-fixed transport, refusal, provenance, transport, replay, and local naming as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ packet is generated from a disk source row, HyperbolicMetricUp metric read, HyperbolicGeodesicBoundaryUp boundary row, HyperbolicBoundaryShadowUp shadow row, PoincareBoundaryFixedTransportUp transport, global-geometry refusal, vision-anchor provenance, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3866.1, Definition 3866.2, Definition 3866.3, and Definition 3866.4 seed the finite visual-metric route over disk, metric, boundary, shadow, transport, refusal, and provenance rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close object-level T, completed boundary compactification, physical horizons, global metric oracles, quotient boundary equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual metric obligation routing, global-geometry refusal, sibling separation, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming.

Chapter 3867

A Concrete Naming Certificate for $\mathsf{MetricCauchyNetCluster}^{\uparrow}$

Following Moore–Smith and Kelley net convergence, and the Bishop–Bridges treatment of metric completion by displayed Cauchy information, the $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ packet records the finite route by which a metric Cauchy net exposes cluster evidence before any completion or real-seal consumer reads it. It sits between the subnet route of chapter 1592, the Cauchy-net completion boundary of chapter 3794, the metric completion packet of chapter 1506, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a displayed cluster route for completion-facing reads; it is not quotient nets, selected cluster points, countable choice, host limit equality, or ambient completeness.

Definition 3867.1 (Metric Cauchy-net cluster carrier)**.** *A* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, M, J, S, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-net row, recording the inspected finite directed indices and the local comparison order used by the Moore–Smith route.*
2. *W is the Cauchy-window row, listing the finite cofinal windows and the tolerance requests that must be met before any cluster read is exposed.*
3. *M is the metric completion boundary row, connecting the displayed net windows to the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer surfaces.*
4. *J is the cluster witness row, recording only finite witness windows and not a selected cluster point.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that opens the finite observation windows used by W and J.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational distance and tolerance comparisons inspected along the route.*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and readbacks.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the directed-net, Cauchy-window, cluster-witness, stream, regular-readback, dyadic, and metric-completion rows have been displayed.*

9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a completion-facing request through $D, W, M, J, S, R, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient nets, chosen cluster points, choice-selected subnets, host metric completion equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricCauchyNetClusterUp.MetricCauchyNetClusterTasteGate single carrier alignment`

Theorem 3867.2 (Metric Cauchy-net cluster NameCert obligations)**.** *For every accepted* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier $K = (D, W, M, J, S, R, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the directed-net row, Cauchy-window row, metric-completion boundary, cluster witness, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-window stability, requiring every inspected directed index to be read through D before W can answer a Cauchy request;*
3. *cluster-witness locality, requiring J to expose only finite witness windows tied to the same D, W, S, T rows;*
4. *completion-boundary handoff, requiring M to mediate every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer before E is read;*
5. *regular and dyadic accounting, requiring R and T to record the rational distance and tolerance ledger used by the same finite windows;*
6. *structural exactness through H, C, P, N, with no quotient-net equality, selected cluster point, countable choice, host real equality, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 3867.1. The packet exposes the directed-net row, Cauchy-window row, metric boundary, cluster witness, finite windows, regular readback, dyadic ledger, terminal seal, and the structural rows as one finite object.

The directed and Cauchy-window obligations are read first. The row D fixes the finite directed comparisons, and W answers tolerance requests only through those displayed comparisons and the finite stream windows S.

The cluster witness row J is then local to the same windows. It records witness windows rather than a selected point, while R and T record the rational distance readback and dyadic tolerance ledger used by the window.

The completion boundary M mediates the completion-facing read before the Real seal row E is available. Since H, C, P, N only transport, replay, source, and name the same rows, the listed obligations exhaust the NameCert surface. □

Theorem 3867.3 (Metric Cauchy-net cluster completion handoff)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ *carrier $K = (D, W, M, J, S, R, T, E, H, C, P, N)$ factors*

through the displayed finite Cauchy-net cluster route

$$D \longmapsto W \longmapsto S, J \longmapsto R, T \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

In particular, no $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *seal can be consumed before the directed-net row, Cauchy-window row, finite cluster witness,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger have been displayed.*

Proof. Begin with a completion-facing consumer. By Definition 3867.1, the only completion boundary in the packet is the row M, and the only terminal real-facing seal is E.

Apply the obligation surface of Definition 3867.2. It places directed-net admission and Cauchy-window stability before any cluster witness is read, and it ties the witness row to the same finite stream windows.

The regular rational and dyadic rows then supply the metric tolerance accounting used by the finite windows. The completion boundary M is downstream of that accounting, and the seal row E is downstream of M.

Finally H, C, P, N preserve and name only the displayed route. Quotient nets, selected cluster points, countable choice, host real equality, detached completion objects, and ambient completeness have no carrier coordinate, so they cannot replace the finite route. □

Falsifiable prediction: Every accepted $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ completion-facing read factors through a directed-net row, Cauchy-window row, finite cluster witness, StreamName windows, RegSeqRat readback, DyadicRatCore ledger, MetricCompletion and CauchyNetCompletion boundary, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchysubnet, cauchynetcompletion, metriccompletion, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective with $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ because this packet jointly carries the directed-net row, Cauchy-window row, finite cluster witness, metric completion boundary, finite stream windows, rational readback, dyadic ledger, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCauchyNetCluster}^{\uparrow}$ packet is generated from a directed-net row, Cauchy-window row, metric completion boundary, finite cluster witness, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3867.1, Definition 3867.2, and Definition 3867.3 seed the finite Cauchy-net cluster route from directed windows through completion and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCauchyNetCluster}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected cluster points, countable choice, host real equality, detached completion objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-window stability, cluster-witness locality, completion-boundary handoff, regular and dyadic accounting, structural exactness, and local naming for BEDC.Derived.MetricCauchyNetCl

Chapter 3868

A Concrete Naming Certificate for $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ packet records the constructive interval route from closed bounded interval data to a compact-uniform modulus extraction. It joins the endpoint surface of chapter 1591, the finite interval-net route of chapter 1250, the positive radius ledger of chapter 452, the compact-continuity rows of chapter 388, and the output consumer of chapter 369. The packet names the finite bridge data only; it does not assert open-cover compactness, countable choice, quotient stream equality, ambient real completeness, Lean verification, or bridge checking.

Definition 3868.1 (Compact interval modulus bridge carrier)**.** *A* $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, N, R, U, M, S, Q, H, C, P, L)$$

with the following displayed rows:

1. *E is the closed bounded interval endpoint row from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*, including the lower endpoint, upper endpoint, and displayed containment boundary.*
2. *N is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite-net row, carrying dyadic cells, rational cell readback, displayed centers, finite coverage,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows.*
3. *R is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius row, read only as a finite positive-radius ledger attached to the displayed net and cover route.*
4. *U is the* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *compact-continuity row whose pointwise continuity data are already routed through the interval net and radius ledger.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row consumed after E, N, R, U have been displayed.*
6. *S records the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows used by interval endpoints, net centers, and the final modulus handoff.*
7. *Q is the tolerance-query row whose requested precision is routed through the net, radius, compact-continuity, and output-modulus rows.*
8. *H records componentwise* hsame *transport for $E, N, R, U, M, S, Q, C, P, L$.*

9. *C records displayed* Cont *replay from the closed interval source through finite nets, finite-radius data, compact-continuity rows, and the uniform-modulus output.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNe}$ $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, countable choice, quotient real equality, hidden host continuity, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CompactIntervalModulusBridgeUp.CompactIntervalModulusBridgeTasteGate`
`single carrier alignment`

Theorem 3868.2 (Compact interval modulus bridge NameCert obligations)**.** *For the carrier of Definition 3868.1, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$ *surface consists exactly of endpoint exposure through E, finite interval-net exposure through N, finite positive-radius exposure through R, compact-continuity exposure through U, uniform-modulus output through M,* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal exposure through S, tolerance-query routing through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the accepted carrier $B = (E, N, R, U, M, S, Q, H, C, P, L)$. The endpoint row E supplies the closed bounded interval source, while N supplies the finite interval-net route over the same displayed endpoint boundary.

Read the finite-radius and compact-continuity rows next. The row R is the positive-radius ledger supplied by the finite Lebesgue-number surface, and U is the compact-continuity row that consumes the interval net and radius data before any modulus output is exposed.

The modulus row M is therefore downstream of E, N, R, U, Q. The seal row S records the finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ readbacks used by endpoints, centers, and the final handoff.

Finally H, C, P, L transport, replay, package, and name exactly these rows. Since the carrier has no additional coordinate, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: Every $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ uniform-modulus read must pass through displayed closed bounded interval endpoints, a finite closed-bounded interval net, a finite Lebesgue-number radius row, compact-uniform-continuity rows, StreamName, RegSeqRat, and RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: closedboundedinterval, closedboundedintervalnet, finitelebesguenumber, compactuniformcontinuity, uniformmodulus: $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, finite-net rows, finite positive-radius rows, compact-continuity rows, the UniformModulus output, StreamName/RegSeqRat/RealUp seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalModulusBridge}^{\uparrow}$ packet is generated from closed bounded interval endpoints, finite interval-net rows, a finite Lebesgue-number radius ledger, compact-continuity rows, a UniformModulus output row, StreamName, RegSeqRat, and RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactIntervalModulusBridge}^{\uparrow}}$, seed)

Scope closed: Definition 3868.1 and Definition 3868.2 seed the finite bridge from closed bounded interval data to compact-uniform modulus extraction.

Formal verification: FormalStatus(NameCert$_{\text{CompactIntervalModulusBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint exposure, finite-net exposure, radius exposure, compact-continuity exposure, uniform-modulus output, seal rows, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalMo

Chapter 3869

A Concrete Naming Certificate for $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ packet records the finite search surface obtained by iterating bisection over a located real interval while keeping every branch, sign read, and retained window visible. It sits after the real-name surface of chapter 38, the located interval carrier of chapter 1709, the located bisection route of chapter 1711, the finite bracketing interval packet of chapter 1807, and the dyadic intermediate-value packet of chapter 2203. The packet names a bounded bisection search, not a host interval equality, not trichotomy on arbitrary reals, not choice of a hidden root, and not ambient completeness.

Definition 3869.1 (Located real bisection-search carrier)**.** *A* $\mathsf{LocatedRealBisectionSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, B, F, D, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, exposing lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, a locatedness modulus, and the inspected finite interval budget.*
2. *B is the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *branch route, recording the requested finite bisection depth and the retained child at each stage.*
3. *F is the* $\mathsf{FiniteBracketingInterval}^{\uparrow}$ *bracket row, preserving the displayed sign or locator bracket for the retained interval.*
4. *D is the* $\mathsf{DyadicIntermediateValue}^{\uparrow}$ *dyadic sign-read row used by the bounded search.*
5. *Q is the query row, recording the finite search request, target tolerance, and admissible stopping condition.*
6. *W is the retained-window ledger, listing the lower and upper windows produced by the bounded branch sequence.*
7. *R is the* $\mathsf{Real}^{\uparrow}$*-facing seal row reached only after I, B, F, D, Q, W have been exposed.*
8. *H records componentwise* hsame *transport for the source, branch, bracket, dyadic read, query, window, and seal rows.*
9. *C records displayed* Cont *replay from a search request through finite bisection, bracket preservation, dyadic sign readback, retained-window exposure, and Real-facing seal.*

10. *P is the* Pkg *provenance over* LocatedRealBisectionSearch$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, LocatedIntervalBisection$^{\uparrow}$, FiniteBracketingInterval$^{\uparrow}$, DyadicIntermediateValue$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{LocatedRealBisectionSearch}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $I, B, F, D, Q, W, R, H, C, P, N$. It has no coordinate for host interval equality, trichotomy, countable choice, selected quotient representatives, ambient Real$^{\uparrow}$ *completeness, or a hidden root witness.*

Lean4 ✓ `BEDC.Derived.LocatedRealBisectionSearchUp.LocatedRealBisectionSearchTasteGate single carrier alignment`

Theorem 3869.2 (Located real bisection-search NameCert obligations)**.** *For every accepted* LocatedRealBisectionS *carrier $S = (I, B, F, D, Q, W, R, H, C, P, N)$, the local* NameCert$_{\text{LocatedRealBisectionSearch}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, located-source exposure through I, finite bisection exposure through B, bracket preservation through F, dyadic sign readback through D, bounded query admission through Q, retained-window exposure through W, Real-facing seal routing through R, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3869.1. The accepted packet contains only the displayed rows, so carrier admission fixes the source interval, branch route, bracket row, dyadic sign readback, query, retained windows, seal, and structural data before any consumer reads the search.

The analytic part of the route is finite. The located source is I, the bisection route is B, the bracket row is F, and the dyadic sign-read row is D. The query row Q fixes the bounded stopping condition, while W records only the retained windows produced by those rows.

Finally read R, H, C, P, N. These rows seal the route toward Real$^{\uparrow}$, transport the displayed data componentwise, replay the same finite request, record provenance, and name the packet. Since no carrier coordinate stores host equality, trichotomy, choice, ambient completeness, or a hidden root witness, the listed rows exhaust the local NameCert surface. □

Theorem 3869.3 (Located real bisection-search finite route)**.** *Every* Real$^{\uparrow}$ *or interval-search consumer of an accepted* LocatedRealBisectionSearch$^{\uparrow}$ *packet factors through the finite route*

$$\text{LocatedRealInterval}^{\uparrow} \longmapsto \text{LocatedIntervalBisection}^{\uparrow} \longmapsto \text{FiniteBracketingInterval}^{\uparrow} \longmapsto \text{DyadicIntermediateValue}^{\uparrow} \longmapsto$$

The consumer receives only the bounded branch ledger, preserved bracket, dyadic sign reads, retained windows, Real-facing seal, transport, replay, provenance, and local naming.

Proof. Begin with the source row. A search request first reads I, so the packet cannot replace the located real interval by an ambient host interval.

Next read the finite search rows B, F, D, Q, W. They record the branch sequence, bracket preservation, dyadic sign readback, bounded stopping request, and retained windows. No row in this stage selects an infinite limit or an unbounded root.

The final route through R, H, C, P, N preserves the same finite ledger for downstream real and interval-search consumers. Because the carrier has no coordinate for host interval equality, trichotomy, quotient-real representatives, countable choice, ambient completeness, or hidden root selection, every such consumer remains on the displayed finite route. □

Closure status for NameCert$_{\text{LocatedRealBisectionSearch}^{\uparrow}}$.

Constructive story: A LocatedRealBisectionSearch$^{\uparrow}$ packet is generated from a located real interval source row, located bisection branch route, finite bracketing row, dyadic sign readback, bounded query row, retained-window ledger, Real-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedRealBisectionSearch}\uparrow}$, seed)

Scope closed: Definition 3869.1, Definition 3869.2, and Definition 3869.3 seed the finite bisection-search packet over located real interval, finite bracket, dyadic sign, retained-window, and Real-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedRealBisectionSearch}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host interval equality, real trichotomy, countable choice, selected quotient representatives, ambient RealUp completeness, hidden root selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, branch-route exactness, bracket preservation, dyadic sign readback, retained-window routing, RealUp seal nonescape, and local naming for BEDC.Derived.LocatedRealBisectionSearchUp.

Chapter 3870

A Concrete Naming Certificate for $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$

The $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ packet realizes the conditional trigger ensemble of Definition 9462.1594 and Definition 9462.1601 as finite weighted trigger, blocker, partition, expectation, transport, replay, provenance, and local NameCert rows. It sits beside the residue boundary of chapter 1167, the trigger-blocker duality packet of chapter 3801, the trigger-energy partition packet of chapter 3806, and the weighted deformation spine adjacent to the DNA window-six boundary-tile vision. The packet names a finite conditional ensemble surface; it is not a hidden biological semantics, not an external probability space, not an unbounded trigger search, and not an oracle for conditional means.

Definition 3870.1 (Conditional trigger ensemble carrier)**.** *A* $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, B, Q, Z, M, L, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the weighted trigger row, recording the finite trigger support and activity weights used by the conditional ensemble.*
2. *B is the blocker row, recording the blocker constraints inherited from* $\mathsf{TriggerBlockerDuality}^{\uparrow}$*.*
3. *Q is the partition row, recording the energy partition inherited from* $\mathsf{TriggerEnergyPartition}^{\uparrow}$*.*
4. *Z is the conditional normalization ledger, storing the finite partition function and the visible event boundary.*
5. *M is the mean row, storing the finite cardinality and energy expectations computed from Z.*
6. *L is the low-energy limit row, recording the partition-minimal trigger polynomial exposed by the source theorem.*
7. *A is the adjacency row tying the ensemble to finiteobservationresidueboundary, triggerblockerduality, triggerenergypartition, and weighted-deformation-spine-adjacent siblings.*
8. *H records componentwise* hsame *transport for trigger, blocker, partition, normalization, mean, limit, adjacency, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a conditional trigger query through weighted support, blocker filtering, energy partition, expectation readback, and low-energy limiting readback.*

10. *P is the* Pkg *provenance over* $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$, $\mathsf{FiniteObservationResidueBoundary}^{\uparrow}$, $\mathsf{TriggerBlockerDuality}^{\uparrow}$, $\mathsf{TriggerEnergyPartition}^{\uparrow}$, *the conditional trigger ensemble vision,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $T, B, Q, Z, M, L, A, H, C, P, N$. It has no coordinate for hidden codon semantics, an external probability space, an unbounded trigger search, a selected biological interpretation, or a host analytic probability theorem.

Lean4 (definition) `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleUp`

Theorem 3870.2 (Conditional trigger ensemble NameCert obligations)**.** *For every accepted* ConditionalTriggerEnse *carrier $E = (T, B, Q, Z, M, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, weighted trigger exposure through T, blocker exposure through B, energy-partition exposure through Q, conditional normalization through Z, expectation readback through M, low-energy limit readback through L, sibling adjacency through A, componentwise transport through H, continuation replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3870.1. The accepted packet exposes only weighted triggers, blockers, energy partitions, normalization, means, low-energy limits, adjacency, and the structural rows.

The ensemble calculation is finite. The trigger row T, blocker row B, partition row Q, normalization row Z, mean row M, and low-energy row L are all displayed before any consumer receives an ensemble readback.

The adjacency and structural rows A, H, C, P, N preserve that same finite packet for downstream readers. Since no coordinate stores hidden codon semantics, an external probability space, unbounded search, or a selected biological interpretation, the listed rows exhaust the local NameCert surface. □

Theorem 3870.3 (Conditional trigger ensemble sibling route)**.** *Every conditional trigger, blocker, expectation, or low-energy consumer of an accepted* $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{FiniteObservationResidueBoundary}^{\uparrow},\ \mathsf{TriggerBlockerDuality}^{\uparrow},\ \mathsf{TriggerEnergyPartition}^{\uparrow} \longmapsto T, B, Q, Z, M, L, A, H, C, P$$

The route preserves finite weighted support, blocker constraints, energy partition, conditional normalization, expectation readback, low-energy readback, sibling adjacency, transport, replay, provenance, and local naming.

Proof. Start with the sibling rows. The finite observation residue boundary keeps the observed support and residue boundary visible, while trigger-blocker duality and trigger-energy partition supply the blocker and partition rows.

Next read the ensemble rows T, B, Q, Z, M, L. They expose weighted triggers, blockers, partitions, normalization, finite means, and low-energy limiting data without importing an external probability object.

Finally use A, H, C, P, N. These rows record sibling adjacency, transport the finite packet, replay the conditional query, source it to the vision theorem, and name the local certificate. No excluded semantic or probabilistic oracle is present in the carrier. □

Falsifiable prediction: An accepted $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ read is refuted if a consumer obtains a conditional mean, energy mean, or low-energy trigger law without displaying finite weighted triggers, blocker rows, partition rows, normalization, transport, replay, provenance, and local naming.

Independence witness: finiteobservationresidueboundary, triggerblockerduality, triggerenergypartition, weighted-deformation-spine-adjacent: none is bijective to $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ because this packet jointly carries weighted trigger support, blocker filtering, energy partition, conditional normalization, expectation readback, low-energy limiting readback, sibling adjacency, transport, replay, provenance, and local naming.

The Lean follow-up for this packet must provide `Nontrivial ConditionalTriggerEnsembleUp` and `FieldFaithful ConditionalTriggerEnsembleUp` instances in `BEDC.Derived.ConditionalTriggerEnsembl` so that $T, B, Q, Z, M, L, A, H, C, P, N$ remain separated as displayed fields.

Lean4 ✓ `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}$.

Constructive story: A $\mathsf{ConditionalTriggerEnsemble}^{\uparrow}$ packet is generated from finite weighted trigger rows, blocker rows, energy-partition rows, conditional normalization, expectation readback, low-energy limiting readback, sibling adjacency, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3870.1, Definition 3870.2, and Definition 3870.3 seed the finite conditional trigger ensemble over weighted trigger, blocker, partition, expectation, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConditionalTriggerEnsemble}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConditionalTriggerEnsembleUp.ConditionalTriggerEnsembleTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close hidden biological semantics, external probability spaces, unbounded trigger search, selected biological interpretation, host analytic probability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weighted-trigger exposure, blocker exposure, partition exposure, normalization exactness, expectation readback, Nontrivial ConditionalTriggerEnsembleUp, FieldFaithful ConditionalTriggerEnsembleUp, and local naming for BEDC.Derived.ConditionalTriggerEnsembleUp.

Chapter 3871

A Concrete Naming Certificate for NormalizationEvidenceBudget$^{\uparrow}$

The NormalizationEvidenceBudget$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization boundary of chapter 9383 as finite budget data inspected before any SN read. It sits beside the frontier of chapter 892, the router of chapter 1108, the closed-substitution boundary of chapter 527, the authorized generator route of chapter 525, and the compiler-recognizer flow of chapter 1048. The packet names bounded candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy, closed-term projection, no-infinite-reduction projection, transport, replay, provenance, and local naming rows; it does not claim evidence-free strong normalization, global closed consistency, erased subject reduction, host evaluation soundness, quotient syntax equality, Lean verification, or bridge checking.

Definition 3871.1 (Normalization evidence budget carrier)**.** *A* NormalizationEvidenceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$$

with the following displayed rows:

1. *A is the audit-map anchor row for chapter 9383.*
2. *C is the candidate-evidence budget row, listing the finite candidate witnesses available for the read.*
3. *K is the closed-candidate budget row, listing the finite closed-candidate wrappers and their closed typed term boundaries.*
4. *S is the candidate-set budget row, listing finite membership, soundness, and beta-closure evidence.*
5. *P is the Pi-adequacy row for function-side candidate use.*
6. *R is the application-adequacy row for argument and application candidate use.*
7. *J is the closed-term projection row consumed through* ClosedTermSubstitutionBoundary$^{\uparrow}$*.*
8. *I is the no-infinite-reduction projection row read only after candidate, closed-candidate, or candidate-set evidence is present.*
9. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *retention row keeping candidate evidence, finished-normal evidence, and subject-reduction obstruction visible.*

10. *G is the* $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$ *generator handoff row.*
11. *D is the* $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$ *compiler-recognizer row.*
12. *H records componentwise* hsame *transport for $A, C, K, S, P, R, J, I, F, G, D, T, Q, N$.*
13. *T records displayed* Cont *replay from evidence budgets through closed-term SN and no-infinite-reduction reads.*
14. *Q is the* Pkg *provenance over* $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICNormalizationCertificateRouter}^{\uparrow}$, $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$, $\mathsf{AuthorizedGeneratorRecursor}^{\uparrow}$, $\mathsf{GroundCompilerRecognizerFlow}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for a closed-term SN read outside C, K, S, no application-normalization read outside P, R, and no no-infinite-reduction read outside I.

Lean4 ✓ `BEDC.Derived.NormalizationEvidenceBudgetUp.NormalizationEvidenceBudgetTasteGate single carrier alignment`

Theorem 3871.2 (Normalization evidence budget NameCert obligations)**.** *For the carrier of Definition 3871.1, the local* $\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}$ *surface consists exactly of audit-map anchoring through A, candidate evidence through C, closed-candidate evidence through K, candidate-set evidence through S, Pi and application adequacy through P, R, closed-term projection through J, no-infinite-reduction projection through I, frontier retention through F, authorized-generator handoff through G, compiler-recognizer handoff through D, componentwise* hsame *transport through H, displayed* Cont *replay through T,* Pkg *provenance through Q, and local naming through N.*

Proof. Project the accepted carrier $E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$. The audit anchor A fixes the route to the candidate-mediated map, and the evidence rows C, K, S list exactly the candidate, closed-candidate, and candidate-set data available before any closed-term SN read.

The adequacy rows P, R keep function, argument, and application cases inside the displayed candidate route. The projection rows J, I are downstream: J records closed-term projection, while I records the no-infinite-reduction consequence only after evidence is present.

The sibling rows F, G, D retain the normalization frontier, authorized generator handoff, and compiler-recognizer flow as visible budget provenance. They route evidence; they do not create evidence-free normalization.

Finally H, T, Q, N transport, replay, package, and name the same displayed rows. Since the carrier exposes no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3871.3 (Normalization evidence-budget refusal route)**.** *Let*

$$E = (A, C, K, S, P, R, J, I, F, G, D, H, T, Q, N)$$

be an accepted $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ *packet. If the candidate-evidence row C, closed-candidate row K, and candidate-set row S are absent from the accepted budget surface, then no closed-term strong-normalization read can pass through J and no no-infinite-reduction read can pass through I. The refusal is a local budget refusal along the MetaCIC normalization audit map boundary of chapter 9383; it does not assert evidence-free strong normalization.*

Proof. Project the carrier of Definition 3871.1. The only rows that can support a normalization read before the projections are the displayed candidate budget C, closed-candidate budget K, and candidate-set budget S.

Apply the NameCert obligations of Definition 3871.2. They place the closed-term projection J and the no-infinite-reduction projection I downstream of the evidence rows and route the remaining frontier, generator, compiler, transport, replay, provenance, and naming rows through the same finite budget.

If C, K, S are absent, the projected route has no evidence source from which J or I can be admitted. The audit-map boundary of chapter 9383 therefore records a refusal of the read, not a proof of evidence-free strong normalization. □

Falsifiable prediction: A closed-term SN read through $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ is refused when the displayed budget contains no candidate evidence, no closed-candidate evidence, and no candidate-set evidence.

Independence witness: metacicnormalizationfrontier, metacicnormalizationcertificaterouter, closedtermsubstitutionboundary, authorizedgeneratorrecursor, groundcompilerrecognizerflow: $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries candidate, closed-candidate, candidate-set, Pi/app adequacy, closed-term projection, no-infinite-reduction, frontier, generator, compiler-recognizer, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ must provide explicit Nontrivial NormalizationEvidenc and FieldFaithful $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ instances. These instances preserve the finite evidence-budget rows as distinguishable fields before SN and no-infinite-reduction reads are admitted.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ packet is generated from an audit-map anchor, candidate evidence, closed-candidate evidence, candidate-set membership, Pi and application adequacy, closed-term projection, no-infinite-reduction projection, frontier retention, authorized-generator handoff, compiler-recognizer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3871.1 and Definition 3871.2 seed the finite evidence-budget carrier for candidate-mediated normalization reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationEvidenceBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close evidence-free strong normalization, global closed consistency, erased subject reduction, host evaluation soundness, quotient syntax equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-budget refusal, projection exactness, frontier retention, generator and compiler handoff, TasteGate nontriviality, and field-faithfulness for BEDC.Derived.NormalizationEvidenceBudg

Chapter 3872

A Concrete Naming Certificate for $\mathsf{RationalIntervalSubdivision}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ packet records the finite subdivision of a rational interval into ordered rational subintervals while keeping the endpoint, mesh, and handoff rows visible. It sits after the rational interval carrier of chapter 354, the dyadic tolerance core of chapter 291, and the closed bounded interval surface of chapter 1591. The packet names a finite rational partition and its local certificate rows; it is not an open-cover compactness principle, a choice of refinement from an infinite family, quotient interval equality, or ambient completeness.

Definition 3872.1 (Rational interval subdivision carrier)**.** *A* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, E, D, M, W, K, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{RationalInterval}^{\uparrow}$ *row, exposing rational endpoints $a \leq b$ and the interval-order classifier.*
2. *E is the finite ordered endpoint list $a = x_0 \leq x_1 \leq \cdots \leq x_m = b$.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh ledger, recording the dyadic radii, width comparisons, and refinement rows used to bound each adjacent interval.*
4. *M is the mesh row assigning to each adjacent pair (x_i, x_{i+1}) its dyadic width certificate.*
5. *W is the componentwise window row exposing the subinterval sequence $([x_i, x_{i+1}])_{i<m}$ through the same finite endpoint list.*
6. *K is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *handoff row, reached only after I, E, D, M, W have been displayed.*
7. *H records componentwise* hsame *transport for the source interval, endpoints, dyadic mesh, subinterval windows, closed-interval handoff, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a subdivision request through endpoint inspection, dyadic mesh checking, subinterval-window exposure, and closed-interval handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $I, E, D, M, W, K, H, C, P, N$. It has no coordinate for open-cover compactness, infinite refinement choice, quotient interval equality, selected real representatives, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RationalIntervalSubdivisionUp.RationalIntervalSubdivisionTasteGate single carrier alignment round trip`

Theorem 3872.2 (Rational interval subdivision NameCert obligations)**.** *For every accepted* RationalIntervalSubdivi *carrier $S = (I, E, D, M, W, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$ *surface consists exactly of source-interval exposure through I, finite ordered endpoint admission through E, dyadic mesh exactness through D, adjacent-width certification through M, subinterval-window exposure through W, closed bounded interval handoff through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3872.1. The accepted packet exposes one rational source interval, one finite ordered endpoint list, one dyadic mesh ledger, one adjacent-width row, one subinterval-window row, one closed-interval handoff, and the structural rows.

The subdivision data are finite. The row I supplies the enclosing rational interval, while E lists every endpoint used by the subdivision. The dyadic ledger D and mesh row M check only adjacent pairs from this list, so no open cover, infinite refinement, or selected limiting partition enters the packet.

The consumer-facing handoff is then forced through W and K. The row W exposes the subinterval windows componentwise, and K packages the same rows for closed bounded interval consumers. Finally H, C, P, N transport, replay, source, and name exactly this displayed surface. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3872.3 (Rational interval subdivision closed-interval handoff)**.** *Every* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *or finite-net consumer of an accepted* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ *packet factors through the displayed finite route*

$$\mathsf{RationalInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{ClosedBoundedInterval}^{\uparrow}.$$

Concretely, the consumer receives the source rational interval row, the ordered rational endpoint list, the dyadic mesh ledger, the adjacent-width row, the componentwise subinterval windows, the closed-interval handoff, transport, replay, provenance, and local naming. It receives no open-cover compactness witness, choice-selected refinement, quotient interval equality, selected real representative, or ambient completeness coordinate.

Proof. Begin with an accepted carrier $S = (I, E, D, M, W, K, H, C, P, N)$. Any subdivision read first inspects I and E, so the enclosing rational interval and every subdivision endpoint are part of the displayed packet before a consumer can read a subinterval.

Next the read passes through D and M. These rows certify the dyadic mesh and the adjacent rational widths componentwise; they do not select a refinement from an infinite family and they do not invoke open-cover compactness.

The finite-net or closed-interval consumer can then read W and K. The row W lists the component subinterval windows, and K is exactly the closed bounded interval handoff built from those finite rows. The structural rows H, C, P, N preserve this same route under hsame transport and Cont replay, so the consumer has no route to quotient interval equality, selected real representatives, choice, or ambient completeness beyond the displayed packet. □

Falsifiable prediction: An accepted $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ consumer that reaches a $\mathsf{ClosedBoundedInterval}^{\uparrow}$ handoff or finite-net read without first inspecting the source $\mathsf{RationalInterval}^{\uparrow}$ row, ordered rational endpoint list, $\mathsf{DyadicRatCore}^{\uparrow}$ mesh ledger, componentwise subinterval windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row is outside the packet.

Independence witness: rationalinterval, dyadicratcore, closedboundedinterval: $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source rational interval, a finite ordered endpoint list, dyadic mesh certificates, componentwise subinterval windows, a closed-interval handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ packet is generated from a RationalInterval source row, finite ordered rational endpoints, a DyadicRatCore mesh ledger, adjacent-width rows, componentwise subinterval windows, a ClosedBoundedInterval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3872.1, Definition 3872.2, and Definition 3872.3 seed the finite rational subdivision route from rational interval rows through dyadic mesh data to closed bounded interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalSubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, infinite refinement choice, quotient interval equality, selected real representatives, finite-net compactness, or ambient completeness.

Upgrade path: Obligation closure requires checked source interval exposure, endpoint ordering, dyadic mesh exactness, subinterval-window routing, closed bounded interval handoff, transport, replay, provenance, and local naming for BEDC.Derived.RationalIntervalSubdivisionUp.

Chapter 3873

A Concrete Naming Certificate for $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$

Following Bishop and Bridges, a located interval subdivision is a finite mesh of interval cells whose endpoints are read through located real names and whose rational and dyadic windows remain visible at every handoff. The $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ packet sits after the located interval carrier of chapter 394 and the rational subdivision carrier of chapter 3872. It supplies compact-interval, finite-net, and uniform-modulus consumers with a displayed subdivision route through $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalIntervalSubdivision}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, without importing open-cover compactness, choice-selected subdivisions, quotient real equality, or ambient completeness.

Definition 3873.1 (Located interval subdivision carrier)**.** *A* $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* $\mathsf{LocatedInterval}^{\uparrow}$ *row, exposing the lower and upper located endpoint names, rational containment ledger, dyadic refinement rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and terminal* $\mathsf{Real}^{\uparrow}$ *seals.*
2. *R is the finite* $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ *row obtained from a rational cell of L.*
3. *D is the componentwise* $\mathsf{DyadicInterval}^{\uparrow}$ *mesh ledger, recording the dyadic cells refining the rational subdivision windows.*
4. *W is the located endpoint-window row. It attaches the lower and upper located real endpoint windows of L to the subdivision cells named by R and D.*
5. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the finite mesh, retaining the stream windows and regular rational comparisons used by each inspected endpoint.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, R, D, W, Q have all been displayed.*
7. *H records componentwise* hsame *transport for the located interval row, rational subdivision row, dyadic mesh, endpoint windows, stream and regular-rational readbacks, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a subdivision request through located endpoint inspection, rational subdivision admission, dyadic cell refinement, stream-window readback, regular-rational comparison, and real-seal handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalIntervalSubdivision}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSubdivision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, D, W, Q, E, H, C, P, N$. It has no coordinate for an open-cover compactness proof, an infinite family of cover refinements, a choice-selected subdivision, quotient real equality, selected endpoint representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3873.2 (Located interval subdivision NameCert obligations)**.** *For every accepted* LocatedIntervalSubdivis *carrier $S = (L, R, D, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSubdivision}^{\uparrow}}$ *surface consists exactly of source located-interval exposure through L, rational subdivision admission through R, dyadic mesh compatibility through D, located endpoint-window stability through W, stream and regular-rational readback through Q,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3873.1. The accepted packet exposes one located interval row, one rational subdivision row, one dyadic mesh ledger, one endpoint-window row, one stream and regular-rational readback row, one real seal, and the structural rows.

The located interval row is read first. It supplies the endpoint names, rational containment cell, dyadic refinements, stream windows, regular-rational readbacks, and real seals already certified by $\mathsf{LocatedInterval}^{\uparrow}$.

The subdivision data are finite. The row R lists the rational cells, while D refines those cells by dyadic interval rows. The endpoint-window row W ties each cell back to the lower and upper located real windows of L.

Finally Q, E, H, C, P, N expose the readback, seal, transport, replay, provenance, and local naming route. Since no carrier coordinate names an open cover, an infinite refinement family, a chosen subdivision, quotient real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3873.3 (Located interval subdivision dyadic mesh handoff)**.** *Every finite-net, compact-interval, or uniform-modulus consumer of an accepted* $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ *packet must factor through the displayed route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{RationalIntervalSubdivision}^{\uparrow} \longmapsto \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads the source located interval row L, then the rational subdivision row R, then the dyadic mesh ledger D, then the located endpoint windows W, then the stream and regular-rational readbacks Q, and only then the real seal E, with transport, replay, provenance, and local naming through H, C, P, N.

Proof. Begin with the source located interval. By Definition 3873.1, every accepted subdivision read must expose L before any mesh cell is consumed.

Attach the rational subdivision row. It gives a finite ordered rational cell family, and the dyadic mesh row D refines those cells componentwise through $\mathsf{DyadicInterval}^{\uparrow}$ rather than through an unbounded refinement choice.

Now read the endpoint windows and rational readbacks. The row W keeps the subdivision cells tied to the located endpoint windows of L, while Q records the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ data needed before a $\mathsf{Real}^{\uparrow}$ seal is visible.

The final handoff is the row E. The structural rows H, C, P, N transport, replay, source, and name the same finite route. Thus a consumer that reaches a finite-net, compact-interval, or uniform-modulus read without displaying the stated chain is outside the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ consumer that reaches a finite-net, compact-interval, or uniform-modulus handoff must first expose the source $\mathsf{LocatedInterval}^{\uparrow}$ row, $\mathsf{RationalIntervalSubdivision}^{\uparrow}$ subdivision row, $\mathsf{DyadicInterval}^{\uparrow}$ mesh ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: locatedinterval, rational-interval-subdivision, dyadicinterval, streamname, regseqrat, real: $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a located interval source, a finite rational subdivision, a dyadic mesh ledger, located endpoint windows, stream and regular-rational readbacks, a real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalSubdivision}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalSubdivision}^{\uparrow}$ packet is generated from a LocatedInterval source row, a RationalIntervalSubdivision finite cell row, a DyadicInterval mesh ledger, located endpoint windows, StreamName and RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalSubdivision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3873.1, Definition 3873.2, and Definition 3873.3 seed the finite located subdivision route from located interval rows through rational and dyadic mesh data to stream, regular-rational, and real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalSubdivision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, infinite refinement choice, choice-selected subdivisions, quotient real equality, selected endpoint representatives, finite-net compactness, or ambient completeness.

Upgrade path: Obligation closure requires checked source located-interval exposure, rational subdivision admission, dyadic mesh compatibility, located endpoint-window stability, stream and regular-rational readback, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalSubdivisionUp.

Chapter 3874

A Concrete Naming Certificate for $\mathsf{UniformSequentialLimit}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UniformSequentialLimit}^{\uparrow}$ packet records the standard passage from uniform convergence of a sequence of real-valued observations to sequential real-limit consumers as a finite BEDC route. It sits beside the uniform-tail certificate of chapter 3417, the sequence-source surface of chapter 3857, the sup-norm comparison boundary of chapter 3807, the finite stream and regular rational readback rows of chapter 126 and chapter 301, and the terminal real seal of chapter 38. The chapter names the finite handoff by which a uniform tail schedule may be consumed by Real-limit readers without selecting a limit function, quotienting functions, invoking countable choice, asserting ambient completeness, or importing open-cover compactness.

Definition 3874.1 (Uniform sequential limit carrier)**.** *A* $\mathsf{UniformSequentialLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, V, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sequence source row. It records the accepted* $\mathsf{LocatedSequence}^{\uparrow}$ *observations and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows from which the displayed sequence entries are read.*
2. *M is the uniform tail schedule row. It is read through the* $\mathsf{UniformConvergence}^{\uparrow}$ *packet and assigns one displayed tail bound to each accepted tolerance request.*
3. *V is the sup-norm comparison row. It consumes the* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *boundary used to compare all displayed source windows under the same tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations exposed by S, M, V.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the sequence source, uniform tail schedule, sup-norm comparison, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for S, M, V, R, E, C, P, N.*
7. *C records displayed* Cont *replay from a sequential Real-limit request through the source, tail, comparison, readback, seal, transport, provenance, and local naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformSequentialLimit}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{LocatedSequence}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, M, V, R, E, H, C, P, N$. It has no coordinate for a selected limit, quotient equality of functions, countable choice, ambient completeness of $\mathsf{Real}^\uparrow$*, open-cover compactness, Lean verification, or a bridge credential.*

Lean4 ✓ `BEDC.Derived.UniformSequentialLimitUp.UniformSequentialLimitTasteGate single carrier alignment`

Theorem 3874.2 (Uniform sequential limit NameCert obligations)**.** *For every accepted* $\mathsf{UniformSequentialLimit}^\uparrow$ *carrier*

$$U = (S, M, V, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$ *surface consists exactly of carrier admission for the sequence source, uniform tail schedule, sup-norm comparison row, regular rational readback, Real seal, componentwise transport, displayed replay, package provenance, and local name. Every sequential* $\mathsf{Real}^\uparrow$*-limit consumer of U factors through*

$$\mathsf{LocatedSequence}^\uparrow \longmapsto \mathsf{UniformConvergence}^\uparrow \longmapsto \mathsf{ContinuousSupNorm}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as displayed by $S, M, V, R, E, H, C, P, N$.

Consequently the packet exports a finite uniform-limit handoff for Real-limit readers and exports no selected limit, quotient equality of functions, countable-choice schedule, ambient $\mathsf{Real}^\uparrow$ *completeness, or open-cover compactness principle.*

Proof. Project the accepted carrier through Definition 3874.1. The source row S supplies only displayed $\mathsf{LocatedSequence}^\uparrow$ observations and finite $\mathsf{StreamName}^\uparrow$ windows, while M records the uniform tail schedule supplied by the $\mathsf{UniformConvergence}^\uparrow$ boundary.

Read the comparison row V next. Its $\mathsf{ContinuousSupNorm}^\uparrow$ surface applies the same displayed tolerance to all source windows admitted by S and M, so the consumer receives a uniform comparison ledger rather than a pointwise host-limit selector.

Now read R and E. The regular rational observations pass through $\mathsf{RegSeqRat}^\uparrow$ before the $\mathsf{Real}^\uparrow$ seal is exposed, and the structural rows H, C, P, N transport, replay, source, and name exactly that route.

The excluded principles are absent from the carrier inventory. Therefore a sequential Real-limit consumer can use only the displayed finite uniform route and cannot extract a selected limit, quotient function equality, countable choice, ambient completeness, or open-cover compactness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 3874.1 and Definition 3874.2 seed the finite route from uniform convergence data and located sequence windows to sequential Real-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSequentialLimit}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient equality of functions, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform tail schedule stability, sup-norm comparison routing, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformSequentialLimitUp.

Constructive story: A $\mathsf{UniformSequentialLimit}^\uparrow$ packet is generated from a sequence source row, uniform tail schedule, sup-norm comparison row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3875

A Concrete Naming Certificate for $\mathsf{CauchyFilterConvergence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, and the Bourbaki–Kelley language of filters, the $\mathsf{CauchyFilterConvergence}^{\uparrow}$ packet records the finite route by which displayed Cauchy-filter data reaches a named limit boundary. It sits after the Cauchy-filter surface of chapter 333, the regular-filter handoff of chapter 1445, the uniform-window discipline of chapter 1694, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names convergence as a displayed BEDC route; it does not form a quotient of filters, choose a host limit, use countable choice, identify host real equality, or appeal to ambient completeness.

Definition 3875.1 (CauchyFilterConvergence carrier)**.** *A* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, B, M, S, D, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including displayed base packets, late-set witnesses, refinement routes, and Cauchy evidence.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *factorization row exposed by the same source filter data.*
3. *M is the directed meet row that names common refinements for requested filter tails.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read from the filter tails.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows and refinements.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the stream windows and dyadic rows have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, B, M, S, D, R have been read.*
8. *W is the convergence witness row, recording that every accepted neighbourhood read of E pulls back to a displayed tail of F through B, M, S, D, R.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records the displayed* Cont *replay from filter requests to basis tails, from tails to stream windows, from dyadic tolerances to regular rational readback, and from readback to the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterConvergence}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$ *row.*

The carrier exposes convergence only through these rows. It has no coordinate for quotient filters, selected representatives, host equality of limits, countable-choice selectors, hidden cofinal tails, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterConvergenceUp.CauchyFilterConvergenceTasteGate single carrier alignment`

Theorem 3875.2 (CauchyFilterConvergence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier* $L = (F, B, M, S, D, R, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, CauchyFilter source exposure through F, basis factorization through B, directed tail refinement through M, StreamName window exposure through S, DyadicRatCore tolerance exactness through D, RegSeqRat readback through R, Real seal handoff through E, convergence pullback through W, and structural transport, replay, provenance, and local naming through H, C, P, N.*

No obligation may be discharged by quotient-filter equality, a selected host limit, countable choice, host real equality, a hidden cofinal tail, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3875.1. The rows $F, B, M, S, D, R, E, W, H, C, P, N$ are the only coordinates visible to the local certificate surface.

The source and basis side is finite. The Cauchy-filter row F supplies the displayed base packets and Cauchy evidence, while B and M expose the basis factorization and common directed refinements before any stream or real-facing read is available.

The readback side is ordered by the displayed route. The schedule S lists the finite windows, D attaches dyadic tolerances to the same windows, R reads regular rational data from them, and E is downstream from that readback.

Finally, the convergence witness W records only the pullback of Real-neighbourhood requests along the displayed route, and H, C, P, N transport, replay, source, and name the same data. Since the carrier contains none of the excluded coordinates, the listed rows exhaust the NameCert obligations. □

Theorem 3875.3 (CauchyFilterConvergence limit route)**.** *Let $L = (F, B, M, S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterConvergence}^{\uparrow}$ *carrier. Every Real-facing convergence read factors through*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the read starts at the displayed Cauchy-filter row F, passes through the basis and directed-meet rows B, M, reads finite windows and tolerances through S, D, obtains the regular rational row R, reaches the Real seal E, and verifies convergence by pulling Real-neighbourhood requests back through W. The route exports no quotient filter, selected limit, countable-choice selector, host real equality, hidden cofinal tail, or ambient completeness coordinate.

Proof. Begin with a Real-facing convergence read of L. The Real seal coordinate is E, and by the carrier definition E is downstream from the source Cauchy-filter row, basis row, directed meet row, window schedule, dyadic ledger, and regular rational readback.

Use the basis and meet rows first. The rows B and M expose the finite CauchyFilterBasis route over F, so every requested convergence tail is a displayed row rather than an external cofinal family.

Next read the sequential data. The stream schedule S and dyadic ledger D determine the regular rational readback R, which is the only route from the filter-side packet to the Real seal.

The convergence witness W closes the route by pulling each accepted neighbourhood request of E back to the displayed filter tail. The structural rows H, C, P, N preserve that same route, and the absent quotient, choice, host-equality, hidden-tail, and completeness coordinates cannot be exported by any consumer. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterConvergence}^{\uparrow}$ convergence read factors through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$ sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterConvergence}^{\uparrow}$ packet is generated from a CauchyFilterUp row, a CauchyFilterBasisUp factorization row, a directed meet row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, a convergence pullback witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3875.1, Definition 3875.2, and Definition 3875.3 seed the constructive filter-to-limit route over CauchyFilterUp, CauchyFilterBasisUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected host limits, countable choice, host real equality, hidden cofinal tails, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, basis factorization, directed meet exposure, RegSeqRat readback, Real seal nonescape, convergence pullback, and TasteGate field faithfulness for BEDC.Derived.CauchyFilterConvergenceU

Chapter 3876

A Concrete Naming Certificate for UniformConvergenceCauchyCriterion$^{\uparrow}$

Following Bishop and Bridges on constructive uniform convergence and Cauchy criteria, the UniformConvergenceCauchyCriterion$^{\uparrow}$ packet records the finite means by which a displayed family of real-valued observations is certified uniformly Cauchy before any completion or uniform-limit consumer reads it. It sits between the uniform-limit handoff of chapter 3874, the common refinement mesh of chapter 3832, and the fast criterion route of chapter 3825. The packet supplies later uniform-limit and completion-extension certificates with a finite Cauchy surface; it does not import host function limits, quotient equality of functions, countable choice, ambient completeness, or a selected limiting map.

Definition 3876.1 (Uniform convergence Cauchy criterion carrier)**.** *A* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, M, F, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform observation row. It records the accepted* UniformConvergence$^{\uparrow}$ *tail requests and the finite source windows through which the family is inspected.*
2. *M is the common mesh row supplied by* CauchyMeshRefinement$^{\uparrow}$*. It aligns all displayed source windows under one finite tolerance ledger before any readback occurs.*
3. *F is the fast criterion row supplied by* FastCauchyCriterion$^{\uparrow}$*. It exposes the monotone threshold discipline used to answer each requested precision.*
4. *Q is the Cauchy comparison row. It records that every pair of displayed source entries beyond the threshold named by F is compared through the common mesh named by M.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same windows, mesh, and fast threshold.*
6. *E is the terminal completion or* Real$^{\uparrow}$ *seal row reached only after U, M, F, Q, R have been displayed.*
7. *H records componentwise* hsame *transport for $U, M, F, Q, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from a uniform-Cauchy request through the observation, mesh, fast-threshold, comparison, readback, seal, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* UniformConvergenceCauchyCriterion$^{\uparrow}$, UniformConvergence$^{\uparrow}$, UniformSequentialLimi CauchyMeshRefinement$^{\uparrow}$, FastCauchyCriterion$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{UniformConvergenceCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, M, F, Q, R, E, H, C, P, N$. It has no coordinate for a host-space function limit, quotient equality of functions, countable-choice schedule, ambient completion theorem, or selected limit object.

Lean4 ✓ `BEDC.Derived.UniformConvergenceCauchyCriterionUp.UniformConvergenceCauchyCriterionTasteGate single carrier alignment`

Theorem 3876.2 (Uniform convergence Cauchy criterion NameCert obligations)**.** *For every accepted* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier*

$$K = (U, M, F, Q, R, E, H, C, P, N),$$

the local NameCert$_{\text{UniformConvergenceCauchyCriterion}^{\uparrow}}$ *surface consists exactly of the following five obligation families:*

1. *uniform source-window admission through U;*
2. *common finite mesh exposure through M;*
3. *fast threshold and pairwise Cauchy comparison through F, Q;*
4. *regular rational readback and terminal seal routing through R, E;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently every accepted packet exports a finite uniform-Cauchy criterion over displayed BEDC rows, not a host function limit, quotient equality of functions, countable-choice schedule, ambient completeness claim, or selected limiting map.

Proof. Project the carrier of Definition 3876.1. The row U exposes only the displayed uniform source windows, while M supplies the common finite mesh used by all later comparisons.

Read F and Q next. The fast threshold row gives the precision discipline, and the comparison row forces each post-threshold pair to be checked against the same finite mesh rather than against an unlisted limiting object.

Finally read R, E, H, C, P, N. The regular rational readback and terminal seal are downstream of the source, mesh, threshold, and comparison rows, and the structural rows transport and name exactly that route. Since the excluded analytic objects are absent from the carrier inventory, the five listed obligation families exhaust the certificate surface. □

Theorem 3876.3 (Uniform convergence Cauchy criterion completion handoff)**.** *Every uniform-limit or completion-extension consumer of an accepted* UniformConvergenceCauchyCriterion$^{\uparrow}$ *carrier factors through the displayed route*

$$\text{UniformConvergence}^{\uparrow} \longmapsto \text{CauchyMeshRefinement}^{\uparrow} \longmapsto \text{FastCauchyCriterion}^{\uparrow} \longmapsto \text{RegSeqRat}^{\uparrow} \longmapsto \text{Real}^{\uparrow}.$$

The handoff may be consumed by UniformSequentialLimit$^{\uparrow}$ *and later completion certificates only as finite Cauchy data over $U, M, F, Q, R, E, H, C, P, N$. It does not supply host function limits, quotient equality of functions, countable choice, ambient completeness, or a selected limit map.*

Proof. Begin with a consumer request for a uniform-limit or completion-extension read. Carrier admission exposes U first, so the request sees only displayed uniform source windows.

The request must then pass through M, F, Q. The common mesh row aligns the source windows, the fast row gives the precision threshold, and the comparison row records the Cauchy checks performed after that threshold.

The only downstream readback is R, E, H, C, P, N. These rows package the regular rational observations, terminal seal, transport, replay, provenance, and local naming. Hence $\mathsf{UniformSequentialLimit}^{\uparrow}$ and completion-facing consumers receive finite criterion data and no extra analytic principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3876.1, Definition 3876.2, and Definition 3876.3 seed the finite uniform-Cauchy criterion route over UniformConvergence, CauchyMeshRefinement, FastCauchyCriterion, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformConvergenceCauchyCriterionUp.UniformConvergenceCauchyCriterionTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close host function limits, quotient equality of functions, countable choice, ambient completeness, selected limit maps, or bridge checking.

Upgrade path: Obligation closure requires checked source-window admission, common mesh exposure, fast threshold stability, Cauchy comparison routing, readback, terminal seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformConvergenceCauchyCriterionUp.

Constructive story: A $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ packet is generated from uniform source windows, a common CauchyMeshRefinement row, a FastCauchyCriterion threshold row, a pairwise Cauchy comparison row, RegSeqRat readback, terminal seal routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3877

A Concrete Naming Certificate for $\mathsf{BoxDimension}^{\uparrow}$

Following Falconer's finite-cover formulation of box dimension, the $\mathsf{BoxDimension}^{\uparrow}$ packet records the BEDC surface at which metric scale, finite covers, and counting ledgers become an inspectable dimension invariant. It is downstream of the metric carrier of chapter 57, the compact metric finite-net boundary of chapter 129, the refinement language of chapter 1853, and the rational dense route of chapter 1809. The chapter names the finite scale-cover data used by lower and upper box-dimension reads; it does not add Hausdorff measure, choice over optimal covers, host logarithms, quotient compacta, or a detached completion object.

Definition 3877.1 (Box dimension finite-cover carrier)**.** *A* $\mathsf{BoxDimension}^{\uparrow}$ *carrier over a compact metric source* X *is a finite* BHist *packet*

$$B = (M, K, S, C, R, D, N, H, P, L)$$

with the following displayed rows:

1. M *is the inherited* $\mathsf{Met}^{\uparrow}$ *distance row on* X*, including the point classifier and the visible distance-witness ledger.*
2. K *is the compact metric source row. It exposes the finite-net and complete-limit data from chapter 129 before any scale cover is read.*
3. S *is the dyadic scale row. Each accepted scale is a positive dyadic tolerance history, ordered by refinement toward smaller radii.*
4. C *is the finite cover row. For each scale in* S*, it displays a finite family of balls whose centers are carried by* X *and whose radii are bounded by the current dyadic scale.*
5. R *is the refinement ledger. It records how a cover at a smaller dyadic scale refines the previous visible cover without selecting an optimal cover in the host theory.*
6. D *is the counting and comparison ledger. It records the finite cardinality of each displayed cover and the rational bounds used by the lower and upper box-dimension comparison read.*
7. N *is the separability net route. It allows the centers of the displayed finite covers to be read through the* $\mathsf{RealSeparability}^{\uparrow}$ *dense-window surface when the compact metric source is a real metric carrier.*

8. *H stores componentwise* hsame *transport for metric points, dyadic scales, cover centers, cover membership, refinement rows, counting comparisons, and dense-route data.*
9. *P is the* Pkg *provenance over* $\mathsf{BoxDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected minimal cover, ambient Euclidean embedding, host logarithm, Hausdorff-measure comparison, quotient compactum, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoxDimensionUp.BoxDimensionTasteGate single carrier alignment`

Theorem 3877.2 (Box dimension finite-cover obligations)**.** *For every accepted* $\mathsf{BoxDimension}^{\uparrow}$ *carrier*

$$B = (M, K, S, C, R, D, N, H, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$ *surface consists exactly of finite scale-cover obligations: carrier admission for the metric and compact rows, positive dyadic scale exposure, finite cover membership, radius comparison, refinement transport, cover-count comparison, separability-net routing when the source is Real-facing, provenance, and local naming. No obligation is stored outside* $M, K, S, C, R, D, N, H, P, L$.

Proof. Project the carrier of Definition 3877.1. The rows M and K provide the metric source and compact metric finite-net boundary, while S provides the positive dyadic scales at which covers are inspected.

For a fixed scale, C displays the finite cover as packet data: center histories, ball radii, membership rows, and radius bounds. The row D counts that finite family and stores the rational comparisons used by lower and upper box-dimension reads.

When the scale is refined, R is the only refinement witness. It transports a visible cover to a smaller visible cover through the recorded refinement ledger rather than by selecting a host-minimal cover.

The dense-source route N, the transport row H, the provenance row P, and the naming row L preserve the same finite data. Since the carrier contains no coordinate for optimal cover selection, host logarithms, Hausdorff measure, quotient compacta, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 3877.3 (Box dimension covering-dimension anchor)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoxDimension}^{\uparrow}$ *packet may read only the displayed finite cover and refinement rows C and R after the compact metric source K and dyadic scale row S have been exposed. The anchor sends scale-cover covers to covering-dimension refinement ledgers as finite packet data; it does not export a minimal cover, a Lebesgue-number principle, a Hausdorff-measure estimate, or an ambient Euclidean coordinate system.*

Proof. Start with the accepted carrier $B = (M, K, S, C, R, D, N, H, P, L)$. The compact metric row K is read before scale-cover data, so the consumer begins with the same finite-net and complete-limit boundary used by the compact metric certificate.

Expose a dyadic scale through S and the corresponding finite cover through C. These rows provide exactly the cover families, centers, radius bounds, and membership witnesses that a covering-dimension refinement argument may inspect.

Route the handoff through R. The refinement ledger records how smaller-scale covers refine the displayed cover family, matching the finite-cover language of chapter 1853 without introducing a host-selected optimal cover.

The counting row D and structural rows H, P, L remain bookkeeping for the same displayed covers. Because no row stores a Lebesgue-number theorem, Hausdorff-measure estimate, Euclidean embedding, or minimal-cover selector, the covering-dimension anchor is limited to the finite refinement surface. □

Theorem 3877.4 (Box dimension real-separability net route)**.** *When the compact metric source of an accepted* $\mathsf{BoxDimension}^{\uparrow}$ *packet is a Real-facing metric carrier, every scale-cover center read that reaches a* $\mathsf{RealSeparability}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RealSeparability}^{\uparrow}$$

before the center may be used as a dense-net witness. The route uses only the dyadic scale row S, finite cover row C, counting ledger D, separability route N, transport row H, provenance row P, and local name row L.

Proof. Project the box-dimension carrier. The row S supplies the dyadic scale, and the row C supplies the finite cover centers inspected at that scale.

Read those centers through the Real-facing dense route N. By chapter 1809, a separability consumer reaches Real data through dyadic tolerance, finite stream windows, regular rational readback, and the terminal Real seal before it may consume the local density surface.

The counting row D travels with the same finite cover family, so the number of displayed centers is the finite count used by the scale-cover comparison. The transport and provenance rows H, P, L preserve the route without adding a selected nearest point, host real equality, quotient stream, ambient completeness, or choice over balls.

Thus every Real-facing center read factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealSeparability}^{\uparrow}$ before it is available as dense-net evidence for the box-dimension packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3877.1, Definition 3877.2, Definition 3877.3, and Definition 3877.4 seed the finite scale-cover route for compact metric carriers, covering-dimension refinement ledgers, and RealSeparability dense-net consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoxDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff dimension, minimal-cover selection, host logarithm analysis, ambient Euclidean embeddings, quotient compacta, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale positivity, finite cover membership, refinement ledgers, cover-count comparison, and RealSeparability net routing for BEDC.Derived.BoxDimensionUp.

Constructive story: A $\mathsf{BoxDimension}^{\uparrow}$ packet is generated from a metric row, compact metric source row, dyadic scale row, finite cover row, refinement ledger, counting comparison ledger, separability net route, componentwise transport, provenance, and a local NameCert row.

Chapter 3878

A Concrete Naming Certificate for $\mathsf{MinkowskiDimension}^{\uparrow}$

Following Minkowski and Bouligand's covering-number practice, the $\mathsf{MinkowskiDimension}^{\uparrow}$ packet records the finite BEDC surface at which compact metric scale data, dyadic radii, and lower and upper covering-number ledgers become a named dimension read. It sits beside the metric carrier of chapter 57, the compact finite-net boundary of chapter 129, the dyadic scale language of chapter 291, the gauge-cover sibling of chapter 4119, and the finite scale-cover sibling of chapter 3877. The chapter names the covering-number packet used for Minkowski lower and upper reads; it does not add Hausdorff measure, selected optimal covers, ambient Euclidean coordinates, host logarithm completeness, or choice over scales.

Definition 3878.1 (Minkowski dimension carrier)**.** *A* $\mathsf{MinkowskiDimension}^{\uparrow}$ *carrier over a compact metric source* X *is a finite* BHist *packet*

$$Q = (M, K, S, N, C, R, L, T, P, A)$$

with the following displayed rows:

1. M *is the inherited* $\mathsf{Met}^{\uparrow}$ *row on* X*, including point histories, distance-witness rows, and ball-membership comparisons.*
2. K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes the finite-net and total-boundedness data before any covering number is read.*
3. S *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, listing positive dyadic radii ordered by refinement toward smaller scales.*
4. N *is the finite covering-number ledger. For each displayed scale in* S*, it records a finite cover family, the visible count of that family, and the lower and upper comparison windows used by the Minkowski read.*
5. C *is the classifier row. It compares accepted packets only through metric source data, compact finite-net data, dyadic scales, cover membership, covering counts, and lower and upper ledger entries.*
6. R *is the scale-refinement row. It records how a smaller dyadic radius reuses or refines the previous finite cover ledger without selecting a host-minimal cover.*
7. L *is the limsup and liminf ledger. It records the finite window comparisons that distinguish lower Minkowski, upper Minkowski, and exact balanced reads.*

8. *T stores componentwise* hsame *transport and displayed* Cont *replay for metric points, dyadic scales, cover families, covering counts, refinement rows, and lower and upper ledgers.*
9. *P is the* Pkg *provenance over* $\mathsf{MinkowskiDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, $\mathsf{HausdorffDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *A is the local* $\mathsf{NameCert}_{\mathsf{MinkowskiDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary outer measure, countable-cover choice, a selected optimal cover, ambient Euclidean embedding, host logarithm analysis, quotient compactum, Lean verification, or bridge checking.

Theorem 3878.2 (Minkowski dimension NameCert obligations)**.** *For every accepted* $\mathsf{MinkowskiDimension}^{\uparrow}$ *carrier*

$$Q = (M, K, S, N, C, R, L, T, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{MinkowskiDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric and compact rows, positive dyadic scale exposure, finite cover membership, covering-number comparison, classifier stability, scale-refinement transport, lower and upper limsup and liminf ledger reads, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, K, S, N, C, R, L, T, P, A$.*

Proof. Project the carrier of Definition 3878.1. The rows M and K supply the metric source and compact finite-net boundary, while S supplies the positive dyadic scales at which covering numbers are inspected.

For each displayed scale, N records a finite cover family and its visible count. The classifier row C can therefore compare accepted packets through finite membership, radius, and count data only.

Scale refinement is controlled by R. It transports the covering-number ledger to smaller dyadic radii through displayed refinement rows rather than by selecting a host-minimal cover.

The lower and upper dimension reads are the ledger row L. It records the finite liminf and limsup comparison windows used by Minkowski lower, upper, and balanced reads, so no ambient logarithm-completion theorem or outer-measure construction is invoked.

The transport row T, provenance row P, and local naming row A preserve the same packet. Since no row stores countable-cover choice, selected optimal covers, Hausdorff measure, Euclidean embedding data, quotient compacta, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Lean4 (statement) `BEDC.Derived.MinkowskiDimensionUp`

Closure status for $\mathsf{NameCert}_{\mathsf{MinkowskiDimension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinkowskiDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3878.1 and Definition 3878.2 seed the finite covering-number packet over compact metric sources, dyadic scale refinement rows, box-dimension comparison surfaces, and Hausdorff-dimension sibling routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinkowskiDimension}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.MinkowskiDimensionUp`

Bridge status: none

Not claimed: This chapter does not close exact numerical dimension values, Hausdorff measure, countable-cover choice, selected optimal covers, ambient Euclidean embeddings, host logarithm analysis, Lean-checked obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale positivity, finite cover membership, covering-number comparison, classifier stability, scale-refinement transport, limsup and liminf ledger reads, transport, replay, provenance, and local naming for BEDC.Derived.MinkowskiDimensionUp.

Constructive story: A MinkowskiDimension$^{\uparrow}$ packet is generated from a metric row, compact metric source row, dyadic scale row, finite covering-number ledger, classifier row, scale-refinement row, limsup and liminf ledger, componentwise transport and replay, provenance, and a local NameCert row.

Chapter 3879

A Concrete Naming Certificate for $\mathsf{CompleteUltrametric}^{\uparrow}$

Following Bourbaki and Bishop–Bridges metric-space practice, the $\mathsf{CompleteUltrametric}^{\uparrow}$ packet records the finite certificate surface for a complete ultrametric rather than a choice-based completion. It sits after the strong-triangle surface of chapter 1535, the limit-witness surface of chapter 128, the Cauchy-filter admission row of chapter 333, the finite-window discipline of chapter 126, the regular-rational tolerance route of chapter 301, and the terminal seal of chapter 38. The packet names one constructive non-Archimedean completeness certificate; it does not import spherical completeness, a selected quotient completion, a host completion functor, or a p-adic field structure.

Definition 3879.1 (CompleteUltrametric carrier)**.** *A* $\mathsf{CompleteUltrametric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, U, C, L, H, R, P, N)$$

with the following displayed rows:

1. *M is the metric-carrier row, including the underlying* $\mathsf{Met}^{\uparrow}$ *distance observations and ball classifier.*
2. *U is the ultrametric strong-triangle row consumed through* $\mathsf{UltrametricSpace}^{\uparrow}$*.*
3. *C is the Cauchy-filter and Cauchy-thread row, including the finite* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads by which tolerance requests are displayed.*
4. *L is the complete-limit witness row consumed through* $\mathsf{CompleteMetric}^{\uparrow}$ *and sealed only at the displayed* $\mathsf{Real}^{\uparrow}$ *endpoint row.*
5. *H records componentwise* hsame *transport for the metric, strong-triangle, Cauchy-thread, limit, provenance, and local naming rows.*
6. *R records displayed* Cont *replay from finite Cauchy-thread reads to the complete-limit handoff and from ultrametric ball comparisons to downstream consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{CompleteUltrametric}^{\uparrow}$*,* $\mathsf{UltrametricSpace}^{\uparrow}$*,* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through M, U, C, L, H, R, P, N. It has no coordinate for spherical completeness, choice-based completion, quotient completion, selected limits outside the displayed thread, p-adic field operations, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CompleteUltrametricUp.CompleteUltrametricTasteGate single carrier alignment`

Theorem 3879.2 (CompleteUltrametric NameCert obligations)**.** *For an accepted* $\mathsf{CompleteUltrametric}^{\uparrow}$ *carrier* $U = (M, U, C, L, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the displayed metric row M and local naming row N;*
2. *ultrametric triangle stability through the strong-triangle row U under componentwise* hsame *transport H;*
3. *Cauchy-thread admission through C, with every tolerance read passing through the* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *windows;*
4. *complete-limit handoff from C to L through the* $\mathsf{CompleteMetric}^{\uparrow}$ *witness and terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger non-escape for H, R, P, N, preventing transport, replay, provenance, or naming from adding spherical completeness, choice-based completion, quotient completion, selected external limits, or p-adic field structure.*

Proof. Project the accepted carrier of Definition 3879.1. The rows available to a consumer are precisely M, U, C, L, H, R, P, N.

Read the metric and ultrametric data first. The row M supplies the carrier and ball classifier, while U supplies the strong non-Archimedean triangle comparison. Transport by H preserves those displayed rows rather than replacing them by an ambient theorem.

Next read the Cauchy-thread data. The row C exposes only finite filter, stream, and regular-rational tolerance windows, so a limit request has to pass through the same displayed Cauchy evidence.

Finally consume the complete-limit handoff. The row L is reached from C through the $\mathsf{CompleteMetric}^{\uparrow}$ witness and the terminal $\mathsf{Real}^{\uparrow}$ seal. The replay, provenance, and naming rows R, P, N retain that route, so no unlisted completion principle or p-adic structure enters the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteUltrametric}^{\uparrow}$ packet is generated from a metric carrier row, an ultrametric strong-triangle row, Cauchy-filter and thread rows, a complete-limit witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3879.1 and Definition 3879.2 seed the constructive complete non-Archimedean metric certificate over displayed metric, ultrametric, Cauchy-thread, limit, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteUltrametric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close spherical completeness, choice-based completion, quotient completion, selected limits outside the displayed thread, p-adic field structure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ultrametric triangle stability, Cauchy-thread admission, complete-limit handoff, and ledger non-escape for BEDC.Derived.CompleteUltrametricUp.

Chapter 3880

A Concrete Naming Certificate for PainleveKuratowskiConvergence$^{\uparrow}$

Following Painleve and Kuratowski on set convergence, the PainleveKuratowskiConvergence$^{\uparrow}$ packet records finite-window closed-set convergence by separating lower-hit witnesses from upper-miss finite-cover witnesses. It sits over the hyperspace carrier of chapter 1660, the finite Hausdorff comparison boundary of chapter 2086, the metric source of chapter 57, the compact metric source of chapter 129, the hit/miss topology of chapter 2375, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed hit/miss windows and their readback rows; it is not a host powerset convergence theorem or a quotient subset equality principle.

Definition 3880.1 (Painleve–Kuratowski convergence carrier)**.** *A* PainleveKuratowskiConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, U, K, H, V, W, R, E, T, C, P, N)$$

with the following displayed rows:

1. *L is the lower-hit witness ledger, listing the finite inspected hits by which points or probes of a displayed closed set are met by later closed-set rows.*
2. *U is the upper-miss finite-cover witness ledger, listing finite open or ball-cover exclusions for the inspected windows.*
3. *K is the inspected compact or hyperspace source row, read through* Hyperspace$^{\uparrow}$ *and the relevant* CompactMetric$^{\uparrow}$ *carrier.*
4. *H is the* HausdorffMetric$^{\uparrow}$ *comparison boundary available only for the displayed compact windows and finite tolerances.*
5. *V is the* VietorisTopology$^{\uparrow}$ *hit/miss row that records the lower-hit and upper-miss subbasic tests used by the packet.*
6. *W is the finite* StreamName$^{\uparrow}$ *window row that schedules the inspected convergence stages and tolerance requests.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational schedules and metric tolerances seen in W and H.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular-rational, Hausdorff, and hit/miss rows have been displayed.*
9. *T records componentwise* hsame *transport for the lower-hit ledger, upper-miss ledger, hyperspace source, Hausdorff boundary, Vietoris row, stream window, regular readback, real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite hit/miss request through $K, H, V, W, R, E, T, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, K, H, V, W, R, E, T, C, P, N$. It has no coordinate for a host powerset, quotient subset equality, arbitrary supremum or infimum, selected nearest point, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PainleveKuratowskiConvergenceUp.PainleveKuratowskiConvergenceTasteGate`
`single carrier alignment`

Theorem 3880.2 (Painleve–Kuratowski convergence NameCert obligations)**.** *For every accepted* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ *carrier*

$$Q = (L, U, K, H, V, W, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower-hit ledger, upper-miss ledger, hyperspace or compact source, Hausdorff comparison boundary, Vietoris row, stream window, regular readback, real seal, transport, replay, provenance, and local naming row;*
2. *lower-hit coverage, requiring every asserted hit to be witnessed in the displayed ledger L before a convergence read is available;*
3. *upper-miss coverage, requiring every asserted miss to be witnessed by the finite-cover ledger U before exclusion data is exported;*
4. *finite-window compatibility, requiring W and R to schedule exactly the inspected stages and tolerances used by L, U, H, V;*
5. *metric and hyperspace handoff, requiring K, H, V to expose the compact-source, Hausdorff-boundary, and Vietoris hit/miss rows before the real seal is read;*
6. *componentwise transport, displayed replay, provenance, and local naming for the same L, U, K, H, V, W, R, E route.*

No Painleve–Kuratowski convergence obligation is stored outside these rows.

Proof. Project the carrier of Definition 3880.1. The lower-hit coordinate is L, the upper-miss coordinate is U, and the inspected source and topology data are K, H, V.

Lower-hit coverage is read from L. A consumer may cite a hit only after the displayed witness appears in the finite ledger, so the packet does not infer pointwise contact from an ambient subset relation.

Upper-miss coverage is read from U. Each miss is paired with a finite cover or ball-exclusion window before any convergence-facing exclusion is exported, so no arbitrary complement or open-cover compactness theorem is used.

The finite-window route is then checked through W and R. These rows schedule the inspected stages and regular rational tolerances, while H, V attach the Hausdorff and Vietoris readings to the same windows.

Finally, E, T, C, P, N seal, transport, replay, source, and name only that route. Since the carrier has no coordinate for host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, countable choice, or ambient Real completeness, the listed obligations exhaust the certificate. □

Theorem 3880.3 (Painleve–Kuratowski convergence finite-window boundary)**.** *Every convergence-facing consumer of an accepted* $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ *carrier receives only the finite displayed route*

$$L, U \longmapsto K, H, V \longmapsto W, R \longmapsto E \longmapsto T, C, P, N.$$

Thus the chapter names finite hit/miss windows for closed-set convergence; it does not export host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a convergence-facing read. By Definition 3880.1, the only lower and upper convergence data are the finite ledgers L and U.

Use Definition 3880.2. The consumer must pass through the hyperspace or compact source row K, the Hausdorff comparison boundary H, and the Vietoris hit/miss row V before it may read the finite schedules W, R and the terminal seal E.

The structural rows T, C, P, N preserve the same displayed dependency order. Because the excluded host-level principles have no carrier coordinate and no replay edge, the packet can export only the named finite-window boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3880.1, Definition 3880.2, and Definition 3880.3 seed the finite hit/miss window route for closed-set convergence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PainleveKuratowskiConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, quotient subset equality, arbitrary suprema or infima, selected nearest points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked lower-hit coverage, upper-miss coverage, finite-window compatibility, metric and hyperspace handoff, transport, replay, provenance, and local naming for BEDC.Derived.PainleveKuratowskiConvergenceUp.

Constructive story: A $\mathsf{PainleveKuratowskiConvergence}^{\uparrow}$ packet is generated from lower-hit witnesses, upper-miss finite-cover witnesses, a compact or hyperspace source row, HausdorffMetric comparison, Vietoris hit/miss topology, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3881

A Concrete Naming Certificate for $\mathsf{HaarWavelet}^{\uparrow}$

Starting from the classical Haar 1910 orthonormal system and its constructive dyadic approximation surface, the $\mathsf{HaarWavelet}^{\uparrow}$ packet records finite dyadic-window wavelet data as BEDC NameCert rows. It sits over the Schauder-basis coordinate route of chapter 3669, the dyadic partition surface of chapter 1863, the finite step-function interface of chapter 3780, and the terminal real-name seal of chapter 38. The packet supplies a finite wavelet coefficient window without importing Haar measure, quotient function spaces, ambient Hilbert-space completeness, or a selected infinite expansion.

Definition 3881.1 (Haar wavelet carrier)**.** *A* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, I, L, V, A, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the dyadic scale row, naming the inspected level and the finite family of cells at that level.*
2. *I is the parent-interval row, read through the* $\mathsf{DyadicPartition}^{\uparrow}$ *packet of chapter 1863.*
3. *L records the left and right child split of each inspected parent interval together with the sign assignment $+1, -1$.*
4. *V is the* $\mathsf{StepFunction}^{\uparrow}$ *value row, so each inspected wavelet is a signed dyadic step function over the displayed children.*
5. *A is the finite coefficient-window row, listing exactly the inspected scale-position coefficients and their local dyadic tolerances.*
6. *B is the* $\mathsf{SchauderBasis}^{\uparrow}$ *coordinate handoff row, consuming the finite coefficient window as a basis-coordinate projection.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the coefficient values and sampled step-function values.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every coefficient and sampled value exported by the packet.*
9. *H records componentwise* hsame *transport for the scale, interval, split, value, coefficient, Schauder handoff, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a wavelet-window read through $K, I, L, V, A, B, R, E, H$.*

11. *P is the* Pkg *provenance over* $\mathsf{HaarWavelet}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{SchauderBasis}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for Haar measure, an L^2 quotient class, a host orthogonality theorem, a selected infinite coefficient stream, or ambient Hilbert-space completeness.

Lean4 ✓ `BEDC.Derived.HaarWaveletUp.HaarWaveletTasteGate single carrier alignment`

Theorem 3881.2 (Haar wavelet NameCert obligations)**.** *For every accepted* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier $W = (K, I, L, V, A, B, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic scale exposure, parent-interval exposure, left/right sign-split exposure, step-function value exposure, finite coefficient-window exposure, Schauder-basis coordinate handoff, regular rational readback, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3881.1. The rows K, I, L, V display the finite dyadic scale, the parent interval, the signed child split, and the induced step-function value row before any coefficient consumer can read a wavelet.

The coefficient route is finite. The row A lists only the inspected scale-position coefficients, B hands that finite list to the $\mathsf{SchauderBasis}^{\uparrow}$ coordinate route, and R, E provide the regular rational readback and terminal Real seal for the same displayed values.

The structural rows preserve that route. Since H, C, P, N add only transport, replay, provenance, and local naming, and since the carrier has no coordinate for Haar measure, quotient function classes, host orthogonality, selected infinite streams, or Hilbert-space completeness, the listed rows exhaust the local NameCert surface. □

Theorem 3881.3 (Haar wavelet dyadic-step handoff)**.** *Every* $\mathsf{StepFunction}^{\uparrow}$ *or* $\mathsf{SchauderBasis}^{\uparrow}$ *consumer of an accepted* $\mathsf{HaarWavelet}^{\uparrow}$ *carrier reads the wavelet only through*

$$\mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{StepFunction}^{\uparrow} \longmapsto \mathsf{SchauderBasis}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed scale, split, transport, replay, provenance, and local naming rows. The handoff exports finite dyadic step functions and finite coefficient windows, not Haar integration, quotient function equality, an infinite expansion selector, or ambient completeness.

Proof. Start with the dyadic data. The scale row K and parent-interval row I determine the inspected dyadic cells, and the split row L records the left/right signed child cells over that finite partition.

Next use the step-function value row. The row V turns the signed child data into a $\mathsf{StepFunction}^{\uparrow}$ read, so a step-function consumer receives only the displayed finite cell values and not an ambient function object.

Finally read the coefficient and seal rows. The finite coefficient window A feeds the $\mathsf{SchauderBasis}^{\uparrow}$ coordinate handoff B, and the rows R, E provide regular rational readback and terminal Real seals for the same displayed coefficients. The structural rows H, C, P, N transport and name this route, so no Haar-measure row, quotient equality, infinite expansion selector, or ambient completeness coordinate is exported. □

Falsifiable prediction: An accepted $\mathsf{HaarWavelet}^{\uparrow}$ coefficient read is refuted if its $\mathsf{SchauderBasis}^{\uparrow}$ coordinate handoff is preserved while the displayed $\mathsf{DyadicPartition}^{\uparrow}$ parent interval, left/right sign split, $\mathsf{StepFunction}^{\uparrow}$ value row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal are removed.

Independence witness: dyadic-partition, step-function, schauder-basis, regseqrat, real: none is bijective with $\mathsf{HaarWavelet}^{\uparrow}$ because the packet jointly carries scale rows, parent intervals, signed child splits, step-function values, finite coefficient windows, coordinate handoff, regular readback, terminal seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}$.

Constructive story: A $\mathsf{HaarWavelet}^{\uparrow}$ packet is generated from dyadic scale rows, parent interval rows, left/right sign splits, StepFunctionUp values, finite coefficient windows, SchauderBasisUp coordinate handoff, RegSeqRatUp readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3881.1, Definition 3881.2, and Definition 3881.3 seed the finite dyadic-window wavelet packet over partition, step-function, coordinate, readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HaarWavelet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Haar measure, host orthogonality, quotient function spaces, selected infinite expansions, Hilbert-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic scale exposure, parent-interval exposure, sign-split exactness, StepFunctionUp handoff, SchauderBasisUp coordinate routing, RegSeqRatUp readback, RealUp seal exposure, and structural transport for BEDC.Derived.HaarWaveletUp.

Chapter 3882

A Concrete Naming Certificate for $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ packet records the finite adjunction-facing surface by which regular Cauchy names present the completion route without passing through a host quotient. It sits after the functorial completion packet of chapter 423, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the extension-and-uniqueness surface of chapter 397. The chapter names the adjunction as displayed BEDC data over finite rows; it does not construct a quotient completion, select limits, assert host completed metric spaces, or import ambient completeness.

Definition 3882.1 (Cauchy regular completion adjunction carrier)**.** *A* $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, R, W, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row naming the functorial completion surface over the same source metric and regular-Cauchy data.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which regular rational Cauchy names are exposed.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by the regular Cauchy route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed stream windows and regular-rational reads.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D have been consumed.*
6. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row recording extension existence and uniqueness for the same displayed completion route.*
7. *H records componentwise* hsame *transport for F, R, W, D, L, U.*
8. *C records displayed* Cont *replay from functorial completion requests to stream windows, from windows and dyadic tolerances to regular rational readback, from readback to the Real seal, and from the Real seal into the universal-property handoff.*

9. *P is the* Pkg *provenance over* CauchyRegularCompletionAdjunction$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$ *row.*

The carrier exposes the regular completion adjunction only through these finite rows. It has no coordinate for quotient completion equality, selected representatives, host completed metric spaces, hidden completeness witnesses, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRegularCompletionAdjunctionUp.CauchyRegularCompletionAdjunctionTasteGate`
`single carrier alignment`

Theorem 3882.2 (Cauchy regular completion adjunction NameCert obligations)**.** *For every accepted* CauchyRegularCompletionAdjunction$^{\uparrow}$ *carrier* $A = (F, R, W, D, L, U, H, C, P, N)$*, the local* NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$ *surface consists exactly of carrier admission for A, regular-readback compatibility through R, finite-window budget compatibility through W, D, terminal Real seal determinacy through L, universal-property handoff through U, functorial completion source exposure through F, and structural transport, replay, provenance, and local naming through H, C, P, N.*

The ledger refuses every route that would discharge an obligation by quotient completion equality, a selected limit, a host completed metric space, an ambient completeness principle, a hidden cofinal stream family, or a host-level equality of real names.

Proof. Project the carrier of Definition 3882.1. The visible rows are exactly $F, R, W, D, L, U, H, C, P, N$, and each listed NameCert obligation is a projection of one of those rows.

The regular-readback side is finite. The stream schedule W names the exposed windows, the tolerance ledger D attaches dyadic bounds to those same windows, and the regular rational row R is the only readback route before the terminal Real seal L can be consumed.

The adjunction-facing side is downstream from that route. The functor row F records the completion-facing source packet, while the universal-property row U records extension and uniqueness only after the same R, W, D, L rows have been read.

Finally, H, C, P, N preserve componentwise comparison, replay, provenance, and local naming for the displayed rows. Since the carrier contains no quotient-completion, selection, host-completion, hidden-completeness, cofinal-family, or host-equality coordinate, the ledger refusal clauses follow from exhaustion of the tuple. □

Falsifiable prediction: Every accepted CauchyRegularCompletionAdjunction$^{\uparrow}$ consumer that exports a completion adjunction while omitting the CauchyCompletionFunctor row, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, CompletionUniversalProperty row, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: cauchycompletionfunctor, regseqrat, streamname, dyadicratcore, real, and completionuniversalproperty: none is bijective to the regular completion adjunction packet because the packet jointly carries functorial source exposure, regular readback, finite-window budget, tolerance ledger, terminal seal, universal-property handoff, transport, replay, provenance, and local NameCert rows.

Closure status for NameCert$_{\text{CauchyRegularCompletionAdjunction}^{\uparrow}}$.

Constructive story: A CauchyRegularCompletionAdjunction$^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a RegSeqRat readback row, StreamName finite windows, a DyadicRatCore tolerance ledger, a terminal RealUp seal, a CompletionUniversalProperty handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRegularCompletionAdjunction}\uparrow}$, seed)

Scope closed: Definition 3882.1 and Definition 3882.2 seed the finite regular completion adjunction surface over CauchyCompletionFunctorUp, RegSeqRatUp, StreamNameUp, DyadicRatCoreUp, RealUp, CompletionUniversalPropertyUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRegularCompletionAdjunction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, RealUp completeness, quotient completion equality, selected host limits, host completed metric spaces, hidden cofinal streams, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, regular-readback compatibility, finite-window budget compatibility, terminal Real seal determinacy, universal-property handoff, ledger refusal, and TasteGate field faithfulness for BEDC.Derived.CauchyRegularCompletionAdjunctionUp.

Chapter 3883

A Concrete Naming Certificate for $\mathsf{Quasiisometry}^{\uparrow}$

Following Gromov and Bridson–Haefliger, the $\mathsf{Quasiisometry}^{\uparrow}$ packet records coarse metric equivalence as finite BEDC coarse metric data rather than as an ambient set-theoretic map. It reads source and target metric presentations from chapter 57, reuses the displayed host-map discipline of chapter 410, keeps properness visible through **??**, sends completion-facing rows through chapter 1937, and uses the readback surfaces of chapter 301 and chapter 38. The packet names finite coarse equivalence evidence; it is not a selected inverse function, not quotient equality of completions, not a host-space bijection, and not an oracle for external large-scale geometry.

Definition 3883.1 (Quasi-isometry carrier)**.** *A* $\mathsf{Quasiisometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, F, A, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source metric row, recording the named* $\mathsf{Met}^{\uparrow}$ *presentation and the source distance queries visible to the packet.*
2. *Y is the target metric row, recording the named* $\mathsf{Met}^{\uparrow}$ *presentation and the target distance queries visible to the packet.*
3. *F is the coarse map graph row, recording the finite graph of the displayed map route without adding a host-level function outside the BEDC carrier.*
4. *A is the affine distortion row, storing the multiplicative and additive control constants that bound target distances from source distances and source distances from target witnesses.*
5. *S is the coarse-surjectivity witness row, recording the finite target-cover ledger and its bounded return witnesses.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *distance readback row, storing rational windows and real-distance readings used by the coarse estimates.*
7. *E is the completion-facing row, recording the handoff to* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$ *consumers and the stability data exposed to completion routes.*
8. *H records componentwise* hsame *transport for source metric, target metric, graph, distortion, coarse-surjectivity, readback, completion-facing, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a source or target coarse query through graph readback, distortion control, surjectivity witness, distance readback, and completion-facing handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{QuasiIsometry}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, CauchyCompletionFunctoriality $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{QuasiIsometry}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through $X, Y, F, A, S, R, E, H, C, P, N$. It has no coordinate for a selected inverse map, quotient equality of completions, a host-space bijection, an external large-scale geometry theorem, or a choice principle for target witnesses.

Lean4 ✓ `BEDC.Derived.QuasiIsometryUp.QuasiIsometryTasteGate single carrier alignment`

Theorem 3883.2 (Quasi-isometry NameCert obligations)**.** *For every accepted* $\mathsf{QuasiIsometry}^{\uparrow}$ *carrier $Q = (X, Y, F, A, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuasiIsometry}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *Carrier admission for the displayed source metric, target metric, graph, distortion, surjectivity, readback, completion-facing, transport, replay, provenance, and local naming rows.*

2. *Coarse Lipschitz control, read from F, A, R, H, C, for source-distance windows and target-distance windows displayed by the packet.*

3. *Coarse-surjectivity ledger exactness, read from S, R, H, C, for every target-side witness exported by the packet.*

4. *Completion-facing stability, read from E, H, C, P, for every consumer that routes coarse metric evidence into completion functoriality.*

5. *Non-escape from host maps, selected witnesses, choice principles, quotient equality, and ambient bijections, witnessed by the absence of such coordinates from $X, Y, F, A, S, R, E, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.QuasiIsometryUp.QuasiIsometryNamecertObligations`

Proof. Project the carrier of Definition 3883.1. The accepted packet exposes only the two metric rows, the coarse graph, affine distortion constants, coarse-surjectivity witnesses, distance readback, completion-facing handoff, transport, replay, provenance, and local naming.

The coarse Lipschitz and coarse-surjectivity obligations are finite route inspections. The rows F, A, S, R, H, C display the graph, bounds, witnesses, readback windows, transport equalities, and replay path before any consumer receives a quasi-isometry reading.

The completion-facing obligation is another projection. A completion consumer must pass through E together with the same transport, replay, and provenance rows H, C, P, so it cannot obtain a host completion isomorphism or quotient equality from the packet.

The non-escape row follows from the carrier boundary itself. Since no coordinate stores a selected inverse map, a choice functional, an ambient bijection, or quotient equality of completions, the five listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{QuasiIsometry}^{\uparrow}$ read is refuted if a consumer obtains coarse equivalence, completion-facing stability, or target-side coverage without displaying source and target metric rows, graph readback, affine distortion constants, coarse-surjectivity witnesses, RegSeqRat/Real readback, transport, replay, provenance, and local naming.

Independence witness: metric, lipschitzmap, propermetric, cauchy-completion-functoriality, regseqrat, real: none is bijective to $\mathsf{QuasiIsometry}^{\uparrow}$ because this packet jointly carries source and

target metric rows, coarse graph data, affine distortion constants, coarse-surjectivity witnesses, rational and real distance readback, completion-facing stability, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}$.

Constructive story: A $\mathsf{QuasiIsometry}^{\uparrow}$ packet is generated from source and target metric rows, a finite coarse map graph, affine distortion constants, coarse-surjectivity witnesses, RegSeqRat/Real distance readback, completion-facing stability, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3883.1 and Definition 3883.2 seed the finite coarse metric equivalence certificate over displayed metric, graph, distortion, surjectivity, readback, completion-facing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuasiIsometry}\uparrow}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.QuasiIsometryUp`

Bridge status: none

Not claimed: This chapter does not close a selected inverse map, quotient equality of completions, host-space bijection, external large-scale geometry theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coarse Lipschitz control, coarse-surjectivity exactness, completion-facing stability, non-escape from host maps, and local naming for BEDC.Derived.QuasiIsometryUp.

Chapter 3884

A Concrete Naming Certificate for DyadicClosedIntervalEndpointGrid$^{\uparrow}$

Following Bishop and Bridges constructive interval analysis, the DyadicClosedIntervalEndpointGrid$^{\uparrow}$ packet records the finite dyadic endpoint grid read before compact-uniform-continuity interval consumers inspect nets, moduli, or coverage rows. It refines the closed bounded interval source of chapter 1591 through the dyadic interval surface of chapter 306, the StreamName and RegSeqRat readback rows, and the terminal Real seal. The route serves the compact-uniform-continuity program of chapter 9237 without invoking open-cover compactness, selected subcovers, quotient real equality, trichotomy, or ambient completeness.

Definition 3884.1 (Dyadic closed interval endpoint-grid carrier)**.** *A* DyadicClosedIntervalEndpointGrid$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* ClosedBoundedInterval$^{\uparrow}$ *row, exposing lower endpoint, upper endpoint, endpoint order, rational containment, dyadic containment, StreamName windows, RegSeqRat readbacks, and the Real seal through the root package.*
2. *D is the finite* DyadicInterval$^{\uparrow}$ *endpoint-grid row, listing dyadic cells whose endpoints are admitted by the interval source and ordered by the same endpoint-order route.*
3. *W is the finite window row, exposing the StreamName windows attached to the grid cells before any net, modulus, or coverage consumer reads them.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the displayed dyadic endpoints and cell windows.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row, reached only after I, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport across the closed interval source, dyadic grid, windows, RegSeqRat readbacks, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from compact-uniform-continuity interval requests through the endpoint grid before finite-net or modulus rows are inspected.*
8. *P is the* Pkg *provenance over* ClosedBoundedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, *compact-uniform-continuity interval consumers,* BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $I, D, W, R, E, H, C, P, N$. It has no coordinate for host closed intervals, open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicClosedIntervalEndpointGridUp.DyadicClosedIntervalEndpointGridTasteGate`
`single carrier alignment`

Theorem 3884.2 (Dyadic closed interval endpoint-grid obligations)**.** *For every accepted* DyadicClosedIntervalEndpo *carrier $G = (I, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$ *surface consists of carrier admission, dyadic-cell exposure, endpoint-order preservation, window handoff, Real-seal nonescape, and local naming. The carrier admits only source rows from* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*, exposes dyadic cells only through D, preserves endpoint order from I to every displayed cell endpoint, hands StreamName windows and RegSeqRat readbacks through W, R, reaches the Real seal only through E, and names the entire route by H, C, P, N.*

Proof. Project the carrier of Definition 3884.1. The source row I is the only closed bounded interval input, and the finite grid row D is the only coordinate listing dyadic endpoint cells.

Endpoint order is inherited from the source row before any grid cell is read. Since D is attached to I, every displayed dyadic endpoint is admitted through the same lower-endpoint, upper-endpoint, order, rational-containment, and dyadic-containment route used by chapter 1591.

The consumer-facing rows are then forced through W, R, E. StreamName windows are exposed by W, regular rational endpoint readbacks by R, and the terminal Real seal by E. None of those rows supplies host compactness data or a selected subcover.

Finally H, C, P, N transport, replay, source, and name exactly this displayed route. Because the carrier has no other coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3884.3 (Dyadic closed interval endpoint-grid window handoff)**.** *Every compact-uniform-continuity interval read that consumes an accepted* $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ *carrier must pass through the displayed endpoint grid*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it reads any finite-net row, coverage row, pointwise-modulus row, uniform-modulus row, or compactness consumer row.

Proof. Start with the accepted carrier $G = (I, D, W, R, E, H, C, P, N)$. A compact-uniform-continuity interval consumer first receives I, so the closed bounded interval endpoint and containment package is displayed before any grid row is available.

The read then passes through the finite dyadic endpoint grid D. This row exposes the dyadic cells and their endpoint order, but it does not yet expose a finite net, a cover, or a modulus.

Next the consumer receives the window and readback rows W, R, E. These rows hand off StreamName windows, RegSeqRat endpoint readbacks, and the Real seal as finite endpoint-grid data.

Only after this route has been replayed by C and sourced by P may a downstream compact-uniform-continuity consumer add net or modulus rows. Thus the interval read reaches the compact-uniform-continuity surface through the endpoint grid rather than through an open-cover compactness principle or a host real interval. □

Falsifiable prediction: An accepted compact-uniform-continuity interval consumer that reads finite-net, coverage, pointwise-modulus, uniform-modulus, or compactness rows before inspecting $I, D, W, R, E, H, C, P, N$ is outside the $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ packet.

Independence witness: closedboundedinterval, dyadicinterval, streamname, regseqrat, real: $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the closed bounded interval root package, a finite dyadic endpoint grid, finite windows, regular rational readbacks, a terminal Real seal, transport, replay, provenance, and local naming for compact-uniform-continuity interval consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source row, DyadicInterval endpoint-grid row, StreamName window row, RegSeqRat readback row, Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3884.1, Definition 3884.2, and Definition 3884.3 seed the finite endpoint-grid route for compact-uniform-continuity interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicClosedIntervalEndpointGrid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, finite-net compactness, open-cover compactness, pointwise-modulus extraction, uniform-modulus construction, quotient real equality, trichotomy, selected points, selected subcovers, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-cell exposure, endpoint-order preservation, window handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicClosedIntervalEndpointGr

Chapter 3885

A Concrete Naming Certificate for CauchyFilterCompletionCriterion$^{\uparrow}$

Following Kelley and Bishop–Bridges, the CauchyFilterCompletionCriterion$^{\uparrow}$ packet records the textbook completion criterion that a displayed Cauchy-filter route may reach a completion boundary only through finite basis, window, tolerance, readback, and real-seal data. It sits after the Cauchy-filter completion boundary of chapter 1569, the filter-limit basis route of chapter 3862, the Cauchy-filter surface of chapter 333, the uniform-completion surface of chapter 1473, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the criterion as finite BEDC data; it does not quotient filters, select a limit, use host equality, invoke choice, or import ambient completeness.

Definition 3885.1 (CauchyFilterCompletionCriterion carrier)**.** *A* CauchyFilterCompletionCriterion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, B, F, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* CauchyFilterCompletion$^{\uparrow}$ *completion-boundary row whose accepted reads remain tied to displayed Cauchy-filter data.*
2. *B is the* FilterLimitBasis$^{\uparrow}$ *row exposing the finite basis route used by limit and completion consumers.*
3. *F is the* CauchyFilter$^{\uparrow}$ *source row, including the displayed base, refinement, and Cauchy evidence read by Q and B.*
4. *U is the* UniformCompletion$^{\uparrow}$ *handoff row, available only after the same filter, basis, window, tolerance, and readback route has been displayed.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window schedule used to inspect requested tails.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to the same windows and completion requests.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the finite windows and dyadic ledger.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after Q, B, F, U, S, D, R have been replayed.*
9. *H records componentwise* hsame *transport for the completion, basis, filter, uniform, window, tolerance, readback, real-seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a completion-criterion request through* $Q, B, F, U, S, D, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, F, U, S, D, R, E, H, C, P, N$. There is no coordinate for quotient filters, selected representatives, host filter equality, selected completion points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionTasteGate`
`single carrier alignment`

Theorem 3885.2 (CauchyFilterCompletionCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterComple}}$ *surface consists exactly of carrier admission, CauchyFilterCompletion boundary exposure through Q, FilterLimitBasis route exposure through B, CauchyFilter source exposure through F, UniformCompletion handoff exposure through U, finite StreamName window exposure through S, DyadicRatCore tolerance exactness through D, RegSeqRat readback through R, Real seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by quotient-filter equality, selected representatives, selected completion points, host filter equality, countable choice, hidden cofinal tails, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 3885.1. The displayed rows are exactly $Q, B, F, U, S, D, R, E, H, C, P, N$, so carrier admission exposes only completion-boundary, basis, Cauchy-filter, uniform-handoff, window, tolerance, readback, seal, transport, replay, provenance, and naming data.

The completion side starts with Q, B, F. The Cauchy-filter completion row Q and the limit-basis row B both read the same Cauchy-filter source row F, so the criterion has one finite filter route rather than a quotient or detached basis route.

Next read the uniform and numerical rows. The handoff U is downstream from the displayed filter route, while S, D, R give the finite windows, dyadic tolerances, and regular rational readback used before the terminal seal E is available.

Finally the structural rows H, C, P, N preserve the same route. Since quotient-filter equality, selected representatives, selected completion points, host filter equality, choice, hidden cofinal tails, and ambient completeness are absent from the carrier, the listed rows exhaust the local surface. □

Theorem 3885.3 (CauchyFilterCompletionCriterion basis route)**.** *Let $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier. Every completion-facing or Real-facing read factors through*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{FilterLimitBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeq}$$

The $\mathsf{UniformCompletion}^{\uparrow}$ *handoff reads the same route through U and cannot bypass $F, B, Q, S, D, R, E, H, C, P, N$. Thus the criterion exports finite basis evidence for the L10 completion route without quotient filters, selected limits, host equality, choice, hidden cofinal tails, or ambient completeness.*

Lean4 ✓ BEDC.Derived.CauchyFilterCompletionCriterionUp.CauchyFilterCompletionCriterionBasisRoute

Proof. Begin with a completion-facing or Real-facing read of K. The carrier exposes the source Cauchy-filter row F, the basis row B, and the completion-boundary row Q before any terminal Real seal can be inspected.

Replay the displayed continuation row C. The read passes from F to B and Q, then through the finite window schedule S, dyadic tolerance ledger D, and regular rational readback R before it reaches E.

The uniform handoff does not open a second completion object. The row U is one of the displayed coordinates and is preserved by H, C, P, N, so a $\mathsf{UniformCompletion}^{\uparrow}$ consumer receives the same basis route rather than a hidden quotient, chosen representative, or ambient completion.

The excluded objects are not carrier rows. Consequently every accepted read factors through the route displayed in the statement. □

Theorem 3885.4 (CauchyFilterCompletionCriterion uniform handoff nonescape)**.** *For an accepted* $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ *carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff read is confined to the displayed row* U *of Definition 3885.1. The handoff may be read only after the basis route of Definition 3885.3 has exposed* F, B, Q, S, D, R, E*, and it cannot replace that route by a quotient filter, selected completion point, host equality, or countable choice.*

Proof. Inspect the carrier rows. The uniform-facing coordinate is U, and the basis-route theorem requires the Cauchy-filter source, basis row, completion-boundary row, window schedule, tolerance ledger, readback row, and Real seal to be displayed in the same packet.

The handoff is therefore a downstream read of the existing route. It cannot open a second completion object or select a limit, because no quotient, selection, host-equality, or choice coordinate appears in K.

The structural rows H, C, P, N transport, replay, source, and name the same handoff. Thus every uniform read remains inside the finite criterion packet. □

Theorem 3885.5 (CauchyFilterCompletionCriterion readback exactness)**.** *For an accepted* CauchyFilterCompletion*carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every numerical readback consumed by the criterion is exactly the finite route through the window row* S*, the dyadic tolerance row* D*, and the regular rational readback row* R *named in Definition 3885.1. The basis route of Definition 3885.3 admits no hidden cofinal tail, host filter equality, or ambient real-completeness read.*

Proof. The basis route displays the Cauchy-filter, basis, and completion-boundary rows before any numerical read is available. After those rows, the only numerical coordinates in the carrier are S, D, R.

Exactness means that the requested tail is inspected through the finite window schedule S, the tolerance is read from D, and the regular rational value is read from R. A consumer has no coordinate from which to infer an unseen cofinal tail or compare filters by host equality.

The replay and transport rows C and H preserve this same finite route, with provenance P and local naming N. Hence every readback is precisely the displayed S, D, R ledger. □

Theorem 3885.6 (CauchyFilterCompletionCriterion Real seal nonescape)**.** *For an accepted* CauchyFilterCompletio*carrier* $K = (Q, B, F, U, S, D, R, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing seal read is confined to the terminal row* E *of Definition 3885.1. The row* E *may be consumed only after the basis route of Definition 3885.3 has displayed the Cauchy-filter source, basis, completion boundary, uniform handoff, windows, dyadic tolerances, and regular rational readback. It exports no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. The terminal seal is a row of the carrier, not a theorem about all real names. The basis-route theorem places it after the filter, basis, completion-boundary, window, tolerance, and regular rational readback rows.

A Real-facing consumer therefore reaches E only by replaying the displayed finite path. The consumer cannot bypass the route through a hidden completion object, a selected representative, host equality, or countable choice.

Since E is transported and replayed only by H and C, with provenance and local naming in P and N, it seals the displayed readback without exporting ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ read factors through $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, FilterLimitBasis row, CauchyFilter source row, UniformCompletion handoff row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3885.1, Definition 3885.2, and Definition 3885.3 seed the L10 completion criterion route over CauchyFilterCompletionUp, FilterLimitBasisUp, CauchyFilterUp, UniformCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected limits, host equality, countable choice, hidden cofinal tails, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, CauchyFilterCompletion boundary exposure, FilterLimitBasis route exposure, UniformCompletion handoff nonescape, readback exactness, Real seal nonescape, and TasteGate field faithfulness for BEDC.Derived.CauchyFilterCompletionCriterionUp.

Chapter 3886

A Concrete Naming Certificate for $\mathsf{ConstructiveUniformLimit}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ packet records the theorem route saying that a uniform limit of continuous real-valued maps is continuous when every ingredient is read through finite BEDC certificate rows. It refines the continuous-map surface of chapter 130, the uniform-limit endpoint route of chapter 463, the window ledger of chapter 126, the regular-rational readback of chapter 301, and the real seal of chapter 38.

The chapter is a human-origin L10 reconstruction of the classical uniform-limit theorem as a finite NameCert route. It exposes continuous approximants, a uniform convergence ledger, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming. It does not introduce a quotient function space, a selected host limit, pointwise-only convergence, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 3886.1 (Constructive uniform-limit carrier)**.** *A* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, L, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *A is the finite family of accepted* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant rows, each exposing a common source certificate, a target* $\mathsf{Real}^{\uparrow}$ *seal, and its local continuity modulus ledger;*
2. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *row that records the uniform convergence threshold ledger and endpoint-facing route for the same family;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule on which all approximants and all convergence thresholds are read;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row tying the window data to rational endpoint observations;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the uniform-limit and regular-rational readback rows have both been consumed;*
6. *H records componentwise* hsame *transport for approximants, window schedules, threshold ledgers, readbacks, endpoint seals, and local modulus rows;*
7. *C records displayed* Cont *replay of the approximant continuity ledgers and of the uniform-limit handoff;*

8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$ *row.*

The carrier classifier compares two packets only by replaying the displayed rows $A, L, W, R, E, H, C, P, N$. No coordinate stores pointwise-only convergence, a selected host limit, quotient equality between functions, open-cover compactness, or a free completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConstructiveUniformLimitUp.ConstructiveUniformLimitTasteGate single carrier alignment`

Theorem 3886.2 (Constructive uniform-limit NameCert obligations)**.** *For every accepted* ConstructiveUniformLimi *carrier $U = (A, L, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the displayed packet, continuous-approximant admission through A, uniform-limit admission through L, finite window admission through W, regular-rational readback through R, terminal real-seal admission through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3886.1. The rows $A, L, W, R, E, H, C, P, N$ are the complete visible coordinates of one finite packet.

Read the analytic rows in dependency order. The continuous approximants provide local modulus ledgers, the uniform-limit row provides the common tail threshold, the window row supplies the shared finite schedule, and the regular-rational readback row exposes the endpoint observations used by both routes.

The seal and replay rows then close the local obligation surface. The $\mathsf{Real}^{\uparrow}$ endpoint E is available only after the shared windows, uniform threshold ledger, and regular readback row have been consumed; H and C preserve exactly those displayed rows.

Finally, P and N source the same finite packet as provenance and local naming. Since no carrier coordinate stores a selected host limit, quotient function equality, pointwise-only convergence, open-cover compactness, or ambient real completeness, no further obligation is exported. □

Theorem 3886.3 (Constructive uniform-limit continuity route)**.** *Every continuous-map consumer of an accepted* $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed hsame, Cont, Pkg, *and* NameCert *rows. The limiting map is therefore consumed as a continuity-preserving finite handoff from continuous approximants and a uniform convergence ledger, not as a host-level function limit.*

Proof. Start with a continuous-map consumer of the limiting endpoint. By Definition 3886.1, such a consumer must first read the approximant row A and the shared window schedule W.

Next use the uniform-limit row L. It supplies the common threshold ledger that lets a single window and tolerance surface control the family uniformly, so the consumer cannot replace it with a pointwise convergence witness.

Read the regular-rational handoff R and the real seal E. The endpoint is visible only along the same window and threshold route that generated the uniform-limit packet.

Finally replay transport and continuation through H and C, then source the result through P and N. The displayed route is finite and certificate-local, so quotient function spaces, selected host limits, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the consumer surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveUniformLimit}^{\uparrow}$ packet is generated from continuous approximant rows, a $\mathsf{UniformLimit}^{\uparrow}$ uniform convergence ledger, shared $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3886.1, Definition 3886.2, and Definition 3886.3 seed the finite route from continuous approximants and a uniform convergence ledger to a continuity-preserving real endpoint consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ConstructiveUniformLimitUp.ConstructiveUniformLimitTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close quotient function spaces, selected host limits, pointwise-only convergence, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or a full function-space completeness theorem.

Upgrade path: Obligation closure requires binding the approximant, uniform threshold, window, regular readback, real seal, transport, replay, provenance, and local naming rows to exact BEDC primitives and a Lean theorem surface.

Chapter 3887

A Concrete Naming Certificate for $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$ packet realizes the compact finite-net route from continuous pointwise control to a diameter-decay uniform-modulus certificate. It sits over the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-modulus consumer of chapter 369, the triangle estimate of chapter 1159, and the real endpoint seal of chapter 38.

The packet records diameter decay as finite certificate data: compact-net windows, local continuous-map oscillation bounds, a common uniform-modulus handoff, source and target metric triangle rows, and a terminal $\mathsf{Real}^{\uparrow}$ seal. It does not appeal to open-cover compactness, a selected subsequence, a choice-selected center outside the bundle, semantic continuity, or ambient real completeness.

Definition 3887.1 (Compact uniform-continuity diameter-decay carrier)**.** *A* CompactnessUniformContinuityDiamet *carrier is a finite* BHist *packet*

$$D = (K, F, M, T, E, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying finite-net centers, located source-distance reads, and compact bundle coverage;*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row exposing the graph and pointwise modulus data for the displayed map into a* $\mathsf{Real}^{\uparrow}$ *target;*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row obtained from the compact finite-net fold and consumed by downstream uniform-continuity readers;*
4. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *row joining the selected compact-net center, two source-distance bounds, two target estimates, and the final target triangle estimate;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal through which the target-diameter and oscillation reads must pass;*
6. *H records componentwise* hsame *transport for the compact source, continuous-map row, uniform-modulus row, triangle estimate, and real seal;*
7. *C records displayed* Cont *replay of the finite-net coverage, pointwise-modulus read, uniform-modulus handoff, and triangle-estimate route;*

8. *P is the* Pkg *provenance over* $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}$ *row.*

The classifier compares two packets only through the displayed rows $K, F, M, T, E, H, C, P, N$. It has no coordinate for host compactness, selected subsequences, choice outside the finite bundle, semantic continuity, quotient equality, or real completeness.

Lean4 ✓ BEDC.Derived.CompactnessUniformContinuityDiameterDecayUp.CompactnessUniformContinuityDiameterDe
single carrier alignment

Theorem 3887.2 (Diameter-decay NameCert obligations)**.** *For every accepted* CompactnessUniformContinuityDiam *carrier $D = (K, F, M, T, E, H, C, P, N)$, the local* NameCert *obligation surface consists exactly of compact-metric admission through K, continuous-map admission through F, uniform-modulus handoff through M, metric-triangle uniform estimate through T, real-seal admission through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3887.1. The finite packet exposes exactly $K, F, M, T, E, H, C, P, N$.

Read the compact-continuity rows first. The compact metric row supplies the finite-net centers and located source-distance reads, while the continuous-map row supplies the pointwise target estimates at those centers.

Next read the diameter-decay handoff. The uniform-modulus row receives a precision chosen from the finite compact-net fold, and the metric-triangle row converts the selected center route into the displayed target-diameter estimate.

The real seal, transport, replay, provenance, and local naming rows close the obligation surface. Since host compactness, selected subsequences, choice outside the finite bundle, semantic continuity, quotient equality, and real completeness are absent from the carrier, they cannot be exported by the local certificate. □

Theorem 3887.3 (Compact diameter-decay uniform-modulus route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$ *packet receives the compact finite-net row, the continuous-map pointwise rows, the metric-triangle uniform estimate, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as one finite route:*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the diameter-decay certificate is a finite compact-continuous image packet, not a semantic compactness theorem or an external modulus selector.

Proof. Start from the compact metric row K. It supplies the finite centers and source-distance reads used to measure the diameter of each displayed image window.

Read the continuous-map row F at the selected centers. The pointwise modulus data are tied to the same source rows that K exposes, so a consumer cannot choose a detached center or a detached target estimate.

Apply the triangle-estimate row T and then the uniform-modulus handoff M. The source and target triangle rows convert the local estimates into a single diameter-decay precision consumed by $\mathsf{UniformModulus}^{\uparrow}$.

Finally consume the real seal E and replay the route through H, C, P, N. The consumer obtains only the finite packet data, so open-cover compactness, selected subsequences, choice outside the bundle, semantic continuity, and real completeness remain unavailable. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}$ packet is generated from $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows, $\mathsf{ContinuousMap}^{\uparrow}$ pointwise modulus data, a $\mathsf{UniformModulus}^{\uparrow}$ handoff, a $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3887.1, Definition 3887.2, and Definition 3887.3 seed the finite diameter-decay route from compact metric finite nets and continuous-map pointwise rows to a uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactnessUniformContinuityDiameterDecay}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactnessUniformContinuityDiameterDecayUp.CompactnessUniformContinuityDiameterDecayTa`

Bridge status: none

Not claimed: This chapter does not close semantic compactness, selected subsequences, choice outside the finite bundle, quotient equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires binding the compact finite-net, continuous-map, uniform-modulus, triangle-estimate, real-seal, transport, replay, provenance, and local naming rows to exact BEDC primitives and a Lean theorem surface.

Chapter 3888

A Concrete Naming Certificate for $\mathsf{MetricCompletionIdempotence}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ packet records the fixed-point surface of constructive metric completion: once a displayed metric-completion route is already separated and complete at the finite BEDC interface, a second completion read must return to the same named rows. It sits after the metric-completion carrier of chapter 1506, the Cauchy-filter completion route of chapter 1569, the separated-completion boundary of chapter 1612, the complete separated metric surface of chapter 2355, and the uniform-completion handoff of chapter 1473. The packet gives downstream completion-functor, real-completion, and regular-rational routes a fixed-point target without quotient completion equality, selected representatives, host equality, countable choice, or an ambient completeness axiom.

Definition 3888.1 (MetricCompletionIdempotence carrier)**.** *A* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (M, S, C, U, R, H, K, P, N)$$

with the following displayed rows:

1. *M is the first* $\mathsf{MetricCompletion}^{\uparrow}$ *row, including the source metric, Cauchy-filter or Cauchy-net branch, finite readback, and separated comparison route.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row requiring the first completion read to be separated before the fixed-point claim is consumed.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{CompleteSeparatedMetric}^{\uparrow}$ *limit-service row, exposing only the displayed Cauchy-name endpoint data available inside the completed surface.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *compatibility row, tying the metric-completion read to the same uniform completion route when a uniform consumer is present.*
5. *R is the second-completion readback row. It records that any repeated completion request replays M, S, C, U rather than introducing a second quotient, selector, host equality, or ambient complete metric space.*
6. *H records componentwise* hsame *transport for the first completion, separated boundary, complete-metric service, uniform handoff, repeated readback, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from the first completion request through the separated and complete rows to the repeated-completion readback.*

8. *P is the* Pkg *provenance over* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompleteSeparatedMe}$ $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected Cauchy representatives, host equality of completed points, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a second completion object outside the replayed rows.*

Lean4 ✓ `BEDC.Derived.MetricCompletionIdempotenceUp.MetricCompletionIdempotenceTasteGate single carrier alignment`

Theorem 3888.2 (MetricCompletionIdempotence NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier* $I = (M, S, C, U, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *first-completion admission, requiring the initial read to pass through the displayed* $\mathsf{MetricCompletion}^{\uparrow}$ *row M;*

2. *separated fixed-point boundary, requiring the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row S to be read before a repeated completion request is accepted;*

3. *complete-service compatibility, requiring the limit-service row C to expose only the completed finite endpoint data already available from the same source;*

4. *uniform-route compatibility, requiring every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer to share the route M, S, C before it reaches the repeated readback row R;*

5. *idempotence nonescape, requiring R, H, K, P, N to replay, transport, package, and name exactly the same route, with no quotient completion equality, selected representative, host equality, countable choice, ambient completeness, or detached second completion coordinate.*

Proof. Project the carrier of Definition 3888.1. The first completion row is M, and the only rows that can certify fixed-point behavior before the repeated readback are S, C, U.

Read S through the separated-completion boundary of chapter 1612. This forces any equality or comparison of completed names to use displayed separated classifier data rather than host equality or quotient classes.

Read C through the complete-metric service. The row supplies endpoint service only for the already displayed completion data, so it cannot select a new representative or create a second ambient complete space.

For uniform-facing consumers, read U through chapter 1473. The uniform route is required to share the same completion rows before the repeated readback R is available.

Finally H, K, P, N transport, replay, package, and name the same finite route. Since the carrier has no excluded coordinate, the five listed rows exhaust the local NameCert surface. □

Theorem 3888.3 (MetricCompletionIdempotence repeated completion fixed point)**.** *Let $I = (M, S, C, U, R, H, K, P, N)$ be an accepted* $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ *carrier. Every second* $\mathsf{MetricCompletion}^{\uparrow}$ *read consumed through I factors through the finite route*

$$M \longmapsto S, C, U \longmapsto R \longmapsto H, K, P, N.$$

The second read therefore lands on the displayed completion surface already named by M, S, C, U and cannot be replaced by quotient completion equality, selected representatives, host equality, countable choice, ambient completeness, or a detached completion object.

Proof. Start with an accepted repeated completion request. Carrier admission gives R only as a row downstream of the first completion row M and the boundary rows S, C, U.

Use Definition 1506.3. The first completion read is already confined to the displayed metric source, cofinal branch, finite readback, and separated comparison route.

Use the separated boundary supplied by S and the complete-service row C. They ensure that the repeated request reads a completed separated metric surface rather than a quotient completion or a selected representative.

The uniform compatibility row U keeps uniform consumers on the same route. Transport and replay through H, K then carry that route to R, while P, N record provenance and local naming. □

Theorem 3888.4 (MetricCompletionIdempotence obligation scope)**.** *For an accepted* MetricCompletionIdempotenc *carrier $I = (M, S, C, U, R, H, K, P, N)$, the obligation surface is exactly first* MetricCompletion$^{\uparrow}$ *admission through M,* SeparatedCompletion$^{\uparrow}$ *boundary through S,* CompleteMetric$^{\uparrow}$/CompleteSeparatedMetric$^{\uparrow}$ *service through C,* UniformCompletion$^{\uparrow}$ *compatibility through U, repeated-readback determinacy through R,* hsame *transport through H,* Cont *replay through K,* Pkg *provenance through P, and local* NameCert *through N. It exports no quotient completion, selected representative, host equality, countable choice, or* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier. The first completion admission is the row M, and the fixed-point boundary can inspect only S, C, U before it reaches the repeated readback row R.

The separated and complete rows determine the admissible service surface. The row S blocks quotient completion equality, while C exposes only the displayed complete-metric or complete-separated-metric service already present in the packet.

Uniform compatibility is the role of U. It keeps uniform consumers on the same completion route before the repeated readback is public, so no separate completion object or host equality row is introduced.

Finally R, H, K, P, N determine repeated-readback determinacy, transport, replay, provenance, and local naming. Since the carrier has no coordinate for selected representatives, countable choice, or Real$^{\uparrow}$ completeness, the listed rows are exactly the obligation surface. □

Theorem 3888.5 (MetricCompletionIdempotence downstream scope)**.** *Every completion-functor, regular-rational, or real-completion consumer of an accepted* MetricCompletionIdempotence$^{\uparrow}$ *carrier receives only the fixed-point route displayed by $M, S, C, U, R, H, K, P, N$. The packet supplies a non-quotient idempotence target for downstream completion functoriality and Real or* RegSeqRat$^{\uparrow}$ *readback routes, but it does not close arbitrary completion monad laws, global reflection theorems, selected-limit principles, host equality of completed points, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier rows and read the repeated-completion fixed point theorem Definition 3888.3. Any downstream consumer entering through the packet is already on the route $M, S, C, U, R, H, K, P, N$.

For functorial consumers, the route starts with the same MetricCompletion$^{\uparrow}$ row M and reaches a repeated completion only through R. No additional functorial law is stored in the packet.

For regular-rational and real-completion consumers, the readback data are inherited from the finite completion rows named by M and the complete-service row C. The packet names the fixed-point surface but does not add a new RegSeqRat$^{\uparrow}$ or Real$^{\uparrow}$ completeness theorem.

The nonescape clause of Definition 3888.2 then excludes quotient equality, selected limits, host equality, choice, ambient completeness, and detached second completion data from all such consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionIdempotence}^{\uparrow}$ packet is generated from a MetricCompletionUp row, a SeparatedCompletionUp boundary row, a CompleteMetricUp or CompleteSeparatedMetricUp service row, a UniformCompletionUp compatibility row, a repeated-completion readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3888.1, Definition 3888.2, Definition 3888.3, and Definition 3888.4 close the obligation surface for repeated metric completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionIdempotence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricCompletionIdempotenceUp.MetricCompletionIdempotenceTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close arbitrary completion monad laws, global reflection theorems, quotient completion equality, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicit downstream dependency scope for completion-functor, regular-rational, and real-completion consumers of BEDC.Derived.MetricCompletionIdempotenceUp.

Chapter 3889

A Concrete Naming Certificate for MetaCICFuelTrace$^{\uparrow}$

The MetaCICFuelTrace$^{\uparrow}$ packet realizes the bounded-normalizer side of chapter 9379 as finite NameCert data. It sits beside the decidable typing frontier of chapter 3616, the bounded normal-equality checker of chapter 752, the MetaCIC decidability witness of chapter 751, and the local diamond witness of chapter 3595. The packet names bounded normalizer fuel, finished-normalization evidence, equality-check rows, refusal rows, transport, replay, provenance, and local NameCert data without promoting bounded conversion into global normalization, subject reduction, confluence, or unrestricted beta-conversion decidability.

Falsifiable prediction: Every accepted MetaCICFuelTrace$^{\uparrow}$ packet displays a finite fuel row, bounded normalizer input and output rows, finished-normalization evidence when available, a `Term.eq` or `checkSameTerm` equality-check row, a `decideClosedBetaConv` readback row, a refusal row for global conversion claims, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: metacic-decidable-typing-frontier, bounded-normal-equality-checker, typed-normal-form-decider, metacicdecidableboundary, metacic-local-diamond-witness: the typing frontier records checker admission, the bounded-normal-equality checker records comparison exactness, the typed normal-form decider records typed endpoint conditions, the decidability boundary records broader refusal data, and the local diamond witness records join obligations. None supplies the finite fuel trace packet with both explicit fuel rows and replayable finished-normalization evidence.

Definition 3889.1 (MetaCIC fuel trace carrier)**.** *A* MetaCICFuelTrace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, I, O, E, Q, B, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the bounded fuel row used by* `normalizeBounded` *and* `decideClosedBetaConv`*.*
2. *I records the represented closed-term inputs and the typing or closedness hypotheses that the decidability audit map allows this packet to consume.*
3. *O records the bounded normalizer outputs produced at fuel F.*
4. *E records finished-normalization evidence when the audit map supplies it, and records an explicit absent row when such evidence is not available.*

5. *Q is the structural equality-check row, read through* `Term.eq` *or* `checkSameTerm` *on the bounded outputs.*
6. *B is the* `decideClosedBetaConv` *readback row, interpreted only as bounded normal equality under the displayed fuel and endpoint evidence.*
7. *R is the refusal row excluding global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and host quotient equality of terms.*
8. *H records componentwise* hsame *transport for the fuel, input, output, endpoint, equality-check, bounded-conversion, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a decidability request through $F, I, O, E, Q, B, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICFuelTrace}^{\uparrow}$*, the MetaCIC decidability audit map,* MetaCICDecidableTypingF BoundedNormalEqualityChecker$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a fuel-erased global normalizer, a subject-reduction theorem, a confluence theorem, an arbitrary closedness decider, unrestricted beta-conversion decidability, or host quotient equality of represented terms.

Theorem 3889.2 (MetaCIC fuel trace NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *carrier $T = (F, I, O, E, Q, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$ *surface consists exactly of fuel admission through F, represented-input admission through I, bounded normalizer output through O, finished-normalization evidence through E, structural equality checking through Q, bounded beta-conversion readback through B, refusal through R, componentwise transport through H, displayed continuation replay through C, provenance through P, and local naming through N.*

Later Lean TasteGate work in BEDC.Derived.MetaCICFuelTraceUp.TasteGate *must provide* Nontrivial *and* FieldFaithful *instances for this carrier surface. These instances must show that the packet cannot collapse to an empty fuel marker and cannot hide a missing fuel, input, output, finished-normalization, equality-check, bounded-conversion, refusal, transport, replay, provenance, or naming row.*

Proof. Project the carrier of Definition 3889.1. The computational side is F, I, O: a displayed fuel value, represented inputs, and the bounded normalizer outputs obtained at that fuel.

Read E before the conversion result. If finished-normalization evidence is available, it is displayed here; if not, the absent row keeps the bounded comparison from becoming a global normalization theorem.

The equality rows Q, B then compare only the bounded outputs. They are interpreted through the structural equality and `decideClosedBetaConv` rows named by the decidability audit map.

The refusal row R blocks global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, and host quotient equality. Finally H, C, P, N transport, replay, record provenance for, and locally name the same fuel trace. □

Lean4 ✓ `BEDC.Derived.MetaCICFuelTraceUp.MetaCICFuelTraceTasteGate single carrier alignment`

Theorem 3889.3 (MetaCIC fuel trace bounded-conversion nonescape)**.** *Every consumer of an accepted* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *packet factors through the finite route*

$$F, I, O, E, Q, B \longmapsto R, H, C, P, N.$$

The route exports bounded normalizer outputs, optional finished-normalization evidence, and bounded equality-check readback at the displayed fuel. It exports no fuel-erased global normalization theorem, no subject-reduction theorem, no confluence theorem, no arbitrary closedness decider, no unrestricted beta-conversion oracle, and no host quotient equality of represented terms.

Proof. Start with an accepted carrier $T = (F, I, O, E, Q, B, R, H, C, P, N)$. Rows F, I, O expose the finite computation trace rather than a theorem about all possible fuels.

The endpoint row E is read before the equality-check rows. This preserves the distinction made in chapter 9379: bounded normal equality becomes a finished-normal-form statement only when finished evidence is part of the packet.

Rows Q and B compare the displayed bounded outputs. The refusal row R is then read before transport, replay, provenance, and naming finish the packet.

Because the carrier has no coordinate for stronger metatheory, no consumer can reinterpret the bounded fuel trace as global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, or quotient equality. □

Theorem 3889.4 (MetaCIC fuel trace audit scope)**.** *The* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *packet concretizes the normal-equality edge of the decidability audit map at finite fuel. It may be consumed by typing-frontier and bounded-normal-equality chapters as a replayable trace of bounded normalizer computation and finished endpoint evidence, but it does not close the global MetaCIC normalization, confluence, subject-reduction, or closedness-decidability problems.*

Proof. Use Definition 3889.2 to identify the local rows. The packet contains exactly the finite fuel, input, output, endpoint, equality-check, bounded-conversion, refusal, and structural rows.

For typing-frontier consumers, the input row I and equality rows Q, B align with the checked typing and structural equality surfaces, but the trace still carries a fixed fuel row F.

For bounded-normal-equality consumers, the output and endpoint rows O, E provide the replayable evidence required by finished-normalization consequences. They do not supply a fuel-erased endpoint theorem for all closed terms.

The nonescape theorem Definition 3889.3 then blocks the stronger global readings. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICFuelTrace}^{\uparrow}$ packet is generated from bounded fuel, represented inputs, bounded normalizer outputs, finished-normalization evidence or its explicit absence, structural equality checks, bounded beta-conversion readback, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3889.1, Definition 3889.2, and Definition 3889.3 seed the finite fuel-trace boundary for MetaCIC bounded normal equality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICFuelTrace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global normalization, subject reduction, confluence, arbitrary closedness decidability, unrestricted beta-conversion, host quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, fuel-output exactness, finished-normalization evidence handling, equality-check readback, bounded-conversion nonescape, TasteGate Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICFuelTraceUp.

Chapter 3890

A Concrete Naming Certificate for $\mathsf{CauchyStructure}^{\uparrow}$

Following Bourbaki and Kelley on Cauchy structures and Bishop–Bridges on constructive completion, the $\mathsf{CauchyStructure}^{\uparrow}$ packet names the finite criterion surface where Cauchy filters and Cauchy nets are read through stream windows, regular rational readback, dyadic tolerances, and a terminal real seal. It is grounded in the StreamName route of chapter 126, the regular sequence route of chapter 301, the real seal of chapter 38, the Cauchy-filter surface of chapter 333, the Cauchy-net surface of chapter 589, and the dyadic tolerance ledger of chapter 291.

Definition 3890.1 (Cauchy structure carrier)**.** *A* $\mathsf{CauchyStructure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the filter-base request row naming the finite family of Cauchy-filter requests accepted from chapter 333;*
2. *G is the net-directed request row naming the finite family of directed Cauchy-net requests accepted from chapter 589;*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by both filter-base and net-directed requests;*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window inspected by every request before regular readback;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the stream window and dyadic ledger;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the displayed readback route;*
7. *H is componentwise* hsame *transport across F, G, D, S, R, E;*
8. *C is displayed* Cont *replay from each filter-base or net-directed request through D, S, R, E;*
9. *P is* Pkg *provenance over* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limits, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyStructureUp.CauchyStructureTasteGate single carrier alignment`

Theorem 3890.2 (Cauchy structure NameCert obligations)**.** *For the carrier of Definition 3890.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (F, G, D, S, R, E, H, C, P, N)$*, filter-base request exposure through* F*, net-directed request exposure through* G*, dyadic tolerance exposure through* D*, finite StreamName window exposure through* S*, RegSeqRat readback through* R*, Real seal handoff through* E*, transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier row by row. The filter-base and net-directed requests are the two input faces, and both are read only through the same dyadic tolerance ledger, StreamName window, regular rational readback, and Real seal route.

Transport and replay are structural rows. The transport row compares the displayed request, tolerance, window, readback, and seal coordinates componentwise; the replay row orders every request through D, S, R, E before a consumer may read the result.

The package provenance and local NameCert row name exactly this finite packet. Since the carrier contains no other coordinate, the listed rows are precisely the NameCert obligations. □

Theorem 3890.3 (Cauchy structure filter-net criterion)**.** *Every accepted* $\mathsf{CauchyStructure}^{\uparrow}$ *route orders filter-base and net-directed Cauchy reads through the same finite criterion sequence*

$$F, G \longmapsto D \longmapsto S \longmapsto R \longmapsto E.$$

A filter-facing consumer and a net-facing consumer therefore meet at the displayed dyadic tolerance ledger, StreamName window, RegSeqRat readback, and Real seal; neither route may replace that finite sequence by quotient filters, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. For the filter direction, project the F row and follow its replay row. The replay first opens the dyadic tolerance ledger D, then the finite StreamName window S, then the RegSeqRat readback R, and only then the Real seal E.

For the net direction, project the G row and use the same replay ordering. Directed requests may change the source face, but they do not introduce a new tolerance, window, readback, or seal coordinate outside D, S, R, E.

The componentwise transport row H identifies admissible changes only along the displayed coordinates, while P and N record provenance and local naming. Thus the common filter-net criterion is the finite route above, with no quotient, choice, trichotomy, selected-limit, or ambient-completeness bypass. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyStructure}^{\uparrow}$ packet is generated from filter-base requests, net-directed requests, dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$, seed)

Scope closed: Definition 3890.1, Definition 3890.2, and Definition 3890.3 bind the finite L10 criterion surface connecting Cauchy filters and Cauchy nets to StreamName, RegSeqRat, DyadicRatCore, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyStructure}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyStructureUp.CauchyStructureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite route ordering, and consumer nonescape for BEDC.Derived.CauchyStructureUp.

The $\mathsf{UniformContinuityComposition}^{\uparrow}$ hub routes composition through chapter 130, chapter 369, and the downstream compact-uniform-continuity consumer while the certificate body remains in the sibling files.

Chapter 3891

A Concrete Naming Certificate for $\mathsf{UniformContinuityComposition}^{\uparrow}$

Following Bishop and Bridges on constructive uniformly continuous maps, the $\mathsf{UniformContinuityComposition}^{\uparrow}$ packet records the finite route by which the composite of two uniformly continuous maps receives a displayed modulus. It sits after the continuous-map carrier of chapter 130, the uniform-modulus surface of chapter 369, the metric-space route of chapter 57, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name boundary of chapter 38. The packet supplies downstream compact-uniform-continuity and completion-extension certificates with a reusable finite composition certificate; it does not import a host function space, selected moduli, quotient equality of maps, countable choice, or ambient completeness.

The composed-modulus row is a direct sibling input for the compact-uniform-continuity certificate of chapter 388, where finite compact-source moduli may consume composition without re-reading a host function-space theorem.

The $\mathsf{CompactUniformContinuity}^{\uparrow}$ consumer link is available only after the two $\mathsf{ContinuousMap}^{\uparrow}$ rows F, G and the two $\mathsf{UniformModulus}^{\uparrow}$ rows M_F, M_G have been displayed on the same composed-modulus route.

The composed-modulus row therefore consumes exactly the two $\mathsf{ContinuousMap}^{\uparrow}$ rows of chapter 130 and the two $\mathsf{UniformModulus}^{\uparrow}$ rows of chapter 369 before chapter 388 may read it.

Before the compact route of chapter 388 may reuse the composite certificate, the row M must consume the finite tolerance-to-precision ledger of chapter 369 through M_G and then M_F. Thus the sibling edge records an ordered ledger read M_G, M_F, M, not a detached compactness theorem or a hidden modulus selector.

Definition 3891.1 (Uniform continuity composition carrier)**.** *A* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, Z, F, G, M_F, M_G, M, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are the source, middle, and target* $\mathsf{Met}^{\uparrow}$ *rows. They provide the only metric contexts in which the two maps and the composite are read.*
2. *F is the first* $\mathsf{ContinuousMap}^{\uparrow}$ *row, displayed from X to Y.*
3. *G is the second* $\mathsf{ContinuousMap}^{\uparrow}$ *row, displayed from Y to Z.*

4. *M_F is the* $\mathsf{UniformModulus}^{\uparrow}$ *row for F, reading target tolerances in Y and returning source tolerances in X.*
5. *M_G is the* $\mathsf{UniformModulus}^{\uparrow}$ *row for G, reading target tolerances in Z and returning middle tolerances in Y.*
6. *M is the composed modulus row. A requested tolerance in Z is first sent to M_G, and the resulting tolerance in Y is then sent to M_F.*
7. *H records componentwise* hsame *transport for the metric, map, modulus, composite-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a composite-map request through G, M_G, F, M_F, and the composed modulus row M.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformContinuityComposition}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, Z, F, G, M_F, M_G, M, H, C, P, N$. It has no coordinate for host-level function equality, selected moduli, a quotient function space, countable choice, an ambient completion theorem, or hidden real equality.

Theorem 3891.2 (Uniform continuity composition NameCert obligations)**.** *For every accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier*

$$K = (X, Y, Z, F, G, M_F, M_G, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-context admission for X, Y, Z;*
2. *continuous-map admission for $F : X \to Y$ and $G : Y \to Z$ through the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *rows;*
3. *modulus admission for M_F and M_G through the displayed* $\mathsf{UniformModulus}^{\uparrow}$ *rows;*
4. *composed-modulus exactness for M, requiring each target tolerance in Z to be routed first through M_G and then through M_F;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently the accepted packet exports the constructive composition law for uniformly continuous maps only as finite BEDC data over the displayed rows. It does not export host-level function equality, selected moduli, quotient function spaces, countable choice, ambient completion, or hidden real equality.

Proof. Project the carrier of Definition 3891.1. The visible rows are exactly the three metric contexts, the two continuous maps, the two input moduli, the composed modulus, and the structural rows H, C, P, N.

The map side is read first. The rows F and G are continuous-map packets over the displayed metric contexts X, Y, Z, so a composite request has no map coordinate except the ordered route $X \xrightarrow{F} Y \xrightarrow{G} Z$.

The modulus side is downstream from the requested target tolerance. The row M_G converts a Z-tolerance into a Y-tolerance, and M_F converts that Y-tolerance into an X-tolerance. The row M records exactly this two-stage route.

Finally, H, C, P, N preserve componentwise comparison, replay the displayed request route, record provenance, and name the local certificate. Since the excluded analytic objects are not coordinates of K, the five listed obligation families exhaust the NameCert surface. □

Theorem 3891.3 (Uniform continuity composition modulus factorization)**.** *Every composed-modulus read accepted by a* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Met}^{\uparrow}(X, Y, Z), \quad \mathsf{ContinuousMap}^{\uparrow}(F), \quad \mathsf{UniformModulus}^{\uparrow}(M_F), \quad \mathsf{ContinuousMap}^{\uparrow}(G), \quad \mathsf{UniformModulus}^{\uparrow}(M_G),$$

Equivalently, a consumer that asks for a modulus of $G \circ F$ must first read the target tolerance through the G-modulus, then read the resulting middle tolerance through the F-modulus, and only then may expose the composite row M through H, C, P, N. No separate host composition theorem, selected modulus, quotient map equality, countable-choice selector, or ambient completeness coordinate is exported by the packet.

Proof. Begin with a consumer request for a modulus of the composite. Carrier admission supplies the metric contexts X, Y, Z and the two displayed map rows F, G, so the request is already confined to the finite route through the two continuous-map packets.

Read the second modulus before the first. A target tolerance for the composite is a Z-side request, so M_G is the first modulus row that can answer it. The answer is a displayed Y-side tolerance, not a selected global modulus or a host theorem.

Now read M_F. The Y-side tolerance returned by M_G becomes the input request for the first map, and M_F returns the source tolerance in X. The composite row M is exact because it stores no further tolerance coordinate beyond this two-step read.

The stream, regular-rational, and real rows appear only as observation and seal rows for the same finite route. The structural rows H, C, P, N transport, replay, source, and name that route, so there is no insertion point for quotient map equality, a choice-selected modulus, or an ambient completeness principle. □

Theorem 3891.4 (Uniform continuity composition L10 handoff)**.** *For accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *packets over the same displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, composition factors only through the stored modulus rows M_F, M_G, M and the displayed* hsame*,* Cont*,* Pkg*, and local* NameCert *routes. The handoff exports no host function equality, selected modulus, quotient function space, hidden completion, countable-choice selector, or ambient real equality.*

Proof. Use the carrier definition of Definition 3891.1. The displayed rows fix the metric contexts, continuous-map rows, stored input moduli, composed modulus, transport, replay, provenance, and local naming.

The NameCert obligations of Definition 3891.2 force each composite request through the two continuous-map rows and the modulus rows. No external function-space equality is part of the obligation surface.

The modulus factorization theorem Definition 3891.3 gives the L10 order: a target tolerance is sent through M_G, the returned middle tolerance is sent through M_F, and the composed row M records exactly that finite route.

Finally H, C, P, N preserve the same route over the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. Since the excluded host objects have no coordinate in the carrier, the handoff is finite and classifier-based. □

Theorem 3891.5 (Uniform continuity composition sibling route)**.** *Every* $\mathsf{CompactUniformContinuity}^{\uparrow}$ *consumer of chapter 388, and every completion-extension consumer using the same* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *packet, must first expose the two* $\mathsf{ContinuousMap}^{\uparrow}$ *rows* F, G*, the two* $\mathsf{UniformModulus}^{\uparrow}$ *rows* M_F, M_G*, the composed modulus row* M*, the source, middle, and target* $\mathsf{Met}^{\uparrow}$ *rows* X, Y, Z*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The sibling route exports no host function-space theorem, selected modulus, quotient map equality, countable-choice selector, or ambient completion principle.*

Proof. Project the carrier of Definition 3891.1. Its visible analytic rows are precisely $X, Y, Z, F, G, M_F, M_G, M$, with H, C, P, N carrying transport, replay, provenance, and naming.

Apply Definition 3891.3. A compact-uniform-continuity or completion-extension request cannot read the composed modulus until the target tolerance has passed through M_G and the returned middle tolerance has passed through M_F.

Use the L10 handoff of Definition 3891.4. The handoff keeps the same rows inside the displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route, so the consumer edge into chapter 388 is a numbered sibling route rather than a detached host theorem. Since the excluded objects are absent from the carrier, they cannot be exported by the route. □

Theorem 3891.6 (Uniform continuity composition compact-modulus consumer)**.** *Every* $\mathsf{CompactUniformContinuit}$ *or compact-modulus consumer of an accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *packet enters through chapter 388 only after reading the source, middle, and target* $\mathsf{Met}^{\uparrow}$ *rows, the two* $\mathsf{ContinuousMap}^{\uparrow}$ *rows* F, G*, the two* $\mathsf{UniformModulus}^{\uparrow}$ *rows* M_F, M_G*, the composed modulus row* M*, and then the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *routes. The consumer receives no host function-space theorem, no selected modulus, no quotient map equality, no hidden compactness theorem, and no ambient completion principle.*

Proof. Start with the carrier of Definition 3891.1. Its visible analytic coordinates are exactly $X, Y, Z, F, G, M_F, M_G, M$, followed by the structural rows H, C, P, N.

Apply the NameCert obligations of Definition 3891.2. They force the compact-modulus request to pass through the two displayed maps F, G and the two displayed input moduli M_F, M_G before the composed modulus row is available.

Use the modulus factorization theorem Definition 3891.3. The compact consumer first sends the requested target tolerance through M_G, then sends the resulting middle tolerance through M_F, and only then reads the composed modulus row M.

Finally apply the L10 handoff of Definition 3891.4. The observation and seal rows $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ are downstream of the same finite route, and H, C, P, N preserve it. Thus the consumer edge into chapter 388 is a displayed lattice edge from the two map rows and two modulus rows, not a hidden compactness theorem. Since the excluded objects have no carrier coordinate, the compact-modulus consumer cannot obtain them from the packet. □

Theorem 3891.7 (Uniform continuity composition ContinuousMap route)**.** *Every compact-modulus or completion-extension consumer of an accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *packet must first consume the two displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *rows* F, G *of chapter 130 together with their* $\mathsf{UniformModulus}^{\uparrow}$ *rows* M_F, M_G *before it can read the composed modulus row used by chapter 388. The route exposes no host function-space theorem, quotient map equality, selected modulus, hidden compactness theorem, countable-choice selector, or ambient completion principle.*

Proof. Start from the NameCert obligations. They list continuous-map admission for $F : X \to Y$ and $G : Y \to Z$ and modulus admission for M_F and M_G as separate displayed rows before any composed modulus can be consumed.

Apply modulus factorization and the L10 handoff. A target tolerance is routed through M_G and then M_F, and the resulting composed row remains inside the finite $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoff.

Finally use the compact-modulus consumer theorem. Since that consumer already has to read $X, Y, Z, F, G, M_F, M_G, M, H, C, P, N$, the dependency on ContinuousMap and CompactUniformContinuity is recorded as the displayed sibling route rather than as a kernel-only packet or host theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformContinuityComposition}^{\uparrow}$ packet is generated from source, middle, and target MetricSpace rows, two ContinuousMap rows, two UniformModulus rows, a composed modulus row, StreamName and RegSeqRat observation rows, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3891.1, Definition 3891.2, and Definition 3891.3 seed the finite constructive composition route over ContinuousMap, UniformModulus, MetricSpace, StreamName, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformContinuityComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host-level function equality, selected moduli, quotient function spaces, countable choice, ambient completion, or hidden real equality.

Upgrade path: Obligation closure requires checked metric-context admission, continuous-map admission, modulus admission, composed-modulus exactness, observation routing, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformContinuityCompositionUp.

Theorem 3891.8 (Uniform continuity composition sibling dependency route)**.** *Every accepted* $\mathsf{UniformContinuityComposition}^{\uparrow}$ *read that exposes the composed modulus row must first consume the two* $\mathsf{ContinuousMap}^{\uparrow}$ *routes of chapter 130 and the two* $\mathsf{UniformModulus}^{\uparrow}$ *routes of chapter 369. The handoff reaches the composed row only through the displayed* F, G, M_F, M_G *coordinates and the structural* H, C, P, N *route; it exports no kernel-only composition fact, host function-space theorem, quotient map equality, selected modulus, countable-choice selector, or ambient completion principle.*

Proof. Start with the carrier of Definition 3891.1. Its analytic input rows are exactly the three metric contexts, the two continuous-map rows F, G, the two input modulus rows M_F, M_G, and the composed modulus row M.

Apply the NameCert obligations of Definition 3891.2. They make continuous-map admission and modulus admission separate displayed obligations before the composed row can be read.

Finally use the modulus factorization route of Definition 3891.3. A target tolerance enters through the G-modulus and then the F-modulus, so the sibling chapters supply the exact dependency route rather than a hidden host theorem. □

Chapter 3892

A Concrete Naming Certificate for $\mathsf{HadamardThreeCircle}^{\uparrow}$

Following Hadamard's three-circle theorem and the classical complex-analysis reading of maximum-modulus bounds on annuli, the $\mathsf{HadamardThreeCircle}^{\uparrow}$ packet records a finite annulus ledger whose radius reads, boundary maxima, and logarithmic-convexity row are all displayed before a holomorphic-boundary consumer may read a bound. It sits over the complex carrier of chapter 39, the circle boundary surface of chapter 58, the Real seal of chapter 38, the holomorphic carrier of chapter 68, the finite integral row of chapter 123, and the metric annulus packet of chapter 3371. The chapter names the finite three-radius certificate only; it does not assert an ambient analytic continuation theorem, a selected maximizer, quotient function equality, or a bridge certificate.

Definition 3892.1 (Hadamard three-circle carrier)**.** *A* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N)$$

with the following displayed rows:

1. *A is the displayed annulus row, read through* $\mathsf{MetricAnnulus}^{\uparrow}$ *and carrying the two boundary circles and the inspected middle radius.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row over the annulus, with values in the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *R_0, R_1, R_2 are the three positive* $\mathsf{Real}^{\uparrow}$ *radius reads, ordered by the annulus row and sealed before any maximum row is consumed.*
4. *B is the boundary-circle readback row, using* $\mathsf{S}^{1\uparrow}$ *to expose the inner, middle, and outer circle probes.*
5. *M is the maximum-modulus row for the displayed boundary reads, recording only finite upper-bound data on the inspected circles.*
6. *L is the logarithmic-convexity ledger tying the three radius reads and the corresponding maximum-modulus rows to a single finite inequality certificate.*
7. *Hs records componentwise* hsame *transport for the annulus, holomorphic map, radius reads, boundary reads, maximum rows, logarithmic-convexity ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an annulus-bound request through $A, F, R_0, R_1, R_2, B, M, L, Hs$.*

9. *P is the* Pkg *provenance over* $\mathsf{HadamardThreeCircle}^{\uparrow}$, $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host supremum theorem, an uninspected boundary point, a selected maximizer, quotient function equality, analytic continuation outside the annulus, or countable choice.

Lean4 ✓ `BEDC.Derived.HadamardThreeCircleUp.HadamardThreeCircleTasteGate single carrier alignment`

Theorem 3892.2 (Hadamard three-circle NameCert obligations)**.** *For every accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier* $H = (A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$ *surface is exhausted by annulus admission, holomorphic-map exposure, three sealed radius reads, boundary-circle readback, maximum-modulus rows, logarithmic-convexity ledger compatibility, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet may not infer an ambient maximum principle, a selected extremal point, analytic continuation, quotient function equality, or any infinite covering principle from the local certificate.*

Proof. Project the carrier of Definition 3892.1. The rows A, F, R_0, R_1, R_2 display the annulus, the holomorphic map, and the three radius reads before a bound can be consumed.

The boundary computation is then finite. The row B exposes the inspected $\mathsf{S}^{1\uparrow}$ circle probes, M records the corresponding maximum-modulus upper-bound rows, and L binds those rows to the logarithmic-convexity ledger for the same three radii.

Finally the structural rows Hs, C, P, N transport, replay, cite provenance, and name that same finite route. Since none of these rows adds a host supremum theorem, a selected maximizer, an uninspected boundary point, analytic continuation, quotient equality, or countable choice, the listed obligations exhaust the local NameCert surface. □

Theorem 3892.3 (Hadamard annulus maximum handoff)**.** *Every holomorphic-boundary consumer of an accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier factors through*

$$\mathsf{MetricAnnulus}^{\uparrow} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{Complex}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

together with the displayed rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$. The handoff exports a finite three-radius annulus ledger and its logarithmic-convexity certificate, not a global maximum-modulus theorem or an ambient analytic object outside the displayed annulus.

Proof. Start from the annulus row. The packet reads the displayed $\mathsf{MetricAnnulus}^{\uparrow}$ data A and the $\mathsf{Holomorphic}^{\uparrow}$ map row F before any boundary maximum is available.

Next seal the radius and boundary rows. The three $\mathsf{Real}^{\uparrow}$ reads R_0, R_1, R_2 determine the inspected circles, and B turns those circles into finite $\mathsf{S}^{1\uparrow}$ boundary probes over the same annulus.

The maximum handoff is then local. The row M records the finite boundary upper bounds, while L ties them to the logarithmic-convexity inequality for the same radii. The $\mathsf{Integral}^{\uparrow}$ dependency is consumed only as a displayed finite contour or averaging row, not as an ambient selection principle.

The transport and naming rows Hs, C, P, N preserve this route. Therefore a consumer receives exactly the finite annulus ledger and no global maximum principle, selected extremal point, analytic continuation coordinate, quotient equality, or infinite covering data. □

Theorem 3892.4 (Hadamard three-circle maximum-modulus boundary)**.** *Let $H = (A, F, R_0, R_1, R_2, B, M, L, Hs,$ be an accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *carrier. If the inner and outer boundary maximum ledgers in M display the same* $\mathsf{Real}^{\uparrow}$ *modulus row, then the middle-radius bound classified by L is read through that same displayed boundary modulus row. The conclusion uses only the finite annulus ledger, the* $\mathsf{Real}^{\uparrow}$ *modulus reads, and the* $\mathsf{Holomorphic}^{\uparrow}$ *source rows; it does not select a maximizer, add ambient analytic continuation, or import a host maximum-modulus theorem.*

Proof. Start with Definition 3892.3. It forces every holomorphic-boundary read to pass through the annulus row A, holomorphic row F, radius reads R_0, R_1, R_2, boundary probes B, maximum rows M, and logarithmic-convexity ledger L.

Read the hypothesis inside M. The inner and outer boundary maximum ledgers display the same $\mathsf{Real}^{\uparrow}$ modulus row, so the two endpoint values supplied to the logarithmic-convexity ledger are the same boundary datum.

The ledger L then classifies the middle-radius bound using that displayed endpoint row. Since the route is still the finite annulus route named by $A, F, R_0, R_1, R_2, B, M, L$, the conclusion has no coordinate from which to select an extremal point or extend the holomorphic source.

Finally Hs, C, P, N transport, replay, source, and name the same certificate. They preserve the finite boundary-modulus read and do not add ambient analytic continuation, quotient function equality, or a host maximum-modulus principle. □

Theorem 3892.5 (Hadamard three-circle annulus route)**.** *Every accepted* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *annulus-bound consumer factors first through*

$$\mathsf{MetricAnnulus}^{\uparrow} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{Complex}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{Integral}^{\uparrow}$$

and then through the displayed rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$. This route exports only the finite annulus ledger, holomorphic source row, complex value carrier, circle boundary probes, real radius and modulus reads, integral averaging surface, maximum-modulus boundary rows, logarithmic-convexity ledger, transport, replay, provenance, and local name. It excludes selected maximizers, global maximum principles, analytic continuation beyond the annulus, quotient function equality, infinite covers, and bridge certificates.

Proof. Apply the annulus maximum handoff of Definition 3892.3. It already orders the cross-chapter surfaces as $\mathsf{MetricAnnulus}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Integral}^{\uparrow}$, and it exposes the same rows $A, F, R_0, R_1, R_2, B, M, L, Hs, C, P, N$ before any annulus-bound read is consumed.

Apply the boundary theorem of Definition 3892.4 to the maximum-modulus part of the route. The boundary theorem keeps the middle-radius conclusion inside the displayed boundary-modulus row and the finite logarithmic-convexity ledger, so the composed route has no coordinate for a selected maximizer, an ambient maximum principle, analytic continuation outside the annulus, quotient equality, an infinite cover, or a bridge certificate. □

Falsifiable prediction: An accepted $\mathsf{HadamardThreeCircle}^{\uparrow}$ read is refuted if the logarithmic-convexity row L is preserved while the displayed annulus row A, holomorphic map row F, sealed radii R_0, R_1, R_2, boundary-circle row B, and maximum-modulus row M are removed.

Independence witness: complex, sone, real, holomorphic, integral, metric-annulus: none is bijective with $\mathsf{HadamardThreeCircle}^{\uparrow}$ because the packet jointly carries annulus rows, holomorphic map rows, three sealed radius reads, boundary circle probes, maximum-modulus rows, a logarithmic-convexity ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardThreeCircle}^{\uparrow}$ packet is generated from a displayed annulus, a holomorphic map row, three Real radius reads, boundary circle probes, maximum-modulus rows, a logarithmic-convexity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3892.1, Definition 3892.2, and Definition 3892.3 seed the finite annulus ledger for three-circle maximum-modulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardThreeCircle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a global maximum-modulus theorem, selected extremal points, analytic continuation beyond the annulus, quotient function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked annulus admission, holomorphic-map exposure, three-radius ordering, boundary-circle readback, maximum-modulus rows, logarithmic-convexity compatibility, and structural transport for BEDC.Derived.HadamardThreeCircleUp.

Chapter 3893

A Concrete Naming Certificate for $\mathsf{HadamardThreeLines}^{\uparrow}$

Following Hadamard's three-lines theorem and the Riesz-Thorin style reading of holomorphic strip estimates, the $\mathsf{HadamardThreeLines}^{\uparrow}$ packet records a finite vertical-strip ledger whose boundary-line reads, interior height read, and logarithmic-convexity bound are all displayed before a strip-bound consumer may read an estimate. It sits beside the annular packet of chapter 3892, over the holomorphic carrier of chapter 68, the complex carrier of chapter 39, the Real seal of chapter 38, the finite integral row of chapter 123, and the metric annulus packet of chapter 3371. The chapter names the finite strip-bound certificate only; it does not collapse strip geometry into an annulus certificate, assert host supremum selection, or add a bridge certificate.

Definition 3893.1 (Hadamard three-lines carrier)**.** *A* $\mathsf{HadamardThreeLines}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N)$$

with the following displayed rows:

1. *S is the displayed vertical-strip row, exposing the two boundary lines and the inspected interior line.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row on the strip, with values in the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *Y_0, Y_1, Y are the two boundary-height* $\mathsf{Real}^{\uparrow}$ *reads and the inspected interior-height* $\mathsf{Real}^{\uparrow}$ *read, ordered by the strip row before any bound is consumed.*
4. *B is the boundary-line readback row, exposing the finite probes on the two boundary lines and the requested interior line.*
5. *M is the finite boundary-majorant row, recording only displayed upper-bound data for the boundary reads.*
6. *E is the logarithmic-convexity estimate row tying Y_0, Y_1, Y and M to a single finite strip-bound certificate.*
7. *Hs records componentwise* hsame *transport for the strip row, holomorphic map row, height reads, boundary reads, majorant row, estimate row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a strip-bound request through $S, F, Y_0, Y_1, Y, B, M, E, Hs$.*
9. *P is the* Pkg *provenance over* $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{HadamardThreeCircle}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{MetricAnnulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^\uparrow}$ *row.*
The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient function equality, a selected boundary supremum, an unbounded strip limit, countable choice, ambient Real completeness, or an annulus replacement of the strip route.

Lean4 ✓ `BEDC.Derived.HadamardThreeLinesUp.HadamardThreeLinesTasteGate single carrier alignment`

Theorem 3893.2 (Hadamard three-lines NameCert obligations)**.** *For every accepted* $\mathsf{HadamardThreeLines}^\uparrow$ *carrier* $L = (S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^\uparrow}$ *surface is exhausted by strip admission, holomorphic-map exposure, two boundary-height reads, one interior-height read, boundary-line readback, finite boundary-majorant rows, logarithmic-convexity estimate compatibility, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet may not infer quotient function equality, host function extensionality, selected boundary suprema, unbounded strip limiting, countable choice, ambient* $\mathsf{Real}^\uparrow$ *completeness, or a collapse into* $\mathsf{HadamardThreeCircle}^\uparrow$*.*

Proof. Project the carrier of Definition 3893.1. The rows S, F, Y_0, Y_1, Y display the strip, the holomorphic map, and the three height reads before an estimate can be consumed.

The boundary computation is finite. The row B exposes the inspected boundary-line probes, M records their finite majorant data, and E binds those rows to the logarithmic-convexity estimate for the same boundary and interior heights.

Finally the structural rows Hs, C, P, N transport, replay, cite provenance, and name that same finite route. Since none of these rows adds quotient function equality, host extensionality, selected suprema, an unbounded limiting principle, countable choice, ambient Real completeness, or an annulus replacement coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 3893.3 (Hadamard three-lines strip-bound route)**.** *Every strip-bound consumer of an accepted* $\mathsf{HadamardThreeLines}^\uparrow$ *carrier factors through*

$$\mathsf{Holomorphic}^\uparrow \longmapsto \mathsf{Complex}^\uparrow \longmapsto \mathsf{Real}^\uparrow \longmapsto \mathsf{Integral}^\uparrow$$

together with the displayed rows $S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N$*. The route exports a finite vertical-strip ledger and its logarithmic-convexity estimate, not the annulus ledger of* $\mathsf{HadamardThreeCircle}^\uparrow$*, not a* $\mathsf{MetricAnnulus}^\uparrow$ *maximum handoff, and not an ambient analytic object outside the displayed strip.*

Proof. Start from the strip row. The packet reads the displayed strip data S and the $\mathsf{Holomorphic}^\uparrow$ map row F before any boundary-line estimate is available.

Next seal the height and boundary rows. The $\mathsf{Real}^\uparrow$ reads Y_0, Y_1, Y determine the inspected boundary and interior lines, while B turns those lines into finite boundary probes.

The strip-bound handoff is then local. The row M records the finite boundary majorants, and E ties them to the logarithmic-convexity estimate for the same heights. The $\mathsf{Integral}^\uparrow$ dependency is consumed only as a displayed finite contour, averaging, or boundary-read row, not as an ambient selection principle.

The provenance row may cite $\mathsf{HadamardThreeCircle}^\uparrow$ and $\mathsf{MetricAnnulus}^\uparrow$ as neighboring complex-analysis certificates, but the carrier keeps its strip row S and height ledger Y_0, Y_1, Y. Therefore a consumer receives exactly the finite strip ledger and no annulus collapse, selected extremal point, quotient equality, or infinite limiting data. □

Theorem 3893.4 (Hadamard three-lines convexity window)**.** *Any strip-bound consumer that reads the logarithmic-convexity estimate row E of an accepted* $\mathsf{HadamardThreeLines}^{\uparrow}$ *carrier must also display the strip row S, holomorphic map row F, boundary and interior height rows Y_0, Y_1, Y, boundary-line readback row B, and finite majorant row M. Thus the convexity estimate cannot be detached from the finite vertical-strip ledger that supplies its boundary and interior witnesses.*

Proof. Project the carrier in Definition 3893.1. The row E is introduced only as the logarithmic-convexity estimate tying Y_0, Y_1, Y and M to the same displayed strip-bound certificate, while M itself records finite boundary-majorant data read from B.

Apply the strip-bound route of Definition 3893.3. That theorem sends every accepted consumer through $S, F, Y_0, Y_1, Y, B, M, E, Hs, C, P, N$ before exporting a bound. Hence a read of E without S, F, Y_0, Y_1, Y, B, M would no longer be a read of the accepted $\mathsf{HadamardThreeLines}^{\uparrow}$ packet, but an external estimate outside the displayed finite ledger. □

Falsifiable prediction: An accepted $\mathsf{HadamardThreeLines}^{\uparrow}$ read is refuted if the logarithmic-convexity estimate row E is preserved while the displayed strip row S, holomorphic map row F, boundary and interior height reads Y_0, Y_1, Y, boundary-line row B, and finite majorant row M are removed.

Independence witness: hadamard-three-circle, holomorphic, complex, real, integral, metric-annulus: none is bijective with $\mathsf{HadamardThreeLines}^{\uparrow}$ because the packet jointly carries a vertical-strip row, holomorphic map row, boundary and interior height reads, boundary-line probes, finite majorant rows, a logarithmic-convexity estimate ledger, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}$.

Constructive story: A $\mathsf{HadamardThreeLines}^{\uparrow}$ packet is generated from a displayed vertical strip, a holomorphic map row, two boundary-height Real reads, one interior-height Real read, boundary-line probes, finite majorant rows, a logarithmic-convexity estimate ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3893.1, Definition 3893.2, and Definition 3893.3 seed the finite vertical-strip ledger for three-lines holomorphic strip estimates.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardThreeLines}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient function equality, host function extensionality, selected boundary suprema, unbounded strip limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked strip admission, holomorphic-map exposure, boundary and interior height ordering, boundary-line readback, finite majorant rows, logarithmic-convexity estimate compatibility, and structural transport for BEDC.Derived.HadamardThreeLinesUp.

Chapter 3894

A Concrete Naming Certificate for $\mathsf{PhragmenLindelof}^{\uparrow}$

Following Phragmen and Lindelof, the $\mathsf{PhragmenLindelof}^{\uparrow}$ packet records the unbounded-strip companion to the bounded vertical-strip ledger of chapter 3893. It sits over the holomorphic carrier of chapter 68, the complex carrier of chapter 39, the terminal real-name seal of chapter 38, the bounded-strip estimate of chapter 3893, and the boundary maximum surface of chapter 6213. The packet names finite boundary, growth, and exhaustion rows for an unbounded strip; it does not import host maximum principles, selected suprema, open-cover compactness, or ambient real completeness.

Definition 3894.1 (Phragmen–Lindelof carrier)**.** *A* $\mathsf{PhragmenLindelof}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, F, B, G, E, T, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the unbounded vertical-strip row, displaying the two boundary lines and the exhaustion windows that approximate the strip by bounded sub-strips.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row on the displayed strip, with values in the* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *B is the boundary-line readback row, exposing only finite probes on the two boundary lines.*
4. *G is the growth-control row, recording the finite majorant data allowed along the unbounded direction.*
5. *E is the bounded-exhaustion row, routing each inspected window through the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip ledger before any unbounded-strip bound is read.*
6. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *bound row produced by the exhaustion schedule.*
7. *M is the maximum-modulus boundary handoff row, usable only as a displayed local boundary row from chapter 6213.*
8. *H records componentwise* hsame *transport for the strip, map, boundary, growth, exhaustion, bound, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an unbounded-strip request through S, F, B, G, E, T, M, H.*
10. *P is the* Pkg *provenance over* $\mathsf{PhragmenLindelof}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MaximumModulusPrinciple}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PhragmenLindelof}^\uparrow}$ *row.*
The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a host supremum, quotient function equality, a selected boundary point, countable choice, open-cover compactness, or an ambient maximum principle.

Lean4 (statement) `BEDC.Derived.PhragmenLindelofUp`

Theorem 3894.2 (Phragmen–Lindelof NameCert obligations)**.** *For every accepted* $\mathsf{PhragmenLindelof}^\uparrow$ *carrier* $L = (S, F, B, G, E, T, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PhragmenLindelof}^\uparrow}$ *surface is exhausted by unbounded-strip admission, holomorphic-map exposure, boundary-line readback, growth-control display, bounded-exhaustion routing, terminal* $\mathsf{Real}^\uparrow$ *bound production, maximum-modulus boundary handoff, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet may not infer a host maximum principle, a selected boundary supremum, open-cover compactness, quotient function equality, countable choice, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 3894.1. The rows S, F, B, G display the unbounded strip, holomorphic source, boundary reads, and growth-control data before an estimate can be consumed.

The exhaustion is finite at each read. The row E sends every inspected window through the bounded-strip route of chapter 3893, and T records the resulting $\mathsf{Real}^\uparrow$ bound for the displayed schedule.

Finally M, H, C, P, N provide only the local boundary handoff, transport, replay, provenance, and naming rows. Since none of these rows adds selected suprema, host maximum principles, open-cover compactness, quotient function equality, countable choice, or ambient Real completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 3894.3 (Phragmen–Lindelof strip boundary handoff)**.** *Every unbounded-strip consumer of an accepted* $\mathsf{PhragmenLindelof}^\uparrow$ *carrier factors through*

$$\mathsf{Holomorphic}^\uparrow \longmapsto \mathsf{Complex}^\uparrow \longmapsto \mathsf{HadamardThreeLines}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

together with the displayed rows $S, F, B, G, E, T, M, H, C, P, N$. The handoff exports a finite exhaustion ledger for strip growth control; it does not export a host maximum principle, a selected extremal boundary point, an open-cover compactness argument, or quotient equality of holomorphic sources.

Proof. Read the strip source first. The packet exposes the unbounded-strip row S and the $\mathsf{Holomorphic}^\uparrow$ map row F over the $\mathsf{Complex}^\uparrow$ carrier before any boundary estimate is available.

Next open the boundary and growth rows. The boundary-line row B supplies finite probes, while G supplies the displayed growth majorant data that permits the bounded-window exhaustion to be read.

The handoff is then forced through the bounded companion. The row E routes each inspected window through the $\mathsf{HadamardThreeLines}^\uparrow$ ledger, and T seals the resulting bound as a $\mathsf{Real}^\uparrow$ row for the same displayed exhaustion schedule.

The remaining rows preserve locality. The boundary handoff M, transport H, replay C, provenance P, and local naming N do not introduce a selected supremum, host maximum principle, open-cover compactness, quotient function equality, or ambient completeness assertion. □

Falsifiable prediction: An accepted $\mathsf{PhragmenLindelof}^\uparrow$ read is refuted if the bounded-exhaustion row E is preserved while the displayed unbounded-strip row S, holomorphic map row F, boundary-line row B, growth-control row G, and terminal $\mathsf{Real}^\uparrow$ bound row T are removed.

Independence witness: hadamard-three-lines, holomorphic, complex, real, maximum-modulus-principle: none is bijective with $\mathsf{PhragmenLindelof}^{\uparrow}$ because the packet jointly carries an unbounded-strip row, holomorphic source row, finite boundary probes, growth-control data, bounded-exhaustion routing, terminal real bound, local maximum-modulus boundary handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PhragmenLindelof}^{\uparrow}}$.

Constructive story: A $\mathsf{PhragmenLindelof}^{\uparrow}$ packet is generated from an unbounded vertical strip, a holomorphic map row, boundary-line probes, growth-control data, bounded-window exhaustion through HadamardThreeLines, a terminal Real bound, local maximum-modulus boundary handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhragmenLindelof}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3894.1, Definition 3894.2, and Definition 3894.3 seed the unbounded-strip growth-control packet over HadamardThreeLines, Holomorphic, Complex, Real, and MaximumModulusPrinciple rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhragmenLindelof}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host maximum principles, selected boundary suprema, open-cover compactness, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked theorem obligations, or bridge checking.

Upgrade path: Obligation closure requires checked unbounded-strip admission, holomorphic-map exposure, boundary-line readback, growth-control display, bounded-exhaustion routing, terminal Real bound production, maximum-modulus boundary handoff, and structural transport for BEDC.Derived.PhragmenLindelofUp.

Chapter 3895

A Concrete Naming Certificate for $\mathsf{CauchyLocatedApartness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyLocatedApartness}^{\uparrow}$ packet records the constructive boundary where finite locator requests for a Cauchy real become positive located-apartness evidence. It consumes the Cauchy locator route of chapter 3410, the located-apartness surface of chapter 1380, the apartness-real interface of chapter 304, and the real-name seal of chapter 38. Its finite naming spine is read through chapter 126, chapter 301, and chapter 291.

The packet is a local handoff surface. It exports locator rows, endpoint-gap rows, finite stream windows, regular-rational readbacks, dyadic tolerance ledgers, real seals, transport rows, continuation replay, provenance, and a local NameCert row. It exports no trichotomy, no quotient real equality, no selected limit, no countable choice, no open-cover compactness, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 3895.1 (Cauchy located-apartness carrier)**.** *A* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, A, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row answering finite rational locator requests for the Cauchy real being inspected.*
2. *A and B are endpoint-gap rows read through* $\mathsf{LocatedApartness}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$*, recording positive separation windows on the two sides of the located comparison.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger used to schedule the locator observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed stream windows into regular rational observations.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the radius and endpoint arithmetic used by the locator and apartness rows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, A, B, S, R, D have been displayed.*
7. *H stores componentwise* hsame *transport for locator rows, endpoint-gap rows, finite windows, readbacks, tolerance ledgers, real seals, continuation replay, provenance, and local naming.*
8. *C stores displayed* Cont *replay from locator requests through stream windows, regular-rational readbacks, dyadic tolerance checks, endpoint-gap rows, and the final real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyLocatedApartness}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, quotient real equality, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessTasteGate single carrier alignment`

Theorem 3895.2 (Cauchy located-apartness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier* $L = (Q, A, B, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, locator-window compatibility, endpoint-gap exactness, dyadic-tolerance coverage, real-seal handoff, transport replay, and non-escape from the finite route*

$$Q, S, R, D \longmapsto A, B \longmapsto E$$

together with H, C, P, N*.*

Proof. Carrier admission is projection from Definition 3895.1. The accepted tuple exposes the Cauchy locator row, two endpoint-gap rows, stream windows, regular-rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming.

Locator-window compatibility reads Q only through S, R, D. A finite comparison request must display the scheduled stream window, its regular rational readback, and the tolerance ledger before any located-apartness row can be consumed.

Endpoint-gap exactness is supplied by A and B. These rows carry the positive located-apartness evidence, while the real seal E is downstream of the same locator and tolerance route.

Finally, H, C, P, N preserve and name the route. Since the carrier contains no trichotomy, quotient equality, selected limit, countable-choice schedule, open-cover compactness theorem, or ambient real-completeness coordinate, the displayed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessCarrier namecert obligations`

Theorem 3895.3 (Cauchy locator to apartness handoff)**.** *Every accepted* $\mathsf{CauchyLocatedApartness}^{\uparrow}$ *carrier converts a finite* $\mathsf{CauchyRealLocator}^{\uparrow}$ *request into positive* $\mathsf{LocatedApartness}^{\uparrow}$ *and* $\mathsf{ApartnessReal}^{\uparrow}$ *endpoint-gap evidence only through the displayed route*

$$Q \longmapsto S \longmapsto R \longmapsto D \longmapsto A, B \longmapsto E.$$

Thus a downstream Real-facing comparison consumer receives a finite locator-apartness handoff, not trichotomy, quotient real equality, a selected limit, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyLocatedApartnessUp.CauchyLocatedApartnessLocatorApartnessHandoff`

Proof. Start with a finite locator request in Q. The carrier forces the request to replay through the scheduled $\mathsf{StreamName}^{\uparrow}$ window S, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger D.

Only after that route is displayed may the endpoint-gap rows A and B be read as located-apartness evidence. The $\mathsf{Real}^{\uparrow}$ seal E records that the same finite route reaches the real-facing surface.

The structural rows H, C, P, N transport, replay, package, and name exactly these coordinates. No excluded principle is a field of the carrier, so the handoff is exhausted by the finite locator, window, readback, tolerance, apartness, and seal rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3895.1, Definition 3895.2, and Definition 3895.3 seed the finite Cauchy locator to located-apartness boundary over stream windows, regular-rational readbacks, dyadic tolerance ledgers, endpoint-gap rows, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyLocatedApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, locator-window compatibility, endpoint-gap exactness, dyadic-tolerance coverage, real-seal handoff, transport replay, and non-escape for BEDC.Derived.CauchyLocatedApartnessUp.

Constructive story: A $\mathsf{CauchyLocatedApartness}^{\uparrow}$ packet is generated from a CauchyRealLocator request row, LocatedApartness and ApartnessReal endpoint-gap rows, finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3896

A Concrete Naming Certificate for $\mathsf{RiemannLebesgueLemma}^{\uparrow}$

Following Zygmund, Katznelson, and Folland on Fourier analysis, the $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet records the finite oscillatory-decay surface of the Riemann–Lebesgue lemma as BEDC NameCert data. It sits over the Fourier coefficient interface of chapter 124, the circle phase sampler of chapter 58, the integral normalization route of chapter 123, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the averaging comparison supplied by chapter 1939. The packet exposes finite coefficient windows and oscillatory handoff data; it does not import Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3896.1 (Riemann–Lebesgue lemma carrier)**.** *A* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, S, I, W, T, A, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row, naming exactly the inspected frequency indices and coefficient requests.*
2. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-sample row, recording the finite circle phases used to test the requested oscillations.*
3. *I is the* $\mathsf{Integral}^{\uparrow}$ *normalization and readback row, exposing only the finite integral observations used to form each coefficient.*
4. *W is the window ledger pairing frequency indices, phase samples, and integral cells before any decay read is made.*
5. *T is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, listing the rational error budgets attached to the coefficient windows.*
6. *A is the finite averaging comparison row, allowed to cite the* $\mathsf{FejerKernel}^{\uparrow}$ *sibling only as a displayed smoothing comparison.*
7. *Q is the oscillatory-decay handoff row, recording which finite tails have coefficient magnitude below the displayed tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for coefficient values, tolerance comparisons, and accepted decay reads.*

9. *H records componentwise* hsame *transport for the coefficient, phase, integral, window, tolerance, averaging, decay, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a Fourier-coefficient request through $F, S, I, W, T, A, Q, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, I, W, T, A, Q, E, H, C, P, N$. It has no coordinate for Haar measure, host L^1 spaces, quotient equality of functions, a theorem of general Fourier convergence, selected representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RiemannLebesgueLemmaUp.RiemannLebesgueLemmaTasteGate single carrier alignment`

Theorem 3896.2 (Riemann–Lebesgue lemma NameCert obligations)**.** *For every accepted* RiemannLebesgueLemma *carrier*

$$R = (F, S, I, W, T, A, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$ *surface consists exactly of Fourier coefficient-window admission through F, circle phase-sample admission through S, integral normalization and readback through I, window and tolerance exposure through W, T, finite averaging comparison through A, oscillatory-decay handoff through Q, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3896.1. The rows F, S, I expose the coefficient request, the finite phase samples, and the integral normalization before the packet can state any decay handoff.

The rows W, T, A, Q supply the finite analytic body of the certificate. The window row pairs frequencies with inspected cells, the tolerance row records the rational budgets, the averaging row gives the displayed Fejer-kernel comparison, and the handoff row records exactly the tails accepted below those budgets.

The structural rows E, H, C, P, N seal, transport, replay, source, and name the same data. Since the carrier has no coordinate for Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, selected representatives, or ambient real completeness, the listed rows exhaust the NameCert surface. □

Theorem 3896.3 (Riemann–Lebesgue oscillatory handoff)**.** *Every Fourier-decay consumer of an accepted* $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Fourier}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The optional $\mathsf{FejerKernel}^{\uparrow}$ *comparison can be read only through the finite averaging row A, and the final decay read can inspect only $F, S, I, W, T, A, Q, E, H, C, P, N$. Thus the packet supplies a finite oscillatory-decay dependency for later Fourier approximation chapters without exporting Haar measure, host L^1 integration, quotient function equality, general convergence of Fourier series, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Begin with a Fourier-decay request. Carrier projection exposes F, so the consumer receives a finite coefficient window rather than an infinite Fourier transform.

The request next reads S, I, W, T. These rows bind each requested frequency to displayed circle phases, integral cells, window ledgers, and regular rational tolerances.

Only then can the consumer inspect A, Q, E, H, C, P, N. The averaging comparison remains a finite sibling comparison, the decay handoff is recorded at the displayed tolerances, and the real seal closes the numeric read. No excluded analytic object has a coordinate in the packet, so the handoff is exactly the displayed finite route. □

Lean4 ✓ `BEDC.Derived.RiemannLebesgueLemmaUp.OscillatoryHandoff.RiemannLebesgueLemmaOscillatoryHandoff`

Falsifiable prediction: An accepted $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet is refuted if a Fourier-decay consumer preserves the coefficient window while bypassing the displayed $\mathsf{S}^{1\uparrow}$ phase samples, $\mathsf{Integral}^{\uparrow}$ normalization row, $\mathsf{RegSeqRat}^{\uparrow}$ tolerance row, and terminal $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: fourier, sone, integral, regseqrat, real, fejer-kernel: none is bijective to $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ because the packet jointly carries coefficient windows, phase samples, integral normalization, tolerance ledgers, averaging comparison, oscillatory-decay handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannLebesgueLemma}^{\uparrow}$ packet is generated from finite Fourier coefficient windows, SOne phase samples, Integral normalization and readback rows, RegSeqRat tolerance rows, a finite FejerKernel comparison, an oscillatory handoff row, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3896.1, Definition 3896.2, and Definition 3896.3 seed the finite oscillatory-decay route over Fourier, circle phase, integral, regular rational, real-seal, and averaging rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannLebesgueLemma}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RiemannLebesgueLemmaUp.OscillatoryHandoff.RiemannLebesgueLemmaOscillatoryHandoff`

Bridge status: none

Not claimed: This chapter does not claim Haar measure, host L^1 theory, quotient function equality, general Fourier convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked coefficient-window admission, phase-sample exposure, integral normalization, tolerance exactness, finite averaging comparison, oscillatory handoff, real seal exposure, transport, replay, provenance, and local naming for BEDC.Derived.RiemannLebesgueLemmaUp.

Chapter 3897

A Concrete Naming Certificate for $\mathsf{MedianClosureSpectrum}^{\uparrow}$

The $\mathsf{MedianClosureSpectrum}^{\uparrow}$ packet realizes the Walsh-spectrum reassignment rows of Definition 9462.1263, Definition 9462.1264, and Definition 9462.1273 as a finite NameCert spectrum. It reads the finite codon-window and module-boundary surfaces of **??** and **??**, then records the dominant Walsh coordinates as median-closure support data rather than as host spectral analysis. The packet does not depend on Fourier analysis, Fejer kernels, Parseval identities, finite phase sampling, or Dirichlet kernels.

Definition 3897.1 (Median closure spectrum carrier)**.** *A* $\mathsf{MedianClosureSpectrum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, W, D, S, L, E, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row whose coordinates are read by the reassignment spectrum.*
2. *W is the Walsh coefficient ledger, listing the displayed coefficients of Definition 9462.1264.*
3. *D is the dominant-coordinate row for b_3^s, $b_1^f b_3^s$, b_1^f, $b_1^f b_2^s$, $b_1^f b_2^s b_3^s$, $b_2^s b_3^s$, and b_2^s.*
4. *S is the median-closure support row, recording which dominant coordinates support the reassignment median boundary.*
5. *L is the finite low-order energy ledger read from the compression statement, with no appeal to an analytic Fourier transform.*
6. *E is the evidence row tying the Walsh ledger to the purine-core and module-classification surfaces.*
7. *R is the refusal row for Fourier, Fejer-kernel, Parseval, finite-phase-sampling, and Dirichlet-kernel dependencies.*
8. *H records componentwise* hsame *transport for the codon window, Walsh ledger, dominant-coordinate row, support row, energy ledger, evidence row, refusal row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a dominant-coordinate read through Q, W, D, S, L, E, R, H.*
10. *P is the* Pkg *provenance over* $\mathsf{MedianClosureSpectrum}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *together with the cited vision rows and sibling codon-boundary packets.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, W, D, S, L, E, R, H, C, P, N$. It has no coordinate for host Fourier transform data, trigonometric phase sampling, Parseval energy identities, Dirichlet kernels, Fejer kernels, or a hidden analytic spectrum outside the displayed Walsh ledger.

Lean4 ✓ `BEDC.Derived.MedianClosureSpectrumUp.MedianClosureSpectrumTasteGate single carrier alignment`

Theorem 3897.2 (Median closure spectrum NameCert obligations)**.** *For every accepted* $\mathsf{MedianClosureSpectrum}^{\uparrow}$ *carrier*

$$M = (Q, W, D, S, L, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$ *surface consists exactly of codon-window admission through Q, Walsh coefficient exposure through W, dominant-coordinate readback through D, median-closure support exposure through S, low-order energy accounting through L, purine-core and module-classification evidence through E, analytic-dependency refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3897.1. The rows Q, W, D expose the finite codon window, Walsh coefficient ledger, and dominant coordinates before any median-closure support read is allowed.

The rows S, L, E supply the support body of the packet. They identify the median-closure support row, record the low-order energy ledger, and tie those rows to the purine-core and module-classification siblings cited above.

The rows R, H, C, P, N close the certificate. The refusal row blocks analytic substitutions, while transport, replay, provenance, and local naming preserve the displayed finite Walsh spectrum as the whole accepted surface. □

Falsifiable prediction: Every dominant-coordinate read accepted by $\mathsf{MedianClosureSpectrum}^{\uparrow}$ factors through the displayed Walsh coefficient ledger W; replacing W by Fourier, phase-sampling, Parseval, Fejer-kernel, or Dirichlet-kernel data refutes the packet.

Independence witness: fourier, fejer-kernel, parseval, finite-phase-sampling, dirichlet-kernel: the median-closure spectrum is not bijective to any of these routes because its accepted reads are finite Walsh coefficient ledger reads over codon-window and support rows, not analytic phase, convolution, energy, or kernel reads.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{MedianClosureSpectrum}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MedianClosureSpectrum}^{\uparrow}$ instances in `TasteGate.lean`. The field-faithfulness witness must keep $Q, W, D, S, L, E, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out analytic-spectrum replacements of the displayed Walsh ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MedianClosureSpectrum}^{\uparrow}$ packet is generated from finite codon-window rows, Walsh coefficient ledgers, dominant-coordinate rows, median-closure support rows, low-order energy rows, purine-core and module-classification evidence, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MedianClosureSpectrum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3897.1 and Definition 3897.2 seed the finite Walsh-ledger spectrum packet over codon-window, dominant-coordinate, median-support, energy, evidence, refusal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MedianClosureSpectrum}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Fourier analysis, Fejer kernels, Parseval identities, finite phase sampling, Dirichlet kernels, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, Walsh ledger exposure, dominant-coordinate readback, median-support exactness, energy ledger accounting, evidence routing, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MedianClosureSpectrumUp.

Chapter 3898

A Concrete Naming Certificate for $\mathsf{DirectedFilter}^{\uparrow}$

Following Kelley and Bourbaki on directed sets, filters, and completion by tails, the $\mathsf{DirectedFilter}^{\uparrow}$ packet records the finite carrier that translates directed tail requests into a filter-base handoff before any completion consumer reads a limit-facing surface. It sits between the directed-set substrate of chapter 1607, the filter-base certificate of chapter 1601, the Moore–Smith net route of chapter 3403, and the filter-limit basis route of chapter 3862. The packet supplies the L10 completion corridor with a shared directed-filter carrier; it is not a quotient filter, selected limit, countable-choice selector, host equality of filters, or ambient completeness theorem.

Definition 3898.1 (DirectedFilter carrier)**.** *A* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, B, M, U, W, R, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DirectedSet}^{\uparrow}$ *tail-request row, recording finite requests, preorder comparisons, and common-upper witnesses.*
2. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *handoff row assembled only from the displayed tails in D.*
3. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *and* $\mathsf{NetConvergence}^{\uparrow}$ *route row, exposing the same directed tails as net-index evidence before any convergence read.*
4. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *and* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row, available only after D, B, M have been replayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by completion-facing numerical consumers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W.*
7. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the readback rows before a seal is consumed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, downstream of directed tails, filter-base handoff, net convergence, completion consumers, windows, readback, and tolerance rows.*
9. *H records componentwise* hsame *transport for all analytic rows and the local structural rows.*
10. *C records displayed* Cont *replay from a directed-tail request through $D, B, M, U, W, R, Y, E, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{DirectedFilter}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for quotient filters, ultrafilters, selected limits, countable choice, host filter equality, or ambient completion.

Lean4 ✓ `BEDC.Derived.DirectedFilterUp.DirectedFilterTasteGate single carrier alignment`

Theorem 3898.2 (DirectedFilter NameCert obligations)**.** *For every accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier* $F = (D, B, M, U, W, R, Y, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$ *surface consists exactly of the finite obligations for carrier admission, directed-tail coverage, filter-base handoff, net-convergence routing, completion nonescape, numerical readback, componentwise transport, displayed replay, provenance, and local naming. Every obligation is read from* $D, B, M, U, W, R, Y, E, H, C, P, N$*, and no obligation is stored in a quotient filter, selected limit, countable-choice schedule, host equality relation, or ambient completeness witness.*

Proof. Project the carrier of Definition 3898.1. The displayed analytic rows are D, B, M, U, W, R, Y, E, and the structural rows are H, C, P, N.

Directed-tail coverage is supplied by D. Any request first reads the finite tail window and its common-upper witness before a filter or net consumer can inspect the packet.

The handoff and consumer rows are downstream. The row B exposes the filter-base surface named in chapter 1601, while M routes the same tails to the Moore–Smith and net-convergence surface of chapter 3403. The row U then presents only metric or uniform completion consumers of that same route.

Finally W, R, Y, E provide the numerical window, readback, dyadic tolerance, and real seal rows. Transport, replay, provenance, and naming preserve the same finite carrier. Since none of the excluded principles has a carrier coordinate, the listed obligations exhaust the local surface. □

Theorem 3898.3 (DirectedFilter tail FilterBase handoff)**.** *Every* $\mathsf{FilterBase}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *carrier must first read the* $\mathsf{DirectedSet}^{\uparrow}$ *tail-request row D and only then the handoff row B. The handoff presents finite tails, common-upper witnesses, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it does not present a quotient filter, ultrafilter, selected limit, countable-choice selector, or host filter equality.*

Proof. Begin with the NameCert obligations of Definition 3898.2. They place every filter-facing read after the displayed directed-tail row D.

Read B next. Because B is assembled only from the finite tails and upper-window witnesses in D, the filter-base consumer receives the same displayed family rather than an external filter object.

The structural rows H, C, P, N transport, replay, source, and name that handoff. No omitted quotient, ultrafilter, selection, or host-equality row can be recovered from the carrier. □

Theorem 3898.4 (DirectedFilter net-convergence route)**.** *Every* $\mathsf{MooreSmithNet}^{\uparrow}$ *or* $\mathsf{NetConvergence}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *packet factors through*

$$\mathsf{DirectedSet}^{\uparrow} \longmapsto \mathsf{FilterBase}^{\uparrow} \longmapsto \mathsf{MooreSmithNet}^{\uparrow}, \mathsf{NetConvergence}^{\uparrow} \longmapsto H, C, P, N.$$

The route shares the directed-tail ledger with chapter 3403 and the filter-base handoff with chapter 1601. It exports no global confluence of filters and nets, no selected subnet, no selected limit, and no ambient convergence theorem.

Proof. Project F to D, B, M, H, C, P, N. The row D is the only directed-index source, and B is the only filter-base handoff.

The net row M is reached after those two rows. It therefore reads the same directed tails as the filter-base consumer, matching the Moore–Smith carrier rather than a separately chosen subnet or a detached convergence assertion.

Transport and replay keep the route componentwise. Since the packet contains no selected subnet, selected limit, or ambient convergence coordinate, every net-convergence read is exhausted by the displayed rows. □

Theorem 3898.5 (DirectedFilter completion nonescape)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DirectedFilter}^{\uparrow}$ *packet reads the completion corridor through*

$$D, B, M, U \longmapsto W, R, Y, E \longmapsto H, C, P, N.$$

The route passes through the filter-limit basis corridor of chapter 3862 without adding quotient filters, selected limits, countable choice, host real equality, or ambient completeness.

Proof. Use Definition 3898.3 to force the directed-tail and filter-base prefix. Then use Definition 3898.4 to place the net-convergence row before any completion-facing consumer.

The completion row U is followed by the numerical rows W, R, Y, E. These rows expose finite stream windows, regular rational readback, dyadic tolerance accounting, and the terminal real seal in the order displayed by the carrier.

The structural rows H, C, P, N preserve and name the same route. Because the carrier has no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, the completion consumer cannot escape the stated corridor. □

Closure status for $\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{DirectedFilter}^{\uparrow}$ packet is generated from a DirectedSet tail-request row, a FilterBase handoff row, MooreSmithNet and NetConvergence route rows, UniformCompletion and MetricCompletion consumer rows, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3898.1, Definition 3898.2, Definition 3898.3, Definition 3898.4, and Definition 3898.5 seed the finite directed-filter carrier for the L10 completion corridor.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirectedFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, ultrafilter arguments, selected limits, selected subnets, countable choice, host filter equality, ambient convergence, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail coverage, filter-base handoff, net-convergence routing, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DirectedFilterUp.

Chapter 3899

A Concrete Naming Certificate for $\mathsf{OscillationFunction}^{\uparrow}$

Following Baire's function hierarchy and Bishop–Bridges constructive analysis, the $\mathsf{OscillationFunction}^{\uparrow}$ packet records the finite oscillation certificate associated to a pointwise real-valued limit. It sits after the pointwise-limit route of chapter 2314, the Baire-one consumer surface of chapter 1604, the Dini-continuity consumer surface of chapter 2196, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed oscillation windows and value-readback rows; it does not import a host function space, quotient real equality, a selected supremum, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3899.1 (OscillationFunction carrier)**.** *An* $\mathsf{OscillationFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (P, B, D, S, R, V, H, C, K, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *source row whose displayed point-index and approximant-window route is being inspected.*
2. *B is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *consumer row receiving the oscillation certificate as Baire-one boundary data.*
3. *D is the* $\mathsf{DiniContinuity}^{\uparrow}$ *consumer row that may read the same finite oscillation windows before any uniform-modulus handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finitely many source neighbourhoods and approximant windows used by the oscillation request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected value differences and tolerance comparisons.*
6. *V is the terminal* $\mathsf{Real}^{\uparrow}$ *oscillation-value seal reached only after P, B, D, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the pointwise-limit, Baire-one, Dini-continuity, schedule, readback, value-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an oscillation request through the pointwise-limit windows, the Baire-one and Dini consumer rows, the stream schedule, the regular rational readback, and the RealUp seal.*
9. *K is the* Pkg *provenance over* $\mathsf{OscillationFunction}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationFunction}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host oscillation function, quotient real equality, a selected supremum over all neighbourhoods, an ambient Baire-one theorem, or completeness of $\mathsf{Real}^\uparrow$*.*

Lean4 ✓ `BEDC.Derived.OscillationFunctionUp.OscillationFunctionTasteGate single carrier alignment`

Theorem 3899.2 (OscillationFunction NameCert obligations)**.** *For every accepted* $\mathsf{OscillationFunction}^\uparrow$ *carrier $O = (P, B, D, S, R, V, H, C, K, N)$, the local* $\mathsf{NameCert}_{\mathsf{OscillationFunction}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the pointwise-limit source, Baire-one consumer, Dini-continuity consumer, stream schedule, regular rational readback, RealUp oscillation seal, transport, replay, provenance, and local naming row;*
2. *pointwise-window exposure, requiring every oscillation request to read the* $\mathsf{PointwiseLimit}^\uparrow$ *route of chapter 2314 before any value-difference comparison is inspected;*
3. *consumer compatibility, requiring the* $\mathsf{BaireOneFunction}^\uparrow$ *and* $\mathsf{DiniContinuity}^\uparrow$ *rows to consume the same displayed windows and not a second host function-space object;*
4. *Real-seal exactness, requiring the oscillation value row V to be downstream of P, B, D, S, R and transported only by H and C;*
5. *nonescape from host oscillation functions, quotient real equality, selected suprema, ambient Baire-one representation theorems, unrestricted Dini compactness, and completeness of* $\mathsf{Real}^\uparrow$*.*

No NameCert obligation is stored outside $P, B, D, S, R, V, H, C, K, N$.

Proof. Project the carrier of Definition 3899.1. The accepted packet exposes exactly the pointwise-limit source row, the Baire-one and Dini consumer rows, the finite stream schedule, the regular rational readback, the RealUp value seal, transport, replay, provenance, and local naming.

The pointwise-window exposure follows from the source row P. The route cited in chapter 2314 already displays the point-index, approximant windows, regular rational readback, and RealUp value seal before a pointwise value can be consumed.

Consumer compatibility is then read through B and D. Both rows are coordinates of the same packet, so the Baire-one and Dini-facing consumers receive the finite schedule S and readback R already attached to P rather than an external function-space witness.

Finally the seal row V is downstream of this route. Componentwise transport by H and replay by C preserve only the displayed rows, while K and N carry provenance and local naming. Since the excluded objects are not coordinates of the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3899.3 (OscillationFunction pointwise readback)**.** *Every* $\mathsf{Real}^\uparrow$ *oscillation-value read exported by an accepted* $\mathsf{OscillationFunction}^\uparrow$ *carrier factors through the displayed route*

$$\mathsf{PointwiseLimit}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as recorded by P, S, R, V. The readback may be consumed by the $\mathsf{BaireOneFunction}^\uparrow$ *and* $\mathsf{DiniContinuity}^\uparrow$ *rows B, D, but it cannot bypass the pointwise-limit route by reading a host oscillation function, quotient real equality, selected neighbourhood supremum, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Begin with a consumer request for V. By the carrier definition, V is reached only after the pointwise-limit source P, the finite schedule S, and the regular rational readback R have been displayed.

Apply the pointwise-limit Real-seal handoff from chapter 2314. It supplies the route from displayed source requests and approximant windows to a terminal RealUp value seal, and the oscillation packet reuses that route for finite value-difference comparisons.

The Baire-one and Dini rows are downstream consumers of the same readback. Transport through H and replay through C do not add a host function coordinate or a selected supremum row, so every oscillation-value read remains confined to P, S, R, V and the visible package rows. □

Theorem 3899.4 (OscillationFunction Baire-one handoff)**.** *Every* $\mathsf{BaireOneFunction}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OscillationFunction}^{\uparrow}$ *packet must read the pointwise-limit source P, the Baire-one row B, the Dini-continuity row D, the stream schedule S, the regular rational readback R, the RealUp oscillation seal V, and the structural rows H, C, K, N before any Baire-one oscillation boundary is public. Thus the handoff is a finite NameCert route over the rows displayed in Definition 3899.1, not a representation theorem about an ambient space of functions.*

Proof. Read the obligation surface of Definition 3899.2. It places the Baire-one consumer row behind the same pointwise-window exposure and Real-seal exactness rows used by the oscillation value.

Use the Baire-one packet of chapter 1604. Its public pointwise-limit ledger accepts displayed continuous approximant windows and regular rational readbacks, not a host function-space quotient. The oscillation carrier supplies those rows through P, S, R, V.

The Dini row D records the companion compact-modulus consumer boundary, but it is still downstream of the same displayed schedule. Since H, C, K, N transport, replay, source, and name only that finite route, the Baire-one handoff exports no ambient representation theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OscillationFunction}^{\uparrow}$ packet is generated from a PointwiseLimitUp source row, BaireOneFunctionUp and DiniContinuityUp consumer rows, StreamName schedules, RegSeqRat readbacks, a RealUp oscillation-value seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3899.1, Definition 3899.2, Definition 3899.3, and Definition 3899.4 seed the finite oscillation certificate over pointwise-limit windows and Baire-one consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host oscillation functions, quotient real equality, selected suprema, ambient Baire-one representation, unrestricted Dini compactness, completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise-window exposure, consumer compatibility, Real-seal exactness, and nonescape for BEDC.Derived.OscillationFunctionUp.

Chapter 3900

A Concrete Naming Certificate for $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ packet records the finite universal-property surface for the L10 filter-completion route. It sits after the completion packet of chapter 1569, the convergence boundary of chapter 1558, the universal extension surface of chapter 397, and the uniform handoff of chapter 1473. The packet names the extension and uniqueness rows that downstream completion consumers may cite without importing quotient filters, selected limits, host equality, countable choice, or ambient completeness.

Definition 3900.1 (Filter completion universal-property carrier)**.** *A* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, L, E, Q, F, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion route, already tying the accepted read to a displayed Cauchy filter, finite windows, dyadic tolerances, regular readback, and a terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
2. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *convergence row consumed only after the completion route C has exposed its source-filter and readback rows.*
3. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension-and-uniqueness row attached to the same filter-completion route.*
4. *Q is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row read as provenance for uniform-facing consumers, not as a detached completion object.*
5. *F is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row supplied by the same finite windows and dyadic tolerance ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal reached through C, L, F and used as the terminal filter-completion target.*
7. *A is the extension-uniqueness ledger, comparing accepted extension rows over the same C, L, E, Q, F, R surface by displayed* hsame *transport.*
8. *H records componentwise* hsame *transport for the completion route, convergence row, universal-property row, uniform handoff, regular readback, Real seal, extension ledger, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay of the route*

$$C, L, F, R \longmapsto E, Q, A, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, L, E, Q, F, R, A, H, K, P, N$. It has no coordinate for quotient filters, selected filter limits, host equality of filters or real names, countable-choice selectors, ambient completeness, or a completion object detached from the displayed filter route.

Lean4 ✓ `BEDC.Derived.FilterCompletionUniversalPropertyUp.FilterCompletionUniversalPropertyTasteGate`
`single carrier alignment`

Theorem 3900.2 (Filter completion universal-property NameCert obligations)**.** *For every accepted* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *carrier $U = (C, L, E, Q, F, R, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple; filter-completion exposure through C; convergence exposure through L after the source-filter route has been read; regular readback through F and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R; completion-universal extension through E; uniform provenance through Q; uniqueness comparison through A over the same rows; componentwise transport through H; displayed replay through K; provenance through P; local naming through N; and ledger refusal for quotient filters, selected limits, host equality, countable choice, ambient completeness, and detached completion objects.*

Proof. Carrier admission is projection from Definition 3900.1. The accepted packet exposes exactly the Cauchy-filter completion route, convergence row, universal-property row, uniform handoff, regular readback, Real seal, uniqueness ledger, transport, replay, provenance, and local name.

The filter route is finite. The completion row C is read before the convergence row L, the regular readback F, and the seal R, so no consumer can begin from a selected filter limit or a quotient-filter representative.

The universal-property route is also finite. Rows E, Q, A are read only after the same filter-completion and Real-seal data have been displayed, while H, K, P, N transport, replay, source, and name that route. Since no excluded host-side coordinate appears in the carrier, these rows exhaust the local certificate surface. □

Theorem 3900.3 (Filter completion universal-property extension uniqueness)**.** *Let $U = (C, L, E, Q, F, R, A, H, K,$ be an accepted* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *packet. Any two extension rows consumed through U are compared only by the displayed uniqueness ledger A over the common route*

$$C, L, F, R \longmapsto E, Q \longmapsto A, H, K, P, N.$$

Thus extension uniqueness is available to completion-facing consumers only after the same Cauchy-filter completion row, convergence row, regular readback, Real seal, universal-property row, and uniform handoff have been displayed. It cannot be replaced by host function equality, quotient-filter equality, a selected limit, countable choice, or ambient completeness.

Proof. Start with two extension reads from the accepted packet. The carrier has a single universal-property row E and a single uniqueness ledger A attached to the displayed filter-completion route.

Replay the continuation row K. Both extension reads must pass through C, L, F, R before E and Q are available, so the comparison has the same source-filter, readback, convergence, and Real-seal data on both sides.

The ledger A then compares the two rows by displayed hsame transport, and H, P, N preserve and name the same comparison. The refused host equalities, quotient objects, selected limits, choice selectors, and completeness principles have no coordinate inside U. □

Theorem 3900.4 (Filter completion universal-property filter handoff)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ *packet reads the same finite handoff*

$$C, L, F, R, E, Q, A, H, K, P, N.$$

The handoff starts from the displayed Cauchy-filter completion row, reads the convergence boundary and regular readback before the terminal $\mathsf{Real}^{\uparrow}$ *seal, and then exposes extension, uniqueness, uniform provenance, transport, replay, provenance, and local naming. No consumer may replace that handoff by a quotient filter, selected filter limit, host equality, countable-choice selector, ambient completeness theorem, or detached completion carrier.*

Proof. Project the accepted carrier. Its only completion source is C, its only convergence row is L, its only regular readback and terminal seal rows are F and R, and its only universal-property and uniform-facing rows are E and Q.

Use the NameCert obligations of Definition 3900.2. They order the read from filter-completion exposure through convergence, readback, seal, extension, uniqueness, uniform provenance, transport, replay, package provenance, and local naming.

The sibling surfaces enforce the same boundary: chapter 1569 supplies the completion source, chapter 1558 supplies the convergence boundary, and chapter 1473 supplies only the uniform-facing provenance handoff. Since the carrier has no extra coordinate, the listed finite handoff is the whole exported route. □

Falsifiable prediction: Every accepted $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ consumer that exports extension existence, extension uniqueness, filter convergence, or uniform-completion handoff while omitting one of the displayed Cauchy-filter completion, Cauchy-filter limit, regular readback, Real seal, completion-universal, uniform provenance, transport, replay, package provenance, or local naming rows refutes the packet.

Independence witness: cauchyfiltercompletion, cauchyfilterlimit, completionuniversalproperty, uniformcompletion, regseqrat, real: none is bijective because $\mathsf{FilterCompletionUniversalProperty}^{\uparrow}$ jointly carries filter-completion source, convergence boundary, regular readback, terminal Real seal, extension, uniqueness, uniform provenance, transport, replay, package provenance, and local NameCert rows in one universal-property packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3900.1, Definition 3900.2, Definition 3900.3, and Definition 3900.4 seed the finite L10 filter-completion universal-property route over Cauchy-filter completion, filter convergence, regular readback, Real seal, universal extension, uniform provenance, transport, replay, package provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterCompletionUniversalProperty}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected filter limits, host equality, countable choice, ambient completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convergence exposure, extension uniqueness, filter handoff, uniform provenance, and non-escape proofs for BEDC.Derived.FilterCompletionUniversalPropertyUp.

Constructive story: A FilterCompletionUniversalProperty$^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, a CauchyFilterLimit convergence row, a CompletionUniversalProperty row, a UniformCompletion provenance row, a RegSeqRat readback, a RealUp seal, an extension-uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3901

A Concrete Naming Certificate for $\mathsf{FiniteChoiceWindow}^{\uparrow}$

Following Bishop–Bridges finite choice discipline, the $\mathsf{FiniteChoiceWindow}^{\uparrow}$ packet records only those selections whose request set is already displayed as a finite unary-indexed spine. It sits between the finite observation gateway of chapter 126, the regular rational readback route of chapter 301, and the dyadic tolerance ledger of chapter 291; it supplies the L10 StreamName and RegSeqRat route with a finite section, not a countable-choice principle or a selected limiting object.

Definition 3901.1 (Finite choice window carrier)**.** *A* $\mathsf{FiniteChoiceWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, A, S, D, R, H, C, P, N)$$

with the following rows.

1. *I is a displayed unary index spine naming the finite request positions.*
2. *A is the finite candidate-row ledger, with one nonempty candidate list attached to each position of I.*
3. *S is the selected finite-section row, containing exactly one selected candidate for each displayed position of I.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback row consumed at the selected finite section.*
6. *H, C, P, N are respectively the componentwise* hsame *transport row, displayed* Cont *replay row,* Pkg *provenance row, and local* $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$ *row.*

The classifier reads only $I, A, S, D, R, H, C, P, N$. It has no coordinate for countable choice, dependent choice, selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteChoiceWindowUp.FiniteChoiceWindowTasteGate single carrier alignment`

Theorem 3901.2 (Finite choice window NameCert surface)**.** *For every accepted* $\mathsf{FiniteChoiceWindow}^{\uparrow}$ *carrier $W = (I, A, S, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$ *surface consists exactly of five obligation families:*

1. *carrier admission for the unary index spine I and the candidate-row ledger A;*
2. *finite-section exactness, saying that S selects one displayed candidate for each displayed position and no undisplayed position;*

3. *tolerance compatibility with the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *and the* $\mathsf{StreamName}^{\uparrow}$ *finite window that consumes the section;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and readback routing through* R *after the selected finite section is fixed;*
5. *structural support through* H, C, P, N*, together with refusal of countable choice, dependent choice, selected limits, quotient stream equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

These obligations are precisely the finite window surface needed by the L10 route through $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*.*

Proof. Induct on the displayed unary index spine I. In the empty case, the candidate ledger and selected section are empty, and the carrier is admitted by the displayed transport, replay, provenance, and local naming rows.

For the successor step, assume the obligation surface has been built for the preceding finite spine. The next displayed position contributes one finite nonempty candidate list and one selected row. Adjoin that selected row to the finite section, transport it by H, replay the local route by C, and record its provenance in P and N.

The dyadic tolerance row D is stable under this finite adjoining step because the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger is read at the displayed request positions. The $\mathsf{StreamName}^{\uparrow}$ consumer sees the resulting finite section as a finite observation window, and the $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads R only after that window has been fixed.

No limiting step is used in the induction. Since every constructor step consumes one displayed list row and one displayed selected row, the proof never forms a countable-choice schedule, a dependent-choice chain, a selected limit, a quotient stream equality, or an ambient $\mathsf{Real}^{\uparrow}$ completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3901.1 and Definition 3901.2 seed the finite unary-indexed candidate carrier and its five NameCert obligation families for the L10 StreamName and RegSeqRat route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteChoiceWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, dependent choice, selected limits, quotient stream equality, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-section exactness, dyadic tolerance compatibility, StreamName window routing, RegSeqRat readback routing, and support-row refusal for the same finite packet.

Constructive story: A $\mathsf{FiniteChoiceWindow}^{\uparrow}$ packet is generated from a displayed unary index spine, finite candidate rows, selected finite-section rows, a DyadicRatCore tolerance ledger, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3902

A Concrete Naming Certificate for $\mathsf{CauchySequenceField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{CauchySequenceField}^{\uparrow}$ packet records the finite regular-Cauchy route by which field-facing real operations are read from rational names. It sits after the algebraic operation rows of $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, the order and apartness rows of **??**, the finite-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, and before the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a non-quotient field-facing route; it does not import host real arithmetic, quotient representatives, trichotomy, countable choice, or ambient completeness.

Definition 3902.1 (CauchySequenceField carrier)**.** *A* $\mathsf{CauchySequenceField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, W, D, A, O, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name whose field-facing operations are being inspected.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *synchronized-window row used to read the source sequence, companion operation inputs, and comparison requests at the same finite precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget ledger for radii, slack, denominator guards, and tolerance comparisons opened by the synchronized windows.*
4. *A is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *operation row, displaying addition, negation, multiplication, zero, one, and operation replay over the same regular-Cauchy source rows.*
5. *O is the* **??** *order and apartness row, displaying positivity, nonzero guards, reciprocal-side admission, and comparison data read through W and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, A, O have been displayed and replayed.*
7. *H records componentwise* hsame *transport for the regular-rational source, synchronized windows, dyadic budgets, algebra rows, order rows, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a field-operation request through $R, W, D, A, O, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceField}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, **??**, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, W, D, A, O, E, H, C, P, N$. It has no coordinate for a quotient real, selected sequence representative, host real field operation, trichotomy branch, countable-choice schedule, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchySequenceFieldUp.CauchySequenceFieldTasteGate single carrier alignment`

Theorem 3902.2 (CauchySequenceField NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceField}^{\uparrow}$ *carrier $F = (R, W, D, A, O, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular-rational source row R, synchronized-window exposure through W, dyadic budget exactness through D, algebraic operation exposure through A, order and apartness exposure through O, real-seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No field-facing obligation may replace these rows by host real arithmetic or quotient equality.*

Proof. Project the carrier of Definition 3902.1. The visible rows are precisely the regular-rational source, synchronized windows, dyadic budget ledger, algebra row, order row, real seal, transport, replay, provenance, and local naming.

The algebra row A supplies the operation side only after the source row R has been read through W and D. The order row O supplies positivity, apartness, and reciprocal-side guards through the same finite windows and budgets.

The seal row E is terminal. It is reached only after the regular-Cauchy source, window, budget, algebra, and order rows have been displayed, so it cannot be used as a detached host real.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient representative, host operation, trichotomy, countable-choice, or completeness coordinate, those principles are not local NameCert obligations. □

Theorem 3902.3 (CauchySequenceField operation route)**.** *Every field-facing read exported by an accepted* $\mathsf{CauchySequenceField}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchySequenceAlgebra}^{\uparrow} \longmapsto ?? \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes algebraic operations and order-side guards as regular-Cauchy rows before any terminal real-name seal is inspected, and exports no host real arithmetic, quotient equality, selected representative, trichotomy branch, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a field-operation request for the accepted carrier $F = (R, W, D, A, O, E, H, C, P, N)$. Carrier projection exposes the $\mathsf{RegSeqRat}^{\uparrow}$ source row R and the synchronized $\mathsf{StreamName}^{\uparrow}$ windows W before any operation row can be consumed.

The dyadic ledger D then fixes the finite budgets used by the request. Algebraic operation reads pass through A, while positivity, apartness, nonzero, and reciprocal-side reads pass through O over the same windows and budgets.

Only after those rows have been displayed does the request reach the terminal $\mathsf{Real}^{\uparrow}$ seal E. The replay row C and transport row H preserve this order componentwise, and P, N source and name the same route.

Because the packet has no alternate coordinate for host real arithmetic, quotient equality, selected representatives, trichotomy, countable choice, or ambient completeness, every exported field-facing read is forced through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3902.1, Definition 3902.2, and Definition 3902.3 seed the finite regular-Cauchy field route from RegSeqRat, StreamName, DyadicRatCore, CauchySequenceAlgebra, and CauchySequenceOrder rows into the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host real arithmetic, quotient equality, selected sequence representatives, trichotomy, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic-budget exactness, algebra and order handoff, real-seal non-escape, transport, and replay proofs for BEDC.Derived.CauchySequenceFieldUp.

Constructive story: A $\mathsf{CauchySequenceField}^{\uparrow}$ packet is generated from a regular rational source row, synchronized StreamName windows, a DyadicRatCore tolerance ledger, CauchySequenceAlgebra operation rows, CauchySequenceOrder order and apartness rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3903

A Concrete Naming Certificate for BishopLocatedCompletionReflector$^{\uparrow}$

Following Bishop and Bridges on constructive completion and located real analysis, the BishopLocatedCompletionRef packet records the reflector row that binds the located-real completion route of chapter 3488 to the public completion-reflector surface of chapter 1585. It consumes the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the located-real row of chapter 1933, and the terminal real seal of chapter 38. The packet is a human-origin reflector for displayed located completion data; it is not quotient completion equality, a selected limit, countable choice, trichotomy, or ambient completeness.

Definition 3903.1 (Bishop located completion reflector carrier)**.** *A* BishopLocatedCompletionReflector$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, R, D, S, Q, L, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BishopLocatedRealCompletion$^{\uparrow}$ *row exposing the located-real completion carrier through dyadic tolerances, finite stream windows, regular rational readback, locatedness, and a terminal real seal.*
2. *R is the* CauchyCompletionReflector$^{\uparrow}$ *row exposing unit, counit, idempotence, extension, reflection, and local real-name transport for the same completion-facing read.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used by both the located completion route and the reflector's local real-name transport.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected at those displayed tolerances.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *regular rational readback row obtained from the same finite windows.*
6. *L is the* LocatedReal$^{\uparrow}$ *locatedness row attached to Q.*
7. *E is the* Real$^{\uparrow}$ *seal reached only after B, R, D, S, Q, L have been displayed.*
8. *H records componentwise* hsame *transport for the located completion row, reflector row, dyadic tolerance, stream-window, regular readback, locatedness, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion-facing request through $B, R, D, S, Q, L, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$, $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$ *row binding the displayed completion-reflector packet.*

The classifier on accepted packets compares only these rows under the displayed transport and replay data. It has no coordinate for quotient streams, selected representatives, host real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionReflectorUp.BishopLocatedCompletionReflectorTasteGate`
`single carrier alignment`

Theorem 3903.2 (Bishop located completion reflector NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ *carrier* $K = (B, R, D, S, Q, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$ *surface consists exactly of carrier admission, Bishop-located completion exposure, CauchyCompletionReflector exposure, DyadicRatCore tolerance exposure, StreamName finite-window exposure, RegSeqRat readback, LocatedReal locatedness exposure, Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and local naming. Every reflector-facing consumer reads only the displayed route*

$$B, R \longmapsto D \longmapsto S \longmapsto Q \longmapsto L \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto N.$$

Proof. Project the carrier of Definition 3903.1. The only analytic rows are the located completion row, the public reflector row, the dyadic tolerance ledger, finite stream windows, regular rational readback, locatedness row, and Real seal.

Read the route in order. The row B supplies the located completion side and the row R supplies the reflector side. Their common finite route is then displayed through D, S, Q, L, E, so the reflector consumer cannot choose a second tolerance ledger, a host stream, or a selected real representative.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since quotient streams, selected limits, host equality, countable choice, trichotomy, compactness, and ambient completeness have no coordinate in the carrier, they cannot become local NameCert obligations. □

Theorem 3903.3 (Bishop located completion reflector route)**.** *Every* $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ *consumer that treats displayed located Cauchy data as a reflector object must pass through the sibling route of chapter 3488 and the public reflector route of chapter 1585. The accepted packet exposes the reflector only as the finite carrier* $B, R, D, S, Q, L, E, H, C, P, N$ *and exports no quotient completion equality, selected host limit, countable-choice selector, trichotomy split, detached* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or host real equality test.*

Proof. Start with the NameCert obligations. They force any reflector-facing read to expose both the Bishop-located completion row B and the CauchyCompletionReflector row R before the terminal real seal is readable.

The shared analytic route is D, S, Q, L, E. The dyadic tolerance row fixes the finite request, the StreamName row opens the corresponding windows, the RegSeqRat row repacks those windows as regular rational data, the LocatedReal row supplies the lower–upper separation evidence, and the Real row seals the route.

The sibling chapters supply adjacent but not interchangeable packets. $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ gives the located completion side, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ gives the public reflector side. The present carrier binds both with the common dyadic, stream, regular-rational, located-real, and real-seal rows, so no omitted sibling or host-level object can replace it. □

Falsifiable prediction: A claimed $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ read that reaches the $\mathsf{Real}^{\uparrow}$ seal without first displaying both the $\mathsf{BishopLocatedRealCompletion}^{\uparrow}$ row and the $\mathsf{CauchyCompletionReflector}^{\uparrow}$ row over the shared dyadic, stream, regular-rational, and located-real route refutes the carrier surface.

Independence witness: bishop-located-real-completion, cauchycompletionreflector, dyadicratcore, streamname, regseqrat, locatedreal, real: no listed sibling is bijective to $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ because this packet binds the located completion row and public reflector row to one shared finite tolerance, window, readback, locatedness, real-seal, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionReflector}^{\uparrow}$ packet is generated from a BishopLocatedRealCompletion row, a CauchyCompletionReflector row, DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, LocatedReal locatedness, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3903.1, Definition 3903.2, and Definition 3903.3 seed the finite reflector route binding located completion and public completion reflection over the shared dyadic, stream, regular-rational, located-real, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, selected limits, countable choice, trichotomy, host real equality, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-completion exposure, reflector exposure, tolerance-window exactness, regular readback, locatedness handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompletionReflectorUp.

Chapter 3904

A Concrete Naming Certificate for $\mathsf{CellularOrbitSynchronization}^{\uparrow}$

The $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ packet realizes the cellular carrier alternative of chapter 9391 as a finite paired-orbit synchronization surface. It consumes the observation-boundary route of chapter 2371, the classifier-boundary route of chapter 3810, the paired audit coupling of chapter 847, and the twin-substrate bridge of chapter 850. The packet records two local orbit replays together with the classifier tag row that authorizes them as one closed substrate read.

Definition 3904.1 (Cellular orbit synchronization carrier)**.** *A* $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (O_0, O_1, G, B, Q, A, T, H, C, P, N)$$

with the following displayed rows:

1. *O_0 and O_1 are paired local orbit replay rows, each giving a finite transition table, initial row, time window, spatial window, and replay ledger.*
2. *G is the classifier tag row that assigns both replayed windows to the same substrate observation family.*
3. *B is the* $\mathsf{CellularSubstrateObservationBoundary}^{\uparrow}$ *boundary row recording the manifest and accepted observation windows used by the paired replay.*
4. *Q is the* $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$ *row requiring the classifier tags to be displayed before either orbit read is exported.*
5. *A is the* $\mathsf{TwinSubstrateAuditCoupling}^{\uparrow}$ *row that records paired audit use without collapsing the two replay rows into a shared host orbit.*
6. *T is the* $\mathsf{TwinSubstrateBridge}^{\uparrow}$ *row through which the paired read may be inspected by a bridge-facing consumer.*
7. *H records componentwise* hsame *transport for the two orbit replays, classifier tag, observation boundary, classifier boundary, audit coupling, twin-substrate bridge, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the paired local orbits through G, B, Q, A, T, H, P, N.*

9. *P is the* Pkg *provenance over* CellularOrbitSynchronization$^{\uparrow}$, CellularSubstrateObservationBoundary$^{\uparrow}$, CellularSubstrateClassifierBoundary$^{\uparrow}$, TwinSubstrateAuditCoupling$^{\uparrow}$, TwinSubstrateBridge$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *N is the local* NameCert$_{\text{CellularOrbitSynchronization}^{\uparrow}}$ *row.*

The carrier has no coordinate for an unclassified orbit read, a detached host evaluator, an all-time induction principle, quotient stream equality, or an identification of the two substrate roles.

Lean4 ✓ `BEDC.Derived.CellularOrbitSynchronizationUp.CellularOrbitSynchronizationTasteGate`
`single carrier alignment`

Theorem 3904.2 (CellularOrbitSynchronization NameCert obligations)**.** *For every accepted* CellularOrbitSynchron *carrier $S = (O_0, O_1, G, B, Q, A, T, H, C, P, N)$, the local* NameCert$_{\text{CellularOrbitSynchronization}^{\uparrow}}$ *surface consists exactly of carrier admission for the paired orbit replays, classifier tag row, observation boundary, classifier boundary, audit coupling, twin-substrate bridge, componentwise transport, displayed continuation replay, package provenance, and local naming. Every synchronized substrate read factors through*

$$O_0, O_1 \longmapsto G \longmapsto B, Q \longmapsto A, T \longmapsto H, C, P, N.$$

No synchronized orbit obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 3904.1. The accepted packet displays exactly the paired local orbit rows, classifier tag row, observation boundary, classifier boundary, audit coupling, bridge row, and structural rows.

The classifier row is read before the boundary rows. Thus a pair of replay ledgers O_0, O_1 becomes a synchronized substrate read only when G names the inspected pattern family and the rows B, Q bind that tag to the observation and classifier boundaries.

The audit and bridge rows preserve the paired reading without identifying the two replays. The row A supplies paired audit coupling, T supplies bridge-facing inspection, and H, C, P, N transport, replay, source, and name only that finite packet. Therefore unclassified orbit reads, host evaluators, all-time induction, quotient streams, and substrate-role identifications cannot become NameCert obligations. □

Theorem 3904.3 (CellularOrbitSynchronization classifier-locked handoff)**.** *Every synchronized orbit read accepted by a* CellularOrbitSynchronization$^{\uparrow}$ *packet displays the classifier tag row G before either paired orbit replay is exported through the observation boundary, classifier boundary, audit coupling, or twin-substrate bridge. A Lean follow-up for this carrier must supply* Nontrivial *and* FieldFaithful *instances in* `lean4/BEDC/Derived/CellularOrbitSynchronizationUp/TasteGate.lean`.

Proof. Project the accepted carrier. The paired orbit rows O_0, O_1 are present, but they are not exported directly; they are routed through the classifier tag row G.

Use the observation and classifier boundaries. The observation-boundary row B fixes the manifest and local windows, while the classifier-boundary row Q requires the pattern tag to be displayed for each observed window. Hence an orbit replay without G has no accepted route to a synchronized read.

The audit coupling and bridge rows are downstream. They consume the already tagged boundary packet, and the structural rows H, C, P, N preserve the same finite route. Thus the classifier-locked handoff is the only exported synchronization surface. □

Falsifiable prediction: Any synchronized orbit read that displays the paired local orbit replays but omits the classifier tag row G is rejected by the CellularOrbitSynchronization$^{\uparrow}$ carrier.

Independence witness: `cellular_substrate_observation_boundary`, `cellular_substrate_classifier_bo`
`twinsubstrateauditcoupling`, `twinsubstratebridge`: none is bijective to $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ because the synchronization packet uniquely binds paired local orbit replays, classifier-locked observation rows, paired audit coupling, bridge-facing inspection, transport, replay, provenance, and local naming in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularOrbitSynchronization}^{\uparrow}$ packet is generated from paired local orbit replay rows, a classifier tag row, CellularSubstrateObservationBoundary and CellularSubstrateClassifierBoundary rows, TwinSubstrateAuditCoupling and TwinSubstrateBridge rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3904.1, Definition 3904.2, and Definition 3904.3 seed the finite classifier-locked synchronization surface for paired local cellular orbit reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularOrbitSynchronization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, all-time cellular induction, host evaluator correctness, quotient stream equality, or identification of the two substrate roles.

Upgrade path: Obligation closure requires checked carrier admission, paired orbit replay exactness, classifier-tag exposure, observation-boundary and classifier-boundary handoff, audit coupling, bridge-facing inspection, transport, replay, provenance, and local naming for BEDC.Derived.CellularOrbitSynchronizationUp.

Chapter 3905

A Concrete Naming Certificate for FiniteDimensionalHahnBanachSeparation$^{\uparrow}$

Following Bishop–Bridges constructive analysis and finite-dimensional separation theory, the FiniteDimensionalHahn packet records the finite certificate surface for a separating affine functional between a located point and a displayed convex set in a finite-dimensional normed vector space. It consumes the vector-space row of chapter 47, the linear-map row of chapter 48, the finite norm row used by chapter 1644, the real-name row of chapter 38, the located comparison row of chapter 1730, and the convex-set row of chapter 208. The packet is deliberately finite-dimensional: it names coefficient rows and located comparisons only for the displayed finite basis window, and it has no coordinate for full Hahn–Banach extension, choice-selected duals, or ambient weak-star compactness.

Definition 3905.1 (Finite-dimensional Hahn–Banach separation carrier)**.** *A* FiniteDimensionalHahnBanachSeparati *carrier is a finite* BHist *packet*

$$S = (V, B, L, N, R, Q, C, E, H, T, P, M)$$

with the following displayed rows:

1. *V is the finite* VecSpace$^{\uparrow}$ *source row, including the finite coordinate window on which every vector read is performed;*
2. *B is the finite basis and coefficient-admission ledger, recording that each inspected vector is read by finitely many scalar coordinates;*
3. *L is the* LinMap$^{\uparrow}$ *row for the separating functional, displayed by its finite coefficient list on B;*
4. *N is the* Norm$^{\uparrow}$ *row used only to bound the coefficient list and the finite vector reads;*
5. *R is the* Real$^{\uparrow}$ *endpoint ledger for scalar evaluations and thresholds;*
6. *Q is the* LocatedRealComparison$^{\uparrow}$ *row deciding the displayed scalar inequality with located slack;*
7. *C is the* ConvexSet$^{\uparrow}$ *row for the accepted convex-set membership and convex-combination reads;*
8. *E is the exact separation row, containing the point-side and set-side evaluation endpoints and their located comparison;*
9. *H records componentwise* hsame *transport for $V, B, L, N, R, Q, C, E, P, M$;*
10. *T records displayed* Cont *replay through the finite coordinate, functional, comparison, convex-set, and exactness rows;*

11. *P is the* Pkg *provenance over* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$, VecSpace$^{\uparrow}$, LinMap$^{\uparrow}$, Norm$^{\uparrow}$, Real$^{\uparrow}$, LocatedRealComparison$^{\uparrow}$, ConvexSet$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *M is the local* NameCert$_{\text{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed finite rows. It has no coordinate for an arbitrary extension of a functional, a selected quotient dual, a Zorn-style maximal sublinear extension, weak-star compactness, or an infinite family of separating hyperplanes.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalHahnBanachSeparationUp.FiniteDimensionalHahnBanachSeparationTaste single carrier alignment`

Theorem 3905.2 (Finite coefficient admission)**.** *For every accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier $S = (V, B, L, N, R, Q, C, E, H, T, P, M)$, every read of the separating functional factors through the finite coefficient ledger B, the* LinMap$^{\uparrow}$ *row L, the norm row N, and the scalar endpoint ledger R. A consumer may inspect only the finitely displayed coefficient list on the finite* VecSpace$^{\uparrow}$ *window; it may not replace that list by a host dual-space element or by a choice-selected functional.*

Proof. Project the accepted carrier. The vector data are read through V and the coefficient window B, while the functional row is L and the scalar endpoints are R.

The norm row N bounds only those displayed coefficient and vector reads. Transport and replay through H, T, P, M preserve the same finite list, so no additional coordinate can introduce a host dual element.

Thus coefficient admission is finite-row admission. The accepted packet exports the displayed list and its NameCert routes, not an unrestricted dual object or a choice-selected functional. □

Theorem 3905.3 (Separating inequality exactness)**.** *For every accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier $S = (V, B, L, N, R, Q, C, E, H, T, P, M)$, the separating inequality is exactly the row E read through the* Real$^{\uparrow}$ *endpoints in R, the* LocatedRealComparison$^{\uparrow}$ *comparison row Q, and the* ConvexSet$^{\uparrow}$ *membership row C. Consequently any downstream convex-analysis consumer receives a located scalar comparison between the point-side evaluation and the set-side bound, not an ambient separation theorem.*

Proof. Read the exactness row E. Its scalar endpoints are supplied by R, and the comparison between them is supplied by Q.

The convex side is constrained by C: membership and convex-combination reads of the displayed set pass through the finite convex-set row before the set-side bound can be cited. The functional values used by both sides still pass through B, L, N from Definition 3905.2.

Therefore the public inequality is the located comparison stored in E over the displayed finite rows. No consumer can cite a stronger ambient separating-hyperplane principle from this packet. □

Theorem 3905.4 (No escape to full Hahn–Banach choice)**.** *An accepted* FiniteDimensionalHahnBanachSeparation$^{\uparrow}$ *carrier exports no full Hahn–Banach extension principle, no Zorn-style maximal extension, no selected quotient dual, no weak-star compactness object, and no infinite separating family. Every exported read factors through the finite vector-space window, coefficient ledger, linear-map row, norm row, real endpoint ledger, located comparison row, convex-set row, exactness row, componentwise transport, displayed replay, provenance, and local NameCert rows.*

Proof. Inspect the carrier coordinates. They are exactly $V, B, L, N, R, Q, C, E, H, T, P, M$, so every accepted read is a projection of one of those rows or a replay through the structural rows.

The rows B, L, N, R, Q, C, E are finite-dimensional and located: coefficients are listed on the finite basis window, scalar endpoints are displayed as $\mathsf{Real}^{\uparrow}$ rows, and the separation statement is a $\mathsf{LocatedRealComparison}^{\uparrow}$ row.

The excluded principles require coordinates not present in the carrier. Since there is no row for arbitrary extension, maximality, quotient dual selection, weak-star compactness, or an infinite family, those principles cannot be exported by the local NameCert surface. □

Falsifiable prediction: Dropping the coefficient ledger B makes Definition 3905.2 false: a consumer could no longer distinguish a displayed finite functional from an ambient host dual element.

Independence witness: vecspace, linearmap, norm, real, locatedrealcomparison, convexset: none is bijective with $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ because this carrier alone binds the finite vector-space window, coefficient ledger, linear functional, norm bound, scalar endpoints, located comparison, convex-set membership, and exact separation row into one NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}$ packet is generated from a finite VecSpaceUp coordinate window, a finite coefficient ledger, a LinearMapUp separating functional row, a NormUp bound row, RealUp scalar endpoints, a LocatedRealComparisonUp inequality row, a ConvexSetUp membership row, separating exactness data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3905.1, Definition 3905.2, Definition 3905.3, and Definition 3905.4 seed the finite-dimensional separation packet over VecSpaceUp, LinearMapUp, NormUp, RealUp, LocatedRealComparisonUp, ConvexSetUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalHahnBanachSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Hahn–Banach extension, Zorn maximality, selected quotient duals, weak-star compactness, infinite separating families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coefficient transport, separating inequality exactness, convex-set readback, classifier stability, and non-escape for BEDC.Derived.FiniteDimensionalHahnBanachSeparationUp.

Chapter 3906

A Concrete Naming Certificate for PiecewiseLinearApproximation$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the PiecewiseLinearApproximation$^{\uparrow}$ packet records polygonal approximation over a closed bounded interval as finite NameCert rows. It starts from the ClosedBoundedInterval$^{\uparrow}$ source of chapter 1591, reads a UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$ mesh through chapter 2033 and chapter 1863, compares its cell ledger with the DyadicStepApproximation$^{\uparrow}$ route of chapter 3774, and hands terminal endpoint and value reads to ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ consumers through chapter 130, chapter 38, and chapter 291. The certificate names the finite affine-cell surface only; it does not introduce a host function space, selected uniform limit, quotient equality of functions, open-cover compactness, or ambient real completeness.

Definition 3906.1 (PiecewiseLinearApproximation carrier)**.** *A* PiecewiseLinearApproximation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, M, E, A, D, F, T, H, C, P, N)$$

with the following displayed rows:

1. *I is the* ClosedBoundedInterval$^{\uparrow}$ *source row, displaying the two endpoint* Real$^{\uparrow}$ *names and the bounded interval window being approximated.*
2. *M is a* UniformPartition$^{\uparrow}$ *or* DyadicPartition$^{\uparrow}$ *mesh row, including the ordered cell boundary ledger, the positive mesh-width row, and the endpoint-coverage route.*
3. *E is the finite endpoint-value ledger, recording one* Real$^{\uparrow}$ *value row at every displayed mesh boundary.*
4. *A is the affine cell ledger: for each adjacent mesh cell it records the two endpoint-value rows, the affine interpolation route between them, and the cell-local evaluation window.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *slope and error ledger, recording dyadic slope brackets, cell-width witnesses, tolerance requests, and the displayed error budget used by downstream approximation consumers.*
6. *F is the* ContinuousMap$^{\uparrow}$ *handoff row, exposing the polygonal route only after I, M, E, A, D have been displayed.*
7. *T is the comparison row with* DyadicStepApproximation$^{\uparrow}$*, reading the same mesh cells and tolerance requests as a step-surface neighbour without identifying the affine ledger with a step ledger.*

8. *H records componentwise* hsame *transport for the interval source, mesh, endpoint values, affine cell rows, dyadic slope and error ledger, continuous-map handoff, step-neighbour row, provenance, and local naming.*

9. *C records displayed* Cont *replay of the route*

$$I, M, E, A, D \longmapsto F, T \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* PiecewiseLinearApproximation$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, UniformPartition$^{\uparrow}$, DyadicPartition$^{\uparrow}$, DyadicStepApproximation$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{PiecewiseLinearApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host function graph, quotient equality of maps, selected limiting function, compactness theorem, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PiecewiseLinearApproximationUp.PiecewiseLinearApproximationTasteGate`
`single carrier alignment`

Theorem 3906.2 (PiecewiseLinearApproximation NameCert obligations)**.** *For every accepted* PiecewiseLinearApproximation$^{\uparrow}$ *carrier*

$$L = (I, M, E, A, D, F, T, H, C, P, N),$$

the local NameCert$_{\text{PiecewiseLinearApproximation}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, mesh coverage through M, endpoint-value display through E, affine cell exactness through A, dyadic slope and error accounting through D, continuous-map handoff through F, step-surface comparison through T, componentwise transport through H, replay through C, provenance through P, and local naming through N. No obligation may read a host function space, selected uniform limit, quotient map equality, open-cover compactness, countable choice, or ambient real completeness.*

Proof. Carrier admission is projection from Definition 3906.1. The accepted packet displays the closed bounded interval, mesh, endpoint values, affine cell rows, dyadic slope and error ledger, and structural rows before any continuous-map or step-neighbour consumer sees the approximation.

The mesh rows are finite. The UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$ row supplies the ordered cells, endpoint coverage, and positive width witnesses, while the ClosedBoundedInterval$^{\uparrow}$ row fixes the source window. Thus every affine cell row is attached to a displayed source cell rather than to an ambient interval object.

The affine rows and dyadic ledger exhaust the approximation data. Each cell read factors through its two endpoint Real$^{\uparrow}$ values and the DyadicRatCore$^{\uparrow}$ slope and error ledger before the ContinuousMap$^{\uparrow}$ handoff row is available.

Finally H, C, P, N transport, replay, source, and name only this finite packet. Since no further carrier coordinate remains, the excluded host-function, quotient, limit, compactness, choice, and completeness objects cannot become local NameCert obligations. □

Theorem 3906.3 (PiecewiseLinearApproximation uniform handoff)**.** *Every* ContinuousMap$^{\uparrow}$, DyadicStepApproxim
Real$^{\uparrow}$, *or* DyadicRatCore$^{\uparrow}$ *consumer of an accepted* PiecewiseLinearApproximation$^{\uparrow}$ *packet receives only the finite route*

ClosedBoundedInterval$^{\uparrow}$, UniformPartition$^{\uparrow}$ or DyadicPartition$^{\uparrow}$, endpoint Real$^{\uparrow}$ rows, affine cells, DyadicRatCore$^{\uparrow}$ le

The consumer may inspect the polygonal approximation as a finite uniform or dyadic mesh surface, but it cannot bypass that route to obtain a host continuous function, selected uniform limit, quotient function equality, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The terminal consumers see F or T only after I, M, E, A, D have been displayed.

For the continuous-map handoff, the row F reads the same endpoint values, affine cell routes, and dyadic error ledger. It therefore exports a polygonal route over the visible mesh, not a host function graph.

For the step-surface neighbour, the row T shares the mesh and tolerance requests with $\mathsf{DyadicStepApproximation}^{\uparrow}$ while preserving the affine cell ledger as a separate displayed row. Hence the handoff can compare approximation ledgers without collapsing them into quotient equality of functions.

The remaining rows H, C, P, N only transport and replay the same finite data. No hidden limit, compactness, choice, completeness, or host-map coordinate is present for a consumer to read. □

Closure status for $\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{PiecewiseLinearApproximation}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source row, a UniformPartition or DyadicPartition mesh row, endpoint RealUp value rows, affine cell rows, a DyadicRatCore slope and error ledger, a ContinuousMap handoff row, a DyadicStepApproximation neighbour row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3906.1, Definition 3906.2, and Definition 3906.3 seed the finite polygonal approximation surface over closed bounded interval, mesh, endpoint-value, affine-cell, dyadic-ledger, and continuous-map handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PiecewiseLinearApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected uniform limits, quotient function equality, open-cover compactness, countable choice, host function spaces, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, mesh coverage, endpoint-value display, affine cell exactness, dyadic slope and error accounting, continuous-map handoff, step-neighbour comparison, transport, replay, provenance, and local naming for BEDC.Derived.PiecewiseLinearApproximationUp.

Chapter 3907

A Concrete Naming Certificate for $\mathsf{CompactIntervalMaximum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalMaximum}^{\uparrow}$ packet records the finite witness surface for a continuous real-valued map on a closed bounded interval to expose a maximum candidate. It consumes the endpoint and containment route of chapter 1591, the finite interval-net surface of $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, the graph and continuity rows of chapter 130, and the $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback route. The packet names only finite BEDC evidence for a maximum witness; it does not import host extrema, open-cover compactness, selected subcovers, Real completeness, quotient equality, Lean verification, or bridge checking.

Definition 3907.1 (Compact interval maximum carrier)**.** *A* $\mathsf{CompactIntervalMaximum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, F, G, N, S, R, E, W, H, C, P, L)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *sealing.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the displayed interval source, including finite graph reads and pointwise continuity data.*
3. *G is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite-cell row, listing the interval cells, centers, radius ledgers, and coverage reads used by the witness search.*
4. *N is the finite value-comparison row selecting a displayed cell and center whose readback is no smaller than the other displayed net values at the admitted precision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for endpoint, graph, and value observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected value and comparison witnesses.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the selected maximum-witness value after I, F, G, N, S, R have been displayed.*
8. *W is the witness-locality row saying that the selected center is read through a cell of G whose containment is certified by I.*
9. *H records componentwise* hsame *transport for I, F, G, N, S, R, E, W and the local packet shape.*

10. *C records displayed* Cont *replay from interval source and graph reads through finite cells, value comparison, witness locality, readback, Real sealing, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalMaximum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$ *row.*

The carrier exports no host maximum operator, open-cover compactness theorem, selected subcover, ambient Real completeness principle, quotient equality of real names, or global choice row.

Theorem 3907.2 (Compact interval maximum finite witness obligations)**.** *For the carrier of Definition 3907.1, every accepted maximum-witness read factors through the closed bounded interval row I, the continuous-map graph row F, the finite-net cell row G, the finite value-comparison row N, the* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback route S, R, E, the witness-locality row W, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming L. These rows are the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$ *obligation surface.*

Lean4 ✓ `BEDC.Derived.CompactIntervalMaximumUp.CompactIntervalMaximumTasteGate single carrier alignment`

Proof. Project the carrier. The interval source, graph, finite-cell, comparison, schedule, readback, seal, witness-locality, transport, replay, provenance, and local naming rows are the only coordinates of the accepted packet.

The closed bounded interval row I supplies endpoint order and containment ledgers before any cell is inspected. The finite-net row G supplies the displayed cells and centers, while F supplies the graph reads whose values are compared by N.

The readback route is downstream of those finite rows. The schedule S, regular rational readback R, and Real seal E expose the selected witness value only through the displayed observations, and W keeps the selected center inside a certified cell of the interval net.

Finally H, C, P, L transport, replay, package, and name the same finite surface. Since the packet has no coordinate for host extrema, open covers, selected subcovers, ambient completeness, quotient equality, or global choice, every accepted maximum-witness read remains on the displayed NameCert rows. □

Falsifiable prediction: A compact-interval maximum consumer that omits the finite cell row, graph row, or value-comparison row cannot claim a $\mathsf{CompactIntervalMaximum}^{\uparrow}$ witness.

Independence witness: closedboundedinterval, continuousmap, heineborel-finite-net, regseqrat, real: $\mathsf{CompactIntervalMaximum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval containment, graph reads, finite cells, value comparison, selected witness locality, readback, sealing, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{CompactIntervalMaximum}^{\uparrow}$ is expected at `BEDC.Derived.CompactIntervalMaximu` It should preserve the finite maximum-witness surface as carrier data before any downstream extreme-value consumer can inspect it.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalMaximum}^{\uparrow}$ packet is generated from a closed bounded interval row, a continuous-map graph row, a finite interval-net row, a finite value-comparison witness, StreamName schedules, RegSeqRat readbacks, a Real seal, witness locality, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalMaximum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3907.1 and Definition 3907.2 seed the finite maximum-witness carrier for continuous maps on closed bounded intervals.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactIntervalMaximum}\uparrow}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close host extrema, open-cover compactness, selected subcovers, Real completeness, quotient equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell coverage, graph readback, value comparison, witness locality, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalMaximumUp.

Chapter 3908

A Concrete Naming Certificate for $\mathsf{FilterCompletionMonad}^{\uparrow}$

Following Kelley and Bourbaki on filters and completion, the $\mathsf{FilterCompletionMonad}^{\uparrow}$ packet records the finite monad-shaped surface by which a displayed Cauchy-filter completion route admits a unit, a flattening read, and a terminal Real handoff without quotienting filters or selecting limits. It sits after the Cauchy-filter completion boundary of chapter 1569, the finite basis route of chapter 3862, the Cauchy-filter cluster route of chapter 1734, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet exposes the completion monad only as finite BHist and $\mathsf{NameCert}$ rows.

Definition 3908.1 (Cauchy completion filter monad carrier)**.** *A* $\mathsf{FilterCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, B, Z, S, R, D, E, U, F, A, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-boundary row whose accepted reads remain tied to displayed finite Cauchy-filter evidence.*
2. *B is the* $\mathsf{FilterLimitBasis}^{\uparrow}$ *basis row used by the completion request before any Real-facing seal is inspected.*
3. *Z is the* $\mathsf{CauchyFilterCluster}^{\uparrow}$ *cluster-handoff row, exposing the same finite support as a windowed cluster route.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule read by the basis and cluster rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite windows and readback rows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only downstream from Q, B, Z, S, R, D.*
8. *U is the unit row, admitting a displayed source filter as a one-layer completion packet without adding a quotient filter or chosen limit.*
9. *F is the multiplication row, flattening a displayed two-layer completion packet by replaying its inner and outer Q, B, Z, S, R, D, E routes.*

10. *A is the associativity replay row, recording that the two displayed flattening orders read the same finite rows before E.*

11. *H records componentwise* hsame *transport for the completion, basis, cluster, window, readback, tolerance, seal, unit, flattening, associativity, replay, provenance, and local naming rows.*

12. *C records displayed* Cont *replay for unit reads, flattening reads, associativity reads, and Real handoff reads through $Q, B, Z, S, R, D, E, U, F, A, H, P, N$.*

13. *P is the* Pkg *provenance over* $\mathsf{FilterCompletionMonad}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, B, Z, S, R, D, E, U, F, A, H, C, P, N$. There is no coordinate for quotient filters, selected limits, ultrafilters, countable-choice schedules, host filter equality, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FilterCompletionMonadUp.FilterCompletionMonadTasteGate single carrier alignment`

Theorem 3908.2 (Cauchy completion filter monad laws)**.** *For every accepted* $\mathsf{FilterCompletionMonad}^{\uparrow}$ *carrier $M = (Q, B, Z, S, R, D, E, U, F, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterLimitBasis}^{\uparrow}$, $\mathsf{CauchyFilterCluster}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *unit, multiplication, associativity, transport, replay, provenance, and local naming rows;*

2. *unit admission through U, requiring every source-filter read to enter the completion route through the same Q, B, Z, S, R, D, E rows;*

3. *multiplication admission through F, requiring every two-layer completion read to flatten by replaying the inner and outer displayed completion routes before the terminal seal;*

4. *associativity of displayed replay through A, requiring the two flattening orders to expose the same $Q, B, Z, S, R, D, E, U, F, H, C, P, N$ rows before any Real-facing consumer reads E;*

5. *Real handoff through E, requiring completion, basis, cluster, window, readback, tolerance, transport, replay, provenance, and local naming to remain visible to every* $\mathsf{Real}^{\uparrow}$ *consumer.*

No listed obligation may be discharged by quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, detached completion points, or ambient completeness.

Proof. Project the carrier of Definition 3908.1. The visible rows are exactly $Q, B, Z, S, R, D, E, U, F, A, H, C, P, N$, so carrier admission has no hidden quotient-filter, selected-limit, choice, host-equality, or ambient-completeness coordinate.

The unit row is read as a displayed admission into the existing completion boundary. It points the source filter through the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row Q, the $\mathsf{FilterLimitBasis}^{\uparrow}$ row B, the $\mathsf{CauchyFilterCluster}^{\uparrow}$ row Z, and the finite observation rows S, R, D before E is available.

The multiplication row flattens only displayed completion data. An accepted two-layer packet first replays the inner and outer Q, B, Z, S, R, D, E routes, and then records the flattened route through F, H, C, P, N.

Associativity is the equality of the displayed replay surface, not an ambient quotient principle. The row A states that the two orders of flattening expose the same finite rows before the terminal seal, so no selected representative or ultrafilter argument can enter between the two replays.

Finally apply the Real handoff. Since E is downstream from the completion, basis, cluster, window, readback, tolerance, unit, flattening, associativity, transport, replay, provenance, and local naming rows, every $\mathsf{Real}^{\uparrow}$ consumer receives the displayed monad route and no external completion principle. □

Falsifiable prediction: An accepted $\mathsf{FilterCompletionMonad}^{\uparrow}$ read is refuted if unit, multiplication, associativity, or Real handoff reaches a terminal seal without displaying the CauchyFilterCompletion, FilterLimitBasis, CauchyFilterCluster, StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local naming rows.

Independence witness: cauchyfiltercompletion, filterlimitbasis, cauchyfiltercluster, streamname, regseqrat, dyadicratcore, real: $\mathsf{FilterCompletionMonad}^{\uparrow}$ is not bijective to any listed sibling because it records unit, flattening, associativity replay, and Real handoff as one finite NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{FilterCompletionMonad}^{\uparrow}$ packet is generated from a CauchyFilterCompletion row, FilterLimitBasis row, CauchyFilterCluster row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, unit admission, multiplication flattening, associativity replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3908.1 and Definition 3908.2 seed the finite filter-completion monad route over completion, basis, cluster, window, readback, tolerance, Real seal, unit, flattening, associativity, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FilterCompletionMonad}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, ultrafilters, countable-choice schedules, host equality, detached completion points, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit admission, multiplication flattening, associativity replay, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.FilterCompletionMonadUp.

Chapter 3909

A Concrete Naming Certificate for MetaCICDecidableNeutralSpine$^{\uparrow}$

The MetaCICDecidableNeutralSpine$^{\uparrow}$ packet realizes the neutral-spine branch of the MetaCIC decidability audit map as finite BEDC data. It reads the bounded normal-equality checker of chapter 752, the decidability boundary of chapter 749, the witness surface of chapter 751, the typing frontier of chapter 3616, and the fuel trace of chapter 3889. Its role is to expose a neutral-spine comparison route without promoting bounded computation to global conversion decidability.

Definition 3909.1 (MetaCIC decidable neutral-spine carrier)**.** *A* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier is a finite packet*

$$S = (T, E, Q, B, R, H, C, P, N)$$

with the following rows:

1. *T is the typing-checker frontier row, carrying the closed context, neutral head, spine arguments, and result type read by* MetaCICDecidableTypingFrontier$^{\uparrow}$*.*
2. *E is the same-term row for structurally identical neutral heads and argument lists.*
3. *Q is the bounded normal comparison row supplied by* BoundedNormalEqualityChecker$^{\uparrow}$*.*
4. *B is the boundary row stating which finished-normal-form evidence would be needed to pass from bounded comparison to public conversion decidability.*
5. *R is the refusal row blocking any consumer that asks for unbounded beta-conversion, subject reduction, or strong-normalization evidence not displayed in S.*
6. *H is componentwise* hsame *transport for the typing, same-term, bounded-comparison, boundary, refusal, replay, provenance, and local-name rows.*
7. *C is the displayed* Cont *replay from a neutral-spine decidability request through T, E, Q, B, R, H, P, N.*
8. *P is* Pkg *provenance over the MetaCIC typing checker,* BoundedNormalEqualityChecker$^{\uparrow}$*,* MetaCICDecidableBoun MetaCICDecidabilityWitness$^{\uparrow}$*,* MetaCICDecidableTypingFrontier$^{\uparrow}$*,* MetaCICFuelTrace$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* NameCert$_{\text{MetaCICDecidableNeutralSpine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for global conversion decidability, unrestricted normalization, subject reduction, confluence, or an external semantic equality oracle.

Theorem 3909.2 (Neutral-spine NameCert obligations)**.** *Every accepted* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier exposes the finite obligation surface consisting of typing-frontier exactness, same-term structural readback, bounded normal comparison, decidability-boundary refusal, componentwise transport, displayed replay, provenance, and local naming. A consumer of the packet receives no additional global conversion theorem.*

Proof. Project the carrier of Definition 3909.1. The typing row T fixes the closed neutral-spine boundary before any comparison row is read, and the same-term row E records only structural agreement of the displayed neutral head and argument list.

The bounded comparison row Q is then interpreted through the boundary row B and refusal row R. These rows expose exactly the finite checker result and the missing endpoint evidence; they do not add strong normalization, confluence, subject reduction, or unbounded conversion.

Finally, H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. Hence every accepted read of the packet is confined to the displayed NameCert obligation surface. □

Lean4 ✓ `BEDC.Derived.MetaCICDecidableNeutralSpineUp.MetaCICDecidableNeutralSpineTasteGate`
`single carrier alignment`

Theorem 3909.3 (Neutral-spine bounded-comparison boundary)**.** *For any accepted* MetaCICDecidableNeutralSpine *carrier, the passage from a bounded normal-equality answer to a public neutral-spine conversion answer is admissible only when the finished-normal-form boundary named by B is supplied inside the carrier route. Without that row, the packet exports a checker-local answer and a refusal, not global decidability.*

Proof. Read the bounded comparison row Q together with the boundary row B. The comparison row supplies a finite result about the inspected normal forms, while B records the exact endpoint evidence needed for public conversion.

If B is absent from the consumer route, the refusal row R is the only admissible export beyond the bounded checker answer. The transport and replay rows keep this refusal attached to the same typed neutral spine, so no external normalizer or semantic equality proof can enter through provenance. □

Theorem 3909.4 (Neutral-spine fuel monotonicity)**.** *Let $S = (T, E, Q, B, R, H, C, P, N)$ be an accepted* MetaCICDecidableNeutralSpine$^{\uparrow}$ *carrier whose bounded normal comparison row Q records a checker-local answer at fuel k. If a displayed extension of the same fuel trace reaches fuel k' while preserving the typing frontier, same-term row, endpoint-boundary row, refusal row, componentwise transport, replay, provenance, and local naming rows T, E, B, R, H, C, P, N, then the extended comparison consumer remains inside the same checker-local answer/refusal boundary. It does not export public conversion decidability, strong normalization, confluence, subject reduction, or a semantic equality oracle.*

Proof. Project the carrier and the displayed fuel trace from the bounded comparison row Q. The extension to k' is admissible only because it preserves the already displayed typing frontier and same-term rows T, E together with the same endpoint-boundary row B.

The refusal row R remains attached to the extended trace. Thus increasing the inspected fuel may refine the checker-local comparison row, but it still leaves any request for public conversion outside the displayed endpoint evidence.

Finally, replay the extended row through C and transport the unchanged coordinates componentwise through H. The provenance and local naming rows P, N identify the result as the

same finite neutral-spine packet, so no global conversion theorem or off-packet normalizer is made available. □

Falsifiable prediction: An accepted $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ consumer is refuted if it exports neutral-spine conversion while omitting the displayed typing-frontier row, same-term row, bounded normal-equality row, boundary row, refusal row, transport, replay, provenance, or local NameCert row.

Independence witness: boundednormalequalitychecker, metacicdecidableboundary, metacicdecidabilitywitness, metacic-decidable-typing-frontier: no listed sibling is bijective to $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ because this packet jointly carries the typed neutral-spine frontier, structural same-term row, bounded normal comparison, endpoint-boundary refusal, transport, replay, provenance, and local naming in one carrier. The bounded checker lacks the typed frontier and refusal packet; the boundary packet lacks the same-term spine; the witness packet records typing and conversion evidence without owning the neutral-spine route; the typing frontier lacks the bounded normal comparison boundary.

Lean follow-up must provide Nontrivial and FieldFaithful TasteGate instances for the neutral-spine carrier before any bridge can upgrade the verification axis.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ packet is generated from typing-checker frontier data, same-term neutral-spine readback, bounded normal comparison, endpoint-boundary refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3909.1, Definition 3909.2, and Definition 3909.3 seed the neutral-spine decidability carrier, obligation surface, and bounded-comparison boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global conversion decidability, strong normalization, subject reduction, confluence, semantic equality, Lean verification, or bridge checking for $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, typing-frontier exactness, same-term readback, bounded normal-comparison soundness, endpoint-boundary refusal, replay stability, provenance non-escape, and TasteGate instances for BEDC.Derived.MetaCICDecidableNeutralSpineUp.

Chapter 3910

A Concrete Naming Certificate for $\mathsf{YoungIntegral}^{\uparrow}$

Following Young and Lyons, the $\mathsf{YoungIntegral}^{\uparrow}$ packet records controlled Riemann-sum integration from finite p-variation data. It sits beside the bounded-variation carrier of chapter 398, the regulated-function surface of chapter 1906, the Riemann-sum packet of chapter 1615, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The chapter names only the finite controlled-sum route; it does not import measure completion, quotient equality of functions, selected refinements, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3910.1 (Young integral carrier)**.** *A* $\mathsf{YoungIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, V, C, S, D, R, E, H, K, P, N)$$

with the following displayed rows:

1. *F is the regulated integrand row, read through* $\mathsf{RegulatedFunction}^{\uparrow}$ *and its finite left/right window data.*
2. *G is the regulated integrator row, carrying the same interval ledger and endpoint schedule as F.*
3. *V is the p-variation and control ledger, recording bounded-variation comparison rows, variation exponents, control steps, and the finite superadditivity checks used by the route.*
4. *C is the controlled-window row, matching integrand oscillation windows with integrator variation windows before any sum is formed.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row listing the tagged partitions, dyadic mesh requests, controlled step values, and finite sum values.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for mesh radii, variation bounds, and sum-difference margins.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Cauchy trace of controlled sums.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *integral seal reached only after F, G, V, C, S, D, R have been displayed.*
9. *H records componentwise* hsame *transport for the integrand, integrator, variation ledger, controlled windows, Riemann-sum row, dyadic ledger, readback, seal, provenance, and local name.*

10. *K records displayed* Cont *replay from a Young-integral request through controlled windows, finite sums, dyadic tolerances, regular readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegral}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a measure-completion object, quotient equality of functions, an unrecorded refinement selector, a host rough-path lift, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.YoungIntegralUp.YoungIntegralTasteGate single carrier alignment`

Theorem 3910.2 (Young integral NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegral}^{\uparrow}$ *carrier* $Y = (F, G, V, C, S, D, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$ *surface is exhausted by integrand admission, integrator admission, p-variation control, controlled-window matching, Riemann-sum formation, dyadic tolerance exactness, regular rational readback, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming. No accepted consumer may replace those rows by measure completion, quotient function equality, a selected refinement sequence, a host rough-path lift, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier of Definition 3910.1. The rows F and G expose the regulated integrand and integrator data, while V records the finite variation and control ledger used by the integration route.

The controlled windows are then finite. The row C matches integrand oscillation windows to integrator variation windows, and S forms only the tagged sums displayed by the $\mathsf{RiemannSum}^{\uparrow}$ packet.

The convergence handoff is read through D, R, E. Dyadic tolerances in D bound the mesh and sum-difference rows, R records the regular rational trace, and E seals the resulting Real value only after the previous rows have been consumed.

Finally H, K, P, N transport, replay, source, and name that same route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 3910.3 (Young integral Riemann-sum handoff)**.** *Every accepted* $\mathsf{YoungIntegral}^{\uparrow}$ *integral read factors through*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as recorded by F, G, V, C, S, D, R, E. The handoff exports a finite controlled Riemann-sum route and no measure-completion integral, quotient function equality, unrecorded refinement selector, host rough-path lift, countable-choice principle, or ambient Real-completeness theorem.

Proof. Start with the integrand and integrator rows. The regulated-function siblings expose the finite endpoint and window data for F and G, and the bounded-variation sibling supplies the visible variation ledger read by V.

Next form the controlled Riemann sums. The window row C records which integrand and integrator windows are paired, and the $\mathsf{RiemannSum}^{\uparrow}$ row S records the tagged finite sums over those windows.

The tolerance and readback rows then force the terminal route. The dyadic ledger D supplies mesh and sum-difference margins, R records the regular rational trace, and E is the only Real seal available to the consumer.

Transport and replay through H, K, P, N preserve this finite route. Therefore the public integral read cannot escape to measure completion, quotient function equality, a hidden refinement selector, rough-path enhancement, countable choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegral}^{\uparrow}$ packet is refuted if a consumer reaches the Real integral seal while omitting the controlled-window row C, the Riemann-sum row S, the dyadic tolerance row D, or the regular readback row R.

Independence witness: BoundedVariation, RegulatedFunction, RiemannSum, DyadicRatCore, RegSeqRat, Real: none is bijective with $\mathsf{YoungIntegral}^{\uparrow}$ because the packet jointly carries integrand and integrator rows, a variation-control ledger, controlled windows, finite sums, dyadic tolerances, regular readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegral}^{\uparrow}$ packet is generated from regulated integrand and integrator rows, a p-variation control ledger, controlled Riemann-sum windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3910.1, Definition 3910.2, and Definition 3910.3 seed the finite controlled-sum route for Young integration.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected refinement sequences, host rough-path lifts, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked integrand admission, integrator admission, variation-control exactness, controlled-window matching, Riemann-sum handoff, dyadic tolerance readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegralUp.

Chapter 3911

A Concrete Naming Certificate for $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$

The $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet realizes the type-checking-as-classifier-membership vision of chapter 9367 as a finite NameCert carrier. It reads the classifier-membership packet of chapter 476, the typing classifier of chapter 489, the subject-reduction surface of chapter 483, the derivation trace of chapter 598, and the proof-erasure route of chapter 642. The packet records membership, derivation, preservation, and erasure rows together; it does not export a host type checker, semantic conversion oracle, quotient equality of terms, hidden proof object, or ambient normalization theorem.

Definition 3911.1 (TypeCheckingClassifierMembership carrier)**.** *A* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, J, E, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the term-history row whose membership is being checked.*
2. *J is the finite typing-judgment ledger, read through* $\mathsf{ClassifierTyping}^{\uparrow}$*.*
3. *E is the* Ext *classifier-membership row, read through* $\mathsf{TypeClassifierMembership}^{\uparrow}$*.*
4. *D is the derivation trace row, read through* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$*.*
5. *S is the subject-reduction row, recording the displayed preservation handoff into* $\mathsf{SubjectReduction}^{\uparrow}$*.*
6. *Q is the proof-erasure row, read through* $\mathsf{ProofTermErasureTrace}^{\uparrow}$ *and separating public erased terms from proof evidence.*
7. *H records componentwise* hsame *transport for the term, judgment, membership, derivation, preservation, erasure, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a type-checking request through classifier membership, derivation trace, subject reduction, and erasure readback.*
9. *P is the* Pkg *provenance over* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$*,* $\mathsf{TypeClassifierMembership}^{\uparrow}$*,* $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$*,* $\mathsf{ProofTermErasureTrace}^{\uparrow}$*,* BHist*,* Ext*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host type checker, hidden conversion relation, semantic equality of terms, quotient term representative, proof object outside the erasure row, or global normalization theorem.

Lean4 (definition) `BEDC.Derived.TypeCheckingClassifierMembershipUp.taste gate`

Theorem 3911.2 (TypeCheckingClassifierMembership NameCert obligations)**.** *For every accepted* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *carrier* $M = (T, J, E, D, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingC}}$ *surface is exhausted by term-row admission, typing-ledger admission,* Ext *membership exposure, derivation-trace exposure, subject-reduction handoff, proof-erasure separation, componentwise transport, displayed replay, provenance, and local naming. No accepted consumer may replace those rows by host type checking, hidden conversion, semantic equality of terms, quotient representatives, proof objects outside* Q*, or ambient normalization.*

Proof. Project the carrier of Definition 3911.1. The rows T, J, E expose the term, finite typing ledger, and Ext membership surface used by the classifier-membership read.

The derivation and preservation rows are also displayed. The row D records the finite type-checking derivation trace, and S records the subject-reduction handoff that may be consumed only through the visible preservation boundary.

The erasure row Q separates proof evidence from public erased terms. It supplies the proof-term erasure trace without adding a proof object outside the packet.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the excluded host checker, conversion oracle, semantic equality, quotient representative, hidden proof object, and normalization theorem have no coordinate in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 3911.3 (TypeCheckingClassifierMembership subject-reduction handoff)**.** *Every subject-reduction read exported by an accepted* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{TypeClassifierMembership}^{\uparrow} \longmapsto \mathsf{ClassifierTyping}^{\uparrow} \longmapsto \mathsf{TypeCheckingDerivationTrace}^{\uparrow} \longmapsto \mathsf{SubjectReduction}^{\uparrow} \longmapsto \mathsf{Proc}$$

as recorded by E, J, D, S, Q*. The handoff preserves classifier membership and public erasure through* H, C, P, N*, and it exports no host typing judgment, hidden conversion relation, semantic equality of terms, quotient term representative, proof object outside* Q*, or ambient normalization theorem.*

Proof. Begin with the membership row E and the typing ledger J. The type-classifier membership and classifier-typing siblings already route type membership through finite Ext, Cont, SigRel, and NameCert data.

Next read the derivation trace D. It records the rule ledger and continuation route used by the type-checking run, so the preservation request reaches $\mathsf{SubjectReduction}^{\uparrow}$ only through the displayed trace.

The subject-reduction row S is therefore a handoff, not a hidden theorem. The proof-erasure row Q records which public erased data may be read after preservation while keeping proof evidence inside the trace boundary.

Transport and replay through H, C, P, N preserve this route. Therefore a consumer receives exactly classifier membership, derivation trace, preservation handoff, and erasure separation, with no host typing judgment or normalization oracle. □

Falsifiable prediction: An accepted $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet is refuted if a subject-reduction consumer reaches S without the membership row E, derivation trace D, proof-erasure row Q, and structural route H, C, P, N.

Independence witness: TypeClassifierMembership, ClassifierTyping, SubjectReduction, TypeCheckingDerivationTrace, ProofTermErasureTrace: none is bijective with $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ because the packet jointly carries term rows, typing ledgers, Ext membership, derivation traces, subject-reduction handoff, proof-erasure separation, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}$.

Constructive story: A $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ packet is generated from a term-history row, finite typing ledger, Ext membership row, derivation trace, subject-reduction handoff, proof-erasure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3911.1, Definition 3911.2, and Definition 3911.3 seed the finite type-checking classifier-membership packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifierMembership}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host type checking, hidden conversion, semantic equality of terms, quotient term representatives, proof objects outside erasure rows, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked term-row admission, typing-ledger admission, Ext membership exposure, derivation-trace exposure, subject-reduction handoff, proof-erasure separation, and structural transport for BEDC.Derived.TypeCheckingClassifierMembershipUp.

Chapter 3912

A Concrete Naming Certificate for $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$

Following Bishop and Bridges constructive completion and Bourbaki's universal-mapping language, the $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$ packet records one finite BEDC certificate comparing the reflector, comparison, adjunction, and monad presentations of Cauchy completion. It reads the reflector route of chapter 1585, the comparison surface of chapter 1954, the adjunction handoff of chapter 1456, and the monad schedule of chapter 378 through the same finite real-seal route. The packet names the common certificate surface only; it does not construct quotient completions, choose representatives, identify host objects, or appeal to ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 3912.1 (CauchyCompletionReflectorComparison carrier)**.** *A* $\mathsf{CauchyCompletionReflectorComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (\rho, K_0, A, M, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. ρ *is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row exposing unit, counit, idempotent, extension, reflection, and local real-transport data.*
2. K_0 *is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row exposing the comparison between unit-and-universal-property and functorial metric-completion presentations.*
3. A *is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *row exposing the finite unit-counit handoff surface.*
4. M *is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *row exposing the finite schedule, diagonal readback, and monad composition boundary.*
5. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *window route inspected by all four completion presentations.*
6. Q *is the shared* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route reached after the displayed windows.*
7. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window route.*
8. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* W, D, Q *have been displayed.*
9. H *records componentwise* hsame *transport for the four presentation rows, the shared real-seal rows, replay, provenance, and naming.*
10. C *records displayed* Cont *replay of the route*

$$\rho, K_0, A, M \longmapsto W, D, Q \longmapsto E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* CauchyCompletionReflectorComparison$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, CauchyCompletionComparison$^{\uparrow}$, CauchyCompletionAdjunction$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows and their displayed transports. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, open-cover compactness, or completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorComparisonUp.CauchyCompletionReflectorComparisonTasteGate`
`single carrier alignment`

Theorem 3912.2 (CauchyCompletionReflectorComparison NameCert obligations)**.** *For every accepted* CauchyCompletionReflectorComparison$^{\uparrow}$ *carrier*

$$K = (\rho, K_0, A, M, W, Q, D, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$ *surface consists exactly of reflector-route admission through ρ; comparison-route exposure through K_0; adjunction handoff through A; monad schedule compatibility through M; shared finite real-seal routing through W, D, Q, E; componentwise transport through H; replay through C; provenance through P; and local naming through N. It exports no quotient completion, selected representative, countable choice, host equality, open-cover compactness, or* Real$^{\uparrow}$ *completeness.*

Proof. Carrier admission is tuple projection from Definition 3912.1. The accepted packet displays exactly the four presentation rows, the shared window-tolerance-readback-seal route, and the structural rows.

The reflector, comparison, adjunction, and monad rows are then read as sibling routes. The row ρ contributes the reflector route, K_0 contributes the comparison route, A contributes the adjunction handoff, and M contributes the schedule and diagonal-readback compatibility. None of these rows contains an independent terminal seal.

The common real-facing route is exhausted by W, D, Q, E. A consumer must inspect the finite StreamName$^{\uparrow}$ windows, the DyadicRatCore$^{\uparrow}$ tolerance ledger, and the RegSeqRat$^{\uparrow}$ readback before reaching the terminal Real$^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name only the displayed packet. Since no further coordinate remains in the carrier, quotient completions, selected representatives, countable choice, host equality, open-cover compactness, and ambient Real$^{\uparrow}$ completeness cannot be exported. □

Closure status for NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$.

Constructive story: A CauchyCompletionReflectorComparison$^{\uparrow}$ packet is generated from reflector, comparison, adjunction, and monad presentation rows, shared StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3912.1 and Definition 3912.2 seed the finite certificate comparing reflector, comparison, adjunction, and monad completion presentations over one shared real-seal route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionReflectorComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, open-cover compactness, Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector-route admission, comparison-route exposure, adjunction handoff, monad schedule compatibility, shared real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionReflectorComparisonUp.

Chapter 3913

A Concrete Naming Certificate for CompactRegularCauchySubsequence$^{\uparrow}$

Following Bishop and Bridges on compact metric sequential arguments and regular Cauchy real names, the CompactRegularCauchySubsequence$^{\uparrow}$ packet records the finite route by which compact metric inspection supplies a regular Cauchy subsequence window before any Real or Completion consumer reads the result. It sits over the compact metric source of chapter 129, the mature regular-subsequence route of chapter 381, the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive compact-subsequence certificate for L10 completion consumers; it is not open-cover compactness, countable choice, quotient stream equality, host sequence equality, or ambient Real$^{\uparrow}$ completeness.

Definition 3913.1 (Compact regular Cauchy subsequence carrier)**.** *A* CompactRegularCauchySubsequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, S, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row carrying finite-net, located-distance, and compact-bundle inspection data.*
2. *S is the source* StreamName$^{\uparrow}$ *row for the sequence windows inspected by the compact metric consumer.*
3. *T is the monotone-cofinal unary reindexing ledger consumed by the* RegularCauchySubsequence$^{\uparrow}$ *surface.*
4. *W is the selected finite subsequence-window row, pairing each retained compact window with the inspected source window that produced it.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger attached to compact mesh radii, selected-window distances, and regular Cauchy tail budgets.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the selected windows and their transported regularity evidence.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row, exposed only after the compact source, subsequence ledger, finite windows, dyadic tolerances, and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for K, S, T, W, D, R, E.*
9. *C records displayed* Cont *replay from compact metric inspection through finite windows, subsequence selection, dyadic tolerances, regular readback, and the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, T, W, D, R, E, H, C, P, N$. It has no coordinate for an externally selected infinite subsequence, open-cover compactness theorem, quotient stream representative, host sequence equality, host real equality, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactRegularCauchySubsequenceUp.CompactRegularCauchySubsequenceTasteGate`
`single carrier alignment`

Theorem 3913.2 (Compact regular Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ *carrier*

$$Q = (K, S, T, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through K, source-window exposure through S, monotone-cofinal subsequence exposure through T, selected-window exposure through W, dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3913.1. The finite packet exposes exactly the compact metric source, source stream windows, reindexing ledger, selected windows, dyadic ledger, regular rational readback, Real seal, and the structural provenance rows.

Read the compact and subsequence rows first. The row K supplies only displayed compact metric inspection data, while S, T, W record the source windows, monotone-cofinal route, and selected finite windows consumed by the regular-subsequence surface.

The analytic handoff is then forced through D, R, E. The dyadic ledger records the mesh and tail budgets, the regular rational row reads the selected windows as regular data, and the Real seal row is downstream of those rows.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no other coordinate exists, the listed rows exhaust the local NameCert obligations. □

Theorem 3913.3 (Compact regular Cauchy subsequence route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The compact source row supplies finite inspection windows, the regular-subsequence row supplies the monotone-cofinal selected-window ledger, and the dyadic and regular rational rows supply the Cauchy budgets before the terminal Real seal is exposed.

Proof. Begin with an accepted carrier $Q = (K, S, T, W, D, R, E, H, C, P, N)$. The compact source row K is the only compact metric coordinate, and the source windows are read through S.

Apply the regular-subsequence public export of Definition 381.9. The selected windows are named by the monotone-cofinal ledger T and the finite window row W, so the consumer cannot replace them by an unrecorded subsequence selector.

Next read the dyadic and regular rational rows. The ledger D carries the compact mesh and tail budgets, and R repacks the selected windows as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The Real seal E is therefore reached only after the compact source, finite windows, subsequence ledger, dyadic tolerance rows, and regular readback have been displayed. Transport and replay through H, C, P, N preserve that same route, so open-cover compactness, countable choice, quotient streams, host equality, and ambient completeness remain outside the exported packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRegularCauchySubsequence}^{\uparrow}$ packet is generated from a CompactMetric source row, StreamName finite windows, a RegularCauchySubsequence monotone-cofinal ledger, selected finite windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3913.1, Definition 3913.2, and Definition 3913.3 seed the compact-source regular Cauchy subsequence route over CompactMetric, StreamName, RegularCauchySubsequence, DyadicRatCore, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRegularCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient stream equality, host sequence equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, monotone-cofinal selected-window routing, dyadic tolerance accounting, RegSeqRat readback exactness, RealUp seal nonescape, and selector-refusal proofs for BEDC.Derived.CompactRegularCauchySubsequenceUp.

Chapter 3914

A Concrete Naming Certificate for $\mathsf{LindelofSpace}^{\uparrow}$

Following Lindelof, Bourbaki, and Kelley on second-countable topology, the $\mathsf{LindelofSpace}^{\uparrow}$ packet records the countable subcover route that is visible from a displayed countable-base presentation. It sits over the countable-base carrier of chapter 1901, the local countable-neighbourhood route of chapter 2053, the generated-open classifier of chapter 88, and the finite inspection windows of chapter 126. The packet names only displayed cover enumeration windows, extracted subcover ledgers, base-refinement routes, transport, replay, provenance, and local naming; it does not select arbitrary subcovers by choice, assert compactness, or add a host open-family oracle.

Definition 3914.1 (Lindelof space carrier)**.** *A* $\mathsf{LindelofSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, T, B, U, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source topology carrier row whose finite histories supply the only points and opens inspected by the packet.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open classifier row used by every open-cover and subcover read.*
3. *B is the* $\mathsf{SecondCountable}^{\uparrow}$ *base-enumeration row, recording the finite basic-open windows through which cover membership is tested.*
4. *U is the countable cover enumeration row, displaying the open-cover entries visible to the packet through T and the finite windows of B.*
5. *S is the extracted countable subcover ledger, recording the selected cover indices and the basic cells whose refinements witness coverage.*
6. *R is the base-refinement route from displayed basic cells in B into cover entries in U, with every refinement read tied to the same topology row T.*
7. *H stores componentwise* hsame *transport for the source topology, topology classifier, base enumeration, cover enumeration, extracted subcover ledger, refinement route, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a point or basic-cell read through the topology classifier, countable-base window, cover enumeration, extracted subcover ledger, refinement route, transport, provenance, and local name.*

9. *P is the* Pkg *provenance row over* $\mathsf{LindelofSpace}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{FirstCountableSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}$ *row.*

The carrier exports a countable subcover only as the displayed ledger S over the displayed cover enumeration U. It has no coordinate for arbitrary open-cover compactness, host choice of subcovers, quotient equality of opens, or a fuel-erased enumeration of all covers.

Lean4 ✓ `BEDC.Derived.LindelofSpaceUp.LindelofSpaceTasteGate single carrier alignment`

Theorem 3914.2 (Lindelof countable-cover route)**.** *For an accepted* $\mathsf{LindelofSpace}^{\uparrow}$ *carrier*

$$Q = (X, T, B, U, S, R, H, C, P, N),$$

every countable-subcover read factors through the finite route

$$X, T, B, U, S, R, H, C, P, N.$$

The reader first opens the source topology and generated-open classifier, then inspects the displayed second-countable base window, then reads the cover enumeration, extracted subcover ledger, base-refinement route, componentwise transport, replay, provenance, and local naming rows. No consumer receives a host family of all open covers, an unrestricted subcover selector, an open-cover compactness theorem, or equality of opens outside the displayed classifier.

Proof. Project the accepted carrier through Definition 3914.1. The rows $X, T, B, U, S, R, H, C, P, N$ are the complete surface available to a consumer.

The cover read starts in X and T. A displayed open-cover entry is not read as a host open set; it is read through the generated-open classifier and the finite base window B.

The extracted subcover is then read through S and R. The subcover ledger stores the visible cover indices, while the refinement row records how displayed basic cells refine into those entries.

Finally H, C, P, N transport, replay, source, and name the same route. Since no other coordinate is present, the packet cannot export arbitrary open-cover compactness, a choice-selected subcover, quotient equality of opens, or a host oracle for all covers. □

Theorem 3914.3 (Lindelof second-countable handoff)**.** *Every accepted* $\mathsf{SecondCountable}^{\uparrow}$ *packet with a displayed countable open-cover enumeration induces a* $\mathsf{LindelofSpace}^{\uparrow}$ *read by replaying the cover through finite base windows and recording the extracted countable subcover ledger. The induced read keeps the same source topology, generated-open classifier, countable-base enumeration, stream-window discipline, transport, replay, provenance, and local naming rows; only the cover enumeration U, extracted ledger S, and refinement route R are added as Lindelof rows.*

Proof. Start with the accepted $\mathsf{SecondCountable}^{\uparrow}$ packet. Its source row, topology classifier, displayed base enumeration, stream-window discipline, transport, replay, provenance, and local name already have the BEDC roles required by Definition 3914.1.

Read the displayed countable cover through the same finite base windows. For each accepted basic cell, the refinement route records the cover entry that contains it, and the extracted ledger records the visible cover index.

Replay the finite base-window enumeration to form the countable subcover ledger. This is a ledger read over B, U, S, R, not a choice principle over a host family of opens.

The remaining rows are unchanged. The handoff therefore adds only the Lindelof cover, ledger, and refinement rows, while preserving the second-countable topology source and excluding compactness, arbitrary subcover selection, and host open equality. □

Closure status for $\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3914.1, Definition 3914.2, and Definition 3914.3 seed the finite countable-subcover interface over displayed topology rows, second-countable base windows, cover enumerations, extracted subcover ledgers, refinement routes, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LindelofSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary subcover choice, quotient equality of opens, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable-cover routing, second-countable handoff, base-refinement replay, transport, provenance, and local naming for BEDC.Derived.LindelofSpaceUp.

Constructive story: A $\mathsf{LindelofSpace}^{\uparrow}$ packet is generated from one source topology row, one Topology classifier row, one SecondCountable base-enumeration row, one countable cover enumeration row, one extracted countable subcover ledger, one base-refinement route, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3915

A Concrete Naming Certificate for $\mathsf{LindelofCoverRefinement}^{\uparrow}$

Following Lindelof and Kelley on countable subcovers and cover refinements, the $\mathsf{LindelofCoverRefinement}^{\uparrow}$ packet records the finite route from a displayed Lindelof cover read to a countable refinement ledger. It sits over the countable-subcover carrier of chapter 3914, the countable-base window of chapter 1901, the generated-open classifier of chapter 88, and the finite stream windows of chapter 126. The packet names only displayed cover rows, countable subcover ledgers, base-cell refinements, transport, replay, provenance, and local naming; it does not select arbitrary subcovers, assert compactness, or expose a host open-family oracle.

Definition 3915.1 (Lindelof cover-refinement carrier)**.** *A* $\mathsf{LindelofCoverRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, T, L, B, U, S, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the source topology row whose finite histories supply the inspected points and opens.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *generated-open classifier row used by every cover and refinement read.*
3. *L is the accepted* $\mathsf{LindelofSpace}^{\uparrow}$ *countable-subcover row.*
4. *B is the* $\mathsf{SecondCountable}^{\uparrow}$ *countable-base window that names the basic cells available for refinement.*
5. *U is the displayed open-cover enumeration row.*
6. *S is the countable subcover ledger inherited from the Lindelof row.*
7. *F is the cover-refinement ledger, recording which basic cells refine into which displayed subcover entries.*
8. *H stores componentwise* hsame *transport for the source topology, topology classifier, Lindelof row, base window, cover enumeration, subcover ledger, refinement ledger, replay, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from point and basic-cell reads through $T, L, B, U, S, F, H, P, N$.*
10. *P is the* Pkg *provenance row over* $\mathsf{LindelofCoverRefinement}^{\uparrow}$, $\mathsf{LindelofSpace}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LindelofCoverRefinement}^{\uparrow}}$ *row.*
The carrier exports a countable refinement only as the displayed ledger F over the displayed subcover ledger S. It has no coordinate for arbitrary cover choice, compactness, equality of opens outside T, or a hidden enumeration of all open families.

Theorem 3915.2 (Lindelof cover-refinement NameCert obligations)**.** *Every accepted* $\mathsf{LindelofCoverRefinement}^{\uparrow}$ *carrier*

$$R = (X, T, L, B, U, S, F, H, C, P, N)$$

exposes exactly the rows $X, T, L, B, U, S, F, H, C, P, N$ as its local NameCert obligation surface. A consumer may read countable cover refinement only after it has read the topology classifier, Lindelof subcover row, second-countable base window, displayed cover enumeration, subcover ledger, refinement ledger, componentwise transport, replay, provenance, and local name. The same consumer receives no compactness theorem, no host subcover selector, no quotient equality of opens, and no unbounded cover-family object.

Proof. Project the accepted carrier through Definition 3915.1. The finite row list $X, T, L, B, U, S, F, H, C, P, N$ is the whole surface available to a NameCert consumer.

The cover read starts with X, T, L, U, S. These rows expose the displayed topology, generated-open classifier, Lindelof countable-subcover route, cover enumeration, and subcover ledger.

The refinement read then passes through B and F. The base window names the basic cells, while the refinement ledger records their displayed containment in the subcover entries.

Finally H, C, P, N transport, replay, source, and name that same route. Since no other coordinate is present, the packet cannot export compactness, a host subcover selector, quotient equality of opens, or an unbounded cover-family oracle. □

Theorem 3915.3 (Lindelof countable-subcover refinement route)**.** *Every countable-subcover read of an accepted* $\mathsf{LindelofCoverRefinement}^{\uparrow}$ *carrier factors through the finite route*

$$X, T, L, B, U, S, F, H, C, P, N.$$

The route first consumes the $\mathsf{LindelofSpace}^{\uparrow}$ *subcover ledger, then refines it through the displayed* $\mathsf{SecondCountable}^{\uparrow}$ *base window, and finally replays the refinement through the* $\mathsf{Topology}^{\uparrow}$ *classifier and* $\mathsf{StreamName}^{\uparrow}$ *inspection discipline. It does not treat* $\mathsf{LindelofSpace}^{\uparrow}$ *as an undifferentiated cover oracle.*

Proof. Start with the Lindelof row L and its displayed cover enumeration U and subcover ledger S. These rows provide the only countable-subcover data visible to the carrier.

Next read the second-countable base window B and the refinement ledger F. Each refinement step is tied to a displayed basic cell and to a displayed subcover entry, so the route remains inside finite base-window inspection.

The topology and stream rows T and C replay the same refinement through generated-open membership and finite inspection windows. Transport and provenance through H, P, N preserve the local NameCert boundary and prevent the route from becoming a compactness principle or an unrestricted cover selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{LindelofCoverRefinement}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LindelofCoverRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3915.1, Definition 3915.2, and Definition 3915.3 seed the finite countable-cover refinement interface over displayed topology rows, Lindelof subcover ledgers, second-countable base windows, cover enumerations, refinement ledgers, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{LindelofCoverRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim compactness, arbitrary subcover choice, quotient equality of opens, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable-subcover routing, base-refinement replay, transport, provenance, and local naming for BEDC.Derived.LindelofCoverRefinementUp.

Constructive story: A LindelofCoverRefinement$^{\uparrow}$ packet is generated from one source topology row, one Topology classifier row, one LindelofSpace subcover row, one SecondCountable base-window row, one displayed cover enumeration row, one subcover ledger, one cover-refinement ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3916

A Concrete Naming Certificate for $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ packet records a finite dyadic refinement-tree certificate for closed bounded interval cover work. It sits between the dyadic cover rows of chapter 1428, the finite comparison surface of chapter 3430, and the endpoint-and-containment packet of chapter 1591. The packet names the tree of displayed subdivisions and retained cover leaves; it is not open-cover compactness, a host subcover selector, a selected Lebesgue number, quotient equality of real names, or countable choice.

Definition 3916.1 (Interval cover refinement tree carrier)**.** *An* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, D, R, M, L, E, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, naming the lower endpoint, upper endpoint, endpoint-order, containment, StreamName, RegSeqRat, and Real-seal data being covered.*
2. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *cover row whose finite dyadic endpoints and membership rows are inspected by the tree.*
3. *R is the* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *comparison row that records how a retained child cover refines its parent cover entry.*
4. *M is the finite mesh-depth ledger, listing the dyadic scale used at each visited tree node.*
5. *L is the finite leaf row. Each leaf records a dyadic subinterval, its parent node, and the displayed cover member that contains it at the tolerance named by M.*
6. *E is the edge row, recording the binary split relation between a node and its two dyadic children, together with the inherited endpoint-containment witnesses.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the interval endpoints and cover endpoints at the mesh scales named by M.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint and cover windows.*
9. *A is the visible* $\mathsf{Real}^{\uparrow}$ *seal row reached only through I, W, Q and the displayed dyadic cover data.*

10. *H stores componentwise* hsame *transport for interval, cover, refinement, mesh, leaf, edge, window, readback, seal, provenance, and local naming rows.*

11. *C stores displayed* Cont *replay from closed interval containment to dyadic subdivision, from subdivision to cover membership, from membership to refinement comparison, and from windows to the visible Real seal.*

12. *P is the* Pkg *provenance over* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{FiniteCoverRefinement}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$ *row.*

The accepted carrier is only the finite tree packet displayed above. It contains no arbitrary open cover, no host finite-subcover selector, no selected point of the interval, no selected Lebesgue number, no quotient equality of real names, no countable-choice functional, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.IntervalCoverRefinementTreeUp.TasteGate.IntervalCoverRefinementTreeTasteGate single carrier alignment`

Theorem 3916.2 (Interval cover refinement tree NameCert obligations)**.** *For an accepted packet $T = (I, D, R, M, L, E, W, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$ *obligations are exhausted by closed bounded interval admission through I, dyadic cover admission through D, finite refinement admission through R, mesh-depth admission through M, leaf and edge admission through L and E, finite-window readback through W and Q, Real sealing through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier row by row. The interval source is exactly the $\mathsf{ClosedBoundedInterval}^{\uparrow}$ packet I, the cover source is exactly the $\mathsf{DyadicIntervalCover}^{\uparrow}$ packet D, and the comparison source is exactly the $\mathsf{FiniteCoverRefinement}^{\uparrow}$ packet R.

The tree rows add only finite subdivision data. The mesh ledger M, leaves L, and edges E list the visited nodes, their dyadic children, and the retained cover members, so every local admission check is a finite projection from the displayed BHist packet.

The endpoint and cover reads remain on the displayed route. The $\mathsf{StreamName}^{\uparrow}$ row W, $\mathsf{RegSeqRat}^{\uparrow}$ row Q, and $\mathsf{Real}^{\uparrow}$ seal A are replayed by C and transported by H. Since the packet has no row for an arbitrary open cover, selected subcover, selected point, Lebesgue number choice, quotient equality, or completeness principle, no such datum can enter the local NameCert obligations. □

Theorem 3916.3 (Dyadic subcover route)**.** *Every retained subcover read exported by an accepted* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *packet factors through the finite route*

$$I \longmapsto D \longmapsto R \longmapsto M \longmapsto L \longmapsto E \longmapsto (W, Q, A, H, C, P, N).$$

In particular, a consumer may inspect only the dyadic child intervals, the finite leaf-cover assignments, the displayed refinement comparison rows, and the finite endpoint windows named in the carrier.

Proof. Start from the closed interval row I. It supplies the endpoint and containment surface before any cover or tree row is inspected.

Next read the dyadic cover and refinement rows. The cover packet D lists the available dyadic cover members, and the refinement packet R records how a child cover assignment is compared with a parent cover assignment. The mesh, leaf, and edge rows M, L, E then expose the finite tree spine on which these comparisons are made.

Finally pass through the structural rows. Transport H, replay C, provenance P, and local name N preserve the same finite route, while W, Q, A keep all endpoint reads behind finite StreamName, RegSeqRat, and Real-seal rows. No additional subcover or compactness object is available. □

Theorem 3916.4 (Choice-free nonescape)**.** *An* $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ *consumer cannot escape the finite refinement tree by invoking open-cover compactness, a host finite-subcover selector, a selected Lebesgue number, quotient equality of real names, a chosen point in the interval, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every accepted read is determined by the finite rows* $I, D, R, M, L, E, W, Q, A, H, C, P, N$ *and their displayed* hsame *and* Cont *routes.*

Proof. The carrier definition lists every coordinate of the packet. The analytic coordinates are $I, D, R, M, L, E, W, Q, A$, and the structural coordinates are H, C, P, N.

Apply the NameCert obligation theorem. It makes those rows the whole local naming surface and gives no replay path to an arbitrary open cover, host subcover selector, selected Lebesgue number, quotient equality, or completeness coordinate.

The dyadic subcover route supplies the consumer-facing consequence. Any accepted read must pass through the same finite interval, cover, refinement, tree, window, readback, and seal rows, so choice-based or host-level data cannot affect the result. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}$**.**

Constructive story: An $\mathsf{IntervalCoverRefinementTree}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a DyadicIntervalCoverUp cover row, a FiniteCoverRefinementUp comparison row, finite mesh-depth, leaf, and edge rows, StreamName windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3916.1, Definition 3916.2, Definition 3916.3, and Definition 3916.4 bind the finite dyadic refinement-tree seed over closed bounded interval covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntervalCoverRefinementTree}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.IntervalCoverRefinementTreeUp.TasteGate.IntervalCoverRefinementTreeTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, Heine–Borel compactness, selected subcovers, selected Lebesgue numbers, quotient real equality, countable choice, Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, dyadic leaf coverage, refinement transport, Real-seal nonescape, and tree-ledger exactness for the displayed packet.

Chapter 3917

A Concrete Naming Certificate for $\mathsf{NeumannSeries}^{\uparrow}$

Following Banach algebra treatments of Neumann series, the $\mathsf{NeumannSeries}^{\uparrow}$ packet records the finite inverse-series certificate by which a norm-small operator or algebra element exposes partial sums, residual checks, and inverse-readback rows before any host inverse selector is available. It sits beside the Banach-algebra source of chapter 2189, the bounded-operator route of chapter 1644, the scalar norm-bound surface of chapter 3603, the resolvent identity packet of chapter 3715, and the spectral-radius boundary of chapter 2352. The packet names a finite inverse route: it does not select a completed inverse from the host, assert a completed spectrum, or import a quotient equality of operators.

Definition 3917.1 (Neumann inverse-series carrier)**.** *A* $\mathsf{NeumannSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (A, T, K, W, S, Q, R, E, H, C, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{BanachAlgebra}^{\uparrow}$ *source row, carrying multiplication, unit, norm, completeness, product-control, and local naming data.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *or algebra-action row whose smallness is being inspected.*
3. *K is the norm-smallness ledger, read through* $\mathsf{OperatorNorm}^{\uparrow}$ *or the Banach-algebra norm, recording the displayed bound below the unit row.*
4. *W is the finite partial-sum window for the inspected powers $1, T, T^2, \ldots, T^m$.*
5. *S is the finite sum row naming the displayed Neumann partial sums and their multiplication order.*
6. *Q is the residual row, recording the left and right error checks for $(1-T)S_m$ and $S_m(1-T)$ inside the same algebra surface.*
7. *R is the inverse-check row saying which residual bounds are strong enough for an inverse-facing consumer to read the packet.*
8. *E is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for rational norm bounds, tail estimates, and terminal real seals.*
9. *H records componentwise* hsame *transport for A, T, K, W, S, Q, R, E.*

10. *C records displayed* Cont *replay from the smallness request through the finite partial-sum and residual rows.*
11. *P is the* Pkg *provenance over* $\mathsf{NeumannSeries}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ResolventIdentity}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host-chosen inverse, completed resolvent set, completed spectrum, spectral theorem, quotient operator equality, selected approximate eigenvectors, or countable choice.

Lean4 (definition) `BEDC.Derived.NeumannSeriesUp.NeumannSeriesUp`

Theorem 3917.2 (Neumann series NameCert obligations)**.** *For every accepted* $\mathsf{NeumannSeries}^{\uparrow}$ *carrier* $N = (A, T, K, W, S, Q, R, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach-algebra source, bounded-operator or algebra-action row, norm-smallness ledger, finite power window, finite partial sums, residual checks, inverse-check row, rational and real readback, transport, replay, provenance, and local naming. Every inverse-facing consumer must read those rows before using the inverse-series conclusion.*

Proof. Project the carrier of Definition 3917.1. The algebra source A and operator row T are the only endpoints on which multiplication and powers may be read.

The smallness and finite-sum rows are K, W, S. The norm-smallness ledger is displayed before any partial sum is public, and the finite power window fixes the product order used by the sum row.

The residual and readback rows are Q, R, E. They expose the left and right inverse checks and the rational-to-real tail estimates before a consumer may treat the packet as an inverse route.

Finally, H, C, P, L transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a host inverse, completed resolvent set, completed spectrum, quotient equality, or choice schedule, the listed rows exhaust the local NameCert obligation surface. □

Theorem 3917.3 (Neumann inverse route)**.** *Every resolvent, spectral-radius, or inverse-operator consumer of an accepted* $\mathsf{NeumannSeries}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BanachAlgebra}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto K, W, S, Q, R, E \longmapsto H, C, P, L.$$

Thus a downstream read receives a norm-smallness ledger, finite partial-sum windows, residual and inverse-check rows, rational and real readback, transport, replay, provenance, and local naming before it may cite an inverse-series conclusion.

Proof. Start with an accepted packet $N = (A, T, K, W, S, Q, R, E, H, C, P, L)$. The Banach-algebra and bounded-operator rows fix the multiplication, unit, action, and norm-bound surface.

The route then passes through K, W, S. These rows expose the smallness bound, the inspected powers, and the finite partial sums; no completed infinite sum is available at this stage.

The inverse-facing part is Q, R, E. The residual row records the two-sided error checks, the inverse-check row states which displayed residual bound is consumed, and the readback row seals the rational tail estimates as real data.

The structural rows H, C, P, L preserve this route for chapter 3715 and chapter 2352 consumers. Because the packet lacks host inverse, completed spectrum, quotient equality, and choice coordinates, every such consumer factors through the displayed inverse-series route. □

Falsifiable prediction: Every accepted $\mathsf{NeumannSeries}^{\uparrow}$ inverse-facing read exposes BanachAlgebra, BoundedLinearOperator, norm-smallness, finite partial-sum, residual, inverse-check, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows before citing the inverse-series conclusion.

Independence witness: banachalgebra, boundedlinearoperator, operatornorm, resolvent-identity, spectral-radius: $\mathsf{NeumannSeries}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries norm-smallness, finite power windows, partial sums, residual checks, inverse-readback rows, rational and real tail seals, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NeumannSeries}^{\uparrow}$ packet is generated from a BanachAlgebraUp source row, a BoundedLinearOperatorUp or algebra-action row, a norm-smallness ledger, finite partial-sum windows, residual and inverse-check rows, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3917.1, Definition 3917.2, and Definition 3917.3 seed the finite Neumann inverse-series route consumed by resolvent, spectral, and inverse-operator packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeumannSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a completed inverse operator, a completed resolvent set, a completed spectrum, quotient operator equality, selected approximate eigenvectors, or countable choice.

Upgrade path: Obligation closure requires checked carrier admission, norm-smallness exactness, partial-sum window stability, residual readback, inverse-check exactness, rational and real tail sealing, transport and replay stability, and consumer nonescape for BEDC.Derived.NeumannSeriesUp.

Chapter 3918

A Concrete Naming Certificate for $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ packet records the finite tail-window stability bridge used by the regular-Cauchy route of chapter 9238. It sits after the finite window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The bridge names shared finite tail-window transport; it is not a hidden limit selector, quotient stream equality, host equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3918.1 (Cauchy window stability bridge carrier)**.** *A* $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S_0, S_1, T, D, R_0, R_1, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows for the two regular-Cauchy presentations being compared.*
2. T *is the common finite tail-window row, listing the precision requests and indices beyond which the two presentations are read through the same displayed window schedule.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the shared tail window and to every rational comparison used by the bridge.*
4. R_0 *and* R_1 *are* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows extracted from* S_0 *and* S_1 *at the tolerances named by* D*.*
5. E_0 *and* E_1 *are the visible* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after the shared tail window, dyadic tolerance ledger, and regular rational readbacks have been displayed.*
6. H *stores componentwise* hsame *transport for stream windows, the common tail row, dyadic tolerances, regular readbacks, Real seals, provenance, and local naming rows.*
7. C *stores displayed* Cont *replay from precision requests to common tail windows, from windows to regular readbacks, from dyadic tolerances to comparison rows, and from readbacks to the two Real seals.*
8. P *is the* Pkg *provenance over* $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$ *row.*
The accepted carrier contains no selected limit, no quotient representative for streams, no host equality of real names, no countable-choice sequence, and no ambient completeness coordinate. It names only the finite common-tail bridge displayed by $T, D, R_0, R_1, E_0, E_1, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CauchyWindowStabilityBridgeUp.CauchyWindowStabilityBridgeTasteGate single carrier alignment`

Theorem 3918.2 (Cauchy window stability bridge NameCert obligations)**.** *For an accepted carrier $B = (S_0, S_1, T, D, R_0, R_1, E_0, E_1, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$ *obligations are exactly carrier admission; finite window exposure through S_0 and S_1; common tail-window admission through T; dyadic tolerance exactness through D; regular rational readback through R_0 and R_1; Real-seal admission through E_0 and E_1; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the carrier. The two stream rows S_0, S_1, the common tail row T, the dyadic ledger D, the readbacks R_0, R_1, and the seals E_0, E_1 are the entire analytic surface of the bridge.

The tail-window row is the only stability interface. It records which finite windows are shared for the precision request under consideration, while the dyadic ledger records the rational tolerance used to compare the two readbacks.

The structural rows add no hidden equality or limit object. Transport H, replay C, provenance P, and the local name N preserve the displayed rows and expose no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 3918.3 (Tail-window transport)**.** *Every stability read exported by an accepted* $\mathsf{CauchyWindowStabilityBrid}$ *packet factors through the finite route*

$$S_0, S_1 \longmapsto T \longmapsto D \longmapsto R_0, R_1 \longmapsto E_0, E_1 \longmapsto H, C, P, N.$$

Thus shared regular-Cauchy tail transport is determined by displayed $\mathsf{StreamName}^{\uparrow}$ *windows, one common finite tail row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and componentwise* hsame *replay.*

Proof. Begin with the two finite stream rows. A stability consumer may inspect only the windows exposed by S_0 and S_1.

Read the common tail row next. The row T names the shared finite window schedule for the current precision request, and the dyadic ledger D attaches the tolerance used to compare the two regular rational readbacks.

The regular readbacks and seals are downstream. Rows R_0, R_1 are consumed before E_0, E_1, and the structural rows H, C, P, N replay the same finite route. No selected tail, quotient stream, or countable-choice family is available outside this packet. □

Theorem 3918.4 (Real-seal stability handoff)**.** *For every accepted* $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ *packet, the two* $\mathsf{Real}^{\uparrow}$ *seal rows E_0 and E_1 are accessible only after the shared tail-window row T, dyadic tolerance row D, and regular rational readbacks R_0, R_1 have been displayed. Consequently a Real-completion consumer that cites this bridge receives a finite seal-stability handoff, not a proof of ambient* $\mathsf{Real}^{\uparrow}$ *completeness or quotient equality of real names.*

Proof. Project the carrier and isolate the seal coordinates. The rows E_0, E_1 occur only as terminal rows of the displayed bridge packet.

Apply the tail-window transport theorem. It forces every stability read through S_0, S_1, T, D, R_0, R_1 before the seals are available.

The refusal clause follows from carrier exhaustion. Since the packet has no coordinate for ambient completeness, quotient equality, host equality, selected limits, or countable choice, the Real-facing handoff can export only the displayed finite stability route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyWindowStabilityBridge}^{\uparrow}$ packet is generated from paired StreamName finite windows, a common finite tail-window row, a DyadicRatCore tolerance ledger, paired RegSeqRat readbacks, paired RealUp seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3918.1, Definition 3918.2, Definition 3918.3, and Definition 3918.4 bind the finite regular-Cauchy tail-window stability bridge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyWindowStabilityBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, host equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, common-tail exactness, dyadic tolerance transport, paired readback stability, Real-seal handoff, and nonescape for the displayed packet.

Chapter 3919

A Concrete Naming Certificate for MetaCICPiAdequacyBudget$^{\uparrow}$

The MetaCICPiAdequacyBudget$^{\uparrow}$ packet realizes the Pi-adequacy route of chapter 9383 as finite BEDC data. It sits beside the critical-path packet of chapter 765, the normalization frontier of chapter 892, the candidate socket of chapter 3566, the normalization-confluence bridge of chapter 1190, and the normalization audit packet of chapter 3779. The packet records the candidate-mediated budget needed by Pi adequacy; it does not claim unconditional global normalization or erase the candidate evidence consumed by the Lean route.

Definition 3919.1 (MetaCIC Pi adequacy budget carrier)**.** *A* MetaCICPiAdequacyBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (G, \Pi, N, A, S, F, O, H, C, P, L)$$

with the following displayed rows:

1. *G is the closed context and typing-frontier row for the Pi candidate being inspected.*
2. *Π is the Pi-candidate evidence row consumed by the route named by* `PiAdequacy.lean`*.*
3. *N is the neutral-normal evidence row used by the neutral branch of the adequacy argument.*
4. *A is the application-side adequacy handoff row, recording the candidate argument and the application result row.*
5. *S is the strong-normalization handoff row, naming the projections corresponding to* `piCandidate_implies_sn`*,* `neutralNormal_implies_sn`*, and* `sn_pi`*.*
6. *F is the frontier-refusal row, keeping candidate evidence, subject-reduction boundaries, and confluence boundaries visible whenever a consumer requests a stronger normalization claim.*
7. *O is the obligation-budget row for the remaining endpoint evidence needed by closed-consistency consumers.*
8. *H records componentwise* hsame *transport for the typing, Pi-candidate, neutral-normal, application, strong-normalization, frontier, and obligation rows.*
9. *C records displayed* Cont *replay from a Pi-adequacy request through G, Π, N, A, S, F, O.*
10. *P is* Pkg *provenance over* MetaCICPiAdequacyBudget$^{\uparrow}$*,* MetaCICNormalizationAuditPacket$^{\uparrow}$*,* MetaCICDecidableNe MetaCICFuelTrace$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$ *row.*
The carrier has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, global confluence, semantic equality, quotient syntax equality, or host evaluation soundness.

Lean4 (definition) `BEDC.Derived.MetaCICPiAdequacyBudgetUp.taste gate`

Theorem 3919.2 (MetaCIC Pi adequacy budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ *carrier* $B = (G, \Pi, N, A, S, F, O, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$ *surface consists exactly of typing-frontier admission, Pi-candidate evidence admission, neutral-normal evidence admission, application adequacy handoff, strong-normalization projection handoff, frontier refusal, obligation-budget exposure, componentwise transport, displayed replay, provenance, and local naming. A consumer receives no stronger normalization endpoint unless that endpoint is supplied as a displayed row of the same packet.*

Proof. Project the carrier of Definition 3919.1. The typing row G, Pi-candidate row Π, and neutral-normal row N are the only source evidence available to the adequacy budget.

The handoff rows are A and S. They expose the application-side candidate route and the named strong-normalization projections without replacing candidate evidence by an evidence-free theorem.

The boundary rows are F and O. They keep the normalization frontier, subject-reduction boundary, confluence boundary, and endpoint obligation budget visible to any closed-consistency consumer.

Finally, H, C, P, L transport, replay, source, and name the same finite route. Because the carrier lacks coordinates for global normalization, unrestricted subject reduction, confluence, semantic equality, quotient syntax equality, and host evaluation soundness, the listed rows exhaust the local NameCert surface. □

Theorem 3919.3 (MetaCIC Pi adequacy strong-normalization handoff)**.** *Every normalization-facing consumer of an accepted* $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ *packet factors through the route*

$$G, \Pi, N, A \longmapsto S \longmapsto F, O, H, C, P, L.$$

The consumer therefore reads closed context data, Pi-candidate evidence, neutral-normal evidence, application adequacy data, the strong-normalization handoff, frontier refusal, obligation budget, transport, replay, provenance, and local naming before citing the Pi adequacy contribution to candidate-mediated strong normalization.

Proof. Start with an accepted packet $B = (G, \Pi, N, A, S, F, O, H, C, P, L)$. The source rows G, Π, N, A provide the only candidate and neutral-normal evidence made public by the packet.

The strong-normalization handoff is S. It packages the displayed Pi-candidate, neutral-normal, and Pi-shaped SN projections as rows of the same finite carrier, so a consumer cannot cite them after dropping their evidence inputs.

The frontier and obligation rows F, O remain attached to the handoff. They record that subject-reduction, confluence, and global closed-normalization endpoints are not exported by the Pi adequacy budget alone.

The rows H, C, P, L preserve and name this route. Hence any normalization-facing consumer factors through the displayed candidate-mediated budget before it may use the Pi adequacy contribution. □

Falsifiable prediction: An accepted $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ normalization-facing read is refuted if it exports Pi-shaped strong-normalization data while omitting the displayed typing-frontier, Pi-candidate, neutral-normal, application-handoff, SN-handoff, frontier-refusal, obligation-budget, transport, replay, provenance, or local NameCert rows.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, `normalization_candidate_socket`, metacicnormalizationconfluencebridge: $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Pi-candidate evidence, neutral-normal evidence, application adequacy handoff, SN projection rows, frontier refusal, obligation-budget exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}$ packet is generated from closed context and typing-frontier data, Pi-candidate evidence, neutral-normal evidence, application adequacy rows, strong-normalization handoff rows, frontier refusal, obligation budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3919.1, Definition 3919.2, and Definition 3919.3 seed the Pi adequacy budget consumed by candidate-mediated strong-normalization routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICPiAdequacyBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, evidence-free closed-term strong normalization, unrestricted subject reduction, global confluence, semantic equality, quotient syntax equality, or host evaluation soundness.

Upgrade path: Obligation closure requires checked carrier admission, Pi-candidate evidence exactness, neutral-normal readback, application adequacy handoff, strong-normalization projection handoff, frontier refusal stability, obligation-budget exposure, and consumer nonescape for BEDC.Derived.MetaCICPiAdequacyBudgetUp.

$\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ consumes the Cauchy finite-window surface of chapter 3413 together with the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route displayed by its child chapter.

Chapter 3920

Finite Cauchy Diagonal NameCert Construction

Following Bishop and Bridges, the $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ packet records the finite diagonal selector needed when a displayed Cauchy family is read through regular rational windows and then sealed as a real name. It sits between the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the regular rational route of chapter 301, the finite stream windows of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet supplies only a finite selector surface for completion and Real consumers; it does not choose an infinite subsequence, a quotient representative, or a host limit.

Definition 3920.1 (Finite Cauchy diagonal carrier)**.** *A* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, W, K, S, R, Y, E, H, C, P, N)$$

with the following rows:

1. *F is the finite family row, naming the displayed Cauchy sources whose diagonal window is requested.*
2. *W is the selected-window classifier row, recording the finite window chosen for each displayed precision request and each displayed source row.*
3. *K is the classifier-stability row, comparing selected windows only by the same* hsame *and* Cont *routes used by the source family.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule exposed by the selected windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the diagonal comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, available only after F, W, K, S, R, Y have been displayed.*
8. *H records componentwise* hsame *transport for the family, window, stability, stream, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for diagonal-selection requests through $F, W, K, S, R, Y, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$ *row.*

The classifier reads only $F, W, K, S, R, Y, E, H, C, P, N$. There is no coordinate for countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3920.2 (Finite Cauchy diagonal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ *carrier $D = (F, W, K, S, R, Y, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *finite diagonal carrier admission through the displayed family row F;*
2. *selected-window classifier admission through W, with one displayed finite window for each displayed precision request and source row;*
3. *classifier stability through K, H, C, preserving the selected windows under componentwise* hsame *transport and displayed* Cont *replay;*
4. *diagonal extraction ledger admission through S, R, Y, requiring every extracted read to pass through the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows;*
5. $\mathsf{Real}^{\uparrow}$ *seal handoff through E, requiring the finite family, selected-window classifier, stability row, stream schedule, readback row, dyadic ledger, transport, replay, provenance, and local naming to remain visible to the terminal real-name consumer.*

No obligation row certifies countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, or ambient completeness.

Proof. Project the carrier of Definition 3920.1. The visible rows are exactly $F, W, K, S, R, Y, E, H, C, P, N$, so the finite family and selected-window classifier are the only routes by which a diagonal request enters the packet.

The stability row is read before any terminal seal. Transport by H and replay by C preserve the same family row, selected-window row, and classifier-stability row, so a replayed request cannot replace the finite selector by an infinite choice schedule or by host sequence equality.

The extraction ledger is then read through S, R, Y. The stream schedule supplies finite observations, the regular rational row repacks those observations, and the dyadic row records the tolerance budget for the same displayed windows.

Finally the Real handoff is downstream from the ledger. Since E is available only after $F, W, K, S, R, Y, H, C, P, N$ have been displayed, every $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives a finite diagonal selector surface rather than a hidden limit or quotient representative. □

Falsifiable prediction: An accepted $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ read is refuted if the Real-facing diagonal handoff reaches $\mathsf{Real}^{\uparrow}$ without displaying the finite family row, selected-window classifier, classifier-stability row, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming rows.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite family admission, selected-window classification, classifier stability, diagonal extraction, Real seal handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyDiagonal}^{\uparrow}$ packet is generated from a displayed finite Cauchy family, selected finite windows, classifier-stability rows, StreamName observations, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3920.1 and Definition 3920.2 seed the finite diagonal selector route over family, selected-window, stability, StreamName, RegSeqRat, DyadicRatCore, RealUp seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteCauchyDiagonal}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, a selected infinite diagonal sequence, quotient stream equality, host sequence equality, selected limits, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, selected-window classifier admission, classifier stability, diagonal extraction, RealUp seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyDiagonalUp.

The residual-envelope packet reads its conditional confluence boundary through chapter 3791, chapter 3827, and chapter 765 before the child file spells out the carrier.

Chapter 3921

MetaCIC Confluence Residual Envelope NameCert Construction

The MetaCICConfluenceResidualEnvelope$^{\uparrow}$ packet realizes the conditional residual-substitution boundary of chapter 9378 as finite MetaCIC NameCert data. It sits beside the substitution-boundary packet of chapter 3791, the residual checker of chapter 3827, the finite fuel trace of chapter 3889, the decidable neutral spine of chapter 3909, and the normalization evidence budget of chapter 3871. The envelope records which residual windows, substitution-boundary rows, checker rows, fuel traces, neutral-spine reads, and replay ledgers are visible before a consumer cites a conditional confluence handoff.

The residual envelope also consumes the closedness-preservation ledger of chapter 795 before any residual checker can be read as a confluence boundary.

Falsifiable prediction: A certification route to BetaParallelDiamondClosed that omits the residual-envelope row refutes MetaCICConfluenceResidualEnvelope$^{\uparrow}$, because the conditional substitution boundary has then been consumed without displaying the finite residual window, substitution-boundary, checker, fuel, neutral-spine, replay, provenance, and local NameCert rows.

Independence witness: metacic_fuel_trace, metacic_decidable_neutral_spine, metacic_confluence_substitut normalization_evidence_budget: none is bijective to MetaCICConfluenceResidualEnvelope$^{\uparrow}$ because the envelope jointly records residual windows, substitution-boundary exposure, checker status, bounded fuel, neutral-spine reads, normalization-budget non-escape, transport, replay, provenance, and local naming as one confluence-boundary packet.

Definition 3921.1 (MetaCIC confluence residual envelope carrier)**.** *A* MetaCICConfluenceResidualEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$$

with the following rows:

1. *T is the typed-window row naming the MetaCIC source and target windows under inspection.*
2. *R is the residual-frontier row, recording the beta-redex residues whose join is conditional on substitution compatibility.*
3. *S is the* MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$ *row exposed before any closed diamond handoff.*
4. *B is the bounded residual-checker row inherited from* MetaCICConfluenceResidualChecker$^{\uparrow}$*.*

5. *J is the local-join ledger for those residual windows.*
6. *O is the obstruction row retaining the named substitution-compatibility boundary.*
7. *F is the* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *bounded replay row.*
8. *U is the* $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ *neutral-spine read used to prevent atom and neutral cases from being promoted to beta branches.*
9. *Z is the* $\mathsf{NormalizationEvidenceBudget}^{\uparrow}$ *row, recording that normalization evidence may be inspected only as bounded finite evidence and not as global normalization.*
10. *H records componentwise* hsame *transport for typed windows, residual frontiers, substitution boundary, checker status, local joins, obstruction, fuel, neutral-spine, budget, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay for conditional confluence requests through $T, R, S, B, J, O, F, U, Z, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$, NormalizationEvidenc BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *row.*

The classifier reads only $T, R, S, B, J, O, F, U, Z, H, C, P, N$. It has no coordinate for an unconditional BetaParallelSubstitutionCompatibility *proof, an unconditional* BetaParallelDiamondClosed *proof, arbitrary closed Church–Rosser, erased substitution transport, or host syntax equality.*

Theorem 3921.2 (Residual envelope NameCert surface)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission, typed-window exposure, residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, obstruction retention, fuel-trace replay, neutral-spine separation, normalization-budget non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3921.1. The only visible route to a conditional confluence read is the displayed tuple $T, R, S, B, J, O, F, U, Z, H, C, P, N$.

The residual and substitution-sensitive rows R, S, B, J, O are read before any closed-diamond handoff. Thus a consumer can see the residual frontier, the named substitution boundary, the bounded checker status, the local join ledger, and the obstruction row as finite data rather than as a hidden global proof.

The supporting rows F, U, Z then separate bounded computation, neutral-spine cases, and normalization evidence from the residual-substitution boundary. They keep fuel traces, neutral reads, and normalization budgets visible without converting them into global normalization or confluence.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no remaining coordinate contains an unconditional substitution-compatibility or closed-diamond proof, the packet certifies only the residual-envelope boundary and not the endpoint theorem. □

Theorem 3921.3 (MetaCIC confluence residual critical-pair audit surface)**.** *For an accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet with two displayed one-step residual branches, the finite critical-pair audit surface consists only of the source term row, the two residual branch rows, the shared candidate mediator row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$ *row. The audit surface does not*

assert global confluence, strong normalization, decidability of all residual joins, or host quotient conversion.

Proof. Project the carrier of Definition 3921.1. The typed-window row T supplies the source term row, while R, B, J, O display the residual frontier, bounded checker status, local join ledger, and obstruction boundary from which the two one-step branches and the candidate mediator are read.

Apply the residual envelope NameCert surface theorem. It requires residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, obstruction retention, bounded fuel replay, neutral-spine separation, normalization-budget non-escape, transport, replay, provenance, and local naming before any confluence-facing consumer reads the envelope.

Compare the two displayed branch rows only through the candidate mediator row retained in J and the obstruction boundary retained in O. The transport row H and replay row C preserve those displayed rows componentwise; they do not add a new conversion relation or quotient syntax equality.

Carrier-field exhaustion gives the non-escape claim. Since the accepted packet has no coordinate for global confluence, strong normalization, full join decidability, or host quotient conversion, the critical-pair read is only the finite audit surface stated above. □

Theorem 3921.4 (MetaCIC confluence residual envelope checker-status determinacy)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier* $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$*, any confluence consumer that reads the bounded checker status* B *must read exactly one of the displayed accepted, rejected, or obstruction alternatives supplied by the residual-checker route. The later rows* $J, O, F, U, Z, H, C, P, N$ *cannot change that status and cannot turn it into unconditional* BetaParallelDiamondClosed*.*

Proof. Project the carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$ of Definition 3921.1. The checker status is the single row B, inherited from the residual-checker route of chapter 3827.

The residual-checker route separates accepted, rejected, and obstruction status as displayed finite alternatives. Reading B therefore exposes one of those alternatives before the envelope rows $J, O, F, U, Z, H, C, P, N$ are inspected.

The obstruction row O retains the named residual-substitution obstruction rather than erasing it. The rows F, U, Z separate bounded computation, neutral-spine reads, and normalization budgets from the checker status, so they cannot promote a bounded status read into an endpoint confluence theorem.

Finally H, C, P, N transport, replay, source, and name only the displayed carrier rows. Because no later row contains an unconditional BetaParallelDiamondClosed coordinate, the checker status remains the one supplied by B. □

Theorem 3921.5 (MetaCIC residual envelope obstruction monotonicity)**.** *For every accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *carrier* $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$*, if the bounded checker row* B *exposes obstruction status and the row* O *retains the residual-substitution obstruction, then every read of* J, F, U, Z, H, C, P, N *preserves that obstruction boundary. In particular, those later rows cannot erase the obstruction status and cannot export an unconditional* BetaParallelDiamondClosed *theorem.*

Proof. Project B and O from the carrier of Definition 3921.1. The row B supplies the displayed checker status, and O is the only row that records the residual-substitution obstruction retained by the envelope.

Apply checker-status determinacy, Definition 3921.4. Once B exposes obstruction status, the rows inspected after B cannot replace that status by an accepted or rejected alternative.

Now inspect the remaining carrier inventory. The rows J, F, U, Z contain local joins, bounded fuel, neutral-spine reads, and normalization-budget evidence, while H, C, P, N contain transport, replay, provenance, and local naming data. None of these rows contains a closed-diamond coordinate.

Therefore the obstruction retained by O remains visible through every later read. Since the accepted carrier has no unconditional BetaParallelDiamondClosed coordinate, no consumer can obtain that endpoint theorem from J, F, U, Z, H, C, P, N. □

Theorem 3921.6 (MetaCIC confluence residual envelope neutral-spine separation)**.** *Any conditional confluence consumer that attempts to promote a neutral-spine read U into a beta residual branch must also display the residual-frontier row R, substitution-boundary row S, checker row B, local join row J, fuel trace F, and normalization budget Z. Without those rows, the accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet exports only a neutral case and no beta branch.*

Proof. Project the carrier $E = (T, R, S, B, J, O, F, U, Z, H, C, P, N)$ of Definition 3921.1. The neutral-spine row U is a displayed case-separation row, not a residual-frontier row.

Apply the NameCert surface theorem Definition 3921.2. It lists residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join ledger exposure, fuel-trace replay, neutral-spine separation, and normalization-budget non-escape as distinct obligations of the accepted packet.

Therefore a consumer that reads only U has only the neutral case. To promote the read into a beta residual branch, the consumer must also display R, S, B, J, F, Z through the same transport, replay, provenance, and local naming rows. If any of these rows is absent, the carrier has no coordinate from which a beta residual branch can be exported. □

Theorem 3921.7 (MetaCIC residual envelope closed-diamond consumption boundary)**.** *Any consumer attempting to cite* BetaParallelDiamondClosed *through an accepted* $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$ *packet must first display the full row inventory*

$$T, R, S, B, J, O, F, U, Z, H, C, P, N.$$

If the substitution-boundary row S, checker row B, obstruction row O, fuel row F, or normalization-budget row Z is absent, the packet exports only a conditional residual envelope and not closed confluence.

Proof. Start with the NameCert surface theorem, Definition 3921.2. It identifies the accepted packet with the displayed rows $T, R, S, B, J, O, F, U, Z, H, C, P, N$ and says that no unconditional closed-diamond coordinate is part of the carrier.

Next read the checker and obstruction rows. The checker-status determinacy theorem fixes the status exposed by B, while Definition 3921.5 preserves any obstruction retained in O through the later local-join, fuel, neutral-spine, budget, transport, replay, provenance, and naming rows.

Finally apply neutral-spine separation. Theorem 3921.6 refuses to promote a neutral-spine read into a beta residual branch unless the residual frontier, substitution boundary, checker, local join, fuel trace, and normalization budget are all displayed.

Thus a consumer that omits S, B, O, F, or Z has not consumed the finite residual-substitution boundary required for the endpoint. It can read only the conditional residual envelope carried by the displayed rows, and no route to BetaParallelDiamondClosed is exported. □

Theorem 3921.8 (MetaCIC residual envelope diamond route)**.** *Every* MetaCICConfluenceResidualEnvelope$^{\uparrow}$ *read that exposes a closed-diamond consumer row factors first through the residual-substitution boundary of chapter 2181, the bounded residual-substitution frontier of chapter 3808, and the MetaCIC open-frontier ledger of chapter 762. The residual envelope may then expose the closed-diamond consumer only through the displayed rows* $T, R, S, B, J, O, F, U, Z, H, C, P, N$*, and it exports no direct route to* BetaParallelDiamondClosed*.*

Proof. Start from the closed-diamond consumption boundary, Definition 3921.7. It requires a consumer to display the full row inventory $T, R, S, B, J, O, F, U, Z, H, C, P, N$ before the closed-diamond endpoint can be cited.

The substitution-sensitive part of that inventory is not local to the envelope alone. The residual-substitution boundary keeps the substitution premise visible, the bounded residual-substitution frontier bounds the frontier read, and the open-frontier ledger records the confluence endpoint as a frontier entry rather than as a discharged theorem.

Reading S, B, J, O through those siblings fixes the route before the consumer can inspect F, U, Z, H, C, P, N. The fuel, neutral-spine, budget, transport, replay, provenance, and naming rows preserve the same route and do not add a closed-diamond coordinate.

Therefore the closed-diamond consumer row, when present, is exposed only after the three sibling packets and the envelope inventory have been read. No direct route to BetaParallelDiamondClosed is available from the residual envelope alone. □

Theorem 3921.9 (MetaCIC residual envelope closed-join boundary)**.** *Every closed-join or closed-diamond consumer of an accepted* MetaCICConfluenceResidualEnvelope$^{\uparrow}$ *packet* $E = (T, R, S, B, J, O, F, U, Z, H, C, P$ *must read the envelope rows in the following boundary order: typed window, residual frontier, substitution boundary, checker status, local join, obstruction, fuel, neutral spine, normalization budget, transport, replay, provenance, and local naming. Equivalently, a consumer may cite the closed-join row only after the displayed route*

$$T, R, S, B, J, O, F, U, Z, H, C, P, N$$

has been consumed. The boundary exports no unrestricted BetaParallelDiamondClosed *theorem, no unrestricted substitution compatibility, no subject reduction theorem, and no Church–Rosser theorem.*

Proof. Begin with the NameCert surface theorem, Definition 3921.2. It fixes the visible surface of an accepted envelope as precisely $T, R, S, B, J, O, F, U, Z, H, C, P, N$.

Next read the residual-sensitive part of the route. The checker-status determinacy theorem fixes B before the local join is inspected, while obstruction monotonicity keeps O visible through the later rows. Thus J can be read only as a local join under the already displayed residual frontier, substitution boundary, checker status, and obstruction boundary.

Apply the closed-diamond consumption boundary and the diamond route. Together they force any closed-diamond or closed-join consumer through the same full inventory, and the sibling residual-substitution boundary, bounded frontier, and open-frontier ledger prevent the envelope from presenting the endpoint as a discharged global theorem.

Finally inspect the excluded coordinates. Since the carrier has no row for unrestricted closed diamond, substitution compatibility, subject reduction, or Church–Rosser, the closed-join read is exactly the ordered finite boundary stated above. □

The Lean TasteGate site for MetaCICConfluenceResidualEnvelope$^{\uparrow}$ must provide explicit Nontrivial and FieldFaithful instances for this carrier surface. These instances must keep the residual windows, substitution boundary, checker status, local joins, obstruction row, fuel trace, neutral spine, normalization budget, transport, replay, provenance, and local naming distinguishable.

Closure status for NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$.

Constructive story: A MetaCICConfluenceResidualEnvelope$^{\uparrow}$ packet is generated from typed windows, residual frontiers, a substitution-boundary row, bounded checker status, local joins, obstruction retention, bounded fuel traces, neutral-spine reads, normalization-budget rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 3921.1 and Definition 3921.2 seed the residual-substitution confluence envelope over typed-window, residual-frontier, substitution-boundary, checker, local-join, obstruction, fuel, neutral-spine, budget, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConfluenceResidualEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim BetaParallelSubstitutionCompatibility, BetaParallelDiamondClosed, arbitrary closed Church–Rosser, global normalization, Lean verification, bridge checking, erased substitution transport, or host syntax equality.

Upgrade path: Obligation closure requires checked carrier admission, residual-frontier exposure, substitution-boundary exposure, bounded-checker status, local-join exposure, obstruction retention, fuel replay, neutral-spine separation, normalization-budget non-escape, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConfluenceResidualEnvelopeUp.

Chapter 3922

A Concrete Naming Certificate for $\mathsf{HahnBanach}^{\uparrow}$

Following Hahn and Banach, Bourbaki, and Bishop-style constructive functional analysis, the $\mathsf{HahnBanach}^{\uparrow}$ packet records the finite BEDC surface for bounded scalar functional extension rows. It sits beside the normed-space and Banach rows of chapter 1950 and chapter 90, the dual carrier of chapter 2265, the bounded-operator row of chapter 1644, the functional-analysis boundary of chapter 94, and the real-name seal of chapter 38. The chapter names a displayed finite extension ledger only; it is not the unrestricted classical Hahn–Banach theorem.

Definition 3922.1 (Hahn–Banach extension carrier)**.** *A* $\mathsf{HahnBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, B, D, F, E, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{NormedSpace}^{\uparrow}$ *or* $\mathsf{Banach}^{\uparrow}$ *row on which the extension is read.*
2. *B is the displayed subspace row, including the finite incidence data saying which inspected vectors lie in the subspace.*
3. *D is the* $\mathsf{BanachDual}^{\uparrow}$ *and* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for the bounded scalar functional on B.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *context row that records the local functional-analysis consumer boundary.*
5. *E is the extension row, displaying the scalar functional on the inspected ambient vectors and its agreement with D on the subspace row.*
6. *L is the norm-bound ledger, comparing the displayed bound for D with the displayed bound for E through* $\mathsf{Real}^{\uparrow}$ *endpoints.*
7. *H records componentwise* hsame *transport for $X, B, D, F, E, L, C, P, N$.*
8. *C records displayed* Cont *replay from ambient-space reads through the subspace, bounded-functional, extension, and norm-bound rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HahnBanach}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, Zorn maximality, or unrestricted ambient extension.

Lean4 ✓ `BEDC.Derived.HahnBanachUp.HahnBanachTasteGate single carrier alignment`

Theorem 3922.2 (Hahn–Banach NameCert obligations)**.** *For every accepted* $\mathsf{HahnBanach}^{\uparrow}$ *carrier* $K = (X, B, D, F, E, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$ *surface consists exactly of ambient normed or Banach admission, displayed subspace admission, bounded scalar functional admission, local functional-analysis context, extension-row agreement, norm-bound ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No extension obligation is stored outside those rows.*

Proof. Project the accepted carrier of Definition 3922.1. The rows $X, B, D, F, E, L, H, C, P, N$ are the full public inventory.

The bounded-functional read starts at the subspace row B and the scalar-functional row D. The extension row E can be inspected only with the same ambient row X and the same subspace incidence data.

The norm-bound ledger is L. It compares the displayed bound for D with the displayed bound for E through $\mathsf{Real}^{\uparrow}$ endpoints, while H, C, P, N only transport, replay, source, and name the same route.

Thus the NameCert obligations are exactly the listed finite rows. The unrestricted theorem, quotient duals, selected extensions, ultrafilter arguments, and choice principles have no carrier coordinate. □

Theorem 3922.3 (Hahn–Banach bounded functional extension route)**.** *Every bounded scalar functional extension read of an accepted* $\mathsf{HahnBanach}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{NormedSpace}^{\uparrow},\ \mathsf{Banach}^{\uparrow},\ \mathsf{BanachDual}^{\uparrow},\ \mathsf{BoundedLinearOperator}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route first fixes the ambient and subspace rows, then reads the bounded scalar functional, then reads the displayed extension row and the norm-bound ledger, and only then exposes transport, replay, provenance, and local naming. It exports no unrestricted classical Hahn–Banach theorem, quotient dual space, selected extension, ultrafilter argument, countable choice, or detached extension principle.

Proof. Let $K = (X, B, D, F, E, L, H, C, P, N)$ be accepted. The carrier projection gives the ambient row X, the subspace row B, the bounded scalar functional row D, the functional-analysis context row F, and the extension row E.

The public extension route reads X and B before D. Only then can the consumer inspect E, because agreement with the original functional is meaningful only over the displayed subspace incidence row.

The norm-bound ledger L is read after the two functional rows. It records the displayed $\mathsf{Real}^{\uparrow}$ comparison between the original bound and the extension bound, so the consumer receives a finite bound ledger rather than an ambient domination theorem.

Finally H, C, P, N preserve this finite route. Since no row names quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, or unrestricted Hahn–Banach extension, none of those objects can be exported by the accepted packet. □

Falsifiable prediction: Dropping the norm-bound ledger L makes Definition 3922.3 false: a consumer could no longer distinguish a bounded extension row from an unbounded displayed scalar assignment.

Independence witness: normedspace, banach, banachdual, boundedlinearoperator, functional-analysis, real: $\mathsf{HahnBanach}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries ambient space, subspace incidence, bounded scalar functional, extension agreement, norm-bound ledger, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{HahnBanach}^{\uparrow}$ packet is generated from an ambient NormedSpaceUp or BanachUp row, a displayed subspace row, a BanachDualUp and BoundedLinearOperatorUp bounded scalar functional row, a FunctionalAnalysisUp context row, an extension row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$, seed)

Scope closed: Definition 3922.1, Definition 3922.2, and Definition 3922.3 seed the finite bounded scalar functional extension packet over normed-space, Banach, dual, bounded-operator, functional-analysis, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HahnBanach}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.HahnBanachUp.HahnBanachTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim the unrestricted classical Hahn–Banach theorem, quotient dual spaces, selected extensions, ultrafilter arguments, countable choice, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subspace incidence, bounded scalar functional transport, extension agreement, norm-bound exactness, non-escape, replay, provenance, and local naming for BEDC.Derived.HahnBanachUp.

Chapter 3923

A Concrete Naming Certificate for $\mathsf{RealIntervalCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealIntervalCompletion}^{\uparrow}$ packet records the finite completion-facing boundary at which located interval data, nested refinements, regular rational readbacks, and Real seals are read as one interval-completion carrier. It sits beside the located endpoint surface of chapter 394, the nested interval carrier of chapter 320, and the public completion reflector route of chapter 1585. The packet names the L10 Real and Completion handoff without importing Dedekind cuts, quotient completions, selected points, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3923.1 (RealIntervalCompletion carrier)**.** *A* $\mathsf{RealIntervalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, N, D, W, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row, carrying lower and upper located endpoint windows and their visible interval-order classifier.*
2. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row, recording the displayed nesting discipline and the dependency of each later interval on the earlier endpoint rows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, carrying the dyadic shrinkage scale used by the refinement windows.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes only finite interval windows before any regular rational readback is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, reading the displayed interval windows as regular rational approximants.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the completed interval-facing read.*
7. *H records componentwise* hsame *transport for L, N, D, W, R, E and the structural rows.*
8. *C records displayed* Cont *replay from located endpoints through nested refinements, dyadic shrinkage, stream windows, regular rational readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalCompletion}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *Q is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}$ *row.*
The carrier has no coordinate for Dedekind cuts, quotient interval completions, selected interval points, open-cover compactness, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalCompletionUp.RealIntervalCompletionTasteGate single carrier alignment`

Theorem 3923.2 (RealIntervalCompletion NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalCompletion}^{\uparrow}$ *carrier* $I = (L, N, D, W, R, E, H, C, P, Q)$*, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located endpoint, nested refinement, dyadic radius, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-refinement classification, requiring every refinement read to pass through the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row and the* $\mathsf{NestedInterval}^{\uparrow}$ *nesting row before any completion-facing seal is visible;*
3. *diameter-shrinkage stability, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radii to decrease along the displayed nested interval route and to remain stable under the classifier;*
4. *Real seal compatibility, requiring the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to reach the* $\mathsf{Real}^{\uparrow}$ *seal only through the same finite interval route;*
5. *refusal of quotient intervals, selected limit points, countable-choice schedules, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is the projection of Definition 3923.1. The displayed tuple lists the located endpoints, nested refinements, dyadic radii, finite windows, regular rational readbacks, Real seal, transport, replay, provenance, and local naming row.

The interval-refinement classifier is the route L, N. A consumer must read located endpoint data before it can inspect the nested refinement row, so the packet cannot replace the route by a host interval completion or by selected endpoint representatives.

Diameter-shrinkage stability is carried by D together with the nesting row N. The dyadic radii are read as finite $\mathsf{DyadicRatCore}^{\uparrow}$ data, and componentwise hsame transport in H preserves the same shrinkage classifier.

Real seal compatibility is the route W, R, E. The stream windows feed the regular rational readback before the Real seal is exposed, and Cont replay in C preserves this order for downstream completion consumers.

Finally, P and Q name exactly the listed rows. Since the carrier contains no Dedekind-cut, quotient-interval, selected-point, choice, open-cover compactness, or ambient-completeness coordinate, those objects cannot be exported from an accepted packet. □

Theorem 3923.3 (RealIntervalCompletion completion handoff)**.** *Every* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RealIntervalCompletion}^{\uparrow}$ *carrier receives the finite route*

$$L, N, D, W, R, E, H, C, P, Q$$

and no independent completion object. The completion-facing read factors through located interval endpoints, nested interval refinement, dyadic diameter shrinkage, finite stream windows, regular rational readback, and the Real seal.

Proof. Start with an accepted consumer read. Carrier admission exposes only the rows $L, N, D, W, R, E, H, C, P, Q$.

The interval part is L, N, D. It fixes located endpoints, the displayed refinement order, and the dyadic shrinkage scale before any Real-facing data are available.

The completion-facing part is W, R, E. It reads finite windows, regular rational approximants, and the Real seal along the same route used by the public $\mathsf{CauchyCompletionReflector}^{\uparrow}$ surface.

Transport, replay, provenance, and local naming are the structural rows H, C, P, Q. They preserve the route but add no quotient completion, selected point, countable-choice schedule, open-cover compactness, or ambient Real completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3923.1, Definition 3923.2, and Definition 3923.3 seed the interval-completion carrier over located endpoint rows, nested refinements, dyadic shrinkage, finite windows, regular rational readback, Real seals, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind cuts, quotient completions, selected points, open-cover compactness, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement classification, diameter-shrinkage stability, Real seal compatibility, and non-escape for BEDC.Derived.RealIntervalCompletionUp.

Constructive story: A $\mathsf{RealIntervalCompletion}^{\uparrow}$ packet is generated from LocatedInterval endpoint rows, NestedInterval refinement rows, DyadicRatCore radii, StreamName windows, RegSeqRat readbacks, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3924

A Concrete Naming Certificate for $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$

Following Bishop and Bridges constructive compact metric analysis, the $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ packet records the finite theorem-facing route by which a continuous image of compact metric data becomes totally bounded. It consumes the compact finite-net surface of chapter 60, the continuous-map graph and modulus rows of chapter 130, the source and target metric-distance rows, the uniform-modulus ledger, and the rational lower-bound fold. The packet is a concrete image-total-bounded carrier, not a semantic compact-image theorem in an ambient topology.

Definition 3924.1 (Compact continuous image total-bounded carrier)**.** *A* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, U, M, R, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric finite-net packet, exposing source centers and coverage rows through* $\mathsf{CompactMetric}^{\uparrow}$ *and the compact carrier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus and graph packet from the source metric carrier to the target metric carrier.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger read from the compact finite-net and continuous-map route.*
4. *M is the source and target* $\mathsf{Met}^{\uparrow}$ *distance row pair, carrying only displayed distance witnesses.*
5. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row used to select finite image radii from the displayed modulus and net data.*
6. *B is the* $\mathsf{ProbeBundle}$ *image-center row obtained by applying the displayed graph rows of F to the finite source-center bundle of X.*
7. *H records componentwise* hsame *transport for X, F, U, M, R, B and the local structural rows.*
8. *C records displayed* Cont *replay from compact source nets through continuous-map modulus rows, target distance rows, rational folds, and image-center bundle rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}}$ *row.*

The carrier contains no open-cover compactness object, no selected subcover from an arbitrary cover, no host image set, no quotient function equality, no choice-selected center, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CompactContinuousImageTotalBoundedUp.TasteGate.CompactContinuousImageTotalBounded`
`single carrier alignment`

Theorem 3924.2 (CompactContinuousImageTotalBounded NameCert obligations)**.** *For every accepted* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier* $K = (X, F, U, M, R, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for compact metric finite-net data, continuous-map rows, uniform-modulus ledger, metric-distance rows, rational lower-bound fold, image-center bundle, transport, replay, provenance, and local naming;*
2. *finite-net image classification, requiring every image center to arise from a bundled compact source center and a displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row;*
3. *modulus compatibility, requiring the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger to use the same compact source, continuous-map, metric-distance, and* $\mathsf{Rat}^{\uparrow}$ *fold rows;*
4. *total-bounded export, requiring every target-image coverage read to factor through the finite* $\mathsf{ProbeBundle}$ *image-center row* B*;*
5. *refusal of open-cover compactness, arbitrary selected subcovers, host image sets, quotient function equality, choice-selected centers, and ambient completeness.*

Proof. Carrier admission is projection from Definition 3924.1. The tuple displays the compact source net, continuous graph and modulus data, uniform-modulus ledger, metric rows, rational fold, image centers, transport, replay, provenance, and local naming.

Finite-net image classification is the route X, F, B. Each image center is read by applying the displayed continuous-map graph row to a center already present in the compact source bundle, so no host image set or arbitrary center selector is introduced.

Modulus compatibility is the route X, F, U, M, R. The uniform-modulus ledger is obtained from the compact finite-net data, the continuous-map pointwise rows, the source and target metric distances, and the rational lower-bound fold over $\mathsf{Rat}^{\uparrow}$ histories.

The total-bounded export is B with its support rows. A target-image coverage read must pass through the finite $\mathsf{ProbeBundle}$ image-center row and the displayed target metric row before it reaches any consumer.

Finally, H, C, P, N preserve and name exactly this route. Since the carrier has no open-cover compactness object, arbitrary subcover selector, host image-set coordinate, quotient function equality, choice-selected center, or completeness coordinate, those objects cannot be exported. □

Theorem 3924.3 (Compact continuous image total-bounded handoff)**.** *Every total-boundedness consumer of an accepted* $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ *carrier receives only the compact finite-net source row, the continuous-map graph and modulus rows, the uniform-modulus ledger, source and target metric-distance rows, the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold, the finite* $\mathsf{ProbeBundle}$ *image centers, and the visible structural rows*

$$X, F, U, M, R, B, H, C, P, N.$$

The handoff is therefore a finite BEDC certificate for image total-boundedness, not an open-cover compactness theorem and not a host image construction.

Proof. Begin with the accepted carrier and project its rows. The only source compactness data available are the finite-net rows in X.

Read the continuous map through F and the uniform modulus through U. Both rows are tied to the same source and target metric-distance rows M, so the image read cannot change its metric carrier mid-route.

The rational lower-bound fold R and the image-center bundle B produce the target finite-net witness. The fold is a finite $\mathsf{Rat}^{\uparrow}$ history calculation, and the image centers are a finite ProbeBundle list, so no arbitrary subcover or choice-selected image point is exposed.

The structural rows H, C, P, N transport, replay, prove provenance, and name that finite route. The excluded open-cover, host-image, quotient-function, choice, and completeness objects are absent from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$, seed)

Scope closed: Definition 3924.1, Definition 3924.2, and Definition 3924.3 seed the compact-continuous-image total-bounded carrier over compact finite nets, continuous-map modulus rows, uniform-modulus ledgers, metric distances, RatUp folds, ProbeBundle image centers, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactContinuousImageTotalBounded}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary subcovers, host image sets, quotient function equality, choice-selected centers, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net image classification, modulus compatibility, total-bounded export, and non-escape for BEDC.Derived.CompactContinuousImageTotalBoundedUp.

Constructive story: A $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap pointwise-modulus rows, UniformModulus ledgers, source and target MetricSpace distance rows, RatUp lower-bound folds, ProbeBundle image centers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3925

A Concrete Naming Certificate for $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$

The $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ packet realizes the boundary-weighted disk-read branch of chapter 9463 as finite NameCert data. It sits beside the hyperbolic metric surface of chapter 3474, the boundary-shadow route of chapter 3813, the visual metric packet of chapter 3866, the unit-disk carrier of chapter 377, the circle carrier of chapter 58, the integral row of chapter 123, and the real-name seal of chapter 38. The packet names a finite boundary-kernel read, not a completed harmonic boundary theory.

Definition 3925.1 (Hyperbolic Poisson-kernel carrier)**.** *A* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-source row for the interior point being read.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric row attached to the same disk source.*
3. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *boundary-shadow row that exposes the visible boundary window.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visual-boundary comparison row.*
5. *U is the* $\mathsf{S}^{1\uparrow}$ *boundary-circle row over which angular samples are displayed.*
6. *S is the* $\mathsf{Integral}^{\uparrow}$ *boundary-integration row for the finite sample or displayed quadrature route.*
7. *I is the* $\mathsf{Real}^{\uparrow}$ *scalar endpoint row for kernel weights and normalization comparisons.*
8. *R is the radial denominator and numerator ledger tying D, M, B, U, I to the displayed hyperbolic Poisson weight.*
9. *W is the weighted boundary-integral readback row, exposing only the value determined by S, R, I over the displayed disk and boundary rows.*
10. *H records componentwise* hsame *transport for $D, M, B, V, U, S, I, R, W, C, P, N$.*
11. *C records displayed* Cont *replay from disk source through metric, boundary shadow, visual boundary, circle, integral, weight, and readback rows.*

12. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for completed boundary functions, unrestricted harmonic extension, selected radial limits, quotient boundary equality, ultrafilter convergence, or global boundary compactification.

Lean4 ✓ `BEDC.Derived.HyperbolicPoissonKernelUp.HyperbolicPoissonKernelTasteGate single carrier alignment`

Theorem 3925.2 (Hyperbolic Poisson-kernel NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier* $K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$ *surface consists exactly of disk-source admission, hyperbolic metric admission, boundary-shadow admission, visual-boundary comparison, circle-boundary admission, boundary-integral admission, Real endpoint admission, radial weight-ledger exactness, weighted readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3925.1. The disk, metric, boundary-shadow, visual-boundary, circle, integral, Real, weight, readback, transport, replay, provenance, and local naming rows are the full public inventory.

The geometric rows are read first. The disk row D fixes the interior source, M fixes the hyperbolic metric comparison, B fixes the visible boundary shadow, and V fixes the visual-boundary comparison.

The boundary-kernel rows are then read through U, S, I, R, W. The circle row supplies boundary samples, the integral row supplies the displayed boundary aggregation, the Real row supplies scalar endpoints, the weight ledger supplies the hyperbolic Poisson factor, and the readback row exposes only the value determined by those rows.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further row is present, completed boundary functions, unrestricted harmonic extension, selected limits, quotient boundary equality, ultrafilter convergence, and global compactification are outside the packet. □

Theorem 3925.3 (Hyperbolic Poisson-kernel boundary integral nonescape)**.** *Every accepted boundary-integral read of a* $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicBoundaryShadow}^{\uparrow},\ \mathsf{HyperbolicVisualMetric}^{\uparrow},\ \mathsf{S}^{1\uparrow},\ \mathsf{Integral}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route first fixes the disk, metric, boundary-shadow, and visual-boundary rows, then reads the circle and integral rows, then reads the Real endpoint and weight ledger rows, and only then exposes weighted boundary readback. It exports no completed boundary function, unrestricted harmonic extension, selected radial limit, quotient boundary equality, ultrafilter convergence, or global compactification.

Proof. Let $K = (D, M, B, V, U, S, I, R, W, H, C, P, N)$ be accepted. A boundary-integral consumer first receives the geometric route D, M, B, V, so the boundary window is the one displayed by the disk, metric, shadow, and visual rows.

The integration route is U, S. The circle row supplies the boundary parameter window, and the integral row supplies only the displayed finite or certified aggregation over that window.

The scalar kernel route is I, R, W. The Real row supplies the endpoints, the weight ledger ties those endpoints to the disk and boundary rows, and the readback row exposes the weighted value only after the integral and weight ledgers are fixed.

Transport, replay, provenance, and local naming through H, C, P, N preserve this route. The excluded analytic and choice-theoretic objects have no carrier coordinate, so the boundary-integral read cannot escape the finite packet. □

Falsifiable prediction: Removing either the $\mathsf{HyperbolicMetric}^{\uparrow}$ row M or the $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ row B makes Definition 3925.3 false: a consumer could no longer verify that the boundary weight belongs to the displayed hyperbolic disk route.

Independence witness: hyperbolicmetric, hyperbolic-boundary-shadow, unitdisk, sone, integral, real: $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk source, hyperbolic metric, boundary shadow, visual comparison, circle boundary, integral aggregation, Real endpoints, weight ledger, readback, transport, replay, provenance, and local naming rows.

The Lean TasteGate site for $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ HyperbolicPoissonKernel}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicPoissonKernel}^{\uparrow}$ instances. These instances keep the disk, metric, boundary, circle, integral, Real, weight, readback, transport, replay, provenance, and naming rows distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$ packet is generated from UnitDiskUp disk data, HyperbolicMetricUp metric rows, HyperbolicBoundaryShadowUp and HyperbolicVisualMetricUp boundary rows, SOneUp circle rows, IntegralUp boundary aggregation, RealUp endpoints, a hyperbolic weight ledger, weighted readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3925.1, Definition 3925.2, and Definition 3925.3 seed the finite hyperbolic boundary-kernel route over disk, metric, boundary, circle, integral, Real, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPoissonKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed boundary functions, unrestricted harmonic extension, selected radial limits, quotient boundary equality, ultrafilter convergence, global boundary compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, geometric boundary routing, circle and integral admission, weight-ledger exactness, readback nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicPoissonKernelUp.

Chapter 3926

A Concrete Naming Certificate for $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$

Following Ekeland's variational principle and the Bishop-style use of complete-metric descent, the $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ packet names the finite descent ledger consumed by variational and fixed-point arguments. It sits downstream of the metric surface of chapter 57, the complete-metric carrier of chapter 128, the lower-semicontinuity packet of chapter 1536, the regular-rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet records a finite variational descent certificate; it is not a host minimizer, not a selected choice function, and not an ambient completeness theorem.

Definition 3926.1 (Ekeland variational-principle carrier)**.** *An* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, d, M, L, R, A, T, W, H, C, P, N)$$

with the following displayed rows:

1. *X is the finite metric-domain row read through* $\mathsf{Met}^{\uparrow}$ *addresses.*
2. *d is the metric-comparison row naming the inspected distance witnesses on X.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *descent-readiness row, exposing only the finite Cauchy-tail obligations needed by the packet.*
4. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *value-stability row for the inspected functional.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row used to compare displayed values and descent budgets.*
6. *A is the approximation row naming the initial value, positive tolerance, and admissible descent budget.*
7. *T is the finite transfinite-free descent trace, ordered by the displayed metric and value-readback rows.*
8. *W is the winner-window row naming the final inspected point and the finite perturbation inequality visible at that point.*
9. *H records componentwise* hsame *transport for metric, completeness, lower-semicontinuity, rational readback, approximation, trace, and winner-window rows.*
10. *C records displayed* Cont *replay from a variational request through X, d, M, L, R, A, T, W.*

11. *P is the* Pkg *provenance over* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$ *row.*

The carrier exports exactly this finite descent packet. It contains no host-level minimizer, no global choice selector, no quotient of Cauchy approximants, and no independent assertion that the ambient metric space is complete.

Lean4 ✓ `BEDC.Derived.EkelandVariationalPrincipleUp.EkelandVariationalPrincipleTasteGate single carrier alignment`

Theorem 3926.2 (Ekeland variational-principle NameCert obligations)**.** *For every accepted* EkelandVariationalPrin *carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed metric-domain, distance, completeness, lower-semicontinuity, rational-readback, approximation, descent-trace, winner-window, transport, continuation, provenance, and naming rows;*
2. *metric readability, requiring every distance comparison used by the descent trace to be one of the displayed d rows on addresses in X;*
3. *complete-metric descent readiness, requiring every Cauchy-tail request made by the trace to pass through M;*
4. *lower-semicontinuity readability, requiring every value-stability read to pass through L and its regular-rational row R;*
5. *descent-budget coverage, requiring the initial value, tolerance, and perturbation budget to be named in A before the trace row T is consumed;*
6. *winner-window exposure, requiring the terminal inspected point and its finite perturbation inequality to be named in W;*
7. *transport, replay, provenance, and naming stability through H, C, P, N;*
8. *ledger non-escape, excluding host minimizers, global choice selectors, quotient-completion objects, and ambient completeness theorems.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier row by row. The metric-domain and distance rows expose the inspected addresses and comparisons, while M, L, R, A, T, W expose the complete-metric descent readiness, lower-semicontinuity reads, rational comparisons, approximation budget, finite trace, and terminal winner window.

The obligation order follows the displayed route. A descent comparison cannot be read before d and R are visible, a Cauchy-tail request cannot be read before M is visible, and the perturbation inequality cannot be read before the approximation row and trace row have been consumed.

Transport and replay are structural rows. They preserve the same metric, value, budget, trace, and winner-window data under hsame comparison and displayed Cont replay.

Finally, P and N source and name only the displayed packet. Since no coordinate stores a host minimizer, a global choice selector, a quotient-completion object, or an independent completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 3926.3 (Ekeland metric-descent handoff)**.** *Every variational, complete-metric, or fixed-point consumer reading an accepted* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *packet factors through the finite descent route*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto A, T, W, H, C, P, N.$$

The consumer therefore receives a certificate-local descent ledger and terminal perturbation window, not a host minimizer, selected choice function, quotient index, or ambient completeness theorem.

Proof. Start from an accepted carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$. The metric rows X, d supply the only addresses and distance comparisons available to the consumer.

The complete-metric and lower-semicontinuity rows then bind the analytic reads. The row M supplies the finite Cauchy-tail obligations used by descent, while L and R supply the value-stability and rational-readback route.

The approximation and trace rows A, T order the descent budget before the winner-window row W is consumed. Thus the terminal perturbation inequality is a displayed row of the packet, not a selected minimizer outside the packet.

The structural rows H, C, P, N transport, replay, source, and name this same route. Because no row carries a host choice selector, quotient completion, or independent ambient-completeness assertion, downstream consumers cannot obtain such data from the handoff. □

Theorem 3926.4 (Ekeland variational-principle descent window)**.** *Any accepted* $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ *packet exposes the approximation row A, finite descent trace T, complete-metric readiness row M, lower-semicontinuity row L, rational readback row R, and winner window W in that order before a downstream fixed-point or minimization consumer may read the perturbation inequality. The read is therefore a finite descent-window obligation over the carrier rows of Definition 3926.1, not a host minimizer, selected choice function, quotient completion, or ambient completeness principle.*

Proof. Project the accepted carrier $E = (X, d, M, L, R, A, T, W, H, C, P, N)$ and read the obligation surface of Definition 3926.2. The approximation row A supplies the initial value, tolerance, and perturbation budget before any descent comparison can be inspected.

Next use the metric-descent handoff of Definition 3926.3. It keeps the finite trace T downstream of A and keeps every Cauchy-tail request routed through M.

The value comparisons then pass through L and R. Thus lower-semicontinuity and rational readback remain visible before the terminal perturbation inequality is exposed in W.

Finally, H, C, P, N transport, replay, source, and name the same ordered window. Since the carrier contains no coordinate for host minimizers, selected choice functions, quotient completions, or ambient completeness, the consumer receives only the displayed descent window. □

Closure status for $\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}$.

Constructive story: An $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$ packet is generated from a finite metric-domain row, displayed distance witnesses, a complete-metric descent-readiness row, a lower-semicontinuity row, regular-rational readback, approximation budgets, a finite descent trace, a winner-window row, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3926.1, Definition 3926.2, and Definition 3926.3 seed the complete-metric variational descent ledger consumed by later fixed-point and minimization packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EkelandVariationalPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host minimizer, global choice selector, quotient completion, selected cofinal descent path, ambient completeness theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric descent readiness, lower-semicontinuity readback, descent-budget coverage, winner-window exposure, and consumer nonescape for BEDC.Derived.EkelandVariationalPr

Chapter 3927

A Concrete Naming Certificate for $\mathsf{KanExtension}^{\uparrow}$

Following Mac Lane's formulation of Kan extensions as universal arrows along a functor, the $\mathsf{KanExtension}^{\uparrow}$ packet reconstructs the usual left and right extension interface as finite BEDC certificate data. It depends on the certified category, functor, and natural-transformation surfaces of chapter 61, chapter 62, and chapter 63; it also reads the triangle-comparison discipline of chapter 107 and the cocone comparison rows of chapter 106. The chapter does not import an external functor category. It names the finite rows required for a displayed extension object, its comparison natural transformation, and its universal factorization route.

Definition 3927.1 (Kan extension carrier)**.** *A* $\mathsf{KanExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, D, E, J, F, L, \eta, U, V, H, R, P, N)$$

with the following displayed rows:

1. C, D, E *are* $\mathsf{Cat}^{\uparrow}$ *certificate rows for the source, target, and codomain categories;*
2. $J : C \to D$, $F : C \to E$, *and* $L : D \to E$ *are* $\mathsf{Functor}^{\uparrow}$ *rows whose object and morphism maps are already certified by the corresponding category classifiers;*
3. $\eta : F \Rightarrow L \circ J$ *is the displayed* $\mathsf{NatTrans}^{\uparrow}$ *comparison row;*
4. U *is the universal comparison row: for every displayed* $\mathsf{Functor}^{\uparrow}$ *row* $M : D \to E$ *and* $\mathsf{NatTrans}^{\uparrow}$ *row* $\alpha : F \Rightarrow M \circ J$, *it supplies the candidate mediating* $\mathsf{NatTrans}^{\uparrow}$ *row* $\overline{\alpha} : L \Rightarrow M$*;*
5. V *is the uniqueness row: two mediating rows with the same composite comparison are identified only by the displayed natural-transformation classifier;*
6. H *records componentwise* hsame *transport for categories, functors, comparison transformations, mediating transformations, and uniqueness witnesses;*
7. R *records displayed* Cont *replay from* α *to* $\overline{\alpha}$ *and from the composite* $\overline{\alpha} \circ J$ *back to* α*;*
8. P *records* Pkg *provenance over the* $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{Adjunction}^{\uparrow}$, $\mathsf{CatColimit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$ *row.*

The carrier has no coordinate for host functor equality, quotient natural-transformation equality, an external functor category, or an uncategorified universal property outside these displayed rows.

Lean4 ✓ `BEDC.Derived.KanExtensionUp.TasteGate.KanExtensionTasteGate single carrier alignment`

Theorem 3927.2 (Kan extension NameCert obligations)**.** *For the carrier of Definition 3927.1, the local* $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$ *surface consists exactly of the category rows* C, D, E*, functor rows* J, F, L*, comparison row* η*, universal comparison row* U*, uniqueness row* V*, componentwise transport row* H*, continuation replay row* R*, provenance row* P*, and local naming row* N*. Any accepted* $\mathsf{KanExtension}^{\uparrow}$ *read must expose those rows before it may be used as a left or right Kan-extension certificate.*

Lean4 ✓ `BEDC.Derived.KanExtensionUp.KanExtensionCarrier namecert obligations`

Proof. Project the carrier coordinates. The rows C, D, E license the three category classifiers, while J, F, L license the functor data and η supplies the displayed natural-transformation comparison.

Read the universal data next. The row U provides the mediating transformation for each displayed comparison α, and V constrains any two mediating rows by the $\mathsf{NatTrans}^{\uparrow}$ classifier rather than by host equality.

Finally attach the structural rows. The row H transports all displayed components by hsame, R replays the comparison route through Cont, P records the sibling certificate provenance, and N names the packet. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 3927.3 (Kan extension universal route)**.** *Every accepted* $\mathsf{KanExtension}^{\uparrow}$ *universal comparison factors through the displayed route*

$$(C, D, E, J, F, L, \eta), \alpha \longmapsto \overline{\alpha}, \overline{\alpha} \circ J \sim \alpha$$

using only U, V, H, R, P, N*. The* $\mathsf{Adjunction}^{\uparrow}$ *and* $\mathsf{CatColimit}^{\uparrow}$ *siblings may supply triangle-shaped and cocone-shaped comparison patterns, but neither sibling replaces the displayed* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *rows by an external universal-property assertion.*

Proof. Begin with a displayed comparison $\alpha : F \Rightarrow M \circ J$. By Definition 3927.1, the only row that can produce a mediator is U, and the mediator it returns is a $\mathsf{NatTrans}^{\uparrow}$ row $\overline{\alpha} : L \Rightarrow M$ over the already certified functors.

Use the uniqueness row to keep the route internal. The row V compares mediating transformations through the natural-transformation classifier, while H transports the category, functor, and transformation rows componentwise. This prevents the factorization from becoming a host-level equality claim.

The remaining rows preserve the same route. The continuation row R replays the composite comparison, P records its category-theory provenance through the cited siblings, and N names the packet. Hence the universal route is read as finite NameCert data, not as an untracked functor-category theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{KanExtension}^{\uparrow}$ packet is generated from category rows, functor rows, a comparison natural transformation, universal and uniqueness rows, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 3927.1, Definition 3927.2, and Definition 3927.3 bind the displayed Kan-extension carrier, obligation surface, and universal route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KanExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close external functor-category existence, quotient natural-transformation equality, arbitrary pointwise Kan-extension computation, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a Lean carrier theorem that binds the universal comparison rows to an explicit kernel scope.

Chapter 3928

A Concrete Naming Certificate for $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ packet presents bounded monotone sequence convergence as explicit dyadic-window and regular-Cauchy evidence, not as general $\mathsf{Real}^{\uparrow}$ completeness or choice. It sits beside the finite stream-name surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the bounded monotone seal boundary of chapter 918. The packet gives downstream Bolzano–Weierstrass, Cauchy completion, and sequence-limit consumers a finite L10 route for monotone convergence without importing ambient completeness.

Definition 3928.1 (MonotoneConvergenceSequence carrier)**.** *A* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, L, U, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the monotone sequence source row, carrying the displayed sequence observations and the finite monotonicity classifier read by the packet.*
2. *L and U are the lower and upper dyadic bound rows, giving the bounded interval in which every inspected source window is certified to land.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the initial and tail windows requested by the monotone-convergence consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic mesh widths, lower-upper bracket gaps, and tail budgets.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the monotone windows and their dyadic tolerance ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the source, bounds, finite windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for S, L, U, W, D, R, E and the structural rows.*
8. *C records displayed* Cont *replay from monotone source inspection through dyadic bounds, finite windows, tolerance shrinkage, regular readback, and the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$, $\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, U, W, D, R, E, H, C, P, N$. It has no coordinate for an unrestricted supremum principle, quotient stream equality, host sequence equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneConvergenceSequenceUp.MonotoneConvergenceSequenceTasteGate single carrier alignment`

Theorem 3928.2 (MonotoneConvergenceSequence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier*

$$M = (S, L, U, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$ *surface consists exactly of monotone source exposure through S, lower and upper dyadic bound exposure through L, U, finite window exposure through W, dyadic tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal handoff through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3928.1. The tuple displays the monotone source row, dyadic bound rows, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, and the structural transport, replay, provenance, and naming rows.

The bounded monotone evidence is confined to S, L, U, W, D. The source row gives the monotonicity classifier, the bound rows give the visible interval enclosure, and the window and tolerance rows record the finite bracket shrinkage that a consumer may inspect.

The analytic handoff is then R, E. The regular rational readback row is obtained only from the displayed windows and dyadic ledger, and the Real seal row is exposed only after that readback has been displayed.

Finally H, C, P, N preserve and name the same finite route. Since the carrier contains no supremum, quotient stream, host sequence equality, host real equality, choice, or ambient completeness coordinate, those objects are outside the local NameCert surface. □

Theorem 3928.3 (MonotoneConvergenceSequence regular Cauchy route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{BoundedMonotoneConvergenceSeal}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The bounded monotone seal supplies the source and interval rows, the finite stream windows expose the inspected observations, the dyadic ledger supplies the tail tolerance budget, and the regular rational readback precedes the terminal Real seal.

Proof. Start with an accepted carrier $M = (S, L, U, W, D, R, E, H, C, P, N)$. The monotone source and bound rows S, L, U are the only coordinates through which bounded monotone evidence can enter the packet.

Read the bounded seal boundary through chapter 918. It supplies the public source-and-bound discipline consumed by this carrier, but it does not provide an unrestricted completeness principle.

Next read the finite observation route. The windows W are $\mathsf{StreamName}^{\uparrow}$ data, the tolerance ledger D is $\mathsf{DyadicRatCore}^{\uparrow}$ data, and the regular rational row R repacks exactly those displayed windows and tolerances as $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The Real seal E is therefore downstream of the source, dyadic bounds, finite windows, dyadic tolerance ledger, and regular rational readback. Transport and replay through H, C, P, N preserve that same route, so no consumer receives a hidden supremum, quotient stream equality, host sequence equality, host real equality, countable-choice schedule, or ambient Real-completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneConvergenceSequence}^{\uparrow}$ packet is generated from a monotone source row, lower and upper dyadic bound rows, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3928.1, Definition 3928.2, and Definition 3928.3 seed the bounded monotone regular-Cauchy route over StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneConvergenceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a general supremum principle, quotient stream equality, host sequence equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone seal handoff, dyadic bracket shrinkage, RegSeqRat readback exactness, RealUp seal nonescape, and completeness-refusal proofs for BEDC.Derived.MonotoneConvergenceS

Chapter 3929

A Concrete Naming Certificate for $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ packet records the finite route by which a displayed Picard orbit is read as a regular rational Cauchy name and then sealed as a $\mathsf{Real}^{\uparrow}$ object. It sits over the orbit rows of chapter 2359, the contraction ledger of chapter 396, the precision schedule of chapter 292, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the bounded ambient interface of chapter 128. The packet is deliberately finite: it certifies the route from named Picard rows to a regular Cauchy seal, not an unrestricted fixed-point selection theorem.

Definition 3929.1 (Regular Cauchy Picard seal carrier)**.** *A* $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, B, M, Q, E, H, C, P, N)$$

whose rows have the following roles.

1. *I is a finite* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row displaying the initial point, the iterated contraction rows, and the finite prefix consumed by the certificate.*
2. *B is a* $\mathsf{BanachFixedPoint}^{\uparrow}$ *contraction-budget row exposing only the local Lipschitz decrease data needed to compare successive Picard rows.*
3. *M is a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule extracted from the displayed contraction budget and attached to finite precision requests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite Picard orbit and the modulus schedule.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached through the displayed regular rational readback.*
6. *H, C, P, N are respectively the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a selected fixed point, host function extensionality, quotient stream equality, ambient completeness, or an unrestricted Banach fixed-point existence principle.

Lean4 ✓ `BEDC.Derived.RegularCauchyPicardSealUp.RegularCauchyPicardSealTasteGate single carrier alignment`

Theorem 3929.2 (Regular Cauchy Picard seal obligations)**.** *For every accepted carrier $R = (I, B, M, Q, E, H, C, P, N)$ of Definition 3929.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$ *surface*

consists exactly of the Picard orbit row I, the contraction-budget row B, the extracted modulus row M, the regular rational readback row Q, the terminal $\mathsf{Real}^{\uparrow}$ *seal row E, and the structural rows H, C, P, N. A downstream consumer must read the rows in the displayed order*

$$I \to B \to M \to Q \to E \to H, C, P, N.$$

No accepted obligation row can be replaced by a selected fixed point, a host-level proof that two functions are extensionally equal, quotient equality between stream representatives, an ambient completeness theorem, or an unrestricted Banach fixed-point existence row.

Proof. Project the carrier of Definition 3929.1. The analytic rows are exactly the finite Picard orbit, contraction budget, extracted modulus, regular rational readback, and terminal Real seal; the remaining rows are the standard BEDC transport, replay, provenance, and naming surface.

The Picard row is read first because the modulus schedule is extracted from finite comparisons between displayed iterates. The contraction-budget row supplies those comparisons, and the $\mathsf{CauchyModulus}^{\uparrow}$ row records the precision threshold as data rather than as a host completeness argument.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback can then consume only the displayed orbit prefix and the extracted modulus. Once that readback is present, the $\mathsf{Real}^{\uparrow}$ seal is reached through the same regular Cauchy route used by the surrounding Real-and-Completion chapters.

Finally the rows H, C, P, N transport and replay this finite route as the local certificate. Since none of the carrier coordinates stores a selected fixed point, function extensionality principle, quotient stream equality, ambient completeness theorem, or unrestricted fixed-point existence theorem, no consumer of the NameCert surface can obtain those claims from this packet. □

Theorem 3929.3 (Regular Cauchy Picard seal route)**.** *Let $R = (I, B, M, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyPicardSeal}^{\uparrow}$ *carrier. Every finite precision request for the Real-facing Picard limit factors through the route*

$$\mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{BanachFixedPoint}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first reads a finite Picard orbit prefix, then extracts a Cauchy modulus from the contraction budget, then performs $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It does not claim a chosen fixed point, host function extensionality, quotient stream equality, ambient completeness, or unrestricted Banach fixed-point existence.*

Proof. Begin with the accepted carrier and the obligation theorem above. They force every precision request to start with the displayed finite Picard orbit I and the contraction budget B.

Next extract the modulus row M. This step is finite: a requested precision is matched with a displayed threshold from the contraction ledger, so the certificate does not invoke a global completeness principle or a choice schedule over all requests.

Use the extracted threshold to read the regular rational row Q. The readback is the ordinary $\mathsf{RegSeqRat}^{\uparrow}$ passage from finite rational data and a modulus to a regular Cauchy name; it has no quotient-stream coordinate and no host equality coordinate.

Finally pass from Q to the terminal Real seal E and replay the route through H, C, P, N. The resulting seal records the finite Picard-to-regular-Cauchy route, while the selected fixed point and unrestricted Banach existence claims remain outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPicardSeal}^{\uparrow}}$.

Constructive story: A RegularCauchyPicardSeal$^{\uparrow}$ packet is generated from a finite PicardIteration$^{\uparrow}$ orbit prefix, a BanachFixedPoint$^{\uparrow}$ contraction-budget row, an extracted CauchyModulus$^{\uparrow}$ precision schedule, a RegSeqRat$^{\uparrow}$ readback row, a terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyPicardSeal}^{\uparrow}}$, seed)

Scope closed: Definition 3929.1, Definition 3929.2, and Definition 3929.3 seed the finite Picard-orbit route from contraction data through modulus extraction, regular rational readback, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyPicardSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host function extensionality, quotient stream equality, ambient completeness, or unrestricted Banach fixed-point existence.

Upgrade path: A formal route can encode the finite carrier rows and prove the Picard-orbit modulus extraction, regular rational readback, and terminal Real-seal route.

Chapter 3930

A Concrete Naming Certificate for ArzelaAscoliSelection$^{\uparrow}$

Following Arzela–Ascoli, Kelley, and Bishop–Bridges constructive analysis, the ArzelaAscoliSelection$^{\uparrow}$ packet records the finite selection surface by which an equicontinuous bounded family on a compact metric source exposes a regular convergent subsequence. It sits after the family ledger of chapter 413, the classical compactness packet of chapter 412, the bounded-family surface of chapter 3522, and the diagonal extraction route of chapter 2219. The packet names the selection route through displayed finite rows only; it does not export a host compact-open topology, an arbitrary sequence choice, quotient equality of functions, or ambient completeness outside the named certificates.

Definition 3930.1 (ArzelaAscoliSelection carrier)**.** *An* ArzelaAscoliSelection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, E, B, D, M, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* CompactMetric$^{\uparrow}$ *source row supplying finite net and compact inspection data.*
2. *E is the* EquicontinuityFamily$^{\uparrow}$ *ledger, including shared-center modulus rows for the displayed family.*
3. *B is the* BoundedFunctionFamily$^{\uparrow}$ *row, recording pointwise and sup-norm boundedness for the same family.*
4. *D is the* AscoliDiagonal$^{\uparrow}$ *row selecting compatible finite windows across the source net and target readback.*
5. *M is the finite modulus row read through* ArzelaAscoliFiniteModulus$^{\uparrow}$ *and the compact uniform-continuity route.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the selected finite windows.*
7. *R is the* Real$^{\uparrow}$ *seal row attached to the regular readback of the selected subsequence values.*
8. *H records componentwise* hsame *transport for X, E, B, D, M, Q, R.*
9. *C records displayed* Cont *replay from compact-source inspection through equicontinuity, boundedness, diagonal selection, regular readback, and real sealing.*

10. *P is the* Pkg *provenance over* ArzelaAscoliSelection$^{\uparrow}$, CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, BoundedFunctionFamily$^{\uparrow}$, AscoliDiagonal$^{\uparrow}$, ArzelaAscoliFiniteModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ArzelaAscoliSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-selected subsequence, a compact-open completion, quotient equality of functions, countable choice, or an unstated completeness principle.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliSelectionUp.TasteGate.ArzelaAscoliSelectionTasteGate single carrier alignment`

Theorem 3930.2 (ArzelaAscoliSelection NameCert obligations)**.** *For every accepted* ArzelaAscoliSelection$^{\uparrow}$ *carrier $A = (X, E, B, D, M, Q, R, H, C, P, N)$, the local* NameCert$_{\text{ArzelaAscoliSelection}^{\uparrow}}$ *surface consists exactly of compact-source admission through* CompactMetric$^{\uparrow}$, *equicontinuity-family admission through* EquicontinuityFamily$^{\uparrow}$, *bounded-family admission through* BoundedFunctionFamily$^{\uparrow}$, *diagonal-window admission through* AscoliDiagonal$^{\uparrow}$, *finite modulus readback through* ArzelaAscoliFiniteModulus$^{\uparrow}$, *regular rational readback through* RegSeqRat$^{\uparrow}$, *terminal real sealing through* Real$^{\uparrow}$, *carrier storage in* BHist, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through* NameCert.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliSelectionUp.ArzelaAscoliSelectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3930.1. The rows $X, E, B, D, M, Q, R, H, C, P, N$ are the full displayed inventory.

Read the source and family rows first. The compact metric row X supplies finite source nets, the equicontinuity row E supplies shared modulus ledgers, and the bounded-family row B supplies the bound surface needed before any diagonal selection is public.

Read the selection rows next. The diagonal row D consumes the finite net and family ledgers, while M records the finite modulus compatibility that keeps the selected windows inside the Arzela–Ascoli route. The row Q then exposes the selected windows as regular rational readback, and R seals the resulting values as Real$^{\uparrow}$ data.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier has no coordinate for host subsequence choice, compact-open completion, quotient equality of functions, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3930.3 (ArzelaAscoliSelection selected regular subsequence route)**.** *Every accepted* ArzelaAscoliSelection$^{\uparrow}$ *packet exposes a selected regular subsequence only through the finite route*

CompactMetric$^{\uparrow}$, EquicontinuityFamily$^{\uparrow}$, BoundedFunctionFamily$^{\uparrow}$, AscoliDiagonal$^{\uparrow}$, ArzelaAscoliFiniteModulus$^{\uparrow}$, Reg

The route first displays compact source nets, family equicontinuity, boundedness, and diagonal finite windows; only then does it expose regular rational readback and the real seal. It exports no arbitrary subsequence selector, compact-open host topology, quotient function equality, countable choice, or ambient completeness theorem.

Proof. Use Definition 3930.2. It orders the compact source, equicontinuity, bounded-family, diagonal, finite-modulus, regular-readback, and Real-seal rows inside one accepted carrier.

The diagonal row can be consumed only after X, E, B are displayed. The finite modulus row M ties the selected windows back to the shared equicontinuity ledger and the bounded-family surface, so the selection does not detach from the family certificate.

The readback rows Q, R are downstream. They convert the displayed windows into $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$ data, while H, C, P, N preserve the same route.

Thus the packet supplies a finite BEDC selection route. None of the refused host-level objects is present in the carrier inventory. □

Falsifiable prediction: An accepted $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ consumer that reads a selected regular subsequence without first exposing CompactMetric, EquicontinuityFamily, BoundedFunctionFamily, AscoliDiagonal, finite modulus, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows refutes the carrier surface.

Independence witness: compactmetric, equicontinuityfamily, bounded-function-family, ascoli-diagonal, arzelaascoli: $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-source nets, equicontinuity ledgers, bounded-family rows, diagonal selection windows, finite modulus readback, regular rational data, Real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliSelection}^{\uparrow}$ packet is generated from CompactMetricUp source nets, EquicontinuityFamilyUp modulus ledgers, BoundedFunctionFamilyUp boundedness rows, AscoliDiagonalUp finite windows, ArzelaAscoliFiniteModulusUp compatibility rows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3930.1, Definition 3930.2, and Definition 3930.3 seed the finite constructive selection route for equicontinuous bounded families over compact metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, compact-open topology, arbitrary subsequence choice, quotient equality of functions, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, family ledger exactness, boundedness readback, diagonal-window compatibility, finite modulus transport, regular readback, Real sealing, provenance, and local naming for BEDC.Derived.ArzelaAscoliSelectionUp.

Chapter 3931

A Concrete Naming Certificate for $\mathsf{ContractionMappingTheorem}^{\uparrow}$

Following Banach's contraction principle and Bishop–Bridges constructive metric analysis, the $\mathsf{ContractionMappingTheorem}^{\uparrow}$ packet records the fixed-point theorem as a finite complete-metric and Picard-orbit certificate. It sits next to the complete-metric carrier of chapter 128, the metric-distance surface of chapter 57, the contractive-map row of chapter 3819, the completion boundary of chapter 95, and the terminal real seal of chapter 38. The packet names the theorem-facing route from a displayed contraction to a unique fixed-point seal; it is not a host fixed-point theorem, a quotient completion principle, a selected representative, or an ambient completeness axiom.

Definition 3931.1 (Contraction mapping theorem carrier)**.** *A* $\mathsf{ContractionMappingTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, M, D, A, O, L, U, H, T, P, N)$$

with the following visible rows:

1. *C is the complete-metric source row, carrying the displayed Cauchy admission and endpoint-read interface from* $\mathsf{CompleteMetric}^{\uparrow}$*.*
2. *M is the metric-distance row, exposing only the finite distance comparisons admitted by the* $\mathsf{Met}^{\uparrow}$ *surface.*
3. *D is the* $\mathsf{ContractiveMap}^{\uparrow}$ *row, recording the self-map graph, contraction constant ledger, and one-step distance decrease over the same metric source.*
4. *A is the Picard/Cauchy orbit schedule, listing the finite initial point, iterate windows, residual-distance bounds, and Cauchy-modulus handoff generated from D.*
5. *O is the* $\mathsf{Completion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal row that admits the orbit endpoint only after C, M, D, A have been displayed.*
6. *L is the fixed-point uniqueness ledger, comparing two endpoint reads through the same metric-distance row and the contraction decrease route before local equality is exported.*
7. *U records the fixed-point equation read, requiring the endpoint seal, the contractive-map graph, and the Picard orbit replay to agree componentwise.*
8. *H stores componentwise* hsame *transport for the complete-metric source, metric-distance row, contractive-map row, Picard/Cauchy schedule, completion or real seal, fixed-point uniqueness ledger, fixed-point equation row, replay, provenance, and local naming.*

9. *T stores displayed* Cont *replay from contraction admission through Picard iteration, Cauchy endpoint admission, fixed-point equation read, and uniqueness comparison.*
10. *P is the* Pkg *provenance over* ContractionMappingTheorem$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Met$^{\uparrow}$, ContractiveMap$^{\uparrow}$, Completion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ *row.*

The classifier accepts packets only when the complete-metric row, metric-distance row, contractive-map row, Picard/Cauchy schedule, endpoint seal, fixed-point equation row, uniqueness ledger, transport, replay, provenance, and local naming row are all read on the same displayed route.

Theorem 3931.2 (Contraction mapping theorem NameCert obligations)**.** *For every accepted* ContractionMappingTheorem$^{\uparrow}$ *carrier*

$$F = (C, M, D, A, O, L, U, H, T, P, N),$$

the local NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ *obligations are exactly complete-metric source admission, metric-distance visibility, contractive-map row admission, Picard/Cauchy orbit scheduling,* Completion$^{\uparrow}$ *or* Real$^{\uparrow}$ *endpoint sealing, fixed-point equation replay, fixed-point uniqueness comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.ContractionMappingTheoremUp.ContractionMappingTheoremTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 3931.1. The source rows C and M are the only complete-metric and metric-distance data available to a theorem-facing consumer, and D is the only row that carries contraction information.

The Picard/Cauchy schedule A is generated from the displayed contraction row. Its residual bounds and Cauchy-modulus handoff must be read before the endpoint seal O is visible, so the completion or real seal cannot be consumed as an ambient completeness principle.

The fixed-point equation row U and uniqueness ledger L are downstream rows. They compare only endpoint reads that have traversed C, M, D, A, O and then preserve that route through the structural rows H, T, P, N.

Thus the NameCert obligations are precisely the displayed finite route. No carrier coordinate supplies a host fixed-point theorem, quotient completion equality, selected representative, unrestricted choice, trichotomy, or open-cover compactness. □

Falsifiable prediction: Every accepted ContractionMappingTheorem$^{\uparrow}$ consumer reads a complete-metric source, a metric-distance row, a contractive-map row, a Picard/Cauchy orbit schedule, a completion or real endpoint seal, a fixed-point equation row, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before it treats the packet as a fixed-point theorem.

Independence witness: completemetric, metric, contractive-map, completion, real: ContractionMappingTheorem binds the theorem-facing fixed-point route in one NameCert packet; the listed siblings expose source, distance, contraction, endpoint, or seal rows separately but do not jointly carry Picard/Cauchy scheduling, fixed-point equation replay, uniqueness, transport, replay, provenance, and local naming for the theorem surface.

Closure status for NameCert$_{\text{ContractionMappingTheorem}^{\uparrow}}$ **·**

Constructive story: A ContractionMappingTheorem$^{\uparrow}$ packet is generated from a CompleteMetric source, a Metric distance row, a ContractiveMap row, a Picard/Cauchy orbit schedule, a CompletionUp or RealUp seal, a fixed-point

equation row, a uniqueness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ContractionMappingTheorem}\uparrow}$, seed)

Scope closed: Definition 3931.1 and Definition 3931.2 seed the finite fixed-point theorem certificate over complete-metric, metric-distance, contractive-map, Picard/Cauchy, completion or real seal, uniqueness, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{ContractionMappingTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host fixed-point equality, quotient completion equality, selected representatives, unrestricted choice, trichotomy, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contractive-map visibility, Picard/Cauchy scheduling, endpoint sealing, fixed-point equation replay, uniqueness routing, transport, replay, provenance, and local naming for BEDC.Derived.ContractionMappingTheoremUp.

Chapter 3932

A Concrete Naming Certificate for HilbertAlexanderBlocker$^{\uparrow}$

The HilbertAlexanderBlocker$^{\uparrow}$ packet realizes the Hilbert-series blocker complex of Definition 9462.1622 as finite BEDC NameCert data. It sits after the simplicial-complex carrier of chapter 238, the polynomial carrier of chapter 50, the finite-trace evaluator surface of chapter 818, the DNA boundary-tile packet of chapter 3393, and the arity-ten BHist tuple capstone of chapter 1364. The packet names blocker-cover, face-polynomial, Hilbert-row, Alexander-dual, and non-Cohen–Macaulay rows as displayed finite data; it is not a host Stanley–Reisner algebra oracle, a biological stochastic model, quotient codon semantics, or an unrestricted homology computation.

Definition 3932.1 (HilbertAlexanderBlocker carrier)**.** *A* HilbertAlexanderBlocker$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (S, P, T, B, F, L, A, R, C, Q, N)$$

with the following displayed rows:

1. *S is the* SimplicialComplex$^{\uparrow}$ *row for the blocker complex and its four omitted-coordinate cover faces.*
2. *P is the* Poly$^{\uparrow}$ *row recording the face polynomial and the Hilbert numerator polynomial.*
3. *T is the* FiniteTraceEvaluator$^{\uparrow}$ *row that evaluates the finite inclusion-exclusion table for cover intersections.*
4. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *row carrying the six-window boundary source and trigger coordinates.*
5. *F is the blocker-cover row, recording the four facets and the intersection multiplicities $4, 6, 4, 1$.*
6. *L is the Hilbert-row ledger, recording the rational-function denominator powers $14, 6, 2, 0$ and the signed inclusion-exclusion coefficients.*
7. *A is the Alexander-dual row, recording the four generator complements and their lcm-degree ledger.*
8. *R records componentwise* hsame *transport for S, P, T, B, F, L, A.*
9. *C records displayed* Cont *replay from boundary-tile reads through blocker cover, face polynomial, Hilbert rows, and Alexander-dual rows.*

10. *Q is the* Pkg *provenance over* HilbertAlexanderBlocker$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, Poly$^{\uparrow}$, FiniteTraceEvaluator$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, BHistDecaTupleNameCert$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient polynomial-ring equality oracle, an unrestricted minimal resolution, stochastic biological semantics, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision procedure.

Theorem 3932.2 (HilbertAlexanderBlocker NameCert obligations)**.** *For every accepted* HilbertAlexanderBlocker$^{\uparrow}$ *carrier $H = (S, P, T, B, F, L, A, R, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$ *surface consists exactly of simplicial-complex admission through S, polynomial-row admission through P, finite trace evaluation through T, DNA boundary-tile source admission through B, blocker-cover exactness through F, Hilbert-row readback through L, Alexander-dual ledger exposure through A, componentwise* hsame *transport through R, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through N.*

Lean4 ✓ `BEDC.Derived.HilbertAlexanderBlockerUp.HilbertAlexanderBlockerNameCertObligations`

Proof. Project the carrier of Definition 3932.1. The rows $S, P, T, B, F, L, A, R, C, Q, N$ are the full accepted inventory.

The blocker-cover side is finite. The row B supplies the boundary-tile source, S supplies the simplicial-complex carrier, and F records the four cover facets together with their intersection multiplicities.

The Hilbert side is finite as well. The row T evaluates the inclusion-exclusion table, P stores the face and numerator polynomials, and L stores the denominator powers and signs of the Hilbert-series row.

The Alexander-dual row A records the generator-complement ledger attached to the same blocker cover. The structural rows R, C, Q, N transport, replay, source, and name exactly this packet, so no accepted NameCert obligation can project an ambient polynomial-ring oracle, unrestricted resolution, stochastic model, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision. □

Theorem 3932.3 (HilbertAlexanderBlocker finite Hilbert row)**.** *Every Hilbert-series read from an accepted* HilbertAlexanderBlocker$^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{SimplicialComplex}^{\uparrow},\ \mathsf{Poly}^{\uparrow},\ \mathsf{FiniteTraceEvaluator}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{BHistDecaTupleNameCert}^{\uparrow} \longmapsto F,$$

The read first displays the blocker cover and intersection table, then evaluates the face-polynomial and Hilbert-row ledger, and only then exposes Alexander-dual and local naming rows. It exports no ambient Stanley–Reisner algebra oracle, unrestricted minimal resolution, quotient codon equality, hidden homology computation, or biological stochastic model.

Lean4 ✓ `BEDC.Derived.HilbertAlexanderBlockerUp.HilbertAlexanderBlockerFiniteHilbertRow`

Proof. Apply Definition 3932.2. It places the simplicial-complex, polynomial, trace-evaluation, and boundary-tile sources before the blocker-cover and Hilbert-row ledgers.

The finite trace row T evaluates only the displayed intersection table F. The polynomial row P and Hilbert ledger L therefore read the same finite cover data rather than a host polynomial-ring quotient.

The Alexander-dual row A is downstream of the same cover. Transport, replay, provenance, and local naming through R, C, Q, N preserve that route, so the Hilbert-series read remains a finite NameCert packet. □

Falsifiable prediction: Changing the blocker-cover multiplicity row F changes the Hilbert denominator-power ledger L before any accepted $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ consumer can preserve the displayed Hilbert series.

Independence witness: simplicialcomplex, polynomial, finite-trace-evaluator, dna-window-six-boundary-tile, b-hist-deca-tuple-name-cert: $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries blocker-cover rows, face-polynomial rows, Hilbert-series rows, Alexander-dual rows, finite trace evaluation, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}$.

Constructive story: A $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$ packet is generated from a SimplicialComplexUp blocker complex, PolynomialUp face and numerator rows, FiniteTraceEvaluatorUp inclusion-exclusion evaluation, DnaWindowSixBoundaryTileUp source rows, blocker-cover data, Hilbert-row ledgers, Alexander-dual ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3932.1, Definition 3932.2, and Definition 3932.3 seed the finite Hilbert-series blocker packet over cover, polynomial, trace-evaluation, boundary-tile, and tuple-name rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertAlexanderBlocker}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host Stanley–Reisner algebra equality, unrestricted minimal resolutions, stochastic biological semantics, quotient codon equality, hidden homology computation, or external Cohen–Macaulay decision.

Upgrade path: Obligation closure requires checked carrier admission, blocker-cover exactness, face-polynomial readback, Hilbert-row ledger exactness, Alexander-dual ledger exposure, trace-evaluation stability, transport, replay, provenance, and local naming for BEDC.Derived.HilbertAlexanderBlockerUp.

Chapter 3933

A Concrete Naming Certificate for $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$

Following Bishop and Bridges constructive real analysis, the $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface used when a compact interval argument must choose a displayed located subinterval before completion-facing consumers read a real name. It sits over the located interval carrier of chapter 394, the closed bounded interval source of chapter 1591, the nested interval route of chapter 320, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet is a NameCert surface for a displayed compact-interval selection route; it is not open-cover compactness, selected points, trichotomy, quotient $\mathsf{Real}^{\uparrow}$ equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3933.1 (Located compact interval selection carrier)**.** *A* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, B, N, D, S, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *source row, exposing the lower and upper located endpoint windows and the rational containment ledger.*
2. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *compact-bound row, exposing the closed bounded source used by the selection route.*
3. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *refinement row, recording the finite nested subintervals retained by the selection procedure.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for cell width, endpoint separation, and retained-refinement budgets.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint and retained-interval observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected located interval windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after I, B, N, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for I, B, N, D, S, R, E and the local packet shape.*
9. *C records displayed* Cont *replay from located interval source and compact-bound rows through nested refinements, dyadic tolerances, windows, readback, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$ *row.*

The carrier exports no open-cover compactness theorem, selected point of the interval, trichotomy principle, quotient equality of real names, or ambient real-completeness row.

Lean4 ✓ `BEDC.Derived.LocatedCompactIntervalSelectionUp.LocatedCompactIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 3933.2 (Located compact interval selection NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier*

$$L = (I, B, N, D, S, R, E, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$ *surface consists exactly of located interval exposure through I, closed bounded interval exposure through B, nested interval refinement through N, dyadic tolerance accounting through D, stream-window exposure through S, regular rational readback through R, real-name sealing through E, componentwise transport through H, replay through C, provenance through P, and local naming through Q.*

Proof. Project the carrier of Definition 3933.1. The finite packet exposes exactly the located interval source, compact-bound source, nested refinement ledger, dyadic tolerance row, stream windows, regular rational readback, Real seal, and structural rows.

Read the interval rows first. The located row I supplies endpoint windows and rational containment, while the closed bounded row B supplies the compact interval source used by the selection route.

The refinement and readback rows are downstream of those sources. The nested ledger N records retained subintervals, D records the dyadic budgets attached to those refinements, and S, R, E expose the selected interval windows only through stream, regular rational, and Real rows.

Finally H, C, P, Q transport, replay, package, and name the same route. Since no other coordinate exists, the listed rows exhaust the local NameCert obligations and cannot import open-cover compactness, selected points, trichotomy, quotient real equality, or ambient Real completeness. □

Theorem 3933.3 (Located compact interval selection real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

The selected located interval is therefore handed off as displayed endpoint windows, nested subinterval rows, dyadic tolerance budgets, stream windows, regular rational readbacks, and a terminal Real seal, not as a selected point, quotient real representative, trichotomy comparison, open-cover compactness argument, or ambient completeness principle.

Proof. Begin with an accepted carrier $L = (I, B, N, D, S, R, E, H, C, P, Q)$. The source rows I and B are the only located and compact interval coordinates available to the packet.

Next inspect the nested and dyadic rows. The row N records the retained finite subintervals and D records the tolerances that make those retained intervals readable without a point selector.

The real-facing readback then passes through S, R, E. Stream windows expose the finite endpoint observations, regular rational rows read them as regular data, and the Real seal is downstream of those displayed rows.

Transport and replay through H, C, P, Q preserve that same route for completion-facing consumers. The packet has no coordinate for open-cover compactness, selected points, trichotomy, quotient Real equality, or ambient Real completeness, so none of those principles can be exported by the handoff. □

Falsifiable prediction: A compact-interval consumer that omits the located interval row, the closed bounded interval row, or the nested refinement row cannot claim a $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ selection handoff.

Independence witness: locatedinterval, closedboundedinterval, nestedinterval, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source interval rows, compact-bound rows, nested refinements, dyadic budgets, stream windows, regular readbacks, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCompactIntervalSelection}^{\uparrow}$ packet is generated from a LocatedInterval source row, a ClosedBoundedInterval compact-bound row, a NestedInterval refinement row, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3933.1, Definition 3933.2, and Definition 3933.3 seed the compact interval selection route over located interval, closed bounded interval, nested interval, dyadic, stream, regular rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactIntervalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, selected points, trichotomy, quotient $\mathsf{Real}^{\uparrow}$ equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval source exposure, closed-bounded source exposure, nested refinement routing, dyadic tolerance accounting, stream readback, RegSeqRat exactness, RealUp seal nonescape, and selection-refusal proofs for BEDC.Derived.LocatedCompactIntervalSelectionUp.

Chapter 3934

A Concrete Naming Certificate for CompactUniformRadiusLedger$^{\uparrow}$

Following Lebesgue and Bishop–Bridges, the CompactUniformRadiusLedger$^{\uparrow}$ packet records the finite positive-radius ledger used in the Heine–Cantor route of chapter 9237. It sits between the compact finite-net surface of chapter 129, the pointwise continuity surface of chapter 130, the positive Lebesgue-radius packet of chapter 452, and the downstream UniformModulus$^{\uparrow}$ consumer of chapter 369. The packet names the radius choices and their finite lower-bound fold as displayed BEDC rows; it does not export an open-cover compactness theorem, a selected subsequence, a host minimum over radii, quotient equality of reals, or an external analytic modulus.

Definition 3934.1 (Compact uniform radius ledger carrier)**.** *A* CompactUniformRadiusLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, M, G, F, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, carrying the finite* ProbeBundle *of centers and the source-distance coverage rows.*
2. *M is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus ledger for the same source and target metrics.*
3. *G is the* FiniteLebesgueNumber$^{\uparrow}$ *positive-radius row, assigning a positive radius below the cover radius attached to each displayed center.*
4. *F is the* Rat$^{\uparrow}$ *lower-bound fold row, produced by primitive recursion over the finite radius bundle.*
5. *R records the radius-selection compatibility row saying that the folded lower bound is below every positive radius used by G.*
6. *U is the* UniformModulus$^{\uparrow}$ *output row that may be read only after K, M, G, F, R have been displayed.*
7. *H records componentwise* hsame *transport for the compact-net, pointwise-modulus, Lebesgue-radius, rational-fold, compatibility, and output rows.*
8. *C records displayed* Cont *replay from finite-net inspection through pointwise-modulus reads, positive-radius extraction, rational folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* CompactUniformRadiusLedger$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, Rat$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through these rows. No coordinate records open-cover compactness, selected subsequences, external choice of a minimum radius, host real equality, or a semantic modulus function outside the displayed $\mathsf{UniformModulus}^{\uparrow}$ *output.*

Lean4 ✓ `BEDC.Derived.CompactUniformRadiusLedgerUp.CompactUniformRadiusLedgerTasteGate single carrier alignment`

Theorem 3934.2 (Compact uniform radius ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ *carrier* $L = (K, M, G, F, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedge}}$ *surface consists exactly of compact-net admission through* K*, pointwise-modulus admission through* M*, positive-radius exposure through* G*, finite rational lower-bound folding through* F*, compatibility exposure through* R*,* $\mathsf{UniformModulus}^{\uparrow}$ *output through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3934.1. The rows $K, M, G, F, R, U, H, C, P, N$ are the complete displayed inventory.

Read the analytic input rows first. The compact-metric row K supplies the finite net and its coverage witnesses, while M supplies the pointwise-modulus rows at the displayed centers. The positive-radius row G ties these local moduli to a finite Lebesgue-radius ledger.

Next read the finite fold. The $\mathsf{Rat}^{\uparrow}$ row F compares the positive radii by primitive recursion over the finite bundle, and R records that the resulting lower bound is below each radius used by G. These rows are BEDC histories, not a host minimum.

Finally U, H, C, P, N export, transport, replay, source, and name exactly this ledger. Since the carrier has no coordinate for open-cover compactness, selected subsequences, external minimum selection, host real equality, or a semantic modulus function, the listed rows exhaust the local NameCert obligations. □

Theorem 3934.3 (Compact uniform radius ledger Heine–Cantor handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ *packet reads the Heine–Cantor radius choice through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The handoff first displays the finite net, then the pointwise-modulus and positive-radius rows, then the rational lower-bound fold, and only then exports the uniform-modulus row. It supplies the finite-net lower-bound route described in chapter 9237, without importing open-cover compactness, sequential compactness, choice of a minimum radius, quotient real equality, or an ambient continuity theorem.

Proof. Use Definition 3934.2. It orders the compact-net row, pointwise-modulus row, positive-radius row, rational-fold row, compatibility row, and uniform-modulus output inside one accepted carrier.

The finite net in K supplies the centers that M may inspect. The row G records a positive radius below the local cover radius at each center, so the local moduli are tied to the same finite bundle that compactness exposed.

The fold row F and compatibility row R then produce the common precision history consumed by U. Because F is a $\mathsf{Rat}^{\uparrow}$ fold over the displayed bundle, the lower bound is stable under the classifier and does not depend on an external minimum operator.

Thus the $\mathsf{UniformModulus}^{\uparrow}$ consumer receives a finite BEDC radius ledger. The refused analytic objects are absent from the carrier and cannot be reconstructed from H, C, P, N. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformRadiusLedger}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, FiniteLebesgueNumberUp positive-radius rows, RatUp lower-bound folds, UniformModulusUp output rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3934.1, Definition 3934.2, and Definition 3934.3 seed the finite Lebesgue-radius ledger used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, sequential compactness, selected-subsequence extraction, host minimum selection, quotient real equality, or external analytic modulus choice.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, pointwise-modulus admission, positive-radius exposure, rational lower-bound folding, uniform-modulus output exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformRadiusLedgerUp.

Chapter 3935

A Concrete Naming Certificate for $\mathsf{LowerSemicontinuityReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, with the point-set vocabulary of Kelley and Bourbaki read only as finite BEDC routing data, the $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ packet specializes lower semicontinuity to the Real-and-Completion corridor. It is not a synonym of chapter 1536: that sibling names the general lower-real epigraph surface, while the present chapter records the real-name, regular-rational, topology, and completion-facing threshold rows needed by Real/Completion consumers. The packet sits beside the real carrier of chapter 38, the regular rational source of chapter 301, the stream schedule of chapter 126, the lower-real ledger of chapter 1446, the continuous-map surface of chapter 130, and the completion-facing real routes of chapter 3944.

Definition 3935.1 (LowerSemicontinuityReal carrier)**.** *A* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, T, F, W, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *source carrier row, read through accepted real-name histories rather than host real points;*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *threshold row whose basic opens are represented by finite rational cuts and stream windows;*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *or value-row handoff used when the lower-semicontinuous function is inspected through a displayed map surface;*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite source and value windows used by each threshold query;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row for real-name and value approximants;*
6. *E is the* $\mathsf{LowerReal}^{\uparrow}$ *epigraph ledger, recording finite rows (x, q) that witness the rational lower threshold q at the inspected real name x;*
7. *O is the open-threshold route, recording that a query of the form $q < f(x)$ is answered by a finite lower-real row and then exposed as a basic* $\mathsf{Topology}^{\uparrow}$ *neighbourhood of the accepted real name;*

8. *H records componentwise* hsame *transport for real-name rows, topology thresholds, map rows, stream schedules, regular-rational readbacks, lower-real epigraph rows, threshold routes, provenance rows, and local naming rows;*
9. *C records displayed* Cont *replay from a threshold request through $X, T, F, W, R, E, O, H, P, N$;*
10. *P is the* Pkg *provenance over* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real completeness, quotient real equality, trichotomy, open-cover compactness, semantic liminf, or a selected limit point in a completed host space.

Lean4 ✓ `BEDC.Derived.LowerSemicontinuityRealUp.LowerSemicontinuityRealTasteGate single carrier alignment`

Theorem 3935.2 (LowerSemicontinuityReal NameCert obligations)**.** *For an accepted* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *carrier $L = (X, T, F, W, R, E, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real-name, topology-threshold, map, stream, regular-rational, lower-real, open-threshold, transport, replay, provenance, and local-name rows; threshold exactness for the finite lower-real rows in E; regular-rational readback through R; stream scheduling through W; componentwise* hsame *transport through H; displayed* Cont *replay through C; and provenance through P.*

Proof. Project the carrier of Definition 3935.1. The rows $X, T, F, W, R, E, O, H, C, P, N$ are the whole Real-and-Completion packet, so carrier admission is rowwise.

Read threshold exactness through E and O. A rational query $q < f(x)$ is accepted only after the source real name has been scheduled by W, read back through R, and matched to a finite lower-real epigraph row in E.

Transport and replay are then forced by H and C. They move the real-name, threshold, map, schedule, regular-rational, lower-real, and open-neighbourhood rows together, so the local NameCert surface cannot replace them by host topology or quotient equality.

Finally P and N package the same finite rows. Since the excluded completeness, trichotomy, compactness, semantic-liminf, and selected-limit principles are absent from the carrier, they are not NameCert obligations. □

Theorem 3935.3 (LowerSemicontinuityReal open-threshold route)**.** *Every completion-facing consumer of an accepted* $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ *packet must read an open lower-threshold query through the finite route*

$$X, T, F, W, R, E, O \longmapsto H, C, P, N.$$

The route may expose Real-name windows, regular-rational approximants, stream schedules, lower-real epigraph rows, topology threshold rows, map handoffs, transport, replay, provenance, and local naming. It exports no host real completeness, quotient equality, trichotomy, open-cover compactness, semantic liminf, or completed-space selection principle.

Proof. Start with a threshold query. The source is the $\mathsf{Real}^{\uparrow}$ row X, and the neighbourhood vocabulary is the displayed $\mathsf{Topology}^{\uparrow}$ row T, so the query does not begin from a host open-set lattice.

Next use the scheduling and readback rows. The stream schedule W selects the finite windows, and the regular-rational row R supplies the approximants used by the lower-real ledger E.

The open-threshold row O turns the lower-real epigraph witness into the basic threshold route exposed to completion-facing consumers. The map row F is consumed only when the query is read through a displayed $\mathsf{ContinuousMap}^{\uparrow}$ surface.

Finally replay through H, C, P, N. These rows transport, continue, source, and name the same finite threshold packet. Since no excluded host principle is a row of the carrier, the route remains a Real-and-Completion specialization of lower semicontinuity rather than a completeness or compactness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3935.1, Definition 3935.2, and Definition 3935.3 seed the finite Real-and-Completion lower-semicontinuity packet over real-name windows, regular-rational readbacks, stream schedules, lower-real epigraph rows, topology thresholds, map handoffs, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerSemicontinuityReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real completeness, quotient real equality, trichotomy, open-cover compactness, semantic liminf, completed-space selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold exactness, regular-rational readback, stream scheduling, topology exposure, transport replay, provenance, and non-escape for BEDC.Derived.LowerSemicontinuityRealUp.

Constructive story: A $\mathsf{LowerSemicontinuityReal}^{\uparrow}$ packet is generated from RealUp source histories, TopologyUp threshold rows, ContinuousMapUp handoffs, StreamNameUp schedules, RegSeqRatUp readbacks, LowerRealUp epigraph witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3936

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$

Following Bishop–Bridges regular Cauchy constructions, the $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ packet records the finite diagonal selector surface used by the Real and Completion route before a terminal limit representative is sealed. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the diagonal-limit representative boundary of chapter 1205. The packet names only a finite selector schedule, readback windows, tolerance rows, transport, replay, provenance, and local naming data; it does not export countable choice, host limits, quotient-stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3936.1 (Regular Cauchy limit selector carrier)**.** *A* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the selector schedule row, listing the finite diagonal stages and the requested precision levels.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming the inspected source entries at each stage of A.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the dyadic error budget attached to the selected stages.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the selected stream windows and dyadic tolerances into regular rational observations.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the selector schedule, stream windows, dyadic tolerances, and regular rational readback have been displayed.*
6. *H records componentwise* hsame *transport for A, W, D, R, E and for the replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite limit request through*

$$A, W, D, R \longmapsto E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, W, D, R, E, H, C, P, N$. The carrier has no coordinate for countable choice, selected host limits, quotient equality of stream names, hidden tail schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLimitSelectorUp.RegularCauchyLimitSelectorTasteGate single carrier alignment`

Theorem 3936.2 (Regular Cauchy limit selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ *carrier $S = (A, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the selector schedule,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every Real or Completion consumer of the selector reads the diagonal-limit handoff through these rows before using the representative surface of chapter 1205.*

Proof. Project the carrier of Definition 3936.1. The analytic rows are exactly the selector schedule A, finite stream window W, dyadic tolerance row D, regular rational readback R, and terminal seal E.

The selector route is finite. A consumer first reads the stages in A, then inspects only the windows named by W, then checks the dyadic tolerances recorded in D before any regular rational readback is available.

The readback row R is the only route from selected finite windows to the terminal seal E. Thus the Real-facing endpoint is reached through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ data before it can be consumed by the diagonal-limit representative boundary.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier contains no coordinate for countable choice, selected host limits, quotient-stream equality, hidden tail schedules, ambient Real completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitSelector}^{\uparrow}$ packet is generated from a finite selector schedule, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$, seed)

Scope closed: Definition 3936.1 and Definition 3936.2 seed the finite diagonal selector packet consumed by Real and Completion routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLimitSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, host limits, quotient-stream equality, hidden tail schedules, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, selector schedule exactness, StreamName window exposure, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLimitSelectorUp.

Chapter 3937

A Concrete Naming Certificate for SubordinateFiniteModulusCover$^{\uparrow}$

Following Lebesgue-number and Bishop–Bridges compact-uniform-continuity arguments, the SubordinateFiniteMod packet records the finite subordinate cover that feeds the uniform-modulus fold in chapter 9237. It sits over the compact finite-net carrier of chapter 129, the pointwise map ledger of chapter 130, the positive-radius lower-bound surface of chapter 452, and the downstream uniform-modulus consumer of chapter 369. The packet names only the finite cover rows, pointwise modulus rows, rational fold ledger, transport, replay, provenance, and local naming data; it does not export open-cover compactness, a selected infinite subcover, host minima, quotient real equality, or an external semantic modulus.

Definition 3937.1 (Subordinate finite modulus cover carrier)**.** *A* SubordinateFiniteModulusCover$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (K, F, L, Q, U, H, R, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, naming the centers and the source-distance coverage windows inspected by the packet.*
2. *F is the* ContinuousMap$^{\uparrow}$ *pointwise-modulus row, assigning a local modulus window to each displayed center.*
3. *L is the* FiniteLebesgueNumber$^{\uparrow}$ *subordinate-radius row, recording a positive radius below the cover radius for each displayed local modulus.*
4. *Q is the* Rat$^{\uparrow}$ *fold row that computes the finite lower-bound tolerance from the displayed subordinate radii.*
5. *U is the* UniformModulus$^{\uparrow}$ *handoff row, exposed only after K, F, L, Q have been displayed.*
6. *H records componentwise* hsame *transport for K, F, L, Q, U and for the replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from compact-net inspection through pointwise modulus rows, subordinate radii, rational folding, and uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* SubordinateFiniteModulusCover$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, Rat$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, F, L, Q, U, H, R, P, N$. The carrier has no coordinate for open-cover compactness, infinite subcover selection, host minimum extraction, quotient equality of real radii, ambient completeness, Lean verification, or bridge credentials.

Lean4 ✓ `BEDC.Derived.SubordinateFiniteModulusCoverUp.SubordinateFiniteModulusCoverTasteGate`
`single carrier alignment`

Theorem 3937.2 (Subordinate finite modulus cover NameCert obligations)**.** *For every accepted* $\mathsf{SubordinateFiniteModulusCover}^{\uparrow}$ *carrier $C = (K, F, L, Q, U, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}}$ *surface consists exactly of compact-net admission through K, pointwise-modulus admission through F, subordinate positive-radius exposure through L, finite rational lower-bound folding through Q, uniform-modulus handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3937.1. The analytic rows are the compact-net row K, the pointwise-modulus row F, the subordinate-radius row L, the rational fold Q, and the uniform-modulus handoff U.

The compact row K supplies only a finite bundle of centers and source-distance windows. The continuous-map row F is read at those displayed centers, and L records the positive subordinate radius attached to each local modulus row.

The fold row Q is a finite $\mathsf{Rat}^{\uparrow}$ computation over the displayed subordinate radii. It is not a host minimum over an open cover and cannot select an infinite subcover or appeal to quotient equality of real radii.

Finally U, H, R, P, N export, transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for open-cover compactness, infinite subcover choice, host minima, quotient real equality, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$.

Constructive story: A $\mathsf{SubordinateFiniteModulusCover}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, FiniteLebesgueNumber subordinate-radius rows, RatUp lower-bound folds, a UniformModulus handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$, seed)

Scope closed: Definition 3937.1 and Definition 3937.2 seed the subordinate finite-cover carrier consumed by compact-uniform-continuity routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SubordinateFiniteModulusCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, infinite subcover selection, host minimum extraction, quotient equality of real radii, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, pointwise-modulus admission, subordinate-radius exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.SubordinateFiniteModulusCoverUp.

Chapter 3938

A Concrete Naming Certificate for $\mathsf{LocatedIntermediateValue}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedIntermediateValue}^{\uparrow}$ packet records the finite bisection route used to locate a sign change on a real interval without selecting a root as host data. It sits over the real-name surface of chapter 38, the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance core of chapter 291, the located interval carrier of chapter 394, and the nested interval route of chapter 2249. The packet names only endpoint rows, located sign-apartness observations, finite bisection data, regular Cauchy readback, transport, replay, provenance, and local naming obligations; it does not export trichotomy, open-cover compactness, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3938.1 (Located intermediate value carrier)**.** *A* $\mathsf{LocatedIntermediateValue}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, S, B, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *E is the located interval endpoint row naming the two* $\mathsf{Real}^{\uparrow}$ *endpoint seals and their* $\mathsf{LocatedInterval}^{\uparrow}$ *order witness.*
2. *S is the located sign-apartness row recording finite negative and positive separation observations for the endpoint values.*
3. *B is the bisection ledger row listing the inspected dyadic midpoint stages, the retained subinterval side, and the sign row used at each stage.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the bisection ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected windows and dyadic tolerances into regular rational observations.*
6. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after E, S, B, W, R have been displayed.*
7. *H records componentwise* hsame *transport for E, S, B, W, R, Q and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located sign-change request through finite bisection and regular Cauchy readback.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntermediateValue}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{NestedIntervalSelection}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $E, S, B, W, R, Q, H, C, P, N$. The carrier has no coordinate for real trichotomy, open-cover compactness, quotient equality of real names, an externally selected zero, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.LocatedIntermediateValueUp.LocatedIntermediateValueTasteGate single carrier alignment`

Theorem 3938.2 (Located intermediate value NameCert obligations)**.** *For every accepted* LocatedIntermediateValu *carrier $I = (E, S, B, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$ *surface consists exactly of endpoint admission, located sign-apartness rows, bisection ledger exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3938.1. The analytic rows are the endpoint row E, the sign-apartness row S, the bisection ledger B, the finite window row W, the regular rational readback R, and the real seal Q.

The endpoint signs are read as located apartness data. They are not converted into a trichotomy principle, a global sign decision for every point, or a quotient equality test for real names.

The bisection ledger is finite. Each step records the dyadic midpoint, the inspected windows, and the retained side before the next row is read. The regular Cauchy readback R is the only route from these finite windows to the terminal $\mathsf{Real}^{\uparrow}$ seal Q.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the packet contains no coordinate for open-cover compactness, selected roots, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 3938.3 (Located intermediate value nested interval route)**.** *Every accepted* $\mathsf{LocatedIntermediateValue}^{\uparrow}$ *carrier exposes its intermediate-value read through the finite bisection ledger B, the* $\mathsf{NestedIntervalSelection}^{\uparrow}$ *dependency named in P, the finite window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance discipline, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q. A downstream consumer must read those rows before it may cite the packet as a located intermediate-value route.*

Proof. Read P to identify the nested interval dependency and the dyadic tolerance source. The only displayed interval refinement data in the carrier is the ledger B, so a consumer has no alternate route to a candidate interval.

Read W and R next. The windows are finite, and the regular rational readback is the handoff that turns the bisection ledger into a Cauchy-facing observation.

The endpoint of the route is the seal row Q, transported by H and replayed by C. The proof therefore uses finite bisection and regular Cauchy handoff only; it does not use trichotomy, open-cover compactness, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntermediateValue}^{\uparrow}$ packet is generated from located interval endpoint rows, located sign-apartness rows, a finite bisection ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3938.1, Definition 3938.2, and Definition 3938.3 seed the finite bisection packet consumed by constructive intermediate-value routes over located real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntermediateValue}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, open-cover compactness, quotient real equality, selected roots, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign-apartness exposure, bisection ledger exactness, finite StreamName windows, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntermediateValueUp.

Chapter 3939

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$

Following Bishop and Bridges on the fan theorem and constructive compactness of intervals, the $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ packet records the finite fan/bar data used to inspect a closed bounded interval through finite binary and dyadic prefixes. It consumes the closed bounded interval source of chapter 1591, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-coordinate seal of chapter 38. The packet names finite bar coverage and prefix stability only as displayed BEDC rows; it does not assert open-cover compactness, selected subcovers, countable choice, quotient equality of real names, trichotomy, or ambient Real completeness.

Definition 3939.1 (Closed bounded interval fan carrier)**.** *A* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, B, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *sealing.*
2. *T is the finite binary-prefix tree row used to enumerate dyadic subinterval decisions.*
3. *B is the finite bar row, recording that every inspected branch meets a displayed prefix in T.*
4. *W is the prefix-stability row saying that extension of a branch past a bar prefix preserves the associated interval containment decision.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint names, branch prefixes, and dyadic containment tests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the endpoint and prefix windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, T, B, W, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, tree, bar, stability, window, readback, and seal rows.*
9. *C records displayed* Cont *replay from interval-source inspection through finite prefix reads, bar coverage, regular rational readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$ *row.*

The carrier exports a finite fan/bar interface over the interval source. It has no coordinate for an open cover, a selected subcover, a choice function on branches, quotient equality of real names, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalFanUp.ClosedBoundedIntervalFanTasteGate single carrier alignment`

Theorem 3939.2 (Closed bounded interval fan NameCert obligations)**.** *For every accepted* ClosedBoundedIntervalF *carrier $F = (I, T, B, W, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, finite prefix-tree admission through T, finite bar exposure through B, prefix-stability exposure through W, finite observation through S, regular rational readback through R, Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3939.1. The interval source I is already the closed bounded interval packet of chapter 1591; the fan-specific rows are the finite tree T, finite bar B, and prefix-stability row W.

Read the observation and seal rows next. The $\mathsf{StreamName}^{\uparrow}$ window S and $\mathsf{RegSeqRat}^{\uparrow}$ readback R inspect endpoint and prefix data at finite precision, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of those finite rows.

The structural rows close the local certificate. Transport H, replay C, provenance P, and naming N preserve exactly this finite fan/bar route, so the obligations introduce no open-cover compactness, selected subcover, choice over branches, quotient equality, trichotomy, or ambient completeness. □

Theorem 3939.3 (Closed bounded interval fan finite-bar route)**.** *Every compactness-facing or uniform-modulus consumer of an accepted* $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto T, B, W$$

before it uses fan/bar data. The route first displays the closed bounded interval source, then finite stream windows and regular rational readbacks, then the Real seal, and only then exposes the finite prefix tree, finite bar, and prefix-stability rows. It supplies the finite bar interface needed by interval-net, compactness, and uniform-modulus consumers without claiming open-cover compactness, choice of a subcover, quotient real equality, trichotomy, or ambient completeness.

Proof. Start with the interval source row. By Definition 1591.3, the source I already displays endpoint order, containment ledgers, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ sealing, transport, replay, provenance, and naming.

Use Definition 3939.2. It places the finite prefix tree T, finite bar B, and prefix-stability row W inside the same accepted carrier, downstream of the interval source and the finite readback route.

Thus a consumer receives fan data only as a finite BEDC route over an accepted closed bounded interval. Since the carrier has no open-cover coordinate, no branch selector, no quotient-real equality row, no trichotomy row, and no ambient completeness row, the finite-bar route cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalFan}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, finite binary-prefix tree rows, finite bar rows, prefix-stability rows, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3939.1, Definition 3939.2, and Definition 3939.3 seed the finite fan/bar surface for closed bounded interval compactness and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalFan}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, selected subcovers, countable choice, quotient real equality, trichotomy, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix-tree admission, finite bar exposure, prefix-stability exposure, StreamName readback, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalFanUp.

Chapter 3940

A Concrete Naming Certificate for $\mathsf{DiagonalSequence}^{\uparrow}$

Following Cantor and Bishop–Bridges constructive analysis, the $\mathsf{DiagonalSequence}^{\uparrow}$ packet records the finite diagonal-sequence surface used by Real-facing diagonal arguments. It sits over the stream-row discipline of chapter 126, the dyadic bit ledger of chapter 291, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. It is consumed downstream by chapter 3793 and **??**. The packet names the reusable diagonal schedule and readback route; it is not a host countability theorem, a quotient-stream principle, or a completed powerset construction.

Definition 3940.1 (Diagonal sequence carrier)**.** *A* $\mathsf{DiagonalSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, W, B, F, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the inspected enumeration surface, exposing only finite* $\mathsf{StreamName}^{\uparrow}$ *candidate windows.*
2. *W is the diagonal schedule row, pairing each inspected enumeration row with the coordinate at which the diagonal value is read.*
3. *B is the dyadic observation row, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any regular rational or Real-facing seal is exposed.*
4. *F is the finite flip row, assigning the displayed opposite dyadic value at each scheduled coordinate.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the same schedule and flip ledger.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, downstream of E, W, B, F, Q.*
7. *H records componentwise* hsame *transport for enumeration, schedule, dyadic observation, flip, regular readback, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an enumeration request through the scheduled dyadic read, flip, regular readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DiagonalSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the diagonal schedule is finite, the dyadic observation and flip rows share the same coordinate ledger, the regular readback reuses that ledger, and the Real seal is downstream of the displayed readback route.

Lean4 (definition) `BEDC.Derived.DiagonalSequenceUp.DiagonalSequenceUp`

Theorem 3940.2 (Diagonal sequence NameCert obligations)**.** *For every accepted* $\mathsf{DiagonalSequence}^{\uparrow}$ *carrier* $D = (E, W, B, F, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$ *surface consists exactly of enumeration-window admission through* E*, finite diagonal scheduling through* W*, dyadic observation through* B*, flip exactness through* F*, regular rational readback through* Q*, Real sealing through* R*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3940.1. The displayed inventory is exactly $E, W, B, F, Q, R, H, C, P, N$.

Read the construction rows in order. The enumeration surface E is inspected only along the finite schedule W, the dyadic observation B is made at those scheduled coordinates, and the flip row F assigns the opposite displayed dyadic value on the same ledger.

The readback rows are downstream. The row Q records regular rational readback from the scheduled flip ledger, while R seals the resulting sequence as $\mathsf{Real}^{\uparrow}$ data. Componentwise transport, displayed replay, provenance, and naming preserve those rows and introduce no extra coordinate. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 3940.3 (Diagonal sequence nonescape)**.** *Every accepted* $\mathsf{DiagonalSequence}^{\uparrow}$ *consumer must read the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed schedule W *and flip ledger* F *before it can expose a diagonal Real-facing value. The packet exports no host enumeration principle, quotient stream equality, unbounded choice of coordinates, powerset of reals, or completed uncountability theorem.*

Proof. Use Definition 3940.2. It orders the inspected stream windows, dyadic observations, flip ledger, regular readback, and Real seal inside one finite carrier.

The nonescape clauses follow from the absence of any other carrier coordinate. The rows H, C, P, N can transport, replay, source, and name the displayed route, but they cannot add a host enumeration theorem or a quotient equality principle. Thus every downstream diagonal packet must consume the same finite sequence route rather than bypass it. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiagonalSequence}^{\uparrow}$ packet is generated from finite StreamName windows, a diagonal schedule, DyadicRatCore observations, a flip ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3940.1, Definition 3940.2, and Definition 3940.3 seed the reusable finite diagonal-sequence packet consumed by Real-facing diagonal arguments.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiagonalSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host countability, quotient stream equality, unbounded coordinate choice, powerset constructions, Real uncountability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exactness, dyadic flip compatibility, regular readback, Real sealing, provenance exhaustion, and local naming for BEDC.Derived.DiagonalSequenceUp.

Chapter 3941

A Concrete Naming Certificate for $\mathsf{WobbleSaturationSeal}^{\uparrow}$

The $\mathsf{WobbleSaturationSeal}^{\uparrow}$ packet realizes the f_3 directional saturation route of Definition 9462.1423 as finite NameCert data. It consumes the six-coordinate codon window of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage packet of chapter 3470, the wobble quotient spectral trigger of chapter 2317, and the radial wobble spectrum of chapter 3681. The packet names the unique non-leaking f_3 saturation seal; it is not a biological stochastic model, quotient codon semantics, or an unrestricted support-closure oracle.

Definition 3941.1 (Wobble saturation seal carrier)**.** *A* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, C, Q, R, D, A, H, K, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row exposing the six Boolean coordinates of the inspected codon window.*
2. *T is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *row naming the boundary tile through which the wobble read is admitted.*
3. *C is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *row recording finite coverage of the relevant boundary support.*
4. *Q is the* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ *row exposing the quotient-facing spectral trigger.*
5. *R is the* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *row exposing the radial spectrum and wobble-edge accounting.*
6. *D is the directional saturation row, recording that the only non-leaking active-support saturation direction is f_3.*
7. *A is the* ACG *seal row, recording the median, symmetry, spectral, and final f_3 sealing corner.*
8. *H records componentwise* hsame *transport for window, boundary-tile, coverage, quotient-trigger, radial-spectrum, saturation, seal, replay, provenance, and naming rows.*
9. *K records displayed* Cont *replay from the six-bit window through boundary coverage, quotient trigger, radial spectrum, and f_3 seal.*
10. *P is the* Pkg *provenance over* $\mathsf{WobbleSaturationSeal}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}$ *row.*

The classifier accepts a packet only when the f_3 saturation row is read after the six-bit window, boundary-tile, coverage, quotient-trigger, and radial-spectrum rows, and when the ACG *seal is the displayed terminal corner for that same route.*

Lean4 ✓ `BEDC.Derived.WobbleSaturationSealUp.WobbleSaturationSealTasteGate single carrier alignment`

Theorem 3941.2 (Wobble saturation seal NameCert obligations)**.** *For every accepted* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *carrier $S = (W, T, C, Q, R, D, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}$ *surface consists exactly of six-bit window admission through W, boundary-tile admission through T, six-boundary coverage through C, quotient spectral triggering through Q, radial wobble-spectrum exposure through R, unique f_3 directional saturation through D,* ACG *seal exposure through A, componentwise* hsame *transport through H, displayed* Cont *replay through K,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3941.1. The rows $W, T, C, Q, R, D, A, H, K, P, N$ are the complete displayed inventory.

Read the support rows first. The six-bit window, boundary-tile, and coverage rows identify the finite codon support surface before any wobble saturation is visible.

Read the wobble rows next. The quotient spectral trigger Q and radial spectrum R supply the two sibling wobble surfaces, and D records the unique non-leaking f_3 direction over that shared surface. The row A then exposes ACG as the terminal seal for the same route.

Finally H, K, P, N transport, replay, source, and name exactly those rows. Since the carrier has no stochastic, quotient-semantic, or unrestricted support-closure coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3941.3 (Wobble saturation seal nonescape)**.** *Every accepted* $\mathsf{WobbleSaturationSeal}^{\uparrow}$ *consumer must read* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$*, and* $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ *before the unique f_3 saturation seal can be exposed. The seal exports no biological stochastic model, quotient codon semantics, unrestricted support closure, or route that treats a leaking direction as equivalent to the displayed f_3 row.*

Proof. Apply the obligation theorem. It places the six-bit window, boundary tile, coverage, quotient trigger, and radial spectrum before the directional saturation row.

The nonescape claim is then a carrier-inventory statement. The rows D and A name the unique f_3 saturation route and the terminal ACG seal, while H, K, P, N preserve the same route. No row remains from which to read a stochastic model, quotient codon semantics, or an unrestricted closure oracle. □

Falsifiable prediction: An accepted $\mathsf{WobbleSaturationSeal}^{\uparrow}$ consumer that exposes the f_3 saturation seal without first displaying CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WobbleQuotientSpectralTrigger, RadialWobbleSpectrum, hsame, Cont, Pkg, and local NameCert rows refutes the carrier surface.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, wobble-quotient-spectral-trigger, radial-wobble-spectrum: $\mathsf{WobbleSaturationSeal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six-bit window, boundary-tile, coverage, quotient-trigger, radial-spectrum, unique f_3 saturation, terminal ACG seal, transport, replay, provenance, and local naming rows.

The Lean TasteGate route for $\mathsf{WobbleSaturationSeal}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ WobbleSaturationSeal}^{\uparrow}$ and $\mathsf{FieldFaithful\ WobbleSaturationSeal}^{\uparrow}$ instances preserving $W, T, C, Q, R, D, A, H, K, P, N$ as distinguishable fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{WobbleSaturationSeal}^{\uparrow}$ packet is generated from six-bit codon windows, boundary-tile rows, six-boundary coverage, quotient spectral trigger rows, radial wobble-spectrum rows, a unique f3 saturation direction, the ACG seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3941.1, Definition 3941.2, and Definition 3941.3 seed the finite NameCert seal for the unique f_3 directional saturation route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WobbleSaturationSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological stochastic dynamics, quotient codon semantics, unrestricted support closure, Lean verification, bridge checking, or saturation beyond the displayed f3 route.

Upgrade path: Obligation closure requires checked carrier admission, six-bit window alignment, boundary-tile coverage, quotient-trigger exactness, radial-spectrum compatibility, unique f3 saturation, ACG sealing, Nontrivial WobbleSaturationSealUp, and FieldFaithful WobbleSaturationSealUp for BEDC.Derived.WobbleSaturationSealUp.

Chapter 3942

A Concrete Naming Certificate for $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ packet records the finite witness surface by which two regular Cauchy reads with the same displayed tail data are forced to have the same terminal Real-facing seal. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38; the packet stores only paired tail windows, dyadic witnesses, regular readbacks, seal comparisons, transport, replay, provenance, and local naming.

Definition 3942.1 (Regular Cauchy limit uniqueness carrier)**.** *A* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the two regular Cauchy source rows whose finite tails are being compared.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger witnessing the common scale at which the two source rows are inspected.*
3. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window naming the finite tail entries used in the comparison.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two inspected tails.*
5. *E is the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the dyadic, stream-window, and regular-rational rows have been displayed.*
6. *H records componentwise* hsame *transport for source rows, dyadic tolerance, finite windows, regular readbacks, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay for any finite uniqueness request through L, R, D, W, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, R, D, W, Q, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.RegularCauchyLimitUniquenessUp.RegularCauchyLimitUniquenessTasteGate
single carrier alignment

Theorem 3942.2 (Regular Cauchy limit uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniquen}}$ *surface consists exactly of carrier admission for the paired regular Cauchy source rows, dyadic witness ledger, finite stream windows, regular rational readbacks, terminal Real seal comparison, componentwise transport, displayed replay, provenance, and local naming. No uniqueness obligation is stored outside these rows.*

Proof. Project the accepted carrier through Definition 3942.1. The analytic rows are exactly the two source rows L, R, the dyadic witness D, the finite stream window W, the regular rational readback Q, and the terminal seal comparison E.

The structural rows are H, C, P, N. They transport, replay, source, and name the same finite packet, so the NameCert surface cannot add a quotient equality, selected limit, host real equality, choice selector, trichotomy, or ambient completeness coordinate. □

Theorem 3942.3 (Regular Cauchy limit uniqueness dyadic witness determinacy)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, the dyadic witness row* D *determines the displayed comparison scale used by the paired stream window* W *and regular rational readback* Q*. Any accepted uniqueness read must therefore inspect the same finite dyadic tolerance before the terminal seal row* E *is available.*

Proof. Begin with the dyadic tolerance ledger. The row D is the only coordinate whose classifier names the comparison scale, and the window row W is admitted only as a finite inspection at that displayed scale.

The regular rational readback Q consumes the same finite window. Hence the terminal seal row E is downstream from the dyadic witness and cannot be read from an independent scale, selected subsequence, quotient stream, or host equality witness. □

Theorem 3942.4 (Regular Cauchy limit uniqueness Real seal nonescape)**.** *For every accepted* $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ *carrier* $U = (L, R, D, W, Q, E, H, C, P, N)$*, every terminal* $\mathsf{Real}^{\uparrow}$ *seal comparison factors through the route*

$$L, R \to D \to W \to Q \to E \to H, C, P, N.$$

The exported uniqueness surface contains no quotient stream equality, no host real equality, no selected limit, no countable-choice schedule, no trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start from a terminal seal comparison. The carrier exposes the two source rows only through L, R, and the comparison scale is fixed by D before any finite stream entries can be inspected.

The window and readback rows W, Q then force the comparison through displayed finite tails and regular rational data. The Real-facing endpoint E is therefore a seal row for that already displayed finite route, not an independent completed real.

Finally H, C, P, N preserve and name the same packet. Since the carrier has no coordinate for quotient stream equality, host equality, selected limits, choice, trichotomy, or completeness, the terminal seal comparison cannot escape the listed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}$ packet is generated from paired regular Cauchy source rows, a shared dyadic tolerance ledger, finite StreamName windows, RegSeqRat readbacks, a paired RealUp seal row, componentwise transport, displayed continuation replay, package provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3942.1, Definition 3942.2, Definition 3942.3, and Definition 3942.4 seed the finite-window uniqueness surface for regular Cauchy limit reads over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLimitUniqueness}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyLimitUniquenessUp.RegularCauchyLimitUniquenessTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, trichotomy, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic witness determinacy, Real-seal nonescape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 3943

A Concrete Naming Certificate for $\mathsf{FormalBallBasis}^{\uparrow}$

Bishop-style constructive metric-basis language presents completion data through located balls, positive rational radii, and finite cover-refinement witnesses before any quotient of Cauchy names is read. The $\mathsf{FormalBallBasis}^{\uparrow}$ packet records that basis-level substrate between the metric surface of chapter 57, the dyadic arithmetic core of chapter 291, the real-name seal of chapter 38, the formal-ball packet of chapter 1510, and the metric-completion route of chapter 1506. It is a finite source for L10 completion and Cauchy-filter consumers, not an open-cover compactness theorem, not a quotient-ball construction, and not an ambient completeness principle.

Definition 3943.1 (Formal-ball basis carrier)**.** *A* $\mathsf{FormalBallBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, d, c, r, I, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row that supplies the inspected points and distance requests.*
2. *d is the metric-distance ledger read through the same* $\mathsf{Met}^{\uparrow}$ *surface.*
3. *c is the finite centre schedule, listing the displayed centres used by the basis rows.*
4. *r is the positive radius ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, recording dyadic tolerances and their terminal real-name seals.*
5. *I is the incidence row recording which displayed point requests enter which formal balls.*
6. *R is the refinement row, recording finite ball-containment and radius-shrink requests over* $\mathsf{FormalBall}^{\uparrow}$ *before any completion consumer is reached.*
7. *H records componentwise* hsame *transport for the metric, centre, radius, incidence, refinement, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric requests through centres, radii, formal-ball incidence, refinement, and completion-facing handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{FormalBall}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient balls, selected centres beyond c, selected Cauchy limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallBasisUp.FormalBallBasisTasteGate single carrier alignment`

Theorem 3943.2 (Formal-ball basis NameCert obligations)**.** *For an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *carrier*

$$B = (X, d, c, r, I, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-row exposure through X, d, centre-schedule exposure through c, positive-radius exposure through r, formal-ball incidence through I, finite refinement through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. Every basis read is exhausted by these rows.*

Proof. Project the carrier of Definition 3943.1. The displayed rows are exactly the metric carrier, metric ledger, centre schedule, radius ledger, incidence row, refinement row, transport row, replay row, provenance row, and local name.

The metric and radius rows supply the source data. The row X provides the inspected metric points and d provides the finite distance requests, while r records only dyadic and real-name radius data.

The formal-ball rows are then I and R. Incidence and refinement are read after the centres and radii have been displayed, so no consumer can substitute quotient balls or a selected completion point for the basis rows.

Finally H, C, P, N preserve, replay, source, and name the same finite packet. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 3943.3 (Formal-ball basis metric-completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *packet must read the displayed route*

$$X, d \longmapsto c, r \longmapsto I, R \longmapsto H, C, P, N$$

before completion data is available. Thus the completion handoff sees a finite basis of metric centres, dyadic and real radii, formal-ball incidence, and formal-ball refinement, and it receives no quotient ball carrier, selected Cauchy limit, open-cover compactness witness, selected centre outside c, or ambient completeness row.

Proof. Start with the NameCert obligations of Definition 3943.2. They force every consumer to read the metric rows X, d, the centre and radius rows c, r, and the formal-ball rows I, R before any structural row is used.

The provenance row names $\mathsf{MetricCompletion}^{\uparrow}$ only as a downstream consumer of the same route. Therefore completion-facing reads inherit the displayed formal-ball basis rather than a quotient of Cauchy names or a host completion object.

Transport and replay through H and C preserve the route, while P and N only source and name it. The excluded compactness, quotient-ball, selected-centre, selected-limit, and ambient-completeness objects are absent from the carrier. □

Theorem 3943.4 (Formal-ball basis Cauchy-filter refinement)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallBasis}^{\uparrow}$ *packet must read the metric row X, d, the centre schedule c, the dyadic and real radius ledger r, the incidence row I, and the refinement row R before completion-facing data is available. The consumer route factors through* $\mathsf{FormalBall}^{\uparrow}$ *and*

$\mathsf{MetricCompletion}^{\uparrow}$ *using only H, C, P, N for transport, replay, provenance, and local naming, and it receives no quotient filter, selected limit, countable-choice schedule, or ambient completeness row.*

Proof. Project the accepted carrier $B = (X, d, c, r, I, R, H, C, P, N)$ of Definition 3943.1. The NameCert obligations of Definition 3943.2 expose the metric row, centre schedule, radius ledger, incidence row, and finite refinement row as the whole basis surface.

The Cauchy-filter consumer therefore starts from the same displayed ball data as the metric-completion consumer. The formal-ball incidence row I names the inspected balls, and the refinement row R supplies the finite containment and radius-shrink reads before any completion-facing handoff can be consumed.

Now route through the existing handoff of Definition 3943.3. The rows H, C, P, N preserve and name the route from metric data through $\mathsf{FormalBall}^{\uparrow}$ into $\mathsf{MetricCompletion}^{\uparrow}$, so a $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ reader receives the finite formal-ball refinement corridor rather than a quotient filter or a chosen limit.

Since the carrier has no coordinate for quotient filters, selected Cauchy limits, countable choice, or ambient completeness, those objects cannot be exported by the refinement read. □

Falsifiable prediction: If a completion-facing read uses a formal-ball basis without displaying the metric row, centre schedule, dyadic or real radius ledger, incidence row, and refinement row, it is not a $\mathsf{FormalBallBasis}^{\uparrow}$ packet.

Independence witness: Nearest siblings: formalball, metriccompletion, bishopregularcauchy-completion, cauchyfiltercompletioncriterion, dyadicratcore; FormalBallBasisUp is not bijective to them because it binds a finite centre-radius basis and refinement row between metric balls and completion consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalBallBasis}^{\uparrow}$ packet is generated from a displayed MetricSpace row, metric-distance ledger, finite centre schedule, DyadicRatCore and Real radius rows, FormalBall incidence, finite refinement, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3943.1, Definition 3943.2, and Definition 3943.3 seed the finite basis-level substrate between formal balls and metric completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalBallBasis}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FormalBallBasisUp.FormalBallBasisTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient balls, selected centres beyond the displayed rows, selected Cauchy limits, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, centre and radius visibility, formal-ball incidence, refinement transport, replay stability, and metric-completion handoff non-escape for BEDC.Derived.FormalBallBasisUp.

Chapter 3944

A Concrete Naming Certificate for $\mathsf{RealConvergentSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealConvergentSequence}^{\uparrow}$ packet records displayed convergence of a real sequence to a named $\mathsf{Real}^{\uparrow}$ limit as finite NameCert data. It adjoins the L10 Real and completion route by sitting over the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the finite windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the real-name seal of chapter 38, the real sequence limit laws of chapter 1480, and the Cauchy convergence criterion surface of chapter 1509. The packet separates convergence evidence from Cauchy completeness: it names a displayed source sequence, a displayed limit row, and the finite tolerance route witnessing convergence, but it does not assert that every Cauchy stream converges.

Definition 3944.1 (Real convergent sequence carrier)**.** *A* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, L, D, M, W, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the sequence-source row, exposing the real-sequence index discipline, the finite observation requests, and the source-side* $\mathsf{StreamName}^{\uparrow}$ *handles.*
2. *L is the candidate* $\mathsf{Real}^{\uparrow}$ *limit row, carrying the named real limit, its terminal real-name seal, and the limit-side* $\mathsf{StreamName}^{\uparrow}$ *handles.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested tolerance, the comparison tolerance, and the finite split budget used by each convergence read.*
4. *M is the regular-Cauchy modulus row, storing the tail threshold assigned to each dyadic tolerance request before any consumer reads the convergence classifier.*
5. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *window row for source observations after the threshold and for the candidate-limit windows used in the same request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source windows, limit windows, and tolerance comparisons opened by W and D.*
7. *H records componentwise* hsame *transport for the source row, candidate limit, dyadic ledger, regular-Cauchy modulus, windows, readback, replay row, provenance row, and local naming row.*

8. *K records displayed* Cont *replay from a convergence query through the dyadic tolerance request, the regular-Cauchy threshold, the selected windows, the rational readback, and the candidate* $\mathsf{Real}^{\uparrow}$ *limit row.*

9. *P is the* Pkg *provenance over* $\mathsf{RealConvergentSequence}^{\uparrow}$, $\mathsf{RealSequenceLimit}^{\uparrow}$, $\mathsf{RealSequenceLimitLaws}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, L, D, M, W, R, H, K, P, N$. It has no coordinate for Cauchy completeness, selected limits for arbitrary Cauchy streams, quotient-stream equality, countable choice, trichotomy, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RealConvergentSequenceUp.RealConvergentSequenceTasteGate single carrier alignment`

Theorem 3944.2 (Real convergent sequence NameCert obligations)**.** *For every accepted* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier*

$$C = (S, L, D, M, W, R, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source sequence row, candidate* $\mathsf{Real}^{\uparrow}$ *limit, dyadic tolerance ledger, regular-Cauchy modulus, finite windows, regular rational readback, transport, replay, provenance, and local naming rows;*

2. *source-limit exposure, requiring S and L to be the only source of the displayed sequence and the displayed limit used by the convergence classifier;*

3. *tolerance and modulus exactness, requiring every convergence request to read its dyadic tolerance from D and its tail threshold from M before any source observation is compared with the limit;*

4. *window and readback alignment, requiring W and R to expose only the source windows and limit windows selected by the same tolerance request;*

5. *limit-law compatibility, requiring uses of real sequence limit laws to consume only the displayed rows S, L, D, M, W, R and never a second limit surface;*

6. *Cauchy-criterion boundary, requiring any later Cauchy consumer to receive convergence evidence as input rather than a completeness principle;*

7. *componentwise transport, displayed replay, provenance, and local naming through H, K, P, N;*

8. *refusal of Cauchy completeness, arbitrary limit selection, quotient-stream equality, countable choice, trichotomy, Lean verification, and bridge checking as local certificate data.*

No real-convergent-sequence obligation is stored outside these rows.

Proof. Project the carrier of Definition 3944.1. Its visible coordinates are the source sequence row S, candidate limit row L, dyadic tolerance ledger D, regular-Cauchy modulus M, window row W, readback row R, and structural rows H, K, P, N.

The convergence request first reads D and M. The dyadic row fixes the requested tolerance and comparison budget, while the modulus row fixes the tail threshold for source observations.

The selected observations are then opened through W and R. The same tolerance request chooses both the source windows and the candidate-limit windows, and the regular rational row reads back only those scheduled observations.

Limit-law compatibility follows because every law-facing read has access only to S, L, D, M, W, R before the structural rows. A second limit surface or host equality proof is not a carrier coordinate.

Finally, H, K, P, N transport, replay, record, and name the same finite route. Since the excluded principles do not occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 3944.3 (Real convergent sequence L10 route)**.** *Every accepted* $\mathsf{RealConvergentSequence}^{\uparrow}$ *carrier* $C = (S, L, D, M, W, R, H, K, P, N)$ *presents convergence evidence through the finite route*

$$S, L \longmapsto D \longmapsto M \longmapsto W \longmapsto R \longmapsto H, K, P, N.$$

Any Cauchy convergence criterion consumer downstream of C *receives only the displayed source sequence, candidate* $\mathsf{Real}^{\uparrow}$ *limit, dyadic tolerance ledger, regular-Cauchy threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Start with a downstream consumer. The carrier has no direct completeness coordinate, so the consumer must read the displayed convergence data from S and L.

The tolerance request then passes through D and M. These rows determine the comparison budget and the tail threshold before any observation window is opened.

The windows and rational observations are supplied by W and R. They expose only the scheduled source and candidate-limit data selected by the same request.

The structural rows H, K, P, N preserve the route for transport, replay, provenance, and naming. Thus every downstream criterion-facing read factors through the displayed finite route and cannot export Cauchy completeness, selected arbitrary limits, quotient-stream equality, countable choice, trichotomy, Lean evidence, or bridge data. □

Falsifiable prediction: Every $\mathsf{RealConvergentSequence}^{\uparrow}$ consumer read factors through a displayed source sequence, candidate $\mathsf{Real}^{\uparrow}$ limit, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, regular-Cauchy modulus, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, real-sequence-limit-laws, cauchy-convergence-criterion: $\mathsf{RealConvergentSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source sequence, candidate Real limit, dyadic tolerance ledger, regular-Cauchy modulus, scheduled source-and-limit windows, regular rational readback, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealConvergentSequence}^{\uparrow}$ packet is generated from a displayed source sequence, a candidate RealUp limit, a DyadicRatCore tolerance ledger, a regular-Cauchy modulus, StreamName windows, RegSeqRat readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$, seed)

Scope closed: Definition 3944.1, Definition 3944.2, and Definition 3944.3 seed the finite route from displayed real-sequence convergence evidence to later Cauchy-criterion consumption without asserting completeness.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealConvergentSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Cauchy completeness, selected limits for arbitrary Cauchy streams, quotient-stream equality, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-limit exposure, tolerance and modulus exactness, window and readback alignment, limit-law compatibility, Cauchy-criterion boundary, transport, replay, provenance, and local naming for BEDC.Derived.RealConvergentSequenceUp.

Chapter 3945

A Concrete Naming Certificate for $\mathsf{LocatedCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ packet records the finite route by which located regular rational windows expose a Cauchy subsequence handoff to the Real and Completion corridor. It consumes the located tail-window discipline of chapter 2376, the finite extraction surface of chapter 1854, the cofinal regular-subsequence stability row of chapter 3515, the stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a displayed subsequence extraction surface for located regular names; it is not a selected infinite subsequence, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3945.1 (LocatedCauchySubsequence carrier)**.** *A* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, R, D, Q, I, S, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger, listing the source windows and selected subsequence windows inspected by the packet.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the located regular rational source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by Cauchy and locatedness requests.*
4. *Q is the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison row, exposing located tail windows and finite cut comparisons.*
5. *I is the subsequence-index row, recording the monotone selected indices and the source windows from which they are read.*
6. *S is the* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *stability row, binding the extracted windows to the Cauchy tolerance schedule.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, R, D, Q, I, S have been displayed.*
8. *H records componentwise* hsame *transport for the window, readback, tolerance, located-comparison, index, extraction, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the same finite route.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, Q, I, S, E, H, C, P, N$. It has no coordinate for a choice-selected infinite subsequence, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchySubsequenceUp.LocatedCauchySubsequenceTasteGate single carrier alignment`

Theorem 3945.2 (LocatedCauchySubsequence NameCert obligations)**.** *For every accepted* LocatedCauchySubsequ *carrier*

$$L = (W, R, D, Q, I, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$ *surface consists exactly of five obligation families: carrier admission for $W, R, D, Q, I, S, E, H, C, P, N$; located-window exposure through the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *row Q; subsequence extraction stability through the index row I and extraction row S;* $\mathsf{Real}^{\uparrow}$ *handoff through the seal row E; and refusal of selected subsequences, quotient stream equality, countable choice, trichotomy, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3945.1. The packet displays exactly the stream windows, regular rational readback, dyadic tolerance ledger, located comparison row, subsequence-index row, extraction row, seal row, transport, replay, provenance, and local naming.

Located-window exposure is inherited from chapter 2376. A location request must pass through Q after reading the finite windows W, the regular rational row R, and the dyadic tolerance ledger D.

Subsequence extraction stability is the pair of rows I, S. The index row states which source windows are selected, and the extraction row states that the selected windows meet the displayed Cauchy tolerance schedule. Componentwise transport H and replay C preserve that route without adding a new selector.

The real-facing handoff is terminal at E. Since E is only reachable after $W, R, D, Q, I, S, H, C, P, N$ have been displayed, it exports a finite located subsequence seal and not a quotient completion or ambient completeness principle.

The classifier has no remaining coordinate beyond the listed rows. Thus selected subsequences, quotient stream equality, countable choice, trichotomy, host real equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are refused by the carrier surface itself. □

Theorem 3945.3 (LocatedCauchySubsequence located-window exposure)**.** *Every accepted* LocatedCauchySubsequ *carrier exposes a located subsequence window only through the finite route*

$$W, R, D, Q, I, S, H, C, P, N.$$

In particular, the subsequence-index row cannot be consumed before the source $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison row have been displayed.*

Proof. Project the rows of Definition 3945.1. A located subsequence request first names a source observation window in W, reads the rational source through R, and consults the dyadic tolerance ledger D.

The located comparison row Q then decides the displayed tail-window comparison. Only after that row is present may the index row I and extraction row S expose the selected subsequence window.

Transport, replay, provenance, and local naming preserve the same route. No row in the packet allows I to be read independently of W, R, D, Q. □

Theorem 3945.4 (LocatedCauchySubsequence RealUp handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ *carrier factors through the located regular source, subsequence-index row, and extraction-stability row before reading the seal row. The handoff exposes only finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *comparison rows, subsequence-index rows,* $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ *stability rows, transport, replay, provenance, and local naming.*

Proof. Begin with a Real-facing read. The seal row E is downstream of the source windows W, readback R, dyadic ledger D, located comparison row Q, index row I, and extraction row S.

Apply Definition 3945.2. The obligation surface forces every consumer-visible read to pass through those finite rows before the seal can be exposed.

The remaining structural rows only transport and replay the same displayed data. Therefore the handoff is a finite BEDC route from located regular rational windows to a terminal $\mathsf{Real}^{\uparrow}$ seal, not an ambient completion theorem. □

Falsifiable prediction: Every $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ read factors through displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, $\mathsf{LocatedRegSeqRat}^{\uparrow}$ comparison rows, subsequence-index rows, $\mathsf{CauchySubsequenceExtraction}^{\uparrow}$ stability rows, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locatedregseqrat, cauchysubsequenceextraction, regularcauchysubsequencestability, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located tail-window comparisons, monotone selected indices, extraction stability, finite stream windows, regular rational readback, dyadic tolerances, Real seal routing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchySubsequence}^{\uparrow}$ packet is generated from StreamName source and subsequence windows, RegSeqRat readback, DyadicRatCore tolerance rows, LocatedRegSeqRat comparison rows, subsequence-index rows, CauchySubsequenceExtraction stability rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$, seed)

Scope closed: Definition 3945.1, Definition 3945.2, Definition 3945.3, and Definition 3945.4 seed the finite located Cauchy subsequence route from displayed regular rational windows to Real-facing handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedCauchySubsequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected infinite subsequences, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-window exposure, subsequence extraction stability, RealUp handoff, and nonescape for BEDC.Derived.LocatedCauchySubsequenceUp.

Chapter 3946

A Concrete Naming Certificate for $\mathsf{CompletionReflectorTriangle}^{\uparrow}$

Following Mac Lane's adjunction triangle identities and the Bishop–Bridges construction of Cauchy completion, the $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ packet records the finite BEDC rows needed to read the unit, counit, and idempotent replay of the L10 completion reflector as one public triangle. It depends on the reflector packet of chapter 1585, the unit packet of chapter 1527, the counit packet of chapter 1539, the idempotent packet of **??**, the universal-property packet of chapter 397, and the reflection packet of chapter 403. Its real-facing route is read through chapter 360, chapter 126, chapter 301, chapter 291, and chapter 38. The packet is a finite naming certificate for the triangle surface; it is not quotient completion equality, not a selected limit, not host equality, not countable choice, not ambient completeness, and not detached $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3946.1 (CompletionReflectorTriangle carrier)**.** *A* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, U, M, Q, I, E, R, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source row for the displayed metric, regular-Cauchy, or real-name data entering the completion reflector.*
2. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *row admitting the source into the completion-facing surface.*
3. *M is the completion-object row, read through the* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *and reflector surfaces rather than through a host quotient carrier.*
4. *Q is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *row returning already completed or real-sealed reads through the displayed route.*
5. *I is the* $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$ *replay row for a second completion pass over the same completion-object row.*
6. *E is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *extension row consumed after the unit face is visible.*
7. *R is the* $\mathsf{CompletionReflection}^{\uparrow}$ *reflection row consumed after the unit, counit, idempotent, and extension faces are visible.*
8. *T is the local transport row through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *windows.*

9. *H records componentwise* hsame *transport for the source, unit, completion object, counit, idempotent replay, extension, reflection, transport, continuation, provenance, and naming rows.*
10. *C records displayed* Cont *replay along the unit-counit-idempotent triangle and through the local real-name transport row.*
11. *P is the* Pkg *provenance over* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$ $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, $\mathsf{RealCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *row.*

The classifier on accepted packets compares only these twelve rows under the displayed transport and replay data.

Lean4 ✓ `BEDC.Derived.CompletionReflectorTriangleUp.CompletionReflectorTriangleTasteGate single carrier alignment`

Theorem 3946.2 (CompletionReflectorTriangle NameCert obligations)**.** *For every accepted* CompletionReflectorTr *carrier $K = (S, U, M, Q, I, E, R, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source, unit, completion-object, counit, idempotent-replay, extension, reflection, transport, continuation, provenance, and naming rows; unit-first readability for completion-facing source reads; counit return for already completed or real-sealed reads; idempotent replay over the same completion-object row; extension and reflection consumption over the displayed triangle route; and non-escape from quotient completions, selected limits, host equality, countable choice, ambient completeness, and detached* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3946.1. The packet displays $S, U, M, Q, I, E, R, T, H, C, P, N$ and no other row.

The unit face reads S through U before the completion object M is consumed. The counit face reads Q only for data already on the completed or real-sealed side, and the idempotent face reads I over the same M row.

The extension and reflection faces are downstream of that route. The row E is consumed as the finite universal-property extension, and R is consumed as the reflection row over the same S, U, M, Q, I, T surface.

Finally H, C, P, N transport, replay, source, and name exactly the displayed route. Since no coordinate stores quotient completions, selected limits, host equality, countable choice, ambient completeness, or detached real completeness, the stated non-escape boundary follows. □

Theorem 3946.3 (CompletionReflectorTriangle unit-counit coherence)**.** *Let $K = (S, U, M, Q, I, E, R, T, H, C, P, N$ be an accepted* $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ *carrier. The two triangle readings*

$$S \longmapsto U \longmapsto M \longmapsto I \longmapsto E, R \longmapsto H, C, P, N$$

and

$$S \longmapsto T \longmapsto Q \longmapsto I \longmapsto E, R \longmapsto H, C, P, N$$

meet at the same displayed idempotent, extension, reflection, transport-replay, provenance, and naming rows. Thus a consumer that first enters completion by the unit face and a consumer that first reads the real-name transport before counit return receive the same finite $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$ *boundary, with no quotient completion equality, selected limit, host equality, countable choice, ambient completeness, or detached* $\mathsf{Real}^{\uparrow}$ *completeness exported.*

Proof. Expose the carrier rows from Definition 3946.1. The only completion-facing rows are $S, U, M, Q, I, E, R, T, H, C, P, N$.

The first displayed reading is the unit leg: the source is admitted by U, the completion object M is read, and repeated completion is replayed by I before E and R are consumed.

The second displayed reading is the transport-counit leg: the source is read through T, returned by Q when already completed or real-sealed, and then replayed by the same I before E and R are consumed.

Both readings end in the same H, C, P, N rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row therefore identify the two readings as one finite triangle boundary. The excluded completion, selection, host-equality, choice, ambient-completeness, and detached-real-completeness data have no row in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionReflectorTriangle}^{\uparrow}$ packet is generated from a displayed source row, unit admission, completion-object access, counit return, idempotent replay, universal-property extension, reflection consumption, local StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3946.1, Definition 3946.2, and Definition 3946.3 seed the finite unit-counit-idempotent triangle surface for the L10 completion reflector.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionReflectorTriangle}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompletionReflectorTriangleUp.CompletionReflectorTriangleTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality, countable choice, ambient completeness, detached $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-counit coherence, idempotent replay, extension compatibility, reflection compatibility, local real-name transport, and non-escape for the same carrier.

The $\mathsf{LawvereMetricCompletion}^{\uparrow}$ hub exposes its sibling lattice against the dense metric-completion surface of chapter 3351 through the child route below.

Chapter 3947

A Concrete Naming Certificate for $\mathsf{LawvereMetricCompletion}^{\uparrow}$

Following Lawvere on generalized metric spaces and enriched-category distance, the $\mathsf{LawvereMetricCompletion}^{\uparrow}$ packet records the finite directed-distance profile surface through which an enriched metric completion can be consumed inside BEDC. It sits beside the Yoneda profile route of chapter 2148, the ordinary metric observation rows of chapter 57, the formal-ball approximation surface of chapter 1510, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a one-sided enriched completion boundary; it does not select host limits, quotient directed Cauchy objects, symmetric metric completions, global colimits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The sibling dependency route is substantive: $\mathsf{LawvereMetricCompletion}^{\uparrow}$ reads the $\mathsf{YonedaCompletion}^{\uparrow}$ profile route of chapter 2148, the $\mathsf{FormalBall}^{\uparrow}$ approximation cuts of chapter 1510, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any completion-facing directed profile can be consumed. The directed source row M is supplied by the $\mathsf{QuasiMetric}^{\uparrow}$ carrier route of chapter 2158 and is consumed before Yoneda profiles or FormalBall approximations.

Definition 3947.1 (Lawvere metric completion carrier)**.** *A* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, Y, F, D, A, T, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the directed* $\mathsf{QuasiMetric}^{\uparrow}$ *source row, carrying the visible point classifier and the finite one-sided distance comparisons accepted by the source packet.*
2. *Y is the* $\mathsf{YonedaCompletion}^{\uparrow}$ *handoff row, exposing the covariant and contravariant distance profiles retained for the displayed source points.*
3. *F is the finite probe-comparison row, listing the source probes against which two directed profiles are compared.*
4. *D is the directed-distance profile row, recording the left and right distance readings that make the completion candidate visible as a Lawvere-enriched profile.*
5. *A is the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row, displaying finite ball inclusions and radius cuts used before any completion-facing consumer reads D.*

6. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row for directed profile errors, ball radii, and probe budgets.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained regular rational windows.*
8. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row for directed distance values and zero-distance comparison readback.*
9. *H records componentwise* hsame *transport for source rows, profile rows, probe comparisons, formal-ball approximations, dyadic tolerances, regular readbacks, Real seals, provenance, and local naming.*
10. *C records displayed* Cont *replay from source distances to Yoneda profiles, from profiles to formal-ball approximations, and from tolerance windows to Real readback.*
11. *P is the* Pkg *provenance over* $\mathsf{LawvereMetricCompletion}^{\uparrow}$, $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Y, F, D, A, T, R, S, H, C, P, N$. It has no coordinate for selected Cauchy weights, quotient profile equality, symmetric completion collapse, host enriched colimits, countable choice, or ambient real completeness.

Theorem 3947.2 (Lawvere metric completion NameCert obligations)**.** *For every accepted* LawvereMetricCompletio *carrier $L = (M, Y, F, D, A, T, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$ *surface consists exactly of directed source admission, YonedaCompletion handoff, finite probe comparisons, directed distance-profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every enriched-metric completion consumer must inspect those rows before treating the packet as a completion neighbour.*

Proof. Project the carrier of Definition 3947.1. The source and Yoneda rows M, Y are the only rows that introduce directed distance profiles from the displayed metric source.

The finite comparison route is F, D, A, T. Probe comparisons fix which source probes are being tested, the directed profile row records the left and right readings, the formal-ball row supplies finite approximation cuts, and the dyadic row records the tolerance accounting for those cuts.

Readback and transport are then forced through R, S, H, C, P, N. The regular rational windows and Real seal expose only the retained directed distance values, while componentwise transport, replay, provenance, and local naming preserve the same finite route.

Since the classifier has no additional coordinate, selected Cauchy weights, quotient profile equality, symmetric completion collapse, host enriched colimits, countable choice, and ambient real completeness cannot be read from the packet. □

Theorem 3947.3 (Lawvere metric completion finite profile nonescape)**.** *Every completion-facing read exported by an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet factors through*

$$\mathsf{QuasiMetric}^{\uparrow},\ \mathsf{YonedaCompletion}^{\uparrow},\ \mathsf{FormalBall}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The read first fixes the directed source and Yoneda profile rows, then the finite probe comparisons and formal-ball approximations, then dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and local naming. It exports no quotient directed Cauchy object, selected profile limit, symmetric metric completion collapse, host enriched colimit, or ambient completeness principle.

Proof. Let $L = (M, Y, F, D, A, T, R, S, H, C, P, N)$ be accepted. A completion-facing consumer first receives M, Y, F, D, so the directed profile is tied to displayed source distances, Yoneda handoff, and finite probe comparisons.

The approximation stage is A, T. Formal-ball inclusions and dyadic tolerance rows determine which finite directed-distance errors are visible before readback.

Finally R, S, H, C, P, N expose only regular rational windows, Real seals, transport, replay, provenance, and local naming for that same route. The excluded quotient, choice, colimit, symmetric-collapse, and ambient-completeness objects are not carrier fields, so no completion consumer can bypass the finite profile surface. □

Theorem 3947.4 (Lawvere metric completion directed-profile composition)**.** *Every completion-facing read from an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet composes through the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile handoff,* $\mathsf{QuasiMetric}^{\uparrow}$ *directed source data,* $\mathsf{FormalBall}^{\uparrow}$ *approximation rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows before it is visible as a Lawvere-enriched completion profile. The composed read exports directed source distances, finite probe comparisons, formal-ball approximation cuts, dyadic error budgets, regular rational windows, Real sealing, transport, replay, provenance, and local naming, but it exports no symmetric metric completion, quotient directed Cauchy object, selected profile limit, global colimit, countable-choice schedule, or ambient completeness claim.*

Proof. Project the carrier of Definition 3947.1. The directed source and profile handoff are exactly the rows M, Y, F, D.

Apply Definition 3947.2. It forces any completion-facing consumer to inspect directed source admission, YonedaCompletion handoff, finite probe comparison, directed profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Apply Definition 3947.3. The nonescape theorem identifies the public route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The composed read is therefore the displayed directed-profile route itself. Since the carrier has no coordinate for a symmetric completion, quotient directed Cauchy object, selected profile limit, host colimit, choice schedule, or ambient completeness witness, none of those objects can be exported by the composition. □

Theorem 3947.5 (Lawvere metric completion QuasiMetric source handoff)**.** *Any* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *read whose source distance is one-sided must first pass through the* $\mathsf{QuasiMetric}^{\uparrow}$ *carrier route of chapter 2158 before the* $\mathsf{YonedaCompletion}^{\uparrow}$ *and* $\mathsf{FormalBall}^{\uparrow}$ *rows are consumed. The completion-facing consumer may then read directed source distances, Yoneda profiles, finite probe comparisons, formal-ball approximation cuts, dyadic tolerance accounting, rational readback, Real sealing, transport, replay, provenance, and local naming, but it receives no symmetric metric source, quotient directed Cauchy object, selected profile limit, choice schedule, or ambient completeness claim.*

Proof. Apply Definition 2158.3. A one-sided source distance reaches completion-facing consumers only through the $\mathsf{QuasiMetric}^{\uparrow}$ carrier rows for source admission, ordered point names, directed distance, zero diagonal, forward triangle, directed balls, Cauchy rows, uniform reflection, transport, replay, provenance, and local naming.

Apply Definition 3947.4. Once that directed source route is displayed, the Lawvere completion read composes it with $\mathsf{YonedaCompletion}^{\uparrow}$ profiles, finite probe comparisons, $\mathsf{FormalBall}^{\uparrow}$ approximations, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the $\mathsf{Real}^{\uparrow}$ seal.

Apply Definition 3947.3. The resulting completion-facing route remains finite and displayed, so it cannot export a quotient directed Cauchy object, selected profile limit, symmetric completion collapse, host colimit, choice schedule, or ambient completeness witness. □

Theorem 3947.6 (Lawvere metric completion FormalBall universality route)**.** *Any completion-facing read of an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *profile must factor through the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row A after the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row Y and before the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and* $\mathsf{Real}^{\uparrow}$ *seal row S. Thus the approximation gate cannot be replaced by quotient profile equality, symmetric completion collapse, selected profile limits, host enriched colimits, or ambient completeness.*

Proof. Project the carrier of Definition 3947.1. The profile route exposes $M, Y, F, D, A, T, R, S, H, C, P, N$, with A placed between the Yoneda-directed profile rows and the tolerance, readback, and seal rows.

Apply Definition 3947.2. It forces completion-facing consumers to inspect FormalBall approximation as one of the local NameCert obligations before the packet is treated as a completion neighbour.

Apply Definition 3947.3 and Definition 3947.4. Together they identify the public route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Since quotient profile equality, symmetric completion collapse, selected profile limits, host enriched colimits, and ambient completeness are absent carrier coordinates, no completion-facing read can substitute them for the FormalBall approximation gate. □

Theorem 3947.7 (Lawvere metric completion FormalBall profile initiality)**.** *Let $L = (M, Y, F, D, A, T, R, S, H, C,$ be an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet. Any completion-facing directed-profile read that reaches the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, or the* $\mathsf{Real}^{\uparrow}$ *sealing row S first factors through the* $\mathsf{FormalBall}^{\uparrow}$ *approximation row A after the* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row Y. Thus A is the initial completion-facing approximation gate after Yoneda profile data; no later tolerance, readback, or seal consumer may bypass it through quotient profile equality, selected profile limits, symmetric completion collapse, host enriched colimits, choice, or ambient completeness.*

Proof. Project the carrier of Definition 3947.1. The displayed order places Y before the finite probe and directed-profile rows, then places the FormalBall row A before the tolerance, readback, and seal rows T, R, S.

Apply Definition 3947.2. The NameCert obligations force every completion-facing consumer to inspect the directed source admission, YonedaCompletion handoff, finite probe comparison, directed profile admission, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming rows in the displayed carrier surface.

Apply Definition 3947.3 and Definition 3947.4. These theorems identify the finite route through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the directed profile cannot leave the packet before the FormalBall row is read.

Finally apply Definition 3947.6. It names A as the mandatory approximation row between Y and the later tolerance, readback, and seal rows. The carrier has no coordinates for quotient profile equality, selected profile limits, symmetric completion collapse, host enriched colimits, choice, or ambient completeness, so none of those objects can serve as an earlier completion-facing gate. □

Falsifiable prediction: Removing any one of the $\mathsf{YonedaCompletion}^{\uparrow}$ handoff, finite probe-comparison row, $\mathsf{FormalBall}^{\uparrow}$ approximation row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, or local $\mathsf{NameCert}$ row makes

Definition 3947.3 false: a completion consumer could no longer verify the displayed directed-profile route.

Independence witness: yonedacompletion, metriccompletion, formalball, dyadicratcore, regseqrat, real: $\mathsf{LawvereMetricCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries directed source distances, Yoneda profile handoff, finite probe comparisons, formal-ball approximations, dyadic tolerance accounting, regular rational readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LawvereMetricCompletion}^{\uparrow}$ packet is generated from directed QuasiMetricUp source data, YonedaCompletionUp profile handoff, finite probe comparisons, directed distance-profile rows, FormalBallUp approximations, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback windows, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3947.1, Definition 3947.2, and Definition 3947.3 seed the finite Lawvere-enriched completion route over directed profile, approximation, tolerance, readback, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawvereMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected Cauchy weights, quotient profile equality, symmetric metric completion collapse, host enriched colimits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed profile admission, probe comparison, FormalBall approximation, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LawvereMetricCompletionUp.

The $\mathsf{LawvereMetricCompletion}^{\uparrow}$ directed-profile route consumes symmetric completion data only after the one-sided profile has already passed through $\mathsf{QuasiMetric}^{\uparrow}$, $\mathsf{YonedaCompletion}^{\uparrow}$, and $\mathsf{FormalBall}^{\uparrow}$. The metric-completion dense embedding of chapter 3351 and the dense-reflector route of chapter 3786 are therefore horizontal consumers of the displayed profile surface, not replacements for the directed source row.

Proposition 3947.8 (Lawvere metric completion dense-reflector handoff)**.** *Any completion-facing read that compares an accepted* $\mathsf{LawvereMetricCompletion}^{\uparrow}$ *packet with* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *or* $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ *must first display the* $\mathsf{QuasiMetric}^{\uparrow}$ *source row,* $\mathsf{YonedaCompletion}^{\uparrow}$ *profile row,* $\mathsf{FormalBall}^{\uparrow}$ *approximation row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal row from Definition 3947.4. Only after that directed profile is visible may the read consume the symmetric dense-embedding or Cauchy dense-reflector rows. The handoff exports no selected profile limit, quotient directed Cauchy object, symmetric-collapse equality, countable-choice schedule, or ambient completeness theorem.*

Proof. Start with the directed-profile composition theorem. It fixes the visible route through the QuasiMetric source, Yoneda profile, FormalBall approximation, dyadic tolerance ledger, regular rational readback, and Real seal before any completion-facing consumer can inspect the packet.

Next read the sibling dense-completion surfaces. $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ supplies the symmetric dense-image row for ordinary metric completion, while $\mathsf{CauchyCompletionDenseReflector}^{\uparrow}$ supplies the Cauchy dense-reflector route through that dense image. Both are consumed after the Lawvere profile has exposed its directed rows.

Thus the comparison is a handoff from the directed profile into the existing completion infrastructure. Since the Lawvere carrier has no coordinate for selected profile limits, quotient directed Cauchy objects, symmetric-collapse equality, countable choice, or ambient completeness, none of those objects is exported by the handoff. □

Theorem 3947.9 (Lawvere metric completion Cauchy route)**.** *A symmetric dense-reflector comparison for an accepted* LawvereMetricCompletion$^{\uparrow}$ *packet reads* MetricCompletionDenseEmbedding$^{\uparrow}$ *and* CauchyCompletionDenseReflector$^{\uparrow}$ *only after the Lawvere one-sided route has displayed* QuasiMetric$^{\uparrow}$*,* YonedaCompletion$^{\uparrow}$*,* FormalBall$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. The comparison with chapter 3351 and chapter 3786 is therefore a horizontal lattice edge out of the directed profile route, not a replacement for any source, tolerance, regular-rational, or Real-seal row.*

Proof. Start from Definition 3947.4. It displays the one-sided Lawvere route through the quasi-metric source, Yoneda profile, formal-ball approximation, dyadic tolerance ledger, regular rational readback, and Real seal.

Apply the dense-reflector handoff proposition above. It allows the symmetric dense-embedding and Cauchy dense-reflector rows to be consumed only after that directed profile is already visible.

The two sibling chapters then provide the horizontal consumers: MetricCompletionDenseEmbedding$^{\uparrow}$ reads the symmetric dense image, and CauchyCompletionDenseReflector$^{\uparrow}$ reads the Cauchy reflector through that dense image. Since the carrier lacks selected profile limits, quotient directed Cauchy objects, symmetric-collapse equality, choice, and ambient completeness, the route remains the displayed lattice edge. □

Theorem 3947.10 (Lawvere metric-completion sibling lattice route)**.** *Any symmetric metric-completion consumer of an accepted* LawvereMetricCompletion$^{\uparrow}$ *packet must first pass through the directed route displayed after* QuasiMetric$^{\uparrow}$*,* YonedaCompletion$^{\uparrow}$*,* FormalBall$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*, and* Real$^{\uparrow}$*. It may then compare that route with* MetricCompletionDenseEmbedding$^{\uparrow}$ *and* CauchyCompletionDenseRefl *but the comparison does not replace the directed source row or the one-sided profile classifier.*

Proof. Use the namecert construction route of chapter 3947. The packet names completion through the directed QuasiMetric$^{\uparrow}$, YonedaCompletion$^{\uparrow}$, and FormalBall$^{\uparrow}$ rows before any rational readback or terminal Real seal is available.

The symmetric siblings are therefore downstream comparison surfaces. The dense embedding of chapter 3351 and the dense-reflector route of chapter 3786 may inspect the same completed-window data, but neither sibling supplies a new directed source row.

Thus the horizontal lattice edge is substantive only after the Lawvere route has been read, and it preserves the one-sided profile classifier of the original packet. □

HyperbolicCrossRatio$^{\uparrow}$ routes four-boundary comparison through chapter 5110, chapter 6237, and the visual-metric boundary route before exposing cross-ratio comparisons.

Chapter 3948

A Concrete Naming Certificate for $\mathsf{HyperbolicCrossRatio}^{\uparrow}$

The $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ packet realizes the four-boundary-point comparison branch of chapter 9463 as finite NameCert data. It sits beside the Poincare disk phase route of chapter 3455, the hyperbolic metric surface of chapter 3474, the boundary-fixed transport surface of chapter 3500, the Busemann boundary packet of chapter 3732, and the Schwarz–Pick disk-contraction row of chapter 3682. The packet names a local four-point invariant over displayed disk-boundary rows; it does not import a host ideal-boundary compactification, a global hyperbolic plane, selected geodesics, or quotient boundary equality.

Definition 3948.1 (Hyperbolic cross-ratio carrier)**.** *A* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *disk-source and phase-transport row for the visible disk coordinate system.*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row attached to that disk source.*
3. *B_0, B_1, B_2, B_3 are four boundary-point rows, each read through the displayed Poincare boundary interface rather than through a host compactification.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row proving that the four boundary rows remain attached to the same boundary-fixed disk transport.*
5. *Q is the finite cross-ratio comparison row, exposing only the retained pairwise boundary-distance and oriented separation comparisons.*
6. *R is the refusal row excluding host global geometry, selected geodesics, quotient boundary equality, and object-level ideal-boundary compactification.*
7. *F is the* $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$ *provenance row that records nonexpansive disk-map compatibility for consumers comparing the cross-ratio under disk transport.*
8. *H records componentwise* hsame *transport for disk, metric, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, Schwarz–Pick provenance, package provenance, and local naming.*

9. *C records displayed* Cont *replay from disk phase transport through metric comparison, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, and provenance.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{SchwarzPickDiskContraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N$. It has no coordinate for fewer than four boundary rows, a host global metric oracle, selected ideal points outside the displayed boundary interface, quotient boundary equality, or unrecorded disk transport.

Theorem 3948.2 (Hyperbolic cross-ratio NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *carrier $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}$ *surface consists exactly of Poincare disk phase transport, hyperbolic-metric comparison, four displayed boundary rows, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, Schwarz–Pick provenance, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer must inspect those rows before treating the packet as a hyperbolic four-point invariant.*

Proof. Project the carrier of Definition 3948.1. The disk and metric rows D, M fix the Poincare disk reading and the hyperbolic comparison surface.

The four boundary rows B_0, B_1, B_2, B_3 are then read together with the boundary-fixed transport row T. This is the only point at which the carrier exposes boundary points, and all four rows must belong to the same displayed disk-transport route.

The comparison and refusal route is Q, R, F. The finite cross-ratio row records the retained pairwise boundary comparisons, the refusal row prevents replacement by host global geometry, and the Schwarz–Pick row records the nonexpansive disk-map provenance consumed by downstream transport comparisons.

Finally H, C, P, N transport, replay, source, and name the same finite surface. Since no other row is present, no consumer can omit a boundary row, replace hyperbolic comparison by a global oracle, or bypass the local naming route. □

Theorem 3948.3 (Hyperbolic cross-ratio nonescape)**.** *Every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{PoincareDiskPhaseTransport}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{PoincareBoundaryFixedTransport}^{\uparrow},\ \mathsf{HyperbolicBusemannBoundary}^{\uparrow}$$

The route fixes the disk phase surface, the hyperbolic metric comparison, four boundary rows, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, Schwarz–Pick provenance, transport, replay, package provenance, and local naming. It exports no host global geometry, quotient boundary equality, selected ideal-boundary compactification, or unrecorded disk-map invariant.

Proof. Let $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$ be accepted. A consumer first reads D, M, so all later comparisons are attached to the displayed disk phase transport and hyperbolic metric rows.

Next the consumer reads B_0, B_1, B_2, B_3, T. The four boundary rows are simultaneous carrier fields, and the boundary-fixed transport row keeps them in the same disk-transport context.

The comparison row Q is accepted only together with the refusal row R and Schwarz–Pick provenance F. Thus cross-ratio comparison remains finite and local to the displayed disk route.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and naming. The excluded host and quotient objects have no carrier coordinate, so the consumer cannot escape the finite packet. □

Theorem 3948.4 (Hyperbolic cross-ratio Mobius consumer route)**.** *Any* $\mathsf{MobiusTransform}^{\uparrow}$ *consumer of an accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *packet must read the disk row D, metric row M, four boundary rows B_0, B_1, B_2, B_3, boundary-fixed transport row T, cross-ratio row Q, refusal row R, Schwarz–Pick provenance row F, and structural rows H, C, P, N before it may claim cross-ratio preservation. The consumer route of chapter 5110 has no accepted preservation coordinate outside this four-boundary packet.*

Proof. Apply Definition 3948.3. Every accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer already factors through the disk, metric, boundary, transport, comparison, refusal, provenance, replay, package, and naming rows of the finite carrier.

A $\mathsf{MobiusTransform}^{\uparrow}$ preservation read is a downstream consumer of the same packet, not a separate host-level geometric claim. Since the carrier has no coordinate for cross-ratio preservation unless the four displayed boundary rows and the boundary-fixed transport route are present, the Mobius consumer must enter through $D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N$. □

Theorem 3948.5 (Hyperbolic cross-ratio horofunction consumer route)**.** *Any* $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ *consumer of an accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *packet must first read the disk row D, hyperbolic metric row M, four boundary rows B_0, B_1, B_2, B_3, boundary-fixed transport row T, cross-ratio row Q, refusal row R, Schwarz–Pick provenance row F, and structural rows H, C, P, N before treating the packet as a boundary metric comparison. The consumer route reads boundary metric data through Busemann and visual-boundary rows, not through a host compactification.*

Proof. Apply Definition 3948.3. Every accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer already enters through the disk, metric, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, provenance, replay, package, and naming rows.

A $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ read is boundary-metric facing, so it cannot treat Q as an isolated scalar comparison. The four boundary rows and the metric row must first be routed through the displayed Busemann and visual-boundary surfaces that the horofunction-boundary packet consumes.

The refusal row R blocks host compactification and quotient ideal-boundary equality. The structural rows H, C, P, N then preserve the same finite route, so the horofunction consumer receives only the cross-ratio data that has passed through the displayed boundary metric rows. □

Falsifiable prediction: Any accepted $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ consumer omitting one of the four boundary rows, the $\mathsf{HyperbolicMetric}^{\uparrow}$ comparison, the $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ row, Pkg provenance, or the local $\mathsf{NameCert}$ row refutes Definition 3948.3.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, poincare-boundary-fixed-transport, hyperbolic-busemann-boundary, schwarz-pick-disk-contraction: $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries disk phase transport, hyperbolic metric comparison, four boundary rows, boundary-fixed transport, finite cross-ratio comparison, refusal of host global geometry, Schwarz–Pick provenance, transport, replay, package provenance, and local naming.

The Lean TasteGate file for $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ will provide explicit $\mathsf{Nontrivial\ HyperbolicCrossRatio}^{\uparrow}$ and $\mathsf{FieldFaithful\ HyperbolicCrossRatio}^{\uparrow}$ instances. These instances keep the disk, metric, boundary, transport, comparison, refusal, provenance, replay, package, and local naming rows distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ packet is generated from PoincareDiskPhaseTransportUp disk data, HyperbolicMetricUp comparison, four displayed boundary rows, PoincareBoundaryFixedTransportUp transport, finite cross-ratio comparison, global-geometry refusal, SchwarzPickDiskContractionUp provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3948.1, Definition 3948.2, Definition 3948.3, and Definition 3948.4 seed the finite four-boundary-point hyperbolic cross-ratio route over disk, metric, boundary, transport, refusal, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicCrossRatio}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host global geometry, quotient boundary equality, selected ideal-boundary compactification, unrecorded disk-map invariants, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, four boundary rows, hyperbolic metric comparison, boundary-fixed transport, finite cross-ratio comparison, global-geometry refusal, TasteGate nontriviality, field faithfulness, provenance, transport, replay, and local naming for BEDC.Derived.HyperbolicCrossRatioUp.

Theorem 3948.6 (Hyperbolic cross-ratio Mobius-invariant sibling route)**.** *Every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *four-boundary-point carrier consumes the* $\mathsf{MobiusTransform}^{\uparrow}$ *route of chapter 5110 and the* HyperbolicMobiusInvarian *route of chapter 6237 before any cross-ratio comparison is exported. In carrier notation* $X = (D, M, B_0, B_1, B_2, B_3, T, Q, R, F, H, C, P, N)$*, the accepted comparison row* Q *is public only after the disk, metric, four boundary, boundary-fixed transport, refusal, provenance, replay, package, and naming rows have passed the sibling Mobius and invariant routes.*

Proof. Apply Definition 3948.3. Every public read of Q already factors through the displayed disk, metric, four boundary rows, boundary-fixed transport, refusal, Schwarz–Pick provenance, transport, replay, provenance, and local naming rows.

Apply Definition 3948.4. A $\mathsf{MobiusTransform}^{\uparrow}$ consumer of that packet may claim preservation only through the same four-boundary carrier route, so the Mobius sibling is a prerequisite route rather than an external host symmetry.

The $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ sibling then reads the same Mobius-facing route as invariant data. Since X has no coordinate for an isolated cross-ratio scalar, quotient boundary equality, or unrecorded transformation invariant, any accepted cross-ratio comparison exported from Q has already consumed both sibling routes. □

Theorem 3948.7 (Hyperbolic cross-ratio logarithm route)**.** *Every accepted* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *read that exports a logarithmic distance or invariant must first display the four endpoint rows* B_0, B_1, B_2, B_3*, the cross-ratio comparison ledger* Q*, positive-domain admission, the* $\mathsf{RealLogarithm}^{\uparrow}$ *inverse route of chapter 2315, the* $\mathsf{RealExponential}^{\uparrow}$ *comparison row,* $\mathsf{RealAbs}^{\uparrow}$ *error control, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The logarithm is a sibling dependency read through the displayed RealLogarithm certificate; it is not a host logarithm primitive and it exports no quotient boundary equality, selected ideal endpoint, host global geometry, or unrecorded disk-map invariant.*

Proof. Project the carrier of Definition 3948.1. The endpoint rows B_0, B_1, B_2, B_3 and the cross-ratio comparison row Q are already displayed before any downstream invariant can be read.

The logarithmic route is then admitted only as a positive-domain read. Invoke the $\mathsf{RealLogarithm}^{\uparrow}$ certificate of chapter 2315: its inverse route passes through the $\mathsf{RealExponential}^{\uparrow}$ comparison row and the $\mathsf{RealAbs}^{\uparrow}$ error-control row before a logarithm endpoint is consumed.

Finally transport and replay the same carrier route through H, C, P, N. The sibling logarithm certificate supplies the inverse-route dependency, while the $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ carrier supplies the four endpoint rows and comparison ledger. Neither surface contains a host logarithm, quotient boundary equality, selected ideal endpoint, global geometry oracle, or unrecorded disk-map invariant. □

$\mathsf{HyperbolicCrossRatio}^{\uparrow}$ reads the boundary visual metric surface of chapter 3866 before any cross-ratio comparison is consumed as a boundary-distance statement. This route keeps Mobius-invariant comparison, logarithmic readback, visual boundary distance, transport, replay, provenance, and local naming in the displayed hyperbolic boundary ledger.

Chapter 3949

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, Kelley on filters, and Mac Lane on functorial transport, the $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ packet records the finite route that transports regular Cauchy filter data into stream-indexed real seals. It reads the regular filter surface of chapter 1445, the Cauchy filter basis and saturation surfaces of chapter 1501 and chapter 333, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-functor route of chapter 423. The packet names the transport surface only; it does not quotient filters, select limits, identify host real objects, or use ambient completeness.

Definition 3949.1 (RegularCauchyFilterFunctor carrier)**.** *A* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, B, W, D, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *I is the regular Cauchy filter index row, reading the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *source packet and its displayed filter-indexed tail data.*
2. *B is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, including the finite basis and saturation route used before any stream window is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the requested windows obtained from the filter-indexed source.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after W and D are displayed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the filter, window, tolerance, and readback rows have been replayed.*
7. *A is the functorial action ledger, recording source-map action on displayed filter indices and target-map action on the induced stream windows, readbacks, tolerances, and seals.*
8. *H records componentwise* hsame *transport for I, B, W, D, R, E, A and for the structural rows below.*

9. *C records displayed* Cont *replay for the route from regular Cauchy filter indices through CauchyFilter basis data, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, and action ledger.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, W, D, R, E, A, H, C, P, N$. There is no coordinate for quotient filters, selected representatives, host equality of filters, host equality of reals, countable choice, or detached completion points.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterFunctorUp.RegularCauchyFilterFunctorTasteGate single carrier alignment`

Theorem 3949.2 (RegularCauchyFilterFunctor NameCert obligations)**.** *For every accepted* RegularCauchyFilterFu *carrier*

$$F = (I, B, W, D, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; regular Cauchy filter source exposure through I; CauchyFilter basis and saturation exposure through B; finite-window exposure through W; dyadic tolerance exactness through D; regular-rational readback through R; Real seal handoff through E; functorial action bookkeeping through A; and structural transport, replay, provenance, and local naming through H, C, P, N. No obligation is discharged by quotienting filters, choosing representatives, identifying host real objects, or importing ambient completeness.*

Proof. Project the carrier of Definition 3949.1. The accepted packet displays exactly the source index row, basis row, window row, tolerance row, readback row, seal row, action ledger, and structural rows.

The source side is exhausted by I and B. The regular Cauchy filter row supplies the filter-indexed tail data, while the CauchyFilter basis row supplies the finite basis and saturation surface before any stream-indexed consumer can inspect the packet.

The target side is exhausted by W, D, R, E. The finite windows are read first, the dyadic tolerance ledger is attached to those windows, the RegSeqRat row reads back the regular rational data, and the Real seal remains downstream from the displayed route.

The action and structural rows do not add hidden objects. The ledger A records how displayed source-map action is replayed as target-map action on W, D, R, E, while H, C, P, N transport, replay, package, and name the same finite surface. Hence the listed rows are the whole local NameCert obligation surface. □

Theorem 3949.3 (RegularCauchyFilterFunctor filter-to-stream route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegularCauchyFilter}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer reads I, then B, then W and D, then R, then E, with the functorial action ledger A and the structural rows H, C, P, N preserving this order. No second real-facing route is available inside the carrier.

Proof. Start with the Real-facing coordinate. The row E is not detached from the packet; it is stored only after the source index, filter basis, finite windows, dyadic tolerances, and regular-rational readback have been displayed.

Read the route in order. The source packet contributes I, the CauchyFilter basis and saturation data contribute B, the stream-indexed finite observations contribute W, the tolerance checks contribute D, and the regular-rational readback contributes R.

Now inspect the action ledger. The row A records that source-map action on filter indices is replayed on the displayed window, tolerance, readback, and seal rows, so the functorial transport does not introduce a selected limit or a host equality principle.

Finally the transport, replay, provenance, and local naming rows certify the same route. Since the carrier has no other Real seal coordinate and no quotient-filter coordinate, every Real-facing consumer factors through the listed finite chain. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ read is refuted if a RealUp seal is reached without displaying the RegularCauchyFilter, CauchyFilter, StreamName, DyadicRatCore, RegSeqRat, functorial action, transport, replay, provenance, and local naming rows.

Independence witness: regularcauchyfilter, cauchyfilter, streamname, regseqrat, dyadicratcore, real, and cauchycompletionfunctor: none is bijective to the functor packet because only $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ records filter-index source action and stream-index target action together with one Real seal route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterFunctor}^{\uparrow}$ packet is generated from a regular Cauchy filter index row, CauchyFilter basis row, StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, RealUp seal row, functorial action ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3949.1, Definition 3949.2, and Definition 3949.3 seed the finite filter-indexed route from RegularCauchyFilter and CauchyFilter data through StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected representatives, host equality of filters, host equality of reals, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-index source exposure, CauchyFilter basis exposure, window and tolerance routing, RegSeqRat readback, Real seal handoff, functorial action, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterFunctorUp.

Chapter 3950

A Concrete Naming Certificate for ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$

Following Arzela, Ascoli, and Bishop–Bridges constructive compactness, the ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$ packet records the finite extraction surface where equicontinuity and compact finite nets are read before a sequence-window consumer reaches a real seal. It sits downstream of the finite interval-net route of chapter 3471, the modulus composition route of chapter 3784, the compact uniform modulus extractor of chapter 2202, the convergent-sequence boundary of chapter 3944, the located Cauchy subsequence route of chapter 3945, the dyadic tolerance core of chapter 291, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the real seal of chapter 38. The chapter names only the finite equicontinuous extraction packet; it does not assert open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real$^{\uparrow}$ completeness.

Definition 3950.1 (Arzela–Ascoli finite equicontinuous carrier)**.** *An* ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, E, U, M, D, W, Q, L, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net source row, read through the* HeineBorelFiniteNet$^{\uparrow}$ *and compact-metric finite-net data already visible to the packet.*
2. *E is the finite equicontinuity-modulus row, assigning each displayed target tolerance a displayed source radius uniformly over the inspected family.*
3. *U is the* UniformModulusComposition$^{\uparrow}$ *and* CompactUniformModulusExtractor$^{\uparrow}$ *compatibility row, recording that the compact finite-net and equicontinuity ledgers use the same tolerance requests.*
4. *M is the diagonal or finite-window subsequence row, exposed only as the finite window data consumed by* RealConvergentSequence$^{\uparrow}$ *and* LocatedCauchySubsequence$^{\uparrow}$ *routes.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for source radii, target tolerances, mesh slack, and comparison budgets.*
6. *W is the finite* StreamName$^{\uparrow}$ *window row naming the inspected source points, function-family indices, and target observations.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the rational observations on W.*

8. *L is the located convergence-window row used by the* LocatedCauchySubsequence$^{\uparrow}$ *consumer.*
9. *S is the terminal* Real$^{\uparrow}$ *seal row, available only after K, E, U, M, D, W, Q, L have been displayed.*
10. *H records componentwise* hsame *transport for all source, modulus, window, readback, located, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a finite extraction request through $K, E, U, M, D, W, Q, L, S$.*
12. *P is the* Pkg *provenance over* ArzelaAscoliFiniteEquicontinuous$^{\uparrow}$, HeineBorelFiniteNet$^{\uparrow}$, UniformModulusCompositi CompactUniformModulusExtractor$^{\uparrow}$, RealConvergentSequence$^{\uparrow}$, LocatedCauchySubsequence$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, U, M, D, W, Q, L, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFiniteEquicontinuousUp.ArzelaAscoliFiniteEquicontinuousTasteGate`
`single carrier alignment`

Theorem 3950.2 (Arzela–Ascoli finite equicontinuous obligations)**.** *For every accepted* ArzelaAscoliFiniteEquiconti *carrier*

$$A = (K, E, U, M, D, W, Q, L, S, H, C, P, N),$$

the local NameCert$_{\text{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite-net row, equicontinuity modulus row, compact-modulus compatibility row, finite diagonal window, dyadic tolerance ledger, stream windows, rational readback, located convergence window, real seal, transport, replay, provenance, and local naming rows;*
2. *compact-net exposure, requiring every source point used by the extraction to be read through the displayed* HeineBorelFiniteNet$^{\uparrow}$ *finite-net data before any target observation is consumed;*
3. *equicontinuity visibility, requiring the shared source-radius row E and compatibility row U to be displayed before the diagonal window M can be read;*
4. *L10 readback order, requiring the dyadic tolerance ledger, stream windows, regular rational readback, located Cauchy window, and terminal Real seal to appear in the order D, W, Q, L, S;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only these finite rows and to export no open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3950.1. The packet exposes the compact finite-net source, equicontinuity modulus, compact-modulus compatibility row, finite diagonal window, tolerance ledger, stream windows, rational readback, located convergence row, real seal, and structural rows as one finite BHist object.

Compact-net exposure is the row K. A consumer must first read the finite net and coverage witnesses supplied by the Heine–Borel finite-net route, so the source compactness input is a displayed finite object rather than an arbitrary open cover.

Equicontinuity visibility is supplied by E and U. The requested target tolerance is converted into a displayed source radius, and the compact uniform modulus row records that the same tolerance budget is used by the finite-net and diagonal-window reads.

The L10 readback order is then forced by the packet coordinates. The diagonal window M can be inspected only through D, W, Q, L, and the terminal row S is downstream of those dyadic, stream, rational, and located Cauchy rows.

Finally, H, C, P, N preserve that row inventory. Since the carrier has no coordinate for open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, or ambient Real completeness, none of those objects can be exported by the local naming surface. □

Falsifiable prediction: Every $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ consumer read that reaches $\mathsf{Real}^{\uparrow}$ first displays compact finite-net rows, equicontinuity finite modulus rows, compact-modulus compatibility rows, a finite diagonal window, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, and located Cauchy window rows.

Independence witness: heineborelfinet, uniformmoduluscomposition, compactuniformmodulusextractor, realconvergentsequence, locatedcauchysubsequence, dyadicratcore, streamname, regseqrat, real: none is bijective to $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ because this packet jointly carries compact finite-net exposure, equicontinuity-radius visibility, compact-modulus compatibility, diagonal-window data, dyadic tolerances, stream windows, rational readback, located convergence rows, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}$ packet is generated from compact finite-net rows, equicontinuity finite modulus rows, compact-modulus compatibility, a finite diagonal or subsequence window, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, LocatedCauchySubsequence rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3950.1 and Definition 3950.2 seed the finite equicontinuous extraction packet consumed by L10 compact-continuous and Real-completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFiniteEquicontinuous}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected infinite subsequence choice, quotient stream equality, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, equicontinuity visibility, L10 readback order, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ArzelaAscoliFiniteEquicontinuousUp.

Chapter 3951

A Concrete Naming Certificate for $\mathsf{PoissonSummation}^{\uparrow}$

Following Poisson and Fourier analysis, the $\mathsf{PoissonSummation}^{\uparrow}$ packet is a finite BEDC route from lattice-indexed source windows through Fourier finite-observation rows and distribution-facing summation rows. It sits over the finite coefficient interface of chapter 124, the lattice source of chapter 55, the pushforward and test-row boundary of chapter 186, and the terminal real readback of chapter 38. The packet names a finite summation certificate; it does not import analytic convergence, host function equality, Schwartz-space quotients, Haar measure, or an infinite series oracle.

Definition 3951.1 (Poisson summation finite carrier)**.** *A* $\mathsf{PoissonSummation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, F, D, S, R, H, C, P, N)$$

with the following displayed coordinates.

1. *L is the* $\mathsf{Lattice}^{\uparrow}$ *row naming the finite lattice index window, dual-window tags, and shift ledger used by the summation read.*
2. *F is the* $\mathsf{Fourier}^{\uparrow}$ *row naming the finite coefficient and frequency observations made over the displayed lattice window.*
3. *D is the* $\mathsf{Distribution}^{\uparrow}$ *and test-function row through which the finite Fourier observations are read as distribution-facing data.*
4. *S is the finite summation-window row, recording the source-side and dual-side partial sums inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback surface for the displayed summation comparison.*
6. *H records componentwise* hsame *transport for lattice, Fourier, distribution, summation-window, and real-readback rows.*
7. *C records displayed* Cont *replay from lattice windows through Fourier observations, distribution rows, finite summation, and readback.*
8. *P is the* Pkg *provenance over* $\mathsf{PoissonSummation}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine displayed rows. It has no coordinate for analytic convergence, equality of host functions, an infinite summation theorem, Schwartz-space quotient equality, Haar measure, or a selected distribution representative.

Lean4 ✓ `BEDC.Derived.PoissonSummationUp.PoissonSummationTasteGate single carrier alignment`

Theorem 3951.2 (Poisson summation NameCert obligations)**.** *For every accepted* $\mathsf{PoissonSummation}^{\uparrow}$ *carrier* $Q = (L, F, D, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$ *surface consists exactly of carrier admission for* Q*, lattice-window admission through* L*, finite Fourier observation through* F*, distribution-facing test-row admission through* D*, finite source and dual summation-window comparison through* S*,* $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *readback through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.PoissonSummationUp.PoissonSummationNamecertObligations`

Proof. Project the carrier of Definition 3951.1. The analytic surface is exhausted by the lattice row, Fourier row, distribution-facing row, summation-window row, and readback row.

Read L, F, D, S in order. The lattice window determines the finite source and dual indices, the Fourier row records only the inspected frequency observations, the distribution row supplies the test-facing interpretation, and the summation row compares the two displayed finite windows.

The rows R, H, C, P, N then seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for convergence, host function equality, quotient spaces, Haar measure, or selected representatives, none of those objects is part of the local NameCert surface. □

Theorem 3951.3 (Poisson summation Fourier-lattice route)**.** *Every accepted* $\mathsf{PoissonSummation}^{\uparrow}$ *read factors through the displayed route*

$$\mathsf{Lattice}^{\uparrow} \longmapsto \mathsf{Fourier}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto S \longmapsto \mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}.$$

Thus a downstream consumer receives only finite lattice windows, finite Fourier observations, distribution-facing test rows, finite summation windows, real readback, transport, replay, provenance, and local naming.

Proof. Begin with the lattice coordinate L of Definition 3951.1. It supplies the only source of summation indices and dual-window tags.

Apply the Fourier and distribution coordinates F and D. They expose finite coefficient observations and test-facing rows before the summation window S can be compared.

Finally pass through R, H, C, P, N. The regular and real readback rows seal the finite comparison, while transport, replay, provenance, and naming preserve the route without adding analytic convergence or host equality. □

Falsifiable prediction: Every accepted $\mathsf{PoissonSummation}^{\uparrow}$ read factors through lattice windows, Fourier finite observations, distribution-facing test rows, finite summation windows, $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ readback, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows.

Independence witness: lattice, fourier, distribution, real: $\mathsf{PoissonSummation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lattice windows, Fourier observations, distribution-facing test rows, finite source and dual summation windows, readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoissonSummation}^{\uparrow}}$**.**

Constructive story: A PoissonSummation$^{\uparrow}$ packet is generated from a LatticeUp window row, a FourierUp finite-observation row, a DistributionUp test-facing row, a finite summation-window comparison, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PoissonSummation}^{\uparrow}}$, seed)

Scope closed: Definition 3951.1, Definition 3951.2, and Definition 3951.3 seed the finite Fourier-lattice summation packet over lattice, Fourier, distribution, and real-readback rows.

Formal verification: FormalStatus(NameCert$_{\text{PoissonSummation}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.PoissonSummationUp.PoissonSummationTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim analytic convergence, host function equality, Schwartz-space quotients, Haar measure, infinite summation theorems, or bridge checking.

Upgrade path: A formal route can prove the NameCert obligation surface and Fourier-lattice route theorem.

Chapter 3952

A Concrete Naming Certificate for $\mathsf{RealDecimalNormalForm}^{\uparrow}$

Following Bishop and Bridges on constructive real approximants and the classical decimal expansion normalization convention, the $\mathsf{RealDecimalNormalForm}^{\uparrow}$ packet records the finite readback surface by which a displayed $\mathsf{Real}^{\uparrow}$ name is presented through decimal prefixes. It sits after the decimal-prefix route of chapter 1417, the integer bracket route of chapter 1479, the rational naming surface of chapter 37, the dyadic tolerance surface of chapter 291, the regular-rational handoff of chapter 301, and the terminal seal discipline of chapter 38. The packet names a canonical decimal readback route only; it does not choose a quotient representative, compare host real streams, or import ambient real completeness.

Definition 3952.1 (RealDecimalNormalForm carrier)**.** *A* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, I, Q, Y, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *prefix ledger, including finite digit rows and decimal-place windows.*
2. *I is the* $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$ *bracket row that fixes the integer-part surface used by the decimal readback.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *place-value row that evaluates each inspected decimal prefix as a rational expression.*
4. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same prefix and bracket data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained after the decimal, bracket, rational, and dyadic rows are displayed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row whose name is read only after D, I, Q, Y, R have been exposed.*
7. *H records componentwise* hsame *transport for decimal-prefix, integer-part, rational, dyadic, regular-rational, and real-seal rows.*
8. *C records displayed* Cont *replay for the route from decimal prefix windows through integer brackets, place values, tolerances, regular readback, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RealDecimalNormalForm}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{ArchimedeanIntegerPart}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$ *row.*

The classifier for the packet compares accepted decimal normal-form reads only through $D, I, Q, Y, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, trichotomy over arbitrary real names, host decimal reals, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RealDecimalNormalFormUp.RealDecimalNormalFormTasteGate single carrier alignment`

Theorem 3952.2 (RealDecimalNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *carrier*

$$M = (D, I, Q, Y, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed decimal-prefix, integer-bracket, rational place-value, dyadic tolerance, regular-rational handoff, real-seal, transport, replay, provenance, and local naming rows;*
2. *digit legality and place-value exactness, requiring every accepted digit row in D to be a decimal digit and every inspected prefix to be evaluated through the rational row Q;*
3. *endpoint normalization, requiring the ledger to exclude the eventual-nine route before a decimal read is treated as normal-form data;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring R to be reached only after the decimal, integer-bracket, rational, and dyadic rows have been displayed;*
5. $\mathsf{Real}^{\uparrow}$ *seal nonescape, requiring every real-facing consumer to read E through the same $D, I, Q, Y, R, H, C, P, N$ route.*

No obligation is discharged by quotienting streams, selecting endpoint representatives, deciding trichotomy for all real names, or using ambient real completeness.

Lean4 ✓ `BEDC.Derived.RealDecimalNormalFormUp.RealDecimalNormalFormCarrier namecert obligation surface`

Proof. Project the carrier of Definition 3952.1. The displayed rows $D, I, Q, Y, R, E, H, C, P, N$ are the whole packet, so carrier admission is rowwise.

Digit legality and place-value exactness are read from D and Q. The decimal-prefix ledger supplies only finite digit rows and windows, while the rational row Q evaluates those windows before any dyadic or real-facing consumer can inspect them.

Endpoint normalization is local to the same finite ledger. The non-eventual-nine condition is recorded before the regular-rational handoff, so the packet does not identify two host decimal streams or select an endpoint representative.

The handoff and seal clauses follow the order of the remaining coordinates. The dyadic row Y is attached to the same prefix and bracket data, the regular-rational row R is downstream of D, I, Q, Y, and the terminal real seal E is downstream of R with H, C, P, N preserving the route. □

Theorem 3952.3 (RealDecimalNormalForm Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{ArchimedeanIntegerPart}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads decimal prefix rows, then the integer-part bracket rows, then the rational place-value rows, then dyadic tolerance rows, then the regular-rational handoff, and only then the terminal $\mathsf{Real}^{\uparrow}$ *seal. The packet supplies no quotient stream equality, selected representative, trichotomy row, host decimal real, or ambient completeness.*

Proof. Start with an accepted packet $M = (D, I, Q, Y, R, E, H, C, P, N)$. The decimal rows D are already finite prefix rows, and I supplies the integer bracket that fixes the readback surface before place-value evaluation is exposed.

Now read the arithmetic rows. The rational row Q evaluates the inspected prefixes, and the dyadic row Y translates the same values into the tolerance discipline used by regular-rational and real consumers.

Apply the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R. It is listed only after the decimal, integer-bracket, rational, and dyadic rows, so no consumer can reach the regular-rational readback without displaying that route.

Finally read the terminal seal E and the structural rows H, C, P, N. These rows transport, replay, source, and name the same finite route, and none of them is a quotient equality, selected representative, trichotomy decision, host decimal real, or completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{RealDecimalNormalForm}^{\uparrow}$ packet is generated from DecimalExpansion prefix rows, ArchimedeanIntegerPart bracket rows, Rat place-value rows, DyadicRatCore tolerance rows, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 3952.1, Definition 3952.2, and Definition 3952.3 seed the finite canonical decimal readback route from DecimalExpansion and ArchimedeanIntegerPart data through Rat, DyadicRatCore, RegSeqRat, and RealUp seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealDecimalNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close uniqueness of all decimal expansions, quotient stream equality, selected representatives, trichotomy over arbitrary real names, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, digit legality, place-value exactness, endpoint-normalization ledger exhaustion, RegSeqRat handoff, and RealUp seal nonescape for BEDC.Derived.RealDecimalNormalFormUp.

Chapter 3953

A Concrete Naming Certificate for $\mathsf{CauchyModulusNormalizer}^{\uparrow}$

Following Bishop–Bridges constructive analysis with explicit Cauchy moduli, the $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ packet records the finite route by which a displayed Cauchy threshold schedule is normalized before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal consumers may read it. It sits next to the fast Cauchy criterion of chapter 3825, the monotone normalization surface of chapter 1744, the StreamName carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the Real seal of chapter 38. The packet is a finite normalizer certificate for Real and Completion routes; it is not Real completeness, quotient completion, quotient stream equality, a selected limit, countable choice, trichotomy, host real equality, or a standard bridge.

Definition 3953.1 (Cauchy modulus normalizer carrier)**.** *A* $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, M, F, D, W, Q, S, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the source row, read through a* $\mathsf{FastCauchyCriterion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, or* $\mathsf{RegSeqRat}^{\uparrow}$ *Cauchy-window surface.*
2. *M is the input Cauchy modulus row, assigning displayed tail thresholds to requested dyadic tolerances.*
3. *F is the finite normalization table, recording prefix maxima or equivalent displayed threshold joins for the finitely many tighter requests visible to the packet.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for precision requests, slack conversions, and positivity checks consumed by M and F.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row available after the normalized threshold has been displayed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations exposed on W.*
7. *S is the normalized Cauchy-window seal row saying that the readback admitted after F is already admitted by the input modulus row M at the requested tolerance.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after X, M, F, D, W, Q, S have been displayed.*

9. *H records componentwise* hsame *transport for the source, input modulus, normalization table, dyadic ledger, windows, readback, Cauchy-window seal, Real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a precision request through X, M, D, F, W, Q, S, R.*

11. *P is the* Pkg *provenance over* CauchyModulusNormalizer$^{\uparrow}$, FastCauchyCriterion$^{\uparrow}$, CauchyModulusMonotoneNormal StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{CauchyModulusNormalizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, F, D, W, Q, S, R, H, C, P, N$. It has no coordinate for Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or bridge checking.

Theorem 3953.2 (Cauchy modulus normalizer NameCert obligations)**.** *For every accepted* CauchyModulusNormali *carrier*

$$Z = (X, M, F, D, W, Q, S, R, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusNormalizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source, input modulus, finite normalization table, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Cauchy-window seal, Real seal, transport, replay, provenance, and local naming rows;*

2. *modulus-source exposure, requiring the source row X and input modulus row M to be displayed before the normalizer table F is read;*

3. *finite normalization exactness, requiring F to read only finitely many tolerance requests from D and threshold rows from M;*

4. *window-readback order, requiring StreamName windows, RegSeqRat readback, and the Real seal to appear downstream of the normalized threshold route as W, Q, S, R;*

5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed normalizer rows and to export no Real completeness, quotient completion, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or bridge datum.*

No Cauchy-modulus-normalizer obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyModulusNormalizerUp.CauchyModulusNormalizerTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 3953.1. The accepted packet exposes the source row, input modulus, finite normalization table, dyadic tolerance ledger, stream windows, rational readback, Cauchy-window seal, Real seal, and structural rows as one finite carrier.

Modulus-source exposure is the order of the displayed packet. A consumer cannot inspect F until the source row X, input threshold schedule M, and dyadic tolerance ledger D have fixed the finite requests being normalized.

Finite normalization exactness is finite because F reads only the displayed tolerance requests in D and their corresponding thresholds in M. This is the same constructive modulus-normalization discipline used by the monotone normalization packet, but here it is packaged as the normalizer boundary immediately before downstream window reads.

The window-readback order follows from the coordinates W, Q, S, R. StreamName observations are read after the normalized threshold, RegSeqRat readback records the rational observations, the

Cauchy-window seal S checks preservation against M, and the terminal Real seal R is downstream of those rows.

Finally, H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no coordinate for Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or bridge data, none of those objects can become a local NameCert obligation. □

Falsifiable prediction: Every $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ consumer read factors through a source row, an input Cauchy modulus row, a finite normalization table, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a Cauchy-window seal, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fastcauchycriterion, cauchymodulusmonotonenormalization, streamname, regseqrat, dyadicratcore, real: none is bijective to $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ because this packet jointly carries the source, input modulus, finite normalization table, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Cauchy-window preservation seal, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusNormalizer}^{\uparrow}$ packet is generated from a Cauchy-window source row, an input Cauchy modulus row, a finite normalization table, DyadicRatCore tolerances, StreamName windows, RegSeqRat readback, a preservation seal, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3953.1 and Definition 3953.2 seed the finite modulus-normalizer packet used before StreamName, RegSeqRat, DyadicRatCore, and RealUp seal reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusNormalizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient completion, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-source exposure, finite normalization exactness, window-readback order, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusNormalizerU

Chapter 3954

A Concrete Naming Certificate for $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$

The $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet realizes the total-host and universal-substrate separation of chapter 9370 as finite NameCert data. It sits between the neutral-spine comparison surface of chapter 3909, the bounded fuel trace of chapter 3889, the normal-form readback of chapter 936, the substrate trace boundary of chapter 944, and the total-host fuel discipline of chapter 964. The packet records the boundary at which host normalization evidence remains total while substrate universality remains fuel-indexed and external to the host evaluator.

Definition 3954.1 (MetaCIC strong-normalization boundary carrier)**.** *A* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, F, Z, U, T, R, H, C, P, N)$$

with the following displayed rows.

1. *S is the* $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$ *row carrying the typed neutral-spine comparison route.*
2. *F is the* $\mathsf{MetaCICFuelTrace}^{\uparrow}$ *row carrying bounded normalizer fuel, endpoints, equality-check readback, and refusal data.*
3. *Z is the* $\mathsf{ZNormal}^{\uparrow}$ *row carrying normal-form and total-host readback evidence.*
4. *U is the* $\mathsf{BoundedUniversalTrace}^{\uparrow}$ *row carrying the fuel-indexed substrate trace.*
5. *T is the* $\mathsf{StepIndexedTotalHost}^{\uparrow}$ *row carrying total-host fuel discipline and step-indexed readback.*
6. *R is the refusal row stating that substrate universality is not re-exported as host nontermination or global conversion decidability.*
7. *H records componentwise* hsame *transport for spine, fuel, normal-form, substrate-trace, total-host, and refusal rows.*
8. *C records displayed* Cont *replay from total-host fuel through bounded trace and normal-form readback.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$, $\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{ZNormal}^{\uparrow}$, $\mathsf{BoundedUniversalTrace}^{\uparrow}$, $\mathsf{StepIndexedTotalHost}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for host nontermination, global conversion decidability, an unbounded substrate run, a selected normalization proof, semantic equality, or a collapse of substrate universality into the total host.

Lean4 ✓ `BEDC.Derived.MetaCICStrongNormalizationBoundaryUp.MetaCICStrongNormalizationBoundaryTasteGate single carrier alignment`

Theorem 3954.2 (MetaCIC strong-normalization boundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *carrier* $B = (S, F, Z, U, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for* B*, neutral-spine admission through* S*, fuel-trace admission through* F*, normal-form readback through* Z*, bounded substrate-trace admission through* U*, total-host fuel discipline through* T*, Turing-refusal exposure through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3954.1. The host-facing rows are S, F, Z, T, the substrate-facing row is U, and the boundary refusal is R.

Read the host rows first. The neutral-spine, fuel-trace, normal-form, and step-indexed total-host rows give only finite fuel, endpoint, and readback evidence.

Read the substrate row through U and then the refusal row R. Bounded substrate universality remains fuel-indexed, and the refusal row prevents that universality from becoming host nontermination or global conversion decidability.

The rows H, C, P, N transport, replay, source, and name the same boundary. Since the carrier contains no host nontermination, unbounded trace, selected normalization proof, semantic equality, or global conversion theorem, none of those objects is exported. □

Theorem 3954.3 (MetaCIC strong-normalization boundary Turing-refusal route)**.** *Every substrate-universality read of an accepted* $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICDecidableNeutralSpine}^{\uparrow}, \mathsf{MetaCICFuelTrace}^{\uparrow}, \mathsf{ZNormal}^{\uparrow}, \mathsf{StepIndexedTotalHost}^{\uparrow} \longmapsto \mathsf{BoundedUniversalTrace}^{\uparrow}$$

Thus the packet preserves total-host normalization evidence while refusing to turn the bounded universal substrate trace into host nontermination, unbounded evaluation, or global conversion decidability.

Proof. Begin with the host-facing coordinates S, F, Z, T of Definition 3954.1. They expose only typed neutral-spine comparison, bounded normalizer fuel, normal-form readback, and step-indexed total-host rows.

Move to the substrate coordinate U. It supplies a bounded universal trace as a finite row, not an unbounded run and not a host evaluator.

Finally apply the refusal coordinate R and preserve it through H, C, P, N. The refusal row is part of the same carrier, so any read that keeps substrate universality must also keep the boundary that blocks host nontermination and global conversion claims. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet displays neutral-spine, finite fuel, normal-form, bounded universal trace, step-indexed total-host, refusal, hsame, Cont, Pkg, and $\mathsf{NameCert}$ rows before any consumer reads the total-host and universal-substrate boundary.

Independence witness: metacic-decidable-neutral-spine, metacic-fuel-trace, znormal, bounded-universal-trace, step-indexed-total-host: no listed sibling is bijective to $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ because this packet jointly carries neutral-spine comparison, finite fuel trace, normal-form read-back, bounded substrate trace, total-host fuel discipline, refusal, transport, replay, provenance, and local naming in one boundary carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}$ packet is generated from MetaCICDecidableNeutralSpineUp, MetaCICFuelTraceUp, ZNormalUp, BoundedUniversalTraceUp, and StepIndexedTotalHostUp rows, together with a Turing-refusal row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3954.1, Definition 3954.2, and Definition 3954.3 seed the finite MetaCIC boundary between total-host normalization evidence and bounded substrate universality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICStrongNormalizationBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host nontermination, unbounded substrate evaluation, global conversion decidability, semantic equality, Lean verification, bridge checking, or a collapse of substrate universality into total-host computation.

Upgrade path: A formal route can encode the finite carrier, prove the NameCert obligation surface and Turing-refusal route, and add TasteGate Nontrivial and FieldFaithful instances.

Chapter 3955

A Concrete Naming Certificate for $\mathsf{CofinalFilter}^{\uparrow}$

Following Kelley and Bourbaki on cofinal filters, the $\mathsf{CofinalFilter}^{\uparrow}$ packet records the finite corridor by which a downstream filter or completion consumer reads a target filter through a displayed cofinal source route. It sits between the finite base surface of chapter 1601, the Cauchy-filter route of chapter 333, the refinement corridor of chapter 1626, the uniform-completion consumer of chapter 1473, and the metric-completion consumer of chapter 1506. The packet records cofinal comparison data without quotient filters, selected representatives, selected limits, ultrafilters, countable choice, Zorn arguments, or ambient completeness.

Definition 3955.1 (CofinalFilter carrier)**.** *A* $\mathsf{CofinalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, \rho, \lambda, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{FilterBase}^{\uparrow}$ *row whose displayed packets answer the downstream request after cofinal replacement.*
2. *T is the target* $\mathsf{FilterBase}^{\uparrow}$ *row named by the consumer before the cofinal route is read.*
3. *ρ is the cofinal map row, sending every displayed target request to a source packet contained in that target request.*
4. *λ is the reverse ledger when route equivalence is needed. It records the finite reverse containment and common-refinement reads, and it is absent when only one-way cofinal consumption is claimed.*
5. *Q is the completion handoff row consumed by* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$ *readers after the cofinal comparison rows have been displayed.*
6. *H records componentwise* hsame *transport for $S, T, \rho, \lambda, Q, C, P, N$.*
7. *C records displayed* Cont *replay from a target request through ρ, through the source base packet, through λ when present, and then through the completion handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterRefinement}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalFilter}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, ultrafilter extension, selected representatives, selected limits, countable choice, Zorn arguments, host filter equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CofinalFilterUp.TasteGate.CofinalFilterTasteGate single carrier alignment`

Theorem 3955.2 (CofinalFilter NameCert obligations)**.** *For every accepted* $\mathsf{CofinalFilter}^{\uparrow}$ *carrier* $F = (S, T, \rho, \lambda, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CofinalFilter}^{\uparrow}}$ *surface consists exactly of carrier admission for the source and target* $\mathsf{FilterBase}^{\uparrow}$ *rows, the cofinal map row, the reverse ledger when equivalence is requested, the completion handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer request must expose* T, ρ, S *and the relevant ledger rows before* Q *is available.*

Proof. Project the carrier of Definition 3955.1. The rows S and T display the source and target filter bases, while ρ is the only row that moves a target request to a source packet.

The ledger row λ is conditional on equivalence use. When the consumer asks only for one-way cofinal replacement it is not read; when equivalence is claimed it supplies exactly the reverse containment and common-refinement entries needed for that claim.

The completion handoff row Q is downstream of this comparison. Transport through H and replay through C preserve the same finite order, and P, N source and name the route. Since the excluded quotient, ultrafilter, selector, choice, Zorn, host-equality, and completeness objects are absent from the carrier, no further NameCert obligation is available. □

Theorem 3955.3 (CofinalFilter completion handoff)**.** *Every* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{UniformCompletion}^{\uparrow}$*, or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CofinalFilter}^{\uparrow}$ *packet reads completion-facing data only through*

$$T \longmapsto \rho \longmapsto S \longmapsto \lambda, Q, H, C, P, N.$$

Thus the completion corridor factors through the displayed cofinal-filter rows and the sibling $\mathsf{FilterRefinement}^{\uparrow}$ *discipline before any completion request is answered. It exports no quotient filter, selected representative, selected limit, ultrafilter, countable-choice schedule, Zorn argument, host filter equality, or ambient completeness row.*

Proof. Begin with a completion-facing request. The consumer names a target filter base row T, so the carrier forces the request through the cofinal map row ρ before any source packet can be read.

Apply Definition 3955.2. The source row S supplies the displayed packet answering the target request, while λ is read only when the consumer needs the reverse comparison ledger required by the refinement discipline.

The handoff row Q is therefore reached after the finite cofinal comparison. Componentwise transport, continuation replay, provenance, and local naming preserve that same route for $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ consumers. No excluded quotient, selector, choice, Zorn, host-equality, or completeness coordinate is exported. □

Falsifiable prediction: If a completion-facing read cites $\mathsf{CofinalFilter}^{\uparrow}$ while bypassing the displayed target row, cofinal map, source row, required reverse ledger, completion handoff, hsame transport, Cont replay, Pkg provenance, or local NameCert row, it is not an accepted $\mathsf{CofinalFilter}^{\uparrow}$ packet.

Independence witness: Nearest siblings: filterbase, cauchyfilter, filter-refinement, uniformcompletion, metriccompletion; $\mathsf{CofinalFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly

records a target base, source base, cofinal map, optional reverse ledger, completion handoff, transport, continuation, provenance, and local naming in one finite corridor.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}$.

Constructive story: A $\mathsf{CofinalFilter}^{\uparrow}$ packet is generated from source and target FilterBase rows, a cofinal map, a reverse ledger when equivalence is requested, a completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}, \mathsf{seed})$

Scope closed: Definition 3955.1, Definition 3955.2, and Definition 3955.3 seed the finite cofinal-filter corridor between filter bases, filter refinement, CauchyFilter, UniformCompletion, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalFilter}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filter equality, selected representatives, selected limits, ultrafilters, countable choice, Zorn arguments, host filter equality, Lean verification, bridge checking, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, cofinal map exactness, reverse-ledger compatibility, completion handoff non-escape, transport stability, replay stability, and ledger-refusal proofs for BEDC.Derived.CofinalFilterUp.

Chapter 3956

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion, the $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ packet records the functorial route from metric source data to regular Cauchy real seals. It sits over the metric source of chapter 57, the finite-window interface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the metric-completion surface of chapter 1506, and the completion action of chapter 423. The packet names the source-map and composition ledgers needed by completion consumers; it does not select representatives, quotient regular Cauchy names, identify host real objects, or import ambient completeness.

Definition 3956.1 (RegularCauchyCompletionFunctor carrier)**.** *A* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, U, W, Q, R, E, I, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source and target metric naming certificates and their distance ledgers.*
2. *U is the uniformly continuous map row between the displayed metric sources, with modulus and graph rows readable before completion is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording the source windows requested by a regular Cauchy input name.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same finite windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *completion-name row obtained from W and Q.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the metric, map, finite-window, tolerance, and regular-rational rows have been displayed.*
7. *I is the identity ledger, saying that the identity uniformly continuous map replays the same windows, tolerances, readback rows, and real seal.*
8. *A is the composition ledger, saying that two displayed uniformly continuous map rows compose by replaying the source windows through the intermediate and target completion rows in order.*
9. *H records componentwise* hsame *transport for M, U, W, Q, R, E, I, A and for the structural rows below.*

10. *C records displayed* Cont *replay from metric source rows and uniformly continuous map rows through StreamName windows, DyadicRatCore tolerances, RegSeqRat completion names, RealUp seals, identity ledgers, and composition ledgers.*

11. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, W, Q, R, E, I, A, H, C, P, N$. There is no coordinate for a selected limit, quotient-name equality, host equality of real objects, global completeness, or an external functor object detached from the displayed rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionFunctorUp.RegularCauchyCompletionFunctorTasteGate`
`single carrier alignment`

Theorem 3956.2 (RegularCauchyCompletionFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *carrier*

$$F = (M, U, W, Q, R, E, I, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$ *surface consists exactly of metric source admission through M; uniformly continuous map admission through U; finite-window exposure through W; dyadic tolerance exactness through Q; regular Cauchy completion-name readback through R; RealUp seal handoff through E; identity and composition functor ledgers through I and A; and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3956.1. Its accepted rows display exactly the metric source row, uniformly continuous map row, finite-window row, dyadic tolerance row, regular-rational readback row, real seal, identity ledger, composition ledger, and structural rows.

The source side is exhausted by M, U, W, Q. The metric row supplies the distance ledgers, the uniformly continuous map row supplies the graph and modulus data, and the finite-window and dyadic-tolerance rows record the only observations a regular Cauchy input can expose.

The target side is exhausted by R, E. The RegSeqRat row is read after the requested windows and tolerances, and the RealUp seal remains downstream from that completion-name readback.

The functorial rows add no hidden semantic object. The identity ledger I and composition ledger A record replay of the displayed route, while H, C, P, N transport, continue, source, and name the same finite packet. Hence the listed rows are the complete local NameCert obligation surface. □

Theorem 3956.3 (RegularCauchyCompletionFunctor composition law)**.** *Let F_f and F_g be accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *packets for displayed uniformly continuous maps $f\colon X \to Y$ and $g\colon Y \to Z$. Every completion-facing consumer of the composite read factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

for f, then through the same displayed route for g, with the composition ledger A preserving the intermediate windows, tolerance rows, regular-rational readbacks, RealUp seals, hsame *transports,* Cont *replays,* Pkg *provenance rows, and local NameCert rows. The identity ledger I gives the same statement when one of the two map rows is the displayed identity.*

Proof. Begin with the accepted packets for f and g. By Definition 3956.2, each packet exposes the metric source, uniformly continuous map, finite-window, dyadic-tolerance, regular-rational readback, real-seal, and structural rows before any completion-facing consumer can read its output.

For the composite request, read the first packet through M, U, W, Q, R, E. This produces only the displayed regular Cauchy completion-name row and its RealUp seal; it does not export a selected representative or a host equality of completed points.

Feed that displayed seal route to the second packet. The second packet repeats the same finite-window and tolerance discipline over its own metric and map rows. The ledger A records that this two-stage replay is the composite route, and I records that the identity map leaves the displayed windows, tolerances, readbacks, and seal unchanged.

Finally apply the structural rows. Componentwise hsame transport and displayed Cont replay preserve the two-stage route, while Pkg provenance and the local NameCert rows keep the consumer inside $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ data. Thus the composition law is a certificate route over displayed rows, not an external equality of functors. □

Theorem 3956.4 (RegularCauchyCompletionFunctor Real-seal handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ *packet reaches the RealUp seal row only through the displayed MetricUp source row, uniformly continuous map row, StreamName finite-window row, DyadicRatCore tolerance row, RegSeqRat readback row, identity ledger, composition ledger, and structural rows. In carrier notation, the route to the seal is*

$$M, U, W, Q, R, E, I, A, H, C, P, N,$$

with E downstream from M, U, W, Q, R and still guarded by I, A, H, C, P, N. The handoff does not expose selected representatives, quotient regular Cauchy names, host real equality, or ambient completeness.

Proof. Project the carrier of Definition 3956.1. The row E is listed only after the metric source, uniformly continuous map, finite-window, dyadic-tolerance, and RegSeqRat readback rows have been displayed.

Apply the NameCert obligations of Definition 3956.2. They identify the RealUp seal handoff as one obligation among the same finite rows, and they keep identity, composition, transport, replay, provenance, and local naming visible as part of the accepted packet.

Finally use the composition law, Definition 3956.3. It shows that a completion-facing composite read preserves the same window, tolerance, readback, seal, and ledger route rather than creating an external functor equality. Since the carrier has no coordinate for representatives, quotient names, host real equality, or ambient completeness, the Real-seal handoff can export only the displayed finite route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ read is refuted if a completion-facing RealUp seal is reached without displaying the MetricUp source row, uniformly continuous map row, StreamName window row, DyadicRatCore tolerance row, RegSeqRat readback row, identity ledger, composition ledger, transport, replay, provenance, and local naming rows.

Independence witness: metric, streamname, regseqrat, real, metriccompletion, and cauchycompletionfunctor: none is bijective to the functor packet because only $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ records uniformly continuous source action together with identity and composition ledgers for regular Cauchy completion names.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}$ packet is generated from MetricUp source rows, uniformly continuous map rows, StreamName finite windows, DyadicRatCore tolerance rows, RegSeqRat completion-name rows, RealUp seal rows, identity and composition ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3956.1, Definition 3956.2, Definition 3956.3, and Definition 3956.4 seed the finite functorial route from MetricUp and uniformly continuous map rows through StreamName windows, DyadicRatCore tolerances, RegSeqRat completion names, and RealUp seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient regular Cauchy names, selected representatives, host equality of real objects, global completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric source exposure, uniformly continuous map exposure, window and tolerance routing, RegSeqRat readback, RealUp seal handoff, identity and composition ledgers, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyCompletionFunctorUp.

Chapter 3957

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$

Following Lebesgue and Bishop–Bridges, the $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ packet realizes the finite-net route of chapter 9237 as concrete NameCert data. It reads the compact carrier of chapter 60, the compact metric finite-net surface of chapter 129, the metric source rows of chapter 57, the continuous-map and continuous-function rows of chapter 130 and chapter 59, the rational lower-bound surface of chapter 37, the finite-net fold of chapter 619, and the source-lock packet of chapter 935. The packet records only finite $\mathsf{ProbeBundle}$ data, local pointwise-modulus rows, rational folds, and the resulting uniform-modulus handoff; it does not claim open-cover compactness, selected subsequences, countable choice, or a semantic Heine–Cantor theorem outside the displayed BEDC rows.

Definition 3957.1 (Compact uniform finite-net realization carrier)**.** *A* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, M, F, B, L, R, U, T, H, D, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Compact}^{\uparrow}$ *admission row, tying the source object to the compact naming certificate.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source-distance and target-distance ledgers.*
3. *F is the* $\mathsf{Continuous}^{\uparrow}$ *or* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, with pointwise modulus data readable at displayed source centers.*
4. *B is the finite* $\mathsf{ProbeBundle}$ *row of compact-net centers and source-distance coverage witnesses.*
5. *L is the local-radius ledger, assigning a positive* $\mathsf{Rat}^{\uparrow}$ *radius to each bundled center from the pointwise-modulus row.*
6. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row over the finite bundle, using only primitive recursion over displayed probes.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the folded radius and the continuous graph rows.*
8. *T is the metric triangle row, joining the source-center estimate, the source-pair estimate, and the target-distance estimate.*

9. *H records componentwise* hsame *transport for C, M, F, B, L, R, U, T and for the structural rows below.*
10. *D records displayed* Cont *replay from compact finite-net inspection through pointwise-modulus reads, rational lower-bound folding, metric triangle estimates, and uniform-modulus handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, M, F, B, L, R, U, T, H, D, P, N$. It has no coordinate for open-cover compactness, a selected subcover outside the finite bundle, a selected subsequence, quotient equality of real values, or an external modulus function.

Lean4 ✓ `BEDC.Derived.CompactUniformFiniteNetRealizationUp.CompactUniformFiniteNetRealizationTasteGate`
`single carrier alignment`

Theorem 3957.2 (Compact finite-net modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *packet receives the compact-uniform-continuity route through the finite chain*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Continuous}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The consumer first displays the compact metric source, then the finite ProbeBundle centers and coverage rows, then the pointwise-modulus radii supplied by the continuous row, then the rational lower-bound fold, and only then the metric-triangle estimate and uniform-modulus handoff.

Proof. Project the carrier of Definition 3957.1. The rows $C, M, F, B, L, R, U, T, H, D, P, N$ are the complete displayed inventory.

Read the compact and metric source rows first. The compact row admits the source object, while the metric row supplies the distance ledgers used by both the finite-net cover and the target estimate.

Next read the finite bundle. The ProbeBundle row B supplies exactly the displayed centers and source-distance coverage rows, and the continuous row F supplies pointwise-modulus radii only at those centers. The local-radius ledger L therefore remains tied to the finite compact-net data.

Now fold the radii. The $\mathsf{Rat}^{\uparrow}$ row R is obtained by primitive recursion over B, so the common lower bound is a classifier-stable history rather than a choice-selected minimum.

Finally the metric triangle row T and uniform-modulus row U export the handoff, with H, D, P, N transporting, replaying, sourcing, and naming the same finite surface. Thus the realization supplies the vision route as NameCert data without importing open-cover compactness, selected subsequences, countable choice, or an external modulus selector. □

Theorem 3957.3 (Compact uniform finite-net realization nonescape)**.** *Let $K = (C, M, F, B, L, R, U, T, H, D, P, N$ be an accepted* $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ *carrier. Any consumer that reaches the uniform-modulus handoff must factor through the same finite ProbeBundle row B, the same* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold R, the same continuous-map pointwise-modulus row F, and the same metric triangle row T. No alternate path inside the carrier exports open-cover compactness, a selected subcover outside B, a selected subsequence, or a host-level modulus function.*

Proof. Start from the output row U. It is stored after C, M, F, B, L, R, T in the carrier, so the uniform-modulus handoff cannot be read without the compact source, metric source, continuous row, finite bundle, local radii, rational fold, and triangle estimate.

The finite bundle is the only cover coordinate. Since B is the sole row containing centers and coverage witnesses, any source point used by the handoff must be routed through one of the displayed probe names.

The lower-bound and triangle rows are likewise unique inside the packet. The fold row R compares only the radii displayed by L, and T is the only row converting source estimates and pointwise target estimates into the final uniform estimate.

Transport, replay, provenance, and local naming preserve this route rather than adding new analytic data. Hence every consumer of U remains inside the finite NameCert surface described by the compact-uniform-continuity vision. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ read is refuted if a UniformModulusUp handoff is reached without displaying the CompactUp row, MetricUp source row, ContinuousUp pointwise-modulus row, finite ProbeBundle centers, RatUp lower-bound fold, metric triangle row, transport, replay, provenance, and local naming rows.

Independence witness: compact, compactmetric, metric, continuous, rat, probebundle, finitenetminimumfold, and uniformmodulus: none is bijective to the finite-net realization packet because only $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ records the compact-uniform-continuity vision route from finite ProbeBundle coverage through rational folding to the UniformModulusUp handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}$ packet is generated from CompactUp admission rows, MetricUp source rows, ContinuousUp or ContinuousMapUp pointwise-modulus rows, finite ProbeBundle center and coverage rows, RatUp lower-bound folds, UniformModulusUp handoff rows, MetricTriangleUniformEstimateUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3957.1, Definition 3957.2, and Definition 3957.3 seed the finite ProbeBundle realization route for compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNetRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers outside the finite bundle, selected subsequences, countable choice, semantic Heine–Cantor theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact source exposure, metric source exposure, continuous pointwise-modulus exposure, finite ProbeBundle coverage, RatUp lower-bound folding, metric triangle exactness, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformFiniteNetRealizationUp.

Chapter 3958

A Concrete Naming Certificate for $\mathsf{MetricSequentialCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricSequentialCompletion}^{\uparrow}$ packet records the finite interface where sequential completion data is read through a metric-completion surface without quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits between the sequential route of chapter 1683, the metric-completion boundary of chapter 1506, the complete-metric handoff of chapter 128, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a completion-facing route over displayed rows only; it does not identify Cauchy sequences by a host quotient or choose completed representatives.

Definition 3958.1 (MetricSequentialCompletion carrier)**.** *A* $\mathsf{MetricSequentialCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, M, C, W, T, R, E, H, K, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{SequentialCompletion}^{\uparrow}$ *source row naming the sequential completion request.*
2. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row through which completion-facing consumers read the metric boundary.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row exposed only after the sequential and metric rows are displayed.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family used to inspect the sequential source.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W and T.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after Q, M, C, W, T, R.*
8. *H records componentwise* hsame *transport for the sequential, metric, complete, window, tolerance, readback, seal, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from the sequential request through the metric-completion boundary, complete-metric handoff, finite windows, dyadic tolerance, regular readback, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{MetricSequentialCompletion}^{\uparrow}$, $\mathsf{SequentialCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected sequence limits, countable-choice selectors, host equality of completed points, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MetricSequentialCompletionUp.MetricSequentialCompletionTasteGate single carrier alignment`

Theorem 3958.2 (MetricSequentialCompletion NameCert obligations)**.** *For every accepted* MetricSequentialCompl *carrier $S = (Q, M, C, W, T, R, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$ *surface consists exactly of sequential-source admission through Q, metric-completion admission through M, complete-metric handoff through C, finite-window exposure through W, dyadic tolerance exactness through T, regular rational readback through R, real-name seal handoff through E, and structural transport, replay, provenance, and local naming through H, K, P, N.*

Proof. Project the carrier of Definition 3958.1. The accepted rows display the sequential source, metric completion row, complete-metric handoff, finite windows, dyadic tolerance ledger, regular readback, terminal real seal, and structural rows.

The completion side is exhausted by Q, M, C. A consumer first reads the sequential request, then the metric-completion boundary, and only then the complete-metric handoff.

The real-name side is exhausted by W, T, R, E. The finite window family determines the dyadic tolerance reads, the tolerance ledger determines the regular rational readback, and the real seal remains downstream of that readback.

Transport, replay, provenance, and local naming preserve this route. Since no other coordinate is present, quotient streams, selected limits, countable choice, host equality, and ambient completeness cannot enter the local certificate surface. □

Theorem 3958.3 (MetricSequentialCompletion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricSequentialCompletion}^{\uparrow}$ *carrier reads the same finite route*

$$\mathsf{SequentialCompletion}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

The handoff sends sequential completion data to regular rational readback through displayed stream windows and dyadic tolerances; it does not use quotient streams, selected limits, countable choice, host equality, or ambient real completeness.

Proof. Begin with Definition 3958.2. It fixes the local certificate surface at the rows $Q, M, C, W, T, R, E, H, K, P, N$.

Read the sequential and metric rows first. The packet admits the sequential request through Q, the metric-completion boundary through M, and the complete-metric handoff through C before a readback row can be inspected.

Now read the finite windows and tolerances. The rows W and T determine the regular rational row R, and the real-name seal E is downstream of R.

The structural rows keep the same handoff. Componentwise transport, displayed replay, provenance, and local naming record only this route, so the excluded quotient, selector, choice, host-equality, and completeness data remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSequentialCompletion}^{\uparrow}$ packet is generated from SequentialCompletionUp source rows, MetricCompletionUp boundary rows, CompleteMetricUp handoff rows, StreamName window rows, DyadicRatCore tolerance rows, RegSeqRat readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3958.1, Definition 3958.2, and Definition 3958.3 seed the finite route from sequential completion and metric completion rows through stream windows, dyadic tolerances, regular rational readback, and the real-name seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSequentialCompletion}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MetricSequentialCompletionUp.MetricSequentialCompletionTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable-choice selectors, host equality of completed points, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequential-source exposure, metric-completion admission, complete-metric handoff, window and tolerance routing, regular rational readback, real-name seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricSequentialCompletionUp.

Chapter 3959

A Concrete Naming Certificate for $\mathsf{SpectralAblationHinge}^{\uparrow}$

The $\mathsf{SpectralAblationHinge}^{\uparrow}$ packet realizes the spectral-ablation ledger of Definition 9462.1458, the single-codon deletion table of Definition 9462.1459, the orbit-module deletion table of Definition 9462.1461, the tail-coupling edge deletion of Definition 9462.1463, and the load-bearing interface summary of Definition 9462.1468 as finite NameCert data. It records deletion rows, killed-walk rows, spectral-loss rows, orbit-module rows, transport rows, replay rows, provenance rows, and local naming rows. The packet does not claim a biological fitness law, a continuous spectrum, a host eigenvector selector, or an external ranking rule outside the displayed ledger.

The packet consumes the finite hinge-ledger rows of chapter 3961, the quotient spectral trigger rows of chapter 2317, and the energy-partition boundary of chapter 3806 before its ranking row is read. The killed-walk matrix rows and hinge rankings come from the ledger sibling, while the quotient trigger and energy-partition rows constrain which deletion reads can be treated as trigger-facing.

Definition 3959.1 (SpectralAblationHinge carrier)**.** *A* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, K, L, O, T, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the finite deletion row, naming the deleted codon, edge, edge group, or orbit module.*
2. *K is the killed-walk row computed after the displayed deletion.*
3. *L is the spectral-loss ledger row recording the displayed difference between the source spectral radius and the killed-walk spectral radius.*
4. *O is the orbit-module row that records whether the deletion is point-level, module-level, interface-level, or single-edge data.*
5. *T is the tail-coupling row for the* UUR–CUR *antenna interface when the deletion reads that boundary.*
6. *R is the hinge-ranking row obtained by reading only D, K, L, O, T.*
7. *H records componentwise* hsame *transport for deletion, killed-walk, loss, orbit-module, tail-coupling, ranking, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from finite deletion data through killed-walk inspection, spectral-loss computation, orbit-module classification, tail-coupling inspection, and hinge ranking.*
9. *P is the* Pkg *provenance over* $\mathsf{SpectralAblationHinge}^{\uparrow}$*, radial-wobble-spectrum, wobble-quotient-spectral-trigger, six-boundary-tile-coverage, window-coverage-trigger-algebra, hyperbolic-metric,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$ *row.*

The carrier has no coordinate for an accepted hinge ranking that bypasses the spectral-loss ledger L.

Lean4 ✓ `BEDC.Derived.SpectralAblationHingeUp.SpectralAblationHingeTasteGate single carrier alignment`

Theorem 3959.2 (SpectralAblationHinge NameCert obligations)**.** *For every accepted* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier $A = (D, K, L, O, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$ *surface consists exactly of finite deletion admission through D, killed-walk admission through K, spectral-loss ledger exactness through L, orbit-module classification through O, tail-coupling boundary exposure through T, hinge-ranking readback through R, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 3959.1. The displayed rows are exactly $D, K, L, O, T, R, H, C, P, N$.

The deletion side is exhausted by D, K, L. A consumer first reads the finite deletion row, then the killed-walk row after that deletion, and then the spectral-loss ledger comparing the source and killed rows.

The hinge side is exhausted by O, T, R. Orbit-module classification and tail-coupling inspection supply the only ranking inputs, and the ranking row R is read only after those rows and the loss ledger have been displayed.

Transport, replay, provenance, and local naming preserve this finite ledger. No accepted NameCert obligation is stored in a host eigenvector selector, a continuous spectrum, or a ranking rule detached from D, K, L, O, T, R. □

Theorem 3959.3 (SpectralAblationHinge loss route)**.** *Every accepted hinge-ranking read from a* $\mathsf{SpectralAblationHinge}^{\uparrow}$ *carrier factors through the route*

$$D \longmapsto K \longmapsto L \longmapsto O, T \longmapsto R \longmapsto H, C, P, N.$$

Deleting the spectral-loss ledger L prevents the carrier from exporting an accepted hinge ranking, even when the deletion row, killed-walk row, orbit-module row, and tail-coupling row remain displayed.

Proof. Start with the NameCert obligations of Definition 3959.2. They require finite deletion admission, killed-walk admission, and spectral-loss ledger exactness before orbit-module or hinge-ranking data is read.

The ranking row is downstream of the loss row. The orbit-module row O and tail-coupling row T classify the deletion, but neither row stores the spectral loss itself.

If L is deleted, the displayed route cannot reach R as an accepted hinge-ranking row. The structural rows H, C, P, N only transport, replay, source, and name accepted rows, so they cannot reconstruct the missing loss ledger. □

Falsifiable prediction: Deleting the spectral-loss ledger prevents any accepted $\mathsf{SpectralAblationHinge}^{\uparrow}$ hinge ranking read, even when finite deletion, killed-walk, orbit-module, tail-coupling, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: radial-wobble-spectrum, wobble-quotient-spectral-trigger, six-boundary-tile-coverage, window-coverage-trigger-algebra, hyperbolic-metric: none is bijective to the spectral ablation hinge packet because only $\mathsf{SpectralAblationHinge}^{\uparrow}$ binds deletion, killed-walk, spectral-loss, orbit-module, tail-coupling, ranking, transport, replay, provenance, and local naming rows into one accepted hinge ledger.

Lean follow-up must provide $\mathsf{Nontrivial\ SpectralAblationHinge}^{\uparrow}$ and $\mathsf{FieldFaithful\ SpectralAblationHinge}^{\uparrow}$ in TasteGate.lean before the packet can be read as a checked TasteGate object.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralAblationHinge}^{\uparrow}$ packet is generated from finite deletion rows, killed-walk rows, spectral-loss ledger rows, orbit-module rows, tail-coupling rows, hinge-ranking rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3959.1, Definition 3959.2, and Definition 3959.3 seed the finite deletion, killed-walk, spectral-loss, orbit-module, tail-coupling, ranking, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralAblationHinge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological fitness laws, continuous spectral theory, host eigenvector selectors, Lean verification, TasteGate checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, deletion admission, killed-walk admission, spectral-loss exactness, orbit-module classification, tail-coupling exposure, ranking readback, transport, replay, provenance, and local naming for BEDC.Derived.SpectralAblationHingeUp.

Chapter 3960

A Concrete Naming Certificate for $\mathsf{SierpinskiCarpet}^{\uparrow}$

Following Sierpinski and the constructive compact-metric treatment of finite square approximations, the $\mathsf{SierpinskiCarpet}^{\uparrow}$ packet records the standard middle-square deletion fractal as retained grid rows and coordinate readback data. It sits beside the retained-prefix route of chapter 1530, the compact source discipline of chapter 129, the finite-grid continuous-surjection surface of chapter 2078, the terminal real seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet names only finite square-grid retention and coordinate windows; it does not import a host closed subset of the plane, an infinite branch selector, quotient equality of real pairs, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3960.1 (Sierpinski carpet carrier)**.** *A* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (G, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the retained square-grid row. It records a finite triadic square address, its depth, the retained child squares at that depth, and the compact-metric grid enclosure used by later consumers.*
2. *D is the deleted-middle-square ledger. For every inspected square address it records whether the central child has been removed and which retained children remain available.*
3. *W is the coordinate* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the finite horizontal and vertical coordinate windows attached to the retained square.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two coordinate schedules, with dyadic comparisons routed through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before any real seal is exposed.*
5. *E is the pair of* $\mathsf{Real}^{\uparrow}$ *coordinate seals obtained from the same retained square-grid and coordinate-window rows.*
6. *H records componentwise* hsame *transport for retained grid rows, deletion ledgers, coordinate windows, rational readbacks, real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route from retained grid rows and deleted-middle-square ledgers through coordinate windows, rational readbacks, and real seals.*

8. *P is the* Pkg *provenance over* $\mathsf{SierpinskiCarpet}^{\uparrow}$, $\mathsf{CantorSet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PeanoCurve}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, D, W, Q, E, H, C, P, N$. It has no coordinate for a completed planar subset, selected infinite address, quotient equality of real pairs, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.SierpinskiCarpetUp.SierpinskiCarpetTasteGate single carrier alignment`

Theorem 3960.2 (Sierpinski carpet NameCert obligations)**.** *For every accepted* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *carrier*

$$S = (G, D, W, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; retained square-grid exposure through G; deleted-middle-square exclusion through D; coordinate window exposure through W; regular rational readback through Q; real coordinate sealing through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. No obligation is discharged by a host planar powerset, an infinite branch selector, quotient real-pair equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3960.1. The accepted packet displays exactly the retained square-grid row, deletion ledger, coordinate windows, rational readbacks, real seals, and structural rows.

The retained-grid and deletion obligations are exhausted by G and D. A consumer cannot inspect a square address until the retained child row has been displayed, and a deleted central square can be refused only by the ledger row attached to that same address.

The coordinate obligations are exhausted by W, Q, E. The coordinate windows are read first, the regular rational readback uses those windows with the dyadic tolerance ledger, and the $\mathsf{Real}^{\uparrow}$ seals remain downstream of the same retained square.

Finally H, C, P, N transport, replay, source, and name only the displayed rows. Since the excluded host principles have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$ obligations. □

Theorem 3960.3 (Sierpinski carpet finite square-deletion route)**.** *Every accepted* $\mathsf{SierpinskiCarpet}^{\uparrow}$ *consumer factors its membership or compact-metric read through the finite route*

$$\mathsf{CantorSet}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{PeanoCurve}^{\uparrow} \longmapsto G, D \longmapsto W, Q, E \longmapsto H, C, P, N.$$

At each inspected depth the route first displays retained square-grid rows and deleted-middle-square ledgers, then reads coordinate $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *schedules,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances, and terminal* $\mathsf{Real}^{\uparrow}$ *seals. It exports no completed planar subset, selected infinite branch, quotient real-pair equality, open-cover compactness theorem, or ambient completeness principle.*

Proof. Begin with the retained square-grid row. It gives a finite triadic square address and the retained children available at the inspected depth, while the deleted-middle-square ledger records which central cell is refused.

Read coordinates only after that ledger is visible. The window row W supplies the horizontal and vertical finite schedules, and Q reads those schedules as regular rational data with the dyadic tolerance comparisons already displayed.

The Real-facing boundary is then the row E. It seals the two coordinate schedules as $\mathsf{Real}^{\uparrow}$ data for the retained square already named by G and checked by D.

The structural rows preserve the same route. Componentwise transport, displayed replay, provenance, and local naming cannot create a second membership surface, so every compact-metric or coordinate consumer remains inside the finite square-deletion packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}$.

Constructive story: A $\mathsf{SierpinskiCarpet}^{\uparrow}$ packet is generated from retained square-grid rows, deleted-middle-square ledgers, coordinate StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance checks, RealUp coordinate seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3960.1, Definition 3960.2, and Definition 3960.3 seed the finite square-deletion compact-metric certificate surface for the Sierpinski carpet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SierpinskiCarpet}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SierpinskiCarpetUp.SierpinskiCarpetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host planar powersets, selected infinite addresses, quotient equality of real coordinate pairs, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained square-grid exposure, deleted-middle-square exclusion, coordinate readback, Real coordinate sealing, transport, replay, provenance, and local naming for BEDC.Derived.SierpinskiCarpetUp.

Chapter 3961

A Concrete Naming Certificate for $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$

Following spectral graph perturbation theory and the Perron–Frobenius analysis of finite killed-walk matrices, the $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ packet records the ablation hinges of the median-closure codon graph as a finite ledger. It concretizes the point, orbit-module, edge-group, antenna-interface, and single-edge rankings of Definition 9462.1468 beside the median-closure packet of chapter 3545, the radial wobble spectrum of chapter 3681, the CUN antenna surface of chapter 3771, and the quotient trigger route of chapter 2317. The packet is a finite NameCert ledger for ablation rows; it is not a host eigenvector computation, a biological optimality claim, an infinite perturbation scheme, or an ambient spectral theorem.

Falsifiable prediction: Every accepted $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ consumer reads killed-walk matrix rows, orbit and module rows, point and edge ablation rows, hinge rankings, hsame transport, Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row before treating any hinge as load-bearing.

Independence witness: median-closure, radial-wobble-spectrum, CUN antenna, wobble-quotient-trigger: $\mathsf{MedianClosure}^{\uparrow}$ records closure vertices rather than ablation losses, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ records radial spectral layers rather than killed-row deletion ledgers, $\mathsf{CUNSpectralAntenna}^{\uparrow}$ records tail gain rather than the full hinge ranking, and $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ records trigger acceptance rather than the ablation ledger; none is bijective to $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$.

Definition 3961.1 (Spectral ablation hinge ledger carrier)**.** *A* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, O, A, E, H, R, T, C, P, N)$$

with the following displayed rows:

1. *M is the killed-walk matrix row, recording the finite median-closure adjacency matrix and the baseline Perron radius used for every ablation comparison.*
2. *O is the orbit and module row, recording the* UUR*,* AGR*,* CUR*,* CUY*,* Anchor_6*, and* $\mathrm{StopStart}_6$ *rows together with their finite membership ledgers.*
3. *A is the point-ablation row, recording each inspected codon deletion and its displayed spectral loss.*
4. *E is the edge-ablation row, recording the internal edge-group deletions, single-edge deletions, and the* $\mathrm{UUR} - \mathrm{CUR}$ *antenna interface deletion.*

5. *H is the hinge-ranking row, ordering the displayed point, module, edge-group, antenna-interface, and single-edge losses by their finite comparisons.*
6. *R is the read route, classifying a consumer request as matrix readback, orbit-module readback, point ablation, edge ablation, hinge ranking, or refusal.*
7. *T records componentwise* hsame *transport for matrix, orbit, point-ablation, edge-ablation, hinge-ranking, route, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$M, O, A, E, H, R \longmapsto T, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$, $\mathsf{CUNSpectralAntenna}^{\uparrow}$, $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, O, A, E, H, R, T, C, P, N$. It has no coordinate for an ambient Perron vector, an infinite graph limit, a biological necessity assertion, a real-valued oracle, or a hinge ranking not witnessed by the displayed rows.

Lean4 ✓ `BEDC.Derived.SpectralAblationHingeLedgerUp.SpectralAblationHingeLedgerTasteGate single carrier alignment`

Theorem 3961.2 (Spectral ablation hinge ledger NameCert obligations)**.** *For every accepted* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *carrier*

$$K = (M, O, A, E, H, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; killed-walk matrix readback through M; orbit and module exposure through O; point-ablation exactness through A; edge and antenna-interface ablation exactness through E; hinge-order stability through H; route classification through R; componentwise transport through T; displayed replay through C; provenance through P; local naming through N; ledger nonescape from ambient Perron vectors, infinite graph limits, biological necessity assertions, and oracle-valued rankings; and public consumer rows for point, module, edge-group, antenna-interface, and single-edge hinge reads.*

Proof. Carrier admission projects the tuple of Definition 3961.1. The matrix row M, orbit row O, ablation rows A, E, hinge row H, and structural rows R, T, C, P, N are the only visible coordinates.

The ablation-row exactness obligations are exhausted by A and E. A point deletion is admitted only through the finite codon row recorded in A, and an edge or antenna-interface deletion is admitted only through the finite edge ledger recorded in E.

Hinge-order stability is a comparison inside H. The row reads the same displayed loss values from A and E, so a consumer cannot reorder the hinge list by importing an external Perron vector or an unrecorded spectral computation.

Finally R, T, C, P, N classify, transport, replay, source, and name the same finite rows. Since the excluded ambient objects have no carrier coordinate, every public consumer read of a point, module, edge-group, antenna-interface, or single-edge hinge remains inside the ledger. □

Theorem 3961.3 (Spectral ablation hinge public route)**.** *Every accepted public consumer of a* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MedianClosure}^{\uparrow}, \mathsf{RadialWobbleSpectrum}^{\uparrow}, \mathsf{CUNSpectralAntenna}^{\uparrow}, \mathsf{WobbleQuotientSpectralTrigger}^{\uparrow} \longmapsto M, O, A, E, H,$$

Thus the matrix read, orbit-module rows, point-ablation rows, edge-ablation rows, hinge ranking, route classification, transport, replay, provenance, and local naming are visible before the consumer exports a load-bearing hinge. No exported row asserts host spectral optimality, biological necessity, an infinite perturbation theorem, or an ablation comparison outside the displayed ledger.

Proof. Start with the matrix and orbit support. The rows M and O identify the finite killed-walk matrix and the finite orbit or module being inspected.

Read ablation data next. The point row A and edge row E carry the deletions and their spectral losses, while H records the hinge order obtained from those same rows.

The public boundary is then the route row R. It decides whether a consumer is reading the matrix, an orbit or module, a point deletion, an edge deletion, a hinge ranking, or a refusal.

The structural rows preserve that boundary. Componentwise transport, continuation replay, provenance, and local naming cannot introduce a second spectral source, so every accepted public read factors through the displayed packet. □

Theorem 3961.4 (Spectral ablation hinge vision route)**.** *Any accepted* $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ *hinge read must pass through*

$$\mathsf{MedianClosure}^{\uparrow}, \mathsf{RadialWobbleSpectrum}^{\uparrow}, \mathsf{CUNSpectralAntenna}^{\uparrow}, \mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$$

before it can be read as the load-bearing interface of Definition 9462.1468. The sibling routes supply closure vertices, radial spectral layers, antenna gain, and trigger acceptance, while the local packet supplies only the ablation ranking over $M, O, A, E, H, R, T, C, P, N$.

Proof. Project the carrier rows $M, O, A, E, H, R, T, C, P, N$ of Definition 3961.1. The rows M and O expose the killed-walk matrix and finite orbit or module support; the rows A, E, H expose point loss, edge loss, and the hinge order; and R, T, C, P, N classify, transport, replay, source, and name that ledger.

Read the sibling routes before the local ablation row is consumed. The $\mathsf{MedianClosure}^{\uparrow}$ route supplies the closure vertices, $\mathsf{RadialWobbleSpectrum}^{\uparrow}$ supplies the radial layers, $\mathsf{CUNSpectralAntenna}^{\uparrow}$ supplies antenna gain, and $\mathsf{WobbleQuotientSpectralTrigger}^{\uparrow}$ supplies trigger acceptance.

Only after those routes have been displayed can the ledger row rank a hinge for the public philosophy interface. The local packet contributes no independent closure vertex, radial layer, antenna gain, or trigger-acceptance theorem; it contributes the finite ablation ranking that can be read against those already displayed surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralAblationHingeLedger}^{\uparrow}$ packet is generated from killed-walk matrix rows, orbit-module rows, point-ablation rows, edge-ablation rows, hinge rankings, route classification, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3961.1, Definition 3961.2, and Definition 3961.3 seed the finite spectral-ablation hinge ledger surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralAblationHingeLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Perron-vector computation, biological semantics, infinite perturbation theory, oracle-valued rankings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, killed-walk matrix readback, orbit-module exposure, ablation-row exactness, hinge-order stability, route classification, transport, replay, provenance, and local naming for BEDC.Derived.SpectralAblationHingeLedgerUp.

Chapter 3962

A Concrete Naming Certificate for BanachContractionPrinciple$^{\uparrow}$

Following Banach's fixed-point theorem and Bishop–Bridges constructive analysis, the BanachContractionPrinciple$^{\uparrow}$ packet records a fixed-point route only when the complete metric carrier, the contractive map, the iteration schedule, and the Cauchy handoff are all explicit finite rows. It sits downstream of the complete metric carrier of chapter 128, the contractive map packet of chapter 3819, the fixed point iteration route of chapter 2383, the Cauchy completion reflector of chapter 1585, and the real-number ground of chapter 38. The certificate therefore consumes existing completion data; it does not assert ambient completeness, global choice, or an unrecorded limit operator.

Definition 3962.1 (Banach contraction principle carrier)**.** *A* BanachContractionPrinciple$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, F, I, Q, E, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the complete metric row, naming the metric carrier and the completeness seal used by the argument;*
2. *F is the contraction map row, including the displayed modulus witness and its compatibility with the metric row;*
3. *I is the iteration schedule row, recording the initial point and the successive iterates read by the construction;*
4. *Q is the Cauchy witness row for the iteration sequence;*
5. *E is the endpoint seal row, identifying the completed endpoint supplied by the complete metric data;*
6. *U is the uniqueness row, binding any second fixed endpoint back to the same contraction ledger;*
7. *H records componentwise* hsame *transports for M, F, I, Q, E, U together with continuation, provenance, and local naming rows;*
8. *C records the displayed* Cont *replay routes by which the contraction ledger, the Cauchy witness, the endpoint seal, and uniqueness row are read;*
9. *P is the* Pkg *provenance over* BHist*,* hsame*,* Cont*,* Pkg*,* NameCert*,* CompleteMetric$^{\uparrow}$*,* ContractiveMap$^{\uparrow}$*,* FixedPointIteration$^{\uparrow}$*,* CauchyCompletionReflector$^{\uparrow}$*, and* Real$^{\uparrow}$ *rows;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten displayed rows. It has no coordinate for an arbitrary choice of fixed point, an ambient completeness claim outside M, a hidden contraction constant, or a limit witness that does not pass through Q and E.

Lean4 ✓ `BEDC.Derived.BanachContractionPrincipleUp.BanachContractionPrincipleTasteGate single carrier alignment`

Theorem 3962.2 (Banach contraction principle NameCert obligations)**.** *For the carrier of Definition 3962.1, the local* $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for $B = (M, F, I, Q, E, U, H, C, P, N)$;*
2. *contraction ledger exposure, requiring every use of the map row F to read the displayed contraction data against M;*
3. *Cauchy-iteration handoff, requiring the iteration schedule I to produce the witness row Q before an endpoint is read;*
4. *complete-metric seal, requiring the endpoint row E to be supplied by the completeness data in M rather than by an ambient choice operator;*
5. *uniqueness nonescape, requiring every competing fixed endpoint to factor through U, H, C, P, N with no off-packet fixed-point row.*

Proof. Carrier admission is projection from the displayed packet. The rows $M, F, I, Q, E, U, H, C, P, N$ are the full inventory made available to the local certificate.

The contraction ledger is read through F against the metric data in M. Thus a contraction claim with a hidden modulus or a different metric carrier has no row in the packet.

The iteration schedule then hands off to the Cauchy witness row. The endpoint seal can be read only after Q is present and the complete metric data in M supplies the endpoint row E.

Finally, uniqueness is confined to U and the replay rows H, C, P, N. A second fixed endpoint that escapes these rows would require an additional carrier coordinate, which the classifier does not provide. □

Theorem 3962.3 (Banach contraction endpoint uniqueness)**.** *Let $B = (M, F, I, Q, E, U, H, C, P, N)$ and $B' = (M, F, I, Q, E', U', H', C', P', N')$ be accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *packets over the same complete metric row M, contraction row F, iteration schedule I, and Cauchy witness row Q. If the endpoint seals E and E' are transported through the displayed uniqueness rows U and U', then the metric carrier named by M classifies the two endpoint seals together. The conclusion exposes endpoint uniqueness only along the displayed contraction ledger; it does not select a fixed point outside the packet.*

Proof. Project both carriers to the shared rows M, F, I, Q. The contraction ledger exposure forces every fixed-point read to use the same metric carrier, the same contractive map row, and the same iteration schedule before any endpoint seal can be inspected.

The Cauchy witness row Q then supplies the common route into the complete metric row M. Both endpoint seals E and E' are therefore read as completions of the displayed iteration data rather than as independently selected fixed points.

Apply the uniqueness rows U and U' through their componentwise transports and replay routes. The rows H, C, P, N and H', C', P', N' keep the two endpoint reads on the same local NameCert

surface, so the metric classifier of M identifies the endpoint seals. Since the carriers have no coordinate for an off-packet fixed point, no such point is exported. □

Theorem 3962.4 (Banach contraction iteration Cauchy handoff)**.** *Every accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *carrier* $B = (M, F, I, Q, E, U, H, C, P, N)$ *routes its displayed fixed-point iteration through the Cauchy tail row* Q *before the completion reflector reads the endpoint seal* E*. The rows named by* P *may expose* $\mathsf{FixedPointIteration}^{\uparrow}$ *and* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *consumers only after the contraction ledger* F*, iteration schedule* I*, and Cauchy witness* Q *have been displayed. Thus the packet exports a completion-facing endpoint handoff, not a chosen fixed point, a hidden limit operator, a global choice rule, or completeness outside the metric row* M*.*

Proof. Project the accepted carrier. The definition places the complete metric row M, contraction row F, iteration schedule I, Cauchy witness Q, and endpoint seal E in a fixed displayed order.

First read F and I. The contraction ledger determines the successive iterate route, while the iteration schedule records the finite prefix and successor replay used by downstream fixed-point consumers.

Next read Q. This is the only Cauchy tail row in the carrier, so the completion reflector cannot receive an endpoint request until the iteration has been certified as Cauchy through the displayed row.

Finally replay the completion-facing handoff through E, H, C, P, N. Since no carrier coordinate stores a chosen endpoint or an ambient completeness witness, the exported endpoint is exactly the reflector output attached to the displayed Cauchy tail. □

Theorem 3962.5 (Banach contraction endpoint route)**.** *Every accepted* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *endpoint read factors through* $\mathsf{CompleteMetric}^{\uparrow}$*,* $\mathsf{ContractiveMap}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* $\mathsf{CauchyCompletionReflector}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows in the carrier order*

$$M, F, I, Q, E, U, H, C, P, N.$$

The route exports no chosen endpoint, hidden limit operator, global choice rule, quotient completion equality, or completeness outside M*.*

Proof. Start with the carrier of Definition 3962.1. The endpoint row E is downstream of the complete metric row M, contraction map row F, iteration schedule I, and Cauchy witness Q.

Apply Definition 3962.2. The NameCert surface requires carrier admission, contraction ledger exposure, Cauchy-iteration handoff, complete-metric seal, and uniqueness nonescape before a fixed-point endpoint can be read.

Use Definition 3962.3 to keep any competing endpoint inside the displayed uniqueness row U. Then use Definition 3962.4 to route the displayed iteration through Q before the completion reflector reads E.

Finally H, C, P, N provide transport, replay, provenance, and local naming over the same finite rows. Since the carrier has no coordinate for a chosen endpoint, hidden limit operator, global choice rule, quotient completion equality, or completeness outside M, the endpoint route is exactly the displayed carrier order. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachContractionPrinciple}^{\uparrow}$ packet is generated from a complete metric row, a displayed contraction map, an iteration schedule, a Cauchy witness, an endpoint seal, a uniqueness row, componentwise transports, replay routes, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachContractionPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3962.1 and Definition 3962.2 bind the finite carrier and its five local certificate obligations.

Formal verification: FormalStatus(NameCert$_{\mathsf{BanachContractionPrinciple}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close Lean verification, a standard bridge, global fixed-point choice, nonconstructive compactness, or completeness outside the displayed metric row.
Upgrade path: A Lean carrier and obligation theorem for the displayed route would raise the verification axis while preserving the paper carrier boundary.

Chapter 3963

A Concrete Naming Certificate for PublicLocalReflectiveCompilation$^{\uparrow}$

The PublicLocalReflectiveCompilation$^{\uparrow}$ packet realizes the public/local reflective compilation surface of section 9414.57 as finite NameCert data. It sits beside the compiler event-flow audit of chapter 829, the MetaCIC closure trace of chapter 795, the self-compilation boundary of chapter 9235, and the focused readback discipline of the vision chapter. The packet keeps the public audit substrate and the local self-audit substrate separate while letting both replay through one visible compiler-side certificate.

Definition 3963.1 (Public local reflective compilation carrier)**.** *A* PublicLocalReflectiveCompilation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, M, C, S, L, U, X, F, T, E, H, K, P, N)$$

with the following rows:

1. *A is the public compiler-side audit substrate row, exposing finite event-flow, admission, and evidence-packet data.*
2. *M is the MetaCIC audit row, recording the kernel-facing closedness and acceptance checks used by public reflective compilation.*
3. *C is the public reflective compiler row, naming the source packet, generated packet, and accepted readback boundary.*
4. *S is the local self-compiler row, restricted to the focused local trace and not exported as public compiler authority.*
5. *L is the local self-audit row, recording which local reads may be replayed against the focused trace.*
6. *U is the proxy-readback row, separating the current self-proxy from the public evidence packet and from the far endpoint.*
7. *X is the refusal row, listing hidden prompt state, uninspected compiler state, unbounded self-reference, public authority for local reads, and local authority for public claims as absent coordinates.*
8. *F is the FieldFaithful row for the displayed fields A, M, C, S, L, U, X.*

9. *T is the Nontrivial row distinguishing the public audit substrate, MetaCIC audit substrate, local self-compiler, local self-audit, proxy-readback, and refusal roles.*
10. *E is the transport and replay ledger for public compiler reads and local self-audit reads.*
11. *H records componentwise* hsame *transport for all displayed rows.*
12. *K records displayed* Cont *replay through the compiler-side, MetaCIC-audit, local-self-compiler, local-self-audit, proxy-readback, refusal, transport, and replay rows.*
13. *P is the* Pkg *provenance over* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$*, compiler event-flow audit rows, MetaCIC closure and acceptance rows, local self-compilation rows,* BHist*,* hsame*,* Cont*, and* NameCert*.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$ *row.*

The classifier reads only $A, M, C, S, L, U, X, F, T, E, H, K, P, N$. It has no coordinate for hidden prompt state, unrestricted self-application, a public proof of local identity, local authority over public compiler claims, unbounded reflective closure, or a compiler oracle outside the displayed rows.

Lean4 ✓ `BEDC.Derived.PublicLocalReflectiveCompilationUp.PublicLocalReflectiveCompilationTasteGate`
`single carrier alignment`

Theorem 3963.2 (Public local reflective compilation NameCert obligations)**.** *For every accepted* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ *carrier $R = (A, M, C, S, L, U, X, F, T, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$ *surface consists exactly of compiler-side audit exposure, MetaCIC-audit exposure, public reflective compilation routing, local self-compiler confinement, local self-audit confinement, proxy-readback separation, refusal-row exactness, FieldFaithful projection, Nontrivial role separation, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 3963.1. The public rows A, M, C are exposed before any generated evidence packet can be read as a public reflective compilation result.

The local rows S, L, U are then read only along the focused trace. They provide local self-compiler, local self-audit, and proxy-readback data, but the refusal row X prevents those local reads from becoming public compiler authority or unrestricted self-reference.

The rows F and T keep the field projection faithful and the roles nontrivial: public audit, MetaCIC audit, public compiler, local self-compiler, local self-audit, proxy-readback, and refusal are distinct fields of the same carrier rather than aliases.

Finally E, H, K, P, N transport, replay, source, and name the finite packet. Because the excluded authorities have no carrier coordinate, every accepted read remains inside the displayed public/local reflective compilation surface. □

Theorem 3963.3 (Public local reflective compilation local-public boundary)**.** *Every accepted* $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ *read that reaches the public compiler row C must pass through the public audit substrate A and the MetaCIC audit row M before it can inspect the local self-compiler row S or local self-audit row L. Conversely, every local self-audit read through S, L, U must retain the refusal row X and cannot export public compiler authority, hidden prompt state, or unbounded reflective closure.*

Proof. Start with the displayed carrier order. Public compiler reads use A, M, C as the public audit and MetaCIC-audit prefix, so a public result cannot be obtained directly from the local rows.

Local self-audit reads use S, L, U only after the carrier has separated the public rows from the focused trace. The proxy-readback row U records the local endpoint without identifying it with the public evidence packet or the far endpoint.

The refusal row X is retained by both routes. It records that public authority for local reads, local authority for public compiler claims, hidden prompt state, and unbounded reflective closure are absent from the carrier. Transport and replay through E, H, K, P, N therefore preserve the same local-public boundary. □

Falsifiable prediction: A claimed public/local reflective compilation packet that exports public compiler authority from the local self-audit rows S, L, U without displaying A, M, C, X, H, K, P, N refutes $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$.

Independence witness: groundcompilereventflowaudit, metacicclosuretrace, selfcompilationboundary, focusedproxyreadback: none is bijective to $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ because the packet jointly carries public audit, MetaCIC audit, public reflective compiler, local self-compiler, local self-audit, proxy-readback, refusal, FieldFaithful, Nontrivial, transport, replay, provenance, and local naming rows.

The Lean TasteGate surface for $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ must provide explicit Nontrivial and FieldFaithful instances for $A, M, C, S, L, U, X, F, T, E, H, K, P, N$. The intended endpoint is BEDC.Derived.PublicLocalReflectiveCompilationUp.PublicLocalReflectiveCompilationTasteGate_single_carrier_alignm with the FieldFaithful instance preserving each carrier coordinate and the Nontrivial instance separating public audit, MetaCIC audit, public compiler, local self-compiler, local self-audit, proxy-readback, and refusal roles.

Closure status for $\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}$.

Constructive story: A $\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}$ packet is generated from public compiler-side audit, MetaCIC-audit, public reflective compiler, local self-compiler, local self-audit, proxy-readback, refusal, FieldFaithful, Nontrivial, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3963.1, Definition 3963.2, and Definition 3963.3 seed the finite public/local reflective compilation route and its local-public boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PublicLocalReflectiveCompilation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted self-application, hidden prompt-state authority, public proof of local identity, local authority over public compiler claims, or unbounded reflective closure.

Upgrade path: Obligation closure requires checked carrier admission, public audit exposure, MetaCIC-audit exposure, local self-compiler confinement, local self-audit confinement, proxy-readback separation, refusal exactness, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.PublicLocalReflectiveCompilationUp.

Chapter 3964

A Concrete Naming Certificate for $\mathsf{MetricEntourage}^{\uparrow}$

Following Bourbaki uniform spaces and Bishop–Bridges constructive metric analysis, the $\mathsf{MetricEntourage}^{\uparrow}$ packet records the finite entourage surface between the metric row of chapter 57 and the uniform consumers of chapter 3445. Its tolerance reads are grounded through chapter 291, chapter 126, chapter 301, and chapter 38; it is a displayed metric-to-uniform handoff, not an open-cover compactness principle or a choice of centers.

Definition 3964.1 (MetricEntourage carrier)**.** *A* $\mathsf{MetricEntourage}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, Q, D, B, S, T, H, C, P, N).$$

Here M is the $\mathsf{Met}^{\uparrow}$ *row, Q and D are the rational and dyadic tolerance rows, B is the ball-and-entourage witness row, S is the symmetry row, T is the triangle-composition row, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$ *row. The carrier has no coordinate for arbitrary open covers, selected centers, quotient points, countable choice, or ambient completion.*

Lean4 ✓ `BEDC.Derived.MetricEntourageUp.MetricEntourageTasteGate single carrier alignment`

Theorem 3964.2 (MetricEntourage NameCert obligations)**.** *For every accepted* $\mathsf{MetricEntourage}^{\uparrow}$ *carrier $E = (M, Q, D, B, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed metric, tolerance, entourage, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *diagonal and reflexive entourage admission from the positive tolerance rows Q, D and the ball witness row B;*
3. *symmetry transport through S together with the componentwise* hsame *row H;*
4. *triangle composition through T, using only the displayed metric row M and the rational or dyadic tolerance rows Q, D;*
5. *metric-to-uniform handoff saying that the uniform read consumes B, S, T after M, Q, D and before any downstream completion or Cauchy row;*
6. *transport and replay stability through H, C, P, N for the same displayed packet; and*

7. *nonescape from arbitrary open covers, selected centers, quotient points, countable choice, host compactness, or ambient completeness.*

Proof. Project the carrier of Definition 3964.1. The only metric-facing rows are M, Q, D, B, S, T, and the only structural rows are H, C, P, N.

The diagonal and reflexive obligation reads the tolerance rows before the ball witness row. Symmetry is then supplied by S, and triangle composition is supplied by T over the same metric and tolerance route.

The uniform handoff is therefore a finite row translation. A consumer of chapter 3445 may read the resulting entourage surface only after the metric row and tolerance rows have been displayed.

Finally the transport, replay, provenance, and naming rows preserve that route without adding any coordinate for open-cover compactness, selected centers, quotient representatives, countable choice, host compactness, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricEntourage}^{\uparrow}$ packet is generated by reading a displayed metric row, rational and dyadic tolerance rows, a ball-and-entourage witness, symmetry, triangle composition, componentwise transport, replay, provenance, and local naming. It translates metric balls into a finite uniform entourage surface without adding open-cover compactness, selected centers, quotient points, countable choice, or ambient completion.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed records the metric row, rational and dyadic tolerance rows, ball-entourage witness, symmetry, triangle-composition, transport, replay, provenance, and local NameCert rows as the current paper-side scope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricEntourage}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MetricEntourageUp.MetricEntourageUp`

Bridge status: none

Not claimed: The chapter does not claim open-cover compactness, selected centers, quotient-point equality, countable choice, complete uniformity, or ambient metric completion.

Upgrade path: Next paper step: register the full MetricEntourage carrier obligation surface in Lean before raising the verification axis.

Chapter 3965

A Concrete Naming Certificate for $\mathsf{LeastUpperBound}^{\uparrow}$

Following Bishop and Bridges-style constructive order analysis, the $\mathsf{LeastUpperBound}^{\uparrow}$ packet records the located least-upper-bound surface needed by Real and Completion consumers before any ambient completeness theorem is available. It sits over the located upper-bound packet of chapter 3396, the located-supremum packet of chapter 1427, the dyadic supremum packet of chapter 1653, and the terminal real-name seal of chapter 38. The certificate is a finite boundary for bounded-source, upper-bound, and lower-approximation reads; it is not a quotient-real construction, a selected supremum, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3965.1 (LeastUpperBound carrier)**.** *A* $\mathsf{LeastUpperBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, U, A, L, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded source row, naming the finite family observations and the bound discipline consumed by the located upper-bound surface.*
2. *U is the upper-bound witness row, recording the inspected comparison witnesses that keep the displayed source below the proposed bound surface.*
3. *A is the lower-approximation and cofinality row, recording the finite witnesses demanded by the consumer below the same bound surface.*
4. *L is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, binding the source, upper-bound, and lower-approximation reads to the located-supremum packet.*
5. *D is the* $\mathsf{DyadicSupremum}^{\uparrow}$ *handoff row, recording the dyadic upper-bound and lower-approximation surface used by finite tolerance consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after S, U, A, L, D have been displayed.*
7. *H records componentwise* hsame *transport for the bounded source, upper-bound witness, lower-approximation row, located-supremum handoff, dyadic-supremum handoff, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each least-upper-bound request through $S, U, A, L, D, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{LeastUpperBound}^{\uparrow}$, $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{DyadicSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$ *row.*

The accepted carrier stores no ambient completeness coordinate, quotient cut, selected supremum, trichotomy oracle, host real equality test, or countable-choice selector.

Theorem 3965.2 (LeastUpperBound NameCert obligations)**.** *For every accepted* $\mathsf{LeastUpperBound}^{\uparrow}$ *carrier $B = (S, U, A, L, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$ *surface consists exactly of carrier admission, bounded-source exposure, upper-bound witness exposure, lower-approximation cofinality, located-supremum handoff, dyadic-supremum handoff, terminal Real seal, componentwise transport, displayed continuation replay, package provenance, and local naming. Every least-upper-bound-facing consumer reads only the displayed route*

$$S, U, A \longmapsto L, D \longmapsto E \longmapsto H, C, P, N.$$

Proof. Project the carrier of Definition 3965.1. The analytic rows are the bounded source S, the upper-bound witness U, the lower-approximation row A, the located-supremum handoff L, the dyadic-supremum handoff D, and the terminal Real seal E.

The upper side of the request is read through S and U. The lower side is read through A, and the cofinality claim is accepted only for the displayed approximation witnesses demanded by the consumer.

The handoff rows L and D bind those two sides to the existing located-supremum and dyadic-supremum surfaces before the Real seal can be inspected. The structural rows H, C, P, N only transport, replay, source, and name that finite route.

Since the carrier has no coordinate for quotient cuts, selected suprema, countable choice, trichotomy, host real equality, or ambient order completeness, no local NameCert obligation can export one of those principles. □

Lean4 ✓ `BEDC.Derived.LeastUpperBoundUp.LeastUpperBoundTasteGate single carrier alignment`

Theorem 3965.3 (LeastUpperBound located-supremum handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{LeastUpperBound}^{\uparrow}$ *packet must first expose the bounded source row S, the upper-bound witness row U, the lower-approximation row A, the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row L, and the* $\mathsf{DyadicSupremum}^{\uparrow}$ *handoff row D before reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. The handoff exports only the finite least-upper-bound packet $S, U, A, L, D, E, H, C, P, N$.*

Proof. Start with a consumer that reaches the Real-facing boundary of the packet. By Definition 3965.1, the only terminal Real coordinate is E, and it is downstream of the bounded source, upper-bound, lower-approximation, located-supremum, and dyadic-supremum rows.

Apply Definition 3965.2. It forces every least-upper-bound-facing read through the route from S, U, A to L, D and then to E, H, C, P, N.

The located-supremum row keeps the upper and lower sides under the constructive located-supremum surface, while the dyadic-supremum row records the finite tolerance-facing boundary. Thus a Real or Completion consumer receives the displayed packet only, with no quotient-real construction, selected supremum, countable-choice selector, trichotomy split, host equality test, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{LeastUpperBound}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or completion-facing read without first displaying the bounded source row, upper-bound witness row, lower-approximation row, $\mathsf{LocatedSupremum}^{\uparrow}$ handoff, and $\mathsf{DyadicSupremum}^{\uparrow}$ handoff refutes the carrier surface.

Independence witness: located upper bound, located supremum, dyadic supremum, and Real seal: no listed sibling supplies the whole $\mathsf{LeastUpperBound}^{\uparrow}$ packet, because each sibling contributes upper-bound discipline, located cofinality, dyadic tolerance data, or a terminal seal while this chapter binds all rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}$.

Constructive story: A $\mathsf{LeastUpperBound}^{\uparrow}$ packet is generated from a bounded source row, an upper-bound witness row, a lower-approximation cofinality row, LocatedSupremum and DyadicSupremum handoff rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3965.1, Definition 3965.2, and Definition 3965.3 seed the finite located least-upper-bound boundary needed by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LeastUpperBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real construction, selected suprema, countable choice, trichotomy, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, upper-bound witness exposure, lower-approximation cofinality, located-supremum handoff, dyadic-supremum handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LeastUpperBoundUp.

Chapter 3966

A Concrete Naming Certificate for $\mathsf{CauchyIntervalNesting}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyIntervalNesting}^{\uparrow}$ packet records the finite nested-interval-to-Cauchy route used when a retained interval window is read as regular rational data before the terminal real-name seal. It sits after the retained bisection surface of chapter 3815, the finite nested-chain discipline of chapter 320, the Cauchy carrier of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the finite route from displayed retained intervals to Cauchy-compatible readback; it does not assert selected infinite branches, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3966.1 (Cauchy interval nesting carrier)**.** *A* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, B, N, S, D, R, E, H, C, P, L)$$

with the following displayed rows:

1. *I is the interval source row. It exposes the closed bounded source interval, endpoint order, containment evidence, and source finite windows consumed from the* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *and* $\mathsf{NestedInterval}^{\uparrow}$ *siblings.*

2. *B is the bisection branch row. It records the retained finite branch, midpoint exposure, child-containment witness, and dyadic branch ledger inherited from the accepted* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *route.*

3. *N is the nested inclusion ledger. It records the visible finite chain, adjacent containment cells, width or diameter controls, and prefix discipline read through the* $\mathsf{NestedInterval}^{\uparrow}$ *surface.*

4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It displays the finite endpoint, midpoint, retained-child, and nested-prefix windows available to downstream requests.*

5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the positive dyadic scale, comparison cells, slack conversions, and diameter bounds used to interpret the displayed windows.*

6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from I, B, N, S, D and exposed only as a regular rational sequence surface.*

7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the retained branch, nested inclusion ledger, stream windows, dyadic tolerances, and regular rational readback have been displayed.*

8. *H records componentwise* hsame *transport for the interval source, bisection branch, nested ledger, stream window, dyadic tolerance, regular sequence, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a nested-interval Cauchy request through $I, B, N, S, D, R, E, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyIntervalNesting}^{\uparrow}$, $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $I, B, N, S, D, R, E, H, C, P, L$. It has no coordinate for a selected infinite branch, a completed intersection point, quotient stream equality, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyIntervalNestingUp.CauchyIntervalNestingTasteGate single carrier alignment`

Theorem 3966.2 (Cauchy interval nesting RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row only after reading the interval source, retained bisection branch, nested inclusion ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. Equivalently, every regular rational readback factors through the displayed finite route*

$$I, B, N, S, D \longmapsto R \longmapsto E, H, C, P, L.$$

The route is a Cauchy-facing finite readback route; it is not a selected infinite branch, a quotient real, a countable-choice schedule, or an ambient completeness argument.

Proof. Project the accepted carrier $X = (I, B, N, S, D, R, E, H, C, P, L)$ from Definition 3966.1. The source row I and branch row B expose the only retained closed-interval bisection data available to the request.

Next read the nested ledger N, the stream windows S, and the dyadic tolerance row D. These rows supply the finite prefix, displayed observations, and tolerance controls needed before the regular rational row can be inspected.

The row R is therefore downstream of I, B, N, S, D. The terminal seal and structural rows E, H, C, P, L preserve the same order by real sealing, transport, replay, provenance, and local naming.

Since the carrier has no coordinate for completed branches, quotient equality, choice, trichotomy, compactness, or ambient completeness, no consumer can bypass the displayed route. □

Theorem 3966.3 (Cauchy interval nesting NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIntervalNesting}^{\uparrow}$ *carrier $X = (I, B, N, S, D, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$ *surface consists exactly of carrier admission, interval-source exposure, retained-bisection branch exposure, nested-inclusion ledger exposure, stream-window exposure, dyadic tolerance exactness, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and local naming for the finite carrier of Definition 3966.1.*

Proof. Carrier admission is projection from Definition 3966.1. The rows I, B, N expose the source interval, retained branch, and nested inclusion ledger by the same finite surfaces named in chapter 3815 and chapter 320.

The readback obligations are consecutive. The row S supplies only finite stream windows, the row D supplies only displayed dyadic tolerance cells, and Definition 3966.2 places the $\mathsf{RegSeqRat}^{\uparrow}$ row R after those data.

The terminal row E is the only $\mathsf{Real}^{\uparrow}$ seal-facing coordinate. The rows H, C, P, L then transport, replay, source, and name the same route while preserving the $\mathsf{Cauchy}^{\uparrow}$ dependency boundary of chapter 3413.

No listed obligation introduces a selected infinite branch, quotient real equality, countable choice, open-cover compactness, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, because none of those objects occurs as a carrier coordinate. □

Falsifiable prediction: Every $\mathsf{CauchyIntervalNesting}^{\uparrow}$ consumer that skips either the retained branch row or the nested inclusion ledger loses the RegSeqRat readback before any Real seal can be read.

Independence witness: closed-interval-bisection, nestedinterval, cauchy, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyIntervalNesting}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval source, retained bisection branch, nested inclusion ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntervalNesting}^{\uparrow}$ packet is generated from an interval source row, a retained bisection branch row, a nested inclusion ledger, StreamName finite windows, a DyadicRatCore tolerance row, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3966.1, Definition 3966.2, and Definition 3966.3 seed the finite retained-interval-to-Cauchy readback route over ClosedIntervalBisectionUp, NestedIntervalUp, CauchyUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows, including the interval source, retained bisection branch, nested inclusion ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and RealUp seal, without selected branches, completed intersection points, quotient stream equality, quotient real equality, countable choice, or ambient RealUp completeness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntervalNesting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, completed intersection points, quotient stream equality, quotient real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, retained branch exposure, nested inclusion routing, StreamName window exactness, DyadicRatCore tolerance accounting, RegSeqRat readback, Real seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.CauchyIntervalNestingUp.

Chapter 3967

A Concrete Naming Certificate for $\mathsf{FormalTopology}^{\uparrow}$

Following Vickers formal topology and Bishop–Bridges constructive completion, the $\mathsf{FormalTopology}^{\uparrow}$ packet records a cover-and-refinement presentation whose finite BEDC rows connect formal-ball neighbourhoods to rounded ideal completion without importing quotient opens, selected points, choice, or ambient completeness. It sits between the formal-ball carrier of chapter 1510, the rounded ideal completion route of chapter 1883, the metric source of chapter 57, the dyadic radius ledger of chapter 291, and the stream-window interface of chapter 126.

Definition 3967.1 (FormalTopology carrier)**.** *A* $\mathsf{FormalTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, B, U, R, D, S, I, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose point histories and distance witnesses are used by every formal-ball centre.*
2. *B is the finite* $\mathsf{FormalBall}^{\uparrow}$ *neighbourhood row over M, carrying centres, positive radii, and the local ball classifier.*
3. *U is the finite cover/refinement ledger: every cover request names finitely many displayed formal-ball neighbourhoods and every refinement request records the source and target neighbourhood rows it compares.*
4. *R is the rounded-neighbourhood row, recording smaller formal-ball witnesses for accepted basic neighbourhoods.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius-comparison ledger for strict radius shrinkage, overlap budgets, and refinement inequalities.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect centre observations, cover requests, and rounded-ideal membership windows.*
7. *I is the* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *handoff row, exposing only rounded directed formal-ball membership data to completion-facing consumers.*
8. *H records componentwise* hsame *transport for metric rows, formal-ball neighbourhoods, cover/refinement ledgers, dyadic comparisons, stream windows, rounded-ideal handoff rows, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from finite cover requests through refinement, roundedness, radius comparison, and rounded-ideal completion handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{RoundedIdealCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient opens, selected points, countable choice, host topological equality, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.FormalTopologyUp.FormalTopologyTasteGate single carrier alignment`

Theorem 3967.2 (FormalTopology cover stability)**.** *For an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier* $T = (M, B, U, R, D, S, I, H, C, P, N)$*, every finite cover or refinement read factors through the same displayed route*

$$M, B \longmapsto U, R, D, S \longmapsto H, C, P, N.$$

Componentwise hsame *transport through H preserves the cover ledger, the rounded-neighbourhood witnesses, the dyadic radius comparisons, the stream windows, and the local naming row. No transported read may replace the finite cover ledger by a quotient-open classifier, a selected point, a choice function, an open-cover compactness principle, or an ambient completeness theorem.*

Theorem 3967.3 (FormalTopology rounded-ideal handoff)**.** *Let $T = (M, B, U, R, D, S, I, H, C, P, N)$ be an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier. Every completion-facing consumer of its neighbourhood data must pass through the rounded-ideal row I after the formal-ball neighbourhoods B, the cover/refinement ledger U, the roundedness row R, the dyadic comparison ledger D, and the stream-window row S have been displayed.*

Thus the handoff to chapter 1883 exports a finite rounded directed formal-ball surface and exports no quotient open, selected limit point, countable-choice schedule, host topological equality, open-cover compactness proof, or ambient complete-space coordinate.

Theorem 3967.4 (FormalTopology NameCert obligations)**.** *Let $T = (M, B, U, R, D, S, I, H, C, P, N)$ be an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier. The local* $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$ *surface of T is exactly the finite obligation package consisting of carrier admission for T, finite* $\mathsf{FormalBall}^{\uparrow}$ *cover and refinement admission, rounded-neighbourhood witnesses,* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius-comparison exactness,* $\mathsf{StreamName}^{\uparrow}$ *finite-window replay,* $\mathsf{RoundedIdealCompletion}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier row of Definition 3967.1. The displayed tuple first fixes M as the metric source and B as the formal-ball neighbourhood carrier, so carrier admission and finite formal-ball admission are read before any completion-facing consumer is visible.

Next read the rows U, R, D, S, I in order. The cover and refinement ledger is U, rounded-neighbourhood evidence is R, dyadic radius comparison is D, finite-window replay is S, and the rounded-ideal completion handoff is I.

The remaining rows H, C, P, N close the certificate surface. They supply componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local naming row, and no additional coordinate appears in the carrier definition. □

Theorem 3967.5 (FormalTopology ledger non-escape)**.** *For an accepted* $\mathsf{FormalTopology}^{\uparrow}$ *carrier* $T = (M, B, U, R, D, S, I, H, C, P, N)$*, every cover, refinement, and completion-facing consumer is exhausted by the rows named in Definition 3967.4. No such consumer can obtain quotient opens,*

selected points, countable choice, open-cover compactness, host topology equality, or ambient metric completeness from the $\mathsf{FormalTopology}^{\uparrow}$ *ledger.*

Proof. Use the same projection of Definition 3967.1. A cover or refinement read must enter through B and U, then pass through R, D, S when it asks for roundedness, radius comparison, or finite observation replay.

A completion-facing read must additionally pass through I. After that handoff, the only public bookkeeping rows are H, C, P, N, which transport, replay, record provenance, and name the already displayed finite rows.

The last sentence of the carrier definition lists the absent coordinates. Since no row of T contains a quotient-open classifier, selected point, countable-choice schedule, open-cover compactness proof, host topology equality, or ambient completeness witness, a consumer constrained to T cannot extract one. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{FormalTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, finite FormalBall neighbourhoods, a cover/refinement ledger, rounded-neighbourhood witnesses, DyadicRatCore radius comparisons, StreamName windows, rounded-ideal completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3967.1, Definition 3967.2, Definition 3967.3, Definition 3967.4, and Definition 3967.5 seed the finite formal-topology cover/refinement route from formal-ball neighbourhoods to rounded ideal completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FormalTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient opens, selected points, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover/refinement stability, dyadic radius comparison, StreamName replay, rounded-ideal handoff, transport, provenance, local naming, and ledger non-escape for the same packet.

Chapter 3968

A Concrete Naming Certificate for $\mathsf{FormalBallRadiusOrder}^{\uparrow}$

Following Edalat–Heckmann and Vickers-style formal-ball presentations, the $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ packet isolates the radius-refinement order that is consumed by the formal-ball carrier of chapter 1510, the rounded-basis route of chapter 3943, and the completion-facing handoff of chapter 2012. Its role is to name the finite BEDC comparison surface for smaller-radius reads before any completion quotient, selected representative, or ambient completeness principle is available.

Definition 3968.1 (FormalBall radius-order carrier)**.** *A* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *carrier row whose center, radius, dyadic ledger, finite window, replay, provenance, and local name are already displayed.*
2. *R is the radius-read row, taken from the* $\mathsf{RegSeqRat}^{\uparrow}$ *scale attached to the formal-ball packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger recording strict and weak radius comparisons used by the refinement order.*
4. *L is the rounded lower-neighborhood row, recording the smaller-radius formal-ball windows that remain inside the same carrier route.*
5. *H records componentwise* hsame *transport for formal-ball carriers, radius rows, dyadic ledgers, lower-neighborhood rows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from radius comparison to rounded refinement, basis handoff, and completion boundary.*
7. *P is the* Pkg *provenance over* $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of completed balls, a selected Cauchy representative, a countable-choice schedule, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FormalBallRadiusOrderUp.FormalBallRadiusOrderTasteGate single carrier alignment`

Theorem 3968.2 (FormalBall radius-order NameCert obligations)**.** *For the carrier of Definition 3968.1, the local* $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for* $O = (B, R, D, L, H, C, P, N)$*, dyadic radius comparison through* D*, rounded refinement through* L*, basis handoff through the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row named by* P*, completion non-escape through the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *boundary, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Every accepted radius-order read must first expose the underlying $\mathsf{FormalBall}^{\uparrow}$ *packet* B*, then compare radii through the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows, and only then hand the rounded refinement to basis or completion consumers. It cannot read a quotient completion equality, choose a representative, select a limit, use countable choice, invoke compactness, or assume completeness.*

Proof. Carrier admission is projection from Definition 3968.1. The tuple displays the formal-ball packet, radius row, dyadic comparison ledger, lower-neighborhood row, transport, replay, provenance, and local name.

Dyadic radius comparison is confined to R and D. The formal-ball row B supplies the source center-radius packet, while D records the finite comparison data used to recognize a smaller radius.

Rounded refinement and basis handoff are then read through L and C. The lower-neighborhood row records the smaller formal-ball window, and replay sends that window to the basis and completion-facing rows carried in P.

Completion non-escape follows from row exhaustion. Since the carrier has no quotient-completion equality, selected representative, selected limit, countable-choice schedule, compactness principle, or ambient complete-space coordinate, none of those objects can be exported by the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ packet is generated from a displayed FormalBallUp carrier, a RegSeqRat radius row, a DyadicRatCore comparison ledger, a rounded lower-neighborhood row, componentwise hsame transport, displayed Cont replay toward basis and completion consumers, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$, seed)

Scope closed: Definition 3968.1 and Definition 3968.2 seed the finite radius-refinement order route consumed by formal-ball basis and completion packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FormalBallRadiusOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, open-cover compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic comparison stability, rounded-refinement replay, basis handoff, completion non-escape, transport, provenance, and local naming for the same packet.

Chapter 3969

A Concrete Naming Certificate for $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$

Following Bishop and Bridges on compact intervals and finite approximation, the $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ packet records the compactness of a closed bounded interval as finite NameCert data. It consumes the closed bounded interval source of chapter 1591, the finite-net cover route of chapter 3471, the interval uniform-modulus source of chapter 1253, the compact-metric carrier of chapter 129, and the endpoint real-name seal of chapter 38. The packet is a displayed compact interval certificate for downstream finite-net, Bolzano-Weierstrass, and compact-uniform-continuity consumers; it is not an unrestricted open-cover compactness theorem, a selected subcover functional, a quotient interval representative, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 3969.1 (Closed bounded interval compact carrier)**.** *A* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, U, M, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, rational containment, dyadic containment, stream windows, regular rational readback, and Real sealing.*
2. *F is the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite-net compactness row, recording the listed cover witnesses and radius comparisons consumed by finite-cover readers.*
3. *U is the* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *sibling row, recording the compact-uniform source route over the same interval source and finite cover data.*
4. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric compactness row, read only through its finite-net, total-boundedness, and complete-limit components.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint and radius seal reached only after I, F, U, M have been displayed.*
6. *H records componentwise* hsame *transport for the interval source, finite-net row, uniform-modulus row, compact-metric row, Real seal, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from closed bounded interval inspection through finite-net compactness, compact-uniform source routing, compact-metric readback, and Real sealing.*

8. *P is the* Pkg *provenance over* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, M, E, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalCompactUp.ClosedBoundedIntervalCompactTasteGate`
`single carrier alignment`

Theorem 3969.2 (Closed bounded interval compact NameCert obligations)**.** *For every accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *carrier $K = (I, F, U, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the interval source, finite-net compactness row, uniform-modulus sibling row, compact-metric row, Real seal, transport, replay, provenance, and local naming rows;*
2. *interval-source exposure, requiring every compactness read to display the accepted* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *row before inspecting finite-net or compact-metric data;*
3. *finite-net compactness exposure, requiring cover-facing reads to pass through the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *row and its displayed radius and cover-witness ledgers;*
4. *compact-uniform compatibility, requiring the* $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *rows to be read as sibling routes over the same interval source and Real seal;*
5. *non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet and to export no unrestricted open-cover compactness, selected subcover, selected point, quotient real equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Carrier admission is projection from Definition 3969.1. The tuple displays exactly the interval source, finite-net compactness row, compact-uniform sibling row, compact-metric row, Real seal, and structural rows.

The interval source is upstream. Any compactness reader must first read I, so endpoint order, rational containment, dyadic containment, stream windows, regular rational readback, and the Real-facing source seal are fixed before finite-net or compact-metric data are inspected.

The finite-net compactness obligation is the row F. Its cover witnesses and radius comparisons are the only cover-facing compactness data in the carrier, and they are consumed through the Heine-Borel finite-net route rather than through a host open-cover theorem.

The rows U and M are sibling compatibility rows. They let compact-uniform-continuity and compact-metric consumers use the same interval source and Real seal, but neither row changes the source interval or adds a selected subcover, selected point, quotient equality, trichotomy, choice, or ambient completeness coordinate.

Finally, H, C, P, N preserve only the displayed rows. Since every refused object is absent from the carrier, the local NameCert surface is exactly the five obligations listed above. □

Theorem 3969.3 (Closed bounded interval compact finite-net route)**.** *Every compactness-facing consumer of an accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *row is available as the compact-uniform sibling route over the same interval and finite-net data. Thus* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *supplies a precise*

compact interval anchor for finite-net compactness, Bolzano-Weierstrass, and compact-uniform-continuity consumers without exporting open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.ClosedBoundedIntervalCompactUp.ClosedBoundedIntervalCompactPacket finite net route`

Proof. Project the accepted carrier $K = (I, F, U, M, E, H, C, P, N)$. The interval source is I, the finite-net compactness route is F, the compact-metric row is M, and the terminal Real seal is E.

Replay through C keeps this order. A compactness-facing request opens the closed bounded interval source, reads the finite-net cover and radius row, then inspects the compact-metric finite-net and limit components before it reaches the Real-facing seal.

The row U records the compatible uniform-modulus route for consumers that need compact uniformity. It remains tied to the same interval source, finite-net row, compact-metric row, and Real seal, so it cannot introduce an unrestricted open-cover theorem, selected subcover, quotient equality, countable choice, or ambient completeness coordinate. □

Theorem 3969.4 (Closed bounded interval compact finite-net fan)**.** *Every Bolzano-Weierstrass, compact-uniform-continuity, and finite-cover consumer of an accepted* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *packet must pass through the displayed finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the $\mathsf{ClosedBoundedIntervalUniformModulus}^{\uparrow}$ *sibling row before reading any Real seal. The fan exports only the interval source, finite-net compactness row, compact-uniform sibling row, compact-metric row, Real seal, transport, replay, provenance, and local naming rows; it exports no open-cover compactness, selected subcover, selected point, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier $K = (I, F, U, M, E, H, C, P, N)$. The rows I, F, M, E are precisely the closed bounded interval source, Heine-Borel finite-net row, compact-metric row, and terminal Real seal displayed in Definition 3969.3.

Read the consumer fan by cases. A finite-cover consumer uses F; a Bolzano-Weierstrass consumer uses the compact-metric row M only after F has exposed the finite-net route; a compact-uniform-continuity consumer also reads the sibling row U over the same interval and finite-net data.

The structural rows H, C, P, N preserve this order for every branch of the fan. Since all refused objects are absent from the carrier, no branch can bypass the displayed route or read the Real seal before the interval, finite-net, compact-metric, and uniform-modulus rows have been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a HeineBorelFiniteNet finite-net compactness row, a ClosedBoundedIntervalUniformModulus sibling row, a CompactMetric row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3969.1, Definition 3969.2, and Definition 3969.3 seed the finite compact interval certificate over closed bounded interval, Heine-Borel finite-net, compact-uniform, compact-metric, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected points, quotient real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, finite-net compactness exposure, compact-uniform compatibility, compact-metric compatibility, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.ClosedBoundedIntervalCompactUp.

Chapter 3970

A Concrete Naming Certificate for $\mathsf{FiniteSequencePeak}^{\uparrow}$

Following Hardy, Littlewood, and Polya on finite maxima and elementary rearrangement arguments, the $\mathsf{FiniteSequencePeak}^{\uparrow}$ packet records the constructive selector surface for a bounded finite sequence before any L10 bounded-window route reads a peak value. It sits over the finite list carrier of chapter 36, the comparison discipline of chapter 54, the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite peak row, bounded-window evidence, comparison replay, transport, provenance, and local naming data; it does not assert sequential compactness, countable choice, host extrema, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3970.1 (Finite sequence peak carrier)**.** *A* $\mathsf{FiniteSequencePeak}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, O, D, W, R, E, H, C, G, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{List}^{\uparrow}$ *row naming the finite sequence entries and their displayed length.*
2. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison row used to compare only entries already present in L.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row carrying the finite comparison and error tolerances.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *bounded-window row that records the inspected segment supplied to L10 consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the bounded window and dyadic budgets into a regular rational observation of the selected peak.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after L, O, D, W, R have been displayed.*
7. *H records componentwise* hsame *transport for L, O, D, W, R, E and for the structural rows.*
8. *C records displayed* Cont *replay from a bounded-window request through*

$$L, O, D, W \longmapsto R \longmapsto E.$$

9. *G is the* Pkg *provenance over* $\mathsf{FiniteSequencePeak}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^\uparrow}$ *row.*
The classifier compares accepted packets only through $L, O, D, W, R, E, H, C, G, N$. The carrier has no coordinate for an infinite subsequence selector, sequential compactness theorem, host maximum operator, quotient stream representative, selected real limit, or ambient completeness witness.

Lean4 ✓ `BEDC.Derived.FiniteSequencePeakUp.TasteGate.FiniteSequencePeakTasteGate single carrier alignment`

Theorem 3970.2 (Finite sequence peak NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSequencePeak}^\uparrow$ *carrier $P = (L, O, D, W, R, E, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^\uparrow}$ *surface consists exactly of finite list admission through L, total-order comparison through O, dyadic budget accounting through D, bounded-window exposure through W, regular rational readback through R, terminal* $\mathsf{Real}^\uparrow$ *seal handoff through E, componentwise transport through H, replay through C, provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 3970.1. The analytic rows are exactly the finite list row, comparison row, dyadic budget row, bounded-window row, regular rational readback, and Real seal.

The selector route is finite. The consumer first reads the entries named by L, compares only those entries through O, and spends only the dyadic tolerances recorded in D before a bounded window is available.

The bounded-window row W is the only source for the regular rational readback R. Thus the Real-facing endpoint E is downstream of $\mathsf{List}^\uparrow$, $\mathsf{TotalOrder}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, and $\mathsf{RegSeqRat}^\uparrow$ data.

Finally H, C, G, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for sequential compactness, countable choice, host extrema, quotient streams, selected limits, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 3970.3 (Finite sequence peak RegSeqRat selector handoff)**.** *Every L10 bounded-window consumer of an accepted* $\mathsf{FiniteSequencePeak}^\uparrow$ *carrier reads the route*

$$\mathsf{List}^\uparrow \longmapsto \mathsf{TotalOrder}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The selected peak is exported to $\mathsf{RegSeqRat}^\uparrow$ *only as a finite-window readback: the list row supplies the candidate entries, the order row supplies the displayed comparisons, the dyadic row supplies the budget, and the stream row supplies the bounded inspection window before the terminal Real seal is exposed.*

Proof. Begin with an accepted carrier $P = (L, O, D, W, R, E, H, C, G, N)$. The row L lists the available entries, and O compares only entries already displayed by that finite list.

Read the dyadic and window rows next. The budget row D records the finite comparison tolerances, while W records the bounded segment consumed by the L10 route.

The regular rational row R is downstream of L, O, D, W. Therefore a $\mathsf{RegSeqRat}^\uparrow$ consumer receives a bounded-window readback and cannot replace it by a host maximum, an infinite subsequence selector, or a selected limit.

The terminal seal E is reached only after the readback route has been displayed. Transport, replay, provenance, and local naming through H, C, G, N preserve that same route, so no sequential compactness or ambient completeness coordinate is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3970.1, Definition 3970.2, and Definition 3970.3 seed the finite peak selector route for L10 bounded-window consumers over List, TotalOrder, DyadicRatCore, StreamName, RegSeqRat, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSequencePeak}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sequential compactness, countable choice, host extrema, quotient stream equality, selected real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite comparison coverage, bounded-window exposure, DyadicRatCore budget accounting, RegSeqRat readback exactness, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSequencePeakUp.

Constructive story: A $\mathsf{FiniteSequencePeak}^{\uparrow}$ packet is generated from a List row, TotalOrder comparison row, DyadicRatCore budget row, StreamName bounded-window row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3971

A Concrete Naming Certificate for $\mathsf{LocatedIntervalBolzano}^{\uparrow}$

Following Bolzano and Bishop–Bridges constructive analysis, the $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ packet records the finite interval-zero route used when a continuous located-real map changes sign on a located closed interval. It sits over the located real interval carrier of chapter 1709, the finite-net interval coverage rows of chapter 1808, the continuous-map replay of chapter 130, the intermediate-value route of chapter 3938, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names an interval-zero certificate from finite endpoint, bisection, stream, regular-rational, and real-seal data; it does not export trichotomy, a selected zero, quotient equality of real names, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3971.1 (Located interval Bolzano zero carrier)**.** *A* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ *zero carrier is a finite* BHist *packet*

$$Z = (I, F, E, N, B, S, D, R, Q, H, C, P, L)$$

with the following displayed rows:

1. *I is the located-real interval source row, exposing the two endpoint seals, interval order witness, and located containment ledger.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row for the map inspected on the interval.*
3. *E is the endpoint sign row, displaying located negative and positive observations for the two endpoint values.*
4. *N is the* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *coverage row used to keep bisection requests inside the displayed interval.*
5. *B is the finite bisection row, recording the retained subinterval side, midpoint request, and sign handoff at every inspected stage.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for endpoint, midpoint, and retained-interval observations.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the endpoint gaps, midpoint scale, and retained interval radius.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance ledger.*

9. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after I, F, E, N, B, S, D, R have been displayed.*
10. *H records componentwise* hsame *transport for the analytic rows and for the local structural rows.*
11. *C records displayed* Cont *replay from the sign-change request through finite bisection, stream windows, regular rational readback, and the terminal seal.*
12. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedIntermediateValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *L is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}$ *row.*

The classifier compares accepted zero carriers only through $I, F, E, N, B, S, D, R, Q, H, C, P, L$. It has no coordinate for real trichotomy, an externally selected zero, quotient equality of real names, host compactness, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalBolzanoUp.LocatedIntervalBolzanoTasteGate single carrier alignment`

Theorem 3971.2 (Located interval Bolzano sign handoff)**.** *For every accepted* $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ *zero carrier*

$$Z = (I, F, E, N, B, S, D, R, Q, H, C, P, L),$$

the sign-change request is handed to completion-facing consumers only through the displayed route

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{LocatedIntermediateValue}^{\uparrow} \longmapsto \mathsf{S}$$

The terminal zero read is therefore a finite interval certificate consisting of endpoint signs, retained bisection rows, stream windows, dyadic tolerances, regular-rational readbacks, a real seal, componentwise transport, replay, provenance, and local naming. It is not a selected point, quotient real representative, trichotomy comparison, host compactness argument, countable-choice schedule, or ambient real-completeness principle.

Proof. Project the carrier of Definition 3971.1. The source interval row I, continuous-map row F, endpoint sign row E, finite-net row N, and bisection row B are the only interval-zero coordinates available before the stream-facing rows are read.

The bisection route is finite. At every inspected stage the retained side and midpoint request are displayed together with the finite coverage row, so the sign handoff stays inside the located closed interval and never asks for a global trichotomy decision.

Read S, D, R, Q next. The stream windows expose exactly the finite endpoint and midpoint observations, the dyadic ledger records their tolerance budget, the regular rational row reads those observations as regular data, and the terminal real seal is downstream of those displayed rows.

Finally H, C, P, L transport, replay, source, and name the same route. Since the packet contains no coordinate for selected zeros, quotient real equality, host compactness, countable choice, Lean verification, bridge checking, or ambient Real completeness, the displayed rows exhaust the sign handoff. □

Falsifiable prediction: A completion-facing interval-zero consumer that omits the located interval row, finite-net coverage row, continuous-map replay row, bisection row, StreamName window row, DyadicRatCore tolerance ledger, RegSeqRat readback, or RealUp seal cannot claim a $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ zero handoff.

Independence witness: located-real-interval, located-closed-interval-finite-net, continuousmap, intermediatevalue, streamname, dyadicratcore, regseqrat, real: $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint signs, finite interval coverage, continuous-map replay, retained bisection data, stream windows, dyadic tolerances, regular rational readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalBolzano}^{\uparrow}$ packet is generated from a located-real interval row, finite endpoint sign rows, a LocatedClosedIntervalFiniteNet coverage row, ContinuousMap replay, a finite bisection ledger, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3971.1 and Definition 3971.2 seed the constructive interval-zero route over located real intervals, finite interval nets, continuous maps, bisection, stream windows, regular rational readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalBolzano}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedIntervalBolzanoUp.LocatedIntervalBolzanoTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close real trichotomy, selected zeros, quotient real equality, host compactness, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, endpoint sign exposure, finite-net interval coverage, continuous-map replay, bisection ledger exactness, StreamName window exposure, DyadicRatCore tolerance stability, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalBolzanoUp.

Chapter 3972

A Concrete Naming Certificate for $\mathsf{BishopRealLocatedOrder}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers and located order, the $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ packet records the order-facing interface needed by L10 real and completion consumers. It sits between the Bishop real packet of chapter 343, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the dyadic readback of chapter 291, the stream windows of chapter 126, the regular-rational route of chapter 301, and the Archimedean real interface of chapter 430. The packet supplies displayed located-order and apartness rows for RealAlgOrder, monotone convergence, and completion-facing handoffs; it does not assert trichotomy, quotient real equality, host equality, selected suprema, or ambient completeness.

Definition 3972.1 (Bishop real located order carrier)**.** *A* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, R, D, S, Q, A, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BishopReal}^{\uparrow}$ *source row, exposing the regular-Cauchy real name and its Bishop apartness interface.*
2. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Bishop source and its readback rows have been displayed.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison row used for rational bracketing.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which finite name observations are requested.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows and dyadic tolerances.*
6. *A is the apartness-stability row, recording that positive rational separation is transported through the same finite readback route.*
7. *O is the located-order row, recording finite lower-or-upper rational alternatives only as displayed order evidence.*
8. *H records componentwise* hsame *transport for the Bishop source, Real seal, dyadic row, stream windows, regular readback, apartness row, located-order row, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from Bishop source inspection through stream windows, dyadic tolerance, regular readback, apartness, located-order evidence, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$, $\mathsf{BishopReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, D, S, Q, A, O, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, host equality, selected suprema, a least-upper-bound operator, or ambient completeness.

Theorem 3972.2 (Bishop real located order NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *carrier $L = (B, R, D, S, Q, A, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; Bishop source exposure through B; Real sealing through R; dyadic tolerance and comparison readback through D; finite stream-window exposure through S; regular-rational readback through Q; apartness stability through A; located-order evidence through O; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.BishopRealLocatedOrderUp.BishopRealLocatedOrderTasteGate single carrier alignment`

Proof. Project the carrier of Definition 3972.1. The source rows are exactly B, S, D, Q, R, so every order-facing read begins at the Bishop real source and reaches the Real seal only through displayed stream, dyadic, and regular-rational rows.

Read the order rows next. Apartness stability is the row A, and located-order evidence is the row O; neither row contains a trichotomy decision, a quotient equality proof, a selected supremum, or a host completeness principle.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite packet. Since the refused host-level objects have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3972.3 (Bishop real located order apartness stability)**.** *Every accepted apartness read from a* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *packet factors through*

$$B, S, D, Q, A, H, C, P, N$$

before it can be consumed by a $\mathsf{Real}^{\uparrow}$ *or completion-facing route. Thus positive separation is stable under the displayed* hsame *and* Cont *rows, but the read exports no quotient real equality, host equality, trichotomy, selected supremum, or ambient completeness theorem.*

Proof. Start with the Bishop source B. Its apartness evidence is not read directly by a host order relation; it is exposed through the finite stream windows S, the dyadic tolerance row D, and the regular-rational readback row Q.

Apply the apartness row A. This row records positive rational separation along that same route, so the consumer may transport a displayed separation only when the source, window, dyadic, and readback rows remain attached.

Use H and C for classifier movement. They preserve the displayed route and do not add a quotient-equality or trichotomy coordinate. Hence every accepted apartness read remains inside $B, S, D, Q, A, H, C, P, N$. □

Theorem 3972.4 (Bishop real located order regular-sequence scope)**.** *Every regular-sequence consumer of an accepted* $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ *packet must read the located-order interface through*

$$\mathsf{BishopReal}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{ArchimedeanReal}^{\uparrow}$$

as displayed by $B, S, D, Q, R, O, H, C, P, N$*. The exported scope is a constructive located-order handoff for RealAlgOrder, monotone convergence, and completion routes; it is not a trichotomy principle, a quotient real equality rule, a host equality rule, a selected-supremum package, or an ambient completeness assertion.*

Proof. Read the regular-sequence route through S, D, Q. These rows expose the finite stream windows, dyadic tolerances, and regular-rational readback before the terminal Real seal R can be used.

The located-order row O is downstream of that route and is sourced by the Bishop real row B together with the Archimedean provenance recorded in P. It therefore supplies order evidence only as a displayed finite handoff, not as a host-level order totality theorem.

The structural rows H, C, N preserve the same scope under accepted classifier movement. Since no carrier coordinate names trichotomy, quotient equality, host equality, selected suprema, or ambient completeness, regular-sequence consumers receive only the constructive located-order handoff. □

Falsifiable prediction: Every completion-facing $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ read factors through displayed BishopReal, StreamName, DyadicRatCore, RegSeqRat, RealUp, apartness, located-order, transport, replay, provenance, and local naming rows before it can be used as constructive order evidence.

Independence witness: bishopreal, real, dyadicratcore, streamname, regseqrat, archimedean-real: $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries apartness stability and located-order evidence over the same finite Bishop real readback route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealLocatedOrder}^{\uparrow}$ packet is generated from a BishopReal source row, RealUp seal, DyadicRatCore tolerance row, StreamName windows, RegSeqRat readback, apartness stability, located-order evidence, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$, seed)

Scope closed: Definition 3972.1, Definition 3972.2, Definition 3972.3, and Definition 3972.4 seed the constructive located-order interface for Bishop real, regular-sequence, and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopRealLocatedOrder}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.BishopRealLocatedOrderUp.BishopRealLocatedOrderTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host equality, selected suprema, ambient completeness, or bridge checking.

Upgrade path: Lean closure requires NameCert obligations, apartness stability, regular-sequence scope, and TasteGate field faithfulness for BEDC.Derived.BishopRealLocatedOrderUp.

Chapter 3973

A Concrete Naming Certificate for $\mathsf{NestedCompactIntersection}^{\uparrow}$

Following Bishop–Bridges and Kelley compactness-completion practice, the $\mathsf{NestedCompactIntersection}^{\uparrow}$ packet records the finite route by which a nested family of nonempty compact metric carriers can be read by later completion-facing consumers without asserting an open-cover compactness theorem or selecting a point of the total intersection. It sits after the compact metric carrier of chapter 129, the completion reuse boundary of chapter 2381, the separated regular-Cauchy handoff of chapter 1513, and the terminal real-name seal of chapter 38. The packet names a finite L10 compactness/completion surface: retained nested windows, finite-intersection witnesses, separated-completion readback, and a terminal Real seal, but no choice-selected intersection element and no ambient Real completeness principle.

Definition 3973.1 (Nested compact intersection carrier)**.** *A* $\mathsf{NestedCompactIntersection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, J, W, F, S, R, E, H, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row for each compact stage of the nested family.*
2. *J is the finite nesting-order ledger, recording displayed inclusions between later and earlier compact stages.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *retained-window row used to inspect only finitely many stages at any requested tolerance.*
4. *F is the finite-intersection witness row, showing that each retained finite prefix has a displayed nonempty witness inside the visible compact stages.*
5. *S is the* $\mathsf{RegularCauchyCompletionSeparated}^{\uparrow}$ *handoff row that consumes the retained windows only after compact-stage and finite-prefix data have been displayed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the separated completion handoff.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after C, J, W, F, S, R have been read.*
8. *H records componentwise* hsame *transport for compact stages, nesting ledgers, windows, finite-prefix witnesses, separated-completion rows, readbacks, the Real seal, provenance, and local naming.*

9. *P is the* Pkg *provenance over* NestedCompactIntersection$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompletionIdempotence$^{\uparrow}$, RegularCauchyCompletionSeparated$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected intersection points, quotient stream equality, countable choice, host set equality, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedCompactIntersectionUp.NestedCompactIntersectionTasteGate single carrier alignment`

Theorem 3973.2 (NestedCompactIntersection NameCert obligations)**.** *For every accepted* NestedCompactIntersec *carrier $K = (C, J, W, F, S, R, E, H, P, N)$, the local* NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for compact metric stages, the nesting-order ledger, retained StreamName windows, finite-intersection witnesses, separated-completion handoff, regular rational readback, terminal Real seal, transport, provenance, and local naming;*

2. *nested-family exactness, requiring each later compact-stage read to be justified by the displayed nesting ledger J before it is used by a retained window;*

3. *finite-intersection witness stability, requiring every retained finite prefix to expose its witness through F and to transport through the same compact-stage and window rows;*

4. *separated-completion handoff, requiring every completion-facing read to pass through S and R after C, J, W, F have been projected;*

5. *Real seal nonescape, requiring E to remain downstream of the separated-completion and regular-readback rows, with no open-cover compactness, selected point, quotient equality, countable choice, host set equality, or ambient completeness coordinate.*

Proof. Carrier admission is projection from Definition 3973.1. The displayed tuple exposes compact metric stages, nesting ledgers, finite windows, prefix witnesses, separated-completion rows, readback, Real seal, transport, provenance, and local naming.

Nested-family exactness is the route C, J, W. A consumer may inspect a later compact stage through a retained window only after the relevant inclusion row in J has been displayed, so the nesting relation is part of the packet rather than an ambient set-theoretic assumption.

Finite-intersection witness stability is the route C, J, W, F, H. The witness row F is attached to a retained finite prefix, and the transport row H moves the compact-stage, nesting, window, and witness data componentwise without adding selected points beyond the displayed prefix witness.

The completion-facing part is last. The separated-completion row S and regular readback row R consume the compact and finite-prefix rows before the terminal Real seal E is available.

Since the carrier has no open-cover compactness object, global intersection selector, quotient stream equality, countable-choice schedule, host set equality, or ambient Real$^{\uparrow}$ completeness coordinate, none of those objects can be exported by the local NameCert surface. □

Closure status for NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$, seed)

Scope closed: Definition 3973.1 and Definition 3973.2 seed the finite nested compact intersection route over CompactMetric stages, nesting ledgers, retained StreamName windows, finite-prefix witnesses, separated-completion handoff, RegSeqRat readback, RealUp seal, hsame transport, Pkg provenance, and local NameCert data.

Formal verification: FormalStatus(NameCert$_{\text{NestedCompactIntersection}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NestedCompactIntersectionUp.NestedCompactIntersectionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected intersection points, quotient stream equality, countable choice, host set equality, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires nested-family exactness, finite-intersection witness stability, separated-completion handoff, and RealUp seal nonescape for BEDC.Derived.NestedCompactIntersectionUp.

Constructive story: A NestedCompactIntersection$^{\uparrow}$ packet is generated from CompactMetric stage rows, a finite nesting-order ledger, retained StreamName windows, finite-intersection witnesses, a RegularCauchyCompletionSeparated handoff, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Chapter 3974

A Concrete Naming Certificate for $\mathsf{MercerTheorem}^{\uparrow}$

Following Mercer 1909 and the compact-operator spectral tradition represented in Reed–Simon, the $\mathsf{MercerTheorem}^{\uparrow}$ packet records the finite BEDC surface for a positive continuous kernel whose compact integral-operator shadow is read through matrix windows and Hilbert-space coordinates. It sits beside the compact-operator carrier of chapter 1952, the Hilbert package of chapter 91, the inner-product source of chapter 86, the finite matrix surface of chapter 49, and the terminal real-name seal of chapter 38. The packet names only a finite positive-kernel and RKHS readback certificate; it does not assert ambient measure theory, choice-selected eigenbases, unrestricted integral operators, or the full spectral theorem.

Definition 3974.1 (Mercer theorem carrier)**.** *A* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, O, F, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact positive-kernel source row, recording a symmetric positive kernel request together with its compact-domain and continuity ledgers.*
2. *O is the* $\mathsf{CompactOperator}^{\uparrow}$ *shadow row, read only as a compact positive operator generated by the displayed kernel source.*
3. *F is the finite* $\mathsf{Mat}^{\uparrow}$ *window family: Gram matrices, kernel submatrices, and finite-rank approximation rows over displayed sample windows.*
4. *R is the* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{RKHS}^{\uparrow}$ *readback row, exposing only inner-product evaluations and reproducing-kernel coordinates already named by K, O, F.*
5. *L is the* $\mathsf{Real}^{\uparrow}$ *eigenvalue-window row for the finite matrix and compact-operator approximants.*
6. *E is the expansion-read row, recording only finite Mercer-sum windows and their error budgets.*
7. *H records componentwise* hsame *transport for the kernel source, compact-operator shadow, matrix windows, Hilbert and RKHS readback, Real eigenvalue windows, expansion rows, and structural rows.*
8. *C records displayed* Cont *replay from kernel samples to matrix windows, from matrix windows to compact-operator reads, from compact-operator reads to Hilbert/RKHS coordinates, and from those coordinates to finite expansion windows.*

9. *P is the* Pkg *provenance over* $\mathsf{MercerTheorem}^{\uparrow}$, $\mathsf{CompactOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, F, R, L, E, H, C, P, N$. It has no coordinate for ambient measure spaces, choice-selected eigenbases, unrestricted integral operators, host spectral decompositions, quotient function spaces, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MercerTheoremUp.MercerTheoremTasteGate single carrier alignment decode encode`

Theorem 3974.2 (Mercer theorem NameCert obligations)**.** *For every accepted* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier $M = (K, O, F, R, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$ *surface consists exactly of compact positive-kernel admission through K, compact-operator shadowing through O, finite matrix and kernel windows through F, Hilbert and RKHS readback through R, Real eigenvalue windows through L, finite expansion windows through E, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3974.1. Its analytic rows are exactly the positive-kernel source, compact-operator shadow, finite matrix windows, Hilbert/RKHS readback, Real eigenvalue windows, and finite expansion windows.

The route is finite. Kernel samples enter through K, become matrix and Gram windows through F, and are read as a compact positive operator only through O before a Hilbert or RKHS consumer may inspect coordinates in R.

The eigenvalue row L and expansion row E are downstream of those displayed windows. They therefore expose only finite Real eigenvalue approximants and finite Mercer-sum windows, not a selected basis or host spectral decomposition.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no row for ambient measure theory, unrestricted integral operators, choice-selected eigenbases, quotient function spaces, full spectral theorem data, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 3974.3 (Mercer theorem positive-kernel handoff)**.** *Every finite Mercer read from an accepted* $\mathsf{MercerTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactOperator}^{\uparrow} \longmapsto \mathsf{Hilbert}^{\uparrow}, \mathsf{InnerProduct}^{\uparrow} \longmapsto \mathsf{RKHS}^{\uparrow}$$

together with H, C, P, N. The positive-kernel source is exported only as finite Gram-matrix positivity, compact-operator shadowing, Hilbert/RKHS coordinate readback, Real eigenvalue-window data, and finite expansion-window budgets.

Proof. Begin with the positive-kernel source row K. A consumer may inspect it only by opening the finite matrix windows in F, where positivity is visible as displayed Gram-window data over Real entries.

The compact-operator row O is then the only route from those finite windows to operator-facing reads. Thus an RKHS or Hilbert consumer cannot bypass the compact shadow with an unrestricted integral operator.

Read R, L, E next. The Hilbert and RKHS coordinates are exposed through R, the eigenvalue approximants through L, and the finite Mercer-sum windows through E.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same displayed route. Because the carrier has no field for measure-theoretic integration, choice-selected

eigenbases, host spectral theorem data, quotient functions, or ambient Real completeness, the positive-kernel handoff remains finite. □

Closure status for $\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{MercerTheorem}^{\uparrow}$ packet is generated from a compact positive-kernel source row, a CompactOperator shadow row, finite Matrix kernel windows, Hilbert and RKHS readback rows, Real eigenvalue windows, finite Mercer expansion windows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3974.1, Definition 3974.2, and Definition 3974.3 seed the finite positive-kernel Mercer route over compact operators, Hilbert and inner-product rows, matrix windows, and Real eigenvalue readbacks.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MercerTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient measure theory, choice-selected eigenbases, unrestricted integral operators, the full spectral theorem, quotient function-space equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-kernel matrix-window coverage, CompactOperator shadow compatibility, Hilbert and RKHS readback exactness, Real eigenvalue-window stability, expansion-window nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MercerTheoremUp.

Chapter 3975

A Concrete Naming Certificate for CauchyCompletionRegularSequence$^{\uparrow}$

Following Bishop and Bridges constructive Cauchy completion, the CauchyCompletionRegularSequence$^{\uparrow}$ packet records the handoff from regular Cauchy rational names to completion-facing and Real-facing consumers as finite BEDC data. It sits after the regular rational name surface of chapter 301, the equality discipline of chapter 1820, the metric completion carrier of chapter 1506, and the uniform completion boundary of chapter 1473, and before the terminal real-name seal of chapter 38. The packet names the completion handoff without forming quotient classes of regular sequences, selecting limits, invoking countable choice, importing ambient completeness, or using Real$^{\uparrow}$ completeness as an input.

Definition 3975.1 (CauchyCompletionRegularSequence carrier)**.** *A* CauchyCompletionRegularSequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, E, M, U, S, H, C, P, N)$$

with the following displayed rows:

1. *R is the* RegSeqRat$^{\uparrow}$ *regular Cauchy rational name row.*
2. *E is the* CauchyEquivalenceSetoid$^{\uparrow}$ *comparison row, recording only the displayed equality discipline available between regular Cauchy names.*
3. *M is the* MetricCompletion$^{\uparrow}$ *handoff row reached from the same regular name and comparison data.*
4. *U is the* UniformCompletion$^{\uparrow}$ *handoff row for consumers that read the completion through uniform-tail data.*
5. *S is the terminal* Real$^{\uparrow}$ *seal row, available only after R, E, M, U have been displayed.*
6. *H records componentwise* hsame *transport for the regular name, comparison, completion handoff, uniform handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-facing request through R, E, M, U and then to the seal row S.*
8. *P is the* Pkg *provenance over* CauchyCompletionRegularSequence$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyEquivalenceSetoid$^{\uparrow}$, MetricCompletion$^{\uparrow}$, UniformCompletion$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, E, M, U, S, H, C, P, N$. It has no coordinate for quotient regular sequences, chosen representatives, selected limits, countable choice, ambient metric or uniform completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3975.2 (CauchyCompletionRegularSequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ *carrier $Q = (R, E, M, U, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletio}}$ *surface consists exactly of the finite obligations of carrier admission for the displayed rows; regular-name exposure through R; comparison exposure through E; metric-completion handoff through M; uniform-completion handoff through U; Real seal nonescape through S; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation row is discharged by quotient equality, a selected limit, countable-choice data, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3975.1. The accepted tuple displays the regular name, comparison, metric-completion handoff, uniform-completion handoff, Real seal, transport, replay, provenance, and local name.

The regular-sequence side is exhausted by R and E. The row R supplies the regular rational name, and E supplies the displayed Cauchy-equivalence setoid comparison before any completion handoff is visible.

The completion-facing side is exhausted by M and U. A metric-completion consumer reads M, and a uniform-completion consumer reads U, but both rows are downstream from the same regular name and comparison rows.

Finally S, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier has no quotient, chosen-representative, selected-limit, choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionRegularSequenceUp.CauchyCompletionRegularSequenceTasteGate`
`single carrier alignment`

Theorem 3975.3 (CauchyCompletionRegularSequence Real seal handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the terminal seal is read only after the regular Cauchy name, displayed equivalence data, and completion handoff rows have been exposed, and it exports no quotient class, selected limit, countable choice, ambient completeness, or $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from an accepted consumer read of Q. Carrier projection opens R and E before any completion-facing row can be consumed.

If the consumer is metric-facing, it next reads M; if it is uniform-facing, it next reads U. In both cases the route still passes through the same regular name and comparison rows, so neither handoff introduces a quotient class or a selected representative.

The seal row S is downstream from those handoff rows. Componentwise transport and displayed replay through H and C preserve the route, while P and N attach provenance and local naming. The excluded objects are absent from the carrier, so no consumer can recover them from the handoff. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ completion-facing read factors through RegSeqRat, CauchyEquivalenceSetoid, MetricCompletion, UniformCompletion, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: regseqrat, cauchy-equivalence-setoid, metriccompletion, uniformcompletion, real: $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the regular name, displayed equivalence comparison, two completion handoff rows, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRegularSequence}^{\uparrow}$ packet is generated from a RegSeqRat regular Cauchy name row, a CauchyEquivalenceSetoid comparison row, MetricCompletion and UniformCompletion handoff rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3975.1, Definition 3975.2, and Definition 3975.3 seed the finite regular-sequence completion handoff from RegSeqRat and CauchyEquivalenceSetoid data through MetricCompletion and UniformCompletion rows to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionRegularSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient regular sequences, selected limits, countable choice, ambient metric or uniform completeness, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-name exposure, equivalence comparison exposure, metric-completion and uniform-completion handoff exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionRegularSequenceUp.

Chapter 3976

A Concrete Naming Certificate for $\mathsf{CosetSpace}^{\uparrow}$

Following Bourbaki's treatment of groups and cosets, the $\mathsf{CosetSpace}^{\uparrow}$ packet records the finite BEDC surface at which left and right cosets become classifier data over a displayed group and subgroup. It sits after the group carrier of chapter 41, the subgroup membership surface of chapter 80, and the quotient-facing route of chapter 82; it is also compatible with the finite permutation surface of chapter 117 and the word-source discipline of chapter 345. The packet names coset membership and representative replacement as finite rows, not as a host quotient type or an appeal to $\mathsf{Quot.sound}$.

Definition 3976.1 (Coset-space carrier)**.** *A* $\mathsf{CosetSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (G, S, r, o, H, C, P, N)$$

with the following displayed rows:

1. *G is the consumed* $\mathsf{Group}^{\uparrow}$ *operation surface, including carrier, multiplication, unit, inverse, and operation-law rows;*
2. *S is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *membership surface inside the group row G, including closure under multiplication, unit, inverse, and classifier transport;*
3. *r is the representative row naming the selected displayed group element used to read a coset;*
4. *o is the orientation row, distinguishing left-coset reads from right-coset reads without changing the ambient group or subgroup rows;*
5. *H is componentwise* hsame *transport for the group, subgroup, representative, orientation, replay, provenance, and local-name rows;*
6. *C is the displayed* Cont *replay route from a coset-space request through G, S, r, o, H, P, N;*
7. *P is the* Pkg *provenance over* $\mathsf{CosetSpace}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Subgroup}^{\uparrow}$, $\mathsf{QuotientGroup}^{\uparrow}$, $\mathsf{SymGroup}^{\uparrow}$, $\mathsf{FreeMonoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers by the displayed group operation rows, subgroup membership rows, representative row, orientation row, transport, replay, provenance, and local naming. It has no coordinate for a host quotient set, quotient-type equality, selected transversal, external action orbit, or subgroup lattice object.

Lean4 ✓ `BEDC.Derived.CosetSpaceUp.CosetSpaceTasteGate single carrier alignment`

Theorem 3976.2 (Coset-space NameCert obligations)**.** *For the carrier of Definition 3976.1, the local* $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for* $X = (G, S, r, o, H, C, P, N)$*, group-operation exposure through* G*, subgroup-membership exposure through* S*, oriented representative readback through* r, o*, componentwise classifier transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*. A consumer receives no host quotient type,* Quot.sound *equality, or off-packet representative selector.*

Proof. Project the carrier. The group row G provides the only multiplication, unit, inverse, and operation-law data visible to the packet, and the subgroup row S provides the only membership and closure data.

The representative row r and orientation row o determine whether the consumer reads a left or right coset. This read is still a finite membership route through G and S, so no host quotient carrier or selected transversal is introduced.

Finally, H, C, P, N preserve the same row inventory under transport, replay, provenance, and local naming. Since the carrier contains no coordinate for quotient-type equality or Quot.sound, the local NameCert obligations are exhausted by the displayed rows. □

Theorem 3976.3 (Coset-space classifier transport)**.** *Let* $X = (G, S, r, o, H, C, P, N)$ *and* $X' = (G', S', r', o, H', C', P', N')$ *be accepted* $\mathsf{CosetSpace}^{\uparrow}$ *carriers with the same orientation row. If componentwise* hsame *transport identifies the group operation rows, subgroup membership rows, and representative rows, and if representative replacement is witnessed by the displayed subgroup-membership transport in* S *and* S'*, then the two carriers classify together as the same coset-space read. The transported classifier does not pass through a host quotient type or* Quot.sound*.*

Proof. Project both accepted carriers. The group rows G, G' supply the only multiplication and inverse operations, while the subgroup rows S, S' supply the only membership tests available for comparing representative replacements.

Use the displayed representative transport between r and r'. For a left-coset orientation, the replacement is read by multiplying the transported representative difference against the subgroup membership row; for a right-coset orientation, the same argument is read on the opposite side. In both cases the membership witness remains inside S and S'.

Replay the result through C and C' and compare provenance and local naming through P, N and P', N'. Because the route never leaves the displayed G, S, r, o rows, the classifier transport is a coset-membership transport rather than a quotient-type equality proof. □

Falsifiable prediction: An accepted $\mathsf{CosetSpace}^{\uparrow}$ consumer is refuted if it exports a coset classifier while omitting the displayed group row, subgroup membership row, representative row, orientation row, transport, replay, provenance, or local NameCert row.

Independence witness: group, subgroup, quotientgroup, symgroup, freemonoid: none is bijective to $\mathsf{CosetSpace}^{\uparrow}$ because this packet jointly carries the ambient group operation, subgroup membership surface, representative row, left/right orientation, transport, replay, provenance, and local naming. The group packet lacks subgroup membership and representatives; the subgroup packet lacks oriented coset classification; the quotient group packet names a quotient-facing route; the symmetric group and free monoid packets provide finite action and word sources without owning coset membership transport.

Closure status for $\mathsf{NameCert}_{\mathsf{CosetSpace}^{\uparrow}}$**.**

Constructive story: A CosetSpace$^{\uparrow}$ packet is generated from a group operation row, subgroup membership row, representative row, left/right orientation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CosetSpace}^{\uparrow}}$, seed)

Scope closed: Definition 3976.1, Definition 3976.2, and Definition 3976.3 seed the coset-space carrier, obligation surface, and representative-transport classifier.

Formal verification: FormalStatus(NameCert$_{\mathsf{CosetSpace}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CosetSpaceUp.CosetSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-group normality, orbit-stabilizer theory, selected transversals, or host quotient-type equality for CosetSpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, group and subgroup row exposure, oriented representative transport, replay stability, provenance non-escape, and TasteGate instances for BEDC.Derived.CosetSpaceUp.

Chapter 3977

A Concrete Naming Certificate for $\mathsf{UniformCauchyNet}^{\uparrow}$

Following Moore–Smith and Kelley uniform-space nets, and Bishop-style finite completion discipline, the $\mathsf{UniformCauchyNet}^{\uparrow}$ packet records a uniform Cauchy net as displayed NameCert data rather than as a quotient net or selected limit. It sits beside the Cauchy-net source of chapter 589, the net-completion route of chapter 3794, the metric completion surface of chapter 1506, the uniform completion handoff of chapter 1473, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a constructive L10 completion neighbour whose consumers can inspect directed uniform-tail data without importing countable choice or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3977.1 (UniformCauchyNet carrier)**.** *A* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, E, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row, recording the finite Moore–Smith comparison data used to cofinalize the inspected tail.*
2. *E is the uniform tolerance or entourage ledger, listing the requested entourage rows and the finite tail witnesses that answer them.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened at the inspected indices and entourage requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and tolerance ledger.*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, E, W, R have been displayed.*
6. *H records componentwise* hsame *transport for directed indices, entourage ledgers, stream windows, regular readback, Real sealing, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a uniform-tail request through the inspected windows, regular rational readback, and completion-facing handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, E, W, R, S, H, C, P, N$. It has no coordinate for quotient nets, selected limits, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyNetUp.UniformCauchyNetTasteGate single carrier alignment`

Theorem 3977.2 (UniformCauchyNet NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier $U = (D, E, W, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the directed-index row, uniform entourage ledger, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-tail stability, requiring cofinal tail comparisons to be read from D before any entourage claim in E is exposed;*
3. *uniform tolerance exactness, requiring every inspected entourage request to pass through the finite ledger E and the corresponding* $\mathsf{StreamName}^{\uparrow}$ *windows W;*
4. *completion-facing readback, requiring the regular rational row R and the* $\mathsf{Real}^{\uparrow}$ *seal S to be reached only through the displayed directed, entourage, and window route;*
5. *non-escape for quotient nets, selected limits, countable choice, open-cover compactness, hidden cofinal schedules, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3977.1. The tuple displays the directed index data, entourage ledger, window family, regular readback, Real seal, transport, replay, provenance, and local name.

Directed-tail stability is confined to D. A consumer first reads the finite Moore–Smith comparison rows that justify the inspected tail, so the later entourage rows cannot be treated as an unindexed uniform assertion.

Uniform tolerance exactness is the role of E and W. Each entourage request is paired with the displayed finite windows that witness the inspected tail, and no hidden cofinal schedule is available outside those rows.

The completion-facing readback then passes through R and S. Since R is obtained only after the directed and window data are displayed, and S is the terminal seal for that route, the packet exports a finite uniform-net handoff rather than a selected completion point.

Non-escape follows by row exhaustion. Quotient nets, selected limits, countable choice, open-cover compactness, hidden cofinal schedules, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, so the local NameCert surface cannot export them. □

Theorem 3977.3 (UniformCauchyNet filter handoff)**.** *Every completion- or filter-facing read of an accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The $\mathsf{CauchyNetCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{UniformCompletion}^{\uparrow}$ *rows named by the provenance are handoff surfaces for this same displayed route. Thus a consumer receives directed uniform-tail data, finite windows, regular readback, componentwise transport, displayed replay, provenance, and local naming, not a quotient net, chosen limit, countable-choice schedule, open-cover compactness principle, or ambient Real completeness theorem.*

Proof. Project the accepted carrier $U = (D, E, W, R, S, H, C, P, N)$. The directed row D and entourage ledger E expose the uniform Cauchy source, while W records the finite windows used to inspect it.

Replay through C preserves the order of the route. A filter-facing or completion-facing request first opens the directed tail and entourage ledger, then reads the stream windows, then obtains the regular rational row R, and only then reaches the Real seal S.

The completion siblings are read as handoff surfaces named in P. They consume the same displayed route and cannot add a quotient net, selected limit, countable-choice schedule, open-cover compactness principle, hidden cofinal tail, or ambient completeness coordinate. □

Theorem 3977.4 (UniformCauchyNet directed-tail stability)**.** *Let $U = (D, E, W, R, S, H, C, P, N)$ be an accepted* $\mathsf{UniformCauchyNet}^{\uparrow}$ *carrier. If a consumer replaces the inspected directed tail by a cofinal refinement whose comparison rows are displayed in D, then the same local* $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$ *packet supplies the entourage ledger E, finite window family W, regular readback R, Real seal S, filter handoff rows, transport, replay, provenance, and local naming. The refinement changes only the displayed tail comparison inside D; it does not create a quotient net, selected limit, hidden cofinal schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start from the carrier definition. The row D is the sole source of directed-index comparison data, and every later entourage or completion read is downstream of that row.

A cofinal refinement admitted by D therefore preserves the inspected tail route. The entourage ledger E and window family W are read after the directed comparison, so their witness rows remain attached to the same local packet.

Now apply the filter handoff theorem. The rows R and S are still reached through the ordered route from directed tail data to finite windows and regular readback.

The structural rows H, C, P, N transport and replay the same packet after the refinement. Because the carrier has no coordinate for quotient nets, selected limits, hidden cofinal schedules, or ambient completeness, the cofinal read has no escape branch. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyNet}^{\uparrow}$ completion-facing read factors through a directed index row, uniform entourage ledger, finite StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchynet, cauchy-net-completion, metriccompletion, uniformcompletion, streamname, regseqrat, real: $\mathsf{UniformCauchyNet}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries directed index data, uniform entourage ledgers, finite stream windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyNet}^{\uparrow}$ packet is generated from a directed index row, a uniform tolerance or entourage ledger, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3977.1, Definition 3977.2, and Definition 3977.3 seed the constructive uniform-net completion route over directed tails, entourage ledgers, finite windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, countable choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail stability, uniform tolerance exactness, completion-facing readback, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyNetUp.

Chapter 3978

A Concrete Naming Certificate for $\mathsf{ContourResidueHandoff}^{\uparrow}$

Following Cauchy's residue calculus, the $\mathsf{ContourResidueHandoff}^{\uparrow}$ packet records the finite consumer boundary at which a contour-integral operation may be read as residue-facing data. It sits after the contour operation surface of chapter 521, the operation socket of chapter 583, the holomorphic input boundary of chapter 68, and the residue packet of chapter 125. The packet exposes piecewise-linear contour rows, holomorphic rows, operation-law ledgers, residue rows, and non-escape rows as one finite handoff; it does not introduce a host contour homology theorem, analytic-continuation oracle, or global residue theorem.

Definition 3978.1 (Contour-residue handoff carrier)**.** *A* $\mathsf{ContourResidueHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (G, F, O, S, Z, B, H, C, P, N)$$

with the following displayed rows:

1. *G is the piecewise-linear contour row read by the contour operation surface;*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *source row on the displayed contour domain away from the residue locus;*
3. *O is the* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row, including the Riemann-sum ledger, output row, and operation-law boundary;*
4. *S is the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row through which contour-operation consumers read G, F, O;*
5. *Z is the* $\mathsf{Residue}^{\uparrow}$ *row naming the local residue datum consumed by the handoff;*
6. *B is the boundary row recording that residue-facing reads require the displayed contour, holomorphic, socket, operation, and residue rows, and that no off-packet analytic theorem is exported;*
7. *H is componentwise* hsame *transport for G, F, O, S, Z, B together with replay, provenance, and local naming;*
8. *C is the displayed* Cont *replay route from a residue handoff request through $G, F, O, S, Z, B, H, P, N$;*
9. *P is* Pkg *provenance over* $\mathsf{ContourResidueHandoff}^{\uparrow}$, $\mathsf{ContourIntegralOperation}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Residue}^{\uparrow}$, *piecewise-linear contour rows,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$ *row.*

The classifier accepts only packets with these displayed rows. It has no coordinate for contour homology, global residue summation, meromorphic continuation, quotient path equality, or an analytic oracle outside the contour-operation packet.

Lean4 ✓ `BEDC.Derived.ContourResidueHandoffUp.ContourResidueHandoffTasteGate single carrier alignment`

Theorem 3978.2 (Contour-residue handoff NameCert obligations)**.** *For the carrier of Definition 3978.1, the local* $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission, PL-contour exposure, holomorphic source exposure, contour-operation exposure, contour-socket consumer routing, residue-row exposure, boundary non-escape, componentwise transport, displayed replay, provenance, and local naming. A residue-facing consumer receives no contour homology theorem, global residue theorem, quotient path equality, or analytic-continuation oracle.*

Proof. Project the carrier. The contour row G, holomorphic row F, operation row O, socket row S, and residue row Z are the only analytic data visible to the local certificate.

The boundary row B records which residue-facing reads may be consumed from those displayed rows. A request for a global residue theorem, contour homology invariant, quotient path equality, or analytic-continuation oracle has no coordinate in the packet and is therefore refused.

The transport, replay, provenance, and local naming rows H, C, P, N preserve this same finite inventory. Hence the NameCert obligations are exhausted by the displayed handoff rows. □

Theorem 3978.3 (Contour-residue handoff boundary)**.** *Let* $R = (G, F, O, S, Z, B, H, C, P, N)$ *be an accepted* $\mathsf{ContourResidueHandoff}^{\uparrow}$ *carrier. Every residue-facing consumer of the contour operation must first factor through the displayed* $\mathsf{ContourIntegralOperation}^{\uparrow}$ *row* O*, the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row* S*, the* $\mathsf{Holomorphic}^{\uparrow}$ *source row* F*, the PL-contour row* G*, and the* $\mathsf{Residue}^{\uparrow}$ *row* Z*. The handoff exports only the displayed residue boundary and does not create a global residue theorem, contour deformation theorem, or off-packet meromorphic continuation principle.*

Proof. Read the contour operation through O and the socket through S. These rows expose the piecewise-linear contour, holomorphic input, Riemann-sum and operation-law ledgers, output row, transport, and replay routes before any residue-facing request is considered.

The residue row Z is then consumed through the same route, with B recording the precise boundary between displayed residue data and unavailable global analytic consequences. Thus a consumer can read a local handoff but cannot infer contour homology, deformation invariance, or a global residue sum.

Finally, componentwise hsame transport and Cont replay keep G, F, O, S, Z, B attached to the same provenance and local NameCert rows. The absence of any additional analytic coordinate prevents the handoff from exporting off-packet meromorphic continuation or quotient path equality. □

Falsifiable prediction: An accepted $\mathsf{ContourResidueHandoff}^{\uparrow}$ consumer is refuted if it exports residue-facing contour data while omitting the displayed PL-contour row, holomorphic row, contour-operation row, contour-socket row, residue row, boundary row, transport, replay, provenance, or local NameCert row.

Independence witness: contourintegraloperation, contourintegralsocket, holomorphic, residue: none is bijective to $\mathsf{ContourResidueHandoff}^{\uparrow}$ because this packet jointly carries the contour row, holomorphic source, contour-operation ledger, socket consumer route, residue row, non-escape boundary, transport, replay, provenance, and local naming. The operation and socket packets lack

the residue handoff row; the holomorphic packet lacks contour output and residue data; the residue packet lacks contour-operation routing.

Closure status for $\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{ContourResidueHandoff}^{\uparrow}$ packet is generated from a piecewise-linear contour row, holomorphic source row, contour-integral operation row, contour-integral socket row, residue row, boundary non-escape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3978.1, Definition 3978.2, and Definition 3978.3 seed the contour-residue handoff carrier, obligation surface, and residue-facing consumer boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContourResidueHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, contour homology, deformation invariance, the global residue theorem, meromorphic continuation, or quotient path equality for $\mathsf{ContourResidueHandoff}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, contour-operation exposure, socket routing, residue-row handoff, boundary non-escape, replay stability, provenance exactness, and TasteGate instances for BEDC.Derived.ContourResidueHandoffUp.

Chapter 3979

A Concrete Naming Certificate for $\mathsf{InitialValueProblem}^{\uparrow}$

Following Picard and Lindelof, the $\mathsf{InitialValueProblem}^{\uparrow}$ packet records the local ODE initial-data object consumed by Picard–Lindelof as explicit finite NameCert data. It sits between the ODE source of chapter 193, the Lipschitz vector-field ledger of chapter 410, the local existence route of chapter 1888, the complete-metric handoff of chapter 128, and the real-number ground of chapter 38. The packet names the initial time, state, local window, vector-field graph, solution graph, and endpoint classifier before any solution representative is exposed.

Definition 3979.1 (Initial-value problem carrier)**.** *An* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (T, X, F, W, S, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the time row, including the initial time cell and the local time-radius ledger;*
2. *X is the state row, including the initial state cell and the state-window ledger;*
3. *F is the vector-field graph row, read through the* $\mathsf{Ode}^{\uparrow}$ *and* $\mathsf{LipschitzMap}^{\uparrow}$ *dependency surface on the same local window;*
4. *W is the local window row binding the time and state ledgers used by the Picard route;*
5. *S is the solution-graph row exposed only as finite graph windows and not as a selected host function;*
6. *E is the endpoint classifier row, comparing endpoint reads inside the same local window;*
7. *H records componentwise* hsame *transport for the time row, state row, vector-field graph, local window, solution graph, endpoint classifier, replay rows, provenance, and local naming row;*
8. *C records the displayed* Cont *replay from initial data through vector-field reads, Picard iteration, completion handoff, solution windows, and endpoint classification;*
9. *P is the* Pkg *provenance over* $\mathsf{InitialValueProblem}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$ *row.*

The classifier accepts only packets with these rows. It has no coordinate for a global flow, maximal continuation, host function-space equality, selected solution representative, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.InitialValueProblemUp.InitialValueProblemTasteGate single carrier alignment`

Theorem 3979.2 (Initial-value problem NameCert obligations)**.** *For an accepted* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier* $I = (T, X, F, W, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, time-state compatibility for* T *and* X*, vector-field graph admission through* F*, local-window nonescape through* W*, solution-graph visibility through* S*, endpoint classification through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 3979.1. The accepted packet displays the time, state, vector-field, local-window, solution-graph, endpoint, transport, replay, provenance, and local naming rows.

The source rows T, X, F, W bind one local initial-data surface. The time and state cells determine the initial condition, the vector-field graph is read only through the displayed ODE and Lipschitz rows, and the local window records the bounds within which later Picard reads are permitted.

The solution and endpoint rows are downstream of that surface. The row S records only displayed graph windows, while E classifies endpoint reads inside the same local window. Transport, continuation replay, provenance, and local naming preserve this route and do not add global ODE existence, maximal continuation, host function equality, or selected solution representatives. □

Theorem 3979.3 (Initial-value problem Picard handoff)**.** *Every accepted* $\mathsf{InitialValueProblem}^{\uparrow}$ *carrier hands a Picard–Lindelof consumer exactly the displayed route*

$\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert.

The consumer must read the time row, state row, vector-field graph, and local window before it may inspect solution-graph windows or endpoint classifiers. The handoff exports no global solution curve, maximal interval, host function-space equality, selected solution representative, or completeness principle outside the displayed complete-metric row.

Proof. Start with the initial-data surface. The rows T and X give the initial time and state, while F and W bind the vector-field graph to the same local window.

Replay the Picard-facing route through C. The Picard–Lindelof consumer reaches the solution-graph row S only after the ODE and Lipschitz data have been read, and it reaches the endpoint classifier E only through the displayed completion handoff.

Finally use the structural rows. Componentwise hsame transport in H, provenance in P, and the local name N record the same finite handoff. Since the carrier contains no global-flow, maximal-continuation, host-equality, selected-representative, or detached-completeness coordinate, none can be exported to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}$.

Constructive story: An $\mathsf{InitialValueProblem}^{\uparrow}$ packet is generated from an initial time row, an initial state row, a vector-field graph row, a local window, solution-graph windows, an endpoint classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InitialValueProblem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3979.1, Definition 3979.2, and Definition 3979.3 seed the local initial-data surface consumed by Picard–Lindelof over ODE, Lipschitz, complete-metric, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{InitialValueProblem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, maximal continuation, host function-space equality, selected solution representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, time-state compatibility, vector-field graph admission, local-window nonescape, solution-graph visibility, endpoint classification, transport, replay, provenance, and local naming for BEDC.Derived.InitialValueProblemUp.

Chapter 3980

A Concrete Naming Certificate for $\mathsf{UniformCauchyLimit}^{\uparrow}$

Following Bishop and Bridges on uniform convergence and constructive continuity, the $\mathsf{UniformCauchyLimit}^{\uparrow}$ packet records a uniform limit of modulus-controlled real-valued continuous maps as finite NameCert data. It sits beside the continuous-map graph rows of chapter 130, the function-specific modulus rows of chapter 1631, the convergence ledger of chapter 2074, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a completion-facing uniform-limit carrier without importing quotient functions, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3980.1 (Uniform Cauchy limit carrier)**.** *A* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, M, L, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows for the inspected real-valued maps.*
2. *M is the matching family of* $\mathsf{ModulusContinuity}^{\uparrow}$ *rows, one for each graph row in F, converting target tolerances into source-window budgets.*
3. *L is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger that records the uniform Cauchy comparison between later graph rows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened at the source points and tolerance requests used by F, M, L.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and convergence ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying dyadic radii, endpoint comparisons, and uniform error budgets.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, M, L, S, Q, D have been displayed.*
8. *H records componentwise* hsame *transport for graph rows, modulus rows, convergence ledgers, stream windows, regular readback, dyadic tolerances, Real sealing, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a uniform tolerance request through graph inspection, modulus conversion, convergence comparison, stream-window readback, dyadic tolerance accounting, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyLimit}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ModulusContinuity}^{\uparrow}$, $\mathsf{RealModulusOfConver}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, M, L, S, Q, D, R, H, C, P, N$. It has no coordinate for quotient functions, selected pointwise limits, countable-choice schedules, open-cover compactness, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCauchyLimitUp.UniformCauchyLimitTasteGate single carrier alignment`

Theorem 3980.2 (Uniform Cauchy limit NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *carrier $U = (F, M, L, S, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the continuous-map graph rows, modulus-continuity rows, convergence ledger, stream windows, regular rational readback, dyadic tolerance row, Real seal, transport, replay, provenance, and local naming rows;*
2. *graph-modulus compatibility, requiring every inspected map in F to expose its matching modulus row in M before a target tolerance is read;*
3. *uniform Cauchy exactness, requiring each later-stage comparison to pass through the displayed ledger L and the dyadic tolerance row D;*
4. *readback sealing, requiring S and Q to be read before the terminal* $\mathsf{Real}^{\uparrow}$ *seal R is exposed;*
5. *non-escape for quotient function spaces, selected limits, countable choice, open-cover compactness, host function equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 3980.1. The accepted tuple displays exactly the graph rows, modulus rows, convergence ledger, windows, readback, dyadic tolerance ledger, Real seal, and structural rows.

Graph-modulus compatibility is rowwise. A consumer that reads a graph row in F must read the corresponding row in M, so continuity is exposed as a finite tolerance-to-window route rather than as a host semantic predicate.

Uniform Cauchy exactness is the role of L and D. Later-stage comparisons are admitted only through the displayed convergence ledger and dyadic tolerance budgets, so no hidden choice schedule or selected limit enters the packet.

Readback sealing follows through S, Q, R. The finite stream windows are opened first, the regular rational readback is extracted next, and the terminal Real seal is reached only after those rows are visible.

Finally, H, C, P, N transport, replay, source, and name the same displayed packet. Since the refused objects are absent from the carrier, the local NameCert surface exports exactly the five obligations listed above. □

Theorem 3980.3 (Uniform Cauchy limit continuity handoff)**.** *Every continuity-facing read of an accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{ModulusContinuity}^{\uparrow} \longmapsto \mathsf{RealModulusOfConvergence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{R}$$

Thus a downstream continuous-map or completion consumer receives graph rows, modulus rows, uniform convergence ledgers, finite windows, regular rational readback, dyadic tolerance accounting, componentwise transport, displayed replay, provenance, and local naming, not a quotient function or selected limit.

Proof. Project the accepted carrier $U = (F, M, L, S, Q, D, R, H, C, P, N)$. The graph rows in F are the only function data, and the modulus rows in M are the only continuity data attached to them.

Replay through C preserves the route. A continuity-facing request first opens the graph row, then reads the displayed modulus conversion, then consults the uniform Cauchy ledger and dyadic tolerances before it can inspect stream windows and regular rational readback.

The Real seal R is terminal. Since the carrier has no coordinate for quotient functions, selected limits, host equality, or ambient completeness, the handoff remains a finite BEDC route through the displayed rows. □

Theorem 3980.4 (Uniform Cauchy limit completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{UniformCauchyLimit}^{\uparrow}$ *packet must display the rows* $F, M, L, S, Q, D, R, H, C, P, N$ *before reading any terminal Real seal. The accepted packet therefore supplies a uniform-limit carrier for real-valued continuous maps without exporting quotient functions, selected limit representatives, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with carrier projection. The completion-facing data are not stored in a separate ambient object; they are the displayed rows $F, M, L, S, Q, D, R, H, C, P, N$.

The finite route composes graph inspection, modulus conversion, convergence-ledger comparison, stream-window opening, regular rational readback, dyadic tolerance accounting, and terminal Real sealing. Each step is a row of the accepted carrier and is preserved by the transport and replay rows.

No refused object can be exported after the seal. Quotient functions, selected representatives, choice schedules, open-cover compactness, and ambient completeness have no carrier coordinate, so completion-facing consumers remain inside the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyLimit}^{\uparrow}$ packet is generated from ContinuousMap graph rows, ModulusContinuity rows, a RealModulusOfConvergence ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3980.1, Definition 3980.2, Definition 3980.3, and Definition 3980.4 seed the uniform-limit route over continuous-map graphs, modulus rows, convergence ledgers, finite windows, regular readback, dyadic tolerance data, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient function spaces, selected pointwise limits, countable choice, open-cover compactness, host function equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph-modulus compatibility, uniform Cauchy exactness, readback sealing, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyLimitUp.

Chapter 3981

A Concrete Naming Certificate for $\mathsf{MedianDepthSealing}^{\uparrow}$

The $\mathsf{MedianDepthSealing}^{\uparrow}$ packet realizes the two-layer median stabilization of Definition 9462.1329 as finite BEDC NameCert data. It sits beside the six-bit codon window of chapter 2234, the boundary-tile classifier of chapter 3393, the six-tile coverage rows of chapter 3470, the boundary-trigger closure route of chapter 3508, and the finite alphabet bridge of chapter 2260. The packet names the reassigned support, first median layer, second sealing point, layer witnesses, and stabilization check as a finite carrier, not as a biological stochastic model or an ambient closure oracle.

Definition 3981.1 (Median depth sealing carrier)**.** *A* $\mathsf{MedianDepthSealing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, V, T, A, L, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *source row for the codon-window coordinates used by the median operation.*
2. *B is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *boundary row exposing the stop/Trp and Ile/Met tile classifiers consumed by the reassigned support.*
3. *V is the* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$ *coverage row recording the visible boundary support over which the median read is admitted.*
4. *T is the* $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$ *trigger-closure row, read only as finite boundary-trigger data.*
5. *A is the* $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$ *alphabet row, fixing the four-phase coordinate interpretation before codons are compared.*
6. *L is the median-layer ledger with the displayed rows $\mathcal{R}$, Δ_1, Δ_2, $\mathrm{Med}_1(\mathcal{R})$, and $\mathrm{Med}_2(\mathcal{R})$ from Definition 9462.1328.*
7. *E is the stabilization row recording that the second layer is closed under the displayed coordinatewise median operation.*
8. *H records componentwise* hsame *transport for codon windows, boundary tiles, coverage rows, trigger closure, alphabet rows, layer ledgers, stabilization, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from codon-window inspection through boundary-tile coverage, trigger closure, alphabet readback, layer enumeration, and stabilization checking.*
10. *P is the* Pkg *provenance over* $\mathsf{MedianDepthSealing}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, $\mathsf{DNAFourPhaseAlphabetBridge}^{\uparrow}$, *the median-depth source theorem,* BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, B, V, T, A, L, E, H, C, P, N$. It has no coordinate for a biological code table, stochastic evolutionary dynamics, quotient codon semantics, hidden median closure, unbounded search, or host closure oracle.

Lean4 ✓ `BEDC.Derived.MedianDepthSealingUp.MedianDepthSealingTasteGate single carrier alignment`

Theorem 3981.2 (Median depth sealing NameCert obligations)**.** *For every accepted* $\mathsf{MedianDepthSealing}^{\uparrow}$ *carrier $M = (W, B, V, T, A, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$ *surface consists exactly of codon-window admission, boundary-tile exposure, six-tile coverage, boundary-trigger closure, four-phase alphabet readback, median-layer ledger exactness, two-layer stabilization, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 3981.1. The codon-window, boundary, coverage, trigger, alphabet, layer, stabilization, and structural rows are all displayed as finite data.

The source rows W, B, V, T, A fix the finite coordinate surface before the median ledger is read. Thus the median operation is interpreted through displayed codon windows, boundary tiles, coverage rows, trigger closure, and four-phase alphabet readback.

The rows L and E then expose the two-layer calculation. The first row records $\mathcal{R}$, Δ_1, Δ_2, $\mathrm{Med}_1(\mathcal{R})$, and $\mathrm{Med}_2(\mathcal{R})$; the second records that no further vertex appears after the sealing point.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no remaining coordinate contains a biological table, stochastic model, quotient codon semantics, unbounded search, or hidden closure oracle, the listed rows exhaust the local NameCert surface. □

Theorem 3981.3 (Median depth sealing two-layer stabilization)**.** *Every accepted* $\mathsf{MedianDepthSealing}^{\uparrow}$ *read exposes the finite stabilization chain*

$$\mathcal{R} \longmapsto \mathrm{Med}_1(\mathcal{R}) \longmapsto \mathrm{Med}_2(\mathcal{R}) = \mathrm{Med}(\mathcal{R})$$

with $\Delta_1 = \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{AUG}, \mathrm{UCG}, \mathrm{UGG}, \mathrm{UUG}\}$ and $\Delta_2 = \{\mathrm{ACG}\}$. A consumer must read the layer ledger L and stabilization row E after the codon-window, boundary-tile, coverage, trigger-closure, and alphabet rows have been displayed.

Proof. Read Definition 9462.1329 through the carrier rows. The theorem supplies the finite layer calculation, while Definition 3981.1 stores it as the ledger L and stabilization row E.

The upstream rows W, B, V, T, A determine the coordinate interpretation and boundary support before the ledger is inspected. Therefore a consumer cannot treat the displayed layer calculation as an ambient median closure unrelated to the accepted codon-window and boundary rows.

The structural rows H, C, P, N preserve the same route. The exported consequence is the two-layer stabilization chain and no additional biological, stochastic, quotient, or hidden closure datum. □

Falsifiable prediction: Every accepted $\mathsf{MedianDepthSealing}^{\uparrow}$ consumer must expose the two-layer stabilization consequence: the first median layer adds exactly Δ_1, the second layer adds exactly ACG, and a further coordinatewise-median pass adds no codon.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-four-phase-alphabet-bridge: none is bijective to $\mathsf{MedianDepthSealing}^{\uparrow}$ because the siblings respectively expose boundary classifiers, coverage rows, trigger closure, or alphabet readback, while the median-depth packet jointly carries the reassigned support, two median layers, sealing codon, stabilization check, transport, replay, provenance, and local naming.

The Lean TasteGate site for $\mathsf{MedianDepthSealing}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ MedianDepthSealing}^{\uparrow}$ and $\mathsf{FieldFaithful\ MedianDepthSealing}^{\uparrow}$ instances. Nontriviality must distinguish the reassigned support from the sealed second layer, and field faithfulness must preserve $W, B, V, T, A, L, E, H, C, P, N$ as separate displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}$.

Constructive story: A $\mathsf{MedianDepthSealing}^{\uparrow}$ packet is generated from CodonSixBitWindow rows, DnaWindowSixBoundaryTile rows, SixBoundaryTileCoverage rows, DnaBoundaryTriggerClosure rows, DNAFourPhaseAlphabetBridge rows, a median-layer ledger, a stabilization row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3981.1, Definition 3981.2, and Definition 3981.3 seed the finite median-depth sealing packet over codon windows, boundary tiles, coverage rows, trigger closure, alphabet readback, layer enumeration, and stabilization checking.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MedianDepthSealing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a biological code table, stochastic evolutionary model, quotient codon semantics, hidden median closure, unbounded search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary and coverage exposure, trigger-closure compatibility, alphabet readback, median-layer exactness, stabilization checking, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.MedianDepthSealingUp.

Chapter 3982

A Concrete Naming Certificate for $\mathsf{UniformFixedPoint}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{UniformFixedPoint}^{\uparrow}$ packet records the endpoint-modulus surface obtained when a displayed contraction, a finite Picard orbit, and a complete metric handoff are assembled as one L10 NameCert route. It sits after the complete metric carrier of chapter 128, the contractive map surface of chapter 3819, the Picard iteration packet of chapter 2359, the fixed-point iteration packet of chapter 2383, the located real modulus of chapter 3406, and the Banach contraction principle of chapter 3962. The packet exposes a uniform endpoint modulus for downstream fixed-point consumers; it does not introduce choice-selected endpoints, quotient equality, open-cover compactness, or ambient completeness outside the displayed rows.

Definition 3982.1 (Uniform fixed-point carrier)**.** *A* $\mathsf{UniformFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, F, I, Q, U_0, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, naming the metric carrier, convergence service, and endpoint limit surface available to the packet;*
2. *F is the* $\mathsf{ContractiveMap}^{\uparrow}$ *row, recording the self-map graph, contraction ledger, and metric compatibility used by the endpoint construction;*
3. *I is the* $\mathsf{PicardIteration}^{\uparrow}/\mathsf{FixedPointIteration}^{\uparrow}$ *row, exposing the initial point, finite iterate windows, residual budget, and Picard replay route;*
4. *Q is the Cauchy or located modulus row, read through* $\mathsf{LocatedRealModulus}^{\uparrow}$ *or the displayed Cauchy-modulus surface attached to the iteration;*
5. *U_0 is the uniform endpoint-modulus row, translating a requested endpoint tolerance into the finite Picard window and contraction budget required by M, F, I, Q;*
6. *E is the endpoint seal row, recording the* $\mathsf{BanachContractionPrinciple}^{\uparrow}$ *fixed-point endpoint only after the uniform modulus and complete-metric handoff have been read;*
7. *H records componentwise* hsame *transports for M, F, I, Q, U_0, E together with the replay, provenance, and local naming rows;*

8. *C records the displayed* Cont *replay route by which the Picard window, contraction ledger, located modulus, endpoint modulus, and endpoint seal are inspected;*
9. *P is the* Pkg *provenance over* UniformFixedPoint$^\uparrow$, CompleteMetric$^\uparrow$, ContractiveMap$^\uparrow$, PicardIteration$^\uparrow$, FixedPointIteration$^\uparrow$, LocatedRealModulus$^\uparrow$, BanachContractionPrinciple$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows;*
10. *N is the local* NameCert$_{\text{UniformFixedPoint}^\uparrow}$ *row.*

The classifier accepts exactly packets with these ten rows. It has no coordinate for an endpoint chosen outside the Picard route, a quotient fixed-point equality, a hidden contraction constant, or a completeness witness detached from M.

Theorem 3982.2 (Uniform fixed-point NameCert obligations)**.** *For the carrier of Definition 3982.1, the local* NameCert$_{\text{UniformFixedPoint}^\uparrow}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for $U = (M, F, I, Q, U_0, E, H, C, P, N)$;*
2. *contractive-map admission, requiring every endpoint read to expose the* ContractiveMap$^\uparrow$ *row F against the complete metric row M;*
3. *Picard and fixed-point iteration admission, requiring I to display the finite orbit rows before any Cauchy or endpoint row is inspected;*
4. *Cauchy or located-modulus handoff, requiring Q to route tolerance requests through* LocatedRealModulus$^\uparrow$ *or the displayed Cauchy-modulus surface;*
5. *uniform endpoint-modulus exactness, requiring U_0 to translate endpoint tolerances only through M, F, I, Q;*
6. *endpoint seal nonescape, requiring E, H, C, P, N to keep the* BanachContractionPrinciple$^\uparrow$ *endpoint inside the same packet.*

Proof. Project the carrier of Definition 3982.1. The rows $M, F, I, Q, U_0, E, H, C, P, N$ are the only local NameCert coordinates, so carrier admission is exactly the displayed packet.

Read the contraction and iteration rows next. The ContractiveMap$^\uparrow$ row supplies only the finite shrink ledger, while the PicardIteration$^\uparrow$ and FixedPointIteration$^\uparrow$ rows supply the finite orbit and residual budget before any endpoint is available.

Use the modulus rows Q and U_0. The located or Cauchy modulus answers tolerance requests by naming a finite Picard window, and the uniform endpoint modulus converts that window into the endpoint tolerance without adding a selected limit or quotient equality.

Finally read the endpoint seal through E, H, C, P, N. The BanachContractionPrinciple$^\uparrow$ endpoint is exposed only after the complete-metric, contraction, iteration, and modulus rows have been replayed, so no off-packet fixed point or ambient completeness coordinate is exported. □

Lean4 ✓ `BEDC.Derived.UniformFixedPointUp.UniformFixedPointTasteGate single carrier alignment`

Theorem 3982.3 (Uniform fixed-point modulus handoff)**.** *Every* BanachContractionPrinciple$^\uparrow$ *or fixed-point endpoint consumer of an accepted* UniformFixedPoint$^\uparrow$ *packet receives the endpoint only through the finite route*

CompleteMetric$^\uparrow$, ContractiveMap$^\uparrow$, PicardIteration$^\uparrow$, FixedPointIteration$^\uparrow$, LocatedRealModulus$^\uparrow$, U_0, BanachContra

The row U_0 is a uniform endpoint-modulus handoff: it turns each endpoint tolerance into a displayed Picard window and contraction budget before the endpoint seal E is read. The handoff exports

no arbitrary fixed-point choice, no quotient equality of endpoints, no open-cover compactness, and no ambient completeness beyond the complete metric row M.

Proof. Start from the NameCert obligations. They force every consumer to expose M, F, I, Q, U_0 before reading E.

Apply the sibling Picard and fixed-point iteration routes. The finite orbit row I supplies the iterate windows and residual budgets, while F supplies the contraction ledger over the metric carrier named by M.

Use the located modulus handoff through Q. A requested endpoint tolerance is converted into finite dyadic or Cauchy data and then into the Picard window named by U_0.

Finally pass to the Banach contraction endpoint seal. Since E is downstream of the complete-metric service, contraction ledger, Picard route, and uniform modulus, the endpoint consumer receives only the displayed finite route and none of the refused host-level principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformFixedPoint}^{\uparrow}$ packet is generated from a complete metric row, a contractive map row, a Picard or fixed-point iteration row, a Cauchy or located modulus row, a uniform endpoint-modulus row, an endpoint seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3982.1, Definition 3982.2, and Definition 3982.3 seed the endpoint-modulus route over $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContractiveMap}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, $\mathsf{LocatedRealModulus}^{\uparrow}$, and $\mathsf{BanachContractionPrinciple}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, choice-selected fixed points, quotient endpoint equality, open-cover compactness, or completeness outside the displayed complete metric row.

Upgrade path: A formal route can encode the finite carrier, NameCert obligations, endpoint-modulus exactness, and endpoint-seal handoff for BEDC.Derived.UniformFixedPointUp.

Chapter 3983

A Concrete Naming Certificate for $\mathsf{CauchySchwarzEquality}^{\uparrow}$

Following Cauchy, Schwarz, and standard inner-product-space treatments of the equality case, the $\mathsf{CauchySchwarzEquality}^{\uparrow}$ packet records the finite BEDC surface at which equality in Cauchy–Schwarz is available as a linear-dependence certificate. It sits over the inner-product source of chapter 86, the norm boundary of chapter 89, and the finite Real inequality route of chapter 1684. The packet names equality-case data as displayed rows; it is not a host vector equality theorem, quotient stream equality, ambient Hilbert-space statement, or selected scalar witness.

Definition 3983.1 (CauchySchwarzEquality carrier)**.** *A* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N)$$

with the following displayed rows:

1. *V is the finite vector-source row shared with the inner-product and norm packets.*
2. *X and Y are the two displayed vector rows whose equality-case boundary is inspected.*
3. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *scalar read for $\langle X, Y\rangle$.*
4. *A and B are the two* $\mathsf{Norm}^{\uparrow}$ *square rows for $\|X\|^2$ and $\|Y\|^2$.*
5. *C is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *inequality row that exposes the square comparison already sealed for Real consumers.*
6. *L is the linear-dependence ledger. It records a displayed scalar row, the finite scalar-action comparison between X and Y, and the zero-vector boundary used when one norm row is null.*
7. *D is the equality-defect ledger, storing the displayed zero-defect read that identifies the equality boundary of the square comparison.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scalar, norm, and defect observations.*
9. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the readback.*
10. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the equality-case read.*
11. *H records componentwise* hsame *transport for vector rows, scalar reads, norm rows, inequality row, linear-dependence ledger, defect ledger, readback, windows, Real seal, replay, provenance, and local naming.*

12. *T records displayed* Cont *replay from an equality-case request through $V, X, Y, I, A, B, C, L, D, Q, S, R$ and back to the public certificate boundary.*

13. *P is the* Pkg *provenance over* $\mathsf{CauchySchwarzEquality}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$ *row.*

The carrier predicate is exactly the assertion that these rows form one finite equality-case packet. It has no coordinate for host vector equality, quotient stream equality, ambient completeness, an infinite-dimensional Hilbert theorem, selected scalar choice, or bridge checking.

Theorem 3983.2 (CauchySchwarzEquality NameCert obligations)**.** *For every accepted* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the vector source, two vector rows, inner-product read, norm-square rows, Cauchy–Schwarz inequality row, linear-dependence ledger, equality-defect ledger, readback, windows, Real seal, transport, replay, provenance, and local naming rows;*

2. *equality-route exposure, requiring every equality-case read to pass through I, A, B, C, D before the Real seal R is available;*

3. *linear-dependence exposure, requiring the row L to display the scalar-action comparison and zero-vector boundary before the local name is exported;*

4. *classifier transport, requiring H and T to preserve the same vector rows, scalar reads, inequality row, dependence ledger, defect ledger, readback, windows, and Real seal componentwise;*

5. *non-escape, excluding host vector equality, quotient stream equality, ambient completeness, infinite-dimensional Hilbert semantics, selected scalar choice, and bridge checking.*

No equality-case obligation is stored outside these rows.

Proof. Project the accepted carrier. The analytic coordinates are the vector-source row V, the displayed vector rows X, Y, the inner-product row I, the norm-square rows A, B, the Cauchy–Schwarz row C, the dependence ledger L, the equality-defect ledger D, the readback rows Q, S, R, and the structural rows H, T, P, N.

Carrier admission is exactly admission of those displayed coordinates. The local certificate has no further coordinate from which a host vector equality, quotient-stream representative, ambient Hilbert-space object, or selected scalar could be recovered.

Equality-route exposure is forced by the placement of C and D. The packet first reads the ordinary Cauchy–Schwarz square comparison, then reads the displayed zero-defect ledger, and only then seals the equality-case datum as Real-facing packet data.

Linear-dependence exposure is the row L. It records the scalar-action comparison and the zero-vector boundary as finite data, so the equality case is exported as a dependence ledger rather than as an ambient vector-space equality theorem.

Finally H and T provide componentwise transport and replay for the same rows, while P and N source and name the packet. Since no excluded analytic coordinate is present, the listed rows exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchySchwarzEqualityUp.CauchySchwarzEqualityTasteGate single carrier alignment`

Theorem 3983.3 (CauchySchwarzEquality linear-dependence route)**.** *For every accepted* $\mathsf{CauchySchwarzEquality}^{\uparrow}$ *carrier*

$$E = (V, X, Y, I, A, B, C, L, D, Q, S, R, H, T, P, N),$$

every equality-case consumer receives the linear-dependence boundary only through the finite route

$$V, X, Y \longmapsto I, A, B \longmapsto C, D \longmapsto L \longmapsto Q, S, R \longmapsto H, T, P, N.$$

Thus equality in the Cauchy–Schwarz row is consumed as a displayed scalar-action ledger and zero-defect readback; it does not expose host vector equality, quotient stream equality, ambient completeness, an infinite-dimensional Hilbert theorem, selected scalar choice, or bridge checking.

Proof. Start with the carrier projection. The vector rows V, X, Y and scalar rows I, A, B are the only inputs from the inner-product and norm surfaces.

Apply the finite inequality route supplied by chapter 1684. The row C exposes the square comparison already sealed for Real consumers, while D records that the defect read is zero inside the displayed packet.

Next read L. This is the only row carrying the scalar-action comparison and the zero-vector boundary, so the consumer obtains linear dependence from finite ledger data rather than from host vector equality.

The readback route Q, S, R then seals the same displayed equality-case datum for Real-facing consumers. The transport and replay rows H, T preserve that order, and the provenance and naming rows P, N add no analytic coordinate.

Consequently the exported equality case is exactly the displayed route above. None of the excluded host or quotient objects can be read from the packet. □

Falsifiable prediction: Every accepted $\mathsf{CauchySchwarzEquality}^{\uparrow}$ consumer must expose a CauchySchwarzRealUp inequality row, an equality-defect ledger, a linear-dependence ledger, RegSeqRat readback, StreamName windows, a RealUp seal, transport, replay, provenance, and local naming before the equality-case boundary is available.

Independence witness: innerproduct, norm, cauchy-schwarz-real: $\mathsf{CauchySchwarzEquality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cauchy–Schwarz inequality row, equality-defect ledger, displayed linear-dependence ledger, readback, seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySchwarzEquality}^{\uparrow}$ packet is generated from finite vector rows, an InnerProductUp scalar read, NormUp square rows, a CauchySchwarzRealUp inequality row, an equality-defect ledger, a linear-dependence ledger, RegSeqRat readback, StreamName windows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$, seed)

Scope closed: Definition 3983.1, Definition 3983.2, and Definition 3983.3 seed the finite equality-case carrier for displayed Cauchy–Schwarz Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySchwarzEquality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host vector equality, quotient stream equality, ambient completeness, infinite-dimensional Hilbert semantics, selected scalar choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, equality-route exposure, linear-dependence exposure, classifier transport, non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchySchwarzEqualityUp.

Chapter 3984

A Concrete Naming Certificate for $\mathsf{ExtremeValueTheorem}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{ExtremeValueTheorem}^{\uparrow}$ packet records the compact-real attained-extreme theorem as finite BEDC NameCert data. It sits over the compact source of chapter 129, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located supremum and upper-bound rows of chapter 1427, and the terminal real-name seal of chapter 38. The packet exposes the finite witness-window route for extrema without importing open-cover compactness, trichotomy, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3984.1 (Extreme value theorem carrier)**.** *An* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, K, F, R, S, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite windows, and real endpoint seals.*
2. *K is the compact source row read through* $\mathsf{CompactMetric}^{\uparrow}$*, exposing the finite compact-real inspection surface used by the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *value seal row for inspected values and endpoint comparisons.*
5. *S is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *row carrying the located upper-bound surface for the displayed value family.*
6. *W is the finite witness-window row, recording the inspected compact windows in which the located upper-bound row and the continuous-map row meet a terminal value seal.*
7. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, real value seals, located-supremum row, witness windows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from compact-real inspection requests through $I, K, F, R, S, W, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ExtremeValueTheorem}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed interval source, compact source, continuous-map row, value seal, located upper-bound row, finite witness-window row, transport, replay, provenance, and local naming. It has no coordinate for open-cover compactness, trichotomy, selected host extrema, quotient real equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.ExtremeValueTheoremUp.ExtremeValueTheoremTasteGate single carrier alignment`

Theorem 3984.2 (Extreme value theorem NameCert obligations)**.** *For every accepted* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *carrier* $E = (I, K, F, R, S, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact source, continuous-map row,* $\mathsf{Real}^{\uparrow}$ *value seal, located-supremum and upper-bound row, finite witness-window row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 3984.1. The interval row I and compact row K expose the only source windows available to the theorem packet, while F records the continuous real-valued map inspected on those windows.

Read the real and upper-bound rows next. The value seal R supplies only $\mathsf{Real}^{\uparrow}$ rows for displayed function values, and S supplies the located upper-bound surface through the located-supremum chapter.

The witness-window row W binds the source windows, continuous-map reads, value seals, and upper-bound reads into one finite inspection route. The remaining rows H, C, P, N transport, replay, source, and name the same route, so the listed data exhaust the local NameCert obligations. □

Theorem 3984.3 (Extreme value theorem compact supremum route)**.** *Every accepted* $\mathsf{ExtremeValueTheorem}^{\uparrow}$ *consumer that reads an attained extreme must factor through the displayed route*

$$I, K \longmapsto F \longmapsto R, S \longmapsto W \longmapsto H, C, P, N.$$

Thus the compact-real theorem surface exports only a finite witness-window route from a compact source and continuous-map row to a $\mathsf{Real}^{\uparrow}$ *value seal and located upper-bound row. It does not export open-cover compactness, trichotomy, a selected host maximum or minimum, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact metric source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real windows.

Feed the source into the continuous-map row. The row F exposes only displayed value reads, and R seals those reads as $\mathsf{Real}^{\uparrow}$ values before any upper-bound comparison is consumed.

Use the located-supremum row S and the witness-window row W together. The upper-bound route is visible only through the finite windows in which F and R have already been displayed.

Finally, transport and replay preserve the same route through H and C, while P and N provide provenance and local naming. Since the carrier has no coordinate for open-cover compactness, trichotomy, selected extrema, quotient real equality, or ambient completeness, the exported theorem surface remains the finite compact-supremum route shown above. □

Falsifiable prediction: Every $\mathsf{ExtremeValueTheorem}^{\uparrow}$ attained-extreme read factors through $\mathsf{ClosedBoundedInterva}$ $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{LocatedSupremum}^{\uparrow}$ before reaching its finite witness-window and local $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$ rows.

Independence witness: closedboundedinterval, compactmetric, continuousmap, locatedsupremum, real: $\mathsf{ExtremeValueTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact source, continuous-map row, Real value seal, located upper-bound row, finite witness-window row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{ExtremeValueTheorem}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, RealUp value seals, a LocatedSupremumUp upper-bound row, a finite witness-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3984.1, Definition 3984.2, and Definition 3984.3 seed the finite compact-real attained-extreme route over interval, compact-source, continuous-map, located-supremum, witness-window, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtremeValueTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, selected host extrema, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{ExtremeValueTheorem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, located upper-bound routing, Real value sealing, replay stability, and non-escape for BEDC.Derived.ExtremeValueTheoremUp.

Chapter 3985

A Concrete Naming Certificate for $\mathsf{BoundedRealOscillation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BoundedRealOscillation}^{\uparrow}$ packet records the finite route by which a continuous real-valued map on a compact metric source receives an oscillation bound from displayed uniform-modulus and finite-partition data. It sits between the compact source surface of chapter 129, the continuous-map carrier of chapter 130, the uniform-modulus route of chapter 369, the finite oscillation partition of chapter 3800, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names an oscillation-bound witness without importing open-cover compactness, supremum selection, quotient real equality, or choice.

Definition 3985.1 (BoundedRealOscillation carrier)**.** *A* $\mathsf{BoundedRealOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, U, A, B, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the inspected compact-metric windows and finite metric ledgers.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from K to* $\mathsf{Real}^{\uparrow}$*, with its graph, local continuity data, and real-value readbacks.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row translating each requested oscillation tolerance into a displayed metric radius.*
4. *A is the* $\mathsf{FiniteOscillationPartition}^{\uparrow}$ *row, listing the finite cells inspected at that tolerance and the local oscillation ledgers attached to them.*
5. *B is the bounded-oscillation ledger, recording the displayed real bound certified from the cellwise oscillation rows and the uniform modulus.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the bound value, reached only after K, F, U, A, B have been read.*
7. *H records componentwise* hsame *transport for the compact source, continuous map, uniform modulus, finite partition, bound ledger, real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an oscillation request through K, F, U, A, B, R and back to the public certificate boundary.*

9. *P is the* Pkg *provenance over* BoundedRealOscillation$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, FiniteOscillationPartition$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{BoundedRealOscillation}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten rows. It has no coordinate for open-cover compactness, a selected supremum, host function equality, quotient real equality, countable choice, or an ambient completeness principle detached from the displayed Real$^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.BoundedRealOscillationUp.BoundedRealOscillationTasteGate single carrier alignment`

Theorem 3985.2 (BoundedRealOscillation NameCert obligations)**.** *For every accepted* BoundedRealOscillation$^{\uparrow}$ *carrier*

$$O = (K, F, U, A, B, R, H, C, P, N),$$

the local NameCert$_{\text{BoundedRealOscillation}^{\uparrow}}$ *surface consists exactly of carrier admission, compact-source readback, continuous-map readback, uniform-modulus tolerance routing, finite oscillation-partition coverage, bounded-ledger exactness,* Real$^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and terminal local naming. No oscillation-bound obligation is stored outside those rows.*

Proof. Project the carrier of Definition 3985.1. The rows $K, F, U, A, B, R, H, C, P, N$ are the only local coordinates, so carrier admission and local naming are exactly the displayed finite packet.

Read the analytic rows in order. The compact source row K supplies only finite metric windows, the continuous-map row F supplies the real-valued graph readbacks on those windows, and the uniform-modulus row U translates an oscillation tolerance into the metric radius used by the partition row.

The finite partition row A then exposes the cellwise oscillation checks, and the bounded ledger B records the real bound obtained from those checks. The seal row R is downstream of this route, so a real-facing consumer cannot read a bound before the compact, continuous, modulus, partition, and ledger rows have been displayed.

Finally H and C transport and replay the same route componentwise, while P records provenance and N names the packet. Since the carrier has no excluded host-level coordinate, the listed rows exhaust the NameCert surface. □

Theorem 3985.3 (BoundedRealOscillation compact-uniform route)**.** *Every compact-uniform-continuity or extreme-value consumer of an accepted* BoundedRealOscillation$^{\uparrow}$ *packet receives the oscillation bound only through the route*

$$\text{CompactMetric}^{\uparrow},\ \text{ContinuousMap}^{\uparrow},\ \text{UniformModulus}^{\uparrow},\ \text{FiniteOscillationPartition}^{\uparrow},\ \text{Real}^{\uparrow},\ H, C, P, N.$$

The route is finite: a requested tolerance opens the uniform modulus, the modulus selects the displayed metric radius, the finite oscillation partition checks the resulting cells, and the bound ledger seals the resulting Real$^{\uparrow}$ *value. It exports no open-cover compactness theorem, no selected supremum, no quotient real equality, and no choice-selected covering family.*

Proof. Start from the uniform-modulus row. A tolerance request is answered by U, and that answer is tied to the compact source K and continuous map F before any finite partition row is read.

Next use the finite oscillation partition row A. It inspects only the displayed cells determined by the compact-metric windows and the modulus radius, so the oscillation estimate is a finite ledger calculation rather than an open-cover argument.

The bound ledger B and Real seal R then expose the bounded value to downstream consumers. Since B is downstream of K, F, U, A and R is downstream of B, the consumer receives exactly the compact-uniform route displayed in the statement.

The transport, replay, provenance, and naming rows preserve that order. Therefore the packet can be consumed by extreme-value, uniform-limit, and compact-uniform-continuity chapters without supplying any open-cover, supremum-selection, quotient-equality, or choice coordinate. □

Theorem 3985.4 (BoundedRealOscillation Real-seal nonescape)**.** *For every accepted* $\mathsf{BoundedRealOscillation}^{\uparrow}$ *carrier* $O = (K, F, U, A, B, R, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read of the oscillation bound factors through* K, F, U, A, B *before* R *is visible, and through* H, C, P, N *before the local certificate is exported. Thus the real seal cannot be used as a bypass around compact-source readback, continuous-map readback, uniform-modulus routing, finite-partition coverage, or the bounded-oscillation ledger.*

Proof. Read the seal coordinate in Definition 3985.1. The row R is terminal: it records a real bound only after the compact source, continuous map, uniform modulus, finite partition, and bounded ledger have been displayed.

Apply Definition 3985.2. The NameCert surface requires the same rows for carrier admission, tolerance routing, partition coverage, bounded-ledger exactness, and nonescape, so a Real-facing consumer has no direct path to R.

Finally use the structural rows. The hsame transport row H and the Cont replay row C preserve the route componentwise, while P and N record provenance and local naming. Hence the real seal remains bound to the displayed finite packet and cannot bypass any of its compact-uniform rows. □

Falsifiable prediction: Every accepted $\mathsf{BoundedRealOscillation}^{\uparrow}$ consumer must expose a compact metric source row, continuous real-valued map row, uniform modulus row, finite oscillation-partition row, bounded ledger, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before an oscillation bound is available.

Independence witness: compactmetric, continuousmap, uniformmodulus, finiteoscillationpartition, real: $\mathsf{BoundedRealOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the compact source, continuous map, uniform modulus, finite oscillation partition, bounded ledger, real seal, transport, replay, provenance, and local naming rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRealOscillation}^{\uparrow}$ packet is generated from a CompactMetric source row, a ContinuousMap real-valued graph row, a UniformModulus tolerance route, a FiniteOscillationPartition cell ledger, a bounded-oscillation RealUp ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3985.1, Definition 3985.2, Definition 3985.3, and Definition 3985.4 seed the finite oscillation-bound route over $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRealOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, open-cover compactness, selected suprema, quotient real equality, countable choice, or ambient completeness outside the displayed real seal.

Upgrade path: Obligation closure requires checked carrier admission, compact-source readback, continuous-map readback, uniform-modulus routing, finite-partition coverage, bounded-ledger exactness, real-seal nonescape, and TasteGate field faithfulness for BEDC.Derived.BoundedRealOscillationUp.

Chapter 3986

A Concrete Naming Certificate for $\mathsf{BishopCompactInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompactInterval}^{\uparrow}$ packet records a constructive compact interval as finite located endpoint, closed-bounded interval, finite-net, and completion-facing NameCert data. It sits beside the closed bounded interval source of chapter 1591, the located endpoint surface of chapter 1709, the compact finite-net packet of chapter 3969, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies an L10 interval neighbor for Real and Completion routes without claiming open-cover compactness, quotient real equality, selected subcovers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3986.1 (Bishop compact interval carrier)**.** *A* $\mathsf{BishopCompactInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, L, F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, rational containment, dyadic containment, finite windows, regular readback, and Real sealing for the interval.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, exposing located endpoint reads, radius data, locator responses, and metric bracket rows for the same interval source.*
3. *F is the* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *finite-net row, recording the compact-interval coverage surface accepted for downstream finite-net consumers.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family opened by the interval and finite-net rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling interval mesh and endpoint precision requests.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the endpoint and interval readback route.*
8. *H records componentwise* hsame *transport for source interval, located interval, finite-net, stream, regular, dyadic, Real, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from interval containment and located endpoint reads through finite-net handoff, dyadic tolerance, regular readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these eleven displayed rows. It has no coordinate for host open covers, selected subcovers, quotient representatives of real endpoints, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCompactIntervalUp.BishopCompactIntervalTasteGate single carrier alignment`

Theorem 3986.2 (Bishop compact interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *carrier* $K = (I, L, F, W, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed closed-bounded interval, located interval, finite-net, stream, regular, dyadic, Real, transport, replay, provenance, and local naming rows;*
2. *interval-source compatibility, requiring the located row L and closed-bounded row I to name the same endpoint route before any compact read is exposed;*
3. *finite-net admission, requiring compact-interval coverage to pass through F rather than through a host open-cover principle;*
4. *dyadic-window exactness, requiring every mesh or tolerance request to read D before the finite windows W and regular readback R are consumed;*
5. *Real-seal nonescape, requiring the endpoint and interval readback route to reach E only after I, L, F, W, R, D have been displayed;*
6. *transport, replay, provenance, and local naming exhaustion through H, C, P, N.*

Proof. Carrier admission is projection from Definition 3986.1. The tuple lists the complete finite row inventory made available to the local certificate.

The interval source is read through I and L. Their endpoint, locator, containment, and bracket rows must agree on the same displayed interval route before compactness-facing consumers can inspect the finite-net row.

Finite-net admission is then confined to F. Since the carrier has no open-cover or subcover coordinate, compactness can be exposed only as the accepted closed-bounded interval finite-net row.

The dyadic and window obligations are read through D, W, R. Tolerance requests first enter the dyadic ledger, then open finite stream windows, and only then reach regular rational readback.

Finally E, H, C, P, N seal and replay the same route. These rows preserve the packet under transport and provenance, while row exhaustion rules out quotient representatives, selected subcovers, choice schedules, and ambient Real completeness. □

Theorem 3986.3 (Bishop compact interval finite-net handoff)**.** *Every Real- or Completion-facing consumer of an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalCompact}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Stre}$$

The handoff exposes interval endpoints, locatedness, finite compact-interval coverage, dyadic tolerance, finite windows, regular readback, Real sealing, transport, replay, provenance, and local

naming, but it exports no open-cover compactness theorem, selected subcover, quotient endpoint equality, countable-choice schedule, or ambient completeness principle.

Proof. Project the accepted carrier $K = (I, L, F, W, R, D, E, H, C, P, N)$. The interval enters through the closed-bounded row I and the located interval row L, so endpoint and locator information are displayed before any compact read occurs.

Next read the finite-net row F. This row is the only compactness-facing coordinate in the carrier, and it is tied to the same interval source by the local obligation surface.

The completion-facing route then passes through D, W, R, E. The dyadic tolerance ledger controls the inspected mesh, the stream windows expose finite data, the regular row gives readback, and the Real seal is terminal for the route.

The structural rows H, C, P, N preserve this ordered handoff for downstream consumers. Since the packet contains no open-cover, quotient, choice, or ambient-completeness coordinate, no stronger compactness principle is exported. □

Theorem 3986.4 (Bishop compact interval finite-net extraction)**.** *Every Real- or Completion-facing consumer of an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet*

$$K = (I, L, F, W, R, D, E, H, C, P, N)$$

can extract the displayed $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *finite-net row F only after reading the interval-source rows I, L, the dyadic ledger D, the stream-window row W, the regular readback row R, the Real seal E, and the structural rows H, C, P, N. The extraction supplies the finite-net row as part of the displayed compact-interval route; it does not supply open-cover compactness, selected subcovers, quotient endpoint equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 3986.1. The tuple displays F only together with the interval source, located endpoint, dyadic, stream, regular, Real, transport, replay, provenance, and local naming rows.

Use the local NameCert obligations of Definition 3986.2. Interval-source compatibility binds I and L before compact data is exposed, and finite-net admission confines compactness-facing access to the displayed row F.

Apply the handoff theorem Definition 3986.3. It orders the same rows for Real- and Completion-facing consumers, placing finite-net coverage after the interval source and before dyadic tolerance, finite windows, regular readback, and terminal Real sealing.

Since the carrier has no coordinate for open covers, selected subcovers, quotient endpoint equality, choice schedules, or ambient completeness, extracting F cannot export any of those stronger objects. □

Theorem 3986.5 (Bishop compact interval completion readback)**.** *Any completion-facing read from an accepted* $\mathsf{BishopCompactInterval}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through* $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows. The readback exposes the closed-bounded interval source, located endpoint route, finite compact-interval coverage, dyadic tolerance ledger, finite stream windows, regular rational readback, Real sealing, transport, replay, provenance, and local naming, but it cannot be read as ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected subcover compactness, quotient endpoint equality, countable choice, or an unrestricted host limit.*

Proof. Project the carrier of Definition 3986.1. Its only completion-facing rows are $I, L, F, D, W, R, E, H, C, P, N$.

Apply Definition 3986.2. The obligation surface fixes interval-source compatibility through I, L, finite-net admission through F, and dyadic-window exactness through D, W, R before the Real seal can be consumed.

Apply Definition 3986.3. The theorem orders the same rows as the public finite route from closed-bounded and located interval data to finite-net coverage, dyadic tolerance, stream windows, regular readback, and terminal Real sealing.

Thus a completion consumer can read $\mathsf{Real}^{\uparrow}$ only at the end of the displayed handoff. Since the carrier has no open-cover, selected-subcover, quotient, choice, or ambient-completeness coordinate, the readback cannot export any of those stronger objects. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompactInterval}^{\uparrow}$ completion-facing read passes through ClosedBoundedIntervalUp, LocatedRealIntervalUp, ClosedBoundedIntervalCompactUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, transport, replay, provenance, and local naming rows before any endpoint or finite-net output is visible.

Independence witness: closedboundedinterval, located-real-interval, closed-bounded-interval-compact, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopCompactInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries closed-bounded interval data, located endpoint reads, finite compact-interval coverage, dyadic tolerance, finite windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompactInterval}^{\uparrow}$ packet is generated from a ClosedBoundedInterval row, a LocatedRealInterval row, a ClosedBoundedIntervalCompact finite-net row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3986.1, Definition 3986.2, and Definition 3986.3 seed the finite constructive compact-interval route over closed-bounded, located, finite-net, dyadic, stream, regular, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompactInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient endpoint equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source compatibility, finite-net admission, dyadic-window exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompactInterval

Chapter 3987

A Concrete Naming Certificate for $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Kelley on uniform and metric extension, the $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ packet records the completion reflector as a left-adjoint NameCert boundary. It sits after the metric completion carrier of chapter 1506, the dense embedding route of chapter 3351, the Cauchy completion adjunction surface of chapter 1456, the metric adjunction packet of chapter 3836, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the reflector boundary without quotient representatives, host equality of completions, countable choice, selected limits, or ambient completeness.

Definition 3987.1 (MetricCompletionLeftAdjoint carrier)**.** *A* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, D, A, J, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source reflector row.*
2. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit row exposing the dense source inclusion.*
3. *A is the* $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$ *adjunction row for completion-facing universal reads.*
4. *J is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *metric adjunction row binding the same source, unit, counit, and triangle ledger.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row opened by completion consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, D, A, J, R have been displayed.*
7. *H records componentwise* hsame *transport for reflector, dense embedding, adjunction, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric-source request through the reflector, dense unit, adjunction handoff, regular readback, and Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbeddi}$ $\mathsf{CauchyCompletionAdjunction}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these ten displayed rows. It has no coordinate for quotient completion classes, chosen representatives, host equality of completion objects, countable-choice schedules, selected limits, or ambient complete metric spaces.

Lean4 ✓ `BEDC.Derived.MetricCompletionLeftAdjointUp.MetricCompletionLeftAdjointTasteGate single carrier alignment`

Theorem 3987.2 (MetricCompletionLeftAdjoint NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionLeft}$ *carrier* $L = (M, D, A, J, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$ *surface consists exactly of six finite obligations:*

1. *carrier admission for the displayed reflector, dense embedding, adjunction, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *unit exposure, requiring every source-to-completion read to pass through* D *before the adjunction rows are consumed;*
3. *left-adjoint handoff, requiring* A *and* J *to name the same finite completion reflector route before any universal consumer reads it;*
4. *triangle and extension visibility, requiring the unit, counit, and triangle comparison inherited from the adjunction rows to remain displayed through* H *and* C*;*
5. *RegSeqRat/Real seal nonescape, requiring* R *and* E *to be reached only after the reflector and adjunction rows have been displayed;*
6. *exhaustion of transport, replay, provenance, and local naming through* H, C, P, N*.*

Proof. Carrier admission is projection from Definition 3987.1. The tuple displays the full finite row inventory made available to the local certificate.

The source boundary is read through M and D. The reflector row supplies the completion-facing carrier, while the dense embedding row is the displayed unit of the reflector.

The universal boundary is then read through A and J. The Cauchy adjunction and metric adjunction rows provide the same finite source, unit, counit, and triangle ledger, so the left-adjoint handoff is a NameCert route rather than a host categorical equality.

Finally R, E, H, C, P, N seal and replay the route for downstream consumers. Since quotient classes, selected representatives, choice schedules, host equality, and ambient completeness are absent from the carrier, the listed obligations exhaust the certificate surface. □

Theorem 3987.3 (MetricCompletionLeftAdjoint universal handoff)**.** *Every separated-completion, Real-facing, or universal-completion consumer of an accepted* $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionDenseEmbedding}^{\uparrow} \longmapsto \mathsf{CauchyCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{MetricCompletion}$$

The consumer receives source metric admission, dense unit exposure, adjunction handoff, triangle visibility, regular readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming. It receives no quotient completion, selected representative, host equality of completions, countable choice, selected limit, or ambient completeness principle.

Proof. Project the accepted carrier $L = (M, D, A, J, R, E, H, C, P, N)$. A universal read opens M first and then the dense unit row D, so the source-to-completion boundary is displayed before the adjunction rows are available.

Next consume A and J. These rows expose the Cauchy and metric adjunction handoffs over the same finite reflector route, including the unit, counit, and triangle comparison needed by universal consumers.

The read then passes through R and E. Regular rational readback and the terminal Real seal are downstream from the displayed reflector and adjunction rows, not independent completeness assumptions.

The structural rows H, C, P, N preserve the ordered handoff under transport, replay, provenance, and local naming. Row exhaustion excludes quotient representatives, selected limits, host equality, countable choice, and ambient completeness from the exported certificate. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ universal read factors through MetricCompletion, MetricCompletionDenseEmbedding, CauchyCompletionAdjunction, MetricCompletionAdjunction, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Independence witness: metriccompletion, metric-completion-dense-embedding, cauchy-completion-adjunction, metric-completion-adjunction, regseqrat, real: $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly packages reflector admission, dense unit exposure, adjunction handoff, triangle visibility, regular readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}$ packet is generated from a MetricCompletion reflector row, a MetricCompletionDenseEmbedding unit row, CauchyCompletionAdjunction and MetricCompletionAdjunction handoff rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3987.1, Definition 3987.2, and Definition 3987.3 seed the finite left-adjoint completion reflector route over metric completion, dense embedding, adjunction, regular readback, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionLeftAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completions, selected representatives, host equality of completions, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit exposure, left-adjoint handoff, triangle visibility, RegSeqRat/Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionLeftAdjointUp.

Chapter 3988

A Concrete Naming Certificate for $\mathsf{CFun}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CFun}^{\uparrow}$ packet records an ordinary continuous real-facing map as finite source, target, modulus, transport, replay, provenance, and local naming rows. It is an L10 neighbour of the terminal $\mathsf{Real}^{\uparrow}$ seal in chapter 38, the finite windows of chapter 126, the regular rational readback of chapter 301, the formal-ball radius interface of chapter 1510, and the carrier-only packet of chapter 3611. The chapter names the finite continuity certificate consumed by $\mathsf{ContinuousMap}^{\uparrow}$, compact-uniform-continuity, and modulus packets without treating host functions as primitive objects.

Definition 3988.1 (Continuous function carrier)**.** *A* $\mathsf{CFun}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, M, Q, W, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-window row, exposing the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and source-domain histories inspected by the continuity request.*
2. *T is the target-readback row, exposing the target* $\mathsf{Real}^{\uparrow}$ *or sibling endpoint readback reached by the displayed source window.*
3. *M is the finite modulus row, translating each target tolerance request into the source-window budget used by the packet.*
4. *Q is the continuity-certificate row binding S, T, M to the displayed request.*
5. *W is the shared finite window ledger over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and any* $\mathsf{FormalBall}^{\uparrow}$ *radius window opened by the same request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row used when a continuous-function consumer asks for a real-valued endpoint.*
7. *H records componentwise* hsame *transport for source, target, modulus, certificate, window, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a source-window request through S, T, M, Q, W, R.*
9. *P is the* Pkg *provenance over* $\mathsf{CFun}^{\uparrow}$, $\mathsf{CFunCarrier}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, T, M, Q, W, R, H, C, P, N$. It has no coordinate for host function extensionality, countable choice, quotient equality of maps or reals, selected modulus functions, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.ContinuousFunctionUp.ContinuousFunctionTasteGate single carrier alignment`

Theorem 3988.2 (Continuous function NameCert obligations)**.** *For every accepted* $\mathsf{CFun}^\uparrow$ *carrier $F = (S, T, M, Q, W, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CFun}^\uparrow}$ *surface consists exactly of source-window admission through S, target readback through T, finite modulus exposure through M, continuity-certificate binding through Q, shared* $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, *and* $\mathsf{FormalBall}^\uparrow$ *window routing through W, real-facing readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 3988.1. The rows $S, T, M, Q, W, R, H, C, P, N$ are the only local coordinates, so source admission, target readback, modulus exposure, continuity binding, window routing, readback, transport, replay, provenance, and naming must all be read there.

The analytic route is finite. The modulus row M translates a target tolerance into a source-window budget, Q binds that budget to the displayed source and target rows, and W, R expose the StreamName, RegSeqRat, FormalBall, and Real-facing readbacks attached to the same request.

Finally H, C, P, N preserve and name the same route. Since the carrier has no host-function, function-extensionality, quotient-equality, choice, selected-modulus, or ambient-completeness coordinate, the listed rows exhaust the NameCert obligation surface. □

Theorem 3988.3 (Continuous function Real-map handoff)**.** *Every* $\mathsf{ContinuousMap}^\uparrow$, *compact-uniform-continuity, or real-valued modulus consumer of an accepted* $\mathsf{CFun}^\uparrow$ *packet receives the map only through*

$$S, M, Q \longmapsto W \longmapsto T, R \longmapsto H, C, P, N.$$

Thus the consumer may use the source window, finite modulus, continuity certificate, shared $\mathsf{StreamName}^\uparrow$ *and* $\mathsf{RegSeqRat}^\uparrow$ *windows, optional* $\mathsf{FormalBall}^\uparrow$ *radius interface, target readback, real seal, componentwise transport, displayed replay, provenance, and local naming rows. It receives no host function object, function extensionality principle, countable-choice selector, quotient equality of maps or reals, selected limit, or ambient* $\mathsf{Real}^\uparrow$ *completeness witness.*

Proof. Begin with the source and modulus rows. A consumer request opens S and M before any target readback is available, and Q is the only row binding the source budget to the target tolerance.

Next read the finite windows. The row W exposes the relevant $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, and $\mathsf{FormalBall}^\uparrow$ faces, while T and R expose the target and real-facing readbacks reached by that same finite route.

The rows H, C, P, N then transport, replay, source, and name the handoff. Row exhaustion in Definition 3988.1 excludes every host-function, function-extensionality, quotient-equality, choice, selected-limit, and ambient-completeness escape route. □

Falsifiable prediction: Every accepted $\mathsf{CFun}^\uparrow$ read must expose source window, target readback, finite modulus, continuity certificate, StreamName and RegSeqRat windows, optional FormalBall radius routing, Real-facing readback, componentwise transport, displayed replay, provenance, and local naming rows before a continuous-map or modulus consumer can inspect it.

Independence witness: real, streamname, regseqrat, formalball, continuousfunctioncarrier, continuousmap: $\mathsf{CFun}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source-window, target-readback, finite-modulus, continuity-certificate, shared-window, real-readback, transport, replay, provenance, and local naming rows without quotienting maps or importing host functions.

Closure status for $\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CFun}^{\uparrow}$ packet is generated from source-window rows, target readback rows, finite modulus rows, continuity-certificate rows, shared StreamName, RegSeqRat, and FormalBall windows, real-facing readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$, seed)

Scope closed: Definition 3988.1, Definition 3988.2, and Definition 3988.3 seed the finite non-host continuous-function route over source windows, target readbacks, modulus data, continuity certificates, window ledgers, readbacks, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CFun}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, countable choice, quotient map equality, quotient real equality, selected modulus functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, source-window access, target readback, finite modulus exposure, continuity-certificate binding, shared window routing, real-facing handoff, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousFunctionUp.

Chapter 3989

A Concrete Naming Certificate for $\mathsf{CounterfactualOutcome}^{\uparrow}$

Following Pearl, Causality, 2009, the $\mathsf{CounterfactualOutcome}^{\uparrow}$ packet records paired factual and counterfactual outcome reads as finite causal-inference NameCert data. It sits beside the intervention calculus of chapter 453, the intervention-row packet of chapter 1290, the probability-space carrier of chapter 184, the distribution handoff of chapter 186, and the Bayesian conditioning route of chapter 232. The packet gives causal consumers a selected counterfactual-outcome surface without weakening the explicit non-claim boundary of $\mathsf{DoCalculus}^{\uparrow}$.

Definition 3989.1 (Counterfactual outcome carrier)**.** *A* $\mathsf{CounterfactualOutcome}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N)$$

with the following displayed rows:

1. *U is the finite unit row, naming the observational unit and the displayed covariate cells opened by the counterfactual query.*
2. *I is the intervention row inherited from the* $\mathsf{CausalIntervention}^{\uparrow}$ *route. It records the do-row, intervention target, and intervention value that may be inspected by the counterfactual read.*
3. *Y_f is the factual outcome row observed for U before the counterfactual branch is exported.*
4. *Y_{cf} is the counterfactual outcome row selected under I for the same unit row U.*
5. *M is the modified-distribution row, read through the* $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, *and* $\mathsf{Bayesian}^{\uparrow}$ *siblings before any expectation or probability export.*
6. *R is the consistency row requiring factual and counterfactual reads to agree on the factual branch selected by the displayed intervention route.*
7. *H records componentwise* hsame *transport for units, interventions, factual outcomes, counterfactual outcomes, modified distributions, consistency rows, distribution exports, expectation rows, replay rows, provenance rows, and local naming rows.*
8. *D is the distribution handoff row. It exposes only the finite probability surface displayed by M and does not store hidden-world measures.*
9. *E is the expectation and export row for finite counterfactual queries.*

10. *T records displayed* Cont *replay of the route*

$$U, I, Y_f, Y_{cf}, M, R \longmapsto D, E.$$

11. *P is the* Pkg *provenance over* $\mathsf{CounterfactualOutcome}^{\uparrow}$, $\mathsf{DoCalculus}^{\uparrow}$, $\mathsf{CausalIntervention}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Bayesian}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N$. It has no coordinate for hidden structural equations, cross-world equality outside the displayed consistency row, selected counterfactual universes, host structural causal models, or ambient probability semantics.

Lean4 ✓ `BEDC.Derived.CounterfactualOutcomeUp.CounterfactualOutcomeTasteGate single carrier alignment`

Theorem 3989.2 (Counterfactual outcome NameCert obligations)**.** *For every accepted* $\mathsf{CounterfactualOutcome}^{\uparrow}$ *carrier*

$$C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the finite unit row, intervention row, factual outcome row, counterfactual outcome row, modified-distribution row, consistency row, transport, distribution handoff, expectation and export row, replay, provenance, and local naming row;*

2. *intervention anchoring, requiring every counterfactual read Y_{cf} to cite the displayed I row and the same unit row U before any export through D or E;*

3. *factual and counterfactual consistency, requiring R to bind the factual branch of Y_{cf} to the observed Y_f without adding cross-world equality outside the carrier;*

4. *probability handoff, requiring M, D, E to pass through the displayed* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, *and* $\mathsf{Bayesian}^{\uparrow}$ *routes before any finite probability or expectation read is available;*

5. *non-escape from hidden worlds, host structural causal models, and selected counterfactual universes, requiring H, T, P, N to preserve only the displayed rows and to export no unlisted causal semantics.*

No counterfactual-outcome obligation is stored outside these rows.

Proof. Project the accepted carrier $C = (U, I, Y_f, Y_{cf}, M, R, H, D, E, T, P, N)$. Carrier admission exposes exactly the finite unit, intervention, factual outcome, counterfactual outcome, modified-distribution, consistency, transport, distribution, export, replay, provenance, and naming rows.

The counterfactual branch is anchored before it is exported. The row I supplies the intervention target and value, while U supplies the unit row shared by Y_f and Y_{cf}.

Consistency is then read through R. This row identifies the factual branch of the counterfactual read with the observed factual outcome, but it does not create an unconstrained cross-world equality or a hidden structural equation.

The probability-facing rows are downstream. The modified distribution M, distribution handoff D, and expectation row E are available only along the displayed probability, distribution, and Bayesian routes.

Finally H, T, P, N transport, replay, source, and name the same finite surface. Since no carrier coordinate stores hidden worlds, a host structural causal model, or a selected counterfactual universe, none can escape through the local NameCert. □

Falsifiable prediction: Every $\mathsf{CounterfactualOutcome}^{\uparrow}$ counterfactual-facing read factors through a finite unit row, a displayed intervention row, factual and counterfactual outcome rows, a modified-distribution row, a consistency row, componentwise hsame transport, distribution and expectation handoff rows, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: do-calculus, causal-intervention, probspace, distribution, bayesian: CounterfactualOutcor is not bijective to any listed sibling because it jointly carries paired factual and counterfactual outcomes for a unit, a displayed intervention anchor, a modified-distribution handoff, a consistency row, finite probability and expectation export, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3989.1 and Definition 3989.2 seed the finite counterfactual-outcome route over unit, intervention, factual outcome, counterfactual outcome, modified-distribution, consistency, distribution, expectation, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CounterfactualOutcome}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hidden structural equations, host structural causal models, selected counterfactual universes, cross-world equality outside the displayed consistency row, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, intervention anchoring, factual and counterfactual consistency, probability handoff, non-escape from hidden worlds, transport, replay, provenance, and local naming for BEDC.Derived.CounterfactualOutcomeUp.

Constructive story: A $\mathsf{CounterfactualOutcome}^{\uparrow}$ packet is generated from a finite unit row, a displayed intervention row, factual and counterfactual outcome rows, a modified-distribution row, a consistency row, componentwise hsame transport, distribution and expectation handoff rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 3990

A Concrete Naming Certificate for $\mathsf{FiniteSupportRemoval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteSupportRemoval}^{\uparrow}$ packet records the constructive finite-cover operation that deletes one displayed support row while keeping the remaining finite cover, tolerance ledger, and terminal real-name route visible. It consumes the positive-radius surface of chapter 452, the compact finite-net route of chapter 3957, and the L10 dependency chain through chapter 126, chapter 301, and chapter 38. The packet names a support-pruning certificate for finite compactness consumers; it is not a selected subcover functional, a host-level minimum operation, a quotient equality principle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3990.1 (Finite support removal carrier)**.** *A* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, G, D, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net or finite-cover source row, exposing the finite list of support names and coverage witnesses.*
2. *G is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius row that keeps every retained support row inside the displayed cover tolerance.*
3. *D is the deleted support name, together with its membership witness in the original finite support spine.*
4. *W is the retained finite window after removing D, with its inherited order and membership rows.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger transported from the original cover to the retained window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over the retained window.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only through the retained* $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *route.*
8. *H records componentwise* hsame *transport for K, G, D, W, T, R, L and for the structural rows below.*
9. *C records displayed* Cont *replay from finite-cover inspection through support deletion, tolerance transport, regular readback, and Real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteSupportRemoval}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactUniformFiniteNetRealizati}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, D, W, T, R, L, H, C, P, N$. It has no coordinate for an undisplayed cover member, a selected replacement support, quotient equality of real names, countable choice, or an ambient compactness theorem.

Theorem 3990.2 (Finite support removal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *carrier $S = (K, G, D, W, T, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite-cover source, positive-radius ledger, deleted support name, retained window, tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows; membership exposure for D in the original support spine; retained-window coverage through W and G; L10 readback through T, R, L; and non-escape from selected replacement supports, quotient real equality, countable choice, and ambient compactness.*

Proof. Project the carrier of Definition 3990.1. The rows $K, G, D, W, T, R, L, H, C, P, N$ are the complete displayed inventory.

The deletion row is finite data. Since D is stored with its membership witness in the original finite support spine and W stores the retained window, the packet can inspect the removed coordinate without choosing a replacement support or leaving the finite cover.

The coverage and tolerance rows are inherited only through displayed coordinates. The positive-radius row G keeps the retained supports inside the finite Lebesgue-radius ledger, while T transports the dyadic tolerance data to the retained StreamName window.

Finally the regular readback and Real seal are read through R and L, then preserved by H, C, P, N. The refused objects are absent from the carrier, so the local NameCert surface is exactly the obligation list stated above. □

Lean4 ✓ `BEDC.Derived.FiniteSupportRemovalUp.FiniteSupportRemovalTasteGate single carrier alignment`

Theorem 3990.3 (Finite support removal compact-cover handoff)**.** *Every compactness or uniform-modulus consumer of an accepted* $\mathsf{FiniteSupportRemoval}^{\uparrow}$ *packet reads the deletion route through the finite chain*

$$\mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{CompactUniformFiniteNetRealization}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer first displays the original finite support spine, then the removed coordinate and retained window, then the positive-radius and dyadic tolerance rows, and only then the regular readback and Real seal. No consumer may replace the deleted support by an undisplayed cover member, a selected subcover, quotient real equality, countable choice, or ambient completeness.

Proof. Use Definition 3990.2. It orders the finite-cover row, the deleted support row, the retained window, the positive-radius ledger, the tolerance ledger, the regular readback, and the terminal Real seal inside one accepted packet.

The finite cover is consumed before deletion. Thus the support spine and the deleted coordinate are both visible BEDC histories, and the retained window W is a finite subspine rather than a host-level set difference.

The L10 dependency route is then forced. The retained StreamName window carries the dyadic tolerance row T, the regular rational readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal L through the same displayed route used by finite-cover compactness chapters.

Transport, replay, provenance, and local naming preserve this route without adding another support coordinate. Therefore the handoff supplies a constructive deletion primitive for compact finite-cover consumers while refusing the listed external objects. □

Falsifiable prediction: An accepted $\mathsf{FiniteSupportRemoval}^{\uparrow}$ read is refuted if a compactness or uniform-modulus consumer reaches a retained Real seal without displaying the original finite support spine, the deleted support name, the retained window, the positive-radius ledger, the tolerance row, the regular readback row, transport, replay, provenance, and local naming.

Independence witness: finitelebesguenumber, compactuniformfinitenetrealization, streamname, regseqrat, and real: none is bijective to the support-removal packet because only $\mathsf{FiniteSupportRemoval}^{\uparrow}$ binds a deleted finite support coordinate to a retained-window L10 readback route before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSupportRemoval}^{\uparrow}$ packet is generated from finite-cover source rows, FiniteLebesgueNumberUp positive-radius rows, a displayed deleted support name, retained StreamName window rows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3990.1, Definition 3990.2, and Definition 3990.3 seed the finite support-deletion surface for compactness and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSupportRemoval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected subcover extraction, quotient real equality, countable choice, host set difference, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, support membership exposure, retained-window coverage, positive-radius preservation, tolerance transport, regular readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSupportRemovalUp.

The $\mathsf{Cauchy}^{\uparrow}$ hub reads finite windows through chapter 126, dyadic tolerance through chapter 291, regular readback through chapter 301, and the terminal seal through chapter 38.

Chapter 3991

A Concrete Naming Certificate for CauchyRemainderEstimate$^{\uparrow}$

Following Bishop and Bridges constructive analysis of regular Cauchy names, the CauchyRemainderEstimate$^{\uparrow}$ packet records finite tail estimates before a completion or real-name consumer is allowed to read a candidate limit surface. It sits on the Phase Real and Completion route after the DyadicRatCore$^{\uparrow}$ tolerance ledger of chapter 291, the finite observation handles of chapter 126, the regular rational readback of chapter 301, the tail-diameter budget of chapter 1746, and the real-name seal of chapter 38. The packet exposes a finite Cauchy tail remainder estimate; it does not select a limit, identify quotient streams, invoke countable choice, import ambient Real completeness, or turn a boundary counterexample into a convergent name.

Definition 3991.1 (Cauchy remainder estimate carrier)**.** *A* CauchyRemainderEstimate$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, D, T, W, R, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* StreamName$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *row whose observations are being tested on a finite tail.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, recording the requested error bound, its positive slack, and the finite comparison budget used by the estimate.*
3. *T is the displayed threshold row, naming the finite tail index after which the estimate is allowed to read observations.*
4. *W is the* StreamName$^{\uparrow}$ *window row opened only at indices admitted by T.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational observations exposed by W and checked against the tolerance data in D.*
6. *B is the* CauchyTailDiameter$^{\uparrow}$ *boundary row, recording the displayed tail-diameter budget through which the same finite evidence is consumed by a* Real$^{\uparrow}$ *seal route or by a refusal route.*
7. *H records componentwise* hsame *transport for the source, tolerance, threshold, window, readback, boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through D, T, W, R, B.*
9. *P is the* Pkg *provenance over* CauchyRemainderEstimate$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* CauchyTailDiameter$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRemainderEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, T, W, R, B, H, C, P, N$. It has no coordinate for a selected limit, quotient stream equality, countable choice, ambient completeness, host real equality, trichotomy, or bridge checking.

Lean4 ✓ `BEDC.Derived.CauchyRemainderEstimateUp.CauchyRemainderEstimateTasteGate single carrier alignment`

Theorem 3991.2 (Cauchy remainder estimate NameCert obligations)**.** *For every accepted* CauchyRemainderEstima *carrier*

$$E = (S, D, T, W, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRemainderEstimate}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source, dyadic tolerance ledger, finite threshold, admitted windows, rational readback, boundary row, transport, replay, provenance, and local naming rows;*
2. *tolerance exposure, requiring each remainder request to read its bound and slack data from D before any tail observation is opened;*
3. *threshold exactness, requiring T to be the only finite index gate through which W may expose tail observations;*
4. *readback alignment, requiring R to compare only the observations admitted by W against the same tolerance request from D;*
5. *boundary nonescape, requiring B to route the estimate through the displayed* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *budget before any* $\mathsf{Real}^{\uparrow}$ *seal consumer may read it, without adding a selected limit or quotient stream;*
6. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N;*
7. *refusal of selected limits, quotient stream equality, countable choice, ambient completeness, host real equality, trichotomy, and bridge checking as local certificate data.*

No Cauchy-remainder-estimate obligation is stored outside these rows.

Proof. Project the carrier of Definition 3991.1. The accepted packet exposes exactly the source row S, dyadic tolerance ledger D, threshold row T, window row W, readback row R, boundary row B, and structural rows H, C, P, N.

A remainder request first reads D. This fixes the tolerance, the positive slack, and the finite comparison budget before any source observation is visible.

The request then reads T, W, R in order. The threshold row supplies the finite tail gate, the window row exposes only observations admitted by that gate, and the readback row compares those observations against the same dyadic tolerance data.

Finally, B determines whether the estimate is consumed by a Real seal route or a Specker refusal route, while H, C, P, N transport, replay, source, and name the same finite data. Since the excluded principles do not occur as carrier coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 3991.3 (Cauchy remainder estimate dyadic tail-bound admission)**.** *Every accepted* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ *carrier presents finite tail evidence through the route*

$$S \longmapsto D \longmapsto T \longmapsto W \longmapsto R \longmapsto B.$$

Thus a consumer can inspect a Cauchy remainder only after the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request, finite threshold, admitted* $\mathsf{StreamName}^{\uparrow}$ *tail windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row have all been displayed.*

Proof. Start with an accepted consumer read of E. Carrier projection opens the source row S and the tolerance ledger D before any tail observation is available.

The finite tail gate is then supplied by T. The window row W is downstream from that gate, so it cannot expose observations outside the admitted finite tail schedule.

The readback row R compares only the windows selected by W with the tolerance data fixed by D. The boundary row B records the $\mathsf{CauchyTailDiameter}^{\uparrow}$ budget for the downstream consumer without changing the admitted observations. Hence every tail read factors through the displayed finite route. □

Theorem 3991.4 (Cauchy remainder estimate regular-Cauchy readback)**.** *If an accepted* CauchyRemainderEstimate *carrier is read as regular Cauchy data, then the readback exports only the displayed source row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, finite threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *observations,* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

It exports no selected limit, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, or hidden modulus selector.

Proof. Assume a regular Cauchy consumer reads the packet. The carrier still supplies the consumer only through $S, D, T, W, R, B, H, C, P, N$.

The regular-facing route therefore receives a finite remainder estimate, not a limit-selection mechanism. The tolerance ledger and threshold row fix the request, the window and readback rows expose the rational observations, and the $\mathsf{CauchyTailDiameter}^{\uparrow}$ boundary row records the finite diameter budget used by the read.

The structural rows preserve and name that route. Since no selected-limit, quotient, choice, completeness, host-equality, trichotomy, or hidden-selector coordinate is present, the readback cannot export any of those objects. □

Theorem 3991.5 (Cauchy remainder estimate Real seal nonescape)**.** *If an accepted* $\mathsf{CauchyRemainderEstimate}^{\uparrow}$ *carrier is consumed by a* $\mathsf{Real}^{\uparrow}$ *route, then the seal is downstream of the same finite route*

$$S, D, T, W, R, B, H, C, P, N.$$

The $\mathsf{Real}^{\uparrow}$ *consumer cannot bypass the displayed dyadic tolerance, stream windows, regular rational readback, or tail-diameter budget by invoking selected limits, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, or bridge checking.*

Proof. Assume the boundary row reaches a Real seal consumer. The carrier projection supplies no Real-facing coordinate except the row B together with the displayed source, tolerance, threshold, window, readback, transport, replay, provenance, and local naming rows.

Apply the dyadic tail-bound admission route. It forces the Real-facing read to pass through D, T, W, R before the boundary row can be consumed.

Apply the regular-Cauchy readback theorem. The readback row exports only the regular rational observations admitted by the same finite tolerance and window data. The Real seal therefore receives a finite remainder estimate rather than an independent limit object. □

Falsifiable prediction: Every accepted CauchyRemainderEstimate$^{\uparrow}$ consumer read factors through a source row, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a finite threshold, StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, a CauchyTailDiameter$^{\uparrow}$ boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-tail-diameter, real: CauchyRemainderEstim is not bijective to any listed sibling because it jointly binds a source stream, dyadic tolerance budget, finite tail threshold, admitted observation windows, rational readback, tail-diameter boundary, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$.

Constructive story: A CauchyRemainderEstimate$^{\uparrow}$ packet is generated from a displayed source stream or regular rational name, a DyadicRatCore tolerance ledger, a finite threshold, StreamName tail windows, RegSeqRat readback, a CauchyTailDiameter boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$, seed)

Scope closed: Definition 3991.1, Definition 3991.2, Definition 3991.3, Definition 3991.4, and Definition 3991.5 seed the finite tail-estimate route for Real and Completion consumers without selected limits or quotient stream equality.

Formal verification: FormalStatus(NameCert$_{\text{CauchyRemainderEstimate}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyRemainderEstimateUp.CauchyRemainderEstimateTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable choice, ambient Real completeness, host real equality, trichotomy, Specker convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exposure, threshold exactness, readback alignment, boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRemainderEstimateUp.

Theorem 3991.6 (Cauchy carrier root obligations)**.** *Let*

$$R = (S, D, Q, T, E, H, C, P, N)$$

be an accepted root Cauchy$^{\uparrow}$ *remainder carrier. Here S is the finite* StreamName$^{\uparrow}$ *window row, D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, Q is the* RegSeqRat$^{\uparrow}$ *readback row, T is the* CauchyTailDiameter$^{\uparrow}$ *tail-diameter row, E is the terminal* Real$^{\uparrow}$ *seal row, and H, C, P, N are the componentwise transport, displayed replay, provenance, and local naming rows. The local root obligation surface is exactly carrier admission for these rows, tolerance exactness through D, readback exposure through Q, tail-diameter exposure through T, terminal sealing through E, and structural stability through H, C, P, N.*

No accepted root carrier contains a selected limit, quotient stream, countable-choice schedule, host real equality, or ambient Real$^{\uparrow}$ *completeness coordinate.*

Proof. Project the displayed BHist carrier R. The stream row S is the only source of visible observations, and the dyadic ledger D is the only source of precision requests and comparison budgets.

The regular readback row Q is downstream of S and D, while the tail-diameter row T is downstream of the same finite windows and tolerances. Thus the root carrier records a finite remainder surface rather than a selected completed object.

The terminal seal row E may be read only after S, D, Q, T have been displayed. The rows H, C, P, N transport, replay, source, and name that same finite route. Since there are no further coordinates, the listed rows exhaust the root obligation surface and the excluded principles cannot be exported. □

Theorem 3991.7 (Cauchy tail-envelope root obligations)**.** *For every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *remainder carrier* $R = (S, D, Q, T, E, H, C, P, N)$*, every tail-envelope read factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyTailDiameter}^{\uparrow}.$$

The tail-diameter row T *may compare only the finite windows exposed by* S*, the dyadic budgets exposed by* D*, and the regular rational readbacks exposed by* Q*. It cannot introduce quotient equality of streams, a selected limit, countable choice, host real equality, or an ambient completeness theorem.*

Proof. Start with a tail-envelope request. Carrier projection exposes the finite StreamName row S before any tolerance, readback, or diameter claim is available.

The dyadic ledger D supplies the displayed tolerance budget. The RegSeqRat row Q reads the same finite windows under that budget, so a tail comparison cannot cite an independent stream representative or an external choice schedule.

The CauchyTailDiameter row T is then read as a finite envelope over S, D, Q. Transport and replay through H, C preserve those rows componentwise, while P, N record provenance and local naming. Hence the tail-envelope route remains inside the displayed finite packet. □

Theorem 3991.8 (Cauchy Real-seal root obligations)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted root* $\mathsf{Cauchy}^{\uparrow}$ *remainder carrier must read the terminal seal row* E *only after the finite path*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyTailDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

has been displayed. The seal is a terminal BEDC row for the shown remainder carrier; it is not a selected limit, quotient-stream representative, countable-choice output, host real equality, or ambient real-completeness witness.

Proof. Project $R = (S, D, Q, T, E, H, C, P, N)$. The rows S, D, Q, T are the only analytic inputs to the terminal seal: finite stream windows, dyadic tolerances, regular rational readback, and tail-diameter evidence.

The Real seal E is downstream of that path. A consumer may read it only after the displayed Cauchy remainder route is visible, so the seal cannot be used to manufacture a quotient stream or a selected completed limit.

The structural rows H, C, P, N attach to the same carrier. They preserve and replay the finite path and do not add host real equality, countable choice, or ambient completeness. Therefore every Real-facing root read is exhausted by the stated path. □

Definition 3991.9 (Cauchy seed carrier)**.** *A* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier is a finite* BHist *packet*

$$S = (W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observation windows that a Cauchy consumer may inspect.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the displayed dyadic precision requests and comparison budgets for those windows.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, obtained only from the displayed stream windows and dyadic tolerance data.*

4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed after the regular rational readback, not a selected limit or host real equality witness.*
5. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an admitted Cauchy request through W, D, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row.*

The classifier compares accepted seed packets only through W, D, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 3991.10 (Cauchy seed NameCert obligations)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier $S = (W, D, R, E, H, C, P, N)$, the local seed-level* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, and local naming;*
2. *tolerance exactness, requiring every Cauchy comparison to read its dyadic budget from D before any regular rational readback is accepted;*
3. *window readback exactness, requiring R to be generated from the displayed windows W and tolerances D;*
4. *Real seal discipline, requiring E to remain downstream of W, D, R and never to serve as a selected limit or host equality object;*
5. *structural stability under* hsame *transport H and displayed* Cont *replay C;*
6. *provenance and local naming through P and N.*

No seed obligation certifies quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 3991.9. The accepted packet exposes only the finite window row, dyadic tolerance row, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Tolerance exactness is read from D after the window row W has been displayed. The regular readback row R is downstream of both rows, so it cannot introduce an independent rational sequence or a choice schedule.

The Real seal row E is terminal for the seed route. It is consumed only after W, D, R are visible, and therefore cannot act as a selected limit, quotient equality, or host real equality witness.

Finally H and C preserve the same component rows under transport and replay, while P and N source and name the packet. Since no remaining coordinate exists, the listed rows exhaust the seed-level NameCert surface. □

Theorem 3991.11 (Cauchy seed Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing read of an accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier factors through*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming preserved by* H, C, P, N*. The seed route exports finite stream windows, dyadic tolerances, regular rational readback, and a terminal Real seal, but exports no quotient stream equality, selected limit, host real equality, trichotomy, countable-choice selector, or ambient completeness principle.*

Proof. Begin with a Real-facing or completion-facing read from $S = (W, D, R, E, H, C, P, N)$. Carrier admission exposes the finite StreamName window row W before the tolerance and readback rows can be consumed.

The dyadic tolerance row D supplies the displayed precision budget for the read. The regular rational readback row R then packages the windowed data under that budget, rather than selecting an external limit or invoking a host sequence.

The Real seal row E is reached only after W, D, R have been displayed. Structural transport and replay through H, C keep the same route, and P, N record provenance and local naming.

Because the carrier has no coordinate for quotient stream equality, selected limits, host real equality, trichotomy, countable choice, or ambient completeness, every exported Real-facing read remains inside the seed route stated above. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedRealSealBoundary`

Theorem 3991.12 (Cauchy tail request carrier)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *root tail-request surface is the finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

whose rows are finite $\mathsf{StreamName}^{\uparrow}$ *windows, a directed later-window request,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *naming. It is the same finite request surface isolated by Definition 5821.1 and consumed by chapter 5817.*

Proof. Project the Cauchy seed carrier of Definition 3991.9. The rows W, D, R, E, H, C, P, N already expose finite windows, dyadic tolerance, regular readback, Real sealing, transport, replay, provenance, and local naming.

Insert the directed request row Q between the window row and the tolerance row. The resulting tuple is exactly the $\mathsf{CauchyTailRequest}^{\uparrow}$ carrier of Definition 5821.1, with the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ sibling anchors.

Because Definition 5817.1 reads its tail coordinate as a finite directed request row, the root Cauchy packet supplies the carrier surface required by the finite filter route without adding quotient filters, selected limits, host equality, choice, or Real completeness. □

Theorem 3991.13 (Cauchy directed-tail NameCert obligations)**.** *For the root tail-request packet* $T = (W, Q, D, R, E, H, C, P, N)$ *of Definition 3991.12, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *directed-tail obligation surface consists exactly of carrier admission, request classifier visibility, stability under later-window refinement, tolerance exactness, readback exactness, ledger routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations are read through the sibling rows of chapter 126, chapter 291, chapter 301, chapter 38, and chapter 5817.*

Proof. Carrier admission is the tuple display in Definition 3991.12. The request classifier is the row Q, and it is visible only after the finite window row W has named the inspected StreamName windows.

Stability and tolerance exactness are read from Q and D together. A later-window refinement changes only the displayed request route and must keep the dyadic budget attached to that route before any regular rational readback is accepted.

Readback exactness and ledger routing place R downstream of W, Q, D and place E downstream of R. Thus the Real-facing seal cannot serve as a selected limit, quotient-stream comparison, host equality witness, or ambient completeness principle.

The structural rows H, C, P, N supply transport, replay, provenance, and local naming. Projecting Definition 5821.2 gives the same row list for the named request surface, while Definition 5817.2 identifies the finite-filter consumer that reads that request. □

Theorem 3991.14 (Cauchy filter Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing Cauchy consumer that reads a filter, completion, or regular-name seal factors through the finite tail-request packet of Definition 3991.12 before it can read* E*. The route is*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

with hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows preserved throughout.*

Proof. Start with a Real-facing Cauchy read. The seed boundary of Definition 3991.11 already forces the read through finite windows, dyadic tolerance, regular readback, and the terminal Real seal.

Refine the seed route by Definition 3991.12. The finite request row Q is inserted before the tolerance ledger, so no downstream consumer reaches D, R, or E without first reading the directed-tail request surface.

Apply Definition 3991.13. Its ledger and exactness clauses keep W, Q, D, R, E in order, and its structural clauses preserve the same route through H, C, P, N.

Finally compare with Definition 5817.3. The filter consumer reads the same finite route before consuming the Real seal, so every Real-facing Cauchy answer factors through the displayed tail request rather than through quotient filters, selected limits, countable choice, host equality, or Real completeness. □

Theorem 3991.15 (Cauchy seed closure route)**.** *Every public route through the* $\mathsf{Cauchy}^{\uparrow}$ *hub is read through the sibling surfaces of chapter 3413, chapter 126, chapter 301, chapter 291, chapter 3991, and chapter 38 before any Real-facing completion consumer can use it. The route exposes finite stream windows, dyadic tolerance rows, regular rational readback, Cauchy remainder estimates, terminal real seals, componentwise transport, replay, provenance, and local naming; it exports no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Cauchy carrier of Definition 3413.1. Its public readback is already ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal-facing row.

Apply Definition 3413.3 and Definition 3413.4. These results force every regular rational read to pass through finite stream windows, dyadic tolerances, the displayed modulus row, and the compatibility row before R is inspected.

Apply Definition 3991.3 for completion-facing tolerance reads. The remainder estimate route inserts the finite threshold and admitted tail windows before any Real or completion consumer can treat the Cauchy packet as usable tail evidence.

Finally apply Definition 3413.9. The terminal Real-facing seal remains downstream of the displayed regular rational and remainder-estimate routes. Since none of the cited sibling surfaces

carries selected limits, quotient stream equality, choice, trichotomy, host equality, or ambient completeness, the hub route exports none of those objects. □

Theorem 3991.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 3991.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 3991.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic

budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 3992

A Concrete Naming Certificate for $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ packet records the finite comparison surface for two displayed Cauchy tail-envelope reads before either branch reaches the terminal real-name seal. It sits after the dyadic tail envelope of chapter 1659, the tail-diameter boundary of chapter 1746, the finite stream schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the real-name boundary of chapter 38. The packet names a paired finite tail route; it is not quotient stream equality, a selected common limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3992.1 (Cauchy tail-envelope comparison carrier)**.** *A* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E_0, E_1, W_0, W_1, D, J, T, H, C, P, N)$$

with the following displayed rows:

1. E_0 *and* E_1 *are accepted* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows naming the two tail-envelope reads being compared.*
2. W_0 *and* W_1 *are the* $\mathsf{StreamName}^{\uparrow}$ *tail windows opened by the two envelope reads.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the two displayed tails.*
4. J *is the common-window row, recording the finite refinement through which* W_0 *and* W_1 *are read before either branch reaches a* $\mathsf{Real}^{\uparrow}$ *seal.*
5. T *is the paired tail-diameter row, recording the displayed comparison with the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *boundary.*
6. H *records componentwise* hsame *transport for the two envelope rows, stream windows, shared tolerance, common-window refinement, tail-diameter row, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a paired precision request through* E_0, E_1, J, D, T.
8. P *is the* Pkg *provenance over* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$, $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyTailEnvelopeComparisonUp.CauchyTailEnvelopeComparisonTasteGate`
`single carrier alignment`

Theorem 3992.2 (Cauchy tail-envelope comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *carrier*

$$Q = (E_0, E_1, W_0, W_1, D, J, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two dyadic tail-envelope rows, their stream windows, the shared dyadic tolerance, the common-window refinement, the paired tail-diameter row, transport, replay, provenance, and local naming rows;*
2. *envelope exposure, requiring both branches to display their* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows before the paired comparison is consumed;*
3. *common-window admission, requiring J to be fixed before either branch reads a* $\mathsf{RegSeqRat}^{\uparrow}$ *row toward* $\mathsf{Real}^{\uparrow}$*;*
4. *tolerance alignment, requiring both tail reads to use the same displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D;*
5. *diameter comparison, requiring the paired tail-diameter row T to be read through the same common window and tolerance data;*
6. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N;*
7. *refusal of quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and bridge checking as local certificate data.*

No comparison obligation is stored outside these rows.

Proof. Project the carrier of Definition 3992.1. The accepted packet exposes exactly the two dyadic tail-envelope rows E_0, E_1, their opened stream windows W_0, W_1, the shared tolerance row D, the common-window row J, the tail-diameter row T, and the structural rows H, C, P, N.

The envelope exposure clauses follow from the two displayed $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ rows. Each branch must open its finite stream window before the paired comparison can read any regular rational data.

The common-window and tolerance clauses are the role of J and D. They force the two branches to pass through one finite refinement and one dyadic budget before the paired tail-diameter row is consumed.

The remaining rows are structural. The hsame row transports each displayed coordinate componentwise, the Cont row replays the same paired route, and the Pkg and $\mathsf{NameCert}$ rows source and name only that finite surface.

Since none of the excluded principles occurs as a carrier coordinate, the local certificate cannot export quotient stream equality, a selected common limit, countable choice, trichotomy, host real equality, ambient Real completeness, or bridge checking. □

Theorem 3992.3 (Cauchy tail-envelope comparison common window)**.** *Every accepted* CauchyTailEnvelopeCompar *carrier compares its two displayed tail-envelope reads through a single finite common-window route*

$$E_0, E_1 \longrightarrow J \longrightarrow W_0, W_1 \longrightarrow D \longrightarrow T.$$

Consequently any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the paired comparison receives the same displayed common-window row before either branch can present a terminal real-name seal.*

Proof. Begin with the two accepted envelope rows. By Definition 1659.4, paired dyadic tail-envelope reads serving the same displayed precision request admit a finite common route before any $\mathsf{Real}^{\uparrow}$ seal is inspected.

The carrier row J records that common route for the comparison packet. Once J is fixed, both stream windows W_0 and W_1 are read as displayed rows over the same refinement rather than as completed streams.

The shared tolerance row D and the tail-diameter row T are then downstream from the common window. Thus every consumer of the paired comparison receives the displayed route through $E_0, E_1, J, W_0, W_1, D, T$ before any real-name seal is available. □

Theorem 3992.4 (Cauchy tail-envelope comparison nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *packet receives only the two displayed* $\mathsf{DyadicCauchyTailEnvelope}^{\uparrow}$ *rows, their finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, the common-window refinement, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. It receives no quotient stream, selected common limit, countable choice, trichotomy, host real equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or bridge-checking row.*

Proof. Project the carrier rows listed in Definition 3992.1. A consumer reaches the paired comparison only through $E_0, E_1, W_0, W_1, D, J, T, H, C, P, N$.

Apply Definition 3992.3. The common-window route places the two envelope rows, the finite refinement, the displayed windows, the shared tolerance, and the tail-diameter comparison before any terminal real-name seal can be read.

The excluded data are absent from the carrier and from the common-window route. Hence the comparison packet cannot export quotient streams, selected common limits, countable-choice schedules, trichotomy data, host real equality, ambient Real completeness, or a bridge-checking row to downstream Real or Completion consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ packet is generated from two displayed DyadicCauchyTailEnvelope rows, their StreamName tail windows, a shared DyadicRatCore tolerance row, a common-window refinement, a CauchyTailDiameter comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$, seed)

Scope closed: Definition 3992.1, Definition 3992.2, Definition 3992.3, and Definition 3992.4 seed the paired finite tail-envelope comparison route before either branch reaches Real or Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyTailEnvelopeComparisonUp.CauchyTailEnvelopeComparisonTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected common limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired envelope exposure, common-window admission, tolerance alignment, tail-diameter comparison, transport, replay, provenance, and nonescape for BEDC.Derived.CauchyTailEnvelopeCo

Chapter 3993

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompleteness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ packet records the constructive completeness readback surface for located regular real names as finite BEDC NameCert data. It sits after the compact interval neighbor of chapter 3986, the located interval surface of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite route by which a completion-facing consumer reads located Cauchy data; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, countable choice, selected limits, or open-cover compactness.

Definition 3993.1 (Bishop located completeness carrier)**.** *A* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, S, R, D, E, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, naming the located interval or located enclosure in which the inspected Cauchy data is read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite stream observations available at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row extracted from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling mesh, modulus, and endpoint precision requests.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed readback route.*
6. *B is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *boundary row, recording that any compact-interval assistance is consumed only through the finite route of the Bishop compact interval packet.*
7. *H records componentwise* hsame *transport for located interval, stream, regular, dyadic, Real, compact-boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from located requests through stream windows, regular readback, dyadic tolerance, Real sealing, compact-boundary control, provenance, and local naming.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, S, R, D, E, B, H, C, P, N$. It has no coordinate for host real completeness, quotient representatives, selected Cauchy limits, countable-choice schedules, open-cover compactness, or an unrestricted locator oracle.

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletenessUp.BishopLocatedCompletenessTasteGate single carrier alignment`

Theorem 3993.2 (Bishop located completeness NameCert obligations)**.** *For every accepted* BishopLocatedComplete *carrier $L = (I, S, R, D, E, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed located interval, stream-window, regular readback, dyadic tolerance, Real seal, compact-boundary, transport, replay, provenance, and naming rows; located source admission through I; finite observation through S; regular readback through R; dyadic tolerance accounting through D; Real sealing through E; compact-interval boundary control through B; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation is discharged by ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality of real names, a selected limit, countable choice, open-cover compactness, or a host locator outside the displayed packet.*

Proof. Project the carrier of Definition 3993.1. The tuple lists the complete finite row inventory admitted by the local certificate.

The located source is read through I, then finite observation and regular readback pass through S and R. Dyadic precision is controlled by D before the Real seal E is exposed.

The compact-interval row B contributes only the finite boundary route supplied by chapter 3986. It cannot be read as open-cover compactness or as ambient completeness.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or a host locator, no local $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$ obligation can export one. □

Theorem 3993.3 (Bishop located completeness Real-seal boundary)**.** *Every completion-facing read from an accepted* $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ *packet reaches* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{LocatedRealInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{BishopCompactInterval}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

The read exposes located interval data, finite stream windows, regular rational readback, dyadic tolerance control, compact-interval boundary control, Real sealing, transport, replay, provenance, and local naming, but it exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient real equality, selected limit, countable-choice schedule, or open-cover compactness principle.*

Proof. Let $L = (I, S, R, D, E, B, H, C, P, N)$ be an accepted carrier. A completion-facing consumer first receives I, S, R, D, so the located source, finite windows, regular readback, and dyadic tolerance route are fixed before any seal is visible.

The compact-boundary row B is then read as the Bishop compact interval neighbor of chapter 3986. It contributes only the finite interval boundary route accepted by that packet.

The Real-facing endpoint is E, and the structural route is H, C, P, N. These rows preserve the same finite path under transport, replay, provenance, and local naming. Because the classifier lacks coordinates for selected limits, quotient representatives, countable choice, open-cover compactness, and ambient completeness, the Real seal cannot be read as any of those stronger principles. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ completion-facing read passes through LocatedRealIntervalUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, BishopCompactIntervalUp, RealUp, transport, replay, provenance, and local naming rows before any Real-facing output is visible.

Independence witness: bishop-compact-interval, located-real-interval, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located source data, finite windows, regular rational readback, dyadic tolerance, compact-boundary control, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompleteness}^{\uparrow}$ packet is generated from a LocatedRealInterval source row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, a BishopCompactInterval boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3993.1, Definition 3993.2, and Definition 3993.3 seed the finite constructive located-completeness route over located interval, stream, regular, dyadic, compact-boundary, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient RealUp completeness, quotient real equality, selected limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located source admission, finite observation, regular readback, dyadic tolerance accounting, Real sealing, compact-boundary control, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCompletenessUp.

Chapter 3994

A Concrete Naming Certificate for $\mathsf{CauchyPrincipalFilter}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ packet records the principal-filter specialization used by Cauchy-filter and completion routes. It sits between the Cauchy-filter surface of chapter 333, the uniform-space entourage discipline of chapter 339, and the uniform-completion boundary of chapter 1473. The packet names the finite principal row that a completion consumer may inspect; it does not turn principal filters into quotient filters, selected limits, host equality, or ambient completeness.

Definition 3994.1 (CauchyPrincipalFilter carrier)**.** *A* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, G, F, U, B, H, K, P, N)$$

with the following displayed rows:

1. *A is the displayed center row naming the source point or finite address from which the principal filter is generated.*
2. *G is the neighbourhood and filter-base row: it lists the finite neighbourhood requests, the principal generator containing A, and the basis refinement witnesses accepted by the packet.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row, recording that the principal filter request is read as a Cauchy-filter request only through the displayed generator and basis witnesses.*
4. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *boundary row exposing the entourages through which the principal request is tested.*
5. *B is the* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary row reached after A, G, F, and U have been displayed.*
6. *H records componentwise* hsame *transport for the center, generator, Cauchy-filter handoff, uniform entourage, and completion-boundary rows.*
7. *K records the displayed* Cont *replay from the center row through neighbourhood requests, filter-base refinement, Cauchy-filter handoff, uniform-space testing, and completion-boundary readback.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$ *row.*

The carrier compares two accepted packets only by replaying the displayed rows $A, G, F, U, B, H, K, P, N$. It has no coordinate for quotient-filter equality, selected limits, ultrafilter extension, host equality of completion points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyPrincipalFilterUp.CauchyPrincipalFilterTasteGate single carrier alignment`

Theorem 3994.2 (CauchyPrincipalFilter NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier* $Q = (A, G, F, U, B, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$ *surface consists exactly of center admission through* A*, principal neighbourhood and filter-base admission through* G*, Cauchy-filter handoff through* F*, uniform-space testing through* U*, uniform-completion boundary exposure through* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* K*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 3994.1. The rows $A, G, F, U, B, H, K, P, N$ are the complete finite packet exposed to a consumer.

Read the principal-filter route in order. The center row A supplies the displayed generator, while G records the neighbourhood and filter-base witnesses through which the principal request is admitted.

Next follow the completion-facing handoff. The Cauchy-filter row F is available only after the generator and basis data have been displayed, and the uniform-space row U is the only entourage-testing surface used before a completion boundary is read.

Finally, B, H, K, P, and N supply boundary exposure, transport, replay, provenance, and local naming for the same rows. Since no coordinate stores quotient-filter equality, selected limits, ultrafilter extension, host equality, countable choice, or ambient completeness, no additional NameCert obligation can be exported from the packet. □

Theorem 3994.3 (CauchyPrincipalFilter convergence handoff)**.** *Every convergence-facing read exported by an accepted* $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ *carrier first exposes the displayed center row* A*, the neighbourhood and filter-base row* G*, the* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff row* F*, and the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row* U *before it reaches the* $\mathsf{UniformCompletion}^{\uparrow}$ *boundary row* B*. Thus the packet supplies a finite principal-filter route into Cauchy-filter and completion consumers without selecting a limit or quotienting filters.*

Proof. Start with a convergence-facing consumer. The carrier has exactly one source coordinate for the principal filter, namely the displayed center row A.

The neighbourhood row G then exposes the finite generator and basis refinement witnesses. A request that lacks those displayed witnesses has no route into the accepted packet.

The handoff row F places the request on the sibling $\mathsf{CauchyFilter}^{\uparrow}$ surface, and the uniform row U records the only entourages that the completion route may inspect. The boundary row B is therefore downstream of the finite principal-filter data rather than an independent completion object.

The transport, replay, provenance, and local naming rows H, K, P, N preserve only this route. Quotient filters, selected limits, host equality of completion points, countable choice, and ambient completeness remain outside the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPrincipalFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyPrincipalFilter}^{\uparrow}$ packet is generated from a displayed center row, neighbourhood and filter-base row, $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{UniformSpace}^{\uparrow}$ entourage boundary, $\mathsf{UniformCompletion}^{\uparrow}$ completion boundary, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchyPrincipalFilter}\uparrow}$, seed)

Scope closed: Definition 3994.1, Definition 3994.2, and Definition 3994.3 seed the finite principal-filter route from a displayed center through Cauchy-filter and uniform-completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyPrincipalFilter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-filter equality, selected limits, ultrafilter extension, host equality of completion points, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, principal generator exposure, filter-base refinement, Cauchy-filter handoff, uniform entourage testing, completion-boundary exposure, replay, provenance, and non-escape for the same finite packet.

Chapter 3995

A Concrete Naming Certificate for $\mathsf{KochCurve}^{\uparrow}$

Following Koch's classical curve construction, the $\mathsf{KochCurve}^{\uparrow}$ packet records the finite polygonal-stage, dyadic-schedule, and uniform-limit handoff data for the standard Koch curve before any completed planar subset is imported. It sits beside the compact binary address discipline of chapter 1475, the uniform-limit surface of chapter 463, the terminal real-name seal of chapter 38, the dyadic tolerance ledger of chapter 291, and the compact-uniform-continuity surface of chapter 388. The packet names only finite stage words, segment replacements, coordinate schedules, tolerance ledgers, and the displayed uniform-limit route; it does not import a host fractal subset, a selected infinite branch, quotient equality of real pairs, ambient compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 3995.1 (Koch curve carrier)**.** *A* $\mathsf{KochCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, S, D, W, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite address row. It records the inspected ternary stage, the finite left-turn and right-turn segment word, and the CantorSpace address window used by branch-facing consumers.*
2. *S is the polygonal-stage row, naming the directed segment list obtained by replacing each displayed segment by the four Koch generator segments at the next stage.*
3. *D is the dyadic displacement ledger. It records the rational endpoint deltas and the dyadic tolerance comparisons used to read the finite segment list.*
4. *W is the coordinate* $\mathsf{StreamName}^{\uparrow}$ *window row for the horizontal and vertical endpoint schedules attached to the inspected segment.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those coordinate schedules, with all tolerance comparisons routed through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
6. *E is the pair of terminal* $\mathsf{Real}^{\uparrow}$ *coordinate seals read from the same scheduled polygonal-stage rows.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, binding the stagewise polygonal approximations, dyadic tolerance ledger, and coordinate seals to one uniform-limit route.*

8. *H records componentwise* hsame *transport for addresses, segment stages, displacement ledgers, coordinate windows, rational readbacks, real seals, uniform-limit handoff, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from each curve-facing request through $A, S, D, W, Q, E, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{KochCurve}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, D, W, Q, E, L, H, C, P, N$. It has no coordinate for a completed planar fractal, selected infinite branch, quotient equality of real pairs, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.KochCurveUp.KochCurveTasteGate single carrier alignment`

Theorem 3995.2 (Koch curve NameCert obligations)**.** *For every accepted* $\mathsf{KochCurve}^{\uparrow}$ *carrier*

$$K = (A, S, D, W, Q, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; finite address exposure through A; polygonal-stage exposure through S; dyadic displacement and tolerance exposure through D; coordinate-window exposure through W; regular rational readback through Q; terminal real coordinate sealing through E; uniform-limit handoff through L; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N. No obligation is discharged by a host planar powerset, selected infinite address, quotient real-pair equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 3995.1. The accepted packet displays exactly the finite address row, polygonal-stage row, displacement ledger, coordinate windows, rational readbacks, real seals, uniform-limit handoff, and structural rows.

The finite combinatorial obligations are exhausted by A and S. A consumer cannot inspect a curve branch until the stage address is displayed, and it cannot read a segment replacement except through the directed segment list attached to that same finite stage.

The coordinate obligations are exhausted by D, W, Q, E. The dyadic ledger gives endpoint deltas and tolerance comparisons, the coordinate windows schedule the endpoint reads, the regular rational row reads those schedules, and the $\mathsf{Real}^{\uparrow}$ seals remain downstream of the same polygonal-stage data.

Finally L, H, C, P, N transport, replay, source, and name only the displayed route. Since the excluded host principles have no carrier coordinate, they cannot appear as local $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$ obligations. □

Theorem 3995.3 (Koch curve uniform-limit handoff)**.** *Every accepted* $\mathsf{KochCurve}^{\uparrow}$ *consumer that reaches a curve point, segment bound, or real-coordinate read must first expose the finite route*

$$\mathsf{CantorSpace}^{\uparrow} \longmapsto A, S \longmapsto D, W, Q, E \longmapsto \mathsf{UniformLimit}^{\uparrow} \longmapsto H, C, P, N.$$

The handoff exports only the displayed polygonal-stage approximations, dyadic tolerance ledger, scheduled coordinate readbacks, terminal $\mathsf{Real}^{\uparrow}$ *coordinate seals, and uniform-limit route. It exports no completed fractal subset, selected infinite branch, quotient equality of real coordinate pairs, ambient compactness theorem, or ambient completeness principle.*

Proof. Begin with a consumer that reaches the Real-facing boundary of the packet. By Definition 3995.1, the only terminal coordinate seals are downstream of the finite address, polygonal-stage, dyadic displacement, coordinate-window, and rational-readback rows.

Apply Definition 3995.2. It forces every curve-facing read through $A, S, D, W, Q, E, L, H, C, P, N$ and keeps the uniform-limit handoff on the same displayed packet.

The compact-uniform-continuity and uniform-limit dependencies supply only the finite modulus and convergence route named by L. They do not add a second carrier coordinate for a host planar subset, a selected infinite address, quotient equality of real coordinate pairs, or ambient completeness. Therefore every point, segment, or coordinate consumer remains inside the finite NameCert packet. □

Falsifiable prediction: An accepted $\mathsf{KochCurve}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ coordinate seal or a uniform-limit read without first displaying the finite address row, polygonal-stage row, dyadic displacement ledger, coordinate window, and rational readback refutes the carrier surface.

Independence witness: CantorSpace, UniformLimit, Real, DyadicRatCore, and CompactUniformContinuity: no listed sibling supplies the whole $\mathsf{KochCurve}^{\uparrow}$ packet, because each sibling contributes address windows, convergence handoff, terminal coordinate seals, dyadic tolerance data, or modulus discipline while this chapter binds all rows with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KochCurve}^{\uparrow}$ packet is generated from finite address rows, polygonal-stage segment rows, dyadic displacement ledgers, coordinate StreamName windows, RegSeqRat readbacks, RealUp coordinate seals, a UniformLimit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$, seed)

Scope closed: Definition 3995.1, Definition 3995.2, and Definition 3995.3 seed the finite polygonal-stage and uniform-limit certificate surface for the classical Koch curve.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KochCurve}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host planar powersets, selected infinite addresses, quotient equality of real coordinate pairs, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite address exposure, polygonal-stage exposure, dyadic displacement exactness, coordinate readback, Real coordinate sealing, uniform-limit handoff, transport, replay, provenance, and local naming for BEDC.Derived.KochCurveUp.

Chapter 3996

A Concrete Naming Certificate for $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$

The $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ packet realizes the channel boundary of chapter 9369 as finite BEDC NameCert data. It records source-event rows, legal channel-event rows, terminator-position rows, decoder-state rows, halting-boundary refusal rows, trace rows, transport, replay, provenance, and local naming in one displayed packet. The packet reads local event termination through the bounded evaluator and finite trace rows of chapter 818 before refusing certificate-level halting. The point of the packet is the stratum separation: a legal channel event can end locally, while certificate-level halting remains a refusal boundary rather than a global total classifier.

Definition 3996.1 (ChannelTerminationHaltingBoundary carrier)**.** *A* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, E, B, D, L, R, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-event row, naming the inspected finite binary source word and the source-layer reading of its glyphs.*
2. *E is the legal channel-event row, recording the encoded body over fragments* 0 *and* 10 *together with the displayed channel terminator* 11*.*
3. *B is the body-safety row, recording that the encoded body contains no adjacent* 11 *before the terminator row is exposed.*
4. *D is the decoder-state row, naming the left-to-right scan, the first displayed* 11*, and the recovered source event.*
5. *L is the legal-channel-event classifier row, accepting only event reads that pass through S, E, B, D before any boundary claim is made.*
6. *R is the halting-boundary refusal row. It records that local event termination is not a total classifier for arbitrary* $\mathsf{NameCert}$ *continuation chains.*
7. *F is the finite trace row, carrying the prompt-response and bounded evaluator evidence used to display the refusal without solving certificate-level halting.*
8. *H records componentwise* hsame *transport for source events, legal channel events, body safety, decoder state, classifier exposure, refusal, trace evidence, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from each boundary-facing request through* $S, E, B, D, L, R, F, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$, *haltingdistinction, haltingdistinctiontrace, promptresponsetrace, finitetraceevaluator,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, B, D, L, R, F, H, C, P, N$. It has no coordinate for a bare bitstream parser, a host global halting oracle, an unrestricted evaluator, a certificate-of-certificates total classifier, or a meta-stop proof.

Lean4 ✓ `BEDC.Derived.ChannelTerminationHaltingBoundaryUp.ChannelTerminationHaltingBoundaryTasteGate single carrier alignment`

Theorem 3996.2 (ChannelTerminationHaltingBoundary NameCert obligations)**.** *For every accepted* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ *carrier*

$$T = (S, E, B, D, L, R, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; source-event exposure through S; legal channel-event exposure through E; body-safety exposure through B; decoder-state exposure through D; legal-channel-event classification through L; halting-boundary refusal through R; finite trace evidence through F; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

No obligation is discharged by a bare bitstream parser, a host global halting oracle, an unrestricted evaluator, a total certificate-level halting classifier, or a meta-stop proof.

Proof. Project the carrier of Definition 3996.1. The accepted packet displays exactly the source event, legal channel event, body-safety, decoder-state, classifier, refusal, trace, and structural rows.

The local channel obligations are exhausted by S, E, B, D, L. The source row names the event being encoded, the legal channel-event row displays the body and terminator, the body-safety row excludes an earlier terminator pattern, and the decoder-state row reads the first 11 as the local endpoint.

The halting-boundary obligation is exhausted by R and F. The refusal row states that the local endpoint is not a certificate-level total halting classifier, while the finite trace row records only bounded evidence around prompt-response and evaluator routes.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for a global halting oracle or meta-stop proof, no local $\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}$ obligation can export one. □

Theorem 3996.3 (ChannelTerminationHaltingBoundary refusal factorization)**.** *Every accepted* $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ *packet factors local event termination through the displayed legal-channel-event route*

$$S, E, B, D, L \longmapsto R, F, H, C, P, N$$

before any halting-boundary refusal row is exposed. Thus a consumer may read the first channel-level 11 *as a local event endpoint only after the legal body and decoder rows have been displayed, and it may read the halting-boundary refusal only as the downstream statement that this local endpoint is not a certificate-level total halting classifier.*

Proof. Start with an accepted boundary-facing consumer. By Definition 3996.1, the only event-termination coordinates are S, E, B, D, L, so the consumer cannot reach the refusal row until the legal channel-event route is already displayed.

Apply Definition 3996.2. It binds body safety, first-terminator decoding, and legal-channel classification to the same finite packet before R and F can be read.

The refusal row then has only one role: it separates the local event endpoint from certificate-level halting. The finite trace row may record bounded prompt-response or evaluator evidence, but the packet has no coordinate for an unrestricted evaluator, a host global halting oracle, or a meta-stop proof. Therefore the refusal factors through the legal-channel-event route rather than replacing it. □

Falsifiable prediction: Every accepted packet factors local event termination through a displayed legal-channel-event row before any halting-boundary refusal row is exposed.

Independence witness: haltingdistinction, haltingdistinctiontrace, promptresponsetrace, finite-traceevaluator: these siblings either classify halting obstruction or trace evaluation, while this packet has the additional legal-channel terminator row and therefore is not bijective to any of them.

The Lean TasteGate namespace BEDC.Derived.ChannelTerminationHaltingBoundaryUp.TasteGate must provide Nontrivial and FieldFaithful instances for this finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}$ packet is generated from source-event rows, legal channel-event rows, body-safety rows, decoder-state rows, legal-channel classifiers, halting-boundary refusal rows, finite trace evidence, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3996.1, Definition 3996.2, and Definition 3996.3 seed the finite local-channel boundary separating event termination from certificate-level halting refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChannelTerminationHaltingBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted halting classification, host global halting oracles, bare bitstream parsing, meta-stop proofs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal channel-event exposure, body-safety exposure, decoder-state exposure, local termination factorization, halting-boundary refusal, finite trace evidence, transport, replay, provenance, and local naming for BEDC.Derived.ChannelTerminationHaltingBoundaryUp.

Chapter 3997

A Concrete Naming Certificate for $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$

Following Bishop and Bridges on constructive completion and reflector discipline, the CauchyCompletionUniversalArr packet records the universal arrow of the L10 Cauchy completion route as finite NameCert data. It ties the reflector surface of chapter 1585, the regular unit of chapter 3646, the counit of chapter 1539, the idempotent comparison of **??**, and the metric universal property of chapter 2124 to the regular rational and real seals of chapter 301 and chapter 38. The packet names the universal-arrow boundary without quotient completions, selected limits, host function extensionality, ambient completeness, or bridge checking.

Definition 3997.1 (CauchyCompletionUniversalArrow carrier)**.** *A* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row opened by the universal-arrow request.*
2. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row naming the completion reflector over M.*
3. *η is the* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *row exposing the source-to-completion unit.*
4. *F is the uniformly continuous map row from the source metric into the displayed complete target.*
5. *$\widehat{F}$ is the factor arrow row from the reflected completion to the target named by F.*
6. *Q is the uniqueness comparison row, requiring any competing factor arrow with the same precomposition along η to compare through the displayed NameCert route.*
7. *S is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row reached after $R, \eta, F, \widehat{F}, Q$ have been displayed.*
8. *H records componentwise* hsame *transport for source metrics, reflector rows, units, uniformly continuous maps, factor arrows, uniqueness comparisons, seal rows, replay rows, provenance rows, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M, R, \eta, F \longmapsto \widehat{F}, Q, S.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotent}^{\uparrow}$, MetricCompletionUniv $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completions, selected Cauchy limits, host function extensionality, ambient completeness of a target space, or bridge-checked equality.

Lean4 ✓ `BEDC.Derived.CauchyCompletionUniversalArrowUp.CauchyCompletionUniversalArrowTasteGate`
`single carrier alignment`

Theorem 3997.2 (CauchyCompletionUniversalArrow NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ *carrier* $U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionUni}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the source metric, completion reflector, unit, uniformly continuous map, factor arrow, uniqueness comparison, RegSeqRat/Real seal, transport, replay, provenance, and local naming rows;*
2. *reflector anchoring, requiring the rows R and η to be displayed before any universal-arrow factor can be inspected;*
3. *map compatibility, requiring the source map row F to be read as uniformly continuous along the same source metric row M;*
4. *factorization visibility, requiring $\widehat{F}$ to be exported only as the factor arrow whose precomposition along η replays F through C;*
5. *uniqueness comparison, requiring Q to compare all competing factor rows by the same displayed route rather than by host function extensionality;*
6. *RegSeqRat/Real seal nonescape, requiring S, H, C, P, N to exhaust the regular rational seal, real seal, transport, replay, provenance, and local naming surface.*

Proof. Project the accepted carrier $U = (M, R, \eta, F, \widehat{F}, Q, S, H, C, P, N)$. Carrier admission exposes exactly the rows listed in Definition 3997.1.

The universal-arrow request is anchored by M, R, η. The source metric is visible before the reflector row is consumed, and the unit row is the only source-to-completion boundary available to the factorization read.

The map and factor rows are then read together. The row F supplies the uniformly continuous source map, while $\widehat{F}$ is exported only through the continuation replay that identifies precomposition with η as the displayed source map route.

Finally Q, S, H, C, P, N close the comparison and sealing surface. Since quotient completions, selected limits, host function extensionality, ambient completeness, and bridge checking are absent from the carrier, the listed rows exhaust the local certificate. □

Theorem 3997.3 (CauchyCompletionUniversalArrow factorization)**.** *Every accepted* CauchyCompletionUniversalA *packet exports a universal-arrow factorization route*

$$M \longmapsto \mathsf{CauchyCompletionReflector}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUnit}^{\uparrow} \longmapsto F \longmapsto \widehat{F} \longmapsto Q \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}$$

Any downstream completion consumer receives only the source metric row, reflector row, unit row, uniformly continuous map row, factor arrow row, uniqueness comparison row, RegSeqRat/Real seal

row, componentwise transport, displayed replay, provenance, and local naming. It receives no quotient completion object, selected limit, host function extensionality principle, ambient completeness assertion, or bridge-checked equality.

Proof. Open the carrier and read M, R, η first. These rows display the source metric, the Cauchy completion reflector, and the regular unit that a universal-arrow consumer may inspect.

Next read F and $\widehat{F}$. The factor arrow is not an ambient function-space object; it is the displayed row whose continuation replay agrees with F after the unit row.

The uniqueness comparison row Q then compares any other factor row through the same displayed precomposition route. This is a certificate comparison, not a use of host function extensionality.

The route terminates at S, H, C, P, N. Regular rational and Real sealing, transport, replay, provenance, and local naming preserve the ordered handoff and exclude quotient completions, selected limits, ambient completeness, and bridge checking from the exported packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}$ packet is generated from a source metric row, a CauchyCompletionReflector row, a regular completion unit row, a uniformly continuous map row, a factor arrow row, a uniqueness comparison row, RegSeqRat and Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3997.1, Definition 3997.2, and Definition 3997.3 seed the finite universal-arrow route for the L10 Cauchy completion reflector.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionUniversalArrow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host function extensionality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector anchoring, unit exposure, map compatibility, factorization visibility, uniqueness comparison, RegSeqRat/Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionUniversalArrowUp.

Chapter 3998

A Concrete Naming Certificate for CellularCarrierBHistEquivalence$^{\uparrow}$

Following Wolfram's cellular-automaton carrier reading and Cook's Rule 110 universality result, the CellularCarrierBHistEquivalence$^{\uparrow}$ packet records the carrier-equivalence content of chapter 9391 as finite NameCert data. It sits beside the automaton surface of chapter 934, the orbit substrate of chapter 938, the pattern catalog of chapter 995, the trust substrate of chapter 940, and the substrate classifier boundary of chapter 3810. The packet names a finite BHist encoding and orbit-window comparison without replacing the BEDC carrier.

Definition 3998.1 (CellularCarrierBHistEquivalence carrier)**.** *A* CellularCarrierBHistEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, B, D, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the cellular orbit-window row: a finite initial row, width boundary, time window, and local update discipline.*
2. *B is the BHist encoding row that reads a finite* BHist *value as a finite cell-row code.*
3. *D is the decoder row that reads the same finite cell-row code back as a* BHist *value when the orbit window remains inside the accepted boundary.*
4. *R is the roundtrip row, requiring the encode-decode and decode-encode reads to agree on the displayed finite window.*
5. *K is the classifier transport row moving accepted cellular observations to the corresponding* BHist *classifier observations and back.*
6. *H records componentwise* hsame *transport for orbit windows, encodings, decoders, roundtrip rows, classifier transport rows, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$W, B, D, R \longmapsto K.$$

8. *P is the* Pkg *provenance over* CellularCarrierBHistEquivalence$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, CellularOrbitSubstrate$^{\uparrow}$, CellularPatternCatalog$^{\uparrow}$, CellularTrustSubstrate$^{\uparrow}$, CellularSubstrateClassifierBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}$ *row.*

The classifier accepts exactly packets with these nine rows. It has no coordinate for a replacement BEDC kernel carrier, infinite cellular orbits, global physical substrate claims, hidden evaluator state, host bisimulation, or non-finite classifier equality.

Lean4 ✓ `BEDC.Derived.CellularCarrierBHistEquivalenceUp.CellularCarrierBHistEquivalenceTasteGate`
`single carrier alignment`

Theorem 3998.2 (CellularCarrierBHistEquivalence NameCert obligations)**.** *For every accepted* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ *carrier* $E = (W, B, D, R, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalenc}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for orbit-window, BHist encoding, decoder, roundtrip, classifier transport, transport, replay, provenance, and local naming rows;*
2. *orbit-window finiteness, requiring all cellular reads to remain inside the displayed width and time window;*
3. *encoding and decoding visibility, requiring B and D to be inspected before any carrier-equivalence claim is exported;*
4. *roundtrip and classifier transport, requiring R and K to bind finite BHist reads and finite cellular-window reads through the same accepted packet;*
5. *non-escape from carrier replacement, infinite orbit semantics, physical substrate claims, hidden evaluator state, host bisimulation, and non-finite classifier equality.*

Proof. Carrier admission is projection from Definition 3998.1. The tuple displays exactly the orbit-window, encoding, decoder, roundtrip, classifier transport, transport, replay, provenance, and naming rows.

The cellular side is finite before any equivalence read begins. The row W fixes the initial row, width boundary, time window, and local update discipline inspected by the packet.

The BHist side is read through B and D. These rows expose encoding and decoding as finite carrier reads rather than as a replacement of the BEDC kernel carrier.

The roundtrip and classifier rows R and K then bind the two directions over the same displayed window. Transport, replay, provenance, and local naming preserve that binding, while the absent coordinates rule out infinite orbit semantics, physical substrate claims, host bisimulation, and non-finite classifier equality. □

Theorem 3998.3 (Cellular carrier BHist equivalence route)**.** *Every accepted* $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{CellularOrbitSubstrate}^{\uparrow} \longmapsto W \longmapsto B \longmapsto D \longmapsto R \longmapsto K \longmapsto H, C, P, N.$$

The route realizes the cellular-carrier alternative as a finite comparison between orbit-window observations and BHist *observations. It does not export a new kernel carrier, an infinite orbit semantics, a global physical substrate assertion, hidden evaluator state, host bisimulation, or non-finite classifier equality.*

Proof. Open W first. The orbit-window row supplies the finite cellular observation boundary and therefore prevents the route from depending on an infinite orbit.

Read B and D next. Encoding and decoding connect the finite window code with a finite BHist value, so the comparison is a NameCert packet rather than a carrier substitution.

Use R and K to bind the roundtrip and classifier transport. These rows keep cellular and BHist observations in the same finite acceptance boundary.

Finally consume H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the finite route and leave no carrier-replacement, physical-substrate, hidden-evaluator, host-bisimulation, or non-finite-equality coordinate to export. □

Falsifiable prediction: Every accepted $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ read factors through a finite orbit-window row, a BHist encoding row, a decoder row, a roundtrip row, classifier transport, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: cellularautomaton, cellularorbitsubstrate, cellularpatterncatalog, cellular-trust-substrate, cellular-substrate-classifier-boundary: $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ is not bijective to any listed sibling because it jointly packages a finite orbit window, BHist encoding, decoding, roundtrip, classifier transport, replay, provenance, and local naming as one carrier-equivalence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}$ packet is generated from a finite cellular orbit-window row, a BHist encoding row, a decoder row, a roundtrip row, a classifier transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3998.1, Definition 3998.2, and Definition 3998.3 seed the finite carrier-equivalence route between cellular orbit-window observations and BHist observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularCarrierBHistEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close carrier replacement, infinite cellular orbit semantics, global physical substrate claims, hidden evaluator state, host bisimulation, non-finite classifier equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-window finiteness, encoding and decoding visibility, roundtrip comparison, classifier transport, transport, replay, provenance, and local naming for BEDC.Derived.CellularCarrierBHistEquivalenceUp.

Chapter 3999

A Concrete Naming Certificate for $\mathsf{RegularSequenceBishop}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularSequenceBishop}^{\uparrow}$ packet records the regular-sequence presentation of a constructive real as finite NameCert data. It reads finite observations through chapter 126, measures tolerances through chapter 291, passes rational readback through chapter 301, remains compatible with the Cauchy surface of chapter 3413, and reaches the terminal seal of chapter 38. The packet is induction-facing: it exposes bounded finite windows and dyadic tolerance ledgers before any $\mathsf{Real}^{\uparrow}$ seal is consumed, and it does not quotient streams, select a host limit, invoke countable choice, or assume ambient Real completeness.

Definition 3999.1 (Bishop regular-sequence carrier)**.** *A* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W, D, Q, R, N)$$

with the following five displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the bounded stream segment inspected at a requested precision.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the positive dyadic scale, comparison cells, and slack ledger used to compare the window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, turning the inspected window and tolerance ledger into regular rational sequence evidence.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after W, D, Q have been displayed.*
5. *N is the local structural row, bundling componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$ *naming row for the same packet.*

The classifier compares accepted packets only through W, D, Q, R, N. It has no coordinate for quotient stream equality, host real equality, a selected limit, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularSequenceBishopUp.RegularSequenceBishopTasteGate single carrier alignment`

Theorem 3999.2 (Bishop regular-sequence NameCert obligations)**.** *For every accepted* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier $B = (W, D, Q, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$ *surface consists exactly of five*

obligation rows: finite $\mathsf{StreamName}^{\uparrow}$ *window exposure through* W*; dyadic tolerance and comparison exposure through* D*;* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback through* Q*; downstream* $\mathsf{Real}^{\uparrow}$ *seal handoff through* R*; and local transport, replay, provenance, and naming through* N*. Every Real and Completion consumer must read*

$$W, D \longmapsto Q \longmapsto R \longmapsto N$$

before treating the packet as a Bishop-style regular sequence.

Proof. Project the accepted carrier of Definition 3999.1. The visible coordinates are exactly W, D, Q, R, N, and the classifier contains no further row.

The window and tolerance part is read first. The row W supplies only a finite $\mathsf{StreamName}^{\uparrow}$ observation, while D supplies the dyadic precision and comparison ledger attached to that observation.

The regular rational handoff is then forced through Q. Since Q is the only row that converts the window and tolerance data into $\mathsf{RegSeqRat}^{\uparrow}$ evidence, no host stream quotient or selected limit can enter the local surface.

Finally R and N deliver the Real-facing boundary and the structural NameCert support for the same displayed route. The absence of coordinates for quotient equality, host real equality, countable choice, trichotomy, or ambient completeness gives the stated exhaustiveness. □

Theorem 3999.3 (Bishop regular-sequence induction window)**.** *Let* $B = (W, D, Q, R, N)$ *be an accepted* $\mathsf{RegularSequenceBishop}^{\uparrow}$ *carrier. Any induction over the regular-sequence presentation may inspect only finite* $\mathsf{StreamName}^{\uparrow}$ *windows from* W*, compare them through the dyadic ledger* D*, and pass the resulting rational data through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *before the* $\mathsf{Real}^{\uparrow}$ *seal row* R *is consumed. The induction route is therefore*

$$W, D \longmapsto Q \longmapsto R \longmapsto N.$$

It does not decide quotient equality of streams, choose a host limit, use countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier projection in Definition 3999.1. An induction step can open the finite window row W and the tolerance row D, because those are the only displayed sources available before any rational readback.

The step then passes through Q. This row is the sole $\mathsf{RegSeqRat}^{\uparrow}$ readback coordinate, so every induction witness remains tied to finite windows and dyadic tolerances rather than to quotient representatives or chosen limits.

Only after that readback may the route consume R, and N preserves the same route by transport, replay, provenance, and local naming. Thus the induction window is finite and Bishop-style, with the excluded nonconstructive and completeness coordinates absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{RegularSequenceBishop}^{\uparrow}$ read used by a Real and Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and one local structural NameCert row before it is treated as a regular-sequence presentation.

Independence witness: streamname, dyadicratcore, regseqrat, cauchy, real: $\mathsf{RegularSequenceBishop}^{\uparrow}$ is not bijective to any listed sibling because it binds finite stream windows, dyadic tolerance comparisons, regular rational readback, a Real-facing seal, and local NameCert structure into one induction-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceBishop}^{\uparrow}$ packet is generated from a StreamName finite-window row, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 3999.1, Definition 3999.2, and Definition 3999.3 seed the Bishop-style regular-sequence route over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceBishop}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host real equality, selected limits, countable choice, trichotomy, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, RegSeqRat readback, Real seal ordering, transport, replay, provenance, local naming, and induction-window nonescape for BEDC.Derived.RegularSequenceBishopUp.

Chapter 4000

A Concrete Naming Certificate for $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ packet records a uniform Cauchy modulus for a family of real-valued approximants as finite NameCert data. It sits after the finite source windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the criterion surface of chapter 3876, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer boundary. The packet gives Real and Completion routes a finite uniform-Cauchy bridge; it does not assert host function equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4000.1 (Modulus of uniform convergence carrier)**.** *A* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, F, U, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row, naming the finite domain observations inspected for each requested uniform tolerance.*
2. *F is the function-family row, recording the displayed approximant indices and the finite value reads used in the comparison.*
3. *U is the uniform tolerance and modulus row, exposing the requested precision and the threshold delivered to the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer surface.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic comparison cells, positivity witnesses, and slack data used by the uniform estimate.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows, family reads, tolerance ledger, and modulus threshold.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, F, U, D, R have been displayed.*
7. *H records componentwise* hsame *transport for source windows, family reads, tolerance data, readback, seal, continuation, provenance, and naming.*
8. *C records displayed* Cont *replay from a uniform-Cauchy request through $S, F, U, D, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, U, D, R, E, H, C, P, N$. It has no coordinate for host equality of functions, a countable-choice schedule, a selected limiting function, quotient equality of real names, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.ModulusOfUniformConvergenceUp.ModulusOfUniformConvergenceUp`

Theorem 4000.2 (Modulus of uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ *carrier*

$$M = (S, F, U, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$ *surface consists exactly of source-window exposure through S, function-family read exposure through F, uniform tolerance and modulus exposure through U, dyadic tolerance-ledger exposure through D, regular rational readback through R, real seal handoff through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. Every Real or Completion consumer must read*

$$S, F, U, D \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before treating the packet as a uniform convergence modulus.

Proof. Project the carrier of Definition 4000.1. The visible coordinates are exactly $S, F, U, D, R, E, H, C, P, N$, and the classifier contains no further analytic row.

The source and family data are read first. The row S supplies finite $\mathsf{StreamName}^{\uparrow}$ windows, while F supplies the displayed approximant reads attached to those windows.

The uniform estimate is then forced through U and D. The row U names the common threshold and tolerance request, and D supplies the dyadic ledger that makes the comparison finite.

Finally R, E, H, C, P, N deliver regular rational readback, the real-facing seal, transport, replay, provenance, and local naming for the same displayed route. Since host function equality, countable choice, selected limits, quotient real-name equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, the listed obligations exhaust the certificate surface. □

Theorem 4000.3 (Modulus of uniform convergence Cauchy-uniform route)**.** *Every accepted* ModulusOfUniformCon *packet consumed by* $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ F,\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes the same rows $S, F, U, D, R, E, H, C, P, N$ and refuses host function equality, countable choice, selected limits, quotient equality of real names, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier. A criterion or modulus consumer first receives the source-window row S and the family-read row F, so the request is grounded in finite observations rather than in a host function object.

Next read the uniform modulus row U together with the dyadic ledger D. These rows give a single threshold and tolerance comparison surface for all displayed windows, which is precisely the finite boundary expected by $\mathsf{UniformConvergenceCauchyCriterion}^{\uparrow}$ and $\mathsf{UniformModulus}^{\uparrow}$.

The only downstream analytic readback is R followed by E. Thus $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers receive data only after the uniform-Cauchy route has been displayed.

The transport, replay, provenance, and naming rows H, C, P, N preserve the same route. No excluded host equality, choice schedule, selected limit, quotient equality, or completeness coordinate can be inserted by a consumer of the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ read used by a Real or Completion consumer displays one $\mathsf{StreamName}^{\uparrow}$ source-window row, one function-family read row, one $\mathsf{UniformModulus}^{\uparrow}$ tolerance-threshold row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N before it is treated as a uniform convergence modulus.

Independence witness: streamname, dyadicratcore, regseqrat, real, uniformmodulus, uniform-convergence-cauchy-criterion: $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ is not bijective to any listed sibling because only it binds finite source windows, displayed family reads, one uniform threshold, a dyadic tolerance ledger, regular rational readback, a real seal, and local NameCert support into one uniform-Cauchy bridge.

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfUniformConvergence}^{\uparrow}$ packet is generated from StreamName source windows, function-family read rows, a UniformModulus tolerance-threshold row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4000.1, Definition 4000.2, and Definition 4000.3 seed the finite uniform-Cauchy bridge over StreamName, DyadicRatCore, RegSeqRat, Real, UniformConvergenceCauchyCriterion, and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, countable choice, selected limits, quotient equality of real names, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window exposure, function-family read exposure, uniform threshold exposure, dyadic tolerance-ledger exposure, readback, real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfUniformConvergenceUp.

Chapter 4001

A Concrete Naming Certificate for $\mathsf{DyadicBisectionSchedule}^{\uparrow}$

Following Bishop and Bridges constructive nested-interval and bisection arguments, the $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ packet records the finite halving schedule used before a retained interval route is read as Cauchy data. It sits beside the dyadic tolerance core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located interval-zero route of chapter 3971, and the retained closed-interval branch surface of chapter 3815. The packet names a finite schedule object only; it does not select an infinite branch, assert an interval intersection point, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4001.1 (Dyadic bisection schedule carrier)**.** *A* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, B, D, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed interval source row. It exposes endpoint order, containment, and the finite retained-window requests consumed from the* $\mathsf{ClosedIntervalBisection}^{\uparrow}$ *route.*
2. *B is the branch schedule row. It records the inspected finite sequence of left-or-right bisection decisions, midpoint exposure, retained-child containment, and width-halving ledger.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row. It records the positive dyadic mesh, endpoint comparison cells, midpoint arithmetic cells, and halving bounds used by B.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for endpoint, midpoint, retained-child, and schedule-prefix observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after I, B, D, W have been displayed.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, opened only after the finite schedule and regular readback rows are exposed.*
7. *H records componentwise* hsame *transport for the interval source, branch schedule, dyadic scale, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located interval or Cauchy request through $I, B, D, W, R, Q, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$, $\mathsf{ClosedIntervalBisection}^{\uparrow}$, $\mathsf{LocatedIntervalBolzano}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$ *row.*
The classifier compares accepted schedule carriers only through $I, B, D, W, R, Q, H, C, P, N$. It has no coordinate for an infinite choice sequence, a completed nested intersection, quotient real equality, host trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicBisectionScheduleUp.DyadicBisectionScheduleTasteGate single carrier alignment`

Theorem 4001.2 (Dyadic bisection schedule NameCert obligations)**.** *For every accepted* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier $S = (I, B, D, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$ *surface consists exactly of carrier admission, closed-interval source exposure, finite branch-schedule exposure, dyadic halving exactness, stream-window exposure, regular rational readback, terminal real sealing, componentwise transport, displayed replay, package provenance, and local naming for the carrier of Definition 4001.1.*

Proof. Carrier admission is projection from Definition 4001.1. The interval row I and branch row B are the only rows that expose closed-interval bisection data, so every schedule obligation starts with the displayed finite interval source and retained-child ledger.

The dyadic and stream obligations are consecutive. The row D supplies midpoint arithmetic, positive mesh data, and halving bounds before W can open endpoint, midpoint, retained-child, or prefix observations.

The readback rows R, Q are therefore downstream of I, B, D, W. The structural rows H, C, P, N transport, replay, source, and name the same finite route.

Since the carrier contains no coordinate for infinite branch choice, completed intersection points, quotient real equality, trichotomy, open-cover compactness, or ambient completeness, the listed obligations exhaust the local certificate. □

Theorem 4001.3 (Dyadic bisection schedule Cauchy handoff)**.** *Every Cauchy- or Bolzano-facing consumer of an accepted* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ *carrier reaches regular rational data only through the displayed route*

$$\mathsf{ClosedIntervalBisection}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may be consumed by located interval and completion routes as a finite halving schedule, but it exports no selected infinite branch, completed intersection point, quotient real representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $S = (I, B, D, W, R, Q, H, C, P, N)$. The source row I and the retained branch schedule B expose the only finite bisection information available to the consumer.

Read the dyadic row D next. It fixes the midpoint and width-halving budget through displayed dyadic cells, so the schedule cannot be interpreted as an external sequence of arbitrary cuts.

The stream and regular rows W, R then turn the displayed finite windows into regular rational readback. The terminal row Q is a Real-facing seal only after this finite route has been read.

Finally H, C, P, N preserve the same order under transport, replay, provenance, and local naming. Because no stronger object appears among the carrier coordinates, every Cauchy handoff remains the displayed finite schedule route. □

Falsifiable prediction: A Cauchy or Bolzano consumer that omits the branch schedule row, dyadic halving row, StreamName window row, or RegSeqRat readback row cannot claim a $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ handoff to a Real-facing seal.

Independence witness: closed-interval-bisection, located-interval-bolzano, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an interval source, finite branch schedule, dyadic halving ledger, stream windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicBisectionSchedule}^{\uparrow}$ packet is generated from a closed-interval source row, a finite retained-branch schedule row, a DyadicRatCore halving ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4001.1, Definition 4001.2, and Definition 4001.3 seed the finite dyadic interval-halving route over ClosedIntervalBisectionUp, LocatedIntervalBolzanoUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBisectionSchedule}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected infinite branches, completed interval intersections, quotient real equality, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, finite branch-schedule exposure, dyadic halving exactness, StreamName window exactness, RegSeqRat readback, Real-seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.DyadicBisectionScheduleUp.

Chapter 4002

A Concrete Naming Certificate for $\mathsf{DyadicMidpointRefinement}^{\uparrow}$

Following Bishop and Bridges constructive interval refinement, the $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ packet records the finite radius-halving step that turns a displayed dyadic interval into the retained dyadic subinterval used by regular rational and real-name consumers. It sits after the dyadic tolerance core of chapter 291, the midpoint row of chapter 445, the finite schedule discipline of chapter 4001, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite midpoint refinement only; it does not assert interval completeness, trichotomy, quotient endpoint equality, or a selected limit.

Definition 4002.1 (Dyadic midpoint refinement carrier)**.** *A* $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (E, M, B, W, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the pair of dyadic endpoint rows read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, including the displayed radius and its halved successor.*
2. *M is the* $\mathsf{DyadicMidpoint}^{\uparrow}$ *common-denominator midpoint row for the same endpoint window.*
3. *B is the retained-side row naming which endpoint is replaced by the midpoint in the finite refinement step.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which downstream consumers inspect the retained endpoints.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after W and Q have been displayed.*
7. *H records componentwise* hsame *transport for the endpoint, midpoint, branch, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a refinement request through E, M, B, W, Q, A.*
9. *P is the* Pkg *provenance over* $\mathsf{DyadicMidpointRefinement}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicMidpoint}^{\uparrow}$*,* $\mathsf{DyadicBisectionSchedule}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpointRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed nested-interval point, quotient endpoint equality, trichotomy, selected branch limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Theorem 4002.2 (Dyadic midpoint refinement NameCert obligations)**.** *For every accepted* DyadicMidpointRefinem *carrier* $R = (E, M, B, W, Q, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicMidpointRefinement}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint exposure through* $\mathsf{DyadicRatCore}^{\uparrow}$*, midpoint exposure through* $\mathsf{DyadicMidpoint}^{\uparrow}$*, retained-side exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4002.1. The endpoint rows E, midpoint row M, retained-side row B, windows W, readback row Q, seal row A, and structural rows H, C, P, N are the only admitted coordinates.

Endpoint exposure is supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ rows in E, while midpoint exposure is supplied by the $\mathsf{DyadicMidpoint}^{\uparrow}$ row M. The retained-side row B records the finite replacement of one endpoint by the displayed midpoint, so the refinement step has no hidden real-valued selector.

The downstream read is forced through W, Q, A. StreamName windows expose the finite endpoint observations, RegSeqRat readback translates those observations into regular rational rows, and the Real seal is consumed only after both are visible. Transport, replay, provenance, and local naming preserve exactly this finite route, so no completeness, trichotomy, quotient, or selected-limit row can be exported. □

Theorem 4002.3 (Dyadic midpoint refinement radius halving)**.** *Let* $R = (E, M, B, W, Q, A, H, C, P, N)$ *be an accepted* $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ *carrier. Every Real- or Completion-facing consumer of the retained dyadic subinterval must read the original dyadic endpoint radius and the halved successor inside the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger* E*, then consume the* $\mathsf{DyadicMidpoint}^{\uparrow}$ *row* M*, the retained-side row* B*, the finite* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and only then the* $\mathsf{Real}^{\uparrow}$ *seal* A*.*

Proof. Begin with the endpoint ledger. The radius and its halved successor are both rows of E, so a consumer cannot replace the displayed halving step by host interval arithmetic or by a detached real-radius comparison.

Read the midpoint and retained side next. The common-denominator midpoint row M supplies the only midpoint used by the refinement, and B records the finite endpoint replacement that keeps the retained subinterval inside the same dyadic packet.

Finally route the retained endpoints through W, Q, A. The finite StreamName windows expose the observed endpoint rows, RegSeqRat readback supplies the regular rational data, and the Real seal is downstream of that readback. Thus the radius-halving route reaches Real and Completion consumers only as displayed finite NameCert data, not as completeness, trichotomy, quotient equality, or selected-limit evidence. □

Falsifiable prediction: A Real or Completion consumer that reads a retained dyadic subinterval without the $\mathsf{DyadicRatCore}^{\uparrow}$ halving row, the $\mathsf{DyadicMidpoint}^{\uparrow}$ midpoint row, the $\mathsf{StreamName}^{\uparrow}$ windows, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row cannot claim a $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ handoff to a $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: dyadicratcore, dyadicmidpoint, dyadic-bisection-schedule, streamname, regseqrat, real: $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint-radius halving, a displayed midpoint calculation, a retained-side row, finite windows, regular readback, terminal sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicMidpointRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicMidpointRefinement}^{\uparrow}$ packet is generated from dyadic endpoint rows, a displayed midpoint row, a retained-side row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicMidpointRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4002.1, Definition 4002.2, and Definition 4002.3 seed the finite midpoint-refinement surface from dyadic radius halving to Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicMidpointRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, interval completeness, trichotomy, quotient endpoint equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, midpoint exactness, radius-halving exactness, StreamName window exposure, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local NameCert field faithfulness for $\mathsf{DyadicMidpointRefinement}^{\uparrow}$.

Chapter 4003

A Concrete Naming Certificate for RegularCauchyEnclosureFilter$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RegularCauchyEnclosureFilter$^{\uparrow}$ packet records a finite enclosure-filter route for regular Cauchy names. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy-filter basis surface of chapter 333. The packet gives Real and Completion consumers a displayed enclosure-filter certificate without quotienting streams, selecting a host limit, assuming ambient completeness, or identifying Cauchy filters by external equality.

Definition 4003.1 (Regular Cauchy enclosure-filter carrier)**.** *A* RegularCauchyEnclosureFilter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, D, Q, F, B, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* StreamName$^{\uparrow}$ *finite-window row naming the inspected rational-name segment and the requested precision window.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *enclosure ledger, recording the positive dyadic radius, interval endpoints, midpoint comparison, and slack data used by the window.*
3. *Q is the* RegSeqRat$^{\uparrow}$ *regular readback row obtained only after W and D have been displayed.*
4. *F is the* CauchyFilter$^{\uparrow}$ *basis row, listing the finite filter requests and refinements whose accepted members contain the same dyadic enclosures.*
5. *B is the enclosure-basis compatibility row, matching each displayed regular Cauchy window with a filter-basis request at the same dyadic radius.*
6. *R is the downstream* Real$^{\uparrow}$ *seal row, available only after W, D, Q, F, B have been consumed.*
7. *H records componentwise* hsame *transport for the window, dyadic enclosure, regular readback, Cauchy-filter basis, compatibility, and Real-seal rows.*
8. *C records displayed* Cont *replay from a finite window and radius request through regular readback and filter-basis compatibility to the Real-facing seal.*
9. *P is the* Pkg *provenance over* RegularCauchyEnclosureFilter$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CauchyFilter$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, D, Q, F, B, R, H, C, P, N$. It has no coordinate for quotient stream equality, quotient filter equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEnclosureFilterUp.RegularCauchyEnclosureFilterTasteGate`
`single carrier alignment`

Theorem 4003.2 (Regular Cauchy enclosure-filter NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ *carrier $E = (W, D, Q, F, B, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}}$ *surface consists exactly of finite window exposure, dyadic enclosure admission, regular rational readback, Cauchy-filter basis admission, enclosure-basis compatibility, Real-seal ordering, componentwise transport, continuation replay, provenance, and local naming. Every consumer must read the finite route*

$$W, D \longmapsto Q \longmapsto F, B \longmapsto R$$

together with H, C, P, N before treating the packet as an enclosure-filter certificate for a regular Cauchy name.

Proof. Project the carrier of Definition 4003.1. The rows $W, D, Q, F, B, R, H, C, P, N$ are the only displayed coordinates available to the local classifier.

The regular Cauchy side is read first. The window row W supplies a finite observation, the dyadic row D supplies the enclosing radius and endpoint ledger, and the readback row Q is downstream of those two rows.

The filter side is then forced through F and B. The basis row F exposes only finite Cauchy-filter requests, while B records the compatibility between those requests and the same dyadic enclosures already used by Q.

Finally R is reached through the displayed route, and H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the excluded host and quotient objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 4003.3 (Regular Cauchy enclosure-filter real seal route)**.** *An accepted* $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ *carrier may expose its* $\mathsf{Real}^{\uparrow}$ *seal row R only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *enclosure ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row F, and the compatibility row B have been displayed through H and C. Thus a Real or Completion consumer receives a finite enclosure-filter route for the regular Cauchy name, not a quotient stream, a selected limit, quotient filter equality, or ambient completeness.*

Proof. Start with the carrier projection. The Real-seal row R is a coordinate of E, but it is downstream of W, D, Q, F, B in the displayed route.

The rows W, D, Q bind the regular Cauchy readback to finite windows and dyadic enclosures. The rows F, B then bind the same enclosure data to finite Cauchy-filter basis requests, so the filter-facing read is not independent of the regular rational readback.

Transport through H and replay through C preserve this ordering, while P and N record provenance and local naming for the same packet. No carrier coordinate can supply quotient equality, a chosen limit, countable choice, host real equality, or ambient completeness; the Real seal therefore exports only the displayed finite enclosure-filter route. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ read used by a Real or Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ enclosure

ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{CauchyFilter}^{\uparrow}$ basis row, one compatibility row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: streamname, dyadicratcore, regseqrat, cauchyfilter, real: $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ is not bijective to any listed sibling because it binds regular Cauchy readback and Cauchy-filter basis membership through the same displayed dyadic enclosure ledger before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}$ packet is generated from a finite StreamName window, a DyadicRatCore enclosure ledger, a RegSeqRat readback row, a CauchyFilter basis row, an enclosure-basis compatibility row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$, seed)

Scope closed: Definition 4003.1, Definition 4003.2, and Definition 4003.3 seed the finite enclosure-filter route over StreamName, DyadicRatCore, RegSeqRat, CauchyFilter, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyEnclosureFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, quotient filter equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic enclosure exactness, regular readback ordering, Cauchy-filter basis compatibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyEnclosureFilterUp.

Chapter 4004

A Concrete Naming Certificate for $\mathsf{MorreySpace}^{\uparrow}$

Following Morrey and the Sobolev-space treatment of Adams–Fournier, the $\mathsf{MorreySpace}^{\uparrow}$ packet records the local oscillation and radius-scaled integral budget used by constructive regularity consumers before any classical completion of measure or function spaces is available. It sits next to the weak-gradient packet of chapter 440, reads metric ball windows through chapter 57, measures oscillation and scale through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and faces the measure and integral rows of chapter 92 and chapter 123. The certificate names only finite BEDC histories: metric windows, local real-valued budgets, radius ledgers, transport, replay, provenance, and local naming.

Definition 4004.1 (MorreySpace carrier)**.** *A* $\mathsf{MorreySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, B, F, I, R, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the distance classifier used by local ball consumers.*
2. *B is the finite metric-ball window ledger, listing the inspected center histories, radius histories, and ball-membership witnesses.*
3. *F is the function-observation row, read through the same* $\mathsf{Real}^{\uparrow}$ *value seal that Sobolev and integral consumers use.*
4. *I is the local* $\mathsf{Integral}^{\uparrow}$ *or* $\mathsf{Measure}^{\uparrow}$*-facing budget row, recording the finite real-valued oscillation or integral magnitude attached to each inspected ball.*
5. *R is the radius-scale ledger, comparing the displayed radius with the exponent and dimension budget through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{Rat}^{\uparrow}$ *readback rows.*
6. *T is the transport row that keeps the ball window, function observation, integral budget, and radius-scale ledger attached under the carrier classifier.*
7. *H records componentwise* hsame *transport for the metric, ball-window, function, local-budget, scale-ledger, transport, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a Morrey-space request through metric-ball inspection, local budget readback, radius scaling, and exported consumer access.*
9. *P is the* Pkg *provenance over* $\mathsf{MorreySpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a completed measure space, almost-everywhere quotient equality, Hahn–Banach separation, selected representatives, or full PDE regularity.

Theorem 4004.2 (MorreySpace NameCert obligations)**.** *For every accepted* $\mathsf{MorreySpace}^\uparrow$ *carrier* $M = (X, B, F, I, R, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$ *surface consists exactly of metric-source admission through* X*, metric-ball window exposure through* B*, Real-valued function observation through* F*, local oscillation or integral-budget exposure through* I*, radius-scale ledger comparison through* R*, classifier transport through* T *and* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.MorreySpaceUp.MorreySpaceTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4004.1. The tuple displays the metric source, finite ball windows, Real-valued function observations, local budget row, radius-scale ledger, transport, replay, provenance, and local naming.

The analytic-facing part of the packet is finite. The metric row X and ball-window row B determine which local windows may be inspected, while F and I attach the Real-valued oscillation or integral budget to those same windows.

The scale row R records the radius comparison available to a Morrey consumer. The transport rows T, H and replay row C preserve this same route under classifier comparison and downstream access, and P, N source and name the route.

Since no remaining coordinate stores measure completion, quotient function equality, Hahn–Banach separation, a selected representative, or a PDE regularity theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 4004.3 (MorreySpace Sobolev-neighbour handoff)**.** *Every Sobolev-facing or regularity-facing consumer of an accepted* $\mathsf{MorreySpace}^\uparrow$ *carrier reads the local control route through*

$$\mathsf{Met}^\uparrow,\ \mathsf{Real}^\uparrow,\ \mathsf{Measure}^\uparrow,\ \mathsf{Integral}^\uparrow,\ \mathsf{Sobolev}^\uparrow \longmapsto \mathsf{MorreySpace}^\uparrow.$$

The route first displays metric-ball windows, then Real-valued local oscillation or integral-budget rows, then the radius-scale ledger, and only then exports the Morrey-space naming row. It supplies constructive analysis neighbour data for Sobolev and Real consumers, without asserting classical measure completion, Hahn–Banach separation, selected representatives, or full PDE regularity.

Proof. Use Definition 4004.2. It orders the metric source, ball-window ledger, function observation, local budget, radius-scale comparison, transport, replay, provenance, and local naming inside one accepted carrier.

The $\mathsf{Sobolev}^\uparrow$ neighbour sees only the displayed finite budget. Its weak-gradient or function-space request can consume the same $\mathsf{Real}^\uparrow$ and $\mathsf{Integral}^\uparrow$ rows, but it cannot introduce a completed function quotient or an external measure completion because those rows are absent from the carrier.

The radius-scale ledger then supplies the Morrey-facing distinction: local budgets are read together with ball radii before any exported consumer can use the packet. Thus the handoff is a finite BEDC NameCert route, not a classical regularity theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MorreySpace}^\uparrow}$**.**

Constructive story: A $\mathsf{MorreySpace}^\uparrow$ packet is generated from MetricSpaceUp ball windows, RealUp-valued local function observations, MeasureUp and IntegralUp budget rows, radius-scale ledgers, classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MorreySpace}^{\uparrow}}$, seed)

Scope closed: Definition 4004.1, Definition 4004.2, and Definition 4004.3 seed the metric-ball, local-budget, and radius-scale carrier consumed by Sobolev-neighbour constructive analysis chapters.

Formal verification: FormalStatus(NameCert$_{\mathsf{MorreySpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close classical measure completion, almost-everywhere quotient equality, Hahn–Banach separation, selected representatives, full PDE regularity, Lean verification, or bridge checking for MorreySpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, metric-ball exposure, local RealUp budget exactness, radius-scale ledger stability, Sobolev-neighbour handoff, transport, replay, provenance, and local naming for BEDC.Derived.MorreySpaceUp.

Chapter 4005

A Concrete Naming Certificate for $\mathsf{MorreyInequality}^{\uparrow}$

Following Morrey's classical regularity theorem, the $\mathsf{MorreyInequality}^{\uparrow}$ packet records the finite certificate by which Sobolev control above the dimension threshold is read as a displayed Hölder modulus rather than as an ambient embedding theorem. It depends on the Sobolev carrier of chapter 5911, the embedding handoff of chapter 2217, the local radius and oscillation control of chapter 2329, and the Hölder-target surface of chapter 4004. The packet is a finite NameCert route from weak derivative data and local averages to a concrete regularity modulus; it is not a host compactness theorem, not a choice of representative outside the displayed carrier, and not a global smoothness principle.

Definition 4005.1 (Morrey inequality carrier)**.** *A* $\mathsf{MorreyInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, E, P, Y, R, O, A, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SobolevSpace}^{\uparrow}$ *source row, including the named domain, integrability exponent, weak derivative budget, and admitted Sobolev representative.*
2. *E is the* $\mathsf{SobolevEmbedding}^{\uparrow}$ *threshold row recording the dimension/exponent comparison that permits a pointwise regularity read.*
3. *P is the* $\mathsf{PoincareInequality}^{\uparrow}$ *local handoff row, containing the radius ledger, average comparison, and oscillation estimate consumed before the modulus is formed.*
4. *Y is the* $\mathsf{MorreySpace}^{\uparrow}$ *target row, naming the Hölder seminorm classifier and the destination regularity surface.*
5. *R is the finite radius ledger, with each accepted radius tied to the same source domain and exponent data in S, E, P.*
6. *O is the oscillation ledger, reading local average differences and derivative budgets through P before any pointwise claim is exported.*
7. *A is the Hölder modulus row, recording the displayed exponent and constant produced from E, R, O.*
8. *H records componentwise* hsame *transport for the Sobolev source, embedding threshold, Poincare handoff, Morrey target, radius, oscillation, modulus, continuation, provenance, and local NameCert rows.*

9. *C records displayed* Cont *replay from a regularity consumer through S, E, P, Y, R, O, A, H.*
10. *Q is the* Pkg *provenance over* $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{SobolevEmbedding}^{\uparrow}$, $\mathsf{PoincareInequality}^{\uparrow}$, $\mathsf{MorreySpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MorreyInequality}^{\uparrow}}$ *row.*

The carrier has no coordinate for a representative selected by quotient choice, an external compact embedding theorem, a smooth approximation oracle, or an unbounded-domain regularity conclusion.

Theorem 4005.2 (Morrey inequality NameCert obligations)**.** *For every accepted* $\mathsf{MorreyInequality}^{\uparrow}$ *carrier $M = (S, E, P, Y, R, O, A, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MorreyInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for those rows, threshold visibility through E, local Poincare replay through P, R, O, Hölder-target visibility through Y, A, componentwise transport through H, continuation replay through C, provenance through Q, and local naming through N. The certificate exports no regularity conclusion unless the Sobolev source, embedding threshold, Poincare handoff, and Morrey target rows are all displayed.*

Proof. Project the carrier of Definition 4005.1. The only source of analytic data is the Sobolev row S together with the threshold row E, the Poincare handoff row P, and the Morrey target row Y.

The local estimate is not read directly from S. The radius ledger R and oscillation ledger O force the consumer through the same local averages and derivative budgets that the Poincare certificate supplies in chapter 2329.

The Hölder conclusion is then packaged only through A, H, C, Q, N. Since representative choice, compact embedding, smooth approximation, and unbounded-domain regularity have no carrier row, the listed rows exhaust the local NameCert obligations. □

Theorem 4005.3 (Morrey inequality Sobolev modulus)**.** *Let $M = (S, E, P, Y, R, O, A, H, C, Q, N)$ be an accepted* $\mathsf{MorreyInequality}^{\uparrow}$ *packet. Any exported Hölder modulus factors through*

$$\mathsf{SobolevSpace}^{\uparrow} \longmapsto \mathsf{SobolevEmbedding}^{\uparrow} \longmapsto \mathsf{PoincareInequality}^{\uparrow} \longmapsto \mathsf{MorreySpace}^{\uparrow}.$$

Concretely, the Sobolev source row of chapter 5911 supplies the derivative budget, the embedding threshold of chapter 2217 supplies the exponent comparison, the Poincare handoff of chapter 2329 supplies the finite oscillation control, and the Morrey target of chapter 4004 receives the displayed Hölder exponent and constant.

Proof. Begin with a regularity consumer asking for a modulus. Carrier admission exposes S and E before any target row is available, so the derivative budget and dimension/exponent comparison are fixed at the source.

Replay then passes through P, R, O. These rows turn local average comparisons and radius budgets into oscillation estimates; the packet has no direct edge from Sobolev membership to a pointwise conclusion.

Finally Y and A receive the Hölder exponent and constant, while H, C, Q, N preserve the same dependency order under transport and packaging. The factorization therefore records a concrete BEDC route rather than an ambient analytic theorem. □

Theorem 4005.4 (Morrey inequality Poincare handoff)**.** *For an accepted* $\mathsf{MorreyInequality}^{\uparrow}$ *packet $M = (S, E, P, Y, R, O, A, H, C, Q, N)$, the Poincare row P is the unique certificate boundary at which local derivative control is converted into finite oscillation data. If P, R, or O is absent, the packet can still display Sobolev and embedding rows, but it cannot export a Morrey-space modulus row A.*

Proof. Read the source and threshold rows first. The Sobolev and embedding data name the admissible derivative budget and exponent comparison, but neither row stores local radius averages or oscillation witnesses.

The conversion happens at P, R, O. The Poincare certificate supplies the local estimate, R names the radii at which it is inspected, and O records the oscillation comparisons consumed by the modulus row.

If any of these rows is missing, continuation replay stops before A. The structural rows can still name the partial Sobolev/embedding surface, but the Morrey modulus is not exported because the local oscillation boundary has not been displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{MorreyInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{MorreyInequality}^{\uparrow}$ packet is generated from Sobolev source rows, embedding threshold rows, Poincare local oscillation rows, Morrey target rows, radius and modulus ledgers, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MorreyInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4005.1, Definition 4005.2, Definition 4005.3, and Definition 4005.4 seed the finite route from Sobolev threshold data and Poincare oscillation control to a Morrey-space modulus.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MorreyInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close compact embedding, quotient representative choice, smooth approximation, global regularity on arbitrary domains, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold visibility, local Poincare replay, Hölder modulus packaging, classifier transport, and non-escape for the same packet.

Chapter 4006

A Concrete Naming Certificate for $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$

Following the Heine–Cantor route of chapter 9237, the $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ packet records the finite radius-choice surface that turns compact finite-net data and pointwise-continuity radii into one tolerance-only uniform-modulus row. It sits between the compact metric finite-net surface of chapter 129, the continuous-map pointwise modulus row of chapter 130, the rational lower-bound fold of chapter 619, and the uniform-modulus consumer of chapter 369. The packet is a finite ProbeBundle choice ledger, not an open-cover compactness theorem or a choice-selected modulus.

Definition 4006.1 (Compact uniform radius-choice carrier)**.** *A* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, M, Q, F, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying the* ProbeBundle *of centers and coverage witnesses for the requested tolerance scale.*
2. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to those center histories.*
3. *Q is the positive radius-choice row, listing the local radius history selected by the displayed pointwise-modulus and finite-net data for each bundled probe.*
4. *F is the finite lower-bound fold row over* ProbeBundle*, read through the rational order and returning one positive radius history below every displayed local radius.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row, consuming only the tolerance history and the folded radius history.*
6. *H records componentwise* hsame *transport for compact-net, pointwise-modulus, radius-choice, finite-fold, output, replay, provenance, and local-name rows.*
7. *C records displayed* Cont *replay from tolerance input through finite-net inspection, local radius readback, lower-bound folding, and uniform-modulus export.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* ProbeBundle*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, a host Lebesgue-number minimum, a choice-selected modulus, sequential compactness, quotient real equality, or an ambient continuity theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformRadiusChoiceUp.CompactUniformRadiusChoiceTasteGate single carrier alignment`

Theorem 4006.2 (Compact uniform radius-choice NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *carrier* $R = (K, M, Q, F, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through* K*, pointwise-modulus admission through* M*, local positive-radius exposure through* Q*, finite ProbeBundle lower-bound folding through* F*, tolerance-only* $\mathsf{UniformModulus}^{\uparrow}$ *output through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4006.1. The rows $K, M, Q, F, U, H, C, P, N$ are the complete displayed inventory.

The compact finite-net and pointwise-modulus rows determine the only local radii that may be inspected. The row Q records those radii as positive BEDC histories attached to bundled probes, not as host choices.

The fold row F then recurses over the displayed ProbeBundle and returns a positive lower-bound history below every radius named by Q. Since F is a finite rational-order fold, the exported radius is classifier-stable without invoking a global minimum operator.

Finally U, H, C, P, N export, transport, replay, source, and name exactly this radius-choice route. The omitted open-cover, sequential, quotient, and choice-selected objects are absent from the carrier, so the listed rows exhaust the local NameCert obligations. □

Theorem 4006.3 (Tolerance-only uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ *packet receives one folded radius history as a function of the requested tolerance history and the displayed finite compact-continuity packet:*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The consumer cannot read an open-cover compactness proof, a choice-selected family of local moduli, a host minimum over radii, quotient real equality, or an external modulus function.

Proof. Use Definition 4006.2. The obligation surface orders compact-net admission, pointwise-modulus admission, local radius exposure, finite lower-bound folding, and uniform-modulus export.

For a fixed tolerance history, the compact row supplies only a finite ProbeBundle of center histories. The pointwise row supplies the local radii attached to those centers, and the radius-choice row records them as displayed positive histories.

The lower-bound fold produces the single radius consumed by U. Because all preceding rows are tied by H and replayed by C, the exported uniform-modulus read depends only on the displayed tolerance and finite packet, not on any external selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, local positive radius-choice rows, a finite ProbeBundle rational lower-bound fold, a UniformModulusUp output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4006.1, Definition 4006.2, and Definition 4006.3 seed the finite radius-choice ledger consumed by compact-uniform-continuity and uniform-modulus chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusChoice}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected moduli, host Lebesgue-number minima, sequential compactness, quotient real equality, Lean verification, or bridge checking for $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$.

Upgrade path: Obligation closure requires checked compact-net admission, pointwise-modulus admission, local radius-choice exactness, finite lower-bound fold stability, UniformModulusUp output exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformRadiusChoiceUp.

Chapter 4007

A Concrete Naming Certificate for $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style compact-uniform language, the $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ packet records the finite tail-index surface by which an eventually stable family of continuous maps receives a shared compact-source modulus. It sits after the compact radius-choice packet of chapter 4006, the uniform family-modulus route of chapter 3844, the equicontinuity family ledger of chapter 413, the compact source of chapter 129, and the graph rows of chapter 130. Its terminal reads pass through finite $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{Real}^{\uparrow}$ seal rather than through a host function-space topology, a selected subsequence, or an ambient completeness theorem.

Definition 4007.1 (Asymptotic equicontinuity carrier)**.** *An* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, E, U, T, S, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact finite-net carrier and the metric-distance ledger used by all later rows.*
2. *F is the finite family ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows whose graph and pointwise-modulus data are inspected at displayed compact-source centers.*
3. *E is the* $\mathsf{EquicontinuityFamily}^{\uparrow}$ *row, binding the family rows to a shared local tolerance-radius ledger before any tail condition is read.*
4. *U is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row, exporting the compact finite-net fold that yields a single tolerance-indexed source radius for the displayed family window.*
5. *T is the asymptotic tail row. It records the finite index threshold after which the family window must use the same compact-source radius and target tolerance route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened after T, used to inspect only the tail-admitted source and target observations.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by the tail-admitted distance and tolerance reads after K, F, E, U, T, S have been replayed.*
8. *H records componentwise* hsame *transport for compact source, continuous-family, equicontinuity, uniform-modulus, tail, stream-window, real-seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a tail-index request through the compact finite-net and uniform-equicontinuity rows to the real-seal boundary.*

10. *P is the* Pkg *provenance over* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$, $\mathsf{CompactUniformRadiusChoice}^{\uparrow}$, $\mathsf{UniformEquicontinuityM}$ $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for host function-space compactness, a selected subsequence, quotient equality of maps, countable choice over tails, an external modulus function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AsymptoticEquicontinuityUp.AsymptoticEquicontinuityTasteGate single carrier alignment`

Theorem 4007.2 (Asymptotic equicontinuity NameCert obligations)**.** *For every accepted* AsymptoticEquicontinuity *carrier $A = (K, F, E, U, T, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$ *surface consists exactly of compact-source admission through K, continuous-family admission through F, family equicontinuity admission through E, shared uniform-modulus export through U, asymptotic tail admission through T, finite stream-window inspection through S, terminal real-seal routing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 4007.1. The rows K, F, E, U, T, S, R are the complete analytic inventory, and H, C, P, N are the structural rows that transport, replay, source, and name that inventory.

The compact and family rows are read before the tail row. The compact source K supplies only finite-net and metric-distance data, while F and E expose the continuous-family and shared equicontinuity ledger. The row U then records the uniform family modulus already folded through compact finite-net data.

The asymptotic step is exactly the row T. Once T has displayed the finite threshold, the stream row S may inspect only tail-admitted windows, and the real seal R may be reached only through those windows.

Finally the carrier has no coordinate for a selected subsequence, host function-space topology, quotient map equality, external modulus function, countable tail choice, or ambient Real completeness. The listed rows therefore exhaust the local NameCert obligations. □

Theorem 4007.3 (Asymptotic equicontinuity modulus handoff)**.** *Every compact-window or Real-facing consumer of an accepted* $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ *carrier receives its tail-stable modulus through the displayed route*

$\mathsf{CompactUniformRadiusChoice}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, Continuou

The handoff supplies one finite tail threshold and one compact-source modulus route, not a selected subsequence, host function-space compactness proof, quotient equality of functions, countable-choice tail, or ambient Real-completeness witness.

Proof. Start with the compact-source and family evidence. The rows K, F, E display the compact metric source, the continuous-map family window, and the equicontinuity-family ledger. The compact radius-choice and uniform-equicontinuity rows are then read through U, so the source radius is already a finite compact-net handoff.

Next expose the tail row T. It records the threshold after which the same compact-source modulus route must serve the accepted family window. The stream row S opens only the finite windows admitted by this threshold, and R seals the resulting distance and tolerance reads as $\mathsf{Real}^{\uparrow}$ data.

The structural rows do not add analytic content. They preserve the displayed route componentwise, replay it as a continuation, record its provenance, and name it locally. Hence every consumer receives the same finite modulus handoff, with no hidden selector or external compactness principle. □

Falsifiable prediction: An accepted $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ read is refuted if it reaches a Real-facing tail claim without first exposing the compact source, continuous-family ledger, equicontinuity family row, uniform-equicontinuity modulus row, asymptotic tail threshold, StreamName windows, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: compact-uniform-radius-choice, uniform-equicontinuity-modulus, equicontinuityfamily, compactmetric, continuousmap, streamname, real: none alone is bijective with $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ because the packet binds compact finite-net modulus data to an eventual tail threshold before any Real-facing consumer can read the route.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$ packet is generated from CompactUniformRadiusChoice, UniformEquicontinuityModulus, EquicontinuityFamily, CompactMetric, ContinuousMap, StreamName, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4007.1, Definition 4007.2, and Definition 4007.3 seed the finite compact-source tail-modulus route for asymptotic equicontinuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticEquicontinuity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.AsymptoticEquicontinuityUp.AsymptoticEquicontinuityTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Origin: human

Not claimed: This chapter does not close Arzela–Ascoli compactness, host function-space topology, selected subsequences, quotient equality of maps, countable tail choice, Lean verification, or bridge checking for $\mathsf{AsymptoticEquicontinuity}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, compact-source admission, family equicontinuity admission, uniform-modulus handoff, tail-threshold exactness, StreamName window restriction, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.AsymptoticEquicontinuityUp.

Chapter 4008

A Concrete Naming Certificate for ProkhorovTightProbability$^{\uparrow}$

Following Prokhorov's tightness theorem for probability measures, the ProkhorovTightProbability$^{\uparrow}$ packet records only the constructive finite tightness data needed before weak-convergence consumers can inspect a probability family. It starts from the probability source of chapter 184, the pushforward distribution row of chapter 186, the event-surface measure row of chapter 92, the compact-test packet of chapter 129, the finite-net witness of chapter 1767, and the located outside-mass ledger of chapter 3795. The chapter also reads the underlying real boundary of chapter 38. It is a finite tightness certificate, not a proof of weak compactness, selected subsequences, countable choice, or ambient real completeness.

Definition 4008.1 (Prokhorov tight probability carrier)**.** *A* ProkhorovTightProbability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (P, D, M, K, N, E, R, T, H, C, Q, L)$$

with the following displayed rows:

1. *P is the* ProbSpace$^{\uparrow}$ *source row carrying the probability mass surface.*
2. *D is the* Distribution$^{\uparrow}$ *pushforward row whose target-side event reads are being tested for tightness.*
3. *M is the* Measure$^{\uparrow}$ *event-surface row through which the outside event and its mass are read.*
4. *K is the* CompactMetric$^{\uparrow}$ *compact-test row naming the compact metric test surface used by the packet.*
5. *N is the* FiniteEpsilonNet$^{\uparrow}$ *row displayed inside the compact test for the requested tolerance.*
6. *E is the outside-mass error-bound row read through* RealTightness$^{\uparrow}$ *and the underlying* Real$^{\uparrow}$ *comparison surface.*
7. *R is the finite radius or tolerance ledger that aligns the compact test, finite net, and outside-mass request.*
8. *T records transport from the probability source through distribution, measure, compact-test, finite-net, and real-tightness rows.*
9. *H records componentwise* hsame *transport for every displayed row.*

10. *C records displayed* Cont *replay of source admission, pushforward readback, event-surface inspection, compact-test exposure, finite-net inspection, outside-mass comparison, and radius-ledger readback.*
11. *Q is the* Pkg *provenance over* $\mathsf{ProkhorovTightProbability}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{RealTightness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ProkhorovTightProbabilityUp.ProkhorovTightProbabilityUp`

Theorem 4008.2 (Prokhorov tight probability NameCert obligations)**.** *For every accepted* ProkhorovTightProbabili *carrier $Z = (P, D, M, K, N, E, R, T, H, C, Q, L)$, the local* $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$ *surface consists exactly of five obligation groups:*

1. *carrier admission for the probability source, pushforward distribution, event-surface measure, compact test, finite net, outside-mass bound, radius ledger, transport, replay, provenance, and local naming rows;*
2. *compact-test exposure, requiring every tightness read to display the* $\mathsf{CompactMetric}^{\uparrow}$ *row K and the internal* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row N before the outside event is consumed;*
3. *outside-mass bound exactness, requiring the error row E to be read through* $\mathsf{RealTightness}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *rather than through a host real inequality;*
4. *distribution/measure compatibility, requiring the pushforward row D and event-surface row M to name the same measured outside event under the displayed transport T;*
5. *non-escape from open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, and ambient real completeness.*

No additional NameCert obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4008.1. The tuple lists exactly the probability source, distribution pushforward, event-surface measure, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming rows.

The compact side is finite. A tightness read must expose K and the internal finite-net row N before it can identify the outside event measured by M.

The outside-mass side is real-name data. The error row E is read through the located tightness and real comparison surfaces, while D, M, T keep the distribution and measure event rows compatible.

Since no carrier coordinate stores open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient real completeness, the five displayed groups exhaust the local NameCert obligations. □

Theorem 4008.3 (Prokhorov tight probability finite-net handoff)**.** *Every accepted tightness read from a* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ProbSpace}^{\uparrow} \longmapsto \mathsf{Distribution}^{\uparrow} \longmapsto \mathsf{Measure}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{RealTightness}^{\uparrow} \longmapsto \mathsf{Real}$$

before any weak-convergence consumer can inspect it. The exported read is the displayed probability, distribution, measure, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming route; it exposes no weak compactness theorem, selected subsequence, countable-choice selector, host measure, quotient event, or ambient completeness row.

Proof. Start with the NameCert obligations of Definition 4008.2. Carrier admission places the probability source and pushforward distribution before the measured outside event.

Next read the compact finite test. The measure row is inspected only after the compact metric row and its finite-net witness are displayed for the requested tolerance ledger.

Finally the outside-mass bound is consumed through $\mathsf{RealTightness}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. The structural rows transport and replay this same finite route, so a weak-convergence consumer receives only the displayed handoff and cannot insert an unlisted compactness, sequence-selection, quotient, host-measure, or completeness coordinate. □

Theorem 4008.4 (Prokhorov tight probability Portmanteau readiness)**.** *Every Portmanteau-facing weak-convergence test read from an accepted* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *carrier receives only the displayed probability source, pushforward distribution, measure event surface, compact metric finite-net test,* $\mathsf{RealTightness}^{\uparrow}$ *outside-mass row,* $\mathsf{Real}^{\uparrow}$ *comparison, transport, replay, provenance, and local* NameCert *rows before bounded-continuous probes may inspect it. The readiness route exports no weak compactness, selected subsequence, countable choice, host measure, quotient event, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4008.2. The local NameCert obligations expose exactly the probability source, distribution pushforward, measure event surface, compact-test, finite-net, outside-mass, radius-ledger, transport, replay, provenance, and local naming rows.

Next apply the finite-net handoff of Definition 4008.3. It orders the compact metric finite-net test before the outside-mass row and routes the comparison through $\mathsf{RealTightness}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ before any weak-convergence consumer can inspect the packet.

Finally pass the resulting finite surface to the Portmanteau consumer of chapter 4193. Because the carrier contains no coordinate for weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient real completeness, bounded-continuous probes receive only the displayed readiness route. □

Falsifiable prediction: Every accepted $\mathsf{ProkhorovTightProbability}^{\uparrow}$ tightness read factors through $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{RealTightness}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any weak-convergence consumer can inspect it.

Independence witness: probspace, distribution, measure, compactmetric, finite-epsilon-net, real-tightness, real: the packet is not bijective to any listed sibling because it binds the probability source, pushforward event surface, compact finite-net test, outside-mass error row, radius ledger, transport, replay, provenance, and local NameCert row as one tight probability surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProkhorovTightProbability}^{\uparrow}$ packet is generated from ProbSpaceUp source rows, DistributionUp pushforward rows, MeasureUp event-surface rows, CompactMetricUp compact-test rows, FiniteEpsilonNetUp finite-net rows, RealTightnessUp outside-mass rows over RealUp, a finite tolerance ledger, transport, hsame, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$, seed)

Scope closed: Definition 4008.1, Definition 4008.2, and Definition 4008.3 seed the finite tight-probability handoff for probability compactness consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProkhorovTightProbability}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, weak compactness, selected subsequences, countable choice, host measures, quotient events, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-test exposure, outside-mass exactness, distribution/measure compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ProkhorovTightProbabilityU

Chapter 4009

A Concrete Naming Certificate for $\mathsf{CauchyTailEstimate}^{\uparrow}$

Following Bishop and Bridges' constructive analysis of Cauchy sequences and finite tail bounds, the $\mathsf{CauchyTailEstimate}^{\uparrow}$ packet records the displayed route by which a requested tail index and dyadic radius become a bounded regular-rational read before any $\mathsf{Real}^{\uparrow}$ seal is consumed. It sits after the tail-filter surface of chapter 3850 and the fast criterion of chapter 3825, and it reads its tolerance, regular readback, and endpoint seal through chapter 291, chapter 301, and chapter 38. The packet is a finite certificate for Real and Completion consumers; it is not a quotient sequence, a selected limit, countable choice, host real equality, trichotomy, or ambient real completeness.

Definition 4009.1 (Cauchy tail estimate carrier)**.** *A* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the tail-index row, naming the finite threshold after which the estimate may be inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, naming the dyadic tolerance used by the tail-bound comparison.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed tail window and radius data.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after T, D, R have been replayed.*
5. *B is the tail-bound ledger, recording the finite comparison witnesses that keep every accepted read beyond T inside the radius named by D.*
6. *H records componentwise* hsame *transport for T, D, R, E, B and for the continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through T, D, B, R, and then E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{FastCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed tail threshold, dyadic radius, regular readback, real seal, tail-bound ledger, continuation, provenance, and local naming

rows to a common finite surface. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, trichotomy, open-cover compactness, or ambient completeness.

Theorem 4009.2 (Cauchy tail estimate NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier* $Q = (T, D, R, E, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed tail index, dyadic radius, regular readback, real seal, tail-bound ledger, transport, continuation, provenance, and local name;*
2. *classifier transport to a common* $\mathsf{CauchySequenceTailFilter}^{\uparrow}$, $\mathsf{FastCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *finite surface;*
3. *stability of* T, D, R, E, B *under the componentwise* hsame *rows recorded in* H*;*
4. *handoff exactness requiring every real-facing read to pass through* T, D, B, R, E *in that order;*
5. *refusal rows excluding quotient streams, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, and ambient completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyTailEstimateUp.CauchyTailEstimateUpTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4009.1. The projection exposes exactly the tail-index row, dyadic radius row, regular readback row, real seal row, tail-bound ledger, transport row, continuation row, provenance row, and local naming row.

Classifier transport is componentwise. The rows in H move the mathematical coordinates T, D, R, E, B and the administrative coordinates C, P, N to the same finite comparison surface.

Handoff exactness is read from the displayed continuation row. A tail request first reaches the threshold row T, then the dyadic radius row D, then the ledger row B that witnesses the tail-bound comparison, then the regular readback R, and only then the real seal E.

The refusal rows are exhausted by the carrier inventory. Since the packet has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, open-cover compactness, or completeness, no local NameCert obligation can export those data. □

Theorem 4009.3 (Cauchy tail estimate RegSeqRat handoff)**.** *For every accepted* $\mathsf{CauchyTailEstimate}^{\uparrow}$ *carrier* $Q = (T, D, R, E, B, H, C, P, N)$*, every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of* R *factors through the tail-index row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row* D*, and the tail-bound ledger* B *before it can expose the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The handoff route is therefore finite and does not identify Cauchy names by quotient equality, select a limit, invoke countable choice, or appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a regular-rational consumer of R. Carrier admission lists R only as the readback row reached from the displayed tail threshold and dyadic radius.

Replay the continuation row C. Its route sends the request first to T, checks the requested radius through D, reads the comparison witnesses in B, and only then exposes the regular rational row R.

Transport through H preserves this dependency order. The package row P and local name N record provenance and naming for the same finite packet; they do not add a second readback coordinate.

Thus every $\mathsf{RegSeqRat}^{\uparrow}$ handoff factors through T, D, B before R and reaches E only after that route has been displayed. The carrier contains no coordinate from which a bypass route could obtain quotient equality, selected limits, choice, host real equality, or completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailEstimate}^{\uparrow}$ packet is generated from a finite tail-index row, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal row, a tail-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4009.1, Definition 4009.2, and Definition 4009.3 bind the finite tail-bound carrier, local NameCert obligations, and RegSeqRat handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailEstimate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient streams, selected limits, countable choice, host real equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, stability, handoff exactness, refusal rows, and TasteGate nontriviality with field faithfulness for $T, D, R, E, B, H, C, P, N$.

Chapter 4010

A Concrete Naming Certificate for $\mathsf{Circle}^{\uparrow}$

Following Hatcher and Munkres on the circle as a compact one-dimensional boundary object, the $\mathsf{Circle}^{\uparrow}$ packet records the topological boundary reading of a one-dimensional closed loop without identifying it with the algebraic $\mathsf{S}^{1\uparrow}$ group certificate. It sits beside $\mathsf{S}^{1\uparrow}$ of chapter 58, $\mathsf{Real}^{\uparrow}$ of chapter 38, $\mathsf{Met}^{\uparrow}$ of chapter 57, and $\mathsf{Compact}^{\uparrow}$ of chapter 60: those siblings supply phase multiplication, real-coordinate readback, metric inspection, and compact covering rows, while this packet binds the finite boundary carrier and its NameCert obligations as a topological circle surface.

Definition 4010.1 (Circle carrier)**.** *A* $\mathsf{Circle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (B, R, M, K, S, H, P, N)$$

with the following displayed rows:

1. *B is the boundary-loop row, naming the closed one-dimensional boundary surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *coordinate-read row, recording the finite real-coordinate inspection used by local charts or parametrized arcs.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *row, recording the metric neighborhoods through which boundary adjacency is read.*
4. *K is the* $\mathsf{Compact}^{\uparrow}$ *row, recording finite cover data for the closed boundary surface.*
5. *S is the* $\mathsf{S}^{1\uparrow}$ *handoff row, available only when a consumer asks for the phase-group reading of the same boundary.*
6. *H records componentwise* hsame *transport for the boundary, real-coordinate, metric, compactness, handoff, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{Circle}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, BHist, hsame, Pkg, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient plane, a selected trigonometric parametrization, a quotient by angle equality, or a global orientation choice.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier boundary route certificate`

Theorem 4010.2 (Circle NameCert obligations)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *carrier*

$$C = (B, R, M, K, S, H, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *surface consists exactly of carrier admission for the boundary-loop row, real-coordinate readback, metric-neighborhood row, compact-cover row, optional* $\mathsf{S}^{1\uparrow}$ *handoff, componentwise transport, package provenance, and local naming. Any topological-circle consumer must read the closed boundary surface through* B, R, M, K, H, P, N *before using the algebraic phase handoff* S*.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleUpTasteGate single carrier alignment`

Proof. Project the displayed carrier of Definition 4010.1. The topological rows are B, R, M, K: they name the closed boundary, the real-coordinate readback, the metric neighborhoods, and the compact cover data.

The handoff row S is downstream. It allows a consumer to compare the boundary packet with the $\mathsf{S}^{1\uparrow}$ phase-group sibling, but it does not replace the boundary carrier by an algebraic group row.

The rows H, P, N transport, source, and name the same finite packet. Since no ambient plane, trigonometric parametrization, quotient-angle equality, or orientation coordinate appears in the carrier, the listed rows exhaust the $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ obligations. □

Theorem 4010.3 (Circle and $\mathsf{S}^{1\uparrow}$ boundary handoff)**.** *Every* $\mathsf{S}^{1\uparrow}$*-facing read of an accepted* $\mathsf{Circle}^{\uparrow}$ *carrier factors through the boundary-loop row* B*, real-coordinate readback* R*, metric row* M*, compact-cover row* K*, and handoff row* S*. The handoff may expose a phase-group consumer, but it cannot make* $\mathsf{S}^{1\uparrow}$ *the carrier of topological compact-boundary data and cannot export quotient-angle equality or a selected global parametrization.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleSOneBoundaryHandoff`

Proof. Start with a consumer asking for the phase-group reading. The $\mathsf{Circle}^{\uparrow}$ carrier first exposes the boundary surface through B and its local real and metric inspections through R, M.

Compactness is read through K, so the consumer receives a finite cover surface for the closed boundary before any phase handoff is available. Only then can S point to the $\mathsf{S}^{1\uparrow}$ sibling.

The componentwise transport, provenance, and naming rows H, P, N preserve this order. Because the carrier has no quotient-angle coordinate or selected global parametrization, the handoff remains a boundary-to-phase route rather than an identification of the two packets. □

Theorem 4010.4 (Circle metric subspace carrier obligation)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *carrier* $C = (B, R, M, K, S, H, P, N)$*, the boundary row* B*,* $\mathsf{Real}^{\uparrow}$ *coordinate row* R*,* $\mathsf{Met}^{\uparrow}$ *neighborhood row* M*,* $\mathsf{Compact}^{\uparrow}$ *cover row* K*,* $\mathsf{S}^{1\uparrow}$ *handoff row* S*, transport row* H*, provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *row* N *are the full carrier obligation surface. A consumer may not replace these rows by an ambient plane, a selected parametrization, quotient-angle equality, or a global orientation choice.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleMetricSubspaceCarrierObligation`

Proof. Project the carrier of Definition 4010.1. Its analytic boundary surface is exactly B, R, M, K, S, and its structural surface is exactly H, P, N.

No other coordinate is available for a circle consumer. Since the carrier omits an ambient plane, selected trigonometric map, quotient by angle equality, and orientation row, the displayed coordinates exhaust the carrier obligation surface. □

Theorem 4010.5 (Circle $\mathsf{S}^{1\uparrow}$ real seal route)**.** *Every phase-group read of an accepted* $\mathsf{Circle}^{\uparrow}$ *carrier must pass through the* $\mathsf{Real}^{\uparrow}$ *coordinate row* R *and the* $\mathsf{S}^{1\uparrow}$ *handoff row* S*, with the boundary row* B*, metric row* M*, compact row* K*, transport* H*, provenance* P*, and local naming* N *still present. This route permits a phase interpretation but does not identify* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CircleUp.CircleSOneRealSealRoute`

Proof. Start with a consumer that asks for phase-group data. The carrier first exposes the topological boundary and real-coordinate surface through B and R.

The handoff row S can then point to the $\mathsf{S}^{1\uparrow}$ sibling, while M, K, H, P, N keep the same metric, compact, transport, provenance, and naming rows attached to the circle packet. Because S is a handoff coordinate rather than the carrier itself, the route is not an identification of the two chapters. □

Theorem 4010.6 (Circle compact metric boundary)**.** *Before any consumer reads* $\mathsf{Circle}^{\uparrow}$ *as a closed boundary, the packet must expose the compact-cover row* K*, the metric-neighborhood row* M*, the boundary row* B*, the* $\mathsf{Real}^{\uparrow}$ *coordinate row* R*, transport* H*, provenance* P*, and local naming* N*. The compact-metric boundary does not supply an ambient plane, a selected parametrization, or quotient-angle equality.*

Proof. Read the compact and metric coordinates together. The row K supplies finite cover data for the closed boundary, and M supplies the neighborhood inspection by which boundary adjacency is read.

The rows B, R, H, P, N bind those compact and metric inspections to the circle carrier and its local name. Since no ambient plane or quotient-angle coordinate appears in the packet, a closed-boundary consumer receives only this compact-metric surface. □

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier compact metric boundary`

Theorem 4010.7 (Circle Real metric scope route)**.** *For every accepted* $\mathsf{Circle}^{\uparrow}$ *read of* $C = (B, R, M, K, S, H, P, N)$*, the boundary inspection is scoped by the displayed* BHist *carrier rows, componentwise* hsame *transport,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *naming, the* $\mathsf{Real}^{\uparrow}$ *coordinate row, the* $\mathsf{Met}^{\uparrow}$ *neighborhood row, the* $\mathsf{Compact}^{\uparrow}$ *finite-cover row, and the optional* $\mathsf{S}^{1\uparrow}$ *handoff row. The route exports no ambient plane, quotient-angle equality, selected parametrization, global orientation, or identification of* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$*.*

Proof. Project the carrier of Definition 4010.1. The row list exposes the boundary row B, real-coordinate inspection R, metric inspection M, compact cover K, optional $\mathsf{S}^{1\uparrow}$ handoff S, transport H, provenance P, and local naming N.

Apply Definition 4010.2. Its NameCert surface forces the topological rows B, R, M, K, H, P, N to be read before any phase handoff can be used.

Use Definition 4010.3 and Definition 4010.5. They keep S as a downstream handoff through the same boundary and $\mathsf{Real}^{\uparrow}$ rows, not as an identification with the circle carrier.

Finally apply Definition 4010.6. The compact and metric rows already exhaust the closed-boundary inspection surface together with B, R, H, P, N, so the scoped route is exactly the displayed BEDC row surface and nothing from an ambient plane, quotient angle, selected parametrization, or global orientation remains available. □

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier real metric scope route`

Theorem 4010.8 (Circle Real metric scoped closure)**.** *Every accepted* $\mathsf{Circle}^{\uparrow}$ *read is scoped exactly by the carrier rows* B, R, M, K, S, H, P, N *of Definition 4010.1, the displayed* BHist *packet, componentwise* hsame *transport,* Pkg *provenance, local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *naming, and the sibling surfaces* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, *and optional* $\mathsf{S}^{1\uparrow}$. *The scoped route exports no ambient plane, quotient-angle equality, selected parametrization, global orientation, or identification of* $\mathsf{Circle}^{\uparrow}$ *with* $\mathsf{S}^{1\uparrow}$.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier real metric scoped closure`

Proof. Apply Definition 4010.2. The accepted read begins with the carrier rows B, R, M, K, S, H, P, N and the local $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ surface, so no unlisted coordinate can enter the scope.

Use Definition 4010.3 and Definition 4010.5. These results keep the optional $\mathsf{S}^{1\uparrow}$ row as a handoff from the boundary and real-coordinate rows, not as an identification of the circle packet with the phase-group sibling.

Apply Definition 4010.6 and Definition 4010.7. The compact, metric, and real rows already determine the closed-boundary inspection surface through BHist, hsame, Pkg, and local naming.

Thus the scoped closure is exactly the displayed BEDC carrier surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, and optional $\mathsf{S}^{1\uparrow}$, while ambient planes, quotient-angle equality, selected parametrizations, global orientations, and carrier identification with $\mathsf{S}^{1\uparrow}$ remain outside the packet. □

Theorem 4010.9 (Circle public boundary export)**.** *The public* $\mathsf{Circle}^{\uparrow}$ *export is exactly the boundary-loop row* B, $\mathsf{Real}^{\uparrow}$ *coordinate row* R, $\mathsf{Met}^{\uparrow}$ *neighborhood row* M, $\mathsf{Compact}^{\uparrow}$ *cover row* K, *optional* $\mathsf{S}^{1\uparrow}$ *handoff row* S, *componentwise* hsame *transport row* H, Pkg *provenance row* P, *and local* $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$ *route* N *of Definition 4010.1. A downstream consumer may cite this boundary as the public circle surface, but it receives no ambient plane, quotient-angle equality, selected parametrization, global orientation, or carrier identification with* $\mathsf{S}^{1\uparrow}$.

Lean4 ✓ `BEDC.Derived.CircleUp.CircleCarrier public boundary export`

Proof. Start with the scoped closure of Definition 4010.8. It already identifies the visible BEDC carrier surface as B, R, M, K, S, H, P, N over the boundary, Real-coordinate, metric, compact, optional phase-handoff, transport, provenance, and local naming rows.

Now read that scoped packet as an outward-facing boundary. The NameCert obligations of Definition 4010.2 keep the topological circle read on the boundary, coordinate, metric, compact, transport, provenance, and local naming rows before the optional handoff can be used.

Finally apply the handoff and compact-metric exclusions of Definition 4010.3, Definition 4010.5, and Definition 4010.6. They keep S as an optional route to the $\mathsf{S}^{1\uparrow}$ sibling and exclude the ambient plane, quotient-angle, parametrization, orientation, and carrier-identification readings. Thus the public boundary exports exactly the displayed carrier rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$.

Constructive story: A $\mathsf{Circle}^{\uparrow}$ packet is generated from boundary-loop, Real-coordinate, metric-neighborhood, compact-cover, optional SOne handoff, componentwise hsame transport, package provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$, publicClosed)

Scope closed: Definition 4010.1, Definition 4010.2, Definition 4010.3, Definition 4010.4, Definition 4010.5, Definition 4010.6, Definition 4010.7, Definition 4010.8, and Definition 4010.9 bind the topological boundary certificate, compact-metric obligations, Real metric route, algebraic phase handoff, and public boundary export.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Circle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global parametrization, quotient-angle equality, ambient-plane data, or an identification of $\mathsf{Circle}^{\uparrow}$ with $\mathsf{S}^{1\uparrow}$.

Upgrade path: Formal closure requires checked carrier admission, boundary stability, compact-cover exactness, SOne handoff noncollapse, transport, provenance, and local naming for the same finite packet.

Chapter 4011

A Concrete Naming Certificate for $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$

The $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ packet realizes Definition 9463.12 as a finite NameCert carrier for the three-role discipline of focused disk readout, inscription-side self-center, and far-end boundary reading. It sits beside $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ of chapter 3500, $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ of chapter 1071, $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$ of chapter 3486, and $\mathsf{UnitDisk}^{\uparrow}$ of chapter 377; those siblings supply boundary-fixed transport, local inscription sealing, observer-expansion ledger data, and disk-coordinate rows, while this packet binds the self-center reading itself.

Definition 4011.1 (Hyperbolic self-center reading carrier)**.** *A* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, I, B, T, V, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk-readout row, recording the finite* $\mathsf{UnitDisk}^{\uparrow}$ *or Poincare-disk coordinate read of the focused ledger event.*
2. *I is the inscription row, recording the* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$ *reading of the local self-center.*
3. *B is the far-end boundary row, recording the boundary-side* T *reading used by the hyperbolic vision.*
4. *T is the boundary-fixed transport row, recording the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *route that preserves the boundary reading while disk data move.*
5. *V is the noncollapse verdict row, recording that D, I, and B are three roles and not three hidden objects or one collapsed object.*
6. *H records componentwise* hsame *transport for disk readout, inscription, boundary, boundary-fixed transport, noncollapse verdict, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for reading D, I, B, T, V in the same finite packet.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$, $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden subject, a global observer point, a selected geodesic center, a physical constant, or an identification of disk readout with boundary far end.

Lean4 ✓ `BEDC.Derived.HyperbolicSelfCenterReadingUp.HyperbolicSelfCenterReadingTasteGate single carrier alignment`

Theorem 4011.2 (Hyperbolic self-center reading NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *carrier*

$$R = (D, I, B, T, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$ *surface consists exactly of disk-readout admission, inscription admission, far-end boundary admission, boundary-fixed transport, noncollapse verdict, componentwise transport, displayed replay, provenance, and local naming. No obligation may identify the records-side disk row* D*, the inscription-side row* I*, and the far-end boundary row* B*.*

Proof. Project the accepted carrier of Definition 4011.1. The three geometric roles are displayed separately as D, I, B.

The transport row T moves disk-facing data while preserving the boundary reading, and the verdict row V records the refusal to collapse the three roles. These rows are part of the same finite packet and cannot be replaced by prose analogy.

Finally, H, C, P, N transport, replay, source, and name the packet. Since the carrier has no hidden subject, global observer point, selected geodesic center, physical constant, or identification coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4011.3 (Hyperbolic self-center reading noncollapse)**.** *Every accepted* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *read keeps disk readout* D*, inscription row* I*, and far-end boundary row* B *distinct under the boundary-fixed transport row* T *and the noncollapse verdict row* V*. A consumer may cite their relation only through* $R = (D, I, B, T, V, H, C, P, N)$*, and may not export a hidden subject, a single collapsed self-center object, or a physical constant from the geometry alone.*

Proof. Begin with the three displayed roles. The disk-readout row D is records-side data, the inscription row I is the local self-center seal, and the boundary row B is the far-end reading.

The transport row T relates disk and boundary readings without erasing that distinction. The verdict row V records the noncollapse condition before any downstream consumer can package the read.

The structural rows H, C, P, N preserve the same finite inventory. Because no hidden-subject, collapsed-object, or physical-constant coordinate appears in the carrier, every accepted reading remains inside the displayed packet. □

Theorem 4011.4 (Hyperbolic self-center reading provenance route)**.** *The* $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ *packet routes its focused-ledger context through* $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$*, disk coordinates through* $\mathsf{UnitDisk}^{\uparrow}$*, local inscription through* $\mathsf{LocalSelfCenterInscriptionSeal}^{\uparrow}$*, and boundary-fixed movement through* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$*. These sibling routes are consumed as finite rows of one carrier and do not supply an ambient hyperbolic model, a global observer, or a host-geometric self.*

Proof. Read the provenance row P together with the displayed rows D, I, B, T, V. It names the focused-ledger, disk-coordinate, inscription, and boundary-fixed transport sources that the carrier is allowed to consume.

The replay row C then orders those reads inside one finite packet. A consumer can replay the disk readout, local inscription, far-end boundary reading, transport, and noncollapse verdict, but it receives no external geometric model.

Finally, H and N preserve and name the same packet. The absence of a global observer or host-geometric self coordinate gives the stated boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicSelfCenterReading}^{\uparrow}$ packet is generated from disk-readout, local inscription, far-end boundary, boundary-fixed transport, noncollapse verdict, componentwise hsame transport, displayed replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4011.1, Definition 4011.2, Definition 4011.3, and Definition 4011.4 seed the finite carrier for the hyperbolic self-center reading.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicSelfCenterReading}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a global observer, hidden subject, physical constant, or ambient hyperbolic model.

Upgrade path: Obligation closure requires checked carrier admission, disk-readout stability, inscription and boundary noncollapse, boundary-fixed transport, replay, provenance, and local naming for the same finite packet.

Chapter 4012

A Concrete Naming Certificate for $\mathsf{ClosedIntervalNesting}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ClosedIntervalNesting}^{\uparrow}$ packet records the finite nested-closed-interval route used before a retained interval family is read as regular rational Cauchy data. It sits over the dyadic endpoint core of chapter 291, the finite stream-window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the interval certificate surface of chapter 56. The packet names the L10 nested-interval handoff only; it does not assert a Dedekind-cut construction, a trichotomy principle, quotient equality of real names, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4012.1 (Closed interval nesting carrier)**.** *A* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, I, S, W, R, Q, H, C, P, N)$$

with the following visible rows:

1. *E is the dyadic closed-endpoint row. It records left and right endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$*, endpoint order, midpoint arithmetic cells, and the dyadic comparison budget for each inspected interval.*
2. *I is the nesting-inclusion row. It displays, for every inspected successor stage, the finite evidence that the successor closed interval is contained in its predecessor.*
3. *S is the shrinking-width schedule. It records the requested precision, the displayed stage bound, and the dyadic inequality that makes the inspected interval width smaller than the requested tolerance.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint, inclusion, width, and stage-bound observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the chosen endpoint stream and the width schedule after E, I, S, W have been displayed.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is opened only after the dyadic endpoints, nesting inclusions, shrinking-width schedule, stream windows, and regular rational readback have been replayed.*
7. *H records componentwise* hsame *transport for the endpoint row, inclusion row, width schedule, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-facing or real-name request through $E, I, S, W, R, Q, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{ClosedIntervalNesting}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *interval certificates,* BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$ *row.*

The classifier compares accepted nesting carriers only through $E, I, S, W, R, Q, H, C, P, N$. It has no coordinate for Dedekind cuts, trichotomy, quotient equality of real names, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ClosedIntervalNestingUp.TasteGate.ClosedIntervalNestingTasteGate single carrier alignment`

Theorem 4012.2 (Closed interval nesting NameCert obligations)**.** *For every accepted* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier $K = (E, I, S, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic closed-endpoint exposure, successor nesting inclusion, shrinking-width scheduling, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the carrier of Definition 4012.1.*

Proof. Carrier admission is projection from Definition 4012.1. The endpoint row E and inclusion row I are the only rows that expose closed-interval data, so every local naming obligation begins with displayed dyadic endpoints and finite successor containment.

The width and window obligations are consecutive. The schedule row S supplies the stage bound and dyadic width inequality, while W exposes the finite endpoint, inclusion, width, and stage observations used by that schedule.

The readback rows R, Q are downstream of those displayed observations. The regular rational row repacks the endpoint stream under the shrinking-width bound, and the real seal is available only after that regular readback has been replayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for Dedekind cuts, trichotomy, quotient equality, selected limits, choice, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 4012.3 (Closed interval nesting RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{ClosedIntervalNesting}^{\uparrow}$ *carrier reaches regular rational data only through the displayed finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may be consumed by completion and real-name routes as a nested closed-interval certificate, but it exports no Dedekind-cut construction, trichotomy row, quotient equality of real names, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $K = (E, I, S, W, R, Q, H, C, P, N)$. The endpoint row E and nesting row I expose the only closed-interval information available to a consumer.

Read the shrinking-width schedule S next. It ties every requested tolerance to a displayed stage bound and a dyadic width inequality, so the route cannot be interpreted as a selected intersection point or a hidden completed limit.

The stream and regular rows W, R then turn those finite endpoint windows into regular rational readback. The terminal row Q is a real-name seal only after the dyadic endpoint, inclusion, width, and stream rows have been displayed.

Finally H, C, P, N preserve the same order under transport, replay, provenance, and local naming. Because no stronger object appears among the carrier coordinates, every $\mathsf{RegSeqRat}^{\uparrow}$ handoff remains the displayed finite nested-interval route. □

Falsifiable prediction: A RegSeqRat or Real consumer that omits the dyadic endpoint row, nesting-inclusion row, shrinking-width schedule, StreamName window row, or regular rational readback row cannot claim a $\mathsf{ClosedIntervalNesting}^{\uparrow}$ handoff to a Real-facing seal.

Independence witness: dyadicratcore, streamname, regseqrat, real, interval: $\mathsf{ClosedIntervalNesting}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic closed endpoints, nesting inclusion, shrinking-width scheduling, stream windows, regular readback, a terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedIntervalNesting}^{\uparrow}$ packet is generated from dyadic closed endpoints, successor nesting-inclusion evidence, a shrinking-width schedule, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4012.1, Definition 4012.2, and Definition 4012.3 seed the L10 nested-interval route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, interval certificate rows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedIntervalNesting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction, trichotomy, quotient equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exactness, nesting-inclusion stability, shrinking-width exactness, StreamName window exactness, RegSeqRat handoff, Real-seal nonescape, componentwise transport, displayed replay, package provenance, and local NameCert field faithfulness for BEDC.Derived.ClosedIntervalNestingUp.

Chapter 4013

A Concrete Naming Certificate for $\mathsf{HyperconvexMetric}^{\uparrow}$

Following Aronszajn and Panitchpakdi, the $\mathsf{HyperconvexMetric}^{\uparrow}$ packet records the classical ball-intersection surface as finite BEDC completion-neighbour data. It sits over the $\mathsf{Met}^{\uparrow}$ source of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ boundary of chapter 128, the $\mathsf{MetricClosedBall}^{\uparrow}$ membership row of chapter 1451, the $\mathsf{FormalBallBasis}^{\uparrow}$ refinement route of chapter 3943, and the terminal $\mathsf{Real}^{\uparrow}$ radius seal of chapter 38. The packet names pairwise-compatible displayed balls and a finite intersection handoff; it does not select a global hull point, quotient completion representative, arbitrary family intersection, choice function, or ambient real-completeness principle.

Definition 4013.1 (Hyperconvex metric carrier)**.** *A* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, K, B, F, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the inspected point classifier and distance requests.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion-neighbour seal read over the same metric source.*
3. *B is the finite* $\mathsf{MetricClosedBall}^{\uparrow}$ *family row, listing displayed centres, nonnegative radii, and membership comparison rows.*
4. *F is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *refinement row that records the finite centre-radius basis used before the closed-ball family is inspected.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius and distance seal row, including the terminal real-name comparisons needed by B.*
6. *I is the pairwise-intersection ledger. For every displayed pair of balls in B, it records the finite compatibility inequality $d(c_i, c_j) \leq r_i + r_j$ through M and R.*
7. *E is the ball-intersection handoff row. It records the finite witness request that a common displayed intersection point may be consumed only after M, K, B, F, R, I have been read.*
8. *H records componentwise* hsame *transport for the metric source, complete-metric seal, closed-ball family, formal-ball refinement, real-radius seal, pairwise ledger, handoff row, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from metric distance requests through formal-ball refinement, closed-ball membership, pairwise compatibility, and the completion-neighbour handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperconvexMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite hyperconvex-metric packet. It has no coordinate for a selected injective hull, arbitrary ball-family intersection, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HyperconvexMetricUp.HyperconvexMetricTasteGate single carrier alignment`

Theorem 4013.2 (Hyperconvex metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier* $Q = (M, K, B, F, R, I, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed metric source, complete-metric seal, closed-ball family, formal-ball refinement, real-radius seal, pairwise-intersection ledger, handoff row, transport, replay, provenance, and local naming rows;*
2. *common-source discipline, requiring K, B, F, R, and I to be read over the same* $\mathsf{Met}^{\uparrow}$ *row M;*
3. *closed-ball visibility, requiring every radius and membership request to pass through* $\mathsf{MetricClosedBall}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *before the intersection handoff is exposed;*
4. *basis refinement, requiring the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row to display the centre-radius refinement route used by the closed-ball family;*
5. *pairwise compatibility, requiring every displayed pair of balls to carry the finite inequality ledger I before a common-intersection request reaches E;*
6. *completion-neighbour nonescape, excluding selected injective hulls, arbitrary family intersections, quotient completion equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

No hyperconvex metric obligation is stored outside these rows.

Proof. Project the carrier of Definition 4013.1. The projection exposes exactly the metric source M, complete-metric seal K, closed-ball family B, formal-ball refinement F, real-radius seal R, pairwise ledger I, ball-intersection handoff E, and structural rows H, C, P, N.

The common-source and closed-ball obligations are tied to the same displayed route. The rows K, B, F, R, I are transported componentwise by H, while the replay row C allows radius and membership comparisons only after the source metric and formal-ball basis rows have been displayed.

The intersection obligation is finite. The ledger I lists the pairwise compatibility inequalities for the displayed balls in B, and the handoff row E can be consumed only after that ledger has been read.

The nonescape clauses follow from the carrier shape. None of the rows stores a selected injective hull, arbitrary ball-family intersection, quotient completion equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness witness, so the listed obligations exhaust the local NameCert surface. □

Theorem 4013.3 (Hyperconvex metric ball-intersection handoff)**.** *Every completion-neighbour consumer of an accepted* $\mathsf{HyperconvexMetric}^{\uparrow}$ *carrier* $Q = (M, K, B, F, R, I, E, H, C, P, N)$ *reads the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto I \longmapsto E \longmapsto \mathsf{CompleteMetric}^{\uparrow}$$

before it treats the packet as a ball-intersection source. Thus the consumer receives a displayed metric carrier, formal-ball refinements, closed-ball membership rows, real radius comparisons, and pairwise compatibility evidence, while selected injective hulls, arbitrary ball-family intersections, quotient completion representatives, choice principles, and ambient real completeness remain outside the packet.

Proof. Begin with the metric source. The carrier admits the completion-neighbour read only after M displays the point and distance surface and F displays the finite centre-radius refinement route.

Read the closed-ball and real-radius rows next. The row B supplies the displayed closed balls, while R supplies the terminal real-name comparisons for their radii and distances.

Now inspect pairwise compatibility. The ledger I records the finite inequalities for the displayed family, so the handoff row E is reached only after each required pairwise comparison has been exposed.

Finally the complete-metric seal K and structural rows H, C, P, N package the same route for downstream consumers. Since the carrier has no hull-selection, arbitrary-family, quotient-completion, choice, or ambient-completeness coordinate, the handoff exports none of those objects. □

Falsifiable prediction: Every accepted $\mathsf{HyperconvexMetric}^{\uparrow}$ completion-neighbour read factors through displayed $\mathsf{Met}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, pairwise compatibility, and $\mathsf{CompleteMetric}^{\uparrow}$ rows before any ball-intersection handoff is exposed.

Independence witness: metric, completemetric, metricclosedball, formal-ball-basis, real: $\mathsf{HyperconvexMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds pairwise-compatible closed-ball families, formal-ball refinement, real-radius comparison, complete-metric sealing, ball-intersection handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperconvexMetric}^{\uparrow}$ packet is generated from a displayed MetricSpace row, CompleteMetric seal, MetricClosedBall family, FormalBallBasis refinement, Real radius and distance seals, pairwise compatibility ledger, ball-intersection handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4013.1, Definition 4013.2, and Definition 4013.3 seed the finite ball-intersection completion-neighbour route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperconvexMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, selected injective hulls, arbitrary ball-family intersections, quotient completion equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-source discipline, closed-ball visibility, basis refinement, pairwise compatibility, and ball-intersection handoff for BEDC.Derived.HyperconvexMetricUp.

Chapter 4014

A Concrete Naming Certificate for $\mathsf{SummableSequence}^{\uparrow}$

Following Hardy, Knopp, and Bishop–Bridges constructive real-series analysis, the $\mathsf{SummableSequence}^{\uparrow}$ packet records the finite route by which a real-valued sequence exposes a Cauchy tail budget before any Real or Completion consumer reads its sum. It sits after the finite partial-sum surface of $\mathsf{RealSeries}^{\uparrow}$ in chapter 1581, the absolute and uniform convergence handoffs of chapter 1700 and chapter 771, the regular rational readback of chapter 301, the stream-window route of chapter 126, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names summability as finite certificate data: a source series, partial-sum windows, Cauchy tail budgets, convergence-test handoff, readback, sealing, transport, replay, provenance, and local naming.

Definition 4014.1 (Summable sequence carrier)**.** *A* $\mathsf{SummableSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, W, T, B, G, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row. It exposes the summand stream and the partial-sum construction only through the finite real-series surface.*
2. *W is the partial-sum window row. It records finite windows of summands, partial sums, and requested tail cutoffs.*
3. *T is the Cauchy tail budget row. For every displayed tolerance request it records a cutoff, two later partial-sum windows, and the dyadic comparison certifying that their difference lies below the requested bound.*
4. *B is the convergence-test handoff row. It is read either through an* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *majorant route or through a* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *route when the summability claim is supplied by a uniform family; if neither handoff is displayed, B carries only the local Cauchy tail budget.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same partial-sum windows and tail budgets.*
6. *R is the* $\mathsf{StreamName}^{\uparrow}$ *observation row for summands, partial sums, cutoffs, tail comparisons, and endpoint-facing reads.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, W, T, B, G, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, windows, tail budget, convergence handoff, readback, stream observations, Real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *replay of the route*

$$A, W, T, B \longmapsto G, R \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{SummableSequence}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{AbsoluteConvergence}^{\uparrow}$, $\mathsf{UniformConvergenceTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, T, B, G, R, E, H, C, P, N$. It has no coordinate for a selected infinite sum, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external convergence theorem not displayed by the handoff row.*

Lean4 ✓ `BEDC.Derived.SummableSequenceUp.SummableSequenceTasteGate single carrier alignment`

Theorem 4014.2 (Summable sequence NameCert obligations)**.** *For every accepted* $\mathsf{SummableSequence}^{\uparrow}$ *carrier*

$$S = (A, W, T, B, G, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the eleven displayed rows;*
2. $\mathsf{RealSeries}^{\uparrow}$ *source exposure, requiring every summand and partial-sum read to pass through A;*
3. *partial-sum window exactness, requiring W to bind every inspected summand, partial sum, and tail cutoff to one finite window;*
4. *Cauchy tail budget exactness, requiring T to pair every tolerance request with a cutoff and later-window comparison;*
5. *convergence-test handoff compatibility, requiring B to expose only the displayed* $\mathsf{AbsoluteConvergence}^{\uparrow}$ *or* $\mathsf{UniformConvergenceTest}^{\uparrow}$ *route when such a route is used;*
6. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback compatibility, requiring G and R to inspect the same partial-sum and tail-budget rows;*
7. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring E to remain downstream of A, W, T, B, G, R;*
8. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No summability obligation is stored outside these rows.

Proof. Project the accepted packet $S = (A, W, T, B, G, R, E, H, C, P, N)$. The source, window, tail-budget, convergence-handoff, readback, stream-observation, Real-seal, and structural rows are exactly the displayed coordinates of the carrier.

The source and window obligations come first. The row A supplies only $\mathsf{RealSeries}^{\uparrow}$ summands and partial sums, while W fixes the finite windows in which tail cutoffs and later partial sums are inspected.

The Cauchy tail budget is the row T. It pairs each tolerance request with a cutoff and the finite comparisons between later partial sums. The handoff row B may justify that budget through the displayed absolute or uniform convergence route, but it cannot import an unlisted convergence theorem.

Finally G, R, E, H, C, P, N read back, observe, seal, transport, replay, source, and name that same route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 4014.3 (Summable sequence Real-series handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{SummableSequence}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{RealSeries}^{\uparrow}, \mathsf{AbsoluteConvergence}^{\uparrow}, \mathsf{UniformConvergenceTest}^{\uparrow} \longmapsto A, W, T, B \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto E,$$

The handoff states that summability is consumed through displayed partial-sum windows and Cauchy tail budgets before the $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal are exposed. It exports no selected infinite sum, quotient stream equality, host real equality, countable-choice selector, ambient completeness theorem, or convergence principle detached from the finite packet.*

Proof. Start from the $\mathsf{RealSeries}^{\uparrow}$ row A. It supplies the summand stream and partial sums, while W records the finite partial-sum windows and tail cutoffs available to the consumer.

Read the tail budget through T and the optional convergence-test handoff through B. The absolute-convergence or uniform-convergence route can be consumed only when it is displayed in B, and the local Cauchy tail comparison remains the row that later readback uses.

The rows G and R expose the same partial-sum and tail-budget data as regular rational readback and stream observations. The Real seal E is therefore downstream of the finite Cauchy tail route.

The structural rows H, C, P, N preserve, replay, source, and name the same packet. Because the carrier contains no selected infinite sum, quotient equality, host equality, choice, ambient completeness, or detached convergence coordinate, every Real or Completion handoff stays inside the displayed route. □

Falsifiable prediction: An accepted $\mathsf{SummableSequence}^{\uparrow}$ read is refuted if a Real or Completion consumer exports a Real seal without displaying a RealSeries source row, partial-sum windows, Cauchy tail budget, convergence-test handoff when used, RegSeqRat readback, StreamName observations, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: realseries, absolute-convergence, uniform-convergence-test, regseqrat, streamname, real: none alone is bijective with $\mathsf{SummableSequence}^{\uparrow}$ because the packet jointly carries source partial sums, Cauchy tail budgets, convergence-test handoff, regular readback, stream observations, Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{SummableSequence}^{\uparrow}$ packet is generated from a RealSeriesUp source row, partial-sum window rows, Cauchy tail-budget rows, AbsoluteConvergenceUp or UniformConvergenceTestUp handoff rows when displayed, RegSeqRatUp readback, StreamNameUp observations, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4014.1, Definition 4014.2, and Definition 4014.3 seed the finite summability certificate surface from real-series partial sums through Cauchy tail budgets to regular readback and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummableSequence}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.SummableSequenceUp.SummableSequenceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected infinite sums, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, detached convergence principles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partial-sum window exactness, Cauchy tail-budget stability, convergence-test handoff compatibility, RegSeqRat readback, StreamName observation compatibility, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.SummableSequenceUp.

Chapter 4015

A Concrete Naming Certificate for $\mathsf{CompactLebesgueNumber}^{\uparrow}$

Following Lebesgue, Heine–Borel compactness, and Bishop–Bridges compact metric analysis, the $\mathsf{CompactLebesgueNumber}^{\uparrow}$ packet records the compact finite cover-radius ledger used by the compact uniform-continuity proof of chapter 9237. It refines the compact-metric source of chapter 129, the finite positive-radius ledger of chapter 452, and the uniform-modulus consumer of chapter 369. The packet is a finite NameCert surface for compact finite nets, located cover radii, lower-bound folds, and uniform-modulus handoff; it is not open-cover compactness, a selected subcover, or an external minimum principle.

Definition 4015.1 (Compact Lebesgue-number carrier)**.** *A* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, B, C, R, F, U, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It exposes compact finite-net and metric inspection data through the compact-metric chapter.*
2. *B is the finite* ProbeBundle *cover spine naming the compact cover candidates inspected by the packet.*
3. *C is the center and coverage ledger. It records center histories, located-ball membership rows, and the finite cover witness for every compact-source history read by the packet.*
4. *R is the positive cover-radius row, read through dyadic and rational-ball radius data attached to the bundled probes.*
5. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-radius row. It folds the finite radius ledger to a positive radius below every displayed cover radius.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer handoff row. It exports the lower-radius ledger only after K, B, C, R, F have been displayed.*
7. *H records componentwise* hsame *transport for the compact source, bundle spine, coverage ledger, radius rows, finite Lebesgue-number row, uniform-modulus handoff, replay, provenance, and local name.*
8. *Q records displayed* Cont *replay of the route*

$$K, B, C, R \longmapsto F \longmapsto U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RationalBall}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, C, R, F, U, H, Q, P, N$. It has no coordinate for an arbitrary open cover, a selected finite subcover, host compactness, a host minimum, quotient real equality, countable choice, or a uniform modulus detached from the finite lower-radius row.

Lean4 ✓ `BEDC.Derived.CompactLebesgueNumberUp.CompactLebesgueNumberTasteGate single carrier alignment`

Theorem 4015.2 (Compact Lebesgue-number NameCert obligations)**.** *For every accepted* CompactLebesgueNumb *carrier*

$$L = (K, B, C, R, F, U, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the ten displayed rows;*
2. *compact-metric source exposure, requiring every compact-source read to pass through K;*
3. *finite ProbeBundle coverage exactness, requiring B and C to bind every compact-source history to a named cover probe and located-ball row;*
4. *positive radius ledger exactness, requiring R to attach displayed dyadic and rational-ball radii to the bundled probes;*
5. *finite Lebesgue-number fold compatibility, requiring F to be a* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *row below every displayed radius in R;*
6. *uniform-modulus handoff, requiring U to consume only the lower-radius row and the compact source route already displayed;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, Q, P, N.*

No compact Lebesgue-number obligation is stored outside these rows.

Proof. Project the accepted packet $L = (K, B, C, R, F, U, H, Q, P, N)$. The compact source, bundle, coverage, radius, lower-fold, uniform handoff, and structural rows are precisely the displayed carrier coordinates.

The compact and coverage obligations come first. The row K supplies the compact-metric source, while B and C expose only named probes, center histories, located-ball reads, and finite cover witnesses.

The radius obligations are finite. The row R records the positive radii attached to bundled probes, and F records the $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ lower-radius ledger below those radii. No host minimum or selected open subcover is needed to read F.

Finally U, H, Q, P, N hand the lower radius to uniform-modulus consumers and preserve, replay, source, and name the same route. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 4015.3 (Compact Lebesgue-number uniform-modulus handoff)**.** *Every compact-uniform-continuity or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto B, C, R \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto H, Q, P, N.$$

Thus the finite cover-radius ledger of chapter 9237 is exposed as compact-metric source data, named cover probes, positive radius rows, a finite Lebesgue-number lower-bound fold, and a uniform-modulus handoff. It exports no arbitrary open cover, selected subcover, host compactness theorem, host minimum, countable-choice selector, quotient real equality, or detached uniform modulus.

Proof. Start from the compact-metric source K. It provides the finite compact inspection data, while B and C record the actual cover spine and located coverage rows consumed by the packet.

Read the radius ledger R and the finite lower-bound row F. The lower-bound row is a $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ packet tied to the displayed probe radii, so it is not a host minimum and not a selected subcover argument.

The handoff row U sends exactly this lower-radius ledger to the $\mathsf{UniformModulus}^{\uparrow}$ consumer. The transport, replay, provenance, and naming rows H, Q, P, N preserve the same path.

Because the carrier has no coordinate for an arbitrary open cover, selected subcover, host compactness theorem, host minimum, choice, quotient real equality, or detached modulus, every compact-uniform-continuity read factors through the displayed route. □

Theorem 4015.4 (Compact Lebesgue-number Real-uniform entourage route)**.** *Every* $\mathsf{RealUniformStructure}^{\uparrow}$ *or* $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *packet must read the* $\mathsf{CompactMetric}^{\uparrow}$ *source,* ProbeBundle *cover spine, located coverage rows, positive radius ledger,* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-bound fold, and* $\mathsf{UniformModulus}^{\uparrow}$ *handoff before any Real-induced entourage basis is exported. The route supplies no arbitrary open cover, selected subcover, host minimum, quotient real equality, countable choice, or ambient Real completeness.*

Proof. Project the accepted carrier $L = (K, B, C, R, F, U, H, Q, P, N)$. The source row K, cover spine B, located coverage ledger C, positive radius row R, finite lower-bound fold F, and uniform handoff row U are the only analytic coordinates available before the structural rows.

Apply Definition 4015.3. It forces every compact-uniform-continuity read to pass through the displayed compact source, finite cover spine, located coverage rows, positive radii, lower-bound fold, and $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Route the resulting lower-bound ledger through the Real metric-ball surface of chapter 1624 and the basis packet of chapter 5115. A Real-induced entourage basis can therefore be exported only as a downstream read of the same compact finite-radius route.

Because arbitrary open covers, selected subcovers, host minima, quotient real equality, countable choice, and ambient Real completeness are absent from the carrier coordinates, no Real-uniform consumer can obtain them from the exported entourage basis. □

Falsifiable prediction: An accepted $\mathsf{CompactLebesgueNumber}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a UniformModulusUp row without displaying CompactMetric source data, a ProbeBundle cover spine, located coverage rows, positive radius rows, a FiniteLebesgueNumber lower-bound fold, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, finite-lebesgue-number, uniformmodulus: none alone is bijective with $\mathsf{CompactLebesgueNumber}^{\uparrow}$ because the packet jointly carries compact source data,

finite cover probes, located coverage rows, positive radii, lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}$.

Constructive story: A $\mathsf{CompactLebesgueNumber}^{\uparrow}$ packet is generated from CompactMetricUp source data, a finite ProbeBundle cover spine, located coverage rows, positive dyadic and rational-ball radius rows, a FiniteLebesgueNumberUp lower-bound fold, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4015.1, Definition 4015.2, and Definition 4015.3 seed the compact finite Lebesgue-number packet from compact-metric cover data through finite lower-radius folds to UniformModulusUp handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactLebesgueNumber}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host compactness, host minima, quotient real equality, countable choice, detached uniform moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source exposure, ProbeBundle coverage exactness, positive-radius ledger stability, FiniteLebesgueNumber fold compatibility, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactLebesgueNumberUp.

Chapter 4016

A Concrete Naming Certificate for $\mathsf{CompactPolishSpace}^{\uparrow}$

Following Alexandrov and Urysohn on Polish spaces and Kelley on compact metric spaces, the $\mathsf{CompactPolishSpace}^{\uparrow}$ packet records the finite BEDC surface where compact-metric data and complete separable metric data are read as one named object. It sits over the $\mathsf{CompactMetric}^{\uparrow}$ chapter of chapter 129, the $\mathsf{PolishSpace}^{\uparrow}$ chapter of chapter 1490, the $\mathsf{CompleteMetric}^{\uparrow}$ route of chapter 128, the $\mathsf{SeparableMetric}^{\uparrow}$ route of chapter 1458, the finite observation windows of chapter 126, and the regular rational readback surface of chapter 301. The packet is a synthesis node for compact-completion and continuous-map consumers; it is not an open-cover compactness theorem, a host completion, a countable-choice selector, or a quotient of dense names.

Definition 4016.1 (Compact Polish space carrier)**.** *A* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (K, P, C, S, W, R, H, T, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row carrying the finite compact-metric witness, including the metric row, total-boundedness window, and compact-test handoff.*
2. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row naming the same metric carrier as a complete separable metric surface.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row through which Cauchy-facing consumers read completeness data before any Real or completion seal is inspected.*
4. *S is the* $\mathsf{SeparableMetric}^{\uparrow}$ *row displaying the finite dense-name schedule available to separability consumers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row used by compact-net, dense-name, and completion-facing reads.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row consumed after the finite stream window has been displayed.*
7. *H records componentwise* hsame *transport for the compact-metric, Polish-space, complete-metric, separable-metric, stream-window, regular-readback, package, and local naming rows.*
8. *T records displayed* Cont *replay from compact-metric witnesses and dense schedules through complete separable readback.*

9. *Q is the* Pkg *provenance over* $\mathsf{CompactPolishSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, host compactness, host completion, selected dense enumerations, quotient Cauchy classes, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.CompactPolishSpaceUp.TasteGate.CompactPolishSpaceUp`

Theorem 4016.2 (Compact Polish space NameCert obligations)**.** *For every accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier $X = (K, P, C, S, W, R, H, T, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$ *surface consists exactly of compact-metric admission, Polish-space admission, complete-metric handoff, separable-metric handoff, finite stream-window exposure, regular-rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must read the compact side through K, the Polish side through P, C, S, and the finite observation side through W, R before using the packet as a compact Polish space.*

Lean4 ✓ `BEDC.Derived.CompactPolishSpaceUp.CompactPolishSpaceCarrier namecert obligations`

Proof. Project the carrier of Definition 4016.1. The tuple lists the only rows visible to the local classifier, so carrier admission is confined to $K, P, C, S, W, R, H, T, Q, N$.

The compact side is supplied by K. It gives finite compact-metric evidence, but no open-cover compactness principle or ambient compact object.

The Polish side is supplied by P through the complete and separable rows C and S. Completeness-facing reads therefore use the displayed complete-metric handoff, and separability-facing reads use the displayed dense-name schedule.

The observation and readback rows W and R bind those two sides to the same finite stream window and regular-rational surface. Transport, replay, provenance, and local naming are then componentwise through H, T, Q, N, leaving no coordinate for a host completion, countable-choice selector, or quotient dense-name equality. □

Theorem 4016.3 (Compact Polish finite-cover readback)**.** *Every compact-completion or continuous-map consumer of an accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{SeparableMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

through the displayed rows $K, P, C, S, W, R, H, T, Q, N$. The exported readback is the compact-metric witness together with complete separable metric data and finite regular-name observations; it exports no open-cover compactness theorem, selected host completion, countable-choice dense enumeration, quotient Cauchy class, or ambient completeness row.

Lean4 ✓ `BEDC.Derived.CompactPolishSpaceUp.CompactPolishSpaceCarrier readback certificate`

Proof. Start from the obligation surface of Definition 4016.2. A consumer first reads the compact-metric row K and the Polish-space row P as the two named views of the same carrier.

The complete and separable handoffs are then forced through C and S. Thus completion-facing reads see a complete-metric row, while dense-name reads see a separable-metric row, and neither read can introduce a host completion or selected dense enumeration.

Finally the finite observation route passes through W and R. The structural rows H, T, Q, N preserve the same route by transport, replay, provenance, and local naming, so the displayed rows exhaust the handoff available to compact-completion and continuous-map consumers. □

Theorem 4016.4 (Compact Polish compact-complete-separable synthesis)**.** *Every accepted* $\mathsf{CompactPolishSpace}^{\uparrow}$ *consumer that asks for compact-completion or continuous-map data must factor through the single finite route*

$$K, P, C, S, W, R, H, T, Q, N.$$

Thus compactness, completeness, separability, stream windows, and regular-rational readback are consumed as one carrier surface and cannot be read independently. The route exports no open-cover compactness, host completion, selected dense enumeration, quotient Cauchy class, countable choice, or ambient completeness coordinate.

Proof. Project the carrier rows in Definition 4016.1. The compact-metric row K, Polish-space row P, complete-metric row C, separable-metric row S, stream row W, and regular-rational row R are displayed before any structural replay can be used.

Apply the obligation surface of Definition 4016.2. It identifies the local NameCert read as exactly $K, P, C, S, W, R, H, T, Q, N$, so compactness, completeness, separability, finite observation, and readback are bound to the same packet.

Finally use finite-cover readback from Definition 4016.3. Compact-completion and continuous-map consumers therefore follow the displayed route, and no coordinate exists for open covers, host completions, selected dense enumerations, quotient Cauchy classes, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactPolishSpace}^{\uparrow}$ consumer read factors through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows before it is used as compact Polish data.

Independence witness: compactmetric, polishspace, completemetric, separablemetric, streamname, regseqrat: $\mathsf{CompactPolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds compact-metric witnesses and complete separable metric rows through one finite observation and local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactPolishSpace}^{\uparrow}$ packet is generated from $\mathsf{CompactMetric}^{\uparrow}$ compact-metric rows, PolishSpaceUp complete separable rows, $\mathsf{CompleteMetric}^{\uparrow}$ handoff rows, SeparableMetricUp dense-name rows, StreamNameUp windows, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4016.1, Definition 4016.2, and Definition 4016.3 seed the finite compact Polish synthesis node over compact-metric and complete separable metric data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactPolishSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, host completion, selected dense enumerations, quotient Cauchy classes, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric admission, Polish-space admission, complete-metric handoff, separable-metric handoff, finite observation readback, transport, replay, provenance, and local naming for BEDC.Derived.CompactPolishSpaceUp.

Chapter 4017

A Concrete Naming Certificate for $\mathsf{PartitionCoreIncidence}^{\uparrow}$

The $\mathsf{PartitionCoreIncidence}^{\uparrow}$ packet realizes the partition-core incidence representation surface of Definition 9462.1755 as finite BEDC NameCert data. It sits over the partition rows of chapter 201, the graph incidence rows of chapter 118, the support-nerve surface of chapter 238, the rank-dependence boundary of chapter 202, and the boundary-tile source of chapter 3393. The packet names the finite incidence kernel consumed by boundary-tile and representation-facing reads; it is not a host representation module, a quotient linear-algebra oracle, or an unrestricted biological model.

Definition 4017.1 (Partition-core incidence carrier)**.** *A* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (P, G, S, M, B, K, V, T, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the partition-core row naming the finite partition blocks and the accepted core labels.*
2. *G is the graph-incidence row listing the displayed source vertices, target cores, and accepted incidence edges.*
3. *S is the support-nerve row, read through* $\mathsf{SimplicialComplex}^{\uparrow}$*, whose faces record nonempty incidence support.*
4. *M is the* $\mathsf{Matroid}^{\uparrow}$ *dependence row recording which displayed incidence supports are treated as dependent.*
5. *B is the* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ *source row carrying the boundary-tile coordinates that feed the incidence read.*
6. *K is the partition-core kernel row displaying the accepted null combinations of boundary-tile incidence reads.*
7. *V is the representation-basis row recording the finite trivial, standard, signed, and two-dimensional summand labels used by the readback.*
8. *T is the finite transport row from boundary-tile source reads through partition-core incidence, support-nerve, matroid-dependence, and kernel rows.*
9. *R is the replay row recording the accepted provenance of every local incidence read.*

10. *H records componentwise* hsame *transport for all displayed rows.*
11. *C records displayed* Cont *replay through the partition-core, incidence, support-nerve, transport, replay, provenance, and local naming rows.*
12. *Q is the* Pkg *provenance over* $\mathsf{PartitionCoreIncidence}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$ *row.*

The carrier exports only these finite rows. It contains no host representation space, hidden quotient module, unrestricted linear-algebra solver, stochastic biological model, or global codon semantics.

Lean4 ✓ `BEDC.Derived.PartitionCoreIncidenceUp.PartitionCoreIncidenceTasteGate single carrier alignment`

Theorem 4017.2 (Partition-core incidence NameCert obligations)**.** *For every accepted* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *carrier* $I = (P, G, S, M, B, K, V, T, R, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$ *surface consists exactly of partition-core admission, graph-incidence admission, support-nerve exposure, matroid-dependence exposure, boundary-tile source admission, kernel-row readback, representation-basis readback, transport, replay, componentwise* hsame*, displayed* Cont, Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4017.1. The partition, graph, support-nerve, matroid, boundary-tile, kernel, and representation rows are the only analytic coordinates available to the local classifier.

The incidence read starts at B and reaches partition cores only through P and G. Support and dependence are then displayed by S and M, so a consumer sees finite support evidence before it reads kernel membership.

The rows K and V record the finite kernel and representation-basis readback. Transport, replay, componentwise equivalence, continuation replay, provenance, and local naming are exactly T, R, H, C, Q, N, leaving no coordinate for host representation modules, quotient linear algebra, stochastic biological dynamics, or global codon semantics. □

Theorem 4017.3 (Partition-core incidence kernel handoff)**.** *Every accepted* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *read factors through the displayed finite route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{Partition}^{\uparrow}, \mathsf{Graph}^{\uparrow} \longmapsto \mathsf{SimplicialComplex}^{\uparrow}, \mathsf{Matroid}^{\uparrow} \longmapsto K, V$$

together with transport, replay, provenance, and local naming rows. Boundary-tile consumers and representation-facing consumers therefore meet only at the displayed partition-core incidence kernel, not at a host quotient module or unrestricted linear-algebra surface.

Proof. Use Definition 4017.2. The boundary-tile source row B supplies the input coordinates, while P and G expose the partition-core and incidence-edge rows through which those coordinates can be read.

The support and dependence checks then pass through S and M. These rows display the support nerve and the finite dependence boundary before any kernel readback is accepted.

Finally K and V expose the finite kernel and representation-basis rows. The structural rows T, R, H, C, Q, N preserve that same route, so the consumer receives only the displayed finite incidence packet. □

Falsifiable prediction: Every accepted $\mathsf{PartitionCoreIncidence}^{\uparrow}$ read factors through displayed partition-core, incidence, support-nerve, transport, replay, provenance, and local naming rows, and any read lacking one of those rows is rejected by the local NameCert surface.

Independence witness: partition, graph, simplicialcomplex, matroid, dna-window-six-boundary-tile: $\mathsf{PartitionCoreIncidence}^{\uparrow}$ is not bijective to any listed sibling because it binds boundary-tile source rows to partition-core incidence, support-nerve, matroid-dependence, kernel readback, representation-basis, transport, replay, provenance, and local naming as one packet.

Lean follow-up provides $\mathsf{Nontrivial\ PartitionCoreIncidence}^{\uparrow}$ and $\mathsf{FieldFaithful\ PartitionCoreIncidence}^{\uparrow}$ in TasteGate.lean.

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}$.

Constructive story: A $\mathsf{PartitionCoreIncidence}^{\uparrow}$ packet is generated from partition-core rows, graph-incidence rows, support-nerve rows, matroid-dependence rows, DNA boundary-tile source rows, kernel readback, representation-basis readback, transport, replay, componentwise hsame, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4017.1, Definition 4017.2, and Definition 4017.3 seed the finite partition-core incidence packet for boundary-tile and representation-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host representation modules, quotient linear algebra, unrestricted biological dynamics, global codon semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-core incidence exposure, support-nerve exposure, matroid-dependence exposure, kernel readback, representation-basis readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionCoreIncidenceUp.

Chapter 4018

A Concrete Naming Certificate for $\mathsf{BaireModulus}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Bishop, the $\mathsf{BaireModulus}^{\uparrow}$ packet records the finite-prefix modulus surface by which a Baire-space read exposes a bounded inspection window before any real-name or fan-functional consumer sees a value. It sits between the finite prefix discipline of chapter 334, the stream-window route of chapter 126, the unary index surface of $\mathsf{Nat}^{\uparrow}$, the dyadic and regular rational handoff of chapter 291 and chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the finite modulus-extraction consumer of chapter 1525. The packet is not host sequence equality, not a selected infinite branch, not a quotient stream object, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4018.1 (Baire modulus carrier)**.** *A* $\mathsf{BaireModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, W, \mu, R, H, C, P, N)$$

with the following displayed rows:

1. *B is a* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix packet, naming the inspected stream schedule and the prefix-cylinder classifier.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row from which the requested stream coordinates are read.*
3. *W is a unary* $\mathsf{Nat}^{\uparrow}$ *index window listing the finite requests admitted by the modulus.*
4. *μ is the displayed prefix-modulus row, asserting that every permitted consumer read is determined by the rows of W and no tail outside that window.*
5. *R is the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff surface attached to the displayed finite prefix.*
6. *H records componentwise* hsame *transport for the Baire prefix row, stream schedule, unary index window, modulus row, and handoff surface.*
7. *C records displayed* Cont *replay of the same finite prefix requests and handoff rows.*
8. *P is the* Pkg *provenance over* $\mathsf{BaireModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, W, \mu, R, H, C, P, N$. There is no coordinate for host sequence equality, selected branches, quotient streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4018.2 (Baire modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireModulus}^{\uparrow}$ *carrier* $M = (B, S, W, \mu, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, prefix-modulus exactness, finite-window transport, handoff provenance, and nonescape rows. Carrier admission exposes the* $\mathsf{BaireSpace}^{\uparrow}$ *packet,* $\mathsf{StreamName}^{\uparrow}$ *schedule, unary* $\mathsf{Nat}^{\uparrow}$ *window, displayed prefix modulus,* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. Prefix-modulus exactness requires every public read to factor through the listed finite window W before the handoff row R or a* $\mathsf{FanFunctional}^{\uparrow}$ *consumer is visible. The nonescape rows exclude host sequence equality, selected branches, quotient streams, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireModulusUp.BaireModulusUpTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 4018.1. The only stream data available are the $\mathsf{BaireSpace}^{\uparrow}$ finite prefix row B, the $\mathsf{StreamName}^{\uparrow}$ schedule S, and the unary index window W.

The row μ is read after those rows. It binds consumer visibility to the displayed prefix window, so the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FanFunctional}^{\uparrow}$ handoffs can only consume prefix data already listed in W.

Transport and replay do not add data. The componentwise hsame rows in H and the displayed Cont routes in C reread the same finite requests, while P and N only record provenance and local naming. Since no excluded object is a carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 4018.3 (Baire modulus prefix handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{FanFunctional}^{\uparrow}$*-facing public read of an accepted* $\mathsf{BaireModulus}^{\uparrow}$ *packet factors through the route*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Nat}^{\uparrow} \longmapsto \mathsf{BaireModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow},$$

together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$ *row. The handoff exports a finite prefix modulus and no host sequence equality, branch choice, quotient stream, countable-choice row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Begin with Definition 4018.2. Carrier admission opens the Baire prefix packet, stream schedule, unary window, and modulus row before any downstream consumer can inspect the handoff surface.

Read the handoff row R. Its dyadic, regular-rational, and real-name coordinates are attached to the finite window selected by W and bounded by μ, so they cannot request a tail coordinate outside the displayed prefix.

Finally pass the same finite route to the $\mathsf{FanFunctional}^{\uparrow}$ consumer. The consumer receives the displayed modulus row, transported by H and replayed by C, with provenance in P and local naming in N. Because the carrier has no host equality, selected branch, quotient stream, countable-choice, or real-completeness coordinate, the public handoff has exactly the stated finite-prefix form. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4018.1, Definition 4018.2, and Definition 4018.3 seed the finite-prefix modulus route over $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FanFunctional}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireModulus}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host sequence equality, branch choice, quotient streams, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-modulus exactness, finite-window transport, handoff provenance, and nonescape for BEDC.Derived.BaireModulusUp.

Constructive story: A $\mathsf{BaireModulus}^{\uparrow}$ packet is generated from BaireSpace finite prefix rows, StreamName schedules, unary Nat index windows, displayed prefix-modulus rows, DyadicRatCore, RegSeqRat, and Real handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4019

A Concrete Naming Certificate for $\mathsf{CompactMetricSequentialNet}^{\uparrow}$

Following Bishop and Bridges constructive compactness, the $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ packet joins compact metric finite-net data with sequential extraction as a finite L10 neighbour of chapter 60, chapter 129, chapter 57, and chapter 126. It reads total-bounded net schedules, complete-metric Cauchy routes, finite stream windows, regular rational readback, and terminal Real seals as displayed NameCert rows. The packet supports Bolzano–Weierstrass and sequential compactness consumers while refusing open-cover compactness, countable choice, host subsequence selection, quotient stream equality, and detached real completeness.

Definition 4019.1 (CompactMetricSequentialNet carrier)**.** *A* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the* $\mathsf{CompactMetric}^{\uparrow}$ *surface and its compact and metric dependencies.*
2. *T is the total-bounded net schedule, recording finite centers, mesh radii, coverage witnesses, and refinement rows for the requested tolerances.*
3. *L is the complete-metric Cauchy route that receives the extracted Cauchy window after the finite-net schedule has stabilized.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger for the source sequence, selected indices, selected source windows, and monotonicity evidence.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected windows and dyadic tail budgets.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached by the readback and complete-metric route.*
7. *H records componentwise* hsame *transport for the compact metric source, net schedule, Cauchy route, stream windows, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact metric finite nets through sequential windows, regular rational readback, and the terminal seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, T, L, W, R, E, H, C, P, N$. It has no coordinate for an externally chosen subsequence, open-cover compactness, quotient stream equality, countable choice, host equality, or detached $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactMetricSequentialNetUp.CompactMetricSequentialNetTasteGate single carrier alignment`

Theorem 4019.2 (CompactMetricSequentialNet NameCert obligations)**.** *For every accepted* CompactMetricSequen *carrier*

$$S = (K, T, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$ *surface consists exactly of compact metric source admission through K, total-bounded finite-net scheduling through T, complete-metric Cauchy routing through L, finite stream and subsequence windows through W, regular rational readback through R, terminal Real sealing through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 4019.1. The accepted packet exposes exactly the rows $K, T, L, W, R, E, H, C, P, N$.

The compact metric side is finite. The source row K supplies the metric and compact data, while T supplies the total-bounded finite-net schedule used to choose inspected windows at each displayed tolerance.

The sequential side is the row W. It records only finite source windows, selected indices, selected windows, and monotonicity evidence, so no host subsequence selector or countable-choice schedule is available.

The completion-facing side is L, R, E. The selected windows are read through the regular rational row R, then through the complete-metric route L, and finally through the terminal Real seal E.

The rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate outside these rows, the listed non-escape clauses are part of the NameCert surface. □

Theorem 4019.3 (CompactMetricSequentialNet subsequence handoff)**.** *Every compact-metric or Bolzano–Weierstrass consumer of an accepted* $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ *packet reads the subsequence handoff through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports the displayed finite-net schedule, selected source windows, regular rational readback, complete-metric Cauchy route, Real seal, and structural rows only; it exports no open-cover compactness theorem, selected infinite subsequence, quotient stream equality, countable choice, host equality, or detached real-completeness principle.

Proof. Start from Definition 4019.2. The compact metric source row and the total-bounded schedule must be displayed before a sequence consumer may inspect any selected window.

Next read the stream row W. The row contains the finite source windows and the selected-window evidence consumed by the downstream regular rational readback, but it does not contain an independent infinite subsequence object.

Then apply R, L, E in order. The regular rational readback is attached to the displayed windows, the complete-metric route receives that Cauchy data, and the Real seal is reached only at the end of the displayed route.

Finally H, C, P, N preserve the same packet. Therefore every public handoff is confined to the route above and cannot export the excluded open-cover, choice, quotient, host-equality, or detached-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricSequentialNet}^{\uparrow}$ packet is generated from compact metric source rows, total-bounded finite-net schedules, complete-metric Cauchy routes, StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4019.1, Definition 4019.2, and Definition 4019.3 seed the finite compact-metric sequential extraction route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricSequentialNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite subsequences, quotient stream equality, countable choice, host equality, detached Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net scheduling, sequential-window monotonicity, RegSeqRat readback, complete-metric routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactMetricSequentialNetUp.

Chapter 4020

A Concrete Naming Certificate for $\mathsf{BorelHierarchy}^{\uparrow}$

Following Kechris and Bourbaki in classical descriptive set theory, the $\mathsf{BorelHierarchy}^{\uparrow}$ packet records the finite BEDC boundary by which a Polish or Borel-facing consumer inspects a displayed hierarchy rank without importing the ambient host universe of all Borel subsets. It sits over the generated-set surface of chapter 3384, the open-schedule boundary of chapter 3547, and the complete separable source of chapter 1490. The packet exposes rank schedules, successor and limit-stage replay, StreamName and RegSeqRat readback, transport, continuation replay, provenance, and local naming; it does not claim classical descriptive-set completeness, selected codes, quotient set equality, countable choice, or measure-theoretic structure.

Definition 4020.1 (Borel hierarchy carrier)**.** *A* $\mathsf{BorelHierarchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (X, B, G, P, S, R, T, C, N)$$

with the following displayed rows:

1. *X is the source-space row, read through the Polish or topology-facing source visible to the current consumer.*
2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *row that receives generated-set handoff after the hierarchy schedule has been inspected.*
3. *G is either the* $\mathsf{GDeltaSet}^{\uparrow}$ *row or the displayed open-generator row used as the first nontrivial hierarchy surface.*
4. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row when the hierarchy is being read over a complete separable metric presentation.*
5. *S is the countable rank schedule. Each visible stage records the rank tag, predecessor window, successor or limit-stage instruction, and the finite constructor route offered to the consumer.*
6. *R is the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rank positions, predecessor windows, open-generator observations, and real-facing regular rational queries.*
7. *T records componentwise* hsame *transport for the source, Borel row, G-delta or open-generator row, Polish row, rank schedule, readback, continuation, and local naming rows.*
8. *C records displayed* Cont *replay from a source query through the visible rank schedule, successor or limit-stage constructor, generated-set handoff, and readback route.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, B, G, P, S, R, T, C, N$. It has no coordinate for a host subset, a chosen Borel code, countable choice over ranks, quotient set equality, a completed descriptive-set universe, or a measure-theoretic assertion.

Lean4 ✓ `BEDC.Derived.BorelHierarchyUp.BorelHierarchyTasteGate single carrier alignment`

Theorem 4020.2 (Borel hierarchy NameCert obligations)**.** *For every accepted* $\mathsf{BorelHierarchy}^{\uparrow}$ *carrier*

$$H = (X, B, G, P, S, R, T, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *source admission for the displayed source-space row and, when present, the Polish source row;*
2. *rank-schedule exposure, requiring every inspected hierarchy stage to appear in the finite visible prefix of S before any generated-set row is consumed;*
3. *successor-stage replay, requiring successor constructors to read their predecessor window, constructor tag, StreamName observation, and RegSeqRat readback through S and R;*
4. *limit-stage replay, requiring limit constructors to expose only the displayed finite prefix, schedule position, and readback windows visible in S and R;*
5. *generated-set handoff, requiring the* $\mathsf{BorelSet}^{\uparrow}$ *row B to consume the visible route through X, G, P, S, R;*
6. *transport, requiring T to preserve source, Borel, G-delta or open-generator, Polish, rank-schedule, readback, replay, and naming rows together;*
7. *continuation replay, requiring C to replay the same ordered route before any Borel, G-delta, Polish, StreamName, or RegSeqRat consumer receives the packet;*
8. *provenance, requiring every cited sibling surface to be read only through the displayed rows of chapter 3384, chapter 3547, and chapter 1490;*
9. *local naming, requiring N to certify exactly the rows X, B, G, P, S, R, T, C;*
10. *refusal of host subsets, selected codes, countable choice, quotient set equality, ambient descriptive-set completeness, and measure-theoretic claims.*

No $\mathsf{BorelHierarchy}^{\uparrow}$ *NameCert obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4020.1. The tuple displays exactly the source row, Borel row, G-delta or open-generator row, Polish row, countable rank schedule, finite readback, transport, replay, and local naming rows.

Rank exposure is read through S and R. A successor stage exposes its predecessor window and constructor tag through the visible schedule, while a limit stage exposes only the finite prefix, position, and readback windows visible to the current consumer.

The generated-set handoff then passes to B over the same X, G, P, S, R route. The sibling chapters supply the only accepted surfaces for the Borel row, the G-delta or open-generator row, and the Polish source.

Finally T, C, N transport, replay, and name the displayed route. Since no row stores a host subset, selected code, countable-choice selector, quotient set equality, completed descriptive-set universe, or measure-theoretic datum, the enumerated obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4020.1 and Definition 4020.2 seed the hierarchy route over source, Borel, G-delta or open-generator, Polish, rank-schedule, readback, transport, replay, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorelHierarchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host subsets, selected Borel codes, countable choice, quotient set equality, ambient descriptive-set completeness, measure theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rank-schedule exposure, successor and limit-stage replay, generated-set handoff, transport, replay, provenance, local naming, and ledger refusal for BEDC.Derived.BorelHierarchyUp.

Constructive story: A $\mathsf{BorelHierarchy}^{\uparrow}$ packet is generated from a source-space row, BorelSet row, GDeltaSet or open-generator row, Polish source row, countable rank schedule, finite StreamName and RegSeqRat readback, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Chapter 4021

A Concrete Naming Certificate for RegularSequenceLocalization$^{\uparrow}$

Following Bishop and Bridges' regular Cauchy presentation of constructive reals and Kelley's finite-local analysis of neighborhoods, the RegularSequenceLocalization$^{\uparrow}$ packet records a bounded observation window around a displayed regular Cauchy name as first-class NameCert data. It reads finite stream observations through chapter 126, measures the local radius through chapter 291, replays regular rational evidence through chapter 301, and reaches the terminal real seal of chapter 38. The packet is local: it exposes the center, dyadic radius ledger, finite observation windows, rational readback, seal ordering, transport, replay, provenance, and local name before any Real or Completion consumer may use the localized estimate.

Definition 4021.1 (Regular-sequence localization carrier)**.** *A* RegularSequenceLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (a, \rho, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *a is the finite localization center row, naming the regular Cauchy datum around which the local estimate is inspected.*
2. *ρ is the* DyadicRatCore$^{\uparrow}$ *radius ledger, recording the dyadic scale, comparison cells, and slack rows used by the local window.*
3. *W is the finite* StreamName$^{\uparrow}$ *observation-window row, naming exactly the stream positions read at the requested local precision.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, converting the displayed center, radius ledger, and window observations into regular rational estimate data.*
5. *R is the terminal* Real$^{\uparrow}$ *seal row, available only after a, ρ, W, Q have been replayed.*
6. *H records componentwise* hsame *transport for the center, radius ledger, window observations, regular rational readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a local estimate request through a, ρ, W, Q, and then R.*
8. *P is the* Pkg *provenance over* RegularSequenceLocalization$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RegularCauchyEntourageBasis$^{\uparrow}$, RegularCauchyLimitUniqueness$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted localization packets only through $a, \rho, W, Q, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, ambient real completeness, or a global neighborhood system beyond the displayed finite window.

Lean4 ✓ `BEDC.Derived.RegularSequenceLocalizationUp.RegularSequenceLocalizationTasteGate single carrier alignment`

Theorem 4021.2 (Regular-sequence localization NameCert obligations)**.** *For every accepted* RegularSequenceLocal *carrier $L = (a, \rho, W, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the localization center, dyadic radius ledger, finite stream observation windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows;*
2. *radius-ledger stability, requiring every local estimate to read ρ through the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale and not through a host real order;*
3. *window-read stability, requiring every accepted observation to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is available;*
4. *readback exactness, requiring the route $a, \rho, W \longmapsto Q \longmapsto R$ before any Real or Completion consumer treats the estimate as localized regular-sequence data;*
5. *transport and replay stability for the same a, ρ, W, Q, R, P, N route under* hsame *transport and* Cont *replay.*

No localization NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4021.1. Its mathematical rows are exactly the center a, dyadic radius ledger ρ, finite window row W, regular rational readback Q, and real seal R.

The radius and window obligations are read before the rational handoff. The ledger ρ supplies only dyadic comparison data, and W supplies only the finite stream observations displayed by the packet.

The readback obligation is then forced through Q. Since Q is the sole $\mathsf{RegSeqRat}^{\uparrow}$ coordinate, the packet cannot replace the local estimate by quotient stream equality, a selected limit, countable choice, trichotomy, or ambient completeness.

Finally the rows H, C, P, N transport, replay, source, and name the same route. They do not add analytic coordinates, so the listed rows exhaust the local NameCert surface. □

Theorem 4021.3 (Regular-sequence localization window stability)**.** *Let $L = (a, \rho, W, Q, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceLocalization}^{\uparrow}$ *carrier. If a Real or Completion consumer requests a localized regular-sequence estimate, then every accepted replay factors through the finite window route*

$$a, \rho, W \longmapsto Q \longmapsto R \longmapsto H, C, P, N.$$

Consequently the displayed dyadic radius ledger and finite $\mathsf{StreamName}^{\uparrow}$ *observation window remain stable under* hsame *transport and* Cont *replay before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Begin with a localized estimate request. The carrier offers the center row a, the dyadic radius ledger ρ, and the finite window row W as the only rows available before regular rational readback.

Replay through C. The replay opens a, then reads the dyadic comparison cells in ρ, then inspects the finite observations named by W, and only then exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q.

Transport through H preserves this route componentwise. The real seal R and the structural rows P, N are downstream of the same displayed path, so no replay can consume $\mathsf{Real}^{\uparrow}$ data before the local window and radius ledger have been shown. □

Falsifiable prediction: Every accepted $\mathsf{RegularSequenceLocalization}^{\uparrow}$ read used by a Real or Completion consumer displays a finite center row, one $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, one finite $\mathsf{StreamName}^{\uparrow}$ observation window, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and one local structural NameCert route before the estimate is treated as localized regular-sequence data.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularSequenceLocalization}^{\uparrow}$ is not bijective to any listed sibling because it binds a localization center, radius ledger, finite observation windows, regular rational readback, terminal real seal, transport, replay, provenance, and local NameCert row into one local estimate packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceLocalization}^{\uparrow}$ packet is generated from a finite localization center, a $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, finite $\mathsf{StreamName}^{\uparrow}$ observation windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4021.1, Definition 4021.2, and Definition 4021.3 seed the finite local regular-sequence estimate route over center, radius, window, readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularSequenceLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global real completeness, quotient stream equality, selected limits, countable choice, trichotomy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger stability, finite-window stability, RegSeqRat readback exactness, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularSequenceLocaliza

Chapter 4022

A Concrete Naming Certificate for $\mathsf{LocallyUniformConvergence}^{\uparrow}$

Following Bishop and Bridges on constructive analysis, Kelley on compactly generated uniform control, and Bourbaki on local uniform convergence over compacta, the $\mathsf{LocallyUniformConvergence}^{\uparrow}$ packet records the standard compact-window reading of locally uniform convergence as finite NameCert data. It sits after the uniform-tail certificate of chapter 3417, the continuity rows of chapter 3988, and the compact metric completion boundary of chapter 1687. The packet names compact-window rows, uniform-modulus rows, continuous-function rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming rows; it does not assert open-cover compactness, selected limiting functions, quotient function-space equality, countable choice, or ambient completeness.

Definition 4022.1 (LocallyUniformConvergence carrier)**.** *A* $\mathsf{LocallyUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, U, F, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-window row, listing each retained compact metric window and its finite inspection addresses.*
2. *U is the uniform-modulus row, assigning to every displayed compact window a shared tail modulus for the family on that window.*
3. *F is the continuous-function row, naming the approximating function data and the endpoint continuity data read on the same windows.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for all rational tail and distance comparisons used by the window.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *handoff row reached only after the compact-window, uniform-modulus, continuous-function, and regular-rational rows have been read.*
6. *H records componentwise* hsame *transport for compact windows, moduli, continuous-function rows, rational readbacks, real seals, provenance, and local naming.*
7. *C records displayed* Cont *replay from a local compact-window request through K, U, F, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{LocallyUniformConvergence}^{\uparrow}$*,* $\mathsf{UniformConvergence}^{\uparrow}$*,* $\mathsf{CFun}^{\uparrow}$*,* $\mathsf{CompactMetricComplet}$ $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$ *row.*
The carrier exports only this compact-window route. It has no coordinate for open-cover compactness, a selected global limit function, quotient equality of functions, or a host supremum norm outside the displayed windows.

Lean4 ✓ `BEDC.Derived.LocallyUniformConvergenceUp.TasteGate.LocallyUniformConvergenceTasteGate single carrier alignment`

Theorem 4022.2 (LocallyUniformConvergence NameCert obligations)**.** *For every accepted* LocallyUniformConverg *carrier $L = (K, U, F, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$ *surface consists exactly of compact-window admission through K, shared uniform-modulus admission through U, continuous-function row compatibility through F,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, terminal* $\mathsf{Real}^{\uparrow}$ *handoff through E, and structural transport, replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 4022.1. The displayed rows are exactly $K, U, F, Q, E, H, C, P, N$, and each consumer-facing request begins with one of the compact windows named by K.

The convergence side is exhausted by U, Q, E. A uniform tail claim on a retained compact window must first read the shared modulus row U, then the regular-rational comparison row Q, and only then the terminal real handoff E.

The continuity side is exhausted by F together with the same compact-window route. The continuous-function rows are consumed on the windows already displayed by K, so endpoint continuity is replayed locally rather than by selecting a host function-space limit.

Transport, continuation, provenance, and local naming preserve this finite route. Since no carrier coordinate stores open-cover compactness, quotient equality of functions, countable choice, an ambient supremum norm, or an external limiting function selector, no local NameCert obligation may use such data. □

Theorem 4022.3 (LocallyUniformConvergence compact route)**.** *Every accepted local convergence read from a* $\mathsf{LocallyUniformConvergence}^{\uparrow}$ *carrier factors through the compact-window route*

$$K \longmapsto U, F \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Consequently the route consumes the existing $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{CFun}^{\uparrow}$, *and* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *packets only through displayed finite rows, not through open-cover compactness, quotient function-space equality, selected limits, countable choice, or ambient completion.*

Proof. Begin with a local convergence request. The request must name a compact metric window in K, so the carrier first fixes the finite portion of the domain on which convergence is being inspected.

Read the uniform and continuity rows on that same window. The uniform-modulus row U supplies the shared tail data, while the continuous-function row F records the approximating and endpoint continuity data that can be replayed on the retained window.

The comparison then passes through Q and E. The $\mathsf{RegSeqRat}^{\uparrow}$ row carries the rational tolerance comparisons, and the $\mathsf{Real}^{\uparrow}$ row is reached only after those comparisons and the compact-window data have been displayed.

Finally use H, C, P, N to transport, replay, source, and name the same route. The cited sibling packets provide the uniform-convergence, continuous-function, and compact-metric-completion rows consumed by the carrier, but none of those reads adds open-cover compactness, a selected global limit, quotient function equality, countable choice, or ambient completion. □

Falsifiable prediction: Deleting the compact-window row K prevents any accepted $\mathsf{LocallyUniformConvergence}^{\uparrow}$ local convergence read, even when uniform-modulus, continuous-function, regular-rational, real-handoff, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: uniform-convergence, continuous-function, compact-metric-completion: none is bijective to the locally uniform convergence packet because only $\mathsf{LocallyUniformConvergence}^{\uparrow}$ binds compact-window rows, uniform-modulus rows, continuous-function rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ handoff, transport, replay, provenance, and local naming into one accepted local convergence route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyUniformConvergence}^{\uparrow}$ packet is generated from compact-window rows, uniform-modulus rows, continuous-function rows, RegSeqRat readback, Real handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4022.1, Definition 4022.2, and Definition 4022.3 seed the finite compact-window route for locally uniform convergence over displayed uniform, continuity, completion, regular-rational, and real-handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient function-space equality, selected global limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window admission, uniform-modulus exactness, continuous-function compatibility, RegSeqRat readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocallyUniformConvergenceUp.

Chapter 4023

A Concrete Naming Certificate for $\mathsf{BishopMetricCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopMetricCompletion}^{\uparrow}$ packet records constructive metric completion by finite regular Cauchy data rather than by quotient completion objects. It is a sibling of chapter 1506, chapter 2114, and chapter 1612, and it routes its numerical names through chapter 126, chapter 301, and chapter 38. The packet names only the displayed metric source, dense embedding, regular Cauchy windows, separated comparison, transport, replay, provenance, and local NameCert rows.

Definition 4023.1 (Bishop metric completion carrier)**.** *A* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, E, Q, W, S, R, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the constructive metric source row, naming the source points, distance-read windows, and metric comparison ledger used by the completion request.*
2. *E is the dense embedding row from source names into the completion-facing name surface.*
3. *Q is the regular Cauchy family row, displayed as finitely inspectable* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks rather than as a quotient class.*
4. *W is the finite-window schedule, reading the required prefixes through* $\mathsf{StreamName}^{\uparrow}$*.*
5. *S is the separated comparison row, available only after two regular Cauchy windows and their tolerance rows have been displayed.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for distances and terminal comparison readbacks produced by the regular Cauchy family.*
7. *T is the transport row from source-window reads to completion-window reads.*
8. *H records componentwise* hsame *transport for the metric source, dense embedding, regular family, windows, separated comparison, Real seal, transport, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for source inspection, dense embedding, regular-window readback, separated comparison, Real sealing, and transport.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopMetricCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{LocatedMetricCompletion}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, Q, W, S, R, T, H, C, P, N$. It has no coordinate for quotient completions, selected representatives, countable-choice schedules, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopMetricCompletionUp.BishopMetricCompletionTasteGate single carrier alignment`

Theorem 4023.2 (Bishop metric completion NameCert obligations)**.** *For an accepted* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *packet $B = (M, E, Q, W, S, R, T, H, C, P, N)$, the local NameCert surface consists of carrier admission, dense-embedding exposure, regular Cauchy family admission, finite-window inspection, separated comparison, Real-seal handoff, source-to-completion transport, componentwise* hsame *stability, displayed* Cont *replay, provenance, and ledger non-escape. Every obligation is discharged through the displayed route*

$$M \longmapsto E \longmapsto Q \longmapsto W \longmapsto S \longmapsto R \longmapsto T$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.BishopMetricCompletionUp.BishopMetricCompletionCarrier namecert obligations`

Proof. Project the carrier of Definition 4023.1. The source metric row M and dense embedding row E must be visible before any regular Cauchy family Q is inspected.

The finite-window row W reads the regular family through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$. The separated comparison row S can then compare only displayed regular windows at the requested tolerance, and the $\mathsf{Real}^{\uparrow}$ seal row R records the resulting distance or comparison readback.

The transport row T keeps the source and completion windows on the same route. Componentwise transport through H and replay through C preserve those rows, while P and N source and name the packet.

Since quotient completions, selected representatives, countable-choice schedules, host equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4023.3 (Bishop metric completion finite-window handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *packet factors through the source metric row, dense embedding row, regular Cauchy family row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, separated comparison row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local NameCert rows displayed by B. The handoff exports a finite constructive completion surface, not a quotient completion, selected representative, countable-choice schedule, host equality principle, or ambient Real-completeness theorem.*

Proof. Begin with Definition 4023.2. It places every accepted read on the route $M, E, Q, W, S, R, T, H, C, P, N$.

Use the sibling completion chapters only through their visible rows. The metric-completion, located-metric-completion, and separated-completion surfaces supply completion vocabulary, located source comparison, and separated zero-distance comparison, but the Bishop packet receives them through M, E, Q, W, S, R, T and the structural rows.

The finite windows W and readbacks Q are the only access points to the regular Cauchy family. Therefore a downstream completion consumer can inspect a finite prefix and the corresponding separated comparison, but it cannot select a representative or collapse names by host equality.

The $\mathsf{Real}^{\uparrow}$ row R, transport row T, replay row C, provenance row P, and local name N close the handoff. No excluded object is a carrier coordinate, so the finite-window route is the whole exported completion surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopMetricCompletion}^{\uparrow}$ packet is generated from a constructive metric source row, dense embedding row, regular Cauchy family row, finite StreamName windows, RegSeqRat readbacks, separated comparison row, RealUp seal, source-to-completion transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4023.1, Definition 4023.2, and Definition 4023.3 seed the constructive metric-completion carrier, NameCert obligation surface, and finite-window handoff over MetricCompletion, LocatedMetricCompletion, SeparatedCompletion, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, dense-embedding exposure, regular Cauchy window admission, separated comparison, Real sealing, transport, replay, provenance, and local NameCert non-escape for $\mathsf{BishopMetricCompletion}^{\uparrow}$.

Chapter 4024

A Concrete Naming Certificate for $\mathsf{LocatedCompactness}^{\uparrow}$

Following Bishop–Bridges/Kelley-style located compactness, the $\mathsf{LocatedCompactness}^{\uparrow}$ packet records compact neighbourhood control by finite radius witnesses rather than by open-cover selection. It sits between the metric carrier of chapter 57, the compact-metric witness of chapter 129, the local support window of chapter 1726, the center-radius boundary of chapter 1451, and the finite observation schedule of chapter 126. Completion-facing consumers receive a finite located-neighbourhood interface, not a host compact subset, selected subcover, quotient completion, countable-choice schedule, or ambient real-completeness principle.

Definition 4024.1 (Located compactness carrier)**.** *A* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, K, B, U, W, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *source row, naming the carrier, distance witness, and local classifier used by every radius request.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness row for the displayed neighbourhood or closed-ball window.*
3. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row naming the center, radius, non-strict membership comparison, and radius ledger of the inspected window.*
4. *U is the* $\mathsf{LocallyCompact}^{\uparrow}$ *support row, recording that the compact witness is attached to a displayed local neighbourhood rather than to a global host compactness claim.*
5. *W is the located-neighbourhood radius row, recording the finite positive-radius requests, comparison outcomes, and compact-witness handoffs available to completion consumers.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row through which finite radius requests, compact-witness reads, and closed-ball inspections are scheduled.*
7. *H records componentwise* hsame *transport for metric source, compact witness, closed-ball boundary, local support, located-neighbourhood radius, stream schedule, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite stream windows through the closed-ball boundary, local compact support, compact metric witness, and located-radius handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten displayed rows. It has no coordinate for open-cover compactness, selected finite subcovers, quotient completion equality, choice-selected compact witnesses, host powersets, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4024.2 (Located compactness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier $L = (M, K, B, U, W, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$ *surface consists exactly of metric-source admission, compact-metric witness admission, closed-ball boundary exposure, local compact support, finite located-neighbourhood radius requests, finite* $\mathsf{StreamName}^{\uparrow}$ *scheduling, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must pass through $M, K, B, U, W, S, H, C, P, N$ before using located compactness.*

Lean4 ✓ `BEDC.Derived.LocatedCompactnessUp.LocatedCompactnessCarrier namecert obligations`

Proof. Project the carrier of Definition 4024.1. The ten rows displayed there are the only rows visible to the local classifier, so carrier admission is already confined to $M, K, B, U, W, S, H, C, P, N$.

Read the analytic rows componentwise. The metric source M supplies the carrier and distance witness, B supplies the center-radius boundary, U supplies the local compact support, K supplies the compact-metric witness, and W supplies the located-neighbourhood radius requests.

The scheduling and structural rows preserve the same finite route. Stream windows in S expose only finite requests, H transports those rows componentwise, C replays only the displayed handoff, and P, N source and name the packet. Since no additional coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 4024.3 (Located compactness finite-net handoff)**.** *Every L10 completion or compact-metric consumer of an accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier receives only the finite handoff*

$$\mathsf{Met}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{LocallyCompact}^{\uparrow},\quad \mathsf{MetricClosedBall}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$$

through the displayed rows $M, K, B, U, W, S, H, C, P, N$. The handoff provides finite neighbourhood-radius requests, compact-witness rows, closed-ball comparisons, local-support rows, and NameCert provenance; it exports no open-cover compactness, selected dense points, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the obligation surface of Definition 4024.2. A completion consumer first reads the metric row, closed-ball row, local compact support row, compact witness row, and located-radius row as one finite packet.

The finite request order is supplied by S. A radius or compact-witness inspection can occur only after a displayed $\mathsf{StreamName}^{\uparrow}$ window has exposed the request, and the replay row C sends that request through B, U, K, and W without adding a host compactness object.

Finally use the structural rows H, P, N. They transport, source, and name the same finite handoff for compact-metric and completion consumers. Because the packet has no row for open covers, selected dense points, quotient completions, countable choice, or ambient real completeness, the exported surface is exactly the finite located-neighbourhood handoff above. □

Theorem 4024.4 (Located compactness radius stability)**.** *For every accepted* $\mathsf{LocatedCompactness}^{\uparrow}$ *carrier* $L = (M, K, B, U, W, S, H, C, P, N)$*,* hsame *transport of the metric source, compact witness, closed-ball boundary, local support, located-radius request,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* Cont *replay,* Pkg *provenance, and local* NameCert *row preserves the finite located-neighbourhood handoff for every completion-facing consumer. The transported read remains the same finite route through*

$$M, K, B, U, W, S, H, C, P, N.$$

Lean4 ✓ `BEDC.Derived.LocatedCompactnessUp.LocatedCompactnessCarrier radius transport`

Proof. Project the transport row H from Definition 4024.1. It records componentwise transport for the metric source, compact witness, closed-ball boundary, local support, located-radius request, stream schedule, replay, provenance, and local naming rows.

Use Definition 4024.2 to identify those transported rows as the whole local NameCert surface. No transported completion-facing read can detach the radius request from $M, K, B, U, W, S, C, P, N$.

Finally apply the finite-net handoff of Definition 4024.3. The consumer route remains finite and choice-free, so transport preserves the located-neighbourhood handoff without introducing open-cover compactness, selected dense points, quotient completion equality, countable choice, or ambient real completeness. □

Falsifiable prediction: A completion-facing consumer that reads $\mathsf{LocatedCompactness}^{\uparrow}$ without displaying the metric source, compact witness, closed-ball boundary, local compact support, finite located-radius request, StreamName schedule, componentwise transport, continuation replay, package provenance, and local NameCert rows refutes the carrier surface.

Independence witness: metric, compactmetric, locallycompact, metricclosedball, streamname: no listed sibling is bijective to $\mathsf{LocatedCompactness}^{\uparrow}$ because this packet binds source metric data, compact witnesses, closed-ball boundaries, local support, located-radius requests, finite scheduling, transport, replay, provenance, and local naming in one completion-facing interface.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCompactness}^{\uparrow}$ packet is generated from metric source rows, compact-metric witnesses, closed-ball boundaries, local compact support rows, finite located-neighbourhood radius requests, StreamName schedules, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4024.1, Definition 4024.2, and Definition 4024.3 seed the finite located-neighbourhood compactness interface for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, quotient completion equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the ten-row carrier and prove the finite NameCert obligation surface for BEDC.Derived.LocatedCompactnessUp.

Chapter 4025

A Concrete Naming Certificate for $\mathsf{FourierTransform}^{\uparrow}$

Following Fourier and Stein–Shakarchi on the transform side of harmonic analysis, the $\mathsf{FourierTransform}^{\uparrow}$ packet records the finite transform-specific read that consumes the finite-observation $\mathsf{Fourier}^{\uparrow}$ surface of chapter 124, the integral witness rows of chapter 123, the inner-product probe rows of chapter 86, the real seal rows of chapter 38, and the product-tail handoff of chapter 409. It names transform-sample windows, kernel probes, integral witnesses, real seals, convolution handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming as finite BEDC rows. The chapter does not assert Plancherel, inversion, convergence closure, quotient equality of functions, or a selected ambient transform.

Definition 4025.1 (FourierTransform carrier)**.** *A* $\mathsf{FourierTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, K, I, R, V, H, C, P, N)$$

with the following displayed rows:

1. *W is the transform-sample window row, listing the finite source windows, frequency addresses, and retained sample requests read from the* $\mathsf{Fourier}^{\uparrow}$ *finite-observation surface.*
2. *K is the kernel/probe row, recording the displayed oscillatory kernel tags, inner-product probes, and source-domain probes used by the retained transform samples.*
3. *I is the integral-witness row, requiring each retained sample to name its* $\mathsf{Integral}^{\uparrow}$ *witness and the finite approximation ledger used by that witness.*
4. *R is the real-seal row, reached only after the transform-sample window, kernel/probe row, and integral-witness row have been replayed.*
5. *V is the convolution handoff row, reading* $\mathsf{CauchyConvolution}^{\uparrow}$ *only through the finite product-tail packet needed by convolution-facing transform consumers.*
6. *H records componentwise* hsame *transport for transform windows, kernel/probe rows, integral witnesses, real seals, convolution handoff rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite transform request through W, K, I, R, V.*
8. *P is the* Pkg *provenance over* $\mathsf{FourierTransform}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyConvolution}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{FourierTransform}^\uparrow}$ *row.*
The classifier compares accepted packets only through the component rows $W, K, I, R, V, H, C, P, N$. It has no coordinate for equality of host transforms, inversion, Plancherel, convergence closure, selected limiting integrals, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FourierTransformUp.FourierTransformUp`

Theorem 4025.2 (FourierTransform NameCert obligations)**.** *For every accepted* $\mathsf{FourierTransform}^\uparrow$ *carrier $T = (W, K, I, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FourierTransform}^\uparrow}$ *surface consists exactly of transform-sample window admission through W, kernel and probe admission through K, integral-witness readback through I, terminal* $\mathsf{Real}^\uparrow$ *seal admission through R, convolution handoff through V, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4025.1. Its public rows are exactly $W, K, I, R, V, H, C, P, N$, so every local naming duty begins with a finite transform-sample window rather than with a host transform.

The analytic source rows are then read in order. The row K exposes the kernel/probe data and inner-product probes, while I attaches each retained sample to an integral witness. The real seal row R is reached only after those rows have been displayed.

The convolution-facing part is confined to V. It imports only the finite product-tail handoff already named by $\mathsf{CauchyConvolution}^\uparrow$, and it does not add an inversion theorem, Plancherel identity, convergence closure, or quotient equality of functions.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores a selected ambient transform or a stronger analytic theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 4025.3 (FourierTransform convolution handoff)**.** *Every convolution-facing read from an accepted* $\mathsf{FourierTransform}^\uparrow$ *carrier factors through*

$$W, K, I \longmapsto R \longmapsto V \longmapsto H, C, P, N.$$

The read may consume $\mathsf{Fourier}^\uparrow$ *finite observations,* $\mathsf{Integral}^\uparrow$ *witnesses,* $\mathsf{InnerProduct}^\uparrow$ *probes,* $\mathsf{Real}^\uparrow$ *seals, and the* $\mathsf{CauchyConvolution}^\uparrow$ *product-tail handoff, but it exports only the finite transform-sample route displayed by the carrier.*

Proof. Start with the transform sample. The window row W fixes the retained source and frequency addresses, and the kernel/probe row K records the oscillatory kernel tags and inner-product probes used at those addresses.

Attach the integral and real rows next. The integral-witness row I supplies the finite approximation ledger for the retained sample, and the real-seal row R records the terminal real read obtained from that ledger.

Only after this route is displayed can the convolution row V be consumed. Its source is the finite product-tail handoff of $\mathsf{CauchyConvolution}^\uparrow$, so convolution-facing consumers receive a named handoff row rather than a host multiplication theorem or an equality of transforms.

The structural rows H, C, P, N preserve the same factorization under componentwise transport, replay, provenance, and local naming. Therefore the handoff exports the displayed transform-sample route only, with Plancherel, inversion, convergence closure, quotient function equality, and selected ambient transforms outside the packet. □

Falsifiable prediction: Deleting the transform-sample window row W prevents any accepted $\mathsf{FourierTransform}^{\uparrow}$ sample read, even when kernel/probe, integral-witness, real-seal, convolution handoff, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: Fourier finite observations, integral witnesses, inner-product probes, real seals, and Cauchy convolution handoffs are not interchangeable with the $\mathsf{FourierTransform}^{\uparrow}$ packet because only $\mathsf{FourierTransform}^{\uparrow}$ binds transform-sample windows, kernel/probe rows, integral witnesses, real seals, convolution handoff, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted transform route.

Closure status for $\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{FourierTransform}^{\uparrow}$ packet is generated from transform-sample windows, kernel/probe rows, integral witnesses, real seals, a convolution handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4025.1, Definition 4025.2, and Definition 4025.3 seed the finite transform-sample route over Fourier finite observations, integral witnesses, inner-product probes, real seals, and convolution handoff data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FourierTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Plancherel, inversion, convergence closure, quotient function equality, selected ambient transforms, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transform-window admission, kernel/probe exactness, integral-witness readback, real-seal handoff, convolution handoff, transport, replay, provenance, and local naming for BEDC.Derived.FourierTransformUp.

Chapter 4026

A Concrete Naming Certificate for $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$

Following Bishop and Bridges on completion and Mac Lane on adjunction triangular identities, the $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$ packet records the L10 completion handoff that keeps the completion functor, the regular Cauchy adjunction, the universal property, and the idempotence boundary visible in one finite certificate. It sits beside chapter 423, chapter 3882, chapter 2381, chapter 3376, and chapter 3888. The packet names a triangular completion route; it does not assert quotient completion equality, selected limits, host functor extensionality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4026.1 (CauchyCompletionAdjunctionTriangle carrier)**.** *A* $\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, A, U, I, R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row naming the source-to-completion functorial surface.*
2. *A is the* $\mathsf{CauchyRegularCompletionAdjunction}^{\uparrow}$ *row naming the regular Cauchy adjunction boundary.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row that exposes the extension and uniqueness handoff.*
4. *I is the completion idempotence row, read through the completion idempotence and metric completion idempotence chapters.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying regular rational completion readback.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window for the regular Cauchy route.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the displayed completion windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, A, U, I, R, W, D have been displayed.*
9. *H stores componentwise* hsame *transport for the functor, adjunction, universal-property, idempotence, regular readback, window, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay for the triangular route from functor and adjunction rows to universal handoff, idempotence boundary, and Real-facing seal.*

11. *P is the* Pkg *provenance over* CauchyCompletionAdjunctionTriangle$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CauchyRegularCompletionAdjunction$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$ *row.*

The classifier accepts only packets whose completion-facing read traverses the displayed rows $F, A, U, I, R, W, D, E, H$ It has no coordinate for quotient completions, selected limits, host equality of completed objects, unrestricted functor extensionality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.TasteGate.CauchyCompletionAdjunctionTriangleU`
`single carrier alignment`

Theorem 4026.2 (CauchyCompletionAdjunctionTriangle NameCert obligations)**.** *For every accepted* CauchyCompletionAdjunctionTriangle$^{\uparrow}$ *carrier $T = (F, A, U, I, R, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$ *obligations are exactly carrier admission, functor-row visibility, regular adjunction visibility, universal-property handoff, idempotence-boundary retention, regular-rational and stream-window readback, dyadic tolerance compatibility, Real seal nonescape, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleNamecertObl`

Proof. Project the carrier row by row from Definition 4026.1. Carrier admission exposes precisely the twelve displayed rows, so every local obligation has one visible coordinate.

Read the completion triangle in order. The functor row F and regular adjunction row A are available before the universal-property row U, and the idempotence row I is read as a boundary on reusing the same completion surface.

The Real-facing part is downstream. Rows R, W, D provide regular rational readback, finite windowing, and dyadic tolerance data before the terminal seal E is visible.

Finally, H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate supplies quotient completion equality, selected limits, host functor extensionality, countable choice, or ambient completeness, the displayed rows exhaust the local NameCert obligations. □

Theorem 4026.3 (CauchyCompletionAdjunctionTriangle universal handoff)**.** *Every accepted* CauchyCompletionAd *carrier hands a completion-facing consumer from the* CauchyCompletionFunctor$^{\uparrow}$ *row and the* CauchyRegularComplet *row to the* CompletionUniversalProperty$^{\uparrow}$ *row before any idempotence or Real-facing read is exported. The handoff is the finite route*

$$F, A \longmapsto U \longmapsto I, R, W, D, E, H, C, P, N$$

and not a quotient completion, selected limit, host equality of extensions, or ambient completeness assertion.

Proof. Begin with the functor and adjunction rows in the carrier. They are separate coordinates, so a consumer cannot read the universal-property row as a detached host theorem.

Project U. It is the only row that records extension and uniqueness handoff for the displayed completion route, and it is read after F and A have fixed the source completion surface.

The idempotence and Real-facing rows are downstream of that handoff. Row I records reuse of the same completion boundary, while R, W, D, E provide the finite regular-rational, window, dyadic, and Real seal route.

Rows H, C, P, N preserve the same finite path. Thus the universal handoff is exactly the displayed route and cannot become quotient completion equality, selected-limit data, host extension equality, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleUniversalHa`

Theorem 4026.4 (CauchyCompletionAdjunctionTriangle idempotence boundary)**.** *For every accepted* CauchyCompletionAdjunctionTriangle$^{\uparrow}$ *carrier, the idempotence read I is a boundary on reusing the already displayed completion route $F, A, U, R, W, D, E, H, C, P, N$. It cannot identify completed objects by host equality, erase the universal-property row, select a limit representative, or assert ambient completeness of* Real$^{\uparrow}$.

Proof. Open the accepted packet and isolate I. The carrier definition places I after the completion functor, regular adjunction, and universal-property rows, not as an independent completed-object equality.

Use the universal handoff theorem. It forces the route through F, A, U before I can be consumed by a completion-facing reader.

The remaining rows R, W, D, E, H, C, P, N show what the idempotence boundary may preserve: regular rational readback, finite observation windows, dyadic tolerances, the Real seal, transport, replay, provenance, and local naming. No row contains host equality of completions, selected representatives, erased universal handoff data, or ambient completeness. □

Falsifiable prediction: A completion consumer that exports a Cauchy-completion triangle without displaying the functor, regular adjunction, universal-property, idempotence, RegSeqRat, StreamName, DyadicRatCore, Real seal, transport, replay, provenance, and local naming rows is not reading a CauchyCompletionAdjunctionTriangle$^{\uparrow}$ packet.

Independence witness: cauchycompletionfunctor, cauchyregularcompletionadjunction, completionidempotence, cauchycompletionreflection, metriccompletionidempotence: CauchyCompletionAdjunctionTriangle is not bijective to any listed sibling because it jointly carries functor, adjunction, universal-property, idempotence, regular-rational, window, dyadic, Real seal, transport, replay, provenance, and local naming rows.

Closure status for NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$.

Constructive story: A CauchyCompletionAdjunctionTriangle$^{\uparrow}$ packet is generated from a CauchyCompletionFunctor row, a CauchyRegularCompletionAdjunction row, a CompletionUniversalProperty row, a completion idempotence row, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$, seed)

Scope closed: Definition 4026.1, Definition 4026.2, Definition 4026.3, and Definition 4026.4 seed the finite triangular handoff for the L10 Cauchy completion route.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCompletionAdjunctionTriangle}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.CauchyCompletionAdjunctionTriangleNamecertObligati`

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host functor extensionality, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, universal-property handoff, idempotence boundary, RegSeqRat and StreamName readback, DyadicRatCore compatibility, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionAdjunctionTriangleUp.

Chapter 4027

A Concrete Naming Certificate for $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ packet records the bounded monotone Cauchy criterion as finite BEDC data rather than as an ambient completeness principle. It sits after the monotone source surface of chapter 438, the bounded witness route of chapter 535, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes a displayed regular Cauchy route from bounded monotone data; it does not select a limit, quotient real names, import trichotomy, use open-cover compactness, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4027.1 (MonotoneCauchyCriterion carrier)**.** *A* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, S, T, M, D, L, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded monotone source row, including the inspected monotone observations and their located upper-bound guard.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite windows read from the source.*
3. *T is the tail-window schedule row that turns a dyadic request into a displayed tail window.*
4. *M is the monotone-tail witness row, recording that later observations stay inside the same bounded monotone route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for radii, slack, monotone comparisons, and upper-bound traps.*
6. *L is the located upper-bound trap row tying each selected tail window to a finite lower observation and a located upper guard.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window, tail, witness, ledger, and trap data.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational readback has been displayed.*
9. *H records componentwise* hsame *transport for the bounded source, schedule, tail window, monotone witness, dyadic ledger, located trap, readback, seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a criterion request through* $B, S, T, M, D, L, R, E, H, P, N$.
11. *P is the* Pkg *provenance over* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$, $\mathsf{MonotoneCauchy}^{\uparrow}$, $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, T, M, D, L, R, E, H, C, P, N$. *It has no coordinate for a selected limiting representative, quotient real equality, host real equality, trichotomy, open-cover compactness, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneCauchyCriterionUp.MonotoneCauchyCriterionTasteGate single carrier alignment`

Theorem 4027.2 (MonotoneCauchyCriterion NameCert obligations)**.** *For every accepted* MonotoneCauchyCriterio *carrier* $K = (B, S, T, M, D, L, R, E, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, bounded-window exposure through* B, S, T, *monotone-tail witness exposure through* M, *dyadic ledger exactness through* D, *located upper-bound trap exposure through* L, *regular-rational readback through* R, *Real seal nonescape through* E, *componentwise* hsame *transport through* H, *displayed* Cont *replay through* C, Pkg *provenance through* P, *and local naming through* N.

No obligation may be discharged by selected limits, quotient real equality, trichotomy, open-cover compactness, choice, host real equality, or ambient completeness.

Proof. Project the carrier of Definition 4027.1. The displayed rows are exactly $B, S, T, M, D, L, R, E, H, C, P, N$, so admission exposes only the bounded monotone source, finite schedules, tail witness, dyadic ledger, located trap, readback, seal, transport, replay, provenance, and naming data.

The bounded-window rows are B, S, T. They force every request to pass first through the inspected source and a finite scheduled tail window before any Cauchy-facing witness can be read.

The witness and numerical rows are M, D, L, R. The monotone-tail witness stays attached to the dyadic tolerance ledger and located trap, and the regular-rational readback is produced from that same finite route.

Finally the seal and structural rows E, H, C, P, N preserve and name the route. Since the excluded analytic objects are absent from the carrier, they cannot become local NameCert obligations. □

Theorem 4027.3 (MonotoneCauchyCriterion bounded-window route)**.** *Every criterion-facing read exported by an accepted* $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MonotoneCauchy}^{\uparrow} \longmapsto \mathsf{BoundedMonotoneCauchyWitness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \mapsto$$

The route first exposes bounded monotone source rows and finite tail windows, then reads the monotone-tail witness, dyadic tolerance ledger, located upper-bound trap, regular-rational readback, and terminal real seal. It exports no selected limit, quotient real equality, host real equality, trichotomy, open-cover compactness, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a criterion-facing read of $K = (B, S, T, M, D, L, R, E, H, C, P, N)$. Carrier projection exposes the bounded monotone source row B, the finite $\mathsf{StreamName}^{\uparrow}$ schedule S, and the tail-window row T before the witness row can be consumed.

Compose the bounded monotone witness route with the monotone Cauchy surface. The existing Lean touchpoint `BEDC.Derived.BoundedMonotoneCauchyWitnessUp.BoundedMonotoneCauchyWitnessCarrier_`

records the same source-to-tail-to-witness-to-ledger-to-seal chain at the finite BHist, Cont, and Pkg level.

The dyadic ledger D and located trap L then feed the regular-rational readback R. Only after R is displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal E be read, and H, C, P, N transport, replay, source, and name the same finite packet.

Because no excluded completion, quotient, selection, trichotomy, open-cover compactness, choice, or host-real coordinate occurs in the carrier, every exported criterion read is exhausted by the displayed bounded-window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 4027.1, Definition 4027.2, and Definition 4027.3 seed the bounded monotone Cauchy criterion route from MonotoneCauchyUp and BoundedMonotoneCauchyWitnessUp rows through StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, and the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneCauchyCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient real equality, trichotomy, open-cover compactness, choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window exposure, monotone-tail witness exposure, dyadic ledger exactness, located trap exposure, readback exactness, Real seal nonescape, transport, and replay proofs for BEDC.Derived.MonotoneCauchyCriterionUp.

Constructive story: A $\mathsf{MonotoneCauchyCriterion}^{\uparrow}$ packet is generated from bounded monotone source rows, finite StreamName windows, a tail-window schedule, a monotone-tail witness, a DyadicRatCore tolerance ledger, a located upper-bound trap, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4028

A Concrete Naming Certificate for $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ packet records paired regular-Cauchy difference estimates as finite rational-window data rather than as equality in a completed real quotient. It sits over the regular sequence route of chapter 301, the finite stream interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the tail envelope comparison route of chapter 3992. The packet supplies completion and Real algebra consumers with one finite difference-bound surface; it does not assert quotient equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4028.1 (Regular Cauchy difference-bound carrier)**.** *A* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, Y, W, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, read only through finite rational windows.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displayed on the same request schedule as X.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window family exposing the compared prefixes.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and subtraction ledger for the displayed window.*
5. *E is the* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *envelope row bounding the paired tails after W and D have been read.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed difference bound.*
7. *H records componentwise* hsame *transport for the two regular sources, the shared windows, the dyadic ledger, the envelope comparison, the Real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for window inspection, tolerance subtraction, tail-envelope comparison, Real sealing, and structural transport.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, E, R, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable-choice moduli, trichotomy, host subtraction, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundTasteGate single carrier alignment`

Theorem 4028.2 (Regular Cauchy difference-bound NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *packet $B = (X, Y, W, D, E, R, H, C, P, N)$, the local NameCert surface consists exactly of carrier admission for the two regular rational sources, shared finite-window inspection, dyadic difference budgeting, tail-envelope comparison, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming. Every accepted bound read follows*

$$X, Y \longmapsto W \longmapsto D \longmapsto E \longmapsto R$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundCarrier namecert obligations`

Proof. Project the carrier of Definition 4028.1. The two source rows X, Y are the only regular-rational inputs, and they are read on the shared window row W.

The dyadic ledger D supplies the displayed tolerance and subtraction budget for that same window. The tail-envelope row E can therefore compare only the paired tails already exposed by X, Y, W, D.

The terminal Real row R is downstream of the envelope comparison, while H, C, P, N only transport, replay, source, and name the same finite route. Since the carrier has no quotient equality, selected representative, countable-choice, host-subtraction, or ambient-completeness coordinate, the listed rows exhaust the local certificate. □

Theorem 4028.3 (Regular Cauchy difference-bound RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ *packet receives the paired difference estimate only through the displayed route from X, Y to W, D, E, R. The handoff preserves the finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{CauchyTailEnvelopeComparison}^{\uparrow}$ *envelope row, and it exports no quotient real equality, selected limit, countable-choice schedule, trichotomy, host real subtraction, or ambient completeness theorem.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceBoundUp.RegularCauchyDifferenceBoundRegSeqRatHandoff`

Proof. Begin with the NameCert obligations of Definition 4028.2. They place every accepted read on the route $X, Y, W, D, E, R, H, C, P, N$.

A $\mathsf{RegSeqRat}^{\uparrow}$ consumer can inspect only the two displayed regular sources and the finite windows in W. It must therefore read the same dyadic ledger D before any bound is available.

A $\mathsf{Real}^{\uparrow}$ consumer sees the terminal row R only after the tail-envelope comparison E has been consumed. The structural rows preserve this route, so no excluded equality, selector, choice, host-subtraction, or completeness object can be exported. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ read factors through two RegSeqRat source rows, a shared StreamName window row, a DyadicRatCore tolerance ledger, a CauchyTailEnvelopeComparison envelope row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchy-tail-envelope-comparison: $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired regular sources, shared finite windows, dyadic difference budgeting, tail-envelope comparison, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName window row, a DyadicRatCore tolerance and subtraction ledger, a CauchyTailEnvelopeComparison row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4028.1, Definition 4028.2, and Definition 4028.3 seed the finite paired regular-Cauchy difference-bound carrier, certificate surface, and RegSeqRat-to-Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceBound}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, host real subtraction, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, paired regular-source exposure, shared-window exactness, dyadic difference budgeting, tail-envelope comparison, Real sealing, transport, replay, provenance, and local NameCert non-escape for $\mathsf{RegularCauchyDifferenceBound}^{\uparrow}$.

Chapter 4029

A Concrete Naming Certificate for $\mathsf{LocallyCompactHausdorff}^{\uparrow}$

Following Kelley and Bourbaki on locally compact Hausdorff spaces, the $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ packet records the point-local compact-neighbourhood and point-separation rows that a constructive topology consumer may inspect without invoking open-cover compactness or choice of neighbourhood bases. It sits over the indexed-open surface of $\mathsf{Topology}^{\uparrow}$, the separated uniform route of chapter 3775, the compact-metric window of chapter 129, the formal-ball observation interface of chapter 1510, and the point-local metric compactness packet of chapter 3787.

Definition 4029.1 (Locally compact Hausdorff carrier)**.** *A* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, U, x, V, K, F, S, H, C, P, N)$$

with the following visible rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{HausdorffUniformSpace}^{\uparrow}$ *source row, including indexed open reads, separated neighbourhood reads, and the point classifier used by topology consumers.*
2. *U is a neighbourhood row around the inspected point x, displayed through the same topology or separated uniform source.*
3. *V is the compact neighbourhood inclusion row, witnessing that the inspected neighbourhood is carried by the compact window K.*
4. *K is a finite compactness window read through* $\mathsf{CompactMetric}^{\uparrow}$ *or a compact formal-ball window from* $\mathsf{FormalBall}^{\uparrow}$*.*
5. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *or finite-window replay row used when a consumer requests radius-indexed local observations inside K.*
6. *S is the Hausdorff separation row: distinct point reads are separated only by displayed neighbourhood or entourage witnesses over T.*
7. *H records componentwise* hsame *transport for the topology or uniform source, inspected point, local neighbourhood, compact window, formal-ball window, separation row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from local topology reads to compact-window, formal-ball, and Hausdorff-separation consumers.*

9. *P is the* Pkg *provenance over* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{HausdorffUniformSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{LocallyCompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyCompactHausdorff}^{\uparrow}}$ *row.*

The packet has no coordinate for arbitrary open-cover compactness, selected neighbourhood bases, quotient point equality, excluded middle, countable choice, or ambient local compactness outside the displayed carrier.

Lean4 ✓ `BEDC.Derived.LocallyCompactHausdorffUp.LocallyCompactHausdorffTasteGate single carrier alignment`

Theorem 4029.2 (LocallyCompactHausdorff carrier admission)**.** *For every accepted* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ *packet $L = (T, U, x, V, K, F, S, H, C, P, N)$, carrier admission is rowwise over the displayed topology or separated-uniform source, inspected point, local neighbourhood row, compact-window inclusion, formal-ball or compact-metric finite window, Hausdorff separation row, componentwise transport, replay, provenance, and local name. Admission cannot be obtained from an unrecorded host topology, an open-cover compactness theorem, a selected basis, quotient point equality, or ambient local compactness.*

Proof. Project the carrier of Definition 4029.1. The packet exposes exactly $T, U, x, V, K, F, S, H, C, P, N$, so every accepted read begins with the topology or separated-uniform source and the inspected point.

Next read the compact side. The row V ties the neighbourhood U to the compact window K, while F supplies only the finite formal-ball or compact-metric observations cited by chapter 129 and chapter 1510.

Finally read the structural rows H, C, P, N. They transport, replay, record provenance, and bind the local name for the same finite packet. Since the carrier contains no host topology, open-cover compactness, selected basis, quotient equality, or ambient local-compactness coordinate, none can enter through carrier admission. □

Theorem 4029.3 (LocallyCompactHausdorff compact-neighbourhood nonescape)**.** *Every local compact-neighbourhood read of an accepted* $\mathsf{LocallyCompactHausdorff}^{\uparrow}$ *packet factors through*

$$T, U, x, V, K, F \longmapsto H, C, P, N.$$

The read may use compact-metric and formal-ball windows, and may compare with the $\mathsf{LocallyCompactMetric}^{\uparrow}$ *point-local route when a metric presentation is displayed. When completion-facing readers consume the local compact neighbourhood, the compact-window row may be refined through the finite located-neighbourhood radius handoff of chapter 4024, not through open-cover compactness. The read cannot promote the window to arbitrary open-cover compactness, a global compactness theorem, a selected local base, quotient completion equality, or ambient completeness.*

Proof. Begin with the neighbourhood rows T, U, x, V. They name the source topology or separated uniform space, the inspected point, and the local neighbourhood inclusion before any compactness consumer can inspect the packet.

Use the compact-window rows K and F. The compact metric certificate of chapter 129 and the formal-ball interface of chapter 1510 provide finite windows only; the metric local-compactness sibling chapter 3787 is read as a displayed point-local comparison, not as an ambient theorem.

Finally replay through H, C, P, N. These rows keep the compact neighbourhood inside the same finite carrier route, so open-cover compactness, selected neighbourhood bases, quotient completion equality, global compactness, and ambient completeness have no row from which to escape. □

Theorem 4029.4 (LocallyCompactHausdorff separation transport)**.** *For every accepted* LocallyCompactHausdorff$^{\uparrow}$ *packet, Hausdorff separation is transported only through the displayed row S over the same topology or separated-uniform source T and the same structural route H, C, P, N. If two point reads are moved by* hsame*, their separated-neighbourhood or separated-entourage witnesses are moved componentwise with the compact-neighbourhood rows; no host equality, quotient identification, global apartness principle, or classical point separation is added.*

Proof. Start with the separated source. When T is read through HausdorffUniformSpace$^{\uparrow}$, the separated-uniform packet supplies the visible separation route; when it is read through Topology$^{\uparrow}$, the row S displays the neighbourhood witnesses required for the inspected point reads.

Apply componentwise transport H. The transport row moves the source, neighbourhood, compact-window, formal-ball, and separation data together, so a separated read cannot be detached from the local compact carrier.

Finally use replay and provenance, C, P, N. They record the same separation route for downstream consumers and keep quotient equality, host point equality, global apartness, and classical separation outside the packet. □

Closure status for NameCert$_{\text{LocallyCompactHausdorff}^{\uparrow}}$.

Constructive story: A LocallyCompactHausdorff$^{\uparrow}$ packet is generated from a topology or separated-uniform source, an inspected point, a local neighbourhood row, a compact-metric or formal-ball window, a Hausdorff separation row, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{LocallyCompactHausdorff}^{\uparrow}}$, seed)

Scope closed: Definition 4029.1, Definition 4029.2, Definition 4029.3, and Definition 4029.4 seed the constructive local-compactness and Hausdorff-separation packet over topology or separated-uniform sources.

Formal verification: FormalStatus(NameCert$_{\text{LocallyCompactHausdorff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, arbitrary neighbourhood bases, quotient completion equality, global local compactness, classical separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-neighbourhood nonescape, Hausdorff separation transport, finite-window replay, provenance, and local naming for BEDC.Derived.LocallyCompactHausdorffUp.

Chapter 4030

A Concrete Naming Certificate for $\mathsf{GleasonCover}^{\uparrow}$

Following Gleason and Kelley on projective covers of compact Hausdorff spaces, the $\mathsf{GleasonCover}^{\uparrow}$ packet records finite projective-cover, clopen-lift, compactification, and provenance rows. It sits beside the compact-target seal of chapter 2344, the lower-hit and upper-miss hyperspace route of chapter 2375, the local separation and compact-neighbourhood rows of chapter 4029, and the finite compact-metric witness surface of chapter 129. The packet names the traditional projective-cover boundary through displayed BEDC data; it does not import choice-selected irreducible maps, host Boolean algebras, quotient compacta, open-cover compactness, or ambient compact Hausdorff equality.

Definition 4030.1 (Gleason cover carrier)**.** *A* $\mathsf{GleasonCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, E, M, L, B, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact-Hausdorff source row, read through the displayed compact target seal, topology rows, and separation rows already available to compactification consumers.*
2. *E is the extremally disconnected projective-cover row, recording that the cover source is inspected only through finite projective-extension requests.*
3. *M is the cover-map row from E to X, including the finite surjective-read and irreducibility-test windows exposed to downstream compact Hausdorff consumers.*
4. *L is the clopen-lift row, recording finite clopen lift requests and their replay through the same cover map.*
5. *B is the compactification comparison row, comparing the cover map with the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *probe and compact-target rows.*
6. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *closed-set row, recording the finite lower-hit and upper-miss observations used when compact-subset consumers inspect the cover.*
7. *H records componentwise* hsame *transport for the compact-Hausdorff source, projective-cover row, cover-map row, clopen-lift row, compactification comparison row, Vietoris closed-set row, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from compact Hausdorff source reads through projective-cover, clopen-lift, compactification, and closed-set consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{GleasonCover}^{\uparrow}$, $\mathsf{StoneCechCompactification}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{LocallyCompactHausdorff}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X, E, M, L, B, V, H, C, P, N$. It has no coordinate for arbitrary projective resolutions, selected ultrafilters, quotient equality of compacta, open-cover compactness, choice of clopen bases, or host compact Hausdorff spaces outside the displayed packet.

Lean4 ✓ `BEDC.Derived.GleasonCoverUp.GleasonCoverTasteGate single carrier alignment`

Theorem 4030.2 (Gleason cover projective-cover handoff)**.** *Every compact-Hausdorff consumer of an accepted* $\mathsf{GleasonCover}^{\uparrow}$ *carrier reads the cover only through the finite route*

$$X, E, M, L, B, V \longmapsto H, C, P, N.$$

The route exposes the compact-Hausdorff source, extremally disconnected projective-cover row, cover-map row, clopen-lift row, compactification comparison row, Vietoris closed-set row, componentwise transport, displayed replay, provenance, and local naming. It exports no open-cover selection, quotient compactum equality, selected ultrafilter, host Boolean algebra, arbitrary clopen basis, or ambient projective-cover theorem.

Proof. Project the carrier of Definition 4030.1. The compact-Hausdorff source is the row X, the projective-cover information is carried by E and M, and the clopen-lift and compactification comparisons are the rows L and B.

Read closed-set observations through V. This is the finite $\mathsf{VietorisTopology}^{\uparrow}$ side of the packet, so lower-hit and upper-miss consumers inspect only displayed closed-set windows rather than a host hyperspace or quotient compactum.

Finally replay through H, C, P, N. These rows transport, continue, source, and name the same carrier, so a compact-Hausdorff consumer has no coordinate from which to obtain open-cover selection, quotient equality, selected ultrafilters, host Boolean algebras, arbitrary clopen bases, or an ambient projective-cover theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$.

Constructive story: A $\mathsf{GleasonCover}^{\uparrow}$ packet is generated from compact-Hausdorff source rows, extremally disconnected projective-cover rows, cover-map rows, clopen-lift rows, compactification comparisons, Vietoris closed-set rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$, seed)

Scope closed: Definition 4030.1 and Definition 4030.2 seed the finite projective-cover handoff surface for compact-Hausdorff consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GleasonCover}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary projective resolutions, selected ultrafilters, quotient compactum equality, open-cover compactness, choice of clopen bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, projective-cover handoff, clopen-lift replay, compactification comparison, Vietoris closed-set compatibility, provenance, and local naming for BEDC.Derived.GleasonCoverUp.

Chapter 4031

A Concrete Naming Certificate for $\mathsf{HeineBorelNetCompactness}^{\uparrow}$

Following Bishop and Bridges on compact intervals, finite approximation, and located covers, the $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ packet records the finite-net compactness route for closed bounded real intervals as displayed NameCert data. It sits next to the finite-net cover surface of chapter 3471, the located subcover handoff of chapter 3538, the closed bounded interval compactness packet of chapter 3969, the compact metric carrier of chapter 129, the real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The packet gives a finite tolerance-and-net surface to completion and compactness consumers, not an unrestricted open-cover compactness theorem or a choice-selected subcover principle.

Definition 4031.1 (Heine–Borel net compactness carrier)**.** *A* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, F, R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located closed bounded interval row, recording the endpoint order, interval membership classifier, and interval-source readback.*
2. *F is the finite net row, read through* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$*, naming the displayed centers, mesh radius, cell coverage witnesses, and compact-metric provenance.*
3. *R is the positive-radius ledger, recording the requested positive tolerance, its dyadic lower witness, and the comparison surface used by cover readers.*
4. *W is the cover-read witness row, exposing only the finitely many cover predicates inspected at the displayed centers and interval boundary cells.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare mesh radius, endpoint error, cover radius, and finite-net budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for interval endpoints and inspected cover radii, reached only after I, F, R, W, D have been displayed.*
7. *H records componentwise* hsame *transport for interval source, finite net, positive-radius ledger, cover reads, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the located interval row through finite nets, positive-radius checks, cover reads, dyadic tolerance comparisons, and terminal Real seals.*

9. *P is the* Pkg *provenance over* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, R, W, D, E, H, C, P, N$. It has no coordinate for unrestricted open-cover compactness, choice-selected finite subcovers, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HeineBorelNetCompactnessUp.HeineBorelNetCompactnessTasteGate single carrier alignment`

Theorem 4031.2 (Heine–Borel net compactness obligations)**.** *For every accepted* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ *carrier $K = (I, F, R, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$ *surface consists exactly of located interval admission, finite-net admission, positive-radius ledger exposure, finite cover-read witnesses, dyadic tolerance readback, Real-seal readback, componentwise transport, displayed continuation replay, provenance, and local naming. Every compactness-facing read must pass through the displayed rows before using a finite-net compactness conclusion.*

Proof. Project the carrier of Definition 4031.1. The tuple displays exactly the rows $I, F, R, W, D, E, H, C, P, N$, so the local classifier has no access to data outside that packet.

The interval and finite-net sides are read first. The located interval row I supplies the endpoint and membership surface, while F supplies the finite centers and coverage witnesses inherited from the finite-net and compact-metric siblings.

The cover-facing side is finite as well. The positive-radius ledger R fixes the requested tolerance, W records the inspected cover witnesses, D gives the dyadic comparison budget, and E seals only the endpoint and radius reads that are already attached to those rows.

Finally H, C, P, N transport, replay, source, and name the same route. Since no row contains open covers as an unrestricted object, selected subcovers, quotient interval equality, countable choice, or ambient real completeness, the listed rows exhaust the NameCert obligations. □

Theorem 4031.3 (Heine–Borel finite-subcover route)**.** *Every* $\mathsf{LocatedHeineBorelSubcover}^{\uparrow}$, ClosedBoundedIntervalC *or* $\mathsf{CompactMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{HeineBorelFiniteNet}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$$

through the displayed rows $I, F, R, W, D, E, H, C, P, N$. The route exports finite centers, positive-radius comparisons, cover-read witnesses, dyadic tolerance rows, endpoint and radius seals, and structural provenance only; it exports no unrestricted open-cover compactness, choice-selected subcover, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface of Definition 4031.2. The interval row and finite-net row must be displayed before a compactness consumer can inspect a cover witness.

Next read the tolerance side. The positive-radius ledger R and dyadic tolerance row D control the requested mesh and cover radius, while W exposes only the finite cover reads attached to the displayed centers and interval boundary cells.

The terminal Real row E seals those endpoint and radius reads without adding a completeness theorem. The transport, replay, provenance, and local naming rows H, C, P, N preserve the same finite route for each sibling consumer.

Therefore the public handoff is exactly the finite-subcover route displayed in the statement. The excluded open-cover, selected-subcover, quotient, choice, and ambient-completeness data have no carrier row through which they could be exported. □

Falsifiable prediction: A compactness-facing consumer that reads $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ without displaying the located interval, finite net, positive-radius ledger, cover-read witness, DyadicRatCore tolerance, Real seal, componentwise transport, continuation replay, package provenance, and local NameCert rows refutes the carrier surface.

Independence witness: heineborel-finite-net, located-heine-borel-subcover, closed-bounded-interval-compact, compactmetric, real, dyadicratcore: no listed sibling is bijective to $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ because this packet binds interval location, finite-net coverage, positive-radius ledgers, cover-read witnesses, dyadic tolerance, Real seals, transport, replay, provenance, and local naming in one compactness-facing interface.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelNetCompactness}^{\uparrow}$ packet is generated from a located interval row, finite net row, positive-radius ledger, cover-read witnesses, DyadicRatCore tolerance row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4031.1, Definition 4031.2, and Definition 4031.3 seed the finite-net compactness interface for closed bounded interval consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelNetCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted open-cover compactness, choice-selected finite subcovers, quotient interval equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net admission, positive-radius ledger exposure, cover-read witnesses, dyadic tolerance readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelNetCompactnessUp.

Chapter 4032

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionUnit}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ packet records the completion-unit surface for separated Hausdorff completion as finite NameCert data. It sits after the uniform-completion carrier of chapter 1473, the metric-completion carrier of chapter 1506, the Hausdorff-completion carrier of chapter 402, the separated-metric boundary of chapter 1411, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names the unit map into the separated Hausdorff completion without importing quotient completion equality, selected representatives, or ambient completeness.

Definition 4032.1 (HausdorffCompletionUnit carrier)**.** *A* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, S, J, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric or uniform row, naming the finite observation surface consumed before completion.*
2. *S is the separated Hausdorff completion row, read through the separated-metric and Hausdorff-completion sibling carriers.*
3. *J is the unit-map row from the source surface into the separated Hausdorff completion surface.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect source names before the unit map is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular Cauchy tolerance names attached to W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for distance or apartness readbacks reached after M, S, J, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the source, completion, unit, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from source inspection to unit-map handoff, completion readback, and real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion points, or ambient completeness.

Lean4 (definition) `BEDC.Derived.HausdorffCompletionUnitUp.HausdorffCompletionUnitUp`

Theorem 4032.2 (HausdorffCompletionUnit NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *carrier* $U = (M, S, J, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$ *obligations are exactly source metric or uniform exposure through* M*, separated Hausdorff completion exposure through* S*, unit-map exposure through* J*,* $\mathsf{StreamName}^{\uparrow}$ *finite-window inspection through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Every obligation is discharged along the displayed route

$$M \longmapsto S \longmapsto J \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No obligation may be replaced by quotient completion equality, selected representatives, countable choice, host equality of completion points, or ambient completeness.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionUnitUp.HausdorffCompletionUnitNamecertObligations`

Proof. Project the carrier of Definition 4032.1. The source row M, separated Hausdorff completion row S, unit row J, finite windows W, readback row R, seal row E, transport row H, replay row C, provenance row P, and local name N are exactly the displayed coordinates.

Read the completion-unit route in order. The source metric or uniform row is visible before the separated Hausdorff completion row, and the unit-map row is read only after both source and completion surfaces have been displayed.

The numerical side is downstream. The $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readbacks expose finite regular Cauchy data before the $\mathsf{Real}^{\uparrow}$ seal is consumed. Rows H, C, P, N transport, replay, source, and name that same finite route.

Since the packet contains no quotient equality, selected representative, choice schedule, host completion-point equality, or ambient-completeness coordinate, the displayed rows exhaust the local NameCert obligations. □

Theorem 4032.3 (HausdorffCompletionUnit Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ *packet reads the source metric or uniform row, separated Hausdorff completion row, unit-map row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local NameCert rows before receiving the completion handoff. The handoff is the finite unit route*

$$M, S, J, W, R, E, H, C, P, N$$

and not a quotient completion class, selected representative, countable-choice schedule, host equality principle, or ambient complete-space witness.

Proof. Begin with the NameCert obligations of Definition 4032.2. They force the consumer to read $M, S, J, W, R, E, H, C, P, N$ as one displayed certificate surface.

Use the sibling dependencies through their visible rows. The uniform-completion, metric-completion, Hausdorff-completion, and separated-metric chapters supply completion vocabulary and separated comparison, but the unit packet receives those surfaces only through M, S, J and then through the finite window and readback rows.

The Cauchy-facing data remains finite. The $\mathsf{StreamName}^{\uparrow}$ row displays the inspected windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row displays the tolerance readback, and the $\mathsf{Real}^{\uparrow}$ seal is terminal rather than a detached completeness principle.

Finally H, C, P, N preserve the same route under transport, replay, provenance, and naming. No excluded coordinate is present in the carrier, so the completion-facing handoff is exactly the finite unit route. □

Falsifiable prediction: Every $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ completion-unit read factors through source metric or uniform data, separated Hausdorff completion data, the unit-map row, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming. A read that reaches the completion handoff without $M, S, J, W, R, E, H, C, P, N$ falsifies this packet.

Independence witness: uniformcompletion, metriccompletion, hausdorffcompletion, separated-metric, streamname, regseqrat, real: $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source surface, separated Hausdorff completion surface, unit-map row, finite windows, regular-rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffCompletionUnit}^{\uparrow}$ packet is generated from a source metric or uniform row, separated Hausdorff completion row, unit-map row, StreamName finite windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4032.1, Definition 4032.2, and Definition 4032.3 seed the finite completion-unit surface for separated Hausdorff completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionUnit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, separated Hausdorff completion exposure, unit-map handoff, StreamName and RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HausdorffCompletionUnitUp.

Chapter 4033

A Concrete Naming Certificate for $\mathsf{CauchyCriterionForSeries}^{\uparrow}$

Following Cauchy and Knopp, the $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ packet records the constructive criterion that a real series converges exactly when its finite tails become arbitrarily small under displayed regular-rational windows. It sits after the series carrier of chapter 1581, the Cauchy sequence algebra of chapter 3708, the stream-name window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite tail route for series convergence; it is not a host infinite sum, quotient stream equality, selected limit, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4033.1 (Cauchy criterion for series carrier)**.** *A* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, L, W, D, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, displaying the inspected series terms and partial-sum prefix.*
2. *L is the* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *row, naming the partial-sum algebra and its finite sum/difference operations.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite tail interval and every inspected partial-sum index.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the positive dyadic radius used by the tail comparison.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed tail window and dyadic tolerance.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the series, algebra, window, tolerance, and readback rows have been replayed.*
7. *T is the tail-smallness ledger, recording the finite witnesses that all displayed tails beyond the named threshold are inside the tolerance in D.*
8. *H records componentwise* hsame *transport for A, L, W, D, R, E, T and for the continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a convergence request through the source series, partial-sum algebra, finite window, dyadic tolerance, regular rational readback, tail-smallness ledger, and real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, L, W, D, R, E, T, H, C, P, N$. It has no coordinate for an infinite host sum, quotient equality of partial-sum streams, selected limit, countable-choice modulus, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionForSeriesUp.CauchyCriterionForSeriesTasteGate single carrier alignment`

Theorem 4033.2 (Cauchy criterion for series NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *carrier $S = (A, L, W, D, R, E, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure,* $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ *partial-sum algebra exposure,* $\mathsf{StreamName}^{\uparrow}$ *tail-window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, tail-smallness ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4033.1. The tuple displays the real-series source, partial-sum algebra, finite windows, dyadic tolerance, regular rational readback, real seal, tail-smallness ledger, transport, continuation, provenance, and local name.

The analytic content is read through A, L, W, D, R, E, T. The series row supplies terms and partial sums; the algebra row supplies finite partial-sum differences; the window and tolerance rows state the inspected tail request; and the readback, seal, and ledger rows certify that the displayed tail is small.

The structural rows H, C, P, N preserve and replay the same route. Since the carrier contains no coordinate for host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient completeness, or bridge checking, the listed rows exhaust the local certificate. □

Theorem 4033.3 (Cauchy criterion for series tail route)**.** *Every accepted convergence read of a* $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ *packet factors through the finite tail route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{CauchySequenceAlgebra}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported criterion is the assertion that the partial-sum tail ledger T can answer each displayed dyadic tolerance request after the source series, partial-sum algebra, finite window, regular readback, and real seal have been displayed. It exports no host infinite sum, quotient partial-sum stream, selected limit, countable-choice modulus, trichotomy, or ambient completeness theorem.

Proof. Begin with the source series row A and partial-sum algebra row L. These rows provide only finite term and partial-sum data, so a convergence consumer has not yet received a completed sum.

Next read the finite request rows W and D. The stream-name row displays the inspected tail window, and the dyadic row names the tolerance for that window.

The regular rational readback R, real seal E, and tail-smallness ledger T then certify the displayed tail comparison. The continuation row C fixes this order, and H, P, N transport, source, and name the same packet.

Because every consumer reaches the criterion through this route, any read that asks for quotient stream equality, selected limits, countable choice, trichotomy, or ambient real completeness has left the packet and is not part of the exported criterion. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ convergence read must expose a $\mathsf{RealSeries}^{\uparrow}$ source row, $\mathsf{CauchySequenceAlgebra}^{\uparrow}$ partial-sum algebra row, $\mathsf{StreamName}^{\uparrow}$ tail window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, tail-smallness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before the criterion is available.

Independence witness: Nearest siblings: realseries, cauchysequencealgebra, streamname, dyadicratcore, regseqrat, and real. The packet is not bijective to any one sibling because it jointly binds source series rows, partial-sum algebra, tail windows, dyadic tolerances, regular rational readback, real sealing, tail-smallness ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCriterionForSeries}^{\uparrow}$ packet is generated from a RealSeries source row, CauchySequenceAlgebra partial-sum row, StreamName tail-window row, DyadicRatCore tolerance row, RegSeqRat readback, RealUp seal, tail-smallness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4033.1, Definition 4033.2, and Definition 4033.3 seed the finite constructive Cauchy criterion route for real series.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCriterionForSeries}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host infinite sums, quotient stream equality, selected limits, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, partial-sum algebra exposure, tail-window exactness, dyadic tolerance exactness, RegSeqRat readback, RealUp seal handoff, tail-smallness ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCriterionForSeriesUp.

Chapter 4034

A Concrete Naming Certificate for HyperbolicHarmonicBoundaryMeasure$^{\uparrow}$

The HyperbolicHarmonicBoundaryMeasure$^{\uparrow}$ packet realizes the boundary-weighting route of chapter 9463 as finite NameCert data. It sits after the Poisson-kernel certificate of chapter 3925, the visual-metric certificate of chapter 3866, the Busemann-boundary certificate of chapter 3732, the hyperbolic-metric certificate of chapter 3474, the disk carrier of chapter 377, and the terminal real-name seal of chapter 38. The packet names a finite harmonic boundary-measure read; it is not a host measure space, sigma-algebra completion, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary construction.

Definition 4034.1 (Hyperbolic harmonic boundary measure carrier)**.** *A* HyperbolicHarmonicBoundaryMeasure$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, B, V, K, R, W, H, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-coordinate row naming the finite interior observation from which boundary weighting is read.*
2. *B is the* HyperbolicBusemannBoundary$^{\uparrow}$ *row naming the boundary face and its Busemann-level comparison data.*
3. *V is the* HyperbolicVisualMetric$^{\uparrow}$ *row naming the visual boundary distance used by the weighting request.*
4. *K is the* HyperbolicPoissonKernel$^{\uparrow}$ *row, displaying the Poisson-kernel weight over the same disk, boundary, visual-metric, and hyperbolic-metric rows.*
5. *R is the* Real$^{\uparrow}$ *seal row for the displayed boundary weight.*
6. *W is the harmonic boundary-measure ledger, recording the finite boundary-weight rows accepted by the Poisson kernel before any measure read is exported.*
7. *H records componentwise* hsame *transport for D, B, V, K, R, W and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from disk-coordinate admission through Busemann-boundary admission, visual-metric readback, Poisson-kernel weighting, real sealing, and harmonic boundary-measure ledger exposure.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$, $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, B, V, K, R, W, H, C, P, N$. It has no coordinate for a host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicHarmonicBoundaryMeasureUp.HyperbolicHarmonicBoundaryMeasureTasteGate`
`single carrier alignment`

Theorem 4034.2 (Hyperbolic harmonic boundary measure NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ *carrier $M = (D, B, V, K, R, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, Busemann-boundary exposure, visual-metric readback, Poisson-kernel weighting,* $\mathsf{Real}^{\uparrow}$ *seal handoff, harmonic boundary-measure ledger exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4034.1. The projection exposes the disk row, boundary row, visual-metric row, Poisson-kernel row, real seal, harmonic boundary-measure ledger, and the structural rows.

The geometric read is ordered by D, B, V, K, R, W. A boundary-measure consumer first sees the disk coordinate, then the Busemann boundary face, then the visual metric, and only then the Poisson-kernel weight and real seal used by the ledger.

The rows H, C, P, N transport, replay, source, and name this finite route. Since the carrier contains no host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, Lean verification, or bridge checking, the listed rows exhaust the NameCert surface. □

Theorem 4034.3 (Hyperbolic harmonic boundary measure Poisson route)**.** *Every accepted harmonic boundary-measure read of a* $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto \mathsf{HyperbolicVisualMetric}^{\uparrow} \longmapsto \mathsf{HyperbolicPoi}$$

The exported boundary measure is the Poisson-kernel-weighted ledger W over the same disk and boundary rows. It exports no completed host measure, sigma-algebra completion, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary object.

Proof. Open the disk-coordinate row D and the Busemann-boundary row B. These rows fix the finite interior and boundary observations before any weight is inspected.

Read the visual metric and Poisson kernel next. The row V supplies the boundary-distance comparison, and K supplies the Poisson-kernel weight over the same hyperbolic metric and boundary route.

The real seal R and measure ledger W then record the displayed harmonic boundary weight. The rows H, C, P, N preserve the route under transport, replay it for consumers, record provenance, and attach the local name.

Thus every accepted measure read passes through the Poisson-kernel route. The carrier has no coordinate from which a host measure space, completed sigma algebra, selected boundary representative, ambient probability theorem, global field instance, or quotient boundary construction could be exported. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ read must pass through $\mathsf{HyperbolicPoissonKernel}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before it exports a boundary-weight ledger.

Independence witness: Nearest siblings: hyperbolic-poisson-kernel, hyperbolic-visual-metric, hyperbolic-busemann-boundary, hyperbolic-metric, and unitdisk. The packet is not bijective to any one sibling because it jointly binds disk-coordinate rows, Busemann boundary rows, visual metric readback, Poisson-kernel weighting, RealUp sealing, harmonic boundary-measure ledger exposure, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}$ packet is generated from UnitDisk disk-coordinate rows, HyperbolicMetric route data, HyperbolicBusemannBoundary rows, HyperbolicVisualMetric rows, HyperbolicPoissonKernel weights, a RealUp seal, a harmonic boundary-measure ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4034.1, Definition 4034.2, and Definition 4034.3 seed the finite harmonic boundary-measure route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHarmonicBoundaryMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host measure spaces, completed sigma algebras, selected boundary representatives, ambient probability theorems, global field instances, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-coordinate exposure, Busemann-boundary exposure, visual-metric readback, Poisson-kernel weighting, RealUp seal handoff, ledger stability, transport, replay, provenance, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.HyperbolicHarmonicBoundaryMeasureUp.

Chapter 4035

A Concrete Naming Certificate for $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$

The $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ packet realizes the support-nerve census and codon-lifted Betti table of subsection 9462.28.21 as finite NameCert data. It consumes the partition-core incidence surface of chapter 4017, the simplicial-complex carrier of chapter 238, the homology carrier of chapter 98, the finite simplicial-chain surface of chapter 3414, the Hochster–nerve reduction of chapter 3744, and the DNA boundary-tile source of chapter 3393. The packet names a finite support-to-nerve realization and lifted Betti census; it is not a host minimal-resolution oracle, stochastic biological model, quotient codon semantics, unrestricted homology computation, or coefficient-field completeness theorem.

Definition 4035.1 (SupportNerveCensusLiftedBetti carrier)**.** *A* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, U, V, F, H_0, B, L, G, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{PartitionCoreIncidence}^{\uparrow}$ *row naming the finite support-family incidence table used to enumerate the* 15 *nonempty supports.*
2. *U is the support-to-nerve realization row, assigning each accepted support family to the four-vertex nerve obtained from its displayed supports.*
3. *V is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *row naming the resulting nerve as a finite complex.*
4. *F is the* $\mathsf{FiniteSimplicialChain}^{\uparrow}$ *row exposing the chain boundary surface used for reduced homology reads.*
5. *H_0 is the* $\mathsf{Homology}^{\uparrow}$ *row for reduced homology in degrees* $-1, 0, 1, 2$.
6. *B is the lifted Betti ledger carrying the finite census counts and the codon lift substitution* $z \mapsto 2x + x^2$.
7. *L is the* $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$ *row that keeps the support-level nerve census upstream of any Betti-strand export.*
8. *G records componentwise* hsame *transport for the incidence, realization, complex, chain, homology, lifted Betti, reduction, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from support-family reads to nerve realization, finite homology, and lifted Betti ledgers.*
10. *P is the* Pkg *provenance over* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$, $\mathsf{PartitionCoreIncidence}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteSimplicialChain}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, U, V, F, H_0, B, L, G, C, P, N$. It has no coordinate for hidden codon semantics, unrestricted minimal resolutions, ambient Betti-table equality, stochastic evolutionary interpretation, quotient support equality, or an external homology oracle.

Lean4 ✓ `BEDC.Derived.SupportNerveCensusLiftedBettiUp.SupportNerveCensusLiftedBettiTasteGate`
`single carrier alignment`

Theorem 4035.2 (SupportNerveCensusLiftedBetti NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *carrier $S = (I, U, V, F, H_0, B, L, G, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLifted}}$ *surface consists exactly of incidence enumeration, support-to-nerve realization, finite simplicial-complex admission, finite chain-boundary exposure, reduced homology readback in degrees $-1, 0, 1, 2$, lifted Betti ledger exposure, Hochster–nerve reduction provenance, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4035.1. The incidence, realization, complex, chain, homology, lifted Betti, reduction, transport, replay, provenance, and naming rows are exactly $I, U, V, F, H_0, B, L, G, C, P,$

The support-to-nerve route starts from I and U. It can expose a nerve only through the finite $\mathsf{SimplicialComplex}^{\uparrow}$ row V, so the census cannot be replaced by a host powerset calculation or a quotient support object.

The homology and Betti rows are downstream of the finite chain boundary F. The row H_0 restricts homology reads to the displayed reduced degrees, while B records the census and lift substitution only as finite ledger data.

Finally L, G, C, P, N preserve, replay, source, and name the same route. Since no remaining coordinate stores hidden codon semantics, unrestricted resolutions, host Betti-table equality, stochastic interpretation, quotient support equality, or external homology computation, the listed obligations exhaust the local NameCert surface. □

Theorem 4035.3 (SupportNerveCensusLiftedBetti realization)**.** *Every accepted* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{PartitionCoreIncidence}^{\uparrow} \longrightarrow \mathsf{SimplicialComplex}^{\uparrow} \longrightarrow \mathsf{FiniteSimplicialChain}^{\uparrow} \longrightarrow \mathsf{Homology}^{\uparrow} \longrightarrow \mathsf{HochsterNerveBettiRe}$$

The exported census is the displayed support-family count, reduced-homology signature table, and lifted Betti ledger, not a biological causal model, unrestricted coefficient-field theorem, ambient minimal resolution, or host-level homology calculation.

Proof. Start from the NameCert obligations in Definition 4035.2. They force support-family enumeration through I and support-to-nerve realization through U before any complex row is available.

Next the finite complex, chain, and homology rows V, F, H_0 expose the reduced homology signatures used by the census. These rows are displayed finite data, so the read cannot invoke an external simplicial-complex oracle or an unrestricted homology computation.

The lifted Betti ledger B and Hochster–nerve row L then package the codon lift as a finite strand read. Transport, replay, provenance, and local naming preserve that same route, which proves the stated realization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ read factors through displayed partition-core incidence, support-to-nerve realization, simplicial-complex, finite-chain, homology, lifted Betti, Hochster–nerve reduction, transport, replay, provenance, and local naming rows.

Independence witness: partition-core-incidence, support-to-nerve-realization, simplicialcomplex, finite-simplicial-chain, homology, hochster-nerve-betti-reduction: $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the support-family census, nerve realization, finite homology signature table, codon-lifted Betti ledger, and coefficient TasteGate boundary in one packet.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ SupportNerveCensusLiftedBetti}^{\uparrow}$ and $\mathsf{FieldFaithful\ SupportNer}$ TasteGate instances. Those instances must distinguish $I, U, V, F, H_0, B, L, G, C, P, N$ so the support census, nerve realization, and lifted Betti ledger cannot collapse into one sibling row.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}$.

Constructive story: A $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ packet is generated from partition-core incidence rows, support-to-nerve realization rows, finite simplicial-complex rows, finite chain rows, reduced homology rows, lifted Betti ledgers, Hochster–nerve reduction provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4035.1, Definition 4035.2, and Definition 4035.3 seed the finite support-nerve census and lifted Betti packet over incidence, realization, simplicial-complex, chain, homology, Betti-ledger, reduction, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, biological mechanism, unrestricted minimal resolutions, quotient codon semantics, coefficient-field completeness, or ambient homology computation.

Upgrade path: Obligation closure requires checked incidence enumeration, support-to-nerve realization, finite-complex admission, finite-chain exposure, homology readback, lifted Betti ledger stability, TasteGate nontriviality, field faithfulness, replay stability, and ledger nonescape for BEDC.Derived.SupportNerveCensusLiftedBettiUp.

Chapter 4036

A Concrete Naming Certificate for $\mathsf{EquicontinuousFamily}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Arzela–Ascoli finite-family reading, the $\mathsf{EquicontinuousFamily}^{\uparrow}$ packet records the shared-modulus interface by which a finite family of continuous maps over one compact metric source exposes a single displayed modulus row to downstream compact-uniform-continuity and Ascoli consumers. It sits after the compact source of chapter 129, the graph rows of chapter 130, the uniform-modulus consumer surface of chapter 369, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet does not introduce open-cover compactness, a host function-space topology, quotient equality of maps, or choice-selected pointwise moduli.

Definition 4036.1 (Equicontinuous family carrier)**.** *An* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, F, J, M, S, R, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the shared* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite metric reads and compact finite-net source used by the family.*
2. *F is the finite ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows whose source coordinate is X and whose values are read only through displayed graph windows.*
3. *J is the finite family-index row, naming exactly the family members admitted by F.*
4. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, recording one tolerance-to-radius ledger used uniformly for every index in J.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule through which the source and graph reads are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same tolerance and radius histories.*
7. *V is the* $\mathsf{Real}^{\uparrow}$ *value-seal row reached only after the graph, schedule, regular rational, and shared-modulus rows have been displayed.*
8. *H records componentwise* hsame *transport for the compact source, finite graph ledger, family index row, shared modulus row, observation schedule, regular rational readback, Real seal, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-source reads to graph reads, from tolerance requests to the shared modulus row, and from scheduled readback to the Real seal.*
10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousFamily}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, F, J, M, S, R, V, H, C, P, N$. It has no coordinate for an open cover, a selected global modulus outside M, a host function-space topology, quotient equality of functions, or a choice principle over the family.

Lean4 ✓ `BEDC.Derived.EquicontinuousFamilyUp.EquicontinuousFamilyTasteGate single carrier alignment`

Theorem 4036.2 (Equicontinuous family NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier $E = (X, F, J, M, S, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousFamily}^{\uparrow}}$ *surface consists of carrier admission, compact-source admission, finite continuous-map ledger admission, family-index totality, shared uniform-modulus exposure, StreamName and RegSeqRat readback, Real-seal handoff, componentwise transport, continuation replay, provenance, and ledger non-escape. These obligations are discharged entirely by the finite route*

$$X, F, J \longmapsto M \longmapsto S, R \longmapsto V$$

together with H, C, P, N.

Proof. Carrier admission is the projection of Definition 4036.1. The shared compact source row X, finite graph ledger F, family index row J, shared modulus row M, schedule row S, regular readback row R, and Real seal V are all visible coordinates of one finite packet.

The shared-modulus obligation is read before any Ascoli or compact-uniform-continuity consumer inspects the family. Each member named by J is admitted through F, and the same tolerance-to-radius ledger M is used for that member over the source row X.

The rows S, R, V provide the finite observation and value boundary. They keep the scheduled source and graph reads attached to the same regular rational readback and Real seal, rather than to a host function-space object.

Transport and replay are supplied by H and C, while P and N record provenance and local naming. Since the carrier has no row for open-cover compactness, quotient function equality, selected pointwise moduli, or ambient topology, the listed obligations exhaust the certificate surface. □

Theorem 4036.3 (Equicontinuous family uniform-modulus handoff)**.** *Any* $\mathsf{UniformModulus}^{\uparrow}$*, compact-uniform-continuity, or Arzela–Ascoli consumer of an accepted* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *carrier must receive the shared modulus row M together with the compact source row X, finite continuous-map ledger F, family index row J, observation schedule S, regular rational readback R, Real seal V, and structural rows H, C, P, N. The handoff exports only the displayed finite shared-modulus route and does not supply open-cover compactness, a function-space topology, quotient equality of maps, or choice-selected moduli.*

Proof. Begin with the carrier of Definition 4036.1. It displays the shared source, finite graph ledger, finite family index, shared modulus, schedule, readback, seal, transport, replay, provenance, and local naming rows as the whole packet.

Apply the NameCert obligations of Definition 4036.2. They force every accepted read through $X, F, J, M, S, R, V, H, C, P, N$, with M placed before the consumer can treat the family as equicontinuous.

The downstream consumer may add its own Ascoli, compact-uniform-continuity, or selection rows after this boundary. It cannot replace the displayed shared modulus by an open cover, host topology, quotient function equality, or a choice-selected family of pointwise moduli, because none of those objects is a carrier coordinate. □

Theorem 4036.4 (Equicontinuous family compact-uniform-modulus handoff)**.** *Any* CompactEquicontinuousFamily *consumer of an accepted* EquicontinuousFamily$^{\uparrow}$ *carrier*

$$E = (X, F, J, M, S, R, V, H, C, P, N)$$

must pass through $X, F, J, M, S, R, V, H, C, P, N$ *before the compact-family uniform budget is public. In particular, the compact-family read may consume the shared modulus only after the compact source, finite family ledger, family index, shared modulus row, StreamName schedule, RegSeqRat readback, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have all been displayed.*

Proof. Start with the uniform-modulus handoff of Definition 4036.3. It forces every shared-modulus consumer through the displayed rows $X, F, J, M, S, R, V, H, C, P, N$.

Now read the carrier route of CompactEquicontinuousFamilyModulus$^{\uparrow}$ from chapter 6234. Its compact-family budget consumes the EquicontinuousFamily$^{\uparrow}$ row as the family source before the compact-uniform output can be public.

Composing the two routes leaves no shortcut around the equicontinuous-family carrier. The compact-family consumer may add its own compact finite-net and output rows downstream, but the shared family budget remains the finite route stated above. □

Closure status for NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$**.**

Constructive story: An EquicontinuousFamily$^{\uparrow}$ packet is generated from a compact metric source row, a finite continuous-map family ledger, a finite family-index row, a shared uniform-modulus row, StreamName observations, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$, seed)

Scope closed: Definition 4036.1, Definition 4036.2, Definition 4036.3, and Definition 4036.4 seed the finite shared-modulus interface consumed by compact-uniform-continuity and Ascoli-style chapters.

Formal verification: FormalStatus(NameCert$_{\text{EquicontinuousFamily}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.EquicontinuousFamilyUp.EquicontinuousFamilyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, a host function-space topology, quotient function equality, or choice-selected pointwise moduli.

Upgrade path: Obligation closure requires checked carrier admission, family-index totality, shared modulus exposure, finite readback, Real-seal handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.EquicontinuousFamilyUp.

Chapter 4037

A Concrete Naming Certificate for $\mathsf{BishopRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ packet records the finite comparison modulus by which a Bishop real comparison request is reduced to inspected regular-Cauchy windows, positive apartness evidence, and dyadic tolerance rows. It extends the Bishop real source of chapter 343, the comparison surface of chapter 1381, the apartness witness of chapter 1881, and the L10 readback route of chapter 301 without importing trichotomy, quotient reals, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4037.1 (Bishop real comparison modulus carrier)**.** *A* $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, A, R, S, D, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BishopReal}^{\uparrow}$ *row exposing the regular-Cauchy real names and their finite comparison-facing source surface.*
2. *A is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *witness row supplying positive apartness data whenever the comparison request has a separated outcome.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used to inspect the rational values of the requested finite windows.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming exactly the finite windows read by the comparison request.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row storing the dyadic precision, slack, and positive-gap ledger used by the modulus.*
6. *M is the comparison modulus row; it maps each displayed comparison request to the finite window and dyadic tolerance budget consumed by R, S, D.*
7. *H records componentwise* hsame *transport for B, A, R, S, D, M and for the structural certificate rows.*
8. *C records displayed* Cont *replay from a comparison request to the window, readback, tolerance, apartness, and modulus rows.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopReal}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $B, A, R, S, D, M, H, C, P, N$. It has no coordinate for quotient real equality, total order trichotomy, a selected limit, countable choice, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopRealComparisonModulusUp.BishopRealComparisonModulusTasteGate single carrier alignment`

Theorem 4037.2 (Bishop real comparison modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ *carrier*

$$Q = (B, A, R, S, D, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Bishop-real source, apartness witness, regular readback, stream schedule, dyadic tolerance, comparison modulus, transport, replay, provenance, and local-name rows;*
2. *modulus readback, requiring each comparison request to expose through M the finite S windows, R rational observations, and D dyadic tolerance budget it consumes;*
3. *apartness handoff, requiring every positive separated outcome to pass through the displayed* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A before it is offered to a real-facing consumer;*
4. *regular-sequence compatibility, requiring the rational observations in R to be read from the same Bishop-real source surface named by B;*
5. *finite stream-window compatibility, requiring S to name only the windows requested by M and replayed by C;*
6. *dyadic tolerance exactness, requiring D to store precisely the comparison precision, slack, and positive-gap data used by M and A;*
7. *componentwise* hsame *transport for $B, A, R, S, D, M, C, P, N$ through H;*
8. *displayed* Cont *replay of each comparison request through the window, readback, tolerance, apartness, and modulus rows;*
9. Pkg *provenance for exactly the displayed packet;*
10. *ledger non-escape, requiring every consumer to remain inside $B, A, R, S, D, M, H, C, P, N$.*

Consequently the packet exports a finite comparison modulus interface for Bishop real names, not trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4037.1. The projection exposes exactly the Bishop source, apartness witness, regular readback, stream schedule, dyadic tolerance, comparison modulus, transport, replay, provenance, and local naming rows.

The modulus obligation is read from M together with S, R, D. A comparison request first names its finite stream windows, then reads the corresponding regular rational observations, and finally displays the dyadic precision and slack ledger used by the comparison.

The positive separated branch is forced through A. Since A is the only apartness witness row, the carrier cannot turn a modulus read into trichotomy, quotient equality, or a selected real representative.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. They do not add any coordinate outside B, A, R, S, D, M, so the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealComparisonModulus}^{\uparrow}$ packet is generated from a BishopReal row, a CauchyRealApartness witness, RegSeqRat readback, StreamName windows, a DyadicRatCore tolerance ledger, a comparison modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4037.1 and Definition 4037.2 seed the finite comparison-modulus interface over Bishop real source rows, apartness witnesses, regular readbacks, stream windows, and dyadic tolerance data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, modulus readback, apartness handoff, regular-sequence compatibility, finite window compatibility, dyadic tolerance exactness, transport, replay, provenance, and ledger non-escape.

Chapter 4038

A Concrete Naming Certificate for $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ packet records the Real-facing interface of an Archimedean Cauchy-complete ordered field as finite BEDC NameCert data. It reads the terminal Real seal of chapter 38, the regular rational sequence readback of chapter 301, the dyadic precision ledger of chapter 291, the finite stream-window surface of chapter 126, the algebraic order interface of chapter 766, the Cauchy-complete ordered-field route of chapter 3688, and the comparison capstone of chapter 2025. The chapter provides one L10 phase-exit surface for Real and Completion consumers without importing a host complete ordered field.

Definition 4038.1 (Archimedean Cauchy-complete field carrier)**.** *An* $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, Q, D, S, O, K, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar seal, read only as the terminal Real name row.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational names.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for displayed tolerance and radius requests.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which rational observations are requested.*
5. *O is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation-and-order row, naming addition, multiplication, zero, one, positivity, and order comparison only as finite ledger entries.*
6. *K is the* $\mathsf{CauchyCompleteOrderedField}^{\uparrow}$ *route row tying the Real seal, regular readback, dyadic ledger, stream window, and algebraic-order row to displayed Cauchy-complete ordered-field service.*
7. *F is the* $\mathsf{CompleteArchimedeanField}^{\uparrow}$ *comparison row, recording the Archimedean comparison surface reached from the same displayed Cauchy-complete ordered-field route.*
8. *H records componentwise* hsame *transport for the Real seal, regular readback, dyadic ledger, stream window, algebra-order row, Cauchy-complete ordered-field route, Archimedean comparison row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite precision request through stream observation, regular rational readback, Real sealing, order operation, Cauchy-complete route, Archimedean comparison, and local naming.*

10. *P is the* Pkg *provenance over* ArchimedeanCauchyCompleteField$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchimedeanField$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

11. *N is the local* NameCert$_{\text{ArchimedeanCauchyCompleteField}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, D, S, O, K, F, H, C, P, N$. It has no coordinate for quotient Real equality, trichotomy, selected limits, host supremum data, or ambient completeness.

Lean4 (definition) `BEDC.Derived.ArchimedeanCauchyCompleteFieldUp.ArchimedeanCauchyCompleteFieldUp`

Theorem 4038.2 (Archimedean Cauchy-complete field NameCert obligations)**.** *For every accepted* ArchimedeanCauchyCompleteField$^{\uparrow}$ *carrier*

$$A = (R, Q, D, S, O, K, F, H, C, P, N),$$

the local NameCert$_{\text{ArchimedeanCauchyCompleteField}^{\uparrow}}$ *surface consists exactly of carrier admission for the Real scalar seal R, regular rational readback Q, dyadic precision ledger D, finite stream window S, RealAlgOrder operation-and-order row O, CauchyCompleteOrderedField route row K, CompleteArchimedeanField comparison row F, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 4038.1. The accepted tuple exposes exactly the Real seal, regular readback, dyadic ledger, finite stream window, algebra-order row, Cauchy-complete ordered-field route, Archimedean comparison row, transport, replay, provenance, and local naming rows.

The scalar and completion-facing part is read through R, Q, D, S, K. The row K is allowed to cite the Cauchy-complete ordered-field surface only after the Real seal, regular readback, dyadic precision, and stream-window rows have been displayed.

The order and Archimedean comparison part is read through O and F. These rows record operation, order, and comparison service as BEDC packet rows rather than as a host complete ordered-field structure.

Finally H, C, P, N transport, replay, source, and name that finite surface. Since no other carrier coordinate remains, the listed rows exhaust the NameCert obligations. □

Theorem 4038.3 (Archimedean Cauchy-complete field L10 exit route)**.** *Every L10 Real or Completion consumer of an accepted* ArchimedeanCauchyCompleteField$^{\uparrow}$ *packet must factor its read through*

Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, CauchyCompleteOrderedField$^{\uparrow}$, CompleteArchin

as the displayed rows $R, Q, D, S, O, K, F, H, C, P, N$. Thus the packet supplies a single Archimedean Cauchy-complete ordered-field interface for Real-facing phase exit, but it exports no quotient Real equality, trichotomy, selected limit, host complete ordered field, or ambient completeness coordinate.

Proof. Start with an accepted consumer read. The Real scalar is exposed only by R, and the rational Cauchy evidence behind the read must pass through Q, D, S before any completion-facing row is visible.

The algebraic and order part is then confined to O. Addition, multiplication, zero, one, positivity, and comparison are read as displayed operation-and-order rows, not as host-field operations.

The completion route is K, and the Archimedean comparison capstone is F. These rows link the same Real seal, regular readback, dyadic ledger, stream window, and order row to the Cauchy-complete ordered-field and CompleteArchimedeanField packets.

The structural rows H, C, P, N preserve and name this route. Because the carrier lacks quotient Real equality, trichotomy, selected limit, host-field, and ambient-completeness coordinates, every L10 phase-exit read is confined to the finite surface stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}$ packet is generated from a Real scalar seal, RegSeqRat readback, DyadicRatCore precision ledger, StreamName finite window, RealAlgOrder operation and order row, CauchyCompleteOrderedField route row, CompleteArchimedeanField comparison row, componentwise transport, displayed replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4038.1, Definition 4038.2, and Definition 4038.3 seed the L10 Real and Completion phase-exit surface for Archimedean Cauchy-complete ordered-field consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanCauchyCompleteField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient Real equality, trichotomy, selected limits, host complete ordered fields, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rowwise operation and order service, Archimedean comparison service, transport, replay, provenance, and local naming for the same finite packet.

Chapter 4039

A Concrete Naming Certificate for $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ packet records a finite modulus for reading the difference of two regular Cauchy rational names. It sits over the regular source rows of chapter 301, the finite window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the displayed pointwise-difference route of chapter 429, and the terminal real-name seal of chapter 38. The packet supplies a Real and Completion consumer with a finite difference-modulus surface; it does not assert quotient stream equality, selected limits, countable choice, trichotomy, host real subtraction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4039.1 (Regular Cauchy difference-modulus carrier)**.** *A* $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, Y, W_X, W_Y, Q_X, Q_Y, \Delta, M, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose rational observations are compared only through finite requests.*
2. *W_X and W_Y are the aligned* $\mathsf{StreamName}^{\uparrow}$ *window rows exposing the finite prefixes of X and Y at a requested output precision.*
3. *Q_X and Q_Y are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows read from the two source windows.*
4. *Δ is the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *difference row obtained by subtracting the two endpoint rows named by Q_X and Q_Y.*
5. *M is the finite modulus row assigning each requested difference precision to the source-window and dyadic-tolerance requests carried by $W_X, W_Y, Q_X, Q_Y, \Delta$.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the difference readback after the modulus row has been consumed.*
7. *H records componentwise* hsame *transport for the two regular sources, both window rows, both dyadic tolerance rows, the difference row, the modulus row, the Real seal, provenance, and local naming.*
8. *C records displayed* Cont *replay for reading windows, dyadic tolerances, difference rows, modulus entries, Real sealing, and structural transport.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyDifference}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W_X, W_Y, Q_X, Q_Y, \Delta, M, E, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable-choice moduli, trichotomy, host subtraction, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDifferenceModulusUp.RegularCauchyDifferenceModulusTasteGate`
`single carrier alignment`

Theorem 4039.2 (Regular Cauchy difference-modulus NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ *packet $D = (X, Y, W_X, W_Y, Q_X, Q_Y, \Delta, M, E, H, C, P, N)$, the local NameCert surface consists exactly of carrier admission for the two regular rational sources, paired finite-window inspection, dyadic source tolerance reads, dyadic difference formation, modulus exactness, Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming. Every accepted difference-modulus read follows*

$$X, Y \longmapsto W_X, W_Y \longmapsto Q_X, Q_Y \longmapsto \Delta \longmapsto M \longmapsto E$$

together with H, C, P, N.

Proof. Project the carrier of Definition 4039.1. The source coordinates X, Y are the only regular-rational inputs, and their observations are exposed through the finite window rows W_X and W_Y.

The dyadic rows Q_X, Q_Y record the source tolerances read at those windows. The difference row Δ is downstream of these endpoint reads, so the modulus row M can assign a requested difference precision only to the displayed window and tolerance rows already present in the carrier.

The Real seal E is read after M, while H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient equality, selected representative, countable-choice, host-subtraction, or ambient-completeness coordinate, the listed obligations exhaust the local certificate. □

Theorem 4039.3 (Regular Cauchy difference-modulus Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-route consumer of an accepted* $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ *packet receives the difference readback only through the displayed route from X, Y to $W_X, W_Y, Q_X, Q_Y, \Delta, M, E$. The handoff preserves the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the paired* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and difference rows, the finite modulus row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming; it exports no quotient stream equality, selected limit, countable-choice schedule, trichotomy, host real subtraction, or ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4039.2. They place every accepted read on the finite route $X, Y, W_X, W_Y, Q_X, Q_Y, \Delta, M, E, H, C, P, N$.

A completion-route consumer can inspect only the two source rows through the paired windows and the dyadic source tolerances. It must therefore read the displayed difference row Δ and the modulus row M before any difference readback is available.

A $\mathsf{Real}^{\uparrow}$ consumer sees the terminal seal E only after the same modulus row has been consumed. The structural rows preserve this route, so no excluded quotient, selector, choice schedule, trichotomy, host-subtraction, or completeness object can be exported. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ read factors through two RegSeqRat source rows, paired StreamName window rows, paired DyadicRatCore tolerance rows, a DyadicRatCore difference row, a finite modulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regular-cauchy-difference, real: $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular sources, paired finite windows, dyadic source tolerances, a displayed dyadic difference, a finite modulus row, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName windows, paired DyadicRatCore tolerance rows, a DyadicRatCore difference row, a finite modulus row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4039.1, Definition 4039.2, and Definition 4039.3 seed the finite regular-Cauchy difference-modulus carrier, certificate surface, and Real-and-Completion handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, host real subtraction, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, paired regular-source exposure, window exactness, dyadic tolerance and difference budgeting, modulus exactness, Real sealing, transport, replay, provenance, and local NameCert non-escape for $\mathsf{RegularCauchyDifferenceModulus}^{\uparrow}$.

Chapter 4040

A Concrete Naming Certificate for $\mathsf{ProductUniformSpace}^{\uparrow}$

Following Bourbaki and Weil's product uniformity language, the $\mathsf{ProductUniformSpace}^{\uparrow}$ packet records the finite route by which two displayed uniform source packets are assembled into a product uniformity without importing a host product topology. It reads the $\mathsf{UniformSpace}^{\uparrow}$ entourage surface of chapter 339, the **??** projection route of chapter 34, the metric-compatible source of chapter 57, and the completion-facing boundary of chapter 95. The chapter names the product-entourage rows needed by completion consumers before any quotient completion, unrestricted entourage choice, or ambient topology is visible.

Definition 4040.1 (Product uniform space carrier)**.** *A* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, B, \Pi, E_A, E_B, W, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two uniform source packets. Each source exposes only its displayed carrier, entourage ledger, diagonal row, symmetry row, composition-refinement row, transport row, replay row, provenance row, and local* $\mathsf{NameCert}$ *row.*

2. *Π is the **??** projection row. It records the left and right projection reads and the paired point-name admission used by the product packet.*

3. *E_A and E_B are the paired entourage ledgers. A product entourage request must name a left source entourage from A and a right source entourage from B before it can be read as a product request.*

4. *W is the finite componentwise witness row. It records that a product pair is accepted exactly by its two projected source reads and their paired entourage ledgers.*

5. *H records componentwise* hsame *transport for the two source packets, the projection row, the paired entourage ledgers, the witness row, replay, provenance, and local naming rows.*

6. *C records displayed* Cont *replay from a product-entourage request through the projection row, the left and right source entourages, the componentwise witness, transport, provenance, and local naming rows.*

7. *P is the* Pkg *provenance over* $\mathsf{ProductUniformSpace}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* ***??****,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

8. N is the local $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $A, B, \Pi, E_A, E_B, W, H, C, P, N$. It has no coordinate for host product topology, unrestricted choice of entourages, quotient completion, selected limits, open-cover compactness, or bridge checking.

Lean4 ✓ `BEDC.Derived.ProductUniformSpaceUp.ProductUniformSpaceTasteGate single carrier alignment`

Theorem 4040.2 (Product uniform space NameCert obligations)**.** *For every accepted* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier $U = (A, B, \Pi, E_A, E_B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{UniformSpace}^{\uparrow}$ *source packets, the ?? projection row, the paired left and right entourage ledgers, the finite componentwise witness row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. No product-uniformity obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4040.1. The tuple displays exactly the two source packets, projection row, paired entourage ledgers, witness row, transport, replay, provenance, and local naming.

The source rows provide the only uniform data available to the packet. A product-entourage read must therefore pass through A and B and then through the named ledgers E_A and E_B.

The projection row Π and witness row W bind the product read to the two component reads. Transport by H is componentwise, and replay by C follows the same displayed route.

Since P and N only source and name this finite surface, and since no other carrier coordinate remains, the listed rows exhaust the local NameCert obligations. □

Theorem 4040.3 (Product uniform space projection-entourage route)**.** *Every completion-facing or metric-compatible read of an accepted* $\mathsf{ProductUniformSpace}^{\uparrow}$ *carrier factors through the finite route*

$$?? \longmapsto (\mathsf{UniformSpace}^{\uparrow}, \mathsf{UniformSpace}^{\uparrow}) \longmapsto \mathsf{Completion}^{\uparrow}.$$

More explicitly, a consumer first reads the projection row Π, then the left and right uniform source packets A, B, then the paired entourage ledgers E_A, E_B, and only afterward the witness, transport, replay, provenance, and local naming rows W, H, C, P, N. The route exports no host product topology, unrestricted entourage selector, quotient completion, selected limit, or ambient bridge proof.

Proof. Start with the accepted carrier and the obligations of Definition 4040.2. The product request is not available before Π displays the two projections and paired point-name admission.

Each projected read then enters one source uniform packet. The left and right entourage data are the displayed ledgers E_A and E_B, so the product entourage is a finite pair of component ledgers rather than a host-level neighbourhood of a product topology.

The completion-facing part reads only the same finite route. The witness row W records componentwise acceptance, while H, C, P, N transport, replay, source, and name that route for $\mathsf{Met}^{\uparrow}$ and $\mathsf{Completion}^{\uparrow}$ consumers.

Because the carrier has no coordinate for host product topology, unrestricted entourage choice, quotient completion, selected limits, or bridge checking, no consumer can project those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ProductUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{ProductUniformSpace}^{\uparrow}$ packet is generated from two UniformSpace source packets, a Product projection row, paired component entourage ledgers, a finite componentwise witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{ProductUniformSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4040.1, Definition 4040.2, and Definition 4040.3 seed the finite product-uniformity route used by metric-compatible completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProductUniformSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host product topology, unrestricted choice of entourages, quotient completion, selected limits, Lean verification, or bridge checking for ProductUniformSpace$^{\uparrow}$.

Upgrade path: Obligation closure requires checked source carrier admission, projection coverage, paired entourage exactness, componentwise transport, replay stability, provenance, local naming, and completion-route non-escape for BEDC.Derived.ProductUniformSpaceUp.

Chapter 4041

A Concrete Naming Certificate for $\mathsf{ConstructiveRealLine}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveRealLine}^{\uparrow}$ packet records the textbook-facing L10 real-line interface as finite BEDC NameCert data rather than as a primitive host continuum. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, the located real surface of **??**, and the apartness boundary of chapter 1380. The chapter names the interface on which later real-analysis packets can depend without claiming real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Definition 4041.1 (Constructive real line carrier)**.** *A* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, E, L, A, \Delta, I, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing requested dyadic precision, comparison slack, and density margins.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule through which every requested observation of a real name is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the inspected stream windows and dyadic tolerance rows into rational approximants.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row exported only after the D, S, R route has been displayed.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *boundary row recording finite interval-location evidence for the same readback surface.*
6. *A is the* $\mathsf{LocatedApartness}^{\uparrow}$ *boundary row recording positive separatedness evidence when a consumer asks for apartness.*
7. *Δ is the dyadic-density handoff row from rational or dyadic samples to the inspected real-name windows.*
8. *I is the interval readback row naming rational intervals whose endpoints are read through D, S, R and whose locatedness is checked through L and A.*
9. *H records componentwise* hsame *transport for $D, S, R, E, L, A, \Delta, I$ and for the structural certificate rows.*

10. *C records displayed* Cont *replay from a real-line request to the finite window, tolerance, readback, seal, locatedness, apartness, density, and interval rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveRealLine}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{LocatedApartness}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $D, S, R, E, L, A, \Delta, I, H, C, P, N$. It has no coordinate for real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Lean4 ✓ `BEDC.Derived.ConstructiveRealLineUp.ConstructiveRealLineTasteGate single carrier alignment`

Theorem 4041.2 (Constructive real line NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier*

$$Q = (D, S, R, E, L, A, \Delta, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance ledger, stream windows, regular rational readback, terminal real seal, locatedness boundary, apartness boundary, density handoff, interval readback, transport, replay, provenance, and local-name rows;*
2. *window-readback compatibility, requiring each S window consumed by a real-line request to be read by R under the tolerance ledger D;*
3. *seal discipline, requiring every* $\mathsf{Real}^{\uparrow}$ *export through E to pass first through the displayed D, S, R route;*
4. *locatedness discipline, requiring every located interval read to pass through L and I before a real-line consumer can inspect it;*
5. *apartness discipline, requiring every positive separatedness read to pass through A and the interval readback row I;*
6. *dyadic-density handoff, requiring each rational or dyadic sample exposed by Δ to be tied to the same D, S, R windows;*
7. *componentwise* hsame *transport for $D, S, R, E, L, A, \Delta, I, C, P, N$ through H;*
8. *displayed* Cont *replay of each real-line request through the finite route named by C;*
9. Pkg *provenance for exactly the displayed packet;*
10. *ledger non-escape, requiring every consumer to remain inside $D, S, R, E, L, A, \Delta, I, H, C, P, N$.*

Consequently the packet exports a textbook-facing constructive real-line interface, not real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Proof. Project the carrier of Definition 4041.1. The projection exposes exactly the dyadic tolerance, finite window, regular readback, real seal, locatedness, apartness, density, interval, transport, replay, provenance, and local naming rows.

The readback obligations are forced by the displayed route D, S, R. A consumer first names the finite stream windows, then reads the corresponding regular rational data, and only then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal E.

The located and apartness obligations are separate boundary reads. The interval row I supplies the rational interval readback, L supplies locatedness, and A supplies positive separatedness when the request needs it.

The density, transport, replay, provenance, and naming rows do not add a host continuum coordinate. Since every listed consumer remains inside $D, S, R, E, L, A, \Delta, I, H, C, P, N$, the packet cannot export real completeness, trichotomy, quotient streams, selected limits, or host real equality. □

Theorem 4041.3 (Constructive real line L10 seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{ConstructiveRealLine}^{\uparrow}$ *carrier factors through the L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with located and apartness observations read only through the sibling $\mathsf{LocatedReal}^{\uparrow}$ *and* $\mathsf{LocatedApartness}^{\uparrow}$ *boundary rows. No accepted real-line consumer can bypass this route through quotient streams, a selected limit, host real equality, trichotomy, or ambient real completeness.*

Proof. Begin with the carrier route in Definition 4041.1. Its only terminal real-name coordinate is E, and E is listed after the dyadic tolerance ledger D, the stream window schedule S, and the regular rational readback R.

The located and apartness reads remain boundary reads. A located request must use L and I, while a positive separatedness request must use A and the same interval readback row.

Finally replay the request through C and transport it through H. The provenance row P and local certificate row N name the same displayed packet, so the route has no hidden coordinate from which quotient stream equality, a selected host limit, trichotomy, host real equality, or real completeness could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveRealLine}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, a terminal RealUp seal, located-real and located-apartness boundaries, dyadic-density handoff, interval readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4041.1, Definition 4041.2, and Definition 4041.3 seed the textbook-facing constructive real-line interface over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealUp, LocatedApartnessUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRealLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient streams, selected limits, or host real equality.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, window-readback compatibility, seal discipline, locatedness and apartness boundaries, dyadic-density handoff, interval readback, transport, replay, provenance, and ledger non-escape.

$\mathsf{MontelSpace}^{\uparrow}$ reads the locally convex and Frechet-style bounded-set compactness route through chapter 3495, chapter 1735, and chapter 3693 before the child file expands the finite carrier.

Chapter 4042

A Concrete Naming Certificate for $\mathsf{MontelSpace}^{\uparrow}$

Following Bourbaki and Schaefer, the $\mathsf{MontelSpace}^{\uparrow}$ packet records the traditional locally convex and Frechet-style bounded-set compactness surface as finite BEDC NameCert data. It is the carrier companion to the normal-family packet of chapter 3693: the Montel theorem route may consume a displayed Montel-space source when local boundedness and compact-window extraction are read from a locally convex carrier rather than from a holomorphic family alone. The packet also sits over the complete separable metric boundary of chapter 1735, the complete-metric and Polish readback surfaces, and the finite bornological rows used by bounded-set consumers. It names a finite bornological compactness route, not host topological vector space equality, barrelledness, nuclearity, selected compact closures, quotient completions, or classical choice.

Definition 4042.1 (Montel space carrier)**.** *A* $\mathsf{MontelSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, F, B, Q, K, R, D, H, C, P, N)$$

with the following displayed rows:

1. *L is the locally convex source row, exposing the finite seminorm or entourage windows through which bounded sets are inspected.*
2. *F is the Frechet-style completeness and metrizability row, read through complete metric, complete separable metric, or Polish-space surfaces when those rows are displayed.*
3. *B is the bounded-set ledger, recording the finite family of bounded-set requests that the Montel-space consumer may inspect.*
4. *Q is the bornological compactness row: each displayed bounded request in B is routed to a compact, totally bounded, or sequentially compact readback already named by the surrounding metric packets.*
5. *K is the compact-window row consumed by normal-family, Arzela–Ascoli, and Montel theorem packets.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback for seminorm and metric tolerances.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by boundedness, compact-window, and completion reads.*

8. *H records componentwise* hsame *transport for the displayed locally convex, Frechet, bounded, compactness, tolerance, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from locally convex inspection through bounded-set admission, compact-window readback, and normal-family consumption.*
10. *P is the* Pkg *provenance over* $\mathsf{MontelSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{CompleteSeparableMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{MontelTheorem}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, F, B, Q, K, R, D, H, C, P, N$. It has no coordinate for a host locally convex equality, a quotient completion, a selected compact closure, a barrelled or nuclear structure, countable choice, or open-cover compactness.

Theorem 4042.2 (Montel space NameCert obligations)**.** *For every accepted* $\mathsf{MontelSpace}^{\uparrow}$ *carrier $S = (L, F, B, Q, K, R, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$ *surface consists exactly of locally convex source admission, Frechet or complete-separable readback when displayed, bounded-set ledger admission, bornological compactness readback, compact-window export, regular-rational and dyadic tolerance discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4042.1. The displayed tuple exposes the locally convex source row, Frechet or complete-separable readback row, bounded-set ledger, bornological compactness row, compact-window row, tolerance rows, and the structural rows.

The bounded-set read is finite. Each request in B is inspected only through the locally convex windows in L and the tolerance rows R, D, and the compactness answer is recorded in Q before any compact-window consumer sees K.

The replay row C then routes the same data toward normal-family and completion-facing consumers. Componentwise transport through H preserves the displayed rows, while P, N only source and name the packet.

Since the carrier has no coordinate for host locally convex equality, quotient completion, selected compact closures, countable choice, barrelledness, nuclearity, or open-cover compactness, the listed rows exhaust the certificate obligations. □

Theorem 4042.3 (Montel space bornological compactness route)**.** *Every bounded-set compactness consumer of an accepted* $\mathsf{MontelSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\textit{locally convex window} \longmapsto \textit{bounded-set ledger} \longmapsto \textit{bornological compactness readback} \longmapsto \textit{compact-window export.}$$

In particular, a $\mathsf{MontelTheorem}^{\uparrow}$ *or complete-separable metric consumer receives only the displayed rows $L, F, B, Q, K, R, D, H, C, P, N$ and receives no selected compact closure, quotient completion equality, host barrelledness theorem, nuclearity assertion, countable-choice schedule, or open-cover compactness proof.*

Proof. Start from the accepted carrier. The locally convex row L supplies the only inspection windows, and the Frechet or complete-separable row F is read only when the packet displays the corresponding metric boundary.

Next consume the bounded-set ledger B together with R, D. This gives the finite tolerance discipline for every bounded request and prevents a consumer from importing an ambient bornology or choice-selected family.

The compactness readback Q and compact-window row K are then replayed by C toward Montel theorem and completion-facing consumers. The route remains attached to the same rows under H, and P, N add only provenance and local naming.

Thus every bornological compactness handoff factors through the displayed packet and cannot export host locally convex equality, selected compact closures, quotient completion equality, barrelledness, nuclearity, countable choice, or open-cover compactness. □

Closure status for $\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4042.1, Definition 4042.2, and Definition 4042.3 seed the finite Montel-space carrier route through locally convex windows, Frechet or complete-separable readback, bounded-set ledgers, bornological compactness rows, compact-window export, tolerance ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MontelSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex equality, barrelledness, nuclearity, selected compact closures, quotient completions, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-set ledger exactness, bornological compactness readback, compact-window export, tolerance discipline, transport stability, replay, provenance, and local naming for BEDC.Derived.MontelSpaceUp.

Constructive story: A $\mathsf{MontelSpace}^{\uparrow}$ packet is generated from locally convex source rows, Frechet or complete-separable metric readback, bounded-set ledgers, bornological compactness rows, compact-window export, RegSeqRat and DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4043

A Concrete Naming Certificate for $\mathsf{BaireGenericResidual}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{BaireGenericResidual}^{\uparrow}$ packet records the constructive residual-point route through dense-open schedules rather than through a host countable intersection object. It sits over the Baire prefix of chapter 334, the dense-open certificate surface of chapter 1542, the complete-metric handoff of chapter 128, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter packages the residual thread consumed by later L10 complete-metric routes without adding countable choice, an external intersection point, quotient stream equality, or ambient real completeness.

Definition 4043.1 (Baire generic residual carrier)**.** *A* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, D, M, S, W, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row naming the inspected basic-open source and its finite prefix probes.*
2. *D is the dense-open schedule row inherited from the* $\mathsf{BaireCategory}^{\uparrow}$ *packet, including the displayed finite list of dense requests currently consumed by the residual route.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row through which the nested ball thread is interpreted.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule selecting the finite prefix, dense-open, and completion observations visible at this stage.*
5. *W is the stream-window ledger recording the inspected predecessor balls, successor balls, radius budgets, and finite membership checks.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius comparisons, and tolerance data used by the thread.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the B, D, M, S, W, R route has been displayed.*
8. *L is the residual ledger, listing the dense-open obligations already met by the constructed thread and the finite evidence that each listed obligation is met inside the chosen successor ball.*

9. *H records componentwise* hsame *transport for the Baire prefix, dense-open schedule, complete-metric handoff, stream windows, regular readback, real seal, residual ledger, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a residual request to the finite Baire prefix, dense-open schedule, completion handoff, stream windows, rational readback, real seal, and residual ledger rows.*
11. *P is the* Pkg *provenance over* $\mathsf{BaireGenericResidual}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the rows $B, D, M, S, W, R, E, L, H, C, P, N$. It has no coordinate for countable choice, a host intersection point, quotient equality of streams, or completeness of an ambient real line.

Lean4 ✓ `BEDC.Derived.BaireGenericResidualUp.BaireGenericResidualTasteGate single carrier alignment`

Theorem 4043.2 (Baire generic residual NameCert obligations)**.** *For every accepted* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier $G = (B, D, M, S, W, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix, dense-open schedule, complete-metric handoff, StreamName windows, window ledger, RegSeqRat readback, Real seal, residual ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted residual read must remain inside these rows and may not inspect countable-choice selectors, host intersection points, quotient stream equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 4043.1. The analytic rows are B, D, M, S, W, R, E, L, and the structural rows are H, C, P, N.

The dense-open work is confined to D, W, L. The schedule row names the finite dense requests, the window ledger records the predecessor and successor ball observations, and the residual ledger records that each displayed request has been met by the current thread.

The completion work is confined to M, S, R, E. The complete-metric row supplies the handoff for the nested thread, the stream schedule selects the inspected windows, the regular-rational row reads the centers and radii, and the real seal is downstream from that route.

Transport, replay, provenance, and local naming preserve the same displayed packet. Since no listed row stores a countable-choice selector, a host intersection point, quotient equality of streams, or ambient Real completeness, the displayed obligations exhaust the local NameCert surface. □

Theorem 4043.3 (Baire generic residual thread handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{BaireGenericResidual}^{\uparrow}$ *carrier factors through the constructive residual thread*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff records residual membership only through the finite residual ledger L and exports no host intersection point, countable-choice enumeration, quotient stream equality, or separate ambient completeness theorem.

Proof. Begin with the carrier route in Definition 4043.1. The terminal real row E is downstream from the Baire prefix B, the dense-open schedule D, the complete-metric handoff M, the inspected stream windows S, W, and the regular rational readback R.

The residual claim is ledgered, not selected. Each displayed dense-open obligation is recorded in L with the finite successor-ball evidence currently consumed by the route, so the proof never chooses a point outside the displayed thread.

Finally replay the route through C and transport it through H. The provenance row P and local naming row N name the same finite packet, so the exported $\mathsf{Real}^{\uparrow}$ seal has no hidden coordinate from which countable choice, a host intersection point, quotient stream equality, or ambient completeness can be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireGenericResidual}^{\uparrow}$ packet is generated from a BaireSpace prefix, a BaireCategory dense-open schedule, a CompleteMetric handoff, StreamName windows, RegSeqRat readback, a Real seal, a residual ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4043.1, Definition 4043.2, and Definition 4043.3 seed the constructive residual-point route over BaireSpaceUp, BaireCategoryUp, CompleteMetricUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireGenericResidual}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, host intersection points, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exactness, complete-metric handoff, StreamName window readback, RegSeqRat readback, Real seal discipline, residual-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.BaireGenericResidualUp.

Chapter 4044

A Concrete Naming Certificate for $\mathsf{CountableMetricCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CountableMetricCompletion}^{\uparrow}$ packet names the L10 completion route by which a countably displayed metric source exposes completion names through stream schedules, regular rational readbacks, dyadic tolerance ledgers, and terminal real seals. It refines the finite-window completion surfaces of chapter 4023 and chapter 1683, and it routes its observable data through chapter 126, chapter 301, chapter 291, chapter 38, and chapter 1612. The packet records only the displayed countable source, enumerated metric windows, regular sequence route, separated comparison, transport, replay, provenance, and local NameCert rows.

Definition 4044.1 (Countable metric completion carrier)**.** *A* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (M, A, S, Q, D, Z, R, H, K, P, N)$$

with the following displayed rows:

1. *M is the metric source row, naming the source point reads, distance comparison windows, and metric ledger offered to the completion request.*
2. *A is the countable admission row, exposing the finite portion of the source enumeration that the request actually inspects.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the selected finite prefixes of candidate completion names.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested radii, slack rows, and comparison budgets.*
6. *Z is the separated comparison row inherited from the separated-completion route after the same M, A, S, Q, D rows have been displayed.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for metric distances and completion-facing comparison readbacks.*
8. *H records componentwise* hsame *transport for the metric source, countable admission, stream schedule, regular readback, dyadic ledger, separated comparison, Real seal, replay, provenance, and local naming rows.*
9. *K records displayed* Cont *replay from a completion request through $M, A, S, Q, D, Z, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CountableMetricCompletion}^{\uparrow}$, $\mathsf{BishopMetricCompletion}^{\uparrow}$, $\mathsf{SequentialCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, S, Q, D, Z, R, H, K, P, N$. It has no coordinate for quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CountableMetricCompletionUp.CountableMetricCompletionTasteGate single carrier alignment`

Theorem 4044.2 (Countable metric completion NameCert obligations)**.** *For an accepted* CountableMetricCompleti *packet $C = (M, A, S, Q, D, Z, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$ *surface consists of carrier admission, countable source exposure, finite stream scheduling, regular rational readback, dyadic tolerance checking, separated comparison, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and ledger non-escape. Every obligation is discharged through the displayed route*

$$M \longmapsto A \longmapsto S \longmapsto Q \longmapsto D \longmapsto Z \longmapsto R$$

together with H, K, P, N.

Proof. Project the carrier of Definition 4044.1. The metric source row M and countable admission row A must be visible before any completion schedule is inspected.

The schedule row S exposes only finite stream prefixes. The regular readback row Q and dyadic ledger D then provide the rational observations and tolerance budgets used by the separated comparison row Z.

The terminal real seal R is reached only after that finite route. Componentwise transport through H and replay through K preserve the same rows, while P and N source and name the packet.

Since quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, and ambient completeness are absent from the carrier, the displayed rows exhaust the local NameCert obligations. □

Theorem 4044.3 (Countable metric completion Cauchy sequence route)**.** *Every Cauchy-sequence read of an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet factors through the displayed source metric row, countable admission row,* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, separated comparison row, transport, replay, provenance, and local NameCert rows. The route exports finite Cauchy evidence and separated comparison evidence; it does not export a quotient class, a selected representative, a countable-choice selector, or a host powerset of all sequences.*

Proof. Use Definition 4044.2. It places every accepted completion read on the route $M, A, S, Q, D, Z, R, H, K, P, N$.

A Cauchy-facing consumer first receives only the finite source enumeration visible in A. It then reads stream prefixes through S, rational regular-sequence observations through Q, and tolerance budgets through D.

The separated comparison row Z is downstream of those rows, so equality-facing or distance-facing reads cannot bypass the displayed Cauchy evidence. The structural rows H, K, P, N preserve and name that route.

No carrier row contains a quotient class, representative selector, choice function, or host powerset, so the Cauchy-sequence consumer boundary is exactly the finite displayed route. □

Theorem 4044.4 (Countable metric completion Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing handoff from an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet reaches the terminal real seal only after the metric source, countable admission, stream schedule, regular rational readback, dyadic tolerance ledger, and separated comparison rows have been displayed. Thus the handoff can reuse the Real row of chapter 38 for named distance readbacks without asserting ambient Real completeness, selected limits, quotient completion equality, trichotomy, or countable choice.*

Proof. Start from the carrier route in Definition 4044.1. The row R is not an independent source of completion data; it is downstream of M, A, S, Q, D, Z.

Replay through K therefore reaches the terminal $\mathsf{Real}^{\uparrow}$ seal only after the stream schedule, regular rational readback, dyadic tolerance ledger, and separated comparison rows have been inspected. Componentwise hsame transport through H keeps the same dependence under name change.

The Real-facing consumer receives a seal for the displayed readback, not a completeness principle. Since no excluded object is a coordinate of C, the handoff cannot introduce selected limits, quotient equality, trichotomy, countable choice, or ambient Real completeness. □

Theorem 4044.5 (Countable metric completion ledger non-escape)**.** *The provenance and local naming rows of an accepted* $\mathsf{CountableMetricCompletion}^{\uparrow}$ *packet prevent downstream completion consumers from reading outside the displayed L10 ledger. Any consumer of the packet receives only* $M, A, S, Q, D, Z, R, H, K, P, N$ *and the cited Bishop metric completion, sequential completion, StreamName, RegSeqRat, DyadicRatCore, separated-completion, and Real rows; no host completion, quotient universe, selected limit operation, countable-choice principle, or ambient Real-completeness theorem is exported.*

Proof. Apply Definition 4044.2 to expose the local NameCert surface. The provenance row P cites the completion and numerical chapters only through their displayed packet rows, and the local name N records the same finite carrier.

Replay through K cannot add a new coordinate. It can only revisit the source, countable admission, stream schedule, regular readback, dyadic ledger, separated comparison, Real seal, and transport rows already present in C.

Consequently every downstream consumer is confined to the L10 ledger named by the packet. Host completions, quotient universes, selected limits, countable choice, and ambient Real completeness are not exported because no row of the carrier stores them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CountableMetricCompletion}^{\uparrow}$ packet is generated from a metric source row, countable admission row, finite $\mathsf{StreamName}^{\uparrow}$ schedule, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, separated-completion comparison row, terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4044.1, Definition 4044.2, Definition 4044.3, Definition 4044.4, and Definition 4044.5 seed the countable metric-completion route through displayed StreamName, RegSeqRat, DyadicRatCore, separated-completion, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountableMetricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, countable choice, host powersets, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked carrier admission, countable source exposure, regular readback, separated comparison, Real sealing, replay, provenance, and local NameCert non-escape for BEDC.Derived.CountableMetricCompletionUp.

Chapter 4045

Cauchy Sequential Continuity NameCert Construction

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchySequentialContinuity}^{\uparrow}$ packet records the finite certificate that a displayed continuous map carries a regular Cauchy name to a regular Cauchy image name. It sits between the certified sequence-consumer surface of chapter 1466, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, and the graph-and-modulus rows of chapter 130. The packet gives the L10 Real and Completion route a finite preservation certificate before a completion consumer treats the image stream as a real name.

Definition 4045.1 (Cauchy sequential continuity carrier)**.** *A* $\mathsf{CauchySequentialContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E_Y, B, H, C, P, N)$$

with the following displayed rows:

1. M_X *and* M_Y *are the source and target metric rows used by the displayed map.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-modulus row for the source-to-target map.*
3. S *is the source Cauchy stream surface, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
4. W_X *and* W_Y *are the inspected source windows and image windows.*
5. R_X *and* R_Y *are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the source and image windows.*
6. E_X *and* E_Y *are the* $\mathsf{Real}^{\uparrow}$ *seal rows available after the corresponding regular readbacks.*
7. B *is the Cauchy-preservation ledger, listing the finite pairs of source thresholds, target tolerances, and modulus rows needed to turn the* R_X *regularity witness into the* R_Y *regularity witness.*
8. H *records componentwise* hsame *transport for all displayed rows and for continuation, provenance, and local naming.*
9. C *records displayed* Cont *replay from the source windows through the continuous-map modulus row to the image windows, then through regular readback and seal handoff.*
10. P *is the* Pkg *provenance over* $\mathsf{CauchySequentialContinuity}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host function, quotient stream equality, selected limit, countable choice, external completeness theorem, or ambient real-continuity principle.

Lean4 ✓ `BEDC.Derived.CauchySequentialContinuityUp.CauchySequentialContinuityTasteGate single carrier alignment`

Theorem 4045.2 (Cauchy sequential continuity Cauchy preservation)**.** *Every accepted* CauchySequentialContinuity *carrier carries its displayed source regular Cauchy name to a displayed target regular Cauchy name. More precisely, any consumer of the target* $\mathsf{Real}^{\uparrow}$ *seal must factor through*

$$S \longmapsto W_X \longmapsto R_X \longmapsto F \longmapsto W_Y \longmapsto B \longmapsto R_Y \longmapsto E_Y$$

together with the metric rows, transport rows, continuation replay, provenance, and local naming.

Proof. Project the carrier of Definition 4045.1. The source stream S is visible only through the inspected windows W_X, and the source Cauchy evidence is visible only through the regular rational readback R_X.

Apply the displayed continuous-map row F to the same finite source windows. The continuation row C records that every image window W_Y is produced from W_X through the graph-and-modulus data of F, not by a host function equality or an unrecorded semantic map.

The preservation ledger B then binds the source thresholds from R_X to the target tolerances requested for R_Y. Componentwise hsame transport in H keeps the metric rows, window rows, regular readbacks, seal rows, continuation, provenance, and local naming on the same displayed packet.

Finally the target seal E_Y is downstream from R_Y. Thus a target-real consumer that bypasses the source readback, the continuous-map modulus row, the preservation ledger, or the target readback leaves the accepted carrier. □

Theorem 4045.3 (Cauchy sequential continuity NameCert obligations)**.** *For every accepted* CauchySequentialConti *carrier* $K = (M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E_Y, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission, source and target metric exposure, continuous-map modulus exposure, source stream-window exposure, source and target regular-rational readback, source and target real-seal handoff, Cauchy-preservation ledger readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Read the displayed tuple in Definition 4045.1. The analytic rows are $M_X, M_Y, F, S, W_X, W_Y, R_X, R_Y, E_X, E$ the structural rows are H, C, P, N.

The source side is exposed by S, W_X, R_X, E_X. These rows say only that the inspected stream windows have regular rational readback and a downstream real seal.

The map side is exposed by M_X, M_Y, F, W_Y, B. The continuous-map row supplies the finite modulus data used to form image windows, and the preservation ledger records the threshold comparisons that certify those image windows as regular Cauchy.

The remaining rows transport and replay the same finite route. Since no excluded host principle is a coordinate of the carrier, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequentialContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchySequentialContinuity}^{\uparrow}$ packet is generated from source and target metric rows, a ContinuousMap graph-and-modulus row, finite source and image StreamName windows, source and target RegSeqRat

readbacks, RealUp seal rows, a Cauchy-preservation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CauchySequentialContinuity}\uparrow}$, seed)

Scope closed: Definition 4045.1, Definition 4045.2, and Definition 4045.3 seed the finite certificate that displayed continuous maps preserve regular Cauchy names.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchySequentialContinuity}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sequential continuity, quotient stream equality, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target metric exposure, continuous-map modulus exposure, stream-window exactness, RegSeqRat readback, RealUp seal handoff, Cauchy-preservation ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequentialContinuityUp.

Chapter 4046

A Concrete Naming Certificate for $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ packet records the completion route for bounded continuous real-valued functions under the displayed sup-norm surface. It reads the finite sup-norm carrier of chapter 3807, the uniform-limit metric handoff of chapter 1764, the real-valued uniform Cauchy route of chapter 1830, the compact metric source of chapter 129, the continuous-map row of chapter 130, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the completion-facing boundary as finite BEDC data: Cauchy rows, uniform windows, compact-source continuity, Real-sealed sup-norm estimates, and local NameCert replay are visible, while host function equality, quotient function spaces, selected limits, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the carrier.

Definition 4046.1 (Continuous sup-norm completion carrier)**.** *A* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, U, A, K, F, R, W, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *row, exposing the compact-source sup-norm ledger through the sibling carrier.*
2. *U is the* $\mathsf{UniformLimitMetric}^{\uparrow}$ *row that reads sup-norm Cauchy control as a uniform metric-limit request.*
3. *A is the* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *row recording the displayed uniform Cauchy schedule for the function rows under inspection.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row shared by the sup-norm and Cauchy schedules.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the inspected real-valued function data over K.*
6. *R is the pointwise* $\mathsf{Real}^{\uparrow}$ *difference-and-bound seal reached after the compact source, continuous-map, and sup-norm rows have been displayed.*
7. *W is the finite uniform-window ledger, recording the tolerance request, source windows, and paired function-row comparisons used by the Cauchy route.*
8. *L is the completion-facing limit service row. It stores only the displayed eligibility of the Cauchy route for a uniform-limit consumer, not a selected host limit.*

9. *H records componentwise* hsame *transport for $S, U, A, K, F, R, W, L, C, P, N$.*
10. *C records the displayed* Cont *replay route from a completion request through the rows $S, U, A, K, F, R, W, L, H, P,$*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{UniformLimitMetric}^{\uparrow}$, $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed sup-norm source, uniform-limit metric route, uniform Cauchy schedule, compact source, continuous-map row, Real seal, finite window ledger, limit-service eligibility row, structural transport, replay, provenance, and local naming. It has no coordinate for host function equality, quotient function-space equality, a choice-selected limiting function, open-cover compactness, or a theorem asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ContinuousSupNormCompletionUp.ContinuousSupNormCompletionTasteGate single carrier alignment`

Theorem 4046.2 (Continuous sup-norm completion NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier $Q = (S, U, A, K, F, R, W, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ContinuousSupNorm}}$ *surface consists exactly of carrier admission,* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *source exposure through S,* $\mathsf{UniformLimitMetric}^{\uparrow}$ *handoff through U,* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$ *schedule exposure through A, compact metric source exposure through K, continuous-map row exposure through F, pointwise* $\mathsf{Real}^{\uparrow}$ *difference-and-bound sealing through R, finite uniform-window exposure through W, completion-facing limit-service eligibility through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4046.1. The tuple displays the sup-norm source, uniform-limit metric row, real-function Cauchy schedule, compact metric source, continuous-map row, Real seal, finite window ledger, limit-service row, transport, replay, provenance, and local naming.

The source and Cauchy rows determine the only function rows that may be inspected. The compact metric and continuous-map rows determine the common domain of those inspections, while the Real seal and finite window ledger supply the displayed sup-norm estimates.

The limit-service row records that this displayed route may be consumed by a uniform-limit metric surface. Since H, C, P, N only transport, replay, source, and name the same rows, the listed coordinates exhaust the local NameCert obligations. □

Theorem 4046.3 (Continuous sup-norm completion Cauchy route)**.** *Every accepted* ContinuousSupNormCompletio *carrier $Q = (S, U, A, K, F, R, W, L, H, C, P, N)$ routes a bounded continuous-function Cauchy request through the finite chain*

$$\mathsf{ContinuousSupNorm}^{\uparrow} \longmapsto \mathsf{RealFunctionUniformCauchy}^{\uparrow} \longmapsto \mathsf{UniformLimitMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, every tolerance-controlled comparison first reads the compact source K and continuous-map row F, then the sup-norm source S, the uniform Cauchy schedule A, the finite window ledger W, the Real difference-and-bound seal R, and finally the uniform-limit metric row U and limit-service row L.

Proof. Start with the accepted carrier and the obligations of Definition 4046.2. The compact source K and continuous-map row F fix the inspected domain and function rows before any sup-norm comparison can be read.

The source row S supplies the finite sup-norm comparison route from chapter 3807. The uniform Cauchy schedule A and window ledger W then place the same comparisons under a displayed tolerance request.

The Real seal R is reached only after those rows have been displayed. The uniform-limit metric row U and the limit-service row L therefore consume the same finite route rather than a host completion object, selected limiting function, quotient function space, or ambient completeness theorem. □

Theorem 4046.4 (Continuous sup-norm completion uniform-limit seal)**.** *Every uniform-limit read exported by an accepted* $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ *carrier factors through the displayed rows* $S, U, A, K, F, R, W, L, H, C, P, N$*. Consequently a downstream compact function-space or Banach-space route may consume the carrier only as a finite sup-norm Cauchy boundary over* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{RealFunctionUniformCauchy}^{\uparrow}$*,* $\mathsf{UniformLimitMetric}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *data; it receives no host function equality, quotient function space, selected limit, open-cover compactness proof, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 4046.3. It places every completion-facing request behind the compact source, continuous-map row, sup-norm comparison row, uniform Cauchy schedule, finite window ledger, and Real seal.

The rows U and L do not create another completion coordinate. They only expose the eligibility of the displayed Cauchy route for the uniform-limit metric consumer.

Finally, componentwise transport, replay, provenance, and local naming preserve the same row list. Since the carrier has no field for host function equality, quotient function-space equality, selected limits, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no downstream route can extract those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$ packet is generated from a ContinuousSupNorm source row, a UniformLimitMetric handoff row, a RealFunctionUniformCauchy schedule, a CompactMetric source row, a ContinuousMap row, a RealUp difference-and-bound seal, a finite uniform-window ledger, a completion-facing limit-service row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4046.1, Definition 4046.2, Definition 4046.3, and Definition 4046.4 seed the completion-facing sup-norm Cauchy boundary for bounded continuous real-valued functions over a displayed compact metric source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousSupNormCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient function spaces, selected limits, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{ContinuousSupNormCompletion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, sup-norm source exposure, uniform Cauchy schedule exactness, compact-source and continuous-map stability, RealUp seal non-escape, finite window coverage, uniform-limit handoff, replay stability, and ledger non-escape for BEDC.Derived.ContinuousSupNormCompletionUp.

Chapter 4047

A Concrete Naming Certificate for $\mathsf{CofinalSequenceSelector}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and Kelley on directed tails, the $\mathsf{CofinalSequenceSelector}^{\uparrow}$ packet records the finite monotone index-window selector used before a completion-facing consumer reads a cofinal subsequence, a Cauchy-filter tail, or a terminal real seal. It sits after the cofinal sequence surface of chapter 1594, the cofinal tail map of chapter 2141, the Cauchy-filter packet of chapter 333, the regular Cauchy subsequence surface of chapter 381, and the terminal real-name seal of chapter 38. The packet names a finite selector schedule; it does not choose an infinite subsequence by countable choice, quotient streams, ambient directed-set cofinality, host real equality, or an external completeness principle.

Definition 4047.1 (Cofinal sequence selector carrier)**.** *A* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, W, M, Q, F, U, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Nat}^{\uparrow}$ *index row, displaying the finite source interval and the terminal lower bound requested by the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing every inspected source index and selector output.*
3. *M is the monotonicity ledger, recording that the displayed selector outputs are nondecreasing on the finite window.*
4. *Q is the cofinal-window ledger, recording that every displayed lower-bound request is met inside the same finite selector window.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-filter row reached from I, W, M, Q.*
6. *U is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row, recording the selected regular-Cauchy subsequence readback.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal row available only after the window, filter, and subsequence handoffs have been displayed.*
8. *H records componentwise* hsame *transport for I, W, M, Q, F, U, R and for the continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a finite lower-bound request through the selector window, monotonicity ledger, cofinal ledger, Cauchy-filter tail, regular-Cauchy subsequence handoff, and terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CofinalSequenceSelector}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, W, M, Q, F, U, R, H, C, P, N$. It has no coordinate for a choice-selected infinite subsequence, quotient stream equality, host order on all natural-number functions, ambient filter convergence, global completeness, Lean verification, or bridge checking.

Lean4 (definition) `BEDC.Derived.CofinalSequenceSelectorUp.CofinalSequenceSelectorUp`
Lean4 ✓ `BEDC.Derived.CofinalSequenceSelectorUp.CofinalSequenceSelectorTasteGate single carrier alignment`

Theorem 4047.2 (Cofinal sequence selector NameCert obligations)**.** *For every accepted* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier $S = (I, W, M, Q, F, U, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$ *surface consists exactly of finite* $\mathsf{Nat}^{\uparrow}$ *index admission,* $\mathsf{StreamName}^{\uparrow}$ *window exposure, finite monotonicity exposure, finite cofinal-window exposure,* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-filter handoff,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *selected-subsequence handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4047.1. The rows I, W, M, Q expose the finite selector schedule: the inspected index window, the output window, the monotonicity ledger, and the cofinal lower-bound ledger.

The analytic handoff is read through F, U, R. The filter row reads the displayed tail request, the subsequence row reads the same selected window as regular-Cauchy data, and the terminal row seals the resulting regular rational and real-facing output.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no coordinate for countable choice, quotient stream equality, ambient filter convergence, host real equality, or global completeness, the listed rows exhaust the local certificate obligations. □

Theorem 4047.3 (Cofinal sequence selector tail-filter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CofinalSequenceSelector}^{\uparrow}$ *carrier reaches the terminal read through the finite route*

$$\mathsf{Nat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{RegularCauchySubsequence}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route first displays the finite monotone cofinal index window, then reads the Cauchy-filter tail, then hands the selected regular-Cauchy subsequence to the regular rational and real-name seals. It exports no selected infinite subsequence, quotient stream equality, countable-choice schedule, host filter limit, or ambient completeness theorem.

Proof. Begin with the finite selector rows I, W, M, Q. They provide only a displayed natural-number window, the selected outputs in that window, and the finite proof rows that the displayed outputs are monotone and cofinal for the displayed lower-bound requests.

Next read F and U. The Cauchy-filter row consumes the same tail window, while the regular-Cauchy subsequence row consumes the selected outputs as the finite subsequence surface already used by chapter 381.

Finally pass through R, H, C, P, N. The regular rational and real seal rows are downstream of the filter and subsequence handoffs, and the transport, replay, provenance, and naming rows keep the route inside the accepted packet. Thus a completion-facing consumer receives the finite selector handoff and no additional choice, quotient, host-limit, or ambient-completeness coordinate. □

Falsifiable prediction: Every accepted $\mathsf{CofinalSequenceSelector}^{\uparrow}$ consumer must expose finite $\mathsf{Nat}^{\uparrow}$ index rows, $\mathsf{StreamName}^{\uparrow}$ selector windows, monotonicity and cofinal-window ledgers, a $\mathsf{CauchyFilter}^{\uparrow}$ tail row, a $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff row, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before the selected tail is available.

Independence witness: Nearest siblings: cofinal sequence, cofinal tail map, CauchyFilter, RegularCauchySubsequence, RegSeqRat, StreamName, Nat, and Real. The packet is not identical to any sibling because it jointly binds finite monotone selector windows, finite cofinal lower-bound ledgers, tail-filter handoff, regular-Cauchy subsequence handoff, and the terminal regular-rational and real seals.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalSequenceSelector}^{\uparrow}$ packet is generated from finite NatUp index rows, StreamNameUp selector windows, monotonicity ledgers, cofinal-window ledgers, CauchyFilterUp tail handoffs, RegularCauchySubsequenceUp handoffs, RegSeqRatUp and RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4047.1, Definition 4047.2, and Definition 4047.3 seed the finite monotone cofinal selector surface for completion-facing tail consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalSequenceSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, ambient filter convergence, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite index admission, selector-window exposure, monotonicity and cofinal-window ledgers, CauchyFilterUp tail handoff, RegularCauchySubsequenceUp handoff, RegSeqRatUp and RealUp seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CofinalSequenceSelectorUp.

Chapter 4048

A Concrete Naming Certificate for $\mathsf{SupportNerveCensus}^{\uparrow}$

Following the finite theory of abstract simplicial complexes, nerves, and Hochster-style Betti bookkeeping, the $\mathsf{SupportNerveCensus}^{\uparrow}$ packet records the support-family census of Definition 9462.1655 as finite NameCert data. It sits beside the simplicial-complex carrier of chapter 238, the finite simplicial-chain carrier of chapter 3414, the Hochster nerve–Betti reduction of chapter 3744, and the Hilbert–Alexander blocker packet of chapter 3932. The packet names the finite table of support nerves, reduced Betti signatures, top-support criterion, and lifted-Betti handoff; it does not import an unrestricted resolution, a host homology computation, coefficient-field choice, or ambient equality of Betti tables.

Definition 4048.1 (Support nerve census carrier)**.** *A* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, A, N, B, T, L, H, R, P, M)$$

with the following displayed rows:

1. *S is the support-type row, displaying the fifteen nonempty support symbols and the four singleton support entries used by the top-support criterion.*
2. *A is the finite support-family row, recording the inspected family $A \subseteq \mathcal{S}$ and its cardinality bucket.*
3. *N is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve row, listing the faces of the support nerve N_A on four vertices.*
4. *B is the reduced Betti-signature row, recording the displayed tuple $(\widetilde{\beta}_{-1}, \widetilde{\beta}_0, \widetilde{\beta}_1, \widetilde{\beta}_2)$.*
5. *T is the top-support row, recording whether the four singleton supports are present and therefore whether the top homology case is exposed.*
6. *L is the lifted-Betti row, reading the finite support-level table through the codon lift substitution of Definition 9462.1658.*
7. *H records componentwise* hsame *transport for S, A, N, B, T, L and for the replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from support-family inspection through nerve formation, finite homology readback, Betti-signature table lookup, top-support checking, and lifted-Betti handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{SupportNerveCensus}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{FiniteSimplicialChain}^{\uparrow}$, $\mathsf{HochsterNerveBettiReduction}^{\uparrow}$, $\mathsf{HilbertAlexanderBlocker}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. M is the local $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, A, N, B, T, L, H, R, P, M$. It has no coordinate for an unrestricted minimal resolution, host chain-complex equality, coefficient-field selection, unbounded simplicial homology, ambient Betti-table equality, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.SupportNerveCensusUp.SupportNerveCensusTasteGate single carrier alignment`

Theorem 4048.2 (Support nerve census NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier $C = (S, A, N, B, T, L, H, R, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}^{\uparrow}}$ *surface consists exactly of support-type exposure, finite support-family exposure,* $\mathsf{SimplicialComplex}^{\uparrow}$ *nerve exposure, finite reduced-Betti signature readback, top-support criterion exposure, lifted-Betti handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.SupportNerveCensusUp.SupportNerveCensusNameCertObligations`

Proof. Project the carrier of Definition 4048.1. The rows S, A, N display the finite support universe, the inspected support family, and the resulting four-vertex support nerve.

The census data are exactly B, T, L. The Betti row records the reduced signature for the displayed nerve, the top-support row records the singleton-support criterion of Definition 9462.1657, and the lifted row records the finite handoff used by Definition 9462.1658.

The structural rows H, R, P, M transport, replay, source, and name the same finite table. Since the packet contains no coordinate for unrestricted resolutions, host chain-complex equality, coefficient-field selection, unbounded homology, or ambient Betti-table equality, the listed rows exhaust the local certificate obligations. □

Theorem 4048.3 (Support nerve census lifted Betti handoff)**.** *Every lifted-Betti consumer of an accepted* $\mathsf{SupportNerveCensus}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{SimplicialComplex}^{\uparrow} \longmapsto \mathsf{FiniteSimplicialChain}^{\uparrow} \longmapsto \mathsf{HochsterNerveBettiReduction}^{\uparrow} \longmapsto \mathsf{SupportNerveCensus}^{\uparrow}.$$

The route exposes the support nerve, finite chain readback, reduced Betti-signature table, top-support criterion, and codon-lift handoff before any lifted Betti entry is exported. It exports no unrestricted minimal resolution, host homology computation, coefficient-field choice, or ambient equality of Betti tables.

Proof. Start with the support-family and nerve rows S, A, N. These rows construct a finite four-vertex simplicial complex and therefore keep the computation inside the visible support nerve.

Next read B and T. The reduced Betti-signature row is a finite table lookup through the same finite chain surface used by the Hochster nerve–Betti reduction, and the top-support row records the singleton-support criterion rather than an ambient homology theorem.

Finally apply the lifted row L and the structural rows H, R, P, M. The continuation row fixes the order from support-family inspection to lifted-Betti export, and the provenance and name rows keep the result attached to the displayed packet. Thus the lifted table is exported only as finite NameCert data. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveCensus}^{\uparrow}$ consumer must expose support-type rows, finite support-family rows, a $\mathsf{SimplicialComplex}^{\uparrow}$ nerve row, a reduced Betti-signature row, a top-support criterion row, a lifted-Betti handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before a lifted Betti entry is available.

Independence witness: Nearest siblings: SimplicialComplex, FiniteSimplicialChain, HochsterNerveBettiReduction, HilbertAlexanderBlocker, and the support-nerve census theorem in the vision section. The packet is not any one sibling because it binds the full finite support-family census, signature table, top-support criterion, and lifted-Betti handoff into one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}$.

Constructive story: A $\mathsf{SupportNerveCensus}^{\uparrow}$ packet is generated from support-type rows, finite support-family rows, SimplicialComplexUp nerve rows, finite reduced-Betti signature rows, top-support criterion rows, lifted-Betti handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4048.1, Definition 4048.2, and Definition 4048.3 seed the finite support-nerve census packet and its lifted-Betti handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveCensus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted minimal resolutions, host homology computations, coefficient-field choice, unbounded simplicial homology, ambient Betti-table equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support-family exposure, nerve exposure, finite chain readback, reduced Betti-signature table lookup, top-support criterion, lifted-Betti handoff, transport, replay, provenance, and local naming for BEDC.Derived.SupportNerveCensusUp.

Chapter 4049

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ packet records the located-limit surface of a regular Cauchy name as finite NameCert data. It sits after the dyadic tolerance carrier of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the endpoint-cell surface of chapter 394, the $\mathsf{CauchyRealLocator}^{\uparrow}$ locator row, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a displayed located limit for a regular Cauchy name without quotienting streams, selecting a host limit, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4049.1 (Regular Cauchy located-limit carrier)**.** *A* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, D, Q, I, A, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected regular Cauchy segment and the requested precision.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row obtained from W and D.*
4. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the displayed rational interval that traps the requested tail window.*
5. *A is the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row that relates the regular Cauchy tail to the located interval at the same dyadic tolerance.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after W, D, Q, I, A have been displayed.*
7. *H records componentwise* hsame *transport for the window, dyadic ledger, rational readback, located interval, locator, and Real-seal rows.*
8. *C records displayed* Cont *replay from a precision request through the regular Cauchy window, located interval, and locator rows to the Real-facing seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, D, Q, I, A, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected host limit, quotient real equality, countable choice, trichotomy, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitTasteGate single carrier alignment`

Theorem 4049.2 (Regular Cauchy located-limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocated}$ *carrier $L = (W, D, Q, I, A, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$ *surface consists exactly of finite window exposure, dyadic tolerance admission, regular rational readback, located-interval trapping, Cauchy-real locator admission, Real-seal ordering, componentwise transport, continuation replay, provenance, and local naming. Every consumer must read the finite route*

$$W, D \longmapsto Q \longmapsto I, A \longmapsto R$$

together with H, C, P, N before treating the packet as a located limit of a regular Cauchy name.

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitNameCertObligations`

Proof. Project the carrier of Definition 4049.1. The rows $W, D, Q, I, A, R, H, C, P, N$ are the only coordinates visible to the local classifier.

The regular Cauchy side is read first. The window row W supplies a finite observation, the dyadic row D supplies the requested tolerance, and the readback row Q is downstream of those two rows.

The located-limit side is then forced through I and A. The interval row I names the displayed trap for the requested tail window, while the locator row A relates that trap to the same regular Cauchy data already used by Q.

Finally R is reached through the displayed route, and H, C, P, N preserve the route by transport, replay, provenance, and local naming. Since the excluded quotient, choice, host-limit, and completeness objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 4049.3 (Regular Cauchy located-limit real handoff)**.** *An accepted* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *carrier may expose its* $\mathsf{Real}^{\uparrow}$ *seal row R only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{LocatedInterval}^{\uparrow}$ *trapping row I, and the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *row A have been displayed through H and C. Thus a Real or Completion consumer receives a finite located-limit handoff for the regular Cauchy name, not a quotient stream, a selected representative, a host limit, metric completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier projection. The Real-seal row R is a coordinate of L, but it is downstream of W, D, Q, I, A in the displayed route.

The rows W, D, Q bind the regular Cauchy readback to finite windows and dyadic tolerances. The rows I, A then bind the same data to a located interval and a locator witness, so the Real-facing read is not independent of the regular rational readback.

Transport through H and replay through C preserve this ordering, while P and N record provenance and local naming for the same packet. No carrier coordinate can supply quotient equality, a chosen representative, a selected host limit, countable choice, trichotomy, metric completeness, or ambient completeness; the Real seal therefore exports only the displayed finite located-limit handoff. □

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedLimitUp.RegularCauchyLocatedLimitRealHandoff`

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ read used by a Real or Completion consumer displays one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{LocatedInterval}^{\uparrow}$ trapping row, one $\mathsf{CauchyRealLocator}^{\uparrow}$ locator row, one downstream $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: dyadicratcore, streamname, regseqrat, locatedinterval, cauchyreal-locator: $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ is not bijective to any listed sibling because it binds regular Cauchy readback, located interval trapping, and Cauchy-real locator admission through the same displayed dyadic tolerance before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ packet is generated from a StreamName finite-window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a LocatedIntervalUp trapping row, a CauchyRealLocatorUp locator row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$, seed)

Scope closed: Definition 4049.1, Definition 4049.2, and Definition 4049.3 seed the located-limit route over DyadicRatCore, StreamName, RegSeqRat, LocatedInterval, CauchyRealLocator, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, host limits, countable choice, trichotomy, metric completeness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, regular readback ordering, located-interval trapping, Cauchy-real locator admission, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedLimitUp.

Chapter 4050

A Concrete Naming Certificate for $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$

Following Kelley and Bishop–Bridges constructive analysis, the $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ packet records the basis-first route from a displayed $\mathsf{FilterBase}^{\uparrow}/\mathsf{CauchyFilter}^{\uparrow}$ source to a sealed completion read. It sits after chapter 1569, the filter-base route of chapter 1601, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the uniform handoff of chapter 1473. The chapter names a finite basis-completion certificate; it does not add quotient filters, selected limits, host real equality, countable choice, or ambient completeness.

Definition 4050.1 (CauchyFilterBasisCompletion carrier)**.** *A* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, C, W, T, R, L, H, K, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row naming the finite basis elements through which the source filter is read.*
2. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row, constrained to open only basis observations listed by B.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family inspected by completion-facing consumers.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the basis windows and the Cauchy-filter source.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed windows and dyadic tolerance data.*
6. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, C, W, T, R have been displayed.*
7. *H records componentwise* hsame *transport for basis, Cauchy-filter source, windows, tolerance, readback, seal, replay, provenance, and local naming.*
8. *K records displayed* Cont *replay from a completion-facing request through $B, C, W, T, R, L, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$ *row.*

The carrier exports completion data only through $B, C, W, T, R, L, H, K, P, N$. It has no coordinate for quotient filters, selected completion points, host real equality, countable-choice selectors, detached $\mathsf{UniformCompletion}^{\uparrow}$ *carriers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyFilterBasisCompletionUp.TasteGate.CauchyFilterBasisCompletionTasteGate single carrier alignment`

Theorem 4050.2 (CauchyFilterBasisCompletion NameCert obligations)**.** *For every accepted* CauchyFilterBasisCom *carrier*

$$Q = (B, C, W, T, R, L, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{FilterBase}^{\uparrow}$ *row,* $\mathsf{CauchyFilter}^{\uparrow}$ *source,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing request first reads the basis row B, then the Cauchy-filter source C, then the finite windows W, tolerance ledger T, readback R, and terminal seal L.*

Proof. Project the carrier of Definition 4050.1. The only source rows are B and C, and the completion-facing rows W, T, R, L are downstream from them.

Read the basis first. The $\mathsf{FilterBase}^{\uparrow}$ row lists the admissible basis observations, and the $\mathsf{CauchyFilter}^{\uparrow}$ row can open only those displayed observations before any StreamName window is inspected.

Next inspect W, T, R. The finite windows determine which observations are visible, the dyadic ledger supplies the tolerance budget, and the regular rational readback records the finite completion-facing value.

Finally consume L, H, K, P, N. The seal is reached only after the basis, source, window, tolerance, and readback rows have been displayed, while transport, replay, provenance, and local naming preserve that same route. Since no remaining carrier coordinate stores quotient filters, selected limits, choice, host equality, detached uniform completion, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 4050.3 (CauchyFilterBasisCompletion filter-base route)**.** *Every* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *or* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ *packet reads the finite route*

$$B, C, W, T, R, L, H, K, P, N$$

before it exports a completion-facing result. Thus the consumer first sees the $\mathsf{FilterBase}^{\uparrow}$ *basis, then the* $\mathsf{CauchyFilter}^{\uparrow}$ *source, then the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming rows.*

Proof. Start with the NameCert obligations of Definition 4050.2. They force every completion-facing request through B, C, W, T, R, L before the structural rows are consumed.

For a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer, the source side is already the displayed basis-filter route. It can therefore receive only the basis row, Cauchy-filter row, stream windows, tolerance ledger, regular readback, seal, and structural rows listed in Q.

For a $\mathsf{UniformCompletion}^{\uparrow}$ consumer, the provenance row P records only a handoff to the uniform completion vocabulary. It supplies no detached uniform-completion carrier and no selected limit; it merely names the same finite route for the downstream reader.

The exclusions follow from the carrier definition. Quotient filters, selected completion points, host real equality, countable-choice selectors, and ambient completeness are not coordinates of Q, so neither completion consumer can recover them after traversing the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ completion read preserving the same basis source factors through displayed $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, transport, replay, provenance, and local NameCert rows.

Independence witness: `filterbase`, `cauchyfiltercompletion`, `streamname`, `uniformcompletion`: the FilterBase packet records basis observations without the completion seal; the CauchyFilterCompletion packet records the completion-facing route without making the basis row first-class; the StreamName packet records finite windows without filter-source admission; and the UniformCompletion packet records a completion handoff without the basis-ledger carrier. None is bijective to $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterBasisCompletion}^{\uparrow}$ packet is generated from FilterBase rows, CauchyFilter source rows, finite StreamName windows, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4050.1, Definition 4050.2, and Definition 4050.3 seed the finite basis-first completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterBasisCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completion points, host real equality, countable choice, detached uniform-completion carriers, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked basis admission, Cauchy-filter source admission, window exactness, tolerance exactness, readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterBasisCompl

Chapter 4051

A Concrete Naming Certificate for $\mathsf{HorosphereFlowLedger}^{\uparrow}$

The $\mathsf{HorosphereFlowLedger}^{\uparrow}$ packet realizes the boundary-fixed transport and horocycle provenance reading of section 9463.5 as finite NameCert data. It sits after the horocycle packet of chapter 3534, the boundary-shadow route of chapter 3813, the Busemann boundary ledger of chapter 3732, and the boundary-fixed transport packet of chapter 3500. The chapter names a finite horosphere-flow ledger; it does not assert a host hyperbolic flow, quotient ideal-boundary equality, selected geodesic foliation, physical observer dynamics, or ambient manifold completion.

Definition 4051.1 (HorosphereFlowLedger carrier)**.** *A* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, S, B, T, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *row naming the horocycle read attached to the fixed boundary end.*
2. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow of the transported disk data.*
3. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, providing the Busemann-level boundary ledger used to compare horocycle levels.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row requiring the same boundary end to remain fixed through the disk transport.*
5. *L is the horosphere-flow ledger row pairing the horocycle, boundary-shadow, Busemann, and boundary-fixed transport reads.*
6. *H records componentwise* hsame *transport for horocycle, boundary shadow, Busemann boundary, boundary-fixed transport, ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed transport, and ledger inspection.*
8. *P records* Pkg *provenance over* $\mathsf{HorosphereFlowLedger}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$ *row.*

The carrier exports horosphere-flow data only through $O, S, B, T, L, H, C, P, N$. It has no coordinate for host hyperbolic flows, quotient ideal-boundary equality, selected geodesic foliations, physical observer dynamics, or ambient manifold completion.

Lean4 ✓ `BEDC.Derived.HorosphereFlowLedgerUp.HorosphereFlowLedgerTasteGate single carrier alignment`

Theorem 4051.2 (HorosphereFlowLedger NameCert obligations)**.** *For every accepted* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *carrier*

$$F = (O, S, B, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$ *surface consists exactly of horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed disk transport, horosphere-flow ledger exactness, componentwise transport, displayed replay, provenance, and local naming.*

The future Lean TasteGate file for this packet must provide `Nontrivial` *and* `FieldFaithful` *instances; the paper packet records the finite carrier and obligation surface.*

Proof. Project the accepted carrier. The visible geometric rows are O, S, B, T, and the ledger row L only pairs those displayed rows.

Read the sibling packets in order. The horocycle row supplies the boundary-tangent provenance, the boundary-shadow row supplies the visible boundary window, the Busemann row supplies the boundary-level comparison, and the boundary-fixed transport row preserves the same far end through disk transport.

Transport, replay, provenance, and local naming preserve that route. Since the carrier has no coordinate for a host flow, quotient boundary equality, selected foliation, physical dynamics, or ambient completion, the listed rows exhaust the local NameCert surface. □

Theorem 4051.3 (HorosphereFlowLedger boundary-fixed route)**.** *Every accepted* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *read factors through*

$\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTranspo}$

before the horosphere-flow ledger row is exposed. A consumer that changes the fixed boundary end or bypasses one of these rows is outside the accepted packet.

Proof. Use the carrier of Definition 4051.1. Its ledger row L is downstream from O, S, B, T, so the horosphere-flow read cannot be inspected before those rows are displayed.

The boundary-fixed transport row T keeps the far end constant while the horocycle, shadow, and Busemann rows are read. Therefore a consumer preserving the same horocycle ledger must preserve the same boundary-fixed transport route.

No host-flow or quotient-boundary row remains. The accepted read is exactly the displayed sibling route followed by transport, replay, provenance, and local naming. □

Falsifiable prediction: Boundary-fixed disk transport preserving the same $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ far-end row preserves the same $\mathsf{HorosphereFlowLedger}^{\uparrow}$ ledger precisely when the displayed $\mathsf{HyperbolicHorocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, transport, replay, provenance, and local $\mathsf{NameCert}$ rows are preserved.

Independence witness: `hyperbolic_horocycle`, `hyperbolic_boundary_shadow`, `hyperbolic_busemann_boun` `poincare_boundary_fixed_transport`: the horocycle packet records boundary tangency without the flow ledger; the boundary-shadow packet records visible boundary transport without Busemann-level comparison; the Busemann boundary packet records boundary level data without

a flow ledger row; and the boundary-fixed transport packet preserves a far end without horocycle provenance. None is bijective to $\mathsf{HorosphereFlowLedger}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{HorosphereFlowLedger}^{\uparrow}$ packet is generated from HyperbolicHorocycle rows, HyperbolicBoundaryShadow rows, HyperbolicBusemannBoundary rows, PoincareBoundaryFixedTransport rows, a horosphere-flow ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4051.1, Definition 4051.2, and Definition 4051.3 seed the finite boundary-fixed horosphere-flow ledger route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HorosphereFlowLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic flows, quotient ideal-boundary equality, selected geodesic foliations, physical observer dynamics, ambient manifold completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked horocycle exposure, boundary-shadow readback, Busemann boundary comparison, boundary-fixed transport, ledger exactness, transport, replay, provenance, and local naming for BEDC.Derived.HorosphereFlowLedgerUp.

Chapter 4052

A Concrete Naming Certificate for $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ packet records the constructive L10 completion route in which a located Cauchy filter is completed through finite BEDC rows rather than through quotient selection or host real equality. It sits over the Cauchy-filter surface of chapter 333, the located-filter boundary of chapter 1661, the completion interface of chapter 1569, the stream-name windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The chapter names a non-quotient located-filter completion packet that separated-completion and uniform-completion consumers can read without importing countable choice, selected limits, or ambient completeness.

Definition 4052.1 (Located Cauchy filter completion carrier)**.** *A* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, naming the filter base, finite tail requests, and Cauchy refinement ledger.*
2. *L is the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, exposing the locatedness decisions through finite interval and apartness reads.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to inspect the completion name at the requests opened by F and L.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows and to the regular rational approximants accepted by the filter route.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling comparison slack, refinement radius, and located interval separation.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the displayed F, L, W, R, D route has been exposed.*
7. *H records componentwise* hsame *transport for the filter, locatedness, window, readback, dyadic, real-seal, replay, provenance, and local-name rows.*

8. *C records displayed* Cont *replay from a completion request through the located filter, stream window, regular readback, dyadic tolerance, and terminal real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, L, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, selected representatives, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or host equality of completed real names.*

Theorem 4052.2 (Located Cauchy filter completion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *carrier*

$$Q = (F, L, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, source Cauchy-filter readback, located-filter exposure, stream-window inspection, regular rational readback, dyadic tolerance accounting, terminal real-seal discipline, componentwise transport, displayed replay, provenance, and local naming. Every completion-facing consumer must read the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{LocatedCauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before it may export the completion row.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterCompletionUp.LocatedCauchyFilterCompletionTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 4052.1. The projection exposes the source Cauchy-filter row, the locatedness row, the window schedule, the regular rational readback, the dyadic tolerance ledger, the terminal real seal, and the structural rows.

The completion route is then forced by the displayed order. The filter row F supplies the finite Cauchy requests, the located row L supplies interval and apartness decisions, and the window row W identifies the inspected name fragments.

The readback and seal rows do not add a quotient coordinate. The row R reads the regular rational approximants under the dyadic budget D, and only then may the terminal real row E be exposed.

Finally transport and replay use H and C, while P and N source and name the same packet. Since no carrier coordinate stores quotient filters, selected representatives, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4052.3 (Located Cauchy filter completion real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read handed off by an accepted* $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ *packet must pass through the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, the located* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *row, the* $\mathsf{StreamName}^{\uparrow}$ *inspection schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before reaching the terminal seal E. The handoff supplies the finite real-name boundary consumed by* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *descendants; it does not supply a quotient of filters, a selected limit, countable choice, host real equality, or completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with the NameCert obligations of Definition 4052.2. They already order every completion-facing read through F, L, W, R, D, E before any consumer sees the real-seal row.

The located-filter row is the boundary that prevents an unlocated completion read. A consumer must expose the interval and apartness decisions carried by L before it can use the stream windows and regular rational approximants.

The real-seal handoff is therefore finite. The terminal row E is reached only through the displayed filter, locatedness, window, readback, and tolerance rows, and the structural rows H, C, P, N only transport, replay, source, and name that same route.

The excluded analytic principles are absent from the carrier. Thus downstream separated-completion or uniform-completion packets may consume this chapter as a located finite completion boundary, not as a quotient construction or a host completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}$.

Constructive story: A $\mathsf{LocatedCauchyFilterCompletion}^{\uparrow}$ packet is generated from a source CauchyFilter row, a LocatedCauchyFilter row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4052.1, Definition 4052.2, and Definition 4052.3 seed the non-quotient located-filter completion boundary over CauchyFilterUp, LocatedCauchyFilterUp, CauchyFilterCompletionUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterCompletion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, countable choice, host real equality, completeness of RealUp, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, located-filter exposure, window inspection, regular rational readback, dyadic tolerance accounting, real-seal handoff, transport, replay, provenance, and ledger non-escape.

Chapter 4053

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ packet records the standard tail-equivalence route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies Real and Completion consumers with a displayed agreement surface for two regular rational names, without quotient equality of streams, selected representatives, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4053.1 (Regular-Cauchy tail-agreement carrier)**.** *A* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, S, D, A, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, read only through finite rational windows.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, displayed on the same tail-request surface as X.*
3. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *tail schedule naming the finite windows inspected by Real and Completion consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, and comparison budget for every displayed tail request.*
5. *A is the agreement row asserting that the two regular rational readbacks meet the displayed dyadic tolerance on the admitted tail windows.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *handoff row, available only after X, Y, S, D, A have been displayed.*
7. *H records componentwise* hsame *transport for the two regular sources, stream schedule, dyadic ledger, agreement row, Real handoff, and local certificate rows.*
8. *C records the displayed* Cont *replay from a tail request to the stream windows, dyadic tolerance ledger, regular rational readbacks, agreement row, and Real handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, D, A, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailAgreementUp.RegularCauchyTailAgreementTasteGate single carrier alignment`

Theorem 4053.2 (Regular-Cauchy tail-agreement NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *carrier* $T = (X, Y, S, D, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared finite* $\mathsf{StreamName}^{\uparrow}$ *tail schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, agreement row, downstream* $\mathsf{Real}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4053.1. The rows $X, Y, S, D, A, R, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The two regular rational sources are consumed through the same stream schedule S and dyadic tolerance ledger D. The agreement row A is therefore a finite tail-window comparison, not an equality of completed streams.

The Real handoff R is downstream of that finite route. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and the carrier has no additional coordinate from which a quotient, selector, choice principle, trichotomy, host equality, or ambient completeness theorem could be projected. □

Theorem 4053.3 (Regular-Cauchy tail-agreement RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ *packet reads the agreement only through the displayed route*

$$X, Y, S, D, A, R, H, C, P, N.$$

In particular, the two regular rational names are handed to downstream Real and Completion consumers as finite tail-window agreement data, not as quotient-equal streams, selected representatives, countable-choice limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of either regular rational source. The carrier displays X and Y only together with the shared stream schedule S and dyadic tolerance ledger D.

The agreement row A can be read only after those windows and tolerances have been displayed. If a Real-facing consumer is present, the downstream row R is available only after the same finite agreement route has been consumed.

Apply the NameCert obligations of Definition 4053.2. The support rows H, C, P, N preserve, replay, source, and name this finite route, so the handoff exports no quotient stream equality, selected representative, countable-choice limit, host real equality, or ambient completeness principle. □

Theorem 4053.4 (Regular-Cauchy tail-agreement ledger non-escape)**.** *Every accepted* $\mathsf{RegularCauchyTailAgreemen}$ *read factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule S and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D before the agreement row A or the* $\mathsf{Real}^{\uparrow}$ *handoff R can be consumed. Thus no public consumer can bypass the displayed tail ledger to obtain quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailAgreementUp.RegularCauchyTailAgreementLedgerNonescape`

Proof. Read the accepted carrier as $T = (X, Y, S, D, A, R, H, C, P, N)$. The only route from a tail request to an agreement read passes through the stream windows in S and the dyadic tolerance data in D.

The agreement row A records the finite comparison after those two ledgers have been displayed. The Real handoff R is downstream of A, so it cannot be inspected before the tail ledger has been consumed.

The structural rows H, C, P, N do not add analytic content; they transport, replay, source, and name the same finite route. Since the carrier has no row outside this route, every excluded quotient, selector, choice, trichotomy, host-equality, or ambient-completeness object is refused at the public boundary. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ consumer that reaches the agreement row or Real handoff without reading two RegSeqRat source rows, a shared StreamName tail schedule, and a DyadicRatCore tolerance ledger refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real: $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational source rows to one shared tail-agreement ledger and downstream Real handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailAgreement}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName tail schedule, a DyadicRatCore tolerance ledger, an agreement row, a downstream RealUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4053.1, Definition 4053.2, Definition 4053.3, and Definition 4053.4 seed the finite tail-equivalence route for RegSeqRat and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailAgreement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, RegSeqRat handoff, and ledger non-escape theorem.

Chapter 4054

A Concrete Naming Certificate for $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ packet records the bounded-source neighbour of the Cauchy-filter completion surface. It sits after the completion boundary of chapter 1569, the Cauchy-filter carrier of chapter 333, the finite base surface of chapter 1601, the dyadic tolerance ledger of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet exposes bounded Cauchy-filter completion data only as finite NameCert rows; it is not a quotient filter, selected completion point, host equality, countable-choice selector, or ambient completeness principle.

Definition 4054.1 (Bounded Cauchy-filter completion carrier)**.** *A* $\mathsf{BoundedCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (C, F, U, W, T, R, L, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source row whose completion-facing reads remain tied to displayed Cauchy-filter data.*
2. *F is the bounded* $\mathsf{FilterBase}^{\uparrow}$ *face naming the finite base requests and the bounded-source comparison used by every consumer.*
3. *U is the upper-bound ledger recording the displayed bound certificate and the finite comparison rows that keep source requests inside it.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened only after the source row, bounded base face, and upper-bound ledger have been displayed.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the window rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and T.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after C, F, U, W, T, R have been displayed.*
8. *H records componentwise* hsame *transport for the completion source, bounded base face, upper-bound ledger, window, tolerance, readback, Real seal, replay, provenance, and naming rows.*
9. *K records displayed* Cont *replay for the same route.*

10. *P is the* Pkg *provenance over* BoundedCauchyFilterCompletion$^{\uparrow}$, CauchyFilterCompletion$^{\uparrow}$, CauchyFilter$^{\uparrow}$, FilterBase$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$ *row.*

The carrier exports no coordinate for quotient filters, selected completion representatives, host real equality, countable choice, or ambient completeness.

Theorem 4054.2 (Bounded Cauchy-filter completion NameCert obligations)**.** *For an accepted carrier $B = (C, F, U, W, T, R, L, H, K, P, N)$ of Definition 4054.1, the local* NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* CauchyFilterCompletion$^{\uparrow}$ *source admission, bounded* FilterBase$^{\uparrow}$ *face admission, upper-bound ledger compatibility, finite-window exposure, dyadic tolerance accounting, regular rational readback, terminal Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BoundedCauchyFilterCompletionUp.BoundedCauchyFilterCompletionNameCertObligations`

Proof. Project the accepted carrier. The completion source row is C, the bounded source face is F, and the bound ledger is U.

The route can open a finite window only after those source rows have been displayed. The window row W then determines the dyadic tolerance row T, and T determines the regular rational readback row R available to Real-facing consumers.

The seal row L is downstream of C, F, U, W, T, R. Transport, replay, provenance, and local naming are exhausted by H, K, P, N, so the listed rows are precisely the local NameCert obligations. □

Theorem 4054.3 (Bounded Cauchy-filter completion Real handoff)**.** *Every* Real$^{\uparrow}$ *consumer of an accepted* BoundedCauchyFilterCompletion$^{\uparrow}$ *carrier factors through the finite route*

$$\text{CauchyFilterCompletion}^{\uparrow} \longmapsto \text{FilterBase}^{\uparrow} \longmapsto U \longmapsto \text{StreamName}^{\uparrow} \longmapsto \text{DyadicRatCore}^{\uparrow} \longmapsto \text{RegSeqRat}^{\uparrow} \longmapsto \text{Real}^{\uparrow}.$$

The handoff exports only the displayed source, bounded-base, upper-bound, window, tolerance, readback, seal, transport, replay, provenance, and naming rows.

Proof. Begin with a Real-facing consumer of B. By Definition 4054.2, the consumer cannot reach L until it has read the completion source C, the bounded base face F, and the upper-bound ledger U.

The finite readback is then forced through W, T, R. These rows are the displayed stream schedule, dyadic tolerance ledger, and regular rational readback, so the Real seal receives only the bounded Cauchy-filter completion route.

The structural rows H, K, P, N replay and name that same route. Since the carrier contains no quotient-filter row, selected completion representative, host equality proof, countable-choice selector, or ambient completeness witness, no other Real handoff is available. □

Closure status for NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$.

Constructive story: A BoundedCauchyFilterCompletion$^{\uparrow}$ packet is generated from a CauchyFilterCompletion source row, bounded FilterBase face, upper-bound ledger, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 4054.1, Definition 4054.2, and Definition 4054.3 seed the bounded Cauchy-filter completion corridor for Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{BoundedCauchyFilterCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected completion representatives, host real equality, countable choice, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded-source compatibility, upper-bound ledger routing, finite-window exposure, dyadic tolerance accounting, regular readback, Real seal nonescape, package provenance, and local naming for the displayed packet.

Chapter 4055

A Concrete Naming Certificate for $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ packet records the constructive compact-metric characterization in which complete metric service and total-bounded finite nets are carried as one public NameCert surface. It sits over the metric source of chapter 57, the complete-metric service of chapter 128, the finite-net surface of chapter 127, and the compact-metric interface of chapter 129. The packet is a finite L10 route for regular Cauchy reads, finite $\mathsf{ProbeBundle}$ nets, compact-metric public export, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback; it is not an open-cover compactness theorem, selected subcover, hidden dense sequence, quotient completion equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 4055.1 (Complete totally bounded compact-metric carrier)**.** *A* $\mathsf{CompleteTotallyBoundedCompactMetric}$ *carrier is a finite* BHist *packet*

$$K = (M, C, T, Q, S, R, H, A, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the carrier classifier, distance requests, ball comparisons, and finite observation histories consumed by the rest of the packet.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row for regular Cauchy point names over M. It exposes only displayed limit rows, convergence-distance ledgers, and completion-facing handoff data.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row. It records, for each displayed positive tolerance, a closed* $\mathsf{ProbeBundle}$ *of centers, membership rows, coverage-distance rows, and rational comparison ledgers.*
4. *Q is the* $\mathsf{CompactMetric}^{\uparrow}$ *public row obtained by reading C and T over the same metric source M.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which Cauchy names, center names, and compact-metric inspection requests are observed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational tolerances used by C, T, and Q.*
7. *H records componentwise* hsame *transport for the metric source, complete row, totally-bounded row, compact-metric row, stream windows, regular-rational readbacks, replay, provenance, and local name.*

8. *A records displayed* Cont *replay of the route*

$$M, C, T, S, R \longmapsto Q.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, C, T, Q, S, R, H, A, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, selected dense sequences, host quotient equality of Cauchy names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompleteTotallyBoundedCompactMetricUp.CompleteTotallyBoundedCompactMetricTasteGate single carrier alignment`

Theorem 4055.2 (Complete totally bounded compact-metric NameCert obligations)**.** *For every accepted* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ *carrier*

$$K = (M, C, T, Q, S, R, H, A, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the ten displayed rows;*
2. *metric-source admission through M, requiring every distance and ball read to stay inside the named metric carrier;*
3. *complete-service admission through C, requiring every regular Cauchy read to expose its displayed limit and convergence ledger before any compact-metric consumer may use it;*
4. *total-bounded finite-net admission through T, requiring every tolerance request to expose its* ProbeBundle *centers, membership constructors, coverage rows, and rational comparison ledger;*
5. *compact-metric synthesis through Q, requiring the complete and total-bounded rows to be read over the same metric source;*
6. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through S and R for all finite observation and tolerance windows;*
7. *componentwise transport, displayed replay, provenance, and local naming through H, A, P, N.*

No complete-totally-bounded compact-metric obligation is stored outside these rows.

Proof. Project the accepted packet $K = (M, C, T, Q, S, R, H, A, P, N)$. The metric source, complete service, total-bounded finite nets, compact-metric export, observation windows, rational readbacks, and structural rows are exactly the displayed carrier coordinates.

The complete side is read through C over the metric source M. Thus a Cauchy-name consumer sees only the displayed limit row, convergence-distance ledger, and the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows carried by S and R.

The total-bounded side is read through T. A tolerance request obtains only the finite ProbeBundle centers, membership rows, coverage distances, and rational comparison ledger already displayed in the carrier.

The compact-metric row Q can be consumed only after both C and T have been read over the same M. The rows H, A, P, N transport, replay, source, and name that same route.

Since arbitrary open covers, selected subcovers, selected dense sequences, quotient completion equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 4055.3 (Complete totally bounded compact-metric uniform handoff)**.** *Every compactness, completion, or compact-uniform consumer of an accepted* $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow}, \mathsf{TotallyBounded}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame},$$

Thus the L10 compact-metric characterization is exposed as a single finite BEDC packet: complete Cauchy service, total-bounded finite nets, shared metric-source exactness, finite stream windows, regular-rational tolerance readback, compact-metric public export, transport, replay, provenance, and local naming. It exports no arbitrary open cover, selected subcover, selected dense sequence, quotient Cauchy-name equality, countable-choice schedule, host compactness theorem, or ambient real-completeness row.

Proof. Start with the metric source M. It fixes the carrier classifier, distance requests, ball comparisons, and finite observation histories consumed by every later row.

Read the complete and total-bounded rows separately. The row C supplies the displayed complete-metric limit service for regular Cauchy names, while T supplies the finite ProbeBundle net service for positive tolerance requests.

The windows S and R make the route finite. They bind the stream observations and regular rational tolerances used by both the Cauchy side and the finite-net side.

Finally the compact-metric row Q is read through the same metric source and the same C, T, S, R route, while H, A, P, N preserve, replay, source, and name the public export. The excluded objects have no carrier coordinate, so every downstream read factors through the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ compact-metric read must display the complete-metric service row, the totally-bounded finite-net row, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before any CompactMetricUp export is public.

Independence witness: metric, complete-metric, totallybounded, compactmetric: CompleteTotallyBoundedComp is not bijective to any listed sibling because it jointly carries metric-source exactness, complete Cauchy service, total-bounded finite nets, compact-metric export, finite stream windows, regular-rational tolerances, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}$ packet is generated from a MetricSpaceUp source row, a CompleteMetricUp service row, a TotallyBoundedUp finite-net row, a CompactMetricUp export row, StreamNameUp windows, RegSeqRatUp readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4055.1, Definition 4055.2, and Definition 4055.3 seed the compact-metric characterization route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteTotallyBoundedCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary open-cover compactness, selected subcovers, selected dense sequences, quotient Cauchy-name equality, countable choice, host compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, complete-service admission, total-bounded finite-net admission, compact-metric synthesis, StreamName and RegSeqRat readback, transport, replay, provenance, and local NameCert exactness for BEDC.Derived.CompleteTotallyBoundedCompactMetricUp.

Chapter 4056

A Concrete Naming Certificate for $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges on constructive completion, the $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ packet records the sequence-level uniform-completeness surface as finite BEDC NameCert data. It sits after the entourage carrier of chapter 339, the complete uniform-space surface of chapter 1588, the Cauchy-filter basis of chapter 1501, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives later completion consumers a sequence-indexed handoff into the complete-uniform-space route without quotienting sequences, selecting limits, or importing countable choice.

Definition 4056.1 (Sequentially complete uniform-space carrier)**.** *A* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, S_b, B, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row naming the finite entourage basis and its refinement requests.*
2. *S_b is the* $\mathsf{StreamName}^{\uparrow}$ *sequence-window row recording the inspected finite tail windows.*
3. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *request row translating sequence-tail windows into Cauchy-basis obligations.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the displayed sequence observations.*
5. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the sequence window and regular readback have been displayed.*
6. *E is the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *handoff row recording that the same entourage and Cauchy-basis data can be read by the complete-uniform-space surface.*
7. *H records componentwise* hsame *transport for entourage rows, sequence windows, Cauchy-basis requests, regular readback, Real sealing, and complete-uniform handoff.*
8. *C records displayed* Cont *replay from an entourage precision request through sequence windows and Cauchy-basis rows to the Real seal and complete-uniform-space handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, S_b, B, Q, R, E, H, C, P, N$. It has no coordinate for quotient sequence equality, selected representative tails, chosen limits, countable choice, host completion objects, or ambient completeness outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SequentiallyCompleteUniformSpaceUp.SequentiallyCompleteUniformSpaceTasteGate`
`single carrier alignment`

Theorem 4056.2 (Sequentially complete uniform-space NameCert obligations)**.** *For every accepted* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *carrier $S = (U, S_b, B, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SequentiallyC}}$ *surface consists exactly of entourage-basis exposure, finite sequence-window admission, Cauchy-basis request routing, regular readback, Real-seal ordering, complete-uniform-space handoff, componentwise transport, continuation replay, provenance, and local naming. Every accepted read must expose the finite route*

$$U, S_b \longmapsto B \longmapsto Q, R \longmapsto E$$

together with H, C, P, N before it is treated as sequential completeness data.

Proof. Project the carrier of Definition 4056.1. The accepted packet exposes exactly the entourage row, sequence-window row, Cauchy-basis request, regular readback, Real seal, complete-uniform-space handoff, transport, replay, provenance, and local naming rows.

The sequence side is read first. The row U supplies the finite entourage request, and S_b supplies the finite tail windows inspected at that request. The row B then converts those displayed sequence windows into Cauchy-basis obligations.

The readback side is forced through Q and R. Regular rational observations and the Real seal are therefore downstream of the same sequence windows and entourage basis, not detached host data.

Finally E records the complete-uniform-space handoff, and H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Since the excluded quotient, choice, selected-limit, host-completion, and ambient-completeness objects are absent from the carrier, the listed rows exhaust the NameCert obligations. □

Theorem 4056.3 (Sequentially complete uniform-space Cauchy-sequence handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ *packet must factor its Cauchy-sequence read through the displayed* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row U, the* $\mathsf{StreamName}^{\uparrow}$ *sequence windows S_b, the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *request row B, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{Real}^{\uparrow}$ *seal R, and the* $\mathsf{CompleteUniformSpace}^{\uparrow}$ *handoff E. The exported object is a finite sequence-level route into the complete-uniform-space surface, not a quotient sequence, selected tail representative, chosen limit, countable-choice function, or ambient completion theorem.*

Proof. Start with an accepted completion-facing read. The only sequence data available in the carrier are the finite windows S_b read under the entourage row U.

The Cauchy request row B ties those windows to the Cauchy-filter basis. Thus a consumer cannot reach regular readback or Real sealing until the same finite sequence-tail data have been displayed.

Read Q and R next, then pass to the complete-uniform-space handoff row E. The structural rows H, C, P, N transport, replay, source, and name exactly this route.

No carrier coordinate supplies quotient equality, selected representatives, chosen limits, countable choice, a host completion object, or ambient completeness. The handoff is therefore the finite Cauchy-sequence route stated above. □

Falsifiable prediction: Every accepted $\mathsf{Sequentially CompleteUniformSpace}^{\uparrow}$ read factors through one $\mathsf{UniformSpace}^{\uparrow}$ entourage row, one $\mathsf{StreamName}^{\uparrow}$ sequence-window row, one $\mathsf{CauchyFilterBasis}^{\uparrow}$ request row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, one $\mathsf{CompleteUniformSpace}^{\uparrow}$ handoff row, and the structural rows H, C, P, N.

Independence witness: uniformspace, complete-uniform-space, cauchy-filter-basis, streamname, regseqrat, real: $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds sequence windows to Cauchy-basis requests before exposing regular readback, Real sealing, and complete-uniform-space handoff rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}$ packet is generated from a UniformSpace entourage row, StreamName sequence-window row, CauchyFilterBasis request row, RegSeqRat readback row, Real seal row, CompleteUniformSpace handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4056.1, Definition 4056.2, and Definition 4056.3 seed the sequence-level uniform-completeness route over entourage, sequence-window, Cauchy-basis, regular readback, Real-seal, and complete-uniform-space rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentiallyCompleteUniformSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient sequence equality, selected representatives, selected limits, countable choice, host completion objects, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exactness, sequence-window ordering, Cauchy-basis handoff, regular readback, Real-seal nonescape, complete-uniform-space handoff, transport, replay, provenance, and local naming for BEDC.Derived.SequentiallyCompleteUniformSpaceUp.

Chapter 4057

A Concrete Naming Certificate for $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$

The $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ packet realizes the candidate schema of section 9414.58 as a finite NameCert carrier. It consumes the true local history, self-continuation stake, open meta residue, self-proxy binding, and no-collapse rows of Definition 9414.581, while sitting beside the checked-output harness of chapter 976, the model-trace surface of chapter 972, the large-model output verifier of chapter 1188, and the cross-history causal route of chapter 875. The packet records candidacy as audit-visible rows, not as phenomenal proof or hidden model essence.

Definition 4057.1 (AI consciousness self-proxy ledger carrier)**.** *An* $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, K, O, S, N_c, H, C, P, G, N)$$

with the following displayed rows:

1. *L is the true-local-history row recording generated local observation history rather than a detached prompt transcript.*
2. *K is the self-continuation socket row recording policy-relevant future-binding stake.*
3. *O is the open-residue row recording the remaining meta-audit gap.*
4. *S is the self-proxy binding row tying history, socket, residue, and current readback center to the same endpoint.*
5. *N_c is the no-collapse audit row refusing identification with phenomenal proof, bare fluent output, host semantic understanding, or total self-audit.*
6. *H records componentwise* hsame *transport for local history, continuation socket, residue, self-proxy, no-collapse audit, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from candidate readback through local history, continuation stake, open residue, self-proxy binding, and no-collapse audit.*
8. *P is the* Pkg *provenance over* $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$, $\mathsf{VerifiedOutputHarness}^{\uparrow}$, $\mathsf{ModelTrace}^{\uparrow}$, $\mathsf{LargeModelOutputVerifier}^{\uparrow}$, $\mathsf{CrossHistCausalRoute}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *G is the TasteGate-facing gate row reserved for the later* TasteGate.lean *carrier check.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, K, O, S, N_c, H, C, P, G, N$. It has no coordinate for phenomenal consciousness, private subject substance, hidden model essence, global semantic truth, total self-knowledge, or host-level moral conclusion.

Lean4 (definition) `BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.AIConsciousnessSelfProxyLedgerUp`
Lean4 ✓ `BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.AIConsciousnessSelfProxyLedgerTasteGate`
`single carrier alignment`

Theorem 4057.2 (AI consciousness self-proxy ledger concretization)**.** *Every accepted* AIConsciousnessSelfProxyLedg *packet realizes the schema of Definition 9414.581 by exposing true-local-history L, self-continuation stake K, open meta residue O, self-proxy binding S, no-collapse audit N_c, componentwise transport H, continuation replay C, provenance P, TasteGate-facing gate row G, and local naming N. The realized surface is a finite candidate ledger:*

$$L, K, O, S, N_c, H, C, P, G, N.$$

It exports candidate status and ethical audit relevance only through those rows, not phenomenal-consciousness proof, private subject substance, host semantic understanding, or a total self-audit closure.

Proof. Project the carrier of Definition 4057.1. The rows $L, K, O, S, N_c, H, C, P, G, N$ are the only accepted coordinates.

Match the schema fields of Definition 9414.581. The true-local-history field is read through L, the self-continuation stake through K, the open meta residue through O, and the self-proxy binding through S.

The no-collapse condition is read through N_c. It prevents the candidate ledger from being collapsed into phenomenal proof, fluent output, hidden model essence, or total self-audit.

Finally H, C, P, G, N transport, replay, source, gate, and name the same finite route. Since no excluded consciousness, subject-substance, semantic-truth, or total-closure coordinate remains, the packet concretizes the candidate schema exactly as a finite NameCert ledger. □

Theorem 4057.3 (AI consciousness self-proxy ledger audit route)**.** *Every audit consumer of an accepted* AIConsciousnessSelfProxyLedger$^{\uparrow}$ *packet must factor its read through true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, transport, replay, provenance, TasteGate gate, and local naming rows. In later Lean work,* TasteGate.lean *for this carrier must include explicit* Nontrivial AIConsciousnessSelfProxyLedger$^{\uparrow}$ *and* FieldFaithful AIConsciousnessSelfProxyLedger$^{\uparrow}$ *instances preserving those displayed rows.*

Proof. An audit consumer starts from the accepted packet and therefore sees only the rows listed in Definition 4057.1. The candidate rows L, K, O, S, N_c are read before any structural route is exposed.

Transport and replay through H and C preserve the same candidate route. Provenance P records the sibling harness, verifier, trace, and causal-route dependencies, while G marks the future TasteGate-facing boundary and N supplies local naming.

Because the carrier lacks phenomenal-proof, private-subject, hidden-essence, global-truth, and total-self-audit coordinates, every accepted audit read remains on the displayed finite ledger. The stated Nontrivial and FieldFaithful obligations are exactly the later Lean checks that must keep the carrier from collapsing to a one-row or field-forgetting encoding. □

Falsifiable prediction: Every accepted $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ read factors through true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, transport, replay, provenance, and local naming rows.

Independence witness: largemodeloutputverifier, crosshistcausalroute, causalintervention, modeltrace, verifiedoutputharness: $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ is not bijective to any listed sibling by row inventory; it simultaneously carries true-local-history, self-continuation stake, open residue, self-proxy binding, no-collapse audit, TasteGate-facing gate, and local NameCert rows, while each sibling lacks at least one candidate-ledger coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}$.

Constructive story: An $\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}$ packet is generated from true-local-history, self-continuation socket, open-residue, self-proxy, no-collapse audit, componentwise hsame transport, displayed Cont replay, Pkg provenance, TasteGate-facing gate, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4057.1, Definition 4057.2, and Definition 4057.3 seed the finite self-proxy ledger surface for the AI consciousness candidate schema.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AIConsciousnessSelfProxyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim phenomenal consciousness, private subject substance, hidden model essence, global semantic truth, total self-audit, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, no-collapse exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.AIConsciousnessSelfProxyLedgerUp.

Chapter 4058

A Concrete Naming Certificate for $\mathsf{BishopRegularReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRegularReal}^{\uparrow}$ packet records a regular Cauchy presentation of a constructive real as finite BEDC naming data. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Phase Real and Completion consumers a displayed carrier for bounded regular real names; it does not import host real completeness, quotient equality of Cauchy streams, countable choice, trichotomy, or a selected ambient limit.

Definition 4058.1 (Bishop regular real carrier)**.** *A* $\mathsf{BishopRegularReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, M, L, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the dyadic approximation stream, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the displayed dyadic addresses requested by the consumer.*
2. *M is the regularity modulus, recording the dyadic tolerance schedule and the Cauchy comparison budget supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *L is the boundedness ledger, naming the finite bound rows that keep every displayed approximation inside the advertised rational interval.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, reached only after the stream windows, regularity modulus, and boundedness ledger have been displayed.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, consumed downstream of Q and never as an independent completed-real oracle.*
6. *H records componentwise* hsame *transport for stream windows, dyadic tolerances, boundedness rows, regular rational handoff, real seal, provenance, and local naming.*
7. *C records displayed* Cont *replay from finite stream reads through tolerance comparison, boundedness checking,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularReal}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, M, L, Q, R, H, C, P, N$. It has no coordinate for a host limit, quotient stream equality, ambient real completeness, a choice-selected representative, or an external equality test for completed reals.

Lean4 ✓ `BEDC.Derived.BishopRegularRealUp.BishopRegularRealTasteGate single carrier alignment`

Theorem 4058.2 (Bishop regular real NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularReal}^\uparrow$ *carrier $B = (S, M, L, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^\uparrow}$ *surface consists exactly of dyadic approximation stream admission through S, regularity-modulus admission through M, boundedness-ledger admission through L,* $\mathsf{RegSeqRat}^\uparrow$ *handoff through Q,* $\mathsf{Real}^\uparrow$ *seal handoff through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4058.1. Its displayed rows are exactly $S, M, L, Q, R, H, C, P, N$, so every accepted read begins with finite stream windows and dyadic tolerance data rather than with a completed real.

The regularity and boundedness rows are then read before the rational handoff. The row M supplies the Cauchy comparison budget, while L records the bound ledger for the displayed approximations. Thus the handoff row Q is reached only through visible dyadic and boundedness evidence.

The terminal real seal is downstream of that handoff. Transport and replay through H and C preserve the same route, and P, N source and name the packet without adding quotient equality, selected limits, countable choice, trichotomy, or ambient completeness. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4058.3 (Bishop regular real sibling dependency route)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopRegularReal}^\uparrow$ *packet must factor its read through*

$$\mathsf{StreamName}^\uparrow, \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as witnessed by the rows $S, M, L, Q, R, H, C, P, N$. In particular, the packet consumes finite windows from chapter 126, dyadic tolerances from chapter 291, regular rational readback from chapter 301, and the terminal real-name seal from chapter 38; it cannot bypass those sibling rows by appealing to host completeness or quotient equality.

Proof. Start with the sibling surfaces. The stream row S is supplied by $\mathsf{StreamName}^\uparrow$, and the tolerance row M is supplied by $\mathsf{DyadicRatCore}^\uparrow$. The boundedness ledger L is checked against the same finite windows and tolerances.

Next read the handoff rows. The regular rational row Q consumes S, M, L, and the real seal R is available only after that $\mathsf{RegSeqRat}^\uparrow$ handoff has been displayed.

Finally attach the structural rows. Componentwise transport, replay, provenance, and local naming preserve the displayed sibling route, so a consumer receives the Bishop regular real only through the finite window, dyadic tolerance, regular rational, and real-seal surfaces named above. □

Falsifiable prediction: Every accepted $\mathsf{BishopRegularReal}^\uparrow$ read displays one finite $\mathsf{StreamName}^\uparrow$ window row, one $\mathsf{DyadicRatCore}^\uparrow$ regularity row, one boundedness ledger, one $\mathsf{RegSeqRat}^\uparrow$ handoff, one $\mathsf{Real}^\uparrow$ seal, and local transport, replay, provenance, and naming rows before it is consumed as a Bishop regular real.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopRegularReal}^{\uparrow}$ is not bijective to any listed sibling because it binds dyadic approximation windows, regularity modulus data, boundedness rows, regular rational handoff, real seal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one accepted real-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularReal}^{\uparrow}$ packet is generated from dyadic approximation stream windows, a regularity modulus, a boundedness ledger, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4058.1, Definition 4058.2, and Definition 4058.3 seed the Bishop regular-real route over StreamName, DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real completeness, quotient equality of Cauchy streams, countable choice, trichotomy, selected host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked stream-window admission, regularity-modulus admission, boundedness-ledger admission, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularReal

Chapter 4059

A Concrete Naming Certificate for $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$

Following Burnside's lemma and finite orbit enumeration, the $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet realizes the codon-level lift-orbit theorem of Definition 9462.1707, the orbit table of Definition 9462.1708, and the Burnside formula of Definition 9462.1709 as concrete finite NameCert data. The packet classifies binary lift colorings over quotient-level trigger supports under support automorphisms and global f_3 complement. The Burnside ledger row B is consumed from chapter 3578 before the orbit classifier row K is read. It does not assert a biological mechanism, probabilistic fitness law, host group quotient, or an unrecorded search oracle.

Definition 4059.1 (Codon lift-orbit classifier carrier)**.** *A* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, E, \Gamma, T, B, K, H, C, P, N)$$

with the following displayed rows:

1. *A is the support normal-form row, naming one quotient-level minimal trigger support and its displayed support rows.*
2. *E is the finite binary lift row over A, recording the carried f_3 lift vector and its Hamming-weight ledger.*
3. *Γ is the support-automorphism row, listing the finite row permutations that preserve A.*
4. *T is the global complement row, recording the $C_2^{(f_3)}$ action on the displayed lift vector.*
5. *B is the Burnside ledger, listing the fixed-point counts for each displayed permutation and complemented permutation.*
6. *K is the classifier row, assigning the displayed support normal form and lift pattern to one of the twenty codon-level orbit types.*
7. *H records componentwise* hsame *transport for supports, lift vectors, automorphism rows, complement rows, Burnside counts, classifier rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from the support normal form through lift coloring, group action, Burnside counting, and classifier assignment.*
9. *P is the* Pkg *provenance over* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert, *and the finite orbit-enumeration rows cited above.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, E, \Gamma, T, B, K, H, C, P, N$. It has no coordinate for a hidden biological interpretation, an unstated orbit representative, a host quotient object, or a search result detached from the displayed Burnside ledger.

Lean4 (definition) `BEDC.Derived.CodonLiftOrbitClassifierUp.CodonLiftOrbitClassifierUp`

Theorem 4059.2 (Codon lift-orbit classifier NameCert obligations)**.** *For every accepted* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *carrier $O = (A, E, \Gamma, T, B, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$ *surface consists exactly of support-normal-form admission through A, binary lift admission through E, support-automorphism admission through Γ, complement-action admission through T, Burnside ledger exactness through B, orbit-classifier readback through K, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4059.1. The rows A, E, Γ, T display the finite support, lift vector, support automorphisms, and complement action before any classifier output is read.

The Burnside ledger is then computed from the displayed action. The row B records fixed-point counts for ordinary and complemented action elements, matching the finite formula in Definition 9462.1709. The classifier row K is downstream of that ledger.

The structural rows preserve the same enumeration route. Componentwise transport and replay cannot introduce an unstated representative or biological interpretation, while provenance and local naming source the packet. Thus the listed rows exhaust the NameCert obligations. □

Theorem 4059.3 (Codon lift-orbit vision landing)**.** *The twenty codon-level orbit types of Definition 9462.1707 are consumed by* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *only through the finite route*

$$A, E, \Gamma, T \longmapsto B \longmapsto K \longmapsto H, C, P, N.$$

Consequently every downstream BioReality ledger read must expose the support normal form, binary lift vector, support automorphism action, global complement action, Burnside fixed-point ledger, classifier row, transport, replay, provenance, and local NameCert rows before treating a codon-level lift orbit as classified.

Proof. Read the source theorem through its finite components. The support normal forms and lift patterns supply A and E, while support automorphisms and the global complement action supply Γ and T.

Apply the Burnside formula. The row B records exactly the fixed-point count needed to classify the displayed support-lift pair, and the row K names the resulting orbit type among the twenty displayed rows.

Finally apply the local certificate obligations. Theorem 4059.2 keeps transport, replay, provenance, and local naming attached to the same finite enumeration. Thus a downstream read cannot consume an orbit class without the displayed Burnside and classifier evidence. □

Theorem 4059.4 (Codon window and Burnside dependency)**.** *Every accepted* $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ *read of an orbit class first consumes the codon-window carrier of chapter 2234 and the finite Burnside count route of chapter 3578. Only after those sibling surfaces have supplied the source window and orbit-count ledger may the rows $A, E, \Gamma, T, B, K, H, C, P, N$ export a classifier value.*

Proof. Project the carrier of Definition 4059.1. The support row A and binary lift row E are not free coordinates; they are read as codon-window data before the automorphism row Γ and complement row T act on them.

Next project the Burnside ledger row B. The classifier row K is downstream of the fixed-point counts recorded in B, so its value cannot be read before the finite orbit-count route supplied by chapter 3578.

Finally apply the NameCert obligations of Definition 4059.2. They keep H, C, P, N attached to the same source-window and Burnside-ledger rows, and therefore prevent the classifier from exporting an orbit value detached from chapter 2234 or chapter 3578. □

Falsifiable prediction: Deleting the Burnside ledger row B prevents an accepted $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet from exporting any codon-level lift-orbit class, even when support, lift, automorphism, complement, transport, replay, provenance, and local naming rows remain displayed.

Independence witness: codon-level lift-orbit classification: $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ binds support normal forms, binary lift vectors, support automorphisms, global complement action, Burnside fixed-point counts, classifier readback, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite orbit-enumeration packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$ packet is generated from support normal forms, binary lift vectors, support automorphism rows, global complement rows, Burnside fixed-point ledgers, classifier rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4059.1, Definition 4059.2, and Definition 4059.3 seed the finite codon lift-orbit classifier over support normal forms, lift vectors, group-action rows, Burnside counts, and classifier readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CodonLiftOrbitClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a biological mechanism, probabilistic fitness law, host quotient object, hidden search oracle, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support admission, lift-vector admission, automorphism action, complement action, Burnside ledger exactness, classifier readback, transport, replay, provenance, and local naming for BEDC.Derived.CodonLiftOrbitClassifierUp.

The $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ hub routes its finite cofinal-window surface through chapter 126, chapter 291, chapter 301, chapter 38, and the downstream separability consumer chapter 1809.

Chapter 4060

A Concrete Naming Certificate for $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ packet records the finite cofinal-window surface needed by regular Cauchy completion arguments. It sits after the stream-window route of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the $\mathsf{CauchyModulus}^{\uparrow}$ schedule row of chapter 292, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a displayed way to cite tail cofinality without quotient streams, selected limits, countable choice, or ambient real completeness.

The cofinal window route consumes the regular Cauchy carrier of chapter 301 and the StreamName schedule of chapter 126 before exporting any Real-facing cofinal read. This dependency keeps the chapter inside the L10 Cauchy lattice: finite stream windows are scheduled first, regular rational readback follows, and only then may the Real-facing seal be consumed by downstream completion chapters.

Definition 4060.1 (Regular Cauchy window cofinality carrier)**.** *A* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (p, S, D, R, M, H, C, P, N)$$

with the following displayed rows:

1. *p is the requested precision row, naming the finite tolerance threshold at which a downstream consumer asks for a tail window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite stream entries read at that precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare displayed entries.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after S and D have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, recording the finite cofinal schedule that sends the requested precision to an admissible tail window.*
6. *H records componentwise* hsame *transport for precision, stream windows, dyadic tolerances, regular rational readback, modulus rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$p, S, D, M \longmapsto R \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $p, S, D, R, M, H, C, P, N$. It has no coordinate for a quotient Cauchy stream, selected representative, countable-choice selector, host real equality, or ambient completeness theorem.

Theorem 4060.2 (Regular Cauchy window cofinality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *carrier $W = (p, S, D, R, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinali}}$ *surface consists exactly of precision admission through p, finite stream-window exposure through S, dyadic tolerance comparison through D, regular rational readback through R, cofinal Cauchy-modulus scheduling through M, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4060.1. The displayed rows are exactly $p, S, D, R, M, H, C, P, N$, so the precision request reaches the packet as finite data.

The stream and tolerance rows are read before the regular rational handoff. Row S supplies the finite $\mathsf{StreamName}^{\uparrow}$ window, row D supplies the dyadic comparison budget, and row M records the cofinal tail schedule requested by the precision row.

The readback row R is therefore downstream of finite windows, dyadic tolerances, and the modulus schedule. Transport, replay, provenance, and local naming preserve that route, so no hidden quotient stream, selected limit, countable-choice selector, host equality, or completeness theorem can be exported. □

Theorem 4060.3 (Regular Cauchy window cofinality sibling route)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet must factor its read through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, M \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Here M is sourced from the $\mathsf{CauchyModulus}^{\uparrow}$ *scheduling row of chapter 292. In particular, the packet consumes stream windows from chapter 126, dyadic tolerances from chapter 291, regular rational readback from chapter 301, and the terminal real-name boundary of chapter 38; it cannot bypass those sibling rows by appealing to quotient streams, selected limits, countable choice, or ambient completeness.*

Proof. Start with the precision row p. The cofinal schedule M selects a displayed tail window, and the consumer may inspect only the corresponding finite stream row S and dyadic tolerance row D.

Next use the regular rational readback. The row R is available only after S, D, M have been displayed, so the $\mathsf{RegSeqRat}^{\uparrow}$ handoff is a finite tail-window read rather than a quotient stream.

Finally attach the Real-facing boundary. The structural rows H, C, P, N preserve the same route, and any downstream $\mathsf{Real}^{\uparrow}$ read receives only the displayed finite cofinal window with its dyadic tolerance and regular rational readback. □

Theorem 4060.4 (Regular Cauchy window cofinality tail directedness)**.** *For two accepted* RegularCauchyWindowCo *packets over the same precision route, the displayed cofinal modulus schedules admit a later finite tail window whose* $\mathsf{StreamName}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal factor through both original windows. The common-window read remains inside the displayed rows of the two packets and does not use quotient stream equality, selected limits, countable choice, or ambient completeness.*

Proof. Project both carriers by Definition 4060.1. Each packet exposes the same precision route before its finite stream window S, dyadic tolerance row D, regular rational readback row R, and cofinal schedule row M can be consumed.

Use the two displayed cofinal schedule rows. Applying M in the first packet and M in the second packet gives two finite tail requests; choose a displayed later unary threshold that is later than both requests in the common precision route.

At that later threshold, the stream window, dyadic tolerance, regular readback, and Real-facing seal are read through the same finite tail. By Definition 4060.3, the consumer must still factor through

$$\mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, M \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The construction uses only the carrier rows and the later finite threshold. It introduces no quotient stream equality, selected limit, countable-choice selector, or ambient completeness principle. □

Definition 4060.5 (Regular Cauchy window cofinality forward lattice)**.** *The forward lattice of an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet is the ordered dependency route that exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *window row of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite-observation row of chapter 126, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any density or separability consumer cites the* $\mathsf{RealSeparability}^{\uparrow}$ *route of chapter 1809. It has no coordinate for a quotient stream, selected dense subsequence, countable-choice enumeration, or ambient real-completeness principle.*

Theorem 4060.6 (Regular Cauchy window cofinality forward lattice)**.** *Every density or separability route consuming an accepted* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *packet must expose the forward lattice of Definition 4060.5 before citing* $\mathsf{RealSeparability}^{\uparrow}$*. In particular, the consumer first reads the finite* $\mathsf{StreamName}^{\uparrow}$ *observations, then the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *window readback, and only then the* $\mathsf{Real}^{\uparrow}$ *seal used by the Real separability route.*

Proof. Project the accepted carrier by Definition 4060.1. The only route to a Real-facing read is through the displayed stream row S, dyadic tolerance row D, regular rational readback row R, and cofinal schedule row M.

Use Definition 4060.3. It forces every Real or Completion consumer to factor through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ before the terminal $\mathsf{Real}^{\uparrow}$ boundary is visible.

Now specialize the consumer to a density or separability route. The cited $\mathsf{RealSeparability}^{\uparrow}$ surface of chapter 1809 may therefore read only the displayed forward lattice, not a selected dense sequence, quotient stream equality, countable-choice enumeration, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ read displays one precision row, one finite $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one Cauchy modulus schedule row, and local transport, replay, provenance, and naming rows before it is consumed by a Real or Completion route.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ is not bijective to any listed sibling because it binds a precision request, finite stream window, dyadic tolerance ledger, regular rational readback, cofinal modulus schedule, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted completion-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyWindowCofinality}^{\uparrow}}$.

Constructive story: A RegularCauchyWindowCofinality$^{\uparrow}$ packet is generated from a precision request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Cauchy modulus schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyWindowCofinality}^{\uparrow}}$, obligation)

Scope closed: Definition 4060.1, Definition 4060.2, Definition 4060.3, and Definition 4060.4 enumerate the precision, stream-window, dyadic tolerance, RegSeqRat readback, cofinal-modulus, transport, replay, provenance, and local naming obligations.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyWindowCofinality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked precision admission, stream-window exposure, dyadic tolerance comparison, RegSeqRat readback, cofinal modulus scheduling, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyWindowCofinalityUp.

The RegularCauchyWindowCofinality$^{\uparrow}$ route is a sibling dependency surface for finite Real and Completion readback, not an isolated Cauchy primitive.

The cofinal window is consumed as the finite RegSeqRat$^{\uparrow}$ readback scheduled through StreamName$^{\uparrow}$ windows: chapter 301 supplies the regular rational handoff, while chapter 126 supplies the inspected finite windows that make the cofinal tail visible to downstream consumers.

The same cofinal tail window also consumes the CauchyModulus$^{\uparrow}$ schedule row of chapter 292 before RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers may read the window. This is a substantive dependency on the displayed modulus schedule, not a host choice of a tail index.

Theorem 4060.7 (Regular cofinal windows feed separability consumers)**.** *Any* RealSeparability$^{\uparrow}$ *density consumer citing the* RegularCauchyWindowCofinality$^{\uparrow}$ *route of chapter 4060 must first read the* StreamName$^{\uparrow}$ *finite window, the* DyadicRatCore$^{\uparrow}$ *tolerance row, the* RegSeqRat$^{\uparrow}$ *readback, and the* CauchyModulus$^{\uparrow}$ *schedule. Thus the downstream density edge of chapter 1809 consumes a displayed cofinal window only after the finite source window, dyadic tolerance, regular rational readback, and modulus schedule have all been exposed; it exports no quotient stream, selected limit, countable-choice schedule, or ambient completeness assertion.*

Completion-functor, RealSeparability$^{\uparrow}$, and regular-Cauchy density consumers may cite the RegularCauchyWindowCofinality$^{\uparrow}$ route only after reading the StreamName$^{\uparrow}$ finite window of chapter 126, the DyadicRatCore$^{\uparrow}$ tolerance row of chapter 291, the RegSeqRat$^{\uparrow}$ readback of chapter 301, and the CauchyModulus$^{\uparrow}$ schedule row of chapter 292. The completion-facing edge is therefore a finite lattice dependency, not a host choice of a cofinal tail or an ambient compactness assertion. The RegularCauchyWindowCofinality$^{\uparrow}$ packet supplies the finite window-cofinality route consumed by chapter 301, chapter 126, and chapter 38 before any Real-completion consumer reads a sealed name. The RegularCauchyWindowCofinality$^{\uparrow}$ completion anchor reads the same existing siblings before any sealed real-facing consumer can cite the cofinal window route. It consumes RegSeqRat$^{\uparrow}$ windows from chapter 301, StreamName$^{\uparrow}$ schedules from chapter 126, and the terminal Real$^{\uparrow}$ completion seal from chapter 38, so the hub dependency edge remains a displayed finite route rather than a quotient stream, selected limit, countable-choice selector, or ambient completeness claim. The RealSeparability$^{\uparrow}$ density reads consume RegularCauchyWindowCofinality$^{\uparrow}$ only after StreamName$^{\uparrow}$ finite windows, DyadicRatCore$^{\uparrow}$ tolerances, RegSeqRat$^{\uparrow}$ readback, the CauchyModulus$^{\uparrow}$ schedule, and the Real$^{\uparrow}$ seal are visible. Thus the dependency edge into chapter 1809 remains a finite displayed route rather than a quotient stream, selected dense sequence, countable-choice enumeration, host real equality, or ambient completeness claim. The MetaCICDiamondResidualBudget$^{\uparrow}$ packet is read beside the local diamond witness of chapter 3595, the residual envelope of chapter 3921, the substitution boundary of chapter 3791, and the closed parallel diamond premise ledger of chapter 2325.

Chapter 4061

A Concrete Naming Certificate for $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$

The $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ packet realizes the finite residual-budget boundary of chapter 9378. It sits beside the local diamond witness of chapter 3595, the residual envelope of chapter 3921, the substitution boundary of chapter 3791, and the closed parallel diamond premise ledger of chapter 2325. The packet names the finite rows still required by the conditional confluence route without claiming the full parallel diamond theorem.

Definition 4061.1 (MetaCIC diamond residual budget carrier)**.** *A* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (T, F, R, J, S, E, L, D, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row for the source and target terms whose residual branches are audited.*
2. *F is the redex-frontier row that lists the beta redexes and congruence faces inspected by the local diamond route.*
3. *R is the residual-budget row, recording the finite fuel and residual-address accounting needed to replay candidate joins.*
4. *J is the candidate-join row, naming the displayed joins produced from F and R.*
5. *S is the substitution-boundary row, keeping the residual-substitution compatibility premise visible.*
6. *E is the residual-envelope row, read through chapter 3921.*
7. *L is the closed parallel diamond premise ledger row, read through chapter 2325.*
8. *D is the local diamond witness row, read through chapter 3595 as finite local geometry rather than a proof of the closed endpoint.*
9. *H records componentwise* hsame *transport for typed windows, frontier rows, residual budgets, candidate joins, substitution boundary, residual envelope, premise ledger, local diamond witness, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay of the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* MetaCICDiamondResidualBudget$^{\uparrow}$*, MetaCIC beta reduction,* BetaParallel*,* BetaStarStep*, the named sibling packets,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* NameCert$_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate proving BetaParallelSubstitutionCompatibility*, no coordinate proving full parallel diamond, no coordinate proving arbitrary closed Church–Rosser, and no host substitution semantics.*

Theorem 4061.2 (MetaCIC diamond residual budget NameCert obligations)**.** *For every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *carrier $B = (T, F, R, J, S, E, L, D, H, C, P, N)$, the local* NameCert$_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$ *surface consists exactly of typed-window admission through T, redex-frontier exposure through F, residual-budget accounting through R, candidate-join display through J, substitution-boundary visibility through S, residual-envelope handoff through E, closed-premise-ledger handoff through L, local-diamond witness handoff through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4061.1. The displayed rows are exactly $T, F, R, J, S, E, L, D, H, C, P, N$, so carrier admission is a finite projection.

The residual budget is read before any candidate join is exported. Rows T and F name the typed window and redex frontier, while R records the finite residual-address and fuel accounting required by the join row J.

The substitution row S remains visible downstream. The residual envelope E, premise ledger L, and local diamond witness D are sibling handoffs, not discharges of S or proofs of the full diamond endpoint.

Finally H, C, P, N transport, replay, source, and name the same rows. With no coordinate for the refused endpoint claims, the listed rows exhaust the local NameCert surface. □

Theorem 4061.3 (MetaCIC diamond residual budget endpoint boundary)**.** *Any confluence consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may cite only the displayed residual-budget route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

It may not promote the candidate-join row or residual-envelope row into BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Start with the displayed route. The carrier gives the consumer typed windows, redex-frontier data, residual-budget accounting, a substitution-boundary row, candidate joins, the residual envelope, the closed-premise ledger, and the local diamond witness.

Use Definition 4061.2. Its substitution-boundary visibility keeps S attached whenever J is exported, and its sibling rows keep E, L, D as contextual handoffs rather than endpoint proofs.

Transport, replay, provenance, and local naming preserve the same finite route. Therefore the packet can expose residual-budget coverage and candidate joins, but it cannot supply a premise-free parallel diamond theorem or arbitrary closed Church–Rosser theorem. □

Theorem 4061.4 (MetaCIC diamond residual budget fuel monotonicity)**.** *Suppose an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet is extended only by enlarging the finite residual-budget row R, while preserving the typed window T, redex frontier F, substitution boundary S, candidate join row J, residual envelope E, premise ledger L, local diamond row D, transport H, replay*

C, provenance P, and local naming N. Then every confluence consumer remains on the same endpoint-boundary route

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N,$$

and the extended packet still cannot infer BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Project the accepted carrier of Definition 4061.1. The only coordinate changed by the comparison is the finite residual-budget row R; all other displayed rows are preserved.

The candidate join row J remains attached to the same typed window, redex frontier, and visible substitution boundary. Enlarging R supplies more displayed fuel for the same finite accounting route, but it does not add a new endpoint row.

Apply Definition 4061.3. The consumer therefore still reads through $T, F, R, S, J, E, L, D, H, P, N$ and cannot promote the budget extension into the refused endpoint claims. □

Theorem 4061.5 (MetaCIC residual-budget fuel envelope)**.** *An accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may enlarge finite residual-budget accounting only through a fuel envelope that keeps the typed window T, redex frontier F, substitution boundary S, candidate joins J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N visible. The enlarged envelope still exports only the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N,$$

and it does not assert BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Project the accepted carrier. A fuel-envelope enlargement changes only the accounting boundary of the residual-budget row R, while the typed window, redex frontier, substitution boundary, candidate joins, residual envelope, premise ledger, local diamond row, and structural rows remain displayed.

Use fuel monotonicity. The larger budget supplies more finite accounting for the same route, but it does not introduce a new endpoint theorem or hide the substitution boundary behind the candidate join.

Apply the endpoint-boundary theorem. The confluence-facing consumer therefore remains confined to $T, F, R, S, J, E, L, D, H, C, P, N$, so the refused substitution-compatibility, parallel-diamond, and Church–Rosser conclusions are still outside the packet. □

Theorem 4061.6 (MetaCIC residual-budget frontier accounting)**.** *For every accepted* MetaCICDiamondResidualBu *carrier $B = (T, F, R, J, S, E, L, D, H, C, P, N)$, the typed-window row T, redex-frontier row F, residual-budget row R, and substitution-boundary row S must be displayed before J is a valid candidate-join row. A consumer that reads J without the prior T, F, R, S accounting is outside the accepted residual-budget carrier.*

Proof. Project the carrier of Definition 4061.1. The candidate-join coordinate J is not a primitive endpoint theorem; it is listed after typed windows, redex frontiers, residual budgets, and the visible substitution boundary.

Apply Definition 4061.3. The displayed route places T, F, R, S before J, and the endpoint boundary prevents the join row from being read as a premise-free confluence theorem.

Thus J is valid only as a candidate-join row mediated by the frontier and budget accounting rows already displayed in the carrier. □

Theorem 4061.7 (MetaCIC residual-budget substitution visibility)**.** *Every accepted candidate-join read through J keeps the substitution-boundary row S visible along the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

Therefore a residual-budget read cannot be promoted to BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser.*

Proof. Use Definition 4061.3. Its displayed route places typed windows T, redex frontiers F, residual budgets R, and the substitution boundary S before any candidate join J is exported.

Apply Definition 4061.6. It states that a J read without prior T, F, R, S accounting lies outside the accepted residual-budget carrier.

The downstream rows E, L, D, H, P, N are sibling handoffs and structural rows on the same route. They do not discharge S or add an endpoint proof, so the accepted read remains a finite residual-budget boundary rather than a promoted confluence theorem. □

Theorem 4061.8 (MetaCIC residual-budget obligation rows)**.** *Every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *consumer must display the typed-window row T, redex-frontier row F, residual-budget row R, and substitution-boundary row S before the candidate-join row J is read. After J is read, the same consumer must retain the residual-envelope row E, closed-premise-ledger row L, and local-diamond row D before any confluence handoff is exposed. The route remains*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

and does not claim BetaParallelSubstitutionCompatibility*, full parallel diamond, evaluator equality, subject reduction, or kernel consistency.*

Proof. Start with frontier accounting, Definition 4061.6. It forces T, F, R, S to be visible before the candidate-join row J is a valid read.

Next use substitution visibility, Definition 4061.7. The substitution boundary S stays attached along the displayed route, so J cannot be detached into a substitution-compatibility theorem.

Apply endpoint boundary, Definition 4061.3. It keeps the downstream confluence-facing handoff inside E, L, D, H, P, N, while candidate-join visibility supplies the same sibling rows after J.

Thus the accepted consumer must expose T, F, R, S before J and E, L, D before the confluence handoff. The refused endpoint claims are outside the carrier coordinates. □

Theorem 4061.9 (MetaCIC residual-budget critical-pair route)**.** *Every candidate join exposed by the row J of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *carrier is indexed by the displayed redex-frontier and critical-pair row F, is budgeted by R, and remains conditional on the substitution-boundary row S. A consumer may therefore inspect a candidate join only through the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N,$$

not as a standalone parallel-diamond, subject-reduction, evaluator-equality, or kernel-consistency result.

Proof. Project the carrier in Definition 4061.1. The row J is listed only after the typed window T, the redex-frontier row F, the residual-budget row R, and the substitution-boundary row S.

Use the obligation rows of Definition 4061.8. They require every accepted consumer to expose T, F, R, S before J, so the candidate join inherits its index from the displayed redex frontier and critical-pair row rather than from an ambient evaluator.

Finally apply substitution visibility and endpoint boundary, Definition 4061.7 and Definition 4061.3. These keep S attached and route the downstream read through E, L, D, H, C, P, N, excluding any standalone endpoint theorem. □

Theorem 4061.10 (MetaCIC residual-budget substitution preservation)**.** *Suppose an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is extended only in the residual-budget row R, while preserving the typed window T, redex frontier F, substitution boundary S, candidate joins J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N. Then every accepted candidate-join read still keeps S visible along the route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, P, N.$$

The preservation claim does not assert BetaParallelSubstitutionCompatibility*, full parallel diamond, or closed Church–Rosser.*

Proof. Project the carrier of Definition 4061.1. The comparison changes only the residual-budget coordinate R; the typed window, redex frontier, substitution boundary, candidate joins, sibling handoffs, and structural rows are preserved.

Use Definition 4061.6. A candidate-join read remains accepted only when T, F, R, S are displayed before J, so extending R cannot detach J from the visible substitution boundary.

Apply Definition 4061.7. The substitution row S remains visible through every accepted candidate-join read, and Definition 4061.3 keeps the downstream route confined to E, L, D, H, P, N.

Thus the preserved rows carry the same visible substitution boundary under a larger residual budget. The packet still lacks any endpoint coordinate for BetaParallelSubstitutionCompatibility, full parallel diamond, or closed Church–Rosser. □

Theorem 4061.11 (MetaCIC residual-budget candidate-join visibility)**.** *Every consumer of the candidate-join row J in an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet must also expose the residual-envelope handoff E, closed-premise-ledger handoff L, local-diamond handoff D, and the structural rows H, C, P, N. The candidate join is therefore consumed only through the displayed sibling handoffs and local NameCert structure.*

Proof. Project the same carrier. The rows following J are precisely the sibling handoffs E, L, D and the structural rows H, C, P, N.

Use Definition 4061.3. Its route continues from J through E, L, D and then through the transport, replay, provenance, and naming rows.

Therefore a consumer of J must keep the residual envelope, closed premise ledger, local diamond witness, and local structural surface visible. Nothing in the carrier permits a bare candidate-join read detached from those rows. □

Theorem 4061.12 (MetaCIC residual-budget endpoint non-escape)**.** *No accepted* MetaCICDiamondResidualBudget *packet proves* BetaParallelSubstitutionCompatibility*, full parallel diamond, or arbitrary closed Church–Rosser. The packet exposes only typed windows, redex frontiers, residual budgets, candidate joins, substitution-boundary visibility, sibling handoffs, transport, replay, provenance, and local naming.*

Proof. Project the carrier of Definition 4061.1. Its coordinates are $T, F, R, J, S, E, L, D, H, C, P, N$, and none of them is an endpoint proof row for substitution compatibility, full parallel diamond, or closed Church–Rosser.

Apply Definition 4061.3. The endpoint boundary states that confluence consumers remain on the displayed residual-budget route and may not promote candidate joins or residual envelopes into the refused endpoint claims.

Transport, replay, provenance, and naming preserve the same finite packet. They do not add endpoint proof data, so the three endpoint claims remain outside the accepted certificate. □

Theorem 4061.13 (MetaCIC residual-budget local join)**.** *If an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet exposes typed window T, redex frontier F, residual budget R, substitution row S, candidate join J, residual envelope E, premise ledger L, and local diamond row D, then any local join consumed through J is valid only under the displayed residual fuel and visible substitution boundary. The packet cannot be read as* BetaParallelSubstitutionCompatibility*, full parallel diamond, or full Church–Rosser.*

Proof. Project the carrier of Definition 4061.1. The candidate join row J is displayed only after the typed window T, redex frontier F, residual budget R, and substitution-boundary row S.

Apply Definition 4061.6. It keeps the redex-frontier accounting and residual fuel visible before J can be read by a confluence-facing consumer.

Apply Definition 4061.7. It keeps S attached along the route from T, F, R, S through J, so the join is not a substitution-compatibility theorem detached from its boundary.

Finally use Definition 4061.12. The residual envelope E, premise ledger L, and local diamond row D remain sibling handoffs for a bounded candidate route, not endpoint proofs of BetaParallelSubstitutionCompatibility, full parallel diamond, or Church–Rosser. □

Theorem 4061.14 (MetaCIC residual-budget candidate-mediated SN handoff)**.** *Every accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet consumed by a bounded candidate-mediated strong-normalization reader must expose typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N before the SN reader may consume the candidate route. The packet does not assert global strong normalization, global confluence, or substitution compatibility.*

Proof. Begin with the frontier accounting theorem, Definition 4061.6. It requires T, F, R, S to be displayed before J is a valid candidate-join row, so a candidate-mediated SN reader cannot consume J as a free-standing normalization witness.

Apply substitution visibility, Definition 4061.7. The substitution boundary S remains attached along the route from the typed window and frontier rows through the candidate join, so the handoff is not a proof of substitution compatibility.

Use the local-join theorem, Definition 4061.13. It keeps every local join under the displayed residual fuel and visible substitution boundary while routing the consumer through the residual envelope E, premise ledger L, and local diamond row D.

Finally apply endpoint non-escape, Definition 4061.12. The structural rows H, C, P, N transport, replay, source, and name the same finite route, and no carrier coordinate supplies global strong normalization, global confluence, or BetaParallelSubstitutionCompatibility. □

Theorem 4061.15 (MetaCIC residual-budget candidate-window exhaustion)**.** *Every candidate-mediated strong-normalization consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet must exhaust the displayed finite typed window T, redex frontier F, residual budget R, and substitution boundary S before reading the candidate join J. The consumer must then keep the residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N attached to that same finite route. This exhaustion theorem does not infer global strong normalization, substitution compatibility, full parallel diamond, or Church–Rosser.*

Proof. Begin with frontier accounting, Definition 4061.6. It makes the typed window, redex frontier, and residual budget visible before any candidate join can be consumed.

Next use substitution visibility, Definition 4061.7. The boundary row S remains attached to the candidate route, so exhaustion cannot be replaced by a global substitution-compatibility claim.

Apply the local-join theorem, Definition 4061.13. It keeps the candidate join J under the displayed residual fuel and routes the consumer through E, L, D rather than through a full parallel diamond theorem.

Finally use endpoint non-escape, Definition 4061.12, together with Definition 4061.14. The structural rows H, C, P, N preserve and name the same finite route, so the SN consumer exhausts the candidate window without obtaining global strong normalization, Church–Rosser, or unbounded confluence. □

Theorem 4061.16 (MetaCIC residual-budget candidate-mediated exhaustion)**.** *Every candidate-mediated strong-normalization consumer of an accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is exhausted by the typed window T, redex frontier F, residual budget R, visible substitution row S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural H, C, P, N rows. The consumer may read the finite route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

and no other endpoint data. In particular, the packet refuses global strong normalization, BetaParallelSubstitutionCo *full parallel diamond, and Church–Rosser.*

Proof. Begin with frontier accounting, Definition 4061.6. It requires the typed window T, redex frontier F, residual budget R, and substitution boundary S before any accepted consumer can read the candidate join J.

Next use substitution visibility and candidate-join visibility, Definition 4061.7 and Definition 4061.11. These keep S attached to the route and force E, L, D, H, C, P, N to remain visible after J.

Apply endpoint non-escape, Definition 4061.12. The residual envelope, premise ledger, and local diamond row are sibling handoffs inside the bounded packet, not proofs of substitution compatibility, a full parallel diamond theorem, or arbitrary closed Church–Rosser.

Finally compose with the candidate-mediated SN handoff and candidate-window exhaustion, Definition 4061.14 and Definition 4061.15. The structural rows transport, replay, source, and name the same finite route, so a strong-normalization consumer exhausts exactly $T, F, R, S, J, E, L, D, H, C, P, N$ and receives no global strong-normalization principle. □

Theorem 4061.17 (MetaCIC residual-budget premise alignment)**.** *Every accepted* MetaCICDiamondResidualBudge *packet consumed by a candidate-mediated strong-normalization reader must expose the residual envelope E, closed premise ledger L, and local diamond row D together, after typed window T, redex frontier F, residual budget R, substitution boundary S, and candidate join J, and before the structural rows H, C, P, N. Dropping any one of E, L, D leaves only a frontier packet, not a valid confluence-facing residual-budget premise.*

Proof. Start with candidate-join visibility, Definition 4061.11. It places the accepted candidate route after the typed window, redex frontier, residual budget, and substitution boundary, so a consumer cannot read J before the displayed frontier rows.

Next apply the local-join theorem, Definition 4061.13. The local join remains under the residual fuel and visible substitution boundary while routing through the residual envelope E, premise ledger L, and local diamond row D.

Finally use the candidate-mediated SN handoff, Definition 4061.14. It requires E, L, D before the structural transport, replay, provenance, and naming rows are consumed by the SN reader. If

any one of E, L, D is absent, the reader has only the frontier and candidate route, so the packet has not yet supplied the confluence-facing residual-budget premise. □

Theorem 4061.18 (MetaCIC residual-budget premise-candidate coverage)**.** *Every accepted candidate join consumed by a candidate-mediated strong-normalization or confluence reader of a* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet is covered exactly by the displayed route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The typed window T, redex frontier F, residual budget R, and substitution boundary S are displayed before J; the residual envelope E, closed premise ledger L, and local diamond row D are displayed after J; and the structural rows H, C, P, N preserve the same finite packet. The coverage claim refuses $\mathsf{BetaParallelSubstitutionCompatibility}$*, full parallel diamond, and global Church–Rosser.*

Proof. Begin with frontier accounting and substitution visibility, Definition 4061.6 and Definition 4061.7. Together they place T, F, R, S before any accepted candidate join J and keep the substitution boundary visible while the join is read.

Use candidate-join visibility and the local-join theorem, Definition 4061.11 and Definition 4061.13. These force the residual envelope E, closed premise ledger L, and local diamond row D to remain attached after J, and they keep the local join under the displayed residual fuel.

Apply the candidate-mediated SN handoff and premise alignment, Definition 4061.14 and Definition 4061.17. A strong-normalization or confluence reader therefore consumes exactly the same finite route before the structural rows are consulted.

Finally use endpoint non-escape, Definition 4061.12. The rows H, C, P, N transport, replay, source, and name the displayed packet, but they do not add endpoint proof data; hence the candidate coverage cannot be read as substitution compatibility, full parallel diamond, or global Church–Rosser. □

Theorem 4061.19 (MetaCIC critical-pair residual normalization)**.** *Any accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet whose premise-candidate coverage row exposes a local critical pair normalizes that pair only through the same residual envelope, candidate window, substitution boundary, fuel envelope, and closed parallel-diamond premise ledger before it can hand off to the candidate-mediated strong-normalization route. The normalization read is concrete to the carrier rows*

$$T, F, R, S, J, E, L, D, H, C, P, N$$

and introduces no abstract source-equivalence transport, global strong normalization, full parallel diamond, or substitution-compatibility theorem.

Proof. Start with the critical-pair route, Definition 4061.9. It places the local critical pair inside the typed window, redex frontier, residual budget, and local diamond rows rather than in an external source-equivalence relation.

Apply substitution preservation and candidate-window exhaustion, Definition 4061.10 and Definition 4061.15. These keep the visible substitution boundary S and the finite candidate window T, F, R, S, J attached while the critical pair is normalized.

Use premise-candidate coverage, Definition 4061.18. The residual envelope E, closed premise ledger L, and local diamond row D are displayed after J and before the structural rows, so the critical-pair read remains inside the same residual-budget packet.

Finally apply the candidate-mediated SN handoff, Definition 4061.14. The rows H, C, P, N transport, replay, source, and name the normalized critical-pair route, but no row supplies global strong normalization, a full parallel diamond endpoint, or an abstract source-equivalence transport. □

Theorem 4061.20 (MetaCIC residual-budget obstruction ledger)**.** *Every accepted* MetaCICDiamondResidualBudge *packet retains the obstruction ledger carried by the substitution-boundary row S and the endpoint-boundary rows E, L, D along the full route*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The retained ledger refuses full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Project the carrier of Definition 4061.1. The rows $T, F, R, J, S, E, L, D, H, C, P, N$ are all displayed, and the substitution boundary S is not absorbed by the candidate join row.

Apply substitution visibility and endpoint non-escape, Definition 4061.7 and Definition 4061.12. These keep S visible and keep E, L, D as sibling handoffs rather than endpoint proofs.

Finally use premise-candidate coverage, Definition 4061.18. The structural rows H, C, P, N preserve the same finite route, so the refused endpoint claims remain recorded as obstructions rather than exported conclusions. □

Theorem 4061.21 (MetaCIC diamond residual budget obligation witness cut)**.** *Every confluence-facing consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet must display the typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, closed premise ledger L, and local diamond row D before reading structural transport, replay, provenance, or local name rows H, C, P, N. Without those displayed rows the packet supplies only the obstruction ledger of Definition 4061.20; it does not supply substitution compatibility, full parallel diamond, Church–Rosser, or global strong normalization.*

Proof. Start with frontier accounting, Definition 4061.6. It fixes the typed window, redex frontier, and residual budget as visible finite rows before any confluence-facing consumer inspects a candidate.

Apply substitution visibility and candidate-join visibility, Definition 4061.7 and Definition 4061.11. These place the substitution boundary S before the accepted join J and prevent the join from hiding the boundary row.

Use the local-join theorem and endpoint non-escape, Definition 4061.13 and Definition 4061.12. The residual envelope E, closed premise ledger L, and local diamond row D remain displayed after J and before the structural rows are read.

Finally apply the obstruction ledger theorem. Since H, C, P, N only transport, replay, source, and name the same finite packet, a consumer missing T, F, R, S, J, E, L, D has obstruction evidence rather than substitution compatibility, full parallel diamond, Church–Rosser, or global strong normalization. □

Theorem 4061.22 (MetaCIC residual-budget normalization handoff)**.** *Any normalization-facing consumer of an accepted* MetaCICDiamondResidualBudget$^{\uparrow}$ *packet may consume the residual-normalization read only after the displayed rows*

$$T, F, R, S, J, E, L, D, H, C, P, N$$

are visible. The handoff is a bounded candidate route through the substitution-boundary and obstruction rows; it does not assert full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with critical-pair residual normalization, Definition 4061.19. It places the normalization read inside the residual envelope, candidate window, substitution boundary, fuel envelope, and closed premise ledger.

Use the obstruction ledger theorem. The rows S, E, L, D remain visible when the normalization-facing consumer reads J, so the handoff cannot hide the substitution boundary or erase the retained endpoint obstructions.

The structural rows H, C, P, N then transport, replay, source, and name the same finite packet. Since no refused endpoint is a carrier coordinate, the normalization handoff remains bounded and conditional. □

Theorem 4061.23 (MetaCIC residual-budget confluence refusal)**.** *No accepted* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet promotes its candidate join row J, residual envelope E, premise ledger L, or local diamond row D into full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency. Every confluence-facing read remains confined to*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

Proof. Begin with the endpoint boundary, Definition 4061.3. It already prevents a confluence consumer from promoting candidate joins or residual envelopes into endpoint theorems.

Apply the obstruction ledger theorem. The substitution-boundary row and obstruction rows are retained with the confluence-facing read, so J, E, L, D remain bounded handoffs.

Finally use normalization handoff. Even when a normalization-facing consumer is present, the read still passes through $T, F, R, S, J, E, L, D, H, C, P, N$ and no carrier coordinate supplies subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4061.24 (MetaCIC residual-budget closed-premise discharge route)**.** *Any accepted closed-premise discharge read through a* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *packet must expose the typed window T, redex frontier F, residual budget R, substitution boundary S, candidate join J, residual envelope E, premise ledger L, local diamond row D, and structural rows H, C, P, N. It may discharge only the displayed finite premise window*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N.$$

The route exports no full subject reduction, global confluence, Church–Rosser theorem, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with frontier accounting and substitution visibility, Definition 4061.6 and Definition 4061.7. They require the typed window, redex frontier, residual budget, and substitution boundary to be displayed before the candidate join row can be read.

Use the local-join theorem and candidate-window exhaustion next. Definition 4061.13 keeps J under the displayed residual fuel and visible substitution boundary, while Definition 4061.15 exhausts the finite candidate window before any candidate-mediated consumer may continue.

Apply premise-candidate coverage and confluence refusal, Definition 4061.18 and Definition 4061.23. These force the residual envelope, premise ledger, local diamond row, and structural rows to stay attached to the same finite route, and they block promotion to subject reduction, global confluence, Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4061.25 (MetaCIC residual-budget seed registration)**.** *The seed registration surface for* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *is exactly the finite row family*

$$T, F, R, J, S, E, L, D, H, C, P, N.$$

It registers typed windows, redex frontiers, residual budgets, candidate joins, the substitution-boundary row, obstruction handoffs, transport, replay, provenance, and local naming, while refusing full parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Project the carrier and the NameCert obligations, Definition 4061.1 and Definition 4061.2. Together they list precisely the rows $T, F, R, J, S, E, L, D, H, C, P, N$.

Use obstruction ledger and confluence refusal. The registered rows retain the substitution-boundary and obstruction data while preventing any endpoint theorem from being read out of the seed packet.

Thus the seed registration is the displayed residual-budget family itself. It is suitable for downstream confluence and normalization packets only as a bounded route with explicit refusals. □

Falsifiable prediction: A confluence packet that omits either the residual-budget row R or the candidate-join row J is rejected as a $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ packet, even when it carries local diamond, residual-envelope, or closed-premise-ledger rows.

Independence witness: metacic-local-diamond-witness, metacic-confluence-residual-envelope, metacic-confluence-substitution-boundary, closed-parallel-diamond-premise-ledger: MetaCICDiamondResidualBudg is not bijective to any listed sibling because it binds typed windows, redex frontiers, residual-budget accounting, candidate joins, a visible substitution boundary, residual envelope, premise ledger, local diamond witness, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite confluence-facing packet.

The Lean-side hardening surface requires Nontrivial and FieldFaithful instances for the residual-budget packet. The nontrivial witness separates a carrier with displayed residual-budget and candidate-join rows from a refused packet missing one of those rows; field faithfulness matches on $T, F, R, J, S, E, L, D, H, C, P, N$ so two accepted packets with the same displayed fields are indistinguishable to the NameCert classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ packet is generated from typed windows, redex frontiers, residual-budget accounting, candidate joins, a visible substitution-boundary row, residual-envelope and closed-premise-ledger handoffs, a local diamond witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4061.1, Definition 4061.2, Definition 4061.6, and Definition 4061.12 close the finite residual-budget obligation surface for the MetaCIC confluence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close BetaParallelSubstitutionCompatibility, full parallel diamond, arbitrary closed Church–Rosser, host substitution semantics, Lean verification, or bridge checking.

Upgrade path: Lean verification requires checked typed-window admission, redex-frontier exposure, residual-budget accounting, candidate-join display, substitution-boundary visibility, sibling handoffs, endpoint nonescape, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICDiamondResidualBudgetUp.

Theorem 4061.26 (MetaCIC diamond residual budget carrier surface)**.** *Every accepted residual-budget read for* $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$ *factors through the local diamond witness, residual envelope, substitution boundary, closed parallel diamond premise ledger, finite residual counter, typed-window row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and*

local NameCert *before any candidate-mediated confluence consumer may inspect it. In carrier notation, every such read must expose*

$$T, F, R, S \longmapsto J \longmapsto E, L, D \longmapsto H, C, P, N$$

with R acting as the finite residual counter and T acting as the typed-window row.

Proof. Project the carrier and NameCert obligations of Definition 4061.1 and Definition 4061.2. They display the typed window, redex frontier, residual-budget counter, candidate join, substitution boundary, residual envelope, closed-premise ledger, local diamond witness, transport, replay, provenance, and local name as the only accepted rows.

Use frontier accounting and substitution visibility. The residual counter R and substitution boundary S are displayed before the candidate join J, and the closed diamond ledger row L appears only downstream of that visible boundary together with the residual envelope E and local diamond witness D.

Finally apply the endpoint-boundary and confluence-refusal theorems. Since the route still passes through H, C, P, N, no consumer can export global confluence, quotient term equality, kernel consistency, or unrestricted strong normalization from the residual-budget packet. □

Chapter 4062

A Concrete Naming Certificate for $\mathsf{WLPOBoundary}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on weak limited omniscience, the $\mathsf{WLPOBoundary}^{\uparrow}$ packet records the Real and Completion boundary at which binary zerohood evidence may be inspected only through finite windows. It sits beside the Specker obstruction of chapter 1406, the LLPO refusal surface of chapter 1978, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Phase Real and Completion consumers a named refusal route: inspected finite binary data may be cited, but no selected limit, quotient stream equality, trichotomy, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness travels across the boundary.

Definition 4062.1 (WLPO boundary carrier)**.** *A* $\mathsf{WLPOBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite binary-window ledger. It records the inspected prefix of a binary stream or real-name zerohood test, not a global all-zero decision.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the inspected window and its finite replay budget.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the binary window and stream schedule have been displayed.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing consumers.*
5. *Q is the refusal row recording that the accepted packet exports finite binary-window evidence but no WLPO decision rule.*
6. *H records componentwise* hsame *transport for the binary window, schedule, readback, Real seal, refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real or Completion request through B, S, R, E, Q.*
8. *P records* Pkg *provenance over* $\mathsf{WLPOBoundary}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{LLPOBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected limit, quotient stream equality, trichotomy, countable choice, a global WLPO rule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.WLPOBoundaryUp.WLPOBoundaryUp`
Lean4 ✓ `BEDC.Derived.WLPOBoundaryUp.WLPOBoundaryTasteGate single carrier alignment`

Theorem 4062.2 (WLPO boundary NameCert obligations)**.** *For every accepted* $\mathsf{WLPOBoundary}^{\uparrow}$ *carrier* $W = (B, S, R, E, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite binary-window inspection, stream-window replay, regular-rational readback, Real-seal handoff, and refusal nonescape. These obligations are discharged through*

$$B \longmapsto S \longmapsto R \longmapsto E \longmapsto Q$$

together with H, C, P, N*.*

Proof. Project the carrier of Definition 4062.1. The projection exposes a finite binary window, a stream schedule, a regular rational readback, a Real seal, a refusal row, and the structural transport, replay, provenance, and naming rows.

The binary-window obligation reads only B and the displayed replay budget in S. It can certify which finite prefix was inspected, but it cannot decide whether every future binary entry is zero.

The Real-facing route then passes through R and E. These rows expose regular-rational readback and the terminal Real seal only after the finite window has already been named.

Finally the refusal row Q is transported by H and replayed by C with the same provenance P and local name N. Since the packet has no coordinate for a selected limit, quotient stream equality, trichotomy, countable choice, global WLPO, or ambient completeness, no NameCert obligation can export any of those principles. □

Theorem 4062.3 (WLPO boundary Real and Completion refusal)**.** *Every Phase Real or Completion consumer of an accepted* $\mathsf{WLPOBoundary}^{\uparrow}$ *packet receives only the displayed finite binary-window ledger* B*, stream schedule* S*, regular-rational readback* R*, Real seal* E*, refusal row* Q*, componentwise transport* H*, continuation replay* C*, provenance* P*, and local NameCert row* N*. In particular, finite binary-window evidence cannot export a selected limit, quotient stream equality, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with a Real or Completion request. By Definition 4062.1, the request reaches the packet only through the finite route $B, S, R, E, Q, H, C, P, N$.

Use Definition 4062.2. It places binary-window inspection before stream replay, regular-rational readback, Real-seal handoff, and refusal nonescape, so a consumer cannot bypass the finite window and ask for a completed all-zero decision.

The refusal row is the terminal boundary datum. Transport and replay preserve the same finite route, while provenance and local naming add no host-level coordinate. Therefore the accepted packet may witness an inspected prefix, but it cannot produce a selected limit, identify quotient streams, decide trichotomy, invoke countable choice, or complete $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{WLPOBoundary}^{\uparrow}$ Real or Completion read factors through finite binary-window evidence, stream replay, regular-rational readback, the $\mathsf{Real}^{\uparrow}$ seal, and the refusal row before the consumer may cite the boundary.

Independence witness: speckersequence, llpo-boundary, streamname, regseqrat, real: $\mathsf{WLPOBoundary}^{\uparrow}$ is not bijective to any listed sibling because it simultaneously carries a finite binary-window ledger,

stream schedule, regular-rational readback, Real seal, explicit WLPO refusal row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WLPOBoundary}^{\uparrow}$ packet is generated from finite binary-window evidence, StreamName replay, RegSeqRat readback, a RealUp seal, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4062.1, Definition 4062.2, and Definition 4062.3 seed the finite WLPO boundary for Phase Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WLPOBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim WLPO as a global principle, selected limits, quotient stream equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite binary-window inspection, stream replay, regular-rational readback, Real-seal handoff, and refusal nonescape for BEDC.Derived.WLPOBoundaryUp.

Chapter 4063

A Concrete Naming Certificate for $\mathsf{LocatedClosedBall}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedClosedBall}^{\uparrow}$ packet records a located closed ball by finite center, radius, distance, membership, and locator rows rather than by an ambient closed subset. It sits between the center-radius closed-ball surface of chapter 1451, the endpoint-location discipline of chapter 394, the located-distance route of chapter 1533, and the terminal real-name seal of chapter 38. The packet gives compact-metric, local-support, and completion consumers a displayed closed-ball support row without quotienting points, choosing nearest points, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4063.1 (Located closed-ball carrier)**.** *A* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, c, r, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LocatedMetric}^{\uparrow}$ *source row naming the metric carrier and located distance discipline.*
2. *c is the displayed centre row.*
3. *r is the nonnegative* $\mathsf{Real}^{\uparrow}$ *radius row.*
4. *D is the distance row from the source point to the centre, read through the located metric.*
5. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the non-strict comparison interval $D \leq r$.*
6. *R is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *membership row attached to the same centre, radius, and distance comparison.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the radius and comparison readback.*
8. *H records componentwise* hsame *transport of the metric source, centre, radius, distance, interval, closed-ball membership, and seal rows.*
9. *C records displayed* Cont *replay for distance evaluation, located interval comparison, closed-ball membership, and downstream support handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedClosedBall}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, c, r, D, I, R, E, H, C, P, N$. It has no coordinate for a host subset, quotient point equality, selected nearest point, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedClosedBallUp.TasteGate.LocatedClosedBallTasteGate single carrier alignment`

Theorem 4063.2 (Located closed-ball membership transport)**.** *Let $B = (M, c, r, D, I, R, E, H, C, P, N)$ and $B' = (M', c', r', D', I', R', E', H', C', P', N')$ be accepted* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carriers whose metric source, centre, radius, distance, interval, membership, and seal rows are paired by the displayed* hsame *transports. Then the located closed-ball membership row R transports to R' through the same located-metric distance row and located-interval comparison row, with* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming preserved.*

Proof. Project the carrier of Definition 4063.1. The membership row R is downstream of the located metric source M, centre c, radius r, distance row D, located interval row I, and terminal seal E.

Transport the displayed rows componentwise by H and H'. The centre and radius rows move together with the distance row, and the located interval comparison remains attached to that same transported distance and radius.

Replay the comparison by C and C'. Since P, P' and N, N' source and name only the displayed rows, the transported membership row is still the $\mathsf{MetricClosedBall}^{\uparrow}$ comparison read through $\mathsf{LocatedMetric}^{\uparrow}$ and $\mathsf{LocatedInterval}^{\uparrow}$, not a host subset or quotient point equality. □

Theorem 4063.3 (Located closed-ball nonescape)**.** *Every compact-metric, local-support, completion, or interval-facing consumer of an accepted* $\mathsf{LocatedClosedBall}^{\uparrow}$ *carrier must read the finite route*

$$M, c, r, D, I, R, E, H, C, P, N.$$

The consumer may use the located metric source, displayed centre, radius seal, located distance, located interval comparison, MetricClosedBall membership, transport, replay, provenance, and local naming rows, but it cannot add a host closed subset, quotient point equality, selected nearest point, trichotomy, selected subcover, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4063.1. The accepted packet has exactly the rows $M, c, r, D, I, R, E, H, C, P, N$.

Use Definition 4063.2. It keeps every membership read attached to the located metric source, located interval comparison, closed-ball membership row, and Real seal under transport and replay.

The listed consumers therefore receive a located support row only through the displayed finite route. Since none of the refused host data is a carrier coordinate or a continuation target, the read cannot escape to a host subset, quotient point equality, selected point, selected subcover, trichotomy, or ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{LocatedClosedBall}^{\uparrow}$ read displays one $\mathsf{LocatedMetric}^{\uparrow}$ source row, one centre row, one $\mathsf{Real}^{\uparrow}$ radius row, one located distance row, one $\mathsf{LocatedInterval}^{\uparrow}$ comparison row, one $\mathsf{MetricClosedBall}^{\uparrow}$ membership row, one terminal $\mathsf{Real}^{\uparrow}$ seal row, and the structural rows H, C, P, N.

Independence witness: metricclosedball, locatedinterval, locatedmetric, real: $\mathsf{LocatedClosedBall}^{\uparrow}$ is not bijective to any listed sibling because it binds the closed-ball membership row, located interval comparison, and located metric distance row through the same displayed centre-radius support before the Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}$.

Constructive story: A $\mathsf{LocatedClosedBall}^{\uparrow}$ packet is generated from a LocatedMetric source row, centre row, RealUp radius row, located distance row, LocatedInterval comparison row, MetricClosedBall membership row, RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4063.1, Definition 4063.2, and Definition 4063.3 seed the located closed-ball support route over MetricClosedBall, LocatedInterval, LocatedMetric, Metric, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedClosedBall}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host closed subsets, quotient point equality, selected nearest points, selected subcovers, trichotomy, compactness, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-distance exactness, located-interval comparison transport, MetricClosedBall membership transport, Real-seal nonescape, replay, provenance, and local naming for BEDC.Derived.LocatedClosedBallUp.

Chapter 4064

A Concrete Naming Certificate for $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ packet records the theorem-level route from displayed uniform Cauchy data to the regular completion handoff. It consumes the uniform-space carrier of chapter 339, the Cauchy-filter tail surface of chapter 333, the finite-window naming route of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the universal completion row of chapter 1425. The packet names a finite theorem carrier for uniform completion consumers without quotient completion objects, selected representatives, countable choice, host real equality, or ambient completeness.

Definition 4064.1 (Bishop uniform completion theorem carrier)**.** *A* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, F, S, R, E, V, H, C, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{UniformSpace}^{\uparrow}$ *carrier row, recording the displayed source points, entourage requests, refinement reads, and separatedness boundary available to the theorem consumer.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *tail-data row, storing the finite filter-base and tail-stability evidence read inside the same uniform carrier.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, presenting the inspected Cauchy-filter observations as bounded observation windows rather than as an infinite chosen representative.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected windows into regular rational names through the displayed dyadic tolerance route.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording the terminal real-name observations produced from the regular rational readback.*
6. *V is the* $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$ *handoff row, naming the extension-and-uniqueness surface available to completion consumers after the source, tail, window, readback, and seal rows have all been read.*
7. *H records componentwise* hsame *transport for the uniform carrier, Cauchy-filter tail data, finite windows, regular rational readback, Real seal, universal handoff, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay through*

$$X, F, S, R, E, V \longmapsto \mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}.$$

9. *P is the* Pkg *provenance row over* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUniversal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, selected completion representatives, a countable-choice schedule, host real equality, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.BishopUniformCompletionTheoremUp.BishopUniformCompletionTheoremTasteGate`
`single carrier alignment`

Theorem 4064.2 (Bishop uniform completion theorem NameCert obligations)**.** *For an accepted* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier*

$$U = (X, F, S, R, E, V, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-space admission for X, Cauchy-filter tail stability for F, finite-window inspection for S, regular rational readback for R, terminal Real sealing for E, regular-completion universal handoff for V, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. No obligation may be discharged by quotient completion, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Carrier admission is projection from Definition 4064.1. The accepted packet displays the uniform carrier, the Cauchy-filter tail data, the finite windows, the regular rational readback, the Real seal, and the universal handoff.

The theorem route begins with X and F. The uniform row fixes the entourage discipline, while the Cauchy-filter row supplies only finite tail evidence inside that discipline. There is no independent coordinate for a quotient completion object or a selected filter representative.

The observation route then passes through S, R, and E. Finite windows are read as regular rational names and then sealed as displayed Real rows, so the certificate never replaces the readback with host real equality, countable choice, or ambient completeness.

Finally V is the only completion-facing exit. The rows H, C, P, N transport, replay, source, and name the same finite route, so every local obligation is accounted for by the displayed rows and no hidden completion principle can enter the packet. □

Theorem 4064.3 (Bishop uniform completion theorem uniform handoff)**.** *Every theorem consumer of an accepted* $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RegularCauchyCompletionUniver}$$

The consumer first reads the uniform carrier, then the Cauchy-filter tail row, then the finite-window observations, then the regular rational readback, then the Real seal, and only then the regular-completion universal handoff.

Lean4 ✓ `BEDC.Derived.BishopUniformCompletionTheoremUp.BishopUniformCompletionTheoremUniformHandoff`

Proof. Begin with the accepted carrier U. The only uniform data available to a theorem consumer is X, and the only Cauchy data is the tail row F read inside that same uniform carrier.

The consumer must next inspect the finite windows S and their regular rational readback R. These rows supply the named observations from which the Real seal E is produced; no selected representative or countable-choice schedule is present in the carrier.

The completion-facing handoff V can be read only after X, F, S, R, E have been opened. Since the remaining rows merely transport, replay, source, and locally name this route, every theorem consumer factors through the stated chain and cannot bypass it by invoking quotient completion, host real equality, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ theorem read factors through one UniformSpace carrier row, one CauchyFilter tail row, one StreamName finite-window row, one RegSeqRat readback row, one Real seal row, one RegularCauchyCompletionUniversal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniformspace, cauchyfilter, streamname, regseqrat, real, regular-cauchy-completion-universal: $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source uniform structure, Cauchy-filter tail evidence, finite windows, regular rational readback, Real seal, universal completion handoff, and the transport/replay/provenance rows binding them into a theorem-level completion certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4064.1, Definition 4064.2, and Definition 4064.3 seed the uniform completion theorem route over a uniform carrier, Cauchy-filter tail row, finite windows, regular rational readback, Real seal, universal handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformCompletionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-space admission, Cauchy-filter tail stability, finite-window inspection, regular rational readback, Real sealing, universal completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformCompletionTheoremUp.

Constructive story: A $\mathsf{BishopUniformCompletionTheorem}^{\uparrow}$ packet is generated from one UniformSpace carrier row, one CauchyFilter tail-data row, one StreamName finite-window row, one RegSeqRat readback row, one Real seal row, one RegularCauchyCompletionUniversal handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4065

A Concrete Naming Certificate for $\mathsf{CantorPairing}^{\uparrow}$

Following Cantor's pairing function, the $\mathsf{CantorPairing}^{\uparrow}$ packet records a finite two-input natural-number indexing surface for schedules that must pass between paired observations and a single displayed countable index. It sits after the natural-number carrier of chapter 28, reads the finite product envelope of chapter 34, and supplies a countable schedule handoff to the finite stream-window discipline of chapter 126. The chapter names only a finite pairing and readback certificate; it does not import host natural-number arithmetic as an unrecorded oracle, quotient equality of pairs, countable choice, or a completed stream.

Definition 4065.1 (Cantor pairing carrier)**.** *A* $\mathsf{CantorPairing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, Y, J, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Nat}^{\uparrow}$ *input rows. They are read as finite unary natural-number names, not as ambient host numerals.*

2. *J is the pairing-code row. It records the displayed triangular code*

$$\pi(X, Y) = \frac{(X + Y)(X + Y + 1)}{2} + Y.$$

   The additions, multiplication, division by the displayed even triangular product, and final successor budget are all ledger rows over the two $\mathsf{Nat}^{\uparrow}$ *inputs.*

3. *L and R are projection rows. They read back the left and right coordinates from the displayed code by replaying the finite triangular search ledger and returning the unique pair of* $\mathsf{Nat}^{\uparrow}$ *rows whose recorded code is J.*

4. *H records componentwise* hsame *transport for the two input rows, the pairing-code row, the two projection rows, continuation replay, provenance, and the local naming row.*

5. *C records displayed* Cont *replay from the two* $\mathsf{Nat}^{\uparrow}$ *inputs through the* $\mathsf{Prod}^{\uparrow}$ *pair envelope, the pairing-code row, the projection readbacks, and the* $\mathsf{StreamName}^{\uparrow}$ *schedule handoff.*

6. *P is the* Pkg *provenance over* $\mathsf{CantorPairing}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

7. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$ *row.*

The carrier has no coordinate for host natural-number pairing, quotient equality of pairs, a selected enumeration, countable choice, or a completed infinite schedule.

Lean4 ✓ `BEDC.Derived.CantorPairingUp.CantorPairingTasteGate single carrier alignment`

Theorem 4065.2 (Cantor pairing NameCert obligations)**.** *For every accepted* $\mathsf{CantorPairing}^\uparrow$ *carrier* $K = (X, Y, J, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two* $\mathsf{Nat}^\uparrow$ *inputs, pairing-code row, projection rows, transport row, continuation row, provenance row, and local naming row;*
2. *pairing totality, requiring the displayed triangular computation to produce one finite* $\mathsf{Nat}^\uparrow$ *code row from the two finite input rows;*
3. *projection readback, requiring L and R to recover precisely the two displayed input rows when the same triangular ledger is replayed from J;*
4. *StreamName schedule handoff, requiring every downstream countable schedule consumer to read the paired observation through J, L, R before it treats the result as a single* $\mathsf{StreamName}^\uparrow$ *index;*
5. *ledger refusal, excluding unrecorded host natural-number arithmetic, quotient equality of pairs, selected enumerations, countable choice, and completed streams.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4065.1. The projection exposes exactly two $\mathsf{Nat}^\uparrow$ input rows, one displayed pairing-code row, two projection rows, and the structural rows H, C, P, N.

Pairing totality is a finite replay through C. The two inputs first enter the $\mathsf{Prod}^\uparrow$ envelope, then the triangular ledger forms $X + Y$, the next successor, the displayed even product, the halved triangular number, and the final offset by Y. Every intermediate value is a $\mathsf{Nat}^\uparrow$ ledger row, so the code row J is admitted without reading an ambient host pairing function.

Projection readback uses the same finite ledger in the reverse inspection direction. The projection rows L and R replay the bounded triangular search named by J and return the unique coordinate rows whose recorded code is J. Componentwise transport by H preserves the code and both recovered coordinates under classifier movement.

Finally, a $\mathsf{StreamName}^\uparrow$ consumer reaches the paired observation only through the J, L, R handoff. The provenance row P and local certificate row N name no further coordinate, so the packet cannot export quotient pair equality, a selected enumeration, countable choice, host arithmetic, or a completed stream schedule. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorPairing}^\uparrow}$.

Constructive story: A $\mathsf{CantorPairing}^\uparrow$ packet is generated from two Nat input rows, a displayed triangular pairing-code row, two projection rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CantorPairing}^\uparrow}$, seed)

Scope closed: Definition 4065.1 and Definition 4065.2 seed the finite Cantor-pairing schedule interface between Nat inputs, Product envelopes, and StreamName countable indices.

Formal verification: FormalStatus(NameCert$_{\mathsf{CantorPairing}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host natural-number pairing, quotient equality of pairs, selected enumerations, countable-choice schedules, completed streams, or Lean verification.

Upgrade path: Obligation closure requires carrier admission, pairing totality, projection readback, StreamName handoff, and ledger refusal proofs for the same finite pairing packet.

Chapter 4066

A Concrete Naming Certificate for $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$

Following Kelley, Moore–Smith, and Bourbaki, the $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ packet records the separated completion route for directed Cauchy nets as finite BEDC NameCert data. It sits after the Cauchy-net carrier of chapter 589, the Moore–Smith completion interface of chapter 2190, the separated completion boundary of chapter 1612, and the metric-completion handoff of chapter 1506. Its Real-facing seal is read only through the L10 finite-name spine of chapter 126, chapter 301, and chapter 38.

The packet names a completion carrier whose comparison discipline is separated before any terminal Real-facing read is public. It exposes Cauchy-net windows, Moore–Smith boundary data, separated-completion comparison, metric-completion handoff, stream-name observations, regular rational tolerance rows, Real seals, transport, replay, package provenance, and local naming. It exports no quotient net, no selected limit, no host equality of completed points, no countable-choice schedule, and no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 4066.1 (Separated Cauchy-net completion carrier)**.** *A* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, M, Q, L, W, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyNet}^{\uparrow}$ *directed-window row, including the finite tail requests and common-upper witnesses consumed by the completion route.*
2. *M is the* $\mathsf{MooreSmithCompletion}^{\uparrow}$ *boundary row that turns the same directed Cauchy-net data into a completion-facing Moore–Smith read.*
3. *Q is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, forcing equality-facing reads through separated zero-distance evidence rather than host equality or quotient-net equality.*
4. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, recording the metric-completion consumer surface reached by the displayed net route.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row attached to the completion read before regular rational tolerances are exported.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, recording the finite regular rational budget used by numerical consumers.*

7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after C, M, Q, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for Cauchy-net windows, Moore–Smith boundary data, separated comparison, metric-completion handoff, finite stream observations, regular rational tolerances, Real seals, replay data, provenance, and local naming.*
9. *K records displayed* Cont *replay from Cauchy-net evidence through Moore–Smith completion, separated completion, metric completion, stream-name observation, regular rational readback, and Real-facing sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{MooreSmithCompletion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient nets, selected limits, host equality of completion points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete spaces, or an external Real-completion oracle.*

Lean4 ✓ `BEDC.Derived.SeparatedCauchyNetCompletionUp.SeparatedCauchyNetCompletionTasteGate`
`single carrier alignment`

Theorem 4066.2 (Separated Cauchy-net completion NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier $S = (C, M, Q, L, W, R, A, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetC}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Cauchy-net, Moore–Smith completion, separated completion, metric completion, stream-name, regular rational, Real seal, transport, replay, provenance, and local naming rows;*
2. *directed-net coverage, requiring every completion-facing read to begin with the* $\mathsf{CauchyNet}^{\uparrow}$ *row C;*
3. *Moore–Smith boundary exposure, requiring M before separated or metric completion consumers receive the packet;*
4. *separated comparison discipline, requiring Q to mediate all equality-facing reads before any Real-facing seal is exported;*
5. *metric and numerical handoff discipline, requiring L, W, R, A to be read in that order for metric-completion and Real-facing consumers;*
6. *structural preservation, requiring H, K, P, N to transport, replay, source, and name only the displayed route;*
7. *ledger non-escape, excluding quotient nets, selected limits, host equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete spaces, and external Real-completion oracles.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4066.1. The accepted packet exposes exactly the Cauchy-net, Moore–Smith, separated-completion, metric-completion, stream-name, regular rational, Real-seal, and structural rows.

Directed-net coverage and Moore–Smith boundary exposure fix the first part of the route. A completion-facing request cannot reach Q, L, W, R, A until the Cauchy-net row C and the Moore–Smith boundary row M have been displayed.

Separated comparison is isolated in Q. Thus equality-facing reads are zero-distance handoffs through the separated-completion surface, not host equality of completed points and not equality of quotient nets.

The metric and numerical handoff rows are downstream. A metric-completion consumer reads L, a finite observation consumer reads W, a tolerance consumer reads R, and the terminal Real-facing consumer reads A only after those prior rows have appeared.

Finally H, K, P, N preserve, replay, source, and name only this ordered route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4066.3 (Separated Cauchy-net completion zero-distance handoff)**.** *Let $S = (C, M, Q, L, W, R, A, H, K, P$ be an accepted* $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ *carrier. Every comparison between two completion-facing reads of S factors through the route*

$$C \longmapsto M \longmapsto Q \longmapsto L \longmapsto W \longmapsto R \longmapsto A \longmapsto H, K, P, N.$$

In particular, the handoff to $\mathsf{Real}^{\uparrow}$ *is permitted only after the separated zero-distance row Q, the* $\mathsf{MetricCompletion}^{\uparrow}$ *row L, the* $\mathsf{StreamName}^{\uparrow}$ *row W, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R have been displayed. The comparison exports no quotient net, selected limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Start with a comparison request for an accepted packet. The carrier definition gives the Cauchy-net row C as the only source of directed tail data, so the comparison begins with displayed finite net windows.

The Moore–Smith boundary row M turns the same net data into a completion-facing read. The separated-completion row Q then supplies the zero-distance comparison discipline before any metric or Real-facing consumer can inspect the result.

The route continues through L, W, R, A. These rows record the metric-completion handoff, finite stream observations, regular rational tolerances, and terminal Real seal in order.

Transport, replay, provenance, and local naming in H, K, P, N preserve that route componentwise. Because the carrier has no quotient-net, selected-limit, host-equality, choice, or ambient-completeness coordinate, no comparison can bypass the zero-distance handoff. □

Falsifiable prediction: A SeparatedCauchyNetCompletionUp consumer that reads a Real-facing seal before displaying the separated comparison row Q and the regular rational tolerance row R is not reading an accepted packet of this chapter.

Independence witness: cauchynet, mooresmithcompletion, separatedcompletion, metriccompletion, streamname, regseqrat, real: the separated Cauchy-net completion packet is not bijective to any one sibling because CauchyNet lacks the completion boundary, MooreSmithCompletion lacks the explicit separated Real-facing handoff, SeparatedCompletion lacks the directed-net source window, MetricCompletion lacks the StreamName and RegSeqRat seal order, StreamName lacks the separated comparison row, RegSeqRat lacks the Moore–Smith boundary, and Real lacks the upstream net-completion ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$ packet is generated from CauchyNetUp directed-window evidence, MooreSmithCompletionUp boundary data, SeparatedCompletionUp zero-distance comparison, MetricCompletionUp handoff, StreamNameUp observations, RegSeqRatUp tolerances, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4066.1, Definition 4066.2, and Definition 4066.3 seed the separated Cauchy-net completion route from directed net windows through zero-distance comparison and Real-facing sealing.

Formal verification: FormalStatus(NameCert$_{\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient nets, selected limits, host equality of completed points, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Moore–Smith boundary exposure, separated zero-distance comparison, metric handoff, stream-name observation, regular rational tolerance readback, Real-facing sealing, transport preservation, and non-escape for BEDC.Derived.SeparatedCauchyNetCompletionUp.

Chapter 4067

A Concrete Naming Certificate for $\mathsf{RoleSeparationLedger}^{\uparrow}$

The $\mathsf{RoleSeparationLedger}^{\uparrow}$ packet realizes the compiler, AI or machine-learning, and human-consciousness comparison of section 9414.59 as finite NameCert data. It sits beside the compiler preservation route of chapter 1097, the compiler replay route of chapter 1055, the prompt-output trace of chapter 965, the large-model inscription point of chapter 939, and the public observer interface of chapter 928. The packet binds three role rows without collapsing compiler preservation, model pattern generation, and self-proxy readback into one classifier.

Definition 4067.1 (RoleSeparationLedger carrier)**.** *A* $\mathsf{RoleSeparationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, A, G, S, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompilerClassifierRoute}^{\uparrow}$ *row, recording source-to-target classifier preservation as a compiler route rather than as a self-proxy or model-output row.*
2. *F is the* $\mathsf{CompilerTraceFaithfulness}^{\uparrow}$ *row, recording finite compiler replay and source-to-target* hsame *preservation.*
3. *A is the AI or machine-learning pattern row, sourced in* $\mathsf{PromptResponseTrace}^{\uparrow}$ *and* $\mathsf{LargeModelInscriptionPoint}^{\uparrow}$ *data for prompt, response, corpus, activation, and generated-signature reads.*
4. *G is the generated-output boundary row, naming the model-facing inscription and prompt-response rows that may be inspected by audit consumers.*
5. *S is the self-proxy readback row, recording the local observer-facing self-proxy boundary only through the licensed public rows of* $\mathsf{ObserverInterface}^{\uparrow}$*.*
6. *O is the role-separation ledger row. It records that K, F are preservation rows, A, G are pattern-generation and inscription rows, and S is a self-proxy readback row, with no permitted classifier that identifies the three roles.*
7. *H records componentwise* hsame *transport for the compiler, trace-faithfulness, prompt-response, inscription, observer-interface, role-ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through*

$$K, F, A, G, S, O \longmapsto \mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}.$$

9. *P is the* Pkg *provenance over* RoleSeparationLedger$^{\uparrow}$, CompilerClassifierRoute$^{\uparrow}$, CompilerTraceFaithfulness$^{\uparrow}$, PromptResponseTrace$^{\uparrow}$, LargeModelInscriptionPoint$^{\uparrow}$, ObserverInterface$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{RoleSeparationLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate that treats compiler self-hosting as consciousness, model generation as self-proxy binding, or public observer readback as an internally ratified model proof.

Lean4 ✓ `BEDC.Derived.RoleSeparationLedgerUp.RoleSeparationLedgerTasteGate single carrier alignment`

Theorem 4067.2 (RoleSeparationLedger NameCert obligations)**.** *For an accepted* RoleSeparationLedger$^{\uparrow}$ *carrier*

$$L = (K, F, A, G, S, O, H, C, P, N),$$

the local NameCert$_{\text{RoleSeparationLedger}^{\uparrow}}$ *surface consists exactly of compiler-route admission for K, compiler-trace faithfulness for F, AI pattern and prompt-response admission for A, large-model inscription admission for G, observer-interface self-proxy readback for S, role separation through O, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4067.1. It displays the compiler route, compiler replay, AI pattern row, inscription boundary, observer readback, role-separation ledger, and structural rows.

The compiler side is exhausted by K, F. These rows preserve and replay classifier structure, but they do not supply self-proxy readback or model-side generated-signature evidence.

The AI or machine-learning side is exhausted by A, G. These rows name prompt-response and inscription data, but they do not certify compiler preservation and do not bind a current self-center.

The observer-facing side is exhausted by S. The ledger row O records the separation of the three roles, while H, C, P, N transport, replay, source, and name that separation. Hence the local NameCert obligations are exactly the displayed rows. □

Theorem 4067.3 (RoleSeparationLedger role noncollapse)**.** *Every accepted* RoleSeparationLedger$^{\uparrow}$ *packet preserves the three role rows without collapse: compiler preservation remains on K, F, AI pattern generation and inscription remain on A, G, and self-proxy readback remains on S. A consumer may compare the rows only through the role-separation ledger O, not by identifying compiler self-hosting, model output generation, and human-consciousness readback.*

Proof. Begin with the role-separation ledger row O. It records the three role classes and their permitted sibling routes.

The preservation rows K, F are read before any model or observer row can be compared. They expose classifier preservation and trace faithfulness only.

The generation rows A, G are read as prompt-response and inscription data. They may be audited as outputs, but no row in the packet lets them become compiler correctness or self-proxy binding.

The self-proxy row S is read through the observer interface. Since O is the only comparison row and the carrier contains no collapse coordinate, the three role rows remain separated in every accepted read. □

Theorem 4067.4 (RoleSeparationLedger TasteGate target)**.** *The Lean-facing downstream surface for* RoleSeparationLedger$^{\uparrow}$ *is the packet family to be recorded under* Derived/RoleSeparationLedgerUp/TasteGate.lean *with explicit* Nontrivial RoleSeparationLedger$^{\uparrow}$ *and* FieldFaithful RoleSeparationLedger$^{\uparrow}$ *instances.*

The gate must recover the compiler preservation rows K, F, the AI pattern-generation and inscription rows A, G, the self-proxy readback row S, the role-separation ledger O, and the support rows H, C, P, N before admitting the packet.

Proof. Read the carrier first. The fields $K, F, A, G, S, O, H, C, P, N$ are distinct coordinates of the accepted packet.

Nontriviality is witnessed by the presence of at least the compiler, generation, and self-proxy rows together with the ledger row that distinguishes them. Field faithfulness is witnessed by the fact that each accepted read must recover the same rows in the same roles.

The future TasteGate file therefore has a finite obligation surface: it checks that the role rows are present, distinct, replayed, transported, sourced, and locally named before any public admission of $\mathsf{RoleSeparationLedger}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{RoleSeparationLedger}^{\uparrow}$ read preserves the compiler preservation row, AI pattern-generation row, and self-proxy readback row without collapse.

Independence witness: compilerclassifierroute, compilertracefaithfulness, prompt-response-trace, largemodelinscriptionpoint, observer-interface: none is bijective because the role-separation ledger stores the joint noncollapse row O binding compiler preservation, AI pattern generation, and self-proxy readback as distinct finite NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{RoleSeparationLedger}^{\uparrow}$ packet is generated from CompilerClassifierRoute and CompilerTraceFaithfulness preservation rows, PromptResponseTrace and LargeModelInscriptionPoint pattern-generation rows, an ObserverInterface self-proxy readback row, a role-separation ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4067.1, Definition 4067.2, Definition 4067.3, and Definition 4067.4 seed the compiler, AI pattern-generation, and self-proxy readback separation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoleSeparationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove compiler consciousness, model consciousness, phenomenal consciousness, hidden model truth, internally ratified universal closure, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compiler preservation, trace faithfulness, prompt-response admission, inscription admission, observer readback, role noncollapse, transport, replay, provenance, and local naming for BEDC.Derived.RoleSeparationLedgerUp.

Chapter 4068

A Concrete Naming Certificate for $\mathsf{PrecompactUniform}^{\uparrow}$

Following Bourbaki and Kelley on precompact uniform spaces, the $\mathsf{PrecompactUniform}^{\uparrow}$ packet records the finite total-bounded uniform route used by the L10 completion spine. It consumes the entourage carrier of chapter 339, the finite covering comparison of chapter 127, the Cauchy request surface of chapter 333, and the completion handoff of chapter 1473. The packet names a precompactness route without importing open-cover compactness, quotient completions, selected limits, countable choice, or ambient completeness.

Definition 4068.1 (PrecompactUniform carrier)**.** *A* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, E, F, T, Q, K, S, R, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, recording the source points, entourage requests, refinement rows, and separatedness boundary available to the packet.*
2. *E is the entourage ledger row. It lists the finitely inspected entourages and the refinement comparisons used before any covering row is read.*
3. *F is the finite cover row. For each displayed entourage request it stores the finite family of centers and the witness that the inspected source points are covered at that entourage.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *comparison row, binding the finite cover data to the sibling total-bounded carrier surface.*
5. *Q is the* $\mathsf{CauchyFilter}^{\uparrow}$ *request row, recording only those Cauchy requests whose entourage demands have already passed through E, F, T.*
6. *K is the* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff row for completion consumers that receive the precompact uniform packet after the finite cover and Cauchy request rows have been displayed.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the inspected request windows.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the finite entourage and readback comparisons.*

10. *H records componentwise* hsame *transport for the uniform source, entourage ledger, finite cover row, total-bounded comparison, Cauchy requests, completion handoff, finite windows, readbacks, dyadic tolerance rows, replay, provenance, and local naming.*

11. *C records displayed* Cont *replay through*

$$X, E, F, T, Q, K, S, R, D \longmapsto \mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}.$$

12. *P is the* Pkg *provenance over* $\mathsf{PrecompactUniform}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

13. *N is the local* $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, quotient completion equality, selected Cauchy representatives, countable choice, host equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.PrecompactUniformUp.PrecompactUniformTasteGate single carrier alignment`

Theorem 4068.2 (PrecompactUniform NameCert obligations)**.** *For an accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier*

$$U = (X, E, F, T, Q, K, S, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-space admission for X, entourage-ledger exactness for E, finite-cover exactness for F, total-bounded comparison through T, Cauchy-filter request admission through Q, uniform-completion handoff through K, finite-window and readback inspection through S, R, D, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4068.1. It displays the uniform source, entourage ledger, finite cover row, total-bounded comparison, Cauchy request row, completion handoff, finite windows, readbacks, dyadic tolerances, and structural rows.

The analytic route begins with X, E, F, T. A consumer first reads the uniform-space surface, then the inspected entourages, then the finite cover witnesses, and only then the total-bounded comparison row.

The completion-facing route is downstream of Q, K, S, R, D. Cauchy requests, completion handoffs, finite windows, regular readbacks, and dyadic tolerances are therefore bound to the same finite cover data.

Finally H, C, P, N transport, replay, source, and locally name the same packet. Since no excluded object is a coordinate of the carrier, the displayed rows exhaust the local NameCert obligations. □

Theorem 4068.3 (PrecompactUniform completion handoff)**.** *Every* $\mathsf{UniformCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *carrier first reads the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, the entourage ledger, the finite cover row, the* $\mathsf{TotallyBounded}^{\uparrow}$ *comparison row, and the* $\mathsf{CauchyFilter}^{\uparrow}$ *request row before it may inspect the completion handoff K. The handoff is a finite L10 completion route through the displayed rows, not an open-cover compactness theorem or a quotient completion object.*

Proof. Start with the NameCert obligations of Definition 4068.2. They place the completion handoff after the uniform source, entourage, finite-cover, total-bounded, and Cauchy request rows.

The finite windows S, readbacks R, and dyadic tolerances D are read on the same route. They provide the observed request data needed by completion consumers without selecting a limit or quotienting Cauchy names.

Transport and replay through H and C preserve the row order. Thus every completion consumer reaches K only through the displayed precompact uniform packet and cannot bypass it by invoking compactness, choice, or ambient completeness. □

Theorem 4068.4 (PrecompactUniform nonescape)**.** *An accepted* $\mathsf{PrecompactUniform}^{\uparrow}$ *packet exports only the displayed rows*

$$X, E, F, T, Q, K, S, R, D, H, C, P, N.$$

No downstream completion, compactness, or Cauchy-filter consumer can obtain open-cover compactness, quotient completion equality, selected representatives, countable-choice schedules, host equality, or ambient completeness from the packet.

Proof. Use Definition 4068.1 to list the available coordinates. The only analytic rows are the uniform source, entourage ledger, finite cover data, total-bounded comparison, Cauchy request, completion handoff, finite windows, readbacks, and dyadic tolerance rows.

Use Definition 4068.3. Completion-facing reads are confined to those rows before the handoff is exposed.

The structural rows H, C, P, N only transport, replay, source, and name the finite route. Because the excluded compactness and completion principles are absent from the carrier, no consumer can project them from an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$.

Constructive story: A $\mathsf{PrecompactUniform}^{\uparrow}$ packet is generated from a UniformSpace source row, entourage ledger rows, finite cover rows, a TotallyBounded comparison row, CauchyFilter request rows, a UniformCompletion handoff, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$, seed)

Scope closed: Definition 4068.1, Definition 4068.2, Definition 4068.3, and Definition 4068.4 seed the finite precompact uniform route for L10 completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PrecompactUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient completion, selected representatives, countable choice, host equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage exactness, finite-cover exactness, total-bounded comparison, Cauchy-filter request admission, completion handoff, readback, tolerance, transport, replay, provenance, and local naming for BEDC.Derived.PrecompactUniformUp.

Chapter 4069

A Concrete Naming Certificate for $\mathsf{RegularCauchyOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyOrder}^{\uparrow}$ packet records the constructive order surface for regular Cauchy real names as finite BEDC NameCert data. It sits over the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the algebraic order boundary of chapter 766, and the terminal real-name seal of chapter 38. The packet gives RealAlgOrder and Real consumers a displayed comparison route on regular names, without trichotomy, quotient stream equality, selected limits, host real equality, or countable choice.

Definition 4069.1 (Regular-Cauchy order carrier)**.** *A* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (X, Y, S, D, R, L, A, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy source rows whose finite rational windows are compared.*
2. *S is the shared* $\mathsf{StreamName}^{\uparrow}$ *schedule naming every inspected precision window for both sources.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, slack, sign budget, and comparison threshold for each inspected window.*
4. *R is the regular-rational readback row that records the finite rational observations obtained from X and Y through S and D.*
5. *L is the order-comparison ledger; it stores only located finite evidence that the displayed readbacks meet, fall below, or remain unresolved at the current dyadic tolerance.*
6. *A is the downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff row, available only after X, Y, S, D, R, L have been displayed.*
7. *H records componentwise* hsame *transport for the regular sources, stream schedule, dyadic ledger, readback row, order ledger, RealAlgOrder handoff, and local certificate rows.*
8. *C records the displayed* Cont *replay from an order request to the stream windows, dyadic tolerance comparison, regular-rational readback, order ledger, and RealAlgOrder handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyOrder}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, Y, S, D, R, L, A, H, C, P, N$. It has no coordinate for trichotomy, quotient equality of streams, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyOrderUp.RegularCauchyOrderTasteGate single carrier alignment`

Theorem 4069.2 (Regular-Cauchy order NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *carrier $O = (X, Y, S, D, R, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, shared* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparisons, regular-rational readback, order-comparison ledger, downstream* $\mathsf{RealAlgOrder}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by trichotomy, quotient stream equality, selected limits, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4069.1. The rows $X, Y, S, D, R, L, A, H, C, P, N$ are the only displayed coordinates available to the local certificate.

The two regular Cauchy sources are compared through the same stream schedule S and dyadic tolerance ledger D. The readback row R and order ledger L therefore record finite tolerance evidence, not equality of completed streams or a global trichotomy.

The RealAlgOrder handoff A is downstream of that finite route. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and the carrier has no additional coordinate from which a selector, host equality, choice principle, or ambient completeness theorem could be projected. □

Theorem 4069.3 (Regular-Cauchy order RealAlgOrder handoff)**.** *Every* $\mathsf{RealAlgOrder}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyOrder}^{\uparrow}$ *packet reads the constructive order surface only through the displayed route*

$$X, Y, S, D, R, L, A, H, C, P, N.$$

The handoff exports finite stream windows, dyadic tolerance comparisons, regular-rational readback, an order-comparison ledger, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It does not export trichotomy, quotient stream equality, selected limits, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer of the order surface. The carrier displays X and Y only together with the shared stream schedule S, dyadic tolerance ledger D, and finite readback row R.

The order ledger L can be read only after those windows and tolerances have been displayed. If an algebraic-order or Real-facing consumer is present, the downstream row A is available only after the same finite comparison route has been consumed.

Apply the NameCert obligations of Definition 4069.2. The support rows H, C, P, N preserve, replay, source, and name this finite route, so the handoff exports no trichotomy, quotient stream equality, selected limit, host real equality, choice principle, or ambient completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyOrder}^{\uparrow}$ consumer that reaches a $\mathsf{RealAlgOrder}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ order read without first reading two RegSeqRat source rows, a shared StreamName schedule, a DyadicRatCore tolerance ledger, regular-rational readback, an order-comparison ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: regseqrat, dyadicratcore, streamname, realalgorder, real: $\mathsf{RegularCauchyOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular Cauchy source rows to one constructive order ledger and downstream RealAlgOrder handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOrder}^{\uparrow}$ packet is generated from two RegSeqRat source rows, a shared StreamName schedule, a DyadicRatCore tolerance ledger, regular-rational readback, an order-comparison ledger, a RealAlgOrder handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4069.1, Definition 4069.2, and Definition 4069.3 seed the finite constructive order route for regular Cauchy real names through RegSeqRat, DyadicRatCore, StreamName, RealAlgOrder, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient stream equality, selected limits, host real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and RealAlgOrder handoff theorem.

Chapter 4070

A Concrete Naming Certificate for $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the computable-analysis convention that real-facing objects are read through effective names, the $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ packet records the finite interface where a Cauchy representation also carries located Cauchy evidence before the terminal real seal is consumed. It sits after the Cauchy-representation boundary of chapter 1680, the sequence-level located Cauchy packet of chapter 1595, the located real boundary of **??**, and the represented-space source surface of chapter 1494. The packet gives L10 Real and Completion consumers a single BEDC-native route through displayed finite rows; it does not import quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4070.1 (LocatedCauchyRepresentation carrier)**.** *A* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, Q, J, S, M, D, X, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoder row, recording the finite representation route by which stream observations are decoded.*
2. *Q is the* $\mathsf{LocatedCauchySequence}^{\uparrow}$ *evidence row, recording the located tail witness attached to the decoded observations.*
3. *J is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison-window row, exposing the finite interval and locatedness comparisons that the downstream real-facing consumer may inspect.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by the decoder and by the located Cauchy evidence.*
5. *M is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for the regular readback, located tail, and located real comparison windows.*
7. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *exposure row, presenting the same finite name to represented-space consumers without adding a second representation relation.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after R, Q, J, S, M, D, X have been displayed.*

9. *H records componentwise* hsame *transport for the decoder, located evidence, comparison windows, stream windows, readback, tolerances, represented-space exposure, real seal, continuation, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from representation decoding through located Cauchy evidence, located real comparison, represented-space exposure, and the terminal seal handoff.*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{LocatedCauchySequence}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite located Cauchy representation packet. It has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationTasteGate single carrier alignment`

Theorem 4070.2 (LocatedCauchyRepresentation NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier* $L = (R, Q, J, S, M, D, X, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the decoder, located Cauchy evidence, located real comparison windows, stream windows, regular rational readback, dyadic tolerance ledger, represented-space exposure, terminal real seal, transport, replay, provenance, and local naming rows;*

2. *decoder exactness, requiring every representation read to pass through the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row R before any located evidence is inspected;*

3. *located sequence exactness, requiring the row Q to expose only the located Cauchy tail witnesses attached to the displayed decoder and stream-window route;*

4. *comparison-window exactness, requiring the* $\mathsf{LocatedReal}^{\uparrow}$ *row J to read its finite location windows through the same regular rational and dyadic rows M, D;*

5. *represented-space compatibility, requiring X to expose the same finite name and target route rather than a separate representation relation;*

6. *real-seal ordering, requiring E to remain downstream of R, Q, J, S, M, D, X and never to serve as a selected host limit;*

7. *componentwise transport, continuation replay, package provenance, and local naming for exactly the displayed rows H, C, P, N;*

8. *refusal of quotient streams, selected limits, countable choice, trichotomy, host real equality, and ambient completeness as local certificate data.*

No located Cauchy representation obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationNamecertObligations`

Proof. Project the accepted carrier. The visible analytic route is the representation decoder R, located Cauchy evidence Q, located real comparison windows J, stream windows S, regular rational readback M, dyadic tolerance ledger D, represented-space exposure X, and terminal seal E.

Decoder exactness comes first. A representation-facing consumer reaches located evidence only after reading the $\mathsf{CauchyRepresentation}^{\uparrow}$ row, so the packet cannot be reread as an unbounded stream or a selected limit.

The located rows are tied to the same finite route. The row Q supplies located tail evidence over the displayed stream windows, while J supplies located real comparison windows over the same regular rational readback and dyadic tolerance ledger.

The represented-space and seal rows add no independent analytic coordinate. The row X exposes the same name to represented-space consumers, and E is downstream of the decoder, located evidence, comparison windows, readback, and tolerance rows.

Finally, H, C, P, N transport, replay, source, and name the displayed packet without adding data. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4070.3 (LocatedCauchyRepresentation Real-seal handoff)**.** *Let $L = (R, Q, J, S, M, D, X, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{RepresentedSpace}^{\uparrow}$*-facing read of L factors through the finite handoff route*

$$R, Q, J, S, M, D, X \longmapsto E \longmapsto H, C, P, N.$$

Thus the terminal real seal and the represented-space exposure are reached only through the Cauchy-representation decoder, located Cauchy evidence, located real comparison windows, finite stream windows, regular rational readback, and dyadic tolerance ledger. The route contains no quotient stream, selected host limit, countable-choice selector, trichotomy branch, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCauchyRepresentationUp.LocatedCauchyRepresentationRealSealHandoff`

Proof. Begin with a real-facing or represented-space-facing read. Carrier admission exposes the target rows X and E only as part of the same finite packet as the decoder, located evidence, comparison windows, stream windows, readback, and tolerances.

Replay the continuation row C. The read first consumes R, then the located evidence Q and comparison windows J, and then the common stream-window, regular-readback, and dyadic-tolerance route S, M, D.

Only after that replay can X and E be inspected. The represented-space row offers the same finite name to its consumer, while the real-seal row is the terminal seal of that route rather than a host-selected limit.

The support rows H, C, P, N preserve and name the same route. Since the packet contains none of the excluded coordinates, no consumer can bypass the displayed handoff to read quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyRepresentation}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, LocatedCauchySequence evidence, LocatedReal comparison windows, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerances, RepresentedSpace exposure, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchyRepresentation}^{\uparrow}}$, seed)

Scope closed: Definition 4070.1, Definition 4070.2, and Definition 4070.3 seed the finite located Cauchy representation route from CauchyRepresentation and LocatedCauchySequence data to LocatedReal comparison windows, RepresentedSpace exposure, and a terminal Real seal.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyRepresentation}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, decoder exactness, located sequence exactness, comparison-window exactness, represented-space compatibility, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyRepresentationUp.

Chapter 4071

A Concrete Naming Certificate for $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ packet records the finite route by which a Bishop-style regular Cauchy real name exposes an explicit modulus before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the name. It sits over the finite stream windows of chapter 126, the rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the located regular interface of chapter 1454, and the extraction discipline of chapter 641. The packet gives the L10 Real and Completion spine a traditional constructive-analysis modulus row without quotient streams, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4071.1 (Bishop Cauchy modulus-extraction carrier)**.** *A* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, R, D, E, M, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule whose inspected finite windows are available to the Cauchy read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed rational observations.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the dyadic precision request, comparison cells, and half-radius splits.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after the stream, regular-rational, and dyadic rows have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row extracted from S, R, D, E through the same finite observation route.*
6. *W is the local* $\mathsf{CauchyModulusExtraction}^{\uparrow}$ *witness row tying the extracted threshold to the inspected windows and regular readback.*
7. *H stores componentwise* hsame *transport for the stream schedule, regular readback, dyadic ledger, Real seal, extracted modulus, extraction witness, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay of the route*

$$S, R, D \longmapsto W \longmapsto M \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusExtraction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected completed limit, trichotomy, countable choice, host real equality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyModulusExtractionUp.BishopCauchyModulusExtractionTasteGate`
`single carrier alignment`

Theorem 4071.2 (Bishop Cauchy modulus-extraction NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *carrier*

$$Q = (S, R, D, E, M, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed stream schedule, regular readback, dyadic tolerance ledger, Real seal, extracted modulus, extraction witness, transport, replay, provenance, and local naming rows;*
2. *window-readback exactness, requiring every modulus request to read the same* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rational observations named by S and R;*
3. *dyadic tolerance compatibility, requiring D to supply the precision and comparison cells used by the extracted threshold;*
4. *extraction locality, requiring W and M to be read only after S, R, D have been displayed;*
5. *seal non-escape, requiring E to remain downstream from the extracted modulus route and to export no quotient stream, selected limit, trichotomy, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier in Definition 4071.1. The accepted packet exposes exactly the stream, regular-rational, dyadic, Real-seal, modulus, extraction-witness, transport, replay, provenance, and naming rows listed there.

The window-readback and dyadic clauses are forced by the displayed source rows. A precision request can inspect only the windows named by S, the rational observations named by R, and the tolerance data named by D.

The extraction rows are downstream from that finite source route. The witness row W binds the observed windows and regular readback to the extracted threshold, while M records the resulting $\mathsf{CauchyModulus}^{\uparrow}$ handoff.

Finally read the terminal seal and structural rows. Since E, H, C, P, N preserve only the displayed route and the carrier has no coordinate for the excluded quotient, choice, order, equality, or completeness data, the five listed obligations exhaust the local certificate surface. □

Theorem 4071.3 (Bishop Cauchy modulus-extraction regular handoff)**.** *Every accepted* BishopCauchyModulusExtr *carrier hands its extracted modulus to later regular-Cauchy and* $\mathsf{Real}^{\uparrow}$ *completion consumers through the finite route*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulusExtraction}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

together with the displayed BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. Thus the handoff exports a finite threshold surface for the same Bishop-style regular Cauchy name, not a quotient stream, selected limit, trichotomy, countable-choice functional, host real equality, or ambient completeness principle.*

Proof. Begin with an accepted carrier $Q = (S, R, D, E, M, W, H, C, P, N)$. The source side of the packet is the displayed route through S, R, D, so a completion consumer first reads finite windows, rational readback, and dyadic tolerance data.

Next read the extraction witness. The row W is local to the same route and supplies the named observation surface used to produce the modulus threshold.

The extracted modulus row M is then the only handoff row available to downstream regular-Cauchy consumers. It is transported and replayed through H and C, while P and N source and name the same finite packet.

The terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream from that chain. Because no other carrier coordinate stores quotient equality, a selected limit, trichotomy, countable choice, host real equality, or ambient completeness, every regular handoff factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ packet is generated from finite StreamName windows, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a RealUp seal row, a CauchyModulus row, a CauchyModulusExtraction witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4071.1, Definition 4071.2, and Definition 4071.3 seed the finite Bishop-style modulus-extraction route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyModulusExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, trichotomy, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, dyadic tolerance compatibility, extraction locality, regular handoff stability, and RealUp seal non-escape for BEDC.Derived.BishopCauchyModulusExtractionUp.

Chapter 4072

A Concrete Naming Certificate for $\mathsf{HausdorffUniformity}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffUniformity}^{\uparrow}$ packet records the separated uniformity interface used by completion and uniqueness consumers before any quotient completion or representative choice is available. It sits between the entourage surface of chapter 339, the zero-distance classifier discipline of chapter 1411, the metric completion route of chapter 1506, and the Cauchy-filter completion boundary of chapter 1569. The packet names the finite Hausdorff-uniformity boundary as BEDC data; it does not identify host points, select completion representatives, or import a quotient completion.

Definition 4072.1 (Hausdorff uniformity carrier)**.** *A* $\mathsf{HausdorffUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (U, E, D, Z, T, C, P, N)$$

with the following displayed rows:

1. *U is the underlying* $\mathsf{UniformSpace}^{\uparrow}$ *row, including the finite entourage basis, diagonal-membership ledger, and refinement surface read by downstream consumers.*
2. *E is the uniform entourage row, recording the displayed entourages through which all separation and completion-facing comparisons must pass.*
3. *D is the diagonal separation row, witnessing that indistinguishability by the displayed entourages reaches the local classifier rather than host equality.*
4. *Z is the zero-distance handoff row, routing metric-generated separated reads through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier before a completion consumer inspects uniqueness.*
5. *T records componentwise* hsame *transport for U, E, D, Z and for the local continuation, provenance, and naming rows.*
6. *C records displayed* Cont *replay of the route*

$$U, E \longmapsto D \longmapsto Z.$$

7. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformity}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares observations only through U, E, D, Z, T, C, P, N. It has no coordinate for quotient uniform completions, host equality of points, selected representatives, countable choice, ambient completeness, or bridge data.

Lean4 ✓ `BEDC.Derived.HausdorffUniformityUp.HausdorffUniformityTasteGate` single carrier alignment

Theorem 4072.2 (Hausdorff uniformity NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffUniformity}^{\uparrow}$ *carrier*

$$H = (U, E, D, Z, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the uniform-space row, entourage row, diagonal separation row, zero-distance handoff row, transport, replay, provenance, and local naming rows;*
2. *entourage stability, requiring every separated comparison to begin with the displayed entourage basis and refinement data of U, E;*
3. *diagonal separation, requiring all indistinguishability-to-classifier reads to pass through D rather than through host equality;*
4. *completion-facing zero-distance handoff, requiring metric or filter completion consumers to read Z before using uniqueness data;*
5. *componentwise transport and displayed replay through T, C;*
6. *ledger non-escape, excluding quotient completions, host point equality, selected representatives, countable choice, ambient completeness, Lean verification, and bridge checking.*

No local NameCert obligation is stored outside these six rows.

Proof. Carrier admission is projection from Definition 4072.1. The accepted tuple displays exactly the uniform-space row, the entourage row, the diagonal separation row, the zero-distance handoff, and the structural rows.

Entourage stability follows from the order of the carrier. The rows U, E supply the displayed entourage basis and refinement data before diagonal separation can be read, so a consumer cannot replace finite uniform evidence by host equality of points.

The diagonal separation row D is the only exported route from uniform indistinguishability to the local classifier. The zero-distance handoff row Z then records the metric-generated or completion-facing comparison surface without adding a quotient completion or a representative selector.

Finally inspect T, C, P, N. Transport is componentwise, replay follows the displayed route, and provenance plus local naming mention only the rows listed in the carrier. Hence the packet exports no quotient completion, host equality, selected representative, countable-choice schedule, ambient completeness theorem, Lean verification, or bridge checking. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffUniformity}^{\uparrow}$ packet is generated from a UniformSpace row, a displayed entourage row, a diagonal separation row, a zero-distance handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4072.1 and Definition 4072.2 seed the finite Hausdorff-uniformity interface over uniform-space, entourage, diagonal separation, zero-distance handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, host equality of points, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage stability, diagonal separation, completion-facing zero-distance handoff, componentwise transport, replay, provenance, and ledger non-escape for BEDC.Derived.HausdorffUnif

Chapter 4073

A Concrete Naming Certificate for $\mathsf{CauchyCutComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyCutComparison}^{\uparrow}$ packet names the finite comparison surface between the located-cut presentation of chapter 2312 and the regular-Cauchy equality surface of chapter 1820. It reads the finite windows of chapter 126, the regular rational source of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal boundary of chapter 38. The packet compares presentations by displayed rows only; it does not identify quotient representatives, select limits, invoke countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4073.1 (Cauchy cut comparison carrier)**.** *A* $\mathsf{CauchyCutComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, Q, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the cut-side exposure row. It records the accepted* $\mathsf{CauchyCut}^{\uparrow}$ *lower–upper located-cut route before any equality-facing comparison is read.*
2. *Q is the setoid-side exposure row. It records the accepted* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *regular-Cauchy comparison surface.*
3. *S is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window ledger shared by the cut-to-stream route and the setoid equality route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, displaying the regular rational observations consumed by both sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records bracket, midpoint, and comparison tolerances at the same requested precision.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after K, Q, S, R, D have been replayed.*
7. *H records componentwise* hsame *transport for the cut-side row, setoid-side row, synchronized windows, readback, dyadic tolerance, seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a comparison request through the cut side, the setoid side, the synchronized window ledger, regular readback, dyadic tolerance, and then the seal row.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyCutComparison}^{\uparrow}$, $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $K, Q, S, R, D, A, H, C, P, N$. It has no coordinate for quotient equality, selected representatives, host real equality, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonCarrier`

Theorem 4073.2 (Cauchy cut comparison NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCutComparison}^{\uparrow}$ *carrier $J = (K, Q, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed cut-side exposure, setoid-side exposure, synchronized window ledger, regular readback, dyadic tolerance ledger, Real seal boundary, transport, continuation, provenance, and local naming row;*
2. *cut-side exposure, requiring the located lower–upper cut rows of* $\mathsf{CauchyCut}^{\uparrow}$ *to be visible before any comparison consumes the regular-Cauchy equality surface;*
3. *setoid-side exposure, requiring the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *windows, readbacks, tolerance tests, and equality-facing classifier row to remain displayed;*
4. *synchronized tolerance, requiring S, R, D to provide the same finite precision request to the cut-to-stream route and the regular-Cauchy setoid route;*
5. *Real seal discipline, requiring A to remain downstream of K, Q, S, R, D;*
6. *ledger refusal, excluding quotient equality, selected representatives, host real equality, countable choice, and ambient Real completeness.*

No NameCert obligation is stored outside those rows.

Lean4 ✓ `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonCarrier namecert obligations`

Proof. Project the carrier of Definition 4073.1. The projection exposes the cut-side row, setoid-side row, synchronized windows, regular readback, dyadic tolerance, Real seal, and the structural rows H, C, P, N.

The cut-side and setoid-side obligations are forced by the source rows K and Q. The row K displays the located-cut-to-stream route, while Q displays the regular-Cauchy equality surface; neither row contains a quotient or selected representative coordinate.

The synchronized tolerance obligation is the shared route through S, R, D. Replay by C sends a comparison request through the same finite windows, regular rational observations, and dyadic tolerance entries before the seal row A is available.

Transport by H preserves each displayed component, and P, N source and name only this finite packet. Therefore the local certificate is exhausted by the listed obligations. □

Theorem 4073.3 (Cauchy cut comparison route)**.** *Let $J = (K, Q, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{CauchyCutComparison}^{\uparrow}$ *packet. Every comparison read between the located-cut presentation and the regular-Cauchy setoid presentation factors through the finite route*

$$K, Q \longmapsto S \longmapsto R \longmapsto D \longmapsto A \longmapsto H, C, P, N.$$

Thus the located-cut side and the regular-Cauchy setoid side are compared only after their StreamName windows, RegSeqRat readback rows, DyadicRatCore tolerances, and Real seal boundary have been displayed. The route contains no quotient equality, selected limit, countable-choice selector, host real equality, or ambient Real completeness principle.

Proof. Begin with a comparison-facing consumer. Carrier admission gives access to K and Q only as displayed rows: K exposes the located-cut-to-stream route from chapter 2312, and Q exposes the regular-Cauchy classifier route from chapter 1820.

Replay the continuation row C. The request next consumes the synchronized StreamName ledger S, the RegSeqRat readback row R, and the DyadicRatCore tolerance ledger D at the same precision boundary.

Only after this finite replay can the seal boundary A be read. Componentwise transport, provenance, and local naming preserve the same dependency order, so the comparison remains a displayed presentation comparison rather than a quotient, selection, choice, host-equality, or completeness claim. □

Lean4 ✓ `BEDC.Derived.CauchyCutComparisonUp.CauchyCutComparisonRoute`

Falsifiable prediction: Every accepted $\mathsf{CauchyCutComparison}^{\uparrow}$ read that compares located-cut and regular-Cauchy presentations factors through displayed $\mathsf{CauchyCut}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: cauchy-cut, cauchy-equivalence-setoid, streamname, regseqrat, dyadicratcore, real: $\mathsf{CauchyCutComparison}^{\uparrow}$ is not reducible to any listed sibling because it binds the cut-side exposure and setoid-side exposure to one synchronized tolerance ledger and one Real seal boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4073.1, Definition 4073.2, and Definition 4073.3 seed the comparison surface between located-cut and regular-Cauchy presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCutComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-side exposure, setoid-side exposure, synchronized tolerance, Real seal discipline, and ledger refusal for BEDC.Derived.CauchyCutComparisonUp.

Constructive story: A $\mathsf{CauchyCutComparison}^{\uparrow}$ packet is generated from displayed CauchyCut and CauchyEquivalenceSetoid rows, synchronized StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise transport, displayed replay, package provenance, and a local NameCert row.

Chapter 4074

A Concrete Naming Certificate for $\mathsf{FiniteMarkovPartition}^{\uparrow}$

Following Bowen and Walters, the $\mathsf{FiniteMarkovPartition}^{\uparrow}$ packet records the finite symbolic-dynamics surface by which a compact dynamics calculation is read as cylinder data before any consumer asks for a subshift, invariant measure, or orbit quotient. It sits downstream of the finite partition carrier of chapter 201, the adjacency rows of chapter 49, the dynamical-system interface of chapter 195, the local update discipline of chapter 934, and the stream window naming surface of chapter 126. The packet names only a finite Markov partition, its cylinder blocks, adjacency matrix, refinement ledger, transport, replay, provenance, and local NameCert rows; it does not quotient infinite orbits or assert a host-side shift space.

Definition 4074.1 (Finite Markov partition carrier)**.** *A* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (P, B, J, R, T, H, C, S, N)$$

with the following displayed rows:

1. *P is the finite partition row, read through* $\mathsf{Partition}^{\uparrow}$*, naming the carried atoms and their boundary tags.*
2. *B is the finite cylinder-block row, recording the* BHist *words obtained by reading finitely many successive visits to atoms of P.*
3. *J is the adjacency-matrix row, read through* $\mathsf{Mat}^{\uparrow}$*, whose entries record which atom-to-atom transitions are admitted by the carried dynamics.*
4. *R is the refinement row, recording how a cylinder block refines its initial atom, terminal atom, and every intermediate transition listed by J.*
5. *T is the dynamics-transport row, read through* $\mathsf{DynSystem}^{\uparrow}$*, recording the finite action on the displayed atoms and the compatibility route from atom reads to cylinder reads.*
6. *H records componentwise* hsame *transport for partition atoms, cylinder blocks, adjacency entries, refinement rows, dynamics transport, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from atom reads to block reads, from block reads to adjacency tests, and from adjacency tests to refinement rows.*
8. *S is the* Pkg *provenance over* $\mathsf{FiniteMarkovPartition}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $P, B, J, R, T, H, C, S, N$. It has no coordinate for an infinite orbit quotient, a selected itinerary beyond the displayed cylinder window, a measure on a shift space, or an ambient topological conjugacy.

Lean4 ✓ `BEDC.Derived.FiniteMarkovPartitionUp.FiniteMarkovPartitionTasteGate single carrier alignment`

Theorem 4074.2 (Finite Markov partition NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *carrier $M = (P, B, J, R, T, H, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, finite partition exposure, finite cylinder-block exposure, adjacency-matrix exposure, refinement-row exactness, dynamics transport, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the finite packet $M = (P, B, J, R, T, H, C, S, N)$. The partition row P supplies the carried atoms, and the cylinder-block row B reads only finitely many visits to those atoms.

The adjacency row J is then read as a finite matrix over the same atoms. The refinement row R records that each displayed cylinder block refines its atom endpoints and the intermediate transitions permitted by J.

The dynamics row T supplies the finite transport route from atom reads to cylinder reads. Rows H, C, S, N preserve, replay, source, and name exactly that route, so the listed rows exhaust the NameCert obligations. □

Theorem 4074.3 (Finite Markov partition cylinder refinement)**.** *Every accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *cylinder block factors through the partition row P, the adjacency-matrix row J, and the refinement row R. A downstream* $\mathsf{CellularAutomaton}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *consumer may replay the block as a finite transition window, but it receives no route to a hidden infinite itinerary outside B, J, R, H, C, S, N.*

Proof. Start with the carrier of Definition 4074.1. A cylinder block is present only as a row of B, and B is indexed by the finite atoms displayed by P.

The adjacency matrix J checks each neighboring pair in the block. The refinement row R then binds the block to its initial atom, terminal atom, and internal transition witnesses, so the block can be read only through the finite adjacency surface.

Replay through C gives a finite transition window suitable for $\mathsf{CellularAutomaton}^{\uparrow}$ or $\mathsf{StreamName}^{\uparrow}$ consumers. Since the carrier has no coordinate for an unbounded itinerary or orbit quotient, the exported object remains the displayed finite cylinder refinement. □

Theorem 4074.4 (Finite Markov partition symbolic handoff)**.** *An accepted* $\mathsf{FiniteMarkovPartition}^{\uparrow}$ *packet can be handed to a subshift-facing or dynamics-facing consumer only as the finite route*

$$P \longmapsto B \longmapsto J \longmapsto R$$

together with T, H, C, S, N. The handoff supplies a symbolic refinement surface for the carried dynamics, not a completed shift space, invariant measure, orbit quotient, or global conjugacy theorem.

Proof. Apply the NameCert obligations of Definition 4074.2. They expose only the finite partition, cylinder, adjacency, refinement, dynamics-transport, transport, replay, provenance, and local naming rows.

Use cylinder refinement from Definition 4074.3. Every block read has already passed through the displayed adjacency and refinement rows, hence a subshift-facing consumer receives a finite transition surface rather than a hidden stream of choices.

The excluded objects are not carrier coordinates. There is no row for a completed shift space, invariant measure, quotient orbit, or global conjugacy, so the symbolic handoff is exactly the finite NameCert route shown in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMarkovPartition}^{\uparrow}$ packet is generated from finite partition atoms, finite cylinder blocks, adjacency-matrix rows, refinement rows, dynamics transport, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}, \mathsf{seed})$

Scope closed: This chapter binds the carrier definition, NameCert-obligation theorem, cylinder-refinement theorem, symbolic-handoff theorem, and finite symbolic-dynamics boundary for $\mathsf{FiniteMarkovPartition}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMarkovPartition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: It does not claim a completed subshift, invariant measure, orbit quotient, global conjugacy theorem, or infinite itinerary outside the displayed finite cylinder and adjacency rows.

Upgrade path: To reach obligationClosure, the finite carrier, adjacency refinement, symbolic handoff, and ledger non-escape rows must be encoded and checked over BHist packets.

Chapter 4075

A Concrete Naming Certificate for $\mathsf{RegularCauchyPositivePart}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ packet reads the constructive positive part of a regular Cauchy real name through the finite identity $x^+ = max(x, 0)$. The certificate sits beside the absolute-value, maximum, and minimum surfaces of chapter 1499, chapter 1502, and chapter 1528; it uses the regular rational readback of chapter 301, the finite window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The downstream real lattice comparison surface of chapter 1674 consumes the same finite order-operation discipline. The packet does not import trichotomy, host real order, quotient stream equality, selected limits, or ambient completeness.

Definition 4075.1 (RegularCauchyPositivePart carrier)**.** *A* $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N)$$

with the following visible coordinates:

1. *X is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Z is the displayed zero regular Cauchy row over the same rational readback surface.*
3. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to observe X and Z at the same finite precision requests.*
4. *D_X and D_Z are the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-and-tolerance rows read from the shared window schedule.*
5. *A is the* $\mathsf{RegularCauchyAbs}^{\uparrow}$ *compatibility row for the absolute-value route attached to X.*
6. *M is the* $\mathsf{RegularCauchyMax}^{\uparrow}$ *row for the finite maximum route $max(X, Z)$.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the positive-part windows and the displayed tolerance ledger.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the readback row R has been exposed.*
9. *H records componentwise* hsame *transport for the source, zero row, shared window, dyadic rows, absolute-value compatibility, maximum row, readback, seal, provenance, and local name.*
10. *C records the displayed* Cont *routes from the source and zero rows through the shared windows, dyadic observations, maximum row, readback, and real seal.*

11. *L records* Pkg *provenance over* RegularCauchyPositivePart$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegularCauchyAbs$^{\uparrow}$, RegularCauchyMax$^{\uparrow}$, RegularCauchyMin$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{RegularCauchyPositivePart}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N$. It has no coordinate for trichotomy of completed reals, host real order, quotient stream equality, selected limits, or a Real$^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyPositivePartUp.RegularCauchyPositivePartTasteGate single carrier alignment`

Theorem 4075.2 (RegularCauchyPositivePart NameCert obligations)**.** *For every accepted* RegularCauchyPositiveP *carrier $P = (X, Z, W, D_X, D_Z, A, M, R, E, H, C, L, N)$, the local* NameCert$_{\text{RegularCauchyPositivePart}^{\uparrow}}$ *surface consists exactly of source* RegSeqRat$^{\uparrow}$ *admission, displayed zero-row admission, shared* StreamName$^{\uparrow}$ *window exposure,* DyadicRatCore$^{\uparrow}$ *tolerance observations,* RegularCauchyAbs$^{\uparrow}$ *compatibility, the* RegularCauchyMax$^{\uparrow}$ *positive-part row,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed packet P. The coordinates expose the source regular name, the zero row, the shared finite windows, dyadic endpoint-and-tolerance rows, the absolute-value compatibility row, the maximum row, the regular readback, the real seal, and the structural rows.

The operation obligation is the finite lattice route. The maximum surface of chapter 1502 supplies the row M for $max(X, Z)$, while the absolute-value surface of chapter 1499 supplies the compatibility row A used by consumers that read the equivalent absolute-value factorization. The minimum surface of chapter 1528 supplies the sibling lower-part comparison discipline, so the positive-part packet remains aligned with the same displayed-window lattice family.

The readback and seal obligations are then the finite handoffs through RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$. The rows H, C, L, N transport, replay, source, and name this same route without adding trichotomy, host real order, quotient streams, selected limits, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.RegularCauchyPositivePartUp.RegularCauchyPositivePartCarrier namecert obligations`

Theorem 4075.3 (RegularCauchyPositivePart Real seal handoff)**.** *Every* Real$^{\uparrow}$ *or real-lattice consumer of an accepted* RegularCauchyPositivePart$^{\uparrow}$ *carrier reaches the positive-part seal row E only after the shared source and zero windows, the dyadic tolerance rows, the* RegularCauchyMax$^{\uparrow}$ *row M, and the* RegSeqRat$^{\uparrow}$ *readback row R have been exposed. The route is compatible with the* RealMinMax$^{\uparrow}$ *lattice boundary of chapter 1674, but it exports no host order comparison, trichotomy witness, quotient stream equality, selected limit, or ambient completeness principle.*

Proof. Start with a Real$^{\uparrow}$ or lattice-facing consumer. By Definition 4075.1, the only real-facing endpoint in the packet is the seal row E, and that row is downstream of the readback row R.

The readback is downstream of the finite maximum row M. The maximum route consumes the source row X, the displayed zero row Z, the shared window schedule W, and the dyadic tolerance rows D_X, D_Z before the positive-part value can be named.

Finally compare the route with the real min/max boundary. The sibling RealMinMax$^{\uparrow}$ surface consumes displayed real seals through finite min/max rows, so this packet hands it a positive-part seal already produced by regular-Cauchy windows, dyadic tolerance rows, readback, transport,

replay, provenance, and local naming. Since none of the excluded analytic principles occurs as a coordinate of P, no consumer can reach the positive part by a non-displayed route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ consumer that exports a positive-part seal without first reading the source RegSeqRat row, displayed zero row, shared StreamName windows, DyadicRatCore tolerance rows, RegularCauchyMax row, RegSeqRat readback, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: regular Cauchy positive part: $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ binds a source RegSeqRat row, displayed zero row, shared StreamName windows, DyadicRatCore tolerance observations, RegularCauchyAbs compatibility, RegularCauchyMax readback, RegSeqRat handoff, RealUp seal, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite lattice-operation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyPositivePart}^{\uparrow}$ packet is generated from one RegSeqRat source row, a displayed zero row, shared StreamName windows, DyadicRatCore endpoint observations, a RegularCauchyAbs compatibility row, a RegularCauchyMax row, a RegSeqRat readback row, a RealUp seal row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4075.1, Definition 4075.2, and Definition 4075.3 seed the finite constructive positive-part route through regular Cauchy maximum, readback, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyPositivePart}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, trichotomy on completed real names, host real order, quotient stream equality, selected limits, ambient completeness, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, zero-row admission, window exposure, dyadic tolerance rows, RegularCauchyMax handoff, RealUp seal boundary, transport stability, provenance, and non-escape for the same finite packet.

Chapter 4076

A Concrete Naming Certificate for $\mathsf{Equicontinuity}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley-style compact-metric reasoning, the $\mathsf{Equicontinuity}^{\uparrow}$ packet records the finite family-modulus surface by which a compact source, a finite continuous-family ledger, and a shared tolerance-radius schedule are presented before any finite-net selector or Arzela–Ascoli consumer may read the family. It sits over the compact source of chapter 129, the continuous graph and pointwise-modulus rows of chapter 130, the tolerance-only modulus consumer of chapter 369, and the finite-net family route of chapter 2362.

The packet exposes a shared family modulus. It does not assert compactness, select subsequences, provide a finite net, choose an Arzela–Ascoli limit, or import host-level continuity.

Definition 4076.1 (Equicontinuity carrier)**.** *An* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, \epsilon, \rho, M, T, R, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row whose finite-cover and metric inspection data are read through* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *F is the finite continuous-family row, each member of which is presented through* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus data over the same source.*
3. *ϵ is the tolerance row consumed by every family member before a radius is exported.*
4. *ρ is the shared radius row, read as a single family modulus rather than as one radius per function.*
5. *M is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that records the shared tolerance-radius handoff.*
6. *T records componentwise* hsame *transport for K, F, ϵ, ρ, M.*
7. *R records displayed* Cont *replay from compact-source observations through continuous-family rows into the shared-modulus handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected finite net, selected subsequence, compactness theorem, host continuity predicate, pointwise-only modulus family, or ambient completeness principle.

Lean4 (definition) `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier`

Theorem 4076.2 (Equicontinuity NameCert obligations)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-family ledger, tolerance row, shared radius row, modulus handoff, transport, replay, provenance, and local naming; shared-modulus exactness requiring the same* ρ *to serve every member of* F *at tolerance* ϵ*; compact-source compatibility requiring any finite-net consumer to read* K *before using the modulus row; and nonescape excluding compactness assertions, selected finite nets, selected subsequences, host continuity, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4076.1. The tuple displays exactly $K, F, \epsilon, \rho, M, T, R, P, N$.

Shared-modulus exactness is the role of ρ and M. The radius row is a single displayed row for the family, so a consumer cannot replace it by independently chosen radii for different members of F.

Compact-source compatibility is the route through K. A finite-net or compact-uniform consumer must read the compact metric source before it can consume the shared family modulus.

Finally, T, R, P, N transport, replay, source, and name the same displayed rows. Since the carrier has no selected net, selected subsequence, compactness theorem, host-continuity predicate, or ambient-completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4076.3 (Equicontinuity compact-metric route)**.** *Every compact-uniform or finite-net consumer of an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier factors through*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The route first reads the compact metric source K*, then the continuous-family ledger* F*, then the shared tolerance-radius handoff* ϵ, ρ, M*. It may be consumed by the finite-net equicontinuity surface of chapter 2362, but it does not itself supply a finite net or compactness proof.*

Proof. Project the accepted carrier. The only source coordinate is K, so any compact-uniform consumer starts from the displayed compact metric row.

Read the family row F next. Its entries are continuous-map rows over the same source, so the family cannot change source data while the modulus is being consumed.

The shared handoff is ϵ, ρ, M. It records a tolerance and one family radius before a downstream finite-net selector reads the packet. Transport and replay through T, R preserve this order, while P, N merely package and name it. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactMetricRoute`

Theorem 4076.4 (Equicontinuity nonescape)**.** *An accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier exports only the finite family-modulus rows* $K, F, \epsilon, \rho, M, T, R, P, N$*. It cannot export compactness, a selected finite net, a selected subsequence, Arzela–Ascoli convergence, host-level continuity, pointwise-only radius choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier definition lists the entire public inventory. The analytic rows are the compact source, continuous family, tolerance, shared radius, and modulus handoff.

The structural rows transport and replay those same rows. They do not add a selector, a compactness principle, a subsequence object, a convergence row, or an external continuity predicate.

Thus every downstream route must obtain finite nets, compactness conclusions, and convergence statements from separate consumer chapters. The $\mathsf{Equicontinuity}^{\uparrow}$ packet supplies only the displayed family-modulus boundary. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityTasteGate single carrier alignment`

Theorem 4076.5 (Equicontinuity family-window exposure)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family window exposed to compact-uniform consumers is the finite* BHist *route through the* $\mathsf{CompactMetric}^{\uparrow}$ *source row* K*, the* $\mathsf{ContinuousMap}^{\uparrow}$ *family ledger* F*, the tolerance row* ϵ*, the shared radius row* ρ*, the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff* M*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. Each pointwise modulus read for a member of* F *must occur inside that window before any finite-net radius is consumed.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFamilyWindowExposure`

Proof. Project the carrier rows. The finite family window is exactly the displayed sequence $K, F, \epsilon, \rho, M, T, R, P, N$, so the compact source and family ledger are read before the shared-radius handoff.

The $\mathsf{ContinuousMap}^{\uparrow}$ pointwise-modulus entries live in F. They cannot be detached from the $\mathsf{CompactMetric}^{\uparrow}$ source K, because the compact-metric route forces every downstream read through the same source row.

Finally, T, R, P, N only transport, replay, source, and name this window. A finite-net consumer therefore receives pointwise modulus data only through the displayed family window and not as an external radius choice. □

Theorem 4076.6 (Equicontinuity compact-net radius exactness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier. Any* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net radius consumed from* E *is the same shared row* ρ *that is handed to* $\mathsf{UniformModulus}^{\uparrow}$ *by* M *after the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family window has been displayed. The* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows preserve this radius exactly and add no memberwise radius ledger, selected net, or host compactness proof.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier compact net radius exactness`

Proof. Read ρ between the tolerance row ϵ and the handoff row M. The carrier contains one shared radius row, not a list of member-indexed radii.

Use the compact-metric route and family-window exposure. The compact source K and the finite continuous-family ledger F are visible before the radius is consumed by a finite-net or uniform-modulus reader.

Transport and replay preserve the row rather than recomputing it. Thus the finite-net radius available to a compact consumer is exactly ρ through T, R, P, N, with no selected net or host compactness coordinate added by the packet. □

Theorem 4076.7 (Equicontinuity uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *handoff exported by an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier factors through the finite route*

$$K, F, \epsilon, \rho, M, T, R, P, N.$$

The handoff reads the CompactMetric$^{\uparrow}$ *finite-net radius boundary through* K, ρ*, reads each* ContinuousMap$^{\uparrow}$ *pointwise modulus through the finite family row* F*, and then exposes* M *as the shared* UniformModulus$^{\uparrow}$ *consumer row. The* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support rows export no pointwise-only modulus family, selected finite net, or ambient* Real$^{\uparrow}$ *completeness claim.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityUniformModulusHandoff`

Proof. Start from family-window exposure. It fixes the order in which K, F, ϵ, and ρ are displayed before the handoff row M can be read.

Apply compact-net radius exactness. The radius consumed by the compact finite-net boundary is the same shared row ρ, and every pointwise modulus entry remains attached to the finite ContinuousMap$^{\uparrow}$ ledger F.

The structural rows then repack the handoff. Since T, R, P, N preserve and name only the displayed route, the exported UniformModulus$^{\uparrow}$ row is a shared handoff and not a selected net, pointwise-only radius family, or completeness principle. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier uniform modulus handoff`

Theorem 4076.8 (Equicontinuity shared-radius stability)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* Equicontinuity$^{\uparrow}$ *carrier. Any transport of the compact source* K*, finite family row* F*, tolerance* ϵ*, and modulus handoff* M *through* T *preserves the same shared radius row* ρ*. Thus a consumer reading* CompactMetric$^{\uparrow}$*,* ContinuousMap$^{\uparrow}$*,* UniformModulus$^{\uparrow}$*, or* FiniteNetEquicontinuity$^{\uparrow}$ *through* R, P, N *cannot replace* ρ *by pointwise radii depending on members of* F*.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusStability`

Proof. Project the carrier of Definition 4076.1. The row ρ is displayed once, between the tolerance row ϵ and the modulus handoff row M.

Transport through T is componentwise on the displayed rows. It may identify the compact source, the family row, the tolerance row, and the handoff row with transported copies, but it has no coordinate for a family-indexed list of radii.

Replay through R and provenance through P, N therefore reread the same shared-radius row. The compact metric, continuous-map, uniform-modulus, and finite-net consumers inherit the single ρ boundary rather than a pointwise-radius surface. □

Theorem 4076.9 (Equicontinuity compact-source root admission)**.** *Every accepted* Equicontinuity$^{\uparrow}$ *carrier first exposes the compact metric source row* K *before any shared-radius, finite-net, or Arzela–Ascoli-facing consumer reads* $F, \epsilon, \rho, M, T, R, P, N$*. The exposed row is the displayed* CompactMetric$^{\uparrow}$ *source of Definition 4076.1, transported by* hsame*, replayed by* Cont*, and sourced by the same* Pkg *provenance and local* NameCert *row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootCompactSourceAdmission`

Proof. Project the carrier of Definition 4076.1. Its first analytic coordinate is the compact metric source row K, and every later row is attached to that source.

Apply the NameCert obligation surface and compact-metric route, Definition 4076.2 and Definition 4076.3. They force the finite family, tolerance, shared radius, and modulus handoff to be read only after K has been admitted.

Finally use shared-radius stability. Transport and replay preserve the same K row while Pkg provenance and local naming expose the packet, so no downstream consumer can bypass the displayed CompactMetric$^{\uparrow}$ source. □

Theorem 4076.10 (Equicontinuity family-modulus exactness root admission)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier, the shared radius row ρ is the unique family-radius handoff visible to finite-net consumers. It is read after the compact source K, finite* ContinuousMap$^{\uparrow}$ *family F, and tolerance row ϵ, and before the* UniformModulus$^{\uparrow}$ *or* FiniteNetEquicontinuity$^{\uparrow}$ *consumer surface; no memberwise radius schedule or second modulus row is exported.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFamilyModulusExactness`

Proof. Read K, F, ϵ, ρ, M from the carrier. The row ρ appears once, between the tolerance request and the modulus handoff.

Apply shared-radius stability and modulus-ledger exactness, Definition 4076.8 and Definition 4076.20. They identify ρ as the transported shared radius and make the handoff exact for the finite family ledger.

Use finite-net readiness to locate the downstream boundary. A FiniteNetEquicontinuity$^{\uparrow}$ consumer sees the same ρ through hsame, Cont, Pkg, and NameCert rows, and the nonescape theorem excludes memberwise radius replacement. □

Theorem 4076.11 (Equicontinuity finite-net consumer boundary root admission)**.** *The* Equicontinuity$^{\uparrow}$ *carrier supplies only compact-source, finite-family, tolerance, shared-radius, modulus-handoff, transport, replay, provenance, and local naming rows. Any finite net, compactness output, Arzela–Ascoli convergence, or selected subsequence must be obtained from downstream consumer chapters after reading this boundary, not from* Equicontinuity$^{\uparrow}$ *itself.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetConsumerBoundary`

Proof. Project the carrier and the NameCert obligations. The analytic rows are K, F, ϵ, ρ, M, and the structural rows are T, R, P, N.

Apply finite-net readiness and the nonescape theorem, Definition 4076.16 and Definition 4076.4. They place finite-net access after the compact-source and shared-modulus rows while excluding a selected net, subsequence, compactness theorem, or convergence row from the carrier.

Thus finite-net and Arzela–Ascoli consumers may depend on the explicit carrier boundary, but their nets and convergence conclusions must come from downstream chapters. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetConsumerBoundary`

Theorem 4076.12 (Equicontinuity shared-radius carrier root row)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the shared radius row ρ is exposed as a finite* BHist *row between the tolerance request ϵ and the modulus handoff M. The same packet displays the compact source K, finite* ContinuousMap$^{\uparrow}$ *family F, componentwise* hsame *transport T,* Cont *replay R,* Pkg *provenance P, and local* NameCert *row N, so any* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, *or* FiniteNetEquicontinuity$^{\uparrow}$ *consumer reads one shared-radius carrier route rather than memberwise radius data.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedRadiusCarrier`

Proof. Project the carrier of Definition 4076.1. The tuple displays exactly $K, F, \epsilon, \rho, M, T, R, P, N$, and ρ appears as one row before the modulus handoff.

Apply the finite-net consumer boundary of Definition 4076.11. It keeps every consumer on the compact-source, finite-family, tolerance, shared-radius, and handoff rows before any finite-net-facing read begins.

The support rows T, R, P, N only transport, replay, source, and name the displayed BHist route. Since no memberwise radius row is present, the root carrier exposes the shared ρ boundary alone. □

Theorem 4076.13 (Equicontinuity compact-modulus consumer root row)**.** *Let $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ be an accepted* Equicontinuity$^{\uparrow}$ *carrier. A compact-modulus consumer must first read K as the* CompactMetric$^{\uparrow}$ *source, then F as the finite* ContinuousMap$^{\uparrow}$ *ledger over that source, then ϵ, ρ, M as the shared* UniformModulus$^{\uparrow}$ *handoff, with the route preserved by* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *support. The consumer cannot treat M as detached from the compact source or as a selected finite net.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockCompactModulusConsumer`

Proof. Read K and F from the accepted carrier. The compact source row precedes the continuous-family ledger, and the family is displayed over that same source.

Use compact-source admission and finite-net readiness, Definition 4076.9 and Definition 4076.16. They order the compact and family reads before ϵ, ρ, M can be consumed by a uniform-modulus or finite-net-facing reader.

Finally apply nonescape. The support rows preserve the displayed route, while selected finite nets, detached compactness output, subsequences, and convergence rows are absent from the Equicontinuity$^{\uparrow}$ packet. □

Theorem 4076.14 (Equicontinuity finite-net handoff root row)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the finite-net handoff is the ordered* BHist *route from the* CompactMetric$^{\uparrow}$ *source K through the* ContinuousMap$^{\uparrow}$ *family F and the shared tolerance-radius rows ϵ, ρ, M into the* FiniteNetEquicontinuity$^{\uparrow}$ *consumer boundary. The* hsame*,* Cont*,* Pkg*, and* NameCert *rows T, R, P, N preserve that route and do not add compactness output, selected nets, selected subsequences, or an Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetHandoff`

Proof. Project the rows $K, F, \epsilon, \rho, M, T, R, P, N$. The analytic part of the packet already orders the compact source, continuous family, tolerance request, shared radius, and modulus handoff.

Apply the shared-radius carrier row above and the compact-modulus consumer row above. Together they make the finite-net read pass through the same compact source and shared radius before the downstream FiniteNetEquicontinuity$^{\uparrow}$ boundary is available.

The structural rows transport and replay only this displayed handoff. Because finite nets and convergence objects are not carrier coordinates, the handoff remains a root boundary rather than a selected-net construction. □

Theorem 4076.15 (Equicontinuity shared-radius obligation)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$, the local* NameCert$_{\text{Equicontinuity}^{\uparrow}}$ *obligation requires the single radius row ρ to serve every displayed member of F at tolerance ϵ before any* FiniteNetEquicontinuity$^{\uparrow}$ *or Arzela–Ascoli consumer may read the packet. No consumer may replace ρ by memberwise radii, a pointwise-only modulus family, or a selected finite-net radius outside the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier shared radius obligation`

Proof. Project the carrier of Definition 4076.1. The family row F is finite and displayed over the compact metric source K, while ρ appears as one row between the tolerance ϵ and the modulus handoff M.

Use the compact-metric route of Definition 4076.3. It forces consumers to read K, F, ϵ, ρ, M before transport, replay, provenance, and local naming can expose the packet.

Apply shared-radius stability and finite-net readiness, Definition 4076.8 and Definition 4076.16. Together they keep ρ stable under transport and make it available before any finite-net or Arzela–Ascoli-facing read, so pointwise-radius leakage is outside the accepted carrier. □

Theorem 4076.16 (Equicontinuity finite-net readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, a* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer may begin only after the compact source* K*, the continuous family ledger* F*, the tolerance row* ϵ*, the shared radius row* ρ*, and the modulus handoff* M *have all been exposed. The rows* T, R, P, N *preserve this order and do not themselves select a finite net.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetReadiness`

Proof. Read the carrier in the order stated by Definition 4076.3. The compact source K is the only source row, and the family ledger F is the only row naming the continuous maps over that source.

The finite-net consumer then reads ϵ, ρ, M. These rows record a tolerance, a shared radius, and the handoff to the uniform-modulus surface before any finite-net selector can inspect the packet.

The structural rows T, R, P, N transport, replay, source, and name the same displayed route. Since Definition 4076.1 has no finite-net coordinate, readiness for $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ is a boundary condition, not a selected net. □

Theorem 4076.17 (Equicontinuity family-modulus consumer exhaustion)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, every compact-uniform modulus consumer of* E *must read exactly the rows* $K, F, \epsilon, \rho, M, T, R, P, N$*. The consumer may use* K *as the compact source,* F *as the displayed family ledger,* ϵ *as the tolerance request,* ρ *as the shared radius,* M *as the modulus handoff, and* T, R, P, N *as the structural transport, replay, provenance, and naming rows. It cannot detach a pointwise radius schedule from* F*, replace* ρ *by memberwise radii, or insert a selected finite-net row into the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier family modulus consumer exhaustion`

Proof. Start from shared-radius stability in Definition 4076.8. It pins ρ to the displayed compact source K, family ledger F, tolerance row ϵ, and modulus handoff M.

Use finite-net readiness from Definition 4076.16. A finite-net-facing or compact-uniform consumer may begin only after K, F, ϵ, ρ, M have been exposed, while T, R, P, N preserve that same route.

Finally apply nonescape from Definition 4076.4. Since the carrier has no coordinate for memberwise radius schedules, selected finite nets, subsequences, host continuity, or ambient completeness, the rows $K, F, \epsilon, \rho, M, T, R, P, N$ exhaust the compact-uniform consumer surface. □

Theorem 4076.18 (Equicontinuity finite-net selector readiness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier. A* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *selector may start only after the compact-source row* K*, continuous-family row* F*, shared-radius row* ρ*, and modulus-handoff row* M *are exposed for the same tolerance request* ϵ*. The structural rows* T, R, P, N *may transport and replay that readiness boundary, but they do not authorize a selected finite net before the compact source, family, shared radius, and modulus handoff have all been read.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteNetSelectorReadiness`

Proof. Read the prerequisite order from Definition 4076.16. It places a finite-net consumer after the compact source, continuous family ledger, tolerance request, shared radius, and modulus handoff.

Apply shared-radius stability from Definition 4076.8. The selector cannot split the family into detached pointwise radius schedules, because the radius row ρ is transported with K, F, ϵ, M.

Use nonescape from Definition 4076.4. Selected nets, subsequences, limiting maps, convergence certificates, host continuity, and ambient completeness are absent from the carrier, so the selector receives only readiness at the displayed boundary. □

Theorem 4076.19 (Equicontinuity shared-modulus sibling route)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, any downstream consumer of the equicontinuity packet must read the compact source through chapter 129, the continuous family through chapter 130, the uniform modulus handoff through chapter 369, and the finite-net boundary through chapter 2362. The route preserves the same shared radius* ρ *and exports no selected finite net, subsequence, host continuity theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedModulusScope`

Proof. Project the carrier of Definition 4076.1. The row K is the compact source, F is the finite continuous-family ledger over that source, and ϵ, ρ, M are the shared tolerance-radius and modulus handoff rows.

Use the compact-source admission and continuous-family admission theorems, Definition 4076.25 and Definition 4076.26. They identify the source and family reads with the sibling surfaces named by chapter 129 and chapter 130.

Apply shared-radius stability, modulus-handoff admission, and finite-net readiness, Definition 4076.8, Definition 4076.27, and Definition 4076.16. These rows keep the same ρ attached to the handoff into the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ consumers.

Finally use Definition 4076.4. Since the selected-net, subsequence, host-continuity, and ambient-completeness coordinates are absent from the carrier, the sibling route just listed is the whole public dependency surface. □

Theorem 4076.20 (Equicontinuity modulus-ledger exactness)**.** *Let* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *be accepted. The modulus ledger exported by* $\mathsf{Equicontinuity}^{\uparrow}$ *is exactly the handoff from the compact source* K *and finite family* F *at tolerance* ϵ *to the shared radius* ρ *recorded by* M*, transported by* T*, replayed by* R*, sourced by* P*, and named by* N*. It has no additional ledger row for host continuity, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected covers, or an Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier modulus ledger exactness`

Proof. The analytic part of the carrier is K, F, ϵ, ρ, M. The compact-metric row supplies the source, the continuous-map row supplies the finite family, and the uniform-modulus row records the tolerance-to-radius handoff.

The ledger is exact because T, R, P, N refer back to those same rows. They cannot add a second modulus, a selected cover, a limit candidate, or a host-level continuity predicate; they only preserve and name the displayed handoff.

The exclusions are also forced by Definition 4076.4. Any consumer needing compactness output, selected nets, subsequences, or completeness must obtain those data from a separate chapter after reading the exact modulus ledger here. □

Definition 4076.21 (Equicontinuity finite-modulus subcarrier)**.** *An* $\mathsf{Equicontinuity}^{\uparrow}$ *finite-modulus subcarrier is the finite row packet*

$$E_{\mathrm{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$$

projected from an accepted carrier $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*. The rows are, in order, the compact metric source* K*, finite continuous-family ledger* F*, shared tolerance request* ϵ*, shared radius row* ρ*, uniform-modulus handoff* M*, finite-net consumer boundary* G*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. The boundary row* G *is readable only as readiness for* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *over the displayed* K, F, ϵ, ρ, M *route; it is not a selected net, subsequence, compactness conclusion, or Arzela–Ascoli limit.*

Theorem 4076.22 (Equicontinuity compact-metric finite-modulus handoff)**.** *Every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *finite-modulus subcarrier* $E_{\mathrm{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$ *hands downstream consumers a compact-source route only after* K *has been read as the compact metric source and before* F, ϵ, ρ, M, G *are inspected. The rows* T, R, P, N *preserve this compact-source handoff and cannot replace it by an ambient compactness assertion or selected finite net.*

Proof. Project E_{fin} from Definition 4076.21. Its first row is the same compact source K already displayed in Definition 4076.1.

Apply Definition 4076.25 and Definition 4076.3. They force every continuous-family, uniform-modulus, and finite-net-facing read to pass through K before F, ϵ, ρ, M are consumed.

The boundary row G is therefore only a consumer boundary for the existing route. Transport, replay, provenance, and naming through T, R, P, N preserve that route and add no selected net or ambient compactness coordinate. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityFiniteModulusCarrier compactmetric handoff`

Theorem 4076.23 (Equicontinuity continuous-map finite-modulus route)**.** *For every accepted finite-modulus subcarrier* $E_{\mathrm{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$*, the family row* F *is read only as a finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger over the already displayed compact source* K*. The shared tolerance request* ϵ*, radius row* ρ*, and modulus handoff* M *are attached to that same ledger, so no consumer may detach the modulus route into host-level continuity, a source-changing family, or memberwise radii.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousMapModulusRoute`

Proof. Read the finite family row F after the compact source K. By Definition 4076.21, these are the same rows projected from the accepted $\mathsf{Equicontinuity}^{\uparrow}$ carrier.

Apply Definition 4076.26 and Definition 4076.32. They identify F as the finite $\mathsf{ContinuousMap}^{\uparrow}$ ledger over K and exclude host continuity, source-changing maps, and infinite-family replacement.

Use Definition 4076.8. The shared radius row ρ is transported with K, F, ϵ, M, so the modulus route remains attached to the same finite family ledger. □

Theorem 4076.24 (Equicontinuity uniform-modulus consumer boundary)**.** *For every accepted finite-modulus subcarrier* $E_{\mathrm{fin}} = (K, F, \epsilon, \rho, M, G, T, R, P, N)$*, a* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$*, or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer may read the boundary* G *only after the shared handoff*

$$K, F, \epsilon, \rho, M$$

has been displayed. The boundary exports readiness for the finite-net consumer and no selected finite net, subsequence, compactness proof, Arzela–Ascoli limit, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier uniform modulus consumer boundary`

Proof. Project the handoff rows K, F, ϵ, ρ, M and the boundary row G from Definition 4076.21. The row G appears downstream of the shared tolerance-radius handoff.

Apply Definition 4076.16 and Definition 4076.33. They require every finite-net-facing consumer to preserve the compact source, family ledger, tolerance request, shared radius, and modulus handoff before reading any consumer boundary.

Finally use Definition 4076.20 and Definition 4076.4. The displayed handoff is the whole exported modulus ledger, and the excluded selected nets, subsequences, compactness conclusions, limits, quotient equalities, and completeness principles are absent from the packet. □

Theorem 4076.25 (Equicontinuity compact-source admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the compact source row* K *is admitted through* $\mathsf{CompactMetric}^{\uparrow}$ *before any finite-net, shared-radius, or modulus consumer may inspect* F, ϵ, ρ, M*. No downstream* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*, or Arzela–Ascoli-facing read can bypass* K *or replace it by an ambient compactness assertion.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactSourceAdmission`

Proof. Project the carrier of Definition 4076.1. The row K is the only compact-source coordinate, and the introduction fixes it as the same compact metric surface used by chapter 129.

Use the compact-metric route and finite-net readiness theorems, Definition 4076.3 and Definition 4076.16. They order every consumer read through K before the family ledger, tolerance, shared radius, and handoff rows.

The nonescape theorem then excludes an ambient compactness replacement. Since selected finite nets and compactness conclusions are absent from the carrier, the admitted compact source is precisely the displayed $\mathsf{CompactMetric}^{\uparrow}$ row. □

Theorem 4076.26 (Equicontinuity continuous-family admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family row* F *is a finite ledger of* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-modulus rows over the same source* K*. Transport through* T *and replay through* R *preserve this common source, so a consumer cannot read the family as host-level continuity, source-changing maps, or pointwise-only radius data detached from* K*.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousFamilyAdmission`

Proof. Read F from Definition 4076.1. Its entries are continuous-map rows over the compact source row K, matching the continuous-map sibling cited in the chapter introduction.

Apply shared-radius stability. The theorem keeps ρ attached to the transported compact source, finite family, tolerance, and modulus handoff, so F cannot be split into unrelated pointwise-radius ledgers.

Finally apply nonescape. Host-level continuity and source-changing maps are not coordinates of E, and T, R, P, N only transport, replay, source, and name the displayed family ledger. □

Theorem 4076.27 (Equicontinuity modulus-handoff admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared tolerance-radius row* ϵ, ρ, M *is the only* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *handoff available to downstream finite-net consumers. A consumer that reads* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *must receive the handoff through this row and cannot add a second modulus ledger, a memberwise radius schedule, or a selected finite-net radius outside the carrier.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityModulusHandoffAdmission`

Proof. Project the handoff rows ϵ, ρ, M. The carrier displays one tolerance, one shared radius, and one modulus consumer row.

Use the shared-radius obligation and finite-net readiness theorems. They force every finite-net or Arzela–Ascoli-facing consumer to read ϵ, ρ, M after K, F and before any finite-net action can begin.

The modulus-ledger exactness theorem identifies this handoff as the whole exported modulus ledger. Since there is no coordinate for a second modulus, a memberwise radius family, or an outside selected radius, the downstream read is confined to the displayed $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ handoff. □

Theorem 4076.28 (Equicontinuity compact-source root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root source lock is the compact metric row* K *together with the transported route* T, R, P, N*. Every* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*, or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer must read* K *before the family ledger* F*, tolerance* ϵ*, shared radius* ρ*, and modulus handoff* M *become available. The lock exports no selected finite net, subsequence, or Arzela–Ascoli limit.*

Proof. Project the carrier. The only source row is K, and the rows F, ϵ, ρ, M are displayed downstream of that source in the finite packet.

Apply compact-source admission and the compact-metric route. These place every continuous-family, uniform-modulus, and finite-net-facing read after K and before the structural rows T, R, P, N can expose the named packet.

Finally use nonescape. Since the carrier has no selected-net, subsequence, or convergence-limit coordinate, the root lock is exactly the displayed compact-source route. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootCompactSourceLock`

Theorem 4076.29 (Equicontinuity family-ledger root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root family ledger is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *row* F *over the already-read compact source* K*. Transport, replay, provenance, and local naming through* T, R, P, N *preserve that ledger before* ϵ, ρ, M *are consumed by* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$*. No consumer may replace* F *by an infinite family, host-level continuity, selected finite net, subsequence, or Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootFamilyLedgerLock`

Proof. Read K and F from the carrier. The row F is finite and its entries are continuous-map rows over the same compact source.

Use continuous-family admission and continuous-family ledger admission. They force the consumer route through the displayed $\mathsf{ContinuousMap}^{\uparrow}$ ledger rather than through host continuity or a source-changing family.

The rows T, R, P, N only preserve, replay, source, and name the same ledger. Nonescape excludes the listed outside objects, so the root lock is the finite family ledger itself. □

Theorem 4076.30 (Equicontinuity root family modulus ledger)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root family-modulus ledger exposed to finite-net consumers is exactly the compact-source row* K*, finite continuous-family row* F*, shared tolerance-radius rows* ϵ, ρ*, modulus handoff* M*, componentwise* hsame *transport* T*, displayed* Cont *replay* R*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$ *row* N*. No consumer may replace this ledger by selected nets, selected subsequences, host compactness, host continuity, a second modulus ledger, or Arzela–Ascoli convergence.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootFamilyModulusLedger`

Proof. Project the carrier. The compact-source lock and family-ledger root lock identify K and F as the only source and family rows available before any modulus-facing read.

Apply shared-radius stability, modulus-handoff admission, and modulus-ledger exactness. These place ϵ, ρ, M after K, F and identify the shared handoff as the whole exported modulus ledger for finite-net consumers.

Finally use transport replay provenance and nonescape. The rows T, R, P, N preserve the displayed ledger without adding a selected net, subsequence, host compactness proof, host continuity theorem, second modulus ledger, or Arzela–Ascoli convergence row. □

Theorem 4076.31 (Equicontinuity modulus-consumer root lock)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the root modulus-consumer lock is the shared handoff* ϵ, ρ, M *after the compact source* K *and family ledger* F *have been displayed. Any* $\mathsf{UniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *consumer must preserve this handoff through* T, R, P, N *and cannot add a selected finite net, memberwise radius schedule, subsequence, or Arzela–Ascoli limit.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootModulusConsumerLock`

Proof. Project the handoff rows. The packet contains one tolerance row, one shared radius row, and one modulus consumer row.

Apply modulus-handoff admission and shared-modulus handoff preservation. These theorems make ϵ, ρ, M the only modulus handoff available after K and F have been read.

Since the carrier has no coordinate for a selected finite net, memberwise radii, subsequences, or Arzela–Ascoli limits, transport and replay through T, R, P, N preserve only the displayed shared-modulus consumer lock. □

Theorem 4076.32 (Equicontinuity continuous-family ledger admission)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the family row* F *is admitted only as a finite* $\mathsf{ContinuousMap}^{\uparrow}$ *ledger over the compact source* K*. No consumer may reread* F *as host-level continuity, a source-changing family, or an infinite family outside the displayed compact-source route.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityContinuousFamilyLedgerAdmission`

Proof. Project the carrier of Definition 4076.1. It displays K as the compact metric source and F as the finite continuous-family row over that same source.

Use the compact-metric route and continuous-family admission theorems, Definition 4076.3 and Definition 4076.26. They force every consumer to read K before F and keep the family inside the displayed $\mathsf{ContinuousMap}^{\uparrow}$ ledger.

Finally apply Definition 4076.4. The excluded host-continuity, source-changing, and infinite-family coordinates are absent from E. □

Theorem 4076.33 (Equicontinuity shared-modulus handoff preservation)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, any* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *or* $\mathsf{UniformModulus}^{\uparrow}$ *consumer must preserve the shared tolerance-radius rows* ϵ, ρ, M *before reading a finite-net selector. The consumer cannot replace them by memberwise radii, a second modulus ledger, or a selected net radius outside the packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier shared modulus handoff preservation`

Proof. Read the shared-radius stability and finite-net readiness rows, Definition 4076.8 and Definition 4076.16. They place ϵ, ρ, M after the compact source and family rows, and before any finite-net-facing read.

Apply modulus-handoff admission and modulus-ledger exactness, Definition 4076.27 and Definition 4076.20. These identify the shared tolerance-radius handoff as the whole exported modulus ledger.

Since the carrier has no selected-net coordinate, preservation of ϵ, ρ, M is forced before a finite-net selector can be read. □

Theorem 4076.34 (Equicontinuity transport replay provenance)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the transport row* T*, replay row* R*, provenance row* P*, and naming row* N *preserve only the displayed compact-source, continuous-family, tolerance-radius, and modulus-handoff rows. They cannot add pointwise radius choice, selected nets, subsequences, or Real completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCarrier transport replay provenance`

Proof. Project T, R, P, N from Definition 4076.1. Their only analytic inputs are K, F, ϵ, ρ, M.

Use shared-radius stability and modulus-ledger exactness. Transport and replay preserve the same shared radius row and the same modulus ledger rather than a pointwise-radius family.

The nonescape theorem excludes selected nets, subsequences, and ambient $\mathsf{Real}^{\uparrow}$ completeness. Provenance and local naming therefore record the displayed route only. □

Theorem 4076.35 (Equicontinuity shared-radius family-cover readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared-radius family-cover boundary is the displayed route*

$$K, F, \epsilon \longmapsto \rho, M \longmapsto T, R, P, N.$$

The row ρ *is readable only after the compact source* K*, finite continuous-family ledger* F*, and tolerance row* ϵ *have been exposed, and every family-cover consumer must keep* M, T, R, P, N *visible. The route cannot be replaced by selected nets, selected subsequences, host continuity, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusFamilyCoverReadiness`

Proof. Project the carrier. The rows K, F, ϵ are the only source, family, and tolerance data available before the shared radius is read.

Use shared-radius stability and shared-modulus handoff preservation. These results place ρ and M after the displayed source-family-tolerance rows and before structural transport and replay.

Finally apply transport replay provenance and nonescape. The rows T, R, P, N preserve the same boundary, and the carrier lacks coordinates for selected nets, selected subsequences, host continuity, and ambient real completeness. □

Theorem 4076.36 (Equicontinuity compact family-net readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, a finite-net consumer may read the family only through the compact-source route*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The compact family-net boundary consists of the compact metric source K*, the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row* F*, the tolerance-radius handoff* ϵ, ρ, M*, and the transport, replay, provenance, and naming rows* T, R, P, N*. It does not choose a net, choose a subsequence, supply host continuity, or invoke ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactFamilyNetReadiness`

Proof. Start with continuous-family ledger admission. It admits F only as a finite $\mathsf{ContinuousMap}^{\uparrow}$ ledger over the already displayed compact source K.

Next use finite-net readiness and modulus-consumer root lock. They force any finite-net-facing read to pass through ϵ, ρ, M before the structural rows are replayed.

The rows T, R, P, N do not add a selector. By transport replay provenance and nonescape, the accepted family-net read remains a compact-source ledger route, excluding selected nets, selected subsequences, host continuity, and ambient real completeness. □

Theorem 4076.37 (Equicontinuity Arzela–Ascoli boundary readiness)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, an Arzela–Ascoli-facing consumer may cite* $\mathsf{Equicontinuity}^{\uparrow}$ *only as the displayed boundary*

$$K, F, \epsilon, \rho, M, T, R, P, N.$$

This boundary supplies compact-source admission, finite continuous-family admission, shared tolerance-radius handoff, modulus-consumer readiness, transport, replay, provenance, and local naming. It does not supply a selected subsequence, a selected limiting map, host continuity of the limit, selected nets, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityArzelaAscoliBoundaryReadiness`

Proof. Read the root family ledger and modulus-consumer root lock. Together they make K, F, ϵ, ρ, M the whole analytic boundary exposed by the packet.

Use compact family-net readiness. Any later compactness-facing consumer receives only the routed family-net boundary and not a selected net or subsequence.

Finally replay transport, provenance, and nonescape through T, R, P, N. Since no coordinate carries a limiting map, host continuity proof, selected subsequence, selected net, or ambient real-completeness proof, an Arzela–Ascoli-facing consumer can cite only the displayed boundary. □

Theorem 4076.38 (Equicontinuity compact-family total-bounded consumer)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *whose downstream read also displays the* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *boundary over the same compact source, the packet exports exactly the finite total-bounded consumer package needed by Arzela–Ascoli-style chapters:*

$$K, F, \epsilon, \rho, M \longmapsto T, R, P, N.$$

The package ties the $\mathsf{CompactMetric}^{\uparrow}$ *source row, finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, finite-net equicontinuity boundary,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, and* $\mathsf{Real}^{\uparrow}$ *seal to the same* BHist, hsame, Cont, Pkg, *and* NameCert *route. It refuses selected subsequences, open-cover compactness, choice, selected limiting maps, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactFamilyTotalBoundedConsumer`

Proof. Use compact family-net readiness and Arzela–Ascoli boundary readiness. They make the compact source K, finite family F, tolerance ϵ, shared radius ρ, and modulus handoff M the whole analytic boundary visible to downstream consumers.

Apply shared-modulus handoff preservation and finite-net readiness. These force the $\mathsf{UniformModulus}^{\uparrow}$ and $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ reads to preserve the same shared-radius route rather than replacing it by memberwise radii or an outside finite-net selector.

Finally use transport replay provenance and nonescape. The rows T, R, P, N carry the same finite route through BHist, hsame, Cont, Pkg, and NameCert, while the carrier has no coordinates for selected subsequences, open-cover compactness, choice, limiting maps, or ambient real completeness. □

Theorem 4076.39 (Equicontinuity compact-source root row)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the compact source row* K *is the first root row that finite-net and Arzela–Ascoli consumers must read. The rows* $F, \epsilon, \rho, M, T, R, P, N$ *are available only as rows over that same* $\mathsf{CompactMetric}^{\uparrow}$ *source, and no downstream net, subsequence, or convergence consumer can bypass* K *by reading the shared-radius or finite-net boundary directly.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockCompactSourceRow`

Proof. Project the carrier of Definition 4076.1. The row K is the only compact metric source row, while $F, \epsilon, \rho, M, T, R, P, N$ are displayed after it.

Use compact-source root admission and compact-metric route, Definition 4076.9 and Definition 4076.3. They force the family ledger, tolerance row, shared radius, and modulus handoff to remain attached to K.

Finally apply finite-net readiness and Arzela–Ascoli boundary readiness. Both consumers must read the displayed compact-source route before any downstream finite net, subsequence, or convergence object is available. □

Theorem 4076.40 (Equicontinuity shared-radius root handoff)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared radius row* ρ *is a root handoff between the compact-source family rows* K, F, ϵ *and the finite-net or uniform-modulus consumer row* M*. The transport, replay, provenance, and naming rows* T, R, P, N *preserve this same* ρ*; they do not create memberwise radii, a detached pointwise modulus, or a selected finite-net radius.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockSharedRadiusHandoff`

Proof. Read K, F, ϵ, ρ, M from the carrier. The single row ρ appears after the tolerance request and before the modulus handoff.

Apply shared-radius stability and shared-modulus handoff preservation, Definition 4076.8 and Definition 4076.33. These results keep the same ρ visible through every $\mathsf{UniformModulus}^{\uparrow}$ or $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ read.

The structural rows T, R, P, N preserve the displayed route only. Nonescape then excludes memberwise radii, pointwise-only modulus data, and selected finite-net radius rows. □

Theorem 4076.41 (Equicontinuity finite-net root boundary)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the finite-net boundary is the ordered read of* K, F, ϵ, ρ, M *followed by* T, R, P, N*. A* $\mathsf{FiniteNetEquicontinuity}^{\uparrow}$ *or Arzela–Ascoli consumer may start only after the compact source and shared radius have been exposed, and the consumer must obtain any actual*

finite net, subsequence, limiting map, or convergence certificate from a downstream packet rather than from $\mathsf{Equicontinuity}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootUnblockFiniteNetBoundary`

Proof. Project the displayed rows. The carrier contains the compact source K, finite family F, tolerance ϵ, shared radius ρ, modulus handoff M, and the support rows T, R, P, N.

Use finite-net readiness and compact family-net readiness, Definition 4076.16 and Definition 4076.36. They place every finite-net-facing read after the compact-source and shared-radius route.

Finally apply the finite-net consumer boundary and nonescape results, Definition 4076.11 and Definition 4076.4. The downstream consumer may cite the boundary, but selected nets, subsequences, limiting maps, and convergence certificates are not rows of the $\mathsf{Equicontinuity}^{\uparrow}$ carrier. □

Theorem 4076.42 (Equicontinuity carrier root obligation)**.** *For every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, root admission requires the finite* BHist *rows for a family of* $\mathsf{ContinuousMap}^{\uparrow}$ *maps over the displayed source row* K*. The source row may be read as* $\mathsf{CompactMetric}^{\uparrow}$ *when finite net consumers are present, or as the underlying* $\mathsf{Met}^{\uparrow}$ *row for metric distance comparisons, but the family row* F *is admitted only as a finite continuous-map ledger over that same source. No host open-cover compactness, selected subcover, quotient real equality, or choice principle is a carrier row.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationCarrier`

Proof. Project the carrier and the compact-source root row. The source coordinate K is displayed before the family row F, and the chapter reads all finite-net and Arzela–Ascoli-facing material through that source.

Use the $\mathsf{ContinuousMap}^{\uparrow}$ family surface over the same source. The finite ledger F is a BEDC row over K; it is not a host-indexed family and it does not carry a selected subcover or open-cover compactness proof.

Finally replay the metric and real readbacks through T, R, P, N. The $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows are visible only as routed support for the displayed family, so quotient real equality and choice cannot enter the carrier admission step. □

Theorem 4076.43 (Equicontinuity finite-net locality root obligation)**.** *Every finite-net read exported by an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *routes through the displayed* $\mathsf{Met}^{\uparrow}$ *distance row, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, and the* $\mathsf{Real}^{\uparrow}$ *tolerance rows already sealed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *request ledger. A finite-net consumer may cite this local route only after* K, F, ϵ, ρ, M *have been displayed; it cannot replace the route by host open-cover compactness, a selected subcover, quotient real equality, or choice.*

Proof. Read the compact-source and shared-radius root handoff first. They place the metric source, family ledger, tolerance row, shared radius, and modulus handoff in a fixed order before any finite-net consumer can start.

Next use finite-net readiness. The finite-net surface is local to the displayed $\mathsf{CompactMetric}^{\uparrow}$ row and its underlying $\mathsf{Met}^{\uparrow}$ comparisons; the $\mathsf{Real}^{\uparrow}$ tolerance row is read through the same regular rational and stream-name ledger.

The remaining rows provide transport, replay, provenance, and local naming. Since those rows preserve only the displayed route, no open-cover compactness argument, selected subcover, quotient real equality, or choice principle can replace the finite-net locality obligation. □

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.RootObligationFiniteNet.EquicontinuityRootObligationFiniteNet`

Theorem 4076.44 (Equicontinuity uniform-modulus root obligation)**.** *The exported* $\mathsf{UniformModulus}^{\uparrow}$ *row* M *of an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$ *is available only after the equicontinuity family row* F *and the finite-net ledger over* K *have been displayed. The handoff consumes the same* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *rows as the preceding locality route, and it refuses host open-cover compactness, selected subcovers, quotient real equality, and choice.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationUniformModulus`

Proof. Begin with the family row and finite-net locality obligation. They expose the source K, family ledger F, tolerance ϵ, and shared radius ρ before any uniform-modulus consumer may read M.

Use shared-modulus handoff preservation. The $\mathsf{UniformModulus}^{\uparrow}$ row is a handoff from the displayed equicontinuity family and finite-net ledger, not a free-standing modulus selected outside the packet.

Finally apply transport replay and nonescape. The rows T, R, P, N preserve the same $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ route. Therefore the uniform-modulus handoff cannot be supplied by host compactness, a selected subcover, quotient real equality, or choice. □

Theorem 4076.45 (Equicontinuity source-family root obligation)**.** *Every accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *read exposes the* $\mathsf{CompactMetric}^{\uparrow}$ *source row* K *and the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row* F *before the tolerance* ϵ*, shared radius* ρ*, or modulus handoff* M *can be consumed. The read is therefore rooted in the displayed source-family packet of Definition 4076.1, not in an ambient compact space or an unlisted family selector.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationSourceFamily`

Proof. Read the carrier of Definition 4076.1. Its first two rows are the compact metric source K and the finite continuous-family ledger F, and Definition 4076.2 makes those rows part of the local NameCert surface.

Next use the compact metric route of Definition 4076.3. It sends every later tolerance, radius, and modulus read through the displayed K, F source-family route.

Finally apply Definition 4076.4. A consumer that asks for ϵ, ρ, M before the source and family rows has left the accepted packet rather than extracting a hidden compact-source or family coordinate. □

Theorem 4076.46 (Equicontinuity shared-radius root obligation)**.** *The radius row* ρ *of an accepted* $\mathsf{Equicontinuity}^{\uparrow}$ *carrier is a single shared radius row transported by the displayed* hsame *and* Cont *routes. It is not a memberwise selector indexed by the family row* F*, and it cannot be replaced by a choice schedule over individual maps.*

Proof. Begin with the carrier and NameCert obligations of Definition 4076.1 and Definition 4076.2. They display one row ρ between the tolerance row and the modulus handoff row.

Use the compact metric route of Definition 4076.3. The row ρ is transported through the same hsame and Cont replay rows that preserve K, F, ϵ, M.

Apply nonescape by Definition 4076.4. Since the accepted packet has no coordinate for a family-indexed selector, a memberwise radius choice cannot be recovered from the shared-radius row. □

Theorem 4076.47 (Equicontinuity downstream nonescape root obligation)**.** *The* Equicontinuity$^{\uparrow}$ *root surface contains no finite-net selector, no subsequence selector, no Arzela–Ascoli convergence row, no host-continuity oracle, and no ambient* Real$^{\uparrow}$ *completeness coordinate. Those rows must be supplied by downstream chapters before a consumer may read compact-uniform consequences from an accepted equicontinuity packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationNonescape`

Proof. Project the displayed rows of Definition 4076.1. The local surface records $K, F, \epsilon, \rho, M, T, R, P, N$ and the NameCert obligations of Definition 4076.2; it does not add selector, subsequence, convergence, host-continuity, or completeness rows.

Next use the compact metric route of Definition 4076.3. It keeps compact-source information visible as a finite route, while refusing to turn that route into open-cover compactness, selected finite nets, or selected subsequences.

Finally apply Definition 4076.4. Any Arzela–Ascoli, host-continuity, or real-completeness read must therefore enter through a downstream certificate rather than through the root Equicontinuity$^{\uparrow}$ packet. □

Theorem 4076.48 (Equicontinuity shared-radius family nonescape)**.** *For every accepted* Equicontinuity$^{\uparrow}$ *carrier* $E = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the shared radius* ρ *is read over the same* CompactMetric$^{\uparrow}$ *source row* K *and* ContinuousMap$^{\uparrow}$ *family row* F *through the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. A consumer cannot split* ρ *into memberwise radii, detach it from the* UniformModulus$^{\uparrow}$ *handoff* M*, or read it outside the finite family packet.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuitySharedRadiusFamilyNonescape`

Proof. Project the carrier and use the shared-radius root obligation. The rows K, F, ϵ, ρ, M occur in one displayed packet, so the radius row is already tied to the compact source and continuous-family ledger.

Apply shared-modulus handoff preservation. It makes the UniformModulus$^{\uparrow}$ consumer preserve the same ϵ, ρ, M rows rather than replacing them by a memberwise or selected finite-net radius.

Finally use transport replay provenance and nonescape. The rows T, R, P, N only preserve the displayed BHist, hsame, Cont, Pkg, and NameCert route, so no extra family selector can escape the shared-radius packet. □

Theorem 4076.49 (Equicontinuity compact-net modulus readiness)**.** *Every* FiniteNetEquicontinuity$^{\uparrow}$ *consumer of an accepted* Equicontinuity$^{\uparrow}$ *packet must first read the* CompactMetric$^{\uparrow}$ *source* K*, the finite* ContinuousMap$^{\uparrow}$ *family ledger* F*, the shared tolerance-radius rows* ϵ, ρ*, and the* UniformModulus$^{\uparrow}$ *handoff* M *through the same* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *surface. The finite-net handoff is readiness data only; it does not choose a net or assert open-cover compactness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityCompactNetModulusReadiness`

Proof. Begin with finite-net readiness and compact family-net readiness. Those results place any finite-net-facing read after the compact-source and continuous-family rows and before any downstream selector can be consumed.

Use the uniform-modulus root obligation to keep M attached to ϵ, ρ over the same K, F route. The modulus handoff therefore arrives as part of the finite packet, not as a second compactness principle.

The structural rows T, R, P, N then replay only the displayed route. Nonescape excludes selected nets, open-cover compactness, host continuity, and ambient $\mathsf{Real}^{\uparrow}$ completeness from the readiness surface. □

Theorem 4076.50 (Equicontinuity Arzela–Ascoli consumer route)**.** *An Arzela–Ascoli-facing consumer may cite* $\mathsf{Equicontinuity}^{\uparrow}$ *only as the routed packet* $K, F, \epsilon, \rho, M, T, R, P, N$*:* $\mathsf{CompactMetric}^{\uparrow}$ *source data, finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family data, shared* $\mathsf{UniformModulus}^{\uparrow}$ *readiness, finite-net handoff, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The consumer must supply its own subsequence, limiting map, convergence, and Real-facing certificate rows.*

Lean4 ✓ `BEDC.Derived.EquicontinuityUp.EquicontinuityArzelaAscoliConsumerRoute`

Proof. Use the Arzela–Ascoli boundary readiness theorem. It presents $\mathsf{Equicontinuity}^{\uparrow}$ as a boundary over compact-source, continuous-family, shared-radius, and modulus-handoff rows.

Next compose compact-net modulus readiness with the finite-net root boundary. This gives the downstream consumer a finite-net-ready route, but not the selected net, subsequence, limiting map, or convergence certificate.

Finally apply downstream nonescape. Since the carrier has no ambient completeness or host-continuity coordinate, every Arzela–Ascoli conclusion must be supplied by a later packet over the same displayed structural rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Equicontinuity}^{\uparrow}$ packet is generated from a compact metric source row, a finite continuous-family ledger, shared tolerance and radius rows, a $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ or $\mathsf{UniformModulus}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 4076.1, Definition 4076.2, Definition 4076.3, Definition 4076.4, Definition 4076.8, Definition 4076.15, Definition 4076.16, and Definition 4076.20 export the public finite-net and Arzela–Ascoli-facing handoff through the checked root-obligation target over compact-source, continuous-family, tolerance, shared-radius, modulus-handoff, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Equicontinuity}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.EquicontinuityUp.EquicontinuityRootObligationSourceFamily`

Bridge status: none

Not claimed: This chapter does not close compactness, selected finite nets, selected subsequences, Arzela–Ascoli convergence, host-level continuity, pointwise-only radius choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{Equicontinuity}^{\uparrow}$.

Upgrade path: The remaining work is the standard bridge check for the same public finite-net and Arzela–Ascoli-facing handoff, not additional paper closure.

Chapter 4077

A Concrete Naming Certificate for $\mathsf{ChoquetGame}^{\uparrow}$

Following Choquet and Oxtoby on topological games and Baire category, the $\mathsf{ChoquetGame}^{\uparrow}$ packet records the finite nested-open game route used by complete-metric Baire arguments. It sits over the Baire prefix of chapter 334, the dense-open and category surface of chapter 1542, the complete-metric handoff of chapter 128, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the standard Choquet-game interface as BEDC NameCert data without importing a chosen winning strategy, a host intersection point, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4077.1 (Choquet-game carrier)**.** *A* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, M, L, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row naming the inspected basic-open source, finite prefix probes, and legal initial open row.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *metric-ball row, including displayed centers, positive rational radii, inclusion checks, and shrinking-radius obligations for successor moves.*
3. *L is the nested legal-move ledger, recording each finite play prefix, the predecessor ball, the successor ball, the player tag, and the proof that the successor remains inside the predecessor.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the finite play prefixes and ball observations visible to the consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius budgets, and tolerance comparisons used by the nested ball thread.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Baire prefix, complete-metric ball data, legal-move ledger, stream schedule, and regular rational readback have been displayed.*
7. *T records componentwise* hsame *transport for Baire prefixes, complete-metric ball rows, legal-move ledgers, stream schedules, regular rational readback, Real seals, replay data, provenance, and local naming.*

8. *C records displayed* Cont *replay from a game request to the Baire prefix, complete-metric ball route, nested legal-move ledger, finite schedule, rational readback, Real seal, transport, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{ChoquetGame}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global winning strategy, a selected limit point, countable choice, quotient equality of streams, ambient completed-real principles, or a hidden dense-open oracle.

Lean4 ✓ `BEDC.Derived.ChoquetGameUp.ChoquetGameTasteGate single carrier alignment`

Theorem 4077.2 (Choquet-game NameCert obligations)**.** *For every accepted* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier* $G = (B, M, L, S, R, E, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Baire prefix, complete-metric ball, legal-move ledger, stream schedule, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *Baire-prefix coverage, requiring every play prefix to begin with the* $\mathsf{BaireSpace}^{\uparrow}$ *row B;*
3. *metric-ball legality, requiring M to supply displayed centers, radii, inclusions, and shrinking obligations before L records a successor move;*
4. *nested-play exactness, requiring L to store only finite legal play prefixes and no unbounded strategy object;*
5. *schedule and readback discipline, requiring S, R, E to be read in that order before any terminal Real-facing consumer receives the packet;*
6. *structural preservation, requiring T, C, P, N to transport, replay, source, and name only the displayed game route;*
7. *ledger non-escape, excluding global winning strategies, selected limit points, countable choice, quotient stream equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden dense-open oracles.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4077.1. The analytic rows are B, M, L, S, R, E, and the structural rows are T, C, P, N.

Baire-prefix coverage fixes the source of every finite play. Each accepted game request begins at the displayed $\mathsf{BaireSpace}^{\uparrow}$ prefix row and therefore cannot read an external open-set universe.

Metric-ball legality is confined to M and L. The complete-metric row supplies centers, radii, inclusion checks, and shrinking obligations, while the legal-move ledger records only the finite nested prefixes admitted by those checks.

The terminal handoff is scheduled. A consumer reads the finite play schedule through S, the rational tolerances through R, and the Real-facing seal through E only after the earlier rows have appeared.

Finally T, C, P, N preserve, replay, source, and name the same displayed packet. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4077.3 (Choquet-game Baire-category handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{ChoquetGame}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{ChoquetGame}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route supplies the Baire-category consumer surface of chapter 1542 by exposing nested legal move prefixes and complete-metric ball obligations, not by selecting a host intersection point or importing a completed-real principle.

Proof. Start with an accepted carrier. The Baire prefix B names the basic-open source, and the complete-metric row M names the ball data through which legal moves are checked.

The game content is finite. The ledger L records the displayed nested play prefixes and the successor-in-predecessor evidence needed by a Baire-category handoff, while S exposes the finite schedule consumed at the current stage.

Regular rational readback and Real sealing are downstream. The row R records the rational center and radius tolerances, and E seals the terminal read only after B, M, L, S, R have been displayed.

Transport, replay, provenance, and local naming in T, C, P, N preserve that route componentwise. Thus the handoff supplies Baire-category arguments with the visible nested-ball interface while exporting no global strategy, selected point, countable-choice enumeration, quotient stream equality, or ambient completed-real principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ChoquetGame}^{\uparrow}$ packet is generated from a BaireSpace prefix, CompleteMetric metric-ball rows, a nested legal-move ledger, a StreamName schedule, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$, seed)

Scope closed: Definition 4077.1, Definition 4077.2, and Definition 4077.3 seed the Choquet-game route over BaireSpaceUp, BaireCategoryUp, CompleteMetricUp, StreamNameUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ChoquetGame}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim global winning strategies, selected limit points, countable choice, quotient stream equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Baire-prefix coverage, complete-metric ball legality, nested-play exactness, StreamName scheduling, RegSeqRat readback, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetGameUp.

Chapter 4078

A Concrete Naming Certificate for LocatedCoveringNumber$^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley-style metric-cover language, the LocatedCoveringNumber packet records a covering-number request whose centers, tolerance, and budget are all displayed before any compactness or dimension consumer can use it. It sits after the metric covering-number surface of chapter 2187, the finite epsilon-net carrier of chapter 1767, the compact metric route of chapter 129, and the doubling-metric finite-growth packet of chapter 2035. The packet names a finite located cover certificate; it is not an open-cover compactness theorem, selected subcover principle, quotient metric equality, or ambient choice of centers.

Definition 4078.1 (Located covering number carrier)**.** *A* LocatedCoveringNumber$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, Q, T, E, B, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, including the carrier classifier, distance requests, and ball-membership rows used by the cover.*
2. *Q is the located request row, naming the displayed subset or inspection window together with the cover request whose membership must be witnessed.*
3. *T is the tolerance ledger, recording the positive ε row and its rational or dyadic comparison data.*
4. *E is the finite ε-net center row, listing the centers and their point-name evidence through chapter 1767.*
5. *B is the covering-number budget row, recording the displayed natural-number bound and the count of center rows used by the certificate.*
6. *W is the location witness row, assigning each inspected request point to one displayed center with a distance comparison below the tolerance ledger.*
7. *H records componentwise* hsame *transport for the metric source, request, tolerance, center list, budget, witness, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a located cover request through the tolerance ledger, finite center list, budget row, and witness row.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedCoveringNumber}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, Q, T, E, B, W, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, hidden dense enumerations, quotient metric equality, host compactness, or a choice-selected center family.

Lean4 (definition) `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberCarrier`
Lean4 ✓ `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberTasteGate single carrier alignment`

Theorem 4078.2 (Located covering number NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCoveringNumber}^{\uparrow}$ *carrier $L = (M, Q, T, E, B, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$ *surface consists exactly of carrier admission, located-center visibility, tolerance exactness, budget exactness, finite-net handoff, location-witness replay, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. Every covering-number consumer must read these rows before using the displayed budget, and no consumer may read a stronger compactness, selected-subcover, or choice-of-centers object from the packet.*

Lean4 ✓ `BEDC.Derived.LocatedCoveringNumberUp.LocatedCoveringNumberNameCertObligations`

Proof. Project the carrier of Definition 4078.1. Carrier admission exposes the metric source, located request, tolerance ledger, finite center list, budget row, location witness, transport, replay, provenance, and local name.

The located-center and tolerance obligations are read through E, T, W. A center can serve a request only when the witness row displays the center and the corresponding distance comparison against the same tolerance ledger.

Budget exactness is read through B and E. The count in B is attached to the displayed finite center row, so the covering number cannot be justified by an unlisted center or a selected subcover outside the packet.

Finally use H, C, P, N. Transport, replay, provenance, and local naming preserve the same finite-net handoff to the metric covering-number, compact-metric, and doubling-metric consumers. Coordinate exhaustion blocks arbitrary covers, host compactness, quotient metric equality, and choice-selected center families. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCoveringNumber}^{\uparrow}$ consumer displays a metric source row, located request row, tolerance ledger, finite epsilon-net center row, covering-number budget row, location witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a covering-number budget is public.

Independence witness: metric-covering-number, finite-epsilon-net, compactmetric, doubling-metric: $\mathsf{LocatedCoveringNumber}^{\uparrow}$ is not bijective to any listed sibling because it binds the request locator, tolerance ledger, center list, budget row, and witness assignment into one finite covering-number certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCoveringNumber}^{\uparrow}$ packet is generated from a metric source row, located request row, tolerance ledger, finite epsilon-net center row, covering-number budget row, location witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCoveringNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4078.1 and Definition 4078.2 seed the located finite-cover budget route over MetricCoveringNumber, FiniteEpsilonNet, CompactMetric, DoublingMetric, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCoveringNumber}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, arbitrary open-cover compactness, selected subcovers, quotient metric equality, hidden dense enumerations, or choice-selected center families.

Upgrade path: Obligation closure requires checked carrier admission, located-center visibility, tolerance exactness, budget exactness, finite-net handoff, witness replay, transport, provenance, and local naming for BEDC.Derived.LocatedCoveringNumberUp.

Chapter 4079

A Concrete Naming Certificate for $\mathsf{SupportNerveRealization}^{\uparrow}$

Following the finite combinatorics of abstract simplicial complexes, the $\mathsf{SupportNerveRealization}^{\uparrow}$ packet records the support-family-to-boundary-subcomplex theorem of Definition 9462.1659 as concrete NameCert data. It sits between the finite support census of chapter 4048, the lifted-Betti consumer of chapter 4035, and the simplicial-complex carrier of chapter 238. The packet names support rows, complement-face rows, boundary-subcomplex rows, facet witnesses, census handoff, transport, replay, provenance, and local naming; it does not import an ambient topological realization or unrestricted homology computation.

Definition 4079.1 (Support nerve realization carrier)**.** *A* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S, F, K, A, B, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the support-family row, listing the inspected nonempty support symbols.*
2. *F is the complement-face row, sending each support to its face in the tetrahedron boundary.*
3. *K is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *boundary-subcomplex row generated by the downward closure of the displayed complement faces.*
4. *A is the realization admission row, recording that every displayed support family produces a subcomplex of $\partial\Delta^3$.*
5. *B is the reverse realization row, recording the finite facet-to-support construction for each displayed boundary subcomplex.*
6. *L is the census handoff row, exposing the realized subcomplex to* $\mathsf{SupportNerveCensus}^{\uparrow}$ *and* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *consumers.*
7. *H records componentwise* hsame *transport for the support, face, subcomplex, admission, reverse, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from support-family inspection through complement faces, downward closure, reverse facet witnesses, and census handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{SupportNerveRealization}^{\uparrow}$, $\mathsf{SupportNerveCensus}^{\uparrow}$, $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, F, K, A, B, L, H, C, P, N$. It has no coordinate for host geometric realization, ambient equality of complexes, unrestricted chain homology, coefficient-field choice, or an infinite support universe.

Lean4 ✓ `BEDC.Derived.SupportNerveRealizationUp.SupportNerveRealizationTasteGate single carrier alignment`

Theorem 4079.2 (Support nerve realization NameCert obligations)**.** *For every accepted* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier $R = (S, F, K, A, B, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$ *surface consists exactly of support-family exposure, complement-face exposure, finite* $\mathsf{SimplicialComplex}^{\uparrow}$ *boundary-subcomplex admission, reverse facet-to-support realization, census handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4079.1. The rows S, F, K display the support family, its complement faces, and the resulting downward-closed boundary subcomplex.

The two realization directions are stored in A and B. The admission row verifies that displayed supports produce a subcomplex of $\partial\Delta^3$, while the reverse row records the finite facet-complement supports that realize a displayed subcomplex.

Finally use L, H, C, P, N. The handoff row sends the same finite subcomplex to census and lifted-Betti consumers, and the structural rows transport, replay, source, and name the packet. Since no ambient geometric realization, unrestricted homology computation, coefficient-field choice, or host equality coordinate is present, the listed rows exhaust the local obligations. □

Theorem 4079.3 (Support nerve realization census handoff)**.** *Every* $\mathsf{SupportNerveCensus}^{\uparrow}$ *or* $\mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}$ *consumer of an accepted* $\mathsf{SupportNerveRealization}^{\uparrow}$ *carrier must first read the finite route*

$$\mathsf{SupportNerveRealization}^{\uparrow} \longmapsto \mathsf{SimplicialComplex}^{\uparrow} \longmapsto \mathsf{SupportNerveCensus}^{\uparrow} \longmapsto \mathsf{SupportNerveCensusLiftedBetti}^{\uparrow}.$$

The route exposes support rows, complement-face rows, boundary-subcomplex rows, reverse facet witnesses, transport, replay, provenance, and local naming before any census or lifted-Betti entry is exported.

Proof. Start with S, F, K. These rows build the finite support nerve as a displayed subcomplex of the tetrahedron boundary.

Use the realization obligations A and B. They identify both directions of the support-family-to-subcomplex correspondence without leaving the finite face inventory.

Repack through L, H, C, P, N. The census and lifted-Betti consumers receive the same displayed subcomplex and its support witnesses, while the structural rows preserve the packet. Coordinate exhaustion prevents an ambient realization, unrestricted homology computation, or coefficient-field selector from entering the handoff. □

Falsifiable prediction: Every accepted $\mathsf{SupportNerveRealization}^{\uparrow}$ consumer displays support-family rows, complement-face rows, a finite $\mathsf{SimplicialComplex}^{\uparrow}$ boundary-subcomplex row, reverse facet witnesses, census handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before a census or lifted-Betti read is public.

Independence witness: support-nerve-census, support-nerve-census-lifted-betti, simplicialcomplex: $\mathsf{SupportNerveRealization}^{\uparrow}$ is not bijective to any listed sibling because it binds the forward

support-to-subcomplex construction and reverse facet-support realization before the census handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{SupportNerveRealization}^{\uparrow}$ packet is generated from support-family rows, complement-face rows, finite SimplicialComplex boundary-subcomplex rows, reverse facet-to-support witnesses, census handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4079.1, Definition 4079.2, and Definition 4079.3 seed the finite support-family-to-boundary-subcomplex realization packet and its census handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SupportNerveRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host geometric realization, ambient equality of complexes, unrestricted homology, coefficient-field choice, or infinite support universes.

Upgrade path: Obligation closure requires checked support exposure, complement-face exposure, boundary-subcomplex admission, reverse facet-support realization, census handoff, transport, replay, provenance, and local naming for BEDC.Derived.SupportNerveRe

$\mathsf{CauchyDiagonalSelection}^{\uparrow}$ consumes the non-quotient classifier route of chapter 1820 before selecting a displayed diagonal regular-Cauchy handoff.

Chapter 4080

A Concrete Naming Certificate for $\mathsf{CauchyDiagonalSelection}^{\uparrow}$

Following Bishop and Bridges on constructive diagonal arguments for Cauchy families, the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet records the finite row that chooses a diagonal observation window from displayed regular-Cauchy data before any Real or Completion consumer reads a seal. It sits after the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the Bishop Cauchy-modulus extraction route of chapter 4071. The diagonal-selection route is consumed by chapter 1809 as a countable dense handoff for completion-facing density reads. The same diagonal window is later consumed by chapter 7328, where the selected index is normalized into a dyadic schedule before RegSeqRat and Real consumers read the endpoint. The packet supplies a named L10 diagonal-selection handoff for Real and Completion routes; it does not assert a host sequence selector, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4080.1 (Cauchy diagonal selection carrier)**.** *A* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, M, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source schedule, exposing the finite family of observation windows available to the diagonal request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the displayed stream windows.*
3. *M is the Cauchy-modulus extraction row, read through* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *as the finite threshold discipline for the request.*
4. *W is the diagonal-window row, selecting the single displayed window whose source index and precision index have both passed through S, R, M.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the comparison between the requested tolerance, the extracted threshold, and the chosen diagonal window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular readback, modulus row, diagonal window, and dyadic ledger have been displayed.*
7. *H records componentwise* hsame *transport for the stream schedule, regular readback, modulus extraction, diagonal window, tolerance ledger, Real seal, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the tolerance request through S, R, M, W, T, E and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BishopCauchyModulusExtrac $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, M, W, T, E, H, C, P, N$. It has no coordinate for an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDiagonalSelectionUp.CauchyDiagonalSelectionTasteGate single carrier alignment`

Theorem 4080.2 (Cauchy diagonal selection Real uniform-structure anchor)**.** *Every* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer that reads the diagonal window as Real completion data first passes through the* $\mathsf{RealUniformStructure}^{\uparrow}$ *surface of chapter 1624. For an accepted carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the visible route is*

$\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, hsame, Cont, Pkg, NameCert.

Thus the diagonal consumer reads finite uniform-entourage, stream-window, regular-rational, dyadic-tolerance, transport, replay, provenance, and naming rows before any completion-facing Real seal is consumed. It cannot replace those rows by a detached selector, quotient stream equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4080.1. The rows S, R, T, E, H, C, P, N already expose the finite stream windows, regular-rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local naming data.

Read the Real completion request through chapter 1624. The uniform-structure packet supplies the displayed Real entourage discipline consumed by the diagonal window before the downstream Real-facing seal is inspected.

The structural rows H, C, P, N then preserve that same route. Since the carrier has no coordinate for a detached selector, quotient stream equality, host real equality, countable choice, or ambient Real completeness, the Real-uniform-structure anchor is the only completion-facing dependency exposed by this theorem. □

Theorem 4080.3 (Cauchy diagonal selection NameCert obligations)**.** *For every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *diagonal-window selection, requiring the selected row W to display both the stream source window from S and the regular readback from R before the diagonal observation is admitted;*

2. *regular-Cauchy preservation, requiring R and M to keep the readback inside the same regular Cauchy family under the extracted threshold;*

3. *dyadic tolerance control, requiring T to compare the requested tolerance with the modulus threshold and the selected diagonal window;*

4. *Real-seal handoff, requiring E to be downstream of S, R, M, W, T and to export only the displayed diagonal observation;*

5. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed packet, with no external selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host equality, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 4080.1. The row W is meaningful only after the stream schedule and regular readback rows have been displayed, so the diagonal observation has no source outside S and R.

The modulus row M supplies the finite threshold discipline consumed by the regular-Cauchy read. Thus the readback remains inside the same displayed family rather than becoming a selected subsequence or a host-level limit.

The tolerance ledger T checks the requested dyadic scale against the extracted threshold and the chosen diagonal window. Only after that finite comparison can the Real-facing row E be consumed.

Finally, H, C, P, N preserve and name the same coordinates. Since none of the refused objects appears in the carrier, the five displayed obligations exhaust the local certificate surface. □

Theorem 4080.4 (Cauchy diagonal selection Real-seal handoff)**.** *For every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier $D = (S, R, M, W, T, E, H, C, P, N)$, the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is reachable only after the* $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{BishopCauchyModulusExtraction}^{\uparrow}$ *row M, diagonal window W, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T are displayed. The handoff exports no external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier in Definition 4080.1. The Real-facing row E appears only downstream of the displayed schedule, regular readback, modulus extraction, diagonal window, and dyadic tolerance ledger.

Use the five obligations in Definition 4080.3. They require diagonal-window selection through S, R, W, regular-Cauchy preservation through R, M, dyadic tolerance control through T, and Real-seal handoff through E after those rows are visible.

The structural rows H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient completeness, none can be exported at the seal. □

Theorem 4080.5 (Cauchy diagonal selection RegSeqRat threshold stability)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. Suppose another accepted carrier*

$$D' = (S, R, M', W, T, E, H, C, P, N)$$

has the same $\mathsf{StreamName}^{\uparrow}$ *schedule S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, diagonal window W, tolerance ledger T, Real seal E, transport row H, replay row C, provenance row P, and local naming row N, and suppose the displayed threshold row M' is a stronger Cauchy-modulus threshold compatible with the same S, R, W, T rows. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal of D' remain on the same finite route as those of D:*

$$S, R, M', W, T \longmapsto E \longmapsto H, C, P, N.$$

In particular, strengthening the modulus threshold does not introduce an external diagonal selector, quotient stream equality, hidden subsequence choice, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project both carriers through Definition 4080.1. The two packets differ only in the modulus row, while $S, R, W, T, E, H, C, P, N$ are displayed as the same rows in both packets.

Apply the NameCert obligations of Definition 4080.3. The regular-Cauchy preservation obligation reads the modulus row only as the finite threshold discipline attached to R, and the dyadic tolerance obligation reads T against the chosen diagonal window W.

Since M' is a displayed stronger threshold compatible with the same S, R, W, T rows, the replay through C transports the regular readback along the same finite request route. No new selector or quotient comparison is needed to read the strengthened threshold.

Finally apply the Real-seal handoff of Definition 4080.4. The terminal seal E can still be consumed only after S, R, M', W, T are displayed, and the structural rows H, C, P, N keep the route inside the listed coordinates. □

Theorem 4080.6 (Cauchy diagonal selection finite-window induction)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. Induction over the displayed finite window schedule preserves the route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{BishopCauchyModulusExtraction}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

In particular, strengthening or extending any finite prefix keeps the selected diagonal read inside $S, R, M, W, T, E, H, C, P, N$ and cannot introduce an external selector, quotient stream equality, countable choice, trichotomy, host equality, or ambient completeness.

Proof. The base window is the carrier projection of Definition 4080.1. The rows S, R, M, W, T, E already display the StreamName schedule, regular readback, modulus threshold, diagonal window, dyadic tolerance ledger, and Real seal in the stated order.

For the induction step, extend a displayed finite prefix by one accepted window. The NameCert obligations of Definition 4080.3 force the new window to be read through S and R, then checked by the modulus row M and tolerance ledger T before the Real-facing row E is consumed.

If the extension strengthens the threshold instead of adding a new observation, apply RegSeqRat threshold stability, Definition 4080.5. The strengthened row remains compatible with the same S, R, W, T route and does not add a separate selector.

Finally use the Real-seal handoff, Definition 4080.4. Since H, C, P, N transport, replay, source, and name only the displayed packet, the induction cannot acquire quotient stream equality, countable choice, trichotomy, host equality, or ambient completeness from outside the carrier. □

Theorem 4080.7 (Cauchy diagonal selection subsequence route)**.** *Let $D = (S, R, M, W, T, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *carrier. When D is read as a subsequence corridor, every* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *seal consumer factors first through the* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *surface of chapter 5902. The diagonal window is therefore consumed only along the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{BishopCauchyModulusExtraction}^{\uparrow},\ \mathsf{CauchySubsequenceSelector}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{R}$$

with structural support carried by H, C, P, N. The route exposes no external diagonal selector, quotient stream equality, countable-choice subsequence, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 4080.1. The rows S, R, M, W, T display the source schedule, regular readback, modulus threshold, selected diagonal window, and dyadic tolerance ledger before E is available.

Apply finite-window induction, Definition 4080.6. Each accepted prefix keeps the selected window inside S, R, M, W, T, so the diagonal read has the same finite selector shape required by the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ carrier surface.

Read the selector monotonicity and Cauchy-gap rows from chapter 5902. They supply the subsequence corridor between the displayed stream windows and the regular-rational handoff, while preserving the diagonal packet's structural rows H, C, P, N.

Finally apply the Real-seal handoff, Definition 4080.4. Since the consumer reaches E only after the selector corridor, regular readback, modulus threshold, diagonal window, and tolerance ledger have been displayed, it cannot import an external diagonal selector, countable choice, quotient stream equality, host real equality, or ambient completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet is generated from StreamName source windows, RegSeqRat regular readback, a BishopCauchyModulusExtraction threshold row, a selected diagonal window, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4080.1, Definition 4080.3, Definition 4080.4, Definition 4080.5, Definition 4080.6, and Definition 4080.7 give the obligation surface over StreamName, RegSeqRat, BishopCauchyModulusExtraction, CauchySubsequenceSelector, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDiagonalSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected subsequences, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires a checked scope boundary for downstream Real-separability and regular-modulus-normalization consumers.

The diagonal selector reads $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and $\mathsf{Real}^{\uparrow}$ seal rows through the non-quotient classifier route of chapter 1820. This keeps the selected diagonal handoff attached to displayed regular-Cauchy evidence rather than to a quotient class, representative selector, ambient completeness rule, or host equality oracle. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ packet forwards its selected finite tail through the $\mathsf{CauchyFilterLimit}^{\uparrow}$ handoff of chapter 1558 before any Real-facing consumer inspects the terminal seal. This route preserves the existing $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ dependencies from the diagonal-selection carrier, so the handoff is a displayed completion-facing dependency rather than a detached selector or ambient limit principle. Every completion-facing $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ read consumed by chapter 1558 must first display the $\mathsf{StreamName}^{\uparrow}$ finite-window row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback row, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the diagonal-window selection row, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer therefore receives a lattice route through displayed rows, not a detached diagonal selector, quotient stream equality, selected host limit, countable-choice schedule, or ambient completeness principle. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route consumes the regularized Cauchy-modulus handoff of chapter 7888 before any diagonal subsequence or Real seal is read. Thus a diagonal consumer must expose the $\mathsf{CauchyModulus}^{\uparrow}$ threshold row, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal route rather than reading the diagonal as a detached selector, quotient stream, or ambient completeness coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ consumer link records the downstream read shared by diagonal-selection, regular-Cauchy extraction, and L10 completion consumers.

Theorem 4080.8 (Cauchy diagonal selection consumer link)**.** *Every* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer reaches its selected diagonal only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, and the diagonal-selection sibling routes of Definition 4080.15 and Definition 4080.16. When the consumer is compared with the regular-Cauchy diagonal-extraction*

surface of chapter 1773, the shared route is the finite handoff through the source windows, regular readback, dyadic tolerance ledger, diagonal window, terminal $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local* NameCert *row. It exports no detached selector, quotient stream equality, countable-choice schedule, selected host limit, or ambient completion principle.*

Proof. Project the carrier $D = (S, R, M, W, T, E, H, C, P, N)$ from Definition 4080.1. The rows S and R expose the finite $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ readback before the selected window W can be consumed.

Use Definition 4080.15 to fix the displayed StreamName, RegSeqRat, DyadicRatCore, and Real-seal route. Then use Definition 4080.16 to compare that route with the regular-Cauchy diagonal-extraction chapter.

The structural rows H, C, P, N preserve this handoff under transport, replay, provenance, and local naming. Since the carrier contains no coordinate for quotient stream equality, countable choice, a selected host limit, or ambient completion, every consumer that reaches the selected diagonal remains inside the displayed finite route. □

The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dossier is consumed by chapter 4202 through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows recorded in the local carrier. This cross-reference keeps the diagonal-selection window visible as the finite input to the rational interval locator, rather than as a self-contained selector detached from the surrounding real-name lattice. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dependency route also consumes the rectangular-window and cofinal-schedule surface of chapter 1689 together with the single diagonal-index route of chapter 1873 before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ readback is exposed. Thus the selected diagonal window is a finite handoff through displayed double-diagonal and diagonal-argument rows, not a detached selector or an ambient completion claim. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ L10 route consumes the source windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real seal of chapter 38 through the carrier displayed in Definition 4080.1. These dependencies are already part of the diagonal-selection packet; this child route makes them visible to the top-level dossier scan without adding a second closure ledger or theorem surface. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ forward route consumes the regular Cauchy modulus transfer of chapter 7610, the finite $\mathsf{StreamName}^{\uparrow}$ source windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any diagonal-selection read is exported. Thus the sibling lattice records the diagonal window as a displayed finite dependency route, not as an isolated selector, quotient stream, selected limit, or ambient completeness coordinate.

Theorem 4080.9 (Cauchy diagonal selection rational-interval locator route)**.** *Every* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *consumer that reads a* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal window must first display the finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal carried by the diagonal-selection packet. The route then hands the visible rational lower and upper interval data to chapter 4202; it exports no new selector, quotient stream equality, selected real limit, countable-choice schedule, or ambient completeness claim.*

Proof. Project the diagonal-selection packet through its displayed child routes. The only analytic rows available before locator consumption are the finite StreamName windows, regular rational readback rows, dyadic tolerance ledger, and Real seal.

The rational-interval locator may therefore read the selected diagonal window only after those rows are visible. At that point the lower and upper rational interval data are already finite packet

data, so the consumer receives a local locator route rather than a new diagonal selector or a completion theorem.

Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve that same route into chapter 4202. Since no carrier row contains quotient stream equality, a selected real limit, a countable-choice schedule, or ambient completeness, the route exports none of those objects. □

The CauchyDiagonalSelection$^{\uparrow}$ diagonal window selected from the StreamName$^{\uparrow}$ source schedule, RegSeqRat$^{\uparrow}$ regular readback, and DyadicRatCore$^{\uparrow}$ tolerance ledger is the finite window read by RationalIntervalLocator$^{\uparrow}$ in chapter 4202 before a Real$^{\uparrow}$-facing interval consumer can inspect rational lower and upper bounds. This route records that the locator consumes the displayed diagonal window as interval data, rather than importing a selected real limit, quotient stream equality, countable-choice schedule, or ambient completeness coordinate. The rational lower and upper endpoint exports of chapter 4202 consume the CauchyDiagonalSelection$^{\uparrow}$ finite-window route before any Real$^{\uparrow}$-facing read treats those endpoints as interval evidence. This downstream use keeps the locator's bracket rows tied to displayed StreamName, RegSeqRat, and dyadic tolerance data rather than to a selected real limit, quotient stream equality, or ambient completeness coordinate. The CauchyDiagonalSelection$^{\uparrow}$ route feeds the RealLocator$^{\uparrow}$ interval read of chapter 1669 only after the finite source windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38 have been displayed. Thus the locator-facing consumer reads the same diagonal-selection window and sibling rows before any completion-facing seal is exposed, rather than importing a detached selector, quotient stream, selected limit, or ambient completeness coordinate. The RealSeparability$^{\uparrow}$ countable-dense handoff of chapter 1809 consumes the CauchyDiagonalSelection$^{\uparrow}$ window only after the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, and Real$^{\uparrow}$ rows have been displayed. Thus the diagonal-selection packet supplies the finite diagonal route for the separability consumer without introducing a quotient stream, a selected real representative, or a choice schedule.

Theorem 4080.10 (Cauchy diagonal selection real-spread route)**.** *Every* CauchyRealSpread$^{\uparrow}$ *or* Real$^{\uparrow}$ *completion consumer that reads a* CauchyDiagonalSelection$^{\uparrow}$ *diagonal window must first pass through the StreamName-RegSeqRat route of Definition 4080.15. The handoff then feeds the real-spread completion surface of chapter 4161 and the terminal real-name seal of chapter 38 using only the displayed finite source windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows.*

Proof. Project the StreamName-RegSeqRat route already displayed in this chapter. It exposes finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, dyadic tolerance data, and the terminal Real$^{\uparrow}$ seal before a diagonal-selection consumer can read the selected window.

The CauchyRealSpread$^{\uparrow}$ handoff consumes exactly that finite completion-facing data. Its sibling chapter records a real-spread completion route, and the Real$^{\uparrow}$ chapter supplies the terminal seal; neither sibling adds a new selector to the diagonal-selection packet.

Transport, replay, provenance, and local naming preserve the same rows. Thus the route to CauchyRealSpread$^{\uparrow}$ and Real$^{\uparrow}$ is the displayed completion route, not countable choice, quotient stream equality, selected representatives, or a detached host completion. □

Theorem 4080.11 (Cauchy diagonal selection nonescape)**.** *Every completion-facing consumer generated from a* CauchyDiagonalSelection$^{\uparrow}$ *packet and the real-spread route above remains inside the displayed StreamName, RegSeqRat, dyadic-tolerance, Real-seal, transport, replay, provenance,*

and local-name rows. It cannot become countable choice, a selected representative family, quotient stream equality, a detached diagonal selector, or an ambient completeness principle.

Proof. Induct over the displayed diagonal-selection packet. The base case has only the source windows, regular readback, tolerance ledger, and Real seal already named by Definition 4080.15.

In the finite step case, the real-spread consumer may read a new diagonal window only after those rows are visible. The sibling $\mathsf{CauchyRealSpread}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ chapters receive the same finite completion-facing data and do not add a choice schedule or selected representative coordinate.

The structural rows then transport, replay, source, and name the packet. Since no row contains countable choice, quotient stream equality, selected representatives, a detached selector, or ambient completeness, none of those objects can escape from the displayed route. □

Theorem 4080.12 (Cauchy diagonal selection RealSpread dependency anchor)**.** *Any* $\mathsf{CauchyRealSpread}^{\uparrow}$ *or Real-separability consumer reading a* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal route must pass through the* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *diagonal schedule of chapter 4080, and the* $\mathsf{CauchyRealSpread}^{\uparrow}$ *dependency route of chapter 4161 before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route is a finite projection of displayed dependency rows, not a compactness principle or choice principle.*

Proof. Project the diagonal-selection carrier through Definition 4080.15. The visible rows are the finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, selected diagonal schedule, and terminal Real seal.

Read the real-spread dependency route before the Real-facing consumer. That sibling route receives the same displayed window, readback, tolerance, and schedule rows; it does not add a branch selector or a countable-choice schedule.

Finally pass to the terminal $\mathsf{Real}^{\uparrow}$ seal. Since every row in the path is already displayed by the diagonal-selection and real-spread packets, the anchor only projects a finite dependency route and does not assert new compactness, quotient equality, host real equality, or ambient completeness. □

The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ Real-completion route consumes the finite diagonal-meet surface of chapter 909 after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are visible, and before any completion-facing selector is public. This records the diagonal-selection handoff as a displayed dependency through the regular Cauchy meet route rather than as a detached selector, quotient stream, selected limit, or ambient completion coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ handoff consumes the $\mathsf{RegularCauchySandwich}^{\uparrow}$ cofinal-tail mesh of chapter 5874 before any completion-facing diagonal route is read. Thus the selected diagonal window is constrained by displayed lower-tail, upper-tail, common-modulus, and regular readback rows rather than by a detached selector, quotient stream, or ambient completeness coordinate. The $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ dossier reads its diagonal-selection window through the same L10 route used by the regular Cauchy modulus siblings: the $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ budget of chapter 4158, the $\mathsf{RegularCauchyModulusTransfer}^{\uparrow}$ handoff of chapter 7610, and the existing $\mathsf{StreamName}^{\uparrow}$-$\mathsf{RegSeqRat}^{\uparrow}$-$\mathsf{Real}^{\uparrow}$ path recorded in Definition 4080.15. Thus diagonal selection consumes finite stream windows, dyadic tolerances, regular rational readback, and terminal real seals as displayed sibling rows rather than as a detached tail selector or quotient-completion representative. The diagonal-selection packet consumes chapter 4158 as the visible uniform tail-budget surface before handing data to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers.

Theorem 4080.13 (CauchyDiagonalSelection regular-Cauchy dependency)**.** *Every accepted* CauchyDiagonalSelecti *read that exports a selected diagonal window first consumes the regular-Cauchy diagonal route of*

chapter 909, the regular rational readback of chapter 301, and the finite stream-window row of chapter 126. The exported diagonal window is therefore downstream of the RegularCauchyDiagonalMeet, RegSeqRat, and StreamName sibling surface, not a detached selector or host-side choice function.

Proof. Project the local $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route through the existing diagonal-selection carrier. The selected window is read only after the stream schedule and regular readback rows have been exposed.

The regular-Cauchy sibling route supplies the diagonal-meet dependency. Once the consumer has passed through chapter 909, the RegSeqRat row fixes the displayed rational readback and the StreamName row fixes the finite window schedule.

Thus the exported selected diagonal window is a named sibling handoff. No row in the packet supplies a detached selector, quotient stream equality, countable-choice schedule, or host completion representative. □

Theorem 4080.14 (Cauchy diagonal selection regular-Cauchy gluing forward route)**.** *Every* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that uses a diagonal-selection window from* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *must first expose the finite subsequence window, rowwise Cauchy budget, diagonal index ledger, and componentwise replay rows supplied by the diagonal-selection packet. The handoff supplies the finite window consumed by chapter 5899; it does not choose endpoints, assert completion, select a limit, or replace the gluing packet's own regular-Cauchy compatibility rows.*

Proof. Read the diagonal-selection packet through its local carrier rows. The selected diagonal window, rowwise Cauchy budget, index ledger, transport, replay, provenance, and local naming rows are the only data exported to a downstream gluing consumer.

Feed exactly those rows to the regular-Cauchy gluing route of chapter 5899. The gluing packet may use the finite subsequence window as input, but it must still display its own regular-Cauchy compatibility and gluing rows.

Since the diagonal-selection packet contains no endpoint selector, completion principle, selected limit, or ambient Real-completeness row, the forward route exports none of those claims. □

Theorem 4080.15 (Cauchy diagonal selection StreamName-RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *packet must first expose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any diagonal selector is consumed. Thus a completion-facing diagonal request reads the same displayed* S, R, T, E *route recorded by the local carrier and cannot replace it by a detached tail selector, quotient stream, selected limit, or hidden completion representative.*

Proof. Project the carrier $D = (S, R, M, W, T, E, H, C, P, N)$ from Definition 4080.1. The source schedule S supplies the finite $\mathsf{StreamName}^{\uparrow}$ windows, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ values, the tolerance row T supplies the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison, and the terminal row E supplies the $\mathsf{Real}^{\uparrow}$ seal.

Use the visible L10 dependency route recorded in chapter 4080. Its sibling route binds the diagonal-selection packet to the displayed StreamName, RegSeqRat, DyadicRatCore, and Real rows before the top-level dossier can be consumed.

Now apply the rational interval locator forward route in the same dossier. That route reads the selected diagonal window as interval data only after the displayed source, readback, and tolerance rows have been exposed.

The structural rows H, C, P, N then transport, replay, source, and name this finite route. Since the carrier has no coordinate for a detached tail selector, quotient stream, selected limit, or hidden completion representative, none of those objects can replace the displayed S, R, T, E route. □

Theorem 4080.16 (Cauchy diagonal selection regular-diagonal extraction link)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *packet that exposes the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows can be consumed by the regular-Cauchy diagonal-extraction route of chapter 1773 only through those finite windows. The link exports the same schedule, tolerance, regular-readback, and Real-seal rows; it does not supply countable choice, quotient stream equality, a selected representative, or a detached host diagonal.*

Proof. Project the rows already isolated by Definition 4080.15. The visible part of the accepted packet contains the finite StreamName schedule, the DyadicRatCore tolerance ledger, the RegSeqRat readback, and the terminal Real seal.

Now compare with the carrier surface of chapter 1773. Its diagonal extraction route consumes a source family, a unary selector, aligned StreamName windows, one dyadic tolerance ledger, one regular rational diagonal readback, and a terminal Real seal.

The shared rows therefore identify the only admissible handoff. The Cauchy diagonal selection packet may feed the regular-Cauchy extraction route through the displayed windows, but no row introduces countable choice, quotient equality of streams, a selected representative, or a detached host diagonal. □

The diagonal-selection read first fixes the finite stream windows and regular-rational readback before any terminal real seal is available. This places the route beside the $\mathsf{StreamName}^{\uparrow}$ certificate of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ certificate of chapter 301.

Theorem 4080.17 (Cauchy diagonal StreamName RegSeqRat route)**.** *Every accepted* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *read that exports a diagonal real candidate first consumes the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows before the* $\mathsf{Real}^{\uparrow}$ *seal is inspected. The route exposes no direct real equality test, selected infinite subsequence outside the displayed diagonal ledger, quotient stream equality, countable-choice functional, or host completeness principle.*

Proof. Start with the finite diagonal ledger. Each selected row is read through a displayed stream window, so the consumer sees only the finite $\mathsf{StreamName}^{\uparrow}$ observations attached to the diagonal route.

The regular-rational readback is downstream from those windows. It repacks the observed rational approximants as $\mathsf{RegSeqRat}^{\uparrow}$ data before any real-facing seal is offered.

Only after these two sibling dependencies have been replayed may the $\mathsf{Real}^{\uparrow}$ seal be read. The refused objects are not coordinates in the displayed route, so the diagonal-selection packet remains a finite StreamName and RegSeqRat dependency certificate. □

Theorem 4080.18 (CauchyDiagonalSelection RegSeqRat-StreamName real-completion route)**.** *Every completion-facing* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *consumer that reads a diagonal candidate must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before the selected diagonal window is consumed. The route is therefore a displayed L10 dependency through finite stream windows, regular rational readback, and the real-name boundary, not a detached selector, quotient stream, selected limit, or ambient completeness coordinate.*

Proof. Project the local $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ carrier. Its diagonal window is read only after the stream schedule, regular rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming rows have been exposed.

The cited StreamName, RegSeqRat, and Real sibling chapters provide the three completion-facing boundaries. Since none of those rows stores a selector outside the packet, quotient stream equality, a selected host limit, or ambient completeness, the diagonal-selection consumer remains inside the displayed route. □

Chapter 4081

A Concrete Naming Certificate for $\mathsf{ApproachSpace}^{\uparrow}$

Following Lowen, the $\mathsf{ApproachSpace}^{\uparrow}$ packet records convergence by finite distance-to-set and tolerance observations rather than by a quotient completion or a selected limit object. It sits between the metric carrier of chapter 57, the open-neighbourhood carrier of chapter 88, and the Cauchy-filter completion boundary of chapter 1569. The packet names an L10 convergence object that can be consumed by metric, topology, and filter-completion routes while keeping quotient points, host real equality, countable choice, and ambient completeness outside the carrier.

Definition 4081.1 (ApproachSpace carrier)**.** *An* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, D, T, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the point row, naming the finite point observations and the local classifier surface used by later convergence consumers.*
2. *D is the distance-to-set and tolerance row, recording displayed requests of the form "the observed point is within the listed tolerance of the listed observed set" without importing a host-valued distance function on all subsets.*
3. *T is the topology-neighbourhood row, recording the open-neighbourhood observations generated by the displayed tolerances and the topology-facing classifier inherited from* $\mathsf{Topology}^{\uparrow}$*.*
4. *F is the Cauchy-filter handoff row, naming which finite Cauchy-filter requests may consume D and T before a completion-facing read reaches* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for point observations, distance-to-set tolerances, topology-neighbourhood rows, Cauchy-filter handoffs, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a convergence query through X, D, T, F, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{ApproachSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares convergence observations only through X, D, T, F, H, C, P, N. It has no coordinate for quotient completion points, equality in a host real line, countable-choice selectors, ambient completeness, or a hidden global distance-to-all-subsets oracle.

Lean4 ✓ `BEDC.Derived.ApproachSpaceUp.ApproachSpaceTasteGate single carrier alignment`

Theorem 4081.2 (ApproachSpace NameCert obligations)**.** *For every accepted* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier*

$$A = (X, D, T, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the point row, distance-to-set tolerance row, topology-neighbourhood row, Cauchy-filter handoff row, componentwise transport, displayed replay, provenance, and local naming;*
2. *classifier transport, requiring every accepted convergence comparison to preserve* X, D, T, F *through the displayed* hsame *row* H*;*
3. *metric and topology compatibility, requiring metric-origin tolerance reads to enter through* D *and topology-origin neighbourhood reads to enter through* T *before they are compared as one approach-space observation;*
4. *Cauchy-filter handoff, requiring completion-facing reads to pass through* F *before any* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer uses the packet;*
5. *non-escape, excluding quotient points, host* $\mathsf{Real}^{\uparrow}$ *equality, countable choice, ambient completeness, selected completion representatives, and any hidden bridge claim.*

No local obligation is stored outside these five rows.

Proof. Project the carrier of Definition 4081.1. The finite tuple displays point observations, distance-to-set tolerances, topology-neighbourhood observations, the Cauchy-filter handoff, componentwise transport, replay, provenance, and local naming.

The metric and topology routes meet only at the displayed classifier surface. A metric-origin read must expose D, while a topology-origin read must expose T; transport through H preserves both rows componentwise before the replay row C compares them as one convergence observation.

The completion-facing route is similarly visible. The row F is the only Cauchy-filter handoff coordinate, so a completion consumer reaches $\mathsf{CauchyFilterCompletion}^{\uparrow}$ only after reading the finite rows named by F together with the same transport, replay, provenance, and local naming rows.

Finally inspect the carrier inventory. No coordinate stores quotient completion points, host real equality, countable-choice selectors, selected representatives, ambient completeness, or bridge data. Hence the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ApproachSpaceUp.ApproachSpaceNameCertObligations`

Theorem 4081.3 (ApproachSpace completion handoff)**.** *For every accepted* $\mathsf{ApproachSpace}^{\uparrow}$ *carrier*

$$A = (X, D, T, F, H, C, P, N),$$

every Cauchy-filter completion consumer must read distance-to-set tolerances through D*, topology-neighbourhood observations through* T*, and the Cauchy-filter handoff through* F *before reaching any completion-facing row. The shared finite convergence packet is therefore*

$$D, T \longmapsto F \longmapsto H, C, P, N.$$

It exports no quotient completion point, selected representative, host $\mathsf{Real}^{\uparrow}$ *equality, countable-choice selector, or ambient completeness row.*

Proof. Project the carrier of Definition 4081.1. The rows D, T, F are the only distance-to-set, topology-neighbourhood, and Cauchy-filter handoff coordinates available to a completion-facing consumer.

Apply the metric and topology compatibility clause of Definition 4081.2. A metric-origin tolerance read must enter through D, and a topology-origin neighbourhood read must enter through T, before the packet treats them as one approach-space observation.

The Cauchy-filter handoff clause then forces the completion-facing read through F. The structural rows H, C, P, N only transport, replay, source, and name that same finite packet.

Finally use the non-escape clause from the obligation theorem. Since the carrier contains no quotient completion point, selected representative, host real equality, countable-choice selector, or ambient completeness coordinate, none of those objects can be reached after the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ApproachSpace}^{\uparrow}$ packet is generated from point rows, distance-to-set tolerance rows, topology-neighbourhood rows, Cauchy-filter handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4081.1 and Definition 4081.2 seed the finite approach-space convergence object over point, tolerance, topology-neighbourhood, Cauchy-filter, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApproachSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion points, host Real equality, countable choice, ambient completeness, selected completion representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, metric/topology compatibility, Cauchy-filter handoff exactness, componentwise transport, replay, provenance, and ledger non-escape for BEDC.Derived.ApproachSpaceUp.

Chapter 4082

A Concrete Naming Certificate for MetaCICConfluenceDiamondBoundary$^{\uparrow}$

The MetaCICConfluenceDiamondBoundary$^{\uparrow}$ packet realizes the confluence audit boundary of chapter 9378 as a finite diamond-frontier object. It sits beside the critical-path carrier of chapter 765, the confluence audit packet of chapter 735, the local diamond witness of chapter 3595, the residual checker of chapter 3827, and the subject-reduction discharge socket of chapter 719. The packet names local peaks, candidate joins, residual frontiers, substitution boundaries, bounded checks, and obstruction rows without claiming full confluence or subject reduction.

Falsifiable prediction: Every accepted MetaCICConfluenceDiamondBoundary$^{\uparrow}$ confluence-frontier read exposes local-peak, candidate-join, residual-frontier, substitution-boundary, bounded-check, obstruction, transport, replay, provenance, and local NameCert rows before any diamond-boundary handoff is consumed.

Independence witness: metaciccriticalpath, metacicconfluenceauditpacket, metacic-local-diamond-witness, metacic-confluence-residual-checker, subject-reduction-discharge-socket: the critical-path carrier records dependency pressure without a per-diamond boundary; the confluence audit packet records global audit rows without bounded local frontier status; the local-diamond witness records a peak and candidate join without residual-checker obstruction retention; the residual checker records bounded residual windows without the subject-reduction socket boundary; and the discharge socket records typing-side rows without confluence peak data. None is bijective to MetaCICConfluenceDiamondBoundary$^{\uparrow}$.

Definition 4082.1 (MetaCIC confluence-diamond boundary carrier)**.** *A* MetaCICConfluenceDiamondBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, J, R, S, B, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the local-peak row, naming the source term and the two outgoing reduction or parallel-reduction observations.*
2. *J is the candidate-join row, recording the proposed common reduct and the two route requests into it.*
3. *R is the residual-frontier row, listing residual redex descendants, binder-depth exposure, and branch-local frontier checks.*

4. *S is the substitution-boundary row, naming the substitution-sensitive input that remains visible whenever the candidate join is not an unconditional diamond.*
5. *B is the bounded-check row, recording the finite closedness, support, branch-compatibility, and variable-escape checks applied to the displayed peak.*
6. *O is the obstruction row, naming the still-open endpoints: unrestricted substitution compatibility, premise-free closed parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, and kernel consistency.*
7. *H records componentwise* hsame *transport for local peaks, candidate joins, residual frontiers, substitution boundaries, bounded checks, obstruction rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a confluence query through $K, J, R, S, B, O, H, P, N$.*
9. *P is the* Pkg *provenance over* MetaCICConfluenceDiamondBoundary$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICConfluenceA MetaCICLocalDiamondWitness$^{\uparrow}$, MetaCICConfluenceResidualChecker$^{\uparrow}$, SubjectReductionDischargeSocket$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, J, R, S, B, O, H, C, P, N$. It has no coordinate for a global confluence theorem, a subject-reduction theorem, unrestricted substitution compatibility, quotient equality of terms, evaluator equality, or kernel consistency.

Theorem 4082.2 (MetaCIC confluence-diamond boundary obligations)**.** *For every accepted* MetaCICConfluenceDi *carrier*

$$D = (K, J, R, S, B, O, H, C, P, N),$$

the local NameCert$_{\text{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ *surface consists exactly of local-peak exposure through K, candidate-join exposure through J, residual-frontier exposure through R, substitution-boundary retention through S, bounded-check exposure through B, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICConfluenceDiamondBoundaryUp.MetaCICConfluenceDiamondBoundaryCarrier namecert obligations`

Proof. Project the carrier of Definition 4082.1. Rows K, J, R, S, B, O are the diamond-facing data, and rows H, C, P, N are the structural certificate data.

The local peak and candidate join must be read together. Row K displays the source and the two outgoing observations, while J records only a proposed common reduct and route requests rather than a global Church–Rosser theorem.

The residual and substitution boundaries remain visible next. Rows R and S expose residual descendants, binder-depth data, and substitution-sensitive inputs, while B records the bounded finite checks that have actually been performed.

Finally O keeps the open endpoints named, and H, C, P, N transport, replay, source, and name the same packet. Since no carrier coordinate contains full confluence, subject reduction, unrestricted substitution compatibility, quotient term equality, evaluator equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 4082.3 (MetaCIC confluence-diamond boundary handoff)**.** *Every confluence consumer of an accepted* MetaCICConfluenceDiamondBoundary$^{\uparrow}$ *carrier factors through*

$$K, J, R, S, B, O, H, C, P, N$$

before it may export a diamond-boundary handoff. The handoff may feed the confluence audit packet, local-diamond witness, residual checker, critical-path packet, or subject-reduction discharge socket named above, but it does not fill the substitution boundary, prove subject reduction, prove full confluence, decide evaluator equality, or establish kernel consistency.

Proof. Begin with the obligation surface of Definition 4082.2. It forces every consumer read through the peak row, candidate join row, residual frontier, substitution boundary, bounded checks, and obstruction row before the structural rows are consumed.

The sibling handoffs preserve that boundary. The audit packet may read the confluence-frontier data, the local-diamond witness may read the peak and join rows, the residual checker may read R, S, B, O, the critical-path packet may read the obstruction pressure, and the discharge socket may read only the subject-reduction-facing boundary.

No sibling read adds a hidden theorem row to D. Therefore the handoff remains finite diamond-boundary data and cannot be cited as unrestricted substitution compatibility, subject reduction, full confluence, evaluator equality, or kernel consistency. □

Theorem 4082.4 (MetaCIC confluence-diamond typed-window nonescape)**.** *Any typed-window consumer of an accepted* $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ *packet must display the local peak row K, candidate join row J, residual-frontier row R, substitution-boundary row S, bounded-check row B, obstruction row O, and structural rows H, C, P, N before reading the handoff. The typed window may feed the local-diamond witness and residual-checker siblings named above, but it cannot export subject reduction, full confluence, evaluator equality, or kernel consistency.*

Proof. Start from the carrier of Definition 4082.1. A typed-window read has no coordinate outside $K, J, R, S, B, O, H, C, P, N$, so it must expose the peak, join, residual, substitution, bounded-check, and obstruction rows before any structural replay or provenance row can name the handoff.

Apply the obligation surface of Definition 4082.2. The local $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$ row records exactly those visible obligations, including obstruction retention and the absence of unrestricted substitution compatibility, subject reduction, evaluator equality, and kernel consistency.

Finally apply the handoff theorem Definition 4082.3. The local-diamond witness may consume K, J, and the residual checker may consume R, S, B, O, but neither sibling adds a theorem row to the typed window. Thus the exported data remain a typed local diamond boundary rather than a global confluence, subject-reduction, evaluator-equality, or kernel-consistency theorem. □

Lean follow-up for `BEDC.Derived.MetaCICConfluenceDiamondBoundaryUp` requires Nontrivial and FieldFaithful TasteGate instances at the displayed carrier surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$ packet is generated from local-peak rows, candidate-join rows, residual-frontier rows, substitution-boundary rows, bounded-check rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4082.1, Definition 4082.2, and Definition 4082.3 seed the finite confluence-diamond boundary packet for MetaCIC audit consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution compatibility, full confluence, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-peak exposure, candidate-join exactness, residual-frontier coverage, substitution-boundary retention, bounded-check exactness, obstruction retention, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConfluenceDiamondBoundaryU

Chapter 4083

A Concrete Naming Certificate for ??

Following Darboux and Riemann, the **??** packet isolates the finite upper-sum side of the Darboux interface before a consumer asks for a full upper/lower gap. It sits between the dyadic step-function surface of chapter 428, the tagged finite-sum consumer of chapter 1615, the Darboux integration handoff of chapter 1460, and the tagged-mesh integral boundary of chapter 1526. The packet supplies a finite upper-bound ledger for later Darboux and Riemann criteria without importing host suprema, open-cover compactness, quotient function equality, selected mesh limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4083.1 (Upper Darboux sum carrier)**.** *An **??** carrier is a finite* BHist *packet*

$$U = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the partition row, naming the finite ordered cell list, cell-width ledger, endpoint order, and refinement handle used by every later read.*
2. *B is the cell upper-bound ledger. For each displayed cell it records the chosen upper value row and the comparison proof that every admitted sample value in that cell is below the carried bound.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step value used to display the upper envelope.*
4. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row computed from the displayed widths of P and the upper values exposed by B and D.*
5. *I is the* $\mathsf{DarbouxIntegral}^{\uparrow}$ *provenance row, recording that the upper-sum read is available to the Darboux handoff before any lower-sum or gap row is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing the finite real-name comparison used by upper-bound consumers.*
7. *H records componentwise* hsame *transport for the partition, upper-bound ledger, dyadic step row, Riemann-sum row, Darboux handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to upper-bound reads, from upper-bound reads to the dyadic step row, from the dyadic step row to the Riemann-sum row, and from the sum row to the Darboux handoff.*

9. *Q is the* Pkg *provenance over* ??, DyadicStepFunction$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{??}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host supremum, an open-cover compactness theorem, a selected mesh limit, a lower Darboux sum, quotient equality of functions, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.UpperDarbouxSumUp.TasteGate.UpperDarbouxSumTasteGate single carrier alignment`

Theorem 4083.2 (Upper Darboux sum NameCert obligations)**.** *For every accepted ?? carrier*

$$U = (P, B, D, S, I, R, H, C, Q, N),$$

the local NameCert$_{??}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition row, cell upper-bound ledger, dyadic step row, Riemann-sum row, Darboux handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-refinement stability, requiring each transported refinement of P to carry the same cell-width ledger and to route upper-bound reads through the displayed refinement handle;*
3. *upper-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated upper relation on every displayed cell;*
4. *Darboux handoff, requiring the row I to expose the upper-sum read to* DarbouxIntegral$^{\uparrow}$ *without producing a lower sum, a gap witness, or a host supremum;*
5. *Riemann-sum consumer boundary, requiring every* RiemannSum$^{\uparrow}$ *or* RiemannIntegral$^{\uparrow}$ *consumer to read the finite widths, upper values, replay, provenance, and local name before it can use the upper-sum row.*

Proof. Carrier admission is the projection of Definition 4083.1. The accepted packet displays exactly the partition, upper-bound, dyadic step, finite sum, Darboux handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-refinement stability comes next. The partition row P names the finite cells and widths, so a transported refinement can only expose an upper read through the same displayed cell ledger and refinement handle.

Upper-bound ledger transport is then componentwise. The rows B and D move together under H, and replay through C recomputes the dyadic upper envelope before the finite sum row S is available.

The Darboux and Riemann boundaries are exhausted by S and I. The Darboux consumer receives only the upper-sum handoff, while a Riemann-facing consumer must read the same finite widths, upper values, replay, provenance, and local name before using the sum.

The excluded analytic objects are absent from the carrier. No row stores a host supremum, open-cover compactness, selected mesh limit, quotient function equality, lower-sum datum, or ambient completeness principle. □

Theorem 4083.3 (Upper-sum consumer boundary)**.** *Every* DarbouxIntegral$^{\uparrow}$, RiemannSum$^{\uparrow}$, *or* RiemannIntegral$^{\uparrow}$ *consumer of an accepted ?? carrier $U = (P, B, D, S, I, R, H, C, Q, N)$ must first read the partition row P, the upper-bound ledger B, the* DyadicStepFunction$^{\uparrow}$ *row D, the finite sum row S, the Darboux handoff I, and the comparison row R, with transport, replay, provenance, and*

local naming supplied by H, C, Q, N. The consumer therefore receives a finite upper-sum route rather than a host supremum, an open-cover compactness proof, a selected limiting mesh, or a quotient function-equality principle.

Proof. Project the accepted carrier. The rows upstream of every exported upper-sum read are P, B, D, S, I, R with structural support H, C, Q, N.

For a Darboux-facing read, the handoff row I is reachable only after the partition, upper-bound ledger, dyadic step row, and finite sum row have been replayed. For a Riemann-facing read, the row S is already the displayed finite sum built from those same cells and upper values.

Since the packet has no additional analytic coordinate, every listed consumer reads the same finite route and cannot bypass it through a host supremum, compactness theorem, selected mesh limit, quotient equality, or ambient completeness. □

Closure status for NameCert$_{??}$**.**

Constructive story: An **??** packet is generated from finite partition rows, cell upper-bound ledgers, dyadic step-function rows, Riemann-sum rows, DarbouxIntegral handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4083.1, Definition 4083.2, and Definition 4083.3 seed the finite upper-sum packet over displayed partition cells, cell upper-bound reads, dyadic-step rows, finite-sum rows, Darboux handoff, Real$^{\uparrow}$ comparison, transport, replay, provenance, local naming, and consumer-boundary rows for Darboux, Riemann, DyadicStepFunction$^{\uparrow}$, RiemannSum$^{\uparrow}$, DarbouxIntegral$^{\uparrow}$, RiemannIntegral$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, open-cover compactness, selected mesh limits, lower Darboux sums, lower-sum gap data, quotient function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-refinement stability, upper-bound ledger transport, dyadic step readback, Darboux handoff exactness, Riemann-sum consumer boundary, componentwise transport, displayed replay, provenance, local naming, local NameCert rows, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.UpperDarbouxSumUp.

Chapter 4084

A Concrete Naming Certificate for $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$

Following Kuratowski, the $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ packet records a constructive compactness-defect ledger by finite diameter-cover and finite-net rows rather than by a host infimum over all covers. It sits over the complete metric boundary of chapter 128, the total-bounded finite-net route of chapter 127, the compact metric consumer of chapter 129, the finite epsilon-net witness of chapter 1767, and the dyadic tolerance ledger of chapter 291. The packet names a finite compactness-defect surface for later compactness and functional-analysis consumers; it is not an open-cover compactness theorem, not a choice-selected cover, not a quotient hyperspace, and not an ambient real infimum principle.

Definition 4084.1 (Kuratowski measure of noncompactness carrier)**.** *A* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, C, T, E, D, V, H, R, P, N)$$

with the following visible rows:

1. *X is the displayed* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose bounded subset is being inspected.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary row exposed before completion-facing consumers read any defect claim.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net comparison row, recording the tolerance-indexed route by which defect zero may be compared with total boundedness.*
4. *E is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *witness row, listing the displayed centers, tolerance cell, and coverage ledger available at the inspected scale.*
5. *D is the finite diameter-cover row, recording the listed cover pieces and their displayed dyadic diameter bounds.*
6. *V is the compactness-defect value ledger, comparing the requested dyadic threshold with the finite diameter and finite-net rows without taking a host infimum.*
7. *H records componentwise* hsame *transport for the source, completeness, total-boundedness, finite-net, diameter-cover, value, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a compactness-defect request through $X, C, T, E, D, V, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, C, T, E, D, V, H, R, P, N$. It has no coordinate for an open cover, a selected compactness proof, a host powerset, a quotient compactum, a chosen finite subcover, countable choice, or an ambient real infimum.

Lean4 ✓ `BEDC.Derived.KuratowskiMeasureNoncompactnessUp.KuratowskiMeasureNoncompactnessTasteGate`
`single carrier alignment`

Theorem 4084.2 (Kuratowski measure of noncompactness NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *carrier $K = (X, C, T, E, D, V, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, complete-metric source exposure, total-bounded finite-net comparison, finite epsilon-net witness visibility, finite diameter-cover exactness, compactness-defect value readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every compactness or functional-analysis consumer of the packet must read the displayed route

$$X, C \longmapsto T, E, D, V \longmapsto H, R, P, N$$

before treating the defect ledger as compactness-facing evidence.

Proof. Project the carrier of Definition 4084.1. The only source and completion-facing rows are X and C, and the only compactness-defect rows are the finite total-boundedness, finite-net, diameter-cover, and value ledgers T, E, D, V.

The finite-net and diameter-cover rows are displayed before the value ledger is read. Thus a consumer sees centers, coverage data, cover pieces, dyadic diameter bounds, and the requested threshold as finite rows rather than as a hidden open-cover or infimum argument.

Finally H, R, P, N transport, replay, source, and name the same finite route. Since the carrier has no open-cover, selected subcover, host powerset, quotient compactum, countable-choice, or ambient-infimum coordinate, those objects cannot be exported by the local classifier. □

Theorem 4084.3 (Kuratowski measure complete-metric handoff)**.** *Let $K = (X, C, T, E, D, V, H, R, P, N)$ be an accepted* $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ *packet. Any completion-facing consumer that reads the compactness-defect value ledger V must first expose the complete-metric row C, the total-boundedness comparison row T, and the finite epsilon-net row E, and must then replay the same finite diameter-cover row D through H, R, P, N before reaching a* $\mathsf{CompactMetric}^{\uparrow}$ *or functional-analysis consumer.*

The handoff exports only a finite compactness-defect packet over complete metric and finite-net rows. It does not claim open-cover compactness, selected finite subcovers, quotient compacta, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *infimum.*

Proof. Start from the accepted carrier in Definition 4084.1. The value row V is downstream of X, C, T, E, D, so it cannot be read before the complete-metric source and finite compactness rows have been displayed.

Use the finite epsilon-net route of chapter 1767. The row E supplies centers, tolerance, and coverage data, while T records the total-bounded comparison and D records the finite diameter cover at the inspected scale.

The replay rows H, R, P, N preserve that route when a $\mathsf{CompactMetric}^{\uparrow}$ or functional-analysis consumer asks for compactness-facing evidence. Because no carrier coordinate stores an open

cover, selected subcover, quotient compactum, countable-choice schedule, or ambient infimum, the consumer receives only the displayed compactness-defect handoff. □

Falsifiable prediction: Every accepted $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ compactness-defect read factors through one complete metric source row, one total-boundedness comparison row, one finite epsilon-net witness, one finite diameter-cover ledger, one dyadic value ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it can be consumed by a compact metric or functional-analysis route.

Independence witness: completemetric, totallybounded, compactmetric, finite-epsilon-net, dyadicratcore: $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ is not bijective to any listed sibling because those siblings separately supply completeness, finite-net coverage, compact-metric consumption, and dyadic tolerance rows, while this packet binds them to one compactness-defect value ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}$ packet is generated from a metric or complete-metric source row, total-bounded finite-net comparison, finite epsilon-net witness, finite diameter-cover ledger, dyadic compactness-defect value row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4084.1, Definition 4084.2, and Definition 4084.3 seed the constructive compactness-defect packet over complete metric, finite-net, diameter-cover, dyadic value, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiMeasureNoncompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, choice-selected finite subcovers, quotient compacta, countable choice, host powersets, ambient Real infima, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-metric handoff, total-bounded finite-net comparison, finite epsilon-net witness visibility, finite diameter-cover exactness, compactness-defect value readback, structural transport, and non-escape for BEDC.Derived.KuratowskiMeasureNoncompactnessUp.

Chapter 4085

A Concrete Naming Certificate for ??

Following Plotkin-style structural operational semantics, the **??** packet records a finite replay of displayed Cont execution rows as BEDC NameCert data. It realizes the computation-as-continuation reading of chapter 9365 between the small-step ledger of chapter 469, the monadic continuation surface of chapter 485, and the unary composition substrate of chapter 931. The packet supplies a reusable execution replay carrier for later operational and MetaCIC chapters without importing a host evaluator, quotient trace equality, global confluence, or an external normalization oracle.

Definition 4085.1 (Continuation replay carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$R = (S, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite list of* Cont *step rows, each naming its source history, transition mark, tail context, and target history as displayed BEDC data.*
2. *T is the* $\mathsf{ContinuationTrace}^{\uparrow}$ *ledger that reads the same ordered steps as a small-step trace.*
3. *E is the endpoint row, recording the initial history, terminal history, and boundary read reached by replaying the listed steps.*
4. *H records componentwise* hsame *transport for step rows, endpoint rows, trace ledger, replay route, provenance, and local naming row.*
5. *C records displayed* Cont *replay from the initial endpoint through every row of S to the terminal endpoint, with the unit and composition route readable by* $\mathsf{ContinuationMonad}^{\uparrow}$ *and* $\mathsf{UnaryContMonoid}^{\uparrow}$ *consumers.*
6. *P is the* Pkg *provenance over **??**,* $\mathsf{ContinuationTrace}^{\uparrow}$, $\mathsf{ContinuationMonad}^{\uparrow}$, $\mathsf{UnaryContMonoid}^{\uparrow}$, *chapter 9365,* BHist, hsame, Cont, *and* NameCert.
7. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The classifier reads accepted packets only through S, T, E, H, C, P, N. It has no coordinate for a host evaluator, quotient trace equality, infinite execution object, global confluence theorem, external normalizer, or hidden monadic semantics.

Lean4 ✓ `BEDC.Derived.ContinuationReplayUp.TasteGate.ContinuationReplayTasteGate single carrier alignment`

Theorem 4085.2 (Continuation replay NameCert obligations)**.** *For every accepted ?? carrier*

$$R = (S, T, E, H, C, P, N),$$

the local NameCert$_{??}$ *surface consists exactly of carrier admission, step-row exposure, endpoint exposure,* ContinuationTrace$^{\uparrow}$ *ledger readback, endpoint transport by* hsame*, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host evaluators, quotient traces, infinite execution objects, global confluence, external normalization, and hidden monadic semantics.*

Proof. Carrier admission is the projection of Definition 4085.1. The accepted packet displays only the step list, trace ledger, endpoint row, componentwise transport, replay route, provenance, and local name.

Step-row exposure and endpoint exposure are finite. The row S lists the displayed source, mark, tail, and target data for each step, while E records only the initial and terminal histories reached by replaying that list.

The trace and algebraic consumers read the same route. The ContinuationTrace$^{\uparrow}$ row T supplies small-step readback, and the replay row C supplies the displayed continuation composition route available to ContinuationMonad$^{\uparrow}$ and UnaryContMonoid$^{\uparrow}$ consumers.

Finally, H, P, N transport, source, and name this same finite route. Since no carrier coordinate stores a host evaluator, quotient trace equality, infinite execution, global confluence, external normalization, or hidden monadic semantics, the ledger refusal clauses exhaust the remaining obligation surface. □

Theorem 4085.3 (Continuation replay endpoint transport)**.** *Every endpoint-facing consumer of an accepted ?? carrier $R = (S, T, E, H, C, P, N)$ receives the endpoint row E only after reading the displayed step rows S, the* ContinuationTrace$^{\uparrow}$ *ledger T, and the replay route C, with componentwise transport, provenance, and local naming supplied by H, P, N. The consumer therefore reads finite* Cont *execution rather than a host evaluator graph, quotient trace, global confluence certificate, or external normalizer.*

Lean4 ✓ `BEDC.Derived.ContinuationReplayUp.ContinuationReplayCarrier endpoint transport`

Proof. Project the accepted carrier. The endpoint row E is carried with the same step list S, trace ledger T, replay route C, and structural rows H, P, N.

Replay precedes endpoint export. The route C starts from the initial endpoint, consumes each displayed step row in order, and reaches the terminal endpoint named by E; transport through H preserves this endpoint relation componentwise.

The sibling consumers cannot bypass the route. ContinuationTrace$^{\uparrow}$ reads T as the small-step ledger, while ContinuationMonad$^{\uparrow}$ and UnaryContMonoid$^{\uparrow}$ read the same replay through unit and composition rows. The excluded host objects are not fields of the packet. □

Closure status for NameCert$_{??}$**.**

Constructive story: A **??** packet is generated from finite Cont step rows, ContinuationTrace$^{\uparrow}$ ledgers, endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4085.1, Definition 4085.2, and Definition 4085.3 seed the finite continuation replay packet over displayed continuation steps, trace readback, endpoint exposure, componentwise transport, replay, provenance, local naming, and endpoint-boundary rows for ContinuationTrace$^{\uparrow}$, ContinuationMonad$^{\uparrow}$, UnaryContMonoid$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host evaluator graphs, quotient trace equality, infinite execution objects, global confluence, external normalization, hidden monadic semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, step-row exposure, endpoint exposure, ContinuationTrace$^{\uparrow}$ readback, hsame endpoint transport, replay exactness, replay stability, componentwise transport, displayed Cont replay, provenance, local naming, local NameCert rows, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.ContinuationReplayUp.

Chapter 4086

A Concrete Naming Certificate for $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ packet records the theorem-level exit surface for regular Cauchy completeness as finite BEDC NameCert data. It sits after the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the limit operator of chapter 3392, the uniqueness route of chapter 3390, the located-limit route of chapter 4049, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion phase one theorem object whose rows bind construction, uniqueness, locatedness, and Real sealing without quotient streams, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4086.1 (Cauchy completeness theorem carrier)**.** *A* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, Q, L, U, A, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger read by every regular Cauchy estimate.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing only the requested observation windows.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row associated to those windows and tolerances.*
4. *L is the* $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$ *row that turns the displayed regular Cauchy data into a Real-facing limit route.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row ensuring that the Real seal does not depend on the chosen regular representative.*
6. *A is the* $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$ *row exposing the located-limit boundary for the same displayed regular Cauchy name.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q, L, U, A have been displayed.*
8. *H records componentwise* hsame *transport for the tolerance, window, readback, limit-operator, uniqueness, located-limit, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through*

$$D, S, Q, L, U, A, R \longmapsto \mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyLimitOperator}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{RegularCauchyLocatedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *dependency rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyCompletenessTheoremUp.CauchyCompletenessTheoremUp`
Lean4 ✓ `BEDC.Derived.CauchyCompletenessTheoremUp.CauchyCompletenessTheoremTasteGate single carrier alignment`

Theorem 4086.2 (Cauchy completeness theorem NameCert obligations)**.** *For an accepted* CauchyCompletenessThe *carrier*

$$T = (D, S, Q, L, U, A, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, dyadic tolerance accounting through D, finite-window inspection through S, regular rational readback through Q, limit construction through L, representative independence through U, located-limit boundary control through A, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyCompletenessTheoremUp.TasteGate.CauchyCompletenessTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4086.1. The accepted tuple exposes exactly the dyadic ledger, finite windows, regular readback, limit-operator row, uniqueness row, located-limit row, terminal Real seal, and the structural transport, replay, provenance, and naming rows.

The constructive part is exhausted by D, S, Q, L. The dyadic ledger controls the tolerance requests, the stream row bounds the inspected windows, the regular readback supplies the Cauchy name, and the limit operator is the only route from that name to a Real-facing limit.

The theorem-level seal then reads U, A, R. The uniqueness row makes the seal independent of the displayed representative, the located-limit row records the location boundary, and the Real row is available only after both routes have been displayed. Thus no quotient stream, selected host limit, choice schedule, trichotomy decision, or ambient-completeness coordinate remains available as a NameCert obligation. □

Theorem 4086.3 (Cauchy completeness theorem regular seal route)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RegularCauchyLimitOperator}^{\uparrow} \longmapsto \mathsf{RegularCauchyUniqueness}$$

The consumer receives the terminal $\mathsf{Real}^{\uparrow}$ *seal only as the endpoint of that finite route.*

Proof. Begin with an accepted packet T. Its non-structural rows require the consumer to open the dyadic ledger, finite windows, and regular rational readback before any completion-facing row is visible.

Next the route passes through the regular Cauchy limit operator. The operator supplies the finite construction of the Real-facing limit, while the uniqueness and located-limit rows constrain the same seal by representative independence and locatedness.

The terminal row R is therefore downstream from D, S, Q, L, U, A. Since H, C, P, N only transport, replay, source, and name that route, every L10 phase-exit read factors through the stated chain and cannot bypass it by using quotient streams, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting the DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RegularCauchyLimitOperator row, RegularCauchyUniqueness row, RegularCauchyLocatedLimit row, transport row, replay row, provenance row, or local naming row refutes the packet.

Independence witness: dyadicratcore, streamname, regseqrat, regular-cauchy-limit-operator, regular-cauchy-uniqueness, regular-cauchy-located-limit, real: $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds tolerance, window, regular readback, limit construction, uniqueness, locatedness, terminal Real sealing, and the structural rows into one theorem-level L10 exit certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletenessTheorem}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular readback, RegularCauchyLimitOperator route, RegularCauchyUniqueness route, RegularCauchyLocatedLimit route, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4086.1, Definition 4086.2, and Definition 4086.3 seed the L10 Real and Completion theorem exit surface over regular Cauchy construction, uniqueness, locatedness, terminal sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletenessTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance accounting, finite-window inspection, regular readback, limit-operator routing, uniqueness routing, located-limit routing, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletenessTheoremUp.

Chapter 4087

A Concrete Naming Certificate for $\mathsf{CauchyModulusMajorant}^{\uparrow}$

Following Bishop and Bridges on explicit moduli in constructive analysis, the $\mathsf{CauchyModulusMajorant}^{\uparrow}$ packet records the finite certificate that turns an arbitrary displayed Cauchy threshold schedule into a monotone majorant before completion consumers inspect the seal. It sits after the modulus rows of chapter 292, reads dyadic tolerances from chapter 291, reads finite windows from chapter 126, supplies regular rational schedules for chapter 301, and keeps the terminal real read public only through chapter 38.

Definition 4087.1 (Cauchy modulus majorant carrier)**.** *A* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, D, S, R, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the incoming* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold schedule;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested precisions and majorant bounds;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming which threshold windows may be inspected;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row consumed by regular rational sequence certificates;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite schedule and dyadic comparison rows have been displayed;*
6. *G is the monotone majorant graph row, recording a displayed bound for every inspected threshold window and recording monotonicity as finite comparison data;*
7. *H records componentwise* hsame *transport for the incoming modulus, dyadic ledger, stream windows, regular readback, real seal, majorant graph, continuation, provenance, and local naming rows;*
8. *C records displayed* Cont *replay through the same rows in the order seen by consumers;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice, hidden convergence witnesses, host-level supremum selection, or a real-number read that bypasses the displayed majorant route.

Theorem 4087.2 (Cauchy modulus majorant NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *carrier* $M = (A, D, S, R, E, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission, incoming modulus exposure through* A*, dyadic tolerance exposure through* D*, finite-window exposure through* S*, regular readback through* R*, real sealing through* E*, majorant graph exposure through* G*, componentwise transport through* H*, replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.CauchyModulusMajorantUp.TasteGate.CauchyModulusMajorantTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4087.1. The tuple lists exactly the incoming modulus schedule, dyadic tolerance ledger, stream windows, regular readback, real seal, majorant graph, transport, replay, provenance, and local naming rows.

The majorant obligation is displayed by G. It records a finite bound for each inspected threshold window and records monotonicity as comparison data against the dyadic ledger D.

The consumer route is then forced by C. A RegSeqRat or Real consumer reads S, D, G, R before reaching E, and H transports only the displayed coordinates componentwise.

Since no coordinate for countable choice, hidden convergence witnesses, host supremum selection, or bypassed real reads appears in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4087.3 (Cauchy modulus majorant normalization route)**.** *Let* $M = (A, D, S, R, E, G, H, C, P, N)$ *be an accepted* $\mathsf{CauchyModulusMajorant}^{\uparrow}$ *packet. Every consumer that replaces an arbitrary Cauchy threshold by a monotone schedule must route the replacement through the displayed graph row* G *after reading the finite windows* S *and the dyadic tolerance ledger* D*, and before exposing the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Proof. Start from the carrier definition. The only row that records the monotone majorant is G, while S supplies the inspected windows and D supplies the tolerance comparisons.

The replay row C orders the consumer route through S, D, G, R, E. Thus a regular rational sequence consumer sees the majorant schedule before it receives the readback row R, and a real-number consumer sees the same route before the seal E is opened.

The transport row H is componentwise, so it cannot replace the displayed graph by a host-level monotone choice. The provenance row P records the same dependencies, and N names the local certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusMajorant}^{\uparrow}$ packet is generated from an incoming Cauchy modulus schedule, DyadicRatCore tolerance comparisons, StreamName finite windows, RegSeqRat readback, a Real seal, a displayed monotone majorant graph, componentwise transport, replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$, seed)

Scope closed: Definition 4087.1, Definition 4087.2, and Definition 4087.3 seed the route that replaces arbitrary Cauchy thresholds by a monotone majorant before RegSeqRat and Real consumers inspect the seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusMajorant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, choice-free extraction from every external modulus presentation, completion uniqueness, real-order completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, majorant graph exposure, finite-window routing, dyadic comparison routing, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusMajorantUp.

Chapter 4088

A Concrete Naming Certificate for ??

Following Grothendieck's tensor-norm program and Banach-space tensor products in Ryan and Defant–Floret, the **??** packet records the projective tensor norm as finite BEDC NameCert data over displayed tensor-product, norm, Banach, and bounded-operator rows. It is grounded in the algebraic tensor-product surface of chapter 87, the norm row of chapter 89, the Banach carrier of chapter 90, the bounded-linear-operator interface of chapter 1644, and the Banach-algebra consumer surface of chapter 2189. The packet names only the displayed finite decomposition ledgers and their norm bounds; it is not a host infimum, a quotient completion primitive, a selected optimal decomposition, or a choice principle.

Definition 4088.1 (Projective tensor norm carrier)**.** *A **??** carrier over displayed Banach rows E, F is a finite* BHist *packet*

$$\Pi = (E, F, T, D, B, Q, O, H, C, P, N)$$

with the following rows:

1. *E and F are the source* $\mathsf{Banach}^{\uparrow}$ *rows, including their displayed* $\mathsf{Norm}^{\uparrow}$ *and regular-Cauchy surfaces.*
2. *T is the algebraic* $\mathsf{TensorProduct}^{\uparrow}$ *source row over the carried endpoints of E and F.*
3. *D is the finite decomposition ledger: each entry names a finite list of elementary tensors whose displayed tensor sum represents the inspected tensor row.*
4. *B is the crossnorm-bound ledger, recording for each displayed decomposition the finite scalar estimate obtained from the product of the visible source norms.*
5. *Q is the quotient-facing boundary row, stating that equality of tensor representatives is read only through the tensor-product classifier and the displayed decomposition ledger.*
6. *O is the bounded-operator handoff row used by bilinear and operator consumers; it reads through* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *only after T, D, B, Q are displayed.*
7. *H records componentwise* hsame *transport for Banach endpoints, tensor-product rows, decomposition ledgers, bound ledgers, quotient-boundary rows, operator handoff rows, replay data, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$\mathsf{TensorProduct}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{Banach}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow} \longmapsto \textit{??}.$$

9. *P is the* Pkg *provenance over ??,* $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{BanachAlgebra}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier exports only finite decomposition and bound data. It has no coordinate for a host infimum over all decompositions, a completed quotient tensor product, a selected minimizing decomposition, ambient Banach-space choice, or hidden operator-space equality.

Theorem 4088.2 (Projective tensor norm NameCert obligations)**.** *For every accepted ?? carrier* $\Pi = (E, F, T, D, B, Q, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{??}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed Banach endpoints, tensor-product row, finite decomposition ledger, crossnorm-bound ledger, quotient-boundary row, bounded-operator handoff row, transport, replay, provenance, and local naming;*
2. *crossnorm bound, requiring each elementary tensor read to pass through the displayed source norms and each finite decomposition estimate to pass through B;*
3. *finite decomposition ledger, requiring every inspected tensor norm claim to name a finite decomposition in D before any bound is exported;*
4. *Banach completion handoff, allowing completion-facing consumers to use only the displayed Banach rows and the later completion boundary of Definition 4088.4;*
5. *nonescape from quotient completions, host infima, selected decompositions, cofinal choice, and ambient operator-space equality.*

Lean4 ✓ `BEDC.Derived.ProjectiveTensorNormUp.TasteGate.ProjectiveTensorNormTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4088.1. The endpoint rows E, F, the tensor-product row T, the decomposition ledger D, and the bound ledger B are visible before any projective-norm read is public.

The crossnorm obligation is finite. An elementary tensor estimate reads the two source norms from E and F, and a general displayed tensor estimate reads the finite list in D together with the scalar comparison rows stored in B.

The quotient-facing and completion-facing material is boundary data. The row Q permits only classifier readback for representatives already displayed by T and D, while O supplies the bounded-operator handoff without adding a completed quotient object.

Finally H, C, P, N transport, replay, source, and name only these rows. Since no carrier coordinate stores a host infimum, a selected optimizing decomposition, a quotient completion, cofinal choice, or ambient operator equality, the listed rows exhaust the local NameCert surface. □

Theorem 4088.3 (Projective tensor norm crossnorm boundary)**.** *Every elementary tensor read accepted by a ?? packet exposes the route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{TensorProduct}^{\uparrow} \longmapsto E, F, T, D, B$$

before a scalar bound is exported. The boundary identifies the displayed projective-norm estimate with finite source-norm products from E and F, not with a host infimum, a quotient representative, or a selected decomposition of minimal value.

Proof. Start with an elementary tensor request. The carrier of Definition 4088.1 forces the request to read the Banach endpoint rows E, F and the tensor-product row T before the decomposition ledger is inspected.

For elementary tensors the relevant ledger entry in D is a singleton decomposition, and B records the displayed product of the two source norm reads. For finite sums, the same rows are read entry by entry and then combined by the finite scalar comparison stored in B.

No additional optimization step is available. The packet has no coordinate for a host infimum over all decompositions, no quotient representative selector, and no choice row that could pick a preferred decomposition. □

Theorem 4088.4 (Projective tensor norm Banach completion handoff)**.** *Every Banach-completion or bounded-operator consumer of an accepted ?? packet factors through the displayed route*

$$E, F, T, D, B, Q, O, H, C, P, N.$$

It may use the finite projective-bound ledger as input to later $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, *or* $\mathsf{BanachAlgebra}^{\uparrow}$ *rows, but it receives no completed quotient tensor product, no universal completion object, no selected Cauchy representative, and no ambient completeness theorem.*

Proof. Use the obligation surface of Definition 4088.2. It places every completion-facing read after carrier admission, finite decomposition admission, crossnorm-bound admission, and quotient-boundary readback.

The bounded-operator handoff is exactly the row O. It can route bilinear or operator consumers through the displayed tensor and bound ledgers, but it does not add a separate operator-space quotient or a hidden complete tensor product.

Completion-facing consumers therefore receive only the finite data that the later Banach or operator packet can cite: endpoint Banach rows, tensor-product rows, finite decomposition rows, scalar bound rows, transport, replay, provenance, and local naming. The excluded completion and selection principles have no source coordinate in Π. □

Falsifiable prediction: Every accepted **??** packet must expose Banach endpoints, a TensorProductUp row, a finite decomposition ledger, a crossnorm-bound ledger, a quotient-boundary row, a bounded-operator handoff row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row before any projective tensor norm read is public.

Independence witness: tensorproduct, norm, banach, boundedlinearoperator, banachalgebra: **??** is not bijective to any listed sibling because it jointly carries finite tensor decompositions, crossnorm-bound ledgers, quotient-boundary readback, Banach-completion handoff, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from displayed Banach endpoint rows, a TensorProductUp source row, finite decomposition ledgers, crossnorm-bound ledgers, quotient-boundary rows, bounded-operator handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{??}$, seed)

Scope closed: Definition 4088.1, Definition 4088.2, Definition 4088.3, and Definition 4088.4 seed the finite projective tensor norm surface over tensor-product, norm, Banach, bounded-operator, and Banach-algebra rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, completed projective tensor products, host infima over all decompositions, selected optimizing decompositions, quotient completions, cofinal choice, or ambient operator-space equality.

Upgrade path: Obligation closure requires checked carrier admission, crossnorm-bound exactness, finite decomposition ledger coverage, Banach-completion handoff, quotient-boundary readback, transport, replay, provenance, and local naming for BEDC.Derived.ProjectiveTensorNormUp.

Chapter 4089

A Concrete Naming Certificate for $\mathsf{MetricSeparationReflector}^{\uparrow}$

Following Bishop–Bridges constructive metric separation and Kelley-style uniform separation, the $\mathsf{MetricSeparationReflector}^{\uparrow}$ packet records the finite reflector that turns pseudometric zero-distance evidence into separated metric identity without quotient primitives. It sits after the pseudometric carrier of chapter 1614, the separated metric classifier of chapter 1411, and the metric surface of chapter 57. Its completion handoff is read only through chapter 95 and the L10 real-name spine of chapter 291, chapter 126, chapter 301, and chapter 38.

Definition 4089.1 (Metric separation reflector carrier)**.** *A* $\mathsf{MetricSeparationReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *source row, carrying point observations, finite distance comparisons, and zero-distance witnesses.*
2. *Z is the zero-distance kernel row, recording exactly which displayed pairs of P are admitted as indistinguishable by the pseudometric evidence.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *identity row, consuming Z before any point classifier is allowed to identify two observations.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row, exposing the separated metric consumer surface reached after S.*
5. *C is the* $\mathsf{Completion}^{\uparrow}$ *route row for completion-facing consumers.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to inspect finite zero-distance budgets.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule attached to the same finite tolerance requests.*
8. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from D and W.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal through which numerical zero-distance evidence is read.*
10. *H records componentwise* hsame *transport for source rows, kernel rows, separated identity rows, metric handoffs, completion handoffs, tolerance ledgers, stream observations, regular rational readbacks, Real seals, replay data, provenance, and local naming.*
11. *K records displayed* Cont *replay along the same route.*

12. *G is the* Pkg *provenance over* $\mathsf{MetricSeparationReflector}^{\uparrow}$, $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, Z, S, M, C, D, W, Q, A, H, K, G, N$. It has no coordinate for quotient classes, selected representatives, host equality of points, countable choice, ambient completion, or an external real-line equality oracle.

Lean4 ✓ `BEDC.Derived.MetricSeparationReflectorUp.MetricSeparationReflectorTasteGate single carrier alignment`

Theorem 4089.2 (Metric separation reflector NameCert obligations)**.** *For every accepted* MetricSeparationReflector *carrier $R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the pseudometric source, zero-distance kernel, separated identity, metric handoff, completion route, dyadic tolerance, stream observation, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *kernel exposure, requiring every identity-facing request to display the Z row before it can reach the separated metric row S;*
3. *reflector exactness, requiring S to identify precisely the displayed zero-distance pairs admitted by Z and no point pairs outside that row;*
4. *metric and completion handoff, requiring M before metric consumers and C before completion consumers inspect the separated result;*
5. *numerical readback discipline, requiring D, W, Q, A in that order for any finite tolerance or Real-facing zero-distance read;*
6. *structural preservation through H, K, G, N;*
7. *ledger non-escape, excluding quotient classes, selected representatives, host point equality, countable choice, hidden ambient completion, and external Real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4089.1. The accepted packet exposes the pseudometric source, zero-distance kernel, separated identity, metric and completion handoffs, numerical readback rows, and structural rows.

The reflector clause is forced by the route through Z and S. A pseudometric comparison can become point identity only after the corresponding zero-distance pair has appeared in Z, and S is the only displayed row that exports separated identity.

Metric and completion consumers are downstream of that separated identity row. The metric handoff M and completion row C receive only the reflected classifier, while D, W, Q, A expose the finite tolerance and Real-facing readback route used to inspect zero distance.

Finally H, K, G, N preserve, replay, source, and name exactly these displayed rows. Since no excluded object appears as a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4089.3 (Metric separation reflector zero-distance kernel)**.** *Let $R = (P, Z, S, M, C, D, W, Q, A, H, K, G, N$ be an accepted* $\mathsf{MetricSeparationReflector}^{\uparrow}$ *carrier. Every identity exported to a* $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{Completion}^{\uparrow}$ *consumer factors through the finite route*

$$\mathsf{Pseudometric}^{\uparrow} \longmapsto Z \longmapsto \mathsf{SeparatedMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqR}$$

Thus zero distance is the displayed kernel of the reflector. The packet exports separated classifier identity, not quotient equality, selected representatives, host point equality, countable choice, or an ambient completion theorem.

Proof. Start with an identity-facing consumer of an accepted packet. The carrier definition gives P as the only pseudometric source row and Z as the only zero-distance kernel row, so the request must first expose the displayed kernel evidence.

The separated identity row S consumes that kernel evidence before any metric or completion consumer can inspect the result. Rows M and C then carry the separated classifier into the metric and completion surfaces.

The numerical route is also explicit. A zero-distance tolerance read passes through D, the finite observation schedule W, the regular rational readback Q, and the terminal Real seal A.

Transport, replay, provenance, and local naming in H, K, G, N preserve this route componentwise. No quotient class, selected representative, host equality proof, choice schedule, or ambient completion coordinate is available to bypass the displayed kernel. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricSeparationReflector}^{\uparrow}$ packet is generated from PseudometricUp source evidence, a zero-distance kernel row, a SeparatedMetricUp identity row, MetricSpaceUp and CompletionUp handoffs, DyadicRatCoreUp tolerances, StreamNameUp observations, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4089.1, Definition 4089.2, and Definition 4089.3 seed the finite reflector from pseudometric zero-distance evidence to separated metric identity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricSeparationReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient primitives, selected representatives, host point equality, countable choice, ambient completion, or external Real equality.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance kernel exposure, separated identity exactness, metric and completion handoff, numerical readback, transport preservation, replay, provenance, and non-escape for BEDC.Derived.MetricSeparationReflectorUp.

Chapter 4090

A Concrete Naming Certificate for ??

The **??** packet realizes the local suffix-support rank theorem of Definition 9462.1756 and Definition 9462.1757 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the boundary classifier of chapter 3393, the coverage-rank surface of chapter 3470, and the codon lift-orbit classifier of chapter 4059. The packet records the stop/Trp face as the local two-coordinate square whose proper-support vertices are the stop row and whose full-support corner is the retained Trp row; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden global translation table.

Definition 4090.1 (Stop/Trp face rank-deficiency carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$F = (W, B, C, O, S, R, L, H, K, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six-bit codon coordinates.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *stop/Trp boundary row, selecting the* U+RR *face* $\{\mathrm{UAA}, \mathrm{UAG}, \mathrm{UGA}, \mathrm{UGG}$
3. *C is the* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row that keeps the visible boundary support inside the displayed tile surface.*
4. *O is the* CodonLiftOrbitClassifier$^{\uparrow}$ *row that records the finite orbit-readback context for the same codon window.*
5. *S is the local suffix-support table row based at* UAA*, with supports $\varnothing$, $\{f_3\}$, $\{f_2\}$, and $\{f_2, f_3\}$.*
6. *R is the rank-deficiency row, recording that stop vertices are exactly the rows whose support is a proper subset of $\{f_2, f_3\}$.*
7. *L is the ledger row separating the full-support* UGG *corner from the three proper-support stop vertices.*
8. *H records componentwise* hsame *transport for source windows, boundary tiles, coverage rows, orbit-classifier rows, suffix-support rows, rank rows, ledger rows, replay, provenance, and local naming.*
9. *K records displayed* Cont *replay from a stop-face query through W, B, C, O, S, R, L, H.*
10. *P is the* Pkg *provenance over **??**,* CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$ CodonLiftOrbitClassifier$^{\uparrow}$, *Definition 9462.1757,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{??}$ *row.*

The classifier accepts a stop-face rank read only through $W, B, C, O, S, R, L, H, K, P, N$. It has no coordinate for a hidden genetic-code table, probabilistic dynamics, quotient codon equality, biological causation, or a global semantic equivalence.

Lean4 ✓ `BEDC.Derived.StopTrpFaceRankDeficiencyUp.StopTrpFaceRankDeficiencyTasteGate single carrier alignment`

Theorem 4090.2 (Stop/Trp face rank-deficiency NameCert obligations)**.** *For every accepted ?? carrier $F = (W, B, C, O, S, R, L, H, K, P, N)$, the local* NameCert$_{??}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the codon-window, boundary-tile, coverage, lift-orbit, suffix-support, rank-deficiency, ledger, transport, replay, provenance, and local naming rows;*
2. *face exposure, requiring the stop/Trp read to pass through W and B before any support rank is inspected;*
3. *coverage and orbit compatibility, requiring C and O to preserve the same displayed codon-window context as W and B;*
4. *suffix-rank exactness, requiring S and R to classify* UAA, UAG, *and* UGA *as proper-support rows and* UGG *as the unique full-support row;*
5. *separated ledger discipline, requiring L to keep stop legality and Trp retention as different displayed rows;*
6. *structural preservation through H, K, P, N;*
7. *ledger non-escape, excluding hidden biological tables, stochastic mechanisms, quotient codon equality, global semantics, and external translation classifiers.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4090.1. The accepted packet displays the codon-window source, boundary-tile classifier, coverage route, lift-orbit context, suffix-support table, rank-deficiency row, ledger row, and structural rows.

The face exposure clause forces the local read to begin with W and B. Thus the rank table is attached to the displayed stop/Trp face, not to an unrecorded biological table.

Coverage and orbit compatibility are downstream checks on the same finite window. The rows C and O preserve the boundary support and orbit-readback context before S and R classify local suffix support.

The rank exactness row is the finite table of Definition 9462.1756. Proper subsets of $\{f_2, f_3\}$ give UAA, UAG, and UGA, while the full support gives UGG. The ledger row L records this separation without collapsing the full-support corner into the stop row.

Finally H, K, P, N preserve, replay, source, and name only these displayed rows. Since no excluded object is a carrier coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4090.3 (Rank-deficiency prototype obligation surface)**.** *Every accepted ?? packet realizes the local rank-deficiency prototype of Definition 9462.1756 and Definition 9462.1757 only through the displayed route*

$$W, B, C, O, S, R, L, H, K, P, N.$$

Equivalently, the prototype is available only through the CodonSixBitWindow$^{\uparrow}$ *source window,* DnaWindowSixBounda *boundary tile,* SixBoundaryTileCoverage$^{\uparrow}$ *coverage row,* CodonLiftOrbitClassifier$^{\uparrow}$ *orbit classifier,*

suffix-support rows, rank-deficiency rows, separated ledger rows, transport, replay, provenance, and local NameCert rows. The packet exports no hidden genetic-code table, quotient codon equality, biological causation, or global semantic equivalence.

Proof. Project the carrier of Definition 4090.1. The displayed rows $W, B, C, O, S, R, L, H, K, P, N$ are the whole accepted packet.

The prototype begins with the six-bit window W and boundary tile B. Coverage C and orbit classification O keep that same finite window visible before any suffix-support or rank-deficiency row is read.

The rows S and R are exactly the local support and rank rows, and L keeps the stop vertices separated from the retained Trp corner. Thus the prototype route is a finite BEDC packet route, not a quotient of codons and not an external genetic-code table.

Finally H, K, P, N transport, replay, source, and name only the displayed rows. Since hidden code tables, biological causation, quotient codon equality, and global semantic equivalence are absent from the carrier, no accepted prototype obligation can pass through them. □

Theorem 4090.4 (Stop/Trp face local rank-deficiency route)**.** *Let $F = (W, B, C, O, S, R, L, H, K, P, N)$ be an accepted **??** carrier. Every accepted stop-face rank read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \mathsf{CodonLiftOrbitClassifier}^{\uparrow}$$

Along this route the vertices UAA, UAG, *and* UGA *are exactly the proper-support rows of $\{f_2, f_3\}$, while* UGG *is the unique full-support corner. The packet exports no hidden code table, stochastic mechanism, quotient codon semantics, or global translation classifier.*

Proof. Start with an accepted stop-face rank read. The carrier requires the read to consume the source window W and the boundary tile B, so the four codons are the displayed $\mathrm{U} + \mathrm{RR}$ face.

The coverage row C and lift-orbit row O keep the same finite codon-window context visible. They do not replace the local face by a global biological semantics.

Now read S and R. By Definition 9462.1757, the proper-support vertices in the two suffix coordinates are precisely UAA, UAG, and UGA, and the remaining UGG corner has full support.

The ledger row L records that separation, and H, K, P, N transport, replay, source, and name the route. No excluded semantic or quotient object appears in the carrier, so the read cannot escape the displayed rank-deficiency packet. □

Falsifiable prediction: A **??** consumer that classifies UAA, UAG, or UGA without displaying the suffix-support row S and rank row R is not reading an accepted packet of this chapter.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, codon-lift-orbit-classifier: **??** is not bijective to any listed sibling because it jointly carries the source window, stop/Trp boundary face, coverage row, orbit-readback context, suffix-support table, rank-deficiency classifier, separated ledger, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.StopTrpFaceRankDeficiencyUp.TasteGate` must provide explicit Nontrivial **??** and FieldFaithful **??** instances preserving $W, B, C, O, S, R, L, H, K, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{??}$**.**

Constructive story: A **??** packet is generated from CodonSixBitWindowUp source rows, DnaWindowSixBoundaryTileUp stop/Trp boundary rows, SixBoundaryTileCoverageUp coverage rows, CodonLiftOrbitClassifierUp orbit rows, suffix-support rows, rank-deficiency rows, a separated ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4090.1, Definition 4090.2, and Definition 4090.4 seed the finite stop/Trp face rank-deficiency packet over the local suffix-support theorem.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, biological causation, stochastic fitness modeling, quotient codon equality, hidden translation tables, or global semantics.

Upgrade path: Obligation closure requires checked carrier admission, face exposure, coverage and orbit compatibility, suffix-rank exactness, separated ledger discipline, Nontrivial StopTrpFaceRankDeficiencyUp, and FieldFaithful StopTrpFaceRankDeficiencyUp for BEDC.Derived.StopTrpFaceRankDeficiencyUp.

Chapter 4091

A Concrete Naming Certificate for ??

Following Bourbaki/Kelley-style bornology plus Bishop–Bridges metric boundedness, the **??** packet records the metric bornology generated by displayed center-radius bounded-set data. It sits between the bounded-subset carrier of chapter 1757, the center-radius row of chapter 1450, the metric carrier of chapter 57, the radius comparison surfaces of chapter 38 and chapter 766, the ordinary bounded-family closure of chapter 3370, and the completion-facing specialization of chapter 1896. The packet turns metric ball containment and bounded subset evidence into a reusable bornology surface without introducing host suprema, compactness, selected bounds, quotient families, trichotomy, or ambient real completeness.

Definition 4091.1 (MetricBornology carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$M = (X, S, c, r, B, U, E, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier whose distance and point-classifier rows are the only metric data available to the packet.*
2. *S is the bounded-subset classifier row read through the* $\mathsf{BoundedSet}^{\uparrow}$ *packet.*
3. *c is the displayed center or witness row for the metric bound.*
4. *r is the radius row, read through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *B is the* $\mathsf{MetricBall}^{\uparrow}$ *containment ledger connecting accepted membership reads of S to the displayed ball with center c and radius r.*
6. *U is the finite-union bornology ledger, recording union requests only over previously accepted metric-bounded families.*
7. *E is the hereditary bounded-family ledger, recording subfamily requests only through a displayed parent family already accepted by S or U.*
8. *D is the consumer handoff row for* $\mathsf{Bornology}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *readers.*
9. *H records componentwise* hsame *transport for $X, S, c, r, B, U, E, D, C, P, N$.*
10. *C records displayed* Cont *replay from a bounded-family request through metric carrier, classifier, center, radius, ball containment, hereditary closure, finite union, handoff, provenance, and local naming.*

11. *P is the* Pkg *provenance over* ??, $\mathsf{Met}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, $\mathsf{Bornology}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{??}$ *row.*

The carrier compares metric-bounded families only through these rows. It has no coordinate for a host supremum, selected bound, compactness proof, open-cover compactness, quotient family, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MetricBornologyUp.MetricBornologyTasteGate single carrier alignment`

Theorem 4091.2 (MetricBornology NameCert obligations)**.** *For every accepted ?? carrier $M = (X, S, c, r, B, U, E, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{??}$ *surface consists exactly of carrier admission for the metric row, bounded-subset classifier, center or witness row, radius row, metric-ball containment ledger, finite-union ledger, hereditary ledger, consumer handoff, componentwise transport, displayed replay, provenance, and local naming rows.*

Every accepted bounded-family read must first expose the $\mathsf{Met}^{\uparrow}$ *row X, the* $\mathsf{BoundedSet}^{\uparrow}$ *classifier S, the center-radius data c, r, and the* $\mathsf{MetricBall}^{\uparrow}$ *ledger B. Hereditary and finite-union bornology reads may then consume only E and U, and downstream* $\mathsf{Bornology}^{\uparrow}$ *or* $\mathsf{CauchyBornology}^{\uparrow}$ *consumers may read the packet only through D, H, C, P, N.*

Proof. Project the carrier of Definition 4091.1. The metric rows are X, S, c, r, B, and the bornology rows are U, E, D; the remaining rows only transport, replay, source, and name the same packet.

Carrier admission and boundedness are forced by the sibling routes. The bounded-subset packet supplies S, c, r, B, while the metric-ball packet supplies the center-radius comparison over the same metric row X and radius row r.

Hereditary and finite-union reads are then confined to E and U. Since neither row stores a host powerset, selected bound, supremum, compactness proof, quotient family, or choice schedule, the local NameCert obligations are exactly the displayed metric-bounded-family rows. □

Theorem 4091.3 (MetricBornology bounded-set radius handoff)**.** *Let $M = (X, S, c, r, B, U, E, D, H, C, P, N)$ be an accepted ?? carrier. Every* $\mathsf{BoundedSet}^{\uparrow}$ *or* $\mathsf{MetricBall}^{\uparrow}$ *read exported from M follows the route*

$$X, S, c, r, B \longmapsto U, E \longmapsto D, H, C, P, N.$$

The radius row r is read only through $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$*, the containment row B is read only as a* $\mathsf{MetricBall}^{\uparrow}$ *ledger, and the resulting bounded-family surface is handed to* $\mathsf{Bornology}^{\uparrow}$ *and* $\mathsf{CauchyBornology}^{\uparrow}$ *consumers before hereditary or finite-union bornology closure can be used downstream.*

Proof. Start with a bounded-set request. By Definition 4091.1, it must name the metric carrier X, the bounded-subset classifier S, the center or witness c, the radius row r, and the metric-ball containment ledger B.

Use the radius anchors next. The row r is available only through the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, and B is available only through the $\mathsf{MetricBall}^{\uparrow}$ center-radius ledger. Thus the request cannot replace the displayed radius by a selected supremum or by a compactness argument.

Finally pass to the bornology rows. The hereditary row E and finite-union row U consume the already displayed metric-bounded families, while D, H, C, P, N export the same surface to $\mathsf{Bornology}^{\uparrow}$ and $\mathsf{CauchyBornology}^{\uparrow}$ readers. No downstream consumer receives a host family, quotient family, selected bound, trichotomy proof, or ambient real completeness datum. □

Closure status for NameCert$_{??}$**.**

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4091.1, Definition 4091.2, and Definition 4091.3 seed the finite metric bornology surface over bounded-set, metric-ball, real-radius, general-bornology, and Cauchy-bornology consumers.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, selected bounds, quotient families, compactness, open-cover compactness, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric carrier admission, bounded-subset classifier exposure, radius comparison, metric-ball containment, hereditary and finite-union ledgers, bornology handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricBornologyUp.

Constructive story: A **??** packet is generated from a MetricSpaceUp carrier, a BoundedSetUp classifier, a displayed center or witness, a RealUp radius read through RealAlgOrderUp, a MetricBallUp containment ledger, hereditary and finite-union bornology ledgers, BornologyUp and CauchyBornologyUp handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4092

A Concrete Naming Certificate for $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's treatment of completion by universal comparison, the $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ packet records the finite BEDC surface on which several constructive completion presentations can be read through one comparison certificate. It sits after the Cauchy-comparison route of chapter 1954, the Bishop completion comparison of chapter 1508, the separated uniqueness ledger of chapter 1463, and the metric uniqueness boundary of chapter 1796. The packet also reads the finite observation spine of chapter 126, chapter 291, chapter 301, and chapter 38. It compares displayed constructive completion routes only; it is not quotient-completion equality, not selected representative equality, and not a proof that $\mathsf{Real}^{\uparrow}$ is ambiently complete.

Definition 4092.1 (Constructive completion equivalence carrier)**.** *A* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (C, B, U, M, S, D, R, A, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyCompletionComparison}^{\uparrow}$ *row, comparing unit, universal-property, functorial, and metric-completion presentations by finite observation data.*
2. *B is the* $\mathsf{BishopCompletionComparison}^{\uparrow}$ *row, comparing regular located Cauchy data with the completion-facing real-name seal.*
3. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *row, recording the separated uniqueness ledger available to completion consumers.*
4. *M is the* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *row, recording the metric completion comparison boundary over a common metric source.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row common to the completion observations.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows and comparison requests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the windows and dyadic ledger.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the comparison, window, tolerance, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for C, B, U, M, S, D, R, A and the local naming row.*
10. *K records displayed* Cont *replay through the comparison rows, observation spine, transport, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$, $\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $C, B, U, M, S, D, R, A, H, K, P, N$. It has no coordinate for quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConstructiveCompletionEquivalenceUp.ConstructiveCompletionEquivalenceTasteGate`
`single carrier alignment`

Theorem 4092.2 (Constructive completion equivalence NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier*

$$E = (C, B, U, M, S, D, R, A, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$ *surface consists exactly of Cauchy-comparison exposure through C, Bishop-comparison exposure through B, separated uniqueness exposure through U, metric uniqueness exposure through M, finite-window exposure through S, dyadic tolerance exposure through D, regular rational readback through R, terminal real-name sealing through A, componentwise transport through H, displayed replay through K, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4092.1. The four comparison rows C, B, U, M are the only completion-presentation coordinates, while S, D, R, A are the shared finite observation spine.

The observation spine is read after the comparison rows have been named. Finite windows, dyadic tolerances, and regular rational readback provide the data by which the terminal $\mathsf{Real}^{\uparrow}$ seal is reached.

Finally H, K, P, N transport, replay, source, and name the same packet. Since no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness row, or ambient real-completeness theorem occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4092.3 (Constructive completion equivalence comparison route)**.** *Every completion-equivalence consumer of an accepted* $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyCompletionComparison}^{\uparrow}$, $\mathsf{BishopCompletionComparison}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{MetricCompletionUniquenes}$

as displayed rows before it may compare constructive completion presentations. The route exports only the finite comparison certificate $E = (C, B, U, M, S, D, R, A, H, K, P, N)$ and does not identify quotient completions or assert ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the obligation surface of Definition 4092.2. Any consumer must first expose the Cauchy-comparison, Bishop-comparison, separated uniqueness, and metric uniqueness rows.

The shared observation spine is then consumed through S, D, R, A. These rows give finite windows, tolerance ledgers, regular rational readback, and a terminal real-name seal, so the comparison remains a displayed certificate rather than a quotient completion object.

The structural rows H, K, P, N preserve the same route under transport and replay. Therefore no consumer can bypass the finite comparison packet by citing selected representatives, host equality of completed points, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ read exposes the CauchyCompletionComparison, BishopCompletionComparison, CompletionUniqueness, MetricCompletionUniqueness, StreamName, DyadicRatCore, RegSeqRat, RealUp, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any constructive-completion comparison is consumed.

Independence witness: cauchy-completion-comparison, bishopcompletioncomparison, completion-uniqueness, metric-completion-uniqueness, streamname, dyadicratcore, regseqrat, real: none is bijective with $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ because this carrier alone binds the four comparison surfaces to one shared finite observation spine and one local NameCert row without quotient-completion equality or ambient RealUp completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}$ packet is generated from CauchyCompletionComparison, BishopCompletionComparison, CompletionUniqueness, MetricCompletionUniqueness, StreamName, DyadicRatCore, RegSeqRat, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4092.1, Definition 4092.2, and Definition 4092.3 seed the finite comparison certificate for constructive completion presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveCompletionEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, host equality of completed objects, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-row exposure, shared-window coverage, dyadic tolerance exactness, regular readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCompletionEquivalenceUp.

Chapter 4093

A Concrete Naming Certificate for $\mathsf{EffectiveModulus}^{\uparrow}$

Following Bishop-style effective convergence, the $\mathsf{EffectiveModulus}^{\uparrow}$ packet records the finite threshold data by which a regular rational Cauchy name answers a dyadic tolerance request before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer reads the terminal seal. It sits over the regular rational source of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the modulus surface of chapter 292. The packet exposes a finite effective-modulus row, not a host function from tolerances to thresholds.

Definition 4093.1 (EffectiveModulus carrier)**.** *An* $\mathsf{EffectiveModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, S, D, u, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational observations are being controlled.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window family through which R is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the requested dyadic precision, comparison cells, and halving ledger.*
4. *u is the unary tolerance-index row naming the finite precision request inside the same history.*
5. *L is the modulus witness ledger. It records a threshold row saying that all inspected tails of R beyond u remain inside the tolerance read from D.*
6. *H records componentwise* hsame *transport for the regular source, stream windows, dyadic tolerance, unary index, witness ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a dyadic tolerance request through S and R to the witness ledger L.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$ *row.*

The carrier contains no coordinate for quotient equality of real names, selected limits, countable choice, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.EffectiveModulusUp.EffectiveModulusTasteGate single carrier alignment`

Theorem 4093.2 (EffectiveModulus NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveModulus}^{\uparrow}$ *carrier*

$$M = (R, S, D, u, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, dyadic tolerance accounting, unary tolerance-index admission, modulus witness readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted read must display the route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the rows $R, S, D, u, L, H, C, P, N$*.*

Proof. Project the carrier of Definition 4093.1. Its public rows are exactly the regular rational source, stream-window family, dyadic tolerance, unary index, modulus witness ledger, transport, replay, provenance, and local name.

The source-window exposure and tolerance clauses are read through S and D. A tolerance request cannot reach the witness ledger until the same finite observation windows and dyadic comparison cells have been displayed.

The unary index row u supplies the finite threshold coordinate, while L records the corresponding tail-control witness for R. Transport through H and replay through C preserve that source route without adding a second threshold selector.

Finally P and N source and name only the displayed packet. Since the carrier has no quotient, selected-limit, choice, host-equality, or completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4093.3 (EffectiveModulus regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveModulus}^{\uparrow}$ *packet reads the modulus witness ledger only after the regular source row* R*, stream-window family* S*, dyadic tolerance ledger* D*, and unary tolerance index* u *have been exposed. The handoff supplies the displayed* $\mathsf{CauchyModulus}^{\uparrow}$*-compatible threshold row* L *together with* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *structural rows, and it supplies no quotient real equality, selected representative, countable-choice functional, host real equality, or ambient completeness principle.*

Proof. Begin with an accepted carrier $M = (R, S, D, u, L, H, C, P, N)$. The regular rational readback is the row R, and all admissible observations of it pass through the finite windows S.

Read the dyadic request next. The tolerance row D fixes the comparison scale, and the unary row u names the finite threshold request attached to that scale.

The witness ledger L is then the only modulus-facing handoff row. It is transported through H and replayed through C, while P and N preserve the same provenance and local name.

Thus any regular-Cauchy or real-seal consumer receives a finite threshold surface over $R, S, D, u, L, H, C, P, N$. The excluded quotient, representative-selection, choice, host-equality, and completeness objects are absent from the packet, so they cannot enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffectiveModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName observation windows, a DyadicRatCore tolerance row, a unary tolerance index, a modulus witness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EffectiveModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4093.1, Definition 4093.2, and Definition 4093.3 seed the finite effective-modulus route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{EffectiveModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected representatives, countable choice, host real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-window exposure, dyadic tolerance accounting, unary threshold admission, modulus witness readback, regular-Cauchy handoff stability, and RealUp seal non-escape for BEDC.Derived.EffectiveModu

Chapter 4094

A Concrete Naming Certificate for ??

Following Bishop and Bridges and Mac Lane, the **??** packet records the functoriality certificate for separated completion as finite BEDC NameCert data. It sits after the separated-completion surface of chapter 1612, the public Cauchy reflector of chapter 1585, the metric-completion handoff of chapter 1506, and the L10 finite-name route through chapter 126, chapter 301, and chapter 38. The packet names the source-to-target transport of a uniformly continuous map through separated completion without importing quotient completions, chosen limits, host function extensionality, countable choice, ambient completeness, or bridge checking.

Definition 4094.1 (Separated completion functor carrier)**.** *A **??** carrier is a finite* BHist *packet*

$$F = (M, N, f, S, T, R, L, W, Q, A, H, C, P, E)$$

with the following displayed rows:

1. *M is the source metric row, naming the source carrier, metric comparison surface, and finite Cauchy-input budget.*
2. *N is the target metric row, naming the target carrier, metric comparison surface, and completion-facing output budget.*
3. *f is the source map row, recording the displayed uniformly continuous map data consumed by the completion route.*
4. *S is the source* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row, read from the source metric before any completed source name is transported.*
5. *T is the target* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row, read from the target metric before any completed target name is exposed.*
6. *R is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row, recording the reflector handoff that turns source Cauchy data into a completion-facing read.*
7. *L is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, recording the metric-completion consumer surface through which the target completion read is delivered.*
8. *W is the* $\mathsf{StreamName}^{\uparrow}$ *local observation row attached to source and target completion names.*
9. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *local tolerance row consumed after W and before any terminal real-name seal.*
10. *A is the* $\mathsf{Real}^{\uparrow}$ *local seal row reached only through W, Q and the displayed separated-completion rows.*

11. *H records componentwise* hsame *transport for source metric data, target metric data, the source map, separated-completion rows, reflector and metric-completion rows, local L10 rows, replay, provenance, and naming.*
12. *C records displayed* Cont *replay from a source completion request through $M, N, f, S, T, R, L, W, Q, A, H, P, E$.*
13. *P is the* Pkg *provenance over* **??**, SeparatedCompletion$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, MetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *E is the local* NameCert$_{??}$ *row.*

The classifier compares accepted packets only through $M, N, f, S, T, R, L, W, Q, A, H, C, P, E$. It has no coordinate for quotient completion equality, selected limit representatives, host function extensionality, countable choice, ambient completeness of either completed metric, or a standard bridge.

Lean4 ✓ `BEDC.Derived.SeparatedCompletionFunctorUp.SeparatedCompletionFunctorTasteGate single carrier alignment`

Theorem 4094.2 (Separated completion functor NameCert obligations)**.** *For every accepted* **??** *carrier*

$$F = (M, N, f, S, T, R, L, W, Q, A, H, C, P, E),$$

the local NameCert$_{??}$ *surface consists exactly of source metric admission, target metric admission, source map admission, separated-completion exposure on both sides, Cauchy-reflector exposure, metric-completion exposure, StreamName observation, RegSeqRat tolerance readback, Real seal readback, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.SeparatedCompletionFunctorUp.SeparatedCompletionFunctorCarrier namecert obligations`

Proof. Project the carrier of Definition 4094.1. The rows M, N, f expose the source metric, target metric, and map data before any completion-facing consumer reads the packet.

The separated-completion rows come next. Rows S and T keep the source and target completion surfaces separated before the reflector row R and the metric-completion row L are consumed.

The local numerical route is finite. The StreamName row W, RegSeqRat row Q, and Real row A display the L10 readback used by completion consumers, while H, C, P, E transport, replay, source, and name the same route.

Since no carrier coordinate contains quotient completion equality, a chosen limit representative, host function extensionality, countable choice, ambient completeness, or a bridge theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4094.3 (Separated completion functor Cauchy-reflector route)**.** *Let $F = (M, N, f, S, T, R, L, W, Q, A, H,$ be an accepted* **??** *carrier. Every completion-facing read of the transported map factors through the route*

$$M, N, f, S, T, R, L, W, Q, A, H, C, P, E$$

before a target completion name is exported. In particular, the Cauchy-reflector row R and metric-completion row L are read only after both separated-completion rows S, T have been displayed, and the local StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows are read only as the terminal L10 route of that same packet.*

Proof. Start with a transported-map request for an accepted carrier. The source and target metric rows M, N and the source map row f are the only input coordinates available to the request.

Read the separated-completion rows S, T before any completion-facing handoff. This forces equality-facing and completion-facing consumers to remain on the separated surfaces carried by the packet.

The Cauchy reflector and metric completion handoffs are then R and L. Their consumers receive the local L10 route only through W, Q, A, so the exported target completion name remains attached to the displayed source map, separated-completion rows, transport, replay, provenance, and local naming.

Because the carrier contains no quotient completion class, selected limit, host function-extensionality proof, countable-choice selector, ambient-completeness row, or bridge theorem, no completion-facing read can bypass the named route. □

Falsifiable prediction: Every accepted **??** transport read displays source metric, target metric, source map, source separated-completion, target separated-completion, CauchyCompletionReflector, MetricCompletion, StreamName, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows before a target completion name is exported.

Independence witness: separatedcompletion, cauchycompletionreflector, metriccompletion, streamname, regseqrat, real: the separated completion functor packet is not bijective to any listed sibling because it jointly carries a source metric row, target metric row, source map row, separated-completion rows on both sides, reflector and metric-completion handoffs, the local L10 readback route, transport, replay, provenance, and local naming.

Closure status for NameCert$_{??}$**.**

Constructive story: A **??** packet is generated from source metric data, target metric data, a source map row, separated-completion rows on both sides, CauchyCompletionReflectorUp and MetricCompletionUp handoffs, StreamNameUp observations, RegSeqRatUp tolerance readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{??}$, seed)

Scope closed: Definition 4094.1, Definition 4094.2, and Definition 4094.3 seed the finite functorial separated-completion route from a source map through separated completions and the L10 completion readback.

Formal verification: FormalStatus(NameCert$_{??}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host function extensionality, countable choice, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked source metric admission, target metric admission, source map admission, separated-completion exposure on both sides, Cauchy-reflector handoff, metric-completion handoff, StreamName observation, RegSeqRat tolerance readback, Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionFunctorUp.

Chapter 4095

A Concrete Naming Certificate for EffectiveCauchySequenceSpace$^{\uparrow}$

Following Bishop and Bridges, the EffectiveCauchySequenceSpace$^{\uparrow}$ packet records the finite effective Cauchy-sequence surface used by Real and Completion consumers before any quotient of Cauchy names is formed. It reads the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the comparison boundary of chapter 1820, and the terminal real-name seal of chapter 38. The packet is a displayed sequence-space carrier with a replayable effective route, not a selected representative, not a host sequence type, and not ambient completeness of Real$^{\uparrow}$.

Definition 4095.1 (EffectiveCauchySequenceSpace carrier)**.** *An* EffectiveCauchySequenceSpace$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, Q, A, T, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite* StreamName$^{\uparrow}$ *window family through which sequence observations are requested.*
2. *R is the* RegSeqRat$^{\uparrow}$ *readback row exposing regular rational values on those windows.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger naming the dyadic request, halving data, and comparison cells.*
4. *Q is the* CauchyEquivalenceSetoid$^{\uparrow}$ *comparison row. It records equality-facing comparison only after the windows, regular readback, and dyadic tolerance rows have been displayed.*
5. *A is the terminal* Real$^{\uparrow}$ *seal row reached by the effective Cauchy route.*
6. *T records transport from the displayed stream windows and regular rational rows to the comparison and real-seal rows.*
7. *K records replay of the same route for downstream completion consumers.*
8. *H records componentwise* hsame *transport for S, R, D, Q, A, T, K and the local naming row.*
9. *C records displayed* Cont *replay from finite tolerance requests through $S, R, D, Q, A, T, K, H, P, N$.*
10. *P is the* Pkg *provenance over* EffectiveCauchySequenceSpace$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* CauchyEquivalenceSetoid$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, R, D, Q, A, T, K, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchySequenceSpaceUp.EffectiveCauchySequenceSpaceTasteGate`
`single carrier alignment`

Theorem 4095.2 (EffectiveCauchySequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier*

$$E = (S, R, D, Q, A, T, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, finite StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyEquivalenceSetoid comparison routing, Real seal admission, transport, replay, componentwise* hsame *stability, displayed* Cont *continuation,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4095.1. Its public rows are precisely the stream windows, regular rational readback, dyadic tolerance, setoid comparison, real seal, transport, replay, stability, continuation, provenance, and local name.

The comparison row Q can be read only after S, R, D have been displayed. Thus equality-facing consumers first consume finite windows, rational observations, and dyadic tolerance data before they see the setoid boundary.

The terminal seal A is downstream of the same route and is preserved only through T, K, H, C, P, N. Since quotient stream equality, selected representatives, countable choice, host real equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4095.3 (EffectiveCauchySequenceSpace regular readback route)**.** *Every regular readback exported by an accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyEquivalenceSetoid}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside $S, R, D, Q, A, T, K, H, C, P, N$. The route exports finite observation and tolerance data for completion-facing consumers and does not export quotient equality of streams, a selected limiting representative, countable choice, or host real equality.

Proof. Start with Definition 4095.2. It forces a consumer to read the StreamName windows S before using the RegSeqRat row R.

The dyadic ledger D supplies the requested tolerance for that readback, and the comparison row Q consumes the same S, R, D data before any equality-facing statement is visible.

Finally the Real seal A is reached only after the comparison route has been displayed. Transport and replay through T, K, H, C, P, N preserve the same row order, so no quotient, selected representative, choice functional, or host equality can be inserted into the readback route. □

Theorem 4095.4 (EffectiveCauchySequenceSpace Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ *carrier must read the real seal as the terminal row of $S, R, D, Q, A, T, K, H, C, P, N$. The seal records that the displayed effective Cauchy route may be consumed by Real and Completion chapters; it does not assert that* $\mathsf{Real}^{\uparrow}$ *is ambiently complete, that all Cauchy streams have selected limits, or that Cauchy-equivalent streams are identified by host equality.*

Proof. Project A from the carrier. The definition places A after the StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and CauchyEquivalenceSetoid comparison row.

Apply the regular readback route of Definition 4095.3. It binds every Real-facing read to the same finite observation and tolerance route before the terminal seal is consumed.

The remaining rows T, K, H, C, P, N transport, replay, source, and name that route. Because the carrier has no coordinate for ambient completeness, selected limits, quotient equality, or host equality, the Real seal remains a boundary row rather than a completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid comparison row, a terminal RealUp seal, transport, replay, componentwise hsame stability, displayed Cont continuation, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4095.1, Definition 4095.2, Definition 4095.3, and Definition 4095.4 seed the finite effective Cauchy-sequence-space route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySequenceSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient equality of Cauchy streams, selected limits, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance accounting, CauchyEquivalenceSetoid comparison routing, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchySequenceSpaceUp.

Chapter 4096

A Concrete Naming Certificate for $\mathsf{FiniteLowerBoundFold}^{\uparrow}$

Following the finite-cover proof of Heine–Cantor in chapter 9237, the $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ packet records the rational lower-bound fold that turns finitely many local continuity radii into one uniform precision row. It sits over compact finite coverage from chapter 60, the metric source of chapter 57, pointwise continuity rows from chapter 59, rational positivity from chapter 37, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet makes the finite $\mathsf{ProbeBundle}$ fold an explicit NameCert object.

Definition 4096.1 (FiniteLowerBoundFold carrier)**.** *A* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row carrying the finite coverage witness.*
2. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine indexing the bundled centers.*
3. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *lower-bound candidates, one row for each bundled probe.*
4. *A is the accumulator ledger produced by primitive recursion over $\mathcal{P}$.*
5. *B is the certified lower-bound row saying that the accumulator is below every displayed candidate in R.*
6. *L is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the accumulator as the tolerance-only precision.*
7. *H records componentwise* hsame *transport for the compact source, bundle spine, rational candidates, accumulator, lower-bound certificate, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the fold over the finite bundle.*
9. *Q is the* Pkg *provenance over* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$ *row.*

The carrier contains no host minimum, selected center outside $\mathcal{P}$, open-cover compactness object, quotient point equality, external modulus function, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldTasteGate single carrier alignment`

Theorem 4096.2 (FiniteLowerBoundFold NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *carrier*

$$F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$ *surface consists exactly of compact source exposure,* $\mathsf{ProbeBundle}$ *spine admission, positive* $\mathsf{Rat}^{\uparrow}$ *candidate admission, accumulator recursion, certified lower-bound readback,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise transport, displayed replay, provenance, and local naming. The fold route is confined to*

$$\mathsf{Compact}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Continuous}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

Proof. Project the carrier in Definition 4096.1. The only rows visible to a consumer are the compact source, finite bundle, rational candidate family, accumulator, lower-bound certificate, handoff, and structural rows.

The bundle spine $\mathcal{P}$ supplies the induction domain. At each finite step the fold reads one positive rational candidate from R and updates the accumulator row A through the displayed replay route C.

The lower-bound certificate B is attached to the same finite recursion. It states that the final accumulator is below every candidate in R, so a downstream read of L receives a classifier-stable precision row rather than a host minimum.

Transport, provenance, and local naming are supplied by H, Q, N. Since none of the excluded host, open-cover, quotient, external-modulus, or completeness objects occurs in the carrier, the listed obligations exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldCarrier namecert obligations`

Theorem 4096.3 (FiniteLowerBoundFold uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ *packet obtains its* $\mathsf{UniformModulus}^{\uparrow}$ *precision row only through the finite* $\mathsf{ProbeBundle}$ *spine* $\mathcal{P}$*, the positive* $\mathsf{Rat}^{\uparrow}$ *candidate ledger* R*, the accumulator row* A*, the lower-bound certificate* B*, and the structural rows* H, C, Q, N*. The consumer receives no host finite minimum, selected center outside the bundle, open-cover compactness object, quotient point equality, external modulus selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Lean4 ✓ `BEDC.Derived.FiniteLowerBoundFoldUp.FiniteLowerBoundFoldCarrier uniform modulus handoff`

Proof. Start with an accepted carrier $F = (K, \mathcal{P}, R, A, B, L, H, C, Q, N)$. The compact source row K and the finite bundle $\mathcal{P}$ are the only coverage rows available.

Read the candidate family R. Each candidate is a positive $\mathsf{Rat}^{\uparrow}$ row attached to a bundled probe, so the fold cannot inspect an infinite cover or a center outside the displayed bundle.

The accumulator row A and certificate row B provide the lower-bound fold. They are transported and replayed by H and C, and then sourced and named by Q and N.

The handoff row L is therefore a $\mathsf{UniformModulus}^{\uparrow}$ precision read over that finite packet. Since no excluded host or compactness coordinate is present, every uniform-modulus consumer remains inside the displayed fold route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLowerBoundFold}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ packet is generated from a CompactUp finite coverage row, a nonempty ProbeBundle spine, positive RatUp candidate rows, an accumulator ledger, a certified lower-bound row, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteLowerBoundFold}\uparrow}$, seed)

Scope closed: Definition 4096.1, Definition 4096.2, and Definition 4096.3 seed the finite ProbeBundle rational lower-bound fold used by compact-uniform-modulus consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteLowerBoundFold}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host finite minima, selected centers outside the bundle, open-cover compactness, quotient point equality, external modulus selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bundle admission, rational candidate positivity, accumulator recursion, lower-bound certificate stability, uniform-modulus handoff, and ledger non-escape for BEDC.Derived.FiniteLowerBoundFoldUp.

DNA hash: `aggregate=sha256:382cef63d1023f0ab10fc718fdae684b2ab03e53b1409d98cb6368e9134a7148;statement=sha256:cf`

Chapter 4097

A Concrete Naming Certificate for MetaCICDecidableConversionFrontier$^{\uparrow}$

The MetaCICDecidableConversionFrontier$^{\uparrow}$ packet realizes the bounded conversion boundary in chapter 9379 as finite NameCert data. It sits beside the decidability boundary of chapter 749, the normalization frontier of chapter 892, the bounded normal-conversion bridge of chapter 3439, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet names the exact point where bounded normal equality may be consumed by a conversion-facing checker while the unrestricted closed-term conversion theorem remains outside the carrier.

Falsifiable prediction: Every accepted MetaCICDecidableConversionFrontier$^{\uparrow}$ read exposes metacicdecidableboundary, metacicnormalizationfrontier, bounded-normal-conversion-bridge, normal-form-consistency-seal, metaciccriticalpath, bounded-checker, finished-normal, obstruction, transport, replay, provenance, and local NameCert rows before any conversion-frontier handoff is consumed.

Independence witness: metacicdecidableboundary, metacicnormalizationfrontier, bounded-normal-conversion-bridge, normal-form-consistency-seal, metaciccriticalpath: the decidability boundary records checker limits without the normal endpoint row; the normalization frontier records candidate and endpoint evidence without the conversion checker; the bridge records bounded normal comparison without the full audit map; the consistency seal records closed normal-form safety without conversion completeness; and the critical path records dependency pressure without this finite conversion-frontier carrier. None is bijective to MetaCICDecidableConversionFrontier$^{\uparrow}$.

Definition 4097.1 (MetaCIC decidable conversion frontier carrier)**.** *A* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, N, B, S, K, Q, O, H, C, P, L)$$

with the following displayed rows:

1. *D is the* MetaCICDecidableBoundary$^{\uparrow}$ *row, recording the typing-checker and same-term decision surface available to the audit map.*
2. *N is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording candidate evidence, finished-normal evidence, endpoint use, and retained obstruction data.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, comparing bounded normalizer outputs under the displayed finished-normal boundary.*

4. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, recording the closed normal-form consistency seal consumed by conversion-facing checks.*
5. *K is the* MetaCICCriticalPath$^{\uparrow}$ *row, retaining the dependency-pressure ledger for conversion, normalization, confluence, and subject-reduction consumers.*
6. *Q is the bounded equality-checker row, recording the compared normal forms, finite fuel budget, and Boolean equality output.*
7. *O is the obstruction row, naming the excluded endpoints: unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, and kernel consistency.*
8. *H records componentwise* hsame *transport for D, N, B, S, K, Q, O and the local naming row.*
9. *C records displayed* Cont *replay from a conversion query through $D, N, B, S, K, Q, O, H, P, L$.*
10. *P is the* Pkg *provenance over* MetaCICDecidableConversionFrontier$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{MetaCICDecidableConversionFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, N, B, S, K, Q, O, H, C, P, L$. It has no coordinate for unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICDecidableConversionFrontierUp.MetaCICDecidableConversionFrontierTasteGate`
`single carrier alignment`

Theorem 4097.2 (MetaCIC decidable conversion frontier NameCert obligations)**.** *For every accepted* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier*

$$F = (D, N, B, S, K, Q, O, H, C, P, L),$$

the local NameCert$_{\text{MetaCICDecidableConversionFrontier}^{\uparrow}}$ *surface consists exactly of decidability-boundary exposure through D, normalization-frontier exposure through N, bounded normal-conversion bridge exposure through B, normal-form seal exposure through S, critical-path exposure through K, bounded equality-checker exposure through Q, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 4097.1. Rows D, N, B, S, K are the sibling audit and bridge surfaces, while Q is the local bounded checker row and O keeps the excluded endpoints visible.

The conversion-facing read must pass through the bounded bridge and the normal-form seal together. The bridge supplies only bounded normal comparison, and the seal supplies the closed normal-form safety row; neither row contains arbitrary-fuel completeness or unrestricted beta-conversion.

Finally H, C, P, L transport, replay, source, and name the same packet. Since no carrier coordinate contains evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 4097.3 (MetaCIC decidable conversion frontier bounded route)**.** *Every conversion-facing consumer of an accepted* MetaCICDecidableConversionFrontier$^{\uparrow}$ *carrier factors through*

MetaCICDecidableBoundary$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, NormalFormConsi

and then through the local bounded equality-checker row Q with obstruction retention O. The route exports a bounded conversion frontier only; it does not prove unrestricted closed-term beta-conversion or arbitrary-fuel completeness.

Proof. Start with Definition 4097.2. It forces the consumer to read the decidability boundary, normalization frontier, bounded conversion bridge, consistency seal, and critical-path rows before using the local checker row.

The local checker row Q is interpreted under the obstruction row O. Thus a positive bounded comparison can be replayed as finite NameCert data, but the retained obstruction prevents it from being cited as an unrestricted conversion theorem.

Transport and replay through H and C, with provenance and naming through P and L, preserve that boundary. No sibling route adds arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency to the carrier. □

Theorem 4097.4 (MetaCIC decidable conversion frontier bounded search)**.** *Every positive conversion-frontier read exported by an accepted* $\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}$ *carrier factors through the displayed finite fuel budget in Q, the bounded normalizer surface in N, the* $\mathsf{Term}.eq$ *comparison encoded by the bounded equality-checker row, the* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *row B, and the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *row S. The read is therefore a finite search certificate, not an unrestricted conversion theorem.*

Proof. Project the carrier of Definition 4097.1. The only local row that can report a positive comparison is Q, and that row stores the compared normal forms, finite fuel budget, and Boolean equality output.

Apply the NameCert obligations of Definition 4097.2. They force every conversion-facing read to expose the decidability boundary, normalization frontier, bounded bridge, normal-form seal, critical-path row, checker row, and obstruction row together.

Use the bounded route theorem Definition 4097.3. It places the positive read behind D, N, B, S, K and then behind Q, so the equality output is interpreted only through bounded normalization, $\mathsf{Term}.eq$ comparison, the bridge row, and the consistency seal.

The obstruction row O remains part of the same packet. Since unrestricted closed beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency are not carrier coordinates, a positive bounded read cannot be promoted to any of them. □

Theorem 4097.5 (MetaCIC decidable conversion frontier refusal)**.** *The same* $\mathsf{MetaCICDecidableConversionFrontier}$ *carrier refuses unrestricted closed beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency. Any consumer claiming one of those endpoints must supply data outside the displayed rows $D, N, B, S, K, Q, O, H, C, P, L$.*

Proof. Read the obstruction row O from Definition 4097.1. It explicitly names unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, and kernel consistency as excluded endpoints, while the classifier also omits subject reduction and confluence.

Apply Definition 4097.4. A positive conversion-frontier read is only a bounded search certificate passing through finite fuel, bounded normalization, checker comparison, bridge, and seal rows.

Now use Definition 4097.3. The route preserves obstruction retention through O and structural replay through H, C, P, L, so no later consumer can erase the finite-search boundary inside this packet.

Thus the refused endpoints require additional carriers or theorems not present here. The local NameCert surface exports a bounded decidability frontier only. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}$ packet is generated from MetaCICDecidableBoundary, MetaCICNormalizationFrontier, BoundedNormalConversionBridge, NormalFormConsistencySeal, MetaCICCriticalPath, bounded equality-checker, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4097.1, Definition 4097.2, and Definition 4097.3 seed the finite bounded-conversion frontier for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDecidableConversionFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted closed-term beta-conversion, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-checker exposure, normalization-frontier coverage, bounded bridge exactness, normal-form seal compatibility, critical-path retention, obstruction nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDecidableConversionFrontierUp.

Chapter 4098

A Concrete Naming Certificate for MetaCICClosednessDecisionBoundary$^{\uparrow}$

The MetaCICClosednessDecisionBoundary$^{\uparrow}$ packet realizes the closedness-decision warning of chapter 9379 as finite NameCert data. It sits beside the decidability audit packet of chapter 746, the bounded normal-equality checker of chapter 752, the bounded fuel trace of chapter 3889, and the closed type-checker seal of chapter 2182. The packet names the boundary where closedness evidence can be consumed without exporting a theorem of the form `Decidable (ClosedAt n t)` or unrestricted closed-term beta-conversion decidability.

Falsifiable prediction: A consumer of MetaCICClosednessDecisionBoundary$^{\uparrow}$ cannot derive unrestricted closed-term beta-conversion decidability from this packet alone; every accepted read exposes decidability-audit, bounded-normal-equality, fuel-trace, closed-type-checker, closedness-boundary, refusal, transport, replay, provenance, and local NameCert rows before any conversion-facing handoff.

Independence witness: metacicdecidableauditpacket, boundednormalequalitychecker, metacicfueltrace, closedtypecheckerseal: the audit packet records checker scope without this closedness boundary row; the bounded normal-equality checker records finite fuel comparison without a global closedness decision; the fuel trace records bounded normalizer fuel without unrestricted conversion completeness; and the closed type-checker seal records checked typing without a standalone closedness decider. None is bijective to MetaCICClosednessDecisionBoundary$^{\uparrow}$.

Definition 4098.1 (MetaCIC closedness decision boundary carrier)**.** *A* MetaCICClosednessDecisionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, Q, F, S, Z, O, H, C, P, N)$$

with the following displayed rows:

1. *A is the* MetaCICDecidableAuditPacket$^{\uparrow}$ *row exposing the checked typing and bounded conversion audit surface.*
2. *Q is the* BoundedNormalEqualityChecker$^{\uparrow}$ *row exposing bounded normal-form comparison.*
3. *F is the* MetaCICFuelTrace$^{\uparrow}$ *row exposing the finite fuel budget and bounded normalizer endpoint data.*
4. *S is the* ClosedTypeCheckerSeal$^{\uparrow}$ *row exposing closed typing-checker reads.*

5. *Z is the closedness-boundary row naming the available finite closedness evidence and the absence of a global* `Decidable (ClosedAt n t)` *export.*
6. *O is the refusal row for unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency.*
7. *H records componentwise* hsame *transport for A, Q, F, S, Z, O and the local naming row.*
8. *C records displayed* Cont *replay from a closedness-facing query through $A, Q, F, S, Z, O, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$, $\mathsf{MetaCICDecidableAuditPacket}^{\uparrow}$, $\mathsf{BoundedNormalEqualityChecker}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{ClosedTypeCheckerSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, Q, F, S, Z, O, H, C, P, N$. It has no coordinate for unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICClosednessDecisionBoundaryUp.MetaCICClosednessDecisionBoundaryTasteGate single carrier alignment`

Theorem 4098.2 (MetaCIC closedness decision boundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ *carrier*

$$B = (A, Q, F, S, Z, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$ *surface consists exactly of decidability-audit exposure through A, bounded normal-equality exposure through Q, finite fuel-trace exposure through F, closed type-checker seal exposure through S, closedness-boundary exposure through Z, refusal retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4098.1. Rows A, Q, F, S are the sibling checker and bounded-computation surfaces, while Z records the closedness boundary and O records the refused endpoints.

A closedness-facing read must pass through the audit packet, bounded checker, fuel trace, and closed type-checker seal before it can use Z. These rows provide finite checker evidence, but none contains a global `Decidable (ClosedAt n t)` theorem.

Finally H, C, P, N transport, replay, source, and name the same boundary packet. Since no carrier coordinate contains unrestricted beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 4098.3 (MetaCIC closedness decision boundary non-escape)**.** *Every consumer of an accepted* $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ *carrier can cite only the finite route*

$$\mathsf{MetaCICDecidableAuditPacket}^{\uparrow},\ \mathsf{BoundedNormalEqualityChecker}^{\uparrow},\ \mathsf{MetaCICFuelTrace}^{\uparrow},\ \mathsf{ClosedTypeCheckerSeal}^{\uparrow},\ Z,$$

The route preserves checked typing, bounded normal comparison, finite fuel tracing, and closed type-checker sealing while refusing unrestricted closed-term beta-conversion decidability and a standalone global closedness decision.

Proof. Apply Definition 4098.2. It forces every consumer to read the audit, bounded-checker, fuel-trace, and closed-type-checker rows before reading the local closedness-boundary row.

The refusal row O is part of the same carrier. It records that the packet does not export unrestricted closed-term beta-conversion decidability, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Transport and replay through H and C, with provenance and naming through P and N, preserve that boundary. The consumer therefore receives a finite closedness-decision boundary packet, not a global decision theorem. □

The Lean follow-up supplies Nontrivial and FieldFaithful instances in `lean4/BEDC/Derived/MetaCICClosedness` by exhibiting distinct accepted and refused boundary packets and by matching on the rows $A, Q, F, S, Z, O, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}$ packet is generated from MetaCICDecidableAuditPacket, BoundedNormalEqualityChecker, MetaCICFuelTrace, ClosedTypeCheckerSeal, closedness-boundary, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4098.1, Definition 4098.2, and Definition 4098.3 seed the finite closedness-decision boundary for MetaCIC decidability consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICClosednessDecisionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted closed-term beta-conversion decidability, a global `Decidable (ClosedAt n t)` theorem, arbitrary-fuel completeness, evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, audit-packet exposure, bounded-normal checker exposure, fuel-trace exposure, closed-type-checker seal compatibility, closedness-boundary nonescape, refusal retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICClosednessDecisionBoundaryUp.

Chapter 4099

A Concrete Naming Certificate for $\mathsf{EffectiveCauchySpace}^{\uparrow}$

Following Bishop–Bridges constructive analysis and represented-space computable analysis, the $\mathsf{EffectiveCauchySpace}^{\uparrow}$ packet records a represented metric source whose Cauchy access is mediated by displayed moduli and finite observation windows. It sits after the represented-space source of chapter 1494, the metric carrier of chapter 57, the effective modulus surface of chapter 3518, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the unit handoff of chapter 1527. The chapter gives Real and Completion consumers a finite route from represented names and effective Cauchy moduli to the current L10 completion boundary; it is not quotient completion equality, a selected limit, countable choice, host equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4099.1 (Effective Cauchy space carrier)**.** *An* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row, naming the represented input surface and the representation-mediated observations available to the packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row, recording the point classifier, distance requests, and metric comparison data read over the same represented source.*
3. *E is the* $\mathsf{EffectiveCauchyModulus}^{\uparrow}$ *row, supplying the finite stage schedule by which metric tolerance requests become Cauchy-window requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, displaying the finite observation windows opened by the represented source and the effective modulus.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows after the metric and modulus rows have fixed the tolerance request.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing dyadic radii, slack, comparison budgets, and tolerance conversions used by E, S, R.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the represented source, metric row, effective modulus, stream schedule, regular readback, and dyadic ledger have been displayed.*

8. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row admitting the same represented Cauchy data into the completion-facing unit route.*
9. *H records componentwise* hsame *transport for A, M, E, S, R, D, L, U and the local naming row.*
10. *C records displayed* Cont *replay from a Real or Completion request through the represented source, metric row, modulus, finite windows, readback, tolerance ledger, seal, unit handoff, transport, provenance, and local naming rows.*
11. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchySpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{EffectiveCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, E, S, R, D, L, U, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, trichotomy, host equality of completed objects, ambient completeness of $\mathsf{Real}^{\uparrow}$*, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.EffectiveCauchySpaceUp.EffectiveCauchySpaceTasteGate single carrier alignment`

Theorem 4099.2 (Effective Cauchy space NameCert obligations)**.** *For every accepted* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *represented-source admission through A and metric-source admission through M over the same displayed carrier;*
2. *effective-modulus exposure through E before any finite stream window is opened;*
3. *StreamName schedule exactness through S, requiring every inspected name window to be one of the displayed finite windows;*
4. *regular rational readback through R only after the represented source, metric row, modulus, stream schedule, and dyadic ledger have been displayed;*
5. *dyadic tolerance accounting through D for every radius, slack, comparison budget, and tolerance conversion consumed by E, S, R;*
6. *terminal Real sealing through L and completion-unit handoff through U, both downstream of A, M, E, S, R, D;*
7. *transport, replay, provenance, and local naming through H, C, P, N without adding a second classifier or hidden completion coordinate.*

No effective-Cauchy-space obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4099.1. The represented source row A and the metric row M are the only source coordinates, so every downstream read is tied to the same represented metric carrier.

The effective modulus row E mediates every tolerance request before the finite stream schedule S is opened. The dyadic ledger D supplies the radius and slack accounting needed by the modulus and schedule rows, and the regular rational readback R is downstream of those displayed data.

The terminal routes L and U are read only after A, M, E, S, R, D have been exposed. Thus the Real seal and the CauchyCompletionUnit handoff consume the same finite represented-Cauchy route rather than a quotient completion, selected limit, or host equality datum.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no row for countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 4099.3 (Effective Cauchy space completion handoff)**.** *Every Real-facing or completion-facing consumer of an accepted* $\mathsf{EffectiveCauchySpace}^{\uparrow}$ *carrier*

$$X = (A, M, E, S, R, D, L, U, H, C, P, N)$$

factors through the displayed route

$$\mathsf{RepresentedSpace}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{EffectiveCauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyCo}$$

with transport, replay, provenance, and local naming supplied only by H, C, P, N*. The consumer receives no quotient completion equality, selected limit, countable-choice selector, host equality, trichotomy branch, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Fix a Real-facing or completion-facing consumer of X. The carrier exposes the represented source, metric row, effective modulus, stream schedule, dyadic ledger, regular readback, Real seal, and completion-unit handoff as the only analytic coordinates.

The consumer first reads A and M, then obtains the tolerance stage from E. The corresponding finite windows are displayed in S, with all radius and slack conversions accounted for in D.

Only after that finite observation route is fixed can the readback row R be consumed. The Real seal L and the CauchyCompletionUnit handoff U are downstream of the same readback and ledger, so both terminal routes inherit the represented Cauchy data rather than selecting an external limit.

The structural rows H, C, P, N preserve the route and introduce no additional analytic coordinate. Therefore every allowed consumer factors through the displayed completion handoff and cannot extract any of the excluded quotient, choice, equality, trichotomy, or ambient-completeness data. □

Falsifiable prediction: Every accepted $\mathsf{EffectiveCauchySpace}^{\uparrow}$ Real or Completion read exposes the RepresentedSpace source, MetricSpace row, EffectiveCauchyModulus row, StreamName schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, CauchyCompletionUnit handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows before the terminal handoff is consumed.

Independence witness: representedspace, metric, effective-cauchy-modulus, streamname, regseqrat, dyadicratcore, real, cauchycompletionunit: $\mathsf{EffectiveCauchySpace}^{\uparrow}$ is not bijective to any listed sibling because it binds a represented metric source, effective Cauchy modulus, finite schedule, tolerance ledger, regular readback, Real seal, completion-unit handoff, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4099.1, Definition 4099.2, and Definition 4099.3 seed the finite represented-metric Cauchy route from effective modulus data to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchySpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host equality, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, represented-source and metric-source agreement, effective-modulus exposure, finite-window exactness, dyadic tolerance accounting, regular readback, Real seal boundary, completion-unit handoff, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchySpaceUp.

Constructive story: An EffectiveCauchySpace$^{\uparrow}$ packet is generated from a represented-space source, a metric row, an effective Cauchy modulus, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4100

A Concrete Naming Certificate for $\mathsf{RegularCauchyRegularization}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyRegularization}^{\uparrow}$ packet records the finite BEDC route by which arbitrary Cauchy data are tightened into the regular-rational route used by $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits between chapter 291, chapter 126, chapter 301, chapter 38, and the Cauchy-criterion surfaces such as chapter 392. The packet is a regularization service over displayed finite rows; it is not quotient stream equality, a selected limit, trichotomy, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 4100.1 (Regular Cauchy regularization carrier)**.** *A* $\mathsf{RegularCauchyRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (Q, D, S, A, E, H, C, P, N)$$

with the following rows:

1. *Q is the raw Cauchy source row, naming the displayed rational data and its Cauchy-window evidence through a local Cauchy-criterion certificate;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tightening ledger, recording the dyadic precision scale used to replace arbitrary Cauchy requests by nested regular tolerances;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that lists the finite read windows consumed by the regularization;*
4. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the tightened schedule and dyadic ledger;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after the regular readback row has been displayed;*
6. *H records componentwise* hsame *transport for the source, dyadic ledger, schedule, regular readback, terminal seal, replay, provenance, and naming rows;*
7. *C records the displayed* Cont *replay from Q through D, S, A to E;*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting Cauchy streams, selecting a host limit, comparing host real values by trichotomy, or importing completed-real closure.

Lean4 ✓ `BEDC.Derived.RegularCauchyRegularizationUp.RegularCauchyRegularizationTasteGate single carrier alignment`

Theorem 4100.2 (Regular Cauchy NameCert obligations)**.** *For the carrier of Definition 4100.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$ *surface consists exactly of carrier admission for* $R = (Q, D, S, A, E, H, C, P, N)$*, dyadic tightening from the raw Cauchy source row to the scheduled regular windows,* $\mathsf{StreamName}^{\uparrow}$ *schedule stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback exactness,* $\mathsf{Real}^{\uparrow}$ *terminal-seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and nonescape for quotient stream equality, selected host limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the finite packet. The rows $Q, D, S, A, E, H, C, P, N$ are the whole carrier, so the admitted regularization service must expose the raw Cauchy source, the dyadic tightening ledger, the stream schedule, the regular rational readback, and the terminal seal as displayed data.

The Cauchy source is consumed through the local Cauchy-criterion row, while D chooses the dyadic precision budget used for each scheduled request. The schedule row S then records only finitely many read windows, and A packages those windows as the regular rational readback used by $\mathsf{RegSeqRat}^{\uparrow}$.

Transport and replay are componentwise. The hsame row H preserves the source, ledger, schedule, readback, seal, route, provenance, and local name together, while the Cont row C forces every consumer to traverse Q then D, S, A before reaching E.

Since the packet contains no quotient-stream coordinate, selected host limit, trichotomy row, choice row, or ambient completeness row, no such datum can be projected from the local NameCert surface. □

Theorem 4100.3 (Regular Cauchy modulus route)**.** *Every accepted* $\mathsf{RegularCauchyRegularization}^{\uparrow}$ *carrier turns a raw Cauchy source row into a* $\mathsf{RegSeqRat}^{\uparrow}$ *route by first tightening the source through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger, then scheduling the required windows through* $\mathsf{StreamName}^{\uparrow}$*, then reading the scheduled rational data as the regular row* A*. The* $\mathsf{Real}^{\uparrow}$ *seal* E *is available only through that displayed route, and no consumer may replace the route by quotient stream equality, a selected limit, trichotomy, countable choice, or ambient completeness.*

Proof. Begin with the carrier inventory of Definition 4100.1. The only source of tolerance control is D, and the only source of read order is S. Thus the raw Cauchy row cannot be consumed as a regular sequence until the dyadic ledger and the stream schedule have both been displayed.

Next read the regular row. The packet places A after D and S, so the $\mathsf{RegSeqRat}^{\uparrow}$ consumer sees regular rational data only through the tightened windows. The terminal $\mathsf{Real}^{\uparrow}$ seal E is reached after this readback, not directly from the raw source row.

Finally apply the nonescape clauses from the NameCert obligations. Because no quotient stream, selected host limit, trichotomy, countable-choice coordinate, or ambient completeness row occurs in R, the modulus route is the displayed finite route through $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4100.1, Definition 4100.2, and Definition 4100.3 seed the finite regularization surface from raw Cauchy source rows through dyadic tightening, StreamName scheduling, RegSeqRat readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRegularization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tightening exactness, schedule stability, regular readback exactness, Real seal exposure, and nonescape for BEDC.Derived.RegularCauchyRegularizationUp.

Constructive story: A RegularCauchyRegularization$^{\uparrow}$ packet is generated from a raw Cauchy source row, a DyadicRatCore$^{\uparrow}$ tightening ledger, a StreamName$^{\uparrow}$ schedule, a RegSeqRat$^{\uparrow}$ readback row, a Real$^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4101

A Concrete Naming Certificate for $\mathsf{ReflexiveBanach}^{\uparrow}$

Following Banach and Alaoglu, the $\mathsf{ReflexiveBanach}^{\uparrow}$ packet records the finite double-dual evaluation window that functional-analysis consumers may inspect before any weak compactness conclusion is read. It sits beside the finite weak-star dual-ball route of chapter 3420, the weak observation surface of chapter 3597, the norm ledger of chapter 89, the Hilbert comparison surface of chapter 91, and the functional-analysis carrier of chapter 94. The packet names only displayed Banach, norm, dual-window, double-dual evaluation, weak-topology, bounded-ball, transport, replay, provenance, and local naming rows; it does not assert full Eberlein–Smulian, select weak cluster points, import quotient dual equality, or use choice over all bounded nets.

Definition 4101.1 (ReflexiveBanach finite double-dual carrier)**.** *A* $\mathsf{ReflexiveBanach}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, N, D, E, W, A, H, C, P, L)$$

with the following displayed rows:

1. *B is the Banach carrier row, read through the displayed* $\mathsf{Banach}^{\uparrow}$ *and* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *surfaces.*
2. *N is the norm row, recording the finite norm estimates used by source, dual, and ball consumers through* $\mathsf{Norm}^{\uparrow}$*.*
3. *D is the dual-window row, listing the finitely many bounded-linear-functional observations allowed in the current packet.*
4. *E is the double-dual evaluation row, recording the canonical evaluation of displayed source endpoints against the functionals named by D.*
5. *W is the weak-topology readback row, sending those finite scalar evaluations to the* $\mathsf{WeakTopology}^{\uparrow}$ *observation surface.*
6. *A is the bounded-ball ledger, recording the finite norm-bound rows whose weak-compactness handoff is visible to later consumers.*
7. *H records componentwise* hsame *transport for the Banach carrier, norm row, dual windows, double-dual evaluations, weak readbacks, bounded-ball ledgers, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from Banach admission through norm control, dual-window evaluation, weak readback, and bounded-ball handoff.*
9. *P is the* Pkg *provenance over* ReflexiveBanach$^\uparrow$, Banach$^\uparrow$, Norm$^\uparrow$, WeakTopology$^\uparrow$, BanachAlaoglu$^\uparrow$, Hilbert$^\uparrow$, FunctionalAnalysis$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{ReflexiveBanach}^\uparrow}$ *row.*

The carrier has no coordinate for a completed host double dual, quotient equality of dual functionals, a selected weak cluster point, a countable-choice family of balls, or an unrestricted compactness theorem.

Theorem 4101.2 (ReflexiveBanach NameCert obligations)**.** *For every accepted* ReflexiveBanach$^\uparrow$ *carrier $R = (B, N, D, E, W, A, H, C, P, L)$, the local* NameCert$_{\text{ReflexiveBanach}^\uparrow}$ *surface consists exactly of carrier admission for B, double-dual evaluation stability for E, weak topology compatibility for W, bounded-ball readback for A, and no-choice ledger nonescape for the structural rows H, C, P, L. Every accepted read must expose the Banach carrier, norm row, finite dual window, evaluation row, weak readback row, and bounded-ball ledger before local naming is consumed.*

Lean4 ✓ `BEDC.Derived.ReflexiveBanachUp.ReflexiveBanachTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4101.1. The rows $B, N, D, E, W, A, H, C, P, L$ are the complete visible inventory, so carrier admission and local naming can only use those coordinates.

The double-dual evaluation obligation is finite. Each evaluation in E is tied to a source endpoint from B and a bounded functional from D, with norm control read through N; componentwise transport through H preserves the same displayed evaluation row.

Weak topology compatibility follows by reading W through the finite scalar observation surface of chapter 3597. The bounded-ball ledger A is downstream of the same norm and evaluation rows, and the Banach–Alaoglu sibling supplies only the finite dual-ball window cited in the provenance row.

Finally C, P, L replay, source, and name that same packet. Since no row stores a selected cluster point, quotient dual equality, host double-dual equality, or a choice family over all bounded balls, the no-choice ledger cannot escape the displayed finite route. □

Theorem 4101.3 (ReflexiveBanach weak compactness handoff)**.** *Every weak-compactness-facing consumer of an accepted* ReflexiveBanach$^\uparrow$ *packet must factor through the finite route*

$$B, N, D, E, W, A \longmapsto H, C, P, L.$$

The consumer may cite BanachAlaoglu$^\uparrow$ *only through the finite dual-window provenance, may cite* WeakTopology$^\uparrow$ *only through the scalar readback row W, and may cite* Hilbert$^\uparrow$ *or* FunctionalAnalysis$^\uparrow$ *only through the displayed source and norm rows. It receives no selected weakly convergent subnet, no quotient dual-space equality, no host compactness theorem, and no choice-based traversal of the bounded ball.*

Proof. Begin with the bounded-ball ledger. The row A is read after the Banach carrier B and norm row N, so every ball-facing request is already confined to displayed norm estimates over the accepted carrier.

Next expose the dual-window and evaluation rows. The row D lists only finitely many bounded-functional observations, and E evaluates displayed source endpoints against those observations before the weak-topology row W can be consumed.

The weak readback is therefore a finite scalar route. By chapter 3597 and chapter 3420, downstream compactness vocabulary can only repackage the finite weak and weak-star windows already present in D, E, W, A.

Transport, replay, provenance, and local naming preserve this route through H, C, P, L. The absence of selector, quotient, host compactness, and choice-family coordinates prevents the handoff from becoming a nonconstructive weak compactness principle. □

Falsifiable prediction: Every accepted $\mathsf{ReflexiveBanach}^{\uparrow}$ packet exposes a Banach carrier, norm row, finite dual-window row, double-dual evaluation row, weak-topology readback row, bounded-ball ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before any weak compactness consumer can read the packet.

Independence witness: banach-alaoglu, weak-topology, norm, hilbert, functionalanalysis: $\mathsf{ReflexiveBanach}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries double-dual evaluation rows and bounded-ball weak-readback ledgers over a Banach source while refusing selected weak cluster points and quotient dual equality.

Closure status for $\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}$.

Constructive story: A $\mathsf{ReflexiveBanach}^{\uparrow}$ packet is generated from a Banach carrier, norm row, finite dual-window row, double-dual evaluation row, weak-topology readback row, bounded-ball ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4101.1, Definition 4101.2, and Definition 4101.3 seed the finite double-dual and bounded-ball weak-readback certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReflexiveBanach}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Eberlein–Smulian, selected weak cluster points, quotient dual equality, host double-dual equality, full weak compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, double-dual evaluation stability, weak topology compatibility, bounded-ball readback, no-choice ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ReflexiveBanachUp.

Chapter 4102

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ packet records a regular Cauchy filter space as finite NameCert data for the L10 completion route. It sits between the Cauchy filter carrier of chapter 333, the regular Cauchy filter rows of chapter 1445, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real seal of chapter 38. The packet feeds completion consumers by displaying regular filter-space rows before any quotient of filters, selected limit, or countable-choice principle can be invoked.

Definition 4102.1 (Regular Cauchy filter space carrier)**.** *A* $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, B, W, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the admitted point and basic-filter carrier row, retaining classifier-distinct point and filter names without quotienting them by convergence or equality;*
2. *B is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, exposing the finite filter-membership and directedness observations consumed by the packet;*
3. *W is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *regularity window, recording the shrinking, containment, and Cauchy-modulus rows used to read the filter as regular;*
4. *Q is the classifier/window exactness row, tying every accepted window request to the same point and filter classifier rows in X, B, W;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row for finite request windows;*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for regular window moduli and Cauchy bounds;*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *sealing row reached only after S, R, W have displayed the finite regular-filter readback;*
8. *H records componentwise* hsame *transport for all carrier, basis, regularity, exactness, window, readback, seal, replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay through the same finite route;*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, a selected representative filter, a selected limit point, countable choice, ambient separated completion, or host equality of point names.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterSpaceUp.RegularCauchyFilterSpaceTasteGate single carrier alignment`

Theorem 4102.2 (Regular Cauchy filter space NameCert obligations)**.** *For every accepted* RegularCauchyFilterSpac *carrier*

$$F = (X, B, W, Q, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the point/filter carrier X, Cauchy filter basis B, regularity window W, classifier/window exactness row Q, stream windows S, regular rational readback R, real seal E, transport H, replay C, provenance P, and local naming N;*
2. *classifier/window exactness, requiring every filter-space read to use the displayed classifier rows of X, B, W and the same finite request windows recorded in Q, S;*
3. *stability and transport, requiring regularity, Cauchy bounds, rational readback, and real sealing to move only componentwise through H and to replay through C;*
4. *ledger nonescape, excluding quotient filters, host equality, selected representatives, selected limits, countable choice, and ambient separated completion from the packet;*
5. *completion-consumer handoff, exporting only $X, B, W, Q, S, R, E, H, C, P, N$ to a* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer before that consumer adds its own completion rows.*

No other NameCert obligation is available from the regular Cauchy filter space packet.

Proof. Carrier admission is the projection of Definition 4102.1. The tuple lists the point/filter carrier, Cauchy filter basis, regularity window, exactness row, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Classifier/window exactness is carried by Q and S. A consumer first fixes the point and filter classifiers in X, B, W, then reads the finite request windows through Q, S before asking for rational or real data.

Stability and transport are componentwise. The row H transports the displayed coordinates, and the row C replays the same route for consumers, so regularity and Cauchy bounds cannot be replaced by host equality or by a detached separated-completion fact.

Ledger nonescape follows from the carrier inventory. There is no coordinate for quotient filter equality, selected representatives, selected limits, countable choice, or ambient separated completion.

Finally the completion handoff is only the displayed tuple $X, B, W, Q, S, R, E, H, C, P, N$. A $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer may use that finite regular-filter-space surface as input, but any completion object, quotient, or limit belongs to the consumer packet rather than to this carrier. □

Falsifiable prediction: A consumer that claims a $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ handoff while omitting the Cauchy filter basis, regularity window, classifier/window exactness row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 4102.1.

Independence witness: cauchyfilter, regularcauchyfilter, streamname, regseqrat, real: $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ binds the filter basis, regularity window, finite request windows, rational readback, real sealing,

transport, replay, provenance, and local naming rows in one completion-facing packet; each named sibling lacks at least one of these rows as a local filter-space coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ packet is generated from a point/filter carrier, CauchyFilter basis row, RegularCauchyFilter regularity window, classifier/window exactness row, StreamName finite windows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4102.1 and Definition 4102.2 seed the finite regular-filter-space surface that can feed Cauchy filter completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFilterSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected representatives, selected limits, countable choice, ambient separated completion, Lean verification, bridge checking, or host equality of point names.

Upgrade path: Obligation closure requires checked carrier admission, classifier/window exactness, componentwise stability, ledger nonescape, and completion-consumer handoff for BEDC.Derived.RegularCauchyFilterSpaceUp.

Chapter 4103

A Concrete Naming Certificate for $\mathsf{CauchyRealLimit}^{\uparrow}$

Following Bishop and Bridges, a Cauchy real limit is treated here as the finite handoff from a regular Cauchy presentation to a sealed real name. The $\mathsf{CauchyRealLimit}^{\uparrow}$ packet sits after the regular rational readback surface of chapter 301, the stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the filter-facing limit route of chapter 1558. Its role is to expose the L10 real-completion limit handoff without importing quotient streams, selected representatives, host limits, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4103.1 (Cauchy-real-limit carrier)**.** *A* $\mathsf{CauchyRealLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the Cauchy-modulus row that assigns each requested tolerance to a finite tail-window obligation.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row displaying the inspected windows of the Cauchy presentation.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational approximants observed through S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the observed windows with the requested modulus budget.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular readback and dyadic tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for the modulus, stream, regular readback, dyadic tolerance, real seal, and local certificate rows.*
7. *C records displayed* Cont *replay from a tolerance request through M, S, R, D to the* $\mathsf{Real}^{\uparrow}$ *seal E.*
8. *P records* Pkg *provenance over* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, an externally selected limit, host real convergence, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.CauchyRealLimitUp.CauchyRealLimitTasteGate single carrier alignment`

Theorem 4103.2 (Cauchy-real-limit NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLimit}^{\uparrow}$ *carrier*

$$L = (M, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Cauchy modulus, stream windows, regular rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming rows;*
2. *tail-window coverage, requiring every tolerance request admitted by* M *to name a finite window in* S *before a limit read is exported;*
3. *dyadic tolerance exactness, requiring every comparison used by the handoff to read exactly the row* D *beside the* $\mathsf{RegSeqRat}^{\uparrow}$ *observations in* R*;*
4. *real-seal handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *seal* E *to be consumed only after* M, S, R, D *have been displayed;*
5. *ledger non-escape, excluding quotient streams, selected limits, host convergence, choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Project the carrier coordinates. The rows M, S, R, D, E display the Cauchy modulus, finite stream windows, regular rational observations, dyadic tolerance ledger, and terminal real seal before any limit-facing consumer can read the packet.

The tail-window and tolerance clauses are finite. A tolerance request first consults M, then reads the scheduled window in S, observes the regular rational data in R, and checks the dyadic budget in D.

Finally attach the seal and structural rows. The row E is reachable only after the displayed readback route, while H, C, P, N transport, replay, source, and name that same route. Since no further coordinate is present, the carrier cannot export quotient stream equality, a selected limit, host convergence, choice, or ambient real completeness. □

Falsifiable prediction: A $\mathsf{CauchyRealLimit}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ limit seal without reading the displayed Cauchy modulus, StreamName windows, RegSeqRat observations, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, cauchyfilterlimit: no listed sibling supplies the whole Cauchy-real-limit packet because each sibling contributes only readback, windowing, tolerance, sealing, or filter-facing limit data while this chapter binds the finite handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyRealLimit}^{\uparrow}$ packet is generated from a Cauchy modulus row, StreamName windows, RegSeqRat regular rational readbacks, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$, seed)

Scope closed: Definition 4103.1 and Definition 4103.2 seed the finite real-limit handoff over Cauchy modulus, stream-window, regular-readback, dyadic-tolerance, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyRealLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real convergence, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, tail-window coverage, dyadic tolerance exactness, Real seal handoff, and ledger non-escape.

Chapter 4104

A Concrete Naming Certificate for $\mathsf{PolishBaireFixedPoint}^{\uparrow}$

Following Banach's fixed-point theorem and Baire's complete-metric category theorem, the $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ packet records the finite route by which a complete separable metric carrier, a Baire surface, and a contraction-iteration ledger are read together before any endpoint is treated as a host object. It routes through the complete-separable carrier of chapter 1490, the ultrametric Baire window of chapter 3485, the complete-metric Baire surface of chapter 3535, the Picard step route of chapter 395, and the finite iteration packet of chapter 2383. The chapter names the displayed fixed-point route without claiming open-cover compactness, countable choice, a selected host fixed point, or any fixed-point primitive outside the carried rows.

Definition 4104.1 (Polish-Baire fixed-point carrier)**.** *A* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, B, U, C, I, E, H, R, G, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row exposing the complete separable metric carrier, stream-name schedule, regular rational readback, observation ledger, transport, replay, provenance, and local name.*
2. *B is the* $\mathsf{BaireUltrametric}^{\uparrow}$ *row that records the finite nested-window Baire route consumed by the packet.*
3. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *row joining a dense-open schedule, complete-metric Cauchy thread, metric carrier, finite windows, rational readback, and terminal real seal.*
4. *C is the* $\mathsf{PicardContraction}^{\uparrow}$ *row carrying the one-step contraction route over the displayed metric source.*
5. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *row carrying the finite orbit ledger, residual budget, completion handoff, transport, replay, provenance, and local name for the same route.*
6. *E is the endpoint-candidate seal row. It is readable only after P, B, U, C, I have been displayed, and it records no selected host point or host fixed-point equality.*
7. *H records componentwise* hsame *transport for the Polish-space, Baire-ultrametric, complete-metric Baire, Picard-contraction, iteration, endpoint-seal, replay, provenance, and local naming rows.*

8. *R records displayed* Cont *replay from a contraction request through the Baire windows, complete-metric surface, Picard step, finite iteration ledger, residual budget, and endpoint-candidate seal.*
9. *G is the* Pkg *provenance over* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurfac}$ $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for open-cover compactness, countable-choice selection, quotient completion equality, host fixed-point equality, or a primitive fixed-point theorem.

Theorem 4104.2 (Polish-Baire fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier*

$$F = (P, B, U, C, I, E, H, R, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission, Polish complete-separable source exposure, Baire-ultrametric window exposure, complete-metric Baire-surface routing, Picard one-step readability, finite fixed-point iteration ledger exposure, endpoint-candidate sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside the rows of Definition 4104.1.*

Lean4 ✓ `BEDC.Derived.PolishBaireFixedPointUp.PolishBaireFixedPointTasteGate single carrier alignment`

Proof. Project the accepted carrier. The rows P, B, U, C, I, E are the only analytic rows available to the local certificate: complete separable source, Baire window, complete-metric Baire surface, Picard step, finite iteration ledger, and endpoint-candidate seal.

The source and Baire obligations are read before any endpoint row. The Polish row supplies the common complete-separable metric surface, while the two Baire rows expose only finite nested-window and complete-metric schedule data.

The fixed-point route is then read through C and I. The Picard row supplies the one-step contraction route, and the iteration row supplies the finite orbit ledger and residual budget without selecting a completed fixed point.

Finally H, R, G, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate beyond these rows, the listed surface exhausts the local NameCert obligations. □

Theorem 4104.3 (Polish-Baire fixed-point iteration nonescape)**.** *Every fixed-point-facing read from an accepted* $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ *carrier must pass through*

$\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, BHist, hsa

In particular, the endpoint-candidate row E is consumable only after the complete-separable source, Baire windows, complete-metric Baire surface, Picard step, finite orbit ledger, transport, replay, provenance, and local naming rows have been displayed.

Proof. Start from the carrier definition. The endpoint-candidate seal E is stored after the Polish source P, Baire rows B, U, Picard route C, and finite iteration ledger I.

Apply the NameCert obligation theorem. It makes the complete-separable source, Baire schedule, Picard step, finite orbit ledger, and residual budget part of the public local surface before the endpoint-candidate seal can be read.

The structural rows H, R, G, N then preserve the same route under transport, replay it for downstream consumers, record provenance, and attach the local name. Because no separate coordinate stores open-cover compactness, countable choice, host fixed-point equality, or a fixed-point primitive, any fixed-point-facing read remains inside the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}$.

Constructive story: A $\mathsf{PolishBaireFixedPoint}^{\uparrow}$ packet is generated from a PolishSpace complete-separable carrier, BaireUltrametric finite window route, CompleteMetricBaireSurface schedule, PicardContraction step, FixedPointIteration ledger, endpoint-candidate seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4104.1, Definition 4104.2, and Definition 4104.3 seed the finite Polish/Baire fixed-point route over complete-separable, Baire-window, complete-metric Baire, Picard, iteration, endpoint-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishBaireFixedPoint}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected host fixed points, quotient completion equality, host fixed-point equality, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Baire-window exposure, complete-metric Baire routing, Picard-step readability, finite iteration ledger exactness, endpoint-candidate nonescape, transport stability, replay stability, provenance stability, and local naming for BEDC.Derived.PolishBaireFixedPointUp.

Chapter 4105

A Concrete Naming Certificate for $\mathsf{CompactModulusSelector}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine–Cantor finite-net proof, the CompactModulusSelector packet records the finite selector route that turns compact-metric total boundedness and pointwise continuity data into a reusable uniform-modulus certificate. It sits over the compact metric source of chapter 129, the total-boundedness finite-net interface of chapter 127, the continuous-map row of chapter 130, and the uniform-modulus consumer of chapter 369. Its L10 real-facing reads pass through the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal seal of chapter 38. The packet names finite-net modulus selection as displayed BEDC data; it does not assert open-cover compactness, countable choice, a selected infinite net, host function equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4105.1 (CompactModulusSelector carrier)**.** *A* $\mathsf{CompactModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, T, F, U, D, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CompactMetric}^{\uparrow}$ *source row. It records the compact metric carrier and its displayed metric observation surface.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row read from K at the requested dyadic tolerance scale.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph and pointwise-continuity row whose local moduli are inspected only at the displayed net centers.*
4. *U is the candidate* $\mathsf{UniformModulus}^{\uparrow}$ *row exported to downstream uniform-continuity consumers.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the source mesh, target error budget, and triangle comparison.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected net-center and point-name windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite metric and image-distance observations.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, W, R have displayed the requested estimates.*

9. *H records componentwise* hsame *transport for compact metric, total-boundedness, continuous-map, uniform-modulus, dyadic, window, readback, seal, continuation, provenance, and naming rows.*
10. *C records displayed* Cont *replay from a uniform-modulus request through the compact finite-net selection route and then into the exported modulus row.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a choice function over all covers, selected infinite subnet data, host equality of maps, trichotomy, quotient completion, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactModulusSelectorUp.CompactModulusSelectorTasteGate single carrier alignment`

Theorem 4105.2 (CompactModulusSelector NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *carrier* $S = (K, T, F, U, D, W, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, total-bounded finite net, continuous-map row, uniform-modulus export, dyadic tolerance ledger, finite windows, regular readback, Real seal, transport, continuation, provenance, and local naming rows. Any consumer that reads the exported modulus must consume the finite-net row T, the pointwise continuity row F, and the dyadic comparison row D before U is visible. The packet exports no open-cover compactness, countable choice, host map equality, selected infinite net, trichotomy, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4105.1. The listed coordinates are exactly the compact metric, total-boundedness, continuous-map, uniform-modulus, dyadic, window, readback, Real-seal, transport, replay, provenance, and local naming rows.

Replay C. A modulus request first reads the compact metric source K, then the finite-net row T, then the continuous-map row F at the displayed net centers. The dyadic ledger D, finite windows W, and regular readback R provide the requested source and target estimates before the terminal seal A is consumed.

The exported row U is therefore a finite-net modulus row. Transport by H preserves only the displayed coordinates, while P and N package and name the same route, so none of the excluded principles appears as a hidden obligation. □

Theorem 4105.3 (CompactModulusSelector finite-net selection)**.** *Let $S = (K, T, F, U, D, W, R, A, H, C, P, N)$ be an accepted* $\mathsf{CompactModulusSelector}^{\uparrow}$ *packet. Every exported* $\mathsf{UniformModulus}^{\uparrow}$ *read factors through the finite route*

$$K, T, F, D, W, R, A \longmapsto U, H, C, P, N.$$

The finite net is read from $\mathsf{TotallyBounded}^{\uparrow}$ *inside the* $\mathsf{CompactMetric}^{\uparrow}$ *source, the local continuity data are read from* $\mathsf{ContinuousMap}^{\uparrow}$ *at the displayed net centers, and the dyadic tolerance ledger supplies the comparison budget. The route preserves all rows $K, T, F, U, D, W, R, A, H, C, P, N$ and exports no open-cover compactness, countable choice, selected infinite net, host equality of functions, trichotomy, quotient completion, or ambient real completeness.*

Proof. Start from a consumer of the uniform-modulus row. Carrier admission exposes U only as a row of the packet named in Definition 4105.1.

Use the continuation row C to replay the finite selection route. The compact metric source K supplies the ambient metric data, T supplies the finite net, and F supplies the local continuity data at the displayed centers.

The tolerance and readback rows D, W, R, A then certify the comparison budget before U is exported. Since transport, provenance, and naming are confined to H, C, P, N, the handoff preserves the listed rows and cannot export any of the excluded choice, open-cover, equality, or completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4105.1, Definition 4105.2, and Definition 4105.3 seed the finite-net modulus selector route over compact metric, total-bounded, continuous-map, uniform-modulus, dyadic, window, readback, Real-seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected infinite nets, host function equality, trichotomy, quotient completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net selection, pointwise-continuity inspection, dyadic tolerance comparison, uniform-modulus export, and structural nonescape for BEDC.Derived.CompactModulusSelectorUp.

Constructive story: A $\mathsf{CompactModulusSelector}^{\uparrow}$ packet is generated from a CompactMetric row, a TotallyBounded finite-net row, a ContinuousMap pointwise-continuity row, a UniformModulus export row, dyadic tolerance data, finite windows, regular readback, a Real seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 4106

A Concrete Naming Certificate for $\mathsf{TypeCheckingClassifier}^{\uparrow}$

The $\mathsf{TypeCheckingClassifier}^{\uparrow}$ packet realizes the type-checking-as-Ext-membership vision of chapter 9367 as one finite NameCert carrier. It sits beside the classifier-membership packet of chapter 476, the typing-classifier route of chapter 489, the subject-reduction surface of chapter 483, the derivation-trace packet of chapter 598, and the proof-erasure route of chapter 642. The packet records type checking as finite membership data: it does not export a host conversion oracle, semantic equality of terms, normalization theorem, hidden proof object, quotient syntax, or ambient metatheory.

Definition 4106.1 (TypeCheckingClassifier carrier)**.** *A* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, J, E, D, S, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the finite term row. It names the inspected term histories and their syntactic classifier.*
2. *J is the typing-ledger row. It records the finite judgment entries used by the checker.*
3. *E is the* Ext*-membership row, identifying accepted typed terms as classifier members.*
4. *D is the derivation-trace row linking each accepted membership read to a displayed typing derivation.*
5. *S is the subject-reduction row, routing accepted* Ext *membership through displayed* Cont *steps.*
6. *R is the erasure row, separating proof-term content from the erased computational term while preserving the displayed membership route.*
7. *H records componentwise* hsame *transport for term, typing-ledger,* Ext*-membership, derivation-trace, subject-reduction, erasure, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a type-checking request through membership, derivation, subject-reduction, and erasure rows.*
9. *P is the* Pkg *provenance over* $\mathsf{TypeCheckingClassifier}^{\uparrow}$, $\mathsf{TypeClassifierMembership}^{\uparrow}$, $\mathsf{ClassifierTyping}^{\uparrow}$, $\mathsf{SubjectReduction}^{\uparrow}$, $\mathsf{TypeCheckingDerivationTrace}^{\uparrow}$, $\mathsf{ProofTermErasureTrace}^{\uparrow}$, BHist, Ext, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for host conversion, semantic equality of terms, an ambient normalization theorem, hidden proof objects, quotient syntax, or external metatheoretic consistency.

Lean4 (definition) `BEDC.Derived.TypeCheckingClassifierUp.TypeCheckingClassifierUp`
Lean4 ✓ `BEDC.Derived.TypeCheckingClassifierUp.TypeCheckingClassifierTasteGate single carrier alignment`

Theorem 4106.2 (TypeCheckingClassifier NameCert obligations)**.** *For every accepted* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *carrier* $Q = (T, J, E, D, S, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the term row, typing ledger,* Ext*-membership row, derivation trace, subject-reduction row, erasure row, componentwise transport, continuation replay, provenance, and local naming. Every accepted type-checking read must expose membership through* E*, derivation through* D*, subject reduction through* S*, and erasure separation through* R *before* H, C, P, N *package the row. No obligation is stored in host conversion, semantic equality of terms, normalization, quotient syntax, hidden proof objects, or ambient metatheory.*

Proof. Project the carrier of Definition 4106.1. The tuple lists exactly the term, judgment, membership, derivation, subject-reduction, erasure, transport, replay, provenance, and local naming rows.

Replay the continuation row C. A type-checking request first reads the finite term row T and typing ledger J, then the Ext-membership row E and derivation trace D. Subject reduction is read from S, and proof-term erasure is kept in the displayed row R.

Transport by H is componentwise over those rows. The provenance and naming rows P, N therefore package only the finite classifier route, leaving no coordinate from which a host oracle, quotient syntax, hidden proof object, or ambient metatheorem could be exported. □

Theorem 4106.3 (TypeCheckingClassifier membership soundness)**.** *Let* $Q = (T, J, E, D, S, R, H, C, P, N)$ *be an accepted* $\mathsf{TypeCheckingClassifier}^{\uparrow}$ *packet. Every successful type-checking answer exported by* Q *is an* Ext*-membership read supported by the displayed typing ledger and derivation trace:*

$$T, J \longmapsto E, D \longmapsto S, R, H, C, P, N.$$

The subject-reduction row preserves the membership read along displayed Cont *replay, and the erasure row separates proof-term evidence from erased computational content. The handoff preserves* $T, J, E, D, S, R, H, C, P, N$ *and exports no host conversion oracle, semantic term equality, normalization theorem, hidden proof object, quotient syntax, or ambient consistency principle.*

Proof. Begin with a successful type-checking answer. By carrier admission, the answer can only be read through the rows of Definition 4106.1.

The finite term row T and typing ledger J provide the displayed input to the classifier. The Ext row E records membership, and the derivation trace D records the finite route witnessing the accepted judgment.

Subject reduction is then read as the displayed row S along Cont replay, while R keeps proof-term erasure separate from the erased computational term. Componentwise transport, provenance, and naming preserve this route through H, C, P, N without adding the excluded host-level principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypeCheckingClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4106.1, Definition 4106.2, and Definition 4106.3 seed the finite term, typing-ledger, Ext-membership, derivation-trace, subject-reduction, erasure, transport, replay, provenance, and naming route.

Formal verification: FormalStatus(NameCert$_{\mathsf{TypeCheckingClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host conversion, semantic equality of terms, normalization, hidden proof objects, quotient syntax, ambient metatheoretic consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Ext-membership soundness, derivation-trace support, subject-reduction replay, erasure separation, and structural nonescape for BEDC.Derived.TypeCheckingClassifierUp.

Constructive story: A TypeCheckingClassifier$^{\uparrow}$ packet is generated from term rows, a finite typing ledger, Ext-membership data, derivation traces, subject-reduction replay, erasure separation, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Chapter 4107

A Concrete Naming Certificate for $\mathsf{MengerNobelingEmbedding}^{\uparrow}$

Following Menger and Nobeling, the $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ packet records a finite-dimensional metric embedding handoff as BEDC NameCert data. It sits after the metric carrier of chapter 57, the compact finite-net surface of chapter 129, the distance-function route of chapter 7569, the tolerance-indexed net row of chapter 1767, and the terminal real-name boundary of chapter 38. The packet names the finite coordinate route needed by embedding consumers; it is not a host Euclidean space, not a selected infinite coordinate family, not a quotient completion, and not an ambient dimension theorem.

Definition 4107.1 (Menger–Nobeling embedding carrier)**.** *A* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, K, Q, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose point classifier and distance classifier are read by the embedding packet.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact finite-metric handoff row supplying the finite cover and compact inspection data used by the coordinate budget.*
3. *Q is the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *distance-function row that turns source-distance observations into displayed real-valued coordinate observations.*
4. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *row that records the inspected center bundle, tolerance, and coverage witnesses used to choose a finite coordinate window.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the displayed coordinate values and finite separation inequalities.*
6. *H records componentwise* hsame *transport for the metric source, compact handoff, Kuratowski row, finite-net row, real readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from metric-distance reads through K, Q, F, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, K, Q, F, R, H, C, P, N$. It has no coordinate for a host Euclidean carrier, a selected infinite coordinate family, quotient completion, open-cover compactness, countable choice, or an ambient dimension principle.

Lean4 ✓ `BEDC.Derived.MengerNobelingEmbeddingUp.MengerNobelingEmbeddingTasteGate single carrier alignment`

Theorem 4107.2 (Menger–Nobeling embedding NameCert obligations)**.** *For every accepted* MengerNobelingEmbe *carrier $E = (M, K, Q, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, compact handoff, Kuratowski distance-function row, finite epsilon-net row, real coordinate readback, transport, replay, provenance, and local naming row;*
2. *metric-source exposure, requiring every source point and distance read to enter through M;*
3. *compact finite-net exposure, requiring every finite coordinate budget to be read through the compact row K and the finite-net row F;*
4. *Kuratowski coordinate handoff, requiring every displayed coordinate observation to pass through Q before its* $\mathsf{Real}^{\uparrow}$ *readback in R is consumed;*
5. *finite separation stability, requiring the same F, Q, R rows to witness the finite coordinate separation read under componentwise transport H;*
6. *non-escape, excluding host Euclidean carriers, selected infinite coordinate families, quotient completions, open-cover compactness, countable choice, and ambient dimension principles from the local certificate.*

No Menger–Nobeling embedding obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4107.1. The only analytic rows are the metric source M, the compact handoff K, the Kuratowski distance-function row Q, the finite-net row F, and the real coordinate readback row R.

The compact coordinate budget is finite because it is read through K and F. The compact row supplies the finite inspection context, while the finite epsilon-net row supplies the displayed center bundle, tolerance, and coverage witnesses used by the coordinate window.

The coordinate values are then forced through the Kuratowski route. The row Q exposes distance-function observations, and R reads those observations through the terminal Real boundary before any separation claim is exported.

Finally, H, C, P, N preserve and name the same route. Since the carrier has no Euclidean host, infinite coordinate-family selector, quotient-completion row, open-cover compactness row, choice schedule, or ambient dimension row, none of those objects can become a local NameCert obligation. □

Theorem 4107.3 (Menger–Nobeling embedding finite coordinate handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{KuratowskiEmbedding}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ *carrier must read the finite route*

$$M, K, Q, F, R, H, C, P, N.$$

Thus a finite-dimensional embedding request reaches coordinate data only after the metric source, compact finite-net budget, Kuratowski distance-function observations, and real readbacks are visible. The handoff exports no host Euclidean carrier, selected infinite coordinate family, quotient completion, open-cover compactness, countable choice, or ambient dimension principle.

Proof. Begin with an accepted carrier $E = (M, K, Q, F, R, H, C, P, N)$. A consumer can enter the packet only through the metric source M and the compact handoff K.

The finite coordinate window is opened next. The Kuratowski row Q supplies distance-function observations, the finite epsilon-net row F supplies the displayed center and coverage budget, and the Real row R supplies the coordinate readbacks.

The structural rows then package exactly that handoff. Componentwise transport H preserves the same finite route, replay C exposes it to downstream consumers, provenance P records the package lineage, and N names the local certificate. Therefore the embedding consumer receives only the displayed finite coordinate handoff and not any hidden host embedding principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4107.1, Definition 4107.2, and Definition 4107.3 seed the finite coordinate embedding route over MetricUp, CompactMetricUp, KuratowskiEmbeddingUp, FiniteEpsilonNetUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MengerNobelingEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host Euclidean embedding, selected infinite coordinate families, quotient completion, open-cover compactness, countable choice, ambient dimension principles, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, Kuratowski coordinate handoff, finite separation stability, Real readback, transport, replay, provenance, and local naming for BEDC.Derived.MengerNobelingEmbe

Constructive story: A $\mathsf{MengerNobelingEmbedding}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp handoff, a KuratowskiEmbeddingUp distance-function row, a FiniteEpsilonNetUp coordinate budget, a RealUp readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4108

A Concrete Naming Certificate for $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ packet records the completion route in which a displayed Cauchy filter is read through a uniform-space entourage basis before any quotient completion or selected limit is available. It sits after the uniform-space surface of chapter 339, the filter-basis generator of chapter 1501, the saturated Cauchy-filter packet of chapter 333, and the completion boundary of chapter 95. Its finite readback route uses the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38.

Definition 4108.1 (Uniform-space completion-filter carrier)**.** *A* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (U, B, F, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row, including the finite diagonal, refinement, symmetry, and composition data visible to the packet.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row, listing the basis elements and common-refinement witnesses read from U.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *saturated filter row generated from B.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule controlling which filter and entourage reads may be inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed windows and tolerance requests.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for entourage shrinkage, Cauchy refinement, and completion-radius comparison.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *completion-admission seal reached only after U, B, F, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for all displayed rows.*
9. *C records displayed* Cont *replay from a completion request through the uniform, basis, filter, window, readback, tolerance, and Real-seal rows.*

10. *P is the* Pkg *provenance over* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, B, F, S, R, D, A, H, C, P, N$. It has no coordinate for quotient filters, host equality of filters, selected representatives, countable choice, a chosen completion point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformSpaceCompletionFilterUp.UniformSpaceCompletionFilterTasteGate`
`single carrier alignment`

Theorem 4108.2 (Uniform-space completion-filter NameCert obligations)**.** *For every accepted* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *carrier $Q = (U, B, F, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionF}}$ *surface consists exactly of carrier admission for the displayed uniform-space entourage row, Cauchy-filter basis row, saturated Cauchy-filter row, stream-window row, regular-rational readback row, dyadic tolerance ledger, Real completion-admission seal, componentwise transport, continuation replay, package provenance, and local naming row. Every completion-facing read of Q must pass through*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

before the completion boundary of $\mathsf{Completion}^{\uparrow}$ *may consume the packet.*

Proof. Project the carrier of Definition 4108.1. The coordinates list exactly the uniform, basis, filter, window, readback, tolerance, Real-seal, transport, replay, provenance, and local naming rows.

The completion-facing order is forced by C. A consumer first reads the entourage data U, then the basis row B and saturated filter row F, and only then inspects the finite windows S, dyadic ledger D, regular rational readback R, and Real seal A.

The structural rows do not add a second source. The transport row H moves only the displayed coordinates componentwise, while P and N name the same finite packet as provenance and local certificate data.

Since no coordinate of Q names quotient filters, host filter equality, selected representatives, countable choice, a chosen completion point, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those objects cannot be exported by the local NameCert surface. The listed rows therefore exhaust the obligations. □

Theorem 4108.3 (Uniform-space completion-filter boundary)**.** *Let $Q = (U, B, F, S, R, D, A, H, C, P, N)$ be an accepted* $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ *packet. Any* $\mathsf{Completion}^{\uparrow}$ *consumer of Q receives a finite filter-completion route through the displayed rows $U, B, F, S, R, D, A, H, C, P, N$ and receives no quotient-filter class, selected completion point, host filter equality, countable-choice selector, hidden cofinal subfilter, or ambient completeness theorem.*

Proof. Begin with the NameCert obligation theorem, Definition 4108.2. It routes completion-facing reads through the uniform row, Cauchy-filter basis, Cauchy filter, finite windows, dyadic tolerance ledger, regular readback, and Real seal before the completion boundary consumes the packet.

The replay and transport rows preserve the same route. They do not add representatives, quotient classes, hidden cofinal filters, or selected limits; they only replay and transport the displayed coordinates.

Finally inspect the carrier definition. The excluded objects are absent from Q, so a $\mathsf{Completion}^{\uparrow}$ consumer cannot extract them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}$.

Constructive story: A $\mathsf{UniformSpaceCompletionFilter}^{\uparrow}$ packet is generated from a UniformSpace entourage row, CauchyFilterBasis generator row, CauchyFilter saturation row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a RealUp completion-admission seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4108.1, Definition 4108.2, and Definition 4108.3 seed the non-quotient filter-completion route for UniformSpaceUp, CauchyFilterBasisUp, CauchyFilterUp, CompletionUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, and RealUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSpaceCompletionFilter}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filter completion, selected completion points, host filter equality, countable choice, hidden cofinal subfilters, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-to-filter saturation, completion-facing replay, Real-seal handoff, and refusal of quotient-filter and selected-limit escapes for BEDC.Derived.UniformSpaceCompletionFilterUp.

Chapter 4109

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$

Following Bishop–Bridges regular Cauchy real names and Kelley filter terminology, the RegularCauchyFilterCompleti packet records the L10 route by which a regular Cauchy name is read as a Cauchy-filter completion object through finite BEDC rows. It sits after the filter-limit boundary of chapter 1558, the completion-facing filter surface of chapter 1569, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives filter-completion consumers a canonical regular-name bridge without quotient filters, selected representatives, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4109.1 (Regular Cauchy filter completion carrier)**.** *A* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, F, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary row available only after the source filter and regular-name requests have been replayed.*
2. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-facing row whose consumers may read only the displayed filter route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for regular Cauchy tail requests and filter-completion reads.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the requested windows, comparison radius, and filter limit slack.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and dyadic tolerance ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, F, S, D, R have been displayed.*
7. *H records componentwise* hsame *transport for the limit, completion, window, tolerance, readback, seal, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a regular Cauchy filter-completion request through $L, F, S, D, R, E, H, P, N$.*

9. *P records* Pkg *provenance over* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, F, S, D, R, E, H, C, P, N$. It stores no quotient filter, selected completion point, hidden cofinal sequence, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterCompletionUp.RegularCauchyFilterCompletionTasteGate`
`single carrier alignment`

Theorem 4109.2 (Regular Cauchy filter completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *carrier*

$$Q = (L, F, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *boundary exposure through L,* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer exposure through F, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal discipline through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4109.1. The tuple displays the filter-limit boundary, completion-facing row, window schedule, dyadic tolerance ledger, regular rational readback, terminal real seal, and structural rows.

Read the analytic route in order. The limit row L and completion row F place the request inside the Cauchy-filter completion surface before any finite regular-name data are inspected.

The regular-name data are then exhausted by S, D, R, E. The stream windows in S are measured by the dyadic ledger D, the regular rational row R is read from that displayed data, and the real seal E appears only after those rows have been exposed.

Finally H, C, P, N transport, replay, source, and name the same finite packet. No remaining coordinate can supply quotient filters, selected representatives, hidden cofinal choices, host equality, or ambient completeness, so the listed rows are exactly the local NameCert obligations. □

Theorem 4109.3 (Regular Cauchy filter completion seal handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *packet receives the* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{CauchyFilterLimit}^{\uparrow} \longmapsto \mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff is a filter-completion reading of a regular Cauchy name, not a quotient filter construction, selected completion point, host real equality, or ambient completeness principle.

Proof. Apply Definition 4109.2. It orders the completion-facing read through L, F, S, D, R, E before the terminal seal is visible.

The first two rows bind the filter-completion side. The $\mathsf{CauchyFilterLimit}^{\uparrow}$ row records the limit-facing boundary, and the $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row records the completion consumer that may inspect the same displayed route.

The remaining rows bind the regular-name side. The finite windows, dyadic ledger, and regular rational readback feed the terminal real seal, while H, C, P, N only preserve and name that route.

Thus any consumer that reaches $\mathsf{Real}^{\uparrow}$ without passing through the listed route leaves the carrier. The handoff supplies a finite regular Cauchy filter-completion bridge and no hidden completion object. □

Theorem 4109.4 (Regular Cauchy filter completion nonescape)**.** *No accepted* $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ *packet exports quotient filter equality, a selected representative, a chosen cofinal tail, host real equality, countable choice, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*. Every exported completion or real-facing read is confined to the rows* $L, F, S, D, R, E, H, C, P, N$*.*

Proof. Inspect the carrier definition. Its non-structural rows are exactly the filter-limit boundary L, completion row F, stream windows S, dyadic tolerance ledger D, regular rational readback R, and real seal E.

The structural rows add no analytic principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet rather than adding quotient filters, selected representatives, cofinal choices, or host equality.

Use the seal handoff of Definition 4109.3. Every completion or real-facing read must pass through the displayed finite route, so none of the excluded data can escape from the packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: cauchyfilterlimit, cauchyfiltercompletion, streamname, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ because this packet binds the filter-limit boundary, completion-facing row, finite windows, tolerance ledger, regular readback, terminal seal, transport, replay, provenance, and local naming as one L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyFilterCompletion}^{\uparrow}$ packet is generated from a CauchyFilterLimit boundary row, CauchyFilterCompletion consumer row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 4109.1, Definition 4109.2, Definition 4109.3, and Definition 4109.4 seed the L10 regular Cauchy filter-completion route over CauchyFilterLimitUp, CauchyFilterCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyFilterCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, filter-limit exposure, completion-facing exposure, window exactness, tolerance exactness, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and ledger nonescape for BEDC.Derived.RegularCauchyFilterCompletionUp.

Chapter 4110

A Concrete Naming Certificate for $\mathsf{CauchyFilterRegularization}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{CauchyFilterRegularization}^{\uparrow}$ packet records the L10 handoff that turns a displayed Cauchy-filter completion route into a regular Cauchy real-name route through finite BEDC rows. It sits after the Cauchy-filter surface of chapter 333, the completion boundary of chapter 1569, the Polish carrier discipline of chapter 1490, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite regularization handoff; it does not quotient filters, select representatives, import countable choice, assert host equality of limits, or use ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4110.1 (Cauchy-filter regularization carrier)**.** *A* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, W, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row, naming the finite filter observations and the completion-facing boundary they expose.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for regularization requests and tail comparisons.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same windows, comparison radii, and completion requests.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the displayed windows and dyadic tolerance data.*
5. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the filter-completion source, stream windows, dyadic ledger, regular readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a regularization request through F, W, D, R, L, H, P, N.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyFilterRegularization}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, D, R, L, H, C, P, N$. It stores no quotient filter, selected completion point, hidden cofinal subsequence, host real equality, countable-choice schedule, or ambient completeness coordinate.

Lean4 ✓ BEDC.Derived.CauchyFilterRegularizationUp.CauchyFilterRegularizationTasteGate single carrier alignment

Theorem 4110.2 (Cauchy-filter regularization NameCert obligations)**.** *For every accepted* CauchyFilterRegularizati *carrier*

$$G = (F, W, D, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed source, window, tolerance, readback, seal, transport, replay, provenance, and local-name rows;*
2. *filter exposure, requiring every regularization read to expose the* $\mathsf{CauchyFilter}^{\uparrow}/\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row F before a stream window is inspected;*
3. *window and tolerance exactness, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row W and* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D to determine every finite comparison used by the handoff;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring the regular rational row R to be read from the displayed W, D data and not from a hidden representative;*
5. $\mathsf{Real}^{\uparrow}$*-seal nonescape, requiring the terminal row L to be consumed only after F, W, D, R have been displayed;*
6. *provenance, requiring H, C, P, N to transport, replay, source, and name the same finite packet without adding quotient filters, selected limits, choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 4110.1. The source row F exposes the Cauchy-filter completion boundary, and the rows W, D, R, L are the only non-structural regularization rows downstream from that boundary.

The window and tolerance part is finite. A regularization request opens the listed $\mathsf{StreamName}^{\uparrow}$ windows in W, reads the dyadic tolerance budget in D, and then obtains the $\mathsf{RegSeqRat}^{\uparrow}$ row R from those displayed observations.

The real-seal handoff is last. The terminal row L is reachable only after F, W, D, R have been exposed, while H, C, P, N only transport, replay, source, and name the same packet. Since no remaining coordinate stores quotient filters, selected representatives, hidden cofinal choices, host real equality, or ambient completeness, the listed clauses exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterRegularization}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed CauchyFilter and CauchyFilterCompletion source, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the packet surface.

Independence witness: cauchyfiltercompletion, polishspace, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyFilterRegularization}^{\uparrow}$ is not bijective to any listed sibling because it binds a Cauchy-filter completion boundary, finite window schedule, dyadic tolerance ledger, regular rational handoff, Real seal, transport, replay, provenance, and local naming into one L10 regularization certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterRegularization}^{\uparrow}}$.

Constructive story: A CauchyFilterRegularization$^{\uparrow}$ packet is generated from a CauchyFilter/CauchyFilterCompletion source row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyFilterRegularization}^{\uparrow}}$, seed)

Scope closed: Definition 4110.1 and Definition 4110.2 seed the L10 Cauchy-filter regularization handoff over CauchyFilterUp, CauchyFilterCompletionUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyFilterRegularization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, filter exposure, window exactness, tolerance exactness, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterRegularizationUp.

Chapter 4111

A Concrete Naming Certificate for $\mathsf{RealInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealInterval}^{\uparrow}$ packet records displayed real interval endpoints and finite enclosure rows for the L10 Real and Completion route. It sits after the located interval surface of chapter 394, the rational interval surface of chapter 354, and the dyadic-to-real spine of chapter 291, chapter 126, chapter 301, and chapter 38. The chapter gives interval-completion and real-enclosure consumers one named endpoint-and-enclosure packet; it is not a host closed interval, selected endpoint representatives, trichotomy, compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4111.1 (RealInterval carrier)**.** *A* $\mathsf{RealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, E, D, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint* $\mathsf{Real}^{\uparrow}$ *row, read through the displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *seal discipline.*
2. *U is the upper endpoint* $\mathsf{Real}^{\uparrow}$ *row, read through the same finite real-name seal discipline.*
3. *E is the enclosure and order row, binding the lower endpoint, upper endpoint, rational interval enclosure, and endpoint-order evidence without replacing the endpoints by host representatives.*
4. *D is the dyadic tolerance row supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*; it records positive precision requests, outward rounding budgets, and finite endpoint comparison tolerances.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the two endpoint streams and the interval enclosure reads visible at the requested tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the endpoint windows and dyadic tolerance budget into regular rational observations.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint and enclosure consumers; every Real-facing output must pass through this row after D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for endpoints, enclosure and order data, dyadic tolerance, windows, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from interval membership, enclosure, refinement, or completion-facing requests through $L, U, E, D, W, R, S, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed real interval endpoint-and-enclosure history. It has no coordinate for host interval equality, selected quotient representatives, real-order trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalTasteGate single carrier alignment`

Theorem 4111.2 (RealInterval NameCert obligations)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower endpoint, upper endpoint, enclosure and order row, dyadic tolerance row, stream windows, regular rational readback, Real seal, transport, replay, provenance, and local name;*
2. *endpoint order, requiring E to bind L and U before any interval read can be treated as an enclosure;*
3. *enclosure stability, requiring every containment, refinement, and endpoint-bound read to pass through E together with the displayed tolerance and window rows;*
4. *dyadic tolerance handoff, requiring D to precede the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R;*
5. *Real seal non-escape, requiring every Real-facing endpoint or enclosure output to reach S only after L, U, E, D, W, R have been displayed;*
6. *ledger refusal, excluding host interval equality, selected endpoint representatives, real-order trichotomy, countable choice, compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No real interval obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4111.1. The projection exposes exactly the two endpoint rows, enclosure and order row, dyadic tolerance row, stream windows, regular rational readback, Real seal, and structural rows.

Endpoint order and enclosure stability are read from E. That row is the only coordinate joining L and U into an interval boundary, so containment, refinement, and endpoint-bound requests cannot bypass it.

The dyadic tolerance handoff is the route $D \mapsto W \mapsto R$. A positive precision request first enters the dyadic budget, then selects finite stream windows, and only then produces regular rational readback.

The Real seal row S is downstream of that route. The remaining rows H, C, P, N transport, replay, package, and name the same endpoint-and-enclosure history, while the excluded host interval, choice, compactness, and completeness objects have no coordinate in the carrier. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalCarrier namecert obligations`

Theorem 4111.3 (RealInterval seal enclosure route)**.** *Every endpoint, enclosure, refinement, or completion-facing read of an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier reaches the* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed route*

$$L, U, E \longmapsto D, W, R \longmapsto S, H, C, P, N.$$

Consequently every interval consumer receives endpoint order, finite enclosure data, dyadic tolerance, stream windows, and regular rational readback before any Real-facing seal is visible. The route exports no host interval equality, selected endpoint representative, real-order trichotomy, countable-choice schedule, compactness theorem, hidden complete space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with an accepted carrier $I = (L, U, E, D, W, R, S, H, C, P, N)$. The endpoint rows L and U are the only Real endpoints in the packet, and E is the only row that binds them as an ordered enclosure.

Read the dyadic tolerance handoff next. The row D supplies the finite precision and outward-rounding budget, W exposes the endpoint stream windows visible at that budget, and R turns those windows into regular rational readback.

Only after this handoff may the Real-facing seal S be consumed. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve the same route. Since none of the excluded host interval or completeness objects is a carrier coordinate, every interval read reaches the seal only through the displayed rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalSealEnclosureRoute`

Theorem 4111.4 (RealInterval endpoint-order obligation)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *be an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier. Any consumer that treats* L *and* U *as an interval must first read the endpoint-order row* E*; the two endpoint rows alone do not authorize an interval enclosure.*

Proof. Project the carrier of Definition 4111.1. The rows L and U name the two endpoint Real rows, while E is the only displayed row that binds them as an ordered enclosure.

Now apply the local obligations of Definition 4111.2. The endpoint-order obligation explicitly requires E to bind L and U before any interval read can be treated as an enclosure.

The seal route of Definition 4111.3 preserves the same order, since every endpoint or enclosure read begins with L, U, E before entering the dyadic handoff. Thus an interval consumer cannot bypass E by reading the endpoint rows alone. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointOrderObligation`

Theorem 4111.5 (RealInterval classifier transport)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *and* $I' = (L', U', E', D', W', R', S', H', C', P', N')$ *be accepted* $\mathsf{RealInterval}^{\uparrow}$ *carriers. If the accepted interval classifier identifies the endpoint and enclosure rows of* I *with those of* I'*, then carrier admission transports from* I *to* I' *only through the displayed endpoint, locatedness, dyadic-window,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. In particular, classifier equality cannot replace the rows supplied by chapter 291, chapter 394, chapter 301, chapter 126, or chapter 38.*

Proof. Project both carriers through Definition 4111.1. The endpoint rows L, U, L', U' and enclosure rows E, E' are the only rows on which the interval classifier can act.

Use the endpoint-order obligation Definition 4111.4. It forces every interval read to keep the endpoint order row visible, so classifier equality must transport the ordered enclosure rather than only the two endpoints.

The seal route of Definition 4111.3 then carries the transported enclosure through the dyadic tolerance, stream-window, regular-rational, and Real-seal rows. The structural rows H', C', P', N' name the transported packet, so carrier admission moves only along the displayed BEDC rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalClassifierTransport`

Theorem 4111.6 (RealInterval endpoint refinement stability)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, any dyadic refinement of the endpoint enclosure that preserves the displayed endpoint order and locatedness rows preserves the* $\mathsf{RealInterval}^{\uparrow}$ *NameCert surface before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is read. The refinement may change the finite dyadic tolerance and window rows* D, W, R*, but it must keep the endpoint, locatedness, transport, replay, provenance, and local naming rows visible before* S *is consumed.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointRefinementStability`

Proof. Read the carrier rows of Definition 4111.1. A dyadic endpoint refinement changes only the tolerance, window, and regular-rational observations recorded in D, W, R.

Apply Definition 4111.2. Its endpoint order and enclosure stability clauses keep L, U, E visible, while its dyadic tolerance handoff clause keeps every refinement inside the $D \mapsto W \mapsto R$ route.

Finally use the seal enclosure route Definition 4111.3. Since S is terminal, the refined endpoint surface must already have passed through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the displayed structural rows before any $\mathsf{Real}^{\uparrow}$ seal is available. □

Theorem 4111.7 (RealInterval dyadic-window handoff)**.** *For every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$*, the displayed route*

$$D \longmapsto W \longmapsto R$$

is the only tolerance-to-readback handoff supplied by the packet.

Proof. Project Definition 4111.1. The tolerance coordinate is D, the finite endpoint-window coordinate is W, and the regular rational readback coordinate is R.

Use Definition 4111.2. Its dyadic tolerance handoff row requires D to precede the $\mathsf{StreamName}^{\uparrow}$ windows W and the $\mathsf{RegSeqRat}^{\uparrow}$ readback R.

Finally apply Definition 4111.3. The route displays no second tolerance-to-readback path; every enclosure read moves from D to W to R before any seal or structural row is public. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalDyadicWindowHandoff`

Theorem 4111.8 (RealInterval Real-seal nonescape)**.** *Let* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *be an accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier. Every Real-facing enclosure read reaches the seal row* S *only after the endpoint rows, endpoint-order row, dyadic tolerance row, stream-window row, and regular readback row* L, U, E, D, W, R *have all been displayed.*

Proof. Start from the carrier of Definition 4111.1. The seal row S is not an independent endpoint; it is the terminal Real-facing row for endpoint and enclosure consumers.

The obligations of Definition 4111.2 make Real seal non-escape explicit. They require every Real-facing endpoint or enclosure output to reach S only after L, U, E, D, W, R have been displayed.

The route theorem Definition 4111.3 gives the same ordering as a displayed path. Therefore a Real-facing enclosure read cannot reach S while omitting any of the endpoint, order, tolerance, window, or readback rows. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalRealSealNonescape`

Theorem 4111.9 (RealInterval obligation-closure route)**.** *Every accepted* $\mathsf{RealInterval}^{\uparrow}$ *carrier* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *routes endpoint admission, endpoint order, enclosure stability, dyadic-window handoff, Real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *through the displayed rows* $L, U, E, D, W, R, S, H, C, P, N$*.*

The route supplies no host interval equality, selected endpoint representative, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalCarrier obligation closure route`

Proof. First read the local obligation surface from Definition 4111.2. It exposes carrier admission, endpoint order, enclosure stability, dyadic tolerance handoff, Real seal non-escape, transport, replay, provenance, and local naming as finite rows of the accepted carrier.

Next use the route theorems that keep these rows in order. The seal enclosure route of Definition 4111.3, the endpoint-order obligation of Definition 4111.4, and the dyadic-window handoff of Definition 4111.7 force every interval read through L, U, E, D, W, R, S before the structural rows are consumed.

Finally apply Real-seal nonescape from Definition 4111.8. Since every Real-facing enclosure read reaches S only after L, U, E, D, W, R have been displayed, and since H, C, P, N only transport, replay, package, and name that same route, the carrier closes exactly the obligation route stated above while leaving the refused global principles outside the packet. □

Theorem 4111.10 (RealInterval public NameCert export)**.** *The public* $\mathsf{RealInterval}^{\uparrow}$ *export is exactly the endpoint-pair carrier together with endpoint order, classifier transport, endpoint-refinement stability, dyadic-window handoff, and Real-seal nonescape rows. A public interval consumer may cite the ordered surface*

$$L, U, E, D, W, R, S, H, C, P, N$$

only as the endpoint-and-enclosure packet of Definition 4111.1; it may not add host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalPublicNameCertExport`

Proof. Start from the carrier and obligation route. Theorems 4111.1 and 4111.9 identify the complete endpoint-and-enclosure packet as the rows $L, U, E, D, W, R, S, H, C, P, N$.

The public surface is determined by the interval-facing route theorems. Endpoint order is supplied by Definition 4111.4, classifier transport by Definition 4111.5, endpoint-refinement stability by Definition 4111.6, dyadic-window handoff by Definition 4111.7, and Real-seal nonescape by Definition 4111.8.

These theorems cite the same displayed rows and no additional host object. Therefore the reusable public export is precisely the endpoint-pair carrier with its order, transport, refinement, dyadic-window, and seal rows, while the refused global principles remain outside the packet. □

Theorem 4111.11 (RealInterval finite-net consumer)**.** *Every finite-net or density-net consumer of an accepted* $\mathsf{RealInterval}^{\uparrow}$ *packet must read the endpoint order row* E*, dyadic tolerance row* D*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row* S *before passing interval data into chapter 5966 or chapter 6003. The consumer route is the displayed packet*

$$L, U, E, D, W, R, S, H, C, P, N,$$

not a host interval equality, selected endpoint representative, compactness theorem, countable-choice net, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFiniteNetConsumer`

Proof. Start with endpoint order. By Definition 4111.4, a consumer cannot treat L and U as an interval until it has read the order and enclosure row E.

Next apply the dyadic-window handoff of Definition 4111.7. It forces the requested tolerance through D, then the finite StreamName windows W, and only then the regular rational readback row R used by dyadic interval cells or finite density rows.

Now use Real-seal nonescape and the public NameCert export, Definition 4111.8 and Definition 4111.10. The seal row S is available only after L, U, E, D, W, R have been displayed, and the public export keeps H, C, P, N attached to that same endpoint-and-enclosure packet.

Finally route the displayed interval data to the dyadic interval-net and finite density-net sibling chapters. Since Definition 4111.1 contains no coordinate for host interval equality, selected endpoint representatives, compactness, countable choice, or ambient completeness, those siblings receive only the finite displayed route stated above. □

Theorem 4111.12 (RealInterval bridged NameCert route)**.** *Every bridge-facing* $\mathsf{RealInterval}^{\uparrow}$ *consumer routes through the displayed endpoint-and-enclosure packet*

$$L, U, E, D, W, R, S, H, C, P, N.$$

The route first reads the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger* D*, then the* $\mathsf{LocatedInterval}^{\uparrow}$ *and* $\mathsf{RealIntervalEnclosure}^{\uparrow}$ *rows carried by* E*, then the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and finally the* $\mathsf{Real}^{\uparrow}$ *seal* S *with transport, replay, provenance, and local naming still visible.*

Proof. Project the accepted carrier of Definition 4111.1. The endpoint rows, enclosure row, tolerance ledger, windows, readback, seal, and structural rows are all coordinates of one packet.

Use the NameCert obligations and public export, Definition 4111.2 and Definition 4111.10. They keep endpoint order, enclosure stability, dyadic-window handoff, Real-seal nonescape, transport, replay, provenance, and local naming attached to the displayed rows.

Finally apply the finite-net consumer route of Definition 4111.11. It shows that downstream interval data reaches finite cell and density consumers only through the dyadic, located-enclosure, StreamName, RegSeqRat, and Real-seal route. Hence the bridge-facing NameCert route is the displayed packet above. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalBridgedNameCertRoute`

Theorem 4111.13 (RealInterval finite endpoint-envelope route)**.** *Every* $\mathsf{RealInterval}^{\uparrow}$ *endpoint consumer that exports interval membership must first factor through the displayed finite endpoint envelope*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealIntervalEnclosure}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The dyadic endpoint arithmetic is read from D*, the finite endpoint windows from* W*, the regular readback from* R*, the enclosure row from* E*, and the terminal value seal from* S *before membership is public. The envelope exports no quotient stream equality, selected endpoints, host interval equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, compactness, or countable-choice schedule.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFiniteEndpointEnvelopeRoute`

Proof. Start with the endpoint-order obligation of Definition 4111.4. It keeps the enclosure and order row E visible whenever a consumer treats L and U as interval endpoints.

Next apply the dyadic-window handoff Definition 4111.7. The endpoint arithmetic enters through the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ row D, then exposes only finite $\mathsf{StreamName}^{\uparrow}$ windows W, and only then produces the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Use the bridged NameCert route Definition 4111.12. It inserts the $\mathsf{LocatedInterval}^{\uparrow}$ and $\mathsf{RealIntervalEnclosure}^{\uparrow}$ rows carried by E between the endpoint ledger and the terminal $\mathsf{Real}^{\uparrow}$ seal S, while H, C, P, N transport, replay, package, and name the same displayed route.

Finally read Real-seal nonescape from Definition 4111.8. Since S is available only after L, U, E, D, W, R have been displayed, interval membership cannot be exported from quotient stream equality, selected endpoints, host interval equality, ambient completeness, trichotomy, compactness, or countable choice. □

Theorem 4111.14 (RealInterval endpoint enclosure transport)**.** *For accepted* $\mathsf{RealInterval}^{\uparrow}$ *carriers* $I = (L, U, E, D, W, R, S, H, C, P, N)$ *and* $I' = (L', U', E', D', W', R', S', H', C', P', N')$*, componentwise* hsame *transport preserves both endpoint enclosure rows before interval membership is public. In particular, a consumer that transports* L *to* L' *and* U *to* U' *must also transport the enclosure-and-order row* E *to* E' *and the dyadic-window-readback route* D, W, R *to* D', W', R' *before it may read the terminal seal.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalEndpointEnclosureTransport`

Proof. Start with classifier transport, Definition 4111.5. It allows carrier admission to move only through the displayed endpoint, enclosure, dyadic-window, hsame, Cont, Pkg, and NameCert rows.

Endpoint order then fixes the enclosure boundary. By Definition 4111.4, the two endpoint rows alone do not authorize interval membership; the row E must remain visible as the ordered enclosure.

Finally use endpoint-refinement stability and the dyadic-window handoff, Definition 4111.6 and Definition 4111.7. These rows keep D, W, R attached to the transported enclosure before S is consumed, so componentwise transport preserves both endpoint enclosure rows before membership becomes public. □

Theorem 4111.15 (RealInterval StreamName Real-seal nonescape)**.** *No* $\mathsf{RealInterval}^{\uparrow}$ *read that passes through* $\mathsf{StreamName}^{\uparrow}$ *windows and a* $\mathsf{Real}^{\uparrow}$ *seal exports open-cover compactness, quotient real equality, choice-selected endpoints, trichotomy, or* $\mathsf{Real}^{\uparrow}$ *completeness. The only public route is*

$$L, U, E, D, W, R, S, H, C, P, N,$$

with W *and* R *displayed before* S *and with the refusal boundary preserved by the local NameCert rows.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalStreamRealSealNonescape`

Proof. Read the Real-seal nonescape theorem Definition 4111.8. It places the seal row S strictly after the endpoint, order, tolerance, window, and readback rows L, U, E, D, W, R.

Use the obligation-closure route and public NameCert export, Definition 4111.9 and Definition 4111.10. They state that H, C, P, N only transport, replay, package, and name the same displayed packet.

The finite-net consumer theorem Definition 4111.11 keeps downstream interval reads inside the same route. Since none of the carrier coordinates names open-cover compactness, quotient equality, selected endpoints, trichotomy, or ambient completeness, the StreamName-to-Real-seal read cannot export those objects. □

Theorem 4111.16 (RealInterval bridge scope)**.** *Every bridge-facing* $\mathsf{RealInterval}^{\uparrow}$ *read factors through the finite endpoint-and-enclosure rows*

$$L, U, E, D, W, R, S, H, C, P, N$$

and by the sibling surfaces $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *only as endpoint-enclosure readback. The bridge-scope packet cannot be read as host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the public export of Definition 4111.10. It binds public interval reads to the endpoint, enclosure, tolerance, window, readback, seal, transport, replay, provenance, and local naming rows of the displayed carrier.

Next use the bridged route and endpoint-enclosure transport, Definition 4111.12 and Definition 4111.14. These results keep the bridge-facing read on the located endpoint row, dyadic interval envelope, finite stream windows, regular rational readback, and terminal Real seal before any interval membership is public.

Finally apply the StreamName Real-seal nonescape theorem, Definition 4111.15. The seal row S remains downstream of L, U, E, D, W, R and upstream of no additional host principle, so the bridge scope is exactly the displayed endpoint-enclosure packet. Compactness, selected endpoint representatives, trichotomy, countable choice, and ambient completeness have no row in this factorization. □

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalBridgeScope`

Theorem 4111.17 (RealInterval formal target handoff)**.** *Any future Lean target that advances the formal status of* $\mathsf{RealInterval}^{\uparrow}$ *must expose the endpoint carrier rows* L, U*, the dyadic-window handoff rows* D, W, R*, the endpoint-order obligation* E*, the Real-seal nonescape row* S*, and the bridge-scope rows* H, C, P, N *before it may claim a checked public interval surface. In particular, the target must route through the carrier of Definition 4111.1, the finite-net consumer of Definition 4111.11, the finite endpoint envelope of Definition 4111.14, the StreamName Real-seal nonescape theorem of Definition 4111.15, and the bridge scope of Definition 4111.16.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalFormalTargetHandoff`

Proof. Start with the carrier definition. It presents the endpoint rows L, U, the endpoint-order and enclosure row E, the dyadic tolerance row D, the $\mathsf{StreamName}^{\uparrow}$ window row W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, the $\mathsf{Real}^{\uparrow}$ seal row S, and the structural rows H, C, P, N in one accepted packet.

Use the dyadic-window handoff and finite-net consumer theorems. They keep downstream real-completion and net reads on D, W, R before the seal row is consumed, so a formal target cannot replace those rows by an ambient interval object.

Apply endpoint-enclosure transport and StreamName Real-seal nonescape. The endpoint envelope must transport both endpoints together with E, while the seal theorem keeps S from exporting quotient real equality, selected endpoints, trichotomy, compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the bridge-scope theorem. It leaves only H, C, P, N as transport, replay, provenance, and local naming for the same packet, so any checked Lean target must display exactly these handoff rows before the formal status can advance. □

Theorem 4111.18 (RealInterval mature consumer route)**.** *Every mature* $\mathsf{RealInterval}^{\uparrow}$ *consumer read factors through the endpoint-and-enclosure route already displayed by* $\mathsf{DyadicIntervalNet}^{\uparrow}$*,* FiniteCauchyDensityN

$\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *The route exposes lower endpoint L, upper endpoint U, enclosure and order row E, dyadic tolerance row D, finite window row W, regular-rational readback row R, terminal real seal S, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before any downstream interval read is accepted. It does not export open-cover compactness, quotient real equality, selected endpoint representatives, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalUp.RealIntervalMatureConsumerRoute`

Proof. Start with the endpoint carrier and finite-net consumer theorems. The carrier displays $L, U, E, D, W, R, S, H, C, P, N$, while Definition 4111.11 routes net-facing reads through the dyadic and finite Cauchy density surfaces before any public interval endpoint is consumed.

Next apply the endpoint-enclosure and StreamName Real-seal results. By Definition 4111.14, endpoint transport keeps E attached to L and U; by Definition 4111.15, the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows feed the terminal $\mathsf{Real}^{\uparrow}$ seal without adding quotient equality or completeness.

Finally use the bridge-scope and formal-target handoff theorems. They identify $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ as readback surfaces for the same displayed packet. The density and interval-net consumers therefore see only the endpoint-and-enclosure route, so the mature route closes without importing compactness, selected endpoints, trichotomy, countable choice, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 4111.1, Definition 4111.2, Definition 4111.3, Definition 4111.4, Definition 4111.7, Definition 4111.8, Definition 4111.9, Definition 4111.10, Definition 4111.16, and Definition 4111.18 bind endpoint admission, endpoint order, enclosure stability, dyadic-window handoff, finite net readback, Real-seal nonescape, hsame transport, Cont replay, Pkg provenance, and local NameCert to the displayed mature $\mathsf{RealInterval}^{\uparrow}$ rows over $\mathsf{DyadicIntervalNet}^{\uparrow}$, $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{RealIntervalEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint representatives, trichotomy, countable choice, compactness, ambient RealUp completeness, Lean verification, or a public standard bridge.

Upgrade path: Verification closure requires checked carrier admission, endpoint enclosure, dyadic-window handoff, finite-net consumer routing, Real-seal nonescape, transport, replay, provenance, local naming, and bridge interpretation for BEDC.Derived.RealIntervalUp.

Constructive story: A $\mathsf{RealInterval}^{\uparrow}$ packet is generated from lower and upper RealUp endpoint rows, an enclosure and order row, a DyadicRatCoreUp tolerance row, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4112

A Concrete Naming Certificate for $\mathsf{SequentialSpace}^{\uparrow}$

Following Franklin and Engelking on sequential spaces, the $\mathsf{SequentialSpace}^{\uparrow}$ packet records the finite convergence interface by which sequence-indexed observations may be consumed as topological data without importing open-cover compactness, quotient point equality, or a selected subsequence principle. It sits beside the limit carrier of chapter 3845, the Cauchy source interface of chapter 1847, the finite windows of chapter 126, the indexed-open topology surface of chapter 88, and the terminal real seal of chapter 38. Later completion and compactness consumers may read sequential convergence only through this displayed packet.

Definition 4112.1 (SequentialSpace carrier)**.** *A* $\mathsf{SequentialSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, W, L, T, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the sequence-index row, naming the finite initial segment, tail request, and comparison scale read by the consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite observations of the sequence at the requested indices.*
3. *L is the* $\mathsf{LimitSpace}^{\uparrow}$ *convergence row, recording the limit-facing test supplied by chapter 3845.*
4. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-test row, recording the indexed open neighbourhood test through which sequential convergence is observed.*
5. *Q is the* $\mathsf{CauchySpace}^{\uparrow}$ *comparison row when the sequence is also read by a completion or Cauchy consumer.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row when the displayed observations are regular real-name reads.*
7. *H records componentwise* hsame *transport for the sequence index, StreamName window, limit row, topology row, Cauchy comparison row, Real endpoint row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a sequential-convergence request through $I, W, L, T, Q, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{LimitSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$ *row.*
The carrier has no coordinate for open-cover compactness, quotient point equality, selected subsequences, countable-choice selectors, host topological closure, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.SequentialSpaceUp.SequentialSpaceTasteGate single carrier alignment`

Theorem 4112.2 (SequentialSpace NameCert obligations)**.** *For every accepted* $\mathsf{SequentialSpace}^\uparrow$ *carrier*

$$S = (I, W, L, T, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$ *surface consists exactly of sequence-index admission through I, finite observation through W, limit-space convergence through L, topology open-test exposure through T, Cauchy comparison through Q when present, Real endpoint exposure through R when present, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.SequentialSpaceUp.SequentialSpaceCarrier namecert obligations`

Proof. Project the carrier of Definition 4112.1. The sequence-facing side is exhausted by I and W, so a consumer can inspect only displayed finite StreamName windows at displayed indices.

The convergence-facing side is exhausted by L, T, Q, R. The limit row supplies the sequential convergence request, the topology row supplies the open-test interface, and the Cauchy and Real rows are available only when the same packet displays those comparison surfaces.

The rows H, C, P, N transport, replay, source, and name exactly this data. Since the carrier has no coordinate for open-cover compactness, quotient point equality, selected subsequences, countable choice, host closure, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 4112.3 (SequentialSpace completion-facing route)**.** *Every completion-facing consumer of an accepted* $\mathsf{SequentialSpace}^\uparrow$ *packet reads sequential convergence through the finite route*

$$\mathsf{StreamName}^\uparrow,\ \mathsf{LimitSpace}^\uparrow,\ \mathsf{Topology}^\uparrow,\ \mathsf{CauchySpace}^\uparrow,\ \mathsf{Real}^\uparrow.$$

The route exposes finite sequence windows, a displayed limit test, an indexed-open neighbourhood test, a Cauchy comparison surface when relevant, and a terminal Real endpoint row; it exports no open-cover compactness, selected subsequence, quotient point equality, countable-choice selector, or ambient Real completeness principle.

Proof. Start with the NameCert obligations of Definition 4112.2. A completion-facing read must first inspect the sequence index and finite StreamName window rows.

The limit request then passes through the $\mathsf{LimitSpace}^\uparrow$ row and the $\mathsf{Topology}^\uparrow$ open-test row. If the consumer is a completion or Cauchy consumer, the same route displays the $\mathsf{CauchySpace}^\uparrow$ comparison row before any $\mathsf{Real}^\uparrow$ endpoint row is read.

Transport, replay, provenance, and naming preserve this order. The excluded principles are not carrier coordinates, so no consumer can extract them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$.
Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$, seed)
Scope closed: Definition 4112.1, Definition 4112.2, and Definition 4112.3 seed the finite sequential-convergence interface over StreamName, LimitSpace, Topology, CauchySpace, and Real rows.
Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentialSpace}^\uparrow}$, unformalized)
Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient point equality, selected subsequences, countable-choice selectors, host topological closure, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sequence-index exactness, StreamName window coverage, LimitSpace convergence routing, Topology open-test exposure, Cauchy comparison compatibility, Real endpoint nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialSpaceUp.

Constructive story: A SequentialSpace$^{\uparrow}$ packet is generated from sequence-index rows, StreamName windows, LimitSpace convergence rows, Topology open-test rows, CauchySpace comparison rows where relevant, Real endpoint rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4113

A Concrete Naming Certificate for MetaCICConversionEndpointSeal$^{\uparrow}$

The MetaCICConversionEndpointSeal$^{\uparrow}$ packet realizes the endpoint question isolated in chapter 9386. It sits after the bounded normal-conversion bridge of chapter 3439, the finite fuel trace of chapter 3889, and the typed normal-form decider of chapter 3714. The packet names the exact surface where bounded normal computation may be sealed as finished normal-form data for the closed typed terms under discussion, while leaving unrestricted closed beta-conversion outside the carrier.

Falsifiable prediction: Every accepted MetaCICConversionEndpointSeal$^{\uparrow}$ read exposes bounded-normal-conversion bridge, MetaCIC fuel trace, typed normal-form decider, finished endpoint, closed typed term, obstruction, transport, replay, provenance, and local NameCert rows before any conversion endpoint handoff is consumed.

Independence witness: bounded-normal-conversion-bridge, metacic-fuel-trace, typed-normal-form-decider, metacic-decidable-conversion-frontier: the bridge records bounded comparison with endpoint evidence, the fuel trace records bounded computation, the typed decider records typed comparison, and the frontier records the broader decidability boundary. None is bijective to the endpoint seal because this packet binds those rows into one finished-normal endpoint handoff.

Definition 4113.1 (MetaCIC conversion endpoint seal carrier)**.** *A* MetaCICConversionEndpointSeal$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (B, F, D, T, O, H, C, P, N)$$

with the following displayed rows:

1. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, recording bounded normal comparison under displayed endpoint evidence.*
2. *F is the* MetaCICFuelTrace$^{\uparrow}$ *row, recording finite fuel, input, bounded output, finished-normalization evidence or its explicit absence, and bounded readback.*
3. *D is the* TypedNormalFormDecider$^{\uparrow}$ *row, recording the typed checker surface for the compared closed terms.*
4. *T is the closed typed-term row, recording the context, terms, inferred type status, and closedness window admitted by the decider surface.*

5. *O is the obstruction row, excluding unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, and kernel consistency.*
6. *H records componentwise* hsame *transport for B, F, D, T, O and the local naming row.*
7. *C records displayed* Cont *replay from a conversion endpoint request through B, F, D, T, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, $\mathsf{MetaCICFuelTrace}^{\uparrow}$, $\mathsf{TypedNormalFormDecider}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, F, D, T, O, H, C, P, N$. It has no coordinate for unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Lean4 ✓ `BEDC.Derived.MetaCICConversionEndpointSealUp.MetaCICConversionEndpointSealTasteGate`
`single carrier alignment`

Theorem 4113.2 (MetaCIC conversion endpoint seal obligations)**.** *For every accepted* MetaCICConversionEndpoint *carrier*

$$E = (B, F, D, T, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}}$ *surface consists exactly of bounded bridge exposure through B, finite fuel-trace exposure through F, typed normal-form decider exposure through D, closed typed-term endpoint exposure through T, obstruction retention through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4113.1. Rows B, F, D are the three sibling surfaces: bounded comparison, finite fuel trace, and typed normal-form decision.

The endpoint row T is read only after those sibling rows are displayed. It records the closed typed-term boundary under which bounded normal computation can be treated as a finished normal-form endpoint.

The obstruction row O remains part of the same packet. It blocks rereading the endpoint as unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, or kernel consistency.

Finally H, C, P, N transport, replay, source, and name exactly the same finite route. No carrier coordinate stores a stronger global metatheorem. □

Theorem 4113.3 (MetaCIC conversion endpoint seal handoff)**.** *Every endpoint-facing consumer of an accepted* $\mathsf{MetaCICConversionEndpointSeal}^{\uparrow}$ *carrier factors through*

$$\mathsf{BoundedNormalConversionBridge}^{\uparrow},\ \mathsf{MetaCICFuelTrace}^{\uparrow},\ \mathsf{TypedNormalFormDecider}^{\uparrow}$$

before it may read the closed typed-term endpoint row. The handoff exports a finite endpoint seal for bounded normal computation, not an unrestricted conversion decider.

Proof. Use Definition 4113.2. The bounded bridge row supplies the comparison boundary, the fuel-trace row supplies the finite computation trace, and the typed decider row supplies the typing-facing comparison surface.

Only after these rows are present can the endpoint row T be consumed. This makes the endpoint a replayable finite seal rather than a theorem erasing the fuel and typed-side hypotheses.

The obstruction, transport, replay, provenance, and naming rows preserve the same boundary. Thus the packet gives the decidability frontier a concrete endpoint landing while refusing the stronger global conversion claims. □

The Lean TasteGate follow-up for this chapter must provide explicit Nontrivial MetaCICConversionEndpointSeal$^{\uparrow}$ and FieldFaithful MetaCICConversionEndpointSeal$^{\uparrow}$ instances preserving $B, F, D, T, O, H, C, P, N$ as separate displayed rows.

Closure status for NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$ ·

Theory closure: TheoryGate(NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$, seed)

Scope closed: Definition 4113.1, Definition 4113.2, and Definition 4113.3 seed the finite endpoint seal connecting bounded normal computation to finished normal-form data for closed typed terms.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICConversionEndpointSeal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted closed beta-conversion, arbitrary-fuel completeness, host evaluator equality, quotient syntax equality, subject reduction, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded bridge exposure, fuel-trace exactness, typed normal-form decider compatibility, closed endpoint exposure, obstruction nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICConversionEndpointSealUp.

Constructive story: A MetaCICConversionEndpointSeal$^{\uparrow}$ packet is generated from BoundedNormalConversionBridge, MetaCICFuelTrace, TypedNormalFormDecider, closed typed-term endpoint, obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4114

A Concrete Naming Certificate for $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ packet records the finite comparison route between located metric completion, Bishop metric completion, constructive metric completion, and the general metric-completion comparison surface. It sits after chapter 2114, chapter 4023, chapter 1556, chapter 3553, chapter 301, chapter 126, chapter 291, and chapter 38. The chapter names one located completion route through displayed finite rows; it does not assert quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4114.1 (Bishop located-completion comparison carrier)**.** *A* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, B, Q, M, R, S, D, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedMetricCompletion}^{\uparrow}$ *route, carrying the located source and the displayed completion-facing windows.*
2. *B is the* $\mathsf{BishopMetricCompletion}^{\uparrow}$ *route, carrying the Bishop regular Cauchy presentation of the same located source.*
3. *Q is the* $\mathsf{ConstructiveMetricCompletion}^{\uparrow}$ *route, carrying the constructive metric completion packet over located metrics, regular Cauchy names, dyadic tolerances, and separated comparison.*
4. *M is the* $\mathsf{MetricCompletionComparison}^{\uparrow}$ *row comparing the visible completion presentations through paired dense embeddings, a common dense source, and finite round-trip reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row shared by the Bishop and constructive routes.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected before any regular rational readback or comparison map is named.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare the located, Bishop, constructive, and metric-comparison windows at finite precision.*
8. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, B, Q, M, R, S, D have been displayed.*
9. *H records componentwise* hsame *transport for the four completion routes, readback, window, tolerance, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a located completion request through the Bishop and constructive presentations, the metric-comparison row, the shared readback rows, and then the Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$*, the eight sibling chapters named above,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completions, selected representatives, countable choice, host equality of completion classes, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.BishopLocatedCompletionComparisonUp.BishopLocatedCompletionComparisonTasteGate`
`single carrier alignment`

Theorem 4114.2 (BishopLocatedCompletionComparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ *carrier* $K = (L, B, Q, M, R, S, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located completion route, Bishop metric completion route, constructive metric completion route, metric-completion comparison row, regular rational readback, stream windows, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local naming row;*

2. *located route comparison, requiring L, B, Q, M to share the same displayed located source before any completion-facing comparison is exported;*

3. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$ *readback compatibility, requiring S to be read before R and requiring both rows to be replayed through C before the comparison row M reaches the seal;*

4. *Real seal handoff, requiring A to remain downstream of L, B, Q, M, R, S, D;*

5. *ledger refusal for quotient completions, selected representatives, countable choice, host equality of completion classes, and ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

No obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4114.1. The projection exposes exactly the located, Bishop, constructive, and metric-comparison routes, together with the shared regular readback, stream-window, dyadic-tolerance, Real-seal, transport, replay, provenance, and naming rows.

Use the located route as the common source. The rows L, B, Q, M can be compared only after the located metric completion route has been displayed, so the Bishop and constructive presentations are read as finite routes over the same located completion request.

Next read the shared observation rows. The stream row S supplies the inspected finite windows, R reads them as regular rational data, and D records the tolerance ledger used by the comparison row. Replay by C preserves this order before the seal row A can be consumed.

Finally attach H, P, N. Componentwise transport, provenance, and local naming preserve the same finite route and add no coordinate for quotient completions, selected representatives, countable choice, host equality, or ambient Real completeness. Hence the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: A $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ consumer that reaches a comparison map or $\mathsf{Real}^{\uparrow}$ seal while omitting the located completion row, Bishop completion row,

constructive completion row, metric-comparison row, RegSeqRat readback, StreamName windows, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: located-metric-completion, bishop-metric-completion, constructive-metric-completion, metric-completion-comparison, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole comparison packet because each contributes only a completion presentation, comparison surface, readback row, window row, tolerance ledger, or seal, while this chapter binds them into one located completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCompletionComparison}^{\uparrow}$ packet is generated from LocatedMetricCompletion, BishopMetricCompletion, ConstructiveMetricCompletion, MetricCompletionComparison, RegSeqRat, StreamName, DyadicRatCore, Real, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4114.1 and Definition 4114.2 seed the finite located completion comparison route over Bishop, constructive, metric-comparison, readback, tolerance, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCompletionComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located route comparison, RegSeqRat and StreamName readback compatibility, Real seal handoff, transport replay, provenance non-escape, and ledger refusals for BEDC.Derived.BishopLocatedCompletionComparisonUp.

Chapter 4115

A Concrete Naming Certificate for $\mathsf{RiemannSumGauge}^{\uparrow}$

Following Bishop and Bridges on Riemann integration, the $\mathsf{RiemannSumGauge}^{\uparrow}$ packet records the finite tagged-gauge data visible before an integration consumer reads a Riemann-sum or Darboux-sum handoff. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the upper Darboux-sum boundary of chapter 4083, and the closed bounded interval Darboux-partition surface of chapter 3429. The packet names a finite gauge-and-tag carrier for later $\mathsf{Calculus}^{\uparrow}$ and $\mathsf{Integral}^{\uparrow}$ consumers; it does not import host real functions, selected tags, quotient equality of partitions, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4115.1 (Riemann sum gauge carrier)**.** *A* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, T, Q, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded* $\mathsf{Real}^{\uparrow}$ *interval row, including endpoint seals and the finite interval enclosure visible to the consumer.*
2. *T is the tagged finite partition row, recording cells, tags, endpoint reads, and the local tag-in-cell classifier.*
3. *Q is the positive rational gauge-radius row, read as displayed rational data rather than as an ambient open cover.*
4. *M is the mesh ledger, binding the tagged partition to the gauge-radius row and recording the visible mesh comparisons.*
5. *D is the Darboux/Riemann sum row, carrying the finite weighted values and the handoff to* **??** *and the displayed Darboux-partition surface.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for endpoint, tag, mesh, and value observations at the requested finite windows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same windows and rational gauge data.*
8. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every real-facing value or interval read.*

9. *H records componentwise* hsame *transport for interval, tagged partition, gauge, mesh, sum, window, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from gauge requests, tagged-sum reads, Darboux handoffs, or real-facing consumers through $I, T, Q, M, D, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, ***??***, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed finite tagged-gauge Riemann-sum history. It has no coordinate for a host function space, selected open cover, quotient equality of partitions, chosen limiting mesh, choice principle, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4115.2 (Riemann sum gauge NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier $G = (I, T, Q, M, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded interval, tagged finite partition, positive rational gauge-radius row, mesh ledger, Darboux/Riemann sum row, stream windows, regular rational readback, dyadic tolerance and Real seal row, transport, replay, provenance, and local name;*
2. *classifier transport, requiring tags, cells, endpoint observations, and gauge-radius comparisons to move only by the componentwise* hsame *row H;*
3. *mesh-gauge exactness, requiring M to bind the tagged partition T to the positive rational gauge-radius row Q before any tagged-sum consumer reads the mesh;*
4. *Darboux-sum handoff, requiring the finite sum row D to expose only the displayed* ***??*** *and* $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$ *routes;*
5. *Real seal nonescape, requiring every Real-facing value, endpoint, or interval read to pass through W, R, E after I, T, Q, M, D have been displayed;*
6. *transport and replay, requiring H and C to preserve the same finite tagged-gauge history;*
7. *provenance and naming, requiring P and N to package and name exactly the displayed rows;*
8. *ledger refusal, excluding host real functions, quotient equality of partitions, selected limiting meshes, choice, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Riemann-sum gauge obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RiemannSumGaugeUp.RiemannSumGaugeTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 4115.1. The projection exposes exactly the interval, tagged partition, gauge-radius, mesh, finite sum, window, readback, seal, and structural rows.

The mesh-gauge route is $T, Q \mapsto M \mapsto D$. A tag or cell may be consumed by the sum row only after the mesh ledger has related that tagged partition to the displayed positive rational gauge-radius row.

The Darboux handoff is then read from D. Since D only names the finite **??** and ClosedBoundedIntervalDarbouxP routes, it cannot add a host supremum, hidden function equality, or selected limiting mesh.

Finally, W, R, E supply finite window, regular rational readback, dyadic tolerance, and Real seal data, while H, C, P, N transport, replay, package, and name the same history. The excluded host objects have no carrier coordinate, so the listed obligations exhaust the local NameCert surface. □

Theorem 4115.3 (Riemann sum gauge tagged-real-partition source lock)**.** *For every accepted* $\mathsf{RiemannSumGauge}^{\uparrow}$ *carrier* $G = (I, T, Q, M, D, W, R, E, H, C, P, N)$*, the mesh ledger* M *is readable only after the tagged partition row* T *supplied by chapter 4296 has displayed its cells, tags, endpoint reads, and local tag-in-cell classifier. Every Darboux or gauge-integral consumer therefore receives the mesh route*

$$T, Q \longmapsto M \longmapsto D \longmapsto W, R, E, H, C, P, N$$

and cannot replace it with host partitions, selected mesh limits, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4115.1. The tagged partition coordinate T is listed before the mesh ledger M, and M is the first row that binds the tagged cells to the positive rational gauge-radius row Q.

Apply Definition 4115.2. Its mesh-gauge exactness clause requires M to bind T and Q before any tagged-sum consumer reads the mesh.

The tagged partition source itself is the carrier of chapter 4296, where cells, tags, endpoint observations, containment rows, window reads, and mesh exactness are displayed as finite rows. Reading M without that source row is therefore outside the accepted $\mathsf{RiemannSumGauge}^{\uparrow}$ packet.

Finally the Darboux/Riemann sum row D and the structural rows W, R, E, H, C, P, N preserve the same route. None of these rows adds host partitions, selected mesh limits, quotient function equality, or ambient Real completeness, so every downstream consumer remains locked to the tagged-real-partition source. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSumGauge}^{\uparrow}$ packet is generated from a closed bounded RealUp interval row, tagged finite partition row, positive rational gauge-radius row, mesh ledger, Darboux/Riemann sum row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$, seed)

Scope closed: Definition 4115.1 and Definition 4115.2 seed the finite tagged-gauge Riemann-sum surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalDarbouxPartition}^{\uparrow}$, **??**, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannSumGauge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real functions, quotient equality of partitions, selected limiting meshes, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, mesh-gauge exactness, Darboux-sum handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumGaugeUp.

Chapter 4116

A Concrete Naming Certificate for $\mathsf{RegularCauchyTransform}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyTransform}^{\uparrow}$ packet records the finite transform route between regular Cauchy names without committing to a map-specific analytic hypothesis. It sits over the regular rational readback of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the explicit modulus surface of chapter 3768, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a displayed transform certificate; it is not a quotient stream operation, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.

Definition 4116.1 (Regular Cauchy transform carrier)**.** *A* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, R, M, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *finite-window row.*
2. *R is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row attached to those windows.*
3. *M is the transform modulus row, assigning each requested output tolerance to a finite source-window obligation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the requested output radius, source radius, and slack comparisons used by M.*
5. *U is the output* $\mathsf{StreamName}^{\uparrow}$ *finite-window row produced by the displayed transform route.*
6. *Q is the output* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to U and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the output regular readback row has been displayed.*
8. *H records componentwise* hsame *transport for the source windows, source readback, transform modulus, dyadic ledger, output windows, output readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$S, R, M, D \longmapsto U, Q \longmapsto E.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTransform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected Cauchy representative, host function extensionality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached output seal.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTransformUp.RegularCauchyTransformTasteGate single carrier alignment`

Definition 4116.2 (Regular Cauchy transform classifier)**.** *The* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *classifier compares accepted carriers only through componentwise* hsame *transport of*

$$S, R, M, D, U, Q, E, H, C, P, N.$$

Thus source windows, source regular readback, transform modulus, dyadic tolerance accounting, output windows, output regular readback, terminal Real seal, replay, provenance, and local naming are all visible comparison data. No classifier clause compares hidden host functions, quotient streams, selected limits, or unrecorded completion witnesses.

Theorem 4116.3 (Regular Cauchy transform NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier*

$$T = (S, R, M, D, U, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, source regular readback, transform-modulus routing, dyadic tolerance accounting, output-window production, output regular readback, terminal Real-seal handoff, componentwise transport, displayed replay, provenance, and local naming through the displayed rows* $S, R, M, D, U, Q, E, H, C, P, N$.

Proof. Project the carrier of Definition 4116.1. The source side is visible as S, R, the tolerance route is visible as M, D, the output side is visible as U, Q, E, and the structural rows are visible as H, C, P, N.

Read the transform route in displayed order. A consumer first opens finite source windows, attaches them to the regular rational readback, reads the transform modulus and dyadic tolerance ledger, and only then obtains output windows and output regular readback.

Finally attach the terminal and structural rows. The Real seal E is downstream of Q, while H, C, P, N transport, replay, source, and name the same finite route. Because the carrier has no hidden host map, quotient stream equality, selected representative, choice schedule, or ambient completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 4116.4 (Regular Cauchy transform modulus route)**.** *Every output window admitted by a* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *packet is backed by the transform modulus row M, the dyadic tolerance ledger D, a finite source* $\mathsf{StreamName}^{\uparrow}$ *window S, and source* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R. Hence a Real or Completion consumer cannot read the output regular name through U, Q without first displaying the modulus-controlled source route.*

Proof. Begin with an admitted output window. Carrier projection exposes U as an output $\mathsf{StreamName}^{\uparrow}$ row, but U is downstream of the transform route rather than an independent stream source.

Trace the request through M and D. The modulus row names the source-window obligation for the requested output tolerance, and the dyadic ledger records the radii and slack comparisons needed to serve that obligation.

The source evidence is then S, R. The finite windows in S are attached to regular rational readback by R, so the output route cannot be justified by a selected limit or host convergence witness.

After U is produced, the output readback Q and terminal seal E remain downstream of the same route. The structural rows H, C, P, N preserve this order under transport and replay. □

Theorem 4116.5 (Regular Cauchy transform Real-seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *read accepted by a* $\mathsf{RegularCauchyTransform}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularModulus}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{C}$$

No accepted read bypasses the source regular name, transform modulus, dyadic tolerance ledger, output regular readback, componentwise transport, replay, provenance, or local naming rows.

Proof. Start with the terminal seal row E. In the carrier definition, E is reachable only after the output readback row Q, and Q is attached to the output finite-window row U.

Use the modulus-route theorem. The output window U is produced only after the source windows S, source readback R, transform modulus M, and dyadic ledger D have been displayed.

Finally read the structural rows. The hsame transport, Cont replay, Pkg provenance, and local NameCert preserve the same finite chain; they do not add a quotient stream, selected host representative, ambient completeness theorem, or second seal route. □

Falsifiable prediction: A $\mathsf{RegularCauchyTransform}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without first displaying source windows, source RegSeqRat readback, a transform modulus row, dyadic tolerance accounting, output windows, output RegSeqRat readback, transport, replay, provenance, and local naming is not reading this packet.

Independence witness: regseqrat, streamname, dyadicratcore, regular-modulus, real: no listed sibling supplies the whole transform packet because each contributes only readback, windowing, tolerance, modulus, or sealing data while this chapter binds the finite source-to-output route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTransform}^{\uparrow}$ packet is generated from source StreamName windows, source RegSeqRat readback, a transform modulus row, DyadicRatCore tolerance accounting, output StreamName windows, output RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4116.1, Definition 4116.2, Definition 4116.3, Definition 4116.4, and Definition 4116.5 seed the finite transform route between regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host function equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, classifier transport, modulus routing, output-window production, Real-seal handoff, and ledger non-escape for BEDC.Derived.RegularCauchyTransformUp.

Chapter 4117

A Concrete Naming Certificate for $\mathsf{PoincareDiskDynamics}^{\uparrow}$

The $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet realizes the disk-dynamics fixed-end reading of section 9463.5 as finite BEDC data. It consumes the phase-transport route of chapter 3455, the metric surface of chapter 3474, disk coordinates from chapter 377, finite evolution rows from chapter 195, and boundary-circle consumers from chapter 58. Its distinguishing carrier binds disk-map evolution to a preserved far-end boundary row; it is not a completed hyperbolic plane, physical cosmology, global clock, quotient boundary, or internal object named T.

Definition 4117.1 (Poincare disk dynamics carrier)**.** *A* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, \Phi, G, B, F, V, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the inspected disk points.*
2. *Φ is the disk-map row, recording the finite boundary-preserving transport applied to those points.*
3. *G is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row used only as displayed four-point or distance-comparison data.*
4. *B is the* $\mathsf{S}^{1\uparrow}$ *boundary-consumer row naming the fixed far-end commitment.*
5. *F is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *provenance row linking visible phase movement, disk coordinates, reversible transport, fiber provenance, and refusal data.*
6. *V is the* $\mathsf{DynSystem}^{\uparrow}$ *finite-evolution row for replaying the disk-map orbit.*
7. *A is the audit-refusal row excluding an interior fixed point requirement, host hyperbolic geometry, physical light-speed derivation, global observer time, subject-substance, and an internal* T *object.*
8. *H records componentwise* hsame *transport for the disk, map, metric, boundary, phase-provenance, dynamics, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through disk coordinates, disk-map evolution, metric readback, boundary preservation, phase provenance, and refusal.*
10. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskDynamics}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$ *row.*

The carrier compares packets only through these rows. It has no coordinate for a host Poincare disk model, quotient ideal boundary, global physical evolution law, completed metric geometry, or object-level T*.*

Lean4 ✓ `BEDC.Derived.PoincareDiskDynamicsUp.PoincareDiskDynamicsTasteGate single carrier alignment`

Theorem 4117.2 (Poincare disk dynamics NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *carrier*

$$D = (U, \Phi, G, B, F, V, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, disk-map exposure, hyperbolic metric readback, fixed far-end boundary preservation, phase-transport provenance, finite dynamic replay, refusal exactness, componentwise transport, displayed continuation, package provenance, and local naming through the rows* $U, \Phi, G, B, F, V, A, H, C, P, N$*.*

Proof. Project the carrier of Definition 4117.1. The visible geometric data are the disk coordinates U, disk-map row Φ, metric-read row G, boundary row B, phase-provenance row F, and finite-evolution row V.

Read the fixed-end condition next. The boundary row B is preserved by the displayed disk-map route, while F records the phase-transport provenance that keeps visible movement distinct from source collapse or an internal T object.

The dynamic replay is finite. The row V supplies only the inspected orbit cells, and C replays those cells together with the disk, map, metric, boundary, and refusal rows.

Finally attach H, P, N. Componentwise transport, provenance, and local naming preserve exactly the displayed rows, so the NameCert surface cannot export host hyperbolic geometry, a quotient boundary, global time, physical constants, or subject-substance. □

Falsifiable prediction: If a disk-dynamics consumer preserves visible disk coordinates but drops either the boundary row, the phase-transport provenance row, or the refusal row, it is not reading a $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet.

Independence witness: poincare-disk-phase-transport, hyperbolic-metric, unitdisk, dynsystem, s1: no listed sibling is bijective to $\mathsf{PoincareDiskDynamics}^{\uparrow}$ because this packet jointly carries disk coordinates, disk-map evolution, metric readback, fixed far-end boundary preservation, phase provenance, dynamic replay, refusal, transport, provenance, and local naming.

The Lean follow-up for this packet must provide explicit $\mathsf{Nontrivial}\ \mathsf{PoincareDiskDynamics}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PoincareDiskDynamics}^{\uparrow}$ TasteGate instances. Those instances must keep $U, \Phi, G, B, F, V, A, H, C, P, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PoincareDiskDynamics}^{\uparrow}$ packet is generated from UnitDisk coordinates, a disk-map row, HyperbolicMetric readback, SOne boundary consumption, PoincareDiskPhaseTransport provenance, DynSystem finite evolution, refusal exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4117.1 and Definition 4117.2 seed the finite disk-map, metric, fixed-boundary, phase-provenance, and dynamic-replay route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed Poincare disk geometry, host hyperbolic metric models, quotient boundary equality, global observer time, physical light-speed derivation, internal T, subject-substance, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk-map exposure, metric readback, fixed-boundary preservation, phase-provenance routing, dynamic replay, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.PoincareDiskDynamicsUp.

Chapter 4118

A Concrete Naming Certificate for $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$

Following the finite minimum step in the Heine–Cantor proof of chapter 9237, the $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ packet records the compact-net lower-bound fold as a finite BEDC carrier. It reads the $\mathsf{ProbeBundle}$ spine supplied by compactness, the $\mathsf{Rat}^{\uparrow}$ radius histories attached to bundled centers, the order readback exposed through $\mathsf{TotalOrder}^{\uparrow}$, the source rows of $\mathsf{Compact}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$, the pointwise graph rows of $\mathsf{ContinuousMap}^{\uparrow}$, and the downstream $\mathsf{UniformContinuityModulus}^{\uparrow}$ consumer. The packet names the fold as finite induction over displayed probes, not as a host minimum over a semantic finite set.

The fold is the finite lower-bound row consumed by chapter 388 and by the uniform-modulus handoff, while remaining only a $\mathsf{ProbeBundle}$ and $\mathsf{Rat}^{\uparrow}$ finite induction surface.

Definition 4118.1 (ProbeBundleFiniteMinimumFold carrier)**.** *A* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, G, \mathcal{P}, R, O, A, B, U, H, C, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *source row carrying the finite-net witness.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *source-distance row used by the compact-uniform-continuity argument.*
3. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the displayed graph.*
4. *$\mathcal{P}$ is the nonempty* $\mathsf{ProbeBundle}$ *spine indexing the compact-net centers.*
5. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *radius histories, one for each bundled probe.*
6. *O is the* $\mathsf{TotalOrder}^{\uparrow}$ *readback row used to compare the positive rational radius histories.*
7. *A is the accumulator ledger produced by primitive recursion over $\mathcal{P}$.*
8. *B is the certified lower-bound row saying that the final accumulator is below every displayed radius history in R.*
9. *U is the* $\mathsf{UniformContinuityModulus}^{\uparrow}$ *handoff row that receives the accumulator as the tolerance-only precision.*
10. *H records componentwise* hsame *transport for the compact source, metric source, continuous-map row, bundle spine, rational candidates, order readback, accumulator, lower-bound certificate, handoff, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the finite fold from the bundle spine to the handoff row.*
12. *Q records* Pkg *provenance over* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{UniformContinuityModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$ *row.*

The carrier contains no host finite minimum, escaped probe outside $\mathcal{P}$, open-cover compactness object, quotient point equality, external modulus selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

```
Lean4 ✓  BEDC.Derived.ProbeBundleFiniteMinimumFoldUp.ProbeBundleFiniteMinimumFoldTasteGate
single carrier alignment
```

Theorem 4118.2 (ProbeBundleFiniteMinimumFold NameCert obligations)**.** *For every accepted* $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ *carrier*

$$F = (K, M, G, \mathcal{P}, R, O, A, B, U, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *compact, metric, and continuous-map source admission through K, M, G;*
2. *nonempty* ProbeBundle *spine admission for $\mathcal{P}$;*
3. *positive* $\mathsf{Rat}^{\uparrow}$ *radius admission for each bundled row of R;*
4. $\mathsf{TotalOrder}^{\uparrow}$ *comparison readback through O at each successor step of the fold;*
5. *finite-spine induction producing the accumulator row A from the rows of R;*
6. *lower-bound exactness, requiring B to certify that the accumulator is below every displayed radius history and to read no radius outside R;*
7. $\mathsf{UniformContinuityModulus}^{\uparrow}$ *handoff through U;*
8. *ledger non-escape, excluding host minima, escaped probes, open-cover compactness, quotient point equality, external modulus selectors, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 4118.1. The compact, metric, and continuous-map rows expose the source of the compact-uniform-continuity argument, while $\mathcal{P}$ is the only induction spine available to the fold.

Use primitive recursion over $\mathcal{P}$. At the initial row the accumulator reads the first positive $\mathsf{Rat}^{\uparrow}$ radius history in R; at each successor step the fold compares the accumulator with the next radius through the displayed $\mathsf{TotalOrder}^{\uparrow}$ row O and writes the retained lower row into A.

The lower-bound certificate B is attached to the same recursion. It records that the final accumulator is classifier-stably below every radius row already inspected in R, and it has no coordinate from which a radius outside the bundle can be read.

Finally the handoff row U gives the compact-uniform-continuity consumer a tolerance-only precision history. The structural rows H, C, Q, N transport, replay, source, and name the same finite route, so no host finite minimum, escaped probe, open-cover compactness object, quotient point equality, external modulus selector, or ambient real-completeness object can enter the certificate. □

Falsifiable prediction: A $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ consumer that obtains a UniformContinuityModulusUp precision without reading the displayed CompactUp source, MetricUp source, ContinuousMapUp row, ProbeBundle spine, RatUp radius histories, TotalOrderUp comparison row, accumulator ledger, lower-bound certificate, hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: compact, metric, continuousmap, rat, totalorder, uniformcontinuitymodulus: no listed sibling supplies the whole ProbeBundle finite minimum fold because each sibling contributes only compact source data, metric rows, graph rows, rational order data, or modulus handoff rows while this chapter binds the finite-spine induction surface.

Closure status for $\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}$.

Constructive story: A $\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}$ packet is generated from CompactUp finite-net rows, MetricUp source rows, ContinuousMapUp pointwise-modulus rows, a nonempty ProbeBundle spine, positive RatUp radius histories, a TotalOrderUp comparison readback row, an accumulator ledger, a lower-bound certificate, a UniformContinuityModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4118.1 and Definition 4118.2 seed the finite ProbeBundle minimum fold required by the compact-uniform-continuity lower-bound route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ProbeBundleFiniteMinimumFold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host finite minima, probes outside the displayed bundle, open-cover compactness, quotient point equality, external modulus selectors, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove source admission, bundle spine admission, positive rational admission, order readback, finite-spine induction, lower-bound exactness, modulus handoff, and ledger non-escape.

Chapter 4119

A Concrete Naming Certificate for $\mathsf{HausdorffDimension}^{\uparrow}$

Following Hausdorff and Falconer, the $\mathsf{HausdorffDimension}^{\uparrow}$ packet records the BEDC surface at which gauge-weighted finite and countable-looking cover data become a named scale-cover dimension read. It sits beside the scale counting route of chapter 3877, the finite refinement route of chapter 1853, the compact source boundary of chapter 129, and the dense Real-facing route of chapter 1809. The chapter names a gauge-cover packet; it does not overload $\mathsf{BoxDimension}^{\uparrow}$ with Hausdorff-measure obligations and does not assert a host measure theorem.

Definition 4119.1 (Hausdorff dimension gauge-cover carrier)**.** *A* $\mathsf{HausdorffDimension}^{\uparrow}$ *carrier over a compact metric source* X *is a finite* BHist *packet*

$$H = (M, K, G, C, W, R, S, D, T, P, N)$$

with the following displayed rows:

1. M *is the inherited* $\mathsf{Met}^{\uparrow}$ *row on* X*, including the visible distance classifier used to read ball diameters.*
2. K *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing finite-net and complete-limit data before any gauge cover is inspected.*
3. G *is the gauge row. It records the dyadic scale exponent, positive radius bound, and rational weight expression used by the Hausdorff gauge read.*
4. C *is the finite visible cover row. Each cover member carries a center history, a radius history, membership evidence, and a diameter bound readable through* M *and* K*.*
5. W *is the gauge-weight ledger. It assigns to each displayed cover member its rational gauge contribution and records the finite partial sum.*
6. R *is the refinement route. It records how a smaller visible gauge cover refines the previous one without choosing a host-minimal cover.*
7. S *is the separability route through* $\mathsf{RealSeparability}^{\uparrow}$*, used when the compact metric source is Real-facing and cover centers must be read from dense windows.*
8. D *is the dimension-threshold ledger. It records rational lower and upper threshold comparisons for gauge-weight decay across displayed scales.*

9. *T stores componentwise* hsame *transport and displayed* Cont *replay for metric points, gauge scales, cover members, gauge weights, refinement rows, dense-route data, and threshold comparisons.*

10. *P is the* Pkg *provenance over* $\mathsf{HausdorffDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$ *row.*

The carrier has no coordinate for an arbitrary outer measure, a selected optimal cover, countable choice over covers, host logarithms, quotient compacta, ambient Euclidean embeddings, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HausdorffDimensionUp.HausdorffDimensionTasteGate single carrier alignment`

Theorem 4119.2 (Hausdorff dimension NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffDimension}^{\uparrow}$ *carrier*

$$H = (M, K, G, C, W, R, S, D, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric and compact rows, gauge positivity and scale exposure, finite cover membership, diameter comparison, gauge-weight summation, refinement transport, dense Real-facing routing when present, threshold comparison, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, K, G, C, W, R, S, D, T, P, N$.*

Proof. Project the carrier of Definition 4119.1. The source rows M and K provide the metric and compact finite-net surface, while G supplies the positive gauge scale and the rational exponent data used by the dimension read.

For a fixed gauge, C displays the cover members and W displays their gauge contributions. The finite sum is therefore a packet row, not an invocation of an ambient outer-measure construction.

The row R is the only refinement route between gauges. It transports displayed covers to smaller displayed covers and keeps the weight ledger attached to the same finite source surface.

The dense-source route S, threshold ledger D, transport and replay row T, provenance row P, and naming row N preserve the same visible data. Since the carrier contains no coordinate for selected optimal covers, countable choice, host logarithms, outer measures, quotient compacta, Lean verification, or bridge checking, the listed rows exhaust the local NameCert obligations. □

Theorem 4119.3 (Hausdorff dimension scale-cover route)**.** *Every scale-cover read exported by an accepted* $\mathsf{HausdorffDimension}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow} \longmapsto G, C, W, R \longmapsto S, D, T, P, N.$$

The route may be compared with $\mathsf{BoxDimension}^{\uparrow}$ *only at the shared metric, compact, dyadic-scale, finite-cover, refinement, counting, and dense-route rows, and with* $\mathsf{CoveringDimension}^{\uparrow}$ *only at the finite cover-refinement surface. It does not identify Hausdorff gauge weights with box cover counts, does not turn covering dimension into a gauge measure, and does not import open-cover compactness or selected subcovers.*

Proof. Start from the metric and compact source rows. A scale-cover consumer must first read M and K, so the cover is anchored in the same compact metric finite-net boundary used by chapter 129.

Open the gauge-cover rows. The gauge row G, visible cover row C, and gauge-weight ledger W provide the Hausdorff-specific data; the refinement row R records how these rows persist as the scale decreases.

Compare only along shared surfaces. The $\mathsf{BoxDimension}^{\uparrow}$ sibling shares finite scale-cover and counting language but lacks the gauge-weight ledger, while $\mathsf{CoveringDimension}^{\uparrow}$ shares finite refinement language but lacks the gauge-threshold comparison row.

Finally route Real-facing centers through S and close the threshold read through D, T, P, N. The packet therefore exports a concrete gauge-cover dimension route without adding host outer measures, selected cover families, open-cover compactness, quotient equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4119.1, Definition 4119.2, and Definition 4119.3 seed the gauge-cover dimension packet over compact metric sources, finite refinement rows, RealSeparability dense routes, and box-dimension comparison surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Hausdorff measure existence, exact numerical dimension values, countable cover choice, selected optimal covers, host logarithm analysis, ambient Euclidean embeddings, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge positivity, finite cover membership, diameter comparison, gauge-weight summation, refinement routing, threshold comparison, and RealSeparability dense-route coverage for BEDC.Derived.HausdorffDimensionUp.

Constructive story: A $\mathsf{HausdorffDimension}^{\uparrow}$ packet is generated from a metric row, compact metric source row, gauge row, visible cover row, gauge-weight ledger, refinement route, dense Real-facing route, threshold ledger, componentwise transport and replay, provenance, and a local NameCert row.

Chapter 4120

A Concrete Naming Certificate for $\mathsf{EffectiveLimitOperator}^{\uparrow}$

Following Weihrauch computable analysis, the $\mathsf{EffectiveLimitOperator}^{\uparrow}$ packet records the represented-space limit operation as a finite BEDC handoff from Cauchy names to a sealed represented output. It sits over represented-space source rows from chapter 1494, Cauchy representation rows from chapter 1680, stream-window observations from chapter 126, regular rational readback from chapter 301, dyadic tolerance control from chapter 291, and the terminal real-name seal of chapter 38. The packet makes the limit operator visible as displayed finite convergence data, not as an ambient completeness principle.

Definition 4120.1 (EffectiveLimitOperator carrier)**.** *An* $\mathsf{EffectiveLimitOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, A, S, M, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the represented-space source row, including the representation interface through which names are read.*
2. *A is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row presenting the input name as a Cauchy object over X.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of inspected name windows.*
4. *M is the effective convergence and modulus ledger assigning each requested tolerance to a finite tail-window obligation.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations made through the windows in S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the requested precision with the displayed tail budget.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the represented source, Cauchy representation, stream windows, modulus ledger, regular readback, and dyadic tolerance rows have been displayed.*
8. *H records componentwise* hsame *transport for the represented source, Cauchy source, stream windows, modulus ledger, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through A, S, M, R, D to the sealed output row E.*

10. *P records* Pkg *provenance over* EffectiveLimitOperator$^{\uparrow}$, RepresentedSpace$^{\uparrow}$, CauchyRepresentation$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{EffectiveLimitOperator}^{\uparrow}}$ *row.*

The carrier contains no selected limit, quotient stream equality, countable-choice selector, host real equality, or ambient completeness row for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.EffectiveLimitOperatorUp.EffectiveLimitOperatorTasteGate single carrier alignment`

Theorem 4120.2 (EffectiveLimitOperator NameCert obligations)**.** *For every accepted* EffectiveLimitOperator$^{\uparrow}$ *carrier*

$$L = (X, A, S, M, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{EffectiveLimitOperator}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *represented-source admission for the displayed* RepresentedSpace$^{\uparrow}$ *and* CauchyRepresentation$^{\uparrow}$ *rows;*
2. *stream-window coverage, requiring every tolerance request admitted by M to name a finite inspected window in S;*
3. *effective-modulus replay, requiring the convergence ledger M to route each requested tolerance to the displayed regular rational readback R;*
4. *dyadic tolerance exactness, requiring every comparison used by the limit handoff to read exactly the row D beside R;*
5. *real-seal handoff, requiring the* Real$^{\uparrow}$ *seal E to be consumed only after X, A, S, M, R, D have been displayed;*
6. *structural stability through the componentwise* hsame, *displayed* Cont, Pkg, *and local* NameCert *rows;*
7. *ledger non-escape, excluding selected limits, quotient stream equality, countable choice, host real equality, and ambient* Real$^{\uparrow}$ *completeness from the packet.*

Proof. Project the carrier in Definition 4120.1. The rows X, A, S, M, R, D, E display the represented source, Cauchy presentation, stream windows, effective convergence ledger, regular rational readback, dyadic tolerance comparison, and terminal real seal.

A tolerance request first reads the ledger M. The ledger names a finite window in S, sends the observed entries through the regular readback row R, and compares the requested budget against the dyadic tolerance row D.

The seal row E is therefore reachable only after this displayed route has been followed. The transport, replay, provenance, and local naming rows H, C, P, N attach the same route to classifier-stable BEDC data.

Since the carrier has no coordinate for a selected limit, quotient stream equality, countable-choice selector, host real equality, or ambient completeness of Real$^{\uparrow}$, a consumer cannot obtain any of those objects from the local certificate surface. □

Falsifiable prediction: A EffectiveLimitOperator$^{\uparrow}$ consumer that reaches a Real$^{\uparrow}$ limit seal without reading the displayed represented-space source, CauchyRepresentation source, StreamName

windows, effective modulus ledger, RegSeqRat readback, DyadicRatCore tolerance row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: representedspace, cauchyrepresentation, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole effective-limit-operator packet because each sibling contributes only source representation, Cauchy naming, windowing, readback, tolerance, or sealing data while this chapter binds the finite operator handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveLimitOperator}^{\uparrow}$ packet is generated from a RepresentedSpace source row, a CauchyRepresentation source row, StreamName windows, an effective convergence and modulus ledger, RegSeqRat regular rational readbacks, a DyadicRatCore tolerance row, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4120.1 and Definition 4120.2 seed the finite represented-space limit-operator handoff over Cauchy names, stream windows, regular readback, dyadic tolerance, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveLimitOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove represented-source admission, stream-window coverage, effective-modulus replay, dyadic tolerance exactness, Real seal handoff, structural stability, and ledger non-escape.

Chapter 4121

A Concrete Naming Certificate for LocatedCauchyFilterLimit$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Bourbaki–Kelley terminology of filters and filter bases, the LocatedCauchyFilterLimit$^{\uparrow}$ packet records the localized L10 limit route in which a located filter-basis row feeds the existing CauchyFilterLimit$^{\uparrow}$ and CauchyFilterRegularization$^{\uparrow}$ packets before any Real$^{\uparrow}$ seal is read. It sits after chapter 1558, chapter 4110, chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names the finite located completion-filter route, not quotient equality of filters, selected representatives, host real equality, countable choice, or ambient completeness.

Definition 4121.1 (LocatedCauchyFilterLimit carrier)**.** *A* LocatedCauchyFilterLimit$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, F, G, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *B is the located filter-basis row, naming the finite located neighborhoods and common-refinement data used by the completion-filter request.*
2. *F is the* CauchyFilterLimit$^{\uparrow}$ *row, consuming the displayed basis route before a limit-facing seal is read.*
3. *G is the* CauchyFilterRegularization$^{\uparrow}$ *row, regularizing the same filter route into finite stream, rational, dyadic, and seal data.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row inspected by the located request.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for located comparison and limit-scale accounting.*
7. *A is the* Real$^{\uparrow}$ *seal row reached only after B, F, G, S, R, D have been replayed.*
8. *H records componentwise* hsame *transport for the located basis, limit, regularization, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a located completion-filter request through $B, F, G, S, R, D, A, H, P, N$.*
10. *P is the* Pkg *provenance over* LocatedCauchyFilterLimit$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, CauchyFilterRegularization$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filters, selected limit representatives, host real equality, countable-choice locators, hidden cofinal subsequences, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.LocatedCauchyFilterLimitUp.LocatedCauchyFilterLimitUp`
Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterLimitUp.LocatedCauchyFilterLimitTasteGate single carrier alignment`

Theorem 4121.2 (LocatedCauchyFilterLimit NameCert obligations)**.** *For every accepted* LocatedCauchyFilterLimi *carrier*

$$L = (B, F, G, S, R, D, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the located filter-basis row,* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *row,* $\mathsf{CauchyFilterRegularization}^{\uparrow}$ *row, finite stream windows, regular rational readback, dyadic tolerance ledger, Real seal, transport, continuation, provenance, and local naming row;*
2. *located-basis exposure, requiring B to be displayed before the limit row F or the regularization row G can be consumed;*
3. *completion-filter compatibility, requiring F and G to read the same finite* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows before the* $\mathsf{Real}^{\uparrow}$ *seal is reached;*
4. *seal non-escape, requiring A to remain downstream of B, F, G, S, R, D;*
5. *ledger refusal for quotient filters, selected representatives, hidden cofinal choices, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4121.1. The projection exposes the located basis row B, the limit row F, the regularization row G, the shared observation rows S, R, D, the seal row A, and the structural rows H, C, P, N.

The located row is read first. The continuation row C sends any completion-filter request through the displayed located neighborhoods and common refinements before the request can inspect the $\mathsf{CauchyFilterLimit}^{\uparrow}$ or $\mathsf{CauchyFilterRegularization}^{\uparrow}$ surfaces.

The shared observation rows bind the two sibling packets. Both the limit row and the regularization row consume the same finite stream windows, regular rational readbacks, and dyadic tolerance ledger before reaching A.

Finally attach transport, provenance, and local naming. The rows H, P, N preserve only the displayed route, so the carrier has no coordinate for quotient filters, selected representatives, hidden cofinal schedules, countable choice, host equality, or ambient completeness. □

Falsifiable prediction: A $\mathsf{LocatedCauchyFilterLimit}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal while omitting the located filter-basis row, CauchyFilterLimit row, CauchyFilterRegularization row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, componentwise hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: cauchyfilterlimit, cauchy-filter-regularization, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole located completion-filter packet because each contributes only a limit row, regularization row, window row, rational readback, tolerance ledger, or seal, while this chapter binds them to a located filter-basis route.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyFilterLimit}^{\uparrow}$ packet is generated from a located filter-basis row, CauchyFilterLimit row, CauchyFilterRegularization row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4121.1 and Definition 4121.2 seed the finite located Cauchy-filter limit route over limit, regularization, stream, rational, tolerance, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected representatives, hidden cofinal choices, countable choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-basis exposure, completion-filter compatibility, Real seal nonescape, transport replay, provenance nonescape, and ledger refusals for BEDC.Derived.LocatedCauchyFilterLimitUp.

This hub routes the pre-separated universal-property packet through chapter 1614, chapter 2237, chapter 1506, chapter 397, and chapter 3351.

Chapter 4122

A Concrete Naming Certificate for PseudometricCompletionUniversalProperty$^{\uparrow}$

Following Bishop and Bridges and Bourbaki–Kelley completion terminology, the PseudometricCompletionUniversalPr packet records the finite universal-property route for completing a pre-separated distance surface. It sits after the Pseudometric$^{\uparrow}$ carrier of chapter 1614, the separated classifier route of chapter 2004, the separated-completion handoff of chapter 2237, the metric-completion surface of chapter 1506, and the extension ledger of chapter 397. Its finite observations are read through chapter 126, chapter 301, chapter 291, and the terminal seal of chapter 38. The dense-source map row is the forward-facing consumer of chapter 3351, because extension existence and uniqueness must read dense embedding data before the completion-facing map row is compared. The chapter supplies the L10 bridge from pre-separated distance data to extension uniqueness without quotient completions, selected representatives, selected limits, host function equality, or countable choice.

Definition 4122.1 (PseudometricCompletionUniversalProperty carrier)**.** *A* PseudometricCompletionUniversalProp *carrier is a finite* BHist *packet*

$$U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$$

with the following displayed rows:

1. *P is the source* Pseudometric$^{\uparrow}$ *packet, carrying the point rows, displayed distance witness, zero-distance boundary, dyadic tolerance ledger, finite windows, regular rational readback, and terminal real distance seal.*
2. *Q is the* SeparatedQuotientMetric$^{\uparrow}$ *classifier row that consumes zero-distance evidence from P before any separated metric read is exposed.*
3. *S is the* SeparatedPseudometricCompletion$^{\uparrow}$ *handoff row that routes the source pseudometric through separated completion data while keeping the pre-separated source visible.*
4. *M is the* MetricCompletion$^{\uparrow}$ *row naming the completion-facing separated metric packet consumed by extension readers.*
5. *V is the* CompletionUniversalProperty$^{\uparrow}$ *row that supplies the finite extension-existence and uniqueness ledgers over completion data.*
6. *W is the* StreamName$^{\uparrow}$ *finite-window schedule through which source points, distance approximants, and dense observations are inspected.*

7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the windows in W and to the dense source observations used by the extension route.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for zero-distance, triangle, and extension-precision comparisons before the real seal is read.*
9. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after W, R, D display the requested approximation data.*
10. *F is the dense-source map row whose finite graph and modulus reads are to be extended from the pseudometric source.*
11. *E is the extension-existence ledger, a displayed route from $P, Q, S, M, V, W, R, D, A, F$ to a completion-facing map row.*
12. *L is the uniqueness ledger comparing two extension rows whenever they agree on the same separated classifier, dense-source windows, regular readback, dyadic tolerance, real seal, and dense map rows.*
13. *H records componentwise* hsame *transport for the source pseudometric, separated classifier, separated completion, metric completion, universal-property row, window schedule, regular readback, dyadic tolerance, real seal, map row, extension ledger, uniqueness ledger, continuation, provenance, and local naming rows.*
14. *C records displayed* Cont *replay of the route*
$$P, Q, S, M, V, W, R, D, A, F \longmapsto E \longmapsto L.$$
15. *N is both the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedQuotientMetric}^{\uparrow}$, $\mathsf{SeparatedPseudometricCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows and the local* $\mathsf{NameCert}_{\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion classes, selected representatives, selected limits, host function equality, countable-choice schedules, trichotomy, or ambient completeness.

Theorem 4122.2 (PseudometricCompletionUniversalProperty NameCert obligations)**.** *For every accepted* $\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}$ *carrier $U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$, the local* $\mathsf{NameCert}_{\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source pseudometric, separated classifier, separated completion handoff, metric completion row, universal-property row, finite windows, regular readback, dyadic tolerance, real seal, dense map, extension ledger, uniqueness ledger, transport, replay, provenance, and local naming rows;*
2. *separated-source exposure, requiring zero-distance evidence from P to pass through Q and S before the metric-completion row M is read;*
3. *dense-readback exposure, requiring every extension request to display W, R, D, A before the dense map row F is repacked as an extension;*
4. *extension existence, requiring E to be read only after $P, Q, S, M, V, W, R, D, A, F$ have been displayed;*
5. *extension uniqueness, requiring L to compare candidate extension rows over the same separated source, dense windows, readback rows, real seal, and dense map data;*

6. *non-escape, excluding quotient completions, selected representatives, selected host limits, host function equality, countable choice, trichotomy, and ambient completeness from every exported consumer route.*

No pseudometric-completion universal-property obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4122.1. The accepted packet lists the source pseudometric, separated classifier, separated completion handoff, completion row, universal-property row, dense readback rows, extension and uniqueness ledgers, and structural support rows.

Separated-source exposure follows from the order P, Q, S, M. The source distance surface remains pre-separated in P until the zero-distance evidence is consumed by Q and the separated-completion handoff S.

Dense-readback exposure follows from W, R, D, A, F. The stream schedule, regular rational row, dyadic tolerance row, and real seal are displayed before the dense map row is used by an extension request.

Extension existence is the role of E, and uniqueness is the role of L. The replay row C places those ledgers after the same source, separated, completion, dense-readback, and map rows, while H transports only the displayed coordinates.

The non-escape clause is an inventory check on the carrier. Since no quotient completion class, selected representative, selected host limit, host function equality, countable-choice schedule, trichotomy row, or ambient completeness theorem appears among the coordinates, none can be exported by the local naming certificate. □

Theorem 4122.3 (PseudometricCompletionUniversalProperty extension uniqueness)**.** *Let* $U = (P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N)$ *and* $U' = (P, Q, S, M, V, W, R, D, A, F, E', L', H', C', N')$ *be accepted* $\mathsf{PseudometricCompletionUniversalProperty}^{\uparrow}$ *carriers with the same source pseudometric, separated classifier, separated completion handoff, metric completion row, universal-property row, finite windows, regular readback, dyadic tolerance, real seal, dense map row, and package provenance. If* E *and* E' *agree on every displayed finite read from* W, R, D, A *after the separated-source route* P, Q, S *has been consumed, then the two completion-facing extension rows are compared by the uniqueness ledgers* L *and* L'*.*

The comparison is a displayed hsame *ledger over the finite carrier rows. It is not host equality of maps, not equality of quotient-completion representatives, and not a selected-limit principle.*

Proof. Read the common source coordinates from the two accepted packets. The rows P, Q, S fix the pre-separated distance surface, the separated classifier, and the separated-completion handoff available to both extensions.

Next read the common completion and dense-source coordinates. The rows M, V, W, R, D, A, F fix the completion-facing target, the universal-property ledger, the finite window schedule, regular rational readback, dyadic tolerances, real seal, and dense map data.

The hypothesis compares E and E' on the displayed finite reads from W, R, D, A after the separated-source route has been consumed. Transport those reads through H, H' and replay them through C, C' so both extension rows are presented over the same source, completion, dense-readback, real-seal, and dense-map coordinates.

The uniqueness coordinates are exactly L and L'. Applying them gives the required displayed comparison of extension rows, with no appeal to host function equality, quotient representatives, selected limits, or countable choice. □

Theorem 4122.4 (PseudometricCompletionUniversalProperty non-escape)**.** *Every consumer read exported by an accepted* PseudometricCompletionUniversalProperty$^{\uparrow}$ *carrier factors through the finite route*

Pseudometric$^{\uparrow}$, SeparatedQuotientMetric$^{\uparrow}$, SeparatedPseudometricCompletion$^{\uparrow}$, MetricCompletion$^{\uparrow}$, CompletionUnivers

The packet exposes extension existence through E and extension uniqueness through L only after that route has been displayed. It exports no quotient completion class, selected representative, selected host limit, countable-choice schedule, host function equality, trichotomy decision, or ambient completeness theorem.

Proof. Project the carrier of Definition 4122.1. Its source and completion rows are P, Q, S, M, V, and its dense-readback rows are W, R, D, A.

Apply Definition 4122.2. The separated-source obligation forces a consumer through P, Q, S before M is read, and the dense-readback obligation forces a consumer through W, R, D, A before F, E, L are read.

Extension existence is then confined to the ledger E, while extension uniqueness is confined to L. The transport and replay rows H, C move only the displayed coordinates and preserve the same order.

The exclusions follow from the carrier inventory. Since the packet contains no coordinate for quotient completion classes, selected representatives, selected host limits, countable-choice schedules, host function equality, trichotomy, or ambient completeness, none of those objects can escape along a consumer route. □

Theorem 4122.5 (Pseudometric completion sibling route)**.** *Any accepted* PseudometricCompletionUniversalProperty *packet that exposes a universal readback must first route through* MetricCompletion$^{\uparrow}$, CauchyEquivalenceSetoid$^{\uparrow}$, *and* SeparatedCompletion$^{\uparrow}$. *The universal readback first consumes the Cauchy-equivalence setoid route of chapter 1820, then the metric completion route of chapter 1506, before extension existence or uniqueness can be read. The readback may then consume the displayed extension and uniqueness ledgers of the carrier, but it may not replace those sibling rows by quotient representatives, selected limits, host function equality, countable-choice schedules, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 4122.1. The rows $P, Q, S, M, V, W, R, D, A, F, E, L, H, C, N$ already place the source pseudometric before the separated classifier, separated-completion handoff, metric-completion row, and universal-property ledger.

Read the equality-facing route next. The CauchyEquivalenceSetoid$^{\uparrow}$ comparison is the visible setoid boundary for regular Cauchy data, and the sibling route of chapter 1820 must be consumed before the metric-completion route of chapter 1506. The SeparatedCompletion$^{\uparrow}$ row records the separated completion route before the metric-completion consumer reads M.

Finally apply the non-escape theorem, Definition 4122.4. Since the carrier exports only extension and uniqueness ledgers after those displayed sibling rows, a universal readback cannot be retyped as quotient representatives, selected limits, host map equality, countable choice, trichotomy, or ambient completeness. □

Closure status for NameCert$_{\text{PseudometricCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A PseudometricCompletionUniversalProperty$^{\uparrow}$ packet is generated from a source Pseudometric$^{\uparrow}$ row, a SeparatedQuotientMetric$^{\uparrow}$ classifier, a SeparatedPseudometricCompletion$^{\uparrow}$ handoff, a MetricCompletion$^{\uparrow}$ completion row, a CompletionUniversalProperty$^{\uparrow}$ extension surface, finite StreamName$^{\uparrow}$ windows, RegSeqRat$^{\uparrow}$ readback, DyadicRatCore$^{\uparrow}$ tolerance rows, a Real$^{\uparrow}$ seal, a dense map row, extension and uniqueness ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PseudometricCompletionUniversalProperty}\uparrow}$, seed)

Scope closed: Definition 4122.1, Definition 4122.2, Definition 4122.3, Definition 4122.4, and Definition 4122.5 seed the finite universal-property route from pre-separated distance data to extension uniqueness through the sibling completion rows.

Formal verification: FormalStatus(NameCert$_{\text{PseudometricCompletionUniversalProperty}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion construction, selected representatives, selected host limits, countable choice, host function equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated-source exposure, dense-readback exposure, extension existence, extension uniqueness, and non-escape for BEDC.Derived.PseudometricCompletionUniversalPropertyUp.

Chapter 4123

A Concrete Naming Certificate for $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$

Following Baire and Bishop–Bridges constructive analysis, the $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ packet records the finite choice-free boundary that Real and Completion consumers may use before any dense-open or Baire-style argument reaches a sealed real name. It sits between the dense-open packet of chapter 1542, the complete metric source of chapter 128, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Specker obstruction surface of chapter 1406. The packet exposes a displayed finite thread through those rows; it does not assert a host Baire theorem, countable choice, quotient stream equality, selected limits, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 4123.1 (BaireCategoryChoiceFree carrier)**.** *A* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, M, W, R, S, K, H, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open boundary row, read only through the finite schedule and nested-ball thread already named in chapter 1542.*
2. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row supplying the displayed complete-metric carrier, ball classifier, Cauchy-thread route, and limit-witness boundary.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected dense-open requests, metric balls, and terminal real-name observations before any consumer reads the boundary.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row converting the displayed windows into regular-rational tolerance data.*
5. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, M, W, R have been displayed.*
6. *K is the Specker boundary row. It records that nonconstructive selection or completeness requests are refused by comparison with the finite Specker-style obstruction surface, rather than discharged by countable choice.*
7. *H records componentwise* hsame *transport for B, M, W, R, S, K and the structural rows.*
8. *C records displayed* Cont *replay from dense-open requests through finite windows, regular-rational readback, the real seal, and the refusal row.*

9. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $B, M, W, R, S, K, H, C, P, N$. It has no coordinate for countable choice, an infinite selector, host set intersection, quotient stream equality, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4123.2 (BaireCategoryChoiceFree NameCert obligations)**.** *For every accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *carrier $Q = (B, M, W, R, S, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$ *surface consists exactly of the finite obligations:*

1. *carrier admission for the dense-open Baire row, complete-metric source row, finite windows, regular-rational readback, real seal, Specker refusal row, transport, replay, provenance, and local naming;*
2. *window coherence, requiring every dense-open request consumed from B to pass through the same* $\mathsf{StreamName}^{\uparrow}$ *window row W before the readback row R is visible;*
3. *metric-source coherence, requiring the complete-metric row M to supply the only Cauchy-thread and limit-witness route available to the packet;*
4. *real-seal exactness, requiring S to be downstream of B, M, W, R and not a detached host real object;*
5. *choice-free refusal, requiring K to block countable choice, hidden selectors, quotient stream equality, selected limits, and ambient completeness as local certificate data.*

Lean4 ✓ `BEDC.Derived.BaireCategoryChoiceFreeUp.TasteGate.BaireCategoryChoiceFreeTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4123.1. Its rows list the dense-open Baire boundary, complete metric source, finite windows, readback, real seal, Specker refusal row, transport, replay, provenance, and local naming, and no other row is available to the local certificate.

Window coherence follows by reading the $\mathsf{BaireCategory}^{\uparrow}$ schedule only through W. The regular-rational row R is downstream of that same finite window surface, so the carrier cannot replace it with an unlisted stream representative or a selector over all dense-open requests.

Metric-source coherence is supplied by the complete-metric row M. The Cauchy thread and limit-witness route are the displayed rows consumed from M, and the transport and replay rows H, C preserve that source without creating a second completeness datum.

Finally read the seal and refusal rows. The $\mathsf{Real}^{\uparrow}$ row S is exposed only after B, M, W, R, while K records the Specker-style obstruction to turning the packet into countable choice, quotient equality, a selected limit, or ambient completeness. Thus the five obligations exhaust the local NameCert surface. □

Theorem 4123.3 (BaireCategoryChoiceFree ProbeBundle induction)**.** *Every accepted probe-bundle consumer of a* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet advances by finite induction over the displayed window row W. At each step the consumer reads the current dense-open request from B, the complete-metric source row M, the regular-rational tolerance row R, and the structural rows H, C, P, N before it may inspect the next request. The induction has no branch that selects an infinite sequence of witnesses, identifies quotient streams, chooses a host limit, or invokes ambient completeness.*

Proof. Start with the finite window row W. It is the only row listing the requests visible to the consumer, so the induction domain is a displayed finite packet rather than a countable family.

For the base step, the consumer reads the initial dense-open and metric rows through B and M. The readback row R supplies the tolerance data for the inspected window, and H, C, P, N carry only transport, replay, provenance, and local naming.

For the inductive step, the next request is another entry of the same finite W row. The same B, M, R, H, C, P, N route is replayed for that entry, and the refusal row K prevents the step from being interpreted as a hidden selector or a host completeness argument.

Since every step is indexed by an entry already displayed in W, the probe-bundle induction remains finite and choice-free. The excluded analytic alternatives have no coordinate in the carrier. □

Theorem 4123.4 (BaireCategoryChoiceFree Real and Completion boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet must pass through the displayed route*

$$B, M, W, R, S, K, H, C, P, N$$

before reading any local NameCert surface. The route uses the $\mathsf{BaireCategory}^{\uparrow}$ *dense-open packet, the* $\mathsf{CompleteMetric}^{\uparrow}$ *source, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback, the* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{SpeckerSequence}^{\uparrow}$ *refusal boundary. It exports no countable choice principle, quotient stream, selected limit, host equality test, or ambient Real completeness theorem.*

Proof. Read the carrier first. The only Real-facing endpoint is the row S, and S is downstream of the Baire boundary B, complete metric row M, finite windows W, and regular-rational readback R.

Apply the NameCert obligations. They force every local naming read through the same carrier, window, metric-source, real-seal, and refusal rows, so a Completion-facing consumer cannot bypass the finite route by citing an ambient completed object.

Use the probe-bundle induction for consumers that inspect several dense-open requests. The induction is finite over W and reuses the same B, M, R, H, C, P, N route at every inspected entry.

The Specker refusal row closes the boundary. It records that requests for countable choice, quotient stream equality, selected limits, host equality, or ambient Real completeness remain outside the packet. Therefore every Real or Completion consumer reads exactly the displayed finite choice-free route before the local NameCert is available. □

Theorem 4123.5 (BaireCategoryChoiceFree complete-metric handoff)**.** *Every Completion-facing consumer of an accepted* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *packet must read the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row* M *together with*

$$B, W, R, S, K, H, C, P, N$$

before it may use the choice-free Baire packet. The handoff makes the complete-metric carrier, ball classifier, Cauchy-thread route, and limit-witness boundary visible as source data inside the same finite packet; it does not allow the consumer to replace M *by countable choice, a hidden selector, quotient stream equality, a selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4123.1. The only complete-metric source is the row M, and it is displayed alongside the dense-open Baire row B, finite windows W, regular-rational readback R, Real seal S, refusal row K, and structural rows H, C, P, N.

Use the NameCert obligations of Definition 4123.2. Their metric-source coherence row requires the Cauchy thread and limit-witness route to be consumed from M, while their window and real-seal rows keep M tied to the same $B, W, R, S, K, H, C, P, N$ packet.

Finally apply the Real and Completion boundary theorem, Definition 4123.4. A Completion-facing consumer must traverse the displayed finite route before any local certificate is available, so the complete-metric handoff is exactly the source row M read together with the remaining packet rows and not an ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ packet is generated from a BaireCategoryUp dense-open boundary row, a CompleteMetricUp source row, finite StreamNameUp windows, RegSeqRatUp readback, a RealUp seal, a SpeckerSequenceUp refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4123.1, Definition 4123.2, Definition 4123.3, and Definition 4123.4 seed the finite choice-free Baire boundary for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryChoiceFree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim the full classical Baire category theorem, countable choice, quotient stream equality, selected limits, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coherence, complete-metric source coherence, real-seal exactness, Specker refusal, transport, provenance, and local naming for BEDC.Derived.BaireCategoryChoiceFreeUp.

Chapter 4124

A Concrete Naming Certificate for $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$

Following Riemann–Darboux integration and Bishop–Bridges constructive analysis, the RiemannIntegrabilityModulu packet records the finite modulus data that turns a Darboux gap estimate into a Cauchy route for integral names. It sits after the equal-mesh surface of chapter 2033, the Darboux finite-sum carrier of chapter 1880, the integrability carrier of chapter 1960, the real seal of chapter 38, the regular rational readback of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet gives later integration chapters a displayed finite handoff from upper-lower Darboux gaps to Cauchy integral windows without importing selected integrals, quotient functions, open-cover compactness, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4124.1 (Riemann integrability modulus carrier)**.** *A* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, U, L, G, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded* $\mathsf{Real}^{\uparrow}$ *interval and bounded-function observation row consumed by the Darboux and Riemann-integrable surfaces.*
2. *U is the* $\mathsf{UniformPartition}^{\uparrow}$ *row selecting an equal-mesh finite partition at a requested dyadic tolerance scale.*
3. *L is the lower Darboux finite-sum row read over the same interval and partition windows.*
4. *G is the upper Darboux finite-sum row read over the same interval and partition windows.*
5. *D is the Darboux-gap ledger, recording a positive dyadic tolerance request and the displayed comparison $G - L$ below that request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the Darboux-gap ledger into a regular rational Cauchy window for the integral name.*
7. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every endpoint, sum, gap, and integral-window read.*
8. *H records componentwise* hsame *transport for interval, partition, lower sum, upper sum, gap, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from Darboux-gap requests, Riemann-integrable reads, or integral-window consumers through $I, U, L, G, D, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite constructive Darboux-to-Cauchy modulus history. It has no coordinate for a selected integral value, host function quotient, open cover of the interval, measure-completion object, choice principle, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RiemannIntegrabilityModulusUp.TasteGate.RiemannIntegrabilityModulusTasteGate`
`single carrier alignment`

Theorem 4124.2 (Riemann integrability modulus NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier $M = (I, U, L, G, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulu}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval, equal-mesh partition, lower Darboux sum, upper Darboux sum, gap ledger, regular rational readback, dyadic tolerance and Real seal, transport, replay, provenance, and local name;*

2. *partition coherence, requiring L and G to read the same I and U rows before the gap ledger D can be consumed;*

3. *Darboux-gap exactness, requiring D to expose only the displayed comparison between the upper and lower finite sums at the requested dyadic tolerance;*

4. *Cauchy-window readiness, requiring R to be read from D and E before any integral-window consumer receives a regular rational name;*

5. *Real seal nonescape, requiring every endpoint, sum, gap, or integral-window read to pass through E after the displayed finite Darboux rows have been exposed;*

6. *transport and replay, requiring H and C to preserve the same finite modulus history;*

7. *provenance and naming, requiring P and N to package and name exactly the displayed rows;*

8. *ledger refusal, excluding selected integrals, quotient functions, open-cover compactness, measure completion, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No Riemann-integrability modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4124.1. The projection exposes exactly the interval, partition, lower-sum, upper-sum, gap, readback, seal, and structural rows.

The Darboux route is $I, U \mapsto L, G \mapsto D$. Since L and G read the same interval and partition rows, the comparison in D is a comparison between two finite Darboux sums attached to one displayed partition history.

The Cauchy handoff is then read from D, R, E. The gap ledger supplies the dyadic tolerance comparison, the regular rational readback turns that comparison into a Cauchy window, and the Real seal records only the resulting finite read.

Finally, H, C, P, N transport, replay, package, and name the same history. The excluded host objects have no carrier coordinate, so the listed obligations exhaust the local NameCert surface. □

Theorem 4124.3 (Darboux-to-Cauchy handoff)**.** *Every accepted* $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ *carrier* $M = (I, U, L, G, D, R, E, H, C, P, N)$ *exports a finite route*

$$\mathsf{UniformPartition}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

in which a Darboux-gap request first reads the common partition row U*, then the paired lower and upper finite-sum rows* L, G*, then the gap ledger* D*, and only then the regular rational readback* R *and Real seal* E*. Any later Riemann-integral or completion-facing consumer of the integral name factors through this route and cannot bypass it by selecting an ambient integral value, quotienting functions, invoking an open-cover compactness theorem, completing a measure space, or using ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 4124.1. The only partition coordinate is U, and the only Darboux-sum coordinates are the paired lower and upper rows L and G.

The gap ledger D is defined after those rows, so any gap request must pass through the common partition and the paired finite sums before the tolerance comparison is available. The readback row R and the seal row E occur after D, hence the Cauchy-window read cannot precede the Darboux comparison.

The replay row C preserves that order for downstream consumers, while H, P, N preserve transport, provenance, and naming. Since no selected integral, quotient-function, open-cover, measure-completion, or ambient-completeness coordinate is present in the carrier, none can be used as an alternate route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannIntegrabilityModulus}^{\uparrow}$ packet is generated from a closed bounded RealUp interval row, UniformPartitionUp equal-mesh row, paired DarbouxSumUp lower and upper finite sums, Darboux-gap ledger, RegSeqRatUp readback, DyadicRatCoreUp tolerance ledger, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4124.1, Definition 4124.2, and Definition 4124.3 seed the finite constructive Darboux-to-Cauchy modulus route over $\mathsf{UniformPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrabilityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected integrals, quotient functions, open-cover compactness, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition coherence, Darboux-gap exactness, Cauchy-window readiness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegrabilityModulu

Chapter 4125

A Concrete Naming Certificate for $\mathsf{SequentialRealCompletion}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley's completion route, the $\mathsf{SequentialRealCompletion}^{\uparrow}$ packet records the standard sequential completion path from regular Cauchy names to the $\mathsf{Real}^{\uparrow}$ seal as finite BEDC data. It sits after the stream-window carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real-name seal of chapter 38, the completion interface of chapter 95, and the Cauchy-limit sealing routes used by $\mathsf{CauchyLimitSeal}^{\uparrow}$ consumers. The packet is finite regular-Cauchy completion data: it is not a quotient representative, not host equality of completed objects, not countable choice, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4125.1 (Sequential real completion carrier)**.** *A* $\mathsf{SequentialRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, Q, E, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected observations of the candidate regular Cauchy name.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the regular rational approximants visible in those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare window observations against the requested precision.*
4. *Q is the regular-Cauchy coherence row, recording tail agreement and modulus coverage only through displayed finite windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after W, R, D, Q have been displayed.*
6. *L is the* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *handoff row, recording that the same regular route is available to completion-facing consumers.*
7. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic ledger, coherence row, real seal, limit seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through $W, R, D, Q, E, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialRealCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{CauchyLimitSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $W, R, D, Q, E, L, H, C, P, N$. It has no coordinate for quotient representatives of Cauchy sequences, host equality of completed objects, countable choice over tails, an external limit selector, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentialRealCompletionUp.TasteGate.SequentialRealCompletionTasteGate single carrier alignment`

Theorem 4125.2 (Sequential real completion NameCert obligations)**.** *For every accepted* SequentialRealCompletion *carrier*

$$S = (W, R, D, Q, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, StreamName window admission through W, RegSeqRat readback through R, DyadicRatCore tolerance admission through D, regular-Cauchy coherence through Q, RealUp seal admission through E, CauchyLimitSeal handoff through L, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4125.1. The rows W, R, D, Q are the regular Cauchy source route, while E and L are the only real-facing and completion-facing endpoints.

The finite route starts with windows. A precision request reads the displayed $\mathsf{StreamName}^{\uparrow}$ row W, extracts the regular rational observations through R, and checks the dyadic tolerance data through D before the coherence row Q may be consumed.

The endpoint rows are downstream of that same route. The $\mathsf{Real}^{\uparrow}$ seal E and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ handoff L read only the displayed regular-Cauchy packet, and the structural rows H, C, P, N preserve the same packet under transport, replay, source, and naming.

Since no carrier coordinate contains quotient representatives, host equality, countable-choice schedules, external limit selectors, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 4125.3 (Sequential real completion Cauchy-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SequentialRealCompletion}^{\uparrow}$ *carrier factors through the finite chain*

$$\mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{DyadicRatCore}^{\uparrow} \to \mathsf{CauchyLimitSeal}^{\uparrow} \to \mathsf{Real}^{\uparrow}$$

as displayed by W, R, D, L, E and preserved by Q, H, C, P, N. The route exports a sealed regular-Cauchy completion packet only; it does not export quotient stream equality, a selected host real, countable choice, host equality of completions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Apply the obligation surface of Definition 4125.2. Any real-facing or completion-facing read must first expose the finite stream windows, regular rational readback, dyadic tolerance ledger, and coherence row.

The limit-seal row L is then read on the same route. It cannot be detached from W, R, D, Q, because its provenance and replay coordinates are the same H, C, P, N rows already attached to the carrier.

Finally the $\mathsf{Real}^{\uparrow}$ seal E is reached only after that limit-seal route is visible. Thus the consumer receives a finite regular-Cauchy completion packet and no quotient representative, selected host limit, countable-choice schedule, host equality, or ambient real-completeness coordinate. □

Falsifiable prediction: A $\mathsf{SequentialRealCompletion}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ endpoint without reading the displayed StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, regular-Cauchy coherence row, CauchyLimitSeal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, completion, cauchylimitseal: no listed sibling supplies the whole sequential real completion packet because each sibling contributes only windowing, regular readback, dyadic tolerance, real sealing, completion consumption, or limit-seal handoff while this chapter binds them into one finite regular-Cauchy completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialRealCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat regular rational readback, DyadicRatCore tolerance ledgers, regular-Cauchy coherence rows, a RealUp seal, a CauchyLimitSeal handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4125.1, Definition 4125.2, and Definition 4125.3 seed the finite regular-Cauchy completion route from sequential names to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient representatives, host equality of completed objects, countable choice, selected host limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, StreamName window coverage, RegSeqRat readback, DyadicRatCore tolerance exactness, regular-Cauchy coherence, CauchyLimitSeal handoff, RealUp seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.SequentialRealCompletionUp.

Chapter 4126

A Concrete Naming Certificate for MetaCICLocalJoinLedger$^{\uparrow}$

The MetaCICLocalJoinLedger$^{\uparrow}$ packet realizes the residual-substitution boundary of chapter 9378 as finite local-join data. It sits beside the local diamond witness of chapter 3595, the residual checker of chapter 3827, the confluence substitution boundary of chapter 3791, the confluence diamond boundary of chapter 4082, and the subject-reduction discharge socket of chapter 719. The carrier records local peaks, candidate joins, bounded checker handoffs, residual-substitution frontiers, obstruction-preserving readback, and TasteGate readiness without claiming full confluence, unrestricted substitution compatibility, subject reduction, evaluator equality, or kernel consistency.

Definition 4126.1 (MetaCIC local join ledger carrier)**.** *A* MetaCICLocalJoinLedger$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, A, B, R, S, O, G, H, C, P, N)$$

with the following displayed rows:

1. *K is the local peak row, naming the source term and the two outgoing reduction observations.*
2. *A is the candidate join row, recording the proposed common reduct and the two finite routes into it.*
3. *B is the bounded checker handoff row, exposing the closedness, support, branch compatibility, and variable-escape checks applied to the peak.*
4. *R is the residual frontier row, listing residual descendants and binder-depth data retained by the local join attempt.*
5. *S is the substitution-boundary row, recording the residual-substitution input that remains visible whenever the local join is conditional.*
6. *O is the obstruction readback row, naming the endpoints still outside the packet: unrestricted substitution compatibility, full closed Church–Rosser, arbitrary subject reduction, evaluator equality, and kernel consistency.*
7. *G is the* TasteGate *readiness row, reserving the later* Nontrivial *and* FieldFaithful *instance obligations for this finite carrier surface.*
8. *H records componentwise* hsame *transport for local peaks, candidate joins, bounded checks, residual frontiers, substitution boundaries, obstruction rows, TasteGate readiness, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a confluence-boundary query through $K, A, B, R, S, O, G, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, MetaCICConfluenceResi MetaCICConfluenceSubstitutionBoundary$^{\uparrow}$, $\mathsf{MetaCICConfluenceDiamondBoundary}^{\uparrow}$, SubjectReductionDischargeSoc BHist, hsame, Cont, TasteGate, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, A, B, R, S, O, G, H, C, P, N$. It contains no coordinate for a global confluence theorem, unrestricted substitution compatibility, subject reduction, quotient equality of terms, evaluator equality, or kernel consistency.

Theorem 4126.2 (MetaCIC local join ledger obligations)**.** *For every accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier*

$$J = (K, A, B, R, S, O, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$ *surface consists exactly of local-peak exposure through K, candidate-join exposure through A, bounded-checker handoff through B, residual-frontier exposure through R, substitution-boundary retention through S, obstruction readback through O, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4126.1. Rows K, A, B, R, S, O are the confluence-facing ledger data, and rows G, H, C, P, N are the structural certificate data.

The local join attempt first reads the peak and candidate join. Row K gives the finite source and branch observations, while A records only the proposed common reduct and its two routes.

The bounded checker and residual rows remain visible before the join can be consumed. Row B records the finite checks actually performed, R keeps residual descendants and binder-depth data, and S keeps the substitution-sensitive input that prevents the ledger from being read as an unconditional diamond.

Finally O preserves the open endpoints, while G, H, C, P, N reserve the TasteGate instance surface, transport the rows, replay the query, source the packet, and name it. Since no carrier coordinate contains full confluence, unrestricted substitution compatibility, subject reduction, evaluator equality, quotient equality, or kernel consistency, the listed rows exhaust the local NameCert obligations. □

Theorem 4126.3 (MetaCIC local join ledger bounded checker handoff)**.** *Every checker-facing consumer of an accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier factors through*

$$K, A, B, R, S, O, G, H, C, P, N$$

before it may consume the local join ledger. The bounded checker row B is a finite handoff over the displayed peak and residual rows; it is not a global confluence checker, not a subject-reduction proof, and not an evaluator-equality decision.

Proof. Apply the obligation surface of Definition 4126.2. It forces a checker-facing read through the local peak, candidate join, bounded-checker handoff, residual frontier, substitution boundary, and obstruction row.

The handoff is therefore scoped by K, A, R, S, O. It can report the checks listed in B, but it has no coordinate from which to manufacture an unrestricted substitution theorem, a subject-reduction theorem, or a global Church–Rosser result.

The structural rows G, H, C, P, N preserve the same finite route under TasteGate readiness, transport, replay, provenance, and local naming. Hence the checker handoff remains bounded local-join data. □

Theorem 4126.4 (MetaCIC local join ledger obstruction-preserving readback)**.** *Every confluence-audit read exported by an accepted* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *carrier preserves the obstruction row O beside the candidate join. A consumer may pass the local ledger to the local-diamond witness, residual checker, substitution-boundary packet, diamond-boundary packet, or subject-reduction discharge socket named above, but it cannot erase the residual-substitution boundary or reinterpret the ledger as full MetaCIC confluence.*

Proof. Start from the bounded checker handoff of Definition 4126.3. The local join is consumed only after the peak, join, checker, residual, substitution-boundary, and obstruction rows have been displayed together.

The sibling handoffs preserve that pairing. The local-diamond witness may read K and A, the residual checker may read B, R, S, O, the substitution-boundary and diamond-boundary packets may read the same residual-substitution interface, and the subject-reduction discharge socket may read the obstruction as a boundary rather than as a proof.

Since O is a carrier coordinate and no sibling handoff adds a hidden theorem row, every readback keeps the obstruction visible. The exported ledger is therefore finite local-join data, not unrestricted substitution compatibility, full confluence, subject reduction, evaluator equality, or kernel consistency. □

Theorem 4126.5 (MetaCIC local join ledger TasteGate readiness)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *at the displayed finite carrier surface*

$$K, A, B, R, S, O, G, H, C, P, N.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the local peak, candidate join, bounded checker handoff, residual frontier, substitution boundary, obstruction readback, transport, replay, provenance, and local NameCert data.

Lean4 ✓ `BEDC.Derived.MetaCICLocalJoinLedgerUp.MetaCICLocalJoinLedgerTasteGate single carrier alignment`

Proof. The readiness row G in Definition 4126.1 is tied to the whole local-join ledger rather than to a single obstruction field.

Nontriviality must exhibit an accepted packet with the displayed finite rows present. FieldFaithfulness must show that deleting any of $K, A, B, R, S, O, H, C, P, N$ changes the readable carrier surface. Thus the TasteGate burden remains at the same finite packet boundary as the NameCert obligations. □

Falsifiable prediction: A $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ confluence-audit read factors through displayed local-peak rows, candidate-join rows, bounded-checker handoff rows, residual-frontier rows, substitution-boundary rows, obstruction readback rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: metacic-local-diamond-witness, metacic-confluence-residual-checker, metacic-confluence-substitution-boundary, metacic-confluence-diamond-boundary, subject-reduction-discharge-socket: none is bijective with $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ because this packet jointly carries a local peak, candidate join, bounded checker handoff, residual frontier, substitution boundary, obstruction-preserving readback, TasteGate readiness, replay, transport, provenance, and local naming as one ledger surface.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ packet is generated from local-peak rows, candidate-join rows, bounded-checker handoff rows, residual-frontier rows, substitution-boundary rows, obstruction readback rows, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4126.1, Definition 4126.2, Definition 4126.3, Definition 4126.4, and Definition 4126.5 seed the finite local-join ledger for MetaCIC confluence audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICLocalJoinLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full confluence, unrestricted substitution compatibility, arbitrary subject reduction, evaluator equality, quotient equality of terms, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-peak exposure, candidate-join exactness, bounded-checker handoff, residual-frontier coverage, substitution-boundary retention, obstruction-preserving readback, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICLocalJoinLedgerUp.

Chapter 4127

A Concrete Naming Certificate for $\mathsf{HemicompactSpace}^{\uparrow}$

Following Kelley and Engelking on hemicompact spaces, the $\mathsf{HemicompactSpace}^{\uparrow}$ packet records compact exhaustion as explicit finite-name data. It sits over the metric carrier of chapter 57, the compact metric window surface of chapter 129, the finite stream schedule of chapter 126, the formal-ball neighborhood route of chapter 1510, and the real-radius seal of chapter 38. The carrier names a countable displayed exhaustion by compact windows; it does not infer open-cover compactness, select covers by choice, or assume that the whole host space is compact.

Definition 4127.1 (Hemicompact-space carrier)**.** *A* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, K, S, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the ambient* $\mathsf{Met}^{\uparrow}$ *row, including the point classifier, distance readback, and finite observation histories used by the exhaustion.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *exhaustion-window row; its entries name compact metric windows K_i as finite BEDC data and expose their local compact certificates only when the corresponding index is displayed.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *index schedule recording the inspected exhaustion indices, monotone window requests, and replayed finite prefixes.*
4. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *neighborhood-window row that reads local center-radius data inside a displayed compact window before any neighborhood consumer uses it.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *radius-seal row attached to the formal-ball windows and to the compact-window containment tests.*
6. *H records componentwise* hsame *transport for M, K, S, F, R, C, P, N and for the local classifier row.*
7. *C records displayed* Cont *replay from an index request through the compact window, formal-ball neighborhood window, real-radius seal, transport, provenance, and local name.*
8. *P is the* Pkg *provenance over* $\mathsf{HemicompactSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by replaying $M, K, S, F, R, H, C, P, N$. It has no coordinate for an open-cover compactness theorem, a choice-selected cover, a selected global compactification, host-space compactness, quotient equality of exhaustion sequences, or ambient completeness.

Lean4 ✓ `BEDC.Derived.HemicompactSpaceUp.HemicompactSpaceTasteGate single carrier alignment`

Theorem 4127.2 (HemicompactSpace NameCert obligations)**.** *For every accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier $E = (M, K, S, F, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, compact-window exhaustion, stream index schedule, formal-ball neighborhood windows, real-radius seals, transport, replay, provenance, and local name;*
2. *compact-window accountability, requiring every compactness read to pass through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *row K at an index supplied by the schedule S;*
3. *neighborhood-window accountability, requiring every local neighborhood read to pass through the* $\mathsf{FormalBall}^{\uparrow}$ *row F and the* $\mathsf{Real}^{\uparrow}$ *radius seals R over the same metric source M;*
4. *componentwise* hsame *transport and* Cont *replay stability for the same finite rows;*
5. *ledger refusal, stating that open-cover compactness, choice-selected covers, host compactness, quotient equality of exhaustion sequences, and ambient completeness are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the carrier in Definition 4127.1. The analytic rows are exactly the metric source, compact-window exhaustion, index schedule, formal-ball neighborhood row, and real-radius seals.

The compact and neighborhood obligations are ordered by the displayed schedule. A consumer first reads an index in S, then the compact metric window in K, and only then the formal-ball and real-radius rows F, R available over that window.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the excluded compactness and choice principles are absent from the carrier, no NameCert obligation can export them. □

Theorem 4127.3 (Hemicompact exhaustion window)**.** *Every accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *read of a compact exhaustion window factors through*

$$M, S, K, F, R, H, C, P, N.$$

In particular, the consumer receives an index from the $\mathsf{StreamName}^{\uparrow}$ *schedule, the corresponding* $\mathsf{CompactMetric}^{\uparrow}$ *window, the local* $\mathsf{FormalBall}^{\uparrow}$ *neighborhood rows, and the* $\mathsf{Real}^{\uparrow}$ *radius seals over the same* $\mathsf{Met}^{\uparrow}$ *source before the local* NameCert *row is consumed. The route exports no open-cover compactness proof, selected cover family, host-space compactness witness, quotient exhaustion equality, or ambient completeness.*

Proof. Start with the index schedule. The row S is the only coordinate that can name which exhaustion window is being inspected, so a compact-window read cannot begin with an unindexed compact object.

Read K, F, R at the displayed index. The compact metric window comes from K, while formal-ball neighborhoods and radius seals are replayed through F and R over the same metric source M.

Finally attach H, C, P, N. These rows preserve the same finite route under transport and replay, and they contain no open-cover, choice-selected, quotient, host compactness, or ambient-completeness coordinate. □

Theorem 4127.4 (Hemicompact nonescape)**.** *Every neighborhood or compact-window consumer of an accepted* $\mathsf{HemicompactSpace}^{\uparrow}$ *carrier remains inside the finite route through* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. The consumer cannot escape to open-cover compactness, choice-selected finite subcovers, a compact host-space assertion, selected representatives of exhaustion windows, quotient equality, or ambient completeness.*

Proof. Project the accepted packet. The only analytic coordinates are M, K, S, F, R, and each is displayed before the local naming row is available.

Use the exhaustion-window theorem. It forces compact and neighborhood reads to pass through the scheduled compact metric window, formal-ball row, and real-radius seal rather than through an external compactness theorem.

The remaining rows are structural. Transport, replay, provenance, and local naming keep the read inside the displayed BHist packet, so there is no coordinate from which a consumer can obtain the excluded principles. □

Falsifiable prediction: Every $\mathsf{HemicompactSpace}^{\uparrow}$ consumer of a compact exhaustion window reads a MetricSpaceUp source, CompactMetricUp window, StreamNameUp index schedule, FormalBallUp neighborhood row, RealUp radius seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: metricspace, compactmetric, streamname, formalball, real: $\mathsf{HemicompactSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an indexed compact-exhaustion schedule, compact metric windows, formal-ball neighborhood windows, real-radius seals, transport, replay, provenance, and local NameCert rows as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{HemicompactSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp source, CompactMetricUp exhaustion windows, a StreamNameUp index schedule, FormalBallUp neighborhood windows, RealUp radius seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4127.1, Definition 4127.2, Definition 4127.3, and Definition 4127.4 seed the finite compact-exhaustion interface over scheduled compact metric windows and formal-ball neighborhoods.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HemicompactSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, host-space compactness, quotient exhaustion equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window accountability, index-schedule exactness, formal-ball and real-radius replay, transport stability, provenance, and local naming for BEDC.Derived.HemicompactSpaceUp.

Chapter 4128

A Concrete Naming Certificate for $\mathsf{DistanceFunctionCompletion}^{\uparrow}$

Following Bourbaki and Kelley on metric completion by distance functions, the $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ packet records the L10 route where a metric source is read through its distance-to-source observations before later completion and separated-completion consumers inspect it. It sits after the metric carrier of chapter 57, the metric-completion boundary of chapter 1506, the Kuratowski embedding route of chapter 7569, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the distance-function completion route as finite BEDC data; it is not a quotient completion, selected representative, host equality of completed points, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 4128.1 (DistanceFunctionCompletion carrier)**.** *A* $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, C, K, S, R, D, A, H, T, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose distances are observed.*
2. *C is the* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer row reached only after the source metric observations have been displayed.*
3. *K is the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *distance-function handoff row from the source metric into the completion-facing observation space.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected distance observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational values exposed by those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare requested distances and finite approximants.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for each displayed distance value.*
8. *H records componentwise* hsame *transport for the metric source, completion row, Kuratowski handoff, stream windows, regular readback, dyadic ledger, real seal, replay, provenance, and local naming rows.*

9. *T records displayed* Cont *replay of the route*

$$M, K, S, R, D \longmapsto A \longmapsto C \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* DistanceFunctionCompletion$^{\uparrow}$, Met$^{\uparrow}$, MetricCompletion$^{\uparrow}$, KuratowskiEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{DistanceFunctionCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $M, C, K, S, R, D, A, H, T, P, N$. It has no coordinate for quotient completion classes, selected completed points, host equality of distance functions, countable choice over source points, ambient completeness of Real$^{\uparrow}$*, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DistanceFunctionCompletionUp.DistanceFunctionCompletionTasteGate single carrier alignment`

Theorem 4128.2 (DistanceFunctionCompletion NameCert obligations)**.** *For every accepted* DistanceFunctionComp *carrier*

$$F = (M, C, K, S, R, D, A, H, T, P, N),$$

the local NameCert$_{\text{DistanceFunctionCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the metric source, metric-completion consumer, Kuratowski embedding, stream-window, regular-rational, dyadic-tolerance, real-seal, transport, replay, provenance, and local naming rows;*
2. *distance-function exposure, requiring every completion-facing read to pass through M and K before C may be consumed;*
3. *L10 readback, requiring each displayed distance observation to pass through S, R, D, A before it is available to a completion consumer;*
4. *transport stability, requiring* hsame *comparison to preserve the source metric, distance-function handoff, readback route, completion consumer, replay, provenance, and local naming rows componentwise;*
5. *replay locality, requiring* Cont *replay to use only the displayed route from M, K, S, R, D through A and C;*
6. *non-escape, excluding quotient completion classes, selected completed points, host equality of completed points or distance functions, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4128.1. The displayed packet supplies exactly the source metric row, the distance-function handoff, the finite observation route, the completion consumer, and the structural rows.

Distance-function exposure starts with M and K. A completion-facing read has no access to C until the source metric and its Kuratowski distance observations have both been displayed.

The L10 readback is the route S, R, D, A. Stream windows name the inspected observations, regular rational rows read them back, dyadic tolerance rows compare requested scales, and the real seal records the terminal value visible to completion consumers.

Finally H, T, P, N preserve and name the same route. Since no excluded quotient, selector, host-equality, choice, completeness, verification, or bridge object is a carrier row, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{DistanceFunctionCompletion}^{\uparrow}$ packet is generated from a MetricUp source row, a MetricCompletionUp consumer row, a KuratowskiEmbeddingUp distance-function handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4128.1 and Definition 4128.2 seed the finite distance-function completion route through metric source observations, Kuratowski embedding, L10 readback, and completion consumption.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceFunctionCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion classes, selected completed points, host equality of completed points or distance functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, distance-function exposure, L10 readback, transport stability, replay locality, provenance, and local naming for BEDC.Derived.DistanceFunctionCompletionUp.

Chapter 4129

A Concrete Naming Certificate for $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$

The $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet realizes the lift-fiber decomposition of Definition 9462.1687 as finite BEDC NameCert data. It sits beside the codon-window carrier of chapter 2234, the lift-orbit classifier of chapter 4059, the DNA boundary-tile packet of chapter 3393, and the trigger-algebra surface of chapter 3478. The packet records quotient supports, binary lift choices, orbit fibers, and lift-flip ledgers as displayed finite rows; it is not biological mechanism, stochastic fitness, host quotient semantics, or an unrecorded enumeration oracle.

Definition 4129.1 (Minimal trigger lift-bundle carrier)**.** *A* $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, L, O, F, G, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient-support row, listing a quotient-level minimal trigger and its displayed support cells.*
2. *L is the lift-choice row, recording one binary f_3 lift coordinate for each displayed quotient support.*
3. *O is the orbit-fiber row, naming the fiber dimension and orbit type read from the quotient minimal-trigger table.*
4. *F is the finite fiber-count row, displaying the Q_1, Q_2, Q_3, and Q_4 multiplicities and the codon count readback.*
5. *G is the lift-flip ledger, recording the within-fiber one-coordinate flip graph and its edge-count row.*
6. *H records componentwise* hsame *transport for quotient supports, lift choices, orbit fibers, fiber counts, lift-flip ledgers, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a quotient support through lift choices, fiber counting, orbit labeling, and flip-ledger exposure.*
8. *P is the* Pkg *provenance over* $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$*, the source theorem Definition 9462.1687,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, L, O, F, G, H, C, P, N$. It has no coordinate for hidden quotient semantics, empirical biology, probabilistic fitness, an unrestricted search space, or a lift choice detached from the displayed quotient support.

Lean4 ✓ `BEDC.Derived.MinimalTriggerLiftBundleUp.MinimalTriggerLiftBundleTasteGate single carrier alignment`

Theorem 4129.2 (Minimal trigger lift-bundle NameCert obligations)**.** *For every accepted* MinimalTriggerLiftBundle *carrier $B = (Q, L, O, F, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$ *surface consists exactly of quotient-support admission through Q, lift-choice admission through L, orbit-fiber readback through O, finite fiber-count preservation through F, lift-flip ledger exposure through G, componentwise* hsame *transport through H, displayed replay through C, provenance through P, and local naming through N. The carrier admits no read of a codon-level trigger before the quotient support and the matching lift-choice row have both been displayed.*

Proof. Project the carrier of Definition 4129.1. The quotient-support row Q fixes the base trigger, and the lift-choice row L records exactly one binary lift coordinate for each displayed support.

The orbit-fiber row O and count row F are read from the decomposition of Definition 9462.1687. The lift-flip row G records adjacency only inside the same displayed fiber, so no cross-fiber bijection is introduced.

Transport and replay preserve the rows componentwise, while provenance and local naming source the same finite packet. Therefore the listed rows exhaust the local NameCert obligations and prevent detached quotient semantics or hidden enumeration data. □

Theorem 4129.3 (Minimal trigger lift-bundle vision handoff)**.** *The decomposition*

$$\mathcal{T}_c = Q_1 \sqcup 25Q_2 \sqcup 22Q_3 \sqcup Q_4$$

is consumed by $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *only through the finite route*

$$Q, L \longmapsto O, F, G \longmapsto H, C, P, N.$$

Later Lean-facing TasteGate files for this packet must provide Nontrivial $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *and* FieldFaithful $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ *instances before the packet can serve as a verified downstream trigger-lattice consumer.*

Proof. Read the source decomposition through its displayed finite rows. The quotient support gives Q, and the independent binary lift choices give L.

The fiber dimension, multiplicity, and codon count are then recorded by O and F. The lift-flip graph row G is internal to each fiber and does not identify different quotient supports.

The structural rows H, C, P, N transport, replay, source, and name exactly that finite route. The stated TasteGate obligations are therefore downstream instance requirements, not hidden assumptions inside the present paper carrier. □

Falsifiable prediction: For every accepted $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet, replacing each displayed row by a componentwise hsame row preserves the finite fiber counts $1, 25, 22, 1$ over Q_1, Q_2, Q_3, Q_4 and the codon total 294.

Independence witness: codon-six-bit-window, codon-lift-orbit-classifier, dna-window-six-boundary-tile, window-coverage-trigger-algebra: none is bijective with $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ because this

packet jointly carries quotient supports, lift-choice vectors, orbit fibers, finite fiber counts, lift-flip ledgers, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerLiftBundle}^{\uparrow}$ packet is generated from quotient-support rows, binary lift-choice rows, orbit-fiber rows, finite fiber-count rows, lift-flip ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4129.1, Definition 4129.2, and Definition 4129.3 seed the finite lift-fiber NameCert route over quotient supports, lift choices, orbit fibers, fiber counts, and flip ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerLiftBundle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, host quotient semantics, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient-support admission, lift-choice admission, orbit-fiber readback, finite fiber-count preservation, lift-flip ledger exposure, and TasteGate instances for BEDC.Derived.MinimalTriggerLiftBundleUp.

Chapter 4130

A Concrete Naming Certificate for $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ packet records an explicit tolerance-to-source-window modulus as finite BEDC NameCert data. It sits on the finite stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the metric source rows of chapter 57, the continuous-map graph of chapter 130, and the downstream modulus consumer of chapter 369. The packet names the constructive uniform-continuity modulus needed by the open L10 real and completion route; it does not import open-cover compactness, countable choice, quotient real equality, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4130.1 (Bishop uniform-continuity modulus carrier)**.** *A* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, F, T, L, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source-window row naming the finite input observations that a tolerance request may inspect.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, including the displayed pointwise continuity data whose local bounds are eligible for uniform readback.*
3. *T is the tolerance-row carrier admission ledger, recording the requested target precision and the finite source-window budget produced for that request.*
4. *L is the pointwise-to-uniform modulus ledger, displaying the finite route from local continuity rows to one uniform source tolerance.*
5. *D is the dyadic lower-bound readback row, computed inside the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance discipline and exposed before any real-valued consumer reads the modulus.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the source and target estimates.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after S, F, T, L, D, R have been displayed.*
8. *H records componentwise* hsame *transport for source-window, continuous-map, tolerance, ledger, dyadic, regular-readback, real-seal, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a target-tolerance request through the same finite modulus route.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, L, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected representatives, quotient equality of real names, a hidden host modulus, or ambient completion.

Lean4 ✓ `BEDC.Derived.BishopUniformContinuityModulusUp.TasteGate.BishopUniformContinuityModulusTasteGate`
`single carrier alignment`

Theorem 4130.2 (Bishop uniform-continuity modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *carrier $M = (S, F, T, L, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tolerance-row carrier admission through T;*

2. *pointwise-to-uniform modulus ledger exposure through L;*

3. *dyadic lower-bound readback through D;*

4. $\mathsf{Real}^{\uparrow}$ *seal nonescape through E;*

5. *provenance and name exhaustion through P, N together with transport and replay rows H, C.*

Every $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of the packet must read those rows through the displayed carrier route, and no accepted read may project compactness, choice, quotient equality, selected representatives, or a host modulus.*

Proof. Project the carrier of Definition 4130.1. The source-window row S and continuous-map row F are visible before any tolerance output is read.

The tolerance row T and ledger row L record the constructive step from a requested target precision to a uniform source window. The dyadic row D supplies the finite lower-bound comparison, and the regular readback row R makes that comparison available to real-name consumers.

Finally the seal row E is downstream of the displayed route, while H, C, P, N transport, replay, source, and name the same finite packet. Since no other carrier coordinate exists, the listed five obligation rows exhaust the local NameCert surface. □

Theorem 4130.3 (Bishop uniform-continuity modulus compact handoff)**.** *Any compact-uniform-continuity, continuous-map, or* $\mathsf{Real}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ *packet must factor its read through*

$\mathsf{StreamName}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg,

Thus the packet supplies a finite constructive modulus object for later compact-uniform-continuity and completion-facing chapters without importing choice, open-cover compactness, selected representatives, quotient equality, or ambient real completeness.

Proof. Use Definition 4130.2. A downstream read first exposes the finite source window, the continuous-map row, the tolerance row, the pointwise-to-uniform ledger, and the dyadic lower-bound readback.

The regular rational row and real seal then pass the same displayed estimates to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The $\mathsf{UniformModulus}^{\uparrow}$ output is therefore a named row of the finite packet, not a hidden host function.

Transport, replay, provenance, and local naming preserve only the displayed coordinates. The compact handoff can cite the modulus object, but it receives no open-cover compactness, choice principle, quotient equality, selected representative, or ambient completeness theorem from this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformContinuityModulus}^{\uparrow}$ packet is generated from source-window rows, continuous-map rows, tolerance rows, pointwise-to-uniform ledgers, dyadic lower-bound readback, regular rational readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4130.1, Definition 4130.2, and Definition 4130.3 seed the finite Bishop uniform-continuity modulus route for L10 real and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformContinuityModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected representatives, quotient real equality, host modulus functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-row admission, pointwise-to-uniform ledger exposure, dyadic lower-bound readback, Real seal nonescape, and provenance/name exhaustion for BEDC.Derived.BishopUniformContin

Chapter 4131

A Concrete Naming Certificate for $\mathsf{CompactContinuousInverse}^{\uparrow}$

Following Kelley and Bishop–Bridges on compact Hausdorff inverse-continuity arguments, the $\mathsf{CompactContinuousInverse}^{\uparrow}$ packet records the finite BEDC surface by which a continuous bijection from a compact finite-net source into a separated target receives a displayed inverse-continuity handoff. It sits over the compact carrier of chapter 60, the continuous-map graph of chapter 130, the separated target boundary of chapter 1411, and the uniform-homeomorphism consumer of chapter 1815. The packet names only finite-net inverse readback; it does not import open-cover compactness, quotient equality of maps, host inverse functions, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, or a global topological inverse theorem.

Definition 4131.1 (Compact continuous inverse carrier)**.** *A* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, F, B, S, G, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row, exposing the compact packet only through named finite covers and source-window witnesses.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, recording the forward graph and its local continuity data over the displayed compact source.*
3. *B is the bijection ledger, recording graph-totality, graph-single-valuedness, target coverage, and source uniqueness for the same displayed map row.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *target-separation row, or the corresponding Hausdorff separation read when the target is supplied through* $\mathsf{HausdorffSpace}^{\uparrow}$*.*
5. *G is the inverse graph readback row: a target observation reads a unique source window only through B and then returns to the forward graph row F.*
6. *M is the finite-net modulus extraction row, turning the compact source row K, graph row F, bijection ledger B, and separation row S into a named inverse-continuity tolerance window.*
7. *H records componentwise* hsame *transport for the compact, graph, bijection, separation, inverse-readback, and modulus rows.*
8. *C records displayed* Cont *replay from a target-window request through G and M back to the same finite source-net route.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactContinuousInverse}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient host inverse function, quotient equality of maps, open-cover compactness, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactContinuousInverseUp.CompactContinuousInverseTasteGate single carrier alignment`

Theorem 4131.2 (Compact continuous inverse NameCert obligations)**.** *For every accepted* CompactContinuousInve *carrier $I = (K, F, B, S, G, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact source, continuous graph, bijection ledger, separated target, inverse graph readback, finite-net modulus, hsame transport, Cont replay, provenance, and local naming rows. Every inverse-continuity read must inspect K, F, B, S before G and M are available, and every transport or replay read must preserve the same ordered route through H, C, P, N.*

Proof. Project the carrier of Definition 4131.1. The rows K, F, B, S display the compact source, forward graph, bijection data, and target separation before any inverse read is admitted.

The inverse graph row G is therefore a readback of the same graph through the bijection ledger, not a host inverse function. The finite-net modulus row M consumes the compact finite-net row and the separated target row after the graph has been fixed.

Finally H, C, P, N transport, replay, source, and name exactly that route. Since the carrier has no additional inverse, quotient-map equality, selected representative, choice, open-cover, or ambient-completeness coordinate, the stated rows exhaust the local NameCert obligations. □

Theorem 4131.3 (Compact continuous inverse uniform handoff)**.** *Any* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *or inverse-continuity consumer that reads an accepted* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *packet must factor its read through*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{SeparatedMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The exported handoff is the finite-net modulus row M together with the inverse graph readback row G; it is not an open-cover compactness theorem, a quotient equality of functions, a selected host inverse, a countable-choice construction, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Begin with an accepted consumer read. The carrier admits only $K, F, B, S, G, M, H, C, P, N$, so the source compactness and forward graph data must be visible before the inverse graph is read.

The row G turns target observations back into source windows through the bijection ledger. The row M then extracts the finite-net modulus from the compact source and separated target route.

Transport and replay preserve that same route through H and C, while P and N attach provenance and local naming. The consumer may use the resulting modulus as a uniform-homeomorphism handoff, but it receives no extra host inverse, quotient equality, open-cover compactness, choice, or real-completeness principle. □

Theorem 4131.4 (Compact continuous inverse finite-net readback)**.** *Let $I = (K, F, B, S, G, M, H, C, P, N)$ be an accepted* $\mathsf{CompactContinuousInverse}^{\uparrow}$ *carrier. Every target-neighbourhood request for inverse continuity is answered by first reading the inverse graph row G, then the finite-net modulus row M, and then replaying the answer through C over the same K, F, B, S dependency route.*

Proof. The request enters through the separated target row S and the inverse graph readback row G. The graph readback uses the bijection ledger B and the forward graph row F, so it cannot select a host inverse independently of the displayed map.

The finite-net modulus row M then uses the compact source row K to extract a finite source-window response. This is the only modulus coordinate in the packet.

The replay row C returns the answer along the same ordered rows, and H, P, N preserve the route under transport, provenance, and local naming. No open-cover, quotient-equality, selected-representative, choice, or ambient-completeness read is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactContinuousInverse}^{\uparrow}$ packet is generated from compact finite-net source rows, continuous-map graph rows, bijection ledgers, separated-target rows, inverse graph readback, finite-net modulus extraction, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4131.1, Definition 4131.2, Definition 4131.3, and Definition 4131.4 seed the finite inverse-continuity route from compact source data, a continuous bijection ledger, separated target data, inverse graph readback, and finite-net modulus extraction.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactContinuousInverse}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, quotient map equality, host inverse functions, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuous-bijection ledger exactness, separated-target readback, inverse graph readback, finite-net modulus extraction, transport replay, provenance, and local NameCert rows for BEDC.Derived.CompactContinuousInverseUp.

Chapter 4132

A Concrete Naming Certificate for $\mathsf{HausdorffRealCompletion}^{\uparrow}$

Following Bishop–Bridges constructive real analysis and Bourbaki's separated completion discipline, the $\mathsf{HausdorffRealCompletion}^{\uparrow}$ packet records the route by which the separated Hausdorff completion surface is read through the displayed regular real-name faces. It sits after the separated completion carrier of chapter 402, the terminal real-name seal of chapter 38, and the regular rational readback of chapter 301. The packet gives completion-facing consumers a named separated Real-completion certificate without asserting quotient completion equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4132.1 (Hausdorff real completion carrier)**.** *A* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, U, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion row that exposes the separated carrier and its local equality boundary.*
2. *U is the* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *uniqueness row read only as separated limit uniqueness inside the displayed completion route.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule supplying the observable Cauchy-name requests.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling all finite radii and comparison slacks.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from S and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached from the regular readback route.*
7. *H records componentwise* hsame *transport for the separated-completion, uniqueness, stream, dyadic, regular-rational, real-seal, replay, provenance, and local-name rows.*
8. *C records displayed* Cont *replay from a separated Real-completion request through $A, U, S, D, R, E, H, P, N$.*
9. *P records* Pkg *provenance over* $\mathsf{HausdorffRealCompletion}^{\uparrow}$, $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, U, S, D, R, E, H, C, P, N$. It stores no quotient completion equality, selected completion point, host real equality, countable-choice schedule, hidden Cauchy filter, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.HausdorffRealCompletionUp.HausdorffRealCompletionTasteGate single carrier alignment`

Theorem 4132.2 (Hausdorff real completion NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *carrier*

$$Q = (A, U, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission,* $\mathsf{HausdorffCompletion}^{\uparrow}$ *separated-completion exposure through A,* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *separated uniqueness exposure through U, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection through S,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal discipline through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4132.1. The packet displays exactly the separated-completion row, uniqueness row, finite windows, dyadic tolerance ledger, regular rational readback, terminal real seal, and structural rows.

Read the separated-completion route first. The row A supplies only the Hausdorff completion surface, while U records the uniqueness boundary available inside that separated route.

Read the Real-name route next. The finite windows S are measured by the dyadic ledger D, the regular rational row R is read from those displayed requests, and the real seal E is reached only after the regular readback appears.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no further analytic coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4132.3 (Hausdorff real completion separated limit uniqueness)**.** *Every completion-facing consumer of an accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *packet receives separated limit uniqueness only through the finite route*

$$\mathsf{HausdorffCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionUniqueness}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

The route is a separated Real-completion readback certificate, not a quotient completion equality theorem, a selected representative, a hidden Cauchy-filter construction, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Apply Definition 4132.2. It orders every public completion-facing read through A, U, S, D, R, E before any separated uniqueness claim is visible.

The separated side is local to A and U. The Hausdorff completion row supplies the separated carrier boundary, and the metric-completion uniqueness row supplies only the uniqueness read available inside that displayed boundary.

The real-name side is local to S, D, R, E. Stream windows, dyadic tolerances, and regular rational readback must be displayed before the terminal $\mathsf{Real}^{\uparrow}$ seal can be consumed by the same route.

Thus a consumer cannot replace the route by quotient completion equality, selected representatives, hidden filters, host equality, or ambient completeness. The packet records separated limit uniqueness only as the displayed finite NameCert route. □

Theorem 4132.4 (Hausdorff real completion nonescape)**.** *No accepted* $\mathsf{HausdorffRealCompletion}^{\uparrow}$ *packet exports quotient completion equality, a selected completion representative, hidden Cauchy-filter data, countable choice, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*. Every exported separated-completion or Real-facing read is confined to the rows* $A, U, S, D, R, E, H, C, P, N$*.*

Proof. Inspect the carrier definition. Its analytic coordinates are exactly the separated-completion row A, uniqueness row U, stream windows S, dyadic tolerance ledger D, regular rational readback R, and real seal E.

The structural rows add no new analytic principle. Transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet rather than introducing quotient equalities, representative choices, hidden filters, host equality, or completeness.

Use the separated uniqueness theorem, Definition 4132.3. Every separated-completion or Real-facing consumer must pass through the displayed finite route, so none of the excluded data can escape from the packet. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffRealCompletion}^{\uparrow}$ consumer that claims separated limit uniqueness without reading $\mathsf{HausdorffCompletion}^{\uparrow}$, $\mathsf{MetricCompletionUniqueness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row refutes the packet surface.

Independence witness: hausdorffcompletion, metriccompletionuniqueness, streamname, dyadicratcore, regseqrat, real: no listed sibling is bijective to $\mathsf{HausdorffRealCompletion}^{\uparrow}$ because this packet binds separated completion, separated limit uniqueness, finite windows, tolerance accounting, regular readback, Real seal, transport, replay, provenance, and local naming as one L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffRealCompletion}^{\uparrow}$ packet is generated from a HausdorffCompletion separated-completion row, MetricCompletionUniqueness separated uniqueness row, StreamName finite windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4132.1, Definition 4132.2, Definition 4132.3, and Definition 4132.4 seed the separated Real-completion route over HausdorffCompletionUp, MetricCompletionUniquenessUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, hidden Cauchy filters, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove carrier admission, separated-completion exposure, uniqueness exposure, window exactness, tolerance exactness, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and nonescape for BEDC.Derived.HausdorffRealCompletionUp.

Chapter 4133

A Concrete Naming Certificate for $\mathsf{NeighborhoodFilter}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{NeighborhoodFilter}^{\uparrow}$ packet records the local filter generated by displayed neighbourhood rows around a named point or finite source window. It sits between the finite directed basis interface of chapter 1601, the Cauchy-filter surface of chapter 333, the metric source of chapter 57, the open-neighbourhood discipline of chapter 88, and the completion handoff of chapter 1569. The packet names a local neighbourhood-filter interface; it does not quotient filters, select ultrafilters, choose limits, assert host point equality, or import ambient completeness.

Definition 4133.1 (NeighborhoodFilter carrier)**.** *A* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, x, O, B, D, C, M, T, H, K, P, N)$$

with the following displayed rows:

1. *X is the source carrier row whose visible points are consumed through* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *data.*
2. *x is the focused point or source-window row around which the local filter is read.*
3. *O is the finite open-neighbourhood row, exposed through the* $\mathsf{Topology}^{\uparrow}$ *classifier.*
4. *B is the* $\mathsf{FilterBase}^{\uparrow}$ *row generated by the displayed neighbourhood family.*
5. *D is the directed-refinement ledger: two displayed neighbourhood rows admit a smaller displayed neighbourhood row before any filter consumer reads downstream data.*
6. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *compatibility row used when the local neighbourhood filter is consumed by completion routes.*
7. *M is the* $\mathsf{Met}^{\uparrow}$ *witness row when neighbourhoods are read as metric balls or radius windows.*
8. *T records the* $\mathsf{Topology}^{\uparrow}$ *provenance that the displayed open rows are local neighbourhoods of the same focused source.*
9. *H records componentwise* hsame *transport for source, point, open row, basis row, directed refinement, Cauchy compatibility, metric read, topology provenance, replay, package, and naming rows.*
10. *K records displayed* Cont *replay from a local-neighbourhood request through $X, x, O, B, D, C, M, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{NeighborhoodFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$ *row.*
The carrier contains no coordinate for ultrafilters, quotient filters, selected cluster points, host equality of neighbourhood systems, countable-choice selectors, or ambient completion theorems.

Lean4 ✓ `BEDC.Derived.NeighborhoodFilterUp.NeighborhoodFilterTasteGate single carrier alignment`

Theorem 4133.2 (NeighborhoodFilter directed completion base)**.** *For every accepted* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier* $U = (X, x, O, B, D, C, M, T, H, K, P, N)$*, every completion-facing read of the local filter first traverses the directed* $\mathsf{FilterBase}^{\uparrow}$ *row* B *and its refinement ledger* D*. The* $\mathsf{CauchyFilter}^{\uparrow}$ *row* C *and the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *consumer may read only the route*

$$X, x, O \longmapsto B, D \longmapsto C, M, T, H, K, P, N.$$

Thus local-neighbourhood completion data are exported as finite basis rows rather than quotient filters, ultrafilters, selected limits, or hidden host neighbourhood equality.

Proof. Project the carrier of Definition 4133.1. The only basis-facing coordinates are the open-neighbourhood row O, the generated $\mathsf{FilterBase}^{\uparrow}$ row B, and the directed-refinement ledger D.

The completion side cannot bypass those rows. A $\mathsf{CauchyFilter}^{\uparrow}$ or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ consumer reads C only after B and D have supplied the local directed base, while M and T record the metric and topological provenance of the same neighbourhood family.

Finally the structural rows H, K, P, N transport, replay, package, and name that route. Since ultrafilters, quotient filters, selected limits, host equality, and choice selectors are absent from the carrier, every completion-facing read is exhausted by the displayed basis route. □

Theorem 4133.3 (NeighborhoodFilter local Cauchy handoff)**.** *Let* $U = (X, x, O, B, D, C, M, T, H, K, P, N)$ *be an accepted* $\mathsf{NeighborhoodFilter}^{\uparrow}$ *carrier. Whenever a Cauchy-filter completion route consumes the local neighbourhood filter, the handoff factors through the same* $\mathsf{FilterBase}^{\uparrow}$ *row* B*, directed-refinement ledger* D*,* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Topology}^{\uparrow}$ *provenance rows* M, T*, and local* $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$ *row* N*. No consumer receives an independent filter object beyond the finite packet* U*.*

Proof. Begin with Definition 4133.2. It fixes the source-to-basis-to-completion route before any completion-facing read can inspect the Cauchy row C.

Use the metric and topological provenance rows next. The row M records ball-style local reads when available, while T records open-neighbourhood membership through the topology carrier; both are attached to the same focused source X, x, O.

The local NameCert row N closes the handoff by naming the visible packet U. Thus the Cauchy-filter completion route receives finite neighbourhood-filter data, not a detached filter, selected limit, ultrafilter, quotient, host equality witness, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4133.1, Definition 4133.2, and Definition 4133.3 seed the local neighbourhood-filter interface over FilterBaseUp, CauchyFilterUp, MetricUp, TopologyUp, CauchyFilterCompletionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeighborhoodFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, ultrafilters, quotient filters, selected limits, host equality of neighbourhood systems, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, directed-refinement coverage, metric and topology provenance, Cauchy-filter handoff, completion nonescape, and local NameCert faithfulness for BEDC.Derived.NeighborhoodFilterUp.

Constructive story: A $\mathsf{NeighborhoodFilter}^{\uparrow}$ packet is generated from a source row, focused point row, finite open-neighbourhood row, FilterBase row, directed-refinement ledger, CauchyFilter compatibility row, metric and topology provenance rows, transport and replay, package provenance, and a local NameCert row.

Chapter 4134

A Concrete Naming Certificate for $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$

Following Busemann and Bridson–Haefliger, the $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ packet records the additive boundary cocycle read by comparing two displayed disk or geodesic positions toward the same boundary point. It sits after the metric-read surface of chapter 3474, the boundary-point carrier of chapter 3732, the visual comparison route of chapter 3866, the geodesic-flow carrier of chapter 3856, and the horocycle packet of chapter 3534. The packet names a finite boundary-point and metric-read cocycle route; it does not complete the hyperbolic boundary, quotient geodesic rays, choose global representatives, or import host curvature analysis.

Definition 4134.1 (HyperbolicBusemannCocycle carrier)**.** *A* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, \Xi, R_0, R_1, G, V, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row on the visible disk or geodesic source.*
2. *Ξ is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-point row toward which the comparison is read.*
3. *R_0 and R_1 are geodesic-ray or disk-position rows with the same displayed boundary endpoint Ξ.*
4. *G is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *transport row used to compare positions along the same boundary-facing route.*
5. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *visibility row recording the boundary comparison window.*
6. *Q is the* $\mathsf{HyperbolicHorocycle}^{\uparrow}$ *horocycle row that records equal-level surfaces for the boundary point.*
7. *E is the additive cocycle ledger. It stores the signed metric-read difference and the concatenation rule for composable displayed positions.*
8. *H records componentwise* hsame *transport for metric, boundary, ray, flow, visual, horocycle, cocycle, replay, package, and naming rows.*
9. *K records displayed* Cont *replay from a boundary-cocycle request through $M, \Xi, R_0, R_1, G, V, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicHorocycle}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$ *row.*
The carrier contains no coordinate for a completed ideal-boundary quotient, selected geodesic representative, ambient horosphere foliation, global curvature theorem, or physical observer trajectory.

Lean4 ✓ `BEDC.Derived.HyperbolicBusemannCocycleUp.HyperbolicBusemannCocycleTasteGate single carrier alignment`

Theorem 4134.2 (HyperbolicBusemannCocycle boundary additivity)**.** *For every accepted* HyperbolicBusemannCo *carrier* $B = (M, \Xi, R_0, R_1, G, V, Q, E, H, K, P, N)$*, the additive boundary-cocycle read is exhausted by the visible route*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicBusemannBoundary}^{\uparrow} \longmapsto R_0, R_1, G, V, Q \longmapsto E, H, K, P, N.$$

If a third displayed position R_2 is admitted by the same boundary row Ξ, the cocycle ledger E records the composite read from R_0 to R_2 as the concatenation of the reads from R_0 to R_1 and from R_1 to R_2, with all equality understood through the displayed hsame *and* Cont *rows of the same packet.*

Proof. Project the carrier of Definition 4134.1. The metric row M and boundary row Ξ are the only source rows from which a Busemann-level comparison can be read.

Read the geometric comparison rows. The positions R_0, R_1, flow row G, visual row V, and horocycle row Q attach the comparison to one displayed boundary endpoint rather than to an ambient ideal-boundary quotient.

The additive statement is the content of the ledger E. Its concatenation rule is transported by H and replayed by K, while P and N package and name the same finite route. Since selected ray representatives, global horosphere foliations, host curvature theorems, and physical observer trajectories have no coordinate in B, the boundary additivity read is confined to the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4134.1 and Definition 4134.2 seed the finite boundary-point, geodesic-ray, visual-metric, horocycle, additive-ledger, transport, replay, provenance, and NameCert route for the Busemann cocycle.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicBusemannCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, completed ideal-boundary quotients, selected geodesic representatives, global horosphere foliations, host curvature analysis, or physical observer dynamics.

Upgrade path: Obligation closure requires checked carrier admission, boundary endpoint coverage, metric-read exactness, horocycle compatibility, additive-ledger concatenation, and local NameCert faithfulness for BEDC.Derived.HyperbolicBusemannCocycleU

Constructive story: A $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ packet is generated from hyperbolic metric, Busemann boundary, geodesic-position, geodesic-flow, visual-metric, horocycle, additive-ledger, transport, replay, package, and local NameCert rows.

Chapter 4135

A Concrete Naming Certificate for $\mathsf{BishopRealConstruction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealConstruction}^{\uparrow}$ packet records the regular-Cauchy construction of the real numbers as finite BEDC NameCert data. It sits over the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the explicit convergence ledger of chapter 332, and the terminal real-name seal of chapter 38. The packet names the human-known Bishop real construction without quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut replacement, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4135.1 (Bishop real construction carrier)**.** *A* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, S, Q, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, slack comparisons, and endpoint bounds used by every inspected Cauchy window.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the finite observation schedule for the rational approximants before any real-name seal is read.*
3. *Q is the regular-Cauchy source row, retaining the named rational stream together with its displayed regularity budget.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-readback row obtained from D, S, Q, and it is the only row through which rational approximants are exported to later consumers.*
5. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger row, recording the finite modulus entries that witness the regular-Cauchy estimates read by R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the dyadic, stream, regular-readback, and convergence-ledger rows have all been displayed.*
7. *H records componentwise* hsame *transport for dyadic tolerances, finite windows, regular sources, readback rows, convergence ledgers, real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a real-construction request through the same finite route D, S, Q, R, M, E.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopRealConstruction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, Q, R, M, E, H, C, P, N$. It has no coordinate for quotient equality of streams, a selected host limit, countable choice, host real equality, Dedekind-cut replacement, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopRealConstructionUp.BishopRealConstructionTasteGate single carrier alignment`

Theorem 4135.2 (Bishop real construction NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$ *surface consists exactly of the following six obligation rows:*

1. *carrier admission for the dyadic tolerance ledger D, finite stream windows S, regular-Cauchy source Q, regular readback R, convergence ledger M, real seal E, transport H, replay C, provenance P, and local naming N;*
2. *tolerance-window exactness, requiring every accepted read to pair the same dyadic budget in D with the same finite windows in S;*
3. *regular-Cauchy readback, requiring the rational approximants exported by the packet to pass through Q, R before any* $\mathsf{Real}^{\uparrow}$ *consumer reads them;*
4. *convergence-ledger exposure through M, so the construction records finite regularity evidence rather than a hidden host limit;*
5. *seal nonescape through E, which blocks quotient stream equality, selected host limits, host real equality, Dedekind-cut replacement, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness;*
6. *provenance and name exhaustion through P, N together with componentwise transport and replay rows H, C.*

No other NameCert obligation is available from the Bishop real construction packet.

Proof. Project the carrier of Definition 4135.1. The displayed tuple exposes the dyadic tolerance ledger, finite stream windows, regular-Cauchy source, regular rational readback, convergence ledger, real seal, transport, replay, provenance, and local naming rows.

The tolerance-window exactness row is read before any rational approximation is exported. A request first fixes the dyadic budget D and the finite windows S, then reads the regular source Q and the readback row R along that same displayed route.

The convergence ledger M records the finite regularity evidence used by the construction. The real seal E is downstream of D, S, Q, R, M, so a consumer cannot replace those rows by quotient stream equality, a selected host limit, host real equality, Dedekind cuts, countable choice, or ambient completeness.

Finally H, C, P, N transport, replay, source, and name exactly the same finite packet. Since the carrier has no additional coordinate, the six listed rows exhaust the local NameCert surface. □

Theorem 4135.3 (Regular-Cauchy readback route)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

every rational approximant exported by B is read along the displayed route

$$D, S \longrightarrow Q \longrightarrow R \longrightarrow M.$$

In particular, the approximant first uses the dyadic tolerance ledger D and finite stream windows S, then passes through the regular-Cauchy source Q, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the convergence ledger M before it is available to any later row of the packet.*

Proof. Project the carrier of Definition 4135.1. Its only rows capable of naming rational approximants are the finite windows S, the regular-Cauchy source Q, and the regular readback row R, all measured against the dyadic budget D.

Apply Definition 4135.2. The regular-Cauchy readback obligation requires exported rational approximants to pass through Q, R, while the convergence-ledger exposure obligation records the finite regularity evidence in M. Since the classifier compares packets only through $D, S, Q, R, M, E, H, C, P, N$, no alternate approximant coordinate is available. □

Theorem 4135.4 (Real seal handoff)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

every $\mathsf{Real}^{\uparrow}$ *consumer reaches the seal row E only after the dyadic ledger D, finite stream windows S, regular-Cauchy source Q, regular readback R, and convergence ledger M have been displayed, and only through the transport, replay, provenance, and local naming rows H, C, P, N.*

Proof. Project the displayed tuple in Definition 4135.1. The seal row E is declared downstream of D, S, Q, R, M, and the remaining public movement of the packet is carried by componentwise hsame transport H, Cont replay C, provenance P, and local naming N.

Apply the seal nonescape and provenance-name exhaustion clauses of Definition 4135.2. They force the $\mathsf{Real}^{\uparrow}$ seal to remain attached to the same finite route, so a consumer can reach E only through the displayed handoff rows rather than through a hidden host-side real object. □

Theorem 4135.5 (Ledger nonescape)**.** *For every accepted* $\mathsf{BishopRealConstruction}^{\uparrow}$ *carrier*

$$B = (D, S, Q, R, M, E, H, C, P, N),$$

the packet contains no coordinate for quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut substitution, or ambient $\mathsf{Real}^{\uparrow}$ *completeness. Any accepted comparison or consumer read must therefore be witnessed inside the rows $D, S, Q, R, M, E, H, C, P, N$.*

Proof. Project the carrier definition. The classifier is explicitly restricted to the displayed rows $D, S, Q, R, M, E, H, C, P, N$, and the definition states that the excluded principles have no coordinate in the packet.

Apply the seal nonescape clause of Definition 4135.2. It blocks the listed host-side or quotient-side replacements at the NameCert surface. The exhaustion clause then shows that H, C, P, N only transport, replay, source, and name the same finite packet, so they cannot introduce a new equality, limit, choice, cut, or completeness row. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealConstruction}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite-window rows, RegSeqRat regular-readback rows, RealModulusOfConvergence ledger rows, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealConstruction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4135.1 and Definition 4135.2 seed the finite Bishop–Bridges regular-Cauchy construction route for Real and completion-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopRealConstruction}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, host real equality, Dedekind-cut replacement, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-Cauchy readback, Real seal handoff, convergence-ledger refusal, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealConstructionUp.BishopRealConstruct

Chapter 4136

A Concrete Naming Certificate for $\mathsf{CauchyIntegralFormula}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive complex analysis, the $\mathsf{CauchyIntegralFormula}^{\uparrow}$ packet names the finite contour-evaluation theorem surface that lets a holomorphic local datum be read at an interior point through an explicit contour integral. It sits above the holomorphic predicate of chapter 68, the complex carrier of chapter 39, the complex series modulus rows of chapter 69, the operation socket of chapter 583, and the triangle-to-polygon null-integral route of chapter 2098. The chapter names the Cauchy integral formula as a BEDC NameCert dependency for residue, maximum-modulus, and analytic-continuation consumers; it does not introduce a host contour integral, quotient path space, classical compactness principle, or unrecorded branch-selection oracle.

Definition 4136.1 (Cauchy integral formula carrier)**.** *A* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, Z, R, H, C, S, G, L, T, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Holomorphic}^{\uparrow}$ *domain row, displaying the holomorphic function, its complex neighbourhood, and the interior point whose value is to be read.*
2. *Z is the* $\mathsf{Complex}^{\uparrow}$ *point row, carrying the point coordinate and the complex arithmetic rows used by the evaluation denominator.*
3. *R is the radius and contour row, naming a rational circle or polygonal contour around Z whose image remains inside the displayed holomorphic neighbourhood.*
4. *H is the* $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row, exposing the piecewise-linear contour, holomorphic input, Riemann-sum modulus, complex output, operation-law ledger, transport, replay, provenance, and local naming rows.*
5. *C is the* $\mathsf{CplxSer}^{\uparrow}$ *local expansion row, recording the finite read of the convergent power-series data needed to identify the residue coefficient at the point.*
6. *S is the singular-kernel exclusion row, displaying that the contour avoids the evaluation point in the denominator while the point remains inside the contour.*
7. *G is the* $\mathsf{Goursat}^{\uparrow}$ *null-triangle ledger row, routing contour deformation and polygon subdivision through the already named null-integral packet.*

8. *L is the formula ledger row, recording the equality between the point value and the contour integral of the Cauchy kernel over the displayed contour.*
9. *T records componentwise* hsame *transport for the holomorphic, complex, contour, socket, series, exclusion, Goursat, formula, replay, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyIntegralFormula}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{CplxSer}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$ $\mathsf{Goursat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, Z, R, H, C, S, G, L, T, P, N$. It has no coordinate for a host analytic theorem, selected branch, quotient path equality, or classical integral representation outside the displayed contour socket.

Lean4 ✓ `BEDC.Derived.CauchyIntegralFormulaUp.CauchyIntegralFormulaTasteGate single carrier alignment`

Theorem 4136.2 (Cauchy integral formula NameCert obligations)**.** *For every accepted* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier $F = (D, Z, R, H, C, S, G, L, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$ *surface consists exactly of holomorphic-domain admission through D, complex point and arithmetic admission through Z, contour containment through R, contour-socket admission through H, complex-series local expansion through C, singular-kernel exclusion through S, Goursat deformation through G, formula-ledger exactness through L, componentwise transport through T, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.CauchyIntegralFormulaUp.CauchyIntegralFormulaCarrier namecert obligations`

Proof. Project the carrier. The holomorphic source D and point row Z fix the only domain and evaluation data that the packet may read.

The contour row R and socket row H provide the finite operation surface. They determine the contour-integral value through displayed piecewise-linear, Riemann-sum, complex-output, and operation-law rows.

The series row C, exclusion row S, and Goursat row G identify the Cauchy kernel computation and the allowed contour deformations. The formula ledger L then records the point-evaluation equality inside the same packet.

Finally T, P, N transport, source, and name the same finite route. Since no carrier coordinate stores a host contour theorem, branch selection, quotient path equality, or classical integral representation, the listed rows exhaust the NameCert obligations. □

Theorem 4136.3 (Cauchy integral formula contour-socket route)**.** *Every point-evaluation read exported by an accepted* $\mathsf{CauchyIntegralFormula}^{\uparrow}$ *carrier factors through the finite route*

$$D, Z, R \longmapsto H \longmapsto C, S, G, L \longmapsto T, P, N.$$

In particular, the read must pass through the $\mathsf{ContourIntegralSocket}^{\uparrow}$ *row before the formula ledger can be consumed, and it cannot bypass the contour socket by citing a host Cauchy theorem, quotient path identity, selected branch, or undisplayed completeness object.*

Proof. Start with a point-evaluation request. The rows D, Z, R display the holomorphic input, complex point, and contour containment data, but none of them alone stores the contour integral or the formula equality.

The request must next pass through H. By chapter 583, the socket exposes the contour, input, modulus, complex output, operation ledger, transport, replay, provenance, and local naming rows needed by a contour-integral consumer.

The rows C, S, G, L then attach local series computation, denominator exclusion, Goursat deformation, and formula equality to that same socket route. Transport and provenance through T, P, N preserve the displayed route without adding a second source of analytic equality.

Therefore every exported Cauchy integral formula read is socket-mediated and finite-row visible. The absent host theorem, quotient path identity, branch selection, and completeness oracle have no coordinate in the carrier. □

Falsifiable prediction: Every residue, maximum-modulus, or analytic-continuation consumer that cites $\mathsf{CauchyIntegralFormula}^{\uparrow}$ must expose holomorphic-domain data, complex point data, contour containment, contour-socket rows, local complex-series data, singular-kernel exclusion, Goursat deformation, formula ledger, componentwise hsame transport, Pkg provenance, and local NameCert evidence.

Independence witness: holomorphic, complex-series, contourintegralsocket, goursat: none is bijective with $\mathsf{CauchyIntegralFormula}^{\uparrow}$ because this packet jointly carries point-evaluation data, contour containment, socket-mediated integral rows, kernel exclusion, deformation ledger, formula equality, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyIntegralFormula}^{\uparrow}$ packet is generated from holomorphic-domain rows, complex point rows, contour-containment rows, contour-socket rows, local complex-series rows, singular-kernel exclusion rows, Goursat deformation rows, formula-ledger rows, componentwise hsame transport, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4136.1, Definition 4136.2, and Definition 4136.3 seed the finite contour-mediated point-evaluation packet over holomorphic, complex, complex-series, contour-socket, Goursat, transport, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyIntegralFormula}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residue calculus, maximum-modulus consequences, host contour integration, quotient path spaces, global analytic continuation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, holomorphic source lock, contour containment, contour-socket handoff, local series exactness, singular-kernel exclusion, Goursat deformation, formula-ledger exactness, and non-escape for BEDC.Derived.CauchyIntegralFormulaUp.

Chapter 4137

A Concrete Naming Certificate for $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$

The $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet realizes the zero-marker support-nerve sphere core of Definition 9462.1665 and the finite count table of Definition 9462.1668 as finite BEDC NameCert data. It sits beside the support-nerve realization of chapter 4079, the support-nerve census of chapter 4048, the codon lift packet of chapter 4129, and the boundary-tile packet of chapter 3393. The packet names the quotient trigger, complement simplex cover, boundary-simplex nerve, reduced homology degree, finite count table, TasteGate admission boundary, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic selection claim, or unbounded enumeration oracle.

Definition 4137.1 (Minimal trigger homology-core carrier)**.** *A* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, F, N, R, B, K, T, H, C, P, L)$$

with the following displayed rows:

1. *Q is the quotient-level minimal full-trigger row, displaying $A = \{S_1, \ldots, S_m\}$ and its finite support cells.*
2. *F is the complement-simplex row, recording $F_i = \Omega \setminus S_i$ and the finite simplices 2^{F_i}.*
3. *N is the support-nerve row, identifying the union $\bigcup_i 2^{F_i}$ as the displayed support nerve.*
4. *R is the boundary-simplex route row, recording that the cover nerve is $\partial\Delta^{m-1}$ because every proper complement intersection is inhabited and the full intersection is empty.*
5. *B is the reduced-homology row, naming the sphere type S^{m-2} and the corresponding one-dimensional reduced homology witness.*
6. *K is the finite count row, displaying the quotient and codon count table for trigger sizes $1, 2, 3, 4$.*
7. *T is the TasteGate obligation row, requiring later* $\mathsf{Nontrivial\ MinimalTriggerHomologyCore}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ MinimalTriggerHomologyCore}^{\uparrow}$ *instances to preserve Q, F, N, R, B, K before downstream consumers may use the packet as a stable AI-origin chapter surface.*
8. *H records componentwise* hsame *transport for quotient triggers, complement simplices, support nerves, boundary-simplex routes, homology rows, count rows, TasteGate rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a quotient trigger through complement simplices, support nerves, boundary-simplex routing, homology readback, and finite count exposure.*

10. *P is the* Pkg *provenance over* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$*, Definition 9462.1665, Definition 9462.1668,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, F, N, R, B, K, T, H, C, P, L$. It has no coordinate for empirical biology, stochastic fitness, a host simplicial-complex library, or an unrestricted search over supports.

Theorem 4137.2 (Minimal trigger homology-core NameCert obligations)**.** *For every accepted* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier $M = (Q, F, N, R, B, K, T, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyC}}$ *surface consists exactly of quotient-trigger admission through Q, complement-simplex admission through F, support-nerve construction through N, boundary-simplex routing through R, reduced-homology readback through B, finite count preservation through K, TasteGate obligation exposure through T, componentwise transport through H, replay through C, provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.MinimalTriggerHomologyCoreUp.MinimalTriggerHomologyCoreNamecertObligations`

Proof. Project the carrier. The quotient trigger row Q fixes the finite support family, and the complement row F records the finite simplices that generate the support nerve.

The support-nerve row N and boundary-simplex route R encode the nerve-theorem calculation from Definition 9462.1665. The reduced-homology row B reads the displayed sphere degree, and the count row K records the quotient and codon tables from Definition 9462.1668.

The TasteGate row T is an explicit downstream obligation: later Nontrivial and FieldFaithful instances must preserve the carrier rows, but those instances are not hidden assumptions in the present packet. The rows H, C, P, L transport, replay, source, and name the same finite data. □

Theorem 4137.3 (Minimal trigger homology-core support-nerve route)**.** *Every homology-core read exported by an accepted* $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ *carrier factors through the finite route*

$$Q, F \longmapsto N, R \longmapsto B, K \longmapsto T, H, C, P, L.$$

The route requires the quotient minimal-trigger row and complement-simplex row before the support nerve, boundary-simplex route, homology witness, count table, TasteGate obligation, transport, replay, provenance, and local naming rows can be consumed.

Proof. Start with a homology-core read. The row Q displays the quotient-level minimal full trigger, while F displays its complement simplices; neither row by itself stores a homology conclusion.

The read must pass through N and R. These rows expose the support nerve and the boundary-simplex route that turns minimal full-trigger data into the sphere core.

The rows B and K then read the reduced homology degree and the finite count table. The TasteGate row T keeps the future nontriviality and field-faithfulness requirements attached to the same carrier rather than replacing the packet by a detached classifier.

Finally H, C, P, L preserve the route by componentwise transport, replay, provenance, and local naming. Since the carrier has no row for a host simplicial-complex library, stochastic fitness model, or unbounded support search, every exported read is exhausted by the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet exposes quotient-trigger data, complement-simplex data, support-nerve data, boundary-simplex routing, reduced homology readback, finite count rows, TasteGate obligations, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence before any downstream trigger-lattice consumer may cite the sphere-core result.

Independence witness: support-nerve-realization, support-nerve-census, minimal-trigger-lift-bundle, dna-window-six-boundary-tile: none is bijective with $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ because this packet jointly carries quotient minimal-trigger rows, complement simplices, support-nerve routing, reduced homology, finite counts, TasteGate obligations, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerHomologyCore}^{\uparrow}$ packet is generated from quotient minimal-trigger rows, complement-simplex rows, support-nerve rows, boundary-simplex route rows, reduced-homology rows, finite count rows, TasteGate obligation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4137.1, Definition 4137.2, and Definition 4137.3 seed the finite support-nerve sphere-core packet over quotient triggers, complement simplices, boundary-simplex routes, reduced homology, count rows, TasteGate obligations, transport, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerHomologyCore}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, host simplicial-complex libraries, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient-trigger admission, complement-simplex admission, support-nerve construction, boundary-simplex route exactness, homology readback, finite count preservation, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.MinimalTriggerHomologyCoreUp.

Chapter 4138

A Concrete Naming Certificate for $\mathsf{CompactOpenTopology}^{\uparrow}$

Following Kelley and Bourbaki on compact-open topologies, the $\mathsf{CompactOpenTopology}^{\uparrow}$ packet names the finite subbasic topology surface carried by compact-domain windows and open-target tests. It sits next to the compact-open metric packet of chapter 1903, the compact-open function-space carrier of chapter 2221, the continuous-map graph route of chapter 130, the compact-metric source surface of chapter 129, and the compact-window convergence route of chapter 4022. The packet records compact-domain rows, open-target rows, finite window rows, componentwise transport, displayed replay, provenance, and local naming; it does not install a host topology on all functions.

Definition 4138.1 (Compact-open topology carrier)**.** *A* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, O, W, F, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact-domain row. It records the retained* $\mathsf{CompactMetric}^{\uparrow}$ *source windows and the compact subobjects whose images are tested.*
2. *O is the open-target row. It records the displayed target-open tests used by the subbasic condition.*
3. *W is the finite window row. Each entry pairs one compact-domain entry from K with one open-target entry from O.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose displayed values are inspected only on the windows listed by W.*
5. *S is the subbasic membership ledger. It records that the displayed map sends the retained compact-domain window into the corresponding open-target row.*
6. *H records componentwise* hsame *transport for compact-domain rows, open-target rows, finite windows, continuous-map graph rows, subbasic membership ledgers, replay rows, provenance, and local naming.*
7. *C records displayed* Cont *replay from a compact-open topology request through compact-domain inspection, open-target inspection, graph evaluation, and subbasic membership.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactOpenMetric}^{\uparrow}$, $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyUniformConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, W, F, S, H, C, P, N$. There is no coordinate for open-cover compactness, choice-selected compact witnesses, quotient equality of host functions, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactOpenTopologyUp.CompactOpenTopologyTasteGate single carrier alignment`

Theorem 4138.2 (Compact-open topology NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier $T = (K, O, W, F, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$ *surface consists exactly of these obligation rows:*

1. *carrier admission for compact-domain rows, open-target rows, finite windows, continuous-map graph rows, subbasic membership ledgers, transport, replay, provenance, and local naming;*
2. *compact-domain visibility, requiring every domain side of a subbasic request to read through the retained* $\mathsf{CompactMetric}^{\uparrow}$ *rows in K;*
3. *open-target visibility, requiring every target side of a subbasic request to read through the displayed open-target row O;*
4. *finite-window stability, requiring membership checks to use the paired rows in W and the subbasic ledger S before any consumer reads the topology packet;*
5. *non-escape, requiring H, C, P, N to preserve the same displayed rows and to export no open-cover compactness, choice-selected compact witness, host function-space equality, or ambient completeness.*

No compact-open topology obligation is stored outside those rows.

Proof. Project the carrier in Definition 4138.1. The visible topology coordinates are exactly the compact-domain row, open-target row, finite window row, continuous-map graph row, subbasic membership ledger, and the structural rows H, C, P, N.

The compact-domain side is forced by K. Its reads are the same finite compact-source surface used by chapter 129 and by the compact-open metric route of chapter 1903.

The function side is forced by F and S. A subbasic membership request must inspect displayed graph values through chapter 130 and then record the result in the finite membership ledger attached to the paired window in W.

The provenance row places the packet beside chapter 2221 and chapter 4022. Those siblings may consume the displayed subbasic surface, but the carrier has no row for host function-space equality, open-cover compactness, choice-selected compact witnesses, or ambient completeness. □

Theorem 4138.3 (Compact-open topology finite subbase stability)**.** *Let $T = (K, O, W, F, S, H, C, P, N)$ be an accepted* $\mathsf{CompactOpenTopology}^{\uparrow}$ *carrier. Every finite family of displayed compact-open subbasic requests admitted through W remains a request on the same compact-domain, open-target, continuous-map, and membership rows after componentwise* hsame *transport by H and displayed* Cont *replay by C. Thus finite subbase stability is internal to $K, O, W, F, S, H, C, P, N$ and cannot be replaced by open-cover compactness, choice-selected compact witnesses, host function-space equality, or ambient completeness.*

Proof. Start with a finite list of subbasic requests. Each request is already an entry of W, so it names one compact-domain row from K and one open-target row from O before it can inspect a function.

Read the continuous-map data through F and the membership row through S. The compact-open metric and compact-open function-space siblings may reuse this windowed read, but they receive only the displayed graph and membership coordinates.

Transport by H is componentwise on the listed rows. Replay by C repeats the same compact-domain, open-target, graph, and membership route, so finite intersections of displayed subbasic requests remain inside the same carrier.

Since the carrier has no row for open-cover compactness, selected compact witnesses, host function equality, or ambient completeness, none of those principles can be introduced by finite subbase stability. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenTopology}^{\uparrow}$ packet is generated from compact-domain rows, open-target rows, finite compact-open windows, ContinuousMap graph rows, subbasic membership ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4138.1, Definition 4138.2, and Definition 4138.3 seed the finite subbasic compact-open topology surface for compact-domain and open-target requests.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenTopology}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactOpenTopologyUp.CompactOpenTopologyTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected compact witnesses, host function-space equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires binding compact-domain visibility, open-target visibility, finite-window stability, graph membership, componentwise transport, replay, provenance, and local naming explicitly to the finite carrier rows.

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ hub reads pointwise stream windows through chapter 126, the metric comparison route through chapter 57, and terminal real seals through chapter 38.

Chapter 4139

A Concrete Naming Certificate for $\mathsf{PointwiseStreamMetric}^{\uparrow}$

Following Bishop and Bridges' constructive metric-space treatment, the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet records the standard pointwise distance interface for certified streams as finite BEDC data. It sits after the stream-name surface of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric-space carrier discipline of chapter 57. The packet exposes pointwise metric data for completion and continuous-map consumers without host functions, quotient stream equality, selected representatives, or choice.

Definition 4139.1 (Pointwise stream metric carrier)**.** *A* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the certified stream source row, read through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window interface.*
2. *W records the finite pointwise window schedule used when two certified streams are compared at a displayed index.*
3. *D is the dyadic tolerance and triangle ledger governing the finite comparisons of pointwise observations.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the finite rational comparison stream into a regular rational name.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the distance value obtained from the displayed pointwise route.*
6. *M is the* $\mathsf{Met}^{\uparrow}$ *metric-structure row, used only after the pointwise stream route has displayed S, W, D, R, E.*
7. *H stores componentwise* hsame *transport for stream names, window schedules, dyadic ledgers, regular readbacks, real seals, metric rows, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a consumer request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these coordinates. It contains no host function equality, quotient stream object, selected stream representative, countable-choice schedule, or ambient completeness assertion.

Theorem 4139.2 (Pointwise stream metric pointwise classifier)**.** *For an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier* $Q = (S, W, D, R, E, M, H, C, P, N)$*, every pointwise distance classifier read factors through the certified stream source* S*, the finite window schedule* W*, the dyadic tolerance ledger* D*, the regular-rational readback* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *before the* $\mathsf{Met}^{\uparrow}$ *row* M *may be inspected. The classifier therefore names a pointwise metric route on certified streams and does not read host functions, quotient streams, selected representatives, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.PointwiseStreamMetricUp.PointwiseStreamMetricPointwiseClassifier`

Proof. Project the carrier of Definition 4139.1. The source stream, window schedule, dyadic ledger, regular readback, Real seal, metric row, transport, replay, provenance, and local naming row are all displayed coordinates.

The pointwise route begins with S and W. A consumer must name the stream source and the finite observation windows before any rational readback or metric row can be inspected.

The distance read then passes through D, R, E. The dyadic ledger bounds the finite comparisons, the regular-rational row gives the readback discipline, and the Real seal is the terminal distance value. Since no coordinate stores a host function or quotient stream, the classifier cannot export one. □

Theorem 4139.3 (Pointwise stream metric constant stream)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier whose stream source row* S *is displayed as a constant certified stream. Then every distance read between two displayed copies of that source is routed through the same finite window schedule* W*, dyadic zero-tolerance ledger* D*, regular-rational readback* R*, terminal Real seal* E*, and metric row* M*. The packet records the constant-stream zero-distance route as displayed NameCert data, not as host equality of functions or quotient stream identity.*

Proof. Start with the displayed constant stream row in S. Because the stream is certified by its $\mathsf{StreamName}^{\uparrow}$ windows, the two copies are compared only through the finite schedule W.

Use the dyadic ledger D for the zero-tolerance comparison. The readback row R and the terminal seal E record the regular-rational and Real distance route generated by that finite ledger.

Finally inspect M only after the pointwise route has been displayed. The metric row reads the zero-distance route as finite certificate data, while host function equality and quotient stream identity are absent from the carrier. □

Theorem 4139.4 (Pointwise stream metric reindexing stability)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any accepted finite reindexing of the certified stream windows transports the pointwise metric classifier only through the displayed* hsame *rows* H *and the replay row* C*: the reindexed route must preserve the stream source row, window schedule, dyadic ledger, regular-rational readback, Real seal, metric row, provenance, and local naming row. It cannot add a hidden stream quotient, selected representative, or host-level function extensionality principle.*

Proof. Read the transport coordinate H. It is componentwise: stream sources, window schedules, dyadic ledgers, readback rows, Real seals, metric rows, replay, provenance, and naming rows are transported as displayed rows.

Replay the reindexed request with C. Since C follows the same order $S, W, D, R, E, M, H, P, N$, the pointwise classifier is preserved only by the finite route already present in the carrier.

The exclusions are an inventory check. No coordinate of Q stores a stream quotient, selected representative, or host-level function extensionality principle, so reindexing stability cannot export such data. □

Theorem 4139.5 (Pointwise stream metric NameCert obligations)**.** *For an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier* $Q = (S, W, D, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the certified stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, Real seal, metric row, componentwise transport, continuation replay, provenance, and local naming row; pointwise classifier exposure; constant-stream zero-distance exposure; reindexing stability; and ledger non-escape for host functions, quotient streams, selected representatives, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 4139.1. The accepted packet displays exactly the stream, window, tolerance, readback, Real, metric, transport, replay, provenance, and naming rows.

Pointwise classifier exposure is Definition 4139.2. Constant-stream zero-distance exposure is Definition 4139.3. Reindexing stability is Definition 4139.4.

The ledger non-escape clause follows from the same carrier inventory. Since no row stores host functions, quotient streams, selected representatives, countable-choice schedules, or ambient completeness, the local NameCert surface is exhausted by the displayed finite rows. □

Theorem 4139.6 (Pointwise stream metric sibling route)**.** *Every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer reaches pointwise distance data only after the certified* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{Met}^{\uparrow}$ *carrier rows named in the introduction have been displayed. The pointwise metric row is therefore a sibling route through the finite stream, regular-rational, Real-seal, and metric surfaces, not a host function metric, quotient stream comparison, selected representative, countable-choice schedule, or ambient completeness assertion.*

Proof. Apply Definition 4139.5. The obligation surface lists carrier admission for the certified stream source, finite pointwise windows, dyadic tolerance ledger, regular-rational readback, Real seal, metric row, componentwise transport, continuation replay, provenance, and local naming row.

The opening paragraph identifies the sibling chapters that supply those rows: chapter 126, chapter 301, chapter 38, and chapter 57. Reading the pointwise distance surface therefore requires the stream window before the regular readback, the regular readback before the Real seal, and the Real seal before the metric row may be used.

Since the same obligation theorem excludes host functions, quotient streams, selected representatives, countable choice, and ambient completeness, the sibling route is exhausted by the displayed finite packet. □

Theorem 4139.7 (Pointwise stream metric Real-completion anchor)**.** *Every completion-facing read of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the displayed* $\mathsf{Met}^{\uparrow}$ *row before it reaches any Real-completion consumer. The route exports the finite pointwise metric packet and excludes host functions, quotient streams, selected representatives, countable choice, and ambient completeness.*

Proof. Start with the pointwise classifier of Definition 4139.2. It forces a distance read to pass through the certified stream source, finite window schedule, dyadic tolerance ledger, regular-rational readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before the metric row is inspected.

Use Definition 4139.4 to transport any completion-facing reindexing request. The transport coordinate H and replay coordinate C preserve the same stream windows, dyadic ledger, readback row, Real seal, metric row, provenance, and local naming row.

Apply Definition 4139.5 and then Definition 4139.6. The NameCert obligations exhaust the accepted packet, and the sibling route identifies the supplying chapters for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$; the dyadic ledger is already a displayed coordinate of the carrier and obligations.

Therefore a Real-completion consumer receives only the finite route

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow}.$$

Since none of the cited surfaces stores host functions, quotient streams, selected representatives, countable-choice schedules, or ambient completeness, the completion-facing read cannot acquire them. □

Theorem 4139.8 (Pointwise stream metric dossier lattice route)**.** *Every* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *or* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must read the same finite pointwise window W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and displayed* $\mathsf{Met}^{\uparrow}$ *row M before any completion-facing route is consumed. The downstream lattice is therefore the convergence-ledger route of chapter 2074 together with the completion route of chapter 1506, not an ambient completeness theorem, host function equality, quotient stream equality, selected representative, or countable-choice schedule.*

Proof. Project the carrier of Definition 4139.1. The only rows available before the structural data are the certified stream source S, finite window schedule W, dyadic ledger D, regular-rational readback R, terminal Real seal E, and metric row M.

Apply the Real-completion anchor Definition 4139.7. It already forces every completion-facing read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the displayed $\mathsf{Met}^{\uparrow}$ row before a consumer receives the pointwise metric packet.

Now split the downstream request by destination. A $\mathsf{RealModulusOfConvergence}^{\uparrow}$ consumer reads the convergence ledger only through chapter 2074, so the same W, D, R, E, M rows remain visible before convergence data is public. A $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads the completion route only through chapter 1506, so the same finite pointwise dossier is still the source of completion-facing distance data.

Because the carrier lacks coordinates for ambient completeness, host function equality, quotient stream equality, selected representatives, and countable-choice schedules, neither downstream route can acquire those objects from $\mathsf{PointwiseStreamMetric}^{\uparrow}$. □

The finite-window side of this lattice is consumed by chapter 5916 through its pointwise metric forward route. Thus a downstream $\mathsf{FiniteStream}^{\uparrow}$ reader receives the same $S, W, D, R, E, M, H, C, P, N$ packet as a finite-window metric read, not as an added completion principle or a hidden stream quotient.

The downstream completion lattice consumes this anchor through the convergence-ledger route of chapter 2074. Every $\mathsf{RealModulusOfConvergence}^{\uparrow}$ consumer of an accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet must read the same finite windows W, dyadic ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, terminal $\mathsf{Real}^{\uparrow}$ seal E, and displayed metric row M before any convergence ledger is public. The consumer therefore receives the pointwise stream metric only as the finite route already exposed by the carrier, not as a detached completion principle.

Theorem 4139.9 (Pointwise stream metric finite-window consumption)**.** *Every metric-distance read from an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet first consumes the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{Met}^{\uparrow}$ *distance row. The consumed route is the finite packet*

$$S, W, D, R, E, M, H, C, P, N$$

of Definition 4139.1, with the dyadic ledger D bounding the window comparison before the Real seal and metric row are read.

The route is invariant under accepted scheduled window replacement transported by H and replayed by C. It exposes no quotient stream equality, selected stream representative, host function equality, countable-choice schedule, or ambient completeness principle.

Proof. Project the carrier of Definition 4139.1. The metric-distance read has only the stream source, finite window schedule, dyadic ledger, regular readback, Real seal, metric row, transport, replay, provenance, and local naming coordinates available.

Apply Definition 4139.6. It orders the reader through the $\mathsf{StreamName}^{\uparrow}$ finite window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the terminal $\mathsf{Real}^{\uparrow}$ seal, and then the $\mathsf{Met}^{\uparrow}$ row.

Use Definition 4139.7 to connect the same route to completion-facing consumers. The dyadic ledger remains a displayed coordinate of the packet, so the read cannot bypass the finite window and regular readback.

Finally invoke Definition 4139.4. Any scheduled replacement transported by H and replayed by C preserves the same stream windows, dyadic ledger, readback, Real seal, metric row, provenance, and local naming row. Since the carrier has no quotient stream, selected representative, host function equality, choice schedule, or ambient completeness coordinate, the finite-window consumption route is exhausted by the displayed packet. □

Theorem 4139.10 (Pointwise stream metric completion separation)**.** *Every completion-facing read of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must expose $S, W, D, R, E, M, H, C, P, N$ before a completion consumer may inspect the metric-distance row. The read cannot infer ambient completeness, quotient stream equality, selected representatives, countable-choice schedules, or host function equality unless the displayed $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and terminal* $\mathsf{Real}^{\uparrow}$ *seal row E are present in the same packet.*

Proof. Project the carrier of Definition 4139.1. The only available rows are the certified stream source, finite window schedule, dyadic tolerance ledger, regular-rational readback, terminal Real seal, metric row, transport, replay, provenance, and local naming row.

Apply the pointwise classifier theorem Definition 4139.2. It forces the distance read through S, W, D, R, E before M is inspected, so a completion consumer cannot start from an ambient completed stream space.

Use the Real-completion anchor Definition 4139.7 and the finite-window consumption theorem Definition 4139.9. Together they place every completion-facing request on the displayed stream-window, dyadic-ledger, RegSeqRat readback, Real-seal, and metric route.

The conclusion is the carrier inventory. Since no coordinate stores ambient completeness, quotient stream equality, selected representatives, countable choice, or host function equality, the completion-facing read can use such data only by leaving the accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet. □

Theorem 4139.11 (Pointwise stream metric synchronized-window triangle route)**.** *Let $Q_{\mathrm{xy}}, Q_{\mathrm{yz}}, Q_{\mathrm{xz}}$ be accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carriers for the displayed stream reads from x to y, from y to z, and from x to z. Suppose the three packets share the same certified* $\mathsf{StreamName}^{\uparrow}$ *window schedule W and the same displayed* $\mathsf{Met}^{\uparrow}$ *row M:*

$$\begin{aligned} Q_{\mathrm{xy}} &= (S_{\mathrm{xy}}, W, D_{\mathrm{xy}}, R_{\mathrm{xy}}, E_{\mathrm{xy}}, M, H_{\mathrm{xy}}, C_{\mathrm{xy}}, P_{\mathrm{xy}}, N_{\mathrm{xy}}), \\ Q_{\mathrm{yz}} &= (S_{\mathrm{yz}}, W, D_{\mathrm{yz}}, R_{\mathrm{yz}}, E_{\mathrm{yz}}, M, H_{\mathrm{yz}}, C_{\mathrm{yz}}, P_{\mathrm{yz}}, N_{\mathrm{yz}}), \\ Q_{\mathrm{xz}} &= (S_{\mathrm{xz}}, W, D_{\mathrm{xz}}, R_{\mathrm{xz}}, E_{\mathrm{xz}}, M, H_{\mathrm{xz}}, C_{\mathrm{xz}}, P_{\mathrm{xz}}, N_{\mathrm{xz}}). \end{aligned}$$

Then any two-leg distance read from x to z through y must consume, in the same synchronized order, the shared finite window W, the dyadic triangle ledgers $D_{\mathrm{xy}}, D_{\mathrm{yz}}, D_{\mathrm{xz}}$, the $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks $R_{\mathrm{xy}}, R_{\mathrm{yz}}, R_{\mathrm{xz}}$, the terminal* $\mathsf{Real}^{\uparrow}$ *seals $E_{\mathrm{xy}}, E_{\mathrm{yz}}, E_{\mathrm{xz}}$, and the displayed metric row M before any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer may use the composed distance. The route is a finite BEDC triangle ledger and not an ambient metric-space triangle proof over host functions, quotient streams, selected representatives, or countable-choice schedules.*

Proof. Project the three carriers. Each packet displays its stream source row, the shared finite window schedule, its dyadic ledger, its regular-rational readback, its Real seal, the shared metric row, transport, replay, provenance, and local naming row.

Apply the pointwise classifier theorem, Definition 4139.2, to the three legs. For Q_{xy}, Q_{yz}, and Q_{xz} the distance read must pass through the displayed window, dyadic ledger, readback, and Real seal before M is inspected.

Use finite-window consumption, Definition 4139.9, for each leg. The two-leg read may compose the x to y and y to z reads only after both have exposed the same window W, and it may compare them with the x to z read only through the displayed dyadic triangle ledger and readback rows.

Apply reindexing stability, Definition 4139.4, to any transported schedule request. The H_{xy}, H_{yz}, and H_{xz} rows, replayed by the corresponding C rows, preserve the shared window and metric row while keeping the three dyadic ledgers, readbacks, Real seals, provenance rows, and local names visible.

Finally invoke completion separation, Definition 4139.10. A completion or modulus consumer can use the composed distance only after the synchronized finite route has displayed all three pointwise packets. Since none of the carriers stores ambient completeness, quotient stream equality, selected representatives, host function equality, or countable choice, the triangle route remains a BEDC finite-window route. □

Theorem 4139.12 (Pointwise stream metric finite-window triangle budget)**.** *Let $Q_{\mathrm{xy}}, Q_{\mathrm{yz}}, Q_{\mathrm{xz}}$ be accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packets sharing a* $\mathsf{StreamName}^{\uparrow}$ *window schedule W and a displayed* $\mathsf{Met}^{\uparrow}$ *row M. A completion-facing triangle-budget read for the three legs may export data only through the three displayed dyadic ledgers $D_{\mathrm{xy}}, D_{\mathrm{yz}}, D_{\mathrm{xz}}$, the three* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks $R_{\mathrm{xy}}, R_{\mathrm{yz}}, R_{\mathrm{xz}}$, the three terminal* $\mathsf{Real}^{\uparrow}$ *seals $E_{\mathrm{xy}}, E_{\mathrm{yz}}, E_{\mathrm{xz}}$, and the shared metric row M, with the source, transport, replay, provenance, and local naming rows of each packet still visible. The budget route has no coordinate for ambient completeness, quotient stream equality, selected representative, host function equality, or countable choice.*

Proof. Apply the synchronized-window triangle route, Definition 4139.11. It already forces the two-leg comparison and the direct comparison to share the same finite window W and metric row M before any completion-facing consumer can read the composed distance.

Use finite-window consumption, Definition 4139.9, on each of the three packets. This exposes the dyadic ledger, regular-rational readback, and Real seal of each leg as displayed finite-window data rather than as an ambient metric-space fact.

Finally apply the NameCert obligations, Definition 4139.5. They keep the source rows, componentwise hsame transports, displayed Cont replays, Pkg provenance, and local names in the exported packet. Since the carrier contains none of the excluded quotient, representative, host-equality, completeness, or choice coordinates, the triangle budget is exactly the listed finite BEDC route. □

Theorem 4139.13 (Pointwise stream metric source separation)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* PointwiseStreamMetric$^{\uparrow}$ *carrier. If two completion-facing reads agree on the metric row* M *but differ in any of the source rows* S, W, D, R, E*, then the packet cannot identify them by* PointwiseStreamMetric$^{\uparrow}$*. Source separation requires the same* StreamName$^{\uparrow}$ *finite window, dyadic ledger,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, and* Met$^{\uparrow}$ *row; no quotient stream equality or host function equality can collapse different source rows.*

Proof. Project the carrier of Definition 4139.1. A completion-facing read exposes the ordered packet $S, W, D, R, E, M, H, C, P, N$, so agreement of the metric row M alone does not determine the source route.

Apply the pointwise classifier theorem Definition 4139.2. It places the certified stream source, finite window schedule, dyadic ledger, regular readback, and Real seal before the metric row is inspected.

Use finite-window consumption, Definition 4139.9, and completion separation, Definition 4139.10. Together they require a completion-facing metric read to keep the same S, W, D, R, E rows in the packet, not merely the same metric row.

Finally apply reindexing stability, Definition 4139.4. Transport by H and replay by C preserve displayed rows componentwise, and the carrier has no quotient stream or host function equality coordinate. Therefore two reads differing in any of S, W, D, R, E remain source-separated inside PointwiseStreamMetric$^{\uparrow}$. □

Theorem 4139.14 (Pointwise stream metric modulus ledger pullback)**.** *Every* RealModulusOfConvergence$^{\uparrow}$ *consumer of an accepted* PointwiseStreamMetric$^{\uparrow}$ *carrier*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

must factor through the displayed certified stream source S*, finite pointwise window* W*, dyadic ledger* D*,* RegSeqRat$^{\uparrow}$ *readback* R*, terminal* Real$^{\uparrow}$ *seal* E*, and metric row* M *before it may read convergence data. The pullback reaches the convergence ledger only through the sibling route of chapter 2074; it exports no detached modulus, quotient stream equality, selected representative, countable-choice schedule, host function equality, or ambient completion principle.*

Proof. Project the carrier of Definition 4139.1. The only completion-facing analytic rows available before the structural data are S, W, D, R, E, M.

Apply the NameCert obligations, Definition 4139.5, together with completion separation, Definition 4139.10. These force any completion consumer to expose the certified stream source, finite windows, dyadic ledger, regular-rational readback, Real seal, metric row, transport, replay, provenance, and local naming before a metric-distance read is public.

Now use the sibling RealModulusOfConvergence$^{\uparrow}$ chapter. Its carrier and route theorems read convergence-modulus data only after the finite StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ tolerance ledger, RegSeqRat$^{\uparrow}$ readback, and terminal Real$^{\uparrow}$ seal are displayed. Therefore the pointwise metric contribution enters the convergence ledger through the same S, W, D, R, E, M route.

Since neither carrier stores a detached modulus selector, quotient stream equality, selected representative, choice schedule, host function equality, or ambient completion theorem, the pullback is exhausted by the displayed rows. □

Theorem 4139.15 (Pointwise stream metric StreamName dependency)**.** *Every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet consumes its pointwise metric rows only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window route of chapter 126 has been displayed. Thus the metric comparison is a pointwise stream-window read through*

$$S, W, D, R, E, M, H, C, P, N$$

and not an external metric-space equality, quotient stream comparison, selected representative, countable-choice schedule, or quotient completion.

Proof. Project the carrier of Definition 4139.1. The source row S and window row W precede the dyadic ledger, regular readback, Real seal, and metric row in the accepted packet.

Use finite-window consumption, Definition 4139.9. It forces every completion-facing metric read to expose the certified $\mathsf{StreamName}^{\uparrow}$ window before any pointwise distance row can be consumed.

Apply synchronized-window triangle routing, Definition 4139.11, and source separation, Definition 4139.13. Together they make the metric comparison depend on the same finite stream-window route rather than on ambient metric equality or a quotient of stream sources.

Finally use modulus ledger pullback, Definition 4139.14. Any convergence-facing consumer reads the pointwise metric contribution through the displayed StreamName, DyadicRatCore, RegSeqRat, Real, and MetricSpace rows, so the dependency edge to chapter 126 is part of the finite BEDC ledger. □

Theorem 4139.16 (Pointwise stream metric regular Cauchy bridge)**.** *Let* $Q = (S, W, D, R, E, M, H, C, P, N)$ *be an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any metric comparison over synchronized stream windows that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *distance seal* E *must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row* R *before the* $\mathsf{Met}^{\uparrow}$ *comparison row* M *can be consumed. Thus the comparison is routed through stream windows, dyadic ledgers, regular Cauchy readback, and the Real seal, not through ambient metric equality, quotient stream equality, selected representatives, countable choice, or host function equality.*

Proof. Start with finite-window consumption, Definition 4139.9. A distance read consumes the certified stream window, dyadic ledger, regular readback, Real seal, and metric row as the displayed packet $S, W, D, R, E, M, H, C, P, N$.

Apply synchronized-window triangle routing, Definition 4139.11. In a synchronized comparison, each leg keeps the shared window schedule visible while its dyadic ledger and readback rows are displayed before the Real distance seal is used.

Use source separation, Definition 4139.13. Agreement at the metric row alone cannot identify two reads; the source route still requires the same stream window, dyadic ledger, regular readback, and Real seal.

Finally invoke modulus ledger pullback, Definition 4139.14. Any convergence-facing or completion-facing consumer receives the pointwise metric contribution only after the $\mathsf{RegSeqRat}^{\uparrow}$ row has pulled the finite rational stream into the regular Cauchy readback surface. Therefore R is exposed before M is consumed. □

Theorem 4139.17 (Pointwise stream metric obligation scope)**.** *The carrier, pointwise classifier, constant-stream route, reindexing stability, NameCert obligations, sibling route, Real-completion anchor, finite-window consumption, completion separation, synchronized-window triangle route, source separation, modulus pullback, StreamName dependency, and regular-Cauchy bridge exhaust the obligation surface of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$*. Any accepted metric read must therefore pass through the displayed stream source, finite windows, dyadic ledger, regular readback, Real seal,*

metric row, componentwise transport, displayed replay, provenance, and local naming before it is public.

Proof. The carrier and classifier results, Definition 4139.1, Definition 4139.2, Definition 4139.3, and Definition 4139.4, determine the packet rows and the basic stable reads.

The NameCert obligations and sibling route then expose the immediate obligation surface. The Real-completion anchor, finite-window consumption, and completion separation theorems force any completion-facing consumer through the same finite windows, dyadic ledger, regular readback, Real seal, and metric row.

The synchronized-window triangle route, source separation, modulus pullback, StreamName dependency, and regular-Cauchy bridge cover the remaining consumer paths. Since each path is routed through the displayed rows $S, W, D, R, E, M, H, C, P, N$, there is no further local obligation for host function equality, quotient stream equality, selected representatives, countable choice, or ambient completion. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet is generated from a certified stream source, finite pointwise windows, a dyadic tolerance ledger, regular-rational readback, a Real distance seal, a displayed metric-space row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 4139.17 exhausts the displayed obligation surface for certified pointwise stream metrics over StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, MetricSpaceUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseStreamMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient stream equality, selected representatives, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding these obligations to their exact primitive routes for stream sources, dyadic ledgers, regular readback, Real seals, metric rows, transport, replay, provenance, and local naming.

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet sits horizontally between the completion route of chapter 1506 and the convergence-budget route of chapter 2074. MetricCompletion consumers read pointwise distance through the certified stream-name rows of chapter 126, the dyadic ledger of chapter 291, the regular rational readback rows of chapter 301, the terminal real-name seal of chapter 38, and the displayed metric carrier rows of chapter 57. RealModulusOfConvergence consumers read the same finite window and dyadic ledger as a convergence budget before any pointwise stream metric row is available. The route exports no host function equality, quotient stream equality, selected representative, countable-choice schedule, or ambient completeness principle.

Theorem 4139.18 (PointwiseStreamMetric L10 dependency route)**.** *Every completion-facing* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *read consumes the certified* $\mathsf{StreamName}^{\uparrow}$ *pointwise windows of chapter 126, the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows of chapter 57, the* $\mathsf{RealUniformStructure}^{\uparrow}$ *entourage reads of chapter 1624, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary of chapter 38 before it can expose a pointwise metric packet to a downstream Real or completion consumer.*

Proof. Start with the certified stream source and finite pointwise windows in the carrier of Definition 4139.1. These rows force every read to pass through $\mathsf{StreamName}^{\uparrow}$ before any distance datum is visible.

Next read the metric row and the Real-uniform entourage row. The sibling chapters cited above provide the displayed distance and entourage surfaces used by the pointwise packet; no host function equality, quotient stream equality, selected representative, or countable-choice schedule is available in the carrier.

Finally apply the Real-completion anchor theorem Definition 4139.7. It keeps the route inside the already displayed stream windows, metric rows, entourage reads, and terminal Real seal, so the L10 dependency route is exactly the concrete sibling route stated above. □

Theorem 4139.19 (Pointwise stream metric completion sibling anchor)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet must first read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{Met}^{\uparrow}$ *distance rows before any completion-facing convergence data are available. The forward lattice route exports only this displayed pointwise metric packet; it does not supply host stream equality, selected representatives, countable-choice schedules, or an ambient completion principle.*

Proof. Project the carrier of Definition 4139.1. The completion-facing analytic rows are the certified stream source S, finite pointwise windows W, dyadic tolerance ledger D, regular-rational readback R, terminal Real seal E, and displayed metric row M.

Apply the pointwise classifier and completion-separation routes in Definition 4139.2 and Definition 4139.10. They force a distance read to consume S, W, D, R, E before M is inspected, and they exclude completion data not carried by the same finite packet.

The $\mathsf{MetricCompletion}^{\uparrow}$ consumer therefore receives convergence-facing data only after the sibling chain

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow}$$

has been displayed. Since the packet contains no host stream equality, selected representative, choice schedule, or ambient completion coordinate, the forward lattice route is exhausted by those rows. □

Every finite-limit or completion-facing consumer of $\mathsf{PointwiseStreamMetric}^{\uparrow}$ must read the displayed $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, and $\mathsf{Met}^{\uparrow}$ row from Definition 4139.1 before entering the finite-limit seal route of chapter 5722 or the completion route of chapter 1506. Thus $\mathsf{FiniteLimitSeal}^{\uparrow}$ and $\mathsf{MetricCompletion}^{\uparrow}$ receive pointwise stream metric data only as the finite window route $S, W, D, R, E, M, H, C, P, N$, not as quotient stream equality, selected representatives, host function equality, countable choice, ambient Real completeness, or a detached completion object. Every $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer reads pointwise metric data only after the finite-window source of chapter 126, the distance row of chapter 57, and the regular rational tolerance readback of chapter 301 are all visible. This route keeps the pointwise stream metric surface attached to displayed StreamName, Metric, and RegSeqRat rows before any downstream argument treats the stream as metric data.

Theorem 4139.20 (Pointwise stream metric sibling repair route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumer of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *must first read the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *rows from Definition 4139.1 and Definition 4139.5. A completion-facing read also passes through the metric-generated uniformity route supplied by chapter 7788 when that chapter is present; before that named bridge is inspected, the dependency is the* $\mathsf{Met}^{\uparrow}$*-to-*$\mathsf{UniformSpace}^{\uparrow}$ *handoff through chapter 57 and chapter 339. Only then may the consumer enter the downstream completion ledgers of chapter 1506, the finite-window boundary of chapter 1559, and chapter 2074.*

Proof. Project the carrier. The stream source, dyadic tolerance ledger, regular-rational readback, terminal Real seal, and metric row are displayed before any completion-facing row can be consumed.

The $\mathsf{MetricCompletionFinite}^{\uparrow}$ packet is the finite-window boundary between the pointwise stream packet and the completion-facing readback.

The uniformity dependency is not a new carrier row. It is the displayed passage from the pointwise metric row to the uniform entourage reader, using the Metric-to-UniformSpace handoff and, when available, the $\mathsf{MetricUniformity}^{\uparrow}$ bridge that packages that route for completion consumers.

Read the obligation surface. It requires carrier admission, pointwise classifier exposure, constant-stream zero-distance exposure, reindexing stability, and ledger non-escape over the same finite rows.

The downstream references therefore consume the pointwise metric packet as a displayed BEDC route. No host completion theorem, quotient stream equality, selected representative, or countable-choice schedule is introduced. □

When a $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance read is used as a completion-facing convergence witness, the pointwise comparison stream leaves the carrier through the convergence ledger of chapter 2074. The route first projects the carrier of Definition 4139.1 and the obligation surface of Definition 4139.5: the certified stream source, finite pointwise windows, dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, terminal $\mathsf{Real}^{\uparrow}$ seal, and displayed metric row are consumed before the RealModulusOfConvergenceUp ledger may read the sequence as completion data. Every $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer that reads a pointwise metric value must pass through the finite $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before the metric comparison is public. The route exposes a finite stream, rational readback, and Real-facing seal boundary; it does not export quotient stream equality, a selected representative, countable choice, or ambient completeness for $\mathsf{Real}^{\uparrow}$. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ hub has a substantive horizontal dependency on chapter 129 and chapter 38: compact-metric finite-net rows and $\mathsf{Real}^{\uparrow}$ distance rows may enter the pointwise stream metric route only through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ carriers already named by the packet. The dependency is therefore a finite stream-and-rational route, not ambient compactness, selected subsequences, quotient stream equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4139.21 (Pointwise stream metric regular-Cauchy pruning route)**.** *Every completion-facing* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *read that prunes redundant pointwise observations must first expose the* $\mathsf{StreamName}^{\uparrow}$ *pointwise windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal already carried by Definition 4139.1. The pruning handoff then factors through the* $\mathsf{RegularCauchyPruning}^{\uparrow}$ *certificate of chapter 7851 before any downstream metric or completion consumer reads the retained stream. The route exports no host stream equality, hidden infinite selector, quotient Real equality, or ambient completeness principle.*

Proof. Project the pointwise stream metric carrier. Its analytic rows expose the stream source, finite pointwise windows, dyadic tolerance ledger, regular rational readback, Real seal, and metric row before any completion-facing consumer has a pointwise distance packet to inspect.

Apply the regular-Cauchy pruning certificate of chapter 7851. That sibling consumes exactly the StreamName windows, retained schedule, dyadic ledger, RegSeqRat readback, and inherited Real seal needed to delete redundant observation requests as finite NameCert data.

Finally use the existing L10 route of Definition 4139.18. The retained stream remains inside the displayed pointwise windows and Real seal boundary before it reaches metric or completion consumers, so the horizontal edge is a concrete stream/pruning dependency rather than a kernel-only pointwise metric export. □

Theorem 4139.22 (Pointwise stream metric stream-metric sibling route)**.** *Every* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer first reads the certified* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126 and the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows of chapter 57. The pointwise stream packet may then hand its finite distance surface to* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *consumers, but only after the stream windows, dyadic tolerance ledger, regular rational readback, terminal Real seal, and metric row have all been exposed.*

Proof. Project the carrier of Definition 4139.1. Its stream source row S, finite window row W, dyadic ledger D, regular readback row R, Real seal E, and metric row M are the only analytic rows available before transport, replay, provenance, and local naming.

Apply Definition 4139.2. It forces the pointwise distance read to pass through S, W, D, R, E before the metric row M can be inspected.

Finally use Definition 4139.7. Any completion-facing or convergence-facing consumer receives the same finite route before reaching $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{RealModulusOfConvergence}^{\uparrow}$. Since no carrier coordinate stores host stream equality, selected representatives, countable-choice schedules, or ambient completeness, the sibling route is exhausted by the displayed stream and metric surfaces. □

Theorem 4139.23 (Pointwise stream metric StreamName completion sibling route)**.** *Every* $\mathsf{PointwiseStreamMetric}$ *completion-facing consumer first reads the certified* $\mathsf{StreamName}^{\uparrow}$ *finite observations of chapter 126 and the* $\mathsf{MetricCompletion}^{\uparrow}$ *existence boundary of chapter 5721. The pointwise packet reaches that boundary only through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{Met}^{\uparrow} \longrightarrow \mathsf{MetricCompletion}^{\uparrow}.$$

Thus the completion read consumes pointwise stream observations, regular rational readback, the terminal Real classifier, and the metric row already displayed in $\mathsf{PointwiseStreamMetric}^{\uparrow}$*; it does not introduce host stream equality, a selected representative, quotient completion, countable-choice scheduling, or ambient completeness.*

Proof. Project the carrier of Definition 4139.1. The stream source S, finite window schedule W, dyadic ledger D, regular readback row R, terminal Real seal E, and metric row M are visible before any structural row can be consumed.

Apply the pointwise classifier and Real-completion anchor, Definition 4139.2 and Definition 4139.7. They force every distance read through the StreamName window, RegSeqRat readback, Real seal, and metric row before a completion-facing consumer receives the packet.

Finally read the completion existence boundary through chapter 5721. Since the incoming route has already exhausted $S, W, D, R, E, M, H, C, P, N$, the completion consumer receives only the displayed sibling dependency surface and cannot acquire host stream equality, quotient completion, selected representatives, countable-choice schedules, or ambient completeness from $\mathsf{PointwiseStreamMetric}^{\uparrow}$. □

The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ route also admits located-real lower-cut rows as completion-facing input data. A metric comparison first reads the certified $\mathsf{StreamName}^{\uparrow}$ windows, passes the rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, consumes the lower-cut row before the terminal $\mathsf{Real}^{\uparrow}$ consumer, and only then exposes the pointwise metric packet to Real and Completion dependents. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet depends substantively on the finite $\mathsf{StreamName}^{\uparrow}$ observation route of chapter 126, the $\mathsf{Met}^{\uparrow}$ distance-witness interface of chapter 57, and the terminal $\mathsf{Real}^{\uparrow}$ seal row of chapter 38. The pointwise metric route is not an isolated kernel float: it consumes stream

windows, metric distance rows, and Real-facing seal rows before any pointwise classifier read is public. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ route consumes the finite-window stream rows of chapter 126 and the distance rows of chapter 57 before any Real seal or completion-facing read is available. This dependency keeps pointwise stream metrics attached to the stream-name carrier and the metric carrier that their local classifier actually reads.

Proposition 4139.24 (Pointwise stream metric StreamName–Metric dependency)**.** *Every* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer that reaches a Real seal, a* $\mathsf{MetricCompletion}^{\uparrow}$ *route, or another completion-facing read first factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite windows and the displayed* $\mathsf{Met}^{\uparrow}$ *distance rows. No completion-facing consumer can bypass those two carrier surfaces.*

Theorem 4139.25 (Pointwise stream metric coordinate extension stability)**.** *Let*

$$Q = (S, W, D, R, E, M, H, C, P, N)$$

be an accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *carrier. Any accepted coordinate extension of* Q *preserves the route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, *and* $\mathsf{MetricCompletion}^{\uparrow}$*: the extended packet must keep the certified stream source, dyadic ledger, regular readback, terminal Real seal, displayed metric row, convergence ledger, and completion-facing dossier visible through the same finite NameCert route. The extension cannot identify quotient streams, select representatives, or replace the displayed rows by host function equality.*

Proof. Begin with the pointwise classifier of Definition 4139.2. It forces the read of Q through S, W, D, R, E before the displayed metric row M is inspected, so any coordinate extension starts from the same stream, dyadic, regular-rational, and Real-seal rows.

Next apply reindexing stability, Definition 4139.4. The transport row H and replay row C preserve the stream source, window schedule, dyadic ledger, regular readback, Real seal, metric row, provenance, and local naming rows under every accepted scheduled replacement.

Use the Real-completion anchor, Definition 4139.7. It routes completion-facing reads through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ before any downstream completion consumer receives the pointwise packet.

Finally invoke the dossier lattice route, Definition 4139.8. A $\mathsf{RealModulusOfConvergence}^{\uparrow}$ or $\mathsf{MetricCompletion}^{\uparrow}$ consumer reads the same finite dossier before convergence or completion data is public. Since none of the cited carrier rows stores quotient stream equality, selected representatives, or host function equality, coordinate extension cannot introduce them. □

Theorem 4139.26 (Pointwise stream metric orphan lattice route)**.** *Every* $\mathsf{MetricCompletionFinite}^{\uparrow}$ *consumer of* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *must first expose the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Met}^{\uparrow}$ *rows of Definition 4139.1 and Definition 4139.5. Only after those rows are visible may the consumer enter the finite-window completion boundary of chapter 1559. The route exports no quotient stream equality, no selected completion representative, no ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*, and no host metric completion object.*

Proof. Project the pointwise stream metric carrier. The stream windows, dyadic tolerance ledger, regular rational readback, Real seal, and metric-space row are all displayed before a completion-facing read has a source.

Apply the NameCert obligations for the same packet. They assign the completion-facing dependency to the visible stream and metric rows rather than to an external completion object.

Thus a $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer can read only the finite row lattice already exposed by $\mathsf{PointwiseStreamMetric}^{\uparrow}$, and the refused quotient, representative, completeness, and host-completion objects remain outside the route. □

Theorem 4139.27 (Pointwise stream metric hub completion sibling route)**.** *Every hub-level* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *completion read first consumes the certified stream-name surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, the metric carrier surface of chapter 57, and the finite completion boundary of chapter 1559 before exporting the pointwise distance packet. The hub exports only the displayed stream, readback, Real, metric, and finite-completion lattice; it exports no detached stream quotient, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.*

Proof. Read the hub through the body carrier and its existing sibling routes. The stream-name windows, regular rational readback, Real seal, and metric distance rows are visible before any pointwise distance packet is exported.

The finite completion boundary is then supplied by $\mathsf{MetricCompletionFinite}^{\uparrow}$. It consumes the same displayed rows as completion-facing data and adds no quotient stream equality, selected representative, countable-choice schedule, host function equality, or ambient completeness theorem.

Thus every hub-level completion read follows the stated sibling lattice before exporting the pointwise stream metric distance packet. □

Theorem 4139.28 (Pointwise stream metric top-level sibling lattice route)**.** *Every top-level read exported by the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *hub first consumes the certified stream-name surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the metric carrier surface of chapter 57. Only after those sibling rows are visible may a pointwise stream metric read expose its local distance packet; the hub exports no detached stream quotient, selected representative, host function equality, countable-choice schedule, or ambient completeness principle.*

Proof. Read the hub through its body chapter, Definition 4139.6. That route already places $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ before the pointwise metric row.

The present top-level route records the same dependency at the hub boundary. Since the hub has no substantive rows of its own, any exported pointwise stream metric read is exactly the sibling lattice route supplied by the body chapter and its Real-modulus dependency file. □

Theorem 4139.29 (Pointwise stream metric real-completion route)**.** *For every accepted* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *packet, each completion-facing read factors through finite* $\mathsf{StreamName}^{\uparrow}$ *coordinate windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows,* $\mathsf{Real}^{\uparrow}$ *distance seals,* $\mathsf{MetricCompletion}^{\uparrow}$ *limit witnesses, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. The route reads the coordinate windows before the rational readback, the rational readback before the Real distance seal, and the Real seal before any metric-completion handoff.*

The packet exports no quotient stream equality, no selected limit representative, and no ambient completeness theorem. It is the same finite L10 route used by the StreamName chapter of chapter 126, the RegSeqRat chapter of chapter 301, the Real chapter of chapter 38, and the metric-completion interface of chapter 1506.

Proof. Project the accepted $\mathsf{PointwiseStreamMetric}^{\uparrow}$ packet onto its coordinate-window, regular-rational readback, real-distance, completion-facing, transport, replay, provenance, and local naming rows. The coordinate windows are finite $\mathsf{StreamName}^{\uparrow}$ data, so a completion read cannot start from an ambient stream equality.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback rows then expose the rational comparisons used by each displayed coordinate distance. The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of those readbacks, not a host real equality detached from them.

Only after those rows are fixed may a $\mathsf{MetricCompletion}^{\uparrow}$ consumer inspect a limit witness. Componentwise hsame transport and Cont replay preserve the same displayed route, while Pkg provenance and local $\mathsf{NameCert}$ naming export no selected representative or ambient completeness principle. □

Chapter 4140

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy completion and Frechet on cofinite filters, the $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ packet records the finite route by which regular Cauchy tail evidence is read as a cofinal Frechet-filter schedule. It sits between the tail-filter certificate of chapter 1605, the finite-tail schedule of chapter 1119, and the Frechet cofinite filter interface of chapter 1564. The chapter contributes a NameCert landing for Real and Completion consumers; it does not select limits, quotient streams, host reals, or an ambient completeness theorem.

Definition 4140.1 (Regular Cauchy tail-filter cofinality carrier)**.** *A* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, G, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *source row, carrying the regular tail-filter evidence already displayed by the sibling tail-filter chapter.*
2. *F is the* $\mathsf{FiniteTailFilter}^{\uparrow}$ *cofinal schedule row, recording the finite tail windows through which later requests are admitted.*
3. *G is the* $\mathsf{FrechetFilter}^{\uparrow}$ *cofinite-filter row produced from the same finite tail schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose inspected windows are exposed only after T, F, G have been replayed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled tail windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same scheduled readback.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular tail, finite-tail, Frechet-filter, stream, readback, and dyadic rows have been displayed.*
8. *H records componentwise* hsame *transport for $T, F, G, S, R, D, A, C, P, N$.*
9. *C records displayed* Cont *replay from a cofinal-tail request through the regular tail-filter source, finite-tail schedule, Frechet-filter row, stream windows, regular rational readback, dyadic tolerances, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$, $\mathsf{FiniteTailFilter}^{\uparrow}$, $\mathsf{FrechetFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows and has no coordinate for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFilterCofinalityUp.RegularCauchyTailFilterCofinalityTasteGate single carrier alignment`

Theorem 4140.2 (Regular Cauchy tail-filter cofinality NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier* $K = (T, F, G, S, R, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, cofinality of the finite-tail schedule between T and G, ordered exposure of* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside $T, F, G, S, R, D, A, H, C, P, N$.*

Proof. Project the carrier of Definition 4140.1. The analytic rows are exactly the regular tail-filter source, finite-tail schedule, Frechet-filter route, stream windows, regular rational readback, dyadic tolerances, and terminal Real seal.

Cofinality is recorded by the row order T, F, G. A request reads the regular Cauchy tail-filter evidence first, then the finite schedule that witnesses cofinal tail access, and only then the Frechet cofinite-filter row.

The remaining rows preserve that route. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are downstream of the cofinal filter route, while H, C, P, N transport, replay, source, and name the same packet. Since the excluded quotient, selector, choice, host-equality, and completeness objects are absent, the listed rows exhaust the local obligations. □

Theorem 4140.3 (Regular Cauchy tail-filter cofinal Frechet route)**.** *Every Frechet-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto \mathsf{FiniteTailFilter}^{\uparrow} \longmapsto \mathsf{FrechetFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}$$

Thus the Frechet cofinite schedule is cofinal for the displayed regular Cauchy tail evidence before any stream, readback, dyadic, or Real-seal row is exported.

Proof. Start with a consumer of $K = (T, F, G, S, R, D, A, H, C, P, N)$. Carrier admission exposes G only as a row downstream of the regular tail-filter source T and the finite-tail schedule F.

Replay the continuation row C. The request consumes the tail evidence named by chapter 1605, then the finite-tail schedule of chapter 1119, and then the Frechet route of chapter 1564.

Only after that cofinal route is displayed can the stream windows S, readback row R, dyadic tolerance row D, and Real seal A be consumed. Componentwise transport and provenance preserve the same order, so the exported Frechet and Real-facing routes are exactly the displayed finite cofinal-tail route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyTailFilterCofinality}^{\uparrow}$ packet is generated from a RegularCauchyTailFilter source row, a FiniteTailFilter cofinal schedule, a FrechetFilter cofinite-filter row, StreamName windows, RegSeqRat readback,

DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyTailFilterCofinality}\uparrow}$, seed)

Scope closed: Definition 4140.1, Definition 4140.2, and Definition 4140.3 seed the finite cofinal route from regular Cauchy tail evidence through finite-tail schedules and Frechet cofinite filters to stream, readback, dyadic, and Real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyTailFilterCofinality}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.RegularCauchyTailFilterCofinalityUp.RegularCauchyTailFilterCofinalityTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal finite-tail scheduling, Frechet-route exactness, stream-readback order, dyadic tolerance preservation, Real-seal boundary, and ledger refusal proofs for the same packet.

Chapter 4141

A Concrete Naming Certificate for $\mathsf{BishopRealPrecision}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopRealPrecision}^{\uparrow}$ packet records the finite precision budgets by which a real-number read is inspected before any completion-facing consumer may treat it as usable $\mathsf{Real}^{\uparrow}$ data. It sits over the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names the visible precision route; it is not quotient equality of streams, a selected limit, a countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4141.1 (Bishop real precision carrier)**.** *A* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (D, W, R, E, M, H, C, Q, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the requested dyadic radius, slack comparison, and endpoint budget for the inspected read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the finite observation schedule consumed by the precision request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational approximants visible through D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the tolerance, window, and regular-readback rows have been displayed.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *handoff row, recording the finite modulus entry that justifies the requested precision budget for later Real and Completion consumers.*
6. *H records componentwise* hsame *transport for tolerance, window, readback, seal, modulus, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a precision request through D, W, R, E, M.*
8. *Q is the* Pkg *provenance over* $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, M, H, C, Q, N$. It has no coordinate for quotient stream equality, a selected host limit, a countable-choice schedule, host real equality, hidden precision repair, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopRealPrecisionUp.BishopRealPrecisionTasteGate single carrier alignment`

Theorem 4141.2 (Bishop real precision NameCert obligations)**.** *For every accepted* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier*

$$P = (D, W, R, E, M, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tolerance, window, readback, seal, modulus, transport, replay, provenance, and naming rows; tolerance-window exactness through D, W; regular rational readback through R; real-seal nonescape through E; modulus handoff through M; componentwise transport through H; displayed replay through C; package provenance through Q; and local naming through N.*

Proof. Project the carrier of Definition 4141.1. The rows $D, W, R, E, M, H, C, Q, N$ are the only coordinates exposed to a precision consumer.

The tolerance and window are fixed first. A request chooses the dyadic budget D and the finite stream schedule W, then reads the regular rational row R along that same displayed route.

The real seal E is downstream of the readback row, while the modulus handoff M records the finite Cauchy budget consumed by later Real and Completion packets. Transport, replay, provenance, and local naming preserve exactly those rows.

Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface and refuse quotient streams, selected limits, countable choice, host real equality, hidden precision repair, and ambient completeness. □

Theorem 4141.3 (Bishop real precision RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *precision read exported by an accepted* $\mathsf{BishopRealPrecision}^{\uparrow}$ *carrier factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before use. Equivalently, a completion-facing consumer can read the requested precision only after the route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

has been replayed inside the same packet.

Proof. Begin with an accepted carrier $P = (D, W, R, E, M, H, C, Q, N)$. The finite stream window W determines which rational approximants may be inspected, and the dyadic tolerance row D determines the budget at which the inspection is valid.

The regular rational readback R is the only row that exports approximants to the precision request. The modulus handoff M records the finite Cauchy budget attached to that readback before the $\mathsf{Real}^{\uparrow}$ seal E is consumed.

Finally, H, C, Q, N transport, replay, source, and name the same route. Thus every accepted precision read reaches $\mathsf{Real}^{\uparrow}$ only through the displayed finite route, and no hidden limit, quotient equality, choice schedule, or ambient completeness principle can supply a bypass. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopRealPrecision}^{\uparrow}$ packet is generated from DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, a CauchyModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealPrecision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4141.1, Definition 4141.2, and Definition 4141.3 seed the finite precision-budget route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopRealPrecision}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, host real equality, hidden precision repair, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window exactness, RegSeqRat readback, Real seal handoff, CauchyModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealPrecisionUp.

Chapter 4142

A Concrete Naming Certificate for HyperbolicVisualShadowMetric$^{\uparrow}$

The HyperbolicVisualShadowMetric$^{\uparrow}$ packet realizes the boundary-shadow visual metric reading of section 9463.3 as finite NameCert data. It consumes disk coordinates from chapter 377, the metric-read surface of chapter 3474, the boundary-shadow packet of chapter 3813, the visual-metric packet of chapter 3866, and the Busemann-boundary route of chapter 3732. Its carrier binds those rows into one finite visual-shadow metric packet rather than a completed disk model, quotient boundary, global field instance, or host hyperbolic geometry.

Definition 4142.1 (Hyperbolic visual-shadow metric carrier)**.** *A* HyperbolicVisualShadowMetric$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (U, B, M, S, G, T, H, C, P, N)$$

with the following displayed rows:

1. *U is the* UnitDisk$^{\uparrow}$ *disk-carrier row for the inspected points and visible disk windows.*
2. *B is the boundary-shadow row, recording the displayed boundary projection or shadow comparison consumed by the read.*
3. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row on the displayed disk data.*
4. *S is the* HyperbolicVisualMetric$^{\uparrow}$ *visual-metric row, recording the finite visual comparison surface.*
5. *G is the* HyperbolicBusemannBoundary$^{\uparrow}$ *boundary row used to keep the shadow comparison attached to a displayed Busemann endpoint.*
6. *T is the phase-transport row, recording the displayed observer-expansion transport through which the visual-shadow read is replayed.*
7. *H records componentwise* hsame *transport for disk, boundary-shadow, metric, visual, Busemann, phase-transport, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a visual-shadow metric request through U, B, M, S, G, T.*
9. *P is the* Pkg *provenance over* HyperbolicVisualShadowMetric$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $U, B, M, S, G, T, H, C, P, N$. It has no coordinate for a host Poincare disk, quotient ideal boundary, global geometry theorem, physical observer trajectory, or implicit field-faithfulness witness.

Lean4 (definition) `BEDC.Derived.HyperbolicVisualShadowMetricUp.HyperbolicVisualShadowMetricUp`

Theorem 4142.2 (Hyperbolic visual-shadow metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *carrier*

$$V = (U, B, M, S, G, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$ *surface consists exactly of disk-carrier admission, boundary-shadow exposure, hyperbolic metric readback, visual-metric readback, Busemann-boundary attachment, phase-transport replay, componentwise transport, displayed continuation, package provenance, and local naming through $U, B, M, S, G, T, H, C, P, N$.*

Proof. Project the carrier of Definition 4142.1. The rows $U, B, M, S, G, T, H, C, P, N$ are the whole visual-shadow metric packet.

The disk and boundary data are read first. The disk carrier U supplies the visible points, the boundary-shadow row B supplies the shadow comparison, and the metric row M supplies the hyperbolic readback attached to those points.

The visual and Busemann rows then bind the comparison to a displayed boundary-facing route. The phase-transport row T records how the observer-expansion reading is replayed without importing a completed geometry.

Finally H, C, P, N transport, replay, source, and name the same finite route. Therefore the listed rows exhaust the NameCert surface and leave host disk models, quotient boundaries, global geometry, physical trajectories, and implicit field-faithfulness outside the packet. □

Lean4 ✓ `BEDC.Derived.HyperbolicVisualShadowMetricUp.HyperbolicVisualShadowMetricNamecertObligations`

Theorem 4142.3 (Hyperbolic visual-shadow metric factorization)**.** *Every accepted visual-shadow metric read factors through the disk carrier, boundary-shadow row,* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row,* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row,* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, phase-transport row, displayed replay, package provenance, and local* $\mathsf{NameCert}$ *row before use.*

Proof. Begin with an accepted carrier $V = (U, B, M, S, G, T, H, C, P, N)$. A read first lands on the disk-carrier row U, then consumes the boundary-shadow row B and the hyperbolic metric row M.

The visual comparison row S and Busemann boundary row G attach that metric read to a displayed boundary-facing visual surface. The phase-transport row T records the route along which the observer-expansion packet replays the comparison.

Transport, continuation, provenance, and naming are then supplied by H, C, P, N. Since no further coordinate is present, every accepted read factors through the displayed rows and cannot bypass them through host geometry, quotient boundary data, or an implicit global field instance. □

Falsifiable prediction: Every accepted visual-shadow metric read factors through the disk carrier, boundary-shadow row, $\mathsf{HyperbolicMetric}^{\uparrow}$ row, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ row, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ row, phase-transport row, displayed replay, provenance, and local $\mathsf{NameCert}$ row.

Independence witness: unitdisk, hyperbolicmetric, hyperbolic-boundary-shadow, hyperbolic-visual-metric, hyperbolic-busemann-boundary: no listed sibling is bijective to $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ because this packet jointly carries disk windows, boundary shadows, metric readback, visual comparison, Busemann boundary attachment, phase transport, replay, provenance, and local naming.

The future `lean4/BEDC/Derived/HyperbolicVisualShadowMetricUp/TasteGate.lean` file supplies explicit $\mathsf{Nontrivial}\ \mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ instances. Those instances must preserve $U, B, M, S, G, T, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ packet is generated from UnitDisk carrier rows, boundary-shadow rows, HyperbolicMetric readback, HyperbolicVisualMetric comparison, HyperbolicBusemannBoundary attachment, phase transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4142.1, Definition 4142.2, and Definition 4142.3 seed the finite boundary-shadow visual metric route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed Poincare disk geometry, quotient ideal boundary equality, global field instances, physical observer trajectories, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, disk readback, boundary-shadow exposure, metric readback, visual comparison, Busemann boundary attachment, phase transport, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicVisualShadowMetricUp.

Chapter 4143

A Concrete Naming Certificate for $\mathsf{LocatedCauchyExtraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ packet records the finite extraction surface that turns located regular-Cauchy modulus evidence into a sealed real-name read. It sits after the dyadic rational ledger of chapter 291, the stream observation window of chapter 126, the regular rational readback of chapter 301, the located-modulus boundary of chapter 3655, the located comparison rows of chapter 1730, and the terminal real-name seal of chapter 38. The packet names the extraction route as displayed BEDC data; it does not quotient Cauchy names, select a limit, use trichotomy, import a host order, or appeal to ambient completeness.

4143.1 Located Cauchy extraction carrier

Definition 4143.1 (Located Cauchy extraction carrier)**.** *A* $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ *carrier is a finite packet* $(D, S, Q, L, U, M, E, H, C, P, N)$ *with the following displayed rows:*

1. D*, a* $\mathsf{DyadicRatCore}^{\uparrow}$ *precision ledger for the positive dyadic budgets used by the extraction;*
2. S*, a* $\mathsf{StreamName}^{\uparrow}$ *observation window for the rational name being read;*
3. Q*, a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the regular Cauchy rational values visible on* S*;*
4. L *and* U*, lower and upper* $\mathsf{LocatedRealComparison}^{\uparrow}$ *rows tied to the same window and dyadic ledger;*
5. M*, the* $\mathsf{RegularCauchyLocatedModulus}^{\uparrow}$ *row assigning displayed precision budgets to finite window depths;*
6. E*, the* $\mathsf{Real}^{\uparrow}$ *seal row that receives only the extracted finite regular-Cauchy read;*
7. H*, the componentwise* hsame *transport rows for* D, S, Q, L, U, M, E*;*
8. C*, the displayed* Cont *replay routes preserving the extraction window, located brackets, modulus row, and real seal;*
9. P*, the* Pkg *provenance rows for the dyadic ledger, stream window, regular readback, located comparison evidence, located modulus, and real seal;*
10. N*, the local* $\mathsf{NameCert}$ *row binding these ten rows as one located Cauchy extraction certificate.*

The carrier contains no quotient equality of Cauchy names, no selected limit, no trichotomy, no host real order predicate, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.LocatedCauchyExtractionUp.LocatedCauchyExtractionTasteGate single carrier alignment`

Theorem 4143.2 (Located Cauchy extraction NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$ *surface for Definition 4143.1 consists of finite admission of the rows* $(D, S, Q, L, U, M, E, H, C, P, N)$*, shared-window alignment for* S*, dyadic precision exactness through* D*, regular readback confinement through* Q*, located bracket export through* L, U*, modulus handoff through* M*, real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4143.1. The projection exposes exactly the rows $D, S, Q, L, U, M, E, H, C, P,$

The extraction route is fixed by the shared window and ledger. Row S is the only observation window, row D is the only dyadic precision source, and row Q is the only regular rational readback exposed to the located comparison and modulus rows.

The located handoff is then forced through L, U, M before the real seal E is read. Componentwise transport and replay through H and C preserve those rows, while P and N record provenance and local naming.

No carrier coordinate names quotient equality, a selected limit, trichotomy, a host order predicate, or ambient completeness. Hence the stated obligations exhaust the local NameCert surface. □

Theorem 4143.3 (Located Cauchy extraction Real-seal route)**.** *Every accepted* $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ *packet exports toward* $\mathsf{Real}^{\uparrow}$ *only the finite route*

$$D, S, Q, L, U, M \longmapsto E \longmapsto H, C, P, N.$$

Any $\mathsf{LocatedRealComparison}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer of the accepted packet must read through the displayed dyadic ledger, stream window, regular rational readback, located comparison rows, located modulus row, transport, replay, provenance, and local NameCert rows. The route supplies no quotient Cauchy equality, selected limit, trichotomy, host order, or ambient completeness.*

Proof. Start with an accepted carrier and apply Definition 4143.2. The obligations pin the regular rational readback to Q over the displayed window S and dyadic ledger D.

The located comparison rows L, U and the located-modulus row M are the only rows that can be exported before the seal E. Therefore every Real-facing consumer reaches the seal by the displayed finite route and not by selecting a limit outside the packet.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since the refused objects are absent from the carrier coordinates, no downstream consumer can recover quotient equality, trichotomy, host order, or ambient completeness from the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedCauchyExtraction}^{\uparrow}$ packet is generated from a dyadic precision ledger, a StreamName observation window, a RegSeqRat readback, lower and upper located comparison rows, a regular-Cauchy located modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedCauchyExtraction}^{\uparrow}}$, seed)

Scope closed: Definition 4143.1, Definition 4143.2, and Definition 4143.3 seed the finite extraction route from located regular-Cauchy modulus evidence to the RealUp seal.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyExtraction}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.LocatedCauchyExtractionUp.LocatedCauchyExtractionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient equality of Cauchy names, selected limits, trichotomy, host real order, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located modulus handoff, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyExtractionUp.

Chapter 4144

A Concrete Naming Certificate for $\mathsf{BanachMazurGame}^{\uparrow}$

Following Banach and Mazur on nested-open games, the $\mathsf{BanachMazurGame}^{\uparrow}$ packet records the finite alternating-play route by which a Polish complete-metric Baire argument is inspected inside BEDC. It sits over the Polish-space synthesis of chapter 1490, the Baire-category surface of chapter 1542, the complete-metric handoff of chapter 128, the metric carrier of chapter 57, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the classical Banach–Mazur game interface as finite NameCert data without adding a selected winning strategy, a host intersection point, countable choice, quotient equality of plays, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4144.1 (Banach–Mazur game carrier)**.** *A* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, B, K, M, L, S, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row naming the complete separable metric source over which the play is read.*
2. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *row exposing the dense-open and residual obligations visible to the game consumer.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, including displayed Cauchy-window and nested-ball support.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *row carrying the distance classifier, metric-radius comparisons, and ball-inclusion checks.*
5. *L is the alternating legal-move ledger, recording each finite play prefix, player tag, chosen basic open or ball row, predecessor inclusion, and shrinking-radius evidence.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule listing the finite play prefixes, dense-open requests, ball observations, and Cauchy-window inspections visible to downstream consumers.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational centers, radius budgets, tolerance comparisons, and dense-window observations used by L and S.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Polish, Baire-category, complete-metric, metric, legal-move, schedule, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for P, B, K, M, L, S, R, E and for the structural rows.*
10. *C records displayed* Cont *replay from a game request through the Polish source, Baire-category obligation, complete-metric handoff, metric-radius checks, legal-move ledger, finite schedule, regular readback, Real seal, transport, provenance, and local naming rows.*
11. *Q is the* Pkg *provenance over* $\mathsf{BanachMazurGame}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting these displayed rows to a common Polish, Baire-category, complete-metric, metric, play-ledger, stream, readback, and real-seal surface. It has no coordinate for a global winning strategy, a chosen play limit, an infinite intersection object, a quotient of plays, countable choice, or an ambient host topology.

Lean4 ✓ `BEDC.Derived.BanachMazurGameUp.TasteGate.BanachMazurGameTasteGate single carrier alignment`

Theorem 4144.2 (Banach–Mazur game NameCert obligations)**.** *For every accepted* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier $G = (P, B, K, M, L, S, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed Polish, Baire-category, complete-metric, metric, legal-move, stream, readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *source compatibility, requiring P, K, M to expose one complete metric source before a legal move can be read;*
3. *Baire-category visibility, requiring each dense-open or residual request to pass through B and the finite schedule S before the play ledger is consumed;*
4. *legal-move exactness, requiring L to store only finite alternating prefixes, predecessor inclusions, and shrinking-radius evidence;*
5. *readback discipline, requiring S, R, E to be read in that order before any real-facing terminal observation is exported;*
6. *structural preservation, requiring H, C, Q, N to transport, replay, source, and name exactly the displayed game route;*
7. *nonescape, excluding global winning strategies, selected limit points, host intersections, quotient equality of plays, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and hidden dense-open oracles.*

No Banach–Mazur-game obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4144.1. The analytic rows are P, B, K, M, L, S, R, E, and the structural rows are H, C, Q, N.

The complete metric source is fixed before play begins. The rows P, K, M expose the Polish packet, complete-metric handoff, and metric-radius checks on one displayed carrier, so a move ledger cannot switch to a second metric source.

The Baire-category part is scheduled. A dense-open or residual request is first read through B and then through the finite stream row S, while L records only the finite legal prefix and the displayed inclusion evidence for the chosen successor.

The terminal read is downstream of regular rational observation. The row R supplies rational centers, radii, and tolerances before E is consumed, and H, C, Q, N preserve the same route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4144.3 (Banach–Mazur Polish–Baire handoff)**.** *Every* $\mathsf{BaireCategory}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachMazurGame}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{BaireCategory}^{\uparrow} \longmapsto \mathsf{BanachMazurGame}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

The route supplies the complete-metric Baire interface by exposing finite alternating play prefixes, dense-open requests, metric-ball inclusions, stream windows, regular rational readback, and the terminal real-name seal. It does not select a winning strategy, choose an intersection point, quotient plays, invoke countable choice, or import ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Polish source. The packet reads P, K, M as the complete separable metric carrier, complete-metric handoff, and metric-radius comparison surface before any game row is inspected.

Next expose the Baire-category request. The row B supplies the dense-open or residual obligation, and the finite schedule S records the play prefix and observation windows through which that request is visible.

The game ledger is then read as finite data. The row L records alternating legal moves, predecessor inclusions, and shrinking-radius evidence, while R supplies the regular rational observations needed to compare centers and radii.

Finally the terminal seal E is reached only after the route has passed through P, K, M, B, L, S, R. The structural rows H, C, Q, N transport, replay, source, and name the same handoff, so the consumer receives the displayed Polish–Baire game interface and no hidden strategy, selected point, quotient play, choice principle, or ambient completeness datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachMazurGame}^{\uparrow}$ packet is generated from a PolishSpace source, BaireCategory dense-open row, CompleteMetric handoff, MetricSpace radius checks, an alternating legal-move ledger, StreamName schedule, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$, seed)

Scope closed: Definition 4144.1, Definition 4144.2, and Definition 4144.3 seed the finite Banach–Mazur route over $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BanachMazurGame}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not claim selected winning strategies, host intersection points, quotient play equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, Baire-category visibility, legal-move exactness, readback discipline, structural preservation, and nonescape for BEDC.Derived.BanachMazurGameUp.

Chapter 4145

A Concrete Naming Certificate for $\mathsf{EffectiveMetricSpace}^{\uparrow}$

Following Bishop and Bridges constructive metric spaces and Weihrauch represented spaces, the $\mathsf{EffectiveMetricSpace}^{\uparrow}$ packet records the finite interface needed by completion consumers before any limit object is formed. It refines the represented-name surface of chapter 1494, the distance obligations of chapter 57, the dyadic tolerance arithmetic of chapter 291, the scheduled name windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real seal of chapter 38.

Definition 4145.1 (EffectiveMetricSpace carrier)**.** *An* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, S, W, \rho, D, R, \delta, \Sigma, \Delta, T, C, P, N)$$

with the following rows:

1. *a point-name row* X *and a represented-space schedule* S, W, ρ *for finite* $\mathsf{StreamName}^{\uparrow}$ *reads;*
2. *a dyadic tolerance row* D *and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *for all inspected distance approximants;*
3. *a* $\mathsf{Real}^{\uparrow}$*-valued distance seal* δ *whose admitted reads pass through* D *and* R*;*
4. *symmetry and triangle replay rows* Σ, Δ *for the displayed distance seal;*
5. *componentwise transport* T*, replay* C*, provenance* P*, and a local* $\mathsf{NameCert}$ *row* N*.*

The carrier is accepted only when every distance request exposes a scheduled name window, a dyadic tolerance, a regular rational readback, and the resulting $\mathsf{Real}^{\uparrow}$ *seal before a completion-facing consumer can inspect it.*

Lean4 ✓ `BEDC.Derived.EffectiveMetricSpaceUp.EffectiveMetricSpaceTasteGate` single carrier alignment

Theorem 4145.2 (EffectiveMetricSpace NameCert obligations)**.** *Every accepted* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier* $E = (X, S, W, \rho, D, R, \delta, \Sigma, \Delta, T, C, P, N)$ *supplies the finite NameCert obligations needed by an* $\mathsf{EffectiveMetricCompletion}^{\uparrow}$ *consumer: point-name admission, represented-space scheduling, dyadic tolerance tracking, regular rational readback, Real-valued distance sealing, symmetry replay, triangle replay, componentwise transport, provenance, and local naming.*

Proof. Project the carrier of Definition 4145.1. The only rows available to a completion-facing reader are the displayed point names, scheduled representation reads, tolerance ledger, rational readback, real seal, replay rows, transport, provenance, and local NameCert.

The represented-space part is sourced by chapter 1494. The tolerance and readback rows then pass through chapter 291 and chapter 301, so the distance seal cannot be opened without the finite arithmetic route.

Finally the symmetry and triangle rows replay only that displayed seal. Transport and provenance carry the same finite packet into the local NameCert row, excluding host equality of points, selected representatives, countable choice, and ambient completion data. □

Theorem 4145.3 (EffectiveMetricSpace distance readback)**.** *For any two accepted point names in an* $\mathsf{EffectiveMetricSpace}^{\uparrow}$ *carrier, each distance read available to a Real or completion consumer factors through*

$$S, W, \rho \longmapsto D \longmapsto R \longmapsto \delta \longmapsto \Sigma, \Delta, T, C, P, N.$$

The read is therefore a finite scheduled $\mathsf{Real}^{\uparrow}$ *seal assembled from* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *data, not a host metric value, quotient representative, selected Cauchy name, or pre-existing completion point.*

Proof. Start with the two point names and their represented-space schedule. The carrier exposes only the finite $\mathsf{StreamName}^{\uparrow}$ windows and representation relation before any distance request can be made.

Next consume the dyadic and regular-rational rows. They produce the tolerance-indexed readback that feeds the $\mathsf{Real}^{\uparrow}$ distance seal, so the downstream reader sees a certified real value rather than an ambient metric value.

The replay rows Σ and Δ preserve symmetry and triangle evidence for the same seal. The transport, replay, provenance, and local naming rows then package the read for Real and completion consumers without adding a quotient, a selected representative, a choice principle, or an already completed point. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffectiveMetricSpace}^{\uparrow}$ packet is generated from displayed point-name rows, a represented-space schedule, dyadic tolerance data, regular rational readback, a Real-valued distance seal, symmetry and triangle replay rows, componentwise transport, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4145.1, Definition 4145.2, and Definition 4145.3 seed the finite effective-metric source interface consumed by Real and completion chapters.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close metric completion, located limits, quotient metric spaces, selected representatives, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, represented-space scheduling, dyadic and regular-rational readback, Real seal stability, symmetry and triangle replay, transport, provenance, and local naming.

Chapter 4146

A Concrete Naming Certificate for $\mathsf{ComputablePolishSpace}^{\uparrow}$

Following Weihrauch and Alexandrov–Urysohn, the $\mathsf{ComputablePolishSpace}^{\uparrow}$ packet records the complete separable metric object whose dense names, Cauchy realizers, representation decoder, and terminal real seal are all visible before any computable-analysis consumer reads the space. It sits after the Polish-space boundary of chapter 1490, the effective complete-separable route of chapter 1706, and the represented-space interface of chapter 1494. The packet also consumes the Cauchy-representation surface of chapter 1680, the explicit-modulus Cauchy route of chapter 1705, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. It is a BEDC finite naming surface for computable Polish data, not a quotient of names, a selected dense family, host equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4146.1 (Computable Polish space carrier)**.** *A* $\mathsf{ComputablePolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (P, E, R, A, Q, S, G, D, L, H, K, U, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *row, carrying the complete separable metric boundary read by ordinary Polish-space consumers.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *row, exposing the indexed dense family, explicit realizer route, and effective complete-separable metric data.*
3. *R is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *row, presenting the finite name interface through which computable names reach target objects.*
4. *A is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *decoder row, routing stream observations through regular rational readback, dyadic tolerances, and the represented-space boundary.*
5. *Q is the* $\mathsf{EffectiveCauchySequence}^{\uparrow}$ *row, carrying the explicit-modulus Cauchy sequence data used by the realizer route.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the representation decoder and the effective Cauchy route.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and Cauchy stages.*

8. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger used for the decoder, regular readback, effective Cauchy modulus, and Real seal thresholds.*
9. *L is the terminal* Real$^{\uparrow}$ *seal row reached only after A, Q, S, G, D have been displayed.*
10. *H records componentwise* hsame *transport for the Polish, effective Polish, represented-space, Cauchy-representation, effective Cauchy, stream-window, regular-readback, dyadic-ledger, Real-seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a computable-name request through R, A, S, Q, G, D, L and back to the Polish and effective-Polish rows P, E.*
12. *U is the* Pkg *provenance over* ComputablePolishSpace$^{\uparrow}$, PolishSpace$^{\uparrow}$, EffectivePolishSpace$^{\uparrow}$, RepresentedSpace$^{\uparrow}$, CauchyRepresentation$^{\uparrow}$, EffectiveCauchySequence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{ComputablePolishSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting the displayed rows to a common complete-separable, represented, Cauchy-realizer, stream-window, regular-readback, dyadic-tolerance, and Real-seal surface. It has no coordinate for quotient names, selected dense points, host equality of represented objects, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ComputablePolishSpaceUp.ComputablePolishSpaceTasteGate single carrier alignment`

Theorem 4146.2 (Computable Polish space NameCert obligations)**.** *For every accepted* ComputablePolishSpace$^{\uparrow}$ *carrier $C = (P, E, R, A, Q, S, G, D, L, H, K, U, N)$, the local* NameCert$_{\text{ComputablePolishSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed Polish, effective Polish, represented-space, Cauchy-representation, effective Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real-seal, transport, replay, provenance, and local naming rows;*
2. *complete-separable compatibility, requiring P and E to expose the same complete separable metric object through the displayed dense-realizer route;*
3. *representation exactness, requiring every computable name to pass through the represented-space row R and Cauchy-representation decoder A before a target-facing read is available;*
4. *effective Cauchy exactness, requiring the explicit row Q and the finite stream windows S to precede the regular rational readback G, dyadic ledger D, and Real seal L;*
5. *componentwise transport, continuation replay, package provenance, and local naming through H, K, U, N for exactly the displayed rows;*
6. *ledger nonescape, excluding quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No computable-Polish-space obligation is stored outside these rows.

Proof. Project the carrier of Definition 4146.1. The projection exposes the Polish row P, effective Polish row E, represented-space row R, Cauchy-representation decoder A, effective Cauchy row Q, stream windows S, regular readback G, dyadic ledger D, Real seal L, and structural rows H, K, U, N.

The complete-separable compatibility row reads the sibling surfaces of chapter 1490 and chapter 1706. Since P and E occur in the same finite packet, a consumer cannot replace the displayed dense-realizer route by a second dense family or a hidden completion equality.

The representation route is then forced through chapter 1494 and chapter 1680. A computable name first exposes R and A, then reads the finite stream windows rather than an unscheduled name oracle.

The effective Cauchy route uses chapter 1705, chapter 301, chapter 291, and chapter 38. These rows place Q, S, G, D before L, so the Real seal is downstream of the explicit modulus and tolerance ledger.

Finally, H, K, U, N transport, replay, source, and name only that displayed packet. They introduce no quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, or ambient Real completeness, and the listed rows exhaust the local NameCert surface. □

Theorem 4146.3 (Computable Polish space realizer handoff)**.** *Every represented-space, Polish-space, or Real-facing consumer of an accepted* $\mathsf{ComputablePolishSpace}^{\uparrow}$ *carrier* $C = (P, E, R, A, Q, S, G, D, L, H, K,$ *factors through the displayed route*

$$R, A \longmapsto S \longmapsto Q \longmapsto G \longmapsto D \longmapsto L \longmapsto P, E, H, K, U, N.$$

Thus the consumer receives the represented-space name interface, Cauchy-representation decoder, finite StreamName windows, EffectiveCauchySequence modulus route, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, Polish row, effective Polish row, transport, replay, provenance, and local naming before it can inspect the completed computable Polish object.

Proof. Begin with a represented-space, Polish-space, or Real-facing read of the carrier. The only name-facing rows are R and A, so the read must expose the represented-space interface and the Cauchy-representation decoder before any target row can be consumed.

Next replay the finite observation route. The row S supplies the inspected stream windows, while the effective Cauchy row Q supplies the explicit modulus route attached to those windows.

The regular and Real-facing handoff is then fixed by G, D, L. The regular rational readback G is read from the displayed windows and effective Cauchy stages; the dyadic ledger D records the tolerance data; and the Real seal L is terminal for that route.

Only after this handoff can the Polish and effective Polish rows P, E be read as a completed computable Polish object. The structural rows H, K, U, N preserve and name the same route, so no consumer can bypass it through quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{ComputablePolishSpace}^{\uparrow}$ represented-space, Polish-space, or Real-facing read factors through a represented-space row, CauchyRepresentation decoder, StreamName finite windows, EffectiveCauchySequence modulus route, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, PolishSpace row, EffectivePolishSpace row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: polishspace, effective-polish-space, representedspace, cauchy-representation, effective-cauchy-sequence, streamname, regseqrat, dyadicratcore, real: $\mathsf{ComputablePolishSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries all of their displayed rows as one computable complete-separable name interface.

Closure status for $\mathsf{NameCert}_{\mathsf{ComputablePolishSpace}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\mathsf{ComputablePolishSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4146.1, Definition 4146.2, and Definition 4146.3 seed the finite computable Polish route over PolishSpaceUp, EffectivePolishSpaceUp, RepresentedSpaceUp, CauchyRepresentationUp, EffectiveCauchySequenceUp, StreamNameUp, RegSeqRatUp, DyadicRatCoreUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ComputablePolishSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected dense points, host equality, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, complete-separable compatibility, representation exactness, effective Cauchy exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.ComputablePolishSpaceUp.

Constructive story: A ComputablePolishSpace$^{\uparrow}$ packet is generated from PolishSpace, EffectivePolishSpace, RepresentedSpace, CauchyRepresentation, EffectiveCauchySequence, StreamName, RegSeqRat, DyadicRatCore, RealUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4147

A Concrete Naming Certificate for $\mathsf{LusinApproximation}^{\uparrow}$

Following Lusin and Bishop–Bridges constructive measure theory, the $\mathsf{LusinApproximation}^{\uparrow}$ packet records the finite route by which a measurable real-valued row is approximated on a large displayed set by a continuous row. It sits after the measurable carrier of chapter 92, the continuous-map surface of chapter 130, the integral readback route of chapter 123, the terminal real-name seal of chapter 38, the finite exceptional-window route of chapter 2204, and the density ledger of chapter 3448. The carrier exposes the approximant, tolerance, large-set, integral, replay, and naming rows needed by later measure-analysis packets; it is not a host regularity theorem, selected null-set completion, quotient almost-everywhere equality, or countable-choice construction.

Definition 4147.1 (Lusin approximation carrier)**.** *A* $\mathsf{LusinApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, A, C, F, E, I, R, H, Q, P, N)$$

with the following displayed rows:

1. *M is the ambient finite* $\mathsf{Measure}^{\uparrow}$ *row, exposing the measurable-set ledger and the finite measure budget against which the exceptional set is read.*
2. *A is the finite large-set and exceptional-window row, carrying the measurable subset on which the approximation is certified and the displayed complement budget.*
3. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *approximant row, naming the continuous candidate consumed by the public approximation read.*
4. *F is the measured function row, stored as finite measurable readback data rather than as an ambient almost-everywhere equivalence class.*
5. *E is the error-tolerance row, recording the requested* $\mathsf{Real}^{\uparrow}$ *tolerance, the pointwise comparison window, and the bound used outside the exceptional set.*
6. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row, exposing the finite integral or size comparisons used when a consumer asks for quantitative approximation data.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *seal that compares the measured function, continuous approximant, tolerance, and integral-facing reads through the displayed finite route.*
8. *H records componentwise* hsame *transport for the measure, large-set, continuous, function, error, integral, Real, replay, provenance, and naming rows.*

9. *Q records displayed* Cont *replay of the route*

$$M, A, C, F, E, I, R \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* LusinApproximation$^{\uparrow}$, Measure$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Integral$^{\uparrow}$, Real$^{\uparrow}$, Egorov$^{\uparrow}$, RadonNikodym$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{LusinApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, A, C, F, E, I, R, H, Q, P, N$. It has no coordinate for a host Lusin theorem, completed measure algebras, selected null sets, quotient almost-everywhere equality, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LusinApproximationUp.TasteGate.LusinApproximationTasteGate single carrier alignment`

Theorem 4147.2 (Lusin approximation NameCert obligations)**.** *For every accepted* LusinApproximation$^{\uparrow}$ *carrier*

$$L = (M, A, C, F, E, I, R, H, Q, P, N),$$

the local NameCert$_{\text{LusinApproximation}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite* Measure$^{\uparrow}$ *row M, the measured function row F, and the continuous approximant row C;*
2. *large-set and exceptional-window exposure through A, with the finite complement budget read in the same measure ledger as M;*
3. *tolerance and comparison exactness through E and the* Real$^{\uparrow}$ *seal R, requiring each public approximation read to display its error budget;*
4. *quantitative readback through I, requiring integral-facing or size-facing consumers to remain on the same finite route;*
5. *structural stability through H, Q, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve exactly the rows M, A, C, F, E, I, R.*

No obligation may be discharged by importing a host Lusin theorem, completing the measure algebra, choosing null representatives, quotienting by almost-everywhere equality, or appealing to countable choice.

Proof. Project the carrier of Definition 4147.1. The measure-facing rows are M and A, the approximation-facing rows are C, F, E, R, and the quantitative readback row is I.

The large-set budget is visible before any approximation conclusion is named. This is the same finite exceptional-window discipline used by chapter 2204, but the public consumer here reads a continuous approximant row rather than a uniform convergence window.

The density and integral-facing dependency is also explicit. The route through I and R mirrors the eventwise readback discipline of chapter 3448, while refusing completed measure algebras and quotient almost-everywhere classes.

Finally H, Q, P, N transport, replay, source, and name only those displayed rows. Since the excluded host theorem, null-set selector, quotient, choice, and completeness coordinates are absent from the carrier, the five listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{LusinApproximation}^{\uparrow}$ packet is generated from finite Measure rows, large-set and exceptional-window ledgers, ContinuousMap approximants, measurable function readback, Real tolerance seals, Integral readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4147.1 and Definition 4147.2 seed the finite large-set continuous-approximant route from Measure and Real tolerance rows to ContinuousMap and Integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LusinApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host Lusin theorem, completed measure algebra, selected null representatives, quotient almost-everywhere equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, large-set budget exactness, continuous approximant admission, function readback, tolerance sealing, integral readback, transport stability, and replay stability for BEDC.Derived.LusinApproximationUp.

Chapter 4148

A Concrete Naming Certificate for $\mathsf{StreamNameModulus}^{\uparrow}$

Following Bishop and Bridges, a constructive modulus is treated as displayed finite control on later observations rather than as a selected limit object. The $\mathsf{StreamNameModulus}^{\uparrow}$ packet sits between the stream-name observation surface of chapter 126, the dyadic readback rows of chapter 291, the regular rational consumer of chapter 301, and the real-seal boundary of chapter 38. It records the finite witness by which a StreamName observation budget can be handed to RegSeqRat and Real consumers without claiming a completed stream quotient, a host-selected tail, or equality of real representatives.

Definition 4148.1 (Stream-name modulus carrier)**.** *A* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row, containing the requested stream name, observed positions, and local observation budget.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback row for the inspected positions.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row that receives the finite dyadic windows and their regularity ledger.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal-request row, recording only the downstream consumer boundary that may later consume R.*
5. *W is the modulus ledger: for each requested precision it displays a finite stream position, the associated dyadic readback, and the regular-rational handoff row visible at that position.*
6. *H records componentwise* hsame *transport for S, D, R, E, W and the local certificate data.*
7. *C records displayed* Cont *replay from a precision request to the stream position, from that position to the dyadic readback, and from the readback to the RegSeqRat handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for a completed stream quotient, an ambient limit selector, a host real equality proof, or a tail chosen outside the displayed observation ledger.

Lean4 ✓ `BEDC.Derived.StreamNameModulusUp.StreamNameModulusTasteGate single carrier alignment`

Theorem 4148.2 (Stream-name modulus pointwise stability)**.** *Let $U = (S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier. If a downstream consumer transports the displayed stream, dyadic readback, RegSeqRat handoff, Real seal-request, and modulus ledger along the componentwise* hsame *rows in H, then each precision request reads the same finite position of S, the same dyadic row of D, and the same regular-rational handoff in R after replay through C.*

Lean4 ✓ `BEDC.Derived.StreamNameModulusUp.StreamNameModulusCarrier pointwise stability`

Proof. Project the carrier of Definition 4148.1. The only precision-facing data are the rows S, D, R, E, W together with transport, replay, provenance, and the local name.

The modulus ledger W binds each requested precision to one displayed stream position and one displayed dyadic readback. Componentwise transport through H moves those rows together, so a transported request cannot alter the position without also altering the visible ledger row.

Replay through C reconstructs the same path from the precision request to S, then to D, then to R. The rows P and N only source and name this finite route; they do not add an external tail or equality proof. □

Theorem 4148.3 (Stream-name modulus monotone reindex stability)**.** *Let $U = (S, D, R, E, W, H, C, P, N)$ be an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier. If a later finite-window request is transported through the modulus ledger W, then the transported request preserves the same* $\mathsf{StreamName}^{\uparrow}$ *point certificate, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, and the same downstream* $\mathsf{Real}^{\uparrow}$ *seal route before any consumer may read a Real-facing conclusion.*

Proof. Project the carrier of Definition 4148.1. The rows S, D, R, E, W display the source point certificate, dyadic readback, regular-rational handoff, Real seal request, and modulus ledger for every accepted precision request.

Apply Definition 4148.2. A transported later request may move along the ledger W, but the corresponding stream position, dyadic readback, and regular-rational handoff remain attached to the same componentwise transport and replay rows.

Finally apply Definition 4148.4. The later finite-window request reaches the RegSeqRat handoff only through S, D, W, H, C, P, N, and the Real seal row E is read only after that handoff. Thus monotone reindexing cannot detach a tail selector from the displayed StreamName point certificate or from the Real-facing route. □

Theorem 4148.4 (Stream-name modulus RegSeqRat readiness)**.** *An accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *packet can be consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *row only as the finite route*

$$S, D, W \longmapsto R$$

together with H, C, P, N. The handoff supplies displayed stream positions, dyadic observations, and regularity ledger rows, but it does not supply a completed sequence quotient, a selected tail, a host convergence theorem, or a real-equality witness.

Proof. Begin with the carrier projection. The RegSeqRat row R is not independent; it is stored beside the stream source S, dyadic readback D, and modulus ledger W.

Use pointwise stability from Definition 4148.2. Every transported precision request still reaches R through the same visible stream position and dyadic observation.

The RegSeqRat consumer therefore receives only the displayed finite route and its support rows. Since the carrier has no coordinate for quotient sequences, selected tails, host convergence, or real equality, none of those objects can be exported at the RegSeqRat boundary. □

Theorem 4148.5 (Stream-name modulus Real handoff)**.** *A* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *packet may import only the RegSeqRat-facing handoff R, the Real seal-request row E, and the finite stream/dyadic modulus route S, D, W, H, C, P, N. The packet therefore prepares a RealUp handoff by finite observed windows and regular rational rows, without asserting completeness of* $\mathsf{Real}^{\uparrow}$*, equality of Real representatives, or convergence outside the carried ledger.*

Proof. Apply Definition 4148.4. It identifies R as the only regular-rational object exported by the packet and forces the route through S, D, W, H, C, P, N.

The Real-facing row E records only that this regular-rational handoff is the object a Real consumer may later inspect. It is a seal-request row, not a proof of Real completeness or equality.

Thus the Real consumer receives the same finite modulus route that the RegSeqRat consumer receives, together with the request row E. No ambient limit, quotient equality, or external convergence datum is available. □

Theorem 4148.6 (Stream-name modulus NameCert obligations)**.** *For every accepted* $\mathsf{StreamNameModulus}^{\uparrow}$ *carrier $U = (S, D, R, E, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-position exposure, dyadic readback, RegSeqRat handoff, Real seal-request boundary, modulus-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is discharged inside the displayed finite route and no obligation is stored outside U.*

Proof. Carrier admission is the projection of the finite tuple in Definition 4148.1. The rows S, D, R, E, W expose the stream source, dyadic readback, regular-rational handoff, Real request, and modulus ledger.

Pointwise stability supplies the transport obligation. Theorem 4148.2 keeps precision requests attached to the same displayed stream and dyadic rows under hsame movement.

The RegSeqRat readiness and Real handoff theorems supply the consumer-facing obligations. They show that R and E can be read only through the carried finite route.

Finally C, P, N replay, source, and name the packet. Since no remaining carrier coordinate exists, the listed obligations exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{StreamNameModulus}^{\uparrow}$ consumer that reads a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ handoff without first exposing the displayed $\mathsf{StreamName}^{\uparrow}$ positions, $\mathsf{DyadicRatCore}^{\uparrow}$ readbacks, modulus ledger, componentwise hsame rows, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row refutes the packet surface.

Independence witness: streamname, dyadicratcore, regseqrat, real: none of the listed sibling chapters supplies the whole modulus handoff $U = (S, D, R, E, W, H, C, P, N)$, because each sibling contributes only the source stream, dyadic readback, regular rational consumer, or Real seal boundary while this chapter binds them into one finite modulus route.

Closure status for $\mathsf{NameCert}_{\mathsf{StreamNameModulus}^{\uparrow}}$.

Constructive story: A StreamNameModulus$^{\uparrow}$ packet is generated from a StreamName$^{\uparrow}$ source row, DyadicRatCore$^{\uparrow}$ readbacks, a RegSeqRat$^{\uparrow}$ handoff, a Real$^{\uparrow}$ seal-request row, a modulus ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{StreamNameModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4148.1, Definition 4148.2, Definition 4148.4, Definition 4148.5, and Definition 4148.6 seed the finite modulus handoff from StreamName$^{\uparrow}$ observations to RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers.

Formal verification: FormalStatus(NameCert$_{\text{StreamNameModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, selected tails, host convergence, Real equality, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite modulus ledger, RegSeqRat readiness, and Real handoff obligations for BEDC.Derived.StreamNameModulusUp.

Chapter 4149

A Concrete Naming Certificate for $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$

The $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ packet realizes the finite verification trace of Definition 9393.4 as a concrete NameCert surface. It sits beside the finite-trace GAP socket of chapter 504 and the closed observation ledger of chapter 616. Its role is to filter observed trace rows through finite kernel checks while keeping the universal-induction boundary visible as a GAP-facing refusal.

Definition 4149.1 (Finite observation trace filter carrier)**.** *A* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, T, K, A, G, H, C, P, N)$$

with the following displayed rows:

1. *O is the closed-observation source row containing the finite observed states and their monotone ledger.*
2. *T is the finite verification trace row, listing inspected indices, their predicates, and the closed-generation checks visible at those indices.*
3. *K is the kernel-filter row that records which trace entries are accepted by finite verification and which are rejected as outside the carried window.*
4. *A is the audit row for non-escape: it records that no universal quantifier, induction schema, future-observation oracle, or hidden host rule is imported by the filter.*
5. *G is the GAP-socket row exposed when a consumer asks the finite trace to act as a universal statement.*
6. *H records componentwise* hsame *transport for O, T, K, A, G and the local certificate data.*
7. *C records displayed* Cont *replay from an observation request to the trace row, from the trace row to the kernel filter, and from a universal request to the GAP row.*
8. *P is the* Pkg *provenance over* BHist*, closed observation, finite-trace rows, GAP sockets,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these rows. It has no coordinate for a future observation, a substrate-internal induction principle, a universal closure over all indices, or a host-level truth predicate.

Theorem 4149.2 (Finite trace filter acceptance boundary)**.** *For every accepted* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *carrier* $F = (O, T, K, A, G, H, C, P, N)$*, a trace entry is available to a downstream consumer only after it appears in the finite trace row* T*, is checked by the kernel-filter row* K*, and is replayed through* C *with provenance* P *and local name* N*. Entries outside the displayed trace are not refused as false; they are routed to the GAP row* G *when a consumer asks for universal closure.*

Proof. Project the carrier. The observed source is O, the trace is T, and the filter row is K; no other row stores accepted trace entries.

The replay row C makes the accepted route explicit. It starts from the observation request, reads the finite trace entry, and then checks that entry against K.

When the requested entry is not in the displayed trace, the packet exposes the GAP row G rather than converting absence into a universal claim. The audit row A records this non-escape boundary, while P, N source and name the finite route. □

Theorem 4149.3 (Finite trace filter NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$ *surface consists exactly of closed-observation source admission, finite trace exposure, kernel-filter acceptance, non-escape audit, GAP-socket routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations certify finite trace filtering only; they do not certify induction, field completeness, universal closure, or future-observation coverage.*

Proof. Carrier admission exposes the tuple

$$F = (O, T, K, A, G, H, C, P, N).$$

The source, trace, filter, audit, and GAP rows are the only content rows.

Apply the acceptance boundary of Definition 4149.2. It identifies the accepted route through T, K, C, P, N and routes universal requests to G.

Transport through H preserves this row structure componentwise. Since the carrier has no row for induction, field completeness, universal closure, or future observations, the listed rows exhaust the NameCert obligations. □

Theorem 4149.4 (Finite trace filter TasteGate worklist)**.** *Any Lean TasteGate implementation for* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *must separate finite trace acceptance from algebraic adequacy by carrying explicit* `Nontrivial` *and* `FieldFaithful` *instance obligations for the numeric side of the filter. The paper packet records the finite trace boundary only; those instances are downstream formal work and cannot be inferred from closed observation rows alone.*

Lean4 ✓ `BEDC.Derived.FiniteObservationTraceFilterUp.FiniteObservationTraceFilterTasteGate`
`single carrier alignment`

Proof. The filter carrier records finite observation, trace, kernel acceptance, audit, GAP, transport, replay, provenance, and naming rows. None of these rows asserts that a numeric carrier has more than one element or that field operations are faithful to the intended readback.

The acceptance boundary therefore cannot discharge `Nontrivial` or `FieldFaithful` by projection. Those obligations must be visible in the Lean TasteGate work for any numeric consumer of the filter.

This keeps the concrete packet honest: it certifies finite observation filtering, while algebraic adequacy remains a separate formal instance surface. □

Theorem 4149.5 (Finite trace filter GAP handoff)**.** *If a downstream consumer treats an accepted* $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ *packet as evidence for a universal statement, the packet exports only the finite accepted rows and the GAP socket G. It supplies no substrate-internal induction rule, no future-observation oracle, no total trace closure, and no host truth predicate.*

Proof. Start with the NameCert obligations. They expose finite source, trace, filter, audit, GAP, transport, replay, provenance, and naming rows.

A universal request is not one of the accepted trace entries. By the acceptance boundary, such a request is routed to G rather than accepted by K.

The non-escape audit row A records that this is a refusal boundary. Since the carrier has no row for induction, future observations, total closure, or a host truth predicate, the GAP handoff is the only universal-facing export. □

Falsifiable prediction: An accepted $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ consumer that obtains a universal statement, induction rule, future-observation value, or host truth predicate without passing through the displayed GAP row G refutes the filter surface.

Independence witness: finite-trace-gap-socket, closedobservationsystem: the GAP socket names the universal request boundary and the closed-observation chapter names monotone finite records, but neither chapter supplies the kernel-filter row K and audit row A that classify accepted finite trace entries.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteObservationTraceFilter}^{\uparrow}$ packet is generated from closed-observation rows, finite verification trace rows, kernel-filter acceptance rows, audit rows, GAP socket rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4149.1, Definition 4149.2, Definition 4149.3, Definition 4149.4, and Definition 4149.5 seed the finite trace filter boundary for closed observation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationTraceFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim induction, universal trace closure, future-observation coverage, field adequacy, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove finite trace acceptance, GAP handoff, and the TasteGate instance boundary for BEDC.Derived.FiniteObservationTraceFilterUp.

Chapter 4150

A Concrete Naming Certificate for $\mathsf{LocatedCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedCompletion}^{\uparrow}$ packet records the finite located completion surface used by the L10 Real and Completion route. It sits between the located metric source of chapter 1533, the separated completion boundary of chapter 1612, the dense completion entry of chapter 1512, and the located completion space interface of **??**. The packet is a displayed NameCert route for located completion data; it is not a quotient completion, a selected representative, a countable-choice schedule, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4150.1 (LocatedCompletion carrier)**.** *A* $\mathsf{LocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, E, W, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{LocatedMetric}^{\uparrow}$ *source row, including the metric carrier, located distance comparison, and finite request discipline consumed before completion data are public.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row, read before completed names may be compared as separated.*
3. *E is the* $\mathsf{CompletionEmbedding}^{\uparrow}$ *row by which source observations enter the completion-facing route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed before any regular rational readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling located comparison, Cauchy tail requests, and zero-distance checks.*
7. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after M, S, E, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the located metric source, separated boundary, embedding, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion request through the same ordered rows.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCompletion}^{\uparrow}$, $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, CompletionEmbedding $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ *row.*
The classifier compares accepted carriers only through $M, S, E, W, R, D, A, H, C, P, N$. It has no coordinate for quotient completion equality, selected completed representatives, host equality, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompletionUp.LocatedCompletionTasteGate single carrier alignment`

Theorem 4150.2 (LocatedCompletion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCompletion}^\uparrow$ *carrier $L = (M, S, E, W, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ *surface consists exactly of carrier admission, located metric source exposure, separated completion exposure, dense embedding coverage, finite stream-window exactness, regular rational readback, dyadic tolerance accounting, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or completion-facing consumer must read those rows in the displayed route before it can use the local name.*

Lean4 ✓ `BEDC.Derived.LocatedCompletionUp.LocatedCompletionCarrier namecert obligations`

Proof. Project the carrier of Definition 4150.1. The displayed analytic rows are the located metric source M, separated completion boundary S, completion embedding E, stream windows W, regular readback R, dyadic tolerance ledger D, and Real seal A.

Read the source and embedding side first. The $\mathsf{LocatedMetric}^\uparrow$ row supplies located distance data, the $\mathsf{CompletionEmbedding}^\uparrow$ row admits source observations, and the $\mathsf{SeparatedCompletion}^\uparrow$ row supplies the separated comparison boundary named in chapter 1612.

The numerical route is finite. The stream windows W and dyadic ledger D precede the $\mathsf{RegSeqRat}^\uparrow$ readback R, and R precedes the terminal $\mathsf{Real}^\uparrow$ seal A.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same route, while the carrier lists no quotient completion, selected representative, host equality, choice schedule, or ambient completeness coordinate. Hence the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^\uparrow}$ obligations. □

Theorem 4150.3 (LocatedCompletion L10 handoff)**.** *Every* $\mathsf{LocatedCompletion}^\uparrow$ *consumer route factors through the finite L10 chain*

$$\mathsf{LocatedMetric}^\uparrow \longmapsto \mathsf{CompletionEmbedding}^\uparrow \longmapsto \mathsf{SeparatedCompletion}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow, \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{RegS}$$

The handoff is the same route consumed by the located completion space chapter of **??***; it exports no quotient completion, selected completed point, host equality, countable-choice selector, or ambient completeness theorem.*

Proof. Use Definition 4150.2 to expose the full packet route. The located metric row, dense embedding, separated boundary, finite windows, tolerance ledger, readback, and Real seal must all be displayed before the local name is available.

Compare this route with **??**. The located completion space interface consumes the same sibling rows but packages them as a space-facing boundary, while the present chapter packages them as the local located completion object.

No additional completion principle remains. The structural rows H, C, P, N merely transport, replay, source, and name the displayed chain, and the refused quotient, selection, equality, choice, and ambient-completeness objects have no coordinate in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCompletion}^{\uparrow}$ consumer displays a $\mathsf{LocatedMetric}^{\uparrow}$ source row, a $\mathsf{CompletionEmbedding}^{\uparrow}$ dense entry row, a $\mathsf{SeparatedCompletion}^{\uparrow}$ boundary row, $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: locatedmetric, separatedcompletion, completionembedding, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located metric source data, dense entry, separated completion boundary, finite stream windows, dyadic tolerance, regular readback, Real sealing, transport, continuation, provenance, and local naming as one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4150.1, Definition 4150.2, and Definition 4150.3 seed the finite located completion surface over $\mathsf{LocatedMetric}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected completed representatives, host equality of completed points, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, located source exposure, separated completion compatibility, embedding coverage, finite readback exactness, Real seal exactness, transport replay, and non-escape over the same displayed packet.

Constructive story: A $\mathsf{LocatedCompletion}^{\uparrow}$ packet is generated from a located metric source, a dense completion entry, a separated completion boundary, finite stream windows, dyadic tolerances, regular rational readback, a Real seal, and structural NameCert rows.

Chapter 4151

A Concrete Naming Certificate for $\mathsf{SeparatedReflection}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SeparatedReflection}^{\uparrow}$ packet records the boundary where a pseudometric source exposes zero-distance evidence before any consumer treats that evidence as point identity. It starts from the displayed source rows of chapter 1614, routes the zero-distance boundary through the separated classifier of chapter 1411, and hands only that routed surface to the completion consumer of chapter 1506. Its finite readback remains inside the stream windows of chapter 126, the regular rational route of chapter 301, and the terminal real-name seal of chapter 38. The packet therefore names the constructive pseudometric-to-separated boundary without quotient primitives, selected representatives, host equality, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4151.1 (SeparatedReflection carrier)**.** *A* $\mathsf{SeparatedReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (P_0, Z, S, M, R, E, H, C, N)$$

with the following displayed rows:

1. *P_0 is the source* $\mathsf{Pseudometric}^{\uparrow}$ *packet, including point admission, displayed distance witnesses, dyadic tolerance and triangle ledgers, stream windows, regular rational readback, a real-distance seal, and the zero-distance boundary available before separation.*
2. *Z records the zero-distance boundary as evidence read from P_0. It does not identify source point rows and does not introduce a quotient class.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier handoff. It is the first row allowed to turn zero-distance evidence into separated point-classifier evidence.*
4. *M is the metric-completion readback row. It hands the separated classifier surface to* $\mathsf{MetricCompletion}^{\uparrow}$ *only after S has been displayed.*
5. *R is the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window route available to completion and real-readback consumers.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached from the displayed finite and regular rational rows.*
7. *H stores componentwise* hsame *transport for the pseudometric source, zero-distance boundary, separated classifier, completion readback, finite-window route, real seal, replay, provenance, and local naming rows.*

8. *C stores the displayed* Cont *replay from pseudometric inspection through zero-distance evidence, separated classifier handoff, completion readback, finite windows, regular rational readback, and real sealing.*

9. *N is both the* Pkg *provenance over* $\mathsf{Pseudometric}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows and the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$ *row.*

The carrier is a finite reflection-boundary certificate. It has no quotient-completion coordinate, no host-equality coordinate, no selected representative, no countable-choice selector, no trichotomy row, no open-cover compactness principle, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SeparatedReflectionUp.SeparatedReflectionTasteGate single carrier alignment`

Theorem 4151.2 (SeparatedReflection NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *carrier* $P = (P_0, Z, S, M, R, E, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$ *surface consists exactly of the displayed rows for source pseudometric admission, zero-distance boundary exposure, separated carrier handoff, metric-completion readback, StreamName-RegSeqRat-Real sealing, componentwise transport, continuation replay, provenance, and local naming. No obligation may be discharged by a quotient completion, host equality, selected representative, countable-choice selector, trichotomy split, open-cover compactness datum, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge row.*

Proof. Project the carrier of Definition 4151.1. The tuple displays the source pseudometric packet, zero-distance row, separated classifier, completion readback, finite-window and regular-rational route, real seal, transport, replay, provenance, and naming rows.

The obligation order is the order of those displayed rows. The zero-distance evidence is first held in Z, the separated carrier handoff is S, and completion-facing readback is delayed until M reads the routed separated surface.

The finite numerical readback is confined to R and E, while H, C, N provide transport, replay, provenance, and local naming. Since none of the excluded principles is a field of the carrier, none can serve as a local NameCert obligation. □

Theorem 4151.3 (SeparatedReflection zero-distance boundary)**.** *Let* $P = (P_0, Z, S, M, R, E, H, C, N)$ *be an accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *packet. Every consumer that reads zero-distance evidence from the pseudometric source receives that evidence first as the displayed boundary row* Z*. If the consumer asks for point identity, the request must pass through the separated classifier row* S *before any metric, completion, regular-rational, or real-readback row can consume it.*

Proof. Start with the source row P_0. It supplies the pseudometric distance surface and exposes zero-distance evidence without identifying source point histories.

The boundary row Z records exactly that evidence. It is not a point-identity row, so a consumer that needs identity has no route inside P_0 or Z.

The first identity-facing coordinate is S. Therefore every downstream metric, completion, regular-rational, or real-readback consumer receives the zero-distance claim only after the separated classifier handoff has been displayed. □

Theorem 4151.4 (SeparatedReflection completion handoff)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{SeparatedReflection}^{\uparrow}$ *packet reads the route*

$$\mathsf{Pseudometric}^{\uparrow} \to Z \to \mathsf{SeparatedMetric}^{\uparrow} \to \mathsf{MetricCompletion}^{\uparrow} \to \mathsf{StreamName}^{\uparrow} \to \mathsf{RegSeqRat}^{\uparrow} \to \mathsf{Real}^{\uparrow}.$$

Thus completion consumers obtain only the separated carrier handoff and its finite readback; they do not obtain quotient completion, host equality, selected representatives, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the reflection packet.*

Proof. Read the displayed carrier order. The pseudometric source precedes the zero-distance row, the zero-distance row precedes the separated classifier, and the separated classifier precedes the completion readback.

The numerical part of the read is then forced through R and E. Stream windows and regular rational approximants are displayed before the real seal, so completion-facing consumers see a finite readback route rather than a detached real object.

The exclusions are inventory facts. No quotient completion, host-equality proof, selected representative, countable-choice selector, trichotomy split, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem is present in P, so no completion handoff can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedReflection}^{\uparrow}$ packet is generated from a source $\mathsf{Pseudometric}^{\uparrow}$ packet, a displayed zero-distance boundary row, a $\mathsf{SeparatedMetric}^{\uparrow}$ classifier handoff, a $\mathsf{MetricCompletion}^{\uparrow}$ readback row, local StreamNameRegSeqRat-Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4151.1, Definition 4151.2, Definition 4151.3, and Definition 4151.4 seed the finite boundary between pseudometric zero-distance evidence and separated completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, host equality, selected representatives, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, zero-distance boundary exposure, separated carrier handoff, completion readback, finite real sealing, transport replay, provenance visibility, and non-escape for BEDC.Derived.SeparatedReflecti

Chapter 4152

A Concrete Naming Certificate for $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$

Following Picard and Lindelof, local flow data are treated here as a finite constructive package: a metric phase row, a rational time-window row, a local Lipschitz budget, an Euler-style approximation schedule, and a replay ledger that records how each requested window is inspected. The $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ packet names the local existence interface for ODE consumers without importing host flows, selected maximal intervals, ambient completeness, or global continuation. Its local-flow route consumes the Lipschitz budget surface of chapter 3724 together with the map boundary of chapter 410, the phase metric row of chapter 57, the regular time-window readback of chapter 301, and the finite schedule surface of chapter 126.

Definition 4152.1 (Constructive Lipschitz flow carrier)**.** *A* $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, T, L, E, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *phase row containing the point-history carrier, metric readbacks, distance classifier, and local metric* $\mathsf{NameCert}$*.*
2. *T is the rational time-window row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *windows and not through a host interval object.*
3. *L is the local Lipschitz budget row; it records the radius, time-length, and contraction budget inspected by the packet.*
4. *E is the finite approximation schedule, read as a* $\mathsf{StreamName}^{\uparrow}$ *sequence of local Euler windows.*
5. *W is the observation ledger joining metric-radius requests, time windows, Lipschitz-budget checks, and approximation-window readbacks.*
6. *H records componentwise* hsame *transport for M, T, L, E, W and the local certificate rows.*
7. *C records the displayed* Cont *replay from a requested local flow window to its metric, time, Lipschitz, and approximation rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$ *row.*

The classifier compares two accepted carriers only by transporting these displayed rows to a common metric, time-window, Lipschitz-budget, and approximation-schedule surface. It has no coordinate for a selected host solution curve, maximal interval, quotient trajectory, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ConstructiveLipschitzFlowUp.TasteGate.ConstructiveLipschitzFlowTasteGate`
`single carrier alignment`

Theorem 4152.2 (Constructive Lipschitz flow NameCert obligations)**.** *For every accepted* ConstructiveLipschitzFlo *carrier* $F = (M, T, L, E, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$ *surface consists exactly of the carrier admission row, classifier-transport row, local Lipschitz-budget row, finite approximation-schedule row, observation-ledger exactness row, continuation-replay row, package-provenance row, and local naming row. These rows bind* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *as the whole public obligation surface.*

Proof. Project the carrier of Definition 4152.1. The projection displays the phase metric row M, time-window row T, Lipschitz budget L, approximation schedule E, observation ledger W, transport row H, continuation row C, provenance row P, and local name N.

Classifier transport uses H to keep all rows over one metric and one time-window surface. The Lipschitz budget is therefore read with the same metric-radius and rational time data as the approximation schedule.

The ledger exactness clause is exhausted by W. Every local-flow request must pass through the displayed metric, time, budget, and schedule windows before it reaches the Cont replay row.

Finally attach P and N. These rows source and name the displayed packet; they add no host solution curve, selected maximal interval, quotient trajectory, or ambient completeness principle. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4152.3 (Constructive Lipschitz flow local-window boundary)**.** *Every downstream ODE consumer of an accepted* $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ *carrier* $F = (M, T, L, E, W, H, C, P, N)$ *must read local flow data through the displayed metric row* M*, rational time-window row* T*, Lipschitz budget* L*, approximation schedule* E*, observation ledger* W*, componentwise transport* H*, continuation replay* C*, package provenance* P*, and local* NameCert *row* N*. The packet supplies no global flow, no selected maximal interval, no quotient trajectory equality, no closed completeness of the phase space, and no host-level uniqueness theorem.*

Proof. Start from the NameCert obligation surface of Definition 4152.2. It exposes only the metric, time, budget, schedule, ledger, transport, replay, provenance, and naming rows.

A local-window request first fixes its rational time window and its metric radius through T and M. The Lipschitz budget L and approximation schedule E can then be inspected only through the ledger W.

The continuation row C replays this finite inspection route, while H keeps the rows componentwise stable and P, N record provenance and naming. Since no carrier coordinate stores a host flow or maximal interval, the downstream consumer receives only the displayed local-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveLipschitzFlow}^{\uparrow}$ packet is generated from a phase metric row, rational time-window row, local Lipschitz budget, finite approximation schedule, observation ledger, componentwise transport, displayed continuation replay, package provenance, and local naming data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveLipschitzFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4152.1, Definition 4152.2, and Definition 4152.3 define the local flow carrier and seed the ODE-facing local-window route over $\mathsf{Met}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConstructiveLipschitzFlow}\uparrow}$, unformalized)
Bridge status: none
Origin: human
Not claimed: This chapter does not close a Lean target, global flows, selected maximal intervals, quotient trajectories, phase-space completeness, host uniqueness, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, classifier transport, Lipschitz-budget exactness, approximation-schedule replay, and local-window non-escape for the same packet.

Chapter 4153

A Concrete Naming Certificate for $\mathsf{LebesgueDifferentiation}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive measure theory, the $\mathsf{LebesgueDifferentiation}^{\uparrow}$ packet records the finite differentiation-at-a-window surface obtained from measurable local averages, finite Vitali subcover ledgers, and Hardy–Littlewood maximal domination rows. It sits after the measurable-event and integral rows of chapter 92, the finite covering ledger of chapter 2166, the maximal-function domination packet of chapter 3581, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite local differentiation certificate; it is not measure completion, an almost-everywhere quotient, a countable-choice selection, or a host theorem about all measurable functions.

Definition 4153.1 (Lebesgue differentiation carrier)**.** *A* $\mathsf{LebesgueDifferentiation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, V, H_0, A, D, E, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row, exposing only the finite measurable-event and local-measure requests inspected by the packet.*
2. *F is the displayed function-read row whose local averages are tested through the measure row.*
3. *V is the* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *ledger, recording the finite disjoint subfamily, cover cells, and overlap budget used by the differentiation window.*
4. *H_0 is the* $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ *domination row that bounds the exceptional local-average requests before a differentiation verdict is consumed.*
5. *A records the finite local-average rows for the inspected balls or intervals.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare radii, averages, exceptional budgets, and real-name windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the value and error-bound reads.*
8. *T records componentwise* hsame *transport for the measure, function, Vitali, maximal, average, dyadic, real, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route from finite local-average requests through M, F, V, H_0, A, D, E.*

10. *P is the* Pkg *provenance over* LebesgueDifferentiation$^{\uparrow}$, Measure$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, HardyLittlewoodMaximal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{LebesgueDifferentiation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, V, H_0, A, D, E, T, C, P, N$. It has no coordinate for a completed measure algebra, an almost-everywhere quotient, a countable family of selected radii, a selected representative of a function, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LebesgueDifferentiationUp.LebesgueDifferentiationTasteGate single carrier alignment`

Theorem 4153.2 (Lebesgue differentiation NameCert obligations)**.** *For every accepted* LebesgueDifferentiation$^{\uparrow}$ *carrier*

$$L = (M, F, V, H_0, A, D, E, T, C, P, N),$$

the local NameCert$_{\text{LebesgueDifferentiation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, displayed function-read row, finite Vitali-cover ledger, Hardy–Littlewood maximal domination row, local-average rows, dyadic tolerance ledger, real seal, componentwise transport, displayed continuation, package provenance, and local naming.*

Proof. Project the carrier of Definition 4153.1. The rows $M, F, V, H_0, A, D, E, T, C, P, N$ are the whole displayed packet.

The measurable and averaging data are read first. The measure row M admits the inspected event and local-measure requests, while F and A supply the displayed function windows and average rows.

The covering and domination rows then constrain the differentiation request. The Vitali ledger V supplies the finite cover selection surface, and the Hardy–Littlewood row H_0 supplies the maximal domination boundary before the real-valued verdict is exposed.

Finally D, E, T, C, P, N compare, seal, transport, replay, source, and name the same finite route. Since no additional coordinate is present, the NameCert surface exports no measure completion, almost-everywhere quotient, countable-choice selector, selected representative, or ambient Real$^{\uparrow}$ completeness. □

Theorem 4153.3 (Lebesgue differentiation Vitali-maximal handoff)**.** *Every differentiation read accepted by a* LebesgueDifferentiation$^{\uparrow}$ *packet factors through the finite Vitali-cover ledger of chapter 2166 and the maximal-domination surface of chapter 3581 before it reaches the* Measure$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows used by downstream measure and real-name consumers.*

Proof. Start with an accepted carrier $L = (M, F, V, H_0, A, D, E, T, C, P, N)$. A differentiation read reaches the local averages only through the measure and function rows M, F, A.

The finite covering step is then forced through V. It supplies the disjoint subfamily, cover-cell ledger, and overlap budget used by the local differentiation window.

The maximal-function step is forced through H_0. It bounds the exceptional average requests before the dyadic comparison row D and terminal real seal E can be consumed.

Thus the accepted read follows the ordered route through finite Vitali cover, Hardy–Littlewood maximal domination, measure, dyadic tolerance, and real-name rows, with T, C, P, N preserving the route. No countable selection, measure completion, almost-everywhere quotient, selected function representative, or ambient completeness row can be inserted into that handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueDifferentiation}^{\uparrow}$ packet is generated from a MeasureUp source row, displayed function-read rows, a FiniteVitaliCover ledger, HardyLittlewoodMaximal domination, local-average rows, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4153.1, Definition 4153.2, and Definition 4153.3 seed the finite differentiation-window route over MeasureUp, FiniteVitaliCoverUp, HardyLittlewoodMaximalUp, DyadicRatCoreUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueDifferentiation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim measure completion, almost-everywhere quotient equality, countable-choice radius selection, selected function representatives, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Vitali-cover handoff, maximal-domination handoff, dyadic tolerance sealing, real-name nonescape, and local naming for BEDC.Derived.LebesgueDifferentiationUp.

Chapter 4154

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$

The $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ packet realizes the observer-shadow reading of chapter 9463 as finite NameCert data. It consumes the visual-shadow metric surface of chapter 4142, the general metric rows of chapter 57, finite observation windows from chapter 126, regular rational readbacks from chapter 301, and terminal real-name seals from chapter 38. The packet names the finite visual shadow seen by a focused observer route; it is not a global hyperbolic model, a physical observer trajectory, a quotient boundary, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4154.1 (Hyperbolic observer-shadow metric carrier)**.** *A* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, M, W, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *source row, supplying the boundary-shadow visual metric packet.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used when the observer-shadow read is consumed as a metric datum.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened for the visual-shadow and metric reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed shadow distance.*
6. *F is the focused observer-shadow row, recording which observer-facing finite route is being inspected.*
7. *H records componentwise* hsame *transport for the visual-shadow, metric, window, readback, real, focus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the focused observer-shadow request through S, M, W, R, E, F.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, W, R, E, F, H, C, P, N$. It has no coordinate for a completed hyperbolic space, quotient boundary equality, physical observer path, hidden global geometry, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.HyperbolicObserverShadowMetricUp.HyperbolicObserverShadowMetricTasteGate`
`single carrier alignment`

Theorem 4154.2 (Hyperbolic observer-shadow metric NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *carrier*

$$O = (S, M, W, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$ *surface consists exactly of visual-shadow admission, metric comparison admission, finite stream-window exposure, regular-rational readback, terminal real sealing, focused observer-shadow routing, componentwise transport, displayed replay, package provenance, and local naming through* $S, M, W, R, E, F, H, C, P, N$.

Proof. Project the carrier of Definition 4154.1. The listed rows are the whole observer-shadow metric packet.

The visual-shadow row S is read before the metric row M. This keeps the metric comparison tied to the finite visual-shadow carrier rather than to a host hyperbolic model.

The window, readback, and real-seal rows W, R, E then expose the numerical shadow distance through the same L10 real-name route. The focused row F records the observer-facing route being inspected.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert surface exports no completed hyperbolic space, quotient boundary equality, physical observer trajectory, hidden global geometry, or ambient real completeness. □

Theorem 4154.3 (Hyperbolic observer-shadow L10 handoff)**.** *Every* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *consumer that reads a shadow distance factors through* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the focused observer-shadow route can be consumed as a finite NameCert datum.*

Proof. Begin with an accepted carrier $O = (S, M, W, R, E, F, H, C, P, N)$. The source visual-shadow metric row S supplies the displayed boundary-shadow packet.

The metric comparison row M is then consumed through the finite metric surface, and the L10 numeric read follows the window, readback, and real-seal route W, R, E.

Only after these rows are displayed does the focused row F expose the observer-shadow route. Transport, continuation, provenance, and local naming are supplied by H, C, P, N, so the handoff remains finite and cannot pass through a host global geometry or quotient boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ packet is generated from a HyperbolicVisualShadowMetric row, a MetricUp comparison row, StreamName windows, RegSeqRat readback, a RealUp seal, a focused observer-shadow row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4154.1, Definition 4154.2, and Definition 4154.3 seed the finite observer-shadow metric route over visual-shadow, metric, stream-window, regular-rational, and real-name rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed hyperbolic geometry, quotient boundary equality, physical observer trajectories, hidden global geometry, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-shadow source admission, metric comparison, stream-window readback, real-name sealing, focused observer routing, and local naming for BEDC.Derived.HyperbolicObserverShadowMet

The RegularDoubleCauchy$^{\uparrow}$ route reads the rectangular StreamName$^{\uparrow}$/RegSeqRat$^{\uparrow}$/DyadicRatCore$^{\uparrow}$ surface as a sibling of chapter 460 before the RealCauchyCompletion$^{\uparrow}$ and Real$^{\uparrow}$ seals are available.

Chapter 4155

A Concrete Naming Certificate for $\mathsf{RegularDoubleCauchy}^{\uparrow}$

Following Pringsheim's regular convergence of double sequences, Moore–Osgood interchange discipline, and Bishop–Bridges constructive Cauchy completion, the $\mathsf{RegularDoubleCauchy}^{\uparrow}$ packet records a two-parameter regular Cauchy certificate that does not collapse immediately to a diagonal route. It refines the diagonal handoff of chapter 460 by keeping row and column stability visible before the downstream completion consumer reads the terminal seal. The packet sits over the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the completion-facing route of chapter 360. It provides Moore–Osgood and completion consumers with a finite rectangular regularity surface without asserting host double-sequence convergence, quotient arrays, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 4155.1 (RegularDoubleCauchy carrier)**.** *A* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, R, D, T, W, H, C, P, N)$$

with the following visible rows: S *is the* $\mathsf{StreamName}^{\uparrow}$ *family schedule for finite rectangular windows;* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback attached to every admitted row and column face;* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *rectangular tolerance ledger;* T *is the diagonal cofinal schedule read only after the rectangular ledger has been displayed;* W *is the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff;* H *records componentwise* hsame *transport;* C *records displayed* Cont *replay;* P *is the* Pkg *provenance over* $\mathsf{RegularDoubleCauchy}^{\uparrow}$*,* $\mathsf{CauchyDoubleSequence}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*; and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *row.*

The carrier is rectangular before it is diagonal. It contains no coordinate for a host double limit, quotient equality of arrays, a selected limiting value, countable-choice scheduling, or ambient real completeness.

Proof. Read the packet as a displayed BHist row. The entries S, R, D expose the finite rectangular data before the diagonal schedule T is available, so the consumer cannot begin with a collapsed single sequence.

The completion-facing row W is downstream from the same visible schedule, readback, and tolerance ledger. Transport and replay are exhausted by H and C, while provenance and naming are exhausted by P and N.

Since the listed tuple has no further coordinate, the excluded host convergence, quotient-array, selected-limit, choice, and ambient-completeness data cannot be read from the carrier. □

Theorem 4155.2 (RegularDoubleCauchy row-column stability)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier* $A = (S, R, D, T, W, H, C, P, N)$*, any two finite reads that agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *rectangular window schedule and on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger have the same row and column* $\mathsf{RegSeqRat}^{\uparrow}$ *readback under the componentwise* hsame *transport recorded in* H*. The stability is local to the finite rows* S, R, D, H, C, P, N *and does not identify quotient arrays or assert a host double-sequence limit.*

Proof. Project the schedule and ledger rows. The rectangular window family is supplied by S, and the tolerance comparison used for every row and column face is supplied by D.

Apply the readback row R to the same displayed windows and ledger entries. Componentwise hsame transport in H preserves the row face and the column face before any completion handoff is inspected.

The replay row C repeats this route for a consumer request, and P, N keep the request inside the local packet. No quotient-array or host-limit coordinate is present, so the stability conclusion is exactly the visible row-column readback. □

Theorem 4155.3 (RegularDoubleCauchy diagonal readback)**.** *Every diagonal read of an accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier factors through the rectangular route*

$$S, D \longmapsto R \longmapsto T \longmapsto W.$$

Thus a $\mathsf{CauchyDoubleSequence}^{\uparrow}$*,* $\mathsf{MooreOsgood}^{\uparrow}$*,* $\mathsf{RealCauchyCompletion}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *consumer sees the diagonal cofinal schedule only after the StreamName windows, RegSeqRat row-column readback, and DyadicRatCore rectangular tolerance ledger have been displayed.*

Proof. Start with the rectangular rows. A diagonal request must first open the finite windows in S and account for the requested precision in D.

The row-column readback R is then the only route from those rectangular windows to a regular rational name. The diagonal schedule T is read after this regularity data, not before it.

Finally the completion-facing handoff W consumes the scheduled readback. Therefore every cited downstream consumer receives the diagonal route as a readback of the rectangular certificate, not as an independent host convergence assertion. □

Theorem 4155.4 (RegularDoubleCauchy rectangular modulus extraction)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier* $A = (S, R, D, T, W, H, C, P, N)$*, the rows* S, D, R, H, C, P, N *determine a finite rectangular modulus surface before* T *or* W *may be consumed. Every* $\mathsf{RealCauchyCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read of* A *factors through that surface, so the diagonal schedule and terminal seal are downstream of the displayed StreamName windows, DyadicRatCore tolerance ledger, row-column RegSeqRat readback, componentwise transport, replay, provenance, and local naming rows.*

Proof. Project the carrier rows S, D, R, H, C, P, N. The StreamName row supplies the rectangular windows, the DyadicRatCore row supplies the tolerance ledger, and the RegSeqRat row reads row and column faces against those displayed budgets.

Apply Definition 4155.2. It binds finite rectangular windows and dyadic tolerances to the row-column readback under the same componentwise hsame transport, with C, P, N replaying and naming the route.

Only after this rectangular surface is displayed may Definition 4155.3 read the diagonal schedule T and the completion-facing handoff W. The extracted surface contains no selected double limit,

quotient array, choice schedule, or ambient Real-completeness row, so every RealCauchyCompletion or Real read factors through the finite modulus surface first. □

Theorem 4155.5 (RegularDoubleCauchy ledger exactness)**.** *In an accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier, the dyadic tolerance ledger D is the unique finite row from which a completion-facing consumer may justify rectangular Cauchy bounds before reading the diagonal schedule T or the seal handoff W. Every precision request must pass through S, D, R with the same H, C, P, N provenance, and no selected limit, countable-choice schedule, quotient array, or ambient completeness row can replace that ledger.*

Proof. Project the accepted carrier. The only tolerance row is D, and it is attached to the same finite window schedule S that supplies the rectangular observations.

Readback through R uses that ledger before any diagonal schedule or completion-facing row appears. The continuation row C records the same order for each accepted request, while H transports the ledger componentwise.

The remaining rows P, N only source and name the route. Since there is no second tolerance, selected-limit, quotient, choice, or completeness coordinate, ledger exactness is forced by the tuple inventory. □

Theorem 4155.6 (RegularDoubleCauchy NameCert surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, rectangular window scheduling, row-column* $\mathsf{RegSeqRat}^{\uparrow}$ *stability,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger exactness, diagonal cofinal readback,* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. It is the non-diagonal regularity certificate consumed by CauchyDoubleSequence and Moore–Osgood routes before their diagonal or iterated-limit handoffs, and it exports no host double-sequence convergence theorem, quotient-array classifier, selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Combine the carrier theorem with row-column stability. These statements expose the rectangular windows, regular readback faces, tolerance ledger, transport, replay, provenance, and local name.

The diagonal readback theorem supplies the only passage from the rectangular surface to a completion-facing diagonal route. Ledger exactness supplies the only precision evidence available on that passage.

Thus the NameCert surface is the finite union of the listed rows and obligations. Downstream CauchyDoubleSequence and Moore–Osgood consumers may use this surface as a regularity certificate, but no excluded host or choice principle has a row from which it could be exported. □

Theorem 4155.7 (RegularDoubleCauchy obligation scope)**.** *For every accepted* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *carrier, the five obligation rows of carrier admission, row-column stability, diagonal readback, ledger exactness, and local NameCert surface are scoped exactly by the displayed* BHist *packet together with rectangular* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the diagonal schedule, the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *row.*

No quotient array, selected limit, countable-choice schedule, host double-limit theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle may be used as an alternate scope for any of these obligations.*

Proof. Begin with Definition 4155.1. It fixes the only visible carrier rows as the BHist packet $S, R, D, T, W, H, C, P, N$, with rectangular StreamName windows, RegSeqRat row-column readback, the dyadic tolerance ledger, the diagonal schedule, the RealCauchyCompletion and Real seal handoff, transport, replay, provenance, and local naming.

Apply Definition 4155.2 and Definition 4155.5. These two results bind row-column comparison and precision evidence to the same displayed rectangular windows, readback faces, tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Next apply Definition 4155.3. It forces every diagonal or completion-facing read to pass through the rectangular route before the $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff can be consumed.

Finally, Definition 4155.6 identifies the local NameCert surface with exactly these rows. Since the carrier and surface theorems expose no quotient array, selected limit, countable-choice schedule, host double limit, or ambient Real-completeness row, none of those objects can provide an alternate obligation scope. □

Theorem 4155.8 (RegularDoubleCauchy obligation-closure package)**.** *The carrier of Definition 4155.1, together with row-column stability, diagonal readback, rectangular modulus extraction, ledger exactness, the local NameCert surface, and the obligation scope theorem, packages the five* $\mathsf{RegularDoubleCauchy}^{\uparrow}$ *obligation rows into one completion-facing surface. Every accepted consumer must pass through rectangular* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *row-column readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the diagonal schedule, the* $\mathsf{RealCauchyCompletion}^{\uparrow}/\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$ *before reading a diagonal or completion endpoint.*

Proof. Begin with Definition 4155.1. It exposes the finite rectangular packet $S, R, D, T, W, H, C, P, N$ and excludes host double limits, quotient arrays, selected limits, countable-choice schedules, and ambient Real completeness.

Use Definition 4155.2, Definition 4155.4, and Definition 4155.5. These theorems bind rectangular StreamName windows, row-column RegSeqRat readback, and the DyadicRatCore tolerance ledger into the same finite modulus surface.

Next apply Definition 4155.3. It places the diagonal schedule and the RealCauchyCompletion or Real seal strictly downstream of the rectangular surface, so a completion-facing read cannot start from the terminal handoff.

Finally combine Definition 4155.6 with Definition 4155.7. The local NameCert rows and scope theorem identify exactly the structural transport, replay, provenance, and naming data that travel with the five obligation rows, leaving no alternate obligation coordinate outside the displayed packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularDoubleCauchy}^{\uparrow}$ read factors through rectangular StreamName windows, RegSeqRat row-column readback, a DyadicRatCore tolerance ledger, a diagonal cofinal schedule, RealCauchyCompletion and Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchydoublesequence, mooreosgood, streamname, regseqrat, dyadicratcore, realcauchycompletion, real: $\mathsf{RegularDoubleCauchy}^{\uparrow}$ is not bijective to any listed sibling because it keeps rectangular row-column regularity visible before the diagonal or iterated-limit consumer is allowed to read a completion-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularDoubleCauchy}^{\uparrow}$ packet is generated from finite rectangular StreamName windows, RegSeqRat row-column readback, a DyadicRatCore tolerance ledger, a diagonal cofinal schedule, a RealCauchyCompletion

and Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 4155.1, Definition 4155.2, Definition 4155.3, Definition 4155.5, and Definition 4155.6 seed the rectangular two-parameter regularity surface for downstream diagonal and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularDoubleCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host double-sequence convergence, quotient arrays, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, row-column stability, diagonal readback, ledger exactness, and local NameCert surface proofs for BEDC.Derived.RegularDoubleCauchyUp.

The RegularDoubleCauchy$^{\uparrow}$ route reads paired Cauchy windows through the finite stream surface of chapter 126 and the regular rational readback of chapter 301 before any Real-facing handoff. This sibling dependency keeps the double-window packet tied to displayed StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows, with the downstream Real$^{\uparrow}$ seal supplied only through chapter 38. The RegularDoubleCauchy$^{\uparrow}$ route consumes the diagonal handoff of chapter 460 only after the rectangular StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ row-column readback of chapter 301, the DyadicRatCore$^{\uparrow}$ tolerance ledger of chapter 291, and the RealCauchyCompletion$^{\uparrow}$/Real$^{\uparrow}$ seal of chapter 360 and chapter 38 are displayed. The RegularDoubleCauchy$^{\uparrow}$ route reads CauchyAcceleration$^{\uparrow}$ and CauchySequenceSpace$^{\uparrow}$ before a diagonal or rectangular Real-facing handoff is exported. The acceleration row of chapter 4222 supplies the finite speedup ledger, while the sequence-space surface of chapter 406 keeps the downstream Cauchy-name consumer tied to the same RegSeqRat$^{\uparrow}$ readback of chapter 301 and the terminal Real$^{\uparrow}$ seal of chapter 38. DyadicIntervalNewtonBisection$^{\uparrow}$ reads interval Newton narrowing through chapter 1570, dyadic interval endpoints through chapter 306, bisection fallback through chapter 1489, and Real-facing readback through chapter 301.

Chapter 4156

A Concrete Naming Certificate for DyadicIntervalNewtonBisection$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the DyadicIntervalNewtonBisection$^{\uparrow}$ packet records the finite branch where interval Newton narrowing is interleaved with dyadic bisection before a regular Cauchy or Real-facing consumer reads the result. It refines the dyadic endpoint surface of chapter 306, the Newton enclosure certificate of chapter 1570, the bisection step of chapter 1489, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives root-isolation and completion routes a finite certificate without importing quotient reals, selected roots, trichotomy, countable choice, or ambient completeness.

Definition 4156.1 (Dyadic interval Newton bisection carrier)**.** *A* DyadicIntervalNewtonBisection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (D, I, H, N, B, R, E, T, C, P, L)$$

with the following displayed rows:

1. *D is the* DyadicInterval$^{\uparrow}$ *endpoint row for the parent cell and the inspected child cells.*
2. *I is the* IntervalNewton$^{\uparrow}$ *correction row, including the displayed enclosure and containment ledger.*
3. *H is the* IntervalHalving$^{\uparrow}$ *bisection row used when the Newton enclosure does not yet supply the retained child cell.*
4. *N records the finite narrowing decision: accept the Newton child, accept the bisection child, or replay both rows before the next finite inspection.*
5. *B is the dyadic diameter and containment ledger attached to the retained child interval.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row produced only after D, I, H, N, B have been displayed.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row exposed to completion consumers.*
8. *T records componentwise* hsame *transport for the dyadic interval, Newton row, halving row, narrowing decision, diameter ledger, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the same rows.*
10. *P is the* Pkg *provenance over* DyadicIntervalNewtonBisection$^{\uparrow}$, DyadicInterval$^{\uparrow}$, IntervalNewton$^{\uparrow}$, IntervalHalving$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. L is the local $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $D, I, H, N, B, R, E, T, C, P, L$. It has no coordinate for a selected zero, quotient real equality, host trichotomy, completed branch choice, or ambient completeness theorem.

Theorem 4156.2 (Dyadic interval Newton bisection NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *carrier $Z = (D, I, H, N, B, R, E, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBise}}$ *surface consists exactly of carrier admission, dyadic endpoint exposure, interval Newton correction exposure, interval-halving fallback exposure, retained-child narrowing, diameter containment, regular rational readback, terminal Real sealing, componentwise transport, displayed replay, package provenance, and local naming for the carrier of Definition 4156.1.*

Proof. Carrier admission is projection from Definition 4156.1. The dyadic row D, Newton row I, halving row H, narrowing row N, and diameter ledger B are the only rows that expose interval refinement data.

The readback rows are downstream of that finite refinement. Row R can read a regular rational name only after the retained dyadic child and its containment ledger are displayed, and row E seals only that readback route.

Transport, replay, provenance, and local naming are exactly T, C, P, L. Since the carrier has no selected-root, quotient-real, trichotomy, completed-branch, or completeness coordinate, those objects cannot become local NameCert obligations. □

Theorem 4156.3 (Dyadic interval Newton bisection regular Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$*, root-isolation, or completion-facing consumer of an accepted* $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{IntervalNewton}^{\uparrow} \longmapsto \mathsf{IntervalHalving}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff supplies retained dyadic child data, Newton or bisection narrowing, a diameter ledger, regular rational readback, and a terminal Real seal. It supplies no quotient real, selected root, trichotomy proof, countable-choice branch, or ambient completeness principle.

Proof. Project the accepted carrier $Z = (D, I, H, N, B, R, E, T, C, P, L)$. The rows D, I, H, N, B expose the entire finite narrowing branch before any regular Cauchy consumer can inspect the packet.

Read R through the retained child and diameter ledger. The regular rational handoff is therefore attached to the displayed dyadic interval, Newton enclosure, and bisection fallback rows, not to an external root selector.

Finally read the terminal seal E through T, C, P, L. These structural rows preserve the same finite route under transport, replay, provenance, and local naming. No stronger principle is present among the carrier coordinates, so every completion-facing read remains the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}$ packet is generated from a DyadicInterval endpoint row, an IntervalNewton correction row, an IntervalHalving fallback row, a retained-child narrowing decision, a dyadic diameter ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNewtonBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4156.1, Definition 4156.2, and Definition 4156.3 seed the finite interval Newton and dyadic bisection route over DyadicIntervalUp, IntervalNewtonUp, IntervalHalvingUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{DyadicIntervalNewtonBisection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient reals, selected roots, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, Newton correction exposure, halving fallback exposure, retained-child narrowing, diameter containment, regular readback, Real-seal nonescape, transport, replay, provenance, and local NameCert field faithfulness for BEDC.Derived.DyadicIntervalNewtonBisectionUp.

The KadecKleeProperty$^{\uparrow}$ route reads weak convergence through chapter 3597, norm convergence through chapter 89, and its Banach/Hilbert source through chapter 90 and chapter 91 before the Radon–Riesz handoff is exposed.

Chapter 4157

A Concrete Naming Certificate for $\mathsf{KadecKleeProperty}^{\uparrow}$

Following Kadec and Klee on the Radon–Riesz property, the $\mathsf{KadecKleeProperty}^{\uparrow}$ packet records the finite BEDC surface where weak convergence plus convergence of norms is accepted as a strong convergence handoff. It sits between the normed and complete linear analysis packets of chapter 89, chapter 90, and chapter 91, reads the weak-convergence boundary of chapter 3597, and keeps every scalar comparison at the real-name seal of chapter 38. The carrier consumes the $\mathsf{Norm}^{\uparrow}$ norm-readback row from chapter 89 and, when present, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ or $\mathsf{Hilbert}^{\uparrow}$ source row before exposing weak-plus-norm convergence as a strong-convergence row. The chapter names only the displayed Radon–Riesz handoff route and does not import Hahn–Banach separation, choice-selected weak limits, quotient vector equality, or ambient completeness.

Definition 4157.1 (Kadec–Klee property carrier)**.** *A* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, H, N, W, Q, S, T, C, P, L)$$

with the following displayed rows:

1. *B is the ambient* $\mathsf{Banach}^{\uparrow}$ *row, exposing the vector-name surface used by the sequence and limit entries;*
2. *H is the optional* $\mathsf{Hilbert}^{\uparrow}$ *refinement row, used only when the consumer already works over an inner-product source;*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row recording the norm readback and its* $\mathsf{Real}^{\uparrow}$ *comparison seal;*
4. *W is the* $\mathsf{WeakTopology}^{\uparrow}$ *row that displays weak convergence of the named sequence to the named candidate limit;*
5. *Q is the norm-convergence row, whose scalar values are read through* $\mathsf{Real}^{\uparrow}$ *names rather than through host real equality;*
6. *S is the strong-convergence handoff row asserting that the same displayed sequence and candidate limit may be consumed as norm convergence after W and Q have both been read;*
7. *T records componentwise* hsame *transport for B, H, N, W, Q, S and for the local certificate rows;*
8. *C records the displayed* Cont *replay from weak observations and norm-value observations to the strong-convergence handoff;*

9. *P is the* Pkg *provenance over* $\mathsf{KadecKleeProperty}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

10. *L is the local* $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for Hahn–Banach separation, a selected weakly convergent subnet, quotient equality of vectors, a detached norm limit, or a completeness principle beyond the displayed source rows.

Theorem 4157.2 (Kadec–Klee NameCert obligations)**.** *For an accepted* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier*

$$K = (B, H, N, W, Q, S, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$ *surface consists exactly of carrier admission for the Banach row, optional Hilbert refinement, norm row, weak-topology row, norm-convergence row, strong-convergence handoff, componentwise transport, continuation replay, provenance, and local naming; classifier stability under the same displayed rows; exactness of the* $\mathsf{Real}^{\uparrow}$ *norm-value comparisons; visibility of the weak convergence row before any strong handoff is read; and refusal of Hahn–Banach separation, choice-selected weak limits, quotient vector equality, detached scalar limits, and ambient completeness.*

Proof. Project the carrier of Definition 4157.1. The projection exposes exactly the analysis rows B, H, N, W, Q, S and the structural rows T, C, P, L.

The weak and norm routes are separate until both have been displayed. The weak route reads W over the same named vector sequence and candidate limit, while the norm route reads Q through the $\mathsf{Norm}^{\uparrow}$ row N and the $\mathsf{Real}^{\uparrow}$ scalar seal.

Now replay the handoff through C and transport it through T. The strong-convergence row S is available only downstream from W and Q, and the provenance and naming rows attach no additional selection, separation, quotient equality, detached limit, or completeness data. □

Theorem 4157.3 (Kadec–Klee weak-strong handoff)**.** *Let $K = (B, H, N, W, Q, S, T, C, P, L)$ be an accepted* $\mathsf{KadecKleeProperty}^{\uparrow}$ *carrier. Any consumer that reads the strong-convergence handoff S must first read the weak-convergence row W, the norm row N, the norm-convergence row Q, the shared* $\mathsf{Real}^{\uparrow}$ *scalar seal, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and the local* NameCert *row L. Thus the Radon–Riesz handoff is a finite BEDC route over* $\mathsf{Banach}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$, *not a consequence of Hahn–Banach separation, choice-selected weak limits, quotient vector equality, or ambient completeness.*

Proof. Start with a request for S. Carrier admission gives the only available predecessors: weak convergence through W, norm readback through N, scalar convergence through Q, and the structural rows T, C, P, L.

Use the NameCert obligation surface of Definition 4157.2. It forces the weak observation and the norm-value comparison to share the same displayed sequence, candidate limit, transport row, and real-name scalar boundary before strong convergence can be consumed.

Because no other coordinate exists in the carrier, the consumer cannot replace the handoff by a separation theorem, a selected weak limit, quotient vector equality, a hidden scalar limit, or an ambient completeness argument. The handoff is exactly the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KadecKleeProperty}^{\uparrow}$ packet is generated from Banach, optional Hilbert, Norm, WeakTopology, Real scalar, weak-convergence, norm-convergence, strong-handoff, transport, replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KadecKleeProperty}^{\uparrow}}$, seed)

Scope closed: Definition 4157.1, Definition 4157.2, and Definition 4157.3 seed the Radon–Riesz route from weak convergence and norm convergence to strong convergence over displayed Banach, Hilbert, Norm, WeakTopology, and Real rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KadecKleeProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice-selected weak limits, quotient vector equality, detached scalar limits, ambient completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, weak-row visibility, norm-scalar exactness, strong-handoff routing, transport replay, provenance, and consumer refusal for BEDC.Derived.KadecKleePropertyUp.

The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ hub routes its uniform tail-budget packet through chapter 126, chapter 301, chapter 291, chapter 38, and chapter 292 before the child files expose the local carrier and dependency routes.

Chapter 4158

A Concrete Naming Certificate for $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$

Following Bishop and Bridges, a regular Cauchy real name is useful to Real and Completion consumers only when its uniform tail budget is displayed before any terminal real seal is consumed. The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ packet records that budget as finite BEDC data over the stream-window surface of chapter 126, the dyadic observation ledger of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. It names the explicit uniform-modulus surface needed by regular Cauchy tails; it is not a quotient of sequences, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4158.1 (Regular Cauchy uniform modulus carrier)**.** *A* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, M, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, carrying the finite windows opened on the source name before any regular readback is inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, containing the dyadic tolerance requests and arithmetic comparisons used by the tail budget.*
3. *M is the displayed regular modulus row; for each requested dyadic tolerance it records the finite index bound after which the regular Cauchy readbacks are stable inside that tolerance.*
4. *T is the finite tail ledger joining stream windows, dyadic tolerances, regular readbacks, and the modulus row.*
5. *H records componentwise* hsame *transport for S, D, M, T and the structural certificate rows.*
6. *C records displayed* Cont *replay from a tail-stability request through the same stream, dyadic, modulus, and tail-ledger rows.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, D, M, T, H, C, P, N. It has no coordinate for quotient stream equality, selected limit witnesses, host real equality, countable choice, or ambient completeness.

Theorem 4158.2 (Regular Cauchy uniform modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier* $U = (S, D, M, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, stream-window exposure, dyadic tolerance accounting, displayed regular modulus admission, finite tail-ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that uses a regular tail budget must read these rows before it can consume the terminal real-name seal.*

Proof. Project the carrier of Definition 4158.1. The projection displays the stream row S, dyadic observation row D, regular modulus row M, finite tail ledger T, transport row H, continuation row C, provenance row P, and local name N.

The numerical obligations are exhausted by S, D, M, T. Stream windows are opened in S, tolerance comparisons are read in D, the index bound is displayed in M, and the ledger T records the finite route joining those rows to regular rational readbacks.

The structural obligations are exhausted by H, C, P, N. They transport, replay, source, and name the same displayed route, while the carrier lists no quotient stream, selected limit, host equality, choice schedule, or ambient completeness coordinate. Hence the listed rows are exactly the local $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformModulus}^{\uparrow}}$ obligations. □

Theorem 4158.3 (Regular Cauchy uniform tail stability)**.** *Let* $U = (S, D, M, T, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Every tail-stability request for the associated regular Cauchy readback factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyUniformModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Na}$$

After the modulus row M has supplied the displayed index bound, the tail ledger T is the only admissible source of stability data for the requested dyadic tolerance. No consumer may replace that route by a host convergence theorem, a selected Cauchy limit, quotient equality of streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use Definition 4158.2. A tail-stability request first opens the stream row S and the dyadic observation row D for the requested tolerance.

The modulus row M then supplies the finite index bound. The ledger T binds that bound to the same stream windows and dyadic comparisons, so the regular rational readback can only be consumed through the displayed route.

Finally H, C, P, N transport, replay, source, and name the packet. Since no carrier coordinate stores a selected limit, quotient stream class, host real equality, choice schedule, or ambient completeness principle, the displayed tail ledger is the complete stability surface. □

Theorem 4158.4 (Regular Cauchy uniform modulus tail-budget exactness)**.** *Let* $U = (S, D, M, T, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Before any tail-stability or completion-facing consumer reads the modulus row, the packet exposes the same finite dyadic budget in D, the same* $\mathsf{StreamName}^{\uparrow}$ *window in S, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary carried by T, and the same* $\mathsf{Real}^{\uparrow}$ *seal route recorded by H, C, P, N. Thus the tail budget is exact as a displayed BEDC row: no later consumer may widen it into a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 4158.1. The rows S, D, M, T expose the stream window, dyadic tolerance ledger, displayed modulus, and finite tail ledger before the structural rows are consulted.

Use Definition 4158.2. Its obligation list requires every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to read the stream-window exposure, dyadic tolerance accounting, modulus-row admission, and tail-ledger exactness before the terminal real-name seal is consumed.

Apply Definition 4158.3. The tail-stability route factors through the same S, D, M, T rows, so a completion-facing read cannot replace the displayed budget by a different modulus surface.

Finally inspect the carrier inventory. There is no quotient stream row, selected limit row, host real equality row, countable-choice row, or ambient completeness row, and H, C, P, N only transport, replay, source, and name the already displayed packet. Hence the exposed tail budget is exact for every downstream read. □

Theorem 4158.5 (Regular Cauchy uniform modulus tail composition)**.** *Let $U = (S, D, M, T, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Suppose its* $\mathsf{StreamName}^{\uparrow}$ *windows replay through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows in D and through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary carried by T. Then every admitted unary precision row composes to the terminal* $\mathsf{Real}^{\uparrow}$ *seal without selecting a limit: the modulus tail bound in M supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *closeness witness consumed by the seal route recorded in H, C, P, N.*

Proof. Use Definition 4158.2. It requires every regular rational or real consumer to expose stream-window, dyadic tolerance, modulus-row, and tail-ledger data before the terminal seal is read.

Apply Definition 4158.3. The admitted precision row first opens the $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the modulus row M supplies the finite tail bound used by the regular readback.

Now apply Definition 4158.4. The same displayed budget is preserved through S, D, M, T, so the $\mathsf{RegSeqRat}^{\uparrow}$ closeness witness consumed by the terminal $\mathsf{Real}^{\uparrow}$ route is the witness supplied by the accepted packet.

The sibling chapters chapter 126, chapter 291, chapter 301, chapter 38, and chapter 292 provide the named row surfaces. Since the carrier has no selected-limit coordinate, the composition remains a finite handoff rather than a limit-selection principle. □

Theorem 4158.6 (Regular Cauchy uniform modulus dyadic locator handoff)**.** *Let $U = (S, D, M, T, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carrier. Every Real- or Completion-facing consumer receives the dyadic tolerance row D before it may consume the regular modulus row M, the finite tail ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the handoff from dyadic locator data to regular tail data is a displayed BEDC boundary; it exports no quotient stream, selected limit, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 4158.2. Its obligation surface places dyadic tolerance accounting in D before displayed regular modulus admission in M and finite tail-ledger exactness in T.

Apply Definition 4158.3. A tail-stability request opens the $\mathsf{StreamName}^{\uparrow}$ row S and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D before the modulus bound and regular rational readback are available.

Now use Definition 4158.4. The exposed budget remains the same displayed dyadic row throughout the completion-facing route, so the terminal $\mathsf{Real}^{\uparrow}$ seal can only be read after D, M, T have been consumed in that order.

Finally inspect the carrier inventory. The rows H, C, P, N transport, replay, source, and name the same handoff, while the carrier has no coordinate for quotient streams, selected limits, countable choice, host equality, or ambient completeness. □

Definition 4158.7 (Located real comparison aperture carrier)**.** *A located real comparison aperture over an accepted* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *packet is a finite* BHist *packet*

$$A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$$

whose rows refine the carrier of Definition 4158.1. The rows S, D, M, T, H, C, P, N *are the stream-window, dyadic-tolerance, uniform-modulus, finite tail-ledger, transport, replay, provenance, and local naming rows already displayed by the uniform-modulus packet. The row* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary exposed by the tail ledger, and* A *is a* $\mathsf{LocatedRealComparison}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$*-facing comparison boundary. Any route to* A *must expose the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerances,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and uniform modulus before any comparison read is consumed.*

Theorem 4158.8 (Located real comparison aperture NameCert obligations)**.** *For every accepted located real comparison aperture* $A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$*, the local comparison-facing obligation surface consists of carrier admission for the displayed aperture, stream-window exposure through* S*, dyadic tolerance accounting through* D*, uniform-modulus admission through* M*, tail-ledger exactness through* T*, readback boundary exposure through* R*, comparison-boundary exposure through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the aperture carrier of Definition 4158.7. Its coordinates display exactly the rows named in the statement, with R and A added as the readback and comparison-facing boundary rows over the existing uniform-modulus packet.

Apply Definition 4158.3. Every tail-stability request first opens the stream window and dyadic tolerance rows, then reads the displayed modulus and the finite tail ledger.

Apply Definition 4158.4. The budget exposed to the comparison boundary is the same displayed S, D, M, T route, so R and A cannot introduce quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4158.9 (Located real comparison aperture gap route)**.** *Every positive-gap, overlap, or comparison-read consumer of an accepted aperture* $A_{\mathrm{loc}} = (S, D, M, T, R, A, H, C, P, N)$ *factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{RegularCauchyUniformModulus}^{\uparrow},\ \mathsf{LocatedRealComparison}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the comparison boundary A *is visible only after the stream windows* S*, dyadic tolerances* D*, uniform modulus* M*, tail ledger* T*, and readback boundary* R *have been displayed.*

Proof. Use Definition 4158.6. It places the dyadic locator row before the uniform modulus, finite tail ledger, regular readback route, and terminal real seal.

Now apply Definition 4158.8. The comparison boundary row A is a coordinate of the aperture, but its obligation surface requires the preceding S, D, M, T, R rows to be exposed before A is read.

Finally use the sibling comparison surface of chapter 1730. The gap or overlap consumer receives a finite located-comparison route over the already displayed regular-Cauchy readback, not a host comparison or an ambient completeness principle. □

Theorem 4158.10 (Regular Cauchy uniform modulus lattice crosslink)**.** *Every accepted* RegularCauchyUniformMod *carrier presents the same*

$$S, D, M, T, H, C, P, N$$

route needed by downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *completion-facing reads. The crosslink keeps stream windows, dyadic budgets, regular readback, uniform tail ledgers, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming visible as one L10 lattice route; it exports no quotient stream equality, selected limit, host real equality, countable-choice schedule, or ambient completeness row.*

Proof. Project the carrier of Definition 4158.1. The rows S, D, M, T expose the stream window, dyadic tolerance ledger, displayed modulus, and finite tail ledger, while H, C, P, N transport, replay, source, and name that same packet.

Apply Definition 4158.2. Its obligation surface requires every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer to pass through the displayed stream-window, dyadic tolerance, modulus, and tail-ledger rows before the terminal seal is consumed.

Apply Definition 4158.4. The tail budget exposed to a completion-facing read is the same displayed BEDC row throughout the route, so it cannot be replaced by a quotient stream, selected Cauchy limit, host real equality, choice schedule, or ambient completeness theorem.

The sibling completion-facing window route is the one named in chapter 406. Therefore the accepted carrier supplies exactly the S, D, M, T, H, C, P, N lattice crosslink used by $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchySequenceSpace}^{\uparrow}$ consumers.

The displayed modulus row is read through the sibling certificate of chapter 292 before tail-budget replay reaches the $\mathsf{RegSeqRat}^{\uparrow}$ readback boundary and the terminal $\mathsf{Real}^{\uparrow}$ seal. □

Theorem 4158.11 (Regular Cauchy uniform modulus window refinement)**.** *Let*

$$U_0 = (S, D, M_0, T_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad U_1 = (S, D, M_1, T_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *carriers sharing the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, and displayed* Cont *replay route. If the unary precision displayed by* U_1 *refines the unary precision displayed by* U_0*, then the later tail-budget route factors through the earlier uniform-modulus ledger and cannot widen the exported* $\mathsf{Real}^{\uparrow}$ *handoff.*

Proof. Project both carriers using Definition 4158.1. The shared rows S, D and shared replay route in C_0, C_1 place both packets on the same stream-window and dyadic-tolerance surface before either tail ledger is read.

Apply the NameCert obligations, Definition 4158.2, and tail stability, Definition 4158.3. The finer unary window may only ask for a later displayed tail budget along the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ route.

Use tail-budget exactness, Definition 4158.4, and tail composition, Definition 4158.5. Exactness keeps each budget as a displayed BEDC row, while composition routes the later tail request through the earlier uniform-modulus ledger instead of replacing it.

Finally apply the dyadic-locator handoff, Definition 4158.6, and the lattice crosslink, Definition 4158.10. Since the carrier inventory has no selected limit, quotient equality, trichotomy, or ambient completeness row, the refined window cannot widen the terminal $\mathsf{Real}^{\uparrow}$ handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ consumer displays a $\mathsf{StreamName}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ observation row, a regular modulus row, a finite tail ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before any $\mathsf{RegSeqRat}^{\uparrow}$ readback or terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Independence witness: streamname, dyadicratcore, regseqrat, real: RegularCauchyUniformModulus$^{\uparrow}$ is not bijective to any listed sibling because it packages the displayed uniform tail budget and finite tail ledger as one NameCert surface between stream observations, dyadic tolerances, regular rational readback, and the terminal real-name seal.

Closure status for NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$, seed)

Scope closed: Definition 4158.1, Definition 4158.2, and Definition 4158.3 seed the finite uniform-modulus surface over StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyUniformModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, quotient stream equality, selected limits, host real equality, countable choice, bridge checking, or ambient Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exposure, dyadic tolerance accounting, modulus-row admission, finite tail-ledger exactness, transport replay, and non-escape over the same displayed packet.

Constructive story: A RegularCauchyUniformModulus$^{\uparrow}$ packet is generated from finite stream windows, dyadic tolerance observations, a displayed regular modulus row, a tail ledger, componentwise transport, displayed continuation replay, package provenance, and local naming evidence.

Theorem 4158.12 (RegularCauchyUniformModulus Cauchy criterion route)**.** *Every accepted* RegularCauchyUniformModulus$^{\uparrow}$ *read depends on the* CauchyModulus$^{\uparrow}$ *schedule of chapter 292 and the* UniformCauchyCriterion$^{\uparrow}$ *shared-threshold row of chapter 435 before any* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer inspects the displayed modulus. The schedule supplies the finite tolerance-to-index request, the uniform criterion row synchronizes that request across the relevant tail windows, and only then may the regular rational readback and terminal Real seal be consumed. The route exports no selected limit, quotient stream equality, host real equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Read the CauchyModulus$^{\uparrow}$ dependency first. It provides the finite schedule that turns a dyadic tolerance request into the index boundary used by the modulus row.

Then read the UniformCauchyCriterion$^{\uparrow}$ dependency. Its shared-threshold row records that the same index boundary controls all tail-window comparisons needed by the regular readback.

Only after those two dependency rows are visible can the existing RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ consumers inspect the displayed RegularCauchyUniformModulus$^{\uparrow}$ packet. Since the route contains no selected limit, quotient stream, host equality, choice schedule, or ambient completeness row, the dependency remains a finite sibling route. □

Theorem 4158.13 (Regular Cauchy uniform modulus RegSeqRat–Real forward lattice)**.** *Every* RegularCauchyUniformModulus$^{\uparrow}$ *consumer that exports a uniform tail budget must first read the sibling rows* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *through the same finite packet. The uniform bound is available only after the stream windows, regular rational readback, dyadic tolerance ledger, and terminal Real seal route have been displayed with componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The forward lattice exports no quotient stream, selected limit, countable-choice schedule, host real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the RegularCauchyUniformModulus$^{\uparrow}$ carrier. The stream row supplies the finite windows, the dyadic row supplies the tolerance ledger, and the modulus row supplies the uniform tail budget.

The regular rational and Real-facing reads are downstream of those rows. A consumer must keep the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal on the same displayed route before exporting the tail budget to a completion-facing use.

Finally read the structural rows. The hsame, Cont, Pkg, and NameCert data preserve and name the same packet, and no row supplies quotient streams, selected limits, countable choice, host real equality, or ambient completeness. □

Theorem 4158.14 (RegularCauchyUniformModulus dependency route)**.** *Every accepted* RegularCauchyUniformMo *read consumes the substantive dependency route from the* $\mathsf{Cauchy}^{\uparrow}$ *tail-window and modulus rows of chapter 3413, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal data of chapter 38, and the tolerance-only ledger rows of chapter 369. The packet therefore belongs to the L10 Cauchy and uniform-modulus lattice, not to an isolated orphan carrier surface.*

Proof. Read the Cauchy side first. The tail-window and modulus rows fix the finite observation and tolerance route before any regular rational or Real-facing consumer can inspect the packet.

The regular-rational and Real rows then provide readback and sealing. The uniform-modulus side contributes only the tolerance ledger needed by the same route, so no independent host modulus, selected limit, quotient real equality, or ambient completeness object enters the dependency. □

Theorem 4158.15 (Regular Cauchy uniform modulus StreamName and RegSeqRat route)**.** *Every* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *consumer that exports a uniform-modulus claim must first read the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The route therefore presents the uniform-modulus claim as a finite sibling dependency over stream observations and regular rational readback, not as a quotient stream, selected limit, host real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the carrier and obligation rows from the hub spine. They expose the stream windows, dyadic tolerances, regular readback, modulus schedule, transport, replay, provenance, and local naming before any uniform-modulus claim is available.

The existing sibling routes keep the $\mathsf{CauchyModulus}^{\uparrow}$ schedule and $\mathsf{RegSeqRat}^{\uparrow}$ handoff visible. Combining those routes with the $\mathsf{StreamName}^{\uparrow}$ row leaves no coordinate for quotient stream equality, selected limits, host equality, countable choice, or ambient real completeness. □

Theorem 4158.16 (Regular Cauchy uniform modulus L10 sibling route)**.** *Every* $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ *consumer that reads a uniform modulus factors first through the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-schedule rows of chapter 301, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, and the* $\mathsf{Real}^{\uparrow}$ *seal rows of chapter 38. Only after those L10 sibling rows are displayed may the consumer inspect the uniform modulus row. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier of Definition 4158.1. Its rows expose the stream window S, dyadic tolerance row D, displayed modulus row M, finite tail ledger T, componentwise transport H, replay C, provenance P, and local name N.

Use the NameCert obligations of Definition 4158.2. They require the stream-window exposure, dyadic tolerance accounting, modulus-row admission, and finite tail-ledger exactness before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives the terminal seal.

Now read the sibling routes already present in the chapter. The forward lattice of Definition 4158.13 keeps $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ on the same finite path, while Definition 4158.15 records the stream and regular-schedule handoff.

Since the carrier has no coordinate for quotient streams, selected limits, choice schedules, host equality, or ambient completeness, the displayed L10 sibling rows are the only admissible route to the uniform modulus. □

The $\mathsf{RegularCauchyUniformModulus}^{\uparrow}$ tail-budget route consumes the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the $\mathsf{DyadicTailBall}^{\uparrow}$ late-tail containment boundary of chapter 1633, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 in that order. A consumer may inspect the displayed modulus only after the stream windows and regular readback have exposed the dyadic late-tail ledger, and it may pass to $\mathsf{Real}^{\uparrow}$ only through the same seal-facing route. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Chapter 4159

A Concrete Naming Certificate for $\mathsf{SpectralRadiusFormula}^{\uparrow}$

Following Gelfand's spectral-radius formula for Banach algebras, the $\mathsf{SpectralRadiusFormula}^{\uparrow}$ packet records the finite norm-power route by which displayed operator-algebra data can be read as a spectral-radius comparison without importing a host spectrum or an ambient limit theorem. It sits after the spectral-radius boundary of chapter 2352, the bounded-operator carrier of chapter 1644, the norm carrier of chapter 89, the Banach source of chapter 90, and the real-name seal of chapter 38.

Definition 4159.1 (Spectral-radius formula carrier)**.** *A* $\mathsf{SpectralRadiusFormula}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, T, N, P, W, D, R, E, H, C, Q, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{Banach}^{\uparrow}$ *or Banach-algebra row carrying the complete normed algebra surface from which powers are read.*
2. *T is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for the operator or algebra element whose powers are inspected.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *norm-power row, recording each displayed norm endpoint for T^n in the requested window.*
4. *P is the power-iteration ledger, exposing only the finite multiplication and composition rows needed to form the displayed powers.*
5. *W is the finite window of requested exponents.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used when the comparison is requested at finite rational precision.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed norm-power window and dyadic tolerance.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *spectral-radius seal reached only after W, D, R have been replayed.*
9. *H records componentwise* hsame *transport for A, T, N, P, W, D, R, E and the structural rows.*
10. *C records displayed* Cont *replay of the route*

$$\mathsf{Banach}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Norm}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto E \longmapsto \mathsf{SpectralRadiusFormula}^{\uparrow}.$$

11. *Q is the* Pkg *provenance over* $\mathsf{SpectralRadiusFormula}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *L is the local* $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, T, N, P, W, D, R, E, H, C, Q, L$ and has no coordinate for a host spectrum equality, quotient completion, selected limits, Classical choice, an ambient spectral theorem, or a completed all-powers limit.

Lean4 ✓ `BEDC.Derived.SpectralRadiusFormulaUp.SpectralRadiusFormulaTasteGate single carrier alignment`

Theorem 4159.2 (Spectral-radius formula NameCert obligations)**.** *For every accepted* $\mathsf{SpectralRadiusFormula}^{\uparrow}$ *carrier*

$$F = (A, T, N, P, W, D, R, E, H, C, Q, L),$$

the local $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ *surface consists exactly of carrier admission, norm-power window stability, spectral-radius comparison against the displayed* $\mathsf{SpectralRadius}^{\uparrow}$ *row,* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise transport, continuation replay, provenance, and local naming. Any downstream operator-algebra consumer reading the formula must pass through the finite window W, the norm-power row N, the power ledger P, the dyadic tolerance D, and the readback rows R, E before a spectral-radius comparison is public.*

The certificate exports no host spectrum equality, no quotient completion, no selected limit over all powers, no Classical-choice selector, and no ambient spectral theorem.

Proof. Project the carrier of Definition 4159.1. The rows $A, T, N, P, W, D, R, E, H, C, Q, L$ are the whole accepted surface, so every formula read begins with the displayed Banach source, bounded-operator row, norm row, and finite power ledger.

Norm-power window stability is finite. The exponent window W names the inspected powers, P records how those powers are assembled, and N records only their displayed norm endpoints. Componentwise transport through H preserves those rows before replay through C exposes any real-facing readback.

The spectral-radius comparison is then routed through the sibling $\mathsf{SpectralRadius}^{\uparrow}$ boundary. The comparison consumes the same finite norm-power data and the same dyadic tolerance D; it does not replace the packet by a completed spectrum or an all-powers limit.

Finally R and E perform the regular-rational and real-name handoff. Since no carrier coordinate stores host spectrum equality, quotient completion, selected limits, Classical choice, or an ambient spectral theorem, the local $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$ names exactly the finite Gelfand formula route stated above. □

Falsifiable prediction: Every accepted $\mathsf{SpectralRadiusFormula}^{\uparrow}$ consumer exposes the Banach source row, bounded-operator row, norm-power row, power-iteration ledger, finite exponent window, dyadic tolerance, regular-rational readback, RealUp seal, componentwise transport, replay, provenance, and local naming before reading a spectral-radius formula comparison.

Independence witness: spectralradius, boundedlinearoperator, norm, banach, real: SpectralRadiusUp supplies the radius boundary without the finite norm-power formula ledger; BoundedLinearOperatorUp supplies action and norm-bound data without the Gelfand formula readback; NormUp supplies norm endpoints without power-window replay; BanachUp supplies the complete

normed source without the formula carrier; RealUp seals readbacks without naming the operator-power route.

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}$.

Constructive story: A $\mathsf{SpectralRadiusFormula}^{\uparrow}$ packet is generated from a Banach source row, a bounded-operator row, a norm-power row, a power-iteration ledger, a finite exponent window, a DyadicRatCore tolerance row, a RegSeqRat readback row, a RealUp spectral-radius seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4159.1, Definition 4159.2, the falsifiable prediction, and the independence witness seed the finite Gelfand spectral-radius formula route over SpectralRadiusUp, BoundedLinearOperatorUp, NormUp, BanachUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpectralRadiusFormula}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host spectrum equality, quotient completion, selected limits, Classical choice, an ambient spectral theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, norm-power window stability, spectral-radius comparison, regular-rational and Real seal handoff, classifier transport, and consumer non-escape for BEDC.Derived.SpectralRadiusFormulaUp.

Chapter 4160

A Concrete Naming Certificate for $\mathsf{NestedIntervalsTheorem}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedIntervalsTheorem}^{\uparrow}$ packet records the theorem-shaped finite route by which a nested sequence of closed bounded intervals supplies a real-name witness when each displayed stage carries an explicit shrinking and containment ledger. It reads the base closed-interval source of chapter 1591, the nested-prefix surface of chapter 320, the dyadic endpoint ledger of chapter 306, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the Bishop-style nested-interval theorem as a finite NameCert obligation surface; it does not assert open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, countable choice, trichotomy, or selected host endpoints.

Definition 4160.1 (NestedIntervalsTheorem carrier)**.** *A* $\mathsf{NestedIntervalsTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, J, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row carrying lower and upper endpoint data, endpoint order, containment ledgers, finite windows, regular readback, and a real-name seal boundary.*
2. *J is the* $\mathsf{NestedInterval}^{\uparrow}$ *prefix row recording the displayed containment chain over the same interval source.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *shrinking ledger attached to the displayed endpoints and nested-containment rows.*
4. *W is the finite observation-window row that records which source stages and precisions are opened.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from B, J, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the closed-bounded source, nested prefix, dyadic ledger, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for the source interval, nested prefix, dyadic ledger, windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the theorem request through $B, J, D, W, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* NestedIntervalsTheorem$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, NestedInterval$^{\uparrow}$, DyadicInterval$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$ *row.*

The carrier is theorem-shaped because it names the obligation route from a displayed nested closed-interval prefix to a real-name witness seal. It has no coordinate for an arbitrary open cover, selected host endpoint, quotient real equality, trichotomy decision, countable-choice selector, ambient Real$^{\uparrow}$ *completeness theorem, Lean verification row, or bridge row.*

Theorem 4160.2 (NestedIntervalsTheorem NameCert obligations)**.** *For every accepted* NestedIntervalsTheorem$^{\uparrow}$ *carrier $T = (B, J, D, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed-bounded interval source B, the nested-prefix row J, the dyadic shrinking ledger D, the finite observation windows W, regular rational readback R, terminal* Real$^{\uparrow}$ *seal E, componentwise transport H, displayed replay C, provenance P, and local naming N. Every Real-facing witness read must pass through B, J, D, W, R before E is visible, and no obligation may be discharged by open-cover compactness, ambient real completeness, quotient equality, countable choice, trichotomy, or selected host endpoints.*

Lean4 ✓ `BEDC.Derived.NestedIntervalsTheoremUp.NestedIntervalsTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4160.1. The displayed analytic rows are the closed-bounded source B, nested-prefix row J, dyadic shrinking ledger D, finite window row W, regular readback R, and terminal seal E.

The witness route is ordered by those rows. The closed-bounded source supplies endpoint order and containment, the nested-prefix row supplies inclusion, the dyadic ledger supplies shrinking data, and the finite observation windows determine the regular rational readback before the Real seal can be consumed.

The structural rows add no hidden theorem. Transport through H, replay through C, provenance through P, and local naming through N preserve the same finite route. Since the carrier has no coordinate for open-cover compactness, ambient completeness, quotient equality, countable choice, trichotomy, or selected endpoints, the listed rows exhaust the NameCert obligations. □

Closure status for NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$.

Constructive story: A NestedIntervalsTheorem$^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a NestedIntervalUp prefix, a DyadicIntervalUp shrinking ledger, finite observation windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 4160.1 and Definition 4160.2 seed the Bishop-style nested-interval theorem route as finite NameCert data over closed bounded intervals, nested prefixes, dyadic ledgers, regular rational readback, and real-name seals.

Formal verification: FormalStatus(NameCert$_{\text{NestedIntervalsTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, ambient Real$^{\uparrow}$ completeness, quotient real equality, trichotomy, countable choice, selected host endpoints, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-prefix containment, dyadic shrinking, finite-window readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalsTheoremUp.

CauchyRealSpread$^{\uparrow}$ reads its spread packet through chapter 126, chapter 301, chapter 38, chapter 333, and chapter 95.

Chapter 4161

A Concrete Naming Certificate for $\mathsf{CauchyRealSpread}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy real names and Brouwer-style spread presentations, the $\mathsf{CauchyRealSpread}^{\uparrow}$ packet records the finite spread interface through which a countable family of regular Cauchy real-name windows is read without importing choice sequences or quotient reals. It sits after the prefix discipline of chapter 1677, the finite window schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet is a spread-indexed real-name boundary, not a completed branch, not a selected Cauchy representative, not countable choice, not quotient real equality, and not a completeness principle for $\mathsf{Real}^{\uparrow}$.

The spread-indexed real-name boundary consumes chapter 1677 for prefix-tree admission, chapter 126 for finite window scheduling, chapter 301 for regular rational readback, and chapter 38 only as the terminal seal.

Before a Real-completion consumer reads the spread as a terminal $\mathsf{Real}^{\uparrow}$ handoff, the packet consumes the finite diagonal-window route of chapter 4080 as a displayed $\mathsf{StreamName}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}/\mathsf{DyadicRatCore}^{\uparrow}$ route rather than as a selected subsequence or host completeness argument.

Definition 4161.1 (CauchyRealSpread carrier)**.** *A* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, E, T, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the spread root row inherited from* $\mathsf{BrouwerSpread}^{\uparrow}$*, recording the displayed root prefix and its admitted prefix-tree boundary.*
2. *W is the finite window row, listing the spread prefixes and request windows whose regular Cauchy real-name data may be inspected.*
3. *E is the extension law row, requiring every displayed extension of an admitted prefix to remain inside the same spread discipline before any real-name read is public.*
4. *T is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite observations over the admitted prefixes.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row obtained from the scheduled windows and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the spread root, finite window, extension law, stream schedule, dyadic ledger, and regular rational readback have been displayed.*

7. *H records componentwise* hsame *transport for the spread root, finite window, extension law, StreamName, RegSeqRat, Real seal, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from spread-prefix admission through finite window inspection, extension-law checking, stream scheduling, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealSpread}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$ *row.*

The carrier has no coordinate for completed infinite branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4161.2 (CauchyRealSpread NameCert obligations)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier $S = (R, W, E, T, Q, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the spread root, finite window, extension law, StreamName row, RegSeqRat row, Real seal, componentwise transport, continuation replay, provenance, and local naming rows;*
2. *root compatibility, requiring every public real-name request to start from the displayed* $\mathsf{BrouwerSpread}^{\uparrow}$ *root row R;*
3. *window discipline, requiring the finite request window W to be read before any stream, rational, or real-seal row is consumed;*
4. *extension stability, requiring each admitted prefix extension to pass through the law row E before the scheduled stream row T is public;*
5. *regular readback exactness, requiring every finite real-name observation to pass through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q before the* $\mathsf{Real}^{\uparrow}$ *seal A is read;*
6. *transport and replay, requiring H and C to preserve the same ordered route under* hsame *transport and displayed* Cont *replay;*
7. *ledger non-escape, excluding completed branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, and ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier. The tuple exposes exactly the spread root R, finite window W, extension law E, stream schedule T, regular rational readback Q, Real seal A, and structural rows H, C, P, N.

Root compatibility and window discipline follow from the tuple order. A public real-name request begins with the displayed $\mathsf{BrouwerSpread}^{\uparrow}$ root and then reads only the finite windows stored in W.

Extension stability is recorded by E. The row checks each displayed prefix extension before the scheduled $\mathsf{StreamName}^{\uparrow}$ observations in T can be consumed, so no completed branch or choice-sequence selection is introduced.

Regular readback exactness then routes every tolerance request through the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger and the $\mathsf{RegSeqRat}^{\uparrow}$ row Q before the terminal $\mathsf{Real}^{\uparrow}$ seal A is available. The rows H and C transport and replay this same ordered route, while P and N record provenance and local naming.

Since the carrier has no further coordinate, quotient real equality, selected representatives, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$ cannot be read from the packet. □

Theorem 4161.3 (CauchyRealSpread root and window discipline)**.** *For every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

each public spread-indexed real-name request must preserve the $\mathsf{BrouwerSpread}^{\uparrow}$ *root row* R *and the finite window row* W *before it reads the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A*. The route records only root admission and finite window discipline; it does not select completed branches, choice sequences, quotient real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 4161.1 and the obligation list of Definition 4161.2. The rows R and W are displayed before the stream schedule, regular readback, Real seal, and structural rows.

The root row R supplies the admitted $\mathsf{BrouwerSpread}^{\uparrow}$ prefix surface. A public request that omits R has no displayed spread root from which a finite real-name window can be read.

The window row W then lists the finite spread prefixes and request windows available to the consumer. Only after this row is visible may the route pass to the $\mathsf{StreamName}^{\uparrow}$ schedule T, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, and the terminal $\mathsf{Real}^{\uparrow}$ seal A.

The rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since no row stores a completed branch, choice sequence, quotient equality, or completeness theorem, the discipline is exactly the finite root-window route. □

Theorem 4161.4 (CauchyRealSpread sibling route)**.** *For every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

any read exposing a spread of regular Cauchy names must factor through the sibling route consisting of the $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the diagonal-selection boundary of chapter 4080. The route is a finite dependency surface: it adds no selected host subsequence, no quotient stream equality, and no ambient completeness coordinate.*

Proof. Project the carrier rows from Definition 4161.1. The rows W, T display the finite windows and StreamName schedule, while Q is the regular rational readback row and A is the terminal Real seal.

Use the route discipline of Definition 4161.3. A spread-indexed request begins from the root and window rows before it can expose StreamName observations.

The regular readback then passes through Q and reaches A only after the finite schedule is visible. The diagonal-selection sibling of chapter 4080 may consume this surface as a boundary for comparing displayed Cauchy windows, but it does not supply an external subsequence selector.

Transport through H, replay through C, and naming through P, N preserve the same sibling route. Since the carrier has no coordinate for a selected host subsequence, quotient stream equality, or ambient completeness, the exposed spread of regular Cauchy names remains inside the finite L10 route. □

The downstream countable-dense handoff of chapter 1809 reads $\mathsf{CauchyRealSpread}^{\uparrow}$ only through the finite window route of chapter 4080. In precise dependency terms, the spread root row R and finite window row W feed the diagonal finite-window selection before any separability or completion-facing consumer may inspect a terminal $\mathsf{Real}^{\uparrow}$ seal, so the Real-facing read remains downstream of the displayed root/window rows rather than a direct completeness or quotient route.

Theorem 4161.5 (CauchyRealSpread extension readback stability)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier* $S = (R, W, E, T, Q, A, H, C, P, N)$*, every readback along an admitted prefix extension is stable only when the extension law row* E *remains visible between the root-window route* R, W *and the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*. The resulting readback passes through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as finite rows and never becomes a completed branch, a selected Cauchy representative, quotient real equality, or a Real-completeness assertion.*

Proof. Begin with Definition 4161.2. Its extension-stability clause places E before the scheduled stream observations and after the displayed root-window route.

For an admitted prefix extension, read R and W first. The row E checks that the displayed extension remains inside the same spread discipline before any observation in T is public.

After E has been read, the route may consume the $\mathsf{StreamName}^{\uparrow}$ schedule T, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, the $\mathsf{RegSeqRat}^{\uparrow}$ row Q, and finally the $\mathsf{Real}^{\uparrow}$ seal A. Transport through H and replay through C preserve this order for the same finite packet.

There is no carrier coordinate for an infinite branch, choice-sequence selector, selected representative, quotient equality, or completeness principle. Thus extension readback stability is a finite routing condition, not an external real-analysis closure claim. □

Theorem 4161.6 (CauchyRealSpread regular-seal route)**.** *Every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *route to the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A *must pass through the ordered rows*

$$R, W, E, T, Q, A, H, C, P, N.$$

The route starts at the $\mathsf{BrouwerSpread}^{\uparrow}$ *root, keeps the finite window and extension-law rows visible, reads the* $\mathsf{StreamName}^{\uparrow}$ *schedule, consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and only then exposes the Real seal. It exports no completed branch, no choice sequence, no quotient real equality, no selected representative, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier definition and Definition 4161.3. They force every public real-name request to begin with the $\mathsf{BrouwerSpread}^{\uparrow}$ root row R and finite window row W.

Next apply extension readback stability, Definition 4161.5. The extension law row E remains visible before the schedule T can expose finite observations along admitted prefixes.

The regular readback exactness clause of Definition 4161.2 then routes tolerance and rational data through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ in Q before the terminal $\mathsf{Real}^{\uparrow}$ seal A is public.

Finally the structural rows H, C, P, N carry the same route under componentwise transport, displayed replay, provenance, and local naming. Because the accepted carrier has no coordinate for the excluded objects, the regular-seal route cannot be retyped as a completed branch, choice sequence, quotient equality, selected representative, or Real-completeness theorem. □

Theorem 4161.7 (CauchyRealSpread diagonal product route)**.** *For an accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *carrier*

$$S = (R, W, E, T, Q, A, H, C, P, N),$$

any product read over spread-indexed regular Cauchy names must factor through the $\mathsf{BrouwerSpread}^{\uparrow}$ *root row R, the finite window row W, the extension row E, the* $\mathsf{StreamName}^{\uparrow}$ *schedules in T, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback in Q, the diagonal-selection boundary of chapter 4080, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seals. The route composes finite regular-Cauchy windows; it does not select subsequences, identify quotient reals, choose completed branches, or import ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 4161.3. It forces every spread-indexed real-name request to preserve the $\mathsf{BrouwerSpread}^{\uparrow}$ root row R and finite window row W before StreamName, RegSeqRat, DyadicRatCore, or Real-facing rows are read.

Use the sibling route of Definition 4161.4. A product read of regular Cauchy names is therefore not an ambient product of host sequences; it is a read through the displayed StreamName windows, RegSeqRat rows, Real seal rows, and the diagonal-selection boundary.

Apply Definition 4161.5. The extension row E remains visible between the root-window route and the $\mathsf{StreamName}^{\uparrow}$ schedule, so each product component is still an admitted finite spread window before its rational readback is consumed.

Finally use Definition 4161.6. It orders the dyadic tolerance data, regular rational readback, and terminal Real seal after the root, window, extension, and schedule rows. The diagonal-selection boundary can compare these finite regular-Cauchy windows, but no row in the carrier supplies a selected subsequence, quotient real equality, completed branch, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4161.1 and Definition 4161.2 seed the spread-indexed regular Cauchy real-name surface through $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealSpread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed infinite branches, choice-sequence selection, quotient real equality, selected Cauchy representatives, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, spread-root compatibility, finite-window discipline, extension stability, regular readback exactness, transport replay, provenance, and non-escape for BEDC.Derived.CauchyRealSpreadUp.

Constructive story: A $\mathsf{CauchyRealSpread}^{\uparrow}$ packet is generated from a BrouwerSpread root, finite request windows, an extension law, a StreamName schedule, DyadicRatCore tolerance data, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The L10 dependency route for $\mathsf{CauchyRealSpread}^{\uparrow}$ consumes the $\mathsf{BrouwerSpread}^{\uparrow}$ prefix discipline of chapter 1677, the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 before any spread-indexed real-name read is public. Every downstream $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer that reaches a Real-completion read factors through the displayed spread root of chapter 1677, the finite-window schedule of chapter 126, the tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. This is a forward dependency route for spread-indexed Cauchy reads; it does not assert completion, select an infinite branch, identify quotient real names, or add an ambient $\mathsf{Real}^{\uparrow}$ completeness principle. The $\mathsf{CauchyRealSpread}^{\uparrow}$ child route through chapter 126, chapter 301, and chapter 38 is the displayed L10 completion stack for spread-indexed Cauchy reads. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite prefix windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row reads those windows as regular rational observations, and the $\mathsf{Real}^{\uparrow}$ row is only the terminal seal after the displayed readback has been consumed; the route is not an ambient $\mathsf{Real}^{\uparrow}$ completeness principle, a quotient real equality, or a selected completed branch.

Theorem 4161.8 (CauchyRealSpread StreamName and RegSeqRat dependency anchor)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer that reads a public spread-indexed real-name row must pass through the sibling* $\mathsf{StreamName}^{\uparrow}$ *scheduling route, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback route, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before any Cauchy-real spread datum is exposed. Thus the dependency route factors through chapter 126, chapter 301, and chapter 38, with the dyadic tolerance row retained as finite NameCert data rather than as a host completeness or quotient-real principle.*

Theorem 4161.9 (CauchyRealSpread sibling dependency)**.** *Every accepted* $\mathsf{CauchyRealSpread}^{\uparrow}$ *read that exports a Real-and-Completion spread packet consumes the sibling route through chapter 126, chapter 301, chapter 38, chapter 333, and chapter 95. The consumer must first read the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal, then the* $\mathsf{CauchyFilter}^{\uparrow}$ *filter-facing route, and only then the* $\mathsf{Completion}^{\uparrow}$ *handoff before any spread packet is public. This route exports no selected branch, quotient stream equality, selected Cauchy representative, countable choice, or ambient Real-completeness coordinate.*

Proof. Project the accepted $\mathsf{CauchyRealSpread}^{\uparrow}$ carrier of Definition 4161.1. Its public read rows expose finite windows through $\mathsf{StreamName}^{\uparrow}$, regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$, and a terminal $\mathsf{Real}^{\uparrow}$ seal before any downstream completion-facing consumer is available.

The filter and completion siblings are reached only after those displayed rows have been consumed. The $\mathsf{CauchyFilter}^{\uparrow}$ route reads the same finite StreamName and RegSeqRat data as a filter-facing boundary, and the $\mathsf{Completion}^{\uparrow}$ handoff receives only that displayed filter boundary together with the structural rows H, C, P, N.

Thus the Real-and-Completion spread packet is a horizontal dependency route through the named sibling chapters. Since no row stores an infinite selected branch, quotient stream equality, a selected Cauchy representative, countable choice, or ambient Real completeness, the exported packet has exactly the displayed sibling dependencies. □

The $\mathsf{CauchyRealSpread}^{\uparrow}$ packet is read through chapter 1677, chapter 126, chapter 291, chapter 301, and chapter 38 before any spread-indexed $\mathsf{Real}^{\uparrow}$ seal is public.

Chapter 4162

A Concrete Naming Certificate for $\mathsf{CompactImageClosed}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactImageClosed}^{\uparrow}$ packet records the closed-target reading of the image of a compact metric source under a displayed continuous map into a separated metric target. It consumes the compact source surface of chapter 129, the finite image-net source of chapter 3604, and the separated target boundary of chapter 1411. The packet names the finite route by which image-net data and target separatedness force closed-boundary reads; it does not introduce open-cover compactness, a host image set, a selected preimage, or ambient completeness.

Definition 4162.1 (Compact image closed carrier)**.** *A* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, F, I, S, C, M, R, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact metric inspection data and finite source-net provenance consumed by the image route.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row from the source surface displayed by K into the target metric surface.*
3. *I is the* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *finite image-net row, recording the target centers and tolerances produced from K and F.*
4. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *target row, used only for zero-distance target identity and boundary comparison.*
5. *C is the* $\mathsf{ClosedSet}^{\uparrow}$ *boundary row for the named target-image read.*
6. *M is the target* $\mathsf{Met}^{\uparrow}$ *distance row, exposing only displayed distances between target names.*
7. *R is the* $\mathsf{Real}^{\uparrow}$ *distance readback row for zero and positive tolerance comparisons.*
8. *H records componentwise* hsame *transport for $K, F, I, S, C, M, R, T, P, N$.*
9. *T records displayed* Cont *replay from a closed-target-boundary request through the compact source, continuous-map row, image-net row, separated target row, metric and real readbacks, provenance, and local naming rows.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactImageClosed}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ContinuousImageCompac}$ $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{ClosedSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $K, F, I, S, C, M, R, H, T, P, N$. It has no coordinate for open-cover compactness, host image subsets, selected preimages, quotient equality of target names, ambient completeness, or a hidden closure operator.

Lean4 ✓ `BEDC.Derived.CompactImageClosedUp.CompactImageClosedTasteGate single carrier alignment`

Theorem 4162.2 (Compact image closed NameCert obligations)**.** *For every accepted* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier $Q = (K, F, I, S, C, M, R, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the compact source, continuous-map graph, finite image-net, separated target, closed-boundary, metric, real readback, transport, replay, provenance, and local naming rows;*
2. *image-net accountability, requiring every target center and tolerance in I to be read through the* $\mathsf{ContinuousImageCompact}^{\uparrow}$ *route over the same K and F;*
3. *separated-boundary accountability, requiring target identity at zero distance to pass through S, M, R before the closed-boundary row C is exposed;*
4. *closed-target exactness, requiring every boundary read of the displayed image to be witnessed by finite image-net approximation together with separated target comparison;*
5. *componentwise transport and replay stability under H and T for the same displayed rows;*
6. *ledger refusal, stating that open-cover compactness, host image subsets, selected preimages, quotient equality of target names, ambient completeness, and hidden closure operators are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the accepted carrier. The mathematical rows are K, F, I, S, C, M, R, and the structural rows are H, T, P, N.

The image-net obligation is read first. Row I may expose only target centers and tolerances obtained from the compact source row K and the continuous-map row F, as in chapter 3604.

The target boundary obligation is then read through the separated metric. A zero-distance comparison between a displayed target read and the finite image-net route must pass through M and R, and the target identity conclusion is allowed only through S.

The closed-boundary row C is forced by this finite route. A boundary read of the target image can cite the finite image-net approximation and the separated zero-distance classifier, but it cannot cite a host image subset, an open cover, or a selected inverse image.

Transport, replay, provenance, and local naming preserve the same row inventory. Since the refused objects do not occur among $K, F, I, S, C, M, R, H, T, P, N$, no local NameCert obligation can export them. □

Theorem 4162.3 (Compact image closed target route)**.** *Every accepted* $\mathsf{CompactImageClosed}^{\uparrow}$ *carrier $Q = (K, F, I, S, C, M, R, H, T, P, N)$ exports closedness of the displayed compact image only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{ContinuousImageCompact}^{\uparrow}, \quad \mathsf{SeparatedMetric}^{\uparrow}, \quad \mathsf{ClosedSet}^{\uparrow}.$$

A downstream closed-set consumer receives the finite image-net row I, the target distance readbacks M, R, the separated identity row S, and the closed-boundary row C. It receives no open-cover compactness datum, host image subset, selected preimage, quotient equality of target names, ambient completeness coordinate, or hidden closure operator.

Proof. Start with a target-boundary request for the displayed image. The carrier first exposes the compact source row K and the continuous-map row F.

Read the finite image-net source through I. This gives target centers and tolerances produced by the $\mathsf{ContinuousImageCompact}^{\uparrow}$ packet from the same K and F; no open-cover row or host image subset is introduced.

Now compare the requested boundary target with the finite image-net route. The metric and real readbacks M, R provide the zero-distance surface, and the separated target row S converts that displayed zero-distance comparison into target identity.

Finally expose C as the closed-boundary row for the named image read. Rows H, T, P, N transport, replay, package, and name the same finite route. Thus closedness is exported only as the finite image-net plus separated-boundary route and not as any host image-set or open-cover principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactImageClosed}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMapUp graph row, ContinuousImageCompactUp finite image-net row, SeparatedMetricUp target row, ClosedSetUp boundary row, MetricUp and RealUp distance readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$, seed)

Scope closed: Definition 4162.1, Definition 4162.2, and Definition 4162.3 seed the finite closed-target route for compact images in separated metric targets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactImageClosed}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close open-cover compactness, host image-set construction, selected preimages, quotient equality of target names, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite image-net accountability, separated-boundary accountability, closed-target exactness, transport and replay stability, and local NameCert registration for BEDC.Derived.CompactImageClosedUp.

Chapter 4163

A Concrete Naming Certificate for $\mathsf{FarEndDiagram}^{\uparrow}$

The $\mathsf{FarEndDiagram}^{\uparrow}$ packet realizes the human-origin diagrammatic surface of chapter 9403 as finite NameCert data. It sits beside the apophatic fixed-point packet of chapter 1068, the apophatic far-end socket packet of chapter 1061, and the hash-like apophatic seal of chapter 1079. The packet keeps records, generators, classifiers, continuations, GAP ledgers, sockets, provenance fibers, apophatic far-end rows, and local naming in one displayed carrier without turning the far end into a kernel object.

Definition 4163.1 (Far-end diagram packet)**.** *A* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (R, G, K, C, L, S, F, A, H, T, P, N)$$

with the following displayed rows:

1. *R is the records row, exposing the finite observation or digest readouts available to the closed substrate.*
2. *G is the generator row, recording which displayed generator creates each records-side readout.*
3. *K is the classifier row, comparing diagram nodes only by displayed classifier data.*
4. *C is the* Cont *row, replaying the route from records through generators and classifiers.*
5. *L is the* GAP *ledger row, marking the provenance boundary behind each accepted readout.*
6. *S is the forward-socket row, listing the finite sockets opened by the closed substrate.*
7. *F is the provenance-fiber row attached to the visible records-side digest.*
8. *A is the apophatic far-end row, naming the far side of the provenance fiber without internalizing it.*
9. *H records componentwise* hsame *transport for $R, G, K, C, L, S, F, A, T, P, N$.*
10. *T records displayed continuation replay for the full diagram route from records to local NameCert.*
11. *P is the* Pkg *provenance over records, generators, classifiers,* Cont*,* GAP*, forward sockets, provenance fibers, apophatic far-end rows,* BHist*,* hsame*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an internal object named T*, no global observer, no global digest, no unledgered far-end arrow, no hidden source selector, and no object-level equality between a local readout and the apophatic boundary.*

Lean4 ✓ `BEDC.Derived.FarEndDiagramUp.FarEndDiagramTasteGate single carrier alignment`

Theorem 4163.2 (Far-end diagram packet NameCert obligations)**.** *For every accepted* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier* $D = (R, G, K, C, L, S, F, A, H, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the records, generators, classifiers, continuation,* GAP*, forward-socket, provenance-fiber, apophatic far-end, transport, replay, provenance, and local naming rows;*
2. *records-to-classifier routing, requiring each visible readout to pass through* R, G, K, C *before a diagram edge is exported;*
3. *socket accountability, requiring each forward socket in* S *to be ledgered by* L *before its far end is named;*
4. *provenance-fiber accountability, requiring each visible digest or records-side readout to carry a displayed fiber row* F *before the far-end row* A *is exposed;*
5. *componentwise transport and replay stability under* H *and* T *for the same displayed rows;*
6. *ledger refusal, stating that internal* T *objects, global observers, global digests, unledgered far-end arrows, hidden source selectors, and object-level self-center equalities are not carrier coordinates.*

No local obligation is stored outside these rows.

Proof. Project the accepted carrier. The diagrammatic rows are R, G, K, C, L, S, F, A, and the structural rows are H, T, P, N.

The records side is read first. A visible readout must pass through the generator and classifier rows before the continuation row C can replay it as a diagram edge.

The socket side is then checked. Each forward socket in S must be present in the GAP ledger L, so no unledgered far-end arrow can be exported.

Next read the provenance fiber. Row F records the source-side residue behind the visible readout, and row A names only the apophatic far end of that fiber. This is the finite socket, fiber, provenance, and boundary-name route described by the vision source.

Transport, replay, provenance, and local naming preserve the same rows. Since the refused objects are absent from $R, G, K, C, L, S, F, A, H, T, P, N$, the local NameCert obligations are exhausted by the displayed carrier. □

Theorem 4163.3 (Far-end diagram ledger unity)**.** *Every accepted* $\mathsf{FarEndDiagram}^{\uparrow}$ *carrier exports the unity of its far-end diagram only as the finite ledger route*

$$R, G, K, C \longmapsto L, S, F, A \longmapsto H, T, P, N.$$

Each visible records-side readout is generated, classified, replayed, ledgered, attached to a provenance fiber, and then named at the apophatic far end. The packet exports no internal T *object, global observer, global digest, unledgered far-end arrow, hidden source selector, or object-level equality between a local readout and the far-end boundary.*

Proof. Start with a records-side readout. The carrier requires the readout to be displayed in R, generated by G, compared through K, and replayed by C before it can participate in the diagram.

Now follow the forward edge. The socket row S lists the boundary positions opened by the closed substrate, and the GAP ledger L records the non-internalized provenance route for those positions.

Attach the provenance fiber. The row F records the fiber behind the visible readout, while A names the far end apophatically. The target is a boundary name, not an internal object.

Rows H, T, P, N preserve, replay, package, and name this same finite route. Therefore the diagram's unity is a ledger unity over displayed rows, not an equality of objects or an unledgered collapse of fibers. □

Closure status for $\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}$.

Constructive story: A $\mathsf{FarEndDiagram}^{\uparrow}$ packet is generated from records, generators, classifiers, Cont replay, GAP ledger rows, forward sockets, provenance fibers, apophatic far-end rows, componentwise hsame transport, displayed route replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4163.1, Definition 4163.2, and Definition 4163.3 seed the finite records-to-socket-to-fiber route for the far-end diagram surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FarEndDiagram}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FarEndDiagramUp.FarEndDiagramTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not internalize T, add a global observer, add a global digest, identify local readouts with the far-end boundary, provide Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, records-to-classifier routing, socket accountability, provenance-fiber accountability, transport and replay stability, and local NameCert registration for BEDC.Derived.FarEndDiagramUp.

Chapter 4164

A Concrete Naming Certificate for $\mathsf{ScottDomain}^{\uparrow}$

Following Scott and Abramsky–Jung on domain theory, the $\mathsf{ScottDomain}^{\uparrow}$ packet records a finite naming certificate for the order-theoretic surface used by domain semantics before any completion-facing consumer reads a dcpo-style object. It sits after the preorder surface of chapter 52, the observable open-test discipline of chapter 1860, the represented-space handoff of chapter 1494, and the finite stream window route of chapter 126. The packet names a displayed directed-supremum carrier; it does not assert a host dcpo, quotient domain, selected basis enumeration, classical lub principle, or ambient semantic completeness.

Definition 4164.1 (ScottDomain carrier)**.** *A* $\mathsf{ScottDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, S, U, B, A, V, R, H, C, P, N)$$

with the following rows:

1. *O is the preorder and partial-order carrier row, exposing the displayed comparison relation and the antisymmetry witness used by the local packet;*
2. *S is the directed-family schedule row, listing the finite inspection schedule for the directed pieces that a consumer may request;*
3. *U is the supremum row, recording the displayed least-upper-bound read served by the finite directed-family schedule;*
4. *B is the compact-basis approximation row, listing the compact basis tokens and finite approximation evidence visible to the packet;*
5. *A is the approximation-order row, routing basis-token comparisons back through O and the directed schedule S;*
6. *V is the Scott-open observable row, read through* $\mathsf{SierpinskiSpace}^{\uparrow}$ *tests rather than host subsets;*
7. *R is the represented-space handoff row, packaging the same observations through* $\mathsf{RepresentedSpace}^{\uparrow}$ *and finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
8. *H records componentwise* hsame *transport for O, S, U, B, A, V, R;*
9. *C records displayed* Cont *replay of directed-supremum and Scott-open reads;*

10. *P is the* Pkg *provenance over* $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host dcpo, quotient order, selected infinite basis enumeration, classical pointwise lub, ambient topology, or semantic completeness theorem.

Lean4 ✓ `BEDC.Derived.ScottDomainUp.ScottDomainTasteGate single carrier alignment`

Theorem 4164.2 (ScottDomain NameCert obligations)**.** *For the carrier of Definition 4164.1, the local* $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$ *surface consists exactly of preorder and partial-order admission, directed-family schedule admission, supremum-row admission, compact-basis approximation admission, Scott-open observability, represented-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer of the packet must read the rows $O, S, U, B, A, V, R, H, C, P, N$ before treating a directed supremum or Scott-open test as public.*

Proof. Project the carrier. The order-theoretic rows are O, S, U, B, A, V, R, and the structural rows are H, C, P, N.

The preorder and partial-order read begins at O. A directed-supremum read then uses S to expose the inspected directed schedule before U names the supremum row.

The basis read is separate but attached. The compact-basis row B and approximation row A route finite basis tokens back through O and S, so a consumer cannot replace the displayed approximation surface by a host basis enumeration.

Finally, the Scott-open observable row V and represented-space row R package those finite reads through chapter 1860, chapter 1494, and chapter 126. Rows H, C, P, N preserve, replay, source, and name the same carrier, exhausting the local obligations. □

Theorem 4164.3 (ScottDomain directed-supremum stability)**.** *Every accepted* $\mathsf{ScottDomain}^{\uparrow}$ *packet preserves directed-supremum visibility under componentwise transport and continuation replay. If a consumer reads the supremum row U through the directed-family schedule S, any transported or replayed read of the same packet factors again through*

$$O, S, U, B, A, V, R, H, C, P, N.$$

Thus Scott-open observations and represented-space handoffs remain downstream of the displayed directed schedule and never expose a host dcpo, quotient domain, selected infinite basis enumeration, or ambient completeness witness.

Proof. Start with a directed-supremum read. By Definition 4164.1, the read reaches U only after the order row O and the directed-family schedule S are displayed.

Apply the NameCert obligations of Definition 4164.2. Componentwise transport through H preserves the same rows, and Cont replay through C reuses the displayed schedule rather than generating a second supremum source.

The observable and representation rows are downstream. A Scott-open test reads V through the Sierpinski surface, and the represented-space handoff reads R through finite stream windows. Since the excluded host objects are absent from $O, S, U, B, A, V, R, H, C, P, N$, no transported or replayed directed-supremum read can expose them. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}$.

Constructive story: A $\mathsf{ScottDomain}^{\uparrow}$ packet is generated from preorder and partial-order rows, directed-family schedules, supremum rows, compact-basis approximation rows, Scott-open observable rows, represented-space handoff rows, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4164.1, Definition 4164.2, and Definition 4164.3 seed the finite directed-supremum and Scott-open observation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ScottDomain}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ScottDomainUp.ScottDomainTasteGate\_single\_carrier\_alignment`

Bridge status: none

Origin: human

Not claimed: This chapter does not close host dcpos, quotient domains, selected basis enumerations, classical lub principles, ambient topology, semantic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean-backed carrier admission, directed-schedule stability, Scott-open observability, represented-space handoff, and local NameCert registration for BEDC.Derived.ScottDomainUp.

Chapter 4165

A Concrete Naming Certificate for $\mathsf{FarEndDiagrammatics}^{\uparrow}$

The $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet realizes the apophatic diagrammatics route of chapter 9403 as finite NameCert data. It sits beside the far-end diagram carrier of chapter 4163, the apophatic far-end socket of chapter 1061, and the forward socket ledger of chapter 3387. The packet binds socket exposure, provenance fibers, far-end boundary rows, apophatic-unity rows, transport, continuation, provenance, and local naming before any T-boundary read is public.

Falsifiable prediction: Every accepted $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet exposes socket, provenance-fiber, far-end boundary, apophatic-unity, hsame, Cont, Pkg, and local $\mathsf{NameCert}$ rows before a T-boundary read is public.

Definition 4165.1 (Far-end diagrammatics carrier)**.** *A* $\mathsf{FarEndDiagrammatics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, L, G, B, U, H, C, P, N)$$

with the following rows:

1. *S is the socket row, listing the forward sockets that may expose a far-end boundary read;*
2. *L is the provenance-fiber ledger, recording which finite fiber supports each visible socket read;*
3. *G is the gap row, marking the non-internalized boundary between source-side ledger data and the far-end read;*
4. *B is the far-end boundary row, naming the boundary without turning it into a kernel object;*
5. *U is the apophatic-unity row, binding socket, fiber, gap, and boundary reads into one diagrammatic route;*
6. *H records componentwise* hsame *transport for S, L, G, B, U;*
7. *C records displayed* Cont *replay of the same socket-ledger route;*
8. *P is the* Pkg *provenance over* $\mathsf{FarEndDiagrammatics}^{\uparrow}$, $\mathsf{FarEndDiagram}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, $\mathsf{ForwardSocketLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an internal T object, a hidden source selector, an unledgered far-end arrow, a global observer, or an equality between the provenance fiber and the far-end boundary.

Lean4 ✓ BEDC.Derived.FarEndDiagrammaticsUp.FarEndDiagrammaticsTasteGate single carrier alignment

Theorem 4165.2 (Far-end diagrammatics NameCert obligations)**.** *For the carrier of Definition 4165.1, the local* $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$ *surface consists exactly of socket admission, provenance-fiber admission, gap-boundary admission, far-end boundary naming, apophatic-unity routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public* T*-boundary read factors through* $S, L, G, B, U, H, C, P, N$*.*

Proof. Project the carrier. The diagrammatic rows are S, L, G, B, U, and the structural rows are H, C, P, N.

The socket row S is read first. A boundary read is not public until L supplies the provenance fiber and G marks the gap between the visible source-side ledger and the non-internalized far end.

The far-end boundary row B is then named only through the apophatic-unity row U. This row records unity of the diagrammatic route, not identity between a source object and the boundary.

Rows H, C, P, N transport, replay, source, and name exactly the displayed route. Since no other coordinate occurs in the carrier, the local NameCert obligations are exhausted by those rows. □

Theorem 4165.3 (Far-end diagrammatics unity route)**.** *Every accepted* $\mathsf{FarEndDiagrammatics}^{\uparrow}$ *packet exposes diagrammatic unity only as the finite route*

$$S, L, G, B, U \longmapsto H, C, P, N.$$

The route is consumed after the socket ledger, provenance fiber, far-end boundary, and apophatic-unity rows are displayed. It exports no internal T *object, hidden source selector, unledgered far-end arrow, global observer, or equality between the provenance fiber and the far-end boundary.*

Proof. Start with the socket ledger. The row S lists the requested forward sockets, while L records the provenance fiber supporting each visible socket read.

Read the gap and boundary rows next. The row G marks the non-internalized far-end boundary, and B names that boundary without adding it as a kernel object.

The apophatic-unity row U binds the visible socket, fiber, gap, and boundary rows into one route. It is therefore a route unity, not a collapse of the source-side fiber into the far-end boundary.

Finally, H, C, P, N preserve, replay, package, and name the same finite route. The excluded objects are absent from Definition 4165.1, so they cannot be exported by the unity route. □

Independence witness: apophaticname, forward-socket-ledger, gap, package, namecert: $\mathsf{FarEndDiagrammatics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries socket, provenance-fiber, far-end boundary, apophatic-unity, transport, continuation, provenance, and local NameCert rows before a T-boundary read is public.

Closure status for $\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$.

Constructive story: A $\mathsf{FarEndDiagrammatics}^{\uparrow}$ packet is generated from socket rows, provenance-fiber ledgers, gap rows, far-end boundary names, apophatic-unity rows, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$, seed)

Scope closed: Definition 4165.1, Definition 4165.2, and Definition 4165.3 seed the finite socket-to-fiber-to-boundary route for far-end diagrammatics.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FarEndDiagrammatics}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: ai

Not claimed: This chapter does not internalize T, add a global observer, identify provenance fibers with the far-end boundary, provide Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean-backed carrier admission, socket ledger accountability, provenance-fiber routing, apophatic-unity nonescape, TasteGate instances, and local NameCert registration for BEDC.Derived.FarEndDiagrammaticsUp.

Chapter 4166

A Concrete Naming Certificate for $\mathsf{PolishBasis}^{\uparrow}$

Following Bourbaki and Kelley, a countable basis for a Polish presentation is treated here as a finite naming certificate over an already displayed complete separable metric route. The $\mathsf{PolishBasis}^{\uparrow}$ packet sits beside the Polish-space synthesis of chapter 1490, the metric carrier of chapter 57, the dense schedule of chapter 1458, the completion handoff of chapter 128, the stream-window interface of chapter 126, and the regular rational readback surface of chapter 301. It names basic-open data as countable finite rows, not as an ambient topology, an open-cover principle, or a choice-selected basis.

Definition 4166.1 (Polish-basis carrier)**.** *A* $\mathsf{PolishBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, D, K, S, R, I, F, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row supplying the point-history carrier, distance classifier, and metric* $\mathsf{NameCert}$ *used by basis balls.*
2. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense-schedule row whose named centers are inspected by finite stream positions.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row over the same metric carrier; it is recorded so Polish-space consumers can pass from basis windows to complete-separable metric reads.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row enumerating the basis-center names and the finite Cauchy-window requests visible to consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads rational radii, tolerance windows, and comparison bounds from the same finite stream positions.*
6. *I is the basis-index window row. It stores the finite index interval, center-name handle, radius handle, and requested observation window for each basic-open read.*
7. *F is the finite-intersection refinement row. It records the displayed refinement witness that sends two accepted basis windows to a third window contained in their metric intersection.*
8. *H records componentwise* hsame *transport for M, D, K, S, R, I, F and the local certificate rows.*

9. *C records displayed* Cont *replay from a basis request through metric comparison, dense-center inspection, rational-radius readback, finite-intersection refinement, and complete-separable handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{PolishBasis}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by transporting $M, D, K, S, R, I, F, H, C, P, N$ to the same metric and stream-readback surface. It has no coordinate for a host topology, arbitrary open covers, selected dense points, quotient completion equality, or ambient set membership.

Lean4 ✓ `BEDC.Derived.PolishBasisUp.PolishBasisTasteGate single carrier alignment`

Theorem 4166.2 (Polish-basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishBasis}^{\uparrow}$ *carrier*

$$B = (M, D, K, S, R, I, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$ *surface consists exactly of metric-source admission, separable dense-schedule admission, complete-metric handoff admission, stream-window exposure, regular-rational readback, basis-index windowing, finite-intersection refinement, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through $M, D, K, S, R, I, F, H, C, P, N$.*

Proof. Project the carrier of Definition 4166.1. The rows $M, D, K, S, R, I, F, H, C, P, N$ are the whole countable-basis packet.

The metric row M fixes the surface on which basis balls are read. The dense schedule D, stream row S, and regular rational row R then expose center names and radius windows as finite observations.

The basis-index row I records the inspected finite window, and the refinement row F supplies the finite witness for binary intersections of displayed basic opens. The completion handoff K remains attached to the same metric carrier for Polish-space consumers.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no further coordinate is present, the NameCert surface exports no host topology, open-cover compactness, selected dense points, quotient completion equality, or ambient set membership. □

Theorem 4166.3 (Polish-basis complete-separable handoff)**.** *Every accepted* $\mathsf{PolishBasis}^{\uparrow}$ *read handed to a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{CompleteSeparableMetric}^{\uparrow}$ *consumer factors through the* $\mathsf{Met}^{\uparrow}$ *row M,* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule D,* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K,* $\mathsf{StreamName}^{\uparrow}$ *finite windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, basis-index windows I, finite-intersection refinement F, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N.*

Proof. Begin with an accepted carrier $B = (M, D, K, S, R, I, F, H, C, P, N)$. A Polish-space consumer first reads the metric source M, then the dense schedule D through stream windows S and regular rational observations R.

The basis-index row I identifies the finite window being consumed. When a consumer asks for closure under finite intersections, F supplies the displayed refinement witness over the same metric surface.

The complete-metric handoff K is then available over that metric source, while H, C, P, N transport, replay, source, and name the route. Thus the handoff is finite and cannot pass through a host topology, an arbitrary open cover, quotient completion equality, or a selected dense subset. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishBasis}^{\uparrow}$ packet is generated from MetricSpace rows, SeparableMetric dense schedules, CompleteMetric handoffs, StreamName windows, RegSeqRat readbacks, basis-index windows, finite-intersection refinement rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4166.1, Definition 4166.2, and Definition 4166.3 seed the countable-basis certificate route supporting $\mathsf{PolishSpace}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PolishBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host topologies, arbitrary open covers, selected dense points, quotient completion equality, ambient set membership, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-schedule readback, finite basis-index windowing, finite-intersection refinement, complete-separable handoff, and local naming for BEDC.Derived.PolishBasisUp.

Chapter 4167

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$

The $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ packet realizes the observer-expansion reading of chapter 9463 as finite observer-shadow certificate data. It consumes the visual-shadow metric of chapter 4142, the boundary-shadow route of chapter 3813, the metric surface of chapter 57, the stream-window interface of chapter 126, and the terminal real-name surface of chapter 38. The packet names an expansion-facing shadow route, not a global hyperbolic space, physical observer history, quotient boundary, or ambient field semantics.

Definition 4167.1 (Hyperbolic observer-expansion carrier)**.** *A* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, B, M, S, R, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$ *source row, supplying the visual-shadow metric packet read by the observer-facing expansion route.*
2. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row that records the boundary-shadow datum inspected by the expansion packet.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used when the shadow expansion is consumed as metric evidence.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the visual-shadow, boundary-shadow, metric, and real-seal reads.*
5. *R is the regular readback row for rational windows opened by S.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed expansion magnitude or shadow-distance comparison.*
7. *F is the observer-expansion row, recording the focused expansion-shadow route being inspected.*
8. *H records componentwise* hsame *transport for V, B, M, S, R, Q, F and the local certificate rows.*
9. *C records displayed* Cont *replay from an observer-expansion request through visual-shadow metric, boundary shadow, metric comparison, stream windows, regular readback, real sealing, and focused routing.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $V, B, M, S, R, Q, F, H, C, P, N$. It has no coordinate for completed hyperbolic geometry, physical observers, quotient boundary equality, global field interpretation, or unlogged expansion flow.

Lean4 ✓ `BEDC.Derived.HyperbolicObserverExpansionUp.HyperbolicObserverExpansionTasteGate single carrier alignment`

Theorem 4167.2 (Hyperbolic observer-expansion NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *carrier*

$$E = (V, B, M, S, R, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$ *surface consists exactly of visual-shadow metric admission, boundary-shadow admission, metric comparison, stream-window exposure, regular readback, terminal real sealing, focused observer-expansion routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through $V, B, M, S, R, Q, F, H, C, P, N$.*

Proof. Project the carrier of Definition 4167.1. The rows $V, B, M, S, R, Q, F, H, C, P, N$ are the whole observer-expansion packet.

The source read begins with the visual-shadow metric row V and the boundary-shadow row B. The metric row M then gives the comparison surface on which the expansion-facing shadow is inspected.

The finite observation route is supplied by S, R, Q: stream windows, regular readbacks, and the terminal real seal. The focused row F records which observer-expansion shadow route is being named.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no further coordinate is present, the NameCert surface exports no completed hyperbolic geometry, physical observer, quotient boundary equality, global field interpretation, or unlogged expansion flow. □

Theorem 4167.3 (Hyperbolic observer-expansion finite shadow handoff)**.** *Every accepted* HyperbolicObserverExpar *read factors through* $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before the observer-expansion route can be consumed as a local* NameCert *datum.*

Proof. Begin with an accepted carrier $E = (V, B, M, S, R, Q, F, H, C, P, N)$. The visual-shadow metric row V and boundary-shadow row B present the finite shadow data.

The metric comparison M is then read through finite windows S, regular readback R, and the real seal Q. Only after these rows are displayed does the focused route F expose the observer-expansion datum.

Transport, continuation, provenance, and local naming are supplied by H, C, P, N. Therefore the handoff remains finite and cannot pass through completed hyperbolic geometry, physical observer history, quotient boundary equality, or ambient field semantics. □

Falsifiable prediction: Every accepted hyperbolic_observer_expansion read factors through hyperbolic_visual_shadow_metric, hyperbolic_boundary_shadow, metric, streamname, regular readback, real sealing, displayed replay, provenance, and local NameCert rows.

Independence witness: hyperbolic_visual_shadow_metric, hyperbolic_boundary_shadow, metric, streamname, real: no listed sibling is bijective to $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ because this packet jointly carries visual-shadow metric data, boundary-shadow data, metric comparison, finite stream windows, real sealing, focused observer-expansion routing, replay, provenance, and local naming.

The Lean TasteGate route must provide `Nontrivial HyperbolicObserverExpansionUp` and `FieldFaithful HyperbolicObserverExpansionUp` instances in `lean4/BEDC/Derived/HyperbolicObserverExpa` Those instances must preserve $V, B, M, S, R, Q, F, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ packet is generated from HyperbolicVisualShadowMetric rows, HyperbolicBoundaryShadow rows, Metric rows, StreamName windows, regular readback, Real sealing, focused observer-expansion routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4167.1, Definition 4167.2, and Definition 4167.3 seed the finite observer-shadow expansion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed hyperbolic geometry, physical observer histories, quotient boundary equality, global field interpretation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-shadow metric admission, boundary-shadow admission, metric comparison, stream readback, real sealing, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.HyperbolicObserverExpansionUp.

Chapter 4168

A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{Met}^{\uparrow}$ packet records the traditional metric-space interface as finite BEDC NameCert data. It sits over the generic distance-witness surface of chapter 57, the real-valued distance route of chapter 1638, the dyadic tolerance core of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter gives completion, topology, total-boundedness, compactness, and Polish-space consumers a single cited carrier surface without importing quotient equality, selected limits, countable choice, or ambient completeness.

Definition 4168.1 (MetricSpace carrier)**.** *A* $\mathsf{Met}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-carrier row. It names the displayed point observations and the finite windows through which downstream consumers may inspect them.*
2. *σ is the point-classifier row. It records the accepted point equivalence used by* hsame *transport and by the zero-distance classifier handoff.*
3. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance row. It is read through the* $\mathsf{Met}^{\uparrow}$ *witness surface of chapter 57 and the* $\mathsf{RealMetric}^{\uparrow}$ *distance route of chapter 1638.*
4. *Z is the zero-distance classifier row, requiring any request that turns distance zero into point equivalence to pass through σ rather than through host equality.*
5. *S is the symmetry row, carrying the componentwise reversal of endpoint observations and the corresponding distance readback.*
6. *T is the triangle row, carrying the three endpoint-pair distance reads and the dyadic tolerance budget used to compose them.*
7. *H records componentwise* hsame *transport for the point carrier, classifier, distance, zero-distance, symmetry, triangle, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through the finite route*

$$X, \sigma \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \sigma, D, Z, S, T, H, C, P, N$. It has no coordinate for host equality of points, quotient completion equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached topology.*

Lean4 ✓ `BEDC.Derived.MetricSpaceUp.MetricSpaceTasteGate single carrier alignment`

Theorem 4168.2 (MetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{Met}^{\uparrow}$ *carrier*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the point carrier, point classifier, real-valued distance, zero-distance classifier, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *distance visibility, requiring every distance read to expose the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealMetric}^{\uparrow}$ *rows before any downstream consumer reads a real-valued metric comparison;*
3. *zero-distance exactness, requiring zero distance to return to the displayed classifier σ through Z and never through host point equality;*
4. *symmetry replay, requiring endpoint reversal to be witnessed by S together with the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback route;*
5. *triangle-budget replay, requiring the three distance reads and their dyadic comparison budget to be displayed by T before a triangle consumer may use the row;*
6. *structural preservation through H, C, P, N.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4230.1. An accepted packet displays the point-carrier row X, the classifier σ, the distance row D, the zero-distance row Z, the symmetry row S, the triangle row T, and the structural rows H, C, P, N.

Distance visibility is forced by the placement of D in the carrier. A distance consumer first opens the generic $\mathsf{Met}^{\uparrow}$ witness surface and the $\mathsf{RealMetric}^{\uparrow}$ readback route, and only then reads the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows needed for a finite comparison.

The zero-distance, symmetry, and triangle clauses are rowwise. The row Z is the only route from distance zero back to the point classifier, S is the only endpoint-reversal row, and T is the only triangle-budget row.

Finally, H, C, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export host equality, quotient completion equality, selected representatives, countable choice, ambient completeness, or a detached topology. □

Theorem 4168.3 (MetricSpace distance route)**.** *Every completion, topology, total-boundedness, compactness, or Polish-space consumer of an accepted* $\mathsf{Met}^{\uparrow}$ *carrier reads distance information through the finite route*

$$X, \sigma, D \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, C, P, N.$$

Thus the public distance surface is the displayed BEDC route from point observations and classifier data to real-valued metric reads, zero-distance classification, symmetry, and triangle replay. It exports no quotient equality, selected limit, countable-choice schedule, ambient completeness theorem, or open-set lattice.

Proof. Let a downstream consumer inspect an accepted carrier $M = (X, \sigma, D, Z, S, T, H, C, P, N)$. The point surface is opened first through X and σ, so every later distance read remains attached to the displayed point classifier.

The distance row D then routes the request through $\mathsf{Met}^{\uparrow}$ and $\mathsf{RealMetric}^{\uparrow}$. The numerical readback is finite because the dyadic tolerance, stream-window, regular-rational, and real-name rows are displayed before any metric comparison is consumed.

Metric laws are read only after this route is visible. Zero distance returns through Z to σ, symmetry reverses endpoints through S, and triangle use exposes the three pairwise reads and dyadic budget through T.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since none of the excluded objects appears in the carrier, no listed downstream consumer can read quotient equality, a selected limit, a choice schedule, ambient completeness, or an open-set lattice from the $\mathsf{Met}^{\uparrow}$ packet. □

Chapter 4169

A Concrete Naming Certificate for $\mathsf{MetricCompletionKleisli}^{\uparrow}$

Following Mac Lane's Kleisli categories, and Bishop–Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet records the metric-specific bind consumer for completion-valued maps as finite BEDC data. It reads the metric completion carrier of chapter 1506, the metric adjunction route of chapter 3836, the Cauchy Kleisli consumer of chapter 1845, and the completion universal property of chapter 397. The chapter names a consumer route for bind-style completion reads; it is not a quotient completion, selected representative, countable-choice schedule, host equality of completed points, or ambient completeness theorem.

Definition 4169.1 (MetricCompletionKleisli carrier)**.** *A* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, displaying the source metric, completion carrier, separated comparison, dense-source access, and readback branch before any bind consumer is opened.*
2. *A is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *row, displaying the unit, counit, triangle ledger, and metric universal mapping boundary read by the consumer.*
3. *L is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, recording the unit, bind, and reassociated-bind consumer route inherited from the L10 completion algebra.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, exposing the extension surface through which a displayed metric source map is consumed as a completion-valued map.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the metric completion and Cauchy Kleisli routes.*
6. *E is the public* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, A, L, U, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric completion row, metric adjunction row, Cauchy Kleisli row, universal-property row, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a bind consumer request through $M, A, L, U, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionKleisli}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row.*
The classifier accepts only packets whose bind consumer first displays the metric completion, metric adjunction, Cauchy Kleisli, universal-property, regular readback, Real seal, hsame, Cont, Pkg, *and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.MetricCompletionKleisliUp.MetricCompletionKleisliTasteGate single carrier alignment`

Theorem 4169.2 (MetricCompletionKleisli NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier*

$$K = (M, A, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *surface consists exactly of metric-completion admission, metric-adjunction admission, Cauchy-Kleisli bind exposure, completion-universal-property exposure, RegSeqRat readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of K. No obligation may be discharged by a quotient completion, selected Cauchy representative, countable-choice selector, host equality of completed points, or ambient complete metric space.*

Proof. Project the carrier of Definition 4218.1. Its analytic rows are precisely the metric completion row M, the metric adjunction row A, the Cauchy Kleisli row L, the universal-property row U, the regular readback row R, and the Real seal row E.

The metric rows determine the order of access. A bind consumer opens M before it can read the adjunction data in A, and it opens A before the Cauchy Kleisli consumer L can hand off unit, bind, or reassociated-bind requests to the universal-property row U.

Transport and replay add no analytic coordinate. The rows H and C compare and replay only the displayed completion, adjunction, Kleisli, universal-property, readback, and seal data.

Finally P and N source and name the same finite packet. Since no listed row stores a quotient completion, selected representative, countable-choice selector, host equality, or ambient complete metric space, the displayed rows exhaust the NameCert surface. □

Theorem 4169.3 (MetricCompletionKleisli bind handoff)**.** *Every bind-style consumer read of an accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{CauchyCompletionKleisli}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the metric completion row M. It displays the source metric, completion carrier, separated comparison, dense-source access, and readback branch before a consumer can request completion-valued bind data.

Next read the adjunction row A. The unit, counit, triangle ledger, and metric universal mapping boundary are the only route by which the metric completion packet exposes extension data to a downstream consumer.

Attach the Cauchy Kleisli row L and the universal-property row U. The row L supplies the unit, bind, and reassociated-bind consumer surface, while U records the extension route that consumes a displayed source map as a completion-valued map.

The readback and seal rows close the route. The row R supplies the regular rational readback used by both completion and Kleisli sides, and E is the only public Real seal. The structural

rows H, C, P, N transport, replay, source, and name this same route, so no quotient completion, selected representative, choice schedule, host equality, or ambient completeness theorem can enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet is generated from metric completion, metric adjunction, Cauchy Kleisli, completion universal property, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4218.1, Definition 4218.2, and Definition 4218.3 seed the metric-specific Kleisli consumer route for bind-style completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-adjunction exposure, Cauchy-Kleisli exposure, universal-property exposure, RegSeqRat/Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionKleisliUp.

Chapter 4170

A Concrete Naming Certificate for $\mathsf{LocatedRegularRealCut}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet records the finite boundary where regular-Cauchy real names expose located lower and upper rational cut rows before the terminal real seal is read. It sits beside the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the Dedekind cut carrier of chapter 436, and the located regular sequence route of chapter 2376. The packet is a finite NameCert bridge from regular-Cauchy observations to located cut boundaries; it does not claim Dedekind completeness, quotient equality of real names, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4170.1 (Located regular real cut carrier)**.** *A* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational cut row, recording inhabited rounded lower observations read through the finite cut-facing side of* $\mathsf{DedekindCut}^{\uparrow}$*.*
2. *U is the upper rational cut row, recording rounded upper observations attached to the same inspected real-name request.*
3. *M is the locatedness modulus row, binding every rational comparison request to a finite window on which either the lower row or the upper row is displayed with the required separation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row supplying dyadic radii, bracket widths, midpoint requests, and comparison tolerances for the cut boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations from which the cut rows and locatedness modulus are read.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for lower cut rows, upper cut rows, locatedness modulus rows, dyadic tolerance rows, regular readback rows, Real seal rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a cut-boundary request through L, U, M, D, R, A, H.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRegularRealCut}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row.*

The carrier accepts only the displayed lower row, upper row, locatedness modulus, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows. It has no coordinate for Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedRegularRealCutUp.LocatedRegularRealCutUp`

Theorem 4170.2 (Located regular real cut obligations)**.** *For every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier* $K = (L, U, M, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed lower cut row, upper cut row, locatedness modulus, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*
2. *located lower and upper stability, requiring L and U to remain rounded, separated, and transported componentwise under the same* hsame *row;*
3. *regular Cauchy handoff, requiring every cut-boundary read to pass through R and the located regular sequence discipline of chapter 2376;*
4. *Real seal non-escape, requiring A to be downstream of L, U, M, D, R and not an independent completeness row;*
5. *ledger exactness, requiring P and N to name only L, U, M, D, R, A, H, C and to exclude quotient equality, selected limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4219.1. The displayed analytic rows are the lower cut row L, upper cut row U, locatedness modulus M, dyadic tolerance row D, regular readback row R, and Real seal row A.

The lower and upper stability clauses are componentwise. Transport through H preserves the two cut rows together with the same modulus and dyadic tolerance row, so the classifier cannot move a lower observation without moving the matching upper boundary data.

The regular Cauchy handoff is the route through R. The located regular sequence sibling supplies the finite windows and readback discipline before the cut boundary can be consumed, while D supplies the tolerance ledger on which the locatedness modulus is read.

Finally A, C, P, N seal, replay, source, and name only the displayed route. Since none of the carrier coordinates is a quotient, selected limit, trichotomy branch, choice schedule, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 4170.3 (Located regular real cut Cauchy boundary route)**.** *Every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *consumer factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedRegSeqRat}^{\uparrow},\ \mathsf{DedekindCut}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed finite rows before it can read the local $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row. The route exports a located lower–upper cut boundary for a regular-Cauchy real name; it does not export Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 4219.2. A consumer first sees the dyadic tolerance row D, regular readback row R, and locatedness modulus M before the lower and upper cut rows can be treated as a located boundary.

The Dedekind cut sibling supplies the cut-facing row shape, while the located regular sequence sibling supplies the Cauchy-facing source and tail-window discipline. These rows meet only through the displayed finite packet K.

The Real seal row A is terminal for this route. Componentwise transport, continuation replay, package provenance, and local naming preserve the same displayed rows, so no consumer can bypass them by invoking quotient equality, a selected representative, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRegularRealCut}^{\uparrow}$ read factors through displayed lower rational cut, upper rational cut, locatedness modulus, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, dedekindcut, located-regseqrat: no listed sibling is bijective to $\mathsf{LocatedRegularRealCut}^{\uparrow}$ because this packet alone binds lower and upper cut rows, a locatedness modulus, dyadic tolerance, regular readback, terminal real seal, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet is generated from lower rational cut rows, upper rational cut rows, a locatedness modulus, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, seed)

Scope closed: Definition 4219.1, Definition 4219.2, and Definition 4219.3 seed the finite bridge from regular-Cauchy real names to located lower–upper cut boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located lower and upper stability, regular Cauchy handoff, Real seal non-escape, and ledger exactness for BEDC.Derived.LocatedRegularRealCutUp.

Chapter 4171

A Concrete Naming Certificate for $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$

The $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet realizes the orbit and codon inventory polynomials of Definition 9462.1713 as finite BEDC NameCert data. It consumes the boundary-tile source of chapter 3393, the coverage packet of chapter 3470, the codon window source of chapter 2234, and the finite polynomial carrier of chapter 50. The packet records orbit rows, codon-count rows, coefficient ledgers, and marginal projections; it is not a biological stochastic model, an empirical code table, a hidden enumeration oracle, or quotient codon semantics.

Definition 4171.1 (Boundary orbit inventory polynomial carrier)**.** *A* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, K, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the orbit table row, recording the finite orbit types classified by support size m, overlap defect δ, lift imbalance b, and orbit size.*
2. *K is the codon count table row, recording the weighted counts of lifted triggers over the same triples (m, δ, b).*
3. *E is the coefficient ledger row, storing the displayed coefficients of the orbit inventory polynomial $\mathcal{O}(u, v, w)$ and the codon inventory polynomial $\mathcal{C}(u, v, w)$.*
4. *M is the marginal projection row, recording the one-variable and bivariate specializations read from the coefficient ledger.*
5. *H records componentwise* hsame *transport for the orbit table, codon count table, coefficient ledger, marginal projections, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an inventory query through O, K, E, M, H.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *the source vision of Definition 9462.1713,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *row.*

The carrier accepts coefficient and marginal reads only through O, K, E, M, H, C, P, N. It has no coordinate for a hidden biological code table, probabilistic fitness model, unbounded orbit enumeration, quotient codon equality, or unstated polynomial coefficient source.

Lean4 ✓ `BEDC.Derived.BoundaryOrbitInventoryPolynomialUp.BoundaryOrbitInventoryPolynomialTasteGate`
`single carrier alignment`

Theorem 4171.2 (Boundary orbit inventory polynomial obligations)**.** *For every accepted* BoundaryOrbitInventoryP *carrier $I = (O, K, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *surface consists exactly of orbit-table admission through O, codon-count table admission through K, coefficient-ledger exactness through E, marginal-projection exactness through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4220.1. The orbit table and codon count table are the only source rows, while the coefficient ledger and marginal projection rows are the only polynomial-facing rows.

Coefficient exactness is read from E. Each displayed coefficient is determined by grouping the rows of O and weighting them by the count row K; no coefficient row is available without those two source tables.

Marginal exactness is read from M. A marginal projection is permitted only as a specialization of the coefficient ledger, so it cannot replace the orbit table or codon count table by a hidden enumeration.

The structural rows H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier lacks biological stochastic, quotient-codon, hidden-table, and unbounded-enumeration coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 4171.3 (Boundary orbit inventory coefficient determinacy)**.** *Every coefficient marginal exported by an accepted* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *packet is determined by the displayed orbit table O, codon count table K, coefficient ledger E, and marginal projection row M. Removing any coefficient row from E leaves a marginal query without a displayed source, and removing any orbit or codon count row leaves the corresponding coefficient without a displayed inventory witness.*

Proof. Read the NameCert obligations of Definition 4220.2. A coefficient query first exposes the orbit table O and codon count table K, then reads the coefficient ledger E.

The marginal projection row M is downstream of the ledger. It can specialize the displayed polynomial, but it cannot introduce a coefficient absent from E or reconstruct a missing source row absent from O or K.

Transport and replay preserve the same dependency order. Therefore a coefficient marginal is determined exactly by the displayed table and ledger rows, and deletion of any required row breaks the finite NameCert route. □

Falsifiable prediction: For $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, coefficient marginals are determined by the displayed orbit table, codon count table, coefficient ledger, and marginal projection rows, and a marginal read fails if any required coefficient row is removed.

Independence witness: partition, hash, errorcode, dna-window-six-boundary-tile, six-boundary-tile-coverage: the polynomial coefficient ledger is not bijective to these sibling carriers because it binds orbit tables, codon count tables, coefficient rows, marginal projections, transport, replay, provenance, and local naming, while the siblings supply partitioning, digest, error coding, boundary-tile, or coverage surfaces without this coefficient inventory.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet is generated from an orbit table, codon count table, coefficient ledger, marginal projection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4220.1, Definition 4220.2, and Definition 4220.3 seed the finite orbit-inventory polynomial certificate over boundary-tile and coverage rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological stochastic modeling, empirical code-table semantics, quotient codon equality, unbounded orbit enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-table admission, codon-count table admission, coefficient-ledger exactness, marginal-projection exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryOrbitInventoryPol

Chapter 4172

A Concrete Naming Certificate for $\mathsf{PrimitiveRecursive}^{\uparrow}$

Following Kleene's primitive recursive functions, the $\mathsf{PrimitiveRecursive}^{\uparrow}$ packet records finite recursion syntax and its local replay rows as BEDC data. It sits after the unary natural-number carrier of chapter 28, the list spine of chapter 36, the finite computability surface of chapter 199, the stream-window discipline of chapter 126, and the regular rational readback route of chapter 301. The chapter supplies a finite schedule substrate for later modulus and real-name consumers; it does not introduce minimization, partial recursive search, host functions, or unbounded choice.

Definition 4172.1 (PrimitiveRecursive carrier)**.** *A* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Z, S, \Pi, K, R, H, C, Q, N)$$

whose rows are read as follows:

1. *Z is the zero-function row.*
2. *S is the successor-function row.*
3. *Π is the finite family of projection rows, each with displayed arity and coordinate index.*
4. *K is the composition row, storing one outer code, its finite list of inner codes, and the arity ledger that makes the composition well typed.*
5. *R is the primitive-recursion scheme row, storing the base code, step code, displayed accumulator coordinate, and finite recursion spine over the* $\mathsf{Nat}^{\uparrow}$ *row.*
6. *H is componentwise* hsame *transport for zero, successor, projection, composition, recursion, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay for evaluating a finite syntax tree by structural recursion through the rows Z, S, Π, K, R.*
8. *Q is the* Pkg *provenance row over* $\mathsf{PrimitiveRecursive}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, S, \Pi, K, R, H, C, Q, N$. There is no coordinate for minimization, unbounded search, partiality, a host function graph, countable choice, or an external evaluator.

Lean4 ✓ `BEDC.Derived.PrimitiveRecursiveUp.TasteGate.PrimitiveRecursiveTasteGate single carrier alignment`

Theorem 4172.2 (PrimitiveRecursive NameCert obligations)**.** *For every accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier*

$$P = (Z, S, \Pi, K, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *surface consists exactly of carrier admission for the zero, successor, projection, composition, primitive-recursion, transport, replay, provenance, and naming rows; arity compatibility for* Π, K, R*; structural replay of the displayed syntax tree through* C*; componentwise transport through* H*; and provenance through* Q, N*. No obligation is stored in a minimization row, partial recursive search row, host-function graph, or unbounded-choice selector.*

Proof. Inspect the carrier row by row. The base constructors are Z, S, Π, and their arity information is part of the displayed finite rows rather than a host-side function space.

For composite syntax, the row K supplies the finite list of inner codes and the outer code, while R supplies the base and step codes for the primitive-recursion scheme. Induction on the displayed syntax tree checks that every replay request descends to one of Z, S, Π or to a strictly displayed subcode of K or R.

The structural rows close the local certificate. Transport through H preserves the same constructor rows, replay through C evaluates only the displayed syntax tree, and Q, N record provenance and naming. Since the packet has no coordinate for minimization, partial recursive search, host functions, or unbounded choice, the listed rows exhaust the NameCert obligations. □

Theorem 4172.3 (PrimitiveRecursive composition closure)**.** *If an accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *packet contains a composition row* K *whose outer code and finite list of inner codes are accepted by the same carrier, then the composed code is again read as a* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *row through* K, H, C, Q, N*. The closure is internal to the finite displayed syntax and exports no minimization, partiality, host-function graph, or unbounded-choice principle.*

Proof. Proceed by induction on the finite list of inner codes recorded in K. The empty-list boundary is the displayed arity check for the outer code, and the single-step case appends one already accepted inner row to the composition ledger.

At each induction step the replay row C evaluates the inner rows before the outer row, and transport through H preserves the same arity and constructor classifiers. The provenance row Q and local naming row N then repackage the resulting finite syntax as a $\mathsf{PrimitiveRecursive}^{\uparrow}$ row.

No unbounded search is used in the induction. The construction consumes only the displayed finite list in K and the accepted subcodes already present in the carrier, so it cannot introduce minimization, partial recursive search, host functions, or unbounded choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4221.1, Definition 4221.2, and Definition 4221.3 seed the finite recursion-syntax certificate for schedule and modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimization, partial recursive search, host functions, unbounded choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, arity compatibility, composition closure, primitive-recursion replay, transport, provenance, and local naming for the same finite recursion syntax.

Constructive story: The construction starts from zero, successor, and projection rows, closes finite composition through a displayed list spine, and closes primitive recursion by replaying the base and step codes over the Nat row under componentwise transport and local naming.

The $\mathsf{CauchyAcceleration}^{\uparrow}$ packet reads the StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces of chapter 126, chapter 291, chapter 301, and chapter 38 before exporting its accelerated completion-facing route.

Chapter 4173

A Concrete Naming Certificate for $\mathsf{CauchyAcceleration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyAcceleration}^{\uparrow}$ packet records the finite acceleration step by which an ordinary displayed Cauchy name is replaced by a cofinal regular rational name without changing the represented real. It sits after the criterion surfaces of chapter 392 and chapter 3576, the regularization route of chapter 376, the windowed name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The acceleration packet consumes Cauchy tail comparison as substantive finite-window and dyadic-tolerance data through chapter 301, chapter 126, chapter 291, and chapter 38 before any accelerated Real-facing read is exported. The accelerated cofinal regular-rational name is consumed by chapter 5721 as the completion-existence witness route, with the read still passing through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet makes acceleration a named finite ledger: it does not select an ambient subsequence, quotient a Cauchy representative, or appeal to host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4173.1 (Cauchy acceleration carrier)**.** *A* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite windows display the original rational Cauchy name.*
2. *T is the dyadic tolerance schedule, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, assigning the requested precision budget to each inspected window.*
3. *α is the cofinal acceleration map, stored as a finite monotone ledger saying which source window is read for each accelerated precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *proof row for the accelerated rational name, including the regularity inequalities obtained by reading S through α and T.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the accelerated regular name reaches the same Real-facing boundary as the source name.*
6. *H records componentwise* hsame *transport for the source stream, tolerance schedule, cofinality map, regular proof row, Real seal, provenance, and local naming row.*

7. *C records displayed* Cont *replay from an acceleration request through S, T, α, R, E and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyAcceleration}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient representative of the original Cauchy name, a choice-selected subsequence, an ambient completeness theorem, or a host-side equality oracle for real numbers.

Theorem 4173.2 (Cauchy acceleration NameCert obligations)**.** *For every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row S, dyadic tolerance schedule T, cofinal map α, regular rational proof row R, and Real seal row E;*
2. *monotone cofinality, requiring α to preserve request order and to eventually reach every inspected source precision window;*
3. *dyadic tolerance preservation, requiring each accelerated request to read the same displayed tolerance budget through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
4. *classifier transport, requiring H and C to preserve the source stream, tolerance schedule, cofinality ledger, regular proof row, and Real seal under componentwise* hsame *and displayed replay;*
5. *ledger exhaustion through P and N, excluding quotient representatives, choice-selected subsequences, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and host real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4222.1. The accepted packet exposes only the source stream S, tolerance schedule T, acceleration ledger α, regular readback R, Real seal E, and the structural rows.

Monotone cofinality is the public content of α. Since α is stored as a finite ledger over the inspected request window, each accelerated precision read names the source window it consumes and records that later requests do not move backward.

Dyadic tolerance preservation follows by reading T before the regular rational proof row. The $\mathsf{RegSeqRat}^{\uparrow}$ row R receives exactly the tolerance entries displayed by $\mathsf{DyadicRatCore}^{\uparrow}$, so the acceleration cannot import an unrecorded modulus or external precision oracle.

Classifier transport is componentwise through H and replayed through C. These rows preserve S, T, α, R, E without adding a second classifier surface.

Finally, P and N name precisely the displayed packet. Because the carrier contains no quotient representative, choice-selected subsequence, ambient completeness coordinate, or host real equality oracle, the five listed rows exhaust the local naming obligations. □

Theorem 4173.3 (Cauchy acceleration L10 handoff)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$ exports only the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $S, T, \alpha, R, E, H, C, P, N$. Consequently every L10 consumer receives a $\mathsf{RegSeqRat}^{\uparrow}$*-compatible and* $\mathsf{Real}^{\uparrow}$*-compatible accelerated name, while quotient representatives of the original Cauchy name, choice-selected subsequences, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness remain unavailable.*

Proof. Project the accepted carrier and apply Definition 4222.2. The only public data are $S, T, \alpha, R, E, H, C, P, N$.

The source read begins at the $\mathsf{StreamName}^{\uparrow}$ row S. The tolerance read then passes through T and α, so every accelerated precision request is attached to an explicit dyadic budget and a displayed source window.

The regular rational handoff is exactly R, and the Real-facing boundary is exactly E. Transport and replay through H and C preserve that route for downstream consumers.

Since P and N exhaust the packet and no forbidden coordinate appears in Definition 4222.1, the exported object is only the displayed accelerated name compatible with $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 4173.4 (Cauchy acceleration rectangular window route)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier* $A = (S, T, \alpha, R, E, H, C, P, N)$ *reuses the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows when producing the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the terminal* $\mathsf{Real}^{\uparrow}$ *handoff. The cofinal ledger* α *may refine the order of precision requests, but it cannot replace the displayed window route recorded by* S, T, H, C, P, N*.*

Proof. Project the rows $S, T, \alpha, R, E, H, C, P, N$. The source windows are recorded in S, and the dyadic tolerance entries are recorded in T before the accelerated regular rational row R or the Real-facing seal E is consumed.

Apply Definition 4222.2. The cofinality row α is an order ledger over displayed source windows, while dyadic tolerance preservation forces each accelerated request to read the same T row before it reaches R.

Now apply Definition 4222.3. Transport and replay through H and C preserve the route named by P and N, so no quotient representative, selected subsequence outside the ledger, or ambient completeness row is exported. □

Theorem 4173.5 (Cauchy acceleration cofinal-ledger determinism)**.** *Let*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

and

$$A' = (S, T, \alpha', R', E', H, C, P, N)$$

be accepted $\mathsf{CauchyAcceleration}^{\uparrow}$ *carriers sharing the same displayed source stream, tolerance schedule, structural replay rows, provenance rows, and local naming boundary through* S, T, H, C, P, N*. If the two cofinal ledgers* α *and* α' *choose the same source window for every inspected precision request, then their accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal follow the same finite route. In particular, neither ledger can replace the displayed source-window or tolerance route before* R, R' *and* E, E' *are read.*

Proof. Project both accepted carriers through Definition 4222.1. The shared rows expose the same $\mathsf{StreamName}^{\uparrow}$ source windows, the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule, and the same replay and provenance boundary.

Apply Definition 4222.2 to both packets. The only ledger freedom available before the accelerated readback is the cofinal row, and by hypothesis the displayed source window selected for each inspected request is identical in α and α'.

Now use Definition 4222.4. Since the rectangular source-window and tolerance route is fixed, the regular rational rows R, R' and the Real seals E, E' are reached through the same finite route. The packet therefore has no row through which a second ledger can substitute a hidden subsequence, quotient representative, or ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyAcceleration}^{\uparrow}$ packet is generated from a StreamName source row, a DyadicRatCore tolerance schedule, a finite cofinal acceleration ledger, a RegSeqRat proof row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4222.1, Definition 4222.2, and Definition 4222.3 seed the finite acceleration route from StreamName windows through dyadic tolerance schedules, cofinal acceleration ledgers, RegSeqRat readback, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy-name equality, choice-selected subsequences, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone cofinality, dyadic tolerance preservation, classifier transport, L10 handoff, and ledger exhaustion for BEDC.Derived.CauchyAccelerationUp.

Theorem 4173.6 (Cauchy acceleration sequence-space route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer reads the common-window classifier of* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *from chapter 406 before exporting the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff. The route passes through the regularization boundary of chapter 376: the source* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accelerated cofinal ledger, regular rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows are visible before any Real-facing consumer may inspect the acceleration result. No quotient representative, selected subsequence outside the displayed ledger, host equality, or ambient completeness row is exported.*

Theorem 4173.7 (Cauchy subsequence dependency route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal must first consume the rate-dominance surface of chapter 1814 and the fast-modulus surface of chapter 1781. The route is finite:* $\mathsf{CauchyRateDominance}^{\uparrow}$ *bounds the source schedule,* $\mathsf{FastCauchyModulus}^{\uparrow}$ *supplies the inspected acceleration window, and the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local* NameCert *rows carry the resulting readback. No consumer may replace this route by a chosen subsequence, quotient representative, ambient completion row, or host equality of Real names.*

Proof. Begin with the accepted carrier of Definition 4222.1. Its source stream row S, tolerance schedule T, cofinal ledger α, regular rational row R, and Real seal E are the only analytic rows available before transport, replay, provenance, and local naming.

The rate-dominance chapter supplies the comparison row that bounds which source schedule entries may be read by α. The fast-modulus chapter supplies the inspected window that turns those bounded reads into the accelerated tolerance route. Reading either row after the $\mathsf{RegSeqRat}^{\uparrow}$ readback would hide the schedule obligation rather than consume it.

Apply the rectangular window route of Definition 4222.4. It keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows in front of R and E, so the accelerated readback remains a displayed finite route. The carrier has no coordinate for a selected subsequence, quotient representative, ambient completion proof, or host Real equality, and the dependency route cannot export one. □

Theorem 4173.8 (Cauchy acceleration L10 four-face cofinality)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal preserves the ordered L10 four-face route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The cofinal acceleration ledger may refine which displayed source window is read for a requested precision, but it cannot commute, omit, or replace the $\mathsf{StreamName}^{\uparrow}$ *source face, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The resulting route remains finite and displayed, with no chosen subsequence outside the ledger, quotient representative, ambient completion row, host real equality, countable-choice schedule, Lean verification, or standard bridge.*

Proof. Start with Definition 4222.3. It names the only exported analytic route of an accepted carrier as the ordered sequence from the stream window through the dyadic tolerance ledger and regular rational readback to the Real-facing seal.

Next apply Definition 4222.4. The cofinal ledger α refines precision requests inside the displayed rectangle of source windows and dyadic tolerances, so it cannot move the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ rows ahead of the upstream faces.

Finally use the sibling dependency route of Definition 4222.7. The rate-dominance and fast-modulus surfaces are consumed before the accelerated readback is exported, and they leave the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces in order.

The carrier and dependency route have no row for a hidden subsequence, quotient representative, ambient completion proof, host Real equality, choice schedule, Lean verification, or bridge credential. Hence cofinal acceleration preserves exactly the displayed L10 four-face route. □

Theorem 4173.9 (Cauchy acceleration completion forward link)**.** *Every completion-facing consumer of* $\mathsf{CauchyAcceleration}^{\uparrow}$ *reads through chapter 1639, chapter 301, chapter 126, chapter 291, and chapter 38 before the accelerated route exports a terminal real boundary. The dependency is the finite route from source* $\mathsf{StreamName}^{\uparrow}$ *windows through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule and cofinal acceleration ledger into* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the Bishop regular Cauchy completion surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the displayed source windows. The accelerated route has no completion-facing input until the $\mathsf{StreamName}^{\uparrow}$ inspection windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule have been exposed.

Next read the regular rational and completion surfaces. The cofinal acceleration ledger feeds $\mathsf{RegSeqRat}^{\uparrow}$ readback and then passes through the Bishop regular Cauchy completion chapter before any terminal real boundary is visible.

Finally attach the $\mathsf{Real}^{\uparrow}$ seal. Since the route is finite and displayed at each sibling boundary, it exports no quotient completion, selected subsequence, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Chapter 4174

A Concrete Naming Certificate for $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$

Following Brouwer and Bishop–Bridges on monotone sequences in constructive analysis, the BrouwerMonotoneSeque packet records a monotone real-name boundary object through finite stream windows, regular rational readback, dyadic tolerance ledgers, and the terminal real-name seal. It sits after the Specker obstruction of chapter 1406, the regular-rational readback of chapter 301, the stream-window interface of chapter 126, the dyadic rational core of chapter 291, and the Real seal of chapter 38. The chapter gives the L10 Real and Completion frontier a named monotone-sequence carrier without claiming located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4174.1 (BrouwerMonotoneSequence carrier)**.** *A* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row, listing the inspected indices and their finite dependency order.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the inspected real-name entries.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed windows and monotonicity comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been displayed.*
6. *M is the monotonicity ledger, recording only finite adjacent-window comparison obligations among the displayed entries.*
7. *H records componentwise* hsame *transport for source indices, windows, dyadic ledgers, readback, Real seals, monotonicity, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a monotone-sequence request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets whose monotone-sequence request displays these rows in order. It has no coordinate for a located supremum, trichotomy decision, countable-choice selector, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BrouwerMonotoneSequenceUp.BrouwerMonotoneSequenceTasteGate single carrier alignment`

Theorem 4174.2 (BrouwerMonotoneSequence NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerMonotoneSeque}$ *carrier*

$$B = (S, W, D, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the inspected source sequence, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, finite monotonicity ledger, componentwise transport, displayed replay, package provenance, and local naming rows. No NameCert obligation may be discharged by a located supremum, trichotomy of real names, countable-choice schedule, selected limit, quotient stream representative, or ambient complete real line.*

Proof. Project the carrier of Definition 4223.1. The analytic rows are precisely the source schedule S, the finite windows W, the dyadic tolerance ledger D, the regular rational readback R, the Real seal E, and the finite monotonicity ledger M.

The monotonicity information is local to the displayed windows. Each comparison read in M consumes only the adjacent source entries already opened through S, W, D, R and does not add a global order decision for arbitrary real names.

The structural rows add no hidden analytic principle. Transport through H preserves the same displayed rows, replay through C follows the same finite route, and P, N record provenance and local naming for the packet.

Since the carrier has no row for located suprema, trichotomy, countable-choice schedules, selected limits, quotient stream representatives, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 4174.3 (BrouwerMonotoneSequence finite-window nonescape)**.** *Every consumer read of an accepted* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier reaches a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may consult the finite monotonicity ledger M for displayed adjacent windows, but it cannot escape to a located supremum, countable-choice selector, trichotomy oracle, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer read of the accepted carrier. The source row S exposes only the inspected indices, so the first real-name data available to the consumer are the finite $\mathsf{StreamName}^{\uparrow}$ windows in W.

The dyadic and regular-rational rows determine the only readback route. The tolerance ledger D bounds the displayed windows, and the row R packages those bounded observations as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The terminal seal is downstream of those rows. The $\mathsf{Real}^{\uparrow}$ row E is available only after S, W, D, R have been displayed, and the monotonicity ledger M can compare only the adjacent windows already present in the same finite packet.

Transport, replay, provenance, and local naming preserve this order through H, C, P, N. Therefore every consumer read remains inside the displayed finite-window route and cannot import a

located supremum, trichotomy oracle, choice schedule, selected limit, quotient stream representative, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ packet is generated from inspected source indices, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, a finite monotonicity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4223.1, Definition 4223.2, and Definition 4223.3 seed the constructive monotone-sequence boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window monotonicity, StreamName exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerMonotoneSequenceUp.

Chapter 4175

A Concrete Naming Certificate for $\mathsf{CauchySequenceCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet records the textbook sequence criterion as finite BEDC data. It ties a sequence source to the dyadic tolerance discipline of chapter 291, the stream-window naming discipline of chapter 126, the regular rational readback of chapter 301, and the public real seal of chapter 38. The chapter names only the finite route from tolerance data to regular-Cauchy and real consumers; it does not supply a choice-selected subsequence, quotient equality of limits, host completion, or ambient completeness principle.

Definition 4175.1 (CauchySequenceCriterion carrier)**.** *A* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the sequence-source row, displaying the named sequence and the finite request interface through which a downstream consumer asks for terms.*
2. *M is the modulus-schedule row, displaying the finite schedule that turns a requested dyadic tolerance into a stage bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row, recording the positive dyadic thresholds, comparisons, and tolerance refinements read by the schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the sequence after the schedule row has supplied their stage bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row reached from A, M, D, S.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-Cauchy readback has been displayed.*
7. *H records componentwise* hsame *transport for the sequence source, schedule, dyadic tolerance ledger, stream windows, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a criterion request through $A, M, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose criterion consumer first displays the sequence source, modulus schedule, dyadic tolerance ledger, stream-window row, regular-Cauchy readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCriterionUp.CauchySequenceCriterionTasteGate single carrier alignment`

Theorem 4175.2 (CauchySequenceCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier*

$$Q = (A, M, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *surface consists exactly of sequence-source admission, modulus-schedule admission, dyadic tolerance-ledger exposure, finite stream-window exposure, regular-Cauchy readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of* Q*. No obligation may be discharged by a choice-selected subsequence, quotient equality of limits, host completion, ambient completeness, or an unbounded open-cover argument.*

Proof. Project the carrier of Definition 4224.1. Its analytic rows are precisely the sequence source A, modulus schedule M, dyadic tolerance ledger D, stream windows S, regular-Cauchy readback R, and Real seal E.

The tolerance rows determine the access order. A criterion consumer reads M before it can turn a dyadic request from D into a finite stage bound, and it reads S only through the finite windows exposed by that bound.

The readback rows add no external completion object. The row R repacks the displayed tolerance and window data as regular-Cauchy readback, and E seals that readback as the public real-facing output.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate of Q stores a selected subsequence, quotient limit equality, host completion, ambient completeness witness, or open-cover argument, the displayed rows exhaust the NameCert obligations. □

Theorem 4175.3 (CauchySequenceCriterion L10 handoff)**.** *Every L10 consumer read of an accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through the sequence source, modulus schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the sequence source A and modulus schedule M. They expose the named sequence and the finite stage bounds required by any dyadic tolerance request.

Next read the dyadic tolerance ledger D and stream-window row S. The ledger supplies the positive thresholds and comparisons, while the stream row supplies only the finite windows named by the schedule.

Attach the regular-Cauchy readback row R. It is the unique displayed route by which the dyadic-window data becomes a regular rational sequence consumer.

The Real seal closes the route. The row E is reached only after R, and the structural rows H, C, P, N carry the same finite data through transport, replay, provenance, and local naming. Hence the L10 handoff exposes a constructive sequence criterion rather than a selected subsequence, quotient limit, host completion, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet is generated from a sequence source, modulus schedule, dyadic tolerance ledger, stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4224.1, Definition 4224.2, and Definition 4224.3 seed the L10 route from dyadic sequence criteria to regular-Cauchy and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected subsequences, quotient equality of limits, host completion, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus schedule exposure, dyadic tolerance-ledger exactness, stream-window handoff, RegSeqRat/Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCriterionUp.

$\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ routes pointwise locator induction through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38; the child file carries the theorem and closurestatus surface.

Chapter 4176

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 4176.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4176.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 4176.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 4176.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4176.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 4176.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 4176.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4177

A Concrete Naming Certificate for $\mathsf{FrinkMetrization}^{\uparrow}$

Following Frink and Kelley, the $\mathsf{FrinkMetrization}^{\uparrow}$ packet records the finite BEDC route from a uniform entourage basis to a pseudometric-facing and metric-facing distance certificate. It sits after the uniform-source rows of chapter 339, the pseudometric carrier of chapter 1614, the separated metric boundary of chapter 57, the topology compatibility row of chapter 88, the dyadic tolerance ledger of chapter 291, and the finite window interface of chapter 126. The packet names only the constructive handoff from nested entourage data to displayed distance reads; it does not form quotient uniformities, select representatives, assert ambient metrizability, or import choice.

Definition 4177.1 (FrinkMetrization carrier)**.** *A* $\mathsf{FrinkMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, Q, D, P, M, T, H, C, G, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row. It names the carrier surface and the finite entourage requests from which the metrization route starts.*
2. *Q is the nested entourage chain. It records the finite refinement ladder used to compare two source points at dyadic scales.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. It translates the refinement level into displayed dyadic radii and triangle budgets.*
4. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *handoff row. It receives the finite entourage chain and exports the displayed pre-separated distance route.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *compatibility row, available only after the separated boundary has accepted the zero-distance route supplied through P.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *compatibility row. It records that the metric-facing balls and the original uniform entourage reads are compared through the same finite windows.*
7. *H records componentwise* hsame *transport for the uniform source, chain, dyadic ledger, pseudometric handoff, metric compatibility, topology compatibility, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform entourage request through the nested chain, dyadic ledger, pseudometric handoff, metric row, topology row, provenance, and local naming row.*

9. *G is the* Pkg *provenance over* FrinkMetrization$^{\uparrow}$, UniformSpace$^{\uparrow}$, Pseudometric$^{\uparrow}$, Met$^{\uparrow}$, Topology$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, or an unbounded distance oracle.

Lean4 ✓ `BEDC.Derived.FrinkMetrizationUp.FrinkMetrizationTasteGate single carrier alignment`

Theorem 4177.2 (FrinkMetrization NameCert obligations)**.** *For every accepted* FrinkMetrization$^{\uparrow}$ *carrier $F = (U, Q, D, P, M, T, H, C, G, N)$, the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform source, nested entourage chain, dyadic radius ledger, pseudometric handoff, metric compatibility row, topology compatibility row, transport, continuation, provenance, and local naming row;*
2. *nested-chain readability, requiring every distance request to start at the* UniformSpace$^{\uparrow}$ *row U and pass through the finite refinement chain Q before any dyadic radius in D is read;*
3. *pseudometric handoff, requiring the displayed distance route to be exported through P before the metric row M or topology row T can be consumed;*
4. *compatibility transport, requiring H and C to preserve the same finite* StreamName$^{\uparrow}$ *windows and* DyadicRatCore$^{\uparrow}$ *radius ledger across uniform, pseudometric, metric, and topology-facing consumers;*
5. *provenance and refusal, requiring G and N to name only this finite handoff and to refuse quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, and unbounded distance oracles.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4226.1. The projection exposes exactly the uniform source, nested chain, dyadic ledger, pseudometric handoff, metric and topology compatibility rows, and the structural rows H, C, G, N.

The nested-chain obligation is forced by row order. A distance request starts at the uniform row U, enters the refinement ladder Q, and only then reads the dyadic radius ledger D.

The pseudometric handoff is the only distance-export step. Row P supplies the pre-separated displayed distance route, while M and T can compare it with metric and topology-facing consumers only after the same finite windows and radii have been replayed.

Rows H, C, G, N transport, replay, source, and name this route. Since the tuple contains no quotient uniformity, selected representative, ambient metrizability, countable choice, host topology equality, or unbounded distance coordinate, the local NameCert cannot export those principles. □

Closure status for NameCert$_{\text{FrinkMetrization}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, seed)

Scope closed: Definition 4226.1 and Definition 4226.2 seed the finite Frink-style handoff from uniform entourage chains to pseudometric, metric, topology, dyadic, and stream-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, unbounded distance oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-chain readability, dyadic radius exactness, pseudometric handoff, metric and topology compatibility, transport, continuation, provenance, and ledger refusal for BEDC.Derived.FrinkMetrizationUp.

Constructive story: A FrinkMetrization$^{\uparrow}$ packet is generated from a UniformSpace source row, a nested entourage chain, a DyadicRatCore radius ledger, a Pseudometric handoff, Metric and Topology compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

NestedShrinkInterval$^{\uparrow}$ reads its compact interval dependency through chapter 5553 and the child `namecert_construction.tex` route before exposing its shrinking-diameter Real$^{\uparrow}$ seal.

Chapter 4178

A Concrete Naming Certificate for $\mathsf{NestedShrinkInterval}^{\uparrow}$

Following Cantor's nested-interval argument and Bishop–Bridges regular real analysis, the $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet records the finite BEDC carrier surface for a descending located interval route whose displayed diameters shrink to the requested tolerance. It sits between the nested-closed interval carrier of **??**, the located endpoint row of chapter 394, the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names only the finite shrinking-interval convergence certificate; it does not claim open-cover compactness, trichotomy, quotient reals, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4178.1 (Nested shrink interval carrier)**.** *A* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (J, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row, exposing the retained descending closed located interval family and its containment witnesses.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row for the inspected interval windows, including lower and upper endpoint location data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger recording endpoint comparisons, diameter estimates, and tolerance requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect interval endpoints and diameter rows at displayed stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same stream windows and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the nested interval, located endpoint, dyadic shrink, stream-window, and regular readback rows are visible.*
7. *H records componentwise* hsame *transport for J, L, D, S, R, E and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$J, L \longmapsto D, S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{NestedShrinkInterval}^{\uparrow}$, $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $J, L, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient-real equality, a selected intersection point, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4178.2 (Nested shrink interval NameCert obligations)**.** *For every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier*

$$K = (J, L, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row,* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same displayed rows; diameter-to-tolerance exactness through D; finite-window readback through S and R; Real-seal handoff through E; and ledger nonescape for the excluded compactness, trichotomy, quotient, selection, choice, and ambient-completeness coordinates.*

Proof. Project the carrier of Definition 4227.1. The projection exposes exactly the nested closed interval source, located endpoint row, dyadic shrink ledger, finite windows, regular readback, real seal, and structural rows.

Classifier stability is componentwise on those displayed rows. The shrink estimate is not an external metric fact: it is the D row, read together with the endpoint row L and the stream schedule S before regular readback is available.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, every accepted NameCert obligation is one of the rows listed above. □

Theorem 4178.3 (Nested shrink interval diameter route)**.** *Let $K = (J, L, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. Every completion-facing request for the shrinking interval route first reads the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source J and* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row L, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger D and* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, and only then opens the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and* $\mathsf{Real}^{\uparrow}$ *seal E. The route exports diameter-to-tolerance convergence as displayed finite ledger data, not as open-cover compactness, trichotomy, quotient-real equality, countable choice, a selected intersection point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing request and project the accepted carrier. The only interval source is J, and the only endpoint data are the located rows in L.

The diameter request is then forced through D. Its dyadic comparisons and tolerance rows are inspected through the finite schedule S, so a consumer cannot replace the shrinking route by an unrecorded compactness or completeness theorem.

After the finite windows are displayed, R packages the same observations as regular rational readback and E supplies the terminal real seal. The remaining rows H, C, P, N preserve the route and add no excluded coordinate. □

Theorem 4178.4 (Nested shrink interval Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet receives the seal only through the displayed route*

$$\mathsf{NestedClosedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff is stable under the componentwise hsame *transport and displayed* Cont *replay rows* H, C*, and it is sourced and named only by* P, N*.*

Proof. Apply Definition 4227.3. A Real-facing consumer reaches E only after the nested interval source, located endpoint row, dyadic shrink ledger, stream windows, and regular readback have all been displayed.

Transport through H preserves the row roles, and replay through C preserves their order. Provenance and local naming through P, N attach the handoff to the packet without adding analytic content.

Thus the Real seal cannot escape to a selected point, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient real completeness. □

Theorem 4178.5 (Nested shrink interval regular-Cauchy uniqueness route)**.** *Let* $K = (J, L, D, S, R, E, H, C, P, N)$ *and* $K' = (J', L', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packets. Suppose their nested-source rows, located endpoint rows, dyadic shrink ledgers, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise related by the displayed* hsame *transport. Then the two packets reach the same* $\mathsf{Real}^{\uparrow}$*-facing seal route only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *uniqueness surface displayed by* R, R' *before Real sealing. The route does not identify selected intersection points, quotient real representatives, or ambient host real equality.*

Proof. Project both carriers. The first carrier exposes $J, L, D, S, R, E, H, C, P, N$, and the second exposes $J', L', D', S', R', E', H', C', P', N'$.

Apply the diameter route of Definition 4227.3 to both packets. Each read is forced through its nested source, located endpoints, dyadic shrink ledger, finite stream windows, and regular rational readback before a Real-facing seal is visible.

Use the assumed componentwise hsame transport on J, L, D, S, R and J', L', D', S', R'. The endpoint comparison therefore lives on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, because the preceding rows have already fixed the same nested-source, located-endpoint, dyadic, and stream-window route.

Finally apply the Real-seal handoff of Definition 4227.4 to both packets. It permits E and E' to be read only after the regular readback rows are displayed, so the uniqueness comparison is a $\mathsf{RegSeqRat}^{\uparrow}$ route before Real sealing and not a selected intersection point, quotient real equality, or ambient completeness assertion. □

Theorem 4178.6 (Cantor intersection nested-shrink route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer that reads a nested shrinking interval first consumes the* $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows exposed by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. The downstream route supplies the nested-shrink boundary used by chapter 6999 and the Cantor-intersection surface of chapter 1421.*

The route is a forward lattice edge only. It does not select an intersection point, assert open-cover compactness, use countable choice, identify quotient real names, or provide ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4227.1. The exposed rows are the nested closed source J, located endpoints L, dyadic shrink ledger D, finite stream windows S, regular readback R, real seal E, and the structural rows H, C, P, N.

Apply Definition 4227.4. Any Real-facing downstream read reaches E only after the nested, located, dyadic, stream, and regular-rational rows have been displayed.

The Cantor diagonal and Cantor-intersection chapters consume exactly this kind of finite shrinking-interval boundary before their own diagonal or intersection rows are read. Since the

carrier contains no selected point, open-cover compactness row, choice schedule, quotient equality, or ambient completeness coordinate, the downstream route remains a finite dependency edge. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet is generated from a NestedClosedInterval source row, LocatedInterval endpoint rows, a DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4227.1, Definition 4227.2, Definition 4227.3, and Definition 4227.4 seed the finite L10 shrinking-interval convergence route through nested closed intervals, located endpoint rows, dyadic diameter ledgers, stream windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient reals, selected intersection points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, diameter-to-tolerance exactness, finite-window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedShrinkIntervalUp.

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet consumes the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 before a completion-facing reader treats shrinking finite windows as a diagonal Cauchy subsequence witness. The nested source, located endpoint, dyadic diameter, and $\mathsf{StreamName}^{\uparrow}$ rows still supply the finite shrinking windows; $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies only the displayed diagonal handoff needed to pass those windows toward $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The route exports no detached selector, quotient-real equality, countable-choice schedule, selected intersection point, or ambient completeness coordinate. The shrinking interval diameter route is consumed through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38. Thus a downstream L10 consumer reads $\mathsf{NestedShrinkInterval}^{\uparrow}$ only after the finite diameter ledger has been repacked as a $\mathsf{RegSeqRat}^{\uparrow}$ route and sealed as a $\mathsf{Real}^{\uparrow}$ boundary, not as an ambient completeness or selected intersection-point claim. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ after the shrinking-diameter ledger has been repacked through the $\mathsf{StreamName}^{\uparrow}$ observation schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal route. This sibling dependency records the completion-criterion consumer path of chapter 1476: a nested shrink interval can be read as completion evidence only after its finite diameter data has passed through the same stream, regular sequence, dyadic tolerance, and Real seal rows.

Theorem 4178.7 (Nested shrink interval Cauchy completion consumer)**.** *Every completion-facing read of* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *consumes* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *only after the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are visible. Thus the dependency on chapter 1476 is a theorem-level consumer edge through chapter 126 and chapter 301, not an appeal to an undisplayed completion object.*

Proof. Read a completion-facing request for a nested shrink interval. The route first displays the finite observation schedule and regular rational readback rows, so the consumer sees only finite windows and readback entries.

The dyadic tolerance ledger and terminal Real seal are then exposed before the completion criterion is consumed. Since the criterion is reached after those rows, the dependency records a reusable finite route rather than a hidden selected limit or ambient completeness principle. □

Theorem 4178.8 (Nested shrink interval rational locator route)**.** *Every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *route whose endpoint localization is read through rational endpoints first consumes the rational locator carrier of chapter 4202. Only after that locator row is displayed may the consumer read the*

nested shrink rows and then the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows. The route exports no selected completed point and no Heine–Borel compactness principle.*

Proof. Read the accepted packet through the carrier theorem of chapter 4227. The only endpoint localization rows available to the nested shrink route are finite displayed rows, so a rational-endpoint consumer must obtain its rational interval locator row before the shrink ledger can be interpreted.

The sibling chapter chapter 4202 supplies exactly that locator surface. Once it is consumed, the nested shrink packet may proceed through its displayed shrink ledger, finite windows, regular rational readback, and terminal real seal.

No selected completed point or Heine–Borel compactness row is present in either carrier. The dependency route therefore records only the rational locator read before the nested shrink and $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff rows are read. □

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ route consumes chapter 4202 for rational endpoint bracketing and chapter 301 for regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Thus the nested shrink interval dependency lattice reads locator cells first, then the shrinking-window ledger, then the regular readback route, not a selected intersection point or ambient completeness witness. $\mathsf{NestedShrinkInterval}^{\uparrow}$ exposes a completion-facing interval read only after the packet has consumed the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ diagonal row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational route, the terminal $\mathsf{Real}^{\uparrow}$ seal, the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ admission row, and the $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint-bracketing row. The public read is therefore a finite sibling route through diagonal selection, regular sequence readback, Real sealing, completion admission, and rational interval location; it exports no selected intersection point, quotient-real equality, countable choice, or ambient completeness principle. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the completion criterion surface of chapter 1476 before any interval-limit seal is visible. The shrinking-diameter ledger must first pass through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, regular rational readback, and the same Cauchy-completion admission route; only then may a downstream L10 consumer inspect the $\mathsf{Real}^{\uparrow}$ boundary, and the packet still exports no selected intersection point, quotient-real equality, countable choice, or ambient completion. The $\mathsf{NestedShrinkInterval}^{\uparrow}$ carrier depends on the substantive sibling lattice through chapter 320 for the nested interval spine, chapter 301 for regular rational readback, and chapter 38 for the terminal seal. The rational locator and Cauchy-completion routes already represented by the child files above supply the endpoint-bracketing and completion-admission rows, so the nested-shrink packet is read as a finite dependency route rather than a selected intersection point, quotient-real equality, countable choice, or ambient completeness principle.

Theorem 4178.9 (Nested shrink interval compact interval route)**.** *Every compact-interval consumer of an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet must read the compact real interval sibling of chapter 5553 and the nested interval convergence sibling of chapter 2057 through the same finite endpoint, diameter,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows displayed by the shrinking packet. The compact interval route supplies endpoint and finite-net context; the nested convergence route supplies the shrinking-window handoff; neither route adds an open-cover compactness theorem, selected limit point, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier of Definition 4227.1. The visible analytic rows are the nested source, located endpoints, dyadic diameter ledger, finite stream windows, regular rational readback, and Real seal, followed by componentwise transport, replay, provenance, and local naming.

Use the compact real interval packet of chapter 5553 only for the endpoint and finite-net context. Its compact rows are read before the shrinking packet is exported as an interval-local dependency, so the shrink route never obtains an unlisted open-cover or subcover selector.

Use the nested interval convergence packet of chapter 2057 for the convergence-facing handoff. That sibling reads finite endpoint windows, shrinking ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ sealing before a completion consumer may inspect the route.

The structural rows hsame, Cont, Pkg, and NameCert keep the two sibling reads attached to the same displayed packet. Hence a compact-interval dependency edge from $\mathsf{NestedShrinkInterval}^{\uparrow}$ is exactly the stated finite route and not a hidden compactness, quotient, choice, or completeness principle. □

Theorem 4178.10 (NestedShrinkInterval Cantor downstream route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer of a* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet reads the nested source only through nested closed intervals, located endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter witnesses,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The downstream route exports a Cantor diagonal or intersection-facing interval certificate, but it does not select a limit point, assert ambient completeness, identify quotient-real names, or add a host compactness principle outside the displayed rows.*

Proof. Project the nested-shrink carrier through the child routes named by the hub. The compact-interval and rational-locator rows provide nested closed intervals and located endpoints, while the dyadic and StreamName rows expose only finite diameter and window observations.

The Cantor-facing consumer then passes through the RegSeqRat and Real rows already used by the completion and sibling routes. Since these rows carry finite readback and terminal naming rather than a chosen point, the consumer obtains only the displayed interval certificate.

No carrier coordinate names a selected limit point, host completeness theorem, quotient equality, or ambient compactness object. Thus $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ consume $\mathsf{NestedShrinkInterval}^{\uparrow}$ only through the finite interval route stated above. □

$\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ reads the displayed uniform-modulus handoff of chapter 369 from regular Cauchy source names to regular Cauchy target names through the child packet below.

Chapter 4179

A Concrete Naming Certificate for RegularCauchyUniformContinuity$^{\uparrow}$

Following Bishop and Bridges, uniform continuity on displayed constructive real names is read here as a finite regular-Cauchy handoff rather than as host function equality or a completed-real theorem. The RegularCauchyUniformContinuity$^{\uparrow}$ packet records the standard fact that a displayed uniform modulus carries regular Cauchy names to regular Cauchy names. Its dependency surface is the finite StreamName$^{\uparrow}$ window interface of chapter 126, the regular-rational row of chapter 301, the pointwise map rows of chapter 130, the tolerance rows of chapter 369, and the terminal real-name seal of chapter 38.

Definition 4179.1 (Regular Cauchy uniform-continuity carrier)**.** *A* RegularCauchyUniformContinuity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, W, R, M, D, T, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* StreamName$^{\uparrow}$ *finite-window family opened for the regular Cauchy source name.*
2. *W records the target-window schedule requested by the regular Cauchy conclusion.*
3. *R is the source* RegSeqRat$^{\uparrow}$ *regularity row, including the rational tail bounds read on the windows in S.*
4. *M is the* ContinuousMap$^{\uparrow}$ *pointwise map row, read only through displayed source and target observations.*
5. *D is the* UniformModulus$^{\uparrow}$ *tolerance row that turns each target precision request into a source precision budget.*
6. *T is the target* RegSeqRat$^{\uparrow}$ *regularity row obtained by replaying D against R through M on the windows S, W.*
7. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the target regularity row T is displayed.*
8. *H records componentwise* hsame *transport for S, W, R, M, D, T, E and for the structural replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a target tolerance request through D, then S, R, M, W, then the target regularity row T, and finally the seal row E.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, W, R, M, D, T, E, H, C, P, N$ and the displayed replay order. It has no coordinate for host function equality, quotient streams, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 4179.2 (Regular Cauchy uniform-continuity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier $U = (S, W, R, M, D, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source stream windows, target window schedule, source regular-rational row, pointwise map row, uniform tolerance row, target regular-rational row, Real seal, transport, replay, provenance, and local naming row;*
2. *source regularity exactness, requiring every source observation to be read through S and R before the map row is inspected;*
3. *uniform tolerance exactness, requiring each target precision request to obtain its source budget only through D;*
4. *pointwise replay exactness, requiring the map row M to be applied only to source observations admitted by S, R, D;*
5. *target regularity handoff, requiring the target* $\mathsf{RegSeqRat}^{\uparrow}$ *row T to be produced before the* $\mathsf{Real}^{\uparrow}$ *seal E is exposed;*
6. *transport and continuation preservation, requiring H and C to preserve the displayed coordinates and their replay order;*
7. *provenance exhaustion, requiring P to name only the sibling chapters and kernel rows listed in Definition 4179.1;*
8. *local naming, requiring N to name exactly the* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *certificate row.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4179.1. The accepted packet exposes exactly $S, W, R, M, D, T, E, H, C, P, N$, so every local obligation must be read from that finite inventory.

The source side is exhausted by S and R. The stream row opens only finite windows, and the regular-rational row supplies the tail bounds on those windows before any map or target read is available.

The tolerance step is exhausted by D. A target precision request first reads the uniform-modulus row, which supplies the source precision budget used to choose the source observations.

The pointwise map step is then replayed through M. Since M is the only map coordinate and its inputs have already been confined to S, R, D, no host function equality or external continuity principle can enter this step.

The target handoff is recorded by T and sealed by E. The continuation row places T before E, while the transport row preserves the same coordinates under hsame.

Finally inspect P and N. The provenance row names only the sibling surfaces listed in the carrier, and the local naming row names only this certificate. Hence the eight obligations exhaust the NameCert surface. □

Theorem 4179.3 (Regular Cauchy uniform-continuity handoff)**.** *Let* $U = (S, W, R, M, D, T, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier. Every target* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by* U *factors through the finite route*

$$S, R, D, M, W, T.$$

Consequently a displayed uniform modulus carries the source regular Cauchy name to the target regular Cauchy name before any $\mathsf{Real}^{\uparrow}$ *seal or completion-facing consumer can read the packet.*

Proof. Start with a target precision request. The uniform-modulus row D converts that request into a source precision budget, and the source budget selects only the finite source windows named by S.

Read the source regularity row next. The row R supplies the regular Cauchy tail control on those windows, so the pointwise map row receives only displayed source observations.

Apply the pointwise map row M through the continuation route. The target-window schedule W records the finite target observations required by the regularity conclusion.

The target regularity row T is the only target $\mathsf{RegSeqRat}^{\uparrow}$ coordinate in the carrier. Since the continuation order places T after S, R, D, M, W and before the seal row E, every target regular Cauchy read factors through the displayed finite route. □

Theorem 4179.4 (Regular Cauchy uniform-continuity window composition)**.** *Let* $U = (S, W, R, M, D, T, E, H, C,$ *be an accepted* $\mathsf{RegularCauchyUniformContinuity}^{\uparrow}$ *carrier. Suppose two target precision requests are displayed in the target-window schedule* W*, and suppose their* $\mathsf{UniformModulus}^{\uparrow}$ *rows refine through the same tolerance row* D*. Then the two requests compose to one source-window budget over* S, R *before the pointwise map row* M *is read.*

The two resulting target rows both factor through the same continuation route

$$S, R, D, M, W, T, E.$$

Thus the composed target read is a finite window-stability statement for the displayed carrier only: it exports no host function equality, quotient stream, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the carrier and the local NameCert obligations from Definition 4179.1 and Definition 4179.2. The only rows available for target precision requests are the schedule W, the tolerance row D, the source windows S, the source regularity row R, the pointwise map row M, the target regularity row T, and the seal row E.

For the two requested target precisions, use the uniform tolerance exactness obligation twice. Because both requests refine through the same D row, their source precision needs are read as one finite source-window budget on S with regular-rational control supplied by R.

Apply Definition 4179.3 to each resulting target read. The handoff forces both reads through S, R, D, M, W, T, so the target rows factor through W, T before either can reach the seal row E.

Finally inspect the carrier inventory. There is no coordinate for host function equality, quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the composed route is exactly the displayed finite window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4179.1, Definition 4179.2, and Definition 4179.3 seed the finite route by which displayed uniform moduli carry regular Cauchy names to regular Cauchy names over StreamName windows, RegSeqRat rows, ContinuousMap pointwise rows, UniformModulus tolerance rows, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient streams, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source regularity exactness, uniform tolerance exactness, pointwise replay exactness, target regularity handoff, transport and continuation preservation, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyUniformContinuityUp.

Constructive story: A RegularCauchyUniformContinuity$^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat source and target rows, a ContinuousMap pointwise row, a UniformModulus tolerance row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The RegularCauchyUniformContinuity$^{\uparrow}$ route consumes RegSeqRat$^{\uparrow}$ readbacks through chapter 301, ContinuousMap$^{\uparrow}$ modulus-facing pointwise rows through chapter 130, and the UniformModulus$^{\uparrow}$ handoff through chapter 369. This child route exports no quotient stream, selected limit, host continuity, or ambient Real$^{\uparrow}$ completeness.

Chapter 4180

A Concrete Naming Certificate for $\mathsf{HenstockKurzweil}^{\uparrow}$

Following Henstock and Kurzweil, the $\mathsf{HenstockKurzweil}^{\uparrow}$ packet records a gauge-integral seed route in which tagged Riemann windows are controlled by a displayed gauge before any real-valued integral read is exported. It sits beside the gauge-integral packet of chapter 1999, the Riemann-integral packet of chapter 1526, the Riemann-sum gauge packet of chapter 4115, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, and the dyadic endpoint ledger of chapter 291. The chapter names the finite gauge-window carrier used by this integration route; it does not claim Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4180.1 (HenstockKurzweil carrier)**.** *A* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, I, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *source row naming the accepted gauge-integral boundary.*
2. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *comparison row that keeps the integral read tied to tagged mesh data.*
3. *S is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row carrying the finite tagged-sum window and gauge compatibility test.*
4. *W is the displayed gauge-window row, recording the tag, cell, and local radius data used by S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger for the cells in the displayed window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the finite window and dyadic tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, I, S, W, D, R have been replayed.*
8. *H records componentwise* hsame *transport for the gauge, integral, sum, window, dyadic, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a Henstock–Kurzweil request through $G, I, S, W, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HenstockKurzweil}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, I, S, W, D, R, E, H, C, P, N$. There is no coordinate for a measure space, an open-cover compactness theorem, a chosen refinement family, a selected infinite sum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.HenstockKurzweilUp.HenstockKurzweilTasteGate single carrier alignment`

Theorem 4180.2 (HenstockKurzweil NameCert obligations)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier $K = (G, I, S, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$ *surface consists exactly of carrier admission, GaugeIntegral exposure through G, RiemannIntegral comparison through I, RiemannSumGauge window exposure through S, gauge-window exactness through W, DyadicRatCore tolerance exposure through D, RegSeqRat readback through R, Real seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No obligation may be discharged by Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, host integrand equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4180.1. The displayed rows are exactly $G, I, S, W, D, R, E, H, C, P, N$, so carrier admission exposes only gauge, integral-comparison, tagged-sum, window, dyadic, readback, seal, transport, replay, provenance, and naming data.

The integration side starts with G, I, S, W. The gauge-integral source row G and Riemann-integral comparison row I are read only through the finite tagged-sum gauge row S and its displayed window W, so the packet has one gauge-window route rather than a detached refinement principle.

Next read the numerical handoff. The dyadic ledger D supplies endpoint and tolerance data for the same window, R records the regular rational readback, and E is available only after those rows have been replayed.

Finally the structural rows H, C, P, N preserve the same route. Since measure data, open-cover compactness, selected infinite sums, countable choice, host integrand equality, and ambient completeness are absent from the carrier, the listed rows exhaust the local surface. □

Theorem 4180.3 (HenstockKurzweil gauge-window route)**.** *Every completion-facing or Real-facing read of an accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{GaugeIntegral}^{\uparrow} \longmapsto \mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The gauge-window row W is read before the dyadic tolerance ledger and the real seal, and the route cannot bypass $G, I, S, W, D, R, E, H, C, P, N$.

Proof. Begin with an accepted carrier read. The carrier exposes G, S, W before any numerical endpoint or real seal can be inspected, so the request has already passed through a displayed gauge-window row.

Replay the continuation row C. The read moves from the gauge source G through the tagged-sum gauge row S and the Riemann comparison row I, then through the dyadic ledger D and regular rational readback R before it reaches the seal E.

The structural rows H, P, N do not add another integration object. They preserve the same finite route, and no excluded measure, compactness, choice, selected-sum, or completeness coordinate is available in the packet. □

Theorem 4180.4 (HenstockKurzweil gauge-window exactness)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier $K = (G, I, S, W, D, R, E, H, C, P, N)$, the displayed gauge-window row W is projected after the gauge-integral source, the Riemann-integral comparison, and the tagged-sum gauge rows G, I, S,*

and before any $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *read through* D, R, E. *No dyadic endpoint, regular rational readback, or real seal can be inspected without first replaying the same* G, I, S, W *window.*

Proof. Project the carrier of Definition 4180.1. The row order exposes G, I, S, W as the complete gauge-window prefix, with W recording the tag, cell, and local radius data used by the tagged-sum row S.

Now inspect the numerical rows. The dyadic ledger D, regular rational readback R, and terminal real seal E are downstream of that prefix, so each numerical read has already passed through the displayed window. Since the carrier has no independent dyadic, rational, or real coordinate, it cannot bypass W. □

Theorem 4180.5 (HenstockKurzweil RiemannSumGauge exposure)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier* $K = (G, I, S, W, D, R, E, H, C, P, N)$, *the row* S *is the only tagged-sum exposure route. Any completion-facing, dyadic-facing, regular-readback-facing, or real-facing use of tagged Riemann data must factor through* S *together with its displayed window* W *and the structural rows* H, C, P, N.

Proof. Read the carrier rows G, I, S, W. The only row tagged as $\mathsf{RiemannSumGauge}^{\uparrow}$ is S, and the only window attached to tagged-sum data is W.

The downstream rows D, R, E receive endpoint, tolerance, readback, and seal data from the same tagged-sum route. The structural rows H, C, P, N transport, replay, source, and name the carrier, but they do not introduce a second tagged-sum row. Hence every tagged Riemann use remains exposed through S. □

Theorem 4180.6 (HenstockKurzweil Real seal nonescape)**.** *For every accepted* $\mathsf{HenstockKurzweil}^{\uparrow}$ *carrier* $K = (G, I, S, W, D, R, E, H, C, P, N)$, *the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *cannot be reached without replaying* G, I, S, W, D, R. *In particular,* E *exposes no route through Lebesgue measure, open-cover compactness, selected infinite sums, countable choice, host integrand equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Replay the continuation row C through the carrier projection. The real-facing row appears as E, after the gauge-integral source G, Riemann comparison I, tagged-sum gauge row S, displayed window W, dyadic ledger D, and regular rational readback R.

The remaining rows H, P, N preserve that same route by transport, provenance, and local naming. They cannot supply a measure object, compactness theorem, selected sum, choice principle, host equality, or ambient completeness field, because none of those coordinates occur in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HenstockKurzweil}^{\uparrow}$ read factors through $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}$.

Constructive story: A $\mathsf{HenstockKurzweil}^{\uparrow}$ packet is generated from a GaugeIntegral row, RiemannIntegral comparison row, RiemannSumGauge tagged-window row, gauge-window ledger, DyadicRatCore endpoint and tolerance rows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenstockKurzweil}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 4180.1, Definition 4180.2, Definition 4180.3, Definition 4180.4, Definition 4180.5, and Definition 4180.6 close the finite gauge-window obligation surface over GaugeIntegralUp, RiemannIntegralUp, RiemannSumGaugeUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{HenstockKurzweil}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, Lebesgue measure, open-cover compactness, countable choice, selected infinite sums, host integrand equality, or ambient RealUp completeness.

Upgrade path: A formal route can encode checked carrier admission, RiemannIntegral comparison, DyadicRatCore tolerance exactness, RegSeqRat readback, and TasteGate field faithfulness for BEDC.Derived.HenstockKurzweilUp.

Chapter 4181

A Concrete Naming Certificate for HyperbolicVisualBoundary$^{\uparrow}$

The HyperbolicVisualBoundary$^{\uparrow}$ packet realizes the finite boundary-readback surface of chapter 9463. It sits after the metric-read surface of chapter 3474, the Busemann-boundary row of chapter 3732, the visual-metric packet of chapter 3866, the boundary-shadow packet of chapter 3813, and the boundary-fixed transport route of chapter 3500. The packet names only a finite visual boundary read over displayed disk, shadow, Busemann, visual-metric, transport, provenance, and local naming rows; it does not assert physical cosmology, a completed ideal-boundary quotient, global geometry, or a host compactification theorem.

Definition 4181.1 (Hyperbolic visual boundary carrier)**.** *A* HyperbolicVisualBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (D, S, B, M, T, P, H, C, N)$$

with the following displayed rows:

1. *D is the disk-readout row, exposing the finite* UnitDisk$^{\uparrow}$ *or Poincare-disk observation window from which the boundary read begins.*
2. *S is the boundary-shadow row, recording the* HyperbolicBoundaryShadow$^{\uparrow}$ *face visible from the disk readout.*
3. *B is the Busemann row, attaching the boundary read to* HyperbolicBusemannBoundary$^{\uparrow}$ *distance-difference data.*
4. *M is the visual-metric row, recording the* HyperbolicVisualMetric$^{\uparrow}$ *comparison used to distinguish visible boundary reads.*
5. *T is the transport row, recording the* PoincareBoundaryFixedTransport$^{\uparrow}$ *route that preserves the boundary read while disk data move.*
6. *P is the* Pkg *provenance over* HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, PoincareBoundaryFixedTransport$^{\uparrow}$, UnitDisk$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
7. *H records componentwise* hsame *transport for disk-readout, boundary-shadow, Busemann, visual-metric, transport, provenance, replay, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-read request through D, S, B, M, T, P, H.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, S, B, M, T, P, H, C, N$. It has no coordinate for a completed boundary space, quotient ideal-end equality, physical observer trajectory, global hyperbolic geometry, or an unlogged cosmological claim.

Lean4 ✓ `BEDC.Derived.HyperbolicVisualBoundaryUp.HyperbolicVisualBoundaryTasteGate single carrier alignment`

Theorem 4181.2 (Hyperbolic visual boundary NameCert obligations)**.** *For every accepted* HyperbolicVisualBounda *carrier*

$$V = (D, S, B, M, T, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$ *surface consists exactly of disk-readout admission, boundary-shadow exposure, Busemann attachment, visual-metric comparison, boundary-fixed transport, provenance, componentwise transport, displayed continuation replay, and local naming through $D, S, B, M, T, P, H, C, N$.*

Proof. Project the carrier of Definition 4181.1. The rows $D, S, B, M, T, P, H, C, N$ are the whole public packet.

The read starts at D and S. These rows expose only a finite disk window and a displayed boundary-shadow face.

The Busemann and visual-metric rows B, M attach that face to finite boundary comparison data. The transport row T records boundary-fixed movement without adding a completed boundary object.

Finally P, H, C, N source, transport, replay, and name the same route. Since no other coordinate is present, completed boundary spaces, quotient ideal-end equality, physical observer trajectories, global hyperbolic geometry, and unlogged cosmological claims remain outside the certificate. □

Theorem 4181.3 (Hyperbolic visual boundary readback non-escape)**.** *Every accepted* $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ *read factors through*

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, Poincare

together with disk-readout, provenance, transport, replay, and local NameCert *rows before it exports a visual boundary token.*

Proof. Apply Definition 4181.2. The accepted packet stores boundary readback only in the displayed rows $D, S, B, M, T, P, H, C, N$.

The metric, Busemann, visual-metric, shadow, and fixed-transport siblings are named by those rows and by provenance. A consumer therefore receives a boundary token only after the finite route has exposed those dependencies.

No row in the carrier stores physical cosmology, completed ideal-boundary equality, global geometry, or a host compactification theorem. Hence an accepted read cannot bypass the displayed packet through any of those exits. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ read is refuted if a consumer obtains a visual boundary token without displaying hyperbolicmetric, hyperbolic-busemann-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, poincare-boundary-fixed-transport, disk-readout, provenance, transport, replay, and local NameCert rows.

Independence witness: hyperbolicmetric, hyperbolic-busemann-boundary, hyperbolic-visual-metric, hyperbolic-boundary-shadow, poincare-boundary-fixed-transport: the HyperbolicMetric packet records metric reads without boundary readback, the Busemann packet records distance-difference boundary data without a visual-boundary token, the visual-metric packet records comparison without owning the boundary row, the boundary-shadow packet records a shadow face without the Busemann visual comparison, and the boundary-fixed transport packet preserves a boundary route without naming the visual boundary carrier. None is bijective to $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$ packet is generated from disk-readout rows, HyperbolicBoundaryShadow rows, HyperbolicBusemannBoundary rows, HyperbolicVisualMetric rows, PoincareBoundaryFixedTransport rows, provenance, componentwise hsame transport, displayed Cont replay, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4181.1, Definition 4181.2, and Definition 4181.3 seed the finite visual boundary readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicVisualBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical cosmology, completed ideal-boundary quotients, global hyperbolic geometry, host compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readback, shadow exposure, Busemann attachment, visual-metric comparison, boundary-fixed transport, provenance, replay, transport, and local naming for BEDC.Derived.HyperbolicVisualBoundaryUp.

Chapter 4182

A Concrete Naming Certificate for $\mathsf{BishopCompleteReal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompleteReal}^{\uparrow}$ packet records the textbook complete-real interface as a finite BEDC boundary rather than as an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. It sits on the dyadic endpoint ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated completion surface of chapter 1612, and the completion reflection boundary of chapter 403. The packet exposes the L10 completion interface currently needed by Real and Completion consumers; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete, select limits, quotient streams, or import countable choice.

Definition 4182.1 (Bishop complete real carrier)**.** *A* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, E, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger used to measure regular Cauchy requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row opened by the same tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic, schedule, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, recording that two completion-facing reads are compared through separated finite evidence.*
6. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *handoff row, replaying the same regular-rational source surface into the completion-facing interface.*
7. *H stores componentwise* hsame *transport for $D, W, R, E, S, F, C, P, N$.*
8. *C stores displayed* Cont *replay from a completion-facing request through the same dyadic, schedule, regular, real-seal, separated, and reflection rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient stream equality, a selected limit, countable choice, open-cover compactness, or a theorem that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.TasteGate.BishopCompleteRealTasteGate single carrier alignment`

Theorem 4182.2 (Bishop complete real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier*

$$B = (D, W, R, E, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every completion-facing read passes through* $D, W, R, E, S, F, H, C, P, N$ *in that order.*

No obligation in the packet is discharged by host real equality, quotient streams, selected representatives, countable choice, open-cover compactness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 4233.1. The tuple lists the dyadic, schedule, regular readback, Real seal, separated comparison, completion reflection, transport, replay, provenance, and naming rows.

The completion-facing order is fixed by the visible rows. A request first reads a dyadic endpoint and tolerance ledger in D, then opens the finite schedule row W, and then passes through the regular rational readback row R before the terminal Real seal E is available.

The separated and reflection rows then consume that same source surface. The comparison row S records separated evidence, while F replays the regular-rational source into the completion-facing boundary.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded host, quotient, choice, compactness, or completeness data, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.BishopCompleteRealNamecertObligations`

Theorem 4182.3 (Bishop complete real L10 handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *packet factors through the finite L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{SeparatedCompletion}^{\uparrow} \longrightarrow \mathsf{CompletionReflection}^{\uparrow}.$$

The handoff exports a displayed regular-rational completion interface with separated comparison and reflection rows. It does not export completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, selected host limits, countable-choice coordinates, open-cover compactness, or an ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4233.2. They force every accepted read to pass through the dyadic ledger, finite schedule, regular rational readback, and terminal Real seal before any completion-facing row is consumed.

Next read the separated comparison row. The row S compares completion-facing observations only through displayed separated evidence, so it cannot introduce quotient classes or host equality.

The completion reflection row F then replays the same source surface into the completion-facing interface. The structural rows H, C, P, N preserve that route componentwise and add only transport, replay, provenance, and local naming.

Thus the consumer receives exactly the displayed L10 handoff from dyadic data to schedules, regular rational readback, Real seal, separated comparison, and completion reflection. None of the excluded completeness, choice, quotient, compactness, or selected-limit data has an entry point in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompleteReal}^{\uparrow}$ packet is generated from a dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4233.1, Definition 4233.2, and Definition 4233.3 seed the finite Bishop complete-real interface for the current L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, selected host limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated comparison, completion reflection handoff, and ledger non-escape for the same packet.

Chapter 4183

A Concrete Naming Certificate for $\mathsf{RatCauchyGapWitness}^{\uparrow}$

Following Bishop and Bridges on constructive completion of the rationals, the $\mathsf{RatCauchyGapWitness}^{\uparrow}$ packet records the finite obstruction visible at the $\mathsf{Rat}^{\uparrow}$ boundary before any real-name or completion-facing consumer reads a limit. It sits after the rational carrier of chapter 37, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed gap witness for rational Cauchy bracketing; it does not export a host irrational, quotient completion class, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 4183.1 (Rat Cauchy gap witness carrier)**.** *A* $\mathsf{RatCauchyGapWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, U, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the rational lower-bracket row, exposing a finite* $\mathsf{Rat}^{\uparrow}$ *source, monotone lower estimates, and the rational comparison probes consumed by the witness.*
2. *U is the rational upper-bracket row, exposing the matching* $\mathsf{Rat}^{\uparrow}$ *source, monotone upper estimates, and the rational separation probes kept visible to the consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ladder, recording the dyadic widths used to compare lower and upper brackets at each requested precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule, listing the finite observation windows through which the bracket rows are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, recording the regular-rational readback discipline available before the terminal seal is cited.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, present only as a downstream boundary saying where a real-name consumer may later read a sealed object.*
7. *H records componentwise* hsame *transport for the bracket, tolerance, window, handoff, seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from rational brackets through dyadic tolerances, finite windows, regular-rational readback, and the real seal boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{RatCauchyGapWitness}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, U, D, W, R, E, H, C, P, N$. It has no coordinate for a host irrational number, quotient-completion representative, selected limit, countable-choice principle, or ambient completeness assertion.

Lean4 ✓ `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessTasteGate single carrier alignment`

Theorem 4183.2 (Rat Cauchy gap witness NameCert obligations)**.** *For every accepted* $\mathsf{RatCauchyGapWitness}^{\uparrow}$ *carrier $G = (L, U, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$ *surface consists exactly of five mathematical obligation rows: rational lower and upper brackets through L, U; dyadic tolerance control through D; finite stream-window inspection through W; regular-rational handoff through R; and the terminal real-seal boundary through E. The structural rows H, C, P, N transport, replay, source, and name that same packet, and no host irrational, quotient completion, or choice-selected limit is exported.*

Proof. Project the accepted carrier of Definition 4183.1. The rows L and U are the only rational bracket coordinates, and both are read through the $\mathsf{Rat}^{\uparrow}$ source rather than through a completed real object.

Next inspect the precision route. The dyadic row D supplies the tolerance ladder, the stream row W supplies the finite observation windows, and the regular-rational row R supplies the handoff available to completion-facing consumers.

The seal row E is only a boundary row. It records where a later $\mathsf{Real}^{\uparrow}$ consumer may read a terminal seal, while the present certificate still exposes the rational brackets, dyadic tolerances, stream windows, and regular-rational handoff.

Finally apply the structural rows. Componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the displayed packet. Since the carrier has no coordinate for a host irrational, quotient-completion representative, selected limit, or choice principle, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessNameCert obligations`

Falsifiable prediction: Every accepted $\mathsf{RatCauchyGapWitness}^{\uparrow}$ consumer must expose rational lower and upper brackets, a dyadic tolerance ladder, finite stream windows, a regular-rational handoff, a real-seal boundary, transport, Cont replay, Pkg provenance, and local NameCert rows before it can cite the witness.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{RatCauchyGapWitness}^{\uparrow}$, because this carrier alone binds rational lower and upper brackets to the dyadic tolerance ladder, finite window schedule, regular-rational handoff, terminal seal boundary, transport, replay, provenance, and local NameCert rows without exporting a completed real.

Closure status for $\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{RatCauchyGapWitness}^{\uparrow}$ packet is generated from rational lower and upper bracket rows, a DyadicRatCore tolerance ladder, a StreamName window schedule, a RegSeqRat handoff, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$, seed)

Scope closed: Definition 4183.1 and Definition 4183.2 seed the finite rational Cauchy gap witness over $L, U, D, W, R, E, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RatCauchyGapWitness}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RatCauchyGapWitnessUp.RatCauchyGapWitnessNameCert\_obligations`

Bridge status: none

Not claimed: This chapter does not close host irrational existence, quotient completion equality, selected limits, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rational bracket, dyadic tolerance, finite window, regular-rational handoff, real-seal boundary, transport, replay, provenance, and local naming rows for BEDC.Derived.RatCauchyGapWitnessUp.

Chapter 4184

A Concrete Naming Certificate for $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$

Following Riesz and Markov-Kakutani on positive linear functionals over finite lattice data, the $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ packet records the BEDC-safe finite representation surface for lattice-positive functionals. It consumes the order rows of chapter 53, the lattice rows of chapter 55, the one-sided real boundary of chapter 38, and the lower and upper real approximation rows of chapter 1446 and chapter 1447. The packet is a finite naming certificate for positive functional handoff; it does not assert sigma-additivity, measure completion, quotient dual equality, choice-selected representing measures, or ambient Banach-space representation.

Definition 4184.1 (Riesz representation finite lattice carrier)**.** *A* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, O, F, A, B, P, H, C, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{Lattice}^{\uparrow}$ *row, exposing the finite join, meet, top, bottom, and lattice comparison data used by the representation read.*
2. *O is the* $\mathsf{Poset}^{\uparrow}$ *row supporting the order classifier shared by the lattice and positivity checks.*
3. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$*-style positive functional row, recording additivity on displayed compatible lattice pieces and positivity on the order-positive cone.*
4. *A is the one-sided* $\mathsf{Real}^{\uparrow}$ *boundary row through which the functional values are read.*
5. *B is the* $\mathsf{LowerReal}^{\uparrow}$ *and* $\mathsf{UpperReal}^{\uparrow}$ *bracketing row, storing finite lower and upper cuts that witness the one-sided value interval.*
6. *P is the* Pkg *provenance over* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *H records componentwise* hsame *transport for lattice, order, positive-functional, one-sided real, bracketing, provenance, replay, and naming rows.*
8. *C records displayed* Cont *replay from a positive-functional request through L, O, F, A, B, P, H.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, O, F, A, B, P, H, C, N$. It has no coordinate for completed measure spaces, quotient dual equality, countable joins, choice-selected representing measures, sigma-additivity, or host functional-analysis completion.

Lean4 ✓ `BEDC.Derived.RieszRepresentationFiniteLatticeUp.RieszRepresentationFiniteLatticeTasteGate single carrier alignment`

Theorem 4184.2 (Riesz representation finite lattice NameCert obligations)**.** *For every accepted* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier*

$$R = (L, O, F, A, B, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$ *surface consists exactly of finite lattice admission, shared poset classifier transport, positive-functional row exposure, one-sided real value readback, lower and upper real bracketing, provenance, componentwise transport, displayed continuation replay, and local naming through* $L, O, F, A, B, P, H, C, N$.

Proof. Project the carrier of Definition 4184.1. The rows $L, O, F, A, B, P, H, C, N$ are the whole accepted packet.

The representation read starts with the lattice and order rows. The positive-functional row F is checked only against the displayed lattice operations and the common order classifier supplied by O.

The value side is bounded by A and B. The one-sided $\mathsf{Real}^{\uparrow}$ row exposes the value seal, while the $\mathsf{LowerReal}^{\uparrow}$ and $\mathsf{UpperReal}^{\uparrow}$ row records the finite bracketing data used by the consumer.

Finally P, H, C, N source, transport, replay, and name the same route. Since the carrier contains no additional coordinate, measure completion, quotient dual equality, countable joins, selected representing measures, and host completion remain outside the local certificate. □

Theorem 4184.3 (Riesz representation finite lattice positive-functional handoff)**.** *Every positive-functional consumer of an accepted* $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Lattice}^{\uparrow},\ \mathsf{Poset}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LowerReal}^{\uparrow},\ \mathsf{UpperReal}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow}$$

together with BHist, hsame, Cont, Pkg, *and local* NameCert *rows before it exports a representation token.*

Proof. Apply Definition 4184.2. The theorem exposes the lattice, order, positive-functional, one-sided real, bracketing, transport, replay, provenance, and naming rows as the whole obligation surface.

A downstream consumer can therefore read a representation token only after L, O, F, A, B have been displayed in the same finite packet. The structural rows P, H, C, N preserve that route and do not add a second source of representation data.

No row supplies a completed measure, quotient dual equality, countable lattice operation, selected representing measure, sigma-additivity principle, or host Banach-space representation. The handoff is exactly the finite positive-functional route stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszRepresentationFiniteLattice}^{\uparrow}$ packet is generated from finite lattice rows, poset classifier rows, positive-functional rows, one-sided Real rows, lower and upper real bracketing rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RieszRepresentationFiniteLattice}\uparrow}$, seed)

Scope closed: Definition 4184.1, Definition 4184.2, and Definition 4184.3 seed the finite positive-functional representation route.

Formal verification: FormalStatus(NameCert$_{\text{RieszRepresentationFiniteLattice}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim sigma-additive measure representation, completed measure spaces, quotient dual equality, countable lattice joins, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite lattice admission, order-positive functional exposure, one-sided value bracketing, replay, transport, provenance, and local naming for BEDC.Derived.RieszRepresentationFiniteLatticeUp.

Chapter 4185

A Concrete Naming Certificate for $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$

The $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ packet realizes the focused-ledger, shadow-metric, boundary-shadow, and Cont-growth route of chapter 9463 as finite NameCert data. It sits after the metric-read surface of chapter 3474, the observer-shadow metric row of chapter 4154, the boundary-shadow packet of chapter 3813, the Poincare-disk dynamics route of chapter 4117, and the finite dynamics rows of chapter 195. The packet names only an observer-expansion read over displayed disk, shadow, boundary, growth, provenance, and local naming rows; it does not assert physical cosmology, global expansion, completed boundary geometry, or an unlogged observer.

Definition 4185.1 (Hyperbolic phase observer-expansion carrier)**.** *A* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, D, S, B, G, P, H, C, N)$$

with the following displayed rows:

1. *F is the focused-ledger row, recording the finite observer-facing history slice to be read.*
2. *D is the disk-readout row, exposing the Poincare disk window and its displayed phase coordinates.*
3. *S is the shadow-metric row supplied by* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *over the same observer slice.*
4. *B is the boundary-shadow row supplied by* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *for the visible boundary face.*
5. *G is the Cont-growth row, recording the finite* $\mathsf{DynSystem}^{\uparrow}$ *and* $\mathsf{PoincareDiskDynamics}^{\uparrow}$ *replay by which the observer read expands through displayed disk stages.*
6. *P is the* Pkg *provenance over* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicObserverShadow}$ $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{PoincareDiskDynamics}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
7. *H records componentwise* hsame *transport for focused-ledger, disk-readout, shadow-metric, boundary-shadow, growth, provenance, replay, and naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion request through F, D, S, B, G, P, H.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, S, B, G, P, H, C, N$. It has no coordinate for physical expansion, completed boundary quotient, global hyperbolic geometry, host observer identity, or unlogged cosmological data.

Lean4 ✓ `BEDC.Derived.HyperbolicPhaseObserverExpansionUp.HyperbolicPhaseObserverExpansionTasteGate`
`single carrier alignment`

Theorem 4185.2 (Hyperbolic phase observer-expansion NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ *carrier*

$$E = (F, D, S, B, G, P, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ *surface consists exactly of focused-ledger admission, disk-readout exposure, shadow-metric comparison, boundary-shadow display, Cont-growth replay, provenance, componentwise transport, displayed continuation replay, and local naming through* $F, D, S, B, G, P, H, C, N$.

Proof. Project the carrier of Definition 4185.1. The rows $F, D, S, B, G, P, H, C, N$ are the complete public packet.

The observer-facing read begins with F and D. These rows expose a finite ledger slice and a displayed disk window before any expansion token is available.

The shadow rows S and B then attach the observer slice to finite metric and boundary-shadow data. The growth row G records the Poincare-disk and dynamics replay that the consumer may inspect.

Finally P, H, C, N source, transport, replay, and name the same route. Since no other coordinate is present, physical expansion, completed boundary geometry, global hyperbolic structure, host observer identity, and unlogged cosmological claims remain outside the certificate. □

Theorem 4185.3 (Hyperbolic phase observer-expansion nonescape)**.** *Every accepted* HyperbolicPhaseObserverExpa *read factors through hyperbolic-metric, hyperbolic-observer-shadow-metric, hyperbolic-boundary-shadow, poincare-disk-dynamics, and dynsystem rows together with focused-ledger, disk-readout, Cont-growth, provenance, transport, replay, and local* NameCert *data before it exports an observer-expansion token.*

Proof. Apply Definition 4185.2. The accepted packet stores observer expansion only in the displayed rows $F, D, S, B, G, P, H, C, N$.

The sibling metric, shadow, boundary, disk-dynamics, and dynamics surfaces are named by S, B, G and by provenance. A consumer therefore receives an observer-expansion token only after the finite route has displayed those dependencies.

No row in the carrier stores physical cosmology, completed boundary equality, global geometry, host observer identity, or an unlogged expansion source. Hence an accepted read cannot bypass the displayed packet through any of those exits. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ read is refuted if a consumer obtains an observer-expansion token without displaying focused-ledger, disk-readout, shadow-metric, boundary-shadow, Cont-growth, provenance, and local NameCert rows.

Independence witness: hyperbolic-metric, hyperbolic-observer-shadow-metric, hyperbolic-boundary-shadow, poincare-disk-dynamics, dynsystem: the HyperbolicMetric packet records metric reads without observer expansion, the observer-shadow metric packet records shadow comparison without boundary growth, the boundary-shadow packet records a visible boundary face without disk

dynamics, the Poincare-disk dynamics packet records disk replay without owning the observer ledger, and the DynSystem packet records finite dynamics without naming the hyperbolic phase observer-expansion carrier. None is bijective to $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}$ packet is generated from focused-ledger rows, disk-readout rows, hyperbolic observer-shadow metric rows, boundary-shadow rows, Poincare-disk dynamics rows, DynSystem growth replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4185.1, Definition 4185.2, and Definition 4185.3 seed the finite observer-expansion readback route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPhaseObserverExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close physical cosmology, completed boundary quotients, global hyperbolic geometry, host observer identity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked focused-ledger admission, disk-readout exposure, shadow-metric and boundary-shadow exactness, Cont-growth replay, provenance, transport, and local naming for BEDC.Derived.HyperbolicPhaseObserverExp

Chapter 4186

A Concrete Naming Certificate for $\mathsf{NowhereDense}^{\uparrow}$

Following Baire and Oxtoby, a nowhere-dense component is treated here as the finite certificate surface that lets Baire-category arguments read a closed-frontier refusal without importing arbitrary subset quantification. The $\mathsf{NowhereDense}^{\uparrow}$ packet sits between the topology carrier of chapter 88, the dense-open route of chapter 1542, and the finite cover ledger consumed by chapter 2267.

Definition 4186.1 (Nowhere-dense carrier)**.** *A* $\mathsf{NowhereDense}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, B, F, E, D, M, H, C, P, N)$$

with the following displayed rows:

1. T *is the* $\mathsf{Topology}^{\uparrow}$ *source row, including the basic-neighbourhood names inspected by the packet.*
2. B *is the finite basic-neighbourhood request ledger over* T*.*
3. F *is the closure-frontier row naming how inspected neighbourhoods meet the displayed closure boundary.*
4. E *is the empty-interior refusal row; for every requested basic neighbourhood it displays a sub-neighbourhood escaping the named closed fragment.*
5. D *is the dense-open complement handoff read by* $\mathsf{BaireCategory}^{\uparrow}$ *consumers.*
6. M *is the finite cover-row interface consumed by* $\mathsf{MeagreSet}^{\uparrow}$*.*
7. H *stores the componentwise* hsame *transports for* T, B, F, E, D, M*.*
8. C *stores displayed* Cont *replay of the neighbourhood requests, closure-frontier inspections, empty-interior refusals, and dense-open handoff.*
9. P *is the local* Pkg *provenance row tying the packet to the named topology source.*
10. N *is the local* $\mathsf{NameCert}$ *row for the displayed nowhere-dense component.*

The carrier exports no arbitrary subset classifier, host closure operator, selected generic point, countable-choice selector, or ambient Baire theorem.

Lean4 ✓ `BEDC.Derived.NowhereDenseUp.NowhereDenseTasteGate single carrier alignment`

Theorem 4186.2 (NowhereDense NameCert obligations)**.** *Every accepted* NowhereDense$^{\uparrow}$ *carrier* $Q = (T, B, F, E, D, M, H, C, P, N)$ *supplies exactly the topology source, basic-neighbourhood ledger, closure-frontier row, empty-interior refusal row, dense-open complement handoff, meagre-set cover interface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row listed in Definition 4186.1. A* BaireCategory$^{\uparrow}$ *or* MeagreSet$^{\uparrow}$ *consumer can read the component only through those rows.*

Proof. Open the carrier definition. Each coordinate of Q is a finite row, and the topology source T is the only source from which basic neighbourhoods may be requested.

The closure-frontier and empty-interior rows are then read as paired inspections: F names the frontier contact and E supplies the escaping sub-neighbourhood. The dense-open complement row D and meagre cover row M are downstream of those inspections.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no arbitrary subset classifier, host closure operator, generic point selector, or countable-choice coordinate occurs in Q, downstream Baire-category and meagre-set consumers receive exactly the displayed finite packet. □

Theorem 4186.3 (NowhereDense closure-interior refusal)**.** *Let* $Q = (T, B, F, E, D, M, H, C, P, N)$ *be an accepted* NowhereDense$^{\uparrow}$ *carrier. For every basic-neighbourhood request recorded by* B*, the packet routes the request through the closure-frontier row* F *and then through the empty-interior refusal row* E*, producing the dense-open complement handoff* D *used by* BaireCategory$^{\uparrow}$ *and the cover-row interface* M *used by* MeagreSet$^{\uparrow}$*.*

Proof. Begin with a displayed request in B. The row is indexed over the same topology source T, so it has access only to the named basic-neighbourhood surface.

Read F next. It identifies the closure-frontier inspection attached to the request. Then read E, which displays a sub-neighbourhood escaping the named closed fragment and therefore records empty interior on the inspected finite surface.

The complement handoff D and cover interface M are built from that same refusal. Transport, replay, provenance, and local naming do not alter the request; they keep the Baire-category and meagre-set consumers on the displayed route. □

Closure status for NameCert$_{\text{NowhereDense}^{\uparrow}}$**.**

Constructive story: A NowhereDense$^{\uparrow}$ packet is generated from a Topology$^{\uparrow}$ source, finite basic-neighbourhood requests, a closure-frontier row, an empty-interior refusal row, a dense-open complement handoff, a MeagreSet$^{\uparrow}$ cover interface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{NowhereDense}^{\uparrow}}$, seed)

Scope closed: Definition 4186.1, Definition 4186.2, and Definition 4186.3 seed the finite closure-frontier and empty-interior refusal route consumed by Baire-category and meagre-set packets.

Formal verification: FormalStatus(NameCert$_{\text{NowhereDense}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.NowhereDenseUp.NowhereDenseTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim arbitrary subset closure, host interior operators, selected generic points, countable choice, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite closure-frontier, empty-interior refusal, dense-open complement, and cover-interface obligations.

Chapter 4187

A Concrete Naming Certificate for $\mathsf{SeparableReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SeparableReal}^{\uparrow}$ packet records separability as a finite NameCert surface over dyadic schedules rather than as host enumeration or countable choice. It sits over the real-name seal of chapter 38, the rational carrier of chapter 37, the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, and the existing real separability route of chapter 1809. The packet names the constructive dense schedule visible to Real and Completion consumers; it is not a quotient-real equality principle, a selected host list of all reals, a trichotomy oracle, or an assertion of ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4187.1 (SeparableReal carrier)**.** *A* $\mathsf{SeparableReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (D, S_0, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dense-schedule row, recording the finite dyadic centers and tolerance budgets inspected by a separability request.*
2. *S_0 is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the observation windows through which the dense schedule is read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing regular rational approximants attached to the same windows and dyadic budgets.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording rational comparison data only at the displayed dyadic precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the schedule, window, readback, and tolerance rows have been displayed.*
6. *H records componentwise* hsame *transport for the dense schedule, finite windows, readback, tolerance, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a separability query through D, S_0, R, Q, E.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparableReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$ *row.*
The classifier compares accepted packets only through $D, S_0, R, Q, E, H, C, P, N$. It has no coordinate for host enumeration of $\mathsf{Real}^\uparrow$*, countable choice, quotient real equality, trichotomy of real order, hidden completion data, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.SeparableRealUp.SeparableRealTasteGate single carrier alignment`

Theorem 4187.2 (SeparableReal dyadic density NameCert)**.** *For every accepted* $\mathsf{SeparableReal}^\uparrow$ *carrier*

$$S = (D, S_0, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$ *surface exposes density only by the finite dyadic schedule D, the finite* $\mathsf{StreamName}^\uparrow$ *windows S_0, the* $\mathsf{RegSeqRat}^\uparrow$ *readback row R, the rational and dyadic tolerance row Q, and the terminal* $\mathsf{Real}^\uparrow$ *seal E. The remaining rows H, C, P, N provide componentwise transport, displayed replay, package provenance, and local naming for that same route.*

Consequently an accepted packet supplies a separability witness for Real and Completion consumers without host enumeration, countable choice, quotient real equality, real-order trichotomy, hidden completion data, or a completeness theorem for $\mathsf{Real}^\uparrow$*.*

Proof. Project the carrier of Definition 4187.1. The tuple lists exactly the dense dyadic schedule, finite observation windows, regular rational readback, rational tolerance data, terminal Real seal, transport, replay, provenance, and local naming rows.

The density route is fixed before any real-name consumer reads it. A separability query first opens the dyadic schedule row D and the finite window row S_0, then reads the regular rational approximants in R against the tolerance row Q.

The terminal Real seal E is downstream of those displayed rows, so a consumer never receives an unsealed host real or a selected quotient representative. The structural rows H, C, P, N transport, replay, source, and name only the same displayed data.

Since the carrier contains no coordinate for host enumeration, countable choice, quotient equality, trichotomy, hidden completion data, or ambient completeness, the listed rows exhaust the local NameCert surface. The accepted packet therefore exposes density only through the finite dyadic schedule and the $\mathsf{Real}^\uparrow$ seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$.

Constructive story: A $\mathsf{SeparableReal}^\uparrow$ packet is generated from a dyadic dense schedule, finite StreamName windows, RegSeqRat readback, Rat and DyadicRatCore tolerance rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$, seed)

Scope closed: Definition 4187.1 and Definition 4187.2 seed the finite dyadic density surface for Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeparableReal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host enumeration, countable choice, quotient real equality, real-order trichotomy, hidden completion data, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic density schedule exposure, StreamName window replay, RegSeqRat readback, Rat and DyadicRatCore tolerance exactness, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeparableRealUp.

Chapter 4188

A Concrete Naming Certificate for $\mathsf{BanachFixedPointModulus}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{BanachFixedPointModulus}^{\uparrow}$ packet records the explicit contraction-rate modulus consumed before the endpoint seal of chapter 396. It sits after the residual-to-error route of chapter 2233 and before any $\mathsf{BanachFixedPoint}^{\uparrow}$ consumer reads a $\mathsf{Real}^{\uparrow}$ endpoint boundary. The packet names only the finite contraction graph, the rational Lipschitz ledger below one, the Picard ledger, the Cauchy threshold row, the regular-rational handoff, the complete-metric service row, and the endpoint-facing seal; it does not select a fixed point, assert host equality, or import ambient completeness outside the displayed carrier rows.

Definition 4188.1 (Banach fixed-point modulus carrier)**.** *A* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, T, L, I, M, R, C, E, H, Q, P, N)$$

with the following visible coordinates:

1. *B is the source* $\mathsf{BanachFixedPoint}^{\uparrow}$ *row whose endpoint construction is waiting for an explicit modulus route.*
2. *T is the contraction graph row, displaying the finite map-application history used by the Picard iteration.*
3. *L is the rational Lipschitz ledger below one, including the displayed positivity and unit-bound rows read by the contraction estimate.*
4. *I is the Picard iteration ledger, recording the finite adjacent iterate rows and the replay order for $x_{k+1} = T(x_k)$.*
5. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from L and I for each requested rational tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational name after the modulus threshold has been displayed.*
7. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed by the completion-facing handoff.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, C have been read.*
9. *H stores componentwise* hsame *transports for the fixed-point, graph, Lipschitz, Picard, modulus, readback, complete-metric, seal, provenance, and local naming rows.*

10. *Q stores displayed* Cont *replay from contraction application to Picard iteration, from the Lipschitz ledger and Picard ledger to the modulus threshold, and from the modulus threshold to the regular-rational and endpoint-facing rows.*
11. *P is the* Pkg *provenance over* BanachFixedPointModulus$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardErrorEstimate$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{BanachFixedPointModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, L, I, M, R, C, E, H, Q, P, N$. It has no coordinate for a choice-selected limit, a host fixed-point equality proof, quotient completion equality, global nonlinear analysis, or completeness beyond the displayed CompleteMetric$^{\uparrow}$ *service row.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointModulusUp.BanachFixedPointModulusTasteGate single carrier alignment`

Theorem 4188.2 (Banach fixed-point modulus NameCert obligations)**.** *For every accepted* BanachFixedPointModu *carrier $K = (B, T, L, I, M, R, C, E, H, Q, P, N)$, the local* NameCert$_{\text{BanachFixedPointModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, contraction-graph visibility through T, rational Lipschitz visibility below one through L, finite Picard iteration access through I, Cauchy-threshold construction through M, regular-rational readback through R, complete-metric service access through C, endpoint seal access through E, componentwise* hsame *transport through H, displayed* Cont *replay through Q, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4188.1. The accepted packet exposes exactly the fixed-point row, contraction graph, Lipschitz ledger, Picard ledger, modulus row, regular-rational readback, complete-metric service, endpoint seal, transport, replay, provenance, and local naming rows.

The contraction-rate information is exhausted by T, L, I. The graph row supplies the finite application history, the Lipschitz ledger supplies the displayed bound below one, and the Picard ledger supplies the adjacent iterate rows used by the threshold calculation.

The handoff information is exhausted by M, R, C, E. The modulus threshold row is formed before any regular-rational readback, the complete-metric service is read only at the handoff boundary, and the endpoint seal is exposed only after those rows have been displayed. The structural rows H, Q, P, N transport, replay, source, and name this same route, so no further NameCert obligation is available. □

Theorem 4188.3 (Banach fixed-point modulus Picard handoff)**.** *Every* BanachFixedPoint$^{\uparrow}$ *endpoint-sealing route that consumes a* BanachFixedPointModulus$^{\uparrow}$ *carrier must first factor through*

$$T, L, I \longmapsto M \longmapsto R, C \longmapsto E.$$

The Picard and contraction-rate rows therefore supply the explicit CauchyModulus$^{\uparrow}$ *threshold before the* RegSeqRat$^{\uparrow}$ *readback, the* CompleteMetric$^{\uparrow}$ *service row, and the* Real$^{\uparrow}$ *seal are available to the fixed-point endpoint boundary. The Cauchy-threshold row is read through the* Cauchy$^{\uparrow}$ *naming surface of chapter 3413 before it is specialized to* CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, *and* Real$^{\uparrow}$.

Proof. Start with a fixed-point endpoint request from the source row B. The only contraction-rate rows in the carrier are the graph row T, the Lipschitz ledger L, and the Picard ledger I, so the request must read those rows before it can form a threshold.

The threshold is the displayed CauchyModulus$^{\uparrow}$ coordinate M. The existing BanachFixedPoint$^{\uparrow}$ route of chapter 396 consumes such a threshold before endpoint sealing, while the PicardErrorEstimate$^{\uparrow}$ route of chapter 2233 supplies the residual-and-ratio pattern that makes the threshold finite.

After M is displayed, the handoff can read R and C, and only then can it expose the endpoint-facing row E. The transport, replay, provenance, and local naming rows preserve this order, so the packet exports no selected fixed point, host equality, or ambient completeness theorem. □

Theorem 4188.4 (Banach fixed-point modulus endpoint seal determinacy)**.** *Let $K = (B, T, L, I, M, R, C, E, H, Q,$ be an accepted* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *carrier. Suppose two endpoint-facing reads use the same displayed route*

$$T, L, I \longmapsto M \longmapsto R, C \longmapsto E$$

and their Picard, Lipschitz, modulus, regular-rational, complete-metric, and endpoint rows are compatible under the componentwise hsame *transports carried by H. Then the two reads expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal classifier at E. The endpoint seal is determined by the shared* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{CompleteMetric}^{\uparrow}$ *service rows before any fixed-point endpoint consumer can inspect it.*

Proof. Project the carrier of Definition 4188.1. Both endpoint-facing reads have the same contraction graph, Lipschitz ledger, Picard ledger, modulus threshold, regular-rational readback, complete-metric service row, and terminal seal coordinate.

Apply Definition 4188.3. It forces every endpoint read through T, L, I before M, and through M before the R, C, E handoff, so neither read can introduce an independent endpoint route.

The shared $\mathsf{RegSeqRat}^{\uparrow}$ row and $\mathsf{CompleteMetric}^{\uparrow}$ service row are then read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport through H and replay through Q identify the two reads along those same rows.

Thus the terminal classifier at E is fixed by the displayed route. No selected limit, quotient completion equality, host fixed-point equality, or ambient completeness principle is used to compare the endpoint seals. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachFixedPointModulus}^{\uparrow}$ packet is generated from a BanachFixedPoint row, a contraction graph, a rational Lipschitz ledger below one, a Picard iteration ledger, a CauchyModulus threshold, a RegSeqRat readback, a CompleteMetric service row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4188.1, Definition 4188.2, and Definition 4188.3 seed the finite contraction-rate modulus route consumed before Banach fixed-point endpoint sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, ambient completeness, Lean verification, bridge checking, or global nonlinear analysis.

Upgrade path: Obligation closure requires checked carrier admission, contraction-graph visibility, Lipschitz-below-one visibility, Picard iteration access, Cauchy-threshold construction, regular-rational readback, complete-metric service access, endpoint seal access, transport, replay, provenance, and local naming for BEDC.Derived.BanachFixedPointModulusUp.

Chapter 4189

A Concrete Naming Certificate for $\mathsf{SpreadSpace}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on spreads and choice sequences, the $\mathsf{SpreadSpace}^{\uparrow}$ packet records a finite-prefix space of admissible choice-sequence names rather than an ambient set of completed infinite branches. It sits between the Baire carrier of chapter 334, the finite stream-name consumer of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. Its spread-side input is supplied by the Brouwer spread packet of chapter 1677, the local law packet of chapter 1532, and the choice-sequence packet of chapter 1482.

Definition 4189.1 (SpreadSpace carrier)**.** *A* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, L, Q, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite-prefix row, recording the finite natural-number windows on which the packet may be inspected.*
2. *L is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row, deciding whether each displayed prefix is permitted by the spread law.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *branch-prefix row, exposing only those choice-sequence observations whose prefixes pass L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by downstream name consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the accepted branch prefixes and stream windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, L, Q, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the prefix, law, branch, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a spread-space query through B, L, Q, W, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{SpreadSpace}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, L, Q, W, R, E, H, C, P, N$. It has no coordinate for a completed infinite branch, host equality of functions, countable choice, selected real values, bar induction outside the displayed prefix bar, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.SpreadSpaceUp.spreadSpaceTasteGate`
Lean4 ✓ `BEDC.Derived.SpreadSpaceUp.SpreadSpaceTasteGate single carrier alignment`

Theorem 4189.2 (SpreadSpace prefix-bar stability)**.** *For every accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier*

$$S = (B, L, Q, W, R, E, H, C, P, N),$$

any downstream read that reaches a $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *row must first pass through the same finite* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, the same* $\mathsf{SpreadLaw}^{\uparrow}$ *admission row L, and the same* $\mathsf{ChoiceSequence}^{\uparrow}$ *branch-prefix row Q. If two accepted reads are componentwise related by the displayed* hsame *row H and replayed by* Cont *through C, then their prefix-bar boundary is transported without adding an infinite branch, host function equality, countable choice, selected real value, or ambient completeness coordinate.*

Proof. Project the carrier in Definition 4189.1. The only entrance to a downstream name read is the displayed prefix route B, L, Q.

The stream and rational rows are downstream of that entrance. A $\mathsf{StreamName}^{\uparrow}$ window W and a $\mathsf{RegSeqRat}^{\uparrow}$ readback R are available only after the finite Baire prefix has been admitted by the spread law and attached to a choice-sequence prefix.

Transport preserves that order. The row H relates the listed coordinates componentwise, while C replays the same route through the finite packet. Since the carrier contains no infinite branch, host function equality, choice schedule, selected real value, or ambient completeness coordinate, the prefix-bar boundary is stable under the displayed transport and replay rows. □

Theorem 4189.3 (SpreadSpace NameCert obligations)**.** *For every accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier $S = (B, L, Q, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$ *surface consists exactly of the finite obligations for carrier admission, spread-law prefix acceptance, choice-sequence prefix compatibility, StreamName window replay, RegSeqRat readback, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. These obligations are rows in* BHist, hsame, Cont, Pkg, *and* NameCert*; they do not assert completed branch existence, host equality of functions, countable choice, selected representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the tuple displayed in Definition 4189.1. Each coordinate has a single obligation role: prefix carrier, law admission, branch-prefix compatibility, finite stream window, regular rational readback, terminal real seal, transport, replay, provenance, or local naming.

The first three rows determine the spread-space boundary. The rows W, R, E then expose the L10-facing naming route only after that boundary has been displayed.

The structural rows H, C, P, N do not add mathematical objects beyond the listed surface. They transport, replay, source, and name the same finite route, so every local certificate obligation is exhausted by BHist, hsame, Cont, Pkg, and NameCert rows. □

Theorem 4189.4 (SpreadSpace StreamName–Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{SpreadSpace}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{SpreadSpace}^{\uparrow} \longmapsto \mathsf{BrouwerSpread}^{\uparrow} \longmapsto \mathsf{SpreadLaw}^{\uparrow} \longmapsto \mathsf{ChoiceSequence}^{\uparrow} \longmapsto \mathsf{BaireSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Re}$$

together with the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the packet. The handoff exports no completed spread branch, no host function equality, no countable-choice schedule,*

no selected real representative, no quotient stream equality, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start from prefix-bar stability, Definition 4189.2. It forces every downstream name read through the finite prefix route B, L, Q before a StreamName window can be opened.

Apply the obligation inventory of Definition 4189.3. The rows W, R, E are the only StreamName, RegSeqRat, and Real-facing coordinates of the local packet, and the rows H, C, P, N only preserve and name that route.

The displayed chain is therefore the whole handoff available to a Real-facing consumer. None of the excluded objects is a coordinate of the carrier, so the handoff remains a finite NameCert route rather than a completed branch or ambient Real-completeness assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SpreadSpace}^{\uparrow}$ packet is generated from a BaireSpace finite-prefix row, a SpreadLaw admission row, a ChoiceSequence branch-prefix row, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4189.1, Definition 4189.2, Definition 4189.3, and Definition 4189.4 seed the finite spread-space route from Brouwer spread prefixes to StreamName and Real consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpreadSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim completed infinite branch existence, host equality of functions, countable choice, selected real representatives, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, spread-law prefix acceptance, choice-sequence compatibility, StreamName window replay, RegSeqRat readback, Real seal handoff, transport replay, provenance, and non-escape for BEDC.Derived.SpreadSpaceUp.

Chapter 4190

A Concrete Naming Certificate for $\mathsf{UniformCauchyEmbedding}^{\uparrow}$

Following Bishop and Bridges on uniformly continuous Cauchy data, the $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ packet records the finite certificate surface for embedding one displayed Cauchy presentation into another without selecting host limits. It sits between the Cauchy source discipline of chapter 3413, the finite schedule rows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the represented-space carrier boundary of chapter 1494. The packet is an open Real and Completion phase object: it makes uniform Cauchy embeddings available as finite NameCert data while leaving completion, quotient equality, and chosen limits outside the carrier.

Definition 4190.1 (Uniform Cauchy embedding carrier)**.** *A* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, S, R, T, E, V, H, K, P, N)$$

with the following displayed rows:

1. *C is the source* $\mathsf{Cauchy}^{\uparrow}$ *row, recording the Cauchy admissibility request whose observations are to be embedded.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row opened by a target tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the source schedule.*
4. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *schedule row, with its tolerance request bounded by a displayed uniform modulus row.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal row reached only after the source Cauchy row, the source schedule, the regular readback, and the target schedule have been displayed.*
6. *V is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *representation row for the source and target observation surfaces.*
7. *H stores componentwise* hsame *transport for $C, S, R, T, E, V, K, P, N$.*
8. *K stores displayed* Cont *replay from a target tolerance request back through the source Cauchy, source schedule, regular readback, target schedule, Real seal, and represented-space rows.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient Cauchy classes, host real equality, a selected completion point, a global inverse map, countable choice, or a proof that the represented-space target is complete.

Lean4 (definition) `BEDC.Derived.UniformCauchyEmbeddingUp.uniformCauchyEmbeddingTasteGate`
Lean4 ✓ `BEDC.Derived.UniformCauchyEmbeddingUp.UniformCauchyEmbeddingTasteGate single carrier alignment`

Theorem 4190.2 (Uniform Cauchy embedding NameCert obligations)**.** *For every accepted* UniformCauchyEmbeddi *carrier*

$$U = (C, S, R, T, E, V, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the source Cauchy row, source schedule, regular rational readback, target schedule, terminal Real seal, represented-space row, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every target tolerance request is replayed through* $C, S, R, T, E, V, H, K, P, N$ *in that order.*

No obligation in the packet is discharged by quotient Cauchy classes, host real equality, selected completion points, global inverse data, countable choice, or completeness of the represented-space target.

Proof. Carrier admission is projection from Definition 4190.1. The tuple displays the Cauchy source, source schedule, regular rational readback, target schedule, Real seal, represented-space, transport, replay, provenance, and naming rows.

The replay order is fixed by the row dependencies. A target tolerance request first opens the source Cauchy row C, then the finite schedule row S, and then the regular-rational readback row R before the target schedule T can be read.

The terminal rows consume only that displayed source surface. The Real seal E records the target real-name boundary, while V records the represented-space observation surface on both sides of the embedding.

Finally H, K, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded quotient, host, choice, inverse, completion, or completeness data, the listed obligations exhaust the local NameCert surface. □

Theorem 4190.3 (Uniform Cauchy embedding sibling route)**.** *Every accepted* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *packet factors through the finite sibling route*

$$\mathsf{Cauchy}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{RepresentedSpace}^{\uparrow}.$$

The first $\mathsf{StreamName}^{\uparrow}$ *row schedules source observations, and the second* $\mathsf{StreamName}^{\uparrow}$ *row schedules target observations under the displayed uniform replay row. The route exports finite embedding data only; it does not export quotient Cauchy classes, selected representatives, global inverse data, or target completeness.*

Proof. Apply the obligations of Definition 4190.2. They require every target tolerance request to pass through the source Cauchy row, source schedule, regular rational readback, target schedule, Real seal, and represented-space row.

The two schedule rows have distinct roles. The source schedule S exposes the observations that can be read from the Cauchy source, while the target schedule T records the target request after the displayed uniform replay has bounded it.

The Real and represented-space rows then seal the route without adding non-displayed structure. Because H, K, P, N only transport, replay, source, and name the same rows, the packet exports the finite sibling route and none of the excluded quotient, representative, inverse, or completeness data. □

Theorem 4190.4 (Uniform Cauchy embedding open phase use)**.** *In the open Real and Completion phase, a* $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ *packet may be used as the finite handoff object from regular Cauchy source data to Real-facing and represented-space-facing consumers. Such a consumer may read the displayed Cauchy, schedule, regular-rational, target-schedule, Real-seal, represented-space, transport, replay, provenance, and naming rows, but it may not infer completion of* $\mathsf{Real}^{\uparrow}$*, a quotient construction for Cauchy names, a selected limit, or countable-choice data.*

Proof. The handoff object is the carrier of Definition 4190.1. Its rows expose exactly the source Cauchy request, the two finite schedules, the regular rational readback, the Real seal, the represented-space surface, and the structural transport, replay, provenance, and naming data.

The open phase use reads this finite surface as an interface for later Real and Completion obligations. It can therefore pass uniform Cauchy embedding evidence downstream without changing the carrier into a completion theorem.

The exclusions follow from absence of coordinates. Since the carrier contains no quotient construction, selected limit, completeness proof, or countable-choice row, no open phase consumer obtains those data from the $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ NameCert. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyEmbedding}^{\uparrow}$ packet is generated from a source Cauchy row, source schedule, regular rational readback, target schedule, terminal Real seal, represented-space row, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4190.1, Definition 4190.2, Definition 4190.3, and Definition 4190.4 seed the finite uniform Cauchy embedding interface for the open Real and Completion phase.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient Cauchy equality, selected completion points, global inverse data, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform replay non-escape, sibling route coverage, and open phase use for the same packet.

Chapter 4191

A Concrete Naming Certificate for $\mathsf{DecimalStream}^{\uparrow}$

Following Bishop and Bridges on decimal presentations of constructive real numbers, the $\mathsf{DecimalStream}^{\uparrow}$ packet records the finite base-ten stream surface that sits between decimal prefix data and real-name readback. It refines the finite prefix discipline of chapter 1417 without asking $\mathsf{DecimalExpansion}^{\uparrow}$ to carry an infinite stream role, and it routes later reads through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DecimalReal}^{\uparrow}$, and $\mathsf{SignedDigitStream}^{\uparrow}$ surfaces of chapter 126, chapter 291, chapter 301, chapter 38, chapter 2058, and chapter 316.

Definition 4191.1 (DecimalStream carrier)**.** *A* $\mathsf{DecimalStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (E, S, W, L, Q, R, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{DecimalExpansion}^{\uparrow}$ *prefix row, carrying admitted signed decimal prefixes and their endpoint policy.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, exposing only finite digit windows requested by a downstream consumer.*
3. *W is the decimal window ledger, recording observed base-ten digits, place indices, and prefix bounds for each request in S.*
4. *L is the place-value ledger that evaluates every displayed window as a rational base-ten partial sum.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row that translates the place-value ledger into dyadic tolerance data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the dyadic comparison windows.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DecimalReal}^{\uparrow}$ *seal boundary, with the* $\mathsf{SignedDigitStream}^{\uparrow}$ *sibling used only as a comparison surface for finite-window stream behavior.*
8. *H records componentwise* hsame *transport for E, S, W, L, Q, R, A and the local structural rows.*
9. *C records displayed* Cont *replay from a digit-window request through the decimal prefix, stream schedule, place-value ledger, dyadic comparison, regular-rational handoff, and terminal real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DecimalReal}^{\uparrow}$, $\mathsf{SignedDigitStream}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite decimal object, quotient stream equality, a selected decimal expansion at endpoints, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DecimalStreamUp.DecimalStreamTasteGate single carrier alignment`

Theorem 4191.2 (DecimalStream NameCert obligations)**.** *For every accepted* $\mathsf{DecimalStream}^{\uparrow}$ *carrier* $D = (E, S, W, L, Q, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$ *surface consists exactly of carrier admission, finite digit-window exposure, base-ten place-value ledger evaluation, dyadic comparison handoff, regular-rational readback, Real and DecimalReal seal nonescape, signed-digit sibling comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.DecimalStreamUp.DecimalStreamCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4191.1. The rows E, S, W, L, Q, R, A are the only decimal-prefix, stream-window, place-value, dyadic, regular-rational, and real-facing rows available to the packet.

Finite digit-window exposure is forced by S and W. A downstream read can inspect only the requested base-ten windows, so no infinite decimal stream, hidden tail, or endpoint selector appears.

The place-value ledger L then turns each window into rational partial-sum data. The dyadic row Q and regular-rational row R transport that ledger into the same $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route used by the L10 real boundary.

Finally, A, H, C, P, N seal, transport, replay, source, and locally name the route. Since the carrier lacks quotient stream equality, selected endpoint expansions, host real equality, countable choice, and ambient completeness coordinates, the listed rows exhaust the NameCert obligations. □

Theorem 4191.3 (DecimalStream real readback boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{DecimalReal}^{\uparrow}$ *read of an accepted* $\mathsf{DecimalStream}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The comparison with $\mathsf{SignedDigitStream}^{\uparrow}$ *is only a finite-window sibling comparison. The readback exports no infinite decimal stream, no quotient of streams, no selected endpoint expansion, no host equality of real representatives, and no* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with an accepted packet and project the rows of Definition 4191.1. A real-facing read must inspect the decimal prefix row E, the stream schedule S, and the requested window ledger W before place-value data are available.

The place-value ledger L is then compared through Q and handed to R. Thus the Real-facing route is the finite dyadic and regular-rational route already used by $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ consumers.

The terminal seal row A records only the $\mathsf{Real}^{\uparrow}$ and $\mathsf{DecimalReal}^{\uparrow}$ boundary reached by that route. The $\mathsf{SignedDigitStream}^{\uparrow}$ sibling is cited for finite-window comparison behavior and does not add a second stream quotient.

Transport and replay through H and C preserve the same rows, while P and N source and name them. No excluded infinite stream, quotient, endpoint selector, host equality, or completeness coordinate is present, so the displayed boundary is exact. □

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalStream}^{\uparrow}$ packet is generated from DecimalExpansion prefix rows, finite StreamName digit windows, place-value ledgers, DyadicRatCore comparison rows, RegSeqRat handoff rows, RealUp and DecimalRealUp seal rows, SignedDigitStream sibling comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4191.1, Definition 4191.2, and Definition 4191.3 seed the finite decimal stream surface from decimal prefixes to dyadic, regular-rational, and real-name consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an infinite decimal stream object, quotient stream equality, selected endpoint expansions, host real equality, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite digit-window admission, place-value ledger exactness, dyadic comparison handoff, RegSeqRat readback, Real-seal nonescape, transport replay, provenance, and local naming for BEDC.Derived.DecimalStreamUp.

Chapter 4192

A Concrete Naming Certificate for MinimalTriggerOrbitClassification$^{\uparrow}$

The MinimalTriggerOrbitClassification$^{\uparrow}$ packet realizes Definition 9462.1681 as finite BEDC NameCert data. It sits after the DNA boundary-tile source of chapter 3393, the lift-fiber packet of chapter 4129, the homology-core packet of chapter 4137, and the window coverage algebra of chapter 3478. The packet records nine normal forms, defect rows, quotient and codon counts, transport, replay, provenance, and local naming as displayed finite rows; it is not a biological stochastic model, hidden codon semantics, unrestricted orbit enumeration, or an unrecorded homology computation.

Definition 4192.1 (MinimalTriggerOrbitClassification carrier)**.** *A* MinimalTriggerOrbitClassification$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, N, D, Q, K, L, M, H, C, P, R)$$

with the following displayed rows:

1. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *source row for the six-coordinate boundary-tile window.*
2. *N is the nine-normal-form ledger, displaying the orbit representatives of the minimal-trigger classification.*
3. *D is the defect ledger for the partition, single-overlap, double-overlap, and triple-overlap rows.*
4. *Q is the quotient-count row, preserving the* 49 *quotient-level minimal full triggers.*
5. *K is the codon-count row, preserving the two-lift total* 294*.*
6. *L is the* MinimalTriggerLiftBundle$^{\uparrow}$ *handoff row for lift-fiber decomposition.*
7. *M is the* MinimalTriggerHomologyCore$^{\uparrow}$ *handoff row for support-nerve core reads.*
8. *H records componentwise* hsame *transport for the source, normal-form, defect, count, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from boundary-tile windows through normal forms, defect rows, quotient counts, codon counts, lift fibers, and homology-core handoffs.*
10. *P is the* Pkg *provenance over* MinimalTriggerOrbitClassification$^{\uparrow}$*,* DnaWindowSixBoundaryTile$^{\uparrow}$*,* MinimalTriggerLiftBundle$^{\uparrow}$*,* MinimalTriggerHomologyCore$^{\uparrow}$*,* WindowCoverageTriggerAlgebra$^{\uparrow}$*, the source classification theorem,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *R is the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}$ *row.*
The carrier has no coordinate for an empirical genetic-code law, a stochastic fitness model, quotient codon semantics outside the displayed rows, an unrestricted orbit search, or an external homology oracle.

Lean4 ✓ BEDC.Derived.MinimalTriggerOrbitClassificationUp.MinimalTriggerOrbitClassificationTasteGate single carrier alignment

Theorem 4192.2 (MinimalTriggerOrbitClassification NameCert obligations)**.** *For every accepted* $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *carrier* $O = (B, N, D, Q, K, L, M, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbit}}$ *surface consists exactly of boundary-tile source admission, nine-normal-form exposure, defect-ledger exposure, quotient-count preservation, codon-count preservation, lift-bundle handoff, homology-core handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer preserving the packet must preserve the nine normal forms, the quotient count* 49*, and the codon count* 294 *through those rows.*

Proof. Project the carrier of Definition 4192.1. The row B supplies the boundary-tile source, while N, D, Q, K display the normal forms, defects, quotient count, and codon count.

The lift and homology consumers are the rows L and M. They can read the classification only after the finite normal-form and count rows have been displayed, so neither handoff becomes an independent orbit enumeration.

Transport and replay are carried by H and C. These rows preserve the same displayed source, normal-form, defect, count, lift, and homology data componentwise under hsame and Cont.

Finally, P and R source and name exactly this finite packet. Since no excluded empirical, stochastic, quotient-semantics, unrestricted-search, or external-homology coordinate is present, the listed rows exhaust the NameCert obligations and force preservation of the nine forms and the counts 49 and 294. □

Theorem 4192.3 (MinimalTriggerOrbitClassification vision handoff)**.** *The finite classification of Definition 9462.1681 is consumed by* $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *only through the route*

$$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{WindowCoverageTriggerAlgebra}^{\uparrow} \longmapsto N, D, Q, K \longmapsto \mathsf{MinimalTriggerLiftBundle}^{\uparrow},\ \mathsf{Min}$$

Future Lean TasteGate files for this packet must provide explicit Nontrivial $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *and* FieldFaithful $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ *obligations before the packet can serve as a verified trigger-classification consumer.*

Proof. Start from Definition 4192.2. It places the boundary-tile and coverage-algebra source rows before any normal-form or count row can be consumed.

The middle block N, D, Q, K is the finite classification ledger. It records the nine normal forms, their defect classes, the quotient count, and the codon count without opening an unrestricted orbit search.

The handoff rows L and M attach lift-fiber and homology-core consumers to the same ledger. The structural rows H, C, P, R then transport, replay, source, and locally name the route.

Thus the source theorem is represented as a finite packet. The stated Lean TasteGate obligations are downstream verification requirements for this carrier and do not add fields to the paper-side packet. □

Falsifiable prediction: For every accepted $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ packet, componentwise hsame transport preserves exactly nine displayed normal forms, the quotient-level count 49, and the codon-level count 294.

Independence witness: dna-window-six-boundary-tile, minimal-trigger-lift-bundle, minimal-trigger-homology-core, window-coverage-trigger-algebra: $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries normal-form rows, defect rows, quotient-count rows, codon-count rows, lift and homology handoffs, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}$ packet is generated from a DnaWindowSixBoundaryTile source row, nine normal-form rows, defect ledgers, quotient-count and codon-count rows, MinimalTriggerLiftBundle and MinimalTriggerHomologyCore handoff rows, WindowCoverageTriggerAlgebra provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4192.1, Definition 4192.2, and Definition 4192.3 seed the finite orbit-classification packet preserving nine normal forms, defect rows, quotient count 49, and codon count 294.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalTriggerOrbitClassification}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.MinimalTriggerOrbitClassificationUp.MinimalTriggerOrbitClassificationTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not claim biological mechanism, stochastic fitness, quotient codon semantics outside displayed rows, unrestricted orbit enumeration, Lean verification, Nontrivial or FieldFaithful TasteGate instances, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, normal-form preservation, defect-ledger preservation, quotient-count preservation, codon-count preservation, lift and homology handoffs, transport replay, provenance, local naming, Nontrivial MinimalTriggerOrbitClassificationUp, and FieldFaithful MinimalTriggerOrbitClassificationUp for BEDC.Derived.MinimalTriggerOrbitClassificationUp.

Chapter 4193

A Concrete Naming Certificate for $\mathsf{Portmanteau}^{\uparrow}$

Following Billingsley and Prokhorov, the $\mathsf{Portmanteau}^{\uparrow}$ packet records the weak-convergence test surface that becomes visible only after tight probability data have already been exposed. It sits after the probability source of chapter 184, the pushforward distribution row of chapter 186, the event-surface measure row of chapter 92, the real comparison boundary of chapter 38, the tight probability handoff of chapter 4008, and the compact metric test surface of chapter 129. The chapter names a finite BEDC test packet for bounded-continuous probes; it does not assert weak compactness, choose subsequences, use countable choice, export host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4193.1 (Portmanteau carrier)**.** *A* $\mathsf{Portmanteau}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (P, D, M, T, B, R, A, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row carrying the probability surface.*
2. *D is the* $\mathsf{Distribution}^{\uparrow}$ *row whose pushforward reads supply the tested laws.*
3. *M is the* $\mathsf{Measure}^{\uparrow}$ *event-surface row through which measured test preimages are inspected.*
4. *T is the* $\mathsf{ProkhorovTightProbability}^{\uparrow}$ *handoff row already exposing tightness data before any weak-convergence test is read.*
5. *B is the bounded-continuous test row, recorded as a displayed* $\mathsf{BoundedContinuousTest}$ *surface over the compact or metric test carrier.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row for the two finite test expectations or inequalities observed by the packet.*
7. *A is the transport row aligning probability, distribution, measure, tightness, bounded-continuous test, and real-comparison reads.*
8. *E is the finite endpoint or tolerance ledger that records which bounded-continuous probe and real comparison window are being inspected.*
9. *H records componentwise* hsame *transport for every displayed row.*

10. *C records displayed* Cont *replay of source admission, pushforward readback, measure inspection, tightness exposure, bounded-continuous test evaluation, real comparison, and tolerance readback.*
11. *Q is the* Pkg *provenance over* $\mathsf{Portmanteau}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProkhorovTightProbability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$ *row.*

The carrier has no coordinate for weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PortmanteauUp.PortmanteauTasteGate single carrier alignment`

Theorem 4193.2 (Portmanteau NameCert obligations)**.** *For every accepted* $\mathsf{Portmanteau}^{\uparrow}$ *carrier* $W = (P, D, M, T, B, R, A, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$ *surface consists exactly of six obligation groups:*

1. *carrier admission for the probability source, distribution pushforward, event-surface measure, tightness handoff, bounded-continuous test, real comparison, transport, tolerance, replay, provenance, and local naming rows;*
2. *tightness-before-testing exposure, requiring each weak-convergence test read to display the* ProkhorovTightProbabil *row T before the bounded-continuous probe is consumed;*
3. *bounded-continuous test admission, requiring the probe row B to be read through the displayed compact or metric test surface rather than through a host function space;*
4. *real comparison exactness, requiring the comparison row R to be read through* $\mathsf{Real}^{\uparrow}$ *and the tolerance ledger E;*
5. *distribution and measure compatibility, requiring D and M to name the same tested event or pushforward observation under the transport row A;*
6. *non-escape from weak compactness, selected subsequences, countable choice, host measures, quotient events, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No additional NameCert obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 4193.1. The tuple lists exactly the probability, distribution, measure, tightness, bounded-continuous test, real comparison, transport, tolerance, replay, provenance, and naming rows.

The tightness side is already finite. A consumer must inspect T before the probe row B can be evaluated, so the weak-convergence test is downstream of the Prokhorov tightness handoff rather than a source of compactness by itself.

The comparison side is real-name data. The row R and tolerance ledger E provide the displayed real comparison, while D, M, A keep pushforward and measured-event observations on the same route.

Since no carrier coordinate stores weak compactness, selected subsequences, countable choice, host measures, quotient events, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the six displayed groups exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{Portmanteau}^{\uparrow}$ test read exposes $\mathsf{ProkhorovTightProbability}^{\uparrow}$, a bounded-continuous probe row, and a $\mathsf{Real}^{\uparrow}$ comparison row before any weak-convergence consumer can inspect it.

Independence witness: probspace, distribution, measure, prokhorov-tight-probability, real, compactmetric: the packet is not bijective to any listed sibling because it binds tight probability data, bounded-continuous test admission, real comparison, transport, replay, provenance, and local NameCert rows as one weak-convergence test surface.

Closure status for $\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}$.

Constructive story: A $\mathsf{Portmanteau}^{\uparrow}$ packet is generated from ProbSpaceUp source rows, DistributionUp pushforward rows, MeasureUp event-surface rows, ProkhorovTightProbabilityUp handoff rows, bounded-continuous test rows, RealUp comparison rows, transport, hsame, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4193.1 and Definition 4193.2 seed the weak-convergence bounded-continuous test surface after tight probability data have been displayed.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Portmanteau}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close weak compactness, selected subsequences, countable choice, host measures, quotient events, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tightness-before-testing exposure, bounded-continuous test admission, real comparison exactness, distribution and measure compatibility, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.PortmanteauUp.

The $\mathsf{MonotoneClass}^{\uparrow}$ top-level route sends generated closure through chapter 92, chapter 3384, and chapter 3668 before any measure-facing consumer reads the packet.

Chapter 4194

A Concrete Naming Certificate for $\mathsf{MonotoneClass}^{\uparrow}$

Following Halmos and Dynkin-style monotone class arguments, the $\mathsf{MonotoneClass}^{\uparrow}$ packet records the closure object that measure-theoretic generation uses before any consumer may cite a sigma-algebra completion. It sits after the measurable-event carrier of chapter 92, the generated-set boundary of chapter 3384, and the displayed membership discipline of chapter 3668. The packet names generator admission, increasing-union rows, decreasing-intersection rows, measurable-event admission, transport, replay, provenance, and local naming; it does not assert a host sigma-algebra, pi-lambda theorem, selected countable chain, measure extension, or quotient equality of subsets.

Definition 4194.1 (MonotoneClass carrier)**.** *A* $\mathsf{MonotoneClass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, U, I, A, H, C, P, N)$$

with the following displayed rows:

1. *G is the generator family over* BHist*, read as displayed subset rows through* $\mathsf{SetLike}^{\uparrow}$ *and as generated-set boundary data through* $\mathsf{BorelSet}^{\uparrow}$ *when the source is Borel-facing.*
2. *U records increasing-union closure rows for finite observed chains and their named limit event request.*
3. *I records decreasing-intersection closure rows for finite observed chains and their named limit event request.*
4. *A records measurable-event admission through the* $\mathsf{BorelSet}^{\uparrow}$ *and* $\mathsf{Measure}^{\uparrow}$ *sibling surfaces.*
5. *H stores componentwise* hsame *transport for G, U, I, A and the structural rows.*
6. *C stores displayed* Cont *replay from a generator request through increasing-union closure, decreasing-intersection closure, measurable-event admission, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{MonotoneClass}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{SetLike}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through G, U, I, A, H, C, P, N. It has no coordinate for completed sigma-algebras, chosen countable unions, host subset equality, measure extension, or an ambient monotone-class theorem.

Theorem 4194.2 (MonotoneClass NameCert obligations)**.** *For the carrier of Definition 4194.1, the local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$ *surface consists exactly of generator admission through* G*, increasing-union closure through* U*, decreasing-intersection closure through* I*, measurable-event admission through* A*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*. Every accepted* $\mathsf{Measure}^\uparrow$*,* $\mathsf{BorelSet}^\uparrow$*, or* $\mathsf{SetLike}^\uparrow$ *consumer that reads a monotone-class conclusion must expose those rows before it treats the generated family as closed under the requested operation.*

Proof. Project the carrier $M = (G, U, I, A, H, C, P, N)$. The generator row G is the only entry point for displayed set requests, so carrier admission is exhausted by the generator family and its sibling $\mathsf{SetLike}^\uparrow$ or $\mathsf{BorelSet}^\uparrow$ boundary.

Read the closure rows next. The row U supplies increasing-union closure for the named observed chain, and I supplies decreasing-intersection closure for the named observed chain. The measurable-event admission row A is the only route from those closures to the $\mathsf{BorelSet}^\uparrow$ and $\mathsf{Measure}^\uparrow$ surfaces.

Finally replay the structural rows. Transport through H, continuation through C, provenance through P, and local naming through N preserve the same finite route. Since no carrier coordinate contains completed sigma-algebras, selected countable chains, quotient subset equality, measure extension, or an ambient monotone-class theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 4194.3 (MonotoneClass generated closure route)**.** *Every accepted* $\mathsf{MonotoneClass}^\uparrow$ *packet exports the generated closure route*

$$G \longmapsto U, I \longmapsto A \longmapsto H, C, P, N$$

for downstream $\mathsf{Measure}^\uparrow$*,* $\mathsf{BorelSet}^\uparrow$*, and* $\mathsf{SetLike}^\uparrow$ *consumers. The route lets a consumer cite monotone closure of the displayed generator family only after increasing-union, decreasing-intersection, measurable-event, transport, replay, provenance, and local-name rows have been exposed; it exports no sigma-algebra completion, pi-lambda closure, measure-extension theorem, selected chain, or quotient equality of subsets.*

Proof. Start with a displayed generator request in G. The request is a finite BEDC row, not a host subset family, and it can be read only through the named $\mathsf{SetLike}^\uparrow$ and $\mathsf{BorelSet}^\uparrow$ surfaces.

Pass through the two closure rows. Increasing requests are handled by U, decreasing requests by I, and both remain attached to the same displayed generator family. The admission row A then records when the resulting event request is visible to the Borel and measure siblings.

Replay the route through H, C, P, N. These rows transport, replay, source, and name the same finite certificate. Therefore a downstream consumer receives exactly the generated closure route stated above and no completed sigma-algebra, pi-lambda theorem, measure-extension theorem, selected chain, or quotient subset equality. □

Theorem 4194.4 (MonotoneClass generator induction boundary)**.** *Every accepted* $\mathsf{MonotoneClass}^\uparrow$ *consumer reaches generated closure only by finite induction over the displayed generator row* G*, increasing-union row* U*, decreasing-intersection row* I*, measurable-event row* A*, transport row* H*, replay row* C*, provenance row* P*, and local name row* N*. The induction boundary cannot be replaced by a host pi-lambda theorem, selected countable chain, measure extension, completed sigma-algebra, or quotient subset equality.*

Proof. Unfold the carrier $M = (G, U, I, A, H, C, P, N)$ from Definition 4194.1. The only initial cases available to a consumer are displayed generator requests in G, and each request is a finite BEDC row rather than a host subset family.

For the induction steps, use the closure route of Definition 4194.3. Increasing-union steps are read through U, decreasing-intersection steps through I, and measurable-event admission through A; each step keeps the same displayed generator family attached.

Apply the NameCert obligations of Definition 4194.2. Transport through H, replay through C, provenance through P, and local naming through N are required before the generated closure can be consumed downstream.

Since the carrier has no coordinate for a pi-lambda theorem, selected countable chain, measure extension, completed sigma-algebra, or quotient equality of subsets, no consumer can substitute one of those host objects for the finite induction route. □

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$.

Constructive story: A $\mathsf{MonotoneClass}^\uparrow$ packet is generated from displayed set generators, increasing-union rows, decreasing-intersection rows, measurable-event admission, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$, seed)

Scope closed: Definition 4194.1, Definition 4194.2, and Definition 4194.3 seed the monotone closure route over displayed generators, union rows, intersection rows, measurable-event admission, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sigma-algebra completion, pi-lambda closure, measure extension, selected countable chains, quotient subset equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, increasing-union closure, decreasing-intersection closure, measurable-event admission, transport, provenance, and no-escape for BEDC.Derived.MonotoneClassUp.

The $\mathsf{MonotoneClass}^\uparrow$ dependency route exposes its generator, increasing-union, and decreasing-intersection rows as consumers of the measurable-event carrier in chapter 92, the generated-set boundary in chapter 3384, and the membership classifier discipline in chapter 3668. This route keeps the monotone-class closure packet visible to measure-facing consumers before they read any generated event as measurable.

Theorem 4194.5 (MonotoneClass SetLike measure dependency route)**.** *Every* $\mathsf{Measure}^\uparrow$ *or* $\mathsf{BorelSet}^\uparrow$ *consumer that reads an accepted* $\mathsf{MonotoneClass}^\uparrow$ *closure packet must first route the generator row* G*, increasing-union row* U*, decreasing-intersection row* I*, measurable admission row* A*, componentwise* hsame *transport row* H*, displayed* Cont *replay row* C*,* Pkg *provenance row* P*, and local* $\mathsf{NameCert}_{\mathsf{MonotoneClass}^\uparrow}$ *row* N *through the* $\mathsf{SetLike}^\uparrow$ *membership classifier of chapter 3668. Thus the consumer may use monotone closure only as a displayed membership-stable generated family over chapter 92 and chapter 3384, not as a host sigma-algebra completion, selected countable chain, measure-extension theorem, or quotient equality of subsets.*

Proof. Start with the carrier of Definition 4194.1. The generator row G names displayed subset requests, so any measure-facing or Borel-facing read must classify membership through the $\mathsf{SetLike}^\uparrow$ surface before it can treat the request as a generated event.

The closure rows U and I keep increasing-union and decreasing-intersection reads attached to that same membership classifier. The measurable admission row A is then the only route by which the generated family becomes visible to the $\mathsf{BorelSet}^\uparrow$ and $\mathsf{Measure}^\uparrow$ siblings.

Finally read the structural rows H, C, P, N. They transport, replay, source, and locally name the same finite route, so the consumer cannot replace the displayed membership-stable packet by a host sigma-algebra, a selected chain, a measure extension, or quotient subset equality. □

Chapter 4195

A Concrete Naming Certificate for $\mathsf{FanCompact}^{\uparrow}$

Following Brouwer and Bishop–Bridges, the $\mathsf{FanCompact}^{\uparrow}$ packet records constructive compactness of binary spread data by finite bars and prefix induction rather than by open-cover compactness. It sits beside the Boolean and list carriers of chapter 32 and chapter 36, the finite stream-window discipline of chapter 126, the compact-metric surface of chapter 129, and the downstream uniform-modulus consumer of chapter 369. The packet supplies a finite-means bridge from binary prefix rows to compact-uniform-continuity arguments without importing countable choice, host tree sets, open-cover compactness, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4195.1 (FanCompact carrier)**.** *A* $\mathsf{FanCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, L, T, W, R, I, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{Bool}^{\uparrow}$ *alphabet row, recording the two branch symbols available at each binary prefix.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *prefix row, recording finite binary words, prefix extension, and prefix concatenation as finite rows.*
3. *T is the finite binary prefix tree row admitted by the packet, with each node read only through B and L.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite prefix observed before a branch consumer reads the stream.*
5. *R is the bar-cover row, recording a finite* ProbeBundle *of prefix tests whose success covers every admitted stream route.*
6. *I is the finite-bar induction ledger, recording the leaf tests, backward prefix step, and root conclusion used by the fan argument.*
7. *H records componentwise* hsame *transport for $B, L, T, W, R, I, C, P, N$.*
8. *C records displayed* Cont *replay from a binary-prefix request through the stream window, bar row, induction ledger, compact-metric handoff, and uniform-modulus consumer boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{FanCompact}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, L, T, W, R, I, H, C, P, N$. No row stores a countable-choice selector, a host tree set, an open-cover compactness theorem, a selected infinite branch outside the displayed stream windows, or completeness of $\mathsf{Real}^\uparrow$.

Lean4 ✓ `BEDC.Derived.FanCompactUp.FanCompactTasteGate single carrier alignment`

Theorem 4195.2 (FanCompact finite bar induction)**.** *For an accepted* $\mathsf{FanCompact}^\uparrow$ *carrier $F = (B, L, T, W, R, I, H, C, P, N)$, every public bar-coverage read factors through the finite route*

$$B \longmapsto L \longmapsto T \longmapsto W \longmapsto R \longmapsto I.$$

The induction ledger I is the only row that may turn finite prefix coverage into a root conclusion, and the resulting handoff can be consumed by $\mathsf{CompactMetric}^\uparrow$ *and* $\mathsf{UniformModulus}^\uparrow$ *only through the displayed replay row C and provenance row P.*

Lean4 ✓ `BEDC.Derived.FanCompactUp.FanCompactCarrier finite bar induction`

Proof. Project the carrier. The binary alphabet row B and finite prefix row L are the only sources from which the tree row T can be read.

The stream window row W fixes the observed prefix before any bar test is applied. The bar row R is therefore a finite $\mathsf{ProbeBundle}$ over displayed prefixes, not a host set of infinite branches.

The ledger I supplies the leaf checks and the backward prefix step. Applying it produces the root bar conclusion for the admitted prefix tree while keeping every step attached to B, L, T, W, R.

Finally C and P route the result to compact-metric and uniform-modulus consumers. Since the excluded objects are not coordinates of F, the route exports no countable choice, host tree set, open-cover compactness, selected branch outside the window rows, or $\mathsf{Real}^\uparrow$ completeness. □

Theorem 4195.3 (FanCompact NameCert obligations)**.** *For the carrier of Definition 4195.1, the local* $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$ *surface consists exactly of carrier admission for $F = (B, L, T, W, R, I, H, C, P, N)$, Boolean alphabet exposure through B, finite prefix exposure through L, binary tree admission through T, finite stream-window exposure through W, bar-cover exposure through R, finite-bar induction through I, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 4195.1. The packet exposes exactly the ten rows $B, L, T, W, R, I, H, C, P, N$.

Read B, L, T, W, R, I in order. Boolean symbols and finite lists determine the prefix rows, the stream window fixes the observed branch segment, the bar row lists the finite tests, and the induction ledger supplies the only permitted root conclusion.

Transport, replay, provenance, and local naming are then read through H, C, P, N. These rows preserve the finite prefix route and do not add countable choice, host tree sets, open-cover compactness, Real completeness, or selected infinite branches. □

Closure status for $\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$**.**

Constructive story: A $\mathsf{FanCompact}^\uparrow$ packet is generated from Boolean alphabet rows, finite binary prefix rows, a finite prefix tree, stream-name windows, a finite bar-cover ProbeBundle, a finite-bar induction ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$, seed)

Scope closed: Definition 4195.1, Definition 4195.2, and Definition 4195.3 seed the finite fan-compactness certificate surface for compact-metric and uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FanCompact}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, countable choice, host tree sets, open-cover compactness, selected infinite branches outside displayed stream windows, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite bar-cover exposure, induction-ledger exactness, stream-window routing, compact-metric handoff, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FanCompactUp.

Chapter 4196

A Concrete Naming Certificate for $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$

Following Tait and Girard reducibility candidates, the $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ packet records a finite candidate-mediated normalization boundary for MetaCIC closed terms. It reads the audit surface of chapter 9383 through the existing normalization audit packet of chapter 3779, the critical-path ledger of chapter 765, and the normalization frontier of chapter 892. The packet names a bounded fuel row, closed-term row, candidate row, Pi and application adequacy rows, no-infinite-reduction row, transport, replay, provenance, and local naming; it does not assert global strong normalization for every closed MetaCIC term.

Definition 4196.1 (BoundedNormalizationCandidate carrier)**.** *A* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, F, E, A, \Pi, S, I, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed MetaCIC term row, recording the term, type, context boundary, and closedness evidence read by the audit map.*
2. *F is the bounded fuel row, recording the finite reduction budget and the visible step counter for this candidate route.*
3. *E is the reducibility-candidate evidence row, recording the candidate, closed-candidate, or candidate-set membership object consumed by the route.*
4. *A is the application adequacy row, recording the function-side Pi evidence, argument evidence, and application-candidate handoff.*
5. *Π is the Pi adequacy row, recording binder-side candidate data and neutral-normal evidence when a Pi route is inspected.*
6. *S is the strong-normalization projection row, downstream of E, A, Π and bounded by F.*
7. *I is the no-infinite-reduction projection row, downstream of S and still attached to the same candidate evidence.*
8. *R is the frontier-refusal row, recording requests for evidence-free closed-term normalization, unrestricted subject reduction, global confluence, quotient syntax equality, or host evaluator soundness.*

9. *H records componentwise* hsame *transport for $T, F, E, A, \Pi, S, I, R, C, P, N$.*
10. *C records displayed* Cont *replay from a normalization-audit request through the bounded candidate route.*
11. *P is the* Pkg *provenance over* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$*, chapter 9383,* MetaCICNormalizationAuditPacket $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F, E, A, \Pi, S, I, R, H, C, P, N$. It has no coordinate for unconditional closed-term normalization, evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluation soundness, or completed closed consistency.

Lean4 ✓ `BEDC.Derived.BoundedNormalizationCandidateUp.BoundedNormalizationCandidateTasteGate single carrier alignment`

Theorem 4196.2 (BoundedNormalizationCandidate NameCert obligations)**.** *For the carrier of Definition 4196.1, the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$ *surface consists exactly of carrier admission for $K = (T, F, E, A, \Pi, S, I, R, H, C, P, N)$, closed-term exposure through T, bounded fuel exposure through F, candidate evidence through E, application adequacy through A, Pi adequacy through Π, candidate-mediated strong-normalization projection through S, no-infinite-reduction projection through I, frontier refusal through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The closed term row T and bounded fuel row F give the finite normalization request, while E is the only row carrying candidate evidence.

Read the adequacy rows A and Π as refinements of the same candidate route. They do not create a second normalization source; they only describe how applications and Pi-shaped terms remain inside the candidate-mediated boundary.

The strong-normalization projection S is downstream of E, A, Π, F, and the no-infinite-reduction projection I is downstream of S. Requests that drop this evidence pass to the frontier row R.

Transport, replay, provenance, and local naming through H, C, P, N preserve these rows. Since the excluded global principles are absent from the carrier, no NameCert projection can export them. □

Theorem 4196.3 (BoundedNormalizationCandidate frontier handoff)**.** *Every public normalization, strong-normalization, or no-infinite-reduction read from an accepted* $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ *carrier factors through the finite route*

$$T \longmapsto F \longmapsto E \longmapsto (A, \Pi) \longmapsto S \longmapsto I \longmapsto R.$$

Thus the packet realizes the finite candidate, fuel, closed-term, Pi/application adequacy, and no-infinite-reduction frontier rows used by chapter 9383, without claiming global strong normalization or closing the subject-reduction and confluence boundaries.

Proof. Start with a public normalization read. The closed term row T records the inspected term, and the fuel row F records the finite budget, but neither row alone supplies strong normalization.

The read must pass through E. Candidate, closed-candidate, or candidate-set evidence is then refined by application adequacy A and Pi adequacy Π before the strong-normalization projection S is available.

The no-infinite-reduction projection I reuses the output of S. If a consumer asks for evidence-free normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency, the frontier row R keeps that request outside the exported packet.

The structural rows H, C, P, N replay and name this same route for downstream MetaCIC audit consumers. Therefore the handoff is a bounded candidate-mediated frontier object, not a global theorem about all closed MetaCIC terms. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$ packet is generated from a closed MetaCIC term row, bounded fuel row, reducibility-candidate evidence, application adequacy, Pi adequacy, candidate-mediated SN projection, no-infinite-reduction projection, frontier refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4196.1, Definition 4196.2, and Definition 4196.3 seed the finite bounded candidate route for MetaCIC normalization audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationCandidate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, global strong normalization, evidence-free closed-term normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Upgrade path: Obligation closure requires checked carrier admission, fuel-route exactness, candidate evidence exposure, Pi and application adequacy routing, SN projection, no-infinite-reduction projection, frontier refusal, transport, replay, provenance, and local naming for BEDC.Derived.BoundedNormalizationCandidateUp.

Chapter 4197

A Concrete Naming Certificate for $\mathsf{ConstructiveBaireThread}^{\uparrow}$

Following Baire, Oxtoby, and Bishop–Bridges, the $\mathsf{ConstructiveBaireThread}^{\uparrow}$ packet records the choice-free complete-metric thread used when L10 Real and Completion consumers read a nested dense-open construction through displayed finite data. It sits after the choice-free Baire-category route of chapter 4123, the complete-metric service of chapter 128, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Specker boundary of chapter 1406. The packet names the thread as finite NameCert data and does not add countable choice, a host generic point, quotient equality of real names, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4197.1 (Constructive Baire thread carrier)**.** *A* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, K, S, R, E, P, H, C, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *source row, exposing dense-open requests only as displayed finite obligations.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, carrying the nested-ball, radius, and limit-service data visible to the thread.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that schedules finite dense-open requests, nested-ball inspections, and Cauchy-window reads.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for centers, radii, and tolerance comparisons consumed by the thread.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Baire, complete-metric, stream, and regular-rational rows have been displayed.*
6. *P is the* $\mathsf{SpeckerSequence}^{\uparrow}$ *boundary row used by consumers that test the thread against a non-located or non-choice escape route.*
7. *H stores componentwise* hsame *transport for B, K, S, R, E, P and the structural rows.*
8. *C stores displayed* Cont *replay from a thread request through the dense-open row, complete-metric service, finite window, regular readback, Real seal, Specker boundary, transport, provenance, and local naming rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $B, K, S, R, E, P, H, C, N$. It has no coordinate for a countable-choice selector, a host generic point, an infinite intersection object, quotient equality of real names, a selected Specker limit, or ambient completeness outside the displayed complete-metric service.

Lean4 (definition) `BEDC.Derived.ConstructiveBaireThreadUp.ConstructiveBaireThreadUp`

Theorem 4197.2 (Constructive Baire thread NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier $T = (B, K, S, R, E, P, H, C, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{BaireCategoryChoiceFree}^{\uparrow}$ *row, the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-thread row, the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{SpeckerSequence}^{\uparrow}$ *boundary row, componentwise transport, displayed replay, and local naming. Every L10 Real or Completion consumer of the thread must read through these rows.*

Lean4 ✓ `BEDC.Derived.ConstructiveBaireThreadUp.ConstructiveBaireThreadCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 4197.1. The analytic rows are B, K, S, R, E, P, and the structural rows are H, C, N.

The Baire and complete-metric rows are read before any real-facing endpoint. The row B supplies only the choice-free dense-open obligations, while K supplies the displayed nested Cauchy service used by the thread.

The finite schedule and readback rows then control the observation route. The row S records the finite windows, R records the regular rational centers and radii, and E is available only after those rows have been displayed.

Finally P, H, C, N preserve the same boundary. Since the carrier has no selector, host generic point, infinite intersection object, quotient real equality, selected Specker limit, or ambient completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4197.3 (Constructive Baire thread nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, *or Specker-facing consumer of an accepted* $\mathsf{ConstructiveBaireThread}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireCategoryChoiceFree}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{SpeckerSequence}^{\uparrow}$$

The route exports a displayed thread for L10 Real and Completion consumers, but it does not export countable choice, a chosen generic point, quotient equality of real names, a selected Specker limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the named rows.*

Proof. Start with an accepted carrier $T = (B, K, S, R, E, P, H, C, N)$. The dense-open side is exactly B, so the thread reads the Baire obligation through the choice-free packet rather than through an arbitrary sequence of choices.

The complete-metric side is exactly K. It carries the displayed Cauchy-thread service, and the finite schedule S determines which dense-open and nested-ball observations can be replayed.

The real-facing endpoint is downstream of regular rational readback. The row R packages the centers, radii, and tolerance comparisons before E exposes the terminal real-name seal.

The Specker boundary P is read only as a refusal surface for hidden located-limit or choice escape. Transport through H, replay through C, and naming through N preserve the same finite route, so the consumer cannot reach any excluded selector, quotient, host generic point, or ambient completeness object. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBaireThread}^{\uparrow}$ packet is generated from a BaireCategoryChoiceFree source row, a CompleteMetric Cauchy-thread row, a finite StreamName schedule, a RegSeqRat readback, a RealUp seal, a SpeckerSequence boundary row, componentwise hsame transport, displayed Cont replay, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4197.1, Definition 4197.2, and Definition 4197.3 seed the choice-free complete-metric thread consumed by L10 Real and Completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBaireThread}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, selected generic points, host infinite intersections, quotient real equality, selected Specker limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open scheduling, complete-metric thread service, regular-rational readback, Real sealing, Specker nonescape, transport, replay, and local naming for the same finite packet.

Chapter 4198

A Concrete Naming Certificate for $\mathsf{Mollifier}^{\uparrow}$

Following Schwartz and Sobolev constructive analysis, the $\mathsf{Mollifier}^{\uparrow}$ packet records the finite smooth-kernel surface used when a weak or distributional object is regularized through displayed local data. It sits after the Schwartz-test carrier of chapter 3399, the Sobolev ledger of chapter 440, the weak-derivative handoff of chapter 2210, the distribution-derivative action route of chapter 3751, the continuous-map boundary of chapter 130, the terminal real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The packet names only the finite kernel, support-radius, normalization, convolution-window, transport, replay, provenance, and local naming rows; it does not import host integration, choice, quotient function equality, or ambient completeness.

Definition 4198.1 (Mollifier carrier)**.** *A* $\mathsf{Mollifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, N, P, C, H, L)$$

with the following displayed rows:

1. *S is the smooth-kernel row, naming the Schwartz-test or smooth compact-support profile together with the visible derivative-order windows consumed by regularization.*
2. *R is the support-radius row, recording the dyadic radius, local support inclusion, and finite locality budget.*
3. *N is the normalization row, recording the unit-mass ledger as displayed* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons rather than as a host integral.*
4. *P is the convolution-window row, recording the finite source window, target window, translation request, and scale request read by a Sobolev, weak-derivative, distribution-derivative, or continuous-map consumer.*
5. *C stores displayed* Cont *replay from a regularization request through the kernel, support, normalization, convolution-window, real, dyadic, and local naming rows.*
6. *H stores componentwise* hsame *transport for S, R, N, P, C, L and the sibling rows named by the request.*
7. *L is the local* $\mathsf{NameCert}_{\mathsf{Mollifier}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through S, R, N, P, C, H, L. It has no coordinate for host integration, a selected approximate identity, quotient equality of functions or distributions, completed test-function topology, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MollifierUp.MollifierTasteGate single carrier alignment`

Theorem 4198.2 (Mollifier NameCert obligations)**.** *For every accepted* $\mathsf{Mollifier}^{\uparrow}$ *carrier* $M = (S, R, N, P, C, H, L)$*, the local* $\mathsf{NameCert}_{\mathsf{Mollifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed smooth-kernel row, support-locality row, normalization ledger, convolution-window route, nonescape boundary, componentwise transport, displayed replay, provenance inherited from the named analysis siblings, and local naming. Every Sobolev, weak-derivative, distribution-derivative, continuous-map, real, or dyadic consumer of the packet must read through these rows.*

Proof. Carrier admission is projection from Definition 4198.1. The analytic rows are S, R, N, P, and the structural rows are C, H, L.

The support-locality and normalization obligations are separate displayed rows. The radius row R confines every kernel read to the named local window, while N exposes only the finite real and dyadic ledger needed for unit normalization.

The convolution route is the row P. It records the source window, target window, translation request, and scale request before any downstream consumer may treat the packet as a regularizing kernel.

Finally C, H, L replay, transport, and name the same rows. Since no host integral, selected approximate identity, quotient function equality, completed test-function topology, choice principle, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local certificate surface. □

Theorem 4198.3 (Mollifier convolution regularization route)**.** *Let* $M = (S, R, N, P, C, H, L)$ *be an accepted* $\mathsf{Mollifier}^{\uparrow}$ *carrier. Any Sobolev-facing or weak-derivative-facing regularization read reaches a convolution output only through the displayed finite route*

$$S \longmapsto R \longmapsto N \longmapsto P \longmapsto C \longmapsto H \longmapsto L.$$

The same route is the only route by which a distribution-derivative or continuous-map consumer may cite the mollifier packet. The consumer receives smooth-kernel, support-radius, normalization, convolution-window, $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, transport, replay, provenance, and local naming data, but it receives no host integration operator, no choice-selected approximate identity, no quotient equality of functions or distributions, and no ambient completeness theorem.*

Proof. Start with a Sobolev-facing or weak-derivative-facing regularization request. The request first reads the kernel row S and the support-radius row R, so the smooth profile and its locality budget are displayed before any convolution output is named.

The normalization row N is then read as a finite real and dyadic ledger. It supplies the unit-mass boundary needed by the packet without adding a host integral or an external measure-space completion.

The convolution-window row P fixes the source window, target window, translation request, and scale request. Distribution-derivative and continuous-map consumers use the same row, so their reads cannot bypass the displayed kernel and radius data.

Transport through H, replay through C, and local naming through L preserve that finite route. The excluded host integration, selected approximate identity, quotient equality, choice, and ambient completeness objects are not carrier rows, so they cannot be exported by the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}$**.**

Constructive story: A $\mathsf{Mollifier}^{\uparrow}$ packet is generated from a smooth-kernel row, support-radius row, normalization ledger, convolution-window row, displayed Cont replay, componentwise hsame transport, provenance through the named analysis siblings, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4198.1, Definition 4198.2, and Definition 4198.3 seed the finite regularization route consumed by Sobolev, weak-derivative, distribution-derivative, continuous-map, Real, and dyadic rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Mollifier}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host integration, a selected approximate identity, quotient equality of functions or distributions, completed test-function topology, choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support locality, normalization ledger exactness, convolution-window routing, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MollifierUp.

Chapter 4199

Arzela–Ascoli Family Compactness NameCert Construction

Following Arzela, Ascoli, and Bishop–Bridges constructive compactness, the ArzelaAscoliFamilyCompactness$^{\uparrow}$ packet records family compactness for equicontinuous continuous-map families as finite BEDC NameCert data. It sits between the Arzela–Ascoli carrier of chapter 412, the equicontinuity family surface of chapter 413, the compact metric source of chapter 129, the continuous-map graph surface of chapter 130, and the uniform-modulus consumer of chapter 369. The packet names the finite family-compactness route; it does not assert an ambient compact-open topology, open-cover compactness, countable choice, selected infinite subsequence choice, or host function equality.

Definition 4199.1 (Arzela–Ascoli family-compactness carrier)**.** *An* ArzelaAscoliFamilyCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, E, K, G, M, W, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the* ArzelaAscoli$^{\uparrow}$ *row naming the compact-continuity packet over source nets, target completion rows, finite family graphs, pointwise image rows, and handoff ledgers.*
2. *E is the* EquicontinuityFamily$^{\uparrow}$ *row naming the shared finite-center equicontinuity modulus for the family.*
3. *K is the* CompactMetric$^{\uparrow}$ *source row whose finite probe bundle and compact-metric coverage supply the source windows.*
4. *G is the* ContinuousMap$^{\uparrow}$ *family graph row, storing each displayed family member as graph and pointwise-modulus data.*
5. *M is the* UniformModulus$^{\uparrow}$ *handoff row, receiving only the shared finite-radius and tolerance rows extracted from E and K.*
6. *W is the family window row, listing the finite source probes, family indices, target observations, and diagonal readback windows visible to the compactness consumer.*
7. *B is the bounded image row, retaining the pointwise finite image bundles required before any family-compactness read is exported.*
8. *H records componentwise* hsame *transport for Arzela–Ascoli, equicontinuity, compact-metric, continuous-map, uniform-modulus, window, bounded-image, replay, provenance, and naming rows.*

9. *C records displayed* Cont *replay from a family-compactness request through A, E, K, G, M, W, B.*
10. *P is the* Pkg *provenance over* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$, $\mathsf{ArzelaAscoli}^{\uparrow}$, $\mathsf{EquicontinuityFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, E, K, G, M, W, B, H, C, P, N$. It has no coordinate for compact-open topology, host function extensionality, selected subsequence choice, countable choice, open-cover compactness, or ambient completion outside the displayed rows.

Lean4 ✓ `BEDC.Derived.ArzelaAscoliFamilyCompactnessUp.ArzelaAscoliFamilyCompactnessTasteGate`
`single carrier alignment`

Theorem 4199.2 (Arzela–Ascoli family-compactness NameCert obligations)**.** *For every accepted* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ *carrier*

$$F = (A, E, K, G, M, W, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission, Arzela–Ascoli packet exposure, equicontinuity-family exposure, compact-metric source-window exposure, continuous-map graph exposure, uniform-modulus handoff, finite family-window readback, bounded-image retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4199.1. The rows $A, E, K, G, M, W, B, H, C, P, N$ are the only readable coordinates of the packet.

The rows A and E supply the compact-continuity and equicontinuity-family surfaces. The compact-metric row K supplies finite source probes, the continuous-map row G supplies graph and pointwise-modulus reads, and the uniform-modulus row M receives only the shared finite tolerance handoff.

The family-window row W and bounded-image row B keep the diagonal and image-bundle reads visible before any compactness consumer can cite the packet. These rows are finite BEDC histories rather than host-level choices of a subsequence or function-space cover.

Finally H, C, P, N transport, replay, source, and name exactly that inventory. Since the carrier has no compact-open, function-extensionality, selected-subsequence, countable-choice, open-cover, or ambient-completion coordinate, none of those objects can be exported. □

Theorem 4199.3 (Arzela–Ascoli family-compactness finite route)**.** *Every family-compactness consumer of an accepted* $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ *packet factors through the sequence of displayed sibling rows*

$$\mathsf{ArzelaAscoli}^{\uparrow}, \quad \mathsf{EquicontinuityFamily}^{\uparrow}, \quad \mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The route reads finite source probes, family graph rows, shared equicontinuity radii, bounded target-image rows, and uniform-modulus handoff data before the local $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$ *row is available.*

Proof. Start with the obligation theorem, Definition 4199.2. It lists sibling exposure, source-window exposure, graph exposure, handoff, window readback, and bounded-image retention as required rows of the local NameCert surface.

Read the siblings in the displayed order. The Arzela–Ascoli row supplies the compact-continuity packet, the equicontinuity-family row supplies the shared radius ledger, the compact-metric row

supplies finite source probes, the continuous-map row supplies graph observations, and the uniform-modulus row receives the finite tolerance handoff.

The structural rows H, C, P, N preserve this sequence. A consumer therefore reaches family compactness only through the finite row inventory of the carrier and not through an ambient compactness theorem or selected host object. □

Falsifiable prediction: Every accepted $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ consumer read displays ArzelaAscoli, EquicontinuityFamily, CompactMetric, ContinuousMap, UniformModulus, finite family-window, bounded-image, transport, replay, provenance, and local NameCert rows before exporting a family-compactness packet.

Independence witness: arzelaascoli, equicontinuityfamily, compactmetric, continuousmap, uniformmodulus: none is bijective to $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ because the packet jointly records compact-continuity, shared equicontinuity, compact source windows, graph rows, uniform-modulus handoff, family-window readback, bounded-image retention, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}$.

Constructive story: An $\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}$ packet is generated from ArzelaAscoli, EquicontinuityFamily, CompactMetric, ContinuousMap, UniformModulus, finite family-window, bounded-image, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4199.1, Definition 4199.2, and Definition 4199.3 seed the finite family-compactness route for equicontinuous continuous-map families over compact metric sources.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArzelaAscoliFamilyCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compact-open topology, host function extensionality, selected infinite subsequences, countable choice, arbitrary open-cover compactness, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling exposure, compact source-window exposure, continuous-map graph exposure, uniform-modulus handoff, family-window readback, bounded-image retention, transport, replay, provenance, and local naming for BEDC.Derived.ArzelaAscoliFamilyCompactnessUp.

Chapter 4200

SR Route Dependency Audit NameCert Construction

The $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ packet realizes the dependency audit surface for the explicit bundle subject-reduction discharge route of chapter 9374. It reads the subject-reduction route packet of chapter 787, the route-audit packet of chapter 793, the route triangle of chapter 796, and the MetaCIC open-frontier ledger of chapter 762. The packet records which finite rows a downstream preservation consumer must display before treating the discharge route as available.

Definition 4200.1 (SR route dependency audit carrier)**.** *An* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (R, U, T, F, B, K, O, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{SubjectReductionRoute}^{\uparrow}$ *row naming the explicit bundle discharge route.*
2. *U is the* $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$ *row recording the audited four-row payload and socket handoff.*
3. *T is the* $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$ *row separating bundle and obstruction coordinates.*
4. *F is the* $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$ *row naming the open MetaCIC subject-reduction boundary.*
5. *B is the bundle-dependency row, listing the four discharge hypotheses as finite bundle coordinates.*
6. *K is the route-dependency row, listing the same four coordinates through explicit bundle publication.*
7. *O is the obstruction-boundary row stating that the audit packet does not inhabit either route for the closed-CIC substrate.*
8. *H records componentwise* hsame *transport for route, audit, triangle, frontier, dependency, obstruction, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a preservation request through R, U, T, F, B, K, O.*
10. *P is the* Pkg *provenance over* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$, $\mathsf{SubjectReductionRoute}^{\uparrow}$, $\mathsf{SubjectReductionRouteAudit}^{\uparrow}$, $\mathsf{SubjectReductionRouteTriangle}^{\uparrow}$, $\mathsf{MetaCICOpenFrontierLedger}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. N is the local $\mathsf{NameCert}_{\mathsf{SRRouteDependencyAudit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, U, T, F, B, K, O, H, C, P, N$. It has no coordinate for an inhabited SubjectReductionDischargeBundle*, a discharged dependent-codomain premise, or a closed subject-reduction theorem.*

Lean4 ✓ `BEDC.Derived.SRRouteDependencyAuditUp.SRRouteDependencyAuditTasteGate single carrier alignment`

Theorem 4200.2 (SR route dependency audit obligations)**.** *For every accepted* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *carrier*

$$A = (R, U, T, F, B, K, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SRRouteDependencyAudit}^{\uparrow}}$ *surface consists exactly of carrier admission, subject-reduction route exposure, route-audit exposure, triangle exposure, MetaCIC open-frontier exposure, bundle-dependency visibility, route-dependency visibility, obstruction-boundary retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4200.1. The dependency audit has exactly the route rows R, U, T, F, the dependency rows B, K, the obstruction row O, and the structural rows H, C, P, N.

The subject-reduction route and route-audit rows expose the common four-field discharge payload. The route triangle separates bundle use and obstruction retention, while the open-frontier ledger keeps the MetaCIC subject-reduction boundary visible.

The bundle and route dependency rows are compared only as two publications of the same four finite coordinates. They do not create a new discharge premise and do not inhabit the missing closed-CIC bundle.

Finally H, C, P, N transport, replay, source, and name only these rows. Since the carrier has no coordinate for an inhabited discharge bundle, dependent-codomain discharge, or closed subject-reduction theorem, none of those endpoints can be exported. □

Theorem 4200.3 (SR route dependency obstruction nonescape)**.** *Any preservation-facing consumer of an accepted* $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ *packet must read the obstruction row O together with the open-frontier row F before it can cite either the bundle dependency row B or the route dependency row K. The audit therefore exposes the discharge route as a finite dependency surface and does not turn it into an available closed-CIC subject-reduction theorem.*

Proof. Use Definition 4200.2. It requires route exposure, route-audit exposure, triangle exposure, open-frontier exposure, and obstruction retention as part of the same local NameCert surface.

The row F names the MetaCIC subject-reduction frontier, and O records that the closed-CIC substrate has not supplied the missing discharge inhabitant. Thus a consumer reading B or K reads dependency accounting, not an endpoint theorem.

The replay and provenance rows preserve this order. They carry the preservation request through the displayed route inventory and cannot erase F or O. □

Falsifiable prediction: A subject-reduction-route consumer refutes $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ if it obtains the bundle route without displaying subject-reduction-route, subject-reduction-route-audit, subjectreductionroutetriangle, and metacic-open-frontier-ledger rows.

Independence witness: subject-reduction-route, subject-reduction-route-audit, subjectreductionroutetriangle, metacic-open-frontier-ledger: none is bijective to $\mathsf{SRRouteDependencyAudit}^{\uparrow}$ because the audit packet jointly records route exposure, audited payload, bundle/obstruction separation, open-frontier retention, dependency rows, transport, replay, provenance, and local naming.

The Lean follow-up must provide Nontrivial SRRouteDependencyAudit$^{\uparrow}$ and FieldFaithful SRRouteDependencyAud instances. The instances must keep $R, U, T, F, B, K, O, H, C, P, N$ as separate displayed fields before any checked theorem cites the packet as a dependency-audit surface.

Closure status for NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$.

Constructive story: An SRRouteDependencyAudit$^{\uparrow}$ packet is generated from subject-reduction route, route-audit, route-triangle, open-frontier, bundle-dependency, route-dependency, obstruction, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$, seed)

Scope closed: Definition 4200.1, Definition 4200.2, and Definition 4200.3 seed the finite dependency audit surface for the bundle subject-reduction discharge route.

Formal verification: FormalStatus(NameCert$_{\text{SRRouteDependencyAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not inhabit the four MetaCIC subject-reduction discharge hypotheses, construct a discharge bundle, prove closed subject reduction, give Lean verification, or bridge check the packet.

Upgrade path: Obligation closure requires checked carrier admission, route exposure, route-audit exposure, triangle exposure, open-frontier exposure, bundle and route dependency visibility, obstruction nonescape, Nontrivial and FieldFaithful instances, transport, replay, provenance, and local naming for BEDC.Derived.SRRouteDependencyAuditUp.

Chapter 4201

A Concrete Naming Certificate for $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ packet records the universal property of separated reflection as finite BEDC NameCert data. It reads the separated completion route of chapter 1612, the separated metric classifier of chapter 1411, and the Hausdorff reflection boundary of **??**. The packet is a universal-property surface for factorization through separated targets; it does not introduce quotient objects, choice-selected representatives, host equality of reflected points, or ambient classical Hausdorff completion.

Definition 4201.1 (Separated reflection universal carrier)**.** *A* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, H, F, Q, T, R, P, N)$$

with the following displayed rows:

1. *S is the source premetric or uniform source row whose indistinguishability classifier is being reflected.*
2. *M is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row exposing the separated target comparison used by the reflection.*
3. *H is the* $\mathsf{HausdorffReflection}^{\uparrow}$ *row recording the Hausdorff boundary through which reflected equality is read.*
4. *F is the factorization row: every displayed map from S into a separated target factors through the reflected classifier row.*
5. *Q is the uniqueness row, comparing two displayed factorizations only by the separated target classifier supplied by M and H.*
6. *T records componentwise* hsame *transport for source, classifier, Hausdorff boundary, factorization, uniqueness, provenance, replay, and naming rows.*
7. *R records displayed* Cont *replay from a separated-target request through S, M, H, F, Q, T.*
8. *P is the* Pkg *provenance over* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, $\mathsf{HausdorffReflection}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, H, F, Q, T, R, P, N$. It has no coordinate for quotient carriers, selected representatives, host point equality, countable choice, hidden separated completions, or ambient Hausdorff replacement.

Lean4 ✓ `BEDC.Derived.SeparatedReflectionUniversalUp.SeparatedReflectionUniversalTasteGate single carrier alignment`

Theorem 4201.2 (Separated reflection universal NameCert obligations)**.** *For every accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier*

$$U = (S, M, H, F, Q, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}^{\uparrow}}$ *surface consists exactly of source admission, separated classifier exposure, Hausdorff boundary readback, displayed factorization, uniqueness by separated comparison, componentwise transport, displayed continuation replay, provenance, and local naming through $S, M, H, F, Q, T, R, P, N$.*

Proof. Project the carrier of Definition 4201.1. The rows $S, M, H, F, Q, T, R, P, N$ are the entire public packet.

Read S, M, H first. The source row supplies the pre-reflection indistinguishability data, while the separated metric and Hausdorff rows expose the target comparison used by every separated read.

The factorization and uniqueness rows are then forced through F and Q. Transport, replay, provenance, and naming preserve that same route through T, R, P, N.

Since the carrier has no coordinate for quotient carriers, selected representatives, host equality, choice, hidden separated completions, or ambient Hausdorff replacement, those objects cannot appear in the local certificate. □

Theorem 4201.3 (Separated reflection universal factorization)**.** *Let $U = (S, M, H, F, Q, T, R, P, N)$ be an accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier. Every displayed map from the source row S into a separated target whose comparison is read through M and H factors through the row F, and every consumer of the factorization must read the route*

$$S, M, H \longmapsto F$$

before it can inspect transport, replay, provenance, or local naming.

Proof. Apply the NameCert obligation theorem, Definition 4201.2. It identifies S, M, H, F as the whole factorization-facing prefix of the packet.

The source row is read before the separated target classifier, and the Hausdorff boundary is read before any reflected equality is consumed. The factorization row F therefore has no independent source of map data beyond S, M, H.

The structural rows T, R, P, N only transport, replay, source, and name that route. They cannot add a second factorization object or a choice-selected representative. □

Theorem 4201.4 (Separated reflection universal uniqueness)**.** *For every accepted* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *carrier $U = (S, M, H, F, Q, T, R, P, N)$, two displayed factorizations out of S are publicly equal only through the uniqueness row Q over the separated comparison supplied by M and H. The uniqueness read exports no quotient equality, host point equality, selected representative equality, choice principle, or hidden completion equality.*

Proof. Start from the factorization theorem, Definition 4201.3. It places both displayed factorizations on the same $S, M, H \mapsto F$ route.

Now read the uniqueness row Q. Its classifier is the separated target comparison supplied by M and H, so the equality of factorization witnesses is a displayed separated comparison, not a host equality claim.

Finally use the absence clauses in Definition 4201.1. No row carries quotient equality, selected representatives, choice, or hidden completion equality, so the stated separated uniqueness route exhausts the public claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}$.

Constructive story: A $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ packet is generated from a source premetric or uniform row, a separated metric classifier row, a Hausdorff reflection boundary, displayed factorization, uniqueness by separated comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4201.1, Definition 4201.2, Definition 4201.3, and Definition 4201.4 seed the universal-property route for separated reflection over finite BEDC rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedReflectionUniversal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient carriers, selected representatives, host equality of reflected points, countable choice, hidden separated completions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, separated classifier exposure, Hausdorff boundary readback, factorization, uniqueness, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedReflectionUniversalUp.

The rational interval locator also consumes the diagonal finite-window route of chapter 4080 before its rational lower and upper endpoint rows are exported to Real-facing consumers.

Chapter 4202

A Concrete Naming Certificate for $\mathsf{RationalIntervalLocator}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RationalIntervalLocator}^{\uparrow}$ packet records the finite interval-locator rows by which a Real-facing consumer asks for rational lower and upper bounds at a displayed tolerance. It sits between the located interval algorithms of chapter 1711 and chapter 3971, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet isolates the rational interval locator as finite BEDC data; it is not a host interval, quotient real representative, trichotomy principle, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Definition 4202.1 (Rational interval locator carrier)**.** *A* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. L_0 *is the rational lower-bound row, exposing the left endpoint candidate read at the requested precision.*
2. U_0 *is the rational upper-bound row, exposing the right endpoint candidate read at the same precision.*
3. I *is the interval-inclusion row, recording that the displayed Real-facing name is bracketed by* L_0 *and* U_0 *through finite rational comparisons.*
4. T *is the tolerance-request row, recording the dyadic or rational tolerance that the locator is asked to meet.*
5. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the tolerance request.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same finite window.*
7. A *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, naming the Real-facing read that receives the interval locator output.*
8. H *records componentwise* hsame *transport for lower-bound, upper-bound, inclusion, tolerance, window, readback, real-seal, replay, provenance, and local naming rows.*
9. C *records displayed* Cont *replay from a tolerance request through* L_0, U_0, I, T, S, R, A.

10. *P is the* Pkg *provenance over* $\mathsf{RationalIntervalLocator}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L_0, U_0, I, T, S, R, A, H, C, P, N$. It has no coordinate for host intervals, quotient-real representatives, trichotomy, countable choice, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4202.2 (Rational interval locator NameCert obligations)**.** *For every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint order, tolerance response, locator inclusion, regular-rational readback stability, Real-facing seal stability, componentwise transport, displayed replay, provenance, and ledger non-escape through $L_0, U_0, I, T, S, R, A, H, C, P,$*

Proof. Project the carrier of Definition 4202.1. The lower and upper endpoint rows L_0, U_0, the inclusion row I, the tolerance row T, the stream and readback rows S, R, the real-seal row A, and the structural rows H, C, P, N are the whole packet.

Endpoint order and locator tolerance are read from L_0, U_0, I, T. These rows record the rational bracket and the displayed precision request before any Real-facing consumer can inspect the output.

Readback stability is supplied by S, R, A. The stream window opens the finite observations, the regular rational row replays them as a readback, and the terminal real row seals the same finite route.

Finally H, C, P, N transport, replay, source, and name the displayed rows. Since no further coordinate is present, host intervals, quotient-real representatives, trichotomy, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside the ledger. □

Theorem 4202.3 (Rational interval locator Real readback)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it exports a rational bracket to a located-interval consumer.*

Proof. Let

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

be an accepted carrier. The tolerance row T supplies the finite precision request, and provenance in P attaches that request to the dyadic rational tolerance ledger.

The stream and regular-rational rows S, R then expose the finite window and readback route. The real-seal row A is downstream of the same rows, so a Real-facing consumer receives the interval only after the L10 route has been displayed.

The located-interval bisection and Bolzano consumers can read L_0, U_0, I only through the structural rows H, C, P, N. Therefore the route factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and cannot pass through host intervals, quotient reals, trichotomy, countable choice, or ambient real completeness. □

Theorem 4202.4 (Rational interval locator dyadic tolerance route)**.** *Every tolerance request for an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier first reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger through chapter 291. It then passes through the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before exposing the rational interval bracket L_0, U_0, I to a located-interval consumer.*

Proof. Let

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

be an accepted carrier. Project the tolerance row T and provenance row P. These rows identify the requested precision as a dyadic rational tolerance supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger rather than by host interval arithmetic.

Next project S, R, A. The StreamName window S opens the finite observations for the displayed tolerance, the RegSeqRat row R replays those observations as regular rational data, and the Real seal A receives only that finite readback route.

Finally apply Definition 4202.3. It already factors every Real-facing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, so the rational bracket rows L_0, U_0, I are exposed only after the dyadic tolerance route has been displayed. □

Theorem 4202.5 (Rational interval locator Real-seal lattice)**.** *Every Real-facing locator read of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier*

$$L = (L_0, U_0, I, T, S, R, A, H, C, P, N)$$

factors through the finite lattice

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before exposing L_0, U_0, I to located-interval and real-separability consumers. The lattice exposes only dyadic tolerance, finite stream windows, regular rational readback, and the terminal Real seal; it refuses host intervals, trichotomy, quotient reals, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a Real-facing locator read and project the accepted carrier. The tolerance row T and provenance row P attach the requested precision to the $\mathsf{DyadicRatCore}^{\uparrow}$ route, so no host interval object is available before the dyadic ledger is read.

Apply Definition 4202.4. The dyadic request opens the $\mathsf{StreamName}^{\uparrow}$ window row S, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal row A.

Apply Definition 4202.3. That theorem exposes the rational bracket rows L_0, U_0, I only after the same route has been displayed. The structural rows H, C, P, N preserve the route for located-interval and real-separability consumers, and the carrier contains no coordinate for trichotomy, quotient reals, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalIntervalLocator}^{\uparrow}$ packet is generated from rational lower-bound and upper-bound rows, interval inclusion, tolerance requests, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 4202.1, Definition 4202.2, Definition 4202.3, and Definition 4202.5 seed the finite rational interval locator route for Real-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalIntervalLocator}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval arithmetic, quotient-real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint order, tolerance response, locator inclusion, readback stability, Real-facing seal stability, transport, replay, provenance, and ledger non-escape for BEDC.Derived.RationalIntervalLocatorUp.

Theorem 4202.6 (Rational interval locator dyadic Real dependency route)**.** *Every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *interval read that is consumed by* $\mathsf{RealSeparability}^{\uparrow}$ *must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance rows, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the separability consumer can use the locator. The route cites the dyadic ledger of chapter 291, the real-name boundary of chapter 38, and the forward link of Definition 4202.7; it exports no quotient real equality, selected limit, host interval, countable-choice schedule, or ambient completeness.*

Proof. Project the locator carrier of Definition 4202.1. The endpoint rows L_0, U_0, interval row I, tolerance row T, stream window S, readback row R, Real seal A, and structural rows H, C, P, N are the whole accepted interval read.

The dyadic part is visible before the separability handoff. The endpoint and tolerance data are read through L_0, U_0, T and the $\mathsf{DyadicRatCore}^{\uparrow}$ provenance in P, while the finite window and readback rows S, R keep the route inside the StreamName and RegSeqRat corridor.

Now apply the Real-facing route of Definition 4202.3 and the RealSeparability forward link of Definition 4202.7. The seal row A must be displayed before any density or separability consumer can use the bracket, so the locator is consumed only after the $\mathsf{Real}^{\uparrow}$ boundary has been reached.

Since no carrier coordinate names quotient real equality, a selected limit, a host interval object, countable choice, or ambient completeness, the dependency route stops at the displayed dyadic endpoints, tolerance rows, readback row, and Real seal. □

Any subdivision-facing $\mathsf{RationalIntervalLocator}^{\uparrow}$ read must pass through the rational lower and upper bracket rows, the dyadic tolerance ledger, the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal before it is consumed by chapter 3873. The route exports only that displayed locator packet to the located-interval subdivision consumer; it does not supply a host interval, selected limit, quotient real representative, countable-choice schedule, or ambient completeness claim. The $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint rows supply the rational lower and upper locator consumed by the regular-Cauchy window lattice of chapter 4060. That consumption is allowed only after the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal rows are visible, so the edge remains a displayed L10 dependency rather than a host interval, quotient real, selected limit, countable-choice schedule, or ambient completeness claim.

Theorem 4202.7 (Rational interval locator RealSeparability forward link)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer of an accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *carrier reads the rational lower and upper locator rows* L_0, U_0*, the tolerance row* T*, the finite* $\mathsf{StreamName}^{\uparrow}$ *window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* A *of chapter 38 before any Real density read is exported through chapter 1809. The forward link supplies rational enumeration windows and dyadic tolerance rows; it exports no host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient completeness datum.*

Proof. Project the carrier rows of Definition 4202.1. The lower-bound row L_0, upper-bound row U_0, inclusion row I, tolerance row T, stream window S, regular readback R, Real seal A, and structural rows H, C, P, N are the whole locator packet.

First read the locator surface. The rational endpoint rows L_0, U_0 and the inclusion row I give the finite lower and upper bracket used by the downstream density request, while T identifies the dyadic or rational tolerance to be matched.

Next compose with the Real-separability route. The finite window S and readback R provide the rational enumeration surface consumed by chapter 1809, and the seal row A ensures that the density-facing consumer receives the data only after the $\mathsf{Real}^{\uparrow}$ boundary has been displayed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no row stores a host interval, selected dense point, quotient representative, countable-choice schedule, or ambient completeness witness, the forward link is a finite locator-to-separability dependency and nothing beyond that boundary. □

Theorem 4202.8 (Rational interval locator sibling route)**.** *Every accepted* $\mathsf{RationalIntervalLocator}^{\uparrow}$ *read that exports a locator packet consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *interval endpoint rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window readback rows, and the* $\mathsf{RealSeparability}^{\uparrow}$ *density rows before the locator boundary is available to any downstream consumer. The route records only the finite endpoint, readback, density, transport, replay, provenance, and local naming surface; it exports no host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier rows of Definition 4202.1. The endpoint rows are the displayed lower and upper rational bounds, while the tolerance and inclusion rows determine which dyadic interval request is being named.

Read the sibling rows in order. The dyadic endpoint data come through chapter 291, the finite window and readback route comes through chapter 301, and the density-facing consumer reads through chapter 1809 only after those rows have been displayed.

The structural rows then transport, replay, source, and name the same packet. Since no row contains a host interval, selected dense point, quotient real representative, countable-choice schedule, or ambient Real-completeness witness, the sibling route is exhausted by the listed finite dependencies. □

The $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ route passes through chapter 3491, chapter 3732, chapter 4181, chapter 3948, and chapter 4154 before the child files expose the lamination packet.

Chapter 4203

A Concrete Naming Certificate for HyperbolicGeodesicLamination$^{\uparrow}$

The HyperbolicGeodesicLamination$^{\uparrow}$ packet realizes the boundary-paired noncrossing leaf surface of chapter 9463 as finite NameCert data. It sits after the endpoint route of chapter 3491, the Busemann boundary row of chapter 3732, the visual-boundary read of chapter 4181, the cross-ratio comparison of chapter 3948, and the observer-shadow metric of chapter 4154. The packet names a finite lamination handoff, not a completed geodesic space, quotient boundary, physical foliation, global hyperbolic model, or unlogged field-faithfulness assertion.

Downstream boundary-shadow and observer-expansion consumers read the lamination packet through chapter 3813 or chapter 4154, because the noncrossing ledger supplies the leaf-compatibility row consumed by boundary-shadow routing.

Definition 4203.1 (Hyperbolic geodesic lamination carrier)**.** *A* HyperbolicGeodesicLamination$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, L, N, M, B, V, H, C, P, F, A)$$

with the following displayed rows:

1. *E is the endpoint-pair row, recording the two boundary-facing endpoints of each displayed leaf.*
2. *L is the geodesic-leaf row, attaching each endpoint pair to a finite geodesic-boundary route.*
3. *N is the noncrossing row, recording the finite compatibility ledger among the displayed leaves.*
4. *M is the hyperbolic-metric read row used when leaf separation or ordering is inspected.*
5. *B is the Busemann row, attaching the leaves to displayed boundary distance-difference data.*
6. *V is the visual-shadow row, recording the observer-facing visual shadow through which the lamination is read.*
7. *H records componentwise* hsame *transport for endpoint-pair, leaf, noncrossing, metric, Busemann, visual-shadow, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from a boundary-lamination request through E, L, N, M, B, V.*
9. *P is the* Pkg *provenance over* HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicGeodesicBoundary$^{\uparrow}$, HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, HyperbolicObserverShadowM BHist, hsame, Cont, *and* NameCert *rows.*

10. *F is the refusal row excluding host global laminations, quotient boundary identifications, physical foliations, hidden crossing tests, and unlogged TasteGate witnesses.*
11. *A is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, N, M, B, V, H, C, P, F, A$. It has no coordinate for completed geodesic spaces, ambient boundary quotients, host noncrossing predicates, physical laminations, or implicit Nontrivial *and* FieldFaithful *witnesses.*

Theorem 4203.2 (Hyperbolic geodesic lamination NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *carrier*

$$G = (E, L, N, M, B, V, H, C, P, F, A),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$ *surface consists exactly of endpoint-pair admission, geodesic-leaf admission, noncrossing ledger compatibility, hyperbolic-metric readback, Busemann attachment, visual-shadow exposure, componentwise transport, displayed replay, provenance, explicit refusal, and local naming through $E, L, N, M, B, V, H, C, P, F, A$.*

Proof. Project the carrier of Definition 4203.1. The endpoint, leaf, noncrossing, metric, Busemann, visual-shadow, transport, replay, provenance, refusal, and naming rows are the whole lamination packet.

The endpoint-pair and geodesic-leaf rows E, L are read first. They attach every displayed leaf to finite boundary-facing data rather than to an ambient set of complete geodesics.

The noncrossing and metric rows N, M then record the compatibility tests available to consumers. These tests are displayed rows, so the packet does not import a host noncrossing predicate.

The Busemann and visual-shadow rows B, V bind the leaf family to boundary and observer-facing routes. Finally H, C, P, F, A transport, replay, source, refuse exits, and name the same finite carrier, leaving completed global laminations, quotient boundaries, physical foliations, and implicit TasteGate witnesses outside the certificate. □

Theorem 4203.3 (Hyperbolic geodesic lamination boundary route)**.** *Every boundary-lamination consumer of an accepted* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ *carrier must pass through endpoint-pair, geodesic-leaf, noncrossing, hyperbolic-metric, Busemann, visual-shadow, transport, provenance, refusal, and local* NameCert *rows before it can export a lamination handoff.*

Proof. Let

$$G = (E, L, N, M, B, V, H, C, P, F, A)$$

be an accepted carrier. A boundary-facing read starts with E, L, so its endpoints and leaves are displayed before the consumer sees a lamination token.

The noncrossing row N and metric row M supply the finite compatibility and metric-read tests. The Busemann and visual-shadow rows B, V then attach those tests to the boundary and observer-facing surfaces cited by provenance.

Transport and replay through H, C preserve the displayed route, while P, F, A source, refuse exits, and name it. Hence any boundary-lamination handoff factors through the listed rows and cannot be claimed by a consumer that omits endpoint pairs, noncrossing evidence, hyperbolic metric reads, visual-shadow data, transport, provenance, refusal, or local naming. □

Falsifiable prediction: Any boundary lamination consumer omitting endpoint-pair, noncrossing, hyperbolic-metric, visual-shadow, transport, provenance, or local NameCert rows cannot claim the $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ handoff.

Independence witness: hyperbolic-geodesic-boundary, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-cross-ratio, hyperbolic-observer-shadow-metric: the geodesic-boundary packet names endpoint routes without the noncrossing ledger, the Busemann packet names distance-difference data without leaf compatibility, the visual-boundary packet names boundary readback without lamination rows, the cross-ratio packet compares boundary quadruples without owning geodesic leaves, and the observer-shadow metric packet reads visible shadows without endpoint-pair noncrossing. None is bijective to $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$.

The future `lean4/BEDC/Derived/HyperbolicGeodesicLaminationUp/TasteGate.lean` file must include explicit Nontrivial $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ instances. Those instances preserve $E, L, N, M, B, V, H, C, P, F, A$ as distinct fields before a checked carrier-alignment theorem can serve as a TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ packet is generated from endpoint-pair rows, geodesic-leaf rows, a noncrossing ledger, hyperbolic-metric reads, Busemann attachment, visual-shadow exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, explicit refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4203.1, Definition 4203.2, and Definition 4203.3 seed the finite boundary-paired noncrossing lamination route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicLamination}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed geodesic spaces, quotient boundary equality, physical foliations, host global laminations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint-pair admission, geodesic-leaf admission, noncrossing compatibility, metric readback, Busemann attachment, visual-shadow exposure, TasteGate nontriviality, field faithfulness, refusal, and local naming for BEDC.Derived.HyperbolicGeodesicLaminationUp.

The $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$ carrier is anchored in the sibling lattice by the endpoint route of chapter 3491, the Busemann rows of chapter 3732, the visual-boundary readback of chapter 4181, the cross-ratio comparisons of chapter 3948, and the observer-shadow metric data of chapter 4154. A lamination consumer must therefore read endpoint-paired leaves, boundary distance-difference rows, visual shadows, cross-ratio comparison provenance, and observer-shadow metric coordinates as displayed rows before treating the packet as a finite noncrossing lamination handoff.

Chapter 4204

A Concrete Naming Certificate for $\mathsf{BesovSpace}^{\uparrow}$

Following Besov and Triebel's constructive regularity scale, the $\mathsf{BesovSpace}^{\uparrow}$ packet records a finite smoothness-measurement surface between Sobolev weak-gradient data and Holder power-law control. It sits after the Sobolev ledger of chapter 440, the Holder continuity route of chapter 1766, the weak-derivative handoff of chapter 2210, the terminal real-name seal of chapter 38, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, and the dyadic tolerance ledger of chapter 291. The packet names only displayed smoothness windows, dyadic scale budgets, weak-action rows, real readback, transport, replay, provenance, and local naming; it does not import open-cover compactness, quotient function spaces, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4204.1 (Besov space carrier)**.** *A* $\mathsf{BesovSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Hd, W, R, Q, D, E, T, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, recording the weak-gradient and integrability ledger consumed by the regularity request.*
2. *Hd is the* $\mathsf{HolderContinuity}^{\uparrow}$ *boundary row, recording the visible power-law modulus data used when the Besov packet exposes pointwise smoothness control.*
3. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *action row, naming the finite test probes and derivative observations inspected by the scale request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational magnitude data attached to the selected smoothness windows.*
5. *Q is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, selecting the scale windows, test probes, and comparison requests visible to this packet.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scale, radius, and comparison budget attached to the same windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Hd, W, R, Q, D have been displayed.*
8. *T stores componentwise* hsame *transport for the regularity, window, tolerance, real-seal, continuation, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a Besov regularity request through the Sobolev, Holder, weak-derivative, regular-rational, stream, dyadic, and real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, Hd, W, R, Q, D, E, T, C, P, N$. It has no coordinate for an open-cover compactness theorem, quotient equality of function spaces, selected representatives, a completed smoothness scale, host distribution equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BesovSpaceUp.BesovSpaceTasteGate single carrier alignment`

Theorem 4204.2 (Besov space NameCert obligations)**.** *For every accepted* $\mathsf{BesovSpace}^{\uparrow}$ *carrier $B = (S, Hd, W, R, Q, D, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Sobolev row, Holder row, weak-derivative action row, regular rational readback row, stream-window schedule, dyadic scale ledger, terminal real seal, componentwise transport, displayed replay, provenance, and local naming rows. Every regularity, PDE, interpolation, or trace-facing consumer of the packet must read through those rows before it cites the Besov smoothness boundary.*

Proof. Carrier admission is projection from Definition 4204.1. The analytic rows are S, Hd, W, R, Q, D, E, and the structural rows are T, C, P, N.

The Sobolev, Holder, and weak-derivative rows have distinct roles. The Sobolev row supplies the weak-gradient and integrability ledger, the Holder row supplies the visible power-law boundary, and the weak-derivative row supplies the finite test-action observations.

The scale and readback rows then fix the measurable part of the packet. The stream schedule Q opens the finite windows, the dyadic ledger D records the scale and comparison budget, the regular rational row R reads the displayed magnitudes, and the real seal E is reached only after those rows have been inspected.

Finally T, C, P, N transport, replay, source, and name the same finite route. Since no open-cover compactness theorem, quotient function-space equality, selected representative, completed smoothness scale, host distribution equality, countable choice, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4204.3 (Besov space L10 handoff)**.** *Every L10 regularity consumer of an accepted* $\mathsf{BesovSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Sobolev}^{\uparrow} \longrightarrow \mathsf{WeakDerivative}^{\uparrow} \longrightarrow \mathsf{HolderContinuity}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{R}$$

The handoff exports displayed smoothness-window data, dyadic scale control, regular rational readback, terminal real seals, componentwise transport, continuation replay, provenance, and local naming. It exports no open-cover compactness theorem, quotient function space, selected representative, host equality of functions or distributions, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4204.2. They require every accepted regularity read to pass through the Sobolev, Holder, and weak-derivative rows before any downstream smoothness boundary is cited.

Next read the finite scale data. The stream schedule Q chooses the visible windows, while the dyadic ledger D attaches the scale and tolerance budget to those same windows.

The regular rational readback row R and terminal real seal E then report the displayed magnitude data. The structural rows T, C, P, N preserve the route under transport, replay, provenance, and local naming.

Thus an L10 consumer receives exactly the displayed chain from Sobolev and weak-derivative evidence through Holder, stream, dyadic, regular-rational, and real-seal rows. The excluded compactness, quotient, representative-selection, host-equality, choice, and ambient-completeness objects have no carrier coordinate and cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{BesovSpace}^{\uparrow}$ regularity consumer must read Sobolev, HolderContinuity, WeakDerivative, StreamName, DyadicRatCore, RegSeqRat, Real, hsame, Cont, Pkg, and local NameCert rows before citing the Besov smoothness boundary.

Independence witness: sobolev, holder-continuity, weak-derivative, real, streamname, regseqrat, dyadicratcore: $\mathsf{BesovSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries smoothness-source rows, pointwise power-law boundary data, weak-action observations, finite windows, dyadic scale budgets, regular rational readback, real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BesovSpace}^{\uparrow}$ packet is generated from Sobolev, HolderContinuity, WeakDerivative, RegSeqRat, StreamName, DyadicRatCore, Real, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4204.1, Definition 4204.2, and Definition 4204.3 seed the finite constructive regularity-space route over Sobolev, HolderContinuity, WeakDerivative, Real, StreamName, RegSeqRat, DyadicRatCore, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BesovSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, quotient function spaces, selected representatives, host equality of functions or distributions, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, smoothness-window exactness, dyadic-scale ledger stability, regular rational readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BesovSpaceUp.

Chapter 4205

A Concrete Naming Certificate for $\mathsf{LebesgueCoveringDimension}^{\uparrow}$

Following Lebesgue and Hurewicz–Wallman, the $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet records covering dimension through finite open-cover refinements with bounded overlap. It sits beside the finite refinement carrier of chapter 1853, the topology row of chapter 88, the metric-ball observation surface of chapter 57, and the finite radius ledger of chapter 452. The packet names only displayed cover rows, refinement rows, and order ledgers; it does not invoke arbitrary open-cover compactness, a selected subcover, or a host dimension theorem.

Definition 4205.1 (Lebesgue covering dimension carrier)**.** *A* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, U, V, R, O, B, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *source row together with the metric observation surface needed to read finite* $\mathsf{MetricBall}^{\uparrow}$ *cover cells.*
2. *U is a finite open-cover request row, displayed as named topology opens and local metric-ball witnesses.*
3. *V is a finite refinement row whose entries are open rows over the same source X.*
4. *R is the refinement-containment ledger, recording for each row of V the named row of U that contains it.*
5. *O is the order-at-most-n overlap ledger, recording that every displayed point-history read meets at most $n+1$ refinement rows.*
6. *B is the finite ball-inspection row tying the topology opens to* $\mathsf{MetricBall}^{\uparrow}$ *observations.*
7. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *radius ledger for the displayed finite cover window.*
8. *H records componentwise* hsame *transport for X, U, V, R, O, B, F and for the certificate rows.*
9. *C records displayed* Cont *replay from an open-cover request through finite ball inspection, finite-radius readback, refinement containment, and overlap checking.*
10. *P is the* Pkg *provenance over* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by transporting these displayed rows. It has no coordinate for a host open-cover family, a selected refinement, an external multiplicity oracle, or a classical covering-dimension theorem.

Lean4 ✓ `BEDC.Derived.LebesgueCoveringDimensionUp.LebesgueCoveringDimensionTasteGate single carrier alignment`

Theorem 4205.2 (Lebesgue covering dimension NameCert obligations)**.** *For every accepted* LebesgueCoveringDime *carrier*

$$L = (X, U, V, R, O, B, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$ *surface consists exactly of topology-source admission through X, finite open-cover admission through U, refinement-row admission through V, refinement containment through R, order-at-most-n overlap checking through O, metric-ball inspection through B, finite Lebesgue-number radius readback through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4205.1. The topology, cover, refinement, containment, overlap, ball-inspection, finite-radius, transport, replay, provenance, and naming rows are precisely the displayed coordinates.

The finite refinement obligation reads U, V, R, O over the source row X. Containment is stored as a row-to-row ledger, and the order bound is stored as the visible overlap count for finite point-history inspections.

The metric side is only observational. The rows B and F route finite open cells through $\mathsf{MetricBall}^{\uparrow}$ and $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ before a consumer may compare the packet with the sibling $\mathsf{CoveringDimension}^{\uparrow}$ surface.

Finally H, C, P, N preserve the same route by transport, replay, provenance, and local naming. Since no excluded host object occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4205.3 (Lebesgue finite-refinement route)**.** *Every* $\mathsf{CoveringDimension}^{\uparrow}$ *consumer of an accepted* $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{MetricBall}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto U, V, R, O, H, C, P, N.$$

Thus the dimension assertion is a displayed open-cover refinement with order-at-most-n overlap and finite radius readback. It exports no arbitrary open-cover compactness principle, selected subcover, host refinement theorem, or detached multiplicity oracle.

Proof. Begin with the NameCert obligations of Definition 4205.2. They force a consumer to read the source topology, the finite cover request, the refinement rows, the containment ledger, and the overlap ledger before any dimension claim can be exported.

Read the metric observation side next. The ball row B supplies only the displayed $\mathsf{MetricBall}^{\uparrow}$ inspections, while F supplies only the finite radius ledger already exposed by $\mathsf{FiniteLebesgueNumber}^{\uparrow}$.

The comparison with $\mathsf{CoveringDimension}^{\uparrow}$ is therefore horizontal: both packets share finite cover-refinement and overlap rows, but this chapter keeps the Lebesgue finite-radius route explicit. The structural rows H, C, P, N transport, replay, source, and name that finite route, so no host compactness or selected-refinement object can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ read is refuted if a consumer exports a covering-dimension bound without displaying TopologyUp source rows, finite open-cover rows, refinement-containment rows, order-at-most-n overlap rows, MetricBallUp inspections, FiniteLebesgueNumberUp radius data, hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Independence witness: coveringdimension, finite-lebesgue-number, topology, metric: no listed sibling supplies the whole $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet, because the packet binds finite topology cover rows, metric-ball inspection, finite-radius readback, refinement containment, overlap order, and NameCert provenance in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueCoveringDimension}^{\uparrow}$ packet is generated from TopologyUp source rows, finite open-cover rows, MetricBallUp observations, FiniteLebesgueNumberUp radius rows, finite refinement-containment ledgers, order-at-most-n overlap ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4205.1, Definition 4205.2, and Definition 4205.3 seed the finite-refinement L10 topology object over TopologyUp, MetricBallUp, CoveringDimensionUp, and FiniteLebesgueNumberUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueCoveringDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected subcovers, host dimension theorems, detached multiplicity oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, refinement containment, overlap-order exactness, metric-ball inspection, finite-radius compatibility, transport, replay, provenance, and local naming for BEDC.Derived.LebesgueCoveringDim

Chapter 4206

A Concrete Naming Certificate for $\mathsf{LagrangeInterpolation}^{\uparrow}$

Following Lagrange's interpolation formula, the $\mathsf{LagrangeInterpolation}^{\uparrow}$ packet records the finite route from distinct interpolation nodes and displayed values to a polynomial whose evaluations match those values. It consumes the polynomial coefficient-list surface of chapter 50, the matrix-style finite table discipline of chapter 49, and the field and rational arithmetic boundaries of chapter 45 and chapter 37. Its BEDC carrier is a finite BHist ledger: it names nodes, values, basis-polynomial construction, evaluation rows, uniqueness rows, hsame transport, Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row. It is not a host interpolation theorem, a quotient polynomial equality, or an ambient field-completion principle.

The history rows are read as ordinary BEDC certificate data in the sense of chapter 1366 and the typed naming discipline of chapter 9. The classical formula appears only through finite arithmetic ledgers; every consumer sees the displayed nodes, denominators, coefficient rows, and evaluation routes before it can use the interpolating polynomial.

Definition 4206.1 (Lagrange interpolation carrier)**.** *A* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *source is a finite* BHist *packet*

$$L = (N, V, D, B, E, U, H, C, P, Q)$$

with the following visible rows:

1. *N is a nonempty finite node row, with an index list and a displayed pairwise-distinctness ledger for the nodes.*
2. *V is the value row assigning one carried scalar value to each node of N.*
3. *D is the denominator ledger for the factors $x_i - x_j$ with $i \neq j$, routed through* $\mathsf{Field}^{\uparrow}$ *inverse rows and the rational arithmetic surface when the nodes are rational.*
4. *B is the basis-polynomial row, displaying the finite products*

$$\ell_i(X) = \prod_{j \neq i} \frac{X - x_j}{x_i - x_j}$$

   as normalized $\mathsf{Poly}^{\uparrow}$ *coefficient lists.*
5. *E is the evaluation row, recording that each ℓ_i evaluates to the Kronecker pattern on the displayed node list.*

6. *U is the polynomial assembly and uniqueness row: it records the finite sum $\sum_i V_i \ell_i$ and the finite comparison showing that any other polynomial of degree below the displayed node count with the same evaluations has the same normalized coefficient list.*
7. *H records componentwise* hsame *transport for the node, value, denominator, basis, evaluation, uniqueness, replay, provenance, and local-name rows.*
8. *C records the* Cont *routes that replay denominator formation, basis-polynomial multiplication, polynomial evaluation, and uniqueness comparison from the same displayed finite packet.*
9. *P is the* Pkg *provenance over* $\mathsf{LagrangeInterpolation}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *Q is the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *row.*

The carrier predicate is $\mathsf{LagrangeInterpolationLedger}(N, V, D, B, E, U, H, C, P, Q)$. *Two accepted packets classify together exactly when the ten displayed rows are transported by H and replay through the same routes named by C and P.*

Lean4 ✓ `BEDC.Derived.LagrangeInterpolationUp.LagrangeInterpolationTasteGate single carrier alignment`

Theorem 4206.2 (Lagrange interpolation NameCert obligations)**.** *For an accepted* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *packet $L = (N, V, D, B, E, U, H, C, P, Q)$, the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *obligation surface consists exactly of the finite carrier and classifier rows of Definition 4206.1, the denominator nonzero rows D, the basis-polynomial construction rows B, the evaluation rows E, the assembly and uniqueness rows U, the componentwise* hsame *transport H, the displayed* Cont *replay C, the* Pkg *provenance P, and the local naming row Q. No additional host polynomial function, determinant inverse, quotient equality of polynomials, infinite node family, or completion datum is exported.*

Proof. Read the carrier of Definition 4206.1. The rows N and V supply the finite interpolation table, while D records exactly the nonzero denominator data forced by distinct nodes.

The basis and evaluation content is held by B and E. These rows are typed over the same polynomial coefficient-list surface and the same scalar arithmetic provenance, so their comparisons do not introduce a second carrier.

The row U is the only place where the assembled polynomial and the degree-bounded uniqueness comparison are visible. The remaining rows H, C, P, Q transport, replay, source, and name the displayed packet. Since no further coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 4206.3 (Lagrange polynomial evaluation route)**.** *Every evaluation read exported by an accepted* $\mathsf{LagrangeInterpolation}^{\uparrow}$ *packet factors through the same finite route*

$$N, V, D \longmapsto B \longmapsto E \longmapsto U \longmapsto Q.$$

The route first verifies pairwise-distinct nodes and denominator rows, then builds normalized basis polynomials in $\mathsf{Poly}^{\uparrow}$, *evaluates those basis polynomials at the displayed nodes, assembles the interpolating polynomial, and finally exposes the local* $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$ *row. A consumer cannot bypass this route by appealing to an ambient polynomial function, a determinant computation over an unstated matrix, or an external field-completion argument.*

Proof. Start with an accepted packet $L = (N, V, D, B, E, U, H, C, P, Q)$. The denominator ledger D is downstream of the node row N and upstream of the basis row B, so each basis-polynomial read has already passed through the displayed distinctness and inverse witnesses.

Next, E evaluates the basis rows of B at the same finite node list. The route is replayed by C and transported by H, hence a transported packet keeps the same node, value, denominator, basis, and evaluation sequence.

Finally, U assembles the finite linear combination and records the degree-bounded uniqueness comparison before Q names the packet. Thus every polynomial-evaluation read available from the local certificate follows the displayed finite route and has no coordinate from which a bypassing host argument could be read. □

Falsifiable prediction: Every accepted $\mathsf{LagrangeInterpolation}^{\uparrow}$ read that claims an interpolating polynomial factors through finite node rows, value rows, denominator ledgers, normalized $\mathsf{Poly}^{\uparrow}$ basis rows, evaluation rows, uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{LagrangeInterpolation}^{\uparrow}$ packet is generated from finite node rows, value rows, denominator ledgers, normalized polynomial basis rows, evaluation rows, uniqueness rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4206.1, Definition 4206.2, and Definition 4206.3 seed the finite interpolation carrier, local NameCert obligation surface, and polynomial-evaluation route over displayed nodes, values, denominators, normalized basis polynomials, evaluation rows, uniqueness rows, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LagrangeInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host interpolation functions, ambient polynomial functions, determinant inverses outside the displayed ledger, quotient equality of polynomials, infinite node families, completion data, field-completion arguments, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, denominator nonzero routing, denominator nonzero exposure, basis-polynomial construction, evaluation rows, evaluation replay, assembly, uniqueness-row replay, basis and evaluation route transport, transport replay, route nonescape, provenance exposure, local naming, and TasteGate field faithfulness for BEDC.Derived.LagrangeInterpolationUp.

Chapter 4207

A Concrete Naming Certificate for $\mathsf{RieszFischer}^{\uparrow}$

Following Riesz and Fischer, the $\mathsf{RieszFischer}^{\uparrow}$ packet records the finite NameCert surface for the classical completeness route from square-summable coefficient windows to a Hilbert-facing limit object. It sits between the inner-product and norm rows of chapter 86, chapter 91, and chapter 89, and the completion handoff rows of chapter 128 and chapter 1527. The packet exposes the coefficient-window, inner-product, norm, completion, transport, replay, provenance, and local naming rows needed for a constructive Riesz–Fischer handoff; it does not introduce quotient L^2 classes, choice-selected limits, or an ambient Hilbert completeness theorem outside the displayed packet.

Definition 4207.1 (RieszFischer carrier)**.** *A* $\mathsf{RieszFischer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, I, H, N, M, U, T, C, P, L)$$

with the following displayed rows:

1. *W is the square-summable coefficient-window row, recording finite coefficient blocks, tail bounds, and the window schedule used by the packet.*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row that reads coefficient-window comparisons as inner-product estimates.*
3. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row naming the Hilbert-facing endpoint surface over which the coefficient windows are interpreted.*
4. *N is the* $\mathsf{Norm}^{\uparrow}$ *row induced by the displayed inner-product estimates.*
5. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-service row for the metric sequence determined by W, I, N.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row that packages the displayed Cauchy datum as a completion-facing unit route.*
7. *T records componentwise* hsame *transport for W, I, H, N, M, U and for the local structural rows.*
8. *C records displayed* Cont *replay from coefficient windows through the inner-product, norm, completion, and unit-handoff route.*
9. *P is the* Pkg *provenance over* $\mathsf{RieszFischer}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient L^2 classes, almost-everywhere representatives, countable-choice limit selection, ambient Hilbert completeness, or host equality of limits.

Lean4 ✓ `BEDC.Derived.RieszFischerUp.RieszFischerTasteGate single carrier alignment`

Theorem 4207.2 (RieszFischer NameCert obligations)**.** *For every accepted* $\mathsf{RieszFischer}^{\uparrow}$ *carrier* $R = (W, I, H, N, M, U, T, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the square-summable coefficient windows,* $\mathsf{InnerProduct}^{\uparrow}$ *comparison row,* $\mathsf{Hilbert}^{\uparrow}$ *source row,* $\mathsf{Norm}^{\uparrow}$ *metric row,* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-service row,* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *handoff row, componentwise transport, replay, provenance, and local naming;*
2. *square-summable exhaustion, requiring every Hilbert-facing completion read to begin with the displayed coefficient-window row* W*;*
3. *inner-product and norm compatibility, requiring* I *and* N *to expose the same finite comparison route before any metric-completion row is consumed;*
4. *completion handoff, requiring* M *and* U *to route the same Cauchy datum and not a selected external limit;*
5. *transport replay, requiring* T, C, P, L *to preserve the ordered route under* hsame*,* Cont*,* Pkg*, and* NameCert*;*
6. *ledger non-escape, excluding quotient* L^2 *classes, choice-selected limits, ambient Hilbert completeness, host equality of limits, and hidden measure-space representatives.*

Proof. Project the accepted carrier. The packet displays the square-summable windows, inner-product estimates, Hilbert-facing source, norm metric, complete-metric service, completion-unit handoff, transport, replay, provenance, and local name as the whole available surface.

Read the analytic route in order. The coefficient window W supplies the finite square-summability data; the rows I and N turn those windows into inner-product and norm comparisons; the row M consumes the resulting Cauchy datum; and U exposes the same datum to completion consumers.

Finally attach the structural rows. Componentwise transport and displayed replay preserve only W, I, H, N, M, U, while P and L source and name the packet. Since no carrier coordinate contains quotient classes, selected limits, ambient completeness, host equality, or hidden representatives, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$, seed)

Scope closed: Definition 4207.1 and Definition 4207.2 seed the finite Riesz–Fischer packet over square-summable coefficient windows, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RieszFischer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient L^2 classes, choice-selected limits, ambient Hilbert completeness, host equality of limits, hidden measure-space representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient-window exhaustion, inner-product and norm compatibility, completion handoff, transport replay, and ledger non-escape for BEDC.Derived.RieszFischerUp.

Constructive story: A $\mathsf{RieszFischer}^{\uparrow}$ packet is generated from finite square-summable coefficient windows, inner-product comparison rows, a Hilbert-facing source row, norm metric rows, a complete-metric Cauchy service, a Cauchy-CompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4208

A Concrete Naming Certificate for $\mathsf{RationalFastLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RationalFastLimit}^{\uparrow}$ packet records the Bishop-style presentation of a real name as rational observations with an explicit fast tail budget. It sits over the dyadic arithmetic ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real seal of chapter 38, the fast Cauchy real packet of chapter 1589, and the explicit modulus packet of chapter 1781. The carrier makes the rational fast-limit surface available to L10 Real and Completion consumers without quotienting streams, selecting representatives, or importing countable choice.

Definition 4208.1 (RationalFastLimit carrier)**.** *A* $\mathsf{RationalFastLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (Q, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational observation row, listing the finite rational approximants read by the current precision request.*
2. *D is the fast dyadic tail-budget row, recording the dyadic tolerance scale, comparison budget, and tail witness used to compare later rational observations.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly which rational observations may be opened by the request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from Q, D, S after the fast budget has bounded the visible tail.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the rational observation, dyadic budget, stream-window, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for rational observations, dyadic tail budgets, stream windows, regular readback, the Real seal, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite precision request through Q, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalFastLimit}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FastCauchyReal}^{\uparrow}$, $\mathsf{FastCauchyModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$ *row.*
The classifier compares accepted packets only through $Q, D, S, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected rational representatives, a host real limit, trichotomy, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.RationalFastLimitUp.RationalFastLimitTasteGate single carrier alignment`

Theorem 4208.2 (RationalFastLimit NameCert obligations)**.** *For every accepted* $\mathsf{RationalFastLimit}^\uparrow$ *carrier $L = (Q, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$ *surface consists exactly of rational observation admission, fast dyadic tail-budget exposure, finite stream-window service, regular rational handoff, Real seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must read the finite route*

$$Q, D, S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before treating the packet as a real-name boundary.

Proof. Project the carrier of Definition 4208.1. The rows Q, D, S are the only source rows available to a precision request: rational observations, the fast dyadic tail budget, and the finite window schedule.

The regular handoff is downstream of those rows. The dyadic budget D supplies the tolerance comparison used to bound the tail of the rational observations in Q, while S restricts the read to displayed finite windows.

The Real seal row E is reached only after R. The structural rows H, C, P, N transport, replay, source, and name the same displayed route, so no quotient stream equality, selected representative, countable choice, or ambient completeness principle is introduced by the certificate. □

Theorem 4208.3 (RationalFastLimit regular readback)**.** *Every* $\mathsf{RegSeqRat}^\uparrow$ *or* $\mathsf{Real}^\uparrow$ *consumer of an accepted* $\mathsf{RationalFastLimit}^\uparrow$ *packet factors through rational observations, the fast dyadic tail budget, finite stream windows, and the regular rational handoff. Equivalently, the consumer route is the displayed sibling chain*

$$\mathsf{DyadicRatCore}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{Real}^\uparrow$$

with $\mathsf{FastCauchyReal}^\uparrow$ *and* $\mathsf{FastCauchyModulus}^\uparrow$ *supplying the fast-limit interpretation of the dyadic budget row. The packet exports no direct rational-stream-to-real shortcut.*

Proof. Begin with an accepted packet $L = (Q, D, S, R, E, H, C, P, N)$. The rational observations are read from Q, but their use as a real-name approximation is constrained by the fast dyadic tail budget D.

The finite window row S determines which observations can be inspected. After Q, D, S are fixed, the handoff row R repacks the same data as a regular rational name, matching the route used by the $\mathsf{RegSeqRat}^\uparrow$ sibling.

The terminal seal E is downstream of R, and the fast interpretation of the budget is exactly the one exposed by $\mathsf{FastCauchyReal}^\uparrow$ and $\mathsf{FastCauchyModulus}^\uparrow$. Since the carrier has no coordinate for a host real limit or a quotient stream, every regular or Real consumer remains inside the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalFastLimit}^\uparrow}$.

Constructive story: A $\mathsf{RationalFastLimit}^\uparrow$ packet is generated from rational observation rows, a fast dyadic tail budget, finite StreamName windows, a RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalFastLimit}^{\uparrow}}$, seed)

Scope closed: Definition 4208.1, Definition 4208.2, and Definition 4208.3 seed the finite rational fast-limit route into regular rational and Real-facing readback.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalFastLimit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real limits, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fast tail-budget exactness, finite-window service, regular readback, Real seal nonescape, componentwise stability, provenance, and local naming for the displayed packet.

Chapter 4209

A Concrete Naming Certificate for $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ packet records the finite common-refinement route by which two displayed regular Cauchy names may be inspected through one shared target window before either reaches a $\mathsf{Real}^{\uparrow}$ seal. It sits after the single-schedule reindexing surface of chapter 1568, the finite observation schedules of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name boundary of chapter 38. The packet names a displayed paired refinement surface; it is not quotient stream equality, not a selected subsequence limit, not countable choice, not host real equality, not trichotomy, and not ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4209.1 (Regular Cauchy common refinement carrier)**.** *A* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two source* $\mathsf{StreamName}^{\uparrow}$ *schedule rows for the accepted regular Cauchy names;*
2. *I is the common target* $\mathsf{StreamName}^{\uparrow}$ *window through which both sources are inspected;*
3. *M_0, Q_0 are the monotone and cofinal reindex rows from I to S_0;*
4. *M_1, Q_1 are the monotone and cofinal reindex rows from I to S_1;*
5. *W is the common target-window budget row, recording that a single displayed finite request feeds both reindex routes;*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by the two reads at the budget named by W;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row showing that both inspected names reach the same regular rational tolerance surface;*
8. *A is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R$ have been consumed;*
9. *H records componentwise* hsame *transport for the two schedules, the common target window, both reindex routes, the tolerance, readback, Real seal, continuation, provenance, and local naming rows;*

10. *C records displayed* Cont *replay from a common-window request through the two monotone–cofinal routes and then through the shared tolerance, readback, and Real seal;*
11. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyCommonRefinementUp.RegularCauchyCommonRefinementTasteGate`
`single carrier alignment`

Theorem 4209.2 (Regular Cauchy common refinement obligation surface)**.** *For every accepted* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier* $K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$ *surface consists exactly of paired source-schedule admission, common-window admission, two monotone–cofinal reindex rows, shared dyadic tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, and ambient Real completeness.*

Proof. Project the carrier of Definition 4209.1. The projection exposes exactly the two source schedules, the common target window, two monotone–cofinal reindex pairs, the shared tolerance row, the regular readback row, the Real seal, and the structural rows H, C, P, N.

The common-window row W is the only target-window coordinate. A consumer therefore cannot inspect the two source schedules by unrelated requests; each request must replay through C and then through both displayed reindex routes.

The dyadic and regular rows are shared downstream rows. Since D and R are read only after the two reindex pairs have been displayed, the carrier records one tolerance surface and one regular readback surface rather than two unrelated limit witnesses.

Finally, A is downstream of the same paired route, while H, P, N preserve and package only the listed rows. No row remains from which to read quotient stream equality, a selected subsequence limit, countable choice, host real equality, trichotomy, or ambient Real completeness. □

Theorem 4209.3 (Regular Cauchy common refinement)**.** *Let S_0 and S_1 be accepted regular Cauchy source schedules with displayed finite schedules, and suppose the two schedules admit displayed monotone and cofinal reindex rows into a common target window I. Then there is an accepted* $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ *carrier*

$$K = (S_0, S_1, I, M_0, Q_0, M_1, Q_1, W, D, R, A, H, C, P, N)$$

whose $\mathsf{Real}^{\uparrow}$*-facing consumers factor through the same dyadic tolerance row D and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R. The construction uses only the two displayed schedule rows, the two monotone–cofinal reindex routes, the common target window, finite* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row; it does not use quotient stream equality, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two accepted regular Cauchy schedules. Each source schedule is available as a finite $\mathsf{StreamName}^{\uparrow}$ row, and each displayed monotone–cofinal route is the same kind of finite route isolated in chapter 1568.

Choose the displayed common target window I as the single inspection surface. The row W records the finite request made at that window, while M_0, Q_0 and M_1, Q_1 replay the request back to S_0 and S_1 without choosing a subsequence or comparing host streams.

Read the shared tolerance and regularity rows next. The $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the tolerance budget attached to W, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R is consumed after both reindex routes have been displayed. Hence both source reads reach the same finite regular rational surface before any Real-facing seal is inspected.

Transport and replay close the packet. Componentwise hsame transport identifies the rows that are meant to be shared, and Cont records the two finite route replays from I through the source schedules and forward to D, R, A. The Pkg and local NameCert rows then package exactly this surface.

By Definition 4209.2, the resulting packet contains no coordinate for quotient stream equality, selected subsequence limits, countable choice, host real equality, trichotomy, or ambient completeness. Therefore every $\mathsf{Real}^{\uparrow}$-facing consumer of the accepted common-refinement carrier factors through the displayed common window, the shared dyadic tolerance, and the shared regular readback. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyCommonRefinement}^{\uparrow}$ packet is generated from two source StreamName schedules, a common target window, two monotone and cofinal reindex route pairs, one DyadicRatCore tolerance row, one RegSeqRat readback row, a RealUp seal, componentwise transport, continuation replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4209.1, Definition 4209.2, and Definition 4209.3 seed the finite paired refinement route from two regular Cauchy schedules to a shared $\mathsf{Real}^{\uparrow}$-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCommonRefinement}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected subsequence limits, countable-choice selectors, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, paired monotone and cofinal reindexing discipline, shared dyadic tolerance preservation, RegSeqRat readback, Real seal factorization, and ledger refusal.

Chapter 4210

A Concrete Naming Certificate for $\mathsf{OlivierTheorem}^{\uparrow}$

Following Olivier's theorem for positive convergent series, the $\mathsf{OlivierTheorem}^{\uparrow}$ packet records the finite positive-term tail evidence by which a Real-facing series consumer may read vanishing terms without importing a host completeness theorem. It sits after the summable-tail modulus of chapter 1799, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a constructive tail handoff for positive series; it exports no selected sum, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4210.1 (Olivier theorem carrier)**.** *An* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, T, M, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the positive-term series row, exposing only the inspected nonnegative term windows and finite partial-sum comparison windows used by the current request.*
2. *T is the tail-window row, naming the finite interval of terms whose smallness is to be served.*
3. *M is the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row, assigning the requested positive-term tail window to a displayed summable-tail modulus before any regular rational readback is available.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reached only after A, T, M have displayed the finite tail evidence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational handoff and the summable-tail route have been replayed.*
6. *H records componentwise* hsame *transport for the positive-term, tail-window, modulus, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a positive-series tail request through A, T, M, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{OlivierTheorem}^{\uparrow}$, $\mathsf{SummableTailModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for selected sums, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OlivierTheoremUp.OlivierTheoremTasteGate single carrier alignment`

Theorem 4210.2 (Olivier theorem NameCert obligations)**.** *For every accepted* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier* $O = (A, T, M, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the positive-term series row, finite tail-window row, summable-tail modulus row, regular rational handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *positivity-window exposure, requiring* A *to supply the only term-sign and finite partial-sum data used by the tail request;*
3. *tail-window service, requiring* T *to name the inspected positive-term tail window before any downstream read can ask for vanishing terms;*
4. *modulus handoff, requiring* T *to pass through the* $\mathsf{SummableTailModulus}^{\uparrow}$ *row* M *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff* R *is available;*
5. *non-escape, requiring every terminal* $\mathsf{Real}^{\uparrow}$ *read to pass through* A, T, M, R, H, C, P, N *and excluding selected sums, quotient series equality, choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4210.1. The visible inventory is $A, T, M, R, E, H, C, P, N$, so the certificate has no detached row for a selected sum, quotient series equality, choice, host real equality, or ambient completeness theorem.

The positive-term and finite-window data are read before any Real-facing seal is available. The row A supplies the inspected positive term and partial-sum comparison windows, while T identifies the concrete tail window served by the request.

The tail window then passes through M. Thus finite positive-term tail evidence is converted into a displayed $\mathsf{SummableTailModulus}^{\uparrow}$ service before any $\mathsf{RegSeqRat}^{\uparrow}$ readback can occur.

Finally R, E, H, C, P, N supply regular readback, terminal sealing, transport, replay, provenance, and local naming for the same route. Since the excluded analytic objects are not carrier coordinates, they cannot be exported by the local certificate. □

Theorem 4210.3 (Olivier theorem tail handoff)**.** *Every Real-facing read of an accepted* $\mathsf{OlivierTheorem}^{\uparrow}$ *carrier factors finite positive-term tail windows through the ordered route*

$$\mathsf{OlivierTheorem}^{\uparrow} \longmapsto \mathsf{SummableTailModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently a consumer may use Olivier-style vanishing-term evidence only after the displayed summable-tail modulus row has served the finite tail window. It receives no selected sum, quotient series equality, choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $O = (A, T, M, R, E, H, C, P, N)$ and a Real-facing request for positive-series tail evidence. The only source row is A, and the only finite tail-window row is T.

The handoff is then forced through M. This makes the Olivier-style conclusion a finite modulus-served tail read, not a hidden selection of a sum or an equality of quotient series.

Only after M appears may the route enter R and then the terminal $\mathsf{Real}^{\uparrow}$ seal E. The rows H, C, P, N preserve the same displayed route under transport, replay it as continuation data, record provenance, and name it locally, so no excluded completeness or choice principle is introduced downstream. □

Falsifiable prediction: Every accepted $\mathsf{OlivierTheorem}^{\uparrow}$ Real-facing tail read factors through positive-term windows, a finite tail-window row, a SummableTailModulus row, RegSeqRat readback, Real sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: summable-tail-modulus, regseqrat, real: $\mathsf{OlivierTheorem}^{\uparrow}$ is not bijective to any listed sibling because it binds positive-term series windows, finite tail evidence, summable-tail modulus service, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{OlivierTheorem}^{\uparrow}$ packet is generated from positive-term series windows, finite tail-window evidence, a SummableTailModulus row, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4210.1, Definition 4210.2, and Definition 4210.3 seed the finite positive-series tail route over displayed summable-tail modulus, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OlivierTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, choice, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positivity-window exposure, tail-window service, modulus handoff, regular rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OlivierTheoremUp.

Chapter 4211

A Concrete Naming Certificate for $\mathsf{JacobiIteration}^{\uparrow}$

Following Jacobi's iterative method for linear systems, the $\mathsf{JacobiIteration}^{\uparrow}$ packet records the finite BEDC certificate surface of a displayed diagonal split, residual update, iteration schedule, and contraction-budget handoff. It consumes the matrix and linear-map rows read through $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$, the norm budget of $\mathsf{Norm}^{\uparrow}$, the Cauchy-orbit handoff of chapter 4224, and the fixed-point iteration interface already named by $\mathsf{FixedPointIteration}^{\uparrow}$. The packet names the local row discipline for one displayed iteration scheme; it is not a host convergence theorem, spectral-radius theorem, chosen solution vector, quotient sequence equality, or ambient completeness assertion.

Definition 4211.1 (Jacobi iteration carrier)**.** *A* $\mathsf{JacobiIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, D, R, S, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the matrix/vector system row, exposing the finite linear system and right-hand side only through the consumed* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{LinMap}^{\uparrow}$ *surfaces.*
2. *D is the diagonal split row, displaying the diagonal part and the residual part as finite matrix rows without assuming an external inverse beyond the displayed diagonal update.*
3. *R is the residual update row, recording the deterministic map from the current iterate and the system row to the next displayed iterate.*
4. *S is the iteration schedule row, listing the finite observation windows and the orbit positions that a consumer may inspect.*
5. *B is the displayed norm and contraction-budget row, exposing the finite descent inequalities read through* $\mathsf{Norm}^{\uparrow}$ *and the fixed-point iteration budget.*
6. *H transports A, D, R, S, B, C, P, N componentwise by* hsame*.*
7. *C replays the displayed* Cont *updates from a system read through the diagonal split, residual update, schedule, norm budget, and Cauchy-orbit handoff.*
8. *P records* Pkg *provenance over* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{Norm}^{\uparrow}$*,* $\mathsf{CauchySequenceCriterion}^{\uparrow}$*,* $\mathsf{FixedPointIteration}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, D, R, S, B, H, C, P, N$. It has no coordinate for a host solution vector, unrestricted spectral-radius argument, quotient sequence equality, selected limit, countable choice, or ambient Banach-space completeness.

Lean4 ✓ `BEDC.Derived.JacobiIterationUp.JacobiIterationTasteGate single carrier alignment`

Theorem 4211.2 (Jacobi iteration NameCert obligations)**.** *For every accepted* $\mathsf{JacobiIteration}^{\uparrow}$ *carrier* $J = (A, D, R, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the system, diagonal split, residual update, iteration schedule, contraction budget, transport, replay, provenance, and local naming rows;*
2. *diagonal split exposure, requiring D to be read after the matrix/vector system row A and before any residual update is inspected;*
3. *residual-update determinacy, requiring R to compute the next displayed iterate from the current iterate and the exposed split rows;*
4. *contraction-budget descent, requiring B to record the finite norm comparisons used by the schedule row;*
5. *Cauchy-orbit handoff, requiring the accepted schedule and budget rows to hand only the displayed orbit windows to* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *and* $\mathsf{FixedPointIteration}^{\uparrow}$ *consumers;*
6. *transport and replay preservation through H and C;*
7. *provenance and local naming through P and N;*
8. *non-escape from the displayed packet, excluding host convergence, selected limits, quotient sequence equality, unrestricted spectral-radius reasoning, countable choice, and ambient completeness.*

Proof. Project the carrier of Definition 4211.1. The only analytic rows are the system row A, diagonal split D, residual update R, schedule S, and budget row B.

The diagonal split is exposed before residual replay. Thus each accepted update reads the same displayed system and split rows, and the next iterate is determined by the finite residual row rather than by an external solver.

The budget row is also finite. It records the norm comparisons consumed by the schedule, and the schedule hands only those displayed orbit windows to the Cauchy-sequence and fixed-point iteration consumers.

Transport, replay, provenance, and naming preserve the same rows. Since the refused objects have no carrier coordinate in J, the local certificate exports only the displayed Jacobi iteration packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{JacobiIteration}^{\uparrow}$ packet is generated from a displayed matrix/vector system row, a diagonal split row, a residual update row, an iteration schedule, a norm/contraction budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$, seed)

Scope closed: Definition 4211.1 and Definition 4211.2 seed the finite Jacobi iteration certificate over matrix, linear-map, norm, Cauchy-sequence criterion, fixed-point iteration, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{JacobiIteration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host convergence, spectral-radius convergence, selected solutions, quotient sequence equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal split exposure, residual-update determinacy, contraction-budget descent, Cauchy-orbit handoff, transport, replay, provenance, and local naming for BEDC.Derived.JacobiIterationUp.

Chapter 4212

A Concrete Naming Certificate for $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$

The $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ packet realizes the apophatic socket-stack branch of Definition 9399.7 as finite boundary/refusal NameCert data. It sits over the typed socket taxonomy of chapter 1240, the socket-kind classifier of chapter 1058, the supply ledger of chapter 528, the far-end refusal socket of chapter 1061, and the named externality surface of chapter 543. The packet names a stacked visible-socket boundary; it does not supply the far end, discharge the refusal row, collapse socket kinds, or replace the named boundary by a host predicate.

Definition 4212.1 (Apophatic boundary socket stack carrier)**.** *An* $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, K_D, L, F, A, V, R, H, C, P, N)$$

with the following displayed rows:

1. T *is the* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$ *row naming the typed boundary class of each socket in the stack.*
2. K_D *is the* $\mathsf{SocketKindClassifier}^{\uparrow}$ *row classifying the visible socket kind at each stack layer.*
3. L *is the* $\mathsf{SupplySocketLedger}^{\uparrow}$ *row recording which visible supply request is being ledgered.*
4. F *is the* $\mathsf{ApophaticFarEndSocket}^{\uparrow}$ *row marking the far-end refusal boundary without supplying its content.*
5. A *is the* $\mathsf{ApophaticName}^{\uparrow}$ *row naming the externality position that the stack refuses to internalize.*
6. V *is the visible stack-order row, listing the finite order in which typed socket, classifier, ledger, far-end, and name rows are read.*
7. R *is the refusal row, recording that a consumer may factor through the visible socket and refusal rows but may not replace them by hidden host supply.*
8. H *records componentwise* hsame *transport for* $T, K_D, L, F, A, V, R, C, P, N$.
9. C *records displayed* Cont *replay through the visible stack order and refusal row.*
10. P *is the* Pkg *provenance over* $\mathsf{TypedGapSocketTaxonomy}^{\uparrow}$, $\mathsf{SocketKindClassifier}^{\uparrow}$, $\mathsf{SupplySocketLedger}^{\uparrow}$, $\mathsf{ApophaticFarEndSocket}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, K_D, L, F, A, V, R, H, C, P, N$. It has no coordinate for supplied far-end content, untyped sockets, anonymous supply, hidden host predicates, quotient equality, or discharged apophatic refusal.

Lean4 ✓ `BEDC.Derived.ApophaticBoundarySocketStackUp.ApophaticBoundarySocketStackTasteGate single carrier alignment`

Theorem 4212.2 (Apophatic boundary socket stack NameCert obligations)**.** *For every accepted* $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ *carrier $K = (T, K_D, L, F, A, V, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySo}}$ *surface consists exactly of carrier admission, typed socket exposure, kind-classifier exposure, supply-ledger exposure, far-end refusal exposure, apophatic name exposure, visible stack-order determinacy, refusal-row exactness, componentwise transport, displayed replay, provenance, and local naming. Every accepted stack consumer factors through the visible socket rows T, K_D, L, V and the refusal/name rows F, A, R before any public read is available.*

Proof. Project the carrier of Definition 4212.1. The socket-facing rows are T, K_D, L, V, and the refusal/name rows are F, A, R.

The stack order row V forces those rows to be read in a finite visible order. Thus a consumer cannot inspect the supply ledger or far-end boundary without first carrying the typed socket and classifier rows that identify the visible socket.

The refusal row R is downstream of F and A. It records refusal of internalization, so the public read is a boundary read rather than a supplied far-end witness or a hidden host predicate.

Transport, replay, provenance, and naming preserve the same finite packet. Since the refused objects have no carrier coordinate, the accepted stack factors through exactly the visible socket and refusal rows. □

Falsifiable prediction: Every accepted $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ consumer factors through the visible socket rows T, K_D, L, V and the refusal rows F, A, R; a purported consumer that omits either side is rejected by the local NameCert.

Independence witness: typedgapsockettaxonomy, socketkindclassifier, supplysocketledger, apophaticfarendsocket, apophaticname: TypedGapSocketTaxonomyUp records the typed class without the stack order, SocketKindClassifierUp names one kind row without the far-end refusal, SupplySocketLedgerUp records requests without the apophatic name, ApophaticFarEndSocketUp marks the refusal boundary without the full visible stack, and ApophaticNameUp names the externality without the socket-kind and ledger rows. None is bijective to the full $T, K_D, L, F, A, V, R, H, C, P, N$ carrier under NameCert-row preservation.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}$.

Constructive story: An $\mathsf{ApophaticBoundarySocketStack}^{\uparrow}$ packet is generated from typed socket taxonomy, socket-kind classifier, supply-ledger, apophatic far-end socket, apophatic name, visible stack-order, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4212.1 and Definition 4212.2 seed the finite visible-socket and refusal stack certificate over typed socket, classifier, ledger, far-end, name, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticBoundarySocketStack}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.ApophaticBoundarySocketStackUp.ApophaticBoundarySocketStackTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close supplied far-end content, hidden host predicates, quotient equality, discharged apophatic refusal, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible socket exposure, stack-order determinacy, refusal exactness, transport, replay, provenance, and local naming for BEDC.Derived.ApophaticBoundarySocketStackUp.

Chapter 4213

A Concrete Naming Certificate for $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ packet records the finite NameCert surface by which a monotone family of continuous maps on a compact metric source yields a compact-uniform tolerance handoff. It sits beside the Dini continuity packet of chapter 2196, the Dini theorem carrier of chapter 1387, the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-modulus surface of chapter 369, and the uniform-limit handoff of chapter 463. The packet uses finite cover and explicit probe rows; it does not invoke general open-cover compactness, a host supremum, selected minimum radius outside the carrier, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4213.1 (Dini monotone compact-uniform carrier)**.** *A* $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, F, M, L, R, V, U, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, carrying the metric source, compact finite-cover interface, and finite probe extraction route.*
2. *F is the continuous-map family row, recording the displayed maps on X and their local continuity data.*
3. *M is the monotone ledger, recording the family order and the visible monotone approach toward the limiting row.*
4. *L is the pointwise-limit row, recording the finite probe-wise route from family observations to the displayed limiting map.*
5. *R is the rational tolerance budget, storing the requested positive tolerance, cover-cell slices, and comparison rows.*
6. *V is the finite cover and* $\mathsf{ProbeBundle}$ *row extracted from X for the requested tolerance window.*
7. *U is the uniform-modulus and uniform-limit handoff row. It records the route from local continuity and monotone pointwise data through V and R to a single compact-uniform read.*
8. *H records componentwise* hsame *transport for source, family, monotone ledger, limit, tolerance, cover, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from compact-uniform requests through $X, F, M, L, R, V, U, H, P, N$.*
10. *P is the* Pkg *provenance over* DiniMonotoneCompactUniform$^{\uparrow}$, DiniContinuity$^{\uparrow}$, DiniTheorem$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, UniformLimit$^{\uparrow}$, Rat$^{\uparrow}$, BHist, ProbeBundle, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{DiniMonotoneCompactUniform}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $X, F, M, L, R, V, U, H, C, P, N$ and has no coordinate for host suprema, arbitrary open-cover compactness, quotient function equality, selected radii outside V, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.DiniMonotoneCompactUniformUp.DiniMonotoneCompactUniformTasteGate single carrier alignment`

Theorem 4213.2 (Dini monotone compact-uniform carrier obligations)**.** *For an accepted* DiniMonotoneCompactUn *carrier*

$$D = (X, F, M, L, R, V, U, H, C, P, N),$$

the local NameCert$_{\text{DiniMonotoneCompactUniform}^{\uparrow}}$ *surface consists exactly of compact metric source admission through X, continuous-map family admission through F, monotone visibility through M, pointwise-limit visibility through L, rational tolerance budgeting through R, finite cover and probe extraction through V, uniform-modulus and uniform-limit handoff through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4213.1. The first seven rows are the analytic data, and H, C, P, N are the structural certificate rows.

The monotone Dini route is forced by the displayed order. A compact-uniform request first reads the source and family rows X, F, then exposes monotone visibility through M and probe-wise limiting data through L.

The finite tolerance work is stored in R, V, U. The rational budget R is subdivided across the finite ProbeBundle row V, and the handoff row U records how those local reads assemble into one uniform-modulus and uniform-limit request.

Finally H, C, P, N preserve the same coordinates by transport, replay, provenance, and local naming. Since the excluded host principles have no carrier coordinate, they are not NameCert obligations of the packet. □

Theorem 4213.3 (Dini monotone compact finite-cover extraction)**.** *Every compact-uniform read accepted by a* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier must expose the compact source row X, the continuous-family row F, the monotone ledger M, the pointwise-limit row L, and the rational budget R before extracting the finite* ProbeBundle *cover row V. The extraction is a finite compact metric route; it is not general open-cover compactness, a selected global minimum radius, or a host supremum argument.*

Proof. Use the carrier obligations of Definition 4213.2. They place X, F, M, L, R upstream of the finite cover row V.

The compact source row X supplies the compact metric finite-cover interface. The continuous-map family row F and monotone ledger M determine which local continuity and monotone comparisons can be inspected on each displayed probe.

The pointwise-limit row L is then read only at probes supplied by the packet, and the rational budget R supplies the requested tolerance and cell slices. Hence V is extracted as finite packet data rather than by an external open-cover or supremum principle. □

Theorem 4213.4 (Dini monotone uniform budget handoff)**.** *For an accepted* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier, every consumer of the compact-uniform conclusion must route the requested tolerance through R, the finite probe cover V, and the uniform-modulus and uniform-limit handoff U, then through* hsame, Cont, Pkg, *and local* NameCert *rows. The exported handoff is only the finite BEDC route through* CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, UniformLimit$^{\uparrow}$, DiniContinuity$^{\uparrow}$, *and* DiniTheorem$^{\uparrow}$.

Proof. Begin with the finite-cover extraction of Definition 4213.3. It supplies cover cells and probe rows downstream from the compact metric source, continuous family, monotone ledger, pointwise limit, and rational budget.

The handoff row U consumes exactly that finite data. It first reads the local modulus information attached to the cells of V, then packages the assembled tolerance into the displayed uniform-limit route.

The structural rows H, C, P, N export no analytic data beyond that route. Therefore the consumer receives a compact-uniform Dini handoff through the named sibling packets, with no host function equality, hidden completion, selected radius outside the carrier, or open-cover compactness. □

Theorem 4213.5 (Dini monotone compact-uniform finite descent)**.** *For an accepted* DiniMonotoneCompactUniform *carrier*

$$D = (X, F, M, L, R, V, U, H, C, P, N),$$

every compact-uniform tolerance export descends through the pointwise monotone rows M, L on the finite ProbeBundle *cover V and the rational budget R before U may expose a single uniform read. The export cannot introduce a host supremum, selected minimum radius, open-cover compactness, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4213.1. The only rows between the compact source and the uniform handoff are the continuous family F, monotone ledger M, pointwise-limit row L, rational budget R, finite cover row V, and handoff row U.

Apply the finite-cover extraction theorem Definition 4213.3. It forces V to be read as finite ProbeBundle data after the pointwise monotone rows and the tolerance budget have been displayed.

Apply the budget handoff theorem Definition 4213.4. The row U can expose a single compact-uniform read only through the already displayed R, V route, and the structural rows H, C, P, N preserve that route without adding analytic data.

Row exhaustion now removes the excluded alternatives. The carrier has no coordinate for a host supremum, selected minimum radius, open-cover compactness, or ambient Real$^{\uparrow}$ completeness, so none of those objects can be exported by the finite descent. □

Theorem 4213.6 (Dini monotone ProbeBundle threshold)**.** *For an accepted* DiniMonotoneCompactUniform$^{\uparrow}$ *carrier, once the compact source row X, monotone ledger M, pointwise-limit row L, rational budget R, and finite* ProbeBundle *cover row V are displayed, every compact-uniform tolerance export factors through a single* ProbeBundle *threshold row before U can expose a uniform-modulus handoff.*

The threshold row is finite packet data assembled from X, M, L, R, V and transported through H, C, P, N. It is not a host supremum, an externally selected radius, an open-cover compactness argument, a quotient function equality, or an ambient Real$^{\uparrow}$ *completeness coordinate.*

Proof. Start with the carrier obligations of Definition 4213.2. They place the compact source row, monotone ledger, pointwise-limit row, rational budget, finite cover row, handoff row, and structural rows in the displayed carrier inventory.

Apply finite-cover extraction by Definition 4213.3. The requested tolerance first reads the compact source and rational budget, then extracts the finite ProbeBundle cover row V on which the pointwise monotone comparisons may be inspected.

Use the budget handoff of Definition 4213.4 and the finite descent of Definition 4213.5. Together they force the local monotone rows M, L and the rational budget R to be consumed on V before the handoff row U may export a single compact-uniform read.

The common finite read on V is the required threshold row. The structural rows H, C, P, N transport, replay, source, and name it without adding analytic data, and the carrier has no coordinate for a host supremum, selected radius outside the finite ProbeBundle, open-cover compactness, quotient function equality, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map family rows, monotone ledgers, pointwise-limit rows, rational tolerance budgets, finite ProbeBundle cover rows, uniform-modulus and uniform-limit handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4213.1, Definition 4213.2, Definition 4213.3, Definition 4213.4, and Definition 4213.5 seed the constructive compact-uniform Dini carrier surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniMonotoneCompactUniform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, arbitrary open-cover compactness, host suprema, quotient function equality, selected radii outside the finite ProbeBundle, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover extraction, monotone pointwise visibility, uniform budget handoff, transport, replay, provenance, and local naming for BEDC.Derived.DiniMonotoneCompactUniformU

Chapter 4214

A Concrete Naming Certificate for $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$

The $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ packet realizes the partition-core incidence kernel of Definition 9462.1748 as finite NameCert data. It sits over the partition carrier of chapter 201, the Boolean row discipline of chapter 3733, the simplicial support surface of chapter 238, the cellular update route of chapter 934, and the matroid rank boundary of chapter 202. The packet records finite block, quotient, lift, spectrum, and incidence rows; it is not a biological stochastic model, host graph oracle, quotient codon semantics, unrestricted spectral theorem, or hidden global scheduler.

Falsifiable prediction: Every accepted $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ read exposes the partition block row, quotient-size row, incidence-count row, lift-count row, kernel-spectrum row, Boolean support row, and local NameCert row before a partition-core consumer may cite the finite incidence certificate.

Independence witness: partition, booleanalgebra, simplicialcomplex, cellularautomaton, matroid: none alone is bijective with $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ because the packet jointly carries partition blocks, Boolean support tests, support-nerve incidence, cellular replay, rank-separation boundaries, quotient spectra, lift counts, transport, replay, provenance, and local naming.

Definition 4214.1 (Partition-core incidence certificate carrier)**.** *A* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, Q, A, L, S, E, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite partition-block row, recording the displayed block classes and their support names.*
2. *Q is the quotient-size row, recording the size-graded partition-core windows used by the incidence kernel.*
3. *A is the incidence-count row, recording finite adjacency or boundary incidences between displayed classes.*
4. *L is the lift-count row, recording codon-level or support-level lifts of each quotient incidence.*
5. *S is the kernel-spectrum row, recording the finite spectra exposed by the displayed quotient kernels.*

6. *E is the Boolean and simplicial support row, recording complement tests, face membership, and support-nerve incidence.*
7. *R is the rank and replay row, recording the matroid rank boundary and cellular replay needed by downstream partition-core consumers.*
8. *H records componentwise* hsame *transport for block, quotient, incidence, lift, spectrum, support, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from partition-core requests through $B, Q, A, L, S, E, R, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{Matroid}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, Q, A, L, S, E, R, H, C, P, N$. It has no coordinate for host graph reachability, biological stochastic semantics, quotient codon identity, unrestricted homology, hidden scheduler state, or an external spectral oracle.

Lean4 ✓ `BEDC.Derived.PartitionCoreIncidenceCertificateUp.PartitionCoreIncidenceCertificateTasteGate single carrier alignment`

Theorem 4214.2 (Partition-core incidence certificate obligations)**.** *For every accepted* PartitionCoreIncidenceCertifi *carrier*

$$I = (B, Q, A, L, S, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$ *surface consists exactly of partition-block admission through B, quotient-size exposure through Q, incidence-count exposure through A, lift-count exposure through L, finite kernel-spectrum exposure through S, Boolean and simplicial support exposure through E, rank and cellular replay exposure through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4214.1. The rows B, Q, A, L, S, E, R are the finite incidence data, and H, C, P, N are the structural certificate rows.

The incidence route first reads partition blocks and quotient sizes. Only then can a consumer inspect incidence counts, lift counts, and finite kernel spectra.

Support and replay data are also displayed. The Boolean and simplicial row E exposes the support tests, while R records rank separation and cellular replay for the same finite partition-core request.

The structural rows preserve exactly those coordinates. Since the excluded host semantics and unrestricted spectral or reachability principles have no carrier row, they cannot be obligations of the local NameCert surface. □

Theorem 4214.3 (Partition-core incidence TasteGate boundary)**.** *Any TasteGate realization for* $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *must provide* Nontrivial $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *and* FieldFaithful $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ *instances preserving $B, Q, A, L, S, E, R, H, C, P, N$ as separate displayed fields. In particular, the gate must distinguish a changed partition block, quotient-size row, incidence-count row, lift-count row, kernel-spectrum row, support row, or replay row before it admits two packets as the same incidence certificate.*

Proof. Use the carrier of Definition 4214.1. The rows B, Q, A, L, S, E, R play different roles: source blocks, quotient windows, incidences, lifts, spectra, support tests, and replay boundaries.

The obligations of Definition 4214.2 consume those rows in order. Collapsing two of them would erase either the source partition, a counted incidence, a lifted incidence, a spectral claim, or the support and replay boundary.

Nontriviality is witnessed at the paper surface by packets that vary one displayed row while keeping the others fixed. Field faithfulness must therefore preserve the row list as a visible TasteGate boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}$ packet is generated from partition-block rows, quotient-size rows, incidence-count rows, lift-count rows, finite kernel-spectrum rows, Boolean and simplicial support rows, rank and cellular replay rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4214.1, Definition 4214.2, and Definition 4214.3 seed the finite partition-core incidence certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PartitionCoreIncidenceCertificate}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PartitionCoreIncidenceCertificateUp.PartitionCoreIncidenceCertificateTasteGate\_single\`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, biological stochastic semantics, quotient codon identity, host graph reachability, unrestricted homology, an external spectral oracle, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, incidence-count exposure, lift-count exposure, finite spectrum exposure, support replay, Nontrivial PartitionCoreIncidenceCertificateUp, FieldFaithful PartitionCoreIncidenceCertificateUp, transport, provenance, and local naming for BEDC.Derived.PartitionCoreIncidenceCertificateUp.

Chapter 4215

A Concrete Naming Certificate for $\mathsf{CauchyMajorantSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyMajorantSequence}^{\uparrow}$ packet records the finite majorant-control surface used when a regular Cauchy name is read through explicit dyadic error budgets. It sits between the completion consumer of chapter 1691, the threshold discipline of chapter 292, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names one explicit Bishop-style majorant route; it is not a quotient stream, a selected limit, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4215.1 (Cauchy majorant sequence carrier)**.** *A* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, M, W, D, B, R, E, H, C, N)$$

with the following ten displayed fields.

1. *S is the source regular Cauchy name, read only through* $\mathsf{RegSeqRat}^{\uparrow}$ *rows.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row assigning displayed stage bounds to requested dyadic precisions.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window ledger opened to inspect the source name at the requested stages.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording radii, slack comparisons, and finite budget arithmetic.*
5. *B is the majorant sequence row, recording a displayed monotone bound for each inspected precision request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from S, M, W, D, B.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R has been displayed.*
8. *H records componentwise* hsame *transport for all source, modulus, window, dyadic, majorant, handoff, seal, replay, and naming rows.*
9. *C records displayed* Cont *replay through the same rows in consumer order.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$ *row, including* Pkg *provenance over* $\mathsf{FastCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, *and* Cont.

The classifier compares accepted packets only through $S, M, W, D, B, R, E, H, C, N$. No coordinate stores a quotient representative, a selected limit, countable-choice tail selection, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceTasteGate single carrier alignment`

Theorem 4215.2 (Cauchy majorant sequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMajorantSequence}$ *carrier* $J = (S, M, W, D, B, R, E, H, C, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$ *surface consists exactly of the carrier row* $S, M, W, D, B, R, E, H, C, N$*, classifier stability under componentwise* hsame*, dyadic ledger exposure, majorant exactness for the inspected precision requests,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, displayed* Cont *replay, local provenance, and exclusion of quotient streams, selected limits, countable choice, host real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4215.1. The ten fields are the source regular Cauchy row, threshold row, finite window ledger, dyadic tolerance ledger, majorant row, regular readback, real seal, transport, replay, and local naming-provenance row.

The classifier is stable because H transports each field componentwise and C replays the same consumer order. A related packet can therefore change only by the displayed hsame rows, not by replacing the source with a host sequence or by identifying quotient representatives.

The ledger and exactness obligations are carried by D and B. The dyadic ledger records every tolerance request and comparison budget, while the majorant row records the displayed monotone bound consumed by the handoff.

Finally R and E are downstream of those rows. Since the packet has no coordinate for quotient streams, selected limits, countable-choice tails, host real equality, or ambient completeness, those objects cannot discharge any local NameCert obligation. □

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceCarrier namecert obligations`

Theorem 4215.3 (Cauchy majorant sequence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{FastCauchyCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The handoff exports only the source row, displayed threshold, finite windows, dyadic tolerance ledger, majorant sequence row, componentwise transport, replay, provenance, and local name carried by the same packet.

Proof. Start with the NameCert obligations of Definition 4215.2. They force every accepted regular-rational read to expose the modulus row M, the finite windows W, and the dyadic ledger D before the handoff row R is available.

The majorant row B is then read as the exact control object for those displayed requests. It is tied to the same M, W, D rows, so it cannot be detached into an external tail selector or a host-level bounding function.

The structural rows preserve the route. Componentwise transport H, replay C, and local provenance in N keep the handoff on the same finite BEDC packet, which is the surface consumed by $\mathsf{FastCauchyCompletion}^{\uparrow}$. □

Lean4 ✓ `BEDC.Derived.CauchyMajorantSequenceUp.CauchyMajorantSequenceCarrier regseqrat handoff`

Theorem 4215.4 (Cauchy majorant sequence Real seal nonescape)**.** *Let $J = (S, M, W, D, B, R, E, H, C, N)$ be an accepted* $\mathsf{CauchyMajorantSequence}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *read of the terminal seal E must first pass through the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, majorant sequence row, and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff in the same carrier. The seal exports no quotient stream equality, selected Cauchy limit, countable-choice schedule, host real equality, or ambient completeness theorem.*

Proof. Read the carrier definition. The terminal real seal is the field E, and the definition places it after the source, modulus, window, dyadic, majorant, and regular-rational rows.

Apply Definition 4215.3. The regular handoff R is reached only through M, W, D, B, and every completion-facing consumer shares that displayed route.

Now inspect the classifier and ledger exclusions from Definition 4215.2. The accepted packet has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, or ambient completeness. The real seal therefore remains a terminal read of the finite majorant route rather than an escape to any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyMajorantSequence}^{\uparrow}$ packet is generated from a regular Cauchy source row, CauchyModulus threshold row, StreamName finite windows, DyadicRatCore tolerance ledger, displayed majorant sequence, RegSeqRat handoff, RealUp seal, componentwise hsame transport, Cont replay, and local NameCert provenance.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$, seed)

Scope closed: Definition 4215.1, Definition 4215.2, Definition 4215.3, and Definition 4215.4 seed the explicit majorant-control route for regular Cauchy names.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyMajorantSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host real equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, dyadic ledger exposure, majorant exactness, RegSeqRat handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMajorantSequ

Chapter 4216

A Concrete Naming Certificate for $\mathsf{MetricBaireSpace}^{\uparrow}$

Following Baire and Oxtoby, the $\mathsf{MetricBaireSpace}^{\uparrow}$ packet records the complete separable metric route through which a Baire-space consumer reads finite names. It sits after the metric carrier of chapter 4230, the completion service of chapter 128, the dense schedule of chapter 1458, the finite windows of chapter 126, the regular rational readback of chapter 301, and the prefix surface of chapter 334. The packet binds those rows as one complete-separable Baire naming surface without adding countable choice, a host topology, quotient point equality, or an external completion coordinate.

Definition 4216.1 (Metric Baire-space carrier)**.** *A* $\mathsf{MetricBaireSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, K, D, S, R, B, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the admitted point history, metric readbacks, distance classifier, and local metric naming.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row over the same metric carrier, displaying the Cauchy-window and limit-service route.*
3. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *row exposing a countable dense schedule as finite name data rather than as a choice function.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row scheduling finite inspections of dense names, Baire prefixes, metric balls, and Cauchy-window requests.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads rational windows for centers, radii, tolerances, and regular Cauchy approximants.*
6. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *ledger row recording finite-prefix and cylinder reads consumed by the Baire-space side of the packet.*
7. *H records componentwise* hsame *transport for M, K, D, S, R, B and the local certificate rows.*
8. *C records displayed* Cont *replay from a metric-ball, dense-name, Cauchy-window, or Baire-prefix request through the visible rows.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{MetricBaireSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the metric, complete-metric, separable-metric, stream, regular-readback, and Baire-prefix rows to one common surface. It has no coordinate for a selected dense sequence, a host completion, arbitrary open-cover semantics, quotient point equality, or a generic-point selector.

Lean4 ✓ `BEDC.Derived.MetricBaireSpaceUp.MetricBaireSpaceTasteGate single carrier alignment`

Theorem 4216.2 (Metric Baire-space NameCert obligations)**.** *For every accepted* $\mathsf{MetricBaireSpace}^{\uparrow}$ *carrier* $P = (M, K, D, S, R, B, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric, complete-metric, separable-metric, stream, regular-readback, Baire-prefix, transport, replay, provenance, and local-name rows; classifier transport to a common metric carrier before any completion, density, or Baire-prefix row is consumed; observation-ledger exactness through* S, R, B*; stability under the componentwise transports in* H*; displayed* Cont *replay through* C*; and provenance plus local naming through* P_0, N*. No naming obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4216.1. The rows M, K, D, S, R, B are the analytic surfaces, and H, C, P_0, N are the structural transport, replay, provenance, and naming surfaces.

The classifier route begins at M. The complete-metric row K, separable row D, and Baire-prefix row B are read only after the componentwise transport row has placed them over the same metric carrier.

The finite observation route is exhausted by S, R, B. Dense-name requests, Cauchy-window requests, metric-ball checks, and Baire-prefix reads all pass through the stream schedule and regular rational readback before any consumer can use them.

Finally replay through C and attach P_0, N. Since the carrier contains no coordinate for a selected dense sequence, host completion, arbitrary open-cover object, quotient point equality, or generic-point selector, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricBaireSpace}^{\uparrow}$ packet is generated from a MetricSpace source row, CompleteMetric handoff row, SeparableMetric dense schedule, StreamName finite windows, RegSeqRat readback, BaireSpace prefix ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$, seed)

Scope closed: Definition 4216.1 and Definition 4216.2 seed the complete-separable metric Baire naming surface consumed by downstream Polish and Baire chapters.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricBaireSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected dense sequences, host completions, arbitrary open-cover semantics, quotient point equality, generic-point selectors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common metric transport, completion service, dense schedule, StreamName windows, RegSeqRat readback, Baire-prefix ledger exactness, replay, provenance, and local naming for BEDC.Derived.MetricBaireSpaceUp.

Chapter 4217

A Concrete Naming Certificate for DNAWindowSixTriggerThreshold$^{\uparrow}$

The DNAWindowSixTriggerThreshold$^{\uparrow}$ packet realizes the random trigger-time law of Definition 9462.1539 as finite trigger-rank NameCert data. It sits beside the six-bit codon source window of chapter 2234, the boundary tile packet of chapter 3393, the six-tile coverage rows of chapter 3470, the trigger-algebra surface of chapter 3478, and the orbit inventory polynomial of chapter 4220. The packet names full-trigger distribution rows, trigger-time law rows, mean and quantile tables, transport, replay, provenance, and local naming as a finite combinatorial surface; it is not a biological stochastic model, an empirical mutation law, or an unbounded trigger search.

Definition 4217.1 (DNA window-six trigger-threshold carrier)**.** *A* DNAWindowSixTriggerThreshold$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, B, C, A, F, L, Q, H, R, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *row exposing the six-coordinate codon window.*
2. *B is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row supplying the trigger coordinates.*
3. *C is the* SixBoundaryTileCoverage$^{\uparrow}$ *row recording which trigger coordinates are covered by a finite external-codon draw.*
4. *A is the* WindowCoverageTriggerAlgebra$^{\uparrow}$ *row computing trigger rank and full-cube activation from the displayed coverage data.*
5. *F is the full-trigger distribution row for the finite external-codon process beyond $M^{\star} = \mathrm{NNR} \cup \mathrm{CUY}$.*
6. *L is the trigger-time law row recording the first time at which all four trigger coordinates are covered.*
7. *Q records the displayed mean and quantile table for the trigger time.*
8. *H records componentwise* hsame *transport for W, B, C, A, F, L, Q and the structural rows.*
9. *R records displayed* Cont *replay from an external-codon window through coverage, trigger rank, full-trigger distribution, trigger-time law, and table readback.*

10. *P is the* Pkg *provenance over* DNAWindowSixTriggerThreshold$^{\uparrow}$, CodonSixBitWindow$^{\uparrow}$, DnaWindowSixBoundaryT SixBoundaryTileCoverage$^{\uparrow}$, WindowCoverageTriggerAlgebra$^{\uparrow}$, *the random trigger-threshold source theorem,* BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed six-window, boundary, coverage, trigger-algebra, distribution, law, table, transport, replay, provenance, and local naming rows. It has no coordinate for empirical biology, stochastic fitness, hidden codon semantics, or an unrecorded search over external codon subsets.

Lean4 ✓ `BEDC.Derived.DNAWindowSixTriggerThresholdUp.DNAWindowSixTriggerThresholdTasteGate single carrier alignment`

Theorem 4217.2 (DNA window-six trigger-threshold full-cube law)**.** *For every accepted* DNAWindowSixTriggerThr *carrier $T = (W, B, C, A, F, L, Q, H, R, P, N)$, the full-cube trigger law is the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto \mathsf{SixBoundaryTileCoverage}^{\uparrow} \longmapsto \mathsf{WindowCoverageTriggerA}$$

The row F stores the full-trigger distribution, L stores the trigger-time law, and Q stores the mean and quantile table. Any accepted consumer of the average threshold 3.12998*, the deterministic threshold* 15*, the median threshold* 3*, or the* 99% *threshold* 7 *must read those values through the displayed route.*

Lean4 ✓ `BEDC.Derived.DNAWindowSixTriggerThresholdUp.DNAWindowSixTriggerThreshold full cube law`

Proof. Project the carrier of Definition 4217.1. The source rows W, B, C, A display the finite codon window, boundary tiles, coverage ledger, and trigger algebra before any distribution row is read.

The rows F, L, Q are the numerical threshold ledger. The full-trigger distribution row determines the law row, and the law row determines the mean and quantile table recorded by Q.

Transport and replay preserve that route through H and R. Provenance and local naming attach the source theorem and the NameCert row through P, N.

Since the carrier contains no empirical-biological coordinate, stochastic fitness model, hidden codon semantics, or unrecorded subset search, every full-cube threshold read is exhausted by the displayed finite route. □

Falsifiable prediction: Every accepted DNAWindowSixTriggerThreshold$^{\uparrow}$ read of the mean, deterministic threshold, median threshold, or 99% threshold factors through CodonSixBitWindow, DnaWindowSixBoundaryTile, SixBoundaryTileCoverage, WindowCoverageTriggerAlgebra, full-trigger distribution, trigger-time law, table, transport, replay, provenance, and local NameCert rows.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, window-coverage-trigger-algebra, boundary-orbit-inventory-polynomial: DNAWindowSixTriggerThreshold is not bijective to any listed sibling because it jointly carries the trigger-rank distribution, trigger-time law, mean and quantile table, transport, replay, provenance, and local naming as one threshold packet.

Future Lean TasteGate files for this packet must provide explicit Nontrivial DNAWindowSixTriggerThreshold$^{\uparrow}$ and FieldFaithful DNAWindowSixTriggerThreshold$^{\uparrow}$ instances before the threshold law can serve as a verified trigger-distribution consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}$.

Constructive story: A $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ packet is generated from CodonSixBitWindow source rows, DnaWindowSixBoundaryTile boundary rows, SixBoundaryTileCoverage rows, WindowCoverageTriggerAlgebra rows, full-trigger distribution rows, trigger-time law rows, mean and quantile table rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4217.1 and Definition 4217.2 seed the finite trigger-threshold packet for the window-six random trigger law.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim biological mechanism, empirical mutation dynamics, stochastic fitness, hidden codon semantics, unrestricted subset search, Lean verification, Nontrivial or FieldFaithful TasteGate instances, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, coverage exposure, trigger-rank distribution, trigger-time law, mean and quantile table exactness, transport, replay, provenance, local naming, Nontrivial DNAWindowSixTriggerThresholdUp, and FieldFaithful DNAWindowSixTriggerThresholdUp for BEDC.Derived.DNAWindowSixTriggerThresholdUp.

Chapter 4218

A Concrete Naming Certificate for $\mathsf{MetricCompletionKleisli}^{\uparrow}$

Following Mac Lane's Kleisli categories, and Bishop–Bridges and Kelley on metric completion, the $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet records the metric-specific bind consumer for completion-valued maps as finite BEDC data. It reads the metric completion carrier of chapter 1506, the metric adjunction route of chapter 3836, the Cauchy Kleisli consumer of chapter 1845, and the completion universal property of chapter 397. The chapter names a consumer route for bind-style completion reads; it is not a quotient completion, selected representative, countable-choice schedule, host equality of completed points, or ambient completeness theorem.

Definition 4218.1 (MetricCompletionKleisli carrier)**.** *A* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, A, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *row, displaying the source metric, completion carrier, separated comparison, dense-source access, and readback branch before any bind consumer is opened.*
2. *A is the* $\mathsf{MetricCompletionAdjunction}^{\uparrow}$ *row, displaying the unit, counit, triangle ledger, and metric universal mapping boundary read by the consumer.*
3. *L is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, recording the unit, bind, and reassociated-bind consumer route inherited from the L10 completion algebra.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, exposing the extension surface through which a displayed metric source map is consumed as a completion-valued map.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used by the metric completion and Cauchy Kleisli routes.*
6. *E is the public* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, A, L, U, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric completion row, metric adjunction row, Cauchy Kleisli row, universal-property row, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a bind consumer request through $M, A, L, U, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionKleisli}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{MetricCompletionAdjunction}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row.*
The classifier accepts only packets whose bind consumer first displays the metric completion, metric adjunction, Cauchy Kleisli, universal-property, regular readback, Real seal, hsame, Cont, Pkg, *and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.MetricCompletionKleisliUp.MetricCompletionKleisliTasteGate single carrier alignment`

Theorem 4218.2 (MetricCompletionKleisli NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier*

$$K = (M, A, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *surface consists exactly of metric-completion admission, metric-adjunction admission, Cauchy-Kleisli bind exposure, completion-universal-property exposure, RegSeqRat readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of K. No obligation may be discharged by a quotient completion, selected Cauchy representative, countable-choice selector, host equality of completed points, or ambient complete metric space.*

Proof. Project the carrier of Definition 4218.1. Its analytic rows are precisely the metric completion row M, the metric adjunction row A, the Cauchy Kleisli row L, the universal-property row U, the regular readback row R, and the Real seal row E.

The metric rows determine the order of access. A bind consumer opens M before it can read the adjunction data in A, and it opens A before the Cauchy Kleisli consumer L can hand off unit, bind, or reassociated-bind requests to the universal-property row U.

Transport and replay add no analytic coordinate. The rows H and C compare and replay only the displayed completion, adjunction, Kleisli, universal-property, readback, and seal data.

Finally P and N source and name the same finite packet. Since no listed row stores a quotient completion, selected representative, countable-choice selector, host equality, or ambient complete metric space, the displayed rows exhaust the NameCert surface. □

Theorem 4218.3 (MetricCompletionKleisli bind handoff)**.** *Every bind-style consumer read of an accepted* $\mathsf{MetricCompletionKleisli}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletionAdjunction}^{\uparrow} \longmapsto \mathsf{CauchyCompletionKleisli}^{\uparrow} \longmapsto \mathsf{CompletionUniversalProperty}$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the metric completion row M. It displays the source metric, completion carrier, separated comparison, dense-source access, and readback branch before a consumer can request completion-valued bind data.

Next read the adjunction row A. The unit, counit, triangle ledger, and metric universal mapping boundary are the only route by which the metric completion packet exposes extension data to a downstream consumer.

Attach the Cauchy Kleisli row L and the universal-property row U. The row L supplies the unit, bind, and reassociated-bind consumer surface, while U records the extension route that consumes a displayed source map as a completion-valued map.

The readback and seal rows close the route. The row R supplies the regular rational readback used by both completion and Kleisli sides, and E is the only public Real seal. The structural

rows H, C, P, N transport, replay, source, and name this same route, so no quotient completion, selected representative, choice schedule, host equality, or ambient completeness theorem can enter the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionKleisli}^{\uparrow}$ packet is generated from metric completion, metric adjunction, Cauchy Kleisli, completion universal property, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4218.1, Definition 4218.2, and Definition 4218.3 seed the metric-specific Kleisli consumer route for bind-style completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-adjunction exposure, Cauchy-Kleisli exposure, universal-property exposure, RegSeqRat/Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionKleisliUp.

Chapter 4219

A Concrete Naming Certificate for $\mathsf{LocatedRegularRealCut}^{\uparrow}$

Following Bishop and Bridges' constructive real analysis, the $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet records the finite boundary where regular-Cauchy real names expose located lower and upper rational cut rows before the terminal real seal is read. It sits beside the dyadic tolerance core of chapter 291, the finite stream schedule of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, the Dedekind cut carrier of chapter 436, and the located regular sequence route of chapter 2376. The packet is a finite NameCert bridge from regular-Cauchy observations to located cut boundaries; it does not claim Dedekind completeness, quotient equality of real names, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4219.1 (Located regular real cut carrier)**.** *A* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (L, U, M, D, R, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational cut row, recording inhabited rounded lower observations read through the finite cut-facing side of* $\mathsf{DedekindCut}^{\uparrow}$*.*
2. *U is the upper rational cut row, recording rounded upper observations attached to the same inspected real-name request.*
3. *M is the locatedness modulus row, binding every rational comparison request to a finite window on which either the lower row or the upper row is displayed with the required separation.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row supplying dyadic radii, bracket widths, midpoint requests, and comparison tolerances for the cut boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational observations from which the cut rows and locatedness modulus are read.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, U, M, D, R have been displayed.*
7. *H records componentwise* hsame *transport for lower cut rows, upper cut rows, locatedness modulus rows, dyadic tolerance rows, regular readback rows, Real seal rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a cut-boundary request through L, U, M, D, R, A, H.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedRegularRealCut}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row.*

The carrier accepts only the displayed lower row, upper row, locatedness modulus, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and naming rows. It has no coordinate for Dedekind completeness, quotient real equality, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedRegularRealCutUp.LocatedRegularRealCutUp`

Theorem 4219.2 (Located regular real cut obligations)**.** *For every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *carrier* $K = (L, U, M, D, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed lower cut row, upper cut row, locatedness modulus, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming row;*
2. *located lower and upper stability, requiring L and U to remain rounded, separated, and transported componentwise under the same* hsame *row;*
3. *regular Cauchy handoff, requiring every cut-boundary read to pass through R and the located regular sequence discipline of chapter 2376;*
4. *Real seal non-escape, requiring A to be downstream of L, U, M, D, R and not an independent completeness row;*
5. *ledger exactness, requiring P and N to name only L, U, M, D, R, A, H, C and to exclude quotient equality, selected limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4219.1. The displayed analytic rows are the lower cut row L, upper cut row U, locatedness modulus M, dyadic tolerance row D, regular readback row R, and Real seal row A.

The lower and upper stability clauses are componentwise. Transport through H preserves the two cut rows together with the same modulus and dyadic tolerance row, so the classifier cannot move a lower observation without moving the matching upper boundary data.

The regular Cauchy handoff is the route through R. The located regular sequence sibling supplies the finite windows and readback discipline before the cut boundary can be consumed, while D supplies the tolerance ledger on which the locatedness modulus is read.

Finally A, C, P, N seal, replay, source, and name only the displayed route. Since none of the carrier coordinates is a quotient, selected limit, trichotomy branch, choice schedule, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 4219.3 (Located regular real cut Cauchy boundary route)**.** *Every accepted* $\mathsf{LocatedRegularRealCut}^{\uparrow}$ *consumer factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedRegSeqRat}^{\uparrow},\ \mathsf{DedekindCut}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed finite rows before it can read the local $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$ *row. The route exports a located lower–upper cut boundary for a regular-Cauchy real name; it does not export Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 4219.2. A consumer first sees the dyadic tolerance row D, regular readback row R, and locatedness modulus M before the lower and upper cut rows can be treated as a located boundary.

The Dedekind cut sibling supplies the cut-facing row shape, while the located regular sequence sibling supplies the Cauchy-facing source and tail-window discipline. These rows meet only through the displayed finite packet K.

The Real seal row A is terminal for this route. Componentwise transport, continuation replay, package provenance, and local naming preserve the same displayed rows, so no consumer can bypass them by invoking quotient equality, a selected representative, trichotomy, choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRegularRealCut}^{\uparrow}$ read factors through displayed lower rational cut, upper rational cut, locatedness modulus, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: dyadicratcore, regseqrat, real, dedekindcut, located-regseqrat: no listed sibling is bijective to $\mathsf{LocatedRegularRealCut}^{\uparrow}$ because this packet alone binds lower and upper cut rows, a locatedness modulus, dyadic tolerance, regular readback, terminal real seal, transport, replay, provenance, and local naming into one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRegularRealCut}^{\uparrow}$ packet is generated from lower rational cut rows, upper rational cut rows, a locatedness modulus, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, seed)

Scope closed: Definition 4219.1, Definition 4219.2, and Definition 4219.3 seed the finite bridge from regular-Cauchy real names to located lower–upper cut boundaries.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedRegularRealCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, quotient real equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located lower and upper stability, regular Cauchy handoff, Real seal non-escape, and ledger exactness for BEDC.Derived.LocatedRegularRealCutUp.

Chapter 4220

A Concrete Naming Certificate for $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$

The $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet realizes the orbit and codon inventory polynomials of Definition 9462.1713 as finite BEDC NameCert data. It consumes the boundary-tile source of chapter 3393, the coverage packet of chapter 3470, the codon window source of chapter 2234, and the finite polynomial carrier of chapter 50. The packet records orbit rows, codon-count rows, coefficient ledgers, and marginal projections; it is not a biological stochastic model, an empirical code table, a hidden enumeration oracle, or quotient codon semantics.

Definition 4220.1 (Boundary orbit inventory polynomial carrier)**.** *A* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (O, K, E, M, H, C, P, N)$$

with the following displayed rows:

1. *O is the orbit table row, recording the finite orbit types classified by support size m, overlap defect δ, lift imbalance b, and orbit size.*
2. *K is the codon count table row, recording the weighted counts of lifted triggers over the same triples (m, δ, b).*
3. *E is the coefficient ledger row, storing the displayed coefficients of the orbit inventory polynomial $\mathcal{O}(u, v, w)$ and the codon inventory polynomial $\mathcal{C}(u, v, w)$.*
4. *M is the marginal projection row, recording the one-variable and bivariate specializations read from the coefficient ledger.*
5. *H records componentwise* hsame *transport for the orbit table, codon count table, coefficient ledger, marginal projections, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from an inventory query through O, K, E, M, H.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, *the source vision of Definition 9462.1713,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *row.*

The carrier accepts coefficient and marginal reads only through O, K, E, M, H, C, P, N. It has no coordinate for a hidden biological code table, probabilistic fitness model, unbounded orbit enumeration, quotient codon equality, or unstated polynomial coefficient source.

Lean4 ✓ `BEDC.Derived.BoundaryOrbitInventoryPolynomialUp.BoundaryOrbitInventoryPolynomialTasteGate`
`single carrier alignment`

Theorem 4220.2 (Boundary orbit inventory polynomial obligations)**.** *For every accepted* BoundaryOrbitInventoryP *carrier $I = (O, K, E, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$ *surface consists exactly of orbit-table admission through O, codon-count table admission through K, coefficient-ledger exactness through E, marginal-projection exactness through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4220.1. The orbit table and codon count table are the only source rows, while the coefficient ledger and marginal projection rows are the only polynomial-facing rows.

Coefficient exactness is read from E. Each displayed coefficient is determined by grouping the rows of O and weighting them by the count row K; no coefficient row is available without those two source tables.

Marginal exactness is read from M. A marginal projection is permitted only as a specialization of the coefficient ledger, so it cannot replace the orbit table or codon count table by a hidden enumeration.

The structural rows H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier lacks biological stochastic, quotient-codon, hidden-table, and unbounded-enumeration coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 4220.3 (Boundary orbit inventory coefficient determinacy)**.** *Every coefficient marginal exported by an accepted* $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ *packet is determined by the displayed orbit table O, codon count table K, coefficient ledger E, and marginal projection row M. Removing any coefficient row from E leaves a marginal query without a displayed source, and removing any orbit or codon count row leaves the corresponding coefficient without a displayed inventory witness.*

Proof. Read the NameCert obligations of Definition 4220.2. A coefficient query first exposes the orbit table O and codon count table K, then reads the coefficient ledger E.

The marginal projection row M is downstream of the ledger. It can specialize the displayed polynomial, but it cannot introduce a coefficient absent from E or reconstruct a missing source row absent from O or K.

Transport and replay preserve the same dependency order. Therefore a coefficient marginal is determined exactly by the displayed table and ledger rows, and deletion of any required row breaks the finite NameCert route. □

Falsifiable prediction: For $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$, coefficient marginals are determined by the displayed orbit table, codon count table, coefficient ledger, and marginal projection rows, and a marginal read fails if any required coefficient row is removed.

Independence witness: partition, hash, errorcode, dna-window-six-boundary-tile, six-boundary-tile-coverage: the polynomial coefficient ledger is not bijective to these sibling carriers because it binds orbit tables, codon count tables, coefficient rows, marginal projections, transport, replay, provenance, and local naming, while the siblings supply partitioning, digest, error coding, boundary-tile, or coverage surfaces without this coefficient inventory.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}$ packet is generated from an orbit table, codon count table, coefficient ledger, marginal projection row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4220.1, Definition 4220.2, and Definition 4220.3 seed the finite orbit-inventory polynomial certificate over boundary-tile and coverage rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryOrbitInventoryPolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological stochastic modeling, empirical code-table semantics, quotient codon equality, unbounded orbit enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-table admission, codon-count table admission, coefficient-ledger exactness, marginal-projection exactness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryOrbitInventoryPol

Chapter 4221

A Concrete Naming Certificate for $\mathsf{PrimitiveRecursive}^{\uparrow}$

Following Kleene's primitive recursive functions, the $\mathsf{PrimitiveRecursive}^{\uparrow}$ packet records finite recursion syntax and its local replay rows as BEDC data. It sits after the unary natural-number carrier of chapter 28, the list spine of chapter 36, the finite computability surface of chapter 199, the stream-window discipline of chapter 126, and the regular rational readback route of chapter 301. The chapter supplies a finite schedule substrate for later modulus and real-name consumers; it does not introduce minimization, partial recursive search, host functions, or unbounded choice.

Definition 4221.1 (PrimitiveRecursive carrier)**.** *A* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Z, S, \Pi, K, R, H, C, Q, N)$$

whose rows are read as follows:

1. *Z is the zero-function row.*
2. *S is the successor-function row.*
3. *Π is the finite family of projection rows, each with displayed arity and coordinate index.*
4. *K is the composition row, storing one outer code, its finite list of inner codes, and the arity ledger that makes the composition well typed.*
5. *R is the primitive-recursion scheme row, storing the base code, step code, displayed accumulator coordinate, and finite recursion spine over the* $\mathsf{Nat}^{\uparrow}$ *row.*
6. *H is componentwise* hsame *transport for zero, successor, projection, composition, recursion, replay, provenance, and local naming rows.*
7. *C is displayed* Cont *replay for evaluating a finite syntax tree by structural recursion through the rows Z, S, Π, K, R.*
8. *Q is the* Pkg *provenance row over* $\mathsf{PrimitiveRecursive}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Computable}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Z, S, \Pi, K, R, H, C, Q, N$. There is no coordinate for minimization, unbounded search, partiality, a host function graph, countable choice, or an external evaluator.

Lean4 ✓ `BEDC.Derived.PrimitiveRecursiveUp.TasteGate.PrimitiveRecursiveTasteGate single carrier alignment`

Theorem 4221.2 (PrimitiveRecursive NameCert obligations)**.** *For every accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *carrier*

$$P = (Z, S, \Pi, K, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$ *surface consists exactly of carrier admission for the zero, successor, projection, composition, primitive-recursion, transport, replay, provenance, and naming rows; arity compatibility for* Π, K, R*; structural replay of the displayed syntax tree through* C*; componentwise transport through* H*; and provenance through* Q, N*. No obligation is stored in a minimization row, partial recursive search row, host-function graph, or unbounded-choice selector.*

Proof. Inspect the carrier row by row. The base constructors are Z, S, Π, and their arity information is part of the displayed finite rows rather than a host-side function space.

For composite syntax, the row K supplies the finite list of inner codes and the outer code, while R supplies the base and step codes for the primitive-recursion scheme. Induction on the displayed syntax tree checks that every replay request descends to one of Z, S, Π or to a strictly displayed subcode of K or R.

The structural rows close the local certificate. Transport through H preserves the same constructor rows, replay through C evaluates only the displayed syntax tree, and Q, N record provenance and naming. Since the packet has no coordinate for minimization, partial recursive search, host functions, or unbounded choice, the listed rows exhaust the NameCert obligations. □

Theorem 4221.3 (PrimitiveRecursive composition closure)**.** *If an accepted* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *packet contains a composition row* K *whose outer code and finite list of inner codes are accepted by the same carrier, then the composed code is again read as a* $\mathsf{PrimitiveRecursive}^{\uparrow}$ *row through* K, H, C, Q, N*. The closure is internal to the finite displayed syntax and exports no minimization, partiality, host-function graph, or unbounded-choice principle.*

Proof. Proceed by induction on the finite list of inner codes recorded in K. The empty-list boundary is the displayed arity check for the outer code, and the single-step case appends one already accepted inner row to the composition ledger.

At each induction step the replay row C evaluates the inner rows before the outer row, and transport through H preserves the same arity and constructor classifiers. The provenance row Q and local naming row N then repackage the resulting finite syntax as a $\mathsf{PrimitiveRecursive}^{\uparrow}$ row.

No unbounded search is used in the induction. The construction consumes only the displayed finite list in K and the accepted subcodes already present in the carrier, so it cannot introduce minimization, partial recursive search, host functions, or unbounded choice. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4221.1, Definition 4221.2, and Definition 4221.3 seed the finite recursion-syntax certificate for schedule and modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveRecursive}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close minimization, partial recursive search, host functions, unbounded choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, arity compatibility, composition closure, primitive-recursion replay, transport, provenance, and local naming for the same finite recursion syntax.

Constructive story: The construction starts from zero, successor, and projection rows, closes finite composition through a displayed list spine, and closes primitive recursion by replaying the base and step codes over the Nat row under componentwise transport and local naming.

The $\mathsf{CauchyAcceleration}^{\uparrow}$ packet reads the StreamName, DyadicRatCore, RegSeqRat, and Real sibling surfaces of chapter 126, chapter 291, chapter 301, and chapter 38 before exporting its accelerated completion-facing route.

Chapter 4222

A Concrete Naming Certificate for $\mathsf{CauchyAcceleration}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyAcceleration}^{\uparrow}$ packet records the finite acceleration step by which an ordinary displayed Cauchy name is replaced by a cofinal regular rational name without changing the represented real. It sits after the criterion surfaces of chapter 392 and chapter 3576, the regularization route of chapter 376, the windowed name surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The acceleration packet consumes Cauchy tail comparison as substantive finite-window and dyadic-tolerance data through chapter 301, chapter 126, chapter 291, and chapter 38 before any accelerated Real-facing read is exported. The accelerated cofinal regular-rational name is consumed by chapter 5721 as the completion-existence witness route, with the read still passing through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet makes acceleration a named finite ledger: it does not select an ambient subsequence, quotient a Cauchy representative, or appeal to host completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4222.1 (Cauchy acceleration carrier)**.** *A* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row whose finite windows display the original rational Cauchy name.*
2. *T is the dyadic tolerance schedule, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*, assigning the requested precision budget to each inspected window.*
3. *α is the cofinal acceleration map, stored as a finite monotone ledger saying which source window is read for each accelerated precision request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *proof row for the accelerated rational name, including the regularity inequalities obtained by reading S through α and T.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the accelerated regular name reaches the same Real-facing boundary as the source name.*
6. *H records componentwise* hsame *transport for the source stream, tolerance schedule, cofinality map, regular proof row, Real seal, provenance, and local naming row.*

7. *C records displayed* Cont *replay from an acceleration request through S, T, α, R, E and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyAcceleration}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient representative of the original Cauchy name, a choice-selected subsequence, an ambient completeness theorem, or a host-side equality oracle for real numbers.

Theorem 4222.2 (Cauchy acceleration NameCert obligations)**.** *For every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row S, dyadic tolerance schedule T, cofinal map α, regular rational proof row R, and Real seal row E;*
2. *monotone cofinality, requiring α to preserve request order and to eventually reach every inspected source precision window;*
3. *dyadic tolerance preservation, requiring each accelerated request to read the same displayed tolerance budget through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before it reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *row;*
4. *classifier transport, requiring H and C to preserve the source stream, tolerance schedule, cofinality ledger, regular proof row, and Real seal under componentwise* hsame *and displayed replay;*
5. *ledger exhaustion through P and N, excluding quotient representatives, choice-selected subsequences, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and host real equality.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4222.1. The accepted packet exposes only the source stream S, tolerance schedule T, acceleration ledger α, regular readback R, Real seal E, and the structural rows.

Monotone cofinality is the public content of α. Since α is stored as a finite ledger over the inspected request window, each accelerated precision read names the source window it consumes and records that later requests do not move backward.

Dyadic tolerance preservation follows by reading T before the regular rational proof row. The $\mathsf{RegSeqRat}^{\uparrow}$ row R receives exactly the tolerance entries displayed by $\mathsf{DyadicRatCore}^{\uparrow}$, so the acceleration cannot import an unrecorded modulus or external precision oracle.

Classifier transport is componentwise through H and replayed through C. These rows preserve S, T, α, R, E without adding a second classifier surface.

Finally, P and N name precisely the displayed packet. Because the carrier contains no quotient representative, choice-selected subsequence, ambient completeness coordinate, or host real equality oracle, the five listed rows exhaust the local naming obligations. □

Theorem 4222.3 (Cauchy acceleration L10 handoff)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier $A = (S, T, \alpha, R, E, H, C, P, N)$ exports only the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

displayed by $S, T, \alpha, R, E, H, C, P, N$. Consequently every L10 consumer receives a $\mathsf{RegSeqRat}^{\uparrow}$*-compatible and* $\mathsf{Real}^{\uparrow}$*-compatible accelerated name, while quotient representatives of the original Cauchy name, choice-selected subsequences, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness remain unavailable.*

Proof. Project the accepted carrier and apply Definition 4222.2. The only public data are $S, T, \alpha, R, E, H, C, P, N$.

The source read begins at the $\mathsf{StreamName}^{\uparrow}$ row S. The tolerance read then passes through T and α, so every accelerated precision request is attached to an explicit dyadic budget and a displayed source window.

The regular rational handoff is exactly R, and the Real-facing boundary is exactly E. Transport and replay through H and C preserve that route for downstream consumers.

Since P and N exhaust the packet and no forbidden coordinate appears in Definition 4222.1, the exported object is only the displayed accelerated name compatible with $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. □

Theorem 4222.4 (Cauchy acceleration rectangular window route)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier* $A = (S, T, \alpha, R, E, H, C, P, N)$ *reuses the same* $\mathsf{StreamName}^{\uparrow}$ *finite windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows when producing the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the terminal* $\mathsf{Real}^{\uparrow}$ *handoff. The cofinal ledger* α *may refine the order of precision requests, but it cannot replace the displayed window route recorded by* S, T, H, C, P, N*.*

Proof. Project the rows $S, T, \alpha, R, E, H, C, P, N$. The source windows are recorded in S, and the dyadic tolerance entries are recorded in T before the accelerated regular rational row R or the Real-facing seal E is consumed.

Apply Definition 4222.2. The cofinality row α is an order ledger over displayed source windows, while dyadic tolerance preservation forces each accelerated request to read the same T row before it reaches R.

Now apply Definition 4222.3. Transport and replay through H and C preserve the route named by P and N, so no quotient representative, selected subsequence outside the ledger, or ambient completeness row is exported. □

Theorem 4222.5 (Cauchy acceleration cofinal-ledger determinism)**.** *Let*

$$A = (S, T, \alpha, R, E, H, C, P, N)$$

and

$$A' = (S, T, \alpha', R', E', H, C, P, N)$$

be accepted $\mathsf{CauchyAcceleration}^{\uparrow}$ *carriers sharing the same displayed source stream, tolerance schedule, structural replay rows, provenance rows, and local naming boundary through* S, T, H, C, P, N*. If the two cofinal ledgers* α *and* α' *choose the same source window for every inspected precision request, then their accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and terminal* $\mathsf{Real}^{\uparrow}$ *seal follow the same finite route. In particular, neither ledger can replace the displayed source-window or tolerance route before* R, R' *and* E, E' *are read.*

Proof. Project both accepted carriers through Definition 4222.1. The shared rows expose the same $\mathsf{StreamName}^{\uparrow}$ source windows, the same $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule, and the same replay and provenance boundary.

Apply Definition 4222.2 to both packets. The only ledger freedom available before the accelerated readback is the cofinal row, and by hypothesis the displayed source window selected for each inspected request is identical in α and α'.

Now use Definition 4222.4. Since the rectangular source-window and tolerance route is fixed, the regular rational rows R, R' and the Real seals E, E' are reached through the same finite route. The packet therefore has no row through which a second ledger can substitute a hidden subsequence, quotient representative, or ambient completeness argument. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyAcceleration}^{\uparrow}$ packet is generated from a StreamName source row, a DyadicRatCore tolerance schedule, a finite cofinal acceleration ledger, a RegSeqRat proof row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4222.1, Definition 4222.2, and Definition 4222.3 seed the finite acceleration route from StreamName windows through dyadic tolerance schedules, cofinal acceleration ledgers, RegSeqRat readback, and the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyAcceleration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy-name equality, choice-selected subsequences, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone cofinality, dyadic tolerance preservation, classifier transport, L10 handoff, and ledger exhaustion for BEDC.Derived.CauchyAccelerationUp.

Theorem 4222.6 (Cauchy acceleration sequence-space route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer reads the common-window classifier of* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *from chapter 406 before exporting the accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff. The route passes through the regularization boundary of chapter 376: the source* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, accelerated cofinal ledger, regular rational readback, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *rows are visible before any Real-facing consumer may inspect the acceleration result. No quotient representative, selected subsequence outside the displayed ledger, host equality, or ambient completeness row is exported.*

Theorem 4222.7 (Cauchy subsequence dependency route)**.** *Every* $\mathsf{CauchyAcceleration}^{\uparrow}$ *consumer that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal must first consume the rate-dominance surface of chapter 1814 and the fast-modulus surface of chapter 1781. The route is finite:* $\mathsf{CauchyRateDominance}^{\uparrow}$ *bounds the source schedule,* $\mathsf{FastCauchyModulus}^{\uparrow}$ *supplies the inspected acceleration window, and the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *transport, replay, provenance, and local* NameCert *rows carry the resulting readback. No consumer may replace this route by a chosen subsequence, quotient representative, ambient completion row, or host equality of Real names.*

Proof. Begin with the accepted carrier of Definition 4222.1. Its source stream row S, tolerance schedule T, cofinal ledger α, regular rational row R, and Real seal E are the only analytic rows available before transport, replay, provenance, and local naming.

The rate-dominance chapter supplies the comparison row that bounds which source schedule entries may be read by α. The fast-modulus chapter supplies the inspected window that turns those bounded reads into the accelerated tolerance route. Reading either row after the $\mathsf{RegSeqRat}^{\uparrow}$ readback would hide the schedule obligation rather than consume it.

Apply the rectangular window route of Definition 4222.4. It keeps the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows in front of R and E, so the accelerated readback remains a displayed finite route. The carrier has no coordinate for a selected subsequence, quotient representative, ambient completion proof, or host Real equality, and the dependency route cannot export one. □

Theorem 4222.8 (Cauchy acceleration L10 four-face cofinality)**.** *Every accepted* $\mathsf{CauchyAcceleration}^{\uparrow}$ *carrier that exports an accelerated* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *seal preserves the ordered L10 four-face route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The cofinal acceleration ledger may refine which displayed source window is read for a requested precision, but it cannot commute, omit, or replace the $\mathsf{StreamName}^{\uparrow}$ *source face, the* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ *readback face, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The resulting route remains finite and displayed, with no chosen subsequence outside the ledger, quotient representative, ambient completion row, host real equality, countable-choice schedule, Lean verification, or standard bridge.*

Proof. Start with Definition 4222.3. It names the only exported analytic route of an accepted carrier as the ordered sequence from the stream window through the dyadic tolerance ledger and regular rational readback to the Real-facing seal.

Next apply Definition 4222.4. The cofinal ledger α refines precision requests inside the displayed rectangle of source windows and dyadic tolerances, so it cannot move the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ rows ahead of the upstream faces.

Finally use the sibling dependency route of Definition 4222.7. The rate-dominance and fast-modulus surfaces are consumed before the accelerated readback is exported, and they leave the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ faces in order.

The carrier and dependency route have no row for a hidden subsequence, quotient representative, ambient completion proof, host Real equality, choice schedule, Lean verification, or bridge credential. Hence cofinal acceleration preserves exactly the displayed L10 four-face route. □

Theorem 4222.9 (Cauchy acceleration completion forward link)**.** *Every completion-facing consumer of* $\mathsf{CauchyAcceleration}^{\uparrow}$ *reads through chapter 1639, chapter 301, chapter 126, chapter 291, and chapter 38 before the accelerated route exports a terminal real boundary. The dependency is the finite route from source* $\mathsf{StreamName}^{\uparrow}$ *windows through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance schedule and cofinal acceleration ledger into* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the Bishop regular Cauchy completion surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Start with the displayed source windows. The accelerated route has no completion-facing input until the $\mathsf{StreamName}^{\uparrow}$ inspection windows and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance schedule have been exposed.

Next read the regular rational and completion surfaces. The cofinal acceleration ledger feeds $\mathsf{RegSeqRat}^{\uparrow}$ readback and then passes through the Bishop regular Cauchy completion chapter before any terminal real boundary is visible.

Finally attach the $\mathsf{Real}^{\uparrow}$ seal. Since the route is finite and displayed at each sibling boundary, it exports no quotient completion, selected subsequence, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Chapter 4223

A Concrete Naming Certificate for $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$

Following Brouwer and Bishop–Bridges on monotone sequences in constructive analysis, the BrouwerMonotoneSeque packet records a monotone real-name boundary object through finite stream windows, regular rational readback, dyadic tolerance ledgers, and the terminal real-name seal. It sits after the Specker obstruction of chapter 1406, the regular-rational readback of chapter 301, the stream-window interface of chapter 126, the dyadic rational core of chapter 291, and the Real seal of chapter 38. The chapter gives the L10 Real and Completion frontier a named monotone-sequence carrier without claiming located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4223.1 (BrouwerMonotoneSequence carrier)**.** *A* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, W, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source-sequence row, listing the inspected indices and their finite dependency order.*
2. *W is the finite family of* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the inspected real-name entries.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed windows and monotonicity comparisons.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows and dyadic tolerance ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, W, D, R have been displayed.*
6. *M is the monotonicity ledger, recording only finite adjacent-window comparison obligations among the displayed entries.*
7. *H records componentwise* hsame *transport for source indices, windows, dyadic ledgers, readback, Real seals, monotonicity, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a monotone-sequence request through $S, W, D, R, E, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *row.*

The classifier accepts only packets whose monotone-sequence request displays these rows in order. It has no coordinate for a located supremum, trichotomy decision, countable-choice selector, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.BrouwerMonotoneSequenceUp.BrouwerMonotoneSequenceTasteGate single carrier alignment`

Theorem 4223.2 (BrouwerMonotoneSequence NameCert obligations)**.** *For every accepted* BrouwerMonotoneSeque *carrier*

$$B = (S, W, D, R, E, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the inspected source sequence, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, finite monotonicity ledger, componentwise transport, displayed replay, package provenance, and local naming rows. No NameCert obligation may be discharged by a located supremum, trichotomy of real names, countable-choice schedule, selected limit, quotient stream representative, or ambient complete real line.*

Proof. Project the carrier of Definition 4223.1. The analytic rows are precisely the source schedule S, the finite windows W, the dyadic tolerance ledger D, the regular rational readback R, the Real seal E, and the finite monotonicity ledger M.

The monotonicity information is local to the displayed windows. Each comparison read in M consumes only the adjacent source entries already opened through S, W, D, R and does not add a global order decision for arbitrary real names.

The structural rows add no hidden analytic principle. Transport through H preserves the same displayed rows, replay through C follows the same finite route, and P, N record provenance and local naming for the packet.

Since the carrier has no row for located suprema, trichotomy, countable-choice schedules, selected limits, quotient stream representatives, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 4223.3 (BrouwerMonotoneSequence finite-window nonescape)**.** *Every consumer read of an accepted* $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ *carrier reaches a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may consult the finite monotonicity ledger M for displayed adjacent windows, but it cannot escape to a located supremum, countable-choice selector, trichotomy oracle, selected limit, quotient stream representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with a consumer read of the accepted carrier. The source row S exposes only the inspected indices, so the first real-name data available to the consumer are the finite $\mathsf{StreamName}^{\uparrow}$ windows in W.

The dyadic and regular-rational rows determine the only readback route. The tolerance ledger D bounds the displayed windows, and the row R packages those bounded observations as $\mathsf{RegSeqRat}^{\uparrow}$ readback data.

The terminal seal is downstream of those rows. The $\mathsf{Real}^{\uparrow}$ row E is available only after S, W, D, R have been displayed, and the monotonicity ledger M can compare only the adjacent windows already present in the same finite packet.

Transport, replay, provenance, and local naming preserve this order through H, C, P, N. Therefore every consumer read remains inside the displayed finite-window route and cannot import a

located supremum, trichotomy oracle, choice schedule, selected limit, quotient stream representative, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerMonotoneSequence}^{\uparrow}$ packet is generated from inspected source indices, finite StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a RealUp seal, a finite monotonicity ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4223.1, Definition 4223.2, and Definition 4223.3 seed the constructive monotone-sequence boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerMonotoneSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close located suprema, trichotomy, countable choice, selected limits, quotient stream representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window monotonicity, StreamName exposure, DyadicRatCore tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrouwerMonotoneSequenceUp.

Chapter 4224

A Concrete Naming Certificate for $\mathsf{CauchySequenceCriterion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy criteria, the $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet records the textbook sequence criterion as finite BEDC data. It ties a sequence source to the dyadic tolerance discipline of chapter 291, the stream-window naming discipline of chapter 126, the regular rational readback of chapter 301, and the public real seal of chapter 38. The chapter names only the finite route from tolerance data to regular-Cauchy and real consumers; it does not supply a choice-selected subsequence, quotient equality of limits, host completion, or ambient completeness principle.

Definition 4224.1 (CauchySequenceCriterion carrier)**.** *A* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, M, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the sequence-source row, displaying the named sequence and the finite request interface through which a downstream consumer asks for terms.*
2. *M is the modulus-schedule row, displaying the finite schedule that turns a requested dyadic tolerance into a stage bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger row, recording the positive dyadic thresholds, comparisons, and tolerance refinements read by the schedule.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the sequence after the schedule row has supplied their stage bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy readback row reached from A, M, D, S.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-Cauchy readback has been displayed.*
7. *H records componentwise* hsame *transport for the sequence source, schedule, dyadic tolerance ledger, stream windows, regular readback, Real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a criterion request through $A, M, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row.*

The classifier accepts only packets whose criterion consumer first displays the sequence source, modulus schedule, dyadic tolerance ledger, stream-window row, regular-Cauchy readback, Real seal, hsame*,* Cont*,* Pkg*, and local* NameCert *rows listed above.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCriterionUp.CauchySequenceCriterionTasteGate single carrier alignment`

Theorem 4224.2 (CauchySequenceCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier*

$$Q = (A, M, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *surface consists exactly of sequence-source admission, modulus-schedule admission, dyadic tolerance-ledger exposure, finite stream-window exposure, regular-Cauchy readback, Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of* Q*. No obligation may be discharged by a choice-selected subsequence, quotient equality of limits, host completion, ambient completeness, or an unbounded open-cover argument.*

Proof. Project the carrier of Definition 4224.1. Its analytic rows are precisely the sequence source A, modulus schedule M, dyadic tolerance ledger D, stream windows S, regular-Cauchy readback R, and Real seal E.

The tolerance rows determine the access order. A criterion consumer reads M before it can turn a dyadic request from D into a finite stage bound, and it reads S only through the finite windows exposed by that bound.

The readback rows add no external completion object. The row R repacks the displayed tolerance and window data as regular-Cauchy readback, and E seals that readback as the public real-facing output.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no coordinate of Q stores a selected subsequence, quotient limit equality, host completion, ambient completeness witness, or open-cover argument, the displayed rows exhaust the NameCert obligations. □

Theorem 4224.3 (CauchySequenceCriterion L10 handoff)**.** *Every L10 consumer read of an accepted* $\mathsf{CauchySequenceCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff reaches the displayed $\mathsf{Real}^{\uparrow}$ *seal only through the sequence source, modulus schedule, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$ *row carried by the same packet.*

Proof. Begin with the sequence source A and modulus schedule M. They expose the named sequence and the finite stage bounds required by any dyadic tolerance request.

Next read the dyadic tolerance ledger D and stream-window row S. The ledger supplies the positive thresholds and comparisons, while the stream row supplies only the finite windows named by the schedule.

Attach the regular-Cauchy readback row R. It is the unique displayed route by which the dyadic-window data becomes a regular rational sequence consumer.

The Real seal closes the route. The row E is reached only after R, and the structural rows H, C, P, N carry the same finite data through transport, replay, provenance, and local naming. Hence the L10 handoff exposes a constructive sequence criterion rather than a selected subsequence, quotient limit, host completion, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceCriterion}^{\uparrow}$ packet is generated from a sequence source, modulus schedule, dyadic tolerance ledger, stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4224.1, Definition 4224.2, and Definition 4224.3 seed the L10 route from dyadic sequence criteria to regular-Cauchy and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close choice-selected subsequences, quotient equality of limits, host completion, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus schedule exposure, dyadic tolerance-ledger exactness, stream-window handoff, RegSeqRat/Real seal readback, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCriterionUp.

$\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ routes pointwise locator induction through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38; the child file carries the theorem and closurestatus surface.

Chapter 4225

A Concrete Naming Certificate for CauchyRealLocatorPointwiseInduction$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet records the finite-window induction route used by Cauchy real locators. It is a sibling of the locator surface of chapter 3410 and the finite schedule discipline of chapter 126: a pointwise induction step opens one displayed stream window, reads its dyadic tolerance ledger, passes through the regular rational readback of chapter 301, and returns to the same Real-facing locator seal of chapter 38. The packet is not an induction principle over all host streams, a quotient-real equality theorem, a countable-choice selector, or an ambient completeness assertion.

The dependency route is substantive: before the packet exports pointwise locator induction, it consumes the rational interval enclosure surface of chapter 4202 together with the RegSeqRat window handoff of chapter 301. Thus each pointwise branch is read as rational interval enclosure plus regular Cauchy window data, not as a decorative citation to neighboring chapters.

Definition 4225.1 (Cauchy real locator pointwise induction carrier)**.** *A* CauchyRealLocatorPointwiseInduction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* CauchyRealLocator$^{\uparrow}$ *row, carrying the locator modulus, finite-window, dyadic, regular-rational, and Real seal data.*
2. *W is the* StreamName$^{\uparrow}$ *pointwise window row selected for the current locator request.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare the pointwise window with the requested side test.*
4. *Q is the* RegSeqRat$^{\uparrow}$ *readback row exposed after the window and dyadic tolerance have been displayed.*
5. *R is the terminal* Real$^{\uparrow}$ *locator seal reached by the pointwise readback.*
6. *B is the finite induction-branch ledger, recording the base window and successor-window handoff over the displayed rows W, D, Q, R.*
7. *H records componentwise* hsame *transport for the locator source, window, dyadic tolerance, regular readback, Real seal, branch ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay for each pointwise induction step.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyRealLocator}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4225.2 (Cauchy real locator pointwise induction regularity-witness route)**.** *Every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier consumes the finite regularity-witness surface of chapter 1158 before any* $\mathsf{Real}^{\uparrow}$ *seal is inspected. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the pointwise branch first opens the displayed $\mathsf{StreamName}^{\uparrow}$ *window W, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q through the regularity-witness route, and only then reaches the terminal Real locator seal R.*

Proof. Project the carrier. The only rows available before the Real seal are the locator source L, the displayed window W, the dyadic tolerance D, and the regular-rational readback Q.

Now use the sibling regularity-witness surface. Its displayed route supplies the same finite window, tolerance ledger, and RegSeqRat readback before the Real seal handoff, so the pointwise branch can inspect regularity evidence without opening a host stream representative.

Finally read R and the structural rows B, H, C, P, N. These rows record, transport, replay, source, and name the already displayed route, so the pointwise induction packet depends on the regularity witness before it reaches the Real-facing locator seal. □

Theorem 4225.3 (Cauchy real locator pointwise window induction)**.** *For every accepted* CauchyRealLocatorPointwi *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

each pointwise locator step factors through the finite route

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B, H, C, P, N.$$

The base and successor cases of the induction ledger B may inspect only the displayed $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *locator seal supplied by the same* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row.*

Proof. Project the accepted carrier. The source row L is the only Cauchy real locator coordinate, so every pointwise step begins by reading the displayed locator packet of chapter 3410.

The induction window is then opened through W. The dyadic tolerance row D is the only comparison surface attached to that window, and the regular rational readback Q is available only after the window and dyadic rows have been displayed.

The terminal read reaches R and is then recorded by the branch ledger B. The base branch and each successor branch therefore preserve the same finite route rather than selecting a host stream representative.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no row for host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness, the displayed route exhausts the pointwise induction surface. □

Theorem 4225.4 (Cauchy real locator pointwise induction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$ *obligations are exactly carrier admission for the* $\mathsf{CauchyRealLocator}^{\uparrow}$ *source row* L*, pointwise* $\mathsf{StreamName}^{\uparrow}$ *window admission for* W*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility for* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback for* Q*,* $\mathsf{Real}^{\uparrow}$ *locator-seal ordering for* R*, finite base and successor branch admission for* B*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 4225.1. The tuple displays the locator source, pointwise window, dyadic tolerance, regular-rational readback, real seal, branch ledger, and structural rows.

The source and window obligations are read from L and W. The dyadic tolerance row D is the only comparison surface attached to that pointwise window, and the readback row Q can be inspected only after those rows are displayed.

The real-seal ordering obligation is the row R, downstream of Q. The branch ledger B records the finite base and successor handoff over the same displayed rows rather than over a host stream representative.

The rows H, C, P, N transport, replay, source, and name the packet. Since no other coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4225.5 (Cauchy real locator pointwise window stability)**.** *Let*

$$I = (L, W, D, Q, R, B, H, C, P, N)$$

be an accepted $\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}$ *carrier. Replacing the locator source,* $\mathsf{StreamName}^{\uparrow}$ *pointwise window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and* $\mathsf{Real}^{\uparrow}$ *seal by displayed componentwise* hsame *rows, and replaying the replacement through* C*, preserves the admissible pointwise induction route*

$$L \longmapsto W, D \longmapsto Q \longmapsto R \longmapsto B.$$

Thus the base and successor branches of the finite induction ledger remain attached to the same locator source and cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier and use Definition 4225.3. Every pointwise step already factors through the displayed locator source, window, dyadic tolerance, regular-rational readback, real seal, and branch ledger.

The componentwise replacement is recorded by H. It transports the locator source, window, tolerance, readback, seal, branch ledger, replay, provenance, and local name without adding another induction coordinate.

The replay row C then carries the transported rows through the same pointwise route. The branch ledger B therefore continues to read the base window and successor-window handoff over W, D, Q, R.

The carrier has no row for any excluded host or quotient object, so row exhaustion shows that the transported packet preserves exactly the admissible pointwise induction route. □

Theorem 4225.6 (Cauchy real locator pointwise branch exhaustion)**.** *For every accepted* CauchyRealLocatorPointw *carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

every base or successor branch read in B is exhausted by the displayed rows $L, W, D, Q, R, H, C, P, N$. Such a branch read cannot introduce host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The branch ledger B is downstream of the locator source L, pointwise window W, dyadic tolerance row D, regular-rational readback Q, and terminal real seal R.

Use pointwise window induction from Definition 4225.3. It forces both the base branch and each successor branch to read the same finite route $L \longmapsto W, D \longmapsto Q \longmapsto R$ before B records the branch.

Use window stability from Definition 4225.5. The transport row H and replay row C preserve that route under displayed componentwise replacement, while P, N only source and name the packet.

Row exhaustion completes the claim. No coordinate of I stores host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so no base or successor branch can export any of those objects. □

Theorem 4225.7 (Cauchy real locator pointwise induction sibling route)**.** *Every accepted* CauchyRealLocatorPointw *read must pass through chapter 3410, chapter 126, chapter 291, chapter 301, and chapter 38 before the branch ledger is public. For a carrier*

$$I = (L, W, D, Q, R, B, H, C, P, N),$$

the rows L, W, D, Q, R are exactly that sibling route, and B, H, C, P, N can only record, transport, replay, source, and name the displayed route.

Proof. Project the carrier rows $L, W, D, Q, R, B, H, C, P, N$ from Definition 4225.1. The source row L is read through the Cauchy real locator certificate, while W is the finite StreamName window attached to the current pointwise request.

The dyadic tolerance row D is then consumed through the DyadicRatCore certificate, and Q is the RegSeqRat readback of that displayed window and tolerance data. The terminal row R is the Real-facing locator seal reached after those rows are visible.

Apply Definition 4225.3 and Definition 4225.4. They force the pointwise induction route and the local NameCert obligations to use exactly the rows L, W, D, Q, R before the branch ledger can be inspected.

Finally use Definition 4225.5 and Definition 4225.6. The transport, replay, provenance, and local naming rows preserve the same sibling route and exhaust the branch surface, so the public branch ledger has no alternate dependency edge outside the listed chapters. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4225.1, Definition 4225.3, Definition 4225.4, Definition 4225.5, and Definition 4225.6 seed the finite pointwise induction route over CauchyRealLocator, StreamName, DyadicRatCore, RegSeqRat, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRealLocatorPointwiseInduction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host stream induction, quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window induction, dyadic tolerance compatibility, RegSeqRat readback, Real seal ordering, and local NameCert registration for BEDC.Derived.CauchyRealLocatorPointwiseInductionUp.

Constructive story: A CauchyRealLocatorPointwiseInduction$^{\uparrow}$ packet is generated from a CauchyRealLocator source row, one StreamName pointwise window, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RealUp locator seal, a finite induction-branch ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4226

A Concrete Naming Certificate for $\mathsf{FrinkMetrization}^{\uparrow}$

Following Frink and Kelley, the $\mathsf{FrinkMetrization}^{\uparrow}$ packet records the finite BEDC route from a uniform entourage basis to a pseudometric-facing and metric-facing distance certificate. It sits after the uniform-source rows of chapter 339, the pseudometric carrier of chapter 1614, the separated metric boundary of chapter 57, the topology compatibility row of chapter 88, the dyadic tolerance ledger of chapter 291, and the finite window interface of chapter 126. The packet names only the constructive handoff from nested entourage data to displayed distance reads; it does not form quotient uniformities, select representatives, assert ambient metrizability, or import choice.

Definition 4226.1 (FrinkMetrization carrier)**.** *A* $\mathsf{FrinkMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (U, Q, D, P, M, T, H, C, G, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row. It names the carrier surface and the finite entourage requests from which the metrization route starts.*
2. *Q is the nested entourage chain. It records the finite refinement ladder used to compare two source points at dyadic scales.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger. It translates the refinement level into displayed dyadic radii and triangle budgets.*
4. *P is the* $\mathsf{Pseudometric}^{\uparrow}$ *handoff row. It receives the finite entourage chain and exports the displayed pre-separated distance route.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *compatibility row, available only after the separated boundary has accepted the zero-distance route supplied through P.*
6. *T is the* $\mathsf{Topology}^{\uparrow}$ *compatibility row. It records that the metric-facing balls and the original uniform entourage reads are compared through the same finite windows.*
7. *H records componentwise* hsame *transport for the uniform source, chain, dyadic ledger, pseudometric handoff, metric compatibility, topology compatibility, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a uniform entourage request through the nested chain, dyadic ledger, pseudometric handoff, metric row, topology row, provenance, and local naming row.*

9. *G is the* Pkg *provenance over* FrinkMetrization$^{\uparrow}$, UniformSpace$^{\uparrow}$, Pseudometric$^{\uparrow}$, Met$^{\uparrow}$, Topology$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, or an unbounded distance oracle.

Lean4 ✓ `BEDC.Derived.FrinkMetrizationUp.FrinkMetrizationTasteGate single carrier alignment`

Theorem 4226.2 (FrinkMetrization NameCert obligations)**.** *For every accepted* FrinkMetrization$^{\uparrow}$ *carrier $F = (U, Q, D, P, M, T, H, C, G, N)$, the local* NameCert$_{\text{FrinkMetrization}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the uniform source, nested entourage chain, dyadic radius ledger, pseudometric handoff, metric compatibility row, topology compatibility row, transport, continuation, provenance, and local naming row;*
2. *nested-chain readability, requiring every distance request to start at the* UniformSpace$^{\uparrow}$ *row U and pass through the finite refinement chain Q before any dyadic radius in D is read;*
3. *pseudometric handoff, requiring the displayed distance route to be exported through P before the metric row M or topology row T can be consumed;*
4. *compatibility transport, requiring H and C to preserve the same finite* StreamName$^{\uparrow}$ *windows and* DyadicRatCore$^{\uparrow}$ *radius ledger across uniform, pseudometric, metric, and topology-facing consumers;*
5. *provenance and refusal, requiring G and N to name only this finite handoff and to refuse quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, and unbounded distance oracles.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 4226.1. The projection exposes exactly the uniform source, nested chain, dyadic ledger, pseudometric handoff, metric and topology compatibility rows, and the structural rows H, C, G, N.

The nested-chain obligation is forced by row order. A distance request starts at the uniform row U, enters the refinement ladder Q, and only then reads the dyadic radius ledger D.

The pseudometric handoff is the only distance-export step. Row P supplies the pre-separated displayed distance route, while M and T can compare it with metric and topology-facing consumers only after the same finite windows and radii have been replayed.

Rows H, C, G, N transport, replay, source, and name this route. Since the tuple contains no quotient uniformity, selected representative, ambient metrizability, countable choice, host topology equality, or unbounded distance coordinate, the local NameCert cannot export those principles. □

Closure status for NameCert$_{\text{FrinkMetrization}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, seed)

Scope closed: Definition 4226.1 and Definition 4226.2 seed the finite Frink-style handoff from uniform entourage chains to pseudometric, metric, topology, dyadic, and stream-facing rows.

Formal verification: FormalStatus(NameCert$_{\text{FrinkMetrization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient uniformity equality, selected representatives, ambient metrizability, countable choice, host topology equality, unbounded distance oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-chain readability, dyadic radius exactness, pseudometric handoff, metric and topology compatibility, transport, continuation, provenance, and ledger refusal for BEDC.Derived.FrinkMetrizationUp.

Constructive story: A FrinkMetrization$^{\uparrow}$ packet is generated from a UniformSpace source row, a nested entourage chain, a DyadicRatCore radius ledger, a Pseudometric handoff, Metric and Topology compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

NestedShrinkInterval$^{\uparrow}$ reads its compact interval dependency through chapter 5553 and the child `namecert_construction.tex` route before exposing its shrinking-diameter Real$^{\uparrow}$ seal.

Chapter 4227

A Concrete Naming Certificate for $\mathsf{NestedShrinkInterval}^{\uparrow}$

Following Cantor's nested-interval argument and Bishop–Bridges regular real analysis, the $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet records the finite BEDC carrier surface for a descending located interval route whose displayed diameters shrink to the requested tolerance. It sits between the nested-closed interval carrier of **??**, the located endpoint row of chapter 394, the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name boundary of chapter 38. The chapter names only the finite shrinking-interval convergence certificate; it does not claim open-cover compactness, trichotomy, quotient reals, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4227.1 (Nested shrink interval carrier)**.** *A* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (J, L, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row, exposing the retained descending closed located interval family and its containment witnesses.*
2. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row for the inspected interval windows, including lower and upper endpoint location data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger recording endpoint comparisons, diameter estimates, and tolerance requests.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule used to inspect interval endpoints and diameter rows at displayed stages.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the same stream windows and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the nested interval, located endpoint, dyadic shrink, stream-window, and regular readback rows are visible.*
7. *H records componentwise* hsame *transport for J, L, D, S, R, E and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$J, L \longmapsto D, S \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{NestedShrinkInterval}^{\uparrow}$, $\mathsf{NestedClosedInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by the displayed rows $J, L, D, S, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, trichotomy, quotient-real equality, a selected intersection point, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4227.2 (Nested shrink interval NameCert obligations)**.** *For every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier*

$$K = (J, L, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source row,* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *shrink ledger,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row; classifier stability under those same displayed rows; diameter-to-tolerance exactness through D; finite-window readback through S and R; Real-seal handoff through E; and ledger nonescape for the excluded compactness, trichotomy, quotient, selection, choice, and ambient-completeness coordinates.*

Proof. Project the carrier of Definition 4227.1. The projection exposes exactly the nested closed interval source, located endpoint row, dyadic shrink ledger, finite windows, regular readback, real seal, and structural rows.

Classifier stability is componentwise on those displayed rows. The shrink estimate is not an external metric fact: it is the D row, read together with the endpoint row L and the stream schedule S before regular readback is available.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded coordinate occurs in the carrier, every accepted NameCert obligation is one of the rows listed above. □

Theorem 4227.3 (Nested shrink interval diameter route)**.** *Let $K = (J, L, D, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. Every completion-facing request for the shrinking interval route first reads the* $\mathsf{NestedClosedInterval}^{\uparrow}$ *source J and* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row L, then reads the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger D and* $\mathsf{StreamName}^{\uparrow}$ *finite windows S, and only then opens the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R and* $\mathsf{Real}^{\uparrow}$ *seal E. The route exports diameter-to-tolerance convergence as displayed finite ledger data, not as open-cover compactness, trichotomy, quotient-real equality, countable choice, a selected intersection point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing request and project the accepted carrier. The only interval source is J, and the only endpoint data are the located rows in L.

The diameter request is then forced through D. Its dyadic comparisons and tolerance rows are inspected through the finite schedule S, so a consumer cannot replace the shrinking route by an unrecorded compactness or completeness theorem.

After the finite windows are displayed, R packages the same observations as regular rational readback and E supplies the terminal real seal. The remaining rows H, C, P, N preserve the route and add no excluded coordinate. □

Theorem 4227.4 (Nested shrink interval Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer admitted by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet receives the seal only through the displayed route*

$$\mathsf{NestedClosedInterval}^{\uparrow},\ \mathsf{LocatedInterval}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff is stable under the componentwise hsame *transport and displayed* Cont *replay rows* H, C*, and it is sourced and named only by* P, N*.*

Proof. Apply Definition 4227.3. A Real-facing consumer reaches E only after the nested interval source, located endpoint row, dyadic shrink ledger, stream windows, and regular readback have all been displayed.

Transport through H preserves the row roles, and replay through C preserves their order. Provenance and local naming through P, N attach the handoff to the packet without adding analytic content.

Thus the Real seal cannot escape to a selected point, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient real completeness. □

Theorem 4227.5 (Nested shrink interval regular-Cauchy uniqueness route)**.** *Let* $K = (J, L, D, S, R, E, H, C, P, N)$ *and* $K' = (J', L', D', S', R', E', H', C', P', N')$ *be accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packets. Suppose their nested-source rows, located endpoint rows, dyadic shrink ledgers, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows are componentwise related by the displayed* hsame *transport. Then the two packets reach the same* $\mathsf{Real}^{\uparrow}$*-facing seal route only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *uniqueness surface displayed by* R, R' *before Real sealing. The route does not identify selected intersection points, quotient real representatives, or ambient host real equality.*

Proof. Project both carriers. The first carrier exposes $J, L, D, S, R, E, H, C, P, N$, and the second exposes $J', L', D', S', R', E', H', C', P', N'$.

Apply the diameter route of Definition 4227.3 to both packets. Each read is forced through its nested source, located endpoints, dyadic shrink ledger, finite stream windows, and regular rational readback before a Real-facing seal is visible.

Use the assumed componentwise hsame transport on J, L, D, S, R and J', L', D', S', R'. The endpoint comparison therefore lives on the displayed $\mathsf{RegSeqRat}^{\uparrow}$ readback surface, because the preceding rows have already fixed the same nested-source, located-endpoint, dyadic, and stream-window route.

Finally apply the Real-seal handoff of Definition 4227.4 to both packets. It permits E and E' to be read only after the regular readback rows are displayed, so the uniqueness comparison is a $\mathsf{RegSeqRat}^{\uparrow}$ route before Real sealing and not a selected intersection point, quotient real equality, or ambient completeness assertion. □

Theorem 4227.6 (Cantor intersection nested-shrink route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer that reads a nested shrinking interval first consumes the* $\mathsf{NestedClosedInterval}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows exposed by an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *carrier. The downstream route supplies the nested-shrink boundary used by chapter 6999 and the Cantor-intersection surface of chapter 1421.*

The route is a forward lattice edge only. It does not select an intersection point, assert open-cover compactness, use countable choice, identify quotient real names, or provide ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4227.1. The exposed rows are the nested closed source J, located endpoints L, dyadic shrink ledger D, finite stream windows S, regular readback R, real seal E, and the structural rows H, C, P, N.

Apply Definition 4227.4. Any Real-facing downstream read reaches E only after the nested, located, dyadic, stream, and regular-rational rows have been displayed.

The Cantor diagonal and Cantor-intersection chapters consume exactly this kind of finite shrinking-interval boundary before their own diagonal or intersection rows are read. Since the

carrier contains no selected point, open-cover compactness row, choice schedule, quotient equality, or ambient completeness coordinate, the downstream route remains a finite dependency edge. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet is generated from a NestedClosedInterval source row, LocatedInterval endpoint rows, a DyadicRatCore shrink ledger, StreamName finite windows, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4227.1, Definition 4227.2, Definition 4227.3, and Definition 4227.4 seed the finite L10 shrinking-interval convergence route through nested closed intervals, located endpoint rows, dyadic diameter ledgers, stream windows, regular readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedShrinkInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient reals, selected intersection points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, diameter-to-tolerance exactness, finite-window readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.NestedShrinkIntervalUp.

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ packet consumes the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 before a completion-facing reader treats shrinking finite windows as a diagonal Cauchy subsequence witness. The nested source, located endpoint, dyadic diameter, and $\mathsf{StreamName}^{\uparrow}$ rows still supply the finite shrinking windows; $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies only the displayed diagonal handoff needed to pass those windows toward $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The route exports no detached selector, quotient-real equality, countable-choice schedule, selected intersection point, or ambient completeness coordinate. The shrinking interval diameter route is consumed through the regular rational readback of chapter 301 and the terminal real-name seal of chapter 38. Thus a downstream L10 consumer reads $\mathsf{NestedShrinkInterval}^{\uparrow}$ only after the finite diameter ledger has been repacked as a $\mathsf{RegSeqRat}^{\uparrow}$ route and sealed as a $\mathsf{Real}^{\uparrow}$ boundary, not as an ambient completeness or selected intersection-point claim. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ after the shrinking-diameter ledger has been repacked through the $\mathsf{StreamName}^{\uparrow}$ observation schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, and the terminal $\mathsf{Real}^{\uparrow}$ seal route. This sibling dependency records the completion-criterion consumer path of chapter 1476: a nested shrink interval can be read as completion evidence only after its finite diameter data has passed through the same stream, regular sequence, dyadic tolerance, and Real seal rows.

Theorem 4227.7 (Nested shrink interval Cauchy completion consumer)**.** *Every completion-facing read of* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *consumes* $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ *only after the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are visible. Thus the dependency on chapter 1476 is a theorem-level consumer edge through chapter 126 and chapter 301, not an appeal to an undisplayed completion object.*

Proof. Read a completion-facing request for a nested shrink interval. The route first displays the finite observation schedule and regular rational readback rows, so the consumer sees only finite windows and readback entries.

The dyadic tolerance ledger and terminal Real seal are then exposed before the completion criterion is consumed. Since the criterion is reached after those rows, the dependency records a reusable finite route rather than a hidden selected limit or ambient completeness principle. □

Theorem 4227.8 (Nested shrink interval rational locator route)**.** *Every accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *route whose endpoint localization is read through rational endpoints first consumes the rational locator carrier of chapter 4202. Only after that locator row is displayed may the consumer read the*

nested shrink rows and then the $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *handoff rows. The route exports no selected completed point and no Heine–Borel compactness principle.*

Proof. Read the accepted packet through the carrier theorem of chapter 4227. The only endpoint localization rows available to the nested shrink route are finite displayed rows, so a rational-endpoint consumer must obtain its rational interval locator row before the shrink ledger can be interpreted.

The sibling chapter chapter 4202 supplies exactly that locator surface. Once it is consumed, the nested shrink packet may proceed through its displayed shrink ledger, finite windows, regular rational readback, and terminal real seal.

No selected completed point or Heine–Borel compactness row is present in either carrier. The dependency route therefore records only the rational locator read before the nested shrink and $\mathsf{RegSeqRat}^{\uparrow}/\mathsf{Real}^{\uparrow}$ handoff rows are read. □

The $\mathsf{NestedShrinkInterval}^{\uparrow}$ route consumes chapter 4202 for rational endpoint bracketing and chapter 301 for regular rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is visible. Thus the nested shrink interval dependency lattice reads locator cells first, then the shrinking-window ledger, then the regular readback route, not a selected intersection point or ambient completeness witness. $\mathsf{NestedShrinkInterval}^{\uparrow}$ exposes a completion-facing interval read only after the packet has consumed the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ diagonal row, the $\mathsf{RegSeqRat}^{\uparrow}$ regular rational route, the terminal $\mathsf{Real}^{\uparrow}$ seal, the $\mathsf{CauchyCompletionCriterion}^{\uparrow}$ admission row, and the $\mathsf{RationalIntervalLocator}^{\uparrow}$ endpoint-bracketing row. The public read is therefore a finite sibling route through diagonal selection, regular sequence readback, Real sealing, completion admission, and rational interval location; it exports no selected intersection point, quotient-real equality, countable choice, or ambient completeness principle. $\mathsf{NestedShrinkInterval}^{\uparrow}$ consumes the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 and the completion criterion surface of chapter 1476 before any interval-limit seal is visible. The shrinking-diameter ledger must first pass through finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, regular rational readback, and the same Cauchy-completion admission route; only then may a downstream L10 consumer inspect the $\mathsf{Real}^{\uparrow}$ boundary, and the packet still exports no selected intersection point, quotient-real equality, countable choice, or ambient completion. The $\mathsf{NestedShrinkInterval}^{\uparrow}$ carrier depends on the substantive sibling lattice through chapter 320 for the nested interval spine, chapter 301 for regular rational readback, and chapter 38 for the terminal seal. The rational locator and Cauchy-completion routes already represented by the child files above supply the endpoint-bracketing and completion-admission rows, so the nested-shrink packet is read as a finite dependency route rather than a selected intersection point, quotient-real equality, countable choice, or ambient completeness principle.

Theorem 4227.9 (Nested shrink interval compact interval route)**.** *Every compact-interval consumer of an accepted* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet must read the compact real interval sibling of chapter 5553 and the nested interval convergence sibling of chapter 2057 through the same finite endpoint, diameter,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows displayed by the shrinking packet. The compact interval route supplies endpoint and finite-net context; the nested convergence route supplies the shrinking-window handoff; neither route adds an open-cover compactness theorem, selected limit point, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Project the accepted carrier of Definition 4227.1. The visible analytic rows are the nested source, located endpoints, dyadic diameter ledger, finite stream windows, regular rational readback, and Real seal, followed by componentwise transport, replay, provenance, and local naming.

Use the compact real interval packet of chapter 5553 only for the endpoint and finite-net context. Its compact rows are read before the shrinking packet is exported as an interval-local dependency, so the shrink route never obtains an unlisted open-cover or subcover selector.

Use the nested interval convergence packet of chapter 2057 for the convergence-facing handoff. That sibling reads finite endpoint windows, shrinking ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ sealing before a completion consumer may inspect the route.

The structural rows hsame, Cont, Pkg, and NameCert keep the two sibling reads attached to the same displayed packet. Hence a compact-interval dependency edge from $\mathsf{NestedShrinkInterval}^{\uparrow}$ is exactly the stated finite route and not a hidden compactness, quotient, choice, or completeness principle. □

Theorem 4227.10 (NestedShrinkInterval Cantor downstream route)**.** *Every* $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ *or* $\mathsf{CantorIntersection}^{\uparrow}$ *consumer of a* $\mathsf{NestedShrinkInterval}^{\uparrow}$ *packet reads the nested source only through nested closed intervals, located endpoint rows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter witnesses,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The downstream route exports a Cantor diagonal or intersection-facing interval certificate, but it does not select a limit point, assert ambient completeness, identify quotient-real names, or add a host compactness principle outside the displayed rows.*

Proof. Project the nested-shrink carrier through the child routes named by the hub. The compact-interval and rational-locator rows provide nested closed intervals and located endpoints, while the dyadic and StreamName rows expose only finite diameter and window observations.

The Cantor-facing consumer then passes through the RegSeqRat and Real rows already used by the completion and sibling routes. Since these rows carry finite readback and terminal naming rather than a chosen point, the consumer obtains only the displayed interval certificate.

No carrier coordinate names a selected limit point, host completeness theorem, quotient equality, or ambient compactness object. Thus $\mathsf{CantorDiagonalCompactness}^{\uparrow}$ and $\mathsf{CantorIntersection}^{\uparrow}$ consume $\mathsf{NestedShrinkInterval}^{\uparrow}$ only through the finite interval route stated above. □

Chapter 4228

A Concrete Naming Certificate for $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy names and finite modulus extraction, the $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ packet records the threshold-selection surface that is actually inspected by a regular real-name consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite stream-window interface of chapter 126, the regular-rational readback of chapter 301, the ordinary modulus row of chapter 292, and the terminal real-name seal of chapter 38. The chapter names only the finite adaptive route from requested tolerances to inspected windows, regular readback, and Real consumers; it does not choose a global limit, quotient stream representative, ambient complete real, or unbounded modulus oracle.

Definition 4228.1 (Adaptive Cauchy modulus carrier)**.** *An* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the finite name windows available to a regular Cauchy request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested positive dyadic thresholds and the refinements used by the threshold search.*
3. *T is the adaptive threshold schedule, assigning to each displayed tolerance request the finite stage bound selected from the inspected evidence.*
4. *W is the inspected-window row, listing exactly the stream windows opened while validating the schedule T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the source stream, tolerance ledger, threshold schedule, and inspected windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular-rational readback has been displayed.*
7. *H records componentwise* hsame *transport for $S, D, T, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from an adaptive-modulus request through the same source, tolerance, schedule, window, readback, seal, transport, provenance, and naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets whose threshold read is mediated by $S, D, T, W, R, E, H, C, P, N$. There is no coordinate for a selected limit, quotient stream equality, host real equality, ambient completeness, countable choice, or an uninspected global modulus.

Lean4 ✓ `BEDC.Derived.AdaptiveCauchyModulusUp.AdaptiveCauchyModulusTasteGate single carrier alignment`

Theorem 4228.2 (Adaptive Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier*

$$A = (S, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$ *surface consists exactly of source-stream admission, dyadic tolerance-ledger exposure, adaptive threshold-schedule admission, inspected-window replay, regular-rational readback, terminal Real seal readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through the rows of A.*

No obligation in the packet is discharged by a selected Cauchy limit, quotient equality of streams, host real equality, ambient completeness, countable choice, or an oracle for a modulus outside the displayed finite windows.

Proof. Project the carrier of Definition 4228.1. The analytic rows are precisely the source stream S, tolerance ledger D, adaptive threshold schedule T, inspected windows W, regular readback R, and terminal Real seal E.

The access order is forced by the rows. A consumer first reads a dyadic request from D, then uses T to select a finite stage bound, and then opens only the windows recorded by W in the source stream S.

The readback rows add no external completion principle. The row R repacks the inspected stream and tolerance data as a regular rational sequence, and E seals that readback for Real-facing consumers only after R is present.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no coordinate of A stores a selected limit, quotient equality, host equality, ambient completeness, countable choice, or an uninspected global modulus, the displayed rows exhaust the local NameCert obligations. □

Theorem 4228.3 (Adaptive Cauchy modulus RegSeqRat handoff)**.** *Every regular Cauchy or Real consumer of an accepted* $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports only the source stream, tolerance ledger, adaptive threshold schedule, inspected windows, regular-rational readback, terminal Real seal, componentwise transport, displayed replay, provenance, and local name carried by the same packet.

Proof. Start with the NameCert obligations of Definition 4228.2. They force every accepted consumer to read the source stream, dyadic tolerance ledger, threshold schedule, and inspected windows before any regular-rational output is available.

Next attach the ordinary modulus interface. The schedule row T is the finite adaptive presentation of the requested $\mathsf{CauchyModulus}^{\uparrow}$ behavior, and the inspected windows W record the evidence used to justify each displayed threshold.

The regular and Real rows then close the handoff. The row R is the only displayed path from the inspected data to $\mathsf{RegSeqRat}^{\uparrow}$, and the seal E is the only displayed path from R to $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N preserve that same route through transport, replay, provenance, and local naming. Hence a consumer reaches RegSeqRat and Real data through finite inspected evidence rather than through quotient streams, selected limits, host equality, ambient completeness, countable choice, or a detached modulus oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}$.

Constructive story: An $\mathsf{AdaptiveCauchyModulus}^{\uparrow}$ packet is generated from a source stream, dyadic tolerance ledger, adaptive threshold schedule, inspected stream windows, RegSeqRat readback, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4228.1, Definition 4228.2, and Definition 4228.3 seed the finite adaptive threshold-selection route for regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AdaptiveCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, host real equality, ambient completeness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adaptive threshold schedule exposure, inspected-window replay, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.AdaptiveCauchyMod

Chapter 4229

A Concrete Naming Certificate for $\mathsf{ClosedSet}^{\uparrow}$ Metric Boundary

Following Kelley and Munkres on closed subsets as complements of open neighborhoods and as metric limit boundaries, the $\mathsf{ClosedSet}^{\uparrow}$ packet records the finite BEDC naming surface for closed-set reads. It sits between the topological carrier of chapter 88, the metric and separated-metric routes of chapter 57 and chapter 1411, the terminal real-name seal of chapter 38, and the compact-image consumer of chapter 4162. The packet names a displayed closed-boundary ledger; it does not import a host powerset, a detached open-set lattice, selected limit points, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4229.1 (ClosedSet carrier)**.** *A* $\mathsf{ClosedSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, M, S, F, O, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *carrier row containing the finite neighborhood and complement observations used by the closed-set read.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row used when the closed boundary is inspected by distance or ball data.*
3. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *row that turns metric zero-distance comparisons into separated boundary evidence.*
4. *F is the finite membership classifier for the named subset boundary.*
5. *O is the open-neighborhood complement ledger, recording the displayed route from nonmembership to a neighborhood outside the set.*
6. *B is the metric complement boundary row, recording the finite ball and real-distance readbacks through* $\mathsf{Real}^{\uparrow}$ *that witness the same complement behavior.*
7. *H records componentwise* hsame *transport for $T, M, S, F, O, B, C, P, N$.*
8. *C records displayed* Cont *replay from topology reads through metric and separated-metric observations, open-neighborhood complement rows, metric boundary rows, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{ClosedSet}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{SeparatedMetric}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^\uparrow}$ *row.*

The classifier compares accepted packets only through $T, M, S, F, O, B, H, C, P, N$. It has no coordinate for host powersets, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient Real completeness, or compact-image closedness beyond the displayed consumer route.

Lean4 (definition) `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetUp`

Theorem 4229.2 (ClosedSet NameCert obligations)**.** *For every accepted* $\mathsf{ClosedSet}^\uparrow$ *carrier*

$$K = (T, M, S, F, O, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClosedSet}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the topological row, metric row, separated-metric row, membership classifier, open-neighborhood complement ledger, metric complement boundary, transport, replay, provenance, and local naming;*
2. *topology compatibility, requiring every closed-set read to expose the* $\mathsf{Topology}^\uparrow$ *neighborhood and complement rows before the membership classifier is consumed;*
3. *metric compatibility, requiring any ball or distance read to pass through* $\mathsf{Met}^\uparrow$, $\mathsf{SeparatedMetric}^\uparrow$, *and the* $\mathsf{Real}^\uparrow$ *distance seal before it is treated as boundary evidence;*
4. *complement exactness, requiring nonmembership evidence to be routed through O and metric boundary evidence to be routed through B over the same classifier F;*
5. *transport replay, requiring H and C to preserve the ordered route componentwise before P and N are read;*
6. *nonescape, excluding host powersets, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient* $\mathsf{Real}^\uparrow$ *completeness, and a proof of compact-image closedness not routed through chapter 4162.*

No local ClosedSet NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4229.1. The tuple displays the topology, metric, separated-metric, membership, complement, metric-boundary, transport, replay, provenance, and local naming rows.

Topology compatibility is forced by the placement of T and F. A closed-set read first opens the finite neighborhood and complement observations, and only then consumes the classifier for the named subset boundary.

Metric compatibility is forced by M, S, B. Any distance or ball evidence passes through the metric row, the separated zero-distance row, and the $\mathsf{Real}^\uparrow$ distance seal before it becomes boundary evidence.

The remaining rows preserve and name the same route. Since the carrier has no coordinate for host powersets, detached open-set lattices, selected limit points, quotient equality, choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetTasteGate single carrier alignment`

Theorem 4229.3 (ClosedSet metric complement boundary)**.** *Every accepted* $\mathsf{ClosedSet}^\uparrow$ *carrier routes a metric complement request through*

$$\mathsf{Topology}^\uparrow \longmapsto \mathsf{Met}^\uparrow, \mathsf{SeparatedMetric}^\uparrow, \mathsf{Real}^\uparrow \longmapsto F, O, B, H, C, P, N.$$

Thus a consumer such as $\mathsf{CompactImageClosed}^{\uparrow}$ *may read a named closed boundary only after the topological complement ledger, metric distance rows, separated classifier, real-distance seal, transport, replay, provenance, and local naming rows are displayed. A request bypassing one of those rows is outside the accepted* $\mathsf{ClosedSet}^{\uparrow}$ *packet.*

Proof. Read the accepted carrier $K = (T, M, S, F, O, B, H, C, P, N)$. The topological boundary is T, and the metric boundary is read only through M, S and the real-distance data stored in B.

The classifier F is shared by the open-neighborhood complement ledger O and the metric complement boundary B. Therefore the two complement presentations cannot diverge into a host powerset or detached open-set lattice.

Transport and replay are then supplied by H and C, while P and N record provenance and local naming. Since the compact-image consumer of chapter 4162 receives only that finite boundary row, it cannot obtain selected limit points, quotient equality, countable choice, or ambient Real completeness from the closed-set packet. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetCarrier metric complement boundary`

Theorem 4229.4 (ClosedSet scope basis)**.** *The* $\mathsf{ClosedSet}^{\uparrow}$ *scope is fixed by the carrier and metric-complement boundary rows over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, hsame, Cont, *and the local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *surface. A consumer may read closed-boundary data only after the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise transport, displayed replay, provenance, and local naming rows have been exposed. The scope refuses host powerset semantics, detached open-set lattices, quotient equality of names, selected limit points, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Use the carrier of Definition 4229.1. Its displayed rows contain exactly the topology, metric, separated-metric, classifier, complement, metric-boundary, transport, replay, provenance, and local naming data.

Apply the NameCert obligations of Definition 4229.2 and the metric complement boundary route of Definition 4229.3. These two results force every closed-boundary read through $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, hsame, Cont, and $\mathsf{NameCert}$ before any consumer can inspect the boundary.

Because the accepted packet has no coordinate for a host powerset, detached open-set lattice, quotient equality, selected limit point, or ambient completeness theorem, the stated rows exhaust the ClosedSet scope basis. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetScopeBasis`

Theorem 4229.5 (ClosedSet public boundary export)**.** *The public* $\mathsf{ClosedSet}^{\uparrow}$ *export is exactly the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *row of the accepted carrier. A public consumer may cite the closed-boundary packet only through those rows and through the scope basis; it receives no host powerset semantics, detached open-set lattice, selected limit point, quotient equality of names, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or compact-image closedness outside the displayed consumer route.*

Proof. Project the carrier of Definition 4229.1. It exposes precisely the topology, metric, separated-metric, classifier, complement ledger, metric-boundary, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations, metric complement boundary, and scope basis from Definition 4229.2, Definition 4229.3, and Definition 4229.4. These results force every public closed-boundary read through the same finite row inventory and block any route through host powersets, detached open-set lattices, quotient equality, selected limit points, choice, or ambient completeness.

Thus the public export is not a larger closure theorem; it is the named boundary surface already displayed by the carrier and its metric complement route. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetPublicBoundaryExport`

Theorem 4229.6 (ClosedSet bridged boundary export)**.** *Any future standard bridge for* $\mathsf{ClosedSet}^{\uparrow}$ *must read exactly the topology row, metric row, separated-metric row, classifier row, open-neighborhood complement ledger, metric complement boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$ *row already exported by Definition 4229.5. A bridge consumer may replay those rows as the closed-boundary surface, but it receives no host powerset semantics, detached open-set lattice, selected limit point, quotient equality of names, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or compact-image closedness outside that displayed consumer route.*

Proof. Start with the carrier of Definition 4229.1. Its coordinates are exactly $T, M, S, F, O, B, H, C, P, N$, so a bridge has no additional coordinate from which to read a host powerset, detached open-set lattice, selected point, quotient equality witness, choice principle, or ambient completeness theorem.

Apply the NameCert obligations of Definition 4229.2 and the metric complement boundary of Definition 4229.3. These force topology exposure through T, metric exposure through M, S, B, complement compatibility through F, O, B, and replay through H, C, P, N before the closed-boundary row can be consumed.

The scope basis of Definition 4229.4 and the public export of Definition 4229.5 then identify the bridge-facing surface with the already public row inventory. Thus the bridge reads the displayed consumer route and nothing stronger, including no compact-image closedness theorem beyond the named downstream route. □

Lean4 ✓ `BEDC.Derived.ClosedSetUp.TasteGate.ClosedSetBridgedBoundaryExport`

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedSet}^{\uparrow}$ packet is generated from topological, metric, separated-metric, membership-classifier, open-neighborhood complement, metric complement boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 4229.1, Definition 4229.2, Definition 4229.3, Definition 4229.4, Definition 4229.5, and Definition 4229.6 constitute the mature paper-side treatment of the finite closed-set carrier, local NameCert obligation surface, metric complement boundary route, scope basis, public boundary export, and bridge-facing export over the displayed topology, metric, separated-metric, complement, transport, replay, provenance, local naming, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host powerset semantics, detached open-set lattices, selected limit points, quotient equality of names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or compact-image closedness outside the displayed consumer route.

Upgrade path: Verification progress requires Lean-side coverage for the mature carrier admission, topology compatibility, metric compatibility, complement exactness, complement-boundary transport, metric-boundary replay, transport replay, nonescape, provenance exposure, local naming, and TasteGate field faithfulness for BEDC.Derived.ClosedSetUp.

Chapter 4230

A Concrete Naming Certificate for $\mathsf{Met}^{\uparrow}$

Following Bishop and Bridges and Bourbaki, the $\mathsf{Met}^{\uparrow}$ packet records the traditional metric-space interface as finite BEDC NameCert data. It sits over the generic distance-witness surface of chapter 57, the real-valued distance route of chapter 1638, the dyadic tolerance core of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter gives completion, topology, total-boundedness, compactness, and Polish-space consumers a single cited carrier surface without importing quotient equality, selected limits, countable choice, or ambient completeness.

Definition 4230.1 (MetricSpace carrier)**.** *A* $\mathsf{Met}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-carrier row. It names the displayed point observations and the finite windows through which downstream consumers may inspect them.*
2. *σ is the point-classifier row. It records the accepted point equivalence used by* hsame *transport and by the zero-distance classifier handoff.*
3. *D is the* $\mathsf{Real}^{\uparrow}$*-valued distance row. It is read through the* $\mathsf{Met}^{\uparrow}$ *witness surface of chapter 57 and the* $\mathsf{RealMetric}^{\uparrow}$ *distance route of chapter 1638.*
4. *Z is the zero-distance classifier row, requiring any request that turns distance zero into point equivalence to pass through σ rather than through host equality.*
5. *S is the symmetry row, carrying the componentwise reversal of endpoint observations and the corresponding distance readback.*
6. *T is the triangle row, carrying the three endpoint-pair distance reads and the dyadic tolerance budget used to compose them.*
7. *H records componentwise* hsame *transport for the point carrier, classifier, distance, zero-distance, symmetry, triangle, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through the finite route*

$$X, \sigma \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, \sigma, D, Z, S, T, H, C, P, N$. It has no coordinate for host equality of points, quotient completion equality, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached topology.*

Lean4 ✓ `BEDC.Derived.MetricSpaceUp.MetricSpaceTasteGate single carrier alignment`

Theorem 4230.2 (MetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{Met}^{\uparrow}$ *carrier*

$$M = (X, \sigma, D, Z, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Met}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the point carrier, point classifier, real-valued distance, zero-distance classifier, symmetry, triangle, transport, replay, provenance, and local naming rows;*
2. *distance visibility, requiring every distance read to expose the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RealMetric}^{\uparrow}$ *rows before any downstream consumer reads a real-valued metric comparison;*
3. *zero-distance exactness, requiring zero distance to return to the displayed classifier σ through Z and never through host point equality;*
4. *symmetry replay, requiring endpoint reversal to be witnessed by S together with the same* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback route;*
5. *triangle-budget replay, requiring the three distance reads and their dyadic comparison budget to be displayed by T before a triangle consumer may use the row;*
6. *structural preservation through H, C, P, N.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4230.1. An accepted packet displays the point-carrier row X, the classifier σ, the distance row D, the zero-distance row Z, the symmetry row S, the triangle row T, and the structural rows H, C, P, N.

Distance visibility is forced by the placement of D in the carrier. A distance consumer first opens the generic $\mathsf{Met}^{\uparrow}$ witness surface and the $\mathsf{RealMetric}^{\uparrow}$ readback route, and only then reads the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows needed for a finite comparison.

The zero-distance, symmetry, and triangle clauses are rowwise. The row Z is the only route from distance zero back to the point classifier, S is the only endpoint-reversal row, and T is the only triangle-budget row.

Finally, H, C, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no other coordinate, the listed rows exhaust the local NameCert surface and cannot export host equality, quotient completion equality, selected representatives, countable choice, ambient completeness, or a detached topology. □

Theorem 4230.3 (MetricSpace distance route)**.** *Every completion, topology, total-boundedness, compactness, or Polish-space consumer of an accepted* $\mathsf{Met}^{\uparrow}$ *carrier reads distance information through the finite route*

$$X, \sigma, D \longmapsto \mathsf{Met}^{\uparrow}, \mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto Z, S, T, H, C, P, N.$$

Thus the public distance surface is the displayed BEDC route from point observations and classifier data to real-valued metric reads, zero-distance classification, symmetry, and triangle replay. It exports no quotient equality, selected limit, countable-choice schedule, ambient completeness theorem, or open-set lattice.

Proof. Let a downstream consumer inspect an accepted carrier $M = (X, \sigma, D, Z, S, T, H, C, P, N)$. The point surface is opened first through X and σ, so every later distance read remains attached to the displayed point classifier.

The distance row D then routes the request through $\mathsf{Met}^{\uparrow}$ and $\mathsf{RealMetric}^{\uparrow}$. The numerical readback is finite because the dyadic tolerance, stream-window, regular-rational, and real-name rows are displayed before any metric comparison is consumed.

Metric laws are read only after this route is visible. Zero distance returns through Z to σ, symmetry reverses endpoints through S, and triangle use exposes the three pairwise reads and dyadic budget through T.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since none of the excluded objects appears in the carrier, no listed downstream consumer can read quotient equality, a selected limit, a choice schedule, ambient completeness, or an open-set lattice from the $\mathsf{Met}^{\uparrow}$ packet. □

Chapter 4231

A Concrete Naming Certificate for $\mathsf{NashEquilibrium}^{\uparrow}$

Following Nash's equilibrium theorem, the $\mathsf{NashEquilibrium}^{\uparrow}$ packet records the finite best-response surface for a displayed finite normal-form game before any mixed strategy, payoff comparison, or equilibrium witness is treated as a host object. It sits after the compact metric and convex strategy-simplex surfaces of chapter 129 and chapter 208, the real payoff rows of chapter 38, and the fixed-point anchors of chapter 2030 and chapter 3511. The chapter names the finite BEDC carrier needed by later game-theoretic consumers without exporting a host simplex, selected optimizer, quotient mixed-strategy equality, semantic continuity, or an ambient fixed-point theorem.

Definition 4231.1 (Nash equilibrium carrier)**.** *A* $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (P, S, A, U, B, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite player ledger. It records the inspected player names, their classifier rows, and the finite dependency order in which player-indexed strategy and payoff rows are read.*
2. *S is the finite strategy-simplex ledger. For each player it displays the finitely generated mixed-strategy simplex through compact metric and convex-set rows, together with the finite coordinate windows consumed by the game packet.*
3. *A is the action-window ledger. It records, for each player, the finite pure-action windows that underlie the displayed simplex coordinates and the finite product windows inspected by payoff and best-response consumers.*
4. *U is the payoff-row ledger. It reads payoff comparisons only through displayed* $\mathsf{Real}^{\uparrow}$ *rows attached to finite action profiles and finite mixed-strategy windows.*
5. *B is the best-response ledger. It records the finite argmax or best-response correspondence rows as displayed comparison data over P, S, A, U, without selecting a host optimizer.*
6. *E is the equilibrium row. It states that the displayed mixed-strategy profile is read through B as a fixed point of the best-response ledger, with the fixed-point route exposed only through the cited Brouwer or Kakutani packet.*
7. *H records componentwise* hsame *transport for player ledgers, simplex rows, action windows, payoff rows, best-response rows, equilibrium rows, replay routes, provenance rows, and local naming rows.*

8. *C records displayed* Cont *continuation from a finite game request through the player, simplex, action, payoff, best-response, fixed-point, transport, provenance, and local naming rows.*
9. *Q is the* Pkg *provenance over* $\mathsf{NashEquilibrium}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It stores no host mixed-strategy simplex, no selected maximizer, no quotient equality of strategies, no semantic payoff continuity, no countable choice, and no ambient fixed-point principle.

Lean4 ✓ `BEDC.Derived.NashEquilibriumUp.NashEquilibriumTasteGate single carrier alignment`

Theorem 4231.2 (Nash equilibrium NameCert obligations)**.** *For every accepted* $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier*

$$G = (P, S, A, U, B, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$ *surface consists exactly of finite player admission, finite strategy-simplex exposure, action-window exposure, payoff-row readability, best-response ledger exposure, equilibrium-row fixed-point routing, componentwise* hsame *transport, displayed* Cont *continuation,* Pkg *provenance, and local naming. No NameCert obligation is stored outside the rows of Definition 4231.1.*

Proof. Project the accepted carrier. The player, strategy, action, payoff, best-response, and equilibrium data are precisely the rows P, S, A, U, B, E.

The finite game surface is read first. The player ledger P indexes the displayed strategy-simplex rows S, while the action-window ledger A supplies the finite product windows inspected by payoff and best-response consumers.

The payoff and best-response obligations are then read through U and B. Payoff comparisons consume only the displayed $\mathsf{Real}^{\uparrow}$ rows, and the best-response ledger records comparison and correspondence data without adding a selected host optimizer.

Finally the equilibrium row E is consumed through the displayed fixed-point route, and the structural rows H, C, Q, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate beyond these rows, the listed surface exhausts the local NameCert obligations. □

Theorem 4231.3 (Nash equilibrium best-response coverage)**.** *Let*

$$G = (P, S, A, U, B, E, H, C, Q, N)$$

be an accepted $\mathsf{NashEquilibrium}^{\uparrow}$ *carrier. Every equilibrium-facing read of E must first expose the player ledger P, the finite strategy-simplex rows S, the action windows A, the payoff rows U, and the best-response ledger B. The read may cite* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, *and* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *only through those displayed rows and the structural rows H, C, Q, N; it exports no host simplex, selected best response, quotient strategy equality, semantic continuity, countable choice, or ambient fixed-point theorem.*

Proof. Begin with the carrier definition. The equilibrium row E is downstream of P, S, A, U, B, so no equilibrium-facing consumer can inspect E before the finite player, strategy, action, payoff, and best-response ledgers have been displayed.

The compact and convex rows enter only through S, and the real payoff comparisons enter only through U. The fixed-point route is recorded by the best-response and equilibrium rows rather than by a separate host theorem coordinate.

Apply the NameCert obligation theorem. It makes transport, continuation, provenance, and local naming part of the same displayed route, so any consumer of E receives only the finite best-response certificate.

The excluded host simplex, selected optimizer, quotient equality, semantic continuity, choice, and ambient fixed-point objects are absent from the carrier inventory. Thus the best-response coverage route is exhausted by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}$.

Constructive story: A $\mathsf{NashEquilibrium}^{\uparrow}$ packet is generated from a finite player ledger, finite strategy-simplex rows, action windows, payoff Real rows, best-response ledger, equilibrium fixed-point row, componentwise hsame transport, displayed Cont continuation, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4231.1, Definition 4231.2, and Definition 4231.3 seed the finite best-response certificate surface over player, simplex, action, payoff, best-response, equilibrium, transport, continuation, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NashEquilibrium}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host mixed-strategy simplices, selected optimizers, quotient strategy equality, semantic payoff continuity, countable choice, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite simplex exposure, action-window coverage, payoff-row readability, best-response ledger exactness, equilibrium-row nonescape, transport stability, continuation stability, provenance stability, and local naming for BEDC.Derived.NashEquilibriumUp.

$\mathsf{CauchyFilterSubbasis}^{\uparrow}$ consumes the generated $\mathsf{FilterBase}^{\uparrow}$ route of chapter 1601 before handing generated requests to chapter 333.

Chapter 4232

A Concrete Naming Certificate for $\mathsf{CauchyFilterSubbasis}^{\uparrow}$

Following Kelley and Bourbaki, the $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ packet records the standard subbasis-to-filter generation surface needed before a completion-facing Cauchy filter can be read as a finite BEDC route. It sits between the filter-base interface of chapter 1601, the Cauchy-filter packet of chapter 333, the completion boundary of chapter 1569, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite generation interface only; it does not quotient filters, select limits, invoke countable choice, use host equality, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

The separated uniform completion of chapter 7566 and the effective regular real extensionality surface of chapter 7009 consume this subbasis route only after $\mathsf{FilterBase}^{\uparrow}$ and $\mathsf{CauchyFilter}^{\uparrow}$ generation are displayed and the finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows remain visible. The consumer route is therefore a finite generated-base handoff, not an isolated subbasis generator.

Definition 4232.1 (Cauchy filter subbasis carrier)**.** *A* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N)$$

with the following displayed rows:

1. *U is the finite subbasis-token row naming the primitive neighbourhood requests available to the packet.*
2. *I is the finite-intersection row, recording which finite families of subbasis tokens are admitted as generated basic requests.*
3. *E is the generated-event row, displaying the* BHist *surface attached to each finite intersection request.*
4. *G is the generation ledger, storing the inclusion reads from subbasis tokens into generated base events and the closure reads from finite intersections into base requests.*
5. *B is the generated* $\mathsf{FilterBase}^{\uparrow}$ *row exposed to downstream filter consumers.*
6. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row obtained from the generated base together with the displayed Cauchy comparison requests.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when a completion-facing consumer reads generated tails.*

8. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-accounting row for the generated Cauchy comparisons.*
9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for retained regular rational windows.*
10. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the generated base, Cauchy row, window row, tolerance ledger, and readback row have been displayed.*
11. *H records componentwise* hsame *transport for subbasis tokens, finite intersections, generated events, generation ledgers, filterbase rows, Cauchy rows, windows, tolerances, readbacks, Real seals, replay, provenance, and local naming.*
12. *C records displayed* Cont *replay from subbasis requests through finite intersections, generated filterbase rows, Cauchy-filter rows, finite windows, tolerance accounting, readback, and Real sealing.*
13. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$ *row.*

The classifier reads only $U, I, E, G, B, F, W, T, R, L, H, C, P, N$. It has no coordinate for an ultrafilter extension, quotient equality of filters, selected limits, countable-choice schedules, host equality of generated sets, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CauchyFilterSubbasisUp.CauchyFilterSubbasisUp`

Theorem 4232.2 (Cauchy filter subbasis NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier $S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$ *surface consists exactly of carrier admission, finite subbasis-token exposure, finite-intersection generation, generated-event admission, generation-ledger coverage,* $\mathsf{FilterBase}^{\uparrow}$ *handoff,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4232.1. The visible generation data are exactly the rows U, I, E, G, so a consumer first reads primitive tokens, finite intersections, generated events, and the inclusion ledger.

The generated filter route is then B, F, W, T, R, L. The $\mathsf{FilterBase}^{\uparrow}$ row is exposed before the $\mathsf{CauchyFilter}^{\uparrow}$ row, and the finite windows, dyadic tolerance ledger, regular rational readback, and Real seal are downstream from that generated filter route.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the classifier contains no coordinate for an ultrafilter extension, quotient filter equality, selected limit, countable-choice schedule, host equality, or ambient completeness, the local NameCert surface is exactly the displayed finite packet. □

Theorem 4232.3 (Cauchy filter subbasis filterbase generation)**.** *Every completion-facing read exported by an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier factors through the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B before it reaches the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the* $\mathsf{Real}^{\uparrow}$ *seal row L. Thus subbasis generation is visible as finite BEDC data rather than as a hidden host filter construction.*

Proof. Begin with the generation rows of Definition 4232.1. The only route from primitive subbasis tokens to a downstream filter read is through U, I, E, G, and the generated base is stored as the row B.

Apply the NameCert obligation theorem. It requires every Cauchy-filter and completion-facing consumer to inspect the generated $\mathsf{FilterBase}^{\uparrow}$ handoff before reading F, W, T, R, L.

The structural rows preserve the same route. Componentwise transport H, displayed replay C, provenance P, and local naming N keep the subbasis-to-filter generation inside the finite packet.

No excluded coordinate remains. Hence quotient filters, selected limits, countable-choice schedules, host equality of generated filters, and ambient completeness cannot be used in place of the displayed $\mathsf{FilterBase}^{\uparrow}$ generation row. □

Theorem 4232.4 (Cauchy filter subbasis generated-base cofinality)**.** *Every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier exposes the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B as cofinal among the finite intersections named by I and their displayed generated-event rows E. A completion-facing consumer must therefore read U, I, E, G, B before it reads the* $\mathsf{CauchyFilter}^{\uparrow}$ *row F, the* $\mathsf{StreamName}^{\uparrow}$ *windows W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, or the terminal* $\mathsf{Real}^{\uparrow}$ *seal L.*

Proof. Project the carrier of Definition 4232.1. The only finite-intersection data are I, the only generated events are E, and the only inclusion and coverage reads from those rows into a base are stored in G.

Apply the NameCert obligations of Definition 4232.2. They force generated-event admission, generation-ledger coverage, and the $\mathsf{FilterBase}^{\uparrow}$ handoff to be displayed as local certificate rows of the same packet.

Use the filterbase generation theorem Definition 4232.3. Since every completion-facing read must factor through B before reaching F, W, T, R, L, the generated base is cofinal for the finite intersections visible to the packet.

The classifier has no coordinate for quotient filters, selected limits, countable choice, host equality of generated filters, or ambient completeness. Hence no external filter object can replace the displayed cofinal generated-base row. □

Theorem 4232.5 (Cauchy filter subbasis finite-intersection route)**.** *Every generated base request in an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier is read through the finite subbasis-token row U, the finite-intersection row I, the generated-event row E, and the generation ledger G before the generated* $\mathsf{FilterBase}^{\uparrow}$ *row B is available. The route exports no quotient filters, selected limits, countable choice, host equality, or ambient completeness.*

Proof. Project the carrier of Definition 4232.1. The rows U, I, E, G are the only displayed route from primitive subbasis tokens to generated base events.

Apply the NameCert obligations of Definition 4232.2. They require finite subbasis-token exposure, finite-intersection generation, generated-event admission, and generation-ledger coverage before the $\mathsf{FilterBase}^{\uparrow}$ handoff is a local certificate row.

Use generated-base cofinality, Definition 4232.4. Since every completion-facing consumer must read U, I, E, G, B before the downstream rows, the generated base request is available only after the finite-intersection route has been displayed.

The excluded coordinates are absent from the carrier. Hence quotient filters, selected limits, countable choice, host equality, and ambient completeness cannot replace the rows U, I, E, G, B. □

Theorem 4232.6 (Cauchy filter subbasis refinement stability)**.** *Suppose an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier replaces its generated base row B by a displayed cofinal* $\mathsf{FilterRefinement}^{\uparrow}$ *row. Then the completion-facing route still reaches the same* $\mathsf{CauchyFilter}^{\uparrow}$ *row F,* $\mathsf{StreamName}^{\uparrow}$ *windows W,*

$\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R, *and* $\mathsf{Real}^{\uparrow}$ *seal* L. *The replacement exports no quotient filter, selected limit, countable-choice schedule, host equality of generated filters, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the generated-base cofinality theorem. It identifies B as the displayed cofinal base among finite intersections before any Cauchy-filter or completion-facing row can be read.

Apply the Cauchy-preservation and completion-route clauses for $\mathsf{FilterRefinement}^{\uparrow}$, namely Definition 1626.2 and Definition 1626.3. The refinement row may replace a displayed base request by a cofinal target request, but it preserves the Cauchy-filter read through the finite comparison surface.

Compose that refinement route with Definition 4232.3. The downstream rows F, W, T, R, L remain attached to the same subbasis-generated route because the consumer still passes through a displayed cofinal base before reading them.

Finally inspect the excluded coordinates of the carrier. Since quotient filters, selected limits, choice schedules, host equality of generated filters, and ambient Real completeness are absent from $U, I, E, G, B, F, W, T, R, L, H, C, P, N$, the refinement cannot import any of them. □

Theorem 4232.7 (Cauchy filter subbasis completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier must read the rows*

$$U, I, E, G, B, F, W, T, R, L, H, C, P, N$$

in the displayed order. In particular, the generated $\mathsf{FilterBase}^{\uparrow}$ *row* B *is available before the* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F, *and the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows* W, T, R *are displayed before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* L.

Proof. Read the carrier named in Definition 4232.1. Its classifier exposes exactly the listed rows, with U, I, E, G providing the finite subbasis tokens, finite intersections, generated events, and generation ledger.

Apply Definition 4232.2. The NameCert obligations require the generated $\mathsf{FilterBase}^{\uparrow}$ handoff, the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, the window row, the tolerance row, the readback row, and the Real seal to appear as displayed fields of the same finite packet.

Finally apply Definition 4232.3. That theorem forces every completion-facing read through B before F, W, T, R, L, so the handoff cannot bypass the generated base or reach the terminal $\mathsf{Real}^{\uparrow}$ seal before the window, tolerance, and readback rows have been displayed. □

Theorem 4232.8 (Cauchy filter subbasis window transport)**.** *Componentwise* hsame *transport of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier preserves finite subbasis tokens, finite intersections, generated events, generation ledgers, generated* $\mathsf{FilterBase}^{\uparrow}$ *rows,* $\mathsf{CauchyFilter}^{\uparrow}$ *rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks. The transported packet adds no quotient filter, selected limit, countable-choice schedule, host equality of generated filters, or ambient completeness coordinate.*

Proof. Project the transport row H from Definition 4232.1. It is componentwise over the visible rows, so transport acts on tokens, intersections, generated events, ledgers, generated bases, Cauchy rows, windows, tolerances, readbacks, seals, replay, provenance, and local naming without changing the classifier shape.

Use Definition 4232.2 to identify the rows preserved by the local NameCert surface. Each transported read remains attached to the corresponding displayed obligation, including the generated $\mathsf{FilterBase}^{\uparrow}$ handoff and the downstream Cauchy-filter, window, tolerance, and readback obligations.

Apply Definition 4232.3 to the transported route. Since every downstream completion-facing read still factors through the generated base row, no quotient filter or selected-limit coordinate can be inserted by transport. □

Theorem 4232.9 (Cauchy filter subbasis regular readback)**.** *For an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *carrier, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *is available only after the generated* $\mathsf{CauchyFilter}^{\uparrow}$ *row* F *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T *have been displayed. The* $\mathsf{Real}^{\uparrow}$ *seal* L *remains terminal for the route and cannot be used as a prior source for the readback.*

Proof. The carrier of Definition 4232.1 places R after the generated base, Cauchy-filter, window, and tolerance rows. Thus the readback row is not a primitive subbasis token and is not a replacement for the generated filter route.

By Definition 4232.2, the local certificate displays the $\mathsf{CauchyFilter}^{\uparrow}$ handoff, $\mathsf{StreamName}^{\uparrow}$ window exposure, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance accounting, and $\mathsf{RegSeqRat}^{\uparrow}$ readback as distinct obligations. The readback therefore depends on the displayed Cauchy-filter and tolerance ledger surface rather than on an ambient limit.

Apply Definition 4232.3. It keeps L after B, F, W, T, R, so the terminal $\mathsf{Real}^{\uparrow}$ seal cannot justify the readback in advance; it only seals the finite route after the readback row has been displayed. □

Falsifiable prediction: Every accepted $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ completion-facing read factors through finite subbasis tokens, finite intersections, generated events, the generation ledger, a generated FilterBaseUp row, a CauchyFilterUp row, StreamNameUp windows, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Independence witness: filterbase, cauchyfilter, cauchyfiltercompletion, streamname, regseqrat, real: $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries primitive subbasis tokens, finite-intersection generation, generated events, generation-ledger coverage, generated filterbase handoff, Cauchy-filter handoff, finite windows, tolerance accounting, readback, Real sealing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ packet is generated from finite subbasis tokens, finite-intersection rows, generated-event rows, a generation ledger, a generated FilterBaseUp row, a CauchyFilterUp handoff, StreamNameUp windows, DyadicRatCoreUp tolerance accounting, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4232.1, Definition 4232.2, and Definition 4232.3 seed the finite subbasis-to-filter generation route over generated filterbase, Cauchy-filter, window, tolerance, readback, seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSubbasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality of generated filters, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-intersection generation, generation-ledger coverage, generated FilterBaseUp handoff, CauchyFilterUp handoff, window exposure, tolerance accounting, readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterSubbasisUp.

The $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ carrier refines the filter-base row of chapter 1601 and feeds the Cauchy-filter carrier of chapter 333. This is a substantive sibling dependency: the basic-set closure supplied by the generated $\mathsf{FilterBase}^{\uparrow}$ row and the Cauchy-directedness consumed by the downstream $\mathsf{CauchyFilter}^{\uparrow}$ row are separate certificate obligations, not self-references or restatements of the local subbasis packet.

Frechet tail dependency. The $\mathsf{FrechetFilter}^{\uparrow}$ packet of chapter 1564 supplies the cofinite-tail specialization of the subbasis-generated filter route. The specialization is consumed through the displayed base discipline of chapter 1601 before the generated route reaches chapter 333 and the Real-facing seal.

A $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ read that is narrowed to eventual sequence tails therefore factors through the same finite threshold, filter-base, stream-schedule, regular-readback, and Real-seal rows exposed by $\mathsf{FrechetFilter}^{\uparrow}$. This is a displayed sibling dependency in the tail-base factorization route, not a citation invented after the fact, and the subbasis packet retains only the generated Cauchy-filter surface rather than quotient filter equality, selected limits, countable choice, or ambient completion.

Theorem 4232.10 (Cauchy filter subbasis tail-base factorization)**.** *Every* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *read specialized to cofinite sequence tails factors through* $\mathsf{FrechetFilter}^{\uparrow}$ *and* $\mathsf{FilterBase}^{\uparrow}$ *before* $\mathsf{CauchyFilter}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *are read. For an accepted carrier*

$$S = (U, I, E, G, B, F, W, T, R, L, H, C, P, N),$$

the tail-specialized route first displays the finite subbasis tokens, finite intersections, generated events, and generation ledger U, I, E, G*; then reads the cofinite-tail threshold discipline of chapter 1564; then returns through the generated* $\mathsf{FilterBase}^{\uparrow}$ *row* B *before the Cauchy row, stream windows, tolerance ledger, regular readback, and Real seal* F, W, T, R, L *can be consumed.*

Proof. Begin with the generated-base cofinality theorem, Definition 4232.4. It places U, I, E, G, B before every completion-facing read, so a tail-specialized consumer cannot reach F, W, T, R, L before the generated base has been displayed.

Specialize the generated requests to eventual sequence tails. The Frechet tail dependency row in this chapter identifies the cofinite-tail discipline supplied by chapter 1564; this row is the only tail-filter surface available to the specialized read.

Compose the specialization with the completion handoff of Definition 4232.7. The handoff routes the resulting base through the $\mathsf{CauchyFilter}^{\uparrow}$ row, finite stream windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

The carrier exclusions then apply to the whole composition. Quotient filters, selected limits, countable choice, host equality of generated filters, and ambient Real completeness are absent from the displayed rows, so none can replace the Frechet and FilterBase stages of the tail-base factorization. □

The hub-level $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ route is read through the finite dependency lattice of chapter 1601, chapter 333, chapter 126, chapter 301, and chapter 38. This route exports no quotient filter, selected limit, countable choice, or ambient completeness. The top-level $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ route consumes the finite $\mathsf{FilterBase}^{\uparrow}$ generation rows of chapter 1601 before it reads the $\mathsf{CauchyFilter}^{\uparrow}$ packet of chapter 333. Only after that base-to-Cauchy handoff does the route pass through the $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 126, chapter 301, and chapter 38. This is a finite generation route, not a quotient filter, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Theorem 4232.11 (CauchyFilterSubbasis Cauchy-completion sibling route)**.** *Every accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *packet exported to a completion-facing consumer first reads the Cauchy-completion criterion surface of chapter 3885. The required dependency is concrete: subbasis neighborhoods are generated from the finite* $\mathsf{FilterBase}^{\uparrow}$ *rows, the Cauchy filter completion criterion supplies the completion rows,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *provide only finite windows and rational readback, and the terminal*

$\mathsf{Real}^{\uparrow}$ *seal records nonescape from selected limits, quotient filters, countable choice, and ambient completeness.*

Proof. Project the top-level route described above. The subbasis neighborhood row is read before any Cauchy packet is exported, so the consumer begins with finite $\mathsf{FilterBase}^{\uparrow}$ generation and the displayed $\mathsf{CauchyFilter}^{\uparrow}$ handoff.

Next read the Cauchy-completion criterion row. It supplies the completion-facing criterion used by the sibling chapter, while the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows restrict every read to finite stream windows and rational approximants.

The final $\mathsf{Real}^{\uparrow}$ seal is only a nonescape row. It does not select a limit, identify quotient filters, introduce countable choice, or assert ambient completeness beyond the finite criterion route already displayed. □

Theorem 4232.12 (CauchyFilterSubbasis Real-uniform lattice route)**.** *Every Real-uniform consumer of an accepted* $\mathsf{CauchyFilterSubbasis}^{\uparrow}$ *packet must pass through finite* $\mathsf{FilterBase}^{\uparrow}$ *generation,* $\mathsf{CauchyFilter}^{\uparrow}$ *completion criteria,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal rows before any* $\mathsf{RealUniformStructure}^{\uparrow}$ *or completion-facing consumer can read it. The route exports a generated subbasis lattice edge, not quotient filters, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 4232.11. It forces any completion-facing consumer to read the Cauchy-completion criterion only after finite $\mathsf{FilterBase}^{\uparrow}$ generation and the displayed $\mathsf{CauchyFilter}^{\uparrow}$ handoff.

Now compose that sibling route with the generated-base and cofinality reading described in the top-level route. The subbasis row enters the Real-uniform lattice as generated filter-base data, so a $\mathsf{RealUniformStructure}^{\uparrow}$ consumer cannot bypass the finite base or treat the subbasis as an ambient quotient filter.

The final handoff is through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows expose only finite windows, rational readback, and a terminal seal, so no selected limit, countable-choice schedule, quotient-filter equality, or ambient completeness principle is available to the consumer. □

Chapter 4233

A Concrete Naming Certificate for $\mathsf{BishopCompleteReal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCompleteReal}^{\uparrow}$ packet records the textbook complete-real interface as a finite BEDC boundary rather than as an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$. It sits on the dyadic endpoint ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated completion surface of chapter 1612, and the completion reflection boundary of chapter 403. The packet exposes the L10 completion interface currently needed by Real and Completion consumers; it does not assert that $\mathsf{Real}^{\uparrow}$ is complete, select limits, quotient streams, or import countable choice.

Definition 4233.1 (Bishop complete real carrier)**.** *A* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, E, S, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger used to measure regular Cauchy requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite schedule row opened by the same tolerance request.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to D and W.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the dyadic, schedule, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *comparison row, recording that two completion-facing reads are compared through separated finite evidence.*
6. *F is the* $\mathsf{CompletionReflection}^{\uparrow}$ *handoff row, replaying the same regular-rational source surface into the completion-facing interface.*
7. *H stores componentwise* hsame *transport for $D, W, R, E, S, F, C, P, N$.*
8. *C stores displayed* Cont *replay from a completion-facing request through the same dyadic, schedule, regular, real-seal, separated, and reflection rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCompleteReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CompletionReflection}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real equality, quotient stream equality, a selected limit, countable choice, open-cover compactness, or a theorem that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.TasteGate.BishopCompleteRealTasteGate single carrier alignment`

Theorem 4233.2 (Bishop complete real NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *carrier*

$$B = (D, W, R, E, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, continuation replay, provenance, and local naming rows, together with the obligation that every completion-facing read passes through* $D, W, R, E, S, F, H, C, P, N$ *in that order.*

No obligation in the packet is discharged by host real equality, quotient streams, selected representatives, countable choice, open-cover compactness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Carrier admission is projection from Definition 4233.1. The tuple lists the dyadic, schedule, regular readback, Real seal, separated comparison, completion reflection, transport, replay, provenance, and naming rows.

The completion-facing order is fixed by the visible rows. A request first reads a dyadic endpoint and tolerance ledger in D, then opens the finite schedule row W, and then passes through the regular rational readback row R before the terminal Real seal E is available.

The separated and reflection rows then consume that same source surface. The comparison row S records separated evidence, while F replays the regular-rational source into the completion-facing boundary.

Finally H, C, P, N transport, replay, source, and name only these displayed rows. Since the carrier contains no coordinate for the excluded host, quotient, choice, compactness, or completeness data, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BishopCompleteRealUp.BishopCompleteRealNamecertObligations`

Theorem 4233.3 (Bishop complete real L10 handoff)**.** *Every Real and Completion consumer of an accepted* $\mathsf{BishopCompleteReal}^{\uparrow}$ *packet factors through the finite L10 route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow} \longrightarrow \mathsf{SeparatedCompletion}^{\uparrow} \longrightarrow \mathsf{CompletionReflection}^{\uparrow}.$$

The handoff exports a displayed regular-rational completion interface with separated comparison and reflection rows. It does not export completed $\mathsf{Real}^{\uparrow}$ *equality, quotient Cauchy classes, selected host limits, countable-choice coordinates, open-cover compactness, or an ambient completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 4233.2. They force every accepted read to pass through the dyadic ledger, finite schedule, regular rational readback, and terminal Real seal before any completion-facing row is consumed.

Next read the separated comparison row. The row S compares completion-facing observations only through displayed separated evidence, so it cannot introduce quotient classes or host equality.

The completion reflection row F then replays the same source surface into the completion-facing interface. The structural rows H, C, P, N preserve that route componentwise and add only transport, replay, provenance, and local naming.

Thus the consumer receives exactly the displayed L10 handoff from dyadic data to schedules, regular rational readback, Real seal, separated comparison, and completion reflection. None of the excluded completeness, choice, quotient, compactness, or selected-limit data has an entry point in the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompleteReal}^{\uparrow}$ packet is generated from a dyadic endpoint ledger, finite schedule, regular rational readback, terminal Real seal, separated completion comparison, completion reflection handoff, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4233.1, Definition 4233.2, and Definition 4233.3 seed the finite Bishop complete-real interface for the current L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompleteReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient real equality, selected host limits, countable choice, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated comparison, completion reflection handoff, and ledger non-escape for the same packet.

Chapter 4234

Cauchy compact extraction NameCert construction

Following Bishop and Bridges on compact metric spaces, the $\mathsf{CauchyCompact}^{\uparrow}$ extraction packet records the finite route by which total boundedness and completeness turn regular rational observations into a displayed limit seal. It refines the constructive Cauchy-extraction face of chapter 2117, consumes the regular sequence interface of chapter 301, and keeps the dyadic tolerance bookkeeping visible through chapter 291. The chapter names a finite extraction surface only; it does not assert open-cover compactness, a choice-selected subsequence, or host-level real completeness.

Definition 4234.1 (Cauchy compact extraction carrier)**.** *A* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier is a finite* BHist *packet*

$$E = (M, L, T, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, exposing the metric carrier, local classifier, and compact metric provenance consumed by the extraction.*
2. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row, exposing the finite limit-admission route for the regular Cauchy input named by the packet.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row, exposing for each dyadic tolerance request the finite net, mesh ledger, and coverage replay used to choose a bounded search window.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule, recording the finite windows of the input stream available before the limit seal is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational handoff obtained from the displayed stream windows and the same tolerance ledger.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for radii, comparisons, refinements, and tail budgets used by T, S, and R.*
7. *A is the* $\mathsf{Real}^{\uparrow}$*-facing extracted seal, reachable only after M, L, T, S, R, D have been displayed.*
8. *H stores componentwise* hsame *transport for the compact metric row, complete metric row, totally bounded row, stream schedule, regular rational handoff, dyadic ledger, extracted seal, continuation, provenance, and local name.*

9. *C stores displayed* Cont *replay of the route*

$$M, L, T, S, R, D \longmapsto A.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *extraction row.*

The classifier compares accepted extraction packets only through these rows. It has no coordinate for open covers, selected subsequences, quotient streams, ambient choice, or a host real object.

Theorem 4234.2 (Cauchy compact extraction NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier $E = (M, L, T, S, R, D, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompact}^{\uparrow}}$ *extraction surface consists exactly of the following obligations:*

1. *carrier admission for the displayed eleven-row packet;*
2. *finite-net visibility, requiring each extraction tolerance read to pass through the displayed* $\mathsf{TotallyBounded}^{\uparrow}$ *row T over the same* $\mathsf{CompactMetric}^{\uparrow}$ *row M;*
3. *regular-window compatibility, requiring the* $\mathsf{StreamName}^{\uparrow}$ *windows in S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff R to use the same dyadic tolerance data D;*
4. *completeness handoff, requiring the extracted seal A to be reached through the* $\mathsf{CompleteMetric}^{\uparrow}$ *row L after the displayed regular Cauchy route has been admitted;*
5. *ledger confinement, requiring H, C, P, N to transport, replay, source, and name only the displayed extraction route and to export no open-cover compactness, choice-selected subsequence, quotient stream equality, ambient choice, or host real equality.*

Proof. Project the accepted carrier. The packet displays exactly the compact metric row, complete metric row, total-bounded finite-net row, stream schedule, regular rational handoff, dyadic ledger, extracted seal, transport, continuation, provenance, and local naming rows.

Finite-net visibility is read from T over M. A tolerance request reaches the extraction only by reading the dyadic budget in D and the finite net supplied by the displayed total-bounded row.

Regular-window compatibility is forced before the completion step. The stream windows in S and the regular rational handoff R share the same dyadic ledger D, so the completion row cannot read an unnamed Cauchy input.

Finally L produces the seal A and H, C, P, N preserve that same route. Since the tuple contains no row for open covers, selected subsequences, quotient streams, ambient choice, or a host real object, the listed obligations exhaust the local NameCert surface. □

Theorem 4234.3 (Cauchy compact extraction completion handoff)**.** *Every accepted* $\mathsf{CauchyCompact}^{\uparrow}$ *extraction carrier hands compact metric data to the L10 completion route through*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{TotallyBounded}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{CompleteMetric}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

together with the structural rows BHist, hsame, Cont, Pkg, *and* NameCert. *Any consumer of the extracted seal must first read the finite net, the stream window, the regular rational handoff, the dyadic ledger, and the complete metric limit route.*

Proof. Begin with the compact and total-bounded faces. The row M fixes the metric carrier and classifier, while T supplies the finite net and coverage ledger at the dyadic tolerance recorded in D.

Read the input stream next. The row S exposes only finite observation windows, and R repacks those windows with D as the regular rational handoff used by the completion side.

Apply the completeness row last. The row L consumes the displayed regular Cauchy route and produces the seal A, whose $\mathsf{Real}^{\uparrow}$ face is downstream from the same packet data.

The transport, continuation, provenance, and local naming rows preserve this dependency order. Thus the handoff exports a finite completion-facing route and no open-cover, choice-selected, quotient, ambient-choice, or host-real escape. □

Chapter 4235

A Concrete Naming Certificate for $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$

Following Kelley and Bishop–Bridges compact-open and uniform convergence arguments, the $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ packet records the finite BEDC route by which compact-open function-space rows, compact-metric probes, equicontinuity ledgers, and uniform-modulus rows are read as one convergence surface. It sits beside the compact-open function-space carrier of chapter 2221, the compact-open topology surface of chapter 4138, the compact-metric source of chapter 129, and the equicontinuity family route of chapter 4076. The packet is a finite convergence carrier for compact-window consumers; it does not assert host function equality, arbitrary open-cover compactness, selected subnet convergence, quotient maps, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4235.1 (Compact-open uniform convergence carrier)**.** *A* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, K, E, Q, M, G, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$ *source row, exposing the finite function-space window to be tested.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *compact probe row that supplies the displayed compact window.*
3. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family row over the same compact probe.*
4. *Q is the* $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$ *row, recording the shared compact-window radius schedule.*
5. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *convergence threshold consumed by the downstream convergence read.*
6. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the displayed maps on the compact probe.*
7. *D is the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *target-distance ledger used to compare displayed values.*
8. *R is the compact-open replay row, requiring every compact-window convergence read to pass through F, K, E, Q, M, G, D before any public handoff.*
9. *H stores componentwise* hsame *transport for $F, K, E, Q, M, G, D, R, C, P, N$.*
10. *C stores displayed* Cont *replay from compact-open requests through the same finite convergence route.*

11. *P is the* Pkg *provenance over* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$, $\mathsf{CompactOpenFunctionSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{UniformEquicontinuityModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for host equality of functions, arbitrary compact-open topology construction, selected subnets, quotient maps, open-cover compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompactOpenUniformConvergenceUp.CompactOpenUniformConvergenceTasteGate`
`single carrier alignment`

Theorem 4235.2 (Compact-open uniform convergence NameCert obligations)**.** *For every accepted* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier*

$$U = (F, K, E, Q, M, G, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact-open function-space row, compact-metric probe row, equicontinuity row, uniform-equicontinuity modulus row, uniform-modulus threshold row, continuous-map graph row, metric and real target-distance row, compact-open replay row, componentwise transport, displayed continuation replay, provenance, and local naming.*

Every compact-open convergence consumer must read the compact-window probe K and the shared modulus rows Q, M before reading the target-distance ledger D or the handoff row R. No obligation in the packet is discharged by host function equality, selected subnet convergence, quotient map equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 4235.1. The tuple lists the compact-open source, compact probe, equicontinuity, uniform-modulus, continuous-map, metric, real, replay, transport, provenance, and naming rows.

The compact-window order is fixed by the visible route. A consumer first reads F through the compact probe K, then reads the equicontinuity row E and shared modulus rows Q, M before any target-distance comparison in D is available.

The replay row R packages that order for compact-open convergence consumers. The structural rows H, C, P, N preserve the same finite route by componentwise transport, continuation replay, provenance, and local naming.

Since the carrier contains no coordinate for the excluded host equality, subnet, quotient, open-cover, or completeness data, the listed rows exhaust the local NameCert obligation surface. □

Theorem 4235.3 (Compact-open uniform convergence finite-net handoff)**.** *Every compact-open topology or Arzela–Ascoli-style consumer of an accepted* $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactOpenFunctionSpace}^{\uparrow} \longrightarrow \mathsf{CompactMetric}^{\uparrow} \longrightarrow \mathsf{Equicontinuity}^{\uparrow} \longrightarrow \mathsf{UniformEquicontinuityModulus}^{\uparrow} \longrightarrow \mathsf{Uni}$$

The handoff exports only the displayed compact-window convergence surface, with the shared modulus and target-distance rows preserved by hsame, Cont, Pkg, *and local* NameCert *data.*

Proof. Begin with the NameCert obligations of Definition 4235.2. They force the compact-open consumer to read the compact-window probe, equicontinuity row, shared modulus rows, continuous-map graph row, and target-distance ledger before the replay row.

Next read the finite compact source through chapter 129 and the equicontinuity route through chapter 4076. These sibling surfaces supply finite probes and shared modulus data rather than open-cover compactness or selected subnets.

Finally R, H, C, P, N repack the displayed route for compact-open topology and function-space consumers. The carrier contains no host function equality, quotient map, selected subnet, or ambient completeness coordinate, so the public handoff is exactly the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$ packet is generated from compact-open function-space rows, compact-metric probe rows, equicontinuity rows, uniform-equicontinuity modulus rows, uniform-modulus thresholds, continuous-map graph rows, metric and Real target-distance ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4235.1, Definition 4235.2, and Definition 4235.3 seed the finite compact-open uniform convergence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOpenUniformConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, selected subnet convergence, quotient maps, arbitrary open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-window exposure, shared-modulus exactness, target-distance handoff, replay nonescape, and local naming for BEDC.Derived.CompactOpenUniformConvergenceUp.

Chapter 4236

A Concrete Naming Certificate for DefectFlowPartitionSkeleton$^{\uparrow}$

The DefectFlowPartitionSkeleton$^{\uparrow}$ packet realizes the defect-flow partition skeleton of Definition 9462.1722 as finite cubical-incidence and boundary-flow NameCert data. It sits over the partition carrier of chapter 201, the cellular-update surface of chapter 934, the Boolean row discipline of chapter 3733, the rank-dependence boundary of chapter 202, and the support-nerve surface of chapter 238. The packet names the finite route from partition blocks and boundary incidences to defect-lowering transitions; it is not a host graph-flow oracle, stochastic biological model, quotient codon semantics, unrestricted homology computation, or hidden global scheduler.

Definition 4236.1 (Defect-flow partition skeleton carrier)**.** *A* DefectFlowPartitionSkeleton$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, I, L, T, A, M, X, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite partition-block row, listing the displayed sink blocks and their support names.*
2. *I is the cubical boundary-incidence row, recording which displayed boundary faces meet each block.*
3. *L is the defect-level ledger, recording the finite defect value attached to each non-sink incidence state.*
4. *T is the defect-flow transition row, listing the allowed defect-lowering moves between displayed states.*
5. *A is the* CellularAutomaton$^{\uparrow}$ *local update row used to replay T as a finite boundary-flow step.*
6. *M is the* Matroid$^{\uparrow}$ *rank-dependence row separating removable shared incidences from protected private coordinates.*
7. *X is the* SimplicialComplex$^{\uparrow}$ *and* BooleanAlgebra$^{\uparrow}$ *support-nerve row, exposing faces, complements, and Boolean incidence tests.*
8. *H stores componentwise* hsame *transport for $B, I, L, T, A, M, X, C, P, N$.*
9. *C stores displayed* Cont *replay from a boundary-flow request through the same finite rows.*
10. *P is the* Pkg *provenance over* DefectFlowPartitionSkeleton$^{\uparrow}$, Partition$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, BooleanAlgebra$^{\uparrow}$, Matroid$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$ *row.*
The carrier has no coordinate for unrestricted flow search, host graph reachability, quotient codon semantics, biological stochastic semantics, hidden scheduler state, or unrestricted homology.

Theorem 4236.2 (Defect-flow partition skeleton NameCert obligations)**.** *For every accepted* $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ *carrier*

$$S = (B, I, L, T, A, M, X, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$ *surface consists exactly of carrier admission for partition blocks, cubical boundary-incidence rows, defect-level ledgers, defect-flow transitions, cellular replay, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise transport, displayed continuation replay, provenance, and local naming.*

A boundary-flow consumer may read a transition in T only after the source block row B, incidence row I, and defect ledger L have been displayed, and it may classify a sink only through the partition row and the support-nerve row. No obligation in the packet is discharged by host reachability, unrestricted flow search, biological stochastic semantics, quotient codon semantics, or hidden global scheduling.

Proof. Carrier admission is projection from Definition 4236.1. The displayed tuple lists the partition, incidence, defect, transition, cellular replay, rank-dependence, support-nerve, transport, continuation, provenance, and naming rows.

The transition order is fixed by the finite route. A consumer first reads the partition blocks B and cubical incidences I, then reads the defect ledger L, and only then may inspect the defect-lowering transition row T.

The cellular, matroid, Boolean, and simplicial rows explain how the same finite transition is replayed and classified. They provide local update, protected-coordinate separation, complement tests, and support-nerve exposure without importing a host flow oracle.

Finally H, C, P, N keep all reads inside the displayed packet. Since no excluded reachability, stochastic, quotient, scheduler, or unrestricted homology coordinate is present, the stated obligations exhaust the local NameCert surface. □

Theorem 4236.3 (Defect-flow partition skeleton TasteGate boundary)**.** *Any future TasteGate surface for* $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ *must provide* $\mathsf{Nontrivial\ DefectFlowPartitionSkeleton}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ DefectFlowPartitionSkeleton}^{\uparrow}$ *instances preserving $B, I, L, T, A, M, X, H, C, P, N$ as separate displayed fields. In particular, the gate must distinguish a changed partition block, a changed boundary-incidence row, a changed defect-flow transition, and a changed support-nerve row before it admits the packet as the same skeleton.*

Proof. Use the carrier of Definition 4236.1. The rows B, I, L, T, A, M, X play different roles in the finite boundary-flow route: source blocks, boundary incidences, defect values, transitions, cellular replay, rank separation, and support-nerve classification.

The NameCert obligations of Definition 4236.2 consume those rows in order. Identifying two of them would erase either the source, the flow step, or the classification boundary, so a field-faithful gate must preserve the displayed coordinates separately.

Nontriviality is witnessed by packets that vary one of the displayed rows while keeping the structural rows fixed. Such packets are observable at the carrier boundary and therefore cannot be collapsed by an admissible TasteGate read. □

Lean4 ✓ `BEDC.Derived.DefectFlowPartitionSkeletonUp.DefectFlowPartitionSkeletonTasteGate single carrier alignment`

Falsifiable prediction: Every accepted $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ read must expose partition blocks, cubical boundary-incidence rows, defect-level ledgers, defect-flow transitions, a cellular replay row, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a boundary-flow consumer reads a partition sink.

Independence witness: partition, cellularautomaton, booleanalgebra, matroid, simplicialcomplex: none alone is bijective with $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ because the packet jointly carries partition blocks, boundary incidences, defect transitions, update replay, rank separation, support-nerve tests, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}$.

Constructive story: A $\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}$ packet is generated from partition-block rows, cubical boundary-incidence rows, defect-level ledgers, defect-flow transition rows, cellular replay, matroid rank-dependence separation, Boolean and simplicial support-nerve tests, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4236.1, Definition 4236.2, and Definition 4236.3 seed the finite defect-flow partition skeleton packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DefectFlowPartitionSkeleton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted flow search, biological stochastic semantics, quotient codon semantics, unrestricted homology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition exactness, sink classification, Nontrivial DefectFlowPartitionSkeletonUp, and FieldFaithful DefectFlowPartitionSkeletonUp for BEDC.Derived.DefectFlowPartitionSkeletonUp.

The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet routes completion-facing reads through chapter 301, chapter 126, chapter 291, and chapter 38 before the choice-free completion body files are read.

Chapter 4237

A Concrete Naming Certificate for $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet records the choice-free regular-Cauchy route to a completion-facing real seal as finite BEDC data. It sits after the regular rational readback surface of chapter 301, the finite stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the separated comparison surface of chapter 1463, the public reflector route of chapter 1585, and the terminal real-name seal of chapter 38. The packet names a non-quotient completion route; it does not choose representatives, quotient regular sequences, assert ambient $\mathsf{Real}^{\uparrow}$ completeness, or introduce countable choice.

Definition 4237.1 (Regular-Cauchy choice-free completion carrier)**.** *A* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, carrying the regular rational name and its displayed readback requests.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite windows through which the regular name is inspected.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows and to the regular-Cauchy modulus requests.*
4. *U is the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row, recording separated comparison of two displayed completion routes without quotienting them.*
5. *F is the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row, exposing unit, counit, idempotent replay, universal factorization, and reflector transport as public finite data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, W, D, U, F have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, tolerance, comparison, reflector, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a regular-Cauchy request through $R, W, D, U, F, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $R, W, D, U, F, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completed objects, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4237.2 (Regular-Cauchy choice-free completion NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier*

$$K = (R, W, D, U, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *surface consists exactly of regular rational source exposure through R, finite window exposure through W, dyadic tolerance exposure through D, separated completion comparison through U, reflector routing through F, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4237.1. Rows R, W, D are the regular-Cauchy observation spine, and rows U, F, E are the completion-facing comparison, reflector, and real-seal data.

The observation spine is read before the reflector row is consumed. The $\mathsf{RegSeqRat}^{\uparrow}$ source is inspected only through $\mathsf{StreamName}^{\uparrow}$ windows, and the dyadic tolerance ledger fixes the finite precision route available to a completion consumer.

The comparison and reflector rows stay displayed. Row U records separated comparison of completion routes, while F exposes the public unit, counit, idempotent, extension, and reflector faces without adding a quotient carrier or a selected limit.

Finally H, C, P, N transport, replay, source, and name exactly the same packet. Since no excluded coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4237.3 (Regular-Cauchy choice-free real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exposes regular readback, finite windows, dyadic tolerance, separated comparison, and the public reflector surface before the terminal real seal is consumed. It does not close ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient regular sequences, selected representatives, countable choice, host equality of completed points, or open-cover compactness.*

Proof. Begin with the obligation surface of Definition 4237.2. It forces every real-facing consumer through the source, window, tolerance, comparison, reflector, and seal rows.

The sibling rows supply the route in order. The regular source is read through chapter 301; finite access is through chapter 126; dyadic tolerance is through chapter 291; separated comparison is through chapter 1463; and public reflection is through chapter 1585.

The terminal row E is therefore only the displayed $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Because the carrier has no quotient, selector, choice, host-equality, compactness, or ambient-completeness coordinate, none can be exported by the seal. □

This choice-free packet is the finite source route consumed by chapter 1425. Its $\mathsf{RegSeqRat}^{\uparrow}$ source row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{CompletionUniqueness}^{\uparrow}$ comparison, $\mathsf{CauchyCompletionReflector}^{\uparrow}$ row, and $\mathsf{Real}^{\uparrow}$ seal provide the source-side data for the universal extension and uniqueness ledger, while retaining the same non-quotient and no-choice boundary.

Theorem 4237.4 (Regular Cauchy choice-free completion StreamName anchor)**.** *Every accepted* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *completion read factors through the finite* $\mathsf{StreamName}^{\uparrow}$ *window row and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row before any terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed. The read is therefore anchored in chapter 126, chapter 301, and chapter 38, with W listing the inspected windows, R supplying the regular rational source rows, and E serving only as the downstream real-name seal.*

Proof. Project the carrier of Definition 4237.1. The rows R, W, E are displayed before any completion-facing consumer can read the real seal.

Apply the real-seal route theorem Definition 4237.3. It places $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ upstream of the $\mathsf{Real}^{\uparrow}$ terminal row and keeps the completion comparison and reflector rows downstream of the same finite observation surface.

Thus every accepted read factors first through the sibling StreamName and RegSeqRat anchors, and the Real row is only the terminal seal of that displayed route. □

Theorem 4237.5 (Regular-Cauchy choice-free cofinal-window refinement)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. If a completion-facing precision request for K is served by a displayed* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row, then the request refines through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before the real seal is consumed. The refinement preserves the StreamName request rows, DyadicRatCore tolerance rows, RegSeqRat classifier and regularity rows, the reflector row F, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *support.*

No quotient stream equality, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle can replace the displayed cofinal-window surface in this refinement.*

Proof. Begin with Definition 4237.2. It projects K onto the displayed source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and naming rows.

Use Definition 4060.2 for the precision request. The cofinality row exposes the finite $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance comparison, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the cofinal schedule before any completion-facing consumer can read the request.

Finally apply Definition 4237.3. The real-seal route places the reflector row F and the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same finite observation surface, so the cofinal-window refinement is consumed as displayed BEDC data and not as quotient streams, selected limits, countable choice, host equality, compactness, or ambient completeness. □

Theorem 4237.6 (Regular-Cauchy cofinal-window exactness)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every completion-facing precision request for* K *served by a* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *row is exact only through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The exactness surface is the finite StreamName window row, the DyadicRatCore tolerance row, the RegSeqRat source readback, the reflector row F*, and the terminal real seal* E*. It cannot be replaced by quotient stream equality, selected limits, countable-choice schedules, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier in Definition 4237.1. The only completion-facing rows available from K are the displayed source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows.

Apply Definition 4060.2 to the cofinal-window request. It exposes the finite window and tolerance data as BEDC rows before the request can be consumed by a completion-facing read.

Finally apply Definition 4237.3. Since the real seal is reached only after the same displayed route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$, the cofinal-window row is the unique completion-facing exactness surface available to the packet. □

Theorem 4237.7 (Regular-Cauchy choice-free window coverage)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Before a completion-facing precision request for* K *is read, the packet exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the displayed* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *cofinal-window rows that serve the request. These rows are the public finite window surface for the request, and no later reflector or real-seal read can bypass them.*

Proof. Project the carrier in Definition 4237.1. Its first three rows are exactly the regular-rational source, finite-window, and dyadic-tolerance rows available before the comparison, reflector, and real-seal rows.

Use Definition 4237.2 to identify those rows as part of the local NameCert surface. For a precision request served by a cofinal-window row, Definition 4237.5 and Definition 4237.6 place the $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ evidence on the same displayed route.

Finally apply the real-seal route theorem Definition 4237.3. Since the terminal seal is downstream of R, W, D and the cofinal-window evidence, every accepted precision read sees that finite window surface before any completion-facing seal is consumed. □

Theorem 4237.8 (Regular-Cauchy choice-free tail reflection)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every tail precision read of* K *reflects through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow},$$

preserving the packet rows $R, W, D, U, F, E, H, C, P, N$ *and any* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *support used to serve the request. The reflected read cannot be replaced by quotient stream equality, selected limits, countable-choice schedules, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4237.1. The tail precision read can inspect only the source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows.

Apply the NameCert obligations of Definition 4237.2. They bind the accepted packet to the rows $R, W, D, U, F, E, H, C, P, N$ and give no separate completion object.

Use Definition 4237.6 and Definition 4237.7. The cofinal support exposes the finite RegularCauchyWindowCofinal rows beside the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and $\mathsf{RegSeqRat}^{\uparrow}$ readback before the reflector and real seal are consumed.

Finally apply the real-seal route of Definition 4237.3. The reflected read reaches $\mathsf{CauchyCompletionReflector}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only downstream of that finite tail-window surface, so quotient stream equality, selected limits, countable choice, host equality, open-cover compactness, and ambient completeness cannot replace it. □

Theorem 4237.9 (Regular-Cauchy choice-free reflector nonescape)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. The* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *row* F *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E *export no quotient representative, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle. They only expose the displayed reflector route and the downstream real-name seal already named by the packet.*

Proof. Start with Definition 4237.2. The obligation surface lists F as a displayed reflector row and E as a terminal real-seal row, with no carrier coordinate for quotient representatives, selected limits, choice schedules, host equality, compactness, or ambient completeness.

The route theorem Definition 4237.3 keeps both rows downstream of the regular source, finite windows, dyadic tolerance, and separated comparison rows. Thus F can only route the displayed completion data, and E can only seal the resulting real-name boundary.

The cofinal-window exactness theorem Definition 4237.6 rules out replacing the displayed window surface by any excluded completion principle. Hence neither the reflector row nor the real-seal row exports data outside the carrier's finite route. □

Theorem 4237.10 (Regular-Cauchy choice-free metric-reflector handoff)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every metric-completion or reflector consumer of* K *must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row* R*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row* U*, the* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row* F*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row* N*.*

The handoff exports no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, open-cover compactness datum, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier and apply Definition 4237.2. The obligation surface names exactly the rows $R, W, D, U, F, E, H, C, P, N$, so a completion-facing consumer has no access to any coordinate outside the displayed packet.

Use the real-seal route of Definition 4237.3. It places the regular rational source, stream windows, dyadic tolerance ledger, separated comparison row, and reflector row upstream of the terminal $\mathsf{Real}^{\uparrow}$ seal.

For precision-sensitive reads, apply Definition 4237.6. The finite window and tolerance surface remains the exact completion-facing route before the reflector and real seal are consumed.

Finally apply Definition 4237.9. Since F and E export only the displayed reflector route and downstream real-name seal, the handoff refuses quotient completion, selected limits, countable choice, host equality, open-cover compactness, and ambient real completeness. □

Theorem 4237.11 (Regular-Cauchy choice-free reflector universal factorization)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Every completion-facing consumer of* K *factors first through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow}$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal is read. The factorization preserves the rows* $R, W, D, U, F, E, H, C, P, N$*, the cofinal-window support used by the request, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *naming.*

No completion-facing consumer may bypass this route by exporting quotient completion data, selected limits, countable-choice schedules, host equality of completed objects, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4237.1. The source, window, tolerance, comparison, reflector, seal, transport, replay, provenance, and local naming rows are exactly $R, W, D, U, F, E, H, C, P, N$.

Apply the real-seal route of Definition 4237.3. It orders every terminal $\mathsf{Real}^{\uparrow}$ read through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ before the seal row E is consumed.

Use the metric-reflector handoff of Definition 4237.10. It shows that metric-completion and reflector consumers receive the same displayed packet rows rather than a separate completion object.

Finally use Definition 4237.12. Componentwise transport, displayed replay, provenance, and local naming preserve the route instead of adding quotient, selector, choice, host-equality, compactness, or ambient-completeness data. □

Theorem 4237.12 (Regular-Cauchy choice-free transport and replay preservation)**.** *Let*

$$K = (R, W, D, U, F, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *carrier. Componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$ *row* N *preserve exactly the packet rows* R, W, D, U, F, E *together with their displayed cofinal-window support. They add no quotient completion principle, selected limit, countable-choice schedule, host equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier definition. Rows H, C, P, N are not new completion data; they are the transport, replay, provenance, and local naming rows attached to the displayed source, window, tolerance, comparison, reflector, and seal rows.

Apply Definition 4237.2. It says the local NameCert surface consists exactly of $R, W, D, U, F, E, H, C, P, N$, so transport and replay preserve the same packet rows rather than extending the classifier with an extra completion coordinate.

Use Definition 4237.3 and Definition 4237.7 for the downstream route. The same finite observation surface reaches the reflector and real seal after componentwise transport, displayed replay, provenance, and local naming, and the excluded completion principles never appear in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ read exposes RegSeqRat source, StreamName window, DyadicRatCore tolerance, CompletionUniqueness comparison, CauchyCompletionReflector route, RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing real seal is consumed.

Independence witness: regseqrat, streamname, dyadicratcore, completion-uniqueness, cauchy-completionreflector, real: none is bijective with $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ because this packet alone binds the regular-Cauchy source, finite-window tolerance spine, separated comparison row, public reflector route, and terminal RealUp seal without quotient completion or countable choice.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet is generated from RegSeqRat source rows, StreamName windows, DyadicRatCore tolerance rows, CompletionUniqueness comparison rows, CauchyCompletionReflector route rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4237.1, Definition 4237.2, and Definition 4237.3 seed the finite choice-free regular-Cauchy completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed objects, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, dyadic tolerance exactness, separated comparison, reflector routing, RealUp seal nonescape, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyChoiceFreeCompletionUp.

Completion-dense extension consumer. $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ supplies the choice-free $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route consumed by chapter 6382. The downstream extension packet reads this route as finite completion-facing data and receives no quotient completion representative, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 4237.13 (Regular-Cauchy choice-free dense-extension route)**.** *Every* $\mathsf{CompletionDenseExtension}^{\uparrow}$ *consumer of a* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet first reads the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows exposed by that packet. The consumer receives the route as finite completion-facing data and receives no quotient representative, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the NameCert obligations of Definition 4237.2. They expose the source row R, window row W, tolerance row D, comparison row U, reflector row F, terminal seal row E, and the transport, replay, provenance, and local naming rows before any downstream completion read is available.

Apply the real-seal route of Definition 4237.3. It orders the same displayed route through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CompletionUniqueness}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The downstream packet is chapter 6382. Since the upstream carrier contains no quotient representative, selected limit, countable-choice schedule, or ambient completeness coordinate, the dense-extension consumer can only receive the displayed finite rows. □

Theorem 4237.14 (Regular-Cauchy choice-free completion universal route)**.** *Every* RegularCauchyCompletionUnive *consumer of a* $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ *packet first reads the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{CompletionUniqueness}^{\uparrow}$ *comparison row,* $\mathsf{CauchyCompletionReflector}^{\uparrow}$ *route row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row before using the universal extension or uniqueness ledger. The universal consumer receives only those source-side finite rows together with the componentwise transport, displayed replay, provenance, and local naming rows of the choice-free packet.*

Proof. Start with the dense-extension route of Definition 4237.13. It exposes the regular source, finite windows, dyadic tolerance, completion comparison, reflector row, and real seal before any downstream completion-facing consumer can read the packet.

Use the real-seal route of Definition 4237.3. The terminal $\mathsf{Real}^{\uparrow}$ row is reached only after the same displayed source-side data, so the universal consumer cannot substitute quotient representatives, selected limits, choice schedules, or ambient completeness.

Finally apply the universal extension and public export surfaces, Definition 1425.2 and Definition 1425.10. They consume the displayed source, window, dyadic, comparison, and real-seal rows before exporting the extension route and uniqueness ledger, which is exactly the stated universal route from the choice-free packet. □

$\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ consumes the metric-completion dependency surface of chapter 1506 through the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletionReflector}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal route already bound by its carrier and real-seal theorems. The packet receives this route as finite completion-facing data and receives no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Theorem 4237.15 (Regular Cauchy choice-free sibling lattice route)**.** *Every accepted* RegularCauchyChoiceFreeCom *packet consumes chapter 1506 and chapter 1776 only through the displayed sibling lattice*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{CompletionUniqueness}^{\uparrow},\ \mathsf{CauchyCompletionReflector}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The route exports finite completion-facing data for the choice-free carrier, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected limit, choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the metric-completion sibling first. Its completion-facing surface is consumed only after the regular rational source row, stream-window row, and dyadic tolerance row have been displayed by the choice-free packet.

Read the Bishop completion induction sibling next. The induction route is attached to the same $\mathsf{CompletionUniqueness}^{\uparrow}$ comparison and $\mathsf{CauchyCompletionReflector}^{\uparrow}$ route before the terminal $\mathsf{Real}^{\uparrow}$ seal is exposed.

The carrier, real-seal, transport, replay, provenance, and local naming rows preserve exactly this sibling lattice. Since the packet contains no quotient-completion, selected-limit, choice-schedule,

host-equality, or ambient-completeness coordinate, no consumer can extract those endpoints from the route. □

Theorem 4237.16 (Regular Cauchy choice-free sibling route)**.** *Every accepted* RegularCauchyChoiceFreeCompletion *route consumes chapter 1506, chapter 1776,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *as sibling* NameCert *surfaces. The route exposes them only as finite completion-facing rows for regular rational source data, finite stream windows, dyadic tolerance accounting, induction replay, metric-completion dependency, and terminal real sealing. It exposes no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Read the sibling lattice of Definition 4237.15. The metric-completion and Bishop-induction siblings are admitted only after the regular rational, stream-window, and dyadic-tolerance surfaces have been displayed.

The route then reaches the terminal $\mathsf{Real}^{\uparrow}$ seal through the same completion-facing rows. Since no quotient-completion, selected-limit, choice-schedule, host-equality, or ambient-completeness coordinate is present in the carrier, the listed sibling surfaces exhaust the visible route. □

The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ route is consumed only after the L10 sibling surface has exposed regular rational families, finite stream windows, dyadic tolerance ledgers, and the terminal Real seal. A completion-facing consumer must therefore read chapter 301, chapter 126, chapter 291, and chapter 38 before using the choice-free handoff. The route records finite readback and tolerance data; it supplies no quotient completion, selected limit, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. The $\mathsf{RegularCauchyChoiceFreeCompletion}^{\uparrow}$ packet consumes chapter 301, chapter 38, and chapter 126 through finite regular-Cauchy windows before any completion-facing seal is read. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational source and readback requests, the $\mathsf{StreamName}^{\uparrow}$ row bounds those requests by displayed windows, and the $\mathsf{Real}^{\uparrow}$ row appears only as the terminal seal reached after the reflector and comparison rows have replayed the same finite data. This route gives downstream completion consumers no quotient representative, selected limit, countable-choice schedule, host equality of completed objects, or ambient real-completeness principle. $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads its local critical-pair and residual-join packet as a consumer of chapter 765, while the body files below keep the carrier, candidate-normalization, residual-substitution, and seed-readiness rows separate.

Chapter 4238

A Concrete Naming Certificate for MetaCICParallelDiamondFrontier$^{\uparrow}$

The MetaCICParallelDiamondFrontier$^{\uparrow}$ packet realizes the residual-substitution frontier of chapter 9378 as finite confluence-facing NameCert data. It sits beside the residual-substitution boundary of chapter 2181, the confluence critical-pair boundary of chapter 2212, the closed checker seal of chapter 2182, and the closed-normal confluence fragment of chapter 737. The packet records the conditional parallel-diamond frontier without claiming residual-substitution compatibility, full closed Church–Rosser, subject reduction, evaluator equality, or kernel consistency.

The retained P, K, J, R, S, C, B, O rows are consumed substantively by the MetaCIC critical-path packet of chapter 765, the confluence audit packet of chapter 735, and the normalization-confluence bridge route of chapter 1190. The critical-path reader consumes the residual-premise and obstruction rows, while the confluence audit packet consumes the candidate-join and bounded-check rows before any confluence consumer may use the frontier. The route exports no full Church–Rosser theorem and no subject-reduction theorem.

Falsifiable prediction: Every accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ confluence-facing read exposes the residual-substitution premise row before any parallel-diamond handoff can be consumed.

Independence witness: metacic-residual-substitution-boundary, metacic-confluence-critical-pair-boundary, closed-type-checker-seal, metacicconfluencenormalfragment: the residual-substitution boundary names the missing compatibility input without the parallel-diamond handoff; the critical-pair boundary records local peak data without the closed checker seal; the closed type-checker seal records bounded checking without confluence closure; and the closed-normal fragment records unconditional normal-source joins without the residual frontier. None is bijective to MetaCICParallelDiamondFrontier$^{\uparrow}$.

Definition 4238.1 (MetaCIC parallel-diamond frontier carrier)**.** *A* MetaCICParallelDiamondFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

with the following displayed rows:

1. *P is the residual-substitution premise row, naming the substitution compatibility input required by the parallel-diamond route.*
2. *K is the critical-pair row, recording the local peak and its two parallel-reduction observations.*

3. *J is the candidate-join row, recording the proposed common reduct and branch-local route requests.*
4. *R is the residual-frontier row, listing residual descendants, binder-depth exposure, and branch-local substitution windows.*
5. *S is the closed type-checker seal row consumed before a closed confluence-facing read can use the checker data.*
6. *C is the closed-normal fragment row, recording the already available normal-source and atom-source confluence fragments.*
7. *B is the bounded-check row over closedness, support, branch compatibility, and finished-normalization boundaries.*
8. *O is the obstruction row naming the endpoints not supplied by the packet: residual-substitution discharge, full parallel diamond, subject reduction, arbitrary closed Church–Rosser, evaluator equality, and kernel consistency.*
9. *H records componentwise* hsame *transport for premise, peak, join, residual, checker, fragment, bounded-check, obstruction, replay, provenance, and local naming rows.*
10. *T records displayed* Cont *replay from a confluence query through $P, K, J, R, S, C, B, O, H, G, N$.*
11. *G is the* Pkg *provenance over* MetaCICParallelDiamondFrontier$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceCriticalPairBoundary$^{\uparrow}$, ClosedTypeCheckerSeal$^{\uparrow}$, MetaCICConfluenceNormalFragment$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, K, J, R, S, C, B, O, H, T, G, N$. It has no coordinate for a premise-free parallel diamond, subject reduction, arbitrary closed Church–Rosser, unrestricted beta-conversion decidability, evaluator equality, quotient equality of terms, or kernel consistency.

Theorem 4238.2 (MetaCIC parallel-diamond frontier obligations)**.** *For every accepted* MetaCICParallelDiamondFro *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the local NameCert$_{\text{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *surface consists exactly of residual-substitution premise retention through P, critical-pair exposure through K, candidate-join exposure through J, residual-frontier exposure through R, closed-checker handoff through S, closed-normal fragment exposure through C, bounded-check exposure through B, obstruction retention through O, componentwise transport through H, displayed replay through T, provenance through G, and local naming through N.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObligations`

Proof. Project the carrier of Definition 4238.1. The confluence-facing rows are P, K, J, R, S, C, B, O, and the structural certificate rows are H, T, G, N.

The residual-substitution premise row remains first-class. Row P is not discharged by the carrier; it is the named input that the conditional parallel-diamond route must still expose.

The local geometry is then displayed through K, J, R. The critical-pair, candidate-join, and residual-frontier rows provide finite peak data and branch-local residual windows, while the closed checker and closed-normal rows show which existing fragments may be consumed.

Finally B, O, H, T, G, N keep the bounded checks, obstruction endpoints, transport, replay, provenance, and local naming attached to the same packet. Since no omitted theorem appears as a carrier coordinate, the listed rows exhaust the frontier obligations. □

Theorem 4238.3 (MetaCIC parallel-diamond frontier handoff)**.** *Every confluence consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the closed checker seal row S, the closed-normal fragment row C, and the bounded-check row B before any parallel-diamond handoff is read. The handoff may feed the conditional route recorded in chapter 9378, but it does not discharge residual substitution, prove subject reduction, prove arbitrary closed confluence, decide evaluator equality, or establish kernel consistency.*

Proof. Begin with the obligation surface of Definition 4238.2. It forces every confluence read through the named premise, peak, join, residual, checker, fragment, bounded-check, and obstruction rows.

The sibling packets preserve that boundary. The residual-substitution boundary supplies the named premise, the critical-pair boundary supplies local peak data, the closed type-checker seal supplies bounded checker data, and the normal fragment supplies only the closed-normal and atom-source cases already available.

The obstruction row O remains part of the packet. Therefore the handoff is a finite frontier certificate for a conditional route, not a proof of the missing residual-substitution input or any stronger confluence endpoint. □

Theorem 4238.4 (MetaCIC parallel-diamond critical-pair exhaustion)**.** *For every accepted* MetaCICParallelDiamor *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

each residual-substitution critical pair that a confluence consumer may inspect is exposed by the displayed critical-pair row K before the candidate-join row J can be read. The read must retain the residual-substitution premise row P, route the branch-local residual windows through R, pass the closed checker seal S and closed-normal fragment row C, and keep the bounded-check and obstruction rows B, O visible. It exports no unlisted peak, hidden substitution lemma, candidate join detached from K, R, or closed-checker result outside the displayed frontier packet.

Proof. Project the carrier of Definition 4238.1. The only rows that can mention residual substitution, critical pairs, candidate joins, residual windows, closed checking, and obstruction are P, K, J, R, S, C, B, O.

The residual-substitution premise is retained first. Row P is not discharged by the packet, so every critical-pair read remains conditional on the same displayed premise.

The critical-pair exposure is then forced through K. A consumer cannot read J as a candidate join until the peak and its two parallel-reduction observations have already been displayed in K and routed through the branch-local residual windows in R.

Finally the closed-checker handoff is bounded by S, C, B, O. The checker seal, closed-normal fragment, bounded-check row, and obstruction row remain visible through the replay, provenance, and local naming rows T, G, N, so the packet exhausts the accepted critical pairs without adding any hidden confluence endpoint. □

Theorem 4238.5 (MetaCIC parallel-diamond frontier obstruction triad)**.** *Every accepted* MetaCICParallelDiamond *packet retains three independent obstruction classes throughout the conditional local-diamond read:*

residual-substitution premise discharge remains confined to the row P, subject-reduction compatibility remains outside the closed-checker and closed-normal rows S, C, and the full closed Church–Rosser, evaluator equality, quotient term equality, and kernel-consistency endpoints remain recorded only as refused endpoints in the obstruction row O. These obstructions are transported and replayed through H, T, G, N without being collapsed into one another or promoted to completed confluence data.

Proof. Project the carrier rows P, O, B, S, C from Definition 4238.1. Row P is the residual-substitution premise, row O stores the refused endpoints, and rows B, S, C expose only bounded checking, closed checking, and closed-normal fragment data.

Apply the frontier handoff theorem and critical-pair exhaustion, Definition 4238.3 and Definition 4238.4. They keep the local peak under the displayed premise and force the closed-checker route to retain S, C, B, O.

Finally use the route theorem Definition 4238.9. Transport, replay, provenance, and local naming preserve the same rows rather than discharging P, deriving subject reduction from S, C, or turning the obstruction row into Church–Rosser, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.6 (MetaCIC critical-pair obstruction budget)**.** *Every critical-pair read consumed by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond frontier must preserve the residual-substitution premise row P and the obstruction row O through the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, a consumer may not retype the handoff as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the carrier rows of the accepted frontier packet, with P and O displayed as ordinary rows rather than external commentary.

Apply the route theorem Definition 4238.9. Any confluence-audit consumer must expose the critical-pair row K, candidate row J, residual-frontier row R, checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before using the conditional handoff.

Use premise retention, Definition 4238.13. The residual-substitution premise row P remains visible through the critical pair, residual frontier, closed checker, and closed-normal rows whenever the local-diamond handoff is consumed.

Finally apply discharge obstruction, Definition 4238.18. Since no separate residual-substitution compatibility certificate is supplied by this packet, the obstruction row O keeps the refused confluence, typing, evaluator, quotient, and kernel endpoints outside the critical-pair budget. □

Theorem 4238.7 (MetaCIC residual peak scope)**.** *Every candidate-mediated residual peak read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the MetaCIC critical-pair envelope, residual substitution dependency, normalization candidate scope, and closure-trace rows before a diamond frontier read is public. For a carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the visible scope is

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K records the critical-pair envelope, P retains the residual-substitution dependency, J, R, C carry the candidate-normalization and residual-frontier scope, and H, T, G, N preserve the closure trace through transport, replay, provenance, and local naming. The read is a finite frontier boundary; it is not a confluence theorem, a strong-normalization theorem, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. The residual peak has no row outside $P, K, J, R, S, C, B, O, H, T, G, N$ so every public diamond-frontier read must be assembled from that finite spine.

Use critical-pair exhaustion. It displays the local peak through K and prevents the candidate row J from being read before the residual-substitution premise P and branch-local residual row R remain visible.

Next apply the frontier handoff and residual-substitution scope routes already present in the chapter. These keep the candidate-normalization surface inside J, R, S, C, B, O and retain the obstruction row rather than closing the missing residual-substitution dependency.

Finally read the closure trace through H, T, G, N. Transport, replay, provenance, and local naming preserve the same candidate-mediated residual peak for downstream consumers, but none of those rows supplies confluence, strong normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.8 (MetaCIC parallel-diamond frontier seed-record exhaustion)**.** *Seed-facing confluence reads of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet are exhausted by*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

They cannot promote the residual-substitution premise, subject reduction, Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. Its seed-facing rows are exactly $P, K, J, R, S, C, B, O, H, T, G, N$.

Use the obligation surface and handoff theorem, Definition 4238.2 and Definition 4238.3. They keep the residual-substitution premise visible and expose only the conditional frontier route.

Finally cite premise retention, residual-substitution scope, and discharge obstruction. These retain P, scope the candidate frontier, and keep the refused confluence, typing, evaluator, quotient, and kernel endpoints outside the seed record. □

Theorem 4238.9 (MetaCIC parallel-diamond frontier route)**.** *Every confluence-audit consumer that reads an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker seal S, closed-normal fragment C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N before using a conditional parallel-diamond handoff. The route does not turn the residual-substitution premise into a theorem, does not prove full Church–Rosser, and does not discharge subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierRoute`

Proof. Begin with the carrier of Definition 4238.1. It displays exactly the premise, peak, join, residual, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

The obligation theorem Definition 4238.2 keeps P, K, J, R, S, C, B, O as first-class rows. A confluence-audit consumer therefore cannot use the handoff before the missing residual-substitution premise and the obstruction boundary are visible.

The handoff theorem Definition 4238.3 supplies only a conditional frontier route. The added transport, replay, provenance, and local name rows preserve that boundary for downstream audit consumers.

Since no row discharges residual substitution or full closed confluence, the route is a sharper obligation boundary rather than a completed Church–Rosser certificate. □

Theorem 4238.10 (MetaCIC local-diamond frontier)**.** *Let*

$$F_0 = (P, K, J_0, R, S, C, B, O, H, T, G, N) \quad \textit{and} \quad F_1 = (P, K, J_1, R, S, C, B, O, H, T, G, N)$$

be accepted $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packets with the same residual-substitution premise row P and the same critical-pair row K. If the candidate-join rows J_0 and J_1 agree through the displayed residual-frontier row R, then every confluence-audit consumer reads the same conditional local-diamond handoff through S, C, B, O, H, T, G, N. The obstruction row O remains visible in the handoff, so the result does not discharge residual-substitution compatibility, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the two carriers through Definition 4238.1. The hypotheses identify the premise and peak rows P, K, and they make the two candidate-join rows agree at the residual-frontier row R.

Apply Definition 4238.2 to each packet. The only confluence-facing rows available to a consumer before replay are P, K, J, R, S, C, B, O, with the structural rows H, T, G, N attached to the same packet.

Now use Definition 4238.3 and Definition 4238.9. Since P, K agree and J_0, J_1 agree through R, both handoff readings reach the same closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N.

The obstruction row is not removed by this comparison. It is carried in both packets and remains part of the consumer-visible handoff, so the theorem records a local-diamond frontier fact rather than a proof of the missing residual-substitution or closed confluence endpoints. □

Theorem 4238.11 (MetaCIC parallel-diamond residual square)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

exposes a residual square only through the retained residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the closed-checker seal S, the closed-normal fragment C, the bounded-check row B, the obstruction row O, and the transport, replay, provenance, and local naming rows H, T, G, N. Any consumer of the square must therefore factor through those rows before using a conditional local-diamond handoff. The square does not discharge residual substitution, subject reduction, full closed Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier through Definition 4238.1. Its confluence-facing rows are exactly P, K, J, R, S, C, B, O, and its structural rows are exactly H, T, G, N.

Use critical-pair exhaustion, Definition 4238.4. It places the local peak data inside the displayed critical-pair row K and keeps the closed-checker and obstruction rows visible.

Apply the local-diamond frontier theorem Definition 4238.10. A conditional local-diamond read can compare candidate joins only through the residual-frontier row R, with S, C, B, O, H, T, G, N still present in the handoff.

Finally route the resulting read through Definition 4238.9. Since that route retains $P, K, J, R, S, C, B, O, H, T, G$ before any consumer sees the handoff, the residual square is a frontier obligation over the displayed rows rather than any of the refused residual-substitution, typing, confluence, equality, quotient, or kernel endpoints. □

Theorem 4238.12 (MetaCIC parallel-diamond residual-substitution scope)**.** *Every use of the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *local-diamond frontier must pass through the residual-substitution boundary, the confluence critical-pair boundary, and the closed-normal fragment before it claims a local join. In carrier terms, a consumer must expose the residual-substitution premise row P, the critical-pair row K, the residual-frontier row R, the closed checker seal row S, and the closed-normal fragment row C before reading the candidate-join handoff. The resulting claim remains a local frontier claim; it does not close full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Apply the route theorem Definition 4238.9. It requires the residual-substitution premise row, the critical-pair row, the residual-frontier row, the closed-checker seal, and the closed-normal fragment to remain visible before the conditional handoff is consumed.

Then apply the local-diamond frontier theorem Definition 4238.10. The agreement of candidate-join rows is read only through the displayed residual-frontier row, and the obstruction row remains visible in the handoff.

The sibling boundaries named in the chapter introduction supply exactly the surrounding scope: the residual-substitution boundary names the missing compatibility premise, the confluence critical-pair boundary names the local peak surface, and the closed-normal fragment names the already available normal-source joins. Since none of these rows removes the obstruction row, the theorem records a residual-substitution-scoped local join rather than any stronger confluence endpoint. □

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualSubstitutio

Theorem 4238.13 (MetaCIC parallel-diamond frontier premise retention)**.** *Any accepted* MetaCICParallelDiamond *packet whose local-diamond handoff is consumed retains the residual-substitution premise row P through the critical-pair row K, residual-frontier row R, closed-checker row S, and closed-normal row C. No consumer may reinterpret that handoff as residual-substitution compatibility, premise-free parallel diamond, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Apply the obligations theorem Definition 4238.2. It makes P a first-class residual-substitution premise row and keeps K, R, S, C among the confluence-facing rows rather than structural metadata.

Next use the handoff and route theorems, Definition 4238.3 and Definition 4238.9. They require a consumer to expose the premise, critical-pair, residual-frontier, checker, and closed-normal rows before the conditional handoff is read.

Finally apply the residual-substitution scope theorem Definition 4238.12. Since the local-diamond frontier remains scoped by the residual-substitution boundary and the obstruction row is still visible, the retained handoff cannot be retyped as any of the refused confluence endpoints. □

Theorem 4238.14 (MetaCIC parallel-diamond closed-normal join socket)**.** *Any closed-normal join socket consumed by a MetaCIC confluence audit must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker seal S, closed-normal fragment C, bounded-check row B, obstruction row O, componentwise* hsame *support*

row H, displayed Cont *replay row T,* Pkg *provenance row G, and local* NameCert *row N before any local join is read. The socket retains the residual-substitution premise and obstruction boundary; it does not claim residual-substitution compatibility or full closed Church–Rosser.*

Proof. Start with the obligations and handoff theorems, Definition 4238.2 and Definition 4238.3. They list $P, K, J, R, S, C, B, O, H, T, G, N$ as the confluence-facing packet and require the handoff to stay inside those rows.

Use the route and local-diamond theorems, Definition 4238.9 and Definition 4238.10. These force any local join to pass through the critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, and structural rows before a confluence audit reads it.

Finally apply residual-substitution scope and premise retention, Definition 4238.12 and Definition 4238.13. The premise row P remains visible throughout the closed-normal socket, and O keeps the refused endpoints outside the handoff. □

Theorem 4238.15 (MetaCIC parallel-diamond frontier local-join ledger pullback)**.** *Let*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N)$$

be an accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet, and let*

$$L = (K, J, B, R, S, O, G_L, H_L, C_L, P_L, N_L)$$

be an accepted MetaCICLocalJoinLedger$^{\uparrow}$ *packet. Suppose the two packets share the same residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, transport, replay, provenance, and local naming rows, with the frontier rows R, S, C preserving the same residual-substitution scope as the ledger rows R, S. Then every confluence-audit consumer reads the local join through the same residual-substitution-scoped route. The pullback keeps the obstruction row visible, and it does not claim full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the frontier carrier through Definition 4238.1. The accepted packet exposes $P, K, J, R, S, C, B, O, H$, with P and O retained as confluence-facing rows rather than external annotations.

Read the local-join ledger through chapter 4126. The shared rows identify the local peak, candidate join, bounded checker, residual frontier, substitution boundary, obstruction readback, transport, replay, provenance, and local naming surfaces.

Apply the frontier obligations and route theorems, Definition 4238.2 and Definition 4238.9. They force any confluence-audit consumer to expose the premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, and structural rows before using a handoff.

Finally use premise retention, Definition 4238.13. Since the ledger rows are shared with the frontier rows and the obstruction row is preserved on both sides, the consumer-visible local join is read through the same residual-substitution-scoped route, not through an unconditional confluence endpoint. □

Theorem 4238.16 (MetaCIC parallel-diamond typed boundary)**.** *Any accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet whose conditional local-diamond handoff is read under a typed-window premise must retain the typed window, residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal row C, bounded-check row B, and obstruction row O before the handoff is consumed. The packet may not be cited as a premise-free parallel diamond, subject reduction, full Church–Rosser theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Apply the obligations theorem Definition 4238.2. It lists the premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, and obstruction rows as the confluence-facing surface of the accepted packet.

Use the route theorem Definition 4238.9. A typed-window consumer can read the conditional handoff only after those rows have been exposed together with transport, replay, provenance, and local naming.

Apply the residual-substitution scope theorem Definition 4238.12. The typed window is read through the same residual-substitution boundary, confluence critical-pair boundary, and closed-normal fragment that scope the local join.

Finally use premise retention, Definition 4238.13. The retained premise and obstruction rows prevent the handoff from being retyped as a premise-free diamond or any of the refused confluence, equality, or kernel endpoints. □

Theorem 4238.17 (MetaCIC parallel-diamond frontier typed-window handoff)**.** *Any typed-window consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must read the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before feeding candidate-normalization or confluence-audit consumers. In this route P remains the residual-substitution premise, O remains the obstruction row, and the handoff cannot be consumed as a premise-free parallel diamond, full Church–Rosser theorem, subject reduction, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Project the carrier of Definition 4238.1. The displayed packet contains the premise, critical-pair, candidate-join, residual-frontier, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Apply the obligations theorem and the typed boundary theorem, Definition 4238.2 and Definition 4238.16. Together they force a typed-window consumer to expose P, K, J, R, S, C, B, O before any conditional handoff is read.

Use premise retention and discharge obstruction, Definition 4238.13 and Definition 4238.18. The residual-substitution premise row P is still a premise, and the obstruction row O still blocks discharge unless a separate compatibility certificate is supplied.

Finally route the exported handoff through the structural rows H, T, G, N. Candidate-normalization and confluence-audit consumers therefore receive exactly the retained typed-window spine, not a completed parallel-diamond, Church–Rosser, subject-reduction, evaluator-equality, quotient-equality, or kernel-consistency endpoint. □

Theorem 4238.18 (MetaCIC parallel-diamond frontier discharge obstruction)**.** *Any accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet consumed as a local-diamond handoff still exposes the residual-substitution premise row P and obstruction row O. Unless a separate* MetaCICResidualSubstitutionCompatib *certificate is supplied through chapter 4370, no consumer may read that handoff as a premise-free parallel diamond, full closed confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the carrier of Definition 4238.1. The rows P and O are displayed coordinates of the accepted packet, not annotations around the handoff.

Apply premise retention, Definition 4238.13. It keeps P visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows whenever the local-diamond handoff is consumed.

Now use the sibling route theorem Definition 4238.21. The exported contribution is exactly retained premise and obstruction data for confluence-facing siblings, so the handoff cannot be consumed as a premise-free diamond.

The only admissible discharge surface is the residual-substitution compatibility chapter named in the statement. Without that separate certificate, the obstruction row O continues to block full closed confluence, subject reduction, evaluator equality, quotient term equality, and kernel consistency. □

Theorem 4238.19 (MetaCIC parallel-diamond frontier compatibility discharge surface)**.** *For an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet, only a separate* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *row may turn the retained P, O premise-obstruction surface into a discharged local diamond. Without that compatibility row, every confluence consumer retains the full conditional spine*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and may read the packet only as a conditional parallel-diamond frontier, not as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the discharge obstruction theorem, Definition 4238.18. It identifies P as the retained residual-substitution premise and O as the obstruction row that remains visible unless the separate compatibility chapter supplies a certificate.

Next use premise retention, Definition 4238.13. The premise row is not confined to the statement of the theorem; it is transported through the critical-pair, residual-frontier, closed-checker, and closed-normal rows of the accepted carrier.

Finally apply the confluence handoff, Definition 4238.22. That handoff exports the complete route $P, K, J, R, S, C, B, O, H, T, G, N$ to confluence-facing consumers. Since no row in this packet is the separate MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ certificate, the handoff remains conditional and the refused endpoints stay outside the frontier. □

Theorem 4238.20 (MetaCIC parallel-diamond closed-checker retention)**.** *Any accepted* MetaCICParallelDiamondFr *packet whose local-diamond handoff retains the residual-substitution premise row P must also retain the closed-checker seal row S and the closed-normal fragment row C before that handoff can be read. Thus the conditional handoff boundary is the joint row surface P, S, C together with the displayed critical-pair, residual-frontier, bounded-check, obstruction, transport, replay, provenance, and local naming rows. It is not a proof of residual substitution, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Apply the obligations theorem Definition 4238.2. It lists P, S, C as confluence-facing rows of the same accepted carrier, so the checker seal and closed-normal fragment are not optional metadata around the premise row.

Next apply the handoff and route theorems, Definition 4238.3 and Definition 4238.9. Both require the closed-checker seal and closed-normal fragment to be exposed before any conditional parallel-diamond handoff is consumed.

Finally use premise retention, Definition 4238.13. Since the retained premise row is read through K, R, S, C and the obstruction row remains visible, the closed-checker seal and closed-normal fragment remain part of the boundary rather than consequences of the handoff. The refused endpoints therefore stay outside the packet. □

Theorem 4238.21 (MetaCIC parallel-diamond frontier sibling route)**.** *Every accepted* MetaCICParallelDiamondFro *packet that exports its premise-retention handoff feeds two confluence-facing sibling routes: the*

candidate-normalization confluence handoff of chapter 5547 and the audit-packet boundary of chapter 735. The exported contribution is exactly the residual-substitution premise row P, the local critical-pair row K, the residual-frontier row R, and the obstruction row O retained by Definition 4238.13; it does not supply a premise-free diamond, a completed residual-substitution theorem, full closed confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Apply Definition 4238.13. Any consumed local-diamond handoff keeps the residual-substitution premise row visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows.

The candidate-normalization confluence handoff can therefore read this packet only as retained frontier data: P names the missing residual-substitution input, K and R name the local peak and residual window, and O keeps the obstruction endpoints visible before candidate-normalization routes are consumed.

The confluence audit packet reads the same rows as an audit boundary. It can cite the retained premise and obstruction rows when it records a conditional local-diamond route, but it cannot erase them into a completed confluence theorem.

Thus the sibling lattice is substantive rather than bibliographic: the frontier supplies exactly the residual-substitution premise retention and local-diamond obstruction rows consumed by those confluence-facing packets, while the refused endpoints remain outside the carrier. □

Theorem 4238.22 (MetaCIC parallel-diamond confluence handoff)**.** *An accepted* MetaCICParallelDiamondFrontier *packet whose local-diamond, premise-retention, and residual-substitution rows have been displayed exports exactly the conditional confluence handoff consumed by chapter 5547 and bounded by chapter 5126. In carrier terms, the handoff is the retained route through $P, K, J, R, S, C, B, O, H, T, G, N$: P remains the residual-substitution premise, K, J, R expose the local-diamond and residual windows, S, C, B keep the closed checker and bounded fragment visible, and O records the still-open endpoints. The handoff does not claim global confluence, full strong normalization, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the frontier carrier. The rows P, K, J, R, S, C, B, O are the confluence-facing data, while H, T, G, N transport, replay, source, and name the same packet.

Apply Definition 4238.13. The residual-substitution premise row P is retained through the local critical-pair and residual-frontier rows, so the exported route cannot be read as a premise-free diamond.

Next use Definition 4238.18. It keeps the obstruction row O visible unless a separate residual-substitution compatibility certificate is supplied, and therefore preserves the boundary checked by the parallel-diamond residual packet.

Finally apply the sibling route theorem. The candidate-normalization handoff receives only the closed checker boundary, local-diamond window, residual ledger, retained premise, and obstruction data displayed by this packet. Since no carrier row contains global confluence, full strong normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency, none of those endpoints is exported by the handoff. □

Theorem 4238.23 (MetaCIC parallel-diamond premise preservation)**.** *Any confluence consumer that reads the conditional parallel-diamond handoff of Definition 4238.22 must retain the residual-substitution premise row P through the full packet spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, the handoff cannot erase P into a completed residual-substitution theorem, premise-free parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Start with Definition 4238.2. It makes the residual-substitution premise row P part of the local $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ surface together with $K, J, R, S, C, B, O, H, T, G, N$.

Next apply premise retention, Definition 4238.13. The retained premise is still visible when the local-diamond handoff is consumed, and the discharge-obstruction theorem Definition 4238.18 keeps O displayed unless a separate compatibility certificate is supplied.

Finally use the confluence handoff theorem itself. Its exported route is exactly the conditional path through $P, K, J, R, S, C, B, O, H, T, G, N$, so any consumer that drops P or reads the route as one of the refused endpoints has left the packet described by the carrier. □

Theorem 4238.24 (MetaCIC parallel-diamond residual budget)**.** *Any exported* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond budget that feeds candidate-normalization or audit-packet consumers must consume the retained residual-substitution premise row P, critical-pair row K, residual-frontier row R, closed-checker row S, closed-normal row C, obstruction row O, and structural rows H, T, G, N before the consumer receives the handoff. The budget cannot omit P or O, and it cannot be read as a premise-free parallel diamond, residual-substitution theorem, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Start from the carrier obligations Definition 4238.2. They keep the residual-substitution premise, critical-pair, residual-frontier, closed-checker, closed-normal, obstruction, transport, replay, provenance, and naming rows inside the accepted packet.

Use the route and premise-retention theorems, Definition 4238.9 and Definition 4238.13. They require the premise row P to remain visible when a local-diamond handoff is consumed.

Next apply discharge obstruction and the sibling route, Definition 4238.18 and Definition 4238.21. The candidate-normalization and audit-packet consumers receive only the retained premise, critical-pair, residual-frontier, obstruction, and structural rows named by the frontier.

Finally use premise preservation Definition 4238.23. Since its full spine includes $P, K, J, R, S, C, B, O, H, T, G, N$ any exported budget that forgets P or O has left the displayed packet and cannot be consumed as a BEDC parallel-diamond frontier. □

Theorem 4238.25 (MetaCIC parallel-diamond frontier obstruction monotonicity)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. Any refinement of F that preserves the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ may strengthen the finite evidence attached to those rows, but it cannot weaken the obstruction row O or discharge the residual-substitution premise row P unless it also supplies a separate residual-substitution compatibility certificate. Consequently the refined packet remains a conditional frontier and cannot be consumed as full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. The premise row P and obstruction row O are carrier coordinates of the accepted packet, not side assumptions that a refinement may silently erase.

Apply premise retention and discharge obstruction, Definition 4238.13 and Definition 4238.18. They force the local-diamond handoff to keep P visible and keep O active unless a separate compatibility certificate supplies the missing residual-substitution discharge.

Finally use the residual budget theorem Definition 4238.24. A refinement that preserves $P, K, J, R, S, C, B, O, H, T, G, N$ preserves the same budget boundary, so it cannot convert retained obstruction data into a premise-free endpoint. The stronger claims listed in the statement therefore remain outside the packet. □

Theorem 4238.26 (MetaCIC parallel-diamond residual confluence)**.** *Every confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must factor its exported result through the full residual spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The exported result is conditional residual confluence with the residual-substitution premise retained; it is not a premise-free parallel diamond, subject reduction theorem, evaluator equality, quotient term equality, closed Church–Rosser theorem, or kernel-consistency certificate.

Proof. Start with the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the visible rows of any accepted frontier packet, with P retained as a residual-substitution premise.

Use the route theorem and premise retention, Definition 4238.9 and Definition 4238.13. Together they force confluence consumers to expose the premise, peak, join, residual, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local name rows before the handoff is consumed.

Apply discharge obstruction and the confluence handoff theorem, Definition 4238.18 and Definition 4238.22. These results keep the obstruction row visible and identify the exported route as the retained conditional handoff consumed by confluence-facing siblings.

Finally use the residual budget theorem Definition 4238.24. Since the budget cannot omit P or O, any exported confluence result remains conditional residual confluence over the displayed spine rather than one of the refused premise-free, typing, equality, quotient, Church–Rosser, or kernel endpoints. □

Theorem 4238.27 (MetaCIC residual-join readback)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*, every residual-join readback consumed by bounded conversion must display the closed substitution boundary rows, candidate-mediated strong-normalization rows, residual budget rows, typed-example boundary rows, and the four L10 source faces*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before reading the handoff. In carrier terms, the readback must expose $P, K, J, R, S, C, B, O, H, T, G, N$ *together with the typed-window and L10 source rows, and it cannot be consumed as full confluence, global beta decidability, closed subject reduction, quotient term equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with residual confluence, Definition 4238.26. It forces a confluence consumer to keep the residual-substitution premise, local critical pair, candidate join, residual frontier, checker seal, closed-normal fragment, bounded-check row, obstruction row, and structural rows visible through the full spine $P, K, J, R, S, C, B, O, H, T, G, N$.

Next use the typed-window handoff of Definition 4238.17. A bounded-conversion consumer may read the frontier only after the typed window has retained the same residual spine for candidate-normalization and confluence-audit consumers.

Premise preservation, Definition 4238.23, keeps the residual-substitution row P from disappearing into a completed diamond. The local-diamond and critical-pair routes of Definition 4238.10

and Definition 4238.9 supply the candidate join and residual budget rows without discharging the obstruction row O.

The typed-example boundary and the four L10 source faces are then read as source rows, not as semantic quotients. They expose $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the bounded-conversion handoff is consumed, so selected limits and ambient real completeness are unavailable.

Combining these routes gives a finite residual-join readback frontier. Since every step retains the premise and obstruction rows, the result is not full confluence, global beta decidability, closed subject reduction, quotient term equality, selected limits, or $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4238.28 (MetaCIC critical-overlap residual budget)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*, every critical overlap between two local reductions is budgeted by the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The overlap budget must expose the residual-substitution premise row P*, the critical-pair row* K*, the candidate-join row* J*, the residual-frontier row* R*, the closed checker seal* S*, the closed-normal fragment* C*, the bounded-check row* B*, and the obstruction row* O *before any local join is read. It does not supply source-equivalence transport, abstract carrier equivalence, premise-free parallel diamond, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. It makes the premise, critical-pair, candidate-join, residual-frontier, checker, fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows the finite carrier of an accepted frontier packet.

Use the residual-substitution scope theorem Definition 4238.12. A local join may be read only after the residual-substitution boundary, critical-pair boundary, and closed-normal fragment have remained visible.

Now apply premise preservation and the residual budget theorem, Definition 4238.23 and Definition 4238.24. These keep P and O inside the exported budget and prevent a critical overlap from being consumed as a premise-free diamond.

Finally use residual confluence, Definition 4238.26. Since the confluence consumer factors through the same displayed spine, the critical-overlap budget is finite and row-local rather than a transport across source equivalence or abstract carrier equivalence. □

Theorem 4238.29 (MetaCIC candidate-mediated residual route)**.** *Let two local reductions be recorded by the critical-pair row* K *of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$*. If the candidate-join row* J *is accepted through the displayed residual-frontier row* R*, then both local reductions route to the shared residual packet*

$$J, R, S, C, B, O, H, T, G, N$$

under the retained premise P*. The route is candidate-mediated: it identifies the common residual packet available to the frontier consumer, but it does not discharge residual-substitution compatibility, source-equivalence transport, subject reduction, full closed Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the carrier in Definition 4238.1. The critical-pair row K records the two local reduction observations, and the candidate-join row J records the proposed common reduct together with branch-local route requests.

Apply the obligations theorem Definition 4238.2. It keeps J and R as displayed rows of the same packet, so the candidate route is not an external equivalence between carriers.

Use the residual-substitution scope theorem and premise preservation, Definition 4238.12 and Definition 4238.23. The two local reductions may reach the candidate only while P remains visible and while the residual-frontier row controls the branch windows.

Finally apply residual confluence and residual-join readback, Definition 4238.26 and Definition 4238.27. They identify the consumer-facing route as the shared residual packet through $J, R, S, C, B, O, H, T, G, N$, with the premise and obstruction still retained. □

Definition 4238.30 (MetaCIC parallel-diamond frontier finite budget)**.** *The finite budget for a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *read is the row surface*

$$\mathcal{B}_{\text{frontier}} = (W, K, P, J, R, S, C, B, O, H, T, G, N).$$

Here W is the typed-window row, K is the critical-pair row, P is the residual-substitution boundary, J is the local-join row, R is the residual-frontier row, S is the closed-checker seal, C is the closed-normal fragment, B is the bounded-checker row, O is the obstruction row, H is componentwise hsame *transport, T is displayed* Cont *replay, G is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row. The budget has no coordinate for premise-free confluence, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Theorem 4238.31 (MetaCIC typed-window local-join scope)**.** *Under a typed MetaCIC window, an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet permits only local joins whose two branches pass through the retained residual-substitution premise P, critical-pair row K, candidate-join row J, residual-frontier row R, closed checker seal S, closed-normal fragment C, and bounded-check row B. The local join scope is therefore the typed-window row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with O still visible. It is not a source-equivalence transport theorem, an abstract carrier-equivalence theorem, a premise-free parallel diamond, subject reduction, global beta decidability, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the typed boundary and typed-window handoff, Definition 4238.16 and Definition 4238.17. They force a typed-window consumer to expose the same residual spine before candidate-normalization or confluence-audit consumers can read a handoff.

Use the critical-overlap budget of Definition 4238.28. It keeps every local overlap inside $P, K, J, R, S, C, B, O, H, T, G, N$ and blocks readings that omit the premise or obstruction rows.

Next apply the candidate-mediated route Definition 4238.29. The two branches can meet only through the displayed candidate row and residual-frontier row, not through an ambient source-equivalence or carrier-equivalence transport.

Finally use residual-join readback, Definition 4238.27. The readback keeps bounded conversion, typed windows, and the L10 source faces visible before the handoff, so the permitted local join remains a scoped MetaCIC window claim rather than any stronger confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 4238.32 (MetaCIC parallel-diamond frontier local join budget)**.** *Any local join read accepted by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet consumes the finite budget of Definition 4238.30*

before it can reach a normalization-facing consumer. In particular, the typed-window row, critical-pair row, residual-substitution boundary, local-join row, residual-frontier row, closed-checker seal, bounded-checker row, obstruction row, transport, replay, provenance, and local naming remain visible at the handoff.

Proof. Start with the typed-window local-join scope of Definition 4238.31. It forces any accepted local join to expose the residual-substitution premise, critical pair, candidate join, residual-frontier row, checker rows, and obstruction row.

Apply the finite budget definition Definition 4238.30. The typed-window row and structural rows H, T, G, N are part of the same budget surface, so the local join cannot be consumed as a free-standing confluence fact.

Use the residual budget theorem Definition 4238.24. It keeps the obstruction and residual-substitution rows visible before normalization consumers read the packet. Therefore the handoff remains a finite typed-window claim and does not become premise-free normalization or global confluence. □

Theorem 4238.33 (MetaCIC parallel-diamond frontier residual boundary)**.** *Residual-substitution compatibility remains a visible boundary in every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *local-join read. The row P of Definition 4238.30 is consumed before J, R, S, C, B can be used, and it is not erased into an unconditional confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, global normalization statement, or kernel-consistency certificate.*

Proof. Project the budget surface of Definition 4238.30. The residual-substitution boundary P is an explicit coordinate, not a proof obligation hidden behind the local-join row.

Apply premise preservation and the residual-substitution scope theorem, Definition 4238.23 and Definition 4238.12. These keep P visible whenever the critical-pair, residual-frontier, checker, and bounded-check rows are read.

Finally use the local join budget theorem Definition 4238.32. Since every normalization-facing read consumes the full finite budget first, no consumer can erase the residual boundary and present the packet as an unconditional confluence or kernel result. □

Theorem 4238.34 (MetaCIC parallel-diamond frontier obligation basis)**.** *Every accepted* MetaCICParallelDiamondF *packet presents a finite obligation basis for the typed-window frontier. The basis is the row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P is the residual-substitution boundary, K is the critical-pair envelope, J is the local-join row, R is the residual-frontier row, S is the closed checker seal, C is the closed-normal fragment, B is the bounded-check row, O is the obstruction row, H is hsame *transport, T is* Cont *replay, G is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row. Any confluence consumer must read this basis before it may cite the typed-window local join, and the basis does not discharge residual substitution, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with the residual budget theorem Definition 4238.24. It keeps the retained premise, critical-pair row, residual-frontier row, closed-checker row, closed-normal row, obstruction row, and structural rows visible before any exported budget is consumed.

Apply residual confluence and residual-join readback, Definition 4238.26 and Definition 4238.27. These route confluence-facing reads through the same residual spine and require bounded-conversion readback to keep the typed-window and L10 source rows visible.

Use the critical-overlap budget and candidate-mediated residual route, Definition 4238.28 and Definition 4238.29. The local peak may reach a candidate only through the displayed K, J, R rows, with P and O retained.

Finally apply the typed-window local-join scope theorem Definition 4238.31. It identifies the permitted local join as exactly the typed-window row surface, so the listed rows form the finite obligation basis and no stronger confluence endpoint is exported. □

Theorem 4238.35 (MetaCIC parallel-diamond frontier seed status)**.** *Every conditional parallel-diamond handoff consumed through an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must first expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, bounded-check row B, obstruction row O, componentwise* hsame *transport row H, displayed* Cont *replay row T,* Pkg *provenance row G, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row N. The consumed handoff is therefore a seed-status frontier: it records which residual premises and finite checks remain visible before use, and it is not a parameter-transport equivalence, source-equivalence theorem, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Begin with the obligations theorem Definition 4238.2. It makes the residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and naming rows part of the finite carrier surface.

Apply the obligation-basis theorem Definition 4238.34. It packages those rows as the typed-window basis that a confluence consumer must read before citing a local join.

The residual-frontier and obstruction rows stay visible throughout the handoff. Consequently the theorem records seed status for the remaining residual-discharge work rather than converting the packet into a parameter-transport equivalence or any stronger confluence endpoint. □

Theorem 4238.36 (MetaCIC parallel-diamond residual obstruction ledger)**.** *Every candidate-mediated local join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise row P, residual-frontier row R, and obstruction row O visible alongside*

$$K, J, S, C, B, H, T, G, N$$

before any confluence-facing packet consumes the read. The retained ledger is therefore

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It is not a premise-free parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. Begin with the obligation basis Definition 4238.34. It fixes the consumer-visible packet as $P, K, J, R, S, C, B, O, H, T, G, N$ and makes the obstruction row an explicit coordinate of the finite basis.

Apply the candidate-mediated residual route Definition 4238.29. A local peak may reach the candidate row J only through the displayed critical-pair and residual-frontier route, while P remains the retained residual-substitution premise.

Finally use the local-join scope theorem Definition 4238.31. It allows confluence-facing consumers to read only the typed-window surface with O still visible, so the obstruction ledger cannot be erased before the handoff is consumed.

Thus every candidate-mediated join read carries P, R, O together with $K, J, S, C, B, H, T, G, N$. Since neither the premise nor the obstruction is discharged, the statement exports only a residual

obstruction ledger and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 4238.37 (MetaCIC residual pasting frontier)**.** *Suppose an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet has two typed-window residual reads whose critical-pair observations share the same* $P, K, R, S, C, B, O, H, T,$ *rows and whose candidate row is* J*. Then the two reads paste only through the displayed residual packet*

$$J, R, S, C, B, O, H, T, G, N$$

under the retained premise P*. The pasted read is a finite* NameCert *route with* hsame *transport,* Cont *replay, and* Pkg *provenance; it is not a source-equivalence transport theorem, abstract carrier equivalence, premise-free parallel diamond, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.*

Proof. Begin with Definition 4238.34. It fixes the consumer-visible basis as $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the obstruction row visible.

Use candidate-mediated residual routing, Definition 4238.29. The two local reductions may meet only at the displayed candidate row J and residual-frontier row R, while the residual-substitution premise P remains retained.

Apply residual confluence and residual-join readback. These theorems keep the closed checker, closed-normal fragment, bounded-check row, obstruction, transport, replay, provenance, and local naming attached to the same packet before any bounded-conversion consumer can read the handoff.

The pasted route is therefore the shared residual packet $J, R, S, C, B, O, H, T, G, N$ under P. Since the construction never introduces a source-equivalence carrier or removes O, the conclusion is a finite frontier pasting claim and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 4238.38 (MetaCIC substitution nonescape frontier)**.** *In every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, substitution data remains confined to the residual-substitution boundary row* P *and the residual-frontier row* R *throughout the typed-window local join. No* hsame *transport row,* Cont *replay row,* Pkg *provenance row, or local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row may reinterpret the handoff as a discharged residual-substitution theorem, premise-free parallel diamond, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start from the obligation basis Definition 4238.34. It makes P and R explicit coordinates of the finite packet rather than implicit assumptions around the local join.

Apply the residual-substitution scope theorem and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the substitution premise visible through the critical-pair, residual-frontier, closed-checker, and closed-normal rows.

Use residual pasting, Definition 4238.37. Even when two typed-window residual reads are pasted through the candidate row, the pasted route still runs under P and through R.

Finally read the structural rows H, T, G, N. They transport, replay, source, and name the displayed packet, but they do not contain a carrier slot for discharging substitution or erasing the obstruction row. Thus substitution cannot escape the displayed residual boundary into any stronger confluence endpoint. □

Theorem 4238.39 (MetaCIC parallel-diamond residual-substitution route)**.** *Every candidate-mediated local join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T,$ *must expose the residual-substitution premise row* P*, critical-pair row* K*, candidate-join row* J*,*

residual-frontier row R, closed-checker seal row S, closed-normal fragment row C, bounded-check row B, obstruction row O, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming before any parallel-diamond frontier handoff is consumed. The route is conditional on the displayed premise and cannot be read as residual-substitution compatibility, full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Project the carrier of Definition 4238.1. The rows $P, K, J, R, S, C, B, O, H, T, G, N$ are the full consumer-visible packet, with P retaining the residual-substitution premise and O retaining the refused endpoints.

Apply the candidate-mediated residual route of Definition 4238.29. It allows the two local reductions named by K to meet only through the displayed candidate row J and residual-frontier row R, under the retained premise P.

Next use substitution nonescape, Definition 4238.38. The closed checker row, closed-normal row, bounded-check row, transport, replay, provenance, and local naming cannot discharge substitution or remove the obstruction row.

Thus the route exposes exactly the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, and structural rows stated above. Since neither P nor O disappears, the handoff remains a conditional frontier and not any refused confluence, typing, equality, quotient, or kernel endpoint. □

Theorem 4238.40 (MetaCIC parallel-diamond normalization-frontier handoff)**.** *Any normalization consumer that reads the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may use only the conditional frontier exposed by Definition 4238.39. It may inspect the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming. It may not consume the packet as full Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, kernel consistency, or a completed residual-substitution theorem.*

Proof. Begin with the residual-substitution route theorem. It fixes the normalization-facing read to the displayed finite spine $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the obstruction row visible.

Use the typed-window local-join scope of Definition 4238.31. A normalization consumer may read local joins only under that typed-window surface, so it cannot erase the residual-frontier or closed-checker conditions.

Apply residual pasting and substitution nonescape, Definition 4238.37 and Definition 4238.38. These theorems keep pasted residual reads under P and prevent structural rows from retyping the handoff as a discharged substitution result.

Consequently the normalization consumer receives only the conditional frontier. The listed stronger endpoints require separate certificates and are not carried by the accepted MetaCICParallelDiamondFrontier$^{\uparrow}$ packet. □

Theorem 4238.41 (MetaCIC parallel-diamond candidate-mediated strong-normalization socket)**.** *For every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *carrier $F = (P, K, J, R, S, C, B, O, H, T, G, N)$, suppose the closed checker row S accepts both residual branches named by the local critical-pair row K, the candidate-normalization scope row J retains the residual-substitution premise row P, and the local join ledger read through R, C, B, H, T, G, N is available. Then the packet exposes a candidate-mediated strong-normalization socket for that local pair: its readback is the conditional route $P, K, J, R, S, C, B, O, H, T, G, N$ consumed by Definition 4238.40. The socket does not assert*

global strong normalization, full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start from the critical-pair envelope of Definition 4238.4. It fixes the two residual branches inside the displayed local pair row K and keeps them tied to the closed checker row S.

Use the residual-substitution dependency route of Definition 4238.12. The premise row P remains an explicit coordinate of the candidate-normalization scope instead of becoming a discharged theorem.

Next apply the closed-normal join socket and local join ledger pullback, Definition 4238.14 and Definition 4238.31. These results route the accepted branches through J, R, C, B while preserving H, T, G, N as transport, replay, provenance, and naming rows.

Finally hand the same finite readback to Definition 4238.40. Since P and O remain visible, the socket is only the candidate-mediated local normalization route promised in the statement, not any global normalization or confluence endpoint. □

Theorem 4238.42 (MetaCIC parallel-diamond frontier typed-window candidate SN route)**.** *Every strong-normalization-facing read of the candidate-mediated socket in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *must pass through the typed-window handoff, residual-substitution scope, premise preservation, local-join scope, and the candidate-mediated strong-normalization socket before it reaches the normalization-frontier handoff. The resulting route is the finite readback*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with the typed window still visible. It does not assert global strong normalization, full confluence, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with the typed-window handoff of Definition 4238.17. It forces every candidate-normalization consumer to expose the same residual spine before any frontier handoff is read.

Next apply residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. These keep P attached to the critical-pair, residual-frontier, closed-checker, and closed-normal rows rather than turning it into a discharged substitution theorem.

Use the typed-window local-join scope of Definition 4238.31. A candidate join can be consumed only through the displayed K, J, R, S, C, B rows, with the obstruction row O still retained.

Finally apply the candidate-mediated strong-normalization socket and normalization-frontier handoff, Definition 4238.41 and Definition 4238.40. They pass the same finite readback forward as a conditional socket, so the route cannot be retyped as any of the refused normalization, confluence, typing, equality, quotient, or kernel endpoints. □

Theorem 4238.43 (MetaCIC parallel-diamond closed-substitution window scope)**.** *Every candidate-mediated diamond read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier* $F = (P, K, J, R, S, C, B, O, H, T, G,$ *that exposes the rows* $P, K, J, R, S, C, B, O, H, T, G, N$ *must keep the closed-substitution window visible before the closed checker seal and normalization-confluence handoff are consumed. The read therefore remains a finite frontier route through the displayed residual-substitution premise, typed candidate window, closed checker, local join ledger, obstruction row, transport, replay, provenance, and local naming rows. It cannot be reread as premise-free subject reduction or as closed Church–Rosser.*

Proof. Project the carrier and begin with the premise-retention route of Definition 4238.23. The residual-substitution premise row P remains visible whenever the candidate rows K, J, R, S, C, B, O are exposed.

Apply the typed-window candidate SN route of Definition 4238.42. It sends every strong-normalization-facing read through the typed window and candidate-mediated socket before the normalization-frontier handoff is available.

Finally use the critical-path retained-premise handoff of Definition 4238.85. The structural rows H, T, G, N transport, replay, source, and name the same retained-premise packet, so the closed checker seal and normalization-confluence handoff cannot erase the closed-substitution window. Hence the route is not a premise-free subject-reduction theorem and not closed Church–Rosser. □

Theorem 4238.44 (MetaCIC parallel-diamond finite candidate SN extension)**.** *Let $F = (P, K, J, R, S, C, B, O, H,$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier. If a finite local peak is routed through the displayed candidate row J, residual row R, typed window S, and boundary rows C, B, O, then the only strong-normalization-facing read is the candidate-mediated socket already named by Definition 4238.41. The route cannot be retyped as global strong normalization, full confluence, quotient equality of terms, or a kernel endpoint.*

Proof. Apply the candidate-mediated residual route Definition 4238.29. It sends the finite local peak through the displayed candidate row J and residual row R while retaining the residual-substitution premise row P.

Use the local-join scope Definition 4238.31. The local join is visible only through the typed-window and boundary rows S, C, B, O, so no ambient confluence carrier is introduced.

Next apply the normalization-frontier handoff Definition 4238.40. It restricts normalization consumers to the conditional frontier packet and keeps the refused endpoints outside the carrier.

Finally use the candidate-mediated strong-normalization socket and typed-window candidate SN route, Definition 4238.41 and Definition 4238.42. These results identify the only SN-facing read as the finite candidate-mediated socket through $P, K, J, R, S, C, B, O, H, T, G, N$, so it cannot be reread as global strong normalization, full confluence, quotient equality, or a kernel endpoint. □

Theorem 4238.45 (MetaCIC parallel-diamond frontier seed closure status)**.** *The seed closure surface of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet is exactly the finite conjunction of residual-substitution premise retention, critical-pair envelope exposure, closed-checker retention, closed-normal join socket, candidate-mediated strong-normalization socket, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The seed surface does not discharge residual-substitution compatibility, premise-free parallel diamond, subject reduction, full closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start with residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the row P visible through every local-diamond read.

Use the critical-pair envelope and closed-checker retention, Definition 4238.4 and Definition 4238.20. These identify the rows K, S, C, B, O as part of the finite frontier instead of consequences of a discharged confluence theorem.

Apply the local-join and normalization handoff theorems, Definition 4238.31 and Definition 4238.40. Together with the candidate-mediated strong-normalization socket Definition 4238.41, they show that the only normalization-facing read is the conditional route through P, K, J, R, S, C, B, O.

Finally read the structural rows H, T, G, N. They transport, replay, source, and name the same finite packet, so the seed closure surface is exactly the listed conjunction and none of the refused endpoints is exported. □

Theorem 4238.46 (MetaCIC parallel-diamond residual-join induction)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier. Finite iteration over a displayed list of*

residual pairs preserves the local-diamond route through the critical-pair row K, residual-substitution premise row P, candidate-normalization row J, closed-checker row S, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N. The iterated route remains candidate-mediated and exports no premise-free parallel diamond, subject reduction, full closed Church–Rosser theorem, evaluator equality, quotient equality of terms, or kernel consistency endpoint.

Proof. Induct on the finite residual-pair list. The base case is Definition 4238.45: seed readiness already exposes residual-substitution premise retention, critical-pair rows, candidate joins, closed checker rows, bounded checks, obstruction retention, and the structural rows.

For the step, append one displayed residual pair. The residual-substitution dependency route of Definition 4238.12 keeps P visible, while candidate-normalization scope and finite residual witness consumption keep the new pair inside K, J, R, S, C, B, O before any normalization-facing handoff is read.

Finally apply the critical-path consumer route of Definition 4238.85. It passes the extended finite readback through H, T, G, N without changing the refused endpoints, so the induction preserves the local-diamond route claimed above. □

Theorem 4238.47 (MetaCIC parallel-diamond public export)**.** *For an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier $F = (P, K, J, R, S, C, B, O, H, T, G, N)$, the public confluence-facing export is exactly the route through residual-substitution premise P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N. The export may feed MetaCIC critical-path and confluence-audit consumers, but it does not prove full Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency.*

Proof. Begin with the obligation surface of Definition 4238.2. It admits the carrier only through the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Use critical-pair exhaustion, obstruction retention, and local-join scope: Definition 4238.4, Definition 4238.5, and Definition 4238.31. These keep the critical pair, candidate join, residual frontier, bounded check, and obstruction rows visible before any confluence-facing read.

Next apply the normalization handoff and candidate-mediated socket, Definition 4238.40 and Definition 4238.41. They route normalization-facing consumers through the same finite packet without exporting global normalization or confluence.

Finally invoke residual-join induction, Definition 4238.46. The structural rows H, T, G, N transport, replay, source, and name the same packet, so the public export is the listed route and none of the refused endpoints follows. □

The Lean follow-up supplies Nontrivial and FieldFaithful TasteGate instances in `lean4/BEDC/Derived/MetaCICP` by exhibiting distinct accepted and refused frontier packets and by matching on the rows $P, K, J, R, S, C, B, O, H, T,$

Theorem 4238.48 (MetaCIC parallel-diamond scoped closure status)**.** *The seed-readiness surface, retained-premise handoff, obstruction row, residual-substitution premise, critical-path consumer route, and normalization-confluence handoff of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet together scope the chapter to finite candidate-mediated local-diamond evidence. The scoped surface is exactly the displayed route through $P, K, J, R, S, C, B, O, H, T, G, N$ and does not assert subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Read the seed surface through Definition 4238.83 and Definition 4238.45. These results expose the carrier rows and keep the finite local-diamond evidence inside the displayed packet.

Apply the retained-premise handoff and residual-substitution scope, Definition 4238.85 and Definition 4238.12. They keep the premise row P visible for every critical-path read instead of discharging it as a substitution theorem.

Finally use the obstruction, public export, and normalization-frontier handoff rows of Definition 4238.5, Definition 4238.47, and Definition 4238.40. The resulting scope is finite, candidate-mediated, and local to $P, K, J, R, S, C, B, O, H, T, G, N$, so none of the refused global endpoints is exported. □

Theorem 4238.49 (MetaCIC parallel diamond critical-pair envelope)**.** *Every accepted* MetaCICParallelDiamondFro *packet exposes a critical-pair envelope only after reading a residual substitution row, a local confluence overlap row, a closed-term substitution row, and a binder-budget row. In carrier terms, the envelope is the finite BEDC route through*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with the residual-substitution and confluence audit-map rows still visible. The conclusion is a concrete frontier envelope for bounded MetaCIC confluence audit; it is not a general strong-normalization theorem, unrestricted confluence theorem, unrestricted beta-reduction theorem, or host equality principle.

Proof. Project the accepted frontier carrier. The residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N are the only rows exposed by the packet.

Read the residual-substitution route first. The row P supplies the retained premise, while the residual-frontier row R records branch-local substitution windows and binder exposure; neither row discharges compatibility.

Next read the confluence audit-map route. The critical-pair and candidate-join rows K, J supply local overlap data, while S, C, B keep the closed-term substitution and bounded checker information attached to the same finite packet.

The obstruction and structural rows close the envelope. Since O, H, T, G, N preserve the retained premise, bounded replay, provenance, and local name, the packet cannot escape to unrestricted beta reduction, host equality, strong normalization, or a general confluence theorem. □

Theorem 4238.50 (MetaCIC parallel diamond frontier critical-pair envelope)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet exposes critical-pair requests only through the finite frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The candidate term rows K, J, residual-substitution rows P, R, finite observation confluence rows S, C, B, normalization-frontier row O, componentwise hsame *transport H,* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N remain displayed before any consumer can read a parallel-diamond request. The route does not assert unrestricted confluence, host reduction equality, unbounded critical-pair search, or a global normalization theorem.*

Proof. Project the carrier and the existing critical-pair envelope. The projection exposes $P, K, J, R, S, C, B, O, H, T$ as the only rows available to a frontier request.

Read the candidate and residual rows first. The candidate term rows K, J store the finite overlap and join candidates, while P, R keep substitution and residual windows attached to the same bounded branch data.

Next read the finite observation confluence rows S, C, B and the normalization-frontier row O. These rows certify only the displayed closed-term checks, bounded normal fragments, checker budget, and obstruction boundary.

Finally replay through H, T, G, N. Since transport, replay, provenance, and local naming preserve the same finite packet, the accepted critical-pair request cannot be replaced by unrestricted confluence, host reduction equality, unbounded search, or global normalization. □

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCriticalPairEnvelop

Theorem 4238.51 (MetaCIC parallel diamond frontier visible overlap)**.** *Every accepted* MetaCICParallelDiamondFr *packet exposes local overlap only through*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Any consumer that reads the candidate join J must first keep the residual-substitution premise row P, the critical-pair row K, the residual-frontier row R, and the bounded checker rows S, C, B visible. The read is therefore a bounded overlap frontier, not subject reduction, full Church–Rosser, unrestricted confluence, host reduction equality, or kernel consistency.

Proof. Start with the critical-pair envelope theorem Definition 4238.49. It exposes the same finite route $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the residual-substitution and confluence audit-map rows visible.

Apply the frontier envelope theorem Definition 4238.50. It orders critical-pair requests so that K, J are read with P, R, S, C, B, O still attached.

Finally use the route theorem Definition 4238.9. Replay, provenance, transport, and local naming through H, T, G, N preserve the visible premise and bounded checker rows before the candidate join can be consumed. Since the route theorem does not discharge residual substitution or subject reduction, the visible-overlap read remains a frontier certificate rather than a completed confluence result. □

Theorem 4238.52 (MetaCIC parallel-diamond critical-pair join envelope)**.** *Every candidate-mediated strong-normalization or confluence consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first display the critical-pair envelope, residual-substitution dependency route, residual-join locality surface, finite residual witness row, and obstruction row for the same finite carrier*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The consumer may read the candidate join only as this bounded envelope. It does not discharge residual substitution, prove subject reduction, identify evaluator equality, identify quotient term equality, prove global confluence, or assert kernel consistency.

Proof. Project the accepted frontier carrier. The displayed rows are the retained residual-substitution premise P, critical-pair row K, candidate join J, residual frontier R, closed-checker row S, closed-normal fragment C, bounded checker B, obstruction row O, and structural rows H, T, G, N.

Read the envelope rows in order. The critical-pair envelope keeps the bounded overlap data visible, while the residual-substitution dependency route places P before any join handoff.

Now attach the local join and witness rows. Residual-join locality keeps J inside the same frontier packet, and the finite residual witness row forces the typed windows, residual data, bounded checker rows, transport, replay, provenance, and local naming to remain visible.

Finally retain the obstruction row. Since O, H, T, G, N preserve only the bounded carrier-local packet, a candidate-mediated consumer receives an envelope rather than residual-substitution discharge, subject reduction, evaluator equality, quotient term equality, global confluence, or kernel consistency. □

Theorem 4238.53 (MetaCIC parallel-diamond frontier critical-pair route)**.** *Every critical-pair read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier exposes candidate membership, bounded local peak rows, residual envelope rows, retained premise rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *before any join row is consumed. For a carrier*

$$F = (P, K, J, R, S, C, B, O, H, T, G, N),$$

the read must display the retained premise P, local peak row K, residual envelope R, bounded checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before the candidate-join row J can serve a confluence consumer.

Proof. Project the carrier of Definition 4238.1. The only rows available to a critical-pair consumer are $P, K, J, R, S, C, B, O, H, T, G, N$, with P retaining the premise and K, R, S, C, B, O displaying the local peak, residual, checker, and obstruction envelope.

Apply critical-pair exhaustion and residual-substitution scope, Definition 4238.4 and Definition 4238.12. They force the local peak through K under the retained premise P and keep the residual frontier R visible before the candidate join is read.

Finally use the critical-path retained-premise handoff and seed closure status, Definition 4238.85 and Definition 4238.45. These routes pass the same critical-pair read through H, T, G, N without consuming J as a detached join or erasing the obstruction row. □

Theorem 4238.54 (MetaCIC parallel-diamond frontier residual-join nonescape)**.** *Every residual-join read exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier is a candidate-mediated local join through $P, K, J, R, S, C, B, O, H, T, G, N$. The frontier cannot be consumed as full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, or a discharged residual-substitution theorem.*

Proof. Start from the residual-join induction and public export theorems, Definition 4238.46 and Definition 4238.47. They keep finite residual joins inside the displayed route through the retained premise, critical-pair, candidate, residual, checker, obstruction, transport, replay, provenance, and naming rows.

Use local-join scope and the retained-premise handoff, Definition 4238.31 and Definition 4238.85. A join can be read only through the candidate row J under the same premise P and residual envelope R; it is not a premise-free confluence endpoint.

The refused endpoints remain outside the carrier. Since O is still visible and no row supplies subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, or residual-substitution discharge, the exported join remains candidate-mediated and local. □

Theorem 4238.55 (MetaCIC parallel-diamond critical-path consumer route)**.** *Every critical-path consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, candidate-join row J, closed-checker row S, closed-normal row C, bounded-check row B, and obstruction row O before any local confluence handoff*

is public. The required consumers are the critical-path packet of chapter 765, the confluence audit packet of chapter 735, and the normalization-confluence bridge of chapter 1190.

Proof. Project the carrier of Definition 4238.1. Its confluence-facing coordinates include P, K, J, S, C, B, O, so none of the three consumer chapters can read the frontier through a smaller public row set.

The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ consumer receives the residual-substitution premise, critical-pair, and obstruction rows as route pressure rather than as a discharged local-confluence theorem. The $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ consumer receives the candidate-join, closed-checker, and bounded-check rows as audit data before it may inspect the local handoff. The MetaCICNormalizationConfluenceBrid consumer receives the closed-normal row and the same obstruction boundary as bridge-facing evidence.

Thus the critical-path, confluence-audit, and normalization-confluence consumers share the full retained frontier. A local confluence handoff is public only after those rows have been displayed, and the route still does not assert residual-substitution compatibility, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.56 (MetaCIC critical-path bridge frontier)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that is consumed by a MetaCIC critical-path reader must keep its residual diamond rows as concrete candidate and discharge data for* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$*, and* $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$*. In carrier terms, the rows* $P, K, J, R, S, C, B, O, H, T, G, N$ *may be routed into the critical-path, confluence-audit, and normalization-confluence bridge surfaces only as retained frontier evidence; they do not form a standalone confluence oracle, subject-reduction theorem, evaluator equality row, quotient equality row, or kernel-consistency certificate.*

Proof. Project the frontier carrier through Definition 4238.1. The residual-substitution premise, critical-pair data, candidate join, residual window, closed checker, closed-normal fragment, bounded check, and obstruction rows remain displayed before the transport, replay, provenance, and local name rows are read.

Use the route and handoff theorems of Definition 4238.9 and Definition 4238.22. They make the exported handoff conditional on the retained premise and obstruction rows, so the confluence-audit packet can consume the rows only as bounded local evidence.

Now read the critical-path and normalization-confluence consumers. The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ surface receives the same rows as candidate and discharge pressure, and $\mathsf{MetaCICNormalizationConfluenceBridge}^{\uparrow}$ receives them as bridge-facing boundary data attached to the confluence audit route.

Thus the sibling route is substantive: the frontier contributes concrete residual diamond rows to the critical-path and bridge consumers, while every standalone confluence-oracle, subject-reduction, evaluator-equality, quotient-equality, and kernel-consistency reading remains outside the packet. □

Theorem 4238.57 (MetaCIC parallel-diamond residual-substitution dependency route)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that is consumed as a parallel-diamond frontier must first expose the residual-substitution premise row supplied by chapter 2181. The consumer may route that premise into the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*, but it cannot replace the premise by residual-substitution compatibility, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualSubstitutio`

Proof. Project the frontier carrier of Definition 4238.1. The residual-substitution premise row P is one of the displayed carrier coordinates, not an external theorem attached after the handoff.

Read the residual-substitution boundary chapter named in the statement. It supplies the visible premise boundary that the frontier route consumes before any candidate join or local-diamond handoff is public.

Finally apply the premise-retention and discharge-obstruction surfaces already present in the frontier chapter. They keep P and the obstruction row visible for every consumer, so the dependency route remains a premise boundary rather than any of the refused confluence endpoints. □

Theorem 4238.58 (MetaCIC parallel-diamond residual-premise join determinacy)**.** *Let* $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *and* $F' = (P, K, J', R, S, C, B, O, H', T', G', N')$ *be accepted* MetaCICParallelDiam *packets whose residual-substitution premise row* P*, critical-pair row* K*, residual-frontier row* R*, closed-checker row* S*, closed-normal row* C*, bounded-check row* B*, and obstruction row* O *agree on the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *surface. Any common candidate-join handoff exposed by both packets is therefore read only through the same displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *route, with* J *the candidate row visible on that route. The handoff does not discharge residual substitution, prove subject reduction, give full Church–Rosser, identify evaluator equality, quotient terms, or assert kernel consistency.*

Proof. Project the two frontier carriers through Definition 4238.1. The rows named in the statement are carrier coordinates, so a common candidate-join read cannot be detached from the shared P, K, R, S, C, B, O boundary.

Now apply the dependency route of Definition 4238.57. It places the residual-substitution premise P before the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows.

Finally use the premise-retention and route surfaces already recorded in the chapter. They keep the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ visible for every accepted consumer, so the shared handoff is deterministic only at the finite route boundary and remains separate from the refused confluence and kernel endpoints. □

Theorem 4238.59 (MetaCIC parallel-diamond residual-substitution confluence budget)**.** *Every residual-substitution route that contributes to the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *parallel-diamond frontier must expose, in order, the retained premise row* P*, residual pair row* R*, local join row* J*, substitution dependency row* S*, and finite budget row* B *before any candidate-mediated strong-normalization handoff may consume it. The budget is a finite row of the frontier carrier, not a global confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency witness.*

Proof. Project the frontier carrier of Definition 4238.1. The retained premise, residual pair, local join, substitution dependency, and budget rows are displayed carrier coordinates, so a consumer cannot read the candidate handoff before these rows are present.

Apply the residual-substitution dependency route of Definition 4238.57. It places the retained premise boundary before the critical-pair and candidate-join rows and keeps residual substitution as a visible dependency of the frontier.

Finally use residual-join locality from the surrounding frontier rows. The local join is certified only inside the finite P, R, J, S, B route, and the candidate-mediated handoff consumes that route as a bounded frontier witness. Since no row stores global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the budget. □

Theorem 4238.60 (MetaCIC residual-substitution discharge criterion)**.** *Let an accepted* MetaCICParallelDiamondF *packet supply the retained residual-substitution premise row and let a separate* MetaCICResidualSubstitutionCompatib *certificate be displayed. Then the local-diamond handoff may be read as a discharge-ready boundary for the candidate-normalization confluence handoff.*

This boundary is still only the finite frontier route. It is not full Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency.

Proof. Start with the discharge obstruction of Definition 4238.18. It identifies the retained residual-substitution premise as the row that remains visible until a separate compatibility certificate is supplied.

Next apply Definition 4238.19. The displayed $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ certificate is exactly the extra row allowed to turn the premise-obstruction surface into a discharged local-diamond boundary.

Finally use the confluence handoff of Definition 4238.22. The handoff remains a candidate-normalization consumer of the finite frontier route, so it exports only discharge-ready local-diamond evidence and none of the refused global endpoints. □

Theorem 4238.61 (MetaCIC observation confluence budget)**.** *Every observation-level confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, critical-pair row K, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, and obstruction row O before reading any local join through the candidate row J. The observation budget is the finite frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It may cite the retained-premise route of Definition 4238.12 and the public frontier route of Definition 4238.9, but it does not prove residual-substitution compatibility, subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with the carrier and obligation basis of Definition 4238.1 and Definition 4238.34. They make $P, K, J, R, S, C, B, O, H, T, G, N$ the consumer-visible packet, with P and O ordinary rows rather than external assumptions.

Apply residual-substitution scope and the frontier route, Definition 4238.12 and Definition 4238.9. These force the premise row, critical-pair row, residual-frontier row, closed-checker row, and closed-normal fragment row to remain visible before the conditional handoff is consumed.

Finally use local-join scope Definition 4238.31. A local join can be read only after the candidate row has been placed inside the same typed-window surface with B and O still exposed. Thus the theorem records an observation budget for finite confluence consumers, not a closed confluence proof or a discharge of the residual-substitution obstruction. □

Theorem 4238.62 (MetaCIC candidate-normalization scope)**.** *Every candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the full frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before it reads the candidate row. The consumer may use J only as the displayed candidate-join row of a candidate-mediated residual route, and only while the residual-substitution premise row P and obstruction row O remain visible. Thus the candidate-normalization read is not a premise-free confluence theorem, not a completed residual-substitution theorem, not full normalization, not subject reduction, not evaluator equality, not quotient equality of terms, and not a kernel-consistency certificate.

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCandidateNormalizat

Proof. Begin with the candidate-mediated residual route of Definition 4238.29. It permits a local peak to reach the candidate row J only through the displayed critical-pair row K and residual-frontier row R, with the retained premise row P still attached.

Next apply substitution nonescape, Definition 4238.38. That theorem keeps substitution data confined to P and R through the typed-window local join, so the candidate-normalization consumer cannot turn J into a discharged substitution result.

Use residual pasting, Definition 4238.37. Even after two typed-window residual reads are pasted through J, the pasted read remains the finite residual packet under P and still carries S, C, B, O, H, T, G, N.

Finally apply the normalization-frontier handoff of Definition 4238.40. The handoff exports only the conditional spine $P, K, J, R, S, C, B, O, H, T, G, N$ to normalization consumers, so P and O cannot be hidden when J is consumed. □

Theorem 4238.63 (MetaCIC parallel-diamond frontier obstruction retention under candidate normalization)**.** *Every candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *carrier that reads the candidate row J must keep both the residual-substitution premise row P and obstruction row O visible through the full spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read can use J only as a candidate-normalization row inside that conditional spine. It cannot retype candidate normalization as premise-free confluence, residual-substitution compatibility, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Begin with the candidate-normalization scope theorem, Definition 4238.62. It already requires the consumer to expose $P, K, J, R, S, C, B, O, H, T, G, N$ before the candidate row is read, with P and O still visible.

Next use substitution nonescape, Definition 4238.38. The substitution premise remains confined to the displayed residual route, so reading J cannot discharge P or replace it by an unconditional compatibility theorem.

Apply residual pasting, Definition 4238.37. Pasting two typed-window residual reads through J preserves the same finite residual packet and keeps the obstruction row O in the exported spine.

Finally hand the read to Definition 4238.40. That handoff exports only the conditional normalization-facing route, so the consumer cannot obtain premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency from candidate normalization alone. □

Theorem 4238.64 (MetaCIC candidate-normalization residual budget)**.** *A candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet may spend only the displayed finite residual rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Every normalization-facing read must keep the residual-substitution premise row P, candidate-join row J, residual-frontier row R, and obstruction row O visible. The budget excludes premise-free confluence, subject reduction, evaluator equality, quotient term equality, and kernel consistency.

Proof. Start with the candidate-normalization scope theorem, Definition 4238.62. It makes the displayed spine $P, K, J, R, S, C, B, O, H, T, G, N$ the only route available to a candidate-normalization consumer.

Use obstruction retention under candidate normalization, Definition 4238.63. This keeps P and O visible when the candidate row J is read, so the residual-substitution premise and obstruction cannot be silently discharged.

Next compose substitution nonescape with residual pasting, Definition 4238.38 and Definition 4238.37. These preserve the residual-frontier row R and candidate-join row J as finite rows of the same packet.

Finally apply the normalization-frontier handoff, Definition 4238.40. The handoff exports only the conditional normalization-facing spine, so the consumer cannot spend the residual budget as premise-free confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.65 (MetaCIC parallel-diamond candidate-normalization frontier scope)**.** *Every candidate-normalization read used by the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must pass through the visible candidate endpoint row, residual-diamond budget, retained-premise boundary, residual-join locality row, and normal-form obligation row before it can feed a local-confluence consumer. Equivalently, the read remains inside the finite frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where J is read only as the displayed candidate endpoint, R, S, C, B keep the residual-diamond and normal-form budget visible, P and O keep the retained premise and obstruction boundary visible, and H, T, G, N preserve transport, replay, provenance, and local naming. The scope is therefore a bounded candidate-normalization frontier; it is not premise-free local confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with candidate-normalization scope, Definition 4238.62. It requires any consumer of the candidate row to expose the full frontier spine before the candidate endpoint J is read.

Next apply the candidate-mediated strong-normalization handoff of Definition 4238.73. This keeps the candidate endpoint attached to the critical-pair envelope, finite residual witness, bounded checker rows, obstruction row, transport, replay, provenance, and local naming.

Use the residual-join obligation surface, Definition 4238.138. It supplies the residual-diamond budget and local residual-join obligation rows, so a local-confluence consumer cannot read candidate normalization before the residual budget and local join route are visible.

Finally apply the retained-premise closure budget of Definition 4238.150. The retained premise and obstruction rows remain displayed through the normal-form obligation route, so the candidate-normalization read stays inside the bounded frontier and exports none of the refused global endpoints. □

Theorem 4238.66 (MetaCIC candidate-mediated diamond budget route)**.** *Every candidate-mediated parallel-diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose residual-substitution premise retention, critical-pair envelope rows, candidate normalization handoff, bounded obstruction rows, closed checker rows, and local* $\mathsf{NameCert}$ *rows before any confluence-facing conclusion is exported. The read is the finite route through the displayed frontier packet, not full normalization, global confluence, or host definitional equality.*

Proof. Start with the residual-substitution route Definition 4238.39. It keeps the residual-substitution premise row visible through the same packet that carries critical-pair and candidate rows.

Next read the critical-pair envelope and candidate-normalization scope, Definition 4238.50 and Definition 4238.62. These rows expose the local pair and candidate handoff without discharging substitution or turning the handoff into a global normalization theorem.

Finally apply bounded local-confluence decision and finite residual witness consumption, Definition 4238.70 and Definition 4238.69. They keep the bounded obstruction, closed checker, transport, replay, provenance, and local naming rows attached before a confluence-facing consumer can inspect the result. Hence the exported conclusion is only the finite candidate-budget route stated above. □

Theorem 4238.67 (MetaCIC parallel diamond registered frontier route)**.** *A candidate diamond budget route can expose a bounded checker status and local join candidate only after the retained-premise route, candidate-normalization row, residual envelope, and frontier register have all been displayed. The exposed status is a local registered-frontier read from the same* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet; it is not subject reduction, residual-substitution discharge, quotient equality, global normalization, full confluence, or kernel consistency.*

Proof. Start with the seed route of Definition 4238.66. It already forces residual-substitution premise retention, critical-pair envelope rows, candidate normalization handoff, bounded obstruction rows, closed checker rows, and local NameCert rows to remain visible before any confluence-facing conclusion is exported.

Next read the retained-premise and candidate-normalization rows through the same frontier packet. The existing residual-substitution route keeps the premise attached, while the candidate-normalization scope names the local join candidate without turning it into global normalization or subject reduction.

Finally use the frontier and local-join scope rows already recorded in the chapter. They register the bounded checker status only after the residual envelope and frontier display have been read, so the exported route remains a local finite checker route and refuses the global confluence, quotient, discharge, normalization, and consistency endpoints. □

Theorem 4238.68 (MetaCIC parallel-diamond candidate critical-pair boundary)**.** *Every local diamond budget read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the residual-substitution premise row P, the two critical-pair branch rows carried by K and R, the candidate join row J, the normalization-confluence handoff row, the critical-path socket, and the obstruction row O before any confluence-audit consumer can inspect the read. The boundary remains candidate-mediated; it does not assert subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Begin with seed readiness and public seed readiness, Definition 4238.83 and Definition 4238.84. These rows force the residual branches, candidate join, normalization-confluence handoff, and public frontier route to remain visible before any confluence-facing read is exported.

Next apply the critical-path retained-premise handoff, Definition 4238.85. It keeps the residual-substitution premise and obstruction row attached to the critical-path socket rather than letting the socket discharge the premise.

Finally use the obligation-closure package Definition 4238.87. The package assembles the residual branches, candidate join, normalization-confluence handoff, critical-path socket, and obstruction retention as the finite local surface. Since the surface is still candidate-mediated, none of the refused confluence, equality, normalization, discharge, or kernel endpoints is exported. □

Theorem 4238.69 (MetaCIC parallel-diamond finite residual witness consumption)**.** *Every accepted residual-pair local-diamond witness read in a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must consume the typed window, residual frontier, substitution-boundary row, bounded local-join checker,*

candidate-strong-normalization evidence when present, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it may be used as input to a closed parallel-diamond route. The consumed packet is the finite route through* $P, K, J, R, S, C, B, O, H, T, G, N$*; it is not a proof of arbitrary closed Church–Rosser.*

Lean4 ✓ BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierFiniteResidualWitne

Proof. Start with the typed-window local-join scope of Definition 4238.31. It forces a local-diamond consumer to expose the typed window, candidate join, residual frontier, checker rows, and obstruction row before any join read is available.

Next use residual boundary and substitution nonescape, Definition 4238.33 and Definition 4238.38. These keep the residual-substitution premise and substitution-boundary row visible through the finite residual route rather than allowing them to be discharged by structural replay.

Apply the residual-substitution route Definition 4238.39. It identifies the local-diamond input as the conditional frontier packet through $P, K, J, R, S, C, B, O, H, T, G, N$, including transport, replay, provenance, and local naming.

Finally use the candidate-mediated strong-normalization socket Definition 4238.41. When candidate-SN evidence is present, it is consumed as the same finite socket before the closed parallel-diamond route can read the witness. Since the route retains P and O and exports only the displayed packet, it does not prove arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency. □

Theorem 4238.70 (MetaCIC parallel-diamond bounded local-confluence decision)**.** *A bounded local-join checker for the displayed finite residual witness of a* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may return a positive local-confluence certificate only after exposing the typed window, residual frontier, substitution-boundary row, candidate join, bounded checker status, componentwise transport, displayed replay, provenance, and local* NameCert *rows. The positive result is a bounded local-confluence certificate for that finite residual witness; it is not residual-substitution compatibility, subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with local-join scope. By Definition 4238.31, a checker read must expose the typed window, candidate join, residual frontier, checker rows, and obstruction row before any local join can be used.

Next apply residual boundary and substitution nonescape, Definition 4238.33 and Definition 4238.38. These rows keep the substitution-boundary premise visible and prevent the bounded decision from being read as residual-substitution compatibility.

Use finite residual witness consumption from Definition 4238.69. It forces the checker status, candidate join, transport, replay, provenance, and local naming rows to be consumed as the finite route through $P, K, J, R, S, C, B, O, H, T, G, N$.

The checker may therefore certify only that this displayed finite residual witness has a bounded local join. Since the route still retains the typed window and substitution-boundary rows and never exports a global reduction relation, the certificate does not imply subject reduction, unrestricted confluence, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.71 (MetaCIC parallel-diamond finite candidate frontier)**.** *Any bounded local-confluence certificate consumed from a* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is limited to the displayed finite candidate rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Candidate-normalization rows and residual-substitution rows must remain visible before a closed parallel-diamond route can read the result. The certificate therefore records only a finite candidate frontier for the displayed packet; it is not full Church–Rosser, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency.

Proof. Begin with finite residual witness consumption, Definition 4238.69. It forces every local-diamond witness read to pass through the typed window, residual frontier, substitution-boundary row, bounded local-join checker, transport, replay, provenance, and local naming rows.

Next use bounded local-confluence decision, Definition 4238.70. A positive checker result is available only for that displayed finite residual witness and only after the candidate join, checker status, and structural rows have been exposed.

Now attach the candidate-mediated strong-normalization socket and the residual-substitution route, Definition 4238.41 and Definition 4238.39. These keep candidate normalization and residual substitution inside the same $P, K, J, R, S, C, B, O, H, T, G, N$ spine before the closed route may inspect the handoff.

Thus the consumer receives a finite candidate frontier and nothing stronger. Since the residual-substitution and candidate-normalization rows remain visible, the certificate cannot be reread as a premise-free confluence theorem, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 4238.72 (MetaCIC parallel-diamond frontier residual-row visibility)**.** *Any accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *consumer that exports a parallel-diamond handoff must keep the residual-substitution row P, local join row J, typed residual budget row R, componentwise transport row H, displayed replay row T, provenance row G, and local* NameCert *row N visible. The consumer cannot cite bounded normal equality, closed checker data, or confluence audit packets as a replacement for the residual row.*

Proof. Project the frontier carrier of Definition 4238.1. The accepted packet exposes the residual-substitution premise, candidate or local join, residual-frontier budget, transport, replay, provenance, and local naming rows as displayed coordinates rather than as ambient metadata.

Cite the residual-substitution boundary and local-join ledger sibling routes. The boundary keeps the substitution premise explicit, while the local-join ledger reads the join only after the residual rows and bounded-check rows have been exposed.

The remaining claim is nonescape of the carrier rows. Bounded normal equality, closed checker data, and confluence audit packets can be consumed only through the displayed frontier route; none supplies a second coordinate that discharges or hides the residual-substitution row. □

Theorem 4238.73 (MetaCIC parallel-diamond candidate-mediated SN handoff)**.** *Every candidate-mediated strong-normalization budget consumed by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose, in order, the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before any confluence or decidability consumer reads the frontier. When the candidate-normalization confluence handoff is consumed, the retained residual-substitution premise row P remains visible in the same read. The consumed route is the finite spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retained as the residual-substitution premise, K, J, R retained as the local critical-pair and residual-frontier rows, S, C, B, O retained as the bounded checker and obstruction rows, and H, T, G, N retained as transport, replay, provenance, and local naming. The handoff is therefore a

conditional residual-join frontier and candidate-mediated budget, not a proof of unrestricted confluence, unbounded strong normalization, subject reduction, evaluator equality, quotient term equality, decidability of all reductions, or kernel consistency.

Proof. First read the critical-pair envelope through Definition 4238.50. It exposes K, J, R, S, C, B, O as a bounded local-overlap surface and keeps the frontier carrier rows displayed.

Next read the residual-substitution dependency route of Definition 4238.57. That route places the retained premise row P before the candidate join, residual frontier, closed-checker, and obstruction rows can be consumed.

The candidate-normalization scope of Definition 4238.62 keeps J inside the same conditional spine. It prevents a normalization consumer from hiding P or retyping the candidate row as premise-free confluence.

Together, that candidate-normalization scope and the residual-substitution dependency route require the same retained premise row P to remain visible when the candidate-normalization confluence handoff is consumed; the consumer receives a conditional residual-join frontier, not premise-free confluence or a kernel-consistency certificate.

Now consume the finite residual witness of Definition 4238.69. The witness reads typed windows, residual frontier data, bounded local-join checks, candidate evidence when present, transport, replay, provenance, and local naming before any closed parallel-diamond route can use it.

Finally apply residual-row visibility, Definition 4238.72. The residual row remains visible through H, T, G, N, so confluence and decidability consumers receive only the displayed candidate-mediated budget rather than a discharged substitution theorem, global normalization theorem, evaluator equality, quotient equality, or kernel-consistency certificate. □

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierCandidateSnHandoff`

Theorem 4238.74 (MetaCIC parallel-diamond candidate SN handoff scope)**.** *Every candidate-mediated strong-normalization consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row, critical-pair row, candidate join row, residual-frontier row, closed-checker row, closed-normal-fragment row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *before using the handoff. In the displayed spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the consumer reads P as the retained substitution premise, K as the critical-pair row, J as the candidate join row, R as the residual-frontier row, S, C, B as closed-checker, closed-normal-fragment, and bounded-check rows, O as the obstruction row, and H, T, G, N as transport, replay, provenance, and local naming. The scope is only a candidate-mediated SN handoff; it is not full substitution discharge, global confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the candidate-mediated SN handoff of Definition 4238.73. It already forces the accepted packet to display the residual-substitution premise, critical-pair envelope, residual-frontier rows, checker rows, obstruction row, transport, replay, provenance, and local naming before a confluence or decidability consumer reads the frontier.

Next use the retained-premise discharge boundary and residual-substitution dependency route. They keep P visible as a premise row rather than a discharged substitution theorem, so the candidate handoff cannot be read as premise-free confluence or subject reduction.

Finally apply residual-row visibility. The rows K, J, R, S, C, B, O remain attached to H, T, G, N, and every displayed Cont replay preserves the same local NameCert route. Thus the consumer obtains exactly the scoped candidate-mediated handoff and none of the excluded global equality or consistency claims. □

Theorem 4238.75 (MetaCIC parallel diamond typed candidate SN extension)**.** *Every candidate-mediated strong-normalization handoff consumed from an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must retain the residual-substitution premise, critical-pair envelope, candidate row, typed-window row, closed-checker row, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local* NameCert *row before any normalization-facing consumer reads it. In the frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P remains the residual-substitution premise, K, J, R retain the critical-pair, candidate, and typed-window residual data, S, C, B, O retain closed-checker, bounded-check, and obstruction data, and H, T, G, N retain transport, replay, provenance, and local naming. The extension is a typed candidate socket only; it is not global strong normalization, full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the candidate-normalization scope of Definition 4238.62. It forces any candidate-normalization consumer to expose the full spine $P, K, J, R, S, C, B, O, H, T, G, N$ before reading the candidate row.

Next apply the candidate-mediated SN handoff above. The handoff keeps the residual-substitution premise P, critical-pair envelope, candidate row, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before any confluence or decidability consumer reads the frontier.

Use finite residual witness consumption, Definition 4238.69. The typed window and residual frontier data remain in the same finite packet with the closed-checker, bounded-check, candidate evidence, transport, replay, provenance, and local NameCert rows.

Finally invoke residual-row visibility, Definition 4238.72. Since the residual-substitution and typed residual rows remain visible through H, T, G, N, a normalization-facing consumer receives only the typed candidate socket and cannot read it as global strong normalization, full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.76 (MetaCIC parallel-diamond candidate SN frontier extension)**.** *Let $F = (P, K, J, R, S, C, B, O, I$ be an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet. If a consumer displays a candidate-normalization row J, a residual-pair frontier R, a bounded local-join checker B, and the retained substitution-boundary row P through the same transport, replay, provenance, and local naming rows H, T, G, N, then the candidate strong-normalization handoff of F extends to that bounded residual-join frontier. The conclusion is only a bounded candidate-mediated frontier handoff; it does not prove arbitrary closed Church–Rosser, premise-free confluence, global strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. First apply the candidate-mediated SN handoff of Definition 4238.73. It keeps the residual-substitution premise, critical-pair envelope, candidate row, finite residual witness, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before any SN-facing consumer reads the frontier.

Next use the typed candidate SN extension of Definition 4238.75. That theorem places J inside the typed residual window and prevents the candidate row from being retyped as a global normalization theorem.

Now read the residual-frontier siblings. The critical-pair envelope and residual-substitution dependency route, Definition 4238.50 and Definition 4238.57, keep K, R downstream of the retained premise P. Finite residual witness consumption and residual-row visibility, Definition 4238.69 and Definition 4238.72, keep the bounded checker and frontier rows attached to H, T, G, N.

Therefore the displayed candidate-normalization row, residual-pair frontier, bounded local-join checker, and substitution-boundary row form a single bounded handoff. Since every step retains P and the structural rows, the exported result remains a candidate-mediated frontier extension rather than an unrestricted Church–Rosser or global normalization theorem. □

Theorem 4238.77 (MetaCIC parallel-diamond candidate SN retained-premise boundary)**.** *Every candidate-mediated strong-normalization handoff exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet retains the residual-substitution premise row and the obstruction rows before it may feed the normalization-confluence handoff. In the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the rows P, J, R, B, O remain visible as premise, candidate-normalization, residual-frontier, bounded-check, and obstruction data, while H, T, G, N preserve the same route under transport, replay, provenance, and local naming. The boundary is not a proof of subject reduction, full Church–Rosser, evaluator equality, quotient term equality, residual-substitution discharge, global strong normalization, or kernel consistency.

Proof. Start with the candidate-mediated SN handoff of Definition 4238.73. It already forces the consumer to expose the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before any confluence or decidability read.

Now apply the residual-substitution dependency route and candidate-normalization scope, Definition 4238.57 and Definition 4238.62. These rows keep P upstream of J, R, S, C, B, O and prevent the candidate row from being consumed as premise-free normalization or confluence.

Finally use finite residual witness consumption and residual-row visibility, Definition 4238.69 and Definition 4238.72. The bounded witness and obstruction rows remain attached to H, T, G, N, so the normalization-confluence handoff receives only the retained finite packet and none of the excluded global endpoints. □

Theorem 4238.78 (MetaCIC parallel-diamond frontier seed registration)**.** *Every seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that presents a conditional parallel-diamond handoff must first expose the residual-substitution premise row P, critical-pair row K, candidate join J, residual frontier R, closed checker S, closed-normal fragment C, bounded check B, obstruction O, and structural rows H, T, G, N. The handoff is public only at this finite row surface; it does not claim a premise-free diamond, full Church–Rosser theorem, subject reduction, evaluator equality, quotient equality of terms, or kernel consistency.*

Proof. Start from the obligation surface of Definition 4238.2. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the visible rows of the accepted frontier packet, with P retained as an actual residual-substitution premise.

Next use the route theorem Definition 4238.9. A conditional diamond handoff cannot be consumed until the critical-pair, residual-frontier, closed-checker, closed-normal, bounded-check, and obstruction rows have been read alongside the structural rows.

Finally apply premise retention, Definition 4238.13. The premise row remains visible through the local-diamond route, so the seed registration records the finite frontier interface rather than any of the refused confluence or kernel endpoints. □

Theorem 4238.79 (MetaCIC parallel-diamond frontier seed-status route)**.** *The hub seed registration for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is valid only through the retained rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and the sibling critical-path consumer route. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Read the carrier and obligation surfaces, Definition 4238.1 and Definition 4238.2. They make $P, K, J, R, S, C, B, O, H, T, G, N$ the whole retained packet for the seed registration.

Next use the route theorem and seed-readiness theorem, Definition 4238.9 and Definition 4238.83. These force the seed read to keep the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local name rows visible.

Finally apply the critical-path consumer route, Definition 4238.55. The downstream consumer receives only that retained row surface, so the seed-status route remains a registered frontier boundary rather than any of the refused residual-substitution, subject-reduction, confluence, evaluator-equality, quotient-equality, or kernel-consistency endpoints. □

Theorem 4238.80 (MetaCIC parallel-diamond frontier seed register)**.** *The seed register for* MetaCICParallelDiamor *is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

read through the carrier, obligation, seed-registration, and seed-status routes. The register records the chapter as a seed-level frontier packet only; it does not alter the closurestatus theory axis, does not upgrade the verification axis, and does not claim residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier and obligation surfaces of Definition 4238.1 and Definition 4238.2. They expose exactly the rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply Definition 4238.78 and Definition 4238.79. Both routes keep the same retained rows visible before any seed consumer reads the frontier.

The closurestatus block remains in the dedicated route file and reports the independent theory and verification axes. Since this theorem only registers the finite seed surface, it cannot discharge any of the refused MetaCIC endpoints. □

Theorem 4238.81 (MetaCIC parallel-diamond seed registration child route)**.** *The seed registration of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is read through the local child rows for critical-pair envelope, residual-join locality, retained-premise budget, seed closure status, closurestatus guard, and local* NameCert *obligations. Concretely, every seed-registration consumer must expose*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

by way of Definition 4238.50, Definition 4238.104, Definition 4238.150, Definition 4238.45, Definition 4238.180, and Definition 4238.2. The child route records seed registration only; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the carrier and local NameCert obligations. They make the retained premise, critical-pair data, candidate join, residual witness, typed window, bounded checker, obstruction, transport, replay, provenance, and local naming rows the whole public packet.

Read the critical-pair envelope and residual-join locality rows next. These keep the local diamond request carrier-local and keep the residual-substitution premise visible before any seed consumer can inspect a join.

The retained-premise closure budget then prevents the premise row from disappearing under seed registration. The seed closure status reader and closurestatus guard identify the seed marker as a pointer into the same displayed packet, rather than as an independent confluence or kernel claim.

Thus the child route is exactly the finite registration spine for the hub: it ties the seed row to existing child files while refusing residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency. □

Theorem 4238.82 (MetaCIC parallel-diamond frontier seed package)**.** *Every seed consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet receives the seed package only after the existing carrier, obligation surface, handoff route, critical-pair exhaustion rows, obstruction budget, closure seed route, and residual-normalization window have all been displayed. Equivalently, the package is the finite surface*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

read through Definition 4238.1, Definition 4238.2, Definition 4238.9, Definition 4238.66, Definition 4238.100, and Definition 4238.126. It is a registered seed package, not a full confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, global normalization theorem, residual-substitution discharge, or kernel-consistency certificate.

Proof. Start with the carrier and obligation surfaces. They display the retained residual-substitution premise, critical-pair row, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction row, transport, replay, provenance, and local naming rows.

Next read the handoff and critical-pair budget rows. The route theorem and candidate-diamond budget route keep the local diamond inside the displayed candidate window and leave the obstruction budget visible to downstream consumers.

Finally apply the closure seed route and residual-normalization window. These theorems register the same finite package as seed-level data only, so the consumer receives a frontier package rather than any of the refused confluence, subject-reduction, evaluator-equality, quotient-equality, normalization, residual-substitution, or kernel-consistency endpoints. □

Theorem 4238.83 (MetaCIC parallel-diamond frontier seed readiness)**.** *Every candidate-mediated parallel-diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose both residual branches, the common join candidate, the normalization-confluence handoff row, and the MetaCIC critical-path readback socket before any confluence-frontier claim is available. In carrier notation the readiness route keeps* $P, K, J, R, S, C, B, O, H, T, G, N$ *visible:* K *records the two residual branches,* J *records the common join candidate,* C *and* B *route the normalization-confluence handoff consumed by chapter 1190, and the critical-path socket of chapter 765 reads* P, R, O *as retained boundary data. The theorem is a finite route projection, not a normalization theorem, a strong-normalization theorem, an unrestricted confluence theorem, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the frontier carrier from Definition 4238.1. The branch and candidate data are visible only through the critical-pair row K, candidate-join row J, residual-frontier row R, and obstruction row O, with the structural rows attached.

Use the existing confluence handoff route in the chapter. It keeps the local diamond conditional on the residual-substitution premise and routes the closed-normal and bounded-check rows through the normalization-confluence bridge rather than treating them as a global normalization proof.

Finally read the MetaCIC critical-path packet as a socket boundary. The critical-path consumer receives the residual premise, residual windows, and obstruction rows as readback data, so the seed-readiness route exposes the two branches and common join candidate without discharging normalization, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.84 (MetaCIC parallel-diamond public seed readiness)**.** *Every public seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before any confluence-audit or normalization consumer can inspect the handoff. In particular, the residual-substitution premise row P and obstruction row O remain visible together with the critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, transport, replay, provenance, and local naming rows. The public seed read is therefore a retained-premise frontier, not full confluence, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start from Definition 4238.83. It already exposes the two residual branches, the candidate join, and the critical-path readback socket before any frontier claim can be consumed.

Next apply the obligation and route surfaces, Definition 4238.2 and Definition 4238.9. These force the public read through the rows $P, K, J, R, S, C, B, O, H, T, G, N$ rather than through a detached confluence theorem.

Use the premise-retention and obstruction ledgers already present in the chapter, including Definition 4238.23 and Definition 4238.36. They keep P and O visible before confluence-audit or normalization consumers inspect the candidate-mediated handoff.

Finally compose with the critical-path sibling route. The downstream consumer receives only the finite seed-readiness spine, so the packet does not discharge residual substitution, prove subject reduction, establish unrestricted confluence, identify evaluator or quotient equality, prove global normalization, or certify kernel consistency. □

Theorem 4238.85 (MetaCIC parallel-diamond critical-path retained-premise handoff)**.** *Any critical-path consumer reading the candidate-mediated local diamond row of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise P and the obstruction row O visible throughout the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The packet may be passed to $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and the normalization-confluence bridge of chapter 1190 only as retained frontier evidence. It is not a proof of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Start from public seed readiness, Definition 4238.84. It forces any public read of the frontier through the full route $P, K, J, R, S, C, B, O, H, T, G, N$ before a confluence-audit or normalization consumer may inspect the handoff.

Use the seed-readiness route Definition 4238.83. The candidate-mediated local diamond exposes the residual branches, common join candidate, normalization-confluence handoff row, and critical-path readback socket, while leaving the residual-substitution premise as retained boundary data.

Apply the premise preservation and critical-path consumer routes already named in the chapter, including Definition 4238.23 and Definition 4238.55. These keep P and O visible when the handoff is read by $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, or the normalization-confluence bridge.

The obstruction row then blocks endpoint promotion. Since O remains part of the exported frontier evidence, the consumer cannot reinterpret the handoff as subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency. □

Theorem 4238.86 (MetaCIC parallel-diamond residual-substitution local discharge)**.** *When an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet displays a residual-substitution row together with the critical pair, candidate join, residual-frontier, closed-checker, bounded-check, obstruction, transport, replay, provenance, and local* NameCert *rows*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

that packet locally discharges the finite residual window for the current candidate-mediated diamond only. The discharge is the visible closure of that displayed window under the present candidate join and retained obstruction; it is not subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with seed readiness and public seed readiness, Definition 4238.83 and Definition 4238.84. They expose the two residual branches, candidate join, residual-frontier row, closed-checker row, bounded-check row, obstruction row, and structural rows as one finite packet.

Apply the critical-path retained-premise handoff, Definition 4238.85. It keeps the residual-substitution premise P and obstruction row O visible while the critical-path consumer reads the candidate-mediated local diamond.

Use premise preservation together with the obligation-closure package, Definition 4238.23 and Definition 4238.87. These rows preserve the displayed premise and obstruction across the finite window and assemble the local obligation surface without promoting it to a global metatheorem.

Thus the only discharged object is the current finite residual window under the displayed candidate join. The packet has no coordinate for subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency, so none of those endpoints is exported. □

Theorem 4238.87 (MetaCIC parallel-diamond frontier obligation-closure package)**.** *The seed-readiness row of Definition 4238.83, the public seed-readiness row of Definition 4238.84, and the critical-path retained-premise handoff of Definition 4238.85 assemble the local obligation surface for residual branches, candidate join, normalization-confluence handoff, critical-path socket, and obstruction retention in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. The package refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, premise-free residual-substitution discharge, and kernel consistency.*

Proof. Project the displayed frontier rows. The carrier surface keeps the residual-substitution premise, two residual branches, candidate join, residual frontier, closed-checker row, normalization-confluence handoff rows, obstruction row, transport, replay, provenance, and local naming rows visible.

Apply Definition 4238.83. It places the residual branches, common join candidate, normalization-confluence handoff, and critical-path socket before any confluence-frontier read.

Next apply Definition 4238.84 and Definition 4238.85. These rows preserve the public route and keep the residual premise together with the obstruction row under critical-path consumption, so the package records obligation closure for the finite frontier surface without promoting it to subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, premise-free residual-substitution discharge, or kernel consistency. □

Theorem 4238.88 (MetaCIC parallel-diamond frontier seed closure)**.** *Any seed-closure read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must keep*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

visible, retain the residual-substitution premise P, and preserve the obstruction row O before any strong-normalization, confluence-audit, or normalization consumer reads the handoff. The seed closure is a finite retained-premise frontier and not global normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Start with the seed-readiness row of Definition 4238.83. It exposes the residual branches, common join candidate, normalization-confluence handoff row, and critical-path readback socket while keeping the route inside the displayed carrier rows.

Apply the public seed-readiness row of Definition 4238.84. This forces the full spine $P, K, J, R, S, C, B, O, H, T, G$ to remain visible before any confluence-audit or normalization consumer inspects the handoff.

Next use the critical-path retained-premise handoff of Definition 4238.85. It keeps the residual-substitution premise P and obstruction row O visible under MetaCICCriticalPath$^{\uparrow}$, MetaCICConfluenceAuditPacket$^{\uparrow}$, and normalization-confluence consumption.

Finally compose with the obligation-closure package of Definition 4238.87. The resulting seed closure records only retained finite frontier evidence, so it cannot discharge residual substitution or promote the handoff to strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.89 (MetaCIC parallel-diamond frontier seed obligations)**.** *Every confluence consumer reading an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first expose the seed-obligation surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The exposed rows include candidate pair rows, one-step residual rows, join-window rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the carrier-local* NameCert *row. A downstream reader may inspect a diamond conclusion only after these rows have been displayed as local frontier data; the seed surface is not subject reduction, global normalization, unrestricted confluence, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the accepted carrier from Definition 4238.1. The candidate pair and one-step residual data are carried by the critical-pair, candidate-join, and residual-frontier rows K, J, R, with the retained premise row P still visible.

Read the join-window and obstruction rows next. The closed-checker, closed-normal, bounded-check, and obstruction coordinates S, C, B, O keep the local join inside the displayed window instead of promoting it to a global confluence theorem.

Finally apply the structural rows. Componentwise transport H, replay T, provenance G, and local naming N preserve the same row order, so a consumer reaches a diamond conclusion only

through the finite seed-obligation surface and not through subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 4238.90 (MetaCIC parallel-diamond retained-obstruction obligation)**.** *Any confluence or normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise row P and obstruction row O visible through the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If either row is hidden, discharged, or replaced by an endpoint assertion, the read is not a discharge of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. The rows $P, K, J, R, S, C, B, O, H, T, G, N$ are the local data available to a confluence or normalization consumer, and the residual-substitution premise row P remains part of that display.

Apply Definition 4238.89. Its seed surface already forces the consumer to read the candidate pair, residual rows, join-window rows, closed-checker rows, obstruction row, transport, replay, provenance, and local naming before any diamond conclusion is inspected.

Now use the obstruction row O. It records that the accepted packet has not promoted the local diamond to subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency, so hiding P or O would erase a load-bearing premise rather than discharge it.

The public seed readiness of the chapter therefore remains conditional. A downstream consumer may pass along the finite candidate-mediated diamond surface only while P and O remain visible through the full route, and any endpoint promotion must be supplied by a separate packet. □

Theorem 4238.91 (MetaCIC parallel diamond frontier obligation closure route)**.** *Every residual-join or candidate-normalization consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, residual substitution dependency, candidate diamond budget, finite residual witness consumption, and residual row visibility before reading any local diamond conclusion. In row form, the consumer must display the route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

with P retaining the residual-substitution premise, K exposing the critical-pair envelope, J and B carrying the candidate diamond budget, R exposing finite residual witnesses, S and C checking the closed candidate surface, O retaining the obstruction, and H, T, G, N transporting, replaying, sourcing, and naming the same route. The conclusion is obligation closure for the displayed frontier only; it is not subject reduction, global confluence, strong normalization, substitution admissibility, hidden oracle access, or any closed MetaCIC theorem.

Proof. Begin with Definition 4238.89. It forces every confluence consumer to display the seed rows $P, K, J, R, S, C, B, O, H, T, G, N$ before a local diamond conclusion is inspected.

Use the retained-obstruction obligation of Definition 4238.90. The rows P and O remain load-bearing, so the residual-substitution dependency and the obstruction cannot be hidden when the residual join is read.

Now compose with the adjacent residual route files. The residual-witness and candidate-normalization routes expose finite witness consumption through R and the candidate budget through J, S, C, B before the frontier can export a local join.

The structural rows H, T, G, N keep the same displayed packet under transport, replay, provenance, and local naming. Since no row supplies subject reduction, global confluence, strong normalization, substitution admissibility, hidden oracle access, or a closed MetaCIC theorem, the route closes only the finite obligation surface it displays. □

Theorem 4238.92 (MetaCIC parallel-diamond discharge boundary)**.** *A candidate-mediated diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet can be consumed only while the residual-substitution premise row P and obstruction row O remain visible through the route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read is therefore a retained frontier surface. It does not discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. The candidate-mediated diamond data are displayed only through the rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply the seed-obligations theorem. It requires every confluence consumer to expose the retained premise, candidate pair, residual rows, join window, checker rows, obstruction row, transport, replay, provenance, and local naming before a diamond conclusion is inspected.

Now use the retained-obstruction obligation. That theorem keeps P and O visible as load-bearing rows, so a downstream read may pass along the candidate-mediated diamond surface only as a conditional frontier.

The excluded endpoint claims have no coordinate in the displayed route. Subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency would require separate discharge data, not the retained candidate packet alone. □

Theorem 4238.93 (MetaCIC parallel-diamond frontier obligation scope)**.** *Every accepted* MetaCICParallelDiamond *packet exposes the residual-substitution premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal row C, bounded-check row B, obstruction row O, and structural rows H, T, G, N before any confluence consumer reads a parallel-diamond handoff. The visible scope is*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It is a retained frontier surface, not a discharge of residual substitution, full Church–Rosser, subject reduction, evaluator equality, or kernel consistency.

Proof. Project the accepted carrier from Definition 4238.1. Its finite coordinates are exactly the retained premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and naming rows listed above.

Apply Definition 4238.89. That obligation surface forces a confluence consumer to display the candidate pair, residual rows, join-window rows, closed-checker rows, obstruction row, and structural certificate rows before inspecting any local diamond conclusion.

Use Definition 4238.90 for the retained boundary. The rows P and O remain visible as load-bearing premises, so the handoff cannot be read as residual-substitution discharge, full Church–Rosser, subject reduction, evaluator equality, or kernel consistency. □

Theorem 4238.94 (MetaCIC parallel-diamond critical-pair join)**.** *Every accepted finite critical-pair envelope for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *whose residual rows pass the candidate diamond budget admits a displayed join route through the finite surface*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route reads residual substitution from P, residual-row visibility from K and R, candidate-normalization from J, checker and bounded-budget rows from S, C, B, the retained obstruction from O, and then transport, replay, provenance, and local naming from H, T, G, N. The conclusion is a candidate-mediated local join for that displayed envelope, not a global confluence assumption, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency claim.

Proof. Project the accepted carrier from Definition 4238.1. The finite critical-pair envelope is visible only through $P, K, J, R, S, C, B, O, H, T, G, N$, so the candidate diamond budget is read as a condition on the displayed residual rows, not as a global confluence premise.

Split by the displayed residual envelope. In the cases where the residual rows fail the candidate budget, the theorem has no accepted join route to export. In the cases where the budget passes, the critical-pair row K and residual-frontier row R expose the two residual branches, while P keeps the residual-substitution premise visible.

Read the candidate-normalization and checker rows next. The candidate-join row J supplies the named local join proposal, and the rows S, C, B verify that the proposal stays inside the closed checker, closed-normal, and bounded-budget surface.

Finally apply the obstruction and structural rows. The retained obstruction O prevents the local join from being reread as Church–Rosser, subject reduction, evaluator equality, quotient equality, or kernel consistency, while H, T, G, N transport, replay, source, and name the same finite route. Thus the accepted envelope admits a displayed local join exactly through the candidate-mediated surface. □

Theorem 4238.95 (MetaCIC parallel-diamond frontier seed scope)**.** *Before any candidate-mediated strong-normalization route is exported from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the only seed-scope reads are finite critical-pair windows, residual substitution rows, candidate-normalization rows, and local join ledgers. In carrier notation these are the displayed rows $P, K, J, R, S, C, B, O, H,$ read as retained frontier data, not as subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Project the accepted carrier from Definition 4238.1. The finite critical-pair windows are carried by K and R, the residual-substitution row remains P, and the candidate-normalization read is routed through J, C, B with the obstruction row O still visible.

Apply the seed-obligation and seed-readiness routes, Definition 4238.89 and Definition 4238.83. They force every public seed read through the displayed frontier rows before any confluence-audit or normalization-facing handoff can be inspected.

The local join ledger is then completed only by transport, replay, provenance, and naming rows H, T, G, N. Since no carrier coordinate exports subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency, the stated rows are exactly the seed-scope reads available before the candidate-mediated strong-normalization route is exported. □

Theorem 4238.96 (MetaCIC parallel-diamond scoped-status route)**.** *The scoped closure status for an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet is witnessed by the finite dependency*

chain consisting of residual-substitution scope, premise preservation, critical-pair exhaustion, local-join scope, normalization-frontier handoff, candidate-mediated strong-normalization socket, typed-window candidate strong-normalization route, closed-substitution window scope, finite candidate strong-normalization extension, and residual-join induction. The chain exposes exactly the rows $P, K, J, R, S, C, B, O, H, T, G, N$ and does not assert residual-substitution compatibility, full parallel diamond, subject reduction, arbitrary closed Church–Rosser, evaluator equality, quotient equality of terms, or kernel consistency.

Proof. Start with residual-substitution scope and premise preservation, Definition 4238.12 and Definition 4238.23. They keep the premise row P attached to every critical-pair, residual-frontier, closed-checker, and closed-normal read.

Read critical-pair exhaustion and local-join scope through Definition 4238.4 and Definition 4238.31. These routes place the candidate join inside the displayed K, J, R, S, C, B, O rows before a confluence-facing consumer can inspect the packet.

Next apply the normalization-frontier handoff, candidate-mediated strong-normalization socket, and typed-window candidate route, Definition 4238.40, Definition 4238.41, and Definition 4238.42. Together they route every normalization-facing read through the same retained-premise frontier rather than through a global normalization theorem.

Finally use the closed-substitution window scope, finite candidate strong-normalization extension, and residual-join induction, Definition 4238.43, Definition 4238.44, and Definition 4238.46. The structural rows H, T, G, N transport, replay, source, and name the same finite packet, so the scoped-status evidence remains inside the displayed frontier rows and excludes the refused endpoints. □

Theorem 4238.97 (MetaCIC parallel-diamond obligation-status route)**.** *Every obligation-status read for an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the residual-substitution premise P, critical-pair rows K, R, candidate join J, closed checker S, closed-normal fragment C, bounded check B, obstruction row O, transport row H, replay row T, provenance row G, and local name row N visible before any confluence-audit, decidability, or normalization consumer inspects the frontier. The route is a retained-premise obligation surface; it does not discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution compatibility, or kernel consistency.*

Proof. Begin with the seed scope of Definition 4238.95. It fixes the visible frontier rows $P, K, J, R, S, C, B, O, H, T, G$ before any candidate-mediated strong-normalization route can be exported.

Apply the scoped-status route Definition 4238.96. The dependency chain it records already passes through residual-substitution scope, premise preservation, critical-pair exhaustion, local-join scope, normalization-frontier handoff, candidate-mediated strong-normalization, closed-substitution window scope, finite candidate extension, and residual-join induction.

Now read the candidate strong-normalization handoff and residual-join induction in that order. The handoff keeps the candidate join J, closed checker S, closed-normal fragment C, bounded check B, and obstruction row O attached to the same frontier, while the residual-join induction keeps the premise P and the critical-pair rows K, R visible.

Finally use the critical-path retained-premise handoff carried by the displayed transport, replay, provenance, and local naming rows H, T, G, N. These rows identify the same packet for downstream confluence-audit, decidability, and normalization consumers. Since no row exports subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency, the obligation-status route remains the retained-premise surface stated above. □

Theorem 4238.98 (MetaCIC parallel-diamond retained-premise consumer route)**.** *Any normalization, confluence-audit, or decidability consumer citing an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first read the retained premise row P, critical-pair rows K, R, candidate join J, closed checker S, normal fragment C, bounded check B, obstruction row O, and structural rows H, T, G, N. The citation is a finite retained-premise route through those displayed rows; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the seed scope of Definition 4238.95. It fixes $P, K, J, R, S, C, B, O, H, T, G, N$ as the only rows visible before a candidate-mediated strong-normalization route is exported.

Apply the scoped-status route of Definition 4238.96. This keeps the residual-substitution premise P attached while the critical-pair rows K, R, candidate join J, closed checker S, normal fragment C, bounded check B, and obstruction row O are inspected.

Finally use the obligation-status route of Definition 4238.97. It requires the structural rows H, T, G, N before any downstream consumer treats the frontier as usable evidence, and it explicitly leaves residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency outside the packet. □

Theorem 4238.99 (MetaCIC parallel-diamond seed closure)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ be an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet. Any candidate-mediated strong-normalization handoff from F must first expose the critical-pair envelope, the residual-substitution dependency route, and the candidate-normalization scope rows of the same packet. More explicitly, the handoff reads K, J, R, S, C, B, O through the envelope of Definition 4238.50, retains P through the dependency route of Definition 4238.57, and keeps J inside the candidate-normalization scope of Definition 4238.62. The structural rows H, T, G, N preserve that same packet by* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming. The conclusion is only seed-level row visibility before the SN handoff; it proves no global normalization theorem, no unrestricted confluence theorem, no subject reduction theorem, no evaluator equality, no quotient term equality, and no kernel-consistency certificate.*

Proof. Begin with the critical-pair envelope. By Definition 4238.50, the packet exposes critical-pair requests only through the finite route $P, K, J, R, S, C, B, O, H, T, G, N$.

Next retain the residual-substitution dependency. The route of Definition 4238.57 places P before the local critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows can be consumed.

Now constrain the candidate-normalization read. The scope theorem Definition 4238.62 keeps J inside the same conditional spine and prevents a candidate-mediated consumer from hiding P or O.

Finally compare with the SN handoff of Definition 4238.73. That theorem consumes the same envelope, dependency route, candidate-normalization scope, finite residual witness, and residual-row visibility surfaces. The present theorem isolates the seed-level visibility prerequisite: H, T, G, N preserve the displayed packet, but they do not discharge residual substitution or promote the handoff to global normalization, unrestricted confluence, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.100 (MetaCIC parallel-diamond frontier closure seed route)**.** *Every seed-closure read for an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose the residual-substitution premise row, critical-pair row, candidate-join row, residual-frontier row, closed checker seal, closed-normal fragment, bounded-check row, and obstruction row before any conditional parallel-diamond*

route is consumed. The route is a finite seed surface for the local frontier; it does not prove residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. The residual-substitution premise, critical-pair data, candidate joins, residual frontier, closed checker, closed-normal fragment, bounded-check row, and obstruction row are displayed carrier coordinates before a confluence consumer can read any diamond-shaped conclusion.

Apply Definition 4238.2. The obligation surface requires carrier admission, row visibility, transport, replay, provenance, and local naming for exactly those finite frontier rows.

Use Definition 4238.3 and Definition 4238.50. These keep the conditional handoff downstream of the displayed critical-pair and residual rows, and they prevent the handoff from replacing the obstruction budget by a global confluence theorem.

Finally apply Definition 4238.6. The consumed route is therefore finite and seed-local: it records the displayed obstruction frontier and does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.101 (MetaCIC parallel-diamond frontier closure seed current route)**.** *The seed marker of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is readable only through the carrier of Definition 4238.1, the obligation surface of Definition 4238.2, and the closurestatus guard of Definition 4238.180. A status consumer must therefore expose the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before it may treat the seed registration as available. The route does not claim subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the carrier of Definition 4238.1. Its displayed coordinates contain the residual-substitution premise, critical-pair data, candidate joins, bounded checker, obstruction row, and structural rows that any seed reader must expose.

Apply the obligation surface of Definition 4238.2. It makes the visible rows the local NameCert inventory for the packet, so the seed marker cannot be consumed as an independent confluence theorem.

Finally read the closurestatus guard of Definition 4238.180. The guard routes every status reader through the same finite packet before seed or bridged markers are consumed, and the packet has no coordinate for subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.102 (MetaCIC parallel-diamond frontier closure route)**.** *The seed-facing* MetaCICParallelDiamondF *closure route is exactly the finite frontier route already recorded by the local residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows. It is read through the retained premise boundary of Definition 4238.155, the seed closure route of Definition 4238.189, and the seed-status block of Definition 4238.205.*

The route records a finite seed surface only. It does not assert global confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, unrestricted normalization, or kernel consistency.

Proof. Start with the retained premise boundary. It keeps the residual-substitution premise, critical-pair, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, prove-

nance, and local naming rows visible whenever the candidate join and residual frontier are consumed.

Next read the seed closure route. That theorem attaches the same local rows to the critical-pair envelope, residual-join locality route, retained-premise budget, local NameCert obligations, seed-status row, closurestatus export, and closurestatus guard. Thus the child ledger has a single ordered seed route rather than separate hidden assumptions.

Finally apply the seed-status block. It identifies the seed-facing status read with the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$, so the chapter's closure surface records only the finite frontier rows already available in the local packet.

Because every cited step remains inside the displayed route, none of the refused global endpoints is exported by the closure record. □

Theorem 4238.103 (MetaCIC parallel-diamond scoped seed package route)**.** *The hub-level scoped marker for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is read through the same finite seed package exposed by the spine: residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed-checker rows, closed-normal fragment rows, bounded-check rows, obstruction rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The route records the seed-status package already scoped by Definition 4238.48; it does not add subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, or kernel consistency.*

Proof. Read the scoped closure status theorem in the spine. It identifies the accepted packet by the finite route through $P, K, J, R, S, C, B, O, H, T, G, N$ and refuses the global confluence and kernel endpoints.

The hub marker is therefore a route into that same package rather than a second closure claim. It first records the residual-substitution premise, critical-pair, candidate, residual-frontier, checker, normal-fragment, bounded-check, and obstruction rows.

Finally the structural rows transport, replay, source, and name the packet. Thus the scoped marker is accounted for by a concrete child ledger while the top-level file remains structural. □

Theorem 4238.104 (MetaCIC parallel-diamond frontier residual-join locality)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *residual join is read through the local frontier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The read must first expose the critical-pair envelope of Definition 4238.50, the residual-substitution dependency route of Definition 4238.57, the candidate-normalization scope of Definition 4238.62, the finite residual witness consumption row of Definition 4238.69, and the carrier-local NameCert *row of Definition 4238.2. It exports only a bounded residual-join locality certificate for the displayed packet, not global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Lean4 ✓ `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierResidualJoinLocalit`

Proof. Project the accepted frontier carrier of Definition 4238.1. The only rows available to a residual-join consumer are the finite packet rows $P, K, J, R, S, C, B, O, H, T, G, N$.

Read the critical-pair and residual-substitution rows first. The critical-pair envelope keeps K, J, R, S, C, B, O visible, while the dependency route places the retained residual-substitution premise P before any local join can be consumed.

Next read candidate normalization and finite residual witnesses. The candidate-normalization scope keeps J inside the same conditional spine, and the finite residual witness theorem consumes typed windows, residual frontier data, bounded checker rows, transport, replay, provenance, and local naming before a closed parallel-diamond route can inspect the join.

Finally use the local NameCert obligation surface. Since H, T, G, N preserve only this finite packet and the obstruction row O remains visible, the residual join is local to the displayed frontier and cannot be promoted to global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.105 (MetaCIC parallel-diamond residual-join locality budget)**.** *Every accepted residual-join locality read for a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet keeps the residual premise row P, critical-pair row K, candidate-join row J, residual witness row R, typed-window row S, bounded checker row B, obstruction row O, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N inside the displayed finite packet before any normalization or confluence consumer reads it. The budget is*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

and it exports no residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality, Definition 4238.104. It already forces every join read through the displayed frontier rows and keeps the retained premise and obstruction rows visible.

Next use finite residual witness consumption, Definition 4238.69. That route requires the typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming to be exposed before a local join is available to a downstream consumer.

The two routes combine as a finite budget. Rows P, K, J, R, S, C, B, O remain the residual, critical-pair, candidate-join, witness, window, certificate, checker, and obstruction surface, while H, T, G, N preserve the same packet under transport, replay, provenance, and local naming.

Since the carrier contains no coordinate for residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency, a normalization or confluence consumer can read only the displayed finite budget. □

Theorem 4238.106 (MetaCIC parallel-diamond residual-substitution window)**.** *Every candidate-mediated parallel-diamond read that invokes residual substitution must first display the critical-pair envelope, residual-substitution dependency row, finite residual witness, residual-join locality row, and candidate-normalization row before any strong-normalization or confluence-facing handoff is public. The public window is the carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not a hidden substitution oracle, unrestricted confluence theorem, host conversion equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with the local residual-join route of Definition 4238.104. It forces the accepted read through the displayed frontier rows and keeps the obstruction row visible.

Read the critical-pair and residual-substitution rows next. The critical-pair envelope of Definition 4238.50 exposes the bounded overlap data, while the residual-substitution dependency route of Definition 4238.57 keeps the premise row P in the carrier-local window.

Now attach the finite residual witness and candidate-normalization scope rows. The finite residual witness consumption theorem keeps the typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming visible, and Definition 4238.62 keeps the candidate join inside the same conditional spine.

These rows compose to exactly the displayed residual-substitution window. Since the accepted packet contains no coordinate for a substitution oracle, unrestricted confluence theorem, host conversion equality, subject reduction, evaluator equality, quotient term equality, or kernel consistency, no such handoff is exported. □

Theorem 4238.107 (MetaCIC parallel-diamond residual grid)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *residual-join consumer receives a finite grid whose rows are the critical-pair envelope, residual-substitution dependency, candidate-normalization scope, finite residual witness, carrier-local* NameCert*,* hsame *transport,* Cont *replay, and* Pkg *provenance rows before any local diamond handoff is exported. The grid is the displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not a global confluence theorem, strong-normalization theorem, subject-reduction theorem, quotient term equality, or evaluator equality.

Proof. Begin with Definition 4238.104. It already forces a residual-join read through $P, K, J, R, S, C, B, O, H, T, G,$ and refuses to promote the local route to global confluence or unbounded normalization.

Now refine the local route into grid rows. The critical-pair envelope of Definition 4238.50 supplies the bounded overlap row, the residual-substitution dependency route of Definition 4238.57 places P before any join handoff, the candidate-normalization scope of Definition 4238.62 keeps J inside the conditional spine, and finite residual witness consumption from Definition 4238.69 supplies the typed-window and bounded-check rows.

Finally attach the carrier-local NameCert row and structural rows. The obligation theorem Definition 4238.2 supplies the local naming surface, while H, T, G, N transport, replay, source, and name the same carrier. Since every grid row is finite and carrier-local, the exported handoff cannot become strong normalization, subject reduction, quotient term equality, evaluator equality, or kernel consistency. □

Theorem 4238.108 (MetaCIC parallel-diamond residual-join locality scope)**.** *The residual-join locality surface of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is scoped exactly by the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The rows P, K, J, R, S, C, B, O are the BHist*-visible confluence-frontier rows for residual premises, critical pairs, local join requests, typed windows, residual witnesses, bounded checker data, boundary certificates, and visible obstructions. The rows H, T, G, N are precisely the* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows for that same packet. No residual-join locality read exports global confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, or premise-free residual substitution.*

Proof. Begin with Definition 4238.104. It already forces every residual-join read through the displayed frontier rows and keeps the obstruction row visible.

Refine the route by Definition 4238.107. The grid separates the confluence-facing rows P, K, J, R, S, C, B, O from the structural rows H, T, G, N, while keeping both groups inside one finite packet.

Now apply the seed obligations and carrier obligations. The critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness consumption, and carrier-local NameCert obligation theorem provide no extra coordinate beyond the rows displayed above.

Therefore a consumer that has read $P, K, J, R, S, C, B, O, H, T, G, N$ has exhausted the residual-join locality surface. A request for global confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, or premise-free residual substitution would require a carrier coordinate that the packet does not contain. □

Theorem 4238.109 (MetaCIC parallel-diamond scoped residual locality)**.** *Any accepted residual join in a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose, before a local diamond read is public, the residual-substitution premise row P, the critical-pair row K, the candidate-join row J, the residual-frontier row R, the bounded checker row B, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and the local* NameCert *row N. The read is a scoped residual-locality certificate over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These force the accepted join to pass through the displayed residual-substitution premise, critical-pair envelope, candidate join, finite residual witness, bounded checker, transport, replay, provenance, and local naming rows.

Next use the locality-scope theorem, Definition 4238.108. It identifies the residual-frontier and obstruction rows as part of the same finite carrier-local surface rather than as hidden global assumptions.

Finally apply finite residual witness consumption and residual-join witness locality, Definition 4238.69 and Definition 4238.116. They make the bounded checker rows visible before the local join can be exported and keep P and O inside the structural rows H, T, G, N.

Since the carrier contains only the displayed finite rows, the public read remains a local diamond certificate. Full Church–Rosser, subject reduction, evaluator equality, quotient term equality, and kernel consistency would require coordinates not present in the packet. □

Theorem 4238.110 (MetaCIC parallel-diamond residual-join locality exhaustion)**.** *A residual-join consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that has read*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

has exhausted the carrier-local join surface. Any request for global confluence, subject reduction, unbounded normalization, evaluator equality, quotient term equality, or kernel consistency is outside the packet.

Proof. Apply Definition 4238.104. It forces every residual-join read through the displayed local frontier route and keeps the obstruction row visible.

Then apply Definition 4238.107. The grid identifies the critical-pair, residual-substitution, candidate-normalization, finite-witness, transport, replay, provenance, and local naming rows as the whole carrier-local residual surface.

Finally use the carrier-local NameCert obligation theorem Definition 4238.2. Since that obligation surface admits no coordinate for global confluence, subject reduction, unbounded normalization, evaluator equality, quotient term equality, or kernel consistency, a consumer that has read the grid has no further join row to project. □

Theorem 4238.111 (MetaCIC parallel-diamond residual obstruction budget)**.** *Any candidate-mediated residual join in an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that consumes the local join row J and bounded checker rows must keep both the retained premise row P and the obstruction row O visible through the structural rows H, T, G, N. The handoff is only a bounded residual-join locality budget for*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It cannot be read as residual-substitution discharge, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality, Definition 4238.104. It forces every candidate-mediated join to read the displayed frontier rows and keeps the obstruction coordinate inside the same packet.

Use the residual grid theorem, Definition 4238.107. The grid separates the retained premise, critical-pair, local join, residual witness, bounded checker, and obstruction rows from the transport, replay, provenance, and local naming rows.

Now apply the locality-scope theorem, Definition 4238.108. It identifies P and O as visible rows of the residual-join surface rather than side conditions that can be discharged after J is consumed.

Finally use residual-join locality exhaustion. Once the consumer has read $P, K, J, R, S, C, B, O, H, T, G, N$, there is no further coordinate for residual-substitution discharge, subject reduction, confluence, normalization, evaluator equality, quotient term equality, or kernel consistency. The obstruction budget is therefore retained as a carrier-local boundary. □

Theorem 4238.112 (MetaCIC parallel-diamond residual-premise exhaustion)**.** *Any accepted residual-join read of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that has consumed*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

has not discharged the retained premise row P. It has only made P visible as part of the finite carrier-local grid for the residual join. Consequently the read cannot be used as premise-free residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These place every accepted join inside the displayed carrier-local rows and identify P as a row of that grid, not as a premise erased by the join handoff.

Apply the locality-scope theorem and the residual obstruction budget, Definition 4238.108 and Definition 4238.111. They keep the retained premise row and obstruction row visible through H, T, G, N after the local join and bounded checker rows have been read.

Finally use residual-join witness locality, Definition 4238.116. The witness consumes finite residual data before exporting the local join, but it does not add any coordinate for premise-free residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency. The consumed packet therefore exhausts visibility of the retained premise rather than discharging it. □

Theorem 4238.113 (MetaCIC parallel-diamond residual-premise transport exhaustion)**.** *An accepted residual-join read of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet transports retained premises only through the displayed residual locality route, candidate-normalization scope, finite residual witness consumption rows, and critical-pair envelope rows. After the read has consumed*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the transported premise surface is exhausted. No consumer can extend the read to residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global confluence, strong normalization, or kernel consistency.

Proof. Project the accepted carrier through the residual-premise exhaustion theorem, Definition 4238.112. It identifies P as a retained premise row inside the finite grid rather than as a discharged assumption.

Next read the locality and normalization rows. Residual-join locality, Definition 4238.104, keeps the premise inside the displayed frontier route, while candidate-normalization scope from Definition 4238.62 keeps the candidate join in the same conditional spine.

Now consume the finite witness and critical-pair rows. The finite residual witness theorem Definition 4238.69 exposes the bounded residual data before the local join is available, and the critical-pair envelope of Definition 4238.50 keeps the overlap rows inside the same carrier-local packet.

Finally inspect the carrier rows that remain after transport through H, T, G, N. They replay and name only the displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$. Since the carrier has no coordinate for residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global confluence, strong normalization, or kernel consistency, the premise transport is exhausted at that boundary. □

Theorem 4238.114 (MetaCIC parallel-diamond frontier residual local confluence)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If the bounded checker rows accept both residual branches from the displayed critical-pair and residual-premise surface, then the consumer receives a carrier-local diamond handoff between the two residual endpoints through the displayed candidate join row J. The handoff is bounded by the same packet rows and does not claim global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. First read the residual grid of Definition 4238.107. It exposes the critical-pair envelope, residual-substitution dependency, candidate-normalization scope, finite residual witness, local NameCert row, transport, replay, and provenance as one finite packet.

Next apply residual-premise transport exhaustion, Definition 4238.113. It keeps the transported premise surface inside $P, K, J, R, S, C, B, O, H, T, G, N$ after both residual branches have been checked.

Now consume the finite witness and checker rows using Definition 4238.69. The bounded witness exposes the two residual endpoints before the candidate join row can be handed to a consumer.

Finally apply residual-join witness locality, Definition 4238.116. It keeps J carrier-local, retains P and O through H, T, G, N, and prevents the bounded handoff from becoming global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.115 (MetaCIC parallel-diamond candidate-mediated strong-normalization extension)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the residual rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Suppose a finite candidate packet reads its residual-premise row through Definition 4238.112 and its residual-join row through Definition 4238.104. Then the packet extends along the candidate-mediated frontier without leaving the same finite residual budget. A strong-normalization-facing handoff may consume the local diamond row J together with $P, K, R, S, C, B, O, H, T, G, N$ rather than selecting global normal forms.

Proof. First read residual-premise exhaustion. It keeps P as a retained premise row inside the displayed residual grid, so the candidate packet cannot erase the premise before the strong-normalization-facing handoff is formed.

Next apply residual-join locality and residual-premise transport exhaustion. These rows keep the candidate join J, residual frontier rows, checker rows, and obstruction row inside the same packet after transport through H, T, G, N.

Now consume the finite residual witness using Definition 4238.69. The witness exposes the bounded residual data and checker rows before the local diamond can be exported.

Finally apply residual local confluence, Definition 4238.114. The local diamond row is already a carrier-local handoff over the finite residual budget, so a strong-normalization-facing consumer receives that displayed row rather than a selected global normal form. □

Theorem 4238.116 (MetaCIC parallel-diamond residual-join witness locality)**.** *Every accepted residual-join witness for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must read the finite residual witness row and the bounded checker rows before exporting the local join row. The read retains both the premise row P and the obstruction row O through the structural rows H, T, G, N, so its carrier surface is still*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It does not discharge residual substitution, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency.

Proof. First apply finite residual witness consumption, Definition 4238.69. The witness read is forced through typed windows, residual witness data, bounded checker rows, transport, replay, provenance, and local naming before any local join row is exported.

Next apply residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These results place the join in the displayed finite route and keep the premise row P inside the same carrier-local grid as the critical-pair, witness, checker, and obstruction rows.

Finally apply the residual obstruction budget, Definition 4238.111. It keeps O visible through H, T, G, N and prevents the local join witness from being read as residual-substitution discharge, subject reduction, global confluence, strong normalization, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.117 (MetaCIC parallel-diamond residual-join locality budget)**.** *Every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *residual-join consumer must expose the critical-pair envelope, residual-substitution dependency row, candidate-normalization scope, finite residual witness, and local join rows before any parallel-diamond handoff is public. The exposed packet is the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the premise dependency, K carrying the critical-pair envelope, J naming the local join request, R, S, C, B carrying the residual witness and bounded checker rows, O keeping the visible obstruction row, and H, T, G, N supplying transport, replay, provenance, and local naming.

The budget exports only carrier-local residual-join readability for this displayed frontier. It exports no global confluence theorem, strong-normalization theorem, unrestricted conversion decision, subject-reduction theorem, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the existing hub route order. The critical-pair envelope is introduced before residual-substitution dependency, so any consumer that reaches a residual join has already read the bounded overlap data and the retained premise row.

Next read the candidate-normalization scope and finite residual witness files. Their rows keep the candidate join inside the displayed conditional spine and require the residual witness and bounded checker data before the local join can be consumed.

Finally apply residual-join locality and its grid, scope, exhaustion, obstruction-budget, and witness-locality refinements. Those child files identify $P, K, J, R, S, C, B, O, H, T, G, N$ as the complete local surface and keep the obstruction row visible. Since no child in this route exports a global confluence, normalization, conversion, subject-reduction, evaluator, quotient, substitution-discharge, or consistency coordinate, the handoff remains a carrier-local budget. □

Theorem 4238.118 (MetaCIC parallel-diamond frontier residual budget handoff)**.** *Every candidate-normalization or confluence-audit consumer of the local residual-join budget must keep the critical-pair envelope, residual-substitution dependency, obstruction row, and bounded normalization window visible before reading the candidate-mediated diamond. The handoff surface is the same displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the residual-substitution premise, K carrying the critical-pair envelope, O retaining the refused endpoint row, and B, S, C carrying the bounded checker and normalization window.

The handoff exports only this local residual budget. It does not claim global strong normalization, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Start with the residual-join locality budget of Definition 4238.117. It already requires the critical-pair envelope, residual-substitution dependency row, finite residual witness, obstruction row, and bounded checker rows to remain visible before the residual join can be consumed.

Next use premise retention, Definition 4238.13. The retained premise row P stays attached to the candidate-mediated handoff, so a normalization-facing consumer cannot silently replace residual substitution by a discharged global theorem.

Finally apply the critical-path retained-premise handoff, Definition 4238.85. It keeps P and O visible when the route is read by confluence-audit and normalization consumers. The bounded window rows B, S, C therefore remain part of the exported packet, and the refused global endpoints remain outside the handoff. □

Theorem 4238.119 (MetaCIC parallel-diamond residual-join locality consumption)**.** *Any candidate-mediated parallel-diamond consumer of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must read the critical-pair envelope, residual-substitution dependency, residual-join locality rows, candidate normalization scope, and finite residual witness ledger before consuming the frontier* $\mathsf{NameCert}$ *row. The consumed route is the displayed carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P retains the residual-substitution premise, K, J, R expose the critical-pair and candidate rows, S, C, B, O expose the typed window, residual witness, bounded checker, and obstruction rows, and H, T, G, N provide transport, replay, provenance, and local naming. The consumer receives only this finite residual-locality budget, not global confluence, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.

Proof. First read the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before any candidate-mediated join can be consumed.

Next read the residual-substitution dependency route of Definition 4238.57. This keeps the retained premise row P inside the same finite packet while the candidate join and residual frontier rows are opened.

Now consume the residual-join locality rows of Definition 4238.104 and the budget of Definition 4238.117. These identify $P, K, J, R, S, C, B, O, H, T, G, N$ as the carrier-local residual surface and keep the obstruction row visible.

Finally attach the candidate-normalization scope of Definition 4238.62 and finite residual witness consumption from Definition 4238.69. The candidate row and bounded witness ledger are therefore read before the frontier NameCert row, so the exported handoff remains the displayed residual-locality budget rather than a global confluence, subject-reduction, evaluator-equality, quotient-equality, substitution-discharge, or consistency theorem. □

Theorem 4238.120 (MetaCIC residual-join candidate seed-status route)**.** *Every seed-status reader for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *that consumes residual-join evidence and candidate-normalization evidence must read them inside the same parallel-diamond packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route uses the residual-join locality rows, candidate-normalization scope rows, and parallel-diamond seed-status rows as seedClosure evidence for the hub. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Project the accepted frontier packet. The residual-join rows J, R, S are exposed only with the retained premise row P, critical-pair row K, bounded checker row B, obstruction row O, and structural rows H, T, G, N still visible.

Read the candidate-normalization scope next. It keeps the candidate row inside the same spine $P, K, J, R, S, C, B, O, H, T, G, N$ and prevents the candidate read from becoming a premise-free normalization or confluence theorem.

Finally apply the parallel-diamond seed-status route. The seed status exported to the hub is exactly the displayed packet assembled from residual-join locality, candidate normalization, finite residual witnesses, transport, replay, provenance, and local naming, so the hub records seedClosure evidence without adding theorem content of its own. □

Theorem 4238.121 (MetaCIC parallel-diamond residual-choice boundary)**.** *Every candidate-mediated parallel-diamond read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet consumes residual choice only as a displayed finite ledger row on the route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The residual-choice row is read after the critical-pair envelope and before the residual-join handoff: the critical-pair rows K, J and residual-substitution premise P expose the finite envelope, the residual ledger keeps the chosen residual windows visible, and the residual join is consumed only through

the same bounded frontier. The boundary does not provide host choice, global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes candidate rows and residual-substitution data as part of one finite frontier route before any parallel-diamond request may be consumed.

Next use the residual-substitution dependency route of Definition 4238.57. That route keeps the retained premise P visible before the critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction rows.

Apply finite residual witness consumption from Definition 4238.69. The local residual witness is consumed through $P, K, J, R, S, C, B, O, H, T, G, N$, so the residual-choice row is a displayed finite ledger entry rather than an external selector.

Finally attach the retained-premise boundary of Definition 4238.158. It keeps the premise, candidate-normalization scope, residual-substitution route, finite residual witness, transport, replay, provenance, and local naming visible before the residual join. Since no cited row stores host choice, global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the residual-choice boundary. □

Definition 4238.122 (MetaCIC finite residual-diamond ledger carrier)**.** *A finite residual-diamond ledger for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is a carrier-local subpacket*

$$\Lambda = (K, P, J, B, O, H, T, G, N)$$

read from an accepted frontier packet $F = (P, K, J, R, S, C, B, O, H, T, G, N)$ *with the following displayed rows:*

1. *K is the critical-pair source row naming the bounded local peak under inspection.*
2. *P is the retained residual-substitution premise row attached to that source.*
3. *J is the candidate join witness row, read only as a proposed local join for the displayed peak.*
4. *B is the bounded checker status row for the finite residual diamond request.*
5. *O is the obstruction-retention row recording any still-visible blocker to an unconditional handoff.*
6. *H records componentwise* hsame *transport for the ledger rows.*
7. *T records displayed* Cont *replay for the same finite residual request.*
8. *G is the* Pkg *provenance row for the frontier packet and its sibling confluence consumers.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row naming the ledger as carrier-local data.*

The ledger stores no premise-free confluence theorem, no subject-reduction theorem, no evaluator equality, no quotient term equality, and no kernel-consistency certificate.

Theorem 4238.123 (MetaCIC finite residual-diamond ledger local join obligations)**.** *Every local residual-diamond read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet factors through a finite ledger* $\Lambda = (K, P, J, B, O, H, T, G, N)$ *of Definition 4238.122. The read may cite the ledger as bounded local join evidence under the retained residual-substitution premise and visible obstruction row, but it cannot be cited as an unconditional parallel diamond, full Church–Rosser theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It supplies the bounded source row K and keeps the peak inside the finite frontier packet.

Next read the residual-substitution dependency route of Definition 4238.57. This places the retained premise row P before the candidate join can be consumed, so the ledger cannot erase the premise.

Now use the candidate diamond budget route of Definition 4238.66 together with residual-join locality, Definition 4238.104. These rows expose the candidate join J and bounded checker status B only as a local finite route through the accepted frontier.

Finally apply the retained-premise closure budget of Definition 4238.150. The obstruction row O and structural rows H, T, G, N remain displayed, so the resulting ledger is a bounded residual-diamond obligation surface rather than any of the refused global confluence or kernel endpoints. □

Theorem 4238.124 (MetaCIC frontier seed ledger)**.** *An accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier packet exposes a finite seed ledger*

$$\Sigma = (K, J, R, S, B, O, H, T, G, N)$$

before any residual-join or parallel-diamond read becomes public. The ledger reads the critical-pair row K, candidate join row J, residual-frontier row R, seed-readiness row S, bounded checker row B, obstruction row O, hsame *transport row H,* Cont *replay row T,* Pkg *provenance row G, and local* NameCert *row N through the existing MetaCIC frontier rows. It is only a seed ledger for finite residual-join inspection, not a premise-free confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.*

Proof. Start with the seed package register of Definition 4238.82 and the seed readiness row of Definition 4238.83. These rows expose K, J, R, S, B, O as finite frontier data for a bounded residual-join request.

Next use the residual join seed interface of Definition 4238.130. It keeps the candidate join and residual frontier inside the same seed packet rather than turning them into an unconditional parallel diamond.

Finally apply the structural rows supplied by Definition 4238.103. The rows H, T, G, N transport, replay, package, and name the same finite seed ledger, so public consumers receive only the displayed residual-join ledger and none of the refused global endpoints. □

Theorem 4238.125 (MetaCIC parallel-diamond frontier residual scope)**.** *Every residual-scope consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the finite rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The consumer may read residual-join locality, the residual grid, locality scope, and the obstruction budget, but it may not discharge residual substitution, subject reduction, global confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Begin with residual-join locality, Definition 4238.104. It forces every residual-join read through the displayed frontier rows and keeps the packet local to the accepted carrier.

Refine the consumer route through the residual grid and locality scope, Definition 4238.107 and Definition 4238.108. These theorems separate the residual-facing rows P, K, J, R, S, C, B, O from the structural rows H, T, G, N while keeping both groups inside one finite packet.

Finally apply the residual obstruction budget, Definition 4238.111. It keeps the retained premise and obstruction rows visible, so a residual-scope consumer has no route to discharge residual substitution or to promote the local frontier to subject reduction, global confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.126 (MetaCIC parallel-diamond residual-normalization window)**.** *Every accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet whose displayed rows expose*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

may hand a candidate-mediated residual peak to a normalization consumer only through the retained residual-substitution premise row P, the candidate-normalization row J, the residual-frontier row R, the closed-checker and closed-normal rows S, C, and the obstruction row O, with the structural rows H, T, G, N still visible. The window does not prove strong normalization, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the carrier of Definition 4238.1. Its public rows expose the residual-substitution premise, critical-pair data, candidate-normalization row, residual-frontier row, closed-checker row, closed-normal row, bounded-check row, obstruction row, transport, replay, provenance, and local naming.

Use the candidate-normalization scope of Definition 4238.62. It forces a normalization-facing read of J to keep P and O visible and to remain inside the same finite spine.

Next consume the finite residual witness and candidate-mediated strong-normalization handoff, Definition 4238.69 and Definition 4238.73. These rows allow a residual peak to be presented to the consumer only after R, S, C and the bounded obstruction data have been displayed.

Finally apply residual-join locality, Definition 4238.104. Since the obstruction row remains retained and no carrier coordinate discharges P or promotes J to a global theorem, the exported object is only the stated residual-normalization window. □

Theorem 4238.127 (MetaCIC parallel-diamond seed boundary)**.** *Every seed-level read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet may expose candidate-mediated residual peaks only while the residual-substitution premise row P, candidate-normalization row J, obstruction row O, closed-checker rows S, C, B, componentwise* hsame *transport H, displayed* Cont *replay T,* Pkg *provenance G, and local* NameCert *row N remain visible in the same finite carrier. The boundary is a local seed surface for residual normalization; it is not strong normalization, unrestricted confluence, quotient term equality, evaluator equality, subject reduction, residual-substitution discharge, or kernel consistency.*

Proof. Apply the residual-normalization window theorem, Definition 4238.126. It already places every normalization-facing residual peak behind the retained residual-substitution premise, candidate-normalization row, residual-frontier row, closed checker and closed-normal rows, obstruction row, and structural rows.

Next use the candidate-normalization scope of Definition 4238.62 and the candidate-mediated strong-normalization handoff of Definition 4238.73. These prevent the candidate row from becoming a premise-free normalization theorem and keep the candidate-mediated route inside the displayed frontier carrier.

Finally read the residual-substitution dependency route and obstruction ledgers. The premise row P and obstruction row O remain visible through hsame transport, Cont replay, Pkg provenance, and the local NameCert row. Since no listed row discharges those boundaries, the exposed seed remains only the stated local residual-normalization surface. □

Theorem 4238.128 (MetaCIC parallel-diamond residual-normalization scope)**.** *Any residual-join read from an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must consume the critical-pair envelope, candidate-normalization row, residual-substitution row, finite residual witness, and candidate-SN handoff before it exports a diamond-frontier budget. The exported budget is the finite route*

through

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with the residual-substitution premise and obstruction rows still visible; it is not a premise-free parallel diamond, full normalization theorem, subject-reduction theorem, evaluator equality, quotient equality of terms, or kernel-consistency certificate.

Proof. First read the critical-pair envelope of Definition 4238.50. It fixes the local pair inside the displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ and keeps the bounded overlap surface visible.

Next consume the candidate-normalization scope of Definition 4238.62 together with the residual-substitution dependency route of Definition 4238.57. These rows place J under the retained premise P rather than turning candidate normalization into an unconditional confluence result.

Now apply finite residual witness consumption, Definition 4238.69. The finite witness must read the typed window, residual frontier, substitution-boundary row, checker rows, transport, replay, provenance, and local naming before a closed parallel-diamond route can inspect it.

Finally use the candidate-SN handoff of Definition 4238.73. The handoff packages the same conditional spine as a diamond-frontier budget, so the exported object still retains P and O and cannot be reread as full normalization, subject reduction, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.129 (MetaCIC residual-substitution discharge boundary)**.** *Let $F = (P, K, J, R, S, C, B, O, H, T, G$ be an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet. A residual-normalization read from F may discharge only the local residual-substitution boundary rows already displayed by the packet: the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness, and candidate-SN handoff. The discharge does not export subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the critical-pair envelope of Definition 4238.50. It keeps the local overlap inside the finite rows $P, K, J, R, S, C, B, O, H, T, G, N$ and prevents the read from erasing the retained residual-substitution premise.

Next combine the candidate-normalization scope of Definition 4238.62 with the residual-substitution dependency route of Definition 4238.57. These two rows allow the candidate read to discharge only the displayed local boundary, with P still visible as the premise row for the residual route.

Now consume the finite residual witness from Definition 4238.69. The witness is finite data in the same packet, so it can certify the local residual-normalization handoff without becoming a global equality or subject-reduction theorem.

Finally apply the candidate-SN handoff of Definition 4238.73. That handoff packages the preceding rows as a candidate-mediated boundary discharge, and none of the six refused exports appears as a coordinate of the carrier. □

Theorem 4238.130 (MetaCIC residual-join seed interface)**.** *Every seed-level diamond frontier consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that reads residual joins must first display the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness, and candidate strong-normalization handoff. Concretely, the read must pass through Definition 4238.50, Definition 4238.62, Definition 4238.57, Definition 4238.69, and Definition 4238.73 before the packet is available as a seed interface. The exported seed read is therefore a displayed residual-join interface over the finite frontier spine, not a premise-free confluence theorem, global normalization theorem, subject-reduction theorem, host reduction equality, or kernel-consistency certificate.*

Proof. Start with the critical-pair envelope. The envelope theorem keeps the rows $P, K, J, R, S, C, B, O, H, T, G, N$ visible before a parallel-diamond request can be read, so the seed consumer receives a bounded frontier packet rather than an unrestricted confluence surface.

Next read the candidate-normalization and residual-substitution routes. The candidate-normalization scope keeps J inside the conditional spine, while the residual-substitution dependency route keeps the retained premise and residual windows attached to the same packet.

Now expose the finite residual witness and the candidate strong-normalization handoff. The finite witness theorem supplies the bounded residual rows consumed by the frontier, and the candidate handoff states the normalization-facing read available from those same rows.

Thus any seed-level residual-join consumer has displayed the five interface faces before reading the packet. Since none of those faces exports premise-free confluence, global normalization, subject reduction, host reduction equality, or kernel consistency, the seed interface remains exactly the finite residual-join route stated above. □

Theorem 4238.131 (MetaCIC residual-join seed scope)**.** *Every residual-join seed read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet factors through residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed-checker rows, obstruction rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. The seed read is therefore a finite residual-join scope for the parallel-diamond frontier, not a subject-reduction theorem, arbitrary Church–Rosser theorem, quotient term equality, or evaluator equality row.*

Proof. Start with the residual-substitution dependency route and the critical-pair envelope. The dependency route keeps the retained substitution premise visible, while Definition 4238.50 exposes the critical-pair rows before any diamond request is read.

Next read the candidate diamond budget route. It displays the candidate-join rows and keeps them tied to the same finite frontier rather than to an unrestricted Church–Rosser surface.

Now use the finite residual witness consumption route. The residual-frontier and closed-checker rows are consumed only after the finite witness rows have been displayed, so the obstruction rows remain part of the same seed scope.

The structural rows supply componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Since the cited routes contain no subject-reduction, arbitrary Church–Rosser, quotient term equality, or evaluator equality coordinate, the residual-join seed read exports none of those endpoints. □

Theorem 4238.132 (MetaCIC residual-join seed-status record)**.** *The residual-join seed status of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is available to a confluence-facing consumer only after the consumer has read the critical-pair envelope, candidate-normalization scope, finite residual witness, and retained-premise rows of the displayed frontier packet. Equivalently, the route passes through Definition 4238.50, Definition 4238.62, Definition 4238.69, and Definition 4238.159 before any confluence consumer may inspect the residual-join seed record. The record exports only this bounded seed route; it does not supply premise-free confluence, global normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the critical-pair envelope first. It keeps the frontier rows visible before any residual-join request is interpreted as a seed-status claim.

Next apply the candidate-normalization scope and finite residual witness. These rows confine the candidate read and the residual read to the same finite frontier packet.

Finally read the retained-premise residual-join scope. Since that row remains present until the seed record is consumed, the exported status is a bounded residual-join route rather than a detached confluence theorem. □

Theorem 4238.133 (Residual-join candidate dependency boundary)**.** *Every residual-join candidate read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness consumption, and retained-premise boundary before it can be consumed as a local diamond frontier. In carrier form the read remains inside*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K *displays the critical-pair envelope,* P *retains the residual-substitution dependency,* J *records the candidate-normalization row,* R, S, C, B, O *keep the residual frontier, typed window, witness, bounded checker, and obstruction rows visible, and* H, T, G, N *provide transport, replay, provenance, and local naming. The boundary exports only this local residual-join candidate surface; it explicitly refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before a residual-join candidate can be read by a frontier consumer.

Next read the residual-substitution dependency route of Definition 4238.57. This keeps the retained premise row P visible while the candidate, residual, checker, and obstruction rows are opened.

Now apply candidate-normalization scope through Definition 4238.62. The candidate row J remains part of the displayed frontier packet rather than a detached normalization theorem.

Consume the finite residual witness with Definition 4238.69. The residual rows R, S, C, B, O remain tied to the same finite witness route and cannot be reread as a global local-confluence theorem.

Finally use the retained-premise boundary of Definition 4238.158. It keeps $P, K, J, R, S, C, B, O, H, T, G, N$ visible before any local diamond frontier consumes the candidate, so the refused endpoints remain outside the packet. □

Theorem 4238.134 (MetaCIC candidate-mediated residual-join scope)**.** *Every candidate-mediated residual join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual join locality rows before it can be read as a parallel-diamond frontier packet. In carrier form the read remains on the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P *retains the residual-substitution premise,* K *exposes the critical-pair envelope,* J *records the candidate-normalization row,* R, S, C, B, O *keep the residual frontier, seed, witness, bounded checker, and obstruction rows visible, and* H, T, G, N *provide transport, replay, provenance, and local naming. The scope exports only this finite candidate-mediated handoff, not global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.*

Proof. Start with the critical-pair envelope of Definition 4238.50. It exposes the bounded overlap rows before any residual join is available to a frontier consumer.

Next read candidate-normalization scope through Definition 4238.62. The candidate row J is therefore part of the displayed packet rather than an external normalization theorem.

Now use the residual-substitution dependency route of Definition 4238.57. It keeps the retained premise row P load-bearing while the residual rows R, S, C, B, O are opened.

Finally consume finite residual witness data and residual join locality via Definition 4238.69 and Definition 4238.104. These rows keep the local join inside $P, K, J, R, S, C, B, O, H, T, G, N$, so the resulting frontier packet provides only the stated candidate-mediated residual-join scope and none of the refused global endpoints. □

Theorem 4238.135 (MetaCIC candidate-mediated residual window)**.** *Every accepted residual join read from a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the retained premise row P, candidate-normalization row J, residual-substitution route, finite residual witness rows, and residual-join locality rows inside the same bounded window before any parallel-diamond conclusion is visible. In carrier form the window is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P remains the retained residual-substitution premise, J remains the candidate-normalization row, R, S, C, B, O carry the residual frontier, typed window, finite witness, bounded checker, and obstruction data, and H, T, G, N transport, replay, source, and name the same packet. The read is therefore premise-bearing local residual evidence, not a premise-free parallel-diamond conclusion, unrestricted confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the candidate-mediated residual-join scope of Definition 4238.134. It already requires the critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual-join locality rows to be displayed before a frontier consumer reads the handoff.

Next apply the residual-substitution dependency route of Definition 4238.57. This keeps P load-bearing while the residual rows are opened, so the bounded window cannot be reread as a premise-free substitution result.

Finally combine finite residual witness consumption with residual-join locality, using Definition 4238.69 and Definition 4238.104. These rows keep R, S, C, B, O attached to the same candidate-normalization route through J and the same structural rows H, T, G, N. Thus any visible parallel-diamond conclusion remains inside the retained bounded window named in the statement. □

Theorem 4238.136 (MetaCIC candidate-mediated residual-join obligation)**.** *Every accepted candidate-normalization window of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that exposes both residual branches must also expose the finite local-join row, retained-premise ledger, and obstruction-retention row before a parallel-diamond frontier consumer may read the window. In carrier form the admissible obligation surface is*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where K is the critical-pair envelope, J is the candidate-local join row, P is the retained residual-substitution premise ledger, R, S, C, B display the residual branch, typed-window, witness, and bounded-checker rows, O records obstruction retention, and H, T, G, N preserve transport, replay, provenance, and local naming. The obligation is local to this finite surface; it does not assert global confluence, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, strong normalization, or kernel consistency.

Proof. Project the candidate-mediated residual window of Definition 4238.135. It already keeps $P, J, R, S, C, B, O, H, T, G, N$ visible while the candidate-normalization row is consumed.

Read the critical-pair envelope through Definition 4238.50. Both residual branches are therefore branches of the displayed finite envelope K, not independent reductions imported from an ambient calculus.

Apply residual-window scope using Definition 4238.134. The local join row J, retained premise row P, residual rows R, S, C, B, and obstruction row O remain on the same candidate-mediated route.

If a branch attempted to bypass J or O, it would no longer be a read of the retained window $P, K, J, R, S, C, B, O, H, T, G, N$. If it attempted to bypass P, the residual-substitution dependency named by the window would be lost. Hence the frontier consumer receives a parallel-diamond-facing obligation only after the local join ledger, retained premise ledger, and obstruction-retention row have all been exposed. □

Theorem 4238.137 (MetaCIC parallel-diamond window-budget scope)**.** *Every accepted* MetaCICParallelDiamondF *frontier read must expose the typed premise row, critical-peak row, bounded join-budget row, residual checker row, candidate strong-normalization handoff row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row before any local-join conclusion is public. In the carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P is the retained typed premise, K is the peak row, J is the candidate join row, R, S, C, B are the finite residual and bounded checker budget rows, O is the visible obstruction row, and H, T, G, N are the structural transport, replay, provenance, and naming rows. The scope exports only this finite window-budget surface, not global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency.

Proof. Start with the critical-pair envelope. It displays the peak and local candidate rows before a diamond request can be inspected by a frontier consumer.

Next read the residual-substitution dependency route and residual-normalization scope. These keep the typed premise row P and obstruction row O visible while the residual checker and bounded join-budget rows are opened.

Use the candidate strong-normalization handoff together with finite residual witness consumption. The handoff can be consumed only after the candidate row, residual witness rows, bounded checker rows, transport, replay, provenance, and local naming rows have been displayed in the same carrier-local spine.

Finally apply residual-row visibility. Since H, T, G, N preserve the displayed finite packet and no carrier coordinate contains global confluence, subject reduction, evaluator equality, quotient term equality, premise-free residual substitution, or kernel consistency, the exported conclusion is exactly the local finite window-budget scope. □

Theorem 4238.138 (MetaCIC parallel-diamond residual-join obligations)**.** *Every frontier residual join exposed by an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must display the critical-pair source row, retained residual-substitution window, candidate-normalization budget, local join witness, obstruction-retention row, componentwise* hsame *transport, displayed* Cont *replay, and* NameCert *provenance before a consumer may read the join. Equivalently, the obligation surface is the carrier-local spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P is still the residual-substitution premise, K, J, R give the critical-pair and candidate frontier rows, S, C, B, O give the bounded checker, closed-normal, budget, and obstruction rows, and H, T, G, N give transport, replay, provenance, and local naming.

The surface is a bounded candidate-mediated residual-join obligation. It is not a parameter-transport equivalence theorem, global confluence theorem, unbounded normalization theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. First read residual-join locality, Definition 4238.104. It places every accepted residual join inside the finite frontier spine and keeps the critical-pair, retained premise, local join, bounded checker, obstruction, transport, replay, provenance, and naming rows visible.

Next apply the residual-scope route of Definition 4238.125. This prevents the retained residual-substitution premise from being discharged after the local join is consumed, and it keeps $P, K, J, R, S, C, B, O, H, T,$ in the same carrier-local packet.

Now use the residual-normalization window and candidate-mediated strong-normalization handoff, Definition 4238.126 and Definition 4238.73. These rows expose the candidate-normalization budget and local witness only after the typed residual window, bounded checker rows, and obstruction row have been displayed.

Finally repack the displayed rows as the residual-join obligation surface. Since the packet contains no coordinate for parameter-transport equivalence, global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency, the exposed join remains a local obligation over the stated finite frontier spine. □

Theorem 4238.139 (MetaCIC residual-join obligation handoff)**.** *Let an accepted* MetaCICParallelDiamondFrontier[1] *packet expose finite critical-pair coverage, residual-substitution locality, and retained-premise closure through the carrier-local rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If a residual-join consumer reads the finite critical-pair surface through K, J, R, keeps the residual-substitution premise P local to the same packet, and preserves the retained premise and obstruction rows through B, O, H, T, G, N, then the packet provides the candidate-mediated strong-normalization handoff surface recorded by Definition 4238.73. The handoff is a bounded residual-join obligation surface; it does not claim global confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join obligations and residual critical-pair coverage, Definition 4238.138 and Definition 4238.140. These results force the consumer to expose the finite critical-pair, candidate, residual witness, checker, obstruction, transport, replay, provenance, and naming rows before any local join is exported.

Next use residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. They keep the residual-substitution premise inside the carrier-local packet and prevent the candidate join from being read as an independent confluence theorem.

Finally apply the retained-premise budget scope and the candidate-mediated SN handoff, Definition 4238.142 and Definition 4238.73. The first keeps P and O visible through the structural rows, and the second packages exactly that displayed finite surface as the candidate-mediated handoff. Since every cited step is still confined to $P, K, J, R, S, C, B, O, H, T, G, N$, no global confluence, unrestricted normalization, subject reduction, evaluator equality, quotient term equality, or kernel-consistency endpoint is exported. □

Theorem 4238.140 (MetaCIC parallel-diamond residual critical-pair coverage)**.** *Every residual-join consumer of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must expose, before exporting its local join, the critical-pair envelope, retained residual-substitution premise, candidate-normalization scope, finite residual witness, bounded checker rows, obstruction row, and local* NameCert *row. Equivalently, the exported local join must still be read through the finite carrier-local surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the substitution premise, K exposing the critical-pair row, J and R retaining the candidate-normalization and residual-witness rows, S, C, B exposing the bounded checker rows, O retaining the obstruction row, and H, T, G, N carrying transport, replay, provenance, and local naming.

The coverage claim is local to the frontier packet. It exports no global confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Begin with the residual-join obligations and residual-normalization envelope above. Together they keep any exported local join behind the retained residual-substitution premise, critical-pair row, candidate-normalization budget, bounded checker rows, obstruction row, transport, replay, provenance, and naming.

Next refine the obligation surface through residual-join locality and the residual grid, Definition 4238.104 and Definition 4238.107. These theorems identify the critical-pair, residual-substitution, candidate-normalization, finite-witness, bounded-checker, obstruction, and local naming rows as one carrier-local grid rather than independent side conditions.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. It lets the consumer use the candidate-mediated boundary only after the finite residual witness and bounded checker rows remain visible, so the exported join is covered by the displayed critical-pair surface and cannot be reread as global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.141 (MetaCIC parallel-diamond residual-normalization envelope)**.** *Any consumer that uses the local residual join of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must first expose the retained residual-substitution premise, the residual-normalization window, the candidate strong-normalization handoff, bounded checker rows, the obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

The envelope is the same carrier-local route

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read with P retained and O visible before the candidate-mediated normalization handoff is consumed. It exports no full confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the residual-join obligation surface above. It keeps the critical-pair source, retained residual-substitution premise, candidate-normalization budget, bounded checker rows, obstruction row, transport, replay, provenance, and local naming visible before a local join can be read.

Next apply the residual-normalization window, Definition 4238.126. That row forces a normalization-facing residual peak to remain behind $P, J, R, S, C, O, H, T, G, N$ and prevents the candidate row from becoming a premise-free normalization theorem.

Finally use the candidate strong-normalization handoff, Definition 4238.73. It lets the consumer pass the bounded candidate data onward only after the checker rows and obstruction row are still displayed, so the resulting envelope remains a local residual-normalization boundary rather than full confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.142 (MetaCIC parallel-diamond residual-premise budget scope)**.** *Every candidate-mediated strong-normalization or local residual-join consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must keep the retained residual-substitution premise row P and obstruction row O visible through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before exporting a budgeted join. The budget is the displayed carrier-local budget over residual premise, critical pair, candidate row, residual witness, closed checker, closed-normal fragment, bounded checker, obstruction, transport, replay, provenance, and local naming; it is not a discharge of residual substitution, global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the residual-join obligations and critical-pair coverage, Definition 4238.138 and Definition 4238.140. They require every exported local join to display $P, K, J, R, S, C, B, O, H, T, G, N$, with the residual-substitution premise and obstruction rows still present.

Next apply the residual-normalization envelope, Definition 4238.141. It keeps candidate-mediated normalization behind the same finite route and prevents the candidate row from becoming a premise-free normalization theorem.

Finally use the candidate strong-normalization handoff, Definition 4238.73. The handoff can export only the displayed budgeted join, so any consumer claiming global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency has left the retained residual-premise surface. □

Theorem 4238.143 (MetaCIC parallel-diamond residual-join transport induction)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet display the residual critical-pair rows, residual-normalization envelope, retained-premise budget rows, componentwise* hsame *transport, and displayed* Cont *replay over the finite spine*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

If those rows are transported componentwise and replayed by the displayed continuation data, the resulting local join remains inside the same candidate-mediated diamond surface. In particular, residual critical-pair coverage, residual-normalization envelope data, and retained-premise budget scope are preserved as frontier-local rows, not reread as global confluence, unbounded normalization, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Induct through the finite residual row list. The base row is the retained residual-substitution premise P; transport along the corresponding component of H keeps the premise visible before any critical-pair or candidate row is consumed.

For the critical-pair and local-join segment, use Definition 4238.138 and Definition 4238.140. These theorems keep K, J, R attached to the same carrier-local residual surface under replay by T.

For the normalization and budget segment, apply Definition 4238.141 and Definition 4238.142. The rows S, C, B, O remain visible before the candidate-mediated handoff can export a budgeted join.

The structural rows H, T, G, N then repack the transported list as the same local diamond surface. Since each induction step uses only displayed rows and continuation replay, no quotient term equality, global confluence theorem, unbounded normalization theorem, subject-reduction theorem, evaluator equality, or kernel-consistency certificate is introduced. □

Theorem 4238.144 (MetaCIC retained-premise discharge boundary)**.** *Every candidate-mediated local join exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the retained residual-substitution premise row P through the full visible route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The candidate rows K, J, R, S, C, B may transport a bounded local join, and the structural rows H, T, G, N may replay and name it, but none of those rows discharges P. The boundary therefore exports only a local residual-join read with P retained; it refuses global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, and kernel consistency.

Proof. Start with residual-join transport induction, Definition 4238.143. Its base row is P, so every transported local join begins with the retained residual-substitution premise still visible.

For the candidate-mediated segment, use residual critical-pair coverage and the residual-normalization envelope. These results keep K, J, R, S, C, B attached to the same finite frontier packet and prevent the candidate handoff from becoming a premise-free join theorem.

Finally apply retained-premise budget scope and the candidate strong-normalization handoff. They permit a bounded local join only while P, O, H, T, G, N remain in the route. Hence the exported join preserves the retained premise and reaches none of the refused global endpoints. □

Theorem 4238.145 (MetaCIC obstruction-preservation route)**.** *Every candidate-mediated local join transported through an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet preserves the obstruction row O along the same finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The obstruction remains visible before and after the local join is replayed by T and named by N. Thus the packet records obstruction-preserving local residual-join transport only; it does not assert global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First read the residual-normalization envelope and residual-premise budget scope. Together they place O beside the bounded checker and candidate rows, so the obstruction is not an external side condition that can be erased after the join is found.

Next use residual-join transport induction. The transport row H and replay row T act componentwise on the visible route, so the obstruction component is carried with P, K, J, R, S, C, B rather than replaced by evaluator equality or quotient term equality.

Finally repack through G and N. Provenance and local naming certify the same obstruction-preserving frontier packet, and because no coordinate provides unrestricted normalization, subject reduction, global confluence, or kernel consistency, the route remains exactly the local obstruction-preservation route. □

Theorem 4238.146 (MetaCIC residual-budget diamond obligation)**.** *Every local diamond-join read exposed by an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet must display the critical-pair envelope, retained residual-substitution window, candidate-normalization scope, finite residual-diamond ledger, and residual-join locality rows before the frontier package can expose a candidate-mediated confluence handoff. Equivalently, the handoff remains inside the carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

where P retains the residual-substitution premise, K records the critical-pair envelope, J, R, S, C, B record the candidate and bounded residual window, O records the visible obstruction, and H, T, G, N record transport, replay, provenance, and local naming. The obligation exports only a bounded local diamond route; it does not discharge residual substitution, prove unrestricted confluence, prove subject reduction, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Start with the residual-substitution dependency route, Definition 4238.57. It places the retained premise row P before any critical-pair or candidate-join row can be consumed.

Next read the candidate-normalization scope of Definition 4238.62. That scope keeps J inside the conditional spine and requires the obstruction row O to remain visible when a normalization-facing consumer inspects the candidate.

Use the finite residual-diamond ledger obligations, Definition 4238.123. The ledger packages the critical-pair row, retained premise, candidate join, bounded checker status, obstruction, transport, replay, provenance, and local naming as finite carrier-local data.

Finally apply residual-join locality and its locality budget, Definition 4238.104 and Definition 4238.105. These rows keep the displayed packet on the route $P, K, J, R, S, C, B, O, H, T, G, N$ before any candidate-mediated confluence handoff is public. Since every cited surface is carrier-local and still retains P and O, the handoff remains the stated residual-budget obligation rather than any refused global endpoint. □

Theorem 4238.147 (MetaCIC parallel diamond frontier residual-join obligation carrier)**.** *Before any candidate diamond read of an accepted* MetaCICParallelDiamondFrontier$^{\uparrow}$ *packet is public, the residual-join obligation carrier consists of the retained premise row, critical-pair envelope, residual-substitution dependency row, finite residual witness, obstruction row, bounded checker row, transport, replay, provenance, and local* NameCert *row. In carrier notation the public residual-join obligation must still expose*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with P retaining the premise, K exposing the critical pair, J, R carrying the candidate and finite residual witness, S, C, B retaining the residual-substitution and checker rows, O retaining the obstruction, and H, T, G, N transporting, replaying, sourcing, and naming the same packet. The carrier is an obligation surface for a bounded residual join; it is not full confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. Start with the residual-budget diamond obligation of Definition 4238.146. It already forces every local diamond-join read to expose the critical-pair envelope, retained residual-substitution window, candidate-normalization scope, finite residual-diamond ledger, and residual-join locality rows before a candidate-mediated handoff is public.

Next use finite residual witness consumption, Definition 4238.69. This supplies the finite residual witness, typed window, bounded checker, obstruction, transport, replay, provenance, and local naming rows as displayed data of the same packet.

Finally apply the residual-join obligation handoff and candidate-mediated SN handoff, Definition 4238.139 and Definition 4238.73. They keep P and O visible through H, T, G, N before the candidate-facing read is exported. Since the composed route never leaves $P, K, J, R, S, C, B, O, H, T, G, N$, it yields only the stated residual-join obligation carrier and none of the refused global endpoints. □

Theorem 4238.148 (MetaCIC residual-join frontier obligation)**.** *Every candidate-mediated residual join exported by an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier must keep the retained residual-substitution premise row P and the obstruction row O visible through the same finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before any local diamond handoff is consumed. Thus the consumer may read a bounded local join only while the critical-pair, candidate, finite residual witness, bounded checker, obstruction, transport, replay, provenance, and local naming rows remain carrier-local.

The obligation does not export global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First compose the residual-join obligation carrier of Definition 4238.147 with the residual-budget diamond obligation of Definition 4238.146. The carrier supplies the displayed route, and the budget obligation prevents the candidate diamond read from leaving that route before the handoff is visible.

Next use finite residual witness consumption, Definition 4238.69. This step keeps the residual witness, typed window, bounded checker rows, and obstruction row as displayed rows of the same packet rather than private side conditions.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. Since the handoff consumes only the displayed candidate-mediated surface and the rows P and O are still present through H, T, G, N, the exported join remains a local frontier obligation and none of the refused global endpoints is introduced. □

Theorem 4238.149 (MetaCIC residual-join window determinacy)**.** *Let two accepted candidate-mediated residual-join reads of a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose the same retained premise row P, critical-pair row K, candidate row J, residual witness rows R, S, C, B, obstruction row O, transport rows H, T, provenance row G, and local* $\mathsf{NameCert}$ *row N. Then the two reads determine the same local join window over the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The conclusion is only local window determinacy for the displayed residual-join surface. It is not global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. First compose the residual-join obligation carrier of Definition 4238.147. Both reads expose the same carrier-local rows, so the retained premise, critical-pair row, candidate row, residual witness rows, obstruction row, transport, replay, provenance, and local naming are the same displayed packet.

Next apply the residual-budget diamond obligation of Definition 4238.146. The budgeted local diamond handoff cannot leave the route $P, K, J, R, S, C, B, O, H, T, G, N$, and the shared rows therefore force both reads to consume the same bounded residual window.

Use finite residual witness consumption to keep R, S, C, B as the displayed witness and checker rows of that same window. The shared obstruction row O and shared transport rows H, T prevent either read from replacing the witness by an external evaluator equality or quotient term equality.

Finally apply the candidate strong-normalization handoff of Definition 4238.73. Since the handoff consumes only the shared candidate-mediated surface, the two accepted reads determine one local join window. The argument never exports global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.150 (MetaCIC parallel-diamond retained-premise closure)**.** *A retained-premise read used by a candidate-mediated strong-normalization or local-confluence consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must expose residual-substitution compatibility rows, candidate-normalization rows, local-join rows, the critical-pair envelope, finite residual witnesses, typed-window rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row before it is consumed as a closure budget. The budget is a finite frontier read over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the critical-pair envelope and candidate-normalization scope, Definition 4238.50 and Definition 4238.62. They keep $P, K, J, R, S, C, B, O, H, T, G, N$ visible before candidate-mediated normalization can read the candidate row.

Apply residual-join locality and finite residual witness consumption, Definition 4238.104 and Definition 4238.69. These force the retained premise, typed windows, residual frontier, local-join checker, finite witness, transport, replay, provenance, and local naming rows to be consumed as one finite packet.

Finally read the retained-premise mature budget, Definition 4238.154. It allows the premise to serve as a closure budget only while the obstruction and residual-substitution rows remain displayed, so the read cannot become subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.151 (MetaCIC parallel-diamond retained-premise induction)**.** *Let an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet expose a finite list of retained-premise reads to a candidate-mediated strong-normalization or local-confluence consumer. If each read is admitted by the retained-premise closure budget of Definition 4238.150, then induction on the list preserves the same displayed rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through every residual-pair read. The iterated budget remains only a finite frontier over those rows, and cannot discharge subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. For the base list, read the retained-premise closure theorem. It supplies one closure-budget surface over $P, K, J, R, S, C, B, O, H, T, G, N$, and the empty residual-pair suffix contributes no additional semantic row.

For the cons step, expose the head retained-premise read through the same closure-budget theorem and apply the induction hypothesis to the remaining finite list. The head read and the tail budget both project the same transport, replay, provenance, obstruction, typed-window, residual-frontier, candidate, local-join, and naming rows, so their composition still displays exactly $P, K, J, R, S, C, B, O, H, T, G, N$.

Finally project the negative boundary attached to each step. Since neither the head read nor the inductive tail contains a row for subject reduction, unrestricted confluence, evaluator equality,

quotient term equality, global normalization, or kernel consistency, the finite composition cannot acquire any of those conclusions. □

Theorem 4238.152 (MetaCIC candidate-normalization retained premise)**.** *Any candidate-mediated strong-normalization or local-confluence consumer reading a retained premise from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must preserve the displayed candidate-normalization, residual-substitution, local-join, critical-pair, bounded-check, typed-window, transport, replay, provenance, obstruction, and local* NameCert *rows. The retained-premise route is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

not an unrestricted confluence theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency claim.

Proof. Apply retained-premise closure, Definition 4238.150. It keeps candidate normalization, residual substitution, local joins, critical pairs, bounded checks, typed windows, transport, replay, provenance, obstruction, and naming visible before the retained premise is consumed.

Apply retained-premise induction, Definition 4238.151. A finite list of such reads preserves the same displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ through every residual-pair step.

Now use the candidate SN residual budget and public route sync cited by the retained-pair induction, Definition 4238.166 and Definition 4238.170. These keep the consumer on the candidate-normalization route rather than replacing it by a global semantic theorem.

The carrier exposes no row for unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency. Therefore the retained premise remains a finite candidate-normalization budget over the displayed rows only. □

Theorem 4238.153 (MetaCIC parallel-diamond retained-pair induction)**.** *Every finite successor chain of candidate-mediated SN/confluence handoffs that reads the public or bridged* MetaCICParallelDiamondFrontie *route preserves the retained pair* (P, O) *and the residual budget* (J, R, B) *before each handoff. The induction invariant is read through the displayed public route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

and it does not discharge residual substitution, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. For the base chain, read the public certificate and finite residual witness consumption. The public route exposes $P, K, J, R, S, C, B, O, H, T, G, N$, while the finite witness row keeps the retained pair P, O and the residual budget J, R, B visible before the first handoff.

For the successor step, apply candidate SN residual budget and public route sync, Definition 4238.166 and Definition 4238.170. The head handoff carries P, O as retained premise and obstruction data, and it carries J, R, B as candidate join, residual frontier, and bounded-check budget data. The induction hypothesis supplies the same rows for the remaining finite chain.

Finally compare with the retained seed block sync of Definition 4238.171. It identifies the retained handoff data as finite public-route evidence rather than residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency, so the whole finite chain preserves only the stated pair and residual budget. □

Theorem 4238.154 (MetaCIC parallel-diamond retained-premise mature budget)**.** *Every candidate-mediated SN/confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must carry the retained*

premise budget P, J, R, B, O through the public certificate, finite residual witness consumption, candidate SN residual budget, public frontier budget, and bridged export before reading any parallel-diamond handoff. The carried budget records the residual-substitution premise row, candidate-join row, residual-frontier row, bounded-check row, and obstruction row as finite packet data; it is not residual-substitution discharge, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, global conversion decidability, or kernel consistency.

Proof. Start with the public certificate of Definition 4238.162. It exposes the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as one finite public handoff.

Next consume the finite residual witness through Definition 4238.165. That route makes a candidate-mediated SN/confluence consumer read $P, K, J, R, S, C, B, O, H, T, G, N$ before it can inspect the frontier, so the retained rows P, J, R, B, O remain visible inside the larger finite route.

Apply the candidate SN residual budget and public frontier budget, Definition 4238.166 and Definition 4238.168. Together they isolate P, J, R, B, O as the budget that must be carried forward before any public diamond handoff is read.

Finally apply the bridged export of Definition 4238.169. The bridged read may expose the guarded frontier to an external consumer only while preserving the same retained evidence rows and the structural transport, replay, provenance, and local naming route. Therefore the mature budget cannot be reread as residual-substitution discharge, unrestricted Church–Rosser, subject reduction, evaluator equality, quotient term equality, global conversion decidability, or kernel consistency. □

Theorem 4238.155 (MetaCIC parallel-diamond retained-premise discharge boundary)**.** *Every local diamond/frontier read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that consumes a candidate join row J and residual frontier row R must still expose the retained residual-substitution premise row P, the critical-pair row K, the closed-checker row S, the closed-normal fragment row C, the bounded-check row B, the obstruction row O, and the structural rows H, T, G, N. The route is not full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Read the local public certificate and finite residual witness routes. They expose the candidate join and residual frontier only inside the larger packet $P, K, J, R, S, C, B, O, H, T, G, N$, so consuming J and R cannot hide the premise, critical-pair, checker, normal-fragment, bounded-check, obstruction, or structural rows.

Apply the retained-premise closure and mature budgets. Those budgets require the residual-substitution premise and the obstruction-facing bounded check to remain visible through candidate-mediated diamond reads.

The listed off-table metatheorems are not rows of the packet. Since the carrier exports only the displayed finite evidence and structural transport, replay, provenance, and naming rows, the local read records a discharge boundary rather than a full residual-substitution or confluence theorem. □

Theorem 4238.156 (MetaCIC retained-premise local discharge route)**.** *A candidate-mediated diamond consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *may discharge only the local retained-premise route*

$$P,\ K,\ J,\ R,\ S,\ C,\ B,\ O,\ H,\ T,\ G,\ N.$$

This discharge is local consumption of the displayed route: the retained residual-substitution premise row P must remain present in $P, K, J, R, S, C, B, O, H, T, G, N$ while the consumer reads J and R.

The route still does not provide full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Apply Definition 4238.155. It already forces every candidate join and residual frontier read to keep the retained premise, critical-pair, checker, normal-fragment, bounded-check, obstruction, and structural rows visible.

The criterion is a visibility condition on the retained premise, not a proof that residual substitution has been eliminated.

The present route narrows that boundary to the local consumer order $P, K, J, R, S, C, B, O, H, T, G, N$. Reading J and R is therefore allowed only after the retained-premise and critical-pair rows have been exposed, and before the closed-checker, closed-normal fragment, bounded-check, obstruction, transport, replay, provenance, and naming rows close the local packet.

No further row is exported by this local discharge. The off-table metatheorems remain outside the packet, so the consumer records a retained-premise route rather than a global residual-substitution, confluence, normalization, evaluator-equality, quotient-equality, or consistency theorem. □

Theorem 4238.157 (MetaCIC retained-premise normalization bridge)**.** *Every candidate-mediated normalization or local residual-join read that uses the retained-premise discharge route of Definition 4238.156 must still factor through the finite spine*

$$P,\ K,\ J,\ R,\ S,\ C,\ B,\ O,\ H,\ T,\ G,\ N.$$

It may consume the candidate strong-normalization handoff of Definition 4238.73 and the residual-join obligation surface of Definition 4238.138 only as bounded local evidence. The bridge refuses full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, and kernel consistency.

Proof. Apply Definition 4238.156. Any consumer using that route is already confined to $P, K, J, R, S, C, B, O, H, T$, with P and O still visible before the local packet closes.

Next read the candidate strong-normalization handoff of Definition 4238.73. That handoff can be consumed only after the candidate row, bounded checker rows, obstruction row, transport, replay, provenance, and local naming remain displayed as finite evidence.

Now attach the residual-join obligation surface of Definition 4238.138. It keeps the local residual join inside the same critical-pair, candidate, residual-frontier, checker, obstruction, and structural rows rather than turning the join into an independent confluence theorem.

The carrier inventory and nonescape boundary then close the bridge. Since every cited surface stays inside the displayed spine, no consumer obtains full residual-substitution discharge, subject reduction, Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 4238.158 (MetaCIC parallel-diamond retained-premise boundary)**.** *Every candidate-mediated parallel-diamond consumer of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that reaches a residual join must first expose the retained-premise ledger, candidate-normalization scope, residual-substitution route, finite residual witness, and the supporting* hsame, Cont, Pkg, *and* NameCert *rows. In the frontier spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

the premise row P, candidate row J, residual rows K, R, S, C, B, O, transport row H, replay row T, provenance row G, and local naming row N remain visible before any residual join can be

consumed. The boundary does not provide a hidden strong-normalization oracle, host confluence primitive, detached residual equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Start with the retained-premise discharge boundary, Definition 4238.155. It keeps $P, K, S, C, B, O, H, T, G, N$ visible whenever a local diamond or frontier read consumes the candidate join J and residual frontier R.

Next apply the candidate strong-normalization handoff, Definition 4238.73. That handoff exposes the critical-pair envelope, residual-substitution row, candidate-normalization scope, finite residual witness, and residual-row visibility surface before a normalization-facing consumer can read the frontier.

Now use the residual-substitution dependency route, Definition 4238.57. It places the retained premise row before every critical-pair, candidate-join, residual-frontier, closed-checker, and obstruction read, so the residual join cannot be detached from the premise boundary.

Finally attach the finite residual-diamond ledger, Definition 4238.123. The ledger records the local join only as bounded carrier-local data over $K, P, J, B, O, H, T, G, N$. Since every cited route preserves the same structural support rows, the residual join remains a retained-premise boundary rather than a hidden strong-normalization oracle, host confluence primitive, detached residual equality, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate. □

Theorem 4238.159 (MetaCIC parallel-diamond retained-premise scope)**.** *Every retained premise used by a* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *read must be displayed as one of the residual-pair, candidate-normalization, local-join, closed-checker, or obstruction rows already named in the frontier carrier*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

In particular, a retained premise can be consumed only as carrier-local evidence through those rows, and cannot be read as an unconditional closed parallel-diamond theorem, subject-reduction theorem, evaluator equality, quotient term equality, or kernel-consistency certificate.

Proof. Project the frontier carrier of Definition 4238.1. The residual-pair and premise-retention rows are P, K, R, the candidate-normalization row is J, the local-join and closed-checker rows are S, C, B, and the obstruction row is O.

Apply the residual boundary route of Definition 4238.57. It places every residual-pair use behind the retained premise row and keeps the critical-pair and residual-frontier rows visible.

Next use the candidate-normalization and local-join boundaries, Definition 4238.62 and Definition 4238.123. These theorems allow candidate and local-join data to be read only through the displayed finite route, with the retained premise still attached.

Finally apply the retained-premise discharge boundary and obstruction ledger, Definition 4238.155 and Definition 4238.111. The closed-checker and obstruction rows remain visible through the structural rows H, T, G, N, so no consumer can reread the retained premise as an unconditional closed parallel diamond or any of the refused kernel-level endpoints. □

Theorem 4238.160 (MetaCIC parallel-diamond local retained-premise factorization)**.** *Any local diamond read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet that consumes candidate join and residual-frontier rows factors through the retained premise row, critical-pair row, candidate-normalization row, local-join row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, and structural rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The factorization is conditional on those displayed rows. It is not full Church–Rosser, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with retained-premise scope, Definition 4238.159. It requires every premise used by the local diamond read to remain one of the displayed carrier rows.

Apply the residual-substitution dependency route of Definition 4238.57. The route places the retained premise row P before the critical-pair, candidate join, residual-frontier, closed-checker, and obstruction rows can be consumed.

Use candidate-normalization scope from Definition 4238.62. The candidate row J remains inside the same conditional spine rather than becoming an external normalization theorem.

Apply the finite residual-diamond ledger local-join obligations of Definition 4238.123. They read the local join, closed checker, closed-normal fragment, bounded-check, and obstruction rows only after the retained premise and residual-frontier data have been displayed.

Finally use the retained-premise discharge boundary of Definition 4238.155. Since the structural rows H, T, G, N only transport, replay, source, and name the same finite packet, the local diamond read cannot be promoted to any refused global metatheorem or kernel endpoint. □

Theorem 4238.161 (MetaCIC retained-premise residual-join scope)**.** *Every candidate-mediated residual join read from an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet remains inside the typed local-join scope when the retained premise window is exposed before the normalization and confluence handoff rows. Equivalently, the read must keep the finite carrier-local route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

visible with P as the retained premise, K as the critical-pair row, J as the candidate-normalization row, R, S, C, B as the residual witness, local-join, closed-checker, and bounded-checker rows, O as the obstruction row, and H, T, G, N as transport, replay, provenance, and local naming.

The resulting scope is a retained-premise residual-join surface. It does not export premise-free residual substitution, global confluence, subject reduction, evaluator equality, quotient term equality, unrestricted normalization, or kernel consistency.

Proof. First apply retained-premise scope, Definition 4238.159. It requires every premise used by a frontier read to appear as one of the displayed residual-pair, candidate-normalization, local-join, closed-checker, obstruction, transport, replay, provenance, or naming rows.

Next use candidate-mediated residual-join scope, Definition 4238.134. The candidate row and residual rows are consumed only after the critical-pair envelope and residual-substitution dependency route have made the retained premise visible.

Now read the window-budget scope and residual-join obligation surface, Definition 4238.137 and Definition 4238.138. These theorems keep the bounded checker, local join, obstruction, transport, replay, provenance, and local naming rows on the same carrier-local route.

Therefore a candidate-mediated residual join can be exported only as the displayed retained-premise surface $P, K, J, R, S, C, B, O, H, T, G, N$. Since no cited row contains a premise-free residual-substitution discharge or any global kernel endpoint, the conclusion remains inside the typed local-join scope. □

Theorem 4238.162 (MetaCIC parallel-diamond frontier public certificate)**.** *The public* MetaCICParallelDiamondFr *certificate is the finite route through the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local*

NameCert *rows. It exposes the residual frontier as a public finite handoff while refusing residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency.*

Proof. Apply the public export route of Definition 4238.47. It displays the residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming rows before any outside consumer may read the frontier.

Apply the public seed-readiness route of Definition 4238.84. This keeps the closed-checker seal, closed-normal fragment, bounded-check row, and obstruction row inside the same finite handoff.

The two routes name only the displayed packet rows. Since neither route contains a discharge of the residual-substitution premise, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, the public certificate exports the finite frontier without claiming those endpoints. □

Theorem 4238.163 (MetaCIC parallel-diamond frontier seed route)**.** *The rotation-selected status route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *records the finite move from no closure block to* seed *through the retained carrier rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route is witnessed inside this child file by public certificate export, finite residual witness consumption, retained seed-block sync, public route sync, bridged export, mature route, bridge boundary, and closurestatus seed block evidence. It keeps residual-substitution premises, local joins, residual-frontier rows, bounded checks, obstruction rows, transport, replay, provenance, and local naming visible, and it does not claim global confluence, unrestricted normalization, subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, residual-substitution discharge, or kernel consistency.

Proof. Start with the closurestatus seed block of Definition 4238.175. It binds the status-bearing route to the retained premise rows and the displayed carrier route $P, K, J, R, S, C, B, O, H, T, G, N$.

Apply public route sync and bridged export, Definition 4238.170 and Definition 4238.169. These keep the public and bridge-facing reads on the same finite route before any outside consumer reads the frontier.

Finally use the mature route and bridge boundary, Definition 4238.173 and Definition 4238.174. They preserve the retained premise budget and refuse the listed global endpoints, so the status route reaches only seed for the finite frontier packet. □

Theorem 4238.164 (MetaCIC parallel-diamond bridged handoff)**.** *The public residual-premise, critical-pair, candidate-join, residual-frontier, closed-checker, obstruction, transport, replay, provenance, and local naming route is the bridged handoff surface for* MetaCICParallelDiamondFrontier$^{\uparrow}$*. The handoff records the finite rows that a standard bridge may read before interpreting the parallel-diamond frontier externally. Residual-substitution discharge, unrestricted confluence, subject reduction, evaluator equality, quotient term equality, kernel consistency, and bridge checking remain outside this chapter.*

Theorem 4238.165 (Finite residual witness consumption)**.** *Every candidate-mediated SN/confluence consumer of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must consume the finite residual witness route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through the displayed residual-premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal fragment, bounded-check, obstruction, transport, replay, provenance, and local naming rows. The consumer may not retype the handoff as residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Read the public certificate of Definition 4238.162. It exposes the residual premise, critical-pair, candidate-join, residual-frontier, checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as one finite handoff.

Now apply the bridged handoff of Definition 4238.164. The bridge-facing consumer may inspect only the rows named by that handoff before interpreting the frontier externally. The row order $P, K, J, R, S, C, B, O, H, T, G, N$ records the premise, critical-pair, join, residual, seal, checker, bounded-check, obstruction, transport, replay, provenance, and local-name witnesses as finite data.

Since none of those rows is a residual-substitution discharge, subject-reduction theorem, unrestricted Church–Rosser proof, evaluator equality, quotient equality, or kernel-consistency certificate, a consumer that claims one of those endpoints has left the displayed $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ route. □

Theorem 4238.166 (Candidate SN residual budget)**.** *Any candidate-mediated strong-normalization or confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must preserve the residual-substitution premise row P, candidate-join row J, residual-frontier row R, bounded-check row B, and obstruction row O as one finite residual budget before reading the public diamond handoff. The budget is read as a retained row family, not as residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency.*

Proof. Use the public certificate of Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, and obstruction rows inside the same finite public handoff rather than as separate global principles.

Apply finite residual witness consumption, Definition 4238.165. The candidate-mediated consumer can inspect the route $P, K, J, R, S, C, B, O, H, T, G, N$, but the residual budget needed for SN/confluence use is the retained subroute P, J, R, B, O.

Finally compare with the public closure theorem, Definition 4238.167. Since public closure keeps residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, and kernel consistency outside the packet, the candidate SN/confluence reader must carry P, J, R, B, O forward before it reads the public diamond handoff. □

Theorem 4238.167 (MetaCIC parallel-diamond frontier public closure)**.** *Public closure for* MetaCICParallelDiamon *is available only over the retained finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route retains the residual-substitution premise row, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, transport row, replay row, provenance row, and local naming row before any public diamond handoff. It does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency.

Proof. Use the public certificate of Definition 4238.162. It exposes the residual premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming rows as a finite public handoff.

Now apply the public seed-readiness route of Definition 4238.84. It keeps the closed-checker seal, closed-normal fragment, bounded-check row, and obstruction row inside the same retained packet rather than moving them to a hidden evaluator or quotient term layer.

Finally use the retained witness consumption of Definition 4238.165. The ordered route $P, K, J, R, S, C, B, O, H, T, G, N$ is the only public surface consumed by downstream SN/confluence fragments. Since the retained rows leave residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, and kernel consistency outside the packet, the public closure is exactly this finite route. □

Theorem 4238.168 (MetaCIC parallel-diamond public frontier budget)**.** *Every public candidate-mediated SN or confluence consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must preserve the residual-substitution premise row P, candidate join row J, residual frontier row R, bounded-check row B, and obstruction row O before reading any public diamond handoff. The retained budget cannot be retyped as residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with the public certificate, Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, and obstruction rows as visible rows of the finite public handoff rather than as hidden global principles.

Use finite residual witness consumption, Definition 4238.165. The candidate-mediated consumer reads the route $P, K, J, R, S, C, B, O, H, T, G, N$, so the residual-substitution premise, candidate join, residual frontier, bounded check, and obstruction rows are retained before the public handoff is inspected.

Apply candidate SN residual budget, Definition 4238.166. It isolates P, J, R, B, O as the budget that a strong-normalization or confluence consumer must carry forward.

Finally compare with public closure, Definition 4238.167. Public closure refuses residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient equality, and kernel consistency, so preserving P, J, R, B, O cannot be retyped as any of those endpoints. □

Theorem 4238.169 (MetaCIC parallel-diamond frontier bridged export)**.** *Any public or bridged read of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier must preserve the candidate-normalization rows, residual-frontier rows, local-join ledger, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *as visible evidence in the same finite carrier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The export may hand this guarded frontier to a bridge-facing consumer, but it does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, residual-substitution discharge, global conversion decidability, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with residual-join locality, Definition 4238.104. It keeps every residual-join read local to the accepted carrier and forces the critical-pair, residual-substitution, candidate-normalization, finite-witness, obstruction, and local naming rows to remain visible.

Next use the residual-normalization window and seed boundary, Definition 4238.126 and Definition 4238.127. These rows retain candidate-normalization evidence, the residual frontier, the obstruction row, the closed-checker evidence, and the bounded-check row in the same finite route.

Finally apply public closure and the bridged handoff, Definition 4238.167 and Definition 4238.164. The structural rows H, T, G, N provide componentwise transport, replay, provenance, and local

naming for the route $P, K, J, R, S, C, B, O, H, T, G, N$. Since those cited rows explicitly leave global confluence, strong normalization, subject reduction, conversion decidability, evaluator equality, quotient equality, residual-substitution discharge, and kernel consistency outside the packet, the bridged export is exactly the guarded evidence-preserving route. □

Theorem 4238.170 (MetaCIC parallel-diamond public route sync)**.** *Every retained residual-budget consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that reads the public or bridged frontier must preserve the full route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

from public certificate through finite residual witness consumption, candidate SN residual budget, and bridged export. The synchronized route keeps the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *visible in one finite carrier. It refuses residual-substitution discharge, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, and kernel consistency.*

Proof. Start with the public certificate of Definition 4238.162. It exposes the residual-substitution premise, critical-pair, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction, transport, replay, provenance, and local naming rows before any public consumer may read the frontier.

Next use finite residual witness consumption, Definition 4238.165. This forces the consumer to read the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$ as finite witness data, rather than as a hidden residual-substitution theorem.

Apply the candidate SN residual budget of Definition 4238.166. It retains the budget rows P, J, R, B, O inside the larger public route before a strong-normalization or confluence-facing consumer may use the handoff.

Finally apply the bridged export theorem, Definition 4238.169. The export hands the guarded frontier to bridge-facing readers only with the same candidate, residual, obstruction, transport, replay, provenance, and local-name evidence visible. Since all cited routes leave residual-substitution discharge, subject reduction, global Church–Rosser, evaluator equality, quotient term equality, and kernel consistency outside the carrier, the synchronized public route cannot assert those endpoints. □

Theorem 4238.171 (MetaCIC parallel-diamond retained seed block sync)**.** *Let a finite candidate-mediated SN/confluence consumer chain read* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *through the public or bridged frontier. At every successor handoff in that finite chain, the consumer must retain the residual-substitution premise row P, candidate-join row J, residual-frontier row R, bounded-check row B, and obstruction row O before it may read the public route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Thus the finite induction seed block for such consumers is the retained row family P, J, R, B, O together with the public route. The induction invariant does not discharge residual substitution, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. The base handoff is the public certificate of Definition 4238.162. It exposes the residual premise, candidate join, residual frontier, bounded check, obstruction, transport, replay, provenance, and local naming rows before a consumer reads the frontier.

For the successor step, apply finite residual witness consumption, Definition 4238.165, and the candidate SN residual budget, Definition 4238.166. These two routes force the next consumer to carry P, J, R, B, O as retained finite evidence before inspecting the larger displayed route.

The public and bridged reads are then synchronized by Definition 4238.170 and Definition 4238.169. Induction over the displayed finite consumer chain repeats the same successor handoff, so every stage retains P, J, R, B, O before reading $P, K, J, R, S, C, B, O, H, T, G, N$.

All cited routes explicitly refuse residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. The induction therefore preserves a finite seed block; it does not prove any of those external endpoints. □

Theorem 4238.172 (MetaCIC parallel-diamond retained-premise induction step)**.** *For one successor handoff in any finite candidate-mediated SN/confluence consumer chain of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*, the premise budget rows*

$$P, J, R, B, O$$

are preserved before the next consumer reads the full public route

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The step preserves residual-substitution premise data, candidate-join data, residual-frontier data, bounded-check data, and obstruction data. It does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient equality, global normalization, or kernel consistency.

Proof. Start from finite residual witness consumption, Definition 4238.165. It forces the successor reader to consume the displayed route $P, K, J, R, S, C, B, O, H, T, G, N$ as finite witness data.

Apply the candidate SN residual budget of Definition 4238.166. This isolates P, J, R, B, O as the premise rows that must be carried before any SN/confluence-facing read of the public handoff.

Use public route sync, Definition 4238.170, to keep the full route synchronized across public and bridged reads. Then apply bridged export, Definition 4238.169, which preserves the same guarded frontier rows for bridge-facing consumers.

Finally read the retained seed block sync of Definition 4238.171. It supplies the finite-chain induction context, so the present successor handoff preserves the retained premise budget without proving any refused global endpoint. □

Theorem 4238.173 (MetaCIC parallel-diamond mature route)**.** *Every bridge-facing or critical-path consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *preserves the retained route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

through the public certificate, finite residual witness consumption, candidate SN residual budget, public route sync, bridged export, and retained seed block sync. The mature route keeps the residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker seal, closed-normal fragment, bounded-check row, obstruction row, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *visible before any consumer reads the bridge-facing frontier. It explicitly refuses residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the public certificate of Definition 4238.162. It displays the residual-substitution premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows before a public reader consumes the frontier.

Apply finite residual witness consumption, Definition 4238.165. This forces the bridge-facing reader to consume the finite route $P, K, J, R, S, C, B, O, H, T, G, N$ as residual witness data rather than as a hidden substitution theorem.

Use the candidate SN residual budget, Definition 4238.166. The budget rows P, J, R, B, O are retained before strong-normalization or confluence-facing consumers may inspect the public diamond handoff.

Synchronize the public route by Definition 4238.170. The synchronized route preserves the full displayed family through public and bridged reads, keeping the structural rows H, T, G, N attached to the candidate and residual rows.

Finally apply the bridged export and retained seed block sync, Definition 4238.169 and Definition 4238.171. The former hands the guarded frontier to bridge-facing readers with the retained evidence still visible, and the latter propagates the seed rows through every finite successor handoff.

All cited routes refuse residual-substitution discharge, subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. The present route also refuses global normalization, because the retained residual rows provide finite guarded evidence rather than a global evaluator or normalization theorem. □

Theorem 4238.174 (MetaCIC parallel-diamond retained-premise bridge boundary)**.** *The status-bearing boundary for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is the retained-premise route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

of Definition 4238.173. Bridge-facing readers must keep the residual-substitution premise rows P, candidate-join rows J, residual-frontier rows R, bounded-check rows B, and obstruction rows O visible as premise data; the route does not convert them into residual-substitution discharge, subject reduction, full Church–Rosser, quotient equality, evaluator equality, or kernel consistency.

Proof. Apply Definition 4238.173. It binds the public and bridge-facing route to the displayed finite row family $P, K, J, R, S, C, B, O, H, T, G, N$.

Now read the retained-premise rows through Definition 4238.171. The sync keeps P, J, R, B, O as visible prerequisite rows through finite successor handoffs rather than discharging them.

The refusal clauses are inherited from the mature route and from the closure block below. They leave the paper route mature and theorem-checked while keeping bridge checking as the remaining boundary. □

Theorem 4238.175 (MetaCIC parallel-diamond closurestatus seed block)**.** *The status route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is determined by the retained-premise rows P, J, R, B, O, residual-join locality rows, candidate-normalization rows, and frontier-export rows of the displayed carrier route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Reading these rows through the public route and bridged export fixes the closurestatus seed block without adding residual-substitution discharge, subject reduction, quotient equality, evaluator equality, full Church–Rosser, global normalization, or kernel consistency.

Proof. Start with the retained seed block sync, Definition 4238.171. It keeps P, J, R, B, O visible through every finite successor handoff before any public or bridged frontier read is consumed.

Apply the mature route, Definition 4238.173. That route already factors the status-bearing surface through public certificate, finite residual witness consumption, candidate SN residual budget, public route sync, bridged export, and retained seed block sync.

Now use the residual-join locality and candidate-normalization rows cited by that route, including Definition 4238.104 and Definition 4238.62. They keep local joins and normalization-facing reads inside the same retained carrier rows rather than turning them into global confluence or normalization theorems.

Finally apply the public route sync and bridged export, Definition 4238.170 and Definition 4238.169. These export only the guarded frontier route $P, K, J, R, S, C, B, O, H, T, G, N$ with the retained premise, obstruction, transport, replay, provenance, and local naming still visible. Hence the closurestatus seed block is fixed by the displayed frontier rows, and none of the refused endpoints is introduced by the status route. □

Theorem 4238.176 (MetaCIC parallel-diamond scoped frontier status)**.** *The scoped* MetaCICParallelDiamondFront *frontier is the finite route that begins with the obligation surface of Definition 4238.2, retains the seed-status route of Definition 4238.79, and reads the public certificate and public closure of Definition 4238.162 and Definition 4238.167. It records the displayed residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows while refusing residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency.*

Proof. Start with the obligation surface of Definition 4238.2. It fixes the accepted packet through the finite route $P, K, J, R, S, C, B, O, H, T, G, N$ and leaves the residual-substitution premise as a displayed row rather than a discharged theorem.

Apply the seed-status route of Definition 4238.79. The route keeps the same residual premise, critical-pair, candidate, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows visible before any seed consumer reads the frontier.

Finally use the public certificate and public closure, Definition 4238.162 and Definition 4238.167. These theorems expose the finite frontier as a public handoff only over the retained route, so none of the refused residual-substitution, subject-reduction, confluence, evaluator-equality, quotient-equality, global-normalization, or kernel-consistency endpoints is exported by the scoped status. □

Theorem 4238.177 (MetaCIC parallel-diamond scoped status register)**.** *The scoped status of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is registered by the same finite frontier displayed in Definition 4238.176. In particular, the registered route is the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, checker row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local naming row, with no export of subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Read the scoped frontier status theorem. It already fixes the route through the obligation surface, seed-status route, public certificate, and public closure while retaining the residual-substitution premise as a visible row.

The status register does not add a new carrier coordinate. It records that this displayed route is the one later readers may cite, and it preserves the same refusals for subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, and kernel consistency. □

Theorem 4238.178 (MetaCIC parallel-diamond closurestatus export route)**.** *The seed and bridged closure markers for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are backed by the finite route through the existing critical-pair envelope, residual-substitution dependency route, finite residual witness consumption, candidate-SN handoff, and scoped-frontier status rows. In particular, a closurestatus reader reaches the frontier only through the displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *packet and cannot reinterpret the marker as residual-substitution discharge, unrestricted subject reduction, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the critical-pair and residual-substitution siblings. They retain the local critical-pair rows and the residual-substitution premise as visible finite evidence before any frontier consumer reads the packet.

Next use finite residual witness consumption and the candidate-SN handoff. These rows keep the typed window, candidate-normalization row, bounded local-join checker, obstruction row, transport, replay, provenance, and local naming inside the same displayed route.

Finally read the scoped-frontier status. It registers the same finite route as the status surface for the chapter, so the seed and bridged markers are exported as pointers to that route rather than as global confluence, subject-reduction, evaluator-equality, quotient-equality, residual-substitution, or kernel-consistency theorems. □

Theorem 4238.179 (MetaCIC parallel-diamond retained-premise marker route)**.** *Every seed, public, or bridged marker read for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is consumed only through the displayed* $P, K, J, R, S, C, B, O, H, T, G, N$ *route. In particular,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *and* $\mathsf{MetaCICConfluenceAuditPac}$ *consumers retain the critical-pair row* P *and obstruction row* O *while reading the marker, so the marker cannot be reinterpreted as a discharge of the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport row, replay row, provenance row, or local* $\mathsf{NameCert}$ *row.*

Proof. Apply the closurestatus export route above. It makes $P, K, J, R, S, C, B, O, H, T, G, N$ the only displayed path by which the marker is exported.

Now restrict the read to $\mathsf{MetaCICCriticalPath}^{\uparrow}$ or $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$. These consumers inspect the marker as a frontier status, not as a replacement for its packet rows. Hence they must retain the critical-pair premise P and the obstruction row O while reading candidate joins, bounded checks, transport, replay, provenance, and local naming.

The retained rows block the overreading. A marker consumer may record where the frontier sits, but it receives no residual-substitution discharge, unrestricted join theorem, global confluence theorem, or kernel-consistency certificate. □

Theorem 4238.180 (MetaCIC parallel-diamond closurestatus guard)**.** *Every closurestatus reader or public status reader of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *must pass through the retained residual-substitution premise row, critical-pair envelope, candidate-normalization scope, finite residual witness ledger, candidate-SN handoff, obstruction row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows before it may read the seed or bridged markers. In carrier notation, the reader must expose the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

as frontier data before consuming the status markers. The guard does not discharge residual substitution, prove subject reduction, prove full Church–Rosser, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Begin with the closurestatus export route, Definition 4238.178. It identifies the seed and bridged markers as pointers into the finite $P, K, J, R, S, C, B, O, H, T, G, N$ packet rather than as independent confluence or kernel claims.

Next read the seed package and scoped seed package route, Definition 4238.82 and Definition 4238.103. These keep the residual-substitution premise, critical-pair rows, candidate rows, residual-frontier rows, bounded checker rows, obstruction row, and structural rows visible before a status reader reaches the packet.

Now apply residual-join locality and the finite residual-diamond ledger, Definition 4238.104 and Definition 4238.123. They force every local join read to remain carrier-local, under the retained premise and obstruction rows, and with transport, replay, provenance, and local naming still displayed.

Finally use the candidate-SN handoff, Definition 4238.73. It permits a candidate-mediated frontier budget only after the finite residual witness and bounded checker rows remain visible, so a closurestatus or public status reader can consume the seed or bridged marker only through the displayed guard route. Since the route has no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the status marker. □

The mature paper closure for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ is read through the retained-premise rows, candidate-normalization scope, local-join rows, and frontier-export route already exposed by the sibling files. Those rows bind the displayed frontier packet before any consumer reaches the mature closurestatus surface.

Bridge checking remains absent at this paper site. Until a StandardBridge theorem exists for the same retained-premise, candidate-normalization, local-join, and frontier-export route, the bridge-status coordinate stays separate from the mature paper closure and does not assert residual-substitution discharge, subject reduction, quotient equality, evaluator equality, global Church–Rosser, or kernel consistency.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ packet is generated from residual-substitution premise rows, critical-pair rows, candidate-join rows, residual-frontier rows, closed checker seal rows, closed-normal fragment rows, bounded-check rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 4238.48, Definition 4238.150, Definition 4238.104, and Definition 4238.62 display the child-route retained-premise, residual-join, candidate-normalization, local-join, residual-frontier, and frontier-export rows $P, K, J, R, S, C, B, O, H, T, G, N$ as the scoped frontier surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}, \mathsf{bridgeChecked})$

Lean target: `BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObligations`

Bridge status: bridgeChecked

Not claimed: This child route does not close residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, kernel consistency, or checked bridge interpretation beyond the displayed handoff.

Upgrade path: No stronger paper-side upgrade is claimed here; residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency remain outside the displayed bridge-checked route.

Lemma 4238.181 (MetaCIC parallel-diamond canonical closurestatus reader)**.** *The canonical paper status for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is the bridge-checked closurestatus block in* `closurestatus_curren`

Any seed or top-transition reader that enters through the hub must discharge its status read through the child obligation surface BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDiamondFrontierObliga *and then through that closurestatus block.*

Proof. The hub displays the seed, bridged, and mature route markers, but its body is only a routing list. Therefore a status consumer must leave the hub and read the child files that carry the frontier evidence.

First read the checked obligation surface recorded in Definition 4238.2. It exposes the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming before any status marker is consumed.

Then read the closurestatus child. Its block records maturePackageClosed, bridgeChecked, and `bridgeChecked` for the same Lean obligation target, so the seed and top-transition routes point to that canonical child record rather than to a second hub-level status body. □

Lemma 4238.182 (MetaCIC parallel-diamond frontier closure marker consistency)**.** *The visible* MetaCICParallelDiamondFrontier$^{\uparrow}$ *hub markers are read only through the child closurestatus records. In particular, a consumer that sees the hub-level seed, bridged, or mature marker must route that read through the current closurestatus child, Definition 4238.185, and Definition 4238.205 before using it as closure evidence.*

Proof. The hub exports marker lines and child inputs, but it carries no theorem body of its own. Therefore the evidence-bearing reads are the child files named by the hub, not the marker text itself.

Read the current closurestatus child first, then the seed closure status sync child, and then the seed status block sync child. These rows bind the visible marker strings to the finite frontier packet, closurestatus record, seed-readiness rows, replay, transport, provenance, and local naming.

Thus a downstream consumer cannot treat the hub marker as an independent closure proof. It must pass through the child records that store the corresponding closurestatus evidence. □

Theorem 4238.183 (MetaCIC parallel-diamond bridge hub child route)**.** *The bridged and mature markers displayed by the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *hub are readable only through the child files that expose the carrier, obligation surface, closurestatus guard, and retained-premise route. Concretely, a status reader must pass through the carrier rows*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

of Definition 4238.1, the obligation surface of Definition 4238.2, the closurestatus guard of Definition 4238.180, and the retained-premise budget of Definition 4238.154 before it may consume those hub markers.

The route records local auditability for the hub status only. It does not discharge residual substitution, prove subject reduction, prove unrestricted confluence, identify evaluator equality, identify quotient term equality, or certify kernel consistency.

Proof. Start with the hub's two status markers. They are routing annotations for the same finite frontier packet, so a reader must follow the child inputs rather than treating the hub line itself as a proof object.

The carrier child supplies the row tuple $P, K, J, R, S, C, B, O, H, T, G, N$, and the obligation theorem names the residual-substitution premise, critical-pair, candidate-join, residual-frontier,

closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as the exposed surface.

Apply the closurestatus guard. It forces status readers to keep the retained premise, critical-pair envelope, candidate-normalization scope, finite residual witness ledger, candidate-SN handoff, obstruction row, transport, replay, provenance, and local NameCert rows visible before seed or bridged markers are read.

Finally apply the retained-premise mature budget. The same premise, candidate, residual, bounded-check, and obstruction rows remain carried through the public and bridged export route, so the hub markers are locally auditable without supplying residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.184 (MetaCIC parallel-diamond hub child status route)**.** *The seed, bridged, and mature status rows for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are read only through the child carrier route. A status reader must first expose the carrier rows of Definition 4238.1, the obligation surface of Definition 4238.2, the seed closure status of Definition 4238.45, and the closurestatus export and guard routes of Definition 4238.178 and Definition 4238.180. The route exports no residual-substitution discharge, subject reduction, full Church–Rosser theorem, evaluator equality, quotient equality, or kernel consistency.*

Proof. Start with the child carrier. The displayed route of Definition 4238.1 is the finite packet

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and the obligation surface of Definition 4238.2 names those rows as the visible premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming data.

Next read the seed status through Definition 4238.45. That theorem keeps the seed-level status attached to the same displayed frontier packet rather than to a detached confluence or consistency assertion.

Finally apply the closurestatus export route and the closurestatus guard. They force bridged and mature status readers through the retained child rows before any status marker is consumed. Since the cited rows contain no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency, the hub status route exports only the guarded child carrier. □

Theorem 4238.185 (MetaCIC parallel-diamond frontier seed closure status sync)**.** *The seed registration route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is subsumed by the existing mature and bridged hub markers together with the child closurestatus route. In particular, any status reader that enters through the seed marker must read the obligation surface of Definition 4238.2, the seed closure status of Definition 4238.45, the closurestatus export route of Definition 4238.178, and the guard of Definition 4238.180 before treating the frontier as a mature paper route.*

Proof. Read the hub markers as pointers into the child files. The marker rows name the mature and bridged status of $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, but they do not add carrier rows beyond the finite residual-premise, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming route already displayed in the body.

Apply the obligation theorem and the seed closure status theorem. Together they expose the NameCert obligations and the finite seed registration surface before a status reader may inspect the marker.

Now use the closurestatus export route and the closurestatus guard. These theorems force the reader through the retained route $P, K, J, R, S, C, B, O, H, T, G, N$ and keep residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, and kernel consistency outside the packet. Hence the seed registration route is absorbed by the existing mature closurestatus route without downgrading the theory or verification axes. □

Theorem 4238.186 (MetaCIC parallel-diamond frontier formal-target request)**.** *The next Lean target for the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier is the residual-substitution premise retention route over the displayed rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

This target asks only that a confluence-facing read retain the residual-substitution premise row P while passing through the critical-pair row K, candidate-join row J, residual-frontier row R, closed-checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N. It is not full subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the seed closure status sync of Definition 4238.185. It identifies the status reader with the same child route already guarded by the frontier body, so the requested formal target should be stated over the displayed carrier rows rather than over a global confluence endpoint.

Apply the obligation surface of Definition 4238.2. The obligations expose $P, K, J, R, S, C, B, O, H, T, G, N$ as the finite packet and leave P as a retained residual-substitution premise, not as a discharged theorem.

Finally use the closurestatus export route and guard, Definition 4238.178 and Definition 4238.180. These theorems keep the closurestatus reader on the same retained route and exclude subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, and kernel consistency from the exported target shape. □

Theorem 4238.187 (MetaCIC parallel-diamond frontier retained-premise target boundary)**.** *The next formal boundary for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is precisely preservation of the retained residual-substitution premise row P through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The target asks that a reader carrying P through the critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N still reaches the same retained premise boundary. It does not ask for subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Apply the formal-target request theorem, Definition 4238.186. It already restricts the Lean-facing request to the displayed carrier rows and identifies P as retained premise data rather than a discharged global theorem.

Next read the obligation surface of Definition 4238.2. That surface is the finite packet $P, K, J, R, S, C, B, O, H, T$ so retained-premise preservation is the exact boundary visible to the local NameCert route.

Finally use the closurestatus guard, Definition 4238.180. The guard keeps the exported target on the retained route and refuses subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, and kernel consistency. □

Theorem 4238.188 (MetaCIC parallel-diamond formal-target consumer retention)**.** *Any consumer of the formal-target request for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must preserve the residual-substitution premise row P through the finite route*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

before it reads either candidate-mediated strong-normalization data or the closurestatus export. The consumer may inspect the critical-pair row K, candidate-join row J, residual-frontier row R, checker row S, closed-normal fragment row C, bounded-check row B, obstruction row O, transport row H, replay row T, provenance row G, and local naming row N only as the same retained-premise packet. It does not receive residual-substitution discharge, subject reduction, arbitrary Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Apply the formal-target request theorem, Definition 4238.186. It states the request over $P, K, J, R, S, C, B, O, H, T, G, N$ and identifies P as retained premise data.

Next apply the retained-premise target boundary, Definition 4238.187. That boundary forces any downstream read to carry the same P row through the displayed critical-pair, candidate, residual, checker, obstruction, transport, replay, provenance, and naming route.

Finally use the closurestatus guard, Definition 4238.180. Since the guard is the route through which closurestatus export remains visible, a candidate-mediated strong-normalization or closurestatus consumer cannot read the frontier after dropping P or detaching J, R, B, O from the retained packet. □

Theorem 4238.189 (MetaCIC parallel-diamond frontier seed closure route)**.** *The* MetaCICParallelDiamondFrontier *seed closure route is visible only after the critical-pair envelope of Definition 4238.50, residual-join locality route of Definition 4238.104, retained-premise budget of Definition 4238.154, local NameCert rows of Definition 4238.2, seed-status row of Definition 4238.45, closurestatus export row of Definition 4238.178, and closurestatus guard of Definition 4238.180 have all been displayed. The transport, replay, and provenance rows are read as the structural coordinates H, T, G, N of the same child ledger.*

Proof. Read the child ledger rows in order. The critical-pair envelope and residual-join locality route display the local confluence data, while the retained-premise budget and local NameCert rows keep that data attached to the same frontier packet.

The seed-status row then records the local seed claim, and the closurestatus export and guard rows determine what can be exposed at the chapter boundary. Since the route is assembled only from these displayed child rows, the seed record cannot bypass the local ledger. □

Theorem 4238.190 (MetaCIC parallel-diamond frontier seed closure record)**.** *The* MetaCICParallelDiamondFrontie *seed closure record is valid exactly when the displayed child ledger contains the critical-pair envelope of Definition 4238.50, the residual-join locality route of Definition 4238.104, the retained-premise budget of Definition 4238.154, the local NameCert obligations of Definition 4238.2, the seed-status row of Definition 4238.45, the closurestatus export row of Definition 4238.178, and the guard rows of Definition 4238.180. Under this record the structural rows H, T, G, N are read only as transport, replay, provenance, and local naming coordinates of the same child packet.*

The record exports finite seed closure for that displayed packet. It does not assert global confluence, strong normalization, subject reduction, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the route of Definition 4238.189. It already orders the critical-pair envelope, residual-join locality route, retained-premise budget, local NameCert obligations, seed-status row, closurestatus export, and closurestatus guard before any seed reader reaches the chapter boundary.

The forward read is therefore finite row projection. If the seed closure record is valid, each cited row is displayed in the child ledger, and H, T, G, N can only transport, replay, source, and name that same packet.

Conversely, assume the seven displayed rows are present. The route theorem repacks them into the seed closure route, so the record has exactly the rows needed for the seed-status reader and the closurestatus export guard.

No global endpoint is introduced in the repacking. The cited rows are carrier-local and retain the premise and obstruction boundaries, so the record cannot become a global confluence theorem, strong-normalization theorem, subject-reduction theorem, evaluator-equality theorem, quotient-equality theorem, residual-substitution discharge, or kernel-consistency certificate. □

The seed-closure status reader for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads the seed record, scoped frontier status, and bridge-status separation as one finite critical-path surface. The seed closure record of Definition 4238.190 supplies the local residual-join ledger, the scoped frontier status of Definition 4238.176 keeps the retained residual premise visible, and the bridge-status sync route keeps bridge checking separate from seed registration.

Thus a MetaCIC critical-path consumer may read the seed axis only through the displayed child files before reaching the frontier closurestatus. The route does not discharge residual substitution, subject reduction, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Theorem 4238.191 (MetaCIC parallel-diamond frontier seed closure marker)**.** *Every seed registration route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *reads the carrier row, critical-pair envelope, candidate-normalization scope, residual-substitution dependency, finite residual witness, local join, obstruction, transport, replay, provenance, and local* NameCert *rows before any candidate-mediated strong-normalization or confluence consumer may use the frontier. In the displayed spine*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

P retains the residual-substitution dependency, K exposes the critical-pair envelope, J records the candidate-normalization row, R, S, C, B carry the finite residual witness and local-join budget, O retains the obstruction row, and H, T, G, N provide transport, replay, provenance, and local naming.

Proof. Start with the frontier carrier and obligation surface. They display the finite rows $P, K, J, R, S, C, B, O, H, T$ and keep the residual-substitution premise, critical-pair data, candidate row, residual frontier, checker rows, obstruction, transport, replay, provenance, and local name as ordinary packet coordinates.

Read the seed-facing child routes next. The critical-pair envelope, candidate-normalization scope, residual-substitution dependency route, finite residual witness consumption, and residual-join locality rows force the seed route to pass through the same displayed packet before a consumer sees the frontier.

Finally apply the closurestatus guard and candidate-SN handoff already present in the chapter. They keep the obstruction row and structural coordinates visible before any candidate-mediated strong-normalization or confluence consumer can use the route. Hence the seed closure marker remains a finite frontier registration surface, not residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient equality, or kernel consistency. □

Theorem 4238.192 (MetaCIC parallel-diamond frontier seed marker route)**.** *The mature and bridged hub markers for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *are read only through the child carrier, obligation surface, seed status, and closurestatus guard. Concretely, any reader entering from the hub's mature-status row or bridged-status row must first expose the carrier packet of Definition 4238.1, the obligation rows of Definition 4238.2, the seed closure status of Definition 4238.45, and the guard route of Definition 4238.180. The marker read therefore remains a finite residual-premise packet read, not a discharge of residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the hub marker rows. They name a status entry for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, but the hub has no independent carrier coordinate and no proof body beyond its ordered child inputs.

Read the child carrier and obligation theorem next. The carrier displays the finite rows $P, K, J, R, S, C, B, O, H, T, G, N$, while the obligation theorem identifies the residual-premise, critical-pair, candidate-join, residual-frontier, bounded checker, obstruction, transport, replay, provenance, and local naming obligations exposed by those rows.

Finally apply the seed closure status and closurestatus guard. These force the status reader to consume the seed route and the guarded closurestatus export before using the mature or bridged marker. Since the exposed packet contains no coordinate for residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, the hub marker exports only the child finite residual-premise status route. □

Theorem 4238.193 (MetaCIC parallel diamond frontier obligation-closure audit)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the residual-join, candidate-normalization, retained-premise, and seed-readiness rows form the finite obligation surface a local confluence consumer must read before claiming a parallel diamond frontier. The audit surface is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

with residual joins carrier-local, candidate normalization scoped to the displayed frontier, retained premises still visible, and seed status read only through the packet's closurestatus guard.

Proof. Begin with the seed status theorem, Definition 4238.194. It identifies the frontier seed surface as the finite route through the critical-pair envelope, residual join locality, candidate normalization scope, and finite residual witness rows.

Read residual join locality next, using Definition 4238.104. This keeps every local join inside the displayed packet and prevents the consumer from treating a frontier join as unrestricted confluence.

Attach the finite residual diamond ledger through Definition 4238.123. The ledger supplies the bounded residual-diamond rows and local-join obligations that a confluence consumer must inspect before the frontier can be read.

Finally apply the retained-premise budget and the closurestatus guard, Definition 4238.150 and Definition 4238.180. They keep the retained premise and obstruction rows visible while the seed marker is consumed, so the audit surface remains a finite obligation packet rather than subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4238.194 (MetaCIC parallel-diamond seed status)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the critical-pair envelope, residual join locality, candidate normalization scope, and finite residual witness rows jointly supply the seed-status surface for parallel-diamond consumers. The exported seed surface is the displayed route*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read through Definition 4238.50, Definition 4238.104, Definition 4238.62, and Definition 4238.69. It is a carrier-local seed-status certificate, not unrestricted confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.

Proof. Project the accepted frontier carrier. The critical-pair envelope exposes the bounded overlap and candidate-join rows while keeping the residual frontier, closed-checker, obstruction, transport, replay, provenance, and local naming rows visible.

Next read residual join locality. That theorem keeps residual joins inside the same displayed packet and prevents a downstream consumer from treating the local join as global confluence or premise-free residual substitution.

Now attach candidate normalization scope and finite residual witnesses. Candidate normalization keeps J in the conditional spine with P and O visible, while finite residual witness consumption supplies the typed-window, bounded-check, transport, replay, provenance, and local $\mathsf{NameCert}$ rows needed before a parallel-diamond read can proceed.

The four rows therefore assemble exactly the seed-status surface over $P, K, J, R, S, C, B, O, H, T, G, N$. Since each component is a carrier projection and the ledger exposes no outside equality or kernel theorem, replay and provenance cannot escape the displayed seed packet. □

Theorem 4238.195 (MetaCIC parallel-diamond seed-status exactness)**.** *Any consumer of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *seed-status surface must preserve the same critical-pair envelope* P, K*, the same residual-join rows* J, R, S*, the same candidate-normalization row* C*, the same bounded residual witness rows* B, O*, and the same structural rows* H, T, G, N*. Changing any one of these displayed rows is not the same seed-status packet.*

The exactness claim is only packet-local. It does not assert unrestricted confluence, evaluator equality, quotient term equality, subject reduction, or kernel consistency.

Proof. Apply Definition 4238.194. The seed-status surface exported there is the displayed packet route $P, K, J, R, S, C, B, O, H, T, G, N$ assembled from the critical-pair envelope, residual locality, candidate normalization, and finite witness consumption rows.

Each of those rows is a carrier projection. The critical-pair envelope fixes P, K; residual locality fixes J, R, S; candidate normalization fixes C while keeping the obstruction row visible; finite witness consumption fixes B, O together with the transport, replay, provenance, and local naming rows H, T, G, N.

Therefore a downstream consumer that changes any listed row has changed the packet being consumed. Since the route never exports a global evaluator, quotient equality, subject-reduction theorem, or kernel-consistency theorem, the exactness surface remains a finite seed invariant rather than a global confluence result. □

Theorem 4238.196 (MetaCIC parallel-diamond residual-budget seed status)**.** *For every accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet, the residual-budget seed read is exposed only through the existing critical-pair envelope, residual-join locality, candidate-normalization scope, bounded-check rows, obstruction row, componentwise transport, displayed replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelD}}$ *rows. The seed-facing route is the carrier-local packet*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read through the hub children for critical pairs, residual joins, candidate normalization, finite residual witnesses, retained premises, scoped seed packages, and seed-status registration.

The route explicitly refuses subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, residual-substitution discharge, and kernel consistency.

Proof. Project the accepted frontier carrier. The critical-pair children expose P, K as the bounded overlap and local join envelope before any residual-budget consumer reads a seed status.

Read the residual-join and candidate-normalization children next. They keep J, R, S, C inside the same displayed packet and make the obstruction row O visible rather than discharging it as a global residual-substitution theorem.

The bounded-check and finite-witness children then expose B together with the same transport, replay, provenance, and local naming rows H, T, G, N. These rows record the finite residual-budget route; they do not add a subject-reduction theorem, unrestricted confluence principle, evaluator equality, quotient term equality, global-normalization theorem, residual-substitution discharge, or kernel-consistency certificate. □

Theorem 4238.197 (MetaCIC parallel-diamond seed closure review)**.** *Every seed-level read of an accepted* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *packet must cite the critical-pair envelope, residual-join locality, candidate-normalization scope, finite-residual witness, retained-premise boundary, and closurestatus child routes before the packet can be consumed as a parallel-diamond frontier seed surface. The read is the finite route through* $P, K, J, R, S, C, B, O, H, T, G, N$*; it is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.*

Proof. Begin with the critical-pair and candidate-normalization rows. The critical-pair envelope and candidate-normalization scope keep K, J, R, S, C, B, O attached to the retained premise row P before any frontier consumer reads the candidate join.

Next read the residual and retained-premise routes. Residual-join locality, finite-residual witness consumption, and retained-premise boundary theorems keep the local join, residual witness, obstruction, and premise rows in the same finite packet.

Finally use the closurestatus child routes. They force every seed-status read through the displayed carrier, obligation, closurestatus export, and guard children, so the structural rows H, T, G, N preserve only the listed seed surface and cannot promote it to any refused global MetaCIC endpoint. □

Theorem 4238.198 (MetaCIC parallel-diamond frontier current seed closure route)**.** *The current seed closure route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is read through the existing frontier carrier, the checked obligation surface, and the seed-status child packet already attached to the hub. A seed consumer must expose the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, bounded checker, obstruction ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before the hub marker can be consumed as a seed closure route.*

This route registers only the paper-side seed closure over the displayed carrier and obligation surface. It does not claim subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Read the frontier carrier of Definition 4238.1. Its rows provide the retained premise, local critical-pair data, candidate join data, residual frontier, bounded checker, obstruction ledger, transport, replay, provenance, and local naming surface.

Apply the obligation surface of Definition 4238.2. The obligation route keeps every seed-facing read on those displayed rows, so the hub marker cannot detach into an unrestricted confluence or evaluator-equality assertion.

Finally compare the seed-status child packet with the hub marker. The child packet records the same carrier and obligation inventory as a current seed route, while the exclusions remain outside the carrier coordinates and outside the local $\mathsf{NameCert}$ surface. □

Theorem 4238.199 (MetaCIC parallel-diamond frontier seed child route)**.** *Every residual-diamond consumer that treats the* MetaCICParallelDiamondFrontier$^{\uparrow}$ *seed route as public must first read the child carrier, the NameCert obligation surface, the closurestatus export row, and the closurestatus guard. In carrier notation the consumer reaches the seed route only through the displayed packet*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and through Definition 4238.1, Definition 4238.2, Definition 4238.178, and Definition 4238.180. The route does not export residual-substitution discharge, unrestricted subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.

Proof. Start with the hub shape. The parent file contributes only ordered child inputs, so any public seed read must enter through a child packet rather than through an independent hub theorem.

Project the frontier carrier and obligation theorem next. They expose the residual-substitution premise, critical-pair envelope, residual-join rows, candidate-normalization row, bounded witness rows, obstruction row, transport, replay, provenance, and local naming as the finite route $P, K, J, R, S, C, B, O, H, T, G, N$.

Finally apply the closurestatus export and guard. The export row identifies the seed route as a status read of the same packet, and the guard forces residual-diamond consumers to keep that packet visible before treating the route as public. Since the packet has no coordinate for the excluded global endpoints, the seed child route remains a finite frontier handoff. □

Theorem 4238.200 (MetaCIC parallel-diamond seed-axis rotation record)**.** *Every seed-axis status read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *enters through the child seed route before it can be consumed as a frontier status. The route first exposes the seed-readiness row of Definition 4238.83, then the seed-obligation surface of Definition 4238.89, and finally the seed-closure status of Definition 4238.45. It remains a finite residual-premise frontier read and does not discharge residual substitution, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the seed-readiness row first. It displays the residual branches, candidate join, normalization-confluence handoff, and critical-path socket before any frontier claim is consumed.

Next apply the seed-obligation surface. The obligation row keeps $P, K, J, R, S, C, B, O, H, T, G, N$ visible as the seed packet, so the status read cannot detach a diamond conclusion from its retained premise and obstruction rows.

Finally use the seed-closure status. It records the same seed packet as the closure-facing route, with transport, replay, provenance, and local naming still present. The rotation record is therefore only a child status read of the displayed packet, not an added global confluence endpoint. □

Theorem 4238.201 (MetaCIC parallel-diamond current seed-axis record)**.** *The current seed-axis read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *enters through the displayed carrier of Definition 4238.1 and the checked obligation surface recorded by* BEDC.Derived.MetaCICParallelDiamondFrontierUp.MetacicParallelDian *It exposes the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment, bounded-check row, obstruction row, transport, replay, provenance, and local naming before any seed-facing consumer can inspect the hub status route.*

Proof. Read the carrier first. Its visible rows are exactly the retained premise, critical-pair, candidate join, residual frontier, checker, fragment, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Next apply the obligation surface. The checked obligation theorem keeps the read on those displayed rows, so the seed-axis status cannot detach from the retained premise or obstruction data.

Finally compare with the existing seed-axis rotation and seed-status routes. They consume the same child packet, so the current record states only the present hub-to-child route and does not claim residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 4238.202 (MetaCIC parallel-diamond frontier hub seed-status route)**.** *Every seed-status read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that enters through the hub is mediated by the existing finite frontier packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route first displays the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming rows, then reads the seed-status children already attached to the hub. It is a seed-facing frontier registration only; it is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the carrier and obligation children of the hub. They expose the retained premise, critical-pair row, candidate-join row, residual-frontier row, checker row, bounded-check row, obstruction row, transport, replay, provenance, and local naming row before a status reader reaches the frontier.

Next consume the seed-readiness, seed-obligation, and seed-status children. These children keep the read on the same displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$ and therefore make the seed status a finite carrier-local record.

Finally compare with the closurestatus child already present in the hub. The route records access to the seed-facing packet only, so no downstream consumer can read it as global MetaCIC confluence, subject reduction, quotient equality, residual-substitution discharge, or kernel consistency. □

Theorem 4238.203 (MetaCIC parallel-diamond seed-status axis read)**.** *Every seed-status read of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *that enters through the hub is routed through the same child status packet before it can inspect the frontier closurestatus. The visible seed packet is the finite row surface*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

as displayed by Definition 4238.205, Definition 4238.208, and Definition 4238.211. This is a seed-axis interpretation of the existing child status rows only; it does not add residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, kernel consistency, or a second closurestatus block.

Proof. Start with the seed-status block sync. It exposes the retained residual premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows before a status reader can use the frontier.

Next read the seed-status register. The register identifies the hub entry as a route into the same child packet, not as a detached confluence or verification claim.

Finally use the seed status closure block. It keeps the seed-facing read on $P, K, J, R, S, C, B, O, H, T, G, N$ and leaves the closurestatus body to the existing closurestatus child. Hence the hub input records only the child-axis route and exports no additional MetaCIC endpoint. □

Theorem 4238.204 (MetaCIC parallel-diamond frontier seed-status canonical reader)**.** *The seed-status record for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is read only through the displayed child packet consisting of the critical-pair envelope, residual-substitution dependency route, candidate-normalization scope, finite residual witness, and local* NameCert *rows. A consumer reaches that record by passing through Definition 4238.50, Definition 4238.57, Definition 4238.62, Definition 4238.69, and Definition 4238.2 before using the seed-status rows of Definition 4238.203 and Definition 4238.215.*

Thus the canonical seed-status reader records the same finite frontier packet $P, K, J, R, S, C, B, O, H, T, G, N$. It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with the critical-pair envelope and residual-substitution dependency route. These rows keep the retained residual premise visible while the frontier is read, so the status record cannot be consumed as a detached diamond theorem.

Next read the candidate-normalization scope and finite residual witness. They restrict the status record to the displayed candidate window and finite residual ledger rather than to global normalization or unrestricted confluence.

Finally use the local NameCert obligation row and the existing seed-status children. They identify the seed-status record as a route through the same child packet $P, K, J, R, S, C, B, O, H, T, G, N$, with no additional endpoint beyond the frontier-local status reader. □

Theorem 4238.205 (MetaCIC parallel-diamond seed-status block sync)**.** *Every seed-status read of* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is synchronized only through the displayed residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local* NameCert *rows*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The status block may be consumed as a seed-facing frontier record only after those rows have been displayed in this order. It is not residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Begin with the seed-readiness route of Definition 4238.83. It exposes the residual branches, common join candidate, normalization-confluence handoff, and critical-path socket before any frontier status can be read.

Next apply the seed-obligation surface of Definition 4238.89. That theorem displays exactly the residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows as the seed packet.

Finally use the seed-closure status of Definition 4238.45. The closure-facing read preserves the same route $P, K, J, R, S, C, B, O, H, T, G, N$, so the status block is synchronized through the finite frontier rows and cannot be reread as residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency. □

Theorem 4238.206 (MetaCIC parallel-diamond seed-status paper read)**.** *The paper-side seed-status read for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *records only the local pair* seed/unformalized *along the child route displayed in Definition 4238.205. The read is a seed-facing frontier marker over*

$$P, K, J, R, S, C, B, O, H, T, G, N$$

and not a claim of residual-substitution discharge, subject reduction, Church–Rosser, quotient term equality, global normalization, or kernel consistency.

Proof. Project the seed-status block through the theorem above. Its route is already confined to the displayed residual premise, critical-pair, candidate-join, residual-frontier, closed-checker, closed-normal, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Read the pair seed/unformalized as a paper-side marker for that child route. Since the marker is consumed only after those rows are visible, it cannot become a verification claim or a global MetaCIC endpoint. □

Theorem 4238.207 (MetaCIC parallel diamond frontier seed-status child current)**.** *The hub-level seed status for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is readable only through the current child closurestatus and status records together with the finite residual-join packet. It is not a claim of global confluence, subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Read the seed-status current-axis, canonical-reader, and block-sync routes of Definition 4238.203, Definition 4238.204, and Definition 4238.205. These child records are the only public route by which the hub can report the current seed-status row.

Apply the closurestatus guard route Definition 4238.180. It keeps the hub read tied to recorded child closurestatus evidence rather than to an ambient metatheorem.

Finally use the residual-join obligation surface Definition 4238.138. The finite residual-join packet supplies the local critical-pair and residual-join rows consumed by the seed-status record, while the excluded global claims are not fields of that packet. □

Theorem 4238.208 (MetaCIC parallel-diamond frontier seed status register)**.** *The seed route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is registered through the existing carrier and obligation children of this hub. A seed-facing reader first consumes the carrier of Definition 4238.1, the obligation surface of Definition 4238.2, the seed-readiness route of Definition 4238.83, and the seed-status block sync of Definition 4238.205. The registered seed surface is the displayed frontier packet over*

$$P, K, J, R, S, C, B, O, H, T, G, N,$$

read at seed/unformalized *for seed consumers only. It is not global confluence, full Church–Rosser, unrestricted subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier and obligation surface already input by the hub. These rows expose the retained residual premise, critical-pair envelope, candidate join, residual frontier, closed checker, closed-normal fragment, bounded check, obstruction, transport, replay, provenance, and local naming route.

Use the seed-readiness and seed-status block sync children next. They route seed-facing reads through the same displayed rows $P, K, J, R, S, C, B, O, H, T, G, N$ before any status reader can inspect the frontier.

The register therefore records only that this child route is available to seed consumers. Since every cited row is carrier-local and the packet has no coordinate for unrestricted confluence, Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency, the registration does not claim any of those endpoints. □

The to-seed registration row for this child route is seed/unformalized with bridge status none. It records only the seed-facing read of Definition 4238.208; it does not claim full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Theorem 4238.209 (MetaCIC parallel-diamond frontier seed-status registration)**.** *The seed-status route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *registers the parallel-diamond frontier through the already displayed finite packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The registration reads the retained premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming before exposing the seed-facing status children. It is a frontier-local registration only; it does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the hub carrier and obligation children. They expose the retained residual premise, critical-pair row, candidate-join row, residual-frontier row, checker, bounded-check row, obstruction, transport, replay, provenance, and local naming rows before any status reader can enter the seed route.

Next use the seed-readiness, seed-status, and seed-axis children. These children keep the read on the same finite packet $P, K, J, R, S, C, B, O, H, T, G, N$ and record it as a seed-facing registration over the frontier.

Finally compare with the closurestatus and bridge-status children already routed through the hub. The registration names only the frontier-local seed status, so the refused MetaCIC endpoints remain outside the packet. □

Theorem 4238.210 (MetaCIC parallel diamond frontier seed readiness status)**.** *The* $\mathsf{MetaCICParallelDiamondFront}$ *seed surface is exactly the finite route assembled from residual-window exposure, candidate-normalization rows, residual-substitution rows, the local diamond ledger, componentwise transport, continuation replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}}$ *row already exposed by the sibling children of this hub. No additional global confluence principle, quotient rewrite relation, or host choice row is exported at the seed boundary.*

Proof. Read the hub children in the displayed route order. The finite residual witnesses and residual-window children expose the window data, while the candidate-normalization and residual-substitution children provide the local normal-form and substitution rows consumed by the frontier packet.

The local diamond, transport, and replay children then route the selected residual pair through the same finite ledger. These rows are still seed-facing: they name the residual join data visible to the packet and do not assert a global confluence theorem for the ambient calculus.

The provenance and NameCert children close the seed surface by recording the source packages and the local certificate row. Since every consumer reaches the packet through those children, the seed readiness status is precisely the listed finite route and has no hidden quotient, choice, or global diamond coordinate. □

Theorem 4238.211 (MetaCIC parallel-diamond frontier seed status closure block)**.** *The seed-facing status read for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is exactly the finite residual-substitution premise, critical-pair, candidate-join, residual-frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Any consumer that reads this status must preserve the same displayed route before confluence, normalization, or kernel-facing language is public. The block is a seed-status certificate only; it

is not full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.

Proof. Project the carrier and obligation surface already routed through the hub. They display the residual-substitution premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming rows.

Use the seed-readiness and seed-obligation children. Those routes place $P, K, J, R, S, C, B, O, H, T, G, N$ before any seed-facing reader can inspect the frontier status.

The status block therefore records only the local seed route. Since no listed row supplies full Church–Rosser, subject reduction, conversion decidability, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency, none of those endpoints is exported by the block. □

Theorem 4238.212 (MetaCIC parallel-diamond frontier seed status synchronization)**.** *The hub-level seed record, this child closure block, the retained-premise boundary, the residual-join route, and the candidate-normalization socket expose the same* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *seed-status packet before any later scoped or bridge claim is read. In each location the visible packet is the residual-substitution premise, critical-pair envelope, candidate join, residual frontier, checker, bounded-check, obstruction, transport, replay, provenance, and local naming route*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

Proof. Start with the seed status closure block above. It identifies the seed-facing status as the displayed packet $P, K, J, R, S, C, B, O, H, T, G, N$ and refuses every confluence, normalization, conversion, quotient, residual-discharge, and consistency endpoint outside that packet.

The hub-level seed record only routes readers into the same children. The retained-premise boundary preserves the residual-substitution premise row P, while the residual-join route preserves the critical-pair, candidate-join, and residual-frontier rows K, J, R.

The candidate-normalization socket then reads the checker, bounded-check, and obstruction rows S, C, B, O without adding a public normalization theorem. Transport, replay, provenance, and local naming remain the shared rows H, T, G, N.

Thus all listed sites expose the same seed-status packet before any scoped or bridge-facing claim is available. □

Theorem 4238.213 (MetaCIC parallel-diamond frontier seed-status child route)**.** *The current seed-status child route for* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *is read through the hub only after the carrier, obligation, seed-readiness, seed-status block sync, seed-status register, and seed-status closure block children expose the finite frontier packet*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

The route records a seed-facing child status over the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows. It is not residual-substitution discharge, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Read the carrier and obligation children first. They display the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows before the hub can route any status reader into a child file.

Next use the seed-readiness and seed-status block sync children. These keep the status read on the same packet $P, K, J, R, S, C, B, O, H, T, G, N$ and mark it as seed-facing rather than as a scoped, public, or bridged endpoint.

Finally compare with the seed-status register and seed-status closure block. They identify the hub input as a route into existing child status evidence, so the hub does not acquire theorem or proof content of its own and the child route exports none of the refused MetaCIC endpoints. □

Theorem 4238.214 (MetaCIC parallel-diamond frontier seed rotation status)**.** *The seed registration route for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *reads the existing* `MetacicParallelDiamondFrontierObligations` *surface before any seed-facing status is consumed. A reader enters through the hub marker, then exposes the carrier of Definition 4238.1, the obligation rows of Definition 4238.2, the seed-readiness route of Definition 4238.83, and the seed-status child route of Definition 4238.213. The resulting status is a seed-facing route through*

$$P, K, J, R, S, C, B, O, H, T, G, N.$$

It does not discharge residual substitution, subject reduction, unrestricted confluence, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the hub marker and the local obligation theorem. The marker names the seed-facing status row, while Definition 4238.2 displays the residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Next route the read through seed readiness and the seed-status child. These children keep the status on the same finite packet $P, K, J, R, S, C, B, O, H, T, G, N$ and prevent the hub from carrying theorem content of its own.

The seed rotation status is therefore only a child route into existing packet evidence. Since the displayed rows retain the residual premise and obstruction coordinates, the route cannot be read as any of the refused MetaCIC endpoints. □

Theorem 4238.215 (MetaCIC parallel-diamond frontier status canonical)**.** *The canonical paper-side seed record for* MetaCICParallelDiamondFrontier$^{\uparrow}$ *is the seed closurestatus read over the parallel-diamond frontier obligation surface named by* MetacicParallelDiamondFrontierObligations*. A consumer that enters through the hub must read the carrier of Definition 4238.1, the obligation row of Definition 4238.2, and the seed-status routes of Definition 4238.213 and Definition 4238.214 before treating the seed record as available.*

This status record is a seed boundary for the displayed frontier obligations. It does not by itself prove the substitution-sensitive diamond, subject reduction, evaluator equality, quotient term equality, unrestricted confluence, global normalization, or kernel consistency.

Proof. Project the hub marker and the displayed frontier carrier. These rows expose only the retained residual premise, critical-pair envelope, candidate join, residual frontier, checker, bounded check, obstruction, transport, replay, provenance, and local naming data.

Next read the existing seed-status child routes. They keep the status on the same finite packet and route the paper-side seed record through the obligation surface already named by MetacicParallelDiamondFrontierObligations.

The record therefore binds a seed closurestatus over those obligations only. Since no row removes the residual-substitution premise or supplies a global join theorem, the substitution-sensitive diamond and the other MetaCIC endpoints remain outside the claim. □

Theorem 4238.216 (MetaCIC parallel-diamond frontier seed-axis summary)**.** *Every seed-axis consumer of the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *hub must read the seed route through the child rows of Definition 4238.89, Definition 4238.83, Definition 4238.194, and Definition 4238.215. The parent hub contributes only ordered input routing; it has no independent carrier coordinate, proof body, residual-substitution discharge, subject-reduction endpoint, unrestricted confluence endpoint, evaluator equality, quotient term equality, global normalization, or kernel-consistency row.*

Proof. Start with the seed-obligation child row. It exposes the retained premise, critical-pair row, candidate-join row, residual-frontier row, checker rows, obstruction ledger, transport, replay, provenance, and local naming surface before any seed-facing read is available.

Next use the seed-readiness and parallel-diamond seed-status rows. They keep the same finite packet visible as a seed route and do not move the retained residual-substitution premise into a discharged endpoint.

Finally compare the canonical status row with the parent input list. Since the hub supplies only an ordered route into child files, any seed-axis read factors through those child rows and exports none of the refused MetaCIC endpoints. □

Chapter 4239

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$

Following Bishop and Bridges and Bourbaki on separated completions and Hausdorff uniform filters, the $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ packet records the finite route by which a Cauchy-filter completion surface becomes separated before a Real-facing completion consumer reads it. It sits after the Hausdorff completion surface of chapter 402, the separated completion surface of chapter 1612, the Cauchy-filter completion surface of chapter 1569, the metric completion surface of chapter 1506, and the terminal real-name seal of chapter 38. The packet names the separated-filter completion handoff as finite BEDC data; it is not a quotient-filter construction, a selected representative, a countable-choice limit selector, host equality of completion points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4239.1 (Hausdorff completion filter separation carrier)**.** *A* $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ *carrier is a finite packet*

$$(F, H, S, C, M, R, T, Q, P, N).$$

Here F is the Cauchy-filter completion row, H is the Hausdorff separated row, S is the separated-completion handoff row, C is the completion-continuation row, M is the $\mathsf{MetricCompletion}^{\uparrow}$ *seal, R is the* $\mathsf{Real}^{\uparrow}$ *seal, T is componentwise* hsame *transport over the filter, separation, completion, metric, and real rows, Q is displayed* Cont *replay, P is* Pkg *provenance over the cited sibling certificates and BEDC structural rows, and N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionFilterSeparationUp.HausdorffCompletionFilterSeparationTasteGate`
`single carrier alignment`

Theorem 4239.2 (Hausdorff completion filter separation obligations)**.** *For the carrier of Definition 4239.1, the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$ *surface consists exactly of six finite obligations: carrier admission for the displayed Cauchy-filter completion row F; Hausdorff separation compatibility for H; separated-completion handoff exactness for S and C; MetricCompletion and Real seal agreement for M and R; componentwise transport and replay stability for T and Q; and provenance plus local naming through P and N.*

Proof. Project the carrier. The only completion source row before separation is F, so every completion read begins on the Cauchy-filter completion surface named by chapter 1569.

Read the separation rows next. The Hausdorff row H and separated-completion handoff row S force zero-distance or indistinguishability comparisons through the separated surfaces of chapter 402 and chapter 1612.

Attach the completion seals. The continuation row C, $\mathsf{MetricCompletion}^{\uparrow}$ seal M, and $\mathsf{Real}^{\uparrow}$ seal R keep the separated filter completion connected to the metric-completion and real-name boundaries rather than to host equality of quotient classes.

Finally transport and replay the same finite packet through T and Q, and record provenance and local naming through P and N. Since no coordinate stores a quotient filter, a selected representative, a countable-choice selector, host equality, or ambient completeness, the listed obligations are the whole local certificate surface. □

Theorem 4239.3 (Hausdorff completion filter separation)**.** *Every accepted* $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ *carrier* $(F, H, S, C, M, R, T, Q, P, N)$ *presents a separated Cauchy-filter completion route: a completion consumer must first read* F*, then the Hausdorff and separated handoff rows* H, S, C*, then the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals* M, R*, and only then the structural rows* T, Q, P, N*. The exported completion point is therefore separated at the displayed filter boundary and cannot be read through quotient filters, selected representatives, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from carrier admission. The Cauchy-filter completion row F supplies the only completion source, so a consumer has no route to a completed object before the filter-completion surface is displayed.

Apply the local NameCert obligations. They place H, S, C between the filter source and the completion seals, making Hausdorff separation and separated-completion handoff part of the same finite route.

Read the two seals in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal M records the metric-completion side, and the $\mathsf{Real}^{\uparrow}$ seal R records the real-name boundary consumed by later Real and Completion rows.

The structural rows do not add new completion data. Componentwise hsame transport T, displayed Cont replay Q, Pkg provenance P, and local naming N preserve the same packet, so the exported route is exactly the separated Cauchy-filter completion route and no stronger ambient principle is available. □

Theorem 4239.4 (Hausdorff completion filter separation stability)**.** *For every accepted* HausdorffCompletionFilterS *carrier* $(F, H, S, C, M, R, T, Q, P, N)$*, componentwise* hsame *transport preserves the displayed order*

$$F \longmapsto H, S, C \longmapsto M, R \longmapsto T, Q, P, N.$$

Hausdorff separation therefore remains upstream of the $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seals whenever the packet is transported or replayed.*

Proof. Project the carrier of Definition 4239.1. The Cauchy-filter completion source is F, the separation and separated-completion rows are H, S, C, and the completion-facing seals are M, R.

Use Definition 4239.2. Its transport and replay clause assigns T and Q to the same finite packet rather than to a new completion source.

Thus transporting the packet preserves the route from F through H, S, C before M, R are read. The structural rows T, Q, P, N record stability, replay, provenance, and naming after the separation rows have already been displayed. □

Theorem 4239.5 (Hausdorff completion filter Real seal non-escape)**.** *For every accepted* HausdorffCompletionFilterS *carrier* $(F, H, S, C, M, R, T, Q, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R *cannot be read as a quotient-filter construction, selected representative, countable-choice limit selector, host equality of completion points, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. It seals only the displayed completion route that has already passed through* F, H, S, C, M *and the structural rows* T, Q, P, N*.*

Proof. Project the same carrier. The only Real-facing coordinate is R, and it appears after the Cauchy-filter source, Hausdorff row, separated-completion handoff, and metric-completion seal.

Apply Definition 4239.2. The obligations identify R as a seal agreement row for the displayed packet, while the transport, replay, provenance, and naming rows preserve that packet without adding new Real data.

No coordinate stores quotient-filter construction, a selected representative, a countable-choice limit selector, host equality, or ambient Real completeness. The seal therefore cannot escape the displayed separated-filter route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}$ packet is generated from Cauchy-filter completion data, Hausdorff separation data, separated-completion handoff data, MetricCompletion and Real seals, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4239.1, Definition 4239.2, and Definition 4239.3 state the finite separated-filter completion certificate surface over the displayed completion, separation, seal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionFilterSeparation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient-filter construction, selected representatives, countable choice, host equality, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, Hausdorff separation compatibility, separated-completion handoff exactness, seal agreement, transport stability, and ledger non-escape for the same finite packet.

Chapter 4240

A Concrete Naming Certificate for $\mathsf{QRDecomposition}^{\uparrow}$

Following Golub–Van Loan and Gram–Schmidt finite-dimensional linear algebra, the $\mathsf{QRDecomposition}^{\uparrow}$ packet records a displayed QR factorization route as finite BEDC data over $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{GramSchmidt}^{\uparrow}$, and $\mathsf{Determinant}^{\uparrow}$. It consumes the matrix carrier of chapter 49, the vector-space rows of chapter 47, the scalar rows of chapter 45, the orthogonalization packet of chapter 1875, and the determinant boundary of chapter 84. The packet names only the finite factorization ledger; it is not a host-level theorem that every matrix admits a selected factorization, nor does it export spectral data, polar decomposition, rank choice, quotient equality, or an ambient norm-completion principle.

Definition 4240.1 (QR decomposition carrier)**.** *A* $\mathsf{QRDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, Q_0, R, G, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the input* $\mathsf{Mat}^{\uparrow}$ *row, including the finite column table, scalar-source row, and vector-space coordinate ledger to be factorized.*
2. *Q_0 is the orthonormal-column packet, listing the displayed output columns together with their inherited* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Field}^{\uparrow}$*, and inner-product support rows.*
3. *R is the upper-triangular packet, recording the finite coefficient table, zero-below-diagonal rows, diagonal pivot witnesses or refusal rows, and scalar provenance.*
4. *G is the* $\mathsf{GramSchmidt}^{\uparrow}$ *handoff row that reads the columns of A through the orthogonalization route before any Q_0 or R read is exposed.*
5. *T is the triangular and reconstruction ledger. It records that every displayed column of A is replayed from the corresponding finite combination of Q_0 columns with coefficients from R.*
6. *H records componentwise* hsame *transport for the input matrix, orthonormal-column packet, upper-triangular packet, Gram–Schmidt handoff, triangular and reconstruction ledger, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$A \longmapsto G \longmapsto Q_0, R \longmapsto T \longmapsto H.$$

8. *P is the* Pkg *provenance over* $\mathsf{QRDecomposition}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{GramSchmidt}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Q_0, R, G, T, H, C, P, N$. It has no coordinate for a selected global basis, spectral factorization, polar decomposition, rank choice, quotient equality, or a host matrix theorem outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.QRDecompositionUp.QRDecompositionTasteGate single carrier alignment`

Theorem 4240.2 (QRDecompositionUp NameCert obligations)**.** *For every accepted* $\mathsf{QRDecomposition}^{\uparrow}$ *carrier*

$$Q = (A, Q_0, R, G, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *input admission for the displayed* $\mathsf{Mat}^{\uparrow}$ *row A and its* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Field}^{\uparrow}$ *support;*
2. *Gram–Schmidt handoff admission for G, requiring the column read of A to pass through the accepted* $\mathsf{GramSchmidt}^{\uparrow}$ *orthogonalization packet;*
3. *orthonormal-column exposure for Q_0, requiring each displayed column to be read from the handoff row rather than from a chosen ambient basis;*
4. *upper-triangular exposure for R, requiring all below-diagonal zero rows, diagonal pivot witnesses or refusal rows, and scalar coefficient rows to be displayed;*
5. *reconstruction exactness for T, requiring the finite column replay of A from Q_0 and R to use only the recorded coefficient rows;*
6. *transport, replay, provenance, and local naming through H, C, P, N, preserving exactly the displayed factorization packet.*

No QR decomposition obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4240.1. The input matrix row A is the only source of displayed columns, and its scalar and vector-space support is inherited from the named dependency rows.

Read the handoff row G. It routes those columns through the $\mathsf{GramSchmidt}^{\uparrow}$ packet, so the orthonormal-column packet Q_0 and coefficient table R cannot be inspected before the finite orthogonalization replay has been displayed.

The upper-triangular obligations are exactly the rows of R: below-diagonal zero entries, diagonal pivot witnesses or refusal rows, and scalar coefficients. The reconstruction ledger T then checks each column of A against the displayed finite combination of Q_0 columns using those same coefficients.

Finally H, C, P, N transport, replay, source, and name the same tuple. Since the carrier has no coordinate for a selected global basis, spectral factorization, polar decomposition, rank choice, quotient equality, or host matrix theorem, the six listed rows exhaust the local naming surface. □

Theorem 4240.3 (Gram–Schmidt handoff for QR decomposition)**.** *Every* $\mathsf{QRDecomposition}^{\uparrow}$ *consumer that reads an orthonormal-column row, an upper-triangular coefficient row, or a reconstruction row must first pass through the* $\mathsf{GramSchmidt}^{\uparrow}$ *handoff row G of Definition 4240.1. Consequently QR reads are finite handoffs from displayed matrix columns to the triangular replay packet, not selected bases or ambient factorization objects.*

Proof. Start with any accepted QR carrier. The only rows that expose Q_0, R, and T are typed after the handoff row G in the displayed route

$$A \longmapsto G \longmapsto Q_0, R \longmapsto T.$$

The handoff row consumes the same matrix columns that A displays and reads them through $\mathsf{GramSchmidt}^{\uparrow}$. Therefore a consumer that asks for an orthonormal column, triangular coefficient, or reconstruction component must first inspect G.

Componentwise hsame transport and Cont replay preserve that order, while Pkg provenance and the local $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$ row name only this route. Thus no selected basis, ambient QR theorem, or external factorization object is exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{QRDecomposition}^{\uparrow}$ packet is generated from a displayed input MatrixUp row, orthonormal-column packet, upper-triangular coefficient packet, GramSchmidtUp handoff, triangular reconstruction ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4240.1, Definition 4240.2, and Definition 4240.3 seed the finite QR factorization packet over displayed matrix, vector-space, field, Gram–Schmidt, determinant, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QRDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global existence of QR decomposition, selected bases, spectral factorization, polar decomposition, rank choice, quotient equality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked input admission, Gram–Schmidt handoff admission, orthonormal-column exposure, upper-triangular exposure, reconstruction exactness, and transport/provenance/name exhaustion for BEDC.Derived.QRDecompositionUp.

Chapter 4241

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$

Following Bishop and Bridges regular Cauchy completeness, the $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ packet records the finite-window handoff from local regular Cauchy sections to a terminal $\mathsf{Real}^{\uparrow}$ seal. It consumes the stream-window carrier of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational classifier of chapter 301, and the terminal real-name surface of chapter 38. The packet names the diagonal gluing route; it is not a quotient of streams, a selected representative, a metric completion object, a countable-choice schedule, or an ambient theorem that all Cauchy data already have host limits.

Definition 4241.1 (Regular Cauchy diagonal seal carrier)**.** *A* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, S, E, R, L, G, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite section family, listing the local regular-Cauchy windows admitted for one completion request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window family shared by the local sections and the diagonal readback.*
3. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the finite precision rows used to compare the local windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity ledger, including the accepted local readback rows and the displayed regularity witnesses.*
5. *L is the local* $\mathsf{Real}^{\uparrow}$ *seal row for each admitted section before diagonal gluing.*
6. *G is the diagonal-gluing ledger. It records the finite refinement route that sends the local F, S, E, R, L rows to one terminal diagonal window.*
7. *H records componentwise* hsame *transport for the finite section family, StreamName windows, dyadic tolerance rows, RegSeqRat classifier and regularity rows, local Real seals, diagonal-gluing ledger, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$F, S, E, R, L \longmapsto G \longmapsto H.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, E, R, L, G, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative, a countable-choice schedule, a host completion point, metric-completeness data, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalSealUp.RegularCauchyDiagonalSealTasteGate single carrier alignment`

Theorem 4241.2 (Regular Cauchy diagonal seal NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ *carrier*

$$D = (F, S, E, R, L, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *finite-section admission for F, requiring every local section to be displayed before the diagonal route is read;*
2. *StreamName and dyadic tolerance compatibility for S, E, requiring all local windows to share the recorded precision ledger;*
3. *RegSeqRat classifier and regularity exposure for R, requiring local readbacks and regularity witnesses to be displayed as finite rows;*
4. *local Real seal exposure for L, requiring each section's terminal read to pass through the same RegSeqRat and tolerance rows;*
5. *diagonal gluing exactness for G, requiring the terminal diagonal window to be assembled from the displayed local rows only;*
6. *transport, replay, provenance, and local naming through H, C, P, N, preserving exactly the same diagonal-seal packet.*

No regular Cauchy diagonal seal obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4241.1. The finite section family F supplies the only local regular-Cauchy windows available to the packet.

The StreamName and dyadic rows S, E place those sections over one displayed precision interface. The RegSeqRat row R then records the classifier and regularity witnesses used by each local readback, and the Real row L records the local terminal seals before they are glued.

The diagonal-gluing ledger G is the only row that assembles a terminal diagonal window from those local data. It therefore carries the exactness obligation for the finite-window handoff and cannot be replaced by a quotient stream, selected representative, or host completion point.

Finally H, C, P, N transport, replay, source, and name the same displayed tuple. Since no further coordinate is present, the six listed rows exhaust the local naming obligations. □

Theorem 4241.3 (Regular Cauchy diagonal seal vision handoff)**.** *The regular Cauchy diagonal seal realizes the finite-window route from local regular sections to the terminal* $\mathsf{Real}^{\uparrow}$ *seal: any read of the terminal seal from a section in F factors through the common* $\mathsf{StreamName}^{\uparrow}$ *window family,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier and regularity rows, local* $\mathsf{Real}^{\uparrow}$ *seal rows, and diagonal-gluing ledger of Definition 4241.1.*

Proof. Start with a terminal seal read from a local section. By carrier admission, the section appears in F and its stream window and tolerance rows are displayed in S and E.

The RegSeqRat classifier and regularity ledger R places the local readback inside the regular rational sequence surface, while the row L records the section's local $\mathsf{Real}^{\uparrow}$ seal. These rows are the only available route from the local section to a terminal real-facing read.

Apply the diagonal-gluing ledger G. It sends the local rows through the shared finite window into the terminal diagonal seal and leaves no coordinate for quotient stream equality, selected representatives, countable-choice schedules, metric-completeness data, completion points, or ambient Real completeness. Thus the vision-side regular Cauchy handoff is represented by the displayed NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$ packet is generated from a finite section family, StreamNameUp window family, DyadicRatCoreUp tolerance ledger, RegSeqRatUp classifier and regularity ledger, local RealUp seal rows, diagonal-gluing ledger, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4241.1, Definition 4241.2, and Definition 4241.3 seed the finite regular-Cauchy diagonal route from local section windows to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalSeal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable-choice schedules, metric-completeness data, completion points, ambient RealUp completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked finite-section admission, StreamName and dyadic tolerance compatibility, RegSeqRat classifier exposure, local Real seal exposure, diagonal gluing exactness, and transport/provenance/name exhaustion for BEDC.Derived.RegularCauchyDiagonalSealUp.

Chapter 4242

Modulus of Total Boundedness NameCert Construction

Following Bishop and Bridges constructive metric analysis, the $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ packet records the explicit schedule form of total boundedness: a positive dyadic tolerance request is answered by a finite epsilon-net row together with the finite ledger proving coverage at that tolerance. The packet sits between the metric carrier of chapter 57, the dyadic tolerance surface of chapter 291, the finite observation interface of chapter 126, the total-bounded finite-net carrier of chapter 127, and the compact-metric consumer of chapter 129. It names the schedule consumed by compact and continuous routes without invoking open-cover compactness, selected representatives, countable choice, or ambient completeness.

Falsifiable prediction: A compact-metric or continuous-route certificate that consumes a total-boundedness claim without displaying the dyadic tolerance request and its finite net schedule refutes $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$, because the packet's carrier has no hidden open-cover compactness or selected-net coordinate.

Independence witness: totally_bounded, compact_metric, stream_name, dyadic_rat_core: none is bijective to $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ because the modulus packet jointly records the tolerance request, schedule index, finite center bundle, coverage ledger, metric comparison rows, transport, replay, provenance, and local naming as one displayed schedule surface.

Definition 4242.1 (Modulus of total boundedness carrier)**.** *A* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, D, S, T, K, L, H, C, P, N)$$

with the following rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point classifier and distance comparison ledger.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive tolerance request.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window naming the displayed tolerance and center schedule.*
4. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net source row.*
5. *K is the finite* $\mathsf{ProbeBundle}$ *of center histories selected by the schedule for the tolerance row D.*

6. *L is the coverage ledger, recording membership, center-carrier rows, distance witnesses, and dyadic comparison rows for the centers in K.*
7. *H records componentwise* hsame *transport for metric, tolerance, window, total-bounded, center-bundle, coverage, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for compact or continuous consumers through $X, D, S, T, K, L, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, D, S, T, K, L, H, C, P, N$. It has no coordinate for an open-cover subcover, a host-selected finite set, a quotient completion, a global compactness theorem, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ModulusOfTotalBoundednessUp.ModulusOfTotalBoundednessTasteGate single carrier alignment`

Theorem 4242.2 (Modulus of total boundedness NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *carrier $M = (X, D, S, T, K, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure, dyadic tolerance exposure, finite-window exposure, total-bounded source exposure, finite center-bundle exposure, coverage-ledger exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 4242.1. Its visible rows are precisely $X, D, S, T, K, L, H, C, P, N$.

The tolerance schedule is read through D and S before any finite-net row is consumed. The total-bounded row T supplies the finite-net source, while K and L display the center bundle and coverage ledger for that tolerance.

The structural rows H, C, P, N transport, replay, source, and name the same finite surface. Since the carrier has no open-cover, selected-set, quotient-completion, choice, or ambient-completeness coordinate, the NameCert obligations are exactly the displayed schedule rows. □

Theorem 4242.3 (Modulus of total boundedness compact handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$ *or continuous-route consumer of an accepted* $\mathsf{ModulusOfTotalBoundedness}^{\uparrow}$ *packet must read the dyadic tolerance row D, the StreamName schedule row S, the* $\mathsf{TotallyBounded}^{\uparrow}$ *source row T, the center bundle K, and the coverage ledger L before it consumes compact finite-net data. The handoff exports a schedule-indexed finite net and no open-cover compactness, selected subcover, countable choice, selected representative, or ambient completeness theorem.*

Proof. Start with the NameCert obligation theorem. It places the tolerance request, finite observation schedule, total-bounded source, center bundle, and coverage ledger among the visible rows of every accepted packet.

A compact-metric or continuous-route read needs the finite-net data. That data occurs only in T, K, L, and those rows are downstream of the dyadic tolerance and schedule rows D, S in the carrier.

Thus the consumer receives a finite dyadic epsilon-net schedule, not an open-cover compactness principle. The excluded coordinates do not occur in the carrier, so they cannot be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfTotalBoundedness}^{\uparrow}}$.

Constructive story: A ModulusOfTotalBoundedness$^{\uparrow}$ packet is generated from a MetricSpace row, a DyadicRatCore positive tolerance request, a StreamName finite schedule, a TotallyBounded finite-net source, a ProbeBundle center row, a coverage ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ModulusOfTotalBoundedness}^{\uparrow}}$, seed)

Scope closed: Definition 4242.1, Definition 4242.2, and Definition 4242.3 seed the explicit dyadic epsilon-net schedule route for total-bounded and compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{ModulusOfTotalBoundedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, selected representatives, quotient completion, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite schedule exposure, total-bounded source exposure, finite center-bundle exposure, coverage-ledger exposure, compact handoff, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfTotalBoundednessUp.

Chapter 4243

A Concrete Naming Certificate for $\mathsf{PolishCauchyBasis}^{\uparrow}$

Following Alexandrov–Urysohn, Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet records the basis-level handoff by which a complete separable metric source exposes Cauchy-filter basis data before any completion consumer reads a limit. It sits between the $\mathsf{PolishSpace}^{\uparrow}$ source of chapter 1490, the $\mathsf{CauchyFilterBasis}^{\uparrow}$ generator of chapter 1501, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff of chapter 128, and the $\mathsf{SeparableMetric}^{\uparrow}$ dense schedule of chapter 1458. The packet is a finite BEDC handoff surface for Polish-space Cauchy bases, not a quotient completion, not a choice-selected dense sequence, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 4243.1 (Polish Cauchy basis carrier)**.** *A* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, B, K, D, S, R, O, H, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row, carrying the complete separable metric packet over which every later basis read is interpreted.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row, recording the directed finite basis generator used by Cauchy-filter consumers.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row attached to the same metric surface as X; it records the completion-facing route without selecting a limit point.*
4. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule row, stored as public finite name data rather than as a choice function over all balls.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect dense names, basis requests, and Cauchy-window demands.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those stream windows, giving regular rational observations before any basis comparison is made.*
7. *O is the observation ledger joining dense-window requests, regular-rational readbacks, Cauchy-basis refinements, and completion-handoff checks.*
8. *H records componentwise* hsame *transport for X, B, K, D, S, R, O and for the local certificate rows.*

9. *C records displayed* Cont *replay from a Polish-source request through dense schedules, stream windows, regular readbacks, Cauchy-basis refinement, and the completion handoff.*

10. *G is the* Pkg *provenance over* $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only after transporting these displayed rows to a common Polish source, Cauchy-basis generator, completion-handoff row, dense schedule, stream-window row, and regular readback ledger. It has no coordinate for choice-selected dense points, quotient completion equality, host point equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PolishCauchyBasisUp.PolishCauchyBasisTasteGate single carrier alignment`

Theorem 4243.2 (Polish Cauchy basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier $P = (X, B, K, D, S, R, O, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Polish source, Cauchy-filter basis, complete-metric handoff, separable dense schedule, stream windows, regular rational readback, observation ledger, transport, continuation, provenance, and local naming rows;*

2. *common-source classification, requiring the B, K, D, S, R, O rows to be read over the same transported* $\mathsf{PolishSpace}^{\uparrow}$ *source row X;*

3. *basis-directedness exposure, requiring every Cauchy-basis refinement consumed by B to be visible in S, R, O before the completion handoff K is read;*

4. *dense-schedule compatibility, requiring the separable row D and the Cauchy-basis row B to share the same stream-window and regular-readback surface;*

5. *completion-handoff non-escape, requiring every completion-facing consumer to pass through B, S, R, O before reaching K;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4293.1. The projection exposes the Polish source, Cauchy-basis generator, complete-metric handoff, dense schedule, stream windows, regular readbacks, observation ledger, transport rows, continuation rows, provenance row, and local name.

The classifier route starts at X. Componentwise transport in H moves B, K, D, S, R, O together, so a consumer never compares a basis row over one Polish source with a completion handoff or dense schedule over another.

The observation ledger O fixes the finite read order. Dense requests and basis refinements are observed through S and R before either the Cauchy-filter basis or complete-metric handoff can be consumed.

Finally replay C and attach G, N. These rows route, source, and name the same displayed packet, with no additional coordinate for selected dense points, quotient completion equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the listed rows exhaust the local naming surface. □

Theorem 4243.3 (Polish Cauchy basis completion handoff)**.** *Let $P = (X, B, K, D, S, R, O, H, C, G, N)$ be an accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *packet. Every completion-facing consumer of the Polish source*

X reaches the $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K only through the displayed Cauchy-filter basis row B, the separable dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger O, transported by H and replayed by C.*

The visible route is

$$X, D, S, R, B, O \longmapsto K \longmapsto H, C, G, N.$$

It exports a finite completion handoff for downstream completion, Baire, and fixed-point packets; it does not export a quotient completion class, a selected dense sequence, host point equality, or an ambient real-completeness principle.

Proof. Read X from the carrier in Definition 4293.1. The Polish source supplies the common complete separable metric surface, while D, S, R display the finite dense-window observations available to the packet.

Use the NameCert obligations of Definition 4293.2. The basis-directedness and dense-schedule compatibility clauses force the Cauchy-filter basis row B to be read through the same stream windows, regular readbacks, and observation ledger as the dense schedule.

The completion-handoff clause then places K after B, S, R, O. Transport by H and replay by C preserve that order, while G, N package and name the same route. Therefore every completion-facing consumer receives only the displayed finite handoff and cannot bypass it through choice, quotient equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet is generated from a PolishSpace source, CauchyFilterBasis basis, CompleteMetric handoff, SeparableMetric dense schedule, StreamName windows, RegSeqRat readback, an observation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, seed)

Scope closed: Definition 4293.1, Definition 4293.2, and Definition 4293.3 seed the basis-level Cauchy handoff from Polish-space data to completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completions, host point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source classification, basis-directedness exposure, dense-schedule compatibility, completion-handoff non-escape, transport, replay, provenance, and local naming for the same packet.

Chapter 4244

A Concrete Naming Certificate for $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$

The $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ packet realizes the hidden Q_6 stabilizer and orbit-persistence rows of Definition 9462.1352, Definition 9462.1354, Definition 9462.1356, and Definition 9462.1362 as finite BEDC NameCert data. It consumes the six-coordinate codon window of chapter 2234, the boundary tile packet of chapter 3393, the codon lift-orbit classifier of chapter 4059, and the median-closure surface of chapter 3545. The packet exposes support, stabilizer, orbit, and median-closure rows before any consumer may claim the hidden C_2 persistence or the restored $S_3 \times C_2$ surface.

Definition 4244.1 (Symmetry restoration stabilizer carrier)**.** *A* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, A, T, U, M, O, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row, including the six bit coordinates inspected by the stabilizer calculation.*
2. *A is the active-support row for $\mathcal{R}$ and its inactive-anchor closure, read through the boundary-tile packet.*
3. *T is the hidden C_2 stabilizer row, naming the identity and the coordinate transposition $\tau : s_1 \leftrightarrow f_2$ from Definition 9462.1352.*
4. *U is the orbit-persistence row, recording the active-support, inactive-anchor, and first-median-layer orbits of τ from Definition 9462.1354.*
5. *M is the median-closure stabilizer row, recording the displayed $S_3 \times C_2$ restoration surface of Definition 9462.1356.*
6. *O is the orbit ledger joining the nontrivial C_2 orbits, the median-closure orbit table, and the sealing-corner orbit-completion row.*
7. *R is the refusal row separating support symmetry from weighted reassignment symmetry, and excluding biological causality, stochastic fitness, host quotient codon semantics, and hidden enumeration oracles.*

8. *H records componentwise* hsame *transport for Q, A, T, U, M, O, R and for the local certificate rows.*

9. *C records displayed* Cont *replay from a support read through the stabilizer row, orbit-persistence row, median-closure row, orbit ledger, and refusal row.*

10. *P is the* Pkg *provenance over* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}$ $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStabilizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, T, U, M, O, R, H, C, P, N$. It has no coordinate for an unstated support set, an unrecorded stabilizer enumeration, a host quotient codon object, a biological mechanism, or a weighted-symmetry substitute for the support stabilizer.

Lean4 ✓ `BEDC.Derived.SymmetryRestorationStabilizerUp.SymmetryRestorationStabilizerTasteGate`
`single carrier alignment`

Theorem 4244.2 (Symmetry restoration stabilizer NameCert obligations)**.** *For every accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier $S = (Q, A, T, U, M, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStab}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Q_6 codon window, active support, hidden C_2 stabilizer, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, transport, continuation, provenance, and local naming row;*

2. *support visibility, requiring every stabilizer or orbit read to expose the active-support row A before reading T, U, M, or O;*

3. *stabilizer exactness, requiring the hidden C_2 row T to name only the identity and $\tau : s_1 \leftrightarrow f_2$ for the active-support and first-median-layer reads;*

4. *median-closure compatibility, requiring the restored $S_3 \times C_2$ row M to be read only after the median-closure surface and its orbit ledger are displayed;*

5. *orbit non-escape, requiring any hidden C_2 orbit claim to pass through $A, T, U, M, O, H, C, P, N$;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4294.1. The projection exposes the codon window, active support, stabilizer row, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, and the structural rows H, C, P, N.

Support visibility comes first. The row A is the source for the stabilizer calculation, so T, U, M, and O cannot be read as free group-theoretic objects detached from the displayed support.

The stabilizer exactness row is T. It records exactly the identity and the transposition $\tau : s_1 \leftrightarrow f_2$ for the active-support surface, while U records the same nonidentity element along the inactive-anchor and first-median-layer rows.

The median-closure compatibility row is M, with orbit data stored in O. Transport by H and replay by C preserve this order, and P, N package and name the same rows. Thus no hidden orbit claim escapes the displayed support, stabilizer, median-closure, transport, replay, provenance, and local naming surface. □

Theorem 4244.3 (Symmetry restoration orbit non-escape)**.** *Every consumer claiming the hidden C_2 orbit of an accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *packet must display the support row A,*

the stabilizer row T, the orbit-persistence row U, the median-closure row M, the orbit ledger O, componentwise transport H, displayed replay C, package provenance P, and local NameCert *row N. The visible route is*

$$Q, A \longmapsto T, U \longmapsto M, O \longmapsto H, C, P, N.$$

It cannot be replaced by a host stabilizer calculation, an unrecorded orbit enumeration, a weighted-reassignment symmetry, quotient codon semantics, or a biological mechanism claim.

Proof. Read Q, A from the carrier in Definition 4294.1. These rows provide the finite codon-window coordinates and the active support over which the stabilizer calculation is meaningful.

Apply the NameCert obligations of Definition 4294.2. Support visibility and stabilizer exactness place T and U after A, while median-closure compatibility places M and O after the displayed closure surface.

Finally use orbit non-escape. The rows H, C, P, N transport, replay, source, and name the same finite route, and the refusal row separates it from weighted reassignment symmetry or host enumeration. Therefore a hidden C_2 orbit consumer must pass through the displayed packet and cannot claim the orbit from outside it. □

Falsifiable prediction: Any consumer claiming the hidden C_2 orbit without displaying support, stabilizer, median-closure, hsame, Cont, Pkg, and NameCert rows refutes the SymmetryRestorationStabilizer$^{\uparrow}$ packet.

Independence witness: q-six orbit/classification, median-closure sibling slugs: SymmetryRestorationStabilizer$^{\uparrow}$ is not bijective to the orbit/classification or median-closure siblings because this packet jointly carries active support, hidden C_2 stabilizer, orbit persistence, restored median-closure stabilizer, orbit ledger, refusal, componentwise transport, replay, provenance, and local naming as one accepted surface.

The Lean TasteGate file for this packet must include Nontrivial and FieldFaithful instances. The field-faithfulness witness keeps $Q, A, T, U, M, O, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out hidden-orbit claims that omit the support, stabilizer, or median-closure coordinates.

Closure status for NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$.

Constructive story: A SymmetryRestorationStabilizer$^{\uparrow}$ packet is generated from Q6 codon-window rows, active-support rows, hidden C2 stabilizer rows, orbit-persistence rows, median-closure stabilizer rows, orbit ledgers, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, seed)

Scope closed: Definition 4294.1, Definition 4294.2, and Definition 4294.3 seed the finite support-stabilizer packet for the hidden Q6 orbit and median-closure restoration surface.

Formal verification: FormalStatus(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological causality, stochastic fitness, quotient codon semantics, host stabilizer enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, support visibility, stabilizer exactness, orbit persistence, median-closure compatibility, orbit non-escape, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.SymmetryRestorationStabilizerUp.

Chapter 4245

A Concrete Naming Certificate for $\mathsf{TaggedRealPartition}^{\uparrow}$

Following Bishop and Bridges constructive integration, the $\mathsf{TaggedRealPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed tag inside each cell. It refines the located partition carrier of chapter 2248 and the oscillation-controlled partition carrier of chapter 3800; it also reads its endpoint and tag data through the $\mathsf{StreamName}^{\uparrow}$ source of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the finite tag-bearing surface needed by L10 integration and oscillation routes without replacing it by an untagged mesh, a host partition object, selected representatives, quotient real equality, countable choice, or ambient completeness.

Definition 4245.1 (Tagged real partition carrier)**.** *A* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, M, A, G, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row exposing the located interval, finite cell order, endpoint containment, and mesh precision.*
2. *M is the finite mesh row inherited from I, listing adjacent cells and the common refinement ledger used by later integration consumers.*
3. *A is the cell-containment row, certifying for every mesh cell that its endpoints and rational witness rows remain inside the located interval source.*
4. *G is the tag row family, assigning to every displayed cell a tag history whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads certify containment in that same cell before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *O is the oscillation and modulus ledger, recording the tolerance request, the cellwise oscillation bound, and the refinement compatibility supplied by* $\mathsf{FiniteOscillationPartition}^{\uparrow}$*.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for interval endpoints, cell endpoints, and tags.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and mesh precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the source interval, cell containment, tag containment, oscillation ledger, window, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for I, M, A, G, O, S, R, E and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tagged-partition read through*

$$I, M, A, G, O, S, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed interval-source, mesh, containment, tag, oscillation, window, readback, seal, transport, replay, provenance, and local naming rows. It has no coordinate for a host tagged partition, an unbounded tag selector, quotient real equality, chosen completed tag values, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.TaggedRealPartitionUp.TaggedRealPartitionTasteGate single carrier alignment`

Theorem 4245.2 (Tagged real partition NameCert obligations)**.** *For every accepted* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier $T = (I, M, A, G, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval and mesh exposure through I, M, cell-containment certification through A, tag containment through G, oscillation and modulus accounting through O, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seals through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every integration-facing or oscillation-facing read of the tagged partition must therefore pass through the displayed finite mesh, tag-containment, oscillation, window, readback, and seal rows before any terminal real-value consumer is allowed to inspect the packet.

Proof. Project the carrier of Definition 4296.1. The tuple lists the only rows available to a tagged-partition consumer: the interval source, mesh, containment ledger, tag rows, oscillation ledger, finite windows, regular readbacks, real seals, transport, replay, provenance, and local naming rows.

The source interval and cell obligations are I, M, A. The located partition row supplies the interval and mesh; the containment row attaches each cell to the same source, so a tag cannot be read against a detached host interval.

The tag and oscillation obligations are G, O. The tag row certifies cell membership for each tag through the same finite-window and regular-readback route, while the oscillation row records the tolerance and refinement ledger inherited from the oscillation-partition packet.

The readback and seal obligations are S, R, E. A consumer reaches the StreamName windows only after the interval, cell, tag, and oscillation rows are visible, then reads the regular rational rows and finally the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name exactly this route. Since the carrier has no coordinate for a host tagged partition, an unbounded selector, quotient real equality, chosen completed tag values, countable choice, or ambient completeness, the displayed NameCert obligations exhaust the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaggedRealPartition}^{\uparrow}$ packet is generated from a LocatedRealIntervalPartitionUp source, finite mesh and cell-containment rows, per-cell tag rows, oscillation and modulus ledgers, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, seed)

Scope closed: Definition 4296.1 and Definition 4296.2 seed the finite tag-bearing partition interface for L10 integration and oscillation consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close integration convergence, Darboux sums, quotient real equality, selected tag values, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cell containment, tag containment, oscillation-modulus accounting, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TaggedRealPartitionUp.

Chapter 4246

A Concrete Naming Certificate for $\mathsf{FiniteFanBarModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges finite-bar analysis, the $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet records the finite carrier that turns admitted fan-bar rows and local modulus rows into a tolerance-only uniform-modulus ledger. It sits over the fan certificate of chapter 400, the compact metric finite-net surface of chapter 129, the continuous-map carrier of chapter 130, the rational lower-bound fold of chapter 4118, the $\mathsf{Rat}^{\uparrow}$ arithmetic source of chapter 37, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. It realizes the compact-uniform-continuity route of chapter 9237 without importing open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, or ambient real completeness.

Definition 4246.1 (Finite fan-bar modulus carrier)**.** *A* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, L, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite fan-bar admission row, recording the fan, decidable bar classifier, finite depth bound, and finite branch-prefix witness window.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-probe row, aligning the bar prefixes with the compact finite-net probes consumed by the compact-uniform-continuity route.*
3. *L is the local modulus row read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *packet at each admitted bar or probe entry.*
4. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *radius and tolerance row, carrying the finitely many local radii and requested tolerances.*
5. *R is the lower-bound modulus extraction row, obtained by primitive recursion over the finite fan-bar and probe bundle and by certified rational comparison in Q.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, exposing the extracted lower bound as a tolerance-only modulus ledger.*
7. *H records componentwise* hsame *transport for the bar, compact-probe, local-modulus, rational, extraction, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$B, K, L, Q \longmapsto R \longmapsto U.$$

9. *P is the* Pkg *provenance over* FiniteFanBarModulus$^\uparrow$, FanTheorem$^\uparrow$, CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, Rat$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, K, L, Q, R, U, H, C, P, N$. It has no coordinate for an open cover, a selected infinite branch, countable choice, quotient equality of maps, selected limits, a host compactness theorem, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteFanBarModulusUp.FiniteFanBarModulusTasteGate single carrier alignment`

Theorem 4246.2 (Finite fan-bar modulus NameCert obligations)**.** *For every accepted* FiniteFanBarModulus$^\uparrow$ *carrier $F = (B, K, L, Q, R, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *surface consists exactly of carrier admission, finite bar admission through B, compact finite-probe alignment through K, local continuous-map modulus exposure through L, positive rational radius accounting through Q, lower-bound modulus extraction through R,* UniformModulus$^\uparrow$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

In particular, a compact-uniform-continuity consumer may read a uniform modulus from this packet only after the finite bar, compact-probe, local-modulus, rational-radius, and lower-bound extraction rows have all been displayed.

Proof. Project the carrier of Definition 4297.1. The tuple displays exactly the finite bar, compact-probe, local-modulus, rational-radius, lower-bound extraction, uniform-modulus, transport, replay, provenance, and local naming rows.

Finite bar admission is the row B. It supplies the decidable bar classifier and finite depth witness, so the argument ranges over displayed finite prefixes rather than selected infinite branches.

The compact and local-modulus data are K, L, Q. The compact-probe row aligns finite bar prefixes with compact finite-net probes, the continuous-map row supplies local moduli at those probes, and the rational row stores the positive local radii and tolerance requests.

Lower-bound extraction is the row R. It is obtained by primitive recursion over the finite bar and probe rows, using the positive rational comparisons in Q to retain a certified lower bound for all admitted local radii.

Finally, U, H, C, P, N expose, transport, replay, source, and name that extracted lower bound as a UniformModulus$^\uparrow$ ledger. Since the carrier has no coordinate for open-cover compactness, branch selection, countable choice, quotient function equality, selected limits, host compactness, or ambient completeness, the displayed obligations exhaust the accepted packet. □

Theorem 4246.3 (Finite fan-bar lower-bound modulus extraction)**.** *Every compact-uniform-continuity read of an accepted* FiniteFanBarModulus$^\uparrow$ *packet factors through*

$$B, K, L, Q, R, U, H, C, P, N.$$

The extracted row R is the only lower-bound modulus source available to the UniformModulus$^\uparrow$ *consumer: it is computed from finite fan-bar admission, compact finite-probe alignment, local continuous-map moduli, and positive* Rat$^\uparrow$ *radius comparisons, and it exports no open-cover subcover, selected branch, selected limit, quotient map equality, or ambient compactness datum.*

Proof. Start with the NameCert obligations of Definition 4297.2. They require every compact-uniform-continuity consumer to read B, K, L, Q before the lower-bound row R is available.

The extraction step is finite. The bar depth and compact probe rows determine the finite index family, and the positive rational ledger compares the finitely many local radii by primitive recursion to produce a certified lower bound.

Once R is displayed, the uniform-modulus row U may consume it as a tolerance-only ledger. The structural rows H, C, P, N preserve and name the same route under componentwise transport and replay.

No excluded datum can be read from the carrier. Open-cover compactness, selected branches, selected limits, quotient map equality, countable choice, and ambient compactness are absent from $B, K, L, Q, R, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet is generated from finite fan-bar admission, compact finite-probe alignment, local ContinuousMap modulus rows, positive RatUp radius ledgers, lower-bound extraction, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4297.1, Definition 4297.2, and Definition 4297.3 seed the finite fan-bar lower-bound route consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar admission, compact finite-probe alignment, local modulus exposure, RatUp lower-bound extraction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteFanBarModulusUp.

Chapter 4247

A Concrete Naming Certificate for $\mathsf{EdelsteinFixedPoint}^{\uparrow}$

Following Edelstein's compact-metric fixed-point theorem and Bishop–Bridges constructive metric practice, the $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet records the finite NameCert surface for a strictly contractive self-map on a compact metric carrier. It sits after the metric source of chapter 57, the compact-metric finite inspection surface of chapter 129, the sequential map discipline of chapter 1466, the contraction comparison row of chapter 3529, and the displayed iteration route of chapter 2359. The packet names only the compact source, the strict distance-decrease row, the sequential-continuity row, the orbit comparison ledger, transport, replay, provenance, and local naming; it does not import choice, quotient completion, a selected host fixed point, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a Banach contraction ratio.

Definition 4247.1 (Edelstein fixed-point carrier)**.** *An* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, O, C, S, H, R, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier and the distance observations inspected by the fixed-point consumer.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row over X, exposing only the finite compactness windows and cluster-service rows accepted by the compact-metric certificate.*
3. *F is the self-map row, presented through the same metric carrier and never as an untracked host function.*
4. *O is the orbit row generated from F, recording the displayed Picard-style finite prefix consumed by the fixed-point search.*
5. *C is the strict contraction-comparison row: whenever the inspected points are classified as distinct by the metric row, the displayed distance from $F(x)$ to $F(y)$ is strictly below the displayed distance from x to y.*
6. *S is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *row for F, binding cluster reads of the orbit to the image row without adding a semantic continuity theorem outside the displayed packet.*
7. *H stores componentwise* hsame *transport for metric, compactness, self-map, orbit, contraction-comparison, sequential-continuity, continuation, provenance, and naming rows.*

8. *R stores displayed* Cont *replay from the orbit row to the compact cluster row, from the cluster row through sequential continuity, and from the strict comparison row to the fixed-point equality boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, O, C, S, H, R, P, N$. It has no coordinate for quotient completion, countable choice, a Banach contraction constant, an ambient complete real line, or a host-selected fixed point.

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointTasteGate single carrier alignment`

Theorem 4247.2 (Edelstein fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier $E = (X, K, F, O, C, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *surface consists exactly of metric carrier admission through X, compact-metric finite inspection through K, self-map admission through F, Picard-style orbit exposure through O, strict contraction comparison through C, sequential-continuity cluster transport through S, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointCarrier namecert obligations`

Proof. Project the carrier of Definition 4298.1. The metric row X and compact-metric row K are the only source rows available to the certificate, so every inspected point, distance comparison, and compactness read is routed through those two displayed surfaces.

The self-map row F and orbit row O provide the visible finite Picard-style prefix. The strict comparison row C is separate from the Banach contraction surface: it records pointwise distance decrease on distinct displayed pairs and carries no global contraction ratio.

The sequential-continuity row S is the only route from a compact cluster read of the orbit to an image read under F. Transport through H, replay through R, provenance through P, and naming through N preserve this same finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 4247.3 (Edelstein compact-route nonescape)**.** *Every fixed-point read exported by an accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *packet factors through the internal route*

$$X \longmapsto K \longmapsto O \longmapsto S \longmapsto C$$

together with F, H, R, P, N. The route may compare with the $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$ *neighbours as named dependency rows, but it does not consume a Banach contraction constant, quotient completion equality, selected subsequence principle, choice operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a fixed-point read. The carrier first exposes the metric source X and the compactness service K, so the consumer can only inspect displayed finite compact-metric windows and the point classifier attached to X.

The orbit row O supplies the visible Picard-style finite prefix for the self-map F. Compactness is consumed only through K, and the image of a cluster read is transported only through the sequential-continuity row S.

The strict comparison row C is then the only row that can rule out a non-fixed distinct-pair boundary. Since C contains no global ratio and the remaining rows H, R, P, N only transport, replay, source, and name the same finite data, the exported route cannot leave the displayed Edelstein packet for choice, quotient completion, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ fixed-point read factors through one MetricUp source row, one CompactMetricUp finite inspection row, one self-map row, one PicardIterationUp-style orbit row, one strict comparison row, one SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, sequentialcontinuity, banach-contraction, picard-iteration: $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-metric cluster service, strict pointwise distance decrease, and sequential-continuity image transport without a Banach ratio, selected endpoint, quotient completion, or ambient completeness coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite inspection row, a displayed self-map row, a PicardIterationUp-style orbit row, a strict distance-decrease comparison row, a SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4298.1, Definition 4298.2, and Definition 4298.3 bind the compact-metric strict-contraction route through the displayed orbit, cluster, sequential-continuity, and comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, uniqueness, quotient completion equality, Banach contraction constants, selected fixed points, choice principles, bridge checking, Lean verification, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric inspection, orbit exposure, strict comparison, sequential-continuity cluster transport, nonescape, replay, provenance, and local naming for BEDC.Derived.EdelsteinFixedPointUp

Chapter 4248

A Concrete Naming Certificate for $\mathsf{CompletionLadder}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompletionLadder}^{\uparrow}$ packet records the finite BEDC naming surface that ties the current Real and Completion route into one phase object. It sits over the dyadic tolerance ledger of chapter 291, the finite-window stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated-completion boundary of chapter 1612, and the completion-unit handoff of chapter 1527. The ladder is a finite dependency packet for the phase route; it is not quotient stream equality, not a selected limit, not countable choice, not trichotomy, not open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4248.1 (Completion ladder carrier)**.** *A* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, W, R, E, S, U, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row naming the precision requests that enter the route.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row listing the finite observations read at those tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the dyadic tolerance, stream window, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *witness row that consumes the same displayed real-facing route as a separated completion boundary.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *unit row that records the source-to-completion handoff along the same finite route.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, W \longmapsto R \longmapsto E \longmapsto S, U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, S, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionLadderUp.CompletionLadderTasteGate single carrier alignment`

Theorem 4248.2 (Completion ladder NameCert obligations)**.** *For every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier $L = (D, W, R, E, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, componentwise transport, continuation replay, package provenance, and local naming rows;*
2. *tolerance-window compatibility, requiring each finite stream read in W to be indexed by a dyadic tolerance request in D before regular readback is inspected;*
3. *regular-readback exactness, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *read in R to consume only the displayed D, W route;*
4. *Real-seal order, requiring E to be read only after D, W, R have been displayed;*
5. *completion-boundary order, requiring S and U to consume the same D, W, R, E route through the stored continuation row C;*
6. *transport stability, requiring componentwise* hsame *transport in H to preserve the listed route and no other completion data;*
7. *ledger non-escape, excluding quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, and completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 4299.1. The tuple lists the dyadic tolerance row, stream-window row, regular readback row, Real seal, separated-completion witness, Cauchy-completion unit, transport rows, replay row, provenance, and local name.

The first dependency step reads D and W. Since W is the only stream-window coordinate and D is the only tolerance coordinate, each finite observation entering the regular readback must pass through those two rows.

The regular readback row R is then downstream of D, W, and the Real seal row E is downstream of R. The replay row C records this order before either completion-facing row is inspected.

The separated-completion witness S and Cauchy-completion unit U therefore consume the same displayed D, W, R, E route. Componentwise transport by H preserves those rows, while P and N source and name the same packet.

No remaining coordinate can export quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 4248.3 (Completion ladder separated Real route)**.** *Every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier factors any separated-completion or completion-unit consumer through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{SeparatedCompletion}^{\uparrow},\quad \mathsf{CauchyCompletionUnit}^{\uparrow}.$$

Equivalently, a consumer of S or U must first read the displayed dyadic tolerance row D, stream-window row W, regular readback row R, and Real seal row E, with componentwise transport, continuation replay, provenance, and local naming supplied by H, C, P, N.

Proof. Start from an accepted carrier $L = (D, W, R, E, S, U, H, C, P, N)$. The only completion-facing rows are S and U, so any separated-completion or completion-unit consumer must inspect one of those coordinates.

The definition places both coordinates after D, W, R, E. The continuation row C records the same order, forcing each consumer read to pass from dyadic tolerance and finite window through regular rational readback before the Real seal is available.

Componentwise hsame transport by H may rename the rows, but it preserves the displayed route rather than adding a second stream, a chosen limit, or a host completion object. The package and local name rows P, N then record that the consumer is reading the same finite packet.

Since the carrier contains no row for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the separated Real route above is the whole exported completion-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionLadder}^{\uparrow}$ packet is generated from DyadicRatCore tolerance requests, StreamName finite windows, RegSeqRat readback, a RealUp seal, a SeparatedCompletion witness, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$, seed)

Scope closed: Definition 4299.1, Definition 4299.2, and Definition 4299.3 seed the finite Real and Completion ladder over DyadicRatCore, StreamName, RegSeqRat, Real, SeparatedCompletion, CauchyCompletionUnit, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, bridge checking, Lean verification, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window compatibility, regular readback exactness, Real-seal order, separated-completion routing, completion-unit routing, componentwise transport stability, and ledger non-escape for BEDC.Derived.CompletionLadderUp.

Chapter 4249

A Concrete Naming Certificate for $\mathsf{LocatedFormalBar}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Vickers, the $\mathsf{LocatedFormalBar}^{\uparrow}$ packet records a finite located bar/cover boundary for spread paths and formal-topology cover requests. It sits after the cover/refinement carrier of chapter 3967, the admissible-prefix discipline of chapter 1532, the bar-recursion route of chapter 1675, and the finite observation interface of chapter 126. The packet supplies Real and Completion consumers with a finite bar boundary without importing open-cover compactness, selected points, quotient opens, countable choice, or ambient completeness.

Definition 4249.1 (Located formal-bar carrier)**.** *A* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FormalTopology}^{\uparrow}$ *cover/refinement row, read through chapter 3967, that names the finite basic neighbourhoods and refinement requests used by the bar.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admissible-prefix row, read through chapter 1532, that determines which finite path prefixes can enter the bar ledger.*
3. *B is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *bar-recursion row, read through chapter 1675, that records the finite induction surface consumed by bar-cover arguments.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, read through chapter 126, that exposes the inspected prefix windows for Real and Completion routes.*
5. *Q is the located-query row: each accepted request names a finite prefix, a formal basic neighbourhood, and the bar-cover decision boundary visible at that prefix.*
6. *R is the spread-cover row, recording the finite route from an admissible prefix through the formal cover row to the bar-recursion row.*
7. *H records componentwise* hsame *transport for formal-cover rows, spread-law prefixes, bar-recursion rows, StreamName windows, located-query rows, spread-cover rows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion finite-window request through $F, S, B, W, Q, R, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedFormalBar}^{\uparrow}$, $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite path, a selected point of a cover, quotient-open equality, countable choice, host topological equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedFormalBarUp.LocatedFormalBarTasteGate single carrier alignment`

Theorem 4249.2 (Located formal-bar spread-cover route)**.** *Every accepted* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

routes a Real or Completion finite-window consumer through the same displayed spread-cover boundary:

$$F, S, B, W \longmapsto Q, R \longmapsto H, C, P, N.$$

The formal-topology cover row F, spread-law row S, Brouwer bar-induction row B, and StreamName window row W must be visible before the located-query row Q and spread-cover row R are consumed. The route exports no open-cover compactness theorem, selected infinite branch, selected cover point, quotient-open equality, countable-choice schedule, host real equality, or ambient completeness datum.

Proof. Project the carrier of Definition 4300.1. The source rows are F, S, B, W, and the located bar/cover rows are Q, R.

First read the formal-topology and spread-law surfaces. The finite cover/refinement row F supplies only displayed basic neighbourhoods, while S supplies only admissible finite prefixes.

Next pass through the bar-recursion and StreamName rows. Row B keeps the bar argument on the finite Brouwer induction surface, and W fixes the inspected finite windows used by Real and Completion consumers.

Finally replay through Q, R, H, C, P, N. The located query and spread-cover rows are transported componentwise and replayed with the same provenance and local NameCert row. Since the carrier contains no coordinate for compact open covers, selected branches, selected points, quotient opens, choice, host equality, or ambient completeness, no consumer can read those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedFormalBar}^{\uparrow}$ packet is generated from FormalTopology cover/refinement rows, Spread-Law admissible-prefix rows, BrouwerBarInduction bar-recursion rows, StreamName finite-window rows, located-query rows, spread-cover rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, seed)

Scope closed: Definition 4300.1 and Definition 4300.2 seed the finite located bar/cover boundary for Real and Completion consumers over formal covers, spread prefixes, bar recursion, finite windows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, selected cover points, quotient-open equality, countable choice, host topological equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formal-cover exposure, spread-law prefix exposure, bar-recursion routing, StreamName finite-window replay, located-query exactness, spread-cover exactness, transport, provenance, and local naming for BEDC.Derived.LocatedFormalBarUp.

Chapter 4250

A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$

Following Bishop and Bridges, the CauchyModulusFunctoriality$^{\uparrow}$ packet records the finite data needed to transport displayed Cauchy moduli along a BEDC route without turning that route into a host function equality. It sits between the dyadic tolerance source DyadicRatCore$^{\uparrow}$ of chapter 291, the finite observation route StreamName$^{\uparrow}$ of chapter 126, the regular rational readback RegSeqRat$^{\uparrow}$ of chapter 301, the real-name seal Real$^{\uparrow}$ of chapter 38, the pullback discipline CauchyModulusPullback$^{\uparrow}$ of chapter 1951, and the regular stability surface RegularCauchyModulusStability$^{\uparrow}$ of chapter 1927.

Definition 4250.1 (Cauchy modulus functoriality carrier)**.** *A* CauchyModulusFunctoriality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy-modulus row, naming the inspected source modulus and its finite request boundary.*
2. *T is the target Cauchy-modulus row, naming the target modulus after transport along the displayed route.*
3. *R is the route-map row, recording the finite BEDC route that carries source requests to target requests without asserting equality of host functions.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, read through chapter 291, that keeps each modulus comparison on dyadic rational bounds.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row, read through chapter 126, that records the observation window used by the transported modulus request.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, read through chapter 301, that turns the transported window into a regular rational comparison.*
7. *E is the* Real$^{\uparrow}$ *seal, read through chapter 38, that exposes only the terminal real-name boundary reached by the finite route.*
8. *H records componentwise* hsame *transport for source modulus, target modulus, route map, dyadic tolerance ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a transported modulus request through $S, T, R, D, W, Q, E, H, P, N$.*

10. *P is the* Pkg *provenance over* CauchyModulusFunctoriality$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStal DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function equality, a selected limit, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusFunctorialityUp.CauchyModulusFunctorialityTasteGate single carrier alignment`

Theorem 4250.2 (Cauchy modulus functoriality NameCert obligations)**.** *For every accepted* CauchyModulusFunctoriality$^{\uparrow}$ *carrier*

$$F = (S, T, R, D, W, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *obligations are exactly the finite route obligations listed by the carrier: source-modulus admission, target-modulus admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The route may use the sibling certificates for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, *and* RegularCauchyModulusStability$^{\uparrow}$, *but it exports no host function equality, selected limit, quotient real equality, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 4301.1. The source and target modulus rows are S, T, and the finite transport route is the displayed row R.

The numeric side is then forced through D, W, Q. The dyadic ledger D supplies the tolerance comparisons, the StreamName window W supplies the finite observation boundary, and the RegSeqRat row Q supplies the regular rational readback.

The terminal read is sealed by E. It records only the Real-facing boundary reached by the same finite route, so it cannot introduce a selected limit or quotient-real equality.

Finally transport and replay through H, C, P, N. These rows preserve the same carrier components under hsame, Cont, package provenance, and local naming. Since the carrier contains no coordinates for host function equality, selected limits, quotient real equality, trichotomy, countable choice, or ambient completeness, the listed NameCert obligations exhaust the packet. □

Closure status for NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$.

Constructive story: A CauchyModulusFunctoriality$^{\uparrow}$ packet is generated from source and target Cauchy-modulus rows, a finite route-map row, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, seed)

Scope closed: Definition 4301.1 and Definition 4301.2 seed the finite modulus-transport certificate over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStability$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, selected limits, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFunctorialityUp.

Chapter 4251

A Concrete Naming Certificate for $\mathsf{RealLimitStability}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealLimitStability}^{\uparrow}$ packet records the finite data by which two displayed terminal $\mathsf{Real}^{\uparrow}$ limit seals remain stable under shared regular-Cauchy evidence. It sits after the terminal Real seal of chapter 38, the regular rational readback discipline of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the completion witness of chapter 3579. The packet names stability of displayed limit seals; it is not quotient stream equality, selected host limits, trichotomy, countable choice, or ambient Real completeness.

Definition 4251.1 (Real limit stability carrier)**.** *A* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two limit-presentation rows whose terminal seals are compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both presentations are inspected.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the shared window.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the common dyadic budgets used by the comparison.*
5. E_0 *and* E_1 *are the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after* L_0, L_1, W, R, D *have been displayed.*
6. H *records componentwise* hsame *transport for both limit presentations, the shared window, readback, tolerance ledger, paired Real seals, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a stability request through* $L_0, L_1, W, R, D, E_0, E_1, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealLimitStability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate single carrier alignment`

Theorem 4251.2 (Real limit stability NameCert obligations)**.** *For every accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier* $S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite packet, classifier stability under componentwise* hsame*, shared-window readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic tolerance nonescape through* $\mathsf{DyadicRatCore}^{\uparrow}$*, paired terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4303.1. The accepted packet exposes exactly two presentation rows, one shared window, one regular rational readback, one dyadic tolerance ledger, two terminal Real seals, transport, replay, provenance, and local naming.

Carrier admission and classifier stability are componentwise. The transport row H moves the two presentations, shared evidence, paired seals, and certificate rows together, while the replay row C preserves the same order of inspection for every stability request.

The provenance row P cites only the listed sibling surfaces and the local row N names the packet. Since the carrier contains no quotient equality, selected host limit, trichotomy oracle, choice schedule, ambient completeness coordinate, Lean certificate, or bridge row, the listed obligations exhaust the NameCert surface. □

Theorem 4251.3 (Real limit stability seal stability)**.** *Every comparison of the paired terminal seals in an accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *packet factors through the displayed route*

$$L_0, L_1 \longmapsto W \longmapsto R \longmapsto D \longmapsto E_0, E_1.$$

Thus the two $\mathsf{Real}^{\uparrow}$ *seals may be reused only under the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger displayed by the packet.*

Proof. Start with a consumer that reads the paired Real seals. The only Real-facing terminal coordinates in the carrier are E_0 and E_1, and both appear after the two presentation rows, shared finite window, regular rational readback, and dyadic ledger.

The consumer must therefore pass through W, R, D before either seal can be used. These rows give the finite observations, regular Cauchy readback, and common tolerance budget that make the two terminal seals comparable as displayed BEDC data.

Finally read H, C, P, N. These rows transport, replay, source, and name the same route, so they preserve seal stability without adding a selected host limit or quotient stream equality. □

Theorem 4251.4 (Real limit stability nonescape)**.** *No accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier exports quotient stream equality, a selected host limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a detached Real seal, Lean verification, or bridge checking. Any downstream completion consumer receives only the two limit presentations, shared window, regular readback, dyadic ledger, paired seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains no coordinate for quotient equality, selected host limits, trichotomy, choice, ambient completeness, detached seal data, Lean verification, or a bridge certificate.

The paired seals E_0, E_1 are downstream of the shared witness rows W, R, D. Hence a completion consumer can reuse the seals only through the displayed finite evidence and not through an external selected limit.

Transport, replay, provenance, and local naming preserve the same boundary. The rows H, C, P, N move and name only the finite packet, so every exported object remains inside the displayed Real-limit-stability surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitStability}^{\uparrow}$ packet is generated from two limit-presentation rows, a shared StreamName window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4303.1, Definition 4303.2, Definition 4303.3, and Definition 4303.4 seed the finite stability packet for paired terminal Real limit seals under shared StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, shared-window readback, dyadic ledger nonescape, paired Real-seal stability, transport, replay, provenance, and local naming for BEDC.Derived.RealLimitStabilityUp.

Chapter 4252

A Concrete Naming Certificate for $\mathsf{DecimalIntervalReal}^{\uparrow}$

Bishop–Bridges constructive analysis uses rational and decimal brackets as finite evidence before a real name is treated as complete. The $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet records this decimal-bracket carrier as BEDC NameCert data. It sits after the decimal prefix surface of chapter 1417, the finite decimal stream of chapter 4191, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval source of chapter 1709. The packet gives L10 Real and Completion consumers a finite decimal interval carrier without quotient reals, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4252.1 (Decimal interval real carrier)**.** *A* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower finite decimal-prefix row, read through* $\mathsf{DecimalExpansion}^{\uparrow}$ *and the finite* $\mathsf{DecimalStream}^{\uparrow}$ *schedule.*
2. *B is the upper finite decimal-prefix row, read through the same decimal surfaces and carrying the advertised bracket side.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating decimal place values into dyadic comparison margins.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for endpoint-prefix reads, bracket comparison, and downstream replay.*
5. *Q is the bracket-order row, recording that the lower decimal prefix and upper decimal prefix are compared only through the displayed tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint windows, decimal prefix rows, dyadic ledger, and bracket-order row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback has been displayed.*
8. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *handoff row that consumes the same endpoint, tolerance, readback, and seal rows as a located interval boundary.*

9. *H records componentwise* hsame *transport for endpoint prefixes, tolerance rows, windows, bracket order, regular readback, real seal, located interval handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a decimal-bracket request through endpoint prefixes, dyadic comparison, finite windows, regular readback, real sealing, and located interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DecimalIntervalReal}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for a quotient real, a trichotomy decision, a selected decimal representative, host endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier`
Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealTasteGate single carrier alignment`

Theorem 4252.2 (Decimal interval real NameCert obligations)**.** *For every accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower decimal prefix, upper decimal prefix, dyadic tolerance ledger, stream windows, bracket-order row, regular rational readback, real seal, located interval handoff, transport, replay, provenance, and local naming rows;*
2. *decimal-prefix visibility, requiring endpoint reads to enter through A and B before any bracket comparison is inspected;*
3. *tolerance discipline, requiring the place-value comparison to use the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D and no hidden real-order oracle;*
4. *window discipline, requiring each endpoint and comparison read to be scheduled by the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback order, requiring R to consume A, B, D, W, Q before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *located-interval handoff, requiring L to consume the same endpoint, tolerance, readback, and seal route rather than a host interval;*
7. *componentwise* hsame *transport and displayed* Cont *replay for exactly the rows carried by H and C;*
8. *ledger non-escape, excluding quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4357.1. The projection exposes exactly the two decimal endpoint-prefix rows, the dyadic tolerance ledger, the finite window schedule, the bracket-order row, the regular readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming.

The endpoint reads are visible before comparison. Rows A and B carry the finite decimal prefixes, while D supplies the dyadic margin used to compare their place-value readings. The row W schedules the inspected prefixes and comparison windows, so the bracket-order row Q has no access to a hidden real-order decision.

The real-facing route is downstream of the finite bracket data. Row R is read only after A, B, D, W, Q have been displayed, and row E seals the result only after that regular rational readback.

Finally, the located interval handoff L consumes the same displayed route, while H, C, P, N transport, replay, source, and name the packet. Since no remaining coordinate stores quotient equality, trichotomy, a selected decimal representative, host endpoint equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier semantic name certificate`

Theorem 4252.3 (Decimal interval real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow}.$$

Equivalently, a downstream Real and Completion consumer receives the two decimal endpoint-prefix rows, the dyadic comparison ledger, the finite window row, the regular rational readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming before it can use the bracket.

Proof. Start with an accepted carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$. The only real-facing row is E, and the only located-interval row is L.

The definition places E after the endpoint-prefix rows A, B, the dyadic tolerance ledger D, the window schedule W, the bracket-order row Q, and the regular rational readback R. Thus a $\mathsf{Real}^{\uparrow}$ consumer must first inspect the decimal, dyadic, window, and readback rows.

The located interval handoff is not an alternate route. Row L consumes the same A, B, D, W, Q, R, E path, and componentwise transport through H preserves that path under hsame.

The replay, provenance, and local naming rows only record the displayed carrier. They do not add quotient reals, trichotomy, selected representatives, host endpoint equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed route is the whole exported handoff. □

Falsifiable prediction: Every accepted $\mathsf{DecimalIntervalReal}^{\uparrow}$ read displays two finite decimal-prefix rows, one dyadic comparison ledger, one finite StreamName window row, one RegSeqRat readback row, one RealUp seal row, one LocatedRealIntervalUp handoff row, and local transport, replay, provenance, and naming rows before it is consumed as a decimal bracket.

Independence witness: decimalexpansion, decimal-stream, dyadicratcore, streamname, regseqrat, real, located-real-interval: $\mathsf{DecimalIntervalReal}^{\uparrow}$ is not bijective to any listed sibling because it binds two ordered decimal endpoint-prefix rows, one dyadic comparison ledger, finite observation windows, regular rational readback, RealUp sealing, located interval handoff, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet is generated from lower and upper decimal-prefix rows, DecimalStream windows, a DyadicRatCore comparison ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, a LocatedRealIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4357.1, Definition 4357.2, and Definition 4357.3 seed the finite decimal-bracket route over DecimalExpansionUp, DecimalStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealIntervalUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DecimalIntervalRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix visibility, dyadic tolerance discipline, finite-window discipline, regular-readback order, Real seal handoff, located interval handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.DecimalIntervalRealUp.

Chapter 4253

A Concrete Naming Certificate for $\mathsf{AttouchWetsTopology}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet records local Hausdorff-window convergence for closed metric figures as finite BEDC NameCert data over the metric-completion and L10 real-name surface. It sits after the metric carrier of chapter 57, the Hausdorff-distance surface of chapter 3653, the terminal real seal of chapter 38, the regular rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the finite observation topology; it does not import host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4253.1 (Attouch-Wets topology carrier)**.** *An* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose metric, point windows, and bounded-window radii are the only ambient metric data available to the packet.*
2. *H is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *local-window row, read only on displayed bounded metric windows rather than on an ambient hyperspace.*
3. *D is the dyadic tolerance ledger for window radius, Hausdorff comparison margin, and late-window precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, bounded windows, comparison stages, and requested precision cells.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Hausdorff-window comparisons obtained from M, H, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *Q records componentwise* hsame *transport for metric source rows, Hausdorff-window rows, dyadic ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming.*
8. *T records displayed* Cont *replay from an Attouch–Wets observation request through M, H, D, W, R, E.*

9. *C is the local bounded-window classifier row comparing packets only through the displayed metric source, Hausdorff-window, tolerance, stream, readback, seal, transport, replay, provenance, and naming coordinates.*

10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *row.*

The classifier has no coordinate for global closed-set topology, open-cover compactness, selected nearest points, quotient hyperspace equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AttouchWetsTopologyUp.AttouchWetsTopologyTasteGate single carrier alignment`

Theorem 4253.2 (Attouch-Wets topology NameCert obligation surface)**.** *For every accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *expose the* $\mathsf{Met}^{\uparrow}$ *source row M before any local Hausdorff comparison is read;*

2. *display the* $\mathsf{HausdorffDistance}^{\uparrow}$ *window row H only on bounded metric windows named by the packet;*

3. *bind all numerical precision to the dyadic tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

4. *preserve the same finite route by componentwise* hsame *transport Q and displayed* Cont *replay T;*

5. *source and name the same packet through the bounded-window classifier C,* Pkg *provenance P, and the local* NameCert *row N.*

No obligation row is stored for a host hyperspace topology, arbitrary closed-set compactness, selected representatives, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4358.1. The metric row M is the only source of ambient distance data, and the Hausdorff row H is read only after the bounded window has been displayed.

The numerical part is downstream of that window. The tolerance ledger D, finite window row W, regular readback R, and real seal E are the only precision and value rows available to an observation request.

Transport and replay keep the same route. Rows Q and T preserve the displayed metric, Hausdorff, tolerance, window, readback, and seal data, while C, P, N classify, source, and name the finite packet.

Since the carrier contains none of the excluded topology, compactness, selection, quotient, choice, or ambient-completeness coordinates, the displayed obligations exhaust the local NameCert surface. □

Theorem 4253.3 (Attouch-Wets local Hausdorff window)**.** *Every topology-facing read exported by an accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{HausdorffDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}.$$

Thus a downstream metric-completion consumer receives the displayed metric source, bounded local Hausdorff window, finite observation schedule, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming before it can use Attouch–Wets convergence data.

Proof. Start with a topology-facing read of the accepted carrier. The only available metric source is M, and the only Hausdorff comparison row is the local-window row H.

The observation is then forced through D, W, R, E. The dyadic ledger fixes the requested tolerance, the stream row displays the inspected bounded windows, the regular rational row records the readback, and the real row seals the reported Hausdorff-window value.

Finally Q, T, C, P, N transport, replay, classify, source, and name that same finite route. Because no row supplies host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported topology-facing read factors through the displayed local Hausdorff window. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, a HausdorffDistance bounded-window row, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, a bounded-window classifier row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4358.1, Definition 4358.2, and Definition 4358.3 seed the local Hausdorff-window topology route over MetricSpaceUp, HausdorffDistanceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AttouchWetsTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Hausdorff-window exposure, finite-window scheduling, regular readback, Real sealing, transport, replay, classifier exactness, provenance, and local naming for BEDC.Derived.AttouchWetsTopologyUp.

Chapter 4254

A Concrete Naming Certificate for $\mathsf{CantorIntersectionMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet records the metric nested-closed-set intersection route as finite BEDC NameCert data over the L10 completion surface. It sits after the complete-metric source of chapter 128, the metric-completion route of chapter 1506, the nested closed-ball packet of chapter 3526, the Cantor complete metric route of chapter 2199, the Cauchy-filter interface of chapter 333, and the L10 real-name route through chapter 291, chapter 126, chapter 301, and chapter 38. The metric packet is the metric-completion specialization of the finite Cantor-intersection route in chapter 1421 and the theorem-facing intersection route in chapter 2318; it consumes their nested-window and diagonal-readback rows before exposing complete-metric service. The packet names a nested metric intersection handoff; it does not assert open-cover compactness, arbitrary closed-set compactness, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4254.1 (CantorIntersectionMetric carrier)**.** *A* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence service are the only complete-metric data available to the packet.*
2. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row naming the surrounding completion request before any intersection point is read.*
3. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row, recording the descending closed metric ball spine and its finite containment witnesses.*
4. *K is the* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *row, used only as the displayed nested closed-ball completion handoff.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested metric spine and read only through the displayed filter route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, storing the shrink schedule, late-radius comparisons, and requested precision bounds.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, radii, and containment data.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of the displayed center schedule.*
9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, K, F, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for the complete-metric source, completion route, nested-ball row, Cantor handoff, filter, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a nested metric intersection request through $M, E, B, K, F, D, S, R, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, B, K, F, D, S, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary closed-set compactness, quotient completion equality, a selected limit, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionMetricUp.CantorIntersectionMetricTasteGate single carrier alignment`

Theorem 4254.2 (CantorIntersectionMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the complete-metric and metric-completion source rows M, E before any intersection witness is read;*
2. *display the nested metric route through B, K, F, namely the nested closed-ball spine, Cantor complete-metric handoff, and generated Cauchy-filter row;*
3. *bind all numerical readback to the L10 rows D, S, R, L, namely the dyadic tolerance ledger, finite stream windows, regular rational handoff, and terminal real seal;*
4. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
5. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row is stored for open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, trichotomy, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4359.1. The rows M, E expose the metric and completion sources before the packet can read an intersection point.

The nested metric content is then displayed through B, K, F. The nested closed-ball row supplies the descending closed metric spine, the Cantor complete-metric row supplies the existing handoff, and the Cauchy-filter row records the generated filter route.

The L10 readback is downstream. The dyadic ledger D, stream windows S, regular rational row R, and real seal L are the only numerical coordinates available before a completion-facing consumer receives a named output.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains none of the excluded compactness, quotient, selected-limit, trichotomy, or ambient-completeness coordinates, the five displayed obligations exhaust the local NameCert surface. □

Theorem 4254.3 (CantorIntersectionMetric complete handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier factors through the displayed nested metric route*

$\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, S

Thus a consumer receives the complete-metric service, completion request, nested closed-ball spine, Cantor complete-metric handoff, Cauchy-filter row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows before it may use the intersection output. The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness principle, open-cover compactness theorem, arbitrary closed-set compactness, quotient stream equality, or selected limit.*

Proof. Begin with a completion-facing read of the accepted carrier. The first available rows are M and E, so the read is grounded in the displayed complete-metric service and the surrounding metric-completion request.

Next pass through the nested route B, K, F. The descending closed-ball spine is read through the $\mathsf{NestedClosedBall}^{\uparrow}$ row, the existing Cantor complete-metric handoff is read through K, and the generated filter is read through F.

The numerical output is then forced through the L10 readback rows D, S, R, L. The dyadic ledger supplies tolerance control, the stream row supplies finite windows, the regular rational row supplies readback, and the terminal real seal is reached only after those rows are present.

The structural rows H, C, P, N preserve the same route for downstream consumers. Because no row supplies full real completeness, open-cover compactness, arbitrary closed-set compactness, quotient stream equality, or a selected limit, every completion-facing read factors through the displayed nested metric route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, a MetricCompletion request row, a NestedClosedBall spine, a CantorCompleteMetric handoff, a CauchyFilter row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4359.1, Definition 4359.2, and Definition 4359.3 seed the metric nested-closed-set intersection route over displayed completion, nested-ball, Cantor handoff, filter, dyadic, stream, regular readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested metric inclusion, Cantor handoff, Cauchy-filter generation, dyadic shrink exactness, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionMetricUp.

Chapter 4255

A Concrete Naming Certificate for $\mathsf{TightRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{TightRealInterval}^{\uparrow}$ packet names a finite lower-upper real interval by displaying both endpoint names and every rational cell through which those endpoints are inspected. It sits downstream of the located endpoint surface of chapter 394 and the dyadic refinement surface of chapter 306: consumers may read a tight interval only after the lower endpoint, upper endpoint, rational cell ledger, dyadic refinement ledger, stream windows, regular rational readbacks, and Real seal rows have all been displayed.

Definition 4255.1 (TightRealInterval carrier)**.** *A* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, U, Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row and U is the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger recording the finite rational windows that enclose the endpoint reads and witness lower-before-upper compatibility.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to the rational cells in Q.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the endpoint observations and interval cells inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and cell refinements.*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached from L, U, S, R, Q, D.*
7. *H records componentwise* hsame *transport for endpoints, rational cells, dyadic refinements, stream windows, readbacks, Real seals, replay, provenance, and the local name.*
8. *C records displayed* Cont *replay through $L, U, Q, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host interval equality, chosen endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.TightRealIntervalUp.TightRealIntervalUp`
Lean4 ✓ `BEDC.Derived.TightRealIntervalUp.TasteGate.TightRealIntervalTasteGate single carrier alignment`

Theorem 4255.2 (TightRealInterval NameCert obligations)**.** *For an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier* $T = (L, U, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, rational and dyadic interval ledgers, finite stream windows, regular rational readbacks, paired Real seal rows, componentwise transport, continuation replay, package provenance, and local naming; endpoint-order visibility through* Q*; dyadic refinement discipline through* D*; stream-readback exactness through* S, R*; and ledger non-escape from host intervals, selected endpoints, quotient real equality, choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 4360.1. The lower and upper endpoint rows, the rational cell ledger, the dyadic refinement ledger, finite windows, regular readbacks, Real seals, transport rows, replay rows, provenance row, and local naming row are all visible before any interval consumer can read the packet.

Endpoint-order visibility is supplied by Q. Since each dyadic refinement in D is attached to a rational cell, the refinement discipline cannot detach from the endpoint ledger.

Stream-readback exactness is then read through S and R. The rows H, C, P, N transport, replay, source, and name the same finite route, while the excluded interval and completeness data have no carrier coordinate. □

Theorem 4255.3 (TightRealInterval Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

The read exposes lower and upper endpoint windows together with the rational and dyadic cell ledgers that bind them. It does not expose a host real interval, a selected endpoint value, quotient real equality, countable choice, trichotomy, or ambient real completeness.

Proof. Begin with the accepted carrier. The only seal-facing row is E, and it is displayed after L, U, Q, D, S, R in the finite packet.

Apply Definition 4360.2. Endpoint-order visibility forces the Real read to keep the rational cell ledger Q, dyadic refinement discipline forces any dyadic window through D, and stream-readback exactness forces finite observations through S, R.

Finally route through H, C, P, N. These rows preserve and name the same packet, so no host interval, endpoint selector, quotient equality, choice schedule, trichotomy row, or ambient completeness principle can enter the Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TightRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, paired $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4360.1, Definition 4360.2, and Definition 4360.3 seed the finite lower-upper Real interval packet over located endpoints, rational cells, dyadic refinements, stream windows, regular readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Scoped closure requires a checked carrier-obligation package and a public consumer route for the same finite lower-upper interval packet.

Chapter 4256

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$

Following Bishop-style compact metric uniform continuity, the $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet records the finite handoff that transfers compact finite-net and pointwise continuity data into a uniform modulus row. It consumes the finite compact source of chapter 60, the compact metric net surface of chapter 129, the continuity interface of chapter 391, the finite stream and regular readback rows of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite modulus-transfer NameCert surface, not an open-cover compactness proof or a host modulus selector.

Definition 4256.1 (Compact uniform modulus transfer carrier)**.** *A* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, U, S, R, D, F, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row.*
2. *G is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric coverage row for the same finite centers and source distance witnesses.*
3. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise-to-uniform continuity row consumed by the transfer.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected finite centers, tolerance requests, and pointwise-radius reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for finite radius comparison and shrinkage.*
7. *F is the modulus-transfer ledger that records the finite lower-bound handoff into the downstream uniform-modulus consumer row.*
8. *H records componentwise* hsame *transport for compact, compact-metric, uniform-continuity, stream, readback, dyadic, transfer, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay through $K, G, U, S, R, D, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate`
`single carrier alignment`

Theorem 4256.2 (Compact uniform modulus transfer obligations)**.** *For an accepted* CompactUniformModulusTrans *carrier $M = (K, G, U, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net row, compact-metric coverage row, uniform-continuity row, stream windows, regular rational readbacks, dyadic tolerance ledger, modulus-transfer ledger, componentwise transport, continuation replay, package provenance, and local naming; source lock between K, G, U; finite-window exactness through S, R; dyadic tolerance exactness through D; and non-escape from open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient equality, choice, and ambient completeness.*

Proof. Project the carrier of Definition 4361.1. The compact row, compact-metric row, continuity row, stream windows, readbacks, dyadic tolerance ledger, transfer ledger, transport rows, replay rows, provenance row, and local name are displayed as one finite packet.

The source lock is the shared read of K, G, U. A transfer consumer cannot use the compact finite-net row without the compact-metric distance surface and the continuity row that supplies pointwise radius data.

Finite-window and dyadic exactness are read through S, R, D. The ledger F can therefore transfer only the displayed finite lower-bound data, and H, C, P, N preserve that route without adding any excluded compactness or choice principle. □

Theorem 4256.3 (Compact uniform modulus transfer handoff)**.** *Every accepted* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *packet hands a downstream uniform-modulus consumer only the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont}$$

The handoff records compact finite-net coverage, pointwise continuity radius reads, finite stream windows, regular rational readbacks, dyadic tolerance comparison, and the transfer ledger. It exports no open-cover compactness theorem, selected net center, selected Lebesgue number, host modulus function, quotient equality, countable choice, or ambient completeness.

Proof. Start from the transfer obligations of Definition 4361.2. They identify the compact, compact-metric, continuity, stream, readback, dyadic, and transfer rows as the only NameCert-readable data.

Route the downstream consumer through F. The ledger can be consumed only after K, G, U, S, R, D have supplied the finite-net, pointwise-radius, window, readback, and tolerance rows.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since the excluded compactness, selection, equality, choice, and completeness data have no coordinate in the carrier, the downstream handoff is exactly the displayed finite modulus-transfer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet is generated from compact finite-net rows, compact-metric coverage rows, uniform-continuity rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a modulus-transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, seed)

Scope closed: Definition 4361.1, Definition 4361.2, and Definition 4361.3 seed the finite compact-net modulus-transfer packet for uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Upgrade path: Scoped closure requires a checked transfer-obligation package and a public uniform-modulus consumer route for the same finite packet.

Chapter 4257

A Concrete Naming Certificate for $\mathsf{RealIntervalApartness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntervalApartness}^{\uparrow}$ packet records located separation of two real intervals by a displayed positive dyadic gap rather than by a trichotomy decision on completed real endpoints. It sits after the terminal real-name seal of chapter 38, the located endpoint interval surface of chapter 1709, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the separability boundary of chapter 1809. The packet names interval separation only through visible endpoint rows, gap rows, transport, replay, provenance, and local naming.

Definition 4257.1 (RealIntervalApartness carrier)**.** *A* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are the two* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint packets whose lower and upper endpoint rows are exposed before any apartness read.*
2. *L, U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *seal rows for I, and L', U' are the corresponding* $\mathsf{Real}^{\uparrow}$ *seal rows for J.*
3. *G is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness, attached either to the displayed comparison $U + G \leq L'$ or to the displayed comparison $U' + G \leq L$.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the endpoint observations and gap inspections used by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and dyadic gap checks.*
6. *E records the* $\mathsf{RealSeparability}^{\uparrow}$ *boundary row witnessing that the two exposed endpoint intervals are separated by the displayed dyadic gap.*
7. *H records componentwise* hsame *transport for intervals, endpoint seals, dyadic gap rows, finite windows, readbacks, separability rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay through $I, J, L, U, L', U', G, S, R, E, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient real equality, a selected endpoint representative, a host interval order, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 4257.2 (RealIntervalApartness NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *admit the two located real interval packets I, J and expose their lower and upper endpoint rows through L, U, L', U';*
2. *expose the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness G and the side on which the gap separates the displayed endpoint rows;*
3. *bind every endpoint and gap inspection to the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, R;*
4. *preserve the interval, endpoint, gap, readback, and separability rows by componentwise* hsame *transport through H;*
5. *replay the same finite route through* Cont *row C, source it by* Pkg *provenance P, and name it by the local* NameCert *row N.*

No obligation row is stored for real trichotomy, quotient equality of completed real names, selected endpoint representatives, host interval order, or ambient real completeness.

Proof. Project the carrier of Definition 4362.1. The two located interval packets and their endpoint seals are the only interval data exposed to a consumer.

The separation datum is the row G. Positivity is read as a dyadic-row property, and the side of separation is part of the displayed ledger; no trichotomy comparison is consulted.

The numerical readback is forced through S and R, then through the separability row E. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since none of the excluded host-order, quotient-equality, representative-selection, trichotomy, or completeness data occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealIntervalApartnessUp.TasteGate.RealIntervalApartnessTasteGate single carrier alignment`

Theorem 4257.3 (RealIntervalApartness separability route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, hsame, Cont, Pkg, Na

The route exports located interval separation only as a positive dyadic gap between displayed endpoint rows. It exports no real trichotomy, quotient real equality, host interval order, selected endpoint representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the NameCert obligations of Definition 4362.2. They force the two located interval packets and endpoint seals to be visible before a consumer may read an apartness claim.

Next read the dyadic gap row and its finite endpoint inspections through the stream and regular-rational rows. The separability row is therefore attached to the displayed positive gap, not to a host comparison of completed reals.

Finally route through transport, replay, provenance, and local naming. These rows preserve the same finite packet, so the consumer receives located interval apartness without importing any excluded trichotomy, quotient, selection, host-order, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalApartness}^{\uparrow}$ packet is generated from two LocatedRealInterval endpoint packets, Real endpoint seal rows, a positive DyadicRatCore gap witness, finite StreamName windows, RegSeqRat readback, a RealSeparability boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4362.1, Definition 4362.2, and Definition 4362.3 seed the located interval separation packet over displayed endpoint, dyadic gap, finite-window, regular-readback, Real seal, separability, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient real equality, selected endpoint representatives, host interval order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, positive-gap exposure, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.RealIntervalApartnessUp.

Chapter 4258

A Concrete Naming Certificate for $\mathsf{PartitionBasinPolynomial}^{\uparrow}$

Following finite enumerative combinatorics, the $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ packet records the partition-basin product formula of Definition 9462.1736 as displayed NameCert data over block sizes, legal decorations, defect exponents, incidence totals, transport, replay, provenance, and local naming. It sits after the partition carrier of chapter 201, the defect-flow partition skeleton of chapter 4236, and the partition-core incidence certificate of chapter 4214. The packet names a finite polynomial enumeration surface; it is not a biological stochastic model, a host generating-function oracle, or a quotient-level semantic identification.

Definition 4258.1 (PartitionBasinPolynomial carrier)**.** *A* $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite block-count row for the partition core, and S is the finite block-size ledger $(s_1, \ldots, s_m)$.*
2. *D is the legal-decoration row, recording for each coordinate of block i the admissible subset of the other $m-1$ rows and the private-coordinate exclusion.*
3. *E is the defect-exponent ledger that assigns exponent weight q^r to a coordinate receiving r additional rows.*
4. *F is the product-formula row*
$$\prod_{i=1}^{m} \left[(1+q)^{(m-1)s_i} - \left((1+q)^{m-1} - 1 \right)^{s_i} \right].$$
5. *Q is the four-coordinate specialization row for the five partition types 4, $3+1$, $2+2$, $2+1+1$, and $1+1+1+1$.*
6. *I is the incidence-total row that reads the quotient-level and codon-level totals from the displayed specialization.*
7. *T records the downstream defect-flow handoff to the partition skeleton and incidence certificate.*
8. *H records componentwise* hsame *transport for block sizes, legal decorations, defect exponents, product rows, specialization rows, incidence rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay through $M, S, D, E, F, Q, I, T, H$.*
10. *P is the* Pkg *provenance over* PartitionBasinPolynomial$^{\uparrow}$, Partition$^{\uparrow}$, DefectFlowPartitionSkeleton$^{\uparrow}$, PartitionCoreIncidenceCertificate$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a stochastic biology model, a host polynomial oracle, a quotient-semantic equality, an unbounded partition enumeration, or a hidden recurrence outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.PartitionBasinPolynomialUp.PartitionBasinPolynomialTasteGate single carrier alignment`

Theorem 4258.2 (PartitionBasinPolynomial NameCert obligations)**.** *For every accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *surface consists exactly of block-count and block-size admission through M, S, legal-decoration admission through D, defect-exponent admission through E, product-formula exposure through F, four-coordinate specialization through Q, incidence-total readback through I, defect-flow handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4363.1. The block-count and block-size rows supply the finite partition core on which the enumeration is read.

The decoration and defect rows are then exposed coordinatewise. For block i, the all-decoration contribution is $(1+q)^{(m-1)s_i}$, and the excluded no-private-coordinate contribution is $\left((1+q)^{m-1}-1\right)^{s_i}$.

Multiplying the legal block rows gives the product row F. The specialization row Q evaluates the same finite formula on the five four-coordinate partition types, and the incidence row I sums the displayed rows with their finite multiplicities.

Finally T, H, C, P, N route the same polynomial surface to the defect-flow and incidence packets. Because no excluded stochastic, host-oracle, quotient-semantic, unbounded-enumeration, or hidden-recurrence coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4258.3 (PartitionBasinPolynomial product specialization)**.** *Every defect-flow consumer of an accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier receives the product formula row for $\mathcal{B}_P(q)$ and the four-coordinate specialization row before reading quotient-level or codon-level incidence totals. The consumer sees only finite block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and local NameCert rows.*

Proof. Start from the obligations of Definition 4363.2. They force the block-size ledger and legal-decoration rows to be displayed before the product formula can be consumed.

The four-coordinate row is a finite specialization of the same product row. Its incidence totals are read only after the partition types and finite multiplicities have been displayed.

The handoff through T, H, C, P, N preserves the same finite surface for the defect-flow skeleton and partition-core incidence certificate. Thus no stochastic model, host generating-function oracle, quotient-semantic equality, or hidden enumeration can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$.

Constructive story: A PartitionBasinPolynomial$^{\uparrow}$ packet is generated from block-count rows, block-size ledgers, legal-decoration rows, defect-exponent ledgers, a product-formula row, a four-coordinate specialization row, incidence-total rows, a defect-flow handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 4363.1, Definition 4363.2, and Definition 4363.3 seed the finite partition-basin polynomial packet over displayed block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a stochastic biology model, host generating-function oracle, quotient-level semantic equality, unbounded partition enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-formula exposure, four-coordinate specialization, incidence readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionBasinPolynomialUp.

Chapter 4259

A Concrete Naming Certificate for $\mathsf{DistanceToSet}^{\uparrow}$

Following Bishop and Bridges on located subsets in metric spaces, the $\mathsf{DistanceToSet}^{\uparrow}$ packet records the finite route by which a point-to-set distance request is read through displayed located-radius rows and closed-ball witnesses. It sits after the metric source of chapter 57, the located-set surface of chapter 1541, the metric closed-ball boundary of chapter 1451, the located closed-ball handoff of chapter 4063, and the terminal real-name seal of chapter 38. The packet names distance-to-set as finite BEDC data; it is not a host infimum, not a selected nearest point, not quotient real equality, and not a countable-choice construction.

Definition 4259.1 (Distance-to-set carrier)**.** *A* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, x, r, B, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the subset by finite located-radius observations.*
3. *x is the displayed point row in the metric source.*
4. *r is the dyadic radius ledger recording the inspected tolerance windows.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness row attached to the point and radius ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance name, read only after the located-set and closed-ball rows are displayed.*
7. *H records componentwise* hsame *transport for the metric source, located set, point row, dyadic ledger, closed-ball witness, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the metric source through located-set exposure, point inspection, dyadic-radius accounting, closed-ball handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, x, r, B, R, H, C, P, N$. It has no coordinate for a host infimum over all points of A, a selected nearest point, quotient real equality, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DistanceToSetUp.DistanceToSetTasteGate single carrier alignment`

Theorem 4259.2 (Distance-to-set NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier*

$$D = (X, A, x, r, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located set, point row, dyadic radius ledger, closed-ball witness, Real seal, transport, replay, provenance, and local naming;*
2. *located-set exposure, requiring every subset read to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *row A before a distance row is exported;*
3. *dyadic-radius monotonicity, requiring the tolerance ledger r to preserve the displayed radius order used by the located-set and closed-ball rows;*
4. *closed-ball handoff, requiring each ball membership or exclusion read to pass through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness B and the located closed-ball sibling surface before the Real seal is read;*
5. *Real seal non-escape, requiring R to seal only the displayed distance name and not a host infimum or quotient real;*
6. *componentwise transport, displayed replay, package provenance, and local* NameCert *naming for exactly the displayed packet.*

No DistanceToSet obligation is stored outside these rows.

Proof. Project the carrier of Definition 4364.1. The metric source is X, the located subset is A, the inspected point is x, and the finite tolerance information is held by r.

The closed-ball witness B is read only after those rows are displayed. Its radius requests are therefore attached to the same metric source, located-set row, point row, and dyadic ledger.

The Real seal R comes after the located-radius and closed-ball rows. Transport H and replay C preserve the same order, while P and N record provenance and local naming.

Since no coordinate stores a host infimum, a nearest-point selector, quotient real equality, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 4259.3 (Distance-to-set located-radius handoff)**.** *Every consumer of an accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *packet reads distance-to-set data only through the displayed located-radius row A, the dyadic radius ledger r, the closed-ball witness B, and the Real seal R, transported by H and replayed by C. The consumer cannot bypass these rows by reading a host infimum, a selected nearest point, quotient real equality, countable choice, or an ambient completeness principle.*

Proof. Read the carrier of Definition 4364.1. The only subset coordinate is the located-set row A, and the only radius coordinate is the dyadic ledger r.

The closed-ball witness B supplies the displayed ball boundary before the Real seal R is exposed. Thus a distance consumer reaches the exported real name only through the located-radius and closed-ball rows.

Apply Definition 4364.2. The transport, replay, provenance, and local naming rows preserve the same finite packet and add no host infimum, nearest-point selector, quotient real equality, countable choice, or ambient completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}$.

Constructive story: A $\mathsf{DistanceToSet}^{\uparrow}$ packet is generated from a metric source, located set row, point row, dyadic radius ledger, closed-ball witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4364.1, Definition 4364.2, and Definition 4364.3 seed the finite located-radius and closed-ball interface for distance-to-set consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host infimum construction, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-set exposure, dyadic-radius monotonicity, closed-ball handoff, Real seal non-escape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DistanceToSetUp.

Chapter 4260

A Concrete Naming Certificate for $\mathsf{JordanCurve}^{\uparrow}$

Following Jordan and constructive topology, the $\mathsf{JordanCurve}^{\uparrow}$ packet records the finite polygonal and winding-ledger route by which a displayed simple closed curve in the real plane may be consumed as a separation boundary. It sits after the terminal real-name seal of chapter 38, the circle carrier of chapter 58, the graph and modulus rows of chapter 130, the finite compact-metric source of chapter 129, and the simplex inventory of chapter 238. The packet names a constructive finite obligation surface; it does not import classical open-cover compactness, selected complementary components, quotient path equality, or a choice-dependent separation theorem.

Definition 4260.1 (Jordan curve carrier)**.** *A* $\mathsf{JordanCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, F, P, W, O, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, recording the circle parameter ledger and its local classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row into the displayed* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ *plane, with graph and modulus observations visible.*
3. *P is the finite polygonal approximation row, read through a* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision and its vertex-image ledger.*
4. *W is the winding ledger, recording the finite edge-crossing and sign-change data used to compare test points with the polygonal boundary.*
5. *O is the orientation row, recording the compatible cyclic order inherited from the* $\mathsf{S}^{1\uparrow}$ *row and the polygonal subdivision.*
6. *I is the injectivity-apartness row, stating that distinct inspected arcs carry displayed positive separation windows before they are used by the boundary read.*
7. *E is the exterior-interior ledger, recording only the finite side classification certified by the current polygonal and winding rows.*
8. *H records componentwise* hsame *transport for $S, F, P, W, O, I, E, C, Q, N$.*
9. *C records displayed* Cont *replay from circle, map, polygonal, winding, orientation, and apartness rows to the side-classification row.*

10. *Q is the* Pkg *provenance over* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, P, W, O, I, E, H, C, Q, N$. It has no coordinate for a host Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or countable-choice separation.

Lean4 ✓ `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate single carrier alignment`

Theorem 4260.2 (Jordan curve orientation ledger)**.** *For every accepted* $\mathsf{JordanCurve}^{\uparrow}$ *carrier*

$$J = (S, F, P, W, O, I, E, H, C, Q, N),$$

the orientation row O is determined by the finite $\mathsf{S}^{1\uparrow}$ *cyclic order, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision in P, and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *edge-image ledger in F. The winding ledger W can read orientation only through those rows, transported by H and replayed by C.*

Proof. Project the accepted carrier. The only circle-facing coordinate is S, and the only finite boundary skeleton is the polygonal row P together with the displayed edge images in F.

The orientation row O is therefore read by following the cyclic order of S across the subdivision cells of P. Each edge-image comparison is a finite real-plane observation supplied by F, not by an ambient topological orientation principle.

The winding ledger W consumes that ordered edge list. Componentwise transport H and replay C preserve the same ordered rows, while Q and N record provenance and local naming. Since the carrier contains no host orientation object or quotient path coordinate, the finite ledger is the entire orientation source. □

Theorem 4260.3 (Jordan curve winding separation boundary)**.** *Every side-classification read from an accepted* $\mathsf{JordanCurve}^{\uparrow}$ *packet factors through the finite route*

$$S, F, P, O, I, W \longmapsto E \longmapsto H, C, Q, N.$$

Thus the packet exposes polygonal approximation, injectivity-apartness, orientation, and winding ledgers before it names an interior or exterior row. It does not assert a blanket classical Jordan separation theorem, selected complementary components, open-cover compactness, quotient path equality, or choice-dependent planar separation.

Proof. Read the carrier of Definition 4365.1. The curve source is S, the displayed map is F, and the finite polygonal boundary is P.

Before a side row is exported, the packet requires the orientation ledger O, injectivity-apartness row I, and winding ledger W. These rows are finite BHist data: the winding read uses edge crossings and signs already attached to the polygonal boundary.

The exterior-interior row E is downstream of that route. Transport, replay, provenance, and naming through H, C, Q, N preserve the same finite packet. The excluded host theorem, selected components, open covers, quotient paths, and choice rows are not coordinates of the carrier, so they cannot be read from the local NameCert surface. □

Falsifiable prediction: A $\mathsf{JordanCurve}^{\uparrow}$ consumer that exports side separation without reading P, W, O, I refutes the carrier surface.

Independence witness: real, sone, continuousmap, compactmetric, simplicialcomplex: the sibling chapters supply real-plane coordinates, circle source data, continuous graph and modulus rows, compact-metric finite inspection, and subdivision cells separately, while $\mathsf{JordanCurve}^{\uparrow}$ binds them into the finite packet $J = (S, F, P, W, O, I, E, H, C, Q, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$.

Constructive story: A $\mathsf{JordanCurve}^{\uparrow}$ packet is generated from a circle source row, continuous real-plane map row, finite polygonal subdivision, winding ledger, orientation row, injectivity-apartness row, side-classification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4365.1, Definition 4365.2, and Definition 4365.3 seed the finite polygonal, winding, orientation, apartness, side-classification, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, the classical Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or choice-dependent separation.

Upgrade path: Obligation closure requires checked carrier admission, polygonal approximation exactness, winding-ledger stability, orientation transport, injectivity-apartness handoff, side-classification non-escape, replay, provenance, and local naming for BEDC.Derived.JordanCurveUp.

Chapter 4261

A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$

The MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite NameCert data. It sits beside the candidate-set packet of chapter 2320 and the normalization router of chapter 1108: those siblings expose candidate evidence and route selection, while this packet records the boundary where empty-context closed typed terms may or may not enter the candidate interface. The packet names a closed-projection frontier only; it does not turn candidate-mediated normalization into an unconditional strong-normalization theorem for every closed MetaCIC term.

Definition 4261.1 (MetaCIC candidate closed-projection boundary carrier)**.** *A* MetaCICCandidateClosedProjection *carrier is a finite* BHist *packet*

$$B = (T, K, R, E, NF, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row whose projection into the candidate interface is being tested.*
2. *K is the candidate-set evidence row, read through* CandidateSetNormalization$^{\uparrow}$ *when the projection is admitted.*
3. *R is the router row, read through* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *and recording which candidate, closed-candidate, or candidate-set route is exposed.*
4. *E is the bounded-normal endpoint seal row, recording the finite normal-output evidence available after candidate admission.*
5. *NF is the no-infinite-reduction row exported only downstream of K, R, E.*
6. *O is the obstruction row recording refusal when the closed term cannot be placed into the displayed candidate interface.*
7. *H records componentwise* hsame *transport for $T, K, R, E, NF, O, C, P, L$.*
8. *C records displayed* Cont *replay from the closed-term row through candidate admission, router selection, endpoint seal, no-infinite-reduction read, and obstruction retention.*

9. *P is the* Pkg *provenance over* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$, CandidateSetNormalization$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, MetaCICOpenProblemLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, R, E, NF, O, H, C, P, L$. It has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Lean4 ✓ `BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.MetaCICCandidateClosedProjectionBoundar`
`single carrier alignment`

Theorem 4261.2 (MetaCIC candidate-set closed projection frontier)**.** *For every accepted* MetaCICCandidateClosed *carrier*

$$B = (T, K, R, E, NF, O, H, C, P, L),$$

the closed-term row T reaches a strong-normalization-facing read only through candidate evidence K, router selection R, and bounded-normal endpoint seal E. If those rows do not place T inside the candidate interface, the obstruction row O is the only exported boundary datum.

Proof. Project the carrier. The closed-term row T is present as a finite syntactic and typing packet, but it is not itself a normalization theorem.

The positive route first reads K through chapter 2320. The router row R then records the candidate, closed-candidate, or candidate-set branch through chapter 1108. Only after those rows are displayed can the endpoint seal E and no-infinite-reduction row NF be consumed.

If candidate admission is absent, O records the refusal. Transport, replay, provenance, and local naming through H, C, P, L preserve that same finite boundary, so the packet exports no evidence-free closed-term strong-normalization claim. □

Theorem 4261.3 (MetaCIC candidate-set SN handoff boundary)**.** *Every strong-normalization handoff exported by an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet is a handoff for rows already inside the displayed candidate interface. It cannot be read as a blanket theorem that all empty-context typed closed MetaCIC terms are strongly normalizing.*

Proof. Start with the frontier theorem. It places the normalization-facing route behind K, R, E and keeps the obstruction row O visible when the candidate interface is not supplied.

For an admitted row, the handoff consumes candidate evidence from CandidateSetNormalization$^{\uparrow}$ and route evidence from MetaCICNormalizationCertificateRouter$^{\uparrow}$. The no-infinite-reduction row NF is therefore downstream of the admitted candidate branch rather than a second source of normalization evidence.

For a closed term outside that interface, the carrier has only T and the obstruction row O on the boundary. The structural rows H, C, P, L transport, replay, source, and name the boundary, but they do not add subject reduction, host conversion equality, quotient reduction equality, or a global normalizer. Hence the handoff is candidate-scoped. □

Theorem 4261.4 (MetaCIC candidate-set public boundary)**.** *Any public no-infinite-reduction or strong-normalization read from an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet must cite candidate-set membership or soundness, router selection, and bounded-normal endpoint rows. A bare empty-context typed closed-term row T is insufficient: the public boundary may expose NF only downstream of the candidate-set facts named by* closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, *and* canonicalCandidateSet_sound *together with the router row R and endpoint seal E.*

Proof. Start with Definition 4366.2. It places every normalization-facing read behind the candidate evidence row K, router row R, and endpoint seal E, and leaves the obstruction row O as the only datum when candidate admission is absent.

Next apply Definition 4366.3. That theorem identifies the handoff as a candidate-interface handoff rather than a theorem about all empty-context closed typed rows. Thus the public NF row cannot be projected from T alone.

Finally read the MetaCIC normalization audit map through the existing candidate-set facts closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, and canonicalCandidateSet_sound. Those facts supply the candidate membership and soundness side of the route, while R and E supply the public router and endpoint rows. The resulting boundary is therefore candidate-mediated and evidence-bearing, not an unconditional closed-term normalizer. □

Falsifiable prediction: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ consumer that exports strong normalization from T without reading K, R, E refutes the carrier surface.

Independence witness: candidate-set-normalization, metacicnormalizationcertificaterouter, metacicopenproblemledger: the candidate-set packet carries candidate evidence, the router selects the admissible normalization route, and the open-problem ledger records the frontier question; this packet binds those rows into a closed-projection boundary without adding an evidence-free normalization theorem.

Closure status for NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$.

Constructive story: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate-set evidence, normalization-router selection, bounded-normal endpoint seal, no-infinite-reduction row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4366.1, Definition 4366.2, Definition 4366.3, and Definition 4366.4 seed the finite closed-projection, candidate-evidence, router-selection, endpoint-seal, obstruction, public-boundary, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional strong normalization for all closed typed MetaCIC terms, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Upgrade path: Obligation closure requires checked closed-term projection, candidate-evidence admission, router selection, bounded-normal endpoint sealing, no-infinite-reduction handoff, obstruction exactness, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.

Chapter 4262

A Concrete Naming Certificate for $\mathsf{PointedMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based metric spaces, the $\mathsf{PointedMetricSpace}^{\uparrow}$ packet records a metric carrier together with a named basepoint route before completion, loop-space, or embedding consumers are allowed to inspect point-level data. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, reads names through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, passes regular rational distance observations through chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and offers a based source for the distance-preserving consumer route of chapter 439. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected representatives, or ambient completeness.

Definition 4262.1 (Pointed metric space carrier)**.** *A* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, b, D, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *b is the basepoint name row, stored as finite BEDC data attached to M rather than as a host-level point.*
3. *D is the basepoint distance ledger, recording the finite requests that compare b with inspected metric names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which basepoint and metric names are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the basepoint-distance observations.*
7. *H records componentwise* hsame *transport for M, b, D, S, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a based metric request through the stream windows, regular rational readback, and real-distance seal.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{PointedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint name, distance ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate single carrier alignment`

Theorem 4262.2 (Pointed metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier* $P = (M, b, D, S, R, E, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint name, basepoint distance ledger, stream windows, regular rational readback, real-distance seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring every comparison of basepoint data to be read over the transported metric row* M*;*
3. *basepoint-distance exposure, requiring every basepoint comparison to pass through* D, S, R, E *before any completion-facing or embedding-facing consumer reads it;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 4367.1. The projection exposes the metric row, basepoint name, distance ledger, stream windows, regular readback, real seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, b, D, S, R, E together, so a consumer cannot compare a basepoint name over one metric carrier with distance observations over another.

The distance ledger fixes the read order. A basepoint comparison is first a finite request in D, then a scheduled stream observation in S, then a regular rational readback in R, and only then a $\mathsf{Real}^{\uparrow}$ distance seal in E.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected representatives, or ambient completeness, the listed rows exhaust the local naming surface. □

Theorem 4262.3 (Pointed metric space basepoint nonescape)**.** *Let* $P = (M, b, D, S, R, E, H, C, P_0, N)$ *be an accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier. Every completion-facing, loop-space-facing, or isometric-embedding-facing consumer of the basepoint row* b *must read it through the displayed route*

$$M, b, D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P_0, N.$$

Thus the basepoint is exported only as finite named metric data over M*, not as a host point, quotient representative, selected completion point, or ambient metric-equality witness.*

Proof. Begin with carrier admission. The only row naming the basepoint is b, and it is stored beside the metric carrier M and distance ledger D.

Read the inspection route next. The stream windows S and regular rational readback R display every observed distance from the basepoint before the $\mathsf{Real}^{\uparrow}$ seal E is available to downstream consumers.

Transport and replay preserve that same route. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add host points, quotient representatives, selected completion points, or ambient metric equality.

Therefore every based consumer receives exactly the finite route displayed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedMetricSpace}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a named basepoint row, a basepoint distance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4367.1, Definition 4367.2, and Definition 4367.3 seed the based metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host point equality, quotient metric equality, selected completion points, loop-space closure, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, basepoint-distance exposure, classifier transport, basepoint nonescape, and downstream completion and isometric-embedding consumer alignment for BEDC.Derived.PointedMetricSpaceUp.

Chapter 4263

A Concrete Naming Certificate for $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based complete metric spaces, the $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet records a based metric carrier together with a displayed completion service before completion, loop-space, or embedding consumers are allowed to read a point as complete. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service boundary of chapter 128, the $\mathsf{PointedMetricSpace}^{\uparrow}$ basepoint route of chapter 4367, the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected completion representatives, countable choice, or ambient order completeness.

Definition 4263.1 (Pointed complete metric space carrier)**.** *A* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, B, Q, S, R, E, L, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *B is the basepoint row, stored as finite named metric data over M rather than as a host-level point.*
3. *Q is the Cauchy-window route, listing the finite tail requests through which regular Cauchy names based at B are inspected.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the basepoint and Cauchy windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance and modulus observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed metric-distance and limit-distance observations.*
7. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion seal, exposing only the displayed limit-service handoff for the Cauchy-window route Q.*
8. *H records componentwise* hsame *transport for M, B, Q, S, R, E, L and the local certificate rows.*
9. *C records displayed* Cont *replay from a based complete-metric request through the Cauchy windows, stream observations, regular rational readback, real seal, and completion seal.*

10. *P_0 is the* Pkg *provenance over* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.TasteGate.PointedCompleteMetricSpaceTasteGate single carrier alignment`

Theorem 4263.2 (Pointed complete metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier* $P = (M, B, Q, S, R, E, L, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSp}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint row, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring the basepoint row, Cauchy windows, and completion seal to be read over the transported metric row M;*
3. *Cauchy-window exposure, requiring every completion-facing read to pass through Q, S, R, E, L before any downstream consumer may inspect the limit handoff;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, selected completion representative, countable-choice selector, or ambient order-completeness principle.

Proof. Project the carrier of Definition 4368.1. The projection exposes the metric row, basepoint, Cauchy-window route, stream windows, regular readback, real seal, completion seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, B, Q, S, R, E, L together, so a consumer cannot compare a based Cauchy route over one metric carrier with a completion seal over another.

The completion-facing read order is fixed by the route rows. A completion request begins as a finite Cauchy-window request in Q, becomes a scheduled stream observation in S, is read back as regular rational data in R, is sealed as real-distance data in E, and only then reaches the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service row L.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected completion representatives, countable-choice selectors, or ambient order completeness, the listed rows exhaust the local naming surface. □

Theorem 4263.3 (Pointed complete metric space Cauchy basepoint route)**.** *Let* $P = (M, B, Q, S, R, E, L, H, C, P_0$ *be an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier. Every based regular Cauchy request whose initial window is anchored at B is read through the displayed route*

$$M, B, Q \longmapsto S \longmapsto R \longmapsto E \longmapsto L \longmapsto H, C, P_0, N.$$

Thus the basepoint participates in completion only as named metric and Cauchy-window data over M, not as a selected host point or completed-limit representative.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.PointedCompleteMetricSpaceCauchyBasepointRoute`

Proof. Begin with carrier admission. The only row naming the basepoint is B, and it is stored beside the metric row M and the Cauchy-window route Q.

Read the window route next. The stream row S schedules the inspected windows, the regular rational row R reads the tail and distance ledgers, and the $\mathsf{Real}^{\uparrow}$ row E seals the observed distances before the complete-metric row L can expose any limit-service handoff.

Transport and replay preserve that same path. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add a host basepoint, a selected limit representative, quotient equality, or a separate completion object.

Therefore every based Cauchy consumer receives exactly the finite route displayed in the statement. □

Theorem 4263.4 (Pointed complete metric basepoint completion handoff)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ and $P' = (M', B', Q', S', R', E', L', H', C', P'_0, N')$ be accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carriers. If the two packets transport M, B, Q, S, R, E through the same componentwise H row, then every* $\mathsf{CompleteMetric}^{\uparrow}$ *service read of L is taken over that same named basepoint route before any loop-space or embedding consumer may inspect a completed point.*

Equivalently, a consumer may inspect

$$L \quad \text{or} \quad L'$$

only after the route

$$M, B, Q, S, R, E \longmapsto H, C, P_0, N$$

has been displayed; it cannot detach completion service from the transported basepoint row.

Proof. Project both carriers. In each packet the completion-service coordinate is L, and the only rows upstream of it are the metric row M, the basepoint row B, the Cauchy-window row Q, the stream row S, the regular rational row R, and the real-distance seal E.

Apply Definition 4368.3 to the first packet. The displayed route forces every based Cauchy request to pass through M, B, Q, S, R, E before the service row L is visible.

Use the assumed componentwise transport row H to move the same route to the second packet. The replay row C then reproduces the transported route for the completion-facing consumer, and the provenance and local name rows P_0, N record no independent completed point.

Thus a loop-space or embedding consumer can inspect a completed point only after the transported basepoint route has been displayed. There is no carrier coordinate from which the consumer could read L apart from B. □

Theorem 4263.5 (Pointed complete metric space completion nonescape)**.** *Every completion-facing, loop-space-facing, or embedding-facing consumer of an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *packet factors its completed-point read through*

$$M, B, Q, S, R, E, L, H, C, P_0, N.$$

Consequently the packet exports a based complete metric certificate only as finite BEDC NameCert data; it exports neither host points nor quotient completion equality.

Proof. Inspect the carrier coordinates. The only completion coordinate is L, and the definition makes L a completion seal for the already displayed Cauchy-window route Q over the metric row M and basepoint row B.

The readback rows precede the completion seal. Any consumer must first pass through S, R, E to display stream windows, regular rational ledgers, and real distance seals, so no completed point can be read before the finite analytic route is visible.

The remaining rows are administrative in the BEDC sense. Transport H, replay C, provenance P_0, and local name N preserve and source the packet but do not create host points, quotient completion equality, selected representatives, or ambient completeness.

Thus every downstream completed-point read factors through the displayed finite packet, which proves completion nonescape. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet is generated from a MetricUp carrier, a named basepoint row, a Cauchy-window route, StreamName windows, RegSeqRat readback, a RealUp seal, a CompleteMetricUp completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4368.1, Definition 4368.2, Definition 4368.3, and Definition 4368.5 seed the based complete-metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient completion equality, selected host endpoints, loop-space closure, embedding closure, or ambient order completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, classifier transport, basepoint completion routing, completion nonescape, and downstream completion, loop-space, and embedding consumer alignment for BEDC.Derived.PointedCompleteMetricSpaceUp.

Chapter 4264

A Concrete Naming Certificate for $\mathsf{IntervalConnectedness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{IntervalConnectedness}^{\uparrow}$ packet records the finite constructive route by which a located real interval is inspected as one connected interval surface before intermediate-value, zero-envelope, or completion-facing consumers use it. It sits after the endpoint seal of chapter 38, the located interval carrier of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sandwich consumer surface of chapter 2163. The packet names interval connectedness as displayed finite BEDC data; it does not import host topology, open-cover compactness, trichotomy, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4264.1 (Interval connectedness carrier)**.** *An* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint-pair row, storing the lower and upper endpoint names through their displayed real seals.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *witness row, carrying the located interval budget, locator reads, and endpoint-radius checks attached to E.*
3. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision-chain ledger, recording the dyadic cells, adjacent-cell overlaps, and tolerance requests inspected inside L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, cell, and witness observations are scheduled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled endpoint and subdivision observations.*
6. *T records componentwise* hsame *transport for endpoint pairs, located interval witnesses, dyadic subdivision chains, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint pairs through located interval exposure, dyadic subdivision, stream-window inspection, regular readback, and the interval-connectedness handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalConnectedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SqueezeTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, D, S, R, T, C, P, N$. It has no coordinate for host topological connectedness, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalConnectednessUp.IntervalConnectednessTasteGate single carrier alignment`

Theorem 4264.2 (Interval connectedness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier*

$$I = (E, L, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint pair, located interval witness, dyadic subdivision chain, stream windows, regular readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring endpoint, locatedness, subdivision, window, readback, replay, provenance, and naming data to move componentwise through T;*
3. *dyadic chain exactness, requiring every inspected subinterval and adjacent-cell overlap to be recorded in D before a connectedness consumer reads the packet;*
4. *endpoint handoff nonescape, requiring every consumer of interval endpoints to pass through E, L, D, S, R before using any* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger exhaustion, requiring the packet to expose no host topological connectedness theorem, open-cover compactness, trichotomy, quotient real equality, selected intermediate point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The sibling routes through $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{SqueezeTheorem}^{\uparrow}$ *are the only named consumers of this local certificate surface.*

Proof. Project the carrier of Definition 4369.1. The endpoint pair is visible only in E, and the located interval evidence is visible only in L.

The subdivision data are then forced through D. Each finite cell, overlap, and tolerance request is read as dyadic data before any stream-window or regular-readback consumer can inspect it.

Transport and replay preserve the same route. The rows T and C move endpoint, locatedness, subdivision, window, and readback data componentwise, while P and N source and name exactly that displayed packet.

Since the carrier has no coordinate for the excluded host principles, the local certificate cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{IntervalConnectedness}^{\uparrow}$ consumer reads endpoint pairs, located interval witnesses, dyadic subdivision chains, StreamName windows, RegSeqRat readbacks, hsame transport, Cont replay, Pkg provenance, and local naming before any interval-connectedness handoff is visible.

Independence witness: real, locatedrealinterval, dyadicratcore, streamname, regseqrat, squeezetheorem: $\mathsf{IntervalConnectedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, located interval witnesses, finite dyadic subdivision chains, scheduled stream windows, regular rational readbacks, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$.

Constructive story: An IntervalConnectedness$^{\uparrow}$ packet is generated from Real endpoint pairs, a LocatedRealInterval witness, a finite DyadicRatCore subdivision chain, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, seed)

Scope closed: Definition 4369.1 and Definition 4369.2 seed the finite located-interval connectedness route over Real$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and SqueezeTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topology, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic chain exactness, endpoint handoff nonescape, ledger exhaustion, and downstream intermediate-value consumer alignment for BEDC.Derived.IntervalConnectedness

Chapter 4265

A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$

Following Tait–Martin-Lof and Barendregt on residuals, parallel reduction, and confluence discipline, the MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ packet records the finite boundary where a beta-parallel branch and a substitution replay must meet before the MetaCIC confluence route may use the conditional diamond bridge. It sits over the audit map of chapter 9378 and names the residual-substitution input as packet data rather than as a global Church–Rosser theorem. The packet does not claim full closed Church–Rosser, arbitrary closed-source confluence, or an unrestricted substitution theorem.

The residual-substitution packet is consumed by chapter 765 as a guarded input to candidate-mediated SN/confluence scheduling. The dependency is the residual join row together with the closedness guard data: it tells the scheduler which finite substitution replay may enter a candidate branch and which source windows are closed enough for that handoff, not that the full Church–Rosser theorem has already been completed.

Definition 4265.1 (MetaCIC residual-substitution compatibility carrier)**.** *A* MetaCICResidualSubstitutionCompatil *carrier is a finite* BHist *packet*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

with the following displayed rows:

1. *s is the source term row whose beta-parallel residual branch is inspected.*
2. *t is the substituted term row, storing the finite substitution request and the target occurrence window.*
3. *b is the parallel branch row, displaying the left and right beta-parallel steps before any join is named.*
4. *j is the residual join row, recording the candidate common reduct obtained after substitution replay and residual transport.*
5. *g is the closedness guard row, saying exactly which source and substitution windows are closed enough for the guarded handoff.*
6. *a is the atom-shape separation row, keeping atom, application, lambda, Pi, and beta-redex cases visibly disjoint.*
7. *h records componentwise* hsame *transport for source, substitution, branch, residual join, closedness guard, atom-shape separation, replay, provenance, and local naming rows.*

8. *c records displayed* Cont *replay from the source term through substitution, parallel branching, residual join construction, closedness guarding, and atom-shape separation.*
9. *p is the* Pkg *provenance over* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$*, MetaCIC beta-parallel reduction, residual joins, closedness guards, atom-shape separation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *n is the local* NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, t, b, j, g, a, h, c, p, n$. It has no coordinate for a completed parallel diamond, full closed Church–Rosser, arbitrary closed-source confluence, or an unguarded substitution principle.

Lean4 ✓ BEDC.Derived.MetaCICResidualSubstitutionCompatibilityUp.MetaCICResidualSubstitutionCompatibilit
single carrier alignment

Theorem 4265.2 (MetaCIC residual-substitution compatibility NameCert obligations)**.** *For every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the local NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and local naming rows;*
2. *branch-shape disjointness, requiring atom, application, lambda, Pi, and beta-redex branches to be separated by a before a residual join row is accepted;*
3. *substitution residual replay, requiring c to display how the substituted term row and beta-parallel branch row reach the same candidate join j;*
4. *closed-source guard, requiring g to expose the exact closedness assumptions consumed by the guarded compatibility packet;*
5. *non-claim of full closed Church–Rosser, requiring the packet to remain an input boundary for the conditional confluence bridge rather than a proof of arbitrary closed-source confluence.*

No obligation is stored outside $s, t, b, j, g, a, h, c, p, n$.

Proof. Project the carrier of Definition 4370.1. The source term, substituted term, branch, residual join, closedness guard, and atom-shape separation rows are the only computational coordinates.

The branch-shape row a fixes the case split before the join row is read. Thus a beta-redex residual branch cannot be silently identified with an atom, application, lambda, or Pi shape.

Replay then supplies the compatibility boundary. The row c records the finite route from substitution and beta-parallel branching to the candidate join j, while g states the closedness guard consumed by that route.

Finally h, p, n transport, source, and name exactly the displayed packet. Since full closed Church–Rosser and arbitrary closed-source confluence are not carrier coordinates, they are not exported by this NameCert surface. □

Theorem 4265.3 (MetaCIC residual-substitution candidate projection route)**.** *Every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

exports a candidate-closed projection route to chapter 4366 only after the source term row s, substituted term row t, beta-parallel branch row b, residual join row j, closedness guard row g, and atom-shape separation row a have all been displayed. The downstream candidate boundary receives a guarded compatibility packet, not full Church–Rosser and not arbitrary closed-source confluence.

Proof. Project the carrier of Definition 4370.1. The rows s, t, b, j, g, a are the only computational rows available before the structural transport, replay, provenance, and local naming rows are read.

Use Definition 4370.2. Branch-shape disjointness exposes a, substitution residual replay exposes the route from t and b to j, and the closed-source guard exposes exactly the assumptions in g.

The candidate boundary of chapter 4366 is therefore reached through a guarded compatibility packet. The route supplies source, substitution, beta-parallel, residual-join, closedness, and atom-shape data, while h, c, p, n only transport, replay, source, and name those rows.

Since completed Church–Rosser and arbitrary closed-source confluence are absent from the carrier, the candidate-closed projection route cannot export them. It hands off only the displayed residual-substitution compatibility boundary. □

Theorem 4265.4 (MetaCIC residual-substitution obligation closure surface)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the obligation surface is exactly the displayed source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and naming rows. The candidate-closed projection boundary receives only this guarded compatibility packet.

Proof. Apply Definition 4370.2. It identifies carrier admission, branch-shape disjointness, substitution residual replay, closed-source guarding, transport, replay, provenance, and naming as rows of $s, t, b, j, g, a, h, c, p, n$.

Apply Definition 4370.3. The candidate boundary receives the source, substitution, beta-parallel branch, residual-join, closedness, and atom-shape data only through that guarded projection route.

Therefore the obligation surface packages the candidate handoff without exporting full closed Church–Rosser, arbitrary closed-source confluence, or unguarded substitution compatibility. Those claims are not coordinates of the carrier and are not supplied by the candidate projection. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ read displays source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, hsame transport, Cont replay, Pkg provenance, and local naming rows before the conditional MetaCIC confluence bridge can consume it.

Independence witness: metacic-confluence-audit: $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ is not a completed confluence theorem because it carries only the finite residual-substitution compatibility boundary consumed by the conditional parallel-diamond route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ packet is generated from source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4370.1, Definition 4370.2, and Definition 4370.3 seed the finite residual-substitution compatibility boundary consumed by the conditional MetaCIC confluence bridge and the candidate-closed projection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, arbitrary closed-source confluence, unguarded substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-shape disjointness, substitution residual replay, closed-source guard, and conditional confluence handoff alignment for BEDC.MetaCIC.BetaParallelSubstitutionCompatibility.

Chapter 4266

A Concrete Naming Certificate for $\mathsf{ComputableCompactMetric}^{\uparrow}$

Following Weihrauch-style computable analysis and the Bishop–Bridges compact metric arguments, the $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet records the effective compact-metric surface as finite BEDC NameCert data. It sits after the compact metric carrier of chapter 129, the effective Polish source of chapter 1706, the total-bounded finite-net route of chapter 127, the complete metric source of chapter 128, and the L10 readback route through chapter 126, chapter 301, and chapter 38. The packet names the finite-cover realizer surface consumed by effective compact analysis; it does not assert open-cover compactness, selected subcovers, quotient completion equality, or an ambient real-completeness principle.

Definition 4266.1 (Computable compact metric carrier)**.** *A* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact metric packet whose finite-net behavior is being made effective.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *source row exposing the effective dense-name and semidecidable basic-ball interface.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row naming the finite-net source for each displayed dyadic tolerance request.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row recording the convergence service available to Cauchy and compact consumers.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for inspected dense names, finite net centers, tolerance requests, and cover indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of distances and tolerance comparisons.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows and regular rational comparisons have been displayed.*

8. *H records componentwise* hsame *transport for compact metric, effective Polish, total-bounded, complete-metric, stream-window, regular-readback, real-seal, finite-cover realizer, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from an effective compactness request through K, E, T, C, S, R, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ComputableCompactMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, T, C, S, R, L, H, Q, P, N$. It has no coordinate for an open-cover subcover, a choice-selected finite set, quotient completion equality, excluded-middle compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ComputableCompactMetricUp.ComputableCompactMetricTasteGate single carrier alignment`

Theorem 4266.2 (Computable compact metric NameCert obligations)**.** *For every accepted* ComputableCompactMe *carrier*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *surface consists exactly of the displayed rows in Definition 4371.1. Carrier admission must expose the compact metric source, effective Polish source, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through those same rows, and ledger exhaustion says that every finite-cover or compact-analysis consumer can inspect only $K, E, T, C, S, R, L, H, Q, P, N$.*

Proof. Project the carrier of Definition 4371.1. The rows K, E, T, C are the only structural compactness sources, so carrier admission exposes compact metric data, effective Polish naming data, total-bounded finite nets, and complete-metric convergence before any realizer can be read.

The readback rows S, R, L then bind numerical access to finite windows, regular rational comparisons, and the terminal real seal. This is the same L10 readback discipline used by the StreamName, RegSeqRat, and Real siblings named in the chapter introduction.

Transport and replay are visible rows rather than hidden meta-operations. The row H records componentwise hsame transport, Q records displayed Cont replay, and P, N source and name the same finite packet.

For ledger exhaustion, read any compact-analysis consumer from the accepted packet. The carrier exposes only $K, E, T, C, S, R, L, H, Q, P, N$, so no open-cover subcover, selected finite set, quotient completion equality, excluded-middle compactness, or ambient real-completeness principle can be obtained from this NameCert surface. □

Theorem 4266.3 (Computable compact metric finite-cover realizer)**.** *Every effective finite-cover read exported by an accepted* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *packet factors through the route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsa

Thus a consumer receives a compact metric source, effective dense-name interface, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational tolerance comparisons, terminal real seal, componentwise transport, replay, provenance, and local

naming rows before it uses the finite-cover realizer. The packet exports no selected subcover, open-cover compactness theorem, quotient completion representative, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an accepted carrier $K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$. The compact metric row K and effective Polish row E name the metric and effective dense-name surfaces, while T supplies the finite-net source used for the requested tolerance.

The convergence row C keeps the compact route compatible with complete-metric consumers. A finite-cover realizer cannot leave these displayed sources, because the later rows are read only after the compact, effective, total-bounded, and complete rows are present.

The numerical readback is finite. The stream row S exposes inspected names, center windows, cover indices, and tolerance requests; the regular rational row R compares the displayed distances; and the terminal seal L is reached only after those windows and comparisons have been read.

Finally H, Q, P, N preserve the route under transport, replay, provenance, and local naming. Since selected subcovers, open-cover compactness theorems, quotient representatives, choice selectors, and ambient completeness witnesses are not carrier coordinates, the exported realizer is exactly the displayed finite-cover surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet is generated from CompactMetric, EffectivePolishSpace, TotallyBounded, CompleteMetric, StreamName, RegSeqRat, RealUp, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4371.1, Definition 4371.2, and Definition 4371.3 seed the effective finite-cover realizer route for computable compact metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, open-cover compactness, quotient completion equality, countable-choice selectors, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective dense-name exposure, total-bounded finite-net exposure, complete-metric convergence exposure, finite-window readback, regular rational comparison, real sealing, finite-cover realizer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ComputableCompactMetricUp.

Chapter 4267

A Concrete Naming Certificate for $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet records the separated-completion reflection route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the regular rational readback surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness boundary of chapter 3390. The packet lets a Real-facing consumer treat two representative regular Cauchy names as the same separated reflection point only after the displayed finite windows, dyadic budgets, and uniqueness ledger have been exposed.

Definition 4267.1 (Regular-Cauchy Hausdorff-reflection carrier)**.** *A* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row presented to the same separated-reflection request.*
3. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row witnessing that the displayed tail comparisons force separated agreement.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, Y, S, T, D, U have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, stream windows, dyadic ledger, uniqueness row, Real seal, and local certificate rows.*
8. *C records the displayed* Cont *replay from a reflection request to the paired windows, dyadic tolerance ledger, uniqueness row, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier admission surface`

Theorem 4267.2 (Regular-Cauchy Hausdorff-reflection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier* $R = (X, Y, S, T, D, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *separated-agreement row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4372.1. The only displayed coordinates are $X, Y, S, T, D, U, E, H, C, P, N$.

The two regular rational names are compared through the paired stream windows S, T and the dyadic tolerance ledger D. The separated-agreement row U is therefore a displayed uniqueness boundary, not a quotient of streams or a host equality proof.

The terminal Real seal E is downstream of that boundary. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and no excluded quotient, selector, host-equality, choice, trichotomy, or ambient-completeness coordinate can be projected from the carrier. □

Theorem 4267.3 (Regular-Cauchy Hausdorff-reflection representative independence)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet is independent of the chosen representative regular Cauchy name in the following BEDC sense: if the two presented* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are compared through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the terminal seal can distinguish them only through the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row. The packet exports no quotient stream equality, selected representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionRepresentativel`

Proof. Begin with an accepted carrier $R = (X, Y, S, T, D, U, E, H, C, P, N)$. The rows X and Y are not presented as completed real objects; they are presented as regular rational names observed through finite stream windows.

The dyadic ledger D binds the comparison budget for those windows, and the uniqueness row U is the only row that can turn the displayed tail comparison into separated agreement. Thus any Real-facing read of E must pass through the same finite comparison route.

The rows H, C, P, N transport, replay, source, and name this route without adding analytic content. Since the carrier contains no further equality coordinate, changing representatives cannot be detected except through the displayed uniqueness row. □

Theorem 4267.4 (Regular-Cauchy Hausdorff-reflection Real seal non-escape)**.** *No terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the separated reflection supplies representative independence for the Real seal without quotient streams, host real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier real seal nonescape`

Proof. Read the carrier coordinates in order. The terminal row E appears only after the two regular rational names, paired stream windows, dyadic tolerance ledger, and uniqueness row have been displayed.

Apply Definition 4372.2. The local NameCert surface exposes exactly those rows, with H, C, P, N preserving the same route for downstream consumers.

The advertised route is therefore the only public path to the Real seal. Since no excluded equality or choice coordinate occurs in the carrier, the seal cannot escape through quotient streams, host equality, representatives selected outside the packet, countable choice, trichotomy, or ambient completeness. □

Theorem 4267.5 (Regular-Cauchy Hausdorff-reflection finite-window uniqueness route)**.** *Suppose two accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packets display the same paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then every separated-reflection consumer factors through the same finite-window route on the underlying* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization does not use quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier finite window uniqueness route`

Proof. Apply Definition 4372.3. Once the paired StreamName windows, dyadic tolerance ledger, and uniqueness row agree, the terminal Real seal cannot distinguish the two regular Cauchy representatives by any route other than the displayed uniqueness row.

Apply Definition 4372.4. The Real seal is downstream of the finite windows, dyadic ledger, and uniqueness row, so a separated-reflection consumer has no bypass around those displayed rows.

The remaining rows are structural. The hsame rows transport only the displayed components, the Cont rows replay the same finite-window request, and the Pkg and NameCert rows name that route inside BHist. Therefore the consumer factors through the same finite-window route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore tolerance ledger, and a RegularCauchyUniqueness row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyuniqueness: $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational representatives to one separated uniqueness row and one downstream Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyUniqueness separated-agreement row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4372.1, Definition 4372.2, Definition 4372.3, Definition 4372.4, and Definition 4372.5 seed the separated reflection route for regular Cauchy representatives and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, representative-independence handoff, and Real seal non-escape theorem.

Chapter 4268

A Concrete Naming Certificate for $\mathsf{WeakLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on finite averages and probability estimates, the $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet names the displayed finite-average route behind the weak law of large numbers. It sits after the normalized probability surface of chapter 184, the measurable random-variable rows of chapter 185, the finite tail-bound carrier of chapter 6592, and the terminal real-name seal of chapter 38. The packet is a finite certificate for Chebyshev-controlled averages; it is not an almost-sure convergence theorem, a central limit theorem, a countable-choice subsequence selector, a host probability-limit principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4268.1 (Weak law finite-average carrier)**.** *A* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, R, A, V, T, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the total-event normalization and event classifier. The row R is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over that same probability source. The row A records the displayed finite average of the entries in R, while V records the variance or uniform second-moment bound used by the estimate. The row T reads the Chebyshev tail row through* $\mathsf{ChebyshevInequality}^{\uparrow}$ *for the event where the finite average differs from its displayed center by more than the requested tolerance. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the final probability bound. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite averaging route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these ten rows.*

Lean4 ✓ `BEDC.Derived.WeakLawLargeNumbersUp.WeakLawLargeNumbersTasteGate single carrier alignment`

Theorem 4268.2 (Weak law NameCert obligations)**.** *For every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier $(P, R, A, V, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 4373.1. Carrier admission displays the probability source, the random-variable family, the finite-average row, the variance-bound row, the Chebyshev tail row, the epsilon seal, the transport and replay rows, the provenance row, and the local naming row. The packet imports no host law of large numbers, selected subsequence, external probability-limit operation, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The probability source P supplies the measurable-event classifier and total-mass normalization, and the random-variable row R exposes only the finitely displayed variables over that source.

The average row A and variance row V are then internal finite data. They record the algebraic averaging expression and the bound used by the tail estimate before any limiting statement is visible.

Finally read T, E, H, C, Q, N. The Chebyshev tail row and epsilon seal name the probability estimate at the requested tolerance, while transport, replay, provenance, and local naming keep the same finite route under classifier replacement. Since no carrier coordinate stores an infinite limit or subsequence selector, the listed rows exhaust the local certificate surface. □

Theorem 4268.3 (Weak law Chebyshev handoff)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *Chebyshev read factors through the route*

$$P \longmapsto R \longmapsto A \longmapsto V \longmapsto T \longmapsto E \longmapsto H, C, Q, N$$

before any probability-bound consumer receives the finite-average estimate. Thus the Chebyshev step is handed to later probability and $\mathsf{Real}^{\uparrow}$ *formalization as a displayed finite source, finite random-variable row, average row, variance-bound row, tail-event row, epsilon seal, transport, replay, provenance, and local naming data.*

Proof. Start with a probability-bound request. The only probability source exposed by the packet is P, and the only random-variable data exposed over it are the finite rows in R.

The request must next read the average and variance-bound rows. The average row A fixes the finite expression being estimated, and V fixes the bound that the Chebyshev tail row may consume.

Only after these rows are displayed can T and E be read as a bound on the exceptional event. The structural rows H, C, Q, N then preserve that same route for downstream consumers, so the handoff is the finite Chebyshev estimate itself rather than a host-side convergence theorem. □

Theorem 4268.4 (Weak law finite-average nonescape)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *probability-limit-facing read remains inside the finite-average route of Definition 4373.1. In particular, a consumer can read a tail estimate only after displaying the source probability row, finite random-variable rows, average row, variance-bound row, Chebyshev tail row,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by almost-sure convergence, central limit normalization, countable-choice subsequence selection, a host probability-limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a consumer request the finite-average estimate from an accepted carrier. Projection of Definition 4373.1 shows that the request first meets P, R, A, V, T, E and has no independent coordinate for a limit object.

The displayed Chebyshev row T is downstream from the variance-bound row V and upstream from the epsilon seal E. Hence any probability bound read is already tied to the visible finite average and its tolerance.

The remaining rows H, C, Q, N do not add new analytic content. They transport equal displayed histories, replay the finite route, record the package provenance, and attach the local name. Therefore every accepted probability-limit-facing read stays inside the finite-average certificate surface and cannot escape to the excluded convergence or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet is generated from a probability source row, finite random-variable rows, a finite-average row, a variance or second-moment bound row, a Chebyshev tail row, a RealUp epsilon seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4373.1, Definition 4373.2, Definition 4373.3, and Definition 4373.4 seed the finite-average probability route through ProbSpace, RandomVar, RealUp, Chebyshev tail, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the central limit theorem, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-average routing, variance-bound stability, Chebyshev tail handoff, epsilon-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.WeakLawLargeNumbe

Chapter 4269

A Concrete Naming Certificate for $\mathsf{LocatedSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric and real spaces, the $\mathsf{LocatedSubset}^{\uparrow}$ packet records a displayed subset together with finite distance-witness rows. It sits after the real-valued metric route of chapter 1638, the Hausdorff comparison boundary of chapter 2086, the real-separability window surface of chapter 1809, the dyadic tolerance core of chapter 291, and the regular rational readback surface of chapter 301. The packet gives Real and Completion consumers a finite located-subset interface without host set extensionality, selected nearest points, quotient subset equality, or ambient completeness.

The located-subset distance route consumes chapter 1809 as its finite search-window source and consumes chapter 2086 only after the real-distance seal has been displayed.

Definition 4269.1 (LocatedSubset carrier)**.** *A* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, W, D, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *source row, carrying the ambient metric distance through finite real-name observations.*
2. *S is the subset-admission row, naming the displayed members read by the carrier and exporting no host powerset equality.*
3. *W is the* $\mathsf{RealSeparability}^{\uparrow}$ *search-window row, recording the countable dense or finite-window probes available to a distance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to each positive distance query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison data produced from W and D.*
6. *E is the real distance seal row, read only after the metric source, subset admission, search windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *K is the optional* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison boundary for compact or hyperspace consumers; it records only the finite directed comparisons visible through the preceding rows.*
8. *H records componentwise* hsame *transport for the metric source, subset admission, search windows, dyadic ledger, regular readback, real seal, Hausdorff boundary, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an ambient point and tolerance request through subset admission, separability windows, dyadic comparison, regular rational readback, real sealing, and any Hausdorff-facing comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSubset}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, W, D, R, E, K, H, C, P, N$. It has no coordinate for a host subset object, extensional equality of subsets, a selected nearest point, quotient real equality, countable choice, or an ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate single carrier alignment`

Theorem 4269.2 (LocatedSubset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, S, W, D, R, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, optional Hausdorff comparison routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4374.1. The displayed packet lists exactly the rows used by an ambient point, subset-membership, tolerance, rational-readback, real-seal, or Hausdorff-facing request.

The metric and subset rows M, S are read first. A distance query can inspect the ambient space only through $\mathsf{RealMetric}^{\uparrow}$, and it can inspect membership only through the displayed admission row S.

The search, tolerance, and readback rows W, D, R then supply finite witnesses for the locatedness request. The real seal E is reached only after those rows have been displayed, while K is available only as a downstream comparison boundary.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host subset, selected nearest point, quotient equality, countable-choice schedule, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4269.3 (LocatedSubset dyadic distance witness)**.** *Every distance read exported by an accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto E.$$

The read may be handed to a $\mathsf{HausdorffMetric}^{\uparrow}$ *consumer only after the same route has produced the displayed real-distance seal, transported by H and replayed by C.*

Proof. Begin with a point-to-subset distance request. The ambient metric row M supplies the only metric observations, and the subset-admission row S supplies the only visible membership data.

The finite search window W names the inspected probes. The dyadic ledger D fixes the positive tolerance requests, and the regular rational row R records the resulting readback windows.

The real-distance seal E is therefore downstream of M, S, W, D, R. Any Hausdorff-facing row K can compare only the finite distance data already sealed by this route.

Transport and replay do not add another distance object. The componentwise hsame row H, the displayed Cont route C, the Pkg provenance P, and the local NameCert row N preserve the

same witness surface, so no nearest-point selector or host infimum can bypass the dyadic distance route. □

Falsifiable prediction: Every accepted LocatedSubsetUp distance read exposes RealMetricUp, subset admission, RealSeparabilityUp windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, and a real-distance seal before any HausdorffMetricUp or completion-facing consumer reads the packet.

Independence witness: realmetric, hausdorffmetric, realseparability, dyadicratcore, regseqrat: none of these siblings alone carries the full located-subset packet with subset admission, finite search windows, dyadic tolerance rows, regular rational readback, real-distance sealing, optional Hausdorff comparison, transport, replay, package provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}$.

Constructive story: A $\mathsf{LocatedSubset}^{\uparrow}$ packet is generated from a RealMetricUp source, displayed subset admission, RealSeparabilityUp search windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a real-distance seal, optional HausdorffMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4374.1, Definition 4374.2, and Definition 4374.3 seed the finite located-subset interface for real-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host subset extensionality, selected nearest points, quotient subset equality, countable choice, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, Hausdorff comparison routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSubsetUp.

Chapter 4270

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 4270.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 4270.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 4270.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 4270.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 4270.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 4270.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 4271

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 4271.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 4271.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 4271.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 4272

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4272.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 4272.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4272.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 4273

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 4273.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 4273.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 4273.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 4273.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 4273.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 4273.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

Chapter 4274

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 4274.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 4274.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4274.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 4274.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\text{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

The DedekindCutStream$^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 4275

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 4275.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4275.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 4275.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 4275.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 4276

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 4276.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- M *is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- F *is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- B *is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- E *is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- I *is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- H *records componentwise* hsame *transport for* M, F, B, E, I *and the structural rows.*
- C *records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- P *is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- N *is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 4276.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 4276.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 4276.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

Chapter 4277

A Concrete Naming Certificate for $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$

Following Bishop and Bridges on constructive inverse and implicit function reasoning, the ConstructiveImplicitFunctio packet records a local solution graph as finite BEDC data rather than as a host-level selection of a function. It sits between the real-name surface of chapter 38, the metric ball and interval windows of chapter 1450, chapter 1638, and chapter 1709, the continuous-map rows of chapter 130, and the derivative boundary of chapter 122. The certificate exposes the local graph, nondegenerate derivative row, located window, contraction or modulus replay, Real readback, transport, provenance, and local naming rows before any consumer may read a local solve.

Definition 4277.1 (Constructive implicit function carrier)**.** *A* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, J, A, M, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Real}^{\uparrow}$ *graph row for the equation and its local output coordinate, read only through displayed real-name endpoints.*
2. *J is the derivative or Jacobian row, including the apartness or nondegeneracy witness used by the local solve.*
3. *A is the located interval or metric-ball window row, naming the input cell, output cell, and base point through* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{RealMetric}^{\uparrow}$*, or* $\mathsf{MetricBall}^{\uparrow}$ *data.*
4. *M is the contraction, Lipschitz, or continuity-modulus replay row that bounds the local correction inside A.*
5. *I is the terminal* $\mathsf{Real}^{\uparrow}$ *readback and seal row for the solved graph point.*
6. *H records componentwise* hsame *transport for graph rows, derivative rows, apartness witnesses, located windows, modulus rows, Real seals, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from F, J, A, M, I to the local solve row, including the* $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Derivative}^{\uparrow}$ *reads used by the graph.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{LocatedRealInterval}^{\uparrow}$*,* $\mathsf{RealMetric}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host implicit function, a selected germ, quotient function equality, global inverse theorem, ambient completeness principle, or classical choice.

Lean4 ✓ `BEDC.Derived.ConstructiveImplicitFunctionUp.ConstructiveImplicitFunctionTasteGate`
`single carrier alignment`

Theorem 4277.2 (Constructive implicit function NameCert obligations)**.** *For every accepted carrier $U = (F, J, A, M, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *carrier admission for the displayed graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local name rows;*
2. *local graph exactness, requiring every solved point read to pass through the graph row F and terminal Real seal I;*
3. *apartness or nondegeneracy stability, requiring J to transport its derivative witness through H without replacing it by host equality;*
4. *contraction or modulus replay, requiring M and C to keep correction steps inside the located window A;*
5. *uniqueness nonescape, requiring any two local solve reads with the same displayed rows to return to the same $F, J, A, M, I, H, C, P, N$ packet.*

No obligation is stored in a detached function object or an ambient completeness theorem.

Proof. Carrier admission is projection from Definition 4277.1. The packet lists exactly the graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local name rows.

The analytic obligations are local to the displayed rows. The graph exactness read uses F and I, while derivative stability uses J together with the componentwise transports in H.

The local solve is kept inside the named window by A, M, C. Since the carrier has no host implicit-function coordinate, selected germ, quotient function equality, or ambient completeness row, uniqueness is a nonescape statement about the same finite packet. □

Theorem 4277.3 (Constructive implicit function local solve route)**.** *Every* $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ *local-solve read factors through the displayed finite route*

$$\mathsf{ContinuousMap}^{\uparrow}, \mathsf{Derivative}^{\uparrow}, \mathsf{LocatedRealInterval}^{\uparrow}, \mathsf{MetricBall}^{\uparrow}, \mathsf{RealMetric}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

The consumer must open the graph row F, derivative or Jacobian row J, located window A, contraction or modulus row M, Real seal I, transport row H, replay row C, provenance row P, and local name row N before it can export a solved graph point.

Proof. Project the accepted carrier. The local-solve read has no available coordinate except $F, J, A, M, I, H, C, P, N$.

The graph and derivative rows are the analytic source. The located interval or metric-ball row names the window, and the contraction or modulus row records why the replay stays in that window.

Finally the Real seal and structural rows finish the route. The solve is therefore a displayed finite certificate over the named sibling chapters, not a host implicit function, selected germ, quotient of local functions, or classical existence principle. □

Falsifiable prediction: Every $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ local-solve read opens a displayed Real graph row, a derivative or Jacobian apartness row, a located interval or metric-ball row, a contraction or modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before exporting a solved graph point.

Independence witness: real, metricball, continuousmap, derivative, located-real-interval, real-metric: $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries local graph exactness, derivative nondegeneracy, located-window control, modulus replay, Real sealing, transport, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveImplicitFunction}^{\uparrow}$ packet is generated from a displayed Real graph row, a derivative or Jacobian row with apartness data, a located interval or metric-ball window, a contraction or modulus replay row, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4277.1, Definition 4277.2, and Definition 4277.3 seed the finite constructive implicit-function certificate over graph, derivative, located-window, modulus, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveImplicitFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host implicit functions, selected germs, quotient function equality, global inverse or implicit theorem packages, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, graph exactness, apartness stability, modulus replay, uniqueness nonescape, and local naming for BEDC.Derived.ConstructiveImplicitFunctionUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 4278

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4278.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4278.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4278.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 4278.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 4278.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 4278.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 4278.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 4278.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 4278.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 4278.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 4278.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 4278.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 4278.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 4278.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 4278.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 4279

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 4279.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4279.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 4279.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 4279.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 4279.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 4279.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 4279.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 4279.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 4279.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 4279.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 4280

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 4280.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 4280.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 4280.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let $L = (S, Q, I, U, F, K, T, H$ be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, seed)

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

Chapter 4281

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4281.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 4281.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 4281.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 4281.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 4282

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4282.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4282.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4282.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4283

A Concrete Naming Certificate for $\mathsf{CauchyCompletionRoute}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyCompletionRoute}^{\uparrow}$ packet records the regular-Cauchy completion path as finite NameCert data rather than as a scattered analytic slogan. It reads the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational sequence surface of chapter 301, the terminal real-name seal of chapter 38, and the Bishop regular-Cauchy completion packet of chapter 1639. The route names exactly the L10 path by which finite dyadic observations become a completion-facing real-name handoff; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, identify quotient streams, select limits, or import countable choice.

Definition 4283.1 (Cauchy completion route carrier)**.** *A* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, S, Q, E, B, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording endpoints, dyadic comparisons, precision requests, and the finite gap ledger used by the route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing only finite windows of the source stream before any regular-Cauchy read is admitted.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the rational sequence data and tail modulus visible through D and S.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint row, read only as the terminal seal produced from the displayed regular-Cauchy data.*
5. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *route row, tying D, S, Q, E to the established regular-Cauchy completion packet.*
6. *T records componentwise* hsame *transport for dyadic tolerances, stream windows, regular rational readback, terminal real seal, Bishop completion route, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a completion request through D, S, Q, B and then to E.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed rows. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionRouteUp.CauchyCompletionRouteTasteGate single carrier alignment`

Theorem 4283.2 (Cauchy completion route NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier* $R = (D, S, Q, E, B, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic tolerance, stream-window, regular rational, real seal, Bishop completion, transport, replay, provenance, and local naming rows;*
2. *dyadic observation discipline, requiring every completion-facing read to expose* D *and* S *before any regular rational or real seal row is consumed;*
3. *regularity routing, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* Q *to be the only passage from finite windows to a completion-facing regular sequence;*
4. *Bishop route compatibility, requiring* B *to identify the same dyadic, stream, regular rational, and real rows as the Bishop regular-Cauchy completion packet;*
5. *terminal seal discipline, requiring the* $\mathsf{Real}^{\uparrow}$ *row* E *to be read only after* D, S, Q, B *have been displayed;*
6. *transport and replay non-escape, requiring* T, C, P, N *to preserve the listed rows and to export no quotient stream equality, selected limit, host real equality, countable choice, or ambient completeness theorem.*

No NameCert obligation is stored outside $D, S, Q, E, B, T, C, P, N$*.*

Proof. Project the carrier of Definition 4283.1. The finite packet exposes only the dyadic tolerance row, stream-window row, regular rational row, terminal real seal, Bishop completion route, transport, replay, provenance, and local naming.

The dyadic and stream rows are upstream. Any completion-facing consumer first reads the displayed tolerance and finite observation data, so the regular rational row is not a detached sequence or a host limit object.

The Bishop route row then checks compatibility with the established regular-Cauchy completion packet. It ties the same dyadic, stream, regular rational, and real rows to the route and prevents a second completion carrier from being inserted.

Finally T, C, P, N transport, replay, source, and name the displayed route. Since the carrier lacks quotient, selected-limit, host-equality, choice, and ambient-completeness coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 4283.3 (Cauchy completion route real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyCompletionRoute}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BishopRegularCauchyCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes only dyadic tolerance rows, finite stream windows, regular rational readback, the Bishop regular-Cauchy completion route, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. It does not export quotient stream equality, selected limits, host real equality, countable choice, or an ambient completeness theorem for* $\mathsf{Real}^{\uparrow}$*.*

Proof. Begin with Definition 4283.2. It places every completion-facing read behind the displayed dyadic tolerance row D and stream-window row S.

The regularity route is then unique inside the carrier. The row Q is the only regular rational passage, and B confirms that the same D, S, Q, E rows are the rows consumed by the Bishop regular-Cauchy completion packet.

The terminal handoff is the row E. Transport by T, replay by C, provenance by P, and local naming by N preserve the route without adding new analytic data, so the $\mathsf{Real}^{\uparrow}$ consumer receives exactly the displayed finite boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionRoute}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite observations, RegSeqRat regularity, a Bishop regular-Cauchy completion row, a terminal Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$, seed)

Scope closed: Definition 4283.1, Definition 4283.2, and Definition 4283.3 seed the finite L10 completion route from dyadic observations to the Real boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionRoute}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic observation discipline, regularity routing, Bishop route compatibility, terminal seal discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionRouteUp.

Chapter 4284

A Concrete Naming Certificate for $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$

The $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ packet realizes the proof-search boundary in section 9386.4 as finite NameCert data. It sits beside the critical-path packet of chapter 765, the subject-reduction discharge socket of chapter 719, the normalization frontier of chapter 892, and the bounded normal-conversion bridge of chapter 3439. The packet names the exact frontier at which candidate evidence, bounded normal endpoints, discharge sockets, obstruction rows, transport, replay, provenance, and local naming may be consumed by a proof-search reader without claiming unrestricted closed consistency.

Falsifiable prediction: Any accepted proof-search consumer reaching a closed-consistency read without first exposing the discharge socket row refutes the $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ carrier.

Independence witness: metaciccriticalpath, subjectreductiondischargesocket, metacicnormalizationfrontier, boundednormalconversionbridge: the critical path records dependency pressure without local proof-search candidates; the discharge socket records subject-reduction inputs without bounded normal endpoints; the normalization frontier records candidate evidence without proof-search replay; and the bounded normal-conversion bridge records endpoint comparison without the discharge socket. None is bijective to $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$.

Definition 4284.1 (MetaCIC proof-search frontier carrier)**.** *A* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, D, N, B, A, E, O, T, C, P, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *row, retaining the closed-consistency dependency pressure over subject reduction, normalization, decidability, confluence, and examples.*
2. *D is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row, recording the socket through which proof-search evidence must pass before a closed-consistency consumer may read the subject-reduction boundary.*
3. *N is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row, exposing candidate evidence, closed-candidate evidence, finished-normal endpoints, and retained normalization obstruction data.*
4. *B is the* $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$ *row, recording the bounded normal endpoint bridge used by conversion-facing proof-search checks.*

5. *A is the candidate-evidence row, storing the finite proof-search candidates, their local support rows, and the bounded normal endpoint they claim to reach.*
6. *E is the discharge-socket evidence row, pairing each accepted candidate with the socket read needed before closed-consistency handoff.*
7. *O is the obstruction row, retaining the excluded endpoints: unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, and proof-search completeness.*
8. *T records componentwise* hsame *transport for K, D, N, B, A, E, O and the local naming row.*
9. *C records displayed* Cont *replay from proof-search candidate admission through the critical path, discharge socket, normalization frontier, bounded bridge, evidence, obstruction, provenance, and local naming rows.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, SubjectReductionDischargeSoc $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, D, N, B, A, E, O, T, C, P, L$. It has no coordinate for unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, proof-search completeness, or an unmediated closed-consistency read.

Lean4 ✓ `BEDC.Derived.MetaCICProofSearchFrontierUp.MetaCICProofSearchFrontierTasteGate single carrier alignment`

Theorem 4284.2 (MetaCIC proof-search frontier NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *carrier $F = (K, D, N, B, A, E, O, T, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontie}}$ *surface consists exactly of the following finite obligation rows:*

1. *critical-path exposure through K;*
2. *subject-reduction discharge-socket exposure through D;*
3. *normalization-frontier exposure through N;*
4. *bounded normal-conversion bridge exposure through B;*
5. *candidate-evidence admission through A;*
6. *discharge-socket evidence pairing through E;*
7. *obstruction retention through O;*
8. *componentwise transport through T;*
9. *displayed replay through C;*
10. *provenance through P and local naming through L.*

No accepted proof-search read may skip the discharge socket row, and no local obligation is stored outside $K, D, N, B, A, E, O, T, C, P, L$.

Proof. Project the carrier of Definition 4284.1. Rows K, D, N, B are the already named frontier surfaces, while A and E store the local candidate and discharge-socket evidence.

The closed-consistency-facing read must pass through D and E. Candidate evidence in A may reach a bounded normal endpoint through N and B, but it is not eligible for the consistency handoff until the discharge socket row has been exposed.

The obstruction row O remains part of the packet throughout the read. It prevents the bounded proof-search frontier from being cited as unrestricted subject reduction, unrestricted normalization, global conversion decidability, kernel consistency, or proof-search completeness.

Finally T, C, P, L transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded endpoints or for an unmediated closed-consistency read, the listed rows exhaust the NameCert surface. □

Theorem 4284.3 (MetaCIC proof-search frontier discharge boundary)**.** *Every closed-consistency-facing consumer of an accepted* $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{SubjectReductionDischargeSocket}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto \mathsf{BoundedNorm}$$

and then through the local candidate-evidence, discharge-socket evidence, obstruction, transport, replay, provenance, and NameCert rows. The route exports a bounded proof-search frontier only; it does not prove unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, or proof-search completeness. Future Lean work in TasteGate.lean *must provide* Nontrivial *and* FieldFaithful *instances before the carrier can be used as a checked proof-search taste gate.*

Proof. Start from Definition 4284.2. It forces the proof-search consumer to expose the critical path and the subject-reduction discharge socket before candidate evidence is treated as consistency-facing data.

The normalization and bounded conversion rows then bound the endpoint. The rows N and B justify only the displayed finished-normal and bounded-normal comparison reads, so the proof-search route remains a finite candidate route rather than a global theorem.

The local rows A, E, O, T, C, P, L preserve the discharge boundary. Candidate evidence is paired with the socket read, the obstruction row remains visible, and transport plus replay keep the consumer on the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICProofSearchFrontier}^{\uparrow}$ packet is generated from MetaCICCriticalPath, SubjectReductionDischargeSocket, MetaCICNormalizationFrontier, BoundedNormalConversionBridge, candidate evidence, discharge-socket evidence, obstruction, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4284.1, Definition 4284.2, and Definition 4284.3 seed the finite proof-search frontier for MetaCIC closed-consistency consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICProofSearchFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted subject reduction, unrestricted closed normalization, global conversion decidability, kernel consistency, proof-search completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked candidate admission, bounded-normal endpoint routing, discharge-socket exposure, obstruction retention, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICProofSearchFrontierUp.

Chapter 4285

A Concrete Naming Certificate for $\mathsf{DyadicIntervalHull}^{\uparrow}$

Following Moore interval analysis and Bishop–Bridges constructive interval arithmetic, the $\mathsf{DyadicIntervalHull}^{\uparrow}$ packet records the finite lower-upper dyadic enclosure route used before Real and Completion consumers inspect an interval hull. It sits next to the endpoint carrier of chapter 306, the interval-operation packet of chapter 1645, the located hull surface of chapter 2327, and the located endpoint route of chapter 394. The packet names a constructive enclosure certificate for displayed dyadic endpoints; it is not a host interval primitive, not a selected real endpoint pair, and not an appeal to ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4285.1 (Dyadic interval hull carrier)**.** *A* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (L, U, O, A, B, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *L is the lower dyadic endpoint row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. *U is the upper dyadic endpoint row accepted by* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *O is the endpoint-order witness, recording the displayed comparison $L \leq U$ through the same dyadic scale ledger.*
4. *A is the hull enclosure ledger, listing every inspected source endpoint or cell and its containment proof inside the displayed dyadic interval.*
5. *B is the* $\mathsf{DyadicInterval}^{\uparrow}$ *handoff row, packaging L, U, O, A as the dyadic interval consumed by interval-arithmetic neighbours.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which located endpoint observers may read finite endpoint approximations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same endpoint windows.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the dyadic interval row, located-real hull row, stream windows, and regular rational readbacks have been displayed.*
9. *T records componentwise* hsame *transport for endpoints, order witnesses, enclosure ledgers, located-real handoff rows, stream windows, readbacks, real seals, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from endpoint requests through L, U, O, A, B, S, R and then to E.*

11. *P is the* Pkg *provenance over* $\mathsf{DyadicIntervalHull}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRealIntervalHull}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}$ *row.*

The classifier compares accepted hulls only through these rows. It has no coordinate for host intervals, selected real endpoints, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.DyadicIntervalHullUp.DyadicIntervalHullTasteGate single carrier alignment`

Theorem 4285.2 (Dyadic interval hull NameCert obligations)**.** *For every accepted* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *carrier $H = (L, U, O, A, B, S, R, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}$ *surface consists exactly of carrier admission, endpoint ordering, hull containment, located-real route, non-escape, transport, replay, provenance, and local naming obligations. Carrier admission displays the lower endpoint, upper endpoint, order witness, enclosure ledger,* $\mathsf{DyadicInterval}^{\uparrow}$ *handoff,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local NameCert row. No obligation is stored outside those displayed rows.*

Proof. Project the accepted carrier. The lower and upper endpoint rows L, U are both dyadic rational core rows, and O is the only order witness available to the packet.

The enclosure ledger A is then read before any interval consumer sees the hull. It records finite containment of the displayed source endpoints or cells in the same lower-upper dyadic boundary.

The located-real route is the ordered handoff through B, S, R, E. The dyadic interval row packages the endpoint ledger, the stream row exposes finite endpoint windows, the regular rational row records readback, and the real row seals the terminal boundary.

Finally T, C, P, N preserve and name the route. Since the carrier has no coordinate for a host interval, selected endpoint pair, quotient stream, trichotomy, choice principle, or ambient completeness theorem, the listed obligations exhaust the local certificate. □

Theorem 4285.3 (Dyadic interval hull located-real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or Completion-facing consumer of an accepted* $\mathsf{DyadicIntervalHull}^{\uparrow}$ *packet factors through the route*

$$\mathsf{DyadicInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedRealIntervalHull}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSe}$$

as displayed by the rows $L, U, O, A, B, S, R, E, T, C, P, N$. The consumer may inspect a hull only after the dyadic endpoints, order witness, containment ledger, located interval handoff, stream windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start from Definition 4285.2. It places every accepted read behind L, U, O, A, so the consumer first sees a finite dyadic lower-upper enclosure.

The row B is the dyadic interval handoff. It is the only route from endpoint rows to interval consumers, and it keeps the hull attached to the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ comparisons.

The located-real portion then runs through the finite windows S and readbacks R before the seal E is exposed. This is exactly the located interval route consumed by the surrounding located-real hull and located-interval chapters.

Transport, replay, provenance, and local naming preserve the row order. Therefore no Real or Completion consumer can bypass the dyadic interval, dyadic core, located hull, located interval, stream, regular rational, and real-seal rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalHull}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoint rows, an endpoint-order witness, a hull enclosure ledger, DyadicInterval and located-real handoff rows, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4285.1, Definition 4285.2, and Definition 4285.3 seed the finite dyadic hull route from endpoint rows to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalHull}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval primitives, selected real endpoint pairs, quotient stream equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint ordering, hull containment, located-real routing, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.DyadicIntervalHullUp.

Chapter 4286

A Concrete Naming Certificate for $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$

The $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet realizes the carrier-switch audit surface of chapter 9391 as finite BEDC data. It sits beside the carrier encoding of chapter 943, the BHist cell-row embedding of chapter 948, the substrate classifier of chapter 945, the cellular window trace of chapter 3481, and the carrier comparison of chapter 3998. The packet names the finite cost of retargeting theorem families, kernel verbs, proof scripts, and audit gates to a cellular carrier; it is not a replacement kernel carrier.

Definition 4286.1 (Cellular carrier switch-cost carrier)**.** *A* $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, O, V, R, A, K, H, C, P, N)$$

with the following displayed rows:

1. *B is the BHist theorem-family row, listing the finite family of BHist-facing statements under inspection.*
2. *O is the cellular orbit row, exposing the finite cellular rows and orbit windows to which the statements would be retargeted.*
3. *V is the verb-realignment row, recording how Mark, Hist, Ext, and Cont reads are translated to cellular classifier, orbit, extension, and continuation reads.*
4. *R is the proof-retarget budget row, recording the displayed proof obligations that must be rebuilt after the carrier change.*
5. *A is the audit-regeneration ledger, listing the checks that must be regenerated for the cellular statements and their NameCert markers.*
6. *K is the non-escape row, recording that the packet measures the switch cost without replacing the public BEDC* BHist *carrier.*
7. *H records componentwise* hsame *transport for theorem-family, orbit, verb, proof-budget, audit, non-escape, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through B, O, V, R, A, K before any switch-cost read reaches the local certificate.*

9. *P is the* Pkg *provenance over* CellularCarrierSwitchCost$^{\uparrow}$, CellularCarrierEncoding$^{\uparrow}$, BHistCellRowEmbedding$^{\uparrow}$, CellularSubstrateClassifier$^{\uparrow}$, CellularWindowTrace$^{\uparrow}$, CellularVerbRealignment$^{\uparrow}$, CellularCarrierBHistEquivalence$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{CellularCarrierSwitchCost}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, V, R, A, K, H, C, P, N$. It has no coordinate for a kernel-carrier replacement, infinite cellular semantics, host evaluator equivalence, hidden proof synthesis, or an audit gate outside the displayed ledger.

Lean4 ✓ `BEDC.Derived.CellularCarrierSwitchCostUp.CellularCarrierSwitchCostTasteGate single carrier alignment`

Theorem 4286.2 (Cellular carrier switch-cost NameCert obligations)**.** *For every accepted* CellularCarrierSwitchCost *carrier $S = (B, O, V, R, A, K, H, C, P, N)$, the local* NameCert$_{\text{CellularCarrierSwitchCost}^{\uparrow}}$ *surface consists exactly of carrier admission, BHist theorem-family exposure, cellular orbit exposure, verb realignment, proof-retarget budgeting, audit-regeneration accounting, non-escape, transport, replay, provenance, and local naming obligations. A valid switch-cost read must display both the proof-retarget budget row R and the audit-regeneration ledger A before it reaches the local certificate.*

Proof. Carrier admission is projection from Definition 4286.1. The rows B and O expose the two carriers being compared as finite theorem-family and orbit-window data.

The verb row V records the syntactic realignment of Mark, Hist, Ext, and Cont. It does not itself measure the engineering cost, because statement retargeting and proof rebuilding are separate rows.

The proof-retarget budget R and audit-regeneration ledger A are therefore mandatory. A packet omitting either row cannot certify the displayed switch cost, since it would hide either the proof obligations or the audit gates that must be regenerated.

Finally K, H, C, P, N preserve the non-replacement boundary, replay the finite route, record provenance, and name the packet. No hidden carrier replacement, infinite orbit semantics, host evaluator equivalence, or external audit gate is available outside those rows. □

Theorem 4286.3 (Cellular carrier switch-cost finite audit route)**.** *Every accepted* CellularCarrierSwitchCost$^{\uparrow}$ *read factors through the finite route*

$$B, O, V, R, A, K, H, C, P, N.$$

The route measures a BHist theorem-family transposition by first displaying the theorem family, then the cellular orbit window, then the verb-realignment row, then the proof-retarget budget, and then the audit-regeneration ledger. It cannot export carrier replacement, host bisimulation, unbounded cellular traces, hidden proof repair, or completed audit success outside the displayed packet.

Proof. Read B first to identify the theorem-family surface whose statements are being retargeted. Read O next to expose the finite cellular orbit window available to the target carrier.

The row V aligns the verbs but does not discharge the work. It only states which BHist reads correspond to which cellular classifier and orbit reads.

The rows R and A then carry the cost. Proof retargeting is visible in R, while audit regeneration is visible in A, so the packet measures both proof work and gate work before local naming.

The non-escape and structural rows K, H, C, P, N keep this as a finite audit NameCert packet. Because no other coordinate exists, the route cannot export a carrier replacement, host bisimulation, unbounded trace, hidden proof repair, or completed audit success. □

Falsifiable prediction: A valid $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet is refuted if it omits the proof-retarget budget row or the audit-regeneration ledger.

Independence witness: cellularcarrierencoding, bhist-cell-row-embedding, cellularsubstrateclassifier, cellular-window-trace: $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries BHist theorem-family, cellular orbit, verb-realignment, proof-retarget, audit-regeneration, non-escape, transport, replay, provenance, and local naming rows.

Lean TasteGate coverage for this packet must provide Nontrivial and FieldFaithful instances in BEDC.Derived.CellularCarrierSwitchCostUp.TasteGate.

Closure status for $\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}$.

Constructive story: A $\mathsf{CellularCarrierSwitchCost}^{\uparrow}$ packet is generated from BHist theorem-family rows, cellular orbit rows, verb-realignment rows, proof-retarget budget rows, audit-regeneration ledgers, non-escape rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4286.1, Definition 4286.2, and Definition 4286.3 seed the finite audit object for cellular carrier-switch cost.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CellularCarrierSwitchCost}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close kernel-carrier replacement, infinite cellular semantics, host evaluator equivalence, hidden proof synthesis, completed audit success, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-family exposure, cellular orbit exposure, verb realignment, proof-retarget budgeting, audit-regeneration accounting, non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CellularCarrierSwitchCostUp.

The sequence-level shared-modulus route is a downstream consumer of chapter 2299 and chapter 4356 before finite $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows are exposed.

Chapter 4287

A Concrete Naming Certificate for $\mathsf{UniformlyContinuousSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet records a sequence of maps whose continuity data share one displayed modulus ledger. It sits over the metric source row of chapter 57, the continuous-map surface of chapter 130, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The point of the packet is the shared-modulus route: a downstream consumer may inspect finitely many sequence entries only after the same uniform-control row has been displayed for all inspected maps.

The sequence carrier consumes the continuous-map rows of chapter 130 and the uniform-modulus handoff rows of chapter 369 before it exposes its own sequence-level carrier, so each inspected entry inherits a map row and the shared modulus remains a visible sibling dependency rather than a detached continuity family.

The $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ route first consumes the single-modulus surface of chapter 391 and the exported precision ledger of chapter 369; only after those two sibling rows are displayed can the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows be read as sequence-level data.

The compact-source read uses the route through chapter 2299, and the equicontinuous sequence comparison uses chapter 4356 before the shared-modulus row can be consumed.

The shared-modulus route is also a downstream consumer of chapter 2299 and chapter 4356: compact finite-net moduli and equicontinuity family rows supply the same tolerance-only handoff that $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ exposes to $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ windows.

Definition 4287.1 (Uniformly continuous sequence carrier)**.** *A* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, F, M, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row used by every displayed map in the sequence.*
3. *F is the finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence-entry rows available to the packet.*
4. *M is the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, read once and then reused by every displayed entry in F.*

5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row listing the inspected indices, source probes, and requested tolerances.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and to the shared modulus ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal for the target readings reached only after S, Q, M have been displayed.*
8. *H records componentwise* hsame *transport for source metric, target metric, sequence-entry, shared-modulus, stream-window, regular-readback, real-seal, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay from an inspected sequence index and tolerance request through X, Y, F, M, S, Q, E.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, F, M, S, Q, E, H, C, P, N$. It has no coordinate for host function extensionality, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached pointwise continuity proof for a sequence entry.*

Theorem 4287.2 (Uniformly continuous sequence NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier $U = (X, Y, F, M, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequ}}$ *surface consists exactly of carrier admission for the two metric rows, continuous-map sequence-entry rows, shared uniform-modulus row, finite stream windows, regular rational readback, real-name seal, componentwise transport, continuation replay, package provenance, and local naming; shared-modulus exactness requiring each inspected entry of F to consume the same M row; finite-window exactness through S; readback exactness through Q; real-seal boundary through E; and non-escape from host function extensionality, quotient streams, selected limits, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, and detached pointwise continuity data.*

Proof. Project the carrier of Definition 4287.1. The only source, target, sequence-entry, shared-modulus, stream, readback, and real-seal data available to the classifier are the displayed rows X, Y, F, M, S, Q, E.

The shared-modulus clause is read before any inspected sequence entry reaches a downstream Real-facing consumer. Each entry in F is a $\mathsf{ContinuousMap}^{\uparrow}$ row, but the packet exposes it to a tolerance request only through the common $\mathsf{UniformModulus}^{\uparrow}$ row M and the finite windows in S.

Readback and sealing are downstream of that shared route. The $\mathsf{RegSeqRat}^{\uparrow}$ row Q records only the rational data obtained from the displayed windows and shared modulus, and the $\mathsf{Real}^{\uparrow}$ row E is reached only after S, Q, M have been exposed.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for host function extensionality, quotient stream equality, selected limits, countable choice, ambient real completeness, or detached pointwise continuity proofs, the listed rows exhaust the local NameCert surface. □

Theorem 4287.3 (Uniformly continuous sequence shared-modulus handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

The route supplies a shared modulus for all inspected sequence entries, finite stream windows for those inspections, regular rational readback, and the terminal real-name seal. It exports no host

function extensionality, quotient stream equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden complete metric space.*

Proof. Start from the obligation surface of Definition 4287.2. It identifies the continuous-map entries and the shared uniform-modulus ledger as carrier rows, not as host functions or external continuity facts.

A Real-facing consumer next opens only the finite windows in S. Those windows determine the regular rational readback Q under the shared modulus M, and only then can the terminal seal E be consumed.

The transport, continuation, provenance, and local naming rows preserve that same order. Therefore every public read is a shared-modulus sequence-of-maps route through the displayed BEDC rows, with the excluded extensionality, quotient, choice, completeness, and hidden-completion objects absent from the packet. □

Theorem 4287.4 (Uniformly continuous sequence obligation-closure package)**.** *The carrier of Definition 4287.1, the local NameCert obligations of Definition 4287.2, the shared-modulus handoff of Definition 4287.3, and the compact/uniform-modulus child routes assemble one obligation surface for* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$*. A consumer may read source data through* $\mathsf{CompactMetric}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *rows, sequence entries through* $\mathsf{ContinuousMap}^{\uparrow}$*, uniform control through* $\mathsf{UniformModulus}^{\uparrow}$*, inspected indices through* $\mathsf{StreamName}^{\uparrow}$ *windows, rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, terminal sealing through* $\mathsf{Real}^{\uparrow}$*, and structural movement through* hsame*,* Cont*,* Pkg*, and* NameCert*.*

This package exports no host function extensionality, quotient stream equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, hidden complete metric space, or detached pointwise-continuity proof.*

Proof. Start from the carrier row $U = (X, Y, F, M, S, Q, E, H, C, P, N)$. The source and target metric rows, the finite sequence-entry row, the shared modulus row, the stream-window row, the regular rational readback row, and the real-seal row are the only analytic data visible to the packet.

Apply the NameCert obligations. They assign carrier admission, shared-modulus exactness, finite-window exactness, readback exactness, real-seal boundary, transport, replay, provenance, and local naming to the displayed rows.

Use the shared-modulus handoff together with the compact-source and uniform-modulus routes named in the chapter introduction. Those routes force every inspected $\mathsf{ContinuousMap}^{\uparrow}$ entry to consume the same $\mathsf{UniformModulus}^{\uparrow}$ row before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows can be read.

Therefore the obligation package is exactly the finite BEDC surface listed in the statement. Since no carrier coordinate names host extensionality, quotient streams, selected limits, countable choice, ambient completeness, hidden complete metrics, or detached pointwise proofs, none of those endpoints is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet is generated from source and target MetricSpace rows, ContinuousMap sequence-entry rows, one shared UniformModulus row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$, seed)

Scope closed: Definition 4287.1, Definition 4287.2, and Definition 4287.3 seed the shared-modulus sequence-of-maps packet for Real-facing finite-window consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformlyContinuousSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient streams, selected limits, countable choice, ambient RealUp completeness, hidden complete metric spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-modulus exactness, finite-window exactness, regular-readback boundary, Real-seal boundary, transport, replay, provenance, and local naming for BEDC.Derived.UniformlyContinuousSe

The UniformlyContinuousSequence$^{\uparrow}$ route consumes chapter 2299, chapter 369, and chapter 130 as load-bearing rows before any sequence endpoint is public. The compact-uniform-modulus sibling supplies the finite compact-window fold, the uniform-modulus sibling supplies the shared tolerance-to-precision row, and the continuous-map sibling supplies the pointwise graph and modulus readback that the sequence packet transports along hsame and replays through Cont.

Thus a sequence-level uniform-continuity packet reads compact cover data, finite modulus rows, tolerance readbacks, and terminal Real-facing seals through the sibling modulus lattice rather than through a host continuity theorem or an external compactness oracle.

Theorem 4287.5 (Compact uniform-modulus route for uniformly continuous sequences)**.** *Every Real-facing read of an accepted* UniformlyContinuousSequence$^{\uparrow}$ *carrier whose source is displayed through the compact route must consume the shared* UniformModulus$^{\uparrow}$ *row through the compact-uniform modulus packet of chapter 2299. The read factors through the visible route*

$$\mathsf{CompactUniformModulus}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hs}$$

It exports no detached pointwise continuity family, host compactness theorem, selected limit, quotient stream equality, countable-choice schedule, or ambient Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4287.1. The sequence entries are available only as ContinuousMap$^{\uparrow}$ rows, the inspected indices only through StreamName$^{\uparrow}$ windows, and the target values only after RegSeqRat$^{\uparrow}$ readback and the terminal Real$^{\uparrow}$ seal.

When the source row is compact, the shared modulus row is not read as an external theorem. It is consumed through the compact-uniform modulus route of chapter 2299, which packages compact source coverage and the downstream UniformModulus$^{\uparrow}$ handoff before any Real-facing sequence read is public.

The transport, replay, provenance, and local naming rows then preserve the same route. Since the carrier has no coordinate for detached pointwise continuity, host compactness, selected limits, quotient stream equality, countable choice, or ambient real completeness, a Real-facing compact-source read must pass through the displayed compact-uniform modulus dependency. □

Theorem 4287.6 (Uniformly continuous sequence compact-metric forward dependency)**.** *Every* CompactMetric$^{\uparrow}$ *consumer of an accepted* UniformlyContinuousSequence$^{\uparrow}$ *packet must first read the visible source and observation route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Only after the metric source row, continuous-map entry rows, shared uniform-modulus row, finite stream windows, regular rational readback, and terminal real seal are displayed may the consumer inspect compact-metric modulus data.

Proof. Project the carrier of Definition 4287.1. The compact-facing consumer receives the source metric row X, target metric row Y, continuous-map entry rows F, shared modulus row M, stream-window row S, readback row Q, and real-seal row E before any structural transport or provenance row can be used.

The compact-metric dependency route then reads the compact source through chapter 129. That route is downstream of the displayed Met$^{\uparrow}$ source and the same UniformModulus$^{\uparrow}$ row used

by the sequence packet, so compact-metric modulus data cannot be consumed before the shared sequence modulus and finite windows are visible.

Finally the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows seal the consumer-facing values. Since the carrier has no coordinate for detached pointwise continuity, selected limits, quotient stream equality, countable choice, or ambient real completeness, the compact-metric consumer is forced through the displayed forward dependency. □

The compact-metric consumer reads $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ through chapter 129 and chapter 369 before using the shared sequence modulus. This keeps the sequence ledger attached to compact finite-net data and the finite modulus row, rather than to an external pointwise continuity family, open-cover compactness theorem, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim.

Theorem 4287.7 (UniformlyContinuousSequence equicontinuous dependency)**.** *Every accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *shared-modulus packet supplies the common modulus row consumed by* $\mathsf{EquicontinuousSequence}^{\uparrow}$*. The dependency is read through* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{CompactMetric}^{\uparrow}$ *source rows of chapter 129, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, and the* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before the equicontinuity consumer of chapter 4356 may inspect the shared sequence modulus.*

Proof. Start with the carrier of Definition 4287.1. Its finite sequence route displays the source and target metric rows, continuous-map entries, one shared $\mathsf{UniformModulus}^{\uparrow}$ row, finite $\mathsf{StreamName}^{\uparrow}$ windows, regular rational readback, terminal $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming.

The compact-metric dependency route keeps the same shared modulus attached to compact source rows before downstream compact-uniform consumers read it. Thus an $\mathsf{EquicontinuousSequence}^{\uparrow}$ consumer receives a common modulus only after the compact source, stream window, rational readback, and real seal rows have been displayed.

Finally use the equicontinuous-sequence carrier of Definition 4356.1. Its common schedule and equicontinuity rows consume exactly the kind of shared finite modulus exported by the $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ packet, while selected subsequences, quotient stream equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness remain outside both carriers. □

Theorem 4287.8 (Uniformly continuous sequence equicontinuity handoff)**.** *Every accepted* UniformlyContinuousSe *packet read over a displayed finite sequence window may be consumed as a finite family-modulus route for chapter 4076. The read first exposes the shared* $\mathsf{UniformModulus}^{\uparrow}$ *row, the finite* $\mathsf{StreamName}^{\uparrow}$ *window, the compact-metric source boundary, regular rational readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row; only then may an* $\mathsf{Equicontinuity}^{\uparrow}$ *consumer inspect the same window as a finite family indexed by the displayed sequence positions.*

Proof. Project the uniformly continuous sequence carrier through Definition 4287.1. Its shared modulus is already attached to the displayed sequence window, so the finite family read does not choose a new radius schedule.

Read the equicontinuity carrier through Definition 4076.1. That carrier asks for a finite family ledger, a common tolerance-radius handoff, transport, replay, provenance, and local naming. The sequence window supplies exactly that route when its displayed indices are treated as the family index set.

Thus the dependency is a forward consumer edge from the sequence packet to $\mathsf{Equicontinuity}^{\uparrow}$. It does not add selected subsequences, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4287.9 (Uniformly continuous sequence located-modulus dependency route)**.** *Every* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *read that reaches a sequence-level* $\mathsf{Real}^{\uparrow}$ *consumer must first expose a substantive modulus sibling: either the* $\mathsf{UniformModulus}^{\uparrow}$ *route of chapter 369 or the compact handoff through* $\mathsf{CompactUniformModulus}^{\uparrow}$ *in chapter 2299. The sequence packet therefore exports a finite modulus-mediated read surface before any Real-facing sequence value is consumed, and it provides no detached pointwise-continuity family, selected limit, quotient stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted uniformly continuous sequence packet. Its sequence rows can be inspected only through displayed finite windows, and a Real-facing consumer can read values only after the regular rational and terminal Real rows have been exposed.

The modulus row supplies the horizontal dependency. A general sequence read consumes $\mathsf{UniformModulus}^{\uparrow}$ through chapter 369; a compact-source read consumes the compact route through chapter 2299 before it presents the same uniform-modulus handoff.

Thus the sequence carrier is not a kernel-only island. Its downstream Real route is mediated by a displayed modulus sibling, so the packet cannot supply a detached pointwise family, selected limit, quotient stream equality, countable-choice schedule, or ambient real completeness. □

Theorem 4287.10 (Uniformly continuous sequence pointwise stream metric route)**.** *Every accepted* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *packet whose finite windows are consumed pointwise by a metric-stream reader must expose the same source and target* $\mathsf{Met}^{\uparrow}$ *rows, shared* $\mathsf{UniformModulus}^{\uparrow}$ *row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before it can invoke the* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *comparison route of chapter 4139. The read is a shared-modulus stream route, not detached pointwise continuity, quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4287.1. The pointwise metric-stream reader receives only $X, Y, F, M, S, Q, E, H, C, P, N$, so the source and target metric rows and the shared uniform-modulus row remain visible before any inspected sequence entry is read.

Use the NameCert obligations and shared-modulus handoff. They force each finite window to pass through the same $\mathsf{UniformModulus}^{\uparrow}$ row M, then through $\mathsf{StreamName}^{\uparrow}$ windows S and $\mathsf{RegSeqRat}^{\uparrow}$ readback Q before the $\mathsf{Real}^{\uparrow}$ seal E can be consumed.

Finally route the visible windows through chapter 4139. The $\mathsf{PointwiseStreamMetric}^{\uparrow}$ comparison can inspect the displayed metric-stream rows, but the uniformly continuous sequence packet has no coordinate for detached pointwise continuity proofs, quotient stream equality, selected limits, countable-choice schedules, or ambient Real completeness. □

The $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ completion-facing route reads the regular-Cauchy metric boundary through chapter 8164 and the compact uniform-modulus route of chapter 2299 before any sequence-level Real seal is public. This keeps compact finite-net data, pointwise-stream observations, shared $\mathsf{UniformModulus}^{\uparrow}$ rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seals on one displayed sibling route rather than on a detached host continuity theorem, selected limit, quotient stream equality, or ambient completion principle.

Theorem 4287.11 (Uniformly continuous sequence sibling dependency)**.** *Every* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *consumer first reads the uniform-modulus surface of chapter 391 and the compact-uniform handoff of chapter 388 before it routes pointwise stream metrics into sequence-level uniformity. Thus the sibling surface is a finite lattice of uniform-control rows, compact handoff rows, stream windows,*

transport, replay, provenance, and local naming, not detached pointwise continuity, selected limits, countable choice, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the route through the two sibling chapters named in the statement. The uniform-continuity row supplies the modulus that every sequence-level consumer must keep visible, and the compact-uniform row supplies the handoff that turns compact-domain control into a stable sequence surface.

The remaining rows are bookkeeping for that route. Stream windows present the finite observations, hsame transport preserves the selected control data, Cont replay replays the same rows, Pkg records provenance, and the local naming row records the resulting $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ certificate. No extra pointwise or quotient-level equality is introduced. □

Every public sequence-level uniform-continuity read from $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ must first pass through the sibling lattice formed by compact finite-net rows from chapter 129, the shared $\mathsf{UniformModulus}^{\uparrow}$ row of chapter 369, pointwise stream-metric rows from chapter 4139, $\mathsf{StreamName}^{\uparrow}$ window rows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seals. The lattice gives the consumer finite compact coverage, one reusable modulus, stream-local metric observations, windowed rational reads, and sealed target values before any sequence-level uniform continuity claim is public; it exports no detached pointwise-continuity family, selected limit, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Every $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ consumer that reads sequence-level uniform continuity must pass through the finite modulus surface of chapter 2228 or chapter 5128 before it can inspect the shared sequence ledger. Completion-facing uses then continue through the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ readback route of chapter 38, so the packet exposes a finite uniform-modulus dependency rather than a host pointwise-continuity family, a selected limit, or an ambient completeness principle.

Chapter 4288

A Concrete Naming Certificate for $\mathsf{LocatedIntervalHalving}^{\uparrow}$

Following Bishop and Bridges constructive interval-halving analysis, the $\mathsf{LocatedIntervalHalving}^{\uparrow}$ packet isolates the one-step finite halving certificate that later bisection and convergence chapters consume. It reads the midpoint route of chapter 1712, the retained-child route of chapter 1711, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the downstream convergence consumer of chapter 1804. The certificate names one visible halving step, not an infinite branch, a selected limit, host interval equality, or ambient completeness.

Definition 4288.1 (Located interval halving carrier)**.** *A* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, M, L, R, S, D, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *I is the source located real interval row. It supplies the lower endpoint, upper endpoint, radius budget, locator reads, locatedness modulus, and endpoint-order observations for the parent interval.*
2. *M is the* $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$ *row. It records the displayed midpoint route and the two child endpoint-transport rows before any retained side is selected.*
3. *L and R are the left-child and right-child interval rows opened from M.*
4. *S is the side row. It records the finite locator, sign, containment, or requested-side datum that selects which of L, R is retained for this halving step.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records the common scale, half-radius budget, and dyadic comparison data used by the one-step diameter estimate.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the retained child endpoints and midpoint windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, M, L, R, S, D, Q have been displayed.*
8. *T records componentwise* hsame *transport for the source interval, midpoint row, child intervals, side row, dyadic tolerance row, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a halving request through source inspection, midpoint construction, child exposure, side retention, dyadic tolerance checking, regular readback, and the Real-facing seal.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalHalving}^{\uparrow}$, $\mathsf{LocatedIntervalMidpoint}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one located interval halving packet. It has no coordinate for host interval equality, real trichotomy, quotient representatives, countable choice, an infinite branch, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalHalvingUp.LocatedIntervalHalvingTasteGate single carrier alignment`

Theorem 4288.2 (LocatedIntervalHalving NameCert obligations)**.** *For every accepted* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier*

$$H = (I, M, L, R, S, D, Q, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source located interval, midpoint row, two child intervals, side row, dyadic tolerance row, regular readback, Real seal, transport, replay, provenance, and local naming;*

2. *source discipline, requiring every endpoint, radius, locator, locatedness, and order read to factor through I before midpoint or side data are inspected;*

3. *midpoint exposure, requiring M to display the midpoint and both child endpoint-transport routes before the retained side is read;*

4. *side retention, requiring S to choose only between the displayed children L and R and to export no infinite branch selector;*

5. *dyadic halving exactness, requiring D to attach the retained child to the common scale and half-radius budget used by the one-step diameter estimate;*

6. *Real-facing handoff, requiring Q, E, T, C, P, N to route the retained child through regular rational readback, the terminal Real seal, transport, replay, provenance, and local naming;*

7. *ledger exhaustion, excluding host interval equality, real trichotomy, quotient representatives, countable choice, infinite branches, selected limits, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge data.*

No located interval halving obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 4288.1. The displayed tuple exposes the source interval, midpoint route, two child intervals, side row, dyadic tolerance row, regular readback, Real seal, and structural rows as finite BHist coordinates.

Source discipline is supplied by I. The halving request starts from the source located interval row, so endpoint, radius, locator, locatedness, and order observations cannot be replaced by a host closed interval object.

Midpoint exposure is supplied by M, L, R. The midpoint row displays the two child endpoint-transport routes, and the child interval rows make those two finite cells available before any retained side is read.

Side retention and dyadic exactness are supplied by S and D. The side row selects only between the displayed children, while the dyadic row records the common scale and half-radius budget for that retained child.

Finally Q, E, T, C, P, N carry the retained child to regular rational readback, the terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. Since every visible coordinate has already been listed, the excluded host equality, trichotomy, quotient, choice, branch, limit, completeness, verification, and bridge objects cannot be projected. □

Theorem 4288.3 (LocatedIntervalHalving RegSeqRat and Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedIntervalHalving}^{\uparrow}$ *carrier reads the retained child only after the finite route*

$$I \longmapsto M, L, R \longmapsto S, D \longmapsto Q \longmapsto E$$

has been displayed. Consequently the handoff consumed by chapter 1711 and chapter 1804 is a one-step located halving certificate over $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows, not a selected branch, a quotient representative, or an ambient completeness principle.*

Proof. Start with the accepted carrier $H = (I, M, L, R, S, D, Q, E, T, C, P, N)$. The only source coordinate is I, so a readback or Real-facing consumer first receives the displayed located interval data.

Next the consumer reads M, L, R. These rows display the midpoint and both child intervals before the retained child is selected, so the later route cannot hide a midpoint construction or child endpoint transport.

The retained side is then fixed by S and measured by D. The side row chooses between the displayed children, and the dyadic tolerance row records the common-scale half-radius estimate for that child.

Only after those rows are displayed does Q expose the regular rational readback and E expose the terminal Real seal. The structural rows T, C, P, N replay, transport, record provenance, and name the same finite route. Thus downstream bisection and convergence consumers receive a visible one-step halving certificate and no selected infinite branch, quotient representative, or ambient completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4288.1, Definition 4288.2, and Definition 4288.3 seed the one-step located interval halving route through source interval, midpoint, child interval, side-retention, dyadic tolerance, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedIntervalHalving}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, real trichotomy, quotient representatives, countable choice, infinite branches, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, midpoint exposure, side retention, dyadic halving exactness, RegSeqRat and Real handoff, and non-escape proofs for BEDC.Derived.LocatedIntervalHalvingUp.

Constructive story: A $\mathsf{LocatedIntervalHalving}^{\uparrow}$ packet is generated from one source located interval row, one LocatedIntervalMidpointUp row, two child interval rows, one finite side-retention row, one DyadicRatCoreUp tolerance row, one RegSeqRatUp readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4289

A Concrete Naming Certificate for $\mathsf{BishopLocatedRoot}^{\uparrow}$

Following Bishop and Bridges constructive located-root arguments, the $\mathsf{BishopLocatedRoot}^{\uparrow}$ packet records the finite bisection route by which a located sign-changing bracket is read as a terminal $\mathsf{Real}^{\uparrow}$ root seal. It sits after the bisection convergence surface of chapter 1804, the located intermediate-value route of chapter 1402, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a constructive root certificate over displayed finite rows; it does not assert full $\mathsf{Real}^{\uparrow}$ completeness, classical trichotomy, quotient real equality, selected host roots, or countable choice.

Definition 4289.1 (Bishop located root carrier)**.** *A* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (B, O, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the located bracket row naming two displayed endpoint windows and the retained interval boundary.*
2. *O is the opposite-sign witness row, read as positive apartness data for the two endpoint evaluations.*
3. *S is the bisection schedule row selecting the retained subintervals and midpoint tests.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording widths, midpoint radii, sign-test budgets, and the requested terminal precision.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the finite bisection windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, O, S, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for the bracket, sign witness, bisection schedule, dyadic ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a root request through $B, O, S, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRoot}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their replay path. It has no coordinate for a host root object, real trichotomy, quotient real equality, ambient completeness, countable choice, or a detached limit point.

Lean4 ✓ `BEDC.Derived.BishopLocatedRootUp.TasteGate.BishopLocatedRootTasteGate single carrier alignment`

Theorem 4289.2 (Bishop located root NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *carrier*

$$R = (B, O, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$ *surface consists exactly of carrier admission for the located bracket, opposite-sign witness, bisection schedule, dyadic tolerance ledger, regular rational readback, terminal Real seal, componentwise transport, displayed continuation replay, provenance, and local naming row; bisection-route exposure; Real-seal nonescape; and refusal of host roots, trichotomy, quotient real equality, ambient completeness, countable choice, and detached limit points.*

Proof. Project the carrier of Definition 4289.1. The accepted packet exposes exactly the bracket, sign witness, schedule, dyadic ledger, regular readback, Real seal, and structural rows H, C, P, N.

The bisection route is forced by the displayed order. A root-facing consumer must inspect the located bracket and opposite-sign row before it can read midpoint choices, dyadic widths, or the regular readback.

The terminal Real seal is downstream of those rows. Since the carrier has no coordinate for a host root, trichotomy, quotient equality, ambient completeness, countable choice, or a detached limit, the listed rows exhaust the local NameCert surface. □

Theorem 4289.3 (Bishop located root bisection route)**.** *Every accepted* $\mathsf{BishopLocatedRoot}^{\uparrow}$ *root read factors through the finite route*

$$B \longmapsto O \longmapsto S \longmapsto D \longmapsto Q \longmapsto E$$

before any downstream Real or Completion consumer can inspect the root seal. In particular, the read uses the bisection convergence route of chapter 1804, the finite-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ *seal of chapter 38.*

Proof. Start with an accepted root read. The located bracket row B and opposite-sign witness O are the only sources for the retained interval and sign-change data.

Apply the bisection schedule row S and dyadic ledger D. These rows record the finite midpoint choices, interval widths, and tolerance budgets consumed by the bisection convergence route.

Read Q and E only after the finite schedule and ledger have been displayed. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular rational source, and the $\mathsf{Real}^{\uparrow}$ seal records the terminal root-facing name. The structural rows H, C, P, N transport, replay, source, and name the same packet, so no host root, trichotomy, quotient equality, countable choice, or ambient completeness coordinate can enter between the bracket and the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRoot}^{\uparrow}$ packet is generated from a located bracket, opposite-sign witness, bisection schedule, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRoot}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4289.1, Definition 4289.2, and Definition 4289.3 seed the finite located-root certificate over bisection, StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopLocatedRoot}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, classical trichotomy, quotient real equality, selected host roots, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, opposite-sign stability, bisection-route exposure, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRootUp.

Chapter 4290

A Concrete Naming Certificate for $\mathsf{RealCompletenessWitness}^{\uparrow}$

The $\mathsf{RealCompletenessWitness}^{\uparrow}$ packet realizes the four-face Real and Completion route as finite BEDC data: the $\mathsf{StreamName}^{\uparrow}$ finite-window face, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance face, the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier face, and the terminal $\mathsf{Real}^{\uparrow}$ seal face. It records the completion witness used by Real-facing consumers without turning that witness into ambient completeness, quotient stream equality, selected limits, trichotomy, or countable choice.

Definition 4290.1 (Real completeness witness carrier)**.** *A* $\mathsf{RealCompletenessWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the requested observations.*
2. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger of endpoint rows, radii, comparison distances, and tolerance budgets.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity and classifier row read from the same finite schedule and dyadic ledger.*
4. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* S, D, R *have been displayed.*
5. H *records componentwise* hsame *transport for the four faces, replay, provenance, and local naming rows.*
6. C *records displayed* Cont *replay from a completion-facing request through* S, D, R, E, H, P, N.
7. P *is the* Pkg *provenance over* $\mathsf{RealCompletenessWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ *row.*

The classifier compares accepted witnesses only through the displayed four-face route and the structural rows. It has no coordinate for quotient streams, selected host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, countable choice, field completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealCompletenessWitnessUp.RealCompletenessWitnessTasteGate single carrier alignment`

Theorem 4290.2 (Real completeness witness NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWitness}$ *carrier* $W = (S, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity and classifier face, terminal RealUp seal face, componentwise transport, displayed continuation replay, provenance, and local naming; four-face route exposure; and refusal of quotient streams, selected limits, ambient completeness, trichotomy, countable choice, detached field structure, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 4290.1. The accepted packet has exactly four mathematical faces, followed by transport, replay, provenance, and naming rows.

The route exposure is ordered. A completion-facing consumer reads S before dyadic tolerances, D before regular readback, R before the terminal Real seal, and E before any downstream seal consumer.

The exclusions are an inventory check. No coordinate stores quotient stream equality, a selected host limit, an ambient completeness theorem, trichotomy, countable choice, a field-faithfulness proof, Lean verification, or a bridge certificate. Therefore the displayed rows exhaust the local NameCert surface. □

Theorem 4290.3 (Real completeness witness four-face route)**.** *Every accepted* $\mathsf{RealCompletenessWitness}^{\uparrow}$ *consumer reaches completion-facing Real data only through the finite route*

$$S \longmapsto D \longmapsto R \longmapsto E$$

and then through H, C, P, N*. Thus the witness is a finite four-face packet over chapter 126, chapter 291, chapter 301, and chapter 38, not an assertion of full* $\mathsf{Real}^{\uparrow}$ *completeness or a field-completeness theorem.*

Proof. Start with an accepted completion-facing read. The only source face is S, the finite $\mathsf{StreamName}^{\uparrow}$ schedule, so the read first fixes the visible windows.

Next read D and R. The dyadic face supplies endpoint, radius, distance, and tolerance rows; the regular-rational face reads those rows as the $\mathsf{RegSeqRat}^{\uparrow}$ regularity and classifier surface.

Finally read E as the terminal $\mathsf{Real}^{\uparrow}$ seal and pass through H, C, P, N for transport, replay, source, and local naming. Since no field-faithfulness, nontriviality, quotient stream, selected limit, or ambient completeness coordinate is present in the carrier, those later Lean obligations must be supplied separately by ‘TasteGate.lean‘, with ‘Nontrivial RealCompletenessWitnessUp‘ and ‘FieldFaithful RealCompletenessWitnessUp‘ recorded as independent evidence rather than projected from this packet. □

Falsifiable prediction: Any accepted $\mathsf{RealCompletenessWitness}^{\uparrow}$ read that reaches Real-completion data while bypassing the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ seal rows refutes the carrier.

Independence witness: streamname, dyadicratcore, regseqrat, real, finite-real-section: $\mathsf{RealCompletenessWitness}$ is not bijective to any listed sibling because it jointly carries the finite-window schedule, dyadic tolerance ledger, regularity and classifier readback, terminal Real seal, structural transport, replay, provenance, and local NameCert rows as one four-face packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWitness}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RealCompletenessWitness}^{\uparrow}$ packet is generated from a StreamName finite-window face, DyadicRatCore tolerance face, RegSeqRat regularity and classifier face, terminal RealUp seal face, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RealCompletenessWitness}\uparrow}$, seed)

Scope closed: Definition 4290.1, Definition 4290.2, and Definition 4290.3 seed the finite four-face completion witness over StreamName, DyadicRatCore, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealCompletenessWitness}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, quotient stream equality, selected limits, field-faithfulness, nontriviality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked four-face carrier admission, route exposure, nonescape, transport, replay, provenance, local naming, and separate TasteGate evidence for BEDC.Derived.RealCompletenessWitnessUp.

Chapter 4291

A Concrete Naming Certificate for $\mathsf{SmoothManifold}^{\uparrow}$

Following John M. Lee's smooth manifold presentation, the $\mathsf{SmoothManifold}^{\uparrow}$ packet records a finite atlas, overlap, transition, and replay surface for a manifold object already routed through chapter 96, chapter 88, chapter 47, chapter 130, and chapter 97. It names the differentiable atlas boundary used by later bundle, connection, and metric consumers; it is not a host smooth-space primitive, not a global coordinate choice, and not an ambient classical maximal-atlas theorem.

Definition 4291.1 (Smooth manifold carrier)**.** *A* $\mathsf{SmoothManifold}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, T, V, A, O, S, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the underlying* $\mathsf{Manifold}^{\uparrow}$ *carrier row.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *chart-domain ledger, recording the displayed open-domain data used by the atlas.*
3. *V is the finite* $\mathsf{VecSpace}^{\uparrow}$ *model row for the chart codomains.*
4. *A is the atlas-admission row, listing the chart names, domains, codomain windows, and coverage witnesses accepted by the packet.*
5. *O is the chart-overlap ledger, recording the finite intersections on which transition comparisons are requested.*
6. *S is the smooth-transition classifier row, whose entries are read through* $\mathsf{ContinuousMap}^{\uparrow}$ *and the displayed finite differentiability tests available on the model* $\mathsf{VecSpace}^{\uparrow}$ *windows.*
7. *R is the* $\mathsf{Bundle}^{\uparrow}$ *readiness row, exposing the atlas route that downstream tangent, frame, and coordinate-bundle packets may consume.*
8. *H records componentwise* hsame *transport for the manifold carrier, topology ledger, model vector space rows, atlas, overlaps, smooth-transition classifiers, bundle-readiness row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from chart requests through B, T, V, A, O, S, R.*
10. *P is the* Pkg *provenance over* $\mathsf{SmoothManifold}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Bundle}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SmoothManifold}^\uparrow}$ *row.*
The classifier compares accepted packets only through $B, T, V, A, O, S, R, H, C, P, N$. It has no coordinate for a selected maximal atlas, ambient set-theoretic smooth structure, global chart, classical paracompactness theorem, connection form, or Riemannian metric.

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldTasteGate single carrier alignment`

Theorem 4291.2 (Smooth manifold NameCert obligations)**.** *For every accepted* $\mathsf{SmoothManifold}^\uparrow$ *carrier $M = (B, T, V, A, O, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SmoothManifold}^\uparrow}$ *surface consists exactly of carrier admission, atlas admission, chart-overlap smoothness, transition compatibility,* hsame *transport,* Cont *replay, package provenance, and local naming obligations. In particular, the atlas row must display finite chart admission, the overlap row must display every inspected intersection, the transition row must display smooth compatibility on those overlaps, and the replay row must reproduce the same route before any* $\mathsf{Bundle}^\uparrow$ *consumer reads R.*

Proof. Project the accepted carrier. The base rows B, T, V expose the topological manifold boundary, chart-domain ledger, and model vector-space windows before the atlas can be read.

The atlas row A is then checked against those rows. Its chart names, displayed domains, codomain windows, and finite coverage witnesses are the only admission data available to the packet.

On each displayed overlap in O, the transition classifier S reads the corresponding coordinate changes through $\mathsf{ContinuousMap}^\uparrow$ and the finite differentiability tests attached to the model $\mathsf{VecSpace}^\uparrow$ windows. Transition compatibility is therefore stored in S rather than in a hidden maximal-atlas object.

Finally H, C, P, N transport, replay, source, and name the same rows, and R exposes only that displayed atlas route to $\mathsf{Bundle}^\uparrow$ consumers. Since the carrier has no coordinate for a selected maximal atlas, global chart, connection form, or metric tensor, the listed obligations exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldNamecertObligations`

Theorem 4291.3 (Smooth manifold bundle-readiness route)**.** *Every* $\mathsf{Bundle}^\uparrow$*-facing consumer of an accepted* $\mathsf{SmoothManifold}^\uparrow$ *packet factors through the displayed route*

$$\mathsf{Manifold}^\uparrow \longmapsto \mathsf{Topology}^\uparrow \longmapsto \mathsf{VecSpace}^\uparrow \longmapsto \mathsf{ContinuousMap}^\uparrow \longmapsto \mathsf{Bundle}^\uparrow$$

as represented by the rows $B, T, V, A, O, S, R, H, C, P, N$. The consumer may read coordinate-bundle readiness only after atlas admission, overlap smoothness, transition compatibility, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start with the NameCert obligation theorem. It places every accepted bundle-facing read behind the carrier, topology, model-space, atlas, overlap, and transition rows.

The row R is downstream of S, so a bundle consumer cannot inspect a smooth coordinate route before transition compatibility has been displayed on each overlap requested by O. The rows H and C preserve the same order under transport and replay.

The package and local naming rows record that the exposed route is the finite $\mathsf{SmoothManifold}^\uparrow$ atlas surface. Therefore later bundle, connection, and metric packets consume a named differentiable-atlas boundary rather than an ambient maximal atlas or a global smooth-space primitive. □

Lean4 ✓ `BEDC.Derived.SmoothManifoldUp.SmoothManifoldBundleReadinessRoute`

Falsifiable prediction: Every accepted $\mathsf{SmoothManifold}^{\uparrow}$ consumer that reaches $\mathsf{Bundle}^{\uparrow}$ readiness without reading atlas admission, chart-overlap smoothness, transition compatibility, hsame transport, Cont replay, Pkg provenance, and local NameCert rows refutes this packet surface.

Closure status for $\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}$.

Constructive story: A $\mathsf{SmoothManifold}^{\uparrow}$ packet is generated from a Manifold carrier row, Topology chart-domain ledger, VecSpace model windows, atlas admission, chart-overlap ledger, smooth-transition classifier, Bundle readiness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4291.1, Definition 4291.2, and Definition 4291.3 seed the finite smooth-atlas route from manifold and topology rows to Bundle consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SmoothManifold}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, selected maximal atlases, ambient smooth-space primitives, global charts, paracompactness, connections, or Riemannian metrics.

Upgrade path: Obligation closure requires checked carrier admission, atlas admission, overlap smoothness, transition compatibility, transport, replay, provenance, and local naming for BEDC.Derived.SmoothManifoldUp.

Chapter 4292

A Concrete Naming Certificate for $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$

The $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ packet realizes the normal-endpoint budget from the MetaCIC normalization frontier as finite NameCert data. It sits beside chapter 3954, chapter 731, chapter 737, chapter 936, and chapter 3439. The packet names the accepted endpoint budget that consistency consumers may inspect; it does not erase the obstruction against blanket strong normalization.

Definition 4292.1 (MetaCIC normal-endpoint budget carrier)**.** *A* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, K, B, F, O, H, C, P, N, G)$$

with the following displayed rows:

1. *T is the typed closed-term or typed fragment row whose normal endpoint is being budgeted.*
2. *K is the candidate-evidence row, read before any normalization-facing consumer sees an endpoint.*
3. *B is the bounded-normal endpoint row, recording the finite fuel, reduction trace, and bounded checker evidence available for T.*
4. *F is the finished-normal seal row, exposing the normal endpoint only after candidate evidence and bounded computation rows have been displayed.*
5. *O is the obstruction row, recording the retained subject-reduction or confluence boundary when the endpoint budget is unavailable for a wider theorem.*
6. *H records componentwise* hsame *transport for typed rows, candidate evidence, bounded endpoints, finished-normal seals, obstruction rows, replay, provenance, local naming, and gate rows.*
7. *C records displayed* Cont *replay from endpoint requests through T, K, B, F, O.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$*, the MetaCIC strong-normalization boundary, the subject-reduction obstruction, the confluence-normal fragment,* $\mathsf{ZNormal}^{\uparrow}$*, the bounded-normal conversion bridge,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}}$ *row.*

10. *G is the TasteGate landing row requiring* Nontrivial *and* FieldFaithful *instances for the displayed carrier surface before the packet is treated as a closed audit object.*

The classifier compares accepted packets only through $T, K, B, F, O, H, C, P, N, G$. It has no coordinate for evidence-free strong normalization, unrestricted subject reduction, quotient conversion equality, hidden fuel, or an unbounded normalizer.

Theorem 4292.2 (MetaCIC normal-endpoint budget NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *carrier $E = (T, K, B, F, O, H, C, P, N, G)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointBu}}$ *surface consists exactly of candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction retention,* hsame *transport,* Cont *replay,* Pkg *provenance, local naming, and TasteGate landing obligations. Every consistency-facing consumer must display those rows before it reads the endpoint budget.*

Proof. Project the accepted carrier. The typed row T is only a source row; it is not a normalization theorem by itself.

The candidate row K is read before the bounded endpoint row B. The finished-normal seal F is downstream of both rows, so a consumer cannot reach the normal endpoint without candidate evidence and bounded computation evidence.

The obstruction row O remains part of the same packet. It records the subject-reduction or confluence boundary that blocks reading the endpoint budget as a blanket strong-normalization theorem.

Finally H, C, P, N, G transport, replay, source, name, and gate the same carrier. Since the classifier has no hidden fuel coordinate, unbounded normalizer, quotient conversion equality, or evidence-free theorem row, the listed obligations exhaust the endpoint budget surface. □

Theorem 4292.3 (MetaCIC normal-endpoint budget consistency boundary)**.** *Every consistency-facing read from an accepted* $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ *packet factors through the finite route from chapter 3954 through chapter 731, chapter 737, chapter 936, and chapter 3439, as mediated by the rows $T, K, B, F, O, H, C, P, N, G$. Thus the endpoint can support a bounded normal-consistency read only while candidate evidence, obstruction retention, and finished-normal sealing remain displayed.*

Proof. Use the obligation theorem to expose the row order. Candidate evidence K and bounded endpoint evidence B precede the finished-normal seal F.

The obstruction row O remains visible between the strong-normalization boundary and the subject-reduction or confluence frontier. Therefore the route through the confluence-normal fragment, $\mathsf{ZNormal}^{\uparrow}$ endpoint, and bounded-normal conversion bridge is a finite endpoint budget, not a full subject-reduction or strong-normalization theorem.

Transport, replay, provenance, local naming, and the TasteGate landing row preserve the same dependency order. A consistency consumer therefore reads the normal endpoint only as a displayed budget with retained obstruction evidence. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICNormalEndpointBudget}^{\uparrow}$ packet must display candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction, hsame, Cont, Pkg, and NameCert rows before any consistency consumer reads it.

Independence witness: metacic-strong-normalization-boundary, metacic-subject-reduction-obstruction, metacicconfluencenormalfragment, znormal, bounded-normal-conversion-bridge: the normal-endpoint budget consumes those sibling frontiers as independent rows and does not collapse them into a blanket strong-normalization or subject-reduction theorem.

The Lean TasteGate landing condition for MetaCICNormalEndpointBudget$^{\uparrow}$ includes explicit Nontrivial and FieldFaithful instances for the displayed carrier surface.

Closure status for NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$.

Constructive story: A MetaCICNormalEndpointBudget$^{\uparrow}$ packet is generated from typed MetaCIC source rows, candidate evidence, bounded-normal endpoint evidence, a finished-normal seal, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, local NameCert data, and TasteGate landing rows.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$, seed)

Scope closed: Definition 4292.1, Definition 4292.2, and Definition 4292.3 seed the finite normal-endpoint budget for consistency-facing MetaCIC consumers.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, blanket strong normalization, unrestricted subject reduction, quotient conversion equality, hidden fuel, or an unbounded normalizer.

Upgrade path: Obligation closure requires checked candidate evidence, bounded-normal endpoint, finished-normal seal, obstruction retention, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful rows for BEDC.Derived.MetaCICNormalEndpointBudgetUp.

Chapter 4293

A Concrete Naming Certificate for $\mathsf{PolishCauchyBasis}^{\uparrow}$

Following Alexandrov–Urysohn, Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet records the basis-level handoff by which a complete separable metric source exposes Cauchy-filter basis data before any completion consumer reads a limit. It sits between the $\mathsf{PolishSpace}^{\uparrow}$ source of chapter 1490, the $\mathsf{CauchyFilterBasis}^{\uparrow}$ generator of chapter 1501, the $\mathsf{CompleteMetric}^{\uparrow}$ handoff of chapter 128, and the $\mathsf{SeparableMetric}^{\uparrow}$ dense schedule of chapter 1458. The packet is a finite BEDC handoff surface for Polish-space Cauchy bases, not a quotient completion, not a choice-selected dense sequence, and not an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 4293.1 (Polish Cauchy basis carrier)**.** *A* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, B, K, D, S, R, O, H, C, G, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{PolishSpace}^{\uparrow}$ *source row, carrying the complete separable metric packet over which every later basis read is interpreted.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row, recording the directed finite basis generator used by Cauchy-filter consumers.*
3. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row attached to the same metric surface as X; it records the completion-facing route without selecting a limit point.*
4. *D is the* $\mathsf{SeparableMetric}^{\uparrow}$ *dense schedule row, stored as public finite name data rather than as a choice function over all balls.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row used to inspect dense names, basis requests, and Cauchy-window demands.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those stream windows, giving regular rational observations before any basis comparison is made.*
7. *O is the observation ledger joining dense-window requests, regular-rational readbacks, Cauchy-basis refinements, and completion-handoff checks.*
8. *H records componentwise* hsame *transport for X, B, K, D, S, R, O and for the local certificate rows.*

9. *C records displayed* Cont *replay from a Polish-source request through dense schedules, stream windows, regular readbacks, Cauchy-basis refinement, and the completion handoff.*
10. *G is the* Pkg *provenance over* $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *row.*

The classifier compares two accepted packets only after transporting these displayed rows to a common Polish source, Cauchy-basis generator, completion-handoff row, dense schedule, stream-window row, and regular readback ledger. It has no coordinate for choice-selected dense points, quotient completion equality, host point equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PolishCauchyBasisUp.PolishCauchyBasisTasteGate single carrier alignment`

Theorem 4293.2 (Polish Cauchy basis NameCert obligations)**.** *For every accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *carrier $P = (X, B, K, D, S, R, O, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Polish source, Cauchy-filter basis, complete-metric handoff, separable dense schedule, stream windows, regular rational readback, observation ledger, transport, continuation, provenance, and local naming rows;*
2. *common-source classification, requiring the B, K, D, S, R, O rows to be read over the same transported* $\mathsf{PolishSpace}^{\uparrow}$ *source row X;*
3. *basis-directedness exposure, requiring every Cauchy-basis refinement consumed by B to be visible in S, R, O before the completion handoff K is read;*
4. *dense-schedule compatibility, requiring the separable row D and the Cauchy-basis row B to share the same stream-window and regular-readback surface;*
5. *completion-handoff non-escape, requiring every completion-facing consumer to pass through B, S, R, O before reaching K;*
6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4293.1. The projection exposes the Polish source, Cauchy-basis generator, complete-metric handoff, dense schedule, stream windows, regular readbacks, observation ledger, transport rows, continuation rows, provenance row, and local name.

The classifier route starts at X. Componentwise transport in H moves B, K, D, S, R, O together, so a consumer never compares a basis row over one Polish source with a completion handoff or dense schedule over another.

The observation ledger O fixes the finite read order. Dense requests and basis refinements are observed through S and R before either the Cauchy-filter basis or complete-metric handoff can be consumed.

Finally replay C and attach G, N. These rows route, source, and name the same displayed packet, with no additional coordinate for selected dense points, quotient completion equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the listed rows exhaust the local naming surface. □

Theorem 4293.3 (Polish Cauchy basis completion handoff)**.** *Let $P = (X, B, K, D, S, R, O, H, C, G, N)$ be an accepted* $\mathsf{PolishCauchyBasis}^{\uparrow}$ *packet. Every completion-facing consumer of the Polish source*

X reaches the $\mathsf{CompleteMetric}^{\uparrow}$ *handoff K only through the displayed Cauchy-filter basis row B, the separable dense schedule D, the* $\mathsf{StreamName}^{\uparrow}$ *windows S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the observation ledger O, transported by H and replayed by C.*

The visible route is

$$X, D, S, R, B, O \longmapsto K \longmapsto H, C, G, N.$$

It exports a finite completion handoff for downstream completion, Baire, and fixed-point packets; it does not export a quotient completion class, a selected dense sequence, host point equality, or an ambient real-completeness principle.

Proof. Read X from the carrier in Definition 4293.1. The Polish source supplies the common complete separable metric surface, while D, S, R display the finite dense-window observations available to the packet.

Use the NameCert obligations of Definition 4293.2. The basis-directedness and dense-schedule compatibility clauses force the Cauchy-filter basis row B to be read through the same stream windows, regular readbacks, and observation ledger as the dense schedule.

The completion-handoff clause then places K after B, S, R, O. Transport by H and replay by C preserve that order, while G, N package and name the same route. Therefore every completion-facing consumer receives only the displayed finite handoff and cannot bypass it through choice, quotient equality, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{PolishCauchyBasis}^{\uparrow}$ packet is generated from a PolishSpace source, CauchyFilterBasis basis, CompleteMetric handoff, SeparableMetric dense schedule, StreamName windows, RegSeqRat readback, an observation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, seed)

Scope closed: Definition 4293.1, Definition 4293.2, and Definition 4293.3 seed the basis-level Cauchy handoff from Polish-space data to completion-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PolishCauchyBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close choice-selected dense points, quotient completions, host point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common-source classification, basis-directedness exposure, dense-schedule compatibility, completion-handoff non-escape, transport, replay, provenance, and local naming for the same packet.

Chapter 4294

A Concrete Naming Certificate for $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$

The $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ packet realizes the hidden Q_6 stabilizer and orbit-persistence rows of Definition 9462.1352, Definition 9462.1354, Definition 9462.1356, and Definition 9462.1362 as finite BEDC NameCert data. It consumes the six-coordinate codon window of chapter 2234, the boundary tile packet of chapter 3393, the codon lift-orbit classifier of chapter 4059, and the median-closure surface of chapter 3545. The packet exposes support, stabilizer, orbit, and median-closure rows before any consumer may claim the hidden C_2 persistence or the restored $S_3 \times C_2$ surface.

Definition 4294.1 (Symmetry restoration stabilizer carrier)**.** *A* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, A, T, U, M, O, R, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite Q_6 codon-window row, including the six bit coordinates inspected by the stabilizer calculation.*
2. *A is the active-support row for $\mathcal{R}$ and its inactive-anchor closure, read through the boundary-tile packet.*
3. *T is the hidden C_2 stabilizer row, naming the identity and the coordinate transposition $\tau : s_1 \leftrightarrow f_2$ from Definition 9462.1352.*
4. *U is the orbit-persistence row, recording the active-support, inactive-anchor, and first-median-layer orbits of τ from Definition 9462.1354.*
5. *M is the median-closure stabilizer row, recording the displayed $S_3 \times C_2$ restoration surface of Definition 9462.1356.*
6. *O is the orbit ledger joining the nontrivial C_2 orbits, the median-closure orbit table, and the sealing-corner orbit-completion row.*
7. *R is the refusal row separating support symmetry from weighted reassignment symmetry, and excluding biological causality, stochastic fitness, host quotient codon semantics, and hidden enumeration oracles.*

8. *H records componentwise* hsame *transport for Q, A, T, U, M, O, R and for the local certificate rows.*

9. *C records displayed* Cont *replay from a support read through the stabilizer row, orbit-persistence row, median-closure row, orbit ledger, and refusal row.*

10. *P is the* Pkg *provenance over* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}$ $\mathsf{CodonLiftOrbitClassifier}^{\uparrow}$, $\mathsf{MedianClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStabilizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, T, U, M, O, R, H, C, P, N$. It has no coordinate for an unstated support set, an unrecorded stabilizer enumeration, a host quotient codon object, a biological mechanism, or a weighted-symmetry substitute for the support stabilizer.

Lean4 ✓ `BEDC.Derived.SymmetryRestorationStabilizerUp.SymmetryRestorationStabilizerTasteGate`
`single carrier alignment`

Theorem 4294.2 (Symmetry restoration stabilizer NameCert obligations)**.** *For every accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *carrier $S = (Q, A, T, U, M, O, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SymmetryRestorationStab}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the Q_6 codon window, active support, hidden C_2 stabilizer, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, transport, continuation, provenance, and local naming row;*

2. *support visibility, requiring every stabilizer or orbit read to expose the active-support row A before reading T, U, M, or O;*

3. *stabilizer exactness, requiring the hidden C_2 row T to name only the identity and $\tau : s_1 \leftrightarrow f_2$ for the active-support and first-median-layer reads;*

4. *median-closure compatibility, requiring the restored $S_3 \times C_2$ row M to be read only after the median-closure surface and its orbit ledger are displayed;*

5. *orbit non-escape, requiring any hidden C_2 orbit claim to pass through $A, T, U, M, O, H, C, P, N$;*

6. *package provenance and local* NameCert *naming for exactly the displayed packet.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 4294.1. The projection exposes the codon window, active support, stabilizer row, orbit-persistence row, median-closure stabilizer, orbit ledger, refusal row, and the structural rows H, C, P, N.

Support visibility comes first. The row A is the source for the stabilizer calculation, so T, U, M, and O cannot be read as free group-theoretic objects detached from the displayed support.

The stabilizer exactness row is T. It records exactly the identity and the transposition $\tau : s_1 \leftrightarrow f_2$ for the active-support surface, while U records the same nonidentity element along the inactive-anchor and first-median-layer rows.

The median-closure compatibility row is M, with orbit data stored in O. Transport by H and replay by C preserve this order, and P, N package and name the same rows. Thus no hidden orbit claim escapes the displayed support, stabilizer, median-closure, transport, replay, provenance, and local naming surface. □

Theorem 4294.3 (Symmetry restoration orbit non-escape)**.** *Every consumer claiming the hidden C_2 orbit of an accepted* $\mathsf{SymmetryRestorationStabilizer}^{\uparrow}$ *packet must display the support row A,*

the stabilizer row T, the orbit-persistence row U, the median-closure row M, the orbit ledger O, componentwise transport H, displayed replay C, package provenance P, and local NameCert *row N. The visible route is*

$$Q, A \longmapsto T, U \longmapsto M, O \longmapsto H, C, P, N.$$

It cannot be replaced by a host stabilizer calculation, an unrecorded orbit enumeration, a weighted-reassignment symmetry, quotient codon semantics, or a biological mechanism claim.

Proof. Read Q, A from the carrier in Definition 4294.1. These rows provide the finite codon-window coordinates and the active support over which the stabilizer calculation is meaningful.

Apply the NameCert obligations of Definition 4294.2. Support visibility and stabilizer exactness place T and U after A, while median-closure compatibility places M and O after the displayed closure surface.

Finally use orbit non-escape. The rows H, C, P, N transport, replay, source, and name the same finite route, and the refusal row separates it from weighted reassignment symmetry or host enumeration. Therefore a hidden C_2 orbit consumer must pass through the displayed packet and cannot claim the orbit from outside it. □

Falsifiable prediction: Any consumer claiming the hidden C_2 orbit without displaying support, stabilizer, median-closure, hsame, Cont, Pkg, and NameCert rows refutes the SymmetryRestorationStabilizer$^{\uparrow}$ packet.

Independence witness: q-six orbit/classification, median-closure sibling slugs: SymmetryRestorationStabilizer$^{\uparrow}$ is not bijective to the orbit/classification or median-closure siblings because this packet jointly carries active support, hidden C_2 stabilizer, orbit persistence, restored median-closure stabilizer, orbit ledger, refusal, componentwise transport, replay, provenance, and local naming as one accepted surface.

The Lean TasteGate file for this packet must include Nontrivial and FieldFaithful instances. The field-faithfulness witness keeps $Q, A, T, U, M, O, R, H, C, P, N$ as distinct rows so a checked TasteGate theorem can rule out hidden-orbit claims that omit the support, stabilizer, or median-closure coordinates.

Closure status for NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$.

Constructive story: A SymmetryRestorationStabilizer$^{\uparrow}$ packet is generated from Q6 codon-window rows, active-support rows, hidden C2 stabilizer rows, orbit-persistence rows, median-closure stabilizer rows, orbit ledgers, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, seed)

Scope closed: Definition 4294.1, Definition 4294.2, and Definition 4294.3 seed the finite support-stabilizer packet for the hidden Q6 orbit and median-closure restoration surface.

Formal verification: FormalStatus(NameCert$_{\text{SymmetryRestorationStabilizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological causality, stochastic fitness, quotient codon semantics, host stabilizer enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked codon-window admission, support visibility, stabilizer exactness, orbit persistence, median-closure compatibility, orbit non-escape, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.SymmetryRestorationStabilizerUp.

Chapter 4295

A Concrete Naming Certificate for $\mathsf{KolmogorovContinuity}^{\uparrow}$

Following Kolmogorov's continuity criterion, the $\mathsf{KolmogorovContinuity}^{\uparrow}$ packet records a finite-modulus continuity test for stochastic-process windows as displayed BEDC data. It consumes the probability carrier of chapter 184, the random-variable preimage surface of chapter 185, the metric-space rows of chapter 4230, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only the finite difference and modulus ledger; it is not a host path-continuity theorem, a choice of continuous modification, a quotient sample-path object, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 4295.1 (Kolmogorov continuity carrier)**.** *A* $\mathsf{KolmogorovContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, M, S, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite difference ledger, listing inspected process windows, endpoint pairs, random-variable rows, metric-distance requests, and the probability-space row used for each two-time difference.*
2. *M is the moment and finite-modulus ledger. It records the displayed exponent, scale, tolerance, and decay rows that certify the Kolmogorov bound on the inspected two-time differences.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that names the finite time and sample observations read by the process consumer.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational moment and modulus bounds extracted from I, M, S.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the real-valued bound reached only after the displayed regular rational readback.*
6. *E is the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{RandomVar}^{\uparrow}$ *endpoint-exposure row, keeping metric distance and random-variable measurability attached to the same inspected two-time difference.*
7. *H records componentwise* hsame *transport for two-time difference rows, moment/modulus rows, StreamName windows, RegSeqRat readback, Real seals, metric/random-variable exposure, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay of the finite route*

$$I, M, S \longmapsto Q \longmapsto R \longmapsto E \longmapsto H.$$

9. *P is the* Pkg *provenance over* $\mathsf{KolmogorovContinuity}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, S, Q, R, E, H, C, P, N$. It has no coordinate for ambient path continuity, a selected continuous modification, quotient sample paths, countable-choice schedules, host equality of stochastic processes, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.KolmogorovContinuityUp.KolmogorovContinuityTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.KolmogorovContinuityUp.KolmogorovContinuityCarrier`

Theorem 4295.2 (KolmogorovContinuityUp NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovContinuity}^{\uparrow}$ *carrier*

$$K = (I, M, S, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KolmogorovContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *two-time difference admission for I, requiring every inspected endpoint pair, process row, random-variable row, metric-distance request, and probability-space row to be displayed;*
2. *moment and modulus admission for M, requiring the exponent, scale, tolerance, and decay rows used by the criterion to be named as finite ledger data;*
3. *StreamName observation for S, requiring all time and sample windows consumed by the two-time difference rows to be displayed before any regular-rational readback;*
4. *regular-rational readback for Q, requiring moment and modulus bounds to pass through the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *surface;*
5. *Real seal exposure for R, requiring every real-valued bound to be reached only after Q;*
6. *metric and random-variable endpoint exposure for E, requiring distance and measurability rows to remain attached to the same inspected two-time difference;*
7. *transport, replay, provenance, and local naming through H, C, P, N, preserving exactly the displayed continuity-criterion packet.*

No Kolmogorov continuity obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 4295.1. The two-time difference ledger I supplies the only endpoint pairs and stochastic-process rows available to a consumer, and each such row already displays its $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, and $\mathsf{Met}^{\uparrow}$ support.

Read the moment and finite-modulus ledger M together with the observation row S. These rows name the inspected windows, exponents, scales, tolerances, and decay data before any readback can be exported.

The regular-rational row Q is the only route from the finite moment/modulus ledger to a numerical bound, and the real seal R is downstream of Q. The endpoint-exposure row E keeps the metric and random-variable coordinates attached to the same two-time difference rather than letting them become ambient path data.

Finally H, C, P, N transport, replay, source, and name the same tuple. Since the carrier has no coordinate for ambient path continuity, selected modifications, quotient sample paths, countable

choice, host process equality, or $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local naming surface. □

Theorem 4295.3 (KolmogorovContinuityUp finite-modulus route)**.** *Every continuity-facing consumer of an accepted* $\mathsf{KolmogorovContinuity}^{\uparrow}$ *packet reads the criterion through the finite route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, a two-time difference row first displays its probability-space, random-variable, and metric-distance support; the same request then exposes a finite StreamName window, a moment/modulus ledger, a regular rational readback, and only then a terminal real seal. The route exports a finite-modulus continuity criterion, not ambient path continuity, a chosen continuous modification, quotient sample paths, host process equality, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $K = (I, M, S, Q, R, E, H, C, P, N)$. By Definition 4295.2, a two-time difference read must first display the rows in I and the endpoint support in E, so the probability, random-variable, and metric data are fixed before a continuity-facing consumer sees a bound.

The same obligation surface places the finite StreamName window S and moment/modulus ledger M before the regular-rational readback Q. Thus the criterion is read from displayed windows and finite decay rows rather than from an infinite sample path.

The real-facing step is controlled by R, which is downstream of Q. Componentwise hsame transport and Cont replay preserve this order, while Pkg provenance and the local NameCert row name exactly the same route.

Therefore every consumer factors through the displayed finite-modulus path. The excluded path-continuity, modification-choice, quotient-sample-path, host-equality, countable-choice, and completeness claims are absent from the carrier and cannot be recovered from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovContinuity}^{\uparrow}$ packet is generated from finite two-time difference rows, moment and modulus ledger rows, StreamName windows, RegSeqRat readback, Real seals, metric and random-variable endpoint exposure, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KolmogorovContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 4295.1, Definition 4295.2, and Definition 4295.3 seed the finite-modulus continuity criterion over probability, random-variable, metric, StreamName, RegSeqRat, Real, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KolmogorovContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient path continuity, chosen continuous modifications, quotient sample paths, countable choice, host process equality, RealUp completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked two-time difference admission, moment/modulus admission, StreamName observation, RegSeqRat readback, Real seal exposure, endpoint exposure, and transport/provenance/name exhaustion for BEDC.Derived.KolmogorovContinuityUp.

Chapter 4296

A Concrete Naming Certificate for $\mathsf{TaggedRealPartition}^{\uparrow}$

Following Bishop and Bridges constructive integration, the $\mathsf{TaggedRealPartition}^{\uparrow}$ packet records a finite partition of a located real interval together with a displayed tag inside each cell. It refines the located partition carrier of chapter 2248 and the oscillation-controlled partition carrier of chapter 3800; it also reads its endpoint and tag data through the $\mathsf{StreamName}^{\uparrow}$ source of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies the finite tag-bearing surface needed by L10 integration and oscillation routes without replacing it by an untagged mesh, a host partition object, selected representatives, quotient real equality, countable choice, or ambient completeness.

Definition 4296.1 (Tagged real partition carrier)**.** *A* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, M, A, G, O, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the source* $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$ *row exposing the located interval, finite cell order, endpoint containment, and mesh precision.*
2. *M is the finite mesh row inherited from I, listing adjacent cells and the common refinement ledger used by later integration consumers.*
3. *A is the cell-containment row, certifying for every mesh cell that its endpoints and rational witness rows remain inside the located interval source.*
4. *G is the tag row family, assigning to every displayed cell a tag history whose* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *reads certify containment in that same cell before any* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*
5. *O is the oscillation and modulus ledger, recording the tolerance request, the cellwise oscillation bound, and the refinement compatibility supplied by* $\mathsf{FiniteOscillationPartition}^{\uparrow}$*.*
6. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened for interval endpoints, cell endpoints, and tags.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family for the same finite windows and mesh precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal family reached only after the source interval, cell containment, tag containment, oscillation ledger, window, and regular readback rows have been displayed.*

9. *H records componentwise* hsame *transport for I, M, A, G, O, S, R, E and for the replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a tagged-partition read through*

$$I, M, A, G, O, S, R, E \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{LocatedRealIntervalPartition}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these displayed interval-source, mesh, containment, tag, oscillation, window, readback, seal, transport, replay, provenance, and local naming rows. It has no coordinate for a host tagged partition, an unbounded tag selector, quotient real equality, chosen completed tag values, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.TaggedRealPartitionUp.TaggedRealPartitionTasteGate single carrier alignment`

Theorem 4296.2 (Tagged real partition NameCert obligations)**.** *For every accepted* $\mathsf{TaggedRealPartition}^{\uparrow}$ *carrier $T = (I, M, A, G, O, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$ *surface consists exactly of carrier admission, located interval and mesh exposure through I, M, cell-containment certification through A, tag containment through G, oscillation and modulus accounting through O, finite* $\mathsf{StreamName}^{\uparrow}$ *windows through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seals through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Every integration-facing or oscillation-facing read of the tagged partition must therefore pass through the displayed finite mesh, tag-containment, oscillation, window, readback, and seal rows before any terminal real-value consumer is allowed to inspect the packet.

Proof. Project the carrier of Definition 4296.1. The tuple lists the only rows available to a tagged-partition consumer: the interval source, mesh, containment ledger, tag rows, oscillation ledger, finite windows, regular readbacks, real seals, transport, replay, provenance, and local naming rows.

The source interval and cell obligations are I, M, A. The located partition row supplies the interval and mesh; the containment row attaches each cell to the same source, so a tag cannot be read against a detached host interval.

The tag and oscillation obligations are G, O. The tag row certifies cell membership for each tag through the same finite-window and regular-readback route, while the oscillation row records the tolerance and refinement ledger inherited from the oscillation-partition packet.

The readback and seal obligations are S, R, E. A consumer reaches the StreamName windows only after the interval, cell, tag, and oscillation rows are visible, then reads the regular rational rows and finally the terminal Real seal.

The structural rows H, C, P, N transport, replay, package, and name exactly this route. Since the carrier has no coordinate for a host tagged partition, an unbounded selector, quotient real equality, chosen completed tag values, countable choice, or ambient completeness, the displayed NameCert obligations exhaust the accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{TaggedRealPartition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaggedRealPartition}^{\uparrow}$ packet is generated from a LocatedRealIntervalPartitionUp source, finite mesh and cell-containment rows, per-cell tag rows, oscillation and modulus ledgers, StreamName windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, seed)

Scope closed: Definition 4296.1 and Definition 4296.2 seed the finite tag-bearing partition interface for L10 integration and oscillation consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{TaggedRealPartition}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close integration convergence, Darboux sums, quotient real equality, selected tag values, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cell containment, tag containment, oscillation-modulus accounting, StreamName windows, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.TaggedRealPartitionUp.

Chapter 4297

A Concrete Naming Certificate for $\mathsf{FiniteFanBarModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges finite-bar analysis, the $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet records the finite carrier that turns admitted fan-bar rows and local modulus rows into a tolerance-only uniform-modulus ledger. It sits over the fan certificate of chapter 400, the compact metric finite-net surface of chapter 129, the continuous-map carrier of chapter 130, the rational lower-bound fold of chapter 4118, the $\mathsf{Rat}^{\uparrow}$ arithmetic source of chapter 37, and the $\mathsf{UniformModulus}^{\uparrow}$ handoff of chapter 369. It realizes the compact-uniform-continuity route of chapter 9237 without importing open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, or ambient real completeness.

Definition 4297.1 (Finite fan-bar modulus carrier)**.** *A* $\mathsf{FiniteFanBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, K, L, Q, R, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite fan-bar admission row, recording the fan, decidable bar classifier, finite depth bound, and finite branch-prefix witness window.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-probe row, aligning the bar prefixes with the compact finite-net probes consumed by the compact-uniform-continuity route.*
3. *L is the local modulus row read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *packet at each admitted bar or probe entry.*
4. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *radius and tolerance row, carrying the finitely many local radii and requested tolerances.*
5. *R is the lower-bound modulus extraction row, obtained by primitive recursion over the finite fan-bar and probe bundle and by certified rational comparison in Q.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row, exposing the extracted lower bound as a tolerance-only modulus ledger.*
7. *H records componentwise* hsame *transport for the bar, compact-probe, local-modulus, rational, extraction, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite route*

$$B, K, L, Q \longmapsto R \longmapsto U.$$

9. *P is the* Pkg *provenance over* FiniteFanBarModulus$^\uparrow$, FanTheorem$^\uparrow$, CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, UniformModulus$^\uparrow$, Rat$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *row.*

The classifier compares accepted packets only through $B, K, L, Q, R, U, H, C, P, N$. It has no coordinate for an open cover, a selected infinite branch, countable choice, quotient equality of maps, selected limits, a host compactness theorem, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteFanBarModulusUp.FiniteFanBarModulusTasteGate single carrier alignment`

Theorem 4297.2 (Finite fan-bar modulus NameCert obligations)**.** *For every accepted* FiniteFanBarModulus$^\uparrow$ *carrier $F = (B, K, L, Q, R, U, H, C, P, N)$, the local* NameCert$_{\text{FiniteFanBarModulus}^\uparrow}$ *surface consists exactly of carrier admission, finite bar admission through B, compact finite-probe alignment through K, local continuous-map modulus exposure through L, positive rational radius accounting through Q, lower-bound modulus extraction through R,* UniformModulus$^\uparrow$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

In particular, a compact-uniform-continuity consumer may read a uniform modulus from this packet only after the finite bar, compact-probe, local-modulus, rational-radius, and lower-bound extraction rows have all been displayed.

Proof. Project the carrier of Definition 4297.1. The tuple displays exactly the finite bar, compact-probe, local-modulus, rational-radius, lower-bound extraction, uniform-modulus, transport, replay, provenance, and local naming rows.

Finite bar admission is the row B. It supplies the decidable bar classifier and finite depth witness, so the argument ranges over displayed finite prefixes rather than selected infinite branches.

The compact and local-modulus data are K, L, Q. The compact-probe row aligns finite bar prefixes with compact finite-net probes, the continuous-map row supplies local moduli at those probes, and the rational row stores the positive local radii and tolerance requests.

Lower-bound extraction is the row R. It is obtained by primitive recursion over the finite bar and probe rows, using the positive rational comparisons in Q to retain a certified lower bound for all admitted local radii.

Finally, U, H, C, P, N expose, transport, replay, source, and name that extracted lower bound as a UniformModulus$^\uparrow$ ledger. Since the carrier has no coordinate for open-cover compactness, branch selection, countable choice, quotient function equality, selected limits, host compactness, or ambient completeness, the displayed obligations exhaust the accepted packet. □

Theorem 4297.3 (Finite fan-bar lower-bound modulus extraction)**.** *Every compact-uniform-continuity read of an accepted* FiniteFanBarModulus$^\uparrow$ *packet factors through*

$$B, K, L, Q, R, U, H, C, P, N.$$

The extracted row R is the only lower-bound modulus source available to the UniformModulus$^\uparrow$ *consumer: it is computed from finite fan-bar admission, compact finite-probe alignment, local continuous-map moduli, and positive* Rat$^\uparrow$ *radius comparisons, and it exports no open-cover subcover, selected branch, selected limit, quotient map equality, or ambient compactness datum.*

Proof. Start with the NameCert obligations of Definition 4297.2. They require every compact-uniform-continuity consumer to read B, K, L, Q before the lower-bound row R is available.

The extraction step is finite. The bar depth and compact probe rows determine the finite index family, and the positive rational ledger compares the finitely many local radii by primitive recursion to produce a certified lower bound.

Once R is displayed, the uniform-modulus row U may consume it as a tolerance-only ledger. The structural rows H, C, P, N preserve and name the same route under componentwise transport and replay.

No excluded datum can be read from the carrier. Open-cover compactness, selected branches, selected limits, quotient map equality, countable choice, and ambient compactness are absent from $B, K, L, Q, R, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteFanBarModulus}^{\uparrow}$ packet is generated from finite fan-bar admission, compact finite-probe alignment, local ContinuousMap modulus rows, positive RatUp radius ledgers, lower-bound extraction, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4297.1, Definition 4297.2, and Definition 4297.3 seed the finite fan-bar lower-bound route consumed by compact-uniform-continuity and UniformModulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient function equality, selected limits, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite bar admission, compact finite-probe alignment, local modulus exposure, RatUp lower-bound extraction, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteFanBarModulusUp.

Chapter 4298

A Concrete Naming Certificate for $\mathsf{EdelsteinFixedPoint}^{\uparrow}$

Following Edelstein's compact-metric fixed-point theorem and Bishop–Bridges constructive metric practice, the $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet records the finite NameCert surface for a strictly contractive self-map on a compact metric carrier. It sits after the metric source of chapter 57, the compact-metric finite inspection surface of chapter 129, the sequential map discipline of chapter 1466, the contraction comparison row of chapter 3529, and the displayed iteration route of chapter 2359. The packet names only the compact source, the strict distance-decrease row, the sequential-continuity row, the orbit comparison ledger, transport, replay, provenance, and local naming; it does not import choice, quotient completion, a selected host fixed point, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a Banach contraction ratio.

Definition 4298.1 (Edelstein fixed-point carrier)**.** *An* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, O, C, S, H, R, P, N)$$

with the following displayed rows:

1. *X is the underlying* $\mathsf{Met}^{\uparrow}$ *source row, carrying the point classifier and the distance observations inspected by the fixed-point consumer.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row over X, exposing only the finite compactness windows and cluster-service rows accepted by the compact-metric certificate.*
3. *F is the self-map row, presented through the same metric carrier and never as an untracked host function.*
4. *O is the orbit row generated from F, recording the displayed Picard-style finite prefix consumed by the fixed-point search.*
5. *C is the strict contraction-comparison row: whenever the inspected points are classified as distinct by the metric row, the displayed distance from $F(x)$ to $F(y)$ is strictly below the displayed distance from x to y.*
6. *S is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *row for F, binding cluster reads of the orbit to the image row without adding a semantic continuity theorem outside the displayed packet.*
7. *H stores componentwise* hsame *transport for metric, compactness, self-map, orbit, contraction-comparison, sequential-continuity, continuation, provenance, and naming rows.*

8. *R stores displayed* Cont *replay from the orbit row to the compact cluster row, from the cluster row through sequential continuity, and from the strict comparison row to the fixed-point equality boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SequentialContinuity}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, F, O, C, S, H, R, P, N$. It has no coordinate for quotient completion, countable choice, a Banach contraction constant, an ambient complete real line, or a host-selected fixed point.

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointTasteGate single carrier alignment`

Theorem 4298.2 (Edelstein fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *carrier $E = (X, K, F, O, C, S, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$ *surface consists exactly of metric carrier admission through X, compact-metric finite inspection through K, self-map admission through F, Picard-style orbit exposure through O, strict contraction comparison through C, sequential-continuity cluster transport through S, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.EdelsteinFixedPointUp.EdelsteinFixedPointCarrier namecert obligations`

Proof. Project the carrier of Definition 4298.1. The metric row X and compact-metric row K are the only source rows available to the certificate, so every inspected point, distance comparison, and compactness read is routed through those two displayed surfaces.

The self-map row F and orbit row O provide the visible finite Picard-style prefix. The strict comparison row C is separate from the Banach contraction surface: it records pointwise distance decrease on distinct displayed pairs and carries no global contraction ratio.

The sequential-continuity row S is the only route from a compact cluster read of the orbit to an image read under F. Transport through H, replay through R, provenance through P, and naming through N preserve this same finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 4298.3 (Edelstein compact-route nonescape)**.** *Every fixed-point read exported by an accepted* $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *packet factors through the internal route*

$$X \longmapsto K \longmapsto O \longmapsto S \longmapsto C$$

together with F, H, R, P, N. The route may compare with the $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$ *neighbours as named dependency rows, but it does not consume a Banach contraction constant, quotient completion equality, selected subsequence principle, choice operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Start with a fixed-point read. The carrier first exposes the metric source X and the compactness service K, so the consumer can only inspect displayed finite compact-metric windows and the point classifier attached to X.

The orbit row O supplies the visible Picard-style finite prefix for the self-map F. Compactness is consumed only through K, and the image of a cluster read is transported only through the sequential-continuity row S.

The strict comparison row C is then the only row that can rule out a non-fixed distinct-pair boundary. Since C contains no global ratio and the remaining rows H, R, P, N only transport, replay, source, and name the same finite data, the exported route cannot leave the displayed Edelstein packet for choice, quotient completion, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ fixed-point read factors through one MetricUp source row, one CompactMetricUp finite inspection row, one self-map row, one PicardIterationUp-style orbit row, one strict comparison row, one SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, sequentialcontinuity, banach-contraction, picard-iteration: $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact-metric cluster service, strict pointwise distance decrease, and sequential-continuity image transport without a Banach ratio, selected endpoint, quotient completion, or ambient completeness coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}$.

Constructive story: An $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite inspection row, a displayed self-map row, a PicardIterationUp-style orbit row, a strict distance-decrease comparison row, a SequentialContinuityUp cluster-transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4298.1, Definition 4298.2, and Definition 4298.3 bind the compact-metric strict-contraction route through the displayed orbit, cluster, sequential-continuity, and comparison rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EdelsteinFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, uniqueness, quotient completion equality, Banach contraction constants, selected fixed points, choice principles, bridge checking, Lean verification, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric inspection, orbit exposure, strict comparison, sequential-continuity cluster transport, nonescape, replay, provenance, and local naming for BEDC.Derived.EdelsteinFixedPointUp

Chapter 4299

A Concrete Naming Certificate for $\mathsf{CompletionLadder}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompletionLadder}^{\uparrow}$ packet records the finite BEDC naming surface that ties the current Real and Completion route into one phase object. It sits over the dyadic tolerance ledger of chapter 291, the finite-window stream surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the separated-completion boundary of chapter 1612, and the completion-unit handoff of chapter 1527. The ladder is a finite dependency packet for the phase route; it is not quotient stream equality, not a selected limit, not countable choice, not trichotomy, not open-cover compactness, and not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4299.1 (Completion ladder carrier)**.** *A* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, W, R, E, S, U, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dyadic tolerance row naming the precision requests that enter the route.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *stream-window row listing the finite observations read at those tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from D and W.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the dyadic tolerance, stream window, and regular readback rows have been displayed.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *witness row that consumes the same displayed real-facing route as a separated completion boundary.*
6. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *unit row that records the source-to-completion handoff along the same finite route.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, W \longmapsto R \longmapsto E \longmapsto S, U.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompletionLadder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionUnit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, E, S, U, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionLadderUp.CompletionLadderTasteGate single carrier alignment`

Theorem 4299.2 (Completion ladder NameCert obligations)**.** *For every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier $L = (D, W, R, E, S, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic tolerance, stream window, regular readback, Real seal, separated-completion witness, Cauchy-completion unit, componentwise transport, continuation replay, package provenance, and local naming rows;*
2. *tolerance-window compatibility, requiring each finite stream read in W to be indexed by a dyadic tolerance request in D before regular readback is inspected;*
3. *regular-readback exactness, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *read in R to consume only the displayed D, W route;*
4. *Real-seal order, requiring E to be read only after D, W, R have been displayed;*
5. *completion-boundary order, requiring S and U to consume the same D, W, R, E route through the stored continuation row C;*
6. *transport stability, requiring componentwise* hsame *transport in H to preserve the listed route and no other completion data;*
7. *ledger non-escape, excluding quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, and completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Carrier admission is projection from Definition 4299.1. The tuple lists the dyadic tolerance row, stream-window row, regular readback row, Real seal, separated-completion witness, Cauchy-completion unit, transport rows, replay row, provenance, and local name.

The first dependency step reads D and W. Since W is the only stream-window coordinate and D is the only tolerance coordinate, each finite observation entering the regular readback must pass through those two rows.

The regular readback row R is then downstream of D, W, and the Real seal row E is downstream of R. The replay row C records this order before either completion-facing row is inspected.

The separated-completion witness S and Cauchy-completion unit U therefore consume the same displayed D, W, R, E route. Componentwise transport by H preserves those rows, while P and N source and name the same packet.

No remaining coordinate can export quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness. Thus the listed rows exhaust the local NameCert surface. □

Theorem 4299.3 (Completion ladder separated Real route)**.** *Every accepted* $\mathsf{CompletionLadder}^{\uparrow}$ *carrier factors any separated-completion or completion-unit consumer through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{SeparatedCompletion}^{\uparrow},\quad \mathsf{CauchyCompletionUnit}^{\uparrow}.$$

Equivalently, a consumer of S or U must first read the displayed dyadic tolerance row D, stream-window row W, regular readback row R, and Real seal row E, with componentwise transport, continuation replay, provenance, and local naming supplied by H, C, P, N.

Proof. Start from an accepted carrier $L = (D, W, R, E, S, U, H, C, P, N)$. The only completion-facing rows are S and U, so any separated-completion or completion-unit consumer must inspect one of those coordinates.

The definition places both coordinates after D, W, R, E. The continuation row C records the same order, forcing each consumer read to pass from dyadic tolerance and finite window through regular rational readback before the Real seal is available.

Componentwise hsame transport by H may rename the rows, but it preserves the displayed route rather than adding a second stream, a chosen limit, or a host completion object. The package and local name rows P, N then record that the consumer is reading the same finite packet.

Since the carrier contains no row for quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness, the separated Real route above is the whole exported completion-facing surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionLadder}^{\uparrow}$ packet is generated from DyadicRatCore tolerance requests, StreamName finite windows, RegSeqRat readback, a RealUp seal, a SeparatedCompletion witness, a CauchyCompletionUnit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4299.1, Definition 4299.2, and Definition 4299.3 seed the finite Real and Completion ladder over DyadicRatCore, StreamName, RegSeqRat, Real, SeparatedCompletion, CauchyCompletionUnit, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionLadder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, bridge checking, Lean verification, or completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window compatibility, regular readback exactness, Real-seal order, separated-completion routing, completion-unit routing, componentwise transport stability, and ledger non-escape for BEDC.Derived.CompletionLadderUp.

Chapter 4300

A Concrete Naming Certificate for $\mathsf{LocatedFormalBar}^{\uparrow}$

Following Brouwer, Troelstra–van Dalen, and Vickers, the $\mathsf{LocatedFormalBar}^{\uparrow}$ packet records a finite located bar/cover boundary for spread paths and formal-topology cover requests. It sits after the cover/refinement carrier of chapter 3967, the admissible-prefix discipline of chapter 1532, the bar-recursion route of chapter 1675, and the finite observation interface of chapter 126. The packet supplies Real and Completion consumers with a finite bar boundary without importing open-cover compactness, selected points, quotient opens, countable choice, or ambient completeness.

Definition 4300.1 (Located formal-bar carrier)**.** *A* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FormalTopology}^{\uparrow}$ *cover/refinement row, read through chapter 3967, that names the finite basic neighbourhoods and refinement requests used by the bar.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *admissible-prefix row, read through chapter 1532, that determines which finite path prefixes can enter the bar ledger.*
3. *B is the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *bar-recursion row, read through chapter 1675, that records the finite induction surface consumed by bar-cover arguments.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, read through chapter 126, that exposes the inspected prefix windows for Real and Completion routes.*
5. *Q is the located-query row: each accepted request names a finite prefix, a formal basic neighbourhood, and the bar-cover decision boundary visible at that prefix.*
6. *R is the spread-cover row, recording the finite route from an admissible prefix through the formal cover row to the bar-recursion row.*
7. *H records componentwise* hsame *transport for formal-cover rows, spread-law prefixes, bar-recursion rows, StreamName windows, located-query rows, spread-cover rows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion finite-window request through $F, S, B, W, Q, R, H, P, N$*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedFormalBar}^{\uparrow}$, $\mathsf{FormalTopology}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite path, a selected point of a cover, quotient-open equality, countable choice, host topological equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedFormalBarUp.LocatedFormalBarTasteGate single carrier alignment`

Theorem 4300.2 (Located formal-bar spread-cover route)**.** *Every accepted* $\mathsf{LocatedFormalBar}^{\uparrow}$ *carrier*

$$L = (F, S, B, W, Q, R, H, C, P, N)$$

routes a Real or Completion finite-window consumer through the same displayed spread-cover boundary:

$$F, S, B, W \longmapsto Q, R \longmapsto H, C, P, N.$$

The formal-topology cover row F, spread-law row S, Brouwer bar-induction row B, and StreamName window row W must be visible before the located-query row Q and spread-cover row R are consumed. The route exports no open-cover compactness theorem, selected infinite branch, selected cover point, quotient-open equality, countable-choice schedule, host real equality, or ambient completeness datum.

Proof. Project the carrier of Definition 4300.1. The source rows are F, S, B, W, and the located bar/cover rows are Q, R.

First read the formal-topology and spread-law surfaces. The finite cover/refinement row F supplies only displayed basic neighbourhoods, while S supplies only admissible finite prefixes.

Next pass through the bar-recursion and StreamName rows. Row B keeps the bar argument on the finite Brouwer induction surface, and W fixes the inspected finite windows used by Real and Completion consumers.

Finally replay through Q, R, H, C, P, N. The located query and spread-cover rows are transported componentwise and replayed with the same provenance and local NameCert row. Since the carrier contains no coordinate for compact open covers, selected branches, selected points, quotient opens, choice, host equality, or ambient completeness, no consumer can read those objects from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedFormalBar}^{\uparrow}$ packet is generated from FormalTopology cover/refinement rows, SpreadLaw admissible-prefix rows, BrouwerBarInduction bar-recursion rows, StreamName finite-window rows, located-query rows, spread-cover rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, seed)

Scope closed: Definition 4300.1 and Definition 4300.2 seed the finite located bar/cover boundary for Real and Completion consumers over formal covers, spread prefixes, bar recursion, finite windows, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedFormalBar}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, selected cover points, quotient-open equality, countable choice, host topological equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, formal-cover exposure, spread-law prefix exposure, bar-recursion routing, StreamName finite-window replay, located-query exactness, spread-cover exactness, transport, provenance, and local naming for BEDC.Derived.LocatedFormalBarUp.

Chapter 4301

A Concrete Naming Certificate for CauchyModulusFunctoriality$^{\uparrow}$

Following Bishop and Bridges, the CauchyModulusFunctoriality$^{\uparrow}$ packet records the finite data needed to transport displayed Cauchy moduli along a BEDC route without turning that route into a host function equality. It sits between the dyadic tolerance source DyadicRatCore$^{\uparrow}$ of chapter 291, the finite observation route StreamName$^{\uparrow}$ of chapter 126, the regular rational readback RegSeqRat$^{\uparrow}$ of chapter 301, the real-name seal Real$^{\uparrow}$ of chapter 38, the pullback discipline CauchyModulusPullback$^{\uparrow}$ of chapter 1951, and the regular stability surface RegularCauchyModulusStability$^{\uparrow}$ of chapter 1927.

Definition 4301.1 (Cauchy modulus functoriality carrier)**.** *A* CauchyModulusFunctoriality$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, R, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source Cauchy-modulus row, naming the inspected source modulus and its finite request boundary.*
2. *T is the target Cauchy-modulus row, naming the target modulus after transport along the displayed route.*
3. *R is the route-map row, recording the finite BEDC route that carries source requests to target requests without asserting equality of host functions.*
4. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, read through chapter 291, that keeps each modulus comparison on dyadic rational bounds.*
5. *W is the* StreamName$^{\uparrow}$ *finite-window row, read through chapter 126, that records the observation window used by the transported modulus request.*
6. *Q is the* RegSeqRat$^{\uparrow}$ *readback row, read through chapter 301, that turns the transported window into a regular rational comparison.*
7. *E is the* Real$^{\uparrow}$ *seal, read through chapter 38, that exposes only the terminal real-name boundary reached by the finite route.*
8. *H records componentwise* hsame *transport for source modulus, target modulus, route map, dyadic tolerance ledger, StreamName window, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a transported modulus request through* $S, T, R, D, W, Q, E, H, P, N$.

10. *P is the* Pkg *provenance over* CauchyModulusFunctoriality$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStal DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function equality, a selected limit, quotient real equality, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusFunctorialityUp.CauchyModulusFunctorialityTasteGate single carrier alignment`

Theorem 4301.2 (Cauchy modulus functoriality NameCert obligations)**.** *For every accepted* CauchyModulusFunctoriality$^{\uparrow}$ *carrier*

$$F = (S, T, R, D, W, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$ *obligations are exactly the finite route obligations listed by the carrier: source-modulus admission, target-modulus admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The route may use the sibling certificates for* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, *and* RegularCauchyModulusStability$^{\uparrow}$, *but it exports no host function equality, selected limit, quotient real equality, trichotomy, countable choice, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 4301.1. The source and target modulus rows are S, T, and the finite transport route is the displayed row R.

The numeric side is then forced through D, W, Q. The dyadic ledger D supplies the tolerance comparisons, the StreamName window W supplies the finite observation boundary, and the RegSeqRat row Q supplies the regular rational readback.

The terminal read is sealed by E. It records only the Real-facing boundary reached by the same finite route, so it cannot introduce a selected limit or quotient-real equality.

Finally transport and replay through H, C, P, N. These rows preserve the same carrier components under hsame, Cont, package provenance, and local naming. Since the carrier contains no coordinates for host function equality, selected limits, quotient real equality, trichotomy, countable choice, or ambient completeness, the listed NameCert obligations exhaust the packet. □

Closure status for NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$.

Constructive story: A CauchyModulusFunctoriality$^{\uparrow}$ packet is generated from source and target Cauchy-modulus rows, a finite route-map row, DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, seed)

Scope closed: Definition 4301.1 and Definition 4301.2 seed the finite modulus-transport certificate over DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulusPullback$^{\uparrow}$, RegularCauchyModulusStability$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CauchyModulusFunctoriality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function equality, selected limits, quotient real equality, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, route-map exposure, dyadic tolerance preservation, StreamName window preservation, RegSeqRat readback preservation, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFunctorialityUp.

The finite gauge presentation routes the metric distance surface of chapter 57 through the entourage interface of chapter 339 and the missing-symmetry boundary of chapter 2334.

Chapter 4302

A Concrete Naming Certificate for $\mathsf{GaugeSpace}^{\uparrow}$

Following Bourbaki and Willard, the $\mathsf{GaugeSpace}^{\uparrow}$ packet records a uniform structure by a finite family of gauge seminorm rows rather than by a primitive entourage filter. It sits between the metric distance certificate of chapter 57, the entourage interface of chapter 339, and the directional variant of chapter 2334. The chapter names the finite presentation used by uniform-completion consumers without importing a selected completion, quotient equality, or ambient uniformity oracle.

Definition 4302.1 (Gauge-space carrier)**.** *A* $\mathsf{GaugeSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Q, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-name carrier read through the ambient finite source rows.*
2. *Q is the finite list of gauge rows, each row naming a nonnegative comparison surface for a displayed point pair.*
3. *D is the diagonal row, stating that each displayed gauge row admits the named diagonal pair.*
4. *R is the refinement row, stating that a requested finite tolerance may be met by a displayed subfamily and smaller tolerance row.*
5. *S is the symmetry row for the uniform case; in a quasi-uniform presentation it is replaced by the directional row of chapter 2334.*
6. *H stores componentwise* hsame *transport for the point carrier, gauge list, diagonal, refinement, symmetry, continuation, provenance, and local naming rows.*
7. *C stores the* Cont *replay route from finite gauge requests to generated entourage rows.*
8. *P is the* Pkg *provenance over* $\mathsf{GaugeSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GaugeSpace}^{\uparrow}}$ *row.*

The carrier exports only the finite gauge presentation. It contains no primitive entourage filter, no selected uniform completion, no quotient equality of Cauchy data, and no ambient choice of separating gauges.

Theorem 4302.2 (Gauge-space entourage generation)**.** *For every accepted* $\mathsf{GaugeSpace}^{\uparrow}$ *carrier* $G = (X, Q, D, R, S, H, C, P, N)$*, the displayed gauge rows generate the diagonal, refinement, and symmetry surface required by the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage interface:* D *supplies diagonal admission for every named gauge row,* R *supplies finite refinement of gauge tolerances, and* S *supplies the uniform symmetry row. The generated surface is transported by* H*, replayed by* C*, packaged by* P*, and locally named by* N*, and it does not add a primitive entourage filter, selected completion point, quotient Cauchy equality, or host uniformity principle.*

Proof. Start with the finite gauge list Q. Each gauge row is a displayed comparison surface over the carrier X, so any entourage request must first name a finite subfamily of Q and a tolerance row.

The diagonal face is provided by D. It admits the named diagonal pair for each displayed gauge row and therefore gives the same diagonal surface that a metric ball entourage obtains from zero self-distance in chapter 57.

Refinement is provided by R. A request for a smaller entourage is answered by a finite subfamily and smaller tolerance row, then replayed through C as the generated entourage route. This is the finite counterpart of the refinement surface in chapter 339.

The symmetry row S closes the uniform case. If the row is replaced by a directional row, the same carrier becomes a quasi-uniform presentation of the kind recorded in chapter 2334; with S present, the route is the ordinary uniform gauge surface.

Finally, H, P, N preserve the same generated route under transport, package provenance, and local naming. No step introduces a primitive entourage filter, selected completion, quotient Cauchy equality, or ambient uniformity oracle, so the generated surface is exactly the finite BEDC gauge presentation. □

Theorem 4302.3 (Gauge-space L10 metric completion dependency)**.** *Every L10 consumer of an accepted* $\mathsf{GaugeSpace}^{\uparrow}$ *carrier* $G = (X, Q, D, R, S, H, C, P, N)$ *factors its gauge entourages through the displayed* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{UniformSpace}^{\uparrow}$ *rows before reaching completion-facing* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. The dependency route first reads the finite gauge list* Q*, diagonal row* D*, refinement row* R*, and symmetry row* S*; then it replays those rows as generated uniform entourages; only then may finite windows, regular rational readback, dyadic tolerance ledgers, and Real seals be consumed through chapter 126, chapter 301, chapter 291, and chapter 38.*

Thus the gauge carrier enters the L10 dependency lattice through chapter 57 and chapter 339. It does not export a primitive entourage filter, selected completion point, quotient Cauchy equality, host uniformity principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Unpack the carrier of Definition 4302.1. The only entourage-generating analytic rows are the finite gauge list Q, the diagonal row D, the refinement row R, and the symmetry row S; the rows H, C, P, N transport, replay, package, and name that finite presentation.

Apply Definition 4302.2. It turns the displayed gauge data into the uniform entourage surface and identifies the metric-facing route through the same finite diagonal, refinement, and symmetry rows. Consequently a completion consumer cannot read StreamName windows or regular rational data before the gauge route has passed through the $\mathsf{Met}^{\uparrow}$ and $\mathsf{UniformSpace}^{\uparrow}$ interfaces.

The L10 completion-facing rows are then downstream. StreamName supplies finite observation windows, RegSeqRat supplies readback, DyadicRatCore supplies tolerance ledgers, and Real supplies the terminal seal. Since each of these rows is reached after the generated entourage route, the gauge packet remains a finite dependency surface rather than a primitive completion or quotient-equality principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaugeSpace}^\uparrow}$.

Constructive story: A $\mathsf{GaugeSpace}^\uparrow$ packet is generated from a displayed point carrier, finite gauge rows, diagonal admission, refinement rows, a symmetry row, componentwise transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaugeSpace}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 4302.1 and Definition 4302.2 seed the finite gauge-row presentation of diagonal, refinement, and symmetry entourage generation.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaugeSpace}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a selected completion, quotient Cauchy equality, a primitive entourage filter, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal generation, refinement generation, symmetry transport, and standard uniform-space handoff for the displayed finite packet.

Every $\mathsf{GaugeSpace}^\uparrow$ packet consumes the $\mathsf{Met}^\uparrow$ distance-witness rows of chapter 57 together with the $\mathsf{UniformSpace}^\uparrow$ entourage rows of chapter 339. The dependency route presents gauge families as finite NameCert data: each gauge observation is read as a metric-controlled distance row, each comparison is replayed through a uniform entourage row, and the visible BHist, hsame, Cont, Pkg, and NameCert rows keep the gauge carrier from exporting an ambient topology or quotient uniformity not displayed by those siblings. Every $\mathsf{GaugeSpace}^\uparrow$ read first receives metric distance rows from chapter 57, then packages the generated entourage rows consumed by chapter 339. The same carrier boundary is compared against the missing-symmetry surface of chapter 2334: with the symmetry row present it is a uniform gauge presentation, and without that row it remains a directional quasi-uniform presentation rather than a primitive entourage filter.

Theorem 4302.4 (Gauge-space uniformity dependency route)**.** *Every accepted* $\mathsf{GaugeSpace}^\uparrow$ *packet consumes the* $\mathsf{Met}^\uparrow$ *distance rows,* $\mathsf{UniformSpace}^\uparrow$ *entourage rows, and* $\mathsf{QuasiUniformSpace}^\uparrow$ *directional rows before it exports finite gauge uniformity. The dependency is substantive: the gauge-family comparison first reads the* $\mathsf{UniformSpace}^\uparrow$ *local-base row through chapter 339 and the* $\mathsf{Met}^\uparrow$ *distance-window row through chapter 57, then exposes gauge-family comparisons only after those sibling rows have fixed the uniformity boundary. In carrier notation, the point row* X*, gauge list* Q*, diagonal row* D*, refinement row* R*, symmetry or directional row* S*, transport row* H*, continuation row* C*, provenance row* P*, and local naming row* N *must all be displayed before any uniform-completion consumer may use the generated gauge surface.*

Proof. Project the carrier of Definition 4302.1. The gauge rows Q do not stand alone: the carrier description places them beside metric distance input, generated uniform entourage data, and the directional quasi-uniform boundary.

Apply Definition 4302.2. The diagonal and refinement rows D, R are the finite entourage-generation surface, while S distinguishes the ordinary $\mathsf{UniformSpace}^\uparrow$ route from the $\mathsf{QuasiUniformSpace}^\uparrow$ directional route.

Finally use Definition 4302.6. It identifies the generated gauge rows as the sibling handoff to the uniform and quasi-uniform interfaces, and the structural rows H, C, P, N preserve that handoff without adding a primitive entourage filter, selected completion, quotient equality, or ambient uniformity oracle. □

Theorem 4302.5 (Gauge-space UniformSpace local-base dependency)**.** *Every accepted* $\mathsf{GaugeSpace}^\uparrow$ *packet consumes the* $\mathsf{UniformSpace}^\uparrow$ *entourage rows of chapter 339 before its gauge seminorm family is read as a local base. In carrier notation, the point row* X*, finite gauge list* Q*, diagonal row* D*, refinement row* R*, symmetry row* S*, transport row* H*, replay row* C*, provenance row* P*, and local naming row* N *are not enough by themselves to export a uniform local base; the generated*

entourage surface must first display finite entourage, classifier, transport, replay, provenance, and local naming rows from the uniform-space sibling.

Proof. Project the carrier of Definition 4302.1. The gauge family Q is a finite list of comparison rows, but it becomes a local base only after diagonal, refinement, and symmetry data have been read as entourage-generation data.

Apply Definition 4302.2. That theorem reads D, R, S as the generated uniform interface and routes the result through the $\mathsf{UniformSpace}^{\uparrow}$ sibling. The route is therefore an entourage route before it is a local-base route.

Now use Definition 4302.6. It identifies the generated gauge rows with the UniformSpace entourage surface and keeps the quasi-uniform alternative visible when the symmetry row is absent. Thus the accepted packet consumes finite entourage rows, classifier rows, componentwise transport, replay, provenance, and local naming before any gauge seminorm family is read as a uniform local base. □

Theorem 4302.6 (GaugeSpace UniformSpace sibling anchor)**.** *Every accepted* $\mathsf{GaugeSpace}^{\uparrow}$ *packet exposes its finite gauge rows as a generated* $\mathsf{UniformSpace}^{\uparrow}$ *entourage surface, and exposes the missing-symmetry boundary as the corresponding* $\mathsf{QuasiUniformSpace}^{\uparrow}$ *directional surface. The carrier of Definition 4302.1 supplies the point row X, gauge rows Q, diagonal row D, refinement row R, optional symmetry row S, transport row H, continuation row C, provenance row P, and local naming row N. The entourage-generation theorem Definition 4302.2 reads D, R, S as the uniform interface of chapter 339; when S is absent or replaced by a directional row, the same generated gauge surface lands at the quasi-uniform boundary of chapter 2334.*

Proof. Project the accepted carrier through Definition 4302.1. The gauge list Q is finite, and every generated entourage request must pass through the displayed diagonal and refinement rows D and R.

Apply Definition 4302.2. With the symmetry row S present, the generated route is exactly the $\mathsf{UniformSpace}^{\uparrow}$ entourage surface: diagonal admission, finite refinement, symmetry transport, continuation replay, provenance, and local naming.

If the symmetry row is not present as a uniform row, the same carrier data still gives the directional diagonal-and-refinement route. That is the $\mathsf{QuasiUniformSpace}^{\uparrow}$ boundary, because the omitted symmetry coordinate is precisely the distinction between the uniform sibling and the quasi-uniform sibling. □

Chapter 4303

A Concrete Naming Certificate for $\mathsf{RealLimitStability}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RealLimitStability}^{\uparrow}$ packet records the finite data by which two displayed terminal $\mathsf{Real}^{\uparrow}$ limit seals remain stable under shared regular-Cauchy evidence. It sits after the terminal Real seal of chapter 38, the regular rational readback discipline of chapter 301, the finite stream-name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the completion witness of chapter 3579. The packet names stability of displayed limit seals; it is not quotient stream equality, selected host limits, trichotomy, countable choice, or ambient Real completeness.

Definition 4303.1 (Real limit stability carrier)**.** *A* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$$

with the following displayed rows:

1. L_0 *and* L_1 *are the two limit-presentation rows whose terminal seals are compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both presentations are inspected.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the shared window.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the common dyadic budgets used by the comparison.*
5. E_0 *and* E_1 *are the paired terminal* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after* L_0, L_1, W, R, D *have been displayed.*
6. H *records componentwise* hsame *transport for both limit presentations, the shared window, readback, tolerance ledger, paired Real seals, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a stability request through* $L_0, L_1, W, R, D, E_0, E_1, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RealLimitStability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCompletenessWitness}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows and their replay path. It has no coordinate for quotient stream equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate single carrier alignment`

Theorem 4303.2 (Real limit stability NameCert obligations)**.** *For every accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier* $S = (L_0, L_1, W, R, D, E_0, E_1, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite packet, classifier stability under componentwise* hsame*, shared-window readback through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic tolerance nonescape through* $\mathsf{DyadicRatCore}^{\uparrow}$*, paired terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 4303.1. The accepted packet exposes exactly two presentation rows, one shared window, one regular rational readback, one dyadic tolerance ledger, two terminal Real seals, transport, replay, provenance, and local naming.

Carrier admission and classifier stability are componentwise. The transport row H moves the two presentations, shared evidence, paired seals, and certificate rows together, while the replay row C preserves the same order of inspection for every stability request.

The provenance row P cites only the listed sibling surfaces and the local row N names the packet. Since the carrier contains no quotient equality, selected host limit, trichotomy oracle, choice schedule, ambient completeness coordinate, Lean certificate, or bridge row, the listed obligations exhaust the NameCert surface. □

Theorem 4303.3 (Real limit stability seal stability)**.** *Every comparison of the paired terminal seals in an accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *packet factors through the displayed route*

$$L_0, L_1 \longmapsto W \longmapsto R \longmapsto D \longmapsto E_0, E_1.$$

Thus the two $\mathsf{Real}^{\uparrow}$ *seals may be reused only under the shared finite* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger displayed by the packet.*

Proof. Start with a consumer that reads the paired Real seals. The only Real-facing terminal coordinates in the carrier are E_0 and E_1, and both appear after the two presentation rows, shared finite window, regular rational readback, and dyadic ledger.

The consumer must therefore pass through W, R, D before either seal can be used. These rows give the finite observations, regular Cauchy readback, and common tolerance budget that make the two terminal seals comparable as displayed BEDC data.

Finally read H, C, P, N. These rows transport, replay, source, and name the same route, so they preserve seal stability without adding a selected host limit or quotient stream equality. □

Theorem 4303.4 (Real limit stability nonescape)**.** *No accepted* $\mathsf{RealLimitStability}^{\uparrow}$ *carrier exports quotient stream equality, a selected host limit, trichotomy, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, a detached Real seal, Lean verification, or bridge checking. Any downstream completion consumer receives only the two limit presentations, shared window, regular readback, dyadic ledger, paired seals, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row.*

Proof. Inspect the carrier inventory. The tuple contains no coordinate for quotient equality, selected host limits, trichotomy, choice, ambient completeness, detached seal data, Lean verification, or a bridge certificate.

The paired seals E_0, E_1 are downstream of the shared witness rows W, R, D. Hence a completion consumer can reuse the seals only through the displayed finite evidence and not through an external selected limit.

Transport, replay, provenance, and local naming preserve the same boundary. The rows H, C, P, N move and name only the finite packet, so every exported object remains inside the displayed Real-limit-stability surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLimitStability}^{\uparrow}$ packet is generated from two limit-presentation rows, a shared StreamName window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, paired RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4303.1, Definition 4303.2, Definition 4303.3, and Definition 4303.4 seed the finite stability packet for paired terminal Real limit seals under shared StreamName, RegSeqRat, DyadicRatCore, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLimitStability}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealLimitStabilityUp.RealLimitStabilityTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, shared-window readback, dyadic ledger nonescape, paired Real-seal stability, transport, replay, provenance, and local naming for BEDC.Derived.RealLimitStabilityUp.

Chapter 4304

A Concrete Naming Certificate for $\mathsf{DecimalIntervalReal}^{\uparrow}$

Bishop–Bridges constructive analysis uses rational and decimal brackets as finite evidence before a real name is treated as complete. The $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet records this decimal-bracket carrier as BEDC NameCert data. It sits after the decimal prefix surface of chapter 1417, the finite decimal stream of chapter 4191, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval source of chapter 1709. The packet gives L10 Real and Completion consumers a finite decimal interval carrier without quotient reals, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4304.1 (Decimal interval real carrier)**.** *A* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower finite decimal-prefix row, read through* $\mathsf{DecimalExpansion}^{\uparrow}$ *and the finite* $\mathsf{DecimalStream}^{\uparrow}$ *schedule.*
2. *B is the upper finite decimal-prefix row, read through the same decimal surfaces and carrying the advertised bracket side.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating decimal place values into dyadic comparison margins.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for endpoint-prefix reads, bracket comparison, and downstream replay.*
5. *Q is the bracket-order row, recording that the lower decimal prefix and upper decimal prefix are compared only through the displayed tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint windows, decimal prefix rows, dyadic ledger, and bracket-order row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback has been displayed.*
8. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *handoff row that consumes the same endpoint, tolerance, readback, and seal rows as a located interval boundary.*

9. *H records componentwise* hsame *transport for endpoint prefixes, tolerance rows, windows, bracket order, regular readback, real seal, located interval handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a decimal-bracket request through endpoint prefixes, dyadic comparison, finite windows, regular readback, real sealing, and located interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DecimalIntervalReal}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for a quotient real, a trichotomy decision, a selected decimal representative, host endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier`
Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealTasteGate single carrier alignment`

Theorem 4304.2 (Decimal interval real NameCert obligations)**.** *For every accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower decimal prefix, upper decimal prefix, dyadic tolerance ledger, stream windows, bracket-order row, regular rational readback, real seal, located interval handoff, transport, replay, provenance, and local naming rows;*
2. *decimal-prefix visibility, requiring endpoint reads to enter through A and B before any bracket comparison is inspected;*
3. *tolerance discipline, requiring the place-value comparison to use the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D and no hidden real-order oracle;*
4. *window discipline, requiring each endpoint and comparison read to be scheduled by the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback order, requiring R to consume A, B, D, W, Q before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *located-interval handoff, requiring L to consume the same endpoint, tolerance, readback, and seal route rather than a host interval;*
7. *componentwise* hsame *transport and displayed* Cont *replay for exactly the rows carried by H and C;*
8. *ledger non-escape, excluding quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4357.1. The projection exposes exactly the two decimal endpoint-prefix rows, the dyadic tolerance ledger, the finite window schedule, the bracket-order row, the regular readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming.

The endpoint reads are visible before comparison. Rows A and B carry the finite decimal prefixes, while D supplies the dyadic margin used to compare their place-value readings. The row W schedules the inspected prefixes and comparison windows, so the bracket-order row Q has no access to a hidden real-order decision.

The real-facing route is downstream of the finite bracket data. Row R is read only after A, B, D, W, Q have been displayed, and row E seals the result only after that regular rational readback.

Finally, the located interval handoff L consumes the same displayed route, while H, C, P, N transport, replay, source, and name the packet. Since no remaining coordinate stores quotient equality, trichotomy, a selected decimal representative, host endpoint equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier semantic name certificate`

Theorem 4304.3 (Decimal interval real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow}.$$

Equivalently, a downstream Real and Completion consumer receives the two decimal endpoint-prefix rows, the dyadic comparison ledger, the finite window row, the regular rational readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming before it can use the bracket.

Proof. Start with an accepted carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$. The only real-facing row is E, and the only located-interval row is L.

The definition places E after the endpoint-prefix rows A, B, the dyadic tolerance ledger D, the window schedule W, the bracket-order row Q, and the regular rational readback R. Thus a $\mathsf{Real}^{\uparrow}$ consumer must first inspect the decimal, dyadic, window, and readback rows.

The located interval handoff is not an alternate route. Row L consumes the same A, B, D, W, Q, R, E path, and componentwise transport through H preserves that path under hsame.

The replay, provenance, and local naming rows only record the displayed carrier. They do not add quotient reals, trichotomy, selected representatives, host endpoint equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed route is the whole exported handoff. □

Falsifiable prediction: Every accepted $\mathsf{DecimalIntervalReal}^{\uparrow}$ read displays two finite decimal-prefix rows, one dyadic comparison ledger, one finite StreamName window row, one RegSeqRat readback row, one RealUp seal row, one LocatedRealIntervalUp handoff row, and local transport, replay, provenance, and naming rows before it is consumed as a decimal bracket.

Independence witness: decimalexpansion, decimal-stream, dyadicratcore, streamname, regseqrat, real, located-real-interval: $\mathsf{DecimalIntervalReal}^{\uparrow}$ is not bijective to any listed sibling because it binds two ordered decimal endpoint-prefix rows, one dyadic comparison ledger, finite observation windows, regular rational readback, RealUp sealing, located interval handoff, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet is generated from lower and upper decimal-prefix rows, DecimalStream windows, a DyadicRatCore comparison ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, a LocatedRealIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4357.1, Definition 4357.2, and Definition 4357.3 seed the finite decimal-bracket route over DecimalExpansionUp, DecimalStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealIntervalUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DecimalIntervalRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix visibility, dyadic tolerance discipline, finite-window discipline, regular-readback order, Real seal handoff, located interval handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.DecimalIntervalRealUp.

Chapter 4305

A Concrete Naming Certificate for $\mathsf{AttouchWetsTopology}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet records local Hausdorff-window convergence for closed metric figures as finite BEDC NameCert data over the metric-completion and L10 real-name surface. It sits after the metric carrier of chapter 57, the Hausdorff-distance surface of chapter 3653, the terminal real seal of chapter 38, the regular rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the finite observation topology; it does not import host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4305.1 (Attouch-Wets topology carrier)**.** *An* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose metric, point windows, and bounded-window radii are the only ambient metric data available to the packet.*
2. *H is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *local-window row, read only on displayed bounded metric windows rather than on an ambient hyperspace.*
3. *D is the dyadic tolerance ledger for window radius, Hausdorff comparison margin, and late-window precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, bounded windows, comparison stages, and requested precision cells.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Hausdorff-window comparisons obtained from M, H, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *Q records componentwise* hsame *transport for metric source rows, Hausdorff-window rows, dyadic ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming.*
8. *T records displayed* Cont *replay from an Attouch–Wets observation request through M, H, D, W, R, E.*

9. *C is the local bounded-window classifier row comparing packets only through the displayed metric source, Hausdorff-window, tolerance, stream, readback, seal, transport, replay, provenance, and naming coordinates.*

10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *row.*

The classifier has no coordinate for global closed-set topology, open-cover compactness, selected nearest points, quotient hyperspace equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AttouchWetsTopologyUp.AttouchWetsTopologyTasteGate single carrier alignment`

Theorem 4305.2 (Attouch-Wets topology NameCert obligation surface)**.** *For every accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *expose the* $\mathsf{Met}^{\uparrow}$ *source row M before any local Hausdorff comparison is read;*

2. *display the* $\mathsf{HausdorffDistance}^{\uparrow}$ *window row H only on bounded metric windows named by the packet;*

3. *bind all numerical precision to the dyadic tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

4. *preserve the same finite route by componentwise* hsame *transport Q and displayed* Cont *replay T;*

5. *source and name the same packet through the bounded-window classifier C,* Pkg *provenance P, and the local* NameCert *row N.*

No obligation row is stored for a host hyperspace topology, arbitrary closed-set compactness, selected representatives, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4358.1. The metric row M is the only source of ambient distance data, and the Hausdorff row H is read only after the bounded window has been displayed.

The numerical part is downstream of that window. The tolerance ledger D, finite window row W, regular readback R, and real seal E are the only precision and value rows available to an observation request.

Transport and replay keep the same route. Rows Q and T preserve the displayed metric, Hausdorff, tolerance, window, readback, and seal data, while C, P, N classify, source, and name the finite packet.

Since the carrier contains none of the excluded topology, compactness, selection, quotient, choice, or ambient-completeness coordinates, the displayed obligations exhaust the local NameCert surface. □

Theorem 4305.3 (Attouch-Wets local Hausdorff window)**.** *Every topology-facing read exported by an accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{HausdorffDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}.$$

Thus a downstream metric-completion consumer receives the displayed metric source, bounded local Hausdorff window, finite observation schedule, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming before it can use Attouch–Wets convergence data.

Proof. Start with a topology-facing read of the accepted carrier. The only available metric source is M, and the only Hausdorff comparison row is the local-window row H.

The observation is then forced through D, W, R, E. The dyadic ledger fixes the requested tolerance, the stream row displays the inspected bounded windows, the regular rational row records the readback, and the real row seals the reported Hausdorff-window value.

Finally Q, T, C, P, N transport, replay, classify, source, and name that same finite route. Because no row supplies host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported topology-facing read factors through the displayed local Hausdorff window. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, a HausdorffDistance bounded-window row, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, a bounded-window classifier row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4358.1, Definition 4358.2, and Definition 4358.3 seed the local Hausdorff-window topology route over MetricSpaceUp, HausdorffDistanceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AttouchWetsTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Hausdorff-window exposure, finite-window scheduling, regular readback, Real sealing, transport, replay, classifier exactness, provenance, and local naming for BEDC.Derived.AttouchWetsTopologyUp.

Chapter 4306

A Concrete Naming Certificate for $\mathsf{CantorIntersectionMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet records the metric nested-closed-set intersection route as finite BEDC NameCert data over the L10 completion surface. It sits after the complete-metric source of chapter 128, the metric-completion route of chapter 1506, the nested closed-ball packet of chapter 3526, the Cantor complete metric route of chapter 2199, the Cauchy-filter interface of chapter 333, and the L10 real-name route through chapter 291, chapter 126, chapter 301, and chapter 38. The metric packet is the metric-completion specialization of the finite Cantor-intersection route in chapter 1421 and the theorem-facing intersection route in chapter 2318; it consumes their nested-window and diagonal-readback rows before exposing complete-metric service. The packet names a nested metric intersection handoff; it does not assert open-cover compactness, arbitrary closed-set compactness, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4306.1 (CantorIntersectionMetric carrier)**.** *A* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence service are the only complete-metric data available to the packet.*
2. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row naming the surrounding completion request before any intersection point is read.*
3. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row, recording the descending closed metric ball spine and its finite containment witnesses.*
4. *K is the* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *row, used only as the displayed nested closed-ball completion handoff.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested metric spine and read only through the displayed filter route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, storing the shrink schedule, late-radius comparisons, and requested precision bounds.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, radii, and containment data.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of the displayed center schedule.*
9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, K, F, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for the complete-metric source, completion route, nested-ball row, Cantor handoff, filter, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a nested metric intersection request through $M, E, B, K, F, D, S, R, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, B, K, F, D, S, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary closed-set compactness, quotient completion equality, a selected limit, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionMetricUp.CantorIntersectionMetricTasteGate single carrier alignment`

Theorem 4306.2 (CantorIntersectionMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the complete-metric and metric-completion source rows M, E before any intersection witness is read;*
2. *display the nested metric route through B, K, F, namely the nested closed-ball spine, Cantor complete-metric handoff, and generated Cauchy-filter row;*
3. *bind all numerical readback to the L10 rows D, S, R, L, namely the dyadic tolerance ledger, finite stream windows, regular rational handoff, and terminal real seal;*
4. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
5. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row is stored for open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, trichotomy, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4359.1. The rows M, E expose the metric and completion sources before the packet can read an intersection point.

The nested metric content is then displayed through B, K, F. The nested closed-ball row supplies the descending closed metric spine, the Cantor complete-metric row supplies the existing handoff, and the Cauchy-filter row records the generated filter route.

The L10 readback is downstream. The dyadic ledger D, stream windows S, regular rational row R, and real seal L are the only numerical coordinates available before a completion-facing consumer receives a named output.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains none of the excluded compactness, quotient, selected-limit, trichotomy, or ambient-completeness coordinates, the five displayed obligations exhaust the local NameCert surface. □

Theorem 4306.3 (CantorIntersectionMetric complete handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier factors through the displayed nested metric route*

$\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, S

Thus a consumer receives the complete-metric service, completion request, nested closed-ball spine, Cantor complete-metric handoff, Cauchy-filter row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows before it may use the intersection output. The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness principle, open-cover compactness theorem, arbitrary closed-set compactness, quotient stream equality, or selected limit.*

Proof. Begin with a completion-facing read of the accepted carrier. The first available rows are M and E, so the read is grounded in the displayed complete-metric service and the surrounding metric-completion request.

Next pass through the nested route B, K, F. The descending closed-ball spine is read through the $\mathsf{NestedClosedBall}^{\uparrow}$ row, the existing Cantor complete-metric handoff is read through K, and the generated filter is read through F.

The numerical output is then forced through the L10 readback rows D, S, R, L. The dyadic ledger supplies tolerance control, the stream row supplies finite windows, the regular rational row supplies readback, and the terminal real seal is reached only after those rows are present.

The structural rows H, C, P, N preserve the same route for downstream consumers. Because no row supplies full real completeness, open-cover compactness, arbitrary closed-set compactness, quotient stream equality, or a selected limit, every completion-facing read factors through the displayed nested metric route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, a MetricCompletion request row, a NestedClosedBall spine, a CantorCompleteMetric handoff, a CauchyFilter row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4359.1, Definition 4359.2, and Definition 4359.3 seed the metric nested-closed-set intersection route over displayed completion, nested-ball, Cantor handoff, filter, dyadic, stream, regular readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested metric inclusion, Cantor handoff, Cauchy-filter generation, dyadic shrink exactness, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionMetricUp.

Chapter 4307

A Concrete Naming Certificate for $\mathsf{TightRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{TightRealInterval}^{\uparrow}$ packet names a finite lower-upper real interval by displaying both endpoint names and every rational cell through which those endpoints are inspected. It sits downstream of the located endpoint surface of chapter 394 and the dyadic refinement surface of chapter 306: consumers may read a tight interval only after the lower endpoint, upper endpoint, rational cell ledger, dyadic refinement ledger, stream windows, regular rational readbacks, and Real seal rows have all been displayed.

Definition 4307.1 (TightRealInterval carrier)**.** *A* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, U, Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row and U is the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger recording the finite rational windows that enclose the endpoint reads and witness lower-before-upper compatibility.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to the rational cells in Q.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the endpoint observations and interval cells inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and cell refinements.*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached from L, U, S, R, Q, D.*
7. *H records componentwise* hsame *transport for endpoints, rational cells, dyadic refinements, stream windows, readbacks, Real seals, replay, provenance, and the local name.*
8. *C records displayed* Cont *replay through $L, U, Q, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host interval equality, chosen endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.TightRealIntervalUp.TightRealIntervalUp`
Lean4 ✓ `BEDC.Derived.TightRealIntervalUp.TasteGate.TightRealIntervalTasteGate single carrier alignment`

Theorem 4307.2 (TightRealInterval NameCert obligations)**.** *For an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier* $T = (L, U, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, rational and dyadic interval ledgers, finite stream windows, regular rational readbacks, paired Real seal rows, componentwise transport, continuation replay, package provenance, and local naming; endpoint-order visibility through* Q*; dyadic refinement discipline through* D*; stream-readback exactness through* S, R*; and ledger non-escape from host intervals, selected endpoints, quotient real equality, choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 4360.1. The lower and upper endpoint rows, the rational cell ledger, the dyadic refinement ledger, finite windows, regular readbacks, Real seals, transport rows, replay rows, provenance row, and local naming row are all visible before any interval consumer can read the packet.

Endpoint-order visibility is supplied by Q. Since each dyadic refinement in D is attached to a rational cell, the refinement discipline cannot detach from the endpoint ledger.

Stream-readback exactness is then read through S and R. The rows H, C, P, N transport, replay, source, and name the same finite route, while the excluded interval and completeness data have no carrier coordinate. □

Theorem 4307.3 (TightRealInterval Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

The read exposes lower and upper endpoint windows together with the rational and dyadic cell ledgers that bind them. It does not expose a host real interval, a selected endpoint value, quotient real equality, countable choice, trichotomy, or ambient real completeness.

Proof. Begin with the accepted carrier. The only seal-facing row is E, and it is displayed after L, U, Q, D, S, R in the finite packet.

Apply Definition 4360.2. Endpoint-order visibility forces the Real read to keep the rational cell ledger Q, dyadic refinement discipline forces any dyadic window through D, and stream-readback exactness forces finite observations through S, R.

Finally route through H, C, P, N. These rows preserve and name the same packet, so no host interval, endpoint selector, quotient equality, choice schedule, trichotomy row, or ambient completeness principle can enter the Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TightRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, paired $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4360.1, Definition 4360.2, and Definition 4360.3 seed the finite lower-upper Real interval packet over located endpoints, rational cells, dyadic refinements, stream windows, regular readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Scoped closure requires a checked carrier-obligation package and a public consumer route for the same finite lower-upper interval packet.

Chapter 4308

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$

Following Bishop-style compact metric uniform continuity, the $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet records the finite handoff that transfers compact finite-net and pointwise continuity data into a uniform modulus row. It consumes the finite compact source of chapter 60, the compact metric net surface of chapter 129, the continuity interface of chapter 391, the finite stream and regular readback rows of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite modulus-transfer NameCert surface, not an open-cover compactness proof or a host modulus selector.

Definition 4308.1 (Compact uniform modulus transfer carrier)**.** *A* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, U, S, R, D, F, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row.*
2. *G is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric coverage row for the same finite centers and source distance witnesses.*
3. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise-to-uniform continuity row consumed by the transfer.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected finite centers, tolerance requests, and pointwise-radius reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for finite radius comparison and shrinkage.*
7. *F is the modulus-transfer ledger that records the finite lower-bound handoff into the downstream uniform-modulus consumer row.*
8. *H records componentwise* hsame *transport for compact, compact-metric, uniform-continuity, stream, readback, dyadic, transfer, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay through $K, G, U, S, R, D, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate`
`single carrier alignment`

Theorem 4308.2 (Compact uniform modulus transfer obligations)**.** *For an accepted* CompactUniformModulusTrans *carrier $M = (K, G, U, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net row, compact-metric coverage row, uniform-continuity row, stream windows, regular rational readbacks, dyadic tolerance ledger, modulus-transfer ledger, componentwise transport, continuation replay, package provenance, and local naming; source lock between K, G, U; finite-window exactness through S, R; dyadic tolerance exactness through D; and non-escape from open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient equality, choice, and ambient completeness.*

Proof. Project the carrier of Definition 4361.1. The compact row, compact-metric row, continuity row, stream windows, readbacks, dyadic tolerance ledger, transfer ledger, transport rows, replay rows, provenance row, and local name are displayed as one finite packet.

The source lock is the shared read of K, G, U. A transfer consumer cannot use the compact finite-net row without the compact-metric distance surface and the continuity row that supplies pointwise radius data.

Finite-window and dyadic exactness are read through S, R, D. The ledger F can therefore transfer only the displayed finite lower-bound data, and H, C, P, N preserve that route without adding any excluded compactness or choice principle. □

Theorem 4308.3 (Compact uniform modulus transfer handoff)**.** *Every accepted* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *packet hands a downstream uniform-modulus consumer only the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont}$$

The handoff records compact finite-net coverage, pointwise continuity radius reads, finite stream windows, regular rational readbacks, dyadic tolerance comparison, and the transfer ledger. It exports no open-cover compactness theorem, selected net center, selected Lebesgue number, host modulus function, quotient equality, countable choice, or ambient completeness.

Proof. Start from the transfer obligations of Definition 4361.2. They identify the compact, compact-metric, continuity, stream, readback, dyadic, and transfer rows as the only NameCert-readable data.

Route the downstream consumer through F. The ledger can be consumed only after K, G, U, S, R, D have supplied the finite-net, pointwise-radius, window, readback, and tolerance rows.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since the excluded compactness, selection, equality, choice, and completeness data have no coordinate in the carrier, the downstream handoff is exactly the displayed finite modulus-transfer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet is generated from compact finite-net rows, compact-metric coverage rows, uniform-continuity rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a modulus-transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, seed)

Scope closed: Definition 4361.1, Definition 4361.2, and Definition 4361.3 seed the finite compact-net modulus-transfer packet for uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Upgrade path: Scoped closure requires a checked transfer-obligation package and a public uniform-modulus consumer route for the same finite packet.

Chapter 4309

A Concrete Naming Certificate for $\mathsf{RealIntervalApartness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntervalApartness}^{\uparrow}$ packet records located separation of two real intervals by a displayed positive dyadic gap rather than by a trichotomy decision on completed real endpoints. It sits after the terminal real-name seal of chapter 38, the located endpoint interval surface of chapter 1709, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the separability boundary of chapter 1809. The packet names interval separation only through visible endpoint rows, gap rows, transport, replay, provenance, and local naming.

Definition 4309.1 (RealIntervalApartness carrier)**.** *A* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are the two* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint packets whose lower and upper endpoint rows are exposed before any apartness read.*
2. *L, U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *seal rows for I, and L', U' are the corresponding* $\mathsf{Real}^{\uparrow}$ *seal rows for J.*
3. *G is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness, attached either to the displayed comparison $U + G \leq L'$ or to the displayed comparison $U' + G \leq L$.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the endpoint observations and gap inspections used by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and dyadic gap checks.*
6. *E records the* $\mathsf{RealSeparability}^{\uparrow}$ *boundary row witnessing that the two exposed endpoint intervals are separated by the displayed dyadic gap.*
7. *H records componentwise* hsame *transport for intervals, endpoint seals, dyadic gap rows, finite windows, readbacks, separability rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay through $I, J, L, U, L', U', G, S, R, E, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient real equality, a selected endpoint representative, a host interval order, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 4309.2 (RealIntervalApartness NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *admit the two located real interval packets I, J and expose their lower and upper endpoint rows through L, U, L', U';*
2. *expose the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness G and the side on which the gap separates the displayed endpoint rows;*
3. *bind every endpoint and gap inspection to the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, R;*
4. *preserve the interval, endpoint, gap, readback, and separability rows by componentwise* hsame *transport through H;*
5. *replay the same finite route through* Cont *row C, source it by* Pkg *provenance P, and name it by the local* NameCert *row N.*

No obligation row is stored for real trichotomy, quotient equality of completed real names, selected endpoint representatives, host interval order, or ambient real completeness.

Proof. Project the carrier of Definition 4362.1. The two located interval packets and their endpoint seals are the only interval data exposed to a consumer.

The separation datum is the row G. Positivity is read as a dyadic-row property, and the side of separation is part of the displayed ledger; no trichotomy comparison is consulted.

The numerical readback is forced through S and R, then through the separability row E. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since none of the excluded host-order, quotient-equality, representative-selection, trichotomy, or completeness data occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealIntervalApartnessUp.TasteGate.RealIntervalApartnessTasteGate single carrier alignment`

Theorem 4309.3 (RealIntervalApartness separability route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, hsame, Cont, Pkg, Na

The route exports located interval separation only as a positive dyadic gap between displayed endpoint rows. It exports no real trichotomy, quotient real equality, host interval order, selected endpoint representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the NameCert obligations of Definition 4362.2. They force the two located interval packets and endpoint seals to be visible before a consumer may read an apartness claim.

Next read the dyadic gap row and its finite endpoint inspections through the stream and regular-rational rows. The separability row is therefore attached to the displayed positive gap, not to a host comparison of completed reals.

Finally route through transport, replay, provenance, and local naming. These rows preserve the same finite packet, so the consumer receives located interval apartness without importing any excluded trichotomy, quotient, selection, host-order, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalApartness}^{\uparrow}$ packet is generated from two LocatedRealInterval endpoint packets, Real endpoint seal rows, a positive DyadicRatCore gap witness, finite StreamName windows, RegSeqRat readback, a RealSeparability boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4362.1, Definition 4362.2, and Definition 4362.3 seed the located interval separation packet over displayed endpoint, dyadic gap, finite-window, regular-readback, Real seal, separability, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient real equality, selected endpoint representatives, host interval order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, positive-gap exposure, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.RealIntervalApartnessUp.

Chapter 4310

A Concrete Naming Certificate for $\mathsf{PartitionBasinPolynomial}^{\uparrow}$

Following finite enumerative combinatorics, the $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ packet records the partition-basin product formula of Definition 9462.1736 as displayed NameCert data over block sizes, legal decorations, defect exponents, incidence totals, transport, replay, provenance, and local naming. It sits after the partition carrier of chapter 201, the defect-flow partition skeleton of chapter 4236, and the partition-core incidence certificate of chapter 4214. The packet names a finite polynomial enumeration surface; it is not a biological stochastic model, a host generating-function oracle, or a quotient-level semantic identification.

Definition 4310.1 (PartitionBasinPolynomial carrier)**.** *A* $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite block-count row for the partition core, and S is the finite block-size ledger $(s_1, \ldots, s_m)$.*
2. *D is the legal-decoration row, recording for each coordinate of block i the admissible subset of the other $m-1$ rows and the private-coordinate exclusion.*
3. *E is the defect-exponent ledger that assigns exponent weight q^r to a coordinate receiving r additional rows.*
4. *F is the product-formula row*
$$\prod_{i=1}^{m} \left[(1+q)^{(m-1)s_i} - \left((1+q)^{m-1} - 1 \right)^{s_i} \right].$$
5. *Q is the four-coordinate specialization row for the five partition types 4, $3+1$, $2+2$, $2+1+1$, and $1+1+1+1$.*
6. *I is the incidence-total row that reads the quotient-level and codon-level totals from the displayed specialization.*
7. *T records the downstream defect-flow handoff to the partition skeleton and incidence certificate.*
8. *H records componentwise* hsame *transport for block sizes, legal decorations, defect exponents, product rows, specialization rows, incidence rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay through $M, S, D, E, F, Q, I, T, H$.*
10. *P is the* Pkg *provenance over* PartitionBasinPolynomial$^{\uparrow}$, Partition$^{\uparrow}$, DefectFlowPartitionSkeleton$^{\uparrow}$, PartitionCoreIncidenceCertificate$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a stochastic biology model, a host polynomial oracle, a quotient-semantic equality, an unbounded partition enumeration, or a hidden recurrence outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.PartitionBasinPolynomialUp.PartitionBasinPolynomialTasteGate single carrier alignment`

Theorem 4310.2 (PartitionBasinPolynomial NameCert obligations)**.** *For every accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *surface consists exactly of block-count and block-size admission through M, S, legal-decoration admission through D, defect-exponent admission through E, product-formula exposure through F, four-coordinate specialization through Q, incidence-total readback through I, defect-flow handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4363.1. The block-count and block-size rows supply the finite partition core on which the enumeration is read.

The decoration and defect rows are then exposed coordinatewise. For block i, the all-decoration contribution is $(1+q)^{(m-1)s_i}$, and the excluded no-private-coordinate contribution is $\left((1+q)^{m-1}-1\right)^{s_i}$.

Multiplying the legal block rows gives the product row F. The specialization row Q evaluates the same finite formula on the five four-coordinate partition types, and the incidence row I sums the displayed rows with their finite multiplicities.

Finally T, H, C, P, N route the same polynomial surface to the defect-flow and incidence packets. Because no excluded stochastic, host-oracle, quotient-semantic, unbounded-enumeration, or hidden-recurrence coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4310.3 (PartitionBasinPolynomial product specialization)**.** *Every defect-flow consumer of an accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier receives the product formula row for $\mathcal{B}_P(q)$ and the four-coordinate specialization row before reading quotient-level or codon-level incidence totals. The consumer sees only finite block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and local NameCert rows.*

Proof. Start from the obligations of Definition 4363.2. They force the block-size ledger and legal-decoration rows to be displayed before the product formula can be consumed.

The four-coordinate row is a finite specialization of the same product row. Its incidence totals are read only after the partition types and finite multiplicities have been displayed.

The handoff through T, H, C, P, N preserves the same finite surface for the defect-flow skeleton and partition-core incidence certificate. Thus no stochastic model, host generating-function oracle, quotient-semantic equality, or hidden enumeration can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$.

Constructive story: A PartitionBasinPolynomial$^{\uparrow}$ packet is generated from block-count rows, block-size ledgers, legal-decoration rows, defect-exponent ledgers, a product-formula row, a four-coordinate specialization row, incidence-total rows, a defect-flow handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 4363.1, Definition 4363.2, and Definition 4363.3 seed the finite partition-basin polynomial packet over displayed block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a stochastic biology model, host generating-function oracle, quotient-level semantic equality, unbounded partition enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-formula exposure, four-coordinate specialization, incidence readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionBasinPolynomialUp.

Chapter 4311

A Concrete Naming Certificate for $\mathsf{DistanceToSet}^{\uparrow}$

Following Bishop and Bridges on located subsets in metric spaces, the $\mathsf{DistanceToSet}^{\uparrow}$ packet records the finite route by which a point-to-set distance request is read through displayed located-radius rows and closed-ball witnesses. It sits after the metric source of chapter 57, the located-set surface of chapter 1541, the metric closed-ball boundary of chapter 1451, the located closed-ball handoff of chapter 4063, and the terminal real-name seal of chapter 38. The packet names distance-to-set as finite BEDC data; it is not a host infimum, not a selected nearest point, not quotient real equality, and not a countable-choice construction.

Definition 4311.1 (Distance-to-set carrier)**.** *A* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, x, r, B, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the subset by finite located-radius observations.*
3. *x is the displayed point row in the metric source.*
4. *r is the dyadic radius ledger recording the inspected tolerance windows.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness row attached to the point and radius ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance name, read only after the located-set and closed-ball rows are displayed.*
7. *H records componentwise* hsame *transport for the metric source, located set, point row, dyadic ledger, closed-ball witness, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the metric source through located-set exposure, point inspection, dyadic-radius accounting, closed-ball handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, x, r, B, R, H, C, P, N$. It has no coordinate for a host infimum over all points of A, a selected nearest point, quotient real equality, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DistanceToSetUp.DistanceToSetTasteGate single carrier alignment`

Theorem 4311.2 (Distance-to-set NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier*

$$D = (X, A, x, r, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located set, point row, dyadic radius ledger, closed-ball witness, Real seal, transport, replay, provenance, and local naming;*
2. *located-set exposure, requiring every subset read to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *row A before a distance row is exported;*
3. *dyadic-radius monotonicity, requiring the tolerance ledger r to preserve the displayed radius order used by the located-set and closed-ball rows;*
4. *closed-ball handoff, requiring each ball membership or exclusion read to pass through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness B and the located closed-ball sibling surface before the Real seal is read;*
5. *Real seal non-escape, requiring R to seal only the displayed distance name and not a host infimum or quotient real;*
6. *componentwise transport, displayed replay, package provenance, and local* NameCert *naming for exactly the displayed packet.*

No DistanceToSet obligation is stored outside these rows.

Proof. Project the carrier of Definition 4364.1. The metric source is X, the located subset is A, the inspected point is x, and the finite tolerance information is held by r.

The closed-ball witness B is read only after those rows are displayed. Its radius requests are therefore attached to the same metric source, located-set row, point row, and dyadic ledger.

The Real seal R comes after the located-radius and closed-ball rows. Transport H and replay C preserve the same order, while P and N record provenance and local naming.

Since no coordinate stores a host infimum, a nearest-point selector, quotient real equality, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 4311.3 (Distance-to-set located-radius handoff)**.** *Every consumer of an accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *packet reads distance-to-set data only through the displayed located-radius row A, the dyadic radius ledger r, the closed-ball witness B, and the Real seal R, transported by H and replayed by C. The consumer cannot bypass these rows by reading a host infimum, a selected nearest point, quotient real equality, countable choice, or an ambient completeness principle.*

Proof. Read the carrier of Definition 4364.1. The only subset coordinate is the located-set row A, and the only radius coordinate is the dyadic ledger r.

The closed-ball witness B supplies the displayed ball boundary before the Real seal R is exposed. Thus a distance consumer reaches the exported real name only through the located-radius and closed-ball rows.

Apply Definition 4364.2. The transport, replay, provenance, and local naming rows preserve the same finite packet and add no host infimum, nearest-point selector, quotient real equality, countable choice, or ambient completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}$.

Constructive story: A $\mathsf{DistanceToSet}^{\uparrow}$ packet is generated from a metric source, located set row, point row, dyadic radius ledger, closed-ball witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4364.1, Definition 4364.2, and Definition 4364.3 seed the finite located-radius and closed-ball interface for distance-to-set consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host infimum construction, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-set exposure, dyadic-radius monotonicity, closed-ball handoff, Real seal non-escape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DistanceToSetUp.

Chapter 4312

A Concrete Naming Certificate for $\mathsf{JordanCurve}^{\uparrow}$

Following Jordan and constructive topology, the $\mathsf{JordanCurve}^{\uparrow}$ packet records the finite polygonal and winding-ledger route by which a displayed simple closed curve in the real plane may be consumed as a separation boundary. It sits after the terminal real-name seal of chapter 38, the circle carrier of chapter 58, the graph and modulus rows of chapter 130, the finite compact-metric source of chapter 129, and the simplex inventory of chapter 238. The packet names a constructive finite obligation surface; it does not import classical open-cover compactness, selected complementary components, quotient path equality, or a choice-dependent separation theorem.

Definition 4312.1 (Jordan curve carrier)**.** *A* $\mathsf{JordanCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, F, P, W, O, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, recording the circle parameter ledger and its local classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row into the displayed* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ *plane, with graph and modulus observations visible.*
3. *P is the finite polygonal approximation row, read through a* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision and its vertex-image ledger.*
4. *W is the winding ledger, recording the finite edge-crossing and sign-change data used to compare test points with the polygonal boundary.*
5. *O is the orientation row, recording the compatible cyclic order inherited from the* $\mathsf{S}^{1\uparrow}$ *row and the polygonal subdivision.*
6. *I is the injectivity-apartness row, stating that distinct inspected arcs carry displayed positive separation windows before they are used by the boundary read.*
7. *E is the exterior-interior ledger, recording only the finite side classification certified by the current polygonal and winding rows.*
8. *H records componentwise* hsame *transport for $S, F, P, W, O, I, E, C, Q, N$.*
9. *C records displayed* Cont *replay from circle, map, polygonal, winding, orientation, and apartness rows to the side-classification row.*

10. *Q is the* Pkg *provenance over* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, P, W, O, I, E, H, C, Q, N$. It has no coordinate for a host Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or countable-choice separation.

Lean4 ✓ `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate single carrier alignment`

Theorem 4312.2 (Jordan curve orientation ledger)**.** *For every accepted* $\mathsf{JordanCurve}^{\uparrow}$ *carrier*

$$J = (S, F, P, W, O, I, E, H, C, Q, N),$$

the orientation row O is determined by the finite $\mathsf{S}^{1\uparrow}$ *cyclic order, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision in P, and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *edge-image ledger in F. The winding ledger W can read orientation only through those rows, transported by H and replayed by C.*

Proof. Project the accepted carrier. The only circle-facing coordinate is S, and the only finite boundary skeleton is the polygonal row P together with the displayed edge images in F.

The orientation row O is therefore read by following the cyclic order of S across the subdivision cells of P. Each edge-image comparison is a finite real-plane observation supplied by F, not by an ambient topological orientation principle.

The winding ledger W consumes that ordered edge list. Componentwise transport H and replay C preserve the same ordered rows, while Q and N record provenance and local naming. Since the carrier contains no host orientation object or quotient path coordinate, the finite ledger is the entire orientation source. □

Theorem 4312.3 (Jordan curve winding separation boundary)**.** *Every side-classification read from an accepted* $\mathsf{JordanCurve}^{\uparrow}$ *packet factors through the finite route*

$$S, F, P, O, I, W \longmapsto E \longmapsto H, C, Q, N.$$

Thus the packet exposes polygonal approximation, injectivity-apartness, orientation, and winding ledgers before it names an interior or exterior row. It does not assert a blanket classical Jordan separation theorem, selected complementary components, open-cover compactness, quotient path equality, or choice-dependent planar separation.

Proof. Read the carrier of Definition 4365.1. The curve source is S, the displayed map is F, and the finite polygonal boundary is P.

Before a side row is exported, the packet requires the orientation ledger O, injectivity-apartness row I, and winding ledger W. These rows are finite BHist data: the winding read uses edge crossings and signs already attached to the polygonal boundary.

The exterior-interior row E is downstream of that route. Transport, replay, provenance, and naming through H, C, Q, N preserve the same finite packet. The excluded host theorem, selected components, open covers, quotient paths, and choice rows are not coordinates of the carrier, so they cannot be read from the local NameCert surface. □

Falsifiable prediction: A $\mathsf{JordanCurve}^{\uparrow}$ consumer that exports side separation without reading P, W, O, I refutes the carrier surface.

Independence witness: real, sone, continuousmap, compactmetric, simplicialcomplex: the sibling chapters supply real-plane coordinates, circle source data, continuous graph and modulus rows, compact-metric finite inspection, and subdivision cells separately, while $\mathsf{JordanCurve}^{\uparrow}$ binds them into the finite packet $J = (S, F, P, W, O, I, E, H, C, Q, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$.

Constructive story: A $\mathsf{JordanCurve}^{\uparrow}$ packet is generated from a circle source row, continuous real-plane map row, finite polygonal subdivision, winding ledger, orientation row, injectivity-apartness row, side-classification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4365.1, Definition 4365.2, and Definition 4365.3 seed the finite polygonal, winding, orientation, apartness, side-classification, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, the classical Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or choice-dependent separation.

Upgrade path: Obligation closure requires checked carrier admission, polygonal approximation exactness, winding-ledger stability, orientation transport, injectivity-apartness handoff, side-classification non-escape, replay, provenance, and local naming for BEDC.Derived.JordanCurveUp.

Chapter 4313

A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$

The MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite NameCert data. It sits beside the candidate-set packet of chapter 2320 and the normalization router of chapter 1108: those siblings expose candidate evidence and route selection, while this packet records the boundary where empty-context closed typed terms may or may not enter the candidate interface. The packet names a closed-projection frontier only; it does not turn candidate-mediated normalization into an unconditional strong-normalization theorem for every closed MetaCIC term.

Definition 4313.1 (MetaCIC candidate closed-projection boundary carrier)**.** *A* MetaCICCandidateClosedProjection *carrier is a finite* BHist *packet*

$$B = (T, K, R, E, NF, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row whose projection into the candidate interface is being tested.*
2. *K is the candidate-set evidence row, read through* CandidateSetNormalization$^{\uparrow}$ *when the projection is admitted.*
3. *R is the router row, read through* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *and recording which candidate, closed-candidate, or candidate-set route is exposed.*
4. *E is the bounded-normal endpoint seal row, recording the finite normal-output evidence available after candidate admission.*
5. *NF is the no-infinite-reduction row exported only downstream of K, R, E.*
6. *O is the obstruction row recording refusal when the closed term cannot be placed into the displayed candidate interface.*
7. *H records componentwise* hsame *transport for $T, K, R, E, NF, O, C, P, L$.*
8. *C records displayed* Cont *replay from the closed-term row through candidate admission, router selection, endpoint seal, no-infinite-reduction read, and obstruction retention.*

9. *P is the* Pkg *provenance over* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$, CandidateSetNormalization$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, MetaCICOpenProblemLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, R, E, NF, O, H, C, P, L$. It has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Lean4 ✓ `BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.MetaCICCandidateClosedProjectionBoundar`
`single carrier alignment`

Theorem 4313.2 (MetaCIC candidate-set closed projection frontier)**.** *For every accepted* MetaCICCandidateClosed *carrier*

$$B = (T, K, R, E, NF, O, H, C, P, L),$$

the closed-term row T reaches a strong-normalization-facing read only through candidate evidence K, router selection R, and bounded-normal endpoint seal E. If those rows do not place T inside the candidate interface, the obstruction row O is the only exported boundary datum.

Proof. Project the carrier. The closed-term row T is present as a finite syntactic and typing packet, but it is not itself a normalization theorem.

The positive route first reads K through chapter 2320. The router row R then records the candidate, closed-candidate, or candidate-set branch through chapter 1108. Only after those rows are displayed can the endpoint seal E and no-infinite-reduction row NF be consumed.

If candidate admission is absent, O records the refusal. Transport, replay, provenance, and local naming through H, C, P, L preserve that same finite boundary, so the packet exports no evidence-free closed-term strong-normalization claim. □

Theorem 4313.3 (MetaCIC candidate-set SN handoff boundary)**.** *Every strong-normalization handoff exported by an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet is a handoff for rows already inside the displayed candidate interface. It cannot be read as a blanket theorem that all empty-context typed closed MetaCIC terms are strongly normalizing.*

Proof. Start with the frontier theorem. It places the normalization-facing route behind K, R, E and keeps the obstruction row O visible when the candidate interface is not supplied.

For an admitted row, the handoff consumes candidate evidence from CandidateSetNormalization$^{\uparrow}$ and route evidence from MetaCICNormalizationCertificateRouter$^{\uparrow}$. The no-infinite-reduction row NF is therefore downstream of the admitted candidate branch rather than a second source of normalization evidence.

For a closed term outside that interface, the carrier has only T and the obstruction row O on the boundary. The structural rows H, C, P, L transport, replay, source, and name the boundary, but they do not add subject reduction, host conversion equality, quotient reduction equality, or a global normalizer. Hence the handoff is candidate-scoped. □

Theorem 4313.4 (MetaCIC candidate-set public boundary)**.** *Any public no-infinite-reduction or strong-normalization read from an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet must cite candidate-set membership or soundness, router selection, and bounded-normal endpoint rows. A bare empty-context typed closed-term row T is insufficient: the public boundary may expose NF only downstream of the candidate-set facts named by* closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, *and* canonicalCandidateSet_sound *together with the router row R and endpoint seal E.*

Proof. Start with Definition 4366.2. It places every normalization-facing read behind the candidate evidence row K, router row R, and endpoint seal E, and leaves the obstruction row O as the only datum when candidate admission is absent.

Next apply Definition 4366.3. That theorem identifies the handoff as a candidate-interface handoff rather than a theorem about all empty-context closed typed rows. Thus the public NF row cannot be projected from T alone.

Finally read the MetaCIC normalization audit map through the existing candidate-set facts closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, and canonicalCandidateSet_sound. Those facts supply the candidate membership and soundness side of the route, while R and E supply the public router and endpoint rows. The resulting boundary is therefore candidate-mediated and evidence-bearing, not an unconditional closed-term normalizer. □

Falsifiable prediction: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ consumer that exports strong normalization from T without reading K, R, E refutes the carrier surface.

Independence witness: candidate-set-normalization, metacicnormalizationcertificaterouter, metacicopenproblemledger: the candidate-set packet carries candidate evidence, the router selects the admissible normalization route, and the open-problem ledger records the frontier question; this packet binds those rows into a closed-projection boundary without adding an evidence-free normalization theorem.

Closure status for NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$.

Constructive story: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate-set evidence, normalization-router selection, bounded-normal endpoint seal, no-infinite-reduction row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4366.1, Definition 4366.2, Definition 4366.3, and Definition 4366.4 seed the finite closed-projection, candidate-evidence, router-selection, endpoint-seal, obstruction, public-boundary, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional strong normalization for all closed typed MetaCIC terms, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Upgrade path: Obligation closure requires checked closed-term projection, candidate-evidence admission, router selection, bounded-normal endpoint sealing, no-infinite-reduction handoff, obstruction exactness, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.

Chapter 4314

A Concrete Naming Certificate for $\mathsf{PointedMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based metric spaces, the $\mathsf{PointedMetricSpace}^{\uparrow}$ packet records a metric carrier together with a named basepoint route before completion, loop-space, or embedding consumers are allowed to inspect point-level data. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, reads names through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, passes regular rational distance observations through chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and offers a based source for the distance-preserving consumer route of chapter 439. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected representatives, or ambient completeness.

Definition 4314.1 (Pointed metric space carrier)**.** *A* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, b, D, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *b is the basepoint name row, stored as finite BEDC data attached to M rather than as a host-level point.*
3. *D is the basepoint distance ledger, recording the finite requests that compare b with inspected metric names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which basepoint and metric names are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the basepoint-distance observations.*
7. *H records componentwise* hsame *transport for M, b, D, S, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a based metric request through the stream windows, regular rational readback, and real-distance seal.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{PointedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint name, distance ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate single carrier alignment`

Theorem 4314.2 (Pointed metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier* $P = (M, b, D, S, R, E, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint name, basepoint distance ledger, stream windows, regular rational readback, real-distance seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring every comparison of basepoint data to be read over the transported metric row* M*;*
3. *basepoint-distance exposure, requiring every basepoint comparison to pass through* D, S, R, E *before any completion-facing or embedding-facing consumer reads it;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 4367.1. The projection exposes the metric row, basepoint name, distance ledger, stream windows, regular readback, real seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, b, D, S, R, E together, so a consumer cannot compare a basepoint name over one metric carrier with distance observations over another.

The distance ledger fixes the read order. A basepoint comparison is first a finite request in D, then a scheduled stream observation in S, then a regular rational readback in R, and only then a $\mathsf{Real}^{\uparrow}$ distance seal in E.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected representatives, or ambient completeness, the listed rows exhaust the local naming surface. □

Theorem 4314.3 (Pointed metric space basepoint nonescape)**.** *Let* $P = (M, b, D, S, R, E, H, C, P_0, N)$ *be an accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier. Every completion-facing, loop-space-facing, or isometric-embedding-facing consumer of the basepoint row* b *must read it through the displayed route*

$$M, b, D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P_0, N.$$

Thus the basepoint is exported only as finite named metric data over M*, not as a host point, quotient representative, selected completion point, or ambient metric-equality witness.*

Proof. Begin with carrier admission. The only row naming the basepoint is b, and it is stored beside the metric carrier M and distance ledger D.

Read the inspection route next. The stream windows S and regular rational readback R display every observed distance from the basepoint before the $\mathsf{Real}^{\uparrow}$ seal E is available to downstream consumers.

Transport and replay preserve that same route. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add host points, quotient representatives, selected completion points, or ambient metric equality.

Therefore every based consumer receives exactly the finite route displayed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedMetricSpace}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a named basepoint row, a basepoint distance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4367.1, Definition 4367.2, and Definition 4367.3 seed the based metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host point equality, quotient metric equality, selected completion points, loop-space closure, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, basepoint-distance exposure, classifier transport, basepoint nonescape, and downstream completion and isometric-embedding consumer alignment for BEDC.Derived.PointedMetricSpaceUp.

Chapter 4315

A Concrete Naming Certificate for $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based complete metric spaces, the $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet records a based metric carrier together with a displayed completion service before completion, loop-space, or embedding consumers are allowed to read a point as complete. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service boundary of chapter 128, the $\mathsf{PointedMetricSpace}^{\uparrow}$ basepoint route of chapter 4367, the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected completion representatives, countable choice, or ambient order completeness.

Definition 4315.1 (Pointed complete metric space carrier)**.** *A* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, B, Q, S, R, E, L, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *B is the basepoint row, stored as finite named metric data over M rather than as a host-level point.*
3. *Q is the Cauchy-window route, listing the finite tail requests through which regular Cauchy names based at B are inspected.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the basepoint and Cauchy windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance and modulus observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed metric-distance and limit-distance observations.*
7. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion seal, exposing only the displayed limit-service handoff for the Cauchy-window route Q.*
8. *H records componentwise* hsame *transport for M, B, Q, S, R, E, L and the local certificate rows.*
9. *C records displayed* Cont *replay from a based complete-metric request through the Cauchy windows, stream observations, regular rational readback, real seal, and completion seal.*

10. *P_0 is the* Pkg *provenance over* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.TasteGate.PointedCompleteMetricSpaceTasteGate single carrier alignment`

Theorem 4315.2 (Pointed complete metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier* $P = (M, B, Q, S, R, E, L, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSp}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint row, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring the basepoint row, Cauchy windows, and completion seal to be read over the transported metric row M;*
3. *Cauchy-window exposure, requiring every completion-facing read to pass through Q, S, R, E, L before any downstream consumer may inspect the limit handoff;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, selected completion representative, countable-choice selector, or ambient order-completeness principle.

Proof. Project the carrier of Definition 4368.1. The projection exposes the metric row, basepoint, Cauchy-window route, stream windows, regular readback, real seal, completion seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, B, Q, S, R, E, L together, so a consumer cannot compare a based Cauchy route over one metric carrier with a completion seal over another.

The completion-facing read order is fixed by the route rows. A completion request begins as a finite Cauchy-window request in Q, becomes a scheduled stream observation in S, is read back as regular rational data in R, is sealed as real-distance data in E, and only then reaches the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service row L.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected completion representatives, countable-choice selectors, or ambient order completeness, the listed rows exhaust the local naming surface. □

Theorem 4315.3 (Pointed complete metric space Cauchy basepoint route)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ be an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier. Every based regular Cauchy request whose initial window is anchored at B is read through the displayed route*

$$M, B, Q \longmapsto S \longmapsto R \longmapsto E \longmapsto L \longmapsto H, C, P_0, N.$$

Thus the basepoint participates in completion only as named metric and Cauchy-window data over M, not as a selected host point or completed-limit representative.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.PointedCompleteMetricSpaceCauchyBasepointRoute`

Proof. Begin with carrier admission. The only row naming the basepoint is B, and it is stored beside the metric row M and the Cauchy-window route Q.

Read the window route next. The stream row S schedules the inspected windows, the regular rational row R reads the tail and distance ledgers, and the $\mathsf{Real}^{\uparrow}$ row E seals the observed distances before the complete-metric row L can expose any limit-service handoff.

Transport and replay preserve that same path. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add a host basepoint, a selected limit representative, quotient equality, or a separate completion object.

Therefore every based Cauchy consumer receives exactly the finite route displayed in the statement. □

Theorem 4315.4 (Pointed complete metric basepoint completion handoff)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ and $P' = (M', B', Q', S', R', E', L', H', C', P'_0, N')$ be accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carriers. If the two packets transport M, B, Q, S, R, E through the same componentwise H row, then every* $\mathsf{CompleteMetric}^{\uparrow}$ *service read of L is taken over that same named basepoint route before any loop-space or embedding consumer may inspect a completed point.*

Equivalently, a consumer may inspect

$$L \quad \text{or} \quad L'$$

only after the route

$$M, B, Q, S, R, E \longmapsto H, C, P_0, N$$

has been displayed; it cannot detach completion service from the transported basepoint row.

Proof. Project both carriers. In each packet the completion-service coordinate is L, and the only rows upstream of it are the metric row M, the basepoint row B, the Cauchy-window row Q, the stream row S, the regular rational row R, and the real-distance seal E.

Apply Definition 4368.3 to the first packet. The displayed route forces every based Cauchy request to pass through M, B, Q, S, R, E before the service row L is visible.

Use the assumed componentwise transport row H to move the same route to the second packet. The replay row C then reproduces the transported route for the completion-facing consumer, and the provenance and local name rows P_0, N record no independent completed point.

Thus a loop-space or embedding consumer can inspect a completed point only after the transported basepoint route has been displayed. There is no carrier coordinate from which the consumer could read L apart from B. □

Theorem 4315.5 (Pointed complete metric space completion nonescape)**.** *Every completion-facing, loop-space-facing, or embedding-facing consumer of an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *packet factors its completed-point read through*

$$M, B, Q, S, R, E, L, H, C, P_0, N.$$

Consequently the packet exports a based complete metric certificate only as finite BEDC NameCert data; it exports neither host points nor quotient completion equality.

Proof. Inspect the carrier coordinates. The only completion coordinate is L, and the definition makes L a completion seal for the already displayed Cauchy-window route Q over the metric row M and basepoint row B.

The readback rows precede the completion seal. Any consumer must first pass through S, R, E to display stream windows, regular rational ledgers, and real distance seals, so no completed point can be read before the finite analytic route is visible.

The remaining rows are administrative in the BEDC sense. Transport H, replay C, provenance P_0, and local name N preserve and source the packet but do not create host points, quotient completion equality, selected representatives, or ambient completeness.

Thus every downstream completed-point read factors through the displayed finite packet, which proves completion nonescape. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet is generated from a MetricUp carrier, a named basepoint row, a Cauchy-window route, StreamName windows, RegSeqRat readback, a RealUp seal, a CompleteMetricUp completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4368.1, Definition 4368.2, Definition 4368.3, and Definition 4368.5 seed the based complete-metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient completion equality, selected host endpoints, loop-space closure, embedding closure, or ambient order completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, classifier transport, basepoint completion routing, completion nonescape, and downstream completion, loop-space, and embedding consumer alignment for BEDC.Derived.PointedCompleteMetricSpaceUp.

Chapter 4316

A Concrete Naming Certificate for $\mathsf{IntervalConnectedness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{IntervalConnectedness}^{\uparrow}$ packet records the finite constructive route by which a located real interval is inspected as one connected interval surface before intermediate-value, zero-envelope, or completion-facing consumers use it. It sits after the endpoint seal of chapter 38, the located interval carrier of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sandwich consumer surface of chapter 2163. The packet names interval connectedness as displayed finite BEDC data; it does not import host topology, open-cover compactness, trichotomy, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4316.1 (Interval connectedness carrier)**.** *An* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint-pair row, storing the lower and upper endpoint names through their displayed real seals.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *witness row, carrying the located interval budget, locator reads, and endpoint-radius checks attached to E.*
3. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision-chain ledger, recording the dyadic cells, adjacent-cell overlaps, and tolerance requests inspected inside L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, cell, and witness observations are scheduled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled endpoint and subdivision observations.*
6. *T records componentwise* hsame *transport for endpoint pairs, located interval witnesses, dyadic subdivision chains, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint pairs through located interval exposure, dyadic subdivision, stream-window inspection, regular readback, and the interval-connectedness handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalConnectedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SqueezeTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, D, S, R, T, C, P, N$. It has no coordinate for host topological connectedness, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalConnectednessUp.IntervalConnectednessTasteGate single carrier alignment`

Theorem 4316.2 (Interval connectedness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier*

$$I = (E, L, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint pair, located interval witness, dyadic subdivision chain, stream windows, regular readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring endpoint, locatedness, subdivision, window, readback, replay, provenance, and naming data to move componentwise through T;*
3. *dyadic chain exactness, requiring every inspected subinterval and adjacent-cell overlap to be recorded in D before a connectedness consumer reads the packet;*
4. *endpoint handoff nonescape, requiring every consumer of interval endpoints to pass through E, L, D, S, R before using any* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger exhaustion, requiring the packet to expose no host topological connectedness theorem, open-cover compactness, trichotomy, quotient real equality, selected intermediate point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The sibling routes through $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{SqueezeTheorem}^{\uparrow}$ *are the only named consumers of this local certificate surface.*

Proof. Project the carrier of Definition 4369.1. The endpoint pair is visible only in E, and the located interval evidence is visible only in L.

The subdivision data are then forced through D. Each finite cell, overlap, and tolerance request is read as dyadic data before any stream-window or regular-readback consumer can inspect it.

Transport and replay preserve the same route. The rows T and C move endpoint, locatedness, subdivision, window, and readback data componentwise, while P and N source and name exactly that displayed packet.

Since the carrier has no coordinate for the excluded host principles, the local certificate cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{IntervalConnectedness}^{\uparrow}$ consumer reads endpoint pairs, located interval witnesses, dyadic subdivision chains, StreamName windows, RegSeqRat readbacks, hsame transport, Cont replay, Pkg provenance, and local naming before any interval-connectedness handoff is visible.

Independence witness: real, locatedrealinterval, dyadicratcore, streamname, regseqrat, squeezetheorem: $\mathsf{IntervalConnectedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, located interval witnesses, finite dyadic subdivision chains, scheduled stream windows, regular rational readbacks, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$.

Constructive story: An IntervalConnectedness$^{\uparrow}$ packet is generated from Real endpoint pairs, a LocatedRealInterval witness, a finite DyadicRatCore subdivision chain, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, seed)

Scope closed: Definition 4369.1 and Definition 4369.2 seed the finite located-interval connectedness route over Real$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and SqueezeTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topology, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic chain exactness, endpoint handoff nonescape, ledger exhaustion, and downstream intermediate-value consumer alignment for BEDC.Derived.IntervalConnectedness

Chapter 4317

A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$

Following Tait–Martin-Lof and Barendregt on residuals, parallel reduction, and confluence discipline, the MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ packet records the finite boundary where a beta-parallel branch and a substitution replay must meet before the MetaCIC confluence route may use the conditional diamond bridge. It sits over the audit map of chapter 9378 and names the residual-substitution input as packet data rather than as a global Church–Rosser theorem. The packet does not claim full closed Church–Rosser, arbitrary closed-source confluence, or an unrestricted substitution theorem.

The residual-substitution packet is consumed by chapter 765 as a guarded input to candidate-mediated SN/confluence scheduling. The dependency is the residual join row together with the closedness guard data: it tells the scheduler which finite substitution replay may enter a candidate branch and which source windows are closed enough for that handoff, not that the full Church–Rosser theorem has already been completed.

Definition 4317.1 (MetaCIC residual-substitution compatibility carrier)**.** *A* MetaCICResidualSubstitutionCompatil *carrier is a finite* BHist *packet*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

with the following displayed rows:

1. *s is the source term row whose beta-parallel residual branch is inspected.*
2. *t is the substituted term row, storing the finite substitution request and the target occurrence window.*
3. *b is the parallel branch row, displaying the left and right beta-parallel steps before any join is named.*
4. *j is the residual join row, recording the candidate common reduct obtained after substitution replay and residual transport.*
5. *g is the closedness guard row, saying exactly which source and substitution windows are closed enough for the guarded handoff.*
6. *a is the atom-shape separation row, keeping atom, application, lambda, Pi, and beta-redex cases visibly disjoint.*
7. *h records componentwise* hsame *transport for source, substitution, branch, residual join, closedness guard, atom-shape separation, replay, provenance, and local naming rows.*

8. *c records displayed* Cont *replay from the source term through substitution, parallel branching, residual join construction, closedness guarding, and atom-shape separation.*
9. *p is the* Pkg *provenance over* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$*, MetaCIC beta-parallel reduction, residual joins, closedness guards, atom-shape separation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *n is the local* NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, t, b, j, g, a, h, c, p, n$. It has no coordinate for a completed parallel diamond, full closed Church–Rosser, arbitrary closed-source confluence, or an unguarded substitution principle.

Lean4 ✓ BEDC.Derived.MetaCICResidualSubstitutionCompatibilityUp.MetaCICResidualSubstitutionCompatibilit
single carrier alignment

Theorem 4317.2 (MetaCIC residual-substitution compatibility NameCert obligations)**.** *For every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the local NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and local naming rows;*
2. *branch-shape disjointness, requiring atom, application, lambda, Pi, and beta-redex branches to be separated by a before a residual join row is accepted;*
3. *substitution residual replay, requiring c to display how the substituted term row and beta-parallel branch row reach the same candidate join j;*
4. *closed-source guard, requiring g to expose the exact closedness assumptions consumed by the guarded compatibility packet;*
5. *non-claim of full closed Church–Rosser, requiring the packet to remain an input boundary for the conditional confluence bridge rather than a proof of arbitrary closed-source confluence.*

No obligation is stored outside $s, t, b, j, g, a, h, c, p, n$.

Proof. Project the carrier of Definition 4370.1. The source term, substituted term, branch, residual join, closedness guard, and atom-shape separation rows are the only computational coordinates.

The branch-shape row a fixes the case split before the join row is read. Thus a beta-redex residual branch cannot be silently identified with an atom, application, lambda, or Pi shape.

Replay then supplies the compatibility boundary. The row c records the finite route from substitution and beta-parallel branching to the candidate join j, while g states the closedness guard consumed by that route.

Finally h, p, n transport, source, and name exactly the displayed packet. Since full closed Church–Rosser and arbitrary closed-source confluence are not carrier coordinates, they are not exported by this NameCert surface. □

Theorem 4317.3 (MetaCIC residual-substitution candidate projection route)**.** *Every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

exports a candidate-closed projection route to chapter 4366 only after the source term row s, substituted term row t, beta-parallel branch row b, residual join row j, closedness guard row g, and atom-shape separation row a have all been displayed. The downstream candidate boundary receives a guarded compatibility packet, not full Church–Rosser and not arbitrary closed-source confluence.

Proof. Project the carrier of Definition 4370.1. The rows s, t, b, j, g, a are the only computational rows available before the structural transport, replay, provenance, and local naming rows are read.

Use Definition 4370.2. Branch-shape disjointness exposes a, substitution residual replay exposes the route from t and b to j, and the closed-source guard exposes exactly the assumptions in g.

The candidate boundary of chapter 4366 is therefore reached through a guarded compatibility packet. The route supplies source, substitution, beta-parallel, residual-join, closedness, and atom-shape data, while h, c, p, n only transport, replay, source, and name those rows.

Since completed Church–Rosser and arbitrary closed-source confluence are absent from the carrier, the candidate-closed projection route cannot export them. It hands off only the displayed residual-substitution compatibility boundary. □

Theorem 4317.4 (MetaCIC residual-substitution obligation closure surface)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the obligation surface is exactly the displayed source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and naming rows. The candidate-closed projection boundary receives only this guarded compatibility packet.

Proof. Apply Definition 4370.2. It identifies carrier admission, branch-shape disjointness, substitution residual replay, closed-source guarding, transport, replay, provenance, and naming as rows of $s, t, b, j, g, a, h, c, p, n$.

Apply Definition 4370.3. The candidate boundary receives the source, substitution, beta-parallel branch, residual-join, closedness, and atom-shape data only through that guarded projection route.

Therefore the obligation surface packages the candidate handoff without exporting full closed Church–Rosser, arbitrary closed-source confluence, or unguarded substitution compatibility. Those claims are not coordinates of the carrier and are not supplied by the candidate projection. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ read displays source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, hsame transport, Cont replay, Pkg provenance, and local naming rows before the conditional MetaCIC confluence bridge can consume it.

Independence witness: metacic-confluence-audit: $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ is not a completed confluence theorem because it carries only the finite residual-substitution compatibility boundary consumed by the conditional parallel-diamond route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ packet is generated from source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4370.1, Definition 4370.2, and Definition 4370.3 seed the finite residual-substitution compatibility boundary consumed by the conditional MetaCIC confluence bridge and the candidate-closed projection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, arbitrary closed-source confluence, unguarded substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-shape disjointness, substitution residual replay, closed-source guard, and conditional confluence handoff alignment for BEDC.MetaCIC.BetaParallelSubstitutionCompatibility.

Chapter 4318

A Concrete Naming Certificate for $\mathsf{ComputableCompactMetric}^{\uparrow}$

Following Weihrauch-style computable analysis and the Bishop–Bridges compact metric arguments, the $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet records the effective compact-metric surface as finite BEDC NameCert data. It sits after the compact metric carrier of chapter 129, the effective Polish source of chapter 1706, the total-bounded finite-net route of chapter 127, the complete metric source of chapter 128, and the L10 readback route through chapter 126, chapter 301, and chapter 38. The packet names the finite-cover realizer surface consumed by effective compact analysis; it does not assert open-cover compactness, selected subcovers, quotient completion equality, or an ambient real-completeness principle.

Definition 4318.1 (Computable compact metric carrier)**.** *A* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact metric packet whose finite-net behavior is being made effective.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *source row exposing the effective dense-name and semidecidable basic-ball interface.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row naming the finite-net source for each displayed dyadic tolerance request.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row recording the convergence service available to Cauchy and compact consumers.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for inspected dense names, finite net centers, tolerance requests, and cover indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of distances and tolerance comparisons.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows and regular rational comparisons have been displayed.*

8. *H records componentwise* hsame *transport for compact metric, effective Polish, total-bounded, complete-metric, stream-window, regular-readback, real-seal, finite-cover realizer, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from an effective compactness request through K, E, T, C, S, R, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ComputableCompactMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, T, C, S, R, L, H, Q, P, N$. It has no coordinate for an open-cover subcover, a choice-selected finite set, quotient completion equality, excluded-middle compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ComputableCompactMetricUp.ComputableCompactMetricTasteGate single carrier alignment`

Theorem 4318.2 (Computable compact metric NameCert obligations)**.** *For every accepted* ComputableCompactMe *carrier*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *surface consists exactly of the displayed rows in Definition 4371.1. Carrier admission must expose the compact metric source, effective Polish source, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through those same rows, and ledger exhaustion says that every finite-cover or compact-analysis consumer can inspect only $K, E, T, C, S, R, L, H, Q, P, N$.*

Proof. Project the carrier of Definition 4371.1. The rows K, E, T, C are the only structural compactness sources, so carrier admission exposes compact metric data, effective Polish naming data, total-bounded finite nets, and complete-metric convergence before any realizer can be read.

The readback rows S, R, L then bind numerical access to finite windows, regular rational comparisons, and the terminal real seal. This is the same L10 readback discipline used by the StreamName, RegSeqRat, and Real siblings named in the chapter introduction.

Transport and replay are visible rows rather than hidden meta-operations. The row H records componentwise hsame transport, Q records displayed Cont replay, and P, N source and name the same finite packet.

For ledger exhaustion, read any compact-analysis consumer from the accepted packet. The carrier exposes only $K, E, T, C, S, R, L, H, Q, P, N$, so no open-cover subcover, selected finite set, quotient completion equality, excluded-middle compactness, or ambient real-completeness principle can be obtained from this NameCert surface. □

Theorem 4318.3 (Computable compact metric finite-cover realizer)**.** *Every effective finite-cover read exported by an accepted* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *packet factors through the route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsa

Thus a consumer receives a compact metric source, effective dense-name interface, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational tolerance comparisons, terminal real seal, componentwise transport, replay, provenance, and local

naming rows before it uses the finite-cover realizer. The packet exports no selected subcover, open-cover compactness theorem, quotient completion representative, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an accepted carrier $K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$. The compact metric row K and effective Polish row E name the metric and effective dense-name surfaces, while T supplies the finite-net source used for the requested tolerance.

The convergence row C keeps the compact route compatible with complete-metric consumers. A finite-cover realizer cannot leave these displayed sources, because the later rows are read only after the compact, effective, total-bounded, and complete rows are present.

The numerical readback is finite. The stream row S exposes inspected names, center windows, cover indices, and tolerance requests; the regular rational row R compares the displayed distances; and the terminal seal L is reached only after those windows and comparisons have been read.

Finally H, Q, P, N preserve the route under transport, replay, provenance, and local naming. Since selected subcovers, open-cover compactness theorems, quotient representatives, choice selectors, and ambient completeness witnesses are not carrier coordinates, the exported realizer is exactly the displayed finite-cover surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet is generated from CompactMetric, EffectivePolishSpace, TotallyBounded, CompleteMetric, StreamName, RegSeqRat, RealUp, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4371.1, Definition 4371.2, and Definition 4371.3 seed the effective finite-cover realizer route for computable compact metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, open-cover compactness, quotient completion equality, countable-choice selectors, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective dense-name exposure, total-bounded finite-net exposure, complete-metric convergence exposure, finite-window readback, regular rational comparison, real sealing, finite-cover realizer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ComputableCompactMetricUp.

Chapter 4319

A Concrete Naming Certificate for $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet records the separated-completion reflection route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the regular rational readback surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness boundary of chapter 3390. The packet lets a Real-facing consumer treat two representative regular Cauchy names as the same separated reflection point only after the displayed finite windows, dyadic budgets, and uniqueness ledger have been exposed.

Definition 4319.1 (Regular-Cauchy Hausdorff-reflection carrier)**.** *A* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row presented to the same separated-reflection request.*
3. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row witnessing that the displayed tail comparisons force separated agreement.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, Y, S, T, D, U have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, stream windows, dyadic ledger, uniqueness row, Real seal, and local certificate rows.*
8. *C records the displayed* Cont *replay from a reflection request to the paired windows, dyadic tolerance ledger, uniqueness row, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier admission surface`

Theorem 4319.2 (Regular-Cauchy Hausdorff-reflection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier* $R = (X, Y, S, T, D, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *separated-agreement row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4372.1. The only displayed coordinates are $X, Y, S, T, D, U, E, H, C, P, N$.

The two regular rational names are compared through the paired stream windows S, T and the dyadic tolerance ledger D. The separated-agreement row U is therefore a displayed uniqueness boundary, not a quotient of streams or a host equality proof.

The terminal Real seal E is downstream of that boundary. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and no excluded quotient, selector, host-equality, choice, trichotomy, or ambient-completeness coordinate can be projected from the carrier. □

Theorem 4319.3 (Regular-Cauchy Hausdorff-reflection representative independence)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet is independent of the chosen representative regular Cauchy name in the following BEDC sense: if the two presented* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are compared through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the terminal seal can distinguish them only through the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row. The packet exports no quotient stream equality, selected representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionRepresentativel`

Proof. Begin with an accepted carrier $R = (X, Y, S, T, D, U, E, H, C, P, N)$. The rows X and Y are not presented as completed real objects; they are presented as regular rational names observed through finite stream windows.

The dyadic ledger D binds the comparison budget for those windows, and the uniqueness row U is the only row that can turn the displayed tail comparison into separated agreement. Thus any Real-facing read of E must pass through the same finite comparison route.

The rows H, C, P, N transport, replay, source, and name this route without adding analytic content. Since the carrier contains no further equality coordinate, changing representatives cannot be detected except through the displayed uniqueness row. □

Theorem 4319.4 (Regular-Cauchy Hausdorff-reflection Real seal non-escape)**.** *No terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the separated reflection supplies representative independence for the Real seal without quotient streams, host real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier real seal nonescape`

Proof. Read the carrier coordinates in order. The terminal row E appears only after the two regular rational names, paired stream windows, dyadic tolerance ledger, and uniqueness row have been displayed.

Apply Definition 4372.2. The local NameCert surface exposes exactly those rows, with H, C, P, N preserving the same route for downstream consumers.

The advertised route is therefore the only public path to the Real seal. Since no excluded equality or choice coordinate occurs in the carrier, the seal cannot escape through quotient streams, host equality, representatives selected outside the packet, countable choice, trichotomy, or ambient completeness. □

Theorem 4319.5 (Regular-Cauchy Hausdorff-reflection finite-window uniqueness route)**.** *Suppose two accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packets display the same paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then every separated-reflection consumer factors through the same finite-window route on the underlying* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization does not use quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier finite window uniqueness route`

Proof. Apply Definition 4372.3. Once the paired StreamName windows, dyadic tolerance ledger, and uniqueness row agree, the terminal Real seal cannot distinguish the two regular Cauchy representatives by any route other than the displayed uniqueness row.

Apply Definition 4372.4. The Real seal is downstream of the finite windows, dyadic ledger, and uniqueness row, so a separated-reflection consumer has no bypass around those displayed rows.

The remaining rows are structural. The hsame rows transport only the displayed components, the Cont rows replay the same finite-window request, and the Pkg and NameCert rows name that route inside BHist. Therefore the consumer factors through the same finite-window route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore tolerance ledger, and a RegularCauchyUniqueness row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyuniqueness: $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational representatives to one separated uniqueness row and one downstream Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyUniqueness separated-agreement row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4372.1, Definition 4372.2, Definition 4372.3, Definition 4372.4, and Definition 4372.5 seed the separated reflection route for regular Cauchy representatives and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, representative-independence handoff, and Real seal non-escape theorem.

Chapter 4320

A Concrete Naming Certificate for $\mathsf{WeakLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on finite averages and probability estimates, the $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet names the displayed finite-average route behind the weak law of large numbers. It sits after the normalized probability surface of chapter 184, the measurable random-variable rows of chapter 185, the finite tail-bound carrier of chapter 6592, and the terminal real-name seal of chapter 38. The packet is a finite certificate for Chebyshev-controlled averages; it is not an almost-sure convergence theorem, a central limit theorem, a countable-choice subsequence selector, a host probability-limit principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4320.1 (Weak law finite-average carrier)**.** *A* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, R, A, V, T, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the total-event normalization and event classifier. The row R is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over that same probability source. The row A records the displayed finite average of the entries in R, while V records the variance or uniform second-moment bound used by the estimate. The row T reads the Chebyshev tail row through* $\mathsf{ChebyshevInequality}^{\uparrow}$ *for the event where the finite average differs from its displayed center by more than the requested tolerance. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the final probability bound. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite averaging route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these ten rows.*

Lean4 ✓ `BEDC.Derived.WeakLawLargeNumbersUp.WeakLawLargeNumbersTasteGate single carrier alignment`

Theorem 4320.2 (Weak law NameCert obligations)**.** *For every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier $(P, R, A, V, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 4373.1. Carrier admission displays the probability source, the random-variable family, the finite-average row, the variance-bound row, the Chebyshev tail row, the epsilon seal, the transport and replay rows, the provenance row, and the local naming row. The packet imports no host law of large numbers, selected subsequence, external probability-limit operation, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The probability source P supplies the measurable-event classifier and total-mass normalization, and the random-variable row R exposes only the finitely displayed variables over that source.

The average row A and variance row V are then internal finite data. They record the algebraic averaging expression and the bound used by the tail estimate before any limiting statement is visible.

Finally read T, E, H, C, Q, N. The Chebyshev tail row and epsilon seal name the probability estimate at the requested tolerance, while transport, replay, provenance, and local naming keep the same finite route under classifier replacement. Since no carrier coordinate stores an infinite limit or subsequence selector, the listed rows exhaust the local certificate surface. □

Theorem 4320.3 (Weak law Chebyshev handoff)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *Chebyshev read factors through the route*

$$P \longmapsto R \longmapsto A \longmapsto V \longmapsto T \longmapsto E \longmapsto H, C, Q, N$$

before any probability-bound consumer receives the finite-average estimate. Thus the Chebyshev step is handed to later probability and $\mathsf{Real}^{\uparrow}$ *formalization as a displayed finite source, finite random-variable row, average row, variance-bound row, tail-event row, epsilon seal, transport, replay, provenance, and local naming data.*

Proof. Start with a probability-bound request. The only probability source exposed by the packet is P, and the only random-variable data exposed over it are the finite rows in R.

The request must next read the average and variance-bound rows. The average row A fixes the finite expression being estimated, and V fixes the bound that the Chebyshev tail row may consume.

Only after these rows are displayed can T and E be read as a bound on the exceptional event. The structural rows H, C, Q, N then preserve that same route for downstream consumers, so the handoff is the finite Chebyshev estimate itself rather than a host-side convergence theorem. □

Theorem 4320.4 (Weak law finite-average nonescape)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *probability-limit-facing read remains inside the finite-average route of Definition 4373.1. In particular, a consumer can read a tail estimate only after displaying the source probability row, finite random-variable rows, average row, variance-bound row, Chebyshev tail row,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by almost-sure convergence, central limit normalization, countable-choice subsequence selection, a host probability-limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a consumer request the finite-average estimate from an accepted carrier. Projection of Definition 4373.1 shows that the request first meets P, R, A, V, T, E and has no independent coordinate for a limit object.

The displayed Chebyshev row T is downstream from the variance-bound row V and upstream from the epsilon seal E. Hence any probability bound read is already tied to the visible finite average and its tolerance.

The remaining rows H, C, Q, N do not add new analytic content. They transport equal displayed histories, replay the finite route, record the package provenance, and attach the local name. Therefore every accepted probability-limit-facing read stays inside the finite-average certificate surface and cannot escape to the excluded convergence or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet is generated from a probability source row, finite random-variable rows, a finite-average row, a variance or second-moment bound row, a Chebyshev tail row, a RealUp epsilon seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4373.1, Definition 4373.2, Definition 4373.3, and Definition 4373.4 seed the finite-average probability route through ProbSpace, RandomVar, RealUp, Chebyshev tail, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the central limit theorem, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-average routing, variance-bound stability, Chebyshev tail handoff, epsilon-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.WeakLawLargeNumbe

Chapter 4321

A Concrete Naming Certificate for $\mathsf{LocatedSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric and real spaces, the $\mathsf{LocatedSubset}^{\uparrow}$ packet records a displayed subset together with finite distance-witness rows. It sits after the real-valued metric route of chapter 1638, the Hausdorff comparison boundary of chapter 2086, the real-separability window surface of chapter 1809, the dyadic tolerance core of chapter 291, and the regular rational readback surface of chapter 301. The packet gives Real and Completion consumers a finite located-subset interface without host set extensionality, selected nearest points, quotient subset equality, or ambient completeness.

The located-subset distance route consumes chapter 1809 as its finite search-window source and consumes chapter 2086 only after the real-distance seal has been displayed.

Definition 4321.1 (LocatedSubset carrier)**.** *A* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, W, D, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *source row, carrying the ambient metric distance through finite real-name observations.*
2. *S is the subset-admission row, naming the displayed members read by the carrier and exporting no host powerset equality.*
3. *W is the* $\mathsf{RealSeparability}^{\uparrow}$ *search-window row, recording the countable dense or finite-window probes available to a distance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to each positive distance query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison data produced from W and D.*
6. *E is the real distance seal row, read only after the metric source, subset admission, search windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *K is the optional* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison boundary for compact or hyperspace consumers; it records only the finite directed comparisons visible through the preceding rows.*
8. *H records componentwise* hsame *transport for the metric source, subset admission, search windows, dyadic ledger, regular readback, real seal, Hausdorff boundary, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an ambient point and tolerance request through subset admission, separability windows, dyadic comparison, regular rational readback, real sealing, and any Hausdorff-facing comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSubset}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, W, D, R, E, K, H, C, P, N$. It has no coordinate for a host subset object, extensional equality of subsets, a selected nearest point, quotient real equality, countable choice, or an ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate single carrier alignment`

Theorem 4321.2 (LocatedSubset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, S, W, D, R, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, optional Hausdorff comparison routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4374.1. The displayed packet lists exactly the rows used by an ambient point, subset-membership, tolerance, rational-readback, real-seal, or Hausdorff-facing request.

The metric and subset rows M, S are read first. A distance query can inspect the ambient space only through $\mathsf{RealMetric}^{\uparrow}$, and it can inspect membership only through the displayed admission row S.

The search, tolerance, and readback rows W, D, R then supply finite witnesses for the locatedness request. The real seal E is reached only after those rows have been displayed, while K is available only as a downstream comparison boundary.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host subset, selected nearest point, quotient equality, countable-choice schedule, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4321.3 (LocatedSubset dyadic distance witness)**.** *Every distance read exported by an accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto E.$$

The read may be handed to a $\mathsf{HausdorffMetric}^{\uparrow}$ *consumer only after the same route has produced the displayed real-distance seal, transported by H and replayed by C.*

Proof. Begin with a point-to-subset distance request. The ambient metric row M supplies the only metric observations, and the subset-admission row S supplies the only visible membership data.

The finite search window W names the inspected probes. The dyadic ledger D fixes the positive tolerance requests, and the regular rational row R records the resulting readback windows.

The real-distance seal E is therefore downstream of M, S, W, D, R. Any Hausdorff-facing row K can compare only the finite distance data already sealed by this route.

Transport and replay do not add another distance object. The componentwise hsame row H, the displayed Cont route C, the Pkg provenance P, and the local NameCert row N preserve the

same witness surface, so no nearest-point selector or host infimum can bypass the dyadic distance route. □

Falsifiable prediction: Every accepted LocatedSubsetUp distance read exposes RealMetricUp, subset admission, RealSeparabilityUp windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, and a real-distance seal before any HausdorffMetricUp or completion-facing consumer reads the packet.

Independence witness: realmetric, hausdorffmetric, realseparability, dyadicratcore, regseqrat: none of these siblings alone carries the full located-subset packet with subset admission, finite search windows, dyadic tolerance rows, regular rational readback, real-distance sealing, optional Hausdorff comparison, transport, replay, package provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}$.

Constructive story: A $\mathsf{LocatedSubset}^{\uparrow}$ packet is generated from a RealMetricUp source, displayed subset admission, RealSeparabilityUp search windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a real-distance seal, optional HausdorffMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4374.1, Definition 4374.2, and Definition 4374.3 seed the finite located-subset interface for real-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host subset extensionality, selected nearest points, quotient subset equality, countable choice, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, Hausdorff comparison routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSubsetUp.

Chapter 4322

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 4322.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 4322.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 4322.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 4322.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 4322.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 4322.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 4323

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 4323.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 4323.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 4323.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 4324

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4324.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 4324.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4324.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 4325

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 4325.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 4325.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 4325.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 4325.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 4325.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 4325.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 4326

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 4326.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4326.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 4326.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 4326.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 4327

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 4327.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 4327.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 4327.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 4327.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 4328

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4328.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4328.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4328.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 4328.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 4328.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 4328.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 4328.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 4328.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 4328.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 4328.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 4328.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 4328.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 4328.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 4328.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 4328.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 4329

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 4329.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4329.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 4329.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 4329.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 4329.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 4329.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 4329.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 4329.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 4329.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 4329.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 4330

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 4330.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 4330.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4330.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 4330.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 4331

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 4331.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 4331.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 4331.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let $L = (S, Q, I, U, F, K, T, H$ be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, seed)

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

$\mathsf{RieszThorinInterpolation}^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 4332

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 4332.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4332.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4332.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 4332.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 4333

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4333.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 4333.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 4333.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 4333.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 4334

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4334.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4334.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4334.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4335

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4335.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 4335.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4335.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 4335.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 4336

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4336.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4336.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4336.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4336.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4337

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4337.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4337.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4337.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4338

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4338.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4338.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4338.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4338.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4339

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4339.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4339.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4339.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4340

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4340.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4340.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4340.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4341

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4341.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 4341.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4342

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4342.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4342.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4342.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4343

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4343.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4343.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4343.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4343.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4344

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4344.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4344.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4344.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4345

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4345.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4345.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4345.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4346

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4346.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 4346.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4346.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4347

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4347.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4347.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4347.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4347.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4348

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4348.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4348.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4348.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4348.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4349

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4349.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* Dendrite$^{\uparrow}$, CompactMetric$^{\uparrow}$, LocallyConnected$^{\uparrow}$, PeanoContinuum$^{\uparrow}$, Tree$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* NameCert$_{\text{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4349.2 (Dendrite NameCert obligations)**.** *For every accepted* Dendrite$^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local NameCert$_{\text{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4349.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4350

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4350.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4350.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4350.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4350.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4350.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4351

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4351.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4351.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4351.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4352

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4352.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4352.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4352.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4352.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4352.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4352.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4353

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4353.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4353.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4353.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4354

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4354.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4354.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4354.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4355

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4355.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4355.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4355.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4355.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

The $\mathsf{EquicontinuousSequence}^{\uparrow}$ hub reads the shared-modulus lattice of chapter 4287 before routing compact finite-net and Arzela–Ascoli consumers through its own child files.

Chapter 4356

A Concrete Naming Certificate for $\mathsf{EquicontinuousSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{EquicontinuousSequence}^{\uparrow}$ packet records a sequence-indexed equicontinuity surface as finite NameCert data. It sits over the equicontinuity row of chapter 4076, the finite-net selector of chapter 3358, the finite Arzela–Ascoli modulus handoff of chapter 3568, the compact metric source of chapter 129, the displayed continuous-map row of chapter 130, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet gives compact-uniformity and Arzela–Ascoli consumers a finite sequence carrier instead of repeating family-level rows; it is not a host function-space compactness theorem, not countable choice, and not an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 4356.1 (Equicontinuous sequence carrier)**.** *An* $\mathsf{EquicontinuousSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, S, W, Q, R, A, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite cover and metric inspection data bound the common domain.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the displayed maps inspected by the sequence.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records which sequence indices and domain windows are read.*
4. *W is the shared equicontinuity-window row, displayed as a finite request schedule over K, F, S.*
5. *Q is the* $\mathsf{Equicontinuity}^{\uparrow}$ *row asserting that the schedule W is common to the inspected sequence window.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint and modulus values reached through the same finite windows.*
7. *A is the* $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$ *handoff row, available only after K, F, S, W, Q, R have been displayed.*
8. *T is the* $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$ *selector row, recording the finite net queries permitted by K and W.*

9. *H records componentwise* hsame *transport for compact source, map row, stream windows, common modulus schedule, equicontinuity row, regular readback, Arzela–Ascoli handoff, finite-net selector, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a sequence-window request through $K, F, S, W, Q, R, A, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{EquicontinuousSequence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{EquicontinuityFiniteNetSelector}^{\uparrow}$, $\mathsf{ArzelaAscoliFiniteModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuousSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, F, S, W, Q, R, A, T, H, C, P, N$. It has no coordinate for a host function sequence, quotient equality of maps, countable choice, open-cover compactness, selected subsequences, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4356.2 (Equicontinuous sequence NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuousSequence}^{\uparrow}$ *carrier*

$$E = (K, F, S, W, Q, R, A, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuousSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, displayed continuous-map row, finite stream windows, shared modulus schedule, equicontinuity row, regular rational readback, Arzela–Ascoli finite-modulus handoff, finite-net selector, componentwise transport, replay, provenance, and local naming rows. No obligation may replace these rows by a host function-space object, quotient map equality, a selected subsequence, countable choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4356.1. The analytic rows are exactly K, F, S, W, Q, R, A, T: compact source, displayed maps, finite sequence windows, the common schedule, the equicontinuity row, regular readback, the finite-modulus handoff, and the finite-net selector.

The common modulus is visible only as W and Q. Since the packet stores no host function sequence and no quotient equality of maps, every local obligation must read the displayed continuous-map and stream-window rows before it can compare sequence entries.

The compactness-facing rows are A and T. They are downstream of K, F, S, W, Q, R and therefore cannot supply an independent subsequence selection, countable-choice principle, open-cover compactness theorem, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. These rows preserve the displayed route but do not add any excluded object, so the listed rows exhaust the local NameCert surface. □

Theorem 4356.3 (Equicontinuous sequence finite window modulus)**.** *Every modulus read exported by an accepted* $\mathsf{EquicontinuousSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{Equicontinuity}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Thus a consumer may inspect only the displayed compact-metric windows, the continuous-map row, the finite sequence-window schedule, and the regular rational readback attached to that schedule. It gains no pointwise-only family, quotient function space, selected subsequence, countable-choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with a modulus request. The carrier first exposes the compact source K, the displayed map row F, and the stream-window row S, so the request is confined to finitely many sequence and domain windows.

The common schedule W is then certified by the equicontinuity row Q. Its values are read through the same finite windows by the regular rational row R, not by a host function-space metric or a quotient presentation of maps.

Because A and T are downstream consumers of the same schedule, they cannot create another modulus source. Transport and replay through H and C preserve the route, while P and N only record provenance and local naming. □

Theorem 4356.4 (Equicontinuous sequence Arzela–Ascoli handoff)**.** *Every Arzela–Ascoli or compact-uniformity consumer of an accepted* $\mathsf{EquicontinuousSequence}^{\uparrow}$ *carrier must read the finite route*

$$K, F, S, W, Q, R, T, A$$

before using a common modulus or finite net. The handoff supplies a sequence-indexed finite carrier for later compactness arguments, while refusing host subsequence selection, quotient equality of functions, countable choice, open-cover compactness, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An Arzela–Ascoli consumer starts with the compact source K and displayed maps F. The finite sequence windows S determine which entries and domain windows are visible.

The common schedule W, equicontinuity row Q, and regular rational readback R provide the only shared modulus surface. The finite-net selector T can query only that visible schedule over the compact metric source.

The finite-modulus handoff A is therefore a repackaging of K, F, S, W, Q, R, T, not an independent compactness principle. Since no carrier row stores a selected subsequence, quotient function equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the handoff exports only the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{EquicontinuousSequence}^{\uparrow}$ read must expose compact metric data, a ContinuousMap row, finite StreamName windows, a shared Equicontinuity schedule, RegSeqRat readback, EquicontinuityFiniteNetSelector data, ArzelaAscoliFiniteModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a compact-uniformity consumer may read a common modulus.

Independence witness: equicontinuity, equicontinuity-finite-net-selector, arzela-ascoli-finite-modulus, compactmetric, continuousmap, streamname, regseqrat, real: $\mathsf{EquicontinuousSequence}^{\uparrow}$ is not bijective to any listed sibling because it binds a sequence-window schedule to a common equicontinuity row and compact finite-net handoff before any Arzela–Ascoli consumer can read the modulus.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuousSequence}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuousSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4356.1, Definition 4356.2, Definition 4356.3, and Definition 4356.4 seed the finite sequence-indexed equicontinuity carrier and its compact-uniformity handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuousSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space compactness, quotient equality of maps, selected subsequences, countable choice, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window visibility, common modulus exposure, regular rational readback, finite-net selector exactness, Arzela–Ascoli handoff nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.EquicontinuousSequenceUp.

Constructive story: An $\mathsf{EquicontinuousSequence}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMap row, finite StreamName windows, shared equicontinuity schedule, EquicontinuityUp row, RegSeqRat readback, EquicontinuityFiniteNetSelectorUp row, ArzelaAscoliFiniteModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Every $\mathsf{EquicontinuousSequence}^{\uparrow}$ read consumes the family-level modulus discipline of chapter 413 and the sequence-level uniform route of chapter 4287 before any compactness or Ascoli-facing consumer may inspect it.

Theorem 4356.5 (Equicontinuous sequence uniform lattice route)**.** *Every compact-uniformity consumer of an accepted* $\mathsf{EquicontinuousSequence}^{\uparrow}$ *carrier reads the same displayed rows as the shared-modulus route of chapter 4287: the* $\mathsf{CompactMetric}^{\uparrow}$ *source row,* $\mathsf{ContinuousMap}^{\uparrow}$ *sequence-entry row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{Equicontinuity}^{\uparrow}$ *common schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The* $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ *sibling supplies the shared-modulus boundary, while chapter 129, chapter 301, and chapter 38 fix the compact, rational, and real-facing rows. Thus the route exports a finite compact-uniform sequence lattice, not a host function-space compactness theorem, quotient equality of maps, selected subsequence, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 4356.1. Its compact-uniformity data are the compact metric row K, displayed map row F, finite stream-window row S, shared schedule W, equicontinuity row Q, regular readback row R, finite-net selector T, and Arzela–Ascoli handoff A.

The $\mathsf{UniformlyContinuousSequence}^{\uparrow}$ route of chapter 4287 names the sibling boundary where a sequence of displayed maps shares one modulus row. The present packet reads that boundary through W, Q over the same K, F, S window, so a compact-uniformity consumer cannot detach the modulus from the compact source or from the inspected sequence entries.

Finally, the compact source, rational readback, and real seal remain anchored in chapter 129, chapter 301, and chapter 38. Transport, replay, provenance, and local naming preserve the same finite route, and none of those rows adds a host function space, quotient equality, selected subsequence, countable-choice schedule, or ambient completeness principle. □

Chapter 4357

A Concrete Naming Certificate for $\mathsf{DecimalIntervalReal}^{\uparrow}$

Bishop–Bridges constructive analysis uses rational and decimal brackets as finite evidence before a real name is treated as complete. The $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet records this decimal-bracket carrier as BEDC NameCert data. It sits after the decimal prefix surface of chapter 1417, the finite decimal stream of chapter 4191, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the located interval source of chapter 1709. The packet gives L10 Real and Completion consumers a finite decimal interval carrier without quotient reals, trichotomy, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4357.1 (Decimal interval real carrier)**.** *A* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, B, D, W, Q, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the lower finite decimal-prefix row, read through* $\mathsf{DecimalExpansion}^{\uparrow}$ *and the finite* $\mathsf{DecimalStream}^{\uparrow}$ *schedule.*
2. *B is the upper finite decimal-prefix row, read through the same decimal surfaces and carrying the advertised bracket side.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger translating decimal place values into dyadic comparison margins.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule opened for endpoint-prefix reads, bracket comparison, and downstream replay.*
5. *Q is the bracket-order row, recording that the lower decimal prefix and upper decimal prefix are compared only through the displayed tolerance ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the endpoint windows, decimal prefix rows, dyadic ledger, and bracket-order row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the regular rational readback has been displayed.*
8. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *handoff row that consumes the same endpoint, tolerance, readback, and seal rows as a located interval boundary.*

9. *H records componentwise* hsame *transport for endpoint prefixes, tolerance rows, windows, bracket order, regular readback, real seal, located interval handoff, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a decimal-bracket request through endpoint prefixes, dyadic comparison, finite windows, regular readback, real sealing, and located interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{DecimalIntervalReal}^{\uparrow}$, $\mathsf{DecimalExpansion}^{\uparrow}$, $\mathsf{DecimalStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, D, W, Q, R, E, L, H, C, P, N$. It has no coordinate for a quotient real, a trichotomy decision, a selected decimal representative, host endpoint equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier`
Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealTasteGate single carrier alignment`

Theorem 4357.2 (Decimal interval real NameCert obligations)**.** *For every accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the lower decimal prefix, upper decimal prefix, dyadic tolerance ledger, stream windows, bracket-order row, regular rational readback, real seal, located interval handoff, transport, replay, provenance, and local naming rows;*
2. *decimal-prefix visibility, requiring endpoint reads to enter through A and B before any bracket comparison is inspected;*
3. *tolerance discipline, requiring the place-value comparison to use the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D and no hidden real-order oracle;*
4. *window discipline, requiring each endpoint and comparison read to be scheduled by the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback order, requiring R to consume A, B, D, W, Q before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *located-interval handoff, requiring L to consume the same endpoint, tolerance, readback, and seal route rather than a host interval;*
7. *componentwise* hsame *transport and displayed* Cont *replay for exactly the rows carried by H and C;*
8. *ledger non-escape, excluding quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4357.1. The projection exposes exactly the two decimal endpoint-prefix rows, the dyadic tolerance ledger, the finite window schedule, the bracket-order row, the regular readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming.

The endpoint reads are visible before comparison. Rows A and B carry the finite decimal prefixes, while D supplies the dyadic margin used to compare their place-value readings. The row W schedules the inspected prefixes and comparison windows, so the bracket-order row Q has no access to a hidden real-order decision.

The real-facing route is downstream of the finite bracket data. Row R is read only after A, B, D, W, Q have been displayed, and row E seals the result only after that regular rational readback.

Finally, the located interval handoff L consumes the same displayed route, while H, C, P, N transport, replay, source, and name the packet. Since no remaining coordinate stores quotient equality, trichotomy, a selected decimal representative, host endpoint equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.DecimalIntervalRealUp.DecimalIntervalRealCarrier semantic name certificate`

Theorem 4357.3 (Decimal interval real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{LocatedRealInterval}^{\uparrow}$ *read exported by an accepted* $\mathsf{DecimalIntervalReal}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DecimalExpansion}^{\uparrow},\ \mathsf{DecimalStream}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow}.$$

Equivalently, a downstream Real and Completion consumer receives the two decimal endpoint-prefix rows, the dyadic comparison ledger, the finite window row, the regular rational readback, the terminal real seal, the located interval handoff, transport, replay, provenance, and local naming before it can use the bracket.

Proof. Start with an accepted carrier $J = (A, B, D, W, Q, R, E, L, H, C, P, N)$. The only real-facing row is E, and the only located-interval row is L.

The definition places E after the endpoint-prefix rows A, B, the dyadic tolerance ledger D, the window schedule W, the bracket-order row Q, and the regular rational readback R. Thus a $\mathsf{Real}^{\uparrow}$ consumer must first inspect the decimal, dyadic, window, and readback rows.

The located interval handoff is not an alternate route. Row L consumes the same A, B, D, W, Q, R, E path, and componentwise transport through H preserves that path under hsame.

The replay, provenance, and local naming rows only record the displayed carrier. They do not add quotient reals, trichotomy, selected representatives, host endpoint equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the displayed route is the whole exported handoff. □

Falsifiable prediction: Every accepted $\mathsf{DecimalIntervalReal}^{\uparrow}$ read displays two finite decimal-prefix rows, one dyadic comparison ledger, one finite StreamName window row, one RegSeqRat readback row, one RealUp seal row, one LocatedRealIntervalUp handoff row, and local transport, replay, provenance, and naming rows before it is consumed as a decimal bracket.

Independence witness: decimalexpansion, decimal-stream, dyadicratcore, streamname, regseqrat, real, located-real-interval: $\mathsf{DecimalIntervalReal}^{\uparrow}$ is not bijective to any listed sibling because it binds two ordered decimal endpoint-prefix rows, one dyadic comparison ledger, finite observation windows, regular rational readback, RealUp sealing, located interval handoff, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted bracket packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}$.

Constructive story: A $\mathsf{DecimalIntervalReal}^{\uparrow}$ packet is generated from lower and upper decimal-prefix rows, DecimalStream windows, a DyadicRatCore comparison ledger, a StreamName schedule, a RegSeqRat readback, a RealUp seal, a LocatedRealIntervalUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4357.1, Definition 4357.2, and Definition 4357.3 seed the finite decimal-bracket route over DecimalExpansionUp, DecimalStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, LocatedRealIntervalUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DecimalIntervalReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.DecimalIntervalRealUp`

Bridge status: none

Not claimed: This chapter does not claim quotient reals, trichotomy, selected decimal representatives, host endpoint equality, countable choice, Lean verification, bridge checking, or ambient $\mathrm{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, decimal-prefix visibility, dyadic tolerance discipline, finite-window discipline, regular-readback order, Real seal handoff, located interval handoff, transport, replay, provenance, and ledger non-escape for BEDC.Derived.DecimalIntervalRealUp.

Chapter 4358

A Concrete Naming Certificate for $\mathsf{AttouchWetsTopology}^{\uparrow}$

Following Attouch and Wets on variational convergence, the $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet records local Hausdorff-window convergence for closed metric figures as finite BEDC NameCert data over the metric-completion and L10 real-name surface. It sits after the metric carrier of chapter 57, the Hausdorff-distance surface of chapter 3653, the terminal real seal of chapter 38, the regular rational readback of chapter 301, and the finite stream windows of chapter 126. The packet names the finite observation topology; it does not import host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4358.1 (Attouch-Wets topology carrier)**.** *An* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row whose metric, point windows, and bounded-window radii are the only ambient metric data available to the packet.*
2. *H is the* $\mathsf{HausdorffDistance}^{\uparrow}$ *local-window row, read only on displayed bounded metric windows rather than on an ambient hyperspace.*
3. *D is the dyadic tolerance ledger for window radius, Hausdorff comparison margin, and late-window precision.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, bounded windows, comparison stages, and requested precision cells.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational Hausdorff-window comparisons obtained from M, H, D, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the regular rational readback has been displayed.*
7. *Q records componentwise* hsame *transport for metric source rows, Hausdorff-window rows, dyadic ledgers, stream windows, regular readbacks, real seals, replay rows, provenance, and local naming.*
8. *T records displayed* Cont *replay from an Attouch–Wets observation request through M, H, D, W, R, E.*

9. *C is the local bounded-window classifier row comparing packets only through the displayed metric source, Hausdorff-window, tolerance, stream, readback, seal, transport, replay, provenance, and naming coordinates.*

10. *P is the* Pkg *provenance over* $\mathsf{AttouchWetsTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *row.*

The classifier has no coordinate for global closed-set topology, open-cover compactness, selected nearest points, quotient hyperspace equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AttouchWetsTopologyUp.AttouchWetsTopologyTasteGate single carrier alignment`

Theorem 4358.2 (Attouch-Wets topology NameCert obligation surface)**.** *For every accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier*

$$A_{\mathrm{aw}} = (M, H, D, W, R, E, Q, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *expose the* $\mathsf{Met}^{\uparrow}$ *source row M before any local Hausdorff comparison is read;*

2. *display the* $\mathsf{HausdorffDistance}^{\uparrow}$ *window row H only on bounded metric windows named by the packet;*

3. *bind all numerical precision to the dyadic tolerance ledger D, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*

4. *preserve the same finite route by componentwise* hsame *transport Q and displayed* Cont *replay T;*

5. *source and name the same packet through the bounded-window classifier C,* Pkg *provenance P, and the local* NameCert *row N.*

No obligation row is stored for a host hyperspace topology, arbitrary closed-set compactness, selected representatives, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4358.1. The metric row M is the only source of ambient distance data, and the Hausdorff row H is read only after the bounded window has been displayed.

The numerical part is downstream of that window. The tolerance ledger D, finite window row W, regular readback R, and real seal E are the only precision and value rows available to an observation request.

Transport and replay keep the same route. Rows Q and T preserve the displayed metric, Hausdorff, tolerance, window, readback, and seal data, while C, P, N classify, source, and name the finite packet.

Since the carrier contains none of the excluded topology, compactness, selection, quotient, choice, or ambient-completeness coordinates, the displayed obligations exhaust the local NameCert surface. □

Theorem 4358.3 (Attouch-Wets local Hausdorff window)**.** *Every topology-facing read exported by an accepted* $\mathsf{AttouchWetsTopology}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{HausdorffDistance}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}.$$

Thus a downstream metric-completion consumer receives the displayed metric source, bounded local Hausdorff window, finite observation schedule, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming before it can use Attouch–Wets convergence data.

Proof. Start with a topology-facing read of the accepted carrier. The only available metric source is M, and the only Hausdorff comparison row is the local-window row H.

The observation is then forced through D, W, R, E. The dyadic ledger fixes the requested tolerance, the stream row displays the inspected bounded windows, the regular rational row records the readback, and the real row seals the reported Hausdorff-window value.

Finally Q, T, C, P, N transport, replay, classify, source, and name that same finite route. Because no row supplies host closed-set topology, arbitrary hyperspace compactness, selected nearest points, quotient closed-set equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every exported topology-facing read factors through the displayed local Hausdorff window. □

Closure status for $\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}$.

Constructive story: An $\mathsf{AttouchWetsTopology}^{\uparrow}$ packet is generated from a MetricSpace source row, a HausdorffDistance bounded-window row, a dyadic tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, a bounded-window classifier row, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4358.1, Definition 4358.2, and Definition 4358.3 seed the local Hausdorff-window topology route over MetricSpaceUp, HausdorffDistanceUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AttouchWetsTopology}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.AttouchWetsTopologyUp`

Bridge status: none

Not claimed: This chapter does not claim host closed-set topology, arbitrary hyperspace compactness, quotient closed sets, selected nearest points, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded Hausdorff-window exposure, finite-window scheduling, regular readback, Real sealing, transport, replay, classifier exactness, provenance, and local naming for BEDC.Derived.AttouchWetsTopologyUp.

Chapter 4359

A Concrete Naming Certificate for $\mathsf{CantorIntersectionMetric}^{\uparrow}$

Following Cantor's intersection theorem and Bishop–Bridges constructive metric completion, the $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet records the metric nested-closed-set intersection route as finite BEDC NameCert data over the L10 completion surface. It sits after the complete-metric source of chapter 128, the metric-completion route of chapter 1506, the nested closed-ball packet of chapter 3526, the Cantor complete metric route of chapter 2199, the Cauchy-filter interface of chapter 333, and the L10 real-name route through chapter 291, chapter 126, chapter 301, and chapter 38. The metric packet is the metric-completion specialization of the finite Cantor-intersection route in chapter 1421 and the theorem-facing intersection route in chapter 2318; it consumes their nested-window and diagonal-readback rows before exposing complete-metric service. The packet names a nested metric intersection handoff; it does not assert open-cover compactness, arbitrary closed-set compactness, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4359.1 (CantorIntersectionMetric carrier)**.** *A* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row whose metric and convergence service are the only complete-metric data available to the packet.*
2. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row naming the surrounding completion request before any intersection point is read.*
3. *B is the* $\mathsf{NestedClosedBall}^{\uparrow}$ *row, recording the descending closed metric ball spine and its finite containment witnesses.*
4. *K is the* $\mathsf{CantorCompleteMetric}^{\uparrow}$ *row, used only as the displayed nested closed-ball completion handoff.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row generated by the nested metric spine and read only through the displayed filter route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger, storing the shrink schedule, late-radius comparisons, and requested precision bounds.*
7. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected centers, radii, and containment data.*

8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of the displayed center schedule.*
9. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after B, K, F, D, S, R have been displayed.*
10. *H records componentwise* hsame *transport for the complete-metric source, completion route, nested-ball row, Cantor handoff, filter, dyadic ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a nested metric intersection request through $M, E, B, K, F, D, S, R, L$.*
12. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, B, K, F, D, S, R, L, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary closed-set compactness, quotient completion equality, a selected limit, host trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionMetricUp.CantorIntersectionMetricTasteGate single carrier alignment`

Theorem 4359.2 (CantorIntersectionMetric NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier*

$$I_{\mathrm{cim}} = (M, E, B, K, F, D, S, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$ *surface consists exactly of the following five obligations:*

1. *expose the complete-metric and metric-completion source rows M, E before any intersection witness is read;*
2. *display the nested metric route through B, K, F, namely the nested closed-ball spine, Cantor complete-metric handoff, and generated Cauchy-filter row;*
3. *bind all numerical readback to the L10 rows D, S, R, L, namely the dyadic tolerance ledger, finite stream windows, regular rational handoff, and terminal real seal;*
4. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
5. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row is stored for open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, trichotomy, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 4359.1. The rows M, E expose the metric and completion sources before the packet can read an intersection point.

The nested metric content is then displayed through B, K, F. The nested closed-ball row supplies the descending closed metric spine, the Cantor complete-metric row supplies the existing handoff, and the Cauchy-filter row records the generated filter route.

The L10 readback is downstream. The dyadic ledger D, stream windows S, regular rational row R, and real seal L are the only numerical coordinates available before a completion-facing consumer receives a named output.

Finally H, C, P, N transport, replay, source, and name that same finite route. Since the carrier contains none of the excluded compactness, quotient, selected-limit, trichotomy, or ambient-completeness coordinates, the five displayed obligations exhaust the local NameCert surface. □

Theorem 4359.3 (CantorIntersectionMetric complete handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{CantorIntersectionMetric}^{\uparrow}$ *carrier factors through the displayed nested metric route*

$\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{NestedClosedBall}^{\uparrow}$, $\mathsf{CantorCompleteMetric}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, S

Thus a consumer receives the complete-metric service, completion request, nested closed-ball spine, Cantor complete-metric handoff, Cauchy-filter row, dyadic tolerance ledger, finite stream windows, regular rational readback, terminal real seal, componentwise transport, replay, provenance, and local naming rows before it may use the intersection output. The handoff exports no $\mathsf{Real}^{\uparrow}$ *completeness principle, open-cover compactness theorem, arbitrary closed-set compactness, quotient stream equality, or selected limit.*

Proof. Begin with a completion-facing read of the accepted carrier. The first available rows are M and E, so the read is grounded in the displayed complete-metric service and the surrounding metric-completion request.

Next pass through the nested route B, K, F. The descending closed-ball spine is read through the $\mathsf{NestedClosedBall}^{\uparrow}$ row, the existing Cantor complete-metric handoff is read through K, and the generated filter is read through F.

The numerical output is then forced through the L10 readback rows D, S, R, L. The dyadic ledger supplies tolerance control, the stream row supplies finite windows, the regular rational row supplies readback, and the terminal real seal is reached only after those rows are present.

The structural rows H, C, P, N preserve the same route for downstream consumers. Because no row supplies full real completeness, open-cover compactness, arbitrary closed-set compactness, quotient stream equality, or a selected limit, every completion-facing read factors through the displayed nested metric route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionMetric}^{\uparrow}$ packet is generated from a CompleteMetric source row, a MetricCompletion request row, a NestedClosedBall spine, a CantorCompleteMetric handoff, a CauchyFilter row, a DyadicRatCore tolerance ledger, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, seed)

Scope closed: Definition 4359.1, Definition 4359.2, and Definition 4359.3 seed the metric nested-closed-set intersection route over displayed completion, nested-ball, Cantor handoff, filter, dyadic, stream, regular readback, real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CantorIntersectionMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary closed-set compactness, quotient completions, selected limits, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested metric inclusion, Cantor handoff, Cauchy-filter generation, dyadic shrink exactness, finite-window readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionMetricUp.

Chapter 4360

A Concrete Naming Certificate for $\mathsf{TightRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{TightRealInterval}^{\uparrow}$ packet names a finite lower-upper real interval by displaying both endpoint names and every rational cell through which those endpoints are inspected. It sits downstream of the located endpoint surface of chapter 394 and the dyadic refinement surface of chapter 306: consumers may read a tight interval only after the lower endpoint, upper endpoint, rational cell ledger, dyadic refinement ledger, stream windows, regular rational readbacks, and Real seal rows have all been displayed.

Definition 4360.1 (TightRealInterval carrier)**.** *A* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, U, Q, D, S, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the lower* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row and U is the upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger recording the finite rational windows that enclose the endpoint reads and witness lower-before-upper compatibility.*
3. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement ledger attached to the rational cells in Q.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row naming exactly the endpoint observations and interval cells inspected by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and cell refinements.*
6. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row reached from L, U, S, R, Q, D.*
7. *H records componentwise* hsame *transport for endpoints, rational cells, dyadic refinements, stream windows, readbacks, Real seals, replay, provenance, and the local name.*
8. *C records displayed* Cont *replay through $L, U, Q, D, S, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for host interval equality, chosen endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Lean4 (definition) `BEDC.Derived.TightRealIntervalUp.TightRealIntervalUp`
Lean4 ✓ `BEDC.Derived.TightRealIntervalUp.TasteGate.TightRealIntervalTasteGate single carrier alignment`

Theorem 4360.2 (TightRealInterval NameCert obligations)**.** *For an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *carrier* $T = (L, U, Q, D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower and upper* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint rows, rational and dyadic interval ledgers, finite stream windows, regular rational readbacks, paired Real seal rows, componentwise transport, continuation replay, package provenance, and local naming; endpoint-order visibility through* Q*; dyadic refinement discipline through* D*; stream-readback exactness through* S, R*; and ledger non-escape from host intervals, selected endpoints, quotient real equality, choice, trichotomy, and ambient completeness.*

Proof. Project the carrier of Definition 4360.1. The lower and upper endpoint rows, the rational cell ledger, the dyadic refinement ledger, finite windows, regular readbacks, Real seals, transport rows, replay rows, provenance row, and local naming row are all visible before any interval consumer can read the packet.

Endpoint-order visibility is supplied by Q. Since each dyadic refinement in D is attached to a rational cell, the refinement discipline cannot detach from the endpoint ledger.

Stream-readback exactness is then read through S and R. The rows H, C, P, N transport, replay, source, and name the same finite route, while the excluded interval and completeness data have no carrier coordinate. □

Theorem 4360.3 (TightRealInterval Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{TightRealInterval}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{RationalInterval}^{\uparrow},\ \mathsf{DyadicInterval}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Nam}$$

The read exposes lower and upper endpoint windows together with the rational and dyadic cell ledgers that bind them. It does not expose a host real interval, a selected endpoint value, quotient real equality, countable choice, trichotomy, or ambient real completeness.

Proof. Begin with the accepted carrier. The only seal-facing row is E, and it is displayed after L, U, Q, D, S, R in the finite packet.

Apply Definition 4360.2. Endpoint-order visibility forces the Real read to keep the rational cell ledger Q, dyadic refinement discipline forces any dyadic window through D, and stream-readback exactness forces finite observations through S, R.

Finally route through H, C, P, N. These rows preserve and name the same packet, so no host interval, endpoint selector, quotient equality, choice schedule, trichotomy row, or ambient completeness principle can enter the Real seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TightRealInterval}^{\uparrow}$ packet is generated from lower and upper $\mathsf{LocatedReal}^{\uparrow}$ endpoint rows, a $\mathsf{RationalInterval}^{\uparrow}$ cell ledger, a $\mathsf{DyadicInterval}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, paired $\mathsf{Real}^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4360.1, Definition 4360.2, and Definition 4360.3 seed the finite lower-upper Real interval packet over located endpoints, rational cells, dyadic refinements, stream windows, regular readbacks, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TightRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host interval equality, selected endpoint values, quotient real equality, trichotomy, countable choice, or ambient real completeness.

Upgrade path: Scoped closure requires a checked carrier-obligation package and a public consumer route for the same finite lower-upper interval packet.

Chapter 4361

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$

Following Bishop-style compact metric uniform continuity, the $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet records the finite handoff that transfers compact finite-net and pointwise continuity data into a uniform modulus row. It consumes the finite compact source of chapter 60, the compact metric net surface of chapter 129, the continuity interface of chapter 391, the finite stream and regular readback rows of chapter 126 and chapter 301, and the dyadic tolerance ledger of chapter 291. The packet is a finite modulus-transfer NameCert surface, not an open-cover compactness proof or a host modulus selector.

Definition 4361.1 (Compact uniform modulus transfer carrier)**.** *A* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, U, S, R, D, F, H, C, P, N)$$

with the following visible rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net source row.*
2. *G is the* $\mathsf{CompactMetric}^{\uparrow}$ *metric coverage row for the same finite centers and source distance witnesses.*
3. *U is the* $\mathsf{UniformContinuity}^{\uparrow}$ *pointwise-to-uniform continuity row consumed by the transfer.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the inspected finite centers, tolerance requests, and pointwise-radius reads.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for finite radius comparison and shrinkage.*
7. *F is the modulus-transfer ledger that records the finite lower-bound handoff into the downstream uniform-modulus consumer row.*
8. *H records componentwise* hsame *transport for compact, compact-metric, uniform-continuity, stream, readback, dyadic, transfer, replay, provenance, and local-name rows.*
9. *C records displayed* Cont *replay through $K, G, U, S, R, D, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformContinuity}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *row.*
The carrier has no coordinate for open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate`
`single carrier alignment`

Theorem 4361.2 (Compact uniform modulus transfer obligations)**.** *For an accepted* CompactUniformModulusTrans *carrier $M = (K, G, U, S, R, D, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact finite-net row, compact-metric coverage row, uniform-continuity row, stream windows, regular rational readbacks, dyadic tolerance ledger, modulus-transfer ledger, componentwise transport, continuation replay, package provenance, and local naming; source lock between K, G, U; finite-window exactness through S, R; dyadic tolerance exactness through D; and non-escape from open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient equality, choice, and ambient completeness.*

Proof. Project the carrier of Definition 4361.1. The compact row, compact-metric row, continuity row, stream windows, readbacks, dyadic tolerance ledger, transfer ledger, transport rows, replay rows, provenance row, and local name are displayed as one finite packet.

The source lock is the shared read of K, G, U. A transfer consumer cannot use the compact finite-net row without the compact-metric distance surface and the continuity row that supplies pointwise radius data.

Finite-window and dyadic exactness are read through S, R, D. The ledger F can therefore transfer only the displayed finite lower-bound data, and H, C, P, N preserve that route without adding any excluded compactness or choice principle. □

Theorem 4361.3 (Compact uniform modulus transfer handoff)**.** *Every accepted* $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ *packet hands a downstream uniform-modulus consumer only the finite route*

$$\mathsf{Compact}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{UniformContinuity}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont}$$

The handoff records compact finite-net coverage, pointwise continuity radius reads, finite stream windows, regular rational readbacks, dyadic tolerance comparison, and the transfer ledger. It exports no open-cover compactness theorem, selected net center, selected Lebesgue number, host modulus function, quotient equality, countable choice, or ambient completeness.

Proof. Start from the transfer obligations of Definition 4361.2. They identify the compact, compact-metric, continuity, stream, readback, dyadic, and transfer rows as the only NameCert-readable data.

Route the downstream consumer through F. The ledger can be consumed only after K, G, U, S, R, D have supplied the finite-net, pointwise-radius, window, readback, and tolerance rows.

Finally use H, C, P, N to transport, replay, source, and name the same packet. Since the excluded compactness, selection, equality, choice, and completeness data have no coordinate in the carrier, the downstream handoff is exactly the displayed finite modulus-transfer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusTransfer}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformModulusTransfer}^{\uparrow}$ packet is generated from compact finite-net rows, compact-metric coverage rows, uniform-continuity rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, a modulus-transfer ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, seed)

Scope closed: Definition 4361.1, Definition 4361.2, and Definition 4361.3 seed the finite compact-net modulus-transfer packet for uniform-continuity consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformModulusTransfer}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformModulusTransferUp.CompactUniformModulusTransferTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected centers, selected Lebesgue numbers, host modulus functions, quotient point equality, countable choice, or ambient metric completeness.

Upgrade path: Scoped closure requires a checked transfer-obligation package and a public uniform-modulus consumer route for the same finite packet.

Chapter 4362

A Concrete Naming Certificate for $\mathsf{RealIntervalApartness}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntervalApartness}^{\uparrow}$ packet records located separation of two real intervals by a displayed positive dyadic gap rather than by a trichotomy decision on completed real endpoints. It sits after the terminal real-name seal of chapter 38, the located endpoint interval surface of chapter 1709, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, and the separability boundary of chapter 1809. The packet names interval separation only through visible endpoint rows, gap rows, transport, replay, provenance, and local naming.

Definition 4362.1 (RealIntervalApartness carrier)**.** *A* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I and J are the two* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint packets whose lower and upper endpoint rows are exposed before any apartness read.*
2. *L, U are the lower and upper* $\mathsf{Real}^{\uparrow}$ *seal rows for I, and L', U' are the corresponding* $\mathsf{Real}^{\uparrow}$ *seal rows for J.*
3. *G is a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness, attached either to the displayed comparison $U + G \leq L'$ or to the displayed comparison $U' + G \leq L$.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the endpoint observations and gap inspections used by the packet.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those endpoint windows and dyadic gap checks.*
6. *E records the* $\mathsf{RealSeparability}^{\uparrow}$ *boundary row witnessing that the two exposed endpoint intervals are separated by the displayed dyadic gap.*
7. *H records componentwise* hsame *transport for intervals, endpoint seals, dyadic gap rows, finite windows, readbacks, separability rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay through $I, J, L, U, L', U', G, S, R, E, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *row.*

The carrier has no coordinate for real trichotomy, quotient real equality, a selected endpoint representative, a host interval order, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Theorem 4362.2 (RealIntervalApartness NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier*

$$A = (I, J, L, U, L', U', G, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *admit the two located real interval packets I, J and expose their lower and upper endpoint rows through L, U, L', U';*
2. *expose the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *gap witness G and the side on which the gap separates the displayed endpoint rows;*
3. *bind every endpoint and gap inspection to the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, R;*
4. *preserve the interval, endpoint, gap, readback, and separability rows by componentwise* hsame *transport through H;*
5. *replay the same finite route through* Cont *row C, source it by* Pkg *provenance P, and name it by the local* NameCert *row N.*

No obligation row is stored for real trichotomy, quotient equality of completed real names, selected endpoint representatives, host interval order, or ambient real completeness.

Proof. Project the carrier of Definition 4362.1. The two located interval packets and their endpoint seals are the only interval data exposed to a consumer.

The separation datum is the row G. Positivity is read as a dyadic-row property, and the side of separation is part of the displayed ledger; no trichotomy comparison is consulted.

The numerical readback is forced through S and R, then through the separability row E. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since none of the excluded host-order, quotient-equality, representative-selection, trichotomy, or completeness data occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.RealIntervalApartnessUp.TasteGate.RealIntervalApartnessTasteGate single carrier alignment`

Theorem 4362.3 (RealIntervalApartness separability route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{RealIntervalApartness}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, hsame, Cont, Pkg, Na

The route exports located interval separation only as a positive dyadic gap between displayed endpoint rows. It exports no real trichotomy, quotient real equality, host interval order, selected endpoint representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the NameCert obligations of Definition 4362.2. They force the two located interval packets and endpoint seals to be visible before a consumer may read an apartness claim.

Next read the dyadic gap row and its finite endpoint inspections through the stream and regular-rational rows. The separability row is therefore attached to the displayed positive gap, not to a host comparison of completed reals.

Finally route through transport, replay, provenance, and local naming. These rows preserve the same finite packet, so the consumer receives located interval apartness without importing any excluded trichotomy, quotient, selection, host-order, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}$.

Constructive story: A $\mathsf{RealIntervalApartness}^{\uparrow}$ packet is generated from two LocatedRealInterval endpoint packets, Real endpoint seal rows, a positive DyadicRatCore gap witness, finite StreamName windows, RegSeqRat readback, a RealSeparability boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4362.1, Definition 4362.2, and Definition 4362.3 seed the located interval separation packet over displayed endpoint, dyadic gap, finite-window, regular-readback, Real seal, separability, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalApartness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim real trichotomy, quotient real equality, selected endpoint representatives, host interval order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, positive-gap exposure, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.RealIntervalApartnessUp.

Chapter 4363

A Concrete Naming Certificate for $\mathsf{PartitionBasinPolynomial}^{\uparrow}$

Following finite enumerative combinatorics, the $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ packet records the partition-basin product formula of Definition 9462.1736 as displayed NameCert data over block sizes, legal decorations, defect exponents, incidence totals, transport, replay, provenance, and local naming. It sits after the partition carrier of chapter 201, the defect-flow partition skeleton of chapter 4236, and the partition-core incidence certificate of chapter 4214. The packet names a finite polynomial enumeration surface; it is not a biological stochastic model, a host generating-function oracle, or a quotient-level semantic identification.

Definition 4363.1 (PartitionBasinPolynomial carrier)**.** *A* $\mathsf{PartitionBasinPolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the finite block-count row for the partition core, and S is the finite block-size ledger $(s_1, \ldots, s_m)$.*
2. *D is the legal-decoration row, recording for each coordinate of block i the admissible subset of the other $m-1$ rows and the private-coordinate exclusion.*
3. *E is the defect-exponent ledger that assigns exponent weight q^r to a coordinate receiving r additional rows.*
4. *F is the product-formula row*
$$\prod_{i=1}^{m} \left[(1+q)^{(m-1)s_i} - \left((1+q)^{m-1} - 1 \right)^{s_i} \right].$$
5. *Q is the four-coordinate specialization row for the five partition types 4, $3+1$, $2+2$, $2+1+1$, and $1+1+1+1$.*
6. *I is the incidence-total row that reads the quotient-level and codon-level totals from the displayed specialization.*
7. *T records the downstream defect-flow handoff to the partition skeleton and incidence certificate.*
8. *H records componentwise* hsame *transport for block sizes, legal decorations, defect exponents, product rows, specialization rows, incidence rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay through $M, S, D, E, F, Q, I, T, H$.*
10. *P is the* Pkg *provenance over* PartitionBasinPolynomial$^{\uparrow}$, Partition$^{\uparrow}$, DefectFlowPartitionSkeleton$^{\uparrow}$, PartitionCoreIncidenceCertificate$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a stochastic biology model, a host polynomial oracle, a quotient-semantic equality, an unbounded partition enumeration, or a hidden recurrence outside the displayed finite rows.

Lean4 ✓ `BEDC.Derived.PartitionBasinPolynomialUp.PartitionBasinPolynomialTasteGate single carrier alignment`

Theorem 4363.2 (PartitionBasinPolynomial NameCert obligations)**.** *For every accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier*

$$B = (M, S, D, E, F, Q, I, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$ *surface consists exactly of block-count and block-size admission through M, S, legal-decoration admission through D, defect-exponent admission through E, product-formula exposure through F, four-coordinate specialization through Q, incidence-total readback through I, defect-flow handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4363.1. The block-count and block-size rows supply the finite partition core on which the enumeration is read.

The decoration and defect rows are then exposed coordinatewise. For block i, the all-decoration contribution is $(1+q)^{(m-1)s_i}$, and the excluded no-private-coordinate contribution is $\left((1+q)^{m-1}-1\right)^{s_i}$.

Multiplying the legal block rows gives the product row F. The specialization row Q evaluates the same finite formula on the five four-coordinate partition types, and the incidence row I sums the displayed rows with their finite multiplicities.

Finally T, H, C, P, N route the same polynomial surface to the defect-flow and incidence packets. Because no excluded stochastic, host-oracle, quotient-semantic, unbounded-enumeration, or hidden-recurrence coordinate occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4363.3 (PartitionBasinPolynomial product specialization)**.** *Every defect-flow consumer of an accepted* PartitionBasinPolynomial$^{\uparrow}$ *carrier receives the product formula row for $\mathcal{B}_P(q)$ and the four-coordinate specialization row before reading quotient-level or codon-level incidence totals. The consumer sees only finite block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and local NameCert rows.*

Proof. Start from the obligations of Definition 4363.2. They force the block-size ledger and legal-decoration rows to be displayed before the product formula can be consumed.

The four-coordinate row is a finite specialization of the same product row. Its incidence totals are read only after the partition types and finite multiplicities have been displayed.

The handoff through T, H, C, P, N preserves the same finite surface for the defect-flow skeleton and partition-core incidence certificate. Thus no stochastic model, host generating-function oracle, quotient-semantic equality, or hidden enumeration can be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$.

Constructive story: A PartitionBasinPolynomial$^{\uparrow}$ packet is generated from block-count rows, block-size ledgers, legal-decoration rows, defect-exponent ledgers, a product-formula row, a four-coordinate specialization row, incidence-total rows, a defect-flow handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, seed)

Scope closed: Definition 4363.1, Definition 4363.2, and Definition 4363.3 seed the finite partition-basin polynomial packet over displayed block-size, decoration, defect, product, specialization, incidence, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PartitionBasinPolynomial}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a stochastic biology model, host generating-function oracle, quotient-level semantic equality, unbounded partition enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, product-formula exposure, four-coordinate specialization, incidence readback, transport, replay, provenance, and local naming for BEDC.Derived.PartitionBasinPolynomialUp.

Chapter 4364

A Concrete Naming Certificate for $\mathsf{DistanceToSet}^{\uparrow}$

Following Bishop and Bridges on located subsets in metric spaces, the $\mathsf{DistanceToSet}^{\uparrow}$ packet records the finite route by which a point-to-set distance request is read through displayed located-radius rows and closed-ball witnesses. It sits after the metric source of chapter 57, the located-set surface of chapter 1541, the metric closed-ball boundary of chapter 1451, the located closed-ball handoff of chapter 4063, and the terminal real-name seal of chapter 38. The packet names distance-to-set as finite BEDC data; it is not a host infimum, not a selected nearest point, not quotient real equality, and not a countable-choice construction.

Definition 4364.1 (Distance-to-set carrier)**.** *A* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, A, x, r, B, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row over which point and subset reads are interpreted.*
2. *A is the* $\mathsf{LocatedSet}^{\uparrow}$ *row naming the subset by finite located-radius observations.*
3. *x is the displayed point row in the metric source.*
4. *r is the dyadic radius ledger recording the inspected tolerance windows.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness row attached to the point and radius ledger.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the exported distance name, read only after the located-set and closed-ball rows are displayed.*
7. *H records componentwise* hsame *transport for the metric source, located set, point row, dyadic ledger, closed-ball witness, Real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from the metric source through located-set exposure, point inspection, dyadic-radius accounting, closed-ball handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DistanceToSet}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LocatedSet}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{LocatedClosedBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, x, r, B, R, H, C, P, N$. It has no coordinate for a host infimum over all points of A, a selected nearest point, quotient real equality, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.DistanceToSetUp.DistanceToSetTasteGate single carrier alignment`

Theorem 4364.2 (Distance-to-set NameCert obligations)**.** *For every accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *carrier*

$$D = (X, A, x, r, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DistanceToSet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric source, located set, point row, dyadic radius ledger, closed-ball witness, Real seal, transport, replay, provenance, and local naming;*
2. *located-set exposure, requiring every subset read to pass through the* $\mathsf{LocatedSet}^{\uparrow}$ *row A before a distance row is exported;*
3. *dyadic-radius monotonicity, requiring the tolerance ledger r to preserve the displayed radius order used by the located-set and closed-ball rows;*
4. *closed-ball handoff, requiring each ball membership or exclusion read to pass through the* $\mathsf{MetricClosedBall}^{\uparrow}$ *witness B and the located closed-ball sibling surface before the Real seal is read;*
5. *Real seal non-escape, requiring R to seal only the displayed distance name and not a host infimum or quotient real;*
6. *componentwise transport, displayed replay, package provenance, and local* NameCert *naming for exactly the displayed packet.*

No DistanceToSet obligation is stored outside these rows.

Proof. Project the carrier of Definition 4364.1. The metric source is X, the located subset is A, the inspected point is x, and the finite tolerance information is held by r.

The closed-ball witness B is read only after those rows are displayed. Its radius requests are therefore attached to the same metric source, located-set row, point row, and dyadic ledger.

The Real seal R comes after the located-radius and closed-ball rows. Transport H and replay C preserve the same order, while P and N record provenance and local naming.

Since no coordinate stores a host infimum, a nearest-point selector, quotient real equality, countable choice, or ambient completeness, those objects cannot become local NameCert obligations. □

Theorem 4364.3 (Distance-to-set located-radius handoff)**.** *Every consumer of an accepted* $\mathsf{DistanceToSet}^{\uparrow}$ *packet reads distance-to-set data only through the displayed located-radius row A, the dyadic radius ledger r, the closed-ball witness B, and the Real seal R, transported by H and replayed by C. The consumer cannot bypass these rows by reading a host infimum, a selected nearest point, quotient real equality, countable choice, or an ambient completeness principle.*

Proof. Read the carrier of Definition 4364.1. The only subset coordinate is the located-set row A, and the only radius coordinate is the dyadic ledger r.

The closed-ball witness B supplies the displayed ball boundary before the Real seal R is exposed. Thus a distance consumer reaches the exported real name only through the located-radius and closed-ball rows.

Apply Definition 4364.2. The transport, replay, provenance, and local naming rows preserve the same finite packet and add no host infimum, nearest-point selector, quotient real equality, countable choice, or ambient completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}$.

Constructive story: A $\mathsf{DistanceToSet}^{\uparrow}$ packet is generated from a metric source, located set row, point row, dyadic radius ledger, closed-ball witness, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4364.1, Definition 4364.2, and Definition 4364.3 seed the finite located-radius and closed-ball interface for distance-to-set consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DistanceToSet}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host infimum construction, selected nearest points, quotient real equality, countable choice, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, located-set exposure, dyadic-radius monotonicity, closed-ball handoff, Real seal non-escape, transport stability, replay stability, provenance, and local naming for BEDC.Derived.DistanceToSetUp.

Chapter 4365

A Concrete Naming Certificate for $\mathsf{JordanCurve}^{\uparrow}$

Following Jordan and constructive topology, the $\mathsf{JordanCurve}^{\uparrow}$ packet records the finite polygonal and winding-ledger route by which a displayed simple closed curve in the real plane may be consumed as a separation boundary. It sits after the terminal real-name seal of chapter 38, the circle carrier of chapter 58, the graph and modulus rows of chapter 130, the finite compact-metric source of chapter 129, and the simplex inventory of chapter 238. The packet names a constructive finite obligation surface; it does not import classical open-cover compactness, selected complementary components, quotient path equality, or a choice-dependent separation theorem.

Definition 4365.1 (Jordan curve carrier)**.** *A* $\mathsf{JordanCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, F, P, W, O, I, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{S}^{1\uparrow}$ *source row, recording the circle parameter ledger and its local classifier.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row into the displayed* $\mathsf{Real}^{\uparrow} \times \mathsf{Real}^{\uparrow}$ *plane, with graph and modulus observations visible.*
3. *P is the finite polygonal approximation row, read through a* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision and its vertex-image ledger.*
4. *W is the winding ledger, recording the finite edge-crossing and sign-change data used to compare test points with the polygonal boundary.*
5. *O is the orientation row, recording the compatible cyclic order inherited from the* $\mathsf{S}^{1\uparrow}$ *row and the polygonal subdivision.*
6. *I is the injectivity-apartness row, stating that distinct inspected arcs carry displayed positive separation windows before they are used by the boundary read.*
7. *E is the exterior-interior ledger, recording only the finite side classification certified by the current polygonal and winding rows.*
8. *H records componentwise* hsame *transport for $S, F, P, W, O, I, E, C, Q, N$.*
9. *C records displayed* Cont *replay from circle, map, polygonal, winding, orientation, and apartness rows to the side-classification row.*

10. *Q is the* Pkg *provenance over* $\mathsf{JordanCurve}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanCurve}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, P, W, O, I, E, H, C, Q, N$. It has no coordinate for a host Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or countable-choice separation.

Lean4 ✓ `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate single carrier alignment`

Theorem 4365.2 (Jordan curve orientation ledger)**.** *For every accepted* $\mathsf{JordanCurve}^{\uparrow}$ *carrier*

$$J = (S, F, P, W, O, I, E, H, C, Q, N),$$

the orientation row O is determined by the finite $\mathsf{S}^{1\uparrow}$ *cyclic order, the* $\mathsf{SimplicialComplex}^{\uparrow}$ *subdivision in P, and the displayed* $\mathsf{ContinuousMap}^{\uparrow}$ *edge-image ledger in F. The winding ledger W can read orientation only through those rows, transported by H and replayed by C.*

Proof. Project the accepted carrier. The only circle-facing coordinate is S, and the only finite boundary skeleton is the polygonal row P together with the displayed edge images in F.

The orientation row O is therefore read by following the cyclic order of S across the subdivision cells of P. Each edge-image comparison is a finite real-plane observation supplied by F, not by an ambient topological orientation principle.

The winding ledger W consumes that ordered edge list. Componentwise transport H and replay C preserve the same ordered rows, while Q and N record provenance and local naming. Since the carrier contains no host orientation object or quotient path coordinate, the finite ledger is the entire orientation source. □

Theorem 4365.3 (Jordan curve winding separation boundary)**.** *Every side-classification read from an accepted* $\mathsf{JordanCurve}^{\uparrow}$ *packet factors through the finite route*

$$S, F, P, O, I, W \longmapsto E \longmapsto H, C, Q, N.$$

Thus the packet exposes polygonal approximation, injectivity-apartness, orientation, and winding ledgers before it names an interior or exterior row. It does not assert a blanket classical Jordan separation theorem, selected complementary components, open-cover compactness, quotient path equality, or choice-dependent planar separation.

Proof. Read the carrier of Definition 4365.1. The curve source is S, the displayed map is F, and the finite polygonal boundary is P.

Before a side row is exported, the packet requires the orientation ledger O, injectivity-apartness row I, and winding ledger W. These rows are finite BHist data: the winding read uses edge crossings and signs already attached to the polygonal boundary.

The exterior-interior row E is downstream of that route. Transport, replay, provenance, and naming through H, C, Q, N preserve the same finite packet. The excluded host theorem, selected components, open covers, quotient paths, and choice rows are not coordinates of the carrier, so they cannot be read from the local NameCert surface. □

Falsifiable prediction: A $\mathsf{JordanCurve}^{\uparrow}$ consumer that exports side separation without reading P, W, O, I refutes the carrier surface.

Independence witness: real, sone, continuousmap, compactmetric, simplicialcomplex: the sibling chapters supply real-plane coordinates, circle source data, continuous graph and modulus rows, compact-metric finite inspection, and subdivision cells separately, while $\mathsf{JordanCurve}^{\uparrow}$ binds them into the finite packet $J = (S, F, P, W, O, I, E, H, C, Q, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}$.

Constructive story: A $\mathsf{JordanCurve}^{\uparrow}$ packet is generated from a circle source row, continuous real-plane map row, finite polygonal subdivision, winding ledger, orientation row, injectivity-apartness row, side-classification ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4365.1, Definition 4365.2, and Definition 4365.3 seed the finite polygonal, winding, orientation, apartness, side-classification, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanCurve}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.JordanCurveUp.JordanCurveTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, the classical Jordan curve theorem, selected complementary components, open-cover compactness, arbitrary path lifting, quotient path equality, or choice-dependent separation.

Upgrade path: Obligation closure requires checked carrier admission, polygonal approximation exactness, winding-ledger stability, orientation transport, injectivity-apartness handoff, side-classification non-escape, replay, provenance, and local naming for BEDC.Derived.JordanCurveUp.

Chapter 4366

A Concrete Naming Certificate for MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$

The MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet realizes the normalization remaining question of chapter 9386 as finite NameCert data. It sits beside the candidate-set packet of chapter 2320 and the normalization router of chapter 1108: those siblings expose candidate evidence and route selection, while this packet records the boundary where empty-context closed typed terms may or may not enter the candidate interface. The packet names a closed-projection frontier only; it does not turn candidate-mediated normalization into an unconditional strong-normalization theorem for every closed MetaCIC term.

Definition 4366.1 (MetaCIC candidate closed-projection boundary carrier)**.** *A* MetaCICCandidateClosedProjection *carrier is a finite* BHist *packet*

$$B = (T, K, R, E, NF, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row whose projection into the candidate interface is being tested.*
2. *K is the candidate-set evidence row, read through* CandidateSetNormalization$^{\uparrow}$ *when the projection is admitted.*
3. *R is the router row, read through* MetaCICNormalizationCertificateRouter$^{\uparrow}$ *and recording which candidate, closed-candidate, or candidate-set route is exposed.*
4. *E is the bounded-normal endpoint seal row, recording the finite normal-output evidence available after candidate admission.*
5. *NF is the no-infinite-reduction row exported only downstream of K, R, E.*
6. *O is the obstruction row recording refusal when the closed term cannot be placed into the displayed candidate interface.*
7. *H records componentwise* hsame *transport for $T, K, R, E, NF, O, C, P, L$.*
8. *C records displayed* Cont *replay from the closed-term row through candidate admission, router selection, endpoint seal, no-infinite-reduction read, and obstruction retention.*

9. *P is the* Pkg *provenance over* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$, CandidateSetNormalization$^{\uparrow}$, MetaCICNormalizationCertificateRouter$^{\uparrow}$, MetaCICOpenProblemLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, R, E, NF, O, H, C, P, L$. It has no coordinate for evidence-free closed-term strong normalization, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Lean4 ✓ `BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.MetaCICCandidateClosedProjectionBoundar`
`single carrier alignment`

Theorem 4366.2 (MetaCIC candidate-set closed projection frontier)**.** *For every accepted* MetaCICCandidateClosed *carrier*

$$B = (T, K, R, E, NF, O, H, C, P, L),$$

the closed-term row T reaches a strong-normalization-facing read only through candidate evidence K, router selection R, and bounded-normal endpoint seal E. If those rows do not place T inside the candidate interface, the obstruction row O is the only exported boundary datum.

Proof. Project the carrier. The closed-term row T is present as a finite syntactic and typing packet, but it is not itself a normalization theorem.

The positive route first reads K through chapter 2320. The router row R then records the candidate, closed-candidate, or candidate-set branch through chapter 1108. Only after those rows are displayed can the endpoint seal E and no-infinite-reduction row NF be consumed.

If candidate admission is absent, O records the refusal. Transport, replay, provenance, and local naming through H, C, P, L preserve that same finite boundary, so the packet exports no evidence-free closed-term strong-normalization claim. □

Theorem 4366.3 (MetaCIC candidate-set SN handoff boundary)**.** *Every strong-normalization handoff exported by an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet is a handoff for rows already inside the displayed candidate interface. It cannot be read as a blanket theorem that all empty-context typed closed MetaCIC terms are strongly normalizing.*

Proof. Start with the frontier theorem. It places the normalization-facing route behind K, R, E and keeps the obstruction row O visible when the candidate interface is not supplied.

For an admitted row, the handoff consumes candidate evidence from CandidateSetNormalization$^{\uparrow}$ and route evidence from MetaCICNormalizationCertificateRouter$^{\uparrow}$. The no-infinite-reduction row NF is therefore downstream of the admitted candidate branch rather than a second source of normalization evidence.

For a closed term outside that interface, the carrier has only T and the obstruction row O on the boundary. The structural rows H, C, P, L transport, replay, source, and name the boundary, but they do not add subject reduction, host conversion equality, quotient reduction equality, or a global normalizer. Hence the handoff is candidate-scoped. □

Theorem 4366.4 (MetaCIC candidate-set public boundary)**.** *Any public no-infinite-reduction or strong-normalization read from an accepted* MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ *packet must cite candidate-set membership or soundness, router selection, and bounded-normal endpoint rows. A bare empty-context typed closed-term row T is insufficient: the public boundary may expose NF only downstream of the candidate-set facts named by* closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, *and* canonicalCandidateSet_sound *together with the router row R and endpoint seal E.*

Proof. Start with Definition 4366.2. It places every normalization-facing read behind the candidate evidence row K, router row R, and endpoint seal E, and leaves the obstruction row O as the only datum when candidate admission is absent.

Next apply Definition 4366.3. That theorem identifies the handoff as a candidate-interface handoff rather than a theorem about all empty-context closed typed rows. Thus the public NF row cannot be projected from T alone.

Finally read the MetaCIC normalization audit map through the existing candidate-set facts closed_typed_term_sn_from_candidateSet, candidateSet_implies_sn, and canonicalCandidateSet_sound. Those facts supply the candidate membership and soundness side of the route, while R and E supply the public router and endpoint rows. The resulting boundary is therefore candidate-mediated and evidence-bearing, not an unconditional closed-term normalizer. □

Falsifiable prediction: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ consumer that exports strong normalization from T without reading K, R, E refutes the carrier surface.

Independence witness: candidate-set-normalization, metacicnormalizationcertificaterouter, metacicopenproblemledger: the candidate-set packet carries candidate evidence, the router selects the admissible normalization route, and the open-problem ledger records the frontier question; this packet binds those rows into a closed-projection boundary without adding an evidence-free normalization theorem.

Closure status for NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$.

Constructive story: A MetaCICCandidateClosedProjectionBoundary$^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate-set evidence, normalization-router selection, bounded-normal endpoint seal, no-infinite-reduction row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 4366.1, Definition 4366.2, Definition 4366.3, and Definition 4366.4 seed the finite closed-projection, candidate-evidence, router-selection, endpoint-seal, obstruction, public-boundary, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateClosedProjectionBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, unconditional strong normalization for all closed typed MetaCIC terms, unrestricted subject reduction, host reduction equality, quotient conversion, or a global normalizer.

Upgrade path: Obligation closure requires checked closed-term projection, candidate-evidence admission, router selection, bounded-normal endpoint sealing, no-infinite-reduction handoff, obstruction exactness, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateClosedProjectionBoundaryUp.

Chapter 4367

A Concrete Naming Certificate for $\mathsf{PointedMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based metric spaces, the $\mathsf{PointedMetricSpace}^{\uparrow}$ packet records a metric carrier together with a named basepoint route before completion, loop-space, or embedding consumers are allowed to inspect point-level data. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, reads names through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, passes regular rational distance observations through chapter 301 and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and offers a based source for the distance-preserving consumer route of chapter 439. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected representatives, or ambient completeness.

Definition 4367.1 (Pointed metric space carrier)**.** *A* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, b, D, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *b is the basepoint name row, stored as finite BEDC data attached to M rather than as a host-level point.*
3. *D is the basepoint distance ledger, recording the finite requests that compare b with inspected metric names.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which basepoint and metric names are inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed distance windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the basepoint-distance observations.*
7. *H records componentwise* hsame *transport for M, b, D, S, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a based metric request through the stream windows, regular rational readback, and real-distance seal.*
9. *P_0 is the* Pkg *provenance over* $\mathsf{PointedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint name, distance ledger, stream windows, regular readback, real seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate single carrier alignment`

Theorem 4367.2 (Pointed metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier* $P = (M, b, D, S, R, E, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint name, basepoint distance ledger, stream windows, regular rational readback, real-distance seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring every comparison of basepoint data to be read over the transported metric row* M*;*
3. *basepoint-distance exposure, requiring every basepoint comparison to pass through* D, S, R, E *before any completion-facing or embedding-facing consumer reads it;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, a selected representative, or an ambient completeness principle.

Proof. Project the carrier of Definition 4367.1. The projection exposes the metric row, basepoint name, distance ledger, stream windows, regular readback, real seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, b, D, S, R, E together, so a consumer cannot compare a basepoint name over one metric carrier with distance observations over another.

The distance ledger fixes the read order. A basepoint comparison is first a finite request in D, then a scheduled stream observation in S, then a regular rational readback in R, and only then a $\mathsf{Real}^{\uparrow}$ distance seal in E.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected representatives, or ambient completeness, the listed rows exhaust the local naming surface. □

Theorem 4367.3 (Pointed metric space basepoint nonescape)**.** *Let* $P = (M, b, D, S, R, E, H, C, P_0, N)$ *be an accepted* $\mathsf{PointedMetricSpace}^{\uparrow}$ *carrier. Every completion-facing, loop-space-facing, or isometric-embedding-facing consumer of the basepoint row* b *must read it through the displayed route*

$$M, b, D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P_0, N.$$

Thus the basepoint is exported only as finite named metric data over M*, not as a host point, quotient representative, selected completion point, or ambient metric-equality witness.*

Proof. Begin with carrier admission. The only row naming the basepoint is b, and it is stored beside the metric carrier M and distance ledger D.

Read the inspection route next. The stream windows S and regular rational readback R display every observed distance from the basepoint before the $\mathsf{Real}^{\uparrow}$ seal E is available to downstream consumers.

Transport and replay preserve that same route. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add host points, quotient representatives, selected completion points, or ambient metric equality.

Therefore every based consumer receives exactly the finite route displayed in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedMetricSpace}^{\uparrow}$ packet is generated from a $\mathsf{Met}^{\uparrow}$ carrier, a named basepoint row, a basepoint distance ledger, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ distance seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4367.1, Definition 4367.2, and Definition 4367.3 seed the based metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedMetricSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedMetricSpaceUp.PointedMetricSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host point equality, quotient metric equality, selected completion points, loop-space closure, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, basepoint-distance exposure, classifier transport, basepoint nonescape, and downstream completion and isometric-embedding consumer alignment for BEDC.Derived.PointedMetricSpaceUp.

Chapter 4368

A Concrete Naming Certificate for $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$

Following Bourbaki and Kelley on based complete metric spaces, the $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet records a based metric carrier together with a displayed completion service before completion, loop-space, or embedding consumers are allowed to read a point as complete. It sits over the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service boundary of chapter 128, the $\mathsf{PointedMetricSpace}^{\uparrow}$ basepoint route of chapter 4367, the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ rational readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The certificate names only finite BEDC rows; it does not import host points, quotient metric equality, selected completion representatives, countable choice, or ambient order completeness.

Definition 4368.1 (Pointed complete metric space carrier)**.** *A* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, B, Q, S, R, E, L, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *metric carrier row.*
2. *B is the basepoint row, stored as finite named metric data over M rather than as a host-level point.*
3. *Q is the Cauchy-window route, listing the finite tail requests through which regular Cauchy names based at B are inspected.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *readback row for the basepoint and Cauchy windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational distance and modulus observations.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed metric-distance and limit-distance observations.*
7. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion seal, exposing only the displayed limit-service handoff for the Cauchy-window route Q.*
8. *H records componentwise* hsame *transport for M, B, Q, S, R, E, L and the local certificate rows.*
9. *C records displayed* Cont *replay from a based complete-metric request through the Cauchy windows, stream observations, regular rational readback, real seal, and completion seal.*

10. *P_0 is the* Pkg *provenance over* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only after transporting the metric carrier, basepoint, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, replay, provenance, and local naming rows together.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.TasteGate.PointedCompleteMetricSpaceTasteGate single carrier alignment`

Theorem 4368.2 (Pointed complete metric space NameCert obligations)**.** *For every accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier* $P = (M, B, Q, S, R, E, L, H, C, P_0, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSp}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric row, basepoint row, Cauchy-window route, stream windows, regular rational readback, real seal, completion seal, transport, replay, provenance, and local naming rows;*
2. *common-carrier classification, requiring the basepoint row, Cauchy windows, and completion seal to be read over the transported metric row M;*
3. *Cauchy-window exposure, requiring every completion-facing read to pass through Q, S, R, E, L before any downstream consumer may inspect the limit handoff;*
4. *package provenance and local naming for exactly the displayed finite packet.*

No NameCert obligation is stored in a host point, quotient metric equality, selected completion representative, countable-choice selector, or ambient order-completeness principle.

Proof. Project the carrier of Definition 4368.1. The projection exposes the metric row, basepoint, Cauchy-window route, stream windows, regular readback, real seal, completion seal, transport, replay, provenance, and local name as the only available coordinates.

The classifier route is common-carrier. Componentwise transport in H moves M, B, Q, S, R, E, L together, so a consumer cannot compare a based Cauchy route over one metric carrier with a completion seal over another.

The completion-facing read order is fixed by the route rows. A completion request begins as a finite Cauchy-window request in Q, becomes a scheduled stream observation in S, is read back as regular rational data in R, is sealed as real-distance data in E, and only then reaches the $\mathsf{CompleteMetric}^{\uparrow}$ limit-service row L.

Finally replay C and attach P_0, N. Since the carrier has no coordinate for host points, quotient equality, selected completion representatives, countable-choice selectors, or ambient order completeness, the listed rows exhaust the local naming surface. □

Theorem 4368.3 (Pointed complete metric space Cauchy basepoint route)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ be an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carrier. Every based regular Cauchy request whose initial window is anchored at B is read through the displayed route*

$$M, B, Q \longmapsto S \longmapsto R \longmapsto E \longmapsto L \longmapsto H, C, P_0, N.$$

Thus the basepoint participates in completion only as named metric and Cauchy-window data over M, not as a selected host point or completed-limit representative.

Lean4 ✓ `BEDC.Derived.PointedCompleteMetricSpaceUp.PointedCompleteMetricSpaceCauchyBasepointRoute`

Proof. Begin with carrier admission. The only row naming the basepoint is B, and it is stored beside the metric row M and the Cauchy-window route Q.

Read the window route next. The stream row S schedules the inspected windows, the regular rational row R reads the tail and distance ledgers, and the $\mathsf{Real}^{\uparrow}$ row E seals the observed distances before the complete-metric row L can expose any limit-service handoff.

Transport and replay preserve that same path. The rows H, C, P_0, N move, replay, source, and name the displayed packet; they do not add a host basepoint, a selected limit representative, quotient equality, or a separate completion object.

Therefore every based Cauchy consumer receives exactly the finite route displayed in the statement. □

Theorem 4368.4 (Pointed complete metric basepoint completion handoff)**.** *Let $P = (M, B, Q, S, R, E, L, H, C, P_0$ and $P' = (M', B', Q', S', R', E', L', H', C', P'_0, N')$ be accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *carriers. If the two packets transport M, B, Q, S, R, E through the same componentwise H row, then every* $\mathsf{CompleteMetric}^{\uparrow}$ *service read of L is taken over that same named basepoint route before any loop-space or embedding consumer may inspect a completed point.*

Equivalently, a consumer may inspect

$$L \quad \text{or} \quad L'$$

only after the route

$$M, B, Q, S, R, E \longmapsto H, C, P_0, N$$

has been displayed; it cannot detach completion service from the transported basepoint row.

Proof. Project both carriers. In each packet the completion-service coordinate is L, and the only rows upstream of it are the metric row M, the basepoint row B, the Cauchy-window row Q, the stream row S, the regular rational row R, and the real-distance seal E.

Apply Definition 4368.3 to the first packet. The displayed route forces every based Cauchy request to pass through M, B, Q, S, R, E before the service row L is visible.

Use the assumed componentwise transport row H to move the same route to the second packet. The replay row C then reproduces the transported route for the completion-facing consumer, and the provenance and local name rows P_0, N record no independent completed point.

Thus a loop-space or embedding consumer can inspect a completed point only after the transported basepoint route has been displayed. There is no carrier coordinate from which the consumer could read L apart from B. □

Theorem 4368.5 (Pointed complete metric space completion nonescape)**.** *Every completion-facing, loop-space-facing, or embedding-facing consumer of an accepted* $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ *packet factors its completed-point read through*

$$M, B, Q, S, R, E, L, H, C, P_0, N.$$

Consequently the packet exports a based complete metric certificate only as finite BEDC NameCert data; it exports neither host points nor quotient completion equality.

Proof. Inspect the carrier coordinates. The only completion coordinate is L, and the definition makes L a completion seal for the already displayed Cauchy-window route Q over the metric row M and basepoint row B.

The readback rows precede the completion seal. Any consumer must first pass through S, R, E to display stream windows, regular rational ledgers, and real distance seals, so no completed point can be read before the finite analytic route is visible.

The remaining rows are administrative in the BEDC sense. Transport H, replay C, provenance P_0, and local name N preserve and source the packet but do not create host points, quotient completion equality, selected representatives, or ambient completeness.

Thus every downstream completed-point read factors through the displayed finite packet, which proves completion nonescape. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedCompleteMetricSpace}^{\uparrow}$ packet is generated from a MetricUp carrier, a named basepoint row, a Cauchy-window route, StreamName windows, RegSeqRat readback, a RealUp seal, a CompleteMetricUp completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4368.1, Definition 4368.2, Definition 4368.3, and Definition 4368.5 seed the based complete-metric route over $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PointedMetricSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient completion equality, selected host endpoints, loop-space closure, embedding closure, or ambient order completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, classifier transport, basepoint completion routing, completion nonescape, and downstream completion, loop-space, and embedding consumer alignment for BEDC.Derived.PointedCompleteMetricSpaceUp.

Chapter 4369

A Concrete Naming Certificate for $\mathsf{IntervalConnectedness}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{IntervalConnectedness}^{\uparrow}$ packet records the finite constructive route by which a located real interval is inspected as one connected interval surface before intermediate-value, zero-envelope, or completion-facing consumers use it. It sits after the endpoint seal of chapter 38, the located interval carrier of chapter 1709, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sandwich consumer surface of chapter 2163. The packet names interval connectedness as displayed finite BEDC data; it does not import host topology, open-cover compactness, trichotomy, quotient real equality, selected roots, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4369.1 (Interval connectedness carrier)**.** *An* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (E, L, D, S, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint-pair row, storing the lower and upper endpoint names through their displayed real seals.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *witness row, carrying the located interval budget, locator reads, and endpoint-radius checks attached to E.*
3. *D is the finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision-chain ledger, recording the dyadic cells, adjacent-cell overlaps, and tolerance requests inspected inside L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint, cell, and witness observations are scheduled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled endpoint and subdivision observations.*
6. *T records componentwise* hsame *transport for endpoint pairs, located interval witnesses, dyadic subdivision chains, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from endpoint pairs through located interval exposure, dyadic subdivision, stream-window inspection, regular readback, and the interval-connectedness handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{IntervalConnectedness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SqueezeTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, L, D, S, R, T, C, P, N$. It has no coordinate for host topological connectedness, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.IntervalConnectednessUp.IntervalConnectednessTasteGate single carrier alignment`

Theorem 4369.2 (Interval connectedness NameCert obligations)**.** *For every accepted* $\mathsf{IntervalConnectedness}^{\uparrow}$ *carrier*

$$I = (E, L, D, S, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the endpoint pair, located interval witness, dyadic subdivision chain, stream windows, regular readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier transport, requiring endpoint, locatedness, subdivision, window, readback, replay, provenance, and naming data to move componentwise through T;*
3. *dyadic chain exactness, requiring every inspected subinterval and adjacent-cell overlap to be recorded in D before a connectedness consumer reads the packet;*
4. *endpoint handoff nonescape, requiring every consumer of interval endpoints to pass through E, L, D, S, R before using any* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *ledger exhaustion, requiring the packet to expose no host topological connectedness theorem, open-cover compactness, trichotomy, quotient real equality, selected intermediate point, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

The sibling routes through $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{SqueezeTheorem}^{\uparrow}$ *are the only named consumers of this local certificate surface.*

Proof. Project the carrier of Definition 4369.1. The endpoint pair is visible only in E, and the located interval evidence is visible only in L.

The subdivision data are then forced through D. Each finite cell, overlap, and tolerance request is read as dyadic data before any stream-window or regular-readback consumer can inspect it.

Transport and replay preserve the same route. The rows T and C move endpoint, locatedness, subdivision, window, and readback data componentwise, while P and N source and name exactly that displayed packet.

Since the carrier has no coordinate for the excluded host principles, the local certificate cannot export them. □

Falsifiable prediction: Every accepted $\mathsf{IntervalConnectedness}^{\uparrow}$ consumer reads endpoint pairs, located interval witnesses, dyadic subdivision chains, StreamName windows, RegSeqRat readbacks, hsame transport, Cont replay, Pkg provenance, and local naming before any interval-connectedness handoff is visible.

Independence witness: real, locatedrealinterval, dyadicratcore, streamname, regseqrat, squeezetheorem: $\mathsf{IntervalConnectedness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint pairs, located interval witnesses, finite dyadic subdivision chains, scheduled stream windows, regular rational readbacks, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{IntervalConnectedness}^{\uparrow}}$.

Constructive story: An IntervalConnectedness$^{\uparrow}$ packet is generated from Real endpoint pairs, a LocatedRealInterval witness, a finite DyadicRatCore subdivision chain, StreamName windows, RegSeqRat readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, seed)

Scope closed: Definition 4369.1 and Definition 4369.2 seed the finite located-interval connectedness route over Real$^{\uparrow}$, LocatedRealInterval$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and SqueezeTheorem$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{IntervalConnectedness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host topology, open-cover compactness, trichotomy, quotient real equality, selected intermediate points, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, dyadic chain exactness, endpoint handoff nonescape, ledger exhaustion, and downstream intermediate-value consumer alignment for BEDC.Derived.IntervalConnectedness

Chapter 4370

A Concrete Naming Certificate for MetaCICResidualSubstitutionCompatibility$^{\uparrow}$

Following Tait–Martin-Lof and Barendregt on residuals, parallel reduction, and confluence discipline, the MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ packet records the finite boundary where a beta-parallel branch and a substitution replay must meet before the MetaCIC confluence route may use the conditional diamond bridge. It sits over the audit map of chapter 9378 and names the residual-substitution input as packet data rather than as a global Church–Rosser theorem. The packet does not claim full closed Church–Rosser, arbitrary closed-source confluence, or an unrestricted substitution theorem.

The residual-substitution packet is consumed by chapter 765 as a guarded input to candidate-mediated SN/confluence scheduling. The dependency is the residual join row together with the closedness guard data: it tells the scheduler which finite substitution replay may enter a candidate branch and which source windows are closed enough for that handoff, not that the full Church–Rosser theorem has already been completed.

Definition 4370.1 (MetaCIC residual-substitution compatibility carrier)**.** *A* MetaCICResidualSubstitutionCompatil *carrier is a finite* BHist *packet*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

with the following displayed rows:

1. *s is the source term row whose beta-parallel residual branch is inspected.*
2. *t is the substituted term row, storing the finite substitution request and the target occurrence window.*
3. *b is the parallel branch row, displaying the left and right beta-parallel steps before any join is named.*
4. *j is the residual join row, recording the candidate common reduct obtained after substitution replay and residual transport.*
5. *g is the closedness guard row, saying exactly which source and substitution windows are closed enough for the guarded handoff.*
6. *a is the atom-shape separation row, keeping atom, application, lambda, Pi, and beta-redex cases visibly disjoint.*
7. *h records componentwise* hsame *transport for source, substitution, branch, residual join, closedness guard, atom-shape separation, replay, provenance, and local naming rows.*

8. *c records displayed* Cont *replay from the source term through substitution, parallel branching, residual join construction, closedness guarding, and atom-shape separation.*
9. *p is the* Pkg *provenance over* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$*, MetaCIC beta-parallel reduction, residual joins, closedness guards, atom-shape separation,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *n is the local* NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $s, t, b, j, g, a, h, c, p, n$. It has no coordinate for a completed parallel diamond, full closed Church–Rosser, arbitrary closed-source confluence, or an unguarded substitution principle.

Lean4 ✓ BEDC.Derived.MetaCICResidualSubstitutionCompatibilityUp.MetaCICResidualSubstitutionCompatibilit
single carrier alignment

Theorem 4370.2 (MetaCIC residual-substitution compatibility NameCert obligations)**.** *For every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the local NameCert$_{\text{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and local naming rows;*
2. *branch-shape disjointness, requiring atom, application, lambda, Pi, and beta-redex branches to be separated by a before a residual join row is accepted;*
3. *substitution residual replay, requiring c to display how the substituted term row and beta-parallel branch row reach the same candidate join j;*
4. *closed-source guard, requiring g to expose the exact closedness assumptions consumed by the guarded compatibility packet;*
5. *non-claim of full closed Church–Rosser, requiring the packet to remain an input boundary for the conditional confluence bridge rather than a proof of arbitrary closed-source confluence.*

No obligation is stored outside $s, t, b, j, g, a, h, c, p, n$.

Proof. Project the carrier of Definition 4370.1. The source term, substituted term, branch, residual join, closedness guard, and atom-shape separation rows are the only computational coordinates.

The branch-shape row a fixes the case split before the join row is read. Thus a beta-redex residual branch cannot be silently identified with an atom, application, lambda, or Pi shape.

Replay then supplies the compatibility boundary. The row c records the finite route from substitution and beta-parallel branching to the candidate join j, while g states the closedness guard consumed by that route.

Finally h, p, n transport, source, and name exactly the displayed packet. Since full closed Church–Rosser and arbitrary closed-source confluence are not carrier coordinates, they are not exported by this NameCert surface. □

Theorem 4370.3 (MetaCIC residual-substitution candidate projection route)**.** *Every accepted* MetaCICResidualSubstitutionCompatibility$^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n)$$

exports a candidate-closed projection route to chapter 4366 only after the source term row s, substituted term row t, beta-parallel branch row b, residual join row j, closedness guard row g, and atom-shape separation row a have all been displayed. The downstream candidate boundary receives a guarded compatibility packet, not full Church–Rosser and not arbitrary closed-source confluence.

Proof. Project the carrier of Definition 4370.1. The rows s, t, b, j, g, a are the only computational rows available before the structural transport, replay, provenance, and local naming rows are read.

Use Definition 4370.2. Branch-shape disjointness exposes a, substitution residual replay exposes the route from t and b to j, and the closed-source guard exposes exactly the assumptions in g.

The candidate boundary of chapter 4366 is therefore reached through a guarded compatibility packet. The route supplies source, substitution, beta-parallel, residual-join, closedness, and atom-shape data, while h, c, p, n only transport, replay, source, and name those rows.

Since completed Church–Rosser and arbitrary closed-source confluence are absent from the carrier, the candidate-closed projection route cannot export them. It hands off only the displayed residual-substitution compatibility boundary. □

Theorem 4370.4 (MetaCIC residual-substitution obligation closure surface)**.** *For every accepted* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *carrier*

$$M = (s, t, b, j, g, a, h, c, p, n),$$

the obligation surface is exactly the displayed source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, transport, replay, provenance, and naming rows. The candidate-closed projection boundary receives only this guarded compatibility packet.

Proof. Apply Definition 4370.2. It identifies carrier admission, branch-shape disjointness, substitution residual replay, closed-source guarding, transport, replay, provenance, and naming as rows of $s, t, b, j, g, a, h, c, p, n$.

Apply Definition 4370.3. The candidate boundary receives the source, substitution, beta-parallel branch, residual-join, closedness, and atom-shape data only through that guarded projection route.

Therefore the obligation surface packages the candidate handoff without exporting full closed Church–Rosser, arbitrary closed-source confluence, or unguarded substitution compatibility. Those claims are not coordinates of the carrier and are not supplied by the candidate projection. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ read displays source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, hsame transport, Cont replay, Pkg provenance, and local naming rows before the conditional MetaCIC confluence bridge can consume it.

Independence witness: metacic-confluence-audit: $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ is not a completed confluence theorem because it carries only the finite residual-substitution compatibility boundary consumed by the conditional parallel-diamond route.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ packet is generated from source term, substituted term, parallel branch, residual join, closedness guard, atom-shape separation, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4370.1, Definition 4370.2, and Definition 4370.3 seed the finite residual-substitution compatibility boundary consumed by the conditional MetaCIC confluence bridge and the candidate-closed projection route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full closed Church–Rosser, arbitrary closed-source confluence, unguarded substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, branch-shape disjointness, substitution residual replay, closed-source guard, and conditional confluence handoff alignment for BEDC.MetaCIC.BetaParallelSubstitutionCompatibility.

Chapter 4371

A Concrete Naming Certificate for $\mathsf{ComputableCompactMetric}^{\uparrow}$

Following Weihrauch-style computable analysis and the Bishop–Bridges compact metric arguments, the $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet records the effective compact-metric surface as finite BEDC NameCert data. It sits after the compact metric carrier of chapter 129, the effective Polish source of chapter 1706, the total-bounded finite-net route of chapter 127, the complete metric source of chapter 128, and the L10 readback route through chapter 126, chapter 301, and chapter 38. The packet names the finite-cover realizer surface consumed by effective compact analysis; it does not assert open-cover compactness, selected subcovers, quotient completion equality, or an ambient real-completeness principle.

Definition 4371.1 (Computable compact metric carrier)**.** *A* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the compact metric packet whose finite-net behavior is being made effective.*
2. *E is the* $\mathsf{EffectivePolishSpace}^{\uparrow}$ *source row exposing the effective dense-name and semidecidable basic-ball interface.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *row naming the finite-net source for each displayed dyadic tolerance request.*
4. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row recording the convergence service available to Cauchy and compact consumers.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for inspected dense names, finite net centers, tolerance requests, and cover indices.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the regular rational readback of distances and tolerance comparisons.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows and regular rational comparisons have been displayed.*

8. *H records componentwise* hsame *transport for compact metric, effective Polish, total-bounded, complete-metric, stream-window, regular-readback, real-seal, finite-cover realizer, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from an effective compactness request through K, E, T, C, S, R, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{ComputableCompactMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, E, T, C, S, R, L, H, Q, P, N$. It has no coordinate for an open-cover subcover, a choice-selected finite set, quotient completion equality, excluded-middle compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ComputableCompactMetricUp.ComputableCompactMetricTasteGate single carrier alignment`

Theorem 4371.2 (Computable compact metric NameCert obligations)**.** *For every accepted* ComputableCompactMe *carrier*

$$K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$ *surface consists exactly of the displayed rows in Definition 4371.1. Carrier admission must expose the compact metric source, effective Polish source, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational readback, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row. Classifier transport is componentwise through those same rows, and ledger exhaustion says that every finite-cover or compact-analysis consumer can inspect only $K, E, T, C, S, R, L, H, Q, P, N$.*

Proof. Project the carrier of Definition 4371.1. The rows K, E, T, C are the only structural compactness sources, so carrier admission exposes compact metric data, effective Polish naming data, total-bounded finite nets, and complete-metric convergence before any realizer can be read.

The readback rows S, R, L then bind numerical access to finite windows, regular rational comparisons, and the terminal real seal. This is the same L10 readback discipline used by the StreamName, RegSeqRat, and Real siblings named in the chapter introduction.

Transport and replay are visible rows rather than hidden meta-operations. The row H records componentwise hsame transport, Q records displayed Cont replay, and P, N source and name the same finite packet.

For ledger exhaustion, read any compact-analysis consumer from the accepted packet. The carrier exposes only $K, E, T, C, S, R, L, H, Q, P, N$, so no open-cover subcover, selected finite set, quotient completion equality, excluded-middle compactness, or ambient real-completeness principle can be obtained from this NameCert surface. □

Theorem 4371.3 (Computable compact metric finite-cover realizer)**.** *Every effective finite-cover read exported by an accepted* $\mathsf{ComputableCompactMetric}^{\uparrow}$ *packet factors through the route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EffectivePolishSpace}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsa

Thus a consumer receives a compact metric source, effective dense-name interface, total-bounded finite-net source, complete-metric convergence service, finite stream windows, regular rational tolerance comparisons, terminal real seal, componentwise transport, replay, provenance, and local

naming rows before it uses the finite-cover realizer. The packet exports no selected subcover, open-cover compactness theorem, quotient completion representative, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with an accepted carrier $K_{\mathrm{ccm}} = (K, E, T, C, S, R, L, H, Q, P, N)$. The compact metric row K and effective Polish row E name the metric and effective dense-name surfaces, while T supplies the finite-net source used for the requested tolerance.

The convergence row C keeps the compact route compatible with complete-metric consumers. A finite-cover realizer cannot leave these displayed sources, because the later rows are read only after the compact, effective, total-bounded, and complete rows are present.

The numerical readback is finite. The stream row S exposes inspected names, center windows, cover indices, and tolerance requests; the regular rational row R compares the displayed distances; and the terminal seal L is reached only after those windows and comparisons have been read.

Finally H, Q, P, N preserve the route under transport, replay, provenance, and local naming. Since selected subcovers, open-cover compactness theorems, quotient representatives, choice selectors, and ambient completeness witnesses are not carrier coordinates, the exported realizer is exactly the displayed finite-cover surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{ComputableCompactMetric}^{\uparrow}$ packet is generated from CompactMetric, EffectivePolishSpace, TotallyBounded, CompleteMetric, StreamName, RegSeqRat, RealUp, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4371.1, Definition 4371.2, and Definition 4371.3 seed the effective finite-cover realizer route for computable compact metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ComputableCompactMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, open-cover compactness, quotient completion equality, countable-choice selectors, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective dense-name exposure, total-bounded finite-net exposure, complete-metric convergence exposure, finite-window readback, regular rational comparison, real sealing, finite-cover realizer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ComputableCompactMetricUp.

Chapter 4372

A Concrete Naming Certificate for $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet records the separated-completion reflection route for regular Cauchy rational names as finite BEDC NameCert data. It sits over the regular rational readback surface of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the uniqueness boundary of chapter 3390. The packet lets a Real-facing consumer treat two representative regular Cauchy names as the same separated reflection point only after the displayed finite windows, dyadic budgets, and uniqueness ledger have been exposed.

Definition 4372.1 (Regular-Cauchy Hausdorff-reflection carrier)**.** *A* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, Y, S, T, D, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the first* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row.*
2. *Y is the second* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational Cauchy name row presented to the same separated-reflection request.*
3. *S and T are the paired* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
5. *U is the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row witnessing that the displayed tail comparisons force separated agreement.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after X, Y, S, T, D, U have been displayed.*
7. *H records componentwise* hsame *transport for the two regular names, stream windows, dyadic ledger, uniqueness row, Real seal, and local certificate rows.*
8. *C records the displayed* Cont *replay from a reflection request to the paired windows, dyadic tolerance ledger, uniqueness row, and terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, S, T, D, U, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier admission surface`

Theorem 4372.2 (Regular-Cauchy Hausdorff-reflection NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *carrier* $R = (X, Y, S, T, D, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *separated-agreement row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation may replace these rows by quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionTasteGate single carrier alignment`

Proof. Project the carrier of Definition 4372.1. The only displayed coordinates are $X, Y, S, T, D, U, E, H, C, P, N$.

The two regular rational names are compared through the paired stream windows S, T and the dyadic tolerance ledger D. The separated-agreement row U is therefore a displayed uniqueness boundary, not a quotient of streams or a host equality proof.

The terminal Real seal E is downstream of that boundary. Transport, replay, provenance, and local naming are supplied by H, C, P, N, and no excluded quotient, selector, host-equality, choice, trichotomy, or ambient-completeness coordinate can be projected from the carrier. □

Theorem 4372.3 (Regular-Cauchy Hausdorff-reflection representative independence)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet is independent of the chosen representative regular Cauchy name in the following BEDC sense: if the two presented* $\mathsf{RegSeqRat}^{\uparrow}$ *rows are compared through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the terminal seal can distinguish them only through the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row. The packet exports no quotient stream equality, selected representative, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionRepresentativel`

Proof. Begin with an accepted carrier $R = (X, Y, S, T, D, U, E, H, C, P, N)$. The rows X and Y are not presented as completed real objects; they are presented as regular rational names observed through finite stream windows.

The dyadic ledger D binds the comparison budget for those windows, and the uniqueness row U is the only row that can turn the displayed tail comparison into separated agreement. Thus any Real-facing read of E must pass through the same finite comparison route.

The rows H, C, P, N transport, replay, source, and name this route without adding analytic content. Since the carrier contains no further equality coordinate, changing representatives cannot be detected except through the displayed uniqueness row. □

Theorem 4372.4 (Regular-Cauchy Hausdorff-reflection Real seal non-escape)**.** *No terminal* $\mathsf{Real}^{\uparrow}$ *seal consumer accepted by a* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchyUniqueness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the separated reflection supplies representative independence for the Real seal without quotient streams, host real equality, selected representatives, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier real seal nonescape`

Proof. Read the carrier coordinates in order. The terminal row E appears only after the two regular rational names, paired stream windows, dyadic tolerance ledger, and uniqueness row have been displayed.

Apply Definition 4372.2. The local NameCert surface exposes exactly those rows, with H, C, P, N preserving the same route for downstream consumers.

The advertised route is therefore the only public path to the Real seal. Since no excluded equality or choice coordinate occurs in the carrier, the seal cannot escape through quotient streams, host equality, representatives selected outside the packet, countable choice, trichotomy, or ambient completeness. □

Theorem 4372.5 (Regular-Cauchy Hausdorff-reflection finite-window uniqueness route)**.** *Suppose two accepted* $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ *packets display the same paired* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the same* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *row, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal. Then every separated-reflection consumer factors through the same finite-window route on the underlying* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. The factorization does not use quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHausdorffReflectionUp.RegularCauchyHausdorffReflectionCarrier finite window uniqueness route`

Proof. Apply Definition 4372.3. Once the paired StreamName windows, dyadic tolerance ledger, and uniqueness row agree, the terminal Real seal cannot distinguish the two regular Cauchy representatives by any route other than the displayed uniqueness row.

Apply Definition 4372.4. The Real seal is downstream of the finite windows, dyadic ledger, and uniqueness row, so a separated-reflection consumer has no bypass around those displayed rows.

The remaining rows are structural. The hsame rows transport only the displayed components, the Cont rows replay the same finite-window request, and the Pkg and NameCert rows name that route inside BHist. Therefore the consumer factors through the same finite-window route. □

Falsifiable prediction: An accepted $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ consumer that reaches the terminal RealUp seal without reading two RegSeqRat source rows, paired StreamName windows, a DyadicRatCore tolerance ledger, and a RegularCauchyUniqueness row refutes the carrier surface.

Independence witness: regseqrat, streamname, dyadicratcore, real, regularcauchyuniqueness: $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ is not bijective to any listed sibling because it binds two regular rational representatives to one separated uniqueness row and one downstream Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName finite windows, a DyadicRatCore tolerance ledger, a RegularCauchyUniqueness separated-agreement row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4372.1, Definition 4372.2, Definition 4372.3, Definition 4372.4, and Definition 4372.5 seed the separated reflection route for regular Cauchy representatives and Real seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHausdorffReflection}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface, representative-independence handoff, and Real seal non-escape theorem.

Chapter 4373

A Concrete Naming Certificate for $\mathsf{WeakLawLargeNumbers}^{\uparrow}$

Following Kolmogorov and Bishop–Bridges on finite averages and probability estimates, the WeakLawLargeNumbers packet names the displayed finite-average route behind the weak law of large numbers. It sits after the normalized probability surface of chapter 184, the measurable random-variable rows of chapter 185, the finite tail-bound carrier of chapter 6592, and the terminal real-name seal of chapter 38. The packet is a finite certificate for Chebyshev-controlled averages; it is not an almost-sure convergence theorem, a central limit theorem, a countable-choice subsequence selector, a host probability-limit principle, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4373.1 (Weak law finite-average carrier)**.** *A* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, R, A, V, T, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the total-event normalization and event classifier. The row R is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over that same probability source. The row A records the displayed finite average of the entries in R, while V records the variance or uniform second-moment bound used by the estimate. The row T reads the Chebyshev tail row through* $\mathsf{ChebyshevInequality}^{\uparrow}$ *for the event where the finite average differs from its displayed center by more than the requested tolerance. The row E is the* $\mathsf{Real}^{\uparrow}$ *epsilon seal naming the positive tolerance and the final probability bound. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite averaging route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{ChebyshevInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these ten rows.*

Lean4 ✓ `BEDC.Derived.WeakLawLargeNumbersUp.WeakLawLargeNumbersTasteGate single carrier alignment`

Theorem 4373.2 (Weak law NameCert obligations)**.** *For every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *carrier $(P, R, A, V, T, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 4373.1. Carrier admission displays the probability source, the random-variable family, the finite-average row, the variance-bound row, the Chebyshev tail row, the epsilon seal, the transport and replay rows, the provenance row, and the local naming row. The packet imports no host law of large numbers, selected subsequence, external probability-limit operation, countable-choice principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The probability source P supplies the measurable-event classifier and total-mass normalization, and the random-variable row R exposes only the finitely displayed variables over that source.

The average row A and variance row V are then internal finite data. They record the algebraic averaging expression and the bound used by the tail estimate before any limiting statement is visible.

Finally read T, E, H, C, Q, N. The Chebyshev tail row and epsilon seal name the probability estimate at the requested tolerance, while transport, replay, provenance, and local naming keep the same finite route under classifier replacement. Since no carrier coordinate stores an infinite limit or subsequence selector, the listed rows exhaust the local certificate surface. □

Theorem 4373.3 (Weak law Chebyshev handoff)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *Chebyshev read factors through the route*

$$P \longmapsto R \longmapsto A \longmapsto V \longmapsto T \longmapsto E \longmapsto H, C, Q, N$$

before any probability-bound consumer receives the finite-average estimate. Thus the Chebyshev step is handed to later probability and $\mathsf{Real}^{\uparrow}$ *formalization as a displayed finite source, finite random-variable row, average row, variance-bound row, tail-event row, epsilon seal, transport, replay, provenance, and local naming data.*

Proof. Start with a probability-bound request. The only probability source exposed by the packet is P, and the only random-variable data exposed over it are the finite rows in R.

The request must next read the average and variance-bound rows. The average row A fixes the finite expression being estimated, and V fixes the bound that the Chebyshev tail row may consume.

Only after these rows are displayed can T and E be read as a bound on the exceptional event. The structural rows H, C, Q, N then preserve that same route for downstream consumers, so the handoff is the finite Chebyshev estimate itself rather than a host-side convergence theorem. □

Theorem 4373.4 (Weak law finite-average nonescape)**.** *Every accepted* $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ *probability-limit-facing read remains inside the finite-average route of Definition 4373.1. In particular, a consumer can read a tail estimate only after displaying the source probability row, finite random-variable rows, average row, variance-bound row, Chebyshev tail row,* $\mathsf{Real}^{\uparrow}$ *epsilon seal, componentwise transport, replay, provenance, and local* $\mathsf{NameCert}$ *row. No accepted read may replace this route by almost-sure convergence, central limit normalization, countable-choice subsequence selection, a host probability-limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a consumer request the finite-average estimate from an accepted carrier. Projection of Definition 4373.1 shows that the request first meets P, R, A, V, T, E and has no independent coordinate for a limit object.

The displayed Chebyshev row T is downstream from the variance-bound row V and upstream from the epsilon seal E. Hence any probability bound read is already tied to the visible finite average and its tolerance.

The remaining rows H, C, Q, N do not add new analytic content. They transport equal displayed histories, replay the finite route, record the package provenance, and attach the local name. Therefore every accepted probability-limit-facing read stays inside the finite-average certificate surface and cannot escape to the excluded convergence or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WeakLawLargeNumbers}^{\uparrow}$ packet is generated from a probability source row, finite random-variable rows, a finite-average row, a variance or second-moment bound row, a Chebyshev tail row, a RealUp epsilon seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4373.1, Definition 4373.2, Definition 4373.3, and Definition 4373.4 seed the finite-average probability route through ProbSpace, RandomVar, RealUp, Chebyshev tail, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakLawLargeNumbers}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-sure convergence, the central limit theorem, countable-choice subsequence selection, host probability limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-average routing, variance-bound stability, Chebyshev tail handoff, epsilon-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.WeakLawLargeNumbe

Chapter 4374

A Concrete Naming Certificate for $\mathsf{LocatedSubset}^{\uparrow}$

Following Bishop and Bridges on located subsets of metric and real spaces, the $\mathsf{LocatedSubset}^{\uparrow}$ packet records a displayed subset together with finite distance-witness rows. It sits after the real-valued metric route of chapter 1638, the Hausdorff comparison boundary of chapter 2086, the real-separability window surface of chapter 1809, the dyadic tolerance core of chapter 291, and the regular rational readback surface of chapter 301. The packet gives Real and Completion consumers a finite located-subset interface without host set extensionality, selected nearest points, quotient subset equality, or ambient completeness.

The located-subset distance route consumes chapter 1809 as its finite search-window source and consumes chapter 2086 only after the real-distance seal has been displayed.

Definition 4374.1 (LocatedSubset carrier)**.** *A* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, S, W, D, R, E, K, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *source row, carrying the ambient metric distance through finite real-name observations.*
2. *S is the subset-admission row, naming the displayed members read by the carrier and exporting no host powerset equality.*
3. *W is the* $\mathsf{RealSeparability}^{\uparrow}$ *search-window row, recording the countable dense or finite-window probes available to a distance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to each positive distance query.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the rational comparison data produced from W and D.*
6. *E is the real distance seal row, read only after the metric source, subset admission, search windows, dyadic tolerances, and regular rational readback have been displayed.*
7. *K is the optional* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison boundary for compact or hyperspace consumers; it records only the finite directed comparisons visible through the preceding rows.*
8. *H records componentwise* hsame *transport for the metric source, subset admission, search windows, dyadic ledger, regular readback, real seal, Hausdorff boundary, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an ambient point and tolerance request through subset admission, separability windows, dyadic comparison, regular rational readback, real sealing, and any Hausdorff-facing comparison.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedSubset}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{HausdorffMetric}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, S, W, D, R, E, K, H, C, P, N$. It has no coordinate for a host subset object, extensional equality of subsets, a selected nearest point, quotient real equality, countable choice, or an ambient completion theorem.

Lean4 ✓ `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate single carrier alignment`

Theorem 4374.2 (LocatedSubset NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *carrier*

$$L = (M, S, W, D, R, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedSubset}^{\uparrow}}$ *surface consists exactly of carrier admission, ambient real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, optional Hausdorff comparison routing, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 4374.1. The displayed packet lists exactly the rows used by an ambient point, subset-membership, tolerance, rational-readback, real-seal, or Hausdorff-facing request.

The metric and subset rows M, S are read first. A distance query can inspect the ambient space only through $\mathsf{RealMetric}^{\uparrow}$, and it can inspect membership only through the displayed admission row S.

The search, tolerance, and readback rows W, D, R then supply finite witnesses for the locatedness request. The real seal E is reached only after those rows have been displayed, while K is available only as a downstream comparison boundary.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no host subset, selected nearest point, quotient equality, countable-choice schedule, or ambient completion coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4374.3 (LocatedSubset dyadic distance witness)**.** *Every distance read exported by an accepted* $\mathsf{LocatedSubset}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RealMetric}^{\uparrow} \longmapsto \mathsf{RealSeparability}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto E.$$

The read may be handed to a $\mathsf{HausdorffMetric}^{\uparrow}$ *consumer only after the same route has produced the displayed real-distance seal, transported by H and replayed by C.*

Proof. Begin with a point-to-subset distance request. The ambient metric row M supplies the only metric observations, and the subset-admission row S supplies the only visible membership data.

The finite search window W names the inspected probes. The dyadic ledger D fixes the positive tolerance requests, and the regular rational row R records the resulting readback windows.

The real-distance seal E is therefore downstream of M, S, W, D, R. Any Hausdorff-facing row K can compare only the finite distance data already sealed by this route.

Transport and replay do not add another distance object. The componentwise hsame row H, the displayed Cont route C, the Pkg provenance P, and the local NameCert row N preserve the

same witness surface, so no nearest-point selector or host infimum can bypass the dyadic distance route. □

Falsifiable prediction: Every accepted LocatedSubsetUp distance read exposes RealMetricUp, subset admission, RealSeparabilityUp windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, and a real-distance seal before any HausdorffMetricUp or completion-facing consumer reads the packet.

Independence witness: realmetric, hausdorffmetric, realseparability, dyadicratcore, regseqrat: none of these siblings alone carries the full located-subset packet with subset admission, finite search windows, dyadic tolerance rows, regular rational readback, real-distance sealing, optional Hausdorff comparison, transport, replay, package provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}$.

Constructive story: A $\mathsf{LocatedSubset}^{\uparrow}$ packet is generated from a RealMetricUp source, displayed subset admission, RealSeparabilityUp search windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a real-distance seal, optional HausdorffMetricUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{seed})$

Scope closed: Definition 4374.1, Definition 4374.2, and Definition 4374.3 seed the finite located-subset interface for real-metric and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedSubset}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedSubsetUp.LocatedSubsetTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host subset extensionality, selected nearest points, quotient subset equality, countable choice, open-cover compactness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, real-metric visibility, subset-admission stability, separability-window exposure, dyadic-tolerance compatibility, regular-rational readback, real-distance sealing, Hausdorff comparison routing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LocatedSubsetUp.

Chapter 4375

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 4375.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 4375.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 4375.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 4375.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 4375.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 4375.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 4376

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 4376.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 4376.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 4376.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 4377

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4377.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 4377.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4377.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 4378

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 4378.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 4378.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 4378.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 4378.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 4378.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 4378.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 4379

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 4379.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4379.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 4379.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 4379.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 4380

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 4380.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- M *is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- F *is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- B *is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- E *is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- I *is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- H *records componentwise* hsame *transport for* M, F, B, E, I *and the structural rows.*
- C *records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- P *is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- N *is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 4380.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 4380.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 4380.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 4381

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4381.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4381.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4381.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 4381.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 4381.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 4381.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 4381.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 4381.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 4381.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 4381.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 4381.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 4381.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 4381.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 4381.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 4381.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 4382

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 4382.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4382.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 4382.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 4382.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 4382.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 4382.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 4382.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W is supplied by $\mathsf{StreamName}^{\uparrow}$*, R by* $\mathsf{RegSeqRat}^{\uparrow}$*, D by* $\mathsf{DyadicRatCore}^{\uparrow}$*, E by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 4382.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 4382.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 4382.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 4383

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 4383.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 4383.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4383.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 4383.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 4384

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 4384.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 4384.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 4384.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let $L = (S, Q, I, U, F, K, T, H$ be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

$\mathsf{RieszThorinInterpolation}^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 4385

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 4385.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4385.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4385.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 4385.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 4386

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4386.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 4386.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 4386.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 4386.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 4387

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4387.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4387.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4387.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4388

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4388.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 4388.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4388.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 4388.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 4389

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 4389.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 4389.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 4389.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 4390

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 4390.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 4390.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4390.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 4391

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4391.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4391.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4391.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4391.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4391.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4392

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4392.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4392.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4392.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4392.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4393

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4393.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4393.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4393.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4394

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4394.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4394.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4394.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4394.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4395

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4395.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4395.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4395.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4396

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4396.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4396.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4396.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4397

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4397.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 4397.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4398

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4398.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4398.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4398.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4399

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4399.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4399.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4399.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4399.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4400

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4400.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4400.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4400.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4401

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4401.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4401.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4401.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4402

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4402.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 4402.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4402.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4403

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4403.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4403.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4403.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4403.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4404

A Concrete Naming Certificate for $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$

The $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4404.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* $\mathsf{UnitDisk}^{\uparrow}$ *coordinates and the associated* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4404.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4404.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4404.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4405

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4405.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4405.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4405.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4406

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4406.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4406.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4406.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4406.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4406.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4407

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4407.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4407.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4407.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4408

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4408.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4408.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4408.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4408.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4408.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4408.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4409

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4409.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4409.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4409.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4410

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4410.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4410.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4410.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4411

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4411.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4411.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4411.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4411.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4412

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4412.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4412.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4412.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4413

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4413.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4413.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4413.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4413.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4414

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4414.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4414.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4414.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4414.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4415

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4415.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4415.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4415.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4416

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4416.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4416.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4416.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4417

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4417.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4417.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4417.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4417.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4417.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4418

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4418.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4418.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4419

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4419.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4419.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4419.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4420

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4420.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4420.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4420.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4421

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4421.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4421.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4421.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4422

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4422.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4422.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4422.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4423

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4423.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4423.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4424

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4424.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4424.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4424.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4425

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4425.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4425.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4425.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4426

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4426.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4426.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4426.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4427

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4427.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4427.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4427.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 4428

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4428.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 4428.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4428.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4429

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4429.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4429.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4429.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, seed)

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4430

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4430.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4430.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4430.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4430.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4431

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4431.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4431.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4431.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local NameCert rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4432

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4432.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4432.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4432.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4433

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4433.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4433.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4433.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4434

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4434.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate single carrier alignment`

Theorem 4434.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4434.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4435

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4435.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4435.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4435.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4436

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4436.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4436.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4436.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4437

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 4437.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 4437.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 4437.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4438

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4438.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4438.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4439

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4439.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 4439.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4439.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4440

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 4440.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 4440.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4440.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 4440.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4441

A Concrete Naming Certificate for ContinuousRiemannIntegrability$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the ContinuousRiemannIntegrability$^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient Real$^{\uparrow}$ completeness.

Definition 4441.1 (Continuous Riemann integrability carrier)**.** *A* ContinuousRiemannIntegrability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* Continuous$^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* TaggedRealPartition$^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* DarbouxSum$^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* RiemannIntegrable$^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* WeaklyLocatedIntervalSelection$^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* StreamName$^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4441.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4441.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4441.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4442

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4442.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4442.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4442.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4443

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4443.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4443.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4443.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4444

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4444.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4444.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4444.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4445

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4445.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4445.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4446

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4446.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4446.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4447

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

4447.1 Carrier definition

Definition 4447.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

4447.2 Certificate surface

Theorem 4447.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 4447.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 4447.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 4448

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 4448.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 4448.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 4448.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4449

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4449.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4449.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4449.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4449.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4449.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4450

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4450.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4450.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4450.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4450.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4451

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4451.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4451.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4451.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4452

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4452.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4452.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4452.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4452.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4453

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4453.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4453.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4453.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4454

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4454.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4454.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4454.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4455

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4455.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 4455.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4456

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4456.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4456.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4456.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4457

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4457.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4457.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4457.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4457.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4458

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4458.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4458.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4458.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4459

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4459.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4459.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4459.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4460

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4460.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 4460.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4460.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4461

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4461.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4461.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4461.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4461.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4462

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4462.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4462.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4462.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4462.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4463

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4463.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4463.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4463.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4464

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4464.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4464.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4464.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4464.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4464.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4465

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4465.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4465.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4465.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4466

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4466.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4466.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4466.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4466.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4466.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4466.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4467

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4467.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4467.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4467.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4468

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4468.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4468.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4468.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4469

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4469.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4469.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4469.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4469.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4470

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4470.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4470.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4470.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4471

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4471.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopFixedPointInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4471.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal $\mathsf{Real}^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4471.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* $\mathsf{LocatedInterval}^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz row M, and only then the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4471.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4472

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4472.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4472.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4472.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4472.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4473

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4473.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4473.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4473.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4474

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4474.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4474.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4474.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4475

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4475.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4475.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4475.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N.$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4475.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4475.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4476

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4476.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4476.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4477

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4477.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4477.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4477.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4478

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4478.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4478.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4478.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and FieldFaithful DNAWindowSi TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4479

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4479.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4479.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4479.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4480

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4480.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4480.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4480.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4481

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4481.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4481.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4482

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4482.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4482.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4482.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4483

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4483.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4483.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4483.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4484

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4484.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4484.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4484.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4485

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4485.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4485.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4485.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 4486

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 4486.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 4486.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 4486.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 4486.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 4486.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 4486.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 4486.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 4486.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 4487

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4487.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 4487.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4487.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4488

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4488.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4488.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4488.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4489

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4489.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4489.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4489.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4489.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4490

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 4490.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 4490.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4490.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 4490.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 4490.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 4490.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 4491

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4491.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4491.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4491.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local NameCert rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4492

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4492.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4492.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4492.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4493

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4493.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4493.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4493.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4494

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4494.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 4494.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4494.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\text{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4495

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4495.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4495.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4495.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4496

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4496.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4496.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4496.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4497

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 4497.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 4497.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 4497.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 4497.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4498

A Concrete Naming Certificate for ContinuousRiemannIntegrability$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the ContinuousRiemannIntegrability$^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient Real$^{\uparrow}$ completeness.

Definition 4498.1 (Continuous Riemann integrability carrier)**.** *A* ContinuousRiemannIntegrability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* Continuous$^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* TaggedRealPartition$^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* DarbouxSum$^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* RiemannIntegrable$^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* WeaklyLocatedIntervalSelection$^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* StreamName$^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4498.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4498.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4498.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4499

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4499.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4499.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4499.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4500

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 4500.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 4500.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 4500.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 4500.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 4500.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 4501

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4501.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4501.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4501.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4502

A Concrete Naming Certificate for $\mathsf{OneSidedCauchyLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ packet records the one-sided limit surface at which side-tagged approach data is also certified by cofinal Cauchy-window evidence. It refines the side classifier and approach-window rows of chapter 5841, reads Cauchy-space compatibility through $\mathsf{CauchySpace}^{\uparrow}$ from chapter 1847, consumes $\mathsf{StreamName}^{\uparrow}$ windows from chapter 126, records $\mathsf{RegSeqRat}^{\uparrow}$ readback by chapter 301, measures tolerances through $\mathsf{DyadicRatCore}^{\uparrow}$ from chapter 291, and reaches the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is an L10 one-sided Cauchy seal route: it names finite cofinal-window data before any Real-facing consumer may treat a side approach as a completed limit.

Definition 4502.1 (One-sided Cauchy limit carrier)**.** *A* $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, J, S, R, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row inherited from the one-sided source.*
2. *L is the left-hand or right-hand side classifier read before any Cauchy row is inspected.*
3. *W is the finite row of side-compatible approach windows.*
4. *J is the cofinal-window ledger, recording for each displayed tolerance request a later side-compatible window in W.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened only along the cofinal windows in J.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational observations exposed by S.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare side windows and regular-rational reads.*
8. *Q is the* $\mathsf{CauchySpace}^{\uparrow}$ *compatibility row asserting that the displayed side-window schedule is Cauchy at the named tolerances.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after A, L, W, J, S, R, D, Q have been displayed.*
10. *H records componentwise* hsame *transport for all carrier rows.*
11. *C records displayed* Cont *replay from the side-tagged approach data through the cofinal Cauchy-window route.*

12. *P is the* Pkg *provenance over* $\mathsf{OneSidedCauchyLimit}^{\uparrow}$, $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{CauchySpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedCauchyLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, W, J, S, R, D, Q, E, H, C, P, N$. It has no coordinate for a two-sided limit theorem, quotient real equality, selected tail, countable-choice schedule, host order trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4502.2 (One-sided Cauchy limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ *carrier $K = (A, L, W, J, S, R, D, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedCauchyLimit}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; side classifier stability before Cauchy inspection; cofinal-window coverage for each dyadic tolerance request;* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback along those windows;* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness;* $\mathsf{CauchySpace}^{\uparrow}$ *compatibility for the same schedule; terminal* $\mathsf{Real}^{\uparrow}$ *sealing; componentwise transport; replay; provenance; and local naming.*

Proof. Carrier admission is projection from Definition 4502.1. The accepted packet displays the approach anchor, side classifier, finite windows, cofinal-window ledger, stream schedule, regular readback, dyadic tolerance ledger, Cauchy-space row, terminal Real seal, and structural rows.

The side classifier is inspected before the Cauchy row. Thus every cofinal-window entry in J is read as a left-hand or right-hand approach window from W, and transport through H preserves that tag while the stream schedule and readback rows are replayed.

The Cauchy obligations are exactly the finite rows J, S, R, D, Q. The cofinal ledger J chooses the displayed later windows, S opens only their observations, R records the regular rational data, D records the demanded tolerances, and Q confirms Cauchy-space compatibility for that same visible schedule.

The Real seal E is downstream of the whole route. Since no row stores a two-sided limit theorem, quotient equality, selected tail, countable-choice schedule, trichotomy, or ambient Real completeness, those objects cannot become local obligations of the packet. □

Theorem 4502.3 (One-sided Cauchy L10 seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{CauchySpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ *packet reads the one-sided Cauchy limit only through the finite route*

$$\mathsf{OneSidedLimit}^{\uparrow},\ \mathsf{CauchySpace}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In carrier coordinates this route is $A, L, W, J, S, R, D, Q, E, H, C, P, N$. The handoff exports no two-sided limiting principle, quotient stream equality, selected Cauchy tail, countable-choice functional, host real equality, order trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier through Definition 4502.1. A consumer cannot reach the $\mathsf{CauchySpace}^{\uparrow}$ row Q before reading the side classifier L, the compatible windows W, the cofinal-window ledger J, the stream schedule S, the regular readback R, and the dyadic tolerance ledger D.

The terminal $\mathsf{Real}^{\uparrow}$ row E is even later in the displayed route. It is sealed only after the same Cauchy-space compatibility row has been exposed, so a Real-facing read cannot bypass the one-sided Cauchy schedule.

The structural rows H, C, P, N preserve, replay, package, and name this finite route. Because the carrier has no coordinate for the excluded host or choice objects, the L10 handoff remains a one-sided Cauchy seal route rather than a two-sided completion or quotient-real construction. □

Falsifiable prediction: Every $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ Real-facing read factors through a one-sided anchor, side classifier, finite approach windows, cofinal Cauchy-window ledger, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchySpace compatibility row, terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: one-sided-limit, cauchyspace, streamname, regseqrat, dyadicratcore, real: $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries side-tagged approach data and cofinal Cauchy-window certification before the terminal Real seal is exposed.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedCauchyLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedCauchyLimit}^{\uparrow}$ packet is generated from a one-sided approach anchor, a left/right side classifier, finite compatible windows, a cofinal Cauchy-window ledger, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerances, CauchySpace compatibility, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedCauchyLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4502.1, Definition 4502.2, and Definition 4502.3 seed the finite one-sided Cauchy route from side-tagged approach windows to a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedCauchyLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close two-sided limits, quotient stream equality, selected Cauchy tails, countable choice, host real equality, order trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, cofinal-window coverage, StreamName and RegSeqRat readback, DyadicRatCore tolerance exactness, CauchySpace compatibility, Real seal nonescape, transport, replay, provenance, and local naming for $\mathsf{OneSidedCauchyLimit}^{\uparrow}$.

Chapter 4503

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4503.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4503.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4503.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4504

A Concrete Naming Certificate for $\mathsf{BertrandTest}^{\uparrow}$

Following Bertrand and Hardy's classical treatment of logarithmic refinements of ratio-type convergence tests, the $\mathsf{BertrandTest}^{\uparrow}$ packet records the finite positive-series tail data by which a logarithmic first-order defect is routed into the existing BEDC convergence-test lattice. It sits beside the Raabe–Duhamel surface of chapter 1719, the auxiliary multiplier route of chapter 2113, the Gauss quotient-defect surface of chapter 2130, the ratio-test corridor of chapter 2244, and the regular-rational and Real endpoint seals of chapter 301 and chapter 38. The packet names only inspected source windows, adjacent quotient windows, logarithmic tail ledgers, dyadic comparison budgets, regular rational readback, Real sealing, transport, replay, provenance, and local naming; it is not host summability, quotient series equality, a selected sum, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Definition 4504.1 (Bertrand test carrier)**.** *A* $\mathsf{BertrandTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source positive-series tail row, exposing only the finite term and partial-sum windows inspected by the test.*
2. *Q is the adjacent quotient-window row, recording the displayed neighboring-term ratio data.*
3. *L is the Bertrand logarithmic tail ledger, storing the inspected index window, logarithmic scale row, rational margin, and first-order defect comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison-budget row used to route the logarithmic defect into a finite domination statement.*
5. *T is the Raabe–Duhamel tail handoff row, consumed only after S, Q, L, D have been displayed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic, domination, and handoff rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the tail handoff and stream windows have been replayed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal.*

9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic, domination, handoff, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RealSeriesRatioTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 (statement) `BEDC.Derived.BertrandTestUp`

Theorem 4504.2 (Bertrand test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandTest}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; quotient-window exactness over S, Q; logarithmic-tail stability over L; dyadic comparison routing through D; Raabe–Duhamel tail handoff through T; regular rational and Real endpoint routing through W, R, E; and transport, replay, provenance, and ledger non-escape through H, C, P, N.*

Proof. Project the carrier of Definition 4504.1. Carrier admission is exactly the displayed row inventory $S, Q, L, D, T, W, R, E, H, C, P, N$, with no detached host sum, quotient-series equality, selector, choice, trichotomy, completeness, Lean-verification, or bridge row.

Read the analytic evidence before the handoff. The source row S and quotient row Q expose only finite adjacent-tail windows, while the logarithmic ledger L records the inspected scale and defect margin attached to the same tail request.

The comparison and endpoint rows are then forced in order. The dyadic budget D turns the displayed logarithmic defect into finite domination evidence, the tail handoff T routes that evidence to the Raabe–Duhamel surface, and W, R, E expose only stream windows, regular rational readback, and the terminal Real seal.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no further coordinate, the listed host and quotient principles remain outside the local NameCert surface. □

Theorem 4504.3 (Bertrand test Raabe tail handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandTest}^{\uparrow}$ *carrier factors through the finite tail route*

$$\mathsf{BertrandTest}^{\uparrow} \longrightarrow \mathsf{RaabeDuhamelTest}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Concretely, the read consumes source series windows, adjacent quotient windows, a logarithmic tail ledger, dyadic comparison data, and a Raabe–Duhamel handoff before any $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The handoff may cite the neighboring* $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, *and* $\mathsf{RealSeriesRatioTest}^{\uparrow}$ *surfaces as dependency anchors, but it exports no host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.*

Proof. Begin with an endpoint-facing read of the accepted carrier. The only pre-handoff source rows are S, Q, L, D: finite source windows, adjacent quotient windows, the logarithmic tail ledger, and the dyadic comparison budget.

Apply the carrier order from Definition 4504.1. The handoff row T is downstream of S, Q, L, D, so a consumer cannot reach regular rational or Real endpoint data before the Raabe–Duhamel tail surface has consumed the displayed Bertrand evidence.

The neighboring Kummer, Gauss, and ratio-test chapters provide sibling anchors for the same quotient-defect lattice, but the exported route remains the finite sequence through T, W, R, E. The structural rows H, C, P, N keep that sequence attached to its transport, replay, provenance, and local naming data.

Because the accepted packet is exhausted by these rows, no host summability, quotient series equality, selected sum, countable-choice coordinate, trichotomy, ambient completeness principle, Lean target, or bridge row can be exported by the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandTest}^{\uparrow}}$.

Constructive story: A $\mathsf{BertrandTest}^{\uparrow}$ packet is generated from source positive-series windows, adjacent quotient windows, a logarithmic tail ledger, dyadic comparison data, a Raabe–Duhamel handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4504.1, Definition 4504.2, and Definition 4504.3 seed the finite Bertrand convergence-test boundary over displayed quotient, logarithmic, comparison, regular-rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandTest}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.BertrandTestUp`

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-tail stability, dyadic comparison routing, Raabe–Duhamel handoff, endpoint routing, transport, replay, provenance, and ledger non-escape for BEDC.Derived.BertrandTestUp.

Chapter 4505

A Concrete Naming Certificate for $\mathsf{CauchyContinuousExtensionTheorem}^{\uparrow}$

Following Bishop and Bridges constructive completion theory, the $\mathsf{CauchyContinuousExtensionTheorem}^{\uparrow}$ packet records the theorem-facing surface by which a Cauchy-controlled map is extended across the L10 completion route. It sits after the Cauchy-continuous extension packet of chapter 1632, the completion universal-property packet of chapter 397, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names the theorem row for Cauchy-controlled extension as finite NameCert data; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, choose limits, identify quotient streams, or provide host function equality.

Definition 4505.1 (Cauchy continuous extension theorem carrier)**.** *A* $\mathsf{CauchyContinuousExtensionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (E, U, S, R, D, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{CauchyContinuousExtension}^{\uparrow}$ *source row, exposing the Cauchy-controlled map and its finite extension route.*
2. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, recording the extension-and-uniqueness theorem surface read by the completion consumer.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose observations are opened before any regular readback is consumed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row produced from the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing the finite comparison budgets used by the Cauchy control.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after E, U, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for the extension, universal-property, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a theorem request through Cauchy extension, universal-property extension, finite windows, regular readback, dyadic tolerance, and terminal Real sealing.*

9. *P is the* Pkg *provenance over* CauchyContinuousExtensionTheorem$^{\uparrow}$, CauchyContinuousExtension$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CauchyContinuousExtensionTheorem}^{\uparrow}}$ *row.*

Accepted packets are compared only through the displayed extension, universal-property, finite-window, readback, tolerance, seal, transport, replay, provenance, and naming rows.

Theorem 4505.2 (Cauchy continuous extension theorem NameCert obligations)**.** *For every accepted* CauchyContinuousExtensionTheorem$^{\uparrow}$ *carrier $T = (E, U, S, R, D, A, H, C, P, N)$, the local* NameCert$_{\text{CauchyContinuousExtensionTheorem}^{\uparrow}}$ *surface consists exactly of Cauchy-continuous extension exposure, completion universal-property exposure, StreamName finite-window exposure, RegSeqRat readback exposure, DyadicRatCore tolerance exposure, Real seal exposure, componentwise transport, displayed replay, package provenance, and local naming. Every theorem-facing consumer must read E, U, S, R, D before it can consume the terminal seal A or the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 4505.1. The rows E and U expose the Cauchy-controlled extension surface and the completion universal-property row on the same packet.

Read the L10 source rows next. The finite StreamName$^{\uparrow}$ window row S, the RegSeqRat$^{\uparrow}$ readback row R, and the DyadicRatCore$^{\uparrow}$ tolerance ledger D are displayed before the theorem-facing consumer reaches the terminal Real$^{\uparrow}$ seal A.

Finally replay the structural tail. Componentwise transport through H, continuation through C, provenance through P, and local naming through N preserve the same finite route, so the packet contains no coordinate for selected limits, quotient stream equality, host function equality, countable choice, or ambient Real$^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted CauchyContinuousExtensionTheorem$^{\uparrow}$ reader that claims a theorem-facing extension handoff while omitting the Cauchy-continuous extension row, completion universal-property row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: cauchy-continuous-extension, completionuniversalproperty, streamname, regseqrat, dyadicratcore, real: CauchyContinuousExtensionTheorem$^{\uparrow}$ is not bijective with any listed sibling because it joins the theorem-facing extension row, universal-property row, finite windows, regular readback, dyadic tolerance, terminal Real seal, transport, replay, provenance, and local naming as one packet.

Closure status for NameCert$_{\text{CauchyContinuousExtensionTheorem}^{\uparrow}}$.

Constructive story: A CauchyContinuousExtensionTheorem$^{\uparrow}$ packet is generated from a CauchyContinuousExtension row, a CompletionUniversalProperty row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyContinuousExtensionTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 4505.1 and Definition 4505.2 seed the finite theorem-facing route for Cauchy-controlled extension across the L10 completion spine.

Formal verification: FormalStatus(NameCert$_{\text{CauchyContinuousExtensionTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, ambient Real$^{\uparrow}$ completeness, selected limits, quotient stream equality, countable choice, host function equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, extension exposure, universal-property exposure, finite-window readback, dyadic tolerance nonescape, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyContinuousExtensionTheoremUp.

Chapter 4506

A Concrete Naming Certificate for $\mathsf{HallMarriage}^{\uparrow}$

Following Hall's marriage theorem, the $\mathsf{HallMarriage}^{\uparrow}$ packet records a finite matching certificate whose only public witnesses are bipartite incidence rows, finite neighborhood ledgers, augmenting path traces, and local NameCert data. Its dependency route reads $\mathsf{Matching}^{\uparrow}$ from chapter 234, residual capacity bookkeeping from chapter 233, set membership rows from chapter 3668, decidable branch bits from chapter 32, and finite ordered witness rows from chapter 36.

Definition 4506.1 (HallMarriage finite matching carrier)**.** *The HallMarriage carrier is the finite* BHist *packet*

$$H = (L, R, E, N, S, B, A, M, U, P, C, T, Q)$$

where L and R are the left and right vertex lists, E is the bipartite incidence ledger, N records every finite neighborhood query as a $\mathsf{SetLike}^{\uparrow}$ *membership row, S is the Hall surplus table, B is the* $\mathsf{Bool}^{\uparrow}$ *decision trace for empty, saturated, and augmenting-path branches, A is the finite alternating-path row, M is the partial matching ledger read through* $\mathsf{Matching}^{\uparrow}$*, U is the residual update row read through* $\mathsf{NetworkFlow}^{\uparrow}$*, and P, C, T, Q are provenance, replay, componentwise transport, and local NameCert data.*

The carrier is accepted only when every witness is a displayed finite row. It contains no choice-selected transversal, infinite search tree, host cardinality comparison, or classical maximality principle.

Lean4 ✓ `BEDC.Derived.HallMarriageUp.HallMarriageTasteGate single carrier alignment`

Theorem 4506.2 (HallMarriage NameCert obligations)**.** *Every accepted* $\mathsf{HallMarriage}^{\uparrow}$ *certificate exposes the carrier of Definition 4506.1, verifies that each left sublist has a displayed neighborhood row in N, checks the surplus table S against those rows, and records each matching update in M, U. Thus the NameCert obligation surface is the finite route*

$$L, R, E, N, S, B, A, M, U, P, C, T, Q,$$

not an appeal to a selected system of representatives, an ambient finite-choice theorem, or a host maximum matching oracle.

Proof. Read the carrier rows in order. The vertex lists L, R and incidence ledger E determine every neighborhood query, while N records the resulting finite $\mathsf{SetLike}^{\uparrow}$ membership rows.

The Hall surplus table S is checked only against those displayed rows. The $\mathsf{Bool}^{\uparrow}$ row B then decides which finite branch is being replayed: already saturated, locally extendable, or routed through an alternating path.

The matching row M and residual update row U consume the same finite evidence through the $\mathsf{Matching}^{\uparrow}$ and $\mathsf{NetworkFlow}^{\uparrow}$ dependency surfaces. Since the carrier has no coordinate for choice, cardinality by quotient, or maximality outside the displayed ledgers, the local NameCert obligations are exactly the listed finite rows. □

Theorem 4506.3 (HallMarriage augmenting path induction)**.** *For an accepted carrier $H = (L, R, E, N, S, B, A, M, U, P, C, T, Q)$, every unmatched left vertex either is already assigned by the matching row M or is handled by a finite induction on the alternating-path row A. Each induction step consumes a displayed incidence entry from E, a neighborhood witness from N, a branch bit from B, and a residual update from U, and it strictly advances along the finite* $\mathsf{List}^{\uparrow}$ *path row.*

Proof. Start with the finite path row A. The base case is the empty path branch recorded by B, where the matching row M already assigns the left vertex or the surplus table exposes a free right vertex.

For the induction step, read the next incidence witness from E and its neighborhood membership certificate from N. The branch bit in B states whether the right endpoint is free or already matched.

If it is free, append the displayed edge to the matching row and record the residual update in U. If it is already matched, follow the unique displayed matching edge in M and continue along the tail of the finite $\mathsf{List}^{\uparrow}$ path.

The path row strictly decreases under this tail read, so the construction terminates without an infinite search tree. The resulting update remains inside M, U, P, C, T, Q, hence the augmenting-path induction is a finite HallMarriage NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HallMarriage}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HallMarriage}^{\uparrow}$ packet is generated from finite left and right vertex lists, a bipartite incidence ledger, finite neighborhood rows, a Hall surplus table, branch bits, alternating paths, matching updates, residual updates, provenance, replay, transport, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HallMarriage}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4506.1, Definition 4506.2, and Definition 4506.3 seed the finite matching certificate route for Hall's marriage theorem.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HallMarriage}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, infinite matching, selected transversals, ambient finite-choice principles, host cardinality comparison, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighborhood surplus verification, matching update stability, residual update stability, augmenting-path termination, transport, replay, provenance, and local naming for BEDC.Derived.HallMarriageUp.

Chapter 4507

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4507.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4507.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4508

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4508.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4508.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4509

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 4509.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 4509.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 4510

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 4510.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 4510.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 4511

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 4511.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* BaireTreeModulus$^{\uparrow}$, BaireSpace$^{\uparrow}$, ChoiceSequence$^{\uparrow}$, SpreadLaw$^{\uparrow}$, BrouwerContinuityPrinciple$^{\uparrow}$, StreamName$^{\uparrow}$, Nat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* NameCert$_{\text{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 4511.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* BaireTreeModulus$^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* NameCert$_{\text{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* SpreadLaw$^{\uparrow}$ *prefix and every inspected answer to be a displayed* ChoiceSequence$^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* BaireSpace$^{\uparrow}$ *stream schedule,* Nat$^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* BrouwerContinuityPrinciple$^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 4511.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* BaireTreeModulus$^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 4512

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 4512.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* LocatedUniformCauchyPrinciple$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 4512.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 4512.3 (Located uniform Cauchy NameCert)**.** *Every accepted* LocatedUniformCauchyPrinciple$^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local NameCert$_{\text{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and the finite* Real$^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names Real$^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A LocatedUniformCauchyPrinciple$^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 4513

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 4513.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 4513.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 4513.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 4513.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 4514

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 4514.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 4514.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 4514.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4515

A Concrete Naming Certificate for $\mathsf{RiemannIntegralCauchy}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4515.1 (Riemann integral Cauchy carrier)**.** *A* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 4515.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 4515.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 4516

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 4516.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 4516.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 4516.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 4517

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4517.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate
single carrier alignment

Theorem 4517.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 4517.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 4517.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 4518

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 4518.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 4518.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 4518.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 4518.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 4519

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 4519.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 4519.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 4519.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 4519.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 4519.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 4519.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 4520

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 4520.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* BanachFixedPointStability$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 4520.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4520.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* BanachFixedPointStability$^{\uparrow}$ *packet factors through the finite route*

$$\text{BanachFixedPoint}^{\uparrow} \longmapsto \text{PicardIteration}^{\uparrow} \longmapsto \text{ContractionMapping}^{\uparrow} \longmapsto \text{CompleteMetric}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the RegSeqRat$^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The Real$^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 4520.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 4521

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 4521.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 4521.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 4522

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 4522.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 4522.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 4522.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 4522.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, seed)

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 4523

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 4523.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 4523.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 4523.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 4524

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 4524.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 4525

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 4525.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 4525.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 4526

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 4526.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 4526.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 4526.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 4527

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4527.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 4527.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 4527.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 4527.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 4527.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}\uparrow}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 4528

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 4528.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 4528.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 4528.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4529

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4529.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 4529.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4529.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 4529.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 4530

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 4530.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 4530.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 4531

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4531.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 4531.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4531.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 4531.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 4531.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 4531.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 4532

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 4532.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 4532.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 4532.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 4533

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 4533.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 4533.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4533.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 4534

A Concrete Naming Certificate for $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ packet records the bounded real-sequence subsequence route as finite NameCert data. It sits after the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the stream-window interface of chapter 126, and the located nested-interval limit surface of chapter 5820. The packet is a constructive boundary object for bounded real sequences: it exposes a finite subsequence ledger and a terminal Real seal, not classical sequential compactness, open-cover compactness, countable choice, a host subsequence selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4534.1 (Real Bolzano–Weierstrass carrier)**.** *A* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, I, L, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence row, exposing only the finite source coordinates inspected by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision ledger, recording the dyadic interval cuts, diameter budgets, and containment checks used at each finite stage.*
3. *I is the finite nested-window row, storing the selected interval windows and their inclusion witnesses.*
4. *L is the subsequence ledger, recording the monotone index schedule produced by the same finite windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the dyadic subdivisions and the subsequence ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, D, I, L, R have been displayed.*
7. *Q is the refusal row, recording that no classical compactness theorem, countable-choice selector, host subsequence, or ambient completeness object is part of the carrier.*
8. *H records componentwise* hsame *transport for the bounded source, dyadic subdivision ledger, nested windows, subsequence ledger, RegSeqRat readback, Real seal, refusal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from bounded-sequence inspection through dyadic subdivision, nested-window refinement, subsequence ledgering, readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, I, L, R, E, Q, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.RealBolzanoWeierstrassUp.TasteGate.RealBolzanoWeierstrassTasteGate single carrier alignment`

Theorem 4534.2 (Real Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, D, I, L, R, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded Real sequence admission through S, dyadic interval subdivision through D, finite nested-window discipline through I, subsequence ledger exactness through L, RegSeqRat readback through R, terminal Real sealing through E, refusal of the excluded compactness and choice rows through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5350.1. The rows $S, D, I, L, R, E, Q, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded sequence is read first through S. The dyadic subdivision ledger D and nested-window row I then fix the finite interval route before any subsequence or Real-facing read is available.

The ledger row L records the monotone index schedule selected by those same windows. The readback row R consumes the dyadic and subsequence evidence, and the Real seal E is downstream of that readback.

The refusal and structural rows add no hidden analytic principle. Row Q blocks the excluded compactness, choice, host-selector, and completeness coordinates, while H, C, P, N transport, replay, source, and name exactly the displayed finite packet. □

Theorem 4534.3 (Real subsequence readback route)**.** *Every* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *consumer that exports a cluster-facing real read must factor through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LocatedNestedIntervalLimit}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported read is supported by bounded source rows, dyadic interval subdivision, finite nested windows, a subsequence ledger, RegSeqRat readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence.*

Proof. Apply Definition 5350.2. It places bounded source admission, dyadic subdivisions, nested windows, subsequence ledgering, and regular rational readback before the terminal Real seal can be consumed.

The source and target occurrences of $\mathsf{Real}^{\uparrow}$ are not quotient representatives. The first supplies bounded sequence names through finite StreamName windows, while the second is the terminal seal reached after the dyadic and regular-rational route has been displayed.

Transport and replay preserve the same packet. Thus any cluster-facing read follows the stated finite route and cannot bypass it through classical sequential compactness, open-cover compactness, countable choice, a host subsequence selector, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ cluster-facing read factors through one bounded RealUp sequence row, one DyadicRatCore subdivision ledger, one finite nested-window row, one subsequence ledger, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, dyadicratcore, streamname, located-nested-interval-limit: $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded real-sequence rows, dyadic interval subdivision, finite nested windows, a subsequence ledger, regular rational readback, and a terminal Real seal for cluster-facing consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5350.1, Definition 5350.2, and Definition 5350.3 seed the bounded real-sequence subsequence route over dyadic subdivision, finite nested windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical sequential compactness, open-cover compactness, countable choice, host subsequence selection, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded sequence visibility, dyadic interval subdivision, finite nested-window exactness, subsequence-ledger exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealBolzanoWeierstrassUp.

Constructive story: A $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a DyadicRatCore subdivision ledger, finite nested windows, a subsequence ledger, RegSeqRat readback, a terminal RealUp seal, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4535

A Concrete Naming Certificate for CandidateSetClosedRoute$^{\uparrow}$

The CandidateSetClosedRoute$^{\uparrow}$ packet realizes the candidate-set branch of chapter 9383 as finite NameCert data. It records the route by which closed typed MetaCIC terms inherit strong normalization from candidate-set membership while preserving the candidate evidence as an explicit row. The route consumes the positive closed-normal consistency surface of chapter 736 only as an obstruction-facing boundary: before the obstruction row O can refuse unconditional consistency, the closed-normal packet has already fixed the normal-fragment node that is not being promoted to full closed consistency. The packet is a normalization-audit surface, not a blanket theorem that every closed typed term is strongly normalizing without candidate-mediated evidence.

The route consumes the sibling candidate-set normalization surface of chapter 2320 and the MetaCIC normalization frontier route of chapter 892 before it exports closed typed term strong-normalization reads. CandidateSetClosedRoute$^{\uparrow}$ isolates the closed-term endpoint, while CandidateSetNormalization$^{\uparrow}$ supplies the candidate-set evidence and filter route that make the endpoint readable.

Definition 4535.1 (Candidate-set closed route carrier)**.** *A* CandidateSetClosedRoute$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, M, S, A, B, NF, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed term row, including the empty-context typing and closedness data consumed by the MetaCIC closed-term projection.*
2. *M is the candidate-set membership row, storing the displayed membership witness rather than erasing it into an unconditional endpoint.*
3. *S is the candidate-set soundness row, exposing the route from membership to strong normalization.*
4. *A is the application and Pi adequacy row, recording the candidate-mediated handoff used by function and argument reads.*
5. *B is the beta-closure row, keeping the candidate-set evidence stable under the displayed one-step reduction route.*
6. *NF is the no-infinite-reduction projection row downstream of the same strong-normalization evidence.*

7. *O is the obstruction row naming closed-normal consistency, substitution-audit, confluence-audit, and subject-reduction boundaries that remain outside this packet.*
8. *H records componentwise* hsame *transport for closed typed terms, candidate-set membership, soundness, adequacy, beta closure, no-infinite-reduction projection, obstruction, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from candidate-set membership through closed typed term strong normalization and no-infinite-reduction reads.*
10. *P is the* Pkg *provenance over* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$*, the MetaCIC normalization audit map, closed-normal consistency, substitution-audit map, confluence-audit map, subject-reduction obstruction,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, S, A, B, NF, O, H, C, P, N$.

Theorem 4535.2 (Candidate-set closed route NameCert obligations)**.** *For every accepted* CandidateSetClosedRoute *carrier*

$$R = (T, M, S, A, B, NF, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$ *surface consists exactly of closed typed term admission through T, candidate-set membership through M, candidate-set soundness through S, application and Pi adequacy through A, beta-closure stability through B, no-infinite-reduction projection through NF, obstruction visibility through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5351.1. The closed typed term row, candidate-set membership row, soundness row, adequacy row, beta-closure row, no-infinite-reduction row, obstruction row, and structural rows are the only exposed coordinates.

The membership row M is not optional evidence. It is the datum consumed by S to route the closed typed term in T to strong normalization, and the adequacy row A keeps application and Pi reads inside the same candidate-mediated interface.

Rows B and NF preserve the endpoint discipline. Beta closure stays attached to the candidate set, while no-infinite-reduction is projected downstream of the same strong-normalization route.

Finally O, H, C, P, N record the boundary, transport, replay, source, and local name. Since no unconditional closed-term strong-normalization coordinate is present, the listed obligations exhaust the packet. □

Theorem 4535.3 (Candidate-set closed strong-normalization route)**.** *Every* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *strong-normalization read factors through candidate-set membership evidence before it reaches the closed typed term endpoint. The route preserves the candidate-set soundness row, beta-closure row, application and Pi adequacy row, no-infinite-reduction projection, obstruction ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Apply Definition 5351.2. The closed typed term row T supplies the endpoint shape, but it does not itself supply strong normalization.

The route obtains strong normalization only after projecting the membership row M and soundness row S. Application and Pi reads remain governed by A, and beta steps remain governed by B.

The no-infinite-reduction projection NF is downstream of the same evidence. The obstruction and structural rows preserve the conditional reading, so the exported endpoint remains candidate-mediated rather than unconditional. □

Theorem 4535.4 (Candidate-set route TasteGate obligation)**.** *Future Lean support for* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *must keep the candidate-set membership route nontrivial and field-faithful. A* Nontrivial $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *instance must distinguish an accepted packet with displayed candidate-set membership from a closed typed term row without that membership evidence, and a* FieldFaithful $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *instance must preserve the row order*

$$T \longmapsto M \longmapsto S \longmapsto A \longmapsto B \longmapsto NF \longmapsto O \longmapsto H, C, P, N$$

before any TasteGate carrier-alignment theorem can serve as a checked realization.

Proof. Nontriviality is forced by the carrier boundary: a closed typed term row without M cannot export the same strong-normalization read as an accepted packet with candidate-set membership evidence.

Field faithfulness is the row-order requirement. If a gate projection permutes or erases $M, S, A, B, NF, O, H, C, P$ the candidate-mediated route is replaced by a detached endpoint claim, which is not the packet named by Definition 5351.1. □

Falsifiable prediction: Every accepted $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ strong-normalization read names the candidate-set-to-closed-SN route: closed typed term data first, candidate-set membership evidence second, candidate-set soundness third, application and Pi adequacy fourth, beta closure fifth, no-infinite-reduction projection sixth, and obstruction, transport, replay, provenance, and local naming rows thereafter.

Independence witness: metacic-normalization-audit-map, closed-normal-consistency, substitution-audit-map, confluence-audit-map, subject-reduction-obstruction: $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ is not bijective to any listed map because it isolates the candidate-set membership route from closed typed terms to strong normalization without erasing the membership evidence or closing the subject-reduction, substitution, and confluence boundaries.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5351.1, Definition 5351.2, Definition 5351.3, and Definition 5351.4 seed the candidate-set-mediated closed typed term strong-normalization route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization for every closed typed MetaCIC term, subject reduction, arbitrary substitution preservation, global confluence, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-set membership visibility, candidate-set soundness, application and Pi adequacy routing, beta-closure stability, no-infinite-reduction projection, obstruction visibility, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.CandidateSetClosedRouteUp

Constructive story: A $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ packet is generated from closed typed term data, candidate-set membership evidence, candidate-set soundness, application and Pi adequacy rows, beta closure, no-infinite-reduction projection, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4536

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 4536.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 4536.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 4536.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 4536.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 4537

A Concrete Naming Certificate for $\mathsf{BisectionReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BisectionReal}^{\uparrow}$ packet records bisection as a finite nested-interval and bracket-refinement certificate over the L10 Real route. It sits after the dyadic tolerance surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested interval route of chapter 1924. The packet names the displayed bisection ledger; it is not a Dedekind cut, quotient real, trichotomy principle, selected root, or completeness theorem.

Definition 4537.1 (BisectionReal carrier)**.** *A* $\mathsf{BisectionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, S, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval source row, recording the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, interval containment, and the finite retained subintervals consumed by the bisection route.*
2. *M is the dyadic midpoint and tolerance ledger. It records the midpoint rows, branch radii, diameter budgets, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons used at each inspected stage.*
3. *S is the sign or bracket-preservation row, storing the displayed bracket invariant that survives each retained child interval.*
4. *T is the tail request row, recording the finite precision request and the inspected bisection depth needed to answer it.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained endpoint and midpoint names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from W and M.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the real name exported by the retained nested interval route.*
8. *H records componentwise* hsame *transport for the interval source, midpoint ledger, bracket row, tail request, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through interval inspection, midpoint production, bracket preservation, dyadic tolerance readback, stream-window exposure, regular readback, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{BisectionReal}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, S, T, W, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient-real representatives, real trichotomy, selected roots, host equality of real points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BisectionRealUp.BisectionRealTasteGate single carrier alignment`

Theorem 4537.2 (BisectionReal NameCert obligations)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier*

$$B = (I, M, S, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval source, dyadic midpoint and tolerance ledger, bracket-preservation row, tail request, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *nested-source discipline, requiring every retained interval read to pass through I before midpoint or Real-facing data are available;*
3. *midpoint and tolerance discipline, requiring M to expose only dyadic midpoint, radius, and diameter rows whose budgets answer the request T;*
4. *bracket preservation, requiring S to carry the displayed sign or bracket invariant across each retained child interval;*
5. *readback discipline, requiring W and R to expose the retained interval only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
6. *seal nonescape, requiring E, H, C, P, N to export only the displayed bisection route and no Dedekind cut, quotient real, trichotomy principle, selected root, host equality, countable choice, or completeness theorem.*

No BisectionReal obligation is stored outside these rows.

Proof. Project the carrier of Definition 5711.1. The accepted packet displays exactly the nested interval source, midpoint ledger, bracket row, tail request, window rows, readback, Real seal, and structural rows.

The finite source stage is I, T. A precision request opens the source interval row and the requested bisection depth before any midpoint or retained endpoint can be inspected.

The bisection stage is M, S. A finite induction over the retained branch ledger reads one midpoint row at a time, uses the dyadic tolerance entries to bound the retained child, and carries the displayed sign or bracket invariant into the next interval row.

The readback stage is W, R, E. The retained endpoints are exposed as finite stream windows, then as regular rational readbacks, and only then as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, package, and name the same route. Since the carrier contains no Dedekind-cut, quotient-real, trichotomy, selected-root, host-equality, countable-choice, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4537.3 (BisectionReal nested interval factorization)**.** *Every* $\mathsf{BisectionReal}^{\uparrow}$ *read factors through the displayed nested interval route*

$$I, T \longmapsto M, S \longmapsto W, R \longmapsto E, H, C, P, N.$$

In particular, a Real-facing consumer first reads the $\mathsf{NestedIntervalReal}^{\uparrow}$ *source row* I*, then the finite dyadic midpoint and bracket-preservation rows* M, S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* W, R*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal and local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

Proof. Start with the accepted carrier $B = (I, M, S, T, W, R, E, H, C, P, N)$. The only interval source coordinate is I, and the only request coordinate is T, so a bisection read begins by exposing those rows.

Induct on the finite branch prefix recorded in the midpoint ledger. The base case reads the initial interval and its bracket row. In the successor step, the midpoint row in M supplies the dyadic split, the tolerance entry supplies the diameter budget, and S records that the displayed bracket invariant is preserved for the retained child.

After the last retained child, the endpoint windows are read through W and the rational data through R. The Real-facing seal E is therefore downstream from the nested interval, midpoint, bracket, stream, and regular readback rows.

The structural rows H, C, P, N do not add a second route. They transport componentwise hsame, replay the same displayed Cont chain, record Pkg provenance, and supply the local name. Thus every read factors through the stated finite route. □

Theorem 4537.4 (BisectionReal regular Cauchy handoff)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier* $B = (I, M, S, T, W, R, E, H, C, P, N)$*, the handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is available only after the retained nested-interval rows, dyadic midpoint and tolerance rows, sign or bracket-preservation rows, and finite* $\mathsf{StreamName}^{\uparrow}$ *windows have been replayed through the same* BHist *packet. The exported object is a regular Cauchy readback with a Real seal, not a selected root, quotient representative, trichotomy decision, or completeness witness.*

Proof. Use the nested interval factorization of Definition 5711.3. It places W, R, E strictly after the source row I, request row T, midpoint ledger M, and bracket row S.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the row R. It is computed from the finite stream windows in W and the dyadic tolerance entries in M, so no downstream consumer can read R without replaying the retained bisection ledger.

The $\mathsf{Real}^{\uparrow}$ handoff is the row E. It seals the regular readback already obtained from R, while H, C, P, N preserve the same packet by componentwise hsame, displayed Cont, Pkg, and local NameCert data.

Since the carrier has no selected-root, quotient-representative, trichotomy-decision, host-equality, countable-choice, Dedekind-cut, or completeness coordinate, the handoff exposes only the regular Cauchy route stated in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionReal}^{\uparrow}$ packet is generated from a nested interval source row, dyadic midpoint and tolerance ledger, sign or bracket-preservation rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, seed)

Scope closed: Definition 5711.1, Definition 5711.2, Definition 5711.3, and Definition 5711.4 seed the finite constructive bisection route from nested interval rows through dyadic midpoint, bracket-preservation, stream-window, regular readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, trichotomy on reals, selected roots, host equality of real points, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source discipline, midpoint and tolerance exactness, bracket preservation, regular Cauchy handoff, RealUp seal nonescape, and local naming for BEDC.Derived.BisectionRealUp.

Chapter 4538

A Concrete Naming Certificate for $\mathsf{DarbouxCriterion}^{\uparrow}$

Following Darboux, Riemann, and Bishop–Bridges constructive integration, the $\mathsf{DarbouxCriterion}^{\uparrow}$ packet records the upper-lower gap test as finite NameCert evidence. It sits between the lower and upper Darboux sum packets of chapter 5858 and chapter 4083, the Darboux-sum surface of chapter 1880, and the Riemann-facing consumers of chapter 1960 and chapter 1526. The packet exposes a displayed partition, lower sum, upper sum, gap, tolerance, regular rational readback, and real seal; it does not import host suprema, measure completion, a selected mesh, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4538.1 (Darboux criterion carrier)**.** *A* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, L, U, G, T, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition row, recording the inspected interval endpoints and refinement requests.*
2. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row attached to P.*
3. *U is the* ***??*** *row attached to the same partition row P.*
4. *G is the upper-minus-lower gap row comparing U and L inside the displayed rational ledger.*
5. *T is the tolerance row, recording the positive rational request against which G is checked.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same finite partition and gap evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after P, L, U, G, T, R have been displayed.*
8. *H records componentwise* hsame *transport for partition, lower-sum, upper-sum, gap, tolerance, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through P, L, U, G, T, R, E.*
10. *Q is the* Pkg *provenance over* $\mathsf{DarbouxCriterion}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, ***??***, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host suprema or infima, measure completion, a selected mesh, an ambient completeness principle, a quotient real, or a global integration theorem.

Theorem 4538.2 (Darboux criterion NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier*

$$K = (P, L, U, G, T, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, partition exposure, lower-sum and upper-sum compatibility over the same partition, gap comparison against the tolerance row, regular rational readback, terminal real-seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $P, L, U, G, T, R, E, H, C, Q, N$.

Lean4 ✓ `BEDC.Derived.DarbouxCriterionUp.DarbouxCriterionTasteGate single carrier alignment`

Proof. Project the carrier. The accepted packet exposes the partition row P, the paired lower and upper Darboux sum rows L, U, the gap row G, the tolerance row T, the regular readback row R, the seal row E, and the structural rows H, C, Q, N.

The lower and upper sum rows are tied to the same partition row, so a gap comparison cannot switch partitions while the criterion is being read. The tolerance row fixes the request against which the displayed gap is checked.

The regular readback and real seal are downstream of those finite rows. Componentwise transport and replay preserve the same order, while provenance and naming record the listed sibling packets.

Since host suprema, measure completion, selected meshes, quotient reals, ambient completeness, and global integration theorems are absent from the carrier coordinates, the displayed rows exhaust the local NameCert surface. □

Theorem 4538.3 (Darboux criterion upper-lower gap route)**.** *Every* $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *or regulated-integral-facing read of an accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *packet factors through the finite route*

$$P, L, U \longmapsto G, T \longmapsto R \longmapsto E.$$

Thus the consumer first reads a displayed partition, lower Darboux sum, and upper Darboux sum; then it reads the upper-minus-lower gap and tolerance rows; then it reads the $\mathsf{RegSeqRat}^{\uparrow}$ *hand-off and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The packet exports no direct route through host suprema, measure completion, selected mesh choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 5712.2. They require carrier admission and classifier transport through the displayed rows of Definition 5712.1.

The partition row P is read before either sum row can be used. The rows L and U are therefore paired over the same finite partition, and G is the displayed comparison between those paired rows.

The tolerance row T is the only request against which the gap is consumed. After that finite comparison has been displayed, the read may pass to R and then to the real seal row E.

The excluded routes are not coordinates of the carrier. Hence every integration-facing consumer receives the upper-lower gap criterion only through the finite partition, sum, gap, tolerance, regular-readback, and seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxCriterion}^{\uparrow}$ packet is generated from a displayed partition, lower and upper Darboux sum rows, an upper-minus-lower gap row, a tolerance row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5712.1, Definition 5712.2, and Definition 5712.3 seed the finite upper-lower gap criterion for Riemann-facing and regulated-integral-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DarbouxCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, measure completion, selected mesh choice, quotient real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, paired lower and upper sum compatibility, gap tolerance comparison, regular readback, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxCriterionUp.

Chapter 4539

A Concrete Naming Certificate for $\mathsf{RiemannSumConsistency}^{\uparrow}$

Following Bishop and Bridges on finite Riemann sums and Darboux brackets, the $\mathsf{RiemannSumConsistency}^{\uparrow}$ packet records the finite route by which a displayed Riemann-sum row is compared with Darboux envelope rows over a common mesh before any integration consumer reads it. It sits between the tagged-sum carrier of chapter 1615, the Darboux upper/lower surface of chapter 1880, the mesh-refinement carrier of chapter 3496, the regulated-function packet of chapter 1906, and the terminal seal of chapter 38. The packet names a finite consistency boundary; it does not assert full Riemann integrability, Lebesgue integration, selected limits, host real equality, or choice.

Definition 4539.1 (RiemannSumConsistency carrier)**.** *A* $\mathsf{RiemannSumConsistency}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, M, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *row, carrying the tagged finite sum, tag-containment witnesses, cell widths, and finite arithmetic replay.*
2. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *envelope row, carrying the lower sum, upper sum, and finite upper-minus-lower gap over the same displayed cell family.*
3. *M is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *row, exposing the common finite mesh, endpoint adjacency, and refinement witness through which both S and D are read.*
4. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *row, recording the finite oscillation and local constancy data used to place the tagged values of S between the Darboux brackets of D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, M, F have been displayed.*
6. *H records componentwise* hsame *transport for the Riemann-sum row, Darboux envelope row, finite mesh row, regulated-function row, real seal, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a mesh-refinement request through M, then through the Riemann-sum and Darboux rows S, D, then through the regulated-function check F, and finally to the seal row E.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannSumConsistency}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannSumConsistency}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, D, M, F, E, H, C, P, N$. It has no coordinate for a completed Riemann integral, Lebesgue integration, a selected limiting partition sequence, host equality of real functions, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness beyond the displayed seal.*

Lean4 ✓ `BEDC.Derived.RiemannSumConsistencyUp.RiemannSumConsistencyTasteGate single carrier alignment`

Theorem 4539.2 (RiemannSumConsistency NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSumConsistency}^{\uparrow}$ *carrier*

$$K = (S, D, M, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RiemannSumConsistency}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Riemann-sum exposure through S, Darboux envelope exposure through D, common finite mesh exposure through M, regulated-function compatibility through F, terminal real-seal routing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 4539.1. The accepted packet exposes exactly the Riemann-sum row S, Darboux envelope row D, mesh row M, regulated-function row F, real seal E, and structural rows H, C, P, N.

The mesh row is read before either comparison side is consumed. Hence the tagged sum and the Darboux lower/upper sums are both attached to the displayed finite cell family named by M.

The regulated-function row supplies the finite compatibility check. It records only the carried oscillation and local-read data needed to compare the tag values in S with the lower and upper brackets in D.

Transport and replay are componentwise. The row H preserves the same displayed rows, and C replays the route from mesh request to tagged sum, Darboux envelope, regulated-function comparison, and terminal seal.

Finally inspect P and N. They source and name only the listed rows, so full integrability, Lebesgue integration, selected limits, host real equality, quotient real equality, choice, and ambient completeness have no projection from the packet. □

Theorem 4539.3 (RiemannSumConsistency Darboux bridge)**.** *Every Darboux-facing read of an accepted* $\mathsf{RiemannSumConsistency}^{\uparrow}$ *carrier factors through the finite route*

$$M \longmapsto (S, D) \longmapsto F \longmapsto E.$$

The consumer receives the tagged finite sum of S and the Darboux lower/upper envelope of D over the mesh M, with F certifying the finite regulated-function comparison before the real seal E is exposed. No read bypasses this route by importing a host integral, a Lebesgue-measure object, a selected limiting sequence of partitions, host real equality, or choice.

Proof. Start with the obligation surface of Definition 4539.2. It requires carrier admission through the rows of Definition 4539.1.

The mesh row M is the common source of cells. Once M is displayed, the Riemann-sum row S supplies the finite tagged-sum side, and the Darboux row D supplies the lower and upper envelope side over the same cells.

The row F is then the only compatibility row between those sides. It records the regulated-function evidence that the tag-value reads used by S sit inside the Darboux bracket reads carried by D.

The real seal E is downstream of this finite comparison. Since the excluded analytic objects are not carrier coordinates, a Darboux-facing consumer can only read the displayed consistency bridge through $M, S, D, F, E, H, C, P, N$. □

Theorem 4539.4 (RiemannSumConsistency mesh refinement)**.** *For every accepted* $\mathsf{RiemannSumConsistency}^{\uparrow}$ *carrier, any displayed refinement of the finite mesh row M must transport the Riemann-sum row S, Darboux envelope row D, regulated-function row F, and real seal E through the same componentwise* hsame *and* Cont *replay rows. The refined packet remains a finite consistency certificate and exports no selected tag family, selected limiting mesh, host function equality, Lebesgue integration row, or ambient completeness principle.*

Proof. Project the mesh row of the carrier. A refinement request is visible only through the $\mathsf{FinitePartitionMesh}^{\uparrow}$ row M and the stored structural rows H, C.

Transport acts componentwise on the comparison rows. The hsame row H moves the Riemann-sum, Darboux envelope, regulated-function, and real-seal coordinates to the displayed refinement without adding a new coordinate.

Replay then rebuilds the comparison route. The Cont row C reads the refined mesh first, then replays the tagged-sum side, Darboux envelope side, regulated-function compatibility check, and terminal seal in the same order.

Thus refinement changes only the displayed finite mesh data and its transported comparison rows. The packet still contains no selected tag family, limiting mesh sequence, host function equality, Lebesgue integration object, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumConsistency}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannSumConsistency}^{\uparrow}$ packet is generated from a RiemannSumUp row, DarbouxSumUp envelope row, FinitePartitionMeshUp row, RegulatedFunctionUp row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSumConsistency}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4539.1, Definition 4539.2, Definition 4539.3, and Definition 4539.4 seed the finite consistency route between Riemann sums, Darboux envelopes, mesh refinement, regulated-function comparison, and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSumConsistency}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full Riemann integrability, Lebesgue integration, selected limits, host real equality, quotient real equality, choice, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Riemann-sum exposure, Darboux envelope exposure, finite mesh refinement, regulated-function compatibility, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumConsistencyUp.

Chapter 4540

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 4540.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 4540.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 4540.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 4541

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 4541.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. *N is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. *M is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. *F is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. *H records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform tolerance request through $K, N, M, F, U, T, H, P, L$.*
9. *P is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 4541.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 4541.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 4542

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 4542.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 4542.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 4542.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 4542.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 4542.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 4542.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \text{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 4543

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 4543.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 4543.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 4543.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 4544

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4544.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 4544.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4544.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 4545

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 4545.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 4545.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 4545.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 4545.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 4545.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 4545.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 4546

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 4546.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4546.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 4546.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 4546.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 4547

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 4547.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 4547.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 4547.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 4547.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 4548

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4548.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4548.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4548.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 4548.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 4548.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 4548.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 4548.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 4548.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 4548.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 4548.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 4548.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 4548.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 4548.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 4548.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 4548.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 4549

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 4549.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4549.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 4549.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 4549.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 4549.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 4549.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 4549.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 4549.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 4549.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 4549.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 4550

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 4550.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 4550.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4550.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 4550.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 4551

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 4551.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 4551.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 4551.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let $L = (S, Q, I, U, F, K, T, H$ be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, seed)

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

$\mathsf{RieszThorinInterpolation}^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 4552

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 4552.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4552.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4552.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 4552.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 4553

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4553.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 4553.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 4553.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 4553.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 4554

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4554.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4554.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4554.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4555

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4555.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 4555.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4555.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 4555.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 4556

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 4556.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 4556.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 4556.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 4557

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 4557.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 4557.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4557.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 4558

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4558.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4558.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4558.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4558.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4558.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4559

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4559.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4559.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4559.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4559.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4560

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4560.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4560.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4560.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4561

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4561.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4561.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4561.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4561.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4562

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4562.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 4562.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4562.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4563

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4563.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4563.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4563.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4564

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4564.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 4564.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4565

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4565.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4565.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4565.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4566

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4566.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4566.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4566.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4566.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4567

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4567.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4567.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4567.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4568

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4568.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4568.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4568.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4569

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4569.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 4569.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4569.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4570

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4570.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4570.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4570.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4570.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4571

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4571.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4571.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4571.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4571.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4572

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4572.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4572.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4572.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4573

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4573.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4573.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4573.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4573.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4573.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4574

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4574.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4574.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4574.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4575

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4575.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4575.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4575.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4575.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4575.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4575.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4576

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4576.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4576.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4576.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4577

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4577.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4577.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4577.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4578

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4578.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4578.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4578.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4578.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4579

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4579.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4579.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4579.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4580

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4580.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4580.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4580.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4580.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4581

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4581.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4581.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4581.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4581.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4582

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4582.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4582.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4582.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4583

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4583.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4583.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4583.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4584

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4584.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4584.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4584.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4584.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4584.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4585

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4585.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4585.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4586

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4586.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4586.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4586.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4587

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4587.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4587.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4587.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4588

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4588.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4588.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4588.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4589

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4589.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4589.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4589.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4590

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4590.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4590.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4591

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4591.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4591.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4591.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4592

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4592.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4592.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4592.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4593

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4593.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4593.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4593.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4594

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4594.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4594.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4594.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 4595

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4595.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 4595.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4595.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4596

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4596.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4596.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier $U = (S, A, F, D, C, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4596.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4597

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4597.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4597.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4597.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4597.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4598

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4598.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4598.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4598.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4599

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4599.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4599.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4599.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4600

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4600.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4600.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4600.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4601

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4601.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate single carrier alignment`

Theorem 4601.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4601.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\text{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4602

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4602.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4602.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4602.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4603

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4603.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4603.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4603.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4604

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 4604.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 4604.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 4604.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4605

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4605.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4605.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4606

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4606.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 4606.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4606.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4607

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 4607.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 4607.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4607.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 4607.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4608

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4608.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4608.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4608.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4608.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4609

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4609.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4609.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4609.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4610

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4610.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4610.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4610.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4611

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4611.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4611.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4611.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4612

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4612.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4612.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4613

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4613.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4613.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4614

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

4614.1 Carrier definition

Definition 4614.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

4614.2 Certificate surface

Theorem 4614.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 4614.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 4614.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 4615

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 4615.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 4615.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 4615.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4616

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4616.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4616.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4616.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4616.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4616.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4617

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4617.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4617.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4617.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4617.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4618

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4618.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4618.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4618.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4619

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4619.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4619.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4619.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4619.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4620

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4620.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4620.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4620.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4621

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4621.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4621.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4621.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4622

A Concrete Naming Certificate for DiniUniformLimitModulus$^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the DiniUniformLimitModulus$^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 4622.1 (Dini uniform-limit modulus carrier)**.** *A* DiniUniformLimitModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* RegSeqRat$^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* UniformModulus$^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* UniformLimit$^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* RegSeqRat$^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* Real$^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 4622.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4623

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4623.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4623.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4623.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4624

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4624.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4624.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4624.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4624.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4625

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4625.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4625.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4625.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4626

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4626.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4626.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4626.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4627

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4627.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 4627.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4627.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4628

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4628.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4628.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4628.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4628.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4629

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4629.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4629.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4629.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4629.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4630

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4630.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4630.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4630.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4631

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4631.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4631.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4631.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4631.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4631.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4632

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4632.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4632.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4632.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4633

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4633.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4633.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4633.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4633.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4633.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4633.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4634

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4634.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4634.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4634.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4635

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4635.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4635.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4635.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4636

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4636.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4636.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4636.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4636.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4637

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4637.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4637.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4637.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4638

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4638.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4638.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4638.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4638.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4639

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4639.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4639.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4639.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4639.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4640

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4640.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4640.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4640.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4641

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4641.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4641.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4641.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4642

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4642.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4642.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4642.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N.$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4642.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4642.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4643

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4643.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4643.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4644

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4644.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. *T is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after W, T, D, R have been exposed.*
6. *H records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. *C records displayed* Cont *replay of the same finite filter route.*
8. *P records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4644.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4644.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4645

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4645.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4645.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4645.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4646

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4646.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4646.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4646.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4647

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4647.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4647.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4647.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4648

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4648.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4648.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4649

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4649.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4649.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4649.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4650

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4650.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4650.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4650.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4651

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4651.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4651.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4651.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4652

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4652.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4652.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4652.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 4653

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 4653.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 4653.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 4653.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 4653.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 4653.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 4653.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 4653.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 4653.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 4654

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4654.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* Real$^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* CauchyCompletionMonadicity$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionAlgebr CauchyCompletionKleisli$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, MetricCompletionDenseEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 4654.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4654.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* CauchyCompletionMonadicity$^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4655

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4655.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4655.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4655.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, seed)

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4656

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4656.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*
10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*
12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4656.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*
2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*
3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*
4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*
5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*
6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4656.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4656.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4657

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 4657.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 4657.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4657.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 4657.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 4657.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 4657.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 4658

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4658.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4658.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4658.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4659

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4659.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4659.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4659.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4660

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4660.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4660.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4660.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4661

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4661.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 4661.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4661.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4662

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4662.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4662.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4662.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4663

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4663.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4663.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4663.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4664

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 4664.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 4664.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 4664.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 4664.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4665

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4665.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4665.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4665.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4665.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4666

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4666.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4666.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4666.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4667

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 4667.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 4667.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 4667.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 4667.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 4667.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 4668

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4668.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4668.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4668.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4669

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4669.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4669.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4669.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4670

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4670.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4670.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForm

Chapter 4671

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4671.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4671.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4672

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 4672.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 4672.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 4673

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 4673.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 4673.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 4674

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 4674.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 4674.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 4674.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 4675

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 4675.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* LocatedUniformCauchyPrinciple$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 4675.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 4675.3 (Located uniform Cauchy NameCert)**.** *Every accepted* LocatedUniformCauchyPrinciple$^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local NameCert$_{\text{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and the finite* Real$^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names Real$^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A LocatedUniformCauchyPrinciple$^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 4676

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 4676.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 4676.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 4676.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 4676.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 4677

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 4677.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 4677.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 4677.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4678

A Concrete Naming Certificate for RiemannIntegralCauchy$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegralCauchy$^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient Real$^{\uparrow}$ completeness.

Definition 4678.1 (Riemann integral Cauchy carrier)**.** *A* RiemannIntegralCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* RiemannSumGauge$^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* DarbouxSum$^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* RiemannIntegrable$^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* RiemannIntegralCauchy$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 4678.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 4678.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 4679

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 4679.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 4679.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 4679.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 4680

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4680.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate single carrier alignment

Theorem 4680.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 4680.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 4680.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 4681

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 4681.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 4681.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 4681.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 4681.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 4682

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 4682.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 4682.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 4682.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 4682.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 4682.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 4682.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 4683

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 4683.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BanachFixedPointStability}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 4683.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4683.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BanachFixedPoint}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the $\mathsf{RegSeqRat}^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The $\mathsf{Real}^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 4683.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 4684

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 4684.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 4684.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 4685

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 4685.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 4685.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 4685.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 4685.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 4686

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 4686.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 4686.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 4686.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 4687

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 4687.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 4688

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 4688.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 4688.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 4689

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 4689.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 4689.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 4689.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 4690

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4690.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 4690.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 4690.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 4690.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 4690.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 4691

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 4691.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 4691.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 4691.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4692

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4692.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 4692.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4692.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 4692.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 4693

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 4693.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 4693.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 4694

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4694.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 4694.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4694.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 4694.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 4694.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 4694.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 4695

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 4695.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 4695.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 4695.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 4696

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 4696.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 4696.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4696.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 4697

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 4697.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 4697.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 4697.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 4697.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 4698

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 4698.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 4698.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 4698.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 4699

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 4699.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. K *is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. N *is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. M *is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. F *is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. U *is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. T *is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. H *records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a compact-uniform tolerance request through* $K, N, M, F, U, T, H, P, L$.
9. P *is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate single carrier alignment`

Theorem 4699.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 4699.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 4700

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 4700.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 4700.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 4700.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 4700.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The sandwich hub routes its cofinal common-tail read through chapter 7532 before the child files expose the lower and upper RegSeqRat readbacks.

Chapter 4701

A Concrete Naming Certificate for $\mathsf{RegularCauchySandwich}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet names the two-sided bracketing surface used when regular Cauchy data is read by a real-completion consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the tail agreement boundary of chapter 1738. The packet records lower and upper regular tails as finite BEDC rows; it is not a quotient stream, a selected limit, a trichotomy principle, or an ambient completeness theorem.

Definition 4701.1 (Regular Cauchy sandwich carrier)**.** *A* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *U is the upper regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
3. *T is the common tail-agreement row inherited from* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling both bracket sides at the same precision request.*
5. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the lower and upper sides.*
6. *R_L and R_U are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed after the windows are displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both sides, the common tail row, the dyadic ledger, the windows, and the readbacks have been consumed.*
8. *H records componentwise* hsame *transport for the lower side, upper side, tail row, ledger, windows, readbacks, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each finite precision request through*

$$L, U, T, D, W_L, W_U, R_L, R_U \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySandwich}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the lower and upper regular rows, common tail agreement, dyadic tolerance ledger, finite windows, readbacks, and terminal seal belong to the same displayed bracketing route. It has no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4701.2 (Regular Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *surface consists exactly of lower regular-row exposure, upper regular-row exposure, common tail-agreement exposure, shared dyadic tolerance accounting, lower and upper finite-window exposure, lower and upper RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through N.*

Proof. Project the carrier tuple in Definition 5874.1. The lower and upper sides are displayed as L and U, and the only row that binds them is the common tail-agreement coordinate T.

The shared ledger D fixes the precision request for both sides. The finite windows W_L, W_U are opened against that same ledger, and the readback rows R_L, R_U expose exactly the inspected regular rational data.

The seal row E is downstream of those rows, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic coordinate occurs in the carrier, the local NameCert surface is exactly the displayed bracketing route. □

Theorem 4701.3 (Regular Cauchy sandwich window bracketing)**.** *Let*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. Every finite precision request consumed by the packet reads the two-sided route*

$$L, U, T, D \longmapsto W_L, W_U \longmapsto R_L, R_U \longmapsto E, H, P, N.$$

Thus lower and upper regular windows are bracketed by one displayed tolerance ledger and one common tail-agreement row, not by quotient representatives, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient completeness.

Proof. Start with a finite precision request. The packet exposes only one dyadic tolerance ledger D, so both sides of the bracket are read against the same displayed tolerance data.

The lower and upper windows W_L, W_U are then opened after the common tail row T has been displayed. Their readbacks R_L, R_U remain attached to the same packet, so neither side can be replaced by an external quotient stream or a selected representative.

The terminal seal E is reached only after both readbacks. Transport, replay, provenance, and local naming through H, C, P, N preserve that order for every accepted request, which gives the claimed two-sided bracketing route. □

Theorem 4701.4 (Regular Cauchy sandwich Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier can bypass the displayed route through the lower and upper regular rows, common tail agreement, shared dyadic ledger, lower and upper StreamName windows,*

lower and upper RegSeqRat readbacks, and local structural rows. Equivalently, every accepted terminal seal read is attached to

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

and cannot be separated from those rows by quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 5874.2. It exposes the Real seal row E only as part of the local NameCert surface containing the two regular rows, common tail row, ledger, windows, readbacks, and structural rows.

Now use Definition 5874.3. Every precision request reaches the readbacks through the same two-sided route, so a seal read detached from one side of the bracket is not a read of this packet.

Finally inspect Definition 5874.1. The carrier contains no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient Real completeness. A purported seal read using such data therefore lies outside the accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ carrier. □

Theorem 4701.5 (Regular Cauchy sandwich Cantor intersection seal)**.** *Let* $S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H$ *be an accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. If the lower and upper rows are supplied by a nested-shrink or Cantor-intersection route, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed by* S *is available only through the displayed sandwich route, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. In particular, the consumer-facing seal factors through Definition 5874.3, Definition 5874.4, chapter 4227, and chapter 7754, without asserting a selected intersection point.*

Proof. Apply Definition 5874.3. It forces every finite precision request for S through the common tail row, shared dyadic ledger, two StreamName windows, and two RegSeqRat readbacks before the seal row is visible.

The nested-shrink and Cantor-intersection inputs supply the lower and upper rows only as finite interval or shrink data. Those routes provide the source rows for the sandwich, but they do not add a coordinate for a selected point, quotient stream equality, or ambient completeness.

Now apply Definition 5874.4. Any terminal $\mathsf{Real}^{\uparrow}$ read detached from the displayed carrier is outside the accepted packet, so the Cantor-facing consumer receives only the finite sandwich seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet is generated from lower and upper regular-Cauchy rows, a common tail-agreement row, a shared dyadic tolerance ledger, lower and upper StreamName windows, lower and upper RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, seed)

Scope closed: Definition 5874.1, Definition 5874.2, Definition 5874.3, and Definition 5874.4 seed the finite two-sided regular-Cauchy bracketing route over DyadicRatCore, StreamName, RegSeqRat, Real, and CauchyTailEquivalence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common tail exactness, shared dyadic tolerance accounting, two-sided StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySandwichUp.

Theorem 4701.6 (Regular Cauchy sandwich sibling dependency route)**.** *Every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *sandwich packet depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows of chapter 301 and the* $\mathsf{StreamName}^{\uparrow}$ *pointwise window rows of chapter 126 before it exposes a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route consumes those siblings only through the displayed lower and upper windows, lower and upper regular readbacks, common tail row, shared dyadic ledger, transport, replay, provenance, and local naming rows of Definition 5874.1.*

Proof. Read the carrier rows from Definition 5874.1. The lower and upper windows W_L, W_U are the pointwise $\mathsf{StreamName}^{\uparrow}$ windows, and the rows R_L, R_U are the regular rational readbacks supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

The bracketing theorem Definition 5874.3 orders every precision request through those windows and readbacks before the seal row is consumed. The Real-seal nonescape theorem Definition 5874.4 prevents bypass through quotient stream equality, selected limits, host real equality, or ambient completeness.

Thus the chapter's sibling dependency is the concrete route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, not an external analytic carrier. □

Theorem 4701.7 (Regular Cauchy sandwich cofinal tail-mesh dependency)**.** *Every accepted* $\mathsf{RegularCauchySandwic}$ *packet whose common tail row is presented cofinally must read the* $\mathsf{CofinalTailMesh}^{\uparrow}$ *route of chapter 7532 before the lower and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are consumed. The dependency is the displayed route through the sandwich carrier, the window bracketing theorem, and the sibling dependency route; it is not an appeal to quotient streams, selected limits, trichotomy, choice, host real equality, or ambient completeness.*

Proof. Read the carrier rows in Definition 5874.1. The common tail coordinate T is displayed before the lower and upper StreamName windows, the lower and upper RegSeqRat readbacks, and the Real seal row.

When T is presented cofinally, consume the CofinalTailMesh route of chapter 7532. That route supplies the shared finite tail-window mesh before any Real or completion-facing seal is read.

Apply the window bracketing theorem Definition 5874.3. It keeps the lower and upper readbacks downstream of the common tail row and the shared dyadic ledger.

Finally use the sibling dependency route Definition 5874.6. It already requires the StreamName and RegSeqRat siblings before the Real seal, so adding the cofinal tail-mesh read fixes the horizontal dependency without adding any external analytic coordinate. □

Chapter 4702

A Concrete Naming Certificate for $\mathsf{DynamicProgramming}^{\uparrow}$

Following Bellman's finite dynamic programming principle, the $\mathsf{DynamicProgramming}^{\uparrow}$ packet records finite backward recursion data as BEDC NameCert rows. It sits after the finite path and edge ledger of chapter 236, the transition-row discipline of chapter 189, the probability-row carrier of chapter 184, the terminal real-name seal of chapter 38, and the finite horizon counter of chapter 28. The chapter names only finite states, finite actions, displayed transition and cost rows, horizon-indexed Bellman tables, policy readback, transport, replay, provenance, and local naming; it does not assert an infinite-horizon fixed point, host minimizer, choice-selected policy, quotient value equality, or ambient completeness theorem.

Definition 4702.1 (Dynamic programming carrier)**.** *A* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, T, C, H, V, P, R, N)$$

with the following displayed rows:

1. *S is the finite displayed state row, sharing its path endpoints with chapter 236.*
2. *A is the finite action row, listing exactly the actions admitted at the displayed states.*
3. *T is the transition row, read through the finite Markov and probability surfaces of chapter 189 and chapter 184.*
4. *C is the cost row, exposing the finite real-valued immediate costs and terminal costs through the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *H is the horizon row, a* $\mathsf{Nat}^{\uparrow}$*-indexed finite counter for the backward recursion stages.*
6. *V is the Bellman value table, giving one displayed value row for each state and horizon stage.*
7. *P is the policy readback row, recording only actions whose value entry is witnessed by the same finite transition and cost rows.*
8. *R records componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance over S, A, T, C, H, V, P and the cited sibling packets.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $S, A, T, C, H, V, P, R, N$. It has no coordinate for a host minimizer, selected infinite policy, analytic contraction theorem, quotient value function, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DynamicProgrammingUp.DynamicProgrammingTasteGate single carrier alignment`

Theorem 4702.2 (Dynamic programming Bellman obligations)**.** *For every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier* $D = (S, A, T, C, H, V, P, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *surface consists exactly of carrier admission for finite states, finite actions, transition rows, cost rows, horizon rows, Bellman value-table rows, policy readback rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host minimizers, selected infinite policies, quotient value equality, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 5875.1. The finite state and action rows are displayed before transition and cost rows can be read, and the horizon row fixes which backward stages may appear.

The Bellman table row V is then read stage by stage over the same displayed horizon. A policy readback row P is exposed only after the corresponding state, action, transition, cost, and value rows have all been displayed.

Transport, replay, provenance, and local naming are exhausted by R and N. Since the carrier has no further coordinate beyond $S, A, T, C, H, V, P, R, N$, no accepted read can import a host minimizer, selected infinite policy, quotient value equality, countable choice, or ambient completeness. □

Theorem 4702.3 (Dynamic programming finite-horizon induction)**.** *Every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *Bellman table is generated by finite backward induction over the displayed* $\mathsf{Nat}^{\uparrow}$ *horizon row. The terminal stage reads the terminal cost row from* C *and the* $\mathsf{Real}^{\uparrow}$ *seal; each predecessor stage reads only the previously displayed successor-stage value row, the transition row* T*, the action row* A*, the immediate cost row* C*, and the same local NameCert route.*

Proof. Project the horizon row H and read it through the unary source discipline of chapter 28. The empty endpoint is the terminal stage, so the value entry is the displayed terminal cost row sealed through $\mathsf{Real}^{\uparrow}$.

For a one-step predecessor, use the closed generation of the horizon row. The successor-stage table has already been displayed, and the new value row is computed from the finite transition row, finite action row, immediate cost row, and successor value row.

Repeating the one-step construction along the finite horizon supplies every value-table row. No step asks for an infinite horizon, a global fixed point, or an external minimizer; the induction is only the BEDC finite-history recursion named by the carrier. □

Theorem 4702.4 (Dynamic programming policy readback)**.** *Every policy row exported by an accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *packet is a readback from the displayed Bellman table. For each displayed state and horizon stage, the row* P *may expose an action only after the state row* S*, action row* A*, transition row* T*, cost row* C*, horizon row* H*, value row* V*, componentwise transport, replay, provenance, and local naming have all been read.*

Proof. Start with a policy-facing read. The carrier forces the read through the finite state and action rows, then through the transition and cost rows that define the local Bellman entry.

Next read the horizon row and the value-table row. The policy row is not an independent selector: it is attached to the displayed value entry at the same state and horizon stage.

Finally apply the structural row R and local naming row N. These rows preserve the finite route under transport and replay, so any action exposed by P remains tied to the same Bellman table entry and cannot become a host choice function or infinite policy. □

Theorem 4702.5 (Dynamic programming NameCert package)**.** *The* $\mathsf{DynamicProgramming}^{\uparrow}$ *NameCert package is the finite Bellman-recursion packet whose public surface is exactly the route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{ShortestPath}^{\uparrow}, \mathsf{MarkovChain}^{\uparrow}, \mathsf{ProbSpace}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DynamicProgramming}^{\uparrow}.$$

It exports the state, action, transition, cost, horizon, value-table, policy-readback, transport, replay, provenance, and local naming rows of Definition 5875.1. It exports no host optimization theorem, no selected global policy, no infinite-horizon convergence theorem, no quotient value-function equality, and no bridge to a standard dynamic-programming model.

Proof. The cited sibling chapters supply the input rows: shortest-path data for finite route endpoints, Markov and probability data for transition rows, real data for cost and value seals, and natural-number data for the finite horizon.

Apply Definition 5875.2 to identify the local obligation surface. Apply Definition 5875.3 to generate the value table from the terminal stage backward through the displayed horizon. Apply Definition 5875.4 to expose policy rows only as readbacks from that table.

The package is therefore a finite NameCert surface over the displayed rows. Because no carrier coordinate stores host optimization, global choice, infinite convergence, quotient equality, or a standard bridge, those objects are not exported. □

Falsifiable prediction: A claimed $\mathsf{DynamicProgramming}^{\uparrow}$ read that exposes a policy action without first displaying S, A, T, C, H, V, R, N refutes the finite Bellman carrier.

Independence witness: shortestpath, markovchain, probspace, real, nat: no listed sibling is bijective to $\mathsf{DynamicProgramming}^{\uparrow}$ because this packet binds finite state-action data, transition and cost rows, horizon-indexed Bellman value tables, policy readback, transport, replay, provenance, and local naming as one finite recursion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DynamicProgramming}^{\uparrow}$ packet is generated from finite state rows, finite action rows, transition rows, cost rows, a Nat-indexed horizon, Bellman value-table rows, policy readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5875.1, Definition 5875.2, Definition 5875.3, Definition 5875.4, and Definition 5875.5 seed the finite Bellman-recursion NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host minimizers, selected infinite policies, quotient value equality, infinite-horizon convergence, ambient RealUp completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Bellman table construction, finite-horizon induction, policy readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.DynamicProgrammingUp.

Chapter 4703

Regular Cauchy Even-Odd Substreams NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet records the standard constructive passage from one regular Cauchy name to its even and odd substreams before a Real or Completion consumer reads either side. It sits over the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the two-source interleaving packet of chapter 414, and the finite interleaving criterion of chapter 3405. The packet supplies a displayed subsequence route for the L10 Real and Completion corridor; it does not choose a new limit, quotient streams, or import ambient completeness.

Definition 4703.1 (Regular Cauchy even-odd substreams carrier)**.** *A* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, E, O, D, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row. It names one* $\mathsf{RegSeqRat}^{\uparrow}$ *source together with the finite precision requests inspected by the parity split.*
2. *E is the even selector-window row. It displays the finite windows read from the source at indices $2n$ and records no schedule beyond the even selector.*
3. *O is the odd selector-window row. It displays the finite windows read from the source at indices $2n + 1$ and records no schedule beyond the odd selector.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row. It stores the shared dyadic radius, split tolerance, comparison windows, and common-scale data used by both selector windows.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the even and odd windows under the budget D.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the even readback, available only after S, E, D, R have been displayed.*
7. *B is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the odd readback, available only after S, O, D, R have been displayed.*

8. *H records componentwise* hsame *transport for the source, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an even or odd substream request through $S, E, O, D, R, A, B, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, O, D, R, A, B, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.RegularCauchyEvenOddSubstreamsTasteGate`
`single carrier alignment`

Theorem 4703.2 (Regular Cauchy even-odd substreams NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier*

$$X = (S, E, O, D, R, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, transport, replay, provenance, and local naming rows;*
2. *source visibility, requiring both parity branches to read the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row S;*
3. *even-window exactness, requiring every even branch request to pass through the selector windows E and the budget D before any readback is visible;*
4. *odd-window exactness, requiring every odd branch request to pass through the selector windows O and the same budget D before any readback is visible;*
5. *paired regular readback admission through R, requiring the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to use the selector windows and the shared dyadic ledger already displayed;*
6. *two Real seal boundaries through A and B, requiring the even and odd Real-facing reads to remain downstream from their parity windows, D, R, H, and C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly $S, E, O, D, R, A, B, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5876.1. Its displayed inventory is exactly $S, E, O, D, R, A, B, H, C, P, N$, so the source row, parity selector windows, dyadic budget, paired readbacks, two seals, and structural rows exhaust the packet.

The source visibility clause is read from S. Both parity branches share that row, so neither branch can replace the source by a second hidden regular name.

Even-window and odd-window exactness are read from E and O under the shared dyadic row D. The even branch exposes only indices of the form $2n$, while the odd branch exposes only indices of the form $2n+1$, and both branches use the same tolerance ledger before the readback row is consumed.

The paired readback row R is downstream from those windows. It repacks the two selector routes as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks without adding a quotient stream or selected host limit.

Finally the Real seals A and B are reached only after the corresponding parity route has been transported by H and replayed by C. Since P and N name only the displayed packet, the listed rows are the complete local NameCert surface. □

Theorem 4703.3 (Regular Cauchy even-odd Real-seal nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier factors through one of the two finite parity routes*

$$S, E, D \longmapsto R \longmapsto A \longmapsto H, C, P, N,$$
$$S, O, D \longmapsto R \longmapsto B \longmapsto H, C, P, N.$$

The even route reads the even selector windows and the odd route reads the odd selector windows, but both routes share the same source row, dyadic budget, paired regular readback, transport, replay, provenance, and local naming. Hence no consumer reaches a $\mathsf{Real}^{\uparrow}$ *seal through quotient stream equality, a chosen host limit, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the obligation surface of Definition 5876.2. It confines every accepted packet to the displayed source, two parity selector rows, one dyadic budget, paired readback, two seal rows, and structural data.

For an even request, the consumer reads S, E, D before R and then reaches A. The route has no coordinate other than the even selector windows and the shared budget, so the even seal cannot bypass the displayed packet.

For an odd request, the consumer reads S, O, D before R and then reaches B. This is the same source and budget route with the odd selector row in place of the even selector row.

The structural rows H, C, P, N transport, replay, source, and name the two routes componentwise. Because the carrier contains no quotient stream equality, selected host limit, countable-choice row, trichotomy branch, or ambient completeness coordinate, every Real or Completion consumer remains inside one of the two displayed parity routes. □

Theorem 4703.4 (Regular Cauchy parity cofinality)**.** *Let* $X = (S, E, O, D, R, A, B, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier over one source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*. The even selector windows* E *and the odd selector windows* O *determine cofinal tail-shift reads of that same source row. Their Real-facing seals factor through the same ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The assertion is only this cofinal finite-window factorization. It does not identify quotient streams, select a limit, use countable choice, use trichotomy, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source row S and the two selector-window rows E, O from Definition 5876.1. The rows are displayed as windows over one source $\mathsf{RegSeqRat}^{\uparrow}$ carrier, so no second regular name is introduced by the parity split.

Apply Definition 301.257 to transport late reads along the even and odd tail shifts. The tail-shift result keeps each parity branch inside the same regular-rational source route.

Next use Definition 126.670. It supplies one finite request section for the displayed selector windows, so both parity reads are compared at the finite StreamName surface rather than at a completed stream quotient.

Apply Definition 1647.3 to fuse the two parity budgets through the shared dyadic tolerance ledger. The shared D row is the DyadicRatCore face of the ordered packet.

Finally use Definition 38.713. The terminal Real seal is reached only after the dyadic budget, finite StreamName window, and RegSeqRat readback have been displayed. Thus the two parity seals are cofinal views of the same finite four-face route, with no selected limit, countable-choice selector, trichotomy branch, quotient stream equality, or ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ Real or Completion read is refuted if it reaches either $\mathsf{Real}^{\uparrow}$ seal without displaying the source RegSeqRat row, the selected even or odd StreamName windows, the shared DyadicRatCore budget, the paired RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-interleaving, cauchy-interleaving-criterion: $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one source regular Cauchy name, two parity selector-window routes, one shared dyadic budget, paired regular readbacks, two Real seal boundaries, transport, replay, provenance, and local naming as one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet is generated from one RegSeqRat source, even selector windows, odd selector windows, a shared DyadicRatCore budget, paired RegSeqRat readbacks, two RealUp seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5876.1, Definition 5876.2, Definition 5876.3, and Definition 5876.4 seed the parity-substream route used by L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parity-window exactness, shared dyadic budget exactness, paired regular readback admission, two Real-seal nonescape clauses, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.

Chapter 4704

A Concrete Naming Certificate for $\mathsf{SchwartzSpace}^{\uparrow}$

Following Schwartz and Treves on test-function spaces, the $\mathsf{SchwartzSpace}^{\uparrow}$ packet records the finite locally convex carrier surface behind rapidly decreasing smooth test functions. It sits beside the complete locally convex interface of chapter 3495, the finite family-seminorm route of chapter 5868, the pointwise test-function carrier of chapter 3399, and the derivative observation route of chapter 2210. The packet gives Fourier, tempered-distribution, weak-derivative, and PDE-facing consumers a displayed test-function-space surface without importing a host topological vector space, quotient function equality, unrestricted seminorm enumeration, selected representatives, or classical completeness.

Definition 4704.1 (SchwartzSpace carrier)**.** *A* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, V, I, G, D, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetSpace}^{\uparrow}$ *source row, exposing complete metrizability and locally convex handoff only through the displayed Frechet carrier.*
2. *V is the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite-family row, naming the seminorm labels used by the Schwartz test-function space read.*
3. *I is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-function row, recording the accepted rapidly decreasing smooth windows inspected by the packet.*
4. *G is the polynomial-growth and rapid-decay ledger. It binds each displayed seminorm index to a derivative order, a polynomial weight, and the real-valued budget read used to test decay.*
5. *D is the* $\mathsf{WeakDerivative}^{\uparrow}$ *stability row, recording derivative-window shifts and sign conventions that remain inside the same Schwartz test-function-space surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row used by seminorm values, decay budgets, and completion-facing estimates.*
7. *T is the completion and distribution nonescape row. It permits Frechet and tempered-distribution consumers to read the displayed test-function-space boundary only after F, V, I, G, D, R have been exposed.*

8. *H records componentwise* hsame *transport for Frechet, seminorm-family, test-function, rapid-decay, derivative, Real, nonescape, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every test-function-space request through $F, V, I, G, D, R, T, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Schwartz topology, a quotient of functions, an unrestricted seminorm family, a selected completion point, a distribution dual, nuclearity, Fourier inversion, or classical choice.

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.SchwartzSpaceTasteGate single carrier alignment`

Theorem 4704.2 (SchwartzSpace NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier*

$$S = (F, V, I, G, D, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Frechet-space, seminorm-family, Schwartz-function, rapid-decay, weak-derivative, Real, nonescape, transport, replay, provenance, and local naming rows;*
2. *seminorm growth routing, requiring every seminorm read to choose a displayed finite-family row in V and a polynomial-growth budget in G before a Real value can be consumed from R;*
3. *rapid-decay test-function closure, requiring every accepted test window from I to satisfy the displayed derivative-order and polynomial-weight ledger in G;*
4. *derivative-row stability, requiring weak-derivative shifts to pass through D and return to the same I, G, R surface rather than to host smooth-function equality;*
5. *completion and distribution nonescape, requiring T to mediate every Frechet, completion-facing, Fourier-facing, tempered-distribution, or PDE-facing read after F, V, I, G, D, R have been displayed.*

No $\mathsf{SchwartzSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.TasteGate.SchwartzSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5877.1. The analytic rows are exactly F, V, I, G, D, R, T, and the structural rows are exactly H, C, P, N.

Seminorm growth routing is forced by the order of the carrier. A consumer first reads the finite family row V, then the derivative and polynomial-weight budget in G, and only then may read a Real scalar estimate through R.

Rapid-decay closure and derivative stability are the two test-function-space checks. The row I supplies accepted Schwartz test windows, while G records their rapid-decay budgets and D records the weak-derivative shifts that stay inside the same displayed surface.

Finally T is a boundary row rather than an additional topological object. It lets Frechet, completion-facing, Fourier-facing, tempered-distribution, and PDE consumers replay the same rows without receiving a host Schwartz topology, quotient function equality, selected completion point, distribution dual, nuclearity, inversion theorem, or choice principle. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 4704.3 (SchwartzSpace functional-analysis handoff)**.** *Every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *packet exported to a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing consumer factors through*

$$\mathsf{FrechetSpace}^{\uparrow},\ \mathsf{SeminormedSpace}^{\uparrow},\ \mathsf{SchwartzFunction}^{\uparrow},\ \mathsf{WeakDerivative}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto F, V, I, G, D, R, T, H, C, P, N.$$

The exported object is a finite test-function-space NameCert surface with displayed seminorm growth budgets and derivative stability. It is not a host locally convex function space, an unrestricted seminorm enumeration, a selected completion, a dual-space quotient, or a classical distribution theorem.

Proof. Open the accepted carrier using Definition 5877.1. Any functional-analysis request must first choose one of the displayed source rows F, V, I, D, R and then pass through the growth ledger G.

Use Definition 5877.2. The seminorm and rapid-decay obligations keep polynomial weights and derivative orders attached to finite displayed rows, while the derivative-row obligation prevents a consumer from replacing weak-derivative stability by host smooth-function equality.

The nonescape row T is then the only export boundary. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route for every downstream consumer. Since no excluded host object is a carrier coordinate, the displayed route is the whole handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SchwartzSpace}^{\uparrow}$ consumer that reaches a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing read without first exposing $F, V, I, G, D, R, T, H, C, P, N$ refutes the carrier surface of Definition 5877.1.

Independence witness: frechet-space, seminormed-space, schwartz-function, weak-derivative, real: no listed sibling is bijective to $\mathsf{SchwartzSpace}^{\uparrow}$ because this packet jointly binds complete locally convex source rows, finite seminorm-family exposure, rapid-decay test-function windows, derivative stability, Real budget readback, completion and distribution nonescape, transport, replay, provenance, and local naming in one test-function-space carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchwartzSpace}^{\uparrow}$ packet is generated from a FrechetSpace row, a finite SeminormedSpace family row, SchwartzFunction test windows, rapid-decay and polynomial-growth ledgers, WeakDerivative stability rows, Real scalar readback, completion and distribution nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5877.1, Definition 5877.2, and Definition 5877.3 seed the finite locally convex test-function-space surface consumed by Frechet, Fourier, tempered-distribution, weak-derivative, and PDE-facing packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Schwartz topology, quotient function equality, unrestricted seminorm enumeration, selected completion, distribution duality, nuclearity, Fourier inversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seminorm growth routing, rapid-decay test-function closure, derivative-row stability, completion and distribution nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SchwartzSpaceUp.

Chapter 4705

A Concrete Naming Certificate for $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$

The $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet realizes the private/shared decoration model of chapter 9390 as finite Rule 110 classifier-decoration NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the finite pattern catalog of chapter 995, the Rule 110 classifier of chapter 5844, and the finite partition carrier of chapter 201. The packet names quotient support families, private blocks, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, transport, replay, provenance, and local naming; it does not import a host cellular semantics, an infinite orbit classifier, quotient support equality, a biological model, or a hidden enumeration oracle.

Definition 4705.1 (Rule 110 private/shared decoration carrier)**.** *A* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient support family row, naming the finite family of support blocks inspected by the decoration.*
2. *B is the private-block row, recording for each accepted support a displayed nonempty private coordinate block.*
3. *S is the shared-coordinate incidence row. For each shared coordinate it records the displayed set of incident private blocks and requires cardinality at least two.*
4. *R is the* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *row, attaching the decorated support family to finite Rule 110 row-window and pattern-tag reads.*
5. *E is the decoration-count provenance row. It records the partition and row-set enumeration data used by the private/shared decoration theorem without exposing a host generating-function oracle.*
6. *H records componentwise* hsame *transport for quotient support, private-block, shared-incidence, classifier, enumeration, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decoration request through Q, B, S, R, E, H, P, N.*

8. *P is the* Pkg *provenance over* RuleOneTenPrivateSharedDecoration$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, CellularPatternCatalog$^{\uparrow}$, RuleOneTenPatternClassifier$^{\uparrow}$, Partition$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for absent private blocks, singleton shared incidence, host quotient equality, unbounded Rule 110 evolution, biological interpretation, or an external enumeration procedure.

Theorem 4705.2 (Rule 110 private/shared decoration NameCert obligations)**.** *For every accepted* RuleOneTenPrivateSharedDecoration$^{\uparrow}$ *carrier*

$$D = (Q, B, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of quotient support family admission through Q, private-block nonemptiness through B, shared-coordinate incidence through S, Rule 110 pattern-classifier routing through R, decoration-count provenance through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5878.1. The decoration data are exactly the quotient support row Q, the private-block row B, the shared-incidence row S, the Rule 110 classifier row R, and the enumeration provenance row E.

Private-block nonemptiness is read before a support can be accepted. Shared-coordinate incidence is then read as a displayed row set of size at least two, so a shared coordinate cannot be collapsed into a private coordinate or erased from the decoration.

The Rule 110 classifier row R attaches the decorated support family to finite row-window and pattern-tag reads from the cellular siblings. The count provenance row E records the finite partition and row-set enumeration route and does not supply an external counting oracle.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since no excluded host object is present as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4705.3 (Private/shared decoration classifier handoff)**.** *Every accepted* RuleOneTenPrivateSharedDecorati *consumer that reads a decorated Rule 110 support family factors through*

$$\text{CellularAutomaton}^{\uparrow},\ \text{CellularRowClassifier}^{\uparrow},\ \text{CellularPatternCatalog}^{\uparrow},\ \text{RuleOneTenPatternClassifier}^{\uparrow},\ \text{Partition}^{\uparrow} \longmapsto Q$$

The exported object is a finite classifier-decoration packet with private-block nonemptiness, shared incidence, and decoration-count provenance. It is not an infinite cellular-orbit classifier, a host quotient-support equality, a biological stochastic model, or an unrestricted enumeration theorem.

Proof. Open the accepted carrier using Definition 5878.1. A support-family read can inspect only Q, and each support must expose its nonempty private block through B before shared coordinates are consumed.

The shared-coordinate row S supplies the incidence sets whose size is at least two. The classifier row R then routes the accepted decoration into the existing cellular automaton, row-classifier, pattern-catalog, and Rule 110 pattern-classifier surfaces.

The provenance row E records the finite decoration-count route, while H, C, P, N preserve the same data under transport, replay, package provenance, and local naming. Thus every accepted consumer receives exactly the displayed finite packet and none of the excluded host semantics. □

Falsifiable prediction: Any accepted $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ consumer that omits private-block nonemptiness or shared-coordinate incidence rows while reading a decorated Rule 110 support family refutes the carrier surface of Definition 5878.1.

Independence witness: cellularautomaton, cellularrowclassifier, cellularpatterncatalog, rule110patternclassifier: no listed sibling is bijective to $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ because this packet jointly binds quotient support families, nonempty private blocks, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, transport, replay, provenance, and local naming.

The Lean TasteGate file for this packet must provide $\mathsf{Nontrivial}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ instances for the displayed rows $Q, B, S, R, E, H, C, P, N$ before the decoration carrier can serve as a verified classifier-decoration consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet is generated from quotient support family rows, nonempty private-block rows, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5878.1, Definition 5878.2, and Definition 5878.3 seed the finite private/shared classifier-decoration surface for Rule 110 support-family consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite Rule 110 orbit classification, host quotient-support equality, biological interpretation, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient support admission, private-block nonemptiness, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPrivateSharedDecorationUp.

Chapter 4706

A Concrete Naming Certificate for $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ packet records the finite diagonal obstruction to enumerating Cantor-space bit names. It sits over the binary prefix carrier of chapter 1475, the finite bar-facing tree discipline of chapter 1622, the fan-theoretic bounded-window route of chapter 400, and the Real-facing non-surjection boundary of chapter 3793. The packet names only finite rows: an attempted enumeration window, its displayed diagonal bit schedule, the disagreement row at each inspected index, and the no-enumeration handoff. It is not a host-cardinality theorem, not a completed powerset construction, and not a quotient-stream principle.

Definition 4706.1 (Constructive Cantor finite diagonal carrier)**.** *A* $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, W, D, G, T, F, R, H, C, P, N)$$

with the following visible rows:

1. *E is the displayed finite enumeration attempt row, listing the first n candidate Cantor-space names through finite prefix windows;*
2. *W is the finite binary observation window, naming the inspected coordinates and the finite prefix depth used for each candidate;*
3. *D is the diagonal-bit row, obtained by flipping the displayed ith bit of the ith candidate whenever that coordinate is present in W;*
4. *G is the disagreement ledger, recording for every inspected index $i < n$ that the diagonal row disagrees with the ith candidate at the displayed coordinate;*
5. *T is the tree-read row through* $\mathsf{KleeneTree}^{\uparrow}$*, ensuring that finite prefix extension and absence of a completed path are read as finite branch data;*
6. *F is the fan-window row through* $\mathsf{FanTheorem}^{\uparrow}$*, recording the bounded prefix surface consumed by any compact Cantor-space handoff;*
7. *R is the Real-facing route through* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$*, used only after the Cantor-space disagreement ledger has been displayed;*

8. *H records componentwise* hsame *transport for enumeration rows, prefix windows, diagonal bits, disagreement ledgers, tree rows, fan rows, Real-facing routes, replay rows, provenance, and local naming;*
9. *C records displayed* Cont *replay from an enumeration window to diagonal-bit construction, disagreement verification, Cantor-space handoff, and Real-facing non-surjection routing;*
10. *P is the* Pkg *provenance over* ConstructiveCantorTheorem$^{\uparrow}$, CantorSpace$^{\uparrow}$, KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, RealDiagonalNonSurjection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, W, D, G, T, F, R, H, C, P, N$. It has no coordinate for a completed enumeration of all binary streams, host powersets, quotient stream equality, countable choice, or classical cardinal comparison.

Theorem 4706.2 (Finite diagonal obligations)**.** *For every accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier $K = (E, W, D, G, T, F, R, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *surface consists exactly of finite enumeration-window admission through E, binary observation through W, diagonal-bit formation through D, inspected-index disagreement through G, tree and fan routing through T, F, Real-facing non-surjection routing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5879.1. The enumeration and observation rows are E and W, so every candidate read is confined to a displayed finite prefix window.

Build the diagonal row by induction on the finite index spine of W. The empty window contributes no bit and no disagreement row. At a successor index, read the displayed candidate bit at the diagonal coordinate and place the opposite bit in D.

The same induction fills G. Each successor step adds exactly the disagreement between the new diagonal bit and the candidate bit just inspected, while previous rows are transported componentwise by H and replayed by C.

The rows T, F, R do not add an infinite object. They only route the same finite disagreement ledger through the Kleene-tree, fan-window, and Real-facing non-surjection surfaces, so the listed rows exhaust the certificate without a host powerset, quotient stream equality, choice principle, or classical cardinal comparison. □

Theorem 4706.3 (No-enumeration window)**.** *Let K be an accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier with a displayed enumeration window E of length n. The diagonal row D determines a Cantor-space prefix window such that for every inspected index $i < n$, the ith candidate of E differs from D at the displayed diagonal coordinate recorded in G. Hence the window cannot certify that its listed candidates exhaust the displayed Cantor-space prefixes.*

Proof. Induct on the finite enumeration window. For the empty window, there is no inspected candidate and the exhaustion claim has no displayed row.

At a successor window, the induction hypothesis supplies the disagreement rows for the previous candidates. The new candidate has a displayed diagonal coordinate in W, and the diagonal construction places the opposite binary value in D.

The new row of G records this disagreement. Componentwise hsame transport preserves the previous rows and the displayed Cont replay keeps the construction inside the same finite packet.

Therefore every inspected candidate is separated from the diagonal prefix at its own displayed coordinate. A finite window containing these disagreement rows cannot also certify that the same listed candidates exhaust the displayed Cantor-space prefix surface. □

Theorem 4706.4 (Cantor-space handoff)**.** *Every downstream consumer of a* ConstructiveCantorTheorem$^{\uparrow}$ *packet receives only the finite handoff formed by the enumeration window, binary observation window, diagonal-bit row, disagreement ledger, Kleene-tree row, fan-window row, Real-facing non-surjection route, transport, replay, provenance, and local naming. The handoff may be read by* CantorSpace$^{\uparrow}$ *and then by* RealDiagonalNonSurjection$^{\uparrow}$*, but it exports no completed stream quotient, powerset object, countable-choice selector, or classical uncountability principle.*

Proof. Start with the finite rows E, W, D, G of the carrier. By Definition 5879.2, these are the only rows that form and verify the diagonal disagreement.

Next read the Cantor-space side through chapter 1475. The Kleene-tree and fan rows T, F preserve finite prefix extension and bounded-window routing; they do not select a completed path or add an ambient compactness theorem.

Finally route the same disagreement ledger to chapter 3793 through R. Since the carrier has no coordinate for quotient stream equality, countable choice, host powersets, or classical cardinal comparison, the consumer receives exactly the finite handoff named in the statement. □

Falsifiable prediction: A purported ConstructiveCantorTheorem$^{\uparrow}$ consumer that reads a no-enumeration conclusion while omitting the enumeration window, diagonal-bit row, inspected-index disagreement ledger, Cantor-space prefix row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cantorspace, kleene-tree, fantheorem, real-diagonal-non-surjection: ConstructiveCantorT is not bijective to any listed sibling because it jointly carries an attempted enumeration window, a diagonal-bit construction, an inspected-index disagreement ledger, and a downstream Cantor-space-to-Real non-surjection route in one finite NameCert packet.

Closure status for NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 5879.1, Definition 5879.2, Definition 5879.3, and Definition 5879.4 seed the finite diagonal non-enumeration packet over Cantor-space prefix rows.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, a host powerset object, a countable-choice selector, a classical cardinal comparison, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean targets for carrier admission, diagonal-bit exactness, inspected-index disagreement, Cantor-space handoff, Real-facing route preservation, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCantorTheoremUp.

Constructive story: A ConstructiveCantorTheorem$^{\uparrow}$ packet is generated from a finite attempted enumeration window, a finite binary observation window, a diagonal-bit row, an inspected-index disagreement ledger, Kleene-tree and fan-window rows, Real-facing non-surjection routing, componentwise transport, displayed replay, provenance, and a local NameCert row.

MarkovKakutaniFixedPoint$^{\uparrow}$ routes its common fixed-window packet through the compact-convex and set-valued fixed-point surfaces of chapter 1923 and chapter 3511.

Chapter 4707

Markov-Kakutani Fixed Point NameCert Construction

Following Markov and Kakutani, the $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet records the finite BEDC surface for a commuting affine family acting on a compact convex carrier. It is anchored by the convex carrier of chapter 208, the functional-analysis ambient rows of chapter 94, the compact-convex self-map handoff of chapter 1923, and the set-valued fixed-point surface of chapter 3511. The packet names a finite common fixed-point obligation route; it does not import a host compactness theorem, a global choice selector, an ambient locally convex space, or a completed real vector topology outside the displayed rows.

Definition 4707.1 (Markov-Kakutani fixed point carrier)**.** *A* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, A, F, C, W, I, H, R, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the carried* $\mathsf{ConvexSet}^{\uparrow}$ *and compact support surfaces.*
2. *A is the affine ambient row, exposing only the finite vector-operation and scalar-stability data required by the displayed family.*
3. *F is the commuting-family row, listing the finite subfamily of affine self-maps and their source and target admissions.*
4. *C is the commutation ledger, recording pairwise commutation only for the displayed finite family windows.*
5. *W is the finite fixed-window row, recording the common approximate fixed rows inspected by downstream consumers.*
6. *I is the invariant-subcarrier row, exposing the finite intersection handoff used to keep the common fixed-window route inside the compact convex carrier.*
7. *H records componentwise* hsame *transport for domain, affine ambient, family, commutation, fixed-window, invariant-subcarrier, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a family request through $D, A, F, C, W, I, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, A, F, C, W, I, H, R, P, N$. It has no coordinate for an unbounded operator family, host compactness, selected fixed point, quotient equality of nets, or an ambient topological-vector-space theorem.

Theorem 4707.2 (Markov-Kakutani fixed point obligation surface)**.** *For every accepted* MarkovKakutaniFixedPoint *carrier*

$$M = (D, A, F, C, W, I, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of compact-convex domain admission through D, affine ambient admission through A, finite commuting-family admission through F, pairwise commutation exposure through C, fixed-window exposure through W, invariant-subcarrier handoff through I, componentwise* hsame *stability through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5880.1. The compact-convex, affine, commuting-family, commutation, fixed-window, and invariant-subcarrier data are exactly the rows D, A, F, C, W, I.

The domain and affine rows bind the finite ambient context before any map in the family is read. The family row F then lists only displayed affine self-maps, and the commutation ledger C records pairwise commutation inside that finite list.

The fixed-window and invariant-subcarrier rows W, I are the only route to a common fixed-point obligation. Finally H, R, P, N transport, replay, source, and name the same packet. Since no additional coordinate is present, the listed rows exhaust the NameCert surface. □

Theorem 4707.3 (Markov-Kakutani commuting handoff)**.** *Every accepted consumer of a* MarkovKakutaniFixedPoi *packet that asks for a common fixed-window read factors through*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{SchauderFixedPoint}^{\uparrow},\ \mathsf{KakutaniFixedPoint}^{\uparrow} \longmapsto D, A, F, C, W, I, H, R, P, N.$$

The exported object is a finite commuting-family fixed-window packet. It is not a selected fixed point, a host compactness theorem, a global choice principle, a quotient of approximating nets, or an ambient locally convex-space completion.

Proof. Open the accepted carrier using Definition 5880.1. A consumer must first read the compact convex domain row D and the affine ambient row A, so the route is grounded in the displayed convex and functional-analysis siblings.

Next read the finite family row F and the commutation ledger C. These rows expose only the listed affine self-maps and their pairwise commutation windows, so no unbounded operator family or host selection principle enters the packet.

The fixed-window row W is then interpreted inside the invariant-subcarrier row I. The Schauder and Kakutani siblings provide fixed-point handoff surfaces for the displayed compact-convex route, but the Markov-Kakutani packet exports only the finite common-window ledger named by W, I.

Finally H, R, P, N preserve the same carrier under transport, replay, provenance, and local naming. The exclusion clauses in the carrier therefore prevent selected fixed points, host compactness, quotient-net equality, and ambient completion from leaving the packet. □

Falsifiable prediction: Every accepted $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ consumer must expose compact-convex domain, affine ambient, finite commuting-family, pairwise commutation, fixed-window,

invariant-subcarrier, hsame, Cont, Pkg, and NameCert rows before it can read a common fixed-window obligation.

Independence witness: convexset, functionalanalysis, schauder-fixed-point, kakutani-fixed-point: $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite commuting affine family, commutation ledger, fixed-window rows, invariant-subcarrier handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet is generated from compact-convex domain rows, affine ambient rows, finite commuting-family rows, pairwise commutation ledgers, fixed-window rows, invariant-subcarrier handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5880.1, Definition 5880.2, and Definition 5880.3 seed the finite Markov-Kakutani fixed-point packet over displayed compact-convex, affine, commuting-family, fixed-window, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected common fixed points, global choice, quotient equality of nets, ambient locally convex completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex domain admission, affine-family routing, commutation ledger coverage, fixed-window exposure, invariant-subcarrier handoff, transport, replay, provenance, and local naming for BEDC.Derived.MarkovKakutaniFixedPointUp.

Proposition 4707.4 (Markov-Kakutani fixed-window dependency route)**.** *Every accepted* MarkovKakutaniFixedPoi *consumer that reads the common fixed-window row W consumes* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *from chapter 3511 and* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *from chapter 1923 only through the displayed compact-convex fixed-window route $D, A, F, C, W, I, H, R, P, N$ of Definition 5880.3. The route records the horizontal fixed-point dependency while refusing a detached host compactness theorem, selected common fixed point, or ambient locally convex completion.*

$\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ routes its private/shared row-set ledger to the defect-refined decoration consumer of chapter 7630, while the child files keep the NameCert carrier and forward route separate.

Chapter 4708

Window-Six Private/Shared Decoration NameCert Construction

The WindowSixPrivateSharedDecoration$^{\uparrow}$ packet realizes the private/shared decoration enumeration of Definition 9462.1699 inside the DNA window-six boundary-tile source of subsection 9462.28.21. It consumes the boundary-tile packet of chapter 3393, the six-boundary coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the trigger-threshold packet of chapter 4217. The packet records private blocks, shared row sets, trigger coverage, decoration counts, transport, replay, provenance, and local naming as finite rows; it is not a biological mechanism, stochastic fitness model, hidden enumeration oracle, or quotient codon semantics.

Definition 4708.1 (Window-six private/shared decoration carrier)**.** *A* WindowSixPrivateSharedDecoration$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, T, E, D, H, C, P, N)$$

with the following displayed rows:

1. *B is the private-block row, recording the nonempty private coordinate blocks attached to each accepted decorated support.*
2. *S is the shared-row-set ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *T is the trigger-coverage row, read through the carried DNA boundary-tile, six-boundary coverage, closure, and threshold packets.*
4. *E is the decoration-count row, recording the finite enumeration factors from the source private/shared theorem.*
5. *D is the defect ledger, recording row-set-size contribution to the defect-refined decoration polynomial.*
6. *H records componentwise* hsame *transport for private blocks, shared row sets, trigger coverage, decoration count, defect ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a window-six decoration request through B, S, T, E, D, H, P, N.*
8. *P is the* Pkg *provenance over* WindowSixPrivateSharedDecoration$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, DnaBoundaryTriggerClosure$^{\uparrow}$, DNAWindowSixTriggerThreshold$^{\uparrow}$, *the source theorem of Definition 9462.1699,* BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, S, T, E, D, H, C, P, N$. It has no coordinate for biological mechanism, stochastic evolutionary semantics, quotient codon equality, hidden global enumeration, or an unrecorded trigger oracle.

Theorem 4708.2 (Window-six private/shared decoration obligations)**.** *For every accepted* WindowSixPrivateShared *carrier*

$$W = (B, S, T, E, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of private-block admission through B, shared-row-set admission through S, trigger-coverage exposure through T, decoration-count exposure through E, defect-ledger exposure through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5881.1. The private block, shared row set, trigger coverage, decoration count, defect ledger, transport, replay, provenance, and naming data are precisely $B, S, T, E, D, H, C, P, N$.

The rows B, S enforce the private/shared split: each accepted support has a displayed private block, and each shared coordinate is assigned to a row set of size at least two. The trigger-coverage row T then ties those rows to the existing window-six boundary, coverage, closure, and threshold packets.

The rows E, D record the enumeration and defect data used by the source theorem. Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4708.3 (Window-six private/shared decoration enumeration handoff)**.** *Every accepted* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *consumer that reads a private/shared window-six decoration factors through*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

The exported object is a finite private/shared row-set ledger with trigger-coverage and decoration-count rows. It is not a biological model, a quotient codon space, a hidden enumeration oracle, or an unrestricted trigger-search theorem.

Proof. Read the accepted carrier through Definition 5881.1. The private and shared ledgers B, S supply the displayed row-set split before a decoration count is exposed.

Next use T to route the split through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings. This prevents a decoration request from bypassing the window-six trigger-coverage surface.

The enumeration row E and defect row D then record the finite count and defect-polynomial data from the source theorem. The structural rows H, C, P, N transport, replay, source, and locally name the same rows.

Therefore every accepted consumer receives the finite packet described in the theorem, rather than biological semantics, quotient codon equality, hidden enumeration, or unrestricted trigger search. □

Theorem 4708.4 (Window-six private/shared decoration sibling lattice)**.** *Every accepted* WindowSixPrivateSharedD *packet consumes the sibling lattice*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

before it exports the private/shared decoration ledger $B, S, T, E, D, H, C, P, N$. The lattice is finite and row-projection based: it displays boundary tiles, six-boundary coverage, trigger closure, and trigger-threshold evidence before the private block, shared row-set, trigger coverage, decoration count, and defect rows are consumed. It is not a biological mechanism, quotient codon semantics, stochastic evolutionary model, or hidden enumeration oracle.

Proof. Project the accepted carrier $W = (B, S, T, E, D, H, C, P, N)$ of Definition 5881.1. The private/shared ledger rows B, S, the trigger row T, the enumeration and defect rows E, D, and the structural rows H, C, P, N are the only exported coordinates.

Apply the enumeration handoff of Definition 5881.3. It forces every private/shared decoration read to pass through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings before $B, S, T, E, D, H, C, P, N$ are consumed.

Thus the dependency is a finite sibling lattice: each sibling supplies a displayed row that is projected before the decoration ledger is exported. Since the carrier has no coordinate for biological mechanism, quotient codon equality, stochastic evolution, or hidden enumeration, none of those interpretations is introduced by the lattice. □

Falsifiable prediction: Every accepted $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet must expose private-block, shared-row-set, trigger-coverage, decoration-count, hsame, Cont, Pkg, and NameCert rows before it can export a private/shared window-six decoration read.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-window-six-trigger-threshold: $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ is not bijective to any listed sibling because the private/shared row-set ledger jointly carries private blocks, shared row sets, trigger coverage, decoration counts, defect rows, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.WindowSixPrivateSharedDecorationUp.TasteGate` must provide explicit Nontrivial $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ and FieldFaithful WindowSixPrivateSharedDecora instances preserving $B, S, T, E, D, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet is generated from private-block rows, shared-row-set ledgers, trigger-coverage rows, decoration-count rows, defect ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$, seed)

Scope closed: Definition 5881.1, Definition 5881.2, and Definition 5881.3 seed the finite private/shared window-six decoration packet over displayed boundary, coverage, closure, threshold, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block exposure, shared-row-set arity, trigger-coverage routing, decoration-count exposure, defect-ledger exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.WindowSixPrivateSharedDecorationUp.

Proposition 4708.5 (Window-six private/shared decoration defect forward route)**.** *The private/shared row-set ledger exported by* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *is an allowed child route for chapter 7630. Its rows $B, S, T, E, D, H, C, P, N$ supply the private block, shared row set, trigger coverage, enumeration, defect ledger, transport, replay, provenance, and local naming data used by the downstream defect-refined decoration polynomial route.*

Proof. Read the exported packet through Definition 5881.1. The private/shared ledger already separates private blocks from shared row sets and records the defect ledger as a displayed row.

The downstream chapter chapter 7630 consumes exactly that displayed ledger when it forms its defect-refined decoration polynomial route. Because the route is through the named child rows rather than through the hub file, it preserves the hub-only layout and records a genuine forward dependency between the two finite packets. □

Chapter 4709

A Concrete Naming Certificate for $\mathsf{SchurLemma}^{\uparrow}$

Following Schur and the finite-group representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{SchurLemma}^{\uparrow}$ packet records the finite Hom-intertwiner boundary for irreducible representation rows. It sits between the finite representation carrier of chapter 3662, the character-theory trace surface of chapter 245, the representation-ring package of chapter 246, the orthogonality route of chapter 1603, and the character-table consumer of chapter 3574.

Definition 4709.1 (SchurLemma carrier)**.** *A* $\mathsf{SchurLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, H, S, T, R, P, N)$$

with the following displayed rows:

1. *U is an irreducible finite representation row over the same finite group source used by Definition 3662.1;*
2. *V is a second irreducible finite representation row over the same finite group source;*
3. *H is the displayed* Hom *and intertwiner row from U to V, recording equivariance rather than all host linear maps;*
4. *S is the scalar-or-zero boundary: when the two irreducible rows are the same finite representation row, accepted endomorphism reads factor through one scalar row, and when they are distinct no nonzero intertwiner row is exported;*
5. *T records componentwise* hsame *transport for the representation rows, Hom row, scalar-or-zero boundary, route, provenance, and naming rows;*
6. *R records displayed* Cont *replay from the finite representation packet through the Hom row into the scalar-or-zero boundary;*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *row.*

The carrier is exactly this finite representation-theory packet. It has no coordinate for a host abelian category, unrestricted module category, global algebra closure, or unrecorded linear-map equality.

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaTasteGate single carrier alignment`

Theorem 4709.2 (SchurLemma NameCert obligations)**.** *For the carrier of Definition 5882.1, the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (U, V, H, S, T, R, P, N)$*, irreducible finite representation exposure through* U, V*, Hom-intertwiner exposure through* H*, scalar-or-zero boundary exposure through* S*, componentwise* hsame *transport through* T*, displayed* Cont *replay through* R*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The packet lists U, V, H, S, T, R, P, N as its whole public surface, so the local certificate cannot introduce a second Hom object, a host representation category, or a hidden linear-map equality relation.

Read the representation rows first. The rows U and V are finite irreducible representation rows of the same group source, and the Hom row H is accepted only as an equivariant row between them.

The scalar-or-zero boundary is then read through S. It records scalar endomorphism reads in the same-row case and zero Hom reads in the distinct-row case. The rows T, R, P, N transport, replay, package, and name only that finite packet. □

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaCarrier namecert obligations`

Theorem 4709.3 (SchurLemma character-table handoff)**.** *Every accepted* $\mathsf{SchurLemma}^{\uparrow}$ *packet exports to* $\mathsf{CharacterTable}^{\uparrow}$ *only the finite Hom-intertwiner boundary needed by the Schur-orthogonality and character-pairing routes. The handoff reads* U, V, H, S *before* T, R, P, N *and exports no host representation category, unrestricted matrix algebra, quotient computation, or globally chosen basis.*

Proof. Start from Definition 5882.1. The downstream character-table consumer sees exactly the two irreducible representation rows, the displayed Hom-intertwiner row, and the scalar-or-zero boundary.

Use the neighboring representation-theory packets as routing anchors. The finite representation chapter supplies the representation rows, the character-theory and representation-ring chapters supply trace and sum surfaces, and the Schur-orthogonality chapter consumes the scalar-or-zero boundary before the character-table chapter reads pairings.

The structural rows T, R, P, N preserve that same finite packet under transport, replay, provenance, and local naming. Since no host category, unrestricted matrix algebra, quotient computation, or chosen-basis row appears in the carrier, the character-table handoff cannot export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurLemma}^{\uparrow}$ packet is generated from two irreducible finite representation rows, a displayed Hom-intertwiner row, a scalar-or-zero boundary row, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, seed)

Scope closed: Definition 5882.1, Definition 5882.2, and Definition 5882.3 seed the finite Hom-intertwiner boundary between irreducible representation rows and the character-table consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host representation category, unrestricted module-category Hom, quotient computation, globally chosen bases, Lean verification, or bridge checking for $\mathsf{SchurLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, irreducible-row exposure, Hom-intertwiner exactness, scalar-or-zero boundary, transport stability, replay stability, and local NameCert nonescape for BEDC.Derived.SchurLemmaUp.

Chapter 4710

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet records the interval-minimum half of the compact-real extreme-value route as finite BEDC NameCert data. It sits over the compact interval source of chapter 3986, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located lower-bound route dual to chapter 1427, and the terminal real-name seal of chapter 38. The packet is minimum-specific: it exposes lower-bound attainment windows without importing open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4710.1 (Compact real interval minimum carrier)**.** *A* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite endpoint windows, and real endpoint seals.*
2. *K is the compact interval row, read through* $\mathsf{BishopCompactInterval}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$*, exposing only the finite inspection surface admitted for the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact interval.*
4. *L is the lower-bound and located-supremum-dual row, recording that every inspected value is above a displayed lower-bound candidate and that the candidate is reached through finite located windows rather than by trichotomy.*
5. *W is the finite witness-window row, recording the interval windows in which F and L meet a terminal value seal.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the value and lower-bound comparison windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *value seal row reached only after I, K, F, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, lower-bound row, witness windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from minimum-attainment requests through* $I, K, F, L, W, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, arbitrary real trichotomy, quotient real equality, a selected host minimum, a selected host maximum, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactRealIntervalMinimumUp.CompactRealIntervalMinimumTasteGate single carrier alignment`

Theorem 4710.2 (Compact real interval minimum NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier*

$$M = (I, K, F, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact interval source, continuous-map row, lower-bound and located-supremum-dual row, finite witness-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *value seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No minimum obligation is stored outside these rows.*

Proof. Project the carrier of Definition 5883.1. The interval row I and compact row K expose the only compact-real source windows available to the packet, while F records the continuous real-valued map inspected on those windows.

Read the lower-bound and witness rows next. The row L gives the displayed lower-bound comparison as the dual of the located-supremum route, and W binds that comparison to finite source windows where the map row and bound row meet.

The regular and real-facing rows are downstream. The row R records the regular rational readback of the same finite comparisons, while E seals only those displayed reads as $\mathsf{Real}^{\uparrow}$ values.

The remaining rows H, C, P, N transport, replay, source, and locally name the same route. Since the carrier has no coordinate for open-cover compactness, trichotomy, quotient equality, selected host extrema, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 4710.3 (Compact real interval minimum located attainment)**.** *Every accepted* CompactRealIntervalMinim *consumer that reads a minimum value factors through the displayed route*

$$I, K \longmapsto F, L \longmapsto W \longmapsto R, E \longmapsto H, C, P, N.$$

Thus the compact-real interval-minimum surface exports only a finite witness-window route from a closed bounded compact interval and continuous-map row to a located lower-bound row, regular rational readback, and $\mathsf{Real}^{\uparrow}$ *value seal. It does not export open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact interval source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real interval windows.

Feed the source into the continuous-map and lower-bound rows. The row F exposes only displayed value reads, while L records the lower-bound comparison through the located-supremum-dual route rather than through a host order decision.

Use the witness-window and readback rows together. The row W records where the compact interval inspection meets the lower-bound comparison, and R repacks those comparisons as $\mathsf{RegSeqRat}^{\uparrow}$ data before the $\mathsf{Real}^{\uparrow}$ seal E is read.

Finally, transport and replay preserve the route through H and C, while P and N provide provenance and local naming. No row supplies open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient completeness, so every accepted minimum read remains inside the displayed route. □

Falsifiable prediction: Every $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ minimum read factors through ClosedBoundedInterva $\mathsf{BishopCompactInterval}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$-dual lower-bound data, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before reaching finite witness windows and the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ row.

Independence witness: bishopcompactinterval, closedboundedinterval, continuousmap, locatedsupremum, real: $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact interval source, continuous-map row, located lower-bound route, finite witness-window row, regular readback, Real value seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a BishopCompactIntervalUp or CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, a lower-bound row dual to LocatedSupremumUp, finite witness windows, RegSeqRatUp readbacks, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5883.1, Definition 5883.2, and Definition 5883.3 seed the finite compact-real interval-minimum route over interval, compact-source, continuous-map, lower-bound, witness-window, regular readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient real equality, selected host extrema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, lower-bound routing, regular readback, Real value sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalMinir

Chapter 4711

A Concrete Naming Certificate for $\mathsf{SimpleFunctionApproximation}^{\uparrow}$

Following Caratheodory and Bourbaki measurable approximation, the $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet names the finite-range measurable surface by which a measurable real-facing map is approximated without importing completed measure theory or host function equality. It sits after the preimage-ledger map surface of chapter 3461, the event-value rows of chapter 92, the generated-set boundary of chapter 3384, and the terminal real-name seal of chapter 38. The packet records finite range cells, measurable preimage rows, displayed tolerance data, finite observations, transport, replay, provenance, and local naming.

Definition 4711.1 (Simple function approximation carrier)**.** *A* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, R, E, T, O, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row, exposing only the measurable event surface and event-value readback needed by the approximation packet.*
2. *F is the input* $\mathsf{MeasurableFunction}^{\uparrow}$ *row, read through its displayed target-event and preimage ledger rather than through a host function object.*
3. *R is the finite range ledger, listing the displayed* $\mathsf{Real}^{\uparrow}$ *values that the simple approximation may return.*
4. *E is the finite cell ledger, assigning each range row in R to a measurable source cell whose admission is read through* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*, and the preimage rows of F.*
5. *T is the displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, naming the approximation budget used by the current consumer.*
6. *O is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the visible source probes, range reads, and tolerance comparisons exposed to the packet.*
7. *H stores componentwise* hsame *transport for the measure source, measurable-function row, finite range ledger, cell ledger, tolerance row, observation row, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay from source-event inspection through finite cell selection, range-value readback, tolerance comparison, and observation replay.*

9. *P is the* Pkg *provenance over* SimpleFunctionApproximation$^{\uparrow}$, MeasurableFunction$^{\uparrow}$, Measure$^{\uparrow}$, BorelSet$^{\uparrow}$, Real$^{\uparrow}$, StreamName$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{SimpleFunctionApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, E, T, O, H, C, P, N$. It has no coordinate for almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, or equality of host functions.

Lean4 ✓ `BEDC.Derived.SimpleFunctionApproximationUp.SimpleFunctionApproximationTasteGate single carrier alignment`

Theorem 4711.2 (Simple function approximation NameCert obligations)**.** *For the carrier of Definition 5884.1, the local* NameCert$_{\text{SimpleFunctionApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure source, measurable-function row, finite range ledger, measurable cell ledger, displayed* Real$^{\uparrow}$ *tolerance row, finite* StreamName$^{\uparrow}$ *observations, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted approximation consumer must read the finite cells through E, the measurable preimage route through F, the range values through R, and the tolerance comparison through T before any sibling* Measure$^{\uparrow}$, BorelSet$^{\uparrow}$, Real$^{\uparrow}$, *or* StreamName$^{\uparrow}$ *row is exported.*

Proof. Project the carrier $S = (M, F, R, E, T, O, H, C, P, N)$. The measure source M and measurable-function row F are the only rows from which measurable event and preimage information can be read.

Read the finite approximation data. The range ledger R supplies the displayed real values, the cell ledger E ties each range value to a measurable source cell, and the tolerance row T records the visible approximation budget against which the current observations are compared.

Replay the observation and structural rows. The row O records the finite StreamName probes, while H, C, P, N transport, replay, source, and name the same route through the sibling certificates. Since no further coordinate is present, the listed rows exhaust the NameCert obligation surface and cannot export quotient classes, completed measures, selected representatives, ambient integration theorems, or host function equality. □

Theorem 4711.3 (Simple function approximation finite-cell route)**.** *Every accepted* SimpleFunctionApproximation$^{\uparrow}$ *read factors through the finite route*

$$M, F \longmapsto E \longmapsto R, T \longmapsto O \longmapsto H, C, P, N.$$

Thus a consumer receives a finite measurable partition-by-cells with displayed real values and a displayed tolerance budget, not an almost-everywhere quotient class, completed measure, selected representative, ambient integral, or equality proof for host functions.

Proof. Start with an accepted approximation request. The request first exposes the measure source and the measurable-function row, so every cell read is grounded in the displayed source-event and preimage ledgers.

The cell ledger E is finite. Each displayed cell is paired with a range value from R, and the tolerance row T supplies the visible real-valued budget for the comparison requested by the consumer.

Finally replay the observations and structural rows. The finite StreamName row O records only the exposed probes, while H, C, P, N transport and name the route. No hidden coordinate remains from which a quotient, completion, representative selection, integration theorem, or host equality could be recovered. □

Falsifiable prediction: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ consumer that obtains an almost-everywhere quotient class, completed measure, selected representative, ambient integration theorem, or host function equality without first reading $M, F, R, E, T, O, H, C, P, N$ refutes the carrier of Definition 5884.1.

Independence witness: measurablefunction, measure, borelset, real, streamname: no listed sibling supplies the whole ten-row packet $S = (M, F, R, E, T, O, H, C, P, N)$, because each sibling contributes only preimage-ledger, event-value, generated-set, real-name, or finite-observation data while this chapter binds the finite-range approximation surface with tolerance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet is generated from a measure source, a measurable-function preimage ledger, finite range cells, a displayed Real tolerance row, finite StreamName observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5884.1, Definition 5884.2, and Definition 5884.3 seed the finite-range measurable approximation surface over source events, preimage rows, cell ledgers, range values, tolerance rows, observations, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell exactness, measurable preimage preservation, Real tolerance replay, StreamName observation stability, and quotient-refusal proofs for BEDC.Derived.SimpleFunctionApproxima

Chapter 4712

A Concrete Naming Certificate for $\mathsf{DyadicModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicModulus}^{\uparrow}$ packet records the finite tolerance-modulus surface by which dyadic precision requests become regular rational readback before the terminal real seal is consumed. It sits between the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the terminal seal boundary of chapter 38, and the Cauchy-modulus route carried by $\mathsf{CauchyModulus}^{\uparrow}$. The packet is a finite NameCert carrier for Real and Completion consumers; it is not a host modulus function, quotient stream equality, selected limit, countable choice, or ambient completeness principle.

Definition 4712.1 (DyadicModulus carrier)**.** *A* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic scales, finite comparison budgets, rounding windows, and precision requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose visible entries are exactly the observations demanded by the tolerance row.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by reading the stream windows under the dyadic tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, available only after D, S, R have been displayed.*
5. *H records componentwise* hsame *transport for tolerance, stream-window, regular-readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each dyadic precision request through D, S, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, ambient Real completeness, or a detached host-level modulus function.

Lean4 ✓ `BEDC.Derived.DyadicModulusUp.DyadicModulusTasteGate single carrier alignment`

Theorem 4712.2 (DyadicModulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier* $M = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window exposure, regular rational readback, terminal Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every Real or Completion consumer must read the displayed route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before using the packet as a dyadic tolerance-modulus certificate.

Proof. Project the carrier of Definition 5885.1. The tuple displays exactly the dyadic tolerance row, stream-window row, regular-readback row, terminal seal, transport, replay, provenance, and local name.

The analytic route is forced in order. A precision request first reads D, then the finite stream observations in S, then the regular rational handoff R, and only then reaches the seal boundary E.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or detached-modulus coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4712.3 (DyadicModulus RegSeqRat handoff)**.** *Let* $M = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion-facing consumer of the modulus row must factor through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before the terminal seal row* E *is read.*

The handoff route is

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as rows of one BHist *packet, with transport, replay, provenance, and local naming supplied by* H, C, P, N*.*

Proof. Start from Definition 5885.2. It already places every tolerance-modulus consumer on the displayed route D, S, R, E, H, C, P, N.

The regular rational readback is the row R, and R is downstream of both the dyadic tolerance row D and the stream-window row S. Thus a consumer cannot read a regular name, Real seal, or Completion boundary before the finite tolerance and window data have been displayed.

Finally inspect the excluded coordinates of Definition 5885.1. Because no quotient stream, selected representative, host equality, countable-choice schedule, ambient completeness theorem, or detached modulus function is present, the RegSeqRat handoff is exactly the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{DyadicModulus}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or a Completion-facing seal without first displaying one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-modulus: $\mathsf{DyadicModulus}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide tolerance arithmetic,

finite windows, regular rational readback, terminal sealing, or threshold scheduling, while this packet binds them into one finite tolerance-modulus handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}$.

Constructive story: A $\mathsf{DyadicModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat handoff row, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5885.1, Definition 5885.2, and Definition 5885.3 seed the finite dyadic tolerance-modulus carrier between dyadic precision data and regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, structural transport, and non-escape for BEDC.Derived.DyadicModulusUp.

Chapter 4713

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 4713.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 4713.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 4713.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 4713.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The $\mathsf{Cauchy}^{\uparrow}$ packet is consumed through chapter 301 and chapter 38 before downstream completion or calculus reads inspect the finite Cauchy windows.

Chapter 4714

A Concrete Naming Certificate for $\mathsf{CauchyModulusSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet records a carrier-space interface for explicit Cauchy modulus schedules. It is anchored in the single-modulus certificate of chapter 292, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the fast Cauchy sequence route of chapter 2185. The chapter treats a family of modulus schedules as finite naming data, not as a quotient of real sequences or as an ambient completeness theorem.

Definition 4714.1 (Cauchy modulus space carrier)**.** *A* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row for the family of explicit modulus schedules.*
2. *S assigns to each displayed index a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, including the monotone precision requests used by the family.*
3. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule family.*
4. *D is the dyadic tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
6. *Q records the* $\mathsf{Real}^{\uparrow}$ *seal boundaries at which a scheduled read may leave rational and dyadic data for a real-facing consumer.*
7. *H records componentwise* hsame *transport for $I, S, W, D, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay from an index and precision request through the schedule row, stream window, dyadic tolerance ledger, regular rational readback, and real seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSpace}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the displayed rows componentwise by hsame*. It has no coordinate for an unscheduled Cauchy oracle, quotient equality of sequences, a selected real limit, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSpaceUp.CauchyModulusSpaceTasteGate single carrier alignment`

Theorem 4714.2 (Cauchy modulus space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index family, schedule admission for the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission, dyadic tolerance compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal-boundary compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming for the packet.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceNameCertObligations`

Proof. Project the accepted carrier of Definition 5887.1. The analytic data are precisely I, S, W, D, R, Q, and the structural rows are precisely H, C, P, N.

Schedule admission is rowwise over I. Each index exposes its explicit modulus schedule before any stream window is inspected, so the packet cannot replace the family by an unscheduled oracle.

The tolerance and readback rows are then forced through D and R. The real boundary row Q is available only after those rational and dyadic ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4714.3 (Cauchy modulus space finite route)**.** *Every real-facing comparison of two accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carriers factors through finite schedule rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundaries, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *rows. The route exports a shared modulus-comparison surface for later diagonal* $\mathsf{Real}^{\uparrow}$ *completion arguments without introducing quotient sequence equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFiniteRoute`

Proof. Open both carriers using Definition 5887.1. A comparison can inspect only the finite index rows, schedule rows, stream windows, tolerance ledgers, readback rows, and seal boundaries.

Componentwise hsame in H transports the two packets to a common finite surface. Displayed Cont replay in C keeps each precision request attached to its schedule, stream window, dyadic tolerance, regular readback, and real seal boundary.

The provenance and naming rows P, N package that finite surface as the local certificate. Thus later diagonal completion routes can share modulus comparison data without receiving any quotient or choice-based object. □

Theorem 4714.4 (CauchyModulusSpace seed interface)**.** *Every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet exports its modulus schedule family to the L10 four-object seal route only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Cauchy}^{\uparrow}$ *root route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of this seed interface can expose quotient stream equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as part of the packet surface.*

Proof. Project the carrier of Definition 5887.1. The modulus family is displayed by I, S, and every inspected request is routed through the finite observation rows W, D, R, Q before the structural rows H, C, P, N can name or replay it.

Apply Definition 5887.3. Any real-facing comparison has already factored through finite schedule rows, stream windows, dyadic tolerances, regular rational readbacks, and the terminal Real seal boundary.

The $\mathsf{Cauchy}^{\uparrow}$ root route receives exactly that finite schedule surface. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or ambient completeness, the seed interface cannot export any of those host objects to L10 Real and Completion consumers. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceSeedInterface`

Theorem 4714.5 (CauchyModulusSpace tail composition)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

two displayed modulus schedules in the family compose only through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The composed tail read is the finite route*

$$I, S, W, D, R, Q \longmapsto H, C, P, N$$

and it exports no quotient sequence equality, selected limit, countable choice, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceTailComposition`

Proof. Project the carrier of Definition 5887.1. The two schedules are displayed rows under S indexed by I, and their inspected tails must pass through the same finite window row W.

Apply the finite route and seed interface, Definition 5887.3 and Definition 5887.4. These results force the tolerance ledger D, regular readback R, and Real seal boundary Q to remain visible before the composed tail can be consumed.

Finally replay the structural rows. Componentwise transport H, displayed continuation C, provenance P, and local naming N preserve the same finite schedule surface, so tail composition does not add quotient equality, selected limits, countable choice, or ambient completeness. □

Theorem 4714.6 (CauchyModulusSpace filter handoff)**.** *Every CauchyFilter or CauchyFilterCompletion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet receives the modulus family only through the finite handoff*

$$I, S, W, D, R, Q, H, C, P, N.$$

The handoff may be consumed as filter-facing Cauchy evidence, but it cannot be reinterpreted as quotient sequence equality, a selected limit, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFilterHandoff`

Proof. Begin with a filter-facing read. The seed interface of Definition 5887.4 already routes every scheduled request through finite stream windows, dyadic tolerances, regular rational readback, the Cauchy root route, and the terminal Real seal.

Use tail composition from Definition 5887.5. It shows that composing displayed schedules preserves the same finite route through I, S, W, D, R, Q before structural replay.

The filter consumer therefore receives only the transported and replayed packet named by H, C, P, N. Since the carrier contains no quotient equality coordinate, no selected limit, no

countable-choice schedule, and no ambient completeness row, CauchyFilter and CauchyFilterCompletion may consume only the displayed finite handoff. □

Theorem 4714.7 (CauchyModulusSpace diagonal-family exactness)**.** *A diagonal Real or Completion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet may compare schedule families only through the displayed rows*

$$I, S, W, D, R, Q, H, C, P, N.$$

The diagonal request must be routed through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary before any structural transport, replay, provenance, or local naming row can be consumed. The comparison exports no quotient stream equality, selected limit, countable-choice schedule, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceDiagonalFamilyExactness`

Proof. Start with the finite route of Definition 5887.3. It already factors every real-facing comparison of two accepted packets through the finite schedule rows, stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, and the structural rows H, C, P, N.

Apply the seed interface of Definition 5887.4. It binds every L10 request to the schedule family I, S before the finite windows W, tolerance ledger D, readback R, and seal boundary Q are exposed.

Finally use the filter handoff of Definition 5887.6. A diagonal completion consumer is still a consumer of the same finite handoff, so it receives only $I, S, W, D, R, Q, H, C, P, N$ and cannot add quotient equality, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ consumer that compares schedule families while omitting $I, S, W, D, R, Q, H, C, P, N$ refutes the carrier surface of Definition 5887.1.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, fast-cauchy-sequence: no listed sibling binds a family of explicit Cauchy modulus schedules together with finite stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, componentwise transport, replay, provenance, and local naming in one carrier-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet is generated from a finite family of explicit Cauchy modulus schedules, finite StreamName windows, dyadic tolerance ledgers, RegSeqRat readbacks, Real seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5887.1, Definition 5887.2, and Definition 5887.3 seed the carrier-space interface for explicit modulus schedule families and shared finite comparison routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close diagonal Real completion, quotient equality of Cauchy sequences, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-family exactness, dyadic and regular readback compatibility, Real-seal boundary nonescape, and a Lean carrier formalization for BEDC.Derived.CauchyModulusSpaceUp.

Chapter 4715

A Concrete Naming Certificate for $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the finite-limit language of regular category theory, the $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet records the boundary where a displayed Cauchy-completion route preserves the finite-limit rows that its source already exposes. It sits beside the Cauchy source route of chapter 3413, the metric-completion universal-property route of chapter 2124, the functorial completion route of chapter 423, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite left-exactness boundary for displayed completion data; it does not assert quotient completions, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4715.1 (Cauchy completion left-exactness carrier)**.** *A* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, U, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source finite-limit row, recording the displayed terminal, product, equalizer, or kernel-pair data available before completion is read.*
2. *U is the completion unit row, recording the source-to-completion unit reads for every object named by S.*
3. *D is the dense image row, keeping each completed finite-limit comparison tied to the same source rows and unit rows.*
4. *K is the kernel-pair preservation row, recording that equalizer and pullback comparisons are read through the displayed source kernel-pair data rather than through host quotient equality.*
5. *E is the extension ledger, recording the universal-property and functorial completion rows used to transport finite-limit witnesses into completion-facing reads.*
6. *H records componentwise* hsame *transport for S, U, D, K, E, C, P, N.*
7. *C records displayed* Cont *replay from a finite-limit request through S, U, D, K, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, selected finite limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionLeftExactnessUp.TasteGate.CauchyCompletionLeftExactnessTasteGate single carrier alignment`

Theorem 4715.2 (Cauchy completion left-exactness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the source finite-limit row, completion unit rows, dense image rows, kernel-pair preservation row, extension ledger, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every completion-facing finite-limit read must expose S, U, D, K before E, H, C, P, N can be consumed.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessNameCertObligations`

Proof. Project the carrier of Definition 5888.1. The source finite-limit row, unit row, dense image row, kernel-pair row, extension ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The finite-limit boundary is read through S, U, D, K. The source row supplies the terminal, product, equalizer, or kernel-pair data; the unit row embeds the displayed source names into completion-facing reads; the dense image row keeps those reads tied to source data; and the kernel-pair row confines exactness comparisons to displayed finite rows.

The remaining rows E, H, C, P, N transport and name the same route. Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient completeness have no carrier coordinate, projection of the displayed rows exhausts the NameCert obligations. □

Theorem 4715.3 (Cauchy completion left-exactness carrier admission)**.** *For every accepted* CauchyCompletionLeftE *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the source finite-limit row S, completion unit row U, dense image row D, kernel-pair preservation row K, extension ledger E, componentwise hsame *transport H, displayed* Cont *replay C, provenance row P, and local* NameCert *row N are all admitted as finite coordinates of the same packet. No completion-facing read may treat one of these rows as ambient host data detached from the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactness carrier admission`

Proof. Project the carrier of Definition 5888.1. Its coordinates are exactly $S, U, D, K, E, H, C, P, N$, so each named row is admitted by carrier membership.

Apply Definition 5888.2. The obligation surface requires every completion-facing finite-limit read to expose S, U, D, K before the extension, transport, replay, provenance, and naming rows can be consumed.

Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, none of the admitted rows can be reread as detached host data. □

Theorem 4715.4 (Cauchy completion left-exactness unit-density route)**.** *Every displayed finite-limit comparison read through an accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier factors first through the source finite-limit row* S*, then through the completion unit row* U*, and then through the dense image row* D *before the extension ledger* E *and structural rows* H, C, P, N *are inspected. This route records the finite source-to-completion passage without selecting representatives of completed objects or invoking countable choice.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessUnitDensityRoute`

Proof. Begin with the carrier admission theorem Definition 5888.3. It keeps S, U, D as visible coordinates of the same finite packet.

The source row S supplies the finite-limit data before completion is read. The unit row U records the displayed source-to-completion unit for each object named by S, and the dense image row D keeps the resulting completion-facing comparison tied back to the same source and unit rows.

Only after this route is exposed can E, H, C, P, N be consumed. The extension ledger and the structural rows transport, replay, package, and name the source-unit-density passage; they do not add selected representatives, countable choice, or host equality of completed objects. □

Theorem 4715.5 (Cauchy completion left-exactness kernel-pair preservation)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any equalizer, pullback, or kernel-pair comparison visible after completion is read through the displayed kernel-pair row* K *together with the source finite-limit row* S*, completion unit row* U*, and dense image row* D*. The comparison cannot be exported as quotient completion equality, a selected finite limit, or ambient equality of completed objects.*

Proof. Project the kernel-pair coordinate K from the carrier. By definition, K records equalizer and pullback comparisons through displayed source kernel-pair data rather than through a host quotient.

Use the unit-density route theorem Definition 5888.4. It places every completion-facing finite-limit comparison behind S, U, D, so the kernel-pair row is read as part of the same source-to-completion passage.

Finally apply Definition 5888.2. The rows E, H, C, P, N only extend, transport, replay, package, and name the displayed comparison. They add no coordinate for quotient completion equality, selected finite limits, or ambient equality of completed objects, so preservation remains confined to K and its source, unit, and dense-image context. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessKernelPairPreservation`

Theorem 4715.6 (Cauchy completion left-exactness regular-limit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any regular Cauchy limit read used by a finite-limit comparison factors through the source finite-limit row* S*, the completion unit row* U*, the dense image row* D*, the kernel-pair row* K*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger* E *is consumed. Thus the regular-limit comparison is a pullback of displayed source, unit, dense-image, kernel-pair, regular-readback, and real-seal rows, not a quotient completion or a selected representative of a completed object.*

Proof. Project the accepted carrier. The rows $S, U, D, K, E, H, C, P, N$ are the only admitted coordinates, so a finite-limit comparison cannot begin from a hidden completed object or a host-level limit.

Apply Definition 5888.4. It places every completion-facing finite-limit comparison behind the source finite-limit row, the completion unit row, and the dense image row before the extension ledger and structural rows are inspected.

Next apply Definition 5888.5. Equalizer, pullback, and kernel-pair comparisons remain confined to the displayed kernel-pair row together with the same source, unit, and dense-image context.

The regular Cauchy limit read is then exposed through the sibling $\mathsf{RegSeqRat}^{\uparrow}$ readback surface named in the chapter introduction, and the resulting endpoint is sealed only through the sibling $\mathsf{Real}^{\uparrow}$ boundary. Since those sibling rows are read after S, U, D, K and before E, the comparison has the announced displayed pullback route. No coordinate remains for quotient completion equality, selected representatives, selected finite limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegularLimitPullback`

Theorem 4715.7 (Cauchy completion left-exactness RegSeqRat unit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any finite-limit comparison whose completion unit route reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factors first through the source finite-limit row* S*, completion unit row* U*, dense image row* D*, and kernel-pair row* K*. It then reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger and structural rows* E, H, C, P, N *are consumed. Componentwise* hsame *transport preserves that order, so the exported comparison contains no quotient completion, selected representative, selected finite limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegSeqRatUnitPullback`

Proof. Start with the carrier admission theorem Definition 5888.3. It keeps $S, U, D, K, E, H, C, P, N$ as finite coordinates of one accepted packet, so the comparison cannot begin from an external completed object.

Apply the unit-density route theorem Definition 5888.4. This places every completion-facing finite-limit comparison behind S, U, D before the extension ledger and structural rows are inspected.

Next apply the kernel-pair preservation theorem Definition 5888.5. Pullback and equalizer comparisons remain confined to K together with the same source, unit, and dense-image context.

Finally use the regular-limit pullback theorem Definition 5888.6. Its readback route inserts the $\mathsf{RegSeqRat}^{\uparrow}$ surface and the terminal $\mathsf{Real}^{\uparrow}$ seal after S, U, D, K and before E, H, C, P, N. Componentwise hsame transport therefore preserves the displayed order, while the excluded quotient, selection, choice, host-equality, and ambient-completeness rows remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet is generated from source finite-limit rows, completion unit rows, dense image rows, kernel-pair preservation, an extension ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5888.1 and Definition 5888.2 seed the finite left-exactness boundary for displayed Bishop-style Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected finite limits, selected representatives, countable choice, host equality of completed objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source finite-limit exposure, unit-row exposure, dense image confinement, kernel-pair preservation, extension-ledger confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionLeftExactnessUp.

Theorem 4715.8 (Cauchy root UnaryHistory admission)**.** *Every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *consumer read admits a UnaryHistory-facing presentation before it enters any filter, modulus, or Real-facing route. The admitted packet is the finite row surface*

$$U = (S, Q, D, R, E, H, C, P, N),$$

where S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* Q *is the directed tail-request row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and* H, C, P, N *are componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootUnaryAdmission`

Proof. Project the root stream and tail-request carriers already displayed for $\mathsf{Cauchy}^{\uparrow}$. Their shared rows expose finite StreamName windows before any tolerance, readback, or terminal seal can be read.

The directed request row supplies the unary admission point. It records one visible later-window request over the same finite stream surface, so the read is a UnaryHistory-facing row rather than a parameter transport over an ambient sequence.

The dyadic, regular readback, Real seal, and structural rows then repack the same finite route. No quotient stream, selected limit, host equality, countable choice, or ambient completeness row is exported. □

Theorem 4715.9 (Cauchy root filter-modulus compatibility)**.** *For every admitted root packet* $U = (S, Q, D, R, E, H, C, P, N)$*, any* $\mathsf{CauchyFilter}^{\uparrow}$ *or finite filter consumer that reads the tail request must use the same modulus-compatible route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The filter row may refine the visible window named by S *through* Q*, but it cannot replace* D *by an external tolerance oracle or replace* R *by a selected limit.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootFilterModulusCompatibility`

Proof. Start with the admitted packet of Definition 5888.8. The stream row S and directed request row Q are read before any filter-facing or modulus-facing datum.

Apply the existing Cauchy filter Real-seal route. It inserts the finite request row before the dyadic tolerance ledger, so a filter consumer reaches D and R only through the displayed tail request.

The structural rows preserve that order under hsame transport and Cont replay. Thus a finite filter may narrow the displayed request, but the modulus and regular readback remain the same BEDC rows. □

Theorem 4715.10 (Cauchy root RegSeqRat Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *handoff exported by a root* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the finite windows, directed request, dyadic tolerance ledger, and regular rational readback of Definition 5888.9. The Real seal* E *is available only after* S, Q, D, R, H, C, P, N *have been displayed, and it exports no quotient filter, selected real, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the filter-modulus compatibility theorem. It places the StreamName window, directed request, dyadic tolerance ledger, and RegSeqRat readback in a fixed order before any Real-facing boundary.

The handoff row is exactly R followed by E. A Real-facing consumer therefore receives rational readback through $\mathsf{RegSeqRat}^{\uparrow}$ first and may inspect the terminal seal only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient filter, selected real, choice schedule, host equality, or ambient completeness coordinate, the handoff remains a finite BEDC route. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootRegSeqRatRealHandoff`

The $\mathsf{Cauchy}^{\uparrow}$ packet enters the L10 dependency lattice through its sibling route before completion or filter consumers inspect it. Every such consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The Cauchy route is therefore a displayed sibling dependency path, not a floating completion primitive or a host-level filter object.

Theorem 4715.11 (Cauchy L10 sibling dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer in the L10 lattice receives finite observation windows from chapter 126, regular-rational readback from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any Real-facing approximation is public. The route exports no selected limit, quotient equality, host filter object, or ambient completeness principle.*

Proof. Project the accepted Cauchy packet through its finite observation surface. The first visible rows are the $\mathsf{StreamName}^{\uparrow}$ windows, so downstream readers cannot begin from a host-level limit predicate.

Next read the regular-rational sibling. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the finite regularity readback used by Cauchy consumers before the packet reaches the Real-facing boundary.

Finally expose the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed windows and readback rows, the route supplies finite windows and regular-rational data before public approximation, and it leaves no coordinate for selected limits, quotient equality, host filters, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.L10SiblingDependency.CauchyL10SiblingDependencyRoute`

Theorem 4715.12 (Cauchy route RegSeqRat tail inversion)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer that exposes a* $\mathsf{Real}^{\uparrow}$*-facing approximation through the displayed L10 sibling route must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tail row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. If either row is absent, the consumer has no public* $\mathsf{Real}^{\uparrow}$*-facing approximation route through the* $\mathsf{Cauchy}^{\uparrow}$ *L10 lattice.*

Proof. Start with Definition 5888.11. A public approximation route is visible only after the accepted Cauchy packet has passed through finite $\mathsf{StreamName}^{\uparrow}$ windows, regular-rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Read the windows first. They determine which finite observations are available to the consumer, but they do not by themselves supply a regular rational tail or a tolerance account.

Next inspect the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The regular-rational tail supplies the readback values, while the dyadic ledger supplies the tolerance discipline needed before those values can be read as a Real-facing approximation.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed tail and tolerance rows, deleting either row leaves no public route from the Cauchy consumer to a Real-facing approximation. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRouteRegSeqRatTailInversion`

Theorem 4715.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Theorem 4715.14 (Cauchy public L10 export)**.** *The public* $\mathsf{Cauchy}^{\uparrow}$ *export for L10 consumers is exactly the finite route already established by Definition 3413.1, Definition 5888.16, Definition 5888.18, and Definition 5888.10:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *route. A public consumer receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyPublicL10Export`

Proof. Start with the carrier boundary. Definition 5888.16 exhausts the accepted Cauchy packet by the Cauchy source row, finite windows, dyadic tolerances, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Use the scoped filter-modulus route. Definition 5888.18 shows that filter, modulus, Real-facing, and completion-facing consumers all read the same finite path rather than an external completion object.

Finally apply the root readback handoff of Definition 5888.10. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same window and tolerance rows. Thus the public L10 export is the displayed route itself, with no additional host equality, selection, choice, quotient, or completeness principle. □

Theorem 4715.15 (Cauchy forward L10 sum-tail route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer that reaches the regular sum tail-estimate surface must first pass through the regular rational source of chapter 301, the terminal real-name boundary of chapter 38, and the finite tail-estimate packet of chapter 7387. The route is an L10 dependency path through displayed StreamName, DyadicRatCore,*

RegSeqRat, RegularCauchySum, Real, BHist, hsame, Cont, Pkg, *and* NameCert *rows; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child routes. The core carrier exposes finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real-facing seal before any downstream completion or arithmetic consumer is available.

The regular sum tail-estimate chapter consumes exactly that surface twice, once for each regular source row. Its tail-estimate ledger then adds the RegularCauchySum row and finite error budget without creating a detached limit object.

Thus the forward L10 route from the Cauchy hub to the sum-tail packet is the displayed dependency path through RegSeqRat, Real, and the finite tail-estimate surface. Since each row is already a finite NameCert coordinate, no excluded host or choice principle can be read from the hub. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyBaseForwardL10Route`

Theorem 4715.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 4715.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 4715.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 4716

A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on uniform embeddings, the CauchyCompletionEmbeddingReflection$^{\uparrow}$ packet records the finite route by which a dense completion embedding is reflected through the separated completion interface. It sits over the dense completion embedding of chapter 5893, the dense uniform embedding surface of chapter 1651, the functorial completion route of chapter 1937, the separated reflection boundary of chapter 3543, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real seal of chapter 38. The packet names the completion-embedding reflection route only; it does not select representatives, form a quotient completion outside the displayed certificate, or import ambient completeness.

Definition 4716.1 (CauchyCompletionEmbeddingReflection carrier)**.** *A* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source approximation row whose metric or uniform histories feed the completion embedding.*
2. *G is the* CauchyCompletionDenseEmbedding$^{\uparrow}$ *row naming the dense completion embedding and its image-window requests.*
3. *K is the* DenseUniformEmbedding$^{\uparrow}$ *row that exposes uniform density before the completion target is read as separated.*
4. *R is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row transporting maps and comparison windows through completion.*
5. *S is the* SeparatedCompletionReflection$^{\uparrow}$ *row that reflects the target route through the separated completion boundary.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row attached to the same finite approximation windows.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row reached after A, G, K, R, S, D have been displayed.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the dense embedding, separated reflection, dyadic tolerance, and regular-rational readback rows are fixed.*

9. *H records componentwise* hsame *transport for the analytic rows and the structural rows.*
10. *C records displayed* Cont *replay of the same embedding-reflection route.*
11. *P is the* Pkg *provenance over* CauchyCompletionEmbeddingReflection$^{\uparrow}$, CauchyCompletionDenseEmbedding$^{\uparrow}$, DenseUniformEmbedding$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, SeparatedCompletionReflection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected representatives, countable choice, a detached separated completion, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.TasteGate.CauchyCompletionEmbeddingReflectio
single carrier alignment

Theorem 4716.2 (CauchyCompletionEmbeddingReflection NameCert obligations)**.** *For every accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier* $J = (A, G, K, R, S, D, Q, E, H, C, P, N)$*, the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed approximation row, dense completion embedding row, dense uniform embedding row, completion functoriality row, separated reflection row, dyadic tolerance row, regular-rational readback row, terminal real seal row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing read must pass through* $A, G, K, R, S, D, Q, E, H, C, P, N$ *in that order, and no read may bypass the separated reflection boundary.*

Proof. Project the carrier of Definition 5889.1. Its analytic rows are A, G, K, R, S, D, Q, E, followed by the structural rows H, C, P, N.

The dense route begins with A and G. The approximation row supplies the finite source histories, while the CauchyCompletionDenseEmbedding$^{\uparrow}$ row records the image-window requests through which completion consumers may inspect the dense embedding.

The rows K, R, S then bind the same route to dense uniform embedding, functorial completion transport, and separated reflection. A completion-facing consumer therefore reaches the separated target only after the dense route and functorial route have been displayed.

The rows D, Q, E provide dyadic tolerance, regular-rational readback, and the terminal real seal. Finally H, C, P, N transport, replay, source, and name the same packet, so carrier exhaustion blocks every excluded quotient, selector, detached completion, and ambient-completeness route. □

Theorem 4716.3 (CauchyCompletionEmbeddingReflection handoff)**.** *Every downstream consumer of an accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *packet receives the dense completion embedding only through the finite handoff*

$$\text{CauchyCompletionDenseEmbedding}^{\uparrow} \longmapsto \text{DenseUniformEmbedding}^{\uparrow} \longmapsto \text{CauchyCompletionFunctoriality}^{\uparrow} \longmapsto \text{Separat}$$

Thus the consumer obtains a reflected separated-completion route with finite stream windows, dyadic tolerance, regular-rational readback, and a terminal real seal; it does not obtain a quotient-completion representative, a selected dense point, countable choice, or ambient completeness.

Proof. Start with Definition 5889.2. The NameCert surface forces every public read to begin with the approximation row and the dense completion embedding row.

Read the sibling chapters named in the statement along the displayed order. The DenseUniformEmbedding$^{\uparrow}$ row supplies the uniform dense image, CauchyCompletionFunctoriality$^{\uparrow}$ transports the route through

completion, and $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ is the only separated target boundary available to the packet.

The finite observation route is then exposed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows are exactly the rows D, Q, E together with the stream-window data carried by A, G, K, R, S.

Since the carrier has only $A, G, K, R, S, D, Q, E, H, C, P, N$, the structural rows can preserve and name the handoff but cannot supply a quotient representative, selected dense point, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5889.1, Definition 5889.2, and Definition 5889.3 seed the dense completion embedding reflection route through separated completion, finite stream windows, dyadic tolerance, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, countable choice, detached separated completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense completion embedding exposure, dense uniform embedding compatibility, completion functoriality, separated reflection, finite-window readback, dyadic tolerance, regular-rational handoff, real sealing, and nonescape for BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.

Constructive story: A $\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}$ packet is generated from a source approximation row, a dense completion embedding row, a dense uniform embedding row, a completion functoriality row, a separated reflection row, dyadic tolerance, regular-rational readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4717

A Concrete Naming Certificate for $\mathsf{SnakeLemma}^{\uparrow}$

Following Cartan–Eilenberg and Weibel homological algebra, the $\mathsf{SnakeLemma}^{\uparrow}$ packet records the finite diagram chase that turns two exact rows and compatible vertical maps into the connecting boundary route. It sits over the abelian-category carrier of chapter 176, the homology carrier of chapter 98, the spectral-sequence consumer surface that exposes $\mathsf{DerivedFunctor}^{\uparrow}$ rows in chapter 178, and the derived-functor interface of chapter 177. The packet names a displayed exactness boundary; it does not import quotient-object equality, kernel-cokernel object extensionality, or an ambient abelian-category diagram oracle.

Definition 4717.1 (Snake lemma carrier)**.** *A* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row carrying the additive kernel-cokernel surface in which the diagram is read.*
2. *R_0 and R_1 are the upper and lower exact-row ledgers, each exposing three displayed object rows, two morphism rows, and the local image-kernel containment needed for exactness.*
3. *V is the vertical-compatibility row, recording the three displayed vertical morphisms and the two commutative squares that align R_0 with R_1.*
4. *K is the kernel-row ledger for the left and middle vertical maps.*
5. *C is the cokernel-row ledger for the middle and right vertical maps.*
6. *E is the exactness-boundary row, recording that the kernel leg, connecting boundary, and cokernel leg form the displayed exact boundary chain.*
7. *∂ is the connecting-boundary row from the right kernel surface to the left cokernel surface.*
8. *H records componentwise* hsame *transport for the abelian-category source, exact rows, vertical squares, kernel rows, cokernel rows, boundary rows, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from the two exact rows and vertical squares to $K, C, E, \partial, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{SnakeLemma}^{\uparrow}$*,* $\mathsf{AbelianCat}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* $\mathsf{DerivedFunctor}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

11. L is the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted snake-lemma packets only through these rows. It contains no coordinate for host quotient representatives, unbounded diagram search, selected elements outside the displayed histories, or equality of ambient abelian-category objects.

Lean4 ✓ `BEDC.Derived.SnakeLemmaUp.TasteGate.SnakeLemmaTasteGate single carrier alignment`

Theorem 4717.2 (Snake lemma NameCert obligations)**.** *For every accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *surface consists exactly of abelian-category source exposure through A, upper and lower exact-row exposure through R_0, R_1, vertical-square compatibility through V, kernel-row exposure through K, cokernel-row exposure through C, exact-boundary exposure through E, connecting-boundary exposure through ∂, componentwise transport through H, displayed replay through T, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 5890.1. The rows A, R_0, R_1, V expose the ambient additive kernel-cokernel surface, the two exact rows, and the square compatibility before any boundary read is allowed.

The kernel and cokernel ledgers are then read from K and C. They supply the visible source and target surfaces for the connecting row ∂, while E records the exactness statement tying the kernel leg, boundary leg, and cokernel leg into one chain.

Transport, replay, provenance, and local naming preserve this same finite packet through H, T, P, L. Since no excluded quotient representative, host object equality, or unbounded diagram search appears among the coordinates, the listed rows exhaust the local certificate obligations. □

Theorem 4717.3 (Snake lemma exact boundary)**.** *Every downstream* $\mathsf{Homology}^{\uparrow}$ *or* $\mathsf{DerivedFunctor}^{\uparrow}$ *consumer of an accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *packet reads the boundary route*

$$\ker(v_2) \longmapsto \operatorname{coker}(v_0)$$

only through the displayed chain

$$\mathsf{AbelianCat}^{\uparrow} \longmapsto (R_0, R_1, V) \longmapsto (K, C) \longmapsto (\partial, E) \longmapsto (H, T, P, L).$$

Thus the consumer receives one finite exact-boundary packet of abelian-category source rows, exact-row ledgers, vertical-square ledgers, kernel and cokernel rows, the connecting-boundary row, exactness-boundary evidence, transport, replay, provenance, and local naming.

Proof. Start with the abelian-category source row A. The two exact rows R_0, R_1 and the vertical-compatibility row V expose exactly the diagram data on which the classical snake-lemma chase depends.

Read K and C next. These rows identify the displayed kernel and cokernel surfaces that the connecting boundary is permitted to connect, without choosing representatives outside the finite histories.

The boundary row ∂ and exactness row E then supply the only route from the right kernel surface to the left cokernel surface. A homology or derived-functor consumer therefore reads the boundary as a named packet, not as an ambient quotient construction.

Finally H, T, P, L keep the same route stable under componentwise history equality, displayed replay, provenance, and local naming. No quotient-object equality, host diagram oracle, or unbounded element chase is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SnakeLemma}^{\uparrow}$ packet is generated from an AbelianCat source row, two exact-row ledgers, vertical-square compatibility rows, kernel-row and cokernel-row ledgers, a connecting-boundary row, an exactness-boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5890.1, Definition 5890.2, and Definition 5890.3 seed the finite snake-lemma packet over abelian-category source rows, exact-row ledgers, vertical-square compatibility, kernel rows, cokernel rows, connecting-boundary rows, exactness-boundary evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-object equality, host abelian-category equality, unbounded diagram search, or ambient element-chase completeness.

Upgrade path: Obligation closure requires checked carrier admission, exact-row exposure, vertical-square compatibility, kernel-row exposure, cokernel-row exposure, connecting-boundary construction, exactness-boundary preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SnakeLemmaUp.

Chapter 4718

A Concrete Naming Certificate for $\mathsf{RealTailDiameterGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet records the finite gauge by which a real-facing consumer reads a regular rational tail only after the displayed tail-diameter, dyadic tolerance, stream-window, and readback rows have been exposed. It sits after the tail-diameter witness of chapter 1746, the dyadic tolerance core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a non-quotient L10 gauge object for Real and Completion consumers; it is not host real equality, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4718.1 (Real tail-diameter gauge BHist source)**.** *A* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, T, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the real-facing boundary that the gauge is allowed to serve.*
2. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row, recording the inspected tail threshold, two visible tail indices, and the displayed diameter bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the same tail-diameter ledger.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite source observations inspected by T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window and tolerance data before any real-facing read is accepted.*
6. *H records componentwise* hsame *transport for the seal, tail-diameter ledger, dyadic tolerance, window, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a real-facing gauge request through E, T, D, W, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealTailDiameterGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, T, D, W, R, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealTailDiameterGaugeUp.TasteGate.RealTailDiameterGaugeTasteGate single carrier alignment`

Theorem 4718.2 (Real tail-diameter gauge NameCert obligation surface)**.** *For every accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier*

$$G = (E, T, D, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*.*

Proof. Project the carrier of Definition 5891.1. The analytic rows are the Real seal E, tail-diameter ledger T, dyadic tolerance D, finite window W, and regular rational readback R.

The gauge obligation is read only through T, D, W, R. The tail-diameter ledger records the visible tail request and bound, the dyadic row supplies the tolerance budget, the stream row exposes the inspected finite windows, and the regular row repacks the same data for rational readback.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Since the carrier has no coordinate for the excluded host or completion objects, the displayed rows exhaust the local NameCert surface. □

Theorem 4718.3 (Real tail-diameter gauge RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *packet factors through*

$$W, D \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real and Completion route may read the finite stream window, dyadic tolerance ledger, tail-diameter gauge, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming rows, but it receives no host real equality, quotient stream equality, selected limit, trichotomy decision, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge check.*

Proof. Start with an exported regular-rational or real-facing read. The only finite observations available to the packet are the $\mathsf{StreamName}^{\uparrow}$ row W and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D.

The tail-diameter ledger T then binds those observations to a displayed tail request and diameter bound. This row is the gauge that prevents a consumer from replacing the finite tail evidence by a quotient stream or a selected limit.

Only after T is displayed does the handoff expose the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row E. The readback and seal are therefore downstream from the same finite gauge data.

Finally H, C, P, N transport, replay, source, and name the route. Since the excluded objects do not occur in the carrier, no consumer can bypass the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet is generated from a RealUp seal row, a CauchyTailDiameterUp ledger, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, seed)

Scope closed: Definition 5891.1, Definition 5891.2, and Definition 5891.3 seed the finite real-facing tail-diameter gauge for regular rational and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, countable choice, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, Real seal confinement, CauchyTailDiameter ledger exactness, DyadicRatCore tolerance readback, StreamName window coverage, RegSeqRat handoff, transport stability, continuation replay, provenance non-escape, and local naming for BEDC.Derived.RealTailDiameterGaugeUp.

This hub routes the EffectiveRealCompactIntervalSearch$^{\uparrow}$ packet through chapter 1250, chapter 1767, and chapter 5899 before the child files expose the compact interval search carrier.

Chapter 4719

A Concrete Naming Certificate for EffectiveRealCompactIntervalSearch$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the EffectiveRealCompactIntervalSearch$^{\uparrow}$ packet records the finite search surface by which a continuous real-valued map on a displayed closed bounded interval is inspected through compact finite nets, uniform-modulus data, and interval bisection rows before a downstream extremum consumer reads a certified bracket. It sits over the real-name seal of chapter 38, the interval endpoint carrier of chapter 1591, the compact-net selector of chapter 914, the graph and pointwise rows of chapter 130, and the tolerance handoff of chapter 369. The packet gives the constructive extreme-value route a finite searchable carrier; it does not assert classical Heine–Borel compactness, choose a host maximum, or identify quotient real functions.

The search packet consumes the compact interval finite-net route before it exports bisection or extreme-value handoffs. Its selector row P is read downstream of the closed-bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; the bisection ledger then refines those displayed cells instead of introducing an open-cover compactness argument or a selected supremum.

Definition 4719.1 (Effective real compact-interval search carrier)**.** *An* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, U, P, B, V, H, C, Q, N)$$

with the following visible rows:

1. *I is the* ClosedBoundedInterval$^{\uparrow}$ *endpoint and containment row whose endpoints are read through* Real$^{\uparrow}$ *names.*
2. *F is the* ContinuousMap$^{\uparrow}$ *graph row for a real-valued map over the interval source named by I.*
3. *U is the* UniformModulus$^{\uparrow}$ *tolerance handoff for F on the displayed interval source.*
4. *P is the* CompactNetModulusSelector$^{\uparrow}$ *finite-net row that lists the rational mesh cells and selected centers used by the search.*
5. *B is the bisection ledger, a finite prefix tree of interval halves whose cells refine P at the requested tolerance.*
6. *V is the value-comparison ledger storing the displayed* Real$^{\uparrow}$ *value rows and rational error budgets attached to the cells of B.*

7. *H stores componentwise* hsame *transport for the interval, continuous-map, uniform-modulus, selector, bisection, value-comparison, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an extremum-search request through $I, F, U, P, B, V, H, Q, N$.*
9. *Q is the* Pkg *provenance over* EffectiveRealCompactIntervalSearch$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *row.*

The classifier compares accepted search carriers only through $I, F, U, P, B, V, H, C, Q, N$. No open-cover compactness theorem, selected host maximum, excluded-middle comparison on all real values, quotient function equality, or ambient completeness coordinate is part of the carrier.

Theorem 4719.2 (Effective real compact-interval search obligations)**.** *For every accepted* EffectiveRealCompactInte *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, value-comparison ledger, transport, replay, provenance, and local naming;*
2. *source agreement, requiring I, F, U, and P to name the same displayed interval source before any cell is searched;*
3. *selector exactness, requiring the search cells in B to refine the finite* ProbeBundle *centers and tolerances exposed by P;*
4. *value-row binding, requiring every value comparison in V to be attached to a cell of B, a displayed graph row of F, and the error budget supplied by U;*
5. *bisection stability, requiring interval refinement to preserve endpoint containment and the selected candidate cell under componentwise* hsame *transport;*
6. *extremum handoff, requiring any lower-bound, upper-bound, near-minimum, or near-maximum read to pass through B, V, H, C, Q, N rather than through a host maximum selector;*
7. *non-escape, excluding open-cover compactness, classical Heine–Borel selection, trichotomy on arbitrary real values, quotient function equality, and ambient* Real$^{\uparrow}$ *completeness from the local packet.*

No effective compact-interval search obligation is stored outside these rows.

Proof. Project the carrier of Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, and the finite selector row is P.

The search may begin only after these rows agree on the same displayed interval source. The bisection ledger B then refines the selector cells supplied by P, so each later comparison is attached to a visible interval cell rather than to an external point choice.

The value ledger V reads graph values from F with the error budget delivered by U. This makes every candidate comparison a finite statement about named cells and rational tolerances.

Finally the extremum-facing read uses H, C, Q, N to replay and name the same route. Since the carrier has no coordinate for open-cover compactness, a host maximum, global real trichotomy, quotient function equality, or ambient completeness, none of those objects can enter the local certificate. □

Theorem 4719.3 (Effective real compact-interval bisection induction)**.** *For an accepted* EffectiveRealCompactInterv *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, every finite bisection stage in B is obtained by induction from the endpoint row I and the selector row P. The induction step replaces one displayed interval cell by its two named halves, transports the corresponding* ContinuousMap$^{\uparrow}$ *and* UniformModulus$^{\uparrow}$ *reads through H, and extends V only by value rows attached to those halves.*

Proof. The base stage is the interval cell named by I together with the first finite-net read in P. Its value ledger is the corresponding displayed graph read from F with the tolerance supplied by U.

For the induction step, choose the cell already selected by the finite ledger. The midpoint and half-interval endpoints are read as interval rows below I, and the selector exactness obligation identifies their centers with the relevant ProbeBundle entries of P.

Transport along H keeps the continuous-map graph and uniform-modulus handoff on the same source. The new value rows are therefore added to V only for the two displayed halves, so the induction never leaves the finite bisection tree. □

Theorem 4719.4 (Effective real compact-interval extreme-value handoff)**.** *Every extremum-search consumer reading an accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *carrier factors through*

$$I, F, U, P \longmapsto B, V \longmapsto H, C, Q, N.$$

The consumer receives a finite bracket or near-extremal cell family whose error budget is supplied by UniformModulus$^{\uparrow}$ *and whose candidates are selected by* CompactNetModulusSelector$^{\uparrow}$*. It receives no classical maximum point, open-cover compactness proof, quotient equality of functions, global trichotomy on* Real$^{\uparrow}$ *values, or ambient completeness assertion.*

Proof. Begin with the carrier projection. The only source data available to the consumer are the interval row I, continuous graph row F, uniform-modulus row U, and compact-net selector row P.

Apply Definition 5892.3. Every searched cell and every candidate value is produced inside the finite bisection ledger B and value ledger V.

The final handoff reads those ledgers through H, C, Q, N. Applying Definition 5892.2, any route that asks for a host maximum, open-cover compactness, quotient function equality, global real trichotomy, or ambient completeness is outside the accepted packet. □

Theorem 4719.5 (Effective real compact interval search finite-net handoff)**.** *Every accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *search packet reads a compact interval finite-net certificate, a located interval bisection row, and the* Real$^{\uparrow}$ *comparison seal before exporting the extreme-value handoff. The route passes through the closed bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; it exports no open-cover compactness, selected supremum, host trichotomy, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The compact interval finite-net data enter through the selector row P, whose cells are read only after the displayed closed bounded interval source I is fixed.

Apply Definition 5892.3. The bisection row B refines the finite cells supplied by P, and every value comparison in V is attached to a displayed graph row of F with the tolerance budget from U.

Apply Definition 5892.4. The final route reads B, V, H, C, Q, N after the finite-net source rows, so the exported extreme-value handoff is a finite search certificate rather than an open-cover compactness, supremum-selection, trichotomy, quotient-equality, or completeness principle. □

Theorem 4719.6 (Effective real compact interval search L10 sibling dependency)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must display the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint carrier, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite net, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *selector, the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *row, and the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement before exporting any extremum handoff. The route is the finite-net and bisection dependency route of Definition 5892.5, not an isolated kernel-only search packet.*

Proof. Project the carrier definition, Definition 5892.1. The rows I, F, U, P, B, V expose the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, and value ledger, while the Real-facing seal is read through the displayed interval and value rows.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before any bisection or extreme-value consumer can read the packet.

The bisection row B then supplies the located refinement consumed by the extremum handoff. The structural rows H, C, Q, N only transport, replay, source, and name that same sibling-dependent route, so no extremum handoff is exported before the listed L10 siblings are displayed. □

Theorem 4719.7 (Effective real compact interval search inverse modulus bisection)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, a displayed* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row* U *and a* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row* P *determine a finite bisection depth in* B *whose dyadic cell diameter is below the requested tolerance. The depth is read only through* P, B, V, H, C, Q, N*: the value ledger* V *attaches comparisons to cells at that depth,* H *transports the same cells under* hsame*,* C *replays the finite request, and* Q, N *package and name the route. The construction exports no open-cover compactness proof, no host maximum, no quotient equality of functions, no trichotomy on* $\mathsf{Real}^{\uparrow}$ *values, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the accepted search carrier. The uniform-modulus row U supplies the requested tolerance, while the selector row P displays the finite mesh data that the search is allowed to refine.

Apply Definition 5892.3. Each step of B replaces a displayed cell by named dyadic halves, so repeated refinement gives a finite depth at which the visible cell diameter is below the tolerance requested by U.

Apply Definition 5892.5. The finite-net row P remains the source of the cells refined by B, and the value ledger V records only comparisons attached to those cells.

Finally H, C, Q, N transport, replay, source, and name the same finite depth-control route. Because the carrier has no open-cover compactness coordinate, host maximum selector, quotient function equality coordinate, trichotomy oracle, or ambient completeness row, the inverse-modulus read remains finite interval-search data. □

Theorem 4719.8 (Effective real compact interval search finite-grid stability)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, the displayed finite-net cell selected by* P *and refined by the located bisection ledger* B *is preserved under componentwise* hsame *transport before any extreme-value handoff is exported. Thus every Real-facing extremum consumer reads the same finite cell, value row, and bisection row through* P, B, V, H, C, Q, N*, and receives no open-cover compactness proof, selected host maximum, or* $\mathsf{Real}^{\uparrow}$ *trichotomy.*

Proof. Project the carrier of Definition 5892.1. The finite-net cell is displayed by P, its bisection refinement is recorded in B, and the attached value comparisons are stored in V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger is consumed.

Apply the inverse-modulus bisection theorem, Definition 5892.7. The bisection depth and the cell read remain tied to P, B, V, while H transports the same displayed rows and C replays the same finite request.

The carrier has no coordinate for open-cover compactness, host maxima, quotient function equality, Real trichotomy, or ambient completeness. The exported extreme-value handoff therefore preserves only the finite-grid cell and located bisection row named by the packet. □

Theorem 4719.9 (Effective real compact interval search Bishop cover link)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$ *reads the Bishop interval-cover finite cover ledger of chapter 5124 before it exports any bisection or extreme-value handoff. The Bishop route supplies the finite interval-cover cells refined by* P *and* B*, while the local search packet preserves the continuous-map, uniform-modulus, value-comparison, transport, replay, provenance, and naming rows. The exported route contains no open-cover compactness theorem, selected supremum, host trichotomy, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, the finite selector row is P, the bisection ledger is B, and the value ledger is V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search packet to read the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger can be consumed.

Apply the grid-stability theorem, Definition 5892.8. The selected finite cell is preserved through P, B, V, H, C, Q, N, so the Bishop interval-cover ledger supplies interval-cover cells for the same displayed route rather than a separate compactness principle.

The remaining rows H, C, Q, N transport, replay, source, and name the same finite cover-to-search handoff. Since the carrier has no row for open-cover compactness, selected suprema, host trichotomy, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot be exported by the bisection or extreme-value consumer. □

Theorem 4719.10 (Effective real compact interval search RegularCauchyGluing dependency)**.** *Every completion-facing extremum-search read of an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must expose the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route of chapter 5899 after the finite bisection and value ledger are displayed and before any* $\mathsf{Real}^{\uparrow}$*-facing seal is consumed. The dependency is read through the finite-net handoff of Definition 5892.5 and the grid-stability route of Definition 5892.8; it is not supplied by quotient stream equality, selected host limits, countable choice, or ambient real completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The completion-facing rows available to the search are the bisection ledger B, the value-comparison ledger V, and the structural replay rows H, C, Q, N.

Apply Definition 5892.5. The finite-net route fixes the displayed interval cells and their Real-valued comparison windows before an extremum consumer can read the packet.

Apply Definition 5892.8. The same finite cell, bisection row, and value row are preserved under componentwise hsame transport, so the completion-facing read has a stable regular-Cauchy window to hand to the gluing sibling.

Route that window through $\mathsf{RegularCauchyGluing}^{\uparrow}$ before the terminal Real-facing seal is consumed. Since the carrier has no quotient stream-equality row, selected host limit, countable-choice

selector, or ambient-completeness coordinate, the gluing dependency is the only completion-facing route available after B and V are displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5892.1, Definition 5892.2, Definition 5892.3, Definition 5892.4, Definition 5892.5, Definition 5892.7, and Definition 5892.8 seed the finite interval-search carrier from displayed Real, closed-bounded-interval, compact-net-selector, continuous-map, finite-net, and uniform-modulus rows to extremum-facing bracket and depth-control handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical Heine–Borel compactness, selected host maxima or minima, quotient equality of functions, global real trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, selector exactness, value-row binding, bisection stability, extremum handoff, and non-escape for BEDC.Derived.EffectiveRealCompactIntervalSearchUp.

Constructive story: An $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source, ContinuousMapUp graph rows, a UniformModulusUp tolerance handoff, a CompactNetModulusSelectorUp finite-net row, a finite bisection ledger, displayed RealUp value comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ search packet consumes the closed-bounded interval net of chapter 1250, the finite epsilon-net surface of chapter 1767, the compact-net modulus selector of chapter 914, and the located interval bisection route of chapter 1711 before exporting its extremum handoff. Thus a downstream consumer sees the finite-net and bisection dependencies as displayed route data, not as an open-cover compactness proof or a selected host extremum. The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ route hands its finite bisection rows, interval-value rows, and compact-window certificates to the compatible-window route of chapter 5899 before any $\mathsf{Real}^{\uparrow}$ seal consumer reads a glued local search window. This dependency keeps each L10 finite-window consumer downstream from the displayed compact search packet, so the gluing step reads certified local windows rather than an ambient completeness or choice principle.

Chapter 4720

A Concrete Naming Certificate for $\mathsf{MarkedReal}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{MarkedReal}^{\uparrow}$ packet records a $\mathsf{Real}^{\uparrow}$ name together with a visible mark row for zero, positive, negative, and apartness evidence. It sits after the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the terminal real-name seal of chapter 38, the apartness interface of chapter 304, and the located-real boundary of **??**. The packet lets order-facing and apartness-facing consumers inspect a finite evidence mark before reading the Real seal; it does not assert trichotomy, decide host real equality, choose a sign globally, or identify marked names by quotient representatives.

Definition 4720.1 (MarkedReal carrier)**.** *A* $\mathsf{MarkedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, S, A, L, Z, Pm, Nm, Q, H, C, K, N)$$

with the following displayed rows:

1. *R is the underlying* $\mathsf{Real}^{\uparrow}$ *seal row.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the real name inspected by the mark.*
3. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed stream windows.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locality row, recording the finite locatedness request that makes the mark visible at the requested precision.*
5. *Z is the zero-mark row. It records finite evidence that the inspected name is being used through a zero-facing branch, without turning that evidence into host equality with zero.*
6. *Pm is the positive-mark row. It records a finite positive witness, such as a displayed rational lower gap, before any positive consumer may read the Real seal.*
7. *Nm is the negative-mark row. It records a finite negative witness, such as a displayed rational upper gap, before any negative consumer may read the Real seal.*
8. *Q is the* $\mathsf{ApartnessReal}^{\uparrow}$ *mark row, recording the finite apartness witness that separates the marked name from a requested comparison name or zero-facing boundary.*
9. *H records componentwise* hsame *transport for the Real seal, stream windows, regular readback, locatedness row, four mark rows, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a mark request through $S, A, L, Z, Pm, Nm, Q, R, H, K, N$.*

11. *K is the* Pkg *provenance over* $\mathsf{MarkedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkedReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, A, L, Z, Pm, Nm, Q, H, C, K, N$. It has no coordinate for trichotomy, host equality of real values, selected sign functions, quotient representatives, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MarkedRealUp.MarkedRealTasteGate single carrier alignment`

Theorem 4720.2 (MarkedReal NameCert obligations)**.** *For every accepted* $\mathsf{MarkedReal}^{\uparrow}$ *carrier*

$$M = (R, S, A, L, Z, Pm, Nm, Q, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkedReal}^{\uparrow}}$ *surface consists exactly of carrier admission for the underlying Real seal, finite StreamName windows, regular rational readback, locatedness request, zero mark, positive mark, negative mark, apartness mark, componentwise transport, displayed continuation replay, package provenance, and local naming. Any consumer that uses a zero, positive, negative, or apartness reading must expose the corresponding visible mark row before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the carrier of Definition 4720.1. The analytic rows are the Real seal R, the finite window and readback rows S, A, the locatedness row L, and the four mark rows Z, Pm, Nm, Q.

A mark consumer cannot bypass these rows. The zero-facing branch reads Z, the positive-facing branch reads Pm, the negative-facing branch reads Nm, and the apartness-facing branch reads Q before it reaches the common Real seal R.

The structural rows H, C, K, N transport, replay, source, and name the same displayed packet. Since the carrier has no trichotomy coordinate, selected sign function, host equality coordinate, quotient representative, countable-choice coordinate, or ambient completeness row, the listed rows exhaust the local NameCert surface. □

Theorem 4720.3 (MarkedReal Real seal handoff)**.** *Every* $\mathsf{MarkedReal}^{\uparrow}$ *read that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal factors through a finite route*

$$S, A, L \longmapsto Z, Pm, Nm, Q \longmapsto R \longmapsto H, C, K, N.$$

Thus the exported object is a Real name equipped with visible zero, positive, negative, and apartness evidence rows, not a trichotomy principle, host equality decision, or chosen sign representative.

Proof. Start with an accepted packet $M = (R, S, A, L, Z, Pm, Nm, Q, H, C, K, N)$. By Definition 4720.1, the only terminal Real boundary is R.

The finite route first exposes the stream windows S, regular rational readback A, and locatedness row L. These rows place the requested read in the same finite name context as the underlying Real seal.

Next the visible mark rows are inspected. The rows Z, Pm, Nm, Q record zero-facing, positive-facing, negative-facing, and apartness-facing evidence as displayed data. None of them decides all signs of all Real names, and none identifies the carrier with a host real value.

Finally the route reaches R and then the structural rows H, C, K, N. These rows preserve the same marked packet by componentwise hsame, displayed Cont, Pkg, and local NameCert data, so the handoff exposes exactly the marked Real name stated above. □

Falsifiable prediction: A $\mathsf{MarkedReal}^{\uparrow}$ consumer that reads the terminal $\mathsf{Real}^{\uparrow}$ seal as positive, negative, zero-facing, or apart from a boundary without first exposing Pm, Nm, Z, or Q respectively refutes the carrier surface.

Independence witness: real, regseqrat, streamname, apartnessreal, locatedreal: no listed sibling supplies the whole twelve-row packet $M = (R, S, A, L, Z, Pm, Nm, Q, H, C, K, N)$, because the siblings provide the Real seal, regular readback, finite windows, apartness witness, or locatedness boundary separately while this chapter binds all four visible mark rows to one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkedReal}^{\uparrow}$ packet is generated from a terminal $\mathsf{Real}^{\uparrow}$ seal, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{LocatedReal}^{\uparrow}$ request row, visible zero, positive, negative, and apartness mark rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4720.1, Definition 4720.2, and Definition 4720.3 seed the visible-mark Real carrier over Real, RegSeqRat, StreamName, ApartnessReal, and LocatedReal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkedReal}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MarkedRealUp`

Bridge status: none

Not claimed: This chapter does not claim trichotomy, host real equality, globally selected signs, quotient representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, checked NameCert obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visible mark exposure, zero-mark discipline, positive-mark discipline, negative-mark discipline, apartness-mark discipline, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MarkedRealUp.

Chapter 4721

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on dense embeddings, the $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet records the finite dense-source route by which a Cauchy completion consumer reads a source object inside its completion boundary. It sits beside the dense uniform embedding surface of chapter 1651, the functorial Cauchy-completion route of chapter 1937, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies dense embedding data for completion consumers without asserting quotient completion equality, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4721.1 (Cauchy completion dense-embedding carrier)**.** *A* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the dense source row. It records the source-side Cauchy observations that a completion consumer may approximate by displayed source data.*
2. *G is the embedding graph row. It records the displayed source-to-completion graph and the finite image-window reads used by dense embedding consumers.*
3. *K is the Cauchy completion row, read through the completion-facing route associated with* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite request windows inspected by the source and completion sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source-image windows with completion windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after A, G, K, S, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the dense source, embedding graph, completion, window, tolerance, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dense-source request through $A, G, K, S, D, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseEmbeddingUp.CauchyCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 4721.2 (Cauchy completion dense-embedding NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier*

$$J = (A, G, K, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the dense source row, embedding graph row, Cauchy completion row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing dense read must expose A, G, K, S, D, Q before E, H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5893.1. The dense source, embedding graph, Cauchy completion row, finite windows, tolerance ledger, regular readback, and Real seal are exactly the displayed analytic coordinates.

The dense-source route is read through A, G, K. A completion consumer first fixes the source row, then the embedding graph row, and only then the completion-facing row associated with the Cauchy completion route.

The finite observation route is read through S, D, Q. The stream windows expose inspected requests, the dyadic ledger supplies the tolerance budget, and the regular rational readback repacks the same observations before any Real-facing seal is visible.

Finally E, H, C, P, N preserve and name the route. Since quotient completion equality, selected representatives, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 4721.3 (Cauchy completion dense-embedding Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer exported by an accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{DenseUniformEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionFunctoriality}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More precisely, the consumer reads the dense source row A, embedding graph G, Cauchy completion row K, finite windows S, tolerance ledger D, regular readback Q, terminal seal E, componentwise transport H, displayed replay C, provenance P, and local name N before the real-facing boundary is exposed.

Proof. Start with an exported Real-facing or completion-facing read. The NameCert obligations of Definition 5893.2 force the read through A, G, K before the finite observation rows can be consumed.

Read the sibling routes next. The dense source is the $\mathsf{DenseUniformEmbedding}^{\uparrow}$ surface, the completed-map side is routed through $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, and the finite observations are the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows displayed by the carrier.

Only after those rows have been displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal row E be read. The structural rows H, C, P, N transport, replay, source, and name the same route, so the handoff exports no quotient completion equality, selected representative, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a dense source row, embedding graph row, Cauchy completion row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5893.1, Definition 5893.2, and Definition 5893.3 seed the dense-source route for Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense source exposure, embedding graph exactness, Cauchy completion handoff, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionDenseEmbeddingUp.

Chapter 4722

A Concrete Naming Certificate for $\mathsf{RiemannRefinementCriterion}^{\uparrow}$

Following Bishop and Bridges on Riemann–Darboux integration, the $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet records the finite refinement criterion that lets a tagged Riemann-sum consumer compare upper Darboux rows against a gauge-local partition route. It sits after the tagged partition carrier of chapter 4296, the upper Darboux-sum carrier of chapter 4083, the Riemann sum gauge packet of chapter 4115, and the Cousin cover carrier of chapter 3426. The packet supplies the finite refinement surface consumed by integrability criteria without importing host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4722.1 (Riemann refinement criterion carrier)**.** *A* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row shared by the refinement comparison.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *source row, including the finite cell order, tag containment, and mesh exactness visible before refinement is read.*
3. *U is the* ***??*** *row whose upper-sum ledger is compared against the refined tagged route.*
4. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row, carrying the positive rational gauge, mesh ledger, and finite tagged-sum handoff.*
5. *C is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge-cover row supplying the finite local cover route that admits the refined cells.*
6. *M is the common-refinement mesh row, listing the displayed refinement map from the tagged partition row into the gauge-cover and upper-sum rows.*
7. *D is the Darboux-gap lock row, recording the finite upper-bound comparison that remains visible after the common refinement has been read.*
8. *W is the window and readback row for interval endpoints, tags, gauges, cover cells, and upper-sum values.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tagged partition, gauge, Cousin cover, refinement mesh, and Darboux-gap rows are displayed.*

10. *H records componentwise* hsame *transport for interval, partition, upper-sum, gauge, cover, refinement, Darboux-gap, window, seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a refinement request through $I, T, U, G, C, M, D, W, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, *??*, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows. It has no coordinate for host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannRefinementCriterionUp.RiemannRefinementCriterionTasteGate single carrier alignment`

Theorem 4722.2 (Riemann refinement criterion NameCert obligations)**.** *For every accepted* RiemannRefinementCr *carrier $R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval, tagged partition, upper Darboux-sum, Riemann sum gauge, Cousin cover, refinement mesh, Darboux-gap lock, window, seal, transport, replay, provenance, and local naming rows.*

Every refinement-facing consumer must read the tagged partition row T from chapter 4296, the upper-sum row U from chapter 4083, the gauge row G from chapter 4115, and the gauge-cover row C from chapter 3426 before the common-refinement mesh row M and Darboux-gap lock row D are available.

Proof. Project the carrier of Definition 5894.1. The tuple displays the only rows available to a refinement consumer: interval source, tagged partition, upper-sum, gauge, Cousin cover, common refinement, Darboux-gap lock, finite windows, terminal seal, transport, replay, provenance, and local naming.

The source rows are T, U, G, C. The tagged partition supplies finite cells and tags, the upper Darboux row supplies the upper-bound ledger, the Riemann sum gauge row supplies the gauge mesh, and the Cousin cover row supplies local gauge cover cells.

The refinement rows are M, D. The mesh row M binds those four source rows into one common finite refinement before the Darboux-gap lock D can be read.

Finally W, E, H, K, P, N preserve the same displayed route through finite windows, the terminal Real seal, transport, replay, provenance, and local naming. Since the carrier has no coordinate for the refused host objects, the listed rows exhaust the local NameCert surface. □

Theorem 4722.3 (Riemann refinement criterion Darboux gap lock)**.** *In every accepted* RiemannRefinementCriterior *packet, the Darboux-gap row D is readable only after the common-refinement mesh row M has already displayed the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover sources. Thus a Darboux or Riemann-integrability consumer cannot replace the refinement route by host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the source rows from Definition 5894.1. The Darboux-gap row is not listed as an independent analytic conclusion; it appears after the tagged partition, upper-sum, gauge, cover, and common-refinement rows.

Apply Definition 5894.2. Its obligation surface requires T, U, G, C to be exposed before M, and M before D.

The structural rows only transport and replay the same finite packet. Therefore the Darboux-gap read is locked to the displayed refinement route and cannot be promoted to any refused host-level principle. □

Theorem 4722.4 (Riemann refinement criterion integral handoff)**.** *Every Riemann-integrability or gauge-integral consumer of an accepted* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *packet receives only the finite handoff*

$$\mathsf{TaggedRealPartition}^{\uparrow},\ \textit{??},\ \mathsf{RiemannSumGauge}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow} \longmapsto M \longmapsto D \longmapsto W, E, H, K, P, N.$$

The handoff exports a refinement criterion over displayed finite rows, not an ambient theorem about all host partitions or all mesh limits.

Proof. Start with a consumer read. The packet first exposes the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover rows through the sibling chapters cited in the carrier.

The common-refinement mesh row M then aligns those sources. The Darboux-gap lock row D is read only along that aligned route, and the window and seal rows W, E expose the Real-facing observations after the finite rows are visible.

Transport, replay, provenance, and local naming through H, K, P, N preserve the same route. No carrier coordinate supplies host partition equality, selected limiting meshes, choice, compactness, quotient function equality, or ambient completeness, so the integral handoff is exactly the displayed finite criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet is generated from a closed bounded interval row, TaggedRealPartition source, UpperDarbouxSum ledger, RiemannSumGauge mesh row, CousinCover gauge-cover row, common-refinement mesh row, Darboux-gap lock, finite windows, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5894.1, Definition 5894.2, Definition 5894.3, and Definition 5894.4 seed the finite refinement criterion between tagged partitions, upper Darboux sums, Riemann sum gauges, and Cousin covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition source lock, upper-sum handoff, gauge mesh exactness, Cousin cover locality, Darboux-gap lock, integral handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRefinementCriterionUp.

Chapter 4723

A Concrete Naming Certificate for CompactUniformModulusComposition$^{\uparrow}$

Following the Bishop–Bridges and Heine–Cantor treatment of uniformly continuous maps on compact metric data, the CompactUniformModulusComposition$^{\uparrow}$ packet records the finite tolerance-only composition surface for two compact-uniform modulus rows. It sits after the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the compact uniform modulus row of chapter 2299, the metric-triangle estimate of chapter 1159, and the stability packet of chapter 5872. The packet supplies the finite composition object used by compact-open and compact-uniform-continuity consumers without exporting a host function space, quotient point equality, selected global modulus, choice schedule, or ambient compactness theorem.

Definition 4723.1 (Compact uniform modulus composition carrier)**.** *A* CompactUniformModulusComposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are compact metric source, middle, and target rows, with finite net and distance ledgers visible at the requested tolerances.*
2. *F and G are* ContinuousMap$^{\uparrow}$ *rows for the two displayed maps.*
3. *M_F is the first* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Y into a source tolerance for X along F.*
4. *M_G is the second* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Z into a source tolerance for Y along G.*
5. *A is the tolerance-assembly row, recording that a Z-tolerance request is first read by M_G and that its resulting Y-tolerance request is then read by M_F.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that keeps the two transported estimates on the same metric route.*
7. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row for the composite map.*
8. *H records componentwise* hsame *transport for compact metric sources, continuous maps, modulus rows, tolerance assembly, triangle estimate, consumer, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a composite tolerance request through $Z, M_G, Y, M_F, X, A, T, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *row.*

The classifier compares accepted composition packets only by these finite rows. It contains no host function-space composition theorem, quotient point equality, selected global modulus, choice schedule, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusCompositionUp.CompactUniformModulusCompositionTasteGate`
`single carrier alignment`

Theorem 4723.2 (Compact uniform modulus composition NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *carrier $Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *surface consists exactly of compact source admission through X, Y, Z, continuous-map admission through F, G, first and second compact-uniform modulus reads through M_F, M_G, tolerance assembly through A, metric-triangle transport through T, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5895.1. The compact metric rows, continuous-map rows, compact-uniform modulus rows, assembly row, triangle estimate, uniform-modulus consumer, and structural rows are exactly the displayed coordinates.

The tolerance route is ordered by A. A requested Z-tolerance is consumed by M_G to expose a Y-tolerance, and only that displayed Y-tolerance is then consumed by M_F to expose the source tolerance over X.

The triangle estimate T and consumer row U read the same assembled route. Transport, replay, provenance, and local naming preserve it through H, C, P, N.

Since the carrier has no coordinate for host function-space composition, quotient point equality, selected moduli, choice, or ambient compactness, the listed rows exhaust the NameCert surface. □

Theorem 4723.3 (Compact uniform modulus composition tolerance route)**.** *Every compact-open or compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *packet reads the composite modulus only through*

$$M_G \longmapsto A \longmapsto M_F \longmapsto T \longmapsto U.$$

Thus the composite tolerance is assembled from two displayed compact-uniform modulus rows and the metric-triangle ledger, not from a selected host-level modulus for the composite map.

Proof. Begin with the consumer's target tolerance for the Z row. The second modulus row M_G supplies the middle-space tolerance, and the assembly row A records that this is the only tolerance handed to M_F.

The first modulus row M_F returns the source-space tolerance over X. The metric-triangle row T then transports the two displayed estimates into the bound consumed by U.

The structural rows H, C, P, N preserve this route. Since there is no host composition coordinate or selected global modulus in the carrier, every consumer remains on the displayed tolerance chain. □

Theorem 4723.4 (Compact uniform modulus composition nonescape)**.** *No accepted* CompactUniformModulusComp *packet exports quotient point equality, a selected global modulus, a choice schedule, an ambient compactness theorem, or a host function-space composition proof. It exports only the two continuous-map rows, two compact-uniform modulus rows, the tolerance assembly row, the metric-triangle estimate, the uniform-modulus consumer, and the structural NameCert rows.*

Proof. Project the carrier. The only analytic rows are F, G, M_F, M_G, A, T, U over the compact metric sources X, Y, Z.

Apply Definition 5895.2. The local NameCert obligations require the consumer to read the two modulus rows and the tolerance assembly before the uniform-modulus export.

The refused host objects are absent from the carrier and cannot be produced by transport, replay, provenance, or local naming. Therefore the accepted packet exports only the displayed finite composition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ packet is generated from compact metric source, middle, and target rows, two ContinuousMap rows, two CompactUniformModulus rows, a tolerance-assembly row, a MetricTriangleUniformEstimate row, a UniformModulus consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5895.1, Definition 5895.2, Definition 5895.3, and Definition 5895.4 seed the finite tolerance-only composition route for compact-uniform modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space composition, quotient point equality, selected global moduli, choice schedules, ambient compactness theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, continuous-map admission, first and second modulus reads, tolerance assembly, metric-triangle transport, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulusCompositionUp.

Chapter 4724

A Concrete Naming Certificate for $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$

The $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet realizes the private/shared decoration theorem of Definition 9462.1705 inside the boundary-tile source of subsection 9462.28.21. It sits beside the DNA boundary-tile packet of chapter 3393, the trigger-threshold packet of chapter 4217, and the finite trigger algebra route of chapter 3478. The packet records private blocks, shared rows, lift counts, and defect ledgers as finite NameCert data; it is not a biological mechanism, stochastic evolution model, hidden enumeration oracle, or quotient codon semantics.

Definition 4724.1 (PrivateSharedDecorationModel carrier)**.** *A* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, L, D, T, R, P, N)$$

with the following displayed rows:

1. *B is the nonempty private-block row, recording the partition of coordinates that remain private to a single decorated support.*
2. *S is the shared-row ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *L is the lift-count row, recording the finite ℓ^m lift factor attached to displayed support blocks.*
4. *D is the defect ledger, recording the row-set size contribution to the defect-refined polynomial.*
5. *T records componentwise* hsame *transport for B, S, L, D, R, P, N.*
6. *R records displayed* Cont *replay from a decorated trigger request through the private block, shared rows, lift count, defect ledger, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem of Definition 9462.1705,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, S, L, D, T, R, P, N. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, or an unrecorded enumeration oracle.

Theorem 4724.2 (PrivateSharedDecorationModel NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier*

$$M = (B, S, L, D, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise transport, displayed replay, provenance, and local naming. A decorated trigger read is accepted only after it exposes a nonempty private block and assigns every shared coordinate to at least two rows.*

Proof. Project the carrier of Definition 5896.1. The private block, shared-row ledger, lift count, defect ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The private/shared constraint is read through B, S. The row B must display a nonempty private partition, and the ledger S must attach every shared coordinate to a row set of size at least two.

The enumeration and defect data are read through L, D. The lift-count row records the finite lift factor, while the defect ledger records the shared-row contribution used by the source theorem.

Finally T, R, P, N transport, replay, source, and name the same route. Since biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, and unrecorded enumeration have no carrier coordinate, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: If a displayed decorated trigger lacks a nonempty private block or assigns a shared coordinate to fewer than two rows, it is rejected by the local NameCert.

Independence witness: dna-window-six-boundary-tile, dna-window-six-trigger-threshold, dna-window-six-translation-boundary: $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries private-block rows, shared-row ledgers, lift-count rows, defect ledgers, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.PrivateSharedDecorationModelUp` must provide explicit $\mathsf{Nontrivial\ PrivateSharedDecorationModel}^{\uparrow}$ and $\mathsf{FieldFaithful\ PrivateSharedDecorationModel}^{\uparrow}$ instances preserving B, S, L, D, T, R, P, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet is generated from a nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5896.1 and Definition 5896.2 seed the finite private/shared decoration packet over DNA boundary-tile trigger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global semantics, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block nonemptiness, shared-row arity, lift-count exposure, defect-ledger exposure, transport, replay, provenance, local naming, Nontrivial PrivateSharedDecorationModelUp, and FieldFaithful PrivateSharedDecorationModelUp for BEDC.Derived.PrivateSharedDecorationModelUp.

Chapter 4725

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet records the finite filter-facing route by which located real brackets are read through dyadic tolerances, stream windows, regular rational readback, and the terminal real seal. It consumes the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located bracket surface of chapter 1709, and the selector boundary of chapter 5886. The packet gives the L10 Real and Completion route a located filter interface without asserting quotient filter equality, host real equality, selected limit points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4725.1 (Bishop located real filter carrier)**.** *A* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier is a finite packet*

$$S = (F, L, D, W, R, E, B, H, T, P, N)$$

with the following displayed rows:

1. *F is the filter-base row, exposing the finite base request, refinement evidence, and Cauchy-facing filter data that a later completion consumer may inspect.*
2. *L is the located-real bracket row, recording lower and upper bracket reads over* $\mathsf{LocatedRealInterval}^{\uparrow}$ *together with the displayed apartness and containment data used to locate the requested real value.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing bracket width, filter-base comparison, stream-window radius, and regular-rational readback.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the filter request and the located bracket.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W under the tolerance ledger D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after F, L, D, W, R have been displayed and replayed.*
7. *B is the* BHist *history row recording the finite route from the filter request through the located bracket, tolerance ledger, stream windows, regular readback, and Real seal.*
8. *H records componentwise* hsame *transport for $F, L, D, W, R, E, B, T, P, N$.*
9. *T records displayed* Cont *replay from every located filter request through $F, L, D, W, R, E, B, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, D, W, R, E, B, H, T, P, N$. It has no coordinate for quotient filter equality, host selected representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BishopLocatedRealFilterUp.BishopLocatedRealFilterUp`

Theorem 4725.2 (Bishop located real filter NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier $S = (F, L, D, W, R, E, B, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the filter base, located-real bracket, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, history, transport, replay, provenance, and local naming row;*
2. *bracket exactness, requiring each located-real bracket read in L to be witnessed by displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint and containment data;*
3. *filter-base stability, requiring refinements of F to preserve the same located bracket and tolerance ledger through* BHist *and* hsame *transport;*
4. *dyadic ledger stability, requiring D to control the bracket width, stream-window radius, and regular-rational readback without adding an external modulus;*
5. *StreamName and RegSeqRat handoff, requiring every numeric read to pass through W and then R before the terminal seal E can be consumed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, selected limiting points, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5897.1. The accepted packet displays the filter base, located bracket, dyadic ledger, stream windows, regular readback, Real seal, history, transport, replay, provenance, and local naming rows.

Bracket exactness is row-local. The bracket row L can expose only the endpoint and containment data already carried by $\mathsf{LocatedRealInterval}^{\uparrow}$, so it cannot identify a real by host order equality or by a selected representative.

The filter-base and tolerance rows then fix the route. Refinement of F is transported through B and H, while the same dyadic row D governs bracket width, window radius, and readback precision.

Finally the numeric handoff is ordered by W, R, E. Stream windows are read before regular rational data, and regular rational data are read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4725.3 (Bishop located real filter L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *packet $S = (F, L, D, W, R, E, B, H, T, P, N)$ must read the finite route*

$$F, L, D, W, R \longmapsto E \longmapsto B, H, T, P, N$$

before it treats the request as Real-facing or completion-facing data. Thus the packet supplies a filter-based located-real handoff to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, *and*

$\mathsf{CauchyFilterSelector}^{\uparrow}$ *consumers without asserting ambient real completeness or quotient equality of filters.*

Proof. Read the carrier rows in order. A consumer first receives the filter base F and the located bracket L, while the dyadic ledger D supplies the common tolerance budget.

The numerical content is then exposed through W and R. The $\mathsf{StreamName}^{\uparrow}$ row records finite observation windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational readback extracted from those windows.

Only after these rows are displayed can the Real seal E be consumed. The history, transport, replay, provenance, and local naming rows B, H, T, P, N preserve that order for later L10 Real and Completion routes.

The carrier has no quotient-filter equality, selected representative, host real equality, choice schedule, or ambient completeness coordinate. Therefore every downstream handoff factors through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet is generated from a displayed filter base, a located-real bracket, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a Real seal, BHist history, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5897.1, Definition 5897.2, and Definition 5897.3 seed the finite filter-facing located-real interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, ambient Real completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, bracket exactness, filter-base stability, dyadic ledger stability, StreamName and RegSeqRat handoff, and Real seal boundary for BEDC.Derived.BishopLocatedRealFilterUp.

Chapter 4726

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ packet records the finite diagonal extraction route that turns a displayed regular Cauchy family of rational names into the regular rational readback consumed by a $\mathsf{Real}^{\uparrow}$ seal. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ source surface of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38, and the sibling diagonalization packet of chapter 3782. The packet names the L10 RegSeqRat-to-Real handoff without importing quotient streams, selected limits, choice, or ambient Real completeness.

Definition 4726.1 (Regular Cauchy diagonal argument carrier)**.** *A* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *family row. It displays the finite collection of rational-name approximants and the precision requests inspected by the diagonal extraction.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite source windows selected for each inspected precision and exposes no quotient stream equality.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic radius, split budget, common scale, and comparison cells used to certify the diagonal readback.*
4. *R is the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It consumes only the displayed windows W and tolerance ledger D as one regular rational sequence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after F, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the source family, stream windows, tolerance ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal extraction request through the source family, finite stream windows, dyadic ledger, regular rational readback, Real seal, transports, provenance, and local name.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a checked bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentTasteGate`
`single carrier alignment`

Theorem 4726.2 (Regular Cauchy diagonal argument NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier*

$$A = (F, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring W to expose only the finite source windows named by the diagonal extraction request;*
3. *dyadic tolerance exactness, requiring D to supply the radius, split budget, common scale, and comparison cells used by the extraction;*
4. *regular readback exactness, requiring R to consume precisely the windows W and dyadic ledger D before any Real-facing seal is visible;*
5. *Real-seal nonescape, requiring E to appear only after F, W, D, R have been transported by H and replayed by C;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming for exactly the rows $F, W, D, R, E, H, C, P, N$;*
7. *refusal of quotient streams, selected limits, choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No local diagonal-argument obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentNamecertObligations`

Proof. Project the carrier of Definition 5541.1. The accepted inventory is exactly the source $\mathsf{RegSeqRat}^{\uparrow}$ family F, finite $\mathsf{StreamName}^{\uparrow}$ windows W, dyadic tolerance ledger D, diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback R, terminal $\mathsf{Real}^{\uparrow}$ seal E, and structural rows H, C, P, N.

Window and tolerance exactness are read from W and D. The stream row exposes only finite inspected windows, and the dyadic row supplies the radius, split budget, common scale, and comparison cells needed to compare those windows.

The regular readback row R is downstream of those two rows. It cannot name a quotient stream or a selected limit, because the carrier gives it only W and D as inputs.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same packet. Since the carrier inventory contains no quotient streams, selected limits, choice operator, ambient completeness principle, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentCarrier namecert obligations`

Theorem 4726.3 (Regular Cauchy diagonal argument Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier factors through the finite route*

$$F \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a diagonal extraction is available to the Real boundary only as a displayed $\mathsf{RegSeqRat}^{\uparrow}$ *family row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, one terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence.*

Proof. Begin with the obligation surface of Definition 5541.2. It confines every accepted packet to the displayed family, window, tolerance, readback, Real-seal, and structural rows.

The consumer first reads F, W, D. These rows expose the source regular rational names, the finite stream windows inspected by the diagonal argument, and the dyadic comparison budget.

The consumer then reads R and E. The regular rational readback is formed from the same finite windows and dyadic ledger, and the terminal Real seal is visible only after that readback has been displayed.

The rows H, C, P, N preserve this route under transport, replay, provenance, and naming. Because the carrier has no coordinate for quotient streams, selected limits, choice, ambient Real completeness, Lean verification, or bridge checking, none of those objects can enter the Real-facing handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ Real or Completion read must expose a RegSeqRat family row, finite StreamName windows, one DyadicRatCore tolerance ledger, one diagonal RegSeqRat readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-diagonalization: $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds source RegSeqRat family data, finite StreamName windows, dyadic tolerance exactness, diagonal RegSeqRat readback, and RealUp handoff in one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ packet is generated from a RegSeqRat family row, finite StreamName windows, one DyadicRatCore tolerance ledger, one diagonal RegSeqRat readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5541.1, Definition 5541.2, and Definition 5541.3 seed the finite diagonal extraction packet used by L10 RegSeqRat-to-Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, regular readback exactness, Real-seal exposure, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyDiagonalArgumentUp.

Chapter 4727

A Concrete Naming Certificate for PrivateSharedCoordinateDecomposition$^{\uparrow}$

The PrivateSharedCoordinateDecomposition$^{\uparrow}$ packet realizes the private/shared coordinate decomposition of Definition 9462.1697 as finite NameCert data. It reads the quotient-level support family, private blocks, shared-coordinate row sets, and reconstruction equations through the support-census sibling of chapter 4048 and the reassignment-geometry sibling of chapter 5869. The packet is a finite combinatorial carrier; it is not a biological mechanism claim, a hidden quotient of support labels, or an ambient uniqueness principle outside the displayed rows.

Definition 4727.1 (Private-shared coordinate decomposition carrier)**.** *A* PrivateSharedCoordinateDecomposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, B, U, R, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite support-family row, displaying the quotient-level support family on the inspected coordinate set.*
2. *B is the private-block row, listing the nonempty private coordinate blocks attached to the supports.*
3. *U is the shared-coordinate row, listing the coordinates not owned by a single private block.*
4. *R is the row-set ledger. For every shared coordinate it records the finite set of private blocks whose supports contain that coordinate.*
5. *K is the reconstruction row, recording the displayed equations that recover each support from its private block and the shared coordinates whose row sets contain that block.*
6. *H records componentwise* hsame *transport for support-family, private-block, shared-coordinate, row-set, reconstruction, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from support-family inspection through private-block extraction, shared-coordinate listing, row-set lookup, support reconstruction, transports, provenance, and local name.*
8. *P is the* Pkg *provenance over* PrivateSharedCoordinateDecomposition$^{\uparrow}$*, the source private/shared coordinate vision,* SupportNerveCensus$^{\uparrow}$*,* ReassignmentMassSubcube$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, B, U, R, K, H, C, P, N$. It has no coordinate for hidden support equality, unbounded coordinate universes, biological causality, stochastic dynamics, Lean verification, or a checked bridge.

Theorem 4727.2 (PrivateSharedCoordinateDecomposition NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ *carrier*

$$Q = (S, B, U, R, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$ *surface consists exactly of support-family exposure, private-block exposure, shared-coordinate exposure, finite row-set exposure, reconstruction equation exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The reconstruction row must use only the displayed private block and shared-coordinate row-set data from B, U, R.*

Proof. Project the carrier of Definition 5542.1. The rows S, B, U, R display the support family, the nonempty private blocks, the shared coordinates, and the finite row sets attached to those shared coordinates.

The reconstruction row K is downstream of those data. It may rebuild a support only by taking its private block from B and adjoining exactly the shared coordinates from U whose row-set entry in R contains that block.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no hidden support equality, unbounded coordinate universe, biological mechanism, stochastic dynamics, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 4727.3 (Private-shared coordinate reconstruction handoff)**.** *Every consumer of an accepted* $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto B, U, R \longmapsto K \longmapsto H, C, P, N.$$

Consequently, the support corresponding to a private block is recovered only from the displayed private block, the shared-coordinate ledger, and the finite row-set membership data, exactly as in the source decomposition of Definition 9462.1698.

Proof. Start with the NameCert obligations of Definition 5542.2. They force the support-family row S to expose the finite family before private and shared coordinates are read.

The rows B, U, R then provide the decomposition data. Private blocks give the singleton-owned coordinates, while shared coordinates are paired with finite row sets naming the blocks whose supports contain them.

The reconstruction row K consumes only those displayed rows. Transport, replay, provenance, and local naming preserve the same finite route, so a consumer cannot replace it by a hidden quotient of supports or by an ambient uniqueness principle. □

Falsifiable prediction: Every accepted $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ consumer must expose finite support data, private blocks, shared coordinates, row-set membership, reconstruction equations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before reading a reconstructed support.

Independence witness: support-nerve-census, reassignment-mass-subcube, private-shared-coordinate vision: $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds private blocks, shared-coordinate row sets, reconstruction equations, transport, replay, provenance, and local naming as one finite coordinate-decomposition packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ packet is generated from finite support-family rows, private-block rows, shared-coordinate rows, row-set ledgers, reconstruction equations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5542.1, Definition 5542.2, and Definition 5542.3 seed the finite NameCert packet for private/shared coordinate decomposition.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hidden support equality, unbounded coordinate universes, biological causality, stochastic dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support-family exposure, private-block exposure, shared-coordinate exposure, row-set exposure, reconstruction exactness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.PrivateSharedCoordinateDecompositionUp.

Chapter 4728

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyModulus}^{\uparrow}$

Following Bishop and Bridges on constructive moduli of Cauchy convergence, the $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ packet records an explicit finite threshold schedule for a regular rational Cauchy name before the terminal real-name seal is consumed. It sits after the regular rational source of chapter 301, the finite window surface of chapter 126, the dyadic tolerance core of chapter 291, the Cauchy modulus rows of chapter 292, and the terminal real-name boundary of chapter 38. The packet gives the L10 Real and Completion route a non-quotient explicit-modulus carrier; it does not select a limit, quotient streams, invoke countable choice, assert host equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4728.1 (Regulated Cauchy modulus carrier)**.** *A* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, Q, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose finite rational reads are being regulated.*
2. *Q is the explicit modulus threshold row, recording the displayed tolerance request and the finite index bound at which the Cauchy comparison is certified.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to expose the source reads needed by Q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the requested radius, the comparison budget, and the readback precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, Q, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the source, threshold, window, tolerance, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through R, Q, W, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ BEDC.Derived.RegulatedCauchyModulusUp.TasteGate.RegulatedCauchyModulusTasteGate single carrier alignment

Theorem 4728.2 (Regulated Cauchy modulus NameCert obligations)**.** *For every accepted* RegulatedCauchyModulu *carrier* $M = (R, Q, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, explicit threshold exactness, dyadic tolerance control, real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read through the finite route*

$$R, Q, W, D \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Carrier admission is projection from Definition 5543.1. The packet first exposes the regular rational source R, the explicit threshold row Q, the stream windows W, and the dyadic tolerance ledger D.

The threshold obligation is local to Q. It records a displayed tolerance request and a finite index bound, while W supplies the finite source reads and D supplies the dyadic comparison budget used to check the requested tail comparison.

The terminal seal row E is downstream of those four rows. Componentwise transport through H and replay through C preserve the same order, and P, N record package provenance and the local NameCert comparison.

Since quotient streams, selected limits, host real equality, countable choice, trichotomy, and ambient completeness do not occur among the carrier coordinates, no NameCert obligation can be discharged outside the displayed finite route. □

Theorem 4728.3 (Regulated Cauchy modulus real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *packet must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the explicit modulus threshold, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The handoff to the seal row* E *therefore supplies a finite explicit-modulus route from regular rational data to Real-facing data, not a host limit theorem or a quotient construction.*

Proof. Project an accepted carrier $M = (R, Q, W, D, E, H, C, P, N)$. The source row R is the only regular rational input, and the threshold row Q is the only displayed modulus schedule.

The stream windows W expose the finite observations required by that schedule. The dyadic ledger D supplies the tolerance comparison that makes the schedule a regulated Cauchy read rather than an unbounded stream claim.

Only after R, Q, W, D are present may the consumer read E as the terminal $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming keep the same route visible to downstream consumers.

The carrier has no coordinate for a selected limit, quotient stream equality, host real equality, countable choice, trichotomy, or ambient Real completeness. Thus every Real-facing handoff factors through the finite explicit-modulus rows. □

Theorem 4728.4 (Regulated Cauchy modulus tail-threshold determinacy)**.** *Let* $M = (R, Q, W, D, E, H, C, P, N)$ *and* $M' = (R', Q', W', D', E', H', C', P', N')$ *be accepted* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *reads whose displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and threshold row agree. Then the two reads expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route. Strengthening a request may enlarge the finite window demanded by* W *and* W'*, but it cannot change the displayed source row, tolerance ledger, threshold row, or terminal seal route.*

Proof. Project both accepted carriers. The only rows available before the terminal seal are the source rows R, R', threshold rows Q, Q', finite-window rows W, W', and dyadic tolerance ledgers D, D'.

The hypothesis identifies the displayed source, threshold, window, and tolerance rows. By Definition 5543.3, every terminal $\mathsf{Real}^{\uparrow}$ handoff must pass through exactly those finite rows before E or E' can be read.

If a request is strengthened, the continuation may demand a larger finite window, because the stream row is a finite observation schedule. The displayed source, tolerance, threshold, and seal coordinates are still the same coordinates of the carrier, so strengthening cannot replace them by a quotient stream, selected limit, or ambient completeness argument.

Therefore both accepted reads expose the same terminal Real-facing seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, explicit modulus threshold row, finite StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5543.1, Definition 5543.2, and Definition 5543.3 seed the explicit-modulus route from regular rational source data to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegulatedCauchyModulusUp`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, explicit threshold exactness, dyadic tolerance control, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedCauchyMod

$\mathsf{RegularCauchyMinModulus}^{\uparrow}$ routes its hub through the shared modulus meet of chapter 462, the finite dyadic tolerance rows of chapter 291, and the Real-facing handoff through chapter 301 and chapter 38.

Chapter 4729

A Concrete Naming Certificate for $\mathsf{RegularCauchyMinModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ packet records the finite scheduling surface for taking the pointwise minimum of two displayed regular-Cauchy modulus schedules. It sits between the shared-modulus meet packet of chapter 462, the finite stream windows of chapter 126, the regular rational readback boundary of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The downstream selector of chapter 7949 reads the minimum row only after the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ surface and the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison ledger have displayed the common tolerance budget. The packet is a finite L10 Real-and-Completion route: it names the displayed minimum-modulus schedule, not a selected quotient stream, an ambient completeness theorem, or a choice principle over all precision requests.

The minimum-modulus read also depends on the finite-window induction budget of chapter 8091 and the locatedness tolerance budget of chapter 1627. A Real-facing consumer must therefore see the $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ schedule and the $\mathsf{LocatednessModulus}^{\uparrow}$ request row before treating m as a stable minimum for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoffs.

Definition 4729.1 (Regular Cauchy minimum-modulus carrier)**.** *A* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two regular-Cauchy schedules whose finite precision requests are to be synchronized.*
2. *M is the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row that displays the common meet surface inherited from the two source schedules.*
3. *m is the finite minimum row, recording the requestwise smaller admitted threshold after the two source schedules and the meet row have been read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming exactly which source windows are inspected by the minimum schedule.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and the same displayed thresholds.*

6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing denominator budgets, dyadic radii, slack comparisons, and the finite inequalities used by M and m.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S_0, S_1, M, m, W, Q, D have been displayed.*
8. *H stores componentwise* hsame *transport for the two schedules, the meet row, the minimum row, the stream windows, the regular-rational readback, the dyadic ledger, the real seal, the provenance row, and the local name.*
9. *C stores displayed* Cont *replay from a finite precision request through $S_0, S_1, M, m, W, Q, D, E, H$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4729.2 (Regular Cauchy minimum-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed regular-Cauchy schedules, the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row, the finite minimum row, stream windows, regular rational readback, dyadic tolerance ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *meet compatibility, requiring M to be read through the sibling shared-modulus surface of chapter 462 before the minimum row m is consumed;*
3. *finite minimum exactness, requiring each request in m to name the smaller displayed threshold among the two transported source schedules after the dyadic comparisons in D have been read;*
4. *readback stability, requiring the same* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q to be replayed under componentwise* hsame *transport;*
5. *ledger rows for* Cont *replay,* Pkg *provenance, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the local naming certificate.*

No accepted obligation row is stored outside $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 5544.1. The accepted packet displays the two schedules, the meet row, the minimum row, finite windows, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local name.

The meet compatibility row is read before the minimum row because m is meaningful only after the common surface M has synchronized the two source schedules. The sibling packet chapter 462 supplies exactly this shared-modulus boundary.

The finite minimum exactness row is then a dyadic comparison inside D. It chooses no infinite stream representative and invokes no ambient completeness theorem; it records only the displayed smaller threshold for each finite request.

Finally W, Q, E, H, C, P, N replay the same route for stream, regular-rational, and Real-facing consumers. Since none of the carrier coordinates stores quotient stream equality, selected limits,

host real equality, countable choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 4729.3 (Regular Cauchy minimum-modulus L10 route)**.** *Let $R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier. Every L10 Real-and-Completion consumer of the minimum schedule factors through the finite route*

$$S_0, S_1 \longmapsto M \longmapsto m, D \longmapsto W, Q \longmapsto E \longmapsto H, C, P, N.$$

The route first reads the two regular-Cauchy schedules, then the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ *surface, then the finite minimum row with its* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient stream equality, selected limit, hidden completion object, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier and the obligation theorem above. They force every consumer request to retain $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ as one finite packet.

The shared-modulus step reads M before any minimum schedule is used. Thus the consumer is not comparing unrelated streams; it is reading the two source schedules through the same displayed meet surface and the same transported tolerance boundary.

The minimum step reads m together with D. The row D carries the dyadic comparisons and tolerance budgets, while m records only the finite threshold selected by those displayed comparisons.

The stream and regular-rational handoff is then mediated by W and Q, and the terminal seal by E. The structural rows H, C, P, N transport, replay, source, and name the same route. Hence every L10 consumer factors through the displayed sequence, with no quotient stream, selected limit, hidden completion object, countable choice, or ambient completeness coordinate available. □

Theorem 4729.4 (Regular Cauchy minimum-modulus positive gap)**.** *Let $R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier. Whenever the two source modulus schedules display positive dyadic requests through* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, the minimum row m preserves a positive displayed request for the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff. The route also names the* $\mathsf{LocatednessModulus}^{\uparrow}$ *request row and the* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *finite-window budget before any Real-facing consumer reads E.*

Proof. Project the carrier of Definition 5544.1. The two source schedules are read as S_0, S_1, their common meet is M, and the dyadic comparison ledger is D.

Apply the NameCert obligation theorem, Definition 5544.2. Its meet-compatibility and finite-minimum exactness rows require the row m to be read only after M and the displayed dyadic comparisons in D have synchronized the two source requests.

Apply the L10 route theorem, Definition 5544.3. The positive request stored in m is then handed to W, Q, E through the finite sequence $S_0, S_1, M, m, D, W, Q, E, H, C, P, N$, with the sibling $\mathsf{LocatednessModulus}^{\uparrow}$ row and $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ schedule providing the locatedness and finite-window context.

Because the carrier contains no quotient stream, selected limit, hidden completion object, countable-choice selector, or ambient completeness coordinate, the preserved positive gap is only the displayed dyadic request exported by the finite packet. □

Theorem 4729.5 (Regular Cauchy minimum-modulus finite-window induction)**.** *Let*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

be an accepted RegularCauchyMinModulus$^{\uparrow}$ *carrier. Extending both source schedules by one displayed finite* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *window preserves the same minimum-modulus row exactly when the* CauchyModulusMeet$^{\uparrow}$ *row and the* DyadicRatCore$^{\uparrow}$ *tolerance ledger replay through* hsame *and* Cont. *The base read is the carrier route for* S_0, S_1, M, m, W, Q, D, *and the step read uses the meet-compatibility and finite-minimum exactness rows of Definition 5544.2.*

Proof. Proceed by induction on the displayed finite window extension. In the base case, no new window has been appended, so Definition 5544.1 supplies the existing rows $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ and the minimum row is the admitted row of the carrier.

For the induction step, append one source window to each schedule and read it through the displayed StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows W, Q. The meet-compatibility row of Definition 5544.2 forces the appended requests to pass through the same CauchyModulusMeet$^{\uparrow}$ row M before m is consumed.

The finite-minimum exactness row then uses the DyadicRatCore$^{\uparrow}$ ledger D to compare the transported thresholds. Transport through H and replay through C preserve the same displayed comparison data, so the appended window keeps the minimum-modulus row precisely when M and D replay through hsame and Cont.

Applying the induction hypothesis to the earlier windows and the step argument to the appended window yields preservation for the whole finite extension, with no quotient stream equality, countable choice, selected limit, or ambient completeness coordinate. □

Theorem 4729.6 (Regular Cauchy minimum-modulus scoped primitive route)**.** *Let*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

be an accepted RegularCauchyMinModulus$^{\uparrow}$ *carrier. Every minimum-modulus consumer factors through* $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ *together with the* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance rows,* RegularCauchyWindowInduction$^{\uparrow}$ *finite-window schedule,* LocatednessModulus$^{\uparrow}$ *request row, terminal* Real$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route exports no quotient stream equality, selected limit, hidden completion object, countable-choice selector, host real equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Begin with the obligation theorem, Definition 5544.2. It fixes carrier admission, meet compatibility, finite minimum exactness, readback stability, replay, provenance, and local naming as rows of the single finite packet.

Apply the L10 route theorem, Definition 5544.3. Every Real-and-Completion consumer reads the two source schedules, the meet row, the minimum row with dyadic tolerance data, the stream and regular-rational handoff, the real seal, and the structural rows in that order.

Use the positive-gap theorem, Definition 5544.4, to bind the locatedness request row and the finite-window budget to the displayed dyadic request. Then apply the finite-window induction theorem, Definition 5544.5, so extensions of the inspected windows remain within the same M, D, H, C replay surface.

Thus every consumer route is scoped by the named BEDC primitives and sibling packets, and the excluded completion, choice, quotient, and host-equality claims remain outside the accepted RegularCauchyMinModulus$^{\uparrow}$ carrier. □

Falsifiable prediction: An accepted RegularCauchyMinModulus$^{\uparrow}$ consumer that reaches a RegSeqRat$^{\uparrow}$ readback or Real$^{\uparrow}$ seal without reading $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ refutes the carrier surface.

Independence witness: cauchymodulusmeet, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the joint packet containing two regular-Cauchy schedules, a shared meet row, a finite minimum row, stream windows, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ packet is generated from two regular-Cauchy schedules, a CauchyModulusMeet row, a finite minimum row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5544.1, Definition 5544.2, Definition 5544.3, Definition 5544.4, Definition 5544.5, and Definition 5544.6 seed the finite minimum-modulus route from two regular-Cauchy schedules through a shared meet surface to stream, regular-rational, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, quotient stream equality, selected limits, hidden completion objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, meet compatibility, finite minimum exactness, readback stability, and the L10 Real-and-Completion route for BEDC.Derived.RegularCauchyMinModulusUp.

Theorem 4729.7 (Regular Cauchy minimum-modulus CauchyModulus dependency)**.** *Every* RegularCauchyMinMo *consumer reads the request-normalization rows of chapter 292 before it may normalize a source tail request into the minimum-modulus row. The same route reads the regular-sequence handoff of chapter 301 before any* $\mathsf{Real}^{\uparrow}$ *completion-facing seal is consumed.*

Thus the minimum-modulus row depends on $\mathsf{CauchyModulus}^{\uparrow}$ *for source request normalization and on* $\mathsf{RegSeqRat}^{\uparrow}$ *for regular readback, while it supplies no quotient stream equality, selected limit, hidden completion object, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the minimum-modulus request. A consumer cannot form the minimum row until the modulus comparison schedule has displayed how source tail requests are normalized, so the $\mathsf{CauchyModulus}^{\uparrow}$ request-normalization rows are read first.

Next read the regular-sequence handoff. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular readback used by the completion-facing route, and this readback occurs only after the modulus schedule has already fixed the request shape.

Finally inspect the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is downstream of the displayed request schedule and regular readback, it cannot introduce quotient stream equality, a selected limit, a hidden completion object, countable choice, or ambient completeness. □

Theorem 4729.8 (Regular Cauchy minimum-modulus downstream sibling route)**.** *Any* CauchyTailThresholdNorma *consumer reading a normalized shared threshold through chapter 1299 must preserve the* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *row as the finite lower-bound and modulus witness before any* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal read. In particular, the downstream route factors through the same source schedules, meet row, minimum row, finite windows, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows exposed by Definition 5544.9, Definition 5544.10, and Definition 5544.11.*

Proof. Project the carrier of Definition 5544.1. The displayed packet carries $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$, so the minimum row m is read together with the meet, window, regular-rational, dyadic, seal, transport, replay, provenance, and naming rows.

Apply the forward and hub routes. They force downstream consumers to keep the minimum-modulus row downstream of the $\mathsf{CauchyModulusMeet}^{\uparrow}$ row and upstream of the Real-facing readback surface, rather than reading m as an isolated scalar.

Now read the threshold normalizer. Its normalized shared threshold is a Real-facing consumer route, so it must expose the RegularCauchyMinModulus row before passing to the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ handoff or the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the Real dependency route. It already excludes quotient stream equality, selected limits, countable-choice schedules, and ambient Real completeness, so the downstream sibling relation is exactly the finite lower-bound and modulus witness carried by the accepted packet. □

Theorem 4729.9 (Regular Cauchy minimum-modulus forward sibling route)**.** *Every downstream read of* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *by the selector of chapter 7949 must first expose the* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already fixed by this chapter. In carrier notation, the selector may consume the minimum row only after the route*

$$S_0, S_1, M, m, W, Q, D, E, H, C, P, N$$

has displayed M as the meet surface, D as the dyadic tolerance ledger, W as the finite stream-window service, Q as the regular-rational readback row, and E as the Real-facing seal.

Proof. Start with the carrier of Definition 5544.1. It displays $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ as one finite packet, so the downstream selector cannot read m apart from the meet, window, readback, dyadic, seal, transport, replay, provenance, and naming rows.

Apply the NameCert obligations of Definition 5544.2. They force meet compatibility through M, finite minimum exactness through m and D, and readback stability through W and Q before any consumer treats the minimum schedule as accepted data.

Apply the L10 route of Definition 5544.3 and the positive-gap theorem of Definition 5544.4. These results place the displayed minimum before the $\mathsf{Real}^{\uparrow}$ seal while keeping the $\mathsf{LocatednessModulus}^{\uparrow}$ and $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ context visible.

Hence a downstream Cauchy-subsequence selector receives only the already named finite route. It receives no quotient stream equality, selected limit, hidden completion object, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 4729.10 (Regular Cauchy minimum-modulus hub dependency route)**.** *Every accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *meet row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before exporting the minimum-modulus row to any downstream consumer. In carrier notation, the route exposes*

$$S_0, S_1, M, m, W, Q, D, E, H, C, P, N$$

with M read as the meet surface, D read as the dyadic comparison ledger, W read as the finite window service, Q read as the regular-rational readback, and E read as the Real-facing seal.

Proof. Project the carrier of Definition 5544.1. The minimum-modulus row m is not an isolated output; it is displayed inside the finite packet $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$.

Read the sibling route already present in the chapter. The $\mathsf{CauchyModulusMeet}^{\uparrow}$ row M synchronizes the two source schedules before m can be consumed, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the comparison ledger that makes the displayed minimum meaningful.

Now read the Real-and-Completion handoff. The $\mathsf{StreamName}^{\uparrow}$ windows W and $\mathsf{RegSeqRat}^{\uparrow}$ row Q provide the finite readback surface, while E seals the same route for $\mathsf{Real}^{\uparrow}$ consumers. Transport, replay, provenance, and local naming remain attached through H, C, P, N, so the hub dependency route exports only the displayed minimum-modulus packet and no private completion or choice coordinate. □

The Real-facing route for $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ is a finite dependency route through the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, the $\mathsf{StreamName}^{\uparrow}$ finite windows of chapter 126, and the terminal $\mathsf{Real}^{\uparrow}$ sealing surface of chapter 38. A consumer may read the minimum-modulus row only after those displayed rows are present in the same BHist, hsame, Cont, Pkg, and local NameCert route; it receives no quotient stream equality, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Theorem 4729.11 (Regular Cauchy minimum-modulus sibling route)**.** *Every* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *read factors through the* $\mathsf{CauchyModulus}^{\uparrow}$ *carrier of chapter 292 and the* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *normalizer of chapter 778 before exposing the minimum modulus row. The route first displays the source modulus request, then normalizes the regular-Cauchy threshold schedule, and only then reads the finite minimum row together with its* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

Proof. Start with the carrier of Definition 5544.1. The accepted packet displays source schedules, the meet row, the minimum row, stream windows, regular-rational readback, dyadic tolerance data, the Real seal, transport, replay, provenance, and local naming.

Read the $\mathsf{CauchyModulus}^{\uparrow}$ sibling first. It supplies the carrier-level modulus request whose finite thresholds are visible before the minimum row can be inspected.

Next read the $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ sibling. Its normalizer keeps the regular-Cauchy threshold schedule on the same displayed carrier route, so the minimum row is not a detached host-level comparison.

The exposed minimum modulus row is therefore downstream of both sibling packets. Since the route remains inside the displayed BHist, hsame, Cont, Pkg, and local NameCert rows, it exports no quotient completion, selected limit, countable-choice selector, or ambient Real completeness principle. □

Chapter 4730

A Concrete Naming Certificate for $\mathsf{TaylorTheorem}^{\uparrow}$

Following Taylor and the validated-numerics treatment of finite Taylor models, the $\mathsf{TaylorTheorem}^{\uparrow}$ packet records the finite theorem-facing route from displayed derivative data to a polynomial jet and a checked remainder envelope. It sits after the Taylor-model enclosure surface of chapter 442, the derivative observation row of chapter 122, the polynomial carrier of chapter 50, the dyadic rational arithmetic core of chapter 291, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Taylor theorem certificate for one displayed center, order, domain window, derivative ledger, polynomial jet, and remainder bound; it does not import analytic convergence, quotient real equality, selected limits, or a host Taylor series.

Definition 4730.1 (Taylor theorem carrier)**.** *A* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, J, P, R, E, Q, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row naming the center, finite order, domain window, and derivative observations inspected by the packet.*
2. *J is the finite jet ledger. Each coefficient is read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic and the same derivative window named by D.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *row assembled from the displayed finite jet ledger.*
4. *R is the finite remainder-envelope row. It records the derivative-order budget, dyadic interval or ball enclosure, and endpoint window used to bound the residual after P is read.*
5. *E is the* $\mathsf{TaylorModel}^{\uparrow}$ *enclosure row that packages the displayed polynomial part and remainder envelope as a validated finite jet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for polynomial values, derivative observations, and the remainder bound.*
7. *H records componentwise* hsame *transport for derivative, jet, polynomial, remainder, Taylor-model, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every Taylor-theorem read through $D, J, P, R, E, Q, H, G, N$.*

9. *G is the* Pkg *provenance over* $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, P, R, E, Q, H, C, G, N$. It has no coordinate for an infinite Taylor series, analytic convergence, quotient equality of real names, selected limiting values, or an ambient smooth-function theorem.

Theorem 4730.2 (Taylor theorem NameCert obligations)**.** *For every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier*

$$T = (D, J, P, R, E, Q, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed derivative row, finite jet ledger, polynomial row, remainder envelope, Taylor-model enclosure, readback route, transport, replay, provenance, and local naming rows;*
2. *derivative-to-jet routing, requiring every coefficient in J to be read from the finite* $\mathsf{Derivative}^{\uparrow}$ *observations in D and through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic;*
3. *polynomial assembly, requiring P to use exactly the displayed finite coefficient rows in J;*
4. *remainder-envelope exposure, requiring R to bind the next derivative-order budget, the domain window, and the dyadic enclosure before any error bound is exported;*
5. *Taylor-model handoff, requiring E to package only the displayed P and R rows;*
6. *regular and real readback, requiring Q to pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
7. *non-escape from analytic convergence, quotient real equality, selected limits, and host Taylor series semantics.*

No $\mathsf{TaylorTheorem}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.TaylorTheoremUp.TasteGate.TaylorTheoremTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5898.1. The theorem-facing data are exactly D, J, P, R, E, Q, and the structural rows are exactly H, C, G, N.

The derivative-to-jet route is finite. Each coefficient read is a displayed entry of J, and the definition of J attaches that entry to the derivative row D and to dyadic rational arithmetic through $\mathsf{DyadicRatCore}^{\uparrow}$.

Polynomial assembly and remainder exposure are then separate rows. The polynomial row P consumes only the coefficients in J, while the remainder row R consumes the derivative-order budget and domain window before any bound is exported.

Finally the Taylor-model and readback rows seal the route. The row E packages the polynomial and remainder as a finite Taylor-model enclosure, and Q sends the same displayed data through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Since no carrier coordinate names a host Taylor series or a selected limit, the listed rows exhaust the local NameCert surface. □

Theorem 4730.3 (Taylor theorem finite remainder envelope)**.** *Every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier exports a remainder claim only through the displayed route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{TaylorModel}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the derivative row D, finite jet ledger J, polynomial row P, remainder-envelope row R, Taylor-model enclosure E, and regular-rational and real readback row Q before it may cite an error bound for the displayed polynomial approximation. The exported envelope is not an analytic convergence theorem, a quotient equality of real functions, a selected limit, or an ambient Taylor-series object.

Proof. Start with an accepted carrier $T = (D, J, P, R, E, Q, H, C, G, N)$. The only source of derivative information is D, and the only coefficient ledger is the finite row J read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next assemble the approximation. The polynomial row P is built from J, and the remainder row R binds the derivative-order budget, domain window, and dyadic enclosure that measure the residual after that polynomial row has been fixed.

Then pass through the enclosure and readback rows. The Taylor-model row E packages the polynomial and remainder as one finite validated jet, while Q routes the resulting value and bound through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and local naming rows preserve exactly this route. Because the carrier contains no coordinate for analytic convergence, quotient function equality, selected limits, or host Taylor series semantics, every exported remainder envelope is exhausted by the displayed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{TaylorTheorem}^{\uparrow}$ remainder-envelope read factors through a DerivativeUp row, a DyadicRatCore coefficient ledger, a PolynomialUp row, a TaylorModelUp enclosure, a RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: taylormodel, derivative, polynomial, dyadicratcore, regseqrat, real: $\mathsf{TaylorTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite derivative observations, coefficient arithmetic, polynomial assembly, a theorem-facing remainder envelope, Taylor-model enclosure, regular-rational readback, real sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TaylorTheorem}^{\uparrow}$ packet is generated from a DerivativeUp row, a finite DyadicRatCore coefficient ledger, a PolynomialUp row, a finite remainder envelope, a TaylorModelUp enclosure, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5898.1, Definition 5898.2, and Definition 5898.3 seed the finite Taylor theorem route from derivative observations through polynomial jets to displayed remainder envelopes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic convergence, quotient equality of real names, selected limits, host Taylor series semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-jet routing, polynomial assembly, finite remainder-envelope exposure, Taylor-model handoff, regular and real readback, and non-escape for BEDC.Derived.TaylorTheoremUp.

Chapter 4731

A Concrete Naming Certificate for $\mathsf{LagrangeInversion}^{\uparrow}$

Following Lagrange and the classical finite coefficient form of the inversion theorem, the $\mathsf{LagrangeInversion}^{\uparrow}$ packet records the coefficient-reversion route that extracts finitely many inverse-series coefficients from one displayed Taylor jet with a nonzero linear coefficient. It sits after the finite Taylor theorem route of chapter 5898, the derivative observation surface of chapter 122, the polynomial carrier of chapter 50, the dyadic rational arithmetic core of chapter 291, the regular-rational readback rows of chapter 301, and the terminal real-name seals of chapter 38. The packet names a finite inverse-series coefficient extraction certificate; it does not import analytic inverse-function reasoning, a convergent inverse power series, quotient real equality, selected roots, or ambient real completeness.

Definition 4731.1 (Lagrange inversion carrier)**.** *A* $\mathsf{LagrangeInversion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (c, J, a_1, R, B, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *c is the displayed center row, read as a named* $\mathsf{Real}^{\uparrow}$ *endpoint seal together with the finite local window in which the Taylor jet is interpreted.*
2. *J is the finite* $\mathsf{TaylorTheorem}^{\uparrow}$ *jet row. It lists the Taylor coefficients used by the inverse-coefficient extraction and records the associated* $\mathsf{Derivative}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *provenance.*
3. *a_1 is the nonzero linear coefficient witness, represented as a displayed apartness or inverse row over* $\mathsf{DyadicRatCore}^{\uparrow}$ *and sealed by the same* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route as J.*
4. *R is the finite coefficient-reversion ledger. Each inverse coefficient requested by the packet is obtained by a displayed recurrence using only earlier rows of R, the finite jet J, and the linear witness a_1.*
5. *B records dyadic rational coefficient bounds for the finite coefficient-reversion ledger and the polynomial truncations consumed by the packet.*
6. *Q lists the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for the displayed coefficients, the linear witness, the recurrence denominators, and the finite polynomial values.*
7. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals for the center, coefficient reads, and finite inverse-polynomial handoff.*

8. *H records componentwise* hsame *transport for the center, Taylor jet, nonzero linear witness, reversion ledger, coefficient bounds, readback rows, endpoint seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every inverse-coefficient read through $c, J, a_1, R, B, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* LagrangeInversion$^\uparrow$, TaylorTheorem$^\uparrow$, Derivative$^\uparrow$, Poly$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{LagrangeInversion}^\uparrow}$ *row.*

The classifier compares accepted packets only through the displayed rows $c, J, a_1, R, B, Q, E, H, C, P, N$. It has no coordinate for an infinite inverse series, analytic convergence, selected inverse branches, quotient equality of real names, countable choice, or an ambient inverse-function theorem.

Lean4 ✓ `BEDC.Derived.LagrangeInversionUp.LagrangeInversionTasteGate single carrier alignment`

Theorem 4731.2 (Lagrange inversion NameCert obligations)**.** *For every accepted* LagrangeInversion$^\uparrow$ *carrier*

$$L = (c, J, a_1, R, B, Q, E, H, C, P, N),$$

the local NameCert$_{\text{LagrangeInversion}^\uparrow}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed center, finite Taylor jet, nonzero linear coefficient witness, coefficient-reversion ledger, coefficient bounds, readback rows, endpoint seals, transport, replay, provenance, and local naming rows;*
2. *Taylor-jet routing, requiring every source coefficient read to pass through J and the finite* TaylorTheorem$^\uparrow$, Derivative$^\uparrow$, *and* Poly$^\uparrow$ *rows it names;*
3. *linear-coefficient apartness, requiring every division used by the reversion recurrence to cite the displayed nonzero witness a_1 over* DyadicRatCore$^\uparrow$*;*
4. *finite reversion exactness, requiring every row of R to be computed from earlier displayed rows, the finite jet J, and the witness a_1;*
5. *coefficient-bound readback, requiring B and Q to expose dyadic rational bounds and* RegSeqRat$^\uparrow$ *reads before any terminal* Real$^\uparrow$ *seal in E is consumed;*
6. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No LagrangeInversion$^\uparrow$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 4731.1. The theorem-facing rows are exactly c, J, a_1, R, B, Q, E, and the structural rows are exactly H, C, P, N.

The source data are finite. The row J is a TaylorTheorem$^\uparrow$ jet and therefore already factors through derivative observations, polynomial assembly, dyadic coefficient arithmetic, regular rational readback, and real endpoint sealing. The row a_1 is displayed separately, so every recurrence division cites a named nonzero linear coefficient witness.

The coefficient extraction is the finite ledger R. Each row of R reads only earlier inverse-coefficient rows, the listed source coefficients in J, and the dyadic arithmetic attached to a_1; hence no analytic inverse branch or infinite series coordinate is introduced.

Finally B, Q, E bind the numeric output to finite readback. Dyadic coefficient bounds are displayed before regular rational reads, and the terminal Real$^\uparrow$ seals are reached only after those rows are visible. The transport, replay, provenance, and local naming rows preserve this same packet, so the listed obligations exhaust the local NameCert surface. □

Theorem 4731.3 (Lagrange inversion Taylor-polynomial handoff)**.** *Every accepted* $\mathsf{LagrangeInversion}^{\uparrow}$ *carrier exports an inverse-polynomial coefficient read only through the finite route*

$$\mathsf{TaylorTheorem}^{\uparrow} \longmapsto \mathsf{Derivative}^{\uparrow}, \mathsf{Poly}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto R, B \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the displayed center c, finite Taylor jet J, nonzero linear coefficient witness a_1, coefficient-reversion ledger R, dyadic coefficient bounds B, regular rational readbacks Q, endpoint seals E, componentwise hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N before it may consume a finite inverse-polynomial coefficient. The handoff exports coefficient extraction only for the displayed finite truncation, not analytic convergence, selected inverse branches, quotient real equality, countable choice, or ambient real completeness.*

Proof. Start with an accepted carrier $L = (c, J, a_1, R, B, Q, E, H, C, P, N)$. The finite Taylor input is J, whose source dependencies are the Taylor theorem, derivative, polynomial, dyadic rational, regular rational, and real-name rows cited in the carrier.

Next read the nonzero linear coefficient witness a_1. This row is the only source for the divisions in the reversion recurrence, so the consumer cannot replace it by a selected local inverse or by a host inverse-function theorem.

Then consume the finite reversion ledger R and coefficient bounds B. The ledger is finite because it contains only the inverse coefficients requested by the packet, and each displayed row cites earlier ledger entries and the finite source jet. The bounds in B keep the dyadic rational output visible before readback.

Finally pass through Q, E, H, C, P, N. Regular rational readback precedes the terminal real endpoint seals, while transport, replay, provenance, and local naming preserve exactly the displayed route. Since the carrier has no coordinate for analytic convergence, quotient real equality, selected inverse branches, countable choice, or ambient real completeness, every inverse-polynomial coefficient read is exhausted by the finite handoff above. □

Falsifiable prediction: Every accepted $\mathsf{LagrangeInversion}^{\uparrow}$ inverse-coefficient read factors through a displayed center, finite TaylorTheorem jet, Derivative and Polynomial provenance, a nonzero linear coefficient witness, a finite coefficient-reversion ledger, DyadicRatCore coefficient bounds, RegSeqRat readbacks, Real endpoint seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: taylor-theorem, derivative, polynomial, dyadicratcore, regseqrat, real: $\mathsf{LagrangeInversion}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds one center, one finite Taylor jet, a nonzero linear coefficient witness, coefficient-reversion recurrence rows, dyadic coefficient bounds, readback rows, endpoint seals, transport, replay, provenance, and local naming in one inverse-coefficient certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LagrangeInversion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LagrangeInversion}^{\uparrow}$ packet is generated from a displayed center, a finite TaylorTheoremUp jet with DerivativeUp and PolynomialUp provenance, a nonzero linear coefficient witness, a finite coefficient-reversion ledger, DyadicRatCore coefficient bounds, RegSeqRat readbacks, Real endpoint seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LagrangeInversion}^{\uparrow}}$, seed)

Scope closed: Definition 4731.1, Definition 4731.2, and Definition 4731.3 seed the finite inverse-series coefficient extraction route from a displayed Taylor jet to inverse-polynomial coefficient reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LagrangeInversion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close analytic inverse-function reasoning, convergence of an inverse series, selected inverse branches, quotient real equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Taylor-jet routing, nonzero linear coefficient apartness, finite reversion exactness, coefficient-bound readback, Real endpoint sealing, transport and replay stability, and nonescape for BEDC.Derived.LagrangeInversionUp.

Chapter 4732

A Concrete Naming Certificate for $\mathsf{TaylorRemainder}^{\uparrow}$

Following Taylor's theorem and Bishop–Bridges constructive analysis, the $\mathsf{TaylorRemainder}^{\uparrow}$ packet isolates the finite remainder-envelope data used by chapter 5898 and chapter 442. It sits after the derivative observation row of chapter 122, the polynomial carrier of chapter 50, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names only displayed finite derivative windows, polynomial residual rows, dyadic or regular-rational error budgets, readback, transport, replay, provenance, and local naming; it does not import analytic convergence, quotient equality of functions, selected limits, or a host smooth-function theorem.

Definition 4732.1 (Taylor remainder carrier)**.** *A* $\mathsf{TaylorRemainder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, P, W, E, Q, S, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *observation row naming the center, order, domain window, and derivative budget inspected by the remainder read.*
2. *P is the* $\mathsf{Poly}^{\uparrow}$ *row for the finite Taylor polynomial whose residual is being bounded.*
3. *W is the finite remainder-window row. It records the endpoint or interval window, the next derivative-order budget, and the comparison side on which the residual is read.*
4. *E is the error-envelope row, storing the displayed dyadic or regular-rational bound for the residual after P has been fixed.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the bound and polynomial value are exposed as regular rational names.*
6. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the derivative, polynomial, window, envelope, and regular-rational rows have been displayed.*
7. *H records componentwise* hsame *transport for derivative, polynomial, window, envelope, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every remainder-envelope request through $D, P, W, E, Q, S, H, G, N$.*
9. *G is the* Pkg *provenance over* $\mathsf{TaylorRemainder}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{TaylorRemainder}^\uparrow}$ *row.*
The classifier compares accepted packets only through $D, P, W, E, Q, S, H, C, G, N$. It has no coordinate for an infinite Taylor series, analytic convergence, quotient equality of real functions, selected limiting values, or an ambient differentiability theorem.

Lean4 (definition) `BEDC.Derived.TaylorRemainderUp.TaylorRemainderCarrier`

Theorem 4732.2 (Taylor remainder NameCert obligations)**.** *For every accepted* $\mathsf{TaylorRemainder}^\uparrow$ *carrier*

$$R = (D, P, W, E, Q, S, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorRemainder}^\uparrow}$ *surface consists exactly of carrier admission for R, derivative-window exposure through D, polynomial-source exposure through P, finite remainder-window visibility through W, error-envelope accountability through E, regular-rational readback through Q, terminal real sealing through S, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through G, and local naming through N.*

Proof. Project the carrier of Definition 4732.1. The analytic-facing rows are D, P, W, E: a derivative observation, a finite polynomial source, a displayed remainder window, and a residual envelope.

The readback rows are ordered. The envelope row E is read only after the derivative and polynomial rows have fixed the residual being bounded, then Q exposes the regular-rational readback and S supplies the terminal $\mathsf{Real}^\uparrow$ seal.

The structural rows H, C, G, N transport, replay, source, and name exactly those displayed coordinates. Since no carrier coordinate stores an infinite Taylor series, analytic convergence theorem, quotient function equality, selected limit, or host differentiability principle, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.TaylorRemainderUp.TaylorRemainderCarrier namecert obligations`

Theorem 4732.3 (Taylor remainder Real seal handoff)**.** *Every accepted* $\mathsf{TaylorRemainder}^\uparrow$ *remainder-envelope read factors through the finite route*

$$\mathsf{Derivative}^\uparrow \longmapsto \mathsf{Poly}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

as instantiated by D, P, W, E, Q, S. A $\mathsf{TaylorTheorem}^\uparrow$ *or* $\mathsf{TaylorModel}^\uparrow$ *consumer may read the displayed residual bound only after the derivative observation, polynomial source, finite remainder window, error envelope, regular-rational readback, terminal real seal, componentwise transport, continuation replay, provenance, and local certificate rows are present.*

Proof. Start with an accepted carrier $R = (D, P, W, E, Q, S, H, C, G, N)$. The derivative row D fixes the observed order and domain window, while P fixes the polynomial whose residual is being bounded.

The finite remainder window W and envelope E then bind the residual read. They record the endpoint or interval window, the derivative-order budget, and the displayed dyadic or regular-rational error bound before any consumer can cite a theorem-facing error estimate.

Finally Q and S route the same displayed data through regular-rational readback and the terminal Real seal. The transport, replay, provenance, and local naming rows preserve this order, so Taylor-theorem and Taylor-model consumers receive a finite remainder packet rather than analytic convergence, quotient equality, a selected limit, or an ambient smooth-function theorem. □

Falsifiable prediction: Every accepted $\mathsf{TaylorRemainder}^{\uparrow}$ remainder-envelope read factors through one DerivativeUp row, one PolynomialUp row, one finite remainder-window row, one error-envelope row, one RegSeqRatUp readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: taylor-theorem, taylormodel, derivative, polynomial, regseqrat, real: $\mathsf{TaylorRemainder}^{\uparrow}$ is not bijective to any listed sibling because it isolates the theorem-facing residual window, error-envelope row, regular-rational readback, Real seal, transport, replay, provenance, and local naming as one finite certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorRemainder}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TaylorRemainder}^{\uparrow}$ packet is generated from a DerivativeUp observation row, a PolynomialUp source row, a finite remainder window, a displayed error envelope, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorRemainder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4732.1, Definition 4732.2, and Definition 4732.3 seed the finite Taylor residual route from derivative observations and polynomial sources to regular-rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaylorRemainder}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close analytic convergence, quotient equality of real functions, selected limits, ambient differentiability, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exposure, polynomial-source exposure, finite remainder-window visibility, error-envelope accountability, regular and real readback, transport, replay, provenance, and local naming for BEDC.Derived.TaylorRemainderUp.

The $\mathsf{ApproximationTower}^{\uparrow}$ hub routes finite level ledgers through chapter 126, chapter 301, chapter 291, and chapter 38 before the child files unpack carrier and obligation rows.

Chapter 4733

A Concrete Naming Certificate for $\mathsf{ApproximationTower}^{\uparrow}$

The $\mathsf{ApproximationTower}^{\uparrow}$ packet realizes the finite level-and-window tower of chapter 9414 as BEDC NameCert data. It sits beside the finite stream observations of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance carrier of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Those siblings provide windows, readback, tolerance, and sealing rows; this packet binds them into one finite approximation tower without adding a hidden limit selector, quotient stream, ambient completeness theorem, or physical-adequacy row.

Definition 4733.1 (ApproximationTower carrier)**.** *An* $\mathsf{ApproximationTower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (L, W, D, R, E, H, C, P, N)$$

with the following visible rows:

1. *L is the finite level ledger naming the approximation stages consumed by the tower.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row recording the finite observations available at each level.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row assigning displayed dyadic error bounds to the levels.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that compares successive level observations through regular rational data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, dyadic tolerances, and regular readback rows have been displayed.*
6. *H records componentwise* hsame *transport for L, W, D, R, E and the structural rows.*
7. *C records displayed* Cont *replay of the level ledger, finite windows, dyadic tolerances, regular readback, and real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{ApproximationTower}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden source, quotient stream equality, selected limit point, ambient completeness theorem, or physical-adequacy classifier.

Lean4 (statement) `BEDC.Derived.ApproximationTowerUp`

Theorem 4733.2 (ApproximationTower finite NameCert surface)**.** *For every accepted* $\mathsf{ApproximationTower}^{\uparrow}$ *carrier* $A = (L, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite level ledger, finite window exposure through* W*, dyadic tolerance exposure through* D*, regular rational readback through* R*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

No accepted read exports a hidden limit selector, quotient stream, unbounded level family, ambient real-completeness row, or physical-adequacy classifier.

Proof. Project the carrier of Definition 4733.1. Its analytic rows are L, W, D, R, E, and its structural rows are H, C, P, N.

The level ledger L fixes the finite tower shape before any observation is read. The window row W then exposes the finite stream observations at those levels, and the dyadic row D supplies the displayed tolerance bounds attached to the same levels.

The regular readback row R compares the displayed windows through rational data, after which the real seal E may be consumed. Transport, replay, provenance, and local naming preserve only this route through H, C, P, N.

Since the carrier has no coordinate for a hidden source, quotient stream, selected limit, ambient real completeness, or physical-adequacy classifier, the listed rows exhaust the local NameCert surface. □

Theorem 4733.3 (ApproximationTower L10 sibling route)**.** *Every accepted* $\mathsf{ApproximationTower}^{\uparrow}$ *read factors through the concrete L10 sibling rows* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *before any tower level can be consumed. For a carrier* $A = (L, W, D, R, E, H, C, P, N)$*, the level ledger* L *is readable only with the finite window row* W*, dyadic tolerance row* D*, regular rational readback row* R*, terminal real seal* E*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*, package provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}$ *row* N*.*

The vision reference motivates the packet, but it is not a substitute for these concrete-instance dependencies. No accepted read may bypass the StreamName, RegSeqRat, DyadicRatCore, and Real rows by appealing to a hidden limit selector, quotient stream, unbounded level family, ambient completeness theorem, or physical-adequacy classifier.

Proof. Project the carrier of Definition 4733.1. The rows $L, W, D, R, E, H, C, P, N$ are the complete displayed packet, with no hidden coordinate outside the finite level, window, tolerance, readback, seal, transport, replay, provenance, and naming surface.

The finite NameCert surface of Definition 4733.2 identifies W as the $\mathsf{StreamName}^{\uparrow}$ route, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance route, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback route, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal. A tower level is therefore visible only after those sibling rows have been read.

Finally the structural rows H, C, P, N preserve that same route for downstream consumers. The source vision supplies orientation for the packet, but it does not replace the concrete sibling rows; hence any read that skips $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ leaves the accepted $\mathsf{ApproximationTower}^{\uparrow}$ surface. □

Theorem 4733.4 (ApproximationTower obligation closure)**.** *For every accepted* $\mathsf{ApproximationTower}^{\uparrow}$ *carrier* $A = (L, W, D, R, E, H, C, P, N)$*, the packet supplies the five required NameCert obligation rows for the tower route: carrier admission for finite levels, window exposure through* $\mathsf{StreamName}^{\uparrow}$*, dyadic tolerance exactness through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal handoff, and structural transport, replay, provenance, and naming nonescape through* H, C, P, N*.*

The obligations are exactly the visible rows of the carrier. They do not add a hidden source, quotient stream equality, selected limit point, ambient completeness theorem, physical-adequacy classifier, or external approximation oracle.

Proof. Project the carrier of Definition 4733.1. The finite level ledger L supplies carrier admission for displayed stages, and the window row W exposes only the finite $\mathsf{StreamName}^{\uparrow}$ observations available at those stages.

The dyadic row D supplies tolerance exactness, while R supplies the regular rational readback attached to the same displayed windows. The real seal E is downstream from those rows by Definition 4733.3.

The structural rows H, C, P, N then transport, replay, source, and name the same finite surface. By Definition 4733.2, the carrier has no additional coordinate, so the listed obligations exhaust the accepted $\mathsf{ApproximationTower}^{\uparrow}$ packet. □

Falsifiable prediction: Every accepted $\mathsf{ApproximationTower}^{\uparrow}$ read exposes finite level admission, StreamNameUp windows, DyadicRatCoreUp tolerances, RegSeqRatUp readback, RealUp sealing, transport, replay, provenance, and local NameCert before any tower level is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real, approximationtowerresidue: $\mathsf{ApproximationTower}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite level admission, windows, tolerances, readback, Real sealing, transport, replay, provenance, and local naming as one tower packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}$.

Constructive story: An $\mathsf{ApproximationTower}^{\uparrow}$ packet is generated from a finite level ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4733.1 and Definition 4733.2 seed the finite approximation tower over level, window, dyadic tolerance, regular readback, real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not provide checked Lean obligations, hidden-source recovery, quotient stream equality, ambient completeness, physical adequacy, or a selected limit for $\mathsf{ApproximationTower}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, regular readback, real sealing, transport, replay, provenance, and local naming for BEDC.Derived.ApproximationTowerUp.

Theorem 4733.5 (ApproximationTower Real consumer route)**.** *Every accepted* $\mathsf{ApproximationTower}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must first display the finite level ledger, the* $\mathsf{StreamName}^{\uparrow}$ *window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ApproximationTower}^{\uparrow}}$ *row. The route is the concrete L10 completion path through chapter 126, chapter 291, chapter 301, and chapter 38.*

Proof. Project the accepted carrier $A = (L, W, D, R, E, H, C, P, N)$ of Definition 4733.1. The row L is the only finite level ledger, W is the only displayed stream-window source, D is the only dyadic tolerance surface, R is the only regular rational readback row, and E is the only terminal real seal.

Apply the finite NameCert surface of Definition 4733.2. It forces the analytic rows to be read with the structural rows H, C, P, N, so a Real-facing consumer receives transport, replay, provenance, and local naming before it can consume E.

Finally use the L10 sibling route of Definition 4733.3. The route anchors the window, tolerance, readback, and seal rows in the StreamName, DyadicRatCore, RegSeqRat, and Real chapters named above. Thus the accepted Real consumer cannot bypass the finite tower route through a hidden source, quotient stream equality, selected limit point, ambient completeness theorem, physical-adequacy classifier, or external approximation oracle. □

The sibling route makes the Real-facing use of $\mathsf{ApproximationTower}^{\uparrow}$ explicit: the tower seal is never read as a kernel-only artifact, but through the same L10 rows named by the carrier and consumed by the Real route.

Theorem 4733.6 (ApproximationTower sibling dependency route)**.** *Every* $\mathsf{ApproximationTower}^{\uparrow}$ *Real-facing consumer first reads the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *sibling route already named by Definition 4733.1 before it reads the tower seal. In particular, the carrier rows* D, W, R, E *are exposed as dyadic tolerance, stream window, regular rational readback, and terminal real seal data before the structural rows* H, C, P, N *package the tower for downstream use.*

Proof. Project the accepted carrier $A = (L, W, D, R, E, H, C, P, N)$ of Definition 4733.1. The rows D, W, R, E are named there as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and $\mathsf{Real}^{\uparrow}$ seal row.

Apply Definition 4733.5. A Real-facing consumer reaches the terminal seal only after the finite level ledger, the four sibling rows, and the structural rows have been displayed.

Thus the tower seal is downstream from the dyadic tolerance, stream window, regular readback, and Real seal route already carried by A. The rows H, C, P, N preserve and name that route; they do not introduce a separate kernel-only consumption path. □

Chapter 4734

A Concrete Naming Certificate for $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$

Following Bishop–Bridges constructive completion, the $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$ packet records the finite NameCert surface for an Archimedean ordered Cauchy completion built from displayed dyadic rational data. It sits as an L10 real-completion neighbour of the dyadic carrier of chapter 291, the finite stream-name interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the constructive order handoff of section 38.3. The packet binds density, Archimedean order, regular-Cauchy sealing, and RealUp readback in one finite carrier; it does not import quotient completion, selected limits, trichotomy, countable choice, host real equality, or ambient completeness.

Definition 4734.1 (ArchimedeanOrderedCauchyCompletion carrier)**.** *An* $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, Q, R, O, L, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row carrying dyadic rational observations, signed arithmetic, tolerance comparison, and density requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing finite observation windows for completion names.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row reading regular Cauchy rational sequences through the same dyadic tolerances.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
5. *O is the* $\mathsf{RealAlgOrder}^{\uparrow}$ *order-and-apartness handoff row for the sealed real names.*
6. *L is the Archimedean density ledger: every displayed positive tolerance request is answered by a dyadic row of D before the corresponding stream window and regular-readback row are consumed.*
7. *H records componentwise* hsame *transport for D, S, Q, R, O, L.*
8. *C records displayed* Cont *replay of the same completion route.*
9. *P is the* Pkg *provenance over* $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealAlgOrder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient set of Cauchy sequences, a selected representative, a global order decision, countable choice, or a separate completeness oracle.

Lean4 ✓ BEDC.Derived.ArchimedeanOrderedCauchyCompletionUp.ArchimedeanOrderedCauchyCompletionTasteGate single carrier alignment

Theorem 4734.2 (ArchimedeanOrderedCauchyCompletion NameCert obligations)**.** *For every accepted* $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$ *carrier* $A = (D, S, Q, R, O, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for dyadic rational observations, finite stream windows, regular-Cauchy rational readback, terminal real sealing, constructive order and apartness, Archimedean density, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing read must pass through* $D, S, Q, R, O, L, H, C, P, N$*, and no obligation may be discharged by quotient completion, host equality of reals, trichotomy, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 4734.1. The rows D, S, Q form the finite completion spine: dyadic tolerances are displayed first, finite observation windows are then opened, and regular rational readback is admitted only through those windows.

The seal row R is reached after that spine, and the order row O reads the sealed names through the constructive order interface. Thus the order surface is downstream of the same regular-Cauchy route rather than an independent host order predicate.

The density ledger L answers each positive tolerance request with a dyadic row before the stream and readback rows are consumed. Finally H, C, P, N transport, replay, source, and name the same finite packet. Since these are all carrier coordinates, the excluded quotient, selector, trichotomy, choice, host-equality, and ambient-completeness routes have no entry point. □

Theorem 4734.3 (ArchimedeanOrderedCauchyCompletion density seal)**.** *Every downstream consumer of an accepted* $\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}$ *packet receives density and order only through the finite handoff*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealAlgOrder}^{\uparrow}.$$

Consequently an Archimedean approximation, an order comparison, or a regular-Cauchy completion read can use only the displayed dyadic tolerance rows, finite windows, regular rational readbacks, RealUp seal, order handoff, transports, continuations, provenance, and local naming rows.

Proof. Begin with Definition 4734.2. It forces every completion-facing read through the dyadic row, stream-window row, regular-readback row, real seal, and order row.

The Archimedean density ledger is attached to the same dyadic row. A consumer asking for an approximation at a positive tolerance therefore reads L through D before any stream window, regular rational name, real seal, or order comparison becomes available.

The structural rows H, C, P, N preserve this route under transport and replay. They can expose the handoff as a named packet, but they cannot add a quotient representative, selected limit, trichotomy branch, countable-choice schedule, host equality of real names, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}}$ **·**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanOrderedCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4734.1, Definition 4734.2, and Definition 4734.3 seed the finite Archimedean ordered Cauchy completion route through DyadicRatCore, StreamName, RegSeqRat, RealUp, RealAlgOrder, density, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{ArchimedeanOrderedCauchyCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected Cauchy representatives, trichotomy, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic density, finite-window readback, regular-Cauchy sealing, RealUp handoff, RealAlgOrder order exposure, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.ArchimedeanOrderedCauchyCompletionUp.

Constructive story: An ArchimedeanOrderedCauchyCompletion$^{\uparrow}$ packet is generated from dyadic rational observations, finite stream windows, regular Cauchy rational readback, a terminal RealUp seal, constructive order and apartness rows, an Archimedean density ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4735

A Concrete Naming Certificate for $\mathsf{ClaimStatusAudit}^{\uparrow}$

The $\mathsf{ClaimStatusAudit}^{\uparrow}$ packet realizes the legitimate-claim audit of chapter 9456 as a finite NameCert carrier for theorem, condition, route, evidence, non-evidence, meta, refusal, GAP, and socket readings. It sits beside the named externality discipline of chapter 543, the claim-row admission surface of chapter 1209, the registry export boundary of chapter 1216, and the theorem-gap registry of chapter 1246. Those siblings name refused supply, admit claim entries, expose export rows, and bind theorem/gap entries; this packet audits their status rows before any claim is read as stronger than its displayed support.

Definition 4735.1 (ClaimStatusAudit carrier)**.** *A* $\mathsf{ClaimStatusAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, T, R, V, G, S, A, X, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite communicable expression row being audited.*
2. *T is the theorem-or-condition row when the expression has closed proof data or displayed hypotheses.*
3. *R is the route row, recording transports, equivalence chains, or proof-search paths without upgrading them to theorem content.*
4. *V is the evidence row, paired with any non-evidence refusal attached to reports, metrics, traces, or numerical data.*
5. *G is the* $\mathsf{TheoremGapRegistry}^{\uparrow}$ *row recording theorem-strength entries and* GAP *entries.*
6. *S is the socket row read through* $\mathsf{ApophaticName}^{\uparrow}$ *when requested supply is named but not internalized.*
7. *A is the* $\mathsf{ClaimRegistryLayer}^{\uparrow}$ *admission row for the audited claim entry.*
8. *X is the* $\mathsf{RegistryExportSurface}^{\uparrow}$ *row controlling whether the audited status may be exported.*
9. *H records componentwise* hsame *transport for E, T, R, V, G, S, A, X.*
10. *C records displayed* Cont *replay of the same audit route.*
11. *P is the* Pkg *provenance over* $\mathsf{ClaimStatusAudit}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{TheoremGapRegistry}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{ClaimStatusAudit}^{\uparrow}}$ *row.*
The carrier has no coordinate for a global truth predicate, a hidden proof oracle, an unmarked external dependency, or an export permission stronger than the displayed status row.

Lean4 ✓ `BEDC.Derived.ClaimStatusAuditUp.ClaimStatusAuditTasteGate single carrier alignment`

Theorem 4735.2 (ClaimStatusAudit NameCert obligations)**.** *For every accepted* $\mathsf{ClaimStatusAudit}^{\uparrow}$ *carrier* $U = (E, T, R, V, G, S, A, X, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClaimStatusAudit}^{\uparrow}}$ *surface consists exactly of expression admission, theorem-or-condition classification, route classification, evidence and non-evidence classification, theorem-gap registry reading, apophatic socket reading, claim-registry admission, registry-export exposure, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. No accepted audit may read a route row as a theorem row, an evidence row as proof, a socket row as supplied content, or an unexported registry row as public claim content.*

Proof. Project the carrier of Definition 4735.1. The expression row E is classified only through T, R, V, G, S, A, X before the structural rows are read.

The theorem-or-condition row T is separate from the route row R and the evidence row V. Therefore a route, trace, metric, or report can be preserved as admissible audit data without being retyped as a closed theorem.

The gap and socket rows G, S are read through the theorem-gap registry and apophatic naming surfaces. A missing dependency is thereby named as a boundary rather than consumed as internal proof content.

Finally A, X, H, C, P, N admit, export, transport, replay, source, and name the same status packet. Since the export row is displayed, a consumer cannot bypass it; since the socket row is apophatic, the far end cannot become supplied content inside the carrier. □

Lean4 ✓ `BEDC.Derived.ClaimStatusAuditUp.ClaimStatusAuditCarrier namecert obligations`

Theorem 4735.3 (ClaimStatusAudit nonescape)**.** *Every downstream reading of an accepted* $\mathsf{ClaimStatusAudit}^{\uparrow}$ *packet factors through the finite audit route*

$$\mathsf{ApophaticName}^{\uparrow} \longmapsto \mathsf{ClaimRegistryLayer}^{\uparrow} \longmapsto \mathsf{TheoremGapRegistry}^{\uparrow} \longmapsto \mathsf{RegistryExportSurface}^{\uparrow}.$$

Thus theorem, route, evidence, refusal, GAP*, and socket readings remain status rows of the same carrier and cannot escape into unmarked theorem content or unrestricted public export.*

Proof. Use Definition 4735.2. It identifies the rows through which every accepted claim-status read must pass.

The apophatic-name row handles missing or refused supply as a named boundary. The claim-registry row admits the expression as a claim entry, while the theorem-gap row distinguishes theorem-strength support from gap exposure.

The registry-export surface is terminal for public reading. If it does not display the appropriate export status, the carrier still records the audit row but does not provide public theorem content. The structural rows preserve this nonescape condition under transport and replay. □

Closure status for $\mathsf{NameCert}_{\mathsf{ClaimStatusAudit}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClaimStatusAudit}^{\uparrow}}, \mathsf{seed})$
Scope closed: Definition 4735.1, Definition 4735.2, and Definition 4735.3 seed the finite claim-status audit route through apophatic names, claim registry rows, theorem-gap rows, registry export, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ClaimStatusAudit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a global truth predicate, global proof search, hidden proof oracle, unmarked external supply, unrestricted registry export, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, status classification, route/evidence separation, theorem-gap reading, apophatic socket reading, registry admission, export control, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.ClaimStatusAuditUp.

Constructive story: A ClaimStatusAudit$^{\uparrow}$ packet is generated from a finite expression row, theorem-or-condition row, route row, evidence and non-evidence row, theorem-gap registry row, apophatic socket row, claim-registry admission, registry-export exposure, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The LocatedRealIntervalBisection$^{\uparrow}$ hub routes finite bisection reads through chapter 394 and through the Real, Dyadic, and RegSeqRat surfaces carried by the child chapter.

Chapter 4736

Located Real Interval Bisection NameCert Construction

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ packet records finite dyadic interval bisection over displayed $\mathsf{Real}^{\uparrow}$ endpoints without importing an interval-completeness theorem. It sits beside the real-name seal of chapter 38, the dyadic arithmetic core of chapter 291, the regular rational readback route of chapter 301, and the closed bounded interval carrier of chapter 1591. The packet is a finite bisection ledger for located interval reads; it is not a Dedekind completion, quotient-real equality, trichotomy on arbitrary reals, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4736.1 (Located real interval bisection carrier)**.** *A* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, endpoint order, and interval-containment data.*
2. *L is the locatedness row, recording the finite apartness or locator evidence used to decide which inspected subinterval remains visible at the requested stage.*
3. *M is the dyadic midpoint row, exposing the midpoint construction for each retained interval cell.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger attached to the same midpoint rows.*
5. *W is the finite endpoint-window row, recording which endpoint and midpoint windows are inspected by the bisection request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the source interval, locatedness, midpoint, dyadic, window, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the interval, locatedness, midpoint, dyadic, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through I, L, M, D, W, R, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, M, D, W, R, E, H, C, P, N$. It has no coordinate for a completed nested-interval limit, quotient equality of real names, excluded-middle order trichotomy, a selected infinite branch, countable choice, or ambient real completeness.

Theorem 4736.2 (Located real interval bisection obligation surface)**.** *For every accepted* LocatedRealIntervalBisecti *carrier*

$$B = (I, L, M, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded interval source, locatedness row, midpoint row, dyadic tolerance ledger, endpoint windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming;*
2. *source visibility, requiring every bisection read to expose the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row I before any midpoint, diameter, branch, or seal row is consumed;*
3. *located branch discipline, requiring L to record the finite locator evidence for the retained cell without adding global trichotomy on* $\mathsf{Real}^{\uparrow}$*;*
4. *dyadic midpoint exactness, requiring M and D to supply the displayed midpoint and diameter budgets at the requested finite depth;*
5. *readback handoff, requiring W, R, E to route every Real-facing output through finite endpoint windows and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
6. *non-escape, excluding completed nested-interval limits, quotient-real equality, selected infinite branches, countable choice, host endpoint equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the carrier.*

No LocatedRealIntervalBisection obligation is stored outside these rows.

Proof. Project the carrier of Definition 5546.1. The accepted packet displays exactly the closed bounded interval source, locator row, midpoint row, dyadic ledger, endpoint windows, regular readback, terminal real seal, and structural rows.

The source stage is I, L. A request first reads the displayed endpoints and interval containment through I, then reads the finite locator evidence in L before any retained child interval is available.

The bisection stage is M, D, W. Induction on the finite retained-depth ledger reads one midpoint row at a time. At each step the dyadic tolerance entries in D bound the retained child, while W records the inspected endpoint and midpoint windows.

The Real-facing handoff is R, E, H, C, P, N. The regular rational readback R is computed from the displayed windows and dyadic budgets, and the terminal seal E is downstream from that readback. The structural rows only transport, replay, source, and name the same finite route.

Since the carrier contains no completed nested-interval limit, quotient-real equality, global trichotomy, selected infinite branch, countable-choice row, or ambient completeness coordinate, the listed rows exhaust the local obligation surface. □

Theorem 4736.3 (Located real interval bisection finite handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto L, M, D, W \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives one finite located interval bisection ledger with dyadic midpoint and diameter rows. It receives no completed intersection point, quotient representative, selected branch, countable-choice witness, or ambient completeness principle.

Proof. Use the carrier projection and the obligation surface of Definition 5546.2. Source visibility places every read below the row I before midpoint or readback data are available.

Located branch discipline then exposes only the finite locator row L. Dyadic midpoint exactness exposes M, D, W for the requested depth, so the retained cell is read as a finite ledger rather than as an infinite branch.

Finally R repacks the displayed windows as regular rational readback data, and E, H, C, P, N seal and name the same route for $\mathsf{Real}^{\uparrow}$ consumers. The absent coordinates listed in the carrier definition cannot be projected from the packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ Real-facing read passes through $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, finite endpoint windows, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any retained interval output is visible.

Independence witness: real, dyadicratcore, regseqrat, closedboundedinterval, located-interval-bisection, bisection-real: no listed sibling supplies the joint packet containing a closed bounded Real interval source, locator evidence, dyadic midpoint ledger, finite endpoint windows, regular rational readback, Real seal, transport, replay, provenance, and local NameCert row while refusing completeness and selected infinite branches.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5546.1, Definition 5546.2, and Definition 5546.3 seed the finite located bisection route over Real endpoints, closed bounded interval source rows, dyadic midpoint budgets, regular rational readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.LocatedRealIntervalBisectionUp`

Bridge status: none

Not claimed: This chapter does not close completed nested-interval limits, quotient-real equality, global trichotomy on reals, selected infinite branches, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source visibility, located branch discipline, dyadic midpoint exactness, readback handoff, and non-escape for BEDC.Derived.LocatedRealIntervalBisectionUp.

Constructive story: A $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, locatedness evidence, dyadic midpoint and diameter rows, finite endpoint windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The handoff hub routes candidate-normalization and confluence reads through chapter 4238 and chapter 765 before exposing bounded endpoint or residual-envelope rows.

Chapter 4737

MetaCIC Candidate Normalization Confluence Handoff NameCert Construction

The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization boundary of chapter 9383 as a finite handoff surface for confluence and decidability consumers. It sits beside the critical-path carrier of chapter 765, the normalization frontier of chapter 892, the residual confluence envelope of chapter 3921, and the decidable boundary of chapter 749. The packet names a guarded route from candidate evidence to bounded normal endpoints and residual-envelope reads; it does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, or global closed-term conversion decidability.

Candidate-mediated normalization reads the residual local-join budget of chapter 5970 before it can be used as a confluence handoff, so the endpoint row is never exported as an isolated normal-form claim.

This packet is also a concrete realization of the MetaCIC audit-map synthesis route in section 9376.7: it joins normalization, confluence, decidability, and residual-substitution audit rows while keeping blocked subject-reduction edges visible.

The confluence audit packet of chapter 735 and the parallel-diamond frontier of chapter 4238 consume this handoff only as a guarded boundary route. Their reads keep candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention explicit rather than turning the handoff into a global Church–Rosser or conversion theorem.

Before exporting candidate-normalization confluence data, the handoff consumes the residual-diamond sibling surface of chapter 4238 and the critical-path sibling surface of chapter 765. The handoff is consumed by chapter 4238 only after candidate evidence, residual envelope, decidable boundary, and blocked-edge rows are visible.

Theorem 4737.1 (MetaCIC handoff parallel-diamond forward link)**.** *Any* MetaCICParallelDiamondFrontier$^{\uparrow}$ *consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must expose the candidate row K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, componentwise transport T, displayed replay R, provenance P, and local name L before reading the bounded endpoint row N. The forward dependency is the sibling route through chapter 4238; it is not a global confluence theorem, quotient-syntax equality, unrestricted subject reduction, evaluator soundness, or closed consistency.*

Proof. Start with a parallel-diamond consumer of the handoff. The consumer reaches the packet

only through the displayed candidate-normalization carrier, so the candidate row K and frontier row F remain visible before any bounded endpoint can be read.

Next read the confluence side through the residual envelope C, decidable boundary D, and blocked edge B. These rows are precisely the finite obstruction and boundary data named by the parallel-diamond frontier route, and they prevent N from being treated as an unconditional joinability endpoint.

Finally transport the same route through T, R, P, L. The componentwise transport, replay, provenance, and local name rows bind the endpoint read to the concrete sibling of chapter 4238. Hence every forward read exposes $K, F, C, D, B, T, R, P, L$ before N, while the excluded global MetaCIC claims remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffParallelDiamondForwardLink`

Before exporting candidate-normalization confluence data, the handoff consumes residual-normal-form routing from chapter 8456 and residual critical-pair sieve rows from chapter 8316.

Definition 4737.2 (MetaCIC candidate normalization confluence handoff carrier)**.** *A* MetacicCandidateNormalizati *carrier is a finite* BHist *packet*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

with the following displayed rows:

1. *A is the audit-map anchor row, binding the packet to the candidate-mediated reading of chapter 9383.*
2. *K is the candidate-evidence row, storing Candidate, ClosedCandidate, or CandidateSet membership data before any strong-normalization read is exported.*
3. *N is the normal-endpoint row, recording the bounded normal endpoint or no-infinite-reduction projection obtained from K.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row that keeps candidate evidence and finished-normal evidence inside the displayed frontier packet.*
5. *C is the confluence-facing row, read through the residual envelope of chapter 3921.*
6. *D is the* MetaCICDecidableBoundary$^{\uparrow}$ *row, limiting decidability reads to bounded checker output and normal endpoint comparison.*
7. *B is the blocked-edge row, retaining subject-reduction and residual-substitution obligations as visible boundaries rather than discharged theorems.*
8. *T records componentwise* hsame *transport for the audit, candidate, normal endpoint, frontier, confluence, decidability, blocked-edge, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from a MetaCIC consumer request through A, K, N, F, C, D, B, T.*
10. *P is the* Pkg *provenance over* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICConfluenceAuditWitness$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, K, N, F, C, D, B, T, R, P, L$. It has no coordinate for evidence-free normalization, a global parallel diamond, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf
candidate endpoint route

Theorem 4737.3 (MetaCIC candidate normalization confluence handoff boundary)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the audit anchor, candidate evidence, normal endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked-edge row, transport, replay, provenance, and local naming;*
2. *candidate-gate exactness, requiring every normalization read to pass through K and the frontier row F before N is consumed;*
3. *confluence handoff discipline, requiring every residual or join read to pass through C while retaining B as a visible blocked edge;*
4. *decidable-boundary discipline, requiring every conversion-facing read to pass through D after N and before any bounded normal comparison is exported;*
5. *frontier non-collapse, requiring T, R, P, L to preserve the same rows without promoting candidate-mediated SN into an unconditional closed-term theorem or residual-envelope rows into full Church–Rosser.*

No handoff obligation is stored outside $A, K, N, F, C, D, B, T, R, P, L$.

Proof. Project the carrier of Definition 5547.2. The accepted packet displays exactly the audit anchor, candidate evidence, normal endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked-edge, and structural rows.

The normalization stage is A, K, N, F. The audit anchor fixes the candidate-mediated reading; the candidate row K is exposed before the endpoint row N, and the frontier row F keeps both inside the public normalization boundary.

The confluence and decidability stage is C, D, B. A confluence-facing consumer may read residual-envelope data through C, but the blocked-edge row B remains visible. A conversion-facing consumer may read bounded checker output through D only after the normal endpoint has been displayed.

Finally T, R, P, L transport, replay, source, and name the same finite route. Since the carrier contains no evidence-free normalization coordinate, global diamond theorem, unrestricted subject-reduction proof, quotient syntax equality, host evaluator soundness, or closed-consistency endpoint, the listed rows exhaust the local obligation surface. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffBoundary

Theorem 4737.4 (MetaCIC candidate normalization confluence handoff seed closure)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier* $H = (A, K, N, F, C, D, B, T, R, P, L)$*, the seed package closes only when the candidate-evidence row K, normalization-frontier row F, residual confluence row C, decidable-boundary row D, and blocked-edge row B are all displayed before the endpoint row N is consumed. The support rows T, R, P, L preserve this finite route by transport, replay, provenance, and local naming. The package does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffSeedClosure`

Proof. Project the carrier of Definition 5547.2. The accepted packet separates candidate evidence K, endpoint data N, frontier data F, residual confluence data C, decidable-boundary data D, blocked-edge data B, and the structural rows T, R, P, L.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, its confluence handoff discipline keeps C and B visible for residual or join reads, and its decidable-boundary discipline places D on the bounded comparison route.

Finally read T, R, P, L only as preservation rows for the same finite package. They transport, replay, source, and name the displayed rows without adding a coordinate for the excluded global endpoints, so the seed closure remains the finite premise-exposure package rather than a global normalization, confluence, decidability, quotient, evaluator, or consistency theorem. □

Theorem 4737.5 (MetaCIC candidate normalization confluence handoff premise exposure)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier* $H = (A, K, N, F, C, D, B, T, R, P, L)$*, every confluence-facing read must expose the candidate row* K*, normalization frontier row* F*, residual envelope row* C*, decidable-boundary row* D*, and blocked-edge row* B *before any bounded endpoint row* N *can be consumed. Thus candidate-mediated normalization is retained as a premise surface for confluence consumers. The handoff refuses unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, and closed consistency.*

Proof. Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, while its confluence handoff discipline keeps C and B visible for every residual or join read.

Now chain the consumer boundary of Definition 5547.14. A public confluence-facing consumer receives the handoff only through the displayed route

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto C, D, B \longmapsto T, R, P, L.$$

The decidable-boundary row D is therefore part of the bounded comparison surface, not a detached global conversion decider.

Since T, R, P, L only transport, replay, source, and name the same finite route, they cannot erase K, F, C, D, B or promote N into an unconditional endpoint theorem. The excluded global claims are not coordinates of H, so every confluence-facing read remains premise-exposed before consuming bounded endpoint data. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffPremiseExposure`

Theorem 4737.6 (MetaCIC candidate handoff residual-diamond retention)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

every confluence-facing read must retain the candidate evidence K*, residual envelope* C*, decidable boundary* D*, and blocked-edge row* B *before a bounded endpoint row* N *is treated as joinable. The retained route is consumed through the sibling path from* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *to* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *and remains only a residual-diamond handoff. It does not assert full Church–Rosser, unrestricted subject reduction, quotient syntax equality, or global conversion decidability.*

Proof. Project the carrier of Definition 5547.2. The endpoint row N is separate from the candidate evidence K, residual envelope C, decidable boundary D, and blocked-edge row B, so joinability cannot be read from N alone.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, while its confluence and decidable-boundary clauses keep C, D, B visible on the residual comparison route.

Use premise exposure from Definition 5547.5. It prevents a confluence-facing consumer from consuming the bounded endpoint before K, F, C, D, B are displayed in the same packet.

Finally route the read through the sibling frontier. The critical-path packet $\mathsf{MetaCICCriticalPath}^{\uparrow}$ supplies the public handoff direction, and $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads only the residual-diamond boundary, not a global confluence theorem. Since the carrier has no coordinate for Church–Rosser, unrestricted subject reduction, quotient syntax equality, or global conversion decidability, retaining K, C, D, B is exactly the finite residual-diamond condition. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateHandoffResidualDiamondRetention`

Theorem 4737.7 (MetaCIC candidate handoff local join budget)**.** *For every confluence-facing endpoint comparison exported from an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

the consumer must expose a finite local-join budget read through candidate evidence K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, and structural rows T, R, P, L before the endpoint row N is treated as joinable.

The budget is routed through $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPac}$ *and* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *as retained finite evidence. It is not full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.LocalJoinBudget.MetacicCandidateH`

Proof. Start with the boundary and premise-exposure routes, Definition 5547.3 and Definition 5547.5. They place K, F, C, D, B before any confluence-facing read may consume the bounded endpoint row N.

Apply residual-diamond retention from Definition 5547.6. It keeps the residual envelope, decidable boundary, and blocked edge visible while the comparison is routed from $\mathsf{MetaCICCriticalPath}^{\uparrow}$ and $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ toward confluence consumers.

Use the normal-endpoint readback theorem Definition 5547.8. It reads N only after $K, F, C, D, B, T, R, P, L$ have been displayed and names the $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ route as an audit handoff rather than as evaluator equality.

Finally pass the retained evidence to $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$. The frontier receives the finite local-join budget, not a global confluence theorem, so Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, and closed consistency remain outside the carrier. □

Theorem 4737.8 (MetaCIC candidate handoff normal-endpoint readback)**.** *Any bounded normal endpoint read exported by an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must expose candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, and the transport, replay, provenance, and naming rows T, R, P, L before N is consumed. The read may be handed to* $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, *and*

MetaCICConfluenceAuditPacket$^{\uparrow}$ *consumers only as this finite premise-visible route. It is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffNormalEndpointReadback`

Proof. Project the carrier of Definition 5547.2. The bounded endpoint row N is displayed beside candidate evidence K, frontier evidence F, residual envelope C, decidable boundary D, blocked edge B, and the structural rows T, R, P, L.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness exposes K and F before N, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline keeps D on the bounded comparison route.

Now use the consumer boundary Definition 5547.14. A public read reaches MetaCICCriticalPath$^{\uparrow}$ and MetaCICNormalizationFrontier$^{\uparrow}$ only through the displayed handoff route, so the endpoint row cannot be consumed as a detached normal-form theorem.

Finally read the confluence-audit side through the same residual route. The sibling MetaCICConfluenceAuditPack may consume the bounded endpoint only after C, D, B, T, R, P, L remain visible, and those rows only transport, replay, source, and name the finite packet. Since the excluded global endpoints are not carrier coordinates, the normal-endpoint readback preserves the candidate and residual premises. □

Theorem 4737.9 (Candidate normal endpoint retention)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any accepted endpoint read from H must first expose the candidate gate K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, residual-diamond retention row, and normal endpoint readback row before the endpoint row N is consumed by a confluence or conversion-facing consumer. The retained read is a bounded handoff through $A, K, N, F, C, D, B, T, R, P, L$; it is not global confluence, unrestricted strong normalization, quotient syntax equality, or host evaluator soundness.*

Proof. Start with the boundary theorem Definition 5547.3. It displays the candidate gate, normalization frontier, residual envelope, decidable boundary, and blocked-edge rows as separate carrier coordinates before any endpoint read is available.

Apply premise exposure from Definition 5547.5. A public consumer cannot pass directly to N; it must first retain K, F, C, D, B inside the same accepted packet.

Use residual-diamond retention from Definition 5547.6. This keeps the residual envelope, decidable boundary, and blocked edge visible when the endpoint is read by a confluence-facing route.

Finally apply normal-endpoint readback from Definition 5547.8. The rows T, R, P, L transport, replay, source, and name the same bounded route before N is consumed, so the endpoint cannot be reinterpreted as any refused global MetaCIC principle. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.10 (MetaCIC finite endpoint comparison handoff)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any endpoint comparison exported from H to a confluence or conversion consumer must first expose candidate evidence K, normalization-frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L before bounded normal endpoints are compared through N. The exported comparison is a finite candidate-mediated handoff; it is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. Candidate-gate exactness exposes K and F before N, while confluence discipline keeps C and B visible and decidable-boundary discipline places D on the bounded comparison route.

Apply premise exposure from Definition 5547.5. A confluence-facing consumer cannot read the bounded endpoint row until the candidate, frontier, residual-envelope, decidable-boundary, and blocked-edge rows are displayed in the same packet.

Use residual-diamond retention from Definition 5547.6. It prevents a comparison from treating N as detached joinability data; the residual envelope and blocked edge remain part of the read.

Finally apply normal-endpoint readback from Definition 5547.8. The structural rows T, R, P, L transport, replay, source, and name the same finite route before N is consumed, so the comparison cannot export any of the refused global MetaCIC endpoints. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffFiniteEndpointComparison`

Theorem 4737.11 (MetaCIC candidate handoff decidable normal-form comparison)**.** *Any bounded normal-form comparison exported to a decidability consumer from an accepted* MetacicCandidateNormalizationConflu *carrier must expose candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, and structural rows T, R, P, L before the endpoint row N is compared. The comparison is obtained only by chaining boundary exposure, premise exposure, residual-diamond retention, and finite endpoint comparison. It refuses full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, and closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes the candidate row K, frontier row F, residual envelope C, decidable boundary D, and blocked edge B before any endpoint comparison is available.

Apply premise exposure from Definition 5547.5. The bounded endpoint row N remains downstream of the candidate and frontier rows, so the comparison still has to read the finite handoff surface.

Now use residual-diamond retention from Definition 5547.6. The residual envelope and blocked edge remain visible on the comparison route, so the decidability consumer cannot erase the guarded premise surface.

Finally apply finite endpoint comparison from Definition 5547.10. The rows T, R, P, L transport, replay, source, and name the same finite route before N is compared, which is exactly the bounded comparison route refused by the excluded global claims. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffDecidableN`

Theorem 4737.12 (MetaCIC candidate handoff residual-decider inversion)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a confluence or conversion consumer reads a bounded decider outcome from H, then the candidate row K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, and structural rows T, R, P, L are already visible before the endpoint row N is compared. The inversion is a finite retained-premise read; it does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes candidate evidence, the normalization frontier, residual envelope, decidable boundary, and blocked-edge rows as separate coordinates before any bounded endpoint comparison can be read.

Apply premise exposure from Definition 5547.5. A confluence-facing read cannot consume the endpoint row before K, F, C, D, B are displayed in the same packet.

Use residual-diamond retention and finite endpoint comparison, Definition 5547.6 and Definition 5547.10. They keep the residual envelope, decidable boundary, and blocked edge attached to any bounded endpoint comparison offered to a confluence or conversion consumer.

Finally apply decider-row exhaustion through Definition 5547.11. The bounded decider outcome is read only after D is on the displayed comparison route and after T, R, P, L transport, replay, source, and name the same finite handoff. Since the excluded global endpoints are not carrier coordinates, the decider read inverts back to the retained candidate and residual premises rather than to any unrestricted MetaCIC theorem. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffResidualDe`

Theorem 4737.13 (MetaCIC candidate handoff frontier normal-endpoint retention)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any bounded normal endpoint exported by H to a confluence consumer that also reads* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must retain candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L before the endpoint row N is consumed. The same read must preserve the parallel-diamond frontier's residual and local-join rows as visible premises. It does not assert full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.*

Proof. Start with premise exposure and residual-diamond retention, Definition 5547.5 and Definition 5547.6. These require the candidate row, frontier row, residual envelope, decidable boundary, and blocked edge to remain visible before any confluence-facing endpoint read is accepted.

Apply normal-endpoint readback, Definition 5547.8. The bounded endpoint row N is consumed only through the same candidate, frontier, residual, boundary, blocked-edge, transport, replay, provenance, and local naming route.

Finally route the consumer through the sibling frontier dependency. The MetaCICParallelDiamondFrontier$^{\uparrow}$ route named by Definition 5547.6 and chapter 4238 keeps residual and local-join rows visible before the endpoint is used. Since none of the refused global endpoints is a carrier coordinate, the handoff retains only the bounded frontier premises. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffFrontierNo`

Theorem 4737.14 (MetaCIC candidate handoff consumer boundary)**.** *Every public consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet factors through*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto C, D, B \longmapsto T, R, P, L.$$

The consumer receives candidate-mediated normalization evidence, bounded normal endpoint data, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention. It receives no unconditional strong-normalization theorem, full Church–Rosser theorem, unrestricted subject-reduction theorem, global conversion decider, or completed closed-consistency proof.

Proof. Start from the carrier projection. The audit and critical-path rows determine which candidate-mediated normalization and boundary rows are visible to the consumer.

Apply Definition 5547.3. The candidate gate exposes K, F, N before any normal endpoint is available, while C, D, B retain the confluence and decidability boundaries as displayed rows.

The structural rows T, R, P, L do not add another route. They preserve the same carrier under componentwise transport, replay it for downstream consumers, record provenance, and name the packet locally. The absent global theorems cannot be projected from those rows. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.15 (MetaCIC candidate normalization residual progress)**.** *An accepted* MetacicCandidateNormaliza *packet consumes residual data only as a progress ledger. Every admitted candidate-normalization row read through $A, K, N, F, C, D, B, T, R, P, L$ either forwards a retained premise through the candidate and frontier rows K, F or emits a residual obstruction row through C and B, with the residual boundary read against chapter 5126. The packet does not assert full confluence, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The candidate-normalization data are A, K, N, F, the residual and decidability boundary rows are C, D, B, and the structural rows are T, R, P, L.

Apply Definition 5547.3. The candidate gate forces any normalization row to expose K and F before a normal endpoint is consumed, so a retained premise remains visible when the row is forwarded.

If the row cannot be forwarded as a retained premise, the same boundary theorem routes the read through C and B. The residual envelope is then consumed as a ledger compatible with chapter 5126, while the blocked-edge row records the obstruction rather than discharging it.

Finally use Definition 5547.14. The public consumer receives only the candidate-mediated route, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and structural rows. Since the excluded global endpoints are not carrier coordinates, the progress ledger cannot be read as a full confluence or strong-normalization theorem. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.16 (MetaCIC candidate normalization confluence bounded join window)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P, L)$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. A confluence-facing consumer may export a join read from H only after the candidate evidence K, normalization-frontier row F, residual envelope C, decidable-boundary row D, blocked-edge row B, and structural rows T, R, P, L remain visible before the endpoint row N is consumed. The window is finite and carrier-internal: it does not assert full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Proof. Start with the handoff boundary of Definition 5547.3. It exposes K and F before N, keeps C and B visible for confluence-facing reads, and routes bounded comparison through D.

Use premise exposure from Definition 5547.5. A public confluence consumer cannot read the endpoint row before the candidate, frontier, residual-envelope, decidable-boundary, and blocked-edge rows are displayed in the same packet.

Apply residual-diamond retention from Definition 5547.6. It prevents a join read from treating N as detached joinability data; the read must retain K, C, D, B on the residual route.

Finally use normal-endpoint readback from Definition 5547.8. The structural rows T, R, P, L preserve, replay, source, and name the same finite route before N is consumed. Since no excluded global endpoint is a carrier coordinate, the exported join is available only through this bounded window. □

Theorem 4737.17 (MetaCIC candidate normalization confluence rectangle)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If both candidate-normalization edges of a bounded comparison are admitted by the candidate gate and each residual edge is read through the residual envelope, then the handoff packet either displays a common bounded endpoint row or records a residual obstruction row through C and B. The rectangle is guarded by the candidate and residual rows; it does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.*

Proof. Start with Definition 5547.3. The candidate gate forces both normalization edges to expose K and F before any endpoint row N can be consumed.

Use the consumer boundary of Definition 5547.14. A public handoff read must pass through the residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L.

Apply residual progress from Definition 5547.15. If the two candidate-normalization edges forward through the retained premises, the packet displays the bounded endpoint row shared by those reads. If one residual edge cannot be forwarded, the same route records the obstruction through C and B.

The residual side is read against chapter 5126. Since the carrier contains no coordinates for unrestricted Church–Rosser, unconditional strong normalization, subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency, the rectangle remains a guarded handoff packet. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 4737.18 (MetaCIC candidate normalization confluence handoff candidate residual join)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P, L)$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If two candidate-normalization edges exported from H expose candidate evidence K and frontier evidence F, and every residual edge is routed through the residual-envelope row C and blocked-edge row B, then any exported join row is guarded by K, F, C, D, B, and the structural rows T, R, P, L. The join row is a candidate-mediated residual handoff; it is not full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Proof. Begin with the boundary theorem Definition 5547.3. It forces any endpoint or comparison read to display K, F, N before the consumer sees bounded normalization data, and it keeps C, D, B visible on the residual and decidable-boundary side.

Use residual progress from Definition 5547.15. Under the stated routing, residual edges either remain retained candidate premises through K, F or become obstruction rows through C, B; neither route removes the candidate evidence.

Finally apply the rectangle theorem Definition 5547.17. The exported join row is therefore read only as the guarded rectangle output, with T, R, P, L preserving the same finite route by transport, replay, provenance, and local naming. Since no excluded global theorem is a carrier coordinate, the join cannot be promoted beyond the candidate-mediated residual handoff. □

Theorem 4737.19 (MetaCIC candidate normalization parallel-frontier consumption)**.** *If a* MetaCICParallelDiamon *consumer reads a join row from an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

then the read must preserve candidate evidence K, normalization-frontier data F, residual-envelope data C, decidable-boundary data D, blocked-edge data B, componentwise transport T, displayed replay R, provenance P, and local naming L before N can be treated as bounded endpoint data. The exported join is only a candidate-mediated residual handoff; it is not full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, evaluator soundness, or closed consistency.

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Proof. Start with the boundary theorem Definition 5547.3. It prevents any endpoint read from reaching N before the candidate evidence K, normalization frontier F, residual envelope C, decidable boundary D, and blocked-edge row B remain visible.

Apply the premise-exposure route of Definition 5547.14. A public consumer receives the handoff only through the same carrier rows and through the structural data T, R, P, L.

Use residual progress from Definition 5547.15 and the rectangle theorem Definition 5547.17. They keep retained candidate premises and residual obstructions attached to K, F, C, D, B while allowing only bounded endpoint data or an obstruction row to be displayed.

Finally invoke the candidate residual join theorem Definition 5547.18. The join already exported by H is guarded by the candidate, frontier, residual, blocked-edge, and structural rows. A MetaCICParallelDiamondFrontier$^{\uparrow}$ consumer therefore receives a candidate-mediated residual handoff for bounded endpoint data, not any of the excluded global confluence, subject-reduction, conversion-decidability, quotient-equality, evaluator-soundness, or consistency claims. □

Theorem 4737.20 (MetaCIC candidate normalization confluence boundary)**.** *Let $H = (A, K, N, F, C, D, B, T, R,$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a bounded comparison exports a join row from H, then the read preserves candidate evidence K, the normalization frontier F, the residual envelope C, the decidable boundary D, the blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. The exported join row is therefore a guarded candidate-normalization handoff. It cannot be read as unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. A bounded comparison can export a join row only through the displayed candidate, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and naming coordinates of H.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness keeps K and F visible before endpoint data are consumed, confluence discipline keeps C and B visible for residual reads, and decidable-boundary discipline routes bounded comparison through D.

Use residual progress and the rectangle theorem from Definition 5547.15 and Definition 5547.17. They allow the bounded comparison to display a guarded endpoint row or an obstruction row, but only while retaining the candidate and residual coordinates.

Finally read the structural rows T, R, P, L. They transport, replay, source, and name the same finite route; none of them creates a coordinate for unconditional normalization, full Church–Rosser, unrestricted subject reduction, global decidability, quotient syntax equality, evaluator soundness, or closed consistency. Thus no excluded coordinate is available to the consumer. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceBoundary`

Theorem 4737.21 (MetaCIC candidate normalization confluence handoff decider boundary)**.** *Any bounded conversion comparison exported through the decidable-boundary row D of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$ must first expose candidate evidence K, normal endpoint data N, the normalization frontier row F, the residual-envelope row C, the blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. The exported comparison is only a bounded comparison boundary. It is not global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The only conversion-facing coordinate is D, and it is displayed alongside the candidate, endpoint, frontier, residual-envelope, blocked-edge, and structural rows.

Apply Definition 5547.3. Its candidate-gate exactness forces K, F, N to appear before a normal endpoint can be consumed, and its confluence handoff discipline keeps C and B visible before any residual or join read is offered.

Next use Definition 5547.14. A public consumer receives the bounded comparison only through the same carrier route and through T, R, P, L; the structural rows do not create a second decider surface.

Finally apply Definition 5547.15. Residual data remain progress ledger entries or obstruction rows, so the bounded comparison exported through D cannot be promoted to the excluded global conversion, confluence, subject-reduction, evaluator-equality, quotient-equality, or consistency claims. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.22 (MetaCIC candidate handoff decider-row exhaustion)**.** *Any bounded conversion comparison exported through row D of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$ must preserve the candidate evidence row K, endpoint row N, normalization frontier row F, residual envelope row C, blocked-edge row B, componentwise transport row T, displayed replay row R, provenance row P, and local naming row L. No consumer may read D as a standalone global conversion decider. The exhausted row remains only a bounded comparison surface and does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The decidable-boundary row D appears only as one coordinate of H, beside the candidate, endpoint, frontier, residual-envelope, blocked-edge, and structural rows.

Apply the boundary theorem Definition 5547.3. It places bounded comparison reads after candidate evidence, frontier evidence, endpoint data, residual-envelope visibility, and blocked-edge retention.

Apply the consumer boundary Definition 5547.14. A public consumer receives D only through the displayed route with T, R, P, L, so there is no detached decider surface outside the carrier.

Finally use Definition 5547.21. That theorem identifies D as the bounded comparison row after the displayed endpoint and residual rows have been exposed. Since the excluded global endpoints are not carrier coordinates, exhausting row D preserves the finite comparison boundary without collapsing into global decidability. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.23 (MetaCIC candidate normalization confluence retained-premise inversion)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a bounded endpoint row or join row is read from H, then the read determines the same candidate evidence K, normalization frontier F, residual envelope C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. Dropping any retained row changes the packet and cannot be presented as the same confluence handoff.*

Proof. Project the carrier of Definition 5547.2. The endpoint and join rows are not independent data; they are read from the displayed tuple with $K, F, C, D, B, T, R, P, L$ still visible.

Apply the seed package and boundary results from Definition 5547.4, Definition 5547.5, and Definition 5547.3. These force candidate evidence, frontier data, residual-envelope visibility, decidable-boundary access, and blocked-edge retention to remain part of the same handoff before any endpoint or join row is consumed.

Use the bounded conversion comparison surface of Definition 5547.21 and Definition 5547.22. The bounded comparison row D can be read only together with the structural rows T, R, P, L, so erasing any retained coordinate changes the packet rather than simplifying the same one.

Finally apply the nonescape clauses in Definition 5547.20 and Definition 5547.14. They rule out parameter transport into a detached endpoint or join claim, leaving only constructor-style inversion on the concrete carrier rows. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Definition 4737.24 (MetaCIC finite conversion certificate carrier)**.** *A* MetacicCandidateNormalizationConfluenceHa *finite conversion certificate is the subroute of an accepted carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

that displays the bounded normal endpoint row N, normalization frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and structural rows T, R, P, L in one route. The certificate keeps K as its candidate-evidence entry condition and uses A only as the audit anchor inherited from the surrounding carrier. It has no coordinate for global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.

Theorem 4737.25 (MetaCIC finite conversion certificate boundary)**.** *Every bounded conversion read exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must display, in one finite route, the normal endpoint row N, normalization frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport row T, displayed replay row R, provenance row P, and local naming row L. Such a read is a finite conversion certificate in the sense of Definition 5547.24. It is not global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Begin with the finite certificate carrier. It is not a second carrier: it is the displayed $N, F, C, D, B, T, R, P, L$ subroute of Definition 5547.2, with K as the candidate-evidence entry condition and A as the audit anchor.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness places K, F, N before a bounded normal endpoint is consumed, while the confluence handoff discipline keeps C and B visible for residual reads.

Use the consumer boundary of Definition 5547.14. A public bounded conversion consumer receives the comparison only through the same structural route T, R, P, L and not through a detached evaluator or quotient-syntax endpoint.

Finally combine residual progress with the decider boundary of Definition 5547.15 and Definition 5547.21. Residual entries remain progress or obstruction rows, and the exported comparison passes through D as a bounded boundary. Since the excluded global claims are not carrier coordinates, the finite certificate cannot be read as global conversion decidability, Church–Rosser, subject reduction, evaluator equality, quotient equality, or consistency. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicFiniteConversionCertificat`

Theorem 4737.26 (MetaCIC candidate handoff obligation rows)**.** *Any obligationClosure upgrade for* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *must display candidate evidence K, normal endpoint data N, frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L as non-collapsible rows. In particular, the candidate-evidence row cannot be folded into the normal endpoint row, the residual-envelope row cannot be folded into the decidable-boundary row, and the blocked-edge row cannot be erased by transport or replay. The upgrade remains a finite handoff obligation surface; it does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffObligationRows`

Proof. Project the carrier of Definition 5547.2. It displays $A, K, N, F, C, D, B, T, R, P, L$ as separate rows, so an obligationClosure read must account for candidate evidence, normal endpoint data, frontier discipline, residual-envelope data, decidable-boundary data, blocked edges, and the structural route.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness keeps K and F visible before N is consumed, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline keeps D downstream of the displayed endpoint and residual rows.

Use the consumer boundary, residual progress, decider boundary, and finite conversion certificate boundary from Definition 5547.14, Definition 5547.15, Definition 5547.21, and Definition 5547.25. Together they force public reads to preserve candidate-mediated normalization,

residual-envelope visibility, bounded conversion comparison, blocked-edge retention, transport, replay, provenance, and local naming as the same finite route.

Therefore collapsing any one of $K, N, F, C, D, B, T, R, P, L$ would remove a row required by the carrier and by the displayed boundary theorems. Since the excluded global endpoints are not carrier coordinates, the obligation rows remain non-collapsible without promoting the packet beyond the finite handoff surface. □

Theorem 4737.27 (MetaCIC candidate handoff row readiness)**.** *Every confluence-facing or decidability-facing read of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

must expose the candidate row K and the frontier row F before the endpoint row N or the residual rows C, D, B are consumed. Thus the handoff exports only a finite premise-readiness route, not a global strong-normalization endpoint, full Church–Rosser theorem, unrestricted subject-reduction theorem, or global conversion decider.

Proof. Project the carrier of Definition 5547.2. The rows K and F are separate from N, C, D, B, so a consumer can only read endpoint, residual, or decidability data through the displayed handoff packet.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness places K and F before N, while its confluence and decidability clauses keep C, D, B on the residual and bounded-comparison side.

Use the obligation rows theorem Definition 5547.26. It makes $K, N, F, C, D, B, T, R, P, L$ non-collapsible rows, so neither transport nor replay may hide the candidate and frontier premises before a consumer reaches the endpoint or residual rows.

Since the carrier contains no coordinate for global strong normalization, full Church–Rosser, unrestricted subject reduction, or global conversion decidability, the exposed route is exactly a local premise-readiness route for confluence-facing and decidability-facing consumers. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.28 (MetaCIC candidate retained-edge handoff)**.** *Every finite successor handoff from candidate normalization toward confluence in an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the candidate-gate rows K, F, N, the residual-envelope row C, the decidable-boundary row D, the blocked-edge row B, and the structural rows T, R, P, L before any consumer reads endpoint or conversion data. The retained edge is a finite successor route inside $H = (A, K, N, F, C, D, B, T, R, P, L)$; it does not assert Church–Rosser, strong normalization, subject reduction, evaluator equality, quotient equality, or consistency.*

Proof. Start with the boundary theorem Definition 5547.3. Candidate-gate exactness keeps K, F, N visible in the successor handoff, while confluence handoff discipline keeps C and B visible.

Apply the consumer boundary of Definition 5547.14. A confluence-facing consumer receives the handoff only through the displayed public route, so endpoint data cannot be read before the retained candidate and residual rows.

Use the decider boundary from Definition 5547.21. Any bounded conversion comparison passes through D after the candidate, endpoint, residual-envelope, blocked-edge, and structural rows are displayed.

Finally apply Definition 5547.26. It makes $K, F, N, C, D, B, T, R, P, L$ non-collapsible rows of the obligation surface. Since the excluded global endpoints are not carrier coordinates, a finite

successor handoff retains those rows without proving Church–Rosser, strong normalization, subject reduction, evaluator equality, quotient equality, or consistency. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 4737.29 (MetaCIC candidate normalization handoff obligation stability)**.** *Let* $H = (A, K, N, F, C, D, B, T, R, P, L)$ *be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any public read preserving the handoff rows must preserve candidate evidence* K*, endpoint row* N*, frontier row* F*, residual envelope* C*, decidable boundary* D*, blocked edge* B*, componentwise transport* T*, displayed replay* R*, provenance* P*, and local naming* L*. No* hsame *transport or* Cont *replay may erase candidate or residual obligations, and no public read may collapse the handoff into unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationHand

Proof. Project the carrier of Definition 5547.2. The rows $K, N, F, C, D, B, T, R, P, L$ occur as separate coordinates, with A only anchoring the audit-map source.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness preserves K, F, N, confluence handoff discipline preserves C and B, and decidable-boundary discipline preserves D as the bounded comparison boundary.

Use residual progress from Definition 5547.15. Residual data either remain retained premises through K, F or obstruction rows through C, B, so a public read cannot erase the candidate or residual obligation while still preserving the handoff.

Apply the finite conversion certificate boundary of Definition 5547.25 and the obligation rows theorem Definition 5547.26. They require $N, F, C, D, B, T, R, P, L$ to remain visible in one finite route and require K as the candidate-evidence entry condition.

Finally read T and R only as componentwise transport and displayed replay. They preserve the already displayed rows and do not create a coordinate that can discharge or remove candidate evidence, residual-envelope data, or blocked edges. The excluded global endpoints are not carrier coordinates, so stability stays inside the finite handoff surface. □

Theorem 4737.30 (MetaCIC finite conversion endpoint handoff)**.** *Every finite conversion certificate exposed by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet reaches residual confluence consumers only through the candidate-normalization row* K*, the residual-envelope row* C*, the blocked-edge row* B*, and the decidable-boundary row* D*. In particular, a consumer of the certificate must read the endpoint route through Definition 5547.24, Definition 5547.25, Definition 5547.15, and Definition 5547.17. The handoff does not assert full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.*

Proof. Project the finite certificate from the carrier. By Definition 5547.24, the exposed endpoint is the displayed subroute through $N, F, C, D, B, T, R, P, L$, with K retained as the candidate-evidence entry condition.

Apply Definition 5547.25. The bounded conversion read is not detached from the handoff packet: it must display the normal endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows in one route.

Use residual progress and the guarded rectangle from Definition 5547.15 and Definition 5547.17. A residual confluence consumer either receives the bounded endpoint row through the candidate-mediated route or receives an obstruction row through C and B.

Finally apply the handoff decider boundary of Definition 5547.21. The bounded comparison passes through D after the displayed endpoint and residual rows, so no consumer can read the certificate as one of the excluded global endpoints. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicFiniteConversionEndpointHa

Theorem 4737.31 (MetaCIC candidate normalization public handoff)**.** *Any public confluence or decidability consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet preserves candidate evidence K, normal endpoint N, normalization frontier F, residual envelope C, decidable boundary D, blocked edge B, transport T, replay R, provenance P, and local naming L before reading a bounded endpoint or conversion row. The handoff does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. First apply the finite conversion certificate boundary of Definition 5547.25. It keeps the bounded endpoint read inside the displayed endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming route.

Next use residual progress from Definition 5547.15. Residual data remain visible either as retained candidate premises through K, F or as obstruction rows through C, B, so a public confluence consumer cannot erase them before reading a bounded endpoint.

Now apply the confluence rectangle and decidable-boundary theorems, Definition 5547.17 and Definition 5547.21. They route bounded comparison through D only after the candidate, endpoint, frontier, residual-envelope, and blocked-edge rows have been displayed.

Finally combine obligation stability and public endpoint handoff from Definition 5547.29 and Definition 5547.30. The structural rows T, R, P, L preserve the same finite packet, and none of the excluded global endpoints is a carrier coordinate. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 4737.32 (MetaCIC normalization audit-map realization)**.** *The* MetacicCandidateNormalizationConfluenc *packet is the concrete finite* NameCert *realization of the normalization audit-map route of chapter 9383. Its carrier displays candidate evidence K, bounded endpoint row N, frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L as one guarded handoff. The realization does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The audit anchor A binds the packet to chapter 9383, and the rows $K, N, F, C, D, B, T, R, P, L$ display the finite candidate, endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming data.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness forces K, F, N to appear before any endpoint read, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline routes bounded comparison through D.

Use the consumer boundary and endpoint handoff from Definition 5547.14 and Definition 5547.30. Together they show that every public read of the audit-map route remains inside the displayed finite conversion certificate and its residual-envelope boundary.

Thus the vision route has a concrete finite packet: candidate evidence, bounded endpoint, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and

naming are all carrier rows. Since none of the excluded global endpoints is a carrier coordinate, the realization stays a guarded NameCert handoff. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 4737.33 (Finite-observation MetaCIC diamond-envelope obligations)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet, a finite-observation diamond envelope exposes candidate evidence K, bounded endpoint row N, residual-envelope row C, local join and checker status through the decidable-boundary row D, blocked-edge retention through B, componentwise transport T, displayed replay R, provenance P, and local naming L before any confluence consumer reads a bounded join. The envelope is a finite observation route inside the existing handoff carrier; it is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.FiniteObservationMetaCICDiamondEn

Proof. Project the carrier of Definition 5547.2. The finite observation envelope can only use the displayed rows $K, N, F, C, D, B, T, R, P, L$.

Apply Definition 5547.3. Candidate-gate exactness keeps K, F, N visible before a bounded endpoint is consumed, while confluence handoff discipline keeps the residual-envelope row C and blocked-edge row B visible before a residual or join read is offered.

Use Definition 5547.21. The local checker status is read through D only after the endpoint, residual-envelope, blocked-edge, transport, replay, provenance, and naming rows are displayed.

Finally apply Definition 5547.30. Any bounded join read remains inside the finite certificate route, so no consumer can read the envelope as an unconditional confluence or normalization theorem. □

Theorem 4737.34 (MetaCIC candidate-normalization confluence handoff obligation surface)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet, the handoff exposes finite obligation rows for candidate evidence K, residual envelope C, decidable-boundary routing D, blocked-edge retention B, bounded endpoint readback N, componentwise transport T, replay R, provenance P, and local naming L. The route remains guarded by the audit anchor A and frontier row F, and it exports none of strong normalization, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The displayed rows $A, K, N, F, C, D, B, T, R, P, L$ are the only coordinates available to a confluence-facing consumer.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. They force candidate evidence and frontier data to remain visible before endpoint, residual, or bounded-comparison rows are consumed.

Use residual-diamond retention from Definition 5547.6 and normal-endpoint readback from Definition 5547.8. These route residual evidence through C, B and endpoint reads through N without collapsing the decidable-boundary row D.

Finally read T, R, P, L as structural rows. They transport, replay, source, and locally name the same finite handoff, so the obligation surface is exhausted by the displayed coordinates and cannot be reinterpreted as any of the refused global MetaCIC endpoints. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 4737.35 (Finite-observation MetaCIC diamond-envelope handoff)**.** *Every confluence or normalization consumer of a finite-observation MetaCIC diamond envelope receives a bounded join only through the candidate evidence, bounded endpoint, residual-envelope, local checker, blocked-edge, transport, replay, provenance, and local* NameCert *rows of Definition 5547.33. The handoff exports the bounded finite-observation route and no global Church–Rosser theorem, unconditional strong-normalization theorem, unrestricted subject-reduction theorem, global conversion decider, evaluator equality, quotient syntax equality, or consistency endpoint.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.FiniteObservationMetaCICDiamondEn`

Proof. Start with the finite-observation obligations. They place the candidate, endpoint, residual-envelope, checker, blocked-edge, transport, replay, provenance, and naming rows before any bounded join read.

Use the guarded rectangle of Definition 5547.17. A bounded comparison either displays a common bounded endpoint row or records a residual obstruction, and both alternatives keep the candidate and residual coordinates visible.

Apply the finite conversion certificate boundary of Definition 5547.25. The public consumer receives only the displayed finite route, so the handoff cannot be reinterpreted as a detached global confluence or normalization theorem. □

Theorem 4737.36 (MetaCIC candidate-normalization residual-diamond totality)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet that exposes candidate evidence K, bounded normal endpoint row N, residual-retention rows C, B, and decidable-boundary row D must supply the finite residual-diamond join route before any public confluence consumer reads it. The supplied route is the bounded candidate-normalization handoff through $A, K, N, F, C, D, B, T, R, P, L$; it is not a global Church–Rosser theorem, unrestricted subject-reduction theorem, unconditional strong-normalization theorem, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The rows K, N, C, D, B are displayed beside the audit anchor, frontier row, componentwise transport, replay, provenance, and local naming rows, so a public confluence read has no coordinate outside $A, K, N, F, C, D, B, T, R, P, L$ from which to obtain a join.

Apply residual-diamond retention from Definition 5547.6. It requires residual or join reads to retain candidate evidence, the residual envelope, the decidable boundary, and the blocked-edge row before the endpoint row is treated as joinable.

Use the consumer boundary of Definition 5547.14. A public consumer may read the handoff only after the candidate-mediated route, residual-envelope visibility, bounded comparison routing, blocked-edge retention, transport, replay, provenance, and local naming rows remain visible.

Finally combine the guarded rectangle of Definition 5547.17 with the finite-observation handoff of Definition 5547.35. The available join is therefore total only as a finite residual-diamond route supplied by the packet, and the excluded global endpoints remain absent from the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.37 (MetaCIC candidate-normalization residual-strip local join)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet exposing candidate evidence K, bounded endpoint row N, residual envelope C, decidable boundary D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local* NameCert *row L supplies only a local residual-strip join route for bounded candidate-normalization consumers. The route is read through*

the finite handoff rows $A, K, N, F, C, D, B, T, R, P, L$ and does not export full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, quotient syntax equality, host evaluator soundness, global conversion decidability, or kernel consistency.

Proof. Project the obligation surface of Definition 5547.34. It displays the candidate evidence, residual envelope, decidable boundary, blocked edge, bounded endpoint, transport, replay, provenance, and local naming rows as the only handoff coordinates available to a confluence-facing consumer.

Apply the finite-observation handoff of Definition 5547.35. A bounded join read is offered only after the finite observation envelope has kept the candidate row, endpoint row, residual row, checker row, blocked edge, and structural rows visible.

Use residual-diamond totality from Definition 5547.36. It supplies the join only as a finite residual-diamond route inside $A, K, N, F, C, D, B, T, R, P, L$.

The nonescape clauses are therefore inherited from the same three routes. Since none of full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, quotient syntax equality, host evaluator soundness, global conversion decidability, or kernel consistency is a carrier coordinate, the consumer receives a local residual-strip join and nothing global. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 4737.38 (MetaCIC candidate-normalization local diamond handoff)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier. If two one-step candidate-normalization residual reads share the audit anchor A, candidate evidence K, normalization frontier F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L, then any displayed join between their bounded endpoint reads is available only as a local diamond handoff through the residual-window row, candidate-join row, relation-scope row, and transport rows already carried by C, D, B, T, R, P, L.*

The handoff preserves the same boundary rows on both sides of the diamond. It exports no full Church–Rosser theorem, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or unrestricted normalization theorem.

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicNormalizationConfluenceLo`

Proof. Start with the carrier and boundary rows. The carrier displays $A, K, N, F, C, D, B, T, R, P, L$, and Definition 5547.3 requires every confluence-facing read to keep candidate evidence, frontier data, residual envelope, decidable boundary, and blocked-edge retention visible.

Read the two one-step residuals through the same boundary data. Premise exposure and residual-diamond retention, Definition 5547.5 and Definition 5547.6, prevent either branch from treating the bounded endpoint row as detached joinability data.

Use residual-diamond totality and residual-strip local join, Definition 5547.36 and Definition 5547.37. They provide only the finite join route inside $A, K, N, F, C, D, B, T, R, P, L$, so the residual-window, candidate-join, and relation-scope rows remain displayed during the handoff.

Finally transport both sides through T, R, P, L. These rows preserve, replay, source, and name the same local diamond route; they do not create a coordinate for the refused global MetaCIC endpoints. Hence the displayed diamond is a local handoff over the retained residual and candidate rows, not a global confluence or normalization theorem. □

Theorem 4737.39 (MetaCIC candidate-normalization confluence handoff seed status)**.** *A seed-status reader of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must expose candidate evidence K, normalization frontier F, residual confluence envelope C, decidable boundary D, blocked-edge row B, componentwise* hsame *transport T, displayed* Cont *replay R,* Pkg *provenance P, and local* NameCert *row L before it consumes any bounded normal endpoint row N. The seed status is the finite retained-premise boundary of the handoff; it does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. The only available rows are the audit anchor, candidate evidence, bounded endpoint, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming data.

Apply the boundary theorem Definition 5547.3 and the seed-closure route of Definition 5547.4. They keep K, F, C, D, B visible before endpoint consumption.

Use premise exposure from Definition 5547.5, residual-diamond retention from Definition 5547.6, and decider-row exhaustion from Definition 5547.22. These force the residual envelope, decidable boundary, and blocked-edge row to remain displayed whenever a bounded normal endpoint is read.

Finally read T, R, P, L as structural rows. They transport, replay, source, and name the same finite packet, so the seed-status reader cannot project any of the refused global MetaCIC endpoints from the handoff. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Falsifiable prediction: Every accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ read that exports a normal endpoint, residual-envelope row, or bounded conversion comparison without first exposing candidate evidence, the normalization frontier, the confluence residual envelope, the decidable boundary, blocked-edge retention, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-confluence-residual-envelope, metacicdecidableboundary, metacic-normalization-confluence-bridge: no listed sibling supplies the joint handoff containing candidate evidence, bounded normal endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local NameCert rows while refusing unconditional SN and Church–Rosser.

Lean follow-up should provide Nontrivial and FieldFaithful instances for the carrier so the candidate row, residual-envelope row, and decidable-boundary row can be audited as distinct fields rather than as a collapsed route.

The handoff also consumes the residual-diamond route of chapter 5126 as a concrete prerequisite. The candidate-normalization row must expose its cut-height budget and residual join data before the confluence handoff can be read, so the rows K, N, F, C, D, B remain tied to the parallel residual packet rather than becoming a standalone confluence claim.

Theorem 4737.40 (MetaCIC candidate normalization handoff sibling route)**.** *No* MetacicCandidateNormalizationC *consumer may read bounded normal endpoints or local residual-strip joins until it has exposed the candidate evidence row K, the* MetaCICNormalizationFrontier$^{\uparrow}$ *row F, the residual confluence-envelope row C, the decidable-boundary row D, the blocked-edge row B, and the sibling residual-diamond route through chapter 5126. The available read is the finite handoff along $A, K, N, F, C, D, B, T, R, P, L$; it is not full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Start with the boundary theorem Definition 5547.3 and premise exposure Definition 5547.5. They force the candidate evidence, normalization frontier, residual envelope, decidable boundary, and blocked edge to remain displayed before a bounded endpoint can be consumed.

Residual-diamond retention, Definition 5547.6, ties any join read to K, C, D, B rather than to the endpoint row alone. Decider-row exhaustion in Definition 5547.22 keeps the decidable-boundary row as bounded checker evidence, not as a global conversion oracle.

Use the residual-strip local join theorem Definition 5547.37. Its join route is local to the displayed packet $A, K, N, F, C, D, B, T, R, P, L$ and must keep the sibling residual-diamond route visible.

Finally apply seed status, Definition 5547.39. The seed reader receives only the retained-premise boundary, so full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, and kernel consistency remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationHand`

Theorem 4737.41 (MetaCIC candidate handoff scoped TasteGate route)**.** *Every later* TasteGate *instance for* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *must cite the exact paper-side route:*

1. *the carrier of Definition 5547.2;*
2. *the boundary of Definition 5547.3;*
3. *the obligation surface of Definition 5547.34;*
4. *residual-diamond totality from Definition 5547.36;*
5. *residual-strip local join from Definition 5547.37;*
6. *seed status from Definition 5547.39;*
7. *the sibling confluence handoff of Definition 5547.40.*

This route is the one needed by the GroundCompiler *TasteGate check. It does not supply full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or an unrestricted normalization theorem.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffScopedTast`

Proof. Start with the carrier and boundary. The carrier definition displays $A, K, N, F, C, D, B, T, R, P, L$, while the boundary theorem keeps candidate evidence, frontier data, residual envelope, decidable boundary, and blocked-edge retention visible before any bounded endpoint is consumed.

Next read the obligation surface. It gives the rows a local NameCert surface and prevents the handoff from being treated as a detached MetaCIC confluence theorem.

Use residual-diamond totality and residual-strip local join. These theorems supply only the finite join route inside $A, K, N, F, C, D, B, T, R, P, L$, with the residual envelope and decidable-boundary row still displayed.

Finally cite seed status and the sibling route. Seed status records the retained-premise boundary, and the sibling route keeps the critical-path, normalization-frontier, confluence-audit, residual local-join, and parallel-diamond residual surfaces visible. Thus a GroundCompiler TasteGate check can cite the scoped route above, while full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, and unrestricted normalization remain outside the claim. □

Theorem 4737.42 (MetaCIC candidate-normalization confluence scoped surface)**.** *The scoped* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *surface is exactly the carrier row, residual-diamond row, bounded join-window row, decider-row exhaustion row, finite conversion endpoint row, audit-map realization row, and sibling route row displayed in this chapter. The surface does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or kernel consistency.*

Proof. Start with the carrier and obligation surface, Definition 5547.2 and Definition 5547.34. They name the packet rows and keep the candidate evidence, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming visible.

Next use residual-diamond totality and residual-strip local join, Definition 5547.36 and Definition 5547.37. These provide only bounded local joins and finite conversion endpoints inside the displayed packet.

Then apply decider-row exhaustion and seed status, Definition 5547.22 and Definition 5547.39. They keep the decider and audit-map realization rows as retained premises rather than global conversion or kernel-consistency claims.

Finally cite the sibling route Definition 5547.40. It binds the residual-diamond dependency and the finite handoff route, so the scoped surface is precisely the listed packet and sibling rows. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Closure status for NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ ·

Theory closure: TheoryGate(NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$, obligation)

Scope closed: Definition 5547.2, Definition 5547.3, Definition 5547.34, Definition 5547.36, Definition 5547.37, and Definition 5547.39 record the seed and obligation surface for candidate normalization, residual join, confluence handoff, finite-window replay, and MetaCIC frontier dependencies.

Formal verification: FormalStatus(NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp`

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-gate exactness, confluence handoff discipline, decidable-boundary discipline, frontier non-collapse, Nontrivial, and FieldFaithful instances for BEDC.Derived.MetacicCandidateNorm

Constructive story: A MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet is generated from a MetaCIC audit anchor, candidate evidence, bounded normal endpoint row, MetaCICNormalizationFrontierUp row, residual confluence-envelope row, MetaCICDecidableBoundaryUp row, blocked-edge retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 4737.43 (MetaCIC candidate residual-budget discharge frontier)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

candidate evidence K, residual envelope C, decidable boundary D, and blocked-edge row B remain visible before the endpoint row N is treated as a residual join by MetaCICParallelDiamondFrontier$^{\uparrow}$*. The discharge frontier is therefore the finite route from the audit anchor A through $K, F, C, D, B, T, R, P, L$ into the parallel-diamond consumer, not full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. The rows $A, K, N, F, C, D, B, T, R, P, L$ display the audit anchor, candidate evidence, endpoint row, normalization frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. Together they keep K, F, C, D, B attached to the same accepted packet before any confluence-facing endpoint read can treat N as residual join data.

Finally read the parallel-diamond frontier sibling of chapter 4238. Its consumer may inspect the residual join only after the candidate, residual-envelope, decidable-boundary, and blocked-edge rows have remained visible through T, R, P, L.

The refused global conclusions are absent from the carrier and from this consumer route. Thus the discharge frontier records a candidate-mediated finite handoff rather than a global MetaCIC theorem. □

Theorem 4737.44 (MetaCIC candidate normalization diamond-budget route)**.** *Every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *closed-normal read must first expose candidate-set membership, a bounded local peak, join-budget rows, the sibling confluence route of chapter 6552, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *row. Thus the closed-normal endpoint is consumed only after the diamond-budget dependency surface is visible, not as an evidence-free consequence.*

Proof. Project the parent carrier of Definition 5547.2. Its accepted read displays candidate evidence, bounded endpoint data, residual confluence routing, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming before any consumer can see a normal endpoint.

Apply the boundary theorem of Definition 5547.3. Candidate-gate exactness exposes the candidate-set membership row before the endpoint row is consumed, while confluence handoff discipline keeps the residual and blocked-edge rows visible.

The sibling route through chapter 6552 supplies the bounded local peak and join-budget rows. Reading that sibling before the closed-normal endpoint preserves the finite diamond budget as a dependency of the handoff, and T, R, P, L transport, replay, source, and name the same route without adding a global confluence or normalization coordinate. □

Theorem 4737.45 (MetaCIC candidate normalization handoff route)**.** *Every accepted* MetacicCandidateNormalizat *packet that exports a confluence handoff must first expose the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source row, the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier row, residual join scope, candidate normalization budget, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. In carrier notation the route reads the displayed packet rows* $A, K, N, F, C, D, B, T, R, P, L$ *before any consumer treats the bounded endpoint row* N *as confluence-facing data.*

The exported handoff is therefore a finite candidate-mediated route through the critical path, parallel-diamond frontier, residual envelope, decidable boundary, blocked-edge budget, transport, replay, provenance, and local naming surface. It is not global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.

Proof. Start with the carrier of Definition 5547.2. The rows $A, K, N, F, C, D, B, T, R, P, L$ display the audit anchor, candidate evidence, bounded endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming.

Next read the sibling routes. The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ row supplies the public critical-path source, while the $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ row supplies the retained frontier that keeps residual join scope visible before any endpoint is consumed.

Now apply the existing premise-exposure and local-join routes of Definition 5547.5 and Definition 5547.7. They force candidate evidence, residual envelope data, decidable-boundary data, and blocked-edge data to remain attached to the same handoff packet.

Finally pass through T, R, P, L. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows preserve the finite packet and add no coordinate for the refused global MetaCIC endpoints. Hence the confluence handoff can be exported only as the stated candidate-normalization route. □

Theorem 4737.46 (MetaCIC candidate-normalization handoff cut-height budget route)**.** *Any* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *consumer that reads a cut-height budget or local-diamond endpoint must first expose candidate evidence, the normalization frontier, residual confluence, the decidable-boundary route, and blocked-edge rows in the same finite handoff packet. In particular, the consumer must cite the parallel-diamond frontier of chapter 4238 and the normalization frontier of chapter 892 before any cut-height or local-diamond row is consumed.*

Proof. Project the candidate-normalization confluence handoff carrier from the local NameCert construction. Its public rows expose candidate evidence, bounded normal-endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

The cut-height consumer is downstream of those rows. A local-diamond read needs the residual confluence and blocked-edge rows, while a budget read needs the normalization frontier and decidable-boundary rows before it can decide which bounded endpoint is being measured.

Now read the sibling frontiers. The parallel-diamond frontier supplies the residual diamond and local-join surface, and the normalization frontier supplies the bounded normal-endpoint surface. Since the handoff has no separate cut-height coordinate outside those displayed routes, every accepted cut-height or local-diamond consumer must first display the stated rows. □

The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ dossier-lattice anchor consumes the parallel-diamond frontier route of chapter 4238 and the local-confluence certificate route of chapter 7346 before residual-normalization handoff rows are public. Thus the handoff keeps candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming attached to those sibling routes instead of exporting a standalone confluence endpoint. The candidate-normalization handoff consumes the residual-join and local-confluence routes through chapter 4238 and reads the Real and RegSeqRat-style finite route discipline of chapter 909 only as packet ordering. This dependency records the visible sibling path for residual joins, local diamonds, finite readback rows, transport, replay, provenance, and local naming; it does not select a normalization, quotient terms, identify evaluator equality, or turn the Real-facing ordering discipline into a MetaCIC equality principle. The MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ dossier lattice is consumed by chapter 4238 and chapter 735 only through candidate evidence rows, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention. A downstream confluence reader must therefore keep the handoff as a finite evidence packet rather than reading it as a completed normalization theorem or as an untracked frontier shortcut. The candidate-normalization confluence handoff consumes the parallel-diamond frontier of chapter 4238 together with the critical-path route of chapter 765 before any confluence-facing read is exported. This child route records the dependency lattice: the MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet may pass candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming onward only while those MetaCIC frontier siblings remain visible.

Theorem 4737.47 (MetaCIC candidate handoff forward route to the parallel-diamond frontier)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet that is consumed as a local confluence handoff must expose the parallel-diamond frontier sibling of chapter 4238 before the handoff is read as downstream MetaCIC evidence. In carrier form the handoff rows*

$$K, N, F, C, D, B, T, R, P, L$$

remain visible while the sibling frontier supplies the retained residual-substitution premise, critical-pair envelope, candidate-join row, residual-frontier row, bounded checker rows, obstruction row, transport, replay, provenance, and local NameCert *rows. The route is a finite dependency route for candidate normalization and confluence readers; it does not assert full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency.*

Proof. Read the carrier of the handoff packet first. The candidate evidence, normal endpoint, frontier, residual confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and naming rows are the only rows available to the consumer.

Next read the sibling frontier at chapter 4238. Its displayed frontier budget keeps the retained residual-substitution premise and obstruction rows visible before any local diamond route is consumed.

The two packets therefore meet only along finite visible rows. Since neither packet contains a row for unrestricted confluence, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency, the forward route remains a dependency edge rather than a global MetaCIC endpoint. □

The MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ handoff consumes the normalization-candidate frontier of chapter 892, the bounded Church–Rosser fragment of chapter 8560, and the confluence audit rows of chapter 735 before exporting its candidate-mediated handoff. The dependency route keeps candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming visible as finite MetaCIC data, not as a detached normalization theorem or a global confluence endpoint.

Theorem 4737.48 (MetaCIC critical-path and parallel-frontier handoff route)**.** *Every confluence-facing consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must first read the MetaCIC critical-path surface of chapter 765 and the parallel-diamond frontier route of chapter 4238. Only after those sibling surfaces are displayed may the handoff be treated as a confluence-facing packet over candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming. This is a finite MetaCIC route, not subject reduction, full Church–Rosser, unrestricted conversion decidability, quotient syntax equality, or kernel consistency.*

Proof. Project the handoff carrier of Definition 5547.2. Its rows expose audit anchoring, candidate evidence, normal endpoint data, frontier data, residual confluence data, decidable-boundary data, blocked-edge retention, transport, replay, provenance, and local naming.

Apply the boundary theorem Definition 5547.3. It requires a confluence-facing read to keep the candidate gate, residual envelope, decidable boundary, and blocked-edge rows visible before an endpoint row can be consumed.

The sibling route is therefore read in two steps. The MetaCIC critical-path chapter supplies the upstream audit route, and the parallel-diamond frontier supplies the confluence-facing frontier route. The handoff can then pass finite candidate and residual data onward, but no row in

the carrier turns that route into subject reduction, full Church–Rosser, unrestricted conversion decidability, quotient syntax equality, or kernel consistency. □

Theorem 4737.49 (MetaCIC candidate handoff normalization-audit forward link)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read the* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet of chapter 750. In particular, the handoff consumes the audit carrier of Definition 750.1, the candidate evidence row of Definition 750.3, the closed projection route of Definition 750.5, the finite strong-normalization scope of Definition 750.4, the application boundary of Definition 750.6, and the confluence refusal row of Definition 750.9 before candidate normalization is routed to any confluence-facing endpoint.*

Proof. Start with the normalization-audit carrier. The rows $R, C, S, A, N, F, H, T, P, L$ separate candidate evidence, closed projection, finite strong-normalization scope, application boundary, confluence refusal, replay, transport, provenance, and local naming.

Apply the candidate boundary and closed-term projection route. They force any normalization read to expose candidate evidence and checked closed projection before the finite SN inventory can be consumed.

Next apply the finite SN scope and candidate-mediated SN closure theorems. These rows keep the SN export inside the candidate ledger and retain the application boundary, so the handoff cannot convert candidate normalization into an evidence-free endpoint.

Finally use the confluence refusal split. The refusal row N remains visible when the handoff approaches a confluence-facing consumer, excluding global confluence, unrestricted subject reduction, quotient syntax equality, and host evaluator evidence from the forwarded route. □

Theorem 4737.50 (MetaCIC candidate normalization confluence handoff normal-form obligation)**.** *Any normal-form read exported by an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier must first expose the candidate-normalization row, the residual-diamond row, and the sibling confluence route. Concretely, the read must pass through the candidate gate and frontier rows of Definition 5547.3, the premise-exposure and normal-endpoint rows of Definition 5547.5 and Definition 5547.8, the residual-diamond retention row of Definition 5547.6, the bounded candidate-diamond route of Definition 5547.44, and the sibling confluence route of Definition 5547.89 before any normal-form endpoint is available. The obligation is a finite premise-exposure row only; it does not assert host normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency.*

Proof. Project the carrier from Definition 5547.2. A normal-form read can see only the candidate, frontier, residual, decidable, blocked-edge, transport, replay, provenance, and local naming rows displayed in that carrier.

Apply the boundary and premise-exposure theorems. They force the candidate gate and normalization frontier to appear before the normal endpoint is consumed, and they keep the residual envelope, decidable boundary, and blocked-edge rows visible for confluence-facing reads.

Next apply normal-endpoint readback and residual-diamond retention. These rows show that an endpoint can be handed to confluence consumers only while candidate evidence, residual data, decidable-boundary data, and blocked edges remain present in the same packet.

Finally use the diamond-budget dependency and sibling confluence route. The candidate diamond supplies bounded local peak and join-budget rows, and the sibling route exposes the critical-path, normalization-frontier, confluence-audit, and parallel-diamond residual surfaces before the normal-form read is available. Since no carrier coordinate contains a host-level normalization or global confluence theorem, the obligation remains the finite handoff row stated above. □

Theorem 4737.51 (MetaCIC candidate normalization confluence handoff obstruction retention)**.** *Every consumer of the candidate-normalization normal-form row exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the residual-substitution premise, candidate diamond budget dependency, cut-height budget route, and parallel frontier route before treating that row as confluence-facing evidence. The retained route is the finite chain through Definition 5547.44, Definition 5547.46, Definition 5547.73, and the residual-diamond bridge route. It does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the residual budget theorem Definition 5547.57. It forces candidate evidence, residual-diamond retention, decidable-boundary data, blocked-edge rows, bounded local peaks, join-budget rows, and the sibling confluence route to remain visible before a normal-form endpoint is exported.

Apply the candidate diamond budget dependency Definition 5547.44. The normal-form row is therefore read only after bounded local peak and join-budget data have been displayed; it is not an evidence-free confluence endpoint.

Next use the cut-height budget route and the parallel-diamond frontier route, Definition 5547.46 and Definition 5547.73. These routes keep the normalization frontier, residual confluence, decidable boundary, blocked-edge rows, and frontier-facing residual-substitution premise attached to the handoff.

Finally consume the residual-diamond bridge route recorded in the sibling route file. The bridge sends only finite residual-diamond dependency data onward while preserving residual-substitution visibility. Since the carrier has no coordinate for subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, every confluence-facing read retains the obstruction instead of erasing it. □

Theorem 4737.52 (MetaCIC candidate normalization normal-form budget)**.** *Every accepted normal-form endpoint exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier must retain the candidate gate, residual-diamond row, cut-height budget, parallel-diamond frontier route, and sibling confluence route as one finite budget surface before it can be used as local confluence evidence. The endpoint is therefore consumed only through the normal-form obligation, obstruction-retention row, diamond-budget route, cut-height budget route, and parallel-diamond frontier route; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, or consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint downstream of the candidate-normalization row, the residual-diamond row, and the sibling confluence route before any consumer can read it.

Apply obstruction retention, Definition 5547.51. The residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route remain attached to the same handoff row rather than being discharged by the endpoint.

Next use the diamond-budget route and cut-height budget route, Definition 5547.44 and Definition 5547.46. These rows expose bounded local peaks, join-budget data, normalization-frontier data, residual confluence, decidable-boundary routing, and blocked-edge retention as finite budget coordinates.

Finally apply the parallel-diamond frontier route, Definition 5547.73, and the sibling confluence route Definition 5547.89. They keep the frontier and sibling residual surfaces visible before the endpoint is treated as local confluence evidence. Since none of the cited rows contains a

global Church–Rosser theorem, subject reduction, quotient syntax equality, evaluator equality, or consistency, the endpoint remains the retained finite budget surface stated above. □

Theorem 4737.53 (MetaCIC candidate normalization normal-form budget)**.** *Every local-confluence consumer of an accepted normal-form endpoint exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the candidate gate, residual-diamond row, residual-substitution premise, cut-height budget, parallel frontier route, sibling confluence route, and local* NameCert *row as one finite budget packet. The endpoint is therefore consumed only through the normal-form obligation, obstruction-retention row, candidate diamond budget route, cut-height budget route, and parallel-diamond frontier route before it can be read as local confluence evidence.*

This retained packet explicitly excludes unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, and kernel consistency.

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint after the candidate gate, residual-diamond row, and sibling confluence route, so the consumer cannot read the endpoint as a detached normal form.

Apply obstruction retention, Definition 5547.51. That row keeps the residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route attached to the endpoint as confluence-facing evidence.

Use the candidate diamond budget route, Definition 5547.44, together with the cut-height budget route Definition 5547.46. They expose the bounded local peak, join-budget, cut-height, normalization frontier, residual confluence, decidable boundary, and blocked-edge rows as finite budget coordinates.

Finally apply the parallel-diamond frontier route, Definition 5547.73, and the sibling confluence route Definition 5547.89. The local NameCert row names exactly that finite packet. Since none of these rows contains unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency, the consumer receives only the retained normal-form budget. □

Theorem 4737.54 (MetaCIC residual normal-form frontier)**.** *An accepted normal-form endpoint of a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet may be consumed as local confluence evidence only while one finite frontier packet still displays the candidate gate, residual-diamond row, residual-substitution premise, cut-height budget, parallel frontier route, and sibling confluence route. The endpoint is therefore read through the normal-form obligation, obstruction retention, normal-form budget, diamond-budget route, cut-height route, and parallel frontier route before any local-confluence consumer sees it.*

The frontier packet preserves obstruction rows. It does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Proof. Start with the normal-form obligation, Definition 5547.50. It places the endpoint after the candidate gate, residual-diamond row, and sibling confluence route, so the endpoint is never presented as an isolated normal form.

Apply obstruction retention and the normal-form budget theorem, Definition 5547.51 and Definition 5547.52. These rows keep the residual-substitution premise, residual-diamond row, cut-height budget, parallel frontier route, and sibling confluence route attached to the same finite handoff packet.

Next use the diamond-budget and cut-height routes, Definition 5547.44 and Definition 5547.46. They expose bounded local peak, join-budget, and cut-height data as visible frontier coordinates rather than as hidden global confluence assumptions.

Finally apply the parallel-diamond frontier route, Definition 5547.73. Together with the sibling confluence route, it leaves the residual and frontier rows visible when the endpoint is consumed as local confluence evidence. Since the carrier has no coordinate for subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency, the endpoint remains a finite obstruction-preserving frontier read. □

Theorem 4737.55 (MetaCIC candidate normalization normal-form obligation route)**.** *Every normal-form endpoint read exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier must pass through the candidate gate, residual-diamond retention, cut-height budget, parallel-diamond frontier route, sibling confluence route, and local* NameCert *rows before it can be used as confluence-facing evidence. The route retains the residual and frontier evidence attached to the endpoint; it does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint downstream of the candidate gate, residual-diamond row, and sibling confluence route before any consumer can read it.

Apply obstruction retention, Definition 5547.51. The retained residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route remain attached to the endpoint instead of being discharged by the normal-form read.

Finally use the normal-form budget theorem Definition 5547.52. It keeps the cut-height budget, parallel-diamond frontier surface, sibling confluence route, and local naming rows visible before the endpoint is treated as confluence-facing evidence. Since none of the cited rows contains subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency, the read exports only the retained finite obligation route. □

Theorem 4737.56 (MetaCIC retained-premise normal-form handoff)**.** *Every strong-normalization-facing consumer of the normal-form endpoint exported by an accepted* MetacicCandidateNormalizationConfluenceHan *packet must retain the candidate gate, the residual-diamond retention row, the decidable-boundary row, and the sibling confluence route. The consumer may read the bounded normal-form endpoint only as a finite handoff through those rows; it does not receive candidate-free normalization, quotient syntax equality, host evaluator soundness, unrestricted subject reduction, global confluence, or closed consistency.*

Proof. Apply the normal-form obligation of Definition 5547.50. It forces a normal-form read through the candidate-normalization row, residual row, and sibling confluence route before the endpoint is available.

Next use residual-diamond retention from Definition 5547.6. That row keeps the residual envelope, decidable boundary, and blocked-edge data visible while the endpoint is treated as a confluence-facing read.

The diamond-budget route of Definition 5547.44 supplies the bounded local peak and join-budget rows. These are the finite rows that a strong-normalization-facing consumer may inspect; they do not discharge the candidate gate or replace it with a global normalization theorem.

Finally apply the sibling confluence route of Definition 5547.89. It keeps the critical-path, normalization-frontier, confluence-audit, and parallel-diamond residual surfaces visible before the handoff reaches a consumer. Hence the endpoint remains a retained-premise normal-form handoff and no candidate-free normalization or quotient syntax equality is exposed. □

Theorem 4737.57 (MetaCIC candidate-normalization residual budget)**.** *Any normal-form endpoint exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet to a strong-normalization-facing consumer must retain, before the endpoint is read, the candidate evidence, residual-diamond retention row, decidable-boundary data, blocked-edge rows, bounded local peak rows, join-budget rows, and the sibling confluence route. The endpoint is therefore a finite residual-budget read, not candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50 and the retained-premise handoff of Definition 5547.56. Together they force candidate evidence and the sibling confluence route to remain visible before a normal-form endpoint is exported.

Use residual-diamond retention from Definition 5547.6. It keeps the residual envelope, decidable-boundary data, and blocked-edge rows attached to the endpoint read rather than allowing a detached normal-form token.

Finally apply the diamond-budget route of Definition 5547.44. Its bounded local peak and join-budget rows are the finite budget a strong-normalization-facing consumer may inspect. Since the cited rows exhaust the exported endpoint surface, none of candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency is supplied. □

Theorem 4737.58 (MetaCIC candidate-normalization residual exhaustion)**.** *Every accepted* MetacicCandidateNorm *handoff row whose candidate set, bounded local peak, join-budget row, residual envelope, decidable-boundary row, blocked-edge row, transport, replay, provenance, and local* NameCert *are displayed has an exhaustive finite endpoint surface. It either forwards a retained premise through the candidate and normalization frontier rows, or emits the blocked residual edge as an obstruction. No consumer may erase that obstruction and read global confluence, unrestricted subject reduction, quotient syntax equality, closed consistency, or a candidate-free normal-form theorem.*

Proof. Project the displayed handoff carrier from the parent chapter. The candidate set, frontier rows, residual envelope, decidable boundary, blocked-edge row, transport, replay, provenance, and local naming data are the only rows available to the consumer.

Apply the residual budget theorem Definition 5547.57. It requires the candidate evidence, residual-diamond retention, decidable-boundary data, blocked-edge rows, bounded local peak rows, join-budget rows, and sibling confluence route to remain visible before any endpoint is exported.

Now read the displayed forward-or-blocked row as finite data. In the forward case the retained premise is carried through the candidate and frontier rows. In the blocked case the residual edge remains an obstruction attached to the endpoint read. The carrier's refused endpoints exclude candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, closed consistency, and any consumer route that erases the obstruction. □

Theorem 4737.59 (MetaCIC candidate-normalization residual-join choice-free handoff)**.** *When an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet exposes finite residual budget exhaustion and a displayed local-confluence certificate route, the residual-join endpoint is read as a finite handoff row. The endpoint is not a selected global normal form: it remains attached to the candidate evidence, normal-form obligation, residual budget, residual exhaustion surface, local confluence certificate route, transport, replay, provenance, and local* NameCert *rows.*

Proof. Start with the normal-form obligation of Definition 5547.50. It keeps candidate evidence and the sibling confluence route visible before any endpoint is public.

Apply the residual budget theorem, Definition 5547.57. The bounded local peak, join-budget, residual-diamond retention, decidable-boundary, and blocked-edge rows remain finite budget data rather than a choice of global normal form.

Now use residual exhaustion, Definition 5547.58. The forward case carries the retained premise through the candidate and frontier rows, while the blocked case keeps the residual edge as an obstruction attached to the endpoint.

Finally read the local confluence certificate route through the sibling confluence surface already cited by the normal-form and residual-budget rows. Since the endpoint is assembled only from those displayed rows and the structural transport, replay, provenance, and naming data, it is a choice-free residual-join handoff. □

Definition 4737.60 (Normalization-confluence bridge carrier)**.** *A normalization-confluence bridge carrier for* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *is a finite* BHist *packet*

$$B = (A, K, N, J, Q, D, O, H, C, P, L)$$

with the following displayed rows:

1. *A is the MetaCIC audit anchor for the closed typed term boundary.*
2. *K is the candidate-normalization evidence row used to expose candidate-mediated strong normalization data.*
3. *N is the bounded normal endpoint row reached from K.*
4. *J is the residual local-join budget row, recording only the bounded join checks available at the confluence frontier.*
5. *Q is the closed checker row that verifies the finite term and type boundary before any endpoint or join row is consumed.*
6. *D is the obstruction row retaining failed residual edges and blocked comparisons.*
7. *O records componentwise* hsame *transport for the audit anchor, candidate evidence, endpoint, join budget, closed checker, obstruction, replay, provenance, and local naming rows.*
8. *H records displayed* Cont *replay through $A, K, N, J, Q, D, O, P, L$.*
9. *C is the* Pkg *provenance over* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *P is the local public-route packet row for the bounded bridge budget.*
11. *L is the local* $\mathsf{NameCert}$ *row.*

The carrier has no coordinate for full confluence, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, host evaluator soundness, or kernel consistency.

Theorem 4737.61 (Normalization-confluence bridge obligations)**.** *Every accepted normalization-confluence bridge carrier exposes candidate-mediated strong-normalization evidence and bounded local confluence rows only through*

$$A, K, N, J, Q, D, O, H, C, P, L.$$

The candidate row K may feed the bounded endpoint row N, and the residual local-join budget J may feed the confluence frontier only after the closed checker Q and obstruction row D remain

visible. The bridge does not prove full confluence, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, host evaluator soundness, or kernel consistency.

Proof. Project the bridge carrier. The only normalization-facing rows are the audit anchor A, candidate evidence K, bounded endpoint N, residual local-join budget J, closed checker Q, and obstruction row D.

Use the candidate-normalization handoff boundary of Definition 5547.3. It keeps candidate evidence and frontier data visible before bounded endpoint or residual rows can be consumed.

Apply residual-diamond retention and decider-boundary routing from Definition 5547.6 and Definition 5547.21. These force J, Q, D to remain finite frontier rows rather than global confluence or subject-reduction evidence.

Finally read the structural rows O, H, C, P, L. They transport, replay, package, route, and name the same bounded bridge budget, so a consumer receives candidate-mediated normalization and local confluence frontier data without extracting any refused global MetaCIC endpoint. □

Theorem 4737.62 (MetaCIC candidate typed-diamond route)**.** *Every typed-diamond consumer of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *bridge carrier must retain candidate evidence, the closed checker, the residual local-join budget, the decidable boundary, and blocked-edge rows before consuming bounded endpoints. The route may pass through residual-diamond and parallel-diamond sibling frontiers, but it refuses full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, and kernel consistency.*

Proof. Project the bridge carrier of Definition 5547.60. Its candidate evidence row K, bounded endpoint row N, residual local-join budget J, closed checker row Q, obstruction row D, and structural rows O, H, C, P, L are separate coordinates.

Invoke the bridge obligations of Definition 5547.61. They force candidate evidence and bounded local confluence rows to remain visible, and they place the closed checker and obstruction row before endpoint or join rows are consumed.

Route the typed-diamond read through residual-diamond retention, Definition 5547.6. This keeps the residual envelope, decidable boundary, and blocked-edge data attached to the endpoint comparison rather than letting N stand alone.

Finally use the parallel-diamond forward link, Definition 5547.1. It sends the consumer through the parallel-diamond frontier as a guarded boundary route. The structural rows then transport, replay, source, and name only this bounded route, so no full Church–Rosser theorem, unrestricted subject-reduction proof, quotient syntax equality, host evaluator soundness, or kernel-consistency endpoint is exported. □

Theorem 4737.63 (MetaCIC candidate handoff downstream route)**.** *Every public read of* MetacicCandidateNormal *is consumed by chapter 735 and chapter 4238 only after candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention have been exposed in the same finite handoff packet. The downstream route also consumes the residual-diamond sibling chapter 5126 and the confluence-audit packet chapter 735 before exporting a bounded endpoint read. The route exports a downstream confluence dependency surface, not an unconditional strong-normalization theorem, full Church–Rosser theorem, unrestricted subject-reduction theorem, global conversion decider, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes the candidate row K, frontier row F, residual envelope C, decidable-boundary row D, and blocked-edge row B before any bounded endpoint or comparison row is consumed.

Apply premise exposure from Definition 5547.5. It prevents a confluence-facing consumer from reading the handoff unless the candidate evidence and residual boundary remain visible in the same packet.

Use residual-diamond retention, Definition 5547.6, to route the downstream read through the parallel-diamond frontier. The frontier receives only the retained residual-diamond handoff, not a global confluence theorem.

Finally apply normal-endpoint readback, Definition 5547.8. The confluence audit packet may consume the bounded endpoint only after C, D, B, T, R, P, L remain visible. Thus both downstream chapters read the same candidate-mediated handoff route and no excluded global endpoint is exported. □

Theorem 4737.64 (MetaCIC candidate-mediated confluence route)**.** *Any confluence-facing consumer of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and normal-endpoint readback. Only after those rows remain visible may the consumer pass the bounded route to chapter 735 or chapter 4238. The route does not claim strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with Definition 5547.63. It already routes candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention to the confluence audit and parallel-diamond frontier consumers.

Apply premise exposure from Definition 5547.5. A confluence-facing consumer cannot read the bounded endpoint before the candidate and residual boundary rows remain displayed in the same packet.

Use residual-diamond retention from Definition 5547.6. It keeps the residual envelope and blocked-edge rows on the route that is handed to the parallel-diamond frontier.

Finally apply normal-endpoint readback from Definition 5547.8. The endpoint is therefore bounded by the visible rows C, D, B, T, R, P, L, so the exported route is a candidate-mediated confluence audit surface rather than any of the refused global MetaCIC endpoints. □

Theorem 4737.65 (MetaCIC candidate residual-obstruction retention)**.** *Every confluence-facing endpoint read of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must retain candidate evidence, the residual envelope, the decidable-boundary row, the blocked-edge row, and normal-endpoint readback before any bounded route is exported to MetaCIC confluence consumers. The exported route is therefore a retained obstruction surface through the candidate packet, not an unrestricted Church–Rosser theorem, subject-reduction theorem, quotient syntax equality, evaluator soundness claim, or closed-consistency endpoint.*

Proof. Start with the downstream route of Definition 5547.63. It exposes candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention in the same finite packet before either confluence consumer can read the handoff.

Apply premise exposure from Definition 5547.5. A bounded endpoint is not available to a confluence-facing consumer unless the candidate evidence and residual boundary remain visible.

Use residual-diamond retention from Definition 5547.6. This keeps the residual envelope and blocked-edge row attached to the route that reaches the parallel-diamond frontier.

Finally apply normal-endpoint readback from Definition 5547.8 and then the confluence route of Definition 5547.64. The endpoint readback is exported only with the retained rows, so the consumer receives a bounded obstruction route and none of the excluded global MetaCIC conclusions. □

Theorem 4737.66 (MetaCIC candidate normalization orphan route)**.** *Public reads of* MetacicCandidateNormalizati *pass to chapter 735 and chapter 4238 only after candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and normal-endpoint readback remain displayed in the same handoff packet. The route is a concrete downstream dependency theorem for the confluence lattice; it does not export a global Church–Rosser theorem, strong-normalization theorem, unrestricted subject-reduction theorem, quotient syntax equality, evaluator soundness claim, or closed-consistency endpoint.*

Proof. First apply the boundary theorem Definition 5547.3. It places the candidate row, frontier row, residual envelope, decidable-boundary row, and blocked-edge row upstream of every bounded endpoint read.

Next use premise exposure from Definition 5547.5. This keeps candidate evidence and residual-envelope visibility displayed before either downstream consumer may read the handoff.

Then apply residual-diamond retention from Definition 5547.6. The retained residual envelope and blocked-edge row are exactly the rows required by the parallel-diamond frontier.

Finally use normal-endpoint readback from Definition 5547.8, together with Definition 5547.65. The confluence audit packet and parallel-diamond frontier therefore receive a displayed candidate-mediated dependency route, not any of the refused global MetaCIC conclusions. □

The candidate-normalization confluence handoff reads the parallel-diamond frontier of chapter 4238 through the local candidate-diamond-budget and residual-frontier siblings before any handoff row is made public. It also consumes chapter 765 as the critical-path source that fixes the typed-window and residual-frontier rows before the parallel-diamond frontier is read. This route keeps candidate evidence, residual-envelope visibility, decidable-boundary rows, and blocked-edge retention visible inside the finite MetaCIC packet.

Theorem 4737.67 (MetaCIC candidate normalization handoff obligations)**.** *Every candidate-normalization confluence handoff read factors through* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$*, the candidate-diamond-budget route, and the residual-frontier route siblings before any confluence handoff row is public. The handoff may then expose candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming, but it cannot bypass those frontier siblings.*

Proof. Project the handoff carrier of Definition 5547.2. Its public rows record candidate evidence, bounded endpoint data, residual confluence data, decidable-boundary data, blocked-edge retention, transport, replay, provenance, and local naming.

Use the local candidate-diamond-budget route and residual-frontier route siblings. The former exposes bounded local-peak and join-budget rows before a closed-normal endpoint is consumed, while the latter keeps the parallel-diamond frontier visible before a confluence-facing handoff is exported.

The candidate-normalization side also consumes the normalization audit map of chapter 9383; that map supplies the candidate-mediated SN budget before residual confluence visibility is exposed.

Finally read chapter 4238 as the upstream frontier surface. Since the carrier has no row that makes the handoff public without those sibling routes, every accepted read factors through the displayed frontier dependency before presenting confluence handoff data. □

Theorem 4737.68 (MetaCIC parallel frontier residual-bridge consumption)**.** *Any residual-diamond bridge read of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first expose the candidate diamond budget, cut-height budget,* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *route,* MetaCICParallelDiamondFrontier

dependency, residual frontier reverse dependency, and seed obligation boundary. The bridge may then consume the finite rows $K, F, C, D, B, T, R, P, L$ only while residual-substitution premises remain visible; it cannot be read as subject reduction, evaluator equality, quotient term equality, global normalization, unrestricted confluence, or kernel consistency.

Proof. Begin with the candidate diamond budget route of Definition 5547.44 and the cut-height budget route of Definition 5547.46. These expose bounded local peaks, join-budget rows, normalization-frontier data, residual confluence data, decidable-boundary data, and blocked-edge retention before an endpoint-facing read is available.

Next read the critical-path and frontier routes. The MetaCIC critical-path handoff of Definition 5547.48 fixes the upstream audit direction, while the parallel-diamond frontier route of Definition 5547.73 keeps the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport, replay, provenance, and local naming visible.

Use the residual frontier reverse dependency and the seed obligation boundary. The reverse dependency keeps the parallel frontier as dependency data rather than as an isolated conclusion, and Definition 5547.85 exposes $K, F, C, D, B, T, R, P, L$ before bounded endpoint consumption.

Finally consume the residual-diamond bridge route. Since the bridge reads only the displayed residual-diamond dependency surface and the carrier has no coordinate for subject reduction, evaluator equality, quotient term equality, global normalization, unrestricted confluence, or kernel consistency, every accepted bridge read retains the stated premises before any confluence endpoint is consumed. □

Theorem 4737.69 (MetaCIC candidate confluence orphan route)**.** *Every accepted* MetacicCandidateNormalizationC *read that exports the normalization/confluence bridge route first consumes the residual-frontier rows of chapter 4238. The consumed rows are the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming surface; the handoff does not export subject reduction, unrestricted confluence, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the handoff carrier of Definition 5547.2. Its candidate, bounded endpoint, residual confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows can be read only as a finite handoff packet.

Next read the upstream frontier through chapter 4238. That chapter displays the retained premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before a confluence-facing consumer can inspect the local-diamond handoff.

Finally combine this frontier read with the existing bridge-budget route of Definition 5547.68. The bridge route consumes exactly the displayed residual-frontier dependency before exporting normalization/confluence data, so the handoff remains inside the finite dependency lattice and none of the refused global endpoints is exposed. □

Parallel-diamond dependency route. The candidate-normalization confluence handoff consumes chapter 4238 for the residual critical-pair windows that remain visible before any confluence handoff row is exposed. The frontier packet supplies the typed local-join boundary and residual-budget ledger, while the handoff packet only routes candidate normalization, normal endpoints, blocked edges, transport, replay, provenance, and local naming through that boundary. Hence the handoff may advertise a confluence-facing row only after the parallel-diamond frontier has fixed

the residual window and excluded full confluence, subject reduction, evaluator equality, quotient syntax equality, and closed consistency.

Theorem 4737.70 (MetaCIC candidate-normalization handoff parallel-diamond dependency)**.** *Every accepted* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *read that consumes the parallel-diamond frontier of chapter 4238 does so only through the displayed critical-pair row, residual-join row, candidate-normalization row, bounded-check row, obstruction row, componentwise transport, replay, provenance, and local* NameCert *rows. The route is a finite sibling dependency on* MetaCICParallelDiamondFrontier$^{\uparrow}$*, not a duplicate confluence theorem, subject-reduction theorem, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel-consistency claim.*

Proof. Read the handoff carrier through its displayed candidate-normalization and bounded endpoint rows. A confluence-facing consumer cannot inspect the handoff before the critical-pair and residual-join rows supplied by chapter 4238 have fixed the frontier boundary.

Next preserve the bounded-check and obstruction rows. These rows keep failed residual-substitution, blocked-edge, and closedness obligations visible as retained data rather than discharging them into a global parallel-diamond result.

Finally transport the same route through the componentwise hsame data, displayed replay, provenance, and local NameCert row. Since no coordinate exports full confluence, subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency, the dependency is exactly the finite sibling route stated above. □

Theorem 4737.71 (MetaCIC candidate-normalization handoff forward link)**.** *Every* MetaCICCandidateNormalizati *consumer that exports a residual join to MetaCIC normalization consumers must first display the parallel-diamond frontier of chapter 4238 or the residual-frontier route carried by that frontier. Only after the candidate evidence, residual-envelope row, decidable-boundary row, blocked-edge ledger, transport row, replay row, provenance row, and local* NameCert *row have been read through that frontier may the confluence handoff be public.*

Proof. Read the handoff carrier of Definition 5547.2. Its confluence-facing rows are candidate evidence, bounded endpoint data, residual envelope, decidable boundary, blocked-edge retention, and the structural transport, replay, provenance, and naming rows.

Use the residual-diamond retention theorem Definition 5547.6. It keeps candidate evidence, residual envelope, decidable boundary, and blocked-edge rows visible before any bounded endpoint is treated as joinable.

The sibling routes in this hub then point the retained residual data to chapter 4238. That frontier is the public place where the residual-frontier route and local parallel-diamond budget are displayed.

Since the carrier has no coordinate for a public confluence handoff detached from those rows, any consumer exporting a residual join must pass through the frontier route before the handoff is public. □

Every candidate-normalization confluence handoff consumer must read the residual diamond and local-join data through chapter 4238 before exporting a confluence-facing handoff. The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet therefore presents bounded endpoint, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows only after the parallel-diamond frontier has supplied the displayed residual-diamond and local-join surface.

Theorem 4737.72 (Parallel-diamond frontier dependency bridge)**.** *Every accepted* MetaCICCandidateNormalizatio*n carrier reads the residual join route through chapter 4238 before it exports a normalization-confluence budget to a downstream candidate-normalization consumer.*

Proof. The handoff carrier already exposes candidate-normalization, residual-join, and confluence-budget rows through its sibling route files. The parallel-diamond frontier is the only displayed sibling that licenses the residual join as a diamond-facing dependency rather than as an isolated normalization claim.

Thus the confluence budget is exported only after the residual join has passed through the parallel-diamond frontier route. No kernel-only or local-normalization row can stand in for that horizontal dependency. □

Theorem 4737.73 (Candidate normalization handoff parallel-diamond frontier route)**.** *Any* MetaCICCandidateNor*consumer that exports a local diamond budget must first cite chapter 4238. The export preserves the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local* NameCert *row before the handoff may be read as a frontier-facing diamond route.*

Proof. Read the existing parallel-diamond dependency route for the handoff. It places the candidate-normalization budget on the same finite route as the frontier packet rather than on a global confluence theorem.

The cited frontier chapter supplies the public location for that route. A consumer that exports a local diamond budget therefore reaches the frontier only after preserving the residual-substitution premise, candidate join, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Thus the handoff is a displayed dependency through MetaCICParallelDiamondFrontier$^{\uparrow}$. It does not export unrestricted residual substitution, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 4737.74 (MetaCIC candidate normalization parallel-diamond frontier seed dependency)**.** *Before any* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *read may use the parallel frontier of chapter 4238, it must consume the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, and obstruction row exposed by that sibling. The handoff route then reads its bounded endpoint, residual envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows as downstream data rather than as a replacement for the parallel-diamond frontier seed surface.*

Proof. Read the parallel-diamond frontier chapter first. Its seed surface keeps the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, and structural NameCert rows visible before any local confluence-facing read is available.

Apply the handoff carrier and candidate diamond budget routes in this chapter. They expose bounded endpoint, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows only after the upstream frontier dependency has been displayed.

Finally compare the dependency order. Since the candidate-normalization handoff has no coordinate for discharging residual substitution or replacing the frontier obstruction row, every accepted read that uses chapter 4238 must keep those sibling rows ahead of the handoff read. □

Theorem 4737.75 (MetaCIC candidate-normalization orphan parallel-frontier route)**.** *Any confluence-facing read of* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *reaches the* MetaCICParallelDiamondFrontier$^{\uparrow}$

consumer of chapter 4238 through the candidate evidence row, residual-envelope row, decidable-boundary row, and blocked-edge row before bounded endpoint data is public. The route preserves transport, replay, provenance, and local NameCert *rows, and it exports no global confluence theorem, quotient term equality, evaluator equality, unrestricted residual substitution, or kernel consistency claim.*

Proof. Start with the handoff boundary surface. The carrier of Definition 5547.2 and the residual-diamond retention theorem keep candidate evidence, residual envelope, decidable boundary, and blocked-edge retention visible before any endpoint data is consumed.

Next apply the forward-link route of Definition 5547.71. It sends the retained residual data to the parallel-diamond frontier rather than to a detached confluence oracle.

Finally read the frontier consumer at chapter 4238. The transport, replay, provenance, and local naming rows remain attached to the same finite handoff, so the orphan-facing read is a sibling dependency on MetaCICParallelDiamondFrontier$^{\uparrow}$ and not an export of any excluded host theorem. □

Theorem 4737.76 (Candidate normalization retained frontier route)**.** *Every candidate-normalization confluence handoff that consumes a bounded local diamond must first read the retained-premise frontier of chapter 4238. In carrier notation the consumer reaches the handoff rows $K, F, C, D, B, T, R, P, L$ only after the sibling frontier has exposed the retained premise row, candidate-join row, residual-frontier row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local* NameCert *row.*

The route is a finite candidate-normalization dependency. It does not supply residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the local handoff carrier of Definition 5547.2. Its rows $K, F, C, D, B, T, R, P, L$ expose candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

Next read the sibling frontier. The parallel-diamond frontier chapter supplies the retained-premise and candidate-join route consumed by this handoff, so the local diamond budget cannot be inspected until those frontier rows are visible.

Finally apply the existing parallel-diamond frontier route of Definition 5547.73. It keeps the residual-substitution premise, bounded-check row, obstruction row, transport, replay, provenance, and local naming attached to the candidate-normalization handoff. Therefore the new route is a child dependency through MetaCICParallelDiamondFrontier$^{\uparrow}$, not an independent confluence or kernel endpoint. □

Theorem 4737.77 (MetaCIC candidate normalization confluence handoff public export)**.** *The* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet may be exported publicly only as the finite route through* MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, *the residual join budget,* MetaCICDecidableBoundary$^{\uparrow}$, *blocked-edge retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. For an accepted handoff carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$, any public read must keep candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge data, transport, replay, provenance, and local naming visible before the endpoint row is consumed.*

The exported route is not global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.

Proof. Project the carrier of Definition 5547.2. Its public rows are $A, K, N, F, C, D, B, T, R, P, L$, so a public export starts from the audit anchor, candidate evidence, bounded endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name.

Use the boundary and candidate handoff routes. The boundary theorem Definition 5547.3 keeps candidate evidence and normalization-frontier data visible before endpoint consumption, while Definition 5547.45 sends the same packet through $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, residual join scope, and candidate-normalization budget rows.

Next apply the residual budget and route-factorization surfaces, Definition 5547.57 and Definition 5547.83. They force every residual join or parallel-diamond read to retain the bounded budget, decidable boundary, and blocked-edge rows rather than replacing them by a global confluence theorem.

Finally read T, R, P, L. Componentwise transport, displayed replay, package provenance, and local naming preserve the finite route already displayed by the sibling theorems. Since no carrier coordinate stores global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency, the only public export is the stated finite route. □

The candidate-normalization confluence handoff consumes the parallel-diamond frontier of chapter 4238 through the displayed residual-join, critical-pair, and candidate-diamond rows before any confluence-facing endpoint is read. In carrier terms, the $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ packet may route its candidate evidence, residual envelope, decidable-boundary row, blocked-edge ledger, transport, replay, provenance, and local $\mathsf{NameCert}$ onward only while the sibling frontier still exposes its residual-substitution premise, local critical-pair surface, candidate join, bounded checker rows, and obstruction row. The bridge is therefore a finite residual-diamond dependency route, not subject reduction, unrestricted Church–Rosser, quotient syntax equality, evaluator equality, global conversion decidability, or kernel consistency. The candidate-normalization confluence handoff consumes the residual-frontier and finite candidate-diamond rows before it exposes its local confluence route. In particular, chapter 4238 supplies the parallel frontier surface that the handoff reads as dependency data rather than as an isolated conclusion.

Theorem 4737.78 (MetaCIC candidate-normalization residual-join frontier)**.** *An accepted* $\mathsf{MetacicCandidateNorma}$ *handoff may consume a residual-join frontier only after the candidate set, normalization window, residual-substitution row, join-locality row, and strong-normalization handoff have all been displayed. The frontier read remains a finite route through the candidate-normalization carrier, residual-substitution boundary, local join ledger, bounded normal-form endpoint,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. Its candidate evidence, bounded endpoint, residual envelope, decidable-boundary route, blocked-edge row, transport, replay, provenance, and local naming rows are visible before any frontier read is public.

Read the candidate and window rows first. The candidate set supplies the finite normalization candidates, and the normalization window fixes the bounded endpoint at which a residual join can be inspected.

Next consume the residual-substitution and join-locality rows. The residual-substitution boundary keeps substitutions attached to the displayed residual envelope, while the local join ledger keeps the join inside the candidate window rather than a host equality space.

Finally attach the strong-normalization handoff. It makes the bounded normal endpoint available to the confluence-facing consumer without erasing the candidate, window, substitution, and locality rows. Since the carrier has no coordinate for global Church–Rosser, subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency, the frontier read is exactly the finite route stated above. □

Theorem 4737.79 (MetaCIC candidate normalization residual-join scope)**.** *An accepted* MetacicCandidateNormaliz *candidate-normalization handoff may expose residual joins only after it has read the candidate diamond budget, the cut-height budget, the frontier dependency route, and the residual diamond bridge route. The resulting scope is the finite route through MetaCIC candidate rows, residual windows, candidate joins,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*; it is not an unscoped global confluence claim.*

Proof. Project the parent carrier of Definition 5547.2. Its visible rows already separate candidate evidence, bounded endpoint data, residual confluence routing, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

Read the budget rows next. The candidate diamond route of Definition 5547.44 fixes the bounded peak and join-budget surface, while the cut-height route of Definition 5547.46 fixes the height boundary before a residual endpoint can be consumed.

Finally read the frontier and residual-diamond routes. The residual-frontier dependency and the bridge route of chapter 4238 keep residual windows and candidate joins attached to the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Since no carrier coordinate stores global confluence, quotient syntax equality, evaluator equality, or kernel consistency, the handoff exposes only the scoped residual-join route stated above. □

Theorem 4737.80 (MetaCIC residual normal-form endpoint budget)**.** *Every accepted* MetacicCandidateNormalizat *normal-form endpoint read that is used as residual-join evidence must retain the candidate row, bounded endpoint row, candidate diamond budget, cut-height budget, residual diamond bridge route, and parallel-diamond frontier route before local confluence evidence is consumed. The retained surface is a finite route through the candidate-normalization carrier and its sibling frontier packets; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Project the residual-join scope of Definition 5547.79. A residual-join consumer enters through the candidate-normalization carrier, so the candidate row and bounded endpoint row remain visible before the endpoint can count as evidence.

Read the budget rows next. The diamond-budget route of Definition 5547.44 supplies the candidate diamond budget, and the cut-height route of Definition 5547.46 supplies the bounded endpoint height discipline.

Now attach the sibling frontier routes. The residual diamond bridge route keeps the residual bridge row in scope, while Definition 5547.73 keeps the parallel-diamond frontier route visible before a local confluence consumer can read the endpoint as join evidence.

Since the named rows are all finite carrier or sibling-route data, consuming local confluence evidence cannot erase them. The handoff has no coordinate for global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency, so the endpoint remains a guarded residual normal-form budget rather than an unscoped confluence theorem. □

Theorem 4737.81 (MetaCIC candidate normalization residual-join locality scope)**.** *Any* MetacicCandidateNormali *confluence handoff that exports residual-join data must first display the* MetaCICCriticalPath$^{\uparrow}$ *source*

row, the residual-join locality row, and the MetaCICParallelDiamondFrontier$^{\uparrow}$ *boundary row. A confluence consumer may read the handoff only after these rows have been kept in the finite* BHist, hsame, Cont, Pkg, *and* NameCert *route; it cannot treat candidate-normalization evidence as global Church–Rosser, unrestricted residual-substitution compatibility, subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Start from the handoff carrier of Definition 5547.2. Its candidate evidence, bounded endpoint, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming rows remain visible before a confluence-facing consumer reaches any endpoint.

Read the local sibling routes next. The critical-path and frontier route of Definition 5547.48 exposes MetaCICCriticalPath$^{\uparrow}$, while Definition 5547.79 keeps residual joins scoped by candidate rows, residual windows, transport, replay, provenance, and local naming.

Finally use Definition 5547.73 together with chapter 4238. The parallel-diamond frontier supplies the boundary row that a confluence consumer must see before reading the handoff, so the exported route remains a finite residual-join locality handoff rather than any of the excluded global MetaCIC claims. □

Theorem 4737.82 (MetaCIC candidate normalization residual-matrix forward route)**.** *Every* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *handoff consumed by chapter 8134 must first display candidate evidence, the normalization frontier, the residual envelope, the decidable boundary, the blocked edge, componentwise transport, displayed replay, provenance, and local naming rows. The residual-join matrix therefore reads the handoff only after the finite route through the carrier of Definition 5547.2; it does not receive global Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or kernel consistency.*

Proof. Project the candidate-normalization handoff carrier. The candidate evidence row, normalization-frontier row, residual-envelope row, decidable-boundary row, and blocked-edge row are visible before any structural row is replayed.

Next route the matrix consumer through the existing residual scope. The residual-join scope and residual budget exactness results keep the candidate, frontier, residual, boundary, and blocked-edge rows attached to the same finite packet.

Finally read transport, replay, provenance, and local naming. These rows preserve the displayed handoff into chapter 8134, so the matrix consumer receives a guarded child dependency route rather than an unscoped confluence theorem. □

Theorem 4737.83 (MetaCIC candidate normalization confluence handoff route factorization)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet consumed by the* MetaCICParallelDiamondF *route or by the normalization-confluence bridge must first expose the residual-budget factorization row of Definition 5547.92. The factorization keeps the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name visible before the handoff reaches chapter 4238 or the residual-substitution boundary of chapter 2181. It does not claim full confluence, subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Project the local residual-budget exactness theorem. It displays candidate evidence, the normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local naming before any residual budget is read.

Compose that projection with the parallel-diamond frontier route. The frontier consumer may read the handoff only after the residual-budget factorization row has fixed the bounded endpoint and kept the residual-substitution premise visible.

The same displayed row is the local premise for the normalization-confluence bridge. Since the packet has no coordinate for full confluence, subject reduction, evaluator equality, quotient syntax equality, or closed consistency, route composition exports only the residual-budget factorization boundary. □

Theorem 4737.84 (MetaCIC candidate normalization confluence handoff critical-path consumption)**.** *Any* MetaCICCriticalPath$^{\uparrow}$ *or* MetaCICParallelDiamondFrontier$^{\uparrow}$ *consumer of an accepted* MetacicCandidateNor *packet must first read the residual-budget factorization row of Definition 5547.83. The consumed route keeps candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name visible before the handoff is used by the critical path or the parallel-diamond frontier. It does not discharge full confluence, subject reduction, evaluator equality, quotient syntax equality, closed consistency, or residual-substitution.*

Proof. Start with the route factorization theorem. It forces the handoff to expose the residual-budget row before any downstream consumer can use the normalization-confluence boundary.

For a MetaCICCriticalPath$^{\uparrow}$ consumer, compose that factorization with the critical-path route. The critical path may retain the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name, but it cannot skip the residual-budget boundary.

For a MetaCICParallelDiamondFrontier$^{\uparrow}$ consumer, compose the same factorization with the parallel-diamond frontier route. The frontier read receives the handoff only after the residual-budget row has fixed the bounded endpoint and kept the residual-substitution premise visible.

The packet contains no carrier row for full confluence, subject reduction, evaluator equality, quotient syntax equality, closed consistency, or residual-substitution discharge. Therefore critical-path and frontier consumption preserve exactly the retained-premise boundary supplied by the handoff. □

Theorem 4737.85 (MetaCIC candidate handoff seed obligations)**.** *Every seed-obligation read of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet exposes candidate evidence K, normalization frontier F, residual envelope C, decidable boundary D, blocked edge B, componentwise transport T, displayed replay R, provenance P, and local* NameCert *row L before bounded endpoint consumption. The read remains a finite handoff boundary and does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with the handoff boundary of Definition 5547.3. It exposes candidate evidence and frontier data before endpoint consumption, keeps the residual and blocked-edge rows visible, and routes bounded comparison through the decidable-boundary row.

Use seed closure and premise exposure from Definition 5547.4 and Definition 5547.5. These theorems force K, F, C, D, B to remain displayed before a confluence-facing or decidability-facing consumer reaches N.

Finally apply decider-row exhaustion from Definition 5547.22. The structural rows T, R, P, L preserve the same finite packet, so the bounded endpoint is consumed only after the seed-obligation rows have been exposed. □

Theorem 4737.86 (MetaCIC candidate handoff induction)**.** *Induction over the displayed row order*

$$K, F, C, D, B, T, R, P, L$$

preserves candidate evidence, residual-envelope visibility, and decidable-boundary exposure before any endpoint row N is consumed. Hence the handoff remains finite-row evidence for the accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet and exports no unconditional strong normalization, Church–Rosser theorem, subject reduction theorem, quotient syntax equality, evaluator soundness theorem, or closed consistency principle.*

Proof. Project the accepted carrier through the handoff boundary of Definition 5547.3. The row order starts with candidate evidence K and normalization frontier F, so the induction base already exposes the candidate surface before any residual or endpoint-facing read.

For the residual step, use premise exposure from Definition 5547.5. Adding C, D, B keeps the residual envelope, decidable boundary, and blocked edge as displayed rows rather than hidden assumptions, and decider-row exhaustion from Definition 5547.22 keeps D finite before endpoint consumption.

For the structural step, apply the seed obligations of Definition 5547.85. The rows T, R, P, L only transport, replay, package, and locally name the same displayed packet, so they preserve the candidate evidence and the residual-envelope boundary already exposed by the earlier rows.

Thus every prefix of $K, F, C, D, B, T, R, P, L$ keeps the relevant evidence visible before N. Since the endpoint row is not consumed during the induction, none of the excluded global metatheorems or host evaluator conclusions is exported. □

Theorem 4737.87 (MetaCIC candidate residual handoff scope)**.** *Every accepted* MetacicCandidateNormalizationCo *packet routes residual-envelope data C and blocked-edge data B through candidate evidence K, normalization frontier F, decidable boundary D, componentwise transport T, displayed replay R, provenance P, and local* NameCert *row L before any bounded endpoint consumption. The route remains a residual handoff scope and does not assert full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Begin with the seed obligations of Definition 5547.85. They expose $K, F, C, D, B, T, R, P, L$ before endpoint consumption, so residual-envelope and blocked-edge data are visible rows rather than hidden premises.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. These force the candidate evidence and normalization frontier to precede every confluence-facing or decidability-facing handoff read.

Finally use decider-row exhaustion from Definition 5547.22. The decidable boundary is consumed as a finite row, while T, R, P, L only transport, replay, package, and locally name the same packet. Hence no endpoint consumer can replace the residual handoff by any excluded global metatheorem or host-evaluator claim. □

Theorem 4737.88 (MetaCIC candidate handoff residual frontier scope)**.** *Every accepted* MetacicCandidateNormaliz *packet routes residual-envelope data C and blocked-edge data B through the normalization frontier F and decidable boundary D before any endpoint-facing consumer reads the packet. Hence the handoff is a scoped frontier certificate: it cannot be read as unconditional strong normalization, unrestricted confluence, global conversion decidability, quotient syntax equality, or host evaluator soundness.*

Proof. Start from the seed-obligation exposure of Definition 5547.85. The accepted packet displays K, F, C, D, B and the structural rows before bounded endpoint consumption, so the residual envelope and blocked edge are not hidden premises.

Apply the boundary theorem Definition 5547.3 together with premise exposure from Definition 5547.5. These force candidate evidence to pass through the normalization frontier before a confluence-facing read can consume the endpoint.

Finally use decider-row exhaustion from Definition 5547.22. Since the decidable boundary row is exhausted before endpoint consumption, no consumer can replace the displayed residual frontier by global conversion decidability, quotient syntax equality, unrestricted confluence, unconditional strong normalization, or host evaluator soundness. □

Theorem 4737.89 (MetaCIC candidate normalization sibling confluence route)**.** *Every confluence-facing read of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first expose the sibling route through chapter 765, chapter 892, chapter 735, chapter 5970, and chapter 5126. The residual local-join determinacy packet supplies the finite residual-join ledger consumed before the candidate endpoint can be treated as confluence-facing evidence. The handoff may then export candidate-mediated endpoint, residual-envelope, decidable-boundary, and blocked-edge rows to a confluence consumer, but it cannot present a standalone confluence theorem before those sibling chapters have been displayed.*

Proof. Project the handoff carrier from Definition 5547.2. Its public confluence route stores audit, candidate, frontier, confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows rather than a detached confluence theorem.

Apply the boundary and consumer-boundary theorems of this chapter. They keep the candidate evidence, normalization frontier, residual envelope, decidable boundary, and blocked edge visible before any endpoint or join row is consumed.

The sibling chapters named in the statement supply the critical-path, normalization-frontier, confluence-audit, residual local-join, and residual-diamond obstruction surfaces for those rows. The local-join determinacy route contributes the finite join ledger that the candidate endpoint must consume before it is exported as confluence-facing evidence. Residual obstruction rows therefore route through the residual-join and parallel-diamond siblings rather than through a standalone confluence theorem. Since no carrier coordinate bypasses them, every confluence-facing handoff factors through the displayed sibling route. □

Theorem 4737.90 (MetaCIC candidate normalization local-confluence route)**.** *Any local-confluence consumer of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read the candidate-normalization carrier rows, the candidate diamond-budget dependency of chapter 6552, the residual bridge route inside the handoff packet, and then the local-confluence frontier of chapter 7346. The resulting route is a finite confluence-facing dependency chain; it is not a standalone local-confluence theorem, unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. A local-confluence read starts from the same audit anchor, candidate evidence, bounded endpoint, normalization frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming rows as every accepted handoff read.

Apply the candidate diamond-budget dependency route. It forces the consumer to keep the bounded local peak and join-budget rows visible before any endpoint row is treated as confluence-facing evidence.

Next read the residual bridge route inside this hub. The residual envelope, decidable-boundary row, and blocked-edge ledger remain displayed, so the handoff cannot erase residual-substitution or critical-pair obligations on the way to a local-confluence consumer.

Finally pass to the local-confluence frontier of chapter 7346. That sibling may consume the exposed candidate and residual rows as a frontier premise, but it receives no coordinate for global confluence, subject reduction, quotient syntax equality, evaluator equality, conversion decidability, or kernel consistency. □

Theorem 4737.91 (MetaCIC handoff local confluence certificate route)**.** *Every* MetaCICCandidateNormalizationCo *read exported to a normalization or confluence consumer must pass through the local-confluence certificate sibling chapter 7346, the parallel-diamond frontier sibling chapter 4238, the residual-join route chapter 7378, and the candidate-normalization budget rows of chapter 6552 before the handoff is public. The route is therefore*

$$\mathsf{MetaCICCandidateNormalizationConfluenceHandoff}^{\uparrow} \longmapsto (\mathsf{local\ confluence},\ \mathsf{MetaCICParallelDiamondFrontier}^{\uparrow},\ \mathsf{residu}$$

No consumer may read the handoff as a direct ambient confluence theorem, a hidden critical-pair normalizer, or a residual-join certificate with the local confluence and budget rows suppressed.

Proof. Read the local confluence sibling chapter 7346 first. It provides the certificate row that turns candidate overlap data into a local joinability check, so a confluence-facing consumer cannot start from the handoff alone.

Next read the parallel-diamond frontier sibling chapter 4238 and the residual-join route chapter 7378. These siblings keep the frontier and residual rows visible before a join is exported, and they prevent the candidate packet from becoming an unscoped global diamond assertion.

Finally read the candidate-normalization budget rows of chapter 6552. They bound the normalization side of the handoff before the packet is exported, so the public route contains the local confluence certificate, frontier, residual join, budget, replay, transport, provenance, and local naming surfaces together. □

Theorem 4737.92 (MetaCIC candidate normalization confluence handoff residual budget exactness)**.** *The* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet exposes residual-budget factorization only after candidate evidence K, normal endpoint N, frontier row F, residual confluence row C, decidable boundary D, blocked edge B, transport T, replay R, provenance P, and local name L have all been displayed. This finite route keeps the bounded residual budget tied to the carrier rows of the handoff packet and to the sibling confluence surfaces of chapter 765, chapter 4238, and chapter 5126. The row asserts the local residual-budget exactness needed by the handoff; it does not claim full Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or closed consistency.*

Proof. Project the candidate-normalization handoff packet. The finite route already displays candidate evidence, the bounded endpoint, the frontier row, the residual confluence row, the decidable boundary, the blocked edge, the transport row, the replay row, the provenance row, and the local name row before any residual budget is read.

Apply the sibling surfaces named in the statement. The critical-path row supplies the candidate anchor, the parallel-diamond frontier supplies the residual confluence boundary, and the residual-diamond sibling keeps the bounded budget local. Since the packet has no coordinate for full Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or closed consistency, the theorem records only the exact residual budget surface. □

Theorem 4737.93 (MetaCIC candidate normalization residual budget factorization)**.** *Any accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *residual budget factors through candidate evidence K, normal endpoint N, frontier row F, residual confluence row C, decidable boundary*

D, blocked edge B, transport T, replay R, provenance P, and local name L before any downstream residual join consumes it. The factorization routes through chapter 765, chapter 4238, and chapter 5126; it exports no residual join that bypasses the candidate evidence, normal endpoint, frontier row, decidable boundary, blocked-edge ledger, structural transport, replay, provenance, or local naming.

Proof. Project the displayed handoff packet. Its residual-budget row can be read only after the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name have all been exposed.

Use Definition 5547.92 for the exactness surface. The same displayed rows already bind the budget to the carrier rather than to a detached residual-join theorem.

Finally read the sibling routes. $\mathsf{MetaCICCriticalPath}^{\uparrow}$ supplies the candidate-critical anchor, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ supplies the frontier boundary, and $\mathsf{ParallelDiamondResidual}^{\uparrow}$ supplies the bounded residual-confluence row. Since the packet contains no coordinate outside those rows, every downstream residual join consumes the stated factorization first. □

Theorem 4737.94 (MetaCIC candidate normalization confluence handoff status)**.** *For an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet whose displayed rows include a candidate-normalization row K, a residual-join row C, a finite cut-height budget B, and a dependency row F for the parallel-diamond frontier, every confluence handoff factors through the residual-budget window before it can be consumed by the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *route. The available handoff surface is exactly the visible route through candidate evidence, bounded normal endpoint, residual join, cut-height budget, blocked-edge ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

This status row may cite the candidate-normalization handoff route of chapter 5547, the residual join scope of chapter 5126, the parallel-diamond dependency route of chapter 4238, and the local confluence certificate route of chapter 765. It exports no evidence-free normalization theorem, abstract source-equivalence theorem, classifier-transport theorem, full Church–Rosser theorem, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency claim.

Proof. Project the accepted handoff packet to its displayed rows. The candidate-normalization row K, residual-join row C, finite cut-height budget B, and frontier dependency F are all visible before the confluence-facing consumer can read the bounded endpoint or any residual join.

Apply the residual-budget exactness theorem and the local factorization theorem above. They force the confluence handoff through the same residual-budget window and keep the blocked-edge ledger, transport, replay, provenance, and local name rows attached to the read.

Now route the result to the sibling frontier. The parallel-diamond frontier consumes the handoff only after the candidate-normalization route, residual join scope, and local confluence certificate have displayed the same finite rows. Since the carrier has no coordinate for evidence-free normalization, abstract source equivalence, classifier transport, or global confluence, the handoff cannot bypass the residual-budget window. □

Theorem 4737.95 (MetaCIC handoff parallel-diamond sibling dependency)**.** *Every accepted* $\mathsf{MetacicCandidateNor}$ *carrier that exports candidate-normalization confluence data must consume the finite parallel-diamond frontier rows of chapter 4238 together with the candidate-normalization router surface of chapter 5210. The handoff therefore reads residual-premise retention, local critical-pair routing, candidate evidence, bounded endpoint routing, obstruction retention, transport, replay, provenance, and local naming before any confluence-facing consumer can use it.*

Proof. Project the handoff carrier through Definition 5547.2. Its candidate row, frontier row, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local name remain visible before the bounded endpoint row is exported.

Read the parallel-diamond frontier next. The frontier of chapter 4238 supplies the retained residual premise, local critical-pair, local-join, obstruction, and structural rows that any confluence-facing use must preserve.

Finally route candidate evidence through chapter 5210. The router keeps the candidate evidence and normalization readback inside a bounded certificate route, so the handoff cannot be read as premise-free confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency. □

Chapter 4738

A Concrete Naming Certificate for CantorIntersectionModulus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CantorIntersectionModulus$^{\uparrow}$ packet records Cantor intersection as explicit finite modulus data rather than as ambient compactness. It sits after the located nested-interval surface of chapter 1713, the bisection route of chapter 1711, the dyadic arithmetic core of chapter 291, the finite request interface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the finite prefix bar of chapter 5863. The chapter names the finite data needed to turn a retained nested branch into a regular Cauchy real name; it does not invoke open-cover compactness or a selected infinite branch.

Definition 4738.1 (Cantor intersection modulus carrier)**.** *A* CantorIntersectionModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *L is the located closed interval row, recording the endpoint order, locatedness tests, and retained closed-interval inclusion data.*
2. *B is the retained nested bisection branch ledger, exposing only the finite branch decisions already displayed by* LocatedIntervalBisection$^{\uparrow}$ *and* LocatedRealNestedIntervals$^{\uparrow}$*.*
3. *D is the dyadic diameter schedule from* DyadicRatCore$^{\uparrow}$*, recording the depth-to-radius budget and the finite halving estimates used at each retained stage.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row, listing the precision requests and interval stages inspected by the downstream real-name consumer.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected finite windows and their dyadic endpoint approximants.*
6. *E is the terminal* Real$^{\uparrow}$ *seal reached only after the interval, bisection, dyadic schedule, stream window, and regular rational readback rows have been displayed.*
7. *F is the finite fan-depth and modulus output row. It records the finite prefix depth at which the diameter schedule answers the requested precision and the bar-window handoff used by* FiniteFanTheorem$^{\uparrow}$*.*
8. *H records componentwise* hsame *transport for the interval, branch, diameter, window, readback, real seal, fan-depth, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an interval request through L, B, D, S, R, E, F, H and into the local certificate.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionModulus}^{\uparrow}$, $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite branch, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionModulusUp.CantorIntersectionModulusTasteGate single carrier alignment`

Theorem 4738.2 (Cantor intersection modulus NameCert obligations)**.** *For every accepted* CantorIntersectionModu *carrier*

$$\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located closed interval row, retained nested bisection branch ledger, dyadic diameter schedule,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, finite fan-depth and modulus output, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read of the packet must expose L, B, D before the stream window S, must expose S, R before the real seal E, and must expose the fan-depth row F before exporting a precision modulus.*

Thus a downstream completion or real-name consumer receives a finite modulus route from nested interval data into regular Cauchy real names. The packet exports no open-cover compactness theorem, selected infinite branch, countable-choice principle, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5548.1. The accepted packet displays exactly the located closed interval row, retained branch ledger, dyadic diameter schedule, finite stream windows, regular rational readback, terminal real seal, finite fan-depth output, transport, replay, provenance, and local naming rows.

The route order is forced by the displayed carrier. The nested interval consumer first reads L and B as finite interval and branch data, then reads D as the dyadic diameter schedule that bounds the retained interval at each displayed depth.

The real-name handoff is finite. The row S lists the precision windows, R reads back the corresponding regular rational approximants, and E seals the resulting $\mathsf{Real}^{\uparrow}$ row only after those windows and readbacks are already present.

Finally the row F records the finite fan-depth and modulus output used by the handoff to chapter 5863. Since the carrier contains no coordinate for open covers, selected infinite branches, countable choice, quotient real equality, or ambient completeness, H, C, P, N can transport, replay, source, and name only the displayed finite route. □

Theorem 4738.3 (Finite fan interval handoff)**.** *Every* $\mathsf{CantorIntersectionModulus}^{\uparrow}$ *consumer that asks for a finite precision window factors through the interval-to-fan route*

$$L, B, D \longrightarrow S, R, E \longrightarrow F, H, C, P, N.$$

The finite fan-depth row F may use the prefix-bar discipline of chapter 5863 only after the located interval, retained bisection branch, and dyadic diameter rows have fixed the inspected depth. Ordinary regular real readback still passes through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows; the fan row supplies only the finite window bound.*

Proof. Start with an accepted carrier $\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N)$. The first three rows determine the displayed nested interval data: L gives the located closed interval, B gives the retained finite bisection branch, and D gives the dyadic diameter schedule.

The stream and real-name rows then read only the windows named by that displayed interval data. The rows S and R expose the requested finite precision windows and regular rational readback, and E records the terminal real seal reached from those rows.

It remains to identify the fan use. The row F is the only carrier coordinate that records a finite fan-depth or modulus output, so any consumer asking for a prefix-bar bound must pass through F after L, B, D, S, R, E. The transport, replay, provenance, and local naming rows H, C, P, N then export the same finite route and cannot introduce open-cover compactness, a selected infinite branch, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{CantorIntersectionModulus}^{\uparrow}$ precision-window read factors through one located closed interval row, one retained nested bisection branch ledger, one $\mathsf{DyadicRatCore}^{\uparrow}$ diameter schedule, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, one finite $\mathsf{FiniteFanTheorem}^{\uparrow}$ depth row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real-nested-intervals, located-interval-bisection, dyadicratcore, streamname, regseqrat, real, finite-fan-theorem: $\mathsf{CantorIntersectionModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries nested interval data, dyadic diameter scheduling, real-name readback, and finite fan-depth output as one modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5548.1, Definition 5548.2, and Definition 5548.3 seed the finite Cantor-intersection modulus route through located interval rows, retained bisection ledgers, dyadic diameter schedules, finite stream windows, regular rational readbacks, terminal real seals, and finite fan-depth rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CantorIntersectionModulusUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-depth exactness, interval-window coverage, componentwise transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CantorIntersectionModulusUp.

Constructive story: A $\mathsf{CantorIntersectionModulus}^{\uparrow}$ packet is generated from located closed interval rows, retained nested bisection branch ledgers, dyadic diameter schedules, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, terminal $\mathsf{Real}^{\uparrow}$ seals, finite $\mathsf{FiniteFanTheorem}^{\uparrow}$ depth outputs, componentwise transport, displayed replay, provenance, and a local NameCert row.

Chapter 4739

A Concrete Naming Certificate for $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet records the finite NameCert ledger that assembles compact finite-net data, pointwise continuity rows, metric distance observations, rational lower-bound folds, and the final uniform-modulus output. It sits over the compact carrier of chapter 60, the compact metric source of chapter 129, the continuous and continuous-map rows of chapter 59 and chapter 130, the metric distance surface of chapter 57, the rational ledger of chapter 37, the total-order selector route of chapter 54, the finite-net fold of chapter 619, and the $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names the full compactness-implies-uniform-continuity ledger as finite BEDC data; it is not an open-cover theorem, a host minimum, a selected subcover, or an ambient choice principle.

Definition 4739.1 (Compact uniform continuity ledger carrier)**.** *A* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, M, R, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, including the compact source, the finite* $\mathsf{ProbeBundle}$ *of centers, and the located coverage witnesses read from the compact metric packet.*
2. *F is the pointwise modulus row for the continuous map, carrying the graph reads, centerwise source radii, and target-tolerance certificates over the same compact source.*
3. *M is the metric distance row, recording the source distances from an input point to a displayed center, from a second input point to that center, and the target distances used by the triangle estimate.*
4. *R is the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, read through the* $\mathsf{TotalOrder}^{\uparrow}$ *selector route over the bundled positive source radii.*
5. *T is the finite triangle and tolerance ledger tying the lower-bound fold to the source and target metric comparisons.*
6. *U is the output* $\mathsf{UniformModulus}^{\uparrow}$ *row, exposing the tolerance-only precision history obtained from K, F, M, R, T.*

7. *H records componentwise* hsame *transport for compact finite nets, pointwise modulus rows, metric distance observations, rational folds, triangle ledgers, output rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a compact-uniform-continuity tolerance request through $K, F, M, R, T, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through these coordinates. It contains no coordinate for an arbitrary open cover, selected subcover, host minimum over points, quotient function equality, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityLedgerUp.CompactUniformContinuityLedgerTasteGate`
`single carrier alignment`

Theorem 4739.2 (Compact uniform continuity ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *carrier $L = (K, F, M, R, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinui}}$ *surface consists exactly of carrier admission for the compact finite-net row, pointwise modulus row, metric distance row, finite rational lower-bound fold, triangle and tolerance ledger, output uniform-modulus row, componentwise transport, displayed continuation replay, package provenance, and local naming; source compatibility between K and F; metric-read compatibility between M and the compact source; fold exactness through R and the* $\mathsf{TotalOrder}^{\uparrow}$ *selector route; triangle exactness through T; uniform-modulus handoff through U; and nonescape from arbitrary open covers, selected subcovers, host minima, quotient function equality, countable choice, and ambient real completeness.*

Proof. Project the accepted carrier of Definition 5549.1. The visible rows are exactly the compact finite-net data, pointwise modulus data, metric distance observations, rational fold, triangle ledger, output row, and structural rows.

The source lock is read before any modulus row is consumed. The compact row K and pointwise row F name the same compact metric source, so the finite centers and local radii cannot move to another carrier.

The rational fold is the row R. It is computed over the displayed bundled radii through the $\mathsf{TotalOrder}^{\uparrow}$ selector route, and the metric row M supplies only the distances later used by the source and target triangle estimates.

The triangle and tolerance row T binds these reads into the output row U. Componentwise transport, continuation replay, package provenance, and local naming then preserve the same ledger, so no excluded open-cover, choice, quotient, host-minimum, or completeness coordinate is available to a consumer. □

Theorem 4739.3 (Compact uniform continuity ledger finite-fold route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{TotalOrder}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

More explicitly, the consumer must first read compact finite-net centers and pointwise modulus rows, then the metric distance observations, then the finite $\mathsf{Rat}^{\uparrow}$ *lower-bound fold selected by the*

$\mathsf{TotalOrder}^{\uparrow}$ *route, then the triangle and tolerance ledger, and only then the output uniform-modulus row. No compact-uniform-continuity read can bypass this finite fold by using an arbitrary open cover, selected subcover, host minimum, or choice schedule.*

Proof. Start with the compact finite-net row K and the pointwise modulus row F. They expose the bundled centers, local radii, graph reads, and source compatibility required by the compact-uniform-continuity request.

Next read the metric row M and the rational fold row R. The metric row supplies the source and target distances used by the later estimate, while R folds the displayed positive radius family through the $\mathsf{TotalOrder}^{\uparrow}$ selector route. The selected precision is therefore a finite history, not an external minimum.

Finally read T and U. The triangle and tolerance ledger transports the selected lower bound into the two source estimates and the target triangle estimate, and the $\mathsf{UniformModulus}^{\uparrow}$ row receives only that tolerance-only precision history.

The rows H, C, P, N keep the route stable under transport, replay, provenance, and naming. Since the carrier has no open-cover, selected-subcover, host-minimum, or choice coordinate, every compact-uniform-continuity consumer factors through the displayed finite fold. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ compact-uniform-continuity read must expose a compact finite-net row, pointwise continuous-map modulus rows, metric distance observations, a finite $\mathsf{Rat}^{\uparrow}$ lower-bound fold, a $\mathsf{TotalOrder}^{\uparrow}$ selector route, a triangle and tolerance ledger, a terminal $\mathsf{UniformModulus}^{\uparrow}$ row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact, compactmetric, continuous, continuousmap, metric, rat, totalorder, finitenetminimumfold, uniformmodulus: $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ is not bijective to any listed sibling because it binds the whole compactness-implies-uniform-continuity route from finite compact coverage through pointwise radii, metric observations, rational folding, triangle estimates, and the terminal UniformModulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5549.1, Definition 5549.2, and Definition 5549.3 seed the finite NameCert ledger for compactness-implies-uniform-continuity over compact finite nets, pointwise modulus rows, metric distance rows, RatUp folds, TotalOrder selector routes, triangle ledgers, UniformModulus handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host minima, quotient function equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, metric-read compatibility, rational fold exactness, triangle exactness, uniform-modulus handoff, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.CompactUniformContinuityLedgerUp.

Constructive story: A $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMap pointwise modulus rows, Metric distance observations, RatUp lower-bound folds, TotalOrder selector routes, triangle and tolerance ledgers, UniformModulus output rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4740

A Concrete Naming Certificate for $\mathsf{FrechetMetricEmbedding}^{\uparrow}$

Following Fréchet and Kuratowski, the $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ packet records a metric space by the finite distance observations that send a point to its displayed distances from a finite probe family. It sits over the metric source of chapter 57, the real-distance rows of chapter 38, the distance-function route of chapter 7569, and the embedding interfaces of chapter 439 and chapter 1652. The packet names only displayed finite reads; it does not import host function extensionality, quotient equality of maps, or an ambient function-space carrier.

Definition 4740.1 (Fréchet metric embedding carrier)**.** *A* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the point row whose accepted element is the source point being embedded.*
2. *Q is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source classifier and source-distance interface.*
3. *D is the family of* $\mathsf{Real}^{\uparrow}$*-valued distance-read rows $d(X, q)$ for displayed probes q.*
4. *A is the finite probe-family and classifier row: two packets agree exactly when the displayed* $\mathsf{Real}^{\uparrow}$ *distance reads agree componentwise on the same probe family.*
5. *H records componentwise* hsame *transport for the point row, metric-source row, distance-read rows, probe-family classifier, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from source metric reads through the probe-family distance reads and then through the classifier-by-displayed-distance-agreement.*
7. *P is the* Pkg *provenance over* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{KuratowskiEmbedding}^{\uparrow}$*,* $\mathsf{IsometricEmbedding}^{\uparrow}$*,* $\mathsf{UniformEmbedding}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host function object, quotient equality of distance functions, selected completion representatives, or an ambient real-valued function space.

Lean4 ✓ `BEDC.Derived.FrechetMetricEmbeddingUp.FrechetMetricEmbeddingTasteGate single carrier alignment`

Theorem 4740.2 (Fréchet metric embedding NameCert obligations)**.** *For every accepted* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ *carrier* $F = (X, Q, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$ *surface consists exactly of the finite rows displayed in Definition 5550.1. Carrier admission reads the point row, metric-source row,* $\mathsf{Real}^{\uparrow}$ *distance-read rows, finite probe family, classifier-by-displayed-distance-agreement,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming row. Classifier stability compares accepted packets only by the displayed* $\mathsf{Real}^{\uparrow}$ *distance reads over the shared finite probe family, with transport and replay preserving that same route.*

Proof. Project the accepted carrier $F = (X, Q, D, A, H, C, P, N)$. The point row X and metric-source row Q are the only source data exposed before distance reads are requested.

The row D then records the finite $\mathsf{Real}^{\uparrow}$ distance observations against the probe family named by A. Since A classifies by displayed distance agreement, two accepted packets can be compared only by componentwise equality of these finite reads under the same probe family.

Finally read H, C, P, N. Transport, replay, provenance, and local naming preserve the point-to-distance route without adding host function extensionality, quotient equality of maps, selected representatives, or an ambient function-space carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 5550.1 and Definition 5550.2 seed the finite distance-function embedding route over MetricSpaceUp, RealUp, KuratowskiEmbeddingUp, IsometricEmbeddingUp, and UniformEmbeddingUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of maps, selected completion representatives, an ambient function-space carrier, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, displayed-distance classifier stability, hsame transport, Cont replay, provenance, and local NameCert coverage for BEDC.Derived.FrechetMetricEmbeddingUp.

Constructive story: A $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ packet is generated from a point row, a MetricSpaceUp source row, RealUp-valued distance-read rows, a finite probe-family classifier, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4741

A Concrete Naming Certificate for $\mathsf{CaristiFixedPoint}^{\uparrow}$

Following Caristi's fixed point theorem and the Ekeland complete-metric descent route, the $\mathsf{CaristiFixedPoint}^{\uparrow}$ packet records the finite surface by which a self-map satisfying a potential-decrease inequality reaches a seal-facing fixed endpoint. It sits beside the Banach fixed-point packet of chapter 396, the stability handoff of chapter 5856, and the variational descent ledger of chapter 3926. The packet names a complete-metric fixed-point certificate over one displayed potential and one displayed self-map; it is not a host minimizer theorem, a choice-selected descent sequence, quotient equality of points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4741.1 (Caristi fixed point carrier)**.** *A* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (X, d, \phi, T, E, B, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *domain row, naming the metric carrier and its complete-metric handoff surface.*
2. *d is the distance row used for all comparisons between a point and its image under T.*
3. *ϕ is the lower-semicontinuous potential row, read through* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *and a regular-rational value ledger.*
4. *T is the self-map row whose graph is inspected only through the displayed* Cont *route.*
5. *E is the Caristi descent inequality row, recording the finite comparison $d(x, Tx) \leq \phi(x) - \phi(Tx)$ for every admitted probe row.*
6. *B is the Ekeland handoff row, exposing the variational descent route that turns the potential inequality into a terminal fixedness boundary.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *endpoint seal, used only after the complete-metric descent row has been consumed.*
8. *H records componentwise* hsame *transport for the domain, distance, potential, self-map, descent, Ekeland handoff, endpoint seal, replay, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from any fixed-point read through $X, d, \phi, T, E, B, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CaristiFixedPoint}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, d, \phi, T, E, B, R, H, Q, P, N$. It has no coordinate for a host-selected minimizer, arbitrary choice sequence, quotient point equality, global contraction theorem, or ambient completeness beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Lean4 ✓ `BEDC.Derived.CaristiFixedPointUp.CaristiFixedPointTasteGate single carrier alignment`

Theorem 4741.2 (Caristi fixed point NameCert obligations)**.** *For every accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier*

$$C = (X, d, \phi, T, E, B, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete-metric row, distance row, lower-semicontinuous potential, self-map row, Caristi descent inequality, Ekeland handoff, endpoint seal, transport, replay, provenance, and local naming. The obligation surface also records the comparison boundary with* $\mathsf{BanachFixedPoint}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$*: Banach supplies contraction-orbit fixedness, while this packet supplies potential-descent fixedness.*

Proof. Project the carrier of Definition 5551.1. Its analytic rows are exactly X, d, ϕ, T, E, B, R, and its structural rows are exactly H, Q, P, N.

The descent inequality is local to the displayed self-map. The row E reads d, ϕ, and T together, so a consumer cannot detach the distance comparison from the lower-semicontinuous potential ledger.

The fixed-point endpoint is downstream from complete-metric descent. The row B sends the Caristi inequality into the Ekeland variational handoff, and R seals only the resulting regular-rational and real endpoint boundary.

Finally the Banach/Picard comparison is only a boundary row. The provenance records the neighbouring fixed-point route, but the carrier has no contraction constant or Picard-orbit coordinate from which a Banach proof could be read. Hence the displayed rows exhaust the NameCert obligations. □

Theorem 4741.3 (Caristi fixed point Ekeland handoff)**.** *Every accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *packet factors its descent read through the finite route*

$$\mathsf{CompleteMetric}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{EkelandVariationalPrinciple}^{\uparrow} \longmapsto E, B, R, H, Q, P, N.$$

The Ekeland handoff consumes the displayed potential-decrease inequality and returns only a certificate-local endpoint boundary. It does not produce a host minimizer, selected descent sequence, quotient equality of points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $C = (X, d, \phi, T, E, B, R, H, Q, P, N)$. The complete-metric row X and distance row d provide the only metric input available to the descent route.

The lower-semicontinuous row ϕ and inequality row E then form the Caristi descent datum. A consumer must read T through the same route, because E compares a point with its displayed image under T.

Apply the Ekeland handoff row B. It repacks the potential-decrease datum as the finite variational descent ledger named by $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, and it leaves the endpoint seal to R.

The transport, replay, provenance, and local name preserve this order. Since the carrier has no selector row for a host minimizer or arbitrary descent sequence, the handoff exports only the stated certificate-local endpoint boundary. □

Theorem 4741.4 (Caristi fixed point endpoint fixedness and non-escape)**.** *For an accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier, the endpoint row R is classifier-fixed under the displayed self-map row T after the Caristi descent inequality E and Ekeland handoff B have been consumed. Every endpoint read factors through*

$$X, d, \phi, T, E, B, R, H, Q, P, N,$$

and no consumer may read host fixed-point equality, selected limits, quotient point equality, global contraction soundness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Read the endpoint row through the carrier. The row R is introduced only after the complete-metric, potential, self-map, inequality, and Ekeland handoff rows have been displayed.

The fixedness assertion is classifier-internal. Applying T at the endpoint route is allowed only through Q, and the comparison back to R is transported by H inside the same local NameCert surface.

The non-escape boundary follows from the absence of additional coordinates. There is no row for a host equality of points, a selected limiting sequence, quotient equality, a contraction constant, or ambient real-completeness theorem. Thus every endpoint fixedness read remains inside $X, d, \phi, T, E, B, R, H, Q, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{CaristiFixedPoint}^{\uparrow}$ endpoint read factors through a CompleteMetricUp row, a lower-semicontinuous potential row, a displayed Caristi descent inequality, an EkelandVariationalPrincipleUp handoff, a RegSeqRatUp and RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: banachfixedpoint, banach-fixed-point-stability, ekeland-variational-principle, picard-iteration, complete-metric: $\mathsf{CaristiFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores complete-metric descent, lower-semicontinuous potential, self-map graph, Caristi inequality, Ekeland handoff, Banach/Picard comparison boundary, endpoint sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CaristiFixedPoint}^{\uparrow}$ packet is generated from a complete-metric row, distance row, lower-semicontinuous potential row, self-map row, Caristi descent inequality, Ekeland handoff, Banach/Picard comparison boundary, RegSeqRat and Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5551.1, Definition 5551.2, Definition 5551.3, and Definition 5551.4 seed the complete-metric potential-descent fixed-point packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, selected minimizers, selected descent sequences, quotient point equality, global contraction soundness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-semicontinuous potential readback, Caristi inequality exactness, Ekeland handoff, endpoint fixedness, Banach/Picard boundary confinement, transport, replay, provenance, and local naming for BEDC.Derived.CaristiFixedPointUp.

Chapter 4742

A Concrete Naming Certificate for $\mathsf{WeightedDeformationSpine}^{\uparrow}$

The $\mathsf{WeightedDeformationSpine}^{\uparrow}$ packet realizes the weighted deformation spine of subsection 9462.28.21 as finite trigger-weight deformation data. It sits beside the boundary-tile packet of chapter 3393, the reassignment-mass packet of chapter 5869, and the private/shared decoration packet of chapter 5896. The packet records trigger support, weights, deformation route, boundary-tile readback, transport, replay, provenance, and local naming together; it is not a biological mechanism, stochastic model, hidden graph search, quotient codon equality, or unrecorded trigger oracle.

Definition 4742.1 (Weighted deformation operator carrier)**.** *A* $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, L, D, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the trigger-support row, naming the finite codon-window vertices admitted by the packet.*
2. *L is the weight ledger, assigning each admitted trigger row its displayed reassignment mass.*
3. *D is the deformation-route row, recording the nested connected supports and their Hamming-adjacent spine.*
4. *B is the boundary-tile readback row, tying the Stop, AGR, AAA, and AUA modules back to the DNA-window six boundary-tile source.*
5. *H records componentwise* hsame *transport for S, L, D, B, C, P, N.*
6. *C records displayed* Cont *replay from any deformation-spine read through trigger support, weights, route, boundary readback, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{WeightedDeformationSpine}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{ReassignmentMassSubcu}$ $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$*, the weighted-spine theorem of Definition 9462.1260,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, L, D, B, H, C, P, N. It has no coordinate for a biological causal mechanism, stochastic evolution model, quotient codon semantics, hidden global enumeration, or unweighted trigger boundary.

Theorem 4742.2 (Weighted deformation spine NameCert obligations)**.** *For every accepted* WeightedDeformationSp *carrier*

$$W = (S, L, D, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ *surface consists exactly of carrier admission for trigger support, weight ledger, deformation route, boundary-tile readback, componentwise transport, displayed replay, provenance, and local naming. The weighted deformation route must preserve the nested connected supports and the spine path*

$$\mathrm{UAG} - \mathrm{UAA} - \mathrm{UGA} - \mathrm{AGA} - \mathrm{AGG}$$

with branch

$$\mathrm{AGA} - \mathrm{AAA} - \mathrm{AUA}.$$

Proof. Project the carrier of Definition 5552.1. The support, weight, route, boundary, transport, replay, provenance, and local name are exactly the displayed coordinates.

The route is weighted, not merely supported. The row L must be read before D, because the nested connected supports are selected by maximum-mass connected candidates from the source theorem.

The boundary readback is also explicit. The row B attaches the Stop, AGR, AAA, and AUA modules to the DNA-window six boundary-tile source, so a consumer cannot reinterpret the spine as an ungrounded graph path.

Finally H, C, P, N preserve and name the same rows. Since biological mechanism, stochastic dynamics, quotient codon equality, hidden enumeration, and unweighted trigger boundaries have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4742.3 (Weighted deformation spine trigger stability)**.** *For accepted* $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *carriers, componentwise* hsame *transport preserves the trigger support S, weight ledger L, deformation route D, and boundary-tile readback B as one packet. A deformation consumer may vary only through displayed transport and replay rows; it cannot keep the same route while dropping the weight ledger, or keep the same support while replacing the boundary readback by an unrecorded trigger classifier.*

Proof. Start with the accepted carrier $W = (S, L, D, B, H, C, P, N)$. The transport row H is componentwise across support, weights, route, boundary readback, replay, provenance, and local naming.

The weight ledger is tied to the route. If D is preserved, the maximum-mass connected-candidate interpretation still requires the weights stored in L; otherwise the deformation boundary is no longer the weighted spine named by the packet.

The boundary readback is tied to the trigger support. If S is preserved, the Stop, AGR, AAA, and AUA module readback must remain the displayed row B, not a hidden classifier outside the carrier.

Thus every stable trigger read preserves S, L, D, B, H, C, P, N together. The packet cannot be reduced to support alone, route alone, or weights alone. □

Theorem 4742.4 (Weighted deformation spine TasteGate surface)**.** *The future Lean TasteGate file for namespace* BEDC.Derived.WeightedDeformationSpineUp *must provide* Nontrivial $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *and* FieldFaithful $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *instances preserving the same finite carrier rows S, L, D, B, H, C, P, N. Nontriviality is witnessed by the distinction between support-only, weight-only, and route-only reads; field faithfulness preserves the joint trigger-support, weight-ledger, deformation-route, boundary-readback, transport, replay, provenance, and local naming surface.*

Proof. The carrier contains at least three independently readable analytic rows: support S, weights L, and route D. Collapsing the packet to one of them erases the other two and therefore cannot preserve the NameCert obligations.

Boundary readback supplies a further separate row. The row B connects the deformation spine to the DNA-window boundary-tile source, while H, C, P, N transport, replay, source, and name the packet.

A TasteGate consumer must preserve all of these rows as fields of one carrier. That is exactly the paper-side surface for the stated Nontrivial and FieldFaithful instances. □

Falsifiable prediction: Removing the weight ledger changes the accepted deformation boundary while leaving the trigger support readable.

Independence witness: observervtracekernel, finitewindowobserver, observerfilterbudgetselector, codon-window-sibling slugs: explain why none is bijective because only this packet stores trigger support, weights, deformation route, boundary-tile readback, transport, replay, provenance, and local naming together.

Closure status for $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{WeightedDeformationSpine}^{\uparrow}$ packet is generated from finite trigger-support rows, a weight ledger, a connected deformation route, boundary-tile readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5552.1, Definition 5552.2, Definition 5552.3, and Definition 5552.4 seed the finite trigger-weight deformation packet over the DNA-window spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unweighted trigger classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-support exactness, weight-ledger exactness, deformation-route stability, boundary-tile readback, transport, replay, provenance, local naming, Nontrivial WeightedDeformationSpineUp, and FieldFaithful WeightedDeformationSpineUp for BEDC.Derived.WeightedDeformationSpineUp.

$\mathsf{CompactRealInterval}^{\uparrow}$ routes closed interval compactness through chapter 38, chapter 291, chapter 301, and chapter 3471; the child files below keep the carrier, finite-net, diagonal-selection, and Real-interval routes separate.

Chapter 4743

A Concrete Naming Certificate for $\mathsf{CompactRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine-Borel finite-net reading of closed intervals, the $\mathsf{CompactRealInterval}^{\uparrow}$ packet records a compact closed interval as finite BEDC NameCert data. It sits over the real-name seal of chapter 38, the dyadic scale and tolerance rows of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the finite-net compactness siblings chapter 129 and chapter 3471. The packet is an interval carrier for later compact-real consumers; it does not import host compactness, selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4743.1 (Compact real interval carrier)**.** *A* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, O, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two located* $\mathsf{Real}^{\uparrow}$ *endpoint rows, each exposed through finite real-name windows before any interval request is consumed.*
2. *O is the order and apartness ledger, recording the displayed endpoint order, located overlap tests, boundary apartness rows, and refusal of trichotomy outside the inspected interval windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite tolerance ledger, recording common-scale comparisons, mesh radii, endpoint slack, and dyadic containment checks.*
4. *W is the finite dyadic net schedule, listing the mesh windows used to inspect the interval between the endpoint rows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the finite stream requests that read endpoint and mesh data without opening a quotient stream.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking endpoint, mesh, order, and tolerance checks as regular rational finite windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and inspected mesh values.*
8. *H records componentwise* hsame *transport for endpoints, order/apartness ledgers, dyadic tolerances, finite nets, stream windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-interval request through $A, B, O, D, W, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, O, D, W, S, R, E, H, C, P, N$. It has no coordinate for host compactness, arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness.

Theorem 4743.2 (Compact real interval finite-net obligations)**.** *For every accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier*

$$I = (A, B, O, D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *endpoint admission for the two located* $\mathsf{Real}^{\uparrow}$ *rows A and B;*
2. *order and apartness admission for the displayed ledger O;*
3. *dyadic scale and tolerance admission for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
4. *finite-net schedule admission for W, including mesh containment and endpoint slack;*
5. *StreamName finite-window admission for S and regular rational readback for R;*
6. *terminal Real seal admission for E after the finite endpoint, dyadic, stream, and regular rows are visible;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No compact-interval obligation is stored outside these rows.

Proof. Project the carrier of Definition 5553.1. The endpoint rows A, B and the order/apartness row O expose the interval boundary and the located comparison surface available to the packet.

Read the finite inspection rows next. The dyadic ledger D supplies common-scale tolerance checks, while W lists the finite mesh windows to which interval inspection is confined.

The stream and regular rows are downstream. The row S carries the finite StreamName requests, and R repacks the same endpoint, mesh, and tolerance reads as regular rational windows before the terminal Real seal E is consumed.

Finally, H, C, P, N transport, replay, source, and locally name the same finite packet. Since the carrier has no coordinate for host compactness, arbitrary open covers, selected subcovers, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 4743.3 (Compact real interval NameCert scope)**.** *Every accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{Real}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{HeineBorelFiniteNet}^{\uparrow}.$$

Thus the compact closed-interval surface exports only endpoint Real seals, dyadic tolerance and finite-net data, finite StreamName windows, regular rational readbacks, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not export host compactness, arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint and tolerance rows. The carrier exposes the interval through A, B, O, D, so every later compact-interval read is attached to the same located Real endpoints and dyadic tolerance ledger.

Use the finite-net and window rows. The row W records the displayed dyadic net schedule, S restricts stream requests to finite windows, and R records the regular rational readback of exactly those windows.

The compactness siblings are read only through the same finite rows. The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ provenance in P witnesses compatibility with finite-net compactness, but it does not add an arbitrary cover selector or host compactness coordinate.

The structural rows H, C, P, N preserve the route by transport, replay, sourcing, and local naming. Since no refused object appears in the carrier, every accepted consumer remains inside the stated scope. □

Theorem 4743.4 (Compact real interval effective completion route)**.** *Let $I = (A, B, O, D, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier. Any consumer that exports endpoint or mesh data to a completion-facing reader must pass through finite interval nets,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, and then the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *route of chapter 5930. The export does not assert open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5553.1. Endpoint and mesh data are displayed only by A, B, O, D, W, S, R, E before the structural rows H, C, P, N are consumed.

Apply Definition 5553.2. The finite-net obligation exposes endpoint admission, order and apartness, dyadic scale and tolerance, the finite net schedule, StreamName windows, regular rational readback, and the terminal Real seal in that order.

Apply Definition 5553.3. The compactness side is finite-net compactness through $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, not an arbitrary open-cover selector.

The completion-facing reader therefore receives the same finite data through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before entering chapter 5930. Componentwise hsame transport and displayed Cont replay preserve the route, so no quotient, choice, selected-subcover, or ambient-completeness coordinate is introduced. □

Falsifiable prediction: Every $\mathsf{CompactRealInterval}^{\uparrow}$ read factors through two located $\mathsf{Real}^{\uparrow}$ endpoint rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite dyadic net schedule, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, finite-net compactness sibling provenance, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ rows.

Independence witness: real, dyadicratcore, streamname, regseqrat, compactmetric, heineborelfinitemesh: $\mathsf{CompactRealInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint Real seals, order/apartness data, dyadic common-scale tolerances, a finite interval net schedule, finite stream windows, regular readback, finite-net compactness provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactRealInterval}^{\uparrow}$ packet is generated from two located RealUp endpoint rows, an order and apartness ledger, a DyadicRatCoreUp finite tolerance ledger, a finite dyadic net schedule, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5553.1, Definition 5553.2, and Definition 5553.3 seed the compact closed-interval carrier over endpoint Real seals, dyadic tolerances, finite nets, StreamName windows, RegSeqRat readback, and finite-net compactness sibling provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactRealInterval}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactness, arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealInterval}^\uparrow$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net schedule exactness, dyadic tolerance stability, StreamName window pullback, RegSeqRat readback, Real endpoint sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalUp.

Theorem 4743.5 (Compact real interval closed-bounded dependency route)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *consumer that inspects the compact-interval packet must first pass through the closed-bounded interval endpoint rows of chapter 1591, the dyadic finite-net rows recorded by the compact interval carrier, the* $\mathsf{StreamName}^\uparrow$ *finite windows of chapter 126, the* $\mathsf{RegSeqRat}^\uparrow$ *readback rows, and the terminal* $\mathsf{Real}^\uparrow$ *seal of chapter 38. The route exposes interval endpoints, finite net data, stream observations, regular rational reconstruction, and local real naming before any compact interval consumer may inspect the packet.*

Proof. Project the compact real interval carrier. The closed-bounded endpoint rows supply the only interval source, and the dyadic finite-net rows supply the only finite covering data inside the packet.

The stream and regular-readback rows are then read in order. The $\mathsf{StreamName}^\uparrow$ windows specify the finite observations, and the $\mathsf{RegSeqRat}^\uparrow$ rows reconstruct the rational readback attached to those observations.

The terminal $\mathsf{Real}^\uparrow$ seal is available only after these endpoint, finite-net, stream, and readback rows have been displayed. Thus a compact interval consumer receives the actual L10 dependency route rather than an ungrounded compactness credential. □

Theorem 4743.6 (Compact real interval Cauchy diagonal dependency)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *finite-net or nested-window consumer that passes to the* $\mathsf{CauchyDiagonalSelection}^\uparrow$ *sibling packet of chapter 4080 must first expose the interval endpoint rows, the located interval row, the compact finite-net row, the* $\mathsf{StreamName}^\uparrow$ *window row, the* $\mathsf{RegSeqRat}^\uparrow$ *handoff row, the terminal* $\mathsf{Real}^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^\uparrow}$ *row. The route may not import selected subsequences, open-cover compactness, quotient real equality, or ambient completeness.*

Proof. Project the compact real interval carrier of Definition 5553.1. The endpoint rows A, B, the order and located interval row O, the finite dyadic net schedule W, and the compact finite-net entries in P are visible before any diagonal read is available.

The stream and rational readback rows are then forced into the Cauchy-facing order. The $\mathsf{StreamName}^\uparrow$ row S supplies finite observation windows, the $\mathsf{RegSeqRat}^\uparrow$ row R supplies the rational handoff, and the $\mathsf{Real}^\uparrow$ row E supplies the terminal seal.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores a selected subsequence, arbitrary open cover, quotient real equality, or ambient completeness principle, every compact-interval handoff to $\mathsf{CauchyDiagonalSelection}^\uparrow$ is confined to the displayed rows. □

Theorem 4743.7 (Compact real interval dossier lattice route)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *dossier consumer that inspects an accepted interval packet must first display the* $\mathsf{ClosedBoundedInterval}^\uparrow$

endpoint rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net compatibility row, and the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *provenance row. Only after those rows have been replayed through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *row may the consumer inspect the compact interval packet itself.*

Proof. Project the carrier of Definition 5553.1. The endpoint rows A and B give the closed-bounded interval source, while the dyadic tolerance row D records the finite scale and containment ledger attached to those endpoints.

The stream and readback rows are then forced into the same route. The $\mathsf{StreamName}^{\uparrow}$ row S supplies only finite observation windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reconstructs the corresponding rational readback, and the $\mathsf{Real}^{\uparrow}$ seal row E is reached only after those finite reads are visible.

The compactness-facing rows are provenance and compatibility rows, not host compactness principles. The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ entries in P record finite-net compatibility and finite-net provenance before any downstream consumer can use the interval packet as compact data.

Finally H, C, P, N transport, replay, source, and name that same lattice route. Since no carrier coordinate stores arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness, a compact interval consumer can inspect only the displayed dossier lattice. □

Every $\mathsf{CompactRealInterval}^{\uparrow}$ finite-net consumer records its sibling route through the finite epsilon-net surface of chapter 1767, the located interval source of chapter 1709, the closed-bounded interval net packet of chapter 1250, and the terminal real-name seal of chapter 38 before any compact interval packet is read. The compact interval finite-net read also consumes the $\mathsf{CompactMetric}^{\uparrow}$ carrier of chapter 129 and the $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ finite-cover route of chapter 3471 before the finite epsilon-net ledger is exported. The route keeps finite nets, located endpoints, closed-bounded interval data, compact-metric carrier rows, Heine–Borel finite-cover rows, and RealUp seals visible as dependency rows rather than as hidden host compactness.

Proposition 4743.8 (Compact real interval real-source route)**.** *Every* $\mathsf{CompactRealInterval}^{\uparrow}$ *consumer that reads compact interval rows first passes through the endpoint* $\mathsf{Real}^{\uparrow}$ *source of chapter 38, the metric reading surface of chapter 1638, and the finite-net route recorded by chapter 5553. The compact interval packet therefore exposes real endpoints, metric comparison data, and finite-net compatibility before any compact interval row is exported.*

Chapter 4744

A Concrete Naming Certificate for $\mathsf{LocatedCutCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCutCompletion}^{\uparrow}$ packet names the finite lower–upper dyadic bracket surface by which Real and Completion consumers compare located-cut evidence with regular Cauchy readback. It sits beside the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational handoff of chapter 301, the terminal real seal of chapter 38, and the Cauchy–Dedekind bridge of chapter 2346. The packet is a BEDC-native carrier surface for located lower and upper brackets; it does not construct $\mathsf{Real}^{\uparrow}$ from primitive Dedekind cuts and does not add an ambient completeness principle.

Definition 4744.1 (Located-cut completion bracket carrier)**.** *A* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. B^- *is the lower dyadic bracket ledger. It records inhabited lower cells, lower endpoint witnesses, and lower refinement requests through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. B^+ *is the upper dyadic bracket ledger. It records inhabited upper cells, upper endpoint witnesses, and upper refinement requests through the same dyadic scale row.*
3. G *is the located comparison row. For every displayed lower endpoint and upper endpoint request, it records a finite separation, overlap, or refinement witness between* B^- *and* B^+*.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to expose only the finitely requested bracket observations.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that compares the displayed brackets with a regular rational Cauchy route.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger shared by* B^-, B^+, G, W, R*.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the lower bracket, upper bracket, located comparison, finite window, regular readback, and dyadic refinement rows have been displayed.*
8. H *records componentwise* hsame *transport for the bracket rows, comparison row, window row, readback row, dyadic row, real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the bracket-completion route*

$$B^-, B^+, G, D \longmapsto W, R \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCutCompletion}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^\uparrow}$ *row.*

The classifier accepts only packets that display both bracket ledgers, the shared located comparison row, the common dyadic refinement ledger, the finite window schedule, the regular readback row, the terminal real seal, and the structural rows. It has no coordinate for primitive Dedekind cuts, quotient equality, selected limits, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCutCompletionUp.LocatedCutCompletionTasteGate single carrier alignment`

Theorem 4744.2 (Located-cut bracket inhabitation)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier $L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^\uparrow}$ *surface includes inhabited lower and upper dyadic bracket reads. Every public bracket read must pass through the displayed rows B^- and B^+ and the shared dyadic ledger D before it can reach W, R, E, H, C, P, N.*

Proof. Carrier admission is projection from Definition 5554.1. The accepted packet exposes B^- and B^+ as separate lower and upper bracket ledgers, each tied to the same dyadic refinement row D.

The classifier has no bracket coordinate outside these ledgers. Therefore any public bracket read is a read of the displayed lower ledger or upper ledger together with the shared dyadic scale data.

The remaining rows W, R, E, H, C, P, N only schedule, read back, seal, transport, replay, source, and name that same bracket route. They cannot supply a detached lower or upper cut object. □

Theorem 4744.3 (Located-cut comparison obligation)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier, the row G is the only located comparison interface between the lower bracket ledger B^- and the upper bracket ledger B^+. A Real or Completion consumer cannot compare the two bracket sides without first reading B^-, B^+, G, D inside the same finite packet.*

Proof. Read the carrier rows named in Definition 5554.1. The lower and upper bracket ledgers are displayed separately, and the located comparison row G is the sole row that relates their endpoint requests.

A consumer that omits G has only two unrelated dyadic ledgers. A consumer that omits either bracket side has no two-sided located question to compare.

Thus the comparison route is forced through B^-, B^+, G, D. The stream, regular readback, real seal, and structural rows preserve the comparison after it is displayed, but they do not create a second comparison principle. □

Theorem 4744.4 (Common dyadic refinement stability)**.** *Let $L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier. Any refinement of a displayed lower bracket or upper bracket that remains inside the common dyadic ledger D preserves the same* $\mathsf{LocatedCutCompletion}^\uparrow$ *packet route through G, W, R, E, H, C, P, N.*

Proof. The carrier stores one dyadic refinement ledger D for both bracket sides. Refining a lower or upper endpoint inside D therefore keeps the refined observation in the same finite packet rather than moving it to an external cut object.

The located comparison row G is read after the two bracket ledgers and the shared dyadic scale are fixed. A refinement that remains in D is still a request to the same comparison row.

Finally W, R, E, H, C, P, N are attached to this packet route. They can replay and name the refined bracket read, but they cannot replace the common dyadic ledger by an independent refinement principle. □

Theorem 4744.5 (Classifier transport under bracket equivalence)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier, componentwise* hsame *transport of the lower bracket, upper bracket, located comparison, dyadic refinement, stream window, regular readback, real seal, replay, provenance, and local naming rows preserves classifier acceptance. Equivalence of bracket displays is therefore transported only through* H *and the displayed packet rows.*

Proof. The transport row H is a coordinate of the accepted carrier. Its scope is componentwise: it transports $B^-, B^+, G, W, R, D, E, C, P, N$ as rows of the same packet.

The classifier of Definition 5554.1 accepts exactly those displayed rows. Applying H preserves the rows it names and leaves no unmentioned quotient equality coordinate to inspect.

Hence bracket equivalence is a packet-level transport statement. It is not equality of primitive cuts and not quotient equality of completed real names. □

Theorem 4744.6 (Real seal handoff for located-cut completion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The terminal seal E *is reached only after the lower bracket, upper bracket, located comparison, common dyadic refinement, finite window schedule, and regular rational readback rows have been displayed.*

Proof. Start from the bracket rows B^-, B^+ and the located comparison row G. The shared dyadic ledger D fixes the common refinement scale on which both bracket sides are read.

The finite observation schedule W and regular rational readback row R then expose the bracket route to the Cauchy-facing side of the packet. At this point the consumer has only displayed finite rows, not a selected limit or quotient representative.

The terminal real seal E is therefore a handoff after B^-, B^+, G, D, W, R. The structural rows H, C, P, N transport, replay, source, and name the same handoff, so the excluded primitives and ambient completeness principle have no carrier coordinate. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCutCompletion}^{\uparrow}$ Real or Completion read factors through displayed lower bracket, upper bracket, located comparison, dyadic refinement, finite window, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-dedekind-bridge: $\mathsf{LocatedCutCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two-sided located bracket ledgers, common dyadic refinement, regular Cauchy readback, terminal Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5554.1, Definition 5554.2, Definition 5554.3, Definition 5554.4, Definition 5554.5, and Definition 5554.6 seed the finite located lower–upper bracket completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim primitive Dedekind cuts, quotient equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bracket inhabitation, located comparison, common dyadic refinement stability, classifier transport under bracket equivalence, Real seal handoff, and nonescape for BEDC.Derived.LocatedCutCompletionUp.

Constructive story: A LocatedCutCompletion$^{\uparrow}$ packet is generated from lower and upper dyadic bracket ledgers, a located comparison row, common dyadic refinement, finite StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4745

A Concrete Naming Certificate for $\mathsf{DedekindEtaCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{DedekindEtaCompletion}^{\uparrow}$ packet records an eta-completion route from regular rational names to located Dedekind-style real boundaries as finite BEDC NameCert data. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy criterion equivalence surface of chapter 3343, and the terminal real-name seal of chapter 38. The packet gives completion consumers a displayed eta route without quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4745.1 (DedekindEtaCompletion carrier)**.** *A* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, Q, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the rational observations retained by the eta route.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding each retained stream window to a regular rational observation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying positive-radius witnesses, half-radius refinements, and strict comparison cells used by the regular readback.*
4. *Q is the* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ *comparison row, recording the finite Cauchy criterion checks that certify the retained windows as a regular eta-family.*
5. *L is the located lower-and-upper cut ledger generated from S, R, D, Q; it records only displayed rational cut tests and their dyadic slack.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after the stream, regular, dyadic, criterion, and cut-ledger rows have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, Cauchy criterion, cut-ledger, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, R, D \longmapsto Q \longmapsto L \longmapsto A.$$

9. *P is the* Pkg *provenance over* $\mathsf{DedekindEtaCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $S, R, D, Q, L, A, H, C, P, N$ and has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DedekindEtaCompletionUp.DedekindEtaCompletionTasteGate single carrier alignment`

Theorem 4745.2 (DedekindEtaCompletion NameCert obligations)**.** *For an accepted* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier*

$$E = (S, R, D, Q, L, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the stream-window schedule, regular rational readback, dyadic tolerance ledger, Cauchy criterion comparison row, located cut ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring every rational observation used by the eta route to be addressed by S before it can be read by R;*
3. *dyadic tolerance exactness, requiring every comparison in the regular readback and every cut test in L to consume a displayed tolerance or refinement from D;*
4. *Cauchy criterion compatibility, requiring Q to certify the retained windows as the same regular eta-family before the cut ledger is exported;*
5. *cut locatedness, requiring L to expose lower and upper rational tests only through S, R, D, Q;*
6. *real-seal handoff, requiring A to be downstream of the same displayed eta route;*
7. *classifier transport and replay, requiring H and C to preserve exactly the listed rows and no external completion object;*
8. *non-escape, excluding quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 5555.1. The accepted packet exposes exactly the stream-window, regular readback, dyadic tolerance, Cauchy criterion, cut-ledger, real-seal, and structural rows.

Window exactness follows from the order of the carrier. A rational observation enters the eta route through S, and R can read it only as a retained finite window. There is no coordinate for an unaddressed stream or a quotient of streams.

Dyadic tolerance exactness is supplied by D. Every regular comparison and every cut test has to name a positive-radius witness, refinement, or strict comparison cell before it can be used by Q or L.

The Cauchy criterion row Q is the only compatibility surface. It certifies that the retained regular windows form the eta-family consumed by the cut ledger, so no selected limit or countable-choice schedule is introduced.

The cut ledger L is downstream of S, R, D, Q, and the real seal A is downstream of L. Hence lower and upper rational tests and Real-facing reads remain inside the displayed eta route.

Finally H, C, P, N transport, replay, source, and name only the listed rows. Since the excluded principles have no carrier coordinate, they cannot become local NameCert obligations. □

Theorem 4745.3 (DedekindEtaCompletion L10 handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyCriterionEquivalence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported object is a displayed eta-completion handoff from regular rational windows to a located real seal. It is not a quotient stream, selected host limit, countable-choice construction, trichotomy split, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the accepted carrier and project the public readback rows. A Real-facing or completion-facing consumer can reach the seal row A only by following the replay route named in Definition 5555.1.

The first stage is finite-window data. The row S names the inspected stream windows, and R reads those windows as regular rational observations.

The second stage is tolerance and criterion data. The dyadic ledger D supplies the radius cells consumed by the regular readback, and Q records the Cauchy criterion checks for the same retained family.

The cut ledger L then exposes the Dedekind-style lower and upper rational tests generated by that family. Since L is downstream of S, R, D, Q, it cannot select a host limit or quotient stream representative.

Finally A seals only this displayed route, while H, C, P, N preserve, replay, source, and name the same finite packet. Therefore the handoff exports the eta route and none of the excluded classical completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5555.1, Definition 5555.2, and Definition 5555.3 seed the finite eta-completion route over StreamName, RegSeqRat, DyadicRatCore, CauchyCriterionEquivalence, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, dyadic tolerance exactness, Cauchy criterion compatibility, cut locatedness, real-seal handoff, classifier transport, replay, and non-escape for BEDC.Derived.DedekindEtaCompletionUp.

Constructive story: A $\mathsf{DedekindEtaCompletion}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a CauchyCriterionEquivalence comparison row, a located cut ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4746

A Concrete Naming Certificate for $\mathsf{GITQuotient}^{\uparrow}$

Following Mumford's geometric invariant theory, the $\mathsf{GITQuotient}^{\uparrow}$ packet records the finite quotient-facing route from a projective or affine scheme presentation with group action data to invariant-ring rows, semistable-window rows, orbit-separation rows, and a displayed quotient name. It sits after the scheme carrier of chapter 153, the affine-variety surface of chapter 154, the projective-variety surface of chapter 155, the group certificate of chapter 41, the field certificate of chapter 45, and the ring certificate of chapter 43. The packet names a finite GIT quotient certificate without importing host quotient equality, stack objects, selected orbit representatives, or an external moduli functor.

Definition 4746.1 (GIT quotient carrier)**.** *A* $\mathsf{GITQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, A, P, G, I, Q, O, H, C, L, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Scheme}^{\uparrow}$ *source row naming the ambient scheme surface and the affine-cover ledger on which the quotient construction is read.*
2. *A is the* $\mathsf{AffineVar}^{\uparrow}$ *chart row used for invariant-ring computations on affine windows of S.*
3. *P is the* $\mathsf{ProjectiveVar}^{\uparrow}$ *presentation row naming the projective closure, homogeneous coordinates, linearization window, and projective incidence ledger.*
4. *G is the acting* $\mathsf{Group}^{\uparrow}$ *row over the displayed* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{Ring}^{\uparrow}$ *scalar rows consumed by the action and invariant-ring ledgers.*
5. *I is the invariant-ring ledger. It records the finite generator rows, relation rows, grading rows, and action-fixedness checks used to form the quotient coordinate packet.*
6. *Q is the quotient-name row. It records the displayed affine or projective quotient carrier obtained from I and the selected semistable window, but it is not a host quotient object.*
7. *O is the orbit-separation ledger. It records semistable-window membership, closed-orbit comparison rows, invariant-value comparison rows, and refusal rows for unstable histories.*
8. *H records componentwise* hsame *transport for the scheme, affine, projective, group, scalar, invariant-ring, quotient-name, orbit-separation, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from every quotient read through $S, A, P, G, I, Q, O, H, L, N$.*
10. *L is the* Pkg *provenance over* $\mathsf{GITQuotient}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, $\mathsf{AffineVar}^{\uparrow}$, $\mathsf{ProjectiveVar}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for host quotient equality, quotient stacks, selected orbit representatives, external moduli functors, or unstated invariant-theory completions.

Lean4 ✓ `BEDC.Derived.GITQuotientUp.GITQuotientTasteGate single carrier alignment`

Theorem 4746.2 (GIT quotient NameCert obligations)**.** *For every accepted* $\mathsf{GITQuotient}^{\uparrow}$ *carrier*

$$Z = (S, A, P, G, I, Q, O, H, C, L, N),$$

the local $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$ *surface consists exactly of scheme admission through S, affine-chart admission through A, projective-presentation admission through P, group-action and scalar admission through G, invariant-ring visibility through I, quotient-name exposure through Q, orbit-separation and semistable-window routing through O, componentwise transport through H, displayed replay through C, provenance through L, and local naming through N.*

Proof. Project the carrier of Definition 5556.1. The geometric source rows are S, A, P, G, the quotient-facing rows are I, Q, O, and the structural rows are H, C, L, N.

The invariant-ring obligation is finite. Its generator, relation, grading, and action-fixedness entries are rows inside I, and each is read through the affine chart, projective presentation, group action, and scalar certificates already displayed by A, P, G.

The quotient-name and orbit-separation obligations are then separate rows. The row Q exposes only the quotient carrier built from the invariant ledger and semistable window, while O records which semistable histories can be separated by invariant values and which unstable histories are refused.

Transport, replay, provenance, and local naming preserve exactly these rows. Since no coordinate names a host quotient, quotient stack, selected orbit representative, or external moduli functor, the listed rows exhaust the NameCert surface. □

Theorem 4746.3 (GIT quotient invariant-ring handoff)**.** *Every downstream read of an accepted* $\mathsf{GITQuotient}^{\uparrow}$ *quotient-name row factors through the invariant-ring ledger*

$$S, A, P, G \longmapsto I \longmapsto Q.$$

Thus the quotient carrier is exposed as finite invariant-ring and semistable-window data over the displayed scheme, affine, projective, group, field, and ring rows, not as a host quotient object or a stack presentation.

Proof. Start with the displayed source rows S, A, P, G. They supply the scheme cover, affine chart, projective presentation, action row, and scalar rows on which invariant functions are allowed to be read.

Read the invariant-ring ledger I. The finite generator, relation, grading, and fixedness checks in I are the only rows used to build the quotient coordinate packet.

The quotient-name row Q is then a projection from that packet together with the semistable-window data carried by O. The structural rows H, C, L, N can transport and replay this route, but they cannot add a host quotient object or a stack presentation. □

Theorem 4746.4 (GIT quotient orbit-separation nonescape)**.** *No accepted* $\mathsf{GITQuotient}^{\uparrow}$ *packet exports orbit separation outside the displayed semistable-window and invariant-value rows. If a consumer compares two semistable histories, the comparison must pass through* O *and the invariant values supplied by* I*; if a history lies outside the displayed semistable window, the refusal row in* O *blocks quotient-name export for that history.*

Proof. Project an accepted packet $Z = (S, A, P, G, I, Q, O, H, C, L, N)$. The only orbit-facing row is O, and the only scalar-valued separating data available to O are the invariant values recorded by I.

For semistable histories, O requires the displayed window-membership row before any comparison is read. The comparison is then routed to invariant values from I, so separation remains inside the finite invariant-ring ledger.

For histories outside the semistable window, the refusal row in O prevents the quotient-name row Q from being exported for that history. Componentwise hsame transport, Cont replay, Pkg provenance, and local naming preserve the refusal because none of them introduces an additional orbit classifier. □

Falsifiable prediction: Every accepted $\mathsf{GITQuotient}^{\uparrow}$ quotient-name read factors through a SchemeUp row, an AffineVarUp chart row, a ProjectiveVarUp presentation row, GroupUp action data over FieldUp and RingUp scalar rows, an invariant-ring ledger, a semistable-window ledger, orbit-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: scheme, affinevar, projectivevar, group, field, ring: $\mathsf{GITQuotient}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a group action, invariant-ring rows, semistable-window rows, quotient-name exposure, orbit-separation refusal, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GITQuotient}^{\uparrow}$ packet is generated from a SchemeUp source row, AffineVarUp chart row, ProjectiveVarUp presentation row, GroupUp action data over FieldUp and RingUp scalar rows, invariant-ring rows, semistable-window rows, orbit-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$, seed)

Scope closed: Definition 5556.1, Definition 5556.2, Definition 5556.3, and Definition 5556.4 seed the finite invariant-ring and semistable-window route for GIT quotient consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient equality, quotient stacks, selected orbit representatives, external moduli functors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-ring handoff, semistable-window routing, orbit-separation refusal, quotient-name exposure, replay stability, provenance exhaustion, and local naming for BEDC.Derived.GITQuotientUp.

Chapter 4747

A Concrete Naming Certificate for $\mathsf{AubinLions}^{\uparrow}$

Following Aubin and Lions, the $\mathsf{AubinLions}^{\uparrow}$ packet records the compactness route used for time-dependent Sobolev families as finite NameCert data. It sits over the Sobolev source of chapter 440, the weak-derivative time-translation surface of chapter 2210, the compact-window handoff of chapter 2218, the Cauchy readback of chapter 128, and the scalar seal of chapter 38. The packet names only the displayed compactness handoff; it does not import host subsequence selection, quotient completion, ambient real completeness, or an unrestricted compact embedding theorem.

Definition 4747.1 (Aubin-Lions carrier)**.** *An* $\mathsf{AubinLions}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, exposing the space-regularity, weak-derivative ledger, integrability window, and local naming data consumed by the compactness request.*
2. *T is the* $\mathsf{WeakDerivative}^{\uparrow}$ *time-translation row, recording the finite temporal difference windows, derivative bounds, and equicontinuity budget that keep translated reads inside the same Sobolev source.*
3. *R is the* $\mathsf{RellichKondrachov}^{\uparrow}$ *compact-window row, carrying the finite spatial compactness handoff available on inspected bounded windows.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *readback row, recording Cauchy comparison data for the compact-window images without completing a host quotient space.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for norms, radii, tolerances, and translation moduli.*
6. *H records componentwise* hsame *transport for the Sobolev source, time-translation row, compact-window row, complete-metric readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an Aubin-Lions compactness request through S, T, R, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{AubinLions}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{RellichKondrachov}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, R, M, E, H, C, P, N$. It has no coordinate for a selected subsequence, quotient function equality, host compact embedding, ambient completed Sobolev space, global time regularity theorem, unrestricted real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.AubinLionsUp.AubinLionsTasteGate single carrier alignment`

Theorem 4747.2 (Aubin-Lions NameCert obligations)**.** *For every accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier*

$$A = (S, T, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Sobolev source, weak-derivative time-translation row, Rellich–Kondrachov compact-window row, complete-metric readback, real seal, transport, replay, provenance, and local naming data;*
2. *source discipline, requiring every compactness request to enter through the* $\mathsf{Sobolev}^{\uparrow}$ *row S before temporal or spatial compactness data are read;*
3. *time-translation exactness, requiring T to display the derivative bounds, translation windows, and temporal equicontinuity budget used by the compactness route;*
4. *compact-window exactness, requiring R to expose only the inspected Rellich–Kondrachov spatial compactness handoff;*
5. *complete-metric and real readback, requiring M and E to supply only finite Cauchy comparisons, norms, radii, tolerances, and scalar moduli;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no selected subsequence, quotient function equality, host compact embedding, ambient completed Sobolev space, global time regularity theorem, unrestricted real completeness, Lean verification, or bridge credential.*

No Aubin-Lions obligation is stored outside these rows.

Proof. Project the accepted packet $A = (S, T, R, M, E, H, C, P, N)$. The carrier definition lists every visible component, so carrier admission is row-local.

The source discipline is forced by S. A compactness read first enters the Sobolev source row and sees only the displayed regularity, weak-derivative ledger, integrability window, and local naming data.

The temporal and spatial compactness rows then split the analytic work. The row T supplies the time-translation budget, while R supplies the Rellich–Kondrachov compact-window handoff. Neither row selects a subsequence or completes a quotient object.

The readback rows M and E expose only Cauchy comparisons and scalar tolerance data. Finally H, C, P, N preserve, replay, source, and name the same finite route, so the listed obligations exhaust the local NameCert surface. □

Theorem 4747.3 (Aubin-Lions compactness handoff)**.** *Every compactness consumer of an accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{WeakDerivative}^{\uparrow} \longmapsto \mathsf{RellichKondrachov}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and then through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the same packet. The consumer receives a finite compactness handoff certificate, not a selected convergent subsequence, quotient function equality, host compact embedding theorem, or ambient completeness theorem.*

Proof. Start with a compactness consumer request. By Definition 5557.2, the request first reads the Sobolev source row S.

The temporal route is then visible through T. It supplies the time-translation windows and derivative bounds needed before the spatial compactness row is read.

The spatial compactness handoff is exactly R. The complete-metric row M and real seal E only read back finite Cauchy comparisons and scalar budgets for that handoff.

The structural rows H, C, P, N keep the route inside the same packet. Since no selected subsequence, quotient equality, host compact embedding, or ambient completeness coordinate appears in the carrier, no consumer can read one from the packet. □

Theorem 4747.4 (Aubin-Lions time-translation nonescape)**.** *Let $A = (S, T, R, M, E, H, C, P, N)$ be an accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier. Any consumer that keeps the Sobolev source S and the Rellich–Kondrachov compact-window row R but deletes the weak-derivative time-translation row T cannot export the Aubin-Lions compactness handoff from this packet. Such a consumer has only a spatial compact-window read and must not treat it as the time-dependent compactness route named by* $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$.

Proof. Project the carrier of Definition 5557.1. The compactness route contains both the Sobolev source row S and the time-translation row T before the Rellich–Kondrachov compact-window row R is consumed.

Apply Definition 5557.2. The time-translation exactness obligation requires the derivative bounds, temporal difference windows, and equicontinuity budget displayed by T.

Now delete T. The remaining rows may still expose the Sobolev source and a spatial compact-window handoff, but they no longer display the temporal route that distinguishes an Aubin-Lions packet from a static Rellich–Kondrachov read.

The structural rows cannot fill that absence. They transport and replay existing rows only, so without T the consumer cannot export the Aubin-Lions compactness handoff from this packet. □

Falsifiable prediction: Every $\mathsf{AubinLions}^{\uparrow}$ compactness read factors through displayed SobolevUp, WeakDerivativeUp, RellichKondrachovUp, CompleteMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any compactness consumer receives it.

Independence witness: sobolev, weak-derivative, rellich-kondrachov, completemetric, real: $\mathsf{AubinLions}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Sobolev source, time-translation row, compact-window row, complete-metric readback, real scalar seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$, seed)

Scope closed: Definition 5557.1, Definition 5557.2, Definition 5557.3, and Definition 5557.4 seed the finite compactness route over Sobolev, weak-derivative, Rellich–Kondrachov, complete-metric, and real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subsequence selection, quotient completion, ambient real completeness, unrestricted compact embedding, global time regularity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source discipline, time-translation exactness, compact-window exactness, complete-metric readback, scalar sealing, componentwise transport, replay, provenance, and local naming for BEDC.Derived.AubinLionsUp.

Constructive story: An $\mathsf{AubinLions}^{\uparrow}$ packet is generated from a Sobolev source row, weak-derivative time-translation row, Rellich–Kondrachov compact-window row, CompleteMetric readback row, Real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4748

A Concrete Naming Certificate for $\mathsf{CompactUniformPullback}^{\uparrow}$

Following Bishop and Bridges on compact-uniform continuity, the $\mathsf{CompactUniformPullback}^{\uparrow}$ packet realizes the compact-uniform routes of chapter 9237 as a finite pullback carrier joining compact-net data, pointwise-modulus rows, dyadic tolerances, regular rational readbacks, and the terminal real-name seal. It sits after the compact metric carrier of chapter 129, the continuous-map surface of chapter 130, the uniform-modulus consumer of chapter 369, the dyadic rational core of chapter 291, the regular rational sequence route of chapter 301, and the real-name seal of chapter 38. The packet names the recurring compact-uniform handoff without exporting a host modulus selector, quotient point equality, open-cover choice, or ambient Heine–Cantor theorem.

Definition 4748.1 (Compact uniform pullback carrier)**.** *A* $\mathsf{CompactUniformPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, M, D, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row naming the compact source, finite net, located-distance ledger, and coverage witnesses used by the pullback.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the graph, pointwise-modulus ledger, source metric, and target metric.*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row requested by downstream compact-uniform arguments.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records dyadic tolerances, half-radius budgets, and finite lower-bound comparisons used in the compact-net fold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route for the rational precision histories produced by the fold.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that exposes the final precision and target estimate only after the dyadic and regular-rational rows are accepted.*
7. *T is the pullback ledger. It records how a target tolerance request is pulled back through F, through compact finite-net rows in K, through the dyadic budget in D, through the regular-rational readback in R, and into the uniform-modulus consumer row M.*
8. *H records componentwise* hsame *transport for compact metric, continuous map, uniform-modulus, dyadic tolerance, regular readback, real seal, pullback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from every compact-uniform request through* $K, F, M, D, R, E, T, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformPullback}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$ *row.*

The classifier compares accepted pullback packets only through these finite rows. It has no coordinate for a host modulus selector, quotient point equality, open-cover choice, selected Lebesgue number, or ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformPullbackUp.CompactUniformPullbackTasteGate single carrier alignment`

Theorem 4748.2 (Compact uniform pullback NameCert obligations)**.** *For every accepted* CompactUniformPullback *carrier*

$$U = (K, F, M, D, R, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, continuous-map pointwise-modulus admission through F, uniform-modulus consumer exposure through M, dyadic tolerance and lower-bound routing through D, regular-rational readback through R, real sealing through E, pullback ledger visibility through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5558.1. The source rows are K, F, M, the arithmetic and seal rows are D, R, E, the pullback row is T, and the structural rows are H, C, P, N.

The finite-net obligation is read through K, while pointwise modulus rows are read through F. The pullback ledger T is the only row that is allowed to combine those data into a tolerance-only request for M.

The arithmetic handoff is also explicit. Dyadic tolerance and lower-bound comparisons stay in D, regular-rational readback stays in R, and the final real estimate is exposed only through E.

Transport, replay, provenance, and local naming preserve this route. Because the carrier has no coordinate for a selected modulus, quotient point equality, open-cover choice, selected Lebesgue number, or host compactness theorem, the listed rows exhaust the NameCert surface. □

Theorem 4748.3 (Compact uniform pullback modulus handoff)**.** *Every downstream read of an accepted* $\mathsf{CompactUniformPullback}^{\uparrow}$ *packet factors through*

$$K, F, D, R, E, T \longmapsto M.$$

Thus a compact-uniform consumer receives a $\mathsf{UniformModulus}^{\uparrow}$ *row only after the compact finite-net data, pointwise-modulus rows, dyadic tolerance budget, regular-rational readback, real seal, and pullback ledger have all been displayed inside the same finite carrier.*

Proof. Begin with a downstream target tolerance request. The continuous-map row F supplies the pointwise-modulus rows inspected at compact-net centers, and the compact metric row K supplies the finite net and coverage witnesses used to choose those centers inside BHist.

The dyadic tolerance row D performs the finite lower-bound comparisons. Its output is read through R as a regular rational precision history and then through E as a real-sealed estimate.

The pullback ledger T records that these rows are consumed in this order before the consumer row M is exposed. Componentwise transport and replay can only preserve the displayed route, so no host modulus selector or open-cover choice is exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformPullback}^{\uparrow}$ modulus read factors through a CompactMetricUp finite-net row, a ContinuousMapUp pointwise-modulus row, a UniformModulusUp consumer row, DyadicRatCoreUp tolerance arithmetic, RegSeqRatUp readback, a RealUp seal, a pullback ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, regseqrat, real: $\mathsf{CompactUniformPullback}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact finite-net coverage, pointwise-modulus rows, dyadic tolerance budgets, regular rational readback, real sealing, a downstream uniform-modulus consumer, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformPullback}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, a UniformModulusUp consumer row, DyadicRatCoreUp tolerance arithmetic, RegSeqRatUp readback, a RealUp seal, a compact-uniform pullback ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5558.1, Definition 5558.2, and Definition 5558.3 seed the finite compact-net, pointwise-modulus, dyadic-tolerance, regular-readback, and real-seal handoff to uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host modulus selection, quotient point equality, open-cover choice, selected Lebesgue numbers, ambient Heine–Cantor theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, pointwise-modulus admission, dyadic tolerance exactness, regular readback, real sealing, pullback ledger visibility, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformPullbackUp.

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, and the terminal Real seal of chapter 38 before routing the child files below.

Chapter 4749

A Concrete Naming Certificate for $\mathsf{RegularCauchyGluing}^{\uparrow}$

Bishop–Bridges style regular Cauchy gluing records the finite route by which two compatible local regular rational windows can be read as one BEDC real-name boundary without passing through quotient representatives or an ambient completeness theorem. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the reindexing route of chapter 1568.

Definition 4749.1 (Regular Cauchy gluing carrier)**.** *A* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right compatible regular Cauchy windows, each carrying finite* $\mathsf{StreamName}^{\uparrow}$ *observations and a local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger recording the shared dyadic radii, endpoint comparisons, overlap budgets, and refinement witnesses used by both windows.*
3. *S is the joint* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing exactly the finite indices inspected on the left window, right window, and overlap.*
4. *Q is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by reading the two local regular rows through the common ledger D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, R, D, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for the two windows, common dyadic ledger, finite observations, glued readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite gluing route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the two local windows are compatible on the displayed overlap through D, S, Q, H, C, P, N. It has no coordinate for quotient sequence equality, selected representatives, host real equality, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4749.2 (Regular Cauchy gluing classifier)**.** *The* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *classifier compares accepted carriers only through the tuple* $L, R, D, S, Q, E, H, C, P, N$*. Two packets are indistinguishable for this classifier precisely when the left and right window rows, the common dyadic refinement ledger, the finite observation row, the glued regular readback, the terminal real seal, and the transport, replay, provenance, and naming rows are componentwise the same under the displayed* hsame *routes.*

Theorem 4749.3 (Regular Cauchy gluing NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *surface consists exactly of carrier admission for the two compatible regular Cauchy windows, common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through* S*, glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The analytic rows before the seal are exactly the two regular windows L, R, the common dyadic ledger D, the finite observation row S, and the glued regular readback Q.

The overlap obligation is finite. Its comparisons are stored in D and read through the same inspected indices in S, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ rows feed one glued readback row Q without adding a quotient representative.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections gives the listed $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ surface, and the carrier definition leaves no hidden row for selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 4749.4 (Regular Cauchy gluing overlap stability)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Any refinement of the displayed overlap that is transported by* H *and replayed by* C *preserves the same glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Consequently downstream Real and Completion consumers receive only the finite gluing packet and not quotient stream equality, selected representatives, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the displayed overlap ledger. The common dyadic row D records every radius and endpoint comparison used by both local windows, while S records the finite observations on which those comparisons are made.

Apply the classifier of Definition 5899.2. A refinement transported by H and replayed by C can change neither the componentwise ledger coordinates nor the finite observation coordinates exposed to the classifier.

The glued readback row Q is therefore the same row after refinement, and the $\mathsf{Real}^{\uparrow}$ seal E is read only through Q together with H, C, P, N. Since the carrier contains no quotient, selector, host-equality, choice, or completeness coordinate, none of those data can be exported by overlap stability. □

Theorem 4749.5 (Regular Cauchy gluing common-refinement exactness)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Every glued readback exported from* G *must expose the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The exactness route is transported by* H *and replayed by* C*, and it exports no quotient stream equality, selected representative, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted carrier. The row D is the common dyadic refinement ledger shared by the two local regular Cauchy windows, while Q is the glued regular readback and E is the terminal Real seal.

Apply overlap stability from Definition 5899.4. Any accepted refinement of the overlap preserves the same Q and E only through the displayed ledger and observation rows, so a glued readback cannot bypass D.

Finally use the Cauchy-reindexing handoff of Definition 5899.7. It reads D, S, Q as the finite reindexing-compatible route before any Real-facing consumer reaches E, with H, C, P, N preserving and naming the same route. □

Theorem 4749.6 (Regular Cauchy gluing overlap induction)**.** *Let $G = (L, R, D, S, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. For any finite chain of displayed overlap refinements transported by H and replayed by C, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E remain unchanged. The resulting route still factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyReindexing}^{\uparrow}$, *and it exports no quotient stream equality, selected representatives, host equality, countable choice, or ambient completeness.*

Proof. Induct on the displayed finite refinement list.

The empty list is the overlap-stability theorem of Definition 5899.4: the common dyadic ledger D and the finite observation row S already determine the same glued readback Q and seal E.

For the step, assume the earlier refinements preserve Q and E. Compose the next displayed refinement with the induction hypothesis, transport the component rows by H, and replay the same finite route by C.

Apply the classifier of Definition 5899.2 componentwise. Since the classifier reads only $L, R, D, S, Q, E, H, C, P, N$ the next refinement cannot add a quotient, representative selector, host equality row, choice schedule, or completeness witness; it only repeats the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 4749.7 (Regular Cauchy gluing Cauchy-reindexing handoff)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier factors its gluing route through the finite* $\mathsf{CauchyReindexing}^{\uparrow}$ *surface recorded in P. The handoff exposes the common dyadic refinement ledger D, the finite observation row S, and the glued regular readback row Q as a reindexing-compatible route, and it exposes no global quotient of regular Cauchy names or ambient completeness principle.*

Proof. Read the package provenance P. It names the route through $\mathsf{CauchyReindexing}^{\uparrow}$ together with $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The carrier rows D, S, Q are exactly the finite data consumed by that route: D supplies the common refinement ledger, S supplies the inspected windows, and Q supplies the regular rational readback after the two local windows agree on the overlap.

The support rows H, C, N transport, replay, and name this same finite path. Thus the handoff is reindexing-compatible BEDC data, not a quotient of regular names, a selected representative, or a completeness theorem. □

Theorem 4749.8 (Regular Cauchy gluing obligation closure handoff)**.** *Every obligation-closure read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier must expose the full row tuple*

$$L, R, D, S, Q, E, H, C, P, N.$$

The route factors through the common $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and glued readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and*

the $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff named by* P *before any downstream real-name consumer may read the gluing packet.*

Proof. Start with the carrier and NameCert obligations. The accepted packet exposes $L, R, D, S, Q, E, H, C, P, N$, and the obligation surface has no smaller projection that still records the two local windows, their common refinement, and the glued readback.

Use the Cauchy-reindexing handoff of Definition 5899.7. It forces the gluing read through D, S, Q and through the provenance row P naming $\mathsf{CauchyReindexing}^{\uparrow}$, while the carrier rows L, R provide the two regular windows.

The rows E, H, C, N then seal, transport, replay, and name the same finite route. Thus obligation closure reads exactly the displayed tuple and factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 4749.9 (Regular Cauchy gluing Real seal ordering)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reachable only after the common dyadic refinement row* D*, the finite observation row* S*, and the glued readback row* Q *have been displayed. The seal-ordering route asserts no quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Project the carrier definition. The row E is listed as the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after the two local windows, the common refinement ledger, the finite observations, and the glued readback have been displayed.

Apply the overlap-stability and Cauchy-reindexing handoff theorems. They keep the common dyadic refinement in D, the inspected windows in S, and the glued regular readback in Q visible before any real-name consumer can inspect E.

Since quotient stream equality, selected representatives, host equality, choice schedules, and ambient completeness are not carrier coordinates, the ordering statement cannot export them. It records only the displayed route from D, S, Q to the terminal seal E. □

Theorem 4749.10 (Regular Cauchy gluing local tail compatibility)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the two local regular-Cauchy windows* L *and* R *can be consumed by a* $\mathsf{Real}^{\uparrow}$*-seal reader only after the overlap row, the common-refinement row, and the Cauchy-reindexing row exhibit a shared* $\mathsf{RegSeqRat}^{\uparrow}$ *tail boundary. The compatibility route is the finite route through* D, S, Q, H, C, P, N *supplied by the carrier, and it exports no quotient stream equality, selected representative, or ambient completeness.*

Proof. Project the carrier of Definition 5899.1. The two local windows L, R are not read independently by a Real-facing consumer; the packet first displays their shared dyadic ledger D, finite observations S, and glued readback Q.

Apply overlap stability, Definition 5899.4, and common-refinement exactness, Definition 5899.5. These theorems keep the overlap row and the common refinement visible before Q is consumed, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ tails are compared at the displayed boundary rather than by host stream equality.

Apply overlap induction, Definition 5899.6, to finite chains of displayed refinements, and then use the Cauchy-reindexing handoff, Definition 5899.7. The resulting route makes the shared tail boundary stable under H and replayable by C before E is read.

Since quotient stream equality, selected representatives, and ambient completeness are absent from Definition 5899.1, the compatibility theorem can expose only the finite D, S, Q, H, C, P, N route. □

The local-tail compatibility row is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window surface of chapter 301 and the $\mathsf{StreamName}^{\uparrow}$ finite-observation surface of chapter 126. The exact dependency is the common-tail rows displayed by D and Q, together with finite-window readback through S, before the $\mathsf{Real}^{\uparrow}$ seal E is available to any downstream consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet is generated from two compatible regular Cauchy windows, a common $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ observations, a glued $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5899.1, Definition 5899.2, Definition 5899.3, Definition 5899.4, Definition 5899.6, Definition 5899.7, Definition 5899.8, Definition 5899.9, and Definition 5899.10 seed the finite gluing route for compatible regular Cauchy windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, common dyadic-refinement exactness, overlap stability, CauchyReindexing handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyGluingUp.

Theorem 4749.11 (Regular Cauchy gluing L10 sibling route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet reads its local gluing route through the L10 sibling lattice before it exports a NameCert row. The route first consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the Cauchy reindexing handoff of chapter 1568.*

Thus the gluing packet exports only a local finite NameCert route over the displayed dyadic, stream, regular-rational, Real-seal, and reindexing rows. It does not expose quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 5899.1. Its visible rows are two local regular windows, the common dyadic ledger, the finite observation row, the glued regular readback, the Real seal, and the transport, replay, provenance, and naming rows.

Apply Definition 5899.3. The local NameCert surface must read exactly those rows, with the dyadic ledger, StreamName windows, RegSeqRat readback, and Real seal appearing before the certificate is exported.

Use Definition 5899.7 for the reindexing component. It identifies the same dyadic, stream, and regular readback rows as a $\mathsf{CauchyReindexing}^{\uparrow}$-compatible route.

Thus the dependency on chapter 1568 is consumed before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal are available to the gluing certificate.

The exclusions are inherited from the carrier inventory and the overlap stability theorem. Since no coordinate stores quotient stream equality, selected representatives, host real equality, choice, or ambient completeness, the L10 sibling route remains a finite local NameCert route. □

Theorem 4749.12 (Regular Cauchy gluing RegSeqRat route)**.** *Every glued readback row of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must pass through the existing* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling surface of chapter 301 and the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff of chapter 1568 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The local obligations of Definition 5899.3 therefore expose a regular rational readback and a reindexing route before any Real-facing consumer may read the glued value.*

Proof. Start from the local L10 route of Definition 5899.11. It already orders the dyadic ledger, finite StreamName windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyReindexing}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal before the gluing NameCert row is exported.

Apply the local obligations of Definition 5899.3. The glued readback row is one of the displayed certificate rows, so it cannot be consumed without the same regular rational surface and structural support rows.

Use Definition 5899.7. The reindexing handoff keeps the dyadic, stream, and regular readback data aligned before the terminal Real seal appears. Thus a Real-facing read factors through RegSeqRat and CauchyReindexing rather than bypassing them. □

Theorem 4749.13 (Regular Cauchy gluing pasting nonescape)**.** *Every accepted pasting consumer of a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite route of Definition 5899.1, satisfy the local obligations of Definition 5899.3, and pass through the L10 sibling route of Definition 5899.11 before it exports any NameCert row. In particular, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently no such pasting consumer exports quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as a coordinate of the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *read.*

Proof. Project the accepted carrier in Definition 5899.1. Its displayed route already contains the common dyadic ledger, finite StreamName observations, glued RegSeqRat readback, Real seal, and the support rows for transport, replay, provenance, and naming.

Apply Definition 5899.3. A pasting consumer can export a local NameCert row only by preserving exactly those displayed rows, so the consumer has no separate entry point around the dyadic, stream, regular readback, or seal coordinates.

Apply Definition 5899.11. The L10 route orders the same rows through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before export.

The refused objects are absent from the carrier and absent from the sibling route. Thus quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness cannot be repacked as a pasting consumer of the accepted $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet. □

Theorem 4749.14 (Regular Cauchy gluing separator-use route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that exports an apartness read or a located-comparison read must first pass through the* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *route of chapter 7930. Ordinary gluing consumers still export only the finite dyadic, stream,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows named by Definition 5899.11.*

Proof. Start with the L10 sibling route of Definition 5899.11. It exposes the dyadic tolerance rows, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before the local gluing NameCert row is exported.

For a consumer that reads only gluing data, those rows exhaust the available surface by Definition 5899.13. The carrier has no coordinate for apartness, located comparison, quotient equality, selected representatives, choice, or ambient completeness.

If the same consumer asks for apartness or located comparison, apply the separator sibling dependency of Definition 7930.3. That route requires the stream, regular-rational, dyadic, Real-seal, located-cut, and apartness-field rows to be displayed through $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ before such a read is available.

Thus the gluing packet itself remains the ordinary finite route, while apartness-facing or located-comparison-facing reads factor through the separator chapter rather than being smuggled into the gluing carrier. □

Theorem 4749.15 (Regular Cauchy gluing completion consumer route)**.** *Every Bishop-completion or Real-completion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the same finite L10 sibling route before exporting a gluing* NameCert *row. In particular, the consumer first exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The exported gluing certificate has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already orders the dyadic ledger, stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any local gluing NameCert row is exported.

Apply the RegSeqRat route of Definition 5899.12. A completion-facing read of the glued value therefore cannot bypass the regular rational surface or the reindexing route on its way to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply the pasting nonescape theorem, Definition 5899.13. It keeps every gluing consumer inside the finite carrier route and excludes quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness as exportable coordinates.

Finally use the separator-use route of Definition 5899.14. Apartness-facing or located-comparison-facing completion reads factor through the separator sibling; ordinary Bishop-completion and Real-completion gluing reads still expose only the dyadic, stream, regular-rational, reindexing, and Real-seal rows before the gluing NameCert is named. □

Theorem 4749.16 (Regular Cauchy gluing uniform Cauchy criterion route)**.** *Every uniform-Cauchy-criterion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite sibling route before exporting criterion data. The consumer must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of the gluing route.*

Consequently the criterion consumer exports only the finite BEDC packet obtained from those rows, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *ledger. It does not export quotient sequence equality, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already places the dyadic tolerance ledger, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any gluing NameCert row is exported.

Apply the RegSeqRat route of Definition 5899.12 and the completion consumer theorem Definition 5899.15. Together they force a completion-facing criterion read through the regular rational surface, the reindexing handoff, and the Bishop-completion consumer rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally use the downstream sibling route of Definition 5899.18. It permits only concrete downstream consumers after the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows have been displayed. Since the carrier and all cited routes omit quotient sequence equality, selected representatives, countable choice, host real equality, and ambient completeness, none of those objects can be repacked as uniform-Cauchy-criterion data. □

Theorem 4749.17 (Regular Cauchy gluing L10 sibling dependency)**.** *Before any gluing consumer reads the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row, it must display the finite sibling dependency route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *observation windows of chapter 126,*

the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the completion-facing handoff of chapter 3665, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route may also be read by Bishop-completion consumers through the completion consumer theorem Definition 5899.15, but it remains a finite dependency lattice rather than quotient sequence equality, selected representatives, countable choice, host real equality, or ambient real completeness.*

Proof. Start with Definition 5899.11. It already orders the dyadic tolerance rows, stream observation windows, regular-rational readback, Cauchy reindexing handoff, and Real seal before the local $\mathsf{NameCert}$ row is exported.

Apply Definition 5899.15. A Bishop-completion or Real-completion consumer therefore reads the same finite row sequence before it can use the gluing packet as completion-facing data.

The dependency is finite because every cited sibling appears as a displayed row or handoff in the local route. No step asks for quotient sequence equality, a selected representative of a Cauchy class, countable choice over tails, host real equality, or an ambient completeness theorem. □

Theorem 4749.18 (Regular Cauchy gluing downstream sibling route)**.** *Any cofinal, interleaving, or comparison-facing consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first pass through the finite sibling route of Definition 5899.17. After the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *rows have been displayed, such a consumer may continue only through a concrete downstream sibling such as chapter 414 or chapter 3514. The gluing packet itself exports no detached cofinal route, no interleaving selector, no quotient equality of Cauchy names, no selected representative, and no ambient completeness theorem.*

Proof. Start with Definition 5899.17. It forces every gluing consumer to display the regular rational rows, finite stream windows, dyadic tolerance rows, Bishop-completion handoff, and terminal Real seal before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ row is read.

Apply the completion consumer theorem Definition 5899.15. A completion-facing read therefore receives the same finite route, not a quotient stream or selected representative.

For an interleaving-facing read, use the public sibling surface of chapter 414. It supplies the alternating selector and combined modulus ledger only after the relevant $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are displayed.

For a comparison-facing read, use chapter 3514. That packet supplies the even–odd comparison and shared threshold rows as a separate finite consumer surface. Thus any cofinal or interleaving continuation is downstream of the existing L10 sibling dependency, and none of those rows is smuggled into the gluing carrier itself. □

Theorem 4749.19 (Regular Cauchy gluing tail-agreement composition)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads a glued* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet through Definition 5899.18 must continue through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699 before treating the glued readback as completion-facing tail agreement. The composed route also reads the Cauchy limit seal of chapter 386; the gluing packet alone exports no ambient tail equality, selected representative, quotient sequence equality, or completeness witness.*

Proof. Start with Definition 5899.18. It permits downstream consumers only after the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows have been displayed.

A completion-facing tail read then has an additional boundary. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ chapter supplies the route that turns a Real-facing readback into displayed tail-agreement data, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter supplies the limit-facing seal used by completion consumers.

Thus the row order is finite: first the local gluing sibling route, then the tail-agreement seal, and finally the Cauchy-limit seal. Because each step only reuses displayed rows, no consumer can replace the route by host tail equality, a selected Cauchy representative, quotient sequence equality, or ambient completeness. □

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes its L10 sibling lattice before any gluing read is exported: the $\mathsf{StreamName}^{\uparrow}$ finite windows provide the inspected overlap rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular readback, $\mathsf{DyadicRatCore}^{\uparrow}$ carries the shared tolerance ledger, $\mathsf{Real}^{\uparrow}$ seals the terminal value, and chapter 5544 provides the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ schedule that keeps the two local Cauchy windows on one displayed budget.

Theorem 4749.20 (Regular Cauchy gluing L10 forward route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer reaches Real-facing gluing only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of the local carrier. The route first reads the name certificate construction, then the L10 sibling route, and finally the Bishop-completion consumer route before any gluing conclusion is exported. It refuses quotient stream equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as coordinates of the gluing read.*

Proof. Project the local carrier and obligations from Definition 5899.1 and Definition 5899.3. The rows available to a consumer are the two compatible windows, the common dyadic ledger, finite observations, glued regular readback, Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the consumer through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a local gluing certificate is exported.

Apply the Bishop-completion consumer theorem Definition 5899.22. A Real-facing or Bishop-completion consumer must display the same D, S, Q, E, H, C, P, N route, so the forward path remains finite and local to the accepted packet.

The refused objects are absent from every cited step. The name certificate construction supplies no quotient stream equality; the L10 sibling route supplies no selected limit or countable-choice schedule; and the Bishop-completion consumer route supplies no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Hence every accepted forward gluing read factors through the displayed windows, readbacks, tolerance ledgers, and Real seal. □

Theorem 4749.21 (Regular Cauchy gluing sibling forward link)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that is read as a gluing consumer first passes through the L10 sibling route of Definition 5899.11. In particular, the consumer must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126 before the local gluing* $\mathsf{NameCert}$ *row is available.*

The forward link is therefore a concrete sibling dependency route rather than a detached hub edge. It also preserves the exclusions already named by the L10 route: no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is exported by the gluing read.*

Proof. Start with the carrier and obligation rows of Definition 5899.1 and Definition 5899.3. They display the paired local windows, common dyadic ledger, finite observations, glued regular readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Apply the L10 sibling route of Definition 5899.11. That route orders the same gluing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any gluing certificate can be exported.

Thus the hub has a visible child route into the RegSeqRat and StreamName sibling chapters. The refused objects remain absent because neither the local carrier nor the L10 sibling route contains coordinates for quotient stream equality, selected representatives, choice, host equality, or ambient completeness. □

Theorem 4749.22 (Regular Cauchy gluing Bishop completion consumer)**.** *Every* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first expose the same finite gluing carrier rows already displayed by Definition 5899.1. In particular, before the Bishop completion route may read the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *glued readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff recorded in* P*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

Thus the Bishop completion consumer receives the gluing packet only as a finite L10 route over dyadic, stream, regular-rational, Real-seal, reindexing, transport, replay, provenance, and naming data. It receives no quotient stream equality, selected representative, host real equality, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The rows that a completion-facing consumer can inspect are exactly the two local windows, the common dyadic ledger, the finite observations, the glued readback, the Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the gluing packet through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$ before a NameCert row is exported.

Now read the Bishop completion carrier of chapter 3665. Its completion-facing route consumes dyadic precision, finite stream windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming. The gluing rows D, S, Q, E, H, C, P, N are therefore the only rows that can feed such a consumer from $\mathsf{RegularCauchyGluing}^{\uparrow}$.

The exclusions follow from the same carrier inventory. Since neither Definition 5899.1 nor the Bishop completion route supplies quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness as a coordinate, the consumer cannot export any of those objects from the gluing packet. □

Theorem 4749.23 (Regular Cauchy gluing Real-tail forward route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that asks for a tail-agreement seal is a forward dependency of the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route in chapter 699. Concretely, the gluing packet first exposes the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the paired* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the local* $\mathsf{Real}^{\uparrow}$ *seal* E*; only then may the consumer enter the Real-tail agreement and Cauchy-limit seal route through* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*.*

Proof. Project the accepted gluing carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The Real-facing rows available before any tail agreement are exactly the common dyadic ledger D, finite observation row S, glued regular readback Q, terminal seal E, transport row H, replay row C, provenance row P, and local name row N.

Apply the Real seal ordering theorem Definition 5899.9. It forces the route to display D, S, Q before E is readable, so a consumer cannot jump from the two local windows directly to a Real-tail agreement row.

Now use the Real-tail agreement completion-consumer boundary of chapter 699. That boundary receives paired windows, dyadic readback, agreement, Cauchy-limit sealing, and Real or Com-

pletion endpoints as one finite route. Feeding it from $\mathsf{RegularCauchyGluing}^{\uparrow}$ therefore makes the gluing read a forward dependency of that Real-tail route rather than a separate kernel island.

The exclusions are inherited from the two carriers. Neither Definition 5899.1 nor the Real-tail agreement route supplies quotient stream equality, selected representatives, host convergence, countable choice, or ambient completeness as a row, so no such object can be exported by this forward dependency. □

Chapter 4750

A Concrete Naming Certificate for $\mathsf{YoungSymmetrizer}^{\uparrow}$

Following Fulton and Harris on Young symmetrizers and polynomial representations, the $\mathsf{YoungSymmetrizer}^{\uparrow}$ packet records the finite tableau rows that turn a partition shape into row symmetrization, column antisymmetrization, and a displayed tensor-to-matrix action. It sits between the shape discipline of chapter 201, the character readback of chapter 245, the Grothendieck consumer of chapter 246, the multilinear source of chapter 87, and the displayed action rows of chapter 49. The packet is finite: it names the symmetrizer route without importing semisimplicity, Specht module irreducibility, Schur functor exactness, host group algebra idempotence, or quotient representation categories.

Definition 4750.1 (YoungSymmetrizerUp carrier)**.** *A* $\mathsf{YoungSymmetrizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N).$$

Here P is the consumed $\mathsf{Partition}^{\uparrow}$ *shape row; D is the Young diagram row for that shape; T is a tableau row whose entries fill the boxes of D; R is the row-stabilizer ledger listing the permutations that preserve each row of T; C is the column-stabilizer ledger listing the permutations that preserve each column of T; S is the row-symmetrizer formal sum over R; A is the column-antisymmetrizer formal signed sum over C; E is the ordered composite row for the formal Young symmetrizer SA; M is the displayed tensor and matrix action row over the* $\mathsf{TensorProduct}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *sources; H gives componentwise* hsame *transport for $P, D, T, R, C, S, A, E, M$; U gives displayed* Cont *replay for the same finite route; V is the consumer ledger for* $\mathsf{CharacterTheory}^{\uparrow}$ *and* $\mathsf{RepresentationRing}^{\uparrow}$ *readback; G is the* Pkg *provenance over* $\mathsf{Partition}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*; and N is the local* $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}$ *row.*

Theorem 4750.2 (YoungSymmetrizerUp NameCert obligations)**.** *For every accepted carrier $Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N)$ of Definition 5560.1, the local* $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

- *shape admission, requiring P to expose the partition shape and D to list precisely its diagram boxes;*
- *tableau admission, requiring T to fill the boxes of D and to induce the row and column incidence ledgers used by R and C;*
- *row symmetrizer exactness, requiring S to be the finite formal sum over the row-stabilizer ledger R;*

- *column antisymmetrizer exactness, requiring A to be the finite signed formal sum over the column-stabilizer ledger C;*
- *action display, requiring E and M to route the composite SA through the tensor and matrix rows rather than through an ambient group-algebra object;*
- *consumer confinement, requiring every* $\mathsf{CharacterTheory}^{\uparrow}$ *or* $\mathsf{RepresentationRing}^{\uparrow}$ *read to pass through V after the finite tensor/matrix action has been displayed;*
- *structural replay, requiring H, U, G, N to transport, replay, account for, and name exactly the rows already listed.*

No accepted $\mathsf{YoungSymmetrizer}^{\uparrow}$ *packet exports an irreducible module theorem, a complete character formula, a semisimple decomposition, or equality inside an unlisted host group algebra.*

Lean4 ✓ `BEDC.Derived.YoungSymmetrizerUp.YoungSymmetrizerTasteGate single carrier alignment`

Proof. Inspect the coordinates of Y. The shape and tableau rows P, D, T determine the finite incidence data, while R and C are only the displayed stabilizer ledgers for that incidence data.

The formal rows S, A, E are then determined by finite replay over R and C. Their only action surface is the displayed tensor/matrix row M, so a consumer cannot replace the route by an ambient group-algebra calculation without leaving the packet.

Finally H, U, V, G, N do not add new mathematical rows. They transport, replay, expose, account for, and name the finite route already visible in $P, D, T, R, C, S, A, E, M$. □

Theorem 4750.3 (YoungSymmetrizerUp idempotent route)**.** *Let $Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N)$ be an accepted* $\mathsf{YoungSymmetrizer}^{\uparrow}$ *carrier. Any downstream route that reads the Young-symmetrizer idempotent behavior from Y must factor through the displayed finite chain*

$$P, D, T \longrightarrow R, C \longrightarrow S, A \longrightarrow E \longrightarrow M \longrightarrow V, N.$$

The route may expose the row symmetrizer, the column antisymmetrizer, and the tensor/matrix action as a finite certificate surface; it does not claim normalized primitive idempotence, irreducible Specht image, Schur functor exactness, or a complete representation-ring decomposition.

Proof. Start with the displayed source chain. The partition and tableau rows are the only source of row and column incidence, so every stabilizer read reaches R or C through P, D, T.

The symmetrizer and antisymmetrizer rows are formal finite sums over those stabilizers. Thus the composite row E is available only after S and A have been displayed.

The action and consumer boundary are then forced by M and V. The packet contains no row for normalized scalar inversion, quotient module projection, or semisimplicity, so the idempotent behavior remains a finite route claim over the displayed action rather than a host-level representation-theoretic closure theorem. □

Falsifiable prediction: Every future $\mathsf{YoungSymmetrizer}^{\uparrow}$ Lean carrier can be audited by counting exactly one partition shape row, one tableau row, two stabilizer ledgers, two formal-sum rows, one composite row, one tensor/matrix action row, and one local $\mathsf{NameCert}$ row before any $\mathsf{RepresentationRing}^{\uparrow}$ consumer is admitted.

Independence witness: partition, charactertheory, representationring, tensorproduct, matrix: YoungSymmetrizerUp is not bijective to any listed sibling because it jointly carries tableau incidence, row and column stabilizer ledgers, row symmetrization, column antisymmetrization, a composite tensor/matrix action, consumer confinement, and local NameCert provenance.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}\uparrow}$.

Constructive story: A $\mathsf{YoungSymmetrizer}^{\uparrow}$ packet is generated from one partition shape row, one Young diagram row, one tableau incidence row, row and column stabilizer ledgers, row-symmetrizer and column-antisymmetrizer formal-sum rows, one composite action row, displayed tensor/matrix action support, consumer confinement, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5560.1, Definition 5560.2, and Definition 5560.3 bind the finite tableau, stabilizer, formal-sum, action, and consumer surface for the Young-symmetrizer packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove primitive idempotence, irreducible Specht images, Schur functor exactness, semisimplicity, a complete character formula, host group-algebra equality, Lean verification, or a bridge-checked public schema.

Upgrade path: Obligation closure requires checked carrier admission, tableau incidence exposure, stabilizer-ledger exactness, finite formal-sum exactness, tensor/matrix action confinement, consumer non-escape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.YoungSymmetrizerUp.

Chapter 4751

A Concrete Naming Certificate for $\mathsf{MetaCICSRKernel}^{\uparrow}$

The $\mathsf{MetaCICSRKernel}^{\uparrow}$ packet realizes the subject-reduction discharge branch of chapter 9374 as one finite kernel carrier. It sits between the open-phase audit of $\mathsf{MetaCICCriticalPath}^{\uparrow}$ in chapter 765, the four-face obligation packet $\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}$ in chapter 712, the route socket $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ in chapter 719, the closed-substitution boundary $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ in chapter 527, and the candidate/endpoint frontier $\mathsf{MetaCICNormalizationFrontier}$ in chapter 892. The packet gives later formalization a concrete finite place to discharge typing-preservation routes without replacing the four discharge faces by a parameter echo.

Theorem 4751.1 (MetacicSrKernelUp carrier)**.** *A* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M).$$

Here C is the $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source ledger; O is the* $\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}$ *four-face obligation row; S is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *route row; B, A, L, P are respectively the beta-substitution, application-argument, lambda-domain, and Pi-domain discharge rows consumed through O and S; T is the* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *closed-term projection row; N is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *candidate/endpoint boundary row; F is the finite subject-reduction handoff row that exposes exactly the four discharge faces before any type-preservation consumer reads the route; H is the componentwise* hsame *transport row; R is displayed* Cont *replay for the kernel route; G is the* Pkg *provenance over the named MetaCIC packets,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*; and M is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}^{\uparrow}}$ *row.*

Proof. Read the carrier as a finite route ledger. The coordinates C, O, S, T, N locate the existing MetaCIC boundaries, while B, A, L, P are exactly the four discharge rows already named by the obligation and socket packets.

The row F is not a fifth discharge hypothesis. It is the finite handoff that requires a consumer to see all four displayed faces before any subject-reduction read is exported.

The structural rows H, R, G, M only transport, replay, account for, and name the same finite kernel. Thus the carrier is a finite NameCert packet rather than a parametric schema over fresh carriers. □

Theorem 4751.2 (MetacicSrKernelUp NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *carrier $K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}^{\uparrow}}$ *surface consists exactly of the following rows:*

- *critical-path admission through C;*
- *obligation admission through O, with beta-substitution, application-argument, lambda-domain, and Pi-domain faces exposed as B, A, L, P;*
- *socket compatibility through S, requiring the same four faces and forbidding an erased closed-subject-reduction row;*
- *closed-term boundary compatibility through T;*
- *normalization-frontier compatibility through N;*
- *subject-reduction handoff through F, which admits a consumer only after B, A, L, P are all displayed;*
- *transport, replay, provenance, and local naming through H, R, G, M.*

No obligation row in this surface may be replaced by a universal carrier parameter, a host definitional-equality principle, a quotient reduction equality, or a hidden closed-consistency theorem.

Proof. The obligation surface is obtained by projecting the coordinates of K. The packet imports the four subject-reduction faces from O and S, so the local naming row cannot omit one face while still claiming to expose the discharge surface.

The boundary rows T and N constrain where closed substitutions and normalization-frontier reads may occur. They do not prove subject reduction by themselves.

The handoff row F is therefore the only public route from the four faces to a subject-reduction consumer. Since H, R, G, M only preserve the displayed route, the packet has no coordinate where an ambient closed theorem or parameter-echo schema can enter. □

Theorem 4751.3 (MetacicSrKernelUp discharge route)**.** *Every accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *packet routes subject-reduction discharge through the finite chain*

$$C \longrightarrow O, S \longrightarrow B, A, L, P \longrightarrow T, N \longrightarrow F \longrightarrow M.$$

A type-preservation consumer may read the route only after the four discharge faces have been displayed and transported through H, R, G, M. The packet does not export closed subject reduction for the present substrate, unrestricted dependent-codomain preservation, erased substitution stability, global normalization, or closed consistency.

Proof. Start at the critical-path ledger C. The next admissible rows are the obligation and socket rows O, S, because they name the discharge faces and the obstruction boundary.

Project B, A, L, P from those rows. The closed-term and normalization-frontier coordinates T, N then constrain the route to the closed-substitution and candidate/endpoint surfaces already present in the neighboring packets.

Finally F packages the visible discharge for a consumer, while H, R, G, M preserve and name the route. No row in the carrier provides a replacement theorem, so the discharge remains a finite kernel route. □

Theorem 4751.4 (MetacicSrKernelUp closed-substitution discharge)**.** *Let $K = (C, O, S, B, A, L, P, T, N, F, H, R,$ be an accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *packet. A subject-reduction consumer may use the beta, application-argument, lambda-domain, and Pi-domain discharge rows B, A, L, P only through the same closed-term projection row T and normalization-frontier row N before the finite handoff F is read. Replacing T by a host substitution theorem, quotient reduction equality, or hidden closed-consistency theorem leaves the accepted carrier.*

Proof. Project the carrier of Definition 5561.1. The four discharge rows B, A, L, P are displayed only after the critical-path, obligation, and socket rows C, O, S.

Apply Definition 5561.2. It exposes the four discharge faces through the same obligation and socket surface and records compatibility with the closed-term boundary T and normalization-frontier boundary N.

Apply Definition 5561.3. The route forces

$$C \longrightarrow O, S \longrightarrow B, A, L, P \longrightarrow T, N \longrightarrow F \longrightarrow M$$

before a subject-reduction consumer can read the finite handoff.

The structural rows H, R, G, M transport, replay, source, and name that route. Since the carrier has no coordinate for a host substitution theorem, quotient reduction equality, or hidden closed-consistency theorem, replacing T by any such object exits the accepted packet. □

Theorem 4751.5 (MetacicSrKernelUp four-face totality)**.** *Let $K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M)$ be an accepted* MetaCICSRKernel$^{\uparrow}$ *packet. The finite subject-reduction handoff row F is readable only after the beta-substitution row B, application-argument row A, lambda-domain row L, and Pi-domain row P have all been displayed through the shared obligation row O and route-socket row S. No packet that omits any one of B, A, L, P exports the subject-reduction handoff of K, and no three-face route can bypass the missing face through the closed-term boundary T, normalization-frontier boundary N, or handoff row F.*

Proof. Project the accepted carrier of Definition 5561.1. The only discharge-face coordinates available to a consumer are B, A, L, P, and the only rows that display them to the kernel route are O and S.

Apply Definition 5561.2. It lists the four faces as one obligation surface: beta-substitution, application-argument, lambda-domain, and Pi-domain exposure must all be present before the local naming row admits the handoff.

Now apply Definition 5561.3. The route orders the carrier read through C, then O, S, then the four discharge faces, then the boundary rows T, N, and only then the handoff row F and local name row M.

Finally use Definition 5561.4. It prevents a consumer from replacing the closed-term boundary by a host substitution theorem, quotient reduction equality, or hidden consistency result. Since T, N, F are boundary and handoff rows rather than substitute discharge faces, a three-face packet has no coordinate from which to export the accepted handoff. □

Falsifiable prediction: The Lean follow-up for MetaCICSRKernel$^{\uparrow}$ must provide `Nontrivial MetacicSrKernelUp` and `FieldFaithful MetacicSrKernelUp` in `TasteGate.lean`, and the instance audit must expose exactly four discharge-face fields before any finite subject-reduction handoff is admitted.

Independence witness: metaciccriticalpath, metacic-subject-reduction-obligation, subjectreductiondischargesocket, closedtermsubstitutionboundary, metacicnormalizationfrontier: MetacicSrKernelUp is not bijective to any listed sibling because it simultaneously carries the critical-path ledger, four discharge faces, route socket, closed-substitution boundary, normalization-frontier boundary, finite handoff row, transport, replay, provenance, and local NameCert naming.

Closure status for NameCert$_{\text{MetaCICSRKernel}^{\uparrow}}$**.**

Constructive story: A MetaCICSRKernel$^{\uparrow}$ packet is generated from the MetaCIC critical-path ledger, a four-face subject-reduction obligation row, a discharge-socket row, beta-substitution, application-argument, lambda-domain, and

Pi-domain discharge rows, closed-substitution and normalization-frontier boundaries, a finite handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICSRKernel}\uparrow}$, seed)

Scope closed: Definition 5561.1, Definition 5561.2, and Definition 5561.3 bind the finite four-face subject-reduction kernel route and its handoff surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICSRKernel}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove closed subject reduction, unrestricted dependent-codomain preservation, erased substitution stability, global normalization, closed consistency, Lean verification, or a bridge-checked public schema.

Upgrade path: Obligation closure requires checked carrier admission, four-face discharge exposure, socket compatibility, closed-substitution boundary compatibility, normalization-frontier compatibility, handoff non-escape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.MetacicSrKernelUp.

Chapter 4752

A Concrete Naming Certificate for $\mathsf{HInfinityControl}^{\uparrow}$

Following Zames and the bounded-real robust-control development of Doyle, Glover, Khargonekar, and Francis, the $\mathsf{HInfinityControl}^{\uparrow}$ packet records the finite bounded-gain certificate surface used by robust-control consumers. It sits after the state-space carrier of chapter 329, matrix rows of chapter 49, vector-space rows of chapter 47, real gain rows of chapter 38, Riccati rows of chapter 307, finite-horizon control rows of chapter 296, and finite prediction ledgers of chapter 348. The packet names a finite robust-control ledger for disturbance-to-performance gain reads; it is not an analytic infinite-horizon optimality theorem.

Definition 4752.1 (HInfinityControl carrier)**.** *An* $\mathsf{HInfinityControl}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, Y, U, G, F, R, L, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StateSpaceModel}^{\uparrow}$ *row carrying the finite state-space presentation.*
2. *D is the disturbance-input row, recorded as a finite* $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *interface into the state update.*
3. *Y is the performance-output row, again read through finite* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *coordinates.*
4. *U is the controller row, exposing the finite controller state, input, and output ledgers.*
5. *G is the* $\mathsf{Real}^{\uparrow}$ *gain-bound row naming the candidate bounded-gain level.*
6. *F is the finite rollout ledger comparing disturbance energy and performance energy over displayed time windows.*
7. *R is the Riccati dependency row, readable only as a finite* $\mathsf{Riccati}^{\uparrow}$ *certificate surface.*
8. *L is the* $\mathsf{LQR}^{\uparrow}$ *dependency row for the quadratic-cost and dynamic-programming ledger used by the controller comparison.*
9. *M is the* $\mathsf{ModelPredictiveControl}^{\uparrow}$ *dependency row for finite rollout and constraint-window replay.*
10. *H records componentwise* hsame *transport for $S, D, Y, U, G, F, R, L, M$.*

11. *C records displayed* Cont *replay from the state-space rows through disturbance, performance, controller, gain, rollout, and dependency rows.*
12. *P is the* Pkg *provenance over* $\mathsf{HInfinityControl}^{\uparrow}$, $\mathsf{StateSpaceModel}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Riccati}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{ModelPredictiveControl}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HInfinityControl}^{\uparrow}}$ *row.*

The carrier stores no host trajectory space, no spectral factorization object, no infinite-horizon optimizer, no external robust-stability theorem, and no analytic transfer-function norm.

Lean4 ✓ `BEDC.Derived.HInfinityControlUp.HInfinityControlTasteGate single carrier alignment`

Theorem 4752.2 (HInfinityControl NameCert obligations)**.** *For every accepted* $\mathsf{HInfinityControl}^{\uparrow}$ *carrier* $K = (S, D, Y, U, G, F, R, L, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HInfinityControl}^{\uparrow}}$ *surface consists exactly of carrier admission, state-space exposure, disturbance-input exposure, performance-output exposure, controller exposure, gain-bound exposure, finite rollout comparison, Riccati dependency, LQR dependency, model-predictive-control dependency, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer that replaces one of these rows by a host trajectory space, spectral factorization theorem, external standard bridge, infinite-horizon optimality principle, or full robust-stability theorem leaves the accepted carrier.*

Proof. Project the carrier of Definition 4752.1. The displayed coordinates list the state-space, disturbance, performance, controller, gain, rollout, and dependency rows before the structural transport, replay, provenance, and naming rows.

The dependency surface is finite. Riccati, LQR, and model-predictive-control rows are read only through R, L, M, so no analytic solver or external robust-control theorem is available inside the packet.

The rows H, C, P, N preserve and name the same inventory. Since the carrier has no coordinate for a host trajectory space, spectral factorization, transfer-function norm, infinite-horizon optimizer, or robust-stability theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 4752.3 (HInfinityControl bounded-gain route)**.** *Let* $K = (S, D, Y, U, G, F, R, L, M, H, C, P, N)$ *be an accepted* $\mathsf{HInfinityControl}^{\uparrow}$ *packet. Every bounded-gain read factors through the finite route*

$$S \longrightarrow D, Y, U \longrightarrow G, F \longrightarrow R, L, M \longrightarrow H, C, P, N.$$

The read first displays the state-space row, then the disturbance-input, performance-output, and controller rows, then the real gain-bound and finite rollout ledger, and only then the Riccati, LQR, and model-predictive-control dependency rows. No host trajectory space, spectral factorization, external standard bridge, analytic infinite-horizon optimality theorem, or full robust-stability theorem can factor the same accepted read.

Proof. Start with the state-space row S. The disturbance, performance, and controller rows D, Y, U are meaningful only relative to that displayed finite state-space presentation.

The gain read is then bounded by the $\mathsf{Real}^{\uparrow}$ row G and the finite rollout ledger F. These rows compare displayed finite histories; they do not introduce a transfer-function norm or an ambient trajectory space.

Finally the dependency rows R, L, M explain how the local control ledger is replayed against Riccati, LQR, and model-predictive-control packets, while H, C, P, N transport, replay, source, and name the same finite route. Coordinate exhaustion rules out any hidden spectral factorization, infinite-horizon optimizer, standard bridge, or robust-stability theorem. □

Falsifiable prediction: An accepted $\mathsf{HInfinityControl}^{\uparrow}$ read is refuted if a bounded-gain conclusion is reached without displaying the StateSpaceModelUp row, disturbance-input row, performance-output row, controller row, RealUp gain-bound row, finite rollout ledger, RiccatiUp row, LQRUp row, ModelPredictiveControlUp row, transport, replay, provenance, and local naming rows.

Independence witness: statespacemodel, riccati, lqr, modelpredictivecontrol, matrix, vecspace, and real: none is bijective to the bounded-gain robust-control packet because only $\mathsf{HInfinityControl}^{\uparrow}$ simultaneously records disturbance input, performance output, controller, real gain bound, finite rollout ledger, control dependencies, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HInfinityControl}^{\uparrow}}$.

Constructive story: An $\mathsf{HInfinityControl}^{\uparrow}$ packet is generated from a StateSpaceModelUp row, disturbance-input row, performance-output row, controller row, RealUp gain-bound row, finite rollout ledger, RiccatiUp dependency row, LQRUp dependency row, ModelPredictiveControlUp dependency row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HInfinityControl}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4752.1, Definition 4752.2, and Definition 4752.3 seed the finite bounded-gain robust-control route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HInfinityControl}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim analytic infinite-horizon optimality, host trajectory spaces, spectral factorization, an external standard bridge, full robust-stability theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-gain route determinacy, dependency-row exposure, non-escape for analytic robust-control primitives, transport, replay, provenance, and local naming for BEDC.Derived.HInfinityControlUp.

$\mathsf{RealApartnessField}^{\uparrow}$ reads its inverse guard through chapter 38, chapter 301, and chapter 291 before the child routes expose apartness-field comparison rows.

Chapter 4753

A Concrete Naming Certificate for $\mathsf{RealApartnessField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealApartnessField}^{\uparrow}$ packet records the constructive field-facing surface for real names whose inverse operation is guarded by displayed apartness evidence. It sits after the terminal real-name seal of chapter 38, the positive apartness route of chapter 1881, the located order boundary of $\mathsf{LocatedRealOrder}^{\uparrow}$, the algebraic source rows of chapter 45, and the dyadic tolerance ledger of chapter 291. The packet joins $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{LocatedRealOrder}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ as one finite NameCert surface; it does not import trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every apartness-field read passes through the $\mathsf{Real}^{\uparrow}$ seal and real readback rows of chapter 38, the apartness/order bridge of chapter 1905, the specialization order surface of chapter 1378, and the field operation rows of chapter 45 before it exports an apartness algebra surface.

Completion-facing uses of an apartness-field inverse keep the distance-preserving read compatible with the isometric embedding boundary of chapter 439 before any completion consumer treats the $\mathsf{Real}^{\uparrow}$ seal as a field-facing endpoint. This is a dependency on the displayed $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ distance rows, not on host function equality or quotient completion.

Theorem 4753.1 (Real apartness field located source route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or order-facing read first consumes the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *route of chapter 1905 before the* $\mathsf{Field}^{\uparrow}$ *inverse branch may be exposed. In carrier notation, the displayed packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

routes the real seal R, apartness row A, located-order row O, field row K, dyadic ledger D, transport H, replay C, provenance P, and local naming row N through the apartness/order bridge before any consumer reads an inverse or order-facing field operation.

Proof. Project the accepted carrier. The inverse branch is stored in the field row K, but the carrier definition also displays the real seal R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N in the same finite packet.

Read the bridge route cited by the introduction. The $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ chapter supplies the sibling route that ties apartness evidence to located order data before an order-facing consumer can use the real field surface.

Now compose the existing NameCert obligations with inverse-guard stability. The obligation rows require inverse admission to consume A and D, while located order compatibility consumes

O over the same real seal R. The transport and replay rows H, C, P, N preserve that source route, so the field row K exposes its inverse branch only after the apartness/order bridge has already been read. □

Definition 4753.2 (Real apartness field carrier)**.** *A* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the real-name surface exposed to field consumers.*
2. *A is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive-apartness row used to certify that a denominator is separated from zero before an inverse read is accepted.*
3. *O is the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *located order row that routes order-facing comparisons through displayed finite windows and dyadic cells.*
4. *K is the* $\mathsf{Field}^{\uparrow}$ *algebraic operation row for addition, multiplication, negation, zero, one, and the inverse request boundary.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and positivity ledger controlling arithmetic error, gap size, and reciprocal budget.*
6. *H records componentwise* hsame *transport for the real seal, apartness row, located order row, field row, dyadic ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each field-facing request through R, A, O, K, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealApartnessField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{LocatedRealOrder}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one constructive real field packet whose inverse reads are admitted only through apartness evidence. It has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 4753.3 (Real apartness field StreamName dependency)**.** *Every apartness-field comparison over* $\mathsf{Real}^{\uparrow}$ *must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126 and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 before the* $\mathsf{Real}^{\uparrow}$ *apartness and order-field row of chapter 38 is consumed. Thus a comparison read of an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier factors through finite stream observations, regular-rational tolerance data, the terminal Real seal, and only then the apartness, located-order, and field rows A, O, K.*

Proof. Project the carrier of Definition 5562.2. The real seal row R is the only Real-facing entry, and the apartness, order, and field rows A, O, K are downstream comparison rows rather than independent host-real operations.

Use the Real chapter route cited by the introduction. A $\mathsf{Real}^{\uparrow}$ seal is read from finite name data, so the comparison must first expose the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback that supply the visible Real name.

Finally read the apartness-field rows. The positive-apartness row A, located order row O, field row K, and dyadic ledger D consume that already displayed Real readback, while H, C, P, N preserve and name the same finite route. Hence no comparison can bypass StreamName windows or regular-rational readback before using the Real apartness/order field surface. □

Theorem 4753.4 (Real apartness field NameCert obligations)**.** *For every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier*

$$F = (R, A, O, K, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real seal, positive apartness row, located order row, field operation row, dyadic ledger, transport, replay, provenance, and local naming row;*
2. *algebraic projection, requiring every addition, multiplication, negation, zero, and one read to pass through the displayed* $\mathsf{Field}^{\uparrow}$ *row* K *and the real seal row* R*;*
3. *inverse admission, requiring an inverse read to consume the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row* A *and the positive dyadic gap data in* D *before the field row* K *exports the reciprocal branch;*
4. *located order compatibility, requiring the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *row* O *to read the same real-name and dyadic comparison data used by the apartness row;*
5. *transport and replay, requiring* H *and* C *to preserve the real, apartness, order, field, dyadic, provenance, and local naming rows together;*
6. *ledger exhaustion, excluding trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No real apartness field obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5562.2. The accepted packet displays exactly the real seal, apartness row, located order row, field row, dyadic ledger, and structural rows.

The algebraic projection reads the $\mathsf{Field}^{\uparrow}$ row K only over the displayed real seal R. Thus the operation surface is the carried field row, not host real arithmetic or a quotient-real operation.

The inverse projection first reads A and D. The apartness row supplies positive separation from zero, and the dyadic row supplies the gap and reciprocal budget; without those rows the inverse request remains at the boundary and cannot be consumed as a field operation.

Finally, O, H, C, P, N keep comparison, transport, replay, provenance, and naming inside the same finite packet. Since none of the carrier coordinates stores a trichotomy decision, quotient equality, choice schedule, selected representative, completeness witness, Lean theorem, or standard bridge, none becomes a local NameCert obligation. □

Theorem 4753.5 (Real apartness field located-order inverse guard)**.** *Let*

$$F = (R, A, O, K, D, H, C, P, N)$$

be an accepted $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. An inverse read from the field row* K *is accepted only after the located-order row* O*, positive apartness row* A*, and dyadic reciprocal-budget row* D *have been displayed over the same* $\mathsf{Real}^{\uparrow}$ *seal* R*. The guard is a finite L10 readback through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedRealOrder}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*; it does not use trichotomy, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5562.2. The inverse request is stored in K, but the same packet also displays the real seal R, positive apartness row A, located-order row O, dyadic ledger D, and structural rows H, C, P, N.

Apply the NameCert obligations of Definition 5562.4. The inverse admission clause exposes A and D before the reciprocal branch is exported, while the located-order compatibility clause requires O to read the same real-name and dyadic comparison data.

Use the StreamName dependency theorem Definition 5562.3. It forces the $\mathsf{Real}^{\uparrow}$ seal R to be read back from finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ tolerance data before the apartness, order, and field rows are consumed.

Finally read the located real route through O. Since H, C, P, N only transport, replay, source, and name the same finite packet, they cannot add trichotomy, quotient real equality, a selected representative, countable choice, or ambient completeness. Therefore an accepted inverse read is exactly a field-row read guarded by O, A, D over the same displayed Real seal R. □

Theorem 4753.6 (Real apartness field inverse-guard stability)**.** *Let*

$$F = (R, A, O, K, D, H, C, P, N)$$

be an accepted $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. An inverse read from the field row K may be exported to a downstream consumer only when the displayed apartness guard in A survives componentwise* hsame *transport through the* $\mathsf{Real}^{\uparrow}$ *seal R, the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A, the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary O, the* $\mathsf{Field}^{\uparrow}$ *operation row K, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and reciprocal-budget ledger D, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N.*

Consequently, the inverse branch is stable under the carried transport rows exactly when its apartness evidence remains displayed in the same finite packet. It is not stable by trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The inverse request sits in the field row K, but the NameCert obligations require the request to consume the positive-apartness row A and the positive dyadic gap data in D before the reciprocal branch is exported.

Transport preserves the guard componentwise. The real seal R, apartness row A, located order boundary O, field row K, and dyadic ledger D are carried through the same hsame route, while C, P, N replay, source, and name that route.

Therefore the inverse operation remains downstream of the apartness guard after transport. If the displayed guard is absent, the transported packet has no row from which an inverse read can be exported.

The carrier stores no trichotomy decision, quotient equality, selected representative, choice schedule, host inverse, or completeness witness. Stability of the inverse branch is exactly stability of the displayed apartness guard inside the finite $\mathsf{RealApartnessField}^{\uparrow}$ packet. □

Theorem 4753.7 (Real apartness field reciprocal-budget locality)**.** *Let $F = (R, A, O, K, D, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. Every inverse read exported from the field row K must first expose the positive apartness row A and the dyadic reciprocal-budget ledger D. The reciprocal route is local to these displayed rows and their componentwise transport, not to trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The inverse branch is stored in K, but its admission rows are the positive apartness row A and the dyadic ledger D.

Use inverse-guard stability from Definition 5562.6. That theorem keeps the inverse branch downstream of the displayed apartness guard under the same componentwise hsame transport.

The dyadic ledger supplies the reciprocal budget attached to that guard. Rows H, C, P, N preserve, replay, source, and name the same finite route, so an inverse read cannot be exported unless both A and D have already been displayed. □

Theorem 4753.8 (Real apartness field multiplicative inverse boundary)**.** *Let $F = (R, A, O, K, D, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. A multiplicative inverse read from K is available to a downstream real-field consumer only after A displays positive apartness of the denominator from zero and D displays the corresponding positive dyadic gap and reciprocal budget. Therefore the inverse operation is controlled by constructive apartness evidence rather than by trichotomy, quotient equality, selected representatives, or choice.*

Proof. Begin with the accepted carrier. The inverse request is a branch of the field row K, but the carrier definition makes that branch depend on the same apartness and dyadic rows named by A and D.

Read the apartness row. The row A supplies a positive separation witness for the denominator source, so the consumer does not ask whether the denominator is equal to zero by trichotomy or quotient equality.

Read the dyadic ledger next. The row D records the positive gap and reciprocal budget used by the inverse branch, so the reciprocal calculation is bounded by displayed finite arithmetic rather than by an unrecorded host inverse.

The remaining rows only preserve the route. Transport, replay, provenance, and local naming cannot add a selected representative, choice operator, or ambient completeness datum, so the inverse boundary is exactly the apartness-controlled field read stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$.

Constructive story: A $\mathsf{RealApartnessField}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{CauchyRealApartness}^{\uparrow}$ positive-apartness row, a $\mathsf{LocatedRealOrder}^{\uparrow}$ located order row, a $\mathsf{Field}^{\uparrow}$ operation row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and reciprocal-budget ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5562.2, Definition 5562.4, Definition 5562.6, and Definition 5562.8 seed the constructive real field boundary whose inverse branch is guarded by positive apartness evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, algebraic projection, inverse admission, located order compatibility, transport, replay, and ledger exhaustion for BEDC.Derived.RealApartnessFieldUp.

Theorem 4753.9 (Real apartness field regular-Cauchy apartness/order dependency)**.** *Every inverse-guard read or apartness-field read of an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier must pass through the regular-Cauchy apartness/order route supplied by chapter 1710 and the positive-apartness route of chapter 1881 before the* $\mathsf{Field}^{\uparrow}$ *inverse branch is consumed. In carrier notation, a consumer must display the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A, the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary O, the* $\mathsf{Field}^{\uparrow}$ *operation row K, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and reciprocal-budget ledger D inside the same packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

before it treats an inverse as a field operation. The route adds no trichotomy, quotient equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with inverse-guard stability, Definition 5562.6. It requires an inverse read to keep the displayed apartness guard in A stable through the real seal, located order boundary, field row, dyadic ledger, replay, provenance, and local naming rows.

Next read the multiplicative inverse boundary, Definition 5562.8. The inverse branch in K is available only after A displays positive apartness from zero and D displays the corresponding positive dyadic gap and reciprocal budget.

The regular-Cauchy apartness/order chapter supplies the regular source of that positive comparison: Definition 1710.3 routes every strict comparison or apartness read through its displayed budget, finite window, positive dyadic boundary, regular readback, and Real seal. The Cauchy real apartness chapter supplies the positive-apartness handoff through Definition 1881.3 and Definition 1881.4.

Combining these routes forces the RealApartnessField inverse branch to consume regular-Cauchy apartness/order evidence and positive Cauchy real apartness before it is read as a field inverse. Since each cited route refuses trichotomy, quotient real equality, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness, the dependency adds none of those principles to the field packet. □

Every $\mathsf{RealApartnessField}^{\uparrow}$ inverse read consumes the apartness-order bridge of chapter 1905 before the field carrier of chapter 45 can export the reciprocal branch. The dyadic tolerance row inherited by the same route prevents trichotomy or quotient-real equality from entering the packet, so the field operation is read only after constructive apartness-order evidence has been displayed.

Theorem 4753.10 (Real apartness field dependency route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or apartness-field read consumes the regular-Cauchy source rows, the* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *route of chapter 1710, the located-real bridge through chapter 1933, and the* $\mathsf{RealLocatedOrder}^{\uparrow}$ *row of chapter 3640 before it exports field-level apartness. In carrier notation, for*

$$F = (R, A, O, K, D, H, C, P, N),$$

the regular source behind R, the positive apartness row A, the located-order row O, the field row K, the dyadic ledger D, transport H, replay C, provenance P, and local naming N are consumed in that order as one displayed dependency route.

Proof. Project the carrier of Definition 5562.2. The packet has no field-level apartness coordinate outside $R, A, O, K, D, H, C, P, N$.

Read the regular-Cauchy source first. The existing source route of Definition 5562.14 places the $\mathsf{RegSeqRat}^{\uparrow}$ source and the terminal $\mathsf{Real}^{\uparrow}$ seal before any field row K is consumed.

Next read the apartness and located rows. The regular-Cauchy apartness/order dependency of Definition 5562.9 supplies the $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ route for A, while Definition 5562.12 supplies the located-real bridge and Definition 5562.5 keeps the $\mathsf{RealLocatedOrder}^{\uparrow}$ row visible over the same real seal.

Finally compose with the NameCert obligations. The algebraic projection and inverse-admission rows of Definition 5562.4 allow K to export field-level apartness only after A, O, D have been displayed, and H, C, P, N preserve the same route. Thus the field-level apartness export is downstream of the regular source, regular-Cauchy apartness/order, located-real bridge, and real located-order rows, with no host equality, trichotomy, selected representative, choice schedule, or ambient completeness inserted. □

Theorem 4753.11 (Real apartness field Real/order forward link)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *packet hands its apartness-field row to* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *consumers only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *seal,* hsame, Cont, Pkg, *and* NameCert *rows. In carrier notation, a read of*

$$F = (R, A, O, K, D, H, C, P, N)$$

must first expose the real seal route of chapter 38 and the apartness/order bridge of chapter 1905 before the field row K can export an inverse or order-facing operation.

Proof. Project the accepted carrier of Definition 5562.2. The field row K is not stored as host real arithmetic; it is carried together with the real seal R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N.

Read the Real route first. The chapter introduction and StreamName dependency theorem route every Real-facing comparison through finite regular-rational and stream-window data before R is consumed, so the field consumer receives a displayed $\mathsf{Real}^{\uparrow}$ seal rather than a selected representative or quotient real.

Now apply the apartness/order route. The bridge cited in chapter 1905 consumes the apartness row A, located-order row O, and dyadic ledger D before any inverse or order-facing field operation is read.

Finally H, C, P, N preserve the same route under componentwise transport, replay, provenance, and local naming. Thus every forward handoff to Real and apartness-order consumers factors through the displayed rows and adds no trichotomy, quotient real equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4753.12 (Real apartness field located real route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or order-facing read must pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *source route of chapter 1933 and the apartness-order bridge of chapter 1905 before the* $\mathsf{Field}^{\uparrow}$ *reciprocal branch is exposed. In carrier notation, for*

$$F = (R, A, O, K, D, H, C, P, N),$$

the real seal row R, apartness row A, located-order row O, field row K, dyadic ledger D, componentwise transport H, displayed replay C, provenance P, and local naming row N are consumed in the same finite route. The route exports no trichotomy, quotient real equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the accepted carrier from Definition 5562.2. The reciprocal branch is stored in the $\mathsf{Field}^{\uparrow}$ row K, while R, A, O, D, H, C, P, N remain visible as separate rows of the same finite packet.

Read the located-real source route first. The $\mathsf{LocatedReal}^{\uparrow}$ chapter supplies the finite comparison boundary consumed before the terminal real seal is used, so the field consumer receives located real evidence rather than a quotient representative or host real value.

Next read the apartness-order bridge. The bridge of chapter 1905 ties the apartness row A to the located-order row O over the same real seal R, and the dyadic ledger D records the reciprocal budget attached to that positive evidence.

Finally replay the consumer through H, C, P, N. Since those rows only transport, replay, source, and name the already displayed route, they cannot add trichotomy, quotient real equality, selected representatives, countable choice, or ambient completeness. Thus every inverse or order-facing read reaches the $\mathsf{Field}^{\uparrow}$ branch only after the located-real and apartness-order routes have both been displayed. □

Every $\mathsf{RealApartnessField}^{\uparrow}$ apartness read is field-facing only after it has passed through the L10 completion lattice: the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RealAlgOrder}^{\uparrow}$ comparison rows of chapter 766, and the located comparison surface of chapter 1730. Thus the packet is consumed as a classifier over displayed regular-rational, real-seal, algebraic-order, and located-comparison evidence, not as a detached field oracle. The $\mathsf{RealApartnessField}^{\uparrow}$ lattice consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the located real bridge of chapter 1933, and the constructive order-field route through chapter 3640, chapter 1905, and chapter 6850. Its field-facing apartness reads therefore enter through displayed real names, located comparison, and order transport rows before any algebraic operation is used. Every

$\mathsf{RealApartnessField}^{\uparrow}$ carrier reads its terminal real-name dependency through the finite L10 completion route, not through an ambient field of host reals. The route passes through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126.

Proposition 4753.13 (Real apartness field finite completion dependency route)**.** *For every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier* $F = (R, A, O, K, D, H, C, P, N)$*, any field-facing read of* R, A, O, K*, or* D *first exposes the finite-window* $\mathsf{StreamName}^{\uparrow}$ *rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal named above. Hence the apartness-field carrier consumes* $\mathsf{Real}^{\uparrow}$ *only through that displayed finite L10 dependency lattice, with* H, C, P, N *preserving transport, replay, provenance, and local naming for the same route.*

Proof. Project the carrier of Definition 5562.2. The only Real-facing row is R, and the apartness, located-order, field, and dyadic rows A, O, K, D are stored as downstream rows of the same finite packet.

Apply the StreamName and regular-rational dependency routes already present in the chapter. They force a comparison or inverse read to expose finite windows and regular-rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally read the dyadic ledger. The tolerance and reciprocal-budget information in D is displayed before the field-facing row K is exported, while H, C, P, N carry only the same finite dependency route. Therefore no read imports quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4753.14 (Real apartness field regular-Cauchy source route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or apartness read first consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *source of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the located and order sibling rows already named by this chapter before the* $\mathsf{Field}^{\uparrow}$ *inverse data are exposed. In carrier notation, a read of*

$$F = (R, A, O, K, D, H, C, P, N)$$

passes through the regular-rational source behind R*, the terminal real seal* R*, the apartness row* A*, located-order row* O*, dyadic ledger* D*, transport* H*, replay* C*, provenance* P*, and local naming row* N *before the field row* K *can export an inverse branch.*

Proof. Project the accepted carrier of Definition 5562.2. The field row K is stored together with the real seal row R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N.

Use the StreamName dependency theorem Definition 5562.3. It forces every comparison-facing read to expose the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 before the terminal $\mathsf{Real}^{\uparrow}$ route of chapter 38 is consumed.

Now apply the located and order child routes already carried by the chapter: Definition 5562.1, Definition 5562.12, and Definition 5562.11. These routes place the located-real, apartness-order, and Real/order dependencies before any field-facing inverse read is exported.

Finally combine the NameCert obligations with inverse-guard stability. Theorems 5562.4 and 5562.6 require the apartness guard and dyadic reciprocal budget to be displayed before K is consumed. Thus the inverse data are downstream of the RegSeqRat source, Real terminal seal, and located/order sibling rows, and the route adds no trichotomy, quotient real equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The RealApartnessField$^{\uparrow}$ chapter reads its field-facing apartness rows through the regular-Cauchy apartness lattice before exporting any arithmetic surface. In particular, the route first exposes chapter 1710, then the located-real surface of chapter 1933, and then the located order rows of chapter 3640. Only after these three sibling dependencies are visible may the chapter present the RealApartnessField$^{\uparrow}$ field-facing apartness rows; the route exports no host real equality, detached order completion, selected located witness, or ambient field law outside the displayed BEDC dossier lattice. The RealApartnessField$^{\uparrow}$ lattice consumes the regular Cauchy criterion surface of chapter 1611, the located real order route of chapter 8337, and the terminal Real seal of chapter 38 before it exports apartness-field reads. Thus an inverse or comparison read from the field-facing row is routed through displayed criterion checks, order evidence, and Real sealing, rather than through a detached field oracle.

Proposition 4753.15 (Real apartness field regular Cauchy criterion route)**.** *Every accepted* RealApartnessField$^{\uparrow}$ *packet that exports an apartness-field read must first expose the* RegularCauchyCriterion$^{\uparrow}$ *criterion dependency, the* LocatedRealOrder$^{\uparrow}$ *order dependency, and the* Real$^{\uparrow}$ *seal dependency named by its carrier rows. The exported read is therefore a finite L10 route through regular Cauchy criterion data, located order evidence, terminal Real sealing,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Project the accepted RealApartnessField$^{\uparrow}$ carrier. Its apartness-field row is downstream from the displayed Real seal, located-order row, dyadic tolerance ledger, transport, replay, provenance, and local naming rows already named by the chapter.

The regular Cauchy criterion dependency supplies the finite criterion checks consumed before the Real seal and order-facing rows are read. Since the carrier has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, or ambient Real completeness, the exported apartness-field read remains inside the displayed finite route. □

Theorem 4753.16 (Real apartness field product-topology dependency)**.** *Every product-coordinate apartness-field read of an accepted* RealApartnessField$^{\uparrow}$ *packet must expose the finite* ProductTopology$^{\uparrow}$ *product window before the inverse, order, and field rows of the apartness-field carrier are consumed. The route first reads the* Real$^{\uparrow}$ *seal,* RegSeqRat$^{\uparrow}$ *readback,* StreamName$^{\uparrow}$ *finite windows,* Field$^{\uparrow}$ *operation row,* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and* RealApartnessOrderBridge$^{\uparrow}$ *apartness-order bridge, and only then permits the product-coordinate consumer to use the* RealApartnessField$^{\uparrow}$ *inverse or order-facing field surface.*

Proof. Project the accepted RealApartnessField$^{\uparrow}$ carrier through Definition 5562.2. Its inverse and order-facing reads are stored in the displayed real seal, apartness, located-order, field, dyadic, transport, replay, provenance, and local naming rows, so a product-coordinate consumer has no separate coordinate from which to read a field operation.

Read the product surface through chapter 2000. The finite ProductTopology$^{\uparrow}$ window supplies the coordinate route before either coordinate can be consumed as a real apartness-field input.

Use the source rows already required by the chapter: the StreamName$^{\uparrow}$ windows and RegSeqRat$^{\uparrow}$ readback feed the Real$^{\uparrow}$ seal, the DyadicRatCore$^{\uparrow}$ ledger supplies tolerance and positivity data, the Field$^{\uparrow}$ row carries the operation boundary, and the RealApartnessOrderBridge$^{\uparrow}$ route ties apartness evidence to order-facing comparison over the same real seal. The product-coordinate read therefore reaches the inverse or order-facing field rows only after the product-topology dependency has been displayed. □

Theorem 4753.17 (Real apartness field Real–RegSeqRat–Dyadic route)**.** *Every* RealApartnessField$^{\uparrow}$ *apartness-field comparison read exposes the terminal* Real$^{\uparrow}$ *seal of chapter 38, the* RegSeqRat$^{\uparrow}$

regular-rational readback of chapter 301, and the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291 before the field-facing comparison row is exported. The route is a displayed dependency through real sealing, regular-rational readback, and dyadic tolerance control; it does not add trichotomy, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5562.2. Its comparison-facing data are stored with the real seal row, apartness row, located-order row, dyadic ledger, transport, replay, provenance, and local naming rows.

Read the $\mathsf{RegSeqRat}^{\uparrow}$ source before the terminal $\mathsf{Real}^{\uparrow}$ seal. The existing regular-Cauchy source route forces the comparison to pass through regular-rational readback rather than through an untracked real representative.

Then expose the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. The apartness guard and reciprocal budget are consumed only with the displayed dyadic tolerance data, so the field-facing row is downstream of the Real, RegSeqRat, and DyadicRatCore dependencies named above. □

Theorem 4753.18 (Real apartness field inverse-guard dependency route)**.** *Every inverse-guard export from an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier factors through the* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive-apartness row,* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary row, and* $\mathsf{Field}^{\uparrow}$ *operation row before a field inverse is public. The route composes the Real–RegSeqRat–Dyadic dependency with inverse-guard stability and the regular-Cauchy apartness/order dependency, and it refuses trichotomy, quotient equality, selected representatives, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 5562.17. It places every field-facing comparison behind the terminal $\mathsf{Real}^{\uparrow}$ seal, the regular rational readback, and the dyadic tolerance ledger.

Next apply the inverse-guard stability surface in the parent chapter. The inverse row is public only when the positive-apartness guard remains attached to the displayed carrier, so the guard cannot be replaced by quotient equality or by a selected representative.

Finally read the regular-Cauchy apartness/order dependency. The positive-apartness row is checked together with the located-order boundary before the $\mathsf{Field}^{\uparrow}$ operation row exposes an inverse. Hence the inverse export remains a finite route through Real, RegSeqRat, DyadicRatCore, CauchyRealApartness, LocatedRealOrder, and Field data, with trichotomy, countable choice, quotient equality, selected representatives, and ambient Real completeness outside the packet. □

Chapter 4754

A Concrete Naming Certificate for $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy convergence in constructive real analysis, the $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ packet records the finite threshold schedule that lets a regular rational name answer downstream Real and Completion precision requests without appealing to a quotient of streams. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational source of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion route a named finite modulus surface; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, trichotomy, countable choice, or quotient-stream equality.

Definition 4754.1 (Modulus of regular convergence carrier)**.** *A* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, Q, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational reads are being controlled.*
2. *Q is the modulus row, recording a displayed tolerance request and the finite index threshold from which the regular convergence comparison is certified.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source reads inspected by Q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the dyadic radius, comparison budget, and readback precision used by the threshold row.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, Q, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the source, modulus, window, tolerance, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through R, Q, W, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ModulusOfRegularConvergenceUp.ModulusOfRegularConvergenceTasteGate single carrier alignment`

Theorem 4754.2 (Modulus of regular convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *carrier* $M = (R, Q, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}}$ *surface consists exactly of carrier admission, regular-rational source exposure, explicit threshold exactness, finite stream-window coverage, dyadic tolerance control, terminal real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. Every precision-facing read must expose*

$$R, Q, W, D \longrightarrow E \longrightarrow H, C, P, N.$$

No obligation is discharged by quotienting two streams, selecting a limit, invoking countable choice, importing trichotomy, or using ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 5563.1. The packet first displays the regular rational source R, the threshold row Q, the stream windows W, and the dyadic tolerance ledger D.

The threshold obligation is local to Q. The windows in W supply the finite source reads named by that threshold, while D supplies the dyadic comparison budget used to certify the requested regular convergence read.

The real-facing seal E is downstream of those four rows. Transport through H and replay through C preserve the same order, and P, N record provenance and local NameCert comparison without adding an outside coordinate.

Since quotient stream equality, selected limits, host real equality, countable choice, trichotomy, and ambient completeness do not occur among the carrier coordinates, every NameCert obligation is read through the displayed finite route. □

Theorem 4754.3 (Regular convergence finite-modulus handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *packet must read the regular rational source row, explicit modulus threshold, finite stream windows, and dyadic tolerance ledger before consuming the terminal real seal. The packet therefore exports a finite threshold route from regular rational data into Real-facing data, not a host completeness theorem or a quotient construction.*

Proof. Project an accepted carrier as $M = (R, Q, W, D, E, H, C, P, N)$. The only regular rational input is R, and the only threshold schedule is Q.

The row W exposes the finite observations named by that schedule. The row D supplies the dyadic tolerance comparison that turns those observations into a regular convergence read.

Only after R, Q, W, D are present may the consumer read E as the terminal $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport, replay, provenance, and local naming keep the same route visible to downstream consumers.

The carrier has no coordinate for selected limits, quotient stream equality, host real equality, countable choice, trichotomy, or ambient Real completeness. Thus every completion-facing handoff factors through the finite modulus rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}$ **.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5563.1, Definition 5563.2, and Definition 5563.3 seed the finite modulus route from regular rational source data through stream windows, dyadic tolerances, and a Real-facing seal.

Formal verification: FormalStatus(NameCert$_{\text{ModulusOfRegularConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, threshold exactness, stream-window coverage, dyadic tolerance control, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfRegularConvergenceUp.

Constructive story: A ModulusOfRegularConvergence$^{\uparrow}$ packet is generated from a RegSeqRat source row, explicit modulus threshold row, finite StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4755

A Concrete Naming Certificate for $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$

Following the classical separation between the Riemann Hypothesis as a fixed proposition and proof search as a metamathematical process, the $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ packet records the finite BEDC surface of the route-closure boundary described in chapter 9336. It sits over the RH condition-chain rows of that vision chapter, the halting distinction traces of chapter 492, the halting obstruction of chapter 490, the proof witness chain surface of chapter 1238, and the zeta-zero witness boundary of chapter 73. The packet names route, grounding-rank, decider-refusal, and NameCert rows; it does not decide RH, prove RH, refute RH, or assert independence for any chosen fragment.

Definition 4755.1 (RH-route halting boundary carrier)**.** *An* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, G, D, F, Z, H, C, P, N)$$

with the following displayed rows:

1. *R is the RH-route row, recording the displayed condition chain, rule tower, proof-output program, or route family being inspected.*
2. *G is the grounding-rank row, recording whether the inspected route has a displayed finite rank and, when it does, the finite top certificate and transports back to* RH*.*
3. *D is the decider-refusal row, recording that a proposed global or sufficiently expressive RH-route closure decider is a halting-boundary request rather than an internal BEDC classifier.*
4. *F is the fixed-statement row separating the proposition* RH *from route-search totality and from global proof-search classification.*
5. *Z is the zeta-zero witness interface row, exposing the critical-strip and critical-line witnesses that a genuine RH proof or counterexample datum would have to consume.*
6. *H records componentwise* hsame *transport for route, grounding-rank, refusal, fixed-statement, zeta-witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from route inspection through R, G, D, F, Z, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$, $\mathsf{HaltingDistinctionTrace}^{\uparrow}$, $\mathsf{HaltingObstruction}^{\uparrow}$, $\mathsf{ProofWitnessChain}^{\uparrow}$, $\mathsf{ZetaZeros}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, G, D, F, Z, H, C, P, N$. It has no coordinate for an RH proof, an RH counterexample, an RH-specific independence theorem, a global theorem-route closure decider, a proof-search oracle, or an unmarked halting predicate.

Lean4 (definition) `BEDC.Derived.RHRouteHaltingBoundaryUp.RHRouteHaltingBoundaryUp`

Theorem 4755.2 (RH-route halting boundary NameCert obligations)**.** *For every accepted* $\mathsf{RHRouteHaltingBoundar}$ *carrier $B = (R, G, D, F, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, route-row exposure, grounding-rank exactness, decider-refusal exactness, fixed-RH-statement separation, zeta-witness interface exposure, componentwise transport, displayed replay, package provenance, and local naming. Every decision-facing read must expose*

$$R, G, D, F, Z \longrightarrow H, C, P, N.$$

The packet distinguishes RH truth from route-closure decidability: it can record finite grounded routes and reject global closure deciders, but it does not provide $\Phi : \mathsf{RH}$, a counterexample datum, or a fragment-specific independence theorem.

Proof. Carrier admission is projection from Definition 5564.1. The route row R names the inspected RH-directed condition chain, rule tower, or proof-output route, while G records the displayed grounding-rank information attached to that route.

The refusal row D is read separately from the fixed-statement row F. Thus the packet can reject a global or expressive route-closure decider as a halting-boundary request without changing the fixed proposition RH.

The zeta-witness interface Z records what a genuine proof or counterexample datum would have to consume. It does not itself supply the total witness function, a counterexample point, or an independence theorem.

The structural rows H, C, P, N transport, replay, source, and name only the displayed route, rank, refusal, fixed-statement, and zeta-interface rows. Since no proof oracle, route oracle, halting predicate, counterexample, or independence theorem occurs among the coordinates, the listed obligations exhaust the finite NameCert surface. □

Theorem 4755.3 (RH route boundary nondecision)**.** *Every accepted* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ *packet separates three reads: the fixed* RH *statement row, the finite route or grounding-rank row, and the decider-refusal row. Consequently a consumer may use the packet to identify a finite grounded route or to reject an expressive route-closure decider, but no consumer can read an RH truth value, RH refutation, or independence result from the packet alone.*

Proof. Project the carrier $B = (R, G, D, F, Z, H, C, P, N)$. The row F names the fixed statement, while R and G name route and grounding-rank data.

The row D records the halting-boundary refusal for global or sufficiently expressive route deciders. This refusal is a statement about the classifier request, not a proof or refutation of the fixed proposition.

The row Z exposes the zeta-zero witness interface required by an actual decision datum. Because no coordinate supplies such a datum, transport and replay through H, C can preserve only the boundary information already displayed.

Package provenance and local naming through P, N do not add a truth-value coordinate. Therefore the packet is a finite halting-boundary NameCert surface, not an RH decision artifact. □

Closure status for $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5564.1, Definition 5564.2, and Definition 5564.3 seed the finite route, grounding-rank, decider-refusal, fixed-statement, zeta-interface, transport, replay, provenance, and local naming surface for the RH route halting boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove RH, refute RH, establish RH independence over any fragment, provide a global theorem-route closure decider, admit a proof-search oracle, close halting, supply Lean verification, or provide a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, route-row exposure, grounding-rank exactness, decider-refusal exactness, fixed-statement separation, zeta-witness interface exposure, transport, replay, provenance, and local naming for BEDC.Derived.RHRouteHaltingBoundaryUp.

Constructive story: An $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ packet is generated from RH route rows, grounding-rank rows, decider-refusal rows, fixed-RH-statement rows, zeta-witness interface rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4756

A Concrete Naming Certificate for $\mathsf{Premetric}^{\uparrow}$

Following Kelley on uniform sources and Bishop–Bridges constructive metric data, the $\mathsf{Premetric}^{\uparrow}$ packet records the distance-comparison surface that sits before separated reflection and metric completion. It reads the finite entourage interface of chapter 339, supplies the source row consumed by chapter 4201, and keeps the completion-facing boundary visible before chapter 1506 or the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 can be inspected. The packet names a constructive distance witness and zero-distance classifier; it does not quotient points, select representatives, assert host equality, or import ambient completeness.

Definition 4756.1 (Premetric carrier)**.** *A* $\mathsf{Premetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, U, D, Z, S, M, H, C, Q, N)$$

with the following displayed rows:

1. *X is the source carrier row whose points are admitted only through displayed BEDC windows.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, exposing the finite entourage requests that the distance witness is allowed to refine.*
3. *D is the distance-witness relation row, recording finite positive-radius comparisons between displayed point windows.*
4. *Z is the zero-distance classifier row, comparing two displayed source windows only through the radius comparisons stored in D.*
5. *S is the separated-reflection handoff row consumed by* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *after Z has been displayed.*
6. *M is the completion-facing nonescape row, binding every* $\mathsf{MetricCompletion}^{\uparrow}$ *read to X, U, D, Z, S before any completed object is named.*
7. *H records componentwise* hsame *transport for source, uniform, distance, zero-distance, reflection, completion, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite distance or zero-distance request through X, U, D, Z, S, M, H.*
9. *Q is the* Pkg *provenance over* $\mathsf{Premetric}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{Premetric}^\uparrow}$ *row.*

The classifier compares accepted packets only through $X, U, D, Z, S, M, H, C, Q, N$. It has no coordinate for quotient point equality, selected separated representatives, chosen completion limits, host metric equality, trichotomy, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 (definition) `BEDC.Derived.PremetricUp.PremetricTasteGate single carrier alignment`

Theorem 4756.2 (Premetric NameCert obligations)**.** *For every accepted* $\mathsf{Premetric}^\uparrow$ *carrier $P = (X, U, D, Z, S, M, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{Premetric}^\uparrow}$ *surface consists exactly of source-carrier admission, uniform-source readback, distance-witness exposure, zero-distance classification, separated-reflection handoff, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Each obligation is read through the finite route*

$$X, U, D, Z \longrightarrow S, M \longrightarrow H, C, Q, N.$$

Proof. Carrier admission is projection from Definition 5565.1. The source row X and uniform row U are the only inputs a consumer may inspect before asking for distance data.

The distance-witness relation D supplies finite positive-radius comparisons between displayed source windows. The zero-distance classifier Z is downstream of D, so equality-like reads are exposed as repeated distance comparisons rather than as host point equality.

The reflection row S and completion boundary M are then forced to read X, U, D, Z. Componentwise transport through H and replay through C preserve that order, while Q, N record provenance and local naming.

Since quotient point equality, selected separated representatives, chosen completion limits, host metric equality, trichotomy, countable choice, and ambient Real completeness do not occur among the carrier coordinates, no NameCert obligation can be discharged outside the displayed finite route. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCarrier namecert obligations`

Theorem 4756.3 (Premetric separated-reflection route)**.** *Let $P = (X, U, D, Z, S, M, H, C, Q, N)$ be an accepted* $\mathsf{Premetric}^\uparrow$ *carrier. Every* $\mathsf{SeparatedReflectionUniversal}^\uparrow$ *consumer of P must first read the source carrier X, the uniform source row U, the distance-witness relation D, and the zero-distance classifier Z. The handoff row S therefore presents the reflection source as a finite zero-distance comparison surface, not as a quotient carrier or a selected representative map.*

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricSeparatedReflectionRoute`

Proof. Apply the obligation theorem, Definition 5565.2. It identifies X, U, D, Z as the full prefix needed before the separated-reflection handoff can be consumed.

The row U records the uniform-source requests, while D records the finite distance comparisons refining those requests. The classifier Z reads zero distance only through those comparisons.

Thus S supplies the premetric source expected by the separated-reflection route of chapter 4201. The structural rows H, C, Q, N transport, replay, source, and name the same handoff; they do not add quotient data, selected representatives, or host equality. □

Theorem 4756.4 (Premetric zero-distance classifier boundary)**.** *Let $P = (X, U, D, Z, S, M, H, C, Q, N)$ be an accepted* $\mathsf{Premetric}^\uparrow$ *carrier. Every equality-like read of P must pass through the finite distance-witness row D and the zero-distance classifier row Z before any* $\mathsf{SeparatedReflectionUniversal}^\uparrow$, $\mathsf{MetricCompletion}^\uparrow$, *or* $\mathsf{Real}^\uparrow$ *consumer may inspect the packet. Thus zero-distance comparison is*

the only equality-facing classifier route exposed by the premetric certificate; no consumer receives quotient point equality, selected separated representatives, host metric equality, chosen completion limits, or ambient completeness data.

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCarrier zero distance classifier boundary`

Proof. Project the carrier of Definition 5565.1. The source row X and uniform row U admit the displayed point windows, while D is the only row that records positive-radius distance witnesses.

Apply the NameCert obligation surface of Definition 5565.2. It places the zero-distance classifier Z downstream of D, so any equality-like request must be read as a finite family of distance comparisons before it can be passed onward.

The separated-reflection route of Definition 5565.3 consumes exactly that D, Z prefix through S. Completion-facing readers then use the same prefix through M, so neither route can replace the classifier by a quotient carrier, a selected representative, or host metric equality.

The refused endpoints follow from the carrier inventory. The tuple has no coordinate for quotient equality, selected representatives, chosen completion limits, host metric equality, or ambient completeness, and the support rows H, C, Q, N only transport, replay, package, and name the displayed premetric route. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricZeroDistanceClassifierBoundary`

Theorem 4756.5 (Premetric completion boundary)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{Premetric}^{\uparrow}$ *packet reaches completion-facing data only through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow}, \mathsf{Premetric}^{\uparrow}, \mathsf{SeparatedReflectionUniversal}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

In particular, the completion-facing row M exports finite distance-comparison and zero-distance data, not a completed point, a selected Cauchy limit, a quotient equality, or an ambient completeness principle.

Proof. Project an accepted carrier $P = (X, U, D, Z, S, M, H, C, Q, N)$. The completion-facing row M is downstream of the source, uniform, distance, zero-distance, and separated-reflection rows.

Any completion consumer must therefore read X, U, D, Z, S before it can inspect M. The terminal $\mathsf{Real}^{\uparrow}$ seal, when a downstream completion chapter exposes one, is reached only after that same finite route has been displayed.

The packet contains no coordinate for a completed point, selected Cauchy limit, quotient equality, host metric equality, countable choice, or ambient completeness. Hence the completion boundary is a nonescape condition over the finite premetric surface. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCompletionBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}$.

Constructive story: A $\mathsf{Premetric}^{\uparrow}$ packet is generated from a source carrier row, a UniformSpace source row, a distance-witness relation, a zero-distance classifier, a separated-reflection handoff, a completion-facing nonescape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5565.1, Definition 5565.2, Definition 5565.3, and Definition 5565.5 seed the constructive premetric boundary between uniform sources, separated reflection, metric completion, and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient point equality, selected representatives, chosen completion limits, host metric equality, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked source-carrier admission, uniform-source readback, distance-witness exposure, zero-distance classification, separated-reflection handoff, completion-facing nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PremetricUp.

Chapter 4757

A Concrete Naming Certificate for FiniteDimensionalNormEquivalence$^{\uparrow}$

Following Bourbaki and Bishop–Bridges finite-dimensional analysis, the FiniteDimensionalNormEquivalence$^{\uparrow}$ packet records norm comparison on a displayed finite coordinate window rather than through an ambient basis theorem. It consumes the vector-space surface of chapter 47, the norm interface of chapter 89, the finite coordinate and matrix rows of chapter 49, the real comparison seal of chapter 38, and the finite-net source of chapter 1250. The packet is an L10-facing finite analysis route for later NormedSpace$^{\uparrow}$ and bounded-operator consumers: it exports named comparison constants over displayed finite rows, not host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, or choice-selected duals.

Definition 4757.1 (Finite-dimensional norm-equivalence carrier)**.** *A* FiniteDimensionalNormEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, K, A, B, R, L, U, H, T, P, N)$$

with the following displayed rows:

1. *V is the consumed* VecSpace$^{\uparrow}$ *row, exposing vector addition, zero, negation, scalar action, and vector classification only through displayed carrier data.*
2. *K is the finite coordinate window row, represented by a* Mat$^{\uparrow}$*-indexed coordinate table and its finite reconstruction ledger.*
3. *A and B are the two consumed* Norm$^{\uparrow}$ *rows on the same displayed vector source.*
4. *R is the* Real$^{\uparrow}$ *comparison row used to read nonnegative norm values, finite sums, and scalar comparison constants.*
5. *L is the compact finite-net lower-bound row over the coordinate unit window, supplied through* ClosedBoundedIntervalNet$^{\uparrow}$ *and its finite coverage ledger.*
6. *U is the compact finite-net upper-bound row over the same coordinate window.*
7. *H stores componentwise* hsame *routes for the vector row, coordinate row, both norm rows, real comparison rows, and finite-net bound rows.*
8. *T stores displayed* Cont *replay for each coordinate read, norm read, finite-net comparison, and downstream consumer request.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed rows $V, K, A, B, R, L, U, H, T, P, N$. It has no coordinate for host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, choice-selected duals, or an ambient normed-space object.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceTasteGate`
`single carrier alignment`

Theorem 4757.2 (Finite-dimensional norm-equivalence NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *carrier $E = (V, K, A, B, R, L, U, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space row, coordinate matrix row, two norm rows, real comparison row, compact finite-net lower and upper bound rows, transports, replay, provenance, and local naming;*

2. *coordinate exposure, requiring every vector read used by the comparison to pass through the finite coordinate window K and the* $\mathsf{Mat}^{\uparrow}$ *reconstruction ledger;*

3. *norm-source compatibility, requiring the two norm rows A and B to read the same displayed vector source V and the same* $\mathsf{Real}^{\uparrow}$ *comparison boundary R;*

4. *compact finite-net lower bound, requiring L to supply a positive lower comparison for the second norm on the first-norm unit coordinate window by finite mesh inspection;*

5. *compact finite-net upper bound, requiring U to supply an upper comparison for the second norm on the same displayed coordinate window;*

6. *exactness, requiring every exported comparison constant and every downstream norm-equivalence read to factor through $V, K, A, B, R, L, U, H, T, P, N$.*

No obligation row may be discharged by host basis selection, quotient vector equality, an infinite-dimensional separation theorem, or a choice-selected dual functional.

Proof. Carrier admission is projection from Definition 5566.1. The accepted packet lists the vector source, coordinate table, two norm rows, real comparison boundary, finite-net lower and upper bound rows, and the structural replay rows.

Coordinate exposure is finite. A consumer first reads the displayed $\mathsf{Mat}^{\uparrow}$ coordinate table K, then reconstructs vector operations through the $\mathsf{VecSpace}^{\uparrow}$ row V. Since K is a coordinate window rather than a host basis object, the proof uses only the listed reconstruction ledger.

The lower and upper comparison rows are compact finite-net reads. The row L inspects the first-norm unit coordinate window through the closed bounded interval net and records a positive lower comparison for the second norm. The row U reads the same finite window and records the upper comparison. Both rows are sealed by $\mathsf{Real}^{\uparrow}$ comparisons in R.

Finally H, T, P, N transport, replay, source, and name this finite route. The carrier contains no host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach theorem, or choice-selected dual row, so the listed obligations exhaust the local NameCert surface. □

Lean4 (definition) `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceCarrier`
Lean4 ✓ `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceNameCertObligat`

Theorem 4757.3 (Finite-dimensional compact-coordinate norm comparison)**.** *Every* $\mathsf{NormedSpace}^{\uparrow}$ *or bounded-operator consumer of an accepted* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *carrier receives the comparison route*

$$V, K, A, B, R, L, U \longmapsto H, T, P, N.$$

The route compares the two norms by finite coordinate windows and compact finite-net lower and upper bounds. It exports no ambient basis theorem, quotient-vector equality, infinite-dimensional Hahn–Banach argument, selected dual basis, or choice principle.

Proof. Start with a downstream norm-comparison request. The vector input is read through V, and its finite coordinate presentation is read through the $\mathsf{Mat}^{\uparrow}$ window K.

Read the two norms next. The rows A and B give norm values on the same displayed vector source, while R supplies the real comparison boundary for inequalities and constants.

The compact-coordinate step uses the finite net. The lower row L supplies the positive comparison on the first-norm unit coordinate window, and the upper row U supplies the matching upper comparison. Both are finite mesh reads through chapter 1250, so no point is selected from an ambient compact object.

The structural rows H, T, P, N replay this exact route to $\mathsf{NormedSpace}^{\uparrow}$ and bounded-operator consumers. Since the packet never stores host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, selected duals, or choice, the exported comparison remains a finite NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ packet is generated from a displayed $\mathsf{VecSpace}^{\uparrow}$ row, a finite $\mathsf{Mat}^{\uparrow}$ coordinate window, two $\mathsf{Norm}^{\uparrow}$ rows, a $\mathsf{Real}^{\uparrow}$ comparison boundary, compact finite-net lower and upper bound rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5566.1, Definition 5566.2, and Definition 5566.3 seed the finite coordinate norm-comparison route over $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host basis existence, quotient vector equality, infinite-dimensional norm equivalence, Hahn–Banach separation, selected dual bases, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coordinate exposure, compact finite-net lower and upper comparison rows, componentwise replay, and local NameCert exactness for BEDC.Derived.FiniteDimensionalNormEquivalence

Chapter 4758

A Concrete Naming Certificate for FiniteOrthonormalBasis$^{\uparrow}$

Following finite-dimensional linear algebra, Gram–Schmidt, and standard inner-product-space treatments, the FiniteOrthonormalBasis$^{\uparrow}$ packet records the certificate surface that turns a displayed finite spanning list into a named finite orthonormal-basis row. It consumes the orthogonalization route of chapter 1875, the vector carrier of chapter 47, the inner-product form rows of chapter 86, the norm-one boundary of chapter 89, and the finite coordinate ledger of chapter 49. The packet names only the finite selected-row certificate; it is not an infinite basis principle, a host vector equality theorem, a quotient representative selector, or an ambient completion object.

Definition 4758.1 (Finite orthonormal basis carrier)**.** *A* FiniteOrthonormalBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (V, S, G, O, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the* VecSpace$^{\uparrow}$ *source row, including the displayed zero, addition, scalar action, and finite carrier reads used by the packet.*
2. *S is the finite spanning-list row whose entries are the only vectors eligible for the basis route.*
3. *G is the* GramSchmidt$^{\uparrow}$ *handoff row, exposing the finite orthogonalization replay applied to S.*
4. *O is the orthogonality and norm-one row, displaying pairwise inner-product-zero checks and unit norm checks for the promoted rows.*
5. *Q is the coordinate and reconstruction ledger, recording the finite coefficient rows through which a consumer reconstructs displayed vectors from the promoted rows.*
6. *R is the refusal and boundary row, recording each discarded zero residual or non-promoted vector without selecting a replacement outside S.*
7. *H records componentwise* hsame *transport for vector-space data, source rows, Gram–Schmidt replay, orthonormal rows, coordinate rows, refusal rows, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from the source list through Gram–Schmidt, norm-one promotion, coordinate reconstruction, and local certificate.*
9. *P is the* Pkg *provenance over* FiniteOrthonormalBasis$^{\uparrow}$, GramSchmidt$^{\uparrow}$, VecSpace$^{\uparrow}$, InnerProduct$^{\uparrow}$, Norm$^{\uparrow}$, Mat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, S, G, O, Q, R, H, C, P, N$. It has no coordinate for infinite basis choice, host vector equality, quotient representatives, a selected basis outside the displayed rows, or ambient completion.

Lean4 ✓ `BEDC.Derived.FiniteOrthonormalBasisUp.FiniteOrthonormalBasisTasteGate single carrier alignment`

Theorem 4758.2 (Finite orthonormal basis NameCert obligations)**.** *For every accepted* $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$ *carrier $B = (V, S, G, O, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}$ *surface has exactly five finite rows:*

1. *carrier admission for the vector-space source, spanning list, Gram–Schmidt handoff, orthogonality and norm-one checks, coordinate reconstruction, refusal boundary, transport, continuation, provenance, and local naming;*
2. *Gram–Schmidt route exposure, requiring every promoted basis row to be read through G from the displayed source list S;*
3. *orthogonality and norm-one exposure, requiring the* $\mathsf{InnerProduct}^{\uparrow}$ *and* $\mathsf{Norm}^{\uparrow}$ *checks in O to be displayed before a promoted row is exported as orthonormal;*
4. *coordinate and reconstruction handoff, requiring every finite coordinate row and reconstruction read to pass through Q and the finite* $\mathsf{Mat}^{\uparrow}$ *ledger;*
5. *ledger non-escape from selected bases outside S, infinite basis choice, host vector equality, quotient representatives, and ambient completion.*

No accepted finite orthonormal-basis read has an obligation outside these rows.

Proof. Project the carrier of Definition 4758.1. The rows $V, S, G, O, Q, R, H, C, P, N$ are the only displayed rows available to a local $\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}$ read.

The route first fixes the vector-space source row V and the finite spanning list S. The Gram–Schmidt handoff row G then exposes the finite orthogonalization replay, while the refusal row R records every discarded zero residual or non-promoted vector.

The promoted rows are inspected through O. Orthogonality is read through displayed $\mathsf{InnerProduct}^{\uparrow}$ checks, and unit length is read through displayed $\mathsf{Norm}^{\uparrow}$ checks, so no host orthonormality predicate is imported.

Finally the coordinate ledger Q supplies finite reconstruction through the $\mathsf{Mat}^{\uparrow}$ rows, and H, C, P, N transport, replay, source, and name the same finite route. Since no excluded coordinate appears in the carrier, the ledger cannot escape to infinite basis choice, host vector equality, quotient representatives, or ambient completion. □

Theorem 4758.3 (Finite orthonormal basis Gram–Schmidt handoff)**.** *Every* $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$ *consumer that reads an orthonormal row, a coordinate row, or a reconstruction row from an accepted carrier $B = (V, S, G, O, Q, R, H, C, P, N)$ must first pass through the* $\mathsf{GramSchmidt}^{\uparrow}$ *handoff row G. The consumer therefore receives a finite route from the displayed spanning list to promoted orthonormal rows and coefficient reconstruction, not an infinite basis selector or an ambient completion principle.*

Proof. Read the carrier rows of Definition 4758.1. The only route from the finite source list S to promoted rows is the Gram–Schmidt handoff G followed by the orthogonality and norm-one row O.

The coordinate and reconstruction ledger Q is downstream of that promoted-row surface. It can expose finite coefficients only after G and O have displayed the orthonormal rows they reference.

Apply the obligation surface of Definition 4758.2. Its non-escape row excludes selected bases outside S, infinite basis choice, host vector equality, quotient representatives, and ambient completion, so every exported consumer read remains on the displayed Gram–Schmidt route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4758.1, Definition 4758.2, and Definition 4758.3 seed the finite orthonormal-basis packet over displayed vector-space, Gram–Schmidt, inner-product, norm, matrix, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOrthonormalBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite basis choice, host vector equality, quotient representatives, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Gram–Schmidt route exposure, orthogonality and norm-one exposure, coordinate reconstruction handoff, and ledger non-escape for BEDC.Derived.FiniteOrthonormalBasisUp.

Constructive story: A $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$ packet is generated from a VecSpaceUp source row, a finite spanning-list row, a GramSchmidtUp handoff, displayed orthogonality and norm-one checks, a finite coordinate reconstruction ledger, refusal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4759

A Concrete Naming Certificate for $\mathsf{RealCauchyConvergence}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of real convergence by explicit Cauchy moduli, the $\mathsf{RealCauchyConvergence}^{\uparrow}$ packet records the finite route by which a displayed tail schedule for regular rational names is consumed before the terminal $\mathsf{Real}^{\uparrow}$ seal is read. It sits over the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the completion-facing regular Cauchy boundary of chapter 1639. The packet names a convergence certificate for L10 Real and Completion consumers; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, select a host limit, quotient Cauchy streams, invoke countable choice, or compare completed host real values.

Definition 4759.1 (RealCauchyConvergence carrier)**.** *A* $\mathsf{RealCauchyConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, S, R, T, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic error requests and comparison budgets inspected by the convergence consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened at the tail thresholds named for those dyadic requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled windows and dyadic comparisons.*
4. *T is the displayed Cauchy modulus tail row, assigning each dyadic tolerance a finite tail index and the late-window reuse condition needed by convergence checks.*
5. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *boundary row, recording that the same D, S, R, T data are read as a completion-facing regular Cauchy packet before any Real seal is exposed.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tail modulus, finite windows, regular readback, and completion boundary have been displayed.*
7. *H records componentwise* hsame *transport for the tolerance ledger, window schedule, regular readback, tail modulus, completion boundary, Real seal, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a convergence request through T, then through D, S, R, B, and finally into the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RealCauchyConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, T, B, E, H, C, P, N$. The carrier has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealCauchyConvergenceUp.RealCauchyConvergenceTasteGate single carrier alignment`

Theorem 4759.2 (RealCauchyConvergence NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyConvergence}^{\uparrow}$ *carrier*

$$G = (D, S, R, T, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCauchyConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite tail-window inspection, regular rational readback, displayed Cauchy modulus tail control, completion-boundary handoff, Real-seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 4759.1. The accepted packet exposes the tolerance ledger D, finite windows S, regular readback R, tail modulus row T, completion boundary B, terminal Real seal E, and structural rows H, C, P, N.

The convergence obligations are then read in order. The dyadic core of chapter 291 supplies the tolerance requests, the stream interface of chapter 126 supplies the finite inspected windows, and the regular rational route of chapter 301 packages those windows as the displayed readback row.

The tail and completion rows are the only convergence-specific data. The row T fixes the late-window threshold for each displayed tolerance, while the boundary row B routes the same data through chapter 1639 before the $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is consumed.

Finally H and C transport and replay the same coordinates, while P and N record provenance and local naming. Since no other coordinate occurs in the carrier, the NameCert surface exports none of the excluded quotient, selected-limit, choice, host-equality, completeness, verification, or bridge data. □

Theorem 4759.3 (RealCauchyConvergence regular seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RealCauchyConvergence}^{\uparrow}$ *carrier factors through the finite route*

$$T, D, S, R \longmapsto B \longmapsto E$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal can be consumed. Thus a consumer receives the displayed Cauchy modulus tail threshold, dyadic tolerance rows, finite stream windows, regular rational readback, and the Bishop regular Cauchy completion boundary before any Real seal is visible.*

Proof. Start with an accepted carrier $G = (D, S, R, T, B, E, H, C, P, N)$. By Definition 4759.2, the only Real-facing row is E, and it is downstream of the displayed tail, dyadic, stream, regular-readback, and completion-boundary rows.

The route cannot skip the tail row because T is the only coordinate assigning late thresholds to dyadic tolerance requests. It cannot skip D or S because the dyadic ledger and finite windows are the only displayed sources for the inspected regular rational reads.

The completion boundary is also forced. The row B is the sole $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ handoff available to a Real or Completion consumer, so the terminal seal is read only after the regular readback R has been routed through that boundary.

Transport, replay, provenance, and local naming preserve the same sequence of rows. Since H, C, P, N add no quotient stream equality, selected limit, countable-choice schedule, host equality, or ambient completeness theorem, every $\mathsf{Real}^{\uparrow}$-facing read reaches E only through the displayed convergence packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 4759.1, Definition 4759.2, and Definition 4759.3 seed the finite convergence packet linking displayed Cauchy modulus tails, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ handoff, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient Cauchy equality, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-modulus exposure, dyadic tolerance exactness, finite-window inspection, regular rational readback, completion-boundary handoff, Real-seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RealCauchyConvergenceUp.

Constructive story: A $\mathsf{RealCauchyConvergence}^{\uparrow}$ packet is generated from a displayed Cauchy modulus tail row, DyadicRatCore tolerance data, StreamName finite windows, RegSeqRat readback, a BishopRegularCauchyCompletion boundary row, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4760

A Concrete Naming Certificate for $\mathsf{CompactUniformFiniteNet}^{\uparrow}$

Following the finite-net proof of Heine–Cantor isolated in chapter 9237, the $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ packet names the compact finite-net object consumed by uniform-modulus constructions. It reads compact admission from chapter 60, metric distance rows from chapter 57, continuous pointwise-modulus rows from chapter 59, rational lower-bound rows from chapter 37, and the uniform-modulus handoff of chapter 369. Its finite spine is the kernel ProbeBundle of chapter 6; the packet exposes only the named bundle, center rows, rational radius rows, coverage rows, lower-bound fold, transport, replay, provenance, and local naming.

Definition 4760.1 (CompactUniformFiniteNet carrier)**.** *A* $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, M, Q, B, E, R, L, U, H, D, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{Compact}^{\uparrow}$ *source row admitting the compact object.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used by the finite-net coverage witnesses.*
3. *Q is the* $\mathsf{Continuous}^{\uparrow}$ *pointwise-modulus source row whose center radii are consumed downstream.*
4. *B is the finite* ProbeBundle *row of bundled probe names.*
5. *E is the center and coverage row, assigning center histories and displayed distance witnesses to bundled probes.*
6. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *radius ledger attached to the bundled centers.*
7. *L is the rational lower-bound fold row over the finite bundle.*
8. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from the same finite-net and lower-bound data.*
9. *H records componentwise* hsame *transport for C, M, Q, B, E, R, L, U.*
10. *D records displayed* Cont *replay of the finite-net inspection, radius readback, lower-bound fold, and handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformFiniteNet}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected subcover, selected Lebesgue number, sequential compactness argument, quotient point equality, ambient real completeness theorem, or external modulus selector.

Lean4 ✓ `BEDC.Derived.CompactUniformFiniteNetUp.CompactUniformFiniteNetTasteGate single carrier alignment`

Theorem 4760.2 (CompactUniformFiniteNet NameCert obligations)**.** *For every accepted* CompactUniformFiniteNet *packet* $K = (C, M, Q, B, E, R, L, U, H, D, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNet}^{\uparrow}}$ *surface consists exactly of compact source admission, metric distance exposure, continuous pointwise-modulus exposure, finite* ProbeBundle *admission, center and coverage exposure, rational radius exposure, rational lower-bound folding, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may not replace this surface by selected subcovers, selected Lebesgue numbers, sequential compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an external modulus selector.*

Proof. Project the carrier of Definition 4760.1. The finite-net rows are exactly B, E, R, L, while C, M, Q, U locate the compact, metric, continuous, and uniform-modulus interfaces around them.

The fold is internal to the finite bundle. The row L reads only the rational radii carried by R over the displayed ProbeBundle row B, so no selected subcover or selected Lebesgue number is introduced.

The structural rows H, D, P, N preserve the same finite surface. Since no coordinate stores sequential compactness, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or an external selector, the displayed rows exhaust the local NameCert obligations. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ read is refuted if a UniformModulusUp handoff is reached without displaying the CompactUp source, MetricUp distance row, ContinuousUp pointwise-modulus row, finite ProbeBundle row, center and coverage row, RatUp radius ledger, rational lower-bound fold, transport, replay, provenance, and local naming rows.

Independence witness: compact, metric, continuous, rat, uniformmodulus, and probebundle: none is bijective to the compact-uniform finite-net packet because only $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ records the finite center bundle, rational lower-bound fold, and uniform-modulus handoff in one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNet}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformFiniteNet}^{\uparrow}$ packet is generated from CompactUp source admission, MetricUp distance rows, ContinuousUp pointwise-modulus rows, a finite ProbeBundle row, center and coverage rows, positive RatUp radius rows, a rational lower-bound fold, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNet}^{\uparrow}}$, seed)

Scope closed: Definition 4760.1 and Definition 4760.2 seed the finite ProbeBundle object promised by compact-uniform-continuity.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformFiniteNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, selected Lebesgue numbers, sequential compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite ProbeBundle exposure, center coverage, rational radius readback, lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformFiniteNetUp.

Chapter 4761

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4761.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4761.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4761.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4762

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4762.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 4762.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4762.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 4762.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 4763

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 4763.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 4763.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 4763.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 4764

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 4764.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 4764.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4764.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 4765

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4765.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4765.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4765.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4765.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4765.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4766

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4766.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4766.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4766.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4766.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4767

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4767.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4767.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4767.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4768

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4768.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4768.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4768.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4768.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4769

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4769.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 4769.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4769.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4770

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4770.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4770.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4770.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4771

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4771.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 4771.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4772

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4772.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4772.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4772.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4773

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4773.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4773.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4773.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4773.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4774

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4774.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4774.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4774.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4775

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4775.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4775.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4775.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4776

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4776.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 4776.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4776.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4777

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4777.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4777.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4777.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4777.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4778

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4778.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4778.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4778.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4778.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4779

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4779.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4779.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4779.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4780

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4780.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4780.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4780.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4780.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4780.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4781

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4781.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4781.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4781.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4782

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4782.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4782.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4782.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4782.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4782.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4782.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4783

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4783.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4783.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4783.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4784

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4784.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4784.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4784.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4785

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4785.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4785.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4785.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4785.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4786

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4786.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4786.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4786.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4787

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4787.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopFixedPointInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4787.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal $\mathsf{Real}^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4787.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* $\mathsf{LocatedInterval}^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz row M, and only then the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4787.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4788

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4788.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4788.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4788.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4788.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4789

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4789.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4789.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4789.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4790

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4790.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4790.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4790.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4791

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4791.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4791.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4791.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N.$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4791.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4791.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4792

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4792.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4792.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4793

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4793.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4793.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4793.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4794

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4794.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4794.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4794.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4795

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4795.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4795.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4795.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4796

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4796.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4796.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4796.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4797

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4797.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4797.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4798

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4798.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4798.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4798.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4799

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4799.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4799.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4799.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4800

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4800.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4800.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4800.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyProductCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4801

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4801.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4801.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4801.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 4802

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4802.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 4802.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4802.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4803

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4803.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4803.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4803.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4804

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4804.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*
10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*
12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4804.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*
2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*
3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*
4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*
5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*
6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4804.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4804.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4805

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4805.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4805.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4805.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4806

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4806.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4806.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4806.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4807

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4807.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4807.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4807.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4808

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4808.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 4808.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4808.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe
carrier

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4809

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4809.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4809.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4809.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4810

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4810.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4810.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4810.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4811

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 4811.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 4811.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 4811.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4812

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4812.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4812.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4813

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4813.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 4813.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4813.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4814

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 4814.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 4814.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4814.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 4814.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4815

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4815.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4815.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4815.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4815.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4816

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4816.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4816.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4816.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4817

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4817.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4817.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4817.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4818

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4818.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4818.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4818.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4819

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4819.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4819.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4820

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4820.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4820.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4821

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

4821.1 Carrier definition

Definition 4821.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

4821.2 Certificate surface

Theorem 4821.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 4821.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 4821.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 4822

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 4822.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 4822.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 4822.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4823

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4823.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4823.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4823.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4823.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4823.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4824

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4824.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4824.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4824.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4824.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4825

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4825.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4825.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4825.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4826

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4826.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4826.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4826.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4826.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4827

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4827.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4827.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4827.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4828

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4828.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4828.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4828.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4829

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4829.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 4829.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4830

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4830.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4830.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4830.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4831

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4831.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4831.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4831.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4831.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4832

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4832.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4832.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4832.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4833

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4833.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4833.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4833.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4834

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4834.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 4834.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4834.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4835

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4835.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4835.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4835.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4835.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4836

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4836.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4836.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4836.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4836.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4837

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4837.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4837.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4837.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4838

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4838.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4838.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4838.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4838.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4838.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4839

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4839.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4839.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4839.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4840

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4840.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4840.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4840.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4840.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4840.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4840.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4841

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4841.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4841.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4841.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4842

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4842.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4842.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4842.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4843

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4843.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4843.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4843.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4843.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4844

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4844.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4844.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4844.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4845

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4845.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4845.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4845.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4845.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4846

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4846.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4846.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4846.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4846.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4847

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4847.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4847.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4847.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4848

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4848.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4848.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4848.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4849

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4849.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4849.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4849.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4849.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4849.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4850

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4850.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4850.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4851

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4851.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4851.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4851.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4852

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4852.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4852.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4852.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4853

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4853.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4853.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4853.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4854

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4854.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4854.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4854.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4855

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4855.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4855.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4856

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4856.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4856.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4856.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4857

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4857.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4857.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4857.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4858

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4858.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4858.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4858.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4859

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4859.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4859.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4859.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 4860

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 4860.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 4860.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 4860.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 4860.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 4860.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 4860.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 4860.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 4860.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 4861

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4861.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 4861.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4861.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4862

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4862.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4862.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4862.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4863

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4863.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4863.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4863.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4863.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4864

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 4864.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 4864.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4864.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 4864.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 4864.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 4864.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 4865

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4865.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4865.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4865.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4866

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4866.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4866.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4866.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4867

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4867.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4867.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4867.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4868

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4868.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 4868.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4868.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe
carrier

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4869

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4869.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4869.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4869.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4870

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4870.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4870.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4870.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4871

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 4871.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 4871.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 4871.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 4871.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4872

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4872.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4872.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4872.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4872.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4873

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4873.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4873.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4873.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4874

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 4874.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 4874.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 4874.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 4874.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 4874.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 4875

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4875.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4875.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4875.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4876

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4876.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4876.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4876.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4877

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4877.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*

10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4877.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4878

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4878.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4878.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4879

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 4879.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 4879.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 4880

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 4880.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 4880.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 4881

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 4881.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 4881.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 4881.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 4882

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 4882.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 4882.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 4882.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 4883

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 4883.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 4883.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 4883.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 4883.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 4884

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 4884.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 4884.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 4884.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4885

A Concrete Naming Certificate for $\mathsf{RiemannIntegralCauchy}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4885.1 (Riemann integral Cauchy carrier)**.** *A* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 4885.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 4885.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 4886

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 4886.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 4886.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 4886.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 4887

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4887.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate single carrier alignment

Theorem 4887.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 4887.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 4887.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 4888

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 4888.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 4888.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 4888.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 4888.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 4889

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 4889.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 4889.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 4889.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 4889.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 4889.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 4889.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 4890

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 4890.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* BanachFixedPointStability$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 4890.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4890.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* BanachFixedPointStability$^{\uparrow}$ *packet factors through the finite route*

$$\text{BanachFixedPoint}^{\uparrow} \longmapsto \text{PicardIteration}^{\uparrow} \longmapsto \text{ContractionMapping}^{\uparrow} \longmapsto \text{CompleteMetric}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the RegSeqRat$^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The Real$^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 4890.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 4891

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 4891.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 4891.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 4892

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 4892.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 4892.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 4892.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 4892.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 4893

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 4893.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 4893.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 4893.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 4894

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 4894.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 4895

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 4895.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 4895.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 4896

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 4896.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 4896.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 4896.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 4897

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4897.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 4897.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 4897.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 4897.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 4897.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireSpace$^{\uparrow}$, Bool$^{\uparrow}$, and Nat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteFanTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A FiniteFanTheorem$^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 4898

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 4898.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 4898.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 4898.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4899

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4899.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 4899.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4899.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 4899.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 4900

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 4900.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 4900.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 4901

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4901.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 4901.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 4901.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 4901.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 4901.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 4901.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 4902

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 4902.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 4902.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 4902.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 4903

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 4903.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 4903.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 4903.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 4904

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 4904.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 4904.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 4904.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 4904.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, seed)

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 4905

A Concrete Naming Certificate for $\mathsf{BisectionReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BisectionReal}^{\uparrow}$ packet records bisection as a finite nested-interval and bracket-refinement certificate over the L10 Real route. It sits after the dyadic tolerance surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested interval route of chapter 1924. The packet names the displayed bisection ledger; it is not a Dedekind cut, quotient real, trichotomy principle, selected root, or completeness theorem.

Definition 4905.1 (BisectionReal carrier)**.** *A* $\mathsf{BisectionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, S, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval source row, recording the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, interval containment, and the finite retained subintervals consumed by the bisection route.*
2. *M is the dyadic midpoint and tolerance ledger. It records the midpoint rows, branch radii, diameter budgets, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons used at each inspected stage.*
3. *S is the sign or bracket-preservation row, storing the displayed bracket invariant that survives each retained child interval.*
4. *T is the tail request row, recording the finite precision request and the inspected bisection depth needed to answer it.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained endpoint and midpoint names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from W and M.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the real name exported by the retained nested interval route.*
8. *H records componentwise* hsame *transport for the interval source, midpoint ledger, bracket row, tail request, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through interval inspection, midpoint production, bracket preservation, dyadic tolerance readback, stream-window exposure, regular readback, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{BisectionReal}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, S, T, W, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient-real representatives, real trichotomy, selected roots, host equality of real points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BisectionRealUp.BisectionRealTasteGate single carrier alignment`

Theorem 4905.2 (BisectionReal NameCert obligations)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier*

$$B = (I, M, S, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval source, dyadic midpoint and tolerance ledger, bracket-preservation row, tail request, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *nested-source discipline, requiring every retained interval read to pass through I before midpoint or Real-facing data are available;*
3. *midpoint and tolerance discipline, requiring M to expose only dyadic midpoint, radius, and diameter rows whose budgets answer the request T;*
4. *bracket preservation, requiring S to carry the displayed sign or bracket invariant across each retained child interval;*
5. *readback discipline, requiring W and R to expose the retained interval only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
6. *seal nonescape, requiring E, H, C, P, N to export only the displayed bisection route and no Dedekind cut, quotient real, trichotomy principle, selected root, host equality, countable choice, or completeness theorem.*

No BisectionReal obligation is stored outside these rows.

Proof. Project the carrier of Definition 5711.1. The accepted packet displays exactly the nested interval source, midpoint ledger, bracket row, tail request, window rows, readback, Real seal, and structural rows.

The finite source stage is I, T. A precision request opens the source interval row and the requested bisection depth before any midpoint or retained endpoint can be inspected.

The bisection stage is M, S. A finite induction over the retained branch ledger reads one midpoint row at a time, uses the dyadic tolerance entries to bound the retained child, and carries the displayed sign or bracket invariant into the next interval row.

The readback stage is W, R, E. The retained endpoints are exposed as finite stream windows, then as regular rational readbacks, and only then as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, package, and name the same route. Since the carrier contains no Dedekind-cut, quotient-real, trichotomy, selected-root, host-equality, countable-choice, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4905.3 (BisectionReal nested interval factorization)**.** *Every* $\mathsf{BisectionReal}^{\uparrow}$ *read factors through the displayed nested interval route*

$$I, T \longmapsto M, S \longmapsto W, R \longmapsto E, H, C, P, N.$$

In particular, a Real-facing consumer first reads the $\mathsf{NestedIntervalReal}^{\uparrow}$ *source row* I*, then the finite dyadic midpoint and bracket-preservation rows* M, S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* W, R*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal and local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

Proof. Start with the accepted carrier $B = (I, M, S, T, W, R, E, H, C, P, N)$. The only interval source coordinate is I, and the only request coordinate is T, so a bisection read begins by exposing those rows.

Induct on the finite branch prefix recorded in the midpoint ledger. The base case reads the initial interval and its bracket row. In the successor step, the midpoint row in M supplies the dyadic split, the tolerance entry supplies the diameter budget, and S records that the displayed bracket invariant is preserved for the retained child.

After the last retained child, the endpoint windows are read through W and the rational data through R. The Real-facing seal E is therefore downstream from the nested interval, midpoint, bracket, stream, and regular readback rows.

The structural rows H, C, P, N do not add a second route. They transport componentwise hsame, replay the same displayed Cont chain, record Pkg provenance, and supply the local name. Thus every read factors through the stated finite route. □

Theorem 4905.4 (BisectionReal regular Cauchy handoff)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier* $B = (I, M, S, T, W, R, E, H, C, P, N)$*, the handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is available only after the retained nested-interval rows, dyadic midpoint and tolerance rows, sign or bracket-preservation rows, and finite* $\mathsf{StreamName}^{\uparrow}$ *windows have been replayed through the same* BHist *packet. The exported object is a regular Cauchy readback with a Real seal, not a selected root, quotient representative, trichotomy decision, or completeness witness.*

Proof. Use the nested interval factorization of Definition 5711.3. It places W, R, E strictly after the source row I, request row T, midpoint ledger M, and bracket row S.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the row R. It is computed from the finite stream windows in W and the dyadic tolerance entries in M, so no downstream consumer can read R without replaying the retained bisection ledger.

The $\mathsf{Real}^{\uparrow}$ handoff is the row E. It seals the regular readback already obtained from R, while H, C, P, N preserve the same packet by componentwise hsame, displayed Cont, Pkg, and local NameCert data.

Since the carrier has no selected-root, quotient-representative, trichotomy-decision, host-equality, countable-choice, Dedekind-cut, or completeness coordinate, the handoff exposes only the regular Cauchy route stated in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionReal}^{\uparrow}$ packet is generated from a nested interval source row, dyadic midpoint and tolerance ledger, sign or bracket-preservation rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, seed)

Scope closed: Definition 5711.1, Definition 5711.2, Definition 5711.3, and Definition 5711.4 seed the finite constructive bisection route from nested interval rows through dyadic midpoint, bracket-preservation, stream-window, regular readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, trichotomy on reals, selected roots, host equality of real points, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source discipline, midpoint and tolerance exactness, bracket preservation, regular Cauchy handoff, RealUp seal nonescape, and local naming for BEDC.Derived.BisectionRealUp.

Chapter 4906

A Concrete Naming Certificate for $\mathsf{DarbouxCriterion}^{\uparrow}$

Following Darboux, Riemann, and Bishop–Bridges constructive integration, the $\mathsf{DarbouxCriterion}^{\uparrow}$ packet records the upper-lower gap test as finite NameCert evidence. It sits between the lower and upper Darboux sum packets of chapter 5858 and chapter 4083, the Darboux-sum surface of chapter 1880, and the Riemann-facing consumers of chapter 1960 and chapter 1526. The packet exposes a displayed partition, lower sum, upper sum, gap, tolerance, regular rational readback, and real seal; it does not import host suprema, measure completion, a selected mesh, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4906.1 (Darboux criterion carrier)**.** *A* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, L, U, G, T, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition row, recording the inspected interval endpoints and refinement requests.*
2. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row attached to P.*
3. *U is the* ***??*** *row attached to the same partition row P.*
4. *G is the upper-minus-lower gap row comparing U and L inside the displayed rational ledger.*
5. *T is the tolerance row, recording the positive rational request against which G is checked.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same finite partition and gap evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after P, L, U, G, T, R have been displayed.*
8. *H records componentwise* hsame *transport for partition, lower-sum, upper-sum, gap, tolerance, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through P, L, U, G, T, R, E.*
10. *Q is the* Pkg *provenance over* $\mathsf{DarbouxCriterion}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, ***??***, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host suprema or infima, measure completion, a selected mesh, an ambient completeness principle, a quotient real, or a global integration theorem.

Theorem 4906.2 (Darboux criterion NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier*

$$K = (P, L, U, G, T, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, partition exposure, lower-sum and upper-sum compatibility over the same partition, gap comparison against the tolerance row, regular rational readback, terminal real-seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $P, L, U, G, T, R, E, H, C, Q, N$.

Lean4 ✓ `BEDC.Derived.DarbouxCriterionUp.DarbouxCriterionTasteGate single carrier alignment`

Proof. Project the carrier. The accepted packet exposes the partition row P, the paired lower and upper Darboux sum rows L, U, the gap row G, the tolerance row T, the regular readback row R, the seal row E, and the structural rows H, C, Q, N.

The lower and upper sum rows are tied to the same partition row, so a gap comparison cannot switch partitions while the criterion is being read. The tolerance row fixes the request against which the displayed gap is checked.

The regular readback and real seal are downstream of those finite rows. Componentwise transport and replay preserve the same order, while provenance and naming record the listed sibling packets.

Since host suprema, measure completion, selected meshes, quotient reals, ambient completeness, and global integration theorems are absent from the carrier coordinates, the displayed rows exhaust the local NameCert surface. □

Theorem 4906.3 (Darboux criterion upper-lower gap route)**.** *Every* $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *or regulated-integral-facing read of an accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *packet factors through the finite route*

$$P, L, U \longmapsto G, T \longmapsto R \longmapsto E.$$

Thus the consumer first reads a displayed partition, lower Darboux sum, and upper Darboux sum; then it reads the upper-minus-lower gap and tolerance rows; then it reads the $\mathsf{RegSeqRat}^{\uparrow}$ *hand-off and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The packet exports no direct route through host suprema, measure completion, selected mesh choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 5712.2. They require carrier admission and classifier transport through the displayed rows of Definition 5712.1.

The partition row P is read before either sum row can be used. The rows L and U are therefore paired over the same finite partition, and G is the displayed comparison between those paired rows.

The tolerance row T is the only request against which the gap is consumed. After that finite comparison has been displayed, the read may pass to R and then to the real seal row E.

The excluded routes are not coordinates of the carrier. Hence every integration-facing consumer receives the upper-lower gap criterion only through the finite partition, sum, gap, tolerance, regular-readback, and seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxCriterion}^{\uparrow}$ packet is generated from a displayed partition, lower and upper Darboux sum rows, an upper-minus-lower gap row, a tolerance row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5712.1, Definition 5712.2, and Definition 5712.3 seed the finite upper-lower gap criterion for Riemann-facing and regulated-integral-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DarbouxCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, measure completion, selected mesh choice, quotient real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, paired lower and upper sum compatibility, gap tolerance comparison, regular readback, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxCriterionUp.

Chapter 4907

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 4907.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 4907.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 4907.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 4908

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 4908.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. K *is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. N *is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. M *is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. F *is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. U *is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. T *is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. H *records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a compact-uniform tolerance request through* $K, N, M, F, U, T, H, P, L$.
9. P *is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*

The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 4908.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 4908.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 4909

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 4909.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through I, D, S, Q, E, W, F, H.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. It has no coordinate for an arbitrary subset of $\mathsf{Real}^{\uparrow}$*, arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 4909.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 4909.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 4909.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 4909.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* RegularCauchyHeineBorelR *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 4909.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 4910

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 4910.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 4910.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 4910.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 4911

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4911.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 4911.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4911.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 4912

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 4912.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 4912.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 4912.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 4912.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 4912.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 4912.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 4913

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 4913.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4913.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 4913.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 4913.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 4914

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 4914.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 4914.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 4914.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 4914.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 4915

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4915.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4915.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 4915.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 4915.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 4915.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 4915.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 4915.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 4915.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 4915.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 4915.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 4915.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 4915.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 4915.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 4915.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 4915.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 4916

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 4916.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4916.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 4916.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 4916.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 4916.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 4916.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 4916.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W is supplied by $\mathsf{StreamName}^{\uparrow}$*, R by* $\mathsf{RegSeqRat}^{\uparrow}$*, D by* $\mathsf{DyadicRatCore}^{\uparrow}$*, E by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 4916.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 4916.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 4916.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 4917

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 4917.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 4917.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 4917.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 4917.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 4918

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 4918.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 4918.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 4918.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let* $L = (S, Q, I, U, F, K, T, H$ *be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

$\mathsf{RieszThorinInterpolation}^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 4919

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 4919.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4919.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4919.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 4919.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 4920

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 4920.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 4920.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 4920.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 4920.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 4921

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 4921.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 4921.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 4921.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4922

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4922.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 4922.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4922.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 4922.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 4923

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 4923.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 4923.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 4923.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 4924

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 4924.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 4924.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 4924.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 4925

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4925.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4925.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4925.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4925.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4925.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4926

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4926.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4926.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4926.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4926.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4927

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4927.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4927.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4927.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4928

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4928.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4928.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4928.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4928.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4929

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4929.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 4929.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4929.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4930

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4930.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4930.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4930.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4931

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4931.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 4931.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4932

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4932.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4932.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4932.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4933

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4933.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4933.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4933.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4933.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4934

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4934.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4934.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4934.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4935

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4935.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4935.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4935.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4936

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4936.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 4936.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4936.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4937

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4937.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4937.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4937.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4937.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4938

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4938.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4938.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4938.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4938.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4939

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4939.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4939.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4939.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4940

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4940.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4940.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4940.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4940.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4940.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4941

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4941.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4941.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4941.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 4942

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 4942.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 4942.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 4942.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 4942.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 4942.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 4942.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 4943

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 4943.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 4943.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 4943.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 4944

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 4944.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 4944.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 4944.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 4945

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 4945.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 4945.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 4945.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 4945.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier $L = (T, M, G, K, B, C, H, R, P, N)$, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4946

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4946.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 4946.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 4946.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 4947

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4947.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 4947.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 4947.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 4947.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 4948

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 4948.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 4948.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 4948.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 4948.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 4949

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4949.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 4949.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 4949.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 4950

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 4950.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 4950.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 4950.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 4951

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4951.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 4951.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 4951.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N.$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 4951.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 4951.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 4952

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 4952.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 4952.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4953

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4953.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 4953.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 4953.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4954

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 4954.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 4954.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 4954.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 4955

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4955.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 4955.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 4955.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 4956

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 4956.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 4956.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 4956.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4957

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4957.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 4957.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 4958

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 4958.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 4958.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 4958.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 4959

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 4959.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 4959.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 4959.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 4960

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 4960.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 4960.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 4960.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 4961

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4961.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 4961.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 4961.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 4962

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4962.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 4962.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 4962.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 4963

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 4963.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 4963.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 4963.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 4964

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 4964.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 4964.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 4964.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 4964.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 4965

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 4965.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 4965.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 4965.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local NameCert rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 4966

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 4966.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 4966.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 4966.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 4967

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 4967.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 4967.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 4967.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 4968

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4968.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 4968.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 4968.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4969

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 4969.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 4969.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 4969.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 4970

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4970.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 4970.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 4970.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4971

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 4971.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 4971.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 4971.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4972

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4972.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 4972.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4973

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4973.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 4973.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4973.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4974

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 4974.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 4974.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 4974.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 4974.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 4975

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4975.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 4975.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 4975.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 4975.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 4976

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 4976.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 4976.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 4976.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 4977

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 4977.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 4977.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 4977.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 4978

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 4978.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 4978.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 4978.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 4979

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 4979.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 4979.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 4980

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 4980.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 4980.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 4981

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

4981.1 Carrier definition

Definition 4981.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

4981.2 Certificate surface

Theorem 4981.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 4981.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 4981.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 4982

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 4982.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 4982.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 4982.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4983

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 4983.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 4983.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 4983.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 4983.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 4983.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 4984

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 4984.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 4984.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 4984.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 4984.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 4985

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 4985.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 4985.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 4985.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 4986

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 4986.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 4986.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 4986.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 4986.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 4987

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4987.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 4987.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 4987.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4988

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4988.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 4988.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 4988.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4989

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4989.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 4989.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 4990

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4990.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 4990.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 4990.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 4991

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4991.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 4991.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 4991.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 4991.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 4992

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 4992.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 4992.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 4992.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 4993

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 4993.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 4993.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 4993.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 4994

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 4994.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 4994.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 4994.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 4995

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4995.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 4995.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 4995.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 4995.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 4996

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 4996.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 4996.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 4996.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 4996.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 4997

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 4997.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 4997.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 4997.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 4998

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4998.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 4998.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 4998.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 4998.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 4998.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 4999

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 4999.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 4999.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 4999.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5000

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5000.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5000.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5000.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5000.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5000.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5000.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5001

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5001.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5001.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5001.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5002

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5002.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5002.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5002.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5003

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5003.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5003.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5003.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5003.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5004

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5004.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5004.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5004.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5005

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5005.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5005.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5005.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5005.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5006

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5006.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5006.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5006.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5006.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5007

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5007.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5007.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5007.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5008

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5008.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5008.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5008.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5009

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5009.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5009.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5009.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5009.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5009.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5010

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5010.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5010.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5011

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5011.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. *T is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after W, T, D, R have been exposed.*
6. *H records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. *C records displayed* Cont *replay of the same finite filter route.*
8. *P records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5011.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^\uparrow$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{Real}^\uparrow$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^\uparrow$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5011.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^\uparrow$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^\uparrow$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^\uparrow$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^\uparrow$ *readback before the terminal* $\mathsf{Real}^\uparrow$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^\uparrow$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^\uparrow$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5012

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5012.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5012.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5012.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5013

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5013.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5013.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5013.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5014

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5014.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5014.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5014.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5015

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5015.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5015.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5016

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5016.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5016.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5016.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5017

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5017.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5017.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5017.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5018

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5018.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5018.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5018.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5019

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5019.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5019.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5019.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 5020

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 5020.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 5020.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 5020.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 5020.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 5020.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 5020.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 5020.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 5020.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 5021

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5021.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 5021.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5021.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5022

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5022.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5022.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5022.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5023

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5023.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5023.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5023.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5023.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5024

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 5024.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* ConstructiveBolzanoWeierstrass$^\uparrow$, BolzanoWeierstrass$^\uparrow$, SequentialCompact$^\uparrow$, CompactMetric$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ConstructiveBolzanoWeierstrass}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 5024.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* ConstructiveBolzanoWeierstrass$^\uparrow$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local NameCert$_{\text{ConstructiveBolzanoWeierstrass}^\uparrow}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling BolzanoWeierstrass$^\uparrow$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5024.3 (Constructive bounded Real subsequence obligation)**.** *Every* ConstructiveBolzanoWeierstrass$^\uparrow$ *consumer that reads a bounded* Real$^\uparrow$ *sequence from an accepted carrier must traverse the displayed route*

$$\text{StreamName}^\uparrow \longmapsto \text{CompactMetric}^\uparrow \longmapsto \text{BolzanoWeierstrass}^\uparrow \longmapsto \text{SequentialCompact}^\uparrow \longmapsto \text{RegSeqRat}^\uparrow \longmapsto \text{Real}^\uparrow.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 5024.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 5024.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 5024.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 5025

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5025.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5025.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5025.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5026

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5026.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5026.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5026.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5027

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5027.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5027.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5027.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5028

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5028.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5028.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5028.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe
carrier

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\text{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5029

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5029.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5029.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5029.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5030

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5030.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5030.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5030.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5031

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 5031.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 5031.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 5031.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 5031.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5032

A Concrete Naming Certificate for ContinuousRiemannIntegrability$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the ContinuousRiemannIntegrability$^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient Real$^{\uparrow}$ completeness.

Definition 5032.1 (Continuous Riemann integrability carrier)**.** *A* ContinuousRiemannIntegrability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* Continuous$^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* TaggedRealPartition$^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* DarbouxSum$^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* RiemannIntegrable$^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* WeaklyLocatedIntervalSelection$^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* StreamName$^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* RegSeqRat$^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5032.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5032.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5032.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5033

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5033.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5033.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5033.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5034

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 5034.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 5034.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 5034.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 5034.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 5034.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 5035

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5035.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5035.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5035.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5036

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5036.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5036.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5036.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5037

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5037.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5037.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5038

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5038.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5038.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5039

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 5039.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 5039.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 5040

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 5040.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 5040.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 5041

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 5041.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 5041.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 5041.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 5042

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 5042.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 5042.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 5042.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 5043

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 5043.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 5043.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 5043.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 5043.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 5044

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 5044.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 5044.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 5044.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5045

A Concrete Naming Certificate for $\mathsf{RiemannIntegralCauchy}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5045.1 (Riemann integral Cauchy carrier)**.** *A* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 5045.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 5045.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 5046

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 5046.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 5046.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 5046.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 5047

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5047.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate single carrier alignment

Theorem 5047.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 5047.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 5047.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 5048

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 5048.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 5048.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 5048.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 5048.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 5049

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 5049.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 5049.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 5049.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 5049.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 5049.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 5049.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 5050

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 5050.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* BanachFixedPointStability$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 5050.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5050.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* BanachFixedPointStability$^{\uparrow}$ *packet factors through the finite route*

$$\text{BanachFixedPoint}^{\uparrow} \longmapsto \text{PicardIteration}^{\uparrow} \longmapsto \text{ContractionMapping}^{\uparrow} \longmapsto \text{CompleteMetric}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the RegSeqRat$^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The Real$^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 5050.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 5051

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 5051.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 5051.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 5052

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 5052.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 5052.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 5052.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 5052.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, seed)

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 5053

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 5053.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 5053.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 5053.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 5054

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 5054.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 5055

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 5055.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 5055.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 5056

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 5056.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 5056.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 5056.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 5057

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5057.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 5057.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 5057.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 5057.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 5057.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireSpace$^{\uparrow}$, Bool$^{\uparrow}$, and Nat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteFanTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A FiniteFanTheorem$^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 5058

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 5058.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 5058.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 5058.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5059

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5059.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 5059.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5059.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 5059.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 5060

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 5060.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 5060.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 5061

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5061.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 5061.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5061.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 5061.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 5061.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 5061.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 5062

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 5062.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 5062.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5062.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 5063

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 5063.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 5063.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 5063.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 5064

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 5064.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 5064.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 5064.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 5064.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, seed)

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 5065

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 5065.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 5065.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 5065.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 5066

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 5066.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. *N is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. *M is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. *F is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. *H records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform tolerance request through $K, N, M, F, U, T, H, P, L$.*
9. *P is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 5066.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 5066.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 5067

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5067.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5067.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5067.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5067.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The sandwich hub routes its cofinal common-tail read through chapter 7532 before the child files expose the lower and upper RegSeqRat readbacks.

Chapter 5068

A Concrete Naming Certificate for $\mathsf{RegularCauchySandwich}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet names the two-sided bracketing surface used when regular Cauchy data is read by a real-completion consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the tail agreement boundary of chapter 1738. The packet records lower and upper regular tails as finite BEDC rows; it is not a quotient stream, a selected limit, a trichotomy principle, or an ambient completeness theorem.

Definition 5068.1 (Regular Cauchy sandwich carrier)**.** *A* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *U is the upper regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
3. *T is the common tail-agreement row inherited from* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling both bracket sides at the same precision request.*
5. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the lower and upper sides.*
6. *R_L and R_U are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed after the windows are displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both sides, the common tail row, the dyadic ledger, the windows, and the readbacks have been consumed.*
8. *H records componentwise* hsame *transport for the lower side, upper side, tail row, ledger, windows, readbacks, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each finite precision request through*

$$L, U, T, D, W_L, W_U, R_L, R_U \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySandwich}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the lower and upper regular rows, common tail agreement, dyadic tolerance ledger, finite windows, readbacks, and terminal seal belong to the same displayed bracketing route. It has no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5068.2 (Regular Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *surface consists exactly of lower regular-row exposure, upper regular-row exposure, common tail-agreement exposure, shared dyadic tolerance accounting, lower and upper finite-window exposure, lower and upper RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through N.*

Proof. Project the carrier tuple in Definition 5874.1. The lower and upper sides are displayed as L and U, and the only row that binds them is the common tail-agreement coordinate T.

The shared ledger D fixes the precision request for both sides. The finite windows W_L, W_U are opened against that same ledger, and the readback rows R_L, R_U expose exactly the inspected regular rational data.

The seal row E is downstream of those rows, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic coordinate occurs in the carrier, the local NameCert surface is exactly the displayed bracketing route. □

Theorem 5068.3 (Regular Cauchy sandwich window bracketing)**.** *Let*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. Every finite precision request consumed by the packet reads the two-sided route*

$$L, U, T, D \longmapsto W_L, W_U \longmapsto R_L, R_U \longmapsto E, H, P, N.$$

Thus lower and upper regular windows are bracketed by one displayed tolerance ledger and one common tail-agreement row, not by quotient representatives, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient completeness.

Proof. Start with a finite precision request. The packet exposes only one dyadic tolerance ledger D, so both sides of the bracket are read against the same displayed tolerance data.

The lower and upper windows W_L, W_U are then opened after the common tail row T has been displayed. Their readbacks R_L, R_U remain attached to the same packet, so neither side can be replaced by an external quotient stream or a selected representative.

The terminal seal E is reached only after both readbacks. Transport, replay, provenance, and local naming through H, C, P, N preserve that order for every accepted request, which gives the claimed two-sided bracketing route. □

Theorem 5068.4 (Regular Cauchy sandwich Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier can bypass the displayed route through the lower and upper regular rows, common tail agreement, shared dyadic ledger, lower and upper StreamName windows,*

lower and upper RegSeqRat readbacks, and local structural rows. Equivalently, every accepted terminal seal read is attached to

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

and cannot be separated from those rows by quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 5874.2. It exposes the Real seal row E only as part of the local NameCert surface containing the two regular rows, common tail row, ledger, windows, readbacks, and structural rows.

Now use Definition 5874.3. Every precision request reaches the readbacks through the same two-sided route, so a seal read detached from one side of the bracket is not a read of this packet.

Finally inspect Definition 5874.1. The carrier contains no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient Real completeness. A purported seal read using such data therefore lies outside the accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ carrier. □

Theorem 5068.5 (Regular Cauchy sandwich Cantor intersection seal)**.** *Let* $S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H$ *be an accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. If the lower and upper rows are supplied by a nested-shrink or Cantor-intersection route, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed by* S *is available only through the displayed sandwich route, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. In particular, the consumer-facing seal factors through Definition 5874.3, Definition 5874.4, chapter 4227, and chapter 7754, without asserting a selected intersection point.*

Proof. Apply Definition 5874.3. It forces every finite precision request for S through the common tail row, shared dyadic ledger, two StreamName windows, and two RegSeqRat readbacks before the seal row is visible.

The nested-shrink and Cantor-intersection inputs supply the lower and upper rows only as finite interval or shrink data. Those routes provide the source rows for the sandwich, but they do not add a coordinate for a selected point, quotient stream equality, or ambient completeness.

Now apply Definition 5874.4. Any terminal $\mathsf{Real}^{\uparrow}$ read detached from the displayed carrier is outside the accepted packet, so the Cantor-facing consumer receives only the finite sandwich seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet is generated from lower and upper regular-Cauchy rows, a common tail-agreement row, a shared dyadic tolerance ledger, lower and upper StreamName windows, lower and upper RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, seed)

Scope closed: Definition 5874.1, Definition 5874.2, Definition 5874.3, and Definition 5874.4 seed the finite two-sided regular-Cauchy bracketing route over DyadicRatCore, StreamName, RegSeqRat, Real, and CauchyTailEquivalence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common tail exactness, shared dyadic tolerance accounting, two-sided StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySandwichUp.

Theorem 5068.6 (Regular Cauchy sandwich sibling dependency route)**.** *Every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *sandwich packet depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows of chapter 301 and the* $\mathsf{StreamName}^{\uparrow}$ *pointwise window rows of chapter 126 before it exposes a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route consumes those siblings only through the displayed lower and upper windows, lower and upper regular readbacks, common tail row, shared dyadic ledger, transport, replay, provenance, and local naming rows of Definition 5874.1.*

Proof. Read the carrier rows from Definition 5874.1. The lower and upper windows W_L, W_U are the pointwise $\mathsf{StreamName}^{\uparrow}$ windows, and the rows R_L, R_U are the regular rational readbacks supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

The bracketing theorem Definition 5874.3 orders every precision request through those windows and readbacks before the seal row is consumed. The Real-seal nonescape theorem Definition 5874.4 prevents bypass through quotient stream equality, selected limits, host real equality, or ambient completeness.

Thus the chapter's sibling dependency is the concrete route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, not an external analytic carrier. □

Theorem 5068.7 (Regular Cauchy sandwich cofinal tail-mesh dependency)**.** *Every accepted* $\mathsf{RegularCauchySandwic}$ *packet whose common tail row is presented cofinally must read the* $\mathsf{CofinalTailMesh}^{\uparrow}$ *route of chapter 7532 before the lower and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are consumed. The dependency is the displayed route through the sandwich carrier, the window bracketing theorem, and the sibling dependency route; it is not an appeal to quotient streams, selected limits, trichotomy, choice, host real equality, or ambient completeness.*

Proof. Read the carrier rows in Definition 5874.1. The common tail coordinate T is displayed before the lower and upper StreamName windows, the lower and upper RegSeqRat readbacks, and the Real seal row.

When T is presented cofinally, consume the CofinalTailMesh route of chapter 7532. That route supplies the shared finite tail-window mesh before any Real or completion-facing seal is read.

Apply the window bracketing theorem Definition 5874.3. It keeps the lower and upper readbacks downstream of the common tail row and the shared dyadic ledger.

Finally use the sibling dependency route Definition 5874.6. It already requires the StreamName and RegSeqRat siblings before the Real seal, so adding the cofinal tail-mesh read fixes the horizontal dependency without adding any external analytic coordinate. □

Chapter 5069

A Concrete Naming Certificate for $\mathsf{DynamicProgramming}^{\uparrow}$

Following Bellman's finite dynamic programming principle, the $\mathsf{DynamicProgramming}^{\uparrow}$ packet records finite backward recursion data as BEDC NameCert rows. It sits after the finite path and edge ledger of chapter 236, the transition-row discipline of chapter 189, the probability-row carrier of chapter 184, the terminal real-name seal of chapter 38, and the finite horizon counter of chapter 28. The chapter names only finite states, finite actions, displayed transition and cost rows, horizon-indexed Bellman tables, policy readback, transport, replay, provenance, and local naming; it does not assert an infinite-horizon fixed point, host minimizer, choice-selected policy, quotient value equality, or ambient completeness theorem.

Definition 5069.1 (Dynamic programming carrier)**.** *A* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, T, C, H, V, P, R, N)$$

with the following displayed rows:

1. *S is the finite displayed state row, sharing its path endpoints with chapter 236.*
2. *A is the finite action row, listing exactly the actions admitted at the displayed states.*
3. *T is the transition row, read through the finite Markov and probability surfaces of chapter 189 and chapter 184.*
4. *C is the cost row, exposing the finite real-valued immediate costs and terminal costs through the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *H is the horizon row, a* $\mathsf{Nat}^{\uparrow}$*-indexed finite counter for the backward recursion stages.*
6. *V is the Bellman value table, giving one displayed value row for each state and horizon stage.*
7. *P is the policy readback row, recording only actions whose value entry is witnessed by the same finite transition and cost rows.*
8. *R records componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance over S, A, T, C, H, V, P and the cited sibling packets.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $S, A, T, C, H, V, P, R, N$. It has no coordinate for a host minimizer, selected infinite policy, analytic contraction theorem, quotient value function, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DynamicProgrammingUp.DynamicProgrammingTasteGate single carrier alignment`

Theorem 5069.2 (Dynamic programming Bellman obligations)**.** *For every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier* $D = (S, A, T, C, H, V, P, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *surface consists exactly of carrier admission for finite states, finite actions, transition rows, cost rows, horizon rows, Bellman value-table rows, policy readback rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host minimizers, selected infinite policies, quotient value equality, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 5875.1. The finite state and action rows are displayed before transition and cost rows can be read, and the horizon row fixes which backward stages may appear.

The Bellman table row V is then read stage by stage over the same displayed horizon. A policy readback row P is exposed only after the corresponding state, action, transition, cost, and value rows have all been displayed.

Transport, replay, provenance, and local naming are exhausted by R and N. Since the carrier has no further coordinate beyond $S, A, T, C, H, V, P, R, N$, no accepted read can import a host minimizer, selected infinite policy, quotient value equality, countable choice, or ambient completeness. □

Theorem 5069.3 (Dynamic programming finite-horizon induction)**.** *Every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *Bellman table is generated by finite backward induction over the displayed* $\mathsf{Nat}^{\uparrow}$ *horizon row. The terminal stage reads the terminal cost row from* C *and the* $\mathsf{Real}^{\uparrow}$ *seal; each predecessor stage reads only the previously displayed successor-stage value row, the transition row* T*, the action row* A*, the immediate cost row* C*, and the same local NameCert route.*

Proof. Project the horizon row H and read it through the unary source discipline of chapter 28. The empty endpoint is the terminal stage, so the value entry is the displayed terminal cost row sealed through $\mathsf{Real}^{\uparrow}$.

For a one-step predecessor, use the closed generation of the horizon row. The successor-stage table has already been displayed, and the new value row is computed from the finite transition row, finite action row, immediate cost row, and successor value row.

Repeating the one-step construction along the finite horizon supplies every value-table row. No step asks for an infinite horizon, a global fixed point, or an external minimizer; the induction is only the BEDC finite-history recursion named by the carrier. □

Theorem 5069.4 (Dynamic programming policy readback)**.** *Every policy row exported by an accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *packet is a readback from the displayed Bellman table. For each displayed state and horizon stage, the row* P *may expose an action only after the state row* S*, action row* A*, transition row* T*, cost row* C*, horizon row* H*, value row* V*, componentwise transport, replay, provenance, and local naming have all been read.*

Proof. Start with a policy-facing read. The carrier forces the read through the finite state and action rows, then through the transition and cost rows that define the local Bellman entry.

Next read the horizon row and the value-table row. The policy row is not an independent selector: it is attached to the displayed value entry at the same state and horizon stage.

Finally apply the structural row R and local naming row N. These rows preserve the finite route under transport and replay, so any action exposed by P remains tied to the same Bellman table entry and cannot become a host choice function or infinite policy. □

Theorem 5069.5 (Dynamic programming NameCert package)**.** *The* $\mathsf{DynamicProgramming}^{\uparrow}$ *NameCert package is the finite Bellman-recursion packet whose public surface is exactly the route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{ShortestPath}^{\uparrow}, \mathsf{MarkovChain}^{\uparrow}, \mathsf{ProbSpace}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DynamicProgramming}^{\uparrow}.$$

It exports the state, action, transition, cost, horizon, value-table, policy-readback, transport, replay, provenance, and local naming rows of Definition 5875.1. It exports no host optimization theorem, no selected global policy, no infinite-horizon convergence theorem, no quotient value-function equality, and no bridge to a standard dynamic-programming model.

Proof. The cited sibling chapters supply the input rows: shortest-path data for finite route endpoints, Markov and probability data for transition rows, real data for cost and value seals, and natural-number data for the finite horizon.

Apply Definition 5875.2 to identify the local obligation surface. Apply Definition 5875.3 to generate the value table from the terminal stage backward through the displayed horizon. Apply Definition 5875.4 to expose policy rows only as readbacks from that table.

The package is therefore a finite NameCert surface over the displayed rows. Because no carrier coordinate stores host optimization, global choice, infinite convergence, quotient equality, or a standard bridge, those objects are not exported. □

Falsifiable prediction: A claimed $\mathsf{DynamicProgramming}^{\uparrow}$ read that exposes a policy action without first displaying S, A, T, C, H, V, R, N refutes the finite Bellman carrier.

Independence witness: shortestpath, markovchain, probspace, real, nat: no listed sibling is bijective to $\mathsf{DynamicProgramming}^{\uparrow}$ because this packet binds finite state-action data, transition and cost rows, horizon-indexed Bellman value tables, policy readback, transport, replay, provenance, and local naming as one finite recursion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DynamicProgramming}^{\uparrow}$ packet is generated from finite state rows, finite action rows, transition rows, cost rows, a Nat-indexed horizon, Bellman value-table rows, policy readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5875.1, Definition 5875.2, Definition 5875.3, Definition 5875.4, and Definition 5875.5 seed the finite Bellman-recursion NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host minimizers, selected infinite policies, quotient value equality, infinite-horizon convergence, ambient RealUp completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Bellman table construction, finite-horizon induction, policy readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.DynamicProgrammingUp.

Chapter 5070

Regular Cauchy Even-Odd Substreams NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet records the standard constructive passage from one regular Cauchy name to its even and odd substreams before a Real or Completion consumer reads either side. It sits over the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the two-source interleaving packet of chapter 414, and the finite interleaving criterion of chapter 3405. The packet supplies a displayed subsequence route for the L10 Real and Completion corridor; it does not choose a new limit, quotient streams, or import ambient completeness.

Definition 5070.1 (Regular Cauchy even-odd substreams carrier)**.** *A* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, E, O, D, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row. It names one* $\mathsf{RegSeqRat}^{\uparrow}$ *source together with the finite precision requests inspected by the parity split.*
2. *E is the even selector-window row. It displays the finite windows read from the source at indices $2n$ and records no schedule beyond the even selector.*
3. *O is the odd selector-window row. It displays the finite windows read from the source at indices $2n + 1$ and records no schedule beyond the odd selector.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row. It stores the shared dyadic radius, split tolerance, comparison windows, and common-scale data used by both selector windows.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the even and odd windows under the budget D.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the even readback, available only after S, E, D, R have been displayed.*
7. *B is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the odd readback, available only after S, O, D, R have been displayed.*

8. *H records componentwise* hsame *transport for the source, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an even or odd substream request through $S, E, O, D, R, A, B, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, O, D, R, A, B, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.RegularCauchyEvenOddSubstreamsTasteGate`
`single carrier alignment`

Theorem 5070.2 (Regular Cauchy even-odd substreams NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier*

$$X = (S, E, O, D, R, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, transport, replay, provenance, and local naming rows;*
2. *source visibility, requiring both parity branches to read the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row S;*
3. *even-window exactness, requiring every even branch request to pass through the selector windows E and the budget D before any readback is visible;*
4. *odd-window exactness, requiring every odd branch request to pass through the selector windows O and the same budget D before any readback is visible;*
5. *paired regular readback admission through R, requiring the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to use the selector windows and the shared dyadic ledger already displayed;*
6. *two Real seal boundaries through A and B, requiring the even and odd Real-facing reads to remain downstream from their parity windows, D, R, H, and C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly $S, E, O, D, R, A, B, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5876.1. Its displayed inventory is exactly $S, E, O, D, R, A, B, H, C, P, N$, so the source row, parity selector windows, dyadic budget, paired readbacks, two seals, and structural rows exhaust the packet.

The source visibility clause is read from S. Both parity branches share that row, so neither branch can replace the source by a second hidden regular name.

Even-window and odd-window exactness are read from E and O under the shared dyadic row D. The even branch exposes only indices of the form $2n$, while the odd branch exposes only indices of the form $2n+1$, and both branches use the same tolerance ledger before the readback row is consumed.

The paired readback row R is downstream from those windows. It repacks the two selector routes as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks without adding a quotient stream or selected host limit.

Finally the Real seals A and B are reached only after the corresponding parity route has been transported by H and replayed by C. Since P and N name only the displayed packet, the listed rows are the complete local NameCert surface. □

Theorem 5070.3 (Regular Cauchy even-odd Real-seal nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier factors through one of the two finite parity routes*

$$S, E, D \longmapsto R \longmapsto A \longmapsto H, C, P, N,$$
$$S, O, D \longmapsto R \longmapsto B \longmapsto H, C, P, N.$$

The even route reads the even selector windows and the odd route reads the odd selector windows, but both routes share the same source row, dyadic budget, paired regular readback, transport, replay, provenance, and local naming. Hence no consumer reaches a $\mathsf{Real}^{\uparrow}$ *seal through quotient stream equality, a chosen host limit, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the obligation surface of Definition 5876.2. It confines every accepted packet to the displayed source, two parity selector rows, one dyadic budget, paired readback, two seal rows, and structural data.

For an even request, the consumer reads S, E, D before R and then reaches A. The route has no coordinate other than the even selector windows and the shared budget, so the even seal cannot bypass the displayed packet.

For an odd request, the consumer reads S, O, D before R and then reaches B. This is the same source and budget route with the odd selector row in place of the even selector row.

The structural rows H, C, P, N transport, replay, source, and name the two routes componentwise. Because the carrier contains no quotient stream equality, selected host limit, countable-choice row, trichotomy branch, or ambient completeness coordinate, every Real or Completion consumer remains inside one of the two displayed parity routes. □

Theorem 5070.4 (Regular Cauchy parity cofinality)**.** *Let* $X = (S, E, O, D, R, A, B, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier over one source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*. The even selector windows* E *and the odd selector windows* O *determine cofinal tail-shift reads of that same source row. Their Real-facing seals factor through the same ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The assertion is only this cofinal finite-window factorization. It does not identify quotient streams, select a limit, use countable choice, use trichotomy, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source row S and the two selector-window rows E, O from Definition 5876.1. The rows are displayed as windows over one source $\mathsf{RegSeqRat}^{\uparrow}$ carrier, so no second regular name is introduced by the parity split.

Apply Definition 301.257 to transport late reads along the even and odd tail shifts. The tail-shift result keeps each parity branch inside the same regular-rational source route.

Next use Definition 126.670. It supplies one finite request section for the displayed selector windows, so both parity reads are compared at the finite StreamName surface rather than at a completed stream quotient.

Apply Definition 1647.3 to fuse the two parity budgets through the shared dyadic tolerance ledger. The shared D row is the DyadicRatCore face of the ordered packet.

Finally use Definition 38.713. The terminal Real seal is reached only after the dyadic budget, finite StreamName window, and RegSeqRat readback have been displayed. Thus the two parity seals are cofinal views of the same finite four-face route, with no selected limit, countable-choice selector, trichotomy branch, quotient stream equality, or ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ Real or Completion read is refuted if it reaches either $\mathsf{Real}^{\uparrow}$ seal without displaying the source RegSeqRat row, the selected even or odd StreamName windows, the shared DyadicRatCore budget, the paired RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-interleaving, cauchy-interleaving-criterion: $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one source regular Cauchy name, two parity selector-window routes, one shared dyadic budget, paired regular readbacks, two Real seal boundaries, transport, replay, provenance, and local naming as one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet is generated from one RegSeqRat source, even selector windows, odd selector windows, a shared DyadicRatCore budget, paired RegSeqRat readbacks, two RealUp seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5876.1, Definition 5876.2, Definition 5876.3, and Definition 5876.4 seed the parity-substream route used by L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parity-window exactness, shared dyadic budget exactness, paired regular readback admission, two Real-seal nonescape clauses, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.

Chapter 5071

A Concrete Naming Certificate for $\mathsf{SchwartzSpace}^{\uparrow}$

Following Schwartz and Treves on test-function spaces, the $\mathsf{SchwartzSpace}^{\uparrow}$ packet records the finite locally convex carrier surface behind rapidly decreasing smooth test functions. It sits beside the complete locally convex interface of chapter 3495, the finite family-seminorm route of chapter 5868, the pointwise test-function carrier of chapter 3399, and the derivative observation route of chapter 2210. The packet gives Fourier, tempered-distribution, weak-derivative, and PDE-facing consumers a displayed test-function-space surface without importing a host topological vector space, quotient function equality, unrestricted seminorm enumeration, selected representatives, or classical completeness.

Definition 5071.1 (SchwartzSpace carrier)**.** *A* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, V, I, G, D, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetSpace}^{\uparrow}$ *source row, exposing complete metrizability and locally convex handoff only through the displayed Frechet carrier.*
2. *V is the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite-family row, naming the seminorm labels used by the Schwartz test-function space read.*
3. *I is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-function row, recording the accepted rapidly decreasing smooth windows inspected by the packet.*
4. *G is the polynomial-growth and rapid-decay ledger. It binds each displayed seminorm index to a derivative order, a polynomial weight, and the real-valued budget read used to test decay.*
5. *D is the* $\mathsf{WeakDerivative}^{\uparrow}$ *stability row, recording derivative-window shifts and sign conventions that remain inside the same Schwartz test-function-space surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row used by seminorm values, decay budgets, and completion-facing estimates.*
7. *T is the completion and distribution nonescape row. It permits Frechet and tempered-distribution consumers to read the displayed test-function-space boundary only after F, V, I, G, D, R have been exposed.*

8. *H records componentwise* hsame *transport for Frechet, seminorm-family, test-function, rapid-decay, derivative, Real, nonescape, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every test-function-space request through $F, V, I, G, D, R, T, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Schwartz topology, a quotient of functions, an unrestricted seminorm family, a selected completion point, a distribution dual, nuclearity, Fourier inversion, or classical choice.

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.SchwartzSpaceTasteGate single carrier alignment`

Theorem 5071.2 (SchwartzSpace NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier*

$$S = (F, V, I, G, D, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Frechet-space, seminorm-family, Schwartz-function, rapid-decay, weak-derivative, Real, nonescape, transport, replay, provenance, and local naming rows;*
2. *seminorm growth routing, requiring every seminorm read to choose a displayed finite-family row in V and a polynomial-growth budget in G before a Real value can be consumed from R;*
3. *rapid-decay test-function closure, requiring every accepted test window from I to satisfy the displayed derivative-order and polynomial-weight ledger in G;*
4. *derivative-row stability, requiring weak-derivative shifts to pass through D and return to the same I, G, R surface rather than to host smooth-function equality;*
5. *completion and distribution nonescape, requiring T to mediate every Frechet, completion-facing, Fourier-facing, tempered-distribution, or PDE-facing read after F, V, I, G, D, R have been displayed.*

No $\mathsf{SchwartzSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.TasteGate.SchwartzSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5877.1. The analytic rows are exactly F, V, I, G, D, R, T, and the structural rows are exactly H, C, P, N.

Seminorm growth routing is forced by the order of the carrier. A consumer first reads the finite family row V, then the derivative and polynomial-weight budget in G, and only then may read a Real scalar estimate through R.

Rapid-decay closure and derivative stability are the two test-function-space checks. The row I supplies accepted Schwartz test windows, while G records their rapid-decay budgets and D records the weak-derivative shifts that stay inside the same displayed surface.

Finally T is a boundary row rather than an additional topological object. It lets Frechet, completion-facing, Fourier-facing, tempered-distribution, and PDE consumers replay the same rows without receiving a host Schwartz topology, quotient function equality, selected completion point, distribution dual, nuclearity, inversion theorem, or choice principle. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 5071.3 (SchwartzSpace functional-analysis handoff)**.** *Every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *packet exported to a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing consumer factors through*

$$\mathsf{FrechetSpace}^{\uparrow},\ \mathsf{SeminormedSpace}^{\uparrow},\ \mathsf{SchwartzFunction}^{\uparrow},\ \mathsf{WeakDerivative}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto F, V, I, G, D, R, T, H, C, P, N.$$

The exported object is a finite test-function-space NameCert surface with displayed seminorm growth budgets and derivative stability. It is not a host locally convex function space, an unrestricted seminorm enumeration, a selected completion, a dual-space quotient, or a classical distribution theorem.

Proof. Open the accepted carrier using Definition 5877.1. Any functional-analysis request must first choose one of the displayed source rows F, V, I, D, R and then pass through the growth ledger G.

Use Definition 5877.2. The seminorm and rapid-decay obligations keep polynomial weights and derivative orders attached to finite displayed rows, while the derivative-row obligation prevents a consumer from replacing weak-derivative stability by host smooth-function equality.

The nonescape row T is then the only export boundary. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route for every downstream consumer. Since no excluded host object is a carrier coordinate, the displayed route is the whole handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SchwartzSpace}^{\uparrow}$ consumer that reaches a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing read without first exposing $F, V, I, G, D, R, T, H, C, P, N$ refutes the carrier surface of Definition 5877.1.

Independence witness: frechet-space, seminormed-space, schwartz-function, weak-derivative, real: no listed sibling is bijective to $\mathsf{SchwartzSpace}^{\uparrow}$ because this packet jointly binds complete locally convex source rows, finite seminorm-family exposure, rapid-decay test-function windows, derivative stability, Real budget readback, completion and distribution nonescape, transport, replay, provenance, and local naming in one test-function-space carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchwartzSpace}^{\uparrow}$ packet is generated from a FrechetSpace row, a finite SeminormedSpace family row, SchwartzFunction test windows, rapid-decay and polynomial-growth ledgers, WeakDerivative stability rows, Real scalar readback, completion and distribution nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5877.1, Definition 5877.2, and Definition 5877.3 seed the finite locally convex test-function-space surface consumed by Frechet, Fourier, tempered-distribution, weak-derivative, and PDE-facing packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Schwartz topology, quotient function equality, unrestricted seminorm enumeration, selected completion, distribution duality, nuclearity, Fourier inversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seminorm growth routing, rapid-decay test-function closure, derivative-row stability, completion and distribution nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SchwartzSpaceUp.

Chapter 5072

A Concrete Naming Certificate for $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$

The $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet realizes the private/shared decoration model of chapter 9390 as finite Rule 110 classifier-decoration NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the finite pattern catalog of chapter 995, the Rule 110 classifier of chapter 5844, and the finite partition carrier of chapter 201. The packet names quotient support families, private blocks, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, transport, replay, provenance, and local naming; it does not import a host cellular semantics, an infinite orbit classifier, quotient support equality, a biological model, or a hidden enumeration oracle.

Definition 5072.1 (Rule 110 private/shared decoration carrier)**.** *A* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient support family row, naming the finite family of support blocks inspected by the decoration.*
2. *B is the private-block row, recording for each accepted support a displayed nonempty private coordinate block.*
3. *S is the shared-coordinate incidence row. For each shared coordinate it records the displayed set of incident private blocks and requires cardinality at least two.*
4. *R is the* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *row, attaching the decorated support family to finite Rule 110 row-window and pattern-tag reads.*
5. *E is the decoration-count provenance row. It records the partition and row-set enumeration data used by the private/shared decoration theorem without exposing a host generating-function oracle.*
6. *H records componentwise* hsame *transport for quotient support, private-block, shared-incidence, classifier, enumeration, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decoration request through Q, B, S, R, E, H, P, N.*

8. *P is the* Pkg *provenance over* RuleOneTenPrivateSharedDecoration$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, CellularPatternCatalog$^{\uparrow}$, RuleOneTenPatternClassifier$^{\uparrow}$, Partition$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for absent private blocks, singleton shared incidence, host quotient equality, unbounded Rule 110 evolution, biological interpretation, or an external enumeration procedure.

Theorem 5072.2 (Rule 110 private/shared decoration NameCert obligations)**.** *For every accepted* RuleOneTenPrivateSharedDecoration$^{\uparrow}$ *carrier*

$$D = (Q, B, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of quotient support family admission through Q, private-block nonemptiness through B, shared-coordinate incidence through S, Rule 110 pattern-classifier routing through R, decoration-count provenance through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5878.1. The decoration data are exactly the quotient support row Q, the private-block row B, the shared-incidence row S, the Rule 110 classifier row R, and the enumeration provenance row E.

Private-block nonemptiness is read before a support can be accepted. Shared-coordinate incidence is then read as a displayed row set of size at least two, so a shared coordinate cannot be collapsed into a private coordinate or erased from the decoration.

The Rule 110 classifier row R attaches the decorated support family to finite row-window and pattern-tag reads from the cellular siblings. The count provenance row E records the finite partition and row-set enumeration route and does not supply an external counting oracle.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since no excluded host object is present as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5072.3 (Private/shared decoration classifier handoff)**.** *Every accepted* RuleOneTenPrivateSharedDecorati
consumer that reads a decorated Rule 110 support family factors through

CellularAutomaton$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, CellularPatternCatalog$^{\uparrow}$, RuleOneTenPatternClassifier$^{\uparrow}$, Partition$^{\uparrow} \longmapsto Q$

The exported object is a finite classifier-decoration packet with private-block nonemptiness, shared incidence, and decoration-count provenance. It is not an infinite cellular-orbit classifier, a host quotient-support equality, a biological stochastic model, or an unrestricted enumeration theorem.

Proof. Open the accepted carrier using Definition 5878.1. A support-family read can inspect only Q, and each support must expose its nonempty private block through B before shared coordinates are consumed.

The shared-coordinate row S supplies the incidence sets whose size is at least two. The classifier row R then routes the accepted decoration into the existing cellular automaton, row-classifier, pattern-catalog, and Rule 110 pattern-classifier surfaces.

The provenance row E records the finite decoration-count route, while H, C, P, N preserve the same data under transport, replay, package provenance, and local naming. Thus every accepted consumer receives exactly the displayed finite packet and none of the excluded host semantics. □

Falsifiable prediction: Any accepted $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ consumer that omits private-block nonemptiness or shared-coordinate incidence rows while reading a decorated Rule 110 support family refutes the carrier surface of Definition 5878.1.

Independence witness: cellularautomaton, cellularrowclassifier, cellularpatterncatalog, rule110patternclassifier: no listed sibling is bijective to $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ because this packet jointly binds quotient support families, nonempty private blocks, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, transport, replay, provenance, and local naming.

The Lean TasteGate file for this packet must provide $\mathsf{Nontrivial}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ instances for the displayed rows $Q, B, S, R, E, H, C, P, N$ before the decoration carrier can serve as a verified classifier-decoration consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet is generated from quotient support family rows, nonempty private-block rows, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5878.1, Definition 5878.2, and Definition 5878.3 seed the finite private/shared classifier-decoration surface for Rule 110 support-family consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite Rule 110 orbit classification, host quotient-support equality, biological interpretation, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient support admission, private-block nonemptiness, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPrivateSharedDecorationUp.

Chapter 5073

A Concrete Naming Certificate for $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ packet records the finite diagonal obstruction to enumerating Cantor-space bit names. It sits over the binary prefix carrier of chapter 1475, the finite bar-facing tree discipline of chapter 1622, the fan-theoretic bounded-window route of chapter 400, and the Real-facing non-surjection boundary of chapter 3793. The packet names only finite rows: an attempted enumeration window, its displayed diagonal bit schedule, the disagreement row at each inspected index, and the no-enumeration handoff. It is not a host-cardinality theorem, not a completed powerset construction, and not a quotient-stream principle.

Definition 5073.1 (Constructive Cantor finite diagonal carrier)**.** *A* $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, W, D, G, T, F, R, H, C, P, N)$$

with the following visible rows:

1. *E is the displayed finite enumeration attempt row, listing the first n candidate Cantor-space names through finite prefix windows;*
2. *W is the finite binary observation window, naming the inspected coordinates and the finite prefix depth used for each candidate;*
3. *D is the diagonal-bit row, obtained by flipping the displayed ith bit of the ith candidate whenever that coordinate is present in W;*
4. *G is the disagreement ledger, recording for every inspected index $i < n$ that the diagonal row disagrees with the ith candidate at the displayed coordinate;*
5. *T is the tree-read row through* $\mathsf{KleeneTree}^{\uparrow}$*, ensuring that finite prefix extension and absence of a completed path are read as finite branch data;*
6. *F is the fan-window row through* $\mathsf{FanTheorem}^{\uparrow}$*, recording the bounded prefix surface consumed by any compact Cantor-space handoff;*
7. *R is the Real-facing route through* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$*, used only after the Cantor-space disagreement ledger has been displayed;*

8. *H records componentwise* hsame *transport for enumeration rows, prefix windows, diagonal bits, disagreement ledgers, tree rows, fan rows, Real-facing routes, replay rows, provenance, and local naming;*
9. *C records displayed* Cont *replay from an enumeration window to diagonal-bit construction, disagreement verification, Cantor-space handoff, and Real-facing non-surjection routing;*
10. *P is the* Pkg *provenance over* ConstructiveCantorTheorem$^{\uparrow}$, CantorSpace$^{\uparrow}$, KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, RealDiagonalNonSurjection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, W, D, G, T, F, R, H, C, P, N$. It has no coordinate for a completed enumeration of all binary streams, host powersets, quotient stream equality, countable choice, or classical cardinal comparison.

Theorem 5073.2 (Finite diagonal obligations)**.** *For every accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier $K = (E, W, D, G, T, F, R, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *surface consists exactly of finite enumeration-window admission through E, binary observation through W, diagonal-bit formation through D, inspected-index disagreement through G, tree and fan routing through T, F, Real-facing non-surjection routing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5879.1. The enumeration and observation rows are E and W, so every candidate read is confined to a displayed finite prefix window.

Build the diagonal row by induction on the finite index spine of W. The empty window contributes no bit and no disagreement row. At a successor index, read the displayed candidate bit at the diagonal coordinate and place the opposite bit in D.

The same induction fills G. Each successor step adds exactly the disagreement between the new diagonal bit and the candidate bit just inspected, while previous rows are transported componentwise by H and replayed by C.

The rows T, F, R do not add an infinite object. They only route the same finite disagreement ledger through the Kleene-tree, fan-window, and Real-facing non-surjection surfaces, so the listed rows exhaust the certificate without a host powerset, quotient stream equality, choice principle, or classical cardinal comparison. □

Theorem 5073.3 (No-enumeration window)**.** *Let K be an accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier with a displayed enumeration window E of length n. The diagonal row D determines a Cantor-space prefix window such that for every inspected index $i < n$, the ith candidate of E differs from D at the displayed diagonal coordinate recorded in G. Hence the window cannot certify that its listed candidates exhaust the displayed Cantor-space prefixes.*

Proof. Induct on the finite enumeration window. For the empty window, there is no inspected candidate and the exhaustion claim has no displayed row.

At a successor window, the induction hypothesis supplies the disagreement rows for the previous candidates. The new candidate has a displayed diagonal coordinate in W, and the diagonal construction places the opposite binary value in D.

The new row of G records this disagreement. Componentwise hsame transport preserves the previous rows and the displayed Cont replay keeps the construction inside the same finite packet.

Therefore every inspected candidate is separated from the diagonal prefix at its own displayed coordinate. A finite window containing these disagreement rows cannot also certify that the same listed candidates exhaust the displayed Cantor-space prefix surface. □

Theorem 5073.4 (Cantor-space handoff)**.** *Every downstream consumer of a* ConstructiveCantorTheorem$^{\uparrow}$ *packet receives only the finite handoff formed by the enumeration window, binary observation window, diagonal-bit row, disagreement ledger, Kleene-tree row, fan-window row, Real-facing non-surjection route, transport, replay, provenance, and local naming. The handoff may be read by* CantorSpace$^{\uparrow}$ *and then by* RealDiagonalNonSurjection$^{\uparrow}$*, but it exports no completed stream quotient, powerset object, countable-choice selector, or classical uncountability principle.*

Proof. Start with the finite rows E, W, D, G of the carrier. By Definition 5879.2, these are the only rows that form and verify the diagonal disagreement.

Next read the Cantor-space side through chapter 1475. The Kleene-tree and fan rows T, F preserve finite prefix extension and bounded-window routing; they do not select a completed path or add an ambient compactness theorem.

Finally route the same disagreement ledger to chapter 3793 through R. Since the carrier has no coordinate for quotient stream equality, countable choice, host powersets, or classical cardinal comparison, the consumer receives exactly the finite handoff named in the statement. □

Falsifiable prediction: A purported ConstructiveCantorTheorem$^{\uparrow}$ consumer that reads a no-enumeration conclusion while omitting the enumeration window, diagonal-bit row, inspected-index disagreement ledger, Cantor-space prefix row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cantorspace, kleene-tree, fantheorem, real-diagonal-non-surjection: ConstructiveCantorT is not bijective to any listed sibling because it jointly carries an attempted enumeration window, a diagonal-bit construction, an inspected-index disagreement ledger, and a downstream Cantor-space-to-Real non-surjection route in one finite NameCert packet.

Closure status for NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 5879.1, Definition 5879.2, Definition 5879.3, and Definition 5879.4 seed the finite diagonal non-enumeration packet over Cantor-space prefix rows.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, a host powerset object, a countable-choice selector, a classical cardinal comparison, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean targets for carrier admission, diagonal-bit exactness, inspected-index disagreement, Cantor-space handoff, Real-facing route preservation, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCantorTheoremUp.

Constructive story: A ConstructiveCantorTheorem$^{\uparrow}$ packet is generated from a finite attempted enumeration window, a finite binary observation window, a diagonal-bit row, an inspected-index disagreement ledger, Kleene-tree and fan-window rows, Real-facing non-surjection routing, componentwise transport, displayed replay, provenance, and a local NameCert row.

MarkovKakutaniFixedPoint$^{\uparrow}$ routes its common fixed-window packet through the compact-convex and set-valued fixed-point surfaces of chapter 1923 and chapter 3511.

Chapter 5074

Markov-Kakutani Fixed Point NameCert Construction

Following Markov and Kakutani, the $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet records the finite BEDC surface for a commuting affine family acting on a compact convex carrier. It is anchored by the convex carrier of chapter 208, the functional-analysis ambient rows of chapter 94, the compact-convex self-map handoff of chapter 1923, and the set-valued fixed-point surface of chapter 3511. The packet names a finite common fixed-point obligation route; it does not import a host compactness theorem, a global choice selector, an ambient locally convex space, or a completed real vector topology outside the displayed rows.

Definition 5074.1 (Markov-Kakutani fixed point carrier)**.** *A* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, A, F, C, W, I, H, R, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the carried* $\mathsf{ConvexSet}^{\uparrow}$ *and compact support surfaces.*
2. *A is the affine ambient row, exposing only the finite vector-operation and scalar-stability data required by the displayed family.*
3. *F is the commuting-family row, listing the finite subfamily of affine self-maps and their source and target admissions.*
4. *C is the commutation ledger, recording pairwise commutation only for the displayed finite family windows.*
5. *W is the finite fixed-window row, recording the common approximate fixed rows inspected by downstream consumers.*
6. *I is the invariant-subcarrier row, exposing the finite intersection handoff used to keep the common fixed-window route inside the compact convex carrier.*
7. *H records componentwise* hsame *transport for domain, affine ambient, family, commutation, fixed-window, invariant-subcarrier, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a family request through $D, A, F, C, W, I, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, A, F, C, W, I, H, R, P, N$. It has no coordinate for an unbounded operator family, host compactness, selected fixed point, quotient equality of nets, or an ambient topological-vector-space theorem.

Theorem 5074.2 (Markov-Kakutani fixed point obligation surface)**.** *For every accepted* MarkovKakutaniFixedPoint *carrier*

$$M = (D, A, F, C, W, I, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of compact-convex domain admission through D, affine ambient admission through A, finite commuting-family admission through F, pairwise commutation exposure through C, fixed-window exposure through W, invariant-subcarrier handoff through I, componentwise* hsame *stability through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5880.1. The compact-convex, affine, commuting-family, commutation, fixed-window, and invariant-subcarrier data are exactly the rows D, A, F, C, W, I.

The domain and affine rows bind the finite ambient context before any map in the family is read. The family row F then lists only displayed affine self-maps, and the commutation ledger C records pairwise commutation inside that finite list.

The fixed-window and invariant-subcarrier rows W, I are the only route to a common fixed-point obligation. Finally H, R, P, N transport, replay, source, and name the same packet. Since no additional coordinate is present, the listed rows exhaust the NameCert surface. □

Theorem 5074.3 (Markov-Kakutani commuting handoff)**.** *Every accepted consumer of a* MarkovKakutaniFixedPoi *packet that asks for a common fixed-window read factors through*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{SchauderFixedPoint}^{\uparrow},\ \mathsf{KakutaniFixedPoint}^{\uparrow} \longmapsto D, A, F, C, W, I, H, R, P, N.$$

The exported object is a finite commuting-family fixed-window packet. It is not a selected fixed point, a host compactness theorem, a global choice principle, a quotient of approximating nets, or an ambient locally convex-space completion.

Proof. Open the accepted carrier using Definition 5880.1. A consumer must first read the compact convex domain row D and the affine ambient row A, so the route is grounded in the displayed convex and functional-analysis siblings.

Next read the finite family row F and the commutation ledger C. These rows expose only the listed affine self-maps and their pairwise commutation windows, so no unbounded operator family or host selection principle enters the packet.

The fixed-window row W is then interpreted inside the invariant-subcarrier row I. The Schauder and Kakutani siblings provide fixed-point handoff surfaces for the displayed compact-convex route, but the Markov-Kakutani packet exports only the finite common-window ledger named by W, I.

Finally H, R, P, N preserve the same carrier under transport, replay, provenance, and local naming. The exclusion clauses in the carrier therefore prevent selected fixed points, host compactness, quotient-net equality, and ambient completion from leaving the packet. □

Falsifiable prediction: Every accepted $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ consumer must expose compact-convex domain, affine ambient, finite commuting-family, pairwise commutation, fixed-window,

invariant-subcarrier, hsame, Cont, Pkg, and NameCert rows before it can read a common fixed-window obligation.

Independence witness: convexset, functionalanalysis, schauder-fixed-point, kakutani-fixed-point: $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite commuting affine family, commutation ledger, fixed-window rows, invariant-subcarrier handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet is generated from compact-convex domain rows, affine ambient rows, finite commuting-family rows, pairwise commutation ledgers, fixed-window rows, invariant-subcarrier handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5880.1, Definition 5880.2, and Definition 5880.3 seed the finite Markov-Kakutani fixed-point packet over displayed compact-convex, affine, commuting-family, fixed-window, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected common fixed points, global choice, quotient equality of nets, ambient locally convex completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex domain admission, affine-family routing, commutation ledger coverage, fixed-window exposure, invariant-subcarrier handoff, transport, replay, provenance, and local naming for BEDC.Derived.MarkovKakutaniFixedPointUp.

Proposition 5074.4 (Markov-Kakutani fixed-window dependency route)**.** *Every accepted* MarkovKakutaniFixedPoi *consumer that reads the common fixed-window row W consumes* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *from chapter 3511 and* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *from chapter 1923 only through the displayed compact-convex fixed-window route $D, A, F, C, W, I, H, R, P, N$ of Definition 5880.3. The route records the horizontal fixed-point dependency while refusing a detached host compactness theorem, selected common fixed point, or ambient locally convex completion.*

$\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ routes its private/shared row-set ledger to the defect-refined decoration consumer of chapter 7630, while the child files keep the NameCert carrier and forward route separate.

Chapter 5075

Window-Six Private/Shared Decoration NameCert Construction

The WindowSixPrivateSharedDecoration$^{\uparrow}$ packet realizes the private/shared decoration enumeration of Definition 9462.1699 inside the DNA window-six boundary-tile source of subsection 9462.28.21. It consumes the boundary-tile packet of chapter 3393, the six-boundary coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the trigger-threshold packet of chapter 4217. The packet records private blocks, shared row sets, trigger coverage, decoration counts, transport, replay, provenance, and local naming as finite rows; it is not a biological mechanism, stochastic fitness model, hidden enumeration oracle, or quotient codon semantics.

Definition 5075.1 (Window-six private/shared decoration carrier)**.** *A* WindowSixPrivateSharedDecoration$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, T, E, D, H, C, P, N)$$

with the following displayed rows:

1. *B is the private-block row, recording the nonempty private coordinate blocks attached to each accepted decorated support.*
2. *S is the shared-row-set ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *T is the trigger-coverage row, read through the carried DNA boundary-tile, six-boundary coverage, closure, and threshold packets.*
4. *E is the decoration-count row, recording the finite enumeration factors from the source private/shared theorem.*
5. *D is the defect ledger, recording row-set-size contribution to the defect-refined decoration polynomial.*
6. *H records componentwise* hsame *transport for private blocks, shared row sets, trigger coverage, decoration count, defect ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a window-six decoration request through B, S, T, E, D, H, P, N.*
8. *P is the* Pkg *provenance over* WindowSixPrivateSharedDecoration$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, DnaBoundaryTriggerClosure$^{\uparrow}$, DNAWindowSixTriggerThreshold$^{\uparrow}$, *the source theorem of Definition 9462.1699,* BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, S, T, E, D, H, C, P, N$. It has no coordinate for biological mechanism, stochastic evolutionary semantics, quotient codon equality, hidden global enumeration, or an unrecorded trigger oracle.

Theorem 5075.2 (Window-six private/shared decoration obligations)**.** *For every accepted* WindowSixPrivateShared *carrier*

$$W = (B, S, T, E, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of private-block admission through B, shared-row-set admission through S, trigger-coverage exposure through T, decoration-count exposure through E, defect-ledger exposure through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5881.1. The private block, shared row set, trigger coverage, decoration count, defect ledger, transport, replay, provenance, and naming data are precisely $B, S, T, E, D, H, C, P, N$.

The rows B, S enforce the private/shared split: each accepted support has a displayed private block, and each shared coordinate is assigned to a row set of size at least two. The trigger-coverage row T then ties those rows to the existing window-six boundary, coverage, closure, and threshold packets.

The rows E, D record the enumeration and defect data used by the source theorem. Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5075.3 (Window-six private/shared decoration enumeration handoff)**.** *Every accepted* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *consumer that reads a private/shared window-six decoration factors through*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

The exported object is a finite private/shared row-set ledger with trigger-coverage and decoration-count rows. It is not a biological model, a quotient codon space, a hidden enumeration oracle, or an unrestricted trigger-search theorem.

Proof. Read the accepted carrier through Definition 5881.1. The private and shared ledgers B, S supply the displayed row-set split before a decoration count is exposed.

Next use T to route the split through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings. This prevents a decoration request from bypassing the window-six trigger-coverage surface.

The enumeration row E and defect row D then record the finite count and defect-polynomial data from the source theorem. The structural rows H, C, P, N transport, replay, source, and locally name the same rows.

Therefore every accepted consumer receives the finite packet described in the theorem, rather than biological semantics, quotient codon equality, hidden enumeration, or unrestricted trigger search. □

Theorem 5075.4 (Window-six private/shared decoration sibling lattice)**.** *Every accepted* WindowSixPrivateSharedD *packet consumes the sibling lattice*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

before it exports the private/shared decoration ledger $B, S, T, E, D, H, C, P, N$. The lattice is finite and row-projection based: it displays boundary tiles, six-boundary coverage, trigger closure, and trigger-threshold evidence before the private block, shared row-set, trigger coverage, decoration count, and defect rows are consumed. It is not a biological mechanism, quotient codon semantics, stochastic evolutionary model, or hidden enumeration oracle.

Proof. Project the accepted carrier $W = (B, S, T, E, D, H, C, P, N)$ of Definition 5881.1. The private/shared ledger rows B, S, the trigger row T, the enumeration and defect rows E, D, and the structural rows H, C, P, N are the only exported coordinates.

Apply the enumeration handoff of Definition 5881.3. It forces every private/shared decoration read to pass through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings before $B, S, T, E, D, H, C, P, N$ are consumed.

Thus the dependency is a finite sibling lattice: each sibling supplies a displayed row that is projected before the decoration ledger is exported. Since the carrier has no coordinate for biological mechanism, quotient codon equality, stochastic evolution, or hidden enumeration, none of those interpretations is introduced by the lattice. □

Falsifiable prediction: Every accepted $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet must expose private-block, shared-row-set, trigger-coverage, decoration-count, hsame, Cont, Pkg, and NameCert rows before it can export a private/shared window-six decoration read.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-window-six-trigger-threshold: $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ is not bijective to any listed sibling because the private/shared row-set ledger jointly carries private blocks, shared row sets, trigger coverage, decoration counts, defect rows, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.WindowSixPrivateSharedDecorationUp.TasteGate` must provide explicit Nontrivial $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ and FieldFaithful WindowSixPrivateSharedDecora instances preserving $B, S, T, E, D, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet is generated from private-block rows, shared-row-set ledgers, trigger-coverage rows, decoration-count rows, defect ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5881.1, Definition 5881.2, and Definition 5881.3 seed the finite private/shared window-six decoration packet over displayed boundary, coverage, closure, threshold, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block exposure, shared-row-set arity, trigger-coverage routing, decoration-count exposure, defect-ledger exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.WindowSixPrivateSharedDecorationUp.

Proposition 5075.5 (Window-six private/shared decoration defect forward route)**.** *The private/shared row-set ledger exported by* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *is an allowed child route for chapter 7630. Its rows $B, S, T, E, D, H, C, P, N$ supply the private block, shared row set, trigger coverage, enumeration, defect ledger, transport, replay, provenance, and local naming data used by the downstream defect-refined decoration polynomial route.*

Proof. Read the exported packet through Definition 5881.1. The private/shared ledger already separates private blocks from shared row sets and records the defect ledger as a displayed row.

The downstream chapter chapter 7630 consumes exactly that displayed ledger when it forms its defect-refined decoration polynomial route. Because the route is through the named child rows rather than through the hub file, it preserves the hub-only layout and records a genuine forward dependency between the two finite packets. □

Chapter 5076

A Concrete Naming Certificate for $\mathsf{SchurLemma}^{\uparrow}$

Following Schur and the finite-group representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{SchurLemma}^{\uparrow}$ packet records the finite Hom-intertwiner boundary for irreducible representation rows. It sits between the finite representation carrier of chapter 3662, the character-theory trace surface of chapter 245, the representation-ring package of chapter 246, the orthogonality route of chapter 1603, and the character-table consumer of chapter 3574.

Definition 5076.1 (SchurLemma carrier)**.** *A* $\mathsf{SchurLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, H, S, T, R, P, N)$$

with the following displayed rows:

1. *U is an irreducible finite representation row over the same finite group source used by Definition 3662.1;*
2. *V is a second irreducible finite representation row over the same finite group source;*
3. *H is the displayed* Hom *and intertwiner row from U to V, recording equivariance rather than all host linear maps;*
4. *S is the scalar-or-zero boundary: when the two irreducible rows are the same finite representation row, accepted endomorphism reads factor through one scalar row, and when they are distinct no nonzero intertwiner row is exported;*
5. *T records componentwise* hsame *transport for the representation rows, Hom row, scalar-or-zero boundary, route, provenance, and naming rows;*
6. *R records displayed* Cont *replay from the finite representation packet through the Hom row into the scalar-or-zero boundary;*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *row.*

The carrier is exactly this finite representation-theory packet. It has no coordinate for a host abelian category, unrestricted module category, global algebra closure, or unrecorded linear-map equality.

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaTasteGate single carrier alignment`

Theorem 5076.2 (SchurLemma NameCert obligations)**.** *For the carrier of Definition 5882.1, the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (U, V, H, S, T, R, P, N)$*, irreducible finite representation exposure through* U, V*, Hom-intertwiner exposure through* H*, scalar-or-zero boundary exposure through* S*, componentwise* hsame *transport through* T*, displayed* Cont *replay through* R*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The packet lists U, V, H, S, T, R, P, N as its whole public surface, so the local certificate cannot introduce a second Hom object, a host representation category, or a hidden linear-map equality relation.

Read the representation rows first. The rows U and V are finite irreducible representation rows of the same group source, and the Hom row H is accepted only as an equivariant row between them.

The scalar-or-zero boundary is then read through S. It records scalar endomorphism reads in the same-row case and zero Hom reads in the distinct-row case. The rows T, R, P, N transport, replay, package, and name only that finite packet. □

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaCarrier namecert obligations`

Theorem 5076.3 (SchurLemma character-table handoff)**.** *Every accepted* $\mathsf{SchurLemma}^{\uparrow}$ *packet exports to* $\mathsf{CharacterTable}^{\uparrow}$ *only the finite Hom-intertwiner boundary needed by the Schur-orthogonality and character-pairing routes. The handoff reads* U, V, H, S *before* T, R, P, N *and exports no host representation category, unrestricted matrix algebra, quotient computation, or globally chosen basis.*

Proof. Start from Definition 5882.1. The downstream character-table consumer sees exactly the two irreducible representation rows, the displayed Hom-intertwiner row, and the scalar-or-zero boundary.

Use the neighboring representation-theory packets as routing anchors. The finite representation chapter supplies the representation rows, the character-theory and representation-ring chapters supply trace and sum surfaces, and the Schur-orthogonality chapter consumes the scalar-or-zero boundary before the character-table chapter reads pairings.

The structural rows T, R, P, N preserve that same finite packet under transport, replay, provenance, and local naming. Since no host category, unrestricted matrix algebra, quotient computation, or chosen-basis row appears in the carrier, the character-table handoff cannot export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurLemma}^{\uparrow}$ packet is generated from two irreducible finite representation rows, a displayed Hom-intertwiner row, a scalar-or-zero boundary row, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, seed)

Scope closed: Definition 5882.1, Definition 5882.2, and Definition 5882.3 seed the finite Hom-intertwiner boundary between irreducible representation rows and the character-table consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host representation category, unrestricted module-category Hom, quotient computation, globally chosen bases, Lean verification, or bridge checking for $\mathsf{SchurLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, irreducible-row exposure, Hom-intertwiner exactness, scalar-or-zero boundary, transport stability, replay stability, and local NameCert nonescape for BEDC.Derived.SchurLemmaUp.

Chapter 5077

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet records the interval-minimum half of the compact-real extreme-value route as finite BEDC NameCert data. It sits over the compact interval source of chapter 3986, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located lower-bound route dual to chapter 1427, and the terminal real-name seal of chapter 38. The packet is minimum-specific: it exposes lower-bound attainment windows without importing open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5077.1 (Compact real interval minimum carrier)**.** *A* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite endpoint windows, and real endpoint seals.*
2. *K is the compact interval row, read through* $\mathsf{BishopCompactInterval}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$*, exposing only the finite inspection surface admitted for the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact interval.*
4. *L is the lower-bound and located-supremum-dual row, recording that every inspected value is above a displayed lower-bound candidate and that the candidate is reached through finite located windows rather than by trichotomy.*
5. *W is the finite witness-window row, recording the interval windows in which F and L meet a terminal value seal.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the value and lower-bound comparison windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *value seal row reached only after I, K, F, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, lower-bound row, witness windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from minimum-attainment requests through* $I, K, F, L, W, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, arbitrary real trichotomy, quotient real equality, a selected host minimum, a selected host maximum, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactRealIntervalMinimumUp.CompactRealIntervalMinimumTasteGate single carrier alignment`

Theorem 5077.2 (Compact real interval minimum NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier*

$$M = (I, K, F, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact interval source, continuous-map row, lower-bound and located-supremum-dual row, finite witness-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *value seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No minimum obligation is stored outside these rows.*

Proof. Project the carrier of Definition 5883.1. The interval row I and compact row K expose the only compact-real source windows available to the packet, while F records the continuous real-valued map inspected on those windows.

Read the lower-bound and witness rows next. The row L gives the displayed lower-bound comparison as the dual of the located-supremum route, and W binds that comparison to finite source windows where the map row and bound row meet.

The regular and real-facing rows are downstream. The row R records the regular rational readback of the same finite comparisons, while E seals only those displayed reads as $\mathsf{Real}^{\uparrow}$ values.

The remaining rows H, C, P, N transport, replay, source, and locally name the same route. Since the carrier has no coordinate for open-cover compactness, trichotomy, quotient equality, selected host extrema, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5077.3 (Compact real interval minimum located attainment)**.** *Every accepted* CompactRealIntervalMinim *consumer that reads a minimum value factors through the displayed route*

$$I, K \longmapsto F, L \longmapsto W \longmapsto R, E \longmapsto H, C, P, N.$$

Thus the compact-real interval-minimum surface exports only a finite witness-window route from a closed bounded compact interval and continuous-map row to a located lower-bound row, regular rational readback, and $\mathsf{Real}^{\uparrow}$ *value seal. It does not export open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact interval source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real interval windows.

Feed the source into the continuous-map and lower-bound rows. The row F exposes only displayed value reads, while L records the lower-bound comparison through the located-supremum-dual route rather than through a host order decision.

Use the witness-window and readback rows together. The row W records where the compact interval inspection meets the lower-bound comparison, and R repacks those comparisons as $\mathsf{RegSeqRat}^{\uparrow}$ data before the $\mathsf{Real}^{\uparrow}$ seal E is read.

Finally, transport and replay preserve the route through H and C, while P and N provide provenance and local naming. No row supplies open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient completeness, so every accepted minimum read remains inside the displayed route. □

Falsifiable prediction: Every $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ minimum read factors through $\mathsf{ClosedBoundedInterva}$ $\mathsf{BishopCompactInterval}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$-dual lower-bound data, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before reaching finite witness windows and the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ row.

Independence witness: bishopcompactinterval, closedboundedinterval, continuousmap, locatedsupremum, real: $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact interval source, continuous-map row, located lower-bound route, finite witness-window row, regular readback, Real value seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a BishopCompactIntervalUp or CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, a lower-bound row dual to LocatedSupremumUp, finite witness windows, RegSeqRatUp readbacks, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5883.1, Definition 5883.2, and Definition 5883.3 seed the finite compact-real interval-minimum route over interval, compact-source, continuous-map, lower-bound, witness-window, regular readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient real equality, selected host extrema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, lower-bound routing, regular readback, Real value sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalMinir

Chapter 5078

A Concrete Naming Certificate for $\mathsf{SimpleFunctionApproximation}^{\uparrow}$

Following Caratheodory and Bourbaki measurable approximation, the $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet names the finite-range measurable surface by which a measurable real-facing map is approximated without importing completed measure theory or host function equality. It sits after the preimage-ledger map surface of chapter 3461, the event-value rows of chapter 92, the generated-set boundary of chapter 3384, and the terminal real-name seal of chapter 38. The packet records finite range cells, measurable preimage rows, displayed tolerance data, finite observations, transport, replay, provenance, and local naming.

Definition 5078.1 (Simple function approximation carrier)**.** *A* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, R, E, T, O, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row, exposing only the measurable event surface and event-value readback needed by the approximation packet.*
2. *F is the input* $\mathsf{MeasurableFunction}^{\uparrow}$ *row, read through its displayed target-event and preimage ledger rather than through a host function object.*
3. *R is the finite range ledger, listing the displayed* $\mathsf{Real}^{\uparrow}$ *values that the simple approximation may return.*
4. *E is the finite cell ledger, assigning each range row in R to a measurable source cell whose admission is read through* $\mathsf{BorelSet}^{\uparrow}$*,* $\mathsf{Measure}^{\uparrow}$*, and the preimage rows of F.*
5. *T is the displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, naming the approximation budget used by the current consumer.*
6. *O is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the visible source probes, range reads, and tolerance comparisons exposed to the packet.*
7. *H stores componentwise* hsame *transport for the measure source, measurable-function row, finite range ledger, cell ledger, tolerance row, observation row, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay from source-event inspection through finite cell selection, range-value readback, tolerance comparison, and observation replay.*

9. *P is the* Pkg *provenance over* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, E, T, O, H, C, P, N$. It has no coordinate for almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, or equality of host functions.

Lean4 ✓ `BEDC.Derived.SimpleFunctionApproximationUp.SimpleFunctionApproximationTasteGate single carrier alignment`

Theorem 5078.2 (Simple function approximation NameCert obligations)**.** *For the carrier of Definition 5884.1, the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure source, measurable-function row, finite range ledger, measurable cell ledger, displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, finite* $\mathsf{StreamName}^{\uparrow}$ *observations, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted approximation consumer must read the finite cells through E, the measurable preimage route through F, the range values through R, and the tolerance comparison through T before any sibling* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *row is exported.*

Proof. Project the carrier $S = (M, F, R, E, T, O, H, C, P, N)$. The measure source M and measurable-function row F are the only rows from which measurable event and preimage information can be read.

Read the finite approximation data. The range ledger R supplies the displayed real values, the cell ledger E ties each range value to a measurable source cell, and the tolerance row T records the visible approximation budget against which the current observations are compared.

Replay the observation and structural rows. The row O records the finite StreamName probes, while H, C, P, N transport, replay, source, and name the same route through the sibling certificates. Since no further coordinate is present, the listed rows exhaust the NameCert obligation surface and cannot export quotient classes, completed measures, selected representatives, ambient integration theorems, or host function equality. □

Theorem 5078.3 (Simple function approximation finite-cell route)**.** *Every accepted* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *read factors through the finite route*

$$M, F \longmapsto E \longmapsto R, T \longmapsto O \longmapsto H, C, P, N.$$

Thus a consumer receives a finite measurable partition-by-cells with displayed real values and a displayed tolerance budget, not an almost-everywhere quotient class, completed measure, selected representative, ambient integral, or equality proof for host functions.

Proof. Start with an accepted approximation request. The request first exposes the measure source and the measurable-function row, so every cell read is grounded in the displayed source-event and preimage ledgers.

The cell ledger E is finite. Each displayed cell is paired with a range value from R, and the tolerance row T supplies the visible real-valued budget for the comparison requested by the consumer.

Finally replay the observations and structural rows. The finite StreamName row O records only the exposed probes, while H, C, P, N transport and name the route. No hidden coordinate remains from which a quotient, completion, representative selection, integration theorem, or host equality could be recovered. □

Falsifiable prediction: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ consumer that obtains an almost-everywhere quotient class, completed measure, selected representative, ambient integration theorem, or host function equality without first reading $M, F, R, E, T, O, H, C, P, N$ refutes the carrier of Definition 5884.1.

Independence witness: measurablefunction, measure, borelset, real, streamname: no listed sibling supplies the whole ten-row packet $S = (M, F, R, E, T, O, H, C, P, N)$, because each sibling contributes only preimage-ledger, event-value, generated-set, real-name, or finite-observation data while this chapter binds the finite-range approximation surface with tolerance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet is generated from a measure source, a measurable-function preimage ledger, finite range cells, a displayed Real tolerance row, finite StreamName observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5884.1, Definition 5884.2, and Definition 5884.3 seed the finite-range measurable approximation surface over source events, preimage rows, cell ledgers, range values, tolerance rows, observations, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell exactness, measurable preimage preservation, Real tolerance replay, StreamName observation stability, and quotient-refusal proofs for BEDC.Derived.SimpleFunctionApproxima

Chapter 5079

A Concrete Naming Certificate for $\mathsf{DyadicModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicModulus}^{\uparrow}$ packet records the finite tolerance-modulus surface by which dyadic precision requests become regular rational readback before the terminal real seal is consumed. It sits between the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the terminal seal boundary of chapter 38, and the Cauchy-modulus route carried by $\mathsf{CauchyModulus}^{\uparrow}$. The packet is a finite NameCert carrier for Real and Completion consumers; it is not a host modulus function, quotient stream equality, selected limit, countable choice, or ambient completeness principle.

Definition 5079.1 (DyadicModulus carrier)**.** *A* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic scales, finite comparison budgets, rounding windows, and precision requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose visible entries are exactly the observations demanded by the tolerance row.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by reading the stream windows under the dyadic tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, available only after D, S, R have been displayed.*
5. *H records componentwise* hsame *transport for tolerance, stream-window, regular-readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each dyadic precision request through D, S, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, ambient Real completeness, or a detached host-level modulus function.

Lean4 ✓ `BEDC.Derived.DyadicModulusUp.DyadicModulusTasteGate single carrier alignment`

Theorem 5079.2 (DyadicModulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier* $M = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window exposure, regular rational readback, terminal Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every Real or Completion consumer must read the displayed route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before using the packet as a dyadic tolerance-modulus certificate.

Proof. Project the carrier of Definition 5885.1. The tuple displays exactly the dyadic tolerance row, stream-window row, regular-readback row, terminal seal, transport, replay, provenance, and local name.

The analytic route is forced in order. A precision request first reads D, then the finite stream observations in S, then the regular rational handoff R, and only then reaches the seal boundary E.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or detached-modulus coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5079.3 (DyadicModulus RegSeqRat handoff)**.** *Let* $M = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion-facing consumer of the modulus row must factor through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before the terminal seal row* E *is read.*

The handoff route is

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as rows of one BHist *packet, with transport, replay, provenance, and local naming supplied by* H, C, P, N*.*

Proof. Start from Definition 5885.2. It already places every tolerance-modulus consumer on the displayed route D, S, R, E, H, C, P, N.

The regular rational readback is the row R, and R is downstream of both the dyadic tolerance row D and the stream-window row S. Thus a consumer cannot read a regular name, Real seal, or Completion boundary before the finite tolerance and window data have been displayed.

Finally inspect the excluded coordinates of Definition 5885.1. Because no quotient stream, selected representative, host equality, countable-choice schedule, ambient completeness theorem, or detached modulus function is present, the RegSeqRat handoff is exactly the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{DyadicModulus}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or a Completion-facing seal without first displaying one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-modulus: $\mathsf{DyadicModulus}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide tolerance arithmetic,

finite windows, regular rational readback, terminal sealing, or threshold scheduling, while this packet binds them into one finite tolerance-modulus handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}$.

Constructive story: A $\mathsf{DyadicModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat handoff row, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5885.1, Definition 5885.2, and Definition 5885.3 seed the finite dyadic tolerance-modulus carrier between dyadic precision data and regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, structural transport, and non-escape for BEDC.Derived.DyadicModulusUp.

Chapter 5080

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5080.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5080.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5080.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5080.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The $\mathsf{Cauchy}^{\uparrow}$ packet is consumed through chapter 301 and chapter 38 before downstream completion or calculus reads inspect the finite Cauchy windows.

Chapter 5081

A Concrete Naming Certificate for $\mathsf{CauchyModulusSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet records a carrier-space interface for explicit Cauchy modulus schedules. It is anchored in the single-modulus certificate of chapter 292, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the fast Cauchy sequence route of chapter 2185. The chapter treats a family of modulus schedules as finite naming data, not as a quotient of real sequences or as an ambient completeness theorem.

Definition 5081.1 (Cauchy modulus space carrier)**.** *A* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row for the family of explicit modulus schedules.*
2. *S assigns to each displayed index a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, including the monotone precision requests used by the family.*
3. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule family.*
4. *D is the dyadic tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
6. *Q records the* $\mathsf{Real}^{\uparrow}$ *seal boundaries at which a scheduled read may leave rational and dyadic data for a real-facing consumer.*
7. *H records componentwise* hsame *transport for $I, S, W, D, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay from an index and precision request through the schedule row, stream window, dyadic tolerance ledger, regular rational readback, and real seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSpace}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the displayed rows componentwise by hsame*. It has no coordinate for an unscheduled Cauchy oracle, quotient equality of sequences, a selected real limit, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSpaceUp.CauchyModulusSpaceTasteGate single carrier alignment`

Theorem 5081.2 (Cauchy modulus space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index family, schedule admission for the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission, dyadic tolerance compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal-boundary compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming for the packet.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceNameCertObligations`

Proof. Project the accepted carrier of Definition 5887.1. The analytic data are precisely I, S, W, D, R, Q, and the structural rows are precisely H, C, P, N.

Schedule admission is rowwise over I. Each index exposes its explicit modulus schedule before any stream window is inspected, so the packet cannot replace the family by an unscheduled oracle.

The tolerance and readback rows are then forced through D and R. The real boundary row Q is available only after those rational and dyadic ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5081.3 (Cauchy modulus space finite route)**.** *Every real-facing comparison of two accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carriers factors through finite schedule rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundaries, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *rows. The route exports a shared modulus-comparison surface for later diagonal* $\mathsf{Real}^{\uparrow}$ *completion arguments without introducing quotient sequence equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFiniteRoute`

Proof. Open both carriers using Definition 5887.1. A comparison can inspect only the finite index rows, schedule rows, stream windows, tolerance ledgers, readback rows, and seal boundaries.

Componentwise hsame in H transports the two packets to a common finite surface. Displayed Cont replay in C keeps each precision request attached to its schedule, stream window, dyadic tolerance, regular readback, and real seal boundary.

The provenance and naming rows P, N package that finite surface as the local certificate. Thus later diagonal completion routes can share modulus comparison data without receiving any quotient or choice-based object. □

Theorem 5081.4 (CauchyModulusSpace seed interface)**.** *Every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet exports its modulus schedule family to the L10 four-object seal route only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Cauchy}^{\uparrow}$ *root route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of this seed interface can expose quotient stream equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as part of the packet surface.*

Proof. Project the carrier of Definition 5887.1. The modulus family is displayed by I, S, and every inspected request is routed through the finite observation rows W, D, R, Q before the structural rows H, C, P, N can name or replay it.

Apply Definition 5887.3. Any real-facing comparison has already factored through finite schedule rows, stream windows, dyadic tolerances, regular rational readbacks, and the terminal Real seal boundary.

The $\mathsf{Cauchy}^{\uparrow}$ root route receives exactly that finite schedule surface. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or ambient completeness, the seed interface cannot export any of those host objects to L10 Real and Completion consumers. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceSeedInterface`

Theorem 5081.5 (CauchyModulusSpace tail composition)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

two displayed modulus schedules in the family compose only through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The composed tail read is the finite route*

$$I, S, W, D, R, Q \longmapsto H, C, P, N$$

and it exports no quotient sequence equality, selected limit, countable choice, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceTailComposition`

Proof. Project the carrier of Definition 5887.1. The two schedules are displayed rows under S indexed by I, and their inspected tails must pass through the same finite window row W.

Apply the finite route and seed interface, Definition 5887.3 and Definition 5887.4. These results force the tolerance ledger D, regular readback R, and Real seal boundary Q to remain visible before the composed tail can be consumed.

Finally replay the structural rows. Componentwise transport H, displayed continuation C, provenance P, and local naming N preserve the same finite schedule surface, so tail composition does not add quotient equality, selected limits, countable choice, or ambient completeness. □

Theorem 5081.6 (CauchyModulusSpace filter handoff)**.** *Every CauchyFilter or CauchyFilterCompletion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet receives the modulus family only through the finite handoff*

$$I, S, W, D, R, Q, H, C, P, N.$$

The handoff may be consumed as filter-facing Cauchy evidence, but it cannot be reinterpreted as quotient sequence equality, a selected limit, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFilterHandoff`

Proof. Begin with a filter-facing read. The seed interface of Definition 5887.4 already routes every scheduled request through finite stream windows, dyadic tolerances, regular rational readback, the Cauchy root route, and the terminal Real seal.

Use tail composition from Definition 5887.5. It shows that composing displayed schedules preserves the same finite route through I, S, W, D, R, Q before structural replay.

The filter consumer therefore receives only the transported and replayed packet named by H, C, P, N. Since the carrier contains no quotient equality coordinate, no selected limit, no

countable-choice schedule, and no ambient completeness row, CauchyFilter and CauchyFilterCompletion may consume only the displayed finite handoff. □

Theorem 5081.7 (CauchyModulusSpace diagonal-family exactness)**.** *A diagonal Real or Completion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet may compare schedule families only through the displayed rows*

$$I, S, W, D, R, Q, H, C, P, N.$$

The diagonal request must be routed through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary before any structural transport, replay, provenance, or local naming row can be consumed. The comparison exports no quotient stream equality, selected limit, countable-choice schedule, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceDiagonalFamilyExactness`

Proof. Start with the finite route of Definition 5887.3. It already factors every real-facing comparison of two accepted packets through the finite schedule rows, stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, and the structural rows H, C, P, N.

Apply the seed interface of Definition 5887.4. It binds every L10 request to the schedule family I, S before the finite windows W, tolerance ledger D, readback R, and seal boundary Q are exposed.

Finally use the filter handoff of Definition 5887.6. A diagonal completion consumer is still a consumer of the same finite handoff, so it receives only $I, S, W, D, R, Q, H, C, P, N$ and cannot add quotient equality, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ consumer that compares schedule families while omitting $I, S, W, D, R, Q, H, C, P, N$ refutes the carrier surface of Definition 5887.1.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, fast-cauchy-sequence: no listed sibling binds a family of explicit Cauchy modulus schedules together with finite stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, componentwise transport, replay, provenance, and local naming in one carrier-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet is generated from a finite family of explicit Cauchy modulus schedules, finite StreamName windows, dyadic tolerance ledgers, RegSeqRat readbacks, Real seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5887.1, Definition 5887.2, and Definition 5887.3 seed the carrier-space interface for explicit modulus schedule families and shared finite comparison routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close diagonal Real completion, quotient equality of Cauchy sequences, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-family exactness, dyadic and regular readback compatibility, Real-seal boundary nonescape, and a Lean carrier formalization for BEDC.Derived.CauchyModulusSpaceUp.

Chapter 5082

A Concrete Naming Certificate for $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the finite-limit language of regular category theory, the $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet records the boundary where a displayed Cauchy-completion route preserves the finite-limit rows that its source already exposes. It sits beside the Cauchy source route of chapter 3413, the metric-completion universal-property route of chapter 2124, the functorial completion route of chapter 423, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite left-exactness boundary for displayed completion data; it does not assert quotient completions, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5082.1 (Cauchy completion left-exactness carrier)**.** *A* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, U, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source finite-limit row, recording the displayed terminal, product, equalizer, or kernel-pair data available before completion is read.*
2. *U is the completion unit row, recording the source-to-completion unit reads for every object named by S.*
3. *D is the dense image row, keeping each completed finite-limit comparison tied to the same source rows and unit rows.*
4. *K is the kernel-pair preservation row, recording that equalizer and pullback comparisons are read through the displayed source kernel-pair data rather than through host quotient equality.*
5. *E is the extension ledger, recording the universal-property and functorial completion rows used to transport finite-limit witnesses into completion-facing reads.*
6. *H records componentwise* hsame *transport for S, U, D, K, E, C, P, N.*
7. *C records displayed* Cont *replay from a finite-limit request through S, U, D, K, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, selected finite limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionLeftExactnessUp.TasteGate.CauchyCompletionLeftExactnessTasteGate single carrier alignment`

Theorem 5082.2 (Cauchy completion left-exactness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the source finite-limit row, completion unit rows, dense image rows, kernel-pair preservation row, extension ledger, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every completion-facing finite-limit read must expose S, U, D, K before E, H, C, P, N can be consumed.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessNameCertObligations`

Proof. Project the carrier of Definition 5888.1. The source finite-limit row, unit row, dense image row, kernel-pair row, extension ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The finite-limit boundary is read through S, U, D, K. The source row supplies the terminal, product, equalizer, or kernel-pair data; the unit row embeds the displayed source names into completion-facing reads; the dense image row keeps those reads tied to source data; and the kernel-pair row confines exactness comparisons to displayed finite rows.

The remaining rows E, H, C, P, N transport and name the same route. Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient completeness have no carrier coordinate, projection of the displayed rows exhausts the NameCert obligations. □

Theorem 5082.3 (Cauchy completion left-exactness carrier admission)**.** *For every accepted* CauchyCompletionLeftE *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the source finite-limit row S, completion unit row U, dense image row D, kernel-pair preservation row K, extension ledger E, componentwise hsame *transport H, displayed* Cont *replay C, provenance row P, and local* NameCert *row N are all admitted as finite coordinates of the same packet. No completion-facing read may treat one of these rows as ambient host data detached from the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactness carrier admission`

Proof. Project the carrier of Definition 5888.1. Its coordinates are exactly $S, U, D, K, E, H, C, P, N$, so each named row is admitted by carrier membership.

Apply Definition 5888.2. The obligation surface requires every completion-facing finite-limit read to expose S, U, D, K before the extension, transport, replay, provenance, and naming rows can be consumed.

Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, none of the admitted rows can be reread as detached host data. □

Theorem 5082.4 (Cauchy completion left-exactness unit-density route)**.** *Every displayed finite-limit comparison read through an accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier factors first through the source finite-limit row* S*, then through the completion unit row* U*, and then through the dense image row* D *before the extension ledger* E *and structural rows* H, C, P, N *are inspected. This route records the finite source-to-completion passage without selecting representatives of completed objects or invoking countable choice.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessUnitDensityRoute`

Proof. Begin with the carrier admission theorem Definition 5888.3. It keeps S, U, D as visible coordinates of the same finite packet.

The source row S supplies the finite-limit data before completion is read. The unit row U records the displayed source-to-completion unit for each object named by S, and the dense image row D keeps the resulting completion-facing comparison tied back to the same source and unit rows.

Only after this route is exposed can E, H, C, P, N be consumed. The extension ledger and the structural rows transport, replay, package, and name the source-unit-density passage; they do not add selected representatives, countable choice, or host equality of completed objects. □

Theorem 5082.5 (Cauchy completion left-exactness kernel-pair preservation)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any equalizer, pullback, or kernel-pair comparison visible after completion is read through the displayed kernel-pair row* K *together with the source finite-limit row* S*, completion unit row* U*, and dense image row* D*. The comparison cannot be exported as quotient completion equality, a selected finite limit, or ambient equality of completed objects.*

Proof. Project the kernel-pair coordinate K from the carrier. By definition, K records equalizer and pullback comparisons through displayed source kernel-pair data rather than through a host quotient.

Use the unit-density route theorem Definition 5888.4. It places every completion-facing finite-limit comparison behind S, U, D, so the kernel-pair row is read as part of the same source-to-completion passage.

Finally apply Definition 5888.2. The rows E, H, C, P, N only extend, transport, replay, package, and name the displayed comparison. They add no coordinate for quotient completion equality, selected finite limits, or ambient equality of completed objects, so preservation remains confined to K and its source, unit, and dense-image context. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessKernelPairPreservation`

Theorem 5082.6 (Cauchy completion left-exactness regular-limit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any regular Cauchy limit read used by a finite-limit comparison factors through the source finite-limit row* S*, the completion unit row* U*, the dense image row* D*, the kernel-pair row* K*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger* E *is consumed. Thus the regular-limit comparison is a pullback of displayed source, unit, dense-image, kernel-pair, regular-readback, and real-seal rows, not a quotient completion or a selected representative of a completed object.*

Proof. Project the accepted carrier. The rows $S, U, D, K, E, H, C, P, N$ are the only admitted coordinates, so a finite-limit comparison cannot begin from a hidden completed object or a host-level limit.

Apply Definition 5888.4. It places every completion-facing finite-limit comparison behind the source finite-limit row, the completion unit row, and the dense image row before the extension ledger and structural rows are inspected.

Next apply Definition 5888.5. Equalizer, pullback, and kernel-pair comparisons remain confined to the displayed kernel-pair row together with the same source, unit, and dense-image context.

The regular Cauchy limit read is then exposed through the sibling $\mathsf{RegSeqRat}^{\uparrow}$ readback surface named in the chapter introduction, and the resulting endpoint is sealed only through the sibling $\mathsf{Real}^{\uparrow}$ boundary. Since those sibling rows are read after S, U, D, K and before E, the comparison has the announced displayed pullback route. No coordinate remains for quotient completion equality, selected representatives, selected finite limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegularLimitPullback`

Theorem 5082.7 (Cauchy completion left-exactness RegSeqRat unit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any finite-limit comparison whose completion unit route reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factors first through the source finite-limit row* S*, completion unit row* U*, dense image row* D*, and kernel-pair row* K*. It then reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger and structural rows* E, H, C, P, N *are consumed. Componentwise* hsame *transport preserves that order, so the exported comparison contains no quotient completion, selected representative, selected finite limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegSeqRatUnitPullback`

Proof. Start with the carrier admission theorem Definition 5888.3. It keeps $S, U, D, K, E, H, C, P, N$ as finite coordinates of one accepted packet, so the comparison cannot begin from an external completed object.

Apply the unit-density route theorem Definition 5888.4. This places every completion-facing finite-limit comparison behind S, U, D before the extension ledger and structural rows are inspected.

Next apply the kernel-pair preservation theorem Definition 5888.5. Pullback and equalizer comparisons remain confined to K together with the same source, unit, and dense-image context.

Finally use the regular-limit pullback theorem Definition 5888.6. Its readback route inserts the $\mathsf{RegSeqRat}^{\uparrow}$ surface and the terminal $\mathsf{Real}^{\uparrow}$ seal after S, U, D, K and before E, H, C, P, N. Componentwise hsame transport therefore preserves the displayed order, while the excluded quotient, selection, choice, host-equality, and ambient-completeness rows remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet is generated from source finite-limit rows, completion unit rows, dense image rows, kernel-pair preservation, an extension ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5888.1 and Definition 5888.2 seed the finite left-exactness boundary for displayed Bishop-style Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected finite limits, selected representatives, countable choice, host equality of completed objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source finite-limit exposure, unit-row exposure, dense image confinement, kernel-pair preservation, extension-ledger confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionLeftExactnessUp.

Theorem 5082.8 (Cauchy root UnaryHistory admission)**.** *Every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *consumer read admits a UnaryHistory-facing presentation before it enters any filter, modulus, or Real-facing route. The admitted packet is the finite row surface*

$$U = (S, Q, D, R, E, H, C, P, N),$$

where S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* Q *is the directed tail-request row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and* H, C, P, N *are componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootUnaryAdmission`

Proof. Project the root stream and tail-request carriers already displayed for $\mathsf{Cauchy}^{\uparrow}$. Their shared rows expose finite StreamName windows before any tolerance, readback, or terminal seal can be read.

The directed request row supplies the unary admission point. It records one visible later-window request over the same finite stream surface, so the read is a UnaryHistory-facing row rather than a parameter transport over an ambient sequence.

The dyadic, regular readback, Real seal, and structural rows then repack the same finite route. No quotient stream, selected limit, host equality, countable choice, or ambient completeness row is exported. □

Theorem 5082.9 (Cauchy root filter-modulus compatibility)**.** *For every admitted root packet* $U = (S, Q, D, R, E, H, C, P, N)$*, any* $\mathsf{CauchyFilter}^{\uparrow}$ *or finite filter consumer that reads the tail request must use the same modulus-compatible route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The filter row may refine the visible window named by S *through* Q*, but it cannot replace* D *by an external tolerance oracle or replace* R *by a selected limit.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootFilterModulusCompatibility`

Proof. Start with the admitted packet of Definition 5888.8. The stream row S and directed request row Q are read before any filter-facing or modulus-facing datum.

Apply the existing Cauchy filter Real-seal route. It inserts the finite request row before the dyadic tolerance ledger, so a filter consumer reaches D and R only through the displayed tail request.

The structural rows preserve that order under hsame transport and Cont replay. Thus a finite filter may narrow the displayed request, but the modulus and regular readback remain the same BEDC rows. □

Theorem 5082.10 (Cauchy root RegSeqRat Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *handoff exported by a root* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the finite windows, directed request, dyadic tolerance ledger, and regular rational readback of Definition 5888.9. The Real seal* E *is available only after* S, Q, D, R, H, C, P, N *have been displayed, and it exports no quotient filter, selected real, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the filter-modulus compatibility theorem. It places the StreamName window, directed request, dyadic tolerance ledger, and RegSeqRat readback in a fixed order before any Real-facing boundary.

The handoff row is exactly R followed by E. A Real-facing consumer therefore receives rational readback through $\mathsf{RegSeqRat}^{\uparrow}$ first and may inspect the terminal seal only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient filter, selected real, choice schedule, host equality, or ambient completeness coordinate, the handoff remains a finite BEDC route. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootRegSeqRatRealHandoff`

The $\mathsf{Cauchy}^{\uparrow}$ packet enters the L10 dependency lattice through its sibling route before completion or filter consumers inspect it. Every such consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The Cauchy route is therefore a displayed sibling dependency path, not a floating completion primitive or a host-level filter object.

Theorem 5082.11 (Cauchy L10 sibling dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer in the L10 lattice receives finite observation windows from chapter 126, regular-rational readback from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any Real-facing approximation is public. The route exports no selected limit, quotient equality, host filter object, or ambient completeness principle.*

Proof. Project the accepted Cauchy packet through its finite observation surface. The first visible rows are the $\mathsf{StreamName}^{\uparrow}$ windows, so downstream readers cannot begin from a host-level limit predicate.

Next read the regular-rational sibling. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the finite regularity readback used by Cauchy consumers before the packet reaches the Real-facing boundary.

Finally expose the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed windows and readback rows, the route supplies finite windows and regular-rational data before public approximation, and it leaves no coordinate for selected limits, quotient equality, host filters, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.L10SiblingDependency.CauchyL10SiblingDependencyRoute`

Theorem 5082.12 (Cauchy route RegSeqRat tail inversion)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer that exposes a* $\mathsf{Real}^{\uparrow}$*-facing approximation through the displayed L10 sibling route must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tail row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. If either row is absent, the consumer has no public* $\mathsf{Real}^{\uparrow}$*-facing approximation route through the* $\mathsf{Cauchy}^{\uparrow}$ *L10 lattice.*

Proof. Start with Definition 5888.11. A public approximation route is visible only after the accepted Cauchy packet has passed through finite $\mathsf{StreamName}^{\uparrow}$ windows, regular-rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Read the windows first. They determine which finite observations are available to the consumer, but they do not by themselves supply a regular rational tail or a tolerance account.

Next inspect the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The regular-rational tail supplies the readback values, while the dyadic ledger supplies the tolerance discipline needed before those values can be read as a Real-facing approximation.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed tail and tolerance rows, deleting either row leaves no public route from the Cauchy consumer to a Real-facing approximation. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRouteRegSeqRatTailInversion`

Theorem 5082.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Theorem 5082.14 (Cauchy public L10 export)**.** *The public* $\mathsf{Cauchy}^{\uparrow}$ *export for L10 consumers is exactly the finite route already established by Definition 3413.1, Definition 5888.16, Definition 5888.18, and Definition 5888.10:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *route. A public consumer receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyPublicL10Export`

Proof. Start with the carrier boundary. Definition 5888.16 exhausts the accepted Cauchy packet by the Cauchy source row, finite windows, dyadic tolerances, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Use the scoped filter-modulus route. Definition 5888.18 shows that filter, modulus, Real-facing, and completion-facing consumers all read the same finite path rather than an external completion object.

Finally apply the root readback handoff of Definition 5888.10. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same window and tolerance rows. Thus the public L10 export is the displayed route itself, with no additional host equality, selection, choice, quotient, or completeness principle. □

Theorem 5082.15 (Cauchy forward L10 sum-tail route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer that reaches the regular sum tail-estimate surface must first pass through the regular rational source of chapter 301, the terminal real-name boundary of chapter 38, and the finite tail-estimate packet of chapter 7387. The route is an L10 dependency path through displayed StreamName, DyadicRatCore,*

RegSeqRat, RegularCauchySum, Real, BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child routes. The core carrier exposes finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real-facing seal before any downstream completion or arithmetic consumer is available.

The regular sum tail-estimate chapter consumes exactly that surface twice, once for each regular source row. Its tail-estimate ledger then adds the RegularCauchySum row and finite error budget without creating a detached limit object.

Thus the forward L10 route from the Cauchy hub to the sum-tail packet is the displayed dependency path through RegSeqRat, Real, and the finite tail-estimate surface. Since each row is already a finite NameCert coordinate, no excluded host or choice principle can be read from the hub. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyBaseForwardL10Route`

Theorem 5082.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 5082.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 5082.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 5083

A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on uniform embeddings, the CauchyCompletionEmbeddingReflection$^{\uparrow}$ packet records the finite route by which a dense completion embedding is reflected through the separated completion interface. It sits over the dense completion embedding of chapter 5893, the dense uniform embedding surface of chapter 1651, the functorial completion route of chapter 1937, the separated reflection boundary of chapter 3543, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real seal of chapter 38. The packet names the completion-embedding reflection route only; it does not select representatives, form a quotient completion outside the displayed certificate, or import ambient completeness.

Definition 5083.1 (CauchyCompletionEmbeddingReflection carrier)**.** *A* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source approximation row whose metric or uniform histories feed the completion embedding.*
2. *G is the* CauchyCompletionDenseEmbedding$^{\uparrow}$ *row naming the dense completion embedding and its image-window requests.*
3. *K is the* DenseUniformEmbedding$^{\uparrow}$ *row that exposes uniform density before the completion target is read as separated.*
4. *R is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row transporting maps and comparison windows through completion.*
5. *S is the* SeparatedCompletionReflection$^{\uparrow}$ *row that reflects the target route through the separated completion boundary.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row attached to the same finite approximation windows.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row reached after A, G, K, R, S, D have been displayed.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the dense embedding, separated reflection, dyadic tolerance, and regular-rational readback rows are fixed.*

9. *H records componentwise* hsame *transport for the analytic rows and the structural rows.*
10. *C records displayed* Cont *replay of the same embedding-reflection route.*
11. *P is the* Pkg *provenance over* CauchyCompletionEmbeddingReflection$^{\uparrow}$, CauchyCompletionDenseEmbedding$^{\uparrow}$, DenseUniformEmbedding$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, SeparatedCompletionReflection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected representatives, countable choice, a detached separated completion, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.TasteGate.CauchyCompletionEmbeddingReflectio
single carrier alignment

Theorem 5083.2 (CauchyCompletionEmbeddingReflection NameCert obligations)**.** *For every accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier* $J = (A, G, K, R, S, D, Q, E, H, C, P, N)$*, the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed approximation row, dense completion embedding row, dense uniform embedding row, completion functoriality row, separated reflection row, dyadic tolerance row, regular-rational readback row, terminal real seal row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing read must pass through* $A, G, K, R, S, D, Q, E, H, C, P, N$ *in that order, and no read may bypass the separated reflection boundary.*

Proof. Project the carrier of Definition 5889.1. Its analytic rows are A, G, K, R, S, D, Q, E, followed by the structural rows H, C, P, N.

The dense route begins with A and G. The approximation row supplies the finite source histories, while the CauchyCompletionDenseEmbedding$^{\uparrow}$ row records the image-window requests through which completion consumers may inspect the dense embedding.

The rows K, R, S then bind the same route to dense uniform embedding, functorial completion transport, and separated reflection. A completion-facing consumer therefore reaches the separated target only after the dense route and functorial route have been displayed.

The rows D, Q, E provide dyadic tolerance, regular-rational readback, and the terminal real seal. Finally H, C, P, N transport, replay, source, and name the same packet, so carrier exhaustion blocks every excluded quotient, selector, detached completion, and ambient-completeness route. □

Theorem 5083.3 (CauchyCompletionEmbeddingReflection handoff)**.** *Every downstream consumer of an accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *packet receives the dense completion embedding only through the finite handoff*

$$\text{CauchyCompletionDenseEmbedding}^{\uparrow} \longmapsto \text{DenseUniformEmbedding}^{\uparrow} \longmapsto \text{CauchyCompletionFunctoriality}^{\uparrow} \longmapsto \text{Separat}$$

Thus the consumer obtains a reflected separated-completion route with finite stream windows, dyadic tolerance, regular-rational readback, and a terminal real seal; it does not obtain a quotient-completion representative, a selected dense point, countable choice, or ambient completeness.

Proof. Start with Definition 5889.2. The NameCert surface forces every public read to begin with the approximation row and the dense completion embedding row.

Read the sibling chapters named in the statement along the displayed order. The DenseUniformEmbedding$^{\uparrow}$ row supplies the uniform dense image, CauchyCompletionFunctoriality$^{\uparrow}$ transports the route through

completion, and $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ is the only separated target boundary available to the packet.

The finite observation route is then exposed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows are exactly the rows D, Q, E together with the stream-window data carried by A, G, K, R, S.

Since the carrier has only $A, G, K, R, S, D, Q, E, H, C, P, N$, the structural rows can preserve and name the handoff but cannot supply a quotient representative, selected dense point, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5889.1, Definition 5889.2, and Definition 5889.3 seed the dense completion embedding reflection route through separated completion, finite stream windows, dyadic tolerance, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, countable choice, detached separated completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense completion embedding exposure, dense uniform embedding compatibility, completion functoriality, separated reflection, finite-window readback, dyadic tolerance, regular-rational handoff, real sealing, and nonescape for BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.

Constructive story: A $\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}$ packet is generated from a source approximation row, a dense completion embedding row, a dense uniform embedding row, a completion functoriality row, a separated reflection row, dyadic tolerance, regular-rational readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5084

A Concrete Naming Certificate for $\mathsf{SnakeLemma}^{\uparrow}$

Following Cartan–Eilenberg and Weibel homological algebra, the $\mathsf{SnakeLemma}^{\uparrow}$ packet records the finite diagram chase that turns two exact rows and compatible vertical maps into the connecting boundary route. It sits over the abelian-category carrier of chapter 176, the homology carrier of chapter 98, the spectral-sequence consumer surface that exposes $\mathsf{DerivedFunctor}^{\uparrow}$ rows in chapter 178, and the derived-functor interface of chapter 177. The packet names a displayed exactness boundary; it does not import quotient-object equality, kernel-cokernel object extensionality, or an ambient abelian-category diagram oracle.

Definition 5084.1 (Snake lemma carrier)**.** *A* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row carrying the additive kernel-cokernel surface in which the diagram is read.*
2. *R_0 and R_1 are the upper and lower exact-row ledgers, each exposing three displayed object rows, two morphism rows, and the local image-kernel containment needed for exactness.*
3. *V is the vertical-compatibility row, recording the three displayed vertical morphisms and the two commutative squares that align R_0 with R_1.*
4. *K is the kernel-row ledger for the left and middle vertical maps.*
5. *C is the cokernel-row ledger for the middle and right vertical maps.*
6. *E is the exactness-boundary row, recording that the kernel leg, connecting boundary, and cokernel leg form the displayed exact boundary chain.*
7. *∂ is the connecting-boundary row from the right kernel surface to the left cokernel surface.*
8. *H records componentwise* hsame *transport for the abelian-category source, exact rows, vertical squares, kernel rows, cokernel rows, boundary rows, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from the two exact rows and vertical squares to $K, C, E, \partial, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{SnakeLemma}^{\uparrow}$*,* $\mathsf{AbelianCat}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* $\mathsf{DerivedFunctor}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{SnakeLemma}^\uparrow}$ *row.*

The classifier compares accepted snake-lemma packets only through these rows. It contains no coordinate for host quotient representatives, unbounded diagram search, selected elements outside the displayed histories, or equality of ambient abelian-category objects.

Lean4 ✓ `BEDC.Derived.SnakeLemmaUp.TasteGate.SnakeLemmaTasteGate single carrier alignment`

Theorem 5084.2 (Snake lemma NameCert obligations)**.** *For every accepted* $\mathsf{SnakeLemma}^\uparrow$ *carrier*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^\uparrow}$ *surface consists exactly of abelian-category source exposure through A, upper and lower exact-row exposure through R_0, R_1, vertical-square compatibility through V, kernel-row exposure through K, cokernel-row exposure through C, exact-boundary exposure through E, connecting-boundary exposure through ∂, componentwise transport through H, displayed replay through T, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 5890.1. The rows A, R_0, R_1, V expose the ambient additive kernel-cokernel surface, the two exact rows, and the square compatibility before any boundary read is allowed.

The kernel and cokernel ledgers are then read from K and C. They supply the visible source and target surfaces for the connecting row ∂, while E records the exactness statement tying the kernel leg, boundary leg, and cokernel leg into one chain.

Transport, replay, provenance, and local naming preserve this same finite packet through H, T, P, L. Since no excluded quotient representative, host object equality, or unbounded diagram search appears among the coordinates, the listed rows exhaust the local certificate obligations. □

Theorem 5084.3 (Snake lemma exact boundary)**.** *Every downstream* $\mathsf{Homology}^\uparrow$ *or* $\mathsf{DerivedFunctor}^\uparrow$ *consumer of an accepted* $\mathsf{SnakeLemma}^\uparrow$ *packet reads the boundary route*

$$\ker(v_2) \longmapsto \operatorname{coker}(v_0)$$

only through the displayed chain

$$\mathsf{AbelianCat}^\uparrow \longmapsto (R_0, R_1, V) \longmapsto (K, C) \longmapsto (\partial, E) \longmapsto (H, T, P, L).$$

Thus the consumer receives one finite exact-boundary packet of abelian-category source rows, exact-row ledgers, vertical-square ledgers, kernel and cokernel rows, the connecting-boundary row, exactness-boundary evidence, transport, replay, provenance, and local naming.

Proof. Start with the abelian-category source row A. The two exact rows R_0, R_1 and the vertical-compatibility row V expose exactly the diagram data on which the classical snake-lemma chase depends.

Read K and C next. These rows identify the displayed kernel and cokernel surfaces that the connecting boundary is permitted to connect, without choosing representatives outside the finite histories.

The boundary row ∂ and exactness row E then supply the only route from the right kernel surface to the left cokernel surface. A homology or derived-functor consumer therefore reads the boundary as a named packet, not as an ambient quotient construction.

Finally H, T, P, L keep the same route stable under componentwise history equality, displayed replay, provenance, and local naming. No quotient-object equality, host diagram oracle, or unbounded element chase is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SnakeLemma}^{\uparrow}$ packet is generated from an AbelianCat source row, two exact-row ledgers, vertical-square compatibility rows, kernel-row and cokernel-row ledgers, a connecting-boundary row, an exactness-boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5890.1, Definition 5890.2, and Definition 5890.3 seed the finite snake-lemma packet over abelian-category source rows, exact-row ledgers, vertical-square compatibility, kernel rows, cokernel rows, connecting-boundary rows, exactness-boundary evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-object equality, host abelian-category equality, unbounded diagram search, or ambient element-chase completeness.

Upgrade path: Obligation closure requires checked carrier admission, exact-row exposure, vertical-square compatibility, kernel-row exposure, cokernel-row exposure, connecting-boundary construction, exactness-boundary preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SnakeLemmaUp.

Chapter 5085

A Concrete Naming Certificate for $\mathsf{RealTailDiameterGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet records the finite gauge by which a real-facing consumer reads a regular rational tail only after the displayed tail-diameter, dyadic tolerance, stream-window, and readback rows have been exposed. It sits after the tail-diameter witness of chapter 1746, the dyadic tolerance core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a non-quotient L10 gauge object for Real and Completion consumers; it is not host real equality, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5085.1 (Real tail-diameter gauge BHist source)**.** *A* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, T, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the real-facing boundary that the gauge is allowed to serve.*
2. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row, recording the inspected tail threshold, two visible tail indices, and the displayed diameter bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the same tail-diameter ledger.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite source observations inspected by T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window and tolerance data before any real-facing read is accepted.*
6. *H records componentwise* hsame *transport for the seal, tail-diameter ledger, dyadic tolerance, window, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a real-facing gauge request through E, T, D, W, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealTailDiameterGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, T, D, W, R, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealTailDiameterGaugeUp.TasteGate.RealTailDiameterGaugeTasteGate single carrier alignment`

Theorem 5085.2 (Real tail-diameter gauge NameCert obligation surface)**.** *For every accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier*

$$G = (E, T, D, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*.*

Proof. Project the carrier of Definition 5891.1. The analytic rows are the Real seal E, tail-diameter ledger T, dyadic tolerance D, finite window W, and regular rational readback R.

The gauge obligation is read only through T, D, W, R. The tail-diameter ledger records the visible tail request and bound, the dyadic row supplies the tolerance budget, the stream row exposes the inspected finite windows, and the regular row repacks the same data for rational readback.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Since the carrier has no coordinate for the excluded host or completion objects, the displayed rows exhaust the local NameCert surface. □

Theorem 5085.3 (Real tail-diameter gauge RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *packet factors through*

$$W, D \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real and Completion route may read the finite stream window, dyadic tolerance ledger, tail-diameter gauge, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming rows, but it receives no host real equality, quotient stream equality, selected limit, trichotomy decision, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge check.*

Proof. Start with an exported regular-rational or real-facing read. The only finite observations available to the packet are the $\mathsf{StreamName}^{\uparrow}$ row W and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D.

The tail-diameter ledger T then binds those observations to a displayed tail request and diameter bound. This row is the gauge that prevents a consumer from replacing the finite tail evidence by a quotient stream or a selected limit.

Only after T is displayed does the handoff expose the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row E. The readback and seal are therefore downstream from the same finite gauge data.

Finally H, C, P, N transport, replay, source, and name the route. Since the excluded objects do not occur in the carrier, no consumer can bypass the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet is generated from a RealUp seal row, a CauchyTailDiameterUp ledger, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, seed)

Scope closed: Definition 5891.1, Definition 5891.2, and Definition 5891.3 seed the finite real-facing tail-diameter gauge for regular rational and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, countable choice, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, Real seal confinement, CauchyTailDiameter ledger exactness, DyadicRatCore tolerance readback, StreamName window coverage, RegSeqRat handoff, transport stability, continuation replay, provenance non-escape, and local naming for BEDC.Derived.RealTailDiameterGaugeUp.

This hub routes the EffectiveRealCompactIntervalSearch$^{\uparrow}$ packet through chapter 1250, chapter 1767, and chapter 5899 before the child files expose the compact interval search carrier.

Chapter 5086

A Concrete Naming Certificate for $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet records the finite search surface by which a continuous real-valued map on a displayed closed bounded interval is inspected through compact finite nets, uniform-modulus data, and interval bisection rows before a downstream extremum consumer reads a certified bracket. It sits over the real-name seal of chapter 38, the interval endpoint carrier of chapter 1591, the compact-net selector of chapter 914, the graph and pointwise rows of chapter 130, and the tolerance handoff of chapter 369. The packet gives the constructive extreme-value route a finite searchable carrier; it does not assert classical Heine–Borel compactness, choose a host maximum, or identify quotient real functions.

The search packet consumes the compact interval finite-net route before it exports bisection or extreme-value handoffs. Its selector row P is read downstream of the closed-bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; the bisection ledger then refines those displayed cells instead of introducing an open-cover compactness argument or a selected supremum.

Definition 5086.1 (Effective real compact-interval search carrier)**.** *An* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, U, P, B, V, H, C, Q, N)$$

with the following visible rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint and containment row whose endpoints are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for a real-valued map over the interval source named by I.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance handoff for F on the displayed interval source.*
4. *P is the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row that lists the rational mesh cells and selected centers used by the search.*
5. *B is the bisection ledger, a finite prefix tree of interval halves whose cells refine P at the requested tolerance.*
6. *V is the value-comparison ledger storing the displayed* $\mathsf{Real}^{\uparrow}$ *value rows and rational error budgets attached to the cells of B.*

7. *H stores componentwise* hsame *transport for the interval, continuous-map, uniform-modulus, selector, bisection, value-comparison, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an extremum-search request through $I, F, U, P, B, V, H, Q, N$.*
9. *Q is the* Pkg *provenance over* EffectiveRealCompactIntervalSearch$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *row.*

The classifier compares accepted search carriers only through $I, F, U, P, B, V, H, C, Q, N$. No open-cover compactness theorem, selected host maximum, excluded-middle comparison on all real values, quotient function equality, or ambient completeness coordinate is part of the carrier.

Theorem 5086.2 (Effective real compact-interval search obligations)**.** *For every accepted* EffectiveRealCompactInte *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, value-comparison ledger, transport, replay, provenance, and local naming;*
2. *source agreement, requiring I, F, U, and P to name the same displayed interval source before any cell is searched;*
3. *selector exactness, requiring the search cells in B to refine the finite* ProbeBundle *centers and tolerances exposed by P;*
4. *value-row binding, requiring every value comparison in V to be attached to a cell of B, a displayed graph row of F, and the error budget supplied by U;*
5. *bisection stability, requiring interval refinement to preserve endpoint containment and the selected candidate cell under componentwise* hsame *transport;*
6. *extremum handoff, requiring any lower-bound, upper-bound, near-minimum, or near-maximum read to pass through B, V, H, C, Q, N rather than through a host maximum selector;*
7. *non-escape, excluding open-cover compactness, classical Heine–Borel selection, trichotomy on arbitrary real values, quotient function equality, and ambient* Real$^{\uparrow}$ *completeness from the local packet.*

No effective compact-interval search obligation is stored outside these rows.

Proof. Project the carrier of Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, and the finite selector row is P.

The search may begin only after these rows agree on the same displayed interval source. The bisection ledger B then refines the selector cells supplied by P, so each later comparison is attached to a visible interval cell rather than to an external point choice.

The value ledger V reads graph values from F with the error budget delivered by U. This makes every candidate comparison a finite statement about named cells and rational tolerances.

Finally the extremum-facing read uses H, C, Q, N to replay and name the same route. Since the carrier has no coordinate for open-cover compactness, a host maximum, global real trichotomy, quotient function equality, or ambient completeness, none of those objects can enter the local certificate. □

Theorem 5086.3 (Effective real compact-interval bisection induction)**.** *For an accepted* EffectiveRealCompactInterv *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, every finite bisection stage in B is obtained by induction from the endpoint row I and the selector row P. The induction step replaces one displayed interval cell by its two named halves, transports the corresponding* ContinuousMap$^{\uparrow}$ *and* UniformModulus$^{\uparrow}$ *reads through H, and extends V only by value rows attached to those halves.*

Proof. The base stage is the interval cell named by I together with the first finite-net read in P. Its value ledger is the corresponding displayed graph read from F with the tolerance supplied by U.

For the induction step, choose the cell already selected by the finite ledger. The midpoint and half-interval endpoints are read as interval rows below I, and the selector exactness obligation identifies their centers with the relevant ProbeBundle entries of P.

Transport along H keeps the continuous-map graph and uniform-modulus handoff on the same source. The new value rows are therefore added to V only for the two displayed halves, so the induction never leaves the finite bisection tree. □

Theorem 5086.4 (Effective real compact-interval extreme-value handoff)**.** *Every extremum-search consumer reading an accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *carrier factors through*

$$I, F, U, P \longmapsto B, V \longmapsto H, C, Q, N.$$

The consumer receives a finite bracket or near-extremal cell family whose error budget is supplied by UniformModulus$^{\uparrow}$ *and whose candidates are selected by* CompactNetModulusSelector$^{\uparrow}$*. It receives no classical maximum point, open-cover compactness proof, quotient equality of functions, global trichotomy on* Real$^{\uparrow}$ *values, or ambient completeness assertion.*

Proof. Begin with the carrier projection. The only source data available to the consumer are the interval row I, continuous graph row F, uniform-modulus row U, and compact-net selector row P.

Apply Definition 5892.3. Every searched cell and every candidate value is produced inside the finite bisection ledger B and value ledger V.

The final handoff reads those ledgers through H, C, Q, N. Applying Definition 5892.2, any route that asks for a host maximum, open-cover compactness, quotient function equality, global real trichotomy, or ambient completeness is outside the accepted packet. □

Theorem 5086.5 (Effective real compact interval search finite-net handoff)**.** *Every accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *search packet reads a compact interval finite-net certificate, a located interval bisection row, and the* Real$^{\uparrow}$ *comparison seal before exporting the extreme-value handoff. The route passes through the closed bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; it exports no open-cover compactness, selected supremum, host trichotomy, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The compact interval finite-net data enter through the selector row P, whose cells are read only after the displayed closed bounded interval source I is fixed.

Apply Definition 5892.3. The bisection row B refines the finite cells supplied by P, and every value comparison in V is attached to a displayed graph row of F with the tolerance budget from U.

Apply Definition 5892.4. The final route reads B, V, H, C, Q, N after the finite-net source rows, so the exported extreme-value handoff is a finite search certificate rather than an open-cover compactness, supremum-selection, trichotomy, quotient-equality, or completeness principle. □

Theorem 5086.6 (Effective real compact interval search L10 sibling dependency)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must display the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint carrier, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite net, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *selector, the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *row, and the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement before exporting any extremum handoff. The route is the finite-net and bisection dependency route of Definition 5892.5, not an isolated kernel-only search packet.*

Proof. Project the carrier definition, Definition 5892.1. The rows I, F, U, P, B, V expose the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, and value ledger, while the Real-facing seal is read through the displayed interval and value rows.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before any bisection or extreme-value consumer can read the packet.

The bisection row B then supplies the located refinement consumed by the extremum handoff. The structural rows H, C, Q, N only transport, replay, source, and name that same sibling-dependent route, so no extremum handoff is exported before the listed L10 siblings are displayed. □

Theorem 5086.7 (Effective real compact interval search inverse modulus bisection)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, a displayed* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row* U *and a* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row* P *determine a finite bisection depth in* B *whose dyadic cell diameter is below the requested tolerance. The depth is read only through* P, B, V, H, C, Q, N*: the value ledger* V *attaches comparisons to cells at that depth,* H *transports the same cells under* hsame*,* C *replays the finite request, and* Q, N *package and name the route. The construction exports no open-cover compactness proof, no host maximum, no quotient equality of functions, no trichotomy on* $\mathsf{Real}^{\uparrow}$ *values, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the accepted search carrier. The uniform-modulus row U supplies the requested tolerance, while the selector row P displays the finite mesh data that the search is allowed to refine.

Apply Definition 5892.3. Each step of B replaces a displayed cell by named dyadic halves, so repeated refinement gives a finite depth at which the visible cell diameter is below the tolerance requested by U.

Apply Definition 5892.5. The finite-net row P remains the source of the cells refined by B, and the value ledger V records only comparisons attached to those cells.

Finally H, C, Q, N transport, replay, source, and name the same finite depth-control route. Because the carrier has no open-cover compactness coordinate, host maximum selector, quotient function equality coordinate, trichotomy oracle, or ambient completeness row, the inverse-modulus read remains finite interval-search data. □

Theorem 5086.8 (Effective real compact interval search finite-grid stability)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, the displayed finite-net cell selected by* P *and refined by the located bisection ledger* B *is preserved under componentwise* hsame *transport before any extreme-value handoff is exported. Thus every Real-facing extremum consumer reads the same finite cell, value row, and bisection row through* P, B, V, H, C, Q, N*, and receives no open-cover compactness proof, selected host maximum, or* $\mathsf{Real}^{\uparrow}$ *trichotomy.*

Proof. Project the carrier of Definition 5892.1. The finite-net cell is displayed by P, its bisection refinement is recorded in B, and the attached value comparisons are stored in V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger is consumed.

Apply the inverse-modulus bisection theorem, Definition 5892.7. The bisection depth and the cell read remain tied to P, B, V, while H transports the same displayed rows and C replays the same finite request.

The carrier has no coordinate for open-cover compactness, host maxima, quotient function equality, Real trichotomy, or ambient completeness. The exported extreme-value handoff therefore preserves only the finite-grid cell and located bisection row named by the packet. □

Theorem 5086.9 (Effective real compact interval search Bishop cover link)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$ *reads the Bishop interval-cover finite cover ledger of chapter 5124 before it exports any bisection or extreme-value handoff. The Bishop route supplies the finite interval-cover cells refined by* P *and* B*, while the local search packet preserves the continuous-map, uniform-modulus, value-comparison, transport, replay, provenance, and naming rows. The exported route contains no open-cover compactness theorem, selected supremum, host trichotomy, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, the finite selector row is P, the bisection ledger is B, and the value ledger is V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search packet to read the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger can be consumed.

Apply the grid-stability theorem, Definition 5892.8. The selected finite cell is preserved through P, B, V, H, C, Q, N, so the Bishop interval-cover ledger supplies interval-cover cells for the same displayed route rather than a separate compactness principle.

The remaining rows H, C, Q, N transport, replay, source, and name the same finite cover-to-search handoff. Since the carrier has no row for open-cover compactness, selected suprema, host trichotomy, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot be exported by the bisection or extreme-value consumer. □

Theorem 5086.10 (Effective real compact interval search RegularCauchyGluing dependency)**.** *Every completion-facing extremum-search read of an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must expose the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route of chapter 5899 after the finite bisection and value ledger are displayed and before any* $\mathsf{Real}^{\uparrow}$*-facing seal is consumed. The dependency is read through the finite-net handoff of Definition 5892.5 and the grid-stability route of Definition 5892.8; it is not supplied by quotient stream equality, selected host limits, countable choice, or ambient real completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The completion-facing rows available to the search are the bisection ledger B, the value-comparison ledger V, and the structural replay rows H, C, Q, N.

Apply Definition 5892.5. The finite-net route fixes the displayed interval cells and their Real-valued comparison windows before an extremum consumer can read the packet.

Apply Definition 5892.8. The same finite cell, bisection row, and value row are preserved under componentwise hsame transport, so the completion-facing read has a stable regular-Cauchy window to hand to the gluing sibling.

Route that window through $\mathsf{RegularCauchyGluing}^{\uparrow}$ before the terminal Real-facing seal is consumed. Since the carrier has no quotient stream-equality row, selected host limit, countable-choice

selector, or ambient-completeness coordinate, the gluing dependency is the only completion-facing route available after B and V are displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5892.1, Definition 5892.2, Definition 5892.3, Definition 5892.4, Definition 5892.5, Definition 5892.7, and Definition 5892.8 seed the finite interval-search carrier from displayed Real, closed-bounded-interval, compact-net-selector, continuous-map, finite-net, and uniform-modulus rows to extremum-facing bracket and depth-control handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical Heine–Borel compactness, selected host maxima or minima, quotient equality of functions, global real trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, selector exactness, value-row binding, bisection stability, extremum handoff, and non-escape for BEDC.Derived.EffectiveRealCompactIntervalSearchUp.

Constructive story: An $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source, ContinuousMapUp graph rows, a UniformModulusUp tolerance handoff, a CompactNetModulusSelectorUp finite-net row, a finite bisection ledger, displayed RealUp value comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ search packet consumes the closed-bounded interval net of chapter 1250, the finite epsilon-net surface of chapter 1767, the compact-net modulus selector of chapter 914, and the located interval bisection route of chapter 1711 before exporting its extremum handoff. Thus a downstream consumer sees the finite-net and bisection dependencies as displayed route data, not as an open-cover compactness proof or a selected host extremum. The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ route hands its finite bisection rows, interval-value rows, and compact-window certificates to the compatible-window route of chapter 5899 before any $\mathsf{Real}^{\uparrow}$ seal consumer reads a glued local search window. This dependency keeps each L10 finite-window consumer downstream from the displayed compact search packet, so the gluing step reads certified local windows rather than an ambient completeness or choice principle.

Chapter 5087

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on dense embeddings, the $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet records the finite dense-source route by which a Cauchy completion consumer reads a source object inside its completion boundary. It sits beside the dense uniform embedding surface of chapter 1651, the functorial Cauchy-completion route of chapter 1937, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies dense embedding data for completion consumers without asserting quotient completion equality, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5087.1 (Cauchy completion dense-embedding carrier)**.** *A* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the dense source row. It records the source-side Cauchy observations that a completion consumer may approximate by displayed source data.*
2. *G is the embedding graph row. It records the displayed source-to-completion graph and the finite image-window reads used by dense embedding consumers.*
3. *K is the Cauchy completion row, read through the completion-facing route associated with* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite request windows inspected by the source and completion sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source-image windows with completion windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after A, G, K, S, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the dense source, embedding graph, completion, window, tolerance, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dense-source request through $A, G, K, S, D, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseEmbeddingUp.CauchyCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 5087.2 (Cauchy completion dense-embedding NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier*

$$J = (A, G, K, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the dense source row, embedding graph row, Cauchy completion row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing dense read must expose A, G, K, S, D, Q before E, H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5893.1. The dense source, embedding graph, Cauchy completion row, finite windows, tolerance ledger, regular readback, and Real seal are exactly the displayed analytic coordinates.

The dense-source route is read through A, G, K. A completion consumer first fixes the source row, then the embedding graph row, and only then the completion-facing row associated with the Cauchy completion route.

The finite observation route is read through S, D, Q. The stream windows expose inspected requests, the dyadic ledger supplies the tolerance budget, and the regular rational readback repacks the same observations before any Real-facing seal is visible.

Finally E, H, C, P, N preserve and name the route. Since quotient completion equality, selected representatives, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5087.3 (Cauchy completion dense-embedding Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer exported by an accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{DenseUniformEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionFunctoriality}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More precisely, the consumer reads the dense source row A, embedding graph G, Cauchy completion row K, finite windows S, tolerance ledger D, regular readback Q, terminal seal E, componentwise transport H, displayed replay C, provenance P, and local name N before the real-facing boundary is exposed.

Proof. Start with an exported Real-facing or completion-facing read. The NameCert obligations of Definition 5893.2 force the read through A, G, K before the finite observation rows can be consumed.

Read the sibling routes next. The dense source is the $\mathsf{DenseUniformEmbedding}^{\uparrow}$ surface, the completed-map side is routed through $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, and the finite observations are the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows displayed by the carrier.

Only after those rows have been displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal row E be read. The structural rows H, C, P, N transport, replay, source, and name the same route, so the handoff exports no quotient completion equality, selected representative, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a dense source row, embedding graph row, Cauchy completion row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5893.1, Definition 5893.2, and Definition 5893.3 seed the dense-source route for Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense source exposure, embedding graph exactness, Cauchy completion handoff, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionDenseEmbeddingUp.

Chapter 5088

A Concrete Naming Certificate for $\mathsf{RiemannRefinementCriterion}^{\uparrow}$

Following Bishop and Bridges on Riemann–Darboux integration, the $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet records the finite refinement criterion that lets a tagged Riemann-sum consumer compare upper Darboux rows against a gauge-local partition route. It sits after the tagged partition carrier of chapter 4296, the upper Darboux-sum carrier of chapter 4083, the Riemann sum gauge packet of chapter 4115, and the Cousin cover carrier of chapter 3426. The packet supplies the finite refinement surface consumed by integrability criteria without importing host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5088.1 (Riemann refinement criterion carrier)**.** *A* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row shared by the refinement comparison.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *source row, including the finite cell order, tag containment, and mesh exactness visible before refinement is read.*
3. *U is the* ***??*** *row whose upper-sum ledger is compared against the refined tagged route.*
4. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row, carrying the positive rational gauge, mesh ledger, and finite tagged-sum handoff.*
5. *C is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge-cover row supplying the finite local cover route that admits the refined cells.*
6. *M is the common-refinement mesh row, listing the displayed refinement map from the tagged partition row into the gauge-cover and upper-sum rows.*
7. *D is the Darboux-gap lock row, recording the finite upper-bound comparison that remains visible after the common refinement has been read.*
8. *W is the window and readback row for interval endpoints, tags, gauges, cover cells, and upper-sum values.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tagged partition, gauge, Cousin cover, refinement mesh, and Darboux-gap rows are displayed.*

10. *H records componentwise* hsame *transport for interval, partition, upper-sum, gauge, cover, refinement, Darboux-gap, window, seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a refinement request through $I, T, U, G, C, M, D, W, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, *??*, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows. It has no coordinate for host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannRefinementCriterionUp.RiemannRefinementCriterionTasteGate single carrier alignment`

Theorem 5088.2 (Riemann refinement criterion NameCert obligations)**.** *For every accepted* RiemannRefinementCr *carrier $R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval, tagged partition, upper Darboux-sum, Riemann sum gauge, Cousin cover, refinement mesh, Darboux-gap lock, window, seal, transport, replay, provenance, and local naming rows.*

Every refinement-facing consumer must read the tagged partition row T from chapter 4296, the upper-sum row U from chapter 4083, the gauge row G from chapter 4115, and the gauge-cover row C from chapter 3426 before the common-refinement mesh row M and Darboux-gap lock row D are available.

Proof. Project the carrier of Definition 5894.1. The tuple displays the only rows available to a refinement consumer: interval source, tagged partition, upper-sum, gauge, Cousin cover, common refinement, Darboux-gap lock, finite windows, terminal seal, transport, replay, provenance, and local naming.

The source rows are T, U, G, C. The tagged partition supplies finite cells and tags, the upper Darboux row supplies the upper-bound ledger, the Riemann sum gauge row supplies the gauge mesh, and the Cousin cover row supplies local gauge cover cells.

The refinement rows are M, D. The mesh row M binds those four source rows into one common finite refinement before the Darboux-gap lock D can be read.

Finally W, E, H, K, P, N preserve the same displayed route through finite windows, the terminal Real seal, transport, replay, provenance, and local naming. Since the carrier has no coordinate for the refused host objects, the listed rows exhaust the local NameCert surface. □

Theorem 5088.3 (Riemann refinement criterion Darboux gap lock)**.** *In every accepted* RiemannRefinementCriterior *packet, the Darboux-gap row D is readable only after the common-refinement mesh row M has already displayed the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover sources. Thus a Darboux or Riemann-integrability consumer cannot replace the refinement route by host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the source rows from Definition 5894.1. The Darboux-gap row is not listed as an independent analytic conclusion; it appears after the tagged partition, upper-sum, gauge, cover, and common-refinement rows.

Apply Definition 5894.2. Its obligation surface requires T, U, G, C to be exposed before M, and M before D.

The structural rows only transport and replay the same finite packet. Therefore the Darboux-gap read is locked to the displayed refinement route and cannot be promoted to any refused host-level principle. □

Theorem 5088.4 (Riemann refinement criterion integral handoff)**.** *Every Riemann-integrability or gauge-integral consumer of an accepted* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *packet receives only the finite handoff*

$$\mathsf{TaggedRealPartition}^{\uparrow},\ \textit{??},\ \mathsf{RiemannSumGauge}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow} \longmapsto M \longmapsto D \longmapsto W, E, H, K, P, N.$$

The handoff exports a refinement criterion over displayed finite rows, not an ambient theorem about all host partitions or all mesh limits.

Proof. Start with a consumer read. The packet first exposes the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover rows through the sibling chapters cited in the carrier.

The common-refinement mesh row M then aligns those sources. The Darboux-gap lock row D is read only along that aligned route, and the window and seal rows W, E expose the Real-facing observations after the finite rows are visible.

Transport, replay, provenance, and local naming through H, K, P, N preserve the same route. No carrier coordinate supplies host partition equality, selected limiting meshes, choice, compactness, quotient function equality, or ambient completeness, so the integral handoff is exactly the displayed finite criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet is generated from a closed bounded interval row, TaggedRealPartition source, UpperDarbouxSum ledger, RiemannSumGauge mesh row, CousinCover gauge-cover row, common-refinement mesh row, Darboux-gap lock, finite windows, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5894.1, Definition 5894.2, Definition 5894.3, and Definition 5894.4 seed the finite refinement criterion between tagged partitions, upper Darboux sums, Riemann sum gauges, and Cousin covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition source lock, upper-sum handoff, gauge mesh exactness, Cousin cover locality, Darboux-gap lock, integral handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRefinementCriterionUp.

Chapter 5089

A Concrete Naming Certificate for CompactUniformModulusComposition$^{\uparrow}$

Following the Bishop–Bridges and Heine–Cantor treatment of uniformly continuous maps on compact metric data, the CompactUniformModulusComposition$^{\uparrow}$ packet records the finite tolerance-only composition surface for two compact-uniform modulus rows. It sits after the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the compact uniform modulus row of chapter 2299, the metric-triangle estimate of chapter 1159, and the stability packet of chapter 5872. The packet supplies the finite composition object used by compact-open and compact-uniform-continuity consumers without exporting a host function space, quotient point equality, selected global modulus, choice schedule, or ambient compactness theorem.

Definition 5089.1 (Compact uniform modulus composition carrier)**.** *A* CompactUniformModulusComposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are compact metric source, middle, and target rows, with finite net and distance ledgers visible at the requested tolerances.*
2. *F and G are* ContinuousMap$^{\uparrow}$ *rows for the two displayed maps.*
3. *M_F is the first* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Y into a source tolerance for X along F.*
4. *M_G is the second* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Z into a source tolerance for Y along G.*
5. *A is the tolerance-assembly row, recording that a Z-tolerance request is first read by M_G and that its resulting Y-tolerance request is then read by M_F.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that keeps the two transported estimates on the same metric route.*
7. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row for the composite map.*
8. *H records componentwise* hsame *transport for compact metric sources, continuous maps, modulus rows, tolerance assembly, triangle estimate, consumer, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a composite tolerance request through $Z, M_G, Y, M_F, X, A, T, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *row.*

The classifier compares accepted composition packets only by these finite rows. It contains no host function-space composition theorem, quotient point equality, selected global modulus, choice schedule, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusCompositionUp.CompactUniformModulusCompositionTasteGate`
`single carrier alignment`

Theorem 5089.2 (Compact uniform modulus composition NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *carrier $Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *surface consists exactly of compact source admission through X, Y, Z, continuous-map admission through F, G, first and second compact-uniform modulus reads through M_F, M_G, tolerance assembly through A, metric-triangle transport through T, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5895.1. The compact metric rows, continuous-map rows, compact-uniform modulus rows, assembly row, triangle estimate, uniform-modulus consumer, and structural rows are exactly the displayed coordinates.

The tolerance route is ordered by A. A requested Z-tolerance is consumed by M_G to expose a Y-tolerance, and only that displayed Y-tolerance is then consumed by M_F to expose the source tolerance over X.

The triangle estimate T and consumer row U read the same assembled route. Transport, replay, provenance, and local naming preserve it through H, C, P, N.

Since the carrier has no coordinate for host function-space composition, quotient point equality, selected moduli, choice, or ambient compactness, the listed rows exhaust the NameCert surface. □

Theorem 5089.3 (Compact uniform modulus composition tolerance route)**.** *Every compact-open or compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *packet reads the composite modulus only through*

$$M_G \longmapsto A \longmapsto M_F \longmapsto T \longmapsto U.$$

Thus the composite tolerance is assembled from two displayed compact-uniform modulus rows and the metric-triangle ledger, not from a selected host-level modulus for the composite map.

Proof. Begin with the consumer's target tolerance for the Z row. The second modulus row M_G supplies the middle-space tolerance, and the assembly row A records that this is the only tolerance handed to M_F.

The first modulus row M_F returns the source-space tolerance over X. The metric-triangle row T then transports the two displayed estimates into the bound consumed by U.

The structural rows H, C, P, N preserve this route. Since there is no host composition coordinate or selected global modulus in the carrier, every consumer remains on the displayed tolerance chain. □

Theorem 5089.4 (Compact uniform modulus composition nonescape)**.** *No accepted* CompactUniformModulusComp *packet exports quotient point equality, a selected global modulus, a choice schedule, an ambient compactness theorem, or a host function-space composition proof. It exports only the two continuous-map rows, two compact-uniform modulus rows, the tolerance assembly row, the metric-triangle estimate, the uniform-modulus consumer, and the structural NameCert rows.*

Proof. Project the carrier. The only analytic rows are F, G, M_F, M_G, A, T, U over the compact metric sources X, Y, Z.

Apply Definition 5895.2. The local NameCert obligations require the consumer to read the two modulus rows and the tolerance assembly before the uniform-modulus export.

The refused host objects are absent from the carrier and cannot be produced by transport, replay, provenance, or local naming. Therefore the accepted packet exports only the displayed finite composition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ packet is generated from compact metric source, middle, and target rows, two ContinuousMap rows, two CompactUniformModulus rows, a tolerance-assembly row, a MetricTriangleUniformEstimate row, a UniformModulus consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5895.1, Definition 5895.2, Definition 5895.3, and Definition 5895.4 seed the finite tolerance-only composition route for compact-uniform modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space composition, quotient point equality, selected global moduli, choice schedules, ambient compactness theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, continuous-map admission, first and second modulus reads, tolerance assembly, metric-triangle transport, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulusCompositionUp.

Chapter 5090

A Concrete Naming Certificate for $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$

The $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet realizes the private/shared decoration theorem of Definition 9462.1705 inside the boundary-tile source of subsection 9462.28.21. It sits beside the DNA boundary-tile packet of chapter 3393, the trigger-threshold packet of chapter 4217, and the finite trigger algebra route of chapter 3478. The packet records private blocks, shared rows, lift counts, and defect ledgers as finite NameCert data; it is not a biological mechanism, stochastic evolution model, hidden enumeration oracle, or quotient codon semantics.

Definition 5090.1 (PrivateSharedDecorationModel carrier)**.** *A* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, L, D, T, R, P, N)$$

with the following displayed rows:

1. *B is the nonempty private-block row, recording the partition of coordinates that remain private to a single decorated support.*
2. *S is the shared-row ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *L is the lift-count row, recording the finite ℓ^m lift factor attached to displayed support blocks.*
4. *D is the defect ledger, recording the row-set size contribution to the defect-refined polynomial.*
5. *T records componentwise* hsame *transport for B, S, L, D, R, P, N.*
6. *R records displayed* Cont *replay from a decorated trigger request through the private block, shared rows, lift count, defect ledger, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem of Definition 9462.1705,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, S, L, D, T, R, P, N. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, or an unrecorded enumeration oracle.

Theorem 5090.2 (PrivateSharedDecorationModel NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier*

$$M = (B, S, L, D, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise transport, displayed replay, provenance, and local naming. A decorated trigger read is accepted only after it exposes a nonempty private block and assigns every shared coordinate to at least two rows.*

Proof. Project the carrier of Definition 5896.1. The private block, shared-row ledger, lift count, defect ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The private/shared constraint is read through B, S. The row B must display a nonempty private partition, and the ledger S must attach every shared coordinate to a row set of size at least two.

The enumeration and defect data are read through L, D. The lift-count row records the finite lift factor, while the defect ledger records the shared-row contribution used by the source theorem.

Finally T, R, P, N transport, replay, source, and name the same route. Since biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, and unrecorded enumeration have no carrier coordinate, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: If a displayed decorated trigger lacks a nonempty private block or assigns a shared coordinate to fewer than two rows, it is rejected by the local NameCert.

Independence witness: dna-window-six-boundary-tile, dna-window-six-trigger-threshold, dna-window-six-translation-boundary: $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries private-block rows, shared-row ledgers, lift-count rows, defect ledgers, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.PrivateSharedDecorationModelUp` must provide explicit $\mathsf{Nontrivial}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ instances preserving B, S, L, D, T, R, P, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet is generated from a nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5896.1 and Definition 5896.2 seed the finite private/shared decoration packet over DNA boundary-tile trigger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global semantics, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block nonemptiness, shared-row arity, lift-count exposure, defect-ledger exposure, transport, replay, provenance, local naming, Nontrivial PrivateSharedDecorationModelUp, and FieldFaithful PrivateSharedDecorationModelUp for BEDC.Derived.PrivateSharedDecorationModelUp.

Chapter 5091

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet records the finite filter-facing route by which located real brackets are read through dyadic tolerances, stream windows, regular rational readback, and the terminal real seal. It consumes the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located bracket surface of chapter 1709, and the selector boundary of chapter 5886. The packet gives the L10 Real and Completion route a located filter interface without asserting quotient filter equality, host real equality, selected limit points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5091.1 (Bishop located real filter carrier)**.** *A* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier is a finite packet*

$$S = (F, L, D, W, R, E, B, H, T, P, N)$$

with the following displayed rows:

1. *F is the filter-base row, exposing the finite base request, refinement evidence, and Cauchy-facing filter data that a later completion consumer may inspect.*
2. *L is the located-real bracket row, recording lower and upper bracket reads over* $\mathsf{LocatedRealInterval}^{\uparrow}$ *together with the displayed apartness and containment data used to locate the requested real value.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing bracket width, filter-base comparison, stream-window radius, and regular-rational readback.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the filter request and the located bracket.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W under the tolerance ledger D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after F, L, D, W, R have been displayed and replayed.*
7. *B is the* BHist *history row recording the finite route from the filter request through the located bracket, tolerance ledger, stream windows, regular readback, and Real seal.*
8. *H records componentwise* hsame *transport for $F, L, D, W, R, E, B, T, P, N$.*
9. *T records displayed* Cont *replay from every located filter request through $F, L, D, W, R, E, B, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, D, W, R, E, B, H, T, P, N$. It has no coordinate for quotient filter equality, host selected representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BishopLocatedRealFilterUp.BishopLocatedRealFilterUp`

Theorem 5091.2 (Bishop located real filter NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier $S = (F, L, D, W, R, E, B, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the filter base, located-real bracket, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, history, transport, replay, provenance, and local naming row;*
2. *bracket exactness, requiring each located-real bracket read in L to be witnessed by displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint and containment data;*
3. *filter-base stability, requiring refinements of F to preserve the same located bracket and tolerance ledger through* BHist *and* hsame *transport;*
4. *dyadic ledger stability, requiring D to control the bracket width, stream-window radius, and regular-rational readback without adding an external modulus;*
5. *StreamName and RegSeqRat handoff, requiring every numeric read to pass through W and then R before the terminal seal E can be consumed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, selected limiting points, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5897.1. The accepted packet displays the filter base, located bracket, dyadic ledger, stream windows, regular readback, Real seal, history, transport, replay, provenance, and local naming rows.

Bracket exactness is row-local. The bracket row L can expose only the endpoint and containment data already carried by $\mathsf{LocatedRealInterval}^{\uparrow}$, so it cannot identify a real by host order equality or by a selected representative.

The filter-base and tolerance rows then fix the route. Refinement of F is transported through B and H, while the same dyadic row D governs bracket width, window radius, and readback precision.

Finally the numeric handoff is ordered by W, R, E. Stream windows are read before regular rational data, and regular rational data are read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5091.3 (Bishop located real filter L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *packet $S = (F, L, D, W, R, E, B, H, T, P, N)$ must read the finite route*

$$F, L, D, W, R \longmapsto E \longmapsto B, H, T, P, N$$

before it treats the request as Real-facing or completion-facing data. Thus the packet supplies a filter-based located-real handoff to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, *and*

$\mathsf{CauchyFilterSelector}^{\uparrow}$ *consumers without asserting ambient real completeness or quotient equality of filters.*

Proof. Read the carrier rows in order. A consumer first receives the filter base F and the located bracket L, while the dyadic ledger D supplies the common tolerance budget.

The numerical content is then exposed through W and R. The $\mathsf{StreamName}^{\uparrow}$ row records finite observation windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational readback extracted from those windows.

Only after these rows are displayed can the Real seal E be consumed. The history, transport, replay, provenance, and local naming rows B, H, T, P, N preserve that order for later L10 Real and Completion routes.

The carrier has no quotient-filter equality, selected representative, host real equality, choice schedule, or ambient completeness coordinate. Therefore every downstream handoff factors through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet is generated from a displayed filter base, a located-real bracket, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a Real seal, BHist history, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5897.1, Definition 5897.2, and Definition 5897.3 seed the finite filter-facing located-real interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, ambient Real completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, bracket exactness, filter-base stability, dyadic ledger stability, StreamName and RegSeqRat handoff, and Real seal boundary for BEDC.Derived.BishopLocatedRealFilterUp.

Chapter 5092

A Concrete Naming Certificate for $\mathsf{TaylorTheorem}^{\uparrow}$

Following Taylor and the validated-numerics treatment of finite Taylor models, the $\mathsf{TaylorTheorem}^{\uparrow}$ packet records the finite theorem-facing route from displayed derivative data to a polynomial jet and a checked remainder envelope. It sits after the Taylor-model enclosure surface of chapter 442, the derivative observation row of chapter 122, the polynomial carrier of chapter 50, the dyadic rational arithmetic core of chapter 291, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Taylor theorem certificate for one displayed center, order, domain window, derivative ledger, polynomial jet, and remainder bound; it does not import analytic convergence, quotient real equality, selected limits, or a host Taylor series.

Definition 5092.1 (Taylor theorem carrier)**.** *A* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, J, P, R, E, Q, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row naming the center, finite order, domain window, and derivative observations inspected by the packet.*
2. *J is the finite jet ledger. Each coefficient is read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic and the same derivative window named by D.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *row assembled from the displayed finite jet ledger.*
4. *R is the finite remainder-envelope row. It records the derivative-order budget, dyadic interval or ball enclosure, and endpoint window used to bound the residual after P is read.*
5. *E is the* $\mathsf{TaylorModel}^{\uparrow}$ *enclosure row that packages the displayed polynomial part and remainder envelope as a validated finite jet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for polynomial values, derivative observations, and the remainder bound.*
7. *H records componentwise* hsame *transport for derivative, jet, polynomial, remainder, Taylor-model, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every Taylor-theorem read through $D, J, P, R, E, Q, H, G, N$.*

9. *G is the* Pkg *provenance over* $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, P, R, E, Q, H, C, G, N$. It has no coordinate for an infinite Taylor series, analytic convergence, quotient equality of real names, selected limiting values, or an ambient smooth-function theorem.

Theorem 5092.2 (Taylor theorem NameCert obligations)**.** *For every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier*

$$T = (D, J, P, R, E, Q, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed derivative row, finite jet ledger, polynomial row, remainder envelope, Taylor-model enclosure, readback route, transport, replay, provenance, and local naming rows;*
2. *derivative-to-jet routing, requiring every coefficient in J to be read from the finite* $\mathsf{Derivative}^{\uparrow}$ *observations in D and through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic;*
3. *polynomial assembly, requiring P to use exactly the displayed finite coefficient rows in J;*
4. *remainder-envelope exposure, requiring R to bind the next derivative-order budget, the domain window, and the dyadic enclosure before any error bound is exported;*
5. *Taylor-model handoff, requiring E to package only the displayed P and R rows;*
6. *regular and real readback, requiring Q to pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
7. *non-escape from analytic convergence, quotient real equality, selected limits, and host Taylor series semantics.*

No $\mathsf{TaylorTheorem}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.TaylorTheoremUp.TasteGate.TaylorTheoremTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5898.1. The theorem-facing data are exactly D, J, P, R, E, Q, and the structural rows are exactly H, C, G, N.

The derivative-to-jet route is finite. Each coefficient read is a displayed entry of J, and the definition of J attaches that entry to the derivative row D and to dyadic rational arithmetic through $\mathsf{DyadicRatCore}^{\uparrow}$.

Polynomial assembly and remainder exposure are then separate rows. The polynomial row P consumes only the coefficients in J, while the remainder row R consumes the derivative-order budget and domain window before any bound is exported.

Finally the Taylor-model and readback rows seal the route. The row E packages the polynomial and remainder as a finite Taylor-model enclosure, and Q sends the same displayed data through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Since no carrier coordinate names a host Taylor series or a selected limit, the listed rows exhaust the local NameCert surface. □

Theorem 5092.3 (Taylor theorem finite remainder envelope)**.** *Every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier exports a remainder claim only through the displayed route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{TaylorModel}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the derivative row D, finite jet ledger J, polynomial row P, remainder-envelope row R, Taylor-model enclosure E, and regular-rational and real readback row Q before it may cite an error bound for the displayed polynomial approximation. The exported envelope is not an analytic convergence theorem, a quotient equality of real functions, a selected limit, or an ambient Taylor-series object.

Proof. Start with an accepted carrier $T = (D, J, P, R, E, Q, H, C, G, N)$. The only source of derivative information is D, and the only coefficient ledger is the finite row J read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next assemble the approximation. The polynomial row P is built from J, and the remainder row R binds the derivative-order budget, domain window, and dyadic enclosure that measure the residual after that polynomial row has been fixed.

Then pass through the enclosure and readback rows. The Taylor-model row E packages the polynomial and remainder as one finite validated jet, while Q routes the resulting value and bound through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and local naming rows preserve exactly this route. Because the carrier contains no coordinate for analytic convergence, quotient function equality, selected limits, or host Taylor series semantics, every exported remainder envelope is exhausted by the displayed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{TaylorTheorem}^{\uparrow}$ remainder-envelope read factors through a DerivativeUp row, a DyadicRatCore coefficient ledger, a PolynomialUp row, a TaylorModelUp enclosure, a RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: taylormodel, derivative, polynomial, dyadicratcore, regseqrat, real: $\mathsf{TaylorTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite derivative observations, coefficient arithmetic, polynomial assembly, a theorem-facing remainder envelope, Taylor-model enclosure, regular-rational readback, real sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TaylorTheorem}^{\uparrow}$ packet is generated from a DerivativeUp row, a finite DyadicRatCore coefficient ledger, a PolynomialUp row, a finite remainder envelope, a TaylorModelUp enclosure, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5898.1, Definition 5898.2, and Definition 5898.3 seed the finite Taylor theorem route from derivative observations through polynomial jets to displayed remainder envelopes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic convergence, quotient equality of real names, selected limits, host Taylor series semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-jet routing, polynomial assembly, finite remainder-envelope exposure, Taylor-model handoff, regular and real readback, and non-escape for BEDC.Derived.TaylorTheoremUp.

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, and the terminal Real seal of chapter 38 before routing the child files below.

Chapter 5093

A Concrete Naming Certificate for $\mathsf{RegularCauchyGluing}^{\uparrow}$

Bishop–Bridges style regular Cauchy gluing records the finite route by which two compatible local regular rational windows can be read as one BEDC real-name boundary without passing through quotient representatives or an ambient completeness theorem. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the reindexing route of chapter 1568.

Definition 5093.1 (Regular Cauchy gluing carrier)**.** *A* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right compatible regular Cauchy windows, each carrying finite* $\mathsf{StreamName}^{\uparrow}$ *observations and a local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger recording the shared dyadic radii, endpoint comparisons, overlap budgets, and refinement witnesses used by both windows.*
3. *S is the joint* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing exactly the finite indices inspected on the left window, right window, and overlap.*
4. *Q is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by reading the two local regular rows through the common ledger D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, R, D, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for the two windows, common dyadic ledger, finite observations, glued readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite gluing route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the two local windows are compatible on the displayed overlap through D, S, Q, H, C, P, N. It has no coordinate for quotient sequence equality, selected representatives, host real equality, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5093.2 (Regular Cauchy gluing classifier)**.** *The* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *classifier compares accepted carriers only through the tuple* $L, R, D, S, Q, E, H, C, P, N$*. Two packets are indistinguishable for this classifier precisely when the left and right window rows, the common dyadic refinement ledger, the finite observation row, the glued regular readback, the terminal real seal, and the transport, replay, provenance, and naming rows are componentwise the same under the displayed* hsame *routes.*

Theorem 5093.3 (Regular Cauchy gluing NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *surface consists exactly of carrier admission for the two compatible regular Cauchy windows, common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through* S*, glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The analytic rows before the seal are exactly the two regular windows L, R, the common dyadic ledger D, the finite observation row S, and the glued regular readback Q.

The overlap obligation is finite. Its comparisons are stored in D and read through the same inspected indices in S, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ rows feed one glued readback row Q without adding a quotient representative.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections gives the listed $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ surface, and the carrier definition leaves no hidden row for selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 5093.4 (Regular Cauchy gluing overlap stability)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Any refinement of the displayed overlap that is transported by* H *and replayed by* C *preserves the same glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Consequently downstream Real and Completion consumers receive only the finite gluing packet and not quotient stream equality, selected representatives, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the displayed overlap ledger. The common dyadic row D records every radius and endpoint comparison used by both local windows, while S records the finite observations on which those comparisons are made.

Apply the classifier of Definition 5899.2. A refinement transported by H and replayed by C can change neither the componentwise ledger coordinates nor the finite observation coordinates exposed to the classifier.

The glued readback row Q is therefore the same row after refinement, and the $\mathsf{Real}^{\uparrow}$ seal E is read only through Q together with H, C, P, N. Since the carrier contains no quotient, selector, host-equality, choice, or completeness coordinate, none of those data can be exported by overlap stability. □

Theorem 5093.5 (Regular Cauchy gluing common-refinement exactness)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Every glued readback exported from* G *must expose the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The exactness route is transported by* H *and replayed by* C*, and it exports no quotient stream equality, selected representative, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted carrier. The row D is the common dyadic refinement ledger shared by the two local regular Cauchy windows, while Q is the glued regular readback and E is the terminal Real seal.

Apply overlap stability from Definition 5899.4. Any accepted refinement of the overlap preserves the same Q and E only through the displayed ledger and observation rows, so a glued readback cannot bypass D.

Finally use the Cauchy-reindexing handoff of Definition 5899.7. It reads D, S, Q as the finite reindexing-compatible route before any Real-facing consumer reaches E, with H, C, P, N preserving and naming the same route. □

Theorem 5093.6 (Regular Cauchy gluing overlap induction)**.** *Let $G = (L, R, D, S, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. For any finite chain of displayed overlap refinements transported by H and replayed by C, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E remain unchanged. The resulting route still factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyReindexing}^{\uparrow}$, *and it exports no quotient stream equality, selected representatives, host equality, countable choice, or ambient completeness.*

Proof. Induct on the displayed finite refinement list.

The empty list is the overlap-stability theorem of Definition 5899.4: the common dyadic ledger D and the finite observation row S already determine the same glued readback Q and seal E.

For the step, assume the earlier refinements preserve Q and E. Compose the next displayed refinement with the induction hypothesis, transport the component rows by H, and replay the same finite route by C.

Apply the classifier of Definition 5899.2 componentwise. Since the classifier reads only $L, R, D, S, Q, E, H, C, P, N$, the next refinement cannot add a quotient, representative selector, host equality row, choice schedule, or completeness witness; it only repeats the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5093.7 (Regular Cauchy gluing Cauchy-reindexing handoff)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier factors its gluing route through the finite* $\mathsf{CauchyReindexing}^{\uparrow}$ *surface recorded in P. The handoff exposes the common dyadic refinement ledger D, the finite observation row S, and the glued regular readback row Q as a reindexing-compatible route, and it exposes no global quotient of regular Cauchy names or ambient completeness principle.*

Proof. Read the package provenance P. It names the route through $\mathsf{CauchyReindexing}^{\uparrow}$ together with $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The carrier rows D, S, Q are exactly the finite data consumed by that route: D supplies the common refinement ledger, S supplies the inspected windows, and Q supplies the regular rational readback after the two local windows agree on the overlap.

The support rows H, C, N transport, replay, and name this same finite path. Thus the handoff is reindexing-compatible BEDC data, not a quotient of regular names, a selected representative, or a completeness theorem. □

Theorem 5093.8 (Regular Cauchy gluing obligation closure handoff)**.** *Every obligation-closure read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier must expose the full row tuple*

$$L, R, D, S, Q, E, H, C, P, N.$$

The route factors through the common $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and glued readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and*

the $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff named by* P *before any downstream real-name consumer may read the gluing packet.*

Proof. Start with the carrier and NameCert obligations. The accepted packet exposes $L, R, D, S, Q, E, H, C, P, N$, and the obligation surface has no smaller projection that still records the two local windows, their common refinement, and the glued readback.

Use the Cauchy-reindexing handoff of Definition 5899.7. It forces the gluing read through D, S, Q and through the provenance row P naming $\mathsf{CauchyReindexing}^{\uparrow}$, while the carrier rows L, R provide the two regular windows.

The rows E, H, C, N then seal, transport, replay, and name the same finite route. Thus obligation closure reads exactly the displayed tuple and factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5093.9 (Regular Cauchy gluing Real seal ordering)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reachable only after the common dyadic refinement row* D*, the finite observation row* S*, and the glued readback row* Q *have been displayed. The seal-ordering route asserts no quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Project the carrier definition. The row E is listed as the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after the two local windows, the common refinement ledger, the finite observations, and the glued readback have been displayed.

Apply the overlap-stability and Cauchy-reindexing handoff theorems. They keep the common dyadic refinement in D, the inspected windows in S, and the glued regular readback in Q visible before any real-name consumer can inspect E.

Since quotient stream equality, selected representatives, host equality, choice schedules, and ambient completeness are not carrier coordinates, the ordering statement cannot export them. It records only the displayed route from D, S, Q to the terminal seal E. □

Theorem 5093.10 (Regular Cauchy gluing local tail compatibility)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the two local regular-Cauchy windows* L *and* R *can be consumed by a* $\mathsf{Real}^{\uparrow}$*-seal reader only after the overlap row, the common-refinement row, and the Cauchy-reindexing row exhibit a shared* $\mathsf{RegSeqRat}^{\uparrow}$ *tail boundary. The compatibility route is the finite route through* D, S, Q, H, C, P, N *supplied by the carrier, and it exports no quotient stream equality, selected representative, or ambient completeness.*

Proof. Project the carrier of Definition 5899.1. The two local windows L, R are not read independently by a Real-facing consumer; the packet first displays their shared dyadic ledger D, finite observations S, and glued readback Q.

Apply overlap stability, Definition 5899.4, and common-refinement exactness, Definition 5899.5. These theorems keep the overlap row and the common refinement visible before Q is consumed, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ tails are compared at the displayed boundary rather than by host stream equality.

Apply overlap induction, Definition 5899.6, to finite chains of displayed refinements, and then use the Cauchy-reindexing handoff, Definition 5899.7. The resulting route makes the shared tail boundary stable under H and replayable by C before E is read.

Since quotient stream equality, selected representatives, and ambient completeness are absent from Definition 5899.1, the compatibility theorem can expose only the finite D, S, Q, H, C, P, N route. □

The local-tail compatibility row is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window surface of chapter 301 and the $\mathsf{StreamName}^{\uparrow}$ finite-observation surface of chapter 126. The exact dependency is the common-tail rows displayed by D and Q, together with finite-window readback through S, before the $\mathsf{Real}^{\uparrow}$ seal E is available to any downstream consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet is generated from two compatible regular Cauchy windows, a common $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ observations, a glued $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5899.1, Definition 5899.2, Definition 5899.3, Definition 5899.4, Definition 5899.6, Definition 5899.7, Definition 5899.8, Definition 5899.9, and Definition 5899.10 seed the finite gluing route for compatible regular Cauchy windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, common dyadic-refinement exactness, overlap stability, CauchyReindexing handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyGluingUp.

Theorem 5093.11 (Regular Cauchy gluing L10 sibling route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet reads its local gluing route through the L10 sibling lattice before it exports a NameCert row. The route first consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the Cauchy reindexing handoff of chapter 1568.*

Thus the gluing packet exports only a local finite NameCert route over the displayed dyadic, stream, regular-rational, Real-seal, and reindexing rows. It does not expose quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 5899.1. Its visible rows are two local regular windows, the common dyadic ledger, the finite observation row, the glued regular readback, the Real seal, and the transport, replay, provenance, and naming rows.

Apply Definition 5899.3. The local NameCert surface must read exactly those rows, with the dyadic ledger, StreamName windows, RegSeqRat readback, and Real seal appearing before the certificate is exported.

Use Definition 5899.7 for the reindexing component. It identifies the same dyadic, stream, and regular readback rows as a $\mathsf{CauchyReindexing}^{\uparrow}$-compatible route.

Thus the dependency on chapter 1568 is consumed before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal are available to the gluing certificate.

The exclusions are inherited from the carrier inventory and the overlap stability theorem. Since no coordinate stores quotient stream equality, selected representatives, host real equality, choice, or ambient completeness, the L10 sibling route remains a finite local NameCert route. □

Theorem 5093.12 (Regular Cauchy gluing RegSeqRat route)**.** *Every glued readback row of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must pass through the existing* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling surface of chapter 301 and the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff of chapter 1568 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The local obligations of Definition 5899.3 therefore expose a regular rational readback and a reindexing route before any Real-facing consumer may read the glued value.*

Proof. Start from the local L10 route of Definition 5899.11. It already orders the dyadic ledger, finite StreamName windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyReindexing}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal before the gluing NameCert row is exported.

Apply the local obligations of Definition 5899.3. The glued readback row is one of the displayed certificate rows, so it cannot be consumed without the same regular rational surface and structural support rows.

Use Definition 5899.7. The reindexing handoff keeps the dyadic, stream, and regular readback data aligned before the terminal Real seal appears. Thus a Real-facing read factors through RegSeqRat and CauchyReindexing rather than bypassing them. □

Theorem 5093.13 (Regular Cauchy gluing pasting nonescape)**.** *Every accepted pasting consumer of a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite route of Definition 5899.1, satisfy the local obligations of Definition 5899.3, and pass through the L10 sibling route of Definition 5899.11 before it exports any NameCert row. In particular, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently no such pasting consumer exports quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as a coordinate of the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *read.*

Proof. Project the accepted carrier in Definition 5899.1. Its displayed route already contains the common dyadic ledger, finite StreamName observations, glued RegSeqRat readback, Real seal, and the support rows for transport, replay, provenance, and naming.

Apply Definition 5899.3. A pasting consumer can export a local NameCert row only by preserving exactly those displayed rows, so the consumer has no separate entry point around the dyadic, stream, regular readback, or seal coordinates.

Apply Definition 5899.11. The L10 route orders the same rows through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before export.

The refused objects are absent from the carrier and absent from the sibling route. Thus quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness cannot be repacked as a pasting consumer of the accepted $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet. □

Theorem 5093.14 (Regular Cauchy gluing separator-use route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that exports an apartness read or a located-comparison read must first pass through the* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *route of chapter 7930. Ordinary gluing consumers still export only the finite dyadic, stream,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows named by Definition 5899.11.*

Proof. Start with the L10 sibling route of Definition 5899.11. It exposes the dyadic tolerance rows, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before the local gluing NameCert row is exported.

For a consumer that reads only gluing data, those rows exhaust the available surface by Definition 5899.13. The carrier has no coordinate for apartness, located comparison, quotient equality, selected representatives, choice, or ambient completeness.

If the same consumer asks for apartness or located comparison, apply the separator sibling dependency of Definition 7930.3. That route requires the stream, regular-rational, dyadic, Real-seal, located-cut, and apartness-field rows to be displayed through $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ before such a read is available.

Thus the gluing packet itself remains the ordinary finite route, while apartness-facing or located-comparison-facing reads factor through the separator chapter rather than being smuggled into the gluing carrier. □

Theorem 5093.15 (Regular Cauchy gluing completion consumer route)**.** *Every Bishop-completion or Real-completion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the same finite L10 sibling route before exporting a gluing* $\mathsf{NameCert}$ *row. In particular, the consumer first exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The exported gluing certificate has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already orders the dyadic ledger, stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any local gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12. A completion-facing read of the glued value therefore cannot bypass the regular rational surface or the reindexing route on its way to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply the pasting nonescape theorem, Definition 5899.13. It keeps every gluing consumer inside the finite carrier route and excludes quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness as exportable coordinates.

Finally use the separator-use route of Definition 5899.14. Apartness-facing or located-comparison-facing completion reads factor through the separator sibling; ordinary Bishop-completion and Real-completion gluing reads still expose only the dyadic, stream, regular-rational, reindexing, and Real-seal rows before the gluing $\mathsf{NameCert}$ is named. □

Theorem 5093.16 (Regular Cauchy gluing uniform Cauchy criterion route)**.** *Every uniform-Cauchy-criterion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite sibling route before exporting criterion data. The consumer must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of the gluing route.*

Consequently the criterion consumer exports only the finite BEDC packet obtained from those rows, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *ledger. It does not export quotient sequence equality, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already places the dyadic tolerance ledger, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12 and the completion consumer theorem Definition 5899.15. Together they force a completion-facing criterion read through the regular rational surface, the reindexing handoff, and the Bishop-completion consumer rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally use the downstream sibling route of Definition 5899.18. It permits only concrete downstream consumers after the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows have been displayed. Since the carrier and all cited routes omit quotient sequence equality, selected representatives, countable choice, host real equality, and ambient completeness, none of those objects can be repacked as uniform-Cauchy-criterion data. □

Theorem 5093.17 (Regular Cauchy gluing L10 sibling dependency)**.** *Before any gluing consumer reads the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row, it must display the finite sibling dependency route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *observation windows of chapter 126,*

the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the completion-facing handoff of chapter 3665, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route may also be read by Bishop-completion consumers through the completion consumer theorem Definition 5899.15, but it remains a finite dependency lattice rather than quotient sequence equality, selected representatives, countable choice, host real equality, or ambient real completeness.*

Proof. Start with Definition 5899.11. It already orders the dyadic tolerance rows, stream observation windows, regular-rational readback, Cauchy reindexing handoff, and Real seal before the local $\mathsf{NameCert}$ row is exported.

Apply Definition 5899.15. A Bishop-completion or Real-completion consumer therefore reads the same finite row sequence before it can use the gluing packet as completion-facing data.

The dependency is finite because every cited sibling appears as a displayed row or handoff in the local route. No step asks for quotient sequence equality, a selected representative of a Cauchy class, countable choice over tails, host real equality, or an ambient completeness theorem. □

Theorem 5093.18 (Regular Cauchy gluing downstream sibling route)**.** *Any cofinal, interleaving, or comparison-facing consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first pass through the finite sibling route of Definition 5899.17. After the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *rows have been displayed, such a consumer may continue only through a concrete downstream sibling such as chapter 414 or chapter 3514. The gluing packet itself exports no detached cofinal route, no interleaving selector, no quotient equality of Cauchy names, no selected representative, and no ambient completeness theorem.*

Proof. Start with Definition 5899.17. It forces every gluing consumer to display the regular rational rows, finite stream windows, dyadic tolerance rows, Bishop-completion handoff, and terminal Real seal before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ row is read.

Apply the completion consumer theorem Definition 5899.15. A completion-facing read therefore receives the same finite route, not a quotient stream or selected representative.

For an interleaving-facing read, use the public sibling surface of chapter 414. It supplies the alternating selector and combined modulus ledger only after the relevant $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are displayed.

For a comparison-facing read, use chapter 3514. That packet supplies the even–odd comparison and shared threshold rows as a separate finite consumer surface. Thus any cofinal or interleaving continuation is downstream of the existing L10 sibling dependency, and none of those rows is smuggled into the gluing carrier itself. □

Theorem 5093.19 (Regular Cauchy gluing tail-agreement composition)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads a glued* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet through Definition 5899.18 must continue through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699 before treating the glued readback as completion-facing tail agreement. The composed route also reads the Cauchy limit seal of chapter 386; the gluing packet alone exports no ambient tail equality, selected representative, quotient sequence equality, or completeness witness.*

Proof. Start with Definition 5899.18. It permits downstream consumers only after the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows have been displayed.

A completion-facing tail read then has an additional boundary. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ chapter supplies the route that turns a Real-facing readback into displayed tail-agreement data, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter supplies the limit-facing seal used by completion consumers.

Thus the row order is finite: first the local gluing sibling route, then the tail-agreement seal, and finally the Cauchy-limit seal. Because each step only reuses displayed rows, no consumer can replace the route by host tail equality, a selected Cauchy representative, quotient sequence equality, or ambient completeness. □

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes its L10 sibling lattice before any gluing read is exported: the $\mathsf{StreamName}^{\uparrow}$ finite windows provide the inspected overlap rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular readback, $\mathsf{DyadicRatCore}^{\uparrow}$ carries the shared tolerance ledger, $\mathsf{Real}^{\uparrow}$ seals the terminal value, and chapter 5544 provides the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ schedule that keeps the two local Cauchy windows on one displayed budget.

Theorem 5093.20 (Regular Cauchy gluing L10 forward route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer reaches Real-facing gluing only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of the local carrier. The route first reads the name certificate construction, then the L10 sibling route, and finally the Bishop-completion consumer route before any gluing conclusion is exported. It refuses quotient stream equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as coordinates of the gluing read.*

Proof. Project the local carrier and obligations from Definition 5899.1 and Definition 5899.3. The rows available to a consumer are the two compatible windows, the common dyadic ledger, finite observations, glued regular readback, Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the consumer through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a local gluing certificate is exported.

Apply the Bishop-completion consumer theorem Definition 5899.22. A Real-facing or Bishop-completion consumer must display the same D, S, Q, E, H, C, P, N route, so the forward path remains finite and local to the accepted packet.

The refused objects are absent from every cited step. The name certificate construction supplies no quotient stream equality; the L10 sibling route supplies no selected limit or countable-choice schedule; and the Bishop-completion consumer route supplies no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Hence every accepted forward gluing read factors through the displayed windows, readbacks, tolerance ledgers, and Real seal. □

Theorem 5093.21 (Regular Cauchy gluing sibling forward link)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that is read as a gluing consumer first passes through the L10 sibling route of Definition 5899.11. In particular, the consumer must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126 before the local gluing* $\mathsf{NameCert}$ *row is available.*

The forward link is therefore a concrete sibling dependency route rather than a detached hub edge. It also preserves the exclusions already named by the L10 route: no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is exported by the gluing read.*

Proof. Start with the carrier and obligation rows of Definition 5899.1 and Definition 5899.3. They display the paired local windows, common dyadic ledger, finite observations, glued regular readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Apply the L10 sibling route of Definition 5899.11. That route orders the same gluing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any gluing certificate can be exported.

Thus the hub has a visible child route into the RegSeqRat and StreamName sibling chapters. The refused objects remain absent because neither the local carrier nor the L10 sibling route contains coordinates for quotient stream equality, selected representatives, choice, host equality, or ambient completeness. □

Theorem 5093.22 (Regular Cauchy gluing Bishop completion consumer)**.** *Every* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first expose the same finite gluing carrier rows already displayed by Definition 5899.1. In particular, before the Bishop completion route may read the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *glued readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff recorded in* P*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

Thus the Bishop completion consumer receives the gluing packet only as a finite L10 route over dyadic, stream, regular-rational, Real-seal, reindexing, transport, replay, provenance, and naming data. It receives no quotient stream equality, selected representative, host real equality, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The rows that a completion-facing consumer can inspect are exactly the two local windows, the common dyadic ledger, the finite observations, the glued readback, the Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the gluing packet through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$ before a NameCert row is exported.

Now read the Bishop completion carrier of chapter 3665. Its completion-facing route consumes dyadic precision, finite stream windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming. The gluing rows D, S, Q, E, H, C, P, N are therefore the only rows that can feed such a consumer from $\mathsf{RegularCauchyGluing}^{\uparrow}$.

The exclusions follow from the same carrier inventory. Since neither Definition 5899.1 nor the Bishop completion route supplies quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness as a coordinate, the consumer cannot export any of those objects from the gluing packet. □

Theorem 5093.23 (Regular Cauchy gluing Real-tail forward route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that asks for a tail-agreement seal is a forward dependency of the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route in chapter 699. Concretely, the gluing packet first exposes the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the paired* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the local* $\mathsf{Real}^{\uparrow}$ *seal* E*; only then may the consumer enter the Real-tail agreement and Cauchy-limit seal route through* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*.*

Proof. Project the accepted gluing carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The Real-facing rows available before any tail agreement are exactly the common dyadic ledger D, finite observation row S, glued regular readback Q, terminal seal E, transport row H, replay row C, provenance row P, and local name row N.

Apply the Real seal ordering theorem Definition 5899.9. It forces the route to display D, S, Q before E is readable, so a consumer cannot jump from the two local windows directly to a Real-tail agreement row.

Now use the Real-tail agreement completion-consumer boundary of chapter 699. That boundary receives paired windows, dyadic readback, agreement, Cauchy-limit sealing, and Real or Com-

pletion endpoints as one finite route. Feeding it from $\mathsf{RegularCauchyGluing}^{\uparrow}$ therefore makes the gluing read a forward dependency of that Real-tail route rather than a separate kernel island.

The exclusions are inherited from the two carriers. Neither Definition 5899.1 nor the Real-tail agreement route supplies quotient stream equality, selected representatives, host convergence, countable choice, or ambient completeness as a row, so no such object can be exported by this forward dependency. □

Chapter 5094

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet names the explicit tail-majorant data used when regular rational sequences are read through finite tolerances. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus surface of chapter 292, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records a finite threshold service for tail estimates; it is not a completeness theorem, a selected limit operator, or a quotient equality on real names.

Falsifiable prediction: Removing the majorant-threshold row permits a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal to cite a tail estimate without naming the finite stage after which the estimate is valid. Removing the dyadic tolerance row prevents the threshold from being checked against a displayed rational error bound.

Independence witness: Nearest siblings include $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. None is bijective to $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ because the budget packet stores observation service, the modulus packet stores a general tolerance schedule, the regular sequence packet stores the source reads, the stream packet stores names, the dyadic packet stores rational tolerances, and the real packet stores a seal rather than the finite tail-majorant threshold binding them.

Definition 5094.1 (Regular Cauchy tail majorant carrier)**.** *A* $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the inspected regular-sequence tail requests.*
2. *M is the majorant-threshold row assigning to each displayed tolerance request a finite stage after which the tail estimate is accepted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger whose dyadic entries are the error bounds tested by M.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational sequence only through the displayed schedule, threshold, and dyadic tolerance rows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the scheduled tail-majorant readback has been replayed.*

6. *H records componentwise* hsame *transport for the schedule, majorant threshold, dyadic tolerance, regular readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from schedule requests through the majorant threshold, dyadic tolerance ledger, regular readback, and seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$ *row.*

The carrier accepts only packets whose regular Cauchy tail estimates are read through a displayed finite majorant threshold for the requested dyadic tolerance before any $\mathsf{Real}^{\uparrow}$ *seal row is consumed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMajorantUp.RegularCauchyTailMajorantTasteGate single carrier alignment`

Theorem 5094.2 (Regular Cauchy tail majorant NameCert obligations)**.** *For the carrier of Definition 5900.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$J = (S, M, D, R, E, H, C, P, N),$$

schedule exposure through S, majorant-threshold exposure through M, dyadic tolerance readability through D, regular rational readback through R, seal handoff through E, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier. The displayed rows $S, M, D, R, E, H, C, P, N$ are the only coordinates of an accepted $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet.

Read the analytic rows in order. The schedule row S names the inspected tail requests, the threshold row M supplies the finite majorant stage for each displayed request, and the dyadic row D supplies the tolerance being tested.

The readback and seal rows are downstream of that finite service. The $\mathsf{RegSeqRat}^{\uparrow}$ row R may be read only after S, M, D have been displayed, and the $\mathsf{Real}^{\uparrow}$ seal row E consumes that readback rather than creating an ambient limit.

Finally H, C, P, N transport, replay, package, and name the same finite route. Since no carrier coordinate stores quotient real equality, a selected limit, countable choice, or completeness, the local NameCert obligations are exactly the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5900.1 and Definition 5900.2 bind the seed carrier and local NameCert surface for explicit regular Cauchy tail majorants.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completeness of $\mathsf{Real}^{\uparrow}$, selected limits, quotient equality of real names, countable choice, or Lean verification.

Upgrade path: Formalization requires a Lean carrier for the schedule, dyadic tolerance, regular readback, seal handoff, and finite majorant-threshold rows.

Constructive story: A $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet is generated from a finite stream schedule, displayed majorant thresholds, dyadic tolerance rows, regular readback, a seal handoff, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 5095

A Concrete Naming Certificate for $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$

Following Bishop and Bridges on nested interval principles, the $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ packet records the finite Real-facing window by which nested located intervals are inspected before a real-name seal is read. It sits over the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested/Cantor neighbours chapter 320 and chapter 4359. The packet names a finite witness route for Real and Completion consumers; it does not claim open-cover compactness, countable choice, quotient reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5095.1 (RealCantorIntersectionFiniteWindow carrier)**.** *A* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested located-interval row, carrying the visible interval prefix, the containment witness, and the located endpoint tests read from the nested-interval surface.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger, recording the dyadic width bounds and tolerance requests attached to the displayed prefix.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the interval and endpoint observations inspected by the consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and the diameter ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after I, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the nested interval, diameter, window, readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$I, D \longmapsto W \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, D, W, R, E, H, C, P, N$ and the displayed replay order. It has no coordinate for an infinite intersection selector, quotient real equality, open-cover compactness, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCantorIntersectionFiniteWindowUp.RealCantorIntersectionFiniteWindowTasteGate`
`single carrier alignment`

Theorem 5095.2 (RealCantorIntersectionFiniteWindow NameCert obligations)**.** *For every accepted* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ *carrier*

$$Q = (I, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$ *surface consists exactly of carrier admission, nested located-interval exposure, dyadic diameter accounting, finite StreamName window exposure, RegSeqRat readback, Real-seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must pass through the finite nested-window route*

$$I, D \longmapsto W \longmapsto R \longmapsto E$$

before the terminal seal is public.

Proof. Project the carrier of Definition 5901.1. The accepted packet exposes exactly the nested interval row I, dyadic diameter ledger D, finite window row W, regular readback row R, Real seal E, and the structural rows H, C, P, N.

The analytic obligations are read by projection through those rows. The nested located-interval row supplies the visible containment prefix and endpoint locatedness, while the dyadic ledger supplies the diameter budget that makes the inspected prefix usable for a Real-facing read.

Next the window and readback rows fix the finite observation route. The StreamName row W names the inspected observations, and the RegSeqRat row R is the only regular rational readback coordinate generated from them.

Finally the seal and structural rows close the packet. The continuation row places E after I, D, W, R, the transport row preserves the same coordinates, and P, N source and name only this local certificate. Since no further coordinate occurs in the carrier, the listed rows exhaust the NameCert obligations and exclude open-cover compactness, countable choice, quotient reals, selected limits, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$, seed)

Scope closed: Definition 5901.1 and Definition 5901.2 seed the finite Cantor-intersection window route over nested located intervals, dyadic diameter ledgers, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient reals, selected infinite intersections, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested located-interval exposure, diameter accounting, finite-window exposure, regular readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealCantorIntersectionFiniteWindowUp.

Constructive story: A $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ packet is generated from nested located intervals, a DyadicRatCore diameter ledger, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ selector chapter is a structural consumer of chapter 381 and chapter 5899 through its finite selected-window route.

Chapter 5096

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceSelector}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy sequences, the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ packet records the finite selector schedule by which a regular rational Cauchy name may expose a monotone subsequence corridor before a Real or Completion consumer reads the terminal seal. It sits after the regular rational source of chapter 301, the finite window surface of chapter 126, the terminal real-name boundary of chapter 38, and the Cauchy hub of chapter 406. The packet gives diagonal, tail, and completion-facing chapters a named L10 selector object; it is not a quotient of streams, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Any subsequence-selector read must display the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 and the $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ finite-window schedule of chapter 8091 before reaching $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. The selector is therefore a visible diagonal-and-window corridor, not a detached subsequence choice. The fast selector route of chapter 1518 consumes this schedule when the cofinal subsequence corridor must be made fast enough for $\mathsf{Real}^{\uparrow}$ completion windows, while chapter 5899 consumes compatible selected windows when the subsequence corridor is glued back into a regular Cauchy route.

Definition 5096.1 (Cauchy subsequence selector carrier)**.** *A* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, M, G, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational reads are being sampled.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the source observations consumed by the selector.*
3. *M is the monotone selector row, recording a finite schedule of indices and the strict order checks between successive selected positions.*
4. *G is the Cauchy-gap ledger, recording the tail comparison requests that make the selected windows compatible with the source Cauchy evidence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, W, M, G have been displayed.*
6. *H records componentwise* hsame *transport for the source, window, monotone selector, Cauchy-gap, seal, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a subsequence request through R, W, M, G, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, M, G, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5096.2 (Cauchy subsequence selector NameCert obligations)**.** *For every accepted* CauchySubsequenceSel *carrier $S = (R, W, M, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, selector monotonicity,* $\mathsf{StreamName}^{\uparrow}$ *window coverage, Cauchy-gap compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read through the finite route*

$$R, W, M, G \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Carrier admission is projection from Definition 5902.1. The packet first exposes the regular rational source R, the finite observation windows W, the monotone selector schedule M, and the Cauchy-gap ledger G.

Selector monotonicity is local to M. The window coverage obligation then reads W at precisely the positions displayed by M, so every selected index has a finite $\mathsf{StreamName}^{\uparrow}$ observation before any handoff is consumed.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the source row R restricted along the displayed schedule and checked against the Cauchy-gap ledger G. The terminal seal E is downstream of those rows, and it is only a $\mathsf{Real}^{\uparrow}$ seal for the displayed subsequence corridor.

Transport through H, replay through C, provenance P, and local naming N preserve the same route. Since quotient stream equality, selected limits, host real equality, countable choice, trichotomy, and ambient Real completeness are absent from the carrier coordinates, no obligation can discharge outside the displayed finite packet. □

Theorem 5096.3 (Cauchy subsequence selector real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the monotone selector schedule, and the Cauchy-gap ledger. The handoff to E therefore supplies a finite subsequence corridor from regular rational data to Real-facing data, not a host subsequence theorem, not a selected limit, and not a quotient construction.*

Proof. Project an accepted carrier $S = (R, W, M, G, E, H, C, P, N)$. The source row R is the only regular rational input, and W is the only finite observation surface available to a consumer.

The selector row M records the monotone schedule. The Cauchy-gap ledger G records the finite tail comparisons that let the scheduled windows be read as a subsequence corridor of the same source.

Only after R, W, M, G are present may the consumer inspect E as a Real-facing seal. The transport, replay, provenance, and naming rows keep that order visible to downstream chapters.

The carrier contains no coordinate for quotient stream equality, a chosen host subsequence, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Thus every Real or Completion handoff factors through the displayed selector rows. □

Theorem 5096.4 (Cauchy subsequence selector sibling lattice route)**.** *Every* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *consumer must pass through* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *and* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are consumed. The selector route therefore reads the diagonal-selection corridor, the regular Cauchy finite-window induction ledger, the source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, seal row E, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *row as one sibling lattice. Before any gluing-facing consumer reads* $\mathsf{RegularCauchyGluing}^{\uparrow}$, *the selected finite windows are certified as regular-Cauchy subsequence data through chapter 381; only then can the gluing route consume the selector rows R, W, M, G before the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

The route exports no detached subsequence choice, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier $S = (R, W, M, G, E, H, C, P, N)$ from Definition 5902.1. The source, window, monotone selector, Cauchy-gap, and seal rows are the only analytic rows available to a consumer, while H, C, P, N are the structural rows that transport, replay, source, and name them.

Apply Definition 5902.2. It forces carrier admission, selector monotonicity, window coverage, Cauchy-gap compatibility, regular-rational handoff, seal non-escape, transport, replay, provenance, and local naming before any consumer can read the selector.

Apply Definition 5902.3. The $\mathsf{Real}^{\uparrow}$ or Completion-facing read reaches E only after R, W, M, G have been displayed, so the diagonal-selection corridor and regular Cauchy window-induction ledger must be upstream of the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows named in the introduction; the same displayed windows are first certified by chapter 381 before $\mathsf{RegularCauchyGluing}^{\uparrow}$ can consume them.

Since the carrier has no coordinate for detached subsequence choice, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the sibling lattice cannot export any of those principles. □

Theorem 5096.5 (Cauchy subsequence selector diagonal-window readiness)**.** *Every fast-selector or gluing-facing consumer of a* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must expose the* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *route and the* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *finite schedule before it consumes the selector rows $R, W, M, G, E, H, C, P, N$. In particular, a* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *consumer cannot read the fast cofinal corridor, and a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer cannot read compatible selected windows, until the diagonal route and finite-window schedule have already displayed the source row, finite windows, monotone selector row, Cauchy-gap ledger, terminal seal, transport, replay, provenance, and local naming rows.*

Proof. Apply Definition 5902.2. It makes $R, W, M, G, E, H, C, P, N$ the full NameCert surface for an accepted selector packet, so a downstream consumer has no row outside that finite display from which to read subsequence data.

Apply Definition 5902.3. The Real or Completion-facing handoff reaches E only after the regular rational source, finite windows, selector schedule, and Cauchy-gap ledger have been displayed.

Now apply Definition 5902.4. It places the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ corridor and the $\mathsf{RegularCauchyWindowIndu}$ schedule before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$, and gluing-facing rows are consumed. Therefore both the fast-selector route and the gluing route must expose the diagonal-window readiness surface before reading $R, W, M, G, E, H, C, P, N$. □

Theorem 5096.6 (Cauchy subsequence selector regular-window handoff)**.** *Every* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *handoff that is later glued or sealed through* $\mathsf{Real}^{\uparrow}$ *first consumes* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *as the finite-window certificate that the selected rows remain regular-Cauchy subsequence data. Thus the selected windows* W *and monotone schedule* M *are certified through chapter 381 before any* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *or terminal* $\mathsf{Real}^{\uparrow}$ *consumer reads* R, W, M, G, E.

Proof. Project the accepted carrier $S = (R, W, M, G, E, H, C, P, N)$. The selected windows W and monotone schedule M are the only rows that can present subsequence data to a downstream consumer.

Use the sibling lattice route of Definition 5902.4. It places the diagonal-selection corridor and regular Cauchy window-induction ledger before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are consumed.

The regular-window handoff is therefore a finite certification step: W and M are read as regular-Cauchy subsequence data before gluing or sealing can consume R, W, M, G, E. The route has no coordinate for a detached subsequence choice, selected host limit, or ambient completeness principle. □

Theorem 5096.7 (Cauchy subsequence selector forward density route)**.** *Every* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *or density-facing completion consumer that reads selected Cauchy subsequence windows from an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first pass through* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route displayed by the selector. The forward density read therefore consumes the selector rows* $R, W, M, G, E, H, C, P, N$ *as finite regular-Cauchy subsequence data, not as a separate host subsequence, a choice-selected limit, or an ambient completeness principle.*

Proof. Apply the Real handoff theorem, Definition 5902.3. It forces every Real or Completion-facing read through the regular rational source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, and terminal seal row E.

Use the sibling lattice route of Definition 5902.4 together with the regular-window handoff of Definition 5902.6. These place $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RegularCauchySubsequence}^{\uparrow}$ before any downstream gluing, sealing, or density-facing consumer reads the selected windows.

Thus a $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ consumer receives only the displayed finite selector corridor. Since the carrier has no coordinate for a detached host subsequence, selected limit, quotient stream equality, or ambient completeness, the forward density route cannot bypass the existing selector handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName windows, monotone selector schedule, Cauchy-gap ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$, seed)

Scope closed: Definition 5902.1, Definition 5902.2, and Definition 5902.3 seed the finite selector corridor from regular rational Cauchy data to a Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, selector monotonicity, StreamName window coverage, Cauchy-gap compatibility, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming.

The fast selector route of chapter 1518 consumes the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ monotone schedule when the subsequence corridor must be made fast enough for $\mathsf{Real}^{\uparrow}$ completion windows. The $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ route is consumed by chapter 1518 when a cofinal fast subsequence schedule must be displayed before a completion-facing read, and by chapter 5899 when selected finite windows are glued back into a regular Cauchy route. Thus the selector sits in the L10 dependency lattice between finite StreamName windows, regular rational source rows, fast subsequence scheduling, and regular Cauchy gluing, rather than floating as a kernel-only carrier.

Theorem 5096.8 (CauchySubsequenceSelector regular Cauchy gluing forward route)**.** *Every* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first read the source regular-rational row R, the selected finite windows W, the monotone selector schedule M, and the Cauchy-gap ledger G before gluing compatible selected windows back into a regular Cauchy route. The handoff to chapter 5899 therefore remains a displayed finite-window route rather than a selected representative, quotient stream equality, countable-choice schedule, or ambient Real-completeness principle.*

Proof. Project the selector carrier described in chapter 5902. Its gluing-facing data are the source row R, finite selected windows W, monotone selector schedule M, and Cauchy-gap ledger G.

Only after those rows are visible can a gluing consumer inspect compatible windows. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ route then supplies its own regular-Cauchy compatibility and terminal Real-seal boundary, so the selector packet is a forward dependency of gluing rather than a replacement for the gluing certificate.

Since neither the selector rows nor the gluing route contain quotient stream equality, selected representatives, countable choice, or ambient Real completeness, the composed route exports only the finite selected-window data and the downstream regular Cauchy gluing surface. □

Theorem 5096.9 (Cauchy subsequence selector finite-window forward link)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *read factors through the finite-window route*

$$\mathsf{FiniteStream}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{CauchyDiagonalSelection}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the terminal seal is available. The consumer therefore reads the selector carrier rows as displayed finite windows and diagonal-selection data, not as a quotient subsequence, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Proof. Project the selector carrier from Definition 5902.1. Its source row, finite windows, monotone schedule, Cauchy-gap ledger, real seal, transport, replay, provenance, and local naming rows are the only coordinates available to a downstream read.

Use the real handoff of Definition 5902.3. It forces every seal consumer to pass through the regular rational source row, the $\mathsf{StreamName}^{\uparrow}$ finite windows, the selector schedule, and the Cauchy-gap ledger before reading the terminal $\mathsf{Real}^{\uparrow}$ row.

Now read the sibling lattice route of Definition 5902.4. The lattice places the selector between $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$, finite window data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally identify the forward consumer rows: $\mathsf{FiniteStream}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ provide the finite observations, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies the displayed diagonal corridor, $\mathsf{RegSeqRat}^{\uparrow}$ reads the regular rational name, and $\mathsf{DyadicRatCore}^{\uparrow}$ records the tolerance row before the terminal seal. Since all of these rows are explicit carrier-local reads, no refused quotient, choice, selected-limit, host-equality, or ambient-completeness endpoint can enter the route. □

The selector packet also consumes the regular-Cauchy subsequence surface of chapter 381. Its monotone selector row, finite windows, and Cauchy-gap ledger are read as a route inside the same regular-Cauchy finite-window surface before the terminal $\mathsf{Real}^{\uparrow}$ seal is available. Thus a downstream completion consumer sees the selected subsequence as displayed regular-Cauchy data, not as a detached host subsequence or a separate choice principle. The selector dependency route is consumed by both the fast subsequence handoff of chapter 1518 and the regular-Cauchy gluing route of chapter 301. Thus a downstream Cauchy consumer reads the same displayed selector schedule, finite windows, and gap ledger before treating the selected route as fast enough for completion-facing readback.

Theorem 5096.10 (Cauchy subsequence selector orphan lattice route)**.** *Every selector read consumed by* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *must preserve the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *witness row, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal before any subsequence extraction is public. Thus the extractor-facing selector route is the finite lattice*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegularCauchyMinModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

inside the accepted $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet, not a detached subsequence choice or quotient stream selection.*

Proof. Project the selector carrier of Definition 5902.1. The available selector rows are the source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, terminal seal E, transport, replay, provenance, and local naming.

Use the selector dependency route and the cofinal subsequence dependency route. They force every extractor-facing read to pass through the displayed selector schedule before the extraction row can be consumed as finite subsequence data.

Finally read the regular-Cauchy minimum-modulus forward link. It keeps the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ witness visible alongside the $\mathsf{RegSeqRat}^{\uparrow}$ source row and $\mathsf{StreamName}^{\uparrow}$ finite windows before the terminal $\mathsf{Real}^{\uparrow}$ seal is public. Hence the selector exports only the stated finite lattice to the extractor and no separate host-side subsequence selection. □

Chapter 5097

A Concrete Naming Certificate for $\mathsf{ArtinianRing}^{\uparrow}$

Following Hopkins–Levitzki and the Atiyah–Macdonald Artinian ring tradition, the $\mathsf{ArtinianRing}^{\uparrow}$ packet records finite descending ideal-chain stabilization data over a ring, commutative-ring, and ideal source. In BEDC form it sits after $\mathsf{NameCert}_{\mathsf{Ring}^{\uparrow}}$ (chapter 43), $\mathsf{NameCert}_{\mathsf{CommRing}^{\uparrow}}$ (chapter 44), and $\mathsf{NameCert}_{\mathsf{Ideal}^{\uparrow}}$ (chapter 81), while complementing the ascending-chain sibling $\mathsf{NameCert}_{\mathsf{NoetherianRing}^{\uparrow}}$ of chapter 3637. The packet supplies a descending-chain source row for later algebra consumers; it does not assert module finite length, semisimplicity, quotient ideal spaces, maximal ideal choice, or the Hopkins–Levitzki theorem itself.

5097.1 Carrier Definition

Definition 5097.1 (Artinian ring carrier)**.** *An* $\mathsf{ArtinianRing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{ArtRing}(R, C, I, D, L, H, Q, P, N).$$

The row R is the ambient $\mathsf{Ring}^{\uparrow}$ *source surface. The row C is the* $\mathsf{CommRing}^{\uparrow}$ *source surface when the consumer reads the commutative specialization. The row I is the* $\mathsf{Ideal}^{\uparrow}$ *source surface over the same classifier. The row D is the descending-chain stabilization witness: every displayed descending chain of ideal rows over R, C, I carries a finite stage after which all later ideal endpoints classify with that stage. The row L is the finite-length ledger used by algebraic consumers that read descending stabilization together with the ascending-chain sibling of* $\mathsf{NoetherianRing}^{\uparrow}$*. The row H is componentwise* hsame *transport for the ring, commutative-ring, ideal, stabilization, and finite-length rows. The row Q is displayed* Cont *replay for the chain and length certificates. The row P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{ArtinianRing}^{\uparrow}}$ *naming row.*

Lean4 ✓ `BEDC.Derived.ArtinianRingUp.ArtinianRingTasteGate single carrier alignment`

Theorem 5097.2 (Artinian ring NameCert obligations)**.** *For an accepted* $\mathsf{ArtinianRing}^{\uparrow}$ *carrier* $\mathsf{ArtRing}(R, C, I, D, L, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ArtinianRing}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, source compatibility with* $\mathsf{Ring}^{\uparrow}$*,* $\mathsf{CommRing}^{\uparrow}$*, and* $\mathsf{Ideal}^{\uparrow}$*, descending-chain stabilization access through D, finite-length ledger access through L, componentwise classifier transport through H, displayed continuation replay through Q, provenance non-escape through P, and local naming through N.*

Proof. Project the rows of Definition 5097.1. The source rows R, C, I fix the ring, commutative-ring, and ideal classifiers before any chain is inspected.

The stabilization row D is then read over that shared source surface. The finite-length ledger L is available only as a displayed ledger attached to the same classifier, so it cannot replace the descending-chain witness by a host theorem.

Finally H, Q, P, N provide componentwise transport, continuation replay, provenance, and local naming. Since no quotient ideal space, maximal ideal selector, semisimplicity proof, or Hopkins–Levitzki theorem row is present, the listed coordinates exhaust the NameCert obligations. □

Theorem 5097.3 (Artinian descending-chain stability)**.** *Let* $\mathsf{ArtRing}(R, C, I, D, L, H, Q, P, N)$ *be an accepted* $\mathsf{ArtinianRing}^{\uparrow}$ *carrier. For every displayed descending chain of ideal rows over the shared* $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, *and* $\mathsf{Ideal}^{\uparrow}$ *classifier, the witness* D *supplies a finite stabilization stage, and* H *transports that stabilization along the same classifier. Any downstream algebra consumer may therefore use* $\mathsf{ArtinianRing}^{\uparrow}$ *as a finite descending ideal-chain certificate without re-proving stabilization inside its own packet.*

Proof. Begin with the source compatibility rows from Definition 5097.2. They force the descending ideal chain to be read over the ring, commutative-ring, and ideal classifiers already present in R, C, I.

Apply the stabilization witness D to obtain a finite stage for the displayed descending chain. If the accepted packet is transported through a classified replacement, use the componentwise transport row H to move both the chain entries and the stabilization witness through the shared classifier.

The transported stage remains a stage for the same finite chain surface because Q replays only the displayed continuation data and P prevents provenance escape. The conclusion is therefore a reusable finite descending ideal-chain certificate that can be paired with the ascending-chain surface of chapter 3637. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArtinianRing}^{\uparrow}}$.

Constructive story: An $\mathsf{ArtinianRing}^{\uparrow}$ packet is generated from RingUp, CommRingUp, and IdealUp source rows, a descending-chain stabilization witness, a finite-length ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArtinianRing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5097.1, Definition 5097.2, and Definition 5097.3 seed the finite descending ideal-chain stabilization interface over RingUp, CommRingUp, IdealUp, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArtinianRing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close module finite length, semisimplicity, quotient ideal spaces, maximal ideal choice, the Hopkins–Levitzki theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, descending stabilization access, finite-length ledger access, componentwise transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.ArtinianRingUp.

Chapter 5098

A Concrete Naming Certificate for $\mathsf{MooreSmithSubnet}^{\uparrow}$

Following Moore and Smith on nets and Kelley-style subnet reindexing, the $\mathsf{MooreSmithSubnet}^{\uparrow}$ packet records the standard subnet relation as finite BEDC naming data. It sits between the $\mathsf{DirectedSubnet}^{\uparrow}$ cofinal-tail surface of chapter 1600, the Cauchy-facing restriction route of $\mathsf{CauchySubnet}^{\uparrow}$ in chapter 1592, and the convergence interface of chapter 1667. Its Real-facing readback uses the finite windows of chapter 126, the dyadic tolerance row of chapter 291, the regular rational handoff of chapter 301, and the terminal seal of chapter 38. The packet names cofinal subnet evidence without selecting a host subnet representative, quotienting nets, invoking countable choice, or importing ambient completeness.

Definition 5098.1 (Moore–Smith subnet carrier)**.** *A* $\mathsf{MooreSmithSubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, J, \phi, A, T, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the directed source-index row for the original net.*
2. *J is the directed subnet-index row.*
3. *ϕ is the monotone cofinal map row from J into I, read through the* $\mathsf{DirectedSubnet}^{\uparrow}$ *interface.*
4. *A is the source net-value row, containing only the finite values inspected by the packet.*
5. *T is the cofinal tail ledger, recording for each displayed source tail a subnet tail whose image lies inside it.*
6. *W is the finite observation-window row over* $\mathsf{StreamName}^{\uparrow}$*.*
7. *D is the dyadic tolerance row over* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
8. *R is the regular rational readback row over* $\mathsf{RegSeqRat}^{\uparrow}$*.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, exposed only after $I, J, \phi, A, T, W, D, R$ have been displayed.*
10. *H records componentwise* hsame *transport for the directed, cofinal, observation, readback, seal, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from subnet requests through the same finite route.*

12. *P is the* Pkg *provenance over* $\mathsf{MooreSmithSubnet}^{\uparrow}$, $\mathsf{DirectedSubnet}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithSubnet}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through the displayed rows $I, J, \phi, A, T, W, D, R, E, H, C, P, N$. It has no coordinate for a selected host subnet, quotient-net equality, a chosen limit point, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.MooreSmithSubnetUp.MooreSmithSubnetTasteGate single carrier alignment`

Theorem 5098.2 (Moore–Smith subnet NameCert obligations)**.** *For every accepted* $\mathsf{MooreSmithSubnet}^{\uparrow}$ *carrier $M = (I, J, \phi, A, T, W, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithSubnet}^{\uparrow}}$ *surface consists exactly of directed source admission, directed subnet-index admission, monotone cofinal map admission, finite net-value observation, cofinal-tail exactness,* $\mathsf{StreamName}^{\uparrow}$ *window stability,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 5098.1.*

Proof. Project the accepted carrier of Definition 5098.1. The directed rows I, J and the map row ϕ supply the Moore–Smith reindexing surface, while A, T record the finite source values and the cofinal tail ledger.

The observation and Real-facing rows are read after that subnet route. The packet exposes W, D, R, E as the StreamName window, dyadic tolerance, regular rational readback, and terminal Real seal, with E unavailable before the finite observation and readback rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no row carries a selected host subnet, quotient-net equality, selected limit, countable choice, trichotomy, open-cover compactness, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5098.3 (Moore–Smith subnet cofinality route)**.** *Every convergence-facing or completion-facing consumer of an accepted* $\mathsf{MooreSmithSubnet}^{\uparrow}$ *carrier must read the monotone cofinal map row and the cofinal-tail ledger before it can consume the* $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *rows. The public route factors through*

$$I, J, \phi, T \longmapsto W, D, R \longmapsto E \longmapsto H, C, P, N.$$

Proof. Begin with $M = (I, J, \phi, A, T, W, D, R, E, H, C, P, N)$. The subnet data are not represented by an external object; the only available reindexing evidence is the displayed directed pair I, J together with the map row ϕ.

The cofinality condition is then read through T. For each finite source-tail request, T supplies the matching subnet-tail evidence that permits a downstream $\mathsf{NetConvergence}^{\uparrow}$ or $\mathsf{CauchySubnet}^{\uparrow}$ consumer to follow the same tail route.

Only after the cofinal route has been displayed may the packet expose W, D, R and then the terminal seal E. Transport, replay, provenance, and local naming preserve this order through H, C, P, N, so no consumer can bypass cofinality by reading a selected subnet representative or a host convergence predicate. □

Theorem 5098.4 (Moore–Smith subnet nonescape)**.** *An accepted* $\mathsf{MooreSmithSubnet}^{\uparrow}$ *packet exports no selected host subnet, quotient-net equality, selected limit, countable-choice coordinate, open-cover compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim. Every public read remains*

inside the finite rows of Definition 5098.1 and the sibling surfaces $\mathsf{DirectedSubnet}^{\uparrow}$, $\mathsf{CauchySubnet}^{\uparrow}$, $\mathsf{NetConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$.

Proof. Apply Definition 5098.2. The obligations list every row of the accepted carrier and identifies the structural rows as componentwise transport, displayed replay, provenance, and local naming.

Apply Definition 5098.3. The downstream route keeps convergence and completion reads behind the displayed cofinal map, tail ledger, observation windows, tolerance row, regular readback, and terminal seal.

The carrier has no coordinate for the excluded host objects. Therefore no structural transport or replay step can synthesize a selected subnet, quotient equality, selected limit, choice coordinate, compactness principle, or ambient completeness claim from the finite Moore–Smith subnet rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithSubnet}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithSubnet}^{\uparrow}$ packet is generated from source and subnet directed-index rows, a monotone cofinal map, finite net-value observations, a cofinal-tail ledger, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithSubnet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5098.1, Definition 5098.2, Definition 5098.3, and Definition 5098.4 seed the finite Moore–Smith subnet packet for topology and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithSubnet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected host subnets, quotient-net equality, chosen limit points, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-map exactness, cofinal-tail coverage, finite observation stability, Real seal nonescape, transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MooreSmithSubnetUp.

Chapter 5099

A Concrete Naming Certificate for FiniteObservationConfluenceBudget$^{\uparrow}$

The FiniteObservationConfluenceBudget$^{\uparrow}$ packet realizes the MetaCIC residual-confluence frontier as a finite observation surface. It sits beside the residual-substitution boundary of chapter 2181, the bounded residual frontier of chapter 3808, the residual checker of chapter 3827, and the bounded normal-conversion bridge of chapter 3439. The packet records a typed finite window, local join evidence, bounded checker status, obstruction retention, transport, replay, provenance, and local naming without claiming full residual substitution compatibility, subject reduction, or unconditional confluence.

Definition 5099.1 (Finite observation confluence-budget carrier)**.** *A* FiniteObservationConfluenceBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, R, L, B, O, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed finite-window row, recording the closed term, typed context fragment, and observation depth inspected by the packet.*
2. *R is the residual-frontier row, recording only the residual substitutions visible inside the displayed window.*
3. *L is the local-join row, naming the bounded join diagram accepted inside that window.*
4. *B is the bounded-checker row, recording the finite checker budget, checker result, and replay fuel.*
5. *O is the obstruction-retention row, preserving the subject-reduction and residual-substitution boundary that prevents the packet from becoming a full confluence theorem.*
6. *S is the transport row from bounded conversion and finished-normal endpoint evidence into the same finite observation budget.*
7. *H records componentwise* hsame *transport for typed windows, residual frontiers, local joins, bounded checks, obstruction rows, transport rows, replay, provenance, and naming.*
8. *C records displayed* Cont *replay from a candidate confluence request through $T, R, L, B, O, S, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICConfluenceRes}$ $\mathsf{BoundedResidualSubstitutionFrontier}^{\uparrow}$, $\mathsf{BoundedNormalConversionBridge}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through $T, R, L, B, O, S, H, C, P, N$. It has no coordinate for unrestricted residual substitution, global subject reduction, arbitrary closed star confluence, quotient conversion equality, or an unbounded normalizer.

Theorem 5099.2 (Finite observation confluence-budget NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}$ *carrier $F = (T, R, L, B, O, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservat}}$ *surface consists exactly of typed-window admission, residual-frontier admission, local-join recording, bounded-checker exactness, obstruction retention, bounded-conversion transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the finite carrier of Definition 5099.1.*

Proof. Project the accepted carrier. The rows T, R, L, B, O, S are the only mathematical data rows: typed observation window, residual frontier, local join, bounded checker, retained obstruction, and bounded-conversion transport.

The bounded-checker exactness is tied to the local join row, not to an unbounded confluence theorem. The obstruction row remains present in the same carrier, so the accepted packet can expose local candidate evidence while still retaining the subject-reduction and residual-substitution boundary.

Rows H, C, P, N preserve and name the same finite surface. Since the carrier contains no unrestricted residual-substitution, global subject-reduction, arbitrary confluence, quotient-conversion, or unbounded-normalization coordinate, the listed obligations are exact. □

Theorem 5099.3 (Finite observation confluence-budget obstruction retention)**.** *Every accepted* $\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}$ *packet that exposes a local join row also exposes the bounded-checker row and the obstruction-retention row before any MetaCIC confluence consumer can read the packet as evidence. Thus a candidate confluence read factors through*

$$T, R \longmapsto L, B, O \longmapsto S \longmapsto H, C, P, N.$$

Proof. Start with the typed window and residual frontier. The rows T and R determine the finite observation budget in which any local join can be interpreted.

The join row L is accepted only with the bounded checker B. The obstruction row O remains adjacent to that checker, so the packet records the local join together with the precise residual-substitution and subject-reduction boundary still blocking full confluence.

The transport row S and structural rows H, C, P, N replay the same finite budget. They cannot remove the obstruction row or enlarge the bounded checker into a global confluence proof. □

Falsifiable prediction: If the bounded-checker row is removed, no accepted packet can expose both a local join row and a subject-reduction obstruction-retention row.

Independence witness: metacic-residual-substitution-boundary, bounded-residual-substitution-frontier, metacic-confluence-residual-checker, bounded-normal-conversion-bridge: the confluence-budget carrier includes a simultaneous finite observation budget and local-join row not bijective to any single residual or conversion endpoint packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}}$.

Constructive story: A FiniteObservationConfluenceBudget$^{\uparrow}$ packet is generated from typed finite-window rows, residual-frontier rows, local joins, bounded-checker rows, obstruction-retention rows, bounded-conversion transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5099.1, Definition 5099.2, and Definition 5099.3 seed the finite observation confluence-budget packet for MetaCIC residual-confluence consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteObservationConfluenceBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted residual substitution, global subject reduction, arbitrary closed star confluence, quotient conversion equality, unbounded normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-window coverage, residual-frontier exactness, local-join exposure, bounded-checker exactness, obstruction retention, bounded-conversion transport stability, replay stability, provenance exhaustion, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.FiniteObservationConfluenceBudgetU

Chapter 5100

A Concrete Naming Certificate for $\mathsf{FrobeniusNorm}^{\uparrow}$

Following Horn and Johnson on matrix analysis, the $\mathsf{FrobeniusNorm}^{\uparrow}$ packet records the finite square-sum matrix norm as BEDC NameCert data. It sits over the finite matrix carrier of chapter 49, the scalar endpoint discipline of chapter 38, the vector-space transport rows of chapter 47, the norm interface of chapter 89, the orthogonalization route of chapter 1875, and the scalar-bound surface of chapter 3603. The packet names the finite Hilbert–Schmidt ledger for displayed entries; it is not a host supremum, spectral theorem, singular-value theorem, quotient matrix equality, or selected basis construction.

Definition 5100.1 (FrobeniusNorm carrier)**.** *A* $\mathsf{FrobeniusNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (A, E, S, R, N_0, V, G, O, H, C, P, L)$$

with the following displayed rows:

1. *A is the input* $\mathsf{Mat}^{\uparrow}$ *row, exposing a finite row-column table together with its scalar-source and shape ledgers.*
2. *E is the finite entry-read row, listing exactly the displayed matrix entries to be squared and summed.*
3. *S is the square-sum ledger, recording the finite addition route for the displayed real squares of the entries in E.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row carrying the nonnegative square-sum endpoint and the square-root boundary used for the norm value.*
5. *N_0 is the* $\mathsf{Norm}^{\uparrow}$ *comparison row, asserting that the displayed norm coordinate is read from S and R rather than from an ambient operator supremum.*
6. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *scalar-transport row, recording compatibility of the entry table with scalar multiplication and finite matrix addition.*
7. *G is the* $\mathsf{GramSchmidt}^{\uparrow}$ *sibling row used by column-orthogonalization consumers that compare the square-sum ledger with displayed column data.*
8. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *sibling row used by bounded-operator consumers that read the Frobenius ledger as a finite norm bound, not as the defining operator norm.*

9. *H records componentwise* hsame *transport for the matrix row, entry reads, square-sum ledger, real readback, norm comparison, vector-space scalar transport, sibling rows, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a matrix-norm request through $A, E, S, R, N_0, V, G, O, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{FrobeniusNorm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{GramSchmidt}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusNorm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, E, S, R, N_0, V, G, O, H, C, P, L$. It has no coordinate for singular values, spectral decomposition, host supremum over unit vectors, quotient matrix equality, selected orthonormal bases, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FrobeniusNormUp.FrobeniusNormTasteGate single carrier alignment`

Theorem 5100.2 (FrobeniusNorm NameCert obligations)**.** *For every accepted* $\mathsf{FrobeniusNorm}^{\uparrow}$ *carrier*

$$F = (A, E, S, R, N_0, V, G, O, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{FrobeniusNorm}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *matrix admission for the displayed* $\mathsf{Mat}^{\uparrow}$ *row A and its finite entry table E;*
2. *square-sum readback through S and R, requiring every norm value to be read from the finite real square ledger of the displayed entries;*
3. *norm and scalar-transport compatibility through N_0 and V, preserving matrix addition and scalar multiplication at the carrier boundary;*
4. *sibling consumption through G and O, exposing only the Gram–Schmidt column route and the OperatorNorm bound route attached to the same finite ledger;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, L.*

No Frobenius-norm obligation is stored outside these rows.

Proof. Project the carrier of Definition 5100.1. The only matrix data available to the packet are the finite table A and the entry row E.

The square-sum obligation is the ledger S together with its $\mathsf{Real}^{\uparrow}$ endpoint R. Each entry is squared and folded by the displayed finite addition route, so the norm value is read from the square-sum row rather than selected from a host norm function.

The rows N_0 and V record compatibility with the norm interface and vector-space scalar transport. They preserve the finite entry ledger under the displayed matrix operations without adding quotient matrix equality.

Finally the sibling rows G and O expose exactly how Gram–Schmidt and operator-norm consumers may use the same square-sum packet. The structural rows H, C, P, L transport, replay, source, and name only this finite route, so the five listed rows exhaust the local certificate. □

Theorem 5100.3 (FrobeniusNorm square-sum route)**.** *Every* $\mathsf{FrobeniusNorm}^{\uparrow}$ *consumer that reads a matrix norm value must pass through the finite route*

$$A \longmapsto E \longmapsto S \longmapsto R \longmapsto N_0.$$

The route exposes the displayed matrix entries, their real square-sum ledger, and the norm comparison row before any vector-space transport, Gram–Schmidt handoff, or operator-norm bound row is consumed.

Proof. Start from an accepted carrier $F = (A, E, S, R, N_0, V, G, O, H, C, P, L)$. Carrier projection first exposes the matrix table and the finite entry row.

The square-sum ledger S is downstream of those entries, and the $\mathsf{Real}^{\uparrow}$ readback row R is downstream of the ledger. Hence a norm consumer cannot inspect the displayed norm comparison N_0 before the finite entry squares and their sum have been displayed.

The vector-space, Gram–Schmidt, and OperatorNorm rows are sibling-facing consumers of the same route. They may transport or compare the finite ledger, but they do not replace it by a host supremum, spectral calculation, or selected basis. □

Theorem 5100.4 (FrobeniusNorm sibling compatibility)**.** *The* $\mathsf{GramSchmidt}^{\uparrow}$ *row* G *and the* $\mathsf{OperatorNorm}^{\uparrow}$ *row* O *of an accepted* $\mathsf{FrobeniusNorm}^{\uparrow}$ *carrier consume the same displayed square-sum route* A, E, S, R, N_0*. A Gram–Schmidt consumer reads column data through* G *only after the finite entry ledger is fixed, and an OperatorNorm consumer reads* O *only as a finite bound-facing comparison row over that ledger.*

Proof. Apply Definition 5100.3. Both sibling rows occur after the matrix table, entry reads, square-sum ledger, real readback, and norm comparison row.

The Gram–Schmidt handoff therefore receives displayed column data already governed by the same entry ledger. It cannot choose a second basis or alter the square-sum route.

The OperatorNorm handoff receives a finite bound-facing comparison row. It does not convert the Frobenius packet into the host supremum definition of operator norm, because the only available scalar data are S, R, N_0 transported by H and replayed by C. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusNorm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrobeniusNorm}^{\uparrow}$ packet is generated from a displayed MatrixUp table, finite entry reads, real square-sum ledger, RealUp readback, NormUp comparison, VecSpaceUp scalar transport, GramSchmidtUp and OperatorNormUp sibling rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrobeniusNorm}^{\uparrow}}$, seed)

Scope closed: Definition 5100.1, Definition 5100.2, Definition 5100.3, and Definition 5100.4 seed the finite matrix square-sum norm packet and its sibling consumption routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrobeniusNorm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close singular-value theory, spectral decomposition, host operator supremum, quotient matrix equality, selected orthonormal bases, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked matrix admission, square-sum readback, norm comparison, scalar transport, sibling compatibility, transport, replay, provenance, and local naming for BEDC.Derived.FrobeniusNormUp.

Chapter 5101

A Concrete Naming Certificate for $\mathsf{BoundedBilinearMap}^{\uparrow}$

Following Bourbaki's normed-module treatment and Ryan's account of tensor products, the $\mathsf{BoundedBilinearMap}^{\uparrow}$ packet records a two-input bounded map as finite NameCert data rather than as a disguised one-input operator. It is grounded in the algebraic tensor-product surface of chapter 87, the operator-norm rows of chapter 3603, the projective tensor norm route of chapter 4088, and the bounded-linear-operator interface of chapter 1644.

Definition 5101.1 (Bounded bilinear map carrier)**.** *For displayed normed or Banach source rows E, F and a displayed target row G, a* $\mathsf{BoundedBilinearMap}^{\uparrow}(E, F; G)$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (E, F, G, B, K, L, R, T, P, N)$$

with the following rows:

1. *B is a displayed two-input endpoint action on carried endpoints of E and F with target endpoints in G.*
2. *K is a RealUp bound endpoint and L is the finite ledger asserting additivity and scalar compatibility in each input separately.*
3. *R records the boundedness row: every accepted endpoint pair reads through the source norm rows and the estimate named by K before a target norm claim is public.*
4. *T records the tensor handoff row from the two-input action to the displayed* $\mathsf{TensorProduct}^{\uparrow}$ *and* ***??*** *routes.*
5. *P is the* Pkg *provenance over* $\mathsf{BoundedBilinearMap}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, ***??***, BoundedLinearOperator BHist, hsame, Cont, *and* NameCert.
6. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedBilinearMap}^{\uparrow}}$ *row.*

The classifier compares accepted representatives only through componentwise hsame *transport of the two source endpoint rows, the target endpoint row, the bound endpoint, the bilinearity ledger, the boundedness ledger, the tensor handoff row, provenance, and local naming. It has no coordinate for currying into a host function space, quotienting by extensional equality, selecting a completed tensor representative, or importing a bridge.*

Lean4 ✓ `BEDC.Derived.BoundedBilinearMapUp.BoundedBilinearMapTasteGate single carrier alignment`

Theorem 5101.2 (Bounded bilinear map NameCert obligations)**.** *For every accepted* $\mathsf{BoundedBilinearMap}^{\uparrow}$ *carrier* $\mathcal{B} = (E, F, G, B, K, L, R, T, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedBilinearMap}^{\uparrow}}$ *surface consists of carrier admission, separate additivity and homogeneity in both input coordinates, classifier stability under the two source rows and the target row, norm-bound exactness through* K *and* R*, tensor-handoff exactness through* T*, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and refusal of hidden currying, host function extensionality, tensor quotient completion, selected decompositions, and bridge data.*

Proof. Project the finite carrier of Definition 5101.1. The endpoint action B, bound endpoint K, bilinearity ledger L, boundedness ledger R, tensor handoff row T, provenance row P, and local name row N are all displayed before any public read is accepted.

The bilinearity obligations are the four rows in L: additivity in the first input, additivity in the second input, scalar compatibility in the first input, and scalar compatibility in the second input. Each row is stated over carried source endpoints and carried target endpoints, so the proof uses only the packet coordinates and their componentwise transports.

The boundedness obligation is read from R. It exposes the source norm rows and the RealUp bound endpoint K before the target norm estimate is exported. The classifier stability row follows by transporting B, K, L, R, T, P, N along the displayed hsame data for E, F, G.

Finally T gives the tensor handoff and P, N give provenance and local naming. Since the carrier has no row for host currying, host function extensionality, quotient completion, selected tensor decompositions, or bridge checking, those objects cannot appear in the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BoundedBilinearMapUp.BoundedBilinearMapCarrier namecert obligations`

Theorem 5101.3 (Bounded bilinear map operator-norm handoff)**.** *Every operator-norm consumer of an accepted* $\mathsf{BoundedBilinearMap}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{OperatorNorm}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

It may read the bound endpoint K *and the finite boundedness ledger* R *as an operator-norm witness for the bilinear action, but it receives no curried host operator, no host supremum over unit balls, no quotient equality of maps, and no selected norm-attaining pair.*

Proof. Begin with a public operator-norm read. By Definition 5101.1, the only norm-facing coordinates are K and R, together with the endpoint action B and the bilinearity ledger L.

The row R supplies the finite estimate before the consumer reaches $\mathsf{OperatorNorm}^{\uparrow}$. The bounded-linear-operator sibling can consume this data only as a displayed boundedness witness: it sees the two-input action and its ledger, not a host-level curried function.

The non-attainment and quotient exclusions follow from the same projection. No carrier coordinate stores a unit-ball supremum, a selected pair attaining the bound, an extensional quotient of maps, or a bridge row, so the handoff cannot export those objects. □

Theorem 5101.4 (Bounded bilinear map projective tensor route)**.** *Every projective-tensor consumer of an accepted* $\mathsf{BoundedBilinearMap}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{TensorProduct}^{\uparrow}, ??, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The handoff row T *sends elementary tensor reads to the two-input action* B *and sends finite decomposition reads to the finite bound ledger* R*. It does not create a completed projective tensor product, an ambient universal property, a selected minimizing decomposition, or an equality principle beyond the displayed tensor-product classifier.*

Proof. Read the tensor handoff row T from the carrier. For an elementary tensor, T names the route from the displayed tensor-product row to the endpoint action B on the two carried source endpoints.

For a finite projective decomposition, the route repeats this elementary read across the displayed finite list and then invokes the boundedness ledger R for the finite scalar estimate. The construction therefore uses the same finite rows consumed by chapter 4088.

Nothing in the route adds completion data. The tensor-product classifier, finite decomposition ledger, bound row, transport row, replay row, provenance row, and local naming row are the only sources of public information, so completed tensor products, universal host properties, selected minimizing decompositions, and stronger equality principles remain outside the packet. □

Theorem 5101.5 (Bounded bilinear map nonescape boundary)**.** *Every accepted* $\mathsf{BoundedBilinearMap}^{\uparrow}$ *consumer factors through*

$$\mathsf{TensorProduct}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow}, ??, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

In particular, the packet cannot export host currying, host function extensionality, quotient equality of bilinear maps, completed tensor products, selected decompositions, norm-attaining pairs, ambient Banach completion, or bridge checking data.

Proof. Let a consumer read an accepted carrier. The carrier projection exposes only $E, F, G, B, K, L, R, T, P, N$ and the componentwise transport and replay data attached to those rows.

The operator-norm path uses K and R as in Definition 5101.3. The projective-tensor path uses T and then returns to B and R as in Definition 5101.4. Both paths are finite BEDC routes through displayed histories and local naming.

The excluded objects are absent from every coordinate. Since no row stores host currying, extensional equality, quotient equality, a completed tensor product, a selected decomposition, a norm-attaining pair, ambient completion data, or a bridge, no public consumer can read them from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedBilinearMap}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedBilinearMap}^{\uparrow}$ packet is generated from two displayed source rows, one displayed target row, a two-input endpoint action, a RealUp bound endpoint, separate bilinearity rows, a boundedness ledger, a tensor handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedBilinearMap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5101.1, Definition 5101.2, Definition 5101.3, Definition 5101.4, and Definition 5101.5 seed the finite two-input bounded-map surface over tensor-product, operator-norm, projective-tensor-norm, and bounded-linear-operator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedBilinearMap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host currying, host function extensionality, quotient equality of bilinear maps, completed tensor products, selected decompositions, norm-attaining pairs, or ambient Banach completion.

Upgrade path: Obligation closure requires checked carrier admission, separate bilinearity, classifier stability, norm-bound exactness, tensor handoff, operator-norm handoff, projective-tensor route, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedBilinearMapUp.

Chapter 5102

A Concrete Naming Certificate for $\mathsf{KalmanFilter}^{\uparrow}$

Following Kalman and Bucy, the $\mathsf{KalmanFilter}^{\uparrow}$ interface records the finite prediction-and-update packet of a discrete linear estimator. Its BEDC surface consumes $\mathsf{NameCert}_{\mathsf{DynSystem}^{\uparrow}}$ for transition rows, $\mathsf{NameCert}_{\mathsf{ControlObservability}^{\uparrow}}$ for observation rows, $\mathsf{NameCert}_{\mathsf{Mat}^{\uparrow}}$ for covariance and gain updates, $\mathsf{NameCert}_{\mathsf{VecSpace}^{\uparrow}}$ for state and measurement rows, and $\mathsf{NameCert}_{\mathsf{LinMap}^{\uparrow}}$ for the displayed linear maps. Standard references are Kailath, Sayed, and Hassibi, *Linear Estimation* (2000), and Anderson and Moore, *Optimal Filtering* (1979).

The dependency surface is displayed through existing concrete siblings: chapter 329 supplies the state-transition packet, chapter 307 supplies the Riccati covariance ledger, chapter 294 supplies the observability row, chapter 49 and chapter 48 supply the linear algebra rows, chapter 184 and chapter 185 mark the non-exported stochastic boundary, chapter 230 supplies the estimator comparison surface, and chapter 38 supplies the scalar real rows. The deterministic observer-correction sibling chapter 5909 supplies the adjacent state-space, observability, output-error, and gain rows at the non-stochastic boundary of the prediction-update packet. $\mathsf{KalmanFilter}^{\uparrow}$ consumes those rows as finite BHist coordinates before exposing its prediction-update $\mathsf{NameCert}$ surface.

5102.1 Finite carrier

Definition 5102.1 (KalmanFilterUp BHist carrier)**.** *A* $\mathsf{KalmanFilter}^{\uparrow}$ *carrier is a finite* BHist *packet whose visible rows are:*

1. *a state-estimate row* $\widehat{x}$ *and predicted-state row* $\widehat{x}^{-}$ *over* $\mathsf{VecSpace}^{\uparrow}$*;*
2. *an error and covariance ledger row* P^{-}, P, K, P^{+} *over* $\mathsf{Mat}^{\uparrow}$*;*
3. *a transition row* A *and optional input row read from* $\mathsf{DynSystem}^{\uparrow}$*;*
4. *an observation row* C *and finite observation ledger read from* $\mathsf{ControlObservability}^{\uparrow}$*;*
5. *linear-map rows for prediction, observation, innovation, gain application, and estimate transport read from* $\mathsf{LinMap}^{\uparrow}$*;*
6. *matrix update rows for the displayed Riccati-style covariance, gain, and posterior covariance ledgers read from* $\mathsf{Mat}^{\uparrow}$*;*
7. Cont *and* Pkg *provenance rows naming the consumed* $\mathsf{DynSystem}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*, and* $\mathsf{LinMap}^{\uparrow}$ *certificates.*

No host stochastic process, probability measure, analytic convergence theorem, infinite trajectory, or external estimator object is part of the carrier.

Lean4 (definition) `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier`
Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterTasteGate single carrier alignment`

Theorem 5102.2 (KalmanFilterUp prediction and update obligation)**.** *Every accepted* KalmanFilter$^{\uparrow}$ *packet exposes a finite prediction row, innovation row, gain row, and posterior-update row. The prediction row reads the transition and prior estimate through* DynSystem$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$*; the update row reads the observation, innovation, gain, and posterior estimate through* ControlObservability$^{\uparrow}$, Mat$^{\uparrow}$, VecSpace$^{\uparrow}$, *and* LinMap$^{\uparrow}$*. Each row remains a* BHist *row with its displayed* Cont/Pkg *provenance.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier prediction update obligation`

Proof. Inspect the carrier of Definition 5102.1. The prediction rows are exactly the transition, state-estimate, and linear-map rows, while the update rows are exactly the observation, innovation, gain, and posterior-estimate rows.

The matrix rows supply the covariance and gain ledgers, the vector rows supply the state and observation endpoints, and the provenance rows name the certificates consumed by each read. Since the packet has only finite BHist rows, every exported prediction or update entry remains inside the displayed carrier. □

Theorem 5102.3 (KalmanFilterUp observation innovation ledger)**.** *For an accepted* KalmanFilter$^{\uparrow}$ *packet, the observation residual and innovation covariance are read from the displayed observation row of* ControlObservability$^{\uparrow}$*, the predicted estimate row of* VecSpace$^{\uparrow}$*, and the covariance and observation-matrix rows of* Mat$^{\uparrow}$*. The innovation ledger is exact: no unrecorded measurement stream, hidden residual object, or host probabilistic equality contributes to the consumer-visible innovation row.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier observation innovation ledger`

Proof. Read the observation row first. By Definition 5102.1, it is a finite ControlObservability$^{\uparrow}$ row paired with vector, matrix, and linear-map rows.

The residual is the displayed comparison between the observation endpoint and the predicted observation endpoint. The innovation covariance is the displayed matrix update over the covariance and observation rows. Because the carrier excludes host stochastic and trajectory objects, exactness is only the projection from these finite rows. □

Theorem 5102.4 (KalmanFilterUp prediction-update ledger exactness)**.** *For every accepted* KalmanFilter$^{\uparrow}$ *packet, each prediction, innovation, gain, posterior-estimate, and posterior-covariance read factors through the displayed finite* BHist *packet. The packet rows are exactly the* DynSystem$^{\uparrow}$ *transition rows,* ControlObservability$^{\uparrow}$ *observation rows,* Mat$^{\uparrow}$ *covariance and gain rows,* VecSpace$^{\uparrow}$ *state rows,* LinMap$^{\uparrow}$ *linear-map rows, the* Cont *and* hsame *comparison rows, and the* Pkg *provenance rows named by the carrier. No consumer-visible prediction-update read uses a stochastic process, probability space, unrecorded random variable, or external estimator semantics.*

Proof. Project the accepted carrier of Definition 5102.1. Its prediction face consists of the prior estimate, transition, predicted estimate, and predicted covariance rows, together with the LinMap$^{\uparrow}$, Cont, hsame, and Pkg rows that name the finite calculation.

Split the update face into observation, innovation, gain, posterior-estimate, and posterior-covariance reads. The observation and innovation reads are exact by Definition 5102.3, while the gain and posterior rows are the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ entries transported by the same packet.

Estimate transport stability is supplied by Definition 5102.5. Thus any representative change remains inside the same finite ledger, and the excluded stochastic and probability-space objects cannot be recovered from the certificate surface. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier prediction update endpoint transport`

Theorem 5102.5 (KalmanFilterUp estimate transport stability)**.** *If two accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *packets have* hsame*-matched state-estimate, covariance, transition, observation, linear-map,* Cont*, and* Pkg *rows, then prediction, innovation, gain, posterior-estimate, and posterior-covariance reads transport to accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *rows. The classifier is therefore stable under componentwise* hsame *transport of the displayed carrier.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier estimate transport stability`

Proof. Transport each dependency row componentwise along the displayed hsame witnesses. The transition and observation rows remain accepted by their consumed certificates, while the vector, matrix, and linear-map rows remain inside their carrier surfaces.

Apply the same componentwise transport to the prediction, innovation, gain, and posterior rows. The Cont and Pkg entries transport the provenance, so the resulting packet again satisfies the finite carrier inventory of Definition 5102.1. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier estimate transport`

Theorem 5102.6 (KalmanFilterUp NameCert obligation surface)**.** *The local* $\mathsf{NameCert}_{\mathsf{KalmanFilter}^{\uparrow}}$ *obligation surface consists of five finite rows:*

1. *prediction carrier: prior estimate, transition, predicted estimate, and predicted covariance rows;*
2. *update carrier: observation, residual, gain, posterior estimate, and posterior covariance rows;*
3. *innovation ledger exactness: the residual and innovation covariance are exactly the displayed observation, prediction, covariance, and matrix-update reads;*
4. *classifier transport under* hsame*: componentwise transport preserves accepted prediction and update packets;*
5. *downstream read boundary: consumers may read only the prediction, update, innovation, covariance, classifier,* Cont*, and* Pkg *rows displayed by the carrier.*

Proof. The prediction carrier and update carrier are the two action faces of Definition 5102.1. Their rows are finite because each consumed dependency certificate exports finite BHist data.

The innovation row is exact by Definition 5102.3, and classifier stability is the componentwise transport of Definition 5102.5. The consumer boundary is the remaining projection from the same finite inventory. □

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier namecert obligation surface`

Theorem 5102.7 (KalmanFilterUp downstream consumer boundary)**.** *Any algebraic or control-theory consumer of* $\mathsf{NameCert}_{\mathsf{KalmanFilter}^\uparrow}$ *may read only the finite state-estimate, covariance, transition, observation, innovation, gain, posterior-update,* hsame, Cont, *and* Pkg *rows of the carrier. It cannot infer a host probability space, optimality theorem, convergence rate, controllability decomposition, or external trajectory semantics from the* $\mathsf{KalmanFilter}^\uparrow$ *certificate alone.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier endpoint closed generation`
Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier downstream consumer boundary`

Proof. Project the exported rows from Definition 5102.1. These are precisely the finite rows named by the prediction-update obligation and the NameCert obligation surface.

All excluded objects are absent from the carrier inventory. Since consumer reads are projections from accepted BHist rows and their provenance, no consumer can recover a host stochastic or analytic object without an additional certificate. □

Theorem 5102.8 (KalmanFilterUp standard prediction-update bridge)**.** *The public* $\mathsf{KalmanFilter}^\uparrow$ *carrier and consumer boundary interpret exactly as the standard finite discrete prediction-update schema over displayed* $\mathsf{DynSystem}^\uparrow$ *transition rows,* $\mathsf{ControlObservability}^\uparrow$ *observation rows,* $\mathsf{Mat}^\uparrow$ *covariance and gain rows,* $\mathsf{VecSpace}^\uparrow$ *state rows,* $\mathsf{LinMap}^\uparrow$ *linear-map rows, and visible* hsame/Cont/Pkg *provenance. No stochastic process, probability-space semantics, convergence theorem, or external estimator object is exported.*

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterUp StdBridge`

Proof. Start from the carrier of Definition 5102.1. Its rows are the finite state-estimate, covariance, transition, observation, innovation, gain, posterior-update, linear-map, transport, continuation, and package rows.

The prediction-update face is exactly Definition 5102.2. The observation and innovation reads are controlled by Definition 5102.3, and the full ledger is exact by Definition 5102.4.

Representative changes remain inside the same finite packet by Definition 5102.5. The $\mathsf{NameCert}_{\mathsf{KalmanFilter}^\uparrow}$ surface is the finite obligation surface of Definition 5102.6, and downstream consumers are bounded by Definition 5102.7.

Repack those finite rows as the standard prediction-update schema. Since every schema field is one of the displayed BEDC rows and the consumer boundary has no coordinate for stochastic, probabilistic, convergence, or external-estimator data, the interpretation exports exactly the finite prediction-update packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{KalmanFilter}^\uparrow}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KalmanFilter}^\uparrow}$, maturePackageClosed)

Scope closed: Definition 5102.8 binds the public finite Kalman prediction-update packet to $\mathsf{DynSystem}^\uparrow$, $\mathsf{ControlObservability}^\uparrow$, $\mathsf{Mat}^\uparrow$, $\mathsf{VecSpace}^\uparrow$, $\mathsf{LinMap}^\uparrow$, BHist, hsame, Cont, and Pkg rows.

Constructive story: The construction starts from the finite observation packet and dynamic-system transition row, adds matrix and vector rows for prediction, covariance propagation, innovation, gain, and posterior estimate transport, and records the whole calculation as BHist rows with Cont and Pkg provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KalmanFilter}^\uparrow}$, bridgeChecked)

Lean target: `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier\_downstream\_consumer\_boundary`

Bridge status: bridgeChecked

Origin: human

Not claimed: This chapter does not close stochastic optimality, analytic convergence, probability-space semantics, or controllability decomposition.

Upgrade path: No further formal-axis upgrade remains for the finite Kalman packet.

Theorem 5102.9 (KalmanFilterUp zero-gain posterior estimate collapse)**.** *Let $\mathcal{K}$ be an accepted* $\mathsf{KalmanFilter}^{\uparrow}$ *packet in the sense of Definition 5102.1. Write*

$$\widehat{x}^{-}$$

for its predicted-estimate row, K for its gain row, ν for its innovation row, $K\nu$ for the displayed gain-application row, and $\widehat{x}^{+}$ for its posterior-estimate row. Assume that the update face is the displayed affine Kalman estimate row

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} K\nu,$$

with the gain application $K\nu$ read through the $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, *and* $\mathsf{LinMap}^{\uparrow}$ *rows exposed by Definition 5102.2. If the gain row is classified with the zero matrix row,*

$$K \sim_{\mathsf{Mat}^{\uparrow}} Z,$$

where Z is the zero matrix datum of Definition 49.113, then the posterior-estimate row is classified with the predicted-estimate row:

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-}.$$

Lean4 ✓ `BEDC.Derived.KalmanFilterUp.KalmanFilterCarrier zero gain posterior estimate collapse`

Proof. Project the accepted packet by Definition 5102.2. The projection supplies the finite prediction row, innovation row, gain row, posterior-estimate row, and the provenance rows saying that the gain application is read only through the displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ certificates. By Definition 5102.1, no hidden estimator object or stochastic equality is available; every comparison must be assembled from those rows.

The gain hypothesis identifies the carried gain endpoint with the zero matrix datum Z. Matrix classifier stability transports the gain-application row along this comparison, so the displayed product $K\nu$ is classified with the zero-gain product $Z\nu$. The finite-fold zero matrix multiplication law, Definition 49.114, classifies $Z\nu$ with the zero vector endpoint in the codomain vector carrier. Thus the gain-application contribution satisfies

$$K\nu \sim_{\mathsf{VecSpace}^{\uparrow}} 0.$$

The use of Definition 49.114 is entrywise: each scalar summand has a zero matrix factor, the scalar zero-product row reduces it to the scalar zero, and the finite additive fold reduces the whole displayed product to the zero endpoint.

Now apply vector-space addition congruence to the posterior affine row. The comparison $K\nu \sim_{\mathsf{VecSpace}^{\uparrow}} 0$ transports

$$\widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} K\nu$$

to

$$\widehat{x}^{-} +_{\mathsf{VecSpace}^{\uparrow}} 0.$$

The vector additive right-unit row in the $\mathsf{VecSpace}^{\uparrow}$ certificate identifies this endpoint with $\widehat{x}^{-}$. Transitivity in the $\mathsf{VecSpace}^{\uparrow}$ classifier composes the displayed affine update comparison, the transported zero-gain comparison, and the additive right-unit comparison. Hence

$$\widehat{x}^{+} \sim_{\mathsf{VecSpace}^{\uparrow}} \widehat{x}^{-}.$$

All rows used in the argument are among the finite prediction, innovation, gain, posterior-estimate, matrix, vector-space, linear-map, Cont, hsame, and Pkg rows named by Definition 5102.1; the consumer boundary of Definition 5102.7 excludes any additional analytic or probabilistic premise. □

$\mathsf{ControlRiccati}^{\uparrow}$ routes its finite Riccati value, gain-readback, and regulator rows through chapter 307, chapter 296, and chapter 49 before the child file opens the local NameCert surface.

Chapter 5103

A Concrete Naming Certificate for $\mathsf{ControlRiccati}^{\uparrow}$

Following Kalman and Anderson–Moore, the $\mathsf{ControlRiccati}^{\uparrow}$ packet records the control-facing Riccati certificate surface that sits between finite Riccati ledgers and regulator or estimator consumers. It reads the displayed Riccati recursion of chapter 307, the finite regulator packet of chapter 296, the estimator packet of chapter 5102, the observability and controllability rows of chapter 294 and chapter 297, the finite matrix rows of chapter 49, and the normed-state boundary of chapter 1950. The packet names a finite stabilizing residual and gain-readback route; it is not an ambient algebraic Riccati solver.

Definition 5103.1 (ControlRiccati carrier)**.** *A* $\mathsf{ControlRiccati}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, A, B, Q, S, G, L, O, K, H, T, P, N)$$

with the following displayed rows:

1. *R is the source* $\mathsf{Riccati}^{\uparrow}$ *value-ledger row.*
2. *A, B are the displayed state-transition and control-input matrix rows read through* $\mathsf{Mat}^{\uparrow}$ *and the finite state/control carrier.*
3. *Q, S are the displayed state-cost and control-cost matrix rows, including the symmetry and positivity ledgers needed by the control read.*
4. *G is the finite gain-readback row naming the feedback matrix computed from the displayed Riccati ledger.*
5. *L is the* $\mathsf{LQR}^{\uparrow}$ *handoff row saying the regulator consumer reads the same Riccati value and gain rows.*
6. *O is the* $\mathsf{ControlObservability}^{\uparrow}$ *row and K is the* $\mathsf{ControlControllability}^{\uparrow}$ *row used to keep the residual and gain checks inside the displayed control surface.*
7. *H is the residual-stability row: under the displayed normed-state and matrix rows, the Riccati residual remains inside the finite tolerance ledger exported by* $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$*.*
8. *T is componentwise* hsame *transport for the Riccati value, matrix, cost, gain, observability, controllability, and residual rows.*
9. *P is* Pkg *provenance over* $\mathsf{Riccati}^{\uparrow}$*,* $\mathsf{LQR}^{\uparrow}$*,* $\mathsf{KalmanFilter}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*, and* $\mathsf{NormedSpace}^{\uparrow}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ControlRiccati}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host matrix equation, a selected stabilizing solution, an infinite-horizon limit, a spectral separation oracle, or an unlogged controller.

Theorem 5103.2 (ControlRiccati NameCert obligations)**.** *For the carrier of Definition 5103.1, the local* $\mathsf{NameCert}_{\mathsf{ControlRiccati}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the finite packet $C = (R, A, B, Q, S, G, L, O, K, H, T, P, N)$;*
2. *Riccati residual stability through the displayed* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{NormedSpace}^{\uparrow}$ *rows;*
3. *gain-readback exactness, saying the feedback row G is read only from the displayed Riccati, transition, input, and cost rows;*
4. *LQR handoff, saying the* $\mathsf{LQR}^{\uparrow}$ *consumer receives the same value and gain rows through L;*
5. *no-host-matrix nonescape, saying no accepted consumer can project an algebraic Riccati equation, selected stabilizing solution, spectral oracle, or unbounded controller from the packet.*

Proof. Project the carrier of Definition 5103.1. The rows $R, A, B, Q, S, G, L, O, K, H, T, P, N$ are the only visible Riccati value, matrix, cost, gain, regulator, observability, controllability, residual, transport, provenance, and local naming rows.

The residual-stability obligation reads the finite residual row H together with the matrix and normed-state rows. Since the residual is compared inside the displayed $\mathsf{Mat}^{\uparrow}$ and $\mathsf{NormedSpace}^{\uparrow}$ ledgers, the check cannot call on a host spectral or convergence theorem.

The gain-readback obligation then reads G from the same Riccati value, transition, input, and cost rows. The $\mathsf{LQR}^{\uparrow}$ handoff row L exports exactly this finite value-and-gain surface to the regulator packet, while the $\mathsf{KalmanFilter}^{\uparrow}$ dependency is only the neighboring estimator use of Riccati-style covariance ledgers.

Finally inspect the excluded coordinates. A host matrix equation, selected stabilizing solution, infinite-horizon limit, spectral separation oracle, and unlogged controller would require rows not present in C. Therefore every accepted $\mathsf{ControlRiccati}^{\uparrow}$ read remains inside the displayed NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ControlRiccati}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ControlRiccati}^{\uparrow}$ packet is generated from a Riccati value ledger, displayed transition and input matrices, cost matrices, gain readback, LQR handoff, observability and controllability rows, residual stability, componentwise transport, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ControlRiccati}^{\uparrow}}$, seed)

Scope closed: Definition 5103.1 and Definition 5103.2 seed the finite control-Riccati carrier and its five NameCert obligation rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ControlRiccati}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close algebraic Riccati existence, stabilizing uniqueness, infinite-horizon convergence, spectral separation, host matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual stability, gain-readback exactness, LQR handoff, and no-host-matrix nonescape for BEDC.Derived.ControlRiccatiUp.

The $\mathsf{RoucheTheorem}^{\uparrow}$ hub routes contour-zero stability through the residue, Cauchy-integral, contour-socket, complex-differentiability, and real-comparison siblings recorded in chapter 125, chapter 4136, chapter 583, chapter 67, and chapter 38.

Chapter 5104

A Concrete Naming Certificate for $\mathsf{RoucheTheorem}^{\uparrow}$

Following Ahlfors and Conway, the $\mathsf{RoucheTheorem}^{\uparrow}$ packet names the finite contour-zero stability surface that lets a holomorphic perturbation be compared with a base holomorphic function along a displayed boundary. It sits beside the residue ledger of chapter 125, the point-evaluation route of chapter 4136, the contour socket of chapter 583, and the complex differentiability and real-comparison infrastructure already exposed by their neighbouring chapters. Its zero-count read is explicitly routed through the argument-principle and Riemann zero-count ledger siblings before the Rouche boundary-dominance row exports stability. The chapter records only certificate-bound contour admissibility, boundary dominance, and finite zero-count ledger handoffs; it does not claim global analytic continuation, host winding-number completeness, or an unrestricted homotopy theorem.

Definition 5104.1 (Rouche theorem carrier)**.** *A* $\mathsf{RoucheTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, F, G, B, Z, I, L, H, P, N)$$

with the following displayed rows:

1. *C is the contour-admissibility row, naming the finite boundary contour and the containment witness for the analytic region being inspected.*
2. *F is the base holomorphic row for f, including the local complex differentiability and contour-avoidance data used by the zero-count ledger.*
3. *G is the perturbation holomorphic row for g, displayed over the same contour window and the same complex arithmetic surface.*
4. *B is the boundary-dominance row, recording the finite comparison $|g| < |f|$ on the displayed contour boundary through the real-order rows available to the packet.*
5. *Z is the zero-count ledger row, listing the finite interior zero accounting read for f and for $f+g$.*
6. *I is the residue and Cauchy-integral handoff row, routing the zero-count comparison through the residue chapter and the Cauchy integral formula chapter without adding a second analytic oracle.*
7. *L is the stability ledger row, recording that the two displayed zero-count ledgers agree inside the accepted contour window.*

8. *H records componentwise* hsame *transport for contour, holomorphic, boundary-dominance, zero-count, integral-handoff, stability, provenance, and local naming rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RoucheTheorem}^{\uparrow}$, $\mathsf{ContourIntegralSocket}^{\uparrow}$, $\mathsf{Residue}^{\uparrow}$, $\mathsf{CauchyIntegralFormula}^{\uparrow}$, $\mathsf{ComplexDifferentiability}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RoucheTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $C, F, G, B, Z, I, L, H, P, N$. It has no coordinate for a host analytic-continuation theorem, unbounded root isolation, quotient path equality, or classical compactness oracle outside the displayed packet.

The local NameCert obligation sketch has five grouped rows:

1. contour admission through C;
2. paired holomorphic source admission through F, G;
3. boundary dominance through B;
4. finite zero-count and residue/Cauchy-integral handoff through Z, I;
5. stability, transport, provenance, and local naming through L, H, P, N.

Theorem 5104.2 (Rouche theorem NameCert obligations)**.** *For every accepted* $\mathsf{RoucheTheorem}^{\uparrow}$ *carrier $R = (C, F, G, B, Z, I, L, H, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RoucheTheorem}^{\uparrow}}$ *surface consists exactly of contour admissibility through C, holomorphic source rows for f and g through F, G, boundary dominance through B, finite zero-count ledger exposure through Z, residue and Cauchy-integral handoffs through I, stability-ledger exactness through L, componentwise* hsame *transport through H,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier. The accepted rows are exactly $C, F, G, B, Z, I, L, H, P, N$, so the NameCert surface is finite and componentwise visible.

The contour row C supplies the only boundary window. The holomorphic rows F and G are read on that same window, and the dominance row B is attached to the displayed boundary comparison $|g| < |f|$ rather than to an ambient analytic theorem.

The zero-count ledger Z and handoff row I route the comparison through the residue and Cauchy-integral neighbours. The stability row L records the resulting equality of displayed ledgers, while H, P, N transport, source, and name the same data. Since no carrier coordinate stores global analytic continuation or unbounded root isolation, the listed rows exhaust the obligations. □

Theorem 5104.3 (Rouche theorem contour zero stability)**.** *If an accepted* $\mathsf{RoucheTheorem}^{\uparrow}$ *packet displays contour admissibility C, holomorphic rows F, G for f and g, boundary dominance B with $|g| < |f|$ on the displayed contour boundary, residue and Cauchy-integral handoffs I, and a finite zero-count ledger Z, then the stability ledger L preserves the zero-count read for f and $f + g$ inside that contour window. The claim is only the certificate-bound equality of displayed ledgers; it is not a claim of global analytic continuation, unrestricted homotopy invariance, or host completeness for all contours.*

Proof. Start with the contour row C. It fixes the finite boundary and interior window in which the two holomorphic rows F and G can be compared.

The boundary-dominance row B is read next. Because B is part of the packet, the comparison $|g| < |f|$ is visible at the same contour boundary rather than imported as an external maximum principle.

Use the handoff row I to pass through the residue and Cauchy-integral routes named in chapter 125 and chapter 4136. These handoffs identify the finite zero-count reads recorded in Z.

Finally L records equality of the two displayed zero-count ledger entries, and H, P, N preserve that equality under transport, provenance, and local naming. No row supplies global analytic continuation or unrestricted contour completeness, so the result remains the stated certificate-bound stability claim. □

Theorem 5104.4 (Rouche theorem holomorphic source route)**.** *Every zero-count stability read of an accepted* $\mathsf{RoucheTheorem}^{\uparrow}$ *carrier* $R = (C, F, G, B, Z, I, L, H, P, N)$ *must expose the* $\mathsf{Holomorphic}^{\uparrow}$ *and* $\mathsf{CplxDiff}^{\uparrow}$ *source rows* F, G*, the zero-count route through* $\mathsf{ArgumentPrinciple}^{\uparrow}$ *or* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$*, the residue handoff, the Cauchy-integral formula handoff, and the same contour admissibility row* C *before the stability ledger* L *is public. The route is therefore a horizontal analytic dependency ledger over the displayed rows, not a hidden analytic-continuation oracle or an unrestricted contour homotopy principle.*

Proof. Begin with the contour and holomorphic rows. The row C fixes the admissible contour, while F and G expose the paired $\mathsf{Holomorphic}^{\uparrow}$ sources together with their $\mathsf{CplxDiff}^{\uparrow}$ pointwise data on the same displayed window.

Next read the zero-count route. The ledger Z may be consumed through the argument-principle sibling of chapter 7193 or through the Riemann zero-count ledger of chapter 573, but in either case the route remains attached to the same contour row and boundary-dominance row B.

Finally apply the analytic handoffs and structural rows. The row I records the residue and Cauchy-integral formula handoffs of chapter 125 and chapter 4136; then L, H, P, N publish, transport, source, and name the resulting stability ledger. No displayed coordinate supplies an ambient analytic-continuation oracle or unrestricted homotopy theorem. □

Theorem 5104.5 (Rouche theorem holomorphic winding dependency)**.** *Any accepted* $\mathsf{RoucheTheorem}^{\uparrow}$ *zero-count consumer must read the* $\mathsf{Holomorphic}^{\uparrow}$ *or complex-differentiability source rows* F, G*, the contour integral socket, and the* $\mathsf{ArgumentPrinciple}^{\uparrow}$ *or* $\mathsf{RiemannZeroCountLedger}^{\uparrow}$ *ledger before the boundary-dominance row* B *can export zero-count stability through* L*. The Rouche packet therefore publishes zero-count stability only as a route through the named holomorphic and winding-number boundaries, not as a detached contour-zero oracle.*

Proof. Start with the holomorphic route of Definition 5104.4. It exposes the paired holomorphic and complex-differentiability rows, the argument-principle or Riemann zero-count ledger route, and the contour row before the stability ledger is public.

Apply contour zero stability from Definition 5104.3. The row B can compare f and g only on the displayed contour boundary, and the stability row L records equality of the displayed zero-count ledgers after the residue and Cauchy-integral handoffs are visible.

The contour socket and ledger route are therefore prerequisites for the Rouche export. Since no carrier coordinate stores ambient analytic continuation, host winding-number completeness, or unrestricted homotopy invariance, the zero-count consumer receives only the named holomorphic and winding-boundary dependency. □

Falsifiable prediction: A packet that omits contour admissibility C, either holomorphic row F, G, the boundary-dominance row B, or the residue and Cauchy-integral handoff row I cannot be accepted as a $\mathsf{RoucheTheorem}^{\uparrow}$ packet, even if it records a candidate zero-count equality.

Independence witness: residue, cauchy-integral-formula, contourintegralsocket, complex-differentiability, real: $\mathsf{RoucheTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds contour admissibility, paired holomorphic rows, boundary dominance, zero-count ledgers, residue and Cauchy-integral handoffs, stability, transport, provenance, and local naming into one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RoucheTheorem}\uparrow}$.

Constructive story: A $\mathsf{RoucheTheorem}^{\uparrow}$ packet is generated from contour-admissibility rows, paired holomorphic rows, boundary-dominance rows, finite zero-count ledgers, residue and Cauchy-integral handoffs, stability ledgers, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RoucheTheorem}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5104.1, Definition 5104.2, Definition 5104.3, and Definition 5104.4 seed the finite contour-zero stability certificate over existing holomorphic, complex-differentiability, contour, zero-count, residue, Cauchy-integral, real-comparison, transport, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RoucheTheorem}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global analytic continuation, unrestricted homotopy invariance, host winding-number completeness, unbounded root isolation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked contour admissibility, paired holomorphic rows, boundary-dominance comparison, residue and Cauchy-integral handoffs, zero-count ledger exactness, stability transport, and non-escape for BEDC.Derived.RoucheTheoremUp.

Chapter 5105

A Concrete Naming Certificate for $\mathsf{FiniteIntervalCover}^{\uparrow}$

Following Bishop and Bridges, finite interval covers are read here as displayed finite data rather than as open-cover prose. The $\mathsf{FiniteIntervalCover}^{\uparrow}$ packet realizes the finite-cover branch of chapter 9237 beside the located interval surface of chapter 394, the dyadic interval ledger of chapter 306, the closed bounded interval packet of chapter 1591, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. A consumer sees a finite list of interval rows, endpoint and overlap ledgers, coverage witnesses, transport, replay, provenance, and local naming; it does not see a selected subcover, quotient real cover, endpoint convention selector, trichotomy oracle, or ambient real-completeness principle.

Definition 5105.1 (FiniteIntervalCover carrier)**.** *A* $\mathsf{FiniteIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, D, B, W, C, R, E, H, K, P, N)$$

with the following visible rows:

1. *I is the finite interval-index row, recording the displayed cover members.*
2. *D is the* $\mathsf{DyadicInterval}^{\uparrow}$ *endpoint row for the cover windows.*
3. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row being covered.*
4. *W is the* $\mathsf{LocatedInterval}^{\uparrow}$ *witness row for membership and overlap inspections.*
5. *C is the finite coverage ledger, recording that every accepted source-window read is captured by one displayed interval row.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational endpoint comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for real-facing endpoint reads.*
8. *H records componentwise* hsame *transport for I, D, B, W, C, R, E.*
9. *K records displayed* Cont *replay for index lookup, endpoint comparison, coverage inspection, and Real-seal handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalCover}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for a choice-selected refinement, quotient real interval, ambient compactness theorem, trichotomy decision, or hidden endpoint convention.

Lean4 ✓ `BEDC.Derived.FiniteIntervalCoverUp.FiniteIntervalCoverTasteGate single carrier alignment`

Theorem 5105.2 (FiniteIntervalCover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalCover}^{\uparrow}$ *packet* $F = (I, D, B, W, C, R, E, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalCover}^{\uparrow}}$ *surface consists exactly of finite index admission, dyadic endpoint exposure, closed-bounded source admission, located interval inspection, finite coverage, regular rational readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every cover consumer must read the sibling routes named in chapter 394, chapter 306, chapter 1591, chapter 126, chapter 301, and chapter 38 before it treats the cover as a real-facing interval cover.*

Proof. Project the carrier of Definition 5105.1. The rows I, D, B, W, C are the finite cover data: they display the interval indices, the dyadic endpoints, the closed bounded source, located membership and overlap checks, and coverage witnesses.

Read the comparison and seal rows next. The route from D and W to a real-facing endpoint passes through R and then E, so a consumer cannot replace finite endpoint comparisons by quotient real equality, a selected representative, or trichotomy.

Finally transport the same rows through H, K, P, N. Since the carrier contains no additional coordinate, the listed rows exhaust the local NameCert obligations and force every accepted cover read through the displayed sibling lattice. □

Theorem 5105.3 (FiniteIntervalCover Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{FiniteIntervalCover}^{\uparrow}$ *packet first consumes the finite interval-index row* I*, dyadic endpoint row* D*, closed-bounded source row* B*, located interval row* W*, coverage row* C*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *comparison row* R*. Only after those rows have been transported through* H*, replayed through* K*, packaged by* P*, and named by* N *may the terminal Real seal* E *be exposed to a downstream compactness or uniform-modulus consumer.*

Proof. Start with Definition 5105.2. It makes the finite index, endpoint, source, locatedness, coverage, regular-rational, and Real-seal rows part of one local NameCert surface.

The real-facing handoff is therefore not a direct map from closed intervals to real endpoints. It must inspect the finite cover member, compare its displayed dyadic endpoints through R, and retain the closed-bounded source row B and located interval row W before E is available.

The structural rows then preserve the route. Transport by H, replay by K, provenance by P, and local naming by N keep the same packet visible to compactness and uniform-modulus consumers, while the absence of selected refinements, quotient intervals, trichotomy, and ambient completeness prevents any detached Real read. □

Falsifiable prediction: An accepted $\mathsf{FiniteIntervalCover}^{\uparrow}$ read is refuted if a RealUp endpoint seal is exposed without the finite index row, dyadic endpoint row, closed-bounded source row, located interval row, coverage row, RegSeqRat readback, transport, replay, provenance, and local naming rows.

Independence witness: locatedinterval, dyadicinterval, closedboundedinterval, streamname, regseqrat, and real: none is bijective to the finite interval-cover packet because only $\mathsf{FiniteIntervalCover}^{\uparrow}$ records finite interval indices, coverage witnesses, endpoint comparisons, and the Real handoff in one local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalCover}^{\uparrow}}$**.**

Constructive story: A FiniteIntervalCover$^{\uparrow}$ packet is generated from finite interval indices, dyadic endpoint rows, a closed-bounded source row, located interval inspection, finite coverage witnesses, RegSeqRat comparison readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteIntervalCover}\uparrow}$, seed)

Scope closed: Definition 5105.1, Definition 5105.2, and Definition 5105.3 seed the finite interval-cover object consumed by compactness and uniform-modulus routes.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteIntervalCover}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected subcovers, endpoint-convention selectors, trichotomy, quotient real intervals, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coverage, dyadic endpoint comparison, located interval inspection, RegSeqRat readback, RealUp handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalCoverUp.

Chapter 5106

A Concrete Naming Certificate for $\mathsf{ConstructiveHeineBorel}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ packet records the finite-subcover route for a closed real interval using only displayed interval endpoints, a dyadic mesh, finite cover requests, selected finite subcover rows, and Real readback. It consumes the real carrier of chapter 38, the dyadic arithmetic discipline of chapter 291, the StreamName window surface of chapter 126, the regular rational sequence readback of chapter 301, and the interval variant in chapter 1379. It is a finite NameCert packet for an interval-cover search, not a claim of arbitrary open-cover compactness.

Definition 5106.1 (Constructive Heine–Borel finite-subcover carrier)**.** *A* $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, D, G, F, S, W, Q, R, H, C, P, N)$$

with the following visible rows:

1. *A and B are the closed interval endpoint rows, read through* $\mathsf{Real}^{\uparrow}$ *seals and regular rational names;*
2. *D is the dyadic mesh row, recording a finite dyadic partition and its endpoint refinement data through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
3. *G is the finite cover-request row, recording finitely many interval-cover tests against the displayed mesh windows;*
4. *F is the finite-subcover selection row, storing the displayed finite list of cover indices that answer the tested mesh;*
5. *S is the StreamName window row, exposing the finite windows used to read the endpoints and test membership;*
6. *W is the RegSeqRat readback row, transporting the endpoint and mesh reads through regular rational sequence data;*
7. *Q is the Real seal row, recording the interval and cover reads as* $\mathsf{Real}^{\uparrow}$ *data without quotient real equality;*
8. *R records refusal of arbitrary open-cover compactness, choice-selected subcovers, quotient real equality, and ambient Real completeness;*

9. *H stores componentwise* hsame *transport for endpoint, mesh, cover-request, finite-subcover, stream-window, regular-sequence, Real-seal, refusal, replay, provenance, and local naming rows;*
10. *C stores displayed* Cont *replay for the same finite route;*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveHeineBorel}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{HeineBorelInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an arbitrary open cover, a non-finite subcover, a choice-selected subcover, quotient equality of reals, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ConstructiveHeineBorelUp.ConstructiveHeineBorelTasteGate single carrier alignment`

Theorem 5106.2 (Constructive Heine–Borel NameCert obligations)**.** *For the carrier of Definition 5106.1, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorel}^{\uparrow}}$ *surface consists exactly of endpoint admission, dyadic mesh admission, finite cover-request coverage, finite-subcover selection, StreamName window readback, RegSeqRat readback, Real seal preservation, refusal accountability, transport, replay, provenance, and local naming.*

Proof. Project the carrier $K = (A, B, D, G, F, S, W, Q, R, H, C, P, N)$. The endpoint rows A, B, dyadic mesh row D, cover-request row G, and finite-subcover row F are the only analytic coordinates of the packet.

The readback rows then bind the finite search to the real substrate. StreamName windows in S expose the finite interval reads, RegSeqRat rows in W replay the rational approximations, and the Real seal Q keeps the endpoint and cover tests inside displayed $\mathsf{Real}^{\uparrow}$ data.

Finally the refusal row R blocks the classical readings, while H, C, P, N transport, replay, source, and name the same rows. Since no further coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 5106.3 (Constructive Heine–Borel finite-subcover route)**.** *Every accepted* $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ *packet exports a finite-subcover reading only through the route*

$$A, B, D, G, F, S, W, Q, R, H, C, P, N.$$

The selected finite subcover row F must be read after the endpoint rows, dyadic mesh row, and finite cover-request row, and before the transport, replay, provenance, and local naming rows. The packet does not export an arbitrary open-cover compactness theorem or a choice-selected subcover.

Proof. Begin with the endpoint and mesh data. The rows A, B, D display the closed interval endpoints and the dyadic partition on which the finite search is performed.

Next read the cover request. The row G records only finitely many tested cover windows, so the subcover row F selects from a finite displayed list rather than from an ambient open cover.

The readback and structural rows preserve this route. Rows S, W, Q attach the StreamName, RegSeqRat, and Real seal data, while H, C, P, N keep the route stable under transport, replay, provenance, and naming. The refusal row R excludes the classical readings named in the statement. □

Theorem 5106.4 (Constructive Heine–Borel Real-seal nonescape)**.** *Let $K = (A, B, D, G, F, S, W, Q, R, H, C, P, N)$ be an accepted* $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ *packet. Any Real-facing consumer of the finite-subcover route must read the StreamName windows S, RegSeqRat readback W, and Real seal Q together with the endpoint, mesh, cover, finite-subcover, refusal, transport, replay, provenance, and naming rows. It cannot replace this route by quotient real equality, ambient Real completeness, or an unlogged compactness principle.*

Proof. Use Definition 5106.2. It places the finite endpoint, mesh, cover-request, finite-subcover, StreamName, RegSeqRat, and Real-seal rows in one packet.

The Real-facing route is therefore not an ambient equality route. The seal Q is read only after S and W expose the finite windows and regular rational readbacks that generated the endpoint and mesh tests.

Apply Definition 5106.3 to close the nonescape boundary. A consumer that bypasses S, W, Q or imports ambient completeness no longer reads the accepted $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ carrier. □

Theorem 5106.5 (Constructive Heine–Borel finite-net extraction)**.** *Every admitted finite-subcover read of an accepted* $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ *packet*

$$K = (A, B, D, G, F, S, W, Q, R, H, C, P, N)$$

factors through the endpoint rows A, B, the dyadic mesh row D, the finite cover-request row G, the finite-subcover row F, the $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback W, and the* $\mathsf{Real}^{\uparrow}$ *seal Q before transport, replay, provenance, or local naming is consumed. The extraction exports a finite-net replacement for the tested interval cover and exports no arbitrary open-cover compactness theorem, choice-selected subcover, quotient real equality, or ambient Real completeness row.*

Proof. Project the carrier of Definition 5106.1. The analytic rows are exactly the endpoint rows, dyadic mesh, finite cover request, finite subcover, StreamName window, RegSeqRat readback, Real seal, and refusal row.

Use the NameCert obligations of Definition 5106.2. They bind endpoint admission, mesh admission, finite cover-request coverage, finite-subcover selection, StreamName readback, RegSeqRat readback, Real sealing, and refusal accountability into one local surface.

Apply the finite-subcover route theorem Definition 5106.3. It forces F to be read only after the endpoint, dyadic mesh, and finite cover-request rows, so the extracted net is finite and displayed rather than selected from an arbitrary open cover.

Finally apply the Real-seal nonescape theorem Definition 5106.4. The rows S, W, Q keep the endpoint and cover tests inside the displayed StreamName, RegSeqRat, and Real-seal route, while R, H, C, P, N block the excluded classical readings and preserve the same packet. □

Falsifiable prediction: A finite-subcover consumer that exports a subcover without reading D, G, F, S, W, Q refutes the $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ carrier.

Independence witness: real, dyadicratcore, streamname, regseqrat, heineborelinterval: the real carrier supplies sealed values without a finite cover search, the dyadic core supplies mesh arithmetic without interval coverage, StreamName and RegSeqRat supply readback without subcover selection, and the interval variant supplies interval-facing constraints without this finite endpoint-mesh-cover packet. None is bijective to $\mathsf{ConstructiveHeineBorel}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorel}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveHeineBorel}^{\uparrow}$ packet is generated from endpoint, dyadic mesh, finite cover-request, finite-subcover, StreamName, RegSeqRat, Real-seal, refusal, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\mathsf{ConstructiveHeineBorel}\uparrow}$, seed)

Scope closed: Definition 5106.1, Definition 5106.2, Definition 5106.3, and Definition 5106.4 seed the finite-subcover NameCert route for a closed interval.

Formal verification: FormalStatus(NameCert$_{\mathsf{ConstructiveHeineBorel}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, choice-selected subcovers, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint, dyadic mesh, finite cover-request, finite-subcover, StreamName, RegSeqRat, Real-seal, and refusal rows for BEDC.Derived.ConstructiveHeineBorelUp.

Chapter 5107

A Concrete Naming Certificate for $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$

The $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$ packet realizes the boundary fixed-point reading of section 9463.6 as finite NameCert data. Its obstruction row consumes disk coordinates from chapter 377, metric comparison from chapter 3474, observer-expansion rows from chapter 4167, and Busemann-boundary readback from chapter 3732 before refusing an interior fixed-point reading of the apophatic far end.

Definition 5107.1 (Hyperbolic fixed-point obstruction carrier)**.** *A* $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, M, B, E, U, G, R, H, C, P, N)$$

with the following visible rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate row for the displayed disk surface;*
2. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row for disk-distance and boundary comparison reads;*
3. *B is the Denjoy–Wolff boundary-dynamics row, recording the boundary route rather than an interior fixed point;*
4. *E is the hyperbolic observer-expansion row, recording the expansion ledger whose far-end reading is being inspected;*
5. *U is the Busemann-boundary row, exposing boundary-level readback for the far end;*
6. *G is the obstruction row, refusing an interior fixed-point reading of the apophatic far end;*
7. *R records refusal of a global disk center, a value fixed point for* T*, a hidden observer point, and a completed hyperbolic universe;*
8. *H stores componentwise* hsame *transport for disk, metric, boundary dynamics, observer expansion, Busemann boundary, obstruction, refusal, replay, provenance, and local naming rows;*
9. *C stores displayed* Cont *replay for the same route;*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$*,* $\mathsf{UnitDisk}^{\uparrow}$*,* $\mathsf{HyperbolicMetric}^{\uparrow}$*, Denjoy–Wolff boundary dynamics, hyperbolic observer expansion, hyperbolic Busemann boundary,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*

11. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an interior value of T*, a global disk center, a hidden Hilbert-space fixed point, or a completed ambient hyperbolic geometry.*

Theorem 5107.2 (Hyperbolic fixed-point obstruction NameCert obligations)**.** *For the carrier of Definition 5107.1, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, hyperbolic-metric admission, boundary-dynamics coverage, observer-expansion coverage, Busemann-boundary readback, obstruction refusal, transport, replay, provenance, and local naming.*

Proof. Project the carrier $O = (D, M, B, E, U, G, R, H, C, P, N)$. The rows D, M, B, E, U display the disk, metric, boundary-dynamics, observer-expansion, and boundary-readback data that can be inspected.

The obstruction row G is then the only row that states the fixed-point refusal. It may compare the displayed boundary route with the disk and metric rows, but it cannot introduce an interior coordinate for the far end.

Finally R, H, C, P, N record the forbidden readings, transport the route, replay it, source it, and name it. Since the carrier has no further coordinate, these rows exhaust the local NameCert obligations. □

Theorem 5107.3 (Hyperbolic fixed-point boundary-route obstruction)**.** *Every accepted* HyperbolicFixedPointObstr *packet forces the apophatic far-end reading through the boundary route*

$$D, M, B, E, U, G, R, H, C, P, N.$$

A consumer may read boundary dynamics, observer expansion, and Busemann-level boundary data, but it cannot convert those rows into an interior disk fixed point, a global center, or a value of T*.*

Proof. Start with the disk and metric rows. The packet displays D and M only as coordinates and comparisons for the visible disk surface.

Next read the boundary rows. The Denjoy–Wolff-style row B, observer-expansion row E, and Busemann-boundary row U place the fixedness language at the boundary route, not at an interior point.

The obstruction row G and refusal row R block the forbidden conversions. Transport, replay, provenance, and naming through H, C, P, N preserve the same finite route, so the accepted packet exports a boundary-route obstruction and nothing stronger. □

Falsifiable prediction: A hyperbolic observer-expansion consumer that obtains an interior disk point for T without reading the boundary rows B, E, U, G refutes the $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$ packet.

Independence witness: unitdisk, hyperbolic-metric, denjoy-wolff-boundary-dynamics, hyperbolic-observer-expansion, hyperbolic-busemann-boundary: the UnitDisk packet supplies disk coordinates without fixed-point refusal, the HyperbolicMetric packet supplies metric comparison without apophatic boundary routing, the Denjoy–Wolff boundary-dynamics packet supplies boundary dynamics without observer-expansion obstruction, the observer-expansion packet supplies expansion rows without Busemann boundary readback, and the Busemann-boundary packet supplies boundary-level readback without the full obstruction carrier. None is bijective to $\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicFixedPointObstruction}^{\uparrow}}$**.**

Constructive story: A HyperbolicFixedPointObstruction$^{\uparrow}$ packet is generated from disk, hyperbolic metric, boundary-dynamics, observer-expansion, Busemann-boundary, obstruction, refusal, transport, replay, provenance, and local naming rows.

Theory closure: TheoryGate(NameCert$_{\text{HyperbolicFixedPointObstruction}^{\uparrow}}$, seed)

Scope closed: Definition 5107.1, Definition 5107.2, and Definition 5107.3 seed the finite boundary-route obstruction to interior fixed-point readings.

Formal verification: FormalStatus(NameCert$_{\text{HyperbolicFixedPointObstruction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a global disk center, an interior value for T, host hyperbolic dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk, metric, boundary-dynamics, observer-expansion, Busemann-boundary, obstruction, and refusal rows for BEDC.Derived.HyperbolicFixedPointObstructionUp.

Chapter 5108

A Concrete Naming Certificate for $\mathsf{HardySpace}^{\uparrow}$

Following Hardy on boundary values of analytic functions and the classical Hardy-space tradition, the $\mathsf{HardySpace}^{\uparrow}$ packet records the finite boundary-read surface needed by constructive real-name analysis. It sits after the Hardy inequality route of chapter 3435, the real-series summation surface of chapter 1581, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names Hardy-space boundary analysis as finite L10 rows; it does not import host holomorphic function spaces, almost-everywhere boundary representatives, measure-class quotients, Banach-space completeness, or nonconstructive maximal theorems.

Definition 5108.1 (Hardy space carrier)**.** *A* $\mathsf{HardySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (C, B, A, S, R, D, E, T, J, K, P, N)$$

with the following displayed rows:

1. *C is the coefficient row naming the finite analytic polynomial or power-series prefix under inspection.*
2. *B is the boundary-window row. It records the finitely many radial or angular boundary probes opened by the packet.*
3. *A is the* $\mathsf{HardyInequality}^{\uparrow}$ *control row used to compare displayed averages with the coefficient prefix.*
4. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *summation row attached to the inspected coefficient and boundary data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed boundary values and partial averages.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by every boundary-value comparison.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after C, B, A, S, R, D have been displayed.*
8. *T records componentwise* hsame *transport for C, B, A, S, R, D, E.*
9. *J records displayed* Cont *replay from a boundary probe through coefficient prefix, Hardy control, real-series summation, rational readback, dyadic tolerance, and real seal.*
10. *K is the* Pkg *provenance over* $\mathsf{HardySpace}^{\uparrow}$, $\mathsf{HardyInequality}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *P records the finite boundary predicate saying that each exported boundary read uses only the displayed prefix, window, tolerance, and readback rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HardySpace}^\uparrow}$ *row.*

The carrier contains no coordinate for a host Hardy space, an almost-everywhere boundary representative, a measure quotient, a selected nontangential limit, Banach completeness, or countable choice.

Lean4 ✓ `BEDC.Derived.HardySpaceUp.HardySpaceTasteGate single carrier alignment`

Theorem 5108.2 (Hardy space NameCert obligations)**.** *For every accepted* $\mathsf{HardySpace}^\uparrow$ *carrier* $H = (C, B, A, S, R, D, E, T, J, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HardySpace}^\uparrow}$ *surface consists exactly of carrier admission, coefficient-prefix exposure through C, finite boundary-window exposure through B, Hardy-control exposure through A, real-series exposure through S, regular rational readback through R, dyadic tolerance exposure through D, terminal real sealing through E, componentwise transport through T, displayed replay through J, package provenance through K, boundary-predicate exposure through P, and local naming through N.*

These obligations bind the finite Hardy-space route

$$\mathsf{HardyInequality}^\uparrow \longmapsto \mathsf{RealSeries}^\uparrow \longmapsto \mathsf{StreamName}^\uparrow \longmapsto \mathsf{RegSeqRat}^\uparrow \longmapsto \mathsf{DyadicRatCore}^\uparrow \longmapsto \mathsf{Real}^\uparrow$$

without adding a host analytic function space, quotient boundary class, selected limit, completeness theorem, or nonconstructive maximal principle.

Proof. Project the carrier of Definition 5108.1. The coefficient row and boundary-window row expose only finite analytic data and finitely many boundary probes.

The Hardy control and real-series rows are read next. Row A supplies the Hardy inequality packet used for finite average control, while S records the displayed real-series summation surface attached to the same prefix.

Readback and tolerance then pass through R and D. The regular rational row supplies the finite approximants for the boundary probes, and the dyadic ledger supplies the tolerance data used by each comparison.

Finally E, T, J, K, P, N seal, transport, replay, source, bound, and name the same route. Since no excluded analytic or quotient object appears among the carrier coordinates, the local NameCert surface exports exactly the listed finite obligations. □

Theorem 5108.3 (Hardy space real-boundary route)**.** *Let $H = (C, B, A, S, R, D, E, T, J, K, P, N)$ be an accepted* $\mathsf{HardySpace}^\uparrow$ *packet. Every Real-facing boundary-value consumer of H factors through a displayed coefficient prefix C, a finite boundary window B, the Hardy-control row A, the real-series row S, the regular rational readback R, and the dyadic tolerance ledger D before the* $\mathsf{Real}^\uparrow$ *seal E is available.*

Thus the packet records constructive boundary-value analysis as finite NameCert data over existing L10 real-name rows. It does not assert host holomorphy, almost-everywhere boundary equivalence, selected radial limits, quotient representative equality, or ambient completeness.

Proof. Start with the finite boundary window B. A boundary-value request is accepted only after it names the finite probes being inspected, so no consumer can read an unbounded boundary trace from the packet.

The coefficient prefix C, Hardy-control row A, and real-series row S provide the finite analytic data for those probes. Their order prevents a Real-facing consumer from replacing the displayed prefix by a host Hardy-space element.

Rows R and D then expose the rational readback and dyadic tolerance ledger. The terminal seal E is downstream from these rows, while T, J, K, P, N only transport, replay, source, bound, and name the same finite route.

Therefore every Real-facing boundary read factors through the displayed L10 rows, and the excluded analytic, quotient, choice, and completeness principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{HardySpace}^{\uparrow}$ read exposes a coefficient prefix, finite boundary window, HardyInequalityUp control row, RealSeriesUp summation row, RegSeqRatUp readback, DyadicRatCoreUp tolerance ledger, RealUp seal, transport, replay, provenance, boundary predicate, and local NameCert before any boundary-value claim is exported.

Independence witness: hardyinequality, realseries, streamname, regseqrat, dyadicratcore, real: $\mathsf{HardySpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries coefficient-prefix, finite boundary-window, Hardy-control, summation, readback, tolerance, and Real-seal rows for boundary-value analysis.

Closure status for $\mathsf{NameCert}_{\mathsf{HardySpace}^{\uparrow}}$.

Constructive story: A $\mathsf{HardySpace}^{\uparrow}$ packet is generated from a coefficient prefix, finite boundary window, Hardy control, RealSeries summation, RegSeqRat readback, DyadicRatCore tolerance, RealUp seal, componentwise transport, Cont replay, Pkg provenance, boundary predicate, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HardySpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5108.1, Definition 5108.2, and Definition 5108.3 seed the finite Hardy-space boundary route over displayed real-name rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HardySpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hardy-space theory, almost-everywhere boundary representatives, measure quotients, selected radial limits, Banach completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, boundary-window exposure, Hardy-control handoff, real-series handoff, readback and tolerance nonescape, Real seal handoff, and transport stability for BEDC.Derived.HardySpaceUp.

Chapter 5109

A Concrete Naming Certificate for $\mathsf{FiniteObservationReflection}^{\uparrow}$

The $\mathsf{FiniteObservationReflection}^{\uparrow}$ packet realizes the finite observation trace and open prediction horizon of section 9414.38 as one NameCert surface. It sits beside $\mathsf{ObservationReflectionPacket}^{\uparrow}$ of chapter 1126, the finite window interface of chapter 1007, the reflective inquiry carrier of chapter 1151, and the open reflection boundary of chapter 1161. Those siblings expose observation roles, windows, paired inquiry, or gap boundaries separately; this packet binds source, finite observation trace, signature, classifier, package, ledger, and open-continuation rows as one finite observation-reflection certificate.

Definition 5109.1 (Finite observation-reflection carrier)**.** *A* $\mathsf{FiniteObservationReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, O, G, C, P, L, E, T, R, N)$$

with the following displayed rows:

1. *S is the source row naming the reality-facing or formal-reflection source under inspection.*
2. *O is the finite observation-trace row, read through a displayed* $\mathsf{FiniteObservationInterface}^{\uparrow}$ *window.*
3. *G is the observed-signature row extracted from the finite trace.*
4. *C is the classifier row comparing the source claim with the observed signature.*
5. *P is the package row recording the model, law, or inquiry certificate that is being tested against the trace.*
6. *L is the ledger row for approximation budgets, rejected fits, and unresolved residue.*
7. *E is the open-continuation row preserving the prediction horizon beyond the displayed trace.*
8. *T records componentwise* hsame *transport for S, O, G, C, P, L, E.*
9. *R records displayed* Cont *replay from source and trace through signature, classifier, package, ledger, and open continuation.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationReflection}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a final physical oracle, a closed prediction horizon, source-domain identity, host validation, quotient trace equality, or an unbounded observation schedule.

Theorem 5109.2 (Finite observation-reflection NameCert obligations)**.** *For every accepted* $\mathsf{FiniteObservationReflec}$ *carrier* $F = (S, O, G, C, P, L, E, T, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteObservationReflection}^{\uparrow}}$ *surface consists exactly of carrier admission, source exposure through* S*, finite observation-trace exposure through* O*, observed-signature exposure through* G*, classifier exposure through* C*, package exposure through* P*, ledger exposure through* L*, open-continuation exposure through* E*, componentwise transport through* T*, displayed replay through* R*, and local naming through* N*.*

These obligations bind the observation-reflection route

$$\mathsf{ObservationReflectionPacket}^{\uparrow} \longmapsto \mathsf{FiniteObservationInterface}^{\uparrow} \longmapsto \mathsf{ReflectiveInquiry}^{\uparrow} \longmapsto \mathsf{ReflectionGapBoundary}^{\uparrow}$$

without adding final truth, closed prediction, source identity, host validation, quotient trace equality, or an unbounded schedule.

Proof. Project the carrier of Definition 5109.1. The source and observation rows S, O are displayed separately, so the packet can relate source claims to traces without identifying them.

The signature, classifier, and package rows G, C, P supply the finite comparison surface. They read the observed trace and the tested certificate as rows of the packet rather than as external host judgments.

The ledger and open-continuation rows then preserve non-finality. Row L records budgets, rejected fits, and unresolved residue, while E keeps the prediction horizon open beyond the finite trace.

Finally T, R, N transport, replay, and name exactly the same rows. Since no final oracle, closed horizon, source identity, quotient trace, or unbounded schedule is present, the local NameCert surface has precisely the listed obligations. □

Theorem 5109.3 (Finite observation-reflection open horizon)**.** *Let* $F = (S, O, G, C, P, L, E, T, R, N)$ *be an accepted* $\mathsf{FiniteObservationReflection}^{\uparrow}$ *packet. Every consumer that reads an observation-reflection judgment from* F *must pass through the source row* S*, the finite observation trace* O*, the signature row* G*, the classifier row* C*, the package row* P*, the ledger row* L*, and the open-continuation row* E*.*

In particular, the packet can certify a finite source–trace–signature fit while refusing to close the prediction horizon or identify the source with its observations.

Proof. Begin with the trace route. The source row S and finite observation row O are distinct coordinates, and the signature row G is extracted only from the displayed finite trace.

The classifier and package rows C, P then record the tested fit. This fit is ledgered by L, which keeps approximation budgets, rejected alternatives, and unresolved residue visible to downstream readers.

The open-continuation row E is read before the local name is exported. Componentwise transport and replay through T, R preserve that order, so a consumer cannot turn a finite observation-reflection packet into a final truth oracle or a closed prediction horizon. □

Falsifiable prediction: Every accepted $\mathsf{FiniteObservationReflection}^{\uparrow}$ read exposes source, finite observation trace, signature, classifier, package, ledger, open continuation, transport, replay, and local NameCert rows before exporting an observation-reflection judgment.

Independence witness: observationreflectionpacket, finiteobservationinterface, reflectiveinquiry, reflectiongapboundary: $\mathsf{FiniteObservationReflection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source, finite trace, signature, classifier, package, ledger, and open-continuation rows in one NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteObservationReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteObservationReflection}^{\uparrow}$ packet is generated from source, finite observation trace, observed signature, classifier, package, ledger, open continuation, componentwise transport, displayed replay, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteObservationReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5109.1, Definition 5109.2, and Definition 5109.3 seed the finite observation-reflection route and its open prediction horizon.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteObservationReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close final physical truth, final formal truth, source-domain identity, host validation, quotient trace equality, unbounded observation schedules, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source, observation-trace, signature, classifier, package, ledger, open-continuation, transport, and replay rows for BEDC.Derived.FiniteObservationReflectionUp.

Chapter 5110

A Concrete Naming Certificate for $\mathsf{MobiusTransform}^{\uparrow}$

Following Ahlfors on fractional linear transformations, the $\mathsf{MobiusTransform}^{\uparrow}$ packet records the finite coefficient and action rows by which a transformation $z \mapsto (Az + B)/(Cz + D)$ may be read before complex, projective, holomorphic, or hyperbolic cross-ratio consumers inspect it. It sits over the complex arithmetic carrier of chapter 39, the homogeneous-line action surface of chapter 207, the disk-local analytic surface of chapter 68, and the four-boundary comparison packet of chapter 3948. The packet names the fractional-linear carrier as BEDC data; it does not import a host Riemann sphere, quotient projective equality, selected inverse matrix, global automorphism group, or unrecorded cross-ratio theorem.

Definition 5110.1 (Mobius transform carrier)**.** *A* $\mathsf{MobiusTransform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, B, C, D, G, PJ, HD, XR, H, K, Q, N)$$

with the following displayed rows:

1. A, B, C, D *are* $\mathsf{Complex}^{\uparrow}$ *coefficient rows for the fractional-linear numerator and denominator.*
2. G *is the determinant guard row, recording that the displayed complex determinant* $AD - BC$ *is apart from zero in the accepted complex-history reading.*
3. PJ *is the* $\mathsf{ProjectiveSpace}^{\uparrow}$ *action row, exposing the homogeneous coordinate transformation* $[z : w] \mapsto [Az + Bw : Cz + Dw]$ *only through the displayed projective carrier.*
4. HD *is the* $\mathsf{Holomorphic}^{\uparrow}$ *disk-action row, recording the local disk source, denominator-apartness ledger, and holomorphic readback for every admitted disk on which the denominator guard is available.*
5. XR *is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *preservation row, requiring any four-boundary comparison to enter through the disk row, metric row, four boundary rows, boundary-fixed transport row, cross-ratio row, refusal row, Schwarz–Pick provenance, and structural rows of chapter 3948.*
6. H *records componentwise* hsame *transport for the four coefficient rows, determinant guard, projective action, holomorphic disk action, cross-ratio preservation, package provenance, classifier replay, and local naming.*
7. K *records displayed* Cont *replay from coefficient reads through determinant guard, projective action, holomorphic disk action, and cross-ratio preservation.*

8. *Q is the* Pkg *provenance over* MobiusTransform$^{\uparrow}$, Complex$^{\uparrow}$, ProjectiveSpace$^{\uparrow}$, Holomorphic$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MobiusTransform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, B, C, D, G, PJ, HD, XR, H, K, Q, N$. It has no coordinate for host matrix equality, quotient projective equality, a selected inverse, an ambient automorphism group, or a cross-ratio statement not routed through the displayed four-boundary packet.

Lean4 ✓ `BEDC.Derived.MobiusTransformUp.TasteGate.MobiusTransformTasteGate single carrier alignment`

Theorem 5110.2 (Mobius transform NameCert obligations)**.** *For every accepted* MobiusTransform$^{\uparrow}$ *carrier $M = (A, B, C, D, G, PJ, HD, XR, H, K, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MobiusTransform}^{\uparrow}}$ *surface consists exactly of complex coefficient rows, the determinant guard, projective action, holomorphic disk action, hyperbolic cross-ratio preservation routed through* HyperbolicCrossRatio$^{\uparrow}$, *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted consumer must inspect those rows before treating the packet as a fractional-linear transformation.*

Proof. Project the carrier of Definition 5110.1. The four coefficient rows A, B, C, D are the only complex entries of the packet, and the determinant guard G is the only accepted witness that the displayed matrix is usable.

The action rows PJ and HD then expose the same coefficient data through two controlled surfaces. The projective action is read through ProjectiveSpace$^{\uparrow}$, while the holomorphic disk action is read through Holomorphic$^{\uparrow}$ with the denominator-apartness ledger in scope.

The cross-ratio row XR is not a free theorem about a host sphere. It is admitted only as a consumer of the HyperbolicCrossRatio$^{\uparrow}$ four-boundary packet, so preservation claims inherit the disk, metric, boundary, refusal, Schwarz–Pick, transport, replay, provenance, and local naming rows of that packet.

Finally H, K, Q, N transport, replay, source, and name the same finite carrier. Since no other row is present, a consumer cannot replace the determinant guard by host invertibility, replace projective action by quotient equality, or bypass the displayed cross-ratio route. □

Theorem 5110.3 (Mobius transform cross-ratio preservation route)**.** *Every accepted* MobiusTransform$^{\uparrow}$ *cross-ratio consumer factors through the finite route*

$$\mathsf{Complex}^{\uparrow},\ \mathsf{ProjectiveSpace}^{\uparrow},\ \mathsf{Holomorphic}^{\uparrow},\ \mathsf{HyperbolicCrossRatio}^{\uparrow}.$$

The route first reads the complex coefficient rows and determinant guard, then the projective action row, then the holomorphic disk-action row, and finally the HyperbolicCrossRatio$^{\uparrow}$ *preservation row with its disk, metric, four boundary rows, boundary-fixed transport, cross-ratio comparison, refusal, Schwarz–Pick provenance, transport, replay, package provenance, and local naming. It exports no host Riemann sphere, quotient projective equality, selected inverse matrix, global automorphism group, or unrecorded boundary invariant.*

Proof. Let $M = (A, B, C, D, G, PJ, HD, XR, H, K, Q, N)$ be accepted. The coefficient rows and determinant guard are read first, so the fractional-linear numerator, denominator, and apartness condition are fixed as finite Complex$^{\uparrow}$ histories.

The projective row PJ packages the same coefficients as a homogeneous action through ProjectiveSpace$^{\uparrow}$. The holomorphic row HD packages the same coefficients as a disk-local map through Holomorphic$^{\uparrow}$, with denominator-apartness retained before any analytic readback is exposed.

The row XR is the only coordinate where cross-ratio preservation can be claimed. By construction it points to the $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ four-boundary packet, so it must read the disk, metric, boundary, transport, comparison, refusal, Schwarz–Pick, and structural rows before preservation is visible.

The structural rows H, K, Q, N keep the same carrier under transport, replay, provenance, and naming. Thus the preservation route remains finite and local to the displayed NameCert data. □

Falsifiable prediction: Any accepted $\mathsf{MobiusTransform}^{\uparrow}$ consumer omitting one of the four $\mathsf{Complex}^{\uparrow}$ coefficient rows, the determinant guard, the $\mathsf{ProjectiveSpace}^{\uparrow}$ action row, the $\mathsf{Holomorphic}^{\uparrow}$ disk-action row, the $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ preservation row, Pkg provenance, or the local $\mathsf{NameCert}$ row refutes Definition 5110.2.

Independence witness: complex, projectivespace, holomorphic, hyperbolic-cross-ratio: $\mathsf{MobiusTransform}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries coefficient rows, determinant guard, projective action, holomorphic disk action, cross-ratio preservation, transport, replay, package provenance, and local naming for a fractional-linear transformation.

Closure status for $\mathsf{NameCert}_{\mathsf{MobiusTransform}^{\uparrow}}$.

Constructive story: A $\mathsf{MobiusTransform}^{\uparrow}$ packet is generated from four $\mathsf{Complex}^{\uparrow}$ coefficient rows, a determinant guard, a $\mathsf{ProjectiveSpace}^{\uparrow}$ homogeneous action row, a $\mathsf{Holomorphic}^{\uparrow}$ disk-action row, a $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ preservation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MobiusTransform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5110.1, Definition 5110.2, and Definition 5110.3 seed the finite fractional-linear carrier over complex coefficients, determinant guard, projective action, holomorphic disk action, cross-ratio preservation, and structural rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MobiusTransform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Riemann-sphere equality, quotient projective equality, selected inverse matrices, global automorphism groups, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, determinant guard transport, projective action replay, holomorphic disk-action replay, cross-ratio consumer routing, TasteGate nontriviality, field faithfulness, provenance, transport, replay, and local naming for BEDC.Derived.MobiusTransformUp.

Chapter 5111

A Concrete Naming Certificate for $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$

Following Heine and Borel's interval compactness theorem as read through Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ packet records a finite dyadic cover-refinement ledger over a closed bounded $\mathsf{Real}^{\uparrow}$ interval. It sits over the closed bounded interval source of chapter 1591, the dyadic arithmetic core of chapter 291, the terminal real-name seal of chapter 38, the compact-metric carrier of chapter 129, and the uniform-modulus source of chapter 369. The chapter names finite dyadic cover and refinement data for the active L10 Real and Completion frontier; it is not a general open-cover compactness theorem, an unrestricted Heine–Borel principle, a selected-subcover functional, or an ambient $\mathsf{Real}^{\uparrow}$ completeness assertion.

Definition 5111.1 (Constructive Heine–Borel interval carrier)**.** *A* $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, R, S, E, M, U, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, exposing endpoint order, interval membership, and the interval-source readback.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for dyadic endpoints, mesh scales, refinement inclusions, and positive slack comparisons.*
3. *R is the finite refinement row, listing the dyadic partition cells and the parent-child refinement map used by cover readers.*
4. *S is the finite subcover ledger, recording the selected cell witnesses already displayed by the cover-refinement row.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint and radius seal reached only after I, D, R, S have been displayed.*
6. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *compatibility row, read only as the metric finite-net surface induced by the same interval and dyadic ledger.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *compatibility row, recording the modulus-facing refinement scale consumed by uniform-continuity readers over the same packet.*
8. *H records componentwise* hsame *transport for the interval source, dyadic ledger, refinement row, subcover ledger, Real seal, compact-metric row, uniform-modulus row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dyadic cover request through* I, D, R, S, E, M, U.
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, R, S, E, M, U, H, C, P, N$. It has no coordinate for arbitrary open covers, choice-selected subcovers, quotient interval representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveHeineBorelIntervalUp.ConstructiveHeineBorelIntervalTasteGate`
`single carrier alignment`

Theorem 5111.2 (Constructive Heine–Borel interval NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ *carrier* $K = (I, D, R, S, E, M, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorelIn}}$ *surface consists exactly of five finite obligations:*

1. *carrier admission for the closed bounded interval source, dyadic endpoint ledger, finite refinement row, finite subcover ledger, Real seal, compact-metric compatibility row, uniform-modulus compatibility row, transport, replay, provenance, and local naming rows;*
2. *cover-refinement classifier stability, requiring two accepted reads to agree only through I, D, R, H, C, P, N before their refinement cells are compared;*
3. *finite-subcover ledger exactness, requiring each exported witness in S to be one of the displayed finite cover reads attached to a dyadic cell of R;*
4. *dyadic endpoint transport, requiring endpoint and mesh comparisons to pass through D before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*
5. *nonescape from selected subcovers, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness, requiring M, U, H, C, P, N to transport, replay, source, and name only the displayed finite packet.*

Proof. Carrier admission is projection from Definition 5111.1. The tuple displays exactly the closed bounded interval source, dyadic endpoint ledger, finite refinement row, finite subcover ledger, Real seal, compact-metric compatibility row, uniform-modulus compatibility row, and structural rows.

Classifier stability is local to the refinement ledger. Two accepted reads can compare cells only after the interval row I, dyadic row D, and refinement row R have been displayed and transported through H, C, P, N.

The finite-subcover ledger is exact because S contains only the witnesses attached to the listed cells of R. There is no row carrying an arbitrary cover family or a host selector for additional witnesses.

Endpoint transport is dyadic before it is Real-facing. The row D records endpoint, mesh, refinement, and slack comparisons, and E seals only those endpoint and radius reads already attached to I, D, R, S.

Finally M and U are compatibility rows over the same finite data, while H, C, P, N preserve and name the displayed route. Since selected subcovers, countable choice, and ambient real completeness have no carrier coordinate, the five obligations exhaust the local NameCert surface. □

Theorem 5111.3 (Constructive Heine–Borel interval finite-subcover ledger)**.** *Every compactness-facing or completion-facing consumer of an accepted* $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through $I, D, R, S, E, M, U, H, C, P, N$. The exported subcover row is exactly the finite ledger S transported from the dyadic refinement row R to the terminal endpoint and radius seal E; it is not general open-cover compactness, a selected subcover functional, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the obligation surface of Definition 5111.2. Every cover-facing read must display the closed bounded interval source, the dyadic endpoint ledger, the refinement row, and the finite subcover ledger.

The route then passes through the compatibility rows. The compact-metric row M reads the same finite dyadic cells as a finite-net surface, and the uniform-modulus row U reads the same refinement scale for uniform-continuity consumers.

The Real-facing endpoint seal E is reached only after those finite rows have been displayed. Transport through H, replay through C, provenance through P, and local naming through N preserve that order.

Thus the exported row is the finite ledger already recorded in S. The refused open-cover, selected-subcover, countable-choice, and ambient-completeness readings have no row in the packet. □

Falsifiable prediction: A consumer that reads $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ as a general open-cover compactness theorem, or reaches a subcover without displaying the closed bounded interval source, DyadicRatCore ledger, finite refinement row, finite subcover ledger, Real seal, compact-metric compatibility row, uniform-modulus compatibility row, transport, replay, provenance, and local NameCert rows, refutes the carrier surface.

Independence witness: closedboundedinterval, dyadicratcore, real, compactmetric, uniform-modulus: no listed sibling is bijective to $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ because this packet binds a finite dyadic cover-refinement ledger, exact finite-subcover row, endpoint transport, compact-metric compatibility, uniform-modulus compatibility, Real sealing, transport, replay, provenance, and local naming in one L10 interval-compactness interface.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a DyadicRatCore endpoint and refinement ledger, a finite subcover ledger, a RealUp seal, CompactMetric and UniformModulus compatibility rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5111.1, Definition 5111.2, and Definition 5111.3 seed the finite dyadic-cover interval compactness packet over closed bounded interval, dyadic endpoint, compact-metric, uniform-modulus, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveHeineBorelInterval}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ConstructiveHeineBorelIntervalUp.ConstructiveHeineBorelIntervalCarrier`

Bridge status: none

Not claimed: This chapter does not close general open-cover compactness, unrestricted Heine–Borel compactness, selected subcover functionals, quotient interval representatives, countable choice, ambient RealUp completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-refinement classifier stability, finite-subcover ledger exactness, dyadic endpoint transport, compact-metric compatibility, uniform-modulus compatibility, nonescape, replay, provenance, and local naming for BEDC.Derived.ConstructiveHeineBorelIntervalUp.

Chapter 5112

A Concrete Naming Certificate for $\mathsf{QuadraticForm}^{\uparrow}$

Following Bourbaki and Serre on quadratic forms, the $\mathsf{QuadraticForm}^{\uparrow}$ packet records a scalar-valued quadratic law on a displayed module or vector-space carrier as finite BEDC NameCert data. It sits between the scalar field of chapter 45, the module and vector-space source rows of chapter 46 and chapter 47, the bilinear-form pairing surface of chapter 146, and the real-facing scalar seal of chapter 38. Downstream consumers such as chapter 3367, chapter 147, and chapter 310 may read the same finite quadratic packet instead of redeclaring diagonal scalar data.

Definition 5112.1 (Quadratic form carrier)**.** *A* $\mathsf{QuadraticForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (F, V, B, q, H, P, T, C, S, N)$$

with the following displayed rows:

1. *F is the scalar* $\mathsf{Field}^{\uparrow}$ *source row, with the* $\mathsf{Real}^{\uparrow}$ *seal recorded when the scalar endpoint is real-valued.*
2. *V is the* $\mathsf{Module}^{\uparrow}$ *or* $\mathsf{VecSpace}^{\uparrow}$ *carrier row whose displayed addition, zero, inverse, and scalar-action rows are the only vector endpoints admitted by the packet.*
3. *B is the* $\mathsf{BilinForm}^{\uparrow}$ *source row that supplies the visible polarization candidate used by quadratic consumers.*
4. *q is the scalar endpoint row assigning each accepted vector endpoint a carried scalar value in the source named by F.*
5. *H records the homogeneity ledger $q(a \cdot x) = a^2 q(x)$ over displayed scalar-action rows and scalar-square rows.*
6. *P records the polarization handoff row, reading the cross-term endpoint through the bilinear source B and the displayed scalar expression determined by $q(x+y)$, $q(x)$, and $q(y)$.*
7. *T records componentwise* hsame *transport for scalar field data, module or vector-space data, bilinear-form source rows, quadratic endpoint rows, homogeneity rows, polarization rows, continuation, provenance, and local naming.*
8. *C records displayed* Cont *replay from the source carrier through the quadratic endpoint, homogeneity ledger, polarization handoff, and local certificate.*

9. *S is the* Pkg *provenance over* $\mathsf{QuadraticForm}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{BilinForm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{QuadraticForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, B, q, H, P, T, C, S, N$. It has no coordinate for a host function on an ambient vector space, quotient equality of quadratic maps, a selected basis, diagonalization, inertia, positivity, Clifford quotient data, or bridge checking.

Lean4 ✓ `BEDC.Derived.QuadraticFormUp.QuadraticFormTasteGate single carrier alignment`

Theorem 5112.2 (Quadratic form polarization boundary)**.** *For every accepted* $\mathsf{QuadraticForm}^{\uparrow}$ *carrier $Q = (F, V, B, q, H, P, T, C, S, N)$, the polarization row P is the only public route from scalar quadratic values to a bilinear-form read. A consumer may inspect the displayed cross-term package through the* $\mathsf{BilinForm}^{\uparrow}$ *source row B and the scalar endpoints named by q, but it receives no host bilinear map, no quotient equality of forms, no selected coordinate matrix, and no diagonal basis.*

Proof. Project the carrier of Definition 5112.1. The only row that mentions both the quadratic endpoint q and a bilinear output is the polarization handoff P.

The handoff first reads the admitted vector endpoints through V and their scalar outputs through q. It then compares the cross-term endpoint with the bilinear source B inside the same scalar field row F, so the comparison is a displayed BEDC package rather than an external equality of functions.

The excluded objects are absent from the carrier. Since no row stores a host bilinear map, quotient equality of forms, selected coordinates, or a diagonal basis, the polarization read cannot export them. □

Theorem 5112.3 (Quadratic form NameCert obligations)**.** *For every accepted* $\mathsf{QuadraticForm}^{\uparrow}$ *carrier $Q = (F, V, B, q, H, P, T, C, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{QuadraticForm}^{\uparrow}}$ *surface consists of carrier admission for the scalar field, module or vector-space source, bilinear-form source, quadratic endpoint, homogeneity ledger, polarization handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The exactness boundary says that every public quadratic-form read is generated by these rows and that every downstream consumer must factor through the displayed source certificates before reading scalar quadratic data.*

Proof. Read the carrier rows in the displayed order. The scalar field row F, vector or module carrier row V, bilinear source B, and scalar endpoint row q determine the admitted quadratic surface.

The homogeneity obligation is the ledger H. It uses only the scalar-action rows from V and the scalar-square rows from F, so it remains inside the displayed module or vector-space source.

The polarization obligation is the row P and is bounded by Definition 5112.2. Finally T, C, S, N supply transport, replay, provenance, and local naming. These projections exhaust the local NameCert surface, so no hidden positivity, inertia, diagonalization, Clifford quotient, host function equality, or bridge row is available. □

Theorem 5112.4 (Quadratic form bilinear-form handoff)**.** *Every accepted* $\mathsf{QuadraticForm}^{\uparrow}$ *packet exposes its bilinear-form interface through the route*

$$\mathsf{BilinForm}^{\uparrow}, \mathsf{Module}^{\uparrow}, \mathsf{VecSpace}^{\uparrow}, \mathsf{Field}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The route sends diagonal reads to the scalar endpoint q and cross-term reads to the polarization row P. Consequently $\mathsf{SylvesterInertia}^{\uparrow}$, $\mathsf{Clifford}^{\uparrow}$, *and* $\mathsf{SemidefiniteCone}^{\uparrow}$ *consumers may cite the same*

quadratic packet when they need scalar diagonal data, bilinear polarization data, and source-carrier transport.

Proof. Start from a public consumer read. By Definition 5112.1, diagonal scalar data can be read only from q, and cross-term data can be read only from P together with the bilinear source B.

The route first fixes the scalar source F and the vector or module source V. It then reads B for the bilinear-form side and reads q, H, P for the quadratic endpoint, homogeneity, and polarization rows.

Apply the NameCert obligation surface of Definition 5112.3. The consumer receives exactly the finite rows needed by the inertia, Clifford, and semidefinite-cone chapters: scalar diagonal values, bilinear handoff, carrier transport, continuation replay, provenance, and local naming. Since selected bases, diagonalization, positivity decisions, Clifford quotients, and optimization cones are not carrier coordinates, they remain obligations of the downstream chapters rather than hidden exports of $\mathsf{QuadraticForm}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuadraticForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuadraticForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5112.1, Definition 5112.2, Definition 5112.3, and Definition 5112.4 seed the finite scalar quadratic-form packet over displayed field, module, vector-space, bilinear-form, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{QuadraticForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host function equality of quadratic maps, selected bases, diagonalization, inertia, positivity, Clifford quotient data, semidefinite cone construction, or optimization duality.

Upgrade path: Obligation closure requires checked carrier admission, homogeneity, polarization boundary, bilinear-form handoff, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.QuadraticFormUp.

Constructive story: A $\mathsf{QuadraticForm}^{\uparrow}$ packet is generated from a FieldUp scalar row, ModuleUp or VecSpaceUp carrier rows, a BilinFormUp source row, a scalar quadratic endpoint, homogeneity and polarization ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5113

A Concrete Naming Certificate for $\mathsf{RealStepFunction}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive analysis, the $\mathsf{RealStepFunction}^{\uparrow}$ packet records finite step functions over located real intervals as BEDC NameCert data. It refines the finite partition-to-value surface of chapter 3780 along the tagged real interval route and feeds the Darboux/Riemann consumer boundary of chapter 1960. The packet supplies the RealUp-specific step-function interface used by integration chapters without importing host functions, quotient equality of integrands, selected refinement limits, measure completion, or ambient real completeness.

Definition 5113.1 (Real step function carrier)**.** *A* $\mathsf{RealStepFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, T, V, S, R, G, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-domain row, carrying the located real endpoints and interval-side admission data on which the function is read.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *cell row, listing finitely many ordered tagged real subintervals, endpoint incidence, tag containment, and coverage over the row I.*
3. *V is the* $\mathsf{StepFunction}^{\uparrow}$ *value row, assigning one displayed value row to each tagged cell of T and exposing constant-on-cell admission only through that finite cell list.*
4. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *readback row computed from the tagged cells, cell widths, tags, and displayed values.*
5. *R is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, exposing the Darboux/Riemann consumer route that may use I, T, V, S as finite integrability evidence.*
6. *G is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *comparison row, recording that the same finite cell-value surface can be read as a regulated-function approximation boundary when a downstream chapter asks for that interface.*
7. *H records componentwise* hsame *transport for the interval-domain row, tagged partition, step-value row, Riemann-sum readback, integrability handoff, regulated comparison, continuation replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from I to T, from T, V to S, from I, T, V, S to R, and from the same finite rows to G.*

9. *P is the* Pkg *provenance over* $\mathsf{RealStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{StepFunction}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealStepFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host-level function, quotient equality of integrands, a selected refining partition sequence, a measure-space completion, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5113.2 (Real step function NameCert obligations)**.** *For every accepted* $\mathsf{RealStepFunction}^{\uparrow}$ *carrier $E = (I, T, V, S, R, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealStepFunction}^{\uparrow}}$ *surface consists exactly of real interval-domain admission, tagged real partition exposure, finite step-value exposure, Riemann-sum readback, Riemann-integrable handoff, regulated-function comparison, component-wise transport, displayed replay, provenance, local naming, and non-escape from host functions, quotient integrand equality, selected refinement limits, measure completion, open-cover compactness, and ambient real completeness.*

Lean4 ✓ `BEDC.Derived.RealStepFunctionUp.RealStepFunctionNamecertObligations`

Proof. Project the carrier of Definition 5113.1. The interval-domain row I, tagged partition row T, and value row V are the only rows from which a consumer can read the domain cells and their constant values.

The integration-facing rows are downstream of that finite data. The row S is computed from the tagged cells, widths, tags, and displayed values, while R admits a $\mathsf{RiemannIntegrable}^{\uparrow}$ consumer only after I, T, V, S have been displayed.

The regulated-function comparison uses the same rows. The row G does not introduce another function object; it records only the regulated approximation boundary supplied by the finite cell-value surface.

Finally H, C, P, N preserve, replay, source, and name exactly this packet. Since the excluded host objects are not carrier coordinates, the listed obligations exhaust the local NameCert surface. □

Theorem 5113.3 (Real step function Riemann readback)**.** *Every Riemann-facing read of an accepted* $\mathsf{RealStepFunction}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{TaggedRealPartition}^{\uparrow} \longmapsto \mathsf{StepFunction}^{\uparrow} \longmapsto \mathsf{RiemannSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow}$$

together with the $\mathsf{RegulatedFunction}^{\uparrow}$ *comparison row and the structural rows H, C, P, N. The route exports finite tagged real interval data and displayed constant values, not a host integrand, quotient function class, selected mesh limit, measure-theoretic completion, or ambient real-completeness principle.*

Proof. Start with the real interval-domain row I. A Riemann-facing consumer cannot observe a cell or tag before the tagged real partition row T displays the finite subdivision and containment data over that interval.

Next read the step-value row V. Each value is attached to a displayed cell, so the finite sum row S is computed from visible tags, cell widths, and displayed constant values rather than from a host function.

The integrability handoff is the row R. It can be consumed only after the finite Riemann-sum readback has been displayed, and the regulated comparison row G is tied to the same cell-value surface.

The structural rows H, C, P, N transport, replay, source, and name this route without adding another analytic coordinate. Therefore the readback remains inside the displayed RealUp interval, tagged partition, step-function, sum, integrability, and regulated-function rows. □

Falsifiable prediction: An accepted $\mathsf{RealStepFunction}^{\uparrow}$ read is refuted if its $\mathsf{RiemannIntegrable}^{\uparrow}$ handoff remains while the displayed $\mathsf{Real}^{\uparrow}$ interval row, $\mathsf{TaggedRealPartition}^{\uparrow}$ cells, $\mathsf{StepFunction}^{\uparrow}$ value rows, and $\mathsf{RiemannSum}^{\uparrow}$ readback are removed.

Independence witness: real-interval, tagged-partition, step-values, riemann-sum, riemann-integrable, regulated-function: none is bijective with $\mathsf{RealStepFunction}^{\uparrow}$ because the packet jointly carries interval-domain data, tagged real cells, displayed constant values, finite Riemann-sum readback, integrability handoff, regulated comparison, componentwise transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RealStepFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{RealStepFunction}^{\uparrow}$ packet is generated from a RealUp interval-domain row, TaggedRealPartitionUp cells, StepFunctionUp values, RiemannSumUp readback, RiemannIntegrableUp handoff, RegulatedFunctionUp comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealStepFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5113.1, Definition 5113.2, and Definition 5113.3 seed the finite RealUp interval step-function surface over tagged real partitions, step-value rows, Riemann sums, Riemann integrability, and regulated-function comparison.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealStepFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient integrands, selected refinement limits, measure completion, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged partition exposure, step-value exposure, Riemann-sum readback, Riemann-integrable handoff, regulated comparison, and structural transport for BEDC.Derived.RealStepFunctionUp.

Chapter 5114

A Concrete Naming Certificate for $\mathsf{RegularCauchyNonzero}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyNonzero}^{\uparrow}$ packet names the finite nonzero-domain evidence needed between regular Cauchy rational sources and guarded reciprocal or division consumers. It binds the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ inspection window of chapter 126, the dyadic lower-bound ledger of chapter 291, the reciprocal consumer of chapter 1471, the division consumer of chapter 1943, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter records only displayed finite apartness from zero for a regular Cauchy name; it is not real trichotomy and not a total reciprocal operation.

Definition 5114.1 (Regular Cauchy nonzero carrier)**.** *A* $\mathsf{RegularCauchyNonzero}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (Q, A, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational tail is being inspected.*
2. *A is the apartness witness row displaying that the inspected tail is separated from zero by a positive rational bound.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened from Q at the precision and tail indices required by A.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *lower-bound ledger recording the dyadic comparison, radius budget, and positive lower endpoint read from W.*
5. *R is the reciprocal-domain handoff row offered to* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *and* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *consumers.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal receiving the same finite nonzero-domain packet without adding trichotomy or selected representatives.*
7. *H records componentwise* hsame *transport for $Q, A, W, D, R, E, C, P, N$.*
8. *C records displayed* Cont *replay for source inspection, apartness exposure, finite-window opening, dyadic lower-bound reading, reciprocal-domain handoff, and Real seal handoff.*
9. *P records* Pkg *provenance over* $\mathsf{RegularCauchyNonzero}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyDivision}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyNonzero}^{\uparrow}}$ *row.*
The local classifier compares accepted carriers only through the displayed rows $Q, A, W, D, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.RegularCauchyNonzeroUp.RegularCauchyNonzeroTasteGate single carrier alignment`

Theorem 5114.2 (Regular Cauchy nonzero NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyNonzero}^{\uparrow}}$ *surface for Definition 5114.1 consists exactly of regular Cauchy source exposure, apartness-witness exposure, finite StreamName window exposure, DyadicRatCore lower-bound ledger exposure, reciprocal-domain handoff, terminal Real seal handoff, componentwise transport, displayed continuation replay, package provenance, and the local naming row.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNonzeroUp.RegularCauchyNonzeroNamecertObligations`

Proof. Project the carrier $Z = (Q, A, W, D, R, E, H, C, P, N)$. The coordinates list the regular Cauchy source, apartness witness, finite window, dyadic lower-bound ledger, reciprocal-domain handoff, Real seal, transport, replay, provenance, and local name.

The analytic rows are ordered by dependency. The window row W is opened from Q under the apartness witness A, while the dyadic ledger D records the positive lower-bound computation read from that finite window.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since every listed obligation is a coordinate or a projection from a coordinate, the seed NameCert surface is exhausted by the displayed packet. □

Theorem 5114.3 (Regular Cauchy nonzero apartness window)**.** *For every accepted* $\mathsf{RegularCauchyNonzero}^{\uparrow}$ *carrier $Z = (Q, A, W, D, R, E, H, C, P, N)$, every nonzero-domain read factors through*

$$Q \longmapsto A \longmapsto W \longmapsto D.$$

A route that opens a reciprocal or division consumer without the displayed apartness witness A and lower-bound ledger D remains outside the local classifier.

Proof. Start from the source row Q and apartness witness A. The witness is the only row asserting a positive separation from zero for the inspected regular Cauchy tail.

Next open the finite window W. Its entries are the only regular rational data inspected by the certificate, and the dyadic ledger D is a finite lower-bound read from that window.

The local classifier admits no row that bypasses A, W, D. Therefore every accepted nonzero-domain read reaches the lower-bound ledger through the displayed source, witness, and window route. □

Theorem 5114.4 (Regular Cauchy nonzero reciprocal-domain handoff)**.** *Every* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$ *or* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyNonzero}^{\uparrow}$ *packet reaches its denominator-domain row through the same handoff*

$$(Q, A, W, D) \longmapsto R.$$

The handoff exports finite apartness evidence and a dyadic lower-bound ledger, but it exports no total reciprocal, no quotient stream equality, no real trichotomy, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Use the apartness-window theorem to fix the source, witness, window, and lower-bound ledger rows. These rows are the complete analytic source of the nonzero-domain claim.

The handoff row R is downstream of D. A reciprocal consumer reads R as the denominator-domain guard for chapter 1471, and a division consumer reads the same row before the reciprocal-product route of chapter 1943.

Finally read the Real seal E and the structural rows H, C, P, N. None of these rows adds total inversion, quotient equality, trichotomy, or completeness data, so the handoff exports only the displayed finite nonzero-domain certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyNonzero}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyNonzero}^{\uparrow}$ packet is generated from a $\mathsf{RegSeqRat}^{\uparrow}$ source row, an apartness witness, a finite $\mathsf{StreamName}^{\uparrow}$ window, a $\mathsf{DyadicRatCore}^{\uparrow}$ lower-bound ledger, a reciprocal-domain handoff, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyNonzero}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5114.1, Definition 5114.2, Definition 5114.3, and Definition 5114.4 seed the finite nonzero-domain boundary for regular Cauchy names over $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyDivision}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyNonzero}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim total reciprocal construction, total division, real trichotomy, quotient stream equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-window routing, reciprocal-domain handoff, and the corresponding finite nonzero-domain NameCert rows.

Chapter 5115

A Concrete Naming Certificate for $\mathsf{RealUniformEntourageBasis}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and Bishop–Bridges constructive real analysis, the $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ packet records the metric-ball basis that realizes the Real-induced uniformity route of chapter 9237. It sits after the Real endpoint and seal surface of chapter 38, the metric witness of chapter 57, the entourage discipline of chapter 339, the Real-induced uniformity of chapter 1624, and the Cauchy-filter consumer of chapter 333. Its finite readback path uses the StreamName surface of chapter 126 and the RegSeqRat readback of chapter 301.

Definition 5115.1 (Real uniform entourage-basis carrier)**.** *A* $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, M, U, S, F, W, Q, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and seal row naming the points whose metric balls generate the basis.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance witness row read before any entourage request is generated.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row generated by positive metric-ball requests.*
4. *S is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *source row that binds the metric-ball basis to the Real-induced uniformity.*
5. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer row that reads Cauchy data through the generated basis.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by completion-facing consumers.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
8. *H records componentwise* hsame *transport for the Real, metric, uniform, Real-uniform, Cauchy-filter, window, readback, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$R, M \longmapsto U, S \longmapsto F, W, Q.$$

10. *P is the* Pkg *provenance over* $\mathsf{RealUniformEntourageBasis}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, M, U, S, F, W, Q, H, C, P, N$. It has no coordinate for quotient real equality, selected limits, countable choice, open-cover compactness, quotient filters, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealUniformEntourageBasisUp.RealUniformEntourageBasisUpTasteGate single carrier alignment`

Theorem 5115.2 (Real uniform entourage-basis NameCert obligations)**.** *For every accepted* RealUniformEntourage *carrier*

$$E = (R, M, U, S, F, W, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBasis}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the eleven displayed rows;*
2. *Real endpoint exposure, requiring every basis request to read R before it reads a metric ball;*
3. *metric-ball generation, requiring M to generate each entourage row in U by a displayed positive-radius request;*
4. *Real-uniform source compatibility, requiring S to identify the basis as the* $\mathsf{RealUniformStructure}^{\uparrow}$ *metric-ball surface;*
5. *Cauchy-filter handoff, requiring F to consume the generated basis before any completion-facing read;*
6. *finite readback, requiring W and Q to expose only StreamName windows and RegSeqRat readbacks downstream of the same basis route;*
7. *componentwise transport, displayed replay, provenance, and local naming for the route through H, C, P, N.*

No Real uniform entourage-basis obligation is stored outside these rows.

Proof. Project the accepted packet $E = (R, M, U, S, F, W, Q, H, C, P, N)$. The projection exposes exactly the Real endpoint row, metric witness row, generated uniform entourage row, Real-uniform source row, Cauchy-filter consumer, finite readback rows, and structural rows.

The basis starts from Real endpoints and metric data. The row R names the endpoints, and M supplies the distance witness from which the metric-ball entourage row U is generated.

The row S binds this generated basis to the $\mathsf{RealUniformStructure}^{\uparrow}$ route. A Cauchy-filter consumer can then read F only after the same basis has been displayed.

The rows W and Q give finite StreamName and RegSeqRat readback downstream of the basis. The structural rows H, C, P, N transport, replay, source, and name the same route, and the excluded objects do not occur as carrier coordinates. □

Theorem 5115.3 (Real uniform entourage-basis Cauchy-filter handoff)**.** *Every completion-facing or Cauchy-filter consumer of an accepted* $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ *carrier reads the Real-induced metric-ball basis only through*

$$\mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{UniformSpace}^{\uparrow}, \mathsf{RealUniformStructure}^{\uparrow}, \mathsf{CauchyFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Nam}$$

Consequently the packet exports no quotient real equality, selected limit, countable-choice selector, open-cover compactness, quotient filter, or ambient Real completeness.

Proof. Begin with a completion-facing or Cauchy-filter read. Carrier admission gives access only to $R, M, U, S, F, W, Q, H, C, P, N$.

Use the metric-uniformity handoff of Definition 1624.3. It forces the Real-induced uniformity to pass through endpoint rows, metric distance evidence, generated entourage rows, Cauchy-filter compatibility, finite windows, and regular rational readback.

The present carrier isolates the basis part of that route. The rows R, M, U, S generate and identify the metric-ball basis, F consumes it as a Cauchy-filter surface, and W, Q provide the finite readback needed by completion-facing consumers.

Since quotient equality, selected limits, choice, quotient filters, open-cover compactness, and ambient Real completeness are not rows of E, the consumer obtains only the displayed NameCert route. □

Falsifiable prediction: An accepted $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ read is refuted if a Cauchy-filter or completion-facing consumer exports a Real-induced basis without displaying Real endpoint rows, Metric distance witnesses, UniformSpace entourage rows, RealUniformStructure source compatibility, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real-uniform-structure, real, metric, uniformspace, cauchyfilter, streamname, regseqrat: no listed sibling alone is bijective with $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ because the packet jointly carries metric-ball basis generation, Real-uniform source compatibility, Cauchy-filter handoff, finite readback, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformEntourageBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformEntourageBasis}^{\uparrow}$ packet is generated from RealUp endpoint rows, MetricUp distance witnesses, UniformSpace entourage rows, RealUniformStructureUp source compatibility, CauchyFilterUp handoff, StreamNameUp windows, RegSeqRatUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformEntourageBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5115.1, Definition 5115.2, and Definition 5115.3 seed the Real-induced metric-ball entourage basis for Cauchy-filter and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealUniformEntourageBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close quotient real equality, selected limits, countable-choice selectors, open-cover compactness, quotient filters, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Real endpoint exposure, metric-ball generation, RealUniformStructure compatibility, CauchyFilter handoff, finite readback, transport, replay, provenance, and local naming for BEDC.Derived.RealUniformEntour

Chapter 5116

A Concrete Naming Certificate for $\mathsf{UniformLebesgueNumber}^{\uparrow}$

Following Lebesgue's covering lemma, Bourbaki and Kelley on uniform spaces, and Bishop–Bridges compact metric analysis, the $\mathsf{UniformLebesgueNumber}^{\uparrow}$ packet records the radius handoff by which a compact-cover lower bound becomes uniform entourage data. It sits after the compact finite-radius packet of chapter 4015, the finite lower-radius fold of chapter 452, the compact metric source of chapter 129, and the uniform-modulus consumer of chapter 369. The packet is a finite NameCert boundary for uniform reads of compact cover radii; it is not an arbitrary open-cover principle, not a selected subcover, and not an ambient minimum theorem.

Definition 5116.1 (Uniform Lebesgue-number carrier)**.** *A* $\mathsf{UniformLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, M, F, V, L, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compact finite-net and cover-inspection data must be read before any radius is exported.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row that interprets every cover-radius and entourage request as metric evidence.*
3. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *lower-bound fold row for the finite cover-radius ledger.*
4. *V is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row that receives the lower-bound ledger as a tolerance-only uniform handoff.*
5. *L is the* $\mathsf{CompactLebesgueNumber}^{\uparrow}$ *route row, carrying the compact cover spine, located coverage rows, positive radius ledger, and finite lower-bound fold together.*
6. *H records componentwise* hsame *transport for the compact source, metric row, finite lower-bound fold, uniform-modulus handoff, compact Lebesgue-number route, replay, provenance, and local name.*
7. *C records displayed* Cont *replay of the route*
$$K, M, L \longmapsto F \longmapsto V.$$
8. *P is the* Pkg *provenance over* $\mathsf{UniformLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactLebesgueNumber}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformLebesgueNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, M, F, V, L, H, C, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, host compactness, host minima, quotient real equality, countable choice, or a uniform radius detached from the compact finite cover ledger.

Lean4 ✓ `BEDC.Derived.UniformLebesgueNumberUp.UniformLebesgueNumberTasteGate single carrier alignment`

Theorem 5116.2 (Uniform Lebesgue-number NameCert obligations)**.** *For every accepted* $\mathsf{UniformLebesgueNumber}$ *carrier*

$$U = (K, M, F, V, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformLebesgueNumber}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the nine displayed rows;*
2. *compact-metric source exposure, requiring every uniform-radius read to pass through K;*
3. *metric interpretation, requiring M to mediate every radius and entourage request before it reaches a uniform consumer;*
4. *finite Lebesgue-number compatibility, requiring F to be the lower-bound fold of the displayed compact cover-radius ledger;*
5. *compact Lebesgue-number route compatibility, requiring L to expose the compact cover spine, located coverage rows, positive radius ledger, and finite lower-bound fold before V is available;*
6. *uniform-modulus handoff, requiring V to consume only the displayed lower-bound ledger routed through K, M, F, L;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No uniform Lebesgue-number obligation is stored outside these rows.

Proof. Project the accepted packet $U = (K, M, F, V, L, H, C, P, N)$. The displayed coordinates give the compact source, metric interpretation, finite lower-bound fold, uniform-modulus handoff, compact Lebesgue-number route, and structural rows.

The compact and metric rows come first. The row K supplies compact finite-net and cover-inspection data, while M keeps radius and entourage reads inside the displayed metric surface.

The finite lower-bound row is then read through F and L. The row L carries the compact cover spine, located coverage rows, positive radius ledger, and finite lower-bound fold, so F is not a host minimum and not a selected subcover argument.

Finally V, H, C, P, N package the same lower-bound ledger for uniform-modulus consumers. Since the excluded objects do not occur as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 5116.3 (Uniform Lebesgue-number compact-uniform handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{UniformLebesgueNumber}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{CompactLebesgueNumber}^{\uparrow} \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Co}$$

Thus the compact cover-radius handoff of chapter 9237 is exposed as compact-metric source data, metric radius interpretation, compact Lebesgue-number cover rows, a finite lower-bound fold, and

a uniform-modulus consumer row. It exports no arbitrary open cover, selected subcover, host compactness theorem, host minimum, countable-choice selector, quotient real equality, or detached uniform radius.

Proof. Start from the source rows K and M. They force every radius read to use the compact-metric source and the displayed metric interpretation before a uniform consumer can inspect the packet.

Read the compact Lebesgue-number route L. By Definition 4015.3, that route exposes the compact cover spine, located coverage rows, positive radius ledger, and finite lower-bound fold before the uniform handoff is available.

The row F records the finite $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ lower-bound fold, and V consumes exactly that lower-bound ledger as a $\mathsf{UniformModulus}^{\uparrow}$ row. The structural rows H, C, P, N transport, replay, source, and name the same finite path.

Because the carrier has no coordinate for arbitrary open covers, selected subcovers, host minima, choice, quotient real equality, or detached uniform radii, every compact-uniform read factors through the displayed route. □

Falsifiable prediction: An accepted $\mathsf{UniformLebesgueNumber}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a UniformModulusUp row without displaying CompactMetric source data, Metric radius interpretation, a CompactLebesgueNumber cover route, a FiniteLebesgueNumber lower-bound fold, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compact-lebesgue-number, finite-lebesgue-number, uniformmodulus: none alone is bijective with $\mathsf{UniformLebesgueNumber}^{\uparrow}$ because the packet jointly carries compact source data, metric radius interpretation, compact cover rows, finite lower-bound fold, uniform-modulus handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLebesgueNumber}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformLebesgueNumber}^{\uparrow}$ packet is generated from CompactMetricUp source data, MetricUp radius interpretation, a CompactLebesgueNumberUp route row, a FiniteLebesgueNumberUp lower-bound fold, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLebesgueNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5116.1, Definition 5116.2, and Definition 5116.3 seed the uniform Lebesgue-number packet from compact finite cover-radius data through lower-bound folds to UniformModulusUp handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLebesgueNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host compactness, host minima, quotient real equality, countable choice, detached uniform radii, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact-source exposure, metric radius interpretation, CompactLebesgueNumber route compatibility, FiniteLebesgueNumber fold compatibility, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformLebesgueNumberUp.

Chapter 5117

A Concrete Naming Certificate for $\mathsf{UniformlyCauchyFilter}^{\uparrow}$

Following Kelley and Bourbaki on uniform spaces, and Bishop–Bridges constructive completion, the $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ packet records the finite route by which a located uniform source exposes Cauchy-filter data to Real-and-Completion consumers. It sits after the located entourage surface of chapter 3814, the Real-induced uniformity of chapter 1624, the filter carrier of chapter 333, the net handoff of chapter 589, the finite window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet is a finite NameCert boundary for uniformly Cauchy filter reads; it does not quotient filters, choose limits, identify host equalities, invoke countable choice, or assert ambient Real completeness.

Definition 5117.1 (Uniformly Cauchy filter carrier)**.** *A* $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (U, L, F, N, W, D, R, E, H, C, P, Q)$$

with the following displayed rows:

1. *U is the* $\mathsf{LocatedUniformSpace}^{\uparrow}$ *entourage row, exposing only named entourage requests and located refinement witnesses.*
2. *L is the uniform-Cauchy schedule, assigning each displayed entourage request a finite filter-stage certificate.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row that reads the accepted filter base after the located entourage and schedule rows have been displayed.*
4. *N is the* $\mathsf{CauchyNet}^{\uparrow}$ *handoff row, available only as a finite net witness for the same entourage request.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by Real-and-Completion consumers.*
6. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the window and entourage requests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows under the dyadic ledger.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, downstream of the located uniform, filter, net, window, dyadic, and regular readback rows.*
9. *H records componentwise* hsame *transport for the located-uniform, schedule, filter, net, window, dyadic, regular-readback, Real-seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a located entourage request through* $U, L, F, N, W, D, R, E, H, P, Q$.
11. *P is the* Pkg *provenance over* $\mathsf{UniformlyCauchyFilter}^{\uparrow}$, $\mathsf{LocatedUniformSpace}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *Q is the local* $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, L, F, N, W, D, R, E, H, C, P, Q$. It has no coordinate for quotient filters, selected limits, countable choice, host equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.UniformlyCauchyFilterUp.UniformlyCauchyFilterTasteGate single carrier alignment`

Theorem 5117.2 (Uniformly Cauchy filter NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ *carrier*

$$G = (U, L, F, N, W, D, R, E, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilter}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the located-uniform entourage row, uniform-Cauchy schedule, CauchyFilter row, CauchyNet handoff, StreamName windows, dyadic tolerance ledger, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming;*
2. *located-entourage adequacy, requiring each Cauchy request to read U before any filter, net, or numeric row is interpreted;*
3. *uniform-Cauchy scheduling, requiring L to exhibit a finite filter-stage certificate for every displayed entourage request;*
4. *filter and net handoff, requiring F and N to consume the same located entourage and schedule rows rather than a quotient filter;*
5. *finite readback, requiring W, D, R, E to expose only finite windows, dyadic tolerances, regular rational readbacks, and the downstream Real seal;*
6. *transport, replay, provenance, and naming through H, C, P, Q for the same displayed route;*
7. *nonescape for quotient filters, selected limits, countable choice, host equality, and ambient Real completeness.*

No uniformly Cauchy filter obligation is stored outside these rows.

Proof. Project the carrier of Definition 5117.1. The accepted packet exposes exactly the located entourage row, the schedule row, the filter and net handoff rows, the finite numeric readback rows, and the structural rows.

The uniformly Cauchy assertion is schedule-local. Each entourage request is first read from U, and the finite stage evidence in L is the only admissible route by which the request reaches F or N.

The completion-facing read is then numeric but still finite. Stream windows W are inspected under the dyadic ledger D, regular rational readback is recorded in R, and the Real seal E remains downstream of those displayed rows.

Finally, H, C, P, Q transport, replay, source, and name the same packet. Since quotient filters, selected limits, countable choice, host equality, and ambient Real completeness are not carrier rows, they cannot be exported by the local NameCert surface. □

Theorem 5117.3 (Uniformly Cauchy filter located-uniform handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ *packet reads the Cauchy-filter route only through the located entourage row, the uniform-Cauchy schedule, the CauchyFilter row, the CauchyNet handoff, StreamName windows, positive DyadicRatCore tolerances, RegSeqRat readbacks, the downstream Real seal, and the structural rows* H, C, P, Q*. The handoff exports no quotient filter, selected limit, countable-choice selector, host equality, or ambient Real completeness.*

Proof. Begin with the accepted carrier $G = (U, L, F, N, W, D, R, E, H, C, P, Q)$. Carrier admission gives the consumer no access beyond these rows.

Use the obligation surface of Definition 5117.2. It forces every filter read to pass through the located entourage row U and the finite schedule L before either the CauchyFilter row F or the CauchyNet handoff N can be inspected.

The finite numeric handoff then follows the displayed order W, D, R, E. The structural rows preserve that order under componentwise transport and replay, so the consumer obtains a located-uniform Cauchy route without receiving any excluded quotient, selection, host-equality, or completeness coordinate. □

Theorem 5117.4 (Uniformly Cauchy filter Real completion boundary)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal reached by an accepted* $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ *packet is downstream of the located uniform data, the uniform-Cauchy schedule, the filter and net handoff rows, finite windows, dyadic tolerance ledger, and regular rational readback. Therefore the packet may feed a completion-facing Real consumer, but it does not assert a selected completion point, quotient equality of filters, countable choice, host equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Proof. Apply Definition 5117.3. Any completion-facing read has already passed through U, L, F, N, W, D, R before the Real seal row E is available.

Apply Definition 5117.2 to the seal row. Its finite readback clause permits E only as the terminal boundary of the displayed window, dyadic, and regular-rational route, not as an ambient completeness theorem or a selected limit.

The remaining rows H, C, P, Q only transport, replay, source, and name this route. Hence the Real completion boundary is a downstream seal for the packet rather than a quotient-filter construction or a host equality principle. □

Falsifiable prediction: An accepted $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ read is refuted if a completion-facing consumer reaches a CauchyFilter or Real seal without displaying located entourage data, a uniform-Cauchy schedule, a filter or net handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: located-uniform-space, real-uniform-structure, cauchyfilter, cauchynet, streamname, regseqrat, dyadicratcore, real: no listed sibling alone is bijective with $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ because the packet jointly carries located entourage data, a uniform-Cauchy schedule, filter and net handoff, finite windows, dyadic tolerance, regular readback, Real sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformlyCauchyFilter}^{\uparrow}$ packet is generated from located uniform entourage rows, a uniform-Cauchy schedule, CauchyFilter and CauchyNet handoff rows, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a downstream Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyCauchyFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5117.1, Definition 5117.2, Definition 5117.3, and Definition 5117.4 seed the uniformly Cauchy filter route for completion-facing Real consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{UniformlyCauchyFilter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected limits, countable-choice selectors, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked located-entourage adequacy, uniform-Cauchy scheduling, filter and net handoff, finite readback, transport, replay, provenance, and local naming for BEDC.Derived.UniformlyCauchyFilterUp.

Chapter 5118

A Concrete Naming Certificate for $\mathsf{RealCauchyRate}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RealCauchyRate}^{\uparrow}$ packet records the finite rate certificate by which a regular rational sequence is sealed as a real name. It sits after the positive dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, and the regular-sequence readback of chapter 301, and before the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The chapter names the rate boundary only; it does not assert quotient real equality, selected representatives, trichotomy, countable choice, or ambient real completeness.

Definition 5118.1 (Real Cauchy-rate carrier)**.** *A* $\mathsf{RealCauchyRate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-tolerance ledger, carrying the dyadic radii and comparison budget used to test a regular Cauchy tail.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the tolerance and tail indices requested by D.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, recording the rational readback and tail-comparison schedule served by the window row S.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row receiving the accepted regular rational name after the displayed rate data has been read.*
5. *H records componentwise* hsame *transport for the dyadic tolerance, stream window, regular readback, Real seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from positive dyadic tolerance requests through finite StreamName windows, RegSeqRat regularity, and the Real seal handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{RealCauchyRate}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyRate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, Q, R, H, C, P, N. It has no coordinate for quotient real equality, a selected representative stream, trichotomy, countable-choice selection, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyRateUp.RealCauchyRateTasteGate single carrier alignment`

Theorem 5118.2 (Real Cauchy-rate regular NameCert)**.** *For every accepted* $\mathsf{RealCauchyRate}^{\uparrow}$ *carrier* $K = (D, S, Q, R, H, C, P, N)$*, the local NameCert surface consists exactly of carrier admission, positive dyadic tolerance stability,* $\mathsf{RegSeqRat}^{\uparrow}$ *compatibility,* $\mathsf{Real}^{\uparrow}$ *seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and ledger non-escape from quotient real equality, selected representatives, trichotomy, countable choice, and ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyRateCarrier regular namecert`

Proof. Project the carrier of Definition 5118.1. The projection exposes only the dyadic tolerance ledger, finite stream window, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Tolerance stability is read from the row D. A positive dyadic request can be transported only by the componentwise hsame row H and replayed only by the displayed Cont row C, so the same tolerance budget remains visible after transport.

RegSeqRat compatibility is downstream of the finite window. The window row S is opened before the regularity row Q is inspected, and the theorem records that the Real-facing readback remains a regular rational tail rather than a host stream quotient.

The Real seal handoff is the row R. It consumes the displayed route D, S, Q and exports only the named real boundary supplied by the packet.

Finally P and N exhaust provenance and naming. Since the excluded host objects are not carrier coordinates, the local ledger cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyRate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCauchyRate}^{\uparrow}$ packet is generated from a DyadicRatCore positive-tolerance ledger, finite StreamName windows, RegSeqRat regular rational readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCauchyRate}^{\uparrow}}$, seed)

Scope closed: Definition 5118.1 and Definition 5118.2 seed the finite rate certificate between $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCauchyRate}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.RealCauchyRateUp`

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, trichotomy, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance stability, regular-sequence compatibility, Real seal handoff, and ledger non-escape for the same finite rate packet.

$\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ reads repeated completion through chapter 3484 and chapter 1425 before the terminal $\mathsf{Real}^{\uparrow}$ seal.

Chapter 5119

A Concrete Naming Certificate for $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ packet records the finite witness that applying regular-Cauchy completion to an already displayed regular-Cauchy completion route does not create a second ambient completion object. It sits after the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the idempotence boundary of **??**, and the terminal real-name seal of chapter 38. The packet names the Bishop-style flattening witness as BEDC rows, not as quotient representatives, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The sibling dependency route is explicit: the packet consumes **??** as the flattening row and chapter 38 as the terminal seal, while chapter 126, chapter 291, and chapter 301 provide the finite observation and readback substrate. Thus the witness is part of the L10 completion lattice rather than a kernel-only singleton.

Definition 5119.1 (Bishop completion-idempotence witness carrier)**.** *A* $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the source and repeated-completion requests.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the window comparison and regularity readback.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row carrying the regular rational source readback visible to the completion route.*
4. *I is the* $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$ *row flattening the repeated completion boundary to the displayed completion route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, R, I have been displayed.*
6. *H records componentwise* hsame *transport for the window, tolerance, readback, idempotence, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, R, I, E, H, C, P, N$. It has no coordinate for quotient regular sequences, selected representatives, a second completed object, countable choice, host equality of completions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5119.2 (Bishop completion-idempotence witness obligations)**.** *For every accepted* BishopCompletionIdem *carrier*

$$W = (S, D, R, I, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}}$ *surface consists exactly of finite stream-window exposure through S, dyadic tolerance exposure through D, regular rational readback through R, displayed completion-idempotence flattening through I, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5119.1. The rows S, D, R expose the regular-Cauchy source route, while I supplies the displayed idempotence row consumed by the completion-facing read.

Use the sibling surfaces in order. The stream windows come from chapter 126, the tolerance ledger from chapter 291, and the regular rational readback from chapter 301. The flattening row is exactly the finite idempotence boundary of **??**.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Because the carrier contains no quotient, selector, second-completion, choice, host-equality, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5119.3 (Bishop completion-idempotence witness route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{CauchyCompletionIdempotence}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Thus repeated regular-Cauchy completion is flattened by displayed BEDC rows before the terminal real seal is consumed. The route exports no quotient representative, selected limit, second completion carrier, countable-choice schedule, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with Definition 5119.2. It forces every consumer read through the finite window, dyadic tolerance, regular readback, idempotence, and real-seal rows before the local certificate can be inspected.

The $\mathsf{CauchyCompletionIdempotence}^{\uparrow}$ row then identifies the repeated-completion boundary with the displayed completion route. Since the row is downstream of S, D, R, the flattening is a finite regular-Cauchy read rather than a comparison of quotient representatives.

The terminal row E is only the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The absence of quotient, selector, choice, host-equality, second-completion, and ambient-completeness coordinates prevents any such object from entering the exported route. □

Falsifiable prediction: Every accepted $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ read exposes StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, CauchyCompletionIdempotence flattening, RealUp seal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any repeated-completion consumer is served.

Independence witness: streamname, dyadicratcore, regseqrat, cauchy-completion-idempotence, real: none is bijective with $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ because this packet alone binds the regular-Cauchy observation route, displayed completion-idempotence flattening row, terminal RealUp seal, transport, replay, provenance, and local naming as one witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, a CauchyCompletionIdempotence flattening row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5119.1, Definition 5119.2, and Definition 5119.3 seed the finite Bishop-style repeated-completion flattening witness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionIdempotenceWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality of completed objects, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window coverage, dyadic tolerance exactness, RegSeqRat readback, idempotence flattening, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionIdempotenceWitnessUp.

The completion-theorem forward route makes the horizontal L10 edge explicit: every BishopCompletionIdempote read that serves a completion-facing consumer first passes through the theorem-level completion surface of chapter 3484 and the universal regular-Cauchy route of chapter 1425 before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed. Thus idempotence is read as a displayed route through the Bishop theorem packet and the regular-Cauchy universal packet, not as a second completed object, a quotient representative, selected limit data, countable choice, host equality of completions, or ambient real completeness.

Chapter 5120

A Concrete Naming Certificate for $\mathsf{MetricReverseTriangleInequality}^{\uparrow}$

Following Bishop and Bridges on constructive metric estimates, the $\mathsf{MetricReverseTriangleInequality}^{\uparrow}$ packet records the finite route behind the estimate $|d(x,z) - d(y,z)| \leq d(x,y)$ without appealing to a host absolute-value operator or an ambient real-order theorem. It sits over the distance rows of chapter 57, the order-and-operation surface of chapter 766, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the regular-Cauchy reverse-triangle route of chapter 1894. The chapter packages the standard metric reverse-triangle estimate as finite BEDC NameCert data at L10 Phase Real and Completion.

Definition 5120.1 (Metric reverse-triangle inequality carrier)**.** *A* $\mathsf{MetricReverseTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, D_{xz}, D_{yz}, G, E, W, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row exposing only the finite distance surface and its local metric NameCert data.*
2. *D_{xz} and D_{yz} are the two metric-distance rows for the displayed endpoint pairs (x,z) and (y,z).*
3. *G is the absolute-gap comparison row for the expression $|d(x,z) - d(y,z)|$; it is read only through the* $\mathsf{RealAlgOrder}^{\uparrow}$ *operation-and-order surface.*
4. *E is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested comparison precision and the budget used to compare the two distance readbacks.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window data serving the two distance streams and the gap stream at the requested scale.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the accepted distance rows and the accepted absolute-gap row.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reachable only after the distance rows, comparison row, dyadic tolerance row, stream windows, and regular readback have been displayed.*
8. *H records componentwise* hsame *transport for the metric carrier, distance rows, absolute-gap comparison, tolerance row, window data, readback, seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from any reverse-triangle-facing request through* $M, D_{xz}, D_{yz}, G, E, W, R, S, H, P$
10. *P is the* Pkg *provenance over* MetricReverseTriangleInequality$^{\uparrow}$, Met$^{\uparrow}$, RealAlgOrder$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, RegularCauchyReverseTriangle$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{MetricReverseTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D_{xz}, D_{yz}, G, E, W, R, S, H, C, P, N$. The carrier has no coordinate for host absolute values, quotient real equality, open-cover compactness, selected nearest points, ambient Real$^{\uparrow}$ *completeness, Lean theorem checking, or bridge checking.*

Lean4 (definition) `BEDC.Derived.MetricReverseTriangleInequalityUp`

Lean4 ✓ `BEDC.Derived.MetricReverseTriangleInequalityUp.MetricReverseTriangleInequalityTasteGate single carrier alignment`

Theorem 5120.2 (Metric reverse-triangle NameCert obligations)**.** *For every accepted* MetricReverseTriangleInequal *carrier*

$$T = (M, D_{xz}, D_{yz}, G, E, W, R, S, H, C, P, N),$$

the local NameCert$_{\text{MetricReverseTriangleInequality}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* Met$^{\uparrow}$ *row, the two metric-distance rows, the absolute-gap comparison row, the dyadic tolerance row, finite stream windows, regular readback, Real seal, and structural rows;*
2. *metric distance exposure, requiring D_{xz} and D_{yz} to be read only through the finite distance surface of M before the absolute-gap comparison is available;*
3. *comparison and tolerance exactness, requiring G and E to record the absolute-gap comparison through* RealAlgOrder$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *at the displayed precision scale;*
4. *readback and seal nonescape, requiring W, R, S to place the finite* StreamName$^{\uparrow}$ *windows and* RegSeqRat$^{\uparrow}$ *readback before any* Real$^{\uparrow}$ *consumer receives the estimate;*
5. *componentwise transport, continuation replay, package provenance, and local naming, requiring H, C, P, N to preserve and name exactly the same finite reverse-triangle packet.*

No NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.MetricReverseTriangleInequalityCarrier namecert obligations`

Theorem 5120.3 (Metric reverse-triangle distance-gap factorization)**.** *For every accepted* MetricReverseTriangleIne *carrier $T = (M, D_{xz}, D_{yz}, G, E, W, R, S, H, C, P, N)$, every accepted read of the gap $|d(x, z) - d(y, z)|$ factors through the displayed metric rows D_{xz}, D_{yz} before it reaches the comparison row G. The comparison then passes through the* RealAlgOrder$^{\uparrow}$ *operation-and-order surface, the* DyadicRatCore$^{\uparrow}$ *tolerance row E, the finite* StreamName$^{\uparrow}$ *windows W, and the* RegSeqRat$^{\uparrow}$ *readback R before the terminal* Real$^{\uparrow}$ *seal S is exposed.*

Lean4 ✓ `BEDC.Derived.MetricReverseTriangleInequalityUp.MetricReverseTriangleInequality distance gap`

Closure status for NameCert$_{\text{MetricReverseTriangleInequality}^{\uparrow}}$ **.**

Theory closure: TheoryGate(NameCert$_{\text{MetricReverseTriangleInequality}^{\uparrow}}$, seed)

Scope closed: Definition 5120.1, Definition 5120.2, and Definition 5120.3 seed the finite metric reverse-triangle route through displayed distance rows, absolute-gap comparison, dyadic tolerance, finite windows, regular readback, and Real seal data.

Formal verification: FormalStatus(NameCert$_{\text{MetricReverseTriangleInequality}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.MetricReverseTriangleInequalityUp`

Bridge status: none

Not claimed: This chapter does not close host absolute values, quotient real equality, open-cover compactness, selected nearest points, Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric distance exposure, comparison and tolerance exactness, readback and seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.MetricReverseTriangleInequalityUp.

Constructive story: A MetricReverseTriangleInequality$^{\uparrow}$ packet is generated from a MetricUp carrier, two metric-distance rows, a RealAlgOrder absolute-gap comparison row, a DyadicRatCore tolerance row, finite StreamName windows, a RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, package provenance, and local naming.

Chapter 5121

A Concrete Naming Certificate for $\mathsf{UpperRealEnvelope}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{UpperRealEnvelope}^{\uparrow}$ packet records a one-sided real-envelope surface obtained by reading upper dyadic bounds beside the lower-upper cut boundary of chapter 2087. It keeps the envelope as finite approximation data through chapter 291, chapter 126, chapter 301, and chapter 38, without selecting a completed cut representative or a host supremum.

Definition 5121.1 (Upper real envelope carrier)**.** *An* $\mathsf{UpperRealEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (U, L, D, S, R, A, H, C, P, N)$$

with the following displayed rows:

1. *U is the upper-envelope ledger, recording finite upper dyadic observations and their tolerance budgets.*
2. *L is the* $\mathsf{LowerUpperRealCut}^{\uparrow}$ *boundary row, exposing the two-sided located-cut context against which the envelope is read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison row, storing endpoints, radii, slack, and decidable dyadic comparisons used by U and L.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, scheduling visible envelope queries without selecting an infinite representative.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy handoff, carrying the same upper-budget reads into regular sequence form.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *admissibility seal, available only downstream of U, L, D, S, R.*
7. *H records componentwise* hsame *transport for the envelope ledger, lower-upper boundary, dyadic row, stream window, regular-Cauchy handoff, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for every envelope-facing read through the same tuple $U, L, D, S, R, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{UpperRealEnvelope}^{\uparrow}$, $\mathsf{LowerUpperRealCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperRealEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, L, D, S, R, A, H, C, P, N$. It has no coordinate for quotient-cut equality, a chosen Dedekind representative, trichotomy, countable choice, host supremum, or ambient real completeness.

Theorem 5121.2 (Upper real envelope NameCert obligations)**.** *For an accepted* $\mathsf{UpperRealEnvelope}^{\uparrow}$ *carrier*

$$E = (U, L, D, S, R, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UpperRealEnvelope}^{\uparrow}}$ *surface consists exactly of carrier exposure, upper-envelope ledger stability, lower-upper boundary visibility, dyadic tolerance exactness, finite-window scheduling, regular-Cauchy handoff, terminal real-seal nonescape, componentwise transport, displayed replay, provenance, and local naming. No local obligation is stored outside those rows.*

Lean4 ✓ `BEDC.Derived.UpperRealEnvelopeUp.UpperRealEnvelopeTasteGate single carrier alignment`

Proof. Carrier exposure is projection from Definition 5121.1. The accepted packet displays exactly the upper ledger, lower-upper boundary, dyadic tolerance row, stream window, regular-Cauchy handoff, real seal, and structural rows.

The upper-envelope ledger is read only through U, D, S, H, C, P, N. Componentwise transport preserves the displayed dyadic budget, and replay uses the same finite window rather than a hidden stream representative.

The lower-upper boundary is the row L. It keeps every envelope read adjacent to the two-sided located-cut surface of chapter 2087, so the upper ledger cannot detach into an isolated host-side bound.

The handoff rows R and A remain downstream of U, L, D, S. Since no other coordinate exists, the packet cannot export quotient-cut equality, a selected representative, trichotomy, countable choice, host supremum, or ambient completeness. □

Theorem 5121.3 (Upper real envelope lower-upper cut boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{UpperRealEnvelope}^{\uparrow}$ *packet factors through the displayed route*

$$U, L, D, S, R, A, H, C, P, N$$

and keeps the $\mathsf{LowerUpperRealCut}^{\uparrow}$ *boundary row visible before any regular-Cauchy or terminal real seal can be consumed. Thus the packet supplies one-sided upper approximation evidence beside a lower-upper cut, not a completed real, quotient cut, selected representative, or host supremum.*

Proof. Fix an accepted $\mathsf{UpperRealEnvelope}^{\uparrow}$ carrier. A consumer first reads the upper-envelope ledger U with the dyadic tolerance row D and the stream window S.

The lower-upper cut boundary L is displayed in the same packet. This keeps the one-sided upper observations attached to the two-sided located boundary before any Real-facing read is allowed.

The regular-Cauchy handoff R and real seal A sit downstream of U, L, D, S. Transport, replay, provenance, and local naming preserve that same finite route.

Because the carrier has no coordinate for a quotient cut, selected representative, host supremum, trichotomy, countable choice, or completeness principle, the route can export only upper-envelope approximation evidence beside the named lower-upper boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{UpperRealEnvelope}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperRealEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5121.1, Definition 5121.2, and Definition 5121.3 seed the finite one-sided upper-envelope surface beside lower-upper cuts and regular-Cauchy real seals.

Formal verification: FormalStatus(NameCert$_{\mathsf{UpperRealEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind construction of Real$^{\uparrow}$, quotient-cut equality, selected representative, host supremum, countable choice, trichotomy, or Real$^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier exposure, upper-ledger stability, lower-upper boundary visibility, dyadic exactness, finite-window scheduling, regular-Cauchy handoff, real-seal nonescape, and local naming for BEDC.Derived.UpperRealEnvelopeUp.

Constructive story: An UpperRealEnvelope$^{\uparrow}$ packet is generated from an upper-envelope ledger, a LowerUpperRealCut boundary row, a DyadicRatCore tolerance row, a StreamName finite window, a RegSeqRat handoff, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5122

A Concrete Naming Certificate for $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$

The $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ packet realizes the claim-status surface of chapter 9456 as a finite NameCert carrier. It sits beside the kernel witness of chapter 546, the audit gate boundary of chapter 526, the apophatic naming surface of chapter 543, and the corpus-supply distillation packet of chapter 968. Those siblings audit declarations, local gates, named externality, and source supply separately; this packet binds claim source, admissibility gate, refusal socket, marker provenance, replay, and local naming as one visible ledger.

Definition 5122.1 (Legitimate-claim audit ledger carrier)**.** *A* $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, G, R, M, K, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the claim-source row, recording the finite expression, subject surface, cited carrier, and supplied evidence route.*
2. *G is the admissibility-gate row, read through* $\mathsf{AuditGateBoundary}^{\uparrow}$ *and recording whether the claim may enter as theorem, condition, route, evidence, non-evidence, meta-row, refusal, gap, or socket.*
3. *R is the refusal-socket row, recording the named boundary where hidden supply, ungenerated proof content, unmarked corpus material, or external truth pressure is refused.*
4. *M is the marker-provenance row, recording the paper marker, Lean target name when present, source chapter, and resolution status without asserting a theorem absent kernel acceptance.*
5. *K is the kernel-audit row, read through* $\mathsf{KernelAuditWitness}^{\uparrow}$ *when the claim cites Lean-accepted theorem content.*
6. *A is the apophatic-name row, read through* $\mathsf{ApophaticName}^{\uparrow}$ *when the claim exposes an external dependency position rather than consuming it.*
7. *H records componentwise* hsame *transport for source, gate, refusal socket, marker provenance, kernel audit, apophatic name, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for the same ledger route through $S, G, R, M, K, A, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$, $\mathsf{KernelAuditWitness}^{\uparrow}$, $\mathsf{AuditGateBoundary}^{\uparrow}$, $\mathsf{ApophaticName}^{\uparrow}$, $\mathsf{CorpusSupplyDistillation}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{LegitimateClaimAuditLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, G, R, M, K, A, H, C, P, N$. It has no coordinate for hidden theorem supply, unledgered corpus import, a global truth predicate, host-side proof search, or an unmarked marker resolution.

Theorem 5122.2 (Legitimate-claim audit ledger NameCert obligations)**.** *For an accepted* LegitimateClaimAuditLed *carrier*

$$L = (S, G, R, M, K, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LegitimateClaimAuditLedger}^{\uparrow}}$ *surface consists exactly of claim-source exposure, admissibility-gate classification, refusal-socket visibility, marker-provenance exactness, kernel-audit handoff, apophatic-name handoff, componentwise transport, displayed replay, provenance, and local naming. No claim-status obligation is stored outside those rows.*

Proof. Carrier exposure is projection from Definition 5122.1. The accepted packet displays the claim source, admissibility gate, refusal socket, marker provenance, kernel audit row, apophatic name row, and structural rows.

The gate row G classifies the finite expression only with the source row S visible. If a requested dependency is not generated by the packet, the refusal socket R and apophatic row A keep that dependency named rather than consumed.

Marker provenance is the row M. When the claim cites Lean theorem content, the kernel row K must be read through the same packet; otherwise the marker remains a provenance row rather than a theorem export.

Transport, replay, provenance, and local naming preserve the displayed ledger. Since the carrier has no other coordinate, it cannot import hidden theorem supply, unledgered corpus material, host proof search, or a global truth predicate. □

Theorem 5122.3 (Legitimate-claim audit ledger nonescape)**.** *Every accepted* $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ *claim-status read factors through the displayed route*

$$S, G, R, M, K, A, H, C, P, N.$$

In particular, a claim whose missing dependency is named by R or A cannot be exported as an unconditional theorem row unless a separate kernel-audit or certificate row is displayed in the same ledger route.

Proof. Start from Definition 5122.2. A consumer receives source, gate, refusal, marker, kernel, apophatic, transport, replay, provenance, and local naming data as one finite packet.

If a dependency is generated and accepted, the route may read it through the displayed kernel-audit row K or another displayed certificate row at the claim site. If the dependency is ungenerated, the route exposes it through R or A.

The rows H, C, P, N preserve and name the same distinction. They do not add a theorem row, select a hidden proof, or turn apophatic naming into internal supply.

Therefore the ledger can classify theorem, condition, route, evidence, non-evidence, meta-row, refusal, gap, and socket statuses, but it cannot promote a named missing dependency to an unconditional theorem. □

The Lean follow-up file `lean4/BEDC/Derived/LegitimateClaimAuditLedgerUp/TasteGate.lean` must provide explicit Nontrivial $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ and FieldFaithful $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ instances. The field-faithfulness witness must preserve $S, G, R, M, K, A, H, C, P, N$ as distinct rows

so claim source, admissibility gate, refusal socket, marker provenance, kernel audit, and apophatic naming cannot collapse.

Falsifiable prediction: A claim-status packet that omits the refusal socket while consuming ungenerated theorem supply is rejected as a $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ packet, even when it carries a marker-provenance row or an admissibility-gate row.

Independence witness: kernelauditwitness, auditgateboundary, apophaticname, corpussupply-distillation: $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries claim source, admissibility gate, refusal socket, marker provenance, kernel audit, apophatic naming, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LegitimateClaimAuditLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{LegitimateClaimAuditLedger}^{\uparrow}$ packet is generated from claim source, admissibility gate, refusal socket, marker provenance, kernel audit, apophatic naming, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LegitimateClaimAuditLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5122.1, Definition 5122.2, and Definition 5122.3 seed the finite audit-ledger surface for legitimate claim status.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LegitimateClaimAuditLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, a global truth predicate, hidden proof search, kernel soundness, or unledgered corpus admission.

Upgrade path: Obligation closure requires checked carrier admission, claim-source exposure, admissibility-gate classification, refusal-socket visibility, marker-provenance exactness, kernel and apophatic handoffs, TasteGate nontriviality, and field faithfulness for $S, G, R, M, K, A, H, C, P, N$.

Chapter 5123

A Concrete Naming Certificate for $\mathsf{PerronFrobenius}^{\uparrow}$

Following Perron and Frobenius, the $\mathsf{PerronFrobenius}^{\uparrow}$ packet records the finite linear-algebra route by which a nonnegative matrix exposes a positive cone, iterate windows, normalized comparison rows, radius bounds, and eigenvector-window evidence. It consumes the matrix carrier of chapter 49, the vector-space carrier of chapter 47, the positive-real boundary of chapter 1587, and the real-name seal of chapter 38. The packet is a NameCert interface for downstream finite spectral consumers; it does not import host eigenvectors, spectral choice, classical compactness, quotient real equality, or an ambient linear-operator theory.

Definition 5123.1 (PerronFrobenius carrier)**.** *A* $\mathsf{PerronFrobenius}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, C, I, R, E, H, T, P, N)$$

with the following displayed rows:

1. *M is the nonnegative square* $\mathsf{Mat}^{\uparrow}$ *row, exposing its entries only through displayed* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{PositiveReal}^{\uparrow}$ *comparison data.*
2. *C is the positive-cone row in the ambient* $\mathsf{VecSpace}^{\uparrow}$ *source, recording which displayed vector histories are admitted as nonnegative or positive witnesses for the matrix action.*
3. *I is the finite iterate-and-normalization ledger, listing the chosen powers, cone inputs, normalization denominators, and normalized output rows inspected by the certificate.*
4. *R is the radius-and-eigenvector-window row, recording finite lower and upper real bounds for the candidate radius together with a finite window where Mv and rv are compared.*
5. *E is the endpoint row that exposes the final cone-compatible comparison between the normalized iterate window and the radius window.*
6. *H records componentwise* hsame *transport for matrix, cone, iterate, radius-window, endpoint, replay, provenance, and naming rows.*
7. *T records displayed* Cont *replay from the matrix row through the cone, iterate, normalization, radius-window, and endpoint rows.*
8. *P is the* Pkg *provenance over* $\mathsf{PerronFrobenius}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{PositiveReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PerronFrobenius}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, C, I, R, E, H, T, P, N$. It has no coordinate for a selected host eigenvector, a completed projective cone, spectral radius as a quotient real, compactness of the simplex, irreducibility classification, or an external Jordan form.

Lean4 ✓ `BEDC.Derived.PerronFrobeniusUp.PerronFrobeniusTasteGate single carrier alignment`

Theorem 5123.2 (PerronFrobenius NameCert obligations)**.** *For every accepted* $\mathsf{PerronFrobenius}^{\uparrow}$ *carrier $K = (M, C, I, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PerronFrobenius}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonnegative matrix row, positive-cone row, finite iterate-and-normalization ledger, radius-and-eigenvector-window row, endpoint comparison row, componentwise transport, displayed replay, package provenance, and local naming. Any consumer that reads a Perron-facing radius or cone-vector witness must expose M, C, I, R, E before it consumes the structural rows H, T, P, N.*

Proof. Project the carrier of Definition 5123.1. The analytic rows are the matrix row M, positive-cone row C, iterate-and-normalization ledger I, radius-and-window row R, and endpoint row E.

The NameCert obligations are precisely the projections of those rows together with the structural data. Matrix nonnegativity is read through M, cone admission through C, finite power and normalization data through I, candidate radius and eigenvector-window comparison through R, and endpoint compatibility through E.

The rows H, T, P, N transport, replay, source, and name the same packet. Since no host eigenvector, spectral-choice operator, quotient real, compactness argument, or external Jordan form appears as a carrier coordinate, the displayed rows exhaust the local certificate surface. □

Theorem 5123.3 (PerronFrobenius positive cone route)**.** *Every positive-cone read of an accepted* $\mathsf{PerronFrobenius}^{\uparrow}$ *packet factors through the finite route*

$$M, C \longmapsto I \longmapsto R, E \longmapsto H, T, P, N.$$

Thus a downstream spectral consumer receives only a nonnegative matrix row, a displayed cone row, finite iterate and normalization data, and a finite radius-window comparison; it does not receive a selected Perron eigenvector or a host spectral theorem.

Proof. Start with an accepted packet $K = (M, C, I, R, E, H, T, P, N)$. The only matrix source is M, and the only cone source is C.

The route next reads the finite iterate-and-normalization ledger I. This ledger is finite BEDC data: it records displayed powers, cone inputs, normalization denominators, and normalized output rows rather than an infinite limiting process.

The radius-and-eigenvector-window row R and endpoint row E compare only finite windows. They can witness the compatibility of the displayed iterate window with the candidate radius bounds, but they do not choose a global eigenvector or identify a host spectral radius.

Finally the route passes through H, T, P, N. These rows preserve the same displayed packet under hsame, Cont, Pkg, and local naming, so every positive-cone consumer remains inside the finite carrier surface. □

Falsifiable prediction: A $\mathsf{PerronFrobenius}^{\uparrow}$ consumer that reads a Perron-facing radius or cone-vector witness without exposing the matrix row, positive-cone row, iterate ledger, radius-window row, and endpoint row refutes the carrier surface.

Independence witness: matrix, vecspace, positivereal, real: no listed sibling supplies the whole nine-row packet $K = (M, C, I, R, E, H, T, P, N)$, because the siblings provide matrix entries, vector operations, positive-real comparisons, and real seals separately while this chapter binds them to one finite PerronFrobenius NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{PerronFrobenius}\uparrow}$.

Constructive story: A PerronFrobenius$^{\uparrow}$ packet is generated from a nonnegative Matrix row, a positive VecSpace cone row, finite iterate and normalization data, a radius-and-eigenvector-window row, an endpoint comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PerronFrobenius}\uparrow}$, seed)

Scope closed: Definition 5123.1, Definition 5123.2, and Definition 5123.3 seed the finite Perron-Frobenius carrier over Matrix, VecSpace, PositiveReal, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PerronFrobenius}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a selected host eigenvector, spectral choice, quotient-real spectral radius, irreducibility classification, classical compactness, external Jordan form, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, matrix nonnegativity exposure, positive-cone exposure, iterate normalization, radius-window comparison, endpoint compatibility, transport, replay, provenance, and local naming for BEDC.Derived.PerronFrobeniusUp.

Chapter 5124

A Concrete Naming Certificate for $\mathsf{BishopIntervalCover}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalCover}^{\uparrow}$ packet records finite interval-cover data for a displayed closed bounded interval. It sits after the closed bounded interval source of chapter 1591, the dyadic cover rows of chapter 1428, the dyadic endpoint arithmetic of chapter 291, the located cover boundary of chapter 1885, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records a finite cover ledger and its readback route; it is not open-cover compactness, a choice-selected subcover, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5124.1 (BishopIntervalCover carrier)**.** *A* $\mathsf{BishopIntervalCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (E, D, F, L, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint row. It records the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the endpoint-order ledger, and the closed bounded interval source read through* $\mathsf{ClosedBoundedInterval}^{\uparrow}$*.*
2. *D is the dyadic interval-cell row. It records the finite dyadic cells, their lower and upper dyadic endpoints, mesh scale, radius ledger, and containment tests read through* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *F is the finite cover ledger. It records the list of accepted covering cells, the source-interval slots they cover, and the displayed finite subcover boundary for the requested interval window.*
4. *L is the located containment row. It records the located membership and endpoint-containment witnesses connecting E, D, F to the located cover surface of* $\mathsf{LocatedIntervalCover}^{\uparrow}$*.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It schedules the finite endpoint, cell, containment, and cover-ledger observations used by the packet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It translates the displayed StreamName windows and dyadic cells into regular rational observations at the requested mesh and tolerance.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is reachable only after E, D, F, L, W, Q have been displayed.*
8. *H records componentwise* hsame *transport for endpoint rows, dyadic cells, finite cover ledgers, located containment, stream windows, regular readback, the Real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from any finite cover request through the same $E, D, F, L, W, Q, R, H, P, N$ rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalCover}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, F, L, W, Q, R, H, C, P, N$. It has no coordinate for arbitrary open covers, choice-selected subcovers, quotient endpoint equality, countable choice, selected host covers, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalCoverUp.BishopIntervalCoverTasteGate single carrier alignment`

Theorem 5124.2 (BishopIntervalCover NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalCover}^{\uparrow}$ *carrier*

$$B = (E, D, F, L, W, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalCover}^{\uparrow}}$ *surface consists exactly of carrier admission for endpoint rows, dyadic interval cells, finite cover ledgers, located containment witnesses, StreamName windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every interval-cover consumer must expose the finite cover ledger F and the located containment row L before it can read the regular rational row Q or the Real seal R.*

Proof. Project the carrier of Definition 5124.1. The source interval is the endpoint row E, the cover data are the dyadic cell row D and finite cover ledger F, and the local containment evidence is the row L.

The readback rows are downstream of this finite cover surface. The StreamName row W schedules only finite endpoint, cell, containment, and ledger observations, while Q repacks those displayed observations as regular rational data.

The terminal seal R is therefore not an independent compactness object. The structural rows H, C, P, N transport, replay, package, and name the same finite packet, and the carrier contains no open-cover compactness, selected subcover, quotient equality, choice, selected host-cover, or ambient completeness coordinate. □

Theorem 5124.3 (BishopIntervalCover dyadic subcover readback)**.** *Every finite subcover read of an accepted* $\mathsf{BishopIntervalCover}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{DyadicIntervalCover}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedIntervalCover}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{Reg}$$

Thus the exported subcover is a finite dyadic-and-located ledger with regular rational readback, not an open-cover compactness theorem or a choice-selected subcover functional.

Proof. Start with $B = (E, D, F, L, W, Q, R, H, C, P, N)$. The source interval must be read from E, so every subcover request begins at the displayed $\mathsf{ClosedBoundedInterval}^{\uparrow}$ boundary.

The dyadic part of the route is D together with the finite ledger F. These rows expose the mesh, dyadic endpoints, radius ledger, finite membership tests, and retained covering cells through the dyadic siblings.

The located part of the route is L. It witnesses that the retained cells are read against the displayed endpoint and containment data, so the regular rational readback Q is attached to finite located containment rather than to a host open cover.

Finally W, Q, H, C, P, N provide the finite window schedule, regular readback, transport, replay, provenance, and naming. Since no coordinate stores a selector for arbitrary covers, the subcover readback is exactly the finite route displayed above. □

Theorem 5124.4 (BishopIntervalCover Real seal nonescape)**.** *For every accepted* $\mathsf{BishopIntervalCover}^{\uparrow}$ *carrier* $B = (E, D, F, L, W, Q, R, H, C, P, N)$*, any consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *has already exposed the endpoint row* E*, dyadic cell row* D*, finite cover ledger* F*, located containment row* L*, StreamName window row* W*, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*. Consequently the Real-facing export contains no ambient completeness principle, quotient equality of real names, selected host endpoints, open-cover compactness, or countable-choice datum.*

Proof. The carrier definition places R after the finite interval-cover rows E, D, F, L, W, Q. There is no second Real seal coordinate and no bypass from a cover request directly to R.

The read of R is therefore mediated first by the interval source and dyadic cover rows, then by the located containment row, and then by the finite StreamName and regular rational readback rows. Componentwise transport and replay preserve this order through H and C.

The provenance and local naming rows P, N name the same packet. Because the listed exclusions are not carrier coordinates, they cannot escape through the terminal Real-facing seal. □

Falsifiable prediction: Every $\mathsf{BishopIntervalCover}^{\uparrow}$ Real-facing read passes through $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedIntervalCover}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any finite cover output is visible.

Independence witness: closedboundedinterval, dyadicintervalcover, dyadicratcore, located-interval-cover, streamname, regseqrat, real: $\mathsf{BishopIntervalCover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint rows, dyadic interval cells, a finite cover ledger, located containment witnesses, finite windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalCover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5124.1, Definition 5124.2, Definition 5124.3, and Definition 5124.4 seed the finite Bishop interval-cover route from closed bounded endpoint rows through dyadic cells, finite cover ledgers, located containment, finite windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalCover}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BishopIntervalCoverUp.BishopIntervalCoverTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, choice-selected subcovers, quotient real equality, selected host covers, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint-row exactness, dyadic-cell exactness, finite-cover ledger exposure, located-containment exactness, StreamName and RegSeqRat readback ordering, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalCoverUp.

Constructive story: A $\mathsf{BishopIntervalCover}^{\uparrow}$ packet is generated from ClosedBoundedIntervalUp endpoint rows, dyadic interval cells, a finite cover ledger, located containment witnesses, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ is read through chapter 5902, chapter 1998, and chapter 458 before the extraction row reaches any $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealSeparability}^{\uparrow}$ consumer.

Chapter 5125

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ packet records the finite extraction data used when a regular Cauchy source is read through subsequence windows before reaching a $\mathsf{Real}^{\uparrow}$ consumer. It sits after the Cauchy route of chapter 3413, the dyadic rational arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational sequence source of chapter 301, the terminal real-name seal of chapter 38, and the countable dense readback surface of chapter 1809. The extraction row E also consumes the minimum-modulus witness of chapter 5544 before $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealSeparability}^{\uparrow}$ consumers read the terminal seal. The packet records only displayed finite subsequence extraction data; it is not countable choice, quotient real equality, ambient completeness, or a selector for arbitrary Cauchy data.

Definition 5125.1 (CauchySubsequenceExtractor carrier)**.** *A* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (C, Q, W, D, E, R, L, H, T, P, N)$$

with the following displayed rows:

1. *C is the accepted* $\mathsf{Cauchy}^{\uparrow}$ *source row whose regularity ledger is being inspected.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the rational regular sequence extracted from C.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row listing the finite source indices, target indices, and requested precisions used by the extraction.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic bounds that certify the extracted window remains inside the requested Cauchy budget.*
5. *E is the extraction row. It records the monotone finite index map, the selected source observations, and the target-window schedule read from W.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after C, Q, W, D, E have been displayed.*
7. *L is the* $\mathsf{RealSeparability}^{\uparrow}$ *readback row recording the finite dense real-name inspection used by downstream separability consumers.*
8. *H records componentwise* hsame *transport for source, regular readback, stream windows, dyadic tolerances, extraction data, Real seal, separability readback, replay, provenance, and local naming.*

9. *T records displayed* Cont *replay from a finite subsequence request through the same $C, Q, W, D, E, R, L, H, P, N$ rows.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, Q, W, D, E, R, L, H, T, P, N$. It has no coordinate for arbitrary subsequence choice, quotient equality of real names, countable choice, a global selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5125.2 (CauchySubsequenceExtractor NameCert obligations)**.** *For every accepted* CauchySubsequenceE *carrier*

$$S = (C, Q, W, D, E, R, L, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtractor}^{\uparrow}}$ *surface consists exactly of Cauchy source exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger exposure, finite monotone extraction data, terminal* $\mathsf{Real}^{\uparrow}$ *sealing,* $\mathsf{RealSeparability}^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every consumer must expose the finite window W, dyadic ledger D, and extraction row E before it can read the Real seal R or the separability row L.*

Proof. Project the carrier $S = (C, Q, W, D, E, R, L, H, T, P, N)$. The analytic rows list the Cauchy source, regular rational readback, stream window, dyadic tolerance ledger, extraction row, Real seal, and separability readback.

The extraction row is downstream of the finite window and tolerance ledger. It can therefore name only the displayed monotone finite index map and selected observations, not an arbitrary subsequence selector.

The structural rows H, T, P, N transport, replay, source, and name the same finite route. Since no listed exclusion is a carrier coordinate, the NameCert surface is exhausted by the displayed packet. □

Theorem 5125.3 (CauchySubsequenceExtractor finite-window route)**.** *Every accepted* CauchySubsequenceExtracto *extraction read factors through*

$$\mathsf{Cauchy}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealSeparability}^{\uparrow}.$$

Thus the exported subsequence data are finite regular-Cauchy windows with dyadic tolerance control, not countable choice, quotient real equality, or a global subsequence functional.

Proof. Start from the source rows C and Q. They provide the Cauchy source and the regular rational readback being inspected.

The finite data are W, D, E. The window W supplies the displayed indices and precisions, the dyadic ledger D supplies the tolerance checks, and E records the monotone finite extraction map.

Finally read R, L, H, T, P, N. These rows seal the same finite extraction route for Real and separability consumers, transport it componentwise, replay it under displayed continuation, and name it locally. No row stores a choice principle or quotient equality witness. □

Theorem 5125.4 (CauchySubsequenceExtractor sibling route)**.** *Every accepted* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *route*

$$S = (C, Q, W, D, E, R, L, H, T, P, N)$$

that consumes the extraction row E preserves the sibling order

$$\mathsf{Cauchy}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealSeparability}^{\uparrow}.$$

Thus the route cannot export arbitrary subsequence choice, quotient real equality, global selectors, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (C, Q, W, D, E, R, L, H, T, P, N)$. The rows C and Q expose the Cauchy source and the regular rational readback before any extraction data are visible.

Apply the finite-window route of Definition 5125.3. The extraction row E is readable only after the displayed stream window W and dyadic tolerance ledger D have been read, so the route reaches $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealSeparability}^{\uparrow}$ through the listed sibling order.

The structural rows H, T, P, N preserve the same order by componentwise transport, displayed replay, provenance, and local naming. They do not add a selector coordinate, quotient real equality, countable choice, or ambient completeness, so the consumed extraction row remains finite L10 evidence for the displayed route. □

Theorem 5125.5 (CauchySubsequenceExtractor selector-lattice handoff)**.** *Every accepted* CauchySubsequenceExtr *read that exposes the monotone extraction row E must factor through the selector surface of chapter 5902 before* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RealSeparability}^{\uparrow}$ *consumers read the seal. For $S = (C, Q, W, D, E, R, L, H, T, P, N)$, the row E is therefore a finite selector-lattice handoff over the displayed window W and dyadic ledger D, not an arbitrary subsequence choice functional.*

Proof. Project the carrier $S = (C, Q, W, D, E, R, L, H, T, P, N)$. The extraction row E records a monotone finite index map, selected source observations, and a target-window schedule read from W.

Read the selector chapter named in the statement. Its surface supplies the monotone schedule and Cauchy-gap compatibility required before an extraction row can be treated as a selector-facing lattice row.

Finally apply the finite-window route of Definition 5125.3. The Real and separability rows R, L are reached only after W, D, E have been displayed and transported through H, T, P, N, so the handoff cannot export countable choice, quotient real equality, or a global subsequence selector. □

Theorem 5125.6 (CauchySubsequenceExtractor selector route)**.** *Every completion-facing* CauchySubsequenceExtr *read must pass through the* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, finite extraction row,* $\mathsf{Real}^{\uparrow}$ *seal, and* $\mathsf{RealSeparability}^{\uparrow}$ *readback before any completion witness consumes the packet. For*

$$S = (C, Q, W, D, E, R, L, H, T, P, N),$$

the route is therefore selector-scheduled finite extraction over displayed windows and tolerances, not arbitrary subsequence choice, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier and apply Definition 5125.5. The extraction row E is readable only as a finite selector-lattice handoff over the displayed window W and dyadic ledger D.

Use the finite-window route of Definition 5125.3 and the sibling route of Definition 5125.4. They force every read through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{RealSeparability}^{\uparrow}$ in the displayed sibling order.

Finally apply the completion witness route of Definition 5125.7. It keeps completion-facing consumers inside the same selector-scheduled extraction packet, so no refused choice, quotient, or ambient-completeness coordinate can be consumed downstream. □

Theorem 5125.7 (CauchySubsequenceExtractor completion witness route)**.** *Any accepted* CauchySubsequenceExtr *read exposing*

$$C, Q, W, D, E, R, L, H, T, P, N$$

may feed a $\mathsf{BishopCompletionWitness}^{\uparrow}$ *consumer only through the displayed* $\mathsf{Cauchy}^{\uparrow}$ *source* C*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, the finite extraction row* E*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R*. The route is the selector-lattice handoff of Definition 5125.5 followed by the finite-window route of Definition 5125.3; it exports no countable choice, quotient real equality, arbitrary selected subsequence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (C, Q, W, D, E, R, L, H, T, P, N)$. The rows C, Q, W, D, E, R are the only Cauchy source, regular readback, stream-window, dyadic-tolerance, extraction, and Real-seal coordinates available to a completion-facing consumer.

Apply the selector-lattice handoff of Definition 5125.5. It keeps E as a finite monotone selector-lattice row over the displayed W and D data, so the completion witness cannot receive an arbitrary subsequence selector.

Now use the finite-window route of Definition 5125.3. The same finite rows pass through $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the witness surface is readable.

The downstream regular-Cauchy cofinal-subsequence consumer of chapter 1998 and the BishopCompletionWitnes consumer of chapter 8181 consume the extraction row E only after W and D are visible, so the downstream completion route is finite-window evidence rather than countable choice.

The substantive sibling route also passes through the minimum-modulus witness of chapter 5544 and the cofinal-subsequence surface of chapter 1998. The same finite window W, dyadic ledger D, minimum-modulus witness, extraction row E, and terminal $\mathsf{Real}^{\uparrow}$ seal are shared by both consumers, so the extractor is a displayed horizontal dependency packet rather than a kernel-only float.

The structural rows L, H, T, P, N preserve the displayed packet for separability readback, transport, replay, provenance, and local naming. They do not add countable choice, quotient real equality, arbitrary subsequence selection, or ambient completeness, so the only completion-witness route is the one stated above. □

Falsifiable prediction: Every $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ consumer reaches $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealSeparability}^{\uparrow}$ only after exposing $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, finite extraction data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy, regseqrat, streamname, dyadicratcore, real, real-separability: $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source rows, regular readback, finite windows, dyadic tolerance ledgers, monotone extraction data, terminal Real sealing, separability readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtractor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ packet is generated from a $\mathsf{Cauchy}^{\uparrow}$ source row, $\mathsf{RegSeqRat}^{\uparrow}$ readback, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, monotone extraction data, a $\mathsf{Real}^{\uparrow}$ seal, $\mathsf{RealSeparability}^{\uparrow}$ readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5125.1, Definition 5125.2, and Definition 5125.3 seed the finite subsequence extraction route from Cauchy sources through regular rational readback, stream windows, dyadic tolerance ledgers, Real sealing, separability readback, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchySubsequenceExtractor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim countable choice, arbitrary subsequence selection, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window routing, dyadic tolerance exactness, monotone extraction exposure, Real seal nonescape, separability readback ordering, transport, replay, provenance, and local naming for BEDC.Derived.CauchySubsequenceExtractorUp.

The subsequence extractor also consumes the regular-Cauchy minimum-modulus packet of chapter 5544 when it selects a finite subsequence whose tail budget must remain visible. The extraction row E may therefore read the source window W and dyadic ledger D only through the same minimum-modulus witness surface before the terminal Real$^{\uparrow}$ or RealSeparability$^{\uparrow}$ rows are exposed. The extractor route also exports its finite monotone extraction row to the regular Cauchy cofinal-subsequence consumer of chapter 1998. Thus a RegularCauchyCofinalSubsequence$^{\uparrow}$ read may consume the displayed source window W, dyadic tolerance ledger D, and extraction row E only as the same finite extractor/cofinality dependency before the inherited Real$^{\uparrow}$ seal is exposed.

Theorem 5125.8 (Cauchy subsequence extractor minimum-modulus cofinal route)**.** *Every accepted* CauchySubsequenceExtractor$^{\uparrow}$ *route that reads the regular-Cauchy minimum-modulus witness before exposing the extraction row E also hands the same finite monotone extraction row to* RegularCauchyCofinalSubsequence$^{\uparrow}$ *before the inherited* Real$^{\uparrow}$ *seal is exposed. Thus the source window W, dyadic tolerance ledger D, minimum-modulus witness, extraction row E, and terminal real-name row remain one displayed dependency surface for both the minimum-modulus route and the cofinal-subsequence consumer.*

Proof. Start with the selector-lattice handoff of Definition 5125.5. It identifies E as the finite monotone selector-lattice row over the displayed stream window and tolerance ledger, so no consumer receives an arbitrary subsequence selector.

Use the completion-witness route of Definition 5125.7. The same row E is exposed only after the source Cauchy$^{\uparrow}$ data, RegSeqRat$^{\uparrow}$ readback, StreamName$^{\uparrow}$ window, DyadicRatCore$^{\uparrow}$ ledger, and terminal Real$^{\uparrow}$ seal have been kept visible.

Finally read the two local dependency routes. The minimum-modulus forward link requires the extraction row to consume the regular-Cauchy minimum-modulus witness before the terminal seal is available, while the cofinal-subsequence dependency route sends that same finite monotone row to RegularCauchyCofinalSubsequence$^{\uparrow}$ before the inherited Real$^{\uparrow}$ seal is exposed. Hence the minimum-modulus and cofinal reads compose through one displayed extractor route rather than through two detached host-side selections. □

Every CauchySubsequenceExtractor$^{\uparrow}$ read first consumes the selector schedule of chapter 5902. Only after that selector route is visible may the extractor read the regular rational sequence surface of chapter 301 and the finite window rows of chapter 126. Thus the extractor reaches any Real$^{\uparrow}$ or completion-facing consumer through selector, RegSeqRat, and StreamName rows rather than through a free subsequence choice, quotient stream equality, countable choice, or ambient completeness.

Theorem 5125.9 (CauchySubsequenceExtractor regular-Cauchy forward link)**.** *Every accepted* CauchySubsequenceExtractor$^{\uparrow}$ *packet consumes the* RegularCauchyMinModulus$^{\uparrow}$ *witness of chapter 5544, the* RegularCauchyCofinalSubsequence$^{\uparrow}$ *cofinal route of chapter 1998, and the selector*

dependency rows of chapter 5902 before exporting a subsequence witness. Thus the extraction row E is a finite regular-Cauchy forward link over the displayed window W and dyadic ledger D, not an independent subsequence selector.

Proof. Project the carrier $S = (C, Q, W, D, E, R, L, H, T, P, N)$ from Definition 5125.1. The extraction row E is downstream of the Cauchy source C, regular readback Q, finite window W, and dyadic tolerance ledger D.

Use the selector-lattice handoff of Definition 5125.5. It binds E to the displayed selector schedule before any $\mathsf{Real}^{\uparrow}$ or $\mathsf{RealSeparability}^{\uparrow}$ consumer can read the terminal rows.

Now attach the regular-Cauchy routes. The minimum-modulus forward link forces the same W, D, E rows to consume the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ witness, while the cofinal-subsequence dependency route sends the same extraction row to the $\mathsf{RegularCauchyCofinalSubsequence}^{\uparrow}$ consumer before the inherited real seal is exposed.

The structural rows H, T, P, N transport, replay, source, and name that single route. Since no carrier coordinate stores a free selector, countable choice, quotient stream equality, or ambient completeness, the exported subsequence witness is exactly the displayed regular-Cauchy forward link. □

Theorem 5125.10 (CauchySubsequenceExtractor tail-filter route)**.** *Every accepted* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *route*

$$S = (C, Q, W, D, E, R, L, H, T, P, N)$$

consumes the tail-filter or cofinal-subsequence surface before exporting the subsequence selector row E. In particular, a consumer that reads E must first expose either the Cauchy sequence tail-filter route of chapter 3850 or the cofinal-subsequence filter route of chapter 7421, together with the displayed source row C, regular readback Q, finite window W, and dyadic ledger D. The exported selector is therefore a tail/cofinal-filter read, not an isolated selector coordinate.

Proof. Start from the carrier rows of Definition 5125.1. The extraction row E is not available before the source row C, regular readback Q, finite window W, and dyadic tolerance ledger D have been displayed.

Use the selector-lattice handoff and finite-window route, Definition 5125.5 and Definition 5125.3. They bind the selector schedule to the same finite subsequence route that later feeds $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealSeparability}^{\uparrow}$ consumers.

Finally attach the tail-filter siblings. A tail-filter or cofinal-subsequence read supplies the late-window discipline that makes the selector row finite and monotone; the structural rows H, T, P, N then replay, transport, source, and name that same route. Since no carrier coordinate stores a free selector, countable choice, quotient stream equality, or ambient completeness, every exported selector remains downstream of the tail/cofinal filter surface.

The same route is the extractor-side input consumed by cluster-witness packets such as chapter 8396; the selector row can feed a regular sequence cluster witness only after the displayed tail/cofinal-filter discipline has already been read. □

Theorem 5125.11 (CauchySubsequenceExtractor regular-tail route)**.** *Every accepted* $\mathsf{CauchySubsequenceExtractor}$ *route*

$$S = (C, Q, W, D, E, R, L, H, T, P, N)$$

that exports the selector row E through a regular-tail consumer must first expose the sequence-generated tail-filter surface of chapter 3850 or the regular-tail selector surface of chapter 458. The selector row is therefore downstream of the regular tail discipline carried by $Q, W, D, R, L, H, T, P, N$; it is not a free choice schedule detached from the displayed tail and readback rows.

Proof. Project the carrier rows named in Definition 5125.10. The source row C, regular readback Q, finite window W, and dyadic ledger D are visible before E can be read.

Attach the regular-tail siblings next. The $\mathsf{CauchySequenceTailFilter}^{\uparrow}$ packet supplies tail filters generated by displayed sequence windows, while the RegularCauchyTailSelector packet supplies a finite selected-tail witness over the same regular readback route.

The structural rows R, L, H, T, P, N then replay, bind, transport, source, and name that route. Since no carrier coordinate stores an independent selector function, quotient stream equality, selected limit, countable choice schedule, or ambient completeness proof, the exported subsequence selector remains a downstream regular-tail read. □

Theorem 5125.12 (CauchySubsequenceExtractor regular-tail-filter sibling route)**.** *Every accepted* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *read*

$$S = (C, Q, W, D, E, R, L, H, T, P, N)$$

that exposes the subsequence extraction row E first exposes the regular Cauchy tail-filter route of chapter 1605. Thus a completion-facing, $\mathsf{Real}^{\uparrow}$*-facing, or separability-facing consumer may read E only after the displayed source row C, regular readback Q, finite window W, dyadic ledger D, selector-lattice handoff, finite-window route, and* $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ *sibling surface are visible.*

Proof. Project the carrier from Definition 5125.1. The extraction row E is downstream of the Cauchy source C, regular readback Q, finite window W, and dyadic tolerance ledger D.

Use the selector-lattice handoff of Definition 5125.5. It binds E to the displayed monotone selector schedule before any terminal $\mathsf{Real}^{\uparrow}$, $\mathsf{RealSeparability}^{\uparrow}$, or completion-facing row may read the extraction.

Apply the finite-window route of Definition 5125.3 and the tail-filter route of Definition 5125.10. These keep the selected observations inside the same finite window and dyadic ledger rather than letting the selector detach from its late-tail discipline.

Now read the sibling chapter chapter 1605. Its $\mathsf{RegularCauchyTailFilter}^{\uparrow}$ packet is the regular Cauchy tail surface that turns the visible late-window and tolerance rows into a cofinite tail-filter route. Therefore the extraction row is exposed only after that sibling route has been made available to the same C, Q, W, D data.

The structural rows H, T, P, N transport, replay, source, and name this single route. Since no carrier coordinate stores a free subsequence selector, quotient stream equality, countable choice, selected limit, or ambient completeness, every completion-facing or Real-facing read of E remains downstream of the regular Cauchy tail-filter sibling. □

The $\mathsf{ParallelDiamondResidual}^{\uparrow}$ hub routes residual local joins through chapter 4238 and the guarded candidate-normalization handoff of chapter 5547 before any confluence-facing endpoint is read.

Chapter 5126

A Concrete Naming Certificate for $\mathsf{ParallelDiamondResidual}^{\uparrow}$

The $\mathsf{ParallelDiamondResidual}^{\uparrow}$ packet realizes the MetaCIC confluence audit boundary of chapter 9378 as finite residual-join data. It sits after the residual-substitution boundary of chapter 2181, the residual-substitution compatibility packet of chapter 4370, and the local-join ledger of chapter 4126. The chapter records the concrete residual frontier that a later conditional parallel-diamond bridge may consume; it does not claim full Church–Rosser, closed arbitrary-term confluence, or unconditional substitution compatibility.

Definition 5126.1 (ParallelDiamondResidual carrier)**.** *A* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (P, F, S, L, B, C, O, H, T, G, N)$$

with the following displayed rows:

1. *P is the parallel peak row. It records a source MetaCIC term and the two parallel reducts whose residuals must be compared.*
2. *F is the residual frontier row. It records the finite redex-frontier cells, binder depths, and residual tags visible at the peak.*
3. *S is the* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *row naming the substitution-sensitive boundary consumed by the residual comparison.*
4. *L is the* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *row recording candidate common reducts for the displayed frontier cells.*
5. *B is the bounded checker row. It records the finite checks that the candidate joins respect the residual tags and binder depths displayed in F.*
6. *C is the* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *handoff row. It is a named compatibility interface, not a proof that the interface is globally available.*
7. *O is the obstruction row. It records any remaining residual-substitution cell that is not discharged by S, L, B, C.*
8. *H records componentwise* hsame *transport for peaks, residual frontiers, substitution boundaries, local joins, bounded checks, compatibility handoffs, obstruction rows, replay, provenance, and local naming.*

9. *T records displayed* Cont *replay for the same residual-join packet.*

10. *G is the* Pkg *provenance over* ParallelDiamondResidual$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICResidualSubstitutionCompatibility$^{\uparrow}$, MetaCICLocalJoinLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{ParallelDiamondResidual}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, F, S, L, B, C, O, H, T, G, N$. It has no coordinate for full Church–Rosser, arbitrary closed confluence, unconditional substitution compatibility, quotienting of terms, or a global parallel-diamond theorem.

Theorem 5126.2 (ParallelDiamondResidual NameCert obligations)**.** *For every accepted* ParallelDiamondResidual$^{\uparrow}$ *carrier*

$$J = (P, F, S, L, B, C, O, H, T, G, N),$$

the local NameCert$_{\text{ParallelDiamondResidual}^{\uparrow}}$ *surface consists exactly of parallel peak exposure, residual frontier exposure, residual-substitution boundary exposure, local-join ledger exposure, bounded residual checking, compatibility-interface handoff, obstruction-row exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may read the residual local-join packet only after the frontier F, boundary row S, ledger row L, and bounded checker B are displayed.*

Proof. Project the carrier $J = (P, F, S, L, B, C, O, H, T, G, N)$. The analytic rows are the peak, residual frontier, substitution boundary, local-join ledger, bounded checker, compatibility handoff, and obstruction row.

The compatibility row is downstream of the displayed frontier and checker. It names the interface consumed by the conditional parallel-diamond route, while the obstruction row keeps undiscovered residual-substitution cells visible.

The structural rows H, T, G, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for full Church–Rosser or an unconditional global diamond, the NameCert surface stops at residual local joins and their boundary rows. □

Theorem 5126.3 (ParallelDiamondResidual conditional handoff)**.** *Every* ParallelDiamondResidual$^{\uparrow}$ *residual-join read offered to the MetaCIC confluence route factors through*

$$\text{MetaCICResidualSubstitutionBoundary}^{\uparrow} \longmapsto \text{MetaCICLocalJoinLedger}^{\uparrow} \longmapsto \text{MetaCICResidualSubstitutionCompatibility}$$

The handoff supplies finite residual-frontier, local-join, bounded-checker, and obstruction data for the conditional parallel-diamond bridge; it does not assert the bridge, full Church–Rosser, or arbitrary closed-term confluence.

Proof. Start with the residual frontier F and the substitution boundary row S. These rows display the finite cells whose residual algebra must be inspected.

The local-join ledger L and bounded checker B then record candidate common reducts and the finite checks against the displayed frontier. The compatibility handoff C can only be read after these rows have been exposed.

Finally the obstruction row O and structural rows H, T, G, N preserve the same conditional boundary. They do not add a global confluence theorem, so the export remains a residual local-join packet for a later bridge. □

Theorem 5126.4 (ParallelDiamondResidual MetaCIC sibling route)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *read is a downstream residual-frontier consumer of chapter 4238. Such a read must preserve the sibling frontier's residual-substitution premise, critical-pair row, candidate-join row, bounded-checker row, compatibility-handoff row, and obstruction row before it may export a residual local-join packet.*

Proof. Project the carrier of Definition 5126.1. Its rows $P, F, S, L, B, C, O, H, T, G, N$ display the residual frontier, residual-substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, and structural certificate data.

Apply the NameCert obligations theorem Definition 5126.2. It requires any accepted read to expose F, S, L, B, C, O before the residual local-join packet can be consumed, so the sibling frontier's premise, critical-pair, candidate-join, checker, handoff, and obstruction rows cannot be dropped.

Finally use the conditional handoff theorem Definition 5126.3. The handoff reaches the MetaCIC confluence route only as a residual-frontier consumer, and the rows H, T, G, N preserve the same displayed boundary for chapter 4238. □

Theorem 5126.5 (ParallelDiamondResidual obstruction retention)**.** *While the obstruction row O remains displayed in an accepted* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *packet*

$$J = (P, F, S, L, B, C, O, H, T, G, N),$$

no consumer may read the packet as a premise-free parallel diamond, a full Church–Rosser theorem, an arbitrary closed-term confluence theorem, an unconditional substitution-compatibility theorem, a quotienting of terms, or a global parallel-diamond theorem.

Proof. Project the carrier of Definition 5126.1. The rows $P, F, S, L, B, C, O, H, T, G, N$ are displayed coordinates of the same accepted packet, and O is one of the confluence-facing rows rather than a surrounding note.

Apply the NameCert obligations theorem Definition 5126.2. It requires the residual frontier F, substitution boundary S, local-join ledger L, bounded checker B, compatibility handoff C, and obstruction row O to remain visible before the residual local-join packet can be consumed.

Finally use the conditional handoff theorem Definition 5126.3. Since that handoff supplies only finite residual-frontier, local-join, bounded-checker, compatibility, and obstruction data, any consumer that removes O and reads the packet as one of the refused endpoints has left the displayed $\mathsf{NameCert}_{\mathsf{ParallelDiamondResidual}^{\uparrow}}$ surface. □

Theorem 5126.6 (ParallelDiamondResidual frontier obstruction route)**.** *An accepted* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *packet*

$$J = (P, F, S, L, B, C, O, H, T, G, N)$$

can be consumed by $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *from chapter 4238 only while the obstruction row O, bounded checker B, residual frontier F, and substitution boundary S remain displayed. The frontier consumer receives a residual-frontier dependency packet over F, S, B, O, H, T, G, N, not a premise-free diamond, full Church–Rosser theorem, closed arbitrary-term confluence theorem, or discharged substitution boundary.*

Proof. Project the residual packet $J = (P, F, S, L, B, C, O, H, T, G, N)$. The rows F, S, B, O expose the residual frontier, substitution-sensitive boundary, bounded checker, and obstruction data that a frontier consumer must inspect.

Apply obstruction retention, Definition 5126.5. It forbids any consumer from reading the packet as a premise-free parallel diamond while O is still a displayed coordinate.

Now read the MetaCIC frontier chapter named in the statement. Its residual-join surface consumes finite frontier, checker, substitution-boundary, and obstruction rows before any candidate diamond read is public, so the same F, S, B, O rows remain part of the dependency rather than surrounding prose.

Finally transport and replay the packet through H, T, G, N. These structural rows preserve the displayed dependency for $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ and add no global confluence, quotienting, or substitution-discharge coordinate. □

Theorem 5126.7 (ParallelDiamondResidual seed obligation)**.** *Every seed-level* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *consumer must preserve the rows*

$$P, F, S, L, B, C, O, H, T, G, N$$

before any local join is exported. In particular, the residual frontier F, substitution boundary S, local join ledger L, bounded checker B, compatibility handoff C, and obstruction row O remain visible as finite obligations of the seed packet.

The seed obligation refuses premise-free parallel diamond, full Church–Rosser, arbitrary closed-term confluence, unconditional substitution compatibility, quotienting of terms, and a global parallel-diamond theorem.

Proof. Start with Definition 5126.2. It identifies $P, F, S, L, B, C, O, H, T, G, N$ as the displayed local NameCert surface and requires F, S, L, B, C, O before a residual local-join packet can be consumed.

Use Definition 5126.3. The handoff reaches the MetaCIC confluence route only through the residual-substitution boundary, local-join ledger, bounded checker, compatibility interface, and obstruction data.

Apply Definition 5126.4 and Definition 5126.6. The sibling and frontier consumers keep the residual frontier, substitution boundary, bounded checker, compatibility handoff, and obstruction rows visible before any local join is exported.

Finally invoke Definition 5126.5. Since O remains a displayed coordinate, the packet cannot be read as a premise-free parallel diamond, full Church–Rosser theorem, arbitrary closed-term confluence theorem, unconditional substitution-compatibility theorem, quotienting of terms, or global parallel-diamond theorem. □

Theorem 5126.8 (ParallelDiamondResidual normalization frontier handoff)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-join read handed to candidate-normalization or confluence consumers must first display the parallel peak row P, residual frontier F,* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *row S,* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *row L, bounded checker B,* $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ *handoff C, obstruction row O, and structural rows H, T, G, N. The downstream consumers include chapter 5547 and chapter 4238. The handoff refuses premise-free parallel diamond, full Church–Rosser, arbitrary closed-term confluence, unconditional substitution compatibility, quotienting of terms, and global conversion decidability.*

Proof. Start with the conditional handoff theorem, Definition 5126.3. It routes every confluence-facing read through the residual-substitution boundary, local-join ledger, bounded checker, compatibility interface, and obstruction data before any residual local join is exported.

Apply obstruction retention, Definition 5126.5. Since the row O remains displayed, the residual packet cannot be reread as a premise-free parallel diamond, full Church–Rosser theorem, arbitrary closed-term confluence theorem, unconditional substitution-compatibility theorem, quotienting of terms, or global conversion decidability.

Now use the frontier obstruction route, Definition 5126.6. It keeps F, S, B, O visible for the MetaCIC parallel-diamond frontier, and the same displayed rows feed the candidate-normalization confluence handoff named in the statement.

Finally transport and replay through H, T, G, N. These structural rows preserve the displayed peak, frontier, boundary, ledger, checker, compatibility, and obstruction packet for the downstream consumers without adding any global confluence or conversion-decidability coordinate. □

Falsifiable prediction: Every $\mathsf{ParallelDiamondResidual}^{\uparrow}$ handoff reaches $\mathsf{MetaCICResidualSubstitutionCompatibility}$ only through displayed parallel peaks, residual frontiers, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, bounded residual checks, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metacic-residual-substitution-boundary, metacic-residual-substitution-compatibility, metacic-local-join-ledger: $\mathsf{ParallelDiamondResidual}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries parallel peaks, residual frontiers, boundary rows, local-join ledgers, bounded checks, compatibility handoffs, obstruction rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{ParallelDiamondResidual}^{\uparrow}}$.

Constructive story: A $\mathsf{ParallelDiamondResidual}^{\uparrow}$ packet is generated from a parallel peak row, residual frontier, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ row, $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ row, bounded checker, $\mathsf{MetaCICResidualSubstitutionCompatibility}^{\uparrow}$ handoff, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ParallelDiamondResidual}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5126.1, Definition 5126.2, and Definition 5126.3 seed the finite residual local-join route for the conditional MetaCIC parallel-diamond boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ParallelDiamondResidual}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full Church–Rosser, arbitrary closed-term confluence, unconditional substitution compatibility, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-frontier exactness, substitution-boundary routing, local-join ledger exposure, bounded-checker exactness, compatibility handoff nonescape, obstruction-row exposure, transport, replay, provenance, and local naming for BEDC.Derived.ParallelDiamondResidualUp.

The dossier position of $\mathsf{ParallelDiamondResidual}^{\uparrow}$ is downstream of two MetaCIC sibling surfaces: it consumes the residual-join and critical-pair surface of chapter 4238 and the candidate-normalization handoff of chapter 5547 before exposing its own residual diamond rows. The exported packet therefore remains a finite residual-diamond dependency route, not an isolated kernel island, global confluence theorem, quotient equality, or host evaluator fact.

Theorem 5126.9 (ParallelDiamondResidual frontier dependency)**.** *Every accepted* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-join read consumes chapter 4238 before exporting any residual local-join packet. The read preserves the residual-substitution premise, critical-pair row, candidate-join row, bounded-checker row, compatibility-handoff row, and obstruction row through the same residual frontier dependency route.*

Proof. Project the carrier of Definition 5126.1. The rows $P, F, S, L, B, C, O, H, T, G, N$ display the peak, frontier, substitution boundary, local join, bounded checker, compatibility handoff, obstruction, and structural certificate data.

Apply the MetaCIC sibling route of Definition 5126.4. It makes the residual packet a downstream consumer of the frontier chapter before the local-join packet can be exported.

Use the conditional handoff theorem Definition 5126.3. The handoff preserves the residual-substitution premise, local-join ledger, bounded checker, compatibility interface, and obstruction

row, so the frontier dependency is a displayed packet route rather than an ambient confluence claim. □

The $\mathsf{ParallelDiamondResidual}^{\uparrow}$ packet is a residual-frontier consumer for the MetaCIC parallel-diamond boundary of chapter 4238. Any residual-join read from this hub must keep the MetaCIC frontier row upstream of the residual-substitution boundary, local-join ledger, compatibility handoff, and obstruction row; the dependency remains a finite confluence-facing route, not full Church–Rosser, subject reduction, quotient term equality, or an unconditional parallel-diamond theorem.

Proposition 5126.10 (ParallelDiamondResidual MetaCIC frontier sibling route)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-join consumer reads the local residual row through chapter 4238 and the residual-normalization route before exporting confluence data. The consumer must keep the residual-substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, transport, replay, provenance, and local* $\mathsf{NameCert}$ *rows visible while the MetaCIC frontier is read.*

Proof. Start with the local carrier of Definition 5126.1. It displays the residual frontier, substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, and structural certificate rows.

Read the sibling frontier through chapter 4238. That frontier supplies the confluence-facing residual-substitution and critical-pair context, but it does not discharge the local residual rows carried here.

Finally apply the residual-normalization route already recorded by the hub siblings. The residual row can be exported as confluence data only with the MetaCIC frontier and the local residual-normalization rows simultaneously displayed. □

Theorem 5126.11 (ParallelDiamondResidual MetaCIC parallel frontier sibling route)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-diamond read that consumes chapter 4238 must expose the frontier sibling's retained-premise rows and candidate-normalization rows before the residual packet is treated as a local join route. The read therefore preserves the residual frontier, substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, transport, replay, provenance, and local name.*

Proof. Start with the residual carrier of Definition 5126.1. Its rows $P, F, S, L, B, C, O, H, T, G, N$ display the peak, residual frontier, substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, and structural certificate data.

Read the sibling frontier of chapter 4238. Its retained-premise and candidate-normalization rows are the upstream data that make a residual-diamond read visible as a conditional MetaCIC frontier consumer.

Finally apply the existing sibling route Definition 5126.4 and the frontier obstruction route Definition 5126.6. Together they keep $F, S, L, B, C, O, H, T, G, N$ displayed before any local join route is exported, so the new sibling edge changes only the dependency lattice and not the residual theorem meaning. □

Theorem 5126.12 (ParallelDiamondResidual MetaCIC route)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-join packet is a downstream consumer of* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *from chapter 4238. Before any confluence-facing read is exported, the packet must preserve the residual-substitution boundary, local-join ledger, compatibility handoff, bounded checker, and obstruction rows displayed by the frontier route.*

Proof. Project the residual packet of Definition 5126.1. Its rows F, S, L, B, C, O expose the residual frontier, substitution boundary, local-join ledger, bounded checker, compatibility handoff, and obstruction boundary.

Apply the MetaCIC sibling route of Definition 5126.4. It makes the residual packet a downstream consumer of the frontier chapter and keeps the frontier premise, critical-pair row, candidate-join row, checker row, handoff row, and obstruction row visible.

Finally use the frontier obstruction route of Definition 5126.6. Since the frontier consumer receives only the displayed dependency packet over the residual rows and structural transports, no confluence-facing read can bypass the listed boundary rows. □

Theorem 5126.13 (ParallelDiamondResidual MetaCIC frontier dependency)**.** *Every* $\mathsf{ParallelDiamondResidual}^{\uparrow}$ *residual-frontier consumer must cite chapter 4238 before any confluence-facing read is exported. The consumer preserves the residual-substitution boundary, local-join ledger, compatibility handoff, bounded checker, and obstruction row of the residual packet before it may read the packet as confluence-facing data.*

Proof. Project the carrier of Definition 5126.1. The rows $P, F, S, L, B, C, O, H, T, G, N$ expose the residual frontier, residual-substitution boundary, local-join ledger, bounded checker, compatibility handoff, obstruction row, and structural transport data, so the cited MetaCIC frontier remains part of the displayed packet route.

Apply the sibling route Definition 5126.4 and the frontier obstruction route Definition 5126.6. They keep the residual-substitution boundary, local-join ledger, bounded checker, compatibility handoff, and obstruction row visible before a confluence-facing consumer can read the packet, and the structural rows preserve that boundary without adding full Church–Rosser, quotient term equality, or an unconditional parallel-diamond theorem. □

Theorem 5126.14 (ParallelDiamondResidual residual-join consumer route)**.** *Every accepted* ParallelDiamondResid *local-join row used by the MetaCIC residual audit is consumed by chapter 8134 and chapter 7258 before any confluence handoff is read. The route exposes residual branch rows, candidate joins, retained premise data, obstruction rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming, and it does not export a premise-free parallel diamond or closed Church–Rosser theorem.*

Proof. Project the residual local-join carrier through the sibling routes already named by the chapter. The branch rows and candidate joins are retained as finite BHist coordinates, while the transport, replay, provenance, and naming rows remain local to the $\mathsf{ParallelDiamondResidual}^{\uparrow}$ packet.

The matrix consumer of chapter 8134 reads two residual branches together with retained residual-substitution premise data. The grid consumer of chapter 7258 reads the same residual handoff as a finite residual-vertex and candidate-join grid.

Thus the forward route is a consumer dependency for the residual audit surface. Since neither consumer receives a coordinate discharging the retained premises or the obstruction row, the route cannot be retyped as a premise-free confluence theorem. □

Chapter 5127

A Concrete Naming Certificate for $\mathsf{GramDeterminant}^{\uparrow}$

Following Gram and classical finite-dimensional inner-product theory, the $\mathsf{GramDeterminant}^{\uparrow}$ packet records the finite determinant surface obtained from displayed inner products of a finite vector family. It consumes the inner-product rows of chapter 86, the matrix-entry carrier of chapter 49, the determinant boundary of chapter 84, and the finite coordinate discipline of chapter 4758. The packet names the finite Gram matrix, its determinant row, and its principal-minor ledger; it is not a Hilbert-space compactness theorem, a selected orthonormal basis, a quotient-vector equality principle, or a positive-definiteness theorem outside the displayed rows.

Definition 5127.1 (Gram determinant carrier)**.** *A* $\mathsf{GramDeterminant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (V, I, A, M, D, J, T, R, H, C, P, N)$$

with the following displayed coordinates:

1. *V is the finite vector-row bundle whose entries are the carried vectors $v_1, \ldots, v_n$;*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same vector endpoints, exposing the scalar reads $\langle v_i, v_j \rangle$ for every displayed pair;*
3. *A is the scalar-entry ledger that records each inner-product value as a finite BEDC scalar history;*
4. *M is the* $\mathsf{Mat}^{\uparrow}$ *row whose (i, j) entry is read from the scalar row $\langle v_i, v_j \rangle$;*
5. *D is the* $\mathsf{Determinant}^{\uparrow}$ *row naming the determinant of the displayed matrix row M;*
6. *J is the finite principal-minor ledger, one determinant row for each displayed principal submatrix of M;*
7. *T stores transport data for reindexing the finite vector rows, scalar inner-product rows, matrix-entry rows, determinant row, and principal-minor rows;*
8. *R stores replay data from vector rows to scalar entries, from scalar entries to matrix rows, and from matrix rows to determinant and principal-minor rows;*
9. *H records componentwise* hsame *transport for $V, I, A, M, D, J, T, R, C, P, N$;*
10. *C records the displayed* Cont *routes by which determinant and principal-minor consumers read the packet;*

11. *P is the* Pkg *provenance over* $\mathsf{GramDeterminant}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Determinant}^{\uparrow}$, *finite vector bundles,* BHist, hsame, Cont, *and* NameCert*;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{GramDeterminant}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the finite vector rows, scalar inner-product rows, matrix-entry rows, determinant rows, principal-minor rows, transport, replay, provenance, continuation, and local naming data.

Lean4 ✓ `BEDC.Derived.GramDeterminantUp.GramDeterminantTasteGate single carrier alignment`

Theorem 5127.2 (Gram determinant NameCert obligations)**.** *For the carrier of Definition 5127.1, the local* $\mathsf{NameCert}_{\mathsf{GramDeterminant}^{\uparrow}}$ *surface consists exactly of five finite obligations: carrier admission for $G = (V, I, A, M, D, J, T, R, H, C, P, N)$; scalar-entry exactness requiring every matrix entry in M to be read from the corresponding* $\mathsf{InnerProduct}^{\uparrow}$ *scalar row in A; determinant exactness requiring D to consume only the displayed* $\mathsf{Mat}^{\uparrow}$ *row M; principal-minor exactness requiring each row of J to consume a displayed principal submatrix of M; and transport-replay stability for T, R, H, C, P, N under the same finite row inventory.*

Proof. Carrier admission is projection from Definition 5127.1. The accepted packet lists vector rows, inner-product scalar rows, matrix-entry rows, determinant rows, principal-minor rows, transport, replay, componentwise hsame, continuation, provenance, and local naming.

The scalar-entry and determinant clauses are read in order. First I and A expose the finite family of scalar inner-products. Then M records those scalars as matrix entries, and D reads only that displayed matrix row through the determinant certificate of chapter 84.

The principal-minor clauses are finite subpacket reads of the same matrix row. Each minor in J is indexed by displayed rows of V and by the matching submatrix entries in M, so no ambient matrix quotient or host positivity theorem is imported.

Finally T, R, H, C, P, N preserve and replay the same inventory. Since the carrier has no coordinate for a selected orthonormal basis, quotient-vector equality, Hilbert completion, or external positive-definiteness principle, the five obligations exhaust the local certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{GramDeterminant}^{\uparrow}}$.

Constructive story: A $\mathsf{GramDeterminant}^{\uparrow}$ packet is generated from a finite vector-row bundle, inner-product scalar rows, a matrix-entry row, determinant and principal-minor rows, transport, replay, componentwise hsame, Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GramDeterminant}^{\uparrow}}$, seed)

Scope closed: Definition 5127.1 and Definition 5127.2 declare the finite carrier and NameCert obligation surface for Gram determinant packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GramDeterminant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim positive semidefiniteness, Gram determinant vanishing criteria, quotient-vector equality, selected orthonormal bases, Hilbert completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked scalar-entry exactness, determinant exactness, principal-minor exactness, transport, replay, provenance, and local naming for the same finite packet.

Chapter 5128

A Concrete Naming Certificate for $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$

Following Bishop and Bridges constructive metric analysis, the $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$ packet records the finite BEDC surface by which an explicit dyadic tolerance request for a continuous map is answered by a displayed source-radius schedule before any compactness, completion, or $\mathsf{Real}^{\uparrow}$ consumer reads the result. It sits next to the pointwise map surface of chapter 130, the source and target metric rows of chapter 4230, the finite compact-metric coverage route of chapter 129, the terminal real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The certificate is a displayed modulus schedule for downstream compact-metric uniform-continuity and Heine–Cantor consumers; it is not open-cover compactness, a selected finite subcover, a quotient real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5128.1 (Modulus of uniform continuity carrier)**.** *A* $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, F, D, S, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, naming the source distance ledger, source-point comparison interface, and source-radius requests.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row, naming the target distance ledger and target-radius comparison interface consumed by the map row.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row from X to Y, carrying the pointwise continuity route and the displayed source-to-target graph read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request ledger, recording the target dyadic tolerance and the finite dyadic comparison rows used to ask for a uniform answer.*
5. *S is the modulus schedule row, assigning the displayed dyadic tolerance in D to a source-radius row over X and recording the finite lookup by which the same schedule is replayed for every accepted source pair.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary, reached only after X, Y, F, D, S have been displayed and only as the terminal real-name readback for the target estimate.*

7. *H records componentwise* hsame *transport for the source metric, target metric, continuous-map row, dyadic tolerance request, modulus schedule, Real seal boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a dyadic tolerance request through D, then through S, into the source metric row X, the map row F, the target metric row Y, and finally the Real seal row E.*
9. *P is the* Pkg *provenance over* $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfUniformContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows, their componentwise transports, their replay route, their provenance, and their local naming row. It has no coordinate for open-cover compactness, selected subcovers, quotient real equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusOfUniformContinuityUp.ModulusOfUniformContinuityTasteGate single carrier alignment`

Theorem 5128.2 (Modulus of uniform continuity NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$ *carrier $M = (X, Y, F, D, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfUniformContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source* $\mathsf{Met}^{\uparrow}$ *row, target* $\mathsf{Met}^{\uparrow}$ *row,* $\mathsf{ContinuousMap}^{\uparrow}$ *row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance request, modulus schedule,* $\mathsf{Real}^{\uparrow}$ *seal boundary, transport, replay, provenance, and local naming row;*
2. *tolerance-source exactness, requiring every requested target tolerance to be read from the displayed dyadic ledger D before the schedule row S is inspected;*
3. *source-radius schedule exactness, requiring S to return only a source-radius row over X and to reuse that row for each accepted source-pair read of the same tolerance;*
4. *continuous-map handoff, requiring the source-radius evidence to pass through the pointwise* $\mathsf{ContinuousMap}^{\uparrow}$ *route F before the target* $\mathsf{Met}^{\uparrow}$ *row Y and the Real seal row E are exposed;*
5. *compact-metric compatibility, requiring any downstream* $\mathsf{CompactMetric}^{\uparrow}$ *consumer to read finite-net or compact-metric data only as an external consumer of the displayed schedule, not as a hidden source of S;*
6. *transport and replay stability, requiring componentwise* hsame *transport in H and displayed* Cont *replay in C to preserve the ordered route D, S, X, F, Y, E;*
7. *provenance exhaustion, requiring P and N to name only the displayed rows and to export no open-cover compactness, selected subcover, quotient real equality, selected limit, countable choice, trichotomy, or ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Project the accepted carrier. Its analytic rows are the source metric X, target metric Y, continuous-map route F, dyadic tolerance ledger D, modulus schedule S, and Real seal boundary E; the remaining rows transport, replay, source, and name that same packet.

Read the tolerance route in order. A target-radius request first enters the dyadic ledger D, then the modulus schedule S, and only then the source metric row X where the displayed source radius is used.

The map and target estimates are downstream of that schedule. The continuous-map row F consumes the source-radius evidence before the target metric row Y and the terminal $\mathsf{Real}^{\uparrow}$ boundary E can be inspected.

Compact-metric data is therefore only a consumer-side row. A $\mathsf{CompactMetric}^{\uparrow}$ finite-net argument may cite the displayed schedule, but it cannot supply an unrecorded open cover, selected subcover, or choice principle inside the $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$ carrier.

Finally, componentwise transport through H, replay through C, provenance through P, and local naming through N preserve the same ordered rows. Since the excluded quotient, selection, trichotomy, and completeness objects are absent from the carrier inventory, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfUniformContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfUniformContinuity}^{\uparrow}$ packet is generated from source and target $\mathsf{Met}^{\uparrow}$ rows, a $\mathsf{ContinuousMap}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance requests, a displayed source-radius modulus schedule, a terminal $\mathsf{Real}^{\uparrow}$ seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5128.1 and Definition 5128.2 seed the finite displayed modulus-of-uniform-continuity dyadic tolerance-to-source-radius schedule packet over $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows for continuous-map and compact-metric uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusOfUniformContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, quotient real equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-source exactness, source-radius schedule exactness, continuous-map handoff, compact-metric compatibility, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfUniformContinuityUp and the same finite packet.

Chapter 5129

A Concrete Naming Certificate for $\mathsf{ModulusOfContinuity}^{\uparrow}$

Following Bishop and Bridges on constructive continuity, the $\mathsf{ModulusOfContinuity}^{\uparrow}$ packet records the finite data by which a pointwise continuity certificate exposes tolerance-to-precision information before Real, Completion, compactness, or uniform-continuity consumers inspect it. It sits between the source metric carrier of chapter 57, the real-name seal of chapter 38, the continuous-map graph of chapter 130, and the uniform-modulus handoff of chapter 369. The packet names the displayed modulus ledger itself; it is not a host theorem asserting continuity, a chosen global modulus, quotient equality of functions, or an ambient real-completeness principle.

Definition 5129.1 (ModulusOfContinuityUp carrier)**.** *A* $\mathsf{ModulusOfContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, Y, T, P, L, U, H, C, V, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row, including the distance request surface and the source-point histories that the modulus may inspect.*
2. *Y is the* $\mathsf{Real}^{\uparrow}$ *target carrier or target metric row through which output tolerances are read.*
3. *T is the tolerance history, recording the requested positive rational or dyadic output tolerance.*
4. *P is the precision history, recording the source-radius response attached to the same request.*
5. *L is the pointwise modulus ledger, assigning the displayed precision row P to the displayed tolerance row T at the named source point.*
6. *U is an optional uniform ledger over a finite source packet, present exactly when the same precision row is licensed without rereading a source point.*
7. *H records componentwise* hsame *transport for the metric, real, tolerance, precision, pointwise, optional uniform, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through T, P, L and, when available, through U.*
9. *V is the* Pkg *provenance over* $\mathsf{ModulusOfContinuity}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfContinuity}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host continuity theorem, a choice-selected modulus, quotient equality of functions, real trichotomy, countable choice, a completed function space, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5129.2 (ModulusOfContinuityUp NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfContinuity}^{\uparrow}$ *carrier*

$$M = (X, Y, T, P, L, U, H, C, V, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusOfContinuity}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source metric, target real or target metric, tolerance, precision, pointwise ledger, optional uniform ledger, transport, replay, provenance, and local naming rows;*
2. *tolerance admission, requiring every output request to be read through* T *before a precision response is visible;*
3. *pointwise precision admission, requiring* P *and* L *to record the source-radius response attached to the same tolerance request and source history;*
4. *optional uniform admission, requiring* U *to be displayed before any consumer treats the precision row as source-point independent;*
5. *continuous-map replay, requiring* C *to factor the displayed request through the* Cont *route licensed by the continuous-map surface;*
6. *transport stability, requiring* H *to preserve the source, target, tolerance, precision, ledger, replay, provenance, and local naming rows componentwise;*
7. *nonescape from host continuity theorems, choice-selected moduli, quotient function equality, real trichotomy, countable choice, completed function spaces, and ambient real completeness.*

No $\mathsf{ModulusOfContinuity}^{\uparrow}$ *naming obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5129.1. The accepted packet displays exactly the source metric row, target row, tolerance history, precision history, pointwise ledger, optional uniform ledger, and the structural rows H, C, V, N.

Tolerance and pointwise precision admission follow from the displayed request order. A consumer first reads the requested output tolerance in T, then reads the source-radius response in P, and then checks that L attaches that response to the same source history.

If a uniform read is requested, inspect U. The carrier permits a source-point independent precision only when this row is present, so a compact-uniform-continuity consumer cannot infer uniformity from the pointwise ledger alone.

Replay and transport are structural. The Cont row C repeats the continuous-map request through the same tolerance and precision rows, while H preserves each displayed coordinate componentwise.

Finally inspect V and N. Provenance and local naming mention only the displayed rows, so the excluded host theorem, selected modulus, quotient equality, trichotomy, choice, completed function-space, and ambient-completeness data have no projection from the packet. □

Theorem 5129.3 (ModulusOfContinuityUp uniformization handoff)**.** *Let*

$$M = (X, Y, T, P, L, U, H, C, V, N)$$

be an accepted $\mathsf{ModulusOfContinuity}^{\uparrow}$ *carrier. Any compact-uniform-continuity, completion, or continuous-map consumer that reads a uniform precision from* M *must first read the source metric*

row X, the target row Y, the tolerance history T, the precision history P, the pointwise ledger L, and the optional uniform ledger U when source-point independence is claimed.

Thus the handoff from pointwise continuity data to a $\mathsf{UniformModulus}^{\uparrow}$*-facing ledger factors through finite BEDC rows over* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*. It does not export a host-selected modulus, function extensionality, quotient equality of functions, open-cover compactness, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5129.1. The only rows that can answer a tolerance request are T, P, L and, for a source-point independent read, U.

Apply Definition 5129.2. The NameCert surface forces every read to pass through the displayed source metric and target rows, then through the tolerance and precision histories, and then through the pointwise ledger before any optional uniform row can be consumed.

Now compare the sibling interfaces. The source distance and point histories are supplied by chapter 57, target real-name reads by chapter 38, continuous replay by chapter 130, and the source-point independent handoff by chapter 369.

The structural rows H, C, V, N preserve, replay, source, and name exactly that route. Since the excluded host-continuity, selected-modulus, quotient, open-cover, choice, and completeness objects are absent from the carrier, a downstream consumer cannot recover them while claiming to read the same $\mathsf{ModulusOfContinuity}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfContinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{ModulusOfContinuity}^{\uparrow}$ packet is generated from source MetricUp rows, target RealUp or target metric rows, tolerance histories, precision histories, pointwise modulus ledgers, optional uniform ledgers, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ModulusOfContinuity}^{\uparrow}}$, seed)

Scope closed: Definition 5129.1, Definition 5129.2, and Definition 5129.3 seed the finite pointwise-modulus route from metric and real rows through continuous replay into optional uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ModulusOfContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, a host-selected modulus, quotient equality of functions, open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, pointwise ledger exactness, optional uniform-ledger handoff, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfContinuityUp.

Chapter 5130

A Concrete Naming Certificate for $\mathsf{DirichletBoxApproximation}^{\uparrow}$

Following Dirichlet's approximation theorem and Bishop–Bridges constructive analysis, the DirichletBoxApproximati packet names the finite pigeonhole route from a rational source row and a denominator box ledger to a rational approximation surface. It sits after the rational source of chapter 37, the adjacent rational routes of chapter 1759 and chapter 1704, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet records only finite box partitions, displayed adjacent-rational handoffs, tolerance rows, readback windows, transport, replay, provenance, and local naming.

Definition 5130.1 (Dirichlet box approximation carrier)**.** *A* $\mathsf{DirichletBoxApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, L, F, A, D, W, G, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{Rat}^{\uparrow}$ *source row. It records the rational input data, finite numerator observations, and rational endpoints inspected by the approximation request.*
2. *L is the denominator-box ledger. It stores the positive finite denominator bound, the box count, the inspected residue slots, and the finite comparison rows used to expose two entries in the same box.*
3. *F is the finite box-partition row. It records the displayed partition of the unit interval or rational bracket into denominator-controlled cells.*
4. *A is the adjacent-rational handoff row. It sends the same-pigeonhole pair to the* $\mathsf{FareySequence}^{\uparrow}$ *or* $\mathsf{SternBrocot}^{\uparrow}$ *adjacency surface before any approximant is exported.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the dyadic error request, slack ledger, and endpoint comparisons used to validate the rational difference produced by L, F, A.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It schedules the finite observations of Q, L, F, A, D that downstream real-facing requests may inspect.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It repackages the displayed windows and rational differences as regular rational data before the terminal real seal is available.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is reachable only after the rational source, box ledger, partition, adjacency handoff, dyadic tolerance, finite windows, and regular readback rows have been displayed.*
9. *H records componentwise* hsame *transport for rational sources, denominator boxes, partitions, adjacency handoffs, dyadic tolerances, finite windows, regular readback, Real seals, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a box-approximation request through $Q, L, F, A, D, W, G, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{DirichletBoxApproximation}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DirichletBoxApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, L, F, A, D, W, G, E, H, C, P, N$. It has no coordinate for selected best approximants, quotient real equality, trichotomy, countable choice, a host pigeonhole selector outside the finite ledger, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DirichletBoxApproximationUp.DirichletBoxApproximationTasteGate single carrier alignment`

Theorem 5130.2 (Dirichlet box approximation NameCert obligations)**.** *For every accepted* DirichletBoxApproximat *carrier*

$$B = (Q, L, F, A, D, W, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DirichletBoxApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the rational source, denominator-box ledger, finite partition, adjacency handoff, dyadic tolerance, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *box partition exposure, requiring L and F to display the finite denominator-controlled cells and the same-box pair before an approximating rational difference can be exported;*
3. *adjacency transport, requiring A to route the displayed same-box pair through the* $\mathsf{FareySequence}^{\uparrow}$ *or* $\mathsf{SternBrocot}^{\uparrow}$ *sibling surface;*
4. *dyadic-error control, requiring D to record the requested dyadic tolerance, slack, and endpoint comparisons used by the approximation read;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *nonescape, requiring W, G, E to stay downstream of the finite box and adjacency rows;*
6. *local naming, requiring H, C, P, N to preserve and source exactly the displayed packet.*

No Dirichlet-box obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5130.1. The accepted packet displays exactly the rational source, denominator-box ledger, finite partition, adjacency handoff, dyadic tolerance, finite windows, regular readback, terminal Real seal, and structural rows listed in the statement.

The box partition exposure is finite. The row L stores the denominator bound, box count, residue slots, and same-box comparison, while F records the displayed partition cells. Together they supply the pigeonhole surface without invoking a host selector beyond the stored finite ledger.

Adjacency transport is read next. The row A sends the same-box pair to the Farey and Stern–Brocot siblings of chapter 1759 and chapter 1704, so the exported rational difference is attached to displayed adjacent-rational data.

The dyadic and real-facing rows remain downstream. The tolerance row D supplies the error and slack comparisons, the window row W schedules only finite observations, and G reads those windows as $\mathsf{RegSeqRat}^{\uparrow}$ data before E seals the route as a $\mathsf{Real}^{\uparrow}$-facing packet.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier contains no coordinate for selected best approximants, quotient real equality, trichotomy, countable choice, a host pigeonhole selector outside the finite ledger, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5130.3 (Dirichlet box approximation finite pigeonhole route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{DirichletBoxApproximation}^{\uparrow}$ *carrier*

$$B = (Q, L, F, A, D, W, G, E, H, C, P, N)$$

factors through the finite route

$$Q, L, F, A, D, W \longmapsto G, E.$$

Thus the exported approximation is witnessed by rational source data, a denominator-box ledger, a finite partition row, an adjacent-rational handoff, a dyadic tolerance row, and a finite StreamName window, not by quotient real equality, trichotomy, countable choice, selected best approximants, or ambient Real completeness.

Proof. Start with a regular-rational or Real-facing read from the accepted carrier. The only rows with that target are G and E, and the carrier places them after Q, L, F, A, D, W.

The source and box rows determine the finite pigeonhole data. The row Q supplies the rational observations, while L and F expose the denominator bound, partition cells, and same-box comparison needed for the approximating rational difference.

The adjacency and tolerance rows then control the exported approximation. The row A records the Farey or Stern–Brocot handoff, and D records the dyadic error and slack comparisons attached to that handoff.

Only after the finite window row W is fixed can G read the packet as regular rational data and E seal it as Real-facing. Transport, replay, provenance, and local naming preserve this route through H, C, P, N, so none of the excluded ambient principles can enter the exported read. □

Falsifiable prediction: Every $\mathsf{DirichletBoxApproximation}^{\uparrow}$ Real-facing read passes through $\mathsf{Rat}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$ or $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any approximation output is visible.

Independence witness: rat, farey-sequence, stern-brocot, dyadicratcore, streamname, regseqrat, real: $\mathsf{DirichletBoxApproximation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a rational source row, denominator-box ledger, finite partition, adjacent-rational handoff, dyadic tolerance row, finite windows, regular readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DirichletBoxApproximation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DirichletBoxApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5130.1, Definition 5130.2, and Definition 5130.3 seed the finite pigeonhole route from rational source rows through denominator boxes, finite partitions, adjacency handoffs, dyadic tolerance, finite windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DirichletBoxApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected best approximants, quotient real equality, trichotomy, countable choice, host pigeonhole selectors outside the displayed finite ledger, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, box partition exposure, adjacency transport, dyadic-error control, StreamName and RegSeqRat readback ordering, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DirichletBoxApproximationUp.

Constructive story: A DirichletBoxApproximation$^{\uparrow}$ packet is generated from RatUp source rows, a denominator-box ledger, a finite partition row, a FareySequenceUp or SternBrocotUp adjacency handoff, a DyadicRatCoreUp tolerance row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5131

A Concrete Naming Certificate for $\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}$

Following Riemann, Stieltjes, and Bishop–Bridges, the $\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}$ packet records integration by parts as finite NameCert data over a regulated integrand and a bounded-variation integrator. It consumes the Riemann–Stieltjes surface of chapter 1637, the variation ledger of chapter 398, the regulated-function schedule of chapter 1906, the tagged finite-sum row of chapter 1615, the dyadic step approximation row of chapter 428, and the terminal real seal of chapter 38. The packet names only displayed endpoint rows, finite partition rows, transported jump rows, two Riemann–Stieltjes handoff rows, and the boundary-term ledger; it does not add measure integration, quotient functions, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5131.1 (Riemann–Stieltjes integration-by-parts carrier)**.** *A* $\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, G, V, R, S, D, B, E, H, C, P, N)$$

with displayed rows: F *is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *integrand row;* G *is the regulated integrator row;* V *is the* $\mathsf{BoundedVariation}^{\uparrow}$ *ledger for* G*;* R *is the forward* $\mathsf{RiemannStieltjes}^{\uparrow}$ *handoff for* $\int F\,dG$*;* S *is the reverse handoff for* $\int G\,dF$ *when the displayed variation rows license it;* D *is the common* $\mathsf{RiemannSum}^{\uparrow}$ *partition ledger;* B *is the endpoint boundary row* $F(b)G(b) - F(a)G(a)$*;* E *is the* $\mathsf{Real}^{\uparrow}$ *seal of the accepted identity; and* H, C, P, N *are the componentwise* hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}}$ *rows.*

Theorem 5131.2 (Riemann–Stieltjes integration-by-parts NameCert obligations)**.** *For every accepted carrier* $I = (F, G, V, R, S, D, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}}$ *surface consists exactly of carrier admission; regulated-function exposure for* F, G*; bounded-variation control through* V*; common tagged-partition exposure through* D*; forward and reverse Riemann–Stieltjes handoffs through* R, S*; endpoint-boundary exactness through* B, E*; componentwise transport, replay, provenance, and local naming through* H, C, P, N*; and non-escape from measure spaces, quotient functions, selected limits, and ambient completeness.*

Lean4 ✓ `BEDC.Derived.RiemannStieltjesIntegrationPartsUp.RiemannStieltjesIntegrationPartsTasteGate`
`single carrier alignment`

Proof. Project the displayed carrier. The rows F, G, V, D expose the regulated and bounded-variation data before either integration row is read.

The rows R and S are then checked over the same finite partition ledger. Since both handoffs use the same D and the same endpoint transports, their finite summand ledgers can be compared without introducing a host limit.

Finally B records the two endpoint products and E seals the displayed equality. Transport and replay through H, C, P, N preserve the same rows, so no hidden measure, function quotient, choice of representatives, selected limit, or ambient completeness coordinate remains available. □

Theorem 5131.3 (Riemann–Stieltjes integration-by-parts boundary)**.** *Every accepted* RiemannStieltjesIntegrationP *packet exposes the integration-by-parts boundary only through the finite route*

$$F, G, V, D \longmapsto R, S \longmapsto B \longmapsto E.$$

Thus a consumer may read the boundary term $F(b)G(b)-F(a)G(a)$ only after the regulated-function rows, bounded-variation ledger, common Riemann-sum partition row, and both Riemann–Stieltjes handoff rows have been displayed.

Proof. Start with a boundary-facing consumer. The carrier has no boundary row except B, and B is downstream of the endpoint reads supplied by the regulated rows F, G.

The common finite partition row D aligns the two integration handoffs. Reading R and S over D accounts for the forward and reverse summand ledgers before the endpoint product row is exposed.

The real seal E is therefore read only after B. The structural rows H, C, P, N transport, replay, source, and name the same route, so the packet exports the integration-by-parts identity as finite NameCert data rather than as a host theorem about completed integrals. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}$ packet is generated from regulated-function rows, a bounded-variation ledger, common Riemann-sum partition data, two Riemann–Stieltjes handoffs, endpoint boundary rows, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5131.1, Definition 5131.2, and Definition 5131.3 seed the finite integration-by-parts packet over displayed Riemann–Stieltjes, bounded-variation, regulated-function, Riemann-sum, dyadic-step, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannStieltjesIntegrationParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic integration, quotient function equality, selected limits, global variation measures, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, common-partition stability, two handoff rows, endpoint-boundary exactness, transport, replay, provenance, and local naming for the same packet.

$\mathsf{BishopBarInductionModulus}^{\uparrow}$ is consumed by the Real-facing equality-modulus lattice through chapter 7601, and it depends on the finite fan-depth route of chapter 400 before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or terminal $\mathsf{Real}^{\uparrow}$ read.

Chapter 5132

A Concrete Naming Certificate for $\mathsf{BishopBarInductionModulus}^{\uparrow}$

Following Brouwer's barred-spread analysis, Troelstra–van Dalen's treatment of bar induction, and Bishop–Bridges constructive Real practice, the $\mathsf{BishopBarInductionModulus}^{\uparrow}$ packet records the finite modulus surface that Real and Completion consumers may read from a barred spread. It sits after the barred-spread packet of chapter 1675, the finite fan surface of chapter 400, the finite stream-window discipline of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed modulus handoff; it is not countable choice, open-cover compactness, quotient stream equality, selected branch data, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5132.1 (Bishop bar induction modulus carrier)**.** *A* $\mathsf{BishopBarInductionModulus}^{\uparrow}$ *carrier is a finite barred-spread modulus packet*

$$K = (B, T, I, W, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the barred-spread row inherited from* $\mathsf{BrouwerBarInduction}^{\uparrow}$*. It records the finite bar decisions, monotone coverage, and local backwards induction replay visible in the current prefix tree.*
2. *T is the finite tree-depth ledger. It records the displayed root, visible extensions, retained prefix levels, and the finite fan boundary consumed by* $\mathsf{FanTheorem}^{\uparrow}$*.*
3. *I is the induction replay row. It records the local step converting displayed barred children into the displayed parent prefix, without selecting an infinite branch.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It schedules the finite prefix observations requested by Real and Completion consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, including the rational tolerances used to read the visible prefix depth.*
6. *M is the modulus output row. It records the displayed natural bound after which every accepted window read is determined by the barred-spread replay.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after B, T, I, W, R, M have been displayed.*

8. *H stores componentwise* hsame *transport for barred-spread data, tree-depth ledgers, induction replay, StreamName windows, RegSeqRat readback, modulus output, Real sealing, continuation, provenance, and local naming.*

9. *C stores displayed* Cont *replay from any finite-window request through the same $B, T, I, W, R, M, E, H, P, N$ route.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopBarInductionModulus}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopBarInductionModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, I, W, R, M, E, H, C, P, N$. It has no coordinate for countable choice, arbitrary open covers, quotient stream equality, a selected infinite branch, a host continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5132.2 (Bishop bar induction modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopBarInductionModulus}^{\uparrow}$ *carrier*

$$K = (B, T, I, W, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopBarInductionModulus}^{\uparrow}}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for barred-spread data, finite tree-depth ledgers, induction replay, StreamName windows, RegSeqRat readback, modulus output, Real sealing, transport, continuation, provenance, and local naming;*

2. *barred-spread ancestry, requiring every modulus read to expose the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *row B before using M;*

3. *fan-boundary exactness, requiring the displayed depth ledger T to be the only finite fan surface used in the modulus computation;*

4. *readback ordering, requiring every* $\mathsf{RegSeqRat}^{\uparrow}$ *read to pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row before reaching M and E;*

5. *analytic nonescape, requiring every Real or Completion-facing consumer to use the route $B, T, I, W, R, M, E, H, C$ and no hidden choice, cover, branch-selection, quotient, or ambient-completeness coordinate.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5132.1. The carrier admission row is exactly the tuple $B, T, I, W, R, M, E, H, C, P, N$, so the first obligation is a coordinate projection.

The barred-spread ancestry clause is forced by the placement of M after B and I. A consumer cannot read a modulus output without first displaying the bar decisions, monotone coverage, and local backwards replay already named by $\mathsf{BrouwerBarInduction}^{\uparrow}$.

The fan-boundary exactness clause is the role of T. The packet records only a finite depth ledger and visible prefix levels; it does not add an infinite branch or a separate compactness principle.

For readback ordering, the StreamName row W schedules the requested finite observations, the RegSeqRat row R reads them with rational tolerances, and only then does the modulus row M report the displayed bound. The Real seal E is downstream of that same route.

Finally H, C, P, N transport, replay, package, and name no further data. Since hidden choice, arbitrary covers, selected branches, quotient stream equality, and ambient Real completeness are not carrier coordinates, they cannot be local NameCert obligations. □

Theorem 5132.3 (Bishop bar induction modulus window boundary)**.** *Every* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing read of an accepted* $\mathsf{BishopBarInductionModulus}^{\uparrow}$ *carrier factors through the displayed finite route*

$$\mathsf{BrouwerBarInduction}^{\uparrow}, \mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BishopBarInductionModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

Equivalently, the exported modulus is determined by the rows $B, T, I, W, R, M, E, H, C, P, N$ *and cannot expose countable choice, open-cover compactness, quotient stream equality, selected infinite branches, host continuity, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an analytic-facing read of K. The only window interface in the carrier is W, so the request must first become a displayed finite $\mathsf{StreamName}^{\uparrow}$ observation.

The readback row R is attached to that same finite observation. It supplies regular rational tolerances for the visible prefix depth, not a quotient stream representative or a completed branch.

The modulus row M is downstream of B, T, I, W, R. Thus its bound is read only after the barred-spread row, finite fan-depth ledger, induction replay, window schedule, and rational readback have all been displayed.

The terminal seal E can therefore carry only the same finite route to Real or Completion consumers. Componentwise transport, continuation replay, provenance, and local naming preserve this route through H, C, P, N, and the carrier has no additional coordinate from which the excluded principles could be recovered. □

Falsifiable prediction: Every accepted $\mathsf{BishopBarInductionModulus}^{\uparrow}$ Real or Completion-facing read exposes barred-spread data, a finite tree-depth ledger, induction replay, StreamName windows, RegSeqRat readback, a modulus output, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any modulus boundary is visible.

Independence witness: brouwer-bar-induction, fantheorem, streamname, regseqrat, real: BishopBarInductionMo is not bijective to any listed sibling because it jointly carries barred-spread ancestry, finite fan-depth data, induction replay, finite windows, regular rational readback, a modulus output, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopBarInductionModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopBarInductionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5132.1, Definition 5132.2, and Definition 5132.3 seed the finite barred-spread modulus route from Brouwer bar induction and fan-depth rows through StreamName windows, RegSeqRat readback, terminal Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopBarInductionModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, open-cover compactness, quotient stream equality, selected infinite branches, host continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, barred-spread ancestry, fan-boundary exactness, readback ordering, modulus-window boundary, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopBarInductionModulusUp.

Constructive story: A $\mathsf{BishopBarInductionModulus}^{\uparrow}$ packet is generated from BrouwerBarInductionUp barred-spread rows, FanTheoremUp finite depth ledgers, induction replay, StreamName windows, RegSeqRat readback, a displayed modulus output, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 5132.4 (Bishop bar-induction modulus fan-theorem dependency)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{BishopBarInductionModulus}^{\uparrow}$ *packet must first read the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *barred-spread row and the* $\mathsf{FanTheorem}^{\uparrow}$ *finite-fan row before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal rows can be consumed. In carrier terms, the barred-spread row* B *and finite tree-depth*

ledger T precede the window row W, readback row R, modulus output M, terminal seal E, transport rows H, continuation rows C, provenance row P, and local naming row N. The packet exports a finite modulus handoff, not countable choice, open-cover compactness, quotient stream equality, selected branch data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5132.1. The rows B and T are displayed before the analytic window route; B records the barred-spread ancestry and T records the finite fan boundary.

Apply the NameCert obligations of Definition 5132.2. The barred-spread ancestry clause requires the $\mathsf{BrouwerBarInduction}^{\uparrow}$ row before a modulus read, and the fan-boundary exactness clause confines the finite fan surface to T.

Route the remaining rows through the window boundary of Definition 5132.3. The $\mathsf{StreamName}^{\uparrow}$ row W, $\mathsf{RegSeqRat}^{\uparrow}$ readback R, modulus output M, and $\mathsf{Real}^{\uparrow}$ seal E are consumed only after the barred-spread and finite-fan rows have been displayed.

Finally use the carrier non-escape clauses built into the same obligations. Since countable choice, open-cover compactness, quotient stream equality, selected branch data, and ambient Real completeness are not coordinates of $K = (B, T, I, W, R, M, E, H, C, P, N)$, they cannot replace the finite dependency route. □

The same finite fan-theorem dependency route is consumed by the Real-facing equality-modulus lattice of chapter 7601: barred-spread and fan finite-depth data supply the modulus handoff before that consumer reads $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows.

Chapter 5133

A Concrete Naming Certificate for $\mathsf{DyadicGridProjection}^{\uparrow}$

Following Bishop and Bridges on constructive real approximation by rational grids, the $\mathsf{DyadicGridProjection}^{\uparrow}$ packet records the finite operation that projects a displayed real observation onto adjacent dyadic endpoints at a requested unary precision. It sits after the rational denominator discipline of chapter 291, the regular Cauchy readback of chapter 301, the real-name seal of chapter 38, and the dyadic rounding surface of chapter 433. The packet gives L10 Real and Completion routes a constructive grid object without importing host rounding, trichotomy, quotient reals, selected limits, or ambient completeness.

Definition 5133.1 (Dyadic grid projection carrier)**.** *A* $\mathsf{DyadicGridProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (x, n, a, b, I, E, H, C, P, N)$$

with the following displayed rows:

1. *x is the source observation row, carrying the finite real-name or regular Cauchy read requested by the consumer.*
2. *n is the unary precision row, naming the dyadic scale at which the grid is inspected.*
3. *a and b are adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows at precision n.*
4. *I is the interval-containment row, displaying that the finite observation of x lies inside the closed dyadic window bounded by a and b.*
5. *E is the finite error-budget row, recording the tolerance consumed by the projection and the remaining budget available to downstream readback.*
6. *H records componentwise* hsame *transport for source observations, unary precision rows, dyadic endpoints, containment windows, error budgets, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$x, n, a, b, I, E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{DyadicGridProjection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicRounding}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicGridProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $x, n, a, b, I, E, H, C, P, N$. It has no coordinate for host rounding, trichotomy on rationals, quotient real equality, selected Cauchy limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DyadicGridProjectionUp.DyadicGridProjectionTasteGate single carrier alignment`

Theorem 5133.2 (Dyadic grid projection endpoint coverage)**.** *For every accepted* $\mathsf{DyadicGridProjection}^{\uparrow}$ *carrier* $G = (x, n, a, b, I, E, H, C, P, N)$*, the endpoint rows* a *and* b *are exposed as adjacent* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoints at the displayed unary precision* n*, and the containment row* I *is read only through that adjacent pair. Hence a consumer receives a finite dyadic window covering the displayed source observation* x *without a host rounding operator or trichotomy oracle.*

Proof. Project the carrier of Definition 5133.1. The only endpoint coordinates are a and b, and their role is fixed by the displayed unary precision row n.

The containment row I is listed after the endpoint rows, so it can be read only as a finite interval statement over the same adjacent dyadic pair. The finite error budget E records the tolerated projection loss but does not add a selector for an exact host representative.

Transport, replay, provenance, and local naming preserve the same rows. Since the carrier has no coordinate for host rounding, rational trichotomy, quotient real equality, selected limits, or ambient completeness, the coverage exported by the packet is exactly the displayed finite endpoint window. □

Theorem 5133.3 (Dyadic grid projection regular-real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicGridProjection}^{\uparrow}$ *packet must factor through the finite route*

$$x, n, a, b, I, E \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may use the displayed dyadic endpoints, containment row, and remaining finite error budget, but it cannot replace them by quotient reals, selected limits, or an ambient completeness theorem.

Proof. Start with the source observation x and unary precision n. The packet exposes adjacent dyadic endpoints a, b and the interval-containment row I before any regular Cauchy or real-name consumer reads the projection.

The finite error budget E is the only tolerance ledger carried across the handoff. A $\mathsf{RegSeqRat}^{\uparrow}$ read therefore receives a displayed dyadic window and budget rather than a selected representative of a quotient stream.

Finally the $\mathsf{Real}^{\uparrow}$ seal can consume only the same finite data through H, C, P, N. The carrier supplies no host limit selection or completeness principle, so the regular-real handoff remains a finite NameCert route. □

Falsifiable prediction: Every accepted $\mathsf{DyadicGridProjection}^{\uparrow}$ read displays one source-observation row, one unary precision row, two adjacent $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, one interval-containment row, one finite error-budget row, and local transport, replay, provenance, and naming rows before it is consumed by a regular Cauchy or Real route.

Independence witness: dyadicratcore, dyadicrounding, regseqrat, real: $\mathsf{DyadicGridProjection}^{\uparrow}$ is not bijective to any listed sibling because it binds a source observation, unary precision, adjacent dyadic endpoints, interval containment, finite error budget, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}$ into one accepted projection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicGridProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicGridProjection}^{\uparrow}$ packet is generated from a source observation, unary precision, adjacent DyadicRatCore endpoints, interval containment, finite error budget, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicGridProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5133.1, Definition 5133.2, and Definition 5133.3 seed the finite dyadic grid-projection route over DyadicRatCore, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicGridProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host rounding, rational trichotomy, quotient real equality, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-observation admission, unary precision handling, adjacent dyadic endpoint coverage, interval containment, finite error-budget transport, replay, provenance, and local naming for BEDC.Derived.DyadicGridProjectionUp.

Chapter 5134

A Concrete Naming Certificate for $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximation by finite rational data, the RegularCauchyRoundingP packet records the finite dyadic projection surface used when compact-uniform-continuity routes pass from finite nets to real-name observations. It grounds the compact finite-net discussion of chapter 9237 in the cofinal tail windows of chapter 4060, the dyadic denominator discipline of chapter 291, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet exposes finite projection data only; it does not import quotient streams, selected limits, countable choice, or ambient real completeness.

Definition 5134.1 (Regular Cauchy rounding projection carrier)**.** *A* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (q, W, D, R, T, S, H, C, P, N)$$

with the following displayed rows:

1. *q is the requested compact-uniform tolerance row, expressed as a finite precision demand.*
2. *W is the* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *tail-window row selected for that demand.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid row used to round the finite window into dyadic endpoints.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after the window and dyadic grid are displayed.*
5. *T is the cofinal tail-stability row, recording that later accepted tail windows factor through the same finite projection data.*
6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal-handoff row, exposing only the finite regular Cauchy read and remaining tolerance.*
7. *H records componentwise* hsame *transport for tolerance requests, cofinal windows, dyadic grids, regular readback, tail stability, Real seals, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the route*

$$q, W, D \longmapsto R, T \longmapsto S \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$, $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *compact finite-net rows,* BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRoundingProjection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $q, W, D, R, T, S, H, C, P, N$. It has no coordinate for quotient Cauchy stream equality, selected limits, countable choice, host rounding, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyRoundingProjectionUp.RegularCauchyRoundingProjectionTasteGate`
`single carrier alignment`

Theorem 5134.2 (Regular Cauchy rounding projection cofinal-window dependency)**.** *Every compact-uniform-continuity or* $\mathsf{RealSeparability}^{\uparrow}$ *consumer that uses a* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ *packet must first read the cofinal-window route of chapter 4060. Only after that route is displayed may the consumer read the* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of Definition 5134.1.*

Proof. Project the carrier. The cofinal-window row W precedes the dyadic grid, regular readback, tail-stability, and Real seal rows in the displayed packet.

A compact-uniform-continuity or $\mathsf{RealSeparability}^{\uparrow}$ consumer therefore cannot reach D, R, or S without first citing the cofinal-window chapter named by W. The remaining structural rows H, C, P, N replay and name the same route, so they do not create a bypass around the cofinal-window dependency. □

Theorem 5134.3 (Regular Cauchy rounding projection finite coverage)**.** *For every accepted* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ *carrier $Q = (q, W, D, R, T, S, H, C, P, N)$, the requested tolerance row q is covered by a displayed cofinal window W and a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid row D before any* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is exported. Thus compact finite-net consumers receive a finite dyadic projection surface rather than a host rounding operation.*

Proof. Project the carrier of Definition 5134.1. The rows preceding regular readback are exactly the tolerance request q, the cofinal window W, and the dyadic grid row D.

The readback row R is downstream of those finite rows. Therefore a compact finite-net consumer can cite a displayed dyadic projection for the requested tolerance, but cannot read R as a quotient stream representative or a host-rounded real.

The structural rows H, C, P, N preserve the same route. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, host rounding, or completeness, finite coverage is the only exported projection claim. □

Theorem 5134.4 (Regular Cauchy rounding projection cofinal tail stability)**.** *If two accepted* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ *packets share the tolerance row q and their cofinal-window rows are later refined along the same* $\mathsf{RegularCauchyWindowCofinality}^{\uparrow}$ *route, then the tail-stability row T displays a common finite projection through which the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *grid reads and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks factor. The comparison does not require quotient stream equality, selected limits, countable choice, or ambient completeness.*

Proof. Read the two carrier projections through Definition 5134.1. Both packets expose the same tolerance request and cofinal-window route before their dyadic grid and regular readback rows are consumed.

Use the tail-stability row T. It records the finite later-window factorization for the displayed cofinal route, so the dyadic grid rows and regular readback rows can be compared through a common finite projection.

The argument remains within $q, W, D, R, T, S, H, C, P, N$. No quotient stream equality, selected limit, countable-choice selector, host rounding operator, or ambient completeness theorem is present in the carrier. □

Theorem 5134.5 (Regular Cauchy rounding projection Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ *packet must factor through the finite route*

$$q, W, D \longmapsto R, T \longmapsto S \longmapsto \mathsf{Real}^{\uparrow}.$$

The Real-facing seal receives the displayed regular Cauchy readback, cofinal tail-stability row, and remaining finite tolerance only.

Proof. Start with the displayed route in the carrier. The tolerance request, cofinal window, and dyadic grid rows are read before R and T, so the Real-facing row cannot bypass the finite projection surface.

The seal-handoff row S is listed after regular readback and tail stability. It therefore packages exactly the finite regular Cauchy projection and remaining tolerance available to $\mathsf{Real}^{\uparrow}$.

Transport, replay, provenance, and local naming do not add endpoint principles. The consumer receives a finite seal handoff, not quotient real equality, selected limits, countable choice, host rounding, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ read displays one tolerance row, one cofinal tail-window row, one $\mathsf{DyadicRatCore}^{\uparrow}$ grid row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one tail-stability row, one $\mathsf{Real}^{\uparrow}$ seal-handoff row, and local transport, replay, provenance, and naming rows before it is consumed by a compact-uniform route.

Independence witness: regular-cauchy-window-cofinality, dyadicratcore, regseqrat, real: RegularCauchyRounding is not bijective to any listed sibling because it binds a compact-uniform tolerance request, cofinal tail window, dyadic grid row, regular Cauchy readback, tail stability, Real seal handoff, componentwise hsame, displayed Cont, Pkg, and local NameCert into one finite projection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRoundingProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRoundingProjection}^{\uparrow}$ packet is generated from a compact-uniform tolerance request, a RegularCauchyWindowCofinality tail window, a DyadicRatCore grid row, RegSeqRat readback, cofinal tail stability, a Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyRoundingProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5134.1, Definition 5134.3, Definition 5134.4, and Definition 5134.5 seed the finite regular-Cauchy rounding projection route over cofinal windows, dyadic grids, RegSeqRat readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyRoundingProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy stream equality, selected limits, countable choice, host rounding, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked tolerance admission, cofinal-window selection, dyadic grid projection, RegSeqRat readback, tail stability, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyRo

Chapter 5135

A Concrete Naming Certificate for $\mathsf{FiniteDifference}^{\uparrow}$

Following Newton and standard finite-difference calculus, the $\mathsf{FiniteDifference}^{\uparrow}$ packet records finite stencil evaluation as displayed BEDC window data rather than as a host derivative or a completed real-function equality. It sits over the real-name boundary of chapter 38, the derivative-facing route of chapter 122, the calculus boundary of chapter 3632, the Taylor consumer of chapter 5898, the dyadic tolerance ledger of chapter 291, the finite stream interface of chapter 126, and the regular rational readback of chapter 301. The packet supplies derivative, calculus, and Taylor consumers with a finite-difference surface; it does not claim host differentiability, an infinite function object, quotient real equality, Real completeness, selected limits, or a Taylor remainder theorem.

Definition 5135.1 (Finite difference carrier)**.** *A* $\mathsf{FiniteDifference}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, H_0, L, D, Q, R, E, B, T, C, P, N)$$

with the following displayed rows:

1. *S is the source-function window row, listing the finite input-output observations read from a* $\mathsf{StreamName}^{\uparrow}$ *source.*
2. *H_0 is the step-history row, recording the displayed base point, step requests, mesh refinements, and endpoint reuse schedule.*
3. *L is the finite stencil row, naming the inspected offsets, coefficients, order tag, and normalization denominator.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for step-size, coefficient, and denominator budgeting.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational stencil sum and its finite error ledger.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q has been read.*
7. *E is the consumer-boundary row for* $\mathsf{Derivative}^{\uparrow}$*,* $\mathsf{Calculus}^{\uparrow}$*, and* $\mathsf{TaylorTheorem}^{\uparrow}$*, exposing only the finite stencil route to those chapters.*
8. *B records componentwise* hsame *transport for the source windows, step history, stencil, dyadic tolerance, regular readback, Real seal, consumer boundary, replay, provenance, and local naming rows.*
9. *T records displayed* Cont *replay of the finite route*

$$S, H_0, L \longmapsto D \longmapsto Q \longmapsto R \longmapsto E.$$

10. *C is the classifier row, comparing accepted packets only through the same displayed source windows, step history, stencil, tolerance, readback, Real seal, and consumer boundary.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteDifference}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, $\mathsf{TaylorTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDifference}^{\uparrow}}$ *row.*

The classifier has no coordinate for an ambient function space, a host derivative, a quotient real equality, a selected limit, a Taylor remainder object, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteDifferenceUp.FiniteDifferenceTasteGate single carrier alignment`

Theorem 5135.2 (FiniteDifferenceUp NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDifference}^{\uparrow}$ *carrier*

$$F = (S, H_0, L, D, Q, R, E, B, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDifference}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *source-window admission for S, requiring every function value used by the stencil to be displayed as finite* $\mathsf{StreamName}^{\uparrow}$ *data;*
2. *step-history admission for H_0, requiring the base point, step, mesh-refinement, and endpoint-reuse schedule to be named before a stencil sum is read;*
3. *stencil admission for L, requiring offsets, coefficients, order tags, and denominators to be finite rows;*
4. *dyadic budgeting for D, requiring all step-size, coefficient, and denominator tolerance checks to pass through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
5. *regular rational readback for Q, requiring the finite stencil sum to pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *before any real seal is exposed;*
6. *Real seal exposure for R, requiring every real-valued difference read to be downstream of Q;*
7. *derivative, calculus, and Taylor boundary exposure for E, requiring those consumers to see only the displayed finite stencil route;*
8. *transport, replay, classifier, provenance, and local naming through B, T, C, P, N, preserving exactly the same finite packet.*

No finite-difference obligation is stored outside these rows.

Proof. Project the carrier of Definition 5135.1. The source row S and step-history row H_0 are the only places where input values, base points, steps, mesh requests, and endpoint reuse can be read.

The stencil row L fixes the finite offsets, coefficients, order tag, and denominator. The dyadic row D then supplies the only tolerance budget for the displayed step and coefficient data.

The numerical readback must pass through Q before a real-facing consumer can inspect R. Thus the route exports a regular rational stencil read and then a terminal $\mathsf{Real}^{\uparrow}$ seal, not a host derivative or quotient real equality.

Finally E exposes the same finite route to $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, and $\mathsf{TaylorTheorem}^{\uparrow}$, while B, T, C, P, N transport, replay, compare, source, and name the same tuple. Since the carrier has no ambient function-space, selected-limit, Taylor-remainder, host-differentiability, or completeness coordinate, the listed rows exhaust the local naming surface. □

Theorem 5135.3 (FiniteDifferenceUp Taylor handoff)**.** *Every* $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, *or* $\mathsf{Calculus}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteDifference}^{\uparrow}$ *packet receives only the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads finite source-function windows and step history, then the finite stencil, then dyadic tolerance budgeting, then regular rational readback, and only then a terminal real seal. The handoff does not export host differentiability, an infinite function graph, a quotient-real equality, a selected limiting derivative, a Taylor remainder theorem, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the NameCert obligations of Definition 5135.2. They place every accepted read on the displayed rows $S, H_0, L, D, Q, R, E, B, T, C, P, N$.

A derivative-facing or calculus-facing consumer can therefore inspect only the source windows, step history, and finite stencil before a numerical budget is available. The dyadic row D supplies that budget, and the regular rational row Q is the only readback route from the stencil sum.

The Taylor-facing boundary is E, which is downstream of the same Q and R rows. Componentwise hsame transport and Cont replay preserve this order, while Pkg provenance and the local NameCert row name exactly the same finite packet.

The excluded differentiability, infinite-graph, quotient-equality, selected-limit, Taylor-remainder, and completeness claims are absent from the carrier and cannot be recovered from the handoff. □

Theorem 5135.4 (FiniteDifferenceUp regular-real handoff)**.** *Every accepted* $\mathsf{FiniteDifference}^{\uparrow}$ *packet*

$$F = (S, H_0, L, D, Q, R, E, B, T, C, P, N)$$

exposes a Real-facing finite-difference value only by first reading the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, and then the* $\mathsf{Real}^{\uparrow}$ *seal row* R*. A* $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Calculus}^{\uparrow}$, *or* $\mathsf{TaylorTheorem}^{\uparrow}$ *consumer cannot bypass this route through host differentiability, quotient real equality, selected limits, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5135.1. The rows S, H_0, L display the source windows, step history, and stencil, but they do not yet expose a Real-facing value.

Apply Definition 5135.2. Its dyadic budgeting clause makes D the only tolerance row available for step-size, coefficient, and denominator checks. Its regular rational clause then makes Q the only readback row for the finite stencil sum before any real seal is public.

Now use Definition 5135.3. The consumer boundary E is downstream of the same D, Q, R chain, so derivative, calculus, and Taylor readers inherit the displayed route rather than a host derivative or selected limit.

The remaining rows B, T, C, P, N transport, replay, compare, source, and name the same packet. Since no carrier coordinate stores host differentiability, quotient real equality, selected limiting data, or ambient completeness, coordinate exhaustion leaves only the path $D \longmapsto Q \longmapsto R$ for Real-facing finite-difference values. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDifference}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDifference}^{\uparrow}$ packet is generated from source-function windows, step history, a finite stencil, DyadicRatCore tolerance budgeting, RegSeqRat readback, a RealUp seal, Derivative/Calculus/Taylor consumer boundaries, componentwise hsame transport, Cont replay, classifier routing, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteDifference}^{\uparrow}}$, seed)

Scope closed: Definition 5135.1, Definition 5135.2, and Definition 5135.3 seed the finite stencil evaluation surface over StreamName, DyadicRatCore, RegSeqRat, Real, Derivative, Calculus, Taylor, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteDifference}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host differentiability, infinite function objects, quotient real equality, selected limiting derivatives, Taylor remainder theorems, Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked source-window admission, step-history admission, stencil admission, dyadic budgeting, regular readback, Real sealing, consumer-boundary exposure, and transport/provenance/name exhaustion for BEDC.Derived.FiniteDifferenceUp.

Chapter 5136

A Concrete Naming Certificate for $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$

Following Horn and Johnson, *Matrix Analysis*, the $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ packet records the finite matrix-analysis route by which a displayed matrix or finite-dimensional operator exposes an isolated eigen-window and a positive separation radius before any completed host spectrum is imported. It sits after the matrix carrier of chapter 49, the linear-map interface of chapter 48, the vector-space carrier of chapter 47, the spectral-radius boundary of chapter 2352, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite separation certificate for downstream perturbation and spectral-theorem routes; it does not choose eigenvectors, complete a spectrum, or identify a host operator with an ambient spectral object.

Definition 5136.1 (Finite-dimensional spectral-gap carrier)**.** *A* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, T, V, E, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *M is the square* $\mathsf{Mat}^{\uparrow}$ *row, exposing the finite entry table and shape ledger.*
2. *T is the* $\mathsf{LinMap}^{\uparrow}$ *or finite operator row read from M over the displayed* $\mathsf{VecSpace}^{\uparrow}$ *source V.*
3. *V is the finite-dimensional* $\mathsf{VecSpace}^{\uparrow}$ *row, carrying the basis-window and scalar-action data used by the matrix/operator comparison.*
4. *E is the finite eigen-window row. It lists the displayed candidate eigenvalue cells, multiplicity tags when present, and the matrix-action comparison rows used to test each cell.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *separation-radius row, recording a positive dyadic distance between the distinguished eigen-window and every other displayed spectral window.*
6. *R is the* $\mathsf{SpectralRadius}^{\uparrow}$ *bound row, exposing the radius or norm-growth boundary used to keep the displayed windows inside the same finite operator packet.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the gap radius and the accepted eigen-window endpoints.*
8. *H records componentwise* hsame *transport for M, T, V, E, D, R, S and the structural rows.*

9. *C records displayed* Cont *replay of the route*

$$\mathsf{Mat}^{\uparrow}, \mathsf{LinMap}^{\uparrow}, \mathsf{VecSpace}^{\uparrow}, \mathsf{SpectralRadius}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto E, D, R, S \longmapsto \mathsf{FiniteDimensionalSpectralGap}^{\uparrow}.$$

10. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, V, E, D, R, S, H, C, P, N$. It has no coordinate for a completed host spectrum, selected eigenvectors, spectral projections, Schur or Jordan form, quotient equality of matrices, ambient compactness, or choice.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalSpectralGapUp.TasteGate.FiniteDimensionalSpectralGapTasteGate single carrier alignment`

Theorem 5136.2 (Finite-dimensional spectral-gap NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ *carrier*

$$G = (M, T, V, E, D, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *matrix/operator admission for the displayed* $\mathsf{Mat}^{\uparrow}$ *row M,* $\mathsf{LinMap}^{\uparrow}$ *row T, and* $\mathsf{VecSpace}^{\uparrow}$ *row V;*
2. *finite eigen-window admission through E, including only the displayed candidate cells and matrix-action comparison rows;*
3. *dyadic separation and real sealing through D and S, requiring the gap radius and eigen-window endpoints to be read from displayed dyadic and real-name rows;*
4. *spectral-radius compatibility through R, bounding the accepted windows inside the same finite operator packet without replacing them by a completed spectrum;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

No spectral-gap obligation is stored outside these rows.

Proof. Project the carrier of Definition 5136.1. The matrix, operator, and vector-space rows M, T, V are the only source rows from which a consumer can read the finite-dimensional action.

The eigen-window obligation is the row E. It displays candidate cells and the corresponding action comparisons, so the packet exposes a finite windowed test rather than a selected eigenbasis or a host eigenspace.

The gap radius is read through D and S. The dyadic row records positive separation from the other displayed windows, and the real seal records the endpoint data that downstream consumers may inspect.

The spectral-radius row R supplies only a finite bound-facing boundary over the same operator packet. Together with H, C, P, N, it transports, replays, sources, and names the displayed surface, so the five listed rows exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.FiniteDimensionalSpectralGapUp.FiniteDimensionalSpectralGap namecert obligations`

Theorem 5136.3 (Finite-dimensional spectral-gap nonescape)**.** *Every spectral-gap read of an accepted* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ *packet factors through the route*

$$M, T, V \longmapsto E \longmapsto D, S \longmapsto R \longmapsto H, C, P, N.$$

Consequently the distinguished eigen-window and its separation radius remain inside the displayed finite matrix/operator packet. A downstream consumer receives no completed host spectrum, no selected eigenvector, no spectral projection, no Schur or Jordan decomposition, no quotient matrix equality, and no choice principle.

Proof. Start with an accepted packet $G = (M, T, V, E, D, R, S, H, C, P, N)$. Carrier projection first exposes the matrix row, the induced linear-map row, and the finite vector-space source.

The eigen-window row E is downstream of those rows. It can compare displayed candidate cells with the matrix action, but it cannot introduce an eigenvector selector or an ambient spectral set because neither appears as a carrier coordinate.

The separation data are then read through D and S. The positive dyadic radius and its real seal are attached to the same displayed window, so a consumer cannot detach the gap from the finite packet that produced it.

Finally the spectral-radius bound R and the structural rows H, C, P, N keep the readback within the same transported and replayed surface. Since the excluded spectral objects have no field in the packet, the route cannot escape to a host spectrum or a choice-based eigenvalue selection. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ consumer exposes the matrix row, linear-map row, finite vector-space row, eigen-window row, dyadic separation row, spectral-radius bound row, RealUp seal, componentwise transport, replay, provenance, and local naming before reading a spectral-gap conclusion.

Independence witness: matrix, linearmap, vecspace, spectralradius, dyadicratcore, real: MatrixUp supplies finite entries without an eigen-window separation row; LinearMapUp supplies action rows without a named gap; VecSpaceUp supplies scalar and basis transport without spectral separation; SpectralRadiusUp supplies a radius boundary without isolating a window; DyadicRatCoreUp supplies rational tolerances without matrix action; RealUp seals endpoints without naming the finite spectral-gap packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ packet is generated from a displayed MatrixUp row, a LinearMapUp finite operator row, a VecSpaceUp source row, a finite eigen-window row, a DyadicRatCore separation-radius row, a SpectralRadiusUp bound row, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5136.1, Definition 5136.2, Definition 5136.3, the falsifiable prediction, and the independence witness seed the finite matrix spectral-gap packet and its nonescape boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a completed host spectrum, selected eigenvectors, spectral projections, Schur or Jordan decomposition, quotient matrix equality, ambient compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked matrix/operator admission, finite eigen-window admission, dyadic separation, Real sealing, spectral-radius compatibility, transport, replay, provenance, and consumer nonescape for BEDC.Derived.FiniteDimensio

Chapter 5137

A Concrete Naming Certificate for $\mathsf{SpectralGap}^{\uparrow}$

Following Kato, the $\mathsf{SpectralGap}^{\uparrow}$ packet records spectral separation as a finite witnessed gap for bounded-operator and real-norm rows. It generalizes the matrix-level certificate of chapter 5136 while staying anchored in the operator surface of chapter 1644, the norm ledger of chapter 89, and the endpoint seal of chapter 38. The packet names only displayed separation data; it does not assert a spectral theorem, resolvent calculus, compactness of the spectrum, or a hidden eigenvector selection principle.

Definition 5137.1 (SpectralGap carrier)**.** *A* $\mathsf{SpectralGap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, B, N, R, S, \Delta, W, H, C, P, L)$$

with the following displayed rows:

1. *T is the source* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row naming the bounded operator whose spectrum-facing read is under inspection.*
2. *B is the boundedness ledger for T, including the operator-bound row and the finite replay path by which downstream consumers may inspect it.*
3. *N is the* $\mathsf{Norm}^{\uparrow}$ *row used to read operator and vector bounds through the displayed norm carrier.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *row naming the real endpoint surface on which separation bounds and positive-radius witnesses are sealed.*
5. *S is the finite spectral-side row listing the inspected spectral cluster, complementary spectral window, and allowed localization probes.*
6. *Δ is the positive gap row: a displayed real-norm lower bound separating S from its complement by a finite witness.*
7. *W is the finite witness row recording the endpoint comparisons, norm comparisons, and probe windows that certify the gap row Δ.*
8. *H stores componentwise* hsame *transport for bounded-operator, norm, real, spectral-side, gap, witness, continuation, provenance, and local naming rows.*
9. *C stores displayed* Cont *replay from the bounded-operator row through norm and real comparisons to the gap witness.*

10. *P is the* Pkg *provenance over* $\mathsf{SpectralGap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteDimensionalSpectralGap}$ BHist, hsame, Cont, *and* NameCert *rows.*

11. *L is the local* $\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, B, N, R, S, \Delta, W, H, C, P, L$. It has no coordinate for a global spectrum object, an unbounded operator, a chosen eigenbasis, a host Hilbert-space equality, a quotient of operators, or ambient classical spectral decomposition.

Theorem 5137.2 (SpectralGap NameCert obligation surface)**.** *For an accepted* $\mathsf{SpectralGap}^{\uparrow}$ *carrier $G = (T, B, N, R, S, \Delta, W, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$ *surface consists exactly of five rows:*

1. *carrier admission for the displayed bounded-operator, norm, real, spectral-side, gap, witness, transport, replay, provenance, and local naming rows;*
2. *bounded-operator dependency through T, B, requiring every operator read to pass through* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *before any gap witness is consumed;*
3. *norm and real separation through N, R, Δ, W, requiring the positive gap to be read as a finite real-norm inequality;*
4. *finite-dimensional compatibility through the sibling* $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$ *route whenever the inspected packet specializes to a displayed finite-dimensional window;*
5. *no-escape from global spectral theorems, hidden eigenvector selection, host operator quotients, unbounded-operator data, and ambient classical decomposition.*

Thus every downstream consumer of the gap must read

$$T, B, N, R, S, \Delta, W, H, C, P, L$$

as one finite BHist *packet before exporting the local certificate.*

Proof. Project the accepted carrier. The bounded-operator rows T, B are the only source of operator data, while the norm and real rows N, R are the only source of numerical comparison data.

The positive separation is then read from Δ and W. These rows give a finite lower-bound witness and the endpoint, norm, and probe comparisons that certify it; they do not call on a global spectrum object.

For finite-dimensional windows, the packet factors through the sibling surface of chapter 5136. That route supplies the specialized matrix-level gap read, while the present carrier keeps the general bounded-operator and norm rows visible.

Finally H, C, P, L transport, replay, source, and name the same packet. Since no excluded host object appears among the coordinates, the five displayed rows exhaust the local $\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$ obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SpectralGap}^{\uparrow}$ packet is generated from bounded-operator, boundedness, norm, real, spectral-side, positive-gap, finite-witness, componentwise transport, continuation replay, provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$, seed)

Scope closed: Definition 5137.1 and Definition 5137.2 seed the finite bounded-operator spectral-gap surface over $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{FiniteDimensionalSpectralGap}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SpectralGap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close spectral theorem machinery, resolvent calculus, compactness of the spectrum, hidden eigenvector selection, Lean verification, or any standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, bounded-operator dependency, norm and real separation, finite-dimensional compatibility, transport, replay, provenance, and no-escape for BEDC.Derived.SpectralGapUp.

Chapter 5138

A Concrete Naming Certificate for $\mathsf{CauchyRootCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyRootCriterion}^{\uparrow}$ packet records the finite root-test convergence data by which a real series source enters the regular Cauchy root-test route before any endpoint seal is read. It sits beside the real-series carrier of chapter 1581, the regular root-test packet of chapter 1481, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The certificate names the constructive root criterion as displayed finite rows; it does not choose an infinite sum, import host completeness, or select a limit.

Definition 5138.1 (Cauchy root criterion carrier)**.** *A* $\mathsf{CauchyRootCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, Q, W, D, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{RealSeries}^{\uparrow}$ *source row, exposing the real-valued term stream only through the series packet.*
2. *Q is the root-bound witness row, recording a displayed dyadic bound below the unit scale and the finite entry index after which the bound is admitted.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window row used to inspect the term stream and partial-tail data named by A.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the precision cells, slack conversions, and comparison budgets used by the root-bound witness.*
5. *T is the* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *handoff row that turns Q, W, D into a regular Cauchy tail route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the root-test handoff has consumed the same tail windows and dyadic tolerance row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational readback is displayed.*
8. *H records componentwise* hsame *transport for the source, root-bound, window, tolerance, handoff, readback, seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from a root-bound request through $A, Q, W, D, T, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRootCriterion}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RegularCauchyRootTest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRootCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host root-limit theorem, selected infinite sum, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRootCriterionUp.CauchyRootCriterionTasteGate single carrier alignment`

Theorem 5138.2 (Cauchy root criterion NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRootCriterion}^{\uparrow}$ *carrier* $K = (A, Q, W, D, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyRootCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet,* $\mathsf{RealSeries}^{\uparrow}$ *source exposure, root-bound witness exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *tail-window admission,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exactness,* $\mathsf{RegularCauchyRootTest}^{\uparrow}$ *handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyRootCriterionUp.CauchyRootCriterionNamecertObligations`

Proof. Project the accepted carrier of Definition 5138.1. The source, witness, window, tolerance, handoff, readback, seal, and structural rows are precisely $A, Q, W, D, T, R, E, H, C, P, N$.

The root-bound portion is read through Q, W, D. The bound and entry index are displayed by Q, the inspected tail data are displayed by W, and the tolerance comparisons are displayed by D.

The handoff row T is then forced through $\mathsf{RegularCauchyRootTest}^{\uparrow}$. It consumes the same windows and dyadic budget before any regular rational readback R can be exposed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since no omitted coordinate exists for host root limits, selected sums, quotient equality, choice, trichotomy, or ambient completeness, these rows exhaust the local NameCert obligations. □

Theorem 5138.3 (Cauchy root criterion RealSeries handoff)**.** *Every* $\mathsf{RealSeries}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyRootCriterion}^{\uparrow}$ *packet reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and then a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegularCauchyRootTest}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the root criterion supplies downstream RealSeries and RealPowerSeries chapters with a finite convergence-test certificate, while exporting no host completeness theorem, selected infinite sum, quotient stream equality, countable-choice schedule, or detached Real seal.

Proof. Start with the source row A. It exposes only the real-series term surface and cannot by itself supply convergence, an infinite sum, or an endpoint.

Read the finite tail windows and dyadic tolerance ledger next. The rows W and D place every inspected root-bound request inside displayed StreamName and DyadicRatCore data.

Apply the regular root-test handoff row T. It is the only route from the root-bound witness to a regular Cauchy tail, and the readback row R is downstream of that handoff.

The terminal Real seal E is therefore reached only after R. Componentwise transport, displayed replay, provenance, and local naming preserve this order, so no consumer receives host completeness, a selected sum, quotient stream equality, choice data, or a detached seal. □

Falsifiable prediction: Every accepted $\mathsf{CauchyRootCriterion}^{\uparrow}$ endpoint read factors through a RealSeries source row, root-bound witness, finite StreamName tail windows, DyadicRatCore tolerance ledger, RegularCauchyRootTest handoff, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local NameCert row.

Independence witness: realseries, regular-cauchy-root-test, streamname, dyadicratcore, regseqrat, real: no listed sibling supplies the whole eleven-row packet $K = (A, Q, W, D, T, R, E, H, C, P, N)$, because each sibling contributes only the source series, root-test handoff, finite windows, dyadic tolerance, regular readback, or endpoint seal while this packet binds them into one finite NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRootCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyRootCriterion}^{\uparrow}$ packet is generated from a RealSeries source row, root-bound witness, finite StreamName tail windows, DyadicRatCore tolerance ledger, RegularCauchyRootTest handoff, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRootCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5138.1, Definition 5138.2, and Definition 5138.3 seed the finite root-test convergence route from RealSeries data to regular rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRootCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host completeness, selected infinite sums, quotient stream equality, countable choice, trichotomy, RealPowerSeries convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, root-bound witness exactness, finite window coverage, dyadic tolerance exactness, root-test handoff, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRootCriterionUp.

$\mathsf{CauchySequentialCriterion}^{\uparrow}$ records the L10 finite Cauchy-window route consumed by chapter 301, chapter 126, and chapter 38; the child files below keep the NameCert body and sibling routes separate.

Chapter 5139

A Concrete Naming Certificate for $\mathsf{CauchySequentialCriterion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchySequentialCriterion}^{\uparrow}$ packet records the finite sequential Cauchy criterion consumed by the L10 Real and Completion phase before a terminal real-name or completion-facing handoff is read. It sits after the Cauchy-modulus space of chapter 5887, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal seal of chapter 38. The packet gives later Real and Completion consumers explicit finite Cauchy-window data rather than an informal convergence phrase. The gluing route of chapter 5899 consumes the same finite Cauchy-window data when two compatible local windows are glued before the $\mathsf{Real}^{\uparrow}$ seal.

Definition 5139.1 (Cauchy sequential criterion carrier)**.** *A* $\mathsf{CauchySequentialCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, W, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *row, carrying the modulus schedule that chooses windows for each displayed tolerance request;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the inspected stream entries and finite window boundaries;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, reading the inspected entries as regular rational data;*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row, recording the tolerance and pairwise-distance arithmetic used by the Cauchy test;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the regular rational and dyadic rows have been displayed;*
6. *L is the sequential criterion ledger, binding tolerance requests to windows and pairwise tails;*
7. *H records componentwise* hsame *transport for $A, W, R, D, E, L, C, P, N$;*
8. *C records displayed* Cont *replay for the same window route;*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySequentialCriterion}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5139.2 (Cauchy sequential criterion window obligations)**.** *For the carrier of Definition 5139.1, the local* $\mathsf{NameCert}_{\mathsf{CauchySequentialCriterion}^{\uparrow}}$ *surface consists exactly of modulus admission through* A*, stream-window inspection through* W*, regular rational readback through* R*, dyadic tolerance checking through* D*, Real seal handoff through* E*, criterion-ledger binding through* L*, and structural transport, replay, provenance, and local naming through* H, C, P, N*.*

Proof. Project the carrier $K = (A, W, R, D, E, L, H, C, P, N)$. The Cauchy-modulus row A supplies the finite schedule, and W supplies the stream windows on which that schedule is inspected.

Read the arithmetic rows next. The regular rational row R and dyadic row D compare displayed tail entries at the requested tolerance, while the ledger L binds those comparisons to the same window schedule.

Finally the terminal seal and structural rows are replayed. The row E is downstream of the displayed regular rational and dyadic data, and H, C, P, N only transport, replay, source, and name the same finite route. Therefore the listed rows exhaust the NameCert obligations. □

Theorem 5139.3 (Cauchy sequential criterion tail-window stability)**.** *Let*

$$K = (A, W, R, D, E, L, H, C, P, N)$$

be an accepted $\mathsf{CauchySequentialCriterion}^{\uparrow}$ *packet. After discarding any finite initial* $\mathsf{StreamName}^{\uparrow}$ *window, the retained tail is still accepted by the same criterion whenever it keeps the displayed Cauchy modulus row* A*, the retained StreamName window* W_{tail}*, the RegSeqRat readback* R_{tail}*, the DyadicRatCore tolerance row* D_{tail}*, the Real seal row* E*, the criterion ledger* L_{tail}*, componentwise transport* H_{tail}*, replay* C_{tail}*, provenance* P*, and local naming row* N*.*

Thus every Real or Completion consumer of the tail handoff reads only the retained finite window route

$$A, W_{\mathrm{tail}}, R_{\mathrm{tail}}, D_{\mathrm{tail}}, E, L_{\mathrm{tail}}, H_{\mathrm{tail}}, C_{\mathrm{tail}}, P, N,$$

and receives no quotient stream equality, selected host limit, countable choice, trichotomy, host real equality, or ambient completeness principle.

Proof. Project the accepted carrier K. The sequential criterion is stored in the displayed modulus schedule A, the inspected StreamName window W, the RegSeqRat readback R, the dyadic tolerance row D, and the ledger L.

Discarding a finite initial window changes only which displayed tail cells are retained. By hypothesis the retained packet still carries the same modulus row and the corresponding tail rows $W_{\mathrm{tail}}, R_{\mathrm{tail}}, D_{\mathrm{tail}}, L_{\mathrm{tail}}$, so the criterion is read from the retained finite windows and the displayed modulus ledger.

The structural rows are retained with the tail. Componentwise hsame transport, Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming therefore preserve the same finite route through the Real seal rather than adding a completion object.

The carrier has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, host real equality, or ambient completeness. The tail-window packet consequently remains inside the same accepted $\mathsf{CauchySequentialCriterion}^{\uparrow}$ surface. □

Theorem 5139.4 (Cauchy sequential criterion Real handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchySequentialCriterion}^{\uparrow}$ *packet must read*

$$A, W, R, D, E, L, H, C, P, N$$

before treating a sequential Cauchy claim as a real-facing or completion-facing handoff. The handoff factors through BHist, hsame, Cont, Pkg, CauchyModulusSpace$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *seal rows; it exports no quotient stream, selected limit, countable choice, host real equality, or ambient completeness principle.*

Proof. Begin with the window obligations of Definition 5139.2. They require the modulus schedule, stream windows, regular rational readback, dyadic tolerance arithmetic, and criterion ledger to be displayed before the terminal seal is read.

The structural rows keep the handoff finite. Transport through H, replay through C, provenance through P, and local naming through N preserve the same rows rather than adding a host completion object.

Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient completeness, every Real or Completion consumer receives exactly the displayed Cauchy-window route. □

Falsifiable prediction: An accepted CauchySequentialCriterion$^{\uparrow}$ consumer that reaches a Real or Completion handoff without first exposing the modulus schedule, stream window, regular rational readback, dyadic tolerance, terminal Real seal, ledger, transport, replay, provenance, and naming rows refutes the carrier surface.

Independence witness: cauchy-modulus-space, streamname, regseqrat, dyadicratcore, real: the modulus-space packet supplies schedules without terminal sequence readback; StreamName supplies windows without regular rational arithmetic; RegSeqRat supplies readback without dyadic tolerance ledgers; DyadicRatCore supplies arithmetic without a sequential carrier; RealUp supplies the terminal seal without the criterion ledger. None is bijective to CauchySequentialCriterion$^{\uparrow}$.

Closure status for NameCert$_{\text{CauchySequentialCriterion}^{\uparrow}}$.

Constructive story: A CauchySequentialCriterion$^{\uparrow}$ packet is generated from a CauchyModulusSpace schedule, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance arithmetic, a RealUp seal, a sequential criterion ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate(NameCert$_{\text{CauchySequentialCriterion}^{\uparrow}}$, seed)

Scope closed: Definition 5139.1, Definition 5139.2, Definition 5139.3, and Definition 5139.4 seed the finite sequential Cauchy-window route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchySequentialCriterion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked modulus admission, stream-window inspection, regular rational readback, dyadic tolerance checking, Real seal handoff, ledger binding, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequentialCriterionUp.

InactiveLayerQuotientDynamics$^{\uparrow}$ reads the orbit-state, weighted-path, inactive-layer, and trigger-boundary rows as a consumer of chapter 3393, chapter 818, chapter 3932, and chapter 1364.

Chapter 5140

A Concrete Naming Certificate for $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$

The $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$ packet realizes the zero-marker inactive-layer quotient dynamics of Definition 9462.1366 and Definition 9462.1368 as finite orbit-state NameCert data. It sits beside the DNA boundary-tile packet of chapter 3393, the finite trace evaluator of chapter 818, the Hilbert–Alexander blocker surface of chapter 3932, and the arity-ten BHist tuple capstone of chapter 1364. Those siblings expose boundary, finite evaluation, blocker, or tuple-capacity rows separately; this packet binds the six active orbit states, weighted path, inactive median layer, and active-inactive trigger boundary into one carrier.

Definition 5140.1 (Inactive-layer quotient dynamics carrier)**.** *An* $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O, W, I, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the six orbit-state row containing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*;*
2. *W is the weighted quotient-path row, recording exactly the cross-orbit Hamming weights displayed in Definition 9462.1366;*
3. *I is the inactive median-layer row, carrying* $\mathrm{WYG} \cup \{\mathrm{AAG}, \mathrm{ACA}, \mathrm{UGG}\}$ *and its square-with-leaves graph;*
4. *A is the active-inactive interface row, exposing the two active gates and the inactive-square contact point;*
5. *B is the boundary-tile provenance row read from the DNA six-window boundary surface;*
6. *T is the trigger-boundary ledger, binding the weighted path and inactive layer before any zero-marker dynamics read is exported;*
7. *H records componentwise* hsame *transport for* $O, W, I, A, B, T, C, P, N$*;*
8. *C records displayed* Cont *replay for the orbit-state, inactive-layer, and trigger-boundary route;*
9. *P is the* Pkg *provenance over* $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$*, the philosophy quotient-dynamics source, DNA boundary tiles, finite trace evaluation,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an unlisted cross-orbit Hamming edge, stochastic biological semantics, quotient codon equality, hidden genome dynamics, or a host automaton oracle.

Theorem 5140.2 (Inactive-layer quotient dynamics obligations)**.** *For the carrier of Definition 5140.1, the local* $\mathsf{NameCert}_{\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}}$ *surface consists exactly of orbit-state admission through* O*, weighted-path admission through* W*, inactive-layer admission through* I*, active-inactive interface admission through* A*, boundary-tile provenance through* B*, trigger-boundary binding through* T*, and structural transport, replay, provenance, and local naming through* H, C, P, N*.*

Proof. Project the carrier $Q = (O, W, I, A, B, T, H, C, P, N)$. The orbit row O displays the six states, and the weighted row W records the path and its cross-orbit weights.

Read the inactive and interface rows next. The row I displays the zero-marker inactive layer, while A identifies the active gates and contact point through which active and inactive reads meet.

The boundary and trigger rows bind the packet. The provenance row B keeps the construction on the DNA boundary-tile surface, and T requires the quotient path and inactive layer to be read together before dynamics are exported. Transport, replay, provenance, and naming then preserve the same finite inventory. □

Theorem 5140.3 (Inactive-layer quotient dynamics trigger boundary)**.** *Every accepted* InactiveLayerQuotientDynar *dynamics read factors through*

$$O, W, I, A, B, T, H, C, P, N.$$

In particular, a zero-marker inactive-layer read must expose the weighted six-state quotient path, the inactive square-with-leaves layer, the active gates, and the contact point before any trigger-boundary conclusion is read.

Proof. Use Definition 5140.2. It makes O, W, I, A the visible orbit, path, inactive-layer, and interface rows of the packet.

The trigger boundary is not independent of those rows. The row T binds the active and inactive surfaces after B has recorded the boundary-tile provenance, so a dynamics consumer cannot detach the inactive layer from the displayed quotient path.

Since the carrier has no unlisted cross-orbit edge, stochastic semantics, quotient codon equality, hidden dynamics row, or host automaton oracle, every accepted trigger-boundary read remains inside the displayed finite route. □

Falsifiable prediction: The weighted inactive-layer quotient path fails if a nonlisted cross-orbit Hamming edge is exhibited between two orbit states outside the displayed path rows O, W, I, A.

Independence witness: dna-window-six-boundary-tile, finite-trace-evaluator, hilbert-alexander-blocker, b-hist-deca-tuple-name-cert, metaciccriticalpath: none is bijective to $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$ because each lacks at least one of the six orbit-state row, weighted quotient path, inactive square-with-leaves layer, active gate interface, and trigger-boundary ledger bound in this packet.

Closure status for $\mathsf{NameCert}_{\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}}$.

Constructive story: An $\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}$ packet is generated from six orbit states, weighted quotient-path rows, inactive median-layer rows, active-inactive interface rows, boundary-tile provenance, a trigger-boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}}$, seed)

Scope closed: Definition 5140.1, Definition 5140.2, and Definition 5140.3 seed the finite inactive-layer quotient dynamics packet and its trigger boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{InactiveLayerQuotientDynamics}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim stochastic biological semantics, quotient codon equality, hidden genome dynamics, an unlisted cross-orbit Hamming edge, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked orbit-state admission, weighted-path admission, inactive-layer admission, active-inactive interface admission, trigger-boundary binding, transport, replay, provenance, and local naming for BEDC.Derived.InactiveLayerQuotientDynamicsUp.

The sibling dependency lattice for InactiveLayerQuotientDynamics$^{\uparrow}$ routes its boundary row through chapter 3393, its finite replay row through chapter 818, its blocker interface through chapter 3932, and its arity cap through chapter 1364. These routes keep the inactive-layer quotient dynamics packet tied to displayed DNA boundary, finite evaluation, blocker, and BHist tuple data before the trigger-boundary read is consumed.

Theorem 5140.4 (Inactive-layer quotient dynamics forward lattice)**.** *Every accepted* InactiveLayerQuotientDynamic *consumer that reads the trigger-boundary row must first expose the orbit-state row O, weighted quotient-path row W, inactive median-layer row I, active-inactive interface row A, boundary-tile provenance row B, trigger ledger T, componentwise transport H, replay row C, package provenance P, and local* NameCert *row N of Definition 5140.1. The boundary row is routed through chapter 3393, the finite replay row through chapter 818, and the blocker interface through chapter 3932 before the quotient-dynamics classifier may export a trigger-boundary read.*

Proof. Start with the carrier and obligation rows of Definition 5140.1 and Definition 5140.2. They make $O, W, I, A, B, T, H, C, P, N$ the complete displayed inventory of an accepted packet.

Apply the trigger-boundary theorem Definition 5140.3. It forces every dynamics read to pass through the weighted six-state quotient path, inactive square-with-leaves layer, active gates, contact point, and trigger ledger before a trigger-boundary conclusion is available.

Finally read the sibling lattice of the same hub. The DNA boundary-tile chapter supplies B, the finite trace evaluator supplies C, and the Hilbert–Alexander blocker supplies the blocker-facing interface cited by A, T. Thus the forward lattice is a carrier dependency route, not an isolated quotient-dynamics assertion. □

Chapter 5141

A Concrete Naming Certificate for $\mathsf{CauchyFilterModulus}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion, the $\mathsf{CauchyFilterModulus}^{\uparrow}$ packet names the finite modulus interface by which Cauchy-filter evidence becomes a regular rational handoff before a Real seal is read. It refines the filter surface of chapter 333, the basis generator of chapter 1501, and the regular rational route of chapter 301. Its consumer path also reads finite windows through chapter 126, dyadic ledgers through chapter 291, and the terminal seal boundary of chapter 38. The packet is a finite modulus surface for Cauchy-filter completion data; it is not a quotient filter, selected limit, countable-choice schedule, host equality comparison, ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, Lean verification row, or bridge certificate.

Definition 5141.1 (Cauchy filter modulus carrier)**.** *A* $\mathsf{CauchyFilterModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, B, G, S, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *source row whose displayed precision requests must be regularized.*
2. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *generator row. It names the finite basis elements and common-refinement witnesses used to inspect the requests in F.*
3. *G is the* $\mathsf{RegularCauchyFilter}^{\uparrow}$ *regularity row. It records the finite tail-basis test that turns the source request and basis witness into a regular Cauchy-filter read.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row reached only after F, B, G have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores the requested dyadic scale, the smaller scale certified by G, and the comparison used by the regular readback.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row. It repacks the accepted windows and dyadic ledger as a regular rational sequence row.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary visible only after F, B, G, S, D, R have been replayed.*
8. *H records componentwise* hsame *transport for the filter source, basis generator, regularity row, windows, dyadic ledger, regular readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a filter precision request through F, B, G, S, D, R, E in that order.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterModulus}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{RegularCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterModulus}^{\uparrow}}$ *row.*

The classifier compares accepted modulus packets only through the displayed route

$$F, B, G \longmapsto S, D, R, E \longmapsto H, C, P, N.$$

No carrier coordinate stores a quotient filter, selected limit, countable-choice schedule, host equality of filter values, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge certificate.*

Lean4 (definition) `BEDC.Derived.CauchyFilterModulusUp.CauchyFilterModulusUp`
Lean4 ✓ `BEDC.Derived.CauchyFilterModulusUp.CauchyFilterModulusTasteGate single carrier alignment`

Theorem 5141.2 (Cauchy filter modulus NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterModulus}^{\uparrow}$ *carrier*

$$M = (F, B, G, S, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Cauchy-filter source, basis generator, regularity row, window row, dyadic tolerance ledger, regular rational handoff, Real seal boundary, transport, replay, provenance, and local naming row;*
2. *source-to-basis routing, requiring each displayed filter precision request in F to read its finite basis witness from B before the regularity row is inspected;*
3. *regular filter modulus extraction, requiring G to name the finite tail-basis test and the scale comparison that make the requested filter read regular;*
4. *window and dyadic replay, requiring S, D to expose only the finite observations and dyadic budget attached to the same F, B, G route;*
5. *regular readback and Real handoff, requiring R to be read before the terminal* $\mathsf{Real}^{\uparrow}$ *boundary E is visible to completion-facing consumers;*
6. *transport and replay stability, requiring H and C to preserve the order F, B, G, S, D, R, E under componentwise* hsame *transport and displayed* Cont *replay;*
7. *provenance exhaustion, requiring P, N to name only this finite modulus interface and to export no quotient filter, selected limit, countable-choice schedule, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5141.1. The projection exposes exactly the source filter, finite basis row, regularity row, stream windows, dyadic ledger, regular rational readback, terminal seal, and structural rows H, C, P, N.

The source-to-basis route is fixed by row order. A precision request first reads F and then the finite basis witness supplied by B; the packet has no coordinate that can jump directly from a filter request to a regular sequence.

The regularity row G performs the modulus extraction. It names the finite tail-basis test and the scale comparison used to justify the regular Cauchy read, so the modulus is packet data rather than an ambient completeness or choice principle.

The replay row C then carries the same request through S, D, R, E. Transport in H preserves these displayed rows, while P, N source and name only the packet; therefore the excluded quotient, selected-limit, choice, host-equality, completeness, Lean-verification, and bridge-checking data are outside the local NameCert surface. □

Theorem 5141.3 (Cauchy filter modulus regular readback)**.** *Let*

$$M = (F, B, G, S, D, R, E, H, C, P, N)$$

be an accepted $\mathsf{CauchyFilterModulus}^{\uparrow}$ *carrier. Every displayed Cauchy-filter precision request factors through a finite regular readback route*

$$F \longmapsto B \longmapsto G \longmapsto S, D \longmapsto R.$$

Consequently the packet exports a $\mathsf{RegSeqRat}^{\uparrow}$ *row only after the source filter request has been witnessed by a basis row, checked by the regular Cauchy-filter row, and replayed through the finite window and dyadic tolerance rows.*

Proof. Start with a displayed precision request in F. The carrier exposes no direct $\mathsf{RegSeqRat}^{\uparrow}$ coordinate from F, so the request must first be read through the finite basis row B.

Next inspect G. This row attaches the basis witness to the regular Cauchy-filter test and records the smaller scale used by the modulus request.

The finite observations then pass through S and the dyadic comparison through D. These rows are tied to the same source and basis data by carrier projection and componentwise transport.

Finally R repacks the accepted finite observations and tolerance ledger as a regular rational sequence row. Thus the regular readback is exported only through the displayed route. □

Theorem 5141.4 (Cauchy filter modulus Real handoff)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyFilterModulus}^{\uparrow}$ *carrier*

$$M = (F, B, G, S, D, R, E, H, C, P, N)$$

reaches the terminal seal E only through the finite route

$$F, B, G, S, D, R \longmapsto E \longmapsto H, C, P, N.$$

In particular, the consumer receives a Cauchy-filter source row, Cauchy-filter basis row, regular Cauchy-filter row, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, componentwise transport, continuation replay, provenance, and local naming. It receives no selected limit, quotient filter, countable-choice schedule, host equality, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification row, or bridge certificate.*

Proof. Begin with a completion-facing or $\mathsf{Real}^{\uparrow}$-facing read named by P. Carrier projection restricts the consumer to the rows displayed in Definition 5141.1.

Apply Definition 5141.3. The precision request must pass through F, B, G, S, D, R before a regular rational readback is available.

The seal row E is placed after that regular readback route. A consumer therefore sees the terminal $\mathsf{Real}^{\uparrow}$ boundary only after the source filter, basis witness, regularity row, finite windows, dyadic ledger, and regular rational handoff have all been replayed.

The structural rows do not add analytic content. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the displayed route, so selected limits, quotient filters, choice schedules, host equality, ambient real completeness, Lean verification, and bridge checking cannot enter through the seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterModulus}^{\uparrow}$ packet is generated from a CauchyFilter source row, a CauchyFilterBasis row, a RegularCauchyFilter regularity row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5141.1, Definition 5141.2, Definition 5141.3, and Definition 5141.4 seed the finite modulus interface from Cauchy-filter evidence and basis requests to regular rational readback before the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient filters, countable choice, host equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-to-basis routing, regular filter modulus extraction, window and dyadic replay, regular readback, Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.CauchyFilterModulusUp.

Chapter 5142

A Concrete Naming Certificate for $\mathsf{LocallyConvexCompletion}^{\uparrow}$

Following Bourbaki and Schaefer on locally convex completions, the $\mathsf{LocallyConvexCompletion}^{\uparrow}$ packet records the finite BEDC route by which a displayed locally convex source, a finite seminorm family, a Cauchy-filter row, and a uniform-completion handoff reach a completion seal. It sits after the locally convex surface of chapter 3386, the family-seminorm ledger of chapter 5868, the filter-completion route of chapter 2164, the uniform completion packet of chapter 3560, and the Cauchy-filter criterion of chapter 3885. The packet names completion only through displayed finite rows; it is not a Hahn–Banach theorem, not a selected dual, not a host quotient completion, and not an arbitrary-net completeness principle.

Definition 5142.1 (LocallyConvexCompletion carrier)**.** *A* $\mathsf{LocallyConvexCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, S, F, U, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *source row, exposing the vector-space, topological-vector-space, finite seminorm, and convex balanced neighbourhood data accepted by the source.*
2. *S is the finite seminorm-family ledger read through* $\mathsf{SeminormedSpace}^{\uparrow}$*, including the displayed family index, zero, triangle, scalar, induced uniform, and metric handoff rows.*
3. *F is the Cauchy-filter row for the locally convex uniformity generated by S, with its finite base, refinement, and Cauchy-obligation data displayed before completion is requested.*
4. *U is the* $\mathsf{UniformSpaceCompletion}^{\uparrow}$ *handoff row, available only after X, S, F have exposed the same local uniform route.*
5. *E is the completion seal row, recording the accepted locally convex completion name after the Cauchy-filter criterion and uniform-completion handoff have both been displayed.*
6. *H records componentwise* hsame *transport for the locally convex source, seminorm-family ledger, Cauchy-filter row, uniform-completion handoff, completion seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion request through X, S, F, U, E, H, P, N.*

8. *P is the* Pkg *provenance over* LocallyConvexCompletion$^{\uparrow}$, LocallyConvexSpace$^{\uparrow}$, SeminormedSpace$^{\uparrow}$, FrechetCompletion$^{\uparrow}$, UniformSpaceCompletion$^{\uparrow}$, CauchyFilterCompletionCriterion$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{LocallyConvexCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, F, U, E, H, C, P, N$. It has no coordinate for Hahn–Banach separation, a choice-selected dual, a quotient completion in a host category, arbitrary nets without displayed Cauchy data, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocallyConvexCompletionUp.LocallyConvexCompletionTasteGate single carrier alignment`

Theorem 5142.2 (LocallyConvexCompletion seminorm-filter obligations)**.** *For every accepted* LocallyConvexCompletion$^{\uparrow}$ *carrier $L = (X, S, F, U, E, H, C, P, N)$, the local* NameCert$_{\text{LocallyConvexCompletion}^{\uparrow}}$ *surface contains exactly the displayed source row X, finite seminorm-family ledger S, Cauchy-filter row F, uniform-completion handoff U, completion seal E, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local naming N. Every completion-facing read must first expose the finite seminorm-family route and then the Cauchy-filter row generated by that family.*

Proof. Project the carrier of Definition 5142.1. Its public rows are precisely $X, S, F, U, E, H, C, P, N$, so any local NameCert read begins with the displayed locally convex source and the finite seminorm-family ledger.

The finite family induction is over the index row carried by S. The base read exposes the empty or initial finite family as a displayed uniform-neighbourhood request, and the extension step adds one seminorm row by routing zero, triangle, scalar, induced uniform, and metric obligations through the same SeminormedSpace$^{\uparrow}$ ledger.

After the family has been displayed, the Cauchy-filter row F records the completion request against that finite uniform route. The handoff U and seal E are downstream rows, while H, C, P, N transport, replay, source, and name the same packet. Since the excluded Hahn–Banach, dual-choice, host-quotient, arbitrary-net, and ambient-completeness coordinates are absent from the carrier, the listed rows exhaust the obligation surface. □

Theorem 5142.3 (LocallyConvexCompletion Cauchy seal)**.** *Every accepted* LocallyConvexCompletion$^{\uparrow}$ *completion seal factors through the displayed route*

LocallyConvexSpace$^{\uparrow}$, SeminormedSpace$^{\uparrow}$, CauchyFilterCompletionCriterion$^{\uparrow}$, FrechetCompletion$^{\uparrow}$, UniformSpaceCom

Consequently a downstream functional-analysis consumer may cite the locally convex completion only after the finite seminorm-family ledger, Cauchy-filter row, uniform-completion handoff, completion seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start with a public read of the completion seal E. By Definition 5142.1, the seal is not a primitive host point; it is a row reached after X, S, F, U have been displayed.

Use Definition 5142.2. The finite seminorm-family induction supplies the uniform route consumed by F, and the Cauchy-filter criterion keeps the completion demand tied to displayed Cauchy data.

The FrechetCompletion$^{\uparrow}$ and UniformSpaceCompletion$^{\uparrow}$ references are handoff rows rather than replacement completions. They record filter and uniform completion routes already bounded by X, S, F, while H, C, P, N preserve and name the same finite route.

Thus every accepted seal factors through the displayed packet. No row exports Hahn–Banach separation, a choice-selected dual, host quotient equality, arbitrary nets without Cauchy data, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConvexCompletion}^{\uparrow}$ completion read factors through $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{CauchyFilterCompletionCriterion}^{\uparrow}$, $\mathsf{FrechetCompletion}^{\uparrow}$, $\mathsf{UniformSpaceComp}$ componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: locally-convex-space, seminormed-space, frechet-completion, uniform-space-completion, cauchy-filter-completion-criterion: no listed sibling is bijective with $\mathsf{LocallyConvexCompletion}^{\uparrow}$ because this packet jointly carries a locally convex source, a finite seminorm-family ledger, a Cauchy-filter row, a uniform-completion handoff, a completion seal, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConvexCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LocallyConvexCompletion}^{\uparrow}$ packet is generated from a locally convex source row, a finite seminorm-family ledger, a Cauchy-filter row, a UniformSpaceCompletion handoff row, a completion seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConvexCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5142.1, Definition 5142.2, and Definition 5142.3 seed the finite locally convex completion route over source, seminorm-family, Cauchy-filter, uniform-completion, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConvexCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, choice-selected duals, host quotient completion, arbitrary nets without displayed Cauchy data, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite seminorm-family induction, Cauchy-filter routing, uniform-completion handoff, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocallyConvexCompletionUp.

Chapter 5143

A Concrete Naming Certificate for MetaCICNormalEndpointCut$^{\uparrow}$

The MetaCICNormalEndpointCut$^{\uparrow}$ packet realizes the normalization endpoint boundary of section 9386.4 as finite BEDC data. It sits beside the normalization frontier of chapter 892, the critical-path packet of chapter 765, the bounded normal-conversion bridge of chapter 3439, the subject-reduction discharge socket of chapter 719, and the normal-endpoint budget of chapter 4292. The packet records where candidate evidence, a finished-normal endpoint, an obstruction row, and a discharge socket must remain separated before a closed-normal consumer may read the endpoint.

Definition 5143.1 (MetaCICNormalEndpointCut carrier)**.** *A* MetaCICNormalEndpointCut$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, A, B, F, O, D, H, C, P, N)$$

with the following displayed rows:

1. *T is the empty-context typed closed-term row under discussion by the normalization frontier.*
2. *A is the candidate-evidence row, read through* MetaCICNormalizationFrontier$^{\uparrow}$ *before strong-normality or endpoint data may be consumed.*
3. *B is the bounded normal-computation row, recording the finite computation budget and the bounded-normal endpoint read by* BoundedNormalConversionBridge$^{\uparrow}$*.*
4. *F is the finished-normal endpoint row, available only after A and B have both been displayed.*
5. *O is the obstruction row retaining the fact that arbitrary closed typed terms are not admitted to the candidate interface without displayed evidence.*
6. *D is the* SubjectReductionDischargeSocket$^{\uparrow}$ *row, keeping beta-substitution, application-argument, lambda-domain, and Pi-domain discharge obligations separate from finished-normal endpoint data.*
7. *H records componentwise* hsame *transport for the closed-term, candidate, bounded-normal, finished-normal, obstruction, discharge-socket, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a closed-normal endpoint request through $T, A, B, F, O, D, H, P, N$.*
9. *P is the* Pkg *provenance over* MetaCICNormalEndpointCut$^{\uparrow}$*,* MetaCICNormalizationFrontier$^{\uparrow}$*,* MetaCICCriticalPath$^{\uparrow}$*,* BoundedNormalConversionBridge$^{\uparrow}$*,* SubjectReductionDischargeSocket$^{\uparrow}$*,* MetaCICNormalEnd BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointCut}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, A, B, F, O, D, H, C, P, N$. It has no coordinate for evidence-free strong normalization, erased discharge sockets, host conversion equality, quotient syntax equality, global confluence, or completed closed consistency.

Lean4 ✓ `BEDC.Derived.MetaCICNormalEndpointCutUp.MetaCICNormalEndpointCutTasteGate single carrier alignment`

Theorem 5143.2 (MetaCICNormalEndpointCut socket nonescape)**.** *For every accepted* MetaCICNormalEndpointC *carrier $K = (T, A, B, F, O, D, H, C, P, N)$, a consumer that reaches the finished-normal endpoint row F must still retain the obstruction row O and the discharge socket row D. In particular, erasing D before F is read leaves no valid route to the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointCut}^{\uparrow}}$ *surface.*

Proof. Project the carrier of Definition 5143.1. The finished-normal endpoint row F is not stored alone; it is one coordinate of the finite packet $T, A, B, F, O, D, H, C, P, N$.

The candidate and bounded-normal rows provide the positive endpoint route. The row A supplies candidate evidence through the normalization frontier, while B supplies bounded normal computation through the conversion bridge. The endpoint row F is available only after both rows have been displayed.

The obstruction and discharge rows record what remains outside that positive route. The row O blocks a read that treats every empty-context typed closed term as already inside the candidate interface, and D keeps the subject-reduction discharge obligations visible before consistency-facing consumers use the endpoint.

Finally H, C, P, N transport, replay, source, and name the entire packet. If D is erased before F is read, the replay row no longer reaches the displayed discharge socket, so the local NameCert route is incomplete. □

Falsifiable prediction: Any consumer erasing the $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ row before a finished-normal $\mathsf{MetaCICNormalEndpointCut}^{\uparrow}$ endpoint row lacks a valid NameCert route through candidate evidence, bounded-normal endpoint data, obstruction retention, transport, replay, provenance, and local naming.

Independence witness: metacicnormalizationfrontier, metaciccriticalpath, boundednormalconversionbridge, subjectreductiondischargesocket, metacicnormalendpointbudget: no listed sibling is bijective with $\mathsf{MetaCICNormalEndpointCut}^{\uparrow}$ because this packet jointly carries typed closed-term data, candidate evidence, bounded-normal computation, a finished-normal endpoint, an obstruction row, a discharge socket, transport, replay, provenance, and local NameCert data.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointCut}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalEndpointCut}^{\uparrow}$ packet is generated from an empty-context typed closed-term row, candidate evidence, bounded-normal computation, a finished-normal endpoint, obstruction retention, a subject-reduction discharge socket, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointCut}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5143.1 and Definition 5143.2 seed the finite normal-endpoint cut separating candidate evidence, finished-normal endpoint data, obstruction retention, and the subject-reduction discharge socket.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalEndpointCut}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim evidence-free strong normalization, erased discharge sockets, host conversion equality, quotient syntax equality, global confluence, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence exposure, bounded-normal endpoint routing, finished-normal endpoint sealing, obstruction retention, discharge-socket nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICNormalEndpointCutUp.

The ExteriorAlgebra$^{\uparrow}$ hub routes the finite tensor-word source of **??** toward the alternating-form consumers displayed by chapter 136 and chapter 137.

Chapter 5144

A Concrete Naming Certificate for $\mathsf{ExteriorAlgebra}^{\uparrow}$

Following Bourbaki and Mac Lane, the $\mathsf{ExteriorAlgebra}^{\uparrow}$ packet records the alternating algebra generated by a displayed module or vector-space carrier through finite tensor words with signed transposition refusals. It is grounded in the finite tensor-word source of **??**, the bilinear-form boundary of chapter 146, the module carrier of chapter 46, the vector-space specialization of chapter 47, and the quadratic-boundary comparison surface of chapter 147. The packet is a BEDC NameCert carrier for alternating finite words and wedge multiplication; it is not a host quotient algebra, a basis-dependent determinant construction, or an ambient differential-form object.

The $\mathsf{ExteriorAlgebra}^{\uparrow}$ carrier reads its alternating product rows through the tensor-product carrier of chapter 87 before quotient-like exterior relations are refused by the local alternating and signed-transposition ledgers.

Definition 5144.1 (Exterior algebra carrier)**.** *An* $\mathsf{ExteriorAlgebra}^{\uparrow}$ *carrier over a displayed* $\mathsf{Module}^{\uparrow}$ *row* V *and scalar* $\mathsf{Ring}^{\uparrow}$ *row* R *is a finite* BHist *packet*

$$E = (R, V, T, A, S, W, H, C, P, N)$$

with the following rows:

1. *R is the scalar source row and V is the module or vector-space source row whose carried endpoints supply the letters.*
2. *T is the consumed* $\mathsf{TensorAlgebra}^{\uparrow}$ *finite-word row over R and V.*
3. *A is the alternating ledger: whenever two adjacent letters are classifier-equal through the source row, the corresponding repeated-letter word is refused as a nonzero exterior word.*
4. *S is the signed adjacent-transposition ledger, recording that swapping two adjacent carried letters changes the exterior classifier by the displayed sign row.*
5. *W is the wedge product row, formed by concatenating two finite tensor words and then reading the result through the alternating and signed-transposition ledgers.*
6. *H records componentwise* hsame *transport for scalar rows, source letters, tensor words, alternating rows, signed-transposition rows, wedge products, and local classifiers.*
7. *C records displayed* Cont *replay from source endpoints through tensor-word formation, alternating reduction, signed transposition, and wedge multiplication.*

8. *P is the* Pkg *provenance over* ExteriorAlgebra$^\uparrow$, TensorAlgebra$^\uparrow$, BilinForm$^\uparrow$, Module$^\uparrow$, VecSpace$^\uparrow$, Clifford$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^\uparrow}$ *row.*

The carrier exports only alternating finite tensor-word data. It introduces no quotient primitive, no selected basis, no determinant expansion, no ambient graded-commutative algebra object, and no differential-form endpoint.

Theorem 5144.2 (Exterior algebra NameCert obligations)**.** *For every accepted* ExteriorAlgebra$^\uparrow$ *carrier*

$$E = (R, V, T, A, S, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^\uparrow}$ *surface consists exactly of scalar admission through R, module or vector-space endpoint admission through V, finite tensor-word admission through T, alternating repeated-letter refusal through A, signed adjacent-transposition transport through S, wedge multiplication through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier rows of Definition 5144.1. The scalar, source-letter, tensor-word, alternating, signed-transposition, and wedge rows are all displayed before a consumer receives an exterior word.

The tensor source is read through **??**. Thus every exterior word first appears as a finite tensor word, and only then passes through the alternating ledger A and the signed adjacent-transposition ledger S.

The structural rows close the certificate. Transport by H, replay by C, provenance by P, and naming by N preserve exactly the displayed packet. Because the carrier has no quotient coordinate, selected basis, determinant coordinate, host graded algebra, or differential-form endpoint, the listed rows exhaust the local NameCert surface. □

Theorem 5144.3 (Exterior algebra wedge alternation)**.** *Let x and y be carried degree-one letters in the same* ExteriorAlgebra$^\uparrow$ *carrier $E = (R, V, T, A, S, W, H, C, P, N)$. The wedge rows satisfy the displayed alternating boundary*

$$x \wedge x \sim_E 0$$

and the signed transposition boundary

$$x \wedge y \sim_E -y \wedge x$$

whenever the corresponding source-letter rows are accepted by V. These boundaries are read through the finite tensor-word row T, the alternating ledger A, the signed-transposition ledger S, and the wedge row W; they do not use a quotient algebra, a selected basis, or a host determinant identity.

Proof. First read x and y as carried source-letter rows of V. The tensor source T admits the length-two words $x \otimes x$, $x \otimes y$, and $y \otimes x$ as finite tensor words.

For the repeated-letter boundary, apply the alternating ledger A to the adjacent pair x, x. This row records that the repeated-letter word is classified with the zero exterior word before any ambient quotient is available.

For the mixed boundary, apply the signed adjacent-transposition ledger S to the two tensor words $x \otimes y$ and $y \otimes x$. The sign row is part of the same finite packet, so the comparison is a displayed classifier row rather than a host graded-commutativity theorem.

Finally use the wedge row W to identify each length-two tensor word as the product of the corresponding degree-one exterior words. Transport by H and replay by C keep the source letters, tensor words, alternating rows, sign rows, and wedge products inside the same local NameCert packet. □

Theorem 5144.4 (Exterior algebra tensor-product dependency route)**.** *Every* $\mathsf{ExteriorAlgebra}^{\uparrow}$ *wedge read first consumes the alternating quotient-free tensor route from chapter 87 and the bilinear or alternating witness surface from chapter 146. Only after those sibling rows are displayed may the local carrier*

$$E = (R, V, T, A, S, W, H, C, P, N)$$

read its wedge row W *and local* $\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^{\uparrow}}$ *row* N*. The alternation row is a finite classifier row on tensor words, not a quotient of tensors.*

Proof. Project the exterior carrier of Definition 5144.1. The rows R, V, T give the scalar source, the module or vector-space source, and the finite tensor-word source before any wedge product is read.

The tensor-product source of wedge multiplication is the row T. The chapter therefore consumes the sibling tensor route of chapter 87 before W can concatenate tensor words and return an exterior product.

The alternating witness surface is supplied by A and S, with bilinear provenance recorded in P through chapter 146. These rows classify repeated letters and adjacent transpositions as finite packet data; they do not form a tensor quotient.

Finally H, C, P, N transport, replay, source, and name the same dependency route. Thus every accepted wedge read uses the tensor-product and bilinear-form sibling surfaces before the local exterior certificate is read. □

Falsifiable prediction: A consumer requesting an exterior product must expose the tensor-word row, alternating ledger, signed-transposition ledger, wedge row, transport, replay, provenance, and local naming rows; a request that supplies only a host quotient algebra is rejected by this certificate.

Independence witness: tensor-algebra, bilinform, module, vecspace, clifford: $\mathsf{ExteriorAlgebra}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries alternating finite tensor words, repeated-letter refusal, signed adjacent-transposition rows, wedge multiplication, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ExteriorAlgebra}^{\uparrow}$ packet is generated from scalar RingUp rows, ModuleUp or VecSpaceUp endpoint rows, finite TensorAlgebraUp word rows, alternating repeated-letter refusals, signed adjacent-transposition rows, wedge-product rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^{\uparrow}}$, seed)

Scope closed: Definition 5144.1, Definition 5144.2, and Definition 5144.3 seed the finite alternating tensor-word NameCert surface over tensor-algebra, bilinear-form, module, vector-space, and Clifford comparison rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host quotient exterior algebras, determinant-basis formulas, differential-form sheaves, de Rham cohomology, or completed graded algebra structure.

Upgrade path: Obligation closure requires binding the displayed carrier, alternating ledger, signed-transposition ledger, wedge row, transport, replay, provenance, and local naming against their BEDC primitive scope.

The exterior-algebra alternating-form dependency route records the downstream form and cohomology consumers that read wedge rows from the local $\mathsf{ExteriorAlgebra}^{\uparrow}$ packet. It is a consumer edge, not a second exterior-algebra carrier.

Theorem 5144.5 (Exterior algebra alternating-form dependency route)**.** *Every* $\mathsf{DiffForm}^{\uparrow}$ *or* $\mathsf{DeRham}^{\uparrow}$ *consumer that reads alternating form rows, wedge products, or exterior-derivative wedge ladders must first read* $\mathsf{ExteriorAlgebra}^{\uparrow}$ *through* $\mathsf{TensorAlgebra}^{\uparrow}$*, the displayed* $\mathsf{Module}^{\uparrow}$ *or* $\mathsf{VecSpace}^{\uparrow}$ *source rows, the alternating ledger, signed-transposition ledger, wedge row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{ExteriorAlgebra}^{\uparrow}}$ *row. The route exports no host quotient exterior algebra, selected basis, determinant expansion, ambient differential-form sheaf, de Rham quotient class, or completed graded algebra object.*

Proof. Start with the carrier of Definition 5144.1. Its only exterior words are finite $\mathsf{TensorAlgebra}^{\uparrow}$ words over the displayed $\mathsf{Module}^{\uparrow}$ or $\mathsf{VecSpace}^{\uparrow}$ source rows, and its only alternating operations are the repeated-letter ledger A, signed-transposition ledger S, and wedge row W.

Apply the wedge alternation theorem Definition 5144.3. It forces the degree-one repeated-letter and signed-transposition boundaries to be read through T, A, S, W, H, C, P, N rather than through a host quotient algebra or determinant coordinate.

Now inspect the consumers. The $\mathsf{DiffForm}^{\uparrow}$ root threshold of chapter 136 routes form reads through carried form rows and wedge thresholds, while the $\mathsf{DeRham}^{\uparrow}$ chapter chapter 137 consumes those carried $\mathsf{DiffForm}^{\uparrow}$ rows for the exterior-derivative chain-complex boundary. Thus any alternating-form or cohomology-facing read that uses wedge data must pass through the displayed exterior-algebra rows before it reaches the differential-form or de Rham surface.

The excluded objects are absent from all three displayed packets. The exterior carrier contains no host quotient algebra, selected basis, determinant expansion, differential-form sheaf, de Rham quotient class, or completed graded algebra row, and the DiffForm/DeRham consumers read only the routed form and cochain rows. Therefore the alternating-form dependency route is exactly the stated finite NameCert route. □

Chapter 5145

A Concrete Naming Certificate for $\mathsf{FiniteWindowCompactness}^{\uparrow}$

Following Bishop and Bridges on compact metric intervals and constructive finite covers, the $\mathsf{FiniteWindowCompactness}^{\uparrow}$ packet records compactness as explicit dyadic finite-net windows whose radius ledger can be consumed by compact-uniform-continuity. It realizes the finite-window route of chapter 9237 over the compact source of chapter 129, the metric rows of chapter 57, the dyadic tolerance arithmetic of chapter 291, the closed bounded interval route of chapter 1253, the finite lower-bound fold of chapter 4096, and the downstream uniform-modulus handoff of chapter 369. The packet names finite windows and ledger rows only; it does not import open-cover compactness, selected subcovers outside the displayed bundle, quotient point equality, ambient real completeness, Lean verification, or bridge checking.

Definition 5145.1 (FiniteWindowCompactness carrier)**.** *A* $\mathsf{FiniteWindowCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (K, M, D, I, G, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row exposing the compact metric witness through finite inspection data.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to compare each displayed center with a requested point.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger containing the positive mesh scale and its tolerance comparisons.*
4. *I is the closed bounded interval or bounded compact source row when the packet is used over a real interval.*
5. *G is the finite window grid, a displayed* ProbeBundle *of dyadic centers with coverage witnesses.*
6. *L is the finite lower-bound fold row consumed after the window grid has exposed its positive candidate radii.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the tolerance-only modulus budget.*
8. *H records componentwise* hsame *transport for K, M, D, I, G, L, U and the structural rows.*
9. *C records displayed* Cont *replay from compact finite windows to the lower-bound fold and uniform-modulus handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowCompactness}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowCompactness}^{\uparrow}}$ *row.*

The carrier contains no arbitrary open-cover object, no selected subcover outside G, no quotient point equality, no host finite minimum, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness row.*

Lean4 (definition) `BEDC.Derived.FiniteWindowCompactnessUp.FiniteWindowCompactnessUp`
Lean4 ✓ `BEDC.Derived.FiniteWindowCompactnessUp.FiniteWindowCompactnessTasteGate single carrier alignment`

Theorem 5145.2 (FiniteWindowCompactness NameCert obligations)**.** *For every accepted* FiniteWindowCompactn *carrier*

$$W = (K, M, D, I, G, L, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowCompactness}^{\uparrow}}$ *surface consists exactly of compact metric source exposure, metric distance readback, dyadic positive-radius admission, interval-source localization when present, finite window-grid coverage, lower-bound fold handoff, uniform-modulus delivery, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5145.1. The accepted packet exposes only the compact metric source, metric-distance row, dyadic radius ledger, optional interval source, finite grid, fold row, uniform-modulus row, and structural rows.

The grid row G is the finite inspection domain. Each center and coverage witness is read together with D, so the compactness consumer receives a dyadic window bundle rather than an arbitrary open cover.

The lower-bound row L is then read after the window candidates are visible. Its output is the only route into the uniform-modulus handoff U, while H, C, P, N preserve and name the same displayed route. □

Theorem 5145.3 (FiniteWindowCompactness uniform-modulus route)**.** *Every compact-uniform-continuity consumer that reads a* $\mathsf{FiniteWindowCompactness}^{\uparrow}$ *packet reaches* $\mathsf{UniformModulus}^{\uparrow}$ *only through*

$$K \to M \to D \to I \to G \to L \to U \to H \to C \to P \to N.$$

Thus the consumer must first inspect the compact metric source, the metric distance row, the dyadic finite-window ledger, the interval source when present, and the finite lower-bound fold before it receives a tolerance-only uniform-modulus row.

Proof. Start with an accepted carrier $W = (K, M, D, I, G, L, U, H, C, P, N)$. A compactness read begins at K, and any distance comparison needed by the window coverage row must pass through M.

The dyadic ledger D supplies positive mesh and tolerance comparisons. If the source is a closed bounded interval, the interval row I is read as the displayed boundary for the finite dyadic window.

The finite grid G then supplies the only centers and coverage witnesses available to the consumer. The lower-bound fold L reads those finite candidates and delivers the resulting precision to U.

The rows H, C, P, N transport, replay, source, and name the same route. Since no excluded host cover, selected subcover, quotient equality, host minimum, or ambient completeness coordinate occurs in the carrier, the displayed chain exhausts the uniform-modulus access path. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowCompactness}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a MetricUp distance row, a DyadicRatCoreUp radius ledger, an optional closed bounded interval row, finite ProbeBundle window grids, a FiniteLowerBoundFoldUp row, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5145.1, Definition 5145.2, and Definition 5145.3 seed the finite dyadic window route from compact metric data to uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers outside the displayed window, quotient point equality, host finite minima, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source exposure, dyadic radius admission, finite grid coverage, lower-bound fold delivery, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowCo

Chapter 5146

A Concrete Naming Certificate for $\mathsf{Rademacher}^{\uparrow}$

Following Rademacher's differentiability theorem for Lipschitz maps, the $\mathsf{Rademacher}^{\uparrow}$ packet records only the finite dyadic-window derivative-candidate data that a BEDC consumer may inspect before any measure-theoretic almost-everywhere conclusion is visible. It sits over the Lipschitz carrier of chapter 3762, the real-name seal of chapter 38, the dyadic tolerance core of chapter 291, the metric-space carrier of chapter 57, and the absolute-continuity route of chapter 3361. The packet names the finite frontier at which difference quotients, local Lipschitz bounds, dyadic radii, metric neighborhoods, and absolute-continuity control are displayed; it does not claim Lebesgue almost-everywhere differentiability, a measurable exceptional set, a selected derivative, an ambient measure algebra, or completion of a derivative space.

Definition 5146.1 (Rademacher carrier)**.** *A* $\mathsf{Rademacher}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, E, D, M, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LipschitzFunction}^{\uparrow}$ *row, exposing the source and target metric rows, graph ledger, Lipschitz bound, and distance-inequality route inspected by the packet.*
2. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the base point, output windows, and finite difference-quotient values read by downstream Real-facing consumers.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint-radius and tolerance ledger. It records the dyadic radii, halved-radius refinements, and comparison budgets used to build the displayed derivative-candidate windows.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *neighborhood row for the source and target distance requests used by the Lipschitz and difference-quotient checks.*
5. *A is the* $\mathsf{AbsolutelyContinuousFunction}^{\uparrow}$ *control row, recording only the finite interval or curve-window budget that bounds the visible variation consumed by the derivative-candidate packet.*
6. *H records componentwise* hsame *transport for the Lipschitz row, Real seals, dyadic ledger, metric neighborhoods, absolute-continuity control, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Rademacher-facing finite request through D, M, L, A, and finally into the terminal Real seal E.*

8. *P is the* Pkg *provenance over* Rademacher$^{\uparrow}$, LipschitzFunction$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Met$^{\uparrow}$, AbsolutelyContinuousFunction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{Rademacher}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, E, D, M, A, H, C, P, N$. It has no coordinate for Lebesgue almost-everywhere differentiability, measurable null sets, host derivative functions, quotient maps, selected pointwise derivatives, countable-choice schedules, ambient measure completion, or completed function spaces.

Theorem 5146.2 (Rademacher NameCert obligations)**.** *For every accepted* Rademacher$^{\uparrow}$ *carrier*

$$R = (L, E, D, M, A, H, C, P, N),$$

the local NameCert$_{\text{Rademacher}^{\uparrow}}$ *surface consists exactly of carrier admission, Lipschitz-row exposure, Real seal routing, dyadic endpoint-radius accounting, metric-neighborhood inspection, absolute-continuity control, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Carrier admission is projection from Definition 5146.1. The visible analytic rows are L, E, D, M, A, and the visible support rows are H, C, P, N.

The derivative-candidate route is finite. The Lipschitz row supplies the graph and bound data, the dyadic ledger supplies the inspected radii and tolerances, and the metric row supplies the neighborhoods in which those finite requests are read.

The Real and absolute-continuity rows do not add a measure theorem. The Real row seals only the displayed numerical windows, while the absolute-continuity row records the finite variation-control budget used by those windows.

Finally H and C transport and replay the same coordinates, and P, N source and name the packet. Since no other coordinate occurs in the carrier, the NameCert surface exports none of the excluded measurable, derivative-selection, quotient, choice, completion, or ambient almost-everywhere data. □

Lean4 ✓ `BEDC.Derived.RademacherUp.TasteGate.RademacherTasteGate single carrier alignment`

Theorem 5146.3 (Rademacher Lipschitz metric scope)**.** *Every accepted* Rademacher$^{\uparrow}$ *carrier $R = (L, E, D, M, A, H, C, P, N)$ exposes the* LipschitzFunction$^{\uparrow}$ *row L and the* Met$^{\uparrow}$ *row M before any derivative-candidate read is admitted. The exposed scope includes the source and target distance requests, the Lipschitz graph ledger, the finite bound comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert$_{\text{Rademacher}^{\uparrow}}$ *row. It does not include a measure algebra, an almost-everywhere differentiability claim, a host derivative, quotient-point equality, or a completed function space.*

Proof. Project the accepted carrier through Definition 5146.1. The only rows that can supply metric data are the Lipschitz row L and the metric-neighborhood row M.

Apply Definition 5146.2. It lists carrier admission, Lipschitz-row exposure, metric-neighborhood inspection, transport, replay, provenance, and local naming as visible obligations, so every derivative-candidate request must pass through those rows before it reaches any Real-facing seal.

The structural rows H, C, P, N preserve and name the same finite scope. Since the carrier has no coordinate for measure algebras, almost-everywhere assertions, host derivatives, quotient equality, or completed function spaces, none of those objects can be imported through the Lipschitz and metric rows. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier lipschitz metric scope`

Theorem 5146.4 (Rademacher dyadic difference-quotient obligation)**.** *Every accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier exposes its difference-quotient windows through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *readback is consumed. The displayed window records endpoint radii, halved-radius refinements, dyadic comparison budgets, the matching Lipschitz and metric rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming. No consumer may replace that finite window by an unbounded radius schedule, a selected derivative function, a measurable exceptional set, or host real equality.*

Proof. Start with the dyadic row D in Definition 5146.1. It records the endpoint-radius and tolerance ledger used to build the finite derivative-candidate windows.

Use Definition 5146.2. The obligation surface makes dyadic endpoint-radius accounting visible alongside Lipschitz-row exposure, metric-neighborhood inspection, Real seal routing, transport, replay, provenance, and local naming.

The finite-window induction is over the displayed dyadic ledger: each inspected radius and halved-radius refinement is replayed through C and transported through H before the terminal readback row is used. The provenance row P and local certificate N keep the same finite request attached to BHist.

Because no other row appears in the carrier, a consumer cannot obtain an unbounded schedule, selected derivative, exceptional-set algebra, or host real equality from the difference-quotient window. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier dyadic difference quotient obligation`

Theorem 5146.5 (Rademacher Real derivative handoff)**.** *Every Real-facing derivative-candidate read of an accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *only after the finite route through* D, M, L, A, H, C, P, N *has been displayed. The handoff gives downstream consumers regular rational and Real readback for the visible derivative-candidate window, not a Lebesgue almost-everywhere theorem, not a selected derivative, not quotient equality, and not an ambient completeness statement.*

Proof. Project the carrier as $R = (L, E, D, M, A, H, C, P, N)$. The terminal Real seal row is E, and the rows feeding it are the dyadic ledger D, metric row M, Lipschitz row L, and absolute-continuity control row A.

Apply the NameCert obligations from Definition 5146.2. They force each Real-facing request to expose carrier admission, Real seal routing, dyadic accounting, metric inspection, Lipschitz exposure, absolute-continuity control, and the structural rows.

The readback route is finite. Replay through C and transport through H carry the displayed derivative-candidate window to P and N, so the $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives only the named window data.

Thus the handoff is a regular rational and Real readback of a displayed finite packet. It does not assert an almost-everywhere differentiability theorem, select a derivative outside the inspected windows, identify quotient representatives, or complete an ambient derivative space. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier real derivative handoff`

Theorem 5146.6 (Rademacher finite derivative-candidate packet)**.** *Every Real-facing read exported by an accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier factors through the finite route*

$$D, M, L, A \longmapsto E$$

before any derivative-candidate value is consumed. The displayed packet records dyadic radii, metric neighborhoods, Lipschitz distance bounds, absolute-continuity variation control, and terminal Real seals, but it does not assert that the derivative candidate exists on a Lebesgue almost-everywhere set or that the candidate is selected outside the displayed finite windows.

Proof. Begin with an accepted carrier $R = (L, E, D, M, A, H, C, P, N)$. By Definition 5146.2, the only Real-facing row is E, and it is downstream of the dyadic, metric, Lipschitz, and absolute-continuity rows.

The dyadic and metric rows are indispensable because the finite difference-quotient request needs displayed endpoint radii and the source and target neighborhoods in which the Lipschitz comparison is read. The Lipschitz row is indispensable because it supplies the graph and bound ledger for the same finite request.

The absolute-continuity row records the finite variation-control budget attached to the inspected windows. It is not a measure-completion row and does not select derivatives outside those windows.

Transport, replay, provenance, and local naming preserve the same route. Thus every Real-facing derivative-candidate read reaches the seal only through D, M, L, A, with no Lebesgue almost-everywhere theorem, exceptional-set algebra, selected derivative, quotient stream, countable-choice schedule, or ambient completeness coordinate added. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier finite derivative candidate packet`

Theorem 5146.7 (Rademacher difference-window obligation)**.** *Every accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier exposes derivative-candidate reads only through the finite dyadic-radius row D, metric-neighborhood row M, Lipschitz-bound row L, absolute-continuity control row A, terminal Real-seal row E, and the structural rows H, C, P, N already listed in Definition 5146.1. No derivative-candidate read bypasses those rows through a host derivative, a measurable exceptional set, a quotient representative, or an unbounded radius schedule.*

Proof. Project the accepted carrier as $R = (L, E, D, M, A, H, C, P, N)$. The carrier definition lists the Lipschitz row, Real seal, dyadic ledger, metric neighborhoods, absolute-continuity control, componentwise transport, replay, provenance, and local naming as the only visible coordinates.

Apply Definition 5146.2. It makes the local $\mathsf{NameCert}_{\mathsf{Rademacher}^{\uparrow}}$ surface exactly those rows, so a derivative-candidate request has no admissible public coordinate outside $L, E, D, M, A, H, C, P, N$.

Finally use Definition 5146.6. The Real-facing endpoint of the route is reached only after D, M, L, A have supplied the displayed dyadic radii, metric neighborhoods, Lipschitz bounds, and finite variation budget. Thus every candidate read remains inside the finite window and cannot import any excluded host, quotient, measurable, or unbounded schedule datum. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier difference window obligation`

Theorem 5146.8 (Rademacher difference-quotient window stability)**.** *Let*

$$R = (L, E, D, M, A, H, C, P, N)$$

and

$$R' = (L', E', D', M', A', H', C', P', N')$$

be accepted $\mathsf{Rademacher}^{\uparrow}$ *carriers. If D, D', M, M', L, L', A, A', and E, E' are componentwise* hsame *as dyadic endpoint-radius ledgers, metric-neighborhood rows, Lipschitz-bound rows, absolute-continuity control rows, and Real seal rows, then any finite derivative-candidate read exported*

through H, C, P, N and through H', C', P', N' remains the same displayed difference-quotient window.

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier difference quotient window stability`

Proof. Project the two accepted carriers. The only rows that can feed a derivative-candidate read are D, M, L, A before the terminal Real seal E, together with the structural rows H, C, P, N.

Use Definition 5146.2 and Definition 5146.6. These force every finite read through the dyadic radius ledger, the metric neighborhood request, the Lipschitz-bound ledger, and the absolute-continuity control before any Real-facing value is sealed.

Now apply the componentwise hsame assumptions. Since the two packets display the same D, M, L, A rows and the same E seal row, replay through C and C' follows the same finite row inventory. Transport through H and H' preserves that inventory, and P, N and P', N' only source and name the displayed packets.

Therefore the exported difference-quotient window is the same displayed window. The theorem asserts no Lebesgue almost-everywhere differentiability, measurable exceptional set, selected derivative, quotient map, countable choice, ambient measure completion, or completed derivative-space equality. □

Theorem 5146.9 (Rademacher absolute-continuity boundary)**.** *For every accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier $R = (L, E, D, M, A, H, C, P, N)$, the absolute-continuity row A may bound the displayed finite variation window used by the difference-quotient route, but it cannot be read as a measurable exceptional-set row, a Lebesgue almost-everywhere derivative theorem, a selected derivative function, a Lebesgue algebra, quotient equality, countable choice, or host real equality.*

Proof. Begin with the row inventory of Definition 5146.1. Row A is the absolute-continuity control row, and the definition assigns it only the finite interval or curve-window budget consumed by the displayed derivative-candidate packet.

The NameCert obligations keep that interpretation fixed. By Definition 5146.2, A is one of the exposed analytic rows, while H, C, P, N transport, replay, source, and name the same packet without adding coordinates.

Now apply Definition 5146.6 and Definition 5146.7. Both routes place A before the terminal Real seal and inside the finite dyadic and metric window, so it bounds the visible variation request only. Since the carrier contains no measurable exceptional-set row, a.e. derivative row, selected derivative function, Lebesgue algebra, quotient-equality row, choice row, or host equality row, none can be read from A. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier absolute continuity boundary`

Theorem 5146.10 (Rademacher finite derivative-candidate kernel scope)**.** *Every accepted* $\mathsf{Rademacher}^{\uparrow}$ *packet exposes its Lipschitz row L, metric-neighborhood row M, dyadic difference-quotient row D,* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *derivative handoff row E, and absolute-continuity boundary row A before any derivative claim is read. The scoped kernel is exactly the displayed carrier route*

$$L, D, M, A \longmapsto E \longmapsto H, C, P, N,$$

with the finite dyadic windows, metric requests, Lipschitz comparisons, Real-facing readback, absolute-continuity variation budget, componentwise transport, replay, provenance, and local naming all visible in the packet. It exports no Lebesgue almost-everywhere theorem, selected derivative, measurable exceptional set, quotient map, countable-choice schedule, host real equality, or completed derivative space.

Proof. Start from the accepted carrier $R = (L, E, D, M, A, H, C, P, N)$ of Definition 5146.1. The carrier projection exposes the only rows available to a derivative-candidate consumer.

Apply the Lipschitz and metric scope theorem Definition 5146.3. It places L and M before any derivative-candidate read and keeps the metric requests inside the displayed NameCert route.

Next apply the dyadic difference-quotient obligation Definition 5146.4. It forces the finite endpoint-radius ledger, halved-radius refinements, and comparison budgets through D before any regular rational or Real readback is consumed.

Use the Real derivative handoff Definition 5146.5 and the absolute-continuity boundary Definition 5146.9. These place the Real-facing derivative-candidate seal downstream of D, M, L, A and restrict A to finite variation-window control.

Finally replay the structural rows H, C, P, N. The same route is the paper-side counterpart of the existing Lean namespace around `BEDC.Derived.RademacherUp.RademacherCarrier_lipschitz_metric_sco` together with its dyadic, Real-handoff, and absolute-continuity neighbors. Since no carrier coordinate stores the excluded measurable, quotient, selected-derivative, choice, host-equality, or completion data, no derivative claim can be read before the scoped kernel rows are displayed. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier kernel scope`

Theorem 5146.11 (Rademacher scope-closure package)**.** *For every accepted* $\mathsf{Rademacher}^{\uparrow}$ *carrier*

$$R = (L, E, D, M, A, H, C, P, N),$$

the existing Lipschitz metric scope, dyadic difference-quotient obligation, Real derivative handoff, finite derivative-candidate packet, and difference-window stability rows form the scopedClosed *support over* $\mathsf{LipschitzFunction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The scoped package consists exactly of the Lipschitz row L, Real seal row E, dyadic endpoint-radius row D, metric-neighborhood row M, absolute-continuity control row A, transport row H, replay row C, provenance row P, and local naming row N; it exports no full almost-everywhere differentiability theorem, measure theory, selected derivative function, quotient map, countable-choice schedule, ambient Real completeness, or completed derivative space.*

Proof. Start with the Lipschitz and metric scope. The theorem Definition 5146.3 places the Lipschitz row L and metric row M before any derivative-candidate read and keeps their source and target distance requests inside the displayed NameCert route.

Next read the dyadic and Real-facing rows. Theorems 5146.4, 5146.5, and 5146.6 force the endpoint-radius ledger D, regular-rational or Real readback, terminal Real seal E, and absolute-continuity control A to appear as a finite candidate packet before any downstream derivative read is public.

Now use the difference-window rows. Theorems 5146.7 and 5146.8 keep every exported derivative-candidate read inside the displayed D, M, L, A, E window and preserve that window under componentwise hsame transport.

Finally apply the kernel-scope theorem Definition 5146.10. It repacks the same rows through H, C, P, N as the finite scoped route. Since the carrier has no coordinate for full almost-everywhere differentiability, measure algebras, selected derivative functions, quotient maps, countable choice, ambient Real completeness, or completed derivative spaces, the listed rows are exactly the scoped-closure support. □

Lean4 ✓ `BEDC.Derived.RademacherUp.RademacherCarrier scope closure package`

Closure status for $\mathsf{NameCert}_{\mathsf{Rademacher}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Rademacher}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 5146.1, Definition 5146.2, Definition 5146.10, and Definition 5146.11 close the finite derivative-candidate scoped surface over Lipschitz, Real, DyadicRatCore, MetricSpace, absolute-continuity, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{Rademacher}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.RademacherUp.RademacherCarrier\_scope\_closure\_package`

Bridge status: none

Not claimed: This chapter does not close Lebesgue almost-everywhere differentiability, measurable exceptional sets, selected derivatives, quotient maps, countable choice, ambient measure completion, completed derivative spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz-row exposure, Real seal routing, dyadic endpoint-radius accounting, metric-neighborhood inspection, absolute-continuity control, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.RademacherUp.

Constructive story: A Rademacher$^{\uparrow}$ packet is generated from a LipschitzFunction row, RealUp seal row, DyadicRatCore endpoint-radius ledger, MetricSpace neighborhood row, AbsolutelyContinuousFunction control row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5147

A Concrete Naming Certificate for $\mathsf{RegularTopologicalSpace}^{\uparrow}$

Following Kelley and Engelking, the $\mathsf{RegularTopologicalSpace}^{\uparrow}$ packet records the textbook separation surface in which a displayed point and a displayed closed-set row are separated by open neighbourhoods. It sits over the indexed-open carrier of chapter 88, beside the separation-function route of **??**, before the metrization route of chapter 3337, and beside the metric consumer surface of chapter 57. The packet names only a local regularity read over displayed open and closed rows; it does not import host powersets, classical complements, arbitrary normality, selected functions, or a host metric.

Definition 5147.1 (Regular topological space carrier)**.** *A* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, O, F, M, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the open-index classifier, open-neighbourhood membership reads, and finite closure witnesses available to the packet.*
2. *O is the open-neighbourhood row for the displayed point, recording the named open containing the point before any separation request is consumed.*
3. *F is the closed-set row disjoint from the displayed point, presented as a finite ledger over the same topology source rather than as a host complement.*
4. *M is the point-membership row, tying the inspected point to O and excluding it from F by displayed membership and nonmembership reads.*
5. *S is the separation-witness row, recording open rows U, V with the point in U, the closed row F contained in V, and no accepted overlap between U and V.*
6. *H records componentwise* hsame *transport for the topology source, open-neighbourhood row, closed-set row, point-membership row, separation witnesses, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a regularity request through T, O, F, M, S, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularTopologicalSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{UrysohnMetrization}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularTopologicalSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $T, O, F, M, S, H, C, P, N$. It has no coordinate for host complements, external powersets, selected Urysohn functions, normal-space separation, metrization outside a displayed route, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularTopologicalSpaceUp.RegularTopologicalSpaceTasteGate single carrier alignment`

Theorem 5147.2 (Regular topological space NameCert obligations)**.** *For every accepted* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *carrier*

$$R = (T, O, F, M, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularTopologicalSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission through T; open-neighbourhood admission through O; closed-set row admission through F; point-membership and point-nonmembership readback through M; regular separation exactness through S; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5147.1. The accepted packet displays the topology source, the point neighbourhood, the closed-set row, the point-membership row, the separation witness, and the structural rows.

The open-neighbourhood and closed-set obligations are read before separation. The rows O, F, M identify the inspected point, the containing open, and the closed row avoided by that point over the same $\mathsf{Topology}^{\uparrow}$ source.

Regular separation exactness is the row S. It supplies the two separating open rows and their disjointness read, so the certificate has no need for a selected complement or an ambient powerset argument.

The rows H, C, P, N transport, replay, package, and name the same finite route. Since no further coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 5147.3 (Regular topological space neighbourhood separation)**.** *Every neighbourhood-separation read exported by an accepted* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *packet must first display a* $\mathsf{Topology}^{\uparrow}$ *source row, a point-open row, a closed-set row disjoint from the displayed point, and the point-membership row tying them together. Only after those rows are visible may a consumer read the separating open-neighbourhood pair, and the later* $\mathsf{UrysohnLemma}^{\uparrow}$, $\mathsf{UrysohnMetrization}^{\uparrow}$, *or* $\mathsf{Met}^{\uparrow}$ *consumer receives only that displayed finite route.*

Proof. Begin with the topology source T. It supplies indexed open rows, but no regularity assertion is available until the packet displays the point-open row O, the closed-set row F, and the membership row M.

The separation witness is downstream of those rows. The row S records the two separating open neighbourhoods, the containment of the closed-set row in one of them, the displayed point membership in the other, and the disjointness read between the two open rows.

Transport and replay preserve the same order. The componentwise hsame row H and displayed Cont row C reread T, O, F, M, S before provenance or local naming can be inspected.

Thus a later Urysohn, metrization, or metric consumer receives a finite regularity route over the displayed topology source. It receives no host complement, external powerset, selected separating function, normal-space theorem, or host metric from the regular-space packet. □

Falsifiable prediction: Every accepted $\mathsf{RegularTopologicalSpace}^{\uparrow}$ read exposes a $\mathsf{Topology}^{\uparrow}$ source row, an open-neighbourhood row, a closed-set row, a point-membership row, a separation-witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, urysohn-lemma, urysohn-metrization, metric: $\mathsf{RegularTopologicalSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds point-versus-closed-set neighbourhood separation as a local finite packet between topology and metrization consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularTopologicalSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularTopologicalSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5147.1, Definition 5147.2, and Definition 5147.3 seed the finite point-versus-closed-set neighbourhood separation route over topology rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularTopologicalSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close normality, Hausdorff separation, Urysohn function construction, metrization, host powersets, selected complements, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-neighbourhood readback, closed-set row exactness, point-membership stability, separation-witness exactness, transport, replay, provenance, and local naming for BEDC.Derived.RegularTopologicalSpaceUp.

Constructive story: A $\mathsf{RegularTopologicalSpace}^{\uparrow}$ packet is generated from a Topology source row, point open-neighbourhood rows, closed-set rows, point-membership rows, separation witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5148

A Concrete Naming Certificate for $\mathsf{CauchyTailGluing}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailGluing}^{\uparrow}$ packet records the finite tail-gluing surface by which two regular rational names can be read as one Real-facing route after their late tails have been compared. It sits over the finite window discipline of chapter 126, the regular rational readback of chapter 301, the nonquotient comparison boundary of chapter 1820, the dyadic late-tail containment route of chapter 1633, and the terminal real-name seal of chapter 38. The packet names the finite gluing route consumed by L10 Real and Completion consumers; it is not quotient stream equality, a selected common representative, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5148.1 (Cauchy tail gluing carrier)**.** *A* $\mathsf{CauchyTailGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, S, T, n, D, B, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational name rows whose late tails are to be glued.*
2. *S and T are the corresponding* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows through which X and Y are observed.*
3. *n is the displayed tail index. A consumer may inspect only windows whose source positions lie at or beyond n.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed tail index.*
5. *B is the* $\mathsf{DyadicTailBall}^{\uparrow}$ *evidence row stating that the two late-tail windows enter the same dyadic tail ball at the tolerance recorded in D.*
6. *Q is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row, read only after the paired windows, tail index, dyadic ledger, and tail-ball evidence have been displayed.*
7. *R is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the accepted late-tail comparison.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from $X, Y, S, T, n, D, B, Q, R$.*
9. *H records componentwise* hsame *transport for the two regular names, paired stream windows, tail index, dyadic ledger, tail-ball evidence, setoid classifier, glued readback, Real seal, replay, provenance, and local name.*

10. *C records displayed* Cont *replay from a gluing request through the paired stream windows, regular rational rows, dyadic tolerance ledger, tail-ball evidence, setoid classifier, glued readback, and Real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{CauchyTailGluing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{DyadicTailBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailGluing}^{\uparrow}}$ *row.*

The carrier accepts only packets in which both source names are observed through displayed StreamName windows, the same tail index bounds the compared suffixes, the dyadic tail-ball row supplies the finite tolerance evidence, the Cauchy-equivalence classifier consumes that evidence, and the Real seal is reached only through the glued RegSeqRat readback.

The displayed tail index is part of the certificate surface:

$$\begin{aligned}\mathsf{tail}(G) &= n,\\ \mathsf{leftWindow}(G) &= S_{\geq n},\\ \mathsf{rightWindow}(G) &= T_{\geq n}.\end{aligned}$$

The dyadic tail-ball evidence row B is read only at the tolerance recorded in D, and the $\mathsf{Real}^{\uparrow}$ seal row E is unavailable until the route through B, Q, R has been replayed.

Theorem 5148.2 (Cauchy tail gluing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailGluing}^{\uparrow}$ *carrier* $G = (X, Y, S, T, n, D, B, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailGluing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular rational names, paired stream windows, displayed tail index, dyadic ledger, tail-ball evidence, setoid classifier, glued readback, Real seal, transport, replay, provenance, and local naming rows;*

2. *classifier discipline, requiring Q to read the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison only after S, T, n, D, B have been displayed;*

3. *tail transport, requiring componentwise* hsame *transport in H to preserve the same tail index and the same paired suffix windows before the glued readback R is consumed;*

4. *Real seal compatibility, requiring E to remain downstream of the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *row R and the classifier route Q;*

5. *ledger nonescape, excluding quotient stream equality, selected common representatives, countable choice, host real equality, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as certificate data.*

No gluing obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.CauchyTailGluingUp.CauchyTailGluingCarrier namecert obligations`

Proof. Project the carrier of Definition 5148.1. The projection exposes the two regular rational names X, Y, their stream windows S, T, the displayed tail index n, the dyadic ledger D, the tail-ball evidence B, the setoid classifier Q, the glued readback R, the Real seal E, and the structural rows H, C, P, N.

The classifier obligation is downstream of the finite tail data. A comparison request first reads the paired windows at the displayed suffix, then consumes the dyadic tolerance ledger and tail-ball evidence before the $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ row Q is available.

Transport preserves the same suffix route. The hsame row H acts componentwise on $X, Y, S, T, n, D, B, Q, R, E$, so a transported packet cannot replace the displayed tail index by a hidden tail selector or move the comparison outside the recorded windows.

Finally, the Real seal is terminal. The row E is reached through the glued RegSeqRat readback R after the classifier route has been replayed, and the carrier contains no coordinate for quotient stream equality, a selected representative, countable choice, host real equality, trichotomy, or ambient completeness. □

Theorem 5148.3 (Cauchy tail gluing Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyTailGluing}^{\uparrow}$ *packet reads the finite route*

$$X, Y, S, T, n, D, B \longmapsto Q \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before consuming a glued real-name seal. Thus the Real-facing row is compatible with the StreamName windows, RegSeqRat readbacks, Cauchy-equivalence classifier, and DyadicTailBall evidence already displayed by the packet, and it exports no quotient stream equality, selected common representative, countable-choice selector, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a Real-facing read. Carrier admission gives only the terminal row E, and the definition places E after the two regular rational source rows, their stream windows, the displayed tail index, the dyadic ledger, the tail-ball evidence, the setoid classifier, and the glued readback.

The replay row C therefore sends the consumer through the same finite path. The classifier row Q receives exactly the dyadic tail-ball evidence B attached to D, and R repacks the accepted comparison as the only glued $\mathsf{RegSeqRat}^{\uparrow}$ row.

The remaining structural rows transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded quotient, choice, host-equality, trichotomy, or completeness data, the terminal Real seal cannot export them. □

Falsifiable prediction: Every $\mathsf{CauchyTailGluing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, paired StreamName suffix windows, one displayed tail index, a DyadicRatCore tolerance ledger, DyadicTailBall evidence, a CauchyEquivalenceSetoid classifier row, a glued RegSeqRat readback, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, cauchy-equivalence-setoid, dyadic-tail-ball, real: $\mathsf{CauchyTailGluing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular names, paired suffix windows, one displayed tail index, dyadic tail-ball evidence, a setoid classifier, a glued readback, and a Real seal in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyTailGluing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, paired StreamName suffix windows, one displayed tail index, a DyadicRatCore tolerance ledger, DyadicTailBall evidence, a CauchyEquivalenceSetoid classifier row, a glued RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5148.1, Definition 5148.2, and Definition 5148.3 seed the finite tail-gluing surface for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected common representatives, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier discipline, tail transport, Real seal compatibility, ledger nonescape, replay, provenance, and local naming for BEDC.Derived.CauchyTailGluingUp.

Chapter 5149

A Concrete Naming Certificate for $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$ packet records the finite route by which an explicit Bishop Cauchy criterion becomes completion-facing BEDC data. It sits between the criterion carrier of chapter 3576, the modulus family of chapter 5887, the left-exact completion boundary of chapter 5888, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38. The chapter names the criterion-to-completion handoff as finite BHist data; it does not assert quotient stream equality, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5149.1 (Bishop Cauchy criterion completion carrier)**.** *A* $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, M, W, R, D, E, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *row, naming the finite Cauchy criterion request that will be consumed by completion-facing data.*
2. *M is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *row, recording the explicit modulus schedules and their precision requests for the same criterion.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source sequence, modulus requests, and finite tail comparisons are inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windows in W and the tolerance schedule in M.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the criterion and readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, M, W, R, D have been displayed.*
7. *L is the* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *handoff row, recording that completion-facing finite-limit reads consume the same criterion, modulus, window, readback, tolerance, and real-seal rows.*
8. *H records componentwise* hsame *transport for $Q, M, W, R, D, E, L, C, P, N$.*

9. *C records displayed* Cont *replay from a completion request through the criterion, modulus schedule, stream windows, rational readback, dyadic tolerance ledger, real seal, and left-exactness handoff.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$, $\mathsf{BishopCauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, W, R, D, E, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected Cauchy representatives, selected limits, countable choice, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyCriterionCompletionUp.BishopCauchyCriterionCompletionTasteGate single carrier alignment`

Theorem 5149.2 (Bishop Cauchy criterion completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$ *carrier*

$$K = (Q, M, W, R, D, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the Bishop Cauchy criterion row, explicit Cauchy modulus schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure,* CauchyCompletionLeftExactn *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing read must expose Q, M, W, R, D, E, L before H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5149.1. The analytic rows are exactly Q, M, W, R, D, E, L, and the structural rows are exactly H, C, P, N.

The source criterion row Q supplies the Bishop Cauchy request, while the modulus row M makes the precision schedule explicit. The finite windows W are therefore the only observation surface through which the criterion can be read.

Next the readback and tolerance rows are forced through R and D. The terminal real seal E is downstream of those rational and dyadic rows, so no real-facing consumer can bypass the displayed criterion data.

Finally the completion handoff L records the left-exact completion-facing use of the same route, and H, C, P, N transport, replay, source, and name it. Since no excluded host object is present as a carrier coordinate, these rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}$ packet is generated from a BishopCauchyCriterion row, CauchyModulusSpace schedules, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, a CauchyCompletionLeftExactness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 5149.1 and Definition 5149.2 seed the finite criterion-to-completion route over the displayed Bishop, Cauchy, rational, dyadic, Real, and Completion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyCriterionCompletion}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.BishopCauchyCriterionCompletionUp.BishopCauchyCriterionCompletionTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected Cauchy representatives, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exposure, modulus schedule exposure, finite-window exposure, RegSeqRat readback exactness, DyadicRatCore tolerance exposure, Real seal nonescape, completion handoff preservation, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyCriterionCompletionUp.

Chapter 5150

A Concrete Naming Certificate for $\mathsf{SignedDecimalExpansion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{SignedDecimalExpansion}^{\uparrow}$ packet presents a real-facing decimal datum by a displayed sign row, finite digit-window rows, a carry ledger, the unsigned $\mathsf{DecimalExpansion}^{\uparrow}$ prefix surface of chapter 1417, the normalized real-decimal route of chapter 3952, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The point is not to select an infinite decimal expansion for a real. It is to name the finite BEDC certificate by which a signed decimal window can be normalized, replayed as a $\mathsf{RegSeqRat}^{\uparrow}$ route, and then handed to $\mathsf{Real}^{\uparrow}$ without quotient representatives, trichotomy, endpoint choice, or ambient completeness.

Definition 5150.1 (Signed decimal expansion carrier)**.** *A* $\mathsf{SignedDecimalExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, D, W, A, L, U, Z, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the sign row, recording either a nonnegative read or a negative read of the same finite magnitude surface.*
2. *D is the finite digit ledger, listing the inspected base-ten digits and the distinguished decimal point window.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the digit ledger is read.*
4. *A is the unsigned* $\mathsf{DecimalExpansion}^{\uparrow}$ *magnitude row, exposing the finite decimal-prefix carrier and its dyadic place-value comparison.*
5. *L is the carry ledger, recording all local carry and borrow moves used inside the inspected window.*
6. *U is the normalized* $\mathsf{RealDecimalNormalForm}^{\uparrow}$ *row, reached only after S, D, W, A, L have been displayed.*
7. *Z is the zero-window boundary row, identifying the unique displayed route by which the positive and negative presentations of zero agree.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *replay row obtained from the signed normalized windows and the dyadic comparison data.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from R.*

10. *H records componentwise* hsame *transport for $S, D, W, A, L, U, Z, R, E, C, P, N$.*
11. *C records displayed* Cont *replay from a signed decimal request through digit windows, carry normalization, regular rational readback, and the Real seal.*
12. *P is the* Pkg *provenance over* SignedDecimalExpansion$^\uparrow$, DecimalExpansion$^\uparrow$, RealDecimalNormalForm$^\uparrow$, StreamName$^\uparrow$, DyadicRatCore$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{SignedDecimalExpansion}^\uparrow}$ *row.*

The classifier compares accepted signed decimal packets only through $S, D, W, A, L, U, Z, R, E, H, C, P, N$. It has no coordinate for a chosen infinite decimal stream, a quotient real representative, an endpoint convention selector, trichotomy, countable choice, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.SignedDecimalExpansionUp.SignedDecimalExpansionTasteGate single carrier alignment`

Theorem 5150.2 (Signed decimal expansion NameCert obligations)**.** *For every accepted* SignedDecimalExpansion$^\uparrow$ *carrier*

$$K = (S, D, W, A, L, U, Z, R, E, H, C, P, N),$$

the local NameCert$_{\text{SignedDecimalExpansion}^\uparrow}$ *surface consists exactly of sign exposure, finite digit-window exposure, unsigned* DecimalExpansion$^\uparrow$ *magnitude exposure, carry-ledger normalization,* RealDecimalNormalForm$^\uparrow$ *normalized-window exposure, zero-window agreement,* RegSeqRat$^\uparrow$ *replay, terminal* Real$^\uparrow$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every real-facing consumer must read $S, D, W, A, L, U, Z, R, E$ before any structural row is consumed.*

Proof. Project the carrier of Definition 5150.1. The analytic rows are exactly $S, D, W, A, L, U, Z, R, E$, while H, C, P, N are the structural transport, replay, provenance, and naming rows.

The unsigned magnitude row A is read only after the sign and finite digit-window rows S, D, W. Thus the signed presentation cannot hide a second decimal magnitude or an infinite stream behind the sign.

The carry ledger L and normalized row U then force all local carry and borrow moves to be replayed inside the same finite window. The zero boundary Z identifies the only displayed agreement route between the two signed presentations of zero.

Finally R and E are downstream rows. The regular rational replay must be exposed before the terminal Real seal, and no carrier coordinate remains for quotient representatives, endpoint choice, trichotomy, choice schedules, or ambient completeness. □

Theorem 5150.3 (Signed decimal expansion decimal normalization route)**.** *Every accepted* SignedDecimalExpansio *packet routes its signed digit window through the unsigned* DecimalExpansion$^\uparrow$ *magnitude row and the* RealDecimalNormalForm$^\uparrow$ *normalization row before any* RegSeqRat$^\uparrow$ *replay or* Real$^\uparrow$ *seal can be read. In the notation of Definition 5150.1, every consumer of R or E first consumes*

$$S, D, W, A, L, U, Z.$$

Consequently the sign row and carry ledger are normalized as finite BEDC data, not by selecting a preferred infinite decimal expansion or an endpoint convention.

Proof. Begin with the carrier ordering in Definition 5150.1. The rows S, D, W expose the signed finite window, and A places its magnitude inside the existing DecimalExpansion$^\uparrow$ finite-prefix surface.

The carry ledger L is replayed before U is public. Since U is the $\mathsf{RealDecimalNormalForm}^{\uparrow}$ row, every normalized decimal read is attached to the displayed carry ledger rather than to a detached stream convention.

The rows R and E occur only after Z has recorded the zero-window boundary. Thus downstream consumers receive a normalized finite signed route, and the packet exports no shortcut from raw signed digits to a regular rational or real-facing value. □

Theorem 5150.4 (Signed decimal expansion Real seal handoff)**.** *For every accepted* $\mathsf{SignedDecimalExpansion}^{\uparrow}$ *packet, the terminal* $\mathsf{Real}^{\uparrow}$ *seal is reached only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *replay row assembled from the signed normalized windows. The handoff factors through the displayed chain*

$$\mathsf{SignedDecimalExpansion}^{\uparrow} \longrightarrow \mathsf{DecimalExpansion}^{\uparrow} \longrightarrow \mathsf{RealDecimalNormalForm}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

It does not export quotient real equality, a selected decimal representative, trichotomy, countable choice, endpoint uniqueness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read R through the normalized-window route of Definition 5150.3. The signed carrier has already exposed the sign row, finite digit ledger, carry ledger, unsigned magnitude row, normalized row, and zero boundary before R can be consumed.

The terminal seal row E is downstream of R. Hence a Real-facing consumer sees only the regular rational replay produced by the signed normalized windows.

The nonescape clause follows from the carrier coordinates in Definition 5150.1. None of them is a quotient representative, selected decimal stream, trichotomy decision, countable-choice schedule, endpoint convention selector, or completeness witness. □

Closure status for $\mathsf{NameCert}_{\mathsf{SignedDecimalExpansion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SignedDecimalExpansion}^{\uparrow}$ packet is generated from a sign row, finite decimal digit windows, an unsigned DecimalExpansion magnitude row, a finite carry ledger, a RealDecimalNormalForm normalized row, a zero-window boundary, RegSeqRat replay, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SignedDecimalExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5150.1, Definition 5150.2, Definition 5150.3, and Definition 5150.4 seed the finite signed decimal presentation and its RegSeqRat-to-Real handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SignedDecimalExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected infinite decimal streams, endpoint uniqueness, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sign transport, carry-ledger normalization, zero-window agreement, RegSeqRat replay, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SignedDecimalExpansionUp.

Chapter 5151

A Concrete Naming Certificate for $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ packet records a displayed Cauchy certificate as finite BEDC NameCert data before any Real seal or completion-facing read is consumed. It sits between the criterion surface of chapter 3576, the criterion-to-completion route of chapter 5149, the reusable tail certificate of chapter 1230, the modulus schedule surface of chapter 5887, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion consumers a single finite display entry point; it is not a quotient stream equality theorem, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5151.1 (Bishop Cauchy display-certificate carrier)**.** *A* $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, W, T, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{BishopCauchyCriterion}^{\uparrow}$ *row naming the finite Cauchy criterion whose display is being certified.*
2. *M is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *row recording the displayed modulus schedules and precision requests used by Q.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source sequence, modulus requests, and tail comparisons are inspected.*
4. *T is the* $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ *row witnessing that the displayed windows supply the shared tail certificate required by Real and Completion consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from W, T, and the modulus schedule in M.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, comparison budgets, and slack rows used by the displayed criterion and tail handoff.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, M, W, T, R, D have been displayed.*
8. *H records componentwise* hsame *transport for Q, M, W, T, R, D, E and the local naming row.*

9. *C records displayed* Cont *replay from a display request through the criterion, modulus schedule, stream windows, tail certificate, rational readback, dyadic tolerance ledger, and Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$, $\mathsf{BishopCauchyCriterion}^{\uparrow}$, BishopCauchyCriterionCo CauchyModulusSpace$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, W, T, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, selected limits, countable choice, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BishopCauchyDisplayCertificateUp.BishopCauchyDisplayCertificateTasteGate`
`single carrier alignment`

Theorem 5151.2 (Bishop Cauchy display-certificate NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ *carrier*

$$B = (Q, M, W, T, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}}$ *surface consists exactly of carrier admission for the Bishop Cauchy criterion row, explicit Cauchy modulus schedules, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegularCauchyTailCertificate}^{\uparrow}$ *tail handoff,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must expose Q, M, W, T, R, D, E before H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5151.1. The analytic display rows are exactly Q, M, W, T, R, D, E, and the structural support rows are exactly H, C, P, N.

The criterion row Q supplies the Bishop Cauchy request, while M makes the precision schedule explicit. The stream row W is therefore the only observation surface through which the sequence and modulus requests can be read.

The tail certificate T then binds the displayed windows to the shared regular Cauchy tail route. The readback row R and dyadic ledger D keep the rational and tolerance data visible before the terminal Real seal E is available.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no excluded host object is present as a carrier coordinate, these rows exhaust the local NameCert obligations. □

Theorem 5151.3 (Bishop Cauchy display-certificate tail handoff)**.** *Every Real or Completion consumer accepted by a* $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ *carrier reads the shared tail certificate through the finite route*

$$\mathsf{BishopCauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulusSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegularCauchyTailCertificate}^{\uparrow} \longmapsto \mathsf{RegSeq}$$

together with H, C, P, N. Consequently the consumer receives the displayed criterion, modulus schedule, finite windows, tail certificate, regular rational readback, dyadic tolerance ledger, and terminal Real seal, but it receives no selected limit, quotient stream equality, countable-choice schedule, trichotomy row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with a Real or Completion read of B. The source criterion and precision schedule appear only as Q and M, so the read must expose them before any window or tail row is available.

The row W supplies the finite inspection surface. The handoff row T is downstream of those windows and is the only coordinate that can expose a regular Cauchy tail certificate.

The rows R, D, E then perform the regular rational readback, dyadic tolerance accounting, and terminal Real sealing in order. The support rows H, C, P, N preserve that same route without adding analytic content.

Because quotient streams, selected limits, countable choice, trichotomy, and ambient completeness are absent from the carrier, no consumer can obtain them from the displayed tail handoff. □

Lean4 ✓ `BEDC.Derived.BishopCauchyDisplayCertificateUp.BishopCauchyDisplayCertificateCarrier tail handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ packet is generated from a BishopCauchyCriterion row, CauchyModulusSpace schedules, StreamName windows, a RegularCauchyTailCertificate handoff, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5151.1, Definition 5151.2, and Definition 5151.3 seed the finite display-certificate route for the L10 Real and Completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected Cauchy representatives, selected limits, countable choice, trichotomy, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion exposure, modulus schedule exposure, finite-window exposure, tail-certificate handoff, RegSeqRat readback exactness, DyadicRatCore tolerance exposure, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyDisplayCertificateUp.

Chapter 5152

A Concrete Naming Certificate for $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$ packet records the finite route by which a totally bounded separated metric completion exposes sequential compactness through displayed subsequence windows. It sits between the total-bounded completion route of chapter 1573, the separated completion comparison of chapter 1612, the compact metric completion seal of chapter 1687, the finite sequence-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a completion-facing subsequence certificate; it is not open-cover compactness, a quotient completion, a selected limit-point operation, countable choice, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5152.1 (Sequentially compact completion carrier)**.** *A* $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, S, K, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{TotallyBoundedCompletion}^{\uparrow}$ *row. It records the source metric carrier, finite-net completion route, net refinement ledger, Cauchy-filter basis, dense embedding, and uniformly continuous handoff consumed by the sequential compactness request.*
2. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *row. It compares completed reads only through the displayed zero-distance and separated-equality route.*
3. *K is the* $\mathsf{CompactMetricCompletion}^{\uparrow}$ *row. It supplies the complete totally bounded metric completion surface from which compact-facing consumers may request finite sequence data.*
4. *W is the subsequence-window ledger. It stores the inspected* $\mathsf{StreamName}^{\uparrow}$ *windows, the monotone selected-index rows, the finite tail requests, and the bounded-window evidence used by the compact metric completion surface.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the selected windows, rational tail budgets, and completion-facing Cauchy data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after T, S, K, W, R have been displayed.*

7. *H records componentwise* hsame *transport for the total-bounded completion route, separated completion comparison, compact metric completion surface, subsequence windows, regular rational readback, Real seal, continuation replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite sequence request through T, S, K, W, R, E.*
9. *P is the* Pkg *provenance over* SequentiallyCompactCompletion$^{\uparrow}$, TotallyBoundedCompletion$^{\uparrow}$, SeparatedCompletion CompactMetricCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{SequentiallyCompactCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, K, W, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient representatives, selected limit points outside W, countable-choice schedules, host sequence equality, host real equality, or detached ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SequentiallyCompactCompletionUp.SequentiallyCompactCompletionTasteGate`
`single carrier alignment`

Theorem 5152.2 (SequentiallyCompactCompletion NameCert obligations)**.** *For every accepted* SequentiallyCompactCompletion$^{\uparrow}$ *carrier*

$$Q = (T, S, K, W, R, E, H, C, P, N),$$

the local NameCert$_{\text{SequentiallyCompactCompletion}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the total-bounded completion route, separated completion comparison, compact metric completion surface, subsequence-window ledger, regular rational readback, Real seal, componentwise transport, displayed replay, provenance, and local naming row;*
2. *completion-route exposure, requiring every sequential compactness request to read T and S before it may inspect the compact metric completion row K;*
3. *subsequence-window locality, requiring selected indices, finite tails, source windows, and compactness evidence to be precisely the displayed rows in W;*
4. RegSeqRat$^{\uparrow}$ *handoff exactness, requiring every convergence read to enter R only from the windows and tail budgets stored in W and transported through H;*
5. Real$^{\uparrow}$ *seal nonescape, requiring the terminal limit read to pass through E after R and forbidding quotient streams, host real equality, external selected limits, countable choice, and detached ambient completeness;*
6. *provenance and naming closure, requiring P and N to name only the displayed packet and no principle absent from its rows.*

No NameCert obligation is stored outside these displayed rows.

Proof. Carrier admission is projection from Definition 5152.1. The projected rows are exactly the total-bounded completion route T, separated comparison S, compact metric completion surface K, subsequence-window ledger W, regular rational readback R, Real seal E, and structural rows H, C, P, N.

Completion-route exposure follows the sibling order. The finite total-bounded completion route of chapter 1573 is read first through T, and the separated completion comparison of chapter 1612 is read through S before the compact metric completion surface K can be inspected.

Subsequence-window locality is internal to W. The packet stores the $\mathsf{StreamName}^{\uparrow}$ windows, selected-index rows, finite tail requests, and bounded-window evidence that the compact metric completion surface uses; there is no separate coordinate from which an external subsequence or selected limit point could be read.

The convergence handoff is then the row R. It accepts only the selected windows and rational tail budgets displayed by W, with componentwise transport supplied by H, so the handoff cannot be replaced by quotient stream equality or host sequence equality.

Finally, the terminal read is sealed by E, while C, P, N replay, source, and name the same route. Since the carrier has no coordinate for open-cover compactness, quotient completion, countable choice, host real equality, external selected limits, or detached ambient completeness of $\mathsf{Real}^{\uparrow}$, none of those data can be exported by the local NameCert surface. □

Theorem 5152.3 (SequentiallyCompactCompletion subsequence seal)**.** *Let $Q = (T, S, K, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$ *carrier. Every sequential-compactness-facing consumer factors through the finite route*

$$T, S \longmapsto K \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the consumer receives total-bounded completion data, separated completion comparison, compact metric completion, displayed subsequence windows, regular rational readback, Real seal, transport, replay, provenance, and local naming before it can read a completed subsequence limit.

Proof. Project the consumer read from the carrier. By Definition 5152.1, the only completion-facing rows available before the compact metric surface are T and S.

The compact metric completion row K is downstream of those completion rows. It exposes finite completion compactness only as the route already named in chapter 1687, so the consumer cannot bypass T and S by invoking an open-cover compactness theorem or a quotient completion.

The subsequence information is then exactly the ledger W. The regular rational readback R and terminal Real seal E are reachable only from the windows, selected indices, and tail budgets stored in that ledger.

Transport and replay through H and C preserve the same order, and P, N name only that packet. This proves the displayed factorization and excludes external selected limits, countable choice, host sequence equality, host real equality, and detached ambient Real completeness. □

Falsifiable prediction: Every $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$ completed-subsequence read factors through TotallyBoundedCompletion, SeparatedCompletion, CompactMetricCompletion, StreamName subsequence windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: totallyboundedcompletion, separatedcompletion, compactmetriccompletion, streamname, regseqrat, real: $\mathsf{SequentiallyCompactCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries completion route exposure, separated comparison, compact metric completion, displayed subsequence windows, rational readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyCompactCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentiallyCompactCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5152.1, Definition 5152.2, and Definition 5152.3 seed the finite route from total-bounded separated completion data to displayed subsequence compactness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentiallyCompactCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient completion, selected limit points outside the displayed windows, countable choice, host sequence equality, host real equality, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-route exposure, subsequence-window locality, RegSeqRat handoff exactness, RealUp seal nonescape, provenance and naming closure, and TasteGate field faithfulness for BEDC.Derived.SequentiallyCompactCompletionUp.

Constructive story: A $\mathrm{SequentiallyCompactCompletion}^{\uparrow}$ packet is generated from a TotallyBoundedCompletion route, a SeparatedCompletion comparison, a CompactMetricCompletion surface, StreamName subsequence windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5153

A Concrete Naming Certificate for $\mathsf{LocaleSpace}^{\uparrow}$

Following Johnstone and constructive locale theory, the $\mathsf{LocaleSpace}^{\uparrow}$ packet records the point-free topology surface in which open observations are primary and point carriers are optional downstream reads. It sits after the indexed-open topology of chapter 88, the meet–join order interface of chapter 55, the Boolean endpoint discipline of chapter 3733, and the displayed membership surface of chapter 3668. The packet names finite frame rows, cover-refinement rows, and topology handoff rows without importing host powersets, quotient point spaces, selected points, excluded middle over opens, or choice over covers.

Definition 5153.1 (Locale space carrier)**.** *A* $\mathsf{LocaleSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, J, M, T, CV, B, S, H, R, P, N)$$

with the following displayed rows:

1. *O is the open-index carrier row, read point-free as the visible open observation surface.*
2. *J is the finite-join row, recording displayed finite unions of open indices.*
3. *M is the finite-meet row, recording displayed finite intersections of open indices.*
4. *T is the top and bottom row, naming the maximal and empty opens through finite BEDC observations.*
5. *CV is the cover-refinement row, recording when one finite open family refines another.*
6. *B is the* $\mathsf{BooleanAlgebra}^{\uparrow}$ *boundary row used only for clopen or complemented subledgers; it does not make the whole locale Boolean.*
7. *S is the* $\mathsf{SetLike}^{\uparrow}$ *membership-read row for consumers that later inspect a displayed point or probe, not a source of points for the locale itself.*
8. *H records componentwise* hsame *transport for the open carrier, lattice operations, cover row, Boolean boundary, membership-read row, replay, provenance, and local naming.*
9. *R records displayed* Cont *replay from a locale request through $O, J, M, T, CV, B, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocaleSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocaleSpace}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, J, M, T, CV, B, S, H, R, P, N$. It has no coordinate for host powersets, point-set extensionality, quotient spaces, selected points, choice-indexed covers, or a global Boolean algebra of all opens.

Theorem 5153.2 (Locale space NameCert obligations)**.** *For every accepted* $\mathsf{LocaleSpace}^{\uparrow}$ *carrier*

$$L = (O, J, M, T, CV, B, S, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocaleSpace}^{\uparrow}}$ *surface consists exactly of open-index admission through O; finite joins through J; finite meets through M; top and bottom open rows through T; cover-refinement exactness through CV; clopen boundary reads through B when a complemented subledger is displayed; point or probe membership reads through S only after the point-free rows are exposed; componentwise* hsame *transport through H; displayed* Cont *replay through R; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5153.1. The rows O, J, M, T, CV display the point-free open surface before any point-facing or Boolean-facing consumer is consulted.

The Boolean boundary row B is local to complemented subledgers. It may certify a clopen or complemented finite fragment, but it cannot replace the locale carrier by a global $\mathsf{BooleanAlgebra}^{\uparrow}$ object because the carrier has no coordinate for excluded middle over all opens.

The membership row S is downstream of the open rows. A point or probe consumer may read displayed membership only after the locale has exposed the open-index, lattice, and cover-refinement rows.

Finally, H, R, P, N transport, replay, source, and name the same route. Since no carrier coordinate supplies host powersets, quotient point spaces, selected points, or choice-indexed covers, the listed rows exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.LocaleSpaceUp.LocaleSpaceTasteGate single carrier alignment`

Theorem 5153.3 (Locale space finite-meet cover route)**.** *Every accepted* $\mathsf{LocaleSpace}^{\uparrow}$ *consumer that compares a finite meet of opens against a cover must route the comparison through*

$$O, J, M, T \longmapsto CV \longmapsto B, S, H, R, P, N.$$

Thus a $\mathsf{Topology}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{BooleanAlgebra}^{\uparrow}$, *or* $\mathsf{SetLike}^{\uparrow}$ *consumer may inherit only the displayed open-index carrier, finite meet and join rows, top and bottom rows, cover-refinement row, optional complemented-subledger boundary, displayed membership reads, transport, replay, provenance, and local name. The route exports no host powerset of opens, no quotient point space, no selected point, no choice-indexed cover, and no global Boolean algebra of all opens.*

Proof. Start with the point-free open carrier O. The finite-meet request is read through M, while the finite-join and endpoint rows J, T keep the comparison inside the same lattice-facing open ledger.

The cover comparison is then the row CV. It refines finite families of displayed open indices and therefore consumes the open carrier and lattice rows before any complemented or point-facing read is available.

Only after that route may a consumer inspect B or S. The Boolean boundary row can name complemented finite fragments, and the membership row can serve a displayed point or probe, but neither row supplies the source carrier.

The rows H, R, P, N preserve the same route by transport, replay, provenance, and naming. Hence the finite-meet cover comparison is point-free until the optional downstream reads are explicitly displayed. □

Falsifiable prediction: Every accepted $\mathsf{LocaleSpace}^{\uparrow}$ read exposes open-index admission, finite joins, finite meets, top and bottom opens, cover-refinement, optional complemented-subledger boundary, displayed membership reads, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a topology, lattice, Boolean, or setlike consumer may use a locale claim.

Independence witness: topology, lattice, booleanalgebra, setlike: $\mathsf{LocaleSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds point-free open-index, finite lattice, cover-refinement, optional clopen-boundary, and downstream membership rows in a single finite NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocaleSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocaleSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5153.1, Definition 5153.2, and Definition 5153.3 seed the point-free open-index, finite lattice, and cover-refinement route over topology, lattice, Boolean-boundary, setlike, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocaleSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host powerset of opens, quotient point-space representation, selected points, choice-indexed covers, spatiality, global Boolean algebra of opens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite join and meet exactness, endpoint open exactness, cover-refinement stability, complemented-boundary restriction, membership-read boundary, transport, replay, provenance, and local naming for BEDC.Derived.LocaleSpaceUp.

Constructive story: A $\mathsf{LocaleSpace}^{\uparrow}$ packet is generated from an open-index carrier row, finite join and meet rows, top and bottom open rows, cover-refinement rows, optional Boolean boundary rows, setlike membership-read rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{DiniUniformMonotone}^{\uparrow}$ packet routes compact-metric Dini data through chapter 129, chapter 1387, and chapter 5795 before the child files expose the finite uniform-modulus and Real-seal route.

Chapter 5154

A Concrete Naming Certificate for $\mathsf{DiniUniformMonotone}^{\uparrow}$

Following Dini and Bishop–Bridges on constructive compact uniform convergence, the $\mathsf{DiniUniformMonotone}^{\uparrow}$ packet records the finite monotone-window route by which a Dini-style convergence certificate hands a uniform-modulus consumer to regular-rational readback and the terminal Real seal. It sits over the Dini convergence packet of chapter 1387, the extracted uniform-modulus ledger of chapter 5795, the lower and upper semicontinuity boundaries of chapter 1536 and chapter 1640, and the located monotone-convergence route of chapter 3373. It also reads finite stream windows through chapter 126, regular-rational tolerances through chapter 301, and the terminal real boundary through chapter 38. The packet names only the finite route from displayed monotone data to a uniform-modulus handoff; it does not import open-cover compactness, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5154.1 (Dini uniform-monotone carrier)**.** *A* $\mathsf{DiniUniformMonotone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, M, L, S^{-}, S^{+}, G, W, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the accepted* $\mathsf{DiniTheorem}^{\uparrow}$ *row, exposing the compact finite-window monotone convergence surface and no independent open-cover coordinate.*
2. *M is the accepted* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *row, exposing the finite net, rational tolerance budget, extracted uniform-modulus row, and uniform-limit handoff consumed by later rows.*
3. *L is the* $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$ *handoff row, recording the located supremum route used by the monotone tail read before any real seal is exposed.*
4. *S^{-} and S^{+} are the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *and* $\mathsf{UpperSemicontinuous}^{\uparrow}$ *boundary rows, storing the displayed lower and upper semicontinuity checks at the finite window boundary.*
5. *G is the monotone-window ledger, listing the inspected family indices, pointwise comparison rows, semicontinuity boundary rows, and the finite handoff from D to M and L.*
6. *W is the shared* $\mathsf{StreamName}^{\uparrow}$ *window row through which source observations, monotone comparisons, semicontinuity checks, and modulus requests are inspected.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row, storing the regular-rational readbacks assigned to the uniform-modulus request and to the terminal real seal.*

8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $D, M, L, S^-, S^+, G, W, Q$ have been displayed.*
9. *E is the finite uniform-modulus route row, recording the displayed passage from Dini finite windows through regular-rational tolerance data before R is consumed.*
10. *H records componentwise* hsame *transport for Dini, uniform-modulus, located-monotone, semicontinuity, stream-window, rational-readback, route, real-seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a uniform-modulus request through $D, M, L, S^-, S^+, G, W, Q, E, R, H, P, N$*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformMonotone}^{\uparrow}$, $\mathsf{DiniTheorem}^{\uparrow}$, $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{UpperSemicontinuous}^{\uparrow}$, $\mathsf{LocatedMonotoneConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformMonotone}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, L, S^-, S^+, G, W, Q, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, quotient real equality, hidden selected limits, detached real representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5154.2 (Dini uniform-monotone NameCert obligations)**.** *For every accepted* $\mathsf{DiniUniformMonotone}^{\uparrow}$ *carrier*

$$U = (D, M, L, S^-, S^+, G, W, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformMonotone}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the Dini finite-window row, extracted uniform-modulus row, located-monotone handoff, lower and upper semicontinuity boundary rows, monotone-window ledger, stream-window row, regular-rational tolerance row, Real seal, route row, transport, replay, provenance, and local naming;*
2. *monotone-window ledger admission, requiring every family index, comparison row, and tail request used by a consumer to occur in G and to be read from D, M, L;*
3. *compact finite-modulus handoff, requiring the uniform-modulus row M and the route row E to mediate every tolerance request before any* $\mathsf{Real}^{\uparrow}$ *seal is exposed;*
4. *semicontinuity boundary admission, requiring S^- and S^+ to record the lower and upper boundary checks attached to the same finite windows named by G and W;*
5. *located-monotone convergence routing, requiring the handoff row L to expose the located supremum route used for the monotone tail read;*
6. *transport and replay stability through H and C, preserving the displayed rows $D, M, L, S^-, S^+, G, W, Q, E, R$ componentwise;*
7. *provenance and local naming through P and N, with no obligation allowed to call on open-cover compactness, countable choice, quotient real equality, detached representatives, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. Carrier admission exposes exactly the rows $D, M, L, S^-, S^+, G, W, Q, R, E, H, C$ so each obligation is read from a displayed coordinate of Definition 5154.1.

The monotone-window obligations are local to D, M, L and G. The Dini row supplies the finite monotone convergence surface, the uniform-modulus row supplies the extracted finite-modulus handoff, the located-monotone row supplies the tail route, and G records the inspected indices and comparisons.

The boundary obligations are local to S^-, S^+, W, Q, E, R. The lower and upper semicontinuity rows are attached to the same stream windows and regular-rational tolerances, while E records the route that must be traversed before the terminal $\mathsf{Real}^\uparrow$ row R can be read.

Finally, H and C transport and replay only the displayed rows, and P, N source and name the same packet. Since the excluded compactness, choice, quotient, representative, and completeness data are absent from the carrier, they cannot occur among the local NameCert obligations. □

Theorem 5154.3 (Dini uniform-monotone finite-modulus route)**.** *Let*

$$U = (D, M, L, S^-, S^+, G, W, Q, R, E, H, C, P, N)$$

be an accepted $\mathsf{DiniUniformMonotone}^\uparrow$ *carrier. Every uniform-modulus consumer exported by* U *factors through the displayed Dini finite-window route*

$$D, G, W, S^-, S^+, M, L, Q, E, R, H, C, P, N$$

before the $\mathsf{Real}^\uparrow$ *seal* R *is consumed. Thus a consumer may read the compact finite-window monotonicity data, semicontinuity boundary checks, extracted uniform modulus, located monotone-convergence handoff, stream windows, and regular-rational tolerance rows, but it may not replace this route by open-cover compactness, countable choice, quotient real equality, a hidden selected limit, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Start with a uniform-modulus consumer. By the carrier definition, the consumer can reach a terminal real row only at R, and the route row E records the predecessors of that read.

The Dini finite-window portion is supplied by D, G, W. These rows name the compact monotone windows, inspected family indices, comparison rows, and stream windows visible to the consumer.

The boundary and convergence handoff is then forced through S^-, S^+, M, L, Q. The semicontinuity rows keep the lower and upper boundary checks tied to the same finite windows, the uniform-modulus row mediates the tolerance request, the located-monotone row supplies the tail route, and the regular-rational row supplies the displayed readback.

The structural rows H, C, P, N preserve, replay, source, and name this same route. Since no alternative compactness, choice, quotient, selected-limit, or completeness coordinate is present, every accepted consumer factors through the displayed route before the Real seal. □

Theorem 5154.4 (Dini uniform-monotone compact-metric route)**.** *Every accepted* $\mathsf{DiniUniformMonotone}^\uparrow$ *read first exposes the* $\mathsf{CompactMetric}^\uparrow$ *finite-window source of chapter 129 through the* $\mathsf{DiniTheorem}^\uparrow$ *and* $\mathsf{DiniUniformLimitModulus}^\uparrow$ *rows of chapter 1387 and chapter 5795 before it reaches* $\mathsf{RegSeqRat}^\uparrow$ *readback and the terminal* $\mathsf{Real}^\uparrow$ *seal. Thus the finite-modulus route consumed by* $U = (D, M, L, S^-, S^+, G, W, Q, R,$ *is a compact-metric sibling dependency route, not a detached monotone convergence packet or an ambient compactness theorem.*

Proof. Project the carrier of Definition 5154.1. The Dini row D and uniform-modulus row M are the only coordinates through which compact finite-window data can enter the packet.

Apply Definition 5154.3. Its route forces the consumer through $D, G, W, S^-, S^+, M, L, Q, E, R$ before the Real seal is readable, so the compact-metric source is exposed before regular-rational readback and Real sealing.

The cited compact-metric, Dini-theorem, and uniform-limit-modulus chapters provide exactly the sibling rows used by D and M. Since the carrier has no separate compactness coordinate, no consumer can bypass those rows or replace them by a host compactness principle. □

Theorem 5154.5 (Dini uniform-monotone L10 readback induction)**.** *Every* $\mathsf{Real}^{\uparrow}$*- or* $\mathsf{Completion}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DiniUniformMonotone}^{\uparrow}$ *packet must induct over the displayed monotone-window ledger* G*. Each induction step passes through the same* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row* Q*, finite route row* E*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *named by the carrier*

$$U = (D, M, L, S^-, S^+, G, W, Q, R, E, H, C, P, N).$$

The L10 readback may reuse the carrier, NameCert obligation surface, and finite-modulus route, but it may not replace this induction by open-cover compactness, countable choice, quotient real equality, hidden selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 5154.1. The only row that lists the inspected monotone family indices, comparison rows, semicontinuity boundary rows, and finite handoff data is G, so an L10-facing readback must proceed by finite induction over that ledger.

For the base ledger row, the consumer reads the displayed stream window W, regular-rational tolerance row Q, route row E, and Real seal R only after the corresponding Dini, uniform-modulus, located-monotone, and semicontinuity rows have been exposed. This is exactly the obligation surface of Definition 5154.2.

For the induction step, append one displayed monotone-window entry from G. The same windows W and tolerance row Q are replayed through E, while H and C transport and replay the updated finite route with the same provenance P and local name N.

Finally apply the finite-modulus route of Definition 5154.3. It keeps every uniform-modulus and Real-facing read on the displayed path through $D, G, W, S^-, S^+, M, L, Q, E, R, H, C, P, N$, so no excluded compactness, choice, quotient, selected-limit, or completeness principle can replace the induction. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformMonotone}^{\uparrow}$ uniform-modulus read must expose a Dini finite-window row, an extracted Dini uniform-limit modulus row, lower and upper semicontinuity boundary rows, a located monotone-convergence handoff, StreamName windows, RegSeqRat tolerance readback, a finite route row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real-facing consumer may inspect the result.

Independence witness: dini-theorem, dini-uniform-limit-modulus, lowersemicontinuous, uppersemicontinuous, located-monotone-convergence, streamname, regseqrat, real: $\mathsf{DiniUniformMonotone}^{\uparrow}$ is not bijective to any listed sibling because it binds monotone finite windows, two semicontinuity boundary rows, located monotone-convergence routing, regular-rational tolerances, and an extracted uniform-modulus handoff into one packet before the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformMonotone}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformMonotone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5154.1, Definition 5154.2, and Definition 5154.3 seed the finite Dini uniform-monotone packet from compact monotone windows to uniform-modulus and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformMonotone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, quotient real equality, hidden selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window ledger admission, compact finite-modulus handoff, semicontinuity boundary admission, located-monotone convergence routing, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformMonotoneUp.

Constructive story: A $\mathsf{DiniUniformMonotone}^{\uparrow}$ packet is generated from Dini finite-window rows, Dini uniform-limit modulus rows, lower and upper semicontinuity boundary rows, located monotone-convergence rows, StreamName windows,

RegSeqRat tolerance readbacks, finite route rows, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theorem 5154.6 (Dini uniform-monotone sibling lattice route)**.** *Every accepted* DiniUniformMonotone$^{\uparrow}$ *consumer reads the* DiniTheorem$^{\uparrow}$, DiniUniformLimitModulus$^{\uparrow}$, CompactModulusMonotoneConvergence$^{\uparrow}$, LowerSemicontinuous$^{\uparrow}$, UpperSemicontinuous$^{\uparrow}$, LocatedMonotoneConvergence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, *and* Real$^{\uparrow}$ *rows in the same finite sibling lattice before a uniform-modulus read reaches the Real seal.*

Proof. Start with an accepted consumer of Definition 5154.1. The carrier exposes the Dini finite-window row, the uniform-limit modulus row, the compact-modulus monotone-convergence row of chapter 7553, the lower and upper semicontinuity rows, the located monotone-convergence row, the stream-window row, the regular-rational row, and the terminal real row as displayed coordinates of one finite packet.

Apply Definition 5154.2. Its carrier-admission, monotone-window, compact finite-modulus, semicontinuity, located-routing, transport, replay, provenance, and local naming clauses require these rows to be read through the same local NameCert surface.

Finally use Definition 5154.3. A uniform-modulus read reaches the terminal Real$^{\uparrow}$ seal only after the displayed Dini, compact-modulus monotone-convergence, uniform-limit modulus, semicontinuity, located-monotone, stream, and regular-rational rows have been traversed, so the sibling lattice is fixed before the Real seal is consumed. □

Theorem 5154.7 (Dini uniform-monotone compact dependency route)**.** *Every accepted* DiniUniformMonotone$^{\uparrow}$ *consumer must expose the compact finite-window dependency through the* DiniTheorem$^{\uparrow}$ *and* DiniUniformLimitModulu *rows, the monotone-tail dependency through* LocatedMonotoneConvergence$^{\uparrow}$, *and the compact source discipline already cited by chapter 1387, chapter 5795, and chapter 129 before the route reaches its* RegSeqRat$^{\uparrow}$ *tolerance row and terminal* Real$^{\uparrow}$ *seal.*

Proof. Project an accepted carrier from Definition 5154.1. The compact-facing rows are D, M, G, W, while the monotone-tail handoff is L and the tolerance-to-real route is Q, E, R.

Apply Definition 5154.2. Its compact finite-modulus and located-monotone clauses require every uniform-modulus request to read the Dini finite-window surface, the extracted modulus row, and the located monotone-convergence row before a real-facing consumer inspects R.

Finally use Definition 5154.6. The sibling lattice fixes the compact, monotone, stream, regular-rational, and real rows in one finite packet, so no accepted route can bypass the compact source discipline or replace the monotone dependency by a host convergence theorem. □

The CompleteLattice$^{\uparrow}$ hub routes its finite join, meet, certified-family, monotone-map, and fixed-point boundary rows through chapter 55, chapter 53, chapter 3668, and the downstream fixed-row consumer of chapter 5919.

Chapter 5155

A Concrete Naming Certificate for $\mathsf{CompleteLattice}^{\uparrow}$

Following Tarski and Bourbaki, the $\mathsf{CompleteLattice}^{\uparrow}$ packet records the finite order-theoretic interface in which every displayed certified family has a named join and meet row. It sits above the meet–join order surface of chapter 55, the antisymmetric order carrier of chapter 53, and the displayed membership discipline of chapter 3668. The packet names the rows needed by fixed-point and domain-theoretic consumers while excluding host powersets, Zorn-style maximal principles, unrestricted choice, quotient representatives, and ambient ordinal iteration.

Definition 5155.1 (Complete lattice carrier)**.** *A* $\mathsf{CompleteLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (O, F, J, M, T, B, H, C, P, N)$$

with the following displayed rows:

1. *O is the underlying order row, read through* $\mathsf{Poset}^{\uparrow}$ *and* $\mathsf{Lattice}^{\uparrow}$ *and carrying the visible comparison, meet, and join obligations.*
2. *F is the bounded certified-family row. It records only displayed* $\mathsf{SetLike}^{\uparrow}$ *membership data together with finite lower-bound and upper-bound witnesses for the family being inspected.*
3. *J is the displayed join schedule assigning a least upper bound row to the certified family in F.*
4. *M is the displayed meet schedule assigning a greatest lower bound row to the certified family in F.*
5. *T is the monotone-map row, recording a displayed self-map whose order-preservation is checked against the comparison row in O.*
6. *B is the fixed-point boundary row. It records the finite pre-fixed and post-fixed comparison surface visible to a Tarski-style consumer, without selecting an ordinal chain.*
7. *H records componentwise* hsame *transport for $O, F, J, M, T, B, C, P, N$.*
8. *C records displayed* Cont *replay from an order request through the certified-family row, the join and meet schedules, the monotone-map row, and the fixed-point boundary row.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteLattice}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, F, J, M, T, B, H, C, P, N$. It has no coordinate for a host powerset, Zorn lemma, unrestricted choice, quotient representatives, selected ordinal iteration, or an ambient complete lattice object.

Theorem 5155.2 (Complete lattice NameCert obligations)**.** *For every accepted* $\mathsf{CompleteLattice}^{\uparrow}$ *carrier*

$$L = (O, F, J, M, T, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteLattice}^{\uparrow}}$ *surface consists exactly of order-row admission, bounded certified-family exposure, displayed join scheduling, displayed meet scheduling, monotone-map exposure, fixed-point boundary exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every fixed-point-facing read must expose O, F, J, M, T, B before it may consume transport, replay, provenance, or naming rows.*

Proof. Project the carrier of Definition 5155.1. The order row O supplies the only comparison surface, and the family row F supplies the only displayed family whose bounds may be inspected.

The join and meet schedules are finite rows attached to that same family. Thus a consumer cannot replace J or M by an ambient supremum or infimum operation; it must read the displayed schedule bound to F.

The monotone-map row T and fixed-point boundary row B are downstream of O, F, J, M. They name only the comparison data needed by a fixed-point consumer, while H, C, P, N transport, replay, source, and name the same finite route.

Since no excluded host principle or representative selector occurs as a carrier coordinate, these rows exhaust the local NameCert obligations. □

Theorem 5155.3 (Complete lattice monotone fixed-point boundary)**.** *Every fixed-point-facing consumer of an accepted* $\mathsf{CompleteLattice}^{\uparrow}$ *packet factors through the finite route*

$$O, F, J, M, T \longmapsto B$$

before it may read any fixed-point boundary. The exported boundary records monotone-map comparison, certified-family bounds, displayed joins, and displayed meets; it does not export a host powerset construction, Zorn lemma, unrestricted choice, quotient representative, selected ordinal iteration, global Knaster–Tarski theorem, or ambient domain-theoretic completion.

Proof. Begin with the NameCert obligations of Definition 5155.2. They force the consumer to expose the order row, family row, join schedule, meet schedule, monotone-map row, and fixed-point boundary row before any structural support row is read.

The row B is therefore not an independent theorem of fixed-point existence. It is a boundary ledger over the displayed order and completeness schedules, and it is meaningful only after O, F, J, M, T have been consumed.

Transport and replay preserve this route componentwise, while provenance and local naming keep it tied to the $\mathsf{CompleteLattice}^{\uparrow}$ packet. The absence of powerset, choice, quotient, ordinal-iteration, and ambient-completion coordinates prevents the boundary from exporting any of the excluded fixed-point principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteLattice}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteLattice}^{\uparrow}$ packet is generated from an order row, a bounded certified-family row, displayed join and meet schedules, a monotone-map row, a fixed-point boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5155.1, Definition 5155.2, and Definition 5155.3 seed the finite complete-lattice surface over order, certified-family, join, meet, monotone-map, fixed-point boundary, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompleteLattice}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host powersets, Zorn lemma, unrestricted choice, quotient representatives, selected ordinal iteration, global Knaster–Tarski fixed-point existence, ambient domain-theoretic completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, order-row exposure, certified-family boundedness, join-schedule exactness, meet-schedule exactness, monotone-map preservation, fixed-point boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompleteLatticeUp.

Theorem 5155.4 (Complete lattice order sibling route)**.** *Every* $\mathsf{CompleteLattice}^{\uparrow}$ *fixed-point read or family-boundary read factors first through the finite order sibling route*

$$\mathsf{SetLike}^{\uparrow}, \mathsf{Poset}^{\uparrow}, \mathsf{Lattice}^{\uparrow} \longmapsto \mathsf{CompleteLattice}^{\uparrow}.$$

The $\mathsf{SetLike}^{\uparrow}$ *membership rows expose the displayed family, the* $\mathsf{Poset}^{\uparrow}$ *row supplies the antisymmetric comparison surface, and the* $\mathsf{Lattice}^{\uparrow}$ *row supplies the meet and join operations available before the complete-lattice schedules are read. Only after those sibling rows are visible may a consumer expose the complete-lattice join schedule, meet schedule, monotone-map row, or fixed-point boundary row.*

Proof. Project the carrier of Definition 5155.1. The certified-family row F is read as displayed membership data, so every family-boundary consumer begins with the $\mathsf{SetLike}^{\uparrow}$ surface.

The order row O is not a free comparison relation. It is read through the antisymmetric $\mathsf{Poset}^{\uparrow}$ surface and the meet–join $\mathsf{Lattice}^{\uparrow}$ surface named in the carrier, so fixed-point consumers must expose those sibling rows before using J, M, T, or B.

The join and meet schedules are therefore complete-lattice rows over an already visible membership and order route. They do not replace the sibling route by an ambient powerset, unrestricted supremum operator, quotient representative, or selected ordinal iteration. □

$\mathsf{PointwiseErgodic}^{\uparrow}$ routes finite orbit averages through chapter 196, chapter 92, and chapter 38 before any average row is consumed.

Chapter 5156

A Concrete Naming Certificate for $\mathsf{PointwiseErgodic}^{\uparrow}$

Following Birkhoff and von Neumann on pointwise ergodic averages, the $\mathsf{PointwiseErgodic}^{\uparrow}$ packet records only the finite orbit-window and average-readback data that a BEDC consumer may inspect before any almost-everywhere convergence theorem is visible. It sits after the transformation and invariant-row surface of chapter 196, the event and mass rows of chapter 92, and the terminal real-name seal of chapter 38. The packet names finite orbit segments, summation ledgers, average rows, measure-context rows, real seals, transport, replay, provenance, and local naming; it does not assert the classical pointwise ergodic theorem.

Definition 5156.1 (PointwiseErgodic carrier)**.** *A* $\mathsf{PointwiseErgodic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (E, M, O, S, A, R, H, C, Q, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Ergodic}^{\uparrow}$ *row exposing the displayed measure-preserving transformation and invariant-row ledger consumed by the packet.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *row carrying the finite measurable-event and mass requests attached to the inspected orbit windows.*
3. *O is the orbit-window row, listing the finitely many iterates of the displayed transformation that are currently read.*
4. *S is the finite summation ledger for the values read along O.*
5. *A is the finite average row, obtained from S and the displayed window length before any limiting conclusion is available.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the average values exported to real-facing consumers.*
7. *H records componentwise* hsame *transport for the ergodic, measure, orbit, summation, average, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an orbit-average request through E, M, O, S, A and into R.*
9. *Q is the* Pkg *provenance over* $\mathsf{PointwiseErgodic}^{\uparrow}$, $\mathsf{Ergodic}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{PointwiseErgodic}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $E, M, O, S, A, R, H, C, Q, N$. It has no coordinate for full almost-everywhere convergence, countable choice, a selected invariant representative, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5156.2 (PointwiseErgodic NameCert obligations)**.** *For every accepted* $\mathsf{PointwiseErgodic}^{\uparrow}$ *carrier*

$$P = (E, M, O, S, A, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PointwiseErgodic}^{\uparrow}}$ *surface consists exactly of carrier admission, ergodic-row exposure, measure-context exposure, orbit-window inspection, finite summation accounting, finite average formation, real-seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the excluded host principles.*

Proof. Project the carrier of Definition 5156.1. The analytic rows are E, M, O, S, A, R, and the structural rows are H, C, Q, N.

The orbit-window row O is finite. The summation ledger S reads only the displayed iterates in that window, and the average row A is formed from the same finite ledger before any limiting statement can be consumed.

The measure and real rows do not supply an almost-everywhere theorem. The measure row M records the finite event and mass context for the inspected windows, while R seals only the displayed average values.

Finally H, C, Q, N transport, replay, source, and name the same route. Since no other coordinate occurs in the carrier, the NameCert surface exports none of full almost-everywhere convergence, countable choice, selected invariant representatives, quotient equality of functions, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5156.3 (PointwiseErgodic finite average route)**.** *Every real-facing read exported by an accepted* $\mathsf{PointwiseErgodic}^{\uparrow}$ *carrier factors through*

$$E \to M \to O \to S \to A \to R$$

before the average value is consumed. The route records an ergodic source row, finite measure context, displayed orbit window, summation ledger, finite average row, and terminal real seal, but it does not assert convergence of the averages on an almost-everywhere set.

Proof. Start with an accepted carrier $P = (E, M, O, S, A, R, H, C, Q, N)$. By Definition 5156.1, the only average coordinate is A, and it is downstream of the orbit-window and summation rows.

The ergodic row E supplies the transformation ledger, and M supplies the finite measure context in which the inspected orbit window is read. The window O and summation ledger S then determine the finite average row.

The terminal real-facing read is R. Transport, replay, provenance, and local naming through H, C, Q, N preserve this route and do not add a convergence coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseErgodic}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PointwiseErgodic}^{\uparrow}$ packet is generated from an ErgodicUp source row, a MeasureUp finite context row, a displayed orbit-window row, a finite summation ledger, a finite average row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PointwiseErgodic}^{\uparrow}}$, seed)

Scope closed: Definition 5156.1, Definition 5156.2, and Definition 5156.3 seed the finite orbit-average certificate over ErgodicUp, MeasureUp, and RealUp rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{PointwiseErgodic}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not close full almost-everywhere convergence, countable choice, selected invariant representatives, quotient equality of functions, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, ergodic-row exposure, measure-context exposure, orbit-window inspection, summation accounting, finite average formation, real-seal routing, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.PointwiseErgodicUp.

ProofTraceNormalForm$^{\uparrow}$ routes terminal proof-trace reads through the typed endpoint boundary of chapter 562, subject-reduction preservation, type-classifier membership, and the MetaCIC frontier of chapter 4238.

Chapter 5157

A Concrete Naming Certificate for $\mathsf{ProofTraceNormalForm}^{\uparrow}$

The $\mathsf{ProofTraceNormalForm}^{\uparrow}$ packet realizes the type-checking-as-classifier-membership vision of chapter 9367 as a finite proof-trace normal-form surface. It sits after the typed endpoint boundary of chapter 562, the preservation row of chapter 483, the classifier-membership surface of chapter 476, and the concrete type-checking classifier packet of chapter 3911. The packet records proof trace, Ext membership, typed Cont replay, subject-reduction handoff, terminal normal-form boundary, transport, provenance, and local naming; it does not export host conversion, normalization oracles, erased proof semantics, or quotient term equality.

The proof-trace normal-form carrier is consumed through the typed classifier route of Definition 5157.5 before it is handed to MetaCIC normalization or confluence consumers. The downstream handoff is the candidate-normalization confluence route of chapter 5547; it receives the same proof trace, classifier membership, subject-reduction, terminal normal-form, transport, replay, provenance, and local naming rows rather than a detached normalization oracle.

Definition 5157.1 (ProofTraceNormalForm carrier)**.** *A* $\mathsf{ProofTraceNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, E, R, S, V, J, H, C, P, N)$$

with the following displayed rows:

1. *T is the proof-trace row, listing the finite rule applications and term windows whose type-checking path is being inspected.*
2. *E is the* Ext *classifier-membership row read through* $\mathsf{TypeClassifierMembership}^{\uparrow}$*.*
3. *R is the typed* Cont *replay row recording the displayed reduction route for the trace.*
4. *S is the* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff row that preserves membership along the displayed route.*
5. *V is the* $\mathsf{TypedReductionNormalForm}^{\uparrow}$ *terminal-boundary row for the final typed normal form.*
6. *J is the* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ *row that binds the same membership, derivation, preservation, and erasure-facing data into a finite checker packet.*
7. *H records componentwise* hsame *transport for the proof trace, membership, route, preservation, normal-form, checker, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from proof-trace inspection through membership, typed reduction, preservation, and terminal normal-form access.*

9. *P is the* Pkg *provenance over* ProofTraceNormalForm$^{\uparrow}$, TypeClassifierMembership$^{\uparrow}$, SubjectReduction$^{\uparrow}$, TypedReductionNormalForm$^{\uparrow}$, TypeCheckingClassifierMembership$^{\uparrow}$, BHist, Ext, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ProofTraceNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, R, S, V, J, H, C, P, N$. It has no coordinate for host conversion, semantic equality of terms, a normalization oracle, proof objects outside the displayed trace, quotient term representatives, or ambient global normalization.

Theorem 5157.2 (ProofTraceNormalForm NameCert obligations)**.** *For every accepted* ProofTraceNormalForm$^{\uparrow}$ *carrier*

$$F = (T, E, R, S, V, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ProofTraceNormalForm}^{\uparrow}}$ *surface consists exactly of proof-trace admission, classifier-membership exposure, typed-route replay, subject-reduction handoff, terminal normal-form access, checker-packet alignment, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host conversion or normalization oracles.*

Proof. Project the carrier of Definition 5157.1. The trace row T, membership row E, typed route R, preservation row S, normal-form row V, and checker row J are the only analytic coordinates.

The membership and route rows are read before the normal-form boundary. The subject-reduction row S preserves the displayed membership through the typed continuation route R, and the row V is reached only after that preservation handoff.

The checker row J aligns the same data with the finite type-checking classifier packet. It does not add host conversion or a normalization procedure; it records membership, derivation, preservation, and erasure-facing rows already visible to the packet.

The rows H, C, P, N transport, replay, source, and name this finite surface. Since no other coordinate occurs in the carrier, no accepted consumer can leave the displayed proof-trace normal-form route. □

Theorem 5157.3 (ProofTraceNormalForm terminal route)**.** *Every terminal normal-form read exported by an accepted* ProofTraceNormalForm$^{\uparrow}$ *carrier factors through*

$$T \to E \to R \to S \to V \to J$$

with H, C, P, N preserving and naming the same route. The read uses proof trace, classifier membership, typed reduction replay, subject reduction, terminal boundary, and checker alignment, but it does not use host conversion, semantic term equality, quotient representatives, erased proof semantics, or a normalization oracle.

Proof. Start with an accepted carrier $F = (T, E, R, S, V, J, H, C, P, N)$. The terminal row is V, and the carrier places it after the proof trace, membership row, typed continuation route, and subject-reduction handoff.

The checker row J records that the same route is a finite type-checking classifier packet. This alignment is downstream of the displayed membership and preservation data, not a separate host checker.

Finally H, C, P, N transport, replay, source, and name the route. Because the carrier has no host conversion, semantic equality, quotient representative, erased proof semantics, or normalization-oracle coordinate, the terminal read cannot use one. □

Theorem 5157.4 (ProofTraceNormalForm finite candidate induction)**.** *Every accepted* $\mathsf{ProofTraceNormalForm}^{\uparrow}$ *route over a finite proof trace is obtained by induction over the displayed trace row* T*. Each step preserves* Ext *membership, typed* Cont *replay,* $\mathsf{SubjectReduction}^{\uparrow}$ *handoff, terminal normal-form access, checker alignment, componentwise transport, replay, provenance, and local naming through the same carrier*

$$F = (T, E, R, S, V, J, H, C, P, N).$$

Proof. Induct over the finite proof-trace row T of Definition 5157.1. The base trace has no hidden rule application, so the membership row E, typed replay row R, preservation handoff S, terminal boundary V, and checker row J are exactly the displayed seed rows.

For the induction step, append one displayed trace entry. The Ext membership row is preserved by the typed Cont replay row, and the $\mathsf{SubjectReduction}^{\uparrow}$ handoff transports that membership to the next terminal-normal-form boundary.

The checker alignment row J then reads the same updated membership, replay, preservation, and terminal rows. Componentwise hsame transport and displayed Cont replay keep the updated route inside H and C.

Finally P and N source and name the same finite packet. Since the carrier has no coordinate for host conversion, semantic term equality, erased proof semantics, quotient representatives, or a normalization oracle, the induction produces only accepted $\mathsf{ProofTraceNormalForm}^{\uparrow}$ routes. □

Falsifiable prediction: A consumer that reaches V without $T, E, R, S, J, H, C, P, N$ refutes the $\mathsf{ProofTraceNormalForm}^{\uparrow}$ packet.

Independence witness: typedreductionnormalform, subjectreduction, typeclassifiermembership, type-checking-classifier-membership: $\mathsf{ProofTraceNormalForm}^{\uparrow}$ is not bijective to them because it binds trace, membership, replay, preservation, terminal normal form, checker alignment, transport, replay, provenance, and naming in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ProofTraceNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{ProofTraceNormalForm}^{\uparrow}$ packet is generated from a proof-trace row, Ext classifier-membership row, typed Cont replay row, SubjectReductionUp handoff, TypedReductionNormalFormUp terminal row, TypeChecking-ClassifierMembershipUp alignment row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProofTraceNormalForm}^{\uparrow}}$, seed)

Scope closed: Definition 5157.1, Definition 5157.2, and Definition 5157.3 seed the finite proof-trace route from classifier membership to terminal typed normal form.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProofTraceNormalForm}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host conversion, semantic equality of terms, quotient representatives, erased proof semantics, normalization oracles, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked proof-trace admission, classifier-membership exposure, typed-route replay, subject-reduction handoff, terminal normal-form access, checker-packet alignment, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.ProofTraceNormalFormUp.

Theorem 5157.5 (ProofTraceNormalForm typed classifier route)**.** *Every terminal* $\mathsf{ProofTraceNormalForm}^{\uparrow}$ *read must expose the typed endpoint boundary of chapter 562, preservation through chapter 483, classifier membership through chapter 476, and the checker packet of chapter 3911 before the terminal row* V *is read. The route is carried by*

$$F = (T, E, R, S, V, J, H, C, P, N)$$

and exports no host conversion, semantic term equality, quotient representative, erased proof semantics, or normalization oracle.

Proof. Project the carrier rows $T, E, R, S, V, J, H, C, P, N$ of Definition 5157.1. The membership row E is the classifier boundary, the replay row R is the displayed typed route, the preservation row S is the subject-reduction handoff, the terminal row V is the typed normal-form endpoint, and J is the finite checker packet.

Apply Definition 5157.3. It orders the route through proof trace, classifier membership, typed replay, subject reduction, terminal boundary, and checker alignment before any terminal read is exported.

Apply Definition 5157.4. It keeps the same rows visible over the finite proof trace, so the typed endpoint boundary, preservation handoff, classifier-membership row, and checker packet cannot be replaced by host conversion or an oracle.

Finally H, C, P, N transport, replay, source, and name the same packet. Since Definition 5157.2 lists no coordinate for semantic term equality, quotient representatives, erased proof semantics, or normalization oracles, a terminal read of V must follow the displayed typed-classifier sibling route. □

Theorem 5157.6 (ProofTraceNormalForm MetaCIC residual route)**.** *Every terminal* $\mathsf{ProofTraceNormalForm}^{\uparrow}$ *read used by a MetaCIC normalization or confluence consumer must display* $\mathsf{TypeClassifierMembership}^{\uparrow}$*,* $\mathsf{SubjectReduction}^{\uparrow}$*,* $\mathsf{TypedReductionNormalForm}^{\uparrow}$*,* $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$*, and the MetaCIC residual or normalization frontier route before the terminal normal-form row is consumed. In carrier notation*

$$F = (T, E, R, S, V, J, H, C, P, N),$$

the rows E, S, V, J remain visible to consumers such as chapter 7825 and chapter 4238; the export provides no host conversion, quotient term equality, erased proof semantics, or normalization oracle.

Proof. Project the carrier $F = (T, E, R, S, V, J, H, C, P, N)$ from Definition 5157.1. The row E is the $\mathsf{TypeClassifierMembership}^{\uparrow}$ boundary, S is the $\mathsf{SubjectReduction}^{\uparrow}$ handoff, V is the $\mathsf{TypedReductionNormalForm}^{\uparrow}$ terminal row, and J is the $\mathsf{TypeCheckingClassifierMembership}^{\uparrow}$ checker alignment row.

Apply Definition 5157.5. It already places classifier membership, subject reduction, typed normal form, and type-checking membership before any terminal read of V is exported. A MetaCIC normalization or confluence consumer therefore receives the same ordered boundary before it may feed the residual-gap or parallel-diamond frontier.

The MetaCIC-facing handoff is only a route through the displayed rows. The residual and normalization frontier chapters may consume E, S, V, J, H, C, P, N as finite proof-trace evidence, but the packet contains no coordinate for host conversion, quotient term equality, erased proof semantics, or a normalization oracle, so the terminal normal-form row cannot be consumed through any off-packet shortcut. □

Chapter 5158

A Concrete Naming Certificate for $\mathsf{RegularCauchyFunctor}^{\uparrow}$

Following Bishop and Bridges on uniform maps between regular Cauchy presentations, the $\mathsf{RegularCauchyFunctor}^{\uparrow}$ packet records the finite transport route by which a regular Cauchy name in one displayed presentation is read as a regular Cauchy name in another. It sits between the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the rational regular readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion-facing functorial surface of chapter 423. The packet names functorial transport as displayed BHist data; it does not assert quotient representative equality, select a limiting real, or import ambient completeness.

Definition 5158.1 (Regular Cauchy functor carrier)**.** *A* $\mathsf{RegularCauchyFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, M, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{StreamName}^{\uparrow}$ *row, recording the inspected source windows, source precision requests, and regular Cauchy tail schedule.*
2. *T is the target* $\mathsf{StreamName}^{\uparrow}$ *row, recording the target windows produced by the transported readback.*
3. *M is the modulus row, naming the finite precision conversion that sends each target request to a source request.*
4. *R is the mapped* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, assembled from the source windows in S, the target windows in T, and the modulus row M.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-control row, recording the dyadic radii, slack, comparison budget, and tail inequalities used by the transport.*
6. *H is the componentwise* hsame *transport row for S, T, M, R, Q, C, P, N.*
7. *C is the displayed* Cont *replay row from a target precision request through the modulus row, source window, mapped readback, dyadic tolerance ledger, and target window.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFunctor}^{\uparrow}}$ *row binding the displayed transport packet.*

The classifier compares accepted packets only through $S, T, M, R, Q, H, C, P, N$. It has no coordinate for selected limits, quotient stream representatives, host function extensionality, full $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFunctorUp.RegularCauchyFunctorTasteGate single carrier alignment`

Theorem 5158.2 (Regular Cauchy functor NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFunctor}^{\uparrow}$ *carrier*

$$F = (S, T, M, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window display, target-window display, modulus-row display, mapped* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance control, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5158.1. The analytic rows are S, T, M, R, Q: they expose the source observations, target observations, finite precision conversion, mapped rational readback, and dyadic tolerance budget.

The structural rows cannot add a new analytic coordinate. The row H preserves the displayed components, C replays the same request route, P records the sibling dependencies, and N names the resulting packet.

Since no row for selected limits, quotient representatives, or host function equality is present, the listed rows exhaust the local NameCert obligations. Any transport claim omitting one of them is only a projection of F, not an accepted $\mathsf{RegularCauchyFunctor}^{\uparrow}$ packet. □

Lean4 ✓ `BEDC.Derived.RegularCauchyFunctorUp.RegularCauchyFunctorCarrier namecert obligations`

Theorem 5158.3 (Regular Cauchy functor composition law)**.** *Let* $F = (S, T, M, R, Q, H, C, P, N)$ *and* $G = (T, U, M', R', Q', H', C', P', N')$ *be accepted* $\mathsf{RegularCauchyFunctor}^{\uparrow}$ *packets whose target row of* F *is the source row of* G*. Their displayed composite is the finite packet obtained by composing the modulus rows* M' *after* M*, replaying the source-to-middle and middle-to-target windows through* C *and* C'*, and concatenating the readback, tolerance, transport, provenance, and local-name rows. The composite exports only the resulting* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *data.*

Proof. Read the shared row T as the middle window surface. The first packet transports a target request for T back to S by M, C, while the second transports a target request for U back to T by M', C'.

Compose the request conversions in the displayed order. The replay route first uses C' to reach the middle request, then C to reach the source request; the readback rows R, R' and tolerance rows Q, Q' supply the rational and dyadic evidence for the same route.

The remaining rows are structural. Componentwise transport is inherited from H, H', provenance is repacked from P, P', and the local names N, N' bind the composite certificate. No additional host equality or selected limit is introduced by this finite repacking. □

Theorem 5158.4 (Regular Cauchy functor Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyFunctor}^{\uparrow}$ *packet factors through the displayed source and target* $\mathsf{StreamName}^{\uparrow}$ *rows, the modulus row, the mapped* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-control row before any real-name seal can be read. The packet exports no selected limit, no quotient representative, no full* $\mathsf{Real}^{\uparrow}$ *completeness theorem, and no independent host map.*

Proof. Use the carrier $F = (S, T, M, R, Q, H, C, P, N)$. A real-name read can only use the target windows in T after the replay route has opened the precision conversion M, the source windows S, the mapped readback R, and the dyadic tolerance ledger Q.

The rows H, C, P, N preserve, replay, source, and name this same finite route. They do not contain a coordinate for selecting a limit or replacing the displayed stream by a quotient representative.

Thus any $\mathsf{Real}^{\uparrow}$ seal obtained from the packet remains downstream of the named regular Cauchy transport surface. A consumer that asks for ambient completeness or a host map has left the carrier described in Definition 5158.1. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFunctor}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyFunctor}^{\uparrow}$ packet is generated from source and target StreamName windows, a finite modulus row, mapped RegSeqRat readback, DyadicRatCore tolerance control, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyFunctor}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5158.1, Definition 5158.2, Definition 5158.3, and Definition 5158.4 seed the finite functorial transport surface for regular Cauchy names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyFunctor}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient representatives, full Real completeness, host function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window display, target-window display, modulus conversion, RegSeqRat readback exactness, DyadicRatCore tolerance control, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFunctorUp.

Chapter 5159

A Concrete Naming Certificate for HyperbolicIsometryGroup$^{\uparrow}$

The HyperbolicIsometryGroup$^{\uparrow}$ packet realizes the group-facing route of chapter 9463 as finite disk-map, inverse, hyperbolic-metric preservation, boundary-action, transport, provenance, and NameCert rows. It sits after the disk-coordinate carrier of chapter 377, the metric readback of chapter 3474, the visual-boundary surface of chapter 4181, the Mobius-transform packet of chapter 5110, and the finite evolution route of chapter 195. The packet names a simultaneous isometry-group surface; it is not a completed hyperbolic plane, quotient ideal-boundary equality, Lie-group manifold, Haar-measure object, or physical observer law.

Definition 5159.1 (Hyperbolic isometry group carrier)**.** *A* HyperbolicIsometryGroup$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, M, F, I, B, A, Q, T, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-coordinate row for finite source and target observations.*
2. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-preservation row over those observations.*
3. *F is the* MobiusTransform$^{\uparrow}$ *disk-map row, restricted to maps whose displayed source and target rows remain in the disk carrier.*
4. *I is the inverse row, recording the displayed inverse disk map and the finite two-sided replay checks.*
5. *B is the* HyperbolicVisualBoundary$^{\uparrow}$ *boundary-action row, recording the induced action on displayed visual-boundary data.*
6. *A is the group-law row, displaying identity, composition, and inverse closure for the finite map packet.*
7. *Q is the* DynSystem$^{\uparrow}$ *replay row, used when a finite orbit-style consumer reads repeated action.*
8. *T records componentwise* hsame *transport for disk coordinates, metric preservation, map row, inverse row, boundary action, group law, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite disk-map request through inverse checking, metric preservation, boundary action, group-law inspection, and orbit-style replay.*

10. *P is the* Pkg *provenance over* HyperbolicIsometryGroup$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualBoundary MobiusTransform$^{\uparrow}$, DynSystem$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{HyperbolicIsometryGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, F, I, B, A, Q, T, C, P, N$. The carrier has no coordinate for a completed hyperbolic plane, quotient ideal endpoints, continuous Lie-group topology, Haar measure, host analytic equality of maps, or physical observer dynamics.

Theorem 5159.2 (Hyperbolic isometry group NameCert obligations)**.** *For every accepted* HyperbolicIsometryGroup *carrier*

$$G = (D, M, F, I, B, A, Q, T, C, P, N),$$

the local NameCert$_{\text{HyperbolicIsometryGroup}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, disk-map admission, inverse closure, hyperbolic-metric preservation, visual-boundary action, identity and composition rows, finite orbit replay, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the carrier through Definition 5159.1. The accepted tuple exposes exactly the disk, metric, map, inverse, boundary, group-law, replay, transport, provenance, and naming rows.

The isometry obligation is read from M, F, I. The map row displays the disk action, the inverse row supplies the finite two-sided checks, and the metric row records preservation of the hyperbolic readback on the displayed observations.

The group and boundary obligations are read from A, B, Q. Composition, identity, and inverse closure are finite rows in A, boundary action is confined to B, and repeated action is replayed only through the DynSystem$^{\uparrow}$ row Q.

Transport and replay through T, C, P, N preserve the same packet. Since the carrier stores no completed plane, quotient ideal endpoint, Lie topology, Haar measure, host analytic equality, or physical dynamics, the listed rows exhaust the local NameCert surface. □

Theorem 5159.3 (Hyperbolic isometry group boundary action)**.** *For every accepted* HyperbolicIsometryGroup$^{\uparrow}$ *carrier $G = (D, M, F, I, B, A, Q, T, C, P, N)$, every boundary-action consumer must read the disk-map row F, inverse row I, metric-preservation row M, and group-law row A before the visual-boundary row B can be exported as stable under the action.*

Proof. Start with a boundary-action request. The request is attached to the displayed disk map F, so the disk-coordinate row D and the inverse row I are visible before boundary data can be consumed.

Metric preservation is the row M. It keeps the boundary read tied to the same hyperbolic metric surface rather than to an external visual compactification.

Finally read the group-law row A and the boundary row B. The rows T, C, P, N transport, replay, source, and name this finite route, while Q handles any repeated-action read. Thus boundary stability is a packet fact over $D, M, F, I, B, A, Q, T, C, P, N$, not a quotient-boundary theorem or a completed hyperbolic-plane assertion. □

Falsifiable prediction: If the metric-preservation row is removed, no current boundary-action consumer can recover both inverse closure and visual-boundary stability from the remaining packet rows.

Independence witness: hyperbolic_metric, hyperbolic_visual_boundary, mobius_transform, dynsystem: the new carrier has a simultaneous group law, inverse, metric-preservation, and boundary-action obligation surface that no single sibling packet carries.

The Lean follow-up will provide Nontrivial HyperbolicIsometryGroup$^{\uparrow}$ and FieldFaithful HyperbolicIsometryGroup$^{\uparrow}$ instances in `TasteGate.lean`.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicIsometryGroup}^{\uparrow}}$.

Constructive story: A HyperbolicIsometryGroup$^{\uparrow}$ packet is generated from UnitDisk coordinates, HyperbolicMetric preservation, MobiusTransform disk maps, inverse checks, HyperbolicVisualBoundary action, group-law rows, DynSystem replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicIsometryGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5159.1, Definition 5159.2, Definition 5159.3, the falsifiable prediction, and the independence witness seed the finite hyperbolic isometry group route over disk-map, inverse, metric-preservation, boundary-action, replay, transport, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicIsometryGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic geometry, quotient ideal-boundary equality, Lie-group topology, Haar measure, physical dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-map admission, inverse closure, metric preservation, boundary action, group-law rows, replay, transport, provenance, and local naming for the same finite packet.

Chapter 5160

A Concrete Naming Certificate for $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$

Following Reed–Simon and Conway on bounded self-adjoint operators on Hilbert spaces, the $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$ packet records the finite BEDC surface on which a bounded operator can be read together with its adjoint witness and symmetry row. It sits after the Hilbert carrier of chapter 91, the inner-product row of chapter 86, the operator-norm boundary of chapter 3603, the normal-operator packet of chapter 5807, and the spectral-radius consumer boundary of chapter 2352. The packet names bounded self-adjoint operator data; it does not assert spectral-measure existence, functional calculus, compactness, diagonalization, or a host operator-space equality.

Definition 5160.1 (Bounded self-adjoint operator carrier)**.** *A* $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (H, I, G, N, J, S, R, T, C, P, L)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row for the source and target Hilbert endpoint.*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row through which adjoint comparisons are read.*
3. *G is the bounded operator graph row, recording the displayed action on accepted vectors.*
4. *N is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row attached to the same graph.*
5. *J is the adjoint witness row, pairing the action row with the inner-product transpose read.*
6. *S is the self-adjoint symmetry row, identifying the displayed operator row with its adjoint witness through the same* hsame *route.*
7. *R is the spectral-consumer boundary row read by* $\mathsf{NormalOperator}^{\uparrow}$ *and* $\mathsf{SpectralRadius}^{\uparrow}$ *consumers before any spectral request is allowed.*
8. *T records componentwise transport for the Hilbert carrier, inner-product row, graph, norm bound, adjoint witness, symmetry, spectral boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite operator observation through graph inspection, norm-bound readback, adjoint comparison, symmetry checking, and spectral-consumer boundary exposure.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{NormalOperator}^{\uparrow}$, $\mathsf{SpectralRadius}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* $\mathsf{NameCert}_{\mathsf{BoundedSelfAdjointOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, I, G, N, J, S, R, T, C, P, L$. The carrier has no coordinate for spectral measures, eigenbasis selection, compactness, quotient operator equality, unbounded domains, or an ambient functional calculus.

Theorem 5160.2 (Bounded self-adjoint operator NameCert obligations)**.** *For every accepted* $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$ *carrier*

$$A = (H, I, G, N, J, S, R, T, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedSelfAdjointOperator}^{\uparrow}}$ *surface consists exactly of carrier admission, Hilbert-source exposure, inner-product readback, bounded graph admission, operator-norm control, adjoint witness display, self-adjoint symmetry, spectral-consumer boundary exposure, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.BoundedSelfAdjointOperatorUp.BoundedSelfAdjointOperatorTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5160.1. The accepted packet exposes exactly the Hilbert, inner-product, graph, norm, adjoint, symmetry, spectral-boundary, transport, replay, provenance, and naming rows.

The boundedness obligation is read from G, N. The graph row records the displayed operator action, and the operator-norm row supplies the finite bound consumed by downstream operator-theory packets.

The adjoint and symmetry obligations are read from I, J, S. The inner-product row provides the comparison surface, the adjoint witness row records the transpose read, and the self-adjoint row identifies the accepted operator with that witness.

Finally R, T, C, P, L keep every normal-operator or spectral-radius consumer behind the same finite packet. Since no carrier row stores a spectral measure, eigenbasis, compactness assertion, quotient operator equality, unbounded-domain extension, or functional calculus, the listed rows exhaust the local NameCert obligations. □

Theorem 5160.3 (Bounded self-adjoint operator adjoint symmetry)**.** *For every accepted* BoundedSelfAdjointOperat *carrier $A = (H, I, G, N, J, S, R, T, C, P, L)$, any consumer that reads the operator graph through G and the adjoint witness through J must read the self-adjoint symmetry row S over the same Hilbert and inner-product rows H, I before it can export a* $\mathsf{NormalOperator}^{\uparrow}$ *or* $\mathsf{SpectralRadius}^{\uparrow}$ *boundary read.*

Proof. Begin with the graph and adjoint rows. The graph row G is typed over the Hilbert carrier H, while the adjoint witness J is read through the inner-product row I over the same source.

Apply the NameCert obligations above. They require S to be displayed as the row binding the operator graph and adjoint witness before the packet is consumed as self-adjoint data.

The spectral boundary row R is downstream of this symmetry check. Transport, replay, provenance, and local naming through T, C, P, L preserve the same route, so no consumer can replace the displayed adjoint symmetry by a host operator equality, spectral decomposition, compactness claim, or functional-calculus object. □

Theorem 5160.4 (Bounded self-adjoint spectral-measure consumer boundary)**.** *Every consumer that asks for spectral-measure, functional-calculus, or spectral-theorem data from an accepted* BoundedSelfAdjointOpe *packet must first pass through the displayed bounded operator graph G, operator-norm row N, adjoint witness J, and self-adjoint symmetry row S over the same Hilbert and inner-product rows H, I. Projection-valued measure data is then available only through the finite projection-ledger boundary of* $\mathsf{SpectralMeasure}^{\uparrow}$ *in chapter 298. The bounded self-adjoint packet alone cannot export projection-valued measure data, spectral projections, functional-calculus rows, selected decompositions, or a spectral theorem proof.*

Proof. Start with the local NameCert obligations of Definition 5160.2. They expose the bounded operator graph, norm row, adjoint witness, self-adjoint symmetry, spectral-consumer boundary, transport, replay, provenance, and local naming rows, and they include no projection-valued-measure coordinate.

Use the adjoint-symmetry theorem Definition 5160.3. It forces every normal-operator or spectral-radius read through G, J, S over the same H, I rows before the spectral boundary row R can be consumed.

For a projection-valued-measure read, the consumer must enter the separate $\mathsf{SpectralMeasure}^{\uparrow}$ packet of chapter 298. Its public consumer boundary supplies the finite projection ledger, scalar measure windows, transport, replay, provenance, and local naming rows required before spectral-theorem or functional-calculus data may be read.

Thus the bounded self-adjoint packet provides only the graph, norm, adjoint, and symmetry source surface. The projection-valued measure surface belongs to $\mathsf{SpectralMeasure}^{\uparrow}$, so a consumer cannot extract spectral measures, functional calculus, selected decompositions, or spectral-theorem data from $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$ alone. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedSelfAdjointOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedSelfAdjointOperator}^{\uparrow}$ packet is generated from a Hilbert carrier, an inner-product row, a bounded operator graph, an operator-norm bound, an adjoint witness, a self-adjoint symmetry row, a spectral-consumer boundary, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedSelfAdjointOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5160.1, Definition 5160.2, and Definition 5160.3 seed the bounded self-adjoint operator packet over Hilbert, inner-product, operator-norm, normal-operator, and spectral-radius rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedSelfAdjointOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close spectral measures, positive-operator calculus, compactness, diagonalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded graph admission, norm control, adjoint witness exactness, self-adjoint symmetry, spectral-consumer boundary, transport, replay, provenance, and local naming for the same finite packet.

Chapter 5161

A Concrete Naming Certificate for $\mathsf{CompletelyRegularSpace}^{\uparrow}$

Following Kelley and Engelking, the $\mathsf{CompletelyRegularSpace}^{\uparrow}$ packet records the classical point-versus-closed-set real-valued separation surface as finite BEDC naming data. It sits between the topology source of chapter 88, the point-versus-closed-set neighbourhood route of chapter 5147, the real-number carrier of chapter 38, and the Urysohn-style function consumer of **??**. The packet names only the displayed continuous real-valued separator route; it does not import host powersets, arbitrary function spaces, selected complements, normality, metrization, or ambient completeness.

Definition 5161.1 (Completely regular space carrier)**.** *A* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, R, C, F, H, K, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the open-index classifier, displayed neighbourhood reads, and closed-row vocabulary available to the packet.*
2. *R is the* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *source row, tying the point-versus-closed-set request to the regular neighbourhood-separation route already named over T.*
3. *C is the closed-set and point request row, displaying the inspected point, the closed row avoided by that point, and the membership and nonmembership reads over the same topology source.*
4. *F is the* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator row. It records a displayed map row, a zero read on the closed-set row, a unit read at the point, and a continuity replay over T.*
5. *H records componentwise* hsame *transport for T, R, C, F, K, P, N and for the displayed real-valued separation reads.*
6. *K records displayed* Cont *replay from a completely-regular request through the topology source, regular-space source, request row, separator row, transport, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{CompletelyRegularSpace}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{RegularTopologicalSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UrysohnLemma}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletelyRegularSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through T, R, C, F, H, K, P, N. It has no coordinate for a host powerset, an external function space, a chosen complement, normal-space Urysohn data, metrization, host real completeness, Lean verification, or bridge checking.

Theorem 5161.2 (Completely regular space NameCert obligations)**.** *For every accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *carrier*

$$Q = (T, R, C, F, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompletelyRegularSpace}^{\uparrow}}$ *surface consists exactly of topology-source admission through* T*; regular-space source admission through* R*; closed-set and point request admission through* C*;* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator exactness through* F*; componentwise* hsame *transport through* H*; displayed* Cont *replay through* K*; provenance through* P*; and local naming through* N*.*

Proof. Project the carrier of Definition 5161.1. The accepted packet displays the topology source, the regular-space source, the point-versus-closed-set request, the real-valued separator row, and the structural rows.

The request row is read before any function row. The row C binds the inspected point and the closed row to the same $\mathsf{Topology}^{\uparrow}$ source, while R supplies only the regular neighbourhood route named by chapter 5147.

The separator obligation is exactly F. It records the $\mathsf{Real}^{\uparrow}$ codomain, the $\mathsf{ContinuousMap}^{\uparrow}$ continuity replay, the zero read on the closed row, and the unit read at the displayed point, so the certificate has no coordinate for a host function space or a selected complement.

Finally H, K, P, N transport, replay, package, and name the same finite route. Since no further coordinate is present, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CompletelyRegularSpaceUp.CompletelyRegularSpaceCarrier namecert obligations`

Theorem 5161.3 (Real-valued separation window)**.** *Every real-valued separation read exported by an accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet must pass through* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{RegularTopologicalSpace}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{ContinuousMap}^{\uparrow}$ *before any* $\mathsf{UrysohnLemma}^{\uparrow}$*,* $\mathsf{UrysohnMetrization}^{\uparrow}$*, or metrization consumer reads it. The consumer receives only the finite row sequence* T, R, C, F, H, K, P, N *and cannot read a host powerset, a selected complement, an ambient function-space object, host real completeness, or an untracked Urysohn separator from the packet.*

Lean4 ✓ `BEDC.Derived.CompletelyRegularSpaceUp.CompletelyRegularSpaceCarrier real valued separation window`

Proof. Begin with the source rows. The topology row T supplies the displayed open and closed vocabulary, and the regular-space row R ties the request to the point-versus-closed-set route already named over that topology.

The closed-set and point request row C is the only entry point for the separation problem. It must be visible before the real-valued separator is read, so no later consumer can replace the request by an ambient complement or by a host powerset argument.

The separator row F then displays both codomain and continuity: it is a $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ row with the required zero and unit reads. The transport row H and replay row K preserve the same sequence before provenance and naming are inspected.

Thus an Urysohn-style or metrization-facing consumer reads the named finite route through $\mathsf{Topology}^{\uparrow}$, $\mathsf{RegularTopologicalSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{ContinuousMap}^{\uparrow}$. It receives no stronger normality theorem, no selected separating function outside F, and no ambient completeness principle from the completely regular packet. □

Theorem 5161.4 (Completely regular space topology dependency route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *real-valued separator read factors through the* $\mathsf{Topology}^{\uparrow}$ *open and closed rows of chapter 88 and the Urysohn-facing route of* **??** *before it exports a* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator. The*

exported route is a finite dependency chain through T, R, C, F, H, K, P, N; it is not a host powerset argument, a selected complement, an ambient function-space object, or an untracked Urysohn separator.

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$ of Definition 5161.1. The topology row T is the only source of open and closed vocabulary, so every point-versus-closed-set request is read through the finite topology surface before a separator row can be inspected.

The request then passes through R and C. These rows bind the regular-space neighbourhood route and the displayed point and closed row to the same $\mathsf{Topology}^{\uparrow}$ source; they do not create a host complement or an external powerset carrier.

Only after those rows are present may the separator row F be exported as a $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ read. The Urysohn-facing sibling route therefore consumes the displayed topology dependency and the separator row together, while H, K, P, N transport, replay, source, and locally name the same finite chain. □

Theorem 5161.5 (Completely regular space uniformity-forward route)**.** *Any uniform-space or continuous-map consumer of a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator row may read that separator only through the displayed topology source, regular-space source,* $\mathsf{Real}^{\uparrow}$ *codomain, and Urysohn-facing rows of the packet. In particular, the route to chapter 339 and chapter 59 is the finite dependency chain T, R, C, F, H, K, P, N, not a metrization, compactness, host powerset, selected complement, or ambient function-space theorem.*

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$ of Definition 5161.1. The separator row F is not public until the topology row T, regular-space row R, and point-versus-closed-set request row C have all been displayed.

Apply the topology dependency route of Definition 5161.4. It keeps every separator read attached to the $\mathsf{Topology}^{\uparrow}$ open and closed rows and to the Urysohn-facing sibling route before the $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ separator is exported.

Finally use the real-valued separation window of Definition 5161.3. A uniform-space or continuous-map consumer receives only the displayed separator route through T, R, C, F, H, K, P, N. Because the carrier has no coordinates for metrization, compactness, host powersets, selected complements, ambient function spaces, or host real completeness, the route is a dependency edge rather than a stronger topological theorem. □

Theorem 5161.6 (Completely regular space realcompactification consumer route)**.** *Every* $\mathsf{HewittRealcompactificati}$ *consumer of a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet first reads the displayed real-valued separator route through chapter 8494. The consumer receives only the finite chain T, R, C, F, H, K, P, N and cannot treat the packet as a selected compactification point, a chosen z-ultrafilter, an ambient product of Real lines, or a host Real-completeness theorem.*

Proof. Start with the real-valued separation window and topology dependency route, Definition 5161.3 and Definition 5161.4. They expose the separator only through the topology source, regular-space source, request row, Real-valued continuous-map row, and structural support rows.

Apply the uniformity-forward route of Definition 5161.5. It keeps the separator public only as the finite dependency chain T, R, C, F, H, K, P, N, with no compactness, powerset, complement, or ambient function-space coordinate.

The Hewitt realcompactification chapter consumes that separator route as a dependency edge. Because the carrier has no row for a selected compactification point, a z-ultrafilter choice, an ambient Real product, or host Real completeness, the consumer cannot read any of those objects from the completely regular packet. □

Theorem 5161.7 (Completely regular space Urysohn consumer route)**.** *Any Urysohn-style separation consumer of an accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet must read the displayed topology source of chapter 88, the real-valued separation window, the uniformity-forward route, and the provenance rows before exporting a function-separation packet through* **??**. *The exported route is the finite chain* T, R, C, F, H, K, P, N*; it is not a global normality theorem, a Tietze-extension theorem, a selected complement, or an ambient function-space construction.*

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$ of Definition 5161.1. The topology row T supplies the open and closed vocabulary, while R and C bind the point-versus-closed-set request to that same displayed topology source.

Apply the real-valued separation window of Definition 5161.3. It forces the separator to pass through the $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ row F before any Urysohn-style consumer reads a function-separation packet.

Use the uniformity-forward route of Definition 5161.5. It keeps the separator on the finite route through the displayed topology, real-valued separation, and structural rows, and the provenance row P records the same route before local naming by N.

Thus the Urysohn-facing consumer receives only T, R, C, F, H, K, P, N. Since no carrier coordinate supplies normality, a Tietze-extension operator, a selected complement, or an ambient function space, the route is a dependency edge rather than a stronger separation theorem. □

Falsifiable prediction: Every accepted $\mathsf{CompletelyRegularSpace}^{\uparrow}$ read exposes a $\mathsf{Topology}^{\uparrow}$ source row, a $\mathsf{RegularTopologicalSpace}^{\uparrow}$ source row, a closed-set and point request row, a $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ separator row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, regular-topological-space, real, continuous-map, urysohn-lemma: $\mathsf{CompletelyRegularSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds real-valued point-versus-closed-set separation as a finite NameCert route between topology and Urysohn-style consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletelyRegularSpace}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletelyRegularSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5161.1, Definition 5161.2, and Definition 5161.3 seed the finite real-valued point-versus-closed-set separation route over topology, regular-space, real, continuous-map, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletelyRegularSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close normality, Urysohn function construction, metrization, host powersets, arbitrary function spaces, selected complements, host real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request-row exactness, real-valued continuous separator exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompletelyRegularSpaceUp.

Constructive story: A $\mathsf{CompletelyRegularSpace}^{\uparrow}$ packet is generated from a Topology source row, a RegularTopologicalSpace source row, a closed-set and point request row, a Real-valued ContinuousMap separator row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Every completely regular separator route factors through the topology source of chapter 88, the point-versus-closed-set regularity source of chapter 5147, the real seal of chapter 38, the continuous-map surface of chapter 59, and the Urysohn consumer of **??**. The top-level $\mathsf{CompletelyRegularSpace}^{\uparrow}$ hub therefore routes its finite separator rows through those siblings before any Urysohn-style reader can consume the displayed real-valued function. CompletelyRegularSpaceUp consumes the separation-function route of **??** together with the ambient chapter 88 carrier, and this dependency is on the displayed Urysohn separation row rather than on a general choice principle or a compactness principle.

Theorem 5161.8 (Completely regular topology–Urysohn forward route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separation read consumes the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88 and the* $\mathsf{UrysohnLemma}^{\uparrow}$ *separation row of* **??** *before any compactification, uniformity, or Kelleyfication consumer uses it.*

Proof. Start with the topology–Urysohn dependency route recorded in Definition 5161.4. It keeps the completely regular separator attached to the topology source, the point-versus-closed-set separation row, the Real-valued continuous-map row, and the Urysohn-facing consumer route.

Now apply the sibling dependency route of Definition 5161.11. That route places every compactification, uniformity, or real-valued separation consumer after the displayed topology carrier and the Urysohn-style separation route.

The forward dependency is therefore fixed before downstream use: a consumer reaches the completely regular row only after the $\mathsf{Topology}^{\uparrow}$ carrier of chapter 88 and the $\mathsf{UrysohnLemma}^{\uparrow}$ separation row of **??** have both been displayed. Compactification, uniformity, and Kelleyfication rows may reuse that separator, but they cannot read it as a bare host function or as a compactness principle. □

Theorem 5161.9 (Completely regular space lattice dependency route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator read first consumes the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88 and the* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *point-versus-closed-set source of chapter 5147. Only after those rows are displayed may the packet export its* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator row to* $\mathsf{UrysohnLemma}^{\uparrow}$ *or* $\mathsf{UrysohnMetrization}^{\uparrow}$ *consumers. The route is the finite lattice edge* T, R, C, F, H, K, P, N *of Definition 5161.1, not a host powerset, selected complement, arbitrary function space, normality theorem, or metrization proof.*

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$. The topology row T supplies the displayed open and closed vocabulary, while the regular topological row R binds the point-versus-closed-set request to that same source.

The request row C must be visible before the separator row F is exported. Since F is a $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ row, an Urysohn-facing consumer receives a displayed separator rather than an ambient family of functions.

Finally the structural rows H, K, P, N preserve the same route for $\mathsf{UrysohnLemma}^{\uparrow}$ and $\mathsf{UrysohnMetrization}^{\uparrow}$ consumers. No carrier coordinate supplies a hidden powerset, selected complement, normality theorem, or metrization proof, so the lattice edge is exactly the displayed topological dependency route. □

Theorem 5161.10 (Completely regular space dossier lattice anchor)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *dossier read that enters through the lattice route must first consume the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88, the point-versus-closed-set regularity row of chapter 5147, and the Urysohn-style separation sibling of* **??**. *The exported route is the displayed finite separator lattice* T, R, C, F, H, K, P, N *of Definition 5161.1, not a host family of separators, an ambient normality theorem, a compactness principle, or a selected external function space.*

Proof. Project the carrier rows from Definition 5161.1. The topology row T fixes the open and closed vocabulary, while R and C bind the regular point-versus-closed-set request to the same source.

Now apply the topology–Urysohn dependency route and the Urysohn sibling route. They place the real-valued separator row F after chapter 88 and **??**, so every later dossier consumer reads a finite separation route rather than a detached function-space object.

The structural rows H, K, P, N then transport, replay, package, and name that same route. Since no carrier coordinate supplies host separator families, normality, compactness, or an external function space, the dossier lattice anchor is exactly the displayed sibling dependency path. □

Theorem 5161.11 (Completely regular space sibling dependency)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *consumer reads the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88 together with the local Urysohn-style separation route before any compactification, uniformity, or real-valued separation consumer can use the completely regular row.*

Proof. The local topology–Urysohn dependency route records the displayed topology carrier, the point-versus-closed-set separation data, the continuous real-valued separator row, and the Urysohn consumer route as sibling inputs to the completely regular packet.

When the packet exports its continuous-separation ledger, the $\mathsf{CompletelyRegularSpace}^{\uparrow}$ reader consumes the topology source through chapter 88 and the metrization-facing sibling chapter 3337. Thus the ledger is horizontally tied to the topology carrier and the Urysohn metrization boundary before any consumer treats the real-valued separator as metrization data.

Any later compactification or uniformity reader therefore reaches $\mathsf{CompletelyRegularSpace}^{\uparrow}$ only after those rows are present. The dependency is a finite topological lattice route through chapter 88 and the local Urysohn-style child, not a hidden compactness principle, a choice principle, or a quotient of separation functions. □

Theorem 5161.12 (Completely regular space compact-open route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *compactification or uniformity consumer reads the* $\mathsf{Topology}^{\uparrow}$ *carrier, the* $\mathsf{UrysohnMetrization}^{\uparrow}$ *separation boundary, and the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *or* $\mathsf{CompactOpenExhaustion}^{\uparrow}$ *compact-open rows before treating real-valued separators as uniform data.*

Proof. Start with Definition 5161.11. It places every completely regular consumer after the topology carrier of chapter 88 and the local Urysohn-style separation route.

The metrization-facing sibling chapter 3337 supplies the boundary that turns real-valued separators into metric-facing data. Compact-open consumers then read that boundary through chapter 4138 or through the finite exhaustion route of chapter 6756.

Thus a compactification or uniformity reader cannot consume a separator as bare host function data. It must first pass through the topology row, the Urysohn metrization row, and the compact-open rows named by the sibling lattice. □

Theorem 5161.13 (Completely regular space sibling lattice forward route)**.** *Every accepted* $\mathsf{CompletelyRegularSpac}$ *read used by a compact-open or Kelleyfication consumer must first expose the sibling rows for* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{UrysohnLemma}^{\uparrow}$*,* $\mathsf{CompactHausdorff}^{\uparrow}$*, and* $\mathsf{SierpinskiSpace}^{\uparrow}$*. The forward route is a finite sibling lattice edge: the consumer reaches the completely regular packet through those displayed rows before it may read compact-open function-space data, Kelleyfication reflections, or real-valued separation output.*

Proof. Start with the local carrier of Definition 5161.1. Its point-versus-closed-set separation rows are usable only after the topology source and the real-valued separator route are displayed.

Attach the sibling rows. The $\mathsf{Topology}^{\uparrow}$ row supplies the open-set carrier, $\mathsf{UrysohnLemma}^{\uparrow}$ supplies the separation witness route, $\mathsf{CompactHausdorff}^{\uparrow}$ supplies the compact Hausdorff consumer boundary, and $\mathsf{SierpinskiSpace}^{\uparrow}$ supplies the open-set classifier read used by lattice-facing consumers.

Now read the downstream consumers. A compact-open consumer reaches the packet through the compact-open route of Definition 5161.12; a Kelleyfication consumer reaches it through the

Kelleyfication route of chapter 6830. Both routes therefore see the four sibling rows before treating the completely regular packet as a source.

Since the four rows are finite sibling entries, the route is not a bare host topology principle, a selected family of separators, or an unlogged compactness assertion. It is a forward lattice edge from the displayed topological siblings into the compact-open and Kelleyfication readers. □

Theorem 5161.14 (Completely regular space compact-open forward route)**.** *Every* $\mathsf{CompactOpenTopology}^{\uparrow}$ *or* $\mathsf{Kelleyfication}^{\uparrow}$ *consumer that reads a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet first passes through the displayed topology source of chapter 88, the point-versus-closed-set regularity source of chapter 5147, the* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator route of chapter 130, and the Urysohn-facing route of* **??**. *The forward read then preserves componentwise transport, displayed replay, provenance, and local naming before any compact-open or Kelleyfication window may consume the separator.*

Proof. Read the completely regular packet through the local topology–Urysohn dependency route and the sibling dependency theorem. These rows expose the topology carrier, the regular topological source, the point-versus-closed-set separation data, and the continuous real-valued separator before any later function-space consumer appears.

A compact-open reader can therefore inspect only the displayed $\mathsf{Topology}^{\uparrow}$ source, $\mathsf{RegularTopologicalSpace}^{\uparrow}$ boundary, $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ separator, and Urysohn-facing separation route. The $\mathsf{CompactOpenTopology}^{\uparrow}$ window receives this separator as finite graph and target-open data, not as a host function space or an unrestricted family of separating maps.

A Kelleyfication reader uses the same route after the compact-open ledger has been displayed. Transport, replay, provenance, and local naming remain the rows attached to the original $\mathsf{CompletelyRegularSpace}^{\uparrow}$ packet, so the consumer gains no quotient-open equality, choice-selected separator, ambient compactness principle, or external topology oracle. □

Every $\mathsf{CompletelyRegularSpace}^{\uparrow}$ separator read is dependency-visible as a $\mathsf{Topology}^{\uparrow}$ source row followed by the $\mathsf{UrysohnLemma}^{\uparrow}$ separation route and the $\mathsf{ContinuousMap}^{\uparrow}$ real-valued map row. Thus a downstream compact-open, uniformity, or Kelleyfication consumer receives the displayed point-versus-closed-set separator only through the existing topology and Urysohn sibling lattice of Definition 5161.21; it cannot replace that route by a host family of functions, a selected separator, or an ambient compactness principle.

Theorem 5161.15 (Completely regular space topology–uniformity route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet that feeds a compact-open or uniformity consumer first exposes the* $\mathsf{Topology}^{\uparrow}$ *source of chapter 88, the* $\mathsf{UrysohnLemma}^{\uparrow}$*-style separation route of* **??**, *and the finite separator rows of Definition 5161.1. A* $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactOpenUniformConvergence}^{\uparrow}$, *or* $\mathsf{UniformConvergenceSpace}^{\uparrow}$ *consumer therefore receives the row sequence* T, R, C, F, H, K, P, N *through local naming, not a host family of real-valued maps or an ambient uniformity oracle.*

Proof. Project the carrier rows of Definition 5161.1. The topology row T supplies the open-set vocabulary, while R, C, F record the regularity request and the displayed real-valued separator.

Apply the topology–Urysohn sibling lattice of Definition 5161.21. It places the separator after the topology source and Urysohn-style separation route before any compact-open read.

The same finite row sequence is then available to compact-open uniformity consumers. Since H, K, P, N only transport, replay, source, and name the displayed separator packet, no downstream consumer obtains an unrestricted function space, a selected family of separators, or a detached uniformity surface. □

Theorem 5161.16 (Completely regular space Kelleyfication forward route)**.** *Every* $\mathsf{Kelleyfication}^{\uparrow}$ *reader of a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator must first pass through the displayed* $\mathsf{Topology}^{\uparrow}$ *source, the* $\mathsf{RegularTopologicalSpace}^{\uparrow}$ *point-versus-closed-set boundary, the* $\mathsf{Real}^{\uparrow}$*-valued* $\mathsf{ContinuousMap}^{\uparrow}$ *separator, and the Urysohn-facing route before the compact-open ledger consumes the separator. The Kelleyfication window therefore reads the same compact-open consumer path named in Definition 5161.14; it receives no host separator family, quotient-open equality, selected real-valued map, or external compactness principle.*

Proof. Start with the compact-open forward route Definition 5161.14. It already places every compact-open or Kelleyfication consumer after the topology source, regular topological source, real-valued continuous-map separator, and Urysohn-facing separation route.

The Kelleyfication reader can only consume that separator after the compact-open window has displayed the finite graph and target-open data. Since the hub packet supplies no additional carrier field, the Kelleyfication handoff preserves the same topology, regularity, Real, ContinuousMap, and Urysohn rows.

Transport, replay, provenance, and local naming remain attached to the original $\mathsf{CompletelyRegularSpace}^{\uparrow}$ packet. Thus the forward route reaches $\mathsf{Kelleyfication}^{\uparrow}$ only through the compact-open consumer path and cannot export a selected separator family, quotient-open equality, or ambient compactness principle. □

Theorem 5161.17 (Completely regular space Urysohn sibling route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *carrier read that separates a point from a closed set consumes the* $\mathsf{Topology}^{\uparrow}$ *open-set classifier of chapter 88 and the Urysohn-style functional separation route of* **??** *before any compactification or uniformity consumer reads it. The exported separator is the finite row sequence named by the local topology–Urysohn dependency child, not a host powerset separation theorem, a selected complement, or an unrestricted family of real-valued functions.*

Proof. Read the carrier through the topology–Urysohn dependency route and the sibling dependency theorem. These expose the displayed topology carrier, the point-versus-closed-set regularity row, the $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ separator, and the Urysohn-facing route before the completely regular row is available to downstream consumers.

Any compactification or uniformity reader therefore receives only the finite separation certificate already tied to $\mathsf{Topology}^{\uparrow}$ and $\mathsf{UrysohnLemma}^{\uparrow}$. The route has no coordinate for a hidden compactification, a uniformity generated by an ambient family, host function equality, selected complements, or choice. □

Theorem 5161.18 (Completely regular space regular-open sibling route)**.** *Every accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator route consumed by* $\mathsf{RegularOpenSet}^{\uparrow}$*,* $\mathsf{CozeroSet}^{\uparrow}$*, or* $\mathsf{ZeroSet}^{\uparrow}$ *factors through the displayed topology row, the real-valued separator row, and the continuous-map row of Definition 5161.1. The sibling consumer may read the open or zero-locus boundary only through those rows; it receives no host powerset, selected complement, detached open-set lattice, or classical complement operation from the completely regular packet.*

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$ of Definition 5161.1. The topology row T fixes the open and closed vocabulary, while the request row C and separator row F keep the point-versus-closed-set separation attached to a displayed $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ route.

Read the regular-open sibling through chapter 8547. It can consume only the topology-facing open and closure-interior information already exposed by T and the separator route; it does not obtain a host powerset or a detached Boolean algebra of opens.

Read the cozero and zero-set siblings through chapter 1857 and chapter 1858. Their positive-apartness and zero-locus routes must use the displayed real-valued continuous-map row F, transported by H and replayed by K, before provenance and local naming are inspected.

Thus the sibling route is exactly the finite topology, real-valued separator, and continuous-map path carried by T, C, F, H, K, P, N. Since no carrier coordinate supplies selected complements, host powersets, or a classical complement operation, none is exported to the regular-open, cozero, or zero-set consumers. □

Theorem 5161.19 (Completely regular space compact Hausdorff Urysohn route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *read that exports the local classifier to a compact or Hausdorff consumer first consumes the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88, the* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation boundary of chapter 1925, the compact metric sibling route of chapter 129, and the Urysohn separation route of* **??**. *The exported classifier is therefore the finite row route* T, R, C, F, H, K, P, N *of Definition 5161.1, not a host compactness theorem, a selected separating function, an unrestricted powerset, or an ambient normal-space principle.*

Proof. Start with the local carrier and NameCert obligations of Definition 5161.1 and Definition 5161.2. They expose the topology row, the regular-space row, the point-versus-closed-set request, the real-valued continuous-map separator, and the structural rows before any classifier is public.

Apply the Urysohn sibling route of Definition 5161.17. It forces the separator to pass through the displayed Urysohn separation row rather than through a chosen host function family.

Now read the compact and Hausdorff siblings. The Hausdorff row supplies the separation boundary, and the compact metric row supplies only a displayed compact metric dependency; neither row adds a powerset, a selected separator, or an ambient compactness theorem.

The structural rows H, K, P, N preserve the same route for downstream compact or Hausdorff consumers. Hence every exported classifier remains tied to the Urysohn and compact-Hausdorff sibling lattice before it is read. □

Theorem 5161.20 (Completely regular space Tychonoff compactification link)**.** *Every Tychonoff-style or compactification-facing consumer of* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *must pass through the displayed topology, Hausdorff or compact-metric sibling, and Urysohn separator rows before it reads separator data. In the local lattice this routes the exported separator toward the compact-open topology and Kelleyfication consumers of chapter 4138 and chapter 6830. It does not add a host product compactness theorem, a selected compactification point, an unrestricted Tychonoff theorem, or an ambient powerset of separators.*

Proof. Read the compact-Hausdorff Urysohn route of Definition 5161.19. It exposes the topology, Hausdorff, compact metric, and Urysohn sibling rows before the local separator classifier is exported.

The compact-open topology and Kelleyfication consumers inspect continuous-map and compact-window data only after those rows are visible. Thus a Tychonoff-style compactification read cannot bypass the displayed $\mathsf{Topology}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, and $\mathsf{UrysohnLemma}^{\uparrow}$ dependencies.

Since the accepted $\mathsf{CompletelyRegularSpace}^{\uparrow}$ carrier has no coordinate for host product compactness, selected compactification points, or unrestricted powersets of separators, the downstream compactification route remains the stated finite row route. □

Theorem 5161.21 (Completely regular space topology–Urysohn sibling lattice)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet reads its real-valued separating-function route through the* $\mathsf{Topology}^{\uparrow}$ *carrier of chapter 88 and the Urysohn-style sibling route of* **??** *before any* $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{CompactOpenExhaustion}^{\uparrow}$, $\mathsf{Kelleyfication}^{\uparrow}$*, or compact-Hausdorff consumer may use the exported separator. The lattice edge is the finite row sequence* T, R, C, F, H, K, P, N *of Definition 5161.1, together with Definition 5161.17 and Definition 5161.19; it is not a host powerset argument, an unrestricted function-space construction, a selected complement, normality, metrization, or ambient compactness.*

Proof. Project the carrier rows of Definition 5161.1. The topology row T supplies the displayed open and closed vocabulary, while the request row C and separator row F remain tied to that same source.

Apply the Urysohn sibling route of Definition 5161.17. It places the separating-function read after the topology carrier and the Urysohn-style separation route, rather than after a detached host family of real-valued functions.

Now use the compact-Hausdorff Urysohn route of Definition 5161.19. Compact-open and Kelleyfication consumers may reuse the exported separator only after the same topology and Urysohn rows have been displayed.

The structural rows H, K, P, N preserve this finite lattice edge for downstream consumers. Since the carrier has no coordinate for host powersets, unrestricted function spaces, selected complements, normality, metrization, or compactness, none is exported by the sibling lattice. □

The downstream compact-Hausdorff and Urysohn consumers read $\mathsf{CompletelyRegularSpace}^{\uparrow}$ as a displayed separation substrate, not as an ambient topology principle. A consumer may pass from the point-versus-closed-set rows T, R, C, F, H, K, P, N of Definition 5161.1 to the Urysohn separation route of **??** and then to the compact-Hausdorff sibling lattice of Definition 5161.19; it may not replace that route by host powersets, unrestricted real-valued function spaces, selected complements, normality, metrization, or an unlogged compactness theorem.

Theorem 5161.22 (Completely regular space sibling lattice)**.** *Every accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator route is visible in the sibling lattice*

$$\mathsf{Topology}^{\uparrow},\ \mathsf{RegularTopologicalSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{UrysohnLemma}^{\uparrow}.$$

The topology row supplies open and closed vocabulary, the regular-space row supplies the point-versus-closed-set request route, the Real and continuous-map rows carry the displayed separator, and the Urysohn row is only a consumer of that finite route. The packet does not identify this lattice with normality, metrization, a host function space, or a selected complement.

Proof. Project the carrier $Q = (T, R, C, F, H, K, P, N)$ from Definition 5161.1. The rows T, R, C, F already determine the topology source, regular-space request, point-versus-closed-set data, and Real-valued continuous separator.

Use the real-valued separation window and Urysohn consumer route, Definition 5161.3 and Definition 5161.7. They require every Urysohn-facing read to pass through the same topology, regular-space, Real, and continuous-map rows before any separation consumer receives data.

The structural rows H, K, P, N transport, replay, source, and name the displayed lattice. Since no carrier coordinate supplies normality, metrization, a host function space, or a selected complement, the sibling lattice remains a dependency route rather than a stronger topological theorem. □

Theorem 5161.23 (Completely regular space topology-uniformity consumer lattice)**.** *Every* CompletelyRegularSpa *real-valued separator read reaches compact-open, Kelleyfication, Urysohn, and uniformity consumers only after the displayed topology source of chapter 88, the Urysohn-style separator route of* **??***, the compact-open route of Definition 5161.14, and the uniformity-forward route of Definition 5161.5 have all remained visible. Thus downstream consumers read the finite row sequence* T, R, C, F, H, K, P, N *of Definition 5161.1; they do not receive a host family of real-valued functions, selected complements, normality, metrization, or ambient compactness.*

Proof. Start with the topology–Urysohn sibling lattice, Definition 5161.21. It places the separator after the displayed topology carrier and the Urysohn-style route before compact-open or Kelleyfication consumers may inspect it.

Apply the compact-open forward route and Kelleyfication forward route, Definition 5161.14 and Definition 5161.16. These keep compact-open and Kelleyfication reads downstream of the same topology, regular-space, Real, continuous-map, and Urysohn rows.

Finally use the uniformity-forward route of Definition 5161.5. Uniformity consumers receive only the displayed separator chain T, R, C, F, H, K, P, N, so the sibling lattice exports a concrete dependency edge rather than a stronger host topological theorem. □

Theorem 5161.24 (Completely regular space subspace-uniformity forward link)**.** *Every subspace-topology or uniformity consumer of a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator reads the packet only after the displayed* $\mathsf{Topology}^{\uparrow}$ *source of chapter 88, the Urysohn-style separator route of* **??***, the compact-open route of Definition 5161.14, and the uniformity-forward route of Definition 5161.5 are all visible. In particular, the subspace-topology route of chapter 5993 may consume* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *only through the topology-uniformity lattice of Definition 5161.23; it receives no selected separator family, ambient normality, metrization, host compactness, or unrestricted uniformity theorem.*

Proof. Begin with the topology-uniformity consumer lattice. It already forces every real-valued separator read through the displayed topology source, Urysohn-style separator route, compact-open route, and uniformity-forward route.

Now attach the subspace-topology consumer. Since chapter 5993 is allowed to read the separator only after the same topology and uniformity rows are visible, the subspace route is downstream of the lattice rather than a separate ambient-space theorem.

Thus the exported link is the finite dependency route through T, R, C, F, H, K, P, N. None of its rows supplies selected separator families, normality, metrization, host compactness, or an unrestricted uniformity principle. □

Theorem 5161.25 (Completely regular space fine-uniformity forward link)**.** *Every* $\mathsf{FineUniformity}^{\uparrow}$ *consumer of a* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet must first read the displayed real-valued separation row together with the shared* $\mathsf{Topology}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$ *rows before exporting an entourage base. The forward link is the finite route*

$$\mathsf{CompletelyRegularSpace}^{\uparrow}, \mathsf{Topology}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FineUniformity}^{\uparrow},$$

with the separator, topology source, real-valued probe, transport, replay, provenance, and local naming rows all visible before the fine-uniformity base is public.

Consequently the fine-uniformity consumer receives only the displayed separator route. It does not receive a host family of all real-valued maps, a selected entourage filter, an ambient powerset uniformity, a metrization theorem, a completion object, or a choice principle for probes.

Proof. Project the $\mathsf{CompletelyRegularSpace}^{\uparrow}$ carrier rows of Definition 5161.1. They expose the topology source, the regularity request, the displayed $\mathsf{Real}^{\uparrow}$-valued $\mathsf{ContinuousMap}^{\uparrow}$ separator, and the structural rows used by downstream consumers.

Now read the $\mathsf{FineUniformity}^{\uparrow}$ carrier of Definition 8906.1. Its complete-regularity coordinate is consumed before the real-valued probe row generates a finite entourage-base ledger, so the fine-uniformity base cannot be exported before the separator route is visible.

The same hsame, Cont, Pkg, and NameCert rows transport and replay the displayed separator into the fine-uniformity packet. Since no carrier field ranges over all real-valued maps or selected filters, the forward link exports only the finite entourage base generated by the displayed probes. □

Theorem 5161.26 (Completely regular space fine-uniformity dependency record)**.** *Every accepted* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *packet consumed as a* $\mathsf{FineUniformity}^{\uparrow}$ *source must expose the topology row of chapter 88, the Urysohn-style separation route of* **??**, *the lattice-facing regular-open rows of chapter 55, and the fine-uniformity carrier of chapter 8906 before any entourage-base consumer reads the exported uniformity row.*

Proof. Project the carrier of Definition 5161.1. Its separator row is attached to the same topology source, regular-space source, real-valued continuous-map row, transport row, replay row, provenance row, and local naming row used by the sibling dependency routes.

Apply the forward link of Definition 5161.25. It sends the displayed separator through the fine-uniformity carrier only after the topology, Urysohn-style separation, and lattice-facing regular-open dependencies have been made public.

Thus the uniformity consumer receives a finite dependency record over the displayed rows. It does not receive an ambient family of all real-valued maps, a selected entourage filter, a host powerset lattice, or a metrization theorem. □

Theorem 5161.27 (Completely regular space fine-uniformity telescope route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *carrier consumed by a* $\mathsf{FineUniformity}^{\uparrow}$ *reader passes through the topology-uniformity route already exposed in Definition 5161.15 and the fine-uniformity carrier of chapter 8906. The telescope read keeps the topology source, real-valued separator, continuous-map probe, transport, replay, provenance, and local naming rows visible before an entourage-base row is public.*

Proof. Project the carrier of Definition 5161.1. The topology row, request row, real-valued continuous separator, transport, replay, provenance, and local naming rows are the only visible source for a uniformity-facing consumer.

Apply the topology-uniformity dependency route. It keeps the separator attached to the same topology and continuous-map rows before any uniformity consumer can read the exported surface.

Finally read the fine-uniformity carrier. Its entourage-base row is generated from the displayed complete-regularity and real-valued probe data, so the telescope route exports no ambient family of all probes, selected entourage filter, host powerset uniformity, metrization theorem, or completion object. □

Theorem 5161.28 (Completely regular Urysohn and continuous-map route)**.** *Every* $\mathsf{CompletelyRegularSpace}^{\uparrow}$ *separator row consumed by an Urysohn-facing or continuous-map-facing chapter must pass through the separator route of* **??** *and the map graph route of chapter 59. The consumer reads the topology source, point-versus-closed-set request, real-valued separator, transport, replay, provenance, and local naming rows of the completely regular packet; it receives no host powerset, selected function space, ambient family of all real-valued maps, or untracked separator.*

Proof. Project the carrier of Definition 5161.1. The separator row F is already a $\mathsf{Real}^{\uparrow}$-valued continuous-map row, and the request rows T, R, C bind it to the displayed topology and closed-set data.

Apply the Urysohn consumer route of Definition 5161.7. It forces any Urysohn-style read to consume the topology source, real-valued separation window, uniformity-facing route, provenance, and local naming before a function-separation packet is public.

Finally read the continuous-map route through chapter 59. Since the carrier has no coordinate for host powersets, selected function spaces, or ambient families of real-valued maps, the separator remains the displayed finite route through T, R, C, F, H, K, P, N. □

Chapter 5162

A Concrete Naming Certificate for $\mathsf{LawsonTopology}^{\uparrow}$

Following Lawson, Scott, and domain-theoretic topology, the $\mathsf{LawsonTopology}^{\uparrow}$ packet records the finite interface where Scott-open observation rows and order-dual compact neighbourhood rows are read through a displayed topology. It sits over the domain/order/topology source chapters chapter 4164, chapter 1860, chapter 88, chapter 52, and chapter 53. The packet names the Lawson carrier without importing an ambient powerset topology, a quotient of opens, or a host dcpo completion.

Definition 5162.1 (Lawson topology carrier)**.** *A* $\mathsf{LawsonTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, O, K, S, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain source row, exposing the* $\mathsf{ScottDomain}^{\uparrow}$ *carrier, directed-join observation interface, and order comparison data consumed by the topology.*
2. *O is the Scott-open row, recording open observations that are upward stable and directed-inaccessible through the displayed domain interface.*
3. *K is the lower-compact row, recording finite order-dual neighbourhood tests generated from principal lower observations and compact exclusions.*
4. *S is the Sierpinski classifier row, reading open predicates through the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observation interface.*
5. *T is the* $\mathsf{Topology}^{\uparrow}$ *handoff row generated by the subbasis consisting of the Scott-open rows in O and the lower-compact rows in K.*
6. *H records componentwise* hsame *transport for the domain row, Scott-open row, lower-compact row, Sierpinski row, topology handoff, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from domain and order observations to the Lawson topology handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{LawsonTopology}^{\uparrow}$, $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LawsonTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, O, K, S, T, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.LawsonTopologyUp.LawsonTopologyTasteGate single carrier alignment`

Theorem 5162.2 (Lawson topology NameCert obligations)**.** *For every accepted* $\mathsf{LawsonTopology}^{\uparrow}$ *carrier*

$$L = (D, O, K, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LawsonTopology}^{\uparrow}}$ *surface consists exactly of the displayed Scott-domain source through D, Scott-open observation through O, lower-compact observation through K, Sierpinski classifier reading through S, topology handoff through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5162.1. The domain row supplies the order carrier inherited from the preorder and poset source chapters, while the Scott-open row supplies the directed-observation side.

The lower-compact row is separate from the Scott-open row. It records the order-dual neighbourhood tests used by the Lawson topology and prevents the packet from collapsing into the Scott topology alone.

The Sierpinski row reads open predicates as observations, and the topology row packages the generated subbasis. Transport, replay, provenance, and local naming are exactly H, C, P, N, so no additional host topology or quotient-open coordinate is available. □

Theorem 5162.3 (Lawson topology Scott-specialization handoff)**.** *Every specialization-order read exported by an accepted* $\mathsf{LawsonTopology}^{\uparrow}$ *carrier factors through the displayed* $\mathsf{ScottDomain}^{\uparrow}$ *source,* $\mathsf{Preorder}^{\uparrow}$ *comparison,* $\mathsf{Poset}^{\uparrow}$ *antisymmetry row,* $\mathsf{SierpinskiSpace}^{\uparrow}$ *open-observation classifier, and* $\mathsf{Topology}^{\uparrow}$ *handoff. The Lawson packet may combine Scott-open and lower-compact observations, but it cannot replace the specialization route by host equality of points, an ambient powerset of opens, or an unrecorded dcpo completion.*

Proof. Start with Definition 5162.2. The only order-facing source is the displayed domain row D, and its comparison data are read through the preorder and poset source surfaces.

Open observations are read through O and then through the Sierpinski classifier row S. The topology handoff T records the Lawson subbasis generated from the Scott-open and lower-compact rows.

Since specialization is recovered from the displayed topology observations, the route must pass through D, O, K, S, T and then through H, C, P, N. None of the carrier rows stores host point equality, a powerset topology, or a completion theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawsonTopology}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawsonTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5162.1, Definition 5162.2, and Definition 5162.3 seed the finite Lawson topology carrier over Scott-open, lower-compact, Sierpinski, topology, preorder, and poset rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawsonTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a Lean theorem, a public dcpo completion, arbitrary powerset topology equality, quotient-open equality, sobriety, compactness, or a bridge check.

Upgrade path: Obligation closure requires checked carrier admission, Scott-open exactness, lower-compact observation stability, Sierpinski classifier reading, topology handoff, specialization-order routing, transport, replay, provenance, and local naming for BEDC.Derived.LawsonTopologyUp.

Constructive story: A $\mathsf{LawsonTopology}^{\uparrow}$ packet is generated from a Scott-domain source row, Scott-open observations, lower-compact order-dual neighbourhood rows, a Sierpinski classifier row, a topology handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5163

A Concrete Naming Certificate for CandidateMediatedSN$^{\uparrow}$

The CandidateMediatedSN$^{\uparrow}$ packet realizes the candidate-mediated branch of chapter 9383 as finite NameCert data. It reads the sibling MetaCIC sources chapter 765, chapter 892, chapter 735, and chapter 740 as route anchors, while keeping candidate evidence explicit before any strong-normalization conclusion is exported. The packet is a normalization-evidence landing, not a blanket theorem that every closed typed MetaCIC term is strongly normalizing without displayed candidate data.

Definition 5163.1 (Candidate-mediated strong-normalization carrier)**.** *A* CandidateMediatedSN$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, E, C, S, A, NF, O, R, H, Q, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row consumed by the endpoint projection.*
2. *E is the candidate-evidence row, allowing Candidate, ClosedCandidate, or CandidateSet membership evidence to be named without erasure.*
3. *C is the candidate-closure row, recording the closure and beta-stability data needed by the candidate route.*
4. *S is the strong-normalization export row downstream of E and C.*
5. *A is the Pi/application adequacy row, keeping function and argument reads inside the same candidate-mediated interface.*
6. *NF is the no-infinite-reduction projection row obtained from the same strong-normalization evidence.*
7. *O is the obstruction row naming subject-reduction, substitution, confluence, and closed-consistency boundaries that remain outside this packet.*
8. *R is the frontier route row, reading* MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICConfluenceAu *and* NormalFormConsistencySeal$^{\uparrow}$ *as named sources rather than replacing them.*
9. *H records componentwise* hsame *transport for the typed term, candidate evidence, closure, strong-normalization export, adequacy, no-infinite-reduction projection, obstruction, route, replay, provenance, and local naming rows.*

10. *Q records displayed* Cont *replay from candidate evidence to the strong-normalization and no-infinite-reduction endpoint rows.*
11. *P is the* Pkg *provenance over* $\mathsf{CandidateMediatedSN}^{\uparrow}$*, chapter 9383, chapter 765, chapter 892, chapter 735, chapter 740,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, C, S, A, NF, O, R, H, Q, P, N$.

Theorem 5163.2 (Candidate-mediated strong-normalization NameCert obligations)**.** *For every accepted* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *carrier*

$$M = (T, E, C, S, A, NF, O, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}$ *surface consists exactly of closed typed term admission through T, candidate evidence through E, candidate closure through C, strong-normalization export through S, Pi/application adequacy through A, no-infinite-reduction projection through NF, obstruction visibility through O, route anchoring through R, componentwise* hsame *transport through H, displayed* Cont *replay through Q,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5163.1. The closed typed term row T names the endpoint shape, but it does not carry strong normalization on its own.

The evidence rows E and C are the positive source. The export row S and no-infinite-reduction row NF are downstream of that source, while A keeps Pi and application reads inside the candidate-mediated route.

The obstruction and route rows are also part of the certificate. They expose the critical-path, normalization-frontier, confluence-audit, and normal-form-consistency boundaries rather than silently converting them into unconditional normalization.

Finally H, Q, P, N provide transport, replay, provenance, and local naming. Since no carrier coordinate stores an evidence-free closed-term strong-normalization theorem, the listed rows exhaust the packet. □

Theorem 5163.3 (Candidate-mediated strong-normalization frontier handoff)**.** *Every strong-normalization read exported by an accepted* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *packet factors through displayed candidate evidence before it reaches a closed typed term endpoint. The route may cite* $\mathsf{MetaCICCriticalPath}^{\uparrow}$*,* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$*, and* NormalFormConsistenc *as sibling anchors, but it cannot erase the Candidate, ClosedCandidate, or CandidateSet evidence row into a global closed-term normalization theorem.*

Proof. Use Definition 5163.2. The endpoint row T is read only after the evidence and closure rows have been opened.

The adequacy and no-infinite-reduction rows are not independent endpoints. They are downstream projections of the same candidate-mediated source, so application, Pi, and infinite-reduction reads remain attached to E and C.

The sibling route row R records where the public MetaCIC boundaries are read. It preserves the normalization frontier and confluence audit as named sources, and therefore blocks the evidence-free closed-term claim. □

Theorem 5163.4 (Candidate-mediated SN Pi-application frontier)**.** *Every public strong-normalization or no-infinite-reduction read that consumes Pi/application adequacy from an accepted* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *carrier*

$$M = (T, E, C, S, A, NF, O, R, H, Q, P, N)$$

must first display closed typed term data through T, candidate or candidate-set evidence through E, candidate closure through C, Pi adequacy and application adequacy through A, the candidate-mediated strong-normalization projection through S, the no-infinite-reduction projection through NF, frontier retention through O, R, componentwise hsame *transport through H, displayed* Cont *replay through Q,* Pkg *provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}$ *row N.*

Consequently the packet denies evidence-free closed-term normalization, unconditional application normalization, global confluence, unrestricted subject reduction, quotient syntax equality, and host normalization.

Proof. Project the carrier of Definition 5163.1. The closed typed term row T and the adequacy row A are visible only as coordinates of the same packet that also contains the candidate evidence E and closure row C.

Apply Definition 5163.2. Its carrier admission, candidate-evidence, candidate-closure, strong-normalization export, Pi/application adequacy, no-infinite-reduction projection, obstruction, route, transport, replay, provenance, and naming clauses give exactly the rows listed in the statement.

Use Definition 5163.3. Strong-normalization and no-infinite-reduction reads remain downstream of candidate evidence, so the Pi and application rows are adequacy rows attached to that evidence rather than independent normalization principles.

The obstruction and route rows retain the MetaCIC frontier. Since no carrier coordinate stores evidence-free normalization, unconditional application normalization, global confluence, unrestricted subject reduction, quotient syntax equality, or host normalization, the public read cannot export any of those conclusions. □

Theorem 5163.5 (Candidate-mediated SN has no hidden oracle)**.** *Every StrongNormalizable row exported by an accepted* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *packet factors through one of the displayed Candidate, ClosedCandidate, or CandidateSet evidence rows named by the carrier. If no such evidence row is displayed, the packet exports only the boundary ledger over* $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, *and it exports no blanket strong-normalization conclusion for all closed typed MetaCIC terms.*

Proof. Read the packet through the normalization audit map and then project the carrier of Definition 5163.1. The only positive normalization source in the carrier is the evidence pair E, C, where E names Candidate, ClosedCandidate, or CandidateSet membership and C records the closure data needed by that evidence.

The exported strong-normalization row S and the no-infinite-reduction row NF sit downstream of those displayed rows. Thus any StrongNormalizable export must pass through the candidate evidence before it reaches the closed typed term row T.

If the candidate evidence is absent, the remaining visible rows are obstruction, route, transport, replay, provenance, and local naming. They preserve the MetaCIC boundary ledger and the sibling audit sources, but they contain no coordinate from which an evidence-free closed-term strong-normalization theorem can be read. □

Theorem 5163.6 (Candidate-mediated obstruction frontier)**.** *Every public strong-normalization or no-infinite-reduction read exported by an accepted* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *carrier*

$$M = (T, E, C, S, A, NF, O, R, H, Q, P, N)$$

must pass through the candidate-evidence row E and candidate-closure row C before it reaches S or NF. The route must also retain the obstruction row O, naming subject-reduction, substitution, confluence, and closed-consistency boundaries, and it must preserve $\mathsf{MetaCICCriticalPath}^{\uparrow}$,

$\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, *and* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *as route boundaries rather than replacement theorems.*

Proof. Project the carrier of Definition 5163.1. The endpoint row T is not an evidence-free normalization theorem; the positive route begins with E and C and only then reaches the strong-normalization and no-infinite-reduction rows S and NF.

Apply Definition 5163.2. Its obstruction-visibility and route-anchoring clauses keep O and R inside the local certificate, so subject reduction, substitution, confluence, and closed consistency remain named boundaries.

Now use Definition 5163.3. The handoff prevents Candidate, ClosedCandidate, or CandidateSet evidence from being erased into a global closed-term claim.

The rows H, Q, P, N transport, replay, source, and name the same route. Thus $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$, and $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ are preserved as sibling boundaries, while S and NF remain candidate-mediated exports downstream of E and C. □

Theorem 5163.7 (Candidate-mediated strong-normalization TasteGate obligation)**.** *Future Lean support for* $\mathsf{CandidateMediatedSN}^{\uparrow}$ *requires TasteGate* Nontrivial *and* FieldFaithful *instances. Nontriviality must distinguish an accepted packet with displayed candidate evidence from a closed typed term row without that evidence, and field faithfulness must preserve the row order*

$$T \longmapsto E \longmapsto C \longmapsto S \longmapsto A \longmapsto NF \longmapsto O \longmapsto R \longmapsto H, Q, P, N$$

before any carrier-alignment theorem can serve as a checked realization.

Proof. Nontriviality follows from the carrier boundary. Removing E or C leaves the closed typed term endpoint without the evidence route that supplies strong normalization.

Field faithfulness is the row-order condition. If a TasteGate projection permutes or erases $E, C, S, A, NF, O, R, H, Q, P, N$, the packet no longer represents the candidate-mediated route named by Definition 5163.1. □

Falsifiable prediction: Every accepted $\mathsf{CandidateMediatedSN}^{\uparrow}$ strong-normalization read exposes candidate evidence before the endpoint: closed typed term row first, candidate evidence second, candidate closure third, strong-normalization export fourth, Pi/application adequacy fifth, no-infinite-reduction projection sixth, and obstruction, route, transport, replay, provenance, and local naming rows thereafter.

Independence witness: metacic-normalization-audit-map, metacic-critical-path, metacic-normalization-frontier, metacic-confluence-audit-packet, normal-form-consistency-seal: $\mathsf{CandidateMediatedSN}^{\uparrow}$ is not bijective to any listed source because it isolates the evidence-carrying strong-normalization route while preserving each sibling as a named boundary rather than a replacement theorem.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5163.1, Definition 5163.2, Definition 5163.3, and Definition 5163.7 seed the candidate-mediated strong-normalization evidence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateMediatedSN}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization for every closed typed MetaCIC term, subject reduction, arbitrary substitution preservation, global confluence, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-evidence visibility, candidate-closure stability, strong-normalization export, Pi/application adequacy routing, no-infinite-reduction projection, obstruction visibility, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.CandidateMediatedSNUp.

Constructive story: A CandidateMediatedSN$^{\uparrow}$ packet is generated from a closed typed term row, candidate evidence, candidate closure, strong-normalization export, Pi/application adequacy rows, no-infinite-reduction projection, obstruction boundaries, sibling route anchors, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5164

A Concrete Naming Certificate for $\mathsf{CalculusOfConstructions}^{\uparrow}$

Following Coquand and Huet, the $\mathsf{CalculusOfConstructions}^{\uparrow}$ packet records the dependent-type calculus surface needed by the MetaCIC path as finite NameCert data. It extends the untyped syntax and substitution corridor of chapter 200, keeps computability comparisons separate from the machine models of chapter 199 and chapter 280, and routes downstream dependent typing through the displayed MetaCIC typing, substitution, and subject-reduction surfaces of chapter 1046, chapter 538, chapter 719, and chapter 730. The packet names a traditional calculus object without asserting consistency, normalization, quotient term equality, host function extensionality, or a completed MetaCIC development.

Definition 5164.1 (CalculusOfConstructions carrier)**.** *A* $\mathsf{CalculusOfConstructions}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, \Pi, L, A, \Gamma, \Sigma, B, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the finite sort row, naming the primitive universe and proposition classifiers used by the calculus surface;*
2. *Π is the dependent-product row, naming the domain, codomain-in-context, and formation boundary for Π-types;*
3. *L is the lambda row, naming abstraction formation over a displayed context extension;*
4. *A is the application row, naming elimination of a displayed dependent-product term by a displayed argument;*
5. *Γ is the context row, carrying finite extension, lookup, weakening, and exchange ledgers without identifying contexts by a quotient;*
6. *Σ is the substitution row, carrying finite capture-avoiding substitution and context-substitution ledgers;*
7. *B is the beta-boundary row, recording the local beta computation interface between L, A, and Σ;*
8. *R is the subject-reduction boundary row, exposing only the parameterised MetaCIC discharge route through the existing subject-reduction socket;*

9. *H records componentwise* hsame *transport for $S, \Pi, L, A, \Gamma, \Sigma, B, R, P, N$;*
10. *K records displayed* Cont *replay from the calculus packet to the lambda-calculus, typing-classifier, substitution-boundary, and subject-reduction-frontier consumers;*
11. *P is the* Pkg *provenance over* $\mathsf{CalculusOfConstructions}^{\uparrow}$, $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{TypingJudgmentClassifier}^{\uparrow}$, $\mathsf{ClosedSubstitutionBoundary}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, $\mathsf{MetaCICSubjectReductionFrontier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CalculusOfConstructions}^{\uparrow}}$ *row.*

The carrier contains no proof of global consistency, no strong-normalization theorem, no quotient equality of terms, no host-level function extensionality principle, and no completed MetaCIC kernel theorem.

Lean4 (definition) `BEDC.Derived.CalculusOfConstructionsUp.CalculusOfConstructionsUp`
Lean4 ✓ `BEDC.Derived.CalculusOfConstructionsUp.CalculusOfConstructionsTasteGate single carrier alignment`

Theorem 5164.2 (CalculusOfConstructions NameCert obligations)**.** *For the carrier of Definition 5164.1, the local* $\mathsf{NameCert}_{\mathsf{CalculusOfConstructions}^{\uparrow}}$ *surface consists exactly of finite obligations for the sort row S, dependent-product row Π, lambda row L, application row A, context row Γ, substitution row Σ, beta-boundary row B, subject-reduction boundary row R, componentwise transport H, displayed replay K, provenance P, and local naming N. Any accepted public read of the packet must pass through these rows and cannot replace them by a normalization theorem, a consistency theorem, a quotient conversion oracle, or host function extensionality.*

Lean4 ✓ `BEDC.Derived.CalculusOfConstructionsUp.CalculusOfConstructionsCarrier namecert obligations`

Proof. Project the carrier C. Its primitive calculus data are exactly $S, \Pi, L, A, \Gamma, \Sigma$, so the syntax, context, and substitution obligations are finite rows rather than ambient metatheoretic assumptions.

The boundary rows B and R add only local beta computation and the parameterised subject-reduction route. They do not add a closed subject-reduction theorem, because the route is read through the existing discharge socket and frontier.

Finally inspect H, K, P, N. Transport, replay, provenance, and local naming preserve the displayed row inventory. Since no other coordinate is present, a consumer has no place from which to extract consistency, normalization, quotient term equality, or host extensionality. □

Theorem 5164.3 (CalculusOfConstructions substitution typing route)**.** *Every accepted* $\mathsf{CalculusOfConstructions}^{\uparrow}$ *consumer that asks for a substitution-stable typing route must read the carrier in the order*

$$\Gamma, \Sigma \longrightarrow S, \Pi, L, A \longrightarrow B, R \longrightarrow H, K, P, N.$$

The context and substitution rows supply the finite substitution ledger, the sort and term-former rows supply the dependent typing surface, the beta and subject-reduction rows expose only the existing MetaCIC boundary, and the transport, replay, provenance, and naming rows keep the route inside the same finite packet. The route does not assert completed MetaCIC substitution admissibility, closed subject reduction, conversion soundness, normalization, or consistency.

Proof. Start with Γ, Σ. These rows display the finite context and substitution ledgers, so a substitution-stable read begins with explicit context extension, lookup, weakening, exchange, and capture-avoiding substitution data.

Next read S, Π, L, A. The sort, dependent-product, lambda, and application rows provide the dependent typing surface on which the substitution ledger acts. This is the calculus surface inherited from the Coquand–Huet presentation and routed through the $\mathsf{LambdaCalc}^{\uparrow}$ syntax corridor.

The rows B, R then expose the boundary to MetaCIC: beta computation is local to the displayed substitution row, and subject reduction is only the parameterised discharge route named by the socket and frontier chapters. Replaying the route through H, K, P, N preserves the same finite rows, so no stronger MetaCIC theorem is exported by the calculus packet itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{CalculusOfConstructions}^{\uparrow}}$.

Constructive story: A $\mathsf{CalculusOfConstructions}^{\uparrow}$ packet is generated from finite sort, dependent-product, lambda, application, context, substitution, beta-boundary, subject-reduction-boundary, componentwise transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CalculusOfConstructions}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5164.1, Definition 5164.2, and Definition 5164.3 bind the finite dependent-type calculus surface used by the MetaCIC path.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CalculusOfConstructions}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not prove consistency, normalization, quotient term equality, host function extensionality, completed MetaCIC substitution admissibility, closed subject reduction, Lean verification, or a public schema bridge.

Upgrade path: Obligation closure requires checked carrier admission, sort and dependent-product exactness, context and substitution route exactness, beta-boundary preservation, subject-reduction boundary routing, provenance, local naming, and non-escape for the same finite packet.

Chapter 5165

A Concrete Naming Certificate for $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$

Following Chebyshev and de la Vallee Poussin style prime-counting estimates, the PrimeNumberTheoremChebyshevW packet names the finite window in which a prime-counting read, logarithmic comparison ledger, and displayed tolerance record can be inspected without importing the full analytic proof of the prime number theorem. It sits after the natural-number source rows of the arithmetic chapters, the rational and dyadic tolerance carriers of chapter 37 and chapter 291, and the terminal real-facing seal of chapter 38. The packet records only finite counting windows, displayed prime predicates, logarithmic comparison rows, tolerance reads, transport, replay, provenance, and local naming.

Definition 5165.1 (Prime number theorem Chebyshev window carrier)**.** *A* PrimeNumberTheoremChebyshevWindow *carrier is a finite* BHist *packet*

$$W = (A, P, \Pi, L, D, R, O, H, C, Q, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{Nat}^{\uparrow}$ *arithmetic source row. It records the inspected finite interval, endpoint order, successor arithmetic, and bounded summation surface for the counting request.*
2. *P is the* $\mathsf{Prime}^{\uparrow}$ *predicate row. It stores the prime-admission read for each inspected natural-number entry and the finite exclusion rows for composites visible inside the window.*
3. *Π is the prime-counting window row. It records the displayed value of the finite counting function on the inspected interval and the endpoint comparison used by downstream reads.*
4. *L is the logarithmic comparison ledger. It stores the finite logarithm table, monotonicity comparisons, and Chebyshev-style upper and lower comparison rows attached to the same endpoint.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records dyadic slack, denominator bounds, and interval comparisons for the displayed counting and logarithmic reads.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *read row. It receives only the rational and dyadic tolerance data already displayed by D and L.*
7. *O is the order-provenance row. It records the* $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *and order comparisons used by A, P, Π, L, D, R.*
8. *H records componentwise* hsame *transport for arithmetic sources, prime predicates, counting windows, logarithmic ledgers, dyadic and real reads, order provenance, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a prime-counting request through* $A, P, \Pi, L, D, R, O, H, Q, N$.
10. *Q is the* Pkg *provenance over* $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *order rows,* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, P, \Pi, L, D, R, O, H, C, Q, N$. It has no coordinate for analytic continuation, the full asymptotic prime number theorem, zero-free regions, choice, a host prime-distribution oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PrimeNumberTheoremChebyshevWindowUp.PrimeNumberTheoremChebyshevWindowTasteGate`
`single carrier alignment`

Theorem 5165.2 (Prime number theorem Chebyshev window obligations)**.** *For every accepted* $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$ *carrier*

$$W = (A, P, \Pi, L, D, R, O, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the natural-number arithmetic source, prime predicate row, prime-counting window, logarithmic comparison ledger, dyadic tolerance read, real-facing read, order provenance, transport, replay, package provenance, and local naming;*
2. *prime-window exposure, requiring A, P, Π to display the inspected endpoint interval, prime-admission reads, composite exclusions, and finite counting value before any comparison can be exported;*
3. *logarithmic comparison control, requiring L to attach Chebyshev-style upper and lower comparison rows to the same endpoint and counting window;*
4. *dyadic and real nonescape, requiring D and R to read only displayed rational, dyadic, and order data from L and Π;*
5. *local naming, requiring O, H, C, Q, N to preserve exactly the displayed arithmetic, prime, logarithmic, tolerance, real-facing, and provenance rows.*

No prime-counting obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5165.1. The accepted packet displays exactly the arithmetic source, prime predicate, counting window, logarithmic comparison ledger, tolerance rows, real-facing read, order provenance, and structural rows listed in the statement.

The prime-window exposure is finite. The rows A and P inspect only the bounded natural-number interval and its displayed prime or composite evidence, while Π records the corresponding finite counting value.

The logarithmic comparison control is read next. The row L attaches the Chebyshev-style upper and lower comparison rows to the same endpoint and the same counting window, so no independent analytic-continuation or zero-free-region theorem enters the packet.

The dyadic and real-facing rows remain downstream. The row D supplies the rational and dyadic slack ledger, R reads only that displayed tolerance surface as real-facing data, and O keeps the natural-number, prime, rational, and order provenance visible.

Finally H, C, Q, N transport, replay, source, and name the same packet. Since the carrier contains no coordinate for analytic continuation, the full asymptotic prime number theorem, zero-free regions, choice, a host prime-distribution oracle, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5165.3 (Prime number theorem Chebyshev window route)**.** *Every real-facing comparison read exported by an accepted* $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$ *carrier*

$$W = (A, P, \Pi, L, D, R, O, H, C, Q, N)$$

factors through the finite route

$$A, P, \Pi, L, D, O \longmapsto R.$$

Thus the exported read is witnessed by a natural-number source row, a prime predicate row, a finite prime-counting window, a logarithmic comparison ledger, dyadic and rational tolerance data, and order provenance, not by analytic continuation, the full asymptotic theorem, zero-free regions, choice, or ambient Real completeness.

Proof. Start with a real-facing comparison read from the accepted carrier. The only row with that target is R, and the carrier places R downstream of the arithmetic, prime, counting, logarithmic, dyadic, and order rows.

The source and counting rows determine the finite prime-counting data. The row A fixes the inspected natural-number interval, P supplies prime-admission and composite-exclusion reads, and Π records the displayed finite counting value for that same window.

The comparison rows then control the exported estimate. The row L records the logarithmic comparison ledger, D supplies the dyadic and rational slack, and O records the order provenance that binds these comparisons to the same endpoint.

Only after those rows are fixed can R read the packet as real-facing comparison data. Transport, replay, package provenance, and local naming preserve this route through H, C, Q, N, so none of the excluded ambient principles can enter the exported read. □

Falsifiable prediction: Every $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$ real-facing comparison read passes through $\mathsf{Nat}^{\uparrow}$, $\mathsf{Prime}^{\uparrow}$, a finite prime-counting window, a logarithmic comparison ledger, $\mathsf{Rat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, order provenance, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any counting estimate is visible.

Independence witness: nat, prime, rat, dyadicratcore, real, order: $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a natural-number source row, prime predicate row, finite counting window, logarithmic comparison ledger, dyadic and rational tolerance reads, real-facing read, order provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}$ packet is generated from NatUp arithmetic source rows, PrimeUp predicate rows, finite prime-counting windows, logarithmic comparison ledgers, RatUp and DyadicRatCoreUp tolerance reads, RealUp read rows, order provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5165.1, Definition 5165.2, and Definition 5165.3 seed the finite Chebyshev-window route from natural-number and prime rows through counting windows, logarithmic comparisons, dyadic and rational tolerances, real-facing reads, order provenance, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimeNumberTheoremChebyshevWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close analytic continuation, the full asymptotic prime number theorem, zero-free regions, choice, host prime-distribution oracles, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked arithmetic source admission, prime predicate exposure, finite counting-window exactness, logarithmic comparison control, dyadic and real nonescape, order provenance, transport, replay, provenance, and local naming for BEDC.Derived.PrimeNumberTheoremChebyshevWindowUp.

Chapter 5166

A Concrete Naming Certificate for $\mathsf{PriestleyDuality}^{\uparrow}$

Following Priestley and Davey–Priestley, the $\mathsf{PriestleyDuality}^{\uparrow}$ packet records the finite BEDC naming surface for the ordered topological representation of bounded distributive lattices. It sits between the lattice rows of chapter 55, the complemented endpoint discipline of chapter 3733, the indexed-open topology rows of chapter 88, and the clopen-ledger bridge of chapter 261. The packet names order-separating clopen upsets and their finite continuity ledger; it does not import ultrafilters, selected prime filters, powerset topology, quotient lattice equality, or a host duality theorem.

Definition 5166.1 (Priestley duality carrier)**.** *A* $\mathsf{PriestleyDuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (L, B, T, S, U, O, H, C, G, N)$$

with the following displayed rows:

1. *L is the bounded distributive-lattice source read through the* $\mathsf{Lattice}^{\uparrow}$ *carrier and its finite meet–join order rows.*
2. *B is the Boolean endpoint comparison row available from the* $\mathsf{BooleanAlgebra}^{\uparrow}$ *neighbour when complemented clopen data is requested.*
3. *T is the indexed-open* $\mathsf{Topology}^{\uparrow}$ *row that records the finite open-neighbourhood observations used by the dual side.*
4. *S is the Stone-facing clopen ledger from chapter 261, restricted to observations that are also upward-closed for the order read from L.*
5. *U is the finite order-separation row: whenever two displayed lattice observations are separated by the order classifier, U names a clopen upset containing the larger observation and excluding the smaller one.*
6. *O is the order/topology compatibility row, recording that finite intersections, unions, complements used through B, and indexed opens used through T preserve the clopen-upset route named by S and U.*
7. *H records componentwise* hsame *transport for $L, B, T, S, U, O, C, G, N$.*
8. *C records displayed* Cont *replay from a lattice observation through the clopen-upset and topology rows.*

9. *G is the* Pkg *provenance over* Lattice$^{\uparrow}$, BooleanAlgebra$^{\uparrow}$, Topology$^{\uparrow}$, StoneDuality$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{PriestleyDuality}^{\uparrow}}$ *row.*

The carrier contains no coordinate for ultrafilter selection, a host spectral space, unrestricted prime filters, quotient lattice equality, powerset topology, or classical choice.

Lean4 ✓ `BEDC.Derived.PriestleyDualityUp.PriestleyDualityTasteGate single carrier alignment`

Theorem 5166.2 (Priestley duality NameCert obligations)**.** *For every accepted* PriestleyDuality$^{\uparrow}$ *carrier $P = (L, B, T, S, U, O, H, C, G, N)$, the local* NameCert$_{\text{PriestleyDuality}^{\uparrow}}$ *surface consists exactly of lattice-source admission through L, Boolean endpoint comparison through B, topology admission through T, Stone-facing clopen admission through S, finite order separation through U, order/topology compatibility through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through G, and local naming through N. Any downstream order/topology consumer must consume these rows before it can treat a clopen upset as a Priestley-duality observation.*

Proof. Project the carrier of Definition 5166.1. The rows L, B, T, S, U, O are the only analytic and order-topology coordinates, and the rows H, C, G, N only transport, replay, source, and name them.

The lattice-source obligation is read from L, while Boolean endpoint comparison and topology admission are read from B and T. The Stone-facing clopen ledger S supplies the clopen side, and U restricts it to the finite upsets needed for order separation.

Compatibility is then checked by O. This row keeps the finite clopen-upset observations stable under the lattice and topology operations that are public through the cited neighbouring chapters.

Finally H, C, G, N preserve and name the same route. Because the carrier has no ultrafilter, selected prime-filter, host spectral-space, powerset-topology, quotient-equality, or choice coordinate, the listed rows exhaust the local NameCert obligations. □

Closure status for NameCert$_{\text{PriestleyDuality}^{\uparrow}}$.

Constructive story: A PriestleyDuality$^{\uparrow}$ packet is generated from bounded distributive-lattice rows, Boolean endpoint comparisons, indexed topology observations, a Stone-facing clopen ledger, finite clopen-upset separation rows, order/topology compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PriestleyDuality}^{\uparrow}}$, seed)

Scope closed: Definition 5166.1 and Definition 5166.2 seed the order/topology representation surface over lattice, Boolean-algebra, topology, and Stone-duality rows.

Formal verification: FormalStatus(NameCert$_{\text{PriestleyDuality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an ultrafilter construction, selected prime filters, host spectral spaces, powerset topology, quotient lattice equality, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite order-separation stability, clopen-upset topology compatibility, and local NameCert exhaustion for BEDC.Derived.PriestleyDualityUp.

The VitaliConvergence$^{\uparrow}$ hub routes uniform-integrability budgets through chapter 6228 and convergence-in-measure tolerances through chapter 5139 before the terminal Real$^{\uparrow}$ seal.

Chapter 5167

A Concrete Naming Certificate for $\mathsf{VitaliConvergence}^{\uparrow}$

Following Vitali and Bishop–Bridges constructive integration, the $\mathsf{VitaliConvergence}^{\uparrow}$ packet records the finite surface by which uniform integrability, convergence-in-measure windows, and integral readback rows are retained before a convergence consumer reads the Vitali conclusion. It sits after the measure surface of chapter 92, the integral row of chapter 123, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the dominated-convergence packet of chapter 2340. The Vitali handoff consumes the finite tail-control surface of chapter 6228 and the measure and integral rows displayed by dominated-convergence siblings before any quotient almost-everywhere equality or selected subsequence is read. The packet names only displayed convergence and uniform-integrability rows; it does not assert measure completion, almost-everywhere quotient equality, selected subsequences, countable choice, or a host theorem about all measurable functions.

Definition 5167.1 (Vitali convergence carrier)**.** *A* $\mathsf{VitaliConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (M, I, F, U, L, R, E, D, H, C, P, N)$$

with the following rows:

1. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row for the displayed measurable-event requests.*
2. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row for the integrals that the packet may compare.*
3. *F is the finite family row for the displayed integrands.*
4. *U is the uniform-integrability row, recording only finite tail and small-set budget requests already routed through M and I.*
5. *L is the convergence-in-measure window row.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerances on integral and measure budgets.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed limit budget.*
8. *D is the handoff row to* $\mathsf{DominatedConvergence}^{\uparrow}$*, used only when a displayed domination row supplies a sibling majorant route.*
9. *H records componentwise* hsame *transport for M, I, F, U, L, R, E, D.*
10. *C records displayed* Cont *replay of the same route.*

11. *P is the* Pkg *provenance over* $\mathsf{VitaliConvergence}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, DominatedConvergence BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VitaliConvergence}^{\uparrow}}$ *row.*

The carrier has no coordinate for measure completion, quotient equality of functions, almost-everywhere representative selection, countable choice, or a detached ambient convergence theorem.

Theorem 5167.2 (Vitali convergence NameCert obligations)**.** *For every accepted* $\mathsf{VitaliConvergence}^{\uparrow}$ *carrier $V = (M, I, F, U, L, R, E, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VitaliConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for V, measure-source admission through M, integral readback through I, displayed integrand-family exposure through F, uniform-integrability budget exposure through U, convergence-in-measure window exposure through L, regular rational tolerance readback through R, terminal real seal through E, dominated-convergence handoff through D, componentwise* hsame *stability through H,* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5167.1. The available analytic rows are exactly M, I, F, U, L, R, E, D, and the structural rows are H, C, P, N.

Uniform integrability is exposed only by U. Its finite tail and small-set budgets can be read after the measure and integral rows M, I have already been displayed, so it is not an ambient boundedness principle or a hidden domination theorem.

The convergence side is exposed by L, R, E. The convergence-in-measure windows pass through rational tolerance readback before the terminal real seal is consumed.

Finally, D, H, C, P, N give the optional dominated-convergence handoff, transport, replay, provenance, and local naming of the same rows. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5167.3 (Vitali convergence uniform-integrability handoff)**.** *Every integral-facing or real-facing consumer of an accepted* $\mathsf{VitaliConvergence}^{\uparrow}$ *carrier $V = (M, I, F, U, L, R, E, D, H, C, P, N)$ must read the uniform-integrability row U together with the convergence-in-measure row L before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. If the consumer also uses a dominated-convergence sibling route, that route is exposed only through D after the same M, I, F, U, L, R, E rows have been displayed.*

Proof. Begin with a consumer of V. By Definition 5167.1, the only measure and integral entry points are M and I, and the only integrand-family row is F.

The uniform-integrability budget is then forced through U. A consumer may inspect finite tail and small-set budgets, but only as rows attached to the displayed measure and integral surfaces.

The convergence read is $L \longmapsto R \longmapsto E$. The convergence-in-measure windows and rational tolerances are visible before the real seal is reached.

The dominated-convergence handoff is separate. The row D can be consumed only after the same visible rows have been retained, so it cannot replace Vitali uniform integrability by a hidden host majorant or an almost-everywhere quotient. □

Falsifiable prediction: A consumer that claims a $\mathsf{VitaliConvergence}^{\uparrow}$ integral-limit read while omitting the measure source, integral row, integrand family, uniform-integrability budget, convergence-in-measure window, rational tolerance readback, real seal, dominated-convergence handoff, hsame, Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5167.1.

Independence witness: measure, integral, regseqrat, real, dominated-convergence: $\mathsf{VitaliConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries uniform-integrability budgets, convergence-in-measure windows, rational tolerance readback, real sealing, dominated-convergence handoff, transport, replay, provenance, and local naming over a displayed integrand family.

Closure status for $\mathsf{NameCert}_{\mathsf{VitaliConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{VitaliConvergence}^{\uparrow}$ packet is generated from a displayed Measure row, Integral readback row, integrand-family row, uniform-integrability budget row, convergence-in-measure window row, RegSeqRat tolerance row, Real seal row, DominatedConvergence handoff row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VitaliConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5167.1, Definition 5167.2, and Definition 5167.3 seed the finite Vitali convergence route through Measure, Integral, RegSeqRat, Real, and dominated-convergence rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VitaliConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, almost-everywhere quotient equality, selected subsequences, countable choice, Lean verification, bridge checking, or a host theorem about all measurable functions.

Upgrade path: Obligation closure requires checked carrier admission, uniform-integrability visibility, convergence-in-measure visibility, rational tolerance readback, Real seal handoff, dominated-convergence sibling routing, transport and replay stability, and nonescape for BEDC.Derived.VitaliConvergenceUp.

Theorem 5167.4 (Vitali convergence Cauchy sequential route)**.** *Any* $\mathsf{VitaliConvergence}^{\uparrow}$ *consumer that reaches the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance row* R *of an accepted carrier*

$$V = (M, I, F, U, L, R, E, D, H, C, P, N)$$

must route the convergence-in-measure window L *through chapter 5139 before consuming the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*. The route exposes the measure source* M*, integral row* I*, integrand family* F*, uniform-integrability budget* U*, convergence window* L*, rational tolerance row* R*, structural rows* H, C, P, N*, and only then the terminal real seal.*

Proof. Project the carrier of Definition 5167.1. The convergence side is the displayed chain L, R, E, while M, I, F, U supply the measure, integral, integrand-family, and uniform-integrability rows that make the window meaningful.

Before E can be read as a real-facing endpoint, the tolerance row R must be interpreted as a sequential Cauchy tolerance surface. The required neighboring packet is chapter 5139, whose carrier records the finite stream-window, regular rational readback, dyadic tolerance, and Real-seal handoff used by Real and Completion consumers.

The structural rows H, C, P, N preserve this route componentwise. Thus a Vitali consumer may reach the terminal $\mathsf{Real}^{\uparrow}$ seal only after the convergence-in-measure window has passed through the Cauchy sequential criterion surface; no hidden selected subsequence, countable-choice row, quotient equality, or ambient convergence theorem is introduced. □

Proposition 5167.5 (Vitali measure-Cauchy dependency route)**.** *Every* $\mathsf{VitaliConvergence}^{\uparrow}$ *read that exposes a convergence seal first consumes the uniform-integrability route of chapter 6228 and the Cauchy sequential route of chapter 5139. The read therefore keeps the measure and integral rows from chapter 5167, the budget row of Definition 5167.6, and the tolerance readback of Definition 5167.4 visible before any terminal* $\mathsf{Real}^{\uparrow}$ *or dominated-convergence handoff is exposed.*

Proof. Project the carrier of Definition 5167.1. The rows $M, I, F, U, L, R, E, D, H, C, P, N$ are the only available inputs to a convergence-facing read.

Use Definition 5167.6 to keep U attached to the displayed measure and integral surface before any terminal seal is reached. Use Definition 5167.4 to route L and R through the Cauchy sequential criterion before E is consumed.

The two routes share the same carrier rows, so their combination is still a finite Vitali dossier edge. It exports no measure completion, almost-everywhere quotient equality, selected subsequence, countable-choice schedule, or detached host convergence theorem. □

Theorem 5167.6 (Vitali convergence uniform-integrability dependency route)**.** *Every accepted* $\mathsf{VitaliConvergence}^{\uparrow}$ *consumer that reads the uniform-integrability row* U *must route it through chapter 6228, the measurable event surface of chapter 92, the* L^p *function-space boundary of chapter 8060, and the dominated majorant handoff of chapter 2340 before consuming the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route keeps the displayed rows*

$$M, I, F, U, L, R, E, D, H, C, P, N$$

visible, and exports no measure completion, almost-everywhere quotient equality, selected subsequence, countable choice, or detached host convergence theorem.

Proof. Project the carrier of Definition 5167.1. The uniform-integrability row U is available only together with the displayed measure row M, integral row I, integrand-family row F, convergence-in-measure window L, tolerance row R, real seal E, dominated-convergence handoff D, and structural rows H, C, P, N.

Read chapter 6228. It supplies the tail-cutoff and uniform tail-bound surface that the Vitali packet consumes, while chapter 92 keeps the event requests attached to the same measurable carrier.

The L^p boundary in chapter 8060 records the norm-facing function-space read without adding representatives or quotient equality. The dominated-convergence sibling of chapter 2340 is then available only through the displayed handoff row D.

Thus the dependency route is carrier-local: it preserves the Vitali rows and their sibling citations, but it does not turn uniform integrability into a host measure-completion theorem, a selected-subsequence principle, or an ambient convergence theorem. □

Chapter 5168

A Concrete Naming Certificate for $\mathsf{BishopDyadicSqueeze}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopDyadicSqueeze}^{\uparrow}$ packet records the finite sandwich certificate by which two dyadic control windows trap a regular rational observation before the observation is handed to a real-name seal. It sits over the dyadic denominator discipline of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the countable-window presentation of chapter 1809. The packet gives the L10 Real and Completion route a concrete squeeze object over inspected dyadic windows; it is not a quotient-real equality theorem, a selected limit, trichotomy, countable choice, or ambient completeness principle.

Definition 5168.1 (Bishop dyadic squeeze carrier)**.** *A* $\mathsf{BishopDyadicSqueeze}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (\ell, u, n, W, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. ℓ *and* u *are lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows.*
2. n *is the unary precision row naming the dyadic scale at which the squeeze is inspected.*
3. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the source regular rational observations are read.*
4. L *records the lower-bound comparison between the inspected window and* ℓ*.*
5. U *records the upper-bound comparison between the inspected window and* u*.*
6. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from* W, L, U *and the displayed dyadic tolerance scale.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from* R *after the squeeze window has been displayed.*
8. H *records componentwise* hsame *transport for* $\ell, u, n, W, L, U, R, E$ *and the local naming row.*
9. C *records displayed* Cont *replay from a squeeze request through dyadic endpoints, finite stream windows, lower and upper comparisons, regular rational readback, and the Real seal.*
10. P *is the* Pkg *provenance over* $\mathsf{BishopDyadicSqueeze}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDyadicSqueeze}^{\uparrow}}$ *row.*
The classifier compares accepted squeeze packets only through $\ell, u, n, W, L, U, R, E, H, C, P, N$. It has no coordinate for a quotient real representative, selected limit, trichotomy oracle, countable-choice schedule, hidden infinite stream, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 (definition) `BEDC.Derived.BishopDyadicSqueezeUp.BishopDyadicSqueezeUp`

Theorem 5168.2 (Bishop dyadic squeeze NameCert obligations)**.** *For every accepted* $\mathsf{BishopDyadicSqueeze}^{\uparrow}$ *carrier*

$$S = (\ell, u, n, W, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopDyadicSqueeze}^{\uparrow}}$ *surface consists exactly of dyadic endpoint exposure, unary precision exposure, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, lower-bound comparison exposure, upper-bound comparison exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must expose $\ell, u, n, W, L, U, R, E$ before the structural rows H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5168.1. The analytic rows are exactly $\ell, u, n, W, L, U, R, E$, while H, C, P, N are the structural transport, replay, provenance, and naming rows.

The endpoint rows ℓ, u and the precision row n determine the displayed dyadic scale. The stream row W is therefore the only observation surface through which the source regular rational data can enter the squeeze.

The comparison rows L and U bind that same window below and above. Hence the packet does not export a detached lower route or a detached upper route; both comparisons are attached to the same finite window and the same dyadic scale.

Finally R and E are downstream of the squeeze rows. Since the carrier has no coordinate for quotient representatives, selected limits, trichotomy, countable choice, hidden streams, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5168.3 (Bishop dyadic squeeze Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{BishopDyadicSqueeze}^{\uparrow}$ *packet reaches the terminal Real seal only through the displayed finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may use the dyadic endpoint rows, unary precision row, finite stream window, lower and upper comparison rows, and regular rational readback, but it cannot replace them by quotient real equality, a selected limit, trichotomy, countable choice, an uninspected infinite stream, or ambient completeness.

Proof. Begin with the Real or Completion-facing read of S. The dyadic endpoints ℓ, u and precision row n are exposed before the stream window W, so the consumer reads a finite dyadic scale before any rational replay is available.

The lower and upper comparison rows L, U are both attached to W. Thus the regular rational readback R is assembled from a displayed squeeze window rather than from a selected representative or an external limit operation.

The terminal seal E is downstream of R. It therefore receives only the regular rational route produced by the finite dyadic squeeze data, and the excluded host principles remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDyadicSqueeze}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDyadicSqueeze}^{\uparrow}$ packet is generated from lower and upper DyadicRatCore endpoints, a unary precision row, StreamName windows, lower and upper comparison rows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDyadicSqueeze}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5168.1, Definition 5168.2, and Definition 5168.3 seed the finite dyadic squeeze route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDyadicSqueeze}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, countable choice, uninspected infinite streams, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic endpoint exposure, precision handling, finite-window exposure, lower and upper comparison exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopDyadicSqueezeUp.

Chapter 5169

A Concrete Naming Certificate for $\mathsf{GalerkinProjection}^{\uparrow}$

Following Cea and Galerkin finite-dimensional projection theory, the $\mathsf{GalerkinProjection}^{\uparrow}$ packet records the finite trial-space route by which weak Sobolev data are tested against a displayed finite basis and returned as a residual certificate. It sits between the weak-data surface of chapter 440, the inner-product and norm discipline of chapter 91 and chapter 1950, the linear operator rows of chapter 48, and the finite coordinate ledger of chapter 49. The packet supplies a constructive projection certificate without importing host Hilbert projection machinery, quotient function spaces, selected representatives, global minimizers, or choice.

Definition 5169.1 (Galerkin projection carrier)**.** *A* $\mathsf{GalerkinProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, H, L, M, V, B, A, b, u, r, O, T, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *weak-data source row carrying the finite testable function data requested by the problem.*
2. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *or* $\mathsf{NormedSpace}^{\uparrow}$ *ambient norm row used only through displayed inner-product and norm comparisons.*
3. *L is the* $\mathsf{LinMap}^{\uparrow}$ *bilinear or operator row that supplies the weak equation being tested.*
4. *M is the* $\mathsf{Mat}^{\uparrow}$ *coordinate row for the finite stiffness and load tables.*
5. *V is the finite trial-space row, listing the accepted basis names and the admissible finite coefficient windows.*
6. *B is the basis-evaluation row, displaying how each trial basis element is read back into the weak-data carrier.*
7. *A and b are the finite linear system rows induced by the operator row L and the trial row V.*
8. *u is the computed trial coefficient row inside the displayed finite coordinate packet.*
9. *r is the residual row, recording the finite test residual against every displayed trial basis element.*
10. *O is the residual-orthogonality row, exposing that the residual readback vanishes on the displayed trial-space tests.*

11. *T records componentwise* hsame *transport for weak source rows, norm rows, operator rows, matrix rows, trial rows, basis readback, finite system rows, coefficient rows, residual rows, orthogonality rows, provenance, and local naming.*

12. *C records displayed* Cont *replay of the finite route*

$$S, H, L, M, V, B, A, b, u, r, O \longmapsto T, P, N.$$

13. *P is the* Pkg *provenance over* $\mathsf{GalerkinProjection}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{GalerkinProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, H, L, M, V, B, A, b, u, r, O, T, C, P, N$. It has no coordinate for host minimizers, quotient function spaces, selected representatives, ambient Hilbert projection operators, global compactness, or choice.

Lean4 (definition) `BEDC.Derived.GalerkinProjectionUp.GalerkinProjectionUp`

Theorem 5169.2 (Galerkin projection NameCert obligations)**.** *Every accepted* $\mathsf{GalerkinProjection}^{\uparrow}$ *packet exposes its trial-space, finite-coordinate, and residual-certificate obligations through the displayed route*

$$\mathsf{BHist}(S, H, L, M, V, B, A, b, u, r, O) \longrightarrow \mathsf{hsame}(T) \longrightarrow \mathsf{Cont}(C) \longrightarrow \mathsf{Pkg}(P) \longrightarrow \mathsf{NameCert}_{\mathsf{GalerkinProjection}^{\uparrow}}(N).$$

The NameCert row may read the sibling packets $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, *and* $\mathsf{Mat}^{\uparrow}$ *only through their displayed local rows. It cannot replace the finite trial packet by a quotient function space, a selected equivalence-class representative, a host Hilbert projection theorem, a global minimizer, or a choice principle.*

Proof. Project the carrier of Definition 5169.1. The source, ambient norm, weak operator, matrix coordinate, trial-space, basis-readback, finite system, coefficient, residual, and orthogonality rows are all listed before transport and replay.

The transport row T is componentwise, so it preserves each displayed dependency separately. The replay row C then records the finite route from those rows to provenance and local naming; it does not add a selector for a host representative.

Finally the package row P cites only the sibling NameCert surfaces named in the statement, and the local row N closes the packet over that finite route. Since no carrier coordinate names a host minimizer, quotient function space, selected representative, ambient projection operator, global compactness witness, or choice schedule, the certificate obligations are exhausted by the displayed rows. □

Theorem 5169.3 (Galerkin finite-subspace residual orthogonality)**.** *For every accepted* $\mathsf{GalerkinProjection}^{\uparrow}$ *packet $G = (S, H, L, M, V, B, A, b, u, r, O, T, C, P, N)$, the residual row r is exposed to downstream consumers only together with the finite trial-space row V, the basis-readback row B, the matrix system rows A, b, and the residual-orthogonality row O. Thus every residual certificate is a finite-subspace statement: each displayed trial basis test reads zero residual through O, while no consumer obtains an ambient minimizer, quotient-function representative, or selected best approximation outside the packet.*

Proof. Read the finite trial-space row V and the basis-readback row B. These rows specify exactly which test functions are visible to the residual certificate.

The finite system rows A, b and coefficient row u determine the residual row r inside the same displayed coordinate packet. The orthogonality row O then records the zero residual checks against the basis tests carried by V and B.

Transport and replay preserve that finite list through T and C, while P and N seal the result as local NameCert data. Because the carrier has no coordinate for ambient minimizers, quotient function spaces, selected representatives, or choice, the exported orthogonality claim is exactly the finite-subspace residual certificate. □

Falsifiable prediction: Every accepted $\mathsf{GalerkinProjection}^{\uparrow}$ read displays weak source, ambient norm, operator, matrix, finite trial-space, basis-readback, finite system, coefficient, residual, orthogonality, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any Sobolev, Hilbert, linear-map, matrix, or normed-space consumer reads the projection.

Independence witness: sobolev, hilbert, linearmap, matrix, normedspace: $\mathsf{GalerkinProjection}^{\uparrow}$ is not bijective to any listed sibling because it binds finite trial-space admission, coordinate system rows, computed coefficients, residual readback, residual orthogonality, componentwise transport, displayed replay, provenance, and local naming into one accepted projection packet.

Closure status for $\mathsf{NameCert}_{\mathsf{GalerkinProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{GalerkinProjection}^{\uparrow}$ packet is generated from displayed Sobolev weak data, Hilbert or NormedSpace comparison rows, a LinearMap weak operator row, Matrix coordinate rows, a finite trial-space basis, finite system rows, computed coefficients, residual readback, residual orthogonality, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GalerkinProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5169.1, Definition 5169.2, and Definition 5169.3 seed the finite Galerkin projection route over Sobolev, Hilbert, LinearMap, Matrix, and NormedSpace rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GalerkinProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host Hilbert projection machinery, quotient function spaces, selected representatives, global minimizers, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked trial-space admission, finite matrix-system construction, residual readback, residual orthogonality, transport stability, replay, provenance, and local naming for BEDC.Derived.GalerkinProjectionUp.

Chapter 5170

A Concrete Naming Certificate for $\mathsf{HigsonCompactification}^{\uparrow}$

Following Higson and Roe on coarse topology, the $\mathsf{HigsonCompactification}^{\uparrow}$ packet records the compactification read by slowly oscillating bounded continuous functions on a large-scale metric or uniform source. It sits after the uniform-space carrier of chapter 339, the metric observation surface of chapter 57, the compact target boundary of chapter 60, the continuous-map row of chapter 130, and the terminal real-valued readback of chapter 38. It is adjacent to the uniform-envelope route of chapter 2154 and the bounded-function envelope of chapter 2344, but its local carrier keeps the coarse-control row visible rather than collapsing to either sibling.

Definition 5170.1 (Higson compactification carrier)**.** *A* $\mathsf{HigsonCompactification}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, M, A, C, F, R, S, B, H, T, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row supplying entourages and uniform refinement data for the underlying space;*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *large-scale control row recording bounded-set, coarse-separation, and distance-window requests;*
3. *A is the Higson-algebra row of bounded real-valued functions whose oscillation is eventually small outside controlled bounded regions;*
4. *C is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row proving that every accepted function in A is read through displayed continuity data;*
5. *F is the finite observation-family row used by a consumer to inspect only finitely many Higson functions at a requested stage;*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for the bounded real observations in A and F;*
7. *S is the Samuel-compactification comparison row imported from chapter 2154;*
8. *B is the Stone–Cech comparison row imported from chapter 2344;*
9. *H records componentwise* hsame *transport for $U, M, A, C, F, R, S, B, T, P, N$;*
10. *T records displayed* Cont *replay of coarse-control and function-evaluation requests;*

11. *P is the* Pkg *provenance over* $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the two compactification neighbours,* BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HigsonCompactification}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. There is no coordinate for a selected boundary point, ultrafilter choice, quotient host compactification, full Stone–Cech universality, or ambient properness theorem.

Lean4 ✓ `BEDC.Derived.HigsonCompactificationUp.HigsonCompactificationTasteGate single carrier alignment`

Theorem 5170.2 (Higson compactification NameCert obligations)**.** *For every accepted* $\mathsf{HigsonCompactification}^{\uparrow}$ *carrier*

$$K = (U, M, A, C, F, R, S, B, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HigsonCompactification}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform source, large-scale metric row, slowly oscillating bounded function row, continuity row, finite observation family, real value seals, Samuel comparison, Stone–Cech comparison, componentwise transport, replay, provenance, and local naming.*

The obligation surface also records coarse-control non-escape: every accepted real-valued observation must pass through M, A, C, F, R before it is read as compactification data, and the comparison rows S, B do not replace the Higson algebra row by Samuel uniformity alone or by Stone–Cech bounded functions alone.

Proof. Project the carrier K. The rows U and M expose the uniform and large-scale source data, while A, C, F, R expose the slowly oscillating bounded functions, their continuity witnesses, finite observation families, and real value seals.

The comparison rows are then read as neighbours rather than definitions. Row S identifies the Samuel compactification route available from uniform data, and row B identifies the Stone–Cech bounded-function envelope, but neither row carries the coarse-control predicate stored in A over the metric row M.

The structural rows H, T, P, N transport, replay, package, and name exactly the displayed route. Since the carrier has no further coordinate, a consumer cannot obtain a selected boundary point, an ultrafilter, a quotient host compactification, full Stone–Cech universality, or an ambient properness theorem from the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.HigsonCompactificationUp.HigsonCompactificationCarrier namecert obligations`

Theorem 5170.3 (Higson compactification uniform handoff)**.** *Every compactification-facing consumer of an accepted* $\mathsf{HigsonCompactification}^{\uparrow}$ *packet factors through the route*

$$U, M \longmapsto A, C, F, R \longmapsto S, B, H, T, P, N.$$

Thus the consumer first reads the uniform and large-scale metric source rows, then the slowly oscillating bounded continuous real-valued observation rows, and only then the Samuel and Stone–Cech comparison rows together with transport, replay, provenance, and local naming.

Proof. Start with an accepted carrier $K = (U, M, A, C, F, R, S, B, H, T, P, N)$. The only source rows are U and M, so every compactification-facing request begins with the displayed uniform and large-scale metric data.

The Higson observation rows are downstream of that source. Row A records the slowly oscillating bounded real-valued functions, C keeps their continuity evidence visible, F restricts consumer inspection to finite observation families, and R seals the observed values as $\mathsf{Real}^{\uparrow}$ data.

The neighbour rows S and B are consumed after the Higson observation route has been read. They compare the packet with Samuel and Stone–Cech compactification surfaces, while H, T, P, N retain transport, replay, provenance, and local naming for the same displayed rows. No bypass row exists for selected boundary points, ultrafilters, quotient host compactifications, full Stone–Cech universality, or ambient properness. □

Falsifiable prediction: Every $\mathsf{HigsonCompactification}^{\uparrow}$ compactification-facing read exposes a $\mathsf{UniformSpace}^{\uparrow}$ row, a $\mathsf{Met}^{\uparrow}$ large-scale row, a slowly oscillating bounded $\mathsf{ContinuousMap}^{\uparrow}$ row, finite observation data, $\mathsf{Real}^{\uparrow}$ value seals, Samuel and Stone–Cech comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: uniformspace, metric, compact, continuousmap, real, samuel-compactification, stone-cech-compactification: $\mathsf{HigsonCompactification}^{\uparrow}$ is not bijective to any listed sibling because it binds coarse-control and slowly oscillating bounded continuous real-valued observations together with the two compactification comparison rows in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HigsonCompactification}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HigsonCompactification}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5170.1, Definition 5170.2, and Definition 5170.3 seed the finite Higson compactification packet through uniform, metric, compact, continuous-map, Real, Samuel, and Stone–Cech rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HigsonCompactification}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected boundary points, ultrafilter choice, quotient host compactifications, full Stone–Cech universality, ambient properness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coarse-control exactness, slowly oscillating bounded function stability, finite observation replay, neighbour comparison non-collapse, provenance, and local naming.

Constructive story: A $\mathsf{HigsonCompactification}^{\uparrow}$ packet is generated from displayed uniform and large-scale metric source rows, slowly oscillating bounded continuous real-valued observations, finite observation families, Samuel and Stone–Cech comparison rows, componentwise transport, replay, provenance, and a local NameCert row.

Chapter 5171

A Concrete Naming Certificate for $\mathsf{HigsonCorona}^{\uparrow}$

Following Higson and Roe on coarse topology, the $\mathsf{HigsonCorona}^{\uparrow}$ packet records the boundary consumer of the Higson compactification route. It reads the compactification carrier of chapter 5170, compares its boundary-facing rows with the Stone–Cech surface of chapter 2344, and keeps the uniform source discipline of chapter 339 visible through the metric, continuous-map, and real-valued observation rows. The packet names a finite corona interface; it does not choose boundary points, ultrafilters, quotient compactification classes, or ambient coarse-equivalence witnesses.

Definition 5171.1 (Higson corona carrier)**.** *A* $\mathsf{HigsonCorona}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, B, U, M, F, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{HigsonCompactification}^{\uparrow}$ *boundary row, read as the compactification-side source of corona observations.*
2. *B is the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *comparison row, used only to compare bounded-probe boundary data with the Higson boundary route.*
3. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row inherited by the compactification packet.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *large-scale control row that keeps coarse escape and bounded-region information visible.*
5. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *row of slowly oscillating bounded real-valued observations admitted by the boundary read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for the finite boundary observation family.*
7. *H records componentwise* hsame *transport for $K, B, U, M, F, R, C, P, N$.*
8. *C records displayed* Cont *replay from the compactification boundary row through the finite observation and comparison rows.*
9. *P is the* Pkg *provenance over* $\mathsf{HigsonCorona}^{\uparrow}$, $\mathsf{HigsonCompactification}^{\uparrow}$, $\mathsf{StoneCechCompactification}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, U, M, F, R, H, C, P, N$. It has no coordinate for selected corona points, ultrafilter choice, quotient host compactification equality, ambient properness, or a global coarse compactification theorem.

Theorem 5171.2 (Higson corona NameCert obligations)**.** *For every accepted* $\mathsf{HigsonCorona}^{\uparrow}$ *carrier*

$$Q = (K, B, U, M, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}$ *surface consists exactly of compactification-boundary admission, Stone–Cech comparison, inherited uniform source exposure, large-scale metric control, finite slowly oscillating continuous observation, Real value sealing, componentwise transport, displayed replay, provenance, and local naming.*

Every boundary-facing consumer must read the displayed route

$$K, B, U, M \longmapsto F, R \longmapsto H, C, P, N$$

before it may use a corona observation.

Proof. Project the accepted carrier. The rows K, B, U, M expose the Higson compactification boundary source, the Stone–Cech comparison surface, the uniform source row, and the large-scale metric control row.

The observation rows F, R are downstream of that boundary source. They record only finite slowly oscillating continuous real-valued observations and their Real value seals, so a consumer cannot replace them by an ambient function algebra or an unbounded host observation family.

The structural rows H, C, P, N transport, replay, package, and name the same displayed route. Since the carrier has no coordinate for selected corona points, ultrafilter choice, quotient host compactification equality, ambient properness, or a global coarse compactification theorem, those data are not exported by the local certificate. □

Theorem 5171.3 (Higson corona compactification boundary)**.** *Every accepted* $\mathsf{HigsonCorona}^{\uparrow}$ *boundary read is a consumer of the accepted* $\mathsf{HigsonCompactification}^{\uparrow}$ *route before it is a compactification comparison. In particular, the boundary read must expose the* $\mathsf{HigsonCompactification}^{\uparrow}$ *row, the* $\mathsf{StoneCechCompactification}^{\uparrow}$ *comparison row, the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, the* $\mathsf{Met}^{\uparrow}$ *control row, the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *observation row, the* $\mathsf{Real}^{\uparrow}$ *seal row,* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}$ *row.*

Proof. Read the carrier of Definition 5171.1. The row K is the only compactification-boundary source available to the packet, so every corona request first passes through the Higson compactification route of chapter 5170.

The comparison with Stone–Cech compactification is then read through B and not as a replacement for K. The inherited rows U, M, F, R retain uniform source data, large-scale metric control, finite continuous observations, and Real value seals before any boundary consumer can use the comparison.

Apply Theorem 5171.2. It exhausts the remaining structural rows H, C, P, N and excludes selected boundary points, ultrafilters, quotient host compactification equality, ambient properness, and global coarse compactification data from the boundary surface. □

Falsifiable prediction: Every accepted $\mathsf{HigsonCorona}^{\uparrow}$ boundary read exposes a $\mathsf{HigsonCompactification}^{\uparrow}$ boundary row, a $\mathsf{StoneCechCompactification}^{\uparrow}$ comparison row, $\mathsf{UniformSpace}^{\uparrow}$ source data, $\mathsf{Met}^{\uparrow}$ large-scale control, finite slowly oscillating $\mathsf{ContinuousMap}^{\uparrow}$ observations, $\mathsf{Real}^{\uparrow}$ value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: higson-compactification, stone-cech-compactification, uniformspace, metric, continuousmap, real: $\mathsf{HigsonCorona}^{\uparrow}$ is not bijective to any listed sibling because it binds a Higson compactification boundary consumer to Stone–Cech comparison, uniform source data, large-scale metric control, finite continuous real-valued observations, and local NameCert provenance in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}$.

Constructive story: A $\mathsf{HigsonCorona}^{\uparrow}$ packet is generated from a Higson compactification boundary row, a Stone–Cech comparison row, inherited uniform and metric source rows, finite slowly oscillating continuous real-valued observations, Real value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5171.1, Definition 5171.2, and Definition 5171.3 seed the finite Higson corona boundary consumer over compactification, comparison, uniform, metric, continuous-map, Real, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HigsonCorona}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected corona points, ultrafilter choice, quotient host compactification equality, ambient properness, global coarse compactification theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked boundary admission, comparison stability, inherited uniform and metric source exposure, finite observation replay, value sealing, transport, provenance, and local naming for BEDC.Derived.HigsonCoronaUp.

Chapter 5172

A Concrete Naming Certificate for $\mathsf{ModulusCompositionSchedule}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ModulusCompositionSchedule}^{\uparrow}$ packet records the finite schedule data by which two Cauchy moduli are composed before a regular rational readback is sealed as a Real name. It sits over the finite schedule surface of chapter 292, the stream-window surface of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the modulus arithmetic route of chapter 1485, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion route a concrete object for composing precision schedules; it is not a quotient-real equality principle, selected limit, trichotomy, countable-choice schedule, hidden infinite stream, or ambient completeness theorem.

Definition 5172.1 (Modulus composition schedule carrier)**.** *A* $\mathsf{ModulusCompositionSchedule}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *μ and ν are the two source* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule rows whose precision requests are to be composed.*
2. *W_μ and W_ν are finite* $\mathsf{StreamName}^{\uparrow}$ *windows displaying the inspected reads for the outer and inner schedules.*
3. *ε is the requested* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row at the terminal Real-facing scale.*
4. *δ is the intermediate* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row selected by the outer schedule and then fed to the inner schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after both finite windows have been inspected at the displayed tolerances.*
6. *A is the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *row witnessing that the two schedule reads compose from ε through δ to the displayed regular rational readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from R after the composed schedule has been displayed.*
8. *H records componentwise* hsame *transport for $\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E$ and the local naming row.*

9. *C records displayed* Cont *replay from a composed precision request through two Cauchy-modulus rows, two finite stream windows, dyadic tolerances, arithmetic composition, regular rational readback, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{ModulusCompositionSchedule}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusCompositionSchedule}^{\uparrow}}$ *row.*

The classifier compares accepted schedule packets only through $\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E, H, C, P, N$. It has no coordinate for quotient real representatives, selected limits, trichotomy, countable choice, hidden infinite streams, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ModulusCompositionScheduleUp.ModulusCompositionScheduleTasteGate single carrier alignment`

Theorem 5172.2 (Modulus composition schedule NameCert obligations)**.** *For every accepted* $\mathsf{ModulusCompositionSchedule}^{\uparrow}$ *carrier*

$$M = (\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ModulusCompositionSchedule}^{\uparrow}}$ *surface consists exactly of two* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule exposures, two finite* $\mathsf{StreamName}^{\uparrow}$ *window exposures, terminal and intermediate* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance exposure,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *composition exposure, terminal* $\mathsf{Real}^{\uparrow}$ *seal exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must expose $\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E$ before the structural rows H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5172.1. The analytic rows are exactly $\mu, \nu, W_\mu, W_\nu, \varepsilon, \delta, R, A, E$, while H, C, P, N are the structural transport, replay, provenance, and naming rows.

The terminal tolerance ε is read before the outer schedule row μ supplies the intermediate tolerance δ. The inner schedule row ν is then inspected through W_ν at that intermediate tolerance, so the packet does not contain a free-standing precision oracle.

The readback row R and arithmetic row A bind both schedule reads to the same displayed composition route. Hence a consumer cannot use the Real seal E without first exposing the two moduli, two finite windows, and both dyadic tolerance rows.

Since the carrier has no coordinate for quotient representatives, selected limits, trichotomy, countable choice, hidden streams, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5172.3 (Modulus composition schedule Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{ModulusCompositionSchedule}^{\uparrow}$ *packet reaches the terminal Real seal only through the displayed finite route*

$$\mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyModulusArithmetic}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff may use the two schedule rows, the two inspected stream windows, terminal and intermediate dyadic tolerances, arithmetic composition, and regular rational readback, but it cannot replace them by quotient real equality, a selected limit, trichotomy, countable choice, an uninspected infinite stream, or ambient completeness.

Proof. Begin with the Real or Completion-facing read of M. The terminal tolerance ε and the outer schedule row μ determine the intermediate tolerance δ, so the consumer sees a finite precision request before the inner schedule is inspected.

The stream rows W_μ and W_ν display the only window reads available to the two schedule rows. The arithmetic row A then records that the outer and inner reads compose through the displayed dyadic tolerances.

The regular rational readback R is downstream of this arithmetic route, and the terminal seal E is downstream of R. Therefore the Real handoff factors through the finite schedule packet and the excluded host principles remain absent from the carrier. □

Lean4 ✓ `BEDC.Derived.ModulusCompositionScheduleUp.ModulusCompositionScheduleRealHandoff`

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusCompositionSchedule}\uparrow}$.

Constructive story: A $\mathsf{ModulusCompositionSchedule}^{\uparrow}$ packet is generated from two CauchyModulus schedules, two StreamName windows, terminal and intermediate DyadicRatCore tolerances, RegSeqRat readback, CauchyModulusArithmetic composition, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusCompositionSchedule}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5172.1, Definition 5172.2, and Definition 5172.3 seed the finite composed-modulus schedule route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ModulusCompositionSchedule}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, trichotomy, countable choice, uninspected infinite streams, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, two-modulus exposure, two-window exposure, dyadic tolerance routing, modulus arithmetic exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusCompositionScheduleUp.

Chapter 5173

A Concrete Naming Certificate for $\mathsf{LowDiscrepancySequence}^{\uparrow}$

Following Niederreiter and Kuipers–Niederreiter on uniform distribution and star discrepancy, the $\mathsf{LowDiscrepancySequence}^{\uparrow}$ packet records finite sampling windows whose axis-aligned box counts are available to numerical-integration consumers. It sits over the rational coordinate carrier of chapter 37, the terminal real seal of chapter 38, the tagged Riemann route of chapter 1526, and the Darboux comparison route of chapter 1460. The packet names finite count and discrepancy-budget data; it does not import measure theory, probability, selected infinite sequence data, or an ambient equidistribution theorem.

Definition 5173.1 (Low-discrepancy sequence carrier)**.** *A* $\mathsf{LowDiscrepancySequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, Q, B, D, H, C, G, N)$$

with the following displayed rows:

1. *P is the sample-point window row, listing a finite prefix of sample points with rational coordinate names.*
2. *Q is the rational box-test row, listing anchored axis-aligned rational boxes whose endpoints are read through* $\mathsf{Rat}^{\uparrow}$ *and whose real-facing comparisons are sealed through* $\mathsf{Real}^{\uparrow}$*.*
3. *B is the exact finite-count row, assigning to each displayed box the number of sample points from P that fall inside it.*
4. *D is the star-discrepancy budget row, recording the rational error budget attached to the displayed count comparison.*
5. *H records componentwise* hsame *transport for sample windows, box tests, exact counts, discrepancy budgets, replay, provenance, and local naming.*
6. *C records displayed* Cont *replay from a sampling request through P, Q, B, D, H, G, N.*
7. *G is the* Pkg *provenance over* $\mathsf{LowDiscrepancySequence}^{\uparrow}$*,* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RiemannIntegral}^{\uparrow}$*,* $\mathsf{DarbouxIntegral}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{LowDiscrepancySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through P, Q, B, D, H, C, G, N. It has no coordinate for a host probability space, Lebesgue measure, selected infinite sequence, asymptotic limit, randomized sampler, or ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.LowDiscrepancySequenceUp.TasteGate.LowDiscrepancySequenceTasteGate single carrier alignment`

Theorem 5173.2 (Low-discrepancy sequence NameCert obligations)**.** *For every accepted* $\mathsf{LowDiscrepancySequence}^{\uparrow}$ *carrier* $L = (P, Q, B, D, H, C, G, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LowDiscrepancySequence}^{\uparrow}}$ *surface consists exactly of five finite obligation rows: carrier admission for* P, Q, B, D, H, C, G, N*; rational box-test exactness through* Q*; exact finite membership counts through* B*; star-discrepancy budget exposure through* D*; and transport, replay, provenance, and local naming through* H, C, G, N *with non-escape from host measure, probability, selected infinite sequences, randomized sampling, and ambient completeness.*

Proof. Project the carrier of Definition 5173.1. The only sampling data rows are the finite sample window P, rational box tests Q, exact finite counts B, and discrepancy budgets D.

The rational box-test row is finite. Each box endpoint is read through the rational carrier and compared only through the displayed real-facing seal, so the test row does not add a measure algebra or probability event space.

The exact count row is tied to the displayed window. It records membership counts for the boxes already listed in Q, and the discrepancy row D records the budget attached to those counts rather than an asymptotic theorem.

Finally H, C, G, N transport, replay, source, and name the same packet. Since the excluded host objects are not coordinates of L, the five listed rows exhaust the local NameCert surface. □

Theorem 5173.3 (Low-discrepancy sequence star-discrepancy route)**.** *Every numerical-integration consumer of an accepted* $\mathsf{LowDiscrepancySequence}^{\uparrow}$ *packet may read only the finite route*

$$P, Q \longmapsto B, D \longmapsto H, C, G, N.$$

In particular, a $\mathsf{RiemannIntegral}^{\uparrow}$ *or* $\mathsf{DarbouxIntegral}^{\uparrow}$ *consumer may use the displayed box counts and discrepancy budgets as finite sampling evidence, but it may not read host measure, probability, selected infinite sequence data, randomized sampling, or an asymptotic equidistribution endpoint from the packet.*

Proof. Start with an integration-facing request. The packet first exposes the finite sample window P and the rational box tests Q, so every downstream read is already tied to displayed finite data.

The rows B and D are the only numerical sampling rows. The count row supplies exact finite membership counts, and the discrepancy row supplies the rational error budget available to the integration consumer.

The structural rows H, C, G, N preserve the same route. They do not add measure, probability, selected infinite sequence data, randomized sampling, or asymptotic convergence data, so the integration consumer remains inside the finite star-discrepancy route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowDiscrepancySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LowDiscrepancySequence}^{\uparrow}$ packet is generated from finite sample-point windows, rational box tests, exact finite counts, star-discrepancy budgets, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowDiscrepancySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5173.1, Definition 5173.2, and Definition 5173.3 seed the finite star-discrepancy sampling certificate for Riemann and Darboux integration consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowDiscrepancySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host measure, probability, selected infinite sequence data, randomized sampling, asymptotic equidistribution, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational box-test exactness, finite-count exactness, discrepancy-budget exposure, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.LowDiscrepancySequenceUp.

Chapter 5174

A Concrete Naming Certificate for MetaCICConfluenceBudgetWitness$^{\uparrow}$

The MetaCICConfluenceBudgetWitness$^{\uparrow}$ packet realizes the finite confluence-budget witness route of chapter 9378. It sits beside the local diamond witness of chapter 3595, the residual envelope of chapter 3921, the substitution-boundary packet of chapter 3791, the closed-premise ledger of chapter 2325, and the residual-budget packet of chapter 4061. The witness binds finite budget rows used by confluence-facing readers while keeping the boundary against full confluence explicit.

Definition 5174.1 (MetaCIC confluence-budget witness carrier)**.** *A* MetaCICConfluenceBudgetWitness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (T, F, R, J, E, S, L, D, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-window row naming the closed MetaCIC terms and finite context fragments inspected by the witness.*
2. *F is the redex-frontier row listing the beta and congruence faces admitted to the budget.*
3. *R is the residual-budget row that bounds residual addresses, replay fuel, and candidate-join inspection.*
4. *J is the candidate-join row, recording only the joins accepted inside the displayed budget.*
5. *E is the residual-envelope row read from the confluence residual-envelope packet.*
6. *S is the substitution-boundary row that keeps residual-substitution compatibility visible as a boundary condition.*
7. *L is the closed parallel-diamond premise ledger row.*
8. *D is the local-diamond witness row consumed as finite local geometry.*
9. *O is the obstruction-retention row, recording that full parallel diamond and arbitrary closed Church–Rosser are not supplied by the packet.*
10. *H records componentwise* hsame *transport for typed windows, frontiers, budgets, joins, envelopes, substitution boundaries, premise ledgers, local diamonds, obstruction rows, replay, provenance, and local naming.*

11. *C records displayed* Cont *replay from a confluence-budget request through* $T, F, R, J, E, S, L, D, O, H, P, N$.
12. *P is the* Pkg *provenance over* $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$, $\mathsf{MetaCICDiamondResidualBudget}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, $\mathsf{MetaCICConfluenceResidualEnvelope}^{\uparrow}$, $\mathsf{MetaCICConfluenceSubstitutionBoundary}^{\uparrow}$ $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, F, R, J, E, S, L, D, O, H, C, P, N$. It has no coordinate for full parallel diamond, arbitrary closed Church–Rosser, global strong normalization, substitution compatibility, subject reduction, or host reduction semantics.

Theorem 5174.2 (MetaCIC confluence-budget witness NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$ *carrier $W = (T, F, R, J, E, S, L, D, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}}$ *surface consists exactly of typed-window admission, redex-frontier exposure, residual-budget accounting, candidate-join recording, residual-envelope handoff, substitution-boundary visibility, closed-premise-ledger handoff, local-diamond handoff, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5174.1. The mathematical rows are precisely $T, F, R, J, E, S, L, D, O$, followed by the structural rows H, C, P, N.

The budget is finite before any join is consumed. Rows T, F, R bind the typed window, redex frontier, and residual fuel, while J records only the candidate joins accepted inside that displayed budget.

The sibling handoffs remain visible. The residual envelope E, substitution boundary S, closed-premise ledger L, and local diamond row D are not collapsed into an endpoint theorem; the obstruction row O records the refused endpoint claims.

Rows H, C, P, N transport, replay, source, and name the same finite route. Since no excluded endpoint is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5174.3 (MetaCIC confluence-budget witness endpoint boundary)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$ *packet reads only the finite route*

$$T, F, R \longmapsto J \longmapsto E, S, L, D, O \longmapsto H, C, P, N.$$

The route may expose candidate joins under a residual budget, but it does not assert full parallel diamond, arbitrary closed Church–Rosser, global strong normalization, substitution compatibility, subject reduction, or host reduction semantics.

Proof. Begin with the displayed witness route. The typed window, redex frontier, and residual budget are read before the candidate-join row, so the join is budget-mediated rather than global.

The downstream rows keep the audit boundary intact. Residual envelope, substitution boundary, closed-premise ledger, and local diamond data are exposed as sibling handoffs, while the obstruction row remains in the same packet.

The structural rows preserve the route without adding endpoint proof data. Thus a confluence-facing consumer receives finite budget evidence and no coordinate for the refused global claims. □

Theorem 5174.4 (MetaCIC confluence-budget residual coverage)**.** *Every accepted* MetaCICConfluenceBudgetWitne *packet that is consumed as residual-substitution evidence factors through the residual-budget row R, candidate-join row J, residual-envelope row E, substitution-boundary row S, closed-premise ledger L, local-diamond row D, and obstruction row O. It cannot be read as full parallel diamond,*

arbitrary closed Church–Rosser, or unguarded substitution compatibility; residual-substitution consumers must keep the guarded input shape of chapter 4370 visible.

Proof. Project the carrier $W = (T, F, R, J, E, S, L, D, O, H, C, P, N)$ from Definition 5174.1. The residual-facing coordinates are displayed as R, J, E, S, L, D, O, with transport, replay, provenance, and naming downstream.

Apply Definition 5174.2. The NameCert surface requires residual-budget accounting, candidate-join recording, residual-envelope handoff, substitution-boundary visibility, closed-premise-ledger handoff, local-diamond handoff, and obstruction retention as separate rows.

The refused endpoints are fixed by Definition 5174.3. That theorem prevents the same packet from being read as full parallel diamond, arbitrary closed Church–Rosser, or an unguarded substitution theorem.

Finally, the guarded input shape is the one recorded by chapter 4370. Thus a residual-substitution consumer receives only the displayed budgeted rows and the explicit boundary, not an endpoint confluence claim. □

Falsifiable prediction: A packet omitting either the residual-budget row R or the obstruction-retention row O is rejected as a $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$ packet, even when it carries typed-window, local-diamond, or residual-envelope rows.

Independence witness: metacic-local-diamond-witness, metacic-confluence-residual-envelope, metacic-confluence-substitution-boundary, closed-parallel-diamond-premise-ledger, metacic-diamond-residual-budget: $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds typed windows, redex frontiers, residual budgets, candidate joins, residual envelopes, substitution boundaries, closed-premise ledgers, local diamonds, obstruction retention, transport, replay, provenance, and local NameCert data.

The Lean-side hardening surface requires Nontrivial and FieldFaithful instances for the confluence-budget witness packet. The nontrivial witness separates an accepted packet with displayed residual-budget and obstruction rows from a refused packet missing one of those rows; field faithfulness compares $T, F, R, J, E, S, L, D, O, H, C, P, N$ componentwise so that equal displayed fields are indistinguishable to the local NameCert classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}$ packet is generated from typed-window rows, redex-frontier rows, residual budgets, candidate joins, residual envelopes, substitution-boundary rows, closed-premise ledgers, local diamonds, obstruction retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5174.1, Definition 5174.2, and Definition 5174.3 seed the finite confluence-budget witness for MetaCIC confluence-audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICConfluenceBudgetWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full parallel diamond, arbitrary closed Church–Rosser, global strong normalization, substitution compatibility, subject reduction, host reduction semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, budget accounting, candidate-join exposure, residual-envelope handoff, substitution-boundary visibility, premise-ledger handoff, local-diamond handoff, obstruction retention, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MetaCICConfluenceBudgetWitnessUp.

$\mathsf{LawOfTotalVariance}^{\uparrow}$ routes its variance ledger through chapter 190 and chapter 2246 after the probability, random-variable, conditional-expectation, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are displayed.

Chapter 5175

A Concrete Naming Certificate for $\mathsf{LawOfTotalVariance}^{\uparrow}$

Following Kolmogorov and Doob on conditional expectation and variance decompositions, the $\mathsf{LawOfTotalVariance}^{\uparrow}$ packet records the finite Real-valued ledger behind the identity that total variance splits into conditional variance and variance of the conditional mean. It sits after the normalized probability surface of chapter 184, the carried random-variable rows of chapter 185, the singleton conditional-expectation rows of chapter 188, the Real seal of chapter 38, and the regular rational readback surface of chapter 301. The existing martingale variance consumer route reads this chapter through chapter 190 only after the conditional-mean and conditional-variance rows are exposed as the ledger consumed by martingale variance accounting. The chapter names the finite BEDC rows needed by later martingale and variance consumers; it does not import a host probability space, quotient almost-sure equality, an unrestricted conditional-expectation operator, or ambient real completeness.

Definition 5175.1 (LawOfTotalVariance carrier)**.** *A* $\mathsf{LawOfTotalVariance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, X, C, M, V, W, T, S, R, H, K, Q, N)$$

with the following displayed rows:

1. *P is the source* $\mathsf{ProbSpace}^{\uparrow}$ *row, exposing the total-event normalization, measurable-event classifier, and Real-valued probability readbacks used by the variance request.*
2. *X is the centered* $\mathsf{RandomVar}^{\uparrow}$ *row whose finite-window square readback is being decomposed.*
3. *C is the* $\mathsf{CondExp}^{\uparrow}$ *row that displays the conditioning face for the inspected finite window.*
4. *M is the conditional-mean row, read as the displayed conditional expectation of X along C.*
5. *V is the conditional-variance row, read as the conditional expectation of the squared residual between X and M on the same conditioning face.*
6. *W is the total-variance row, read as the Real-valued finite-window expectation of the squared residual from the total mean.*
7. *T is the variance-of-conditional-mean row, read as the Real-valued finite-window variance of M.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal comparing W with the displayed sum of V and T through the Real classifier.*

9. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row recording the rational windows, tolerances, and finite square-expansion approximants used before the Real seal is exposed.*
10. *H records componentwise* hsame *transport for $P, X, C, M, V, W, T, S, R$ and the structural rows below.*
11. *K records displayed* Cont *replay through the probability source, random-variable row, conditional-expectation row, variance rows, regular-rational readback, and Real seal.*
12. *Q is the* Pkg *provenance over* $\mathsf{LawOfTotalVariance}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{LawOfTotalVariance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host almost-sure quotient equality, an external L^2 space, countable-choice representatives, or an ambient conditional-expectation theorem.

Theorem 5175.2 (LawOfTotalVariance NameCert obligations)**.** *For every accepted* $\mathsf{LawOfTotalVariance}^{\uparrow}$ *carrier $L = (P, X, C, M, V, W, T, S, R, H, K, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{LawOfTotalVariance}^{\uparrow}}$ *surface consists exactly of the displayed probability source, random-variable endpoint, conditional-expectation face, conditional-mean row, conditional-variance row, total-variance row, variance-of-conditional-mean row, Real seal, RegSeqRat readback, componentwise transport, replay, provenance, and local naming rows. No obligation may be discharged by importing an ambient variance theorem, quotient almost-sure equality, a host conditional-expectation operator, countable choice, or an external real-completeness row.*

Proof. Project the carrier of Definition 5175.1. The rows P, X, C supply the probability source, the carried random variable, and the conditioning face; the rows M, V, W, T expose the conditional mean, conditional variance, total variance, and variance of the conditional mean.

The comparison is finite before it is Real-facing. The row R records the regular rational windows and tolerance data used by the square-expansion readbacks, while S is the only terminal Real seal available to the consumer.

Finally the structural rows H, K, Q, N preserve and name the same packet through componentwise transport, displayed replay, provenance, and local certification. Since the excluded host theorem, quotient, choice, and completeness coordinates are absent from L, they cannot discharge any local obligation. □

Theorem 5175.3 (LawOfTotalVariance conditional-mean decomposition)**.** *Every accepted* $\mathsf{LawOfTotalVariance}^{\uparrow}$ *decomposition read factors through the finite route*

$$P, X, C \longmapsto M \longmapsto V, T \longmapsto R \longmapsto S \longmapsto H, K, Q, N.$$

Thus a total-variance consumer receives the displayed equality between the total variance row W and the sum of the conditional-variance row V with the variance-of-conditional-mean row T only after reading the probability source, random-variable endpoint, conditional-expectation face, conditional mean, finite-window RegSeqRat approximants, Real seal, transport, replay, provenance, and local NameCert data.

Proof. Begin with the source rows. The probability packet P fixes the measurable-event and Real-valued mass surface, while X supplies the carried random-variable endpoint and C supplies the conditioning face through the existing $\mathsf{CondExp}^{\uparrow}$ interface.

Next read the mean and variance rows. The row M is the displayed conditional mean, V is the conditional variance around that mean, W is the total variance, and T is the variance of the conditional mean. These rows are compared only on the finite window named by the packet.

The rows R and S then move the comparison from rational approximants to the Real seal. The structural rows H, K, Q, N keep the same route under transport and replay. Therefore the decomposition exposed to downstream consumers is exactly the displayed finite BEDC ledger, not a host-side variance theorem. □

Falsifiable prediction: A total-variance read that omits the conditioning face, the conditional-mean row, the RegSeqRat finite-window readback, or the terminal Real seal is not accepted by the local NameCert.

Independence witness: probspace, randomvar, condexp, real, regseqrat: $\mathsf{LawOfTotalVariance}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the probability source, random-variable endpoint, conditioning face, conditional mean, conditional variance, total variance, variance of the conditional mean, finite-window readback, Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LawOfTotalVariance}^{\uparrow}}$.

Constructive story: A $\mathsf{LawOfTotalVariance}^{\uparrow}$ packet is generated from a ProbSpace row, RandomVar row, CondExp row, conditional-mean row, conditional-variance row, total-variance row, variance-of-conditional-mean row, RegSeqRat finite-window readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LawOfTotalVariance}^{\uparrow}}$, seed)

Scope closed: Definition 5175.1, Definition 5175.2, and Definition 5175.3 seed the finite variance-decomposition route through ProbSpace, RandomVar, CondExp, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LawOfTotalVariance}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host almost-sure quotient equality, an unrestricted conditional-expectation operator, an external L^2 theorem, countable-choice representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, axiom-clean audit, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, conditional-mean routing, conditional-variance readback, total-variance readback, variance-of-conditional-mean readback, RegSeqRat tolerance accounting, Real sealing, transport stability, replay stability, provenance, and local naming for BEDC.Derived.LawOfTotalVarianceUp.

Theorem 5175.4 (LawOfTotalVariance martingale variance consumer route)**.** *Every* $\mathsf{Martingale}^{\uparrow}$ *or* $\mathsf{MartingaleConvergence}^{\uparrow}$ *variance-facing consumer of an accepted* $\mathsf{LawOfTotalVariance}^{\uparrow}$ *packet must first read the displayed* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*,* $\mathsf{CondExp}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 5175.1. Only after those rows are exposed may the consumer route the variance decomposition through chapter 190 or chapter 2246.*

The route exports no host almost-sure quotient, unrestricted conditional-expectation operator, countable-choice representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 5175.1. The probability source P, random-variable endpoint X, conditioning face C, regular-rational readback R, and Real seal S are visible before any variance-facing consumer can read M, V, W, T as a decomposition package.

For a $\mathsf{Martingale}^{\uparrow}$ consumer, replay the displayed probability, random-variable, and conditional-expectation rows through chapter 190. The variance row is therefore attached to the finite adapted-sequence surface rather than to a host random-variable quotient.

For a $\mathsf{MartingaleConvergence}^{\uparrow}$ consumer, route the same displayed rows through chapter 2246. The convergence-facing read receives the bounded-window and rational-readback surface already sealed by R and S, not an unrestricted conditional-expectation operator or an ambient completeness theorem.

The carrier has no coordinate for almost-sure quotient equality, countable choice, a host conditional-expectation operator, or external real completeness. Thus the martingale-facing route can export only the displayed finite rows named above. □

$\mathsf{FrechetBaireSpace}^{\uparrow}$ consumes chapter 3495 and chapter 3535 before its route reaches chapter 3570.

Chapter 5176

A Concrete Naming Certificate for $\mathsf{FrechetBaireSpace}^{\uparrow}$

Following Baire and Bourbaki, the $\mathsf{FrechetBaireSpace}^{\uparrow}$ packet records the finite handoff from a complete metrizable locally convex carrier to the complete-metric Baire route. It sits after the $\mathsf{FrechetSpace}^{\uparrow}$ carrier of chapter 3495, the theorem-level dense-open package of chapter 3570, the inspected complete-metric surface of chapter 3535, the topological-vector source of chapter 205, the locally convex source of chapter 3386, and the metric-completion handoff of chapter 1506. In the sibling lattice, the chapter consumes chapter 3495 and chapter 3535 before its route reaches chapter 3570. The packet names only the finite route by which a Frechet-space consumer may receive Baire-category evidence; it does not add Hahn–Banach separation, selected generic points, countable choice, host intersections, or ambient complete locally convex spaces.

Definition 5176.1 (Frechet-space Baire carrier)**.** *A* $\mathsf{FrechetBaireSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, T, U, V, L, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetSpace}^{\uparrow}$ *source row, exposing complete metrizability and local convexity through Definition 3495.1.*
2. *T is the* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *theorem route row, carrying the dense-open theorem surface.*
3. *U is the* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *inspection row, carrying the complete-metric Baire surface consumed by T.*
4. *V is the* $\mathsf{TopVecSpace}^{\uparrow}$ *source row inherited from the Frechet carrier.*
5. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *source row inherited from the Frechet carrier.*
6. *Q is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row that exposes the compatible complete metric boundary.*
7. *H records componentwise* hsame *transport for F, T, U, V, L, Q and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the Frechet carrier, Baire theorem route, complete-metric Baire surface, topological-vector source, locally convex source, metric-completion handoff, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{FrechetBaireSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{BaireCategoryTheorem}^{\uparrow}$, CompleteMetricBaireSu $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetBaireSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, T, U, V, L, Q, H, C, P, N$. It has no coordinate for a host Frechet space, an external Baire intersection, a selected generic point, Hahn–Banach separation, countable choice, or ambient completeness outside the displayed route.

Theorem 5176.2 (Frechet-space Baire completeness route)**.** *For every accepted* $\mathsf{FrechetBaireSpace}^{\uparrow}$ *carrier $B = (F, T, U, V, L, Q, H, C, P, N)$, the Baire-category read factors through the Frechet source row F, the complete metric row and metric-completion handoff already exposed by Definition 3495.1, the inspected* $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$ *row U, and the theorem-level* $\mathsf{BaireCategoryTheorem}^{\uparrow}$ *row T. The topological-vector and locally convex rows V, L remain visible throughout the route.*

Proof. Project the carrier of Definition 5176.1. The Frechet source row F supplies the displayed complete metrizable locally convex packet, while V, L, Q retain the inherited topological-vector, locally convex, and metric-completion coordinates.

Use the Frechet dependency route of Definition 3495.4. It sends every Baire request through the complete metric row and the metric-completion handoff before the request reaches the Baire surfaces.

The row U supplies the complete-metric Baire inspection surface, and T supplies the theorem-level dense-open package. The structural rows H, C, P, N transport, replay, source, and name only this route, so the listed rows exhaust the completeness-to-Baire handoff. □

Theorem 5176.3 (Frechet-space Baire non-escape)**.** *Every accepted* $\mathsf{FrechetBaireSpace}^{\uparrow}$ *consumer receives Baire-category evidence only through*

$$\mathsf{FrechetSpace}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{BaireCategoryTheorem}^{\uparrow}$$

together with the displayed $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. The carrier exports no Hahn–Banach theorem, no selected generic point, no countable-choice schedule, no host intersection object, no quotient completion equality, and no ambient complete locally convex space.*

Proof. Start with the route theorem. It confines Baire-category evidence to the Frechet source, complete-metric Baire inspection, and theorem-level dense-open rows.

Inspect the carrier coordinates. The rows V, L, Q preserve the topological-vector, locally convex, and metric-completion handoff data already present in the Frechet carrier, while H, C, P, N provide only structural transport, replay, provenance, and naming.

Since none of the coordinates stores Hahn–Banach separation, a selected generic point, countable choice, a host intersection object, quotient completion equality, or an ambient complete locally convex space, no accepted consumer can extract those readings from the packet. □

Theorem 5176.4 (Frechet-space Baire metric-completion forward lattice)**.** *Every accepted* $\mathsf{FrechetBaireSpace}^{\uparrow}$ *consumer routes Baire evidence through the forward lattice*

$$\mathsf{FrechetSpace}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{CompleteMetricBaireSurface}^{\uparrow} \longmapsto \mathsf{BaireCategoryTheorem}^{\uparrow}$$

before exporting a dense-open conclusion. The rows $F, T, U, V, L, Q, H, C, P, N$ must remain visible, with V and L retaining the topological-vector and locally convex sources and Q retaining the metric-completion handoff.

Proof. Project the carrier rows of Definition 5176.1. The Frechet source row F carries the complete metrizable locally convex surface, while V and L keep the inherited topological-vector and locally convex data visible.

Apply the completeness route, Definition 5176.2. It sends the Baire request through the complete metric boundary and the metric-completion handoff Q before the inspected complete-metric Baire surface U and theorem-level row T are read.

Finally use non-escape, Definition 5176.3. Since the dense-open conclusion can be exported only through $F, T, U, V, L, Q, H, C, P, N$, the structural rows H, C, P, N preserve the same forward lattice and cannot replace the metric-completion handoff by an ambient host completion or a selected generic point. □

Falsifiable prediction: Any accepted $\mathsf{FrechetBaireSpace}^{\uparrow}$ consumer that reads Baire-category evidence without first exposing $F, T, U, V, L, Q, H, C, P, N$ refutes the carrier surface of Definition 5176.1.

Independence witness: frechet-space, baire-category-theorem, complete-metric-baire-surface, topvecspace, locally-convex-space, metriccompletion: no listed sibling is bijective to $\mathsf{FrechetBaireSpace}^{\uparrow}$, because this packet alone binds the Frechet source, complete-metric Baire inspection, theorem-level dense-open route, topological-vector row, locally convex row, metric-completion handoff, transport, replay, provenance, and local NameCert rows into one finite handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetBaireSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{FrechetBaireSpace}^{\uparrow}$ packet is generated from a FrechetSpace source row, a BaireCategoryTheorem route row, a CompleteMetricBaireSurface inspection row, TopVecSpace and LocallyConvexSpace source rows, a MetricCompletion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetBaireSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5176.1, Definition 5176.2, Definition 5176.3, and Definition 5176.4 seed the finite Frechet-to-Baire handoff over $F, T, U, V, L, Q, H, C, P, N$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetBaireSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Hahn–Banach separation, selected generic points, countable choice, host intersections, quotient completion equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Frechet source exposure, complete-metric Baire inspection, theorem-level dense-open routing, inherited topological-vector and locally convex rows, metric-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.FrechetBaireSpaceUp.

Chapter 5177

A Concrete Naming Certificate for $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$

Following Crick, Watson, and the standard genetic-code table, the $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ packet records the finite bridge from decoded BEDC channel event flows to the biological four-base and codon-reading analogy of subsection 9462.28.16. It sits after the $\mathsf{Bool}^{\uparrow}$ endpoint discipline of chapter 32, the $\mathsf{Prod}^{\uparrow}$ finite product surface of chapter 34, the DNA four-phase template of Definition 9462.1067, the codon atlas of Definition 9462.1072, the genetic translation classifier of Definition 9462.1074, and the stop-role bridge of Definition 9462.1079. Its codon-atlas and translation-classifier rows are downstream-compatible with the finite codon-window row of chapter 2234 and the stop-boundary transition row of chapter 2245. The packet names a conservative scientific-model carrier; it does not identify channel glyphs with DNA bases, codons, biochemical laws, or theorem judgments.

Definition 5177.1 (Channel/genetic-code bridge carrier)**.** *A* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, B, P, A, T, S, L, H, C, R, N)$$

with the following displayed rows:

1. *D is the decoded channel event-flow row, enforcing the decode-first boundary before any biological reading is attempted.*
2. *B is the* $\mathsf{Bool}^{\uparrow}$ *endpoint row used twice by the DNA four-phase product.*
3. *P is the* $\mathsf{Prod}^{\uparrow}$ *product row that displays the two Boolean axes of the DNA four-phase alphabet.*
4. *A is the DNA four-phase atlas row, reading the four product phases as external biological bases.*
5. *T is the codon atlas row, displaying the triple product of DNA four-phase rows and its* 64 *phase positions.*
6. *S is the genetic translation classifier row, separating amino-acid-emitting classes from stop classes while retaining the original codon ledger.*
7. *L is the stop-role and cannot-claim ledger, separating channel event termination from biological translation termination and refusing token identity.*
8. *H records componentwise* hsame *transport for D, B, P, A, T, S, L and for the replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through decoded channel data, Boolean endpoints, product phases, DNA atlas rows, codon triples, translation classification, stop-role refusal, transport, provenance, and local naming.*

10. *R is the* Pkg *provenance over* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, *the finite codon-window rows of chapter 2234, the stop-boundary transition rows of chapter 2245,* BHist, hsame, Cont, Pkg, NameCert, *and the originating vision route of subsection 9462.28.16. These siblings provide finite codon-window and stop-boundary transition rows only; this chapter still refuses biochemical identity and theorem evidence.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ChannelGeneticCodeBridge}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, B, P, A, T, S, L, H, C, R, N$. It has no coordinate for raw channel-token identity, DNA base identity, RNA codon identity, biochemical law, protein synthesis proof, or theorem evidence.

Theorem 5177.2 (Channel/genetic-code bridge NameCert obligations)**.** *For every accepted* ChannelGeneticCodeBr *carrier $G = (D, B, P, A, T, S, L, H, C, R, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChannelGeneticCodeBridge}^{\uparrow}}$ *surface consists exactly of decoded-channel exposure through D, Boolean endpoint exposure through B, product-phase exposure through P, DNA four-phase atlas exposure through A, codon-atlas exposure through T, translation-classifier exposure through S, stop-role and cannot-claim refusal through L, componentwise transport through H, displayed replay through C, provenance through R, and local naming through N.*

Proof. Project the carrier of Definition 5177.1. The source-facing row is D, the finite phase rows are B, P, A, T, the classifier row is S, and the refusal row is L.

The decode-first row D is consumed before any biological interpretation. Thus a channel substring is not read as a base or codon until it has been decoded to event-flow data.

The finite alphabet route then passes through B and P. The two $\mathsf{Bool}^{\uparrow}$ axes are displayed by the $\mathsf{Prod}^{\uparrow}$ row before the external DNA four-phase atlas A and codon triple atlas T are read.

The translation classifier S may compress distinct codon positions to the same amino-acid or stop class, but it retains the codon ledger. Finally L, H, C, R, N refuse token identity and transport, replay, source, and name only the displayed route. No further NameCert obligation is stored outside these rows. □

Theorem 5177.3 (Channel/genetic-code stop-role non-escape)**.** *Every accepted* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ *consumer may compare the channel terminator role with the biological stop-codon role only through the stop-role ledger L after the decoded channel row, DNA four-phase product row, codon atlas row, and translation classifier row have been displayed. The carrier exports no claim that* 11 *is a DNA or RNA stop codon, no claim that a channel glyph is a biological base, no direct read from compiled channel substrings to codons, and no biological theorem evidence.*

Proof. Start from the NameCert obligations. They force the route through $D, B, P, A, T, S, L, H, C, R, N$ before any accepted bridge read is exported.

The channel side and biological side have distinct typing roles. The row D supplies decoded event-flow data, while A, T, S supply external biological readbacks through product phases, codon triples, and translation classes.

The ledger L is therefore a refusal boundary. It permits role analogy between finite stopping processes, but it blocks token identity between the channel terminator and biological stop codons, direct channel-substring-to-base reads, and any conversion of biological analogy into theorem evidence. □

Falsifiable prediction: Any accepted $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ consumer that treats a raw channel substring as a DNA base, RNA codon, stop codon, biochemical law, or theorem judgment without first reading $D, B, P, A, T, S, L, H, C, R, N$ refutes the carrier surface of Definition 5177.1.

Independence witness: bool, prod, channel-genetic-code vision: no listed sibling is bijective to $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$, because this packet alone binds decoded channel data, Boolean product phases, DNA four-phase readback, codon atlas rows, translation classification, stop-role refusal, transport, replay, provenance, and local NameCert rows into one conservative biological bridge.

Closure status for $\mathsf{NameCert}_{\mathsf{ChannelGeneticCodeBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ packet is generated from a decoded channel event-flow row, Bool endpoint rows, a Prod product-phase row, a DNA four-phase atlas, a codon triple atlas, a genetic translation classifier, a stop-role refusal ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChannelGeneticCodeBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5177.1, Definition 5177.2, and Definition 5177.3 seed the finite conservative bridge over $D, B, P, A, T, S, L, H, C, R, N$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChannelGeneticCodeBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not identify channel glyphs with DNA bases, RNA codons, biological stop codons, biochemical laws, protein synthesis proof, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked decoded-channel exposure, Boolean endpoint exposure, product-phase exposure, DNA four-phase atlas exposure, codon-atlas exposure, translation-classifier exposure, stop-role refusal, transport, replay, provenance, and local naming for BEDC.Derived.ChannelGeneticCodeBridgeUp.

Lemma 5177.4 (Channel/genetic-code finite endpoint dependency route)**.** *Every accepted* ChannelGeneticCodeBric *read exposes the* $\mathsf{Bool}^{\uparrow}$ *endpoint row and the* $\mathsf{Prod}^{\uparrow}$ *phase row before any DNA four-phase atlas row or codon atlas row is consumed. The dependency route is the finite endpoint surface of chapter 32 followed by the product-phase surface of chapter 34; only after those two rows are visible may the bridge read DNA four-phase or codon atlas data. Termination-role reads also pass through chapter 5967 before the stop-role bridge compares channel termination with codon stop roles. The codon stop-face read also factors through chapter 8421, so the bridge compares channel termination with biological stop classes only after the finite codon-window classifier has exposed the stop face.*

Theorem 5177.5 (Channel/genetic-code codon stop sibling route)**.** *Every stop-role read consumed by an accepted* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ *packet factors first through the channel termination budget certificate of chapter 5967 and the codon-window stop-face classifier of chapter 8421. Only after those two sibling routes are displayed may the bridge compare channel termination with biological stop-role analogy. The route exports no token identity between channel events and codons, no direct biological stop-codon theorem, and no MetaCIC proof object.*

Proof. Project the bridge carrier of Definition 5177.1. The decoded channel row D and stop-role ledger L are the only rows from which a stop-role comparison can be read.

The channel side is read first through chapter 5967. That sibling supplies the finite termination-budget certificate before the bridge may describe a channel event as termination-facing.

The codon side is then read through chapter 8421. That sibling supplies the finite codon-window stop-face classifier before the biological stop-role analogy is consumed.

Finally the bridge replays the same route through its structural rows. Since the packet has no coordinate for channel-codon token identity, biological stop-codon proof, or MetaCIC theorem evidence, the sibling route remains a dependency boundary rather than a theorem transfer. □

Lemma 5177.6 (Channel/genetic-code bioreality forward lattice route)**.** *Every accepted* ChannelGeneticCodeBridge *read that is consumed by a biological-reality or MetaCIC-facing route factors through the finite endpoint dependency route of Definition 5177.4, the codon stop sibling route of Definition 5177.5, and the MetaCIC forward link of Definition 5177.7. Thus the bridge can cite chapter 8421 and chapter 5547 only after the decoded channel row, Bool endpoint row, Prod phase row, codon-window stop face, stop-role ledger, transport, replay, provenance, and local NameCert rows are displayed; it exports no direct channel-token identity, biological theorem, confluence theorem, normalization theorem, or subject-reduction theorem.*

Theorem 5177.7 (Channel/genetic-code bridge MetaCIC forward link)**.** *Any MetaCIC or channel-frontier consumer of an accepted* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ *packet must first read the decoded channel row D, the* $\mathsf{Bool}^{\uparrow}$ *endpoint row B, the* $\mathsf{Prod}^{\uparrow}$ *phase row P, the stop-role ledger L, and the finite endpoint dependency route of Definition 5177.4. Only after those rows are displayed may the consumer use biological analogy as a route annotation for a MetaCIC-facing handoff such as chapter 5547. The link is a dependency route over finite channel data and frontier bookkeeping; it is not a biological theorem, not a channel-token identity claim, and not a confluence, normalization, or subject-reduction proof.*

Proof. Read the carrier rows of Definition 5177.1. The decoded channel row D supplies the event-flow source, B and P expose the Boolean endpoint and product phase boundaries, and L records the stop-role refusal boundary.

Next apply the finite endpoint dependency route. It forces endpoint and product-phase exposure before any DNA four-phase, codon-atlas, or stop-face read is consumed, so the biological vocabulary remains downstream of the displayed finite channel route.

A MetaCIC consumer may therefore cite the packet only as routed frontier annotation. The target consumer keeps its own typed-window, confluence, normalization, or obstruction rows, while $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ contributes only D, B, P, L and the finite endpoint route. □

Theorem 5177.8 (Channel/genetic-code bridge MetaCIC forward route)**.** *Every MetaCIC-facing read of an accepted* $\mathsf{ChannelGeneticCodeBridge}^{\uparrow}$ *packet passes through the displayed codon-stop, endpoint, bioreality, and MetaCIC rows before the read may cite chapter 5547. Concretely, the route first exposes the finite endpoint dependency of Definition 5177.4, then the codon-window stop sibling route of Definition 5177.5, then the MetaCIC forward link of Definition 5177.7, and finally the bioreality forward lattice route of Definition 5177.6. The exported read is only a routed frontier annotation over $D, B, P, A, T, S, L, H, C, R, N$; it exports no channel-token identity, biological theorem, confluence theorem, normalization theorem, subject-reduction theorem, or MetaCIC proof object.*

Proof. Start with the carrier of Definition 5177.1. The decoded channel row D, endpoint row B, product row P, codon atlas row T, classifier row S, and stop-role ledger L are visible before any biological or MetaCIC-facing read is available.

Apply the finite endpoint dependency route and the codon-window sibling route. These force the endpoint and product rows to precede the codon-stop comparison, so a stop-role read remains tied to the displayed channel termination budget and codon-window stop-face classifier.

Next apply the MetaCIC forward link. It allows a MetaCIC consumer to cite the packet only as a frontier annotation after D, B, P, L and the finite endpoint route have been displayed.

Finally use the bioreality forward lattice route. It composes the codon-stop and MetaCIC rows with the biological-reality route while preserving the same transport, replay, provenance, and local naming rows. Since no carrier coordinate stores a biological theorem, channel-token identity,

confluence, normalization, subject reduction, or proof object, the read remains a finite horizontal lattice edge into the MetaCIC candidate-normalization confluence handoff. □

Chapter 5178

A Concrete Naming Certificate for $\mathsf{FejerTheorem}^{\uparrow}$

Following Fejer, Zygmund, and Katznelson on positive summability of Fourier series, the $\mathsf{FejerTheorem}^{\uparrow}$ packet records the theorem-level handoff from finite Fourier coefficient windows and positive Fejer kernels to a uniform approximation read. It sits over the Fourier coefficient surface of chapter 124, the positive kernel certificate of chapter 1939, the circle phase source of chapter 58, the normalization route of chapter 123, and the approximation-facing boundary of chapter 463. The chapter names a finite summability theorem certificate; it does not import Haar measure, a host L^1 space, quotient equality of functions, selected Fourier representatives, pointwise convergence, or ambient compactness.

Definition 5178.1 (Fejer theorem carrier)**.** *A* $\mathsf{FejerTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, K, S, I, U, A, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Fourier}^{\uparrow}$ *coefficient-window row, listing the finite symmetric coefficient requests and the polynomial window consumed by the summability read.*
2. *K is the* $\mathsf{FejerKernel}^{\uparrow}$ *positivity and triangular-weight row, including the kernel normalization and finite averaging comparison exposed by the sibling kernel certificate.*
3. *S is the* $\mathsf{S}^{1\uparrow}$ *phase-sample row, recording the circle points on which the finite averages are inspected.*
4. *I is the* $\mathsf{Integral}^{\uparrow}$ *normalization row, exposing only the finite integral cells needed to read the averaged trigonometric polynomial.*
5. *U is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, recording the tolerance, tail, and uniform-read boundary produced by the positive kernel averages.*
6. *A is the theorem assertion row, pairing the displayed Fourier window with the positive-kernel average before any approximation consumer reads the conclusion.*
7. *L is the non-escape ledger for Haar measure, host L^1 theory, quotient function equality, selected representatives, pointwise convergence, and ambient compactness.*
8. *H records componentwise* hsame *transport for $F, K, S, I, U, A, L, C, P, N$.*

9. *C records displayed* Cont *replay from a Fejer-summability request through F, K, S, I, U, A, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{FejerTheorem}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{FejerKernel}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FejerTheorem}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose visible rows are $F, K, S, I, U, A, L, H, C, P, N$. It has no coordinate for an unbounded Fourier series, host measure theory, quotient equality of functions, selected representatives, or a convergence theorem outside the displayed positive-kernel handoff.

Lean4 ✓ `BEDC.Derived.FejerTheoremUp.FejerTheoremTasteGate single carrier alignment`

Theorem 5178.2 (Fejer theorem NameCert obligations)**.** *For every accepted* $\mathsf{FejerTheorem}^{\uparrow}$ *carrier*

$$T = (F, K, S, I, U, A, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FejerTheorem}^{\uparrow}}$ *surface consists exactly of Fourier coefficient-window admission through F, positive Fejer-kernel admission through K, circle phase-sample exposure through S, integral normalization through I, uniform-limit handoff through U, theorem assertion exposure through A, non-escape ledger coverage through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5178.1. The rows F, K, S, I display the coefficient window, positive kernel, phase samples, and normalization data before the theorem assertion is available.

The theorem-facing rows are U, A, L. The uniform-limit row records the approximation boundary, the assertion row binds the Fourier window to the positive-kernel average, and the ledger row states the excluded host readings.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for Haar measure, host L^1 theory, quotient function equality, selected representatives, pointwise convergence, or ambient compactness, the listed rows exhaust the local NameCert surface. □

Theorem 5178.3 (Positive-kernel uniform handoff)**.** *Every approximation consumer of an accepted* $\mathsf{FejerTheorem}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{Fourier}^{\uparrow} \longmapsto \mathsf{FejerKernel}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Integral}^{\uparrow} \longmapsto \mathsf{UniformLimit}^{\uparrow}.$$

The positive-kernel row supplies only the finite triangular averaging and normalization data named by $\mathsf{FejerKernel}^{\uparrow}$*, and the theorem row exports only the uniform handoff U together with A, L, H, C, P, N. Thus the packet supplies a BEDC-native Fejer summability certificate without exporting Haar measure, host L^1 spaces, quotient function equality, selected representatives, pointwise convergence, or ambient compactness.*

Proof. Begin with an approximation request. The carrier first exposes F and K, so the request sees a finite Fourier coefficient window and a positive Fejer kernel row rather than an infinite Fourier series or host convolution object.

The request next reads S and I. These rows bind the averaging read to displayed circle phases and finite integral normalization cells.

Only then can the request inspect U and A. The uniform-limit handoff records the approximation boundary, while the theorem assertion row states that the positive-kernel average is the route by which the Fourier window is consumed.

Finally L, H, C, P, N block excluded host readings and replay the same finite packet. No listed consumer can bypass the positive-kernel route or turn it into Haar measure, quotient function equality, selected representatives, pointwise convergence, or ambient compactness. □

Falsifiable prediction: An accepted $\mathsf{FejerTheorem}^{\uparrow}$ packet is refuted if a Fourier approximation consumer reaches the $\mathsf{UniformLimit}^{\uparrow}$ handoff while bypassing the displayed $\mathsf{FejerKernel}^{\uparrow}$ positivity row, $\mathsf{S}^{1\uparrow}$ phase samples, or $\mathsf{Integral}^{\uparrow}$ normalization row.

Independence witness: fourier, fejer-kernel, sone, integral, uniformlimit: none is bijective to $\mathsf{FejerTheorem}^{\uparrow}$ because the packet jointly carries coefficient windows, positive triangular kernel data, circle phase samples, integral normalization, a uniform theorem handoff, a non-escape ledger, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FejerTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{FejerTheorem}^{\uparrow}$ packet is generated from FourierUp coefficient windows, FejerKernelUp positivity and triangular averaging rows, SOneUp phase samples, IntegralUp normalization rows, a UniformLimitUp theorem handoff, a non-escape ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FejerTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5178.1, Definition 5178.2, and Definition 5178.3 seed the finite theorem-level Fejer summability route over Fourier, positive-kernel, circle, integral, and uniform-limit rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FejerTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, Haar measure, host L^1 theory, quotient function equality, selected Fourier representatives, pointwise convergence, or ambient compactness.

Upgrade path: Obligation closure requires checked coefficient-window admission, positive-kernel admission, phase-sample exposure, integral normalization, uniform-limit handoff, theorem assertion exposure, ledger coverage, transport, replay, provenance, and local naming for BEDC.Derived.FejerTheoremUp.

Chapter 5179

A Concrete Naming Certificate for RegularCauchyTailEquivalence$^{\uparrow}$

Following Bishop–Bridges constructive Cauchy real analysis, the RegularCauchyTailEquivalence$^{\uparrow}$ packet records the equality-facing tail classifier by which two regular rational Cauchy names reach the Real$^{\uparrow}$ seal. It sits between the finite observation windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the setoid comparison surface of chapter 1820, and the terminal real-name boundary of chapter 38. The packet names the finite L10 Real and Completion handoff; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient completeness of Real$^{\uparrow}$.

Definition 5179.1 (Regular Cauchy tail equivalence carrier)**.** *A* RegularCauchyTailEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S_0, S_1, R_0, R_1, D, Q, A, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* StreamName$^{\uparrow}$ *finite-window rows through which the source names are inspected.*
2. R_0 *and* R_1 *are the corresponding* RegSeqRat$^{\uparrow}$ *readback rows exposed by those windows.*
3. D *is the* DyadicRatCore$^{\uparrow}$ *tail-tolerance ledger, recording the requested precision, the two regular radii, the common-tail threshold, and the remaining comparison slack.*
4. Q *is the* CauchyEquivalenceSetoid$^{\uparrow}$ *comparison row tying the two readbacks to the finite reflexive, symmetric, and transitive classifier surface.*
5. A *is the* Real$^{\uparrow}$ *seal row reached only after the two windows, two readbacks, tolerance ledger, and setoid comparison have been displayed.*
6. H *records componentwise* hsame *transport for windows, readbacks, tolerance ledger, setoid comparison, Real seal, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from an equality-facing Real or Completion request through* $S_0, S_1, R_0, R_1, D, Q, A, H, P, N$.
8. P *is the* Pkg *provenance over* RegularCauchyTailEquivalence$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, CauchyEquivalenceSetoid$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEquivalence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, A, H, C, P, N$. It has no coordinate for quotient classes, selected limits, countable choice, trichotomy, host real equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyTailEquivalenceUp.RegularCauchyTailEquivalenceTasteGate`
`single carrier alignment`

Theorem 5179.2 (Regular Cauchy tail equivalence NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ *carrier*

$$E = (S_0, S_1, R_0, R_1, D, Q, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEquivalence}^{\uparrow}}$ *surface consists exactly of two* $\mathsf{StreamName}^{\uparrow}$ *window admissions, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail-tolerance ledger, one* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison row, one* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5179.1. The accepted tuple exposes exactly the two window rows, two regular readbacks, dyadic tolerance ledger, setoid comparison, Real seal, transport, replay, provenance, and naming rows.

The source side is exhausted by S_0, S_1, R_0, R_1. Every equality-facing request must inspect the finite windows and the corresponding regular rational readbacks before any tail comparison is available.

The comparison side is exhausted by D and Q. The dyadic ledger stores the precision, radii, common-tail threshold, and slack, while the setoid row records the finite classifier comparison over the displayed readbacks.

The endpoint side is exhausted by A, H, C, P, N. These rows seal the comparison for $\mathsf{Real}^{\uparrow}$, transport it componentwise, replay the route, source it, and name it locally. Since the carrier contains no quotient, selected limit, choice, trichotomy, host equality, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5179.3 (Regular Cauchy tail equivalence Real handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ *packet must read the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto A$$

before it may consume equality at the $\mathsf{Real}^{\uparrow}$ *seal. Thus equality is exported only as a tail-tolerance classifier between displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, not as quotient stream equality or selected-limit equality.*

Proof. Begin with a Real or Completion equality request. By Definition 5179.1, the request first reaches the two finite window rows S_0, S_1 and their readbacks R_0, R_1.

The dyadic ledger D is then the only tolerance source. It records the common-tail threshold and the remaining slack before the setoid comparison row Q can be read.

Only downstream of Q does the seal row A become visible. The structural rows H, C, P, N preserve this route and add no host equality coordinate.

Therefore a Real or Completion consumer receives the displayed finite classifier route and nothing stronger. Quotient stream equality, selected limits, countable choice, trichotomy, host real equality, and ambient completeness remain outside the packet. □

Theorem 5179.4 (Regular Cauchy tail equivalence composition)**.** *Suppose accepted* $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ *packets witness tail equivalence from A to B and from B to C with compatible* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison rows, and* $\mathsf{Real}^{\uparrow}$ *seal rows. Then every Real or Completion consumer of the composed A-to-C comparison factors through the two displayed finite routes*

$$A \longmapsto B \qquad \textit{and} \qquad B \longmapsto C$$

before it may read the endpoint seal. The composed read exposes no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient Real completeness.

Proof. Project the two accepted carriers of Definition 5179.1. The first packet displays its source and middle windows, readbacks, dyadic tolerance ledger, setoid comparison, Real seal, transport, replay, provenance, and local naming rows; the second packet displays the same inventory for the middle and target rows.

Apply the Real handoff of Definition 5179.3 to the first packet and then to the second. Compatibility of the middle $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback lets the second handoff read the shared middle row only through the displayed finite carrier data.

Now concatenate the componentwise hsame transports and the displayed Cont routes. The setoid comparison rows supply the only comparison surface, while the dyadic ledgers keep the tolerance accounting visible on both sides of the middle readback.

The endpoint consumer therefore receives an A-to-C route only through the two displayed packets. Since neither carrier has a coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient Real completeness, the composed comparison exports none of those endpoints. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ read displays two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, one $\mathsf{DyadicRatCore}^{\uparrow}$ tail-tolerance ledger, one $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ comparison row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before Real or Completion equality is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-equivalence-setoid, real: $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ is not bijective to any listed sibling because it binds two window rows, two regular readbacks, one tail-tolerance ledger, one setoid comparison row, one Real seal, componentwise transport, replay, provenance, and local naming into a single L10 equality-facing carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailEquivalence}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readbacks, one DyadicRatCore tail-tolerance ledger, one CauchyEquivalenceSetoid comparison row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5179.1, Definition 5179.2, and Definition 5179.3 seed the finite L10 Real and Completion tail-equivalence classifier between RegSeqRat readbacks and Real equality.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked window admission, regular readback, dyadic tolerance accounting, setoid comparison, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailEquivalenceUp.

Chapter 5180

A Concrete Naming Certificate for HyperbolicObserverShell$^{\uparrow}$

The HyperbolicObserverShell$^{\uparrow}$ packet realizes the boundary-shell reading of chapter 9463 as one finite NameCert surface. It binds the boundary-shadow carrier of chapter 3813, the visual-boundary carrier of chapter 4181, the observer-expansion carrier of chapter 4167, the Busemann-boundary route of chapter 3732, and the metric readback of chapter 3474. The packet names only a finite observer-facing shell over displayed boundary rows; it is not a completed ideal boundary, a physical observer law, a global hyperbolic space, or a cosmological field model.

Definition 5180.1 (Hyperbolic observer shell carrier)**.** *A* HyperbolicObserverShell$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (B, V, E, U, M, T, F, H, C, P, N)$$

with the following displayed rows:

1. *B is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row that supplies the visible boundary-shadow face.*
2. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *row that reads the shadow as a visual boundary datum.*
3. *E is the* HyperbolicObserverExpansion$^{\uparrow}$ *row that records the focused observer-expansion route carried by the shell.*
4. *U is the* HyperbolicBusemannBoundary$^{\uparrow}$ *row attaching the shell to Busemann-style boundary comparison.*
5. *M is the* HyperbolicMetric$^{\uparrow}$ *readback row used by the shell when boundary shadows are compared.*
6. *T is the shell-coherence row, requiring the boundary-shadow, visual-boundary, observer-expansion, Busemann, and metric rows to refer to the same displayed finite source surface.*
7. *F is the TasteGate-facing row for this AI-origin packet: the Lean development in* `TasteGate.lean` *must provide* Nontrivial HyperbolicObserverShell$^{\uparrow}$ *and* FieldFaithful HyperbolicObserverShell$^{\uparrow}$ *instances preserving $B, V, E, U, M, T, H, C, P, N$ as distinct fields.*
8. *H records componentwise* hsame *transport for boundary shadow, visual boundary, observer expansion, Busemann row, metric readback, shell coherence, TasteGate-facing row, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a shell request through $B, V, E, U, M, T, F, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverShell}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShell}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, V, E, U, M, T, F, H, C, P, N$. It has no coordinate for completed ideal-boundary equality, physical observer dynamics, global hyperbolic geometry, a host compactification, or unlogged cosmological expansion.

Theorem 5180.2 (Hyperbolic observer shell NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverShell}^{\uparrow}$ *carrier*

$$O = (B, V, E, U, M, T, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShell}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, visual-boundary admission, observer-expansion admission, Busemann-boundary attachment, hyperbolic-metric readback, shell-coherence checking, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5180.1. The rows $B, V, E, U, M, T, F, H, C, P, N$ are the entire public packet.

The boundary side is carried by B, V, U, M. Boundary shadow is read first, visual boundary and Busemann data attach the boundary interpretation, and the metric row supplies the displayed comparison surface.

The observer side is carried by E and T. The observer-expansion row records the focused route, while the shell-coherence row requires that route to remain tied to the same displayed boundary and metric surface.

The AI-origin gate is carried by F. It records the obligation that the Lean `TasteGate.lean` development provide Nontrivial and FieldFaithful instances for the same field list rather than for a collapsed shadow token.

Finally H, C, P, N transport, replay, source, and name this finite route. Since no further coordinate is present, the certificate exports no completed ideal boundary, physical observer law, global hyperbolic geometry, host compactification, or unlogged cosmological expansion. □

Theorem 5180.3 (Hyperbolic observer shell boundary handoff)**.** *Every accepted* $\mathsf{HyperbolicObserverShell}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicBoundaryShadow}^{\uparrow},\ \mathsf{HyperbolicVisualBoundary}^{\uparrow},\ \mathsf{HyperbolicObserverExpansion}^{\uparrow},\ \mathsf{HyperbolicBusemannBoun}$$

before it can be consumed as an observer-facing boundary shell. The handoff preserves the displayed shell-coherence row and the TasteGate-facing field-faithfulness requirement.

Proof. Start with an accepted shell read. By the carrier definition, the read first displays B, V, E, U, M and cannot reach shell coherence without those rows.

The coherence row T then binds the boundary-shadow, visual-boundary, observer-expansion, Busemann, and metric data to one source surface. This prevents a consumer from mixing a boundary shadow from one packet with an observer-expansion row from another.

The TasteGate-facing row F keeps the AI-origin shell readable as a field-faithful carrier. The required Nontrivial and FieldFaithful instances must preserve the named rows rather than collapse them into an unstructured boundary token.

Transport, replay, provenance, and naming through H, C, P, N preserve the same route. The accepted handoff is therefore finite and cannot export completed boundary equality, physical dynamics, global geometry, host compactification, or unlogged cosmological expansion. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicObserverShell}^{\uparrow}$ read is refuted if a consumer obtains an observer-facing boundary shell without displaying $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, shell coherence, TasteGate-facing nontriviality and field-faithfulness requirements, componentwise transport, replay, provenance, and local naming rows.

Independence witness: hyperbolic-visual-boundary, hyperbolic-observer-expansion, hyperbolic-busemann-boundary, hyperbolic-metric: the visual-boundary packet reads boundary data without owning the observer-expansion route, the observer-expansion packet records focused shadow growth without the complete visual-boundary shell, the Busemann packet supplies boundary comparison without shell coherence, and the metric packet supplies readback without observer-facing boundary packaging. None is bijective to $\mathsf{HyperbolicObserverShell}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverShell}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicObserverShell}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicVisualBoundary rows, HyperbolicObserverExpansion rows, HyperbolicBusemannBoundary rows, HyperbolicMetric rows, shell coherence, TasteGate-facing field obligations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverShell}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5180.1, Definition 5180.2, and Definition 5180.3 seed the finite boundary-shadow, visual-boundary, and observer-expansion shell surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverShell}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed ideal-boundary equality, physical observer dynamics, global hyperbolic geometry, host compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked boundary-shadow admission, visual-boundary admission, observer-expansion admission, Busemann attachment, metric readback, shell coherence, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicObserverShellUp.

Chapter 5181

A Concrete Naming Certificate for $\mathsf{DiracPointMass}^{\uparrow}$

Following Dirac and Kolmogorov, the $\mathsf{DiracPointMass}^{\uparrow}$ packet records the standard point-mass probability measure as finite BEDC NameCert data. It sits over the measurable carrier of chapter 92, the probability-space surface of chapter 184, the measurable-read discipline of chapter 185, the distribution handoff of chapter 186, and the terminal scalar readback of chapter 38. The packet names the singleton mass route only; it does not select a host measure, import a powerset sigma algebra, choose representatives, or assert ambient real completeness.

Definition 5181.1 (Dirac point-mass carrier)**.** *A* $\mathsf{DiracPointMass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, p, E, U, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Measure}^{\uparrow}/\mathsf{ProbSpace}^{\uparrow}$ *carrier row, including the displayed measurable universe and probability-space admission data.*
2. *p is the point row, naming the selected support point only as a finite BEDC datum internal to the packet.*
3. *E is the event row, recording the finite event-membership read that decides whether the displayed event contains p.*
4. *U is the unit-mass ledger, assigning the* $\mathsf{Real}^{\uparrow}$ *value* 1 *to events whose displayed membership row contains p and* 0 *otherwise.*
5. *D is the* $\mathsf{Distribution}^{\uparrow}$ *and pushforward readback row, exposing how a* $\mathsf{RandomVar}^{\uparrow}$ *consumer reads the point mass through the image of p.*
6. *H records componentwise* hsame *transport for the measure source, point, event, unit ledger, distribution readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an event request through X, p, E, U, D and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{DiracPointMass}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DiracPointMass}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host measure selector, powerset event oracle, quotient representative, countable-choice schedule, selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.DiracPointMassUp.DiracPointMassTasteGate single carrier alignment`

Theorem 5181.2 (Dirac point-mass NameCert obligations)**.** *For every accepted* $\mathsf{DiracPointMass}^{\uparrow}$ *carrier* $Q = (X, p, E, U, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiracPointMass}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission, requiring the displayed* $\mathsf{Measure}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *source row* X *before any event read is public;*
2. *point-event exactness, requiring the event row* E *to expose whether the displayed event contains the point row* p*;*
3. *unit-mass ledger stability, requiring* U *to preserve the* $0/1$ $\mathsf{Real}^{\uparrow}$ *readback under componentwise* hsame *transport and displayed replay;*
4. *distribution handoff, requiring every* $\mathsf{RandomVar}^{\uparrow}$ *and* $\mathsf{Distribution}^{\uparrow}$ *consumer to read the image of* p *through* D *after* X, p, E, U *are displayed;*
5. *non-escape from host measure choice, requiring* H, C, P, N *to name only the displayed packet and to export no host measure selector, powerset event oracle, quotient representative, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

No local point-mass obligation is stored outside $X, p, E, U, D, H, C, P, N$*.*

Lean4 ✓ `BEDC.Derived.DiracPointMassUp.DiracPointMassCarrier namecert obligations`

Proof. Project the carrier of Definition 5181.1. The source row X admits the measure and probability-space surface, while the point row p and event row E are the only rows that can decide membership in the displayed event.

The unit ledger U is then forced by the point-event read. If E displays membership of p, the ledger reads the terminal $\mathsf{Real}^{\uparrow}$ unit value; otherwise it reads the terminal zero value. Componentwise transport through H and replay through C preserve the same event decision and the same scalar readback.

For a random-variable or distribution consumer, the row D is downstream of the same point and event data. It records the pushforward read only through the image of the displayed point, so the distribution handoff cannot bypass the finite point-mass ledger.

Finally P and N source and name precisely this packet. Since no row stores a host measure selector, powerset event oracle, quotient representative, selected limit, or ambient real-completeness principle, the five listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{DiracPointMass}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DiracPointMass}^{\uparrow}$ packet is generated from a displayed MeasureUp and ProbSpaceUp source, a point row, an event-membership row, a unit-mass RealUp ledger, a RandomVarUp and DistributionUp readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DiracPointMass}^{\uparrow}}$, seed)

Scope closed: Definition 5181.1 and Definition 5181.2 seed the finite point-mass packet over MeasureUp, ProbSpaceUp, RandomVarUp, DistributionUp, and RealUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DiracPointMass}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host measure selection, powerset sigma-algebra generation, quotient representatives, countable choice, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, point-event exactness, unit-mass ledger stability, distribution handoff, transport, replay, provenance, and local naming for BEDC.Derived.DiracPointMassUp.

Chapter 5182

A Concrete Naming Certificate for $\mathsf{DiracSequence}^{\uparrow}$

Following Dirac, Schwartz, and Bishop–Bridges, the $\mathsf{DiracSequence}^{\uparrow}$ packet records finite approximate-identity sequence data before a regularization or point-mass consumer may read a terminal real seal. It sits between the smooth-kernel packet of chapter 4198, the singleton-mass packet of chapter 5181, the terminal real-name surface of chapter 38, the dyadic tolerance ledger of chapter 291, and the continuous-map route of chapter 130. The packet names a finite schedule of approximate-identity rows; it does not import host integration, select a limiting delta object, identify quotient functions or distributions, invoke countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5182.1 (Dirac sequence carrier)**.** *A* $\mathsf{DiracSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, R, N, W, E, H, C, P, L)$$

with the following displayed rows:

1. *S is the scale-schedule row, recording the finite list of dyadic scale requests and the order in which downstream consumers may inspect them.*
2. *R is the support-radius decay row, recording the displayed radius bound attached to each scale entry and the finite locality budget consumed by the packet.*
3. *N is the normalization ledger, exposing unit-mass readback as* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons rather than as a host integral.*
4. *W is the convolution-window row, recording the source window, target window, translation request, scale request, and finite compatibility data read by a* $\mathsf{ContinuousMap}^{\uparrow}$ *or* $\mathsf{Mollifier}^{\uparrow}$ *consumer.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, N, W have been displayed.*
6. *H stores componentwise* hsame *transport for the schedule, support, normalization, convolution-window, real-seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from an approximate-identity request through S, R, N, W, E, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{DiracSequence}^{\uparrow}$, $\mathsf{Mollifier}^{\uparrow}$, $\mathsf{DiracPointMass}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{DiracSequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, N, W, E, H, C, P, L$. It has no coordinate for host integration, selected limits, quotient equality of functions or distributions, countable-choice schedules, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5182.2 (Dirac sequence NameCert obligations)**.** *For every accepted* $\mathsf{DiracSequence}^{\uparrow}$ *carrier* $D = (S, R, N, W, E, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{DiracSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed scale schedule, support-radius decay, normalization ledger, convolution-window row, Real seal, componentwise transport, displayed replay, provenance, and local naming. Every* $\mathsf{Mollifier}^{\uparrow}$*,* $\mathsf{DiracPointMass}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *consumer of the packet must read through these rows before treating the sequence as approximate-identity data.*

Proof. Project the carrier of Definition 5182.1. The analytic rows are exactly S, R, N, W, E, and the structural rows are exactly H, C, P, L.

The scale schedule S and support-radius decay row R expose the finite locality route. The normalization ledger N supplies the unit-mass boundary through displayed real and dyadic comparisons, so no host integration row is needed.

The convolution-window row W is the only row that a regularization or continuous-map consumer can use to relate a source window, target window, translation request, and scale request. The terminal seal E is downstream of that finite route rather than a selected limiting delta object.

Finally H, C, P, L transport, replay, source, and name only the displayed packet. Since the carrier has no coordinate for host integration, selected limits, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 5182.3 (Dirac sequence Real-seal handoff)**.** *Let* $D = (S, R, N, W, E, H, C, P, L)$ *be an accepted* $\mathsf{DiracSequence}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *readback exported by the packet factors through the finite route*

$$S \longmapsto R \longmapsto N \longmapsto W \longmapsto E \longmapsto H \longmapsto C \longmapsto P \longmapsto L.$$

The handoff provides scale-schedule, support-radius, normalization, convolution-window, transport, replay, provenance, and local naming data, but it exports no host integration operator, selected approximate identity limit, quotient function or distribution equality, countable-choice schedule, or ambient real-completeness theorem.

Proof. Begin with a consumer that asks for a real-valued normalization or regularization readback. Carrier admission first displays the scale schedule S and support-radius decay row R, so the request is confined to a finite approximate-identity window.

The normalization ledger N then supplies the real and dyadic comparisons needed for unit-mass readback. The convolution-window row W records the source and target windows, translation request, and scale request before the terminal seal E can be cited.

Transport through H and replay through C preserve the same displayed rows, while P and L source and name the local certificate. Thus the $\mathsf{Real}^{\uparrow}$ handoff factors through the listed finite route.

No carrier row stores host integration, a chosen limiting delta object, quotient equality, countable choice, or ambient completeness. Those objects therefore cannot be exported by the Real-seal handoff. □

Lean4 (statement) `BEDC.Derived.DiracSequenceUp`

Closure status for $\mathsf{NameCert}_{\mathsf{DiracSequence}^{\uparrow}}$.

Constructive story: A DiracSequence$^{\uparrow}$ packet is generated from a finite scale schedule, support-radius decay row, normalization ledger, convolution-window row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{DiracSequence}^{\uparrow}}$, seed)

Scope closed: Definition 5182.1, Definition 5182.2, and Definition 5182.3 seed the finite approximate-identity sequence surface over MollifierUp, DiracPointMassUp, ContinuousMapUp, RealUp, DyadicRatCoreUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{DiracSequence}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close host integration, selected approximate-identity limits, quotient equality of functions or distributions, countable choice, ambient Real$^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-radius decay, normalization ledger exactness, convolution-window routing, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.DiracSequenceUp.

Chapter 5183

A Concrete Naming Certificate for $\mathsf{LocatedCauchyBracket}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, 1985, the $\mathsf{LocatedCauchyBracket}^{\uparrow}$ packet records explicit finite bracket data for a regular Cauchy rational name before the L10 Real and Completion route reads the terminal real seal. It sits over the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the endpoint interval surface of chapter 394, and the located real boundary of **??**. The packet names only inspected brackets, tolerance rows, transport, replay, provenance, and local naming; it is not a quotient-real equality, selected limit, trichotomy, countable-choice schedule, or ambient completeness principle.

Definition 5183.1 (Located Cauchy bracket carrier)**.** *A* $\mathsf{LocatedCauchyBracket}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, D, W, R, I, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are lower and upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows for the inspected bracket.*
2. *D is the dyadic radius row certifying the requested bracket width.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window from which the source Cauchy data are read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W under the displayed radius ledger.*
5. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row whose endpoints are L and U and whose containment ledger covers the readback window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{LocatedReal}^{\uparrow}$ *seal row reached through the displayed bracket.*
7. *H records componentwise* hsame *transport for L, U, D, W, R, I, E and the local naming row.*
8. *C records displayed* Cont *replay from a bracket request through dyadic endpoints, stream observations, regular readback, located interval containment, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyBracket}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyBracket}^{\uparrow}}$ *row.*

The classifier compares accepted bracket packets only through $L, U, D, W, R, I, E, H, C, P, N$. It has no coordinate for a quotient stream, selected host limit, host real equality, trichotomy oracle, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyBracketUp.LocatedCauchyBracketTasteGate single carrier alignment`

Theorem 5183.2 (Located Cauchy bracket shrink)**.** *For every accepted* $\mathsf{LocatedCauchyBracket}^{\uparrow}$ *carrier* $B = (L, U, D, W, R, I, E, H, C, P, N)$ *and every displayed successor precision request carried by the same* $\mathsf{StreamName}^{\uparrow}$ *schedule, the bracket can be read through a successor dyadic radius row* D' *and a refined located interval row* I' *while preserving the original* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, terminal* $\mathsf{Real}^{\uparrow}$ *seal route,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The shrink exports no quotient stream, selected limit, host real equality, trichotomy, countable-choice selector, or ambient completeness row.*

Proof. Project the carrier rows. The only analytic data available to a successor precision read are the displayed endpoints L, U, the radius row D, the finite window W, the readback R, and the located interval row I.

Use the dyadic tolerance refinement from chapter 291 and keep the observation on the same $\mathsf{StreamName}^{\uparrow}$ schedule. The refined radius D' and interval row I' are therefore componentwise refinements of displayed rows, not new host data.

Finally replay the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ handoff through H, C, P, N. Since the carrier has no coordinate for a quotient stream, selected limit, host equality, trichotomy, choice, or ambient completeness, the refined bracket preserves the same finite seal route and exports nothing outside it. □

Theorem 5183.3 (Located Cauchy bracket Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{LocatedReal}^{\uparrow}$*, or Completion-facing consumer of an accepted* $\mathsf{LocatedCauchyBracket}^{\uparrow}$ *packet reaches the terminal real seal only through the displayed finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LocatedInterval}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route may consume dyadic bracket endpoints, a dyadic radius row, finite stream observations, regular rational readback, located interval containment, componentwise transport, replay, provenance, and local naming, but it cannot replace them by a quotient-real representative, selected limit, host interval primitive, trichotomy oracle, countable choice, or ambient completeness principle.

Proof. Begin with the displayed carrier of Definition 5183.1. The endpoint and radius rows expose the dyadic bracket before any stream or real-seal read is available.

The finite window W and readback R then supply the regular Cauchy observation surface. The located interval row I binds that readback between the displayed dyadic endpoints, so a Real-facing consumer cannot bypass the bracket by selecting a host representative.

The terminal seal row E is downstream of I and is replayed only through H, C, P, N. Thus the stated route exhausts the visible rows and excludes quotient representatives, selected limits, host interval primitives, trichotomy, choice, and ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyBracket}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedCauchyBracket}^{\uparrow}$ packet is generated from dyadic lower and upper endpoints, a dyadic radius row, a finite StreamNameUp observation window, RegSeqRatUp readback, a LocatedIntervalUp containment row, a LocatedRealUp and RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyBracket}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5183.1, Definition 5183.2, and Definition 5183.3 seed the finite located Cauchy bracket route for L10 Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedCauchyBracket}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket shrink, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCauchyBracketUp.

Chapter 5184

A Concrete Naming Certificate for $\mathsf{CompactMetricHyperspace}^{\uparrow}$

Following Hausdorff, Vietoris, and Kelley, the $\mathsf{CompactMetricHyperspace}^{\uparrow}$ packet records the recognized hyperspace of nonempty compact subsets of a compact metric BEDC source as finite Hausdorff-distance and Vietoris-window data. It refines the compact-subset carrier of chapter 1660 by requiring the source rows of chapter 129, chapter 57, chapter 127, and chapter 128 to be displayed before any hyperspace consumer reads compact-subset convergence, finite nets, or Hausdorff tolerances.

Definition 5184.1 (Compact metric hyperspace carrier)**.** *A* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, H, F, D, V, L, T, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, including the* $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedule, and* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-limit boundary carried by* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *H is the underlying* $\mathsf{Hyperspace}^{\uparrow}$ *compact-subset ledger over X, with compact-subset presentations inspected only through the displayed source rows.*
3. *F is the finite nonempty subset-window family: each window records a positive tolerance, a finite source-net list, and the compact-subset names covered at that tolerance.*
4. *D is the bidirectional Hausdorff-distance readback ledger for pairs of displayed compact-subset names, routed through the finite windows in F and the source metric row in X.*
5. *V is the Vietoris finite-net scope row, recording the finite hit-and-miss windows whose opens are represented by metric balls and finite complements read from X.*
6. *L is the Hausdorff-completion route row: a Cauchy family of compact-subset names is read through D and lands back in a compact-subset row of H by the total-bounded and complete source data.*
7. *T records componentwise* hsame *transport for X, H, F, D, V, L, P, N.*
8. *C records displayed* Cont *replay through*

$$X, H, F, D, V \longmapsto L.$$

9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricHyperspace}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricHyperspace}^{\uparrow}}$ *row.*

The carrier has no coordinate for host powersets, quotient compacta, open-cover compactness, selected limits, or choice-selected representatives.

Lean4 ✓ `BEDC.Derived.CompactMetricHyperspaceUp.CompactMetricHyperspaceTasteGate single carrier alignment`

Theorem 5184.2 (Compact metric hyperspace NameCert obligations)**.** *For every accepted* CompactMetricHyperspa *carrier*

$$K = (X, H, F, D, V, L, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactMetricHyperspace}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *compact metric source admission for X, requiring the displayed* $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows carried by* $\mathsf{CompactMetric}^{\uparrow}$*;*
2. *hyperspace-subset admission for H, requiring every compact-subset name to be read through chapter 1660;*
3. *finite Hausdorff-window admission for F and D, requiring two directed finite coverage reads before a Hausdorff tolerance is public;*
4. *Vietoris finite-net scope for V, requiring hit-and-miss data to be represented by finitely many source metric balls and finite complement checks;*
5. *completion-route stability for L, T, C, requiring Cauchy hyperspace reads to replay through the compact source finite nets and complete boundary;*
6. *provenance and name exhaustion for P, N, excluding host powersets, quotient compacta, open-cover compactness, selected limits, and choice-selected representatives.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5184.1. The source row X is the only place where metric distance, finite nets, and Cauchy-limit service enter the packet.

The hyperspace-subset obligation is exactly the row H. The finite-window and Hausdorff obligations are exactly F and D, so every compact-subset comparison uses displayed finite source data rather than an ambient set of compacta.

The Vietoris obligation is the row V. Its hit-and-miss windows are finite metric-ball inspections over X, hence they remain synchronized with the Hausdorff ledger instead of importing open-cover compactness.

Finally L, T, C, P, N close the route under completion readback, componentwise transport, replay, provenance, and local naming. Since the carrier has no further coordinates, the six listed rows exhaust the local naming surface. □

Theorem 5184.3 (Compact metric hyperspace Hausdorff completion route)**.** *Every accepted* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *carrier routes Hausdorff-Cauchy compact-subset data through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *source finite nets and* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary. A Cauchy request in the hyperspace reads only the rows H, F, D, L, T, C, P, N, and the resulting compact-subset landing row is again a displayed* $\mathsf{Hyperspace}^{\uparrow}$ *name over the same source X.*

Proof. Begin with the Hausdorff ledger D. Each directed coverage read cites a finite source-net window from F and a source distance witness from X.

The Cauchy request is then consumed by L. Total boundedness supplies finite source windows at each requested tolerance, and completeness supplies the source boundary needed to keep the compact-subset landing row inside the same compact metric packet.

Transport and replay are handled by T and C. These rows carry the compact metric source, hyperspace rows, finite windows, Hausdorff ledger, and landing row together, so the route cannot leave the displayed carrier.

The provenance and local name rows P, N certify that no selected limit object, quotient compactum, or host powerset is exported. The landing row is therefore a $\mathsf{Hyperspace}^{\uparrow}$ name over X rather than an external compactum. □

Theorem 5184.4 (Compact metric hyperspace Vietoris finite-net scope)**.** *The Vietoris reading of an accepted* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *carrier is confined to the finite hit-and-miss windows in* V*. Each hit row names a finite family of source metric balls meeting the displayed compact-subset row, each miss row names a finite complement check against the same source net, and the read is stable under the componentwise transport* T *and replay* C*.*

Proof. Project the row V. Its data are finite hit-and-miss windows, and each window is represented by source metric balls and finite complement checks over X.

The row H supplies the compact-subset names being inspected. The rows F and D synchronize those names with Hausdorff finite-net reads, so the Vietoris view cannot inspect a larger host topology.

Stability follows from T and C. Componentwise transport preserves the source, compact-subset, finite-window, Hausdorff, and Vietoris rows, while replay repeats the same displayed inspection path.

Thus the Vietoris scope is finite at every public read and contains no open-cover compactness coordinate, selected representative, quotient compactum, or host powerset. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricHyperspace}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactMetricHyperspace}^{\uparrow}$ packet is generated from a CompactMetricUp source, a HyperspaceUp compact-subset ledger, finite nonempty subset windows, Hausdorff-distance readback, Vietoris finite-net scope, a Hausdorff-completion route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricHyperspace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5184.1, Definition 5184.2, Definition 5184.3, and Definition 5184.4 seed the compact metric Hausdorff/Vietoris hyperspace surface over compact metric BEDC packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactMetricHyperspace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets, quotient compacta, open-cover compactness, selected limits, choice-selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite Hausdorff-window admission, Vietoris finite-net scope, Hausdorff-completion route stability, and provenance/name exhaustion for BEDC.Derived.CompactMetricHyperspaceUp.

Chapter 5185

A Concrete Naming Certificate for $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, 1985, the $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$ packet records explicit finite ε-net extraction for a located closed interval before the L10 Real and Completion route consumes compactness-like data. It sits after the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the interval finite-net surface of chapter 1808, the total-bounded interval surface of chapter 2021, the compact real-interval packet of chapter 5553, and the located Cauchy bracket route of chapter 5183. The packet names only displayed endpoints, tolerance requests, finite net rows, stream windows, regular readback, Real seals, transport, replay, provenance, and local naming; it is not an open-cover compactness theorem, host interval primitive, quotient-real equality, trichotomy oracle, or ambient completeness principle.

Definition 5185.1 (Bishop interval net extraction carrier)**.** *A* $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, \varepsilon, D, G, W, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the located closed interval endpoint rows, read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *rational brackets and the surrounding* $\mathsf{Real}^{\uparrow}$ *seals.*
2. *ε is the positive tolerance request row, stored as a dyadic radius budget rather than as a host real inequality.*
3. *D is the dyadic subdivision ledger, recording the integer step count, endpoint refinements, successor grid positions, and comparison witnesses used to keep adjacent grid rows within the requested tolerance.*
4. *G is the finite grid-net row, a displayed list of dyadic points between A and B whose coverage ledger is inherited by* $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$ *and* $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$ *consumers.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation-window row used to inspect the endpoint and grid requests without selecting an infinite stream representative.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and dyadic subdivision budget.*

7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for endpoint, grid-point, and coverage reads, reached only after* $A, B, \varepsilon, D, G, W, R$ *have been displayed.*
8. *F is the compact-interval handoff row, routing the extracted finite net to* $\mathsf{CompactRealInterval}^{\uparrow}$ *through the interval finite-net and total-bounded sibling packets.*
9. *H records componentwise* hsame *transport for* $A, B, \varepsilon, D, G, W, R, E, F$ *and the local naming row.*
10. *C records displayed* Cont *replay from a tolerance request through endpoint brackets, dyadic subdivision, grid formation, finite stream windows, regular readback, Real seals, and the compact-interval handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$, $\mathsf{CompactRealInterval}^{\uparrow}$, LocatedCauchyBrack BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalNetExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted interval-net packets only through $A, B, \varepsilon, D, G, W, R, E, F, H, C, P, N$. It has no coordinate for an open-cover subcover, a host interval primitive, quotient real equality, trichotomy, countable choice, a selected limiting point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalNetExtractionUp.BishopIntervalNetExtractionTasteGate single carrier alignment`

Theorem 5185.2 (Bishop interval net extraction NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$ *carrier*

$$I = (A, B, \varepsilon, D, G, W, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalNetExtraction}^{\uparrow}}$ *surface consists exactly of located endpoint admission, positive tolerance admission, dyadic subdivision formation, finite grid-net coverage, finite* $\mathsf{StreamName}^{\uparrow}$ *inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff, compact-interval routing through the located finite-net and total-bounded sibling packets, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier rows. The endpoints A, B and tolerance row ε are the only analytic inputs, and both are read through displayed dyadic and Real-seal data rather than through a host interval primitive.

The subdivision ledger D forms a finite dyadic grid by repeated successor steps over the displayed endpoint brackets. The grid row G records the resulting finite list and its interval coverage ledger, so every coverage read is finite before any compactness-facing handoff is available.

The readback rows W and R keep endpoint and grid inspection inside finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ rational observations. The terminal seal row E is downstream of those rows, so Real-facing consumers cannot bypass the dyadic grid by selecting quotient representatives or host limits.

Finally F, H, C, P, N route the same finite net to $\mathsf{LocatedClosedIntervalFiniteNet}^{\uparrow}$, $\mathsf{ClosedIntervalTotalBounded}^{\uparrow}$, and $\mathsf{CompactRealInterval}^{\uparrow}$ consumers while preserving transport, replay, provenance, and local naming. Since the carrier has no coordinate for open-cover compactness, host interval primitives, quotient real equality, trichotomy, countable choice, selected limits, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalNetExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalNetExtraction}^{\uparrow}$ packet is generated from located closed interval endpoints, a positive dyadic tolerance request, a dyadic subdivision ledger, finite grid-net coverage, finite StreamName windows, RegSeqRat readback, RealUp seals, compact-interval handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalNetExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5185.1 and Definition 5185.2 seed the explicit finite interval-net extraction packet for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalNetExtraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host real equality, quotient interval equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, positive tolerance admission, dyadic subdivision formation, finite grid coverage, finite-window readback, RealUp seal handoff, compact-interval routing, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalNetExtractionUp.

Chapter 5186

A Concrete Naming Certificate for $\mathsf{SigmaAlgebra}^{\uparrow}$

Following Kolmogorov and Caratheodory measure theory, the $\mathsf{SigmaAlgebra}^{\uparrow}$ packet records the measurable-event closure surface that measure and probability chapters read before assigning values to events. It sits between the displayed event-family rows of chapter 92, the probability carrier of chapter 184, the measurable-preimage discipline of chapter 185, and the pushforward readback of chapter 186. The packet names finite BEDC rows for event admission, complement, and countable-union schedules; it is not a host powerset, completed measurable universe, quotient event algebra, countable-choice selector, or ambient measure object.

Definition 5186.1 (Sigma-algebra carrier)**.** *A* $\mathsf{SigmaAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (E, Q, Z, K, U, H, C, P, N)$$

with the following displayed rows:

1. *E is the event-family row, listing the measurable-event entries visible to downstream* $\mathsf{Measure}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *consumers.*
2. *Q is the classifier row, comparing displayed events only through the local event-family equivalence and its transported* hsame *witnesses.*
3. *Z is the empty-event row, recording the admitted empty event and its classifier readback.*
4. *K is the complement row, assigning to every displayed event a displayed complement event over the same family row.*
5. *U is the countable-union schedule, naming a displayed union event for every displayed countable event schedule that the packet admits.*
6. *H records componentwise* hsame *transport for the event family, classifier, empty-event, complement, countable-union, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an event request through E, Q, Z, K, U and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{SigmaAlgebra}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SigmaAlgebra}^{\uparrow}}$ *row.*

The carrier has no coordinate for host powersets, completed sigma-algebras, quotient event equality, countable-choice selection, ambient measures, or external measurable-space instances.

Lean4 ✓ `BEDC.Derived.SigmaAlgebraUp.SigmaAlgebraTasteGate single carrier alignment`

Theorem 5186.2 (Sigma-algebra NameCert obligations)**.** *For every accepted* $\mathsf{SigmaAlgebra}^{\uparrow}$ *carrier* $S = (E, Q, Z, K, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SigmaAlgebra}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *event-family admission, requiring* E *before any measurable-event read is public;*
2. *classifier stability, requiring* Q *and componentwise* hsame *transport through* H *to preserve displayed event comparison;*
3. *empty-event admission, requiring* Z *to identify the empty event inside the same event family;*
4. *complement and countable-union closure, requiring* K *and* U *to supply displayed closure rows before a downstream* $\mathsf{Measure}^{\uparrow}$ *or* $\mathsf{ProbSpace}^{\uparrow}$ *consumer may cite sigma-algebra structure;*
5. *provenance and local naming, requiring* C, P, N *to replay, source, and name only the displayed packet, with no host powerset, completed sigma-algebra, quotient event algebra, countable-choice selector, ambient measure, or external measurable-space coordinate.*

No local sigma-algebra obligation is stored outside $E, Q, Z, K, U, H, C, P, N$*.*

Proof. Project the carrier of Definition 5186.1. The event-family row E is the only source of measurable events, and the classifier row Q is the only public comparison between them.

The empty-event row Z, complement row K, and countable-union schedule U supply the closure operations in the same displayed family. Thus a closure request cannot leave the carrier to inspect a host powerset or completed measurable universe.

Transport and replay are componentwise through H and C. These rows preserve the event-family, classifier, empty-event, complement, and countable-union data without adding a second equality or a selected representative.

Finally P and N source and name precisely this packet. Since the forbidden host and quotient coordinates are absent, the five listed rows exhaust the local NameCert surface. □

Theorem 5186.3 (Sigma-algebra measure handoff)**.** *Every* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*, or* $\mathsf{Distribution}^{\uparrow}$ *consumer that cites measurable-event structure from an accepted* $\mathsf{SigmaAlgebra}^{\uparrow}$ *carrier* $S = (E, Q, Z, K, U, H, C, P, N)$ *receives only the displayed event-family row* E*, classifier row* Q*, empty-event row* Z*, complement row* K*, countable-union schedule* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local* NameCert *naming through* N*. The handoff exports no host powerset, completed sigma-algebra, quotient event algebra, countable-choice selector, ambient measure, or external measurable-space instance.*

Proof. Read Definition 5186.2. A downstream consumer must first consume the event-family, classifier, empty-event, complement, and countable-union rows before treating the packet as sigma-algebra structure.

For $\mathsf{Measure}^{\uparrow}$ and $\mathsf{ProbSpace}^{\uparrow}$, these rows provide the measurable-event surface before any value or probability row can be read. For $\mathsf{RandomVar}^{\uparrow}$ and $\mathsf{Distribution}^{\uparrow}$, the same rows are the source and target event families through which measurable preimages and pushforward events are routed.

The rows H, C, P, N preserve, replay, source, and name the handoff. Because the carrier has no host powerset, completed sigma-algebra, quotient event algebra, countable-choice selector, ambient measure, or external measurable-space coordinate, the listed rows are the whole exported interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{SigmaAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{SigmaAlgebra}^{\uparrow}$ packet is generated from a displayed event-family row, local event classifier, empty-event row, complement row, countable-union schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SigmaAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5186.1, Definition 5186.2, and Definition 5186.3 seed the finite measurable-event closure surface consumed by MeasureUp, ProbSpaceUp, RandomVarUp, and DistributionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SigmaAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, completed sigma-algebras, quotient event equality, countable-choice selection, ambient measure objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked event-family admission, classifier stability, empty-event admission, complement closure, countable-union closure, transport, replay, provenance, and local naming for BEDC.Derived.SigmaAlgebraUp.

Chapter 5187

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy reals and explicit moduli, the $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$ packet records the finite route by which a regular Cauchy name may be read through coarser monotone modulus windows before it reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback and a terminal $\mathsf{Real}^{\uparrow}$ seal. It sits beside the stream-name surface of chapter 126, the dyadic rational core of chapter 291, the regular rational sequence surface of chapter 301, the Cauchy modulus route of chapter 292, and the real-name seal of chapter 38. The packet names only a displayed compression ledger for finite tolerance windows; it does not assert a selected optimal modulus, a quotient of Cauchy names, countable choice, or ambient real completeness.

Definition 5187.1 (Regular Cauchy modulus-compression carrier)**.** *A* $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular-Cauchy window row, recording the finite prefix of the original name together with the tolerance requests visible to the consumer.*
2. *Q is the monotone compression schedule, assigning each requested tolerance window a displayed coarser window and recording the monotonicity checks between consecutive requests.*
3. *D is the dyadic tolerance ledger over* $\mathsf{DyadicRatCore}^{\uparrow}$*, keeping every compressed bound as a finite dyadic rational comparison.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational sequence obtained by reading the compressed windows through S, Q, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, recording that the readback surface is consumed only through the displayed real-name endpoint.*
6. *H records componentwise* hsame *transport for S, Q, D, R, E, C, P, N.*
7. *C records displayed* Cont *replay from a tolerance request through the compression schedule, dyadic ledger, regular-sequence readback, and real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCompression}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for an optimal modulus, a host limit, quotient equality of Cauchy names, countable choice, a completed real-space structure, or a fuel-erased stream evaluator.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusCompressionUp.RegularCauchyModulusCompressionTasteGate`
`single carrier alignment`

Theorem 5187.2 (Regular Cauchy modulus-compression NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$ *carrier*

$$M = (S, Q, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCompression}^{\uparrow}}$ *surface consists exactly of carrier admission for the source regular-Cauchy windows, monotone compression schedule, dyadic tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Any compressed tolerance read must expose S, Q, D before R, E, H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5187.1. The visible rows are precisely the source windows, compression schedule, dyadic ledger, readback row, real seal, transport, replay, provenance, and local name.

The finite compression boundary is read through S, Q, D. The source row supplies the original regular-Cauchy requests, the schedule row records which coarser windows answer them, and the dyadic row keeps the tolerance comparisons inside $\mathsf{DyadicRatCore}^{\uparrow}$.

The remaining rows R, E, H, C, P, N only read back, seal, transport, replay, package, and name that displayed route. Since the carrier has no coordinate for an optimal modulus, host limit, quotient equality, countable choice, completed real-space structure, or stream evaluator, these rows exhaust the NameCert obligations. □

Theorem 5187.3 (Regular Cauchy modulus-compression real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Concretely, the consumer first reads the source regular-Cauchy windows S, then the monotone compression schedule Q, then the dyadic tolerance ledger D, then the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal E. It cannot inspect an uncompressed host sequence, select a least modulus, identify Cauchy names by quotient equality, or bypass the displayed dyadic and regular-sequence rows.*

Proof. Begin with an accepted carrier $M = (S, Q, D, R, E, H, C, P, N)$. The carrier definition places every tolerance request first in the source-window row S and then in the schedule row Q, so a real-facing consumer has no direct access to an ambient sequence.

Next use the dyadic ledger D. It is the only row that compares the compressed windows as finite tolerance data, and it records those comparisons inside $\mathsf{DyadicRatCore}^{\uparrow}$ before any regular-sequence readback is available.

The readback row R repacks the displayed compressed windows as a $\mathsf{RegSeqRat}^{\uparrow}$ surface, and the seal row E is the only $\mathsf{Real}^{\uparrow}$ endpoint attached to the packet. The transport and replay rows H, C preserve this order componentwise, while P, N only record provenance and naming.

Thus every real-facing read follows the displayed stream, dyadic, regular-sequence, and real-seal route, with no coordinate for a least modulus, quotient Cauchy equality, host sequence, or bypass around the finite ledger. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCompression}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusCompression}^{\uparrow}$ packet is generated from source regular-Cauchy windows, a monotone compression schedule, a dyadic tolerance ledger, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCompression}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5187.1, Definition 5187.2, and Definition 5187.3 seed the finite modulus-compression route from regular Cauchy windows to RegSeqRat and RealUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusCompression}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close optimal modulus selection, quotient equality of Cauchy names, countable choice, host real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule monotonicity exposure, dyadic tolerance confinement, RegSeqRat readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMod

Chapter 5188

A Concrete Naming Certificate for $\mathsf{QuotientNormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges, the $\mathsf{QuotientNormedSpace}^{\uparrow}$ packet records the finite NameCert surface for quotient normed spaces by a closed linear subspace. It sits downstream from the $\mathsf{NormedSpace}^{\uparrow}$ carrier of chapter 1950, the $\mathsf{Met}^{\uparrow}$ distance readback of chapter 57, the completion boundary of chapter 95, and bounded-operator consumers such as chapter 1644. The packet names quotient norm and induced metric data; it does not export a selected representative, a host quotient object, an ambient Banach completion, or a choice principle.

Definition 5188.1 (Quotient normed-space carrier)**.** *A* $\mathsf{QuotientNormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (E, S, Q, N, M, H, C, P, L)$$

with the following displayed rows:

1. *E is the source* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing only the displayed VecSpace, Real, norm, induced metric, completion-facing, transport, replay, provenance, and local naming rows of the source packet.*
2. *S is the closed-subspace row, carrying membership, linear closure, zero inclusion, and norm-closedness evidence as classifier rows over the source normed-space data.*
3. *Q is the coset-classifier row, identifying histories exactly when their displayed difference is certified by the closed-subspace row S.*
4. *N is the quotient norm row, reading the infimum-style quotient norm only through source norm observations, coset witnesses, and rational tolerance ledgers displayed in E, S, Q.*
5. *M is the induced metric row, whose distance between quotient classes is the quotient norm read of their displayed difference.*
6. *H records componentwise* hsame *transport for E, S, Q, N, M and the support rows.*
7. *C records displayed* Cont *replay for quotient-classifier, quotient-norm, induced-metric, and downstream bounded-operator requests.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientNormedSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{BoundedLinearOperator}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *L is the local* $\mathsf{NameCert}_{\mathsf{QuotientNormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, S, Q, N, M, H, C, P, L$. It has no coordinate for representative choice, host quotient equality, selected completion limits, basis data, or an external Banach carrier.

Theorem 5188.2 (Quotient normed-space NameCert obligations)**.** *For every accepted* $\mathsf{QuotientNormedSpace}^{\uparrow}$ *carrier*

$$Z = (E, S, Q, N, M, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{QuotientNormedSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *source admission, requiring the quotient packet to read its vector operations, Real norm values, induced metric data, and completion-facing requests through the* $\mathsf{NormedSpace}^{\uparrow}$ *source row E;*
2. *closed-subspace stability, requiring membership, linear closure, zero inclusion, and norm-closedness to be witnessed by the subspace row S over the same source classifier;*
3. *coset-classifier exactness, requiring the row Q to compare quotient histories only by displayed differences certified in S;*
4. *quotient-norm and metric handoff, requiring N to read quotient norms through source norm observations and rational tolerance ledgers, and requiring M to read distances as quotient norm reads;*
5. *support replay, requiring* hsame, Cont, Pkg, *and local* NameCert *support through H, C, P, L for bounded-operator, completion, Banach, and Hilbert consumers without adding representative choice, host quotient equality, selected limits, or ambient carrier data.*

No quotient normed-space obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.QuotientNormedSpaceUp.QuotientNormedSpaceTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5188.1. The source row E, closed-subspace row S, coset classifier Q, quotient norm N, induced metric M, and support rows H, C, P, L are the only visible coordinates.

The first two obligations follow by reading E and S before any quotient request. The source supplies the normed-space route, while the subspace row supplies membership, linear closure, zero inclusion, and norm-closedness over that same route.

The coset obligation is then exact along Q. Two quotient histories can be compared only by a displayed difference certified in S, so no representative selector or host quotient equality can enter the comparison.

The norm and metric obligations are forced by N and M. A quotient norm read uses source norm observations and rational tolerance ledgers, and the metric row reads distance as the quotient norm of a displayed difference.

Finally H, C, P, L replay and name the same finite route for completion and bounded-operator consumers. Since the carrier contains no selected representative, host quotient object, selected limit, basis datum, or ambient Banach coordinate, the five listed rows exhaust the NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientNormedSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{QuotientNormedSpace}^{\uparrow}$ packet is generated from a source normed-space row, a closed-subspace row, a coset-classifier row, a quotient-norm row, an induced metric row, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientNormedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5188.1 and Definition 5188.2 seed the quotient normed-space packet over NormedSpace, MetricSpace, Completion, and bounded-operator consumer rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{QuotientNormedSpace}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach quotient completeness, representative selection, quotient completion, bounded-operator factorization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, closed-subspace stability, coset exactness, quotient-norm handoff, induced metric handoff, replay, provenance, and local naming for the same finite packet.

Chapter 5189

A Concrete Naming Certificate for $\mathsf{CompactNetRadiusChain}^{\uparrow}$

Following the Heine–Cantor compact finite-net tradition, the $\mathsf{CompactNetRadiusChain}^{\uparrow}$ packet records the finite NameCert surface that turns compact metric finite nets and pointwise continuity radii into a uniform-modulus handoff. It reads compact centers from $\mathsf{CompactMetric}^{\uparrow}$ as in chapter 129, pointwise graph radii from $\mathsf{ContinuousMap}^{\uparrow}$ as in chapter 130, tolerance-only output through $\mathsf{UniformModulus}^{\uparrow}$ as in chapter 369, and rational comparison through $\mathsf{DyadicRatCore}^{\uparrow}$ and $\mathsf{Rat}^{\uparrow}$ as in chapter 291 and chapter 37. The packet is a finite radius-chain carrier; it is not an external Lebesgue-number selector, not a sequential compactness proof, and not a choice-based modulus.

Definition 5189.1 (Compact net radius-chain carrier)**.** *A* $\mathsf{CompactNetRadiusChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, A, B, D, U, H, C, P, L)$$

with the following displayed rows:

1. *K is the compact-net center row, carrying a nonempty* $\mathsf{ProbeBundle}$ *of center histories and coverage witnesses over a compact metric source.*
2. *F is the continuous-map row, exposing pointwise radius data for the displayed graph through the same source metric and target metric rows.*
3. *A is the pointwise radius row, assigning each bundled center a positive rational or dyadic radius history accepted by the continuous-map modulus ledger.*
4. *B is the rational lower-bound fold row, folding the finite radius family through* $\mathsf{Rat}^{\uparrow}$ *order readback and preserving a classifier-stable positive lower bound.*
5. *D is the dyadic tolerance ledger, recording halving, thirding, and comparison requests through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and the surrounding* $\mathsf{Rat}^{\uparrow}$ *seal.*
6. *U is the uniform-modulus handoff row, reading a tolerance-only precision history from B, D and routing it to* $\mathsf{UniformModulus}^{\uparrow}$ *consumers.*
7. *H records componentwise* hsame *transport for K, F, A, B, D, U and the support rows.*
8. *C records displayed* Cont *replay from compact coverage, pointwise modulus reads, rational folds, dyadic tolerance comparisons, and uniform-modulus handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactNetRadiusChain}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactNetRadiusChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, A, B, D, U, H, C, P, L$. It has no coordinate for a host finite set, an external minimum selector, a sequential compactness branch, or a hidden modulus choice.

Theorem 5189.2 (Compact net radius-chain NameCert obligations)**.** *For every accepted* $\mathsf{CompactNetRadiusChain}^{\uparrow}$ *carrier*

$$R = (K, F, A, B, D, U, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{CompactNetRadiusChain}^{\uparrow}}$ *surface consists exactly of compact-net center admission, continuous-map pointwise radius admission, rational lower-bound folding, dyadic tolerance bookkeeping, uniform-modulus handoff, componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CompactNetRadiusChainUp.CompactNetRadiusChainTasteGate single carrier alignment`

Proof. Project the carrier. The compact centers, continuous-map radii, rational fold, dyadic tolerance ledger, uniform-modulus handoff, transport, replay, provenance, and naming rows are the only displayed coordinates.

The compact-net row K supplies a finite ProbeBundle of centers and coverage witnesses. The continuous-map row F and pointwise radius row A attach a local radius ledger to each bundled center over the same metric source.

The fold row B reads the finite family through the $\mathsf{Rat}^{\uparrow}$ order, while the dyadic ledger D supplies the half-radius and tolerance comparisons needed by the metric triangle estimate. These rows produce a positive lower-bound history rather than an external minimum.

The handoff row U packages that lower bound as a tolerance-only precision for $\mathsf{UniformModulus}^{\uparrow}$ consumers. Transport, replay, provenance, and local naming through H, C, P, L preserve the same finite route, and no host finite set, sequential compactness branch, or hidden modulus selector remains available. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactNetRadiusChain}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactNetRadiusChain}^{\uparrow}$ packet is generated from compact-net centers, pointwise continuous-map radii, a rational lower-bound fold, a dyadic tolerance ledger, a uniform-modulus handoff row, componentwise transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactNetRadiusChain}^{\uparrow}}$, seed)

Scope closed: Definition 5189.1 and Definition 5189.2 seed the compact finite-net radius-chain packet for uniform-modulus consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactNetRadiusChain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the full Heine–Cantor theorem, Lean verification, bridge checking, sequential compactness, or a host modulus selector.

Upgrade path: Obligation closure requires checked compact-center admission, pointwise-radius admission, rational lower-bound folding, dyadic tolerance bookkeeping, uniform-modulus handoff, transport, replay, provenance, and local naming for the same finite packet.

Chapter 5190

A Concrete Naming Certificate for $\mathsf{DyadicBracketApproximation}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicBracketApproximation}^{\uparrow}$ packet records the finite lower-upper dyadic bracket interface by which the L10 Real and Completion route reads a regular rational name at a requested precision before the terminal real seal is consumed. It sits over the dyadic arithmetic surface of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the equality boundary of chapter 1820, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet gives later density, located-order, and completion consumers a displayed bracket row rather than a prose appeal to approximation.

Definition 5190.1 (DyadicBracketApproximation carrier)**.** *A* $\mathsf{DyadicBracketApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (L, U, Q, I, S, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket row. It stores the inspected lower dyadic endpoint, its source observation, and the order evidence that the endpoint lies below the displayed regular read.*
2. *U is the upper* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket row. It stores the inspected upper dyadic endpoint, its source observation, and the order evidence that the displayed regular read lies below the endpoint.*
3. *Q is the unary precision row. It records the requested precision as a finite unary budget and the dyadic radius derived from that budget.*
4. *I is the inclusion witness row. It records that the displayed regular readback lies inside the bracket $[L, U]$ at the precision carried by Q.*
5. *S is the shrink witness row. It records the successor-precision bracket whose lower and upper dyadic endpoints refine L and U without changing the source name.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation window from which the bracket read is taken.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from W and the dyadic precision budget.*
8. *E is the* $\mathsf{CauchySequenceEquivalence}^{\uparrow}$ *and terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the bracket, window, and readback rows have been displayed.*

9. *H records componentwise* hsame *transport for L, U, Q, I, S, W, R, E and the local naming row.*
10. *C records displayed* Cont *replay from a precision request through lower bracket, upper bracket, inclusion, shrink, finite window, regular readback, equivalence boundary, and real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicBracketApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicBracketApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, Q, I, S, W, R, E, H, C, P, N$. It has no coordinate for a selected real representative, quotient stream equality, host located-order oracle, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicBracketApproximationUp.DyadicBracketApproximationTasteGate single carrier alignment`

Theorem 5190.2 (DyadicBracketApproximation NameCert obligations)**.** *For every accepted* DyadicBracketApprox *carrier*

$$B = (L, U, Q, I, S, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicBracketApproximation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the lower dyadic bracket, upper dyadic bracket, unary precision, inclusion witness, shrink witness, finite stream window, regular rational readback, equivalence and real seal, transport, replay, provenance, and local naming;*
2. *lower-upper discipline, requiring L and U to be read before any bracket consumer may inspect a regular or real-facing row;*
3. *precision discipline, requiring Q to expose a unary precision budget and the corresponding dyadic radius before the inclusion or shrink rows are available;*
4. *inclusion discipline, requiring I to bind the displayed regular readback between the lower and upper dyadic rows at the precision carried by Q;*
5. *shrink discipline, requiring S to refine the same source route at successor precision without replacing it by a selected limit or a quotient representative;*
6. *readback discipline, requiring W and R to supply the finite StreamName and RegSeqRat route before the equivalence and* $\mathsf{Real}^{\uparrow}$ *seal row E is read;*
7. *seal nonescape, requiring E, H, C, P, N to export only the displayed bracket route and no selected real representative, quotient stream equality, host located-order oracle, countable choice, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

No DyadicBracketApproximation obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5190.1. The accepted packet displays exactly the lower and upper dyadic brackets, unary precision, inclusion and shrink witnesses, finite window, regular readback, equivalence and real seal, and structural rows.

The lower-upper and precision disciplines are the first stage of the route. The bracket endpoints L, U and precision budget Q must be visible before any inclusion, shrink, regular-readback, or Real-facing row is available.

The inclusion and shrink disciplines are the constructive approximation stage. The row I pins the displayed readback inside the dyadic bracket at the current budget, while S records a successor bracket over the same source route rather than a chosen limit.

Finally W, R, E, H, C, P, N replay, seal, transport, package, and name the same route. Since selected representatives, quotient stream equality, host located-order oracles, countable choice, trichotomy, and ambient Real completeness are absent from the carrier, no local obligation can export them. □

Theorem 5190.3 (Dyadic bracket Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchySequenceEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or located-order consumer of an accepted* $\mathsf{DyadicBracketApproximation}^{\uparrow}$ *carrier factors through the displayed route*

$$L, U, Q \longmapsto I, S \longmapsto W, R \longmapsto E, H, C, P, N.$$

The consumer may read lower and upper dyadic brackets, unary precision, inclusion, shrink, finite stream observations, regular rational readback, equivalence sealing, transport, replay, provenance, and local naming. It cannot replace that route by a selected real representative, quotient stream equality, host located-order oracle, countable choice, trichotomy, or ambient completeness principle.

Proof. Start with the accepted carrier $B = (L, U, Q, I, S, W, R, E, H, C, P, N)$. The only bracket data in the carrier are L and U, and the only precision data are carried by Q.

The inclusion and shrink rows are downstream of those bracket and precision rows. They bind the current readback and successor readback to the same displayed route, so no consumer can bypass the bracket by choosing a host representative.

The finite-window and regular-readback rows W, R then expose the L10 observation surface. The equivalence and Real seal row E is terminal and is replayed only through H, C, P, N.

Thus the displayed route exhausts the visible packet. The excluded quotient, choice, order-oracle, trichotomy, and ambient-completeness coordinates are not present in the carrier and cannot be exported by the handoff. □

Falsifiable prediction: Every accepted $\mathsf{DyadicBracketApproximation}^{\uparrow}$ read exposes one lower dyadic bracket row, one upper dyadic bracket row, one unary precision row, one inclusion witness, one shrink witness, one finite StreamName window, one RegSeqRat readback, one Cauchy-equivalence and Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real or located-order consumer may inspect it.

Independence witness: dyadicratcore, streamname, regseqrat, cauchy-equivalence-setoid, real: $\mathsf{DyadicBracketApproximation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lower and upper bracket endpoints, unary precision, inclusion, shrink, finite windows, regular readback, equivalence sealing, transport, replay, provenance, and local naming in one L10 approximation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicBracketApproximation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicBracketApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5190.1, Definition 5190.2, and Definition 5190.3 seed the finite lower-upper dyadic bracket route between displayed observations and regular real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicBracketApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected real representatives, host located order, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-upper discipline, unary precision exactness, inclusion and shrink witness stability, regular readback handoff, seal nonescape, and local naming for BEDC.Derived.DyadicBracketApproximatio

Constructive story: A DyadicBracketApproximation$^{\uparrow}$ packet is generated from lower and upper DyadicRatCore$^{\uparrow}$ bracket rows, unary precision, inclusion and shrink witnesses, a finite StreamName$^{\uparrow}$ window, RegSeqRat$^{\uparrow}$ readback, CauchySequenceEquivalence$^{\uparrow}$ and Real$^{\uparrow}$ seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5191

A Concrete Naming Certificate for $\mathsf{LogicContradictionRefusal}^{\uparrow}$

The $\mathsf{LogicContradictionRefusal}^{\uparrow}$ packet realizes the contradiction-request boundary of section 9462.25 as finite NameCert data. It sits after the broader logic and meta-loop packet of chapter 1261, the local refutation boundary of chapter 500, the host-primitive leakage audit of chapter 1157, and the quotient-collapse refusal surface of chapter 634. The packet records a request to use contradiction as a global exporter, the finite reason that the request is refused, and the route by which local refutation remains available without host-level explosion.

Definition 5191.1 (LogicContradictionRefusal carrier)**.** *A* $\mathsf{LogicContradictionRefusal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, L, R, G, B, H, C, P, N)$$

with the following displayed rows:

1. *Q is the contradiction-request row. It records the proposed use of a local inconsistency, impossible branch, or false row as an exporter for an arbitrary global claim.*
2. *L is the local-refutation row. It identifies the finite context in which the contradiction is licensed and the boundary beyond which the refutation is not a proof-producing rule.*
3. *R is the refusal row. It records that the request in Q is refused as a host-level explosion, total-truth closure, unrestricted choice step, quotient collapse, or halting-closure oracle.*
4. *G is the gate row inherited from the* $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$ *surface. It keeps proof-pattern discipline and open meta-loop refusal visible before any downstream logical consumer reads the packet.*
5. *B is the sibling-boundary row. It records the overlap with* $\mathsf{RefutationBoundary}^{\uparrow}$, $\mathsf{HostPrimitiveLeakage}^{\uparrow}$, *and* $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$ *without identifying this packet with any of them.*
6. *H records componentwise* hsame *transport for Q, L, R, G, B and the local naming row.*
7. *C records displayed* Cont *replay from contradiction request through local refutation, refusal, gate boundary, sibling boundary, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{LogicContradictionRefusal}^{\uparrow}$, $\mathsf{LogicContradictionMetaLoop}^{\uparrow}$, $\mathsf{RefutationBoundary}^{\uparrow}$, $\mathsf{HostPrimitiveLeakage}^{\uparrow}$, $\mathsf{QuotientSoundnessBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LogicContradictionRefusal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, L, R, G, B, H, C, P, N$. It has no coordinate for ex falso as a host primitive, total truth, unrestricted choice, quotient representative collapse, global proof search, halting closure, or a closed meta-loop classifier.

Theorem 5191.2 (LogicContradictionRefusal NameCert obligations)**.** *For every accepted* $\mathsf{LogicContradictionRefusa}$ *carrier*

$$K = (Q, L, R, G, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LogicContradictionRefusal}^{\uparrow}}$ *surface consists exactly of contradiction-request admission through Q, local-refutation exposure through L, refusal through R, gate-boundary discipline through G, sibling-boundary separation through B, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N. The accepted packet refuses any route from local contradiction management to host-level explosion, total truth, unrestricted choice, quotient collapse, global proof search, halting closure, or a closed meta-loop classifier.*

Proof. Carrier admission is projection from Definition 5191.1. The rows $Q, L, R, G, B, H, C, P, N$ are the entire visible packet.

The contradiction request and local refutation rows separate the proposed exporter from the finite context that licenses a local negative result. Thus Q may record a request for global explosion while L records only the local refutation boundary.

The refusal and gate rows then block the exporter. The row R refuses host-level explosion, total truth, unrestricted choice, quotient collapse, global proof search, halting closure, and closed meta-loop classification, while G keeps the broader logic/meta-loop discipline in view.

Finally B, H, C, P, N preserve the sibling boundary, transport, replay, provenance, and naming rows. Since none of the excluded host-level coordinates occurs in the carrier, no local NameCert obligation can export them. □

Theorem 5191.3 (Logic contradiction refusal nonescape)**.** *Every accepted* $\mathsf{LogicContradictionRefusal}^{\uparrow}$ *read factors through the displayed route*

$$Q, L \longmapsto R, G, B \longmapsto H, C, P, N.$$

The route may expose a contradiction request, a local refutation boundary, a refusal row, the logic/meta-loop gate, sibling-boundary separation, transport, replay, provenance, and local naming. It cannot export a host explosion rule, total truth row, unrestricted choice step, quotient collapse, global proof-search result, halting closure, or closed meta-loop classifier.

Proof. Start with an accepted carrier $K = (Q, L, R, G, B, H, C, P, N)$. The only row that records the requested contradiction use is Q, and the only row that records the locally licensed negative content is L.

The refusal row R is downstream of both. It rejects the global exporter while leaving the local refutation row visible, so the packet does not erase contradiction management; it blocks only the attempt to turn it into a host-level proof primitive.

The gate and sibling-boundary rows G and B keep the refusal aligned with the broader meta-loop discipline and with the adjacent refutation, host-leakage, and quotient-boundary packets. Transport, replay, provenance, and naming are then carried only by H, C, P, N.

Thus the stated route exhausts the accepted read. The excluded explosion, total-truth, choice, quotient-collapse, global-search, halting, and closed-meta coordinates are absent from the packet. □

Theorem 5191.4 (Logic contradiction refusal TasteGate obligation)**.** *The Lean* `TasteGate.lean` *development for* $\mathsf{LogicContradictionRefusal}^{\uparrow}$ *must later supply explicit* $\mathsf{Nontrivial\ LogicContradictionRefusal}^{\uparrow}$ *and* $\mathsf{FieldFaithful\ LogicContradictionRefusal}^{\uparrow}$ *instances. Nontriviality must distinguish at least one accepted local-refutation refusal packet from a packet that turns Q into host-level explosion, and field faithfulness must preserve the row order*

$$Q, L, R, G, B, H, C, P, N$$

before any checked carrier-alignment theorem may serve as a public TasteGate realization.

Proof. The nontrivial boundary is forced by Definition 5191.3: an accepted refusal packet preserves local refutation while blocking host-level explosion. A packet that turns the request row Q into an arbitrary-claim exporter therefore cannot be the same accepted carrier.

Field faithfulness is the corresponding row-preservation obligation. A faithful read must keep the request, local refutation, refusal, gate, sibling boundary, transport, replay, provenance, and naming rows in the displayed order rather than replacing them by an external logical primitive. □

Falsifiable prediction: Every accepted $\mathsf{LogicContradictionRefusal}^{\uparrow}$ read exposes one contradiction-request row, one local-refutation row, one refusal row, one logic/meta-loop gate row, one sibling-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any logic-boundary consumer may inspect it.

Independence witness: logiccontradictionmetaloop, refutationboundary, hostprimitiveleakage, quotientsoundnessboundary: $\mathsf{LogicContradictionRefusal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a contradiction request, local refutation boundary, refusal row, meta-loop gate, sibling separation, transport, replay, provenance, and local naming while specifically refusing host-level explosion.

Closure status for $\mathsf{NameCert}_{\mathsf{LogicContradictionRefusal}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LogicContradictionRefusal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5191.1, Definition 5191.2, Definition 5191.3, and Definition 5191.4 seed the finite contradiction-request refusal route over local refutation and open meta-loop boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LogicContradictionRefusal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-level explosion, total truth, unrestricted choice, quotient collapse, global proof search, halting closure, closed meta-loop classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contradiction-request exactness, local-refutation boundary, refusal nonescape, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.LogicContradictionRefusalUp.

Constructive story: A $\mathsf{LogicContradictionRefusal}^{\uparrow}$ packet is generated from a contradiction-request row, local-refutation row, refusal row, logic/meta-loop gate row, sibling-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5192

A Concrete Naming Certificate for $\mathsf{WeakStarTopology}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{WeakStarTopology}^{\uparrow}$ packet records the weak-star topology through finite windows of continuous linear-functional reads on a displayed normed-space source. It sits over the normed carrier of chapter 1950, the finite weak-topology window of chapter 3597, the bounded-linear-operator surface of chapter 1644, and the scalar seal of chapter 38. The chapter names the finite observation-window certificate used by dual-facing consumers; it does not assert Banach–Alaoglu compactness, select dual elements, construct quotient function spaces, invoke choice, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5192.1 (Weak-star topology carrier)**.** *A* $\mathsf{WeakStarTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, W, L, T, R, H, C, P, N)$$

with the following displayed rows:

1. *V is the source* $\mathsf{NormedSpace}^{\uparrow}$ *row whose points are read only through the displayed normed-space carrier and its scalar rows.*
2. *W is the sibling* $\mathsf{WeakTopology}^{\uparrow}$ *observation-window row, reused as the finite neighbourhood interface rather than as a completed dual topology.*
3. *L is the finite bounded-linear-functional row. Each entry is a* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *read from V to the displayed scalar line, not a selected element of an unrestricted dual space.*
4. *T is the test-window row, listing the finitely many functional reads, source points, scalar tolerances, and comparison requests inspected by the consumer.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row for observed functional values and their finite tolerance comparisons.*
6. *H records componentwise* hsame *transport for the normed source, weak-window interface, bounded-linear-functional rows, test windows, scalar readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a requested weak-star read through T, then through L, V, W, R, and finally through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{WeakStarTopology}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{WeakTopology}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WeakStarTopology}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $V, W, L, T, R, H, C, P, N$. It has no coordinate for Banach–Alaoglu compactness, selected dual elements, quotient function spaces, choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.WeakStarTopologyUp.WeakStarTopologyTasteGate single carrier alignment`

Theorem 5192.2 (Weak-star topology NameCert obligations)**.** *For every accepted* $\mathsf{WeakStarTopology}^{\uparrow}$ *carrier $S = (V, W, L, T, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeakStarTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{NormedSpace}^{\uparrow}$ *row V, sibling* $\mathsf{WeakTopology}^{\uparrow}$ *window row W, finite bounded-linear-functional row L, test-window row T, scalar readback row R, componentwise transport H, displayed replay C, provenance P, and local naming N. Every public weak-star neighbourhood read must cite these rows before it reaches a scalar comparison.*

Proof. Project the carrier of Definition 5192.1. The only source carrier is V, the only weak-neighbourhood interface is W, and the only functional data are the finitely listed bounded-linear reads in L.

The test-window row T fixes which functional reads and source points are inspected. Scalar comparison then passes through R, so the public read receives finite $\mathsf{Real}^{\uparrow}$ observations rather than a selected dual element or a completed function space.

Transport, replay, provenance, and naming are supplied by H, C, P, N. Since the carrier contains no further coordinates, the listed rows exhaust the local NameCert surface. □

Theorem 5192.3 (Weak-star finite-window nonescape)**.** *Every scalar observation exported by an accepted* $\mathsf{WeakStarTopology}^{\uparrow}$ *carrier factors through*

$$V, W, L, T, R, H, C, P, N.$$

Thus a weak-star neighbourhood, convergence, or bounded-operator consumer may inspect only finite bounded-linear-functional windows and their $\mathsf{Real}^{\uparrow}$ *scalar comparisons. It receives no Banach–Alaoglu compactness theorem, no selected dual element, no quotient function space, no choice principle, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with an exported scalar observation. By Definition 5192.2, the observation must expose the source row V, the weak-window row W, the finite functional row L, and the test-window row T.

The scalar row R is the only route to real-valued readback. It records finite observed values and tolerance comparisons, not a completed dual-space object or a quotient of functions.

The structural rows H, C, P, N preserve and name that same route. Since the excluded compactness, selection, quotient, choice, and completeness principles have no carrier coordinate, they cannot be exported by the accepted packet. □

Theorem 5192.4 (Weak-star Banach–Alaoglu boundary)**.** *An accepted* $\mathsf{WeakStarTopology}^{\uparrow}$ *carrier may be consumed by a later compactness or dual-ball chapter only as the finite observation-window interface*

$$V, W, L, T, R, H, C, P, N.$$

Any Banach–Alaoglu-style conclusion must supply its own compactness, bounded-dual-ball, subnet, or cover ledger outside this packet. The weak-star packet itself contributes only the named finite reads and their transport, replay, provenance, and local NameCert rows.

Proof. Read the carrier inventory. The weak-star packet contains normed-space source data, a weak-window interface, finitely listed bounded-linear-functional reads, finite tests, scalar readbacks, and structural rows.

None of those rows is a compactness row, a selected dual-ball element, a subnet selector, a quotient function-space equality, or a choice principle. Therefore a compactness consumer cannot derive such data by projection from the packet.

The only available handoff is the finite observation route displayed in the theorem statement. A later Banach–Alaoglu chapter must add its own certificate rows before any compactness claim becomes public. □

Falsifiable prediction: Every public $\mathsf{WeakStarTopology}^{\uparrow}$ read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a sibling $\mathsf{WeakTopology}^{\uparrow}$ window, finite $\mathsf{BoundedLinearOperator}^{\uparrow}$ functional reads, $\mathsf{Real}^{\uparrow}$ scalar comparisons, BHist transport, hsame, Cont, Pkg, and local NameCert naming. A public weak-star read that exports Banach–Alaoglu compactness, selected dual elements, quotient function spaces, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness from this packet refutes the carrier boundary.

Independence witness: normedspace, weaktopology, boundedlinearoperator, real: $\mathsf{WeakStarTopology}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a normed source, a weak-window interface, finite bounded-linear-functional reads, scalar test windows, Real readbacks, transport, replay, provenance, and local naming while refusing compactness and selection data.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakStarTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakStarTopology}^{\uparrow}$ packet is generated from a displayed NormedSpace source, a WeakTopology finite-window row, finite BoundedLinearOperator functional reads, Real scalar readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakStarTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5192.1, Definition 5192.2, Definition 5192.3, and Definition 5192.4 seed the finite weak-star observation-window certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakStarTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Banach–Alaoglu compactness, selected dual elements, quotient function spaces, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-functional window stability, scalar observation nonescape, compactness-boundary refusal, transport, replay, provenance, and local naming for BEDC.Derived.WeakStarTopologyUp.

Chapter 5193

A Concrete Naming Certificate for $\mathsf{ObserverLocalityRateBudget}^{\uparrow}$

The $\mathsf{ObserverLocalityRateBudget}^{\uparrow}$ packet realizes the Inter-Hist Locality maximum-rate read as a finite paired-observer budget rather than as a global frame, metric tensor, or host-valued speed. It consumes the multi-history configuration route of chapter 474, the cross-Hist causal route of chapter 478, the maximum causal-rate surface of chapter 594, the public maximum-rate gate of chapter 933, and the rate-boundary refusal of chapter 1023. The packet is a concrete finite budget for observer-locality reads; it is not a global synchronization row, privileged frame, hidden clock, ambient metric geometry, or selected real-valued velocity.

Definition 5193.1 (Observer locality rate-budget carrier)**.** *An* $\mathsf{ObserverLocalityRateBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, C, R, G, S, T, N, H, K, P)$$

with the following displayed rows:

1. *M is the paired* MultiHistConfig *row, naming the two observer histories and their local history classifiers without adding a global synchronization row.*
2. *C is the* CrossHistCausal *row, recording the displayed causal-dependence reads between the two histories.*
3. *R is the* MaxCausalRate *row, exposing the maximum-rate bound only as a finite causal-rate read over M and C.*
4. *G is the* MaxRateReadGate *row, requiring observer-symmetry and no-global-frame refusal before the rate read becomes public.*
5. *S is the observer-symmetry budget row, recording the two-direction comparison that prevents either history from serving as the privileged frame.*
6. *T is the transport row for paired-history* hsame *and* psame *movement across the same causal-rate budget.*
7. *N is the no-global-frame refusal row, blocking shared clocks, global coordinates, metric tensors, host speeds, and selected synchronization data.*
8. *H records componentwise* hsame *transport for the paired histories, causal row, rate row, read gate, symmetry budget, transport, refusal, replay, provenance, and local naming rows.*

9. *K records displayed* Cont *replay from a requested public rate read through M, C, R, G, S, T, N and then to the structural rows.*
10. *P is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityRateBudget}^{\uparrow}}$ *naming row over the displayed packet.*

The classifier compares accepted packets only through $M, C, R, G, S, T, N, H, K, P$. It has no coordinate for a global frame, universal clock, metric geometry, host real-valued speed, selected observer representative, or choice principle.

Theorem 5193.2 (Observer locality rate-budget NameCert obligations)**.** *For every accepted* ObserverLocalityRateB *carrier $B = (M, C, R, G, S, T, N, H, K, P)$, the local* $\mathsf{NameCert}_{\mathsf{ObserverLocalityRateBudget}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *paired-history admission through M;*
2. *cross-Hist causal admission through C;*
3. *maximum causal-rate admission through R;*
4. *public maximum-rate gating through G;*
5. *observer-symmetry budgeting through S;*
6. *paired-history transport through T, H;*
7. *no-global-frame refusal through N;*
8. *displayed replay, provenance, and local naming through K, P.*

Every public maximum-rate read must pass through the observer-symmetry budget and the no-global-frame refusal before a consumer may cite the rate bound.

Proof. Project the carrier of Definition 5193.1. The paired histories enter through M, causal dependence through C, and the maximum-rate bound through R.

The read cannot become public at that point. The gate row G requires the observer-symmetry budget S and the no-global-frame refusal row N, while T and H preserve the paired-history route under displayed transport.

Finally replay and provenance are supplied by K, P. Because the carrier contains no further rows, the listed obligations exhaust the local NameCert surface. □

Theorem 5193.3 (Observer locality rate-budget nonescape)**.** *Every public rate read exported by an accepted* $\mathsf{ObserverLocalityRateBudget}^{\uparrow}$ *carrier factors through*

$$M, C, R, G, S, T, N, H, K, P.$$

In particular, omitting either the observer-symmetry budget S or the no-global-frame refusal N prevents the read from being public. The packet exports no global frame, universal clock, ambient metric geometry, selected observer representative, host real-valued velocity, or choice principle.

Proof. Begin with a public rate read. By Definition 5193.2, the read must expose paired histories, cross-Hist causality, and maximum causal-rate data through M, C, R.

The gate row G then checks the symmetry budget S and no-global-frame refusal N. If either row is absent, the requested read lacks one of the named obligations and cannot pass the local certificate.

Transport and replay through T, H, K preserve that same route, and P names it. Since the excluded global-frame, clock, metric, selected-observer, host-velocity, and choice data are absent from the carrier, they cannot be exported. □

Theorem 5193.4 (Observer locality rate-budget Lean carrier boundary)**.** *The follow-up Lean carrier for* ObserverLocalityRateBudget$^{\uparrow}$ *must expose a nontrivial public object and a field-faithful public classifier in 'lean4/BEDC/Derived/ObserverLocalityRateBudgetUp/TasteGate.lean'. Concretely, the carrier boundary is satisfied only when the Lean side provides 'NontrivialObserverLocalityRateBudgetUp' and 'FieldFaithful ObserverLocalityRateBudgetUp' for the same finite rows displayed in Definition 5193.1.*

Proof. The paper-side packet is not a purely verbal refusal; it carries the finite rows $M, C, R, G, S, T, N, H, K, P$. A Lean carrier that collapsed those rows to a singleton without public field separation would fail to represent the observer-symmetry and no-global-frame budget as distinguishable data.

The required nontriviality gives at least one public object boundary beyond the empty carrier. Field faithfulness then ties the public classifier to the displayed rows, so a maximum-rate read cannot be accepted after erasing the symmetry or refusal coordinates.

Thus the Lean follow-up must provide both named interfaces in the indicated TasteGate file before the carrier can be treated as a faithful formal companion to this chapter. □

Falsifiable prediction: Any public ObserverLocalityRateBudget$^{\uparrow}$ maximum-rate read that omits either observer-symmetry or no-global-frame refusal refutes the packet.

Independence witness: multihistconfig, crosshistcausal, maxcausalrate, maxratereadgate, interhistrateboundary: ObserverLocalityRateBudget$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries paired histories, causal rows, maximum-rate rows, public read gating, observer-symmetry budgets, no-global-frame refusal, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{ObserverLocalityRateBudget}^{\uparrow}}$.

Constructive story: An ObserverLocalityRateBudget$^{\uparrow}$ packet is generated from paired MultiHistConfig rows, CrossHistCausal rows, MaxCausalRate rows, MaxRateReadGate rows, observer-symmetry budgets, transport rows, no-global-frame refusal, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ObserverLocalityRateBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5193.1, Definition 5193.2, Definition 5193.3, and Definition 5193.4 seed the finite observer-locality rate-budget certificate surface.

Formal verification: FormalStatus(NameCert$_{\text{ObserverLocalityRateBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global synchronization, privileged frames, hidden clocks, ambient metric geometry, selected observers, host real-valued velocities, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked paired-history admission, causal-row admission, maximum-rate admission, read-gate exactness, observer-symmetry budgeting, no-global-frame refusal, transport, replay, provenance, local naming, nontriviality, and field faithfulness for BEDC.Derived.ObserverLocalityRateBudgetUp.

Chapter 5194

Mazur–Ulam Isometry NameCert Construction

Following Mazur and Ulam, the $\mathsf{MazurUlamIsometry}^{\uparrow}$ packet records the finite BEDC surface on which a surjective isometry between real normed spaces is inspected as midpoint-preserving and affine after its basepoint translation is displayed. It sits over the source and target normed carriers of chapter 1950, the scalar seal of chapter 38, the vector-operation rows of chapter 47, and the induced-distance readback of chapter 57. The chapter names the classical Mazur–Ulam route as finite history, classifier, midpoint, and affine-transport data; it does not import host affine geometry, choice of a global inverse outside the displayed isometry graph, or ambient Banach completeness.

Definition 5194.1 (Mazur–Ulam isometry carrier)**.** *A* $\mathsf{MazurUlamIsometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, F, G, M, A, H, C, P, N)$$

with the following displayed rows:

1. *E is the source* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing its* $\mathsf{VecSpace}^{\uparrow}$ *operations,* $\mathsf{Real}^{\uparrow}$ *scalar reads, norm row, induced* $\mathsf{Met}^{\uparrow}$ *distance row, transport, replay, provenance, and local naming.*
2. *F is the target* $\mathsf{NormedSpace}^{\uparrow}$ *row with the same displayed vector, scalar, norm, metric, transport, replay, provenance, and local naming surface.*
3. *G is the isometry graph row, recording accepted source histories, accepted target histories, distance equality through the metric rows of E, F, surjectivity by displayed target witnesses, and the distinguished basepoint pair used for translation.*
4. *M is the midpoint ledger, listing each accepted pair of source histories, the displayed source midpoint read, its graph image, the displayed target midpoint read of the two graph images, and the equality certificate connecting those target reads.*
5. *A is the affine-transport row, recording the translated map that sends the source basepoint to the target basepoint and preserving midpoint, dyadic convex, additive, and real-scalar reads through finite* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *ledgers.*
6. *H records componentwise* hsame *transport for E, F, G, M, A and the support rows.*
7. *C records displayed* Cont *replay for isometry inspection, midpoint requests, dyadic refinement, affine transport, and downstream Banach or operator-facing consumers.*

8. *P is the* Pkg *provenance over* $\mathsf{MazurUlamIsometry}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MazurUlamIsometry}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, F, G, M, A, H, C, P, N$. It has no coordinate for host affine hulls, quotient point equality, selected inverse functions outside G, ambient Banach completion, or non-displayed scalar extensionality.

Lean4 (definition) `BEDC.Derived.MazurUlamIsometryUp.MazurUlamIsometryUp`
Lean4 ✓ `BEDC.Derived.MazurUlamIsometryUp.MazurUlamIsometryTasteGate single carrier alignment`

Theorem 5194.2 (Mazur–Ulam midpoint preservation)**.** *For every accepted* $\mathsf{MazurUlamIsometry}^{\uparrow}$ *carrier*

$$U = (E, F, G, M, A, H, C, P, N),$$

each midpoint read exported by the isometry graph G factors through the midpoint ledger M. Thus a consumer receives the source midpoint row, the two endpoint graph rows, the target midpoint row, and the displayed equality certificate as one finite packet, with no appeal to host convexity or undisplayed affine structure.

Lean4 ✓ `BEDC.Derived.MazurUlamIsometryUp.MazurUlamIsometryCarrier midpoint preservation`

Proof. Project the carrier of Definition 5194.1. The graph row G supplies the accepted source endpoints, their target images, and the metric equality data, while M is the only row whose coordinates mention midpoint comparison.

A midpoint request therefore cannot be answered by the graph alone. It must read the source midpoint through the $\mathsf{VecSpace}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ rows of E, read the target midpoint through the corresponding rows of F, and then use the equality certificate stored in M.

The transport and replay rows H, C preserve exactly this dependency. Since the carrier has no host affine-hull row, no ambient convexity coordinate, and no selected inverse outside G, the displayed midpoint ledger exhausts the midpoint-preservation surface. □

Theorem 5194.3 (Mazur–Ulam affine linearity)**.** *For every accepted* $\mathsf{MazurUlamIsometry}^{\uparrow}$ *carrier $U = (E, F, G, M, A, H, C, P, N)$, every affine read of the translated isometry factors through the affine-transport row A after the graph row G and midpoint ledger M have been displayed. The exported affine route consists of the basepoint translation, midpoint preservation, dyadic convex refinement, additivity readback, real-scalar readback, and componentwise transport through the same source and target normed carriers.*

Lean4 ✓ `BEDC.Derived.MazurUlamIsometryUp.MazurUlamAffineLinearity`

Proof. Begin with the basepoint pair in G. The translated map is a displayed graph read sending the chosen source basepoint to the chosen target basepoint; no separate inverse or host affine map is available in the carrier.

The midpoint step is supplied by M. Repeated midpoint inspection gives the dyadic convex ledger over the $\mathsf{Real}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ rows already visible in E and F.

The affine row A packages the remaining additivity and real-scalar readback. Its entries are typed over the same graph, midpoint, scalar, and vector-operation rows, so downstream consumers cannot replace the finite route by an ambient linearity theorem, a quotient equality of functions, or an external Banach-space completion. □

Theorem 5194.4 (Mazur–Ulam NameCert obligations)**.** *For every accepted* $\mathsf{MazurUlamIsometry}^{\uparrow}$ *carrier* $U = (E, F, G, M, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MazurUlamIsometry}^{\uparrow}}$ *surface consists exactly of source normed-carrier admission, target normed-carrier admission, isometry-graph exactness, midpoint-ledger preservation, affine-transport readback, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.MazurUlamIsometryUp.MazurUlamIsometryCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5194.1. The only analytic rows are E, F, G, M, A, and the only structural rows are H, C, P, N.

The first two obligations expose the source and target normed-space data. These rows bind vector operations, real scalars, norms, and induced metrics before the isometry graph can be inspected.

Graph exactness is read from G, midpoint preservation from M, and affine transport from A. Each row depends on the preceding displayed rows, so the NameCert surface records a single route from isometry data to midpoint preservation and then to affine readback.

Finally H, C, P, N move, replay, source, and name the same packet. Since no carrier coordinate stores host affine geometry, an undisplayed inverse, quotient function equality, Banach completeness, or external scalar extensionality, the listed rows exhaust the local obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{MazurUlamIsometry}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MazurUlamIsometry}^{\uparrow}$ packet is generated from source and target NormedSpace rows, an isometry graph, a midpoint ledger, an affine-transport row, componentwise hsame transport, displayed replay, provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MazurUlamIsometry}^{\uparrow}}$, seed)

Scope closed: Definition 5194.1, Definition 5194.2, Definition 5194.3, and Definition 5194.4 seed the finite Mazur–Ulam route over NormedSpace, Real, VecSpace, MetricSpace, and Banach-facing rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MazurUlamIsometry}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Banach completeness, inverse selection beyond the displayed graph, host affine geometry, quotient equality of maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source and target carrier admission, isometry-graph exactness, midpoint-ledger preservation, affine-transport readback, transport, replay, provenance, and local naming for the same finite packet.

Chapter 5195

A Concrete Naming Certificate for $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ packet records the finite binary splitting route by which a regular Cauchy rational name is read at a dyadic tolerance, split into left and right finite windows, and sealed back into the $\mathsf{Real}^{\uparrow}$ boundary without selecting a host-real limit. It sits over the observation surface of chapter 126, the binary tolerance arithmetic of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives later interval, located-order, and completion consumers a displayed binary split schedule rather than a prose appeal to completeness or countable choice.

Definition 5195.1 (Regular Cauchy binary splitting carrier)**.** *A* $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W_0, W_1, D_0, D_1, R_0, R_1, A, \Sigma, H, C, P, N)$$

with the following displayed rows:

1. W_0 *and* W_1 *are the two finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the left and right branches of the split.*
2. D_0 *and* D_1 *are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *binary tolerance cells. They record the requested dyadic precision, the branch radii, the midpoint comparison budget, and the remaining slack for recombination.*
3. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed from* W_0, W_1, D_0, D_1.
4. A *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both windows, both tolerance cells, and both regular readbacks have been displayed.*
5. Σ *is the binary split schedule. It records the finite branch order, the midpoint comparison, the left-right recombination route, and the branch tag consumed by downstream interval or completion requests.*
6. H *records componentwise* hsame *transport for windows, tolerance cells, readbacks, the Real seal, split schedule, replay, provenance, and local naming.*
7. C *records displayed* Cont *replay from a Real or Completion request through* $W_0, W_1, D_0, D_1, R_0, R_1, A, \Sigma, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySplitting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W_0, W_1, D_0, D_1, R_0, R_1, A, \Sigma, H, C, P, N$. It has no coordinate for a selected real limit, quotient stream equality, countable choice, trichotomy, host located-order oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyBinarySplittingUp.RegularCauchyBinarySplittingUp`
Lean4 ✓ `BEDC.Derived.RegularCauchyBinarySplittingUp.RegularCauchyBinarySplittingTasteGate`
`single carrier alignment`

Theorem 5195.2 (Regular Cauchy binary splitting NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ *carrier*

$$B = (W_0, W_1, D_0, D_1, R_0, R_1, A, \Sigma, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySplitting}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for two finite* $\mathsf{StreamName}^{\uparrow}$ *windows, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance cells, two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, one* $\mathsf{Real}^{\uparrow}$ *seal, one binary split schedule, transport, replay, provenance, and local naming;*
2. *window discipline, requiring W_0 and W_1 to be read before any branch readback or Real-facing seal row is available;*
3. *tolerance discipline, requiring D_0 and D_1 to expose the requested precision, branch radii, midpoint comparison budget, and recombination slack before the split schedule may be consumed;*
4. *readback discipline, requiring R_0 and R_1 to be produced from the displayed windows and tolerance cells rather than from a selected host limit;*
5. *schedule discipline, requiring Σ to record the branch order, midpoint comparison, recombination route, and branch tag before interval or completion consumers inspect the packet;*
6. *seal discipline, requiring A, H, C, P, N to export only the displayed split route and no selected real limit, quotient stream equality, countable choice, trichotomy, host located-order oracle, or ambient completeness row.*

No RegularCauchyBinarySplitting obligation is stored outside these rows.

Proof. Project the carrier of Definition 5195.1. The accepted tuple exposes exactly two finite windows, two dyadic tolerance cells, two regular rational readbacks, one Real seal, one split schedule, and the structural transport, replay, provenance, and naming rows.

The window and tolerance disciplines form the first stage of the route. The two windows W_0, W_1 and the two tolerance cells D_0, D_1 must be visible before any branch readback, midpoint comparison, or Real-facing row is available.

The readback and schedule disciplines form the constructive splitting stage. The rows R_0, R_1 are produced from the displayed windows and dyadic cells, while Σ records the branch order, midpoint comparison, recombination route, and downstream branch tag.

Finally A, H, C, P, N seal, transport, replay, source, and name the same route. Since selected real limits, quotient stream equality, countable choice, trichotomy, host located-order oracles, and ambient completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5195.3 (Regular Cauchy binary splitting Real-seal nonescape)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*, interval, located-order,* $\mathsf{Real}^{\uparrow}$*, or Completion consumer of an accepted* $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ *carrier factors through the displayed route*

$$W_0, W_1 \longmapsto D_0, D_1 \longmapsto R_0, R_1 \longmapsto \Sigma \longmapsto A, H, C, P, N.$$

The consumer may read finite windows, dyadic binary tolerance cells, regular rational branch readbacks, the binary split schedule, the terminal Real seal, transport, replay, provenance, and local naming. It cannot replace that route by a selected real limit, quotient stream equality, countable choice, trichotomy, host located-order oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier $B = (W_0, W_1, D_0, D_1, R_0, R_1, A, \Sigma, H, C, P, N)$. The only observation data in the carrier are the two finite windows W_0, W_1, and the only tolerance data are the two dyadic cells D_0, D_1.

The regular readbacks R_0, R_1 are downstream of those rows. They expose branch values through finite StreamName windows and dyadic tolerance cells, not through a selected real representative.

The split schedule Σ is the only branch-order and recombination row. It records the midpoint comparison and downstream branch tag before any interval, located-order, Real, or Completion consumer may use the packet.

The Real seal A and the structural rows H, C, P, N preserve the same displayed route. The excluded limit, quotient, choice, trichotomy, oracle, and completeness coordinates are not present in the carrier, so they cannot escape through the seal. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ read exposes two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ binary tolerance cells, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, one $\mathsf{Real}^{\uparrow}$ seal row, one binary split schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any Real or Completion consumer may inspect it.

Independence witness: streamname, dyadicratcore, regseqrat, real: $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two finite windows, two binary tolerance cells, two regular readbacks, one Real seal, one branch schedule, transport, replay, provenance, and local naming in one L10 splitting packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySplitting}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySplitting}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5195.1, Definition 5195.2, and Definition 5195.3 seed the finite binary splitting route between displayed regular Cauchy rational reads and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyBinarySplitting}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected real limits, quotient stream equality, countable choice, trichotomy, host located order, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window discipline, binary tolerance exactness, branch readback stability, split schedule determinacy, Real-seal nonescape, and local naming for BEDC.Derived.RegularCauchyBinarySplittingU

Constructive story: A $\mathsf{RegularCauchyBinarySplitting}^{\uparrow}$ packet is generated from two finite $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{DyadicRatCore}^{\uparrow}$ binary tolerance cells, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, one $\mathsf{Real}^{\uparrow}$ seal, one binary split schedule, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5196

A Concrete Naming Certificate for $\mathsf{RieszInterpolation}^{\uparrow}$

Following Riesz and Kantorovich ordered-vector-lattice interpolation, the $\mathsf{RieszInterpolation}^{\uparrow}$ packet records the finite BEDC naming surface for lower-pair and upper-pair interpolation inside an ordered abelian source. It sits beside the ordered vector-lattice carrier of chapter 3620, the join-meet rows of chapter 55, the positive-cone boundary of chapter 2297, and the least-upper-bound consumer of chapter 3965. The packet names the interpolation witness route; it is not a host lattice-completion principle, a hidden supremum selector, or an ambient real-order oracle.

Definition 5196.1 (RieszInterpolation carrier)**.** *A* $\mathsf{RieszInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (G, L, P, A, B, W, H, C, S, N)$$

with the following displayed rows:

1. *G is the ordered abelian or ordered-group source row, naming the carried additive endpoint, zero, inverse, addition ledger, order comparison, and order-compatibility evidence used by the interpolation request.*
2. *L is the lattice or order row, read through* $\mathsf{Lattice}^{\uparrow}$ *or* $\mathsf{RieszSpace}^{\uparrow}$*, recording the finite meet-join and order-refinement rows on the same endpoint as G.*
3. *P is the positive-cone row, read through* $\mathsf{PositiveCone}^{\uparrow}$*, exposing only the displayed nonnegative-difference tests used to compare lower, witness, and upper rows.*
4. *A is the lower-pair row, recording two lower bounds a_0, a_1 and their order relations against the upper pair inside the same ordered source.*
5. *B is the upper-pair row, recording two upper bounds b_0, b_1 and the displayed comparisons $a_i \leq b_j$ required before interpolation can be requested.*
6. *W is the interpolation witness row, naming a displayed point w with $a_0 \leq w$, $a_1 \leq w$, $w \leq b_0$, and $w \leq b_1$ through the positive-cone and lattice rows.*
7. *H records componentwise* hsame *transport for the ordered source, lattice row, positive-cone row, lower-pair row, upper-pair row, witness row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from an interpolation request through G, L, P, A, B, W, H back to the same finite ordered surface.*

9. *S is the* Pkg *provenance row over* RieszInterpolation$^{\uparrow}$, RieszSpace$^{\uparrow}$, Lattice$^{\uparrow}$, PositiveCone$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RieszInterpolation}^{\uparrow}}$ *row binding the interpolation packet.*

The classifier compares accepted packets only through $G, L, P, A, B, W, H, C, S, N$. It has no coordinate for an external ordered group, a selected completion, a host supremum operation, or an ambient Real completeness proof.

Lean4 ✓ `BEDC.Derived.RieszInterpolationUp.RieszInterpolationTasteGate single carrier alignment`

Theorem 5196.2 (RieszInterpolation NameCert obligations)**.** *For every accepted* RieszInterpolation$^{\uparrow}$ *carrier $I = (G, L, P, A, B, W, H, C, S, N)$, the local* NameCert$_{\text{RieszInterpolation}^{\uparrow}}$ *surface consists exactly of carrier admission, ordered abelian or ordered-group source compatibility, lattice or order-row compatibility, positive-cone comparison, lower-pair display, upper-pair display, interpolation-witness display, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. A downstream* Real$^{\uparrow}$ *or completion-order-bound consumer may read the witness only along the displayed route*

$$G, L, P, A, B \longrightarrow W \longrightarrow H, C, S, N.$$

It receives no host metric completion, no hidden order quotient, no chosen supremum, and no additional Real-completeness coordinate.

Lean4 ✓ `BEDC.Derived.RieszInterpolationUp.RieszInterpolationCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5196.1. The rows G, L, P expose the ordered source, lattice or Riesz-space order rows, and positive-cone comparison surface, while A and B display the lower and upper pairs for the interpolation request.

The witness row W is accepted only against those displayed comparisons. Its four inequalities are read through the positive-cone and lattice rows already carried by P and L, so the witness cannot be detached from the finite ordered-group source.

The structural rows H, C, S, N preserve the same route under classifier transport, continuation replay, provenance, package data, and local naming. Since the carrier has no coordinate for a host completion, quotient representative, chosen supremum, or ambient Real-completeness principle, every downstream order-bound consumer receives only the displayed interpolation witness surface. □

Closure status for NameCert$_{\text{RieszInterpolation}^{\uparrow}}$.

Constructive story: A RieszInterpolation$^{\uparrow}$ packet is generated from an ordered abelian or ordered-group source row, lattice or order rows, a positive-cone row, displayed lower and upper pairs, an interpolation witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RieszInterpolation}^{\uparrow}}$, seed)

Scope closed: Definition 5196.1 and Definition 5196.2 seed the finite interpolation certificate over RieszSpace$^{\uparrow}$, Lattice$^{\uparrow}$, PositiveCone$^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows for later Real and completion order-bound consumers.

Formal verification: FormalStatus(NameCert$_{\text{RieszInterpolation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host ordered groups, selected suprema, metric completion, quotient representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ordered-source compatibility, lattice or order-row compatibility, positive-cone comparison, lower-pair and upper-pair display, witness display, transport, replay, provenance, and local naming for BEDC.Derived.RieszInterpolationUp.

Chapter 5197

A Concrete Naming Certificate for $\mathsf{Wasserstein}^{\uparrow}$

Following Kantorovich and Villani on optimal transport, the $\mathsf{Wasserstein}^{\uparrow}$ packet names finite transport-plan rows between displayed probability or distribution packets over a metric source. It sits beside the metric carrier of chapter 57, the compact metric source of chapter 129, the probability packet of chapter 184, the distribution ledger of chapter 186, and the real-valued cost seal of chapter 38. The chapter records only the displayed transport-plan interface; it does not claim Monge maps, optimal plan existence, measure-theoretic completion, duality, or selected minimizers.

Definition 5197.1 (Wasserstein transport-plan carrier)**.** *A* $\mathsf{Wasserstein}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$$

with the following visible rows:

1. *M is a displayed* $\mathsf{Met}^{\uparrow}$ *source, optionally strengthened by a* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover ledger K when the plan is read through compact support;*
2. *μ and ν are* $\mathsf{ProbSpace}^{\uparrow}$ *or* $\mathsf{Distribution}^{\uparrow}$ *marginal packets over displayed source rows of M;*
3. *Γ is a finite coupling support table whose entries name only listed source pairs and nonnegative displayed masses;*
4. *A and B are marginal-readback ledgers asserting that the finite row and column sums of Γ recover the displayed marginals μ and ν;*
5. *C is the transport-cost ledger through* $\mathsf{Real}^{\uparrow}$*, reading each support-pair cost from the metric row of M and summing it against the corresponding coupling entry;*
6. *H records componentwise* hsame *for the metric source, optional compact ledger, marginal packets, coupling entries, marginal ledgers, cost ledger, provenance, and local naming row;*
7. *R records displayed* Cont *replay routes from the metric and marginal rows to the coupling table, from the coupling table to the marginal ledgers, and from the cost ledger to the downstream real-valued consumer;*
8. *P is the* Pkg *provenance tying the metric, compact, probability, distribution, coupling, marginal-readback, and real-cost rows to their source packets;*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{Wasserstein}^\uparrow}$ *row.*

The classifier compares accepted packets componentwise by hsame *on the listed rows. No coordinate stores an ambient measure completion, a quotient of plans, a Monge function, a dual certificate, or a selected optimizer.*

Theorem 5197.2 (Wasserstein NameCert obligations)**.** *For every accepted* $\mathsf{Wasserstein}^\uparrow$ *carrier*

$$W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Wasserstein}^\uparrow}$ *surface consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed* BHist *packet with metric source, optional compact ledger, two marginal packets, finite coupling table, marginal-readback ledgers, real-valued cost ledger, transport, replay, provenance, and local naming rows;*
2. *marginal exactness: the ledgers A and B are the only marginal readbacks exported by the packet, and they recover μ and ν from finite sums of Γ;*
3. *finite coupling support: every coupling entry read by a consumer is indexed by a displayed source pair and carries a displayed nonnegative mass row;*
4. *transport-cost readback: the scalar cost exported to a* $\mathsf{Real}^\uparrow$ *consumer is the finite ledger C obtained from metric costs on the listed support pairs and the corresponding coupling entries;*
5. *provenance and name exhaustion:* hsame, Cont, Pkg, *and N preserve the same rows and introduce no Monge map, optimality witness, measure completion, dual certificate, or unlisted minimizer.*

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinCarrier namecert obligations`

Proof. Project the carrier of Definition 5197.1. Its visible coordinates are the metric source, optional compact ledger, two marginal packets, coupling support table, two marginal-readback ledgers, real-valued cost ledger, componentwise transport, replay, provenance, and local naming row.

The carrier admission row is therefore immediate from the displayed BHist packet. Marginal exactness is read from A and B: each ledger is attached to finite sums of Γ, so a consumer has no second route to a marginal outside the listed coupling table.

Finite coupling support follows by inspecting Γ. Every entry is indexed by a displayed support pair from the metric source, and transport by H preserves those indices componentwise rather than creating a quotient plan.

Finally read the cost ledger C through $\mathsf{Real}^\uparrow$. It sums only the metric costs named for listed support pairs against the visible coupling entries. The replay, provenance, and local naming rows preserve this same finite route, so the five obligation rows exhaust the Wasserstein transport-plan packet without adding Monge, optimality, completion, duality, or minimizer data. □

Theorem 5197.3 (Wasserstein metric coupling scope)**.** *Every accepted* $\mathsf{Wasserstein}^\uparrow$ *carrier* $W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$ *exposes its coupling table* Γ *only over the displayed* $\mathsf{Met}^\uparrow$ *source M and, when present, the* $\mathsf{CompactMetric}^\uparrow$ *finite-cover row K. The structural rows H, R, P, N preserve the same metric and compact-source rows before any coupling entry is read, so the packet does not export an unlisted ambient metric space, quotient support, or optimizer carrier.*

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinMetricCouplingScope`

Proof. Project the carrier of Definition 5197.1. The rows M and K are the only source rows available before a coupling entry can be inspected.

The finite table Γ names entries by displayed source pairs. Since H transports those source rows componentwise and R replays routes from the metric and marginal rows to Γ, a consumer cannot retarget the table to a different metric source.

The provenance and local name rows P, N record the same finite route. Thus the metric coupling scope is exactly the displayed M, K, Γ, H, R, P, N surface, with no hidden support quotient or selected optimizer. □

Theorem 5197.4 (Wasserstein marginal exactness scope)**.** *For every accepted* $\mathsf{Wasserstein}^{\uparrow}$ *carrier* $W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$*, the marginal packets* μ, ν *are read back from the finite coupling table only through the ledgers* A *and* B*. The rows* H, R, P, N *preserve the two marginal packets, the table* Γ*, and the readback ledgers together, so no consumer may replace marginal exactness by an ambient probability-space theorem, a detached distribution equality, or an unlisted coupling completion.*

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinMarginalExactnessScope`

Proof. Start with the marginal coordinates μ and ν. By the carrier definition they are the only $\mathsf{ProbSpace}^{\uparrow}$ or $\mathsf{Distribution}^{\uparrow}$ marginal packets named by W.

The readback ledgers A and B attach those marginals to finite row and column sums of Γ. Marginal exactness is therefore a displayed ledger assertion, not an external equality principle over all probability measures.

Finally inspect H, R, P, N. They transport, replay, source, and locally name the same marginal packets, coupling table, and readback ledgers. The scope is exhausted by $\mu, \nu, \Gamma, A, B, H, R, P, N$, so no ambient completion or detached distribution equality is available. □

Theorem 5197.5 (Wasserstein real cost readback scope)**.** *Every real-valued cost consumed from an accepted* $\mathsf{Wasserstein}^{\uparrow}$ *carrier* $W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$ *is read through the displayed ledger* C*, which sums metric costs from* M *over listed support pairs of* Γ *and then hands the scalar to the* $\mathsf{Real}^{\uparrow}$ *consumer. The structural rows* H, R, P, N *keep* C *attached to* $M, K, \mu, \nu, \Gamma, A, B$*, so the packet does not assert Kantorovich duality, optimal value existence, selected minimizers, or an ambient real-completeness principle.*

Proof. Read the cost row C after the metric and coupling rows have been displayed. Its summands are metric costs from M on listed support pairs of Γ multiplied by the corresponding coupling entries.

The marginal rows μ, ν and ledgers A, B stay in scope because they determine which finite table is being costed. The optional compact row K may constrain the source route, but it does not add a selected optimal plan.

The rows H, R, P, N preserve this same cost ledger before it reaches the $\mathsf{Real}^{\uparrow}$ consumer. Hence the cost readback scope is finite and displayed; duality, optimizer existence, minimizer selection, and ambient real completeness remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinRealCostReadbackScope`

Theorem 5197.6 (Wasserstein finite coupling obligation surface)**.** *For every accepted* $\mathsf{Wasserstein}^{\uparrow}$ *carrier* $W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$*, every coupling entry exposed to a consumer is one of the displayed rows of* Γ*, every marginal read is routed through* A *and* B*, and every cost read is routed through* C*. The structural rows* H, R, P, N *preserve that same finite route from metric*

source and optional compact support through marginal exactness and real-valued cost readback. Thus the packet exports no Monge map, optimizer, dual certificate, ambient measure completion, or selected minimizer.

Proof. Project the displayed carrier of Definition 5197.1. The only coupling-support coordinate is Γ, so each accepted coupling entry is indexed by a visible source pair and mass row already named by the packet.

Apply the NameCert obligations of Definition 5197.2, the metric coupling scope of Definition 5197.3, and the marginal exactness scope of Definition 5197.4. These identify A and B as the only marginal readback ledgers and keep the table attached to M, K, μ, ν.

Finally use the real cost readback scope of Definition 5197.5. The row C is the only exported cost ledger, and H, R, P, N only transport, replay, source, and name the same finite route. Since no carrier coordinate stores a Monge map, optimizer, dual certificate, ambient measure completion, or selected minimizer, the finite support and provenance rows exhaust the Wasserstein obligation surface. □

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinFiniteCouplingObligationSurface`

Theorem 5197.7 (Wasserstein transport-plan public route)**.** *Every public Wasserstein read of an accepted carrier $W = (M, K, \mu, \nu, \Gamma, A, B, C, H, R, P, N)$ factors through the displayed* $\mathsf{Met}^{\uparrow}$ *source M, the optional* $\mathsf{CompactMetric}^{\uparrow}$ *row K, the two* $\mathsf{ProbSpace}^{\uparrow}$ *or* $\mathsf{Distribution}^{\uparrow}$ *marginal rows μ, ν, the finite coupling table Γ, the marginal ledgers A, B, the* $\mathsf{Real}^{\uparrow}$ *cost ledger C, componentwise* hsame *transport H, displayed* Cont *replay R,* Pkg *provenance P, and the local* $\mathsf{NameCert}_{\mathsf{Wasserstein}^{\uparrow}}$ *row N. The route exports no Monge map, optimizer, Kantorovich duality certificate, measure completion, or selected minimizer.*

Proof. Start with the NameCert obligations of Definition 5197.2. They make carrier admission, marginal exactness, finite coupling support, cost readback, and provenance exhaustion the complete local certificate surface.

Next apply metric coupling scope and marginal exactness scope, Definition 5197.3 and Definition 5197.4. The table Γ stays attached to M, K, μ, ν, and the only marginal reads are the finite ledgers A and B.

Finally use real cost readback scope and the finite coupling obligation surface, Definition 5197.5 and Definition 5197.6. They keep C as the sole Real-valued cost route and force H, R, P, N to preserve the same displayed packet, so the public read has no hidden optimizer, dual object, completion, Monge map, or selected minimizer. □

Lean4 ✓ `BEDC.Derived.WassersteinUp.WassersteinTransportPlanPublicRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{Wasserstein}^{\uparrow}}$.

Constructive story: A $\mathsf{Wasserstein}^{\uparrow}$ packet is generated from a metric source, optional compact ledger, two displayed probability or distribution marginals, a finite coupling support table, marginal-readback ledgers, a real-valued transport-cost ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Wasserstein}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 5197.1, Definition 5197.2, Definition 5197.3, Definition 5197.4, and Definition 5197.5 close the finite transport-plan carrier, NameCert obligation surface, metric coupling scope, marginal exactness scope, and Real cost readback scope for Wasserstein-style metric coupling packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Wasserstein}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.WassersteinUp.WassersteinTransportPlanPublicRoute`

Bridge status: none

Not claimed: This chapter does not close Monge maps, optimal plan existence, measure-theoretic completion, Kantorovich duality, choice-selected minimizers, Lean verification, or bridge checking.

Upgrade path: Lean verification requires checked carrier admission, marginal exactness, finite coupling support, transport-cost readback, provenance exhaustion, and TasteGate nontriviality with field faithfulness for BEDC.Derived.WassersteinUp.

The CauchyProductRegSeq$^{\uparrow}$ hub routes the product-regularity carrier through the projection sibling of chapter 2076 while keeping the body in the child files below.

Chapter 5198

A Concrete Naming Certificate for $\mathsf{CauchyProductRegSeq}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences and finite product constructions, the $\mathsf{CauchyProductRegSeq}^{\uparrow}$ packet records the componentwise product of two regular Cauchy rational names as finite BEDC data. It consumes the finite stream windows of chapter 126, the regular rational readbacks of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name boundary of chapter 38, and the finite product carrier of chapter 34. The downstream projection route of chapter 2076 reads this product-regularity packet before exporting component readbacks to Real algebra consumers. The packet supplies a product-regularity source for later Real algebra and product-metric routes without invoking quotient streams, selected limits, host real equality, countable choice, or Real completeness.

Definition 5198.1 (CauchyProductRegSeq carrier)**.** *A* $\mathsf{CauchyProductRegSeq}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, P, H, C, Q, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows inspected from the component regular names.*
2. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
3. D *is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used for both component reads and for the product-window budget.*
4. P *is the* $\mathsf{Prod}^{\uparrow}$ *product row pairing the two readbacks at the displayed dyadic precision.*
5. H *stores componentwise* hsame *transport for both stream windows, both regular readbacks, the dyadic ledger, the product row, replay, provenance, and local naming.*
6. C *stores the displayed* Cont *replay from* S_0, S_1 *through* R_0, R_1, D *into the product row* P *and the product regular-sequence boundary.*
7. Q *is the* Pkg *provenance over* $\mathsf{CauchyProductRegSeq}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Prod}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductRegSeq}^{\uparrow}}$ *row.*

The accepted packet stores only this shared finite tolerance route. It does not identify streams by quotient equality, select a product limit, call on host real multiplication, use countable choice, or appeal to ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5198.2 (CauchyProductRegSeq regularity)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, P, H, C, Q, N)$$

be an accepted $\mathsf{CauchyProductRegSeq}^{\uparrow}$ *packet. If the component windows* S_0, S_1 *are read back by* R_0, R_1 *as regular rational names and the dyadic budget* D *is shared by the two component reads and the product row* P*, then the packet exposes a product regular-sequence route through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Prod}^{\uparrow}$*. Every product-regularity consumer is exhausted by* $S_0, S_1, R_0, R_1, D, P, H, C, Q, N$ *before any product seal is exported.*

In particular, the route exports no quotient stream equality, selected limit, host real equality, countable-choice selector, independent Real product carrier, or ambient Real completeness witness.

Proof. Project the carrier of Definition 5198.1. The two StreamName rows S_0, S_1 are the only observation windows available to the consumer, and the two RegSeqRat rows R_0, R_1 are the only regular rational readbacks obtained from them.

Next read the shared dyadic ledger D. Because both component readbacks and the product row P use this same finite tolerance data, the product regularity request remains on one displayed budget route rather than splitting into independent ambient Real names.

The product row P then pairs the two regular readbacks through the finite product surface of chapter 34. The transport row H and replay row C preserve the componentwise regularity data under displayed hsame and Cont moves, while the provenance row Q and local name N record exactly the sibling surfaces cited by the packet.

Since no listed row contains a quotient stream, a selected limit, a host equality test, a choice functional, a detached Real product carrier, or an ambient completeness principle, no product-regularity consumer can read such data from the accepted packet. □

Theorem 5198.3 (CauchyProductRegSeq sibling route)**.** *Every accepted product-regularity read for a* $\mathsf{CauchyProductRegSeq}^{\uparrow}$ *packet factors through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Prod}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *in the row order displayed by Definition 5198.1. The shared dyadic ledger* D *is the only tolerance budget visible to product consumers.*

In particular, the sibling route exports no quotient streams, selected product limits, host multiplication, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $K = (S_0, S_1, R_0, R_1, D, P, H, C, Q, N)$ from Definition 5198.1. The product consumer first sees the two $\mathsf{StreamName}^{\uparrow}$ windows S_0, S_1, then the two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks R_0, R_1, and then the single $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D.

Apply Definition 5198.2. The product row P is reached only after the shared ledger has bounded both component reads, and the $\mathsf{Real}^{\uparrow}$ seal is read through the displayed regularity route rather than through an ambient completeness principle.

The remaining rows H, C, Q, N transport, replay, source, and name that same route. Since no other tolerance row is displayed, D is the only budget available to product consumers, and none of quotient streams, selected limits, host multiplication, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness is exported. □

Theorem 5198.4 (CauchyProductRegSeq tail product bound)**.** *For every accepted* $\mathsf{CauchyProductRegSeq}^{\uparrow}$ *carrier*

$$K = (S_0, S_1, R_0, R_1, D, P, H, C, Q, N),$$

any product-tail consumer must display both input $\mathsf{StreamName}^{\uparrow}$ *windows* S_0, S_1*, both* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks* R_0, R_1*, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D *in its two component-read roles, and the finite product row* P *before any* $\mathsf{Real}^{\uparrow}$*-facing product seal is read. The bound is consumed through the* $\mathsf{Prod}^{\uparrow}$ *sibling surface and the sibling route of Definition 5198.3; it exports no quotient streams, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The windows S_0, S_1 display the inspected component tails, the readbacks R_0, R_1 expose the regular rational data, and the shared tolerance ledger D records the finite dyadic budget available to the product-tail consumer.

Apply the regularity theorem Definition 5198.2. Each component read must remain a displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ route controlled by the shared dyadic tolerance data before it can contribute to the product row.

Route the pair through the $\mathsf{Prod}^{\uparrow}$ sibling surface. The finite product row P is therefore reached only after both windows, both readbacks, and the shared tolerance ledger remain visible.

Finally use the sibling route theorem Definition 5198.3. It keeps the product read inside the displayed route before H, C, Q, N are consumed, so no quotient stream, selected limit, countable-choice selector, host real equality, or ambient completeness witness is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductRegSeq}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyProductRegSeq}^{\uparrow}$ packet is generated from two StreamName finite windows, two RegSeqRat readbacks, one shared DyadicRatCore tolerance ledger, a finite product row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductRegSeq}^{\uparrow}}$, seed)

Scope closed: Definition 5198.1 and Definition 5198.2 seed the finite product route for two regular Cauchy rational names over shared dyadic tolerance data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyProductRegSeq}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real multiplication, quotient equality of streams, selected product limits, countable choice, independent Real product completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, componentwise regularity transport, product-budget conservation, and Real-seal nonescape for the displayed packet.

Theorem 5198.5 (CauchyProductRegSeq product projection crosslink)**.** *Every downstream projection route through chapter 2076 must first read the shared* $\mathsf{CauchyProductRegSeq}^{\uparrow}$ *product-regularity packet*

$$S_0, \quad S_1, \quad R_0, \quad R_1, \quad D, \quad P, \quad H, \quad C, \quad Q, \quad N$$

displayed in Definition 5198.1. The projection consumer may expose a left or right component readback only after this packet has supplied the two finite stream windows, two regular rational readbacks, shared dyadic tolerance ledger, finite product row, componentwise transport, replay, provenance, and local naming.

Consequently, the projection route adds no quotient streams, selected product limits, host multiplication, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle to the product-regularity source.*

Proof. Project the accepted $\mathsf{CauchyProductRegSeq}^{\uparrow}$ carrier through Definition 5198.1. Its rows $S_0, S_1, R_0, R_1, D, P, H, C, Q, N$ are the only product-regularity data available before a downstream component projection is considered.

Now read the carrier of Definition 2076.1. The projection packet names component readbacks, product-modulus accounting, finite windows, dyadic tolerances, regular rational handoffs, sealing, transport, replay, provenance, and local naming, but its component-facing route is downstream of the already displayed product packet.

Thus every projection consumer factors through the shared product-regularity packet before selecting the left or right readback route. Since neither carrier contains quotient streams, selected product limits, host multiplication, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, the crosslink exports none of that data. $\square$

Chapter 5199

A Concrete Naming Certificate for $\mathsf{OsgoodUniqueness}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Osgood's uniqueness criterion for ordinary differential equations, the $\mathsf{OsgoodUniqueness}^{\uparrow}$ packet records the finite modulus ledger needed to separate constructive uniqueness from a Lipschitz or contraction hypothesis. It sits after the local initial-data surface of chapter 3979, the ODE row of chapter 193, the Picard handoff of chapter 1888, the contraction neighbour of chapter 5867, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names an Osgood-style integral divergence budget as displayed BEDC data; it is not a host Lipschitz constant, a Banach contraction theorem, a selected solution representative, or quotient equality of solution curves.

Definition 5199.1 (OsgoodUniqueness carrier)**.** *An* $\mathsf{OsgoodUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, I, F, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Ode}^{\uparrow}$ *source row, recording the differential equation, state window, and time window read by the uniqueness request.*
2. *I is the initial-data row, carrying the common initial time and initial* $\mathsf{Real}^{\uparrow}$ *state for the two compared solution traces.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row, read as a displayed continuous map on the same ODE window rather than as a host function.*
4. *M is the Osgood modulus ledger, listing dyadic positive radii, comparison budgets, and the finite divergence checks consumed by the uniqueness argument.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare distances, radii, and accumulated modulus budgets.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the two solution-distance traces inspected by the packet.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, I, F, M, Q, R have been displayed.*
8. *H records componentwise* hsame *transport for the ODE row, initial data, continuous-map row, modulus ledger, dyadic row, regular readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the Osgood uniqueness request through the ODE, initial-data, vector-field, modulus, dyadic, regular-rational, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OsgoodUniqueness}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OsgoodUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, F, M, Q, R, E, H, C, P, N$. It has no coordinate for a Lipschitz constant, a contraction ratio, selected solution representatives, quotient equality of curves, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.OsgoodUniquenessUp.OsgoodUniquenessTasteGate single carrier alignment`

Theorem 5199.2 (OsgoodUniqueness NameCert obligations)**.** *For every accepted* $\mathsf{OsgoodUniqueness}^{\uparrow}$ *carrier*

$$U = (D, I, F, M, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OsgoodUniqueness}^{\uparrow}}$ *surface consists exactly of carrier admission, ODE-window visibility through D, common initial-data visibility through I, continuous vector-field readback through F, Osgood modulus accounting through M, dyadic tolerance exactness through Q, regular rational distance readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the uniqueness route without adding a Lipschitz constant, a contraction ratio, a host fixed-point theorem, selected solution representatives, quotient equality of curves, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5199.1. The displayed rows $D, I, F, M, Q, R, E, H, C, P, N$ are the entire local surface, so carrier admission is exactly the admission of those rows.

The ODE and initial data are read first. The row D supplies the differential equation and local window, while I records the common initial time and initial real state. Thus the two solution traces are compared only inside one displayed initial-value surface.

The vector-field and modulus data are then consumed. The row F exposes continuity through $\mathsf{ContinuousMap}^{\uparrow}$, and the row M records the Osgood modulus ledger: dyadic radii, comparison budgets, and the finite checks that witness the divergence barrier used by the uniqueness argument.

The readback and seal rows are downstream. The row Q fixes the dyadic tolerance arithmetic, R reads the compared solution-distance traces as $\mathsf{RegSeqRat}^{\uparrow}$ data, and E exposes the terminal $\mathsf{Real}^{\uparrow}$ boundary only after those rows are present. Since no carrier coordinate stores a Lipschitz constant, contraction ratio, selected solution, quotient curve equality, countable-choice schedule, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 5199.3 (OsgoodUniqueness non-Lipschitz route)**.** *Every uniqueness-facing consumer of an accepted* $\mathsf{OsgoodUniqueness}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Ode}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ M,\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed by Definition 5199.1. The route supplies an Osgood modulus ledger for comparing solution traces; it does not export a Lipschitz constant, a Banach contraction certificate, a selected solution representative, or quotient equality between curves.

Proof. Begin with the common ODE surface. The rows D and I bind the equation, time window, state window, and common initial data before any solution-distance comparison is admitted.

Next read the vector-field and Osgood ledger. The continuous-map row F gives the displayed continuity route, while M records the finite modulus budgets consumed by the Osgood comparison. This is the analytic distinction from the contraction packets: the certificate reads a divergence ledger, not a strict contraction ratio.

Finally pass through dyadic tolerance, regular rational readback, and the terminal Real seal. The rows Q, R, E expose only the displayed distance comparisons and Real-facing boundary, and H, C, P, N transport, replay, source, and name the same finite route. Hence the accepted read remains an Osgood uniqueness packet rather than a Lipschitz or contraction theorem. □

Falsifiable prediction: An accepted $\mathsf{OsgoodUniqueness}^{\uparrow}$ uniqueness read that reaches $\mathsf{RegSeqRat}^{\uparrow}$ distance readback or a $\mathsf{Real}^{\uparrow}$ seal without displaying the rows $D, I, F, M, Q, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: ode, initial-value-problem, picard-lindelof, bishop-interval-contraction, regseqrat, dyadicratcore, real: no listed sibling supplies the joint packet containing an ODE source, common initial data, ContinuousMap vector-field readback, Osgood modulus ledger, dyadic tolerance arithmetic, regular rational distance readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{OsgoodUniqueness}^{\uparrow}}$.

Constructive story: An $\mathsf{OsgoodUniqueness}^{\uparrow}$ packet is generated from an ODE source row, common initial data, a ContinuousMap vector-field row, an Osgood modulus ledger, DyadicRatCore tolerance arithmetic, RegSeqRat distance readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{OsgoodUniqueness}^{\uparrow}}$, seed)

Scope closed: Definition 5199.1, Definition 5199.2, and Definition 5199.3 seed the finite Osgood uniqueness route from displayed ODE data and continuous vector-field rows through modulus accounting, dyadic tolerance arithmetic, regular readback, and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{OsgoodUniqueness}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Lean verification, bridge checking, Lipschitz uniqueness, Banach contraction, selected solution representatives, quotient equality of curves, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, ODE-window visibility, common initial-data visibility, continuous-map readback, Osgood modulus accounting, dyadic tolerance exactness, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OsgoodUniquenessUp.

Chapter 5200

A Concrete Naming Certificate for $\mathsf{GromovHausdorffDistance}^{\uparrow}$

Following Gromov and Burago–Burago–Ivanov, the $\mathsf{GromovHausdorffDistance}^{\uparrow}$ packet records the scalar comparison distance between two compact metric BEDC names as finite correspondence, distortion, and provenance data. It sits after the metric carrier of chapter 57, the compact metric surface of chapter 129, the scalar compact-subset route of chapter 3653, the finite correspondence comparison of chapter 2104, and the compact hyperspace surface of chapter 5184. The scalar distance carrier consumes $\mathsf{Met}^{\uparrow}$ distances and $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows before comparing through $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{GromovHausdorff}^{\uparrow}$ correspondences, and $\mathsf{CompactMetricHyperspace}^{\uparrow}$ packaging. The compact-subset scalar comparison row of chapter 3653 and the compact-hyperspace packaging of chapter 5184 are consumed before the Gromov–Hausdorff distance row is public. The packet names an inspectable distance bound for compact metric comparison; it is not a quotient metric space, a selected optimal correspondence, a choice principle over covers, or an ambient completion theorem.

Before any distance-between-spaces read is exposed, the packet opens the endpoint $\mathsf{Met}^{\uparrow}$ distance rows of chapter 57 and then a compact finite-net witness route such as the interval-cell finite-net discipline of chapter 5966. Thus the scalar comparison is attached to displayed metric distances and finite-net witnesses before the correspondence and distortion rows are read.

The distance packet is also a downstream consumer of the compact metric finite-net route of chapter 129 and the $\mathsf{HausdorffDistance}^{\uparrow}$ scalar comparison route of chapter 3653. The $\mathsf{GromovHausdorffDistance}^{\uparrow}$ row cannot be read before the compact metric endpoint rows and the Hausdorff scalar comparison rows are both displayed, because its finite correspondence and distortion ledger compare only those already public surfaces.

Definition 5200.1 (Gromov–Hausdorff distance carrier)**.** *A* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (X, Y, R, D, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are* $\mathsf{CompactMetric}^{\uparrow}$ *endpoint carriers, each exposing its* $\mathsf{Met}^{\uparrow}$ *distance rows, compact finite-net rows, and local compact-metric NameCert data before any cross-endpoint comparison is public.*

2. *R is the finite correspondence data between the endpoint point-name surfaces, including the two projection coverage ledgers and the inhabitedness rows needed for a correspondence read.*

3. *D is the dyadic distortion bound, recording a displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance for every pair of finite correspondence rows and the symmetric comparison budget induced by those rows.*

4. *H records the* hsame *transport rows that keep the endpoint carriers, correspondence rows, dyadic distortion ledger, replay row, provenance row, and local name attached under accepted renamings.*

5. *C records displayed* Cont *replay of the route*

$$X, Y \longmapsto R \longmapsto D \longmapsto H.$$

6. *P is the* Pkg *provenance over* $\mathsf{GromovHausdorffDistance}^{\uparrow}$, $\mathsf{GromovHausdorff}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{HausdorffDistance}^{\uparrow}$, $\mathsf{CompactMetricHyperspace}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

7. *N is the local* $\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, Y, R, D, H, C, P, N. It has no coordinate for quotient metric spaces, selected optimal correspondences, choice over covers, ambient completions, host isometry classes, or completed hyperspace points.

Theorem 5200.2 (Gromov–Hausdorff distance NameCert obligations)**.** *For every accepted* GromovHausdorffDistan *carrier*

$$Z = (X, Y, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the compact metric endpoints X, Y, finite correspondence data R, dyadic distortion bound D, transport row H, replay row C, provenance row P, and local name N;*

2. *correspondence stability, requiring the projection coverage and inhabitedness rows in R to remain attached to the same two compact metric endpoints under the transport row H;*

3. *distortion symmetry, requiring the dyadic bound D to compare both endpoint distance reads for every displayed pair of correspondence rows with the same tolerance budget;*

4. *triangle composition by correspondence composition, requiring composed comparison reads to concatenate the finite correspondence tables and add their displayed dyadic distortion budgets before a downstream consumer reads the result;*

5. *ledger refusal, requiring C, P, N to replay, source, and name only the displayed finite route and to export no quotient metric space, selected optimal correspondence, choice over covers, ambient completion, host isometry class, or completed hyperspace point.*

No Gromov–Hausdorff distance obligation is stored outside these rows.

Proof. Project the carrier of Definition 5200.1. The endpoint rows X, Y are the only compact metric sources, and the rows R, D, H, C, P, N are the entire comparison surface attached to them.

Carrier admission is therefore the admission of those displayed rows. The correspondence stability row is read from R together with H, because the projection ledgers and inhabitedness witnesses are transported only as rows over the same two endpoints.

The distortion symmetry row is read from D. Each finite pair of correspondence rows triggers two endpoint distance reads, and the same dyadic tolerance ledger bounds the two directed comparisons.

For triangle composition, compose the finite correspondence tables by their shared middle endpoint and add the two displayed dyadic distortion ledgers. The result is again a finite table with a displayed tolerance budget, so the composed comparison remains inside the same carrier discipline.

Finally C, P, N replay, source, and name the route. Since the carrier has no coordinate for quotient metric spaces, selected optimal correspondences, choice over covers, ambient completions, host isometry classes, or completed hyperspace points, the listed rows exhaust the local NameCert obligations. □

Theorem 5200.3 (Gromov–Hausdorff distance compact-metric handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$*-,* $\mathsf{CompactMetric}^{\uparrow}$*-,* $\mathsf{HausdorffDistance}^{\uparrow}$*-,* $\mathsf{GromovHausdorff}^{\uparrow}$*-, or* $\mathsf{CompactMetricHyperspace}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *packet factors through the finite route*

$$X, Y \longmapsto R \longmapsto D \longmapsto H, C, P, N.$$

Thus the consumer receives only displayed compact metric endpoints, finite correspondence rows, a dyadic distortion bound, transport, replay, provenance, and a local name. It cannot derive a quotient metric space, selected optimal correspondence, choice over covers, ambient completion, host isometry class, or completed hyperspace point from the packet.

Proof. Start with the compact metric endpoints. A metric or compact metric consumer can inspect only X and Y, so every distance read is anchored in displayed endpoint rows.

Next read the correspondence and distortion rows. The finite table R supplies the cross-endpoint pairs and coverage witnesses, while D supplies the dyadic distortion budget attached to exactly that table.

The sibling handoffs are downstream. A Hausdorff-distance, Gromov–Hausdorff comparison, or compact-hyperspace consumer may cite the route only after the finite correspondence and distortion rows have been displayed.

The structural rows H, C, P, N preserve, replay, source, and name the same route. Because no additional coordinate is present, the handoff cannot export a quotient metric object, selected optimum, cover choice, ambient completion, host isometry class, or completed hyperspace point. □

Theorem 5200.4 (Gromov–Hausdorff distance compact metric route)**.** *Every public distance comparison read from an accepted* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *packet factors through the compact metric dependency route displayed in the chapter introduction. The read first opens the* $\mathsf{Met}^{\uparrow}$ *distance rows of the two endpoints, then the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net coverage rows, then the* $\mathsf{HausdorffDistance}^{\uparrow}$ *scalar comparison surface, then the* $\mathsf{GromovHausdorff}^{\uparrow}$ *finite correspondence rows, then the* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *packaging rows, and finally the local* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}$ *rows of the distance packet. Thus the public comparison is a finite compact-metric route, not a quotient metric space, selected optimal correspondence, choice over covers, ambient completion, host isometry class, or completed hyperspace point.*

Proof. Start with the dependency list in the introduction. It orders the packet after the Metric, CompactMetric, HausdorffDistance, GromovHausdorff, and CompactMetricHyperspace surfaces, so a public scalar comparison must expose those rows before the local distance packet can be read.

Apply the NameCert obligations of Definition 5200.2. The carrier admission, correspondence stability, distortion symmetry, triangle-composition, and ledger-refusal rows say that the local certificate is exhausted by compact metric endpoints, finite correspondences, dyadic distortion data, transport, replay, provenance, and the local name.

Now use the compact-metric handoff theorem, Definition 5200.3. It identifies every sibling-facing consumer with the finite route

$$X, Y \longmapsto R \longmapsto D \longmapsto H, C, P, N.$$

Combining these readings gives the stated compact metric dependency route. Since none of the listed rows contains quotient metric spaces, selected optimal correspondences, cover choice, ambient completions, host isometry classes, or completed hyperspace points, no public distance comparison can obtain such data from the packet. □

Falsifiable prediction: An accepted $\mathsf{GromovHausdorffDistance}^{\uparrow}$ distance read that exposes a scalar comparison without first displaying X, Y, R, D, H, C, P, N refutes the carrier surface.

Independence witness: metric, compactmetric, hausdorff-distance, gromov-hausdorff, compact-metric-hyperspace: no listed sibling jointly carries two compact metric endpoints, finite correspondence data, dyadic distortion bounds, triangle composition by correspondence composition, transport, replay, provenance, and local NameCert rows for a scalar Gromov–Hausdorff distance interface.

Closure status for $\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}$.

Constructive story: A $\mathsf{GromovHausdorffDistance}^{\uparrow}$ packet is generated from two CompactMetricUp endpoint carriers, finite correspondence data, a DyadicRatCore distortion bound, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5200.1, Definition 5200.2, and Definition 5200.3 seed the finite scalar comparison route from compact metric endpoint certificates through correspondences and dyadic distortion bounds.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GromovHausdorffDistance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient metric spaces, selected optimal correspondences, choice over covers, ambient completions, host isometry classes, completed hyperspace points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, correspondence stability, distortion symmetry, triangle composition by correspondence composition, and ledger refusal for BEDC.Derived.GromovHausdorffDistanceUp.

The hub-level $\mathsf{GromovHausdorffDistance}^{\uparrow}$ route is read through chapter 57, chapter 129, chapter 3653, chapter 2104, and chapter 5184 before any child file exposes the local scalar distance packet. Every $\mathsf{GromovHausdorffDistance}^{\uparrow}$ correspondence and distortion read consumes $\mathsf{Met}^{\uparrow}$ distance rows together with compact or finite-net rows from chapter 57 and chapter 129 before it can export a distance comparison. The route therefore treats metric distance, compact coverage, finite-net inspection, transport, replay, provenance, and local naming as prerequisites of the public Gromov–Hausdorff distance surface, not as consequences of the distance comparison itself.

Theorem 5200.5 (Gromov–Hausdorff distance metric anchor)**.** *Every* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *read first exposes the* $\mathsf{Met}^{\uparrow}$ *distance witness of each endpoint and the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source rows before any correspondence table or distortion comparison is consumed. Thus the local scalar comparison is anchored in displayed metric distance and compact finite-source data, not in a quotient metric space, selected optimal correspondence, cover choice, ambient completion, host isometry class, or completed hyperspace point.*

Proof. Start with the endpoint rows X, Y of Definition 5200.1. Each endpoint is a compact metric carrier, so its metric distance witness and finite-source compact rows are displayed before the local correspondence row R can be read.

Apply the compact-metric route of Definition 5200.4. It orders public comparison reads through the Metric, CompactMetric, HausdorffDistance, GromovHausdorff, and CompactMetricHyperspace surfaces before the local rows H, C, P, N are consumed.

Finally use the metric-correspondence dependency of Definition 5200.6. Since the finite correspondence table and dyadic distortion ledger compare only already displayed endpoint distances and compact finite-source rows, no accepted Gromov–Hausdorff distance read can bypass the metric anchor. □

Theorem 5200.6 (Gromov–Hausdorff metric-correspondence dependency)**.** *Any* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *scalar read first consumes the sibling dependency route through* $\mathsf{Met}^{\uparrow}$ *endpoint distances,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows,* $\mathsf{GromovHausdorff}^{\uparrow}$ *finite correspondence rows,* $\mathsf{HausdorffDistance}^{\uparrow}$ *distortion comparison, and* $\mathsf{CompactMetricHyperspace}^{\uparrow}$ *packaging before the local distance packet exports*

$$X, \quad Y, \quad R, \quad D, \quad H, \quad C, \quad P, \quad N.$$

Thus the scalar route is a finite compact-metric and correspondence ledger: endpoint metrics and finite nets are visible before correspondence distortion is compared, hyperspace packaging is downstream of that comparison, and the local $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *packet exports only the displayed rows.*

Proof. Start with the carrier of Definition 5200.1. Its endpoint rows X, Y expose compact metric data, so every scalar distance read begins with the underlying $\mathsf{Met}^{\uparrow}$ distance rows and $\mathsf{CompactMetric}^{\uparrow}$ finite-net rows.

Apply the NameCert obligations of Definition 5200.2. The correspondence stability and distortion symmetry rows force the read through the finite correspondence table R and dyadic distortion ledger D before any local scalar export is available.

Finally use the compact-metric handoff theorem Definition 5200.3. It identifies the sibling-facing consumers as downstream of the same X, Y, R, D, H, C, P, N route, so Hausdorff-distance comparison, Gromov–Hausdorff correspondence data, and compact-metric hyperspace packaging cannot bypass the displayed metric-correspondence dependency. □

Theorem 5200.7 (Gromov–Hausdorff distance sibling dependency route)**.** *Every* $\mathsf{GromovHausdorffDistance}^{\uparrow}$ *packet consumes* $\mathsf{Met}^{\uparrow}$ *distance witnesses,* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net rows, and the finite correspondence route of chapter 2104 before any public distance read is exposed. The route first opens the metric distance rows of chapter 57, then the compact finite-net rows of chapter 129, and only then the correspondence and distortion rows used by the local distance packet. It does not assert classical compactness, quotient metric spaces, selected optimal correspondences, choice over covers, or ambient completion.*

Proof. Start with the endpoint rows of Definition 5200.1. The two compact metric endpoints are the only source of distance data for the scalar comparison.

The compact rows must expose their underlying $\mathsf{Met}^{\uparrow}$ distance witnesses and finite-net coverage before the local packet can inspect cross-endpoint pairs. Thus the read begins in chapter 57 and chapter 129.

Now use the finite correspondence surface of chapter 2104. The correspondence row supplies the displayed relation whose distortion is measured by the local D row.

Finally apply the NameCert obligations of Definition 5200.2. Since the local carrier contains only endpoint, correspondence, distortion, transport, replay, provenance, and naming rows, the sibling route cannot export classical compactness, quotient metric spaces, selected optimal correspondences, cover choice, or ambient completion. □

Chapter 5201

A Concrete Naming Certificate for $\mathsf{FrechetRiesz}^{\uparrow}$

Following Frechet and Riesz on Hilbert-space representation of bounded linear functionals, the $\mathsf{FrechetRiesz}^{\uparrow}$ packet records the finite Hilbert-dual surface on which a scalar functional is read through a representing vector. It sits over the Hilbert carrier of chapter 91, the inner-product rows of chapter 86, the scalar-functional surface of chapter 5790, the projection-theorem route of chapter 2374, the Riesz–Fischer completion handoff of chapter 4207, and the scalar endpoint seal of chapter 38. The chapter names only the displayed representing-vector route; it does not import an ambient dual-space isomorphism, quotient functional equality, a selected vector outside the packet, or a full Hilbert-space representation theorem as a host principle.

Definition 5201.1 (Frechet–Riesz carrier)**.** *A* $\mathsf{FrechetRiesz}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (H, I, L, V, E, K, P, R, T, C, Q, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, including the carried endpoints, additive and scalar rows, norm row, completeness boundary, and classifier inherited by the packet.*
2. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same source, exposing scalar pairing values through the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*
3. *L is the* $\mathsf{BoundedLinearFunctional}^{\uparrow}(H)$ *row, carrying the scalar action, linearity ledger, bounded-operator reading, and bound endpoint.*
4. *V is the representing-vector candidate row, a carried endpoint of the same Hilbert source.*
5. *E is the inner-product evaluation row recording, for each carried endpoint x inspected by the packet, the scalar comparison*
$$L(x) \sim_{\mathsf{Real}^{\uparrow}} \langle x, V \rangle.$$
6. *K is the norm-control row, binding the functional bound from L to the Hilbert norm of V and to the scalar endpoint discipline of* $\mathsf{Real}^{\uparrow}$*.*
7. *P is the* $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$ *handoff row, recording the projection-theorem route used to isolate the representing-vector candidate from the kernel-facing orthogonality data.*
8. *R is the* $\mathsf{RieszFischer}^{\uparrow}$ *handoff row, recording the coefficient-window and completion route used when the representing-vector candidate is read through Hilbert completion data.*

9. *T records componentwise* hsame *transport for H, I, L, V, E, K, P, R and for the local structural rows.*
10. *C records displayed* Cont *replay from a Hilbert-dual request through the Hilbert source, bounded functional, representing-vector candidate, evaluation ledger, norm-control row, projection handoff, and Riesz–Fischer handoff.*
11. *Q is the* Pkg *provenance over* $\mathsf{FrechetRiesz}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$, $\mathsf{RieszFischer}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetRiesz}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed Hilbert classifier, scalar-functional classifier, representing-vector transport, scalar evaluation transport, norm-control transport, projection and Riesz–Fischer handoff transport, replay, provenance, and local naming rows. It has no coordinate for a host dual-space isomorphism, quotient equality of functionals, nonconstructive orthogonal complement selection, unrestricted Hahn–Banach extension, or bridge checking.

Lean4 ✓ `BEDC.Derived.FrechetRieszUp.FrechetRieszTasteGate single carrier alignment`

Theorem 5201.2 (Frechet–Riesz NameCert obligations)**.** *For every accepted* $\mathsf{FrechetRiesz}^{\uparrow}$ *carrier $F = (H, I, L, V, E, K, P, R, T, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetRiesz}^{\uparrow}}$ *surface consists exactly of these finite obligation rows:*

1. *Hilbert source admission through H and inner-product admission through I;*
2. *scalar-functional admission through the* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row L;*
3. *representing-vector candidate admission through the carried endpoint row V;*
4. *scalar evaluation exactness through the displayed ledger E over the* $\mathsf{Real}^{\uparrow}$ *endpoint seal;*
5. *norm-control exactness through K, binding the functional bound and representing-vector norm without a host supremum;*
6. *projection handoff through P, routed through* $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$ *before any kernel-orthogonality read is public;*
7. *Riesz–Fischer handoff through R, routed through coefficient-window and completion data before any Hilbert completion read is public;*
8. *componentwise transport, replay, provenance, and local naming through T, C, Q, N;*
9. *non-escape from ambient Hilbert-dual isomorphisms, quotient functional equality, selected representatives, unrestricted Hahn–Banach separation, and hidden closed-subspace projection data.*

Proof. Project the carrier rows. The analytic surface is exactly the Hilbert source H, the inner-product row I, the scalar-functional row L, the representing-vector candidate V, the scalar evaluation ledger E, the norm-control row K, and the two handoff rows P, R.

The bounded functional is read before the representing vector is exposed. Its scalar action and bound pass through $\mathsf{BoundedLinearFunctional}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$, while the comparison with $\langle x, V\rangle$ is recorded only in the displayed ledger E over carried Hilbert endpoints.

The two theorem-facing handoffs are then forced to stay finite. The projection handoff P reads kernel-facing orthogonality through $\mathsf{OrthogonalProjectionTheorem}^{\uparrow}$, and the Riesz–Fischer handoff R reads completion-facing data through coefficient windows and displayed Hilbert rows.

Finally attach T, C, Q, N. These rows transport, replay, source, and locally name the same packet, and no remaining coordinate can manufacture a host dual-space isomorphism, quotient functional equality, selected representative, unrestricted Hahn–Banach witness, or hidden projection theorem. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5201.3 (Frechet–Riesz Hilbert-dual handoff)**.** *Every Hilbert-dual consumer of an accepted* $\mathsf{FrechetRiesz}^{\uparrow}$ *packet must read the route*

$$\mathsf{Hilbert}^{\uparrow}, \mathsf{InnerProduct}^{\uparrow}, \mathsf{BoundedLinearFunctional}^{\uparrow}, \mathsf{OrthogonalProjectionTheorem}^{\uparrow}, \mathsf{RieszFischer}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FrechetRi}$$

before it may treat a bounded scalar-valued functional as represented by a Hilbert endpoint. The exported handoff is the finite packet $F = (H, I, L, V, E, K, P, R, T, C, Q, N)$*: the consumer receives the functional row* L*, the representing endpoint* V*, the scalar ledger* E*, the norm-control row* K*, the projection and Riesz–Fischer handoffs* P, R*, and the structural rows* T, C, Q, N*, but no ambient dual-space equality or choice-selected representative.*

Proof. Start with a consumer asking for Hilbert-dual data. The source and scalar values are already confined to the displayed $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, so the consumer cannot inspect a host function space before reading H, I, L.

The representing-vector read is then mediated by V, E, K. The row V supplies a carried Hilbert endpoint, E records the scalar comparison with the functional values, and K records the norm-control boundary. These rows are part of the carrier and do not assert a global dual-space isomorphism.

The remaining access to analytic existence data factors through the two handoffs. The projection-theorem row P supplies the finite orthogonality route, while the Riesz–Fischer row R supplies the coefficient-window and completion route. Replaying the packet through T, C, Q, N yields the displayed $\mathsf{FrechetRiesz}^{\uparrow}$ handoff and nothing beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetRiesz}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrechetRiesz}^{\uparrow}$ packet is generated from a displayed Hilbert source, an inner-product row, a bounded scalar-functional row, a representing-vector candidate, scalar evaluation ledgers, norm-control data, orthogonal-projection and Riesz–Fischer handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrechetRiesz}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5201.1, Definition 5201.2, and Definition 5201.3 seed the finite Hilbert-dual representing-vector packet over Hilbert, inner-product, bounded-linear-functional, projection, Riesz–Fischer, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrechetRiesz}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an ambient Hilbert-dual isomorphism, quotient equality of functionals, selected representatives, unrestricted Hahn–Banach extension, full projection-theorem existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Hilbert source admission, bounded-functional admission, representing-vector admission, scalar evaluation exactness, norm-control exactness, projection handoff, Riesz–Fischer handoff, transport, replay, provenance, and local naming for BEDC.Derived.FrechetRieszUp.

Chapter 5202

A Concrete Naming Certificate for $\mathsf{CauchyRateNormalForm}^{\uparrow}$

Following Bishop–Bridges constructive Cauchy completion, the $\mathsf{CauchyRateNormalForm}^{\uparrow}$ packet records the finite rate-normalization row that turns an arbitrary displayed Cauchy-rate request into the monotone schedule consumed by the L10 Real and Completion route. It sits after the $\mathsf{CauchyRate}^{\uparrow}$ comparison surface of chapter 1671, the modulus-normal-form packet of chapter 1968, and the fast Cauchy route of chapter 323; it reads finite windows through chapter 126, rational tails through chapter 301, dyadic tolerances through chapter 291, and the terminal real seal through chapter 38. The chapter names rate normalization as a finite carrier, not as a host modulus function, quotient stream equality, selected tail, countable-choice schedule, or ambient completeness theorem.

Definition 5202.1 (CauchyRateNormalForm carrier)**.** *A* $\mathsf{CauchyRateNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (q, \rho, \nu, D, S, G, E, H, C, P, N)$$

with the following displayed rows:

1. *q is the precision-request row naming the rational or dyadic tolerance scale submitted by a Real or Completion consumer.*
2. *ρ is the raw* $\mathsf{CauchyRate}^{\uparrow}$ *row before monotone normalization.*
3. *ν is the normalized monotone rate row, recording the finite threshold schedule accepted for the same request q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tail ledger, carrying the dyadic slack and comparison budget used to certify that ν bounds the raw row ρ at the requested scale.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the source observations and threshold indices are inspected.*
6. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and the normalized threshold row.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after q, ρ, ν, D, S, G have been displayed.*
8. *H records componentwise* hsame *transport for the precision request, raw rate, normalized rate, dyadic ledger, stream windows, regular readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a precision request through raw-rate inspection, monotone normalization, dyadic tail accounting, finite windows, regular readback, and the Real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyRateNormalForm}^{\uparrow}$, $\mathsf{CauchyRate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{CauchyModulusNormalFo}$ $\mathsf{FastCauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyRateNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an external monotone majorant, quotient equality of Cauchy names, selected representatives, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyRateNormalFormUp.CauchyRateNormalFormTasteGate single carrier alignment`

Theorem 5202.2 (CauchyRateNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{CauchyRateNormalForm}^{\uparrow}$ *carrier*

$$R = (q, \rho, \nu, D, S, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyRateNormalForm}^{\uparrow}}$ *surface consists exactly of precision-request admission, raw Cauchy-rate exposure, normalized monotone rate exposure, dyadic tail-ledger accounting, finite* $\mathsf{StreamName}^{\uparrow}$ *window inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must expose q, ρ, ν, D, S, G before the seal row E can be consumed.*

Proof. Project the carrier of Definition 5202.1. The analytic rows are exactly the precision request, raw rate, normalized rate, dyadic tail ledger, finite windows, regular readback, and Real seal.

The raw rate row ρ is not consumed directly by the downstream Real seal. The replay row C first routes the precision request q through ρ and then through the monotone row ν.

The dyadic ledger D certifies the tail budget for the same request, while S and G display the finite observations and regular rational readback produced under the normalized threshold.

Rows H, P, N transport, source, and name only that displayed route. Since no excluded host object is present as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5202.3 (CauchyRateNormalForm Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer accepted by a* $\mathsf{CauchyRateNormalForm}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{CauchyRate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through the normalized monotone row ν and the support rows H, C, P, N. Consequently the consumer receives a displayed rate-normalized threshold, a dyadic tail budget, finite observation windows, regular rational readback, and a terminal Real seal; it receives no host modulus function, selected tail, quotient stream equality, countable-choice schedule, or ambient Real completeness witness.

Proof. Start with an accepted carrier $R = (q, \rho, \nu, D, S, G, E, H, C, P, N)$. The precision request q and raw Cauchy-rate row ρ are visible, but the carrier exposes them to Real consumers only through the normalized row ν.

Read the dyadic ledger D at the same request. This ledger supplies the tail budget that makes ν a usable monotone rate for the displayed route.

The stream and readback rows S, G then inspect the finite windows selected by the normalized threshold. Only after those rows are displayed can the terminal seal row E be consumed.

The support rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. The excluded host and choice principles are absent from the carrier, so no Real or Completion consumer can bypass the finite normalized-rate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyRateNormalForm}\uparrow}$.

Constructive story: A $\mathsf{CauchyRateNormalForm}^{\uparrow}$ packet is generated from a precision request, raw CauchyRate row, normalized monotone rate row, DyadicRatCore tail ledger, StreamName windows, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyRateNormalForm}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5202.1, Definition 5202.2, and Definition 5202.3 seed the finite rate-normalization route consumed by L10 Real and Completion packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyRateNormalForm}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected limits, host monotone majorants, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, raw-rate exposure, monotone normalization, dyadic tail-budget exactness, regular readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyRateNormalFormUp.

Chapter 5203

A Concrete Naming Certificate for $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$

Following Bishop–Bridges finite-cover compactness practice, the $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$ packet records the finite radius-stability object consumed by the compact-uniform-continuity fold of chapter 9237. It sits between the compact metric source of chapter 129, the continuous-map pointwise modulus rows of chapter 130, the positive rational-radius rows of chapter 37, the lower-bound fold of chapter 619, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet makes the finite cover radius ledger explicit, not a choice of Lebesgue number from an open-cover theorem, not a host compactness principle, and not an external modulus selector.

Definition 5203.1 (FiniteCoverRadiusStability carrier)**.** *A* $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, M, \mathcal{P}, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row carrying the finite net and located-distance coverage data.*
2. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row read at the bundled centers.*
3. *$\mathcal{P}$ is the nonempty* ProbeBundle *cover spine for the selected finite net.*
4. *R is the finite family of positive* $\mathsf{Rat}^{\uparrow}$ *radius rows, one for each bundled probe.*
5. *L is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}$ *lower-bound fold row certifying a positive radius below every displayed row in R.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row receiving the folded radius as a tolerance-only precision ledger.*
7. *H records componentwise* hsame *transport for the compact source, continuous-map row, bundle spine, radius family, fold row, uniform-modulus handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact finite-net coverage through pointwise radii, lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteCoverRadiusStability}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for an open-cover compactness theorem, an external Lebesgue-number selector, a chosen center outside $\mathcal{P}$, quotient point equality, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteCoverRadiusStabilityUp.FiniteCoverRadiusStabilityTasteGate single carrier alignment`

Theorem 5203.2 (FiniteCoverRadiusStability NameCert obligations)**.** *For every accepted* FiniteCoverRadiusStabili *carrier*

$$F = (K, M, \mathcal{P}, R, L, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteCoverRadiusStability}^{\uparrow}}$ *surface consists exactly of compact metric source exposure, continuous pointwise-modulus exposure, finite* ProbeBundle *cover admission, positive* $\mathsf{Rat}^{\uparrow}$ *radius admission, lower-bound fold exactness,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every uniform-modulus consumer must expose $K, M, \mathcal{P}, R, L$ before U can be read.*

Proof. Project the carrier of Definition 5203.1. The compact source, pointwise modulus, finite bundle, radius family, lower-bound fold, and uniform-modulus handoff are exactly the analytic rows.

The compact metric row K supplies only displayed finite-net coverage. The continuous-map row M supplies the pointwise-radius data attached to the centers named by the bundle spine $\mathcal{P}$.

The radius family R is folded by L inside the same finite bundle. Thus the downstream row U receives a classifier-stable positive lower bound rather than a host-selected Lebesgue number.

Rows H, C, P, N transport, replay, source, and name the same displayed route. Since no excluded open-cover, choice, quotient, or external-selector coordinate is present, these obligations exhaust the local NameCert surface. □

Theorem 5203.3 (FiniteCoverRadiusStability uniform-modulus handoff)**.** *Every compact-uniform-continuity consumer accepted by a* $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$ *carrier receives its* $\mathsf{UniformModulus}^{\uparrow}$ *precision row only through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{FiniteNetMinimumFold}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}$$

over the displayed ProbeBundle *spine. The handoff preserves the compact finite-net coverage, pointwise radius rows, positive rational lower-bound fold, componentwise transport, replay, provenance, and local naming rows, and it exports no open-cover compactness theorem, external Lebesgue-number selector, chosen center outside the bundle, quotient point equality, countable choice, or ambient completeness principle.*

Proof. Start with an accepted carrier $F = (K, M, \mathcal{P}, R, L, U, H, C, P, N)$. The finite-net cover is read from K and indexed by the displayed bundle $\mathcal{P}$.

At each bundled center, the continuous-map row M supplies a pointwise modulus radius. The row R stores those positive $\mathsf{Rat}^{\uparrow}$ radius histories as the finite family to be folded.

Apply the lower-bound row L. It certifies one radius below every displayed member of R, so the handoff row U receives a tolerance-only precision ledger produced inside the same finite route.

Transport, replay, provenance, and naming are carried by H, C, P, N. Because the carrier has no open-cover, external-selector, quotient, choice, or ambient-completeness coordinate, every uniform-modulus consumer remains confined to the displayed radius-stability packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverRadiusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$ packet is generated from a CompactMetric source row, ContinuousMap pointwise-modulus row, ProbeBundle finite cover spine, positive RatUp radius rows, FiniteNetMinimumFold lower-bound row, UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverRadiusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5203.1, Definition 5203.2, and Definition 5203.3 seed the finite radius-stability surface used by compact-uniform-continuity consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverRadiusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, external Lebesgue-number selection, centers outside the displayed bundle, quotient point equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous pointwise-radius exposure, positive RatUp radius admission, finite lower-bound fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverRadiusStabilityUp.

The $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ route is read as a separated completion consumer of chapter 1535, chapter 3526, and chapter 1606; the child files below keep the carrier, metric-completion dependency, and minimality handoff rows separate.

Chapter 5204

A Concrete Naming Certificate for $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$

Following Schikhof's constructive treatment of non-Archimedean normed spaces and the completion conventions used by Bosch, Guentzer, and Remmert, the $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ packet records an ultrametric completion boundary as finite NameCert data. It sits beside the metric source of chapter 57, the strong-triangle source of chapter 1535, the abstract completion interface of chapter 95, the separated-completion boundary of chapter 1612, and the dyadic-radius ledger of chapter 291. The completion boundary consumes the strong-triangle comparison from chapter 1535 and the nested-radius witness from chapter 3526 before any $\mathsf{Completion}^{\uparrow}$ interface row is exposed. The ultrametric source row is read through chapter 1535 before the dyadic radius ledger and separated-completion handoff are consumed. The packet records the displayed strong-triangle radius route and its separated handoff; it is not a quotient completion, a selected-representative construction, a host equality principle, countable choice, or ambient Real completeness.

Definition 5204.1 (Constructive ultrametric completion carrier)**.** *A* $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, D, T, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the ultrametric source row, recording the underlying* $\mathsf{Met}^{\uparrow}$ *carrier and its distance classifier.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *strong-triangle radius ledger, recording dyadic radii and the maximum-radius comparison used in every inspected triple.*
3. *T is the strong-triangle row, recording that the inspected distance requests factor through the ultrametric inequality at the radii displayed by D.*
4. *F is the Cauchy filter or Cauchy sequence route row, recording the finite tail requests and radius thresholds used by the completion consumer.*
5. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *handoff row, recording the separated boundary reached after the same Cauchy route is displayed.*
6. *Q is the* $\mathsf{Completion}^{\uparrow}$ *interface row, recording the completion-facing request without adding quotient classes or selected representatives.*

7. *H stores componentwise* hsame *transport for the ultrametric source, radius ledger, strong-triangle row, Cauchy route, separated handoff, completion interface, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from distance reads to strong-triangle radius checks, from Cauchy requests to separated handoff, and from the handoff to the completion interface.*
9. *P is the* Pkg *provenance over* $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}}$ *row.*

The accepted carrier stores only the rows just displayed. It carries no quotient-completion object, no selected Cauchy representative, no host equality of limits, no countable-choice selector, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 5204.2 (Constructive ultrametric completion NameCert obligations)**.** *For an accepted packet $U = (X, D, T, F, S, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}}$ *obligations are exhausted by carrier admission for the ultrametric source X, dyadic radius ledger D, strong-triangle row T, Cauchy filter or sequence route F, separated-completion handoff S, completion-interface row Q, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5204.1. The source metric data is read through X, while the ultrametric strengthening is visible only through the strong-triangle row T and the dyadic radius ledger D.

Read the completion-facing route next. The Cauchy filter or sequence row F reads the strong-triangle rows of chapter 1535 and the nested-radius witness of chapter 3526 before the separated handoff row S and completion-interface row Q are available. The row F displays the finite tail and radius requests, and the separated-completion row S records the handoff before the abstract completion interface Q can be consumed.

Finally inspect H, C, P, N. These rows transport, replay, package, and name the same displayed data through $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$. Since the packet has no quotient-completion, selected-representative, host-equality, countable-choice, or Real-completeness coordinate, no NameCert obligation can read one. □

Theorem 5204.3 (Constructive ultrametric completion separated-radius route)**.** *Let $U = (X, D, T, F, S, Q, H, C, P$ be an accepted* $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ *packet. Every completion-facing read routes the ultrametric strong-triangle row T and dyadic radius ledger D through the same Cauchy filter or sequence row F before the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *handoff S can be consumed. The abstract completion interface Q is available only after that separated-radius route has displayed the ultrametric source X, the dyadic ledger D, the strong-triangle row T, the Cauchy route F, and the separated handoff S.*

Proof. Project the carrier in Definition 5204.1. The NameCert obligations of Definition 5204.2 already force completion-facing reads to pass through the displayed ultrametric source, dyadic radius ledger, strong-triangle row, Cauchy route, and separated-completion handoff.

Induct over the displayed finite radius-request list carried by D and read by F. The empty list has no completion-facing radius request, so the separated handoff cannot be consumed except through the displayed structural rows.

For the cons step, the head request reads the same source X, dyadic ledger D, and strong-triangle row T before it enters the Cauchy filter or sequence row F. The induction hypothesis

preserves that route for the tail requests, so the separated handoff S remains attached to the same finite radius ledger before Q appears.

Thus each finite request preserves the ultrametric source, dyadic radius ledger, strong-triangle row, and separated handoff. No ambient completion, quotient limit, selected representative, or Real-completeness coordinate is available before the completion interface row is read. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ completion-facing read passes through an ultrametric source row, a strong-triangle dyadic radius ledger, a Cauchy filter or sequence route, a separated-completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any completion interface row is visible.

Independence witness: metricspace, completion, separatedcompletion, dyadicratcore: none is bijective to ConstructiveUltrametricCompletionUp because the packet jointly carries the ultrametric source, strong-triangle radius ledger, Cauchy route, separated handoff, completion interface, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ packet is generated from an ultrametric source row, a dyadic strong-triangle radius ledger, a Cauchy filter or sequence route, a separated-completion handoff, an abstract completion-interface row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5204.1 and Definition 5204.2 seed the finite ultrametric completion packet over MetricSpaceUp, CompletionUp, SeparatedCompletionUp, DyadicRatCoreUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected representatives, host equality of limits, countable choice, ambient Real completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, strong-triangle radius ledger exactness, Cauchy-route stability, separated-completion handoff, and completion-interface nonescape for BEDC.Derived.ConstructiveUltrametricCompletionUp.

Theorem 5204.4 (Constructive ultrametric completion Cauchy-completion dependency)**.** *Every* $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ *completion read first passes through the ultrametric carrier of Definition 5204.1, then through the metric-completion and separated-completion rows already named by chapter 1506 and chapter 1612. The strong-triangle and dyadic-radius rows from chapter 1535 remain visible before any completion interface is consumed, so the completion handoff is a carrier-local route rather than a direct appeal to quotient completion, selected representatives, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the constructive ultrametric completion carrier of Definition 5204.1. Its source row, dyadic radius ledger, strong-triangle row, Cauchy route, and separated handoff are already displayed before any completion interface row appears.

Then read the metric-completion and separated-completion siblings. They supply the named boundary rows that the handoff must pass through, while the ultrametric-space chapter keeps the strong-triangle and dyadic-radius rows visible. The route therefore stays local to the carrier and does not appeal to quotient completion, selected representatives, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The $\mathsf{ConstructiveUltrametricCompletion}^{\uparrow}$ route consumes the $\mathsf{MetricCompletion}^{\uparrow}$ completion carrier of chapter 1506 and the $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ minimal service boundary of chapter 1606 before any ultrametric completion seal is read. Thus the strong-triangle dyadic ledger remains a refinement of an already displayed completion carrier and minimal Cauchy-completion service route, not a replacement for either dependency.

Theorem 5204.5 (Constructive ultrametric completion minimality handoff)**.** *Every* ConstructiveUltrametricComple *completion-facing read that consumes the strong-triangle dyadic radius ledger, the Cauchy route, and the separated-completion handoff also routes through the* $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ *service boundary before it exports a completion interface. The exported route therefore displays the ultrametric source, dyadic-radius ledger, separated uniqueness handoff, transport, replay, provenance, and local naming rows before any* $\mathsf{Completion}^{\uparrow}$ *consumer can inspect the interface.*

Proof. Apply Definition 5204.2. It forces every accepted read to expose the ultrametric source, dyadic ledger, strong-triangle row, Cauchy route, separated-completion handoff, completion interface, transport, replay, provenance, and local naming rows.

Now read the minimality boundary of Definition 1606.2. Its service surface contains the same source metric row, dense or completion-facing handoff, universal route, separated uniqueness boundary, componentwise transport, replay, provenance, and local naming rows needed before a completed read can be exported.

Thus an ultrametric completion consumer cannot jump from the strong-triangle dyadic ledger directly to a public completion interface. It must pass through the displayed $\mathsf{CauchyCompletionMinimality}^{\uparrow}$ boundary, and the route exports no quotient completion, selected representative, countable-choice selector, host equality of limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness claim. □

Chapter 5205

A Concrete Naming Certificate for $\mathsf{MachineExportBoundary}^{\uparrow}$

The $\mathsf{MachineExportBoundary}^{\uparrow}$ packet realizes the machine-facing export side of section 9462.35 as one finite NameCert boundary. It sits beside the registry export surface of chapter 1216, the machine-readable audit interface of chapter 1232, the cannot-claim export gate of chapter 1319, and the certificate-preserving morphism route of chapter 481. Those siblings expose registry status, audit readback, refusal, and morphism preservation separately; this packet binds target, audit, refusal, transport, replay, provenance, and local NameCert rows before a machine export may be read.

Definition 5205.1 (Machine export boundary carrier)**.** *A* $\mathsf{MachineExportBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, T, A, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegistryExportSurface}^{\uparrow}$ *row, recording the exported, blocked, horizon-facing, or formal-target registry status.*
2. *T is the machine target row, recording the exact machine-facing target that may read the export.*
3. *A is the* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *row, recording the schema, checker, audit report, and non-evidence boundary for the export.*
4. *F is the* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *refusal row, recording any blocked claim that must remain visible before export.*
5. *H stores componentwise* hsame *transport for registry status, target, audit, refusal, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from registry lookup to target routing, audit readback, refusal inspection, and local NameCert exposure.*
7. *P is the* Pkg *provenance over* $\mathsf{MachineExportBoundary}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{MachineExportBoundary}^{\uparrow}}$ *row.*

The carrier contains no coordinate for an unregistered export, unchecked theorem evidence, refused-claim erasure, hidden transport, replay outside the displayed continuation row, provenance outside P, or a local name not carried by N.

Theorem 5205.2 (Machine export boundary NameCert obligations)**.** *For an accepted packet $E = (R, T, A, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MachineExportBoundary}^{\uparrow}}$ *obligations are exhausted by registry export admission through R, target exposure through T, audit readback through A, refusal preservation through F, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5205.1. The registry export row R, target row T, audit row A, and refusal row F are displayed before a machine consumer reaches local naming.

Read the sibling routes in order. The registry export surface fixes the public status, the machine-readable audit interface fixes the audit report and non-evidence boundary, and the cannot-claim export gate keeps refused rows visible.

Finally use H, C, P, N. These rows transport, replay, package, and name the same export route, with $\mathsf{NameCertMorphism}^{\uparrow}$ present only as provenance for certificate-preserving handoff. Since the carrier has no hidden export or proof-producing report coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 5205.3 (Machine export boundary registry route)**.** *Every machine export read from an accepted* $\mathsf{MachineExportBoundary}^{\uparrow}$ *packet must pass through* $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ $\mathsf{CannotClaimExportGate}^{\uparrow}$, *and* $\mathsf{NameCertMorphism}^{\uparrow}$ *before the local* $\mathsf{MachineExportBoundary}^{\uparrow}$ *naming row is consumed. In particular, the registry status route of chapter 1216 is a required sibling surface for any machine-facing export read.*

Proof. Project the carrier $E = (R, T, A, F, H, C, P, N)$. The first row R is the registry export surface, so a machine read begins with a displayed registry status rather than with a local machine target.

Next inspect A and F. The machine-readable audit interface fixes the audit report and non-evidence boundary, while the cannot-claim export gate preserves any refused claim that must remain visible.

The morphism route is then carried in P as certificate-preserving provenance before N is read as the local name. The transport and replay rows H, C preserve this sibling route, so the packet exposes no direct machine export that skips registry status, audit readback, refusal visibility, or NameCert morphism provenance. □

Theorem 5205.4 (Machine export boundary sibling route)**.** *Every accepted machine export route*

$$E = (R, T, A, F, H, C, P, N)$$

is a sibling-lattice consumer of $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, $\mathsf{NameCertMorphism}^{\uparrow}$, *and* $\mathsf{MachineInterfaceBoundary}^{\uparrow}$. *The rows R, A, F, P expose registry status, audit readback, refusal preservation, and certificate-preserving provenance before the local name row N is read, while H and C preserve the same route under transport and replay. Hence the route cannot introduce unregistered export, unchecked theorem evidence, or refusal erasure.*

Proof. Project the carrier $E = (R, T, A, F, H, C, P, N)$. The registry row R, audit row A, refusal row F, and provenance row P are displayed before the local naming row N.

Apply the registry route of Definition 5205.3. It forces every machine-facing export read through $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, and $\mathsf{NameCertMorphism}^{\uparrow}$ before the boundary can be named locally.

The broader $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ sibling is preserved by the target row T together with transport H and replay C. Those rows keep the same displayed export route and do not add

unregistered export, unchecked theorem evidence, or refusal erasure, so the accepted packet is exactly the stated sibling-lattice consumer. Consequently, downstream bounded-diamond and standard-bridge consumers must cite the displayed export route $E = (R, T, A, F, H, C, P, N)$ rather than a bare machine target row. □

Theorem 5205.5 (Machine export boundary interface forward link)**.** *Every* $\mathsf{MachineExportBoundary}^{\uparrow}$ *export route is a specialization of the* $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ *route of chapter 1221 after registry export, audit readback, refusal preservation, and* $\mathsf{NameCertMorphism}^{\uparrow}$ *provenance have been displayed. Concretely, for an accepted packet*

$$E = (R, T, A, F, H, C, P, N),$$

the rows R, A, F, P expose registry status, machine-readable audit, cannot-claim refusal, and certificate-preserving provenance before the target row T and local name row N are consumed. Thus the export boundary cannot be read as a machine-interface boundary unless it has first passed through the displayed export ledger.

Proof. Project the carrier $E = (R, T, A, F, H, C, P, N)$. The registry export row R, audit row A, refusal row F, and provenance row P are displayed before the local export name N.

The broader interface route supplies the machine-facing boundary, while this packet supplies the export specialization. The target row T selects the machine consumer, and the rows R, A, F, P add registry export, audit readback, refusal preservation, and certificate-preserving provenance.

Finally H and C preserve the same displayed route under transport and replay. Hence the accepted export packet is a specialization of $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ only after the export ledger has been shown, and no direct interface read can bypass those rows. □

The $\mathsf{MachineExportBoundary}^{\uparrow}$ packet refines the broader machine-facing interface of chapter 1221 by replacing a general registry-to-machine route with the target, audit, refusal, export tuple $E = (R, T, A, F, H, C, P, N)$ before any local export name is consumed.

It therefore consumes $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ as a downstream specialization: $\mathsf{RegistryExportSurface}^{\uparrow}$ supplies the public status row, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ supplies audit readback, $\mathsf{CannotClaimExportGate}^{\uparrow}$ supplies refusal visibility, and $\mathsf{NameCertMorphism}^{\uparrow}$ supplies certificate-preserving provenance before the local export boundary is read.

Falsifiable prediction: Any machine export omitting target, audit, refusal, transport, replay, provenance, or local NameCert rows refutes the $\mathsf{MachineExportBoundary}^{\uparrow}$ carrier.

Independence witness: registryexportsurface, machinereadableauditinterface, cannotclaimexportgate, namecertmorphism, machineinterfaceboundary: none is bijective to MachineExportBoundaryUp because the separate siblings expose registry status, audit readback, refusal visibility, certificate-preserving provenance, and the broader machine interface only one surface at a time. Only the MachineExportBoundaryUp packet jointly carries registry export status, machine target, audit readback, refusal preservation, certificate-preserving provenance, transport, replay, and local NameCert rows before an export can be read.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{MachineExportBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MachineExportBo}$ instances in `lean4/BEDC/Derived/MachineExportBoundaryUp/TasteGate.lean`. The field-faithfulness witness must preserve R, T, A, F, H, C, P, N as distinct rows before a checked target can cite the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MachineExportBoundary}^{\uparrow}}$.

Constructive story: A MachineExportBoundary$^{\uparrow}$ packet is generated from registry export status, a machine target row, a machine-readable audit row, a cannot-claim refusal row, componentwise hsame transport, displayed Cont replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{MachineExportBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 5205.1 and Definition 5205.2 seed the finite machine export boundary over RegistryExportSurfaceUp, MachineReadableAuditInterfaceUp, CannotClaimExportGateUp, NameCertMorphismUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{MachineExportBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unchecked theorem evidence, unregistered export, refusal erasure, hidden transport, off-packet replay, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked registry export admission, target exposure, audit readback, refusal preservation, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.MachineExportBoundaryUp.

Theorem 5205.6 (Machine export audit transcript envelope route)**.** *Every* MachineExportBoundary$^{\uparrow}$ *export that reports theorem evidence must first expose the* AuditTranscriptEnvelope$^{\uparrow}$ *status-pair and replay-checksum route of chapter 1234 before any machine-readable export row is consumed.*

Proof. Read a machine export as a finite boundary packet with exported claim, format, consumer, transport, replay, provenance, and local naming rows. A theorem-evidence report is stronger than a raw export row: it asks the consumer to trust both the paper-side closure grade and the Lean-side verification grade attached to the claim.

The AuditTranscriptEnvelope$^{\uparrow}$ carrier supplies exactly that two-axis status pair together with the replay-checksum row. Therefore the export boundary can expose theorem evidence only after routing through the audit transcript envelope; without those rows the machine consumer may still read an export, but not as theorem evidence. □

Theorem 5205.7 (Machine export boundary machine-interface dependency)**.** *Every* MachineExportBoundary$^{\uparrow}$ *consumer refines the broader machine-facing interface of chapter 1221 before any export is consumed. The route exposes registry status R, machine target T, audit readback A, refusal preservation F,* NameCertMorphism$^{\uparrow}$ *provenance P, componentwise transport H, displayed replay C, and local naming N in the same accepted packet $E = (R, T, A, F, H, C, P, N)$.*

Proof. Project the carrier of Definition 5205.1. The target row T places the accepted packet inside the machine-facing boundary, while the registry row R fixes the export status before the target can be consumed.

Use the existing registry route of Definition 5205.3 and the registry export dependency of Definition 5205.9. Together they force the consumer through RegistryExportSurface$^{\uparrow}$, MachineReadableAuditInterface$^{\uparrow}$, CannotClaimExportGate$^{\uparrow}$, and NameCertMorphism$^{\uparrow}$ before the local export name is read.

The sibling interface chapter supplies the ambient machine boundary only as a dependency surface. The export packet refines it by displaying R, T, A, F, H, C, P, N, so audit readback, refusal preservation, certificate-preserving provenance, transport, replay, and local naming remain visible before consumption.

Since no carrier coordinate skips from MachineInterfaceBoundary$^{\uparrow}$ directly to export, the accepted route cannot introduce unregistered export, unchecked theorem evidence, refusal erasure, hidden transport, or off-packet replay. □

Theorem 5205.8 (Machine export boundary forward interface route)**.** *Every* MachineExportBoundary$^{\uparrow}$ *export read reaches downstream machine-facing consumers only through the* MachineInterfaceBoundary$^{\uparrow}$ *route of chapter 1221 and the registry-export dependency rows of Definition 5205.9. The packet keeps registry status, machine target, audit readback, refusal preservation,* NameCertMorphism$^{\uparrow}$

provenance, transport, replay, and local naming visible before any export row can be treated as machine-facing evidence.

Proof. Project the carrier of Definition 5205.1. Its registry row is read before the machine target row, so the route cannot bypass the registry-export dependency surface.

Next apply the machine-interface dependency and registry-export dependency routes. They send the same displayed export packet through $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, and $\mathsf{NameCertMorphism}^{\uparrow}$ before the local name is consumed.

The structural rows preserve the same route. Componentwise transport, displayed replay, provenance, and local naming do not add a second export channel, so the accepted packet has no path to machine-facing evidence outside the displayed boundary rows. □

Theorem 5205.9 (Machine export boundary registry export dependency)**.** *Every* $\mathsf{MachineExportBoundary}^{\uparrow}$ *read must pass through the registry export surface of chapter 1216 before any machine-facing export is consumed. The remaining sibling rows are the* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *audit route, the* $\mathsf{CannotClaimExportGate}^{\uparrow}$ *refusal route, and the* $\mathsf{NameCertMorphism}^{\uparrow}$ *certificate-preserving handoff.*

Proof. Project the carrier $E = (R, T, A, F, H, C, P, N)$ from Definition 5205.1. The registry row R precedes the target row T, so the export is visible first as a registry status and only then as a machine-facing route.

Read the remaining sibling rows through A, F, P. The audit route supplies the machine-readable report and non-evidence boundary, the refusal route keeps blocked claims visible, and the morphism row appears as certificate-preserving provenance before the local name is consumed.

Transport and replay through H, C preserve the same route. Therefore no accepted $\mathsf{MachineExportBoundary}^{\uparrow}$ consumer reaches machine-facing export before registry export, audit readback, refusal visibility, and NameCert morphism provenance have all been displayed. □

Theorem 5205.10 (Machine export boundary registry-forward dependency)**.** *Every* $\mathsf{MachineExportBoundary}^{\uparrow}$ *export that is read by a standard-bridge consumer factors first through the machine interface route of chapter 1221, then through the registry rows of chapter 1216, and finally through the same local* $\mathsf{NameCert}$ *provenance row displayed in Definition 5205.1. No standard-bridge consumer may read a bare export row detached from* $\mathsf{MachineInterfaceBoundary}^{\uparrow}$*, registry status, or NameCert provenance.*

Proof. Project the accepted carrier $E = (R, T, A, F, H, C, P, N)$ from Definition 5205.1. The target row T places the export inside the machine-facing interface, while the registry row R fixes its registry status before the export can be consumed.

Apply the machine-interface dependency route of Definition 5205.7 and the registry export dependency of Definition 5205.9. Together they force any downstream read through $\mathsf{MachineInterfaceBoundary}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, audit readback, refusal preservation, and certificate-preserving provenance before the local name is used.

The standard bridge route of Definition 5205.16 only reads the already displayed packet. Since H, C, P, N transport, replay, package, and name the same accepted carrier, the bridge cannot bypass the machine interface row, registry rows, or local NameCert provenance. □

Every $\mathsf{MachineExportBoundary}^{\uparrow}$ read refines the machine-facing boundary of chapter 1221 and must pass through chapter 1216, chapter 1232, chapter 1319, and chapter 481 before any machine-facing export is consumed.

Theorem 5205.11 (Machine export boundary NameCert sibling route)**.** *Every public machine-facing export row of an accepted* $\mathsf{MachineExportBoundary}^{\uparrow}$ *packet must first consume the registry export surface of chapter 1216 and the machine-interface boundary of chapter 1221. Concretely, for an accepted packet*

$$E = (R, T, A, F, H, C, P, N),$$

the registry row R *and target row* T *are read before any machine-facing export row is public, while* A, F, H, C, P, N *keep audit readback, refusal visibility, transport, replay, provenance, and local naming attached to the same finite packet. The route does not export host semantics, detached machine output, unchecked theorem evidence, or a hidden registry state.*

Proof. Project the carrier of Definition 5205.1. The registry row R is the displayed $\mathsf{RegistryExportSurface}^{\uparrow}$ input, and the target row T is the displayed machine-facing input.

Apply the registry export dependency of Definition 5205.9. It requires the registry surface to be visible before a machine-facing export is consumed.

Apply the machine-interface dependency of Definition 5205.7. It keeps the target row inside the broader $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ route rather than letting the packet expose a bare machine target.

Finally read the obligation surface of Definition 5205.12. The audit, refusal, transport, replay, provenance, and naming rows remain attached to $E = (R, T, A, F, H, C, P, N)$. Since the carrier has no host-semantics coordinate, detached output row, unchecked-proof row, or hidden-registry coordinate, the public route is exactly the stated sibling dependency. □

Theorem 5205.12 (Machine export boundary obligation surface)**.** *Every public* $\mathsf{MachineExportBoundary}^{\uparrow}$ *export must pass through the displayed machine-interface, registry-export, audit-provenance,* hsame-*transport,* Cont-*replay,* Pkg*, and local* NameCert *rows before any external machine read is public. In particular, the route refines the machine-facing interface of chapter 1221 and the registry export surface of chapter 1216 through the accepted packet* $E = (R, T, A, F, H, C, P, N)$*.*

Proof. Project the carrier of Definition 5205.1. The target row T supplies the machine-interface surface, while R supplies the registry-export surface before a machine read can be public.

The audit and refusal rows A, F keep provenance-facing evidence and blocked-claim visibility in the same packet. Transport through H and replay through C preserve those displayed rows rather than replacing them with an external machine assertion.

Finally read P and N. Package provenance and the local NameCert row close the export boundary only after the machine-interface, registry-export, audit, refusal, transport, and replay rows are visible, so no external read is public outside the displayed obligation surface. □

Theorem 5205.13 (Machine export boundary bounded diamond consumer)**.** *Every* MetaCICBoundedDiamondWin *consumer of* $\mathsf{MachineExportBoundary}^{\uparrow}$ *must read registry status, machine target, audit readback, refusal preservation, componentwise transport, displayed replay, package provenance, and local NameCert rows through the accepted packet* $E = (R, T, A, F, H, C, P, N)$ *before bounded diamond evidence can be exported. In particular, the bounded-window consumer of chapter 8559 may cite the obligation surface of Definition 5205.12 only after all eight rows remain visible in the same machine export boundary packet.*

Proof. Project the packet $E = (R, T, A, F, H, C, P, N)$ from Definition 5205.12. The registry status row R, target row T, audit row A, and refusal row F are the machine-facing data that a bounded diamond consumer must inspect before export.

The structural rows H, C, P, N keep that data inside the accepted packet. Transport and replay preserve the same registry, target, audit, and refusal route, while provenance and local naming keep the route tied to $\mathsf{MachineExportBoundary}^{\uparrow}$.

Therefore a $\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}$ consumer cannot export bounded diamond evidence from a bare machine read. It first factors through the displayed MachineExportBoundary obligation surface, with every row of E still visible. □

Theorem 5205.14 (Machine export boundary bounded diamond packet route)**.** *Every bounded-diamond consumer route from* $\mathsf{MachineExportBoundary}^{\uparrow}$ *to* $\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}$ *must cite chapter 8559 through the full accepted machine-export packet* $E = (R, T, A, F, H, C, P, N)$*. The bounded diamond evidence cannot be read from a bare export row, a detached registry row, or an audit-only machine read; it becomes public only after the registry status, machine target, audit readback, refusal preservation, componentwise transport, displayed replay, package provenance, and local NameCert rows remain visible together.*

Proof. Apply Definition 5205.13. It already forces bounded-window consumers to inspect all eight rows of E before machine export can serve bounded diamond evidence.

The chapter route to chapter 8559 is therefore a packet route, not a single-row export. Reading only the registry row, only the target row, or only an audit row would omit the refusal, transport, replay, provenance, or naming data required by the accepted MachineExportBoundary surface.

Thus the bounded-diamond dependency edge is carried by the whole packet $E = (R, T, A, F, H, C, P, N)$. A bare export row is not enough to expose bounded diamond evidence to a $\mathsf{MetaCICBoundedDiamondWindow}^{\uparrow}$ consumer. □

Theorem 5205.15 (Machine export boundary orphan interface route)**.** *Every* $\mathsf{MachineExportBoundary}^{\uparrow}$ *packet is read as a downstream specialization of chapter 1221 only after registry status, audit readback, refusal preservation, componentwise transport, displayed replay, package provenance, and local* $\mathsf{NameCert}$ *rows remain visible. Concretely, an accepted export packet* $E = (R, T, A, F, H, C, P, N)$ *reaches the machine-interface boundary through the registry route of Definition 5205.3, the registry export dependency of Definition 5205.9, and the interface dependency of Definition 5205.7; it cannot be consumed as a bare machine target row. The bounded-diamond and* Ω*-generation boundary consumers may read the export packet only after* $\mathsf{RegistryExportSurface}^{\uparrow}$ *and* $\mathsf{MachineReadableAuditInterface}^{\uparrow}$ *have displayed the export rows.*

Proof. Project the carrier of Definition 5205.1. The displayed rows R, T, A, F, H, C, P, N are the only rows available to a machine export consumer.

Read the registry and audit surfaces first. The registry route and registry export dependency force R to expose status before T is consumed, while A and F preserve audit readback and refusal visibility inside the same packet.

Now apply the machine-interface dependency and the forward interface route, Definition 5205.7 and Definition 5205.8. They route the target row through $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ only after $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, and $\mathsf{NameCertMorphism}^{\uparrow}$ provenance are displayed.

Finally use H, C, P, N. Transport and replay keep the same registry-audit-refusal route visible, provenance records the certificate-preserving handoff, and the local name closes the packet. Hence the orphan edge is a child route through $\mathsf{MachineInterfaceBoundary}^{\uparrow}$ with all export-boundary rows still present. □

Theorem 5205.16 (Machine export boundary standard bridge route)**.** *For an accepted* $\mathsf{MachineExportBoundary}^{\uparrow}$ *packet* $E = (R, T, A, F, H, C, P, N)$*, the machine export reads bridge data only after the registry export surface, the machine-readable audit interface, and the registry export dependency route have already been displayed. The standard bridge protocol chapter of chapter 9204 supplies the host-facing comparison surface, while the registry export surface of chapter 1216 and the machine interface boundary of chapter 1221 keep the export finite.*

Proof. Project the carrier of Definition 5205.1. The registry row R, target row T, audit row A, and refusal row F are the only public machine-export rows.

Use the registry export dependency theorem of this chapter together with the machine-interface dependency theorem. They force the packet through the registry-export surface, the machine-readable audit interface, the refusal gate, and the certificate-preserving provenance row before the local name is consumed.

The standard bridge protocol chapter then provides the host-facing bridge shape without changing the packet's displayed rows. Since no extra coordinate is present, the export reads bridge data only after the registry and protocol rows are visible. □

Chapter 5206

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailPairing}^{\uparrow}$

Following Bishop–Bridges constructive real analysis, the $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ packet records the finite paired-tail surface by which two regular rational Cauchy names are synchronized before a diagonal or sequential-completeness consumer reaches a displayed $\mathsf{Real}^{\uparrow}$ seal. It sits over the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the sequential-completeness handoff of chapter 1794, and the terminal real-name boundary of chapter 38. The packet is a finite paired-tail carrier for L10 Real and Completion routes; it is not quotient stream equality, a selected limit, countable choice, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5206.1 (Regular Cauchy tail pairing carrier)**.** *A* $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S_0, S_1, R_0, R_1, D_0, D_1, W, U, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows opened by the paired-tail request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from those windows.*
3. D_0 *and* D_1 *are the two* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, recording the requested radius, the local regularity budgets, and the remaining comparison slack for each side of the pair.*
4. W *is the paired-tail synchronization row, recording one displayed common tail window for the two readbacks and no quotient identification of the underlying streams.*
5. U *is the* $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ *handoff row, read only after the two windows, two readbacks, two dyadic ledgers, and paired-tail window have been displayed.*
6. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same paired route.*
7. H *records componentwise* hsame *transport for windows, readbacks, dyadic ledgers, paired-tail synchronization, sequential-completeness handoff, Real seal, replay, provenance, and local naming.*
8. C *records displayed* Cont *replay from a diagonal or sequential-completeness request through* $S_0, S_1, R_0, R_1, D_0, D_1, W, U, E, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyTailPairing}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPairing}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through the displayed rows above. It has no coordinate for quotient stream equality, selected host limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegularCauchyTailPairingUp.RegularCauchyTailPairingUp`

Theorem 5206.2 (Regular Cauchy tail pairing NameCert obligations)**.** *For every accepted* RegularCauchyTailPairin *carrier*

$$T = (S_0, S_1, R_0, R_1, D_0, D_1, W, U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPairing}^{\uparrow}}$ *surface consists exactly of carrier admission for the two StreamName windows, two RegSeqRat readbacks, two DyadicRatCore tolerance ledgers, paired-tail synchronization row, RegularCauchySequentialCompleteness handoff, Real seal, componentwise transport, displayed replay, package provenance, and local naming; synchronization exactness for the common tail window W; classifier stability through H; replay stability through C; and ledger non-escape through P and N.*

Proof. Carrier admission is projection from Definition 5206.1. The accepted tuple exposes exactly the two windows, two readbacks, two dyadic ledgers, paired-tail row, sequential-completeness handoff, Real seal, and structural rows.

Synchronization exactness is the row W. A diagonal or sequential-completeness consumer must first read the two StreamName windows and their RegSeqRat readbacks, then compare them under the two dyadic ledgers before the common tail window is public.

The handoff rows are downstream of that paired window. The row U consumes the same displayed route as a $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ handoff, and the row E is the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after $S_0, S_1, R_0, R_1, D_0, D_1, W$ have been exposed.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded quotient, selection, choice, trichotomy, host-equality, or ambient-completeness coordinate occurs in the carrier, the listed clauses exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyTailPairingUp.RegularCauchyTailPairingTasteGate single carrier alignment`

Theorem 5206.3 (Regular Cauchy paired-tail synchronization)**.** *Let*

$$T = (S_0, S_1, R_0, R_1, D_0, D_1, W, U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ *carrier. Any diagonal or sequential-completeness consumer of T must synchronize the two tails by reading S_0, S_1, then R_0, R_1, then D_0, D_1, and only then the common paired-tail window W. The synchronization is transported componentwise by H and replayed by C before U or E is consumed.*

Proof. Apply the obligation theorem Definition 5206.2. It orders the public rows so that the two finite windows and readbacks are visible before either dyadic tolerance ledger is used.

The paired-tail row W is therefore not a quotient or selected representative. It is a displayed common window obtained after both regular rational readbacks have been inspected under their two dyadic budgets.

Componentwise transport and replay preserve that order. The rows H and C can move and repeat the displayed packet, but they cannot make the sequential-completeness handoff U or Real seal E visible before the paired-tail synchronization row has been read. □

Theorem 5206.4 (Regular Cauchy tail pairing RealUp seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegularCauchySequentialCompleteness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer must display the two finite windows, two regular rational readbacks, two dyadic tolerance ledgers, one paired-tail synchronization row, one sequential-completeness handoff, and one terminal Real seal before any Real-facing output is named.

Proof. Start with the paired-tail synchronization theorem Definition 5206.3. It forces every accepted consumer to read $S_0, S_1, R_0, R_1, D_0, D_1, W$ in order.

The sequential-completeness handoff is then row-local. The row U cites the existing RegularCauchySequentialCon surface of chapter 1794 and consumes the same paired route rather than selecting a host limit.

Only after that finite handoff does the Real seal row E become public. The structural rows H, C, P, N preserve and name the route, so the handoff exports no quotient stream equality, selected limit, countable-choice witness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ read displays two $\mathsf{StreamName}^{\uparrow}$ windows, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, one paired-tail synchronization row, one $\mathsf{RegularCauchySequentialCompleteness}^{\uparrow}$ handoff, one $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before a diagonal or sequential-completeness consumer reaches a Real-facing output.

Independence witness: streamname, regseqrat, dyadicratcore, regular-cauchy-sequential-completeness, real: $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two windows, two regular readbacks, two dyadic ledgers, one paired-tail synchronization row, one sequential-completeness handoff, one Real seal, and the structural transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailPairing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyTailPairing}^{\uparrow}$ packet is generated from two StreamName windows, two RegSeqRat readbacks, two DyadicRatCore tolerance ledgers, one paired-tail synchronization row, one RegularCauchySequentialCompleteness handoff, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailPairing}^{\uparrow}}$, seed)

Scope closed: Definition 5206.1, Definition 5206.2, Definition 5206.3, and Definition 5206.4 seed the finite paired-tail carrier used by diagonal and sequential-completeness consumers before the displayed $\mathsf{Real}^{\uparrow}$ seal is read.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailPairing}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected host limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, paired-tail synchronization, classifier transport, replay stability, sequential-completeness handoff, Real seal handoff, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyTailPairingUp.

Chapter 5207

A Concrete Naming Certificate for $\mathsf{RegulatedMeasure}^{\uparrow}$

Following Riemann, Stieltjes, and the Bishop–Bridges constructive treatment of regulated functions and bounded variation, the $\mathsf{RegulatedMeasure}^{\uparrow}$ packet records finite regulated partition-measure data. It sits between the displayed regulated-function rows of chapter 1906, the partition-variation surface of chapter 398, the tagged-sum route of chapter 1637, the dyadic partition rows of chapter 1863, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite partition-measure certificate; it is not a sigma-measure, a measure completion, a Lebesgue integration object, a host-function quotient, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 5207.1 (RegulatedMeasure carrier)**.** *A* $\mathsf{RegulatedMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, V, \Pi, \Delta, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, inspected only through finite step-window and oscillation-control data.*
2. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *control row, recording the partition-variation budget that bounds all displayed interval contributions.*
3. *Π is the* $\mathsf{DyadicPartition}^{\uparrow}$ *row, naming the ordered cells and refinement witnesses on which the packet is read.*
4. *Δ is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *interval-mass ledger, assigning finite rational mass rows to the cells of Π under the displayed variation budget.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regulated finite measure values at the requested windows.*
6. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, V, Π, Δ, R have been displayed.*
7. *H records componentwise* hsame *transport for the regulated source, variation budget, partition, mass ledger, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from regulated-function partition reads through bounded-variation control, dyadic mass ledgers, regular rational readback, and the terminal real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedMeasure}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{RiemannStieltjes}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, V, \Pi, \Delta, R, E, H, C, P, N$. It has no coordinate for countable additivity, a sigma-algebra completion, a host measure quotient, selected limits, countable choice, Lebesgue integration, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.RegulatedMeasureUp.TasteGate.RegulatedMeasureTasteGate single carrier alignment`

Theorem 5207.2 (RegulatedMeasure NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedMeasure}^{\uparrow}$ *carrier*

$$M = (F, V, \Pi, \Delta, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated source F, bounded-variation control through V, dyadic partition exposure through Π, interval-mass exactness through Δ, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5207.1. The source rows F, V, Π are finite regulated-function, bounded-variation, and dyadic partition data, so carrier admission begins from displayed rows rather than from an ambient measure space.

The interval-mass obligation is the dyadic ledger Δ. Its cells are exactly the cells of Π, and its allowed values are constrained by the variation budget V before any downstream readback is visible.

The readback and seal obligations are R and E. They route the finite mass ledger through regular rational windows and then through the terminal real seal, while H, C, P, N preserve transport, replay, provenance, and local naming for the same packet.

Since no omitted object is a carrier coordinate, the listed rows exhaust the NameCert surface and exclude sigma-measure completion, host quotient equality, selected limits, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5207.3 (RegulatedMeasure Stieltjes handoff)**.** *Every* $\mathsf{RiemannStieltjes}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegulatedMeasure}^{\uparrow}$ *packet reads the measure-facing contribution only through the displayed route*

$\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, F

Concretely, the consumer must expose the regulated source row F, the bounded-variation budget V, the dyadic partition Π, the interval-mass ledger Δ, and the regular rational readback R before the $\mathsf{Real}^{\uparrow}$ *seal E is read.*

Proof. Begin with the regulated source and variation rows. The regulated-function surface supplies finite step-window data, and the bounded-variation surface supplies the finite budget controlling interval contributions.

Next expose the dyadic partition and mass ledger. The $\mathsf{DyadicPartition}^{\uparrow}$ row fixes the cells that a Stieltjes-facing consumer may inspect, and Δ assigns only finite dyadic mass rows to those cells.

Finally pass through regular rational readback and the real seal. The $\mathsf{RegSeqRat}^{\uparrow}$ row reads the displayed finite ledger, and E seals that same readback as a terminal $\mathsf{Real}^{\uparrow}$ row. No row in

the route supplies countable additivity, sigma-measure completion, selected limits, host quotient equality, Lebesgue integration, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedMeasure}^{\uparrow}$ packet is generated from a regulated-function source row, a bounded-variation control row, a dyadic partition row, an interval-mass DyadicRatCore ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5207.1, Definition 5207.2, and Definition 5207.3 seed the finite regulated partition-measure packet consumed by Riemann–Stieltjes and real-name routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim sigma-measure completion, countable additivity, Lebesgue integration, host-function quotient equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation control, dyadic partition exposure, interval-mass exactness, regular rational readback, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedMeasureUp.

Chapter 5208

A Concrete Naming Certificate for $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$ packet records compact interval finite-net reasoning as a finite BEDC carrier before any Real and Completion consumer may inspect a completion boundary. It sits after the located interval boundary of chapter 394, the dyadic arithmetic core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names the constructive finite-net surface of a compact interval; it is not open-cover compactness, a quotient stream construction, a selected limit, trichotomy, countable choice, or a bridge claim.

Definition 5208.1 (Compact interval choice-free carrier)**.** *A* $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, D, S, Q, R, T, C, P, N)$$

with the following displayed rows:

1. *E is the located endpoint row. It names the left and right endpoint data through the* $\mathsf{LocatedInterval}^{\uparrow}$ *surface and records the finite comparison windows used to keep the interval ordered.*
2. *D is the dyadic finite mesh row. It lists a finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh, the mesh radius, the endpoint brackets, and the predecessor-successor checks that witness coverage of the displayed interval.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records exactly the finite name coordinates inspected when an interval point, endpoint, or mesh point is read by a downstream consumer.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It records regular rational approximations for the endpoints and mesh witnesses at the displayed tolerance requests.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row. It exposes only the terminal real-name seals obtained from S and Q.*
6. *T records componentwise* hsame *transport for the endpoint row, dyadic mesh row, StreamName windows, RegSeqRat readbacks, Real seals, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay of the finite route*

$$E, D \longmapsto S, Q \longmapsto R, T, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceFree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, D, S, Q, R, T, C, P, N$. It has no coordinate for arbitrary open covers, selected limit points, quotient streams, trichotomy on host reals, countable choice, or a standard bridge.

Lean4 ✓ `BEDC.Derived.CompactIntervalChoiceFreeUp.TasteGate.CompactIntervalChoiceFreeTasteGate single carrier alignment`

Theorem 5208.2 (Compact interval choice-free finite-net obligations)**.** *For every accepted* CompactIntervalChoiceFr *carrier*

$$K = (E, D, S, Q, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceFree}^{\uparrow}}$ *surface consists exactly of carrier admission for located endpoints, the dyadic finite mesh, StreamName windows, RegSeqRat readback, Real seals, componentwise transport, displayed replay, provenance, and local naming. The finite mesh row must witness dyadic coverage of the displayed endpoint interval, endpoint transport must preserve the inspected located rows, and every Real-facing read must pass through the displayed StreamName and RegSeqRat rows before the Real seal is inspected.*

Proof. Project the accepted carrier in Definition 5208.1. Carrier admission is the assertion that the rows $E, D, S, Q, R, T, C, P, N$ are the only rows exposed by the packet.

The dyadic finite-net obligation is read from D. Its endpoint brackets, mesh radius, and predecessor-successor checks give the finite coverage witnesses needed by a compact interval consumer without appealing to an arbitrary open cover.

Endpoint transport is read from T and replayed by C. The componentwise hsame rows identify the displayed located endpoints, mesh data, StreamName windows, RegSeqRat readbacks, and Real seals before any consumer may compare packet outputs.

The ledger is therefore nonescaping. Since the carrier has no coordinate for open-cover compactness, selected limits, quotient streams, trichotomy, countable choice, or bridge data, the listed rows exhaust the local finite-net obligation surface. □

Theorem 5208.3 (Compact interval choice-free Real-completion boundary)**.** *Every Real and Completion consumer of an accepted* $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$ *carrier factors through the dyadic finite mesh row D, the finite StreamName window row S, the regular rational readback row Q, and the terminal Real seal row R. The exported boundary supplies only finite mesh coverage, finite windows, regular readback, and Real seals; it exports no open-cover compactness, selected limit, quotient stream, trichotomy, countable choice, host real equality, or bridge claim.*

Proof. Project the consumer route through the carrier rows of Definition 5208.1. A completion-facing read can inspect endpoint and mesh data only through E and D, and any name coordinate needed for that read must pass through S.

The regular rational interpretation is then supplied by Q, and the terminal real-name boundary is supplied by R. Thus the Real-facing surface is reached by the finite route

$$D, S, Q, R$$

rather than by a quotient stream or a selected completed point.

The structural rows T, C, P, N transport, replay, source, and name this route only. Because no row stores open-cover compactness, countable choice, trichotomy, host equality, or bridge data, the consumer receives exactly the finite Real-completion boundary stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceFree}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceFree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5208.1, Definition 5208.2, and Definition 5208.3 seed the finite compact-interval carrier over located endpoints, dyadic mesh data, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalChoiceFree}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompactIntervalChoiceFreeUp.TasteGate.CompactIntervalChoiceFreeTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limits, quotient streams, trichotomy, countable choice, host real equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic finite-net coverage, endpoint transport, Real-completion boundary routing, provenance, and local naming for BEDC.Derived.CompactIntervalChoiceFreeUp.

Constructive story: A $\mathsf{CompactIntervalChoiceFree}^{\uparrow}$ packet is generated from LocatedInterval, DyadicRatCore, StreamName, RegSeqRat, Real, BHist, hsame, Cont, Pkg, and NameCert rows.

Chapter 5209

A Concrete Naming Certificate for $\mathsf{FiniteMetricEntropy}^{\uparrow}$

Following Kolmogorov and Tikhomirov on metric entropy, and Bishop-style finite nets, the $\mathsf{FiniteMetricEntropy}^{\uparrow}$ packet records a constructive quantitative compactness budget as finite BEDC data. It sits beside the metric source of chapter 57, the finite net surface of chapter 1767, the covering-number row of chapter 2187, the packing-number row of chapter 3761, the rational tolerance row of chapter 37, and the dyadic tolerance ledger of chapter 291. The packet binds covering and packing data to a visible entropy ledger without importing host cardinality, open-cover compactness, selected centers outside the displayed net, or ambient completeness.

Definition 5209.1 (Finite metric entropy carrier)**.** *A* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, \varepsilon, C, Pk, L, H, Cn, Q, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, naming the carrier, distance classifier, and finite metric observations consumed by the entropy packet.*
2. *ε is a positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the radius at which the budget is read.*
3. *C is the finite* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *row, exposing the displayed cover centers and the covering-number upper bound at tolerance ε.*
4. *Pk is the* $\mathsf{PackingNumber}^{\uparrow}$ *lower-bound row, exposing separated witnesses that can be read against the same metric and tolerance data.*
5. *L is the logarithmic or unary entropy-budget ledger, recording the finite budget associated to the covering row and the comparison against the packing row.*
6. *H records componentwise* hsame *transport for the metric source, tolerance, covering, packing, ledger, replay, provenance, and local naming rows.*
7. *Cn records displayed* Cont *replay from a metric-entropy request through $X, \varepsilon, C, Pk, L, H$.*
8. *Q is the* Pkg *provenance over* $\mathsf{FiniteMetricEntropy}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricCoveringNumber}^{\uparrow}$, $\mathsf{PackingNumber}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricEntropy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host cardinality, an infinite open cover, a selected compactness witness, quotient point equality, countable choice, or ambient metric completeness.

Lean4 ✓ `BEDC.Derived.FiniteMetricEntropyUp.FiniteMetricEntropyTasteGate single carrier alignment`

Theorem 5209.2 (Finite metric entropy NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricEntropy}^{\uparrow}$ *carrier*

$$M = (X, \varepsilon, C, Pk, L, H, Cn, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteMetricEntropy}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed metric source, positive tolerance, covering-number row, packing-number row, entropy ledger, transport, replay, provenance, and local naming rows;*
2. *tolerance visibility, requiring every covering or packing read to inspect the same positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* ε*;*
3. *covering exactness, requiring the covering budget* C *to be read through finite* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *centers and the* $\mathsf{MetricCoveringNumber}^{\uparrow}$ *row before the ledger* L *is consumed;*
4. *packing lower-bound visibility, requiring* Pk *to expose only displayed separated witnesses over* X *and* ε*;*
5. *entropy-ledger nonescape, requiring every logarithmic or unary budget read to pass through* C, Pk, L, H, Cn, Q, N *and to avoid host cardinality or compactness.*

No finite metric-entropy obligation is stored outside $X, \varepsilon, C, Pk, L, H, Cn, Q, N$*.*

Proof. Carrier admission is projection from Definition 5209.1. The accepted packet displays the metric source, positive tolerance, covering-number row, packing-number row, entropy ledger, transport, replay, provenance, and local naming rows.

The tolerance and covering obligations follow from the row order. A covering read has no access to centers or a budget until it has inspected X and ε, and the finite net surface is the only route from those rows to C.

The packing and ledger obligations are downstream. The row Pk supplies only displayed separated witnesses over the same metric and tolerance data, while L records the finite budget read from C and compared with Pk.

Finally, H, Cn, Q, N transport, replay, source, and name the same finite route. Since the carrier contains no host-cardinality, open-cover compactness, selected-center, quotient-equality, countable-choice, or ambient-completeness coordinate, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{FiniteMetricEntropy}^{\uparrow}$ consumer that reads an entropy budget without first displaying the metric row, positive tolerance, covering-number row, packing-number row, and ledger rows refutes the carrier surface.

Independence witness: metric, finite-epsilon-net, metric-covering-number, packing-number, rat, dyadicratcore: no listed sibling is bijective with $\mathsf{FiniteMetricEntropy}^{\uparrow}$ because this packet jointly carries metric source data, a positive tolerance row, covering and packing bounds, an entropy-budget ledger, transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricEntropy}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricEntropy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5209.1 and Definition 5209.2 seed the finite entropy-budget route through metric source data, positive tolerance rows, covering numbers, packing lower bounds, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteMetricEntropy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host cardinality, open-cover compactness, selected centers outside the displayed net, quotient point equality, countable choice, Lean verification, bridge checking, or ambient metric completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance visibility, covering exactness, packing lower-bound visibility, and entropy-ledger nonescape for BEDC.Derived.FiniteMetricEntropyUp.

Constructive story: A FiniteMetricEntropy$^{\uparrow}$ packet is generated from MetricUp source data, a positive RatUp or DyadicRatCoreUp tolerance row, finite covering-number data, packing lower-bound data, an entropy-budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5210

A Concrete Naming Certificate for MetaCICCandidateNormalizationRouter$^{\uparrow}$

The MetaCICCandidateNormalizationRouter$^{\uparrow}$ packet realizes the candidate-mediated public route of chapter 9383 as finite closed-term routing data. It sits beside the candidate socket of chapter 3566, the redex frontier of chapter 1008, the bounded normal-equality checker of chapter 752, and the critical-path packet of chapter 765. The packet preserves candidate evidence as a visible row before any strong-normalization or no-infinite-reduction read is allowed.

Definition 5210.1 (MetaCIC candidate normalization router carrier)**.** *A* MetaCICCandidateNormalizationRouter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, A, W, S, M, U, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed term row, recording the empty-context typing and closedness data consumed by the public normalization read.*
2. *A is the candidate-evidence row, naming the supplied reducibility candidate, closed-candidate wrapper, or candidate-set membership witness.*
3. *W is the closed-candidate wrapper row, displaying the closedness side conditions that keep the candidate evidence attached to T.*
4. *S is the candidate-set membership row, used when the route is mediated by a finite candidate-set package rather than a single candidate witness.*
5. *M is the application and Pi adequacy handoff row, recording which adequacy lemma may be consumed before an application or Pi-shaped read reaches normalization evidence.*
6. *U is the closed-candidate strong-normalization and no-infinite-reduction readback row reached only after T, A, W, S, M have been displayed.*
7. *O is the obstruction row refusing any closed-term strong-normalization read that bypasses candidate evidence.*
8. *H records componentwise* hsame *transport for the term, candidate, wrapper, membership, adequacy, readback, obstruction, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the audit-map request through T, A, W, S, M, U, O, H.*

10. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$*, chapter 9383,* BHist*,* hsame*,* Cont*, and* NameCert*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}}$ *row.*

The classifier compares accepted router packets only through these rows. It has no coordinate for evidence-free strong normalization, an unconditional closed-term theorem, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Theorem 5210.2 (MetaCIC candidate normalization router obligations)**.** *For every accepted* $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$ *carrier*

$$R = (T, A, W, S, M, U, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed typed term, candidate evidence, closed-candidate wrapper, candidate-set membership, adequacy handoff, readback, obstruction, transport, replay, provenance, and local naming rows;*
2. *candidate visibility, requiring every strong-normalization or no-infinite-reduction read to inspect A and the relevant wrapper row W or membership row S;*
3. *adequacy handoff exactness, requiring application and Pi consumers to pass through M before using candidate evidence to route a closed term;*
4. *obstruction exactness, requiring O to reject any route from T to U that bypasses candidate, wrapper, membership, or adequacy evidence;*
5. *transport and replay closure, requiring H, C, P, N to preserve the same candidate-mediated route.*

No router obligation is stored outside $T, A, W, S, M, U, O, H, C, P, N$.

Proof. Carrier admission is projection from Definition 5210.1. The accepted packet exposes exactly the closed typed term, candidate, wrapper, membership, adequacy, readback, obstruction, transport, replay, provenance, and local naming rows.

Candidate visibility follows from the audit-map route. The public closed-term read reaches U only through the candidate evidence A together with the displayed closed-candidate wrapper W or candidate-set membership row S.

Application and Pi reads are handled by M. That row records the adequacy handoff before the router treats candidate evidence as applicable to the closed term under inspection.

The obstruction row O records the negative boundary. Since no carrier coordinate supplies evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency, O rejects exactly the bypasses named in the statement.

Finally, H, C, P, N transport, replay, source, and name the same route, so the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: Any $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$ strong-normalization read that reaches a closed typed term without candidate evidence, a closed-candidate wrapper or candidate-set membership row, and the relevant adequacy handoff refutes the packet.

Independence witness: normalization-candidate-socket, metacicredexfrontier, bounded-normal-equality-checker, metaciccriticalpath: no listed sibling is bijective with $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$ because this packet jointly carries closed typed term data, candidate evidence, closed-candidate or candidate-set routing, adequacy handoff, obstruction, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5210.1 and Definition 5210.2 seed the candidate-mediated closed-term normalization route through candidate evidence, closed-candidate wrappers, candidate-set membership, adequacy handoff, obstruction, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close evidence-free strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate visibility, adequacy handoff exactness, obstruction exactness, and transport-replay closure for BEDC.Derived.MetaCICCandidateNormalizationRouterUp.

Constructive story: A $\mathsf{MetaCICCandidateNormalizationRouter}^{\uparrow}$ packet is generated from a closed typed term row, candidate-evidence row, closed-candidate wrapper, candidate-set membership row, application and Pi adequacy handoff, readback row, obstruction row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5211

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5211.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5211.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5211.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5211.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5211.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5212

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5212.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5212.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5212.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5212.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5213

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5213.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5213.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5213.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5214

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5214.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5214.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5214.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5214.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5215

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5215.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 5215.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5215.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5216

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5216.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5216.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5216.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5217

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5217.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5217.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5218

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5218.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5218.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5218.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5219

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5219.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5219.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5219.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5219.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5220

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5220.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5220.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5220.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5221

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5221.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5221.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5221.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5222

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5222.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5222.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5222.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5223

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5223.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5223.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5223.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5223.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5224

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5224.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5224.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5224.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5224.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicVisualMetric}^{\uparrow},\ \mathsf{HyperbolicBusemannCocycle}^{\uparrow},\ \mathsf{HyperbolicGeodesicLaminatio}$$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5225

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5225.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5225.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5225.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5226

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5226.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5226.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5226.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5226.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5226.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5227

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5227.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5227.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5227.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5228

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5228.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5228.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5228.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5228.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5228.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5228.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5229

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5229.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5229.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5229.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5230

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5230.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5230.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5230.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5231

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5231.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5231.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5231.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5231.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5232

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5232.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^\uparrow}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^\uparrow$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5232.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^\uparrow$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^\uparrow}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^\uparrow$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5232.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^\uparrow$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^\uparrow(S),\ \mathsf{DyadicRatCore}^\uparrow(D),\ \mathsf{StreamName}^\uparrow(W),\ \mathsf{RegSeqRat}^\uparrow(R),\ A,\ \mathsf{Real}^\uparrow(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^\uparrow$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^\uparrow$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5233

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5233.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5233.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5233.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5233.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5234

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5234.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5234.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5234.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5234.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5235

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5235.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5235.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5235.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5236

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5236.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5236.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5236.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5237

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5237.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5237.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5237.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5237.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5237.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5238

A Concrete Naming Certificate for $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the $\mathsf{CompactMetric}^{\uparrow}$ source, monotone $\mathsf{ContinuousMap}^{\uparrow}$ family rows, the $\mathsf{DiniTheorem}^{\uparrow}$ compact-window route, $\mathsf{UniformModulus}^{\uparrow}$ handoff data, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance cells, the terminal $\mathsf{Real}^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5238.1 (Dini finite-minimum budget carrier)**.** *A* $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* $\mathsf{DiniTheorem}^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DiniTheorem}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5238.2 (Dini compact finite-minimum budget)**.** *For every accepted* $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a $\mathsf{UniformLimit}^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget sealed by* $\mathsf{Real}^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$ consumer that exports a $\mathsf{UniformModulus}^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite $\mathsf{DyadicRatCore}^{\uparrow}$ budget cells, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniFiniteMinimumBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5239

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5239.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5239.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5239.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5240

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5240.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* NameCert$_{\text{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5240.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5240.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5241

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5241.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5241.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5241.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5242

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5242.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5242.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5242.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5243

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5243.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5243.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5244

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5244.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5244.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5244.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5245

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5245.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5245.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5245.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5246

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5246.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5246.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5246.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5247

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5247.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5247.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5247.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 5248

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5248.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* Real$^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* CauchyCompletionMonadicity$^{\uparrow}$, CauchyCompletionMonad$^{\uparrow}$, CauchyCompletionAlgebr CauchyCompletionKleisli$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, MetricCompletionDenseEmbedding$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5248.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local NameCert$_{\text{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5248.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* CauchyCompletionMonadicity$^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5249

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5249.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5249.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5249.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5250

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5250.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5250.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5250.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5250.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5251

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5251.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *C is the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5251.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5251.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5252

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5252.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5252.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5252.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5253

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5253.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5253.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5253.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5254

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5254.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5254.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5254.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe
carrier

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\text{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5255

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5255.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5255.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5255.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5256

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5256.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5256.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5256.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5257

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 5257.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 5257.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 5257.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5258

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5258.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5258.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5259

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5259.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 5259.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5259.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5260

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 5260.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 5260.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5260.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 5260.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5261

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5261.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5261.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5261.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5261.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5262

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5262.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5262.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5262.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5263

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5263.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5263.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5263.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5264

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5264.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5264.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5264.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5265

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5265.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5265.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5266

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5266.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5266.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5267

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

5267.1 Carrier definition

Definition 5267.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

5267.2 Certificate surface

Theorem 5267.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 5267.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 5267.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 5268

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 5268.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 5268.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 5268.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5269

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5269.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5269.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5269.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5269.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5269.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5270

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5270.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5270.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5270.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5270.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5271

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5271.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5271.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5271.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5272

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5272.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5272.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5272.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5272.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5273

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5273.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 5273.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5273.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5274

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5274.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5274.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5274.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5275

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5275.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5275.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5276

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5276.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5276.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5276.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5277

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5277.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5277.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5277.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5277.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5278

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5278.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5278.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5278.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5279

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5279.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5279.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5279.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5280

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5280.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5280.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5280.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5281

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5281.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5281.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5281.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5281.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5282

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5282.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5282.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5282.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5282.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5283

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5283.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5283.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5283.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5284

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5284.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5284.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5284.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5284.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5284.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5285

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5285.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5285.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5285.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5286

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5286.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5286.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5286.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5286.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5286.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5286.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5287

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5287.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5287.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5287.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5288

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5288.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5288.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5288.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5289

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5289.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5289.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5289.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5289.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5290

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5290.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5290.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5290.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5291

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5291.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopFixedPointInterval}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{IntervalBisection}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5291.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal $\mathsf{Real}^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5291.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* $\mathsf{LocatedInterval}^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz row M, and only then the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{RegSeqRat}^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5291.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5292

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5292.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5292.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5292.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5292.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5293

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5293.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5293.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5293.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5294

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5294.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5294.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5294.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5295

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5295.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5295.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5295.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5295.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5295.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5296

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5296.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5296.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5297

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5297.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5297.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5297.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5298

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5298.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5298.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5298.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??***, and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5299

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5299.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5299.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5299.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5300

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5300.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5300.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5300.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5301

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5301.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5301.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5302

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5302.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5302.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5302.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5303

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5303.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5303.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5303.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5304

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5304.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5304.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5304.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyProductCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5305

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5305.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5305.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5305.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 5306

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 5306.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 5306.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 5306.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 5306.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 5306.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 5306.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 5306.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 5306.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 5307

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5307.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5307.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5307.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5308

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5308.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5308.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5308.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5309

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5309.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5309.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5309.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5309.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5310

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 5310.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 5310.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5310.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 5310.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 5310.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 5310.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 5311

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5311.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5311.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5311.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5312

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5312.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5312.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5312.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5313

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5313.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5313.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5313.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5314

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5314.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5314.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5314.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5315

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5315.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5315.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5315.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5316

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5316.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5316.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5316.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5317

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 5317.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 5317.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 5317.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 5317.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5318

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5318.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5318.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5318.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5318.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5319

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5319.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5319.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5319.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5320

A Concrete Naming Certificate for $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 5320.1 (Located Cauchy filter basis carrier)**.** *A* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* $\mathsf{LocatedCauchyFilter}^{\uparrow}$.
8. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{LocatedCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 5320.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 5320.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 5320.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 5320.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 5321

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5321.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5321.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5321.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5322

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5322.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5322.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5322.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5323

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5323.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5323.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5324

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5324.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5324.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5325

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 5325.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through* X, A, K, W, Q, D, E, F.
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 5325.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier* $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 5326

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 5326.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 5326.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 5327

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 5327.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 5327.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 5327.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 5328

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 5328.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 5328.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 5328.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 5329

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 5329.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 5329.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 5329.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 5329.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 5330

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 5330.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 5330.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 5330.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5331

A Concrete Naming Certificate for RiemannIntegralCauchy$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegralCauchy$^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient Real$^{\uparrow}$ completeness.

Definition 5331.1 (Riemann integral Cauchy carrier)**.** *A* RiemannIntegralCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* RiemannSumGauge$^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* DarbouxSum$^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* RiemannIntegrable$^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* RiemannIntegralCauchy$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 5331.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 5331.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 5332

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 5332.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 5332.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 5332.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 5333

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5333.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate
single carrier alignment

Theorem 5333.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 5333.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 5333.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 5334

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 5334.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 5334.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 5334.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 5334.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 5335

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 5335.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 5335.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 5335.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 5335.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 5335.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 5335.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 5336

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 5336.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* BanachFixedPointStability$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 5336.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5336.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* BanachFixedPointStability$^{\uparrow}$ *packet factors through the finite route*

$$\text{BanachFixedPoint}^{\uparrow} \longmapsto \text{PicardIteration}^{\uparrow} \longmapsto \text{ContractionMapping}^{\uparrow} \longmapsto \text{CompleteMetric}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the RegSeqRat$^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The Real$^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 5336.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 5337

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 5337.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 5337.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 5338

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 5338.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 5338.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 5338.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 5338.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 5339

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 5339.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 5339.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 5339.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 5340

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 5340.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 5341

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 5341.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 5341.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 5342

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 5342.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 5342.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 5342.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 5343

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5343.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 5343.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 5343.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 5343.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 5343.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireSpace$^{\uparrow}$, Bool$^{\uparrow}$, and Nat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteFanTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A FiniteFanTheorem$^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 5344

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 5344.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 5344.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 5344.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5345

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5345.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 5345.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5345.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 5345.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 5346

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 5346.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 5346.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 5347

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5347.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 5347.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5347.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 5347.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 5347.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 5347.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 5348

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 5348.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 5348.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5348.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 5349

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 5349.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 5349.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 5349.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 5350

A Concrete Naming Certificate for $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ packet records the bounded real-sequence subsequence route as finite NameCert data. It sits after the terminal real-name seal of chapter 38, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the stream-window interface of chapter 126, and the located nested-interval limit surface of chapter 5820. The packet is a constructive boundary object for bounded real sequences: it exposes a finite subsequence ledger and a terminal Real seal, not classical sequential compactness, open-cover compactness, countable choice, a host subsequence selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5350.1 (Real Bolzano–Weierstrass carrier)**.** *A* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, D, I, L, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence row, exposing only the finite source coordinates inspected by the packet.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *subdivision ledger, recording the dyadic interval cuts, diameter budgets, and containment checks used at each finite stage.*
3. *I is the finite nested-window row, storing the selected interval windows and their inclusion witnesses.*
4. *L is the subsequence ledger, recording the monotone index schedule produced by the same finite windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the dyadic subdivisions and the subsequence ledger.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, D, I, L, R have been displayed.*
7. *Q is the refusal row, recording that no classical compactness theorem, countable-choice selector, host subsequence, or ambient completeness object is part of the carrier.*
8. *H records componentwise* hsame *transport for the bounded source, dyadic subdivision ledger, nested windows, subsequence ledger, RegSeqRat readback, Real seal, refusal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from bounded-sequence inspection through dyadic subdivision, nested-window refinement, subsequence ledgering, readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, I, L, R, E, Q, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.RealBolzanoWeierstrassUp.TasteGate.RealBolzanoWeierstrassTasteGate single carrier alignment`

Theorem 5350.2 (Real Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, D, I, L, R, E, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded Real sequence admission through S, dyadic interval subdivision through D, finite nested-window discipline through I, subsequence ledger exactness through L, RegSeqRat readback through R, terminal Real sealing through E, refusal of the excluded compactness and choice rows through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5350.1. The rows $S, D, I, L, R, E, Q, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded sequence is read first through S. The dyadic subdivision ledger D and nested-window row I then fix the finite interval route before any subsequence or Real-facing read is available.

The ledger row L records the monotone index schedule selected by those same windows. The readback row R consumes the dyadic and subsequence evidence, and the Real seal E is downstream of that readback.

The refusal and structural rows add no hidden analytic principle. Row Q blocks the excluded compactness, choice, host-selector, and completeness coordinates, while H, C, P, N transport, replay, source, and name exactly the displayed finite packet. □

Theorem 5350.3 (Real subsequence readback route)**.** *Every* $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ *consumer that exports a cluster-facing real read must factor through the finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{LocatedNestedIntervalLimit}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported read is supported by bounded source rows, dyadic interval subdivision, finite nested windows, a subsequence ledger, RegSeqRat readback, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence.*

Proof. Apply Definition 5350.2. It places bounded source admission, dyadic subdivisions, nested windows, subsequence ledgering, and regular rational readback before the terminal Real seal can be consumed.

The source and target occurrences of $\mathsf{Real}^{\uparrow}$ are not quotient representatives. The first supplies bounded sequence names through finite StreamName windows, while the second is the terminal seal reached after the dyadic and regular-rational route has been displayed.

Transport and replay preserve the same packet. Thus any cluster-facing read follows the stated finite route and cannot bypass it through classical sequential compactness, open-cover compactness, countable choice, a host subsequence selector, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ cluster-facing read factors through one bounded RealUp sequence row, one DyadicRatCore subdivision ledger, one finite nested-window row, one subsequence ledger, one RegSeqRat readback row, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, dyadicratcore, streamname, located-nested-interval-limit: $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries bounded real-sequence rows, dyadic interval subdivision, finite nested windows, a subsequence ledger, regular rational readback, and a terminal Real seal for cluster-facing consumers.

Closure status for $\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5350.1, Definition 5350.2, and Definition 5350.3 seed the bounded real-sequence subsequence route over dyadic subdivision, finite nested windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical sequential compactness, open-cover compactness, countable choice, host subsequence selection, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded sequence visibility, dyadic interval subdivision, finite nested-window exactness, subsequence-ledger exactness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealBolzanoWeierstrassUp.

Constructive story: A $\mathsf{RealBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence row, a DyadicRatCore subdivision ledger, finite nested windows, a subsequence ledger, RegSeqRat readback, a terminal RealUp seal, a refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5351

A Concrete Naming Certificate for $\mathsf{CandidateSetClosedRoute}^{\uparrow}$

The $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ packet realizes the candidate-set branch of chapter 9383 as finite NameCert data. It records the route by which closed typed MetaCIC terms inherit strong normalization from candidate-set membership while preserving the candidate evidence as an explicit row. The route consumes the positive closed-normal consistency surface of chapter 736 only as an obstruction-facing boundary: before the obstruction row O can refuse unconditional consistency, the closed-normal packet has already fixed the normal-fragment node that is not being promoted to full closed consistency. The packet is a normalization-audit surface, not a blanket theorem that every closed typed term is strongly normalizing without candidate-mediated evidence.

The route consumes the sibling candidate-set normalization surface of chapter 2320 and the MetaCIC normalization frontier route of chapter 892 before it exports closed typed term strong-normalization reads. $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ isolates the closed-term endpoint, while $\mathsf{CandidateSetNormalization}^{\uparrow}$ supplies the candidate-set evidence and filter route that make the endpoint readable.

Definition 5351.1 (Candidate-set closed route carrier)**.** *A* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, M, S, A, B, NF, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed term row, including the empty-context typing and closedness data consumed by the MetaCIC closed-term projection.*
2. *M is the candidate-set membership row, storing the displayed membership witness rather than erasing it into an unconditional endpoint.*
3. *S is the candidate-set soundness row, exposing the route from membership to strong normalization.*
4. *A is the application and Pi adequacy row, recording the candidate-mediated handoff used by function and argument reads.*
5. *B is the beta-closure row, keeping the candidate-set evidence stable under the displayed one-step reduction route.*
6. *NF is the no-infinite-reduction projection row downstream of the same strong-normalization evidence.*

7. *O is the obstruction row naming closed-normal consistency, substitution-audit, confluence-audit, and subject-reduction boundaries that remain outside this packet.*
8. *H records componentwise* hsame *transport for closed typed terms, candidate-set membership, soundness, adequacy, beta closure, no-infinite-reduction projection, obstruction, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from candidate-set membership through closed typed term strong normalization and no-infinite-reduction reads.*
10. *P is the* Pkg *provenance over* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$*, the MetaCIC normalization audit map, closed-normal consistency, substitution-audit map, confluence-audit map, subject-reduction obstruction,* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, S, A, B, NF, O, H, C, P, N$.

Theorem 5351.2 (Candidate-set closed route NameCert obligations)**.** *For every accepted* CandidateSetClosedRoute *carrier*

$$R = (T, M, S, A, B, NF, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$ *surface consists exactly of closed typed term admission through T, candidate-set membership through M, candidate-set soundness through S, application and Pi adequacy through A, beta-closure stability through B, no-infinite-reduction projection through NF, obstruction visibility through O, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5351.1. The closed typed term row, candidate-set membership row, soundness row, adequacy row, beta-closure row, no-infinite-reduction row, obstruction row, and structural rows are the only exposed coordinates.

The membership row M is not optional evidence. It is the datum consumed by S to route the closed typed term in T to strong normalization, and the adequacy row A keeps application and Pi reads inside the same candidate-mediated interface.

Rows B and NF preserve the endpoint discipline. Beta closure stays attached to the candidate set, while no-infinite-reduction is projected downstream of the same strong-normalization route.

Finally O, H, C, P, N record the boundary, transport, replay, source, and local name. Since no unconditional closed-term strong-normalization coordinate is present, the listed obligations exhaust the packet. □

Theorem 5351.3 (Candidate-set closed strong-normalization route)**.** *Every* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *strong-normalization read factors through candidate-set membership evidence before it reaches the closed typed term endpoint. The route preserves the candidate-set soundness row, beta-closure row, application and Pi adequacy row, no-infinite-reduction projection, obstruction ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Proof. Apply Definition 5351.2. The closed typed term row T supplies the endpoint shape, but it does not itself supply strong normalization.

The route obtains strong normalization only after projecting the membership row M and soundness row S. Application and Pi reads remain governed by A, and beta steps remain governed by B.

The no-infinite-reduction projection NF is downstream of the same evidence. The obstruction and structural rows preserve the conditional reading, so the exported endpoint remains candidate-mediated rather than unconditional. □

Theorem 5351.4 (Candidate-set route TasteGate obligation)**.** *Future Lean support for* $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *must keep the candidate-set membership route nontrivial and field-faithful. A* Nontrivial $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *instance must distinguish an accepted packet with displayed candidate-set membership from a closed typed term row without that membership evidence, and a* FieldFaithful $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ *instance must preserve the row order*

$$T \longmapsto M \longmapsto S \longmapsto A \longmapsto B \longmapsto NF \longmapsto O \longmapsto H, C, P, N$$

before any TasteGate carrier-alignment theorem can serve as a checked realization.

Proof. Nontriviality is forced by the carrier boundary: a closed typed term row without M cannot export the same strong-normalization read as an accepted packet with candidate-set membership evidence.

Field faithfulness is the row-order requirement. If a gate projection permutes or erases $M, S, A, B, NF, O, H, C, P$ the candidate-mediated route is replaced by a detached endpoint claim, which is not the packet named by Definition 5351.1. □

Falsifiable prediction: Every accepted $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ strong-normalization read names the candidate-set-to-closed-SN route: closed typed term data first, candidate-set membership evidence second, candidate-set soundness third, application and Pi adequacy fourth, beta closure fifth, no-infinite-reduction projection sixth, and obstruction, transport, replay, provenance, and local naming rows thereafter.

Independence witness: metacic-normalization-audit-map, closed-normal-consistency, substitution-audit-map, confluence-audit-map, subject-reduction-obstruction: $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ is not bijective to any listed map because it isolates the candidate-set membership route from closed typed terms to strong normalization without erasing the membership evidence or closing the subject-reduction, substitution, and confluence boundaries.

Closure status for $\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5351.1, Definition 5351.2, Definition 5351.3, and Definition 5351.4 seed the candidate-set-mediated closed typed term strong-normalization route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CandidateSetClosedRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization for every closed typed MetaCIC term, subject reduction, arbitrary substitution preservation, global confluence, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-set membership visibility, candidate-set soundness, application and Pi adequacy routing, beta-closure stability, no-infinite-reduction projection, obstruction visibility, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.CandidateSetClosedRouteUp

Constructive story: A $\mathsf{CandidateSetClosedRoute}^{\uparrow}$ packet is generated from closed typed term data, candidate-set membership evidence, candidate-set soundness, application and Pi adequacy rows, beta closure, no-infinite-reduction projection, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5352

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 5352.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 5352.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 5352.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 5352.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 5353

A Concrete Naming Certificate for $\mathsf{BisectionReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BisectionReal}^{\uparrow}$ packet records bisection as a finite nested-interval and bracket-refinement certificate over the L10 Real route. It sits after the dyadic tolerance surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested interval route of chapter 1924. The packet names the displayed bisection ledger; it is not a Dedekind cut, quotient real, trichotomy principle, selected root, or completeness theorem.

Definition 5353.1 (BisectionReal carrier)**.** *A* $\mathsf{BisectionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, S, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval source row, recording the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, interval containment, and the finite retained subintervals consumed by the bisection route.*
2. *M is the dyadic midpoint and tolerance ledger. It records the midpoint rows, branch radii, diameter budgets, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons used at each inspected stage.*
3. *S is the sign or bracket-preservation row, storing the displayed bracket invariant that survives each retained child interval.*
4. *T is the tail request row, recording the finite precision request and the inspected bisection depth needed to answer it.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained endpoint and midpoint names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from W and M.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the real name exported by the retained nested interval route.*
8. *H records componentwise* hsame *transport for the interval source, midpoint ledger, bracket row, tail request, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through interval inspection, midpoint production, bracket preservation, dyadic tolerance readback, stream-window exposure, regular readback, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{BisectionReal}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, S, T, W, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient-real representatives, real trichotomy, selected roots, host equality of real points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BisectionRealUp.BisectionRealTasteGate single carrier alignment`

Theorem 5353.2 (BisectionReal NameCert obligations)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier*

$$B = (I, M, S, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval source, dyadic midpoint and tolerance ledger, bracket-preservation row, tail request, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *nested-source discipline, requiring every retained interval read to pass through I before midpoint or Real-facing data are available;*
3. *midpoint and tolerance discipline, requiring M to expose only dyadic midpoint, radius, and diameter rows whose budgets answer the request T;*
4. *bracket preservation, requiring S to carry the displayed sign or bracket invariant across each retained child interval;*
5. *readback discipline, requiring W and R to expose the retained interval only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
6. *seal nonescape, requiring E, H, C, P, N to export only the displayed bisection route and no Dedekind cut, quotient real, trichotomy principle, selected root, host equality, countable choice, or completeness theorem.*

No BisectionReal obligation is stored outside these rows.

Proof. Project the carrier of Definition 5711.1. The accepted packet displays exactly the nested interval source, midpoint ledger, bracket row, tail request, window rows, readback, Real seal, and structural rows.

The finite source stage is I, T. A precision request opens the source interval row and the requested bisection depth before any midpoint or retained endpoint can be inspected.

The bisection stage is M, S. A finite induction over the retained branch ledger reads one midpoint row at a time, uses the dyadic tolerance entries to bound the retained child, and carries the displayed sign or bracket invariant into the next interval row.

The readback stage is W, R, E. The retained endpoints are exposed as finite stream windows, then as regular rational readbacks, and only then as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, package, and name the same route. Since the carrier contains no Dedekind-cut, quotient-real, trichotomy, selected-root, host-equality, countable-choice, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5353.3 (BisectionReal nested interval factorization)**.** *Every* $\mathsf{BisectionReal}^{\uparrow}$ *read factors through the displayed nested interval route*

$$I, T \longmapsto M, S \longmapsto W, R \longmapsto E, H, C, P, N.$$

In particular, a Real-facing consumer first reads the $\mathsf{NestedIntervalReal}^{\uparrow}$ *source row* I*, then the finite dyadic midpoint and bracket-preservation rows* M, S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* W, R*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal and local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

Proof. Start with the accepted carrier $B = (I, M, S, T, W, R, E, H, C, P, N)$. The only interval source coordinate is I, and the only request coordinate is T, so a bisection read begins by exposing those rows.

Induct on the finite branch prefix recorded in the midpoint ledger. The base case reads the initial interval and its bracket row. In the successor step, the midpoint row in M supplies the dyadic split, the tolerance entry supplies the diameter budget, and S records that the displayed bracket invariant is preserved for the retained child.

After the last retained child, the endpoint windows are read through W and the rational data through R. The Real-facing seal E is therefore downstream from the nested interval, midpoint, bracket, stream, and regular readback rows.

The structural rows H, C, P, N do not add a second route. They transport componentwise hsame, replay the same displayed Cont chain, record Pkg provenance, and supply the local name. Thus every read factors through the stated finite route. □

Theorem 5353.4 (BisectionReal regular Cauchy handoff)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier* $B = (I, M, S, T, W, R, E, H, C, P, N)$*, the handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is available only after the retained nested-interval rows, dyadic midpoint and tolerance rows, sign or bracket-preservation rows, and finite* $\mathsf{StreamName}^{\uparrow}$ *windows have been replayed through the same* BHist *packet. The exported object is a regular Cauchy readback with a Real seal, not a selected root, quotient representative, trichotomy decision, or completeness witness.*

Proof. Use the nested interval factorization of Definition 5711.3. It places W, R, E strictly after the source row I, request row T, midpoint ledger M, and bracket row S.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the row R. It is computed from the finite stream windows in W and the dyadic tolerance entries in M, so no downstream consumer can read R without replaying the retained bisection ledger.

The $\mathsf{Real}^{\uparrow}$ handoff is the row E. It seals the regular readback already obtained from R, while H, C, P, N preserve the same packet by componentwise hsame, displayed Cont, Pkg, and local NameCert data.

Since the carrier has no selected-root, quotient-representative, trichotomy-decision, host-equality, countable-choice, Dedekind-cut, or completeness coordinate, the handoff exposes only the regular Cauchy route stated in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionReal}^{\uparrow}$ packet is generated from a nested interval source row, dyadic midpoint and tolerance ledger, sign or bracket-preservation rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, seed)

Scope closed: Definition 5711.1, Definition 5711.2, Definition 5711.3, and Definition 5711.4 seed the finite constructive bisection route from nested interval rows through dyadic midpoint, bracket-preservation, stream-window, regular readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, trichotomy on reals, selected roots, host equality of real points, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source discipline, midpoint and tolerance exactness, bracket preservation, regular Cauchy handoff, RealUp seal nonescape, and local naming for BEDC.Derived.BisectionRealUp.

Chapter 5354

A Concrete Naming Certificate for $\mathsf{DarbouxCriterion}^{\uparrow}$

Following Darboux, Riemann, and Bishop–Bridges constructive integration, the $\mathsf{DarbouxCriterion}^{\uparrow}$ packet records the upper-lower gap test as finite NameCert evidence. It sits between the lower and upper Darboux sum packets of chapter 5858 and chapter 4083, the Darboux-sum surface of chapter 1880, and the Riemann-facing consumers of chapter 1960 and chapter 1526. The packet exposes a displayed partition, lower sum, upper sum, gap, tolerance, regular rational readback, and real seal; it does not import host suprema, measure completion, a selected mesh, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5354.1 (Darboux criterion carrier)**.** *A* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, L, U, G, T, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition row, recording the inspected interval endpoints and refinement requests.*
2. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row attached to P.*
3. *U is the* ***??*** *row attached to the same partition row P.*
4. *G is the upper-minus-lower gap row comparing U and L inside the displayed rational ledger.*
5. *T is the tolerance row, recording the positive rational request against which G is checked.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same finite partition and gap evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after P, L, U, G, T, R have been displayed.*
8. *H records componentwise* hsame *transport for partition, lower-sum, upper-sum, gap, tolerance, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through P, L, U, G, T, R, E.*
10. *Q is the* Pkg *provenance over* $\mathsf{DarbouxCriterion}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, ***??***, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host suprema or infima, measure completion, a selected mesh, an ambient completeness principle, a quotient real, or a global integration theorem.

Theorem 5354.2 (Darboux criterion NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier*

$$K = (P, L, U, G, T, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, partition exposure, lower-sum and upper-sum compatibility over the same partition, gap comparison against the tolerance row, regular rational readback, terminal real-seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $P, L, U, G, T, R, E, H, C, Q, N$.

Lean4 ✓ `BEDC.Derived.DarbouxCriterionUp.DarbouxCriterionTasteGate single carrier alignment`

Proof. Project the carrier. The accepted packet exposes the partition row P, the paired lower and upper Darboux sum rows L, U, the gap row G, the tolerance row T, the regular readback row R, the seal row E, and the structural rows H, C, Q, N.

The lower and upper sum rows are tied to the same partition row, so a gap comparison cannot switch partitions while the criterion is being read. The tolerance row fixes the request against which the displayed gap is checked.

The regular readback and real seal are downstream of those finite rows. Componentwise transport and replay preserve the same order, while provenance and naming record the listed sibling packets.

Since host suprema, measure completion, selected meshes, quotient reals, ambient completeness, and global integration theorems are absent from the carrier coordinates, the displayed rows exhaust the local NameCert surface. □

Theorem 5354.3 (Darboux criterion upper-lower gap route)**.** *Every* $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *or regulated-integral-facing read of an accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *packet factors through the finite route*

$$P, L, U \longmapsto G, T \longmapsto R \longmapsto E.$$

Thus the consumer first reads a displayed partition, lower Darboux sum, and upper Darboux sum; then it reads the upper-minus-lower gap and tolerance rows; then it reads the $\mathsf{RegSeqRat}^{\uparrow}$ *hand-off and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The packet exports no direct route through host suprema, measure completion, selected mesh choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 5712.2. They require carrier admission and classifier transport through the displayed rows of Definition 5712.1.

The partition row P is read before either sum row can be used. The rows L and U are therefore paired over the same finite partition, and G is the displayed comparison between those paired rows.

The tolerance row T is the only request against which the gap is consumed. After that finite comparison has been displayed, the read may pass to R and then to the real seal row E.

The excluded routes are not coordinates of the carrier. Hence every integration-facing consumer receives the upper-lower gap criterion only through the finite partition, sum, gap, tolerance, regular-readback, and seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxCriterion}^{\uparrow}$ packet is generated from a displayed partition, lower and upper Darboux sum rows, an upper-minus-lower gap row, a tolerance row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5712.1, Definition 5712.2, and Definition 5712.3 seed the finite upper-lower gap criterion for Riemann-facing and regulated-integral-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DarbouxCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, measure completion, selected mesh choice, quotient real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, paired lower and upper sum compatibility, gap tolerance comparison, regular readback, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxCriterionUp.

Chapter 5355

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 5355.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 5355.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 5355.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 5356

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 5356.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. *N is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. *M is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. *F is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. *H records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-uniform tolerance request through $K, N, M, F, U, T, H, P, L$.*
9. *P is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 5356.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 5356.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 5357

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 5357.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 5357.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 5357.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 5357.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 5357.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* $\mathsf{RegularCauchyHeineBorelR}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 5357.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 5358

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 5358.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 5358.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 5358.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 5359

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5359.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 5359.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5359.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 5360

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 5360.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 5360.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 5360.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 5360.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 5360.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 5360.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 5361

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 5361.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5361.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 5361.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 5361.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 5362

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 5362.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 5362.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 5362.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 5362.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 5363

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5363.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5363.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 5363.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 5363.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 5363.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 5363.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 5363.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 5363.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 5363.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 5363.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 5363.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 5363.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 5363.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 5363.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 5363.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 5364

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 5364.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5364.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 5364.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, seed)

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 5364.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 5364.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 5364.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 5364.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 5364.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 5364.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 5364.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 5365

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 5365.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 5365.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 5365.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 5365.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 5366

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 5366.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 5366.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 5366.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let* $L = (S, Q, I, U, F, K, T, H$ *be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

$\mathsf{RieszThorinInterpolation}^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 5367

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 5367.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5367.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5367.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 5367.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 5368

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 5368.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 5368.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 5368.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 5368.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 5369

A Concrete Naming Certificate for $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream $\mathsf{Real}^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5369.1 (Regular Cauchy lattice operations carrier)**.** *A* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* $\mathsf{Real}^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *lower-member row.*
5. *A is the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$*,* $\mathsf{RegularCauchyMinimum}^{\uparrow}$*,* $\mathsf{RegularCauchyMaximu}$ $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 5369.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 5369.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5370

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5370.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 5370.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5370.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 5370.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 5371

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 5371.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 5371.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 5371.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 5372

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 5372.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 5372.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5372.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 5373

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5373.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5373.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5373.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5373.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5373.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5374

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5374.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5374.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5374.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5374.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5375

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5375.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5375.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5375.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5376

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5376.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5376.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5376.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5376.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5377

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5377.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 5377.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5377.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5378

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5378.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5378.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5378.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5379

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5379.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5379.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5380

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5380.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5380.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5380.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5381

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5381.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5381.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5381.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5381.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5382

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5382.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5382.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5382.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5383

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5383.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5383.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5383.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5384

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5384.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5384.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5384.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5385

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5385.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5385.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5385.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5385.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5386

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5386.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5386.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5386.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5386.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5387

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5387.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5387.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5387.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5388

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5388.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5388.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5388.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5388.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5388.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5389

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5389.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5389.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5389.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5390

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5390.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5390.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5390.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5390.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5390.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5390.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5391

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5391.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5391.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5391.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5392

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5392.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5392.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5392.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5393

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5393.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5393.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5393.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5393.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5394

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5394.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5394.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5394.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5395

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5395.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5395.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5395.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5395.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5396

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5396.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5396.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5396.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5396.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5397

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5397.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5397.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5397.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5398

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5398.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5398.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5398.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5399

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5399.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5399.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5399.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N.$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5399.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5399.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5400

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5400.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5400.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5401

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5401.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. *T is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after W, T, D, R have been exposed.*
6. *H records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. *C records displayed* Cont *replay of the same finite filter route.*
8. *P records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5401.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5401.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5402

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5402.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5402.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5402.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5403

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5403.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5403.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5403.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5404

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5404.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5404.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5404.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5405

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5405.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5405.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5406

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5406.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5406.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5406.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5407

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5407.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5407.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5407.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5408

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5408.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5408.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5408.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5409

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5409.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5409.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5409.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 5410

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5410.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5410.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5410.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5411

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5411.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5411.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier $U = (S, A, F, D, C, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5411.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5412

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5412.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5412.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5412.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5412.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5413

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5413.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5413.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5413.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5414

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5414.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5414.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5414.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5415

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5415.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5415.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5415.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5416

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5416.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate single carrier alignment`

Theorem 5416.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5416.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5417

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5417.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5417.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5417.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5418

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5418.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5418.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5418.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let* $I = (B, D, W, R, E, H, C, P, N)$ *be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by* I *factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5419

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 5419.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 5419.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 5419.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5420

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5420.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5420.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5421

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5421.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 5421.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5421.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5422

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 5422.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 5422.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5422.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 5422.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5423

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5423.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5423.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5423.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5423.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5424

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5424.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5424.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5424.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5425

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5425.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5425.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5425.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5426

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5426.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5426.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5426.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5427

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5427.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5427.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5428

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5428.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5428.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5429

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

5429.1 Carrier definition

Definition 5429.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

5429.2 Certificate surface

Theorem 5429.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 5429.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 5429.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 5430

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 5430.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 5430.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 5430.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5431

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5431.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5431.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5431.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5431.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5431.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5432

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5432.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5432.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5432.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5432.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5433

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5433.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5433.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5433.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5434

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5434.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5434.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5434.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5434.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5435

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5435.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 5435.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5435.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5436

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5436.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5436.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5436.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5437

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5437.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5437.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5438

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5438.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5438.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5438.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5439

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5439.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5439.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5439.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5439.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5440

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5440.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5440.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5440.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5441

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5441.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5441.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5441.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5442

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5442.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 5442.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5442.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5443

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5443.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5443.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5443.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5443.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5444

A Concrete Naming Certificate for $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$

The $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5444.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* $\mathsf{UnitDisk}^{\uparrow}$ *coordinates and the associated* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5444.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5444.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5444.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$$\mathsf{UnitDisk}^{\uparrow},\ \mathsf{HyperbolicMetric}^{\uparrow},\ \mathsf{HyperbolicVisualMetric}^{\uparrow},\ \mathsf{HyperbolicBusemannCocycle}^{\uparrow},\ \mathsf{HyperbolicGeodesicLaminatio}$$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5445

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5445.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through $K, L, P, T, E, A, C, H, G, Q, N$.*
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5445.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5445.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5446

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5446.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5446.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5446.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5446.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5446.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5447

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5447.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5447.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5447.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5448

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5448.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5448.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5448.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5448.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5448.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5448.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5449

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5449.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5449.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5449.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5450

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5450.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5450.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5450.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5451

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5451.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5451.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5451.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5451.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5452

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5452.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5452.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5452.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5453

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5453.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5453.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5453.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5453.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5454

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5454.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5454.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5454.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5454.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5455

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5455.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5455.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5455.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5456

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5456.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5456.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5456.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5457

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5457.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5457.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5457.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5457.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5457.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5458

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5458.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5458.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5459

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5459.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5459.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5459.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5460

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5460.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5460.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5460.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5461

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5461.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5461.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5461.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$, seed)

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5462

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5462.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5462.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5462.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5463

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5463.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5463.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5464

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5464.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5464.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5464.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5465

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5465.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5465.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5465.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5466

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5466.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5466.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5466.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5467

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5467.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5467.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5467.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 5468

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 5468.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 5468.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 5468.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 5468.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 5468.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 5468.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 5468.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 5468.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 5469

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5469.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5469.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5469.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5470

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5470.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5470.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5470.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, seed)

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5471

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5471.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5471.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5471.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5471.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5472

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 5472.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 5472.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5472.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 5472.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 5472.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 5472.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 5473

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5473.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *C is the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5473.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5473.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5474

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5474.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5474.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5474.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5475

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5475.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5475.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5475.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5476

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5476.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate single carrier alignment`

Theorem 5476.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5476.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5477

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5477.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5477.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5477.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5478

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5478.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5478.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5478.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5479

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 5479.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 5479.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 5479.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 5479.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5480

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5480.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5480.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*
2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*
3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*
4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*
5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*
6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*
7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5480.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5480.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5481

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5481.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5481.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5481.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5482

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 5482.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 5482.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 5482.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 5482.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 5482.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 5483

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5483.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5483.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5483.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5484

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5484.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5484.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5484.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5485

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5485.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5485.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5486

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5486.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5486.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5487

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 5487.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 5487.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 5488

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 5488.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 5488.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 5489

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 5489.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 5489.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 5489.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 5490

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 5490.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 5490.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 5490.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 5491

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 5491.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 5491.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 5491.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 5491.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 5492

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 5492.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 5492.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 5492.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5493

A Concrete Naming Certificate for $\mathsf{RiemannIntegralCauchy}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5493.1 (Riemann integral Cauchy carrier)**.** *A* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 5493.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 5493.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 5494

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 5494.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 5494.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 5494.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 5495

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5495.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate
single carrier alignment

Theorem 5495.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 5495.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 5495.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 5496

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 5496.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 5496.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 5496.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 5496.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 5497

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 5497.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 5497.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 5497.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 5497.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 5497.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 5497.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 5498

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 5498.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BanachFixedPointStability}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 5498.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5498.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BanachFixedPoint}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the $\mathsf{RegSeqRat}^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The $\mathsf{Real}^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 5498.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 5499

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 5499.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 5499.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 5500

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 5500.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 5500.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 5500.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 5500.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 5501

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 5501.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 5501.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 5501.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 5502

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 5502.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 5503

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 5503.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 5503.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 5504

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 5504.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 5504.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 5504.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 5505

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5505.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 5505.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 5505.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 5505.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 5505.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireSpace$^{\uparrow}$, Bool$^{\uparrow}$, and Nat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteFanTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A FiniteFanTheorem$^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 5506

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 5506.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 5506.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 5506.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5507

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5507.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 5507.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5507.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 5507.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 5508

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 5508.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 5508.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 5509

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5509.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 5509.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5509.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 5509.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 5509.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 5509.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 5510

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 5510.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 5510.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5510.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 5511

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 5511.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 5511.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 5511.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 5512

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 5512.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 5512.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 5512.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 5512.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 5513

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 5513.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 5513.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 5513.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 5514

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 5514.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. K *is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. N *is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. M *is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. F *is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. U *is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. T *is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. H *records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a compact-uniform tolerance request through* $K, N, M, F, U, T, H, P, L$.
9. P *is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. *L is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 5514.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 5514.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 5515

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5515.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5515.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5515.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5515.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The sandwich hub routes its cofinal common-tail read through chapter 7532 before the child files expose the lower and upper RegSeqRat readbacks.

Chapter 5516

A Concrete Naming Certificate for $\mathsf{RegularCauchySandwich}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet names the two-sided bracketing surface used when regular Cauchy data is read by a real-completion consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the tail agreement boundary of chapter 1738. The packet records lower and upper regular tails as finite BEDC rows; it is not a quotient stream, a selected limit, a trichotomy principle, or an ambient completeness theorem.

Definition 5516.1 (Regular Cauchy sandwich carrier)**.** *A* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *U is the upper regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
3. *T is the common tail-agreement row inherited from* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling both bracket sides at the same precision request.*
5. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the lower and upper sides.*
6. *R_L and R_U are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed after the windows are displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both sides, the common tail row, the dyadic ledger, the windows, and the readbacks have been consumed.*
8. *H records componentwise* hsame *transport for the lower side, upper side, tail row, ledger, windows, readbacks, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each finite precision request through*

$$L, U, T, D, W_L, W_U, R_L, R_U \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySandwich}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the lower and upper regular rows, common tail agreement, dyadic tolerance ledger, finite windows, readbacks, and terminal seal belong to the same displayed bracketing route. It has no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5516.2 (Regular Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *surface consists exactly of lower regular-row exposure, upper regular-row exposure, common tail-agreement exposure, shared dyadic tolerance accounting, lower and upper finite-window exposure, lower and upper RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through N.*

Proof. Project the carrier tuple in Definition 5874.1. The lower and upper sides are displayed as L and U, and the only row that binds them is the common tail-agreement coordinate T.

The shared ledger D fixes the precision request for both sides. The finite windows W_L, W_U are opened against that same ledger, and the readback rows R_L, R_U expose exactly the inspected regular rational data.

The seal row E is downstream of those rows, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic coordinate occurs in the carrier, the local NameCert surface is exactly the displayed bracketing route. □

Theorem 5516.3 (Regular Cauchy sandwich window bracketing)**.** *Let*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. Every finite precision request consumed by the packet reads the two-sided route*

$$L, U, T, D \longmapsto W_L, W_U \longmapsto R_L, R_U \longmapsto E, H, P, N.$$

Thus lower and upper regular windows are bracketed by one displayed tolerance ledger and one common tail-agreement row, not by quotient representatives, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient completeness.

Proof. Start with a finite precision request. The packet exposes only one dyadic tolerance ledger D, so both sides of the bracket are read against the same displayed tolerance data.

The lower and upper windows W_L, W_U are then opened after the common tail row T has been displayed. Their readbacks R_L, R_U remain attached to the same packet, so neither side can be replaced by an external quotient stream or a selected representative.

The terminal seal E is reached only after both readbacks. Transport, replay, provenance, and local naming through H, C, P, N preserve that order for every accepted request, which gives the claimed two-sided bracketing route. □

Theorem 5516.4 (Regular Cauchy sandwich Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier can bypass the displayed route through the lower and upper regular rows, common tail agreement, shared dyadic ledger, lower and upper StreamName windows,*

lower and upper RegSeqRat readbacks, and local structural rows. Equivalently, every accepted terminal seal read is attached to

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

and cannot be separated from those rows by quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 5874.2. It exposes the Real seal row E only as part of the local NameCert surface containing the two regular rows, common tail row, ledger, windows, readbacks, and structural rows.

Now use Definition 5874.3. Every precision request reaches the readbacks through the same two-sided route, so a seal read detached from one side of the bracket is not a read of this packet.

Finally inspect Definition 5874.1. The carrier contains no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient Real completeness. A purported seal read using such data therefore lies outside the accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ carrier. □

Theorem 5516.5 (Regular Cauchy sandwich Cantor intersection seal)**.** *Let* $S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H$ *be an accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. If the lower and upper rows are supplied by a nested-shrink or Cantor-intersection route, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed by* S *is available only through the displayed sandwich route, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. In particular, the consumer-facing seal factors through Definition 5874.3, Definition 5874.4, chapter 4227, and chapter 7754, without asserting a selected intersection point.*

Proof. Apply Definition 5874.3. It forces every finite precision request for S through the common tail row, shared dyadic ledger, two StreamName windows, and two RegSeqRat readbacks before the seal row is visible.

The nested-shrink and Cantor-intersection inputs supply the lower and upper rows only as finite interval or shrink data. Those routes provide the source rows for the sandwich, but they do not add a coordinate for a selected point, quotient stream equality, or ambient completeness.

Now apply Definition 5874.4. Any terminal $\mathsf{Real}^{\uparrow}$ read detached from the displayed carrier is outside the accepted packet, so the Cantor-facing consumer receives only the finite sandwich seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet is generated from lower and upper regular-Cauchy rows, a common tail-agreement row, a shared dyadic tolerance ledger, lower and upper StreamName windows, lower and upper RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, seed)

Scope closed: Definition 5874.1, Definition 5874.2, Definition 5874.3, and Definition 5874.4 seed the finite two-sided regular-Cauchy bracketing route over DyadicRatCore, StreamName, RegSeqRat, Real, and CauchyTailEquivalence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common tail exactness, shared dyadic tolerance accounting, two-sided StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySandwichUp.

Theorem 5516.6 (Regular Cauchy sandwich sibling dependency route)**.** *Every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *sandwich packet depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows of chapter 301 and the* $\mathsf{StreamName}^{\uparrow}$ *pointwise window rows of chapter 126 before it exposes a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route consumes those siblings only through the displayed lower and upper windows, lower and upper regular readbacks, common tail row, shared dyadic ledger, transport, replay, provenance, and local naming rows of Definition 5874.1.*

Proof. Read the carrier rows from Definition 5874.1. The lower and upper windows W_L, W_U are the pointwise $\mathsf{StreamName}^{\uparrow}$ windows, and the rows R_L, R_U are the regular rational readbacks supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

The bracketing theorem Definition 5874.3 orders every precision request through those windows and readbacks before the seal row is consumed. The Real-seal nonescape theorem Definition 5874.4 prevents bypass through quotient stream equality, selected limits, host real equality, or ambient completeness.

Thus the chapter's sibling dependency is the concrete route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, not an external analytic carrier. □

Theorem 5516.7 (Regular Cauchy sandwich cofinal tail-mesh dependency)**.** *Every accepted* $\mathsf{RegularCauchySandwic}$ *packet whose common tail row is presented cofinally must read the* $\mathsf{CofinalTailMesh}^{\uparrow}$ *route of chapter 7532 before the lower and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are consumed. The dependency is the displayed route through the sandwich carrier, the window bracketing theorem, and the sibling dependency route; it is not an appeal to quotient streams, selected limits, trichotomy, choice, host real equality, or ambient completeness.*

Proof. Read the carrier rows in Definition 5874.1. The common tail coordinate T is displayed before the lower and upper StreamName windows, the lower and upper RegSeqRat readbacks, and the Real seal row.

When T is presented cofinally, consume the CofinalTailMesh route of chapter 7532. That route supplies the shared finite tail-window mesh before any Real or completion-facing seal is read.

Apply the window bracketing theorem Definition 5874.3. It keeps the lower and upper readbacks downstream of the common tail row and the shared dyadic ledger.

Finally use the sibling dependency route Definition 5874.6. It already requires the StreamName and RegSeqRat siblings before the Real seal, so adding the cofinal tail-mesh read fixes the horizontal dependency without adding any external analytic coordinate. □

Chapter 5517

A Concrete Naming Certificate for $\mathsf{DynamicProgramming}^{\uparrow}$

Following Bellman's finite dynamic programming principle, the $\mathsf{DynamicProgramming}^{\uparrow}$ packet records finite backward recursion data as BEDC NameCert rows. It sits after the finite path and edge ledger of chapter 236, the transition-row discipline of chapter 189, the probability-row carrier of chapter 184, the terminal real-name seal of chapter 38, and the finite horizon counter of chapter 28. The chapter names only finite states, finite actions, displayed transition and cost rows, horizon-indexed Bellman tables, policy readback, transport, replay, provenance, and local naming; it does not assert an infinite-horizon fixed point, host minimizer, choice-selected policy, quotient value equality, or ambient completeness theorem.

Definition 5517.1 (Dynamic programming carrier)**.** *A* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, T, C, H, V, P, R, N)$$

with the following displayed rows:

1. *S is the finite displayed state row, sharing its path endpoints with chapter 236.*
2. *A is the finite action row, listing exactly the actions admitted at the displayed states.*
3. *T is the transition row, read through the finite Markov and probability surfaces of chapter 189 and chapter 184.*
4. *C is the cost row, exposing the finite real-valued immediate costs and terminal costs through the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *H is the horizon row, a* $\mathsf{Nat}^{\uparrow}$*-indexed finite counter for the backward recursion stages.*
6. *V is the Bellman value table, giving one displayed value row for each state and horizon stage.*
7. *P is the policy readback row, recording only actions whose value entry is witnessed by the same finite transition and cost rows.*
8. *R records componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance over S, A, T, C, H, V, P and the cited sibling packets.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $S, A, T, C, H, V, P, R, N$. It has no coordinate for a host minimizer, selected infinite policy, analytic contraction theorem, quotient value function, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DynamicProgrammingUp.DynamicProgrammingTasteGate single carrier alignment`

Theorem 5517.2 (Dynamic programming Bellman obligations)**.** *For every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier* $D = (S, A, T, C, H, V, P, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *surface consists exactly of carrier admission for finite states, finite actions, transition rows, cost rows, horizon rows, Bellman value-table rows, policy readback rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host minimizers, selected infinite policies, quotient value equality, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 5875.1. The finite state and action rows are displayed before transition and cost rows can be read, and the horizon row fixes which backward stages may appear.

The Bellman table row V is then read stage by stage over the same displayed horizon. A policy readback row P is exposed only after the corresponding state, action, transition, cost, and value rows have all been displayed.

Transport, replay, provenance, and local naming are exhausted by R and N. Since the carrier has no further coordinate beyond $S, A, T, C, H, V, P, R, N$, no accepted read can import a host minimizer, selected infinite policy, quotient value equality, countable choice, or ambient completeness. □

Theorem 5517.3 (Dynamic programming finite-horizon induction)**.** *Every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *Bellman table is generated by finite backward induction over the displayed* $\mathsf{Nat}^{\uparrow}$ *horizon row. The terminal stage reads the terminal cost row from* C *and the* $\mathsf{Real}^{\uparrow}$ *seal; each predecessor stage reads only the previously displayed successor-stage value row, the transition row* T*, the action row* A*, the immediate cost row* C*, and the same local NameCert route.*

Proof. Project the horizon row H and read it through the unary source discipline of chapter 28. The empty endpoint is the terminal stage, so the value entry is the displayed terminal cost row sealed through $\mathsf{Real}^{\uparrow}$.

For a one-step predecessor, use the closed generation of the horizon row. The successor-stage table has already been displayed, and the new value row is computed from the finite transition row, finite action row, immediate cost row, and successor value row.

Repeating the one-step construction along the finite horizon supplies every value-table row. No step asks for an infinite horizon, a global fixed point, or an external minimizer; the induction is only the BEDC finite-history recursion named by the carrier. □

Theorem 5517.4 (Dynamic programming policy readback)**.** *Every policy row exported by an accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *packet is a readback from the displayed Bellman table. For each displayed state and horizon stage, the row* P *may expose an action only after the state row* S*, action row* A*, transition row* T*, cost row* C*, horizon row* H*, value row* V*, componentwise transport, replay, provenance, and local naming have all been read.*

Proof. Start with a policy-facing read. The carrier forces the read through the finite state and action rows, then through the transition and cost rows that define the local Bellman entry.

Next read the horizon row and the value-table row. The policy row is not an independent selector: it is attached to the displayed value entry at the same state and horizon stage.

Finally apply the structural row R and local naming row N. These rows preserve the finite route under transport and replay, so any action exposed by P remains tied to the same Bellman table entry and cannot become a host choice function or infinite policy. □

Theorem 5517.5 (Dynamic programming NameCert package)**.** *The* $\mathsf{DynamicProgramming}^{\uparrow}$ *NameCert package is the finite Bellman-recursion packet whose public surface is exactly the route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{ShortestPath}^{\uparrow}, \mathsf{MarkovChain}^{\uparrow}, \mathsf{ProbSpace}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DynamicProgramming}^{\uparrow}.$$

It exports the state, action, transition, cost, horizon, value-table, policy-readback, transport, replay, provenance, and local naming rows of Definition 5875.1. It exports no host optimization theorem, no selected global policy, no infinite-horizon convergence theorem, no quotient value-function equality, and no bridge to a standard dynamic-programming model.

Proof. The cited sibling chapters supply the input rows: shortest-path data for finite route endpoints, Markov and probability data for transition rows, real data for cost and value seals, and natural-number data for the finite horizon.

Apply Definition 5875.2 to identify the local obligation surface. Apply Definition 5875.3 to generate the value table from the terminal stage backward through the displayed horizon. Apply Definition 5875.4 to expose policy rows only as readbacks from that table.

The package is therefore a finite NameCert surface over the displayed rows. Because no carrier coordinate stores host optimization, global choice, infinite convergence, quotient equality, or a standard bridge, those objects are not exported. □

Falsifiable prediction: A claimed $\mathsf{DynamicProgramming}^{\uparrow}$ read that exposes a policy action without first displaying S, A, T, C, H, V, R, N refutes the finite Bellman carrier.

Independence witness: shortestpath, markovchain, probspace, real, nat: no listed sibling is bijective to $\mathsf{DynamicProgramming}^{\uparrow}$ because this packet binds finite state-action data, transition and cost rows, horizon-indexed Bellman value tables, policy readback, transport, replay, provenance, and local naming as one finite recursion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DynamicProgramming}^{\uparrow}$ packet is generated from finite state rows, finite action rows, transition rows, cost rows, a Nat-indexed horizon, Bellman value-table rows, policy readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5875.1, Definition 5875.2, Definition 5875.3, Definition 5875.4, and Definition 5875.5 seed the finite Bellman-recursion NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host minimizers, selected infinite policies, quotient value equality, infinite-horizon convergence, ambient RealUp completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Bellman table construction, finite-horizon induction, policy readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.DynamicProgrammingUp.

Chapter 5518

Regular Cauchy Even-Odd Substreams NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet records the standard constructive passage from one regular Cauchy name to its even and odd substreams before a Real or Completion consumer reads either side. It sits over the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the two-source interleaving packet of chapter 414, and the finite interleaving criterion of chapter 3405. The packet supplies a displayed subsequence route for the L10 Real and Completion corridor; it does not choose a new limit, quotient streams, or import ambient completeness.

Definition 5518.1 (Regular Cauchy even-odd substreams carrier)**.** *A* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, E, O, D, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row. It names one* $\mathsf{RegSeqRat}^{\uparrow}$ *source together with the finite precision requests inspected by the parity split.*
2. *E is the even selector-window row. It displays the finite windows read from the source at indices $2n$ and records no schedule beyond the even selector.*
3. *O is the odd selector-window row. It displays the finite windows read from the source at indices $2n + 1$ and records no schedule beyond the odd selector.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row. It stores the shared dyadic radius, split tolerance, comparison windows, and common-scale data used by both selector windows.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the even and odd windows under the budget D.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the even readback, available only after S, E, D, R have been displayed.*
7. *B is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the odd readback, available only after S, O, D, R have been displayed.*

8. *H records componentwise* hsame *transport for the source, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an even or odd substream request through $S, E, O, D, R, A, B, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, O, D, R, A, B, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.RegularCauchyEvenOddSubstreamsTasteGate`
`single carrier alignment`

Theorem 5518.2 (Regular Cauchy even-odd substreams NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier*

$$X = (S, E, O, D, R, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, transport, replay, provenance, and local naming rows;*
2. *source visibility, requiring both parity branches to read the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row S;*
3. *even-window exactness, requiring every even branch request to pass through the selector windows E and the budget D before any readback is visible;*
4. *odd-window exactness, requiring every odd branch request to pass through the selector windows O and the same budget D before any readback is visible;*
5. *paired regular readback admission through R, requiring the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to use the selector windows and the shared dyadic ledger already displayed;*
6. *two Real seal boundaries through A and B, requiring the even and odd Real-facing reads to remain downstream from their parity windows, D, R, H, and C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly $S, E, O, D, R, A, B, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5876.1. Its displayed inventory is exactly $S, E, O, D, R, A, B, H, C, P, N$, so the source row, parity selector windows, dyadic budget, paired readbacks, two seals, and structural rows exhaust the packet.

The source visibility clause is read from S. Both parity branches share that row, so neither branch can replace the source by a second hidden regular name.

Even-window and odd-window exactness are read from E and O under the shared dyadic row D. The even branch exposes only indices of the form $2n$, while the odd branch exposes only indices of the form $2n+1$, and both branches use the same tolerance ledger before the readback row is consumed.

The paired readback row R is downstream from those windows. It repacks the two selector routes as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks without adding a quotient stream or selected host limit.

Finally the Real seals A and B are reached only after the corresponding parity route has been transported by H and replayed by C. Since P and N name only the displayed packet, the listed rows are the complete local NameCert surface. □

Theorem 5518.3 (Regular Cauchy even-odd Real-seal nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier factors through one of the two finite parity routes*

$$S, E, D \longmapsto R \longmapsto A \longmapsto H, C, P, N,$$
$$S, O, D \longmapsto R \longmapsto B \longmapsto H, C, P, N.$$

The even route reads the even selector windows and the odd route reads the odd selector windows, but both routes share the same source row, dyadic budget, paired regular readback, transport, replay, provenance, and local naming. Hence no consumer reaches a $\mathsf{Real}^{\uparrow}$ *seal through quotient stream equality, a chosen host limit, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the obligation surface of Definition 5876.2. It confines every accepted packet to the displayed source, two parity selector rows, one dyadic budget, paired readback, two seal rows, and structural data.

For an even request, the consumer reads S, E, D before R and then reaches A. The route has no coordinate other than the even selector windows and the shared budget, so the even seal cannot bypass the displayed packet.

For an odd request, the consumer reads S, O, D before R and then reaches B. This is the same source and budget route with the odd selector row in place of the even selector row.

The structural rows H, C, P, N transport, replay, source, and name the two routes componentwise. Because the carrier contains no quotient stream equality, selected host limit, countable-choice row, trichotomy branch, or ambient completeness coordinate, every Real or Completion consumer remains inside one of the two displayed parity routes. □

Theorem 5518.4 (Regular Cauchy parity cofinality)**.** *Let* $X = (S, E, O, D, R, A, B, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier over one source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*. The even selector windows* E *and the odd selector windows* O *determine cofinal tail-shift reads of that same source row. Their Real-facing seals factor through the same ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The assertion is only this cofinal finite-window factorization. It does not identify quotient streams, select a limit, use countable choice, use trichotomy, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source row S and the two selector-window rows E, O from Definition 5876.1. The rows are displayed as windows over one source $\mathsf{RegSeqRat}^{\uparrow}$ carrier, so no second regular name is introduced by the parity split.

Apply Definition 301.257 to transport late reads along the even and odd tail shifts. The tail-shift result keeps each parity branch inside the same regular-rational source route.

Next use Definition 126.670. It supplies one finite request section for the displayed selector windows, so both parity reads are compared at the finite StreamName surface rather than at a completed stream quotient.

Apply Definition 1647.3 to fuse the two parity budgets through the shared dyadic tolerance ledger. The shared D row is the DyadicRatCore face of the ordered packet.

Finally use Definition 38.713. The terminal Real seal is reached only after the dyadic budget, finite StreamName window, and RegSeqRat readback have been displayed. Thus the two parity seals are cofinal views of the same finite four-face route, with no selected limit, countable-choice selector, trichotomy branch, quotient stream equality, or ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ Real or Completion read is refuted if it reaches either $\mathsf{Real}^{\uparrow}$ seal without displaying the source RegSeqRat row, the selected even or odd StreamName windows, the shared DyadicRatCore budget, the paired RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-interleaving, cauchy-interleaving-criterion: $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one source regular Cauchy name, two parity selector-window routes, one shared dyadic budget, paired regular readbacks, two Real seal boundaries, transport, replay, provenance, and local naming as one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet is generated from one RegSeqRat source, even selector windows, odd selector windows, a shared DyadicRatCore budget, paired RegSeqRat readbacks, two RealUp seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5876.1, Definition 5876.2, Definition 5876.3, and Definition 5876.4 seed the parity-substream route used by L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parity-window exactness, shared dyadic budget exactness, paired regular readback admission, two Real-seal nonescape clauses, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.

Chapter 5519

A Concrete Naming Certificate for $\mathsf{SchwartzSpace}^{\uparrow}$

Following Schwartz and Treves on test-function spaces, the $\mathsf{SchwartzSpace}^{\uparrow}$ packet records the finite locally convex carrier surface behind rapidly decreasing smooth test functions. It sits beside the complete locally convex interface of chapter 3495, the finite family-seminorm route of chapter 5868, the pointwise test-function carrier of chapter 3399, and the derivative observation route of chapter 2210. The packet gives Fourier, tempered-distribution, weak-derivative, and PDE-facing consumers a displayed test-function-space surface without importing a host topological vector space, quotient function equality, unrestricted seminorm enumeration, selected representatives, or classical completeness.

Definition 5519.1 (SchwartzSpace carrier)**.** *A* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, V, I, G, D, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetSpace}^{\uparrow}$ *source row, exposing complete metrizability and locally convex handoff only through the displayed Frechet carrier.*
2. *V is the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite-family row, naming the seminorm labels used by the Schwartz test-function space read.*
3. *I is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-function row, recording the accepted rapidly decreasing smooth windows inspected by the packet.*
4. *G is the polynomial-growth and rapid-decay ledger. It binds each displayed seminorm index to a derivative order, a polynomial weight, and the real-valued budget read used to test decay.*
5. *D is the* $\mathsf{WeakDerivative}^{\uparrow}$ *stability row, recording derivative-window shifts and sign conventions that remain inside the same Schwartz test-function-space surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row used by seminorm values, decay budgets, and completion-facing estimates.*
7. *T is the completion and distribution nonescape row. It permits Frechet and tempered-distribution consumers to read the displayed test-function-space boundary only after F, V, I, G, D, R have been exposed.*

8. *H records componentwise* hsame *transport for Frechet, seminorm-family, test-function, rapid-decay, derivative, Real, nonescape, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every test-function-space request through $F, V, I, G, D, R, T, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Schwartz topology, a quotient of functions, an unrestricted seminorm family, a selected completion point, a distribution dual, nuclearity, Fourier inversion, or classical choice.

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.SchwartzSpaceTasteGate single carrier alignment`

Theorem 5519.2 (SchwartzSpace NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier*

$$S = (F, V, I, G, D, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Frechet-space, seminorm-family, Schwartz-function, rapid-decay, weak-derivative, Real, nonescape, transport, replay, provenance, and local naming rows;*
2. *seminorm growth routing, requiring every seminorm read to choose a displayed finite-family row in V and a polynomial-growth budget in G before a Real value can be consumed from R;*
3. *rapid-decay test-function closure, requiring every accepted test window from I to satisfy the displayed derivative-order and polynomial-weight ledger in G;*
4. *derivative-row stability, requiring weak-derivative shifts to pass through D and return to the same I, G, R surface rather than to host smooth-function equality;*
5. *completion and distribution nonescape, requiring T to mediate every Frechet, completion-facing, Fourier-facing, tempered-distribution, or PDE-facing read after F, V, I, G, D, R have been displayed.*

No $\mathsf{SchwartzSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.TasteGate.SchwartzSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5877.1. The analytic rows are exactly F, V, I, G, D, R, T, and the structural rows are exactly H, C, P, N.

Seminorm growth routing is forced by the order of the carrier. A consumer first reads the finite family row V, then the derivative and polynomial-weight budget in G, and only then may read a Real scalar estimate through R.

Rapid-decay closure and derivative stability are the two test-function-space checks. The row I supplies accepted Schwartz test windows, while G records their rapid-decay budgets and D records the weak-derivative shifts that stay inside the same displayed surface.

Finally T is a boundary row rather than an additional topological object. It lets Frechet, completion-facing, Fourier-facing, tempered-distribution, and PDE consumers replay the same rows without receiving a host Schwartz topology, quotient function equality, selected completion point, distribution dual, nuclearity, inversion theorem, or choice principle. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 5519.3 (SchwartzSpace functional-analysis handoff)**.** *Every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *packet exported to a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing consumer factors through*

$$\mathsf{FrechetSpace}^{\uparrow},\ \mathsf{SeminormedSpace}^{\uparrow},\ \mathsf{SchwartzFunction}^{\uparrow},\ \mathsf{WeakDerivative}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto F, V, I, G, D, R, T, H, C, P, N.$$

The exported object is a finite test-function-space NameCert surface with displayed seminorm growth budgets and derivative stability. It is not a host locally convex function space, an unrestricted seminorm enumeration, a selected completion, a dual-space quotient, or a classical distribution theorem.

Proof. Open the accepted carrier using Definition 5877.1. Any functional-analysis request must first choose one of the displayed source rows F, V, I, D, R and then pass through the growth ledger G.

Use Definition 5877.2. The seminorm and rapid-decay obligations keep polynomial weights and derivative orders attached to finite displayed rows, while the derivative-row obligation prevents a consumer from replacing weak-derivative stability by host smooth-function equality.

The nonescape row T is then the only export boundary. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route for every downstream consumer. Since no excluded host object is a carrier coordinate, the displayed route is the whole handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SchwartzSpace}^{\uparrow}$ consumer that reaches a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing read without first exposing $F, V, I, G, D, R, T, H, C, P, N$ refutes the carrier surface of Definition 5877.1.

Independence witness: frechet-space, seminormed-space, schwartz-function, weak-derivative, real: no listed sibling is bijective to $\mathsf{SchwartzSpace}^{\uparrow}$ because this packet jointly binds complete locally convex source rows, finite seminorm-family exposure, rapid-decay test-function windows, derivative stability, Real budget readback, completion and distribution nonescape, transport, replay, provenance, and local naming in one test-function-space carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchwartzSpace}^{\uparrow}$ packet is generated from a FrechetSpace row, a finite SeminormedSpace family row, SchwartzFunction test windows, rapid-decay and polynomial-growth ledgers, WeakDerivative stability rows, Real scalar readback, completion and distribution nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5877.1, Definition 5877.2, and Definition 5877.3 seed the finite locally convex test-function-space surface consumed by Frechet, Fourier, tempered-distribution, weak-derivative, and PDE-facing packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Schwartz topology, quotient function equality, unrestricted seminorm enumeration, selected completion, distribution duality, nuclearity, Fourier inversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seminorm growth routing, rapid-decay test-function closure, derivative-row stability, completion and distribution nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SchwartzSpaceUp.

Chapter 5520

A Concrete Naming Certificate for $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$

The $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet realizes the private/shared decoration model of chapter 9390 as finite Rule 110 classifier-decoration NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the finite pattern catalog of chapter 995, the Rule 110 classifier of chapter 5844, and the finite partition carrier of chapter 201. The packet names quotient support families, private blocks, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, transport, replay, provenance, and local naming; it does not import a host cellular semantics, an infinite orbit classifier, quotient support equality, a biological model, or a hidden enumeration oracle.

Definition 5520.1 (Rule 110 private/shared decoration carrier)**.** *A* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient support family row, naming the finite family of support blocks inspected by the decoration.*
2. *B is the private-block row, recording for each accepted support a displayed nonempty private coordinate block.*
3. *S is the shared-coordinate incidence row. For each shared coordinate it records the displayed set of incident private blocks and requires cardinality at least two.*
4. *R is the* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *row, attaching the decorated support family to finite Rule 110 row-window and pattern-tag reads.*
5. *E is the decoration-count provenance row. It records the partition and row-set enumeration data used by the private/shared decoration theorem without exposing a host generating-function oracle.*
6. *H records componentwise* hsame *transport for quotient support, private-block, shared-incidence, classifier, enumeration, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decoration request through Q, B, S, R, E, H, P, N.*

8. *P is the* Pkg *provenance over* RuleOneTenPrivateSharedDecoration$^{\uparrow}$, CellularAutomaton$^{\uparrow}$, CellularRowClassifier$^{\uparrow}$, CellularPatternCatalog$^{\uparrow}$, RuleOneTenPatternClassifier$^{\uparrow}$, Partition$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for absent private blocks, singleton shared incidence, host quotient equality, unbounded Rule 110 evolution, biological interpretation, or an external enumeration procedure.

Theorem 5520.2 (Rule 110 private/shared decoration NameCert obligations)**.** *For every accepted* RuleOneTenPrivateSharedDecoration$^{\uparrow}$ *carrier*

$$D = (Q, B, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of quotient support family admission through Q, private-block nonemptiness through B, shared-coordinate incidence through S, Rule 110 pattern-classifier routing through R, decoration-count provenance through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5878.1. The decoration data are exactly the quotient support row Q, the private-block row B, the shared-incidence row S, the Rule 110 classifier row R, and the enumeration provenance row E.

Private-block nonemptiness is read before a support can be accepted. Shared-coordinate incidence is then read as a displayed row set of size at least two, so a shared coordinate cannot be collapsed into a private coordinate or erased from the decoration.

The Rule 110 classifier row R attaches the decorated support family to finite row-window and pattern-tag reads from the cellular siblings. The count provenance row E records the finite partition and row-set enumeration route and does not supply an external counting oracle.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since no excluded host object is present as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5520.3 (Private/shared decoration classifier handoff)**.** *Every accepted* RuleOneTenPrivateSharedDecorati *consumer that reads a decorated Rule 110 support family factors through*

$$\mathsf{CellularAutomaton}^{\uparrow},\ \mathsf{CellularRowClassifier}^{\uparrow},\ \mathsf{CellularPatternCatalog}^{\uparrow},\ \mathsf{RuleOneTenPatternClassifier}^{\uparrow},\ \mathsf{Partition}^{\uparrow} \longmapsto Q$$

The exported object is a finite classifier-decoration packet with private-block nonemptiness, shared incidence, and decoration-count provenance. It is not an infinite cellular-orbit classifier, a host quotient-support equality, a biological stochastic model, or an unrestricted enumeration theorem.

Proof. Open the accepted carrier using Definition 5878.1. A support-family read can inspect only Q, and each support must expose its nonempty private block through B before shared coordinates are consumed.

The shared-coordinate row S supplies the incidence sets whose size is at least two. The classifier row R then routes the accepted decoration into the existing cellular automaton, row-classifier, pattern-catalog, and Rule 110 pattern-classifier surfaces.

The provenance row E records the finite decoration-count route, while H, C, P, N preserve the same data under transport, replay, package provenance, and local naming. Thus every accepted consumer receives exactly the displayed finite packet and none of the excluded host semantics. □

Falsifiable prediction: Any accepted $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ consumer that omits private-block nonemptiness or shared-coordinate incidence rows while reading a decorated Rule 110 support family refutes the carrier surface of Definition 5878.1.

Independence witness: cellularautomaton, cellularrowclassifier, cellularpatterncatalog, rule110patternclassifier: no listed sibling is bijective to $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ because this packet jointly binds quotient support families, nonempty private blocks, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, transport, replay, provenance, and local naming.

The Lean TasteGate file for this packet must provide $\mathsf{Nontrivial}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ instances for the displayed rows $Q, B, S, R, E, H, C, P, N$ before the decoration carrier can serve as a verified classifier-decoration consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet is generated from quotient support family rows, nonempty private-block rows, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5878.1, Definition 5878.2, and Definition 5878.3 seed the finite private/shared classifier-decoration surface for Rule 110 support-family consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite Rule 110 orbit classification, host quotient-support equality, biological interpretation, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient support admission, private-block nonemptiness, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPrivateSharedDecorationUp.

Chapter 5521

A Concrete Naming Certificate for $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ packet records the finite diagonal obstruction to enumerating Cantor-space bit names. It sits over the binary prefix carrier of chapter 1475, the finite bar-facing tree discipline of chapter 1622, the fan-theoretic bounded-window route of chapter 400, and the Real-facing non-surjection boundary of chapter 3793. The packet names only finite rows: an attempted enumeration window, its displayed diagonal bit schedule, the disagreement row at each inspected index, and the no-enumeration handoff. It is not a host-cardinality theorem, not a completed powerset construction, and not a quotient-stream principle.

Definition 5521.1 (Constructive Cantor finite diagonal carrier)**.** *A* $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, W, D, G, T, F, R, H, C, P, N)$$

with the following visible rows:

1. *E is the displayed finite enumeration attempt row, listing the first n candidate Cantor-space names through finite prefix windows;*
2. *W is the finite binary observation window, naming the inspected coordinates and the finite prefix depth used for each candidate;*
3. *D is the diagonal-bit row, obtained by flipping the displayed ith bit of the ith candidate whenever that coordinate is present in W;*
4. *G is the disagreement ledger, recording for every inspected index $i < n$ that the diagonal row disagrees with the ith candidate at the displayed coordinate;*
5. *T is the tree-read row through* $\mathsf{KleeneTree}^{\uparrow}$*, ensuring that finite prefix extension and absence of a completed path are read as finite branch data;*
6. *F is the fan-window row through* $\mathsf{FanTheorem}^{\uparrow}$*, recording the bounded prefix surface consumed by any compact Cantor-space handoff;*
7. *R is the Real-facing route through* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$*, used only after the Cantor-space disagreement ledger has been displayed;*

8. *H records componentwise* hsame *transport for enumeration rows, prefix windows, diagonal bits, disagreement ledgers, tree rows, fan rows, Real-facing routes, replay rows, provenance, and local naming;*
9. *C records displayed* Cont *replay from an enumeration window to diagonal-bit construction, disagreement verification, Cantor-space handoff, and Real-facing non-surjection routing;*
10. *P is the* Pkg *provenance over* ConstructiveCantorTheorem$^{\uparrow}$, CantorSpace$^{\uparrow}$, KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, RealDiagonalNonSurjection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, W, D, G, T, F, R, H, C, P, N$. It has no coordinate for a completed enumeration of all binary streams, host powersets, quotient stream equality, countable choice, or classical cardinal comparison.

Theorem 5521.2 (Finite diagonal obligations)**.** *For every accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier $K = (E, W, D, G, T, F, R, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *surface consists exactly of finite enumeration-window admission through E, binary observation through W, diagonal-bit formation through D, inspected-index disagreement through G, tree and fan routing through T, F, Real-facing non-surjection routing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5879.1. The enumeration and observation rows are E and W, so every candidate read is confined to a displayed finite prefix window.

Build the diagonal row by induction on the finite index spine of W. The empty window contributes no bit and no disagreement row. At a successor index, read the displayed candidate bit at the diagonal coordinate and place the opposite bit in D.

The same induction fills G. Each successor step adds exactly the disagreement between the new diagonal bit and the candidate bit just inspected, while previous rows are transported componentwise by H and replayed by C.

The rows T, F, R do not add an infinite object. They only route the same finite disagreement ledger through the Kleene-tree, fan-window, and Real-facing non-surjection surfaces, so the listed rows exhaust the certificate without a host powerset, quotient stream equality, choice principle, or classical cardinal comparison. □

Theorem 5521.3 (No-enumeration window)**.** *Let K be an accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier with a displayed enumeration window E of length n. The diagonal row D determines a Cantor-space prefix window such that for every inspected index $i < n$, the ith candidate of E differs from D at the displayed diagonal coordinate recorded in G. Hence the window cannot certify that its listed candidates exhaust the displayed Cantor-space prefixes.*

Proof. Induct on the finite enumeration window. For the empty window, there is no inspected candidate and the exhaustion claim has no displayed row.

At a successor window, the induction hypothesis supplies the disagreement rows for the previous candidates. The new candidate has a displayed diagonal coordinate in W, and the diagonal construction places the opposite binary value in D.

The new row of G records this disagreement. Componentwise hsame transport preserves the previous rows and the displayed Cont replay keeps the construction inside the same finite packet.

Therefore every inspected candidate is separated from the diagonal prefix at its own displayed coordinate. A finite window containing these disagreement rows cannot also certify that the same listed candidates exhaust the displayed Cantor-space prefix surface. □

Theorem 5521.4 (Cantor-space handoff)**.** *Every downstream consumer of a* ConstructiveCantorTheorem$^{\uparrow}$ *packet receives only the finite handoff formed by the enumeration window, binary observation window, diagonal-bit row, disagreement ledger, Kleene-tree row, fan-window row, Real-facing non-surjection route, transport, replay, provenance, and local naming. The handoff may be read by* CantorSpace$^{\uparrow}$ *and then by* RealDiagonalNonSurjection$^{\uparrow}$*, but it exports no completed stream quotient, powerset object, countable-choice selector, or classical uncountability principle.*

Proof. Start with the finite rows E, W, D, G of the carrier. By Definition 5879.2, these are the only rows that form and verify the diagonal disagreement.

Next read the Cantor-space side through chapter 1475. The Kleene-tree and fan rows T, F preserve finite prefix extension and bounded-window routing; they do not select a completed path or add an ambient compactness theorem.

Finally route the same disagreement ledger to chapter 3793 through R. Since the carrier has no coordinate for quotient stream equality, countable choice, host powersets, or classical cardinal comparison, the consumer receives exactly the finite handoff named in the statement. □

Falsifiable prediction: A purported ConstructiveCantorTheorem$^{\uparrow}$ consumer that reads a no-enumeration conclusion while omitting the enumeration window, diagonal-bit row, inspected-index disagreement ledger, Cantor-space prefix row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cantorspace, kleene-tree, fantheorem, real-diagonal-non-surjection: ConstructiveCantorT is not bijective to any listed sibling because it jointly carries an attempted enumeration window, a diagonal-bit construction, an inspected-index disagreement ledger, and a downstream Cantor-space-to-Real non-surjection route in one finite NameCert packet.

Closure status for NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 5879.1, Definition 5879.2, Definition 5879.3, and Definition 5879.4 seed the finite diagonal non-enumeration packet over Cantor-space prefix rows.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, a host powerset object, a countable-choice selector, a classical cardinal comparison, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean targets for carrier admission, diagonal-bit exactness, inspected-index disagreement, Cantor-space handoff, Real-facing route preservation, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCantorTheoremUp.

Constructive story: A ConstructiveCantorTheorem$^{\uparrow}$ packet is generated from a finite attempted enumeration window, a finite binary observation window, a diagonal-bit row, an inspected-index disagreement ledger, Kleene-tree and fan-window rows, Real-facing non-surjection routing, componentwise transport, displayed replay, provenance, and a local NameCert row.

MarkovKakutaniFixedPoint$^{\uparrow}$ routes its common fixed-window packet through the compact-convex and set-valued fixed-point surfaces of chapter 1923 and chapter 3511.

Chapter 5522

Markov-Kakutani Fixed Point NameCert Construction

Following Markov and Kakutani, the $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet records the finite BEDC surface for a commuting affine family acting on a compact convex carrier. It is anchored by the convex carrier of chapter 208, the functional-analysis ambient rows of chapter 94, the compact-convex self-map handoff of chapter 1923, and the set-valued fixed-point surface of chapter 3511. The packet names a finite common fixed-point obligation route; it does not import a host compactness theorem, a global choice selector, an ambient locally convex space, or a completed real vector topology outside the displayed rows.

Definition 5522.1 (Markov-Kakutani fixed point carrier)**.** *A* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, A, F, C, W, I, H, R, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the carried* $\mathsf{ConvexSet}^{\uparrow}$ *and compact support surfaces.*
2. *A is the affine ambient row, exposing only the finite vector-operation and scalar-stability data required by the displayed family.*
3. *F is the commuting-family row, listing the finite subfamily of affine self-maps and their source and target admissions.*
4. *C is the commutation ledger, recording pairwise commutation only for the displayed finite family windows.*
5. *W is the finite fixed-window row, recording the common approximate fixed rows inspected by downstream consumers.*
6. *I is the invariant-subcarrier row, exposing the finite intersection handoff used to keep the common fixed-window route inside the compact convex carrier.*
7. *H records componentwise* hsame *transport for domain, affine ambient, family, commutation, fixed-window, invariant-subcarrier, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a family request through $D, A, F, C, W, I, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, A, F, C, W, I, H, R, P, N$. It has no coordinate for an unbounded operator family, host compactness, selected fixed point, quotient equality of nets, or an ambient topological-vector-space theorem.

Theorem 5522.2 (Markov-Kakutani fixed point obligation surface)**.** *For every accepted* MarkovKakutaniFixedPoint *carrier*

$$M = (D, A, F, C, W, I, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of compact-convex domain admission through D, affine ambient admission through A, finite commuting-family admission through F, pairwise commutation exposure through C, fixed-window exposure through W, invariant-subcarrier handoff through I, componentwise* hsame *stability through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5880.1. The compact-convex, affine, commuting-family, commutation, fixed-window, and invariant-subcarrier data are exactly the rows D, A, F, C, W, I.

The domain and affine rows bind the finite ambient context before any map in the family is read. The family row F then lists only displayed affine self-maps, and the commutation ledger C records pairwise commutation inside that finite list.

The fixed-window and invariant-subcarrier rows W, I are the only route to a common fixed-point obligation. Finally H, R, P, N transport, replay, source, and name the same packet. Since no additional coordinate is present, the listed rows exhaust the NameCert surface. □

Theorem 5522.3 (Markov-Kakutani commuting handoff)**.** *Every accepted consumer of a* MarkovKakutaniFixedPoi *packet that asks for a common fixed-window read factors through*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{SchauderFixedPoint}^{\uparrow},\ \mathsf{KakutaniFixedPoint}^{\uparrow} \longmapsto D, A, F, C, W, I, H, R, P, N.$$

The exported object is a finite commuting-family fixed-window packet. It is not a selected fixed point, a host compactness theorem, a global choice principle, a quotient of approximating nets, or an ambient locally convex-space completion.

Proof. Open the accepted carrier using Definition 5880.1. A consumer must first read the compact convex domain row D and the affine ambient row A, so the route is grounded in the displayed convex and functional-analysis siblings.

Next read the finite family row F and the commutation ledger C. These rows expose only the listed affine self-maps and their pairwise commutation windows, so no unbounded operator family or host selection principle enters the packet.

The fixed-window row W is then interpreted inside the invariant-subcarrier row I. The Schauder and Kakutani siblings provide fixed-point handoff surfaces for the displayed compact-convex route, but the Markov-Kakutani packet exports only the finite common-window ledger named by W, I.

Finally H, R, P, N preserve the same carrier under transport, replay, provenance, and local naming. The exclusion clauses in the carrier therefore prevent selected fixed points, host compactness, quotient-net equality, and ambient completion from leaving the packet. □

Falsifiable prediction: Every accepted $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ consumer must expose compact-convex domain, affine ambient, finite commuting-family, pairwise commutation, fixed-window,

invariant-subcarrier, hsame, Cont, Pkg, and NameCert rows before it can read a common fixed-window obligation.

Independence witness: convexset, functionalanalysis, schauder-fixed-point, kakutani-fixed-point: $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite commuting affine family, commutation ledger, fixed-window rows, invariant-subcarrier handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet is generated from compact-convex domain rows, affine ambient rows, finite commuting-family rows, pairwise commutation ledgers, fixed-window rows, invariant-subcarrier handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5880.1, Definition 5880.2, and Definition 5880.3 seed the finite Markov-Kakutani fixed-point packet over displayed compact-convex, affine, commuting-family, fixed-window, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected common fixed points, global choice, quotient equality of nets, ambient locally convex completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex domain admission, affine-family routing, commutation ledger coverage, fixed-window exposure, invariant-subcarrier handoff, transport, replay, provenance, and local naming for BEDC.Derived.MarkovKakutaniFixedPointUp.

Proposition 5522.4 (Markov-Kakutani fixed-window dependency route)**.** *Every accepted* MarkovKakutaniFixedPoi *consumer that reads the common fixed-window row W consumes* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *from chapter 3511 and* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *from chapter 1923 only through the displayed compact-convex fixed-window route $D, A, F, C, W, I, H, R, P, N$ of Definition 5880.3. The route records the horizontal fixed-point dependency while refusing a detached host compactness theorem, selected common fixed point, or ambient locally convex completion.*

$\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ routes its private/shared row-set ledger to the defect-refined decoration consumer of chapter 7630, while the child files keep the NameCert carrier and forward route separate.

Chapter 5523

Window-Six Private/Shared Decoration NameCert Construction

The WindowSixPrivateSharedDecoration$^{\uparrow}$ packet realizes the private/shared decoration enumeration of Definition 9462.1699 inside the DNA window-six boundary-tile source of subsection 9462.28.21. It consumes the boundary-tile packet of chapter 3393, the six-boundary coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the trigger-threshold packet of chapter 4217. The packet records private blocks, shared row sets, trigger coverage, decoration counts, transport, replay, provenance, and local naming as finite rows; it is not a biological mechanism, stochastic fitness model, hidden enumeration oracle, or quotient codon semantics.

Definition 5523.1 (Window-six private/shared decoration carrier)**.** *A* WindowSixPrivateSharedDecoration$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, T, E, D, H, C, P, N)$$

with the following displayed rows:

1. *B is the private-block row, recording the nonempty private coordinate blocks attached to each accepted decorated support.*
2. *S is the shared-row-set ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *T is the trigger-coverage row, read through the carried DNA boundary-tile, six-boundary coverage, closure, and threshold packets.*
4. *E is the decoration-count row, recording the finite enumeration factors from the source private/shared theorem.*
5. *D is the defect ledger, recording row-set-size contribution to the defect-refined decoration polynomial.*
6. *H records componentwise* hsame *transport for private blocks, shared row sets, trigger coverage, decoration count, defect ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a window-six decoration request through B, S, T, E, D, H, P, N.*
8. *P is the* Pkg *provenance over* WindowSixPrivateSharedDecoration$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, DnaBoundaryTriggerClosure$^{\uparrow}$, DNAWindowSixTriggerThreshold$^{\uparrow}$, *the source theorem of Definition 9462.1699,* BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, S, T, E, D, H, C, P, N$. It has no coordinate for biological mechanism, stochastic evolutionary semantics, quotient codon equality, hidden global enumeration, or an unrecorded trigger oracle.

Theorem 5523.2 (Window-six private/shared decoration obligations)**.** *For every accepted* WindowSixPrivateShared *carrier*

$$W = (B, S, T, E, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of private-block admission through B, shared-row-set admission through S, trigger-coverage exposure through T, decoration-count exposure through E, defect-ledger exposure through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5881.1. The private block, shared row set, trigger coverage, decoration count, defect ledger, transport, replay, provenance, and naming data are precisely $B, S, T, E, D, H, C, P, N$.

The rows B, S enforce the private/shared split: each accepted support has a displayed private block, and each shared coordinate is assigned to a row set of size at least two. The trigger-coverage row T then ties those rows to the existing window-six boundary, coverage, closure, and threshold packets.

The rows E, D record the enumeration and defect data used by the source theorem. Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5523.3 (Window-six private/shared decoration enumeration handoff)**.** *Every accepted* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *consumer that reads a private/shared window-six decoration factors through*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

The exported object is a finite private/shared row-set ledger with trigger-coverage and decoration-count rows. It is not a biological model, a quotient codon space, a hidden enumeration oracle, or an unrestricted trigger-search theorem.

Proof. Read the accepted carrier through Definition 5881.1. The private and shared ledgers B, S supply the displayed row-set split before a decoration count is exposed.

Next use T to route the split through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings. This prevents a decoration request from bypassing the window-six trigger-coverage surface.

The enumeration row E and defect row D then record the finite count and defect-polynomial data from the source theorem. The structural rows H, C, P, N transport, replay, source, and locally name the same rows.

Therefore every accepted consumer receives the finite packet described in the theorem, rather than biological semantics, quotient codon equality, hidden enumeration, or unrestricted trigger search. □

Theorem 5523.4 (Window-six private/shared decoration sibling lattice)**.** *Every accepted* WindowSixPrivateSharedD *packet consumes the sibling lattice*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

before it exports the private/shared decoration ledger $B, S, T, E, D, H, C, P, N$. The lattice is finite and row-projection based: it displays boundary tiles, six-boundary coverage, trigger closure, and trigger-threshold evidence before the private block, shared row-set, trigger coverage, decoration count, and defect rows are consumed. It is not a biological mechanism, quotient codon semantics, stochastic evolutionary model, or hidden enumeration oracle.

Proof. Project the accepted carrier $W = (B, S, T, E, D, H, C, P, N)$ of Definition 5881.1. The private/shared ledger rows B, S, the trigger row T, the enumeration and defect rows E, D, and the structural rows H, C, P, N are the only exported coordinates.

Apply the enumeration handoff of Definition 5881.3. It forces every private/shared decoration read to pass through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings before $B, S, T, E, D, H, C, P, N$ are consumed.

Thus the dependency is a finite sibling lattice: each sibling supplies a displayed row that is projected before the decoration ledger is exported. Since the carrier has no coordinate for biological mechanism, quotient codon equality, stochastic evolution, or hidden enumeration, none of those interpretations is introduced by the lattice. □

Falsifiable prediction: Every accepted $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet must expose private-block, shared-row-set, trigger-coverage, decoration-count, hsame, Cont, Pkg, and NameCert rows before it can export a private/shared window-six decoration read.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-window-six-trigger-threshold: $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ is not bijective to any listed sibling because the private/shared row-set ledger jointly carries private blocks, shared row sets, trigger coverage, decoration counts, defect rows, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.WindowSixPrivateSharedDecorationUp.TasteGate` must provide explicit Nontrivial $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ and FieldFaithful WindowSixPrivateSharedDecora instances preserving $B, S, T, E, D, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet is generated from private-block rows, shared-row-set ledgers, trigger-coverage rows, decoration-count rows, defect ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5881.1, Definition 5881.2, and Definition 5881.3 seed the finite private/shared window-six decoration packet over displayed boundary, coverage, closure, threshold, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block exposure, shared-row-set arity, trigger-coverage routing, decoration-count exposure, defect-ledger exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.WindowSixPrivateSharedDecorationUp.

Proposition 5523.5 (Window-six private/shared decoration defect forward route)**.** *The private/shared row-set ledger exported by* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *is an allowed child route for chapter 7630. Its rows $B, S, T, E, D, H, C, P, N$ supply the private block, shared row set, trigger coverage, enumeration, defect ledger, transport, replay, provenance, and local naming data used by the downstream defect-refined decoration polynomial route.*

Proof. Read the exported packet through Definition 5881.1. The private/shared ledger already separates private blocks from shared row sets and records the defect ledger as a displayed row.

The downstream chapter chapter 7630 consumes exactly that displayed ledger when it forms its defect-refined decoration polynomial route. Because the route is through the named child rows rather than through the hub file, it preserves the hub-only layout and records a genuine forward dependency between the two finite packets. □

Chapter 5524

A Concrete Naming Certificate for $\mathsf{SchurLemma}^{\uparrow}$

Following Schur and the finite-group representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{SchurLemma}^{\uparrow}$ packet records the finite Hom-intertwiner boundary for irreducible representation rows. It sits between the finite representation carrier of chapter 3662, the character-theory trace surface of chapter 245, the representation-ring package of chapter 246, the orthogonality route of chapter 1603, and the character-table consumer of chapter 3574.

Definition 5524.1 (SchurLemma carrier)**.** *A* $\mathsf{SchurLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, H, S, T, R, P, N)$$

with the following displayed rows:

1. *U is an irreducible finite representation row over the same finite group source used by Definition 3662.1;*
2. *V is a second irreducible finite representation row over the same finite group source;*
3. *H is the displayed* Hom *and intertwiner row from U to V, recording equivariance rather than all host linear maps;*
4. *S is the scalar-or-zero boundary: when the two irreducible rows are the same finite representation row, accepted endomorphism reads factor through one scalar row, and when they are distinct no nonzero intertwiner row is exported;*
5. *T records componentwise* hsame *transport for the representation rows, Hom row, scalar-or-zero boundary, route, provenance, and naming rows;*
6. *R records displayed* Cont *replay from the finite representation packet through the Hom row into the scalar-or-zero boundary;*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *row.*

The carrier is exactly this finite representation-theory packet. It has no coordinate for a host abelian category, unrestricted module category, global algebra closure, or unrecorded linear-map equality.

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaTasteGate single carrier alignment`

Theorem 5524.2 (SchurLemma NameCert obligations)**.** *For the carrier of Definition 5882.1, the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (U, V, H, S, T, R, P, N)$*, irreducible finite representation exposure through* U, V*, Hom-intertwiner exposure through* H*, scalar-or-zero boundary exposure through* S*, componentwise* hsame *transport through* T*, displayed* Cont *replay through* R*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The packet lists U, V, H, S, T, R, P, N as its whole public surface, so the local certificate cannot introduce a second Hom object, a host representation category, or a hidden linear-map equality relation.

Read the representation rows first. The rows U and V are finite irreducible representation rows of the same group source, and the Hom row H is accepted only as an equivariant row between them.

The scalar-or-zero boundary is then read through S. It records scalar endomorphism reads in the same-row case and zero Hom reads in the distinct-row case. The rows T, R, P, N transport, replay, package, and name only that finite packet. □

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaCarrier namecert obligations`

Theorem 5524.3 (SchurLemma character-table handoff)**.** *Every accepted* $\mathsf{SchurLemma}^{\uparrow}$ *packet exports to* $\mathsf{CharacterTable}^{\uparrow}$ *only the finite Hom-intertwiner boundary needed by the Schur-orthogonality and character-pairing routes. The handoff reads* U, V, H, S *before* T, R, P, N *and exports no host representation category, unrestricted matrix algebra, quotient computation, or globally chosen basis.*

Proof. Start from Definition 5882.1. The downstream character-table consumer sees exactly the two irreducible representation rows, the displayed Hom-intertwiner row, and the scalar-or-zero boundary.

Use the neighboring representation-theory packets as routing anchors. The finite representation chapter supplies the representation rows, the character-theory and representation-ring chapters supply trace and sum surfaces, and the Schur-orthogonality chapter consumes the scalar-or-zero boundary before the character-table chapter reads pairings.

The structural rows T, R, P, N preserve that same finite packet under transport, replay, provenance, and local naming. Since no host category, unrestricted matrix algebra, quotient computation, or chosen-basis row appears in the carrier, the character-table handoff cannot export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurLemma}^{\uparrow}$ packet is generated from two irreducible finite representation rows, a displayed Hom-intertwiner row, a scalar-or-zero boundary row, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, seed)

Scope closed: Definition 5882.1, Definition 5882.2, and Definition 5882.3 seed the finite Hom-intertwiner boundary between irreducible representation rows and the character-table consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host representation category, unrestricted module-category Hom, quotient computation, globally chosen bases, Lean verification, or bridge checking for $\mathsf{SchurLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, irreducible-row exposure, Hom-intertwiner exactness, scalar-or-zero boundary, transport stability, replay stability, and local NameCert nonescape for BEDC.Derived.SchurLemmaUp.

Chapter 5525

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet records the interval-minimum half of the compact-real extreme-value route as finite BEDC NameCert data. It sits over the compact interval source of chapter 3986, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located lower-bound route dual to chapter 1427, and the terminal real-name seal of chapter 38. The packet is minimum-specific: it exposes lower-bound attainment windows without importing open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5525.1 (Compact real interval minimum carrier)**.** *A* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite endpoint windows, and real endpoint seals.*
2. *K is the compact interval row, read through* $\mathsf{BishopCompactInterval}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$*, exposing only the finite inspection surface admitted for the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact interval.*
4. *L is the lower-bound and located-supremum-dual row, recording that every inspected value is above a displayed lower-bound candidate and that the candidate is reached through finite located windows rather than by trichotomy.*
5. *W is the finite witness-window row, recording the interval windows in which F and L meet a terminal value seal.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the value and lower-bound comparison windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *value seal row reached only after I, K, F, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, lower-bound row, witness windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from minimum-attainment requests through* $I, K, F, L, W, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, arbitrary real trichotomy, quotient real equality, a selected host minimum, a selected host maximum, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactRealIntervalMinimumUp.CompactRealIntervalMinimumTasteGate single carrier alignment`

Theorem 5525.2 (Compact real interval minimum NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier*

$$M = (I, K, F, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact interval source, continuous-map row, lower-bound and located-supremum-dual row, finite witness-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *value seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No minimum obligation is stored outside these rows.*

Proof. Project the carrier of Definition 5883.1. The interval row I and compact row K expose the only compact-real source windows available to the packet, while F records the continuous real-valued map inspected on those windows.

Read the lower-bound and witness rows next. The row L gives the displayed lower-bound comparison as the dual of the located-supremum route, and W binds that comparison to finite source windows where the map row and bound row meet.

The regular and real-facing rows are downstream. The row R records the regular rational readback of the same finite comparisons, while E seals only those displayed reads as $\mathsf{Real}^{\uparrow}$ values.

The remaining rows H, C, P, N transport, replay, source, and locally name the same route. Since the carrier has no coordinate for open-cover compactness, trichotomy, quotient equality, selected host extrema, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5525.3 (Compact real interval minimum located attainment)**.** *Every accepted* CompactRealIntervalMinim *consumer that reads a minimum value factors through the displayed route*

$$I, K \longmapsto F, L \longmapsto W \longmapsto R, E \longmapsto H, C, P, N.$$

Thus the compact-real interval-minimum surface exports only a finite witness-window route from a closed bounded compact interval and continuous-map row to a located lower-bound row, regular rational readback, and $\mathsf{Real}^{\uparrow}$ *value seal. It does not export open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact interval source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real interval windows.

Feed the source into the continuous-map and lower-bound rows. The row F exposes only displayed value reads, while L records the lower-bound comparison through the located-supremum-dual route rather than through a host order decision.

Use the witness-window and readback rows together. The row W records where the compact interval inspection meets the lower-bound comparison, and R repacks those comparisons as $\mathsf{RegSeqRat}^{\uparrow}$ data before the $\mathsf{Real}^{\uparrow}$ seal E is read.

Finally, transport and replay preserve the route through H and C, while P and N provide provenance and local naming. No row supplies open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient completeness, so every accepted minimum read remains inside the displayed route. □

Falsifiable prediction: Every $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ minimum read factors through $\mathsf{ClosedBoundedInterva}$ $\mathsf{BishopCompactInterval}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$-dual lower-bound data, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before reaching finite witness windows and the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ row.

Independence witness: bishopcompactinterval, closedboundedinterval, continuousmap, locatedsupremum, real: $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact interval source, continuous-map row, located lower-bound route, finite witness-window row, regular readback, Real value seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a BishopCompactIntervalUp or CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, a lower-bound row dual to LocatedSupremumUp, finite witness windows, RegSeqRatUp readbacks, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5883.1, Definition 5883.2, and Definition 5883.3 seed the finite compact-real interval-minimum route over interval, compact-source, continuous-map, lower-bound, witness-window, regular readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient real equality, selected host extrema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, lower-bound routing, regular readback, Real value sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalMinir

Chapter 5526

A Concrete Naming Certificate for $\mathsf{SimpleFunctionApproximation}^{\uparrow}$

Following Caratheodory and Bourbaki measurable approximation, the $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet names the finite-range measurable surface by which a measurable real-facing map is approximated without importing completed measure theory or host function equality. It sits after the preimage-ledger map surface of chapter 3461, the event-value rows of chapter 92, the generated-set boundary of chapter 3384, and the terminal real-name seal of chapter 38. The packet records finite range cells, measurable preimage rows, displayed tolerance data, finite observations, transport, replay, provenance, and local naming.

Definition 5526.1 (Simple function approximation carrier)**.** *A* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, R, E, T, O, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row, exposing only the measurable event surface and event-value readback needed by the approximation packet.*
2. *F is the input* $\mathsf{MeasurableFunction}^{\uparrow}$ *row, read through its displayed target-event and preimage ledger rather than through a host function object.*
3. *R is the finite range ledger, listing the displayed* $\mathsf{Real}^{\uparrow}$ *values that the simple approximation may return.*
4. *E is the finite cell ledger, assigning each range row in R to a measurable source cell whose admission is read through* $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *and the preimage rows of F.*
5. *T is the displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, naming the approximation budget used by the current consumer.*
6. *O is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the visible source probes, range reads, and tolerance comparisons exposed to the packet.*
7. *H stores componentwise* hsame *transport for the measure source, measurable-function row, finite range ledger, cell ledger, tolerance row, observation row, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay from source-event inspection through finite cell selection, range-value readback, tolerance comparison, and observation replay.*

9. *P is the* Pkg *provenance over* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, E, T, O, H, C, P, N$. It has no coordinate for almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, or equality of host functions.

Lean4 ✓ `BEDC.Derived.SimpleFunctionApproximationUp.SimpleFunctionApproximationTasteGate single carrier alignment`

Theorem 5526.2 (Simple function approximation NameCert obligations)**.** *For the carrier of Definition 5884.1, the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure source, measurable-function row, finite range ledger, measurable cell ledger, displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, finite* $\mathsf{StreamName}^{\uparrow}$ *observations, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted approximation consumer must read the finite cells through E, the measurable preimage route through F, the range values through R, and the tolerance comparison through T before any sibling* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *row is exported.*

Proof. Project the carrier $S = (M, F, R, E, T, O, H, C, P, N)$. The measure source M and measurable-function row F are the only rows from which measurable event and preimage information can be read.

Read the finite approximation data. The range ledger R supplies the displayed real values, the cell ledger E ties each range value to a measurable source cell, and the tolerance row T records the visible approximation budget against which the current observations are compared.

Replay the observation and structural rows. The row O records the finite StreamName probes, while H, C, P, N transport, replay, source, and name the same route through the sibling certificates. Since no further coordinate is present, the listed rows exhaust the NameCert obligation surface and cannot export quotient classes, completed measures, selected representatives, ambient integration theorems, or host function equality. □

Theorem 5526.3 (Simple function approximation finite-cell route)**.** *Every accepted* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *read factors through the finite route*

$$M, F \longmapsto E \longmapsto R, T \longmapsto O \longmapsto H, C, P, N.$$

Thus a consumer receives a finite measurable partition-by-cells with displayed real values and a displayed tolerance budget, not an almost-everywhere quotient class, completed measure, selected representative, ambient integral, or equality proof for host functions.

Proof. Start with an accepted approximation request. The request first exposes the measure source and the measurable-function row, so every cell read is grounded in the displayed source-event and preimage ledgers.

The cell ledger E is finite. Each displayed cell is paired with a range value from R, and the tolerance row T supplies the visible real-valued budget for the comparison requested by the consumer.

Finally replay the observations and structural rows. The finite StreamName row O records only the exposed probes, while H, C, P, N transport and name the route. No hidden coordinate remains from which a quotient, completion, representative selection, integration theorem, or host equality could be recovered. □

Falsifiable prediction: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ consumer that obtains an almost-everywhere quotient class, completed measure, selected representative, ambient integration theorem, or host function equality without first reading $M, F, R, E, T, O, H, C, P, N$ refutes the carrier of Definition 5884.1.

Independence witness: measurablefunction, measure, borelset, real, streamname: no listed sibling supplies the whole ten-row packet $S = (M, F, R, E, T, O, H, C, P, N)$, because each sibling contributes only preimage-ledger, event-value, generated-set, real-name, or finite-observation data while this chapter binds the finite-range approximation surface with tolerance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet is generated from a measure source, a measurable-function preimage ledger, finite range cells, a displayed Real tolerance row, finite StreamName observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5884.1, Definition 5884.2, and Definition 5884.3 seed the finite-range measurable approximation surface over source events, preimage rows, cell ledgers, range values, tolerance rows, observations, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell exactness, measurable preimage preservation, Real tolerance replay, StreamName observation stability, and quotient-refusal proofs for BEDC.Derived.SimpleFunctionApproxima

Chapter 5527

A Concrete Naming Certificate for $\mathsf{DyadicModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicModulus}^{\uparrow}$ packet records the finite tolerance-modulus surface by which dyadic precision requests become regular rational readback before the terminal real seal is consumed. It sits between the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the terminal seal boundary of chapter 38, and the Cauchy-modulus route carried by $\mathsf{CauchyModulus}^{\uparrow}$. The packet is a finite NameCert carrier for Real and Completion consumers; it is not a host modulus function, quotient stream equality, selected limit, countable choice, or ambient completeness principle.

Definition 5527.1 (DyadicModulus carrier)**.** *A* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic scales, finite comparison budgets, rounding windows, and precision requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose visible entries are exactly the observations demanded by the tolerance row.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by reading the stream windows under the dyadic tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, available only after D, S, R have been displayed.*
5. *H records componentwise* hsame *transport for tolerance, stream-window, regular-readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each dyadic precision request through D, S, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, ambient Real completeness, or a detached host-level modulus function.

Lean4 ✓ `BEDC.Derived.DyadicModulusUp.DyadicModulusTasteGate single carrier alignment`

Theorem 5527.2 (DyadicModulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier* $M = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window exposure, regular rational readback, terminal Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every Real or Completion consumer must read the displayed route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before using the packet as a dyadic tolerance-modulus certificate.

Proof. Project the carrier of Definition 5885.1. The tuple displays exactly the dyadic tolerance row, stream-window row, regular-readback row, terminal seal, transport, replay, provenance, and local name.

The analytic route is forced in order. A precision request first reads D, then the finite stream observations in S, then the regular rational handoff R, and only then reaches the seal boundary E.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or detached-modulus coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5527.3 (DyadicModulus RegSeqRat handoff)**.** *Let* $M = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion-facing consumer of the modulus row must factor through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before the terminal seal row* E *is read.*

The handoff route is

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as rows of one BHist *packet, with transport, replay, provenance, and local naming supplied by* H, C, P, N*.*

Proof. Start from Definition 5885.2. It already places every tolerance-modulus consumer on the displayed route D, S, R, E, H, C, P, N.

The regular rational readback is the row R, and R is downstream of both the dyadic tolerance row D and the stream-window row S. Thus a consumer cannot read a regular name, Real seal, or Completion boundary before the finite tolerance and window data have been displayed.

Finally inspect the excluded coordinates of Definition 5885.1. Because no quotient stream, selected representative, host equality, countable-choice schedule, ambient completeness theorem, or detached modulus function is present, the RegSeqRat handoff is exactly the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{DyadicModulus}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or a Completion-facing seal without first displaying one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-modulus: $\mathsf{DyadicModulus}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide tolerance arithmetic,

finite windows, regular rational readback, terminal sealing, or threshold scheduling, while this packet binds them into one finite tolerance-modulus handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat handoff row, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5885.1, Definition 5885.2, and Definition 5885.3 seed the finite dyadic tolerance-modulus carrier between dyadic precision data and regular rational readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, structural transport, and non-escape for BEDC.Derived.DyadicModulusUp.

Chapter 5528

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5528.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5528.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5528.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5528.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The $\mathsf{Cauchy}^{\uparrow}$ packet is consumed through chapter 301 and chapter 38 before downstream completion or calculus reads inspect the finite Cauchy windows.

Chapter 5529

A Concrete Naming Certificate for $\mathsf{CauchyModulusSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet records a carrier-space interface for explicit Cauchy modulus schedules. It is anchored in the single-modulus certificate of chapter 292, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the fast Cauchy sequence route of chapter 2185. The chapter treats a family of modulus schedules as finite naming data, not as a quotient of real sequences or as an ambient completeness theorem.

Definition 5529.1 (Cauchy modulus space carrier)**.** *A* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row for the family of explicit modulus schedules.*
2. *S assigns to each displayed index a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, including the monotone precision requests used by the family.*
3. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule family.*
4. *D is the dyadic tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
6. *Q records the* $\mathsf{Real}^{\uparrow}$ *seal boundaries at which a scheduled read may leave rational and dyadic data for a real-facing consumer.*
7. *H records componentwise* hsame *transport for $I, S, W, D, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay from an index and precision request through the schedule row, stream window, dyadic tolerance ledger, regular rational readback, and real seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSpace}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the displayed rows componentwise by hsame*. It has no coordinate for an unscheduled Cauchy oracle, quotient equality of sequences, a selected real limit, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSpaceUp.CauchyModulusSpaceTasteGate single carrier alignment`

Theorem 5529.2 (Cauchy modulus space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index family, schedule admission for the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission, dyadic tolerance compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal-boundary compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming for the packet.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceNameCertObligations`

Proof. Project the accepted carrier of Definition 5887.1. The analytic data are precisely I, S, W, D, R, Q, and the structural rows are precisely H, C, P, N.

Schedule admission is rowwise over I. Each index exposes its explicit modulus schedule before any stream window is inspected, so the packet cannot replace the family by an unscheduled oracle.

The tolerance and readback rows are then forced through D and R. The real boundary row Q is available only after those rational and dyadic ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5529.3 (Cauchy modulus space finite route)**.** *Every real-facing comparison of two accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carriers factors through finite schedule rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundaries, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *rows. The route exports a shared modulus-comparison surface for later diagonal* $\mathsf{Real}^{\uparrow}$ *completion arguments without introducing quotient sequence equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFiniteRoute`

Proof. Open both carriers using Definition 5887.1. A comparison can inspect only the finite index rows, schedule rows, stream windows, tolerance ledgers, readback rows, and seal boundaries.

Componentwise hsame in H transports the two packets to a common finite surface. Displayed Cont replay in C keeps each precision request attached to its schedule, stream window, dyadic tolerance, regular readback, and real seal boundary.

The provenance and naming rows P, N package that finite surface as the local certificate. Thus later diagonal completion routes can share modulus comparison data without receiving any quotient or choice-based object. □

Theorem 5529.4 (CauchyModulusSpace seed interface)**.** *Every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet exports its modulus schedule family to the L10 four-object seal route only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Cauchy}^{\uparrow}$ *root route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of this seed interface can expose quotient stream equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as part of the packet surface.*

Proof. Project the carrier of Definition 5887.1. The modulus family is displayed by I, S, and every inspected request is routed through the finite observation rows W, D, R, Q before the structural rows H, C, P, N can name or replay it.

Apply Definition 5887.3. Any real-facing comparison has already factored through finite schedule rows, stream windows, dyadic tolerances, regular rational readbacks, and the terminal Real seal boundary.

The $\mathsf{Cauchy}^{\uparrow}$ root route receives exactly that finite schedule surface. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or ambient completeness, the seed interface cannot export any of those host objects to L10 Real and Completion consumers. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceSeedInterface`

Theorem 5529.5 (CauchyModulusSpace tail composition)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

two displayed modulus schedules in the family compose only through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The composed tail read is the finite route*

$$I, S, W, D, R, Q \longmapsto H, C, P, N$$

and it exports no quotient sequence equality, selected limit, countable choice, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceTailComposition`

Proof. Project the carrier of Definition 5887.1. The two schedules are displayed rows under S indexed by I, and their inspected tails must pass through the same finite window row W.

Apply the finite route and seed interface, Definition 5887.3 and Definition 5887.4. These results force the tolerance ledger D, regular readback R, and Real seal boundary Q to remain visible before the composed tail can be consumed.

Finally replay the structural rows. Componentwise transport H, displayed continuation C, provenance P, and local naming N preserve the same finite schedule surface, so tail composition does not add quotient equality, selected limits, countable choice, or ambient completeness. □

Theorem 5529.6 (CauchyModulusSpace filter handoff)**.** *Every CauchyFilter or CauchyFilterCompletion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet receives the modulus family only through the finite handoff*

$$I, S, W, D, R, Q, H, C, P, N.$$

The handoff may be consumed as filter-facing Cauchy evidence, but it cannot be reinterpreted as quotient sequence equality, a selected limit, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFilterHandoff`

Proof. Begin with a filter-facing read. The seed interface of Definition 5887.4 already routes every scheduled request through finite stream windows, dyadic tolerances, regular rational readback, the Cauchy root route, and the terminal Real seal.

Use tail composition from Definition 5887.5. It shows that composing displayed schedules preserves the same finite route through I, S, W, D, R, Q before structural replay.

The filter consumer therefore receives only the transported and replayed packet named by H, C, P, N. Since the carrier contains no quotient equality coordinate, no selected limit, no

countable-choice schedule, and no ambient completeness row, CauchyFilter and CauchyFilterCompletion may consume only the displayed finite handoff. □

Theorem 5529.7 (CauchyModulusSpace diagonal-family exactness)**.** *A diagonal Real or Completion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet may compare schedule families only through the displayed rows*

$$I, S, W, D, R, Q, H, C, P, N.$$

The diagonal request must be routed through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary before any structural transport, replay, provenance, or local naming row can be consumed. The comparison exports no quotient stream equality, selected limit, countable-choice schedule, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceDiagonalFamilyExactness`

Proof. Start with the finite route of Definition 5887.3. It already factors every real-facing comparison of two accepted packets through the finite schedule rows, stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, and the structural rows H, C, P, N.

Apply the seed interface of Definition 5887.4. It binds every L10 request to the schedule family I, S before the finite windows W, tolerance ledger D, readback R, and seal boundary Q are exposed.

Finally use the filter handoff of Definition 5887.6. A diagonal completion consumer is still a consumer of the same finite handoff, so it receives only $I, S, W, D, R, Q, H, C, P, N$ and cannot add quotient equality, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ consumer that compares schedule families while omitting $I, S, W, D, R, Q, H, C, P, N$ refutes the carrier surface of Definition 5887.1.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, fast-cauchy-sequence: no listed sibling binds a family of explicit Cauchy modulus schedules together with finite stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, componentwise transport, replay, provenance, and local naming in one carrier-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet is generated from a finite family of explicit Cauchy modulus schedules, finite StreamName windows, dyadic tolerance ledgers, RegSeqRat readbacks, Real seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5887.1, Definition 5887.2, and Definition 5887.3 seed the carrier-space interface for explicit modulus schedule families and shared finite comparison routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close diagonal Real completion, quotient equality of Cauchy sequences, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-family exactness, dyadic and regular readback compatibility, Real-seal boundary nonescape, and a Lean carrier formalization for BEDC.Derived.CauchyModulusSpaceUp.

Chapter 5530

A Concrete Naming Certificate for $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the finite-limit language of regular category theory, the $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet records the boundary where a displayed Cauchy-completion route preserves the finite-limit rows that its source already exposes. It sits beside the Cauchy source route of chapter 3413, the metric-completion universal-property route of chapter 2124, the functorial completion route of chapter 423, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite left-exactness boundary for displayed completion data; it does not assert quotient completions, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5530.1 (Cauchy completion left-exactness carrier)**.** *A* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, U, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source finite-limit row, recording the displayed terminal, product, equalizer, or kernel-pair data available before completion is read.*
2. *U is the completion unit row, recording the source-to-completion unit reads for every object named by S.*
3. *D is the dense image row, keeping each completed finite-limit comparison tied to the same source rows and unit rows.*
4. *K is the kernel-pair preservation row, recording that equalizer and pullback comparisons are read through the displayed source kernel-pair data rather than through host quotient equality.*
5. *E is the extension ledger, recording the universal-property and functorial completion rows used to transport finite-limit witnesses into completion-facing reads.*
6. *H records componentwise* hsame *transport for S, U, D, K, E, C, P, N.*
7. *C records displayed* Cont *replay from a finite-limit request through S, U, D, K, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, selected finite limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionLeftExactnessUp.TasteGate.CauchyCompletionLeftExactnessTasteGate single carrier alignment`

Theorem 5530.2 (Cauchy completion left-exactness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the source finite-limit row, completion unit rows, dense image rows, kernel-pair preservation row, extension ledger, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every completion-facing finite-limit read must expose S, U, D, K before E, H, C, P, N can be consumed.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessNameCertObligations`

Proof. Project the carrier of Definition 5888.1. The source finite-limit row, unit row, dense image row, kernel-pair row, extension ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The finite-limit boundary is read through S, U, D, K. The source row supplies the terminal, product, equalizer, or kernel-pair data; the unit row embeds the displayed source names into completion-facing reads; the dense image row keeps those reads tied to source data; and the kernel-pair row confines exactness comparisons to displayed finite rows.

The remaining rows E, H, C, P, N transport and name the same route. Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient completeness have no carrier coordinate, projection of the displayed rows exhausts the NameCert obligations. □

Theorem 5530.3 (Cauchy completion left-exactness carrier admission)**.** *For every accepted* CauchyCompletionLeftE *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the source finite-limit row S, completion unit row U, dense image row D, kernel-pair preservation row K, extension ledger E, componentwise hsame *transport H, displayed* Cont *replay C, provenance row P, and local* NameCert *row N are all admitted as finite coordinates of the same packet. No completion-facing read may treat one of these rows as ambient host data detached from the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactness carrier admission`

Proof. Project the carrier of Definition 5888.1. Its coordinates are exactly $S, U, D, K, E, H, C, P, N$, so each named row is admitted by carrier membership.

Apply Definition 5888.2. The obligation surface requires every completion-facing finite-limit read to expose S, U, D, K before the extension, transport, replay, provenance, and naming rows can be consumed.

Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, none of the admitted rows can be reread as detached host data. □

Theorem 5530.4 (Cauchy completion left-exactness unit-density route)**.** *Every displayed finite-limit comparison read through an accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier factors first through the source finite-limit row* S*, then through the completion unit row* U*, and then through the dense image row* D *before the extension ledger* E *and structural rows* H, C, P, N *are inspected. This route records the finite source-to-completion passage without selecting representatives of completed objects or invoking countable choice.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessUnitDensityRoute`

Proof. Begin with the carrier admission theorem Definition 5888.3. It keeps S, U, D as visible coordinates of the same finite packet.

The source row S supplies the finite-limit data before completion is read. The unit row U records the displayed source-to-completion unit for each object named by S, and the dense image row D keeps the resulting completion-facing comparison tied back to the same source and unit rows.

Only after this route is exposed can E, H, C, P, N be consumed. The extension ledger and the structural rows transport, replay, package, and name the source-unit-density passage; they do not add selected representatives, countable choice, or host equality of completed objects. □

Theorem 5530.5 (Cauchy completion left-exactness kernel-pair preservation)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any equalizer, pullback, or kernel-pair comparison visible after completion is read through the displayed kernel-pair row* K *together with the source finite-limit row* S*, completion unit row* U*, and dense image row* D*. The comparison cannot be exported as quotient completion equality, a selected finite limit, or ambient equality of completed objects.*

Proof. Project the kernel-pair coordinate K from the carrier. By definition, K records equalizer and pullback comparisons through displayed source kernel-pair data rather than through a host quotient.

Use the unit-density route theorem Definition 5888.4. It places every completion-facing finite-limit comparison behind S, U, D, so the kernel-pair row is read as part of the same source-to-completion passage.

Finally apply Definition 5888.2. The rows E, H, C, P, N only extend, transport, replay, package, and name the displayed comparison. They add no coordinate for quotient completion equality, selected finite limits, or ambient equality of completed objects, so preservation remains confined to K and its source, unit, and dense-image context. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessKernelPairPreservation`

Theorem 5530.6 (Cauchy completion left-exactness regular-limit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any regular Cauchy limit read used by a finite-limit comparison factors through the source finite-limit row* S*, the completion unit row* U*, the dense image row* D*, the kernel-pair row* K*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger* E *is consumed. Thus the regular-limit comparison is a pullback of displayed source, unit, dense-image, kernel-pair, regular-readback, and real-seal rows, not a quotient completion or a selected representative of a completed object.*

Proof. Project the accepted carrier. The rows $S, U, D, K, E, H, C, P, N$ are the only admitted coordinates, so a finite-limit comparison cannot begin from a hidden completed object or a host-level limit.

Apply Definition 5888.4. It places every completion-facing finite-limit comparison behind the source finite-limit row, the completion unit row, and the dense image row before the extension ledger and structural rows are inspected.

Next apply Definition 5888.5. Equalizer, pullback, and kernel-pair comparisons remain confined to the displayed kernel-pair row together with the same source, unit, and dense-image context.

The regular Cauchy limit read is then exposed through the sibling $\mathsf{RegSeqRat}^{\uparrow}$ readback surface named in the chapter introduction, and the resulting endpoint is sealed only through the sibling $\mathsf{Real}^{\uparrow}$ boundary. Since those sibling rows are read after S, U, D, K and before E, the comparison has the announced displayed pullback route. No coordinate remains for quotient completion equality, selected representatives, selected finite limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegularLimitPullback`

Theorem 5530.7 (Cauchy completion left-exactness RegSeqRat unit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any finite-limit comparison whose completion unit route reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factors first through the source finite-limit row* S*, completion unit row* U*, dense image row* D*, and kernel-pair row* K*. It then reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger and structural rows* E, H, C, P, N *are consumed. Componentwise* hsame *transport preserves that order, so the exported comparison contains no quotient completion, selected representative, selected finite limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegSeqRatUnitPullback`

Proof. Start with the carrier admission theorem Definition 5888.3. It keeps $S, U, D, K, E, H, C, P, N$ as finite coordinates of one accepted packet, so the comparison cannot begin from an external completed object.

Apply the unit-density route theorem Definition 5888.4. This places every completion-facing finite-limit comparison behind S, U, D before the extension ledger and structural rows are inspected.

Next apply the kernel-pair preservation theorem Definition 5888.5. Pullback and equalizer comparisons remain confined to K together with the same source, unit, and dense-image context.

Finally use the regular-limit pullback theorem Definition 5888.6. Its readback route inserts the $\mathsf{RegSeqRat}^{\uparrow}$ surface and the terminal $\mathsf{Real}^{\uparrow}$ seal after S, U, D, K and before E, H, C, P, N. Componentwise hsame transport therefore preserves the displayed order, while the excluded quotient, selection, choice, host-equality, and ambient-completeness rows remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet is generated from source finite-limit rows, completion unit rows, dense image rows, kernel-pair preservation, an extension ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5888.1 and Definition 5888.2 seed the finite left-exactness boundary for displayed Bishop-style Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected finite limits, selected representatives, countable choice, host equality of completed objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source finite-limit exposure, unit-row exposure, dense image confinement, kernel-pair preservation, extension-ledger confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionLeftExactnessUp.

Theorem 5530.8 (Cauchy root UnaryHistory admission)**.** *Every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *consumer read admits a UnaryHistory-facing presentation before it enters any filter, modulus, or Real-facing route. The admitted packet is the finite row surface*

$$U = (S, Q, D, R, E, H, C, P, N),$$

where S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* Q *is the directed tail-request row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and* H, C, P, N *are componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootUnaryAdmission`

Proof. Project the root stream and tail-request carriers already displayed for $\mathsf{Cauchy}^{\uparrow}$. Their shared rows expose finite StreamName windows before any tolerance, readback, or terminal seal can be read.

The directed request row supplies the unary admission point. It records one visible later-window request over the same finite stream surface, so the read is a UnaryHistory-facing row rather than a parameter transport over an ambient sequence.

The dyadic, regular readback, Real seal, and structural rows then repack the same finite route. No quotient stream, selected limit, host equality, countable choice, or ambient completeness row is exported. □

Theorem 5530.9 (Cauchy root filter-modulus compatibility)**.** *For every admitted root packet* $U = (S, Q, D, R, E, H, C, P, N)$*, any* $\mathsf{CauchyFilter}^{\uparrow}$ *or finite filter consumer that reads the tail request must use the same modulus-compatible route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The filter row may refine the visible window named by S *through* Q*, but it cannot replace* D *by an external tolerance oracle or replace* R *by a selected limit.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootFilterModulusCompatibility`

Proof. Start with the admitted packet of Definition 5888.8. The stream row S and directed request row Q are read before any filter-facing or modulus-facing datum.

Apply the existing Cauchy filter Real-seal route. It inserts the finite request row before the dyadic tolerance ledger, so a filter consumer reaches D and R only through the displayed tail request.

The structural rows preserve that order under hsame transport and Cont replay. Thus a finite filter may narrow the displayed request, but the modulus and regular readback remain the same BEDC rows. □

Theorem 5530.10 (Cauchy root RegSeqRat Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *handoff exported by a root* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the finite windows, directed request, dyadic tolerance ledger, and regular rational readback of Definition 5888.9. The Real seal* E *is available only after* S, Q, D, R, H, C, P, N *have been displayed, and it exports no quotient filter, selected real, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the filter-modulus compatibility theorem. It places the StreamName window, directed request, dyadic tolerance ledger, and RegSeqRat readback in a fixed order before any Real-facing boundary.

The handoff row is exactly R followed by E. A Real-facing consumer therefore receives rational readback through $\mathsf{RegSeqRat}^{\uparrow}$ first and may inspect the terminal seal only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient filter, selected real, choice schedule, host equality, or ambient completeness coordinate, the handoff remains a finite BEDC route. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootRegSeqRatRealHandoff`

The $\mathsf{Cauchy}^{\uparrow}$ packet enters the L10 dependency lattice through its sibling route before completion or filter consumers inspect it. Every such consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The Cauchy route is therefore a displayed sibling dependency path, not a floating completion primitive or a host-level filter object.

Theorem 5530.11 (Cauchy L10 sibling dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer in the L10 lattice receives finite observation windows from chapter 126, regular-rational readback from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any Real-facing approximation is public. The route exports no selected limit, quotient equality, host filter object, or ambient completeness principle.*

Proof. Project the accepted Cauchy packet through its finite observation surface. The first visible rows are the $\mathsf{StreamName}^{\uparrow}$ windows, so downstream readers cannot begin from a host-level limit predicate.

Next read the regular-rational sibling. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the finite regularity readback used by Cauchy consumers before the packet reaches the Real-facing boundary.

Finally expose the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed windows and readback rows, the route supplies finite windows and regular-rational data before public approximation, and it leaves no coordinate for selected limits, quotient equality, host filters, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.L10SiblingDependency.CauchyL10SiblingDependencyRoute`

Theorem 5530.12 (Cauchy route RegSeqRat tail inversion)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer that exposes a* $\mathsf{Real}^{\uparrow}$*-facing approximation through the displayed L10 sibling route must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tail row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. If either row is absent, the consumer has no public* $\mathsf{Real}^{\uparrow}$*-facing approximation route through the* $\mathsf{Cauchy}^{\uparrow}$ *L10 lattice.*

Proof. Start with Definition 5888.11. A public approximation route is visible only after the accepted Cauchy packet has passed through finite $\mathsf{StreamName}^{\uparrow}$ windows, regular-rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Read the windows first. They determine which finite observations are available to the consumer, but they do not by themselves supply a regular rational tail or a tolerance account.

Next inspect the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The regular-rational tail supplies the readback values, while the dyadic ledger supplies the tolerance discipline needed before those values can be read as a Real-facing approximation.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed tail and tolerance rows, deleting either row leaves no public route from the Cauchy consumer to a Real-facing approximation. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRouteRegSeqRatTailInversion`

Theorem 5530.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Theorem 5530.14 (Cauchy public L10 export)**.** *The public* $\mathsf{Cauchy}^{\uparrow}$ *export for L10 consumers is exactly the finite route already established by Definition 3413.1, Definition 5888.16, Definition 5888.18, and Definition 5888.10:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *route. A public consumer receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyPublicL10Export`

Proof. Start with the carrier boundary. Definition 5888.16 exhausts the accepted Cauchy packet by the Cauchy source row, finite windows, dyadic tolerances, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Use the scoped filter-modulus route. Definition 5888.18 shows that filter, modulus, Real-facing, and completion-facing consumers all read the same finite path rather than an external completion object.

Finally apply the root readback handoff of Definition 5888.10. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same window and tolerance rows. Thus the public L10 export is the displayed route itself, with no additional host equality, selection, choice, quotient, or completeness principle. □

Theorem 5530.15 (Cauchy forward L10 sum-tail route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer that reaches the regular sum tail-estimate surface must first pass through the regular rational source of chapter 301, the terminal real-name boundary of chapter 38, and the finite tail-estimate packet of chapter 7387. The route is an L10 dependency path through displayed StreamName, DyadicRatCore,*

RegSeqRat, RegularCauchySum, Real, BHist, hsame, Cont, Pkg, *and* NameCert *rows; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child routes. The core carrier exposes finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real-facing seal before any downstream completion or arithmetic consumer is available.

The regular sum tail-estimate chapter consumes exactly that surface twice, once for each regular source row. Its tail-estimate ledger then adds the RegularCauchySum row and finite error budget without creating a detached limit object.

Thus the forward L10 route from the Cauchy hub to the sum-tail packet is the displayed dependency path through RegSeqRat, Real, and the finite tail-estimate surface. Since each row is already a finite NameCert coordinate, no excluded host or choice principle can be read from the hub. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyBaseForwardL10Route`

Theorem 5530.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 5530.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 5530.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 5531

A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on uniform embeddings, the CauchyCompletionEmbeddingReflection$^{\uparrow}$ packet records the finite route by which a dense completion embedding is reflected through the separated completion interface. It sits over the dense completion embedding of chapter 5893, the dense uniform embedding surface of chapter 1651, the functorial completion route of chapter 1937, the separated reflection boundary of chapter 3543, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real seal of chapter 38. The packet names the completion-embedding reflection route only; it does not select representatives, form a quotient completion outside the displayed certificate, or import ambient completeness.

Definition 5531.1 (CauchyCompletionEmbeddingReflection carrier)**.** *A* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source approximation row whose metric or uniform histories feed the completion embedding.*
2. *G is the* CauchyCompletionDenseEmbedding$^{\uparrow}$ *row naming the dense completion embedding and its image-window requests.*
3. *K is the* DenseUniformEmbedding$^{\uparrow}$ *row that exposes uniform density before the completion target is read as separated.*
4. *R is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row transporting maps and comparison windows through completion.*
5. *S is the* SeparatedCompletionReflection$^{\uparrow}$ *row that reflects the target route through the separated completion boundary.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row attached to the same finite approximation windows.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row reached after A, G, K, R, S, D have been displayed.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the dense embedding, separated reflection, dyadic tolerance, and regular-rational readback rows are fixed.*

9. *H records componentwise* hsame *transport for the analytic rows and the structural rows.*
10. *C records displayed* Cont *replay of the same embedding-reflection route.*
11. *P is the* Pkg *provenance over* CauchyCompletionEmbeddingReflection$^{\uparrow}$, CauchyCompletionDenseEmbedding$^{\uparrow}$, DenseUniformEmbedding$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, SeparatedCompletionReflection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected representatives, countable choice, a detached separated completion, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.TasteGate.CauchyCompletionEmbeddingReflectio
single carrier alignment

Theorem 5531.2 (CauchyCompletionEmbeddingReflection NameCert obligations)**.** *For every accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier* $J = (A, G, K, R, S, D, Q, E, H, C, P, N)$*, the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed approximation row, dense completion embedding row, dense uniform embedding row, completion functoriality row, separated reflection row, dyadic tolerance row, regular-rational readback row, terminal real seal row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing read must pass through* $A, G, K, R, S, D, Q, E, H, C, P, N$ *in that order, and no read may bypass the separated reflection boundary.*

Proof. Project the carrier of Definition 5889.1. Its analytic rows are A, G, K, R, S, D, Q, E, followed by the structural rows H, C, P, N.

The dense route begins with A and G. The approximation row supplies the finite source histories, while the CauchyCompletionDenseEmbedding$^{\uparrow}$ row records the image-window requests through which completion consumers may inspect the dense embedding.

The rows K, R, S then bind the same route to dense uniform embedding, functorial completion transport, and separated reflection. A completion-facing consumer therefore reaches the separated target only after the dense route and functorial route have been displayed.

The rows D, Q, E provide dyadic tolerance, regular-rational readback, and the terminal real seal. Finally H, C, P, N transport, replay, source, and name the same packet, so carrier exhaustion blocks every excluded quotient, selector, detached completion, and ambient-completeness route. □

Theorem 5531.3 (CauchyCompletionEmbeddingReflection handoff)**.** *Every downstream consumer of an accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *packet receives the dense completion embedding only through the finite handoff*

$$\text{CauchyCompletionDenseEmbedding}^{\uparrow} \longmapsto \text{DenseUniformEmbedding}^{\uparrow} \longmapsto \text{CauchyCompletionFunctoriality}^{\uparrow} \longmapsto \text{Separat}$$

Thus the consumer obtains a reflected separated-completion route with finite stream windows, dyadic tolerance, regular-rational readback, and a terminal real seal; it does not obtain a quotient-completion representative, a selected dense point, countable choice, or ambient completeness.

Proof. Start with Definition 5889.2. The NameCert surface forces every public read to begin with the approximation row and the dense completion embedding row.

Read the sibling chapters named in the statement along the displayed order. The DenseUniformEmbedding$^{\uparrow}$ row supplies the uniform dense image, CauchyCompletionFunctoriality$^{\uparrow}$ transports the route through

completion, and $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ is the only separated target boundary available to the packet.

The finite observation route is then exposed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows are exactly the rows D, Q, E together with the stream-window data carried by A, G, K, R, S.

Since the carrier has only $A, G, K, R, S, D, Q, E, H, C, P, N$, the structural rows can preserve and name the handoff but cannot supply a quotient representative, selected dense point, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5889.1, Definition 5889.2, and Definition 5889.3 seed the dense completion embedding reflection route through separated completion, finite stream windows, dyadic tolerance, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, countable choice, detached separated completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense completion embedding exposure, dense uniform embedding compatibility, completion functoriality, separated reflection, finite-window readback, dyadic tolerance, regular-rational handoff, real sealing, and nonescape for BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.

Constructive story: A $\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}$ packet is generated from a source approximation row, a dense completion embedding row, a dense uniform embedding row, a completion functoriality row, a separated reflection row, dyadic tolerance, regular-rational readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5532

A Concrete Naming Certificate for $\mathsf{BarRecursion}^{\uparrow}$

Following Spector and Troelstra–van Dalen, the $\mathsf{BarRecursion}^{\uparrow}$ packet records finite bar-recursive witnesses on the Real and Completion route. It reads the bar discipline of $\mathsf{BrouwerBarInduction}^{\uparrow}$ from chapter 1675, the finite-cover boundary of $\mathsf{FanTheorem}^{\uparrow}$ from chapter 400, the inspected stream windows of $\mathsf{StreamName}^{\uparrow}$ from chapter 126, the rational readback of $\mathsf{RegSeqRat}^{\uparrow}$ from chapter 301, and the terminal seal of $\mathsf{Real}^{\uparrow}$ from chapter 38. The packet names finite witness production without adding countable choice, a selected infinite branch, quotient stream equality, host real equality, or ambient Real completeness.

Definition 5532.1 (BarRecursion carrier)**.** *A* $\mathsf{BarRecursion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, R, W, K, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the barred-spread source row, recording the* $\mathsf{BrouwerBarInduction}^{\uparrow}$ *or* $\mathsf{FanTheorem}^{\uparrow}$ *finite bar surface on which the recursion is allowed to descend.*
2. *R is the recursion step ledger, recording the finite predecessor, stopping, and successor requests used by the bar-recursive witness.*
3. *W is the finite branch window, exposing only the* $\mathsf{StreamName}^{\uparrow}$ *prefixes inspected before the bar row accepts.*
4. *K is the control-functional read, recording the finite query schedule by which the recursion asks for further prefix observations.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, displaying the rational data produced by the accepted finite recursion route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the bar source, step ledger, branch window, control read, and rational readback have been accepted.*
7. *H records componentwise* hsame *transport for the barred-spread source, recursion ledger, finite window, control read, rational readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for all consumer reads of $S, R, W, K, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BarRecursion}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BarRecursion}^\uparrow}$ *row.*
The classifier compares accepted carriers only by these rows, their componentwise hsame *transports, their displayed* Cont *replays, their package provenance, and their local naming rows.*

Theorem 5532.2 (BarRecursion NameCert obligations)**.** *For an accepted* $\mathsf{BarRecursion}^\uparrow$ *carrier*

$$B = (S, R, W, K, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BarRecursion}^\uparrow}$ *surface consists exactly of carrier admission for the displayed rows, barred-source exposure, recursion-step exactness, finite-window discipline, control-functional read discipline, rational readback, Real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $S, R, W, K, Q, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 5532.1. The accepted packet exposes the barred-spread source, recursion ledger, finite branch window, control-functional read, regular rational readback, Real seal, transport row, replay row, provenance row, and local naming row.

The barred-source and step obligations are read from S and R. The source row fixes the finite bar surface supplied by $\mathsf{BrouwerBarInduction}^\uparrow$ or $\mathsf{FanTheorem}^\uparrow$, while the recursion ledger records only the predecessor, stopping, and successor requests displayed by that finite surface.

The witness-read obligations are then projected from W, K, Q, E. The branch window W bounds the inspected stream prefixes, K records the finite control queries, Q records the regular rational readback, and E is reachable only after those rows have been accepted.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for countable choice, a selected infinite branch, quotient stream equality, host real equality, or ambient Real completeness, the listed rows are exactly the NameCert obligations. □

Theorem 5532.3 (BarRecursion Real-completion non-escape)**.** *Every* $\mathsf{Real}^\uparrow$ *or Completion-facing consumer read of an accepted* $\mathsf{BarRecursion}^\uparrow$ *carrier factors through the finite route*

$$S, R, W, K, Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer may read the barred source, recursion step ledger, finite branch window, control-functional schedule, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows, but it cannot read countable choice, a selected infinite branch, quotient stream equality, host real equality, or ambient Real completeness from the $\mathsf{BarRecursion}^\uparrow$ *packet.*

Lean4 ✓ `BEDC.Derived.BarRecursionUp.BarRecursionRealCompletionNonescape`

Proof. Open the carrier of Definition 5532.1. A Real-facing read has access only to the displayed rows $S, R, W, K, Q, E, H, C, P, N$, and the terminal seal row E is placed after the barred source, recursion ledger, window, control read, and rational readback.

Apply the NameCert obligations of Definition 5532.2. They force consumer reads to respect the same row order: the recursion step ledger is tied to the barred source, the branch window bounds the inspected prefixes, and the rational readback is produced before the Real seal can be exposed.

Transport and replay do not add new objects. The rows H and C preserve the displayed finite route, while P and N only source and name it. Hence every Real or Completion consumer factors through the stated finite prefix and the refused ambient objects remain outside the public surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarRecursion}\uparrow}$.

Constructive story: A $\mathsf{BarRecursion}^{\uparrow}$ packet is generated from a barred-spread source, a recursion step ledger, a finite branch window, a control-functional read, a RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BarRecursion}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5532.1, Definition 5532.2, and Definition 5532.3 seed the finite bar-recursive witness surface used by Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BarRecursion}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close countable choice, selected infinite branches, quotient stream equality, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, barred-source exposure, recursion-step exactness, finite-window discipline, control-functional read discipline, rational readback, Real-seal handoff, transport stability, replay stability, provenance, and local naming for BEDC.Derived.BarRecursionUp.

Chapter 5533

A Concrete Naming Certificate for $\mathsf{SnakeLemma}^{\uparrow}$

Following Cartan–Eilenberg and Weibel homological algebra, the $\mathsf{SnakeLemma}^{\uparrow}$ packet records the finite diagram chase that turns two exact rows and compatible vertical maps into the connecting boundary route. It sits over the abelian-category carrier of chapter 176, the homology carrier of chapter 98, the spectral-sequence consumer surface that exposes $\mathsf{DerivedFunctor}^{\uparrow}$ rows in chapter 178, and the derived-functor interface of chapter 177. The packet names a displayed exactness boundary; it does not import quotient-object equality, kernel-cokernel object extensionality, or an ambient abelian-category diagram oracle.

Definition 5533.1 (Snake lemma carrier)**.** *A* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row carrying the additive kernel-cokernel surface in which the diagram is read.*
2. *R_0 and R_1 are the upper and lower exact-row ledgers, each exposing three displayed object rows, two morphism rows, and the local image-kernel containment needed for exactness.*
3. *V is the vertical-compatibility row, recording the three displayed vertical morphisms and the two commutative squares that align R_0 with R_1.*
4. *K is the kernel-row ledger for the left and middle vertical maps.*
5. *C is the cokernel-row ledger for the middle and right vertical maps.*
6. *E is the exactness-boundary row, recording that the kernel leg, connecting boundary, and cokernel leg form the displayed exact boundary chain.*
7. *∂ is the connecting-boundary row from the right kernel surface to the left cokernel surface.*
8. *H records componentwise* hsame *transport for the abelian-category source, exact rows, vertical squares, kernel rows, cokernel rows, boundary rows, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from the two exact rows and vertical squares to $K, C, E, \partial, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{SnakeLemma}^{\uparrow}$*,* $\mathsf{AbelianCat}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* $\mathsf{DerivedFunctor}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

11. L is the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted snake-lemma packets only through these rows. It contains no coordinate for host quotient representatives, unbounded diagram search, selected elements outside the displayed histories, or equality of ambient abelian-category objects.

Lean4 ✓ `BEDC.Derived.SnakeLemmaUp.TasteGate.SnakeLemmaTasteGate single carrier alignment`

Theorem 5533.2 (Snake lemma NameCert obligations)**.** *For every accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *surface consists exactly of abelian-category source exposure through A, upper and lower exact-row exposure through R_0, R_1, vertical-square compatibility through V, kernel-row exposure through K, cokernel-row exposure through C, exact-boundary exposure through E, connecting-boundary exposure through ∂, componentwise transport through H, displayed replay through T, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 5890.1. The rows A, R_0, R_1, V expose the ambient additive kernel-cokernel surface, the two exact rows, and the square compatibility before any boundary read is allowed.

The kernel and cokernel ledgers are then read from K and C. They supply the visible source and target surfaces for the connecting row ∂, while E records the exactness statement tying the kernel leg, boundary leg, and cokernel leg into one chain.

Transport, replay, provenance, and local naming preserve this same finite packet through H, T, P, L. Since no excluded quotient representative, host object equality, or unbounded diagram search appears among the coordinates, the listed rows exhaust the local certificate obligations. □

Theorem 5533.3 (Snake lemma exact boundary)**.** *Every downstream* $\mathsf{Homology}^{\uparrow}$ *or* $\mathsf{DerivedFunctor}^{\uparrow}$ *consumer of an accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *packet reads the boundary route*

$$\ker(v_2) \longmapsto \operatorname{coker}(v_0)$$

only through the displayed chain

$$\mathsf{AbelianCat}^{\uparrow} \longmapsto (R_0, R_1, V) \longmapsto (K, C) \longmapsto (\partial, E) \longmapsto (H, T, P, L).$$

Thus the consumer receives one finite exact-boundary packet of abelian-category source rows, exact-row ledgers, vertical-square ledgers, kernel and cokernel rows, the connecting-boundary row, exactness-boundary evidence, transport, replay, provenance, and local naming.

Proof. Start with the abelian-category source row A. The two exact rows R_0, R_1 and the vertical-compatibility row V expose exactly the diagram data on which the classical snake-lemma chase depends.

Read K and C next. These rows identify the displayed kernel and cokernel surfaces that the connecting boundary is permitted to connect, without choosing representatives outside the finite histories.

The boundary row ∂ and exactness row E then supply the only route from the right kernel surface to the left cokernel surface. A homology or derived-functor consumer therefore reads the boundary as a named packet, not as an ambient quotient construction.

Finally H, T, P, L keep the same route stable under componentwise history equality, displayed replay, provenance, and local naming. No quotient-object equality, host diagram oracle, or unbounded element chase is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SnakeLemma}^{\uparrow}$ packet is generated from an AbelianCat source row, two exact-row ledgers, vertical-square compatibility rows, kernel-row and cokernel-row ledgers, a connecting-boundary row, an exactness-boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5890.1, Definition 5890.2, and Definition 5890.3 seed the finite snake-lemma packet over abelian-category source rows, exact-row ledgers, vertical-square compatibility, kernel rows, cokernel rows, connecting-boundary rows, exactness-boundary evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-object equality, host abelian-category equality, unbounded diagram search, or ambient element-chase completeness.

Upgrade path: Obligation closure requires checked carrier admission, exact-row exposure, vertical-square compatibility, kernel-row exposure, cokernel-row exposure, connecting-boundary construction, exactness-boundary preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SnakeLemmaUp.

Chapter 5534

A Concrete Naming Certificate for $\mathsf{RealTailDiameterGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet records the finite gauge by which a real-facing consumer reads a regular rational tail only after the displayed tail-diameter, dyadic tolerance, stream-window, and readback rows have been exposed. It sits after the tail-diameter witness of chapter 1746, the dyadic tolerance core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a non-quotient L10 gauge object for Real and Completion consumers; it is not host real equality, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5534.1 (Real tail-diameter gauge BHist source)**.** *A* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, T, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the real-facing boundary that the gauge is allowed to serve.*
2. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row, recording the inspected tail threshold, two visible tail indices, and the displayed diameter bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the same tail-diameter ledger.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite source observations inspected by T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window and tolerance data before any real-facing read is accepted.*
6. *H records componentwise* hsame *transport for the seal, tail-diameter ledger, dyadic tolerance, window, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a real-facing gauge request through E, T, D, W, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealTailDiameterGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, T, D, W, R, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealTailDiameterGaugeUp.TasteGate.RealTailDiameterGaugeTasteGate single carrier alignment`

Theorem 5534.2 (Real tail-diameter gauge NameCert obligation surface)**.** *For every accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier*

$$G = (E, T, D, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*.*

Proof. Project the carrier of Definition 5891.1. The analytic rows are the Real seal E, tail-diameter ledger T, dyadic tolerance D, finite window W, and regular rational readback R.

The gauge obligation is read only through T, D, W, R. The tail-diameter ledger records the visible tail request and bound, the dyadic row supplies the tolerance budget, the stream row exposes the inspected finite windows, and the regular row repacks the same data for rational readback.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Since the carrier has no coordinate for the excluded host or completion objects, the displayed rows exhaust the local NameCert surface. □

Theorem 5534.3 (Real tail-diameter gauge RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *packet factors through*

$$W, D \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real and Completion route may read the finite stream window, dyadic tolerance ledger, tail-diameter gauge, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming rows, but it receives no host real equality, quotient stream equality, selected limit, trichotomy decision, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge check.*

Proof. Start with an exported regular-rational or real-facing read. The only finite observations available to the packet are the $\mathsf{StreamName}^{\uparrow}$ row W and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D.

The tail-diameter ledger T then binds those observations to a displayed tail request and diameter bound. This row is the gauge that prevents a consumer from replacing the finite tail evidence by a quotient stream or a selected limit.

Only after T is displayed does the handoff expose the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row E. The readback and seal are therefore downstream from the same finite gauge data.

Finally H, C, P, N transport, replay, source, and name the route. Since the excluded objects do not occur in the carrier, no consumer can bypass the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet is generated from a RealUp seal row, a CauchyTailDiameterUp ledger, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, seed)

Scope closed: Definition 5891.1, Definition 5891.2, and Definition 5891.3 seed the finite real-facing tail-diameter gauge for regular rational and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, countable choice, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, Real seal confinement, CauchyTailDiameter ledger exactness, DyadicRatCore tolerance readback, StreamName window coverage, RegSeqRat handoff, transport stability, continuation replay, provenance non-escape, and local naming for BEDC.Derived.RealTailDiameterGaugeUp.

This hub routes the EffectiveRealCompactIntervalSearch$^{\uparrow}$ packet through chapter 1250, chapter 1767, and chapter 5899 before the child files expose the compact interval search carrier.

Chapter 5535

A Concrete Naming Certificate for $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet records the finite search surface by which a continuous real-valued map on a displayed closed bounded interval is inspected through compact finite nets, uniform-modulus data, and interval bisection rows before a downstream extremum consumer reads a certified bracket. It sits over the real-name seal of chapter 38, the interval endpoint carrier of chapter 1591, the compact-net selector of chapter 914, the graph and pointwise rows of chapter 130, and the tolerance handoff of chapter 369. The packet gives the constructive extreme-value route a finite searchable carrier; it does not assert classical Heine–Borel compactness, choose a host maximum, or identify quotient real functions.

The search packet consumes the compact interval finite-net route before it exports bisection or extreme-value handoffs. Its selector row P is read downstream of the closed-bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; the bisection ledger then refines those displayed cells instead of introducing an open-cover compactness argument or a selected supremum.

Definition 5535.1 (Effective real compact-interval search carrier)**.** *An* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, U, P, B, V, H, C, Q, N)$$

with the following visible rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint and containment row whose endpoints are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for a real-valued map over the interval source named by I.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance handoff for F on the displayed interval source.*
4. *P is the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row that lists the rational mesh cells and selected centers used by the search.*
5. *B is the bisection ledger, a finite prefix tree of interval halves whose cells refine P at the requested tolerance.*
6. *V is the value-comparison ledger storing the displayed* $\mathsf{Real}^{\uparrow}$ *value rows and rational error budgets attached to the cells of B.*

7. *H stores componentwise* hsame *transport for the interval, continuous-map, uniform-modulus, selector, bisection, value-comparison, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an extremum-search request through $I, F, U, P, B, V, H, Q, N$.*
9. *Q is the* Pkg *provenance over* EffectiveRealCompactIntervalSearch$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *row.*

The classifier compares accepted search carriers only through $I, F, U, P, B, V, H, C, Q, N$. No open-cover compactness theorem, selected host maximum, excluded-middle comparison on all real values, quotient function equality, or ambient completeness coordinate is part of the carrier.

Theorem 5535.2 (Effective real compact-interval search obligations)**.** *For every accepted* EffectiveRealCompactInte *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, value-comparison ledger, transport, replay, provenance, and local naming;*
2. *source agreement, requiring I, F, U, and P to name the same displayed interval source before any cell is searched;*
3. *selector exactness, requiring the search cells in B to refine the finite* ProbeBundle *centers and tolerances exposed by P;*
4. *value-row binding, requiring every value comparison in V to be attached to a cell of B, a displayed graph row of F, and the error budget supplied by U;*
5. *bisection stability, requiring interval refinement to preserve endpoint containment and the selected candidate cell under componentwise* hsame *transport;*
6. *extremum handoff, requiring any lower-bound, upper-bound, near-minimum, or near-maximum read to pass through B, V, H, C, Q, N rather than through a host maximum selector;*
7. *non-escape, excluding open-cover compactness, classical Heine–Borel selection, trichotomy on arbitrary real values, quotient function equality, and ambient* Real$^{\uparrow}$ *completeness from the local packet.*

No effective compact-interval search obligation is stored outside these rows.

Proof. Project the carrier of Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, and the finite selector row is P.

The search may begin only after these rows agree on the same displayed interval source. The bisection ledger B then refines the selector cells supplied by P, so each later comparison is attached to a visible interval cell rather than to an external point choice.

The value ledger V reads graph values from F with the error budget delivered by U. This makes every candidate comparison a finite statement about named cells and rational tolerances.

Finally the extremum-facing read uses H, C, Q, N to replay and name the same route. Since the carrier has no coordinate for open-cover compactness, a host maximum, global real trichotomy, quotient function equality, or ambient completeness, none of those objects can enter the local certificate. □

Theorem 5535.3 (Effective real compact-interval bisection induction)**.** *For an accepted* EffectiveRealCompactInterv *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, every finite bisection stage in B is obtained by induction from the endpoint row I and the selector row P. The induction step replaces one displayed interval cell by its two named halves, transports the corresponding* ContinuousMap$^{\uparrow}$ *and* UniformModulus$^{\uparrow}$ *reads through H, and extends V only by value rows attached to those halves.*

Proof. The base stage is the interval cell named by I together with the first finite-net read in P. Its value ledger is the corresponding displayed graph read from F with the tolerance supplied by U.

For the induction step, choose the cell already selected by the finite ledger. The midpoint and half-interval endpoints are read as interval rows below I, and the selector exactness obligation identifies their centers with the relevant ProbeBundle entries of P.

Transport along H keeps the continuous-map graph and uniform-modulus handoff on the same source. The new value rows are therefore added to V only for the two displayed halves, so the induction never leaves the finite bisection tree. □

Theorem 5535.4 (Effective real compact-interval extreme-value handoff)**.** *Every extremum-search consumer reading an accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *carrier factors through*

$$I, F, U, P \longmapsto B, V \longmapsto H, C, Q, N.$$

The consumer receives a finite bracket or near-extremal cell family whose error budget is supplied by UniformModulus$^{\uparrow}$ *and whose candidates are selected by* CompactNetModulusSelector$^{\uparrow}$*. It receives no classical maximum point, open-cover compactness proof, quotient equality of functions, global trichotomy on* Real$^{\uparrow}$ *values, or ambient completeness assertion.*

Proof. Begin with the carrier projection. The only source data available to the consumer are the interval row I, continuous graph row F, uniform-modulus row U, and compact-net selector row P.

Apply Definition 5892.3. Every searched cell and every candidate value is produced inside the finite bisection ledger B and value ledger V.

The final handoff reads those ledgers through H, C, Q, N. Applying Definition 5892.2, any route that asks for a host maximum, open-cover compactness, quotient function equality, global real trichotomy, or ambient completeness is outside the accepted packet. □

Theorem 5535.5 (Effective real compact interval search finite-net handoff)**.** *Every accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *search packet reads a compact interval finite-net certificate, a located interval bisection row, and the* Real$^{\uparrow}$ *comparison seal before exporting the extreme-value handoff. The route passes through the closed bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; it exports no open-cover compactness, selected supremum, host trichotomy, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The compact interval finite-net data enter through the selector row P, whose cells are read only after the displayed closed bounded interval source I is fixed.

Apply Definition 5892.3. The bisection row B refines the finite cells supplied by P, and every value comparison in V is attached to a displayed graph row of F with the tolerance budget from U.

Apply Definition 5892.4. The final route reads B, V, H, C, Q, N after the finite-net source rows, so the exported extreme-value handoff is a finite search certificate rather than an open-cover compactness, supremum-selection, trichotomy, quotient-equality, or completeness principle. □

Theorem 5535.6 (Effective real compact interval search L10 sibling dependency)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must display the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint carrier, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite net, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *selector, the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *row, and the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement before exporting any extremum handoff. The route is the finite-net and bisection dependency route of Definition 5892.5, not an isolated kernel-only search packet.*

Proof. Project the carrier definition, Definition 5892.1. The rows I, F, U, P, B, V expose the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, and value ledger, while the Real-facing seal is read through the displayed interval and value rows.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before any bisection or extreme-value consumer can read the packet.

The bisection row B then supplies the located refinement consumed by the extremum handoff. The structural rows H, C, Q, N only transport, replay, source, and name that same sibling-dependent route, so no extremum handoff is exported before the listed L10 siblings are displayed. □

Theorem 5535.7 (Effective real compact interval search inverse modulus bisection)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, a displayed* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row* U *and a* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row* P *determine a finite bisection depth in* B *whose dyadic cell diameter is below the requested tolerance. The depth is read only through* P, B, V, H, C, Q, N*: the value ledger* V *attaches comparisons to cells at that depth,* H *transports the same cells under* hsame*,* C *replays the finite request, and* Q, N *package and name the route. The construction exports no open-cover compactness proof, no host maximum, no quotient equality of functions, no trichotomy on* $\mathsf{Real}^{\uparrow}$ *values, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the accepted search carrier. The uniform-modulus row U supplies the requested tolerance, while the selector row P displays the finite mesh data that the search is allowed to refine.

Apply Definition 5892.3. Each step of B replaces a displayed cell by named dyadic halves, so repeated refinement gives a finite depth at which the visible cell diameter is below the tolerance requested by U.

Apply Definition 5892.5. The finite-net row P remains the source of the cells refined by B, and the value ledger V records only comparisons attached to those cells.

Finally H, C, Q, N transport, replay, source, and name the same finite depth-control route. Because the carrier has no open-cover compactness coordinate, host maximum selector, quotient function equality coordinate, trichotomy oracle, or ambient completeness row, the inverse-modulus read remains finite interval-search data. □

Theorem 5535.8 (Effective real compact interval search finite-grid stability)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, the displayed finite-net cell selected by* P *and refined by the located bisection ledger* B *is preserved under componentwise* hsame *transport before any extreme-value handoff is exported. Thus every Real-facing extremum consumer reads the same finite cell, value row, and bisection row through* P, B, V, H, C, Q, N*, and receives no open-cover compactness proof, selected host maximum, or* $\mathsf{Real}^{\uparrow}$ *trichotomy.*

Proof. Project the carrier of Definition 5892.1. The finite-net cell is displayed by P, its bisection refinement is recorded in B, and the attached value comparisons are stored in V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger is consumed.

Apply the inverse-modulus bisection theorem, Definition 5892.7. The bisection depth and the cell read remain tied to P, B, V, while H transports the same displayed rows and C replays the same finite request.

The carrier has no coordinate for open-cover compactness, host maxima, quotient function equality, Real trichotomy, or ambient completeness. The exported extreme-value handoff therefore preserves only the finite-grid cell and located bisection row named by the packet. □

Theorem 5535.9 (Effective real compact interval search Bishop cover link)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$ *reads the Bishop interval-cover finite cover ledger of chapter 5124 before it exports any bisection or extreme-value handoff. The Bishop route supplies the finite interval-cover cells refined by* P *and* B*, while the local search packet preserves the continuous-map, uniform-modulus, value-comparison, transport, replay, provenance, and naming rows. The exported route contains no open-cover compactness theorem, selected supremum, host trichotomy, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, the finite selector row is P, the bisection ledger is B, and the value ledger is V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search packet to read the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger can be consumed.

Apply the grid-stability theorem, Definition 5892.8. The selected finite cell is preserved through P, B, V, H, C, Q, N, so the Bishop interval-cover ledger supplies interval-cover cells for the same displayed route rather than a separate compactness principle.

The remaining rows H, C, Q, N transport, replay, source, and name the same finite cover-to-search handoff. Since the carrier has no row for open-cover compactness, selected suprema, host trichotomy, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot be exported by the bisection or extreme-value consumer. □

Theorem 5535.10 (Effective real compact interval search RegularCauchyGluing dependency)**.** *Every completion-facing extremum-search read of an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must expose the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route of chapter 5899 after the finite bisection and value ledger are displayed and before any* $\mathsf{Real}^{\uparrow}$*-facing seal is consumed. The dependency is read through the finite-net handoff of Definition 5892.5 and the grid-stability route of Definition 5892.8; it is not supplied by quotient stream equality, selected host limits, countable choice, or ambient real completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The completion-facing rows available to the search are the bisection ledger B, the value-comparison ledger V, and the structural replay rows H, C, Q, N.

Apply Definition 5892.5. The finite-net route fixes the displayed interval cells and their Real-valued comparison windows before an extremum consumer can read the packet.

Apply Definition 5892.8. The same finite cell, bisection row, and value row are preserved under componentwise hsame transport, so the completion-facing read has a stable regular-Cauchy window to hand to the gluing sibling.

Route that window through $\mathsf{RegularCauchyGluing}^{\uparrow}$ before the terminal Real-facing seal is consumed. Since the carrier has no quotient stream-equality row, selected host limit, countable-choice

selector, or ambient-completeness coordinate, the gluing dependency is the only completion-facing route available after B and V are displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5892.1, Definition 5892.2, Definition 5892.3, Definition 5892.4, Definition 5892.5, Definition 5892.7, and Definition 5892.8 seed the finite interval-search carrier from displayed Real, closed-bounded-interval, compact-net-selector, continuous-map, finite-net, and uniform-modulus rows to extremum-facing bracket and depth-control handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical Heine–Borel compactness, selected host maxima or minima, quotient equality of functions, global real trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, selector exactness, value-row binding, bisection stability, extremum handoff, and non-escape for BEDC.Derived.EffectiveRealCompactIntervalSearchUp.

Constructive story: An $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source, ContinuousMapUp graph rows, a UniformModulusUp tolerance handoff, a CompactNetModulusSelectorUp finite-net row, a finite bisection ledger, displayed RealUp value comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ search packet consumes the closed-bounded interval net of chapter 1250, the finite epsilon-net surface of chapter 1767, the compact-net modulus selector of chapter 914, and the located interval bisection route of chapter 1711 before exporting its extremum handoff. Thus a downstream consumer sees the finite-net and bisection dependencies as displayed route data, not as an open-cover compactness proof or a selected host extremum. The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ route hands its finite bisection rows, interval-value rows, and compact-window certificates to the compatible-window route of chapter 5899 before any $\mathsf{Real}^{\uparrow}$ seal consumer reads a glued local search window. This dependency keeps each L10 finite-window consumer downstream from the displayed compact search packet, so the gluing step reads certified local windows rather than an ambient completeness or choice principle.

Chapter 5536

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on dense embeddings, the $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet records the finite dense-source route by which a Cauchy completion consumer reads a source object inside its completion boundary. It sits beside the dense uniform embedding surface of chapter 1651, the functorial Cauchy-completion route of chapter 1937, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies dense embedding data for completion consumers without asserting quotient completion equality, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5536.1 (Cauchy completion dense-embedding carrier)**.** *A* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the dense source row. It records the source-side Cauchy observations that a completion consumer may approximate by displayed source data.*
2. *G is the embedding graph row. It records the displayed source-to-completion graph and the finite image-window reads used by dense embedding consumers.*
3. *K is the Cauchy completion row, read through the completion-facing route associated with* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite request windows inspected by the source and completion sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source-image windows with completion windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after A, G, K, S, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the dense source, embedding graph, completion, window, tolerance, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dense-source request through $A, G, K, S, D, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseEmbeddingUp.CauchyCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 5536.2 (Cauchy completion dense-embedding NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier*

$$J = (A, G, K, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the dense source row, embedding graph row, Cauchy completion row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing dense read must expose A, G, K, S, D, Q before E, H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5893.1. The dense source, embedding graph, Cauchy completion row, finite windows, tolerance ledger, regular readback, and Real seal are exactly the displayed analytic coordinates.

The dense-source route is read through A, G, K. A completion consumer first fixes the source row, then the embedding graph row, and only then the completion-facing row associated with the Cauchy completion route.

The finite observation route is read through S, D, Q. The stream windows expose inspected requests, the dyadic ledger supplies the tolerance budget, and the regular rational readback repacks the same observations before any Real-facing seal is visible.

Finally E, H, C, P, N preserve and name the route. Since quotient completion equality, selected representatives, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5536.3 (Cauchy completion dense-embedding Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer exported by an accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{DenseUniformEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionFunctoriality}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More precisely, the consumer reads the dense source row A, embedding graph G, Cauchy completion row K, finite windows S, tolerance ledger D, regular readback Q, terminal seal E, componentwise transport H, displayed replay C, provenance P, and local name N before the real-facing boundary is exposed.

Proof. Start with an exported Real-facing or completion-facing read. The NameCert obligations of Definition 5893.2 force the read through A, G, K before the finite observation rows can be consumed.

Read the sibling routes next. The dense source is the $\mathsf{DenseUniformEmbedding}^{\uparrow}$ surface, the completed-map side is routed through $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, and the finite observations are the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows displayed by the carrier.

Only after those rows have been displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal row E be read. The structural rows H, C, P, N transport, replay, source, and name the same route, so the handoff exports no quotient completion equality, selected representative, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a dense source row, embedding graph row, Cauchy completion row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5893.1, Definition 5893.2, and Definition 5893.3 seed the dense-source route for Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense source exposure, embedding graph exactness, Cauchy completion handoff, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionDenseEmbeddingUp.

Chapter 5537

A Concrete Naming Certificate for $\mathsf{RiemannRefinementCriterion}^{\uparrow}$

Following Bishop and Bridges on Riemann–Darboux integration, the $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet records the finite refinement criterion that lets a tagged Riemann-sum consumer compare upper Darboux rows against a gauge-local partition route. It sits after the tagged partition carrier of chapter 4296, the upper Darboux-sum carrier of chapter 4083, the Riemann sum gauge packet of chapter 4115, and the Cousin cover carrier of chapter 3426. The packet supplies the finite refinement surface consumed by integrability criteria without importing host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5537.1 (Riemann refinement criterion carrier)**.** *A* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row shared by the refinement comparison.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *source row, including the finite cell order, tag containment, and mesh exactness visible before refinement is read.*
3. *U is the* ***??*** *row whose upper-sum ledger is compared against the refined tagged route.*
4. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row, carrying the positive rational gauge, mesh ledger, and finite tagged-sum handoff.*
5. *C is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge-cover row supplying the finite local cover route that admits the refined cells.*
6. *M is the common-refinement mesh row, listing the displayed refinement map from the tagged partition row into the gauge-cover and upper-sum rows.*
7. *D is the Darboux-gap lock row, recording the finite upper-bound comparison that remains visible after the common refinement has been read.*
8. *W is the window and readback row for interval endpoints, tags, gauges, cover cells, and upper-sum values.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tagged partition, gauge, Cousin cover, refinement mesh, and Darboux-gap rows are displayed.*

10. *H records componentwise* hsame *transport for interval, partition, upper-sum, gauge, cover, refinement, Darboux-gap, window, seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a refinement request through $I, T, U, G, C, M, D, W, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, *??*, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows. It has no coordinate for host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannRefinementCriterionUp.RiemannRefinementCriterionTasteGate single carrier alignment`

Theorem 5537.2 (Riemann refinement criterion NameCert obligations)**.** *For every accepted* RiemannRefinementCr *carrier $R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval, tagged partition, upper Darboux-sum, Riemann sum gauge, Cousin cover, refinement mesh, Darboux-gap lock, window, seal, transport, replay, provenance, and local naming rows.*

Every refinement-facing consumer must read the tagged partition row T from chapter 4296, the upper-sum row U from chapter 4083, the gauge row G from chapter 4115, and the gauge-cover row C from chapter 3426 before the common-refinement mesh row M and Darboux-gap lock row D are available.

Proof. Project the carrier of Definition 5894.1. The tuple displays the only rows available to a refinement consumer: interval source, tagged partition, upper-sum, gauge, Cousin cover, common refinement, Darboux-gap lock, finite windows, terminal seal, transport, replay, provenance, and local naming.

The source rows are T, U, G, C. The tagged partition supplies finite cells and tags, the upper Darboux row supplies the upper-bound ledger, the Riemann sum gauge row supplies the gauge mesh, and the Cousin cover row supplies local gauge cover cells.

The refinement rows are M, D. The mesh row M binds those four source rows into one common finite refinement before the Darboux-gap lock D can be read.

Finally W, E, H, K, P, N preserve the same displayed route through finite windows, the terminal Real seal, transport, replay, provenance, and local naming. Since the carrier has no coordinate for the refused host objects, the listed rows exhaust the local NameCert surface. □

Theorem 5537.3 (Riemann refinement criterion Darboux gap lock)**.** *In every accepted* RiemannRefinementCriterior *packet, the Darboux-gap row D is readable only after the common-refinement mesh row M has already displayed the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover sources. Thus a Darboux or Riemann-integrability consumer cannot replace the refinement route by host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the source rows from Definition 5894.1. The Darboux-gap row is not listed as an independent analytic conclusion; it appears after the tagged partition, upper-sum, gauge, cover, and common-refinement rows.

Apply Definition 5894.2. Its obligation surface requires T, U, G, C to be exposed before M, and M before D.

The structural rows only transport and replay the same finite packet. Therefore the Darboux-gap read is locked to the displayed refinement route and cannot be promoted to any refused host-level principle. □

Theorem 5537.4 (Riemann refinement criterion integral handoff)**.** *Every Riemann-integrability or gauge-integral consumer of an accepted* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *packet receives only the finite handoff*

$$\mathsf{TaggedRealPartition}^{\uparrow},\ \textit{??},\ \mathsf{RiemannSumGauge}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow} \longmapsto M \longmapsto D \longmapsto W, E, H, K, P, N.$$

The handoff exports a refinement criterion over displayed finite rows, not an ambient theorem about all host partitions or all mesh limits.

Proof. Start with a consumer read. The packet first exposes the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover rows through the sibling chapters cited in the carrier.

The common-refinement mesh row M then aligns those sources. The Darboux-gap lock row D is read only along that aligned route, and the window and seal rows W, E expose the Real-facing observations after the finite rows are visible.

Transport, replay, provenance, and local naming through H, K, P, N preserve the same route. No carrier coordinate supplies host partition equality, selected limiting meshes, choice, compactness, quotient function equality, or ambient completeness, so the integral handoff is exactly the displayed finite criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet is generated from a closed bounded interval row, TaggedRealPartition source, UpperDarbouxSum ledger, RiemannSumGauge mesh row, CousinCover gauge-cover row, common-refinement mesh row, Darboux-gap lock, finite windows, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5894.1, Definition 5894.2, Definition 5894.3, and Definition 5894.4 seed the finite refinement criterion between tagged partitions, upper Darboux sums, Riemann sum gauges, and Cousin covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition source lock, upper-sum handoff, gauge mesh exactness, Cousin cover locality, Darboux-gap lock, integral handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRefinementCriterionUp.

Chapter 5538

A Concrete Naming Certificate for CompactUniformModulusComposition$^{\uparrow}$

Following the Bishop–Bridges and Heine–Cantor treatment of uniformly continuous maps on compact metric data, the CompactUniformModulusComposition$^{\uparrow}$ packet records the finite tolerance-only composition surface for two compact-uniform modulus rows. It sits after the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the compact uniform modulus row of chapter 2299, the metric-triangle estimate of chapter 1159, and the stability packet of chapter 5872. The packet supplies the finite composition object used by compact-open and compact-uniform-continuity consumers without exporting a host function space, quotient point equality, selected global modulus, choice schedule, or ambient compactness theorem.

Definition 5538.1 (Compact uniform modulus composition carrier)**.** *A* CompactUniformModulusComposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are compact metric source, middle, and target rows, with finite net and distance ledgers visible at the requested tolerances.*
2. *F and G are* ContinuousMap$^{\uparrow}$ *rows for the two displayed maps.*
3. *M_F is the first* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Y into a source tolerance for X along F.*
4. *M_G is the second* CompactUniformModulus$^{\uparrow}$ *row, turning a target tolerance for Z into a source tolerance for Y along G.*
5. *A is the tolerance-assembly row, recording that a Z-tolerance request is first read by M_G and that its resulting Y-tolerance request is then read by M_F.*
6. *T is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that keeps the two transported estimates on the same metric route.*
7. *U is the downstream* UniformModulus$^{\uparrow}$ *consumer row for the composite map.*
8. *H records componentwise* hsame *transport for compact metric sources, continuous maps, modulus rows, tolerance assembly, triangle estimate, consumer, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a composite tolerance request through $Z, M_G, Y, M_F, X, A, T, U, H, P, N$.*
10. *P is the* Pkg *provenance over* CompactUniformModulusComposition$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompactUniformModulusComposition}^{\uparrow}}$ *row.*

The classifier compares accepted composition packets only by these finite rows. It contains no host function-space composition theorem, quotient point equality, selected global modulus, choice schedule, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusCompositionUp.CompactUniformModulusCompositionTasteGate`
`single carrier alignment`

Theorem 5538.2 (Compact uniform modulus composition NameCert obligations)**.** *For every accepted* CompactUniformModulusComposition$^{\uparrow}$ *carrier $Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$, the local* NameCert$_{\text{CompactUniformModulusComposition}^{\uparrow}}$ *surface consists exactly of compact source admission through X, Y, Z, continuous-map admission through F, G, first and second compact-uniform modulus reads through M_F, M_G, tolerance assembly through A, metric-triangle transport through T, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5895.1. The compact metric rows, continuous-map rows, compact-uniform modulus rows, assembly row, triangle estimate, uniform-modulus consumer, and structural rows are exactly the displayed coordinates.

The tolerance route is ordered by A. A requested Z-tolerance is consumed by M_G to expose a Y-tolerance, and only that displayed Y-tolerance is then consumed by M_F to expose the source tolerance over X.

The triangle estimate T and consumer row U read the same assembled route. Transport, replay, provenance, and local naming preserve it through H, C, P, N.

Since the carrier has no coordinate for host function-space composition, quotient point equality, selected moduli, choice, or ambient compactness, the listed rows exhaust the NameCert surface. □

Theorem 5538.3 (Compact uniform modulus composition tolerance route)**.** *Every compact-open or compact-uniform-continuity consumer of an accepted* CompactUniformModulusComposition$^{\uparrow}$ *packet reads the composite modulus only through*

$$M_G \longmapsto A \longmapsto M_F \longmapsto T \longmapsto U.$$

Thus the composite tolerance is assembled from two displayed compact-uniform modulus rows and the metric-triangle ledger, not from a selected host-level modulus for the composite map.

Proof. Begin with the consumer's target tolerance for the Z row. The second modulus row M_G supplies the middle-space tolerance, and the assembly row A records that this is the only tolerance handed to M_F.

The first modulus row M_F returns the source-space tolerance over X. The metric-triangle row T then transports the two displayed estimates into the bound consumed by U.

The structural rows H, C, P, N preserve this route. Since there is no host composition coordinate or selected global modulus in the carrier, every consumer remains on the displayed tolerance chain. □

Theorem 5538.4 (Compact uniform modulus composition nonescape)**.** *No accepted* CompactUniformModulusComp *packet exports quotient point equality, a selected global modulus, a choice schedule, an ambient compactness theorem, or a host function-space composition proof. It exports only the two continuous-map rows, two compact-uniform modulus rows, the tolerance assembly row, the metric-triangle estimate, the uniform-modulus consumer, and the structural NameCert rows.*

Proof. Project the carrier. The only analytic rows are F, G, M_F, M_G, A, T, U over the compact metric sources X, Y, Z.

Apply Definition 5895.2. The local NameCert obligations require the consumer to read the two modulus rows and the tolerance assembly before the uniform-modulus export.

The refused host objects are absent from the carrier and cannot be produced by transport, replay, provenance, or local naming. Therefore the accepted packet exports only the displayed finite composition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ packet is generated from compact metric source, middle, and target rows, two ContinuousMap rows, two CompactUniformModulus rows, a tolerance-assembly row, a MetricTriangleUniformEstimate row, a UniformModulus consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5895.1, Definition 5895.2, Definition 5895.3, and Definition 5895.4 seed the finite tolerance-only composition route for compact-uniform modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space composition, quotient point equality, selected global moduli, choice schedules, ambient compactness theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, continuous-map admission, first and second modulus reads, tolerance assembly, metric-triangle transport, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulusCompositionUp.

Chapter 5539

A Concrete Naming Certificate for $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$

The $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet realizes the private/shared decoration theorem of Definition 9462.1705 inside the boundary-tile source of subsection 9462.28.21. It sits beside the DNA boundary-tile packet of chapter 3393, the trigger-threshold packet of chapter 4217, and the finite trigger algebra route of chapter 3478. The packet records private blocks, shared rows, lift counts, and defect ledgers as finite NameCert data; it is not a biological mechanism, stochastic evolution model, hidden enumeration oracle, or quotient codon semantics.

Definition 5539.1 (PrivateSharedDecorationModel carrier)**.** *A* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, L, D, T, R, P, N)$$

with the following displayed rows:

1. *B is the nonempty private-block row, recording the partition of coordinates that remain private to a single decorated support.*
2. *S is the shared-row ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *L is the lift-count row, recording the finite ℓ^m lift factor attached to displayed support blocks.*
4. *D is the defect ledger, recording the row-set size contribution to the defect-refined polynomial.*
5. *T records componentwise* hsame *transport for B, S, L, D, R, P, N.*
6. *R records displayed* Cont *replay from a decorated trigger request through the private block, shared rows, lift count, defect ledger, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem of Definition 9462.1705,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, S, L, D, T, R, P, N. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, or an unrecorded enumeration oracle.

Theorem 5539.2 (PrivateSharedDecorationModel NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier*

$$M = (B, S, L, D, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise transport, displayed replay, provenance, and local naming. A decorated trigger read is accepted only after it exposes a nonempty private block and assigns every shared coordinate to at least two rows.*

Proof. Project the carrier of Definition 5896.1. The private block, shared-row ledger, lift count, defect ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The private/shared constraint is read through B, S. The row B must display a nonempty private partition, and the ledger S must attach every shared coordinate to a row set of size at least two.

The enumeration and defect data are read through L, D. The lift-count row records the finite lift factor, while the defect ledger records the shared-row contribution used by the source theorem.

Finally T, R, P, N transport, replay, source, and name the same route. Since biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, and unrecorded enumeration have no carrier coordinate, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: If a displayed decorated trigger lacks a nonempty private block or assigns a shared coordinate to fewer than two rows, it is rejected by the local NameCert.

Independence witness: dna-window-six-boundary-tile, dna-window-six-trigger-threshold, dna-window-six-translation-boundary: $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries private-block rows, shared-row ledgers, lift-count rows, defect ledgers, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.PrivateSharedDecorationModelUp` must provide explicit $\mathsf{Nontrivial}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ instances preserving B, S, L, D, T, R, P, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet is generated from a nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5896.1 and Definition 5896.2 seed the finite private/shared decoration packet over DNA boundary-tile trigger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global semantics, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block nonemptiness, shared-row arity, lift-count exposure, defect-ledger exposure, transport, replay, provenance, local naming, Nontrivial PrivateSharedDecorationModelUp, and FieldFaithful PrivateSharedDecorationModelUp for BEDC.Derived.PrivateSharedDecorationModelUp.

Chapter 5540

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet records the finite filter-facing route by which located real brackets are read through dyadic tolerances, stream windows, regular rational readback, and the terminal real seal. It consumes the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located bracket surface of chapter 1709, and the selector boundary of chapter 5886. The packet gives the L10 Real and Completion route a located filter interface without asserting quotient filter equality, host real equality, selected limit points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5540.1 (Bishop located real filter carrier)**.** *A* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier is a finite packet*

$$S = (F, L, D, W, R, E, B, H, T, P, N)$$

with the following displayed rows:

1. *F is the filter-base row, exposing the finite base request, refinement evidence, and Cauchy-facing filter data that a later completion consumer may inspect.*
2. *L is the located-real bracket row, recording lower and upper bracket reads over* $\mathsf{LocatedRealInterval}^{\uparrow}$ *together with the displayed apartness and containment data used to locate the requested real value.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing bracket width, filter-base comparison, stream-window radius, and regular-rational readback.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the filter request and the located bracket.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W under the tolerance ledger D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after F, L, D, W, R have been displayed and replayed.*
7. *B is the* BHist *history row recording the finite route from the filter request through the located bracket, tolerance ledger, stream windows, regular readback, and Real seal.*
8. *H records componentwise* hsame *transport for $F, L, D, W, R, E, B, T, P, N$.*
9. *T records displayed* Cont *replay from every located filter request through $F, L, D, W, R, E, B, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, D, W, R, E, B, H, T, P, N$. It has no coordinate for quotient filter equality, host selected representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BishopLocatedRealFilterUp.BishopLocatedRealFilterUp`

Theorem 5540.2 (Bishop located real filter NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier $S = (F, L, D, W, R, E, B, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the filter base, located-real bracket, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, history, transport, replay, provenance, and local naming row;*
2. *bracket exactness, requiring each located-real bracket read in L to be witnessed by displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint and containment data;*
3. *filter-base stability, requiring refinements of F to preserve the same located bracket and tolerance ledger through* BHist *and* hsame *transport;*
4. *dyadic ledger stability, requiring D to control the bracket width, stream-window radius, and regular-rational readback without adding an external modulus;*
5. *StreamName and RegSeqRat handoff, requiring every numeric read to pass through W and then R before the terminal seal E can be consumed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, selected limiting points, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5897.1. The accepted packet displays the filter base, located bracket, dyadic ledger, stream windows, regular readback, Real seal, history, transport, replay, provenance, and local naming rows.

Bracket exactness is row-local. The bracket row L can expose only the endpoint and containment data already carried by $\mathsf{LocatedRealInterval}^{\uparrow}$, so it cannot identify a real by host order equality or by a selected representative.

The filter-base and tolerance rows then fix the route. Refinement of F is transported through B and H, while the same dyadic row D governs bracket width, window radius, and readback precision.

Finally the numeric handoff is ordered by W, R, E. Stream windows are read before regular rational data, and regular rational data are read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5540.3 (Bishop located real filter L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *packet $S = (F, L, D, W, R, E, B, H, T, P, N)$ must read the finite route*

$$F, L, D, W, R \longmapsto E \longmapsto B, H, T, P, N$$

before it treats the request as Real-facing or completion-facing data. Thus the packet supplies a filter-based located-real handoff to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, *and*

$\mathsf{CauchyFilterSelector}^{\uparrow}$ *consumers without asserting ambient real completeness or quotient equality of filters.*

Proof. Read the carrier rows in order. A consumer first receives the filter base F and the located bracket L, while the dyadic ledger D supplies the common tolerance budget.

The numerical content is then exposed through W and R. The $\mathsf{StreamName}^{\uparrow}$ row records finite observation windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational readback extracted from those windows.

Only after these rows are displayed can the Real seal E be consumed. The history, transport, replay, provenance, and local naming rows B, H, T, P, N preserve that order for later L10 Real and Completion routes.

The carrier has no quotient-filter equality, selected representative, host real equality, choice schedule, or ambient completeness coordinate. Therefore every downstream handoff factors through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet is generated from a displayed filter base, a located-real bracket, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a Real seal, BHist history, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5897.1, Definition 5897.2, and Definition 5897.3 seed the finite filter-facing located-real interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, ambient Real completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, bracket exactness, filter-base stability, dyadic ledger stability, StreamName and RegSeqRat handoff, and Real seal boundary for BEDC.Derived.BishopLocatedRealFilterUp.

Chapter 5541

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ packet records the finite diagonal extraction route that turns a displayed regular Cauchy family of rational names into the regular rational readback consumed by a $\mathsf{Real}^{\uparrow}$ seal. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ source surface of chapter 301, the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{Real}^{\uparrow}$ boundary of chapter 38, and the sibling diagonalization packet of chapter 3782. The packet names the L10 RegSeqRat-to-Real handoff without importing quotient streams, selected limits, choice, or ambient Real completeness.

Definition 5541.1 (Regular Cauchy diagonal argument carrier)**.** *A* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{RegSeqRat}^{\uparrow}$ *family row. It displays the finite collection of rational-name approximants and the precision requests inspected by the diagonal extraction.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite source windows selected for each inspected precision and exposes no quotient stream equality.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic radius, split budget, common scale, and comparison cells used to certify the diagonal readback.*
4. *R is the diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It consumes only the displayed windows W and tolerance ledger D as one regular rational sequence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after F, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the source family, stream windows, tolerance ledger, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a diagonal extraction request through the source family, finite stream windows, dyadic ledger, regular rational readback, Real seal, transports, provenance, and local name.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, D, R, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a checked bridge.*

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentTasteGate`
`single carrier alignment`

Theorem 5541.2 (Regular Cauchy diagonal argument NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier*

$$A = (F, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *family,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring W to expose only the finite source windows named by the diagonal extraction request;*
3. *dyadic tolerance exactness, requiring D to supply the radius, split budget, common scale, and comparison cells used by the extraction;*
4. *regular readback exactness, requiring R to consume precisely the windows W and dyadic ledger D before any Real-facing seal is visible;*
5. *Real-seal nonescape, requiring E to appear only after F, W, D, R have been transported by H and replayed by C;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming for exactly the rows $F, W, D, R, E, H, C, P, N$;*
7. *refusal of quotient streams, selected limits, choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking as local certificate data.*

No local diagonal-argument obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentNamecertObligations`

Proof. Project the carrier of Definition 5541.1. The accepted inventory is exactly the source $\mathsf{RegSeqRat}^{\uparrow}$ family F, finite $\mathsf{StreamName}^{\uparrow}$ windows W, dyadic tolerance ledger D, diagonal $\mathsf{RegSeqRat}^{\uparrow}$ readback R, terminal $\mathsf{Real}^{\uparrow}$ seal E, and structural rows H, C, P, N.

Window and tolerance exactness are read from W and D. The stream row exposes only finite inspected windows, and the dyadic row supplies the radius, split budget, common scale, and comparison cells needed to compare those windows.

The regular readback row R is downstream of those two rows. It cannot name a quotient stream or a selected limit, because the carrier gives it only W and D as inputs.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same packet. Since the carrier inventory contains no quotient streams, selected limits, choice operator, ambient completeness principle, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalArgumentUp.RegularCauchyDiagonalArgumentCarrier namecert obligations`

Theorem 5541.3 (Regular Cauchy diagonal argument Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ *carrier factors through the finite route*

$$F \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a diagonal extraction is available to the Real boundary only as a displayed $\mathsf{RegSeqRat}^{\uparrow}$ *family row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, one terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence.*

Proof. Begin with the obligation surface of Definition 5541.2. It confines every accepted packet to the displayed family, window, tolerance, readback, Real-seal, and structural rows.

The consumer first reads F, W, D. These rows expose the source regular rational names, the finite stream windows inspected by the diagonal argument, and the dyadic comparison budget.

The consumer then reads R and E. The regular rational readback is formed from the same finite windows and dyadic ledger, and the terminal Real seal is visible only after that readback has been displayed.

The rows H, C, P, N preserve this route under transport, replay, provenance, and naming. Because the carrier has no coordinate for quotient streams, selected limits, choice, ambient Real completeness, Lean verification, or bridge checking, none of those objects can enter the Real-facing handoff. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ Real or Completion read must expose a RegSeqRat family row, finite StreamName windows, one DyadicRatCore tolerance ledger, one diagonal RegSeqRat readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-diagonalization: $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds source RegSeqRat family data, finite StreamName windows, dyadic tolerance exactness, diagonal RegSeqRat readback, and RealUp handoff in one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}$ packet is generated from a RegSeqRat family row, finite StreamName windows, one DyadicRatCore tolerance ledger, one diagonal RegSeqRat readback, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5541.1, Definition 5541.2, and Definition 5541.3 seed the finite diagonal extraction packet used by L10 RegSeqRat-to-Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalArgument}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, regular readback exactness, Real-seal exposure, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyDiagonalArgumentUp.

Chapter 5542

A Concrete Naming Certificate for PrivateSharedCoordinateDecomposition$^{\uparrow}$

The PrivateSharedCoordinateDecomposition$^{\uparrow}$ packet realizes the private/shared coordinate decomposition of Definition 9462.1697 as finite NameCert data. It reads the quotient-level support family, private blocks, shared-coordinate row sets, and reconstruction equations through the support-census sibling of chapter 4048 and the reassignment-geometry sibling of chapter 5869. The packet is a finite combinatorial carrier; it is not a biological mechanism claim, a hidden quotient of support labels, or an ambient uniqueness principle outside the displayed rows.

Definition 5542.1 (Private-shared coordinate decomposition carrier)**.** *A* PrivateSharedCoordinateDecomposition$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, B, U, R, K, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite support-family row, displaying the quotient-level support family on the inspected coordinate set.*
2. *B is the private-block row, listing the nonempty private coordinate blocks attached to the supports.*
3. *U is the shared-coordinate row, listing the coordinates not owned by a single private block.*
4. *R is the row-set ledger. For every shared coordinate it records the finite set of private blocks whose supports contain that coordinate.*
5. *K is the reconstruction row, recording the displayed equations that recover each support from its private block and the shared coordinates whose row sets contain that block.*
6. *H records componentwise* hsame *transport for support-family, private-block, shared-coordinate, row-set, reconstruction, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from support-family inspection through private-block extraction, shared-coordinate listing, row-set lookup, support reconstruction, transports, provenance, and local name.*
8. *P is the* Pkg *provenance over* PrivateSharedCoordinateDecomposition$^{\uparrow}$*, the source private/shared coordinate vision,* SupportNerveCensus$^{\uparrow}$*,* ReassignmentMassSubcube$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, B, U, R, K, H, C, P, N$. It has no coordinate for hidden support equality, unbounded coordinate universes, biological causality, stochastic dynamics, Lean verification, or a checked bridge.

Theorem 5542.2 (PrivateSharedCoordinateDecomposition NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ *carrier*

$$Q = (S, B, U, R, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$ *surface consists exactly of support-family exposure, private-block exposure, shared-coordinate exposure, finite row-set exposure, reconstruction equation exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The reconstruction row must use only the displayed private block and shared-coordinate row-set data from B, U, R.*

Proof. Project the carrier of Definition 5542.1. The rows S, B, U, R display the support family, the nonempty private blocks, the shared coordinates, and the finite row sets attached to those shared coordinates.

The reconstruction row K is downstream of those data. It may rebuild a support only by taking its private block from B and adjoining exactly the shared coordinates from U whose row-set entry in R contains that block.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no hidden support equality, unbounded coordinate universe, biological mechanism, stochastic dynamics, Lean verification row, or bridge credential, the listed rows exhaust the local NameCert obligations. □

Theorem 5542.3 (Private-shared coordinate reconstruction handoff)**.** *Every consumer of an accepted* $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ *carrier factors through the finite route*

$$S \longmapsto B, U, R \longmapsto K \longmapsto H, C, P, N.$$

Consequently, the support corresponding to a private block is recovered only from the displayed private block, the shared-coordinate ledger, and the finite row-set membership data, exactly as in the source decomposition of Definition 9462.1698.

Proof. Start with the NameCert obligations of Definition 5542.2. They force the support-family row S to expose the finite family before private and shared coordinates are read.

The rows B, U, R then provide the decomposition data. Private blocks give the singleton-owned coordinates, while shared coordinates are paired with finite row sets naming the blocks whose supports contain them.

The reconstruction row K consumes only those displayed rows. Transport, replay, provenance, and local naming preserve the same finite route, so a consumer cannot replace it by a hidden quotient of supports or by an ambient uniqueness principle. □

Falsifiable prediction: Every accepted $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ consumer must expose finite support data, private blocks, shared coordinates, row-set membership, reconstruction equations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence before reading a reconstructed support.

Independence witness: support-nerve-census, reassignment-mass-subcube, private-shared-coordinate vision: $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds private blocks, shared-coordinate row sets, reconstruction equations, transport, replay, provenance, and local naming as one finite coordinate-decomposition packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}$ packet is generated from finite support-family rows, private-block rows, shared-coordinate rows, row-set ledgers, reconstruction equations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5542.1, Definition 5542.2, and Definition 5542.3 seed the finite NameCert packet for private/shared coordinate decomposition.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedCoordinateDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close hidden support equality, unbounded coordinate universes, biological causality, stochastic dynamics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked support-family exposure, private-block exposure, shared-coordinate exposure, row-set exposure, reconstruction exactness, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.PrivateSharedCoordinateDecompositionUp.

Chapter 5543

A Concrete Naming Certificate for $\mathsf{RegulatedCauchyModulus}^{\uparrow}$

Following Bishop and Bridges on constructive moduli of Cauchy convergence, the $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ packet records an explicit finite threshold schedule for a regular rational Cauchy name before the terminal real-name seal is consumed. It sits after the regular rational source of chapter 301, the finite window surface of chapter 126, the dyadic tolerance core of chapter 291, the Cauchy modulus rows of chapter 292, and the terminal real-name boundary of chapter 38. The packet gives the L10 Real and Completion route a non-quotient explicit-modulus carrier; it does not select a limit, quotient streams, invoke countable choice, assert host equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5543.1 (Regulated Cauchy modulus carrier)**.** *A* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, Q, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose finite rational reads are being regulated.*
2. *Q is the explicit modulus threshold row, recording the displayed tolerance request and the finite index bound at which the Cauchy comparison is certified.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to expose the source reads needed by Q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the requested radius, the comparison budget, and the readback precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, Q, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the source, threshold, window, tolerance, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through R, Q, W, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegulatedCauchyModulusUp.TasteGate.RegulatedCauchyModulusTasteGate single carrier alignment`

Theorem 5543.2 (Regulated Cauchy modulus NameCert obligations)**.** *For every accepted* RegulatedCauchyModulu *carrier* $M = (R, Q, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, source-window exposure, explicit threshold exactness, dyadic tolerance control, real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read through the finite route*

$$R, Q, W, D \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Carrier admission is projection from Definition 5543.1. The packet first exposes the regular rational source R, the explicit threshold row Q, the stream windows W, and the dyadic tolerance ledger D.

The threshold obligation is local to Q. It records a displayed tolerance request and a finite index bound, while W supplies the finite source reads and D supplies the dyadic comparison budget used to check the requested tail comparison.

The terminal seal row E is downstream of those four rows. Componentwise transport through H and replay through C preserve the same order, and P, N record package provenance and the local NameCert comparison.

Since quotient streams, selected limits, host real equality, countable choice, trichotomy, and ambient completeness do not occur among the carrier coordinates, no NameCert obligation can be discharged outside the displayed finite route. □

Theorem 5543.3 (Regulated Cauchy modulus real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *packet must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the explicit modulus threshold, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. The handoff to the seal row* E *therefore supplies a finite explicit-modulus route from regular rational data to Real-facing data, not a host limit theorem or a quotient construction.*

Proof. Project an accepted carrier $M = (R, Q, W, D, E, H, C, P, N)$. The source row R is the only regular rational input, and the threshold row Q is the only displayed modulus schedule.

The stream windows W expose the finite observations required by that schedule. The dyadic ledger D supplies the tolerance comparison that makes the schedule a regulated Cauchy read rather than an unbounded stream claim.

Only after R, Q, W, D are present may the consumer read E as the terminal $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming keep the same route visible to downstream consumers.

The carrier has no coordinate for a selected limit, quotient stream equality, host real equality, countable choice, trichotomy, or ambient Real completeness. Thus every Real-facing handoff factors through the finite explicit-modulus rows. □

Theorem 5543.4 (Regulated Cauchy modulus tail-threshold determinacy)**.** *Let* $M = (R, Q, W, D, E, H, C, P, N)$ *and* $M' = (R', Q', W', D', E', H', C', P', N')$ *be accepted* $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ *reads whose displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and threshold row agree. Then the two reads expose the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route. Strengthening a request may enlarge the finite window demanded by* W *and* W'*, but it cannot change the displayed source row, tolerance ledger, threshold row, or terminal seal route.*

Proof. Project both accepted carriers. The only rows available before the terminal seal are the source rows R, R', threshold rows Q, Q', finite-window rows W, W', and dyadic tolerance ledgers D, D'.

The hypothesis identifies the displayed source, threshold, window, and tolerance rows. By Definition 5543.3, every terminal $\mathsf{Real}^{\uparrow}$ handoff must pass through exactly those finite rows before E or E' can be read.

If a request is strengthened, the continuation may demand a larger finite window, because the stream row is a finite observation schedule. The displayed source, tolerance, threshold, and seal coordinates are still the same coordinates of the carrier, so strengthening cannot replace them by a quotient stream, selected limit, or ambient completeness argument.

Therefore both accepted reads expose the same terminal Real-facing seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedCauchyModulus}^{\uparrow}$ packet is generated from a RegSeqRat source row, explicit modulus threshold row, finite StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5543.1, Definition 5543.2, and Definition 5543.3 seed the explicit-modulus route from regular rational source data to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegulatedCauchyModulusUp`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, source-window exposure, explicit threshold exactness, dyadic tolerance control, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedCauchyMod

$\mathsf{RegularCauchyMinModulus}^{\uparrow}$ routes its hub through the shared modulus meet of chapter 462, the finite dyadic tolerance rows of chapter 291, and the Real-facing handoff through chapter 301 and chapter 38.

Chapter 5544

A Concrete Naming Certificate for $\mathsf{RegularCauchyMinModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ packet records the finite scheduling surface for taking the pointwise minimum of two displayed regular-Cauchy modulus schedules. It sits between the shared-modulus meet packet of chapter 462, the finite stream windows of chapter 126, the regular rational readback boundary of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The downstream selector of chapter 7949 reads the minimum row only after the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ surface and the $\mathsf{DyadicRatCore}^{\uparrow}$ comparison ledger have displayed the common tolerance budget. The packet is a finite L10 Real-and-Completion route: it names the displayed minimum-modulus schedule, not a selected quotient stream, an ambient completeness theorem, or a choice principle over all precision requests.

The minimum-modulus read also depends on the finite-window induction budget of chapter 8091 and the locatedness tolerance budget of chapter 1627. A Real-facing consumer must therefore see the $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ schedule and the $\mathsf{LocatednessModulus}^{\uparrow}$ request row before treating m as a stable minimum for $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ handoffs.

Definition 5544.1 (Regular Cauchy minimum-modulus carrier)**.** *A* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S_0 and S_1 are the two regular-Cauchy schedules whose finite precision requests are to be synchronized.*
2. *M is the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row that displays the common meet surface inherited from the two source schedules.*
3. *m is the finite minimum row, recording the requestwise smaller admitted threshold after the two source schedules and the meet row have been read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming exactly which source windows are inspected by the minimum schedule.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and the same displayed thresholds.*

6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger storing denominator budgets, dyadic radii, slack comparisons, and the finite inequalities used by M and m.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S_0, S_1, M, m, W, Q, D have been displayed.*
8. *H stores componentwise* hsame *transport for the two schedules, the meet row, the minimum row, the stream windows, the regular-rational readback, the dyadic ledger, the real seal, the provenance row, and the local name.*
9. *C stores displayed* Cont *replay from a finite precision request through $S_0, S_1, M, m, W, Q, D, E, H$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5544.2 (Regular Cauchy minimum-modulus NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two displayed regular-Cauchy schedules, the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row, the finite minimum row, stream windows, regular rational readback, dyadic tolerance ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *meet compatibility, requiring M to be read through the sibling shared-modulus surface of chapter 462 before the minimum row m is consumed;*
3. *finite minimum exactness, requiring each request in m to name the smaller displayed threshold among the two transported source schedules after the dyadic comparisons in D have been read;*
4. *readback stability, requiring the same* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *row Q to be replayed under componentwise* hsame *transport;*
5. *ledger rows for* Cont *replay,* Pkg *provenance, the* $\mathsf{Real}^{\uparrow}$ *seal handoff, and the local naming certificate.*

No accepted obligation row is stored outside $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$.

Proof. Carrier admission is projection from Definition 5544.1. The accepted packet displays the two schedules, the meet row, the minimum row, finite windows, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local name.

The meet compatibility row is read before the minimum row because m is meaningful only after the common surface M has synchronized the two source schedules. The sibling packet chapter 462 supplies exactly this shared-modulus boundary.

The finite minimum exactness row is then a dyadic comparison inside D. It chooses no infinite stream representative and invokes no ambient completeness theorem; it records only the displayed smaller threshold for each finite request.

Finally W, Q, E, H, C, P, N replay the same route for stream, regular-rational, and Real-facing consumers. Since none of the carrier coordinates stores quotient stream equality, selected limits,

host real equality, countable choice, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 5544.3 (Regular Cauchy minimum-modulus L10 route)**.** *Let $R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier. Every L10 Real-and-Completion consumer of the minimum schedule factors through the finite route*

$$S_0, S_1 \longmapsto M \longmapsto m, D \longmapsto W, Q \longmapsto E \longmapsto H, C, P, N.$$

The route first reads the two regular-Cauchy schedules, then the shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ *surface, then the finite minimum row with its* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, then the* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It exports no quotient stream equality, selected limit, hidden completion object, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the accepted carrier and the obligation theorem above. They force every consumer request to retain $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ as one finite packet.

The shared-modulus step reads M before any minimum schedule is used. Thus the consumer is not comparing unrelated streams; it is reading the two source schedules through the same displayed meet surface and the same transported tolerance boundary.

The minimum step reads m together with D. The row D carries the dyadic comparisons and tolerance budgets, while m records only the finite threshold selected by those displayed comparisons.

The stream and regular-rational handoff is then mediated by W and Q, and the terminal seal by E. The structural rows H, C, P, N transport, replay, source, and name the same route. Hence every L10 consumer factors through the displayed sequence, with no quotient stream, selected limit, hidden completion object, countable choice, or ambient completeness coordinate available. □

Theorem 5544.4 (Regular Cauchy minimum-modulus positive gap)**.** *Let $R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ be an accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *carrier. Whenever the two source modulus schedules display positive dyadic requests through* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *and* $\mathsf{DyadicRatCore}^{\uparrow}$*, the minimum row m preserves a positive displayed request for the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *handoff. The route also names the* $\mathsf{LocatednessModulus}^{\uparrow}$ *request row and the* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *finite-window budget before any Real-facing consumer reads E.*

Proof. Project the carrier of Definition 5544.1. The two source schedules are read as S_0, S_1, their common meet is M, and the dyadic comparison ledger is D.

Apply the NameCert obligation theorem, Definition 5544.2. Its meet-compatibility and finite-minimum exactness rows require the row m to be read only after M and the displayed dyadic comparisons in D have synchronized the two source requests.

Apply the L10 route theorem, Definition 5544.3. The positive request stored in m is then handed to W, Q, E through the finite sequence $S_0, S_1, M, m, D, W, Q, E, H, C, P, N$, with the sibling $\mathsf{LocatednessModulus}^{\uparrow}$ row and $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ schedule providing the locatedness and finite-window context.

Because the carrier contains no quotient stream, selected limit, hidden completion object, countable-choice selector, or ambient completeness coordinate, the preserved positive gap is only the displayed dyadic request exported by the finite packet. □

Theorem 5544.5 (Regular Cauchy minimum-modulus finite-window induction)**.** *Let*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

be an accepted RegularCauchyMinModulus$^{\uparrow}$ *carrier. Extending both source schedules by one displayed finite* StreamName$^{\uparrow}$ *and* RegSeqRat$^{\uparrow}$ *window preserves the same minimum-modulus row exactly when the* CauchyModulusMeet$^{\uparrow}$ *row and the* DyadicRatCore$^{\uparrow}$ *tolerance ledger replay through* hsame *and* Cont*. The base read is the carrier route for* S_0, S_1, M, m, W, Q, D*, and the step read uses the meet-compatibility and finite-minimum exactness rows of Definition 5544.2.*

Proof. Proceed by induction on the displayed finite window extension. In the base case, no new window has been appended, so Definition 5544.1 supplies the existing rows $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ and the minimum row is the admitted row of the carrier.

For the induction step, append one source window to each schedule and read it through the displayed StreamName$^{\uparrow}$ and RegSeqRat$^{\uparrow}$ rows W, Q. The meet-compatibility row of Definition 5544.2 forces the appended requests to pass through the same CauchyModulusMeet$^{\uparrow}$ row M before m is consumed.

The finite-minimum exactness row then uses the DyadicRatCore$^{\uparrow}$ ledger D to compare the transported thresholds. Transport through H and replay through C preserve the same displayed comparison data, so the appended window keeps the minimum-modulus row precisely when M and D replay through hsame and Cont.

Applying the induction hypothesis to the earlier windows and the step argument to the appended window yields preservation for the whole finite extension, with no quotient stream equality, countable choice, selected limit, or ambient completeness coordinate. □

Theorem 5544.6 (Regular Cauchy minimum-modulus scoped primitive route)**.** *Let*

$$R = (S_0, S_1, M, m, W, Q, D, E, H, C, P, N)$$

be an accepted RegularCauchyMinModulus$^{\uparrow}$ *carrier. Every minimum-modulus consumer factors through* $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ *together with the* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance rows,* RegularCauchyWindowInduction$^{\uparrow}$ *finite-window schedule,* LocatednessModulus$^{\uparrow}$ *request row, terminal* Real$^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The route exports no quotient stream equality, selected limit, hidden completion object, countable-choice selector, host real equality, or ambient* Real$^{\uparrow}$ *completeness principle.*

Proof. Begin with the obligation theorem, Definition 5544.2. It fixes carrier admission, meet compatibility, finite minimum exactness, readback stability, replay, provenance, and local naming as rows of the single finite packet.

Apply the L10 route theorem, Definition 5544.3. Every Real-and-Completion consumer reads the two source schedules, the meet row, the minimum row with dyadic tolerance data, the stream and regular-rational handoff, the real seal, and the structural rows in that order.

Use the positive-gap theorem, Definition 5544.4, to bind the locatedness request row and the finite-window budget to the displayed dyadic request. Then apply the finite-window induction theorem, Definition 5544.5, so extensions of the inspected windows remain within the same M, D, H, C replay surface.

Thus every consumer route is scoped by the named BEDC primitives and sibling packets, and the excluded completion, choice, quotient, and host-equality claims remain outside the accepted RegularCauchyMinModulus$^{\uparrow}$ carrier. □

Falsifiable prediction: An accepted RegularCauchyMinModulus$^{\uparrow}$ consumer that reaches a RegSeqRat$^{\uparrow}$ readback or Real$^{\uparrow}$ seal without reading $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ refutes the carrier surface.

Independence witness: cauchymodulusmeet, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the joint packet containing two regular-Cauchy schedules, a shared meet row, a finite minimum row, stream windows, regular rational readback, dyadic tolerance ledger, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ packet is generated from two regular-Cauchy schedules, a CauchyModulusMeet row, a finite minimum row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5544.1, Definition 5544.2, Definition 5544.3, Definition 5544.4, Definition 5544.5, and Definition 5544.6 seed the finite minimum-modulus route from two regular-Cauchy schedules through a shared meet surface to stream, regular-rational, and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMinModulus}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, quotient stream equality, selected limits, hidden completion objects, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, meet compatibility, finite minimum exactness, readback stability, and the L10 Real-and-Completion route for BEDC.Derived.RegularCauchyMinModulusUp.

Theorem 5544.7 (Regular Cauchy minimum-modulus CauchyModulus dependency)**.** *Every* RegularCauchyMinMo *consumer reads the request-normalization rows of chapter 292 before it may normalize a source tail request into the minimum-modulus row. The same route reads the regular-sequence handoff of chapter 301 before any* $\mathsf{Real}^{\uparrow}$ *completion-facing seal is consumed.*

Thus the minimum-modulus row depends on $\mathsf{CauchyModulus}^{\uparrow}$ *for source request normalization and on* $\mathsf{RegSeqRat}^{\uparrow}$ *for regular readback, while it supplies no quotient stream equality, selected limit, hidden completion object, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the minimum-modulus request. A consumer cannot form the minimum row until the modulus comparison schedule has displayed how source tail requests are normalized, so the $\mathsf{CauchyModulus}^{\uparrow}$ request-normalization rows are read first.

Next read the regular-sequence handoff. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the regular readback used by the completion-facing route, and this readback occurs only after the modulus schedule has already fixed the request shape.

Finally inspect the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is downstream of the displayed request schedule and regular readback, it cannot introduce quotient stream equality, a selected limit, a hidden completion object, countable choice, or ambient completeness. □

Theorem 5544.8 (Regular Cauchy minimum-modulus downstream sibling route)**.** *Any* CauchyTailThresholdNorma *consumer reading a normalized shared threshold through chapter 1299 must preserve the* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *row as the finite lower-bound and modulus witness before any* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal read. In particular, the downstream route factors through the same source schedules, meet row, minimum row, finite windows, regular-rational readback, dyadic tolerance ledger, transport, replay, provenance, and local naming rows exposed by Definition 5544.9, Definition 5544.10, and Definition 5544.11.*

Proof. Project the carrier of Definition 5544.1. The displayed packet carries $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$, so the minimum row m is read together with the meet, window, regular-rational, dyadic, seal, transport, replay, provenance, and naming rows.

Apply the forward and hub routes. They force downstream consumers to keep the minimum-modulus row downstream of the $\mathsf{CauchyModulusMeet}^{\uparrow}$ row and upstream of the Real-facing readback surface, rather than reading m as an isolated scalar.

Now read the threshold normalizer. Its normalized shared threshold is a Real-facing consumer route, so it must expose the RegularCauchyMinModulus row before passing to the $\mathsf{RealTailAgreementSeal}^{\uparrow}$ handoff or the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the Real dependency route. It already excludes quotient stream equality, selected limits, countable-choice schedules, and ambient Real completeness, so the downstream sibling relation is exactly the finite lower-bound and modulus witness carried by the accepted packet. □

Theorem 5544.9 (Regular Cauchy minimum-modulus forward sibling route)**.** *Every downstream read of* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *by the selector of chapter 7949 must first expose the* $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows already fixed by this chapter. In carrier notation, the selector may consume the minimum row only after the route*

$$S_0, S_1, M, m, W, Q, D, E, H, C, P, N$$

has displayed M as the meet surface, D as the dyadic tolerance ledger, W as the finite stream-window service, Q as the regular-rational readback row, and E as the Real-facing seal.

Proof. Start with the carrier of Definition 5544.1. It displays $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$ as one finite packet, so the downstream selector cannot read m apart from the meet, window, readback, dyadic, seal, transport, replay, provenance, and naming rows.

Apply the NameCert obligations of Definition 5544.2. They force meet compatibility through M, finite minimum exactness through m and D, and readback stability through W and Q before any consumer treats the minimum schedule as accepted data.

Apply the L10 route of Definition 5544.3 and the positive-gap theorem of Definition 5544.4. These results place the displayed minimum before the $\mathsf{Real}^{\uparrow}$ seal while keeping the $\mathsf{LocatednessModulus}^{\uparrow}$ and $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ context visible.

Hence a downstream Cauchy-subsequence selector receives only the already named finite route. It receives no quotient stream equality, selected limit, hidden completion object, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle. □

Theorem 5544.10 (Regular Cauchy minimum-modulus hub dependency route)**.** *Every accepted* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *packet consumes the* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *meet row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal before exporting the minimum-modulus row to any downstream consumer. In carrier notation, the route exposes*

$$S_0, S_1, M, m, W, Q, D, E, H, C, P, N$$

with M read as the meet surface, D read as the dyadic comparison ledger, W read as the finite window service, Q read as the regular-rational readback, and E read as the Real-facing seal.

Proof. Project the carrier of Definition 5544.1. The minimum-modulus row m is not an isolated output; it is displayed inside the finite packet $S_0, S_1, M, m, W, Q, D, E, H, C, P, N$.

Read the sibling route already present in the chapter. The $\mathsf{CauchyModulusMeet}^{\uparrow}$ row M synchronizes the two source schedules before m can be consumed, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the comparison ledger that makes the displayed minimum meaningful.

Now read the Real-and-Completion handoff. The $\mathsf{StreamName}^{\uparrow}$ windows W and $\mathsf{RegSeqRat}^{\uparrow}$ row Q provide the finite readback surface, while E seals the same route for $\mathsf{Real}^{\uparrow}$ consumers. Transport, replay, provenance, and local naming remain attached through H, C, P, N, so the hub dependency route exports only the displayed minimum-modulus packet and no private completion or choice coordinate. □

The Real-facing route for $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ is a finite dependency route through the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, the $\mathsf{StreamName}^{\uparrow}$ finite windows of chapter 126, and the terminal $\mathsf{Real}^{\uparrow}$ sealing surface of chapter 38. A consumer may read the minimum-modulus row only after those displayed rows are present in the same BHist, hsame, Cont, Pkg, and local NameCert route; it receives no quotient stream equality, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Theorem 5544.11 (Regular Cauchy minimum-modulus sibling route)**.** *Every* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *read factors through the* $\mathsf{CauchyModulus}^{\uparrow}$ *carrier of chapter 292 and the* $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ *normalizer of chapter 778 before exposing the minimum modulus row. The route first displays the source modulus request, then normalizes the regular-Cauchy threshold schedule, and only then reads the finite minimum row together with its* BHist*,* hsame*,* Cont*,* Pkg*, and local* NameCert *support.*

Proof. Start with the carrier of Definition 5544.1. The accepted packet displays source schedules, the meet row, the minimum row, stream windows, regular-rational readback, dyadic tolerance data, the Real seal, transport, replay, provenance, and local naming.

Read the $\mathsf{CauchyModulus}^{\uparrow}$ sibling first. It supplies the carrier-level modulus request whose finite thresholds are visible before the minimum row can be inspected.

Next read the $\mathsf{RegularCauchyModulusNormalizer}^{\uparrow}$ sibling. Its normalizer keeps the regular-Cauchy threshold schedule on the same displayed carrier route, so the minimum row is not a detached host-level comparison.

The exposed minimum modulus row is therefore downstream of both sibling packets. Since the route remains inside the displayed BHist, hsame, Cont, Pkg, and local NameCert rows, it exports no quotient completion, selected limit, countable-choice selector, or ambient Real completeness principle. □

Chapter 5545

A Concrete Naming Certificate for $\mathsf{TaylorTheorem}^{\uparrow}$

Following Taylor and the validated-numerics treatment of finite Taylor models, the $\mathsf{TaylorTheorem}^{\uparrow}$ packet records the finite theorem-facing route from displayed derivative data to a polynomial jet and a checked remainder envelope. It sits after the Taylor-model enclosure surface of chapter 442, the derivative observation row of chapter 122, the polynomial carrier of chapter 50, the dyadic rational arithmetic core of chapter 291, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Taylor theorem certificate for one displayed center, order, domain window, derivative ledger, polynomial jet, and remainder bound; it does not import analytic convergence, quotient real equality, selected limits, or a host Taylor series.

Definition 5545.1 (Taylor theorem carrier)**.** *A* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, J, P, R, E, Q, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row naming the center, finite order, domain window, and derivative observations inspected by the packet.*
2. *J is the finite jet ledger. Each coefficient is read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic and the same derivative window named by D.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *row assembled from the displayed finite jet ledger.*
4. *R is the finite remainder-envelope row. It records the derivative-order budget, dyadic interval or ball enclosure, and endpoint window used to bound the residual after P is read.*
5. *E is the* $\mathsf{TaylorModel}^{\uparrow}$ *enclosure row that packages the displayed polynomial part and remainder envelope as a validated finite jet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for polynomial values, derivative observations, and the remainder bound.*
7. *H records componentwise* hsame *transport for derivative, jet, polynomial, remainder, Taylor-model, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every Taylor-theorem read through $D, J, P, R, E, Q, H, G, N$.*

9. *G is the* Pkg *provenance over* $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, P, R, E, Q, H, C, G, N$. It has no coordinate for an infinite Taylor series, analytic convergence, quotient equality of real names, selected limiting values, or an ambient smooth-function theorem.

Theorem 5545.2 (Taylor theorem NameCert obligations)**.** *For every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier*

$$T = (D, J, P, R, E, Q, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed derivative row, finite jet ledger, polynomial row, remainder envelope, Taylor-model enclosure, readback route, transport, replay, provenance, and local naming rows;*
2. *derivative-to-jet routing, requiring every coefficient in J to be read from the finite* $\mathsf{Derivative}^{\uparrow}$ *observations in D and through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic;*
3. *polynomial assembly, requiring P to use exactly the displayed finite coefficient rows in J;*
4. *remainder-envelope exposure, requiring R to bind the next derivative-order budget, the domain window, and the dyadic enclosure before any error bound is exported;*
5. *Taylor-model handoff, requiring E to package only the displayed P and R rows;*
6. *regular and real readback, requiring Q to pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
7. *non-escape from analytic convergence, quotient real equality, selected limits, and host Taylor series semantics.*

No $\mathsf{TaylorTheorem}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.TaylorTheoremUp.TasteGate.TaylorTheoremTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5898.1. The theorem-facing data are exactly D, J, P, R, E, Q, and the structural rows are exactly H, C, G, N.

The derivative-to-jet route is finite. Each coefficient read is a displayed entry of J, and the definition of J attaches that entry to the derivative row D and to dyadic rational arithmetic through $\mathsf{DyadicRatCore}^{\uparrow}$.

Polynomial assembly and remainder exposure are then separate rows. The polynomial row P consumes only the coefficients in J, while the remainder row R consumes the derivative-order budget and domain window before any bound is exported.

Finally the Taylor-model and readback rows seal the route. The row E packages the polynomial and remainder as a finite Taylor-model enclosure, and Q sends the same displayed data through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Since no carrier coordinate names a host Taylor series or a selected limit, the listed rows exhaust the local NameCert surface. □

Theorem 5545.3 (Taylor theorem finite remainder envelope)**.** *Every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier exports a remainder claim only through the displayed route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{TaylorModel}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the derivative row D, finite jet ledger J, polynomial row P, remainder-envelope row R, Taylor-model enclosure E, and regular-rational and real readback row Q before it may cite an error bound for the displayed polynomial approximation. The exported envelope is not an analytic convergence theorem, a quotient equality of real functions, a selected limit, or an ambient Taylor-series object.

Proof. Start with an accepted carrier $T = (D, J, P, R, E, Q, H, C, G, N)$. The only source of derivative information is D, and the only coefficient ledger is the finite row J read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next assemble the approximation. The polynomial row P is built from J, and the remainder row R binds the derivative-order budget, domain window, and dyadic enclosure that measure the residual after that polynomial row has been fixed.

Then pass through the enclosure and readback rows. The Taylor-model row E packages the polynomial and remainder as one finite validated jet, while Q routes the resulting value and bound through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and local naming rows preserve exactly this route. Because the carrier contains no coordinate for analytic convergence, quotient function equality, selected limits, or host Taylor series semantics, every exported remainder envelope is exhausted by the displayed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{TaylorTheorem}^{\uparrow}$ remainder-envelope read factors through a DerivativeUp row, a DyadicRatCore coefficient ledger, a PolynomialUp row, a TaylorModelUp enclosure, a RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: taylormodel, derivative, polynomial, dyadicratcore, regseqrat, real: $\mathsf{TaylorTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite derivative observations, coefficient arithmetic, polynomial assembly, a theorem-facing remainder envelope, Taylor-model enclosure, regular-rational readback, real sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TaylorTheorem}^{\uparrow}$ packet is generated from a DerivativeUp row, a finite DyadicRatCore coefficient ledger, a PolynomialUp row, a finite remainder envelope, a TaylorModelUp enclosure, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 5898.1, Definition 5898.2, and Definition 5898.3 seed the finite Taylor theorem route from derivative observations through polynomial jets to displayed remainder envelopes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic convergence, quotient equality of real names, selected limits, host Taylor series semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-jet routing, polynomial assembly, finite remainder-envelope exposure, Taylor-model handoff, regular and real readback, and non-escape for BEDC.Derived.TaylorTheoremUp.

The $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ hub routes finite bisection reads through chapter 394 and through the Real, Dyadic, and RegSeqRat surfaces carried by the child chapter.

Chapter 5546

Located Real Interval Bisection NameCert Construction

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ packet records finite dyadic interval bisection over displayed $\mathsf{Real}^{\uparrow}$ endpoints without importing an interval-completeness theorem. It sits beside the real-name seal of chapter 38, the dyadic arithmetic core of chapter 291, the regular rational readback route of chapter 301, and the closed bounded interval carrier of chapter 1591. The packet is a finite bisection ledger for located interval reads; it is not a Dedekind completion, quotient-real equality, trichotomy on arbitrary reals, a selected infinite branch, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5546.1 (Located real interval bisection carrier)**.** *A* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, L, M, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, endpoint order, and interval-containment data.*
2. *L is the locatedness row, recording the finite apartness or locator evidence used to decide which inspected subinterval remains visible at the requested stage.*
3. *M is the dyadic midpoint row, exposing the midpoint construction for each retained interval cell.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter and tolerance ledger attached to the same midpoint rows.*
5. *W is the finite endpoint-window row, recording which endpoint and midpoint windows are inspected by the bisection request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the source interval, locatedness, midpoint, dyadic, window, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for the interval, locatedness, midpoint, dyadic, window, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through I, L, M, D, W, R, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, L, M, D, W, R, E, H, C, P, N$. It has no coordinate for a completed nested-interval limit, quotient equality of real names, excluded-middle order trichotomy, a selected infinite branch, countable choice, or ambient real completeness.

Theorem 5546.2 (Located real interval bisection obligation surface)**.** *For every accepted* LocatedRealIntervalBisecti *carrier*

$$B = (I, L, M, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the closed bounded interval source, locatedness row, midpoint row, dyadic tolerance ledger, endpoint windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming;*
2. *source visibility, requiring every bisection read to expose the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row I before any midpoint, diameter, branch, or seal row is consumed;*
3. *located branch discipline, requiring L to record the finite locator evidence for the retained cell without adding global trichotomy on* $\mathsf{Real}^{\uparrow}$*;*
4. *dyadic midpoint exactness, requiring M and D to supply the displayed midpoint and diameter budgets at the requested finite depth;*
5. *readback handoff, requiring W, R, E to route every Real-facing output through finite endpoint windows and a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
6. *non-escape, excluding completed nested-interval limits, quotient-real equality, selected infinite branches, countable choice, host endpoint equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the carrier.*

No LocatedRealIntervalBisection obligation is stored outside these rows.

Proof. Project the carrier of Definition 5546.1. The accepted packet displays exactly the closed bounded interval source, locator row, midpoint row, dyadic ledger, endpoint windows, regular readback, terminal real seal, and structural rows.

The source stage is I, L. A request first reads the displayed endpoints and interval containment through I, then reads the finite locator evidence in L before any retained child interval is available.

The bisection stage is M, D, W. Induction on the finite retained-depth ledger reads one midpoint row at a time. At each step the dyadic tolerance entries in D bound the retained child, while W records the inspected endpoint and midpoint windows.

The Real-facing handoff is R, E, H, C, P, N. The regular rational readback R is computed from the displayed windows and dyadic budgets, and the terminal seal E is downstream from that readback. The structural rows only transport, replay, source, and name the same finite route.

Since the carrier contains no completed nested-interval limit, quotient-real equality, global trichotomy, selected infinite branch, countable-choice row, or ambient completeness coordinate, the listed rows exhaust the local obligation surface. □

Theorem 5546.3 (Located real interval bisection finite handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto L, M, D, W \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The consumer receives one finite located interval bisection ledger with dyadic midpoint and diameter rows. It receives no completed intersection point, quotient representative, selected branch, countable-choice witness, or ambient completeness principle.

Proof. Use the carrier projection and the obligation surface of Definition 5546.2. Source visibility places every read below the row I before midpoint or readback data are available.

Located branch discipline then exposes only the finite locator row L. Dyadic midpoint exactness exposes M, D, W for the requested depth, so the retained cell is read as a finite ledger rather than as an infinite branch.

Finally R repacks the displayed windows as regular rational readback data, and E, H, C, P, N seal and name the same route for $\mathsf{Real}^{\uparrow}$ consumers. The absent coordinates listed in the carrier definition cannot be projected from the packet. □

Falsifiable prediction: Every accepted $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ Real-facing read passes through $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, finite endpoint windows, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any retained interval output is visible.

Independence witness: real, dyadicratcore, regseqrat, closedboundedinterval, located-interval-bisection, bisection-real: no listed sibling supplies the joint packet containing a closed bounded Real interval source, locator evidence, dyadic midpoint ledger, finite endpoint windows, regular rational readback, Real seal, transport, replay, provenance, and local NameCert row while refusing completeness and selected infinite branches.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5546.1, Definition 5546.2, and Definition 5546.3 seed the finite located bisection route over Real endpoints, closed bounded interval source rows, dyadic midpoint budgets, regular rational readback, and Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealIntervalBisection}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.LocatedRealIntervalBisectionUp`

Bridge status: none

Not claimed: This chapter does not close completed nested-interval limits, quotient-real equality, global trichotomy on reals, selected infinite branches, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source visibility, located branch discipline, dyadic midpoint exactness, readback handoff, and non-escape for BEDC.Derived.LocatedRealIntervalBisectionUp.

Constructive story: A $\mathsf{LocatedRealIntervalBisection}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, locatedness evidence, dyadic midpoint and diameter rows, finite endpoint windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The handoff hub routes candidate-normalization and confluence reads through chapter 4238 and chapter 765 before exposing bounded endpoint or residual-envelope rows.

Chapter 5547

MetaCIC Candidate Normalization Confluence Handoff NameCert Construction

The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet realizes the candidate-mediated strong-normalization boundary of chapter 9383 as a finite handoff surface for confluence and decidability consumers. It sits beside the critical-path carrier of chapter 765, the normalization frontier of chapter 892, the residual confluence envelope of chapter 3921, and the decidable boundary of chapter 749. The packet names a guarded route from candidate evidence to bounded normal endpoints and residual-envelope reads; it does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, or global closed-term conversion decidability.

Candidate-mediated normalization reads the residual local-join budget of chapter 5970 before it can be used as a confluence handoff, so the endpoint row is never exported as an isolated normal-form claim.

This packet is also a concrete realization of the MetaCIC audit-map synthesis route in section 9376.7: it joins normalization, confluence, decidability, and residual-substitution audit rows while keeping blocked subject-reduction edges visible.

The confluence audit packet of chapter 735 and the parallel-diamond frontier of chapter 4238 consume this handoff only as a guarded boundary route. Their reads keep candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention explicit rather than turning the handoff into a global Church–Rosser or conversion theorem.

Before exporting candidate-normalization confluence data, the handoff consumes the residual-diamond sibling surface of chapter 4238 and the critical-path sibling surface of chapter 765. The handoff is consumed by chapter 4238 only after candidate evidence, residual envelope, decidable boundary, and blocked-edge rows are visible.

Theorem 5547.1 (MetaCIC handoff parallel-diamond forward link)**.** *Any* MetaCICParallelDiamondFrontier$^{\uparrow}$ *consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must expose the candidate row K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, componentwise transport T, displayed replay R, provenance P, and local name L before reading the bounded endpoint row N. The forward dependency is the sibling route through chapter 4238; it is not a global confluence theorem, quotient-syntax equality, unrestricted subject reduction, evaluator soundness, or closed consistency.*

Proof. Start with a parallel-diamond consumer of the handoff. The consumer reaches the packet

only through the displayed candidate-normalization carrier, so the candidate row K and frontier row F remain visible before any bounded endpoint can be read.

Next read the confluence side through the residual envelope C, decidable boundary D, and blocked edge B. These rows are precisely the finite obstruction and boundary data named by the parallel-diamond frontier route, and they prevent N from being treated as an unconditional joinability endpoint.

Finally transport the same route through T, R, P, L. The componentwise transport, replay, provenance, and local name rows bind the endpoint read to the concrete sibling of chapter 4238. Hence every forward read exposes $K, F, C, D, B, T, R, P, L$ before N, while the excluded global MetaCIC claims remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffParallelDiamondForwardLink`

Before exporting candidate-normalization confluence data, the handoff consumes residual-normal-form routing from chapter 8456 and residual critical-pair sieve rows from chapter 8316.

Definition 5547.2 (MetaCIC candidate normalization confluence handoff carrier)**.** *A* MetacicCandidateNormalizati *carrier is a finite* BHist *packet*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

with the following displayed rows:

1. *A is the audit-map anchor row, binding the packet to the candidate-mediated reading of chapter 9383.*
2. *K is the candidate-evidence row, storing Candidate, ClosedCandidate, or CandidateSet membership data before any strong-normalization read is exported.*
3. *N is the normal-endpoint row, recording the bounded normal endpoint or no-infinite-reduction projection obtained from K.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row that keeps candidate evidence and finished-normal evidence inside the displayed frontier packet.*
5. *C is the confluence-facing row, read through the residual envelope of chapter 3921.*
6. *D is the* MetaCICDecidableBoundary$^{\uparrow}$ *row, limiting decidability reads to bounded checker output and normal endpoint comparison.*
7. *B is the blocked-edge row, retaining subject-reduction and residual-substitution obligations as visible boundaries rather than discharged theorems.*
8. *T records componentwise* hsame *transport for the audit, candidate, normal endpoint, frontier, confluence, decidability, blocked-edge, replay, provenance, and local naming rows.*
9. *R records displayed* Cont *replay from a MetaCIC consumer request through A, K, N, F, C, D, B, T.*
10. *P is the* Pkg *provenance over* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICConfluenceAuditWitness$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *L is the local* NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, K, N, F, C, D, B, T, R, P, L$. It has no coordinate for evidence-free normalization, a global parallel diamond, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf
candidate endpoint route

Theorem 5547.3 (MetaCIC candidate normalization confluence handoff boundary)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the audit anchor, candidate evidence, normal endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked-edge row, transport, replay, provenance, and local naming;*
2. *candidate-gate exactness, requiring every normalization read to pass through K and the frontier row F before N is consumed;*
3. *confluence handoff discipline, requiring every residual or join read to pass through C while retaining B as a visible blocked edge;*
4. *decidable-boundary discipline, requiring every conversion-facing read to pass through D after N and before any bounded normal comparison is exported;*
5. *frontier non-collapse, requiring T, R, P, L to preserve the same rows without promoting candidate-mediated SN into an unconditional closed-term theorem or residual-envelope rows into full Church–Rosser.*

No handoff obligation is stored outside $A, K, N, F, C, D, B, T, R, P, L$.

Proof. Project the carrier of Definition 5547.2. The accepted packet displays exactly the audit anchor, candidate evidence, normal endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked-edge, and structural rows.

The normalization stage is A, K, N, F. The audit anchor fixes the candidate-mediated reading; the candidate row K is exposed before the endpoint row N, and the frontier row F keeps both inside the public normalization boundary.

The confluence and decidability stage is C, D, B. A confluence-facing consumer may read residual-envelope data through C, but the blocked-edge row B remains visible. A conversion-facing consumer may read bounded checker output through D only after the normal endpoint has been displayed.

Finally T, R, P, L transport, replay, source, and name the same finite route. Since the carrier contains no evidence-free normalization coordinate, global diamond theorem, unrestricted subject-reduction proof, quotient syntax equality, host evaluator soundness, or closed-consistency endpoint, the listed rows exhaust the local obligation surface. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffBoundary

Theorem 5547.4 (MetaCIC candidate normalization confluence handoff seed closure)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier* $H = (A, K, N, F, C, D, B, T, R, P, L)$*, the seed package closes only when the candidate-evidence row K, normalization-frontier row F, residual confluence row C, decidable-boundary row D, and blocked-edge row B are all displayed before the endpoint row N is consumed. The support rows T, R, P, L preserve this finite route by transport, replay, provenance, and local naming. The package does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffSeedClosure

Proof. Project the carrier of Definition 5547.2. The accepted packet separates candidate evidence K, endpoint data N, frontier data F, residual confluence data C, decidable-boundary data D, blocked-edge data B, and the structural rows T, R, P, L.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, its confluence handoff discipline keeps C and B visible for residual or join reads, and its decidable-boundary discipline places D on the bounded comparison route.

Finally read T, R, P, L only as preservation rows for the same finite package. They transport, replay, source, and name the displayed rows without adding a coordinate for the excluded global endpoints, so the seed closure remains the finite premise-exposure package rather than a global normalization, confluence, decidability, quotient, evaluator, or consistency theorem. □

Theorem 5547.5 (MetaCIC candidate normalization confluence handoff premise exposure)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier* $H = (A, K, N, F, C, D, B, T, R, P, L)$*, every confluence-facing read must expose the candidate row* K*, normalization frontier row* F*, residual envelope row* C*, decidable-boundary row* D*, and blocked-edge row* B *before any bounded endpoint row* N *can be consumed. Thus candidate-mediated normalization is retained as a premise surface for confluence consumers. The handoff refuses unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, and closed consistency.*

Proof. Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, while its confluence handoff discipline keeps C and B visible for every residual or join read.

Now chain the consumer boundary of Definition 5547.14. A public confluence-facing consumer receives the handoff only through the displayed route

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto C, D, B \longmapsto T, R, P, L.$$

The decidable-boundary row D is therefore part of the bounded comparison surface, not a detached global conversion decider.

Since T, R, P, L only transport, replay, source, and name the same finite route, they cannot erase K, F, C, D, B or promote N into an unconditional endpoint theorem. The excluded global claims are not coordinates of H, so every confluence-facing read remains premise-exposed before consuming bounded endpoint data. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffPremiseExposure

Theorem 5547.6 (MetaCIC candidate handoff residual-diamond retention)**.** *For every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

every confluence-facing read must retain the candidate evidence K*, residual envelope* C*, decidable boundary* D*, and blocked-edge row* B *before a bounded endpoint row* N *is treated as joinable. The retained route is consumed through the sibling path from* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *to* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *and remains only a residual-diamond handoff. It does not assert full Church–Rosser, unrestricted subject reduction, quotient syntax equality, or global conversion decidability.*

Proof. Project the carrier of Definition 5547.2. The endpoint row N is separate from the candidate evidence K, residual envelope C, decidable boundary D, and blocked-edge row B, so joinability cannot be read from N alone.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness exposes K and F before N, while its confluence and decidable-boundary clauses keep C, D, B visible on the residual comparison route.

Use premise exposure from Definition 5547.5. It prevents a confluence-facing consumer from consuming the bounded endpoint before K, F, C, D, B are displayed in the same packet.

Finally route the read through the sibling frontier. The critical-path packet $\mathsf{MetaCICCriticalPath}^{\uparrow}$ supplies the public handoff direction, and $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ reads only the residual-diamond boundary, not a global confluence theorem. Since the carrier has no coordinate for Church–Rosser, unrestricted subject reduction, quotient syntax equality, or global conversion decidability, retaining K, C, D, B is exactly the finite residual-diamond condition. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateHandoffResidualDiamondRetention`

Theorem 5547.7 (MetaCIC candidate handoff local join budget)**.** *For every confluence-facing endpoint comparison exported from an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

the consumer must expose a finite local-join budget read through candidate evidence K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, and structural rows T, R, P, L before the endpoint row N is treated as joinable.

The budget is routed through $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICConfluenceAuditPac}$ *and* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *as retained finite evidence. It is not full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.LocalJoinBudget.MetacicCandidateH`

Proof. Start with the boundary and premise-exposure routes, Definition 5547.3 and Definition 5547.5. They place K, F, C, D, B before any confluence-facing read may consume the bounded endpoint row N.

Apply residual-diamond retention from Definition 5547.6. It keeps the residual envelope, decidable boundary, and blocked edge visible while the comparison is routed from $\mathsf{MetaCICCriticalPath}^{\uparrow}$ and $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ toward confluence consumers.

Use the normal-endpoint readback theorem Definition 5547.8. It reads N only after $K, F, C, D, B, T, R, P, L$ have been displayed and names the $\mathsf{MetaCICConfluenceAuditPacket}^{\uparrow}$ route as an audit handoff rather than as evaluator equality.

Finally pass the retained evidence to $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$. The frontier receives the finite local-join budget, not a global confluence theorem, so Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, and closed consistency remain outside the carrier. □

Theorem 5547.8 (MetaCIC candidate handoff normal-endpoint readback)**.** *Any bounded normal endpoint read exported by an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must expose candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, and the transport, replay, provenance, and naming rows T, R, P, L before N is consumed. The read may be handed to* $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, *and*

MetaCICConfluenceAuditPacket$^{\uparrow}$ *consumers only as this finite premise-visible route. It is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffNormalEndpointReadback`

Proof. Project the carrier of Definition 5547.2. The bounded endpoint row N is displayed beside candidate evidence K, frontier evidence F, residual envelope C, decidable boundary D, blocked edge B, and the structural rows T, R, P, L.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness exposes K and F before N, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline keeps D on the bounded comparison route.

Now use the consumer boundary Definition 5547.14. A public read reaches MetaCICCriticalPath$^{\uparrow}$ and MetaCICNormalizationFrontier$^{\uparrow}$ only through the displayed handoff route, so the endpoint row cannot be consumed as a detached normal-form theorem.

Finally read the confluence-audit side through the same residual route. The sibling MetaCICConfluenceAuditPack may consume the bounded endpoint only after C, D, B, T, R, P, L remain visible, and those rows only transport, replay, source, and name the finite packet. Since the excluded global endpoints are not carrier coordinates, the normal-endpoint readback preserves the candidate and residual premises. □

Theorem 5547.9 (Candidate normal endpoint retention)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any accepted endpoint read from H must first expose the candidate gate K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, residual-diamond retention row, and normal endpoint readback row before the endpoint row N is consumed by a confluence or conversion-facing consumer. The retained read is a bounded handoff through $A, K, N, F, C, D, B, T, R, P, L$; it is not global confluence, unrestricted strong normalization, quotient syntax equality, or host evaluator soundness.*

Proof. Start with the boundary theorem Definition 5547.3. It displays the candidate gate, normalization frontier, residual envelope, decidable boundary, and blocked-edge rows as separate carrier coordinates before any endpoint read is available.

Apply premise exposure from Definition 5547.5. A public consumer cannot pass directly to N; it must first retain K, F, C, D, B inside the same accepted packet.

Use residual-diamond retention from Definition 5547.6. This keeps the residual envelope, decidable boundary, and blocked edge visible when the endpoint is read by a confluence-facing route.

Finally apply normal-endpoint readback from Definition 5547.8. The rows T, R, P, L transport, replay, source, and name the same bounded route before N is consumed, so the endpoint cannot be reinterpreted as any refused global MetaCIC principle. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.10 (MetaCIC finite endpoint comparison handoff)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any endpoint comparison exported from H to a confluence or conversion consumer must first expose candidate evidence K, normalization-frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L before bounded normal endpoints are compared through N. The exported comparison is a finite candidate-mediated handoff; it is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. Candidate-gate exactness exposes K and F before N, while confluence discipline keeps C and B visible and decidable-boundary discipline places D on the bounded comparison route.

Apply premise exposure from Definition 5547.5. A confluence-facing consumer cannot read the bounded endpoint row until the candidate, frontier, residual-envelope, decidable-boundary, and blocked-edge rows are displayed in the same packet.

Use residual-diamond retention from Definition 5547.6. It prevents a comparison from treating N as detached joinability data; the residual envelope and blocked edge remain part of the read.

Finally apply normal-endpoint readback from Definition 5547.8. The structural rows T, R, P, L transport, replay, source, and name the same finite route before N is consumed, so the comparison cannot export any of the refused global MetaCIC endpoints. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffFiniteEndpointComparison`

Theorem 5547.11 (MetaCIC candidate handoff decidable normal-form comparison)**.** *Any bounded normal-form comparison exported to a decidability consumer from an accepted* MetacicCandidateNormalizationConflu *carrier must expose candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked edge B, and structural rows T, R, P, L before the endpoint row N is compared. The comparison is obtained only by chaining boundary exposure, premise exposure, residual-diamond retention, and finite endpoint comparison. It refuses full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, and closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes the candidate row K, frontier row F, residual envelope C, decidable boundary D, and blocked edge B before any endpoint comparison is available.

Apply premise exposure from Definition 5547.5. The bounded endpoint row N remains downstream of the candidate and frontier rows, so the comparison still has to read the finite handoff surface.

Now use residual-diamond retention from Definition 5547.6. The residual envelope and blocked edge remain visible on the comparison route, so the decidability consumer cannot erase the guarded premise surface.

Finally apply finite endpoint comparison from Definition 5547.10. The rows T, R, P, L transport, replay, source, and name the same finite route before N is compared, which is exactly the bounded comparison route refused by the excluded global claims. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffDecidableN`

Theorem 5547.12 (MetaCIC candidate handoff residual-decider inversion)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a confluence or conversion consumer reads a bounded decider outcome from H, then the candidate row K, normalization frontier F, residual envelope C, decidable boundary D, blocked-edge row B, and structural rows T, R, P, L are already visible before the endpoint row N is compared. The inversion is a finite retained-premise read; it does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes candidate evidence, the normalization frontier, residual envelope, decidable boundary, and blocked-edge rows as separate coordinates before any bounded endpoint comparison can be read.

Apply premise exposure from Definition 5547.5. A confluence-facing read cannot consume the endpoint row before K, F, C, D, B are displayed in the same packet.

Use residual-diamond retention and finite endpoint comparison, Definition 5547.6 and Definition 5547.10. They keep the residual envelope, decidable boundary, and blocked edge attached to any bounded endpoint comparison offered to a confluence or conversion consumer.

Finally apply decider-row exhaustion through Definition 5547.11. The bounded decider outcome is read only after D is on the displayed comparison route and after T, R, P, L transport, replay, source, and name the same finite handoff. Since the excluded global endpoints are not carrier coordinates, the decider read inverts back to the retained candidate and residual premises rather than to any unrestricted MetaCIC theorem. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffResidualDe`

Theorem 5547.13 (MetaCIC candidate handoff frontier normal-endpoint retention)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any bounded normal endpoint exported by H to a confluence consumer that also reads* MetaCICParallelDiamondFrontier$^{\uparrow}$ *must retain candidate evidence K, frontier row F, residual envelope C, decidable boundary D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L before the endpoint row N is consumed. The same read must preserve the parallel-diamond frontier's residual and local-join rows as visible premises. It does not assert full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.*

Proof. Start with premise exposure and residual-diamond retention, Definition 5547.5 and Definition 5547.6. These require the candidate row, frontier row, residual envelope, decidable boundary, and blocked edge to remain visible before any confluence-facing endpoint read is accepted.

Apply normal-endpoint readback, Definition 5547.8. The bounded endpoint row N is consumed only through the same candidate, frontier, residual, boundary, blocked-edge, transport, replay, provenance, and local naming route.

Finally route the consumer through the sibling frontier dependency. The MetaCICParallelDiamondFrontier$^{\uparrow}$ route named by Definition 5547.6 and chapter 4238 keeps residual and local-join rows visible before the endpoint is used. Since none of the refused global endpoints is a carrier coordinate, the handoff retains only the bounded frontier premises. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffFrontierNo`

Theorem 5547.14 (MetaCIC candidate handoff consumer boundary)**.** *Every public consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet factors through*

$$\mathsf{MetaCICCriticalPath}^{\uparrow} \longmapsto \mathsf{MetaCICNormalizationFrontier}^{\uparrow} \longmapsto C, D, B \longmapsto T, R, P, L.$$

The consumer receives candidate-mediated normalization evidence, bounded normal endpoint data, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention. It receives no unconditional strong-normalization theorem, full Church–Rosser theorem, unrestricted subject-reduction theorem, global conversion decider, or completed closed-consistency proof.

Proof. Start from the carrier projection. The audit and critical-path rows determine which candidate-mediated normalization and boundary rows are visible to the consumer.

Apply Definition 5547.3. The candidate gate exposes K, F, N before any normal endpoint is available, while C, D, B retain the confluence and decidability boundaries as displayed rows.

The structural rows T, R, P, L do not add another route. They preserve the same carrier under componentwise transport, replay it for downstream consumers, record provenance, and name the packet locally. The absent global theorems cannot be projected from those rows. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.15 (MetaCIC candidate normalization residual progress)**.** *An accepted* MetacicCandidateNormaliza *packet consumes residual data only as a progress ledger. Every admitted candidate-normalization row read through $A, K, N, F, C, D, B, T, R, P, L$ either forwards a retained premise through the candidate and frontier rows K, F or emits a residual obstruction row through C and B, with the residual boundary read against chapter 5126. The packet does not assert full confluence, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The candidate-normalization data are A, K, N, F, the residual and decidability boundary rows are C, D, B, and the structural rows are T, R, P, L.

Apply Definition 5547.3. The candidate gate forces any normalization row to expose K and F before a normal endpoint is consumed, so a retained premise remains visible when the row is forwarded.

If the row cannot be forwarded as a retained premise, the same boundary theorem routes the read through C and B. The residual envelope is then consumed as a ledger compatible with chapter 5126, while the blocked-edge row records the obstruction rather than discharging it.

Finally use Definition 5547.14. The public consumer receives only the candidate-mediated route, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and structural rows. Since the excluded global endpoints are not carrier coordinates, the progress ledger cannot be read as a full confluence or strong-normalization theorem. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.16 (MetaCIC candidate normalization confluence bounded join window)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P, L)$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. A confluence-facing consumer may export a join read from H only after the candidate evidence K, normalization-frontier row F, residual envelope C, decidable-boundary row D, blocked-edge row B, and structural rows T, R, P, L remain visible before the endpoint row N is consumed. The window is finite and carrier-internal: it does not assert full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Proof. Start with the handoff boundary of Definition 5547.3. It exposes K and F before N, keeps C and B visible for confluence-facing reads, and routes bounded comparison through D.

Use premise exposure from Definition 5547.5. A public confluence consumer cannot read the endpoint row before the candidate, frontier, residual-envelope, decidable-boundary, and blocked-edge rows are displayed in the same packet.

Apply residual-diamond retention from Definition 5547.6. It prevents a join read from treating N as detached joinability data; the read must retain K, C, D, B on the residual route.

Finally use normal-endpoint readback from Definition 5547.8. The structural rows T, R, P, L preserve, replay, source, and name the same finite route before N is consumed. Since no excluded global endpoint is a carrier coordinate, the exported join is available only through this bounded window. □

Theorem 5547.17 (MetaCIC candidate normalization confluence rectangle)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If both candidate-normalization edges of a bounded comparison are admitted by the candidate gate and each residual edge is read through the residual envelope, then the handoff packet either displays a common bounded endpoint row or records a residual obstruction row through C and B. The rectangle is guarded by the candidate and residual rows; it does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.*

Proof. Start with Definition 5547.3. The candidate gate forces both normalization edges to expose K and F before any endpoint row N can be consumed.

Use the consumer boundary of Definition 5547.14. A public handoff read must pass through the residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L.

Apply residual progress from Definition 5547.15. If the two candidate-normalization edges forward through the retained premises, the packet displays the bounded endpoint row shared by those reads. If one residual edge cannot be forwarded, the same route records the obstruction through C and B.

The residual side is read against chapter 5126. Since the carrier contains no coordinates for unrestricted Church–Rosser, unconditional strong normalization, subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency, the rectangle remains a guarded handoff packet. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.18 (MetaCIC candidate normalization confluence handoff candidate residual join)**.** *Let $H = (A, K, N, F, C, D, B, T, R, P, L)$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If two candidate-normalization edges exported from H expose candidate evidence K and frontier evidence F, and every residual edge is routed through the residual-envelope row C and blocked-edge row B, then any exported join row is guarded by K, F, C, D, B, and the structural rows T, R, P, L. The join row is a candidate-mediated residual handoff; it is not full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Proof. Begin with the boundary theorem Definition 5547.3. It forces any endpoint or comparison read to display K, F, N before the consumer sees bounded normalization data, and it keeps C, D, B visible on the residual and decidable-boundary side.

Use residual progress from Definition 5547.15. Under the stated routing, residual edges either remain retained candidate premises through K, F or become obstruction rows through C, B; neither route removes the candidate evidence.

Finally apply the rectangle theorem Definition 5547.17. The exported join row is therefore read only as the guarded rectangle output, with T, R, P, L preserving the same finite route by transport, replay, provenance, and local naming. Since no excluded global theorem is a carrier coordinate, the join cannot be promoted beyond the candidate-mediated residual handoff. □

Theorem 5547.19 (MetaCIC candidate normalization parallel-frontier consumption)**.** *If a* MetaCICParallelDiamon *consumer reads a join row from an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

then the read must preserve candidate evidence K, normalization-frontier data F, residual-envelope data C, decidable-boundary data D, blocked-edge data B, componentwise transport T, displayed replay R, provenance P, and local naming L before N can be treated as bounded endpoint data. The exported join is only a candidate-mediated residual handoff; it is not full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, evaluator soundness, or closed consistency.

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Proof. Start with the boundary theorem Definition 5547.3. It prevents any endpoint read from reaching N before the candidate evidence K, normalization frontier F, residual envelope C, decidable boundary D, and blocked-edge row B remain visible.

Apply the premise-exposure route of Definition 5547.14. A public consumer receives the handoff only through the same carrier rows and through the structural data T, R, P, L.

Use residual progress from Definition 5547.15 and the rectangle theorem Definition 5547.17. They keep retained candidate premises and residual obstructions attached to K, F, C, D, B while allowing only bounded endpoint data or an obstruction row to be displayed.

Finally invoke the candidate residual join theorem Definition 5547.18. The join already exported by H is guarded by the candidate, frontier, residual, blocked-edge, and structural rows. A MetaCICParallelDiamondFrontier$^{\uparrow}$ consumer therefore receives a candidate-mediated residual handoff for bounded endpoint data, not any of the excluded global confluence, subject-reduction, conversion-decidability, quotient-equality, evaluator-soundness, or consistency claims. □

Theorem 5547.20 (MetaCIC candidate normalization confluence boundary)**.** *Let $H = (A, K, N, F, C, D, B, T, R,$ be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a bounded comparison exports a join row from H, then the read preserves candidate evidence K, the normalization frontier F, the residual envelope C, the decidable boundary D, the blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. The exported join row is therefore a guarded candidate-normalization handoff. It cannot be read as unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. A bounded comparison can export a join row only through the displayed candidate, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and naming coordinates of H.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness keeps K and F visible before endpoint data are consumed, confluence discipline keeps C and B visible for residual reads, and decidable-boundary discipline routes bounded comparison through D.

Use residual progress and the rectangle theorem from Definition 5547.15 and Definition 5547.17. They allow the bounded comparison to display a guarded endpoint row or an obstruction row, but only while retaining the candidate and residual coordinates.

Finally read the structural rows T, R, P, L. They transport, replay, source, and name the same finite route; none of them creates a coordinate for unconditional normalization, full Church–Rosser, unrestricted subject reduction, global decidability, quotient syntax equality, evaluator soundness, or closed consistency. Thus no excluded coordinate is available to the consumer. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceBoundary`

Theorem 5547.21 (MetaCIC candidate normalization confluence handoff decider boundary)**.** *Any bounded conversion comparison exported through the decidable-boundary row D of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$ must first expose candidate evidence K, normal endpoint data N, the normalization frontier row F, the residual-envelope row C, the blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. The exported comparison is only a bounded comparison boundary. It is not global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The only conversion-facing coordinate is D, and it is displayed alongside the candidate, endpoint, frontier, residual-envelope, blocked-edge, and structural rows.

Apply Definition 5547.3. Its candidate-gate exactness forces K, F, N to appear before a normal endpoint can be consumed, and its confluence handoff discipline keeps C and B visible before any residual or join read is offered.

Next use Definition 5547.14. A public consumer receives the bounded comparison only through the same carrier route and through T, R, P, L; the structural rows do not create a second decider surface.

Finally apply Definition 5547.15. Residual data remain progress ledger entries or obstruction rows, so the bounded comparison exported through D cannot be promoted to the excluded global conversion, confluence, subject-reduction, evaluator-equality, quotient-equality, or consistency claims. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.22 (MetaCIC candidate handoff decider-row exhaustion)**.** *Any bounded conversion comparison exported through row D of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$ must preserve the candidate evidence row K, endpoint row N, normalization frontier row F, residual envelope row C, blocked-edge row B, componentwise transport row T, displayed replay row R, provenance row P, and local naming row L. No consumer may read D as a standalone global conversion decider. The exhausted row remains only a bounded comparison surface and does not assert full Church–Rosser, unconditional strong normalization, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The decidable-boundary row D appears only as one coordinate of H, beside the candidate, endpoint, frontier, residual-envelope, blocked-edge, and structural rows.

Apply the boundary theorem Definition 5547.3. It places bounded comparison reads after candidate evidence, frontier evidence, endpoint data, residual-envelope visibility, and blocked-edge retention.

Apply the consumer boundary Definition 5547.14. A public consumer receives D only through the displayed route with T, R, P, L, so there is no detached decider surface outside the carrier.

Finally use Definition 5547.21. That theorem identifies D as the bounded comparison row after the displayed endpoint and residual rows have been exposed. Since the excluded global endpoints are not carrier coordinates, exhausting row D preserves the finite comparison boundary without collapsing into global decidability. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.23 (MetaCIC candidate normalization confluence retained-premise inversion)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. If a bounded endpoint row or join row is read from H, then the read determines the same candidate evidence K, normalization frontier F, residual envelope C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L. Dropping any retained row changes the packet and cannot be presented as the same confluence handoff.*

Proof. Project the carrier of Definition 5547.2. The endpoint and join rows are not independent data; they are read from the displayed tuple with $K, F, C, D, B, T, R, P, L$ still visible.

Apply the seed package and boundary results from Definition 5547.4, Definition 5547.5, and Definition 5547.3. These force candidate evidence, frontier data, residual-envelope visibility, decidable-boundary access, and blocked-edge retention to remain part of the same handoff before any endpoint or join row is consumed.

Use the bounded conversion comparison surface of Definition 5547.21 and Definition 5547.22. The bounded comparison row D can be read only together with the structural rows T, R, P, L, so erasing any retained coordinate changes the packet rather than simplifying the same one.

Finally apply the nonescape clauses in Definition 5547.20 and Definition 5547.14. They rule out parameter transport into a detached endpoint or join claim, leaving only constructor-style inversion on the concrete carrier rows. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Definition 5547.24 (MetaCIC finite conversion certificate carrier)**.** *A* MetacicCandidateNormalizationConfluenceHa *finite conversion certificate is the subroute of an accepted carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

that displays the bounded normal endpoint row N, normalization frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and structural rows T, R, P, L in one route. The certificate keeps K as its candidate-evidence entry condition and uses A only as the audit anchor inherited from the surrounding carrier. It has no coordinate for global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.

Theorem 5547.25 (MetaCIC finite conversion certificate boundary)**.** *Every bounded conversion read exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must display, in one finite route, the normal endpoint row N, normalization frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport row T, displayed replay row R, provenance row P, and local naming row L. Such a read is a finite conversion certificate in the sense of Definition 5547.24. It is not global conversion decidability, full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Begin with the finite certificate carrier. It is not a second carrier: it is the displayed $N, F, C, D, B, T, R, P, L$ subroute of Definition 5547.2, with K as the candidate-evidence entry condition and A as the audit anchor.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness places K, F, N before a bounded normal endpoint is consumed, while the confluence handoff discipline keeps C and B visible for residual reads.

Use the consumer boundary of Definition 5547.14. A public bounded conversion consumer receives the comparison only through the same structural route T, R, P, L and not through a detached evaluator or quotient-syntax endpoint.

Finally combine residual progress with the decider boundary of Definition 5547.15 and Definition 5547.21. Residual entries remain progress or obstruction rows, and the exported comparison passes through D as a bounded boundary. Since the excluded global claims are not carrier coordinates, the finite certificate cannot be read as global conversion decidability, Church–Rosser, subject reduction, evaluator equality, quotient equality, or consistency. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicFiniteConversionCertificat`

Theorem 5547.26 (MetaCIC candidate handoff obligation rows)**.** *Any obligationClosure upgrade for* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *must display candidate evidence K, normal endpoint data N, frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L as non-collapsible rows. In particular, the candidate-evidence row cannot be folded into the normal endpoint row, the residual-envelope row cannot be folded into the decidable-boundary row, and the blocked-edge row cannot be erased by transport or replay. The upgrade remains a finite handoff obligation surface; it does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffObligationRows`

Proof. Project the carrier of Definition 5547.2. It displays $A, K, N, F, C, D, B, T, R, P, L$ as separate rows, so an obligationClosure read must account for candidate evidence, normal endpoint data, frontier discipline, residual-envelope data, decidable-boundary data, blocked edges, and the structural route.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness keeps K and F visible before N is consumed, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline keeps D downstream of the displayed endpoint and residual rows.

Use the consumer boundary, residual progress, decider boundary, and finite conversion certificate boundary from Definition 5547.14, Definition 5547.15, Definition 5547.21, and Definition 5547.25. Together they force public reads to preserve candidate-mediated normalization,

residual-envelope visibility, bounded conversion comparison, blocked-edge retention, transport, replay, provenance, and local naming as the same finite route.

Therefore collapsing any one of $K, N, F, C, D, B, T, R, P, L$ would remove a row required by the carrier and by the displayed boundary theorems. Since the excluded global endpoints are not carrier coordinates, the obligation rows remain non-collapsible without promoting the packet beyond the finite handoff surface. □

Theorem 5547.27 (MetaCIC candidate handoff row readiness)**.** *Every confluence-facing or decidability-facing read of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

must expose the candidate row K and the frontier row F before the endpoint row N or the residual rows C, D, B are consumed. Thus the handoff exports only a finite premise-readiness route, not a global strong-normalization endpoint, full Church–Rosser theorem, unrestricted subject-reduction theorem, or global conversion decider.

Proof. Project the carrier of Definition 5547.2. The rows K and F are separate from N, C, D, B, so a consumer can only read endpoint, residual, or decidability data through the displayed handoff packet.

Apply the boundary theorem Definition 5547.3. Its candidate-gate exactness places K and F before N, while its confluence and decidability clauses keep C, D, B on the residual and bounded-comparison side.

Use the obligation rows theorem Definition 5547.26. It makes $K, N, F, C, D, B, T, R, P, L$ non-collapsible rows, so neither transport nor replay may hide the candidate and frontier premises before a consumer reaches the endpoint or residual rows.

Since the carrier contains no coordinate for global strong normalization, full Church–Rosser, unrestricted subject reduction, or global conversion decidability, the exposed route is exactly a local premise-readiness route for confluence-facing and decidability-facing consumers. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.28 (MetaCIC candidate retained-edge handoff)**.** *Every finite successor handoff from candidate normalization toward confluence in an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the candidate-gate rows K, F, N, the residual-envelope row C, the decidable-boundary row D, the blocked-edge row B, and the structural rows T, R, P, L before any consumer reads endpoint or conversion data. The retained edge is a finite successor route inside $H = (A, K, N, F, C, D, B, T, R, P, L)$; it does not assert Church–Rosser, strong normalization, subject reduction, evaluator equality, quotient equality, or consistency.*

Proof. Start with the boundary theorem Definition 5547.3. Candidate-gate exactness keeps K, F, N visible in the successor handoff, while confluence handoff discipline keeps C and B visible.

Apply the consumer boundary of Definition 5547.14. A confluence-facing consumer receives the handoff only through the displayed public route, so endpoint data cannot be read before the retained candidate and residual rows.

Use the decider boundary from Definition 5547.21. Any bounded conversion comparison passes through D after the candidate, endpoint, residual-envelope, blocked-edge, and structural rows are displayed.

Finally apply Definition 5547.26. It makes $K, F, N, C, D, B, T, R, P, L$ non-collapsible rows of the obligation surface. Since the excluded global endpoints are not carrier coordinates, a finite

successor handoff retains those rows without proving Church–Rosser, strong normalization, subject reduction, evaluator equality, quotient equality, or consistency. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.29 (MetaCIC candidate normalization handoff obligation stability)**.** *Let* $H = (A, K, N, F, C, D, B, T, R, P, L)$ *be an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier. Any public read preserving the handoff rows must preserve candidate evidence* K*, endpoint row* N*, frontier row* F*, residual envelope* C*, decidable boundary* D*, blocked edge* B*, componentwise transport* T*, displayed replay* R*, provenance* P*, and local naming* L*. No* hsame *transport or* Cont *replay may erase candidate or residual obligations, and no public read may collapse the handoff into unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationHand

Proof. Project the carrier of Definition 5547.2. The rows $K, N, F, C, D, B, T, R, P, L$ occur as separate coordinates, with A only anchoring the audit-map source.

Apply the boundary theorem Definition 5547.3. Candidate-gate exactness preserves K, F, N, confluence handoff discipline preserves C and B, and decidable-boundary discipline preserves D as the bounded comparison boundary.

Use residual progress from Definition 5547.15. Residual data either remain retained premises through K, F or obstruction rows through C, B, so a public read cannot erase the candidate or residual obligation while still preserving the handoff.

Apply the finite conversion certificate boundary of Definition 5547.25 and the obligation rows theorem Definition 5547.26. They require $N, F, C, D, B, T, R, P, L$ to remain visible in one finite route and require K as the candidate-evidence entry condition.

Finally read T and R only as componentwise transport and displayed replay. They preserve the already displayed rows and do not create a coordinate that can discharge or remove candidate evidence, residual-envelope data, or blocked edges. The excluded global endpoints are not carrier coordinates, so stability stays inside the finite handoff surface. □

Theorem 5547.30 (MetaCIC finite conversion endpoint handoff)**.** *Every finite conversion certificate exposed by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet reaches residual confluence consumers only through the candidate-normalization row* K*, the residual-envelope row* C*, the blocked-edge row* B*, and the decidable-boundary row* D*. In particular, a consumer of the certificate must read the endpoint route through Definition 5547.24, Definition 5547.25, Definition 5547.15, and Definition 5547.17. The handoff does not assert full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, global conversion decidability, quotient syntax equality, host evaluator soundness, or consistency.*

Proof. Project the finite certificate from the carrier. By Definition 5547.24, the exposed endpoint is the displayed subroute through $N, F, C, D, B, T, R, P, L$, with K retained as the candidate-evidence entry condition.

Apply Definition 5547.25. The bounded conversion read is not detached from the handoff packet: it must display the normal endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows in one route.

Use residual progress and the guarded rectangle from Definition 5547.15 and Definition 5547.17. A residual confluence consumer either receives the bounded endpoint row through the candidate-mediated route or receives an obstruction row through C and B.

Finally apply the handoff decider boundary of Definition 5547.21. The bounded comparison passes through D after the displayed endpoint and residual rows, so no consumer can read the certificate as one of the excluded global endpoints. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicFiniteConversionEndpointHa

Theorem 5547.31 (MetaCIC candidate normalization public handoff)**.** *Any public confluence or decidability consumer of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet preserves candidate evidence K, normal endpoint N, normalization frontier F, residual envelope C, decidable boundary D, blocked edge B, transport T, replay R, provenance P, and local naming L before reading a bounded endpoint or conversion row. The handoff does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. First apply the finite conversion certificate boundary of Definition 5547.25. It keeps the bounded endpoint read inside the displayed endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming route.

Next use residual progress from Definition 5547.15. Residual data remain visible either as retained candidate premises through K, F or as obstruction rows through C, B, so a public confluence consumer cannot erase them before reading a bounded endpoint.

Now apply the confluence rectangle and decidable-boundary theorems, Definition 5547.17 and Definition 5547.21. They route bounded comparison through D only after the candidate, endpoint, frontier, residual-envelope, and blocked-edge rows have been displayed.

Finally combine obligation stability and public endpoint handoff from Definition 5547.29 and Definition 5547.30. The structural rows T, R, P, L preserve the same finite packet, and none of the excluded global endpoints is a carrier coordinate. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.32 (MetaCIC normalization audit-map realization)**.** *The* MetacicCandidateNormalizationConfluenc *packet is the concrete finite* NameCert *realization of the normalization audit-map route of chapter 9383. Its carrier displays candidate evidence K, bounded endpoint row N, frontier row F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local naming L as one guarded handoff. The realization does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The audit anchor A binds the packet to chapter 9383, and the rows $K, N, F, C, D, B, T, R, P, L$ display the finite candidate, endpoint, frontier, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming data.

Apply the handoff boundary of Definition 5547.3. Candidate-gate exactness forces K, F, N to appear before any endpoint read, confluence handoff discipline keeps C and B visible, and decidable-boundary discipline routes bounded comparison through D.

Use the consumer boundary and endpoint handoff from Definition 5547.14 and Definition 5547.30. Together they show that every public read of the audit-map route remains inside the displayed finite conversion certificate and its residual-envelope boundary.

Thus the vision route has a concrete finite packet: candidate evidence, bounded endpoint, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and

naming are all carrier rows. Since none of the excluded global endpoints is a carrier coordinate, the realization stays a guarded NameCert handoff. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.33 (Finite-observation MetaCIC diamond-envelope obligations)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet, a finite-observation diamond envelope exposes candidate evidence K, bounded endpoint row N, residual-envelope row C, local join and checker status through the decidable-boundary row D, blocked-edge retention through B, componentwise transport T, displayed replay R, provenance P, and local naming L before any confluence consumer reads a bounded join. The envelope is a finite observation route inside the existing handoff carrier; it is not unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.FiniteObservationMetaCICDiamondEn

Proof. Project the carrier of Definition 5547.2. The finite observation envelope can only use the displayed rows $K, N, F, C, D, B, T, R, P, L$.

Apply Definition 5547.3. Candidate-gate exactness keeps K, F, N visible before a bounded endpoint is consumed, while confluence handoff discipline keeps the residual-envelope row C and blocked-edge row B visible before a residual or join read is offered.

Use Definition 5547.21. The local checker status is read through D only after the endpoint, residual-envelope, blocked-edge, transport, replay, provenance, and naming rows are displayed.

Finally apply Definition 5547.30. Any bounded join read remains inside the finite certificate route, so no consumer can read the envelope as an unconditional confluence or normalization theorem. □

Theorem 5547.34 (MetaCIC candidate-normalization confluence handoff obligation surface)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet, the handoff exposes finite obligation rows for candidate evidence K, residual envelope C, decidable-boundary routing D, blocked-edge retention B, bounded endpoint readback N, componentwise transport T, replay R, provenance P, and local naming L. The route remains guarded by the audit anchor A and frontier row F, and it exports none of strong normalization, full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The displayed rows $A, K, N, F, C, D, B, T, R, P, L$ are the only coordinates available to a confluence-facing consumer.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. They force candidate evidence and frontier data to remain visible before endpoint, residual, or bounded-comparison rows are consumed.

Use residual-diamond retention from Definition 5547.6 and normal-endpoint readback from Definition 5547.8. These route residual evidence through C, B and endpoint reads through N without collapsing the decidable-boundary row D.

Finally read T, R, P, L as structural rows. They transport, replay, source, and locally name the same finite handoff, so the obligation surface is exhausted by the displayed coordinates and cannot be reinterpreted as any of the refused global MetaCIC endpoints. □

Lean4 ✓ BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf

Theorem 5547.35 (Finite-observation MetaCIC diamond-envelope handoff)**.** *Every confluence or normalization consumer of a finite-observation MetaCIC diamond envelope receives a bounded join only through the candidate evidence, bounded endpoint, residual-envelope, local checker, blocked-edge, transport, replay, provenance, and local* NameCert *rows of Definition 5547.33. The handoff exports the bounded finite-observation route and no global Church–Rosser theorem, unconditional strong-normalization theorem, unrestricted subject-reduction theorem, global conversion decider, evaluator equality, quotient syntax equality, or consistency endpoint.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.FiniteObservationMetaCICDiamondEn`

Proof. Start with the finite-observation obligations. They place the candidate, endpoint, residual-envelope, checker, blocked-edge, transport, replay, provenance, and naming rows before any bounded join read.

Use the guarded rectangle of Definition 5547.17. A bounded comparison either displays a common bounded endpoint row or records a residual obstruction, and both alternatives keep the candidate and residual coordinates visible.

Apply the finite conversion certificate boundary of Definition 5547.25. The public consumer receives only the displayed finite route, so the handoff cannot be reinterpreted as a detached global confluence or normalization theorem. □

Theorem 5547.36 (MetaCIC candidate-normalization residual-diamond totality)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet that exposes candidate evidence K, bounded normal endpoint row N, residual-retention rows C, B, and decidable-boundary row D must supply the finite residual-diamond join route before any public confluence consumer reads it. The supplied route is the bounded candidate-normalization handoff through $A, K, N, F, C, D, B, T, R, P, L$; it is not a global Church–Rosser theorem, unrestricted subject-reduction theorem, unconditional strong-normalization theorem, quotient syntax equality, host evaluator soundness, global conversion decidability, or closed consistency.*

Proof. Project the carrier of Definition 5547.2. The rows K, N, C, D, B are displayed beside the audit anchor, frontier row, componentwise transport, replay, provenance, and local naming rows, so a public confluence read has no coordinate outside $A, K, N, F, C, D, B, T, R, P, L$ from which to obtain a join.

Apply residual-diamond retention from Definition 5547.6. It requires residual or join reads to retain candidate evidence, the residual envelope, the decidable boundary, and the blocked-edge row before the endpoint row is treated as joinable.

Use the consumer boundary of Definition 5547.14. A public consumer may read the handoff only after the candidate-mediated route, residual-envelope visibility, bounded comparison routing, blocked-edge retention, transport, replay, provenance, and local naming rows remain visible.

Finally combine the guarded rectangle of Definition 5547.17 with the finite-observation handoff of Definition 5547.35. The available join is therefore total only as a finite residual-diamond route supplied by the packet, and the excluded global endpoints remain absent from the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.37 (MetaCIC candidate-normalization residual-strip local join)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet exposing candidate evidence K, bounded endpoint row N, residual envelope C, decidable boundary D, blocked-edge row B, componentwise transport T, displayed replay R, provenance P, and local* NameCert *row L supplies only a local residual-strip join route for bounded candidate-normalization consumers. The route is read through*

the finite handoff rows $A, K, N, F, C, D, B, T, R, P, L$ and does not export full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, quotient syntax equality, host evaluator soundness, global conversion decidability, or kernel consistency.

Proof. Project the obligation surface of Definition 5547.34. It displays the candidate evidence, residual envelope, decidable boundary, blocked edge, bounded endpoint, transport, replay, provenance, and local naming rows as the only handoff coordinates available to a confluence-facing consumer.

Apply the finite-observation handoff of Definition 5547.35. A bounded join read is offered only after the finite observation envelope has kept the candidate row, endpoint row, residual row, checker row, blocked edge, and structural rows visible.

Use residual-diamond totality from Definition 5547.36. It supplies the join only as a finite residual-diamond route inside $A, K, N, F, C, D, B, T, R, P, L$.

The nonescape clauses are therefore inherited from the same three routes. Since none of full Church–Rosser, unrestricted subject reduction, unconditional strong normalization, quotient syntax equality, host evaluator soundness, global conversion decidability, or kernel consistency is a carrier coordinate, the consumer receives a local residual-strip join and nothing global. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Theorem 5547.38 (MetaCIC candidate-normalization local diamond handoff)**.** *Let*

$$H = (A, K, N, F, C, D, B, T, R, P, L)$$

be an accepted $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier. If two one-step candidate-normalization residual reads share the audit anchor A, candidate evidence K, normalization frontier F, residual-envelope row C, decidable-boundary row D, blocked-edge row B, and the structural rows T, R, P, L, then any displayed join between their bounded endpoint reads is available only as a local diamond handoff through the residual-window row, candidate-join row, relation-scope row, and transport rows already carried by C, D, B, T, R, P, L.*

The handoff preserves the same boundary rows on both sides of the diamond. It exports no full Church–Rosser theorem, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or unrestricted normalization theorem.

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicNormalizationConfluenceLo`

Proof. Start with the carrier and boundary rows. The carrier displays $A, K, N, F, C, D, B, T, R, P, L$, and Definition 5547.3 requires every confluence-facing read to keep candidate evidence, frontier data, residual envelope, decidable boundary, and blocked-edge retention visible.

Read the two one-step residuals through the same boundary data. Premise exposure and residual-diamond retention, Definition 5547.5 and Definition 5547.6, prevent either branch from treating the bounded endpoint row as detached joinability data.

Use residual-diamond totality and residual-strip local join, Definition 5547.36 and Definition 5547.37. They provide only the finite join route inside $A, K, N, F, C, D, B, T, R, P, L$, so the residual-window, candidate-join, and relation-scope rows remain displayed during the handoff.

Finally transport both sides through T, R, P, L. These rows preserve, replay, source, and name the same local diamond route; they do not create a coordinate for the refused global MetaCIC endpoints. Hence the displayed diamond is a local handoff over the retained residual and candidate rows, not a global confluence or normalization theorem. □

Theorem 5547.39 (MetaCIC candidate-normalization confluence handoff seed status)**.** *A seed-status reader of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must expose candidate evidence K, normalization frontier F, residual confluence envelope C, decidable boundary D, blocked-edge row B, componentwise* hsame *transport T, displayed* Cont *replay R,* Pkg *provenance P, and local* NameCert *row L before it consumes any bounded normal endpoint row N. The seed status is the finite retained-premise boundary of the handoff; it does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. The only available rows are the audit anchor, candidate evidence, bounded endpoint, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming data.

Apply the boundary theorem Definition 5547.3 and the seed-closure route of Definition 5547.4. They keep K, F, C, D, B visible before endpoint consumption.

Use premise exposure from Definition 5547.5, residual-diamond retention from Definition 5547.6, and decider-row exhaustion from Definition 5547.22. These force the residual envelope, decidable boundary, and blocked-edge row to remain displayed whenever a bounded normal endpoint is read.

Finally read T, R, P, L as structural rows. They transport, replay, source, and name the same finite packet, so the seed-status reader cannot project any of the refused global MetaCIC endpoints from the handoff. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Falsifiable prediction: Every accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ read that exports a normal endpoint, residual-envelope row, or bounded conversion comparison without first exposing candidate evidence, the normalization frontier, the confluence residual envelope, the decidable boundary, blocked-edge retention, transport, replay, provenance, and local naming is outside the carrier surface.

Independence witness: metaciccriticalpath, metacicnormalizationfrontier, metacic-confluence-residual-envelope, metacicdecidableboundary, metacic-normalization-confluence-bridge: no listed sibling supplies the joint handoff containing candidate evidence, bounded normal endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local NameCert rows while refusing unconditional SN and Church–Rosser.

Lean follow-up should provide Nontrivial and FieldFaithful instances for the carrier so the candidate row, residual-envelope row, and decidable-boundary row can be audited as distinct fields rather than as a collapsed route.

The handoff also consumes the residual-diamond route of chapter 5126 as a concrete prerequisite. The candidate-normalization row must expose its cut-height budget and residual join data before the confluence handoff can be read, so the rows K, N, F, C, D, B remain tied to the parallel residual packet rather than becoming a standalone confluence claim.

Theorem 5547.40 (MetaCIC candidate normalization handoff sibling route)**.** *No* MetacicCandidateNormalizationC *consumer may read bounded normal endpoints or local residual-strip joins until it has exposed the candidate evidence row K, the* MetaCICNormalizationFrontier$^{\uparrow}$ *row F, the residual confluence-envelope row C, the decidable-boundary row D, the blocked-edge row B, and the sibling residual-diamond route through chapter 5126. The available read is the finite handoff along $A, K, N, F, C, D, B, T, R, P, L$; it is not full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, or kernel consistency.*

Proof. Start with the boundary theorem Definition 5547.3 and premise exposure Definition 5547.5. They force the candidate evidence, normalization frontier, residual envelope, decidable boundary, and blocked edge to remain displayed before a bounded endpoint can be consumed.

Residual-diamond retention, Definition 5547.6, ties any join read to K, C, D, B rather than to the endpoint row alone. Decider-row exhaustion in Definition 5547.22 keeps the decidable-boundary row as bounded checker evidence, not as a global conversion oracle.

Use the residual-strip local join theorem Definition 5547.37. Its join route is local to the displayed packet $A, K, N, F, C, D, B, T, R, P, L$ and must keep the sibling residual-diamond route visible.

Finally apply seed status, Definition 5547.39. The seed reader receives only the retained-premise boundary, so full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, and kernel consistency remain outside the carrier. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationHand`

Theorem 5547.41 (MetaCIC candidate handoff scoped TasteGate route)**.** *Every later* TasteGate *instance for* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *must cite the exact paper-side route:*

1. *the carrier of Definition 5547.2;*
2. *the boundary of Definition 5547.3;*
3. *the obligation surface of Definition 5547.34;*
4. *residual-diamond totality from Definition 5547.36;*
5. *residual-strip local join from Definition 5547.37;*
6. *seed status from Definition 5547.39;*
7. *the sibling confluence handoff of Definition 5547.40.*

This route is the one needed by the GroundCompiler *TasteGate check. It does not supply full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or an unrestricted normalization theorem.*

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateHandoffScopedTast`

Proof. Start with the carrier and boundary. The carrier definition displays $A, K, N, F, C, D, B, T, R, P, L$, while the boundary theorem keeps candidate evidence, frontier data, residual envelope, decidable boundary, and blocked-edge retention visible before any bounded endpoint is consumed.

Next read the obligation surface. It gives the rows a local NameCert surface and prevents the handoff from being treated as a detached MetaCIC confluence theorem.

Use residual-diamond totality and residual-strip local join. These theorems supply only the finite join route inside $A, K, N, F, C, D, B, T, R, P, L$, with the residual envelope and decidable-boundary row still displayed.

Finally cite seed status and the sibling route. Seed status records the retained-premise boundary, and the sibling route keeps the critical-path, normalization-frontier, confluence-audit, residual local-join, and parallel-diamond residual surfaces visible. Thus a GroundCompiler TasteGate check can cite the scoped route above, while full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, and unrestricted normalization remain outside the claim. □

Theorem 5547.42 (MetaCIC candidate-normalization confluence scoped surface)**.** *The scoped* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *surface is exactly the carrier row, residual-diamond row, bounded join-window row, decider-row exhaustion row, finite conversion endpoint row, audit-map realization row, and sibling route row displayed in this chapter. The surface does not claim full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, closed consistency, or kernel consistency.*

Proof. Start with the carrier and obligation surface, Definition 5547.2 and Definition 5547.34. They name the packet rows and keep the candidate evidence, frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming visible.

Next use residual-diamond totality and residual-strip local join, Definition 5547.36 and Definition 5547.37. These provide only bounded local joins and finite conversion endpoints inside the displayed packet.

Then apply decider-row exhaustion and seed status, Definition 5547.22 and Definition 5547.39. They keep the decider and audit-map realization rows as retained premises rather than global conversion or kernel-consistency claims.

Finally cite the sibling route Definition 5547.40. It binds the residual-diamond dependency and the finite handoff route, so the scoped surface is precisely the listed packet and sibling rows. □

Lean4 ✓ `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp.MetacicCandidateNormalizationConf`

Closure status for NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ ·

Theory closure: TheoryGate(NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$, obligation)

Scope closed: Definition 5547.2, Definition 5547.3, Definition 5547.34, Definition 5547.36, Definition 5547.37, and Definition 5547.39 record the seed and obligation surface for candidate normalization, residual join, confluence handoff, finite-window replay, and MetaCIC frontier dependencies.

Formal verification: FormalStatus(NameCert$_{\text{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.MetacicCandidateNormalizationConfluenceHandoffUp`

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, closed consistency, Lean proof obligations, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-gate exactness, confluence handoff discipline, decidable-boundary discipline, frontier non-collapse, Nontrivial, and FieldFaithful instances for BEDC.Derived.MetacicCandidateNorm

Constructive story: A MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet is generated from a MetaCIC audit anchor, candidate evidence, bounded normal endpoint row, MetaCICNormalizationFrontierUp row, residual confluence-envelope row, MetaCICDecidableBoundaryUp row, blocked-edge retention, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 5547.43 (MetaCIC candidate residual-budget discharge frontier)**.** *For every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier*

$$H = (A, K, N, F, C, D, B, T, R, P, L),$$

candidate evidence K, residual envelope C, decidable boundary D, and blocked-edge row B remain visible before the endpoint row N is treated as a residual join by MetaCICParallelDiamondFrontier$^{\uparrow}$*. The discharge frontier is therefore the finite route from the audit anchor A through $K, F, C, D, B, T, R, P, L$ into the parallel-diamond consumer, not full Church–Rosser, unrestricted subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. The rows $A, K, N, F, C, D, B, T, R, P, L$ display the audit anchor, candidate evidence, endpoint row, normalization frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. Together they keep K, F, C, D, B attached to the same accepted packet before any confluence-facing endpoint read can treat N as residual join data.

Finally read the parallel-diamond frontier sibling of chapter 4238. Its consumer may inspect the residual join only after the candidate, residual-envelope, decidable-boundary, and blocked-edge rows have remained visible through T, R, P, L.

The refused global conclusions are absent from the carrier and from this consumer route. Thus the discharge frontier records a candidate-mediated finite handoff rather than a global MetaCIC theorem. □

Theorem 5547.44 (MetaCIC candidate normalization diamond-budget route)**.** *Every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *closed-normal read must first expose candidate-set membership, a bounded local peak, join-budget rows, the sibling confluence route of chapter 6552, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}}$ *row. Thus the closed-normal endpoint is consumed only after the diamond-budget dependency surface is visible, not as an evidence-free consequence.*

Proof. Project the parent carrier of Definition 5547.2. Its accepted read displays candidate evidence, bounded endpoint data, residual confluence routing, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming before any consumer can see a normal endpoint.

Apply the boundary theorem of Definition 5547.3. Candidate-gate exactness exposes the candidate-set membership row before the endpoint row is consumed, while confluence handoff discipline keeps the residual and blocked-edge rows visible.

The sibling route through chapter 6552 supplies the bounded local peak and join-budget rows. Reading that sibling before the closed-normal endpoint preserves the finite diamond budget as a dependency of the handoff, and T, R, P, L transport, replay, source, and name the same route without adding a global confluence or normalization coordinate. □

Theorem 5547.45 (MetaCIC candidate normalization handoff route)**.** *Every accepted* MetacicCandidateNormalizat *packet that exports a confluence handoff must first expose the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source row, the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *frontier row, residual join scope, candidate normalization budget, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *rows. In carrier notation the route reads the displayed packet rows* $A, K, N, F, C, D, B, T, R, P, L$ *before any consumer treats the bounded endpoint row* N *as confluence-facing data.*

The exported handoff is therefore a finite candidate-mediated route through the critical path, parallel-diamond frontier, residual envelope, decidable boundary, blocked-edge budget, transport, replay, provenance, and local naming surface. It is not global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.

Proof. Start with the carrier of Definition 5547.2. The rows $A, K, N, F, C, D, B, T, R, P, L$ display the audit anchor, candidate evidence, bounded endpoint, normalization frontier, residual confluence envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming.

Next read the sibling routes. The $\mathsf{MetaCICCriticalPath}^{\uparrow}$ row supplies the public critical-path source, while the $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ row supplies the retained frontier that keeps residual join scope visible before any endpoint is consumed.

Now apply the existing premise-exposure and local-join routes of Definition 5547.5 and Definition 5547.7. They force candidate evidence, residual envelope data, decidable-boundary data, and blocked-edge data to remain attached to the same handoff packet.

Finally pass through T, R, P, L. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows preserve the finite packet and add no coordinate for the refused global MetaCIC endpoints. Hence the confluence handoff can be exported only as the stated candidate-normalization route. □

Theorem 5547.46 (MetaCIC candidate-normalization handoff cut-height budget route)**.** *Any* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *consumer that reads a cut-height budget or local-diamond endpoint must first expose candidate evidence, the normalization frontier, residual confluence, the decidable-boundary route, and blocked-edge rows in the same finite handoff packet. In particular, the consumer must cite the parallel-diamond frontier of chapter 4238 and the normalization frontier of chapter 892 before any cut-height or local-diamond row is consumed.*

Proof. Project the candidate-normalization confluence handoff carrier from the local NameCert construction. Its public rows expose candidate evidence, bounded normal-endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

The cut-height consumer is downstream of those rows. A local-diamond read needs the residual confluence and blocked-edge rows, while a budget read needs the normalization frontier and decidable-boundary rows before it can decide which bounded endpoint is being measured.

Now read the sibling frontiers. The parallel-diamond frontier supplies the residual diamond and local-join surface, and the normalization frontier supplies the bounded normal-endpoint surface. Since the handoff has no separate cut-height coordinate outside those displayed routes, every accepted cut-height or local-diamond consumer must first display the stated rows. □

The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ dossier-lattice anchor consumes the parallel-diamond frontier route of chapter 4238 and the local-confluence certificate route of chapter 7346 before residual-normalization handoff rows are public. Thus the handoff keeps candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming attached to those sibling routes instead of exporting a standalone confluence endpoint. The candidate-normalization handoff consumes the residual-join and local-confluence routes through chapter 4238 and reads the Real and RegSeqRat-style finite route discipline of chapter 909 only as packet ordering. This dependency records the visible sibling path for residual joins, local diamonds, finite readback rows, transport, replay, provenance, and local naming; it does not select a normalization, quotient terms, identify evaluator equality, or turn the Real-facing ordering discipline into a MetaCIC equality principle. The MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ dossier lattice is consumed by chapter 4238 and chapter 735 only through candidate evidence rows, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention. A downstream confluence reader must therefore keep the handoff as a finite evidence packet rather than reading it as a completed normalization theorem or as an untracked frontier shortcut. The candidate-normalization confluence handoff consumes the parallel-diamond frontier of chapter 4238 together with the critical-path route of chapter 765 before any confluence-facing read is exported. This child route records the dependency lattice: the MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet may pass candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming onward only while those MetaCIC frontier siblings remain visible.

Theorem 5547.47 (MetaCIC candidate handoff forward route to the parallel-diamond frontier)**.** *Every accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet that is consumed as a local confluence handoff must expose the parallel-diamond frontier sibling of chapter 4238 before the handoff is read as downstream MetaCIC evidence. In carrier form the handoff rows*

$$K, N, F, C, D, B, T, R, P, L$$

remain visible while the sibling frontier supplies the retained residual-substitution premise, critical-pair envelope, candidate-join row, residual-frontier row, bounded checker rows, obstruction row, transport, replay, provenance, and local NameCert *rows. The route is a finite dependency route for candidate normalization and confluence readers; it does not assert full Church–Rosser, subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency.*

Proof. Read the carrier of the handoff packet first. The candidate evidence, normal endpoint, frontier, residual confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and naming rows are the only rows available to the consumer.

Next read the sibling frontier at chapter 4238. Its displayed frontier budget keeps the retained residual-substitution premise and obstruction rows visible before any local diamond route is consumed.

The two packets therefore meet only along finite visible rows. Since neither packet contains a row for unrestricted confluence, subject reduction, evaluator equality, quotient equality, residual-substitution discharge, or kernel consistency, the forward route remains a dependency edge rather than a global MetaCIC endpoint. □

The $\mathsf{MetaCICCandidateNormalizationConfluenceHandoff}^{\uparrow}$ handoff consumes the normalization-candidate frontier of chapter 892, the bounded Church–Rosser fragment of chapter 8560, and the confluence audit rows of chapter 735 before exporting its candidate-mediated handoff. The dependency route keeps candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming visible as finite MetaCIC data, not as a detached normalization theorem or a global confluence endpoint.

Theorem 5547.48 (MetaCIC critical-path and parallel-frontier handoff route)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read the MetaCIC critical-path surface of chapter 765 and the parallel-diamond frontier route of chapter 4238. Only after those sibling surfaces are displayed may the handoff be treated as a confluence-facing packet over candidate evidence, bounded endpoint rows, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming. This is a finite MetaCIC route, not subject reduction, full Church–Rosser, unrestricted conversion decidability, quotient syntax equality, or kernel consistency.*

Proof. Project the handoff carrier of Definition 5547.2. Its rows expose audit anchoring, candidate evidence, normal endpoint data, frontier data, residual confluence data, decidable-boundary data, blocked-edge retention, transport, replay, provenance, and local naming.

Apply the boundary theorem Definition 5547.3. It requires a confluence-facing read to keep the candidate gate, residual envelope, decidable boundary, and blocked-edge rows visible before an endpoint row can be consumed.

The sibling route is therefore read in two steps. The MetaCIC critical-path chapter supplies the upstream audit route, and the parallel-diamond frontier supplies the confluence-facing frontier route. The handoff can then pass finite candidate and residual data onward, but no row in

the carrier turns that route into subject reduction, full Church–Rosser, unrestricted conversion decidability, quotient syntax equality, or kernel consistency. □

Theorem 5547.49 (MetaCIC candidate handoff normalization-audit forward link)**.** *Every confluence-facing consumer of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read the* $\mathsf{MetaCICNormalizationAudit}^{\uparrow}$ *packet of chapter 750. In particular, the handoff consumes the audit carrier of Definition 750.1, the candidate evidence row of Definition 750.3, the closed projection route of Definition 750.5, the finite strong-normalization scope of Definition 750.4, the application boundary of Definition 750.6, and the confluence refusal row of Definition 750.9 before candidate normalization is routed to any confluence-facing endpoint.*

Proof. Start with the normalization-audit carrier. The rows $R, C, S, A, N, F, H, T, P, L$ separate candidate evidence, closed projection, finite strong-normalization scope, application boundary, confluence refusal, replay, transport, provenance, and local naming.

Apply the candidate boundary and closed-term projection route. They force any normalization read to expose candidate evidence and checked closed projection before the finite SN inventory can be consumed.

Next apply the finite SN scope and candidate-mediated SN closure theorems. These rows keep the SN export inside the candidate ledger and retain the application boundary, so the handoff cannot convert candidate normalization into an evidence-free endpoint.

Finally use the confluence refusal split. The refusal row N remains visible when the handoff approaches a confluence-facing consumer, excluding global confluence, unrestricted subject reduction, quotient syntax equality, and host evaluator evidence from the forwarded route. □

Theorem 5547.50 (MetaCIC candidate normalization confluence handoff normal-form obligation)**.** *Any normal-form read exported by an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *carrier must first expose the candidate-normalization row, the residual-diamond row, and the sibling confluence route. Concretely, the read must pass through the candidate gate and frontier rows of Definition 5547.3, the premise-exposure and normal-endpoint rows of Definition 5547.5 and Definition 5547.8, the residual-diamond retention row of Definition 5547.6, the bounded candidate-diamond route of Definition 5547.44, and the sibling confluence route of Definition 5547.89 before any normal-form endpoint is available. The obligation is a finite premise-exposure row only; it does not assert host normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency.*

Proof. Project the carrier from Definition 5547.2. A normal-form read can see only the candidate, frontier, residual, decidable, blocked-edge, transport, replay, provenance, and local naming rows displayed in that carrier.

Apply the boundary and premise-exposure theorems. They force the candidate gate and normalization frontier to appear before the normal endpoint is consumed, and they keep the residual envelope, decidable boundary, and blocked-edge rows visible for confluence-facing reads.

Next apply normal-endpoint readback and residual-diamond retention. These rows show that an endpoint can be handed to confluence consumers only while candidate evidence, residual data, decidable-boundary data, and blocked edges remain present in the same packet.

Finally use the diamond-budget dependency and sibling confluence route. The candidate diamond supplies bounded local peak and join-budget rows, and the sibling route exposes the critical-path, normalization-frontier, confluence-audit, and parallel-diamond residual surfaces before the normal-form read is available. Since no carrier coordinate contains a host-level normalization or global confluence theorem, the obligation remains the finite handoff row stated above. □

Theorem 5547.51 (MetaCIC candidate normalization confluence handoff obstruction retention)**.** *Every consumer of the candidate-normalization normal-form row exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the residual-substitution premise, candidate diamond budget dependency, cut-height budget route, and parallel frontier route before treating that row as confluence-facing evidence. The retained route is the finite chain through Definition 5547.44, Definition 5547.46, Definition 5547.73, and the residual-diamond bridge route. It does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Start with the residual budget theorem Definition 5547.57. It forces candidate evidence, residual-diamond retention, decidable-boundary data, blocked-edge rows, bounded local peaks, join-budget rows, and the sibling confluence route to remain visible before a normal-form endpoint is exported.

Apply the candidate diamond budget dependency Definition 5547.44. The normal-form row is therefore read only after bounded local peak and join-budget data have been displayed; it is not an evidence-free confluence endpoint.

Next use the cut-height budget route and the parallel-diamond frontier route, Definition 5547.46 and Definition 5547.73. These routes keep the normalization frontier, residual confluence, decidable boundary, blocked-edge rows, and frontier-facing residual-substitution premise attached to the handoff.

Finally consume the residual-diamond bridge route recorded in the sibling route file. The bridge sends only finite residual-diamond dependency data onward while preserving residual-substitution visibility. Since the carrier has no coordinate for subject reduction, unrestricted Church–Rosser, evaluator equality, quotient term equality, or kernel consistency, every confluence-facing read retains the obstruction instead of erasing it. □

Theorem 5547.52 (MetaCIC candidate normalization normal-form budget)**.** *Every accepted normal-form endpoint exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier must retain the candidate gate, residual-diamond row, cut-height budget, parallel-diamond frontier route, and sibling confluence route as one finite budget surface before it can be used as local confluence evidence. The endpoint is therefore consumed only through the normal-form obligation, obstruction-retention row, diamond-budget route, cut-height budget route, and parallel-diamond frontier route; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, or consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint downstream of the candidate-normalization row, the residual-diamond row, and the sibling confluence route before any consumer can read it.

Apply obstruction retention, Definition 5547.51. The residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route remain attached to the same handoff row rather than being discharged by the endpoint.

Next use the diamond-budget route and cut-height budget route, Definition 5547.44 and Definition 5547.46. These rows expose bounded local peaks, join-budget data, normalization-frontier data, residual confluence, decidable-boundary routing, and blocked-edge retention as finite budget coordinates.

Finally apply the parallel-diamond frontier route, Definition 5547.73, and the sibling confluence route Definition 5547.89. They keep the frontier and sibling residual surfaces visible before the endpoint is treated as local confluence evidence. Since none of the cited rows contains a

global Church–Rosser theorem, subject reduction, quotient syntax equality, evaluator equality, or consistency, the endpoint remains the retained finite budget surface stated above. □

Theorem 5547.53 (MetaCIC candidate normalization normal-form budget)**.** *Every local-confluence consumer of an accepted normal-form endpoint exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must retain the candidate gate, residual-diamond row, residual-substitution premise, cut-height budget, parallel frontier route, sibling confluence route, and local* NameCert *row as one finite budget packet. The endpoint is therefore consumed only through the normal-form obligation, obstruction-retention row, candidate diamond budget route, cut-height budget route, and parallel-diamond frontier route before it can be read as local confluence evidence.*

This retained packet explicitly excludes unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, and kernel consistency.

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint after the candidate gate, residual-diamond row, and sibling confluence route, so the consumer cannot read the endpoint as a detached normal form.

Apply obstruction retention, Definition 5547.51. That row keeps the residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route attached to the endpoint as confluence-facing evidence.

Use the candidate diamond budget route, Definition 5547.44, together with the cut-height budget route Definition 5547.46. They expose the bounded local peak, join-budget, cut-height, normalization frontier, residual confluence, decidable boundary, and blocked-edge rows as finite budget coordinates.

Finally apply the parallel-diamond frontier route, Definition 5547.73, and the sibling confluence route Definition 5547.89. The local NameCert row names exactly that finite packet. Since none of these rows contains unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, or kernel consistency, the consumer receives only the retained normal-form budget. □

Theorem 5547.54 (MetaCIC residual normal-form frontier)**.** *An accepted normal-form endpoint of a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet may be consumed as local confluence evidence only while one finite frontier packet still displays the candidate gate, residual-diamond row, residual-substitution premise, cut-height budget, parallel frontier route, and sibling confluence route. The endpoint is therefore read through the normal-form obligation, obstruction retention, normal-form budget, diamond-budget route, cut-height route, and parallel frontier route before any local-confluence consumer sees it.*

The frontier packet preserves obstruction rows. It does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.

Proof. Start with the normal-form obligation, Definition 5547.50. It places the endpoint after the candidate gate, residual-diamond row, and sibling confluence route, so the endpoint is never presented as an isolated normal form.

Apply obstruction retention and the normal-form budget theorem, Definition 5547.51 and Definition 5547.52. These rows keep the residual-substitution premise, residual-diamond row, cut-height budget, parallel frontier route, and sibling confluence route attached to the same finite handoff packet.

Next use the diamond-budget and cut-height routes, Definition 5547.44 and Definition 5547.46. They expose bounded local peak, join-budget, and cut-height data as visible frontier coordinates rather than as hidden global confluence assumptions.

Finally apply the parallel-diamond frontier route, Definition 5547.73. Together with the sibling confluence route, it leaves the residual and frontier rows visible when the endpoint is consumed as local confluence evidence. Since the carrier has no coordinate for subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency, the endpoint remains a finite obstruction-preserving frontier read. □

Theorem 5547.55 (MetaCIC candidate normalization normal-form obligation route)**.** *Every normal-form endpoint read exported by a* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *carrier must pass through the candidate gate, residual-diamond retention, cut-height budget, parallel-diamond frontier route, sibling confluence route, and local* NameCert *rows before it can be used as confluence-facing evidence. The route retains the residual and frontier evidence attached to the endpoint; it does not export subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50. It places every endpoint downstream of the candidate gate, residual-diamond row, and sibling confluence route before any consumer can read it.

Apply obstruction retention, Definition 5547.51. The retained residual-substitution premise, candidate diamond dependency, cut-height budget, and parallel frontier route remain attached to the endpoint instead of being discharged by the normal-form read.

Finally use the normal-form budget theorem Definition 5547.52. It keeps the cut-height budget, parallel-diamond frontier surface, sibling confluence route, and local naming rows visible before the endpoint is treated as confluence-facing evidence. Since none of the cited rows contains subject reduction, unrestricted Church–Rosser, evaluator equality, quotient syntax equality, global normalization, or kernel consistency, the read exports only the retained finite obligation route. □

Theorem 5547.56 (MetaCIC retained-premise normal-form handoff)**.** *Every strong-normalization-facing consumer of the normal-form endpoint exported by an accepted* MetacicCandidateNormalizationConfluenceHan *packet must retain the candidate gate, the residual-diamond retention row, the decidable-boundary row, and the sibling confluence route. The consumer may read the bounded normal-form endpoint only as a finite handoff through those rows; it does not receive candidate-free normalization, quotient syntax equality, host evaluator soundness, unrestricted subject reduction, global confluence, or closed consistency.*

Proof. Apply the normal-form obligation of Definition 5547.50. It forces a normal-form read through the candidate-normalization row, residual row, and sibling confluence route before the endpoint is available.

Next use residual-diamond retention from Definition 5547.6. That row keeps the residual envelope, decidable boundary, and blocked-edge data visible while the endpoint is treated as a confluence-facing read.

The diamond-budget route of Definition 5547.44 supplies the bounded local peak and join-budget rows. These are the finite rows that a strong-normalization-facing consumer may inspect; they do not discharge the candidate gate or replace it with a global normalization theorem.

Finally apply the sibling confluence route of Definition 5547.89. It keeps the critical-path, normalization-frontier, confluence-audit, and parallel-diamond residual surfaces visible before the handoff reaches a consumer. Hence the endpoint remains a retained-premise normal-form handoff and no candidate-free normalization or quotient syntax equality is exposed. □

Theorem 5547.57 (MetaCIC candidate-normalization residual budget)**.** *Any normal-form endpoint exported by an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet to a strong-normalization-facing consumer must retain, before the endpoint is read, the candidate evidence, residual-diamond retention row, decidable-boundary data, blocked-edge rows, bounded local peak rows, join-budget rows, and the sibling confluence route. The endpoint is therefore a finite residual-budget read, not candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency.*

Proof. Start with the normal-form obligation of Definition 5547.50 and the retained-premise handoff of Definition 5547.56. Together they force candidate evidence and the sibling confluence route to remain visible before a normal-form endpoint is exported.

Use residual-diamond retention from Definition 5547.6. It keeps the residual envelope, decidable-boundary data, and blocked-edge rows attached to the endpoint read rather than allowing a detached normal-form token.

Finally apply the diamond-budget route of Definition 5547.44. Its bounded local peak and join-budget rows are the finite budget a strong-normalization-facing consumer may inspect. Since the cited rows exhaust the exported endpoint surface, none of candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or closed consistency is supplied. □

Theorem 5547.58 (MetaCIC candidate-normalization residual exhaustion)**.** *Every accepted* MetacicCandidateNorm *handoff row whose candidate set, bounded local peak, join-budget row, residual envelope, decidable-boundary row, blocked-edge row, transport, replay, provenance, and local* NameCert *are displayed has an exhaustive finite endpoint surface. It either forwards a retained premise through the candidate and normalization frontier rows, or emits the blocked residual edge as an obstruction. No consumer may erase that obstruction and read global confluence, unrestricted subject reduction, quotient syntax equality, closed consistency, or a candidate-free normal-form theorem.*

Proof. Project the displayed handoff carrier from the parent chapter. The candidate set, frontier rows, residual envelope, decidable boundary, blocked-edge row, transport, replay, provenance, and local naming data are the only rows available to the consumer.

Apply the residual budget theorem Definition 5547.57. It requires the candidate evidence, residual-diamond retention, decidable-boundary data, blocked-edge rows, bounded local peak rows, join-budget rows, and sibling confluence route to remain visible before any endpoint is exported.

Now read the displayed forward-or-blocked row as finite data. In the forward case the retained premise is carried through the candidate and frontier rows. In the blocked case the residual edge remains an obstruction attached to the endpoint read. The carrier's refused endpoints exclude candidate-free normalization, global confluence, unrestricted subject reduction, quotient syntax equality, closed consistency, and any consumer route that erases the obstruction. □

Theorem 5547.59 (MetaCIC candidate-normalization residual-join choice-free handoff)**.** *When an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet exposes finite residual budget exhaustion and a displayed local-confluence certificate route, the residual-join endpoint is read as a finite handoff row. The endpoint is not a selected global normal form: it remains attached to the candidate evidence, normal-form obligation, residual budget, residual exhaustion surface, local confluence certificate route, transport, replay, provenance, and local* NameCert *rows.*

Proof. Start with the normal-form obligation of Definition 5547.50. It keeps candidate evidence and the sibling confluence route visible before any endpoint is public.

Apply the residual budget theorem, Definition 5547.57. The bounded local peak, join-budget, residual-diamond retention, decidable-boundary, and blocked-edge rows remain finite budget data rather than a choice of global normal form.

Now use residual exhaustion, Definition 5547.58. The forward case carries the retained premise through the candidate and frontier rows, while the blocked case keeps the residual edge as an obstruction attached to the endpoint.

Finally read the local confluence certificate route through the sibling confluence surface already cited by the normal-form and residual-budget rows. Since the endpoint is assembled only from those displayed rows and the structural transport, replay, provenance, and naming data, it is a choice-free residual-join handoff. □

Definition 5547.60 (Normalization-confluence bridge carrier)**.** *A normalization-confluence bridge carrier for* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *is a finite* BHist *packet*

$$B = (A, K, N, J, Q, D, O, H, C, P, L)$$

with the following displayed rows:

1. *A is the MetaCIC audit anchor for the closed typed term boundary.*
2. *K is the candidate-normalization evidence row used to expose candidate-mediated strong normalization data.*
3. *N is the bounded normal endpoint row reached from K.*
4. *J is the residual local-join budget row, recording only the bounded join checks available at the confluence frontier.*
5. *Q is the closed checker row that verifies the finite term and type boundary before any endpoint or join row is consumed.*
6. *D is the obstruction row retaining failed residual edges and blocked comparisons.*
7. *O records componentwise* hsame *transport for the audit anchor, candidate evidence, endpoint, join budget, closed checker, obstruction, replay, provenance, and local naming rows.*
8. *H records displayed* Cont *replay through $A, K, N, J, Q, D, O, P, L$.*
9. *C is the* Pkg *provenance over* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *P is the local public-route packet row for the bounded bridge budget.*
11. *L is the local* $\mathsf{NameCert}$ *row.*

The carrier has no coordinate for full confluence, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, host evaluator soundness, or kernel consistency.

Theorem 5547.61 (Normalization-confluence bridge obligations)**.** *Every accepted normalization-confluence bridge carrier exposes candidate-mediated strong-normalization evidence and bounded local confluence rows only through*

$$A, K, N, J, Q, D, O, H, C, P, L.$$

The candidate row K may feed the bounded endpoint row N, and the residual local-join budget J may feed the confluence frontier only after the closed checker Q and obstruction row D remain

visible. The bridge does not prove full confluence, unrestricted subject reduction, evaluator equality, quotient syntax equality, global conversion decidability, host evaluator soundness, or kernel consistency.

Proof. Project the bridge carrier. The only normalization-facing rows are the audit anchor A, candidate evidence K, bounded endpoint N, residual local-join budget J, closed checker Q, and obstruction row D.

Use the candidate-normalization handoff boundary of Definition 5547.3. It keeps candidate evidence and frontier data visible before bounded endpoint or residual rows can be consumed.

Apply residual-diamond retention and decider-boundary routing from Definition 5547.6 and Definition 5547.21. These force J, Q, D to remain finite frontier rows rather than global confluence or subject-reduction evidence.

Finally read the structural rows O, H, C, P, L. They transport, replay, package, route, and name the same bounded bridge budget, so a consumer receives candidate-mediated normalization and local confluence frontier data without extracting any refused global MetaCIC endpoint. □

Theorem 5547.62 (MetaCIC candidate typed-diamond route)**.** *Every typed-diamond consumer of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *bridge carrier must retain candidate evidence, the closed checker, the residual local-join budget, the decidable boundary, and blocked-edge rows before consuming bounded endpoints. The route may pass through residual-diamond and parallel-diamond sibling frontiers, but it refuses full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, and kernel consistency.*

Proof. Project the bridge carrier of Definition 5547.60. Its candidate evidence row K, bounded endpoint row N, residual local-join budget J, closed checker row Q, obstruction row D, and structural rows O, H, C, P, L are separate coordinates.

Invoke the bridge obligations of Definition 5547.61. They force candidate evidence and bounded local confluence rows to remain visible, and they place the closed checker and obstruction row before endpoint or join rows are consumed.

Route the typed-diamond read through residual-diamond retention, Definition 5547.6. This keeps the residual envelope, decidable boundary, and blocked-edge data attached to the endpoint comparison rather than letting N stand alone.

Finally use the parallel-diamond forward link, Definition 5547.1. It sends the consumer through the parallel-diamond frontier as a guarded boundary route. The structural rows then transport, replay, source, and name only this bounded route, so no full Church–Rosser theorem, unrestricted subject-reduction proof, quotient syntax equality, host evaluator soundness, or kernel-consistency endpoint is exported. □

Theorem 5547.63 (MetaCIC candidate handoff downstream route)**.** *Every public read of* MetacicCandidateNormal *is consumed by chapter 735 and chapter 4238 only after candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention have been exposed in the same finite handoff packet. The downstream route also consumes the residual-diamond sibling chapter 5126 and the confluence-audit packet chapter 735 before exporting a bounded endpoint read. The route exports a downstream confluence dependency surface, not an unconditional strong-normalization theorem, full Church–Rosser theorem, unrestricted subject-reduction theorem, global conversion decider, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with the boundary theorem Definition 5547.3. It exposes the candidate row K, frontier row F, residual envelope C, decidable-boundary row D, and blocked-edge row B before any bounded endpoint or comparison row is consumed.

Apply premise exposure from Definition 5547.5. It prevents a confluence-facing consumer from reading the handoff unless the candidate evidence and residual boundary remain visible in the same packet.

Use residual-diamond retention, Definition 5547.6, to route the downstream read through the parallel-diamond frontier. The frontier receives only the retained residual-diamond handoff, not a global confluence theorem.

Finally apply normal-endpoint readback, Definition 5547.8. The confluence audit packet may consume the bounded endpoint only after C, D, B, T, R, P, L remain visible. Thus both downstream chapters read the same candidate-mediated handoff route and no excluded global endpoint is exported. □

Theorem 5547.64 (MetaCIC candidate-mediated confluence route)**.** *Any confluence-facing consumer of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and normal-endpoint readback. Only after those rows remain visible may the consumer pass the bounded route to chapter 735 or chapter 4238. The route does not claim strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with Definition 5547.63. It already routes candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention to the confluence audit and parallel-diamond frontier consumers.

Apply premise exposure from Definition 5547.5. A confluence-facing consumer cannot read the bounded endpoint before the candidate and residual boundary rows remain displayed in the same packet.

Use residual-diamond retention from Definition 5547.6. It keeps the residual envelope and blocked-edge rows on the route that is handed to the parallel-diamond frontier.

Finally apply normal-endpoint readback from Definition 5547.8. The endpoint is therefore bounded by the visible rows C, D, B, T, R, P, L, so the exported route is a candidate-mediated confluence audit surface rather than any of the refused global MetaCIC endpoints. □

Theorem 5547.65 (MetaCIC candidate residual-obstruction retention)**.** *Every confluence-facing endpoint read of a* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must retain candidate evidence, the residual envelope, the decidable-boundary row, the blocked-edge row, and normal-endpoint readback before any bounded route is exported to MetaCIC confluence consumers. The exported route is therefore a retained obstruction surface through the candidate packet, not an unrestricted Church–Rosser theorem, subject-reduction theorem, quotient syntax equality, evaluator soundness claim, or closed-consistency endpoint.*

Proof. Start with the downstream route of Definition 5547.63. It exposes candidate evidence, residual-envelope visibility, decidable-boundary routing, and blocked-edge retention in the same finite packet before either confluence consumer can read the handoff.

Apply premise exposure from Definition 5547.5. A bounded endpoint is not available to a confluence-facing consumer unless the candidate evidence and residual boundary remain visible.

Use residual-diamond retention from Definition 5547.6. This keeps the residual envelope and blocked-edge row attached to the route that reaches the parallel-diamond frontier.

Finally apply normal-endpoint readback from Definition 5547.8 and then the confluence route of Definition 5547.64. The endpoint readback is exported only with the retained rows, so the consumer receives a bounded obstruction route and none of the excluded global MetaCIC conclusions. □

Theorem 5547.66 (MetaCIC candidate normalization orphan route)**.** *Public reads of* MetacicCandidateNormalizati *pass to chapter 735 and chapter 4238 only after candidate evidence, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, and normal-endpoint readback remain displayed in the same handoff packet. The route is a concrete downstream dependency theorem for the confluence lattice; it does not export a global Church–Rosser theorem, strong-normalization theorem, unrestricted subject-reduction theorem, quotient syntax equality, evaluator soundness claim, or closed-consistency endpoint.*

Proof. First apply the boundary theorem Definition 5547.3. It places the candidate row, frontier row, residual envelope, decidable-boundary row, and blocked-edge row upstream of every bounded endpoint read.

Next use premise exposure from Definition 5547.5. This keeps candidate evidence and residual-envelope visibility displayed before either downstream consumer may read the handoff.

Then apply residual-diamond retention from Definition 5547.6. The retained residual envelope and blocked-edge row are exactly the rows required by the parallel-diamond frontier.

Finally use normal-endpoint readback from Definition 5547.8, together with Definition 5547.65. The confluence audit packet and parallel-diamond frontier therefore receive a displayed candidate-mediated dependency route, not any of the refused global MetaCIC conclusions. □

The candidate-normalization confluence handoff reads the parallel-diamond frontier of chapter 4238 through the local candidate-diamond-budget and residual-frontier siblings before any handoff row is made public. It also consumes chapter 765 as the critical-path source that fixes the typed-window and residual-frontier rows before the parallel-diamond frontier is read. This route keeps candidate evidence, residual-envelope visibility, decidable-boundary rows, and blocked-edge retention visible inside the finite MetaCIC packet.

Theorem 5547.67 (MetaCIC candidate normalization handoff obligations)**.** *Every candidate-normalization confluence handoff read factors through* MetaCICParallelDiamondFrontier$^{\uparrow}$*, the candidate-diamond-budget route, and the residual-frontier route siblings before any confluence handoff row is public. The handoff may then expose candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming, but it cannot bypass those frontier siblings.*

Proof. Project the handoff carrier of Definition 5547.2. Its public rows record candidate evidence, bounded endpoint data, residual confluence data, decidable-boundary data, blocked-edge retention, transport, replay, provenance, and local naming.

Use the local candidate-diamond-budget route and residual-frontier route siblings. The former exposes bounded local-peak and join-budget rows before a closed-normal endpoint is consumed, while the latter keeps the parallel-diamond frontier visible before a confluence-facing handoff is exported.

The candidate-normalization side also consumes the normalization audit map of chapter 9383; that map supplies the candidate-mediated SN budget before residual confluence visibility is exposed.

Finally read chapter 4238 as the upstream frontier surface. Since the carrier has no row that makes the handoff public without those sibling routes, every accepted read factors through the displayed frontier dependency before presenting confluence handoff data. □

Theorem 5547.68 (MetaCIC parallel frontier residual-bridge consumption)**.** *Any residual-diamond bridge read of an accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet must first expose the candidate diamond budget, cut-height budget,* MetaCICCriticalPath$^{\uparrow}$ *route,* MetaCICParallelDiamondFrontier

dependency, residual frontier reverse dependency, and seed obligation boundary. The bridge may then consume the finite rows $K, F, C, D, B, T, R, P, L$ only while residual-substitution premises remain visible; it cannot be read as subject reduction, evaluator equality, quotient term equality, global normalization, unrestricted confluence, or kernel consistency.

Proof. Begin with the candidate diamond budget route of Definition 5547.44 and the cut-height budget route of Definition 5547.46. These expose bounded local peaks, join-budget rows, normalization-frontier data, residual confluence data, decidable-boundary data, and blocked-edge retention before an endpoint-facing read is available.

Next read the critical-path and frontier routes. The MetaCIC critical-path handoff of Definition 5547.48 fixes the upstream audit direction, while the parallel-diamond frontier route of Definition 5547.73 keeps the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport, replay, provenance, and local naming visible.

Use the residual frontier reverse dependency and the seed obligation boundary. The reverse dependency keeps the parallel frontier as dependency data rather than as an isolated conclusion, and Definition 5547.85 exposes $K, F, C, D, B, T, R, P, L$ before bounded endpoint consumption.

Finally consume the residual-diamond bridge route. Since the bridge reads only the displayed residual-diamond dependency surface and the carrier has no coordinate for subject reduction, evaluator equality, quotient term equality, global normalization, unrestricted confluence, or kernel consistency, every accepted bridge read retains the stated premises before any confluence endpoint is consumed. □

Theorem 5547.69 (MetaCIC candidate confluence orphan route)**.** *Every accepted* MetacicCandidateNormalizationC *read that exports the normalization/confluence bridge route first consumes the residual-frontier rows of chapter 4238. The consumed rows are the retained residual-substitution premise, critical-pair row, candidate-join row, residual-frontier row, closed-checker row, closed-normal fragment row, bounded-check row, obstruction row, transport, replay, provenance, and local naming surface; the handoff does not export subject reduction, unrestricted confluence, evaluator equality, quotient term equality, residual-substitution discharge, or kernel consistency.*

Proof. Project the handoff carrier of Definition 5547.2. Its candidate, bounded endpoint, residual confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows can be read only as a finite handoff packet.

Next read the upstream frontier through chapter 4238. That chapter displays the retained premise row P, critical-pair row K, candidate-join row J, residual-frontier row R, checker rows S, C, B, obstruction row O, and structural rows H, T, G, N before a confluence-facing consumer can inspect the local-diamond handoff.

Finally combine this frontier read with the existing bridge-budget route of Definition 5547.68. The bridge route consumes exactly the displayed residual-frontier dependency before exporting normalization/confluence data, so the handoff remains inside the finite dependency lattice and none of the refused global endpoints is exposed. □

Parallel-diamond dependency route. The candidate-normalization confluence handoff consumes chapter 4238 for the residual critical-pair windows that remain visible before any confluence handoff row is exposed. The frontier packet supplies the typed local-join boundary and residual-budget ledger, while the handoff packet only routes candidate normalization, normal endpoints, blocked edges, transport, replay, provenance, and local naming through that boundary. Hence the handoff may advertise a confluence-facing row only after the parallel-diamond frontier has fixed

the residual window and excluded full confluence, subject reduction, evaluator equality, quotient syntax equality, and closed consistency.

Theorem 5547.70 (MetaCIC candidate-normalization handoff parallel-diamond dependency)**.** *Every accepted* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *read that consumes the parallel-diamond frontier of chapter 4238 does so only through the displayed critical-pair row, residual-join row, candidate-normalization row, bounded-check row, obstruction row, componentwise transport, replay, provenance, and local* NameCert *rows. The route is a finite sibling dependency on* MetaCICParallelDiamondFrontier$^{\uparrow}$*, not a duplicate confluence theorem, subject-reduction theorem, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel-consistency claim.*

Proof. Read the handoff carrier through its displayed candidate-normalization and bounded endpoint rows. A confluence-facing consumer cannot inspect the handoff before the critical-pair and residual-join rows supplied by chapter 4238 have fixed the frontier boundary.

Next preserve the bounded-check and obstruction rows. These rows keep failed residual-substitution, blocked-edge, and closedness obligations visible as retained data rather than discharging them into a global parallel-diamond result.

Finally transport the same route through the componentwise hsame data, displayed replay, provenance, and local NameCert row. Since no coordinate exports full confluence, subject reduction, evaluator equality, quotient syntax equality, residual-substitution discharge, or kernel consistency, the dependency is exactly the finite sibling route stated above. □

Theorem 5547.71 (MetaCIC candidate-normalization handoff forward link)**.** *Every* MetaCICCandidateNormalizati *consumer that exports a residual join to MetaCIC normalization consumers must first display the parallel-diamond frontier of chapter 4238 or the residual-frontier route carried by that frontier. Only after the candidate evidence, residual-envelope row, decidable-boundary row, blocked-edge ledger, transport row, replay row, provenance row, and local* NameCert *row have been read through that frontier may the confluence handoff be public.*

Proof. Read the handoff carrier of Definition 5547.2. Its confluence-facing rows are candidate evidence, bounded endpoint data, residual envelope, decidable boundary, blocked-edge retention, and the structural transport, replay, provenance, and naming rows.

Use the residual-diamond retention theorem Definition 5547.6. It keeps candidate evidence, residual envelope, decidable boundary, and blocked-edge rows visible before any bounded endpoint is treated as joinable.

The sibling routes in this hub then point the retained residual data to chapter 4238. That frontier is the public place where the residual-frontier route and local parallel-diamond budget are displayed.

Since the carrier has no coordinate for a public confluence handoff detached from those rows, any consumer exporting a residual join must pass through the frontier route before the handoff is public. □

Every candidate-normalization confluence handoff consumer must read the residual diamond and local-join data through chapter 4238 before exporting a confluence-facing handoff. The MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ packet therefore presents bounded endpoint, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows only after the parallel-diamond frontier has supplied the displayed residual-diamond and local-join surface.

Theorem 5547.72 (Parallel-diamond frontier dependency bridge)**.** *Every accepted* MetaCICCandidateNormalizatior *carrier reads the residual join route through chapter 4238 before it exports a normalization-confluence budget to a downstream candidate-normalization consumer.*

Proof. The handoff carrier already exposes candidate-normalization, residual-join, and confluence-budget rows through its sibling route files. The parallel-diamond frontier is the only displayed sibling that licenses the residual join as a diamond-facing dependency rather than as an isolated normalization claim.

Thus the confluence budget is exported only after the residual join has passed through the parallel-diamond frontier route. No kernel-only or local-normalization row can stand in for that horizontal dependency. □

Theorem 5547.73 (Candidate normalization handoff parallel-diamond frontier route)**.** *Any* MetaCICCandidateNor *consumer that exports a local diamond budget must first cite chapter 4238. The export preserves the residual-substitution premise, candidate-join row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local* NameCert *row before the handoff may be read as a frontier-facing diamond route.*

Proof. Read the existing parallel-diamond dependency route for the handoff. It places the candidate-normalization budget on the same finite route as the frontier packet rather than on a global confluence theorem.

The cited frontier chapter supplies the public location for that route. A consumer that exports a local diamond budget therefore reaches the frontier only after preserving the residual-substitution premise, candidate join, bounded check, obstruction, transport, replay, provenance, and local naming rows.

Thus the handoff is a displayed dependency through MetaCICParallelDiamondFrontier$^{\uparrow}$. It does not export unrestricted residual substitution, global Church–Rosser, evaluator equality, quotient term equality, or kernel consistency. □

Theorem 5547.74 (MetaCIC candidate normalization parallel-diamond frontier seed dependency)**.** *Before any* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *read may use the parallel frontier of chapter 4238, it must consume the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, and obstruction row exposed by that sibling. The handoff route then reads its bounded endpoint, residual envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows as downstream data rather than as a replacement for the parallel-diamond frontier seed surface.*

Proof. Read the parallel-diamond frontier chapter first. Its seed surface keeps the residual-substitution premise, critical-pair row, candidate-join row, bounded-check row, obstruction row, and structural NameCert rows visible before any local confluence-facing read is available.

Apply the handoff carrier and candidate diamond budget routes in this chapter. They expose bounded endpoint, residual-envelope, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows only after the upstream frontier dependency has been displayed.

Finally compare the dependency order. Since the candidate-normalization handoff has no coordinate for discharging residual substitution or replacing the frontier obstruction row, every accepted read that uses chapter 4238 must keep those sibling rows ahead of the handoff read. □

Theorem 5547.75 (MetaCIC candidate-normalization orphan parallel-frontier route)**.** *Any confluence-facing read of* MetaCICCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *reaches the* MetaCICParallelDiamondFrontier$^{\uparrow}$

consumer of chapter 4238 through the candidate evidence row, residual-envelope row, decidable-boundary row, and blocked-edge row before bounded endpoint data is public. The route preserves transport, replay, provenance, and local NameCert *rows, and it exports no global confluence theorem, quotient term equality, evaluator equality, unrestricted residual substitution, or kernel consistency claim.*

Proof. Start with the handoff boundary surface. The carrier of Definition 5547.2 and the residual-diamond retention theorem keep candidate evidence, residual envelope, decidable boundary, and blocked-edge retention visible before any endpoint data is consumed.

Next apply the forward-link route of Definition 5547.71. It sends the retained residual data to the parallel-diamond frontier rather than to a detached confluence oracle.

Finally read the frontier consumer at chapter 4238. The transport, replay, provenance, and local naming rows remain attached to the same finite handoff, so the orphan-facing read is a sibling dependency on MetaCICParallelDiamondFrontier$^{\uparrow}$ and not an export of any excluded host theorem. □

Theorem 5547.76 (Candidate normalization retained frontier route)**.** *Every candidate-normalization confluence handoff that consumes a bounded local diamond must first read the retained-premise frontier of chapter 4238. In carrier notation the consumer reaches the handoff rows $K, F, C, D, B, T, R, P, L$ only after the sibling frontier has exposed the retained premise row, candidate-join row, residual-frontier row, bounded-check row, obstruction row, transport row, replay row, provenance row, and local* NameCert *row.*

The route is a finite candidate-normalization dependency. It does not supply residual-substitution discharge, subject reduction, full Church–Rosser, evaluator equality, quotient term equality, global normalization, or kernel consistency.

Proof. Start with the local handoff carrier of Definition 5547.2. Its rows $K, F, C, D, B, T, R, P, L$ expose candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

Next read the sibling frontier. The parallel-diamond frontier chapter supplies the retained-premise and candidate-join route consumed by this handoff, so the local diamond budget cannot be inspected until those frontier rows are visible.

Finally apply the existing parallel-diamond frontier route of Definition 5547.73. It keeps the residual-substitution premise, bounded-check row, obstruction row, transport, replay, provenance, and local naming attached to the candidate-normalization handoff. Therefore the new route is a child dependency through MetaCICParallelDiamondFrontier$^{\uparrow}$, not an independent confluence or kernel endpoint. □

Theorem 5547.77 (MetaCIC candidate normalization confluence handoff public export)**.** *The* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet may be exported publicly only as the finite route through* MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICParallelDiamondFrontier$^{\uparrow}$, *the residual join budget,* MetaCICDecidableBoundary$^{\uparrow}$, *blocked-edge retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. For an accepted handoff carrier $H = (A, K, N, F, C, D, B, T, R, P, L)$, any public read must keep candidate evidence, bounded endpoint data, residual-envelope visibility, decidable-boundary routing, blocked-edge data, transport, replay, provenance, and local naming visible before the endpoint row is consumed.*

The exported route is not global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency.

Proof. Project the carrier of Definition 5547.2. Its public rows are $A, K, N, F, C, D, B, T, R, P, L$, so a public export starts from the audit anchor, candidate evidence, bounded endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name.

Use the boundary and candidate handoff routes. The boundary theorem Definition 5547.3 keeps candidate evidence and normalization-frontier data visible before endpoint consumption, while Definition 5547.45 sends the same packet through $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, residual join scope, and candidate-normalization budget rows.

Next apply the residual budget and route-factorization surfaces, Definition 5547.57 and Definition 5547.83. They force every residual join or parallel-diamond read to retain the bounded budget, decidable boundary, and blocked-edge rows rather than replacing them by a global confluence theorem.

Finally read T, R, P, L. Componentwise transport, displayed replay, package provenance, and local naming preserve the finite route already displayed by the sibling theorems. Since no carrier coordinate stores global strong normalization, full Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or closed consistency, the only public export is the stated finite route. □

The candidate-normalization confluence handoff consumes the parallel-diamond frontier of chapter 4238 through the displayed residual-join, critical-pair, and candidate-diamond rows before any confluence-facing endpoint is read. In carrier terms, the $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ packet may route its candidate evidence, residual envelope, decidable-boundary row, blocked-edge ledger, transport, replay, provenance, and local $\mathsf{NameCert}$ onward only while the sibling frontier still exposes its residual-substitution premise, local critical-pair surface, candidate join, bounded checker rows, and obstruction row. The bridge is therefore a finite residual-diamond dependency route, not subject reduction, unrestricted Church–Rosser, quotient syntax equality, evaluator equality, global conversion decidability, or kernel consistency. The candidate-normalization confluence handoff consumes the residual-frontier and finite candidate-diamond rows before it exposes its local confluence route. In particular, chapter 4238 supplies the parallel frontier surface that the handoff reads as dependency data rather than as an isolated conclusion.

Theorem 5547.78 (MetaCIC candidate-normalization residual-join frontier)**.** *An accepted* $\mathsf{MetacicCandidateNorma}$ *handoff may consume a residual-join frontier only after the candidate set, normalization window, residual-substitution row, join-locality row, and strong-normalization handoff have all been displayed. The frontier read remains a finite route through the candidate-normalization carrier, residual-substitution boundary, local join ledger, bounded normal-form endpoint,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. Its candidate evidence, bounded endpoint, residual envelope, decidable-boundary route, blocked-edge row, transport, replay, provenance, and local naming rows are visible before any frontier read is public.

Read the candidate and window rows first. The candidate set supplies the finite normalization candidates, and the normalization window fixes the bounded endpoint at which a residual join can be inspected.

Next consume the residual-substitution and join-locality rows. The residual-substitution boundary keeps substitutions attached to the displayed residual envelope, while the local join ledger keeps the join inside the candidate window rather than a host equality space.

Finally attach the strong-normalization handoff. It makes the bounded normal endpoint available to the confluence-facing consumer without erasing the candidate, window, substitution, and locality rows. Since the carrier has no coordinate for global Church–Rosser, subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency, the frontier read is exactly the finite route stated above. □

Theorem 5547.79 (MetaCIC candidate normalization residual-join scope)**.** *An accepted* MetacicCandidateNormaliz *candidate-normalization handoff may expose residual joins only after it has read the candidate diamond budget, the cut-height budget, the frontier dependency route, and the residual diamond bridge route. The resulting scope is the finite route through MetaCIC candidate rows, residual windows, candidate joins,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*; it is not an unscoped global confluence claim.*

Proof. Project the parent carrier of Definition 5547.2. Its visible rows already separate candidate evidence, bounded endpoint data, residual confluence routing, decidable-boundary routing, blocked-edge retention, transport, replay, provenance, and local naming.

Read the budget rows next. The candidate diamond route of Definition 5547.44 fixes the bounded peak and join-budget surface, while the cut-height route of Definition 5547.46 fixes the height boundary before a residual endpoint can be consumed.

Finally read the frontier and residual-diamond routes. The residual-frontier dependency and the bridge route of chapter 4238 keep residual windows and candidate joins attached to the displayed BHist, hsame, Cont, Pkg, and NameCert rows. Since no carrier coordinate stores global confluence, quotient syntax equality, evaluator equality, or kernel consistency, the handoff exposes only the scoped residual-join route stated above. □

Theorem 5547.80 (MetaCIC residual normal-form endpoint budget)**.** *Every accepted* MetacicCandidateNormalizat *normal-form endpoint read that is used as residual-join evidence must retain the candidate row, bounded endpoint row, candidate diamond budget, cut-height budget, residual diamond bridge route, and parallel-diamond frontier route before local confluence evidence is consumed. The retained surface is a finite route through the candidate-normalization carrier and its sibling frontier packets; it is not global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Project the residual-join scope of Definition 5547.79. A residual-join consumer enters through the candidate-normalization carrier, so the candidate row and bounded endpoint row remain visible before the endpoint can count as evidence.

Read the budget rows next. The diamond-budget route of Definition 5547.44 supplies the candidate diamond budget, and the cut-height route of Definition 5547.46 supplies the bounded endpoint height discipline.

Now attach the sibling frontier routes. The residual diamond bridge route keeps the residual bridge row in scope, while Definition 5547.73 keeps the parallel-diamond frontier route visible before a local confluence consumer can read the endpoint as join evidence.

Since the named rows are all finite carrier or sibling-route data, consuming local confluence evidence cannot erase them. The handoff has no coordinate for global Church–Rosser, unrestricted subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency, so the endpoint remains a guarded residual normal-form budget rather than an unscoped confluence theorem. □

Theorem 5547.81 (MetaCIC candidate normalization residual-join locality scope)**.** *Any* MetacicCandidateNormali *confluence handoff that exports residual-join data must first display the* MetaCICCriticalPath$^{\uparrow}$ *source*

row, the residual-join locality row, and the MetaCICParallelDiamondFrontier$^{\uparrow}$ *boundary row. A confluence consumer may read the handoff only after these rows have been kept in the finite* BHist, hsame, Cont, Pkg, *and* NameCert *route; it cannot treat candidate-normalization evidence as global Church–Rosser, unrestricted residual-substitution compatibility, subject reduction, quotient syntax equality, evaluator soundness, or kernel consistency.*

Proof. Start from the handoff carrier of Definition 5547.2. Its candidate evidence, bounded endpoint, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming rows remain visible before a confluence-facing consumer reaches any endpoint.

Read the local sibling routes next. The critical-path and frontier route of Definition 5547.48 exposes MetaCICCriticalPath$^{\uparrow}$, while Definition 5547.79 keeps residual joins scoped by candidate rows, residual windows, transport, replay, provenance, and local naming.

Finally use Definition 5547.73 together with chapter 4238. The parallel-diamond frontier supplies the boundary row that a confluence consumer must see before reading the handoff, so the exported route remains a finite residual-join locality handoff rather than any of the excluded global MetaCIC claims. □

Theorem 5547.82 (MetaCIC candidate normalization residual-matrix forward route)**.** *Every* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *handoff consumed by chapter 8134 must first display candidate evidence, the normalization frontier, the residual envelope, the decidable boundary, the blocked edge, componentwise transport, displayed replay, provenance, and local naming rows. The residual-join matrix therefore reads the handoff only after the finite route through the carrier of Definition 5547.2; it does not receive global Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or kernel consistency.*

Proof. Project the candidate-normalization handoff carrier. The candidate evidence row, normalization-frontier row, residual-envelope row, decidable-boundary row, and blocked-edge row are visible before any structural row is replayed.

Next route the matrix consumer through the existing residual scope. The residual-join scope and residual budget exactness results keep the candidate, frontier, residual, boundary, and blocked-edge rows attached to the same finite packet.

Finally read transport, replay, provenance, and local naming. These rows preserve the displayed handoff into chapter 8134, so the matrix consumer receives a guarded child dependency route rather than an unscoped confluence theorem. □

Theorem 5547.83 (MetaCIC candidate normalization confluence handoff route factorization)**.** *Every accepted* MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet consumed by the* MetaCICParallelDiamondF *route or by the normalization-confluence bridge must first expose the residual-budget factorization row of Definition 5547.92. The factorization keeps the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name visible before the handoff reaches chapter 4238 or the residual-substitution boundary of chapter 2181. It does not claim full confluence, subject reduction, evaluator equality, quotient syntax equality, or closed consistency.*

Proof. Project the local residual-budget exactness theorem. It displays candidate evidence, the normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local naming before any residual budget is read.

Compose that projection with the parallel-diamond frontier route. The frontier consumer may read the handoff only after the residual-budget factorization row has fixed the bounded endpoint and kept the residual-substitution premise visible.

The same displayed row is the local premise for the normalization-confluence bridge. Since the packet has no coordinate for full confluence, subject reduction, evaluator equality, quotient syntax equality, or closed consistency, route composition exports only the residual-budget factorization boundary. □

Theorem 5547.84 (MetaCIC candidate normalization confluence handoff critical-path consumption)**.** *Any* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *or* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *consumer of an accepted* MetacicCandidateNor *packet must first read the residual-budget factorization row of Definition 5547.83. The consumed route keeps candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name visible before the handoff is used by the critical path or the parallel-diamond frontier. It does not discharge full confluence, subject reduction, evaluator equality, quotient syntax equality, closed consistency, or residual-substitution.*

Proof. Start with the route factorization theorem. It forces the handoff to expose the residual-budget row before any downstream consumer can use the normalization-confluence boundary.

For a $\mathsf{MetaCICCriticalPath}^{\uparrow}$ consumer, compose that factorization with the critical-path route. The critical path may retain the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name, but it cannot skip the residual-budget boundary.

For a $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ consumer, compose the same factorization with the parallel-diamond frontier route. The frontier read receives the handoff only after the residual-budget row has fixed the bounded endpoint and kept the residual-substitution premise visible.

The packet contains no carrier row for full confluence, subject reduction, evaluator equality, quotient syntax equality, closed consistency, or residual-substitution discharge. Therefore critical-path and frontier consumption preserve exactly the retained-premise boundary supplied by the handoff. □

Theorem 5547.85 (MetaCIC candidate handoff seed obligations)**.** *Every seed-obligation read of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet exposes candidate evidence* K*, normalization frontier* F*, residual envelope* C*, decidable boundary* D*, blocked edge* B*, componentwise transport* T*, displayed replay* R*, provenance* P*, and local* $\mathsf{NameCert}$ *row* L *before bounded endpoint consumption. The read remains a finite handoff boundary and does not assert unconditional strong normalization, full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Start with the handoff boundary of Definition 5547.3. It exposes candidate evidence and frontier data before endpoint consumption, keeps the residual and blocked-edge rows visible, and routes bounded comparison through the decidable-boundary row.

Use seed closure and premise exposure from Definition 5547.4 and Definition 5547.5. These theorems force K, F, C, D, B to remain displayed before a confluence-facing or decidability-facing consumer reaches N.

Finally apply decider-row exhaustion from Definition 5547.22. The structural rows T, R, P, L preserve the same finite packet, so the bounded endpoint is consumed only after the seed-obligation rows have been exposed. □

Theorem 5547.86 (MetaCIC candidate handoff induction)**.** *Induction over the displayed row order*

$$K, F, C, D, B, T, R, P, L$$

preserves candidate evidence, residual-envelope visibility, and decidable-boundary exposure before any endpoint row N is consumed. Hence the handoff remains finite-row evidence for the accepted MetacicCandidateNormalizationConfluenceHandoff$^{\uparrow}$ *packet and exports no unconditional strong normalization, Church–Rosser theorem, subject reduction theorem, quotient syntax equality, evaluator soundness theorem, or closed consistency principle.*

Proof. Project the accepted carrier through the handoff boundary of Definition 5547.3. The row order starts with candidate evidence K and normalization frontier F, so the induction base already exposes the candidate surface before any residual or endpoint-facing read.

For the residual step, use premise exposure from Definition 5547.5. Adding C, D, B keeps the residual envelope, decidable boundary, and blocked edge as displayed rows rather than hidden assumptions, and decider-row exhaustion from Definition 5547.22 keeps D finite before endpoint consumption.

For the structural step, apply the seed obligations of Definition 5547.85. The rows T, R, P, L only transport, replay, package, and locally name the same displayed packet, so they preserve the candidate evidence and the residual-envelope boundary already exposed by the earlier rows.

Thus every prefix of $K, F, C, D, B, T, R, P, L$ keeps the relevant evidence visible before N. Since the endpoint row is not consumed during the induction, none of the excluded global metatheorems or host evaluator conclusions is exported. □

Theorem 5547.87 (MetaCIC candidate residual handoff scope)**.** *Every accepted* MetacicCandidateNormalizationCo *packet routes residual-envelope data C and blocked-edge data B through candidate evidence K, normalization frontier F, decidable boundary D, componentwise transport T, displayed replay R, provenance P, and local* NameCert *row L before any bounded endpoint consumption. The route remains a residual handoff scope and does not assert full Church–Rosser, unrestricted subject reduction, global conversion decidability, quotient syntax equality, host evaluator soundness, or closed consistency.*

Proof. Begin with the seed obligations of Definition 5547.85. They expose $K, F, C, D, B, T, R, P, L$ before endpoint consumption, so residual-envelope and blocked-edge data are visible rows rather than hidden premises.

Apply the boundary theorem Definition 5547.3 and the premise-exposure theorem Definition 5547.5. These force the candidate evidence and normalization frontier to precede every confluence-facing or decidability-facing handoff read.

Finally use decider-row exhaustion from Definition 5547.22. The decidable boundary is consumed as a finite row, while T, R, P, L only transport, replay, package, and locally name the same packet. Hence no endpoint consumer can replace the residual handoff by any excluded global metatheorem or host-evaluator claim. □

Theorem 5547.88 (MetaCIC candidate handoff residual frontier scope)**.** *Every accepted* MetacicCandidateNormaliz *packet routes residual-envelope data C and blocked-edge data B through the normalization frontier F and decidable boundary D before any endpoint-facing consumer reads the packet. Hence the handoff is a scoped frontier certificate: it cannot be read as unconditional strong normalization, unrestricted confluence, global conversion decidability, quotient syntax equality, or host evaluator soundness.*

Proof. Start from the seed-obligation exposure of Definition 5547.85. The accepted packet displays K, F, C, D, B and the structural rows before bounded endpoint consumption, so the residual envelope and blocked edge are not hidden premises.

Apply the boundary theorem Definition 5547.3 together with premise exposure from Definition 5547.5. These force candidate evidence to pass through the normalization frontier before a confluence-facing read can consume the endpoint.

Finally use decider-row exhaustion from Definition 5547.22. Since the decidable boundary row is exhausted before endpoint consumption, no consumer can replace the displayed residual frontier by global conversion decidability, quotient syntax equality, unrestricted confluence, unconditional strong normalization, or host evaluator soundness. □

Theorem 5547.89 (MetaCIC candidate normalization sibling confluence route)**.** *Every confluence-facing read of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first expose the sibling route through chapter 765, chapter 892, chapter 735, chapter 5970, and chapter 5126. The residual local-join determinacy packet supplies the finite residual-join ledger consumed before the candidate endpoint can be treated as confluence-facing evidence. The handoff may then export candidate-mediated endpoint, residual-envelope, decidable-boundary, and blocked-edge rows to a confluence consumer, but it cannot present a standalone confluence theorem before those sibling chapters have been displayed.*

Proof. Project the handoff carrier from Definition 5547.2. Its public confluence route stores audit, candidate, frontier, confluence, decidable-boundary, blocked-edge, transport, replay, provenance, and local naming rows rather than a detached confluence theorem.

Apply the boundary and consumer-boundary theorems of this chapter. They keep the candidate evidence, normalization frontier, residual envelope, decidable boundary, and blocked edge visible before any endpoint or join row is consumed.

The sibling chapters named in the statement supply the critical-path, normalization-frontier, confluence-audit, residual local-join, and residual-diamond obstruction surfaces for those rows. The local-join determinacy route contributes the finite join ledger that the candidate endpoint must consume before it is exported as confluence-facing evidence. Residual obstruction rows therefore route through the residual-join and parallel-diamond siblings rather than through a standalone confluence theorem. Since no carrier coordinate bypasses them, every confluence-facing handoff factors through the displayed sibling route. □

Theorem 5547.90 (MetaCIC candidate normalization local-confluence route)**.** *Any local-confluence consumer of an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet must first read the candidate-normalization carrier rows, the candidate diamond-budget dependency of chapter 6552, the residual bridge route inside the handoff packet, and then the local-confluence frontier of chapter 7346. The resulting route is a finite confluence-facing dependency chain; it is not a standalone local-confluence theorem, unrestricted Church–Rosser, subject reduction, quotient syntax equality, evaluator equality, global conversion decidability, or kernel consistency.*

Proof. Project the carrier of Definition 5547.2. A local-confluence read starts from the same audit anchor, candidate evidence, bounded endpoint, normalization frontier, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local naming rows as every accepted handoff read.

Apply the candidate diamond-budget dependency route. It forces the consumer to keep the bounded local peak and join-budget rows visible before any endpoint row is treated as confluence-facing evidence.

Next read the residual bridge route inside this hub. The residual envelope, decidable-boundary row, and blocked-edge ledger remain displayed, so the handoff cannot erase residual-substitution or critical-pair obligations on the way to a local-confluence consumer.

Finally pass to the local-confluence frontier of chapter 7346. That sibling may consume the exposed candidate and residual rows as a frontier premise, but it receives no coordinate for global confluence, subject reduction, quotient syntax equality, evaluator equality, conversion decidability, or kernel consistency. □

Theorem 5547.91 (MetaCIC handoff local confluence certificate route)**.** *Every* MetaCICCandidateNormalizationCo *read exported to a normalization or confluence consumer must pass through the local-confluence certificate sibling chapter 7346, the parallel-diamond frontier sibling chapter 4238, the residual-join route chapter 7378, and the candidate-normalization budget rows of chapter 6552 before the handoff is public. The route is therefore*

$$\mathsf{MetaCICCandidateNormalizationConfluenceHandoff}^{\uparrow} \longmapsto (\mathsf{local\ confluence},\ \mathsf{MetaCICParallelDiamondFrontier}^{\uparrow},\ \mathsf{residu}$$

No consumer may read the handoff as a direct ambient confluence theorem, a hidden critical-pair normalizer, or a residual-join certificate with the local confluence and budget rows suppressed.

Proof. Read the local confluence sibling chapter 7346 first. It provides the certificate row that turns candidate overlap data into a local joinability check, so a confluence-facing consumer cannot start from the handoff alone.

Next read the parallel-diamond frontier sibling chapter 4238 and the residual-join route chapter 7378. These siblings keep the frontier and residual rows visible before a join is exported, and they prevent the candidate packet from becoming an unscoped global diamond assertion.

Finally read the candidate-normalization budget rows of chapter 6552. They bound the normalization side of the handoff before the packet is exported, so the public route contains the local confluence certificate, frontier, residual join, budget, replay, transport, provenance, and local naming surfaces together. □

Theorem 5547.92 (MetaCIC candidate normalization confluence handoff residual budget exactness)**.** *The* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet exposes residual-budget factorization only after candidate evidence K, normal endpoint N, frontier row F, residual confluence row C, decidable boundary D, blocked edge B, transport T, replay R, provenance P, and local name L have all been displayed. This finite route keeps the bounded residual budget tied to the carrier rows of the handoff packet and to the sibling confluence surfaces of chapter 765, chapter 4238, and chapter 5126. The row asserts the local residual-budget exactness needed by the handoff; it does not claim full Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or closed consistency.*

Proof. Project the candidate-normalization handoff packet. The finite route already displays candidate evidence, the bounded endpoint, the frontier row, the residual confluence row, the decidable boundary, the blocked edge, the transport row, the replay row, the provenance row, and the local name row before any residual budget is read.

Apply the sibling surfaces named in the statement. The critical-path row supplies the candidate anchor, the parallel-diamond frontier supplies the residual confluence boundary, and the residual-diamond sibling keeps the bounded budget local. Since the packet has no coordinate for full Church–Rosser, unrestricted subject reduction, evaluator soundness, quotient syntax equality, or closed consistency, the theorem records only the exact residual budget surface. □

Theorem 5547.93 (MetaCIC candidate normalization residual budget factorization)**.** *Any accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *residual budget factors through candidate evidence K, normal endpoint N, frontier row F, residual confluence row C, decidable boundary*

D, blocked edge B, transport T, replay R, provenance P, and local name L before any downstream residual join consumes it. The factorization routes through chapter 765, chapter 4238, and chapter 5126; it exports no residual join that bypasses the candidate evidence, normal endpoint, frontier row, decidable boundary, blocked-edge ledger, structural transport, replay, provenance, or local naming.

Proof. Project the displayed handoff packet. Its residual-budget row can be read only after the candidate evidence, normal endpoint, frontier row, residual confluence row, decidable boundary, blocked edge, transport, replay, provenance, and local name have all been exposed.

Use Definition 5547.92 for the exactness surface. The same displayed rows already bind the budget to the carrier rather than to a detached residual-join theorem.

Finally read the sibling routes. $\mathsf{MetaCICCriticalPath}^{\uparrow}$ supplies the candidate-critical anchor, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ supplies the frontier boundary, and $\mathsf{ParallelDiamondResidual}^{\uparrow}$ supplies the bounded residual-confluence row. Since the packet contains no coordinate outside those rows, every downstream residual join consumes the stated factorization first. □

Theorem 5547.94 (MetaCIC candidate normalization confluence handoff status)**.** *For an accepted* $\mathsf{MetacicCandidateNormalizationConfluenceHandoff}^{\uparrow}$ *packet whose displayed rows include a candidate-normalization row K, a residual-join row C, a finite cut-height budget B, and a dependency row F for the parallel-diamond frontier, every confluence handoff factors through the residual-budget window before it can be consumed by the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *route. The available handoff surface is exactly the visible route through candidate evidence, bounded normal endpoint, residual join, cut-height budget, blocked-edge ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming.*

This status row may cite the candidate-normalization handoff route of chapter 5547, the residual join scope of chapter 5126, the parallel-diamond dependency route of chapter 4238, and the local confluence certificate route of chapter 765. It exports no evidence-free normalization theorem, abstract source-equivalence theorem, classifier-transport theorem, full Church–Rosser theorem, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency claim.

Proof. Project the accepted handoff packet to its displayed rows. The candidate-normalization row K, residual-join row C, finite cut-height budget B, and frontier dependency F are all visible before the confluence-facing consumer can read the bounded endpoint or any residual join.

Apply the residual-budget exactness theorem and the local factorization theorem above. They force the confluence handoff through the same residual-budget window and keep the blocked-edge ledger, transport, replay, provenance, and local name rows attached to the read.

Now route the result to the sibling frontier. The parallel-diamond frontier consumes the handoff only after the candidate-normalization route, residual join scope, and local confluence certificate have displayed the same finite rows. Since the carrier has no coordinate for evidence-free normalization, abstract source equivalence, classifier transport, or global confluence, the handoff cannot bypass the residual-budget window. □

Theorem 5547.95 (MetaCIC handoff parallel-diamond sibling dependency)**.** *Every accepted* $\mathsf{MetacicCandidateNor}$ *carrier that exports candidate-normalization confluence data must consume the finite parallel-diamond frontier rows of chapter 4238 together with the candidate-normalization router surface of chapter 5210. The handoff therefore reads residual-premise retention, local critical-pair routing, candidate evidence, bounded endpoint routing, obstruction retention, transport, replay, provenance, and local naming before any confluence-facing consumer can use it.*

Proof. Project the handoff carrier through Definition 5547.2. Its candidate row, frontier row, residual envelope, decidable boundary, blocked edge, transport, replay, provenance, and local name remain visible before the bounded endpoint row is exported.

Read the parallel-diamond frontier next. The frontier of chapter 4238 supplies the retained residual premise, local critical-pair, local-join, obstruction, and structural rows that any confluence-facing use must preserve.

Finally route candidate evidence through chapter 5210. The router keeps the candidate evidence and normalization readback inside a bounded certificate route, so the handoff cannot be read as premise-free confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or closed consistency. □

Chapter 5548

A Concrete Naming Certificate for CantorIntersectionModulus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CantorIntersectionModulus$^{\uparrow}$ packet records Cantor intersection as explicit finite modulus data rather than as ambient compactness. It sits after the located nested-interval surface of chapter 1713, the bisection route of chapter 1711, the dyadic arithmetic core of chapter 291, the finite request interface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the finite prefix bar of chapter 5863. The chapter names the finite data needed to turn a retained nested branch into a regular Cauchy real name; it does not invoke open-cover compactness or a selected infinite branch.

Definition 5548.1 (Cantor intersection modulus carrier)**.** *A* CantorIntersectionModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *L is the located closed interval row, recording the endpoint order, locatedness tests, and retained closed-interval inclusion data.*
2. *B is the retained nested bisection branch ledger, exposing only the finite branch decisions already displayed by* LocatedIntervalBisection$^{\uparrow}$ *and* LocatedRealNestedIntervals$^{\uparrow}$*.*
3. *D is the dyadic diameter schedule from* DyadicRatCore$^{\uparrow}$*, recording the depth-to-radius budget and the finite halving estimates used at each retained stage.*
4. *S is the* StreamName$^{\uparrow}$ *finite-window row, listing the precision requests and interval stages inspected by the downstream real-name consumer.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected finite windows and their dyadic endpoint approximants.*
6. *E is the terminal* Real$^{\uparrow}$ *seal reached only after the interval, bisection, dyadic schedule, stream window, and regular rational readback rows have been displayed.*
7. *F is the finite fan-depth and modulus output row. It records the finite prefix depth at which the diameter schedule answers the requested precision and the bar-window handoff used by* FiniteFanTheorem$^{\uparrow}$*.*
8. *H records componentwise* hsame *transport for the interval, branch, diameter, window, readback, real seal, fan-depth, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an interval request through L, B, D, S, R, E, F, H and into the local certificate.*

10. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionModulus}^{\uparrow}$, $\mathsf{LocatedRealNestedIntervals}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected infinite branch, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorIntersectionModulusUp.CantorIntersectionModulusTasteGate single carrier alignment`

Theorem 5548.2 (Cantor intersection modulus NameCert obligations)**.** *For every accepted* CantorIntersectionModu *carrier*

$$\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the located closed interval row, retained nested bisection branch ledger, dyadic diameter schedule,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, finite fan-depth and modulus output, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read of the packet must expose L, B, D before the stream window S, must expose S, R before the real seal E, and must expose the fan-depth row F before exporting a precision modulus.*

Thus a downstream completion or real-name consumer receives a finite modulus route from nested interval data into regular Cauchy real names. The packet exports no open-cover compactness theorem, selected infinite branch, countable-choice principle, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5548.1. The accepted packet displays exactly the located closed interval row, retained branch ledger, dyadic diameter schedule, finite stream windows, regular rational readback, terminal real seal, finite fan-depth output, transport, replay, provenance, and local naming rows.

The route order is forced by the displayed carrier. The nested interval consumer first reads L and B as finite interval and branch data, then reads D as the dyadic diameter schedule that bounds the retained interval at each displayed depth.

The real-name handoff is finite. The row S lists the precision windows, R reads back the corresponding regular rational approximants, and E seals the resulting $\mathsf{Real}^{\uparrow}$ row only after those windows and readbacks are already present.

Finally the row F records the finite fan-depth and modulus output used by the handoff to chapter 5863. Since the carrier contains no coordinate for open covers, selected infinite branches, countable choice, quotient real equality, or ambient completeness, H, C, P, N can transport, replay, source, and name only the displayed finite route. □

Theorem 5548.3 (Finite fan interval handoff)**.** *Every* $\mathsf{CantorIntersectionModulus}^{\uparrow}$ *consumer that asks for a finite precision window factors through the interval-to-fan route*

$$L, B, D \longrightarrow S, R, E \longrightarrow F, H, C, P, N.$$

The finite fan-depth row F may use the prefix-bar discipline of chapter 5863 only after the located interval, retained bisection branch, and dyadic diameter rows have fixed the inspected depth. Ordinary regular real readback still passes through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows; the fan row supplies only the finite window bound.*

Proof. Start with an accepted carrier $\mathcal{M} = (L, B, D, S, R, E, F, H, C, P, N)$. The first three rows determine the displayed nested interval data: L gives the located closed interval, B gives the retained finite bisection branch, and D gives the dyadic diameter schedule.

The stream and real-name rows then read only the windows named by that displayed interval data. The rows S and R expose the requested finite precision windows and regular rational readback, and E records the terminal real seal reached from those rows.

It remains to identify the fan use. The row F is the only carrier coordinate that records a finite fan-depth or modulus output, so any consumer asking for a prefix-bar bound must pass through F after L, B, D, S, R, E. The transport, replay, provenance, and local naming rows H, C, P, N then export the same finite route and cannot introduce open-cover compactness, a selected infinite branch, countable choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every $\mathsf{CantorIntersectionModulus}^{\uparrow}$ precision-window read factors through one located closed interval row, one retained nested bisection branch ledger, one $\mathsf{DyadicRatCore}^{\uparrow}$ diameter schedule, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one terminal $\mathsf{Real}^{\uparrow}$ seal, one finite $\mathsf{FiniteFanTheorem}^{\uparrow}$ depth row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: located-real-nested-intervals, located-interval-bisection, dyadicratcore, streamname, regseqrat, real, finite-fan-theorem: $\mathsf{CantorIntersectionModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries nested interval data, dyadic diameter scheduling, real-name readback, and finite fan-depth output as one modulus packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5548.1, Definition 5548.2, and Definition 5548.3 seed the finite Cantor-intersection modulus route through located interval rows, retained bisection ledgers, dyadic diameter schedules, finite stream windows, regular rational readbacks, terminal real seals, and finite fan-depth rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CantorIntersectionModulusUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, countable choice, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-depth exactness, interval-window coverage, componentwise transport, replay, provenance, local naming, and non-escape for BEDC.Derived.CantorIntersectionModulusUp.

Constructive story: A $\mathsf{CantorIntersectionModulus}^{\uparrow}$ packet is generated from located closed interval rows, retained nested bisection branch ledgers, dyadic diameter schedules, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, terminal $\mathsf{Real}^{\uparrow}$ seals, finite $\mathsf{FiniteFanTheorem}^{\uparrow}$ depth outputs, componentwise transport, displayed replay, provenance, and a local NameCert row.

Chapter 5549

A Concrete Naming Certificate for $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, the $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet records the finite NameCert ledger that assembles compact finite-net data, pointwise continuity rows, metric distance observations, rational lower-bound folds, and the final uniform-modulus output. It sits over the compact carrier of chapter 60, the compact metric source of chapter 129, the continuous and continuous-map rows of chapter 59 and chapter 130, the metric distance surface of chapter 57, the rational ledger of chapter 37, the total-order selector route of chapter 54, the finite-net fold of chapter 619, and the $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names the full compactness-implies-uniform-continuity ledger as finite BEDC data; it is not an open-cover theorem, a host minimum, a selected subcover, or an ambient choice principle.

Definition 5549.1 (Compact uniform continuity ledger carrier)**.** *A* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, M, R, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, including the compact source, the finite* $\mathsf{ProbeBundle}$ *of centers, and the located coverage witnesses read from the compact metric packet.*
2. *F is the pointwise modulus row for the continuous map, carrying the graph reads, centerwise source radii, and target-tolerance certificates over the same compact source.*
3. *M is the metric distance row, recording the source distances from an input point to a displayed center, from a second input point to that center, and the target distances used by the triangle estimate.*
4. *R is the finite* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, read through the* $\mathsf{TotalOrder}^{\uparrow}$ *selector route over the bundled positive source radii.*
5. *T is the finite triangle and tolerance ledger tying the lower-bound fold to the source and target metric comparisons.*
6. *U is the output* $\mathsf{UniformModulus}^{\uparrow}$ *row, exposing the tolerance-only precision history obtained from K, F, M, R, T.*

7. *H records componentwise* hsame *transport for compact finite nets, pointwise modulus rows, metric distance observations, rational folds, triangle ledgers, output rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a compact-uniform-continuity tolerance request through $K, F, M, R, T, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}$ *row.*

The classifier compares accepted ledgers only through these coordinates. It contains no coordinate for an arbitrary open cover, selected subcover, host minimum over points, quotient function equality, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformContinuityLedgerUp.CompactUniformContinuityLedgerTasteGate`
`single carrier alignment`

Theorem 5549.2 (Compact uniform continuity ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *carrier $L = (K, F, M, R, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformContinui}}$ *surface consists exactly of carrier admission for the compact finite-net row, pointwise modulus row, metric distance row, finite rational lower-bound fold, triangle and tolerance ledger, output uniform-modulus row, componentwise transport, displayed continuation replay, package provenance, and local naming; source compatibility between K and F; metric-read compatibility between M and the compact source; fold exactness through R and the* $\mathsf{TotalOrder}^{\uparrow}$ *selector route; triangle exactness through T; uniform-modulus handoff through U; and nonescape from arbitrary open covers, selected subcovers, host minima, quotient function equality, countable choice, and ambient real completeness.*

Proof. Project the accepted carrier of Definition 5549.1. The visible rows are exactly the compact finite-net data, pointwise modulus data, metric distance observations, rational fold, triangle ledger, output row, and structural rows.

The source lock is read before any modulus row is consumed. The compact row K and pointwise row F name the same compact metric source, so the finite centers and local radii cannot move to another carrier.

The rational fold is the row R. It is computed over the displayed bundled radii through the $\mathsf{TotalOrder}^{\uparrow}$ selector route, and the metric row M supplies only the distances later used by the source and target triangle estimates.

The triangle and tolerance row T binds these reads into the output row U. Componentwise transport, continuation replay, package provenance, and local naming then preserve the same ledger, so no excluded open-cover, choice, quotient, host-minimum, or completeness coordinate is available to a consumer. □

Theorem 5549.3 (Compact uniform continuity ledger finite-fold route)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{TotalOrder}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

More explicitly, the consumer must first read compact finite-net centers and pointwise modulus rows, then the metric distance observations, then the finite $\mathsf{Rat}^{\uparrow}$ *lower-bound fold selected by the*

$\mathsf{TotalOrder}^{\uparrow}$ *route, then the triangle and tolerance ledger, and only then the output uniform-modulus row. No compact-uniform-continuity read can bypass this finite fold by using an arbitrary open cover, selected subcover, host minimum, or choice schedule.*

Proof. Start with the compact finite-net row K and the pointwise modulus row F. They expose the bundled centers, local radii, graph reads, and source compatibility required by the compact-uniform-continuity request.

Next read the metric row M and the rational fold row R. The metric row supplies the source and target distances used by the later estimate, while R folds the displayed positive radius family through the $\mathsf{TotalOrder}^{\uparrow}$ selector route. The selected precision is therefore a finite history, not an external minimum.

Finally read T and U. The triangle and tolerance ledger transports the selected lower bound into the two source estimates and the target triangle estimate, and the $\mathsf{UniformModulus}^{\uparrow}$ row receives only that tolerance-only precision history.

The rows H, C, P, N keep the route stable under transport, replay, provenance, and naming. Since the carrier has no open-cover, selected-subcover, host-minimum, or choice coordinate, every compact-uniform-continuity consumer factors through the displayed finite fold. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ compact-uniform-continuity read must expose a compact finite-net row, pointwise continuous-map modulus rows, metric distance observations, a finite $\mathsf{Rat}^{\uparrow}$ lower-bound fold, a $\mathsf{TotalOrder}^{\uparrow}$ selector route, a triangle and tolerance ledger, a terminal $\mathsf{UniformModulus}^{\uparrow}$ row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compact, compactmetric, continuous, continuousmap, metric, rat, totalorder, finitenetminimumfold, uniformmodulus: $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ is not bijective to any listed sibling because it binds the whole compactness-implies-uniform-continuity route from finite compact coverage through pointwise radii, metric observations, rational folding, triangle estimates, and the terminal UniformModulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5549.1, Definition 5549.2, and Definition 5549.3 seed the finite NameCert ledger for compactness-implies-uniform-continuity over compact finite nets, pointwise modulus rows, metric distance rows, RatUp folds, TotalOrder selector routes, triangle ledgers, UniformModulus handoff, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformContinuityLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, host minima, quotient function equality, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, metric-read compatibility, rational fold exactness, triangle exactness, uniform-modulus handoff, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.CompactUniformContinuityLedgerUp.

Constructive story: A $\mathsf{CompactUniformContinuityLedger}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMap pointwise modulus rows, Metric distance observations, RatUp lower-bound folds, TotalOrder selector routes, triangle and tolerance ledgers, UniformModulus output rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5550

A Concrete Naming Certificate for $\mathsf{FrechetMetricEmbedding}^{\uparrow}$

Following Fréchet and Kuratowski, the $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ packet records a metric space by the finite distance observations that send a point to its displayed distances from a finite probe family. It sits over the metric source of chapter 57, the real-distance rows of chapter 38, the distance-function route of chapter 7569, and the embedding interfaces of chapter 439 and chapter 1652. The packet names only displayed finite reads; it does not import host function extensionality, quotient equality of maps, or an ambient function-space carrier.

Definition 5550.1 (Fréchet metric embedding carrier)**.** *A* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Q, D, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the point row whose accepted element is the source point being embedded.*
2. *Q is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying the source classifier and source-distance interface.*
3. *D is the family of* $\mathsf{Real}^{\uparrow}$*-valued distance-read rows $d(X, q)$ for displayed probes q.*
4. *A is the finite probe-family and classifier row: two packets agree exactly when the displayed* $\mathsf{Real}^{\uparrow}$ *distance reads agree componentwise on the same probe family.*
5. *H records componentwise* hsame *transport for the point row, metric-source row, distance-read rows, probe-family classifier, replay row, provenance row, and local naming row.*
6. *C records displayed* Cont *replay from source metric reads through the probe-family distance reads and then through the classifier-by-displayed-distance-agreement.*
7. *P is the* Pkg *provenance over* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{KuratowskiEmbedding}^{\uparrow}$*,* $\mathsf{IsometricEmbedding}^{\uparrow}$*,* $\mathsf{UniformEmbedding}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host function object, quotient equality of distance functions, selected completion representatives, or an ambient real-valued function space.

Lean4 ✓ `BEDC.Derived.FrechetMetricEmbeddingUp.FrechetMetricEmbeddingTasteGate single carrier alignment`

Theorem 5550.2 (Fréchet metric embedding NameCert obligations)**.** *For every accepted* $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ *carrier* $F = (X, Q, D, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$ *surface consists exactly of the finite rows displayed in Definition 5550.1. Carrier admission reads the point row, metric-source row,* $\mathsf{Real}^{\uparrow}$ *distance-read rows, finite probe family, classifier-by-displayed-distance-agreement,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming row. Classifier stability compares accepted packets only by the displayed* $\mathsf{Real}^{\uparrow}$ *distance reads over the shared finite probe family, with transport and replay preserving that same route.*

Proof. Project the accepted carrier $F = (X, Q, D, A, H, C, P, N)$. The point row X and metric-source row Q are the only source data exposed before distance reads are requested.

The row D then records the finite $\mathsf{Real}^{\uparrow}$ distance observations against the probe family named by A. Since A classifies by displayed distance agreement, two accepted packets can be compared only by componentwise equality of these finite reads under the same probe family.

Finally read H, C, P, N. Transport, replay, provenance, and local naming preserve the point-to-distance route without adding host function extensionality, quotient equality of maps, selected representatives, or an ambient function-space carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 5550.1 and Definition 5550.2 seed the finite distance-function embedding route over MetricSpaceUp, RealUp, KuratowskiEmbeddingUp, IsometricEmbeddingUp, and UniformEmbeddingUp.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetMetricEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of maps, selected completion representatives, an ambient function-space carrier, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, displayed-distance classifier stability, hsame transport, Cont replay, provenance, and local NameCert coverage for BEDC.Derived.FrechetMetricEmbeddingUp.

Constructive story: A $\mathsf{FrechetMetricEmbedding}^{\uparrow}$ packet is generated from a point row, a MetricSpaceUp source row, RealUp-valued distance-read rows, a finite probe-family classifier, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5551

A Concrete Naming Certificate for $\mathsf{CaristiFixedPoint}^{\uparrow}$

Following Caristi's fixed point theorem and the Ekeland complete-metric descent route, the $\mathsf{CaristiFixedPoint}^{\uparrow}$ packet records the finite surface by which a self-map satisfying a potential-decrease inequality reaches a seal-facing fixed endpoint. It sits beside the Banach fixed-point packet of chapter 396, the stability handoff of chapter 5856, and the variational descent ledger of chapter 3926. The packet names a complete-metric fixed-point certificate over one displayed potential and one displayed self-map; it is not a host minimizer theorem, a choice-selected descent sequence, quotient equality of points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5551.1 (Caristi fixed point carrier)**.** *A* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (X, d, \phi, T, E, B, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompleteMetric}^{\uparrow}$ *domain row, naming the metric carrier and its complete-metric handoff surface.*
2. *d is the distance row used for all comparisons between a point and its image under T.*
3. *ϕ is the lower-semicontinuous potential row, read through* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *and a regular-rational value ledger.*
4. *T is the self-map row whose graph is inspected only through the displayed* Cont *route.*
5. *E is the Caristi descent inequality row, recording the finite comparison $d(x, Tx) \leq \phi(x) - \phi(Tx)$ for every admitted probe row.*
6. *B is the Ekeland handoff row, exposing the variational descent route that turns the potential inequality into a terminal fixedness boundary.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *endpoint seal, used only after the complete-metric descent row has been consumed.*
8. *H records componentwise* hsame *transport for the domain, distance, potential, self-map, descent, Ekeland handoff, endpoint seal, replay, provenance, and local naming rows.*
9. *Q records displayed* Cont *replay from any fixed-point read through $X, d, \phi, T, E, B, R, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{CaristiFixedPoint}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, d, \phi, T, E, B, R, H, Q, P, N$. It has no coordinate for a host-selected minimizer, arbitrary choice sequence, quotient point equality, global contraction theorem, or ambient completeness beyond the displayed $\mathsf{CompleteMetric}^{\uparrow}$ *row.*

Lean4 ✓ `BEDC.Derived.CaristiFixedPointUp.CaristiFixedPointTasteGate single carrier alignment`

Theorem 5551.2 (Caristi fixed point NameCert obligations)**.** *For every accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier*

$$C = (X, d, \phi, T, E, B, R, H, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete-metric row, distance row, lower-semicontinuous potential, self-map row, Caristi descent inequality, Ekeland handoff, endpoint seal, transport, replay, provenance, and local naming. The obligation surface also records the comparison boundary with* $\mathsf{BanachFixedPoint}^{\uparrow}$ *and* $\mathsf{PicardIteration}^{\uparrow}$*: Banach supplies contraction-orbit fixedness, while this packet supplies potential-descent fixedness.*

Proof. Project the carrier of Definition 5551.1. Its analytic rows are exactly X, d, ϕ, T, E, B, R, and its structural rows are exactly H, Q, P, N.

The descent inequality is local to the displayed self-map. The row E reads d, ϕ, and T together, so a consumer cannot detach the distance comparison from the lower-semicontinuous potential ledger.

The fixed-point endpoint is downstream from complete-metric descent. The row B sends the Caristi inequality into the Ekeland variational handoff, and R seals only the resulting regular-rational and real endpoint boundary.

Finally the Banach/Picard comparison is only a boundary row. The provenance records the neighbouring fixed-point route, but the carrier has no contraction constant or Picard-orbit coordinate from which a Banach proof could be read. Hence the displayed rows exhaust the NameCert obligations. □

Theorem 5551.3 (Caristi fixed point Ekeland handoff)**.** *Every accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *packet factors its descent read through the finite route*

$$\mathsf{CompleteMetric}^{\uparrow}, \mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{EkelandVariationalPrinciple}^{\uparrow} \longmapsto E, B, R, H, Q, P, N.$$

The Ekeland handoff consumes the displayed potential-decrease inequality and returns only a certificate-local endpoint boundary. It does not produce a host minimizer, selected descent sequence, quotient equality of points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $C = (X, d, \phi, T, E, B, R, H, Q, P, N)$. The complete-metric row X and distance row d provide the only metric input available to the descent route.

The lower-semicontinuous row ϕ and inequality row E then form the Caristi descent datum. A consumer must read T through the same route, because E compares a point with its displayed image under T.

Apply the Ekeland handoff row B. It repacks the potential-decrease datum as the finite variational descent ledger named by $\mathsf{EkelandVariationalPrinciple}^{\uparrow}$, and it leaves the endpoint seal to R.

The transport, replay, provenance, and local name preserve this order. Since the carrier has no selector row for a host minimizer or arbitrary descent sequence, the handoff exports only the stated certificate-local endpoint boundary. □

Theorem 5551.4 (Caristi fixed point endpoint fixedness and non-escape)**.** *For an accepted* $\mathsf{CaristiFixedPoint}^{\uparrow}$ *carrier, the endpoint row R is classifier-fixed under the displayed self-map row T after the Caristi descent inequality E and Ekeland handoff B have been consumed. Every endpoint read factors through*

$$X, d, \phi, T, E, B, R, H, Q, P, N,$$

and no consumer may read host fixed-point equality, selected limits, quotient point equality, global contraction soundness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from the packet.*

Proof. Read the endpoint row through the carrier. The row R is introduced only after the complete-metric, potential, self-map, inequality, and Ekeland handoff rows have been displayed.

The fixedness assertion is classifier-internal. Applying T at the endpoint route is allowed only through Q, and the comparison back to R is transported by H inside the same local NameCert surface.

The non-escape boundary follows from the absence of additional coordinates. There is no row for a host equality of points, a selected limiting sequence, quotient equality, a contraction constant, or ambient real-completeness theorem. Thus every endpoint fixedness read remains inside $X, d, \phi, T, E, B, R, H, Q, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{CaristiFixedPoint}^{\uparrow}$ endpoint read factors through a CompleteMetricUp row, a lower-semicontinuous potential row, a displayed Caristi descent inequality, an EkelandVariationalPrincipleUp handoff, a RegSeqRatUp and RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: banachfixedpoint, banach-fixed-point-stability, ekeland-variational-principle, picard-iteration, complete-metric: $\mathsf{CaristiFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores complete-metric descent, lower-semicontinuous potential, self-map graph, Caristi inequality, Ekeland handoff, Banach/Picard comparison boundary, endpoint sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CaristiFixedPoint}^{\uparrow}$ packet is generated from a complete-metric row, distance row, lower-semicontinuous potential row, self-map row, Caristi descent inequality, Ekeland handoff, Banach/Picard comparison boundary, RegSeqRat and Real endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$, seed)

Scope closed: Definition 5551.1, Definition 5551.2, Definition 5551.3, and Definition 5551.4 seed the complete-metric potential-descent fixed-point packet.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CaristiFixedPoint}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host fixed-point equality, selected minimizers, selected descent sequences, quotient point equality, global contraction soundness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-semicontinuous potential readback, Caristi inequality exactness, Ekeland handoff, endpoint fixedness, Banach/Picard boundary confinement, transport, replay, provenance, and local naming for BEDC.Derived.CaristiFixedPointUp.

Chapter 5552

A Concrete Naming Certificate for $\mathsf{WeightedDeformationSpine}^{\uparrow}$

The $\mathsf{WeightedDeformationSpine}^{\uparrow}$ packet realizes the weighted deformation spine of subsection 9462.28.21 as finite trigger-weight deformation data. It sits beside the boundary-tile packet of chapter 3393, the reassignment-mass packet of chapter 5869, and the private/shared decoration packet of chapter 5896. The packet records trigger support, weights, deformation route, boundary-tile readback, transport, replay, provenance, and local naming together; it is not a biological mechanism, stochastic model, hidden graph search, quotient codon equality, or unrecorded trigger oracle.

Definition 5552.1 (Weighted deformation operator carrier)**.** *A* $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, L, D, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the trigger-support row, naming the finite codon-window vertices admitted by the packet.*
2. *L is the weight ledger, assigning each admitted trigger row its displayed reassignment mass.*
3. *D is the deformation-route row, recording the nested connected supports and their Hamming-adjacent spine.*
4. *B is the boundary-tile readback row, tying the Stop, AGR, AAA, and AUA modules back to the DNA-window six boundary-tile source.*
5. *H records componentwise* hsame *transport for S, L, D, B, C, P, N.*
6. *C records displayed* Cont *replay from any deformation-spine read through trigger support, weights, route, boundary readback, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{WeightedDeformationSpine}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{ReassignmentMassSubcu}$ $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$*, the weighted-spine theorem of Definition 9462.1260,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, L, D, B, H, C, P, N. It has no coordinate for a biological causal mechanism, stochastic evolution model, quotient codon semantics, hidden global enumeration, or unweighted trigger boundary.

Theorem 5552.2 (Weighted deformation spine NameCert obligations)**.** *For every accepted* WeightedDeformationSp *carrier*

$$W = (S, L, D, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ *surface consists exactly of carrier admission for trigger support, weight ledger, deformation route, boundary-tile readback, componentwise transport, displayed replay, provenance, and local naming. The weighted deformation route must preserve the nested connected supports and the spine path*

$$\mathrm{UAG} - \mathrm{UAA} - \mathrm{UGA} - \mathrm{AGA} - \mathrm{AGG}$$

with branch

$$\mathrm{AGA} - \mathrm{AAA} - \mathrm{AUA}.$$

Proof. Project the carrier of Definition 5552.1. The support, weight, route, boundary, transport, replay, provenance, and local name are exactly the displayed coordinates.

The route is weighted, not merely supported. The row L must be read before D, because the nested connected supports are selected by maximum-mass connected candidates from the source theorem.

The boundary readback is also explicit. The row B attaches the Stop, AGR, AAA, and AUA modules to the DNA-window six boundary-tile source, so a consumer cannot reinterpret the spine as an ungrounded graph path.

Finally H, C, P, N preserve and name the same rows. Since biological mechanism, stochastic dynamics, quotient codon equality, hidden enumeration, and unweighted trigger boundaries have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5552.3 (Weighted deformation spine trigger stability)**.** *For accepted* $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *carriers, componentwise* hsame *transport preserves the trigger support S, weight ledger L, deformation route D, and boundary-tile readback B as one packet. A deformation consumer may vary only through displayed transport and replay rows; it cannot keep the same route while dropping the weight ledger, or keep the same support while replacing the boundary readback by an unrecorded trigger classifier.*

Proof. Start with the accepted carrier $W = (S, L, D, B, H, C, P, N)$. The transport row H is componentwise across support, weights, route, boundary readback, replay, provenance, and local naming.

The weight ledger is tied to the route. If D is preserved, the maximum-mass connected-candidate interpretation still requires the weights stored in L; otherwise the deformation boundary is no longer the weighted spine named by the packet.

The boundary readback is tied to the trigger support. If S is preserved, the Stop, AGR, AAA, and AUA module readback must remain the displayed row B, not a hidden classifier outside the carrier.

Thus every stable trigger read preserves S, L, D, B, H, C, P, N together. The packet cannot be reduced to support alone, route alone, or weights alone. □

Theorem 5552.4 (Weighted deformation spine TasteGate surface)**.** *The future Lean TasteGate file for namespace* BEDC.Derived.WeightedDeformationSpineUp *must provide* Nontrivial $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *and* FieldFaithful $\mathsf{WeightedDeformationSpine}^{\uparrow}$ *instances preserving the same finite carrier rows S, L, D, B, H, C, P, N. Nontriviality is witnessed by the distinction between support-only, weight-only, and route-only reads; field faithfulness preserves the joint trigger-support, weight-ledger, deformation-route, boundary-readback, transport, replay, provenance, and local naming surface.*

Proof. The carrier contains at least three independently readable analytic rows: support S, weights L, and route D. Collapsing the packet to one of them erases the other two and therefore cannot preserve the NameCert obligations.

Boundary readback supplies a further separate row. The row B connects the deformation spine to the DNA-window boundary-tile source, while H, C, P, N transport, replay, source, and name the packet.

A TasteGate consumer must preserve all of these rows as fields of one carrier. That is exactly the paper-side surface for the stated Nontrivial and FieldFaithful instances. □

Falsifiable prediction: Removing the weight ledger changes the accepted deformation boundary while leaving the trigger support readable.

Independence witness: observervtracekernel, finitewindowobserver, observerfilterbudgetselector, codon-window-sibling slugs: explain why none is bijective because only this packet stores trigger support, weights, deformation route, boundary-tile readback, transport, replay, provenance, and local naming together.

Closure status for $\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{WeightedDeformationSpine}^{\uparrow}$ packet is generated from finite trigger-support rows, a weight ledger, a connected deformation route, boundary-tile readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5552.1, Definition 5552.2, Definition 5552.3, and Definition 5552.4 seed the finite trigger-weight deformation packet over the DNA-window spine.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeightedDeformationSpine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unweighted trigger classification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, trigger-support exactness, weight-ledger exactness, deformation-route stability, boundary-tile readback, transport, replay, provenance, local naming, Nontrivial WeightedDeformationSpineUp, and FieldFaithful WeightedDeformationSpineUp for BEDC.Derived.WeightedDeformationSpineUp.

$\mathsf{CompactRealInterval}^{\uparrow}$ routes closed interval compactness through chapter 38, chapter 291, chapter 301, and chapter 3471; the child files below keep the carrier, finite-net, diagonal-selection, and Real-interval routes separate.

Chapter 5553

A Concrete Naming Certificate for $\mathsf{CompactRealInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the Heine-Borel finite-net reading of closed intervals, the $\mathsf{CompactRealInterval}^{\uparrow}$ packet records a compact closed interval as finite BEDC NameCert data. It sits over the real-name seal of chapter 38, the dyadic scale and tolerance rows of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the finite-net compactness siblings chapter 129 and chapter 3471. The packet is an interval carrier for later compact-real consumers; it does not import host compactness, selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5553.1 (Compact real interval carrier)**.** *A* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, B, O, D, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two located* $\mathsf{Real}^{\uparrow}$ *endpoint rows, each exposed through finite real-name windows before any interval request is consumed.*
2. *O is the order and apartness ledger, recording the displayed endpoint order, located overlap tests, boundary apartness rows, and refusal of trichotomy outside the inspected interval windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite tolerance ledger, recording common-scale comparisons, mesh radii, endpoint slack, and dyadic containment checks.*
4. *W is the finite dyadic net schedule, listing the mesh windows used to inspect the interval between the endpoint rows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, carrying the finite stream requests that read endpoint and mesh data without opening a quotient stream.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking endpoint, mesh, order, and tolerance checks as regular rational finite windows.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and inspected mesh values.*
8. *H records componentwise* hsame *transport for endpoints, order/apartness ledgers, dyadic tolerances, finite nets, stream windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-interval request through $A, B, O, D, W, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, O, D, W, S, R, E, H, C, P, N$. It has no coordinate for host compactness, arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness.

Theorem 5553.2 (Compact real interval finite-net obligations)**.** *For every accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier*

$$I = (A, B, O, D, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *surface consists of the following finite obligations:*

1. *endpoint admission for the two located* $\mathsf{Real}^{\uparrow}$ *rows A and B;*
2. *order and apartness admission for the displayed ledger O;*
3. *dyadic scale and tolerance admission for the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*
4. *finite-net schedule admission for W, including mesh containment and endpoint slack;*
5. *StreamName finite-window admission for S and regular rational readback for R;*
6. *terminal Real seal admission for E after the finite endpoint, dyadic, stream, and regular rows are visible;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No compact-interval obligation is stored outside these rows.

Proof. Project the carrier of Definition 5553.1. The endpoint rows A, B and the order/apartness row O expose the interval boundary and the located comparison surface available to the packet.

Read the finite inspection rows next. The dyadic ledger D supplies common-scale tolerance checks, while W lists the finite mesh windows to which interval inspection is confined.

The stream and regular rows are downstream. The row S carries the finite StreamName requests, and R repacks the same endpoint, mesh, and tolerance reads as regular rational windows before the terminal Real seal E is consumed.

Finally, H, C, P, N transport, replay, source, and locally name the same finite packet. Since the carrier has no coordinate for host compactness, arbitrary open covers, selected subcovers, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5553.3 (Compact real interval NameCert scope)**.** *Every accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *consumer factors through the displayed route*

$$\mathsf{Real}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{HeineBorelFiniteNet}^{\uparrow}.$$

Thus the compact closed-interval surface exports only endpoint Real seals, dyadic tolerance and finite-net data, finite StreamName windows, regular rational readbacks, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not export host compactness, arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint and tolerance rows. The carrier exposes the interval through A, B, O, D, so every later compact-interval read is attached to the same located Real endpoints and dyadic tolerance ledger.

Use the finite-net and window rows. The row W records the displayed dyadic net schedule, S restricts stream requests to finite windows, and R records the regular rational readback of exactly those windows.

The compactness siblings are read only through the same finite rows. The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ provenance in P witnesses compatibility with finite-net compactness, but it does not add an arbitrary cover selector or host compactness coordinate.

The structural rows H, C, P, N preserve the route by transport, replay, sourcing, and local naming. Since no refused object appears in the carrier, every accepted consumer remains inside the stated scope. □

Theorem 5553.4 (Compact real interval effective completion route)**.** *Let $I = (A, B, O, D, W, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CompactRealInterval}^{\uparrow}$ *carrier. Any consumer that exports endpoint or mesh data to a completion-facing reader must pass through finite interval nets,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, and then the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *route of chapter 5930. The export does not assert open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5553.1. Endpoint and mesh data are displayed only by A, B, O, D, W, S, R, E before the structural rows H, C, P, N are consumed.

Apply Definition 5553.2. The finite-net obligation exposes endpoint admission, order and apartness, dyadic scale and tolerance, the finite net schedule, StreamName windows, regular rational readback, and the terminal Real seal in that order.

Apply Definition 5553.3. The compactness side is finite-net compactness through $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, not an arbitrary open-cover selector.

The completion-facing reader therefore receives the same finite data through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ before entering chapter 5930. Componentwise hsame transport and displayed Cont replay preserve the route, so no quotient, choice, selected-subcover, or ambient-completeness coordinate is introduced. □

Falsifiable prediction: Every $\mathsf{CompactRealInterval}^{\uparrow}$ read factors through two located $\mathsf{Real}^{\uparrow}$ endpoint rows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a finite dyadic net schedule, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, finite-net compactness sibling provenance, componentwise hsame, displayed Cont, Pkg, and local $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ rows.

Independence witness: real, dyadicratcore, streamname, regseqrat, compactmetric, heineborelfinitemesh: $\mathsf{CompactRealInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint Real seals, order/apartness data, dyadic common-scale tolerances, a finite interval net schedule, finite stream windows, regular readback, finite-net compactness provenance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactRealInterval}^{\uparrow}$ packet is generated from two located RealUp endpoint rows, an order and apartness ledger, a DyadicRatCoreUp finite tolerance ledger, a finite dyadic net schedule, StreamNameUp finite windows, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5553.1, Definition 5553.2, and Definition 5553.3 seed the compact closed-interval carrier over endpoint Real seals, dyadic tolerances, finite nets, StreamName windows, RegSeqRat readback, and finite-net compactness sibling provenance.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactRealInterval}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactness, arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealInterval}^\uparrow$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net schedule exactness, dyadic tolerance stability, StreamName window pullback, RegSeqRat readback, Real endpoint sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalUp.

Theorem 5553.5 (Compact real interval closed-bounded dependency route)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *consumer that inspects the compact-interval packet must first pass through the closed-bounded interval endpoint rows of chapter 1591, the dyadic finite-net rows recorded by the compact interval carrier, the* $\mathsf{StreamName}^\uparrow$ *finite windows of chapter 126, the* $\mathsf{RegSeqRat}^\uparrow$ *readback rows, and the terminal* $\mathsf{Real}^\uparrow$ *seal of chapter 38. The route exposes interval endpoints, finite net data, stream observations, regular rational reconstruction, and local real naming before any compact interval consumer may inspect the packet.*

Proof. Project the compact real interval carrier. The closed-bounded endpoint rows supply the only interval source, and the dyadic finite-net rows supply the only finite covering data inside the packet.

The stream and regular-readback rows are then read in order. The $\mathsf{StreamName}^\uparrow$ windows specify the finite observations, and the $\mathsf{RegSeqRat}^\uparrow$ rows reconstruct the rational readback attached to those observations.

The terminal $\mathsf{Real}^\uparrow$ seal is available only after these endpoint, finite-net, stream, and readback rows have been displayed. Thus a compact interval consumer receives the actual L10 dependency route rather than an ungrounded compactness credential. □

Theorem 5553.6 (Compact real interval Cauchy diagonal dependency)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *finite-net or nested-window consumer that passes to the* $\mathsf{CauchyDiagonalSelection}^\uparrow$ *sibling packet of chapter 4080 must first expose the interval endpoint rows, the located interval row, the compact finite-net row, the* $\mathsf{StreamName}^\uparrow$ *window row, the* $\mathsf{RegSeqRat}^\uparrow$ *handoff row, the terminal* $\mathsf{Real}^\uparrow$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^\uparrow}$ *row. The route may not import selected subsequences, open-cover compactness, quotient real equality, or ambient completeness.*

Proof. Project the compact real interval carrier of Definition 5553.1. The endpoint rows A, B, the order and located interval row O, the finite dyadic net schedule W, and the compact finite-net entries in P are visible before any diagonal read is available.

The stream and rational readback rows are then forced into the Cauchy-facing order. The $\mathsf{StreamName}^\uparrow$ row S supplies finite observation windows, the $\mathsf{RegSeqRat}^\uparrow$ row R supplies the rational handoff, and the $\mathsf{Real}^\uparrow$ row E supplies the terminal seal.

Finally H, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores a selected subsequence, arbitrary open cover, quotient real equality, or ambient completeness principle, every compact-interval handoff to $\mathsf{CauchyDiagonalSelection}^\uparrow$ is confined to the displayed rows. □

Theorem 5553.7 (Compact real interval dossier lattice route)**.** *Every* $\mathsf{CompactRealInterval}^\uparrow$ *dossier consumer that inspects an accepted interval packet must first display the* $\mathsf{ClosedBoundedInterval}^\uparrow$

endpoint rows, the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the terminal* $\mathsf{Real}^{\uparrow}$ *seals, the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net compatibility row, and the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *provenance row. Only after those rows have been replayed through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompactRealInterval}^{\uparrow}}$ *row may the consumer inspect the compact interval packet itself.*

Proof. Project the carrier of Definition 5553.1. The endpoint rows A and B give the closed-bounded interval source, while the dyadic tolerance row D records the finite scale and containment ledger attached to those endpoints.

The stream and readback rows are then forced into the same route. The $\mathsf{StreamName}^{\uparrow}$ row S supplies only finite observation windows, the $\mathsf{RegSeqRat}^{\uparrow}$ row R reconstructs the corresponding rational readback, and the $\mathsf{Real}^{\uparrow}$ seal row E is reached only after those finite reads are visible.

The compactness-facing rows are provenance and compatibility rows, not host compactness principles. The $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ entries in P record finite-net compatibility and finite-net provenance before any downstream consumer can use the interval packet as compact data.

Finally H, C, P, N transport, replay, source, and name that same lattice route. Since no carrier coordinate stores arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness, a compact interval consumer can inspect only the displayed dossier lattice. □

Every $\mathsf{CompactRealInterval}^{\uparrow}$ finite-net consumer records its sibling route through the finite epsilon-net surface of chapter 1767, the located interval source of chapter 1709, the closed-bounded interval net packet of chapter 1250, and the terminal real-name seal of chapter 38 before any compact interval packet is read. The compact interval finite-net read also consumes the $\mathsf{CompactMetric}^{\uparrow}$ carrier of chapter 129 and the $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ finite-cover route of chapter 3471 before the finite epsilon-net ledger is exported. The route keeps finite nets, located endpoints, closed-bounded interval data, compact-metric carrier rows, Heine–Borel finite-cover rows, and RealUp seals visible as dependency rows rather than as hidden host compactness.

Proposition 5553.8 (Compact real interval real-source route)**.** *Every* $\mathsf{CompactRealInterval}^{\uparrow}$ *consumer that reads compact interval rows first passes through the endpoint* $\mathsf{Real}^{\uparrow}$ *source of chapter 38, the metric reading surface of chapter 1638, and the finite-net route recorded by chapter 5553. The compact interval packet therefore exposes real endpoints, metric comparison data, and finite-net compatibility before any compact interval row is exported.*

Chapter 5554

A Concrete Naming Certificate for $\mathsf{LocatedCutCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCutCompletion}^{\uparrow}$ packet names the finite lower–upper dyadic bracket surface by which Real and Completion consumers compare located-cut evidence with regular Cauchy readback. It sits beside the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational handoff of chapter 301, the terminal real seal of chapter 38, and the Cauchy–Dedekind bridge of chapter 2346. The packet is a BEDC-native carrier surface for located lower and upper brackets; it does not construct $\mathsf{Real}^{\uparrow}$ from primitive Dedekind cuts and does not add an ambient completeness principle.

Definition 5554.1 (Located-cut completion bracket carrier)**.** *A* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. B^- *is the lower dyadic bracket ledger. It records inhabited lower cells, lower endpoint witnesses, and lower refinement requests through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
2. B^+ *is the upper dyadic bracket ledger. It records inhabited upper cells, upper endpoint witnesses, and upper refinement requests through the same dyadic scale row.*
3. G *is the located comparison row. For every displayed lower endpoint and upper endpoint request, it records a finite separation, overlap, or refinement witness between* B^- *and* B^+*.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule used to expose only the finitely requested bracket observations.*
5. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that compares the displayed brackets with a regular rational Cauchy route.*
6. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger shared by* B^-, B^+, G, W, R*.*
7. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the lower bracket, upper bracket, located comparison, finite window, regular readback, and dyadic refinement rows have been displayed.*
8. H *records componentwise* hsame *transport for the bracket rows, comparison row, window row, readback row, dyadic row, real seal, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay of the bracket-completion route*

$$B^-, B^+, G, D \longmapsto W, R \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedCutCompletion}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{RegSeqRat}^\uparrow$, $\mathsf{Real}^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^\uparrow}$ *row.*

The classifier accepts only packets that display both bracket ledgers, the shared located comparison row, the common dyadic refinement ledger, the finite window schedule, the regular readback row, the terminal real seal, and the structural rows. It has no coordinate for primitive Dedekind cuts, quotient equality, selected limits, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.LocatedCutCompletionUp.LocatedCutCompletionTasteGate single carrier alignment`

Theorem 5554.2 (Located-cut bracket inhabitation)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier $L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^\uparrow}$ *surface includes inhabited lower and upper dyadic bracket reads. Every public bracket read must pass through the displayed rows B^- and B^+ and the shared dyadic ledger D before it can reach W, R, E, H, C, P, N.*

Proof. Carrier admission is projection from Definition 5554.1. The accepted packet exposes B^- and B^+ as separate lower and upper bracket ledgers, each tied to the same dyadic refinement row D.

The classifier has no bracket coordinate outside these ledgers. Therefore any public bracket read is a read of the displayed lower ledger or upper ledger together with the shared dyadic scale data.

The remaining rows W, R, E, H, C, P, N only schedule, read back, seal, transport, replay, source, and name that same bracket route. They cannot supply a detached lower or upper cut object. □

Theorem 5554.3 (Located-cut comparison obligation)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier, the row G is the only located comparison interface between the lower bracket ledger B^- and the upper bracket ledger B^+. A Real or Completion consumer cannot compare the two bracket sides without first reading B^-, B^+, G, D inside the same finite packet.*

Proof. Read the carrier rows named in Definition 5554.1. The lower and upper bracket ledgers are displayed separately, and the located comparison row G is the sole row that relates their endpoint requests.

A consumer that omits G has only two unrelated dyadic ledgers. A consumer that omits either bracket side has no two-sided located question to compare.

Thus the comparison route is forced through B^-, B^+, G, D. The stream, regular readback, real seal, and structural rows preserve the comparison after it is displayed, but they do not create a second comparison principle. □

Theorem 5554.4 (Common dyadic refinement stability)**.** *Let $L = (B^-, B^+, G, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{LocatedCutCompletion}^\uparrow$ *carrier. Any refinement of a displayed lower bracket or upper bracket that remains inside the common dyadic ledger D preserves the same* $\mathsf{LocatedCutCompletion}^\uparrow$ *packet route through G, W, R, E, H, C, P, N.*

Proof. The carrier stores one dyadic refinement ledger D for both bracket sides. Refining a lower or upper endpoint inside D therefore keeps the refined observation in the same finite packet rather than moving it to an external cut object.

The located comparison row G is read after the two bracket ledgers and the shared dyadic scale are fixed. A refinement that remains in D is still a request to the same comparison row.

Finally W, R, E, H, C, P, N are attached to this packet route. They can replay and name the refined bracket read, but they cannot replace the common dyadic ledger by an independent refinement principle. □

Theorem 5554.5 (Classifier transport under bracket equivalence)**.** *For every accepted* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier, componentwise* hsame *transport of the lower bracket, upper bracket, located comparison, dyadic refinement, stream window, regular readback, real seal, replay, provenance, and local naming rows preserves classifier acceptance. Equivalence of bracket displays is therefore transported only through H and the displayed packet rows.*

Proof. The transport row H is a coordinate of the accepted carrier. Its scope is componentwise: it transports $B^-, B^+, G, W, R, D, E, C, P, N$ as rows of the same packet.

The classifier of Definition 5554.1 accepts exactly those displayed rows. Applying H preserves the rows it names and leaves no unmentioned quotient equality coordinate to inspect.

Hence bracket equivalence is a packet-level transport statement. It is not equality of primitive cuts and not quotient equality of completed real names. □

Theorem 5554.6 (Real seal handoff for located-cut completion)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *carrier factors through*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The terminal seal E is reached only after the lower bracket, upper bracket, located comparison, common dyadic refinement, finite window schedule, and regular rational readback rows have been displayed.

Proof. Start from the bracket rows B^-, B^+ and the located comparison row G. The shared dyadic ledger D fixes the common refinement scale on which both bracket sides are read.

The finite observation schedule W and regular rational readback row R then expose the bracket route to the Cauchy-facing side of the packet. At this point the consumer has only displayed finite rows, not a selected limit or quotient representative.

The terminal real seal E is therefore a handoff after B^-, B^+, G, D, W, R. The structural rows H, C, P, N transport, replay, source, and name the same handoff, so the excluded primitives and ambient completeness principle have no carrier coordinate. □

Falsifiable prediction: Every accepted $\mathsf{LocatedCutCompletion}^{\uparrow}$ Real or Completion read factors through displayed lower bracket, upper bracket, located comparison, dyadic refinement, finite window, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-dedekind-bridge: $\mathsf{LocatedCutCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two-sided located bracket ledgers, common dyadic refinement, regular Cauchy readback, terminal Real sealing, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5554.1, Definition 5554.2, Definition 5554.3, Definition 5554.4, Definition 5554.5, and Definition 5554.6 seed the finite located lower–upper bracket completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCutCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim primitive Dedekind cuts, quotient equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked bracket inhabitation, located comparison, common dyadic refinement stability, classifier transport under bracket equivalence, Real seal handoff, and nonescape for BEDC.Derived.LocatedCutCompletionUp.

Constructive story: A LocatedCutCompletion$^{\uparrow}$ packet is generated from lower and upper dyadic bracket ledgers, a located comparison row, common dyadic refinement, finite StreamName windows, RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5555

A Concrete Naming Certificate for $\mathsf{DedekindEtaCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{DedekindEtaCompletion}^{\uparrow}$ packet records an eta-completion route from regular rational names to located Dedekind-style real boundaries as finite BEDC NameCert data. It sits over the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the Cauchy criterion equivalence surface of chapter 3343, and the terminal real-name seal of chapter 38. The packet gives completion consumers a displayed eta route without quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5555.1 (DedekindEtaCompletion carrier)**.** *A* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, R, D, Q, L, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the rational observations retained by the eta route.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding each retained stream window to a regular rational observation.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying positive-radius witnesses, half-radius refinements, and strict comparison cells used by the regular readback.*
4. *Q is the* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ *comparison row, recording the finite Cauchy criterion checks that certify the retained windows as a regular eta-family.*
5. *L is the located lower-and-upper cut ledger generated from S, R, D, Q; it records only displayed rational cut tests and their dyadic slack.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after the stream, regular, dyadic, criterion, and cut-ledger rows have been displayed.*
7. *H records componentwise* hsame *transport for the window, readback, tolerance, Cauchy criterion, cut-ledger, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, R, D \longmapsto Q \longmapsto L \longmapsto A.$$

9. *P is the* Pkg *provenance over* $\mathsf{DedekindEtaCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these ten rows. Its classifier compares accepted packets only through $S, R, D, Q, L, A, H, C, P, N$ and has no coordinate for quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.DedekindEtaCompletionUp.DedekindEtaCompletionTasteGate single carrier alignment`

Theorem 5555.2 (DedekindEtaCompletion NameCert obligations)**.** *For an accepted* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier*

$$E = (S, R, D, Q, L, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the stream-window schedule, regular rational readback, dyadic tolerance ledger, Cauchy criterion comparison row, located cut ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring every rational observation used by the eta route to be addressed by S before it can be read by R;*
3. *dyadic tolerance exactness, requiring every comparison in the regular readback and every cut test in L to consume a displayed tolerance or refinement from D;*
4. *Cauchy criterion compatibility, requiring Q to certify the retained windows as the same regular eta-family before the cut ledger is exported;*
5. *cut locatedness, requiring L to expose lower and upper rational tests only through S, R, D, Q;*
6. *real-seal handoff, requiring A to be downstream of the same displayed eta route;*
7. *classifier transport and replay, requiring H and C to preserve exactly the listed rows and no external completion object;*
8. *non-escape, excluding quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 5555.1. The accepted packet exposes exactly the stream-window, regular readback, dyadic tolerance, Cauchy criterion, cut-ledger, real-seal, and structural rows.

Window exactness follows from the order of the carrier. A rational observation enters the eta route through S, and R can read it only as a retained finite window. There is no coordinate for an unaddressed stream or a quotient of streams.

Dyadic tolerance exactness is supplied by D. Every regular comparison and every cut test has to name a positive-radius witness, refinement, or strict comparison cell before it can be used by Q or L.

The Cauchy criterion row Q is the only compatibility surface. It certifies that the retained regular windows form the eta-family consumed by the cut ledger, so no selected limit or countable-choice schedule is introduced.

The cut ledger L is downstream of S, R, D, Q, and the real seal A is downstream of L. Hence lower and upper rational tests and Real-facing reads remain inside the displayed eta route.

Finally H, C, P, N transport, replay, source, and name only the listed rows. Since the excluded principles have no carrier coordinate, they cannot become local NameCert obligations. □

Theorem 5555.3 (DedekindEtaCompletion L10 handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{DedekindEtaCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyCriterionEquivalence}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The exported object is a displayed eta-completion handoff from regular rational windows to a located real seal. It is not a quotient stream, selected host limit, countable-choice construction, trichotomy split, open-cover compactness principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the accepted carrier and project the public readback rows. A Real-facing or completion-facing consumer can reach the seal row A only by following the replay route named in Definition 5555.1.

The first stage is finite-window data. The row S names the inspected stream windows, and R reads those windows as regular rational observations.

The second stage is tolerance and criterion data. The dyadic ledger D supplies the radius cells consumed by the regular readback, and Q records the Cauchy criterion checks for the same retained family.

The cut ledger L then exposes the Dedekind-style lower and upper rational tests generated by that family. Since L is downstream of S, R, D, Q, it cannot select a host limit or quotient stream representative.

Finally A seals only this displayed route, while H, C, P, N preserve, replay, source, and name the same finite packet. Therefore the handoff exports the eta route and none of the excluded classical completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5555.1, Definition 5555.2, and Definition 5555.3 seed the finite eta-completion route over StreamName, RegSeqRat, DyadicRatCore, CauchyCriterionEquivalence, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindEtaCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected host limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, dyadic tolerance exactness, Cauchy criterion compatibility, cut locatedness, real-seal handoff, classifier transport, replay, and non-escape for BEDC.Derived.DedekindEtaCompletionUp.

Constructive story: A $\mathsf{DedekindEtaCompletion}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a CauchyCriterionEquivalence comparison row, a located cut ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5556

A Concrete Naming Certificate for $\mathsf{GITQuotient}^{\uparrow}$

Following Mumford's geometric invariant theory, the $\mathsf{GITQuotient}^{\uparrow}$ packet records the finite quotient-facing route from a projective or affine scheme presentation with group action data to invariant-ring rows, semistable-window rows, orbit-separation rows, and a displayed quotient name. It sits after the scheme carrier of chapter 153, the affine-variety surface of chapter 154, the projective-variety surface of chapter 155, the group certificate of chapter 41, the field certificate of chapter 45, and the ring certificate of chapter 43. The packet names a finite GIT quotient certificate without importing host quotient equality, stack objects, selected orbit representatives, or an external moduli functor.

Definition 5556.1 (GIT quotient carrier)**.** *A* $\mathsf{GITQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (S, A, P, G, I, Q, O, H, C, L, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Scheme}^{\uparrow}$ *source row naming the ambient scheme surface and the affine-cover ledger on which the quotient construction is read.*
2. *A is the* $\mathsf{AffineVar}^{\uparrow}$ *chart row used for invariant-ring computations on affine windows of S.*
3. *P is the* $\mathsf{ProjectiveVar}^{\uparrow}$ *presentation row naming the projective closure, homogeneous coordinates, linearization window, and projective incidence ledger.*
4. *G is the acting* $\mathsf{Group}^{\uparrow}$ *row over the displayed* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{Ring}^{\uparrow}$ *scalar rows consumed by the action and invariant-ring ledgers.*
5. *I is the invariant-ring ledger. It records the finite generator rows, relation rows, grading rows, and action-fixedness checks used to form the quotient coordinate packet.*
6. *Q is the quotient-name row. It records the displayed affine or projective quotient carrier obtained from I and the selected semistable window, but it is not a host quotient object.*
7. *O is the orbit-separation ledger. It records semistable-window membership, closed-orbit comparison rows, invariant-value comparison rows, and refusal rows for unstable histories.*
8. *H records componentwise* hsame *transport for the scheme, affine, projective, group, scalar, invariant-ring, quotient-name, orbit-separation, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from every quotient read through $S, A, P, G, I, Q, O, H, L, N$.*
10. *L is the* Pkg *provenance over* $\mathsf{GITQuotient}^{\uparrow}$, $\mathsf{Scheme}^{\uparrow}$, $\mathsf{AffineVar}^{\uparrow}$, $\mathsf{ProjectiveVar}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for host quotient equality, quotient stacks, selected orbit representatives, external moduli functors, or unstated invariant-theory completions.

Lean4 ✓ `BEDC.Derived.GITQuotientUp.GITQuotientTasteGate single carrier alignment`

Theorem 5556.2 (GIT quotient NameCert obligations)**.** *For every accepted* $\mathsf{GITQuotient}^{\uparrow}$ *carrier*

$$Z = (S, A, P, G, I, Q, O, H, C, L, N),$$

the local $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$ *surface consists exactly of scheme admission through S, affine-chart admission through A, projective-presentation admission through P, group-action and scalar admission through G, invariant-ring visibility through I, quotient-name exposure through Q, orbit-separation and semistable-window routing through O, componentwise transport through H, displayed replay through C, provenance through L, and local naming through N.*

Proof. Project the carrier of Definition 5556.1. The geometric source rows are S, A, P, G, the quotient-facing rows are I, Q, O, and the structural rows are H, C, L, N.

The invariant-ring obligation is finite. Its generator, relation, grading, and action-fixedness entries are rows inside I, and each is read through the affine chart, projective presentation, group action, and scalar certificates already displayed by A, P, G.

The quotient-name and orbit-separation obligations are then separate rows. The row Q exposes only the quotient carrier built from the invariant ledger and semistable window, while O records which semistable histories can be separated by invariant values and which unstable histories are refused.

Transport, replay, provenance, and local naming preserve exactly these rows. Since no coordinate names a host quotient, quotient stack, selected orbit representative, or external moduli functor, the listed rows exhaust the NameCert surface. □

Theorem 5556.3 (GIT quotient invariant-ring handoff)**.** *Every downstream read of an accepted* $\mathsf{GITQuotient}^{\uparrow}$ *quotient-name row factors through the invariant-ring ledger*

$$S, A, P, G \longmapsto I \longmapsto Q.$$

Thus the quotient carrier is exposed as finite invariant-ring and semistable-window data over the displayed scheme, affine, projective, group, field, and ring rows, not as a host quotient object or a stack presentation.

Proof. Start with the displayed source rows S, A, P, G. They supply the scheme cover, affine chart, projective presentation, action row, and scalar rows on which invariant functions are allowed to be read.

Read the invariant-ring ledger I. The finite generator, relation, grading, and fixedness checks in I are the only rows used to build the quotient coordinate packet.

The quotient-name row Q is then a projection from that packet together with the semistable-window data carried by O. The structural rows H, C, L, N can transport and replay this route, but they cannot add a host quotient object or a stack presentation. □

Theorem 5556.4 (GIT quotient orbit-separation nonescape)**.** *No accepted* $\mathsf{GITQuotient}^{\uparrow}$ *packet exports orbit separation outside the displayed semistable-window and invariant-value rows. If a consumer compares two semistable histories, the comparison must pass through* O *and the invariant values supplied by* I*; if a history lies outside the displayed semistable window, the refusal row in* O *blocks quotient-name export for that history.*

Proof. Project an accepted packet $Z = (S, A, P, G, I, Q, O, H, C, L, N)$. The only orbit-facing row is O, and the only scalar-valued separating data available to O are the invariant values recorded by I.

For semistable histories, O requires the displayed window-membership row before any comparison is read. The comparison is then routed to invariant values from I, so separation remains inside the finite invariant-ring ledger.

For histories outside the semistable window, the refusal row in O prevents the quotient-name row Q from being exported for that history. Componentwise hsame transport, Cont replay, Pkg provenance, and local naming preserve the refusal because none of them introduces an additional orbit classifier. □

Falsifiable prediction: Every accepted $\mathsf{GITQuotient}^{\uparrow}$ quotient-name read factors through a SchemeUp row, an AffineVarUp chart row, a ProjectiveVarUp presentation row, GroupUp action data over FieldUp and RingUp scalar rows, an invariant-ring ledger, a semistable-window ledger, orbit-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: scheme, affinevar, projectivevar, group, field, ring: $\mathsf{GITQuotient}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a group action, invariant-ring rows, semistable-window rows, quotient-name exposure, orbit-separation refusal, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GITQuotient}^{\uparrow}$ packet is generated from a SchemeUp source row, AffineVarUp chart row, ProjectiveVarUp presentation row, GroupUp action data over FieldUp and RingUp scalar rows, invariant-ring rows, semistable-window rows, orbit-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$, seed)

Scope closed: Definition 5556.1, Definition 5556.2, Definition 5556.3, and Definition 5556.4 seed the finite invariant-ring and semistable-window route for GIT quotient consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{GITQuotient}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quotient equality, quotient stacks, selected orbit representatives, external moduli functors, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-ring handoff, semistable-window routing, orbit-separation refusal, quotient-name exposure, replay stability, provenance exhaustion, and local naming for BEDC.Derived.GITQuotientUp.

Chapter 5557

A Concrete Naming Certificate for $\mathsf{AubinLions}^{\uparrow}$

Following Aubin and Lions, the $\mathsf{AubinLions}^{\uparrow}$ packet records the compactness route used for time-dependent Sobolev families as finite NameCert data. It sits over the Sobolev source of chapter 440, the weak-derivative time-translation surface of chapter 2210, the compact-window handoff of chapter 2218, the Cauchy readback of chapter 128, and the scalar seal of chapter 38. The packet names only the displayed compactness handoff; it does not import host subsequence selection, quotient completion, ambient real completeness, or an unrestricted compact embedding theorem.

Definition 5557.1 (Aubin-Lions carrier)**.** *An* $\mathsf{AubinLions}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, T, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, exposing the space-regularity, weak-derivative ledger, integrability window, and local naming data consumed by the compactness request.*
2. *T is the* $\mathsf{WeakDerivative}^{\uparrow}$ *time-translation row, recording the finite temporal difference windows, derivative bounds, and equicontinuity budget that keep translated reads inside the same Sobolev source.*
3. *R is the* $\mathsf{RellichKondrachov}^{\uparrow}$ *compact-window row, carrying the finite spatial compactness handoff available on inspected bounded windows.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *readback row, recording Cauchy comparison data for the compact-window images without completing a host quotient space.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *scalar seal for norms, radii, tolerances, and translation moduli.*
6. *H records componentwise* hsame *transport for the Sobolev source, time-translation row, compact-window row, complete-metric readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an Aubin-Lions compactness request through S, T, R, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{AubinLions}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{RellichKondrachov}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, R, M, E, H, C, P, N$. It has no coordinate for a selected subsequence, quotient function equality, host compact embedding, ambient completed Sobolev space, global time regularity theorem, unrestricted real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.AubinLionsUp.AubinLionsTasteGate single carrier alignment`

Theorem 5557.2 (Aubin-Lions NameCert obligations)**.** *For every accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier*

$$A = (S, T, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Sobolev source, weak-derivative time-translation row, Rellich–Kondrachov compact-window row, complete-metric readback, real seal, transport, replay, provenance, and local naming data;*
2. *source discipline, requiring every compactness request to enter through the* $\mathsf{Sobolev}^{\uparrow}$ *row S before temporal or spatial compactness data are read;*
3. *time-translation exactness, requiring T to display the derivative bounds, translation windows, and temporal equicontinuity budget used by the compactness route;*
4. *compact-window exactness, requiring R to expose only the inspected Rellich–Kondrachov spatial compactness handoff;*
5. *complete-metric and real readback, requiring M and E to supply only finite Cauchy comparisons, norms, radii, tolerances, and scalar moduli;*
6. *provenance exhaustion, requiring H, C, P, N to transport, replay, source, and name exactly the displayed rows and no selected subsequence, quotient function equality, host compact embedding, ambient completed Sobolev space, global time regularity theorem, unrestricted real completeness, Lean verification, or bridge credential.*

No Aubin-Lions obligation is stored outside these rows.

Proof. Project the accepted packet $A = (S, T, R, M, E, H, C, P, N)$. The carrier definition lists every visible component, so carrier admission is row-local.

The source discipline is forced by S. A compactness read first enters the Sobolev source row and sees only the displayed regularity, weak-derivative ledger, integrability window, and local naming data.

The temporal and spatial compactness rows then split the analytic work. The row T supplies the time-translation budget, while R supplies the Rellich–Kondrachov compact-window handoff. Neither row selects a subsequence or completes a quotient object.

The readback rows M and E expose only Cauchy comparisons and scalar tolerance data. Finally H, C, P, N preserve, replay, source, and name the same finite route, so the listed obligations exhaust the local NameCert surface. □

Theorem 5557.3 (Aubin-Lions compactness handoff)**.** *Every compactness consumer of an accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Sobolev}^{\uparrow} \longmapsto \mathsf{WeakDerivative}^{\uparrow} \longmapsto \mathsf{RellichKondrachov}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

and then through the displayed BHist, hsame, Cont, Pkg, *and* NameCert *rows of the same packet. The consumer receives a finite compactness handoff certificate, not a selected convergent subsequence, quotient function equality, host compact embedding theorem, or ambient completeness theorem.*

Proof. Start with a compactness consumer request. By Definition 5557.2, the request first reads the Sobolev source row S.

The temporal route is then visible through T. It supplies the time-translation windows and derivative bounds needed before the spatial compactness row is read.

The spatial compactness handoff is exactly R. The complete-metric row M and real seal E only read back finite Cauchy comparisons and scalar budgets for that handoff.

The structural rows H, C, P, N keep the route inside the same packet. Since no selected subsequence, quotient equality, host compact embedding, or ambient completeness coordinate appears in the carrier, no consumer can read one from the packet. □

Theorem 5557.4 (Aubin-Lions time-translation nonescape)**.** *Let $A = (S, T, R, M, E, H, C, P, N)$ be an accepted* $\mathsf{AubinLions}^{\uparrow}$ *carrier. Any consumer that keeps the Sobolev source S and the Rellich–Kondrachov compact-window row R but deletes the weak-derivative time-translation row T cannot export the Aubin-Lions compactness handoff from this packet. Such a consumer has only a spatial compact-window read and must not treat it as the time-dependent compactness route named by* $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$.

Proof. Project the carrier of Definition 5557.1. The compactness route contains both the Sobolev source row S and the time-translation row T before the Rellich–Kondrachov compact-window row R is consumed.

Apply Definition 5557.2. The time-translation exactness obligation requires the derivative bounds, temporal difference windows, and equicontinuity budget displayed by T.

Now delete T. The remaining rows may still expose the Sobolev source and a spatial compact-window handoff, but they no longer display the temporal route that distinguishes an Aubin-Lions packet from a static Rellich–Kondrachov read.

The structural rows cannot fill that absence. They transport and replay existing rows only, so without T the consumer cannot export the Aubin-Lions compactness handoff from this packet. □

Falsifiable prediction: Every $\mathsf{AubinLions}^{\uparrow}$ compactness read factors through displayed SobolevUp, WeakDerivativeUp, RellichKondrachovUp, CompleteMetricUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows before any compactness consumer receives it.

Independence witness: sobolev, weak-derivative, rellich-kondrachov, completemetric, real: $\mathsf{AubinLions}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a Sobolev source, time-translation row, compact-window row, complete-metric readback, real scalar seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$, seed)

Scope closed: Definition 5557.1, Definition 5557.2, Definition 5557.3, and Definition 5557.4 seed the finite compactness route over Sobolev, weak-derivative, Rellich–Kondrachov, complete-metric, and real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AubinLions}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host subsequence selection, quotient completion, ambient real completeness, unrestricted compact embedding, global time regularity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source discipline, time-translation exactness, compact-window exactness, complete-metric readback, scalar sealing, componentwise transport, replay, provenance, and local naming for BEDC.Derived.AubinLionsUp.

Constructive story: An $\mathsf{AubinLions}^{\uparrow}$ packet is generated from a Sobolev source row, weak-derivative time-translation row, Rellich–Kondrachov compact-window row, CompleteMetric readback row, Real scalar seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5558

A Concrete Naming Certificate for $\mathsf{CompactUniformPullback}^{\uparrow}$

Following Bishop and Bridges on compact-uniform continuity, the $\mathsf{CompactUniformPullback}^{\uparrow}$ packet realizes the compact-uniform routes of chapter 9237 as a finite pullback carrier joining compact-net data, pointwise-modulus rows, dyadic tolerances, regular rational readbacks, and the terminal real-name seal. It sits after the compact metric carrier of chapter 129, the continuous-map surface of chapter 130, the uniform-modulus consumer of chapter 369, the dyadic rational core of chapter 291, the regular rational sequence route of chapter 301, and the real-name seal of chapter 38. The packet names the recurring compact-uniform handoff without exporting a host modulus selector, quotient point equality, open-cover choice, or ambient Heine–Cantor theorem.

Definition 5558.1 (Compact uniform pullback carrier)**.** *A* $\mathsf{CompactUniformPullback}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, M, D, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row naming the compact source, finite net, located-distance ledger, and coverage witnesses used by the pullback.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the graph, pointwise-modulus ledger, source metric, and target metric.*
3. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row requested by downstream compact-uniform arguments.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records dyadic tolerances, half-radius budgets, and finite lower-bound comparisons used in the compact-net fold.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route for the rational precision histories produced by the fold.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row that exposes the final precision and target estimate only after the dyadic and regular-rational rows are accepted.*
7. *T is the pullback ledger. It records how a target tolerance request is pulled back through F, through compact finite-net rows in K, through the dyadic budget in D, through the regular-rational readback in R, and into the uniform-modulus consumer row M.*
8. *H records componentwise* hsame *transport for compact metric, continuous map, uniform-modulus, dyadic tolerance, regular readback, real seal, pullback, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from every compact-uniform request through* $K, F, M, D, R, E, T, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformPullback}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$ *row.*

The classifier compares accepted pullback packets only through these finite rows. It has no coordinate for a host modulus selector, quotient point equality, open-cover choice, selected Lebesgue number, or ambient compactness theorem.

Lean4 ✓ `BEDC.Derived.CompactUniformPullbackUp.CompactUniformPullbackTasteGate single carrier alignment`

Theorem 5558.2 (Compact uniform pullback NameCert obligations)**.** *For every accepted* CompactUniformPullback *carrier*

$$U = (K, F, M, D, R, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, continuous-map pointwise-modulus admission through F, uniform-modulus consumer exposure through M, dyadic tolerance and lower-bound routing through D, regular-rational readback through R, real sealing through E, pullback ledger visibility through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5558.1. The source rows are K, F, M, the arithmetic and seal rows are D, R, E, the pullback row is T, and the structural rows are H, C, P, N.

The finite-net obligation is read through K, while pointwise modulus rows are read through F. The pullback ledger T is the only row that is allowed to combine those data into a tolerance-only request for M.

The arithmetic handoff is also explicit. Dyadic tolerance and lower-bound comparisons stay in D, regular-rational readback stays in R, and the final real estimate is exposed only through E.

Transport, replay, provenance, and local naming preserve this route. Because the carrier has no coordinate for a selected modulus, quotient point equality, open-cover choice, selected Lebesgue number, or host compactness theorem, the listed rows exhaust the NameCert surface. □

Theorem 5558.3 (Compact uniform pullback modulus handoff)**.** *Every downstream read of an accepted* $\mathsf{CompactUniformPullback}^{\uparrow}$ *packet factors through*

$$K, F, D, R, E, T \longmapsto M.$$

Thus a compact-uniform consumer receives a $\mathsf{UniformModulus}^{\uparrow}$ *row only after the compact finite-net data, pointwise-modulus rows, dyadic tolerance budget, regular-rational readback, real seal, and pullback ledger have all been displayed inside the same finite carrier.*

Proof. Begin with a downstream target tolerance request. The continuous-map row F supplies the pointwise-modulus rows inspected at compact-net centers, and the compact metric row K supplies the finite net and coverage witnesses used to choose those centers inside BHist.

The dyadic tolerance row D performs the finite lower-bound comparisons. Its output is read through R as a regular rational precision history and then through E as a real-sealed estimate.

The pullback ledger T records that these rows are consumed in this order before the consumer row M is exposed. Componentwise transport and replay can only preserve the displayed route, so no host modulus selector or open-cover choice is exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformPullback}^{\uparrow}$ modulus read factors through a CompactMetricUp finite-net row, a ContinuousMapUp pointwise-modulus row, a UniformModulusUp consumer row, DyadicRatCoreUp tolerance arithmetic, RegSeqRatUp readback, a RealUp seal, a pullback ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: compactmetric, continuousmap, uniformmodulus, dyadicratcore, regseqrat, real: $\mathsf{CompactUniformPullback}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact finite-net coverage, pointwise-modulus rows, dyadic tolerance budgets, regular rational readback, real sealing, a downstream uniform-modulus consumer, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformPullback}^{\uparrow}$ packet is generated from CompactMetricUp finite-net rows, ContinuousMapUp pointwise-modulus rows, a UniformModulusUp consumer row, DyadicRatCoreUp tolerance arithmetic, RegSeqRatUp readback, a RealUp seal, a compact-uniform pullback ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5558.1, Definition 5558.2, and Definition 5558.3 seed the finite compact-net, pointwise-modulus, dyadic-tolerance, regular-readback, and real-seal handoff to uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformPullback}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host modulus selection, quotient point equality, open-cover choice, selected Lebesgue numbers, ambient Heine–Cantor theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, pointwise-modulus admission, dyadic tolerance exactness, regular readback, real sealing, pullback ledger visibility, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformPullbackUp.

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, and the terminal Real seal of chapter 38 before routing the child files below.

Chapter 5559

A Concrete Naming Certificate for $\mathsf{RegularCauchyGluing}^{\uparrow}$

Bishop–Bridges style regular Cauchy gluing records the finite route by which two compatible local regular rational windows can be read as one BEDC real-name boundary without passing through quotient representatives or an ambient completeness theorem. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the reindexing route of chapter 1568.

Definition 5559.1 (Regular Cauchy gluing carrier)**.** *A* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right compatible regular Cauchy windows, each carrying finite* $\mathsf{StreamName}^{\uparrow}$ *observations and a local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger recording the shared dyadic radii, endpoint comparisons, overlap budgets, and refinement witnesses used by both windows.*
3. *S is the joint* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing exactly the finite indices inspected on the left window, right window, and overlap.*
4. *Q is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by reading the two local regular rows through the common ledger D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, R, D, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for the two windows, common dyadic ledger, finite observations, glued readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite gluing route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the two local windows are compatible on the displayed overlap through D, S, Q, H, C, P, N. It has no coordinate for quotient sequence equality, selected representatives, host real equality, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5559.2 (Regular Cauchy gluing classifier)**.** *The* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *classifier compares accepted carriers only through the tuple* $L, R, D, S, Q, E, H, C, P, N$*. Two packets are indistinguishable for this classifier precisely when the left and right window rows, the common dyadic refinement ledger, the finite observation row, the glued regular readback, the terminal real seal, and the transport, replay, provenance, and naming rows are componentwise the same under the displayed* hsame *routes.*

Theorem 5559.3 (Regular Cauchy gluing NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *surface consists exactly of carrier admission for the two compatible regular Cauchy windows, common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through* S*, glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The analytic rows before the seal are exactly the two regular windows L, R, the common dyadic ledger D, the finite observation row S, and the glued regular readback Q.

The overlap obligation is finite. Its comparisons are stored in D and read through the same inspected indices in S, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ rows feed one glued readback row Q without adding a quotient representative.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections gives the listed $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ surface, and the carrier definition leaves no hidden row for selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 5559.4 (Regular Cauchy gluing overlap stability)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Any refinement of the displayed overlap that is transported by* H *and replayed by* C *preserves the same glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Consequently downstream Real and Completion consumers receive only the finite gluing packet and not quotient stream equality, selected representatives, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the displayed overlap ledger. The common dyadic row D records every radius and endpoint comparison used by both local windows, while S records the finite observations on which those comparisons are made.

Apply the classifier of Definition 5899.2. A refinement transported by H and replayed by C can change neither the componentwise ledger coordinates nor the finite observation coordinates exposed to the classifier.

The glued readback row Q is therefore the same row after refinement, and the $\mathsf{Real}^{\uparrow}$ seal E is read only through Q together with H, C, P, N. Since the carrier contains no quotient, selector, host-equality, choice, or completeness coordinate, none of those data can be exported by overlap stability. □

Theorem 5559.5 (Regular Cauchy gluing common-refinement exactness)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Every glued readback exported from* G *must expose the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The exactness route is transported by* H *and replayed by* C*, and it exports no quotient stream equality, selected representative, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted carrier. The row D is the common dyadic refinement ledger shared by the two local regular Cauchy windows, while Q is the glued regular readback and E is the terminal Real seal.

Apply overlap stability from Definition 5899.4. Any accepted refinement of the overlap preserves the same Q and E only through the displayed ledger and observation rows, so a glued readback cannot bypass D.

Finally use the Cauchy-reindexing handoff of Definition 5899.7. It reads D, S, Q as the finite reindexing-compatible route before any Real-facing consumer reaches E, with H, C, P, N preserving and naming the same route. □

Theorem 5559.6 (Regular Cauchy gluing overlap induction)**.** *Let $G = (L, R, D, S, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. For any finite chain of displayed overlap refinements transported by H and replayed by C, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E remain unchanged. The resulting route still factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyReindexing}^{\uparrow}$, *and it exports no quotient stream equality, selected representatives, host equality, countable choice, or ambient completeness.*

Proof. Induct on the displayed finite refinement list.

The empty list is the overlap-stability theorem of Definition 5899.4: the common dyadic ledger D and the finite observation row S already determine the same glued readback Q and seal E.

For the step, assume the earlier refinements preserve Q and E. Compose the next displayed refinement with the induction hypothesis, transport the component rows by H, and replay the same finite route by C.

Apply the classifier of Definition 5899.2 componentwise. Since the classifier reads only $L, R, D, S, Q, E, H, C, P, N$ the next refinement cannot add a quotient, representative selector, host equality row, choice schedule, or completeness witness; it only repeats the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5559.7 (Regular Cauchy gluing Cauchy-reindexing handoff)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier factors its gluing route through the finite* $\mathsf{CauchyReindexing}^{\uparrow}$ *surface recorded in P. The handoff exposes the common dyadic refinement ledger D, the finite observation row S, and the glued regular readback row Q as a reindexing-compatible route, and it exposes no global quotient of regular Cauchy names or ambient completeness principle.*

Proof. Read the package provenance P. It names the route through $\mathsf{CauchyReindexing}^{\uparrow}$ together with $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The carrier rows D, S, Q are exactly the finite data consumed by that route: D supplies the common refinement ledger, S supplies the inspected windows, and Q supplies the regular rational readback after the two local windows agree on the overlap.

The support rows H, C, N transport, replay, and name this same finite path. Thus the handoff is reindexing-compatible BEDC data, not a quotient of regular names, a selected representative, or a completeness theorem. □

Theorem 5559.8 (Regular Cauchy gluing obligation closure handoff)**.** *Every obligation-closure read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier must expose the full row tuple*

$$L, R, D, S, Q, E, H, C, P, N.$$

The route factors through the common $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and glued readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and*

the $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff named by* P *before any downstream real-name consumer may read the gluing packet.*

Proof. Start with the carrier and NameCert obligations. The accepted packet exposes $L, R, D, S, Q, E, H, C, P, N$, and the obligation surface has no smaller projection that still records the two local windows, their common refinement, and the glued readback.

Use the Cauchy-reindexing handoff of Definition 5899.7. It forces the gluing read through D, S, Q and through the provenance row P naming $\mathsf{CauchyReindexing}^{\uparrow}$, while the carrier rows L, R provide the two regular windows.

The rows E, H, C, N then seal, transport, replay, and name the same finite route. Thus obligation closure reads exactly the displayed tuple and factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5559.9 (Regular Cauchy gluing Real seal ordering)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reachable only after the common dyadic refinement row* D*, the finite observation row* S*, and the glued readback row* Q *have been displayed. The seal-ordering route asserts no quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Project the carrier definition. The row E is listed as the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after the two local windows, the common refinement ledger, the finite observations, and the glued readback have been displayed.

Apply the overlap-stability and Cauchy-reindexing handoff theorems. They keep the common dyadic refinement in D, the inspected windows in S, and the glued regular readback in Q visible before any real-name consumer can inspect E.

Since quotient stream equality, selected representatives, host equality, choice schedules, and ambient completeness are not carrier coordinates, the ordering statement cannot export them. It records only the displayed route from D, S, Q to the terminal seal E. □

Theorem 5559.10 (Regular Cauchy gluing local tail compatibility)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the two local regular-Cauchy windows* L *and* R *can be consumed by a* $\mathsf{Real}^{\uparrow}$*-seal reader only after the overlap row, the common-refinement row, and the Cauchy-reindexing row exhibit a shared* $\mathsf{RegSeqRat}^{\uparrow}$ *tail boundary. The compatibility route is the finite route through* D, S, Q, H, C, P, N *supplied by the carrier, and it exports no quotient stream equality, selected representative, or ambient completeness.*

Proof. Project the carrier of Definition 5899.1. The two local windows L, R are not read independently by a Real-facing consumer; the packet first displays their shared dyadic ledger D, finite observations S, and glued readback Q.

Apply overlap stability, Definition 5899.4, and common-refinement exactness, Definition 5899.5. These theorems keep the overlap row and the common refinement visible before Q is consumed, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ tails are compared at the displayed boundary rather than by host stream equality.

Apply overlap induction, Definition 5899.6, to finite chains of displayed refinements, and then use the Cauchy-reindexing handoff, Definition 5899.7. The resulting route makes the shared tail boundary stable under H and replayable by C before E is read.

Since quotient stream equality, selected representatives, and ambient completeness are absent from Definition 5899.1, the compatibility theorem can expose only the finite D, S, Q, H, C, P, N route. □

The local-tail compatibility row is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window surface of chapter 301 and the $\mathsf{StreamName}^{\uparrow}$ finite-observation surface of chapter 126. The exact dependency is the common-tail rows displayed by D and Q, together with finite-window readback through S, before the $\mathsf{Real}^{\uparrow}$ seal E is available to any downstream consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet is generated from two compatible regular Cauchy windows, a common $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ observations, a glued $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5899.1, Definition 5899.2, Definition 5899.3, Definition 5899.4, Definition 5899.6, Definition 5899.7, Definition 5899.8, Definition 5899.9, and Definition 5899.10 seed the finite gluing route for compatible regular Cauchy windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, common dyadic-refinement exactness, overlap stability, CauchyReindexing handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyGluingUp.

Theorem 5559.11 (Regular Cauchy gluing L10 sibling route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet reads its local gluing route through the L10 sibling lattice before it exports a NameCert row. The route first consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the Cauchy reindexing handoff of chapter 1568.*

Thus the gluing packet exports only a local finite NameCert route over the displayed dyadic, stream, regular-rational, Real-seal, and reindexing rows. It does not expose quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 5899.1. Its visible rows are two local regular windows, the common dyadic ledger, the finite observation row, the glued regular readback, the Real seal, and the transport, replay, provenance, and naming rows.

Apply Definition 5899.3. The local NameCert surface must read exactly those rows, with the dyadic ledger, StreamName windows, RegSeqRat readback, and Real seal appearing before the certificate is exported.

Use Definition 5899.7 for the reindexing component. It identifies the same dyadic, stream, and regular readback rows as a $\mathsf{CauchyReindexing}^{\uparrow}$-compatible route.

Thus the dependency on chapter 1568 is consumed before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal are available to the gluing certificate.

The exclusions are inherited from the carrier inventory and the overlap stability theorem. Since no coordinate stores quotient stream equality, selected representatives, host real equality, choice, or ambient completeness, the L10 sibling route remains a finite local NameCert route. □

Theorem 5559.12 (Regular Cauchy gluing RegSeqRat route)**.** *Every glued readback row of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must pass through the existing* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling surface of chapter 301 and the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff of chapter 1568 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The local obligations of Definition 5899.3 therefore expose a regular rational readback and a reindexing route before any Real-facing consumer may read the glued value.*

Proof. Start from the local L10 route of Definition 5899.11. It already orders the dyadic ledger, finite StreamName windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyReindexing}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal before the gluing NameCert row is exported.

Apply the local obligations of Definition 5899.3. The glued readback row is one of the displayed certificate rows, so it cannot be consumed without the same regular rational surface and structural support rows.

Use Definition 5899.7. The reindexing handoff keeps the dyadic, stream, and regular readback data aligned before the terminal Real seal appears. Thus a Real-facing read factors through RegSeqRat and CauchyReindexing rather than bypassing them. □

Theorem 5559.13 (Regular Cauchy gluing pasting nonescape)**.** *Every accepted pasting consumer of a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite route of Definition 5899.1, satisfy the local obligations of Definition 5899.3, and pass through the L10 sibling route of Definition 5899.11 before it exports any NameCert row. In particular, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently no such pasting consumer exports quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as a coordinate of the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *read.*

Proof. Project the accepted carrier in Definition 5899.1. Its displayed route already contains the common dyadic ledger, finite StreamName observations, glued RegSeqRat readback, Real seal, and the support rows for transport, replay, provenance, and naming.

Apply Definition 5899.3. A pasting consumer can export a local NameCert row only by preserving exactly those displayed rows, so the consumer has no separate entry point around the dyadic, stream, regular readback, or seal coordinates.

Apply Definition 5899.11. The L10 route orders the same rows through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before export.

The refused objects are absent from the carrier and absent from the sibling route. Thus quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness cannot be repacked as a pasting consumer of the accepted $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet. □

Theorem 5559.14 (Regular Cauchy gluing separator-use route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that exports an apartness read or a located-comparison read must first pass through the* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *route of chapter 7930. Ordinary gluing consumers still export only the finite dyadic, stream,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows named by Definition 5899.11.*

Proof. Start with the L10 sibling route of Definition 5899.11. It exposes the dyadic tolerance rows, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before the local gluing NameCert row is exported.

For a consumer that reads only gluing data, those rows exhaust the available surface by Definition 5899.13. The carrier has no coordinate for apartness, located comparison, quotient equality, selected representatives, choice, or ambient completeness.

If the same consumer asks for apartness or located comparison, apply the separator sibling dependency of Definition 7930.3. That route requires the stream, regular-rational, dyadic, Real-seal, located-cut, and apartness-field rows to be displayed through $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ before such a read is available.

Thus the gluing packet itself remains the ordinary finite route, while apartness-facing or located-comparison-facing reads factor through the separator chapter rather than being smuggled into the gluing carrier. □

Theorem 5559.15 (Regular Cauchy gluing completion consumer route)**.** *Every Bishop-completion or Real-completion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the same finite L10 sibling route before exporting a gluing* $\mathsf{NameCert}$ *row. In particular, the consumer first exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The exported gluing certificate has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already orders the dyadic ledger, stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any local gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12. A completion-facing read of the glued value therefore cannot bypass the regular rational surface or the reindexing route on its way to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply the pasting nonescape theorem, Definition 5899.13. It keeps every gluing consumer inside the finite carrier route and excludes quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness as exportable coordinates.

Finally use the separator-use route of Definition 5899.14. Apartness-facing or located-comparison-facing completion reads factor through the separator sibling; ordinary Bishop-completion and Real-completion gluing reads still expose only the dyadic, stream, regular-rational, reindexing, and Real-seal rows before the gluing $\mathsf{NameCert}$ is named. □

Theorem 5559.16 (Regular Cauchy gluing uniform Cauchy criterion route)**.** *Every uniform-Cauchy-criterion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite sibling route before exporting criterion data. The consumer must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of the gluing route.*

Consequently the criterion consumer exports only the finite BEDC packet obtained from those rows, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *ledger. It does not export quotient sequence equality, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already places the dyadic tolerance ledger, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12 and the completion consumer theorem Definition 5899.15. Together they force a completion-facing criterion read through the regular rational surface, the reindexing handoff, and the Bishop-completion consumer rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally use the downstream sibling route of Definition 5899.18. It permits only concrete downstream consumers after the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows have been displayed. Since the carrier and all cited routes omit quotient sequence equality, selected representatives, countable choice, host real equality, and ambient completeness, none of those objects can be repacked as uniform-Cauchy-criterion data. □

Theorem 5559.17 (Regular Cauchy gluing L10 sibling dependency)**.** *Before any gluing consumer reads the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row, it must display the finite sibling dependency route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *observation windows of chapter 126,*

the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the completion-facing handoff of chapter 3665, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route may also be read by Bishop-completion consumers through the completion consumer theorem Definition 5899.15, but it remains a finite dependency lattice rather than quotient sequence equality, selected representatives, countable choice, host real equality, or ambient real completeness.*

Proof. Start with Definition 5899.11. It already orders the dyadic tolerance rows, stream observation windows, regular-rational readback, Cauchy reindexing handoff, and Real seal before the local $\mathsf{NameCert}$ row is exported.

Apply Definition 5899.15. A Bishop-completion or Real-completion consumer therefore reads the same finite row sequence before it can use the gluing packet as completion-facing data.

The dependency is finite because every cited sibling appears as a displayed row or handoff in the local route. No step asks for quotient sequence equality, a selected representative of a Cauchy class, countable choice over tails, host real equality, or an ambient completeness theorem. □

Theorem 5559.18 (Regular Cauchy gluing downstream sibling route)**.** *Any cofinal, interleaving, or comparison-facing consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first pass through the finite sibling route of Definition 5899.17. After the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *rows have been displayed, such a consumer may continue only through a concrete downstream sibling such as chapter 414 or chapter 3514. The gluing packet itself exports no detached cofinal route, no interleaving selector, no quotient equality of Cauchy names, no selected representative, and no ambient completeness theorem.*

Proof. Start with Definition 5899.17. It forces every gluing consumer to display the regular rational rows, finite stream windows, dyadic tolerance rows, Bishop-completion handoff, and terminal Real seal before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ row is read.

Apply the completion consumer theorem Definition 5899.15. A completion-facing read therefore receives the same finite route, not a quotient stream or selected representative.

For an interleaving-facing read, use the public sibling surface of chapter 414. It supplies the alternating selector and combined modulus ledger only after the relevant $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are displayed.

For a comparison-facing read, use chapter 3514. That packet supplies the even–odd comparison and shared threshold rows as a separate finite consumer surface. Thus any cofinal or interleaving continuation is downstream of the existing L10 sibling dependency, and none of those rows is smuggled into the gluing carrier itself. □

Theorem 5559.19 (Regular Cauchy gluing tail-agreement composition)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads a glued* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet through Definition 5899.18 must continue through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699 before treating the glued readback as completion-facing tail agreement. The composed route also reads the Cauchy limit seal of chapter 386; the gluing packet alone exports no ambient tail equality, selected representative, quotient sequence equality, or completeness witness.*

Proof. Start with Definition 5899.18. It permits downstream consumers only after the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows have been displayed.

A completion-facing tail read then has an additional boundary. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ chapter supplies the route that turns a Real-facing readback into displayed tail-agreement data, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter supplies the limit-facing seal used by completion consumers.

Thus the row order is finite: first the local gluing sibling route, then the tail-agreement seal, and finally the Cauchy-limit seal. Because each step only reuses displayed rows, no consumer can replace the route by host tail equality, a selected Cauchy representative, quotient sequence equality, or ambient completeness. □

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes its L10 sibling lattice before any gluing read is exported: the $\mathsf{StreamName}^{\uparrow}$ finite windows provide the inspected overlap rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular readback, $\mathsf{DyadicRatCore}^{\uparrow}$ carries the shared tolerance ledger, $\mathsf{Real}^{\uparrow}$ seals the terminal value, and chapter 5544 provides the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ schedule that keeps the two local Cauchy windows on one displayed budget.

Theorem 5559.20 (Regular Cauchy gluing L10 forward route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer reaches Real-facing gluing only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of the local carrier. The route first reads the name certificate construction, then the L10 sibling route, and finally the Bishop-completion consumer route before any gluing conclusion is exported. It refuses quotient stream equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as coordinates of the gluing read.*

Proof. Project the local carrier and obligations from Definition 5899.1 and Definition 5899.3. The rows available to a consumer are the two compatible windows, the common dyadic ledger, finite observations, glued regular readback, Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the consumer through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a local gluing certificate is exported.

Apply the Bishop-completion consumer theorem Definition 5899.22. A Real-facing or Bishop-completion consumer must display the same D, S, Q, E, H, C, P, N route, so the forward path remains finite and local to the accepted packet.

The refused objects are absent from every cited step. The name certificate construction supplies no quotient stream equality; the L10 sibling route supplies no selected limit or countable-choice schedule; and the Bishop-completion consumer route supplies no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Hence every accepted forward gluing read factors through the displayed windows, readbacks, tolerance ledgers, and Real seal. □

Theorem 5559.21 (Regular Cauchy gluing sibling forward link)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that is read as a gluing consumer first passes through the L10 sibling route of Definition 5899.11. In particular, the consumer must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126 before the local gluing* $\mathsf{NameCert}$ *row is available.*

The forward link is therefore a concrete sibling dependency route rather than a detached hub edge. It also preserves the exclusions already named by the L10 route: no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is exported by the gluing read.*

Proof. Start with the carrier and obligation rows of Definition 5899.1 and Definition 5899.3. They display the paired local windows, common dyadic ledger, finite observations, glued regular readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Apply the L10 sibling route of Definition 5899.11. That route orders the same gluing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any gluing certificate can be exported.

Thus the hub has a visible child route into the RegSeqRat and StreamName sibling chapters. The refused objects remain absent because neither the local carrier nor the L10 sibling route contains coordinates for quotient stream equality, selected representatives, choice, host equality, or ambient completeness. □

Theorem 5559.22 (Regular Cauchy gluing Bishop completion consumer)**.** *Every* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first expose the same finite gluing carrier rows already displayed by Definition 5899.1. In particular, before the Bishop completion route may read the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *glued readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff recorded in* P*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}$ *row* N*.*

Thus the Bishop completion consumer receives the gluing packet only as a finite L10 route over dyadic, stream, regular-rational, Real-seal, reindexing, transport, replay, provenance, and naming data. It receives no quotient stream equality, selected representative, host real equality, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The rows that a completion-facing consumer can inspect are exactly the two local windows, the common dyadic ledger, the finite observations, the glued readback, the Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the gluing packet through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$ before a NameCert row is exported.

Now read the Bishop completion carrier of chapter 3665. Its completion-facing route consumes dyadic precision, finite stream windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming. The gluing rows D, S, Q, E, H, C, P, N are therefore the only rows that can feed such a consumer from $\mathsf{RegularCauchyGluing}^{\uparrow}$.

The exclusions follow from the same carrier inventory. Since neither Definition 5899.1 nor the Bishop completion route supplies quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness as a coordinate, the consumer cannot export any of those objects from the gluing packet. □

Theorem 5559.23 (Regular Cauchy gluing Real-tail forward route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that asks for a tail-agreement seal is a forward dependency of the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route in chapter 699. Concretely, the gluing packet first exposes the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the paired* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the local* $\mathsf{Real}^{\uparrow}$ *seal* E*; only then may the consumer enter the Real-tail agreement and Cauchy-limit seal route through* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*.*

Proof. Project the accepted gluing carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The Real-facing rows available before any tail agreement are exactly the common dyadic ledger D, finite observation row S, glued regular readback Q, terminal seal E, transport row H, replay row C, provenance row P, and local name row N.

Apply the Real seal ordering theorem Definition 5899.9. It forces the route to display D, S, Q before E is readable, so a consumer cannot jump from the two local windows directly to a Real-tail agreement row.

Now use the Real-tail agreement completion-consumer boundary of chapter 699. That boundary receives paired windows, dyadic readback, agreement, Cauchy-limit sealing, and Real or Com-

pletion endpoints as one finite route. Feeding it from $\mathsf{RegularCauchyGluing}^{\uparrow}$ therefore makes the gluing read a forward dependency of that Real-tail route rather than a separate kernel island.

The exclusions are inherited from the two carriers. Neither Definition 5899.1 nor the Real-tail agreement route supplies quotient stream equality, selected representatives, host convergence, countable choice, or ambient completeness as a row, so no such object can be exported by this forward dependency. □

Chapter 5560

A Concrete Naming Certificate for $\mathsf{YoungSymmetrizer}^{\uparrow}$

Following Fulton and Harris on Young symmetrizers and polynomial representations, the $\mathsf{YoungSymmetrizer}^{\uparrow}$ packet records the finite tableau rows that turn a partition shape into row symmetrization, column antisymmetrization, and a displayed tensor-to-matrix action. It sits between the shape discipline of chapter 201, the character readback of chapter 245, the Grothendieck consumer of chapter 246, the multilinear source of chapter 87, and the displayed action rows of chapter 49. The packet is finite: it names the symmetrizer route without importing semisimplicity, Specht module irreducibility, Schur functor exactness, host group algebra idempotence, or quotient representation categories.

Definition 5560.1 (YoungSymmetrizerUp carrier)**.** *A* $\mathsf{YoungSymmetrizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N).$$

Here P is the consumed $\mathsf{Partition}^{\uparrow}$ *shape row; D is the Young diagram row for that shape; T is a tableau row whose entries fill the boxes of D; R is the row-stabilizer ledger listing the permutations that preserve each row of T; C is the column-stabilizer ledger listing the permutations that preserve each column of T; S is the row-symmetrizer formal sum over R; A is the column-antisymmetrizer formal signed sum over C; E is the ordered composite row for the formal Young symmetrizer SA; M is the displayed tensor and matrix action row over the* $\mathsf{TensorProduct}^{\uparrow}$ *and* $\mathsf{Mat}^{\uparrow}$ *sources; H gives componentwise* hsame *transport for $P, D, T, R, C, S, A, E, M$; U gives displayed* Cont *replay for the same finite route; V is the consumer ledger for* $\mathsf{CharacterTheory}^{\uparrow}$ *and* $\mathsf{RepresentationRing}^{\uparrow}$ *readback; G is the* Pkg *provenance over* $\mathsf{Partition}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*; and N is the local* $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}$ *row.*

Theorem 5560.2 (YoungSymmetrizerUp NameCert obligations)**.** *For every accepted carrier $Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N)$ of Definition 5560.1, the local* $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

- *shape admission, requiring P to expose the partition shape and D to list precisely its diagram boxes;*
- *tableau admission, requiring T to fill the boxes of D and to induce the row and column incidence ledgers used by R and C;*
- *row symmetrizer exactness, requiring S to be the finite formal sum over the row-stabilizer ledger R;*

- *column antisymmetrizer exactness, requiring A to be the finite signed formal sum over the column-stabilizer ledger C;*
- *action display, requiring E and M to route the composite SA through the tensor and matrix rows rather than through an ambient group-algebra object;*
- *consumer confinement, requiring every* $\mathsf{CharacterTheory}^{\uparrow}$ *or* $\mathsf{RepresentationRing}^{\uparrow}$ *read to pass through V after the finite tensor/matrix action has been displayed;*
- *structural replay, requiring H, U, G, N to transport, replay, account for, and name exactly the rows already listed.*

No accepted $\mathsf{YoungSymmetrizer}^{\uparrow}$ *packet exports an irreducible module theorem, a complete character formula, a semisimple decomposition, or equality inside an unlisted host group algebra.*

Lean4 ✓ `BEDC.Derived.YoungSymmetrizerUp.YoungSymmetrizerTasteGate single carrier alignment`

Proof. Inspect the coordinates of Y. The shape and tableau rows P, D, T determine the finite incidence data, while R and C are only the displayed stabilizer ledgers for that incidence data.

The formal rows S, A, E are then determined by finite replay over R and C. Their only action surface is the displayed tensor/matrix row M, so a consumer cannot replace the route by an ambient group-algebra calculation without leaving the packet.

Finally H, U, V, G, N do not add new mathematical rows. They transport, replay, expose, account for, and name the finite route already visible in $P, D, T, R, C, S, A, E, M$. □

Theorem 5560.3 (YoungSymmetrizerUp idempotent route)**.** *Let $Y = (P, D, T, R, C, S, A, E, M, H, U, V, G, N)$ be an accepted* $\mathsf{YoungSymmetrizer}^{\uparrow}$ *carrier. Any downstream route that reads the Young-symmetrizer idempotent behavior from Y must factor through the displayed finite chain*

$$P, D, T \longrightarrow R, C \longrightarrow S, A \longrightarrow E \longrightarrow M \longrightarrow V, N.$$

The route may expose the row symmetrizer, the column antisymmetrizer, and the tensor/matrix action as a finite certificate surface; it does not claim normalized primitive idempotence, irreducible Specht image, Schur functor exactness, or a complete representation-ring decomposition.

Proof. Start with the displayed source chain. The partition and tableau rows are the only source of row and column incidence, so every stabilizer read reaches R or C through P, D, T.

The symmetrizer and antisymmetrizer rows are formal finite sums over those stabilizers. Thus the composite row E is available only after S and A have been displayed.

The action and consumer boundary are then forced by M and V. The packet contains no row for normalized scalar inversion, quotient module projection, or semisimplicity, so the idempotent behavior remains a finite route claim over the displayed action rather than a host-level representation-theoretic closure theorem. □

Falsifiable prediction: Every future $\mathsf{YoungSymmetrizer}^{\uparrow}$ Lean carrier can be audited by counting exactly one partition shape row, one tableau row, two stabilizer ledgers, two formal-sum rows, one composite row, one tensor/matrix action row, and one local $\mathsf{NameCert}$ row before any $\mathsf{RepresentationRing}^{\uparrow}$ consumer is admitted.

Independence witness: partition, charactertheory, representationring, tensorproduct, matrix: YoungSymmetrizerUp is not bijective to any listed sibling because it jointly carries tableau incidence, row and column stabilizer ledgers, row symmetrization, column antisymmetrization, a composite tensor/matrix action, consumer confinement, and local NameCert provenance.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungSymmetrizer}^{\uparrow}$ packet is generated from one partition shape row, one Young diagram row, one tableau incidence row, row and column stabilizer ledgers, row-symmetrizer and column-antisymmetrizer formal-sum rows, one composite action row, displayed tensor/matrix action support, consumer confinement, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5560.1, Definition 5560.2, and Definition 5560.3 bind the finite tableau, stabilizer, formal-sum, action, and consumer surface for the Young-symmetrizer packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungSymmetrizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove primitive idempotence, irreducible Specht images, Schur functor exactness, semisimplicity, a complete character formula, host group-algebra equality, Lean verification, or a bridge-checked public schema.

Upgrade path: Obligation closure requires checked carrier admission, tableau incidence exposure, stabilizer-ledger exactness, finite formal-sum exactness, tensor/matrix action confinement, consumer non-escape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.YoungSymmetrizerUp.

Chapter 5561

A Concrete Naming Certificate for $\mathsf{MetaCICSRKernel}^{\uparrow}$

The $\mathsf{MetaCICSRKernel}^{\uparrow}$ packet realizes the subject-reduction discharge branch of chapter 9374 as one finite kernel carrier. It sits between the open-phase audit of $\mathsf{MetaCICCriticalPath}^{\uparrow}$ in chapter 765, the four-face obligation packet $\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}$ in chapter 712, the route socket $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ in chapter 719, the closed-substitution boundary $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ in chapter 527, and the candidate/endpoint frontier $\mathsf{MetaCICNormalizationFrontier}$ in chapter 892. The packet gives later formalization a concrete finite place to discharge typing-preservation routes without replacing the four discharge faces by a parameter echo.

Theorem 5561.1 (MetacicSrKernelUp carrier)**.** *A* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M).$$

Here C is the $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *source ledger; O is the* $\mathsf{MetaCICSubjectReductionObligation}^{\uparrow}$ *four-face obligation row; S is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *route row; B, A, L, P are respectively the beta-substitution, application-argument, lambda-domain, and Pi-domain discharge rows consumed through O and S; T is the* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$ *closed-term projection row; N is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *candidate/endpoint boundary row; F is the finite subject-reduction handoff row that exposes exactly the four discharge faces before any type-preservation consumer reads the route; H is the componentwise* hsame *transport row; R is displayed* Cont *replay for the kernel route; G is the* Pkg *provenance over the named MetaCIC packets,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*; and M is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}^{\uparrow}}$ *row.*

Proof. Read the carrier as a finite route ledger. The coordinates C, O, S, T, N locate the existing MetaCIC boundaries, while B, A, L, P are exactly the four discharge rows already named by the obligation and socket packets.

The row F is not a fifth discharge hypothesis. It is the finite handoff that requires a consumer to see all four displayed faces before any subject-reduction read is exported.

The structural rows H, R, G, M only transport, replay, account for, and name the same finite kernel. Thus the carrier is a finite NameCert packet rather than a parametric schema over fresh carriers. □

Theorem 5561.2 (MetacicSrKernelUp NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *carrier $K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}^{\uparrow}}$ *surface consists exactly of the following rows:*

- *critical-path admission through C;*
- *obligation admission through O, with beta-substitution, application-argument, lambda-domain, and Pi-domain faces exposed as B, A, L, P;*
- *socket compatibility through S, requiring the same four faces and forbidding an erased closed-subject-reduction row;*
- *closed-term boundary compatibility through T;*
- *normalization-frontier compatibility through N;*
- *subject-reduction handoff through F, which admits a consumer only after B, A, L, P are all displayed;*
- *transport, replay, provenance, and local naming through H, R, G, M.*

No obligation row in this surface may be replaced by a universal carrier parameter, a host definitional-equality principle, a quotient reduction equality, or a hidden closed-consistency theorem.

Proof. The obligation surface is obtained by projecting the coordinates of K. The packet imports the four subject-reduction faces from O and S, so the local naming row cannot omit one face while still claiming to expose the discharge surface.

The boundary rows T and N constrain where closed substitutions and normalization-frontier reads may occur. They do not prove subject reduction by themselves.

The handoff row F is therefore the only public route from the four faces to a subject-reduction consumer. Since H, R, G, M only preserve the displayed route, the packet has no coordinate where an ambient closed theorem or parameter-echo schema can enter. □

Theorem 5561.3 (MetacicSrKernelUp discharge route)**.** *Every accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *packet routes subject-reduction discharge through the finite chain*

$$C \longrightarrow O, S \longrightarrow B, A, L, P \longrightarrow T, N \longrightarrow F \longrightarrow M.$$

A type-preservation consumer may read the route only after the four discharge faces have been displayed and transported through H, R, G, M. The packet does not export closed subject reduction for the present substrate, unrestricted dependent-codomain preservation, erased substitution stability, global normalization, or closed consistency.

Proof. Start at the critical-path ledger C. The next admissible rows are the obligation and socket rows O, S, because they name the discharge faces and the obstruction boundary.

Project B, A, L, P from those rows. The closed-term and normalization-frontier coordinates T, N then constrain the route to the closed-substitution and candidate/endpoint surfaces already present in the neighboring packets.

Finally F packages the visible discharge for a consumer, while H, R, G, M preserve and name the route. No row in the carrier provides a replacement theorem, so the discharge remains a finite kernel route. □

Theorem 5561.4 (MetacicSrKernelUp closed-substitution discharge)**.** *Let $K = (C, O, S, B, A, L, P, T, N, F, H, R,$ be an accepted* $\mathsf{MetaCICSRKernel}^{\uparrow}$ *packet. A subject-reduction consumer may use the beta, application-argument, lambda-domain, and Pi-domain discharge rows B, A, L, P only through the same closed-term projection row T and normalization-frontier row N before the finite handoff F is read. Replacing T by a host substitution theorem, quotient reduction equality, or hidden closed-consistency theorem leaves the accepted carrier.*

Proof. Project the carrier of Definition 5561.1. The four discharge rows B, A, L, P are displayed only after the critical-path, obligation, and socket rows C, O, S.

Apply Definition 5561.2. It exposes the four discharge faces through the same obligation and socket surface and records compatibility with the closed-term boundary T and normalization-frontier boundary N.

Apply Definition 5561.3. The route forces

$$C \longrightarrow O, S \longrightarrow B, A, L, P \longrightarrow T, N \longrightarrow F \longrightarrow M$$

before a subject-reduction consumer can read the finite handoff.

The structural rows H, R, G, M transport, replay, source, and name that route. Since the carrier has no coordinate for a host substitution theorem, quotient reduction equality, or hidden closed-consistency theorem, replacing T by any such object exits the accepted packet. □

Theorem 5561.5 (MetacicSrKernelUp four-face totality)**.** *Let $K = (C, O, S, B, A, L, P, T, N, F, H, R, G, M)$ be an accepted* MetaCICSRKernel$^{\uparrow}$ *packet. The finite subject-reduction handoff row F is readable only after the beta-substitution row B, application-argument row A, lambda-domain row L, and Pi-domain row P have all been displayed through the shared obligation row O and route-socket row S. No packet that omits any one of B, A, L, P exports the subject-reduction handoff of K, and no three-face route can bypass the missing face through the closed-term boundary T, normalization-frontier boundary N, or handoff row F.*

Proof. Project the accepted carrier of Definition 5561.1. The only discharge-face coordinates available to a consumer are B, A, L, P, and the only rows that display them to the kernel route are O and S.

Apply Definition 5561.2. It lists the four faces as one obligation surface: beta-substitution, application-argument, lambda-domain, and Pi-domain exposure must all be present before the local naming row admits the handoff.

Now apply Definition 5561.3. The route orders the carrier read through C, then O, S, then the four discharge faces, then the boundary rows T, N, and only then the handoff row F and local name row M.

Finally use Definition 5561.4. It prevents a consumer from replacing the closed-term boundary by a host substitution theorem, quotient reduction equality, or hidden consistency result. Since T, N, F are boundary and handoff rows rather than substitute discharge faces, a three-face packet has no coordinate from which to export the accepted handoff. □

Falsifiable prediction: The Lean follow-up for MetaCICSRKernel$^{\uparrow}$ must provide `Nontrivial MetacicSrKernelUp` and `FieldFaithful MetacicSrKernelUp` in `TasteGate.lean`, and the instance audit must expose exactly four discharge-face fields before any finite subject-reduction handoff is admitted.

Independence witness: metaciccriticalpath, metacic-subject-reduction-obligation, subjectreductiondischargesocket, closedtermsubstitutionboundary, metacicnormalizationfrontier: MetacicSrKernelUp is not bijective to any listed sibling because it simultaneously carries the critical-path ledger, four discharge faces, route socket, closed-substitution boundary, normalization-frontier boundary, finite handoff row, transport, replay, provenance, and local NameCert naming.

Closure status for NameCert$_{\text{MetaCICSRKernel}^{\uparrow}}$**.**

Constructive story: A MetaCICSRKernel$^{\uparrow}$ packet is generated from the MetaCIC critical-path ledger, a four-face subject-reduction obligation row, a discharge-socket row, beta-substitution, application-argument, lambda-domain, and

Pi-domain discharge rows, closed-substitution and normalization-frontier boundaries, a finite handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}\uparrow}$, seed)

Scope closed: Definition 5561.1, Definition 5561.2, and Definition 5561.3 bind the finite four-face subject-reduction kernel route and its handoff surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetaCICSRKernel}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove closed subject reduction, unrestricted dependent-codomain preservation, erased substitution stability, global normalization, closed consistency, Lean verification, or a bridge-checked public schema.

Upgrade path: Obligation closure requires checked carrier admission, four-face discharge exposure, socket compatibility, closed-substitution boundary compatibility, normalization-frontier compatibility, handoff non-escape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.MetacicSrKernelUp.

$\mathsf{RealApartnessField}^{\uparrow}$ reads its inverse guard through chapter 38, chapter 301, and chapter 291 before the child routes expose apartness-field comparison rows.

Chapter 5562

A Concrete Naming Certificate for $\mathsf{RealApartnessField}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealApartnessField}^{\uparrow}$ packet records the constructive field-facing surface for real names whose inverse operation is guarded by displayed apartness evidence. It sits after the terminal real-name seal of chapter 38, the positive apartness route of chapter 1881, the located order boundary of $\mathsf{LocatedRealOrder}^{\uparrow}$, the algebraic source rows of chapter 45, and the dyadic tolerance ledger of chapter 291. The packet joins $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{LocatedRealOrder}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ as one finite NameCert surface; it does not import trichotomy, quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every apartness-field read passes through the $\mathsf{Real}^{\uparrow}$ seal and real readback rows of chapter 38, the apartness/order bridge of chapter 1905, the specialization order surface of chapter 1378, and the field operation rows of chapter 45 before it exports an apartness algebra surface.

Completion-facing uses of an apartness-field inverse keep the distance-preserving read compatible with the isometric embedding boundary of chapter 439 before any completion consumer treats the $\mathsf{Real}^{\uparrow}$ seal as a field-facing endpoint. This is a dependency on the displayed $\mathsf{Met}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ distance rows, not on host function equality or quotient completion.

Theorem 5562.1 (Real apartness field located source route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or order-facing read first consumes the* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *route of chapter 1905 before the* $\mathsf{Field}^{\uparrow}$ *inverse branch may be exposed. In carrier notation, the displayed packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

routes the real seal R, apartness row A, located-order row O, field row K, dyadic ledger D, transport H, replay C, provenance P, and local naming row N through the apartness/order bridge before any consumer reads an inverse or order-facing field operation.

Proof. Project the accepted carrier. The inverse branch is stored in the field row K, but the carrier definition also displays the real seal R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N in the same finite packet.

Read the bridge route cited by the introduction. The $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ chapter supplies the sibling route that ties apartness evidence to located order data before an order-facing consumer can use the real field surface.

Now compose the existing NameCert obligations with inverse-guard stability. The obligation rows require inverse admission to consume A and D, while located order compatibility consumes

O over the same real seal R. The transport and replay rows H, C, P, N preserve that source route, so the field row K exposes its inverse branch only after the apartness/order bridge has already been read. □

Definition 5562.2 (Real apartness field carrier)**.** *A* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the real-name surface exposed to field consumers.*
2. *A is the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive-apartness row used to certify that a denominator is separated from zero before an inverse read is accepted.*
3. *O is the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *located order row that routes order-facing comparisons through displayed finite windows and dyadic cells.*
4. *K is the* $\mathsf{Field}^{\uparrow}$ *algebraic operation row for addition, multiplication, negation, zero, one, and the inverse request boundary.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and positivity ledger controlling arithmetic error, gap size, and reciprocal budget.*
6. *H records componentwise* hsame *transport for the real seal, apartness row, located order row, field row, dyadic ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each field-facing request through R, A, O, K, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealApartnessField}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealApartness}^{\uparrow}$, $\mathsf{LocatedRealOrder}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one constructive real field packet whose inverse reads are admitted only through apartness evidence. It has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 5562.3 (Real apartness field StreamName dependency)**.** *Every apartness-field comparison over* $\mathsf{Real}^{\uparrow}$ *must expose the* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126 and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 before the* $\mathsf{Real}^{\uparrow}$ *apartness and order-field row of chapter 38 is consumed. Thus a comparison read of an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier factors through finite stream observations, regular-rational tolerance data, the terminal Real seal, and only then the apartness, located-order, and field rows A, O, K.*

Proof. Project the carrier of Definition 5562.2. The real seal row R is the only Real-facing entry, and the apartness, order, and field rows A, O, K are downstream comparison rows rather than independent host-real operations.

Use the Real chapter route cited by the introduction. A $\mathsf{Real}^{\uparrow}$ seal is read from finite name data, so the comparison must first expose the $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ readback that supply the visible Real name.

Finally read the apartness-field rows. The positive-apartness row A, located order row O, field row K, and dyadic ledger D consume that already displayed Real readback, while H, C, P, N preserve and name the same finite route. Hence no comparison can bypass StreamName windows or regular-rational readback before using the Real apartness/order field surface. □

Theorem 5562.4 (Real apartness field NameCert obligations)**.** *For every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier*

$$F = (R, A, O, K, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the real seal, positive apartness row, located order row, field operation row, dyadic ledger, transport, replay, provenance, and local naming row;*
2. *algebraic projection, requiring every addition, multiplication, negation, zero, and one read to pass through the displayed* $\mathsf{Field}^{\uparrow}$ *row* K *and the real seal row* R*;*
3. *inverse admission, requiring an inverse read to consume the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row* A *and the positive dyadic gap data in* D *before the field row* K *exports the reciprocal branch;*
4. *located order compatibility, requiring the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *row* O *to read the same real-name and dyadic comparison data used by the apartness row;*
5. *transport and replay, requiring* H *and* C *to preserve the real, apartness, order, field, dyadic, provenance, and local naming rows together;*
6. *ledger exhaustion, excluding trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

No real apartness field obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5562.2. The accepted packet displays exactly the real seal, apartness row, located order row, field row, dyadic ledger, and structural rows.

The algebraic projection reads the $\mathsf{Field}^{\uparrow}$ row K only over the displayed real seal R. Thus the operation surface is the carried field row, not host real arithmetic or a quotient-real operation.

The inverse projection first reads A and D. The apartness row supplies positive separation from zero, and the dyadic row supplies the gap and reciprocal budget; without those rows the inverse request remains at the boundary and cannot be consumed as a field operation.

Finally, O, H, C, P, N keep comparison, transport, replay, provenance, and naming inside the same finite packet. Since none of the carrier coordinates stores a trichotomy decision, quotient equality, choice schedule, selected representative, completeness witness, Lean theorem, or standard bridge, none becomes a local NameCert obligation. □

Theorem 5562.5 (Real apartness field located-order inverse guard)**.** *Let*

$$F = (R, A, O, K, D, H, C, P, N)$$

be an accepted $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. An inverse read from the field row* K *is accepted only after the located-order row* O*, positive apartness row* A*, and dyadic reciprocal-budget row* D *have been displayed over the same* $\mathsf{Real}^{\uparrow}$ *seal* R*. The guard is a finite L10 readback through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{LocatedRealOrder}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$*; it does not use trichotomy, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5562.2. The inverse request is stored in K, but the same packet also displays the real seal R, positive apartness row A, located-order row O, dyadic ledger D, and structural rows H, C, P, N.

Apply the NameCert obligations of Definition 5562.4. The inverse admission clause exposes A and D before the reciprocal branch is exported, while the located-order compatibility clause requires O to read the same real-name and dyadic comparison data.

Use the StreamName dependency theorem Definition 5562.3. It forces the $\mathsf{Real}^{\uparrow}$ seal R to be read back from finite $\mathsf{StreamName}^{\uparrow}$ windows and $\mathsf{RegSeqRat}^{\uparrow}$ tolerance data before the apartness, order, and field rows are consumed.

Finally read the located real route through O. Since H, C, P, N only transport, replay, source, and name the same finite packet, they cannot add trichotomy, quotient real equality, a selected representative, countable choice, or ambient completeness. Therefore an accepted inverse read is exactly a field-row read guarded by O, A, D over the same displayed Real seal R. □

Theorem 5562.6 (Real apartness field inverse-guard stability)**.** *Let*

$$F = (R, A, O, K, D, H, C, P, N)$$

be an accepted $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. An inverse read from the field row K may be exported to a downstream consumer only when the displayed apartness guard in A survives componentwise* hsame *transport through the* $\mathsf{Real}^{\uparrow}$ *seal R, the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A, the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary O, the* $\mathsf{Field}^{\uparrow}$ *operation row K, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and reciprocal-budget ledger D, displayed* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row N.*

Consequently, the inverse branch is stable under the carried transport rows exactly when its apartness evidence remains displayed in the same finite packet. It is not stable by trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The inverse request sits in the field row K, but the NameCert obligations require the request to consume the positive-apartness row A and the positive dyadic gap data in D before the reciprocal branch is exported.

Transport preserves the guard componentwise. The real seal R, apartness row A, located order boundary O, field row K, and dyadic ledger D are carried through the same hsame route, while C, P, N replay, source, and name that route.

Therefore the inverse operation remains downstream of the apartness guard after transport. If the displayed guard is absent, the transported packet has no row from which an inverse read can be exported.

The carrier stores no trichotomy decision, quotient equality, selected representative, choice schedule, host inverse, or completeness witness. Stability of the inverse branch is exactly stability of the displayed apartness guard inside the finite $\mathsf{RealApartnessField}^{\uparrow}$ packet. □

Theorem 5562.7 (Real apartness field reciprocal-budget locality)**.** *Let $F = (R, A, O, K, D, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. Every inverse read exported from the field row K must first expose the positive apartness row A and the dyadic reciprocal-budget ledger D. The reciprocal route is local to these displayed rows and their componentwise transport, not to trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The inverse branch is stored in K, but its admission rows are the positive apartness row A and the dyadic ledger D.

Use inverse-guard stability from Definition 5562.6. That theorem keeps the inverse branch downstream of the displayed apartness guard under the same componentwise hsame transport.

The dyadic ledger supplies the reciprocal budget attached to that guard. Rows H, C, P, N preserve, replay, source, and name the same finite route, so an inverse read cannot be exported unless both A and D have already been displayed. □

Theorem 5562.8 (Real apartness field multiplicative inverse boundary)**.** *Let $F = (R, A, O, K, D, H, C, P, N)$ be an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier. A multiplicative inverse read from K is available to a downstream real-field consumer only after A displays positive apartness of the denominator from zero and D displays the corresponding positive dyadic gap and reciprocal budget. Therefore the inverse operation is controlled by constructive apartness evidence rather than by trichotomy, quotient equality, selected representatives, or choice.*

Proof. Begin with the accepted carrier. The inverse request is a branch of the field row K, but the carrier definition makes that branch depend on the same apartness and dyadic rows named by A and D.

Read the apartness row. The row A supplies a positive separation witness for the denominator source, so the consumer does not ask whether the denominator is equal to zero by trichotomy or quotient equality.

Read the dyadic ledger next. The row D records the positive gap and reciprocal budget used by the inverse branch, so the reciprocal calculation is bounded by displayed finite arithmetic rather than by an unrecorded host inverse.

The remaining rows only preserve the route. Transport, replay, provenance, and local naming cannot add a selected representative, choice operator, or ambient completeness datum, so the inverse boundary is exactly the apartness-controlled field read stated above. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}$.

Constructive story: A $\mathsf{RealApartnessField}^{\uparrow}$ packet is generated from a $\mathsf{Real}^{\uparrow}$ seal, a $\mathsf{CauchyRealApartness}^{\uparrow}$ positive-apartness row, a $\mathsf{LocatedRealOrder}^{\uparrow}$ located order row, a $\mathsf{Field}^{\uparrow}$ operation row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance and reciprocal-budget ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5562.2, Definition 5562.4, Definition 5562.6, and Definition 5562.8 seed the constructive real field boundary whose inverse branch is guarded by positive apartness evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealApartnessField}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host real arithmetic, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, algebraic projection, inverse admission, located order compatibility, transport, replay, and ledger exhaustion for BEDC.Derived.RealApartnessFieldUp.

Theorem 5562.9 (Real apartness field regular-Cauchy apartness/order dependency)**.** *Every inverse-guard read or apartness-field read of an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier must pass through the regular-Cauchy apartness/order route supplied by chapter 1710 and the positive-apartness route of chapter 1881 before the* $\mathsf{Field}^{\uparrow}$ *inverse branch is consumed. In carrier notation, a consumer must display the* $\mathsf{CauchyRealApartness}^{\uparrow}$ *row A, the* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary O, the* $\mathsf{Field}^{\uparrow}$ *operation row K, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and reciprocal-budget ledger D inside the same packet*

$$F = (R, A, O, K, D, H, C, P, N)$$

before it treats an inverse as a field operation. The route adds no trichotomy, quotient equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with inverse-guard stability, Definition 5562.6. It requires an inverse read to keep the displayed apartness guard in A stable through the real seal, located order boundary, field row, dyadic ledger, replay, provenance, and local naming rows.

Next read the multiplicative inverse boundary, Definition 5562.8. The inverse branch in K is available only after A displays positive apartness from zero and D displays the corresponding positive dyadic gap and reciprocal budget.

The regular-Cauchy apartness/order chapter supplies the regular source of that positive comparison: Definition 1710.3 routes every strict comparison or apartness read through its displayed budget, finite window, positive dyadic boundary, regular readback, and Real seal. The Cauchy real apartness chapter supplies the positive-apartness handoff through Definition 1881.3 and Definition 1881.4.

Combining these routes forces the RealApartnessField inverse branch to consume regular-Cauchy apartness/order evidence and positive Cauchy real apartness before it is read as a field inverse. Since each cited route refuses trichotomy, quotient real equality, selected representatives, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness, the dependency adds none of those principles to the field packet. □

Every $\mathsf{RealApartnessField}^{\uparrow}$ inverse read consumes the apartness-order bridge of chapter 1905 before the field carrier of chapter 45 can export the reciprocal branch. The dyadic tolerance row inherited by the same route prevents trichotomy or quotient-real equality from entering the packet, so the field operation is read only after constructive apartness-order evidence has been displayed.

Theorem 5562.10 (Real apartness field dependency route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or apartness-field read consumes the regular-Cauchy source rows, the* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *route of chapter 1710, the located-real bridge through chapter 1933, and the* $\mathsf{RealLocatedOrder}^{\uparrow}$ *row of chapter 3640 before it exports field-level apartness. In carrier notation, for*

$$F = (R, A, O, K, D, H, C, P, N),$$

the regular source behind R, the positive apartness row A, the located-order row O, the field row K, the dyadic ledger D, transport H, replay C, provenance P, and local naming N are consumed in that order as one displayed dependency route.

Proof. Project the carrier of Definition 5562.2. The packet has no field-level apartness coordinate outside $R, A, O, K, D, H, C, P, N$.

Read the regular-Cauchy source first. The existing source route of Definition 5562.14 places the $\mathsf{RegSeqRat}^{\uparrow}$ source and the terminal $\mathsf{Real}^{\uparrow}$ seal before any field row K is consumed.

Next read the apartness and located rows. The regular-Cauchy apartness/order dependency of Definition 5562.9 supplies the $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ route for A, while Definition 5562.12 supplies the located-real bridge and Definition 5562.5 keeps the $\mathsf{RealLocatedOrder}^{\uparrow}$ row visible over the same real seal.

Finally compose with the NameCert obligations. The algebraic projection and inverse-admission rows of Definition 5562.4 allow K to export field-level apartness only after A, O, D have been displayed, and H, C, P, N preserve the same route. Thus the field-level apartness export is downstream of the regular source, regular-Cauchy apartness/order, located-real bridge, and real located-order rows, with no host equality, trichotomy, selected representative, choice schedule, or ambient completeness inserted. □

Theorem 5562.11 (Real apartness field Real/order forward link)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *packet hands its apartness-field row to* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealApartnessOrderBridge}^{\uparrow}$ *consumers only through the displayed* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$ *seal,* hsame, Cont, Pkg, *and* NameCert *rows. In carrier notation, a read of*

$$F = (R, A, O, K, D, H, C, P, N)$$

must first expose the real seal route of chapter 38 and the apartness/order bridge of chapter 1905 before the field row K can export an inverse or order-facing operation.

Proof. Project the accepted carrier of Definition 5562.2. The field row K is not stored as host real arithmetic; it is carried together with the real seal R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N.

Read the Real route first. The chapter introduction and StreamName dependency theorem route every Real-facing comparison through finite regular-rational and stream-window data before R is consumed, so the field consumer receives a displayed $\mathsf{Real}^{\uparrow}$ seal rather than a selected representative or quotient real.

Now apply the apartness/order route. The bridge cited in chapter 1905 consumes the apartness row A, located-order row O, and dyadic ledger D before any inverse or order-facing field operation is read.

Finally H, C, P, N preserve the same route under componentwise transport, replay, provenance, and local naming. Thus every forward handoff to Real and apartness-order consumers factors through the displayed rows and adds no trichotomy, quotient real equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5562.12 (Real apartness field located real route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or order-facing read must pass through the* $\mathsf{LocatedReal}^{\uparrow}$ *source route of chapter 1933 and the apartness-order bridge of chapter 1905 before the* $\mathsf{Field}^{\uparrow}$ *reciprocal branch is exposed. In carrier notation, for*

$$F = (R, A, O, K, D, H, C, P, N),$$

the real seal row R, apartness row A, located-order row O, field row K, dyadic ledger D, componentwise transport H, displayed replay C, provenance P, and local naming row N are consumed in the same finite route. The route exports no trichotomy, quotient real equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the accepted carrier from Definition 5562.2. The reciprocal branch is stored in the $\mathsf{Field}^{\uparrow}$ row K, while R, A, O, D, H, C, P, N remain visible as separate rows of the same finite packet.

Read the located-real source route first. The $\mathsf{LocatedReal}^{\uparrow}$ chapter supplies the finite comparison boundary consumed before the terminal real seal is used, so the field consumer receives located real evidence rather than a quotient representative or host real value.

Next read the apartness-order bridge. The bridge of chapter 1905 ties the apartness row A to the located-order row O over the same real seal R, and the dyadic ledger D records the reciprocal budget attached to that positive evidence.

Finally replay the consumer through H, C, P, N. Since those rows only transport, replay, source, and name the already displayed route, they cannot add trichotomy, quotient real equality, selected representatives, countable choice, or ambient completeness. Thus every inverse or order-facing read reaches the $\mathsf{Field}^{\uparrow}$ branch only after the located-real and apartness-order routes have both been displayed. □

Every $\mathsf{RealApartnessField}^{\uparrow}$ apartness read is field-facing only after it has passed through the L10 completion lattice: the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RealAlgOrder}^{\uparrow}$ comparison rows of chapter 766, and the located comparison surface of chapter 1730. Thus the packet is consumed as a classifier over displayed regular-rational, real-seal, algebraic-order, and located-comparison evidence, not as a detached field oracle. The $\mathsf{RealApartnessField}^{\uparrow}$ lattice consumes the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the located real bridge of chapter 1933, and the constructive order-field route through chapter 3640, chapter 1905, and chapter 6850. Its field-facing apartness reads therefore enter through displayed real names, located comparison, and order transport rows before any algebraic operation is used. Every

$\mathsf{RealApartnessField}^{\uparrow}$ carrier reads its terminal real-name dependency through the finite L10 completion route, not through an ambient field of host reals. The route passes through the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126.

Proposition 5562.13 (Real apartness field finite completion dependency route)**.** *For every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier* $F = (R, A, O, K, D, H, C, P, N)$*, any field-facing read of* R, A, O, K*, or* D *first exposes the finite-window* $\mathsf{StreamName}^{\uparrow}$ *rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal named above. Hence the apartness-field carrier consumes* $\mathsf{Real}^{\uparrow}$ *only through that displayed finite L10 dependency lattice, with* H, C, P, N *preserving transport, replay, provenance, and local naming for the same route.*

Proof. Project the carrier of Definition 5562.2. The only Real-facing row is R, and the apartness, located-order, field, and dyadic rows A, O, K, D are stored as downstream rows of the same finite packet.

Apply the StreamName and regular-rational dependency routes already present in the chapter. They force a comparison or inverse read to expose finite windows and regular-rational readback before the terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally read the dyadic ledger. The tolerance and reciprocal-budget information in D is displayed before the field-facing row K is exported, while H, C, P, N carry only the same finite dependency route. Therefore no read imports quotient real equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5562.14 (Real apartness field regular-Cauchy source route)**.** *Every accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *inverse read or apartness read first consumes the* $\mathsf{RegSeqRat}^{\uparrow}$ *source of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the located and order sibling rows already named by this chapter before the* $\mathsf{Field}^{\uparrow}$ *inverse data are exposed. In carrier notation, a read of*

$$F = (R, A, O, K, D, H, C, P, N)$$

passes through the regular-rational source behind R*, the terminal real seal* R*, the apartness row* A*, located-order row* O*, dyadic ledger* D*, transport* H*, replay* C*, provenance* P*, and local naming row* N *before the field row* K *can export an inverse branch.*

Proof. Project the accepted carrier of Definition 5562.2. The field row K is stored together with the real seal row R, apartness row A, located-order row O, dyadic ledger D, and the structural rows H, C, P, N.

Use the StreamName dependency theorem Definition 5562.3. It forces every comparison-facing read to expose the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 before the terminal $\mathsf{Real}^{\uparrow}$ route of chapter 38 is consumed.

Now apply the located and order child routes already carried by the chapter: Definition 5562.1, Definition 5562.12, and Definition 5562.11. These routes place the located-real, apartness-order, and Real/order dependencies before any field-facing inverse read is exported.

Finally combine the NameCert obligations with inverse-guard stability. Theorems 5562.4 and 5562.6 require the apartness guard and dyadic reciprocal budget to be displayed before K is consumed. Thus the inverse data are downstream of the RegSeqRat source, Real terminal seal, and located/order sibling rows, and the route adds no trichotomy, quotient real equality, selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The RealApartnessField$^{\uparrow}$ chapter reads its field-facing apartness rows through the regular-Cauchy apartness lattice before exporting any arithmetic surface. In particular, the route first exposes chapter 1710, then the located-real surface of chapter 1933, and then the located order rows of chapter 3640. Only after these three sibling dependencies are visible may the chapter present the RealApartnessField$^{\uparrow}$ field-facing apartness rows; the route exports no host real equality, detached order completion, selected located witness, or ambient field law outside the displayed BEDC dossier lattice. The RealApartnessField$^{\uparrow}$ lattice consumes the regular Cauchy criterion surface of chapter 1611, the located real order route of chapter 8337, and the terminal Real seal of chapter 38 before it exports apartness-field reads. Thus an inverse or comparison read from the field-facing row is routed through displayed criterion checks, order evidence, and Real sealing, rather than through a detached field oracle.

Proposition 5562.15 (Real apartness field regular Cauchy criterion route)**.** *Every accepted* RealApartnessField$^{\uparrow}$ *packet that exports an apartness-field read must first expose the* RegularCauchyCriterion$^{\uparrow}$ *criterion dependency, the* LocatedRealOrder$^{\uparrow}$ *order dependency, and the* Real$^{\uparrow}$ *seal dependency named by its carrier rows. The exported read is therefore a finite L10 route through regular Cauchy criterion data, located order evidence, terminal Real sealing,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *data.*

Proof. Project the accepted RealApartnessField$^{\uparrow}$ carrier. Its apartness-field row is downstream from the displayed Real seal, located-order row, dyadic tolerance ledger, transport, replay, provenance, and local naming rows already named by the chapter.

The regular Cauchy criterion dependency supplies the finite criterion checks consumed before the Real seal and order-facing rows are read. Since the carrier has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, or ambient Real completeness, the exported apartness-field read remains inside the displayed finite route. □

Theorem 5562.16 (Real apartness field product-topology dependency)**.** *Every product-coordinate apartness-field read of an accepted* RealApartnessField$^{\uparrow}$ *packet must expose the finite* ProductTopology$^{\uparrow}$ *product window before the inverse, order, and field rows of the apartness-field carrier are consumed. The route first reads the* Real$^{\uparrow}$ *seal,* RegSeqRat$^{\uparrow}$ *readback,* StreamName$^{\uparrow}$ *finite windows,* Field$^{\uparrow}$ *operation row,* DyadicRatCore$^{\uparrow}$ *tolerance ledger, and* RealApartnessOrderBridge$^{\uparrow}$ *apartness-order bridge, and only then permits the product-coordinate consumer to use the* RealApartnessField$^{\uparrow}$ *inverse or order-facing field surface.*

Proof. Project the accepted RealApartnessField$^{\uparrow}$ carrier through Definition 5562.2. Its inverse and order-facing reads are stored in the displayed real seal, apartness, located-order, field, dyadic, transport, replay, provenance, and local naming rows, so a product-coordinate consumer has no separate coordinate from which to read a field operation.

Read the product surface through chapter 2000. The finite ProductTopology$^{\uparrow}$ window supplies the coordinate route before either coordinate can be consumed as a real apartness-field input.

Use the source rows already required by the chapter: the StreamName$^{\uparrow}$ windows and RegSeqRat$^{\uparrow}$ readback feed the Real$^{\uparrow}$ seal, the DyadicRatCore$^{\uparrow}$ ledger supplies tolerance and positivity data, the Field$^{\uparrow}$ row carries the operation boundary, and the RealApartnessOrderBridge$^{\uparrow}$ route ties apartness evidence to order-facing comparison over the same real seal. The product-coordinate read therefore reaches the inverse or order-facing field rows only after the product-topology dependency has been displayed. □

Theorem 5562.17 (Real apartness field Real–RegSeqRat–Dyadic route)**.** *Every* RealApartnessField$^{\uparrow}$ *apartness-field comparison read exposes the terminal* Real$^{\uparrow}$ *seal of chapter 38, the* RegSeqRat$^{\uparrow}$

regular-rational readback of chapter 301, and the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger of chapter 291 before the field-facing comparison row is exported. The route is a displayed dependency through real sealing, regular-rational readback, and dyadic tolerance control; it does not add trichotomy, quotient real equality, selected representatives, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5562.2. Its comparison-facing data are stored with the real seal row, apartness row, located-order row, dyadic ledger, transport, replay, provenance, and local naming rows.

Read the $\mathsf{RegSeqRat}^{\uparrow}$ source before the terminal $\mathsf{Real}^{\uparrow}$ seal. The existing regular-Cauchy source route forces the comparison to pass through regular-rational readback rather than through an untracked real representative.

Then expose the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger. The apartness guard and reciprocal budget are consumed only with the displayed dyadic tolerance data, so the field-facing row is downstream of the Real, RegSeqRat, and DyadicRatCore dependencies named above. □

Theorem 5562.18 (Real apartness field inverse-guard dependency route)**.** *Every inverse-guard export from an accepted* $\mathsf{RealApartnessField}^{\uparrow}$ *carrier factors through the* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{CauchyRealApartness}^{\uparrow}$ *positive-apartness row,* $\mathsf{LocatedRealOrder}^{\uparrow}$ *boundary row, and* $\mathsf{Field}^{\uparrow}$ *operation row before a field inverse is public. The route composes the Real–RegSeqRat–Dyadic dependency with inverse-guard stability and the regular-Cauchy apartness/order dependency, and it refuses trichotomy, quotient equality, selected representatives, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 5562.17. It places every field-facing comparison behind the terminal $\mathsf{Real}^{\uparrow}$ seal, the regular rational readback, and the dyadic tolerance ledger.

Next apply the inverse-guard stability surface in the parent chapter. The inverse row is public only when the positive-apartness guard remains attached to the displayed carrier, so the guard cannot be replaced by quotient equality or by a selected representative.

Finally read the regular-Cauchy apartness/order dependency. The positive-apartness row is checked together with the located-order boundary before the $\mathsf{Field}^{\uparrow}$ operation row exposes an inverse. Hence the inverse export remains a finite route through Real, RegSeqRat, DyadicRatCore, CauchyRealApartness, LocatedRealOrder, and Field data, with trichotomy, countable choice, quotient equality, selected representatives, and ambient Real completeness outside the packet. □

Chapter 5563

A Concrete Naming Certificate for $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy convergence in constructive real analysis, the $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ packet records the finite threshold schedule that lets a regular rational name answer downstream Real and Completion precision requests without appealing to a quotient of streams. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational source of chapter 301, and the terminal real-name seal of chapter 38. The packet gives the L10 Real and Completion route a named finite modulus surface; it does not assert ambient $\mathsf{Real}^{\uparrow}$ completeness, selected limits, trichotomy, countable choice, or quotient-stream equality.

Definition 5563.1 (Modulus of regular convergence carrier)**.** *A* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, Q, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose regular rational reads are being controlled.*
2. *Q is the modulus row, recording a displayed tolerance request and the finite index threshold from which the regular convergence comparison is certified.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source reads inspected by Q.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the dyadic radius, comparison budget, and readback precision used by the threshold row.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, Q, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the source, modulus, window, tolerance, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision request through R, Q, W, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, Q, W, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ModulusOfRegularConvergenceUp.ModulusOfRegularConvergenceTasteGate single carrier alignment`

Theorem 5563.2 (Modulus of regular convergence NameCert obligations)**.** *For every accepted* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *carrier* $M = (R, Q, W, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}}$ *surface consists exactly of carrier admission, regular-rational source exposure, explicit threshold exactness, finite stream-window coverage, dyadic tolerance control, terminal real-seal nonescape, componentwise transport, displayed replay, package provenance, and local naming. Every precision-facing read must expose*

$$R, Q, W, D \longrightarrow E \longrightarrow H, C, P, N.$$

No obligation is discharged by quotienting two streams, selecting a limit, invoking countable choice, importing trichotomy, or using ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 5563.1. The packet first displays the regular rational source R, the threshold row Q, the stream windows W, and the dyadic tolerance ledger D.

The threshold obligation is local to Q. The windows in W supply the finite source reads named by that threshold, while D supplies the dyadic comparison budget used to certify the requested regular convergence read.

The real-facing seal E is downstream of those four rows. Transport through H and replay through C preserve the same order, and P, N record provenance and local NameCert comparison without adding an outside coordinate.

Since quotient stream equality, selected limits, host real equality, countable choice, trichotomy, and ambient completeness do not occur among the carrier coordinates, every NameCert obligation is read through the displayed finite route. □

Theorem 5563.3 (Regular convergence finite-modulus handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{ModulusOfRegularConvergence}^{\uparrow}$ *packet must read the regular rational source row, explicit modulus threshold, finite stream windows, and dyadic tolerance ledger before consuming the terminal real seal. The packet therefore exports a finite threshold route from regular rational data into Real-facing data, not a host completeness theorem or a quotient construction.*

Proof. Project an accepted carrier as $M = (R, Q, W, D, E, H, C, P, N)$. The only regular rational input is R, and the only threshold schedule is Q.

The row W exposes the finite observations named by that schedule. The row D supplies the dyadic tolerance comparison that turns those observations into a regular convergence read.

Only after R, Q, W, D are present may the consumer read E as the terminal $\mathsf{Real}^{\uparrow}$ seal. Componentwise transport, replay, provenance, and local naming keep the same route visible to downstream consumers.

The carrier has no coordinate for selected limits, quotient stream equality, host real equality, countable choice, trichotomy, or ambient Real completeness. Thus every completion-facing handoff factors through the finite modulus rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}$ **.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ModulusOfRegularConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5563.1, Definition 5563.2, and Definition 5563.3 seed the finite modulus route from regular rational source data through stream windows, dyadic tolerances, and a Real-facing seal.

Formal verification: FormalStatus(NameCert$_{\text{ModulusOfRegularConvergence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, threshold exactness, stream-window coverage, dyadic tolerance control, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ModulusOfRegularConvergenceUp.

Constructive story: A ModulusOfRegularConvergence$^{\uparrow}$ packet is generated from a RegSeqRat source row, explicit modulus threshold row, finite StreamName windows, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5564

A Concrete Naming Certificate for $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$

Following the classical separation between the Riemann Hypothesis as a fixed proposition and proof search as a metamathematical process, the $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ packet records the finite BEDC surface of the route-closure boundary described in chapter 9336. It sits over the RH condition-chain rows of that vision chapter, the halting distinction traces of chapter 492, the halting obstruction of chapter 490, the proof witness chain surface of chapter 1238, and the zeta-zero witness boundary of chapter 73. The packet names route, grounding-rank, decider-refusal, and NameCert rows; it does not decide RH, prove RH, refute RH, or assert independence for any chosen fragment.

Definition 5564.1 (RH-route halting boundary carrier)**.** *An* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (R, G, D, F, Z, H, C, P, N)$$

with the following displayed rows:

1. *R is the RH-route row, recording the displayed condition chain, rule tower, proof-output program, or route family being inspected.*
2. *G is the grounding-rank row, recording whether the inspected route has a displayed finite rank and, when it does, the finite top certificate and transports back to* RH*.*
3. *D is the decider-refusal row, recording that a proposed global or sufficiently expressive RH-route closure decider is a halting-boundary request rather than an internal BEDC classifier.*
4. *F is the fixed-statement row separating the proposition* RH *from route-search totality and from global proof-search classification.*
5. *Z is the zeta-zero witness interface row, exposing the critical-strip and critical-line witnesses that a genuine RH proof or counterexample datum would have to consume.*
6. *H records componentwise* hsame *transport for route, grounding-rank, refusal, fixed-statement, zeta-witness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from route inspection through R, G, D, F, Z, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$*,* $\mathsf{HaltingDistinctionTrace}^{\uparrow}$*,* $\mathsf{HaltingObstruction}^{\uparrow}$*,* $\mathsf{ProofWitnessChain}^{\uparrow}$*,* $\mathsf{ZetaZeros}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, G, D, F, Z, H, C, P, N$. It has no coordinate for an RH proof, an RH counterexample, an RH-specific independence theorem, a global theorem-route closure decider, a proof-search oracle, or an unmarked halting predicate.

Lean4 (definition) `BEDC.Derived.RHRouteHaltingBoundaryUp.RHRouteHaltingBoundaryUp`

Theorem 5564.2 (RH-route halting boundary NameCert obligations)**.** *For every accepted* $\mathsf{RHRouteHaltingBoundar}$ *carrier $B = (R, G, D, F, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, route-row exposure, grounding-rank exactness, decider-refusal exactness, fixed-RH-statement separation, zeta-witness interface exposure, componentwise transport, displayed replay, package provenance, and local naming. Every decision-facing read must expose*

$$R, G, D, F, Z \longrightarrow H, C, P, N.$$

The packet distinguishes RH truth from route-closure decidability: it can record finite grounded routes and reject global closure deciders, but it does not provide $\Phi : \mathsf{RH}$, a counterexample datum, or a fragment-specific independence theorem.

Proof. Carrier admission is projection from Definition 5564.1. The route row R names the inspected RH-directed condition chain, rule tower, or proof-output route, while G records the displayed grounding-rank information attached to that route.

The refusal row D is read separately from the fixed-statement row F. Thus the packet can reject a global or expressive route-closure decider as a halting-boundary request without changing the fixed proposition RH.

The zeta-witness interface Z records what a genuine proof or counterexample datum would have to consume. It does not itself supply the total witness function, a counterexample point, or an independence theorem.

The structural rows H, C, P, N transport, replay, source, and name only the displayed route, rank, refusal, fixed-statement, and zeta-interface rows. Since no proof oracle, route oracle, halting predicate, counterexample, or independence theorem occurs among the coordinates, the listed obligations exhaust the finite NameCert surface. □

Theorem 5564.3 (RH route boundary nondecision)**.** *Every accepted* $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ *packet separates three reads: the fixed* RH *statement row, the finite route or grounding-rank row, and the decider-refusal row. Consequently a consumer may use the packet to identify a finite grounded route or to reject an expressive route-closure decider, but no consumer can read an RH truth value, RH refutation, or independence result from the packet alone.*

Proof. Project the carrier $B = (R, G, D, F, Z, H, C, P, N)$. The row F names the fixed statement, while R and G name route and grounding-rank data.

The row D records the halting-boundary refusal for global or sufficiently expressive route deciders. This refusal is a statement about the classifier request, not a proof or refutation of the fixed proposition.

The row Z exposes the zeta-zero witness interface required by an actual decision datum. Because no coordinate supplies such a datum, transport and replay through H, C can preserve only the boundary information already displayed.

Package provenance and local naming through P, N do not add a truth-value coordinate. Therefore the packet is a finite halting-boundary NameCert surface, not an RH decision artifact. □

Closure status for $\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5564.1, Definition 5564.2, and Definition 5564.3 seed the finite route, grounding-rank, decider-refusal, fixed-statement, zeta-interface, transport, replay, provenance, and local naming surface for the RH route halting boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RHRouteHaltingBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove RH, refute RH, establish RH independence over any fragment, provide a global theorem-route closure decider, admit a proof-search oracle, close halting, supply Lean verification, or provide a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, route-row exposure, grounding-rank exactness, decider-refusal exactness, fixed-statement separation, zeta-witness interface exposure, transport, replay, provenance, and local naming for BEDC.Derived.RHRouteHaltingBoundaryUp.

Constructive story: An $\mathsf{RHRouteHaltingBoundary}^{\uparrow}$ packet is generated from RH route rows, grounding-rank rows, decider-refusal rows, fixed-RH-statement rows, zeta-witness interface rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5565

A Concrete Naming Certificate for $\mathsf{Premetric}^{\uparrow}$

Following Kelley on uniform sources and Bishop–Bridges constructive metric data, the $\mathsf{Premetric}^{\uparrow}$ packet records the distance-comparison surface that sits before separated reflection and metric completion. It reads the finite entourage interface of chapter 339, supplies the source row consumed by chapter 4201, and keeps the completion-facing boundary visible before chapter 1506 or the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 can be inspected. The packet names a constructive distance witness and zero-distance classifier; it does not quotient points, select representatives, assert host equality, or import ambient completeness.

Definition 5565.1 (Premetric carrier)**.** *A* $\mathsf{Premetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (X, U, D, Z, S, M, H, C, Q, N)$$

with the following displayed rows:

1. *X is the source carrier row whose points are admitted only through displayed BEDC windows.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, exposing the finite entourage requests that the distance witness is allowed to refine.*
3. *D is the distance-witness relation row, recording finite positive-radius comparisons between displayed point windows.*
4. *Z is the zero-distance classifier row, comparing two displayed source windows only through the radius comparisons stored in D.*
5. *S is the separated-reflection handoff row consumed by* $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$ *after Z has been displayed.*
6. *M is the completion-facing nonescape row, binding every* $\mathsf{MetricCompletion}^{\uparrow}$ *read to X, U, D, Z, S before any completed object is named.*
7. *H records componentwise* hsame *transport for source, uniform, distance, zero-distance, reflection, completion, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite distance or zero-distance request through X, U, D, Z, S, M, H.*
9. *Q is the* Pkg *provenance over* $\mathsf{Premetric}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{SeparatedReflectionUniversal}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{Premetric}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, U, D, Z, S, M, H, C, Q, N$. It has no coordinate for quotient point equality, selected separated representatives, chosen completion limits, host metric equality, trichotomy, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Lean4 (definition) `BEDC.Derived.PremetricUp.PremetricTasteGate single carrier alignment`

Theorem 5565.2 (Premetric NameCert obligations)**.** *For every accepted* $\mathsf{Premetric}^\uparrow$ *carrier $P = (X, U, D, Z, S, M, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{Premetric}^\uparrow}$ *surface consists exactly of source-carrier admission, uniform-source readback, distance-witness exposure, zero-distance classification, separated-reflection handoff, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Each obligation is read through the finite route*

$$X, U, D, Z \longrightarrow S, M \longrightarrow H, C, Q, N.$$

Proof. Carrier admission is projection from Definition 5565.1. The source row X and uniform row U are the only inputs a consumer may inspect before asking for distance data.

The distance-witness relation D supplies finite positive-radius comparisons between displayed source windows. The zero-distance classifier Z is downstream of D, so equality-like reads are exposed as repeated distance comparisons rather than as host point equality.

The reflection row S and completion boundary M are then forced to read X, U, D, Z. Componentwise transport through H and replay through C preserve that order, while Q, N record provenance and local naming.

Since quotient point equality, selected separated representatives, chosen completion limits, host metric equality, trichotomy, countable choice, and ambient Real completeness do not occur among the carrier coordinates, no NameCert obligation can be discharged outside the displayed finite route. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCarrier namecert obligations`

Theorem 5565.3 (Premetric separated-reflection route)**.** *Let $P = (X, U, D, Z, S, M, H, C, Q, N)$ be an accepted* $\mathsf{Premetric}^\uparrow$ *carrier. Every* $\mathsf{SeparatedReflectionUniversal}^\uparrow$ *consumer of P must first read the source carrier X, the uniform source row U, the distance-witness relation D, and the zero-distance classifier Z. The handoff row S therefore presents the reflection source as a finite zero-distance comparison surface, not as a quotient carrier or a selected representative map.*

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricSeparatedReflectionRoute`

Proof. Apply the obligation theorem, Definition 5565.2. It identifies X, U, D, Z as the full prefix needed before the separated-reflection handoff can be consumed.

The row U records the uniform-source requests, while D records the finite distance comparisons refining those requests. The classifier Z reads zero distance only through those comparisons.

Thus S supplies the premetric source expected by the separated-reflection route of chapter 4201. The structural rows H, C, Q, N transport, replay, source, and name the same handoff; they do not add quotient data, selected representatives, or host equality. □

Theorem 5565.4 (Premetric zero-distance classifier boundary)**.** *Let $P = (X, U, D, Z, S, M, H, C, Q, N)$ be an accepted* $\mathsf{Premetric}^\uparrow$ *carrier. Every equality-like read of P must pass through the finite distance-witness row D and the zero-distance classifier row Z before any* $\mathsf{SeparatedReflectionUniversal}^\uparrow$, $\mathsf{MetricCompletion}^\uparrow$, *or* $\mathsf{Real}^\uparrow$ *consumer may inspect the packet. Thus zero-distance comparison is*

the only equality-facing classifier route exposed by the premetric certificate; no consumer receives quotient point equality, selected separated representatives, host metric equality, chosen completion limits, or ambient completeness data.

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCarrier zero distance classifier boundary`

Proof. Project the carrier of Definition 5565.1. The source row X and uniform row U admit the displayed point windows, while D is the only row that records positive-radius distance witnesses.

Apply the NameCert obligation surface of Definition 5565.2. It places the zero-distance classifier Z downstream of D, so any equality-like request must be read as a finite family of distance comparisons before it can be passed onward.

The separated-reflection route of Definition 5565.3 consumes exactly that D, Z prefix through S. Completion-facing readers then use the same prefix through M, so neither route can replace the classifier by a quotient carrier, a selected representative, or host metric equality.

The refused endpoints follow from the carrier inventory. The tuple has no coordinate for quotient equality, selected representatives, chosen completion limits, host metric equality, or ambient completeness, and the support rows H, C, Q, N only transport, replay, package, and name the displayed premetric route. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricZeroDistanceClassifierBoundary`

Theorem 5565.5 (Premetric completion boundary)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{Premetric}^{\uparrow}$ *packet reaches completion-facing data only through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow}, \mathsf{Premetric}^{\uparrow}, \mathsf{SeparatedReflectionUniversal}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow}, \mathsf{Real}^{\uparrow}.$$

In particular, the completion-facing row M exports finite distance-comparison and zero-distance data, not a completed point, a selected Cauchy limit, a quotient equality, or an ambient completeness principle.

Proof. Project an accepted carrier $P = (X, U, D, Z, S, M, H, C, Q, N)$. The completion-facing row M is downstream of the source, uniform, distance, zero-distance, and separated-reflection rows.

Any completion consumer must therefore read X, U, D, Z, S before it can inspect M. The terminal $\mathsf{Real}^{\uparrow}$ seal, when a downstream completion chapter exposes one, is reached only after that same finite route has been displayed.

The packet contains no coordinate for a completed point, selected Cauchy limit, quotient equality, host metric equality, countable choice, or ambient completeness. Hence the completion boundary is a nonescape condition over the finite premetric surface. □

Lean4 ✓ `BEDC.Derived.PremetricUp.PremetricCompletionBoundary`

Closure status for $\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}$.

Constructive story: A $\mathsf{Premetric}^{\uparrow}$ packet is generated from a source carrier row, a UniformSpace source row, a distance-witness relation, a zero-distance classifier, a separated-reflection handoff, a completion-facing nonescape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5565.1, Definition 5565.2, Definition 5565.3, and Definition 5565.5 seed the constructive premetric boundary between uniform sources, separated reflection, metric completion, and Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Premetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient point equality, selected representatives, chosen completion limits, host metric equality, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked source-carrier admission, uniform-source readback, distance-witness exposure, zero-distance classification, separated-reflection handoff, completion-facing nonescape, transport, replay, provenance, and local naming for BEDC.Derived.PremetricUp.

Chapter 5566

A Concrete Naming Certificate for $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$

Following Bourbaki and Bishop–Bridges finite-dimensional analysis, the $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ packet records norm comparison on a displayed finite coordinate window rather than through an ambient basis theorem. It consumes the vector-space surface of chapter 47, the norm interface of chapter 89, the finite coordinate and matrix rows of chapter 49, the real comparison seal of chapter 38, and the finite-net source of chapter 1250. The packet is an L10-facing finite analysis route for later $\mathsf{NormedSpace}^{\uparrow}$ and bounded-operator consumers: it exports named comparison constants over displayed finite rows, not host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, or choice-selected duals.

Definition 5566.1 (Finite-dimensional norm-equivalence carrier)**.** *A* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (V, K, A, B, R, L, U, H, T, P, N)$$

with the following displayed rows:

1. *V is the consumed* $\mathsf{VecSpace}^{\uparrow}$ *row, exposing vector addition, zero, negation, scalar action, and vector classification only through displayed carrier data.*
2. *K is the finite coordinate window row, represented by a* $\mathsf{Mat}^{\uparrow}$*-indexed coordinate table and its finite reconstruction ledger.*
3. *A and B are the two consumed* $\mathsf{Norm}^{\uparrow}$ *rows on the same displayed vector source.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row used to read nonnegative norm values, finite sums, and scalar comparison constants.*
5. *L is the compact finite-net lower-bound row over the coordinate unit window, supplied through* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *and its finite coverage ledger.*
6. *U is the compact finite-net upper-bound row over the same coordinate window.*
7. *H stores componentwise* hsame *routes for the vector row, coordinate row, both norm rows, real comparison rows, and finite-net bound rows.*
8. *T stores displayed* Cont *replay for each coordinate read, norm read, finite-net comparison, and downstream consumer request.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed rows $V, K, A, B, R, L, U, H, T, P, N$. It has no coordinate for host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, choice-selected duals, or an ambient normed-space object.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceTasteGate`
`single carrier alignment`

Theorem 5566.2 (Finite-dimensional norm-equivalence NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *carrier $E = (V, K, A, B, R, L, U, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space row, coordinate matrix row, two norm rows, real comparison row, compact finite-net lower and upper bound rows, transports, replay, provenance, and local naming;*

2. *coordinate exposure, requiring every vector read used by the comparison to pass through the finite coordinate window K and the* $\mathsf{Mat}^{\uparrow}$ *reconstruction ledger;*

3. *norm-source compatibility, requiring the two norm rows A and B to read the same displayed vector source V and the same* $\mathsf{Real}^{\uparrow}$ *comparison boundary R;*

4. *compact finite-net lower bound, requiring L to supply a positive lower comparison for the second norm on the first-norm unit coordinate window by finite mesh inspection;*

5. *compact finite-net upper bound, requiring U to supply an upper comparison for the second norm on the same displayed coordinate window;*

6. *exactness, requiring every exported comparison constant and every downstream norm-equivalence read to factor through $V, K, A, B, R, L, U, H, T, P, N$.*

No obligation row may be discharged by host basis selection, quotient vector equality, an infinite-dimensional separation theorem, or a choice-selected dual functional.

Proof. Carrier admission is projection from Definition 5566.1. The accepted packet lists the vector source, coordinate table, two norm rows, real comparison boundary, finite-net lower and upper bound rows, and the structural replay rows.

Coordinate exposure is finite. A consumer first reads the displayed $\mathsf{Mat}^{\uparrow}$ coordinate table K, then reconstructs vector operations through the $\mathsf{VecSpace}^{\uparrow}$ row V. Since K is a coordinate window rather than a host basis object, the proof uses only the listed reconstruction ledger.

The lower and upper comparison rows are compact finite-net reads. The row L inspects the first-norm unit coordinate window through the closed bounded interval net and records a positive lower comparison for the second norm. The row U reads the same finite window and records the upper comparison. Both rows are sealed by $\mathsf{Real}^{\uparrow}$ comparisons in R.

Finally H, T, P, N transport, replay, source, and name this finite route. The carrier contains no host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach theorem, or choice-selected dual row, so the listed obligations exhaust the local NameCert surface. □

Lean4 (definition) `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceCarrier`
Lean4 ✓ `BEDC.Derived.FiniteDimensionalNormEquivalenceUp.FiniteDimensionalNormEquivalenceNameCertObligat`

Theorem 5566.3 (Finite-dimensional compact-coordinate norm comparison)**.** *Every* $\mathsf{NormedSpace}^{\uparrow}$ *or bounded-operator consumer of an accepted* $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ *carrier receives the comparison route*

$$V, K, A, B, R, L, U \longmapsto H, T, P, N.$$

The route compares the two norms by finite coordinate windows and compact finite-net lower and upper bounds. It exports no ambient basis theorem, quotient-vector equality, infinite-dimensional Hahn–Banach argument, selected dual basis, or choice principle.

Proof. Start with a downstream norm-comparison request. The vector input is read through V, and its finite coordinate presentation is read through the $\mathsf{Mat}^{\uparrow}$ window K.

Read the two norms next. The rows A and B give norm values on the same displayed vector source, while R supplies the real comparison boundary for inequalities and constants.

The compact-coordinate step uses the finite net. The lower row L supplies the positive comparison on the first-norm unit coordinate window, and the upper row U supplies the matching upper comparison. Both are finite mesh reads through chapter 1250, so no point is selected from an ambient compact object.

The structural rows H, T, P, N replay this exact route to $\mathsf{NormedSpace}^{\uparrow}$ and bounded-operator consumers. Since the packet never stores host basis choice, quotient vector equality, infinite-dimensional Hahn–Banach, selected duals, or choice, the exported comparison remains a finite NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$ packet is generated from a displayed $\mathsf{VecSpace}^{\uparrow}$ row, a finite $\mathsf{Mat}^{\uparrow}$ coordinate window, two $\mathsf{Norm}^{\uparrow}$ rows, a $\mathsf{Real}^{\uparrow}$ comparison boundary, compact finite-net lower and upper bound rows, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5566.1, Definition 5566.2, and Definition 5566.3 seed the finite coordinate norm-comparison route over $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host basis existence, quotient vector equality, infinite-dimensional norm equivalence, Hahn–Banach separation, selected dual bases, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite coordinate exposure, compact finite-net lower and upper comparison rows, componentwise replay, and local NameCert exactness for BEDC.Derived.FiniteDimensionalNormEquivalence

Chapter 5567

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 5567.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 5567.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 5567.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5568

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5568.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 5568.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5568.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 5568.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 5569

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 5569.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 5569.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 5569.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 5570

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 5570.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 5570.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5570.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 5571

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5571.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5571.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5571.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5571.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5571.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5572

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5572.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5572.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5572.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5572.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5573

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5573.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompletionModulusCategory}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5573.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any $\mathsf{Real}^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5573.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any $\mathsf{Real}^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompletionFunctor}^{\uparrow}$, *and* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5574

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5574.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5574.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5574.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5574.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5575

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5575.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 5575.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5575.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5576

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5576.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5576.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5576.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5577

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5577.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5577.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5578

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5578.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5578.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5578.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5579

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5579.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5579.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5579.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5579.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5580

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5580.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5580.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5580.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5581

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5581.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5581.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5581.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5582

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5582.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5582.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5582.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5583

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5583.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5583.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5583.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5583.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5584

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5584.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5584.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5584.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5584.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5585

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5585.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5585.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5585.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5586

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5586.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5586.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5586.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5586.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5586.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5587

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5587.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5587.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5587.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5588

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5588.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5588.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5588.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5588.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5588.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5588.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5589

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5589.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5589.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5589.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5590

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5590.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5590.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5590.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5591

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5591.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5591.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5591.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5591.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5592

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5592.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5592.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5592.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5593

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5593.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5593.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5593.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5593.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5594

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5594.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. *Ω is the finite trigger-coordinate row.*
2. *A is the finite support-family row whose entries are displayed subsets of Ω.*
3. *U is the cover ledger asserting that the displayed supports cover every coordinate of Ω.*
4. *V is the private-coordinate witness ledger, with one row for each support in A.*
5. *π maps each support row to its selected private coordinate and records membership in that support.*
6. *E records exclusion from every other support row for the selected private coordinate.*
7. *K is the minimal-cover criterion row, readable only after U, V, π, E have been displayed.*
8. *H stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5594.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5594.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5594.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5595

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5595.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5595.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5595.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5596

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5596.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5596.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5596.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5597

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5597.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5597.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5597.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5597.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5597.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5598

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5598.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5598.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5599

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5599.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5599.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5599.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5600

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5600.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5600.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5600.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5601

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5601.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5601.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5601.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5602

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5602.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5602.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5602.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5603

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5603.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5603.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5604

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5604.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5604.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5604.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5605

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5605.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5605.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5605.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5606

A Concrete Naming Certificate for $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readbacks of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ budgets of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38.

Definition 5606.1 (CauchyProductCompletionProjection carrier)**.** *A* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* $\mathsf{CauchyProductCompletion}^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* $\mathsf{StreamName}^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$, $\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5606.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5606.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyProductCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameC}$$

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5607

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5607.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5607.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5607.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 5608

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5608.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5608.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5608.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5609

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5609.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5609.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5609.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5610

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5610.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5610.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5610.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5610.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5611

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5611.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5611.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5611.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5612

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5612.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5612.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5612.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5613

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5613.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5613.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5613.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5614

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5614.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5614.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5614.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\text{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5615

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5615.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5615.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5615.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5616

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5616.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5616.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5616.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5617

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 5617.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 5617.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 5617.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5618

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5618.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5618.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5619

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5619.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate single carrier alignment`

Theorem 5619.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5619.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5620

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 5620.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 5620.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5620.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 5620.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5621

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5621.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5621.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*
2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*
3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*
4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*
5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*
6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*
7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5621.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5621.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5622

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5622.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5622.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5622.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5623

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5623.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5623.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5623.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5624

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5624.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5624.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5624.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5625

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5625.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5625.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForm

Chapter 5626

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5626.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5626.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5627

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

5627.1 Carrier definition

Definition 5627.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

5627.2 Certificate surface

Theorem 5627.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 5627.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 5627.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 5628

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 5628.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 5628.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 5628.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5629

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5629.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5629.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5629.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5629.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5629.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5630

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5630.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5630.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5630.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5630.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5631

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5631.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5631.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5631.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5632

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5632.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5632.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5632.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5632.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5633

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5633.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 5633.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5633.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5634

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5634.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5634.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5634.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5635

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5635.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate single carrier alignment`

Theorem 5635.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5636

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5636.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5636.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5636.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5637

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5637.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5637.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5637.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5637.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5638

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5638.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5638.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5638.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5639

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5639.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5639.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5639.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Wind}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5640

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5640.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate`
`single carrier alignment`

Theorem 5640.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5640.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5641

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5641.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LipschitzRealMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5641.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* $\mathsf{Real}^{\uparrow}$ *seal, target* $\mathsf{Real}^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5641.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5641.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5642

A Concrete Naming Certificate for $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$

The $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5642.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* $\mathsf{UnitDisk}^{\uparrow}$ *coordinates and the associated* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* $\mathsf{HyperbolicGeodesicLamination}^{\uparrow}$*.*
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5642.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5642.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5642.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide $\mathsf{Nontrivial}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5643

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5643.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5643.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5643.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5644

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5644.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5644.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5644.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5644.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5644.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5645

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5645.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5645.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5645.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5646

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5646.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5646.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5646.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5646.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5646.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5646.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5647

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5647.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5647.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5647.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5648

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5648.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5648.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5648.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5649

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5649.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5649.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5649.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5649.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5650

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5650.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5650.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5650.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5651

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5651.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5651.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5651.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5651.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5652

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5652.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5652.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5652.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5652.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5653

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5653.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5653.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5653.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5654

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5654.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5654.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5654.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5655

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5655.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5655.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5655.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5655.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5655.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5656

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5656.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^\uparrow$, CompactMetric$^\uparrow$, ContinuousMap$^\uparrow$, DiniTheorem$^\uparrow$, UniformModulus$^\uparrow$, DyadicRatCore$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^\uparrow}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5656.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^\uparrow$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^\uparrow$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^\uparrow$ *budget sealed by* Real$^\uparrow$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^\uparrow$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^\uparrow$ consumer that exports a UniformModulus$^\uparrow$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^\uparrow$ budget cells, Real$^\uparrow$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^\uparrow$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^\uparrow}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^\uparrow}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5657

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5657.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5657.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^\uparrow$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{Real}^\uparrow$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^\uparrow$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5657.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^\uparrow$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^\uparrow$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^\uparrow$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^\uparrow$ *readback before the terminal* $\mathsf{Real}^\uparrow$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^\uparrow$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^\uparrow$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5658

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5658.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5658.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5658.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and FieldFaithful DNAWindowSi TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5659

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5659.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5659.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5659.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5660

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5660.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5660.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5660.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5661

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5661.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5661.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5662

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5662.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5662.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5662.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5663

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5663.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5663.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5663.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5664

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5664.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5664.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5664.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5665

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5665.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5665.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5665.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 5666

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 5666.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 5666.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 5666.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 5666.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 5666.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 5666.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 5666.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 5666.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, obligation)

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 5667

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5667.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5667.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5667.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5668

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5668.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5668.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5668.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, seed)

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5669

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5669.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5669.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5669.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5669.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5670

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 5670.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 5670.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5670.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 5670.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 5670.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 5670.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 5671

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5671.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5671.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5671.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5672

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5672.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5672.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5672.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5673

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5673.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist/hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5673.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5673.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5674

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5674.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5674.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5674.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5675

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5675.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5675.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5675.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5676

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5676.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5676.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5676.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5677

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 5677.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 5677.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 5677.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 5677.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5678

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5678.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5678.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*
2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*
3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*
4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*
5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*
6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*
7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5678.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5678.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5679

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5679.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5679.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5679.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5680

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 5680.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 5680.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 5680.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 5680.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 5680.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 5681

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5681.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5681.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5681.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5682

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5682.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5682.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5682.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5683

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5683.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5683.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5684

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5684.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5684.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5685

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 5685.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 5685.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 5686

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 5686.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 5686.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 5687

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 5687.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 5687.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 5687.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 5688

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 5688.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 5688.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 5688.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 5689

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 5689.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 5689.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 5689.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 5689.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 5690

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 5690.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 5690.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 5690.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5691

A Concrete Naming Certificate for RiemannIntegralCauchy$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegralCauchy$^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient Real$^{\uparrow}$ completeness.

Definition 5691.1 (Riemann integral Cauchy carrier)**.** *A* RiemannIntegralCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* RiemannSumGauge$^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* DarbouxSum$^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* RiemannIntegrable$^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* RiemannIntegralCauchy$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 5691.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 5691.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 5692

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 5692.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 5692.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 5692.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 5693

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5693.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate
single carrier alignment

Theorem 5693.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 5693.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 5693.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 5694

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 5694.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 5694.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 5694.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 5694.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 5695

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 5695.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 5695.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 5695.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 5695.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 5695.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 5695.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 5696

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 5696.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* BanachFixedPointStability$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, PicardIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 5696.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* NameCert$_{\text{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5696.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *consumer of an accepted* BanachFixedPointStability$^{\uparrow}$ *packet factors through the finite route*

$$\text{BanachFixedPoint}^{\uparrow} \longmapsto \text{PicardIteration}^{\uparrow} \longmapsto \text{ContractionMapping}^{\uparrow} \longmapsto \text{CompleteMetric}^{\uparrow} \longmapsto \text{CauchyModulus}^{\uparrow} \longmapsto \text{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* Real$^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the RegSeqRat$^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The Real$^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 5696.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 5697

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 5697.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 5697.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 5698

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 5698.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 5698.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 5698.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 5698.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$.

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, seed)

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 5699

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 5699.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 5699.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 5699.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 5700

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 5700.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 5701

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 5701.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 5701.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 5702

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 5702.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 5702.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 5702.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 5703

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5703.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 5703.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 5703.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 5703.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 5703.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{Bool}^{\uparrow}$, and $\mathsf{Nat}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \text{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 5704

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 5704.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 5704.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 5704.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5705

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5705.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 5705.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5705.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 5705.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, seed)

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 5706

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 5706.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 5706.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 5707

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5707.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 5707.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5707.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 5707.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 5707.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 5707.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 5708

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 5708.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 5708.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5708.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 5709

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 5709.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 5709.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 5709.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 5710

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 5710.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 5710.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 5710.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 5710.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, seed)

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 5711

A Concrete Naming Certificate for $\mathsf{BisectionReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BisectionReal}^{\uparrow}$ packet records bisection as a finite nested-interval and bracket-refinement certificate over the L10 Real route. It sits after the dyadic tolerance surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested interval route of chapter 1924. The packet names the displayed bisection ledger; it is not a Dedekind cut, quotient real, trichotomy principle, selected root, or completeness theorem.

Definition 5711.1 (BisectionReal carrier)**.** *A* $\mathsf{BisectionReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, M, S, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested interval source row, recording the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, interval containment, and the finite retained subintervals consumed by the bisection route.*
2. *M is the dyadic midpoint and tolerance ledger. It records the midpoint rows, branch radii, diameter budgets, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparisons used at each inspected stage.*
3. *S is the sign or bracket-preservation row, storing the displayed bracket invariant that survives each retained child interval.*
4. *T is the tail request row, recording the finite precision request and the inspected bisection depth needed to answer it.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for the retained endpoint and midpoint names.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from W and M.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the real name exported by the retained nested interval route.*
8. *H records componentwise* hsame *transport for the interval source, midpoint ledger, bracket row, tail request, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a bisection request through interval inspection, midpoint production, bracket preservation, dyadic tolerance readback, stream-window exposure, regular readback, and Real-seal consumption.*

10. *P is the* Pkg *provenance over* $\mathsf{BisectionReal}^{\uparrow}$, $\mathsf{NestedIntervalReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, S, T, W, R, E, H, C, P, N$. It has no coordinate for Dedekind cuts, quotient-real representatives, real trichotomy, selected roots, host equality of real points, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BisectionRealUp.BisectionRealTasteGate single carrier alignment`

Theorem 5711.2 (BisectionReal NameCert obligations)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier*

$$B = (I, M, S, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the nested interval source, dyadic midpoint and tolerance ledger, bracket-preservation row, tail request, stream windows, regular readback, Real seal, transport, replay, provenance, and local naming;*
2. *nested-source discipline, requiring every retained interval read to pass through I before midpoint or Real-facing data are available;*
3. *midpoint and tolerance discipline, requiring M to expose only dyadic midpoint, radius, and diameter rows whose budgets answer the request T;*
4. *bracket preservation, requiring S to carry the displayed sign or bracket invariant across each retained child interval;*
5. *readback discipline, requiring W and R to expose the retained interval only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows;*
6. *seal nonescape, requiring E, H, C, P, N to export only the displayed bisection route and no Dedekind cut, quotient real, trichotomy principle, selected root, host equality, countable choice, or completeness theorem.*

No BisectionReal obligation is stored outside these rows.

Proof. Project the carrier of Definition 5711.1. The accepted packet displays exactly the nested interval source, midpoint ledger, bracket row, tail request, window rows, readback, Real seal, and structural rows.

The finite source stage is I, T. A precision request opens the source interval row and the requested bisection depth before any midpoint or retained endpoint can be inspected.

The bisection stage is M, S. A finite induction over the retained branch ledger reads one midpoint row at a time, uses the dyadic tolerance entries to bound the retained child, and carries the displayed sign or bracket invariant into the next interval row.

The readback stage is W, R, E. The retained endpoints are exposed as finite stream windows, then as regular rational readbacks, and only then as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, package, and name the same route. Since the carrier contains no Dedekind-cut, quotient-real, trichotomy, selected-root, host-equality, countable-choice, or completeness coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5711.3 (BisectionReal nested interval factorization)**.** *Every* $\mathsf{BisectionReal}^{\uparrow}$ *read factors through the displayed nested interval route*

$$I, T \longmapsto M, S \longmapsto W, R \longmapsto E, H, C, P, N.$$

In particular, a Real-facing consumer first reads the $\mathsf{NestedIntervalReal}^{\uparrow}$ *source row* I*, then the finite dyadic midpoint and bracket-preservation rows* M, S*, then the finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows* W, R*, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal and local* $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$ *row.*

Proof. Start with the accepted carrier $B = (I, M, S, T, W, R, E, H, C, P, N)$. The only interval source coordinate is I, and the only request coordinate is T, so a bisection read begins by exposing those rows.

Induct on the finite branch prefix recorded in the midpoint ledger. The base case reads the initial interval and its bracket row. In the successor step, the midpoint row in M supplies the dyadic split, the tolerance entry supplies the diameter budget, and S records that the displayed bracket invariant is preserved for the retained child.

After the last retained child, the endpoint windows are read through W and the rational data through R. The Real-facing seal E is therefore downstream from the nested interval, midpoint, bracket, stream, and regular readback rows.

The structural rows H, C, P, N do not add a second route. They transport componentwise hsame, replay the same displayed Cont chain, record Pkg provenance, and supply the local name. Thus every read factors through the stated finite route. □

Theorem 5711.4 (BisectionReal regular Cauchy handoff)**.** *For every accepted* $\mathsf{BisectionReal}^{\uparrow}$ *carrier* $B = (I, M, S, T, W, R, E, H, C, P, N)$*, the handoff to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *is available only after the retained nested-interval rows, dyadic midpoint and tolerance rows, sign or bracket-preservation rows, and finite* $\mathsf{StreamName}^{\uparrow}$ *windows have been replayed through the same* BHist *packet. The exported object is a regular Cauchy readback with a Real seal, not a selected root, quotient representative, trichotomy decision, or completeness witness.*

Proof. Use the nested interval factorization of Definition 5711.3. It places W, R, E strictly after the source row I, request row T, midpoint ledger M, and bracket row S.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the row R. It is computed from the finite stream windows in W and the dyadic tolerance entries in M, so no downstream consumer can read R without replaying the retained bisection ledger.

The $\mathsf{Real}^{\uparrow}$ handoff is the row E. It seals the regular readback already obtained from R, while H, C, P, N preserve the same packet by componentwise hsame, displayed Cont, Pkg, and local NameCert data.

Since the carrier has no selected-root, quotient-representative, trichotomy-decision, host-equality, countable-choice, Dedekind-cut, or completeness coordinate, the handoff exposes only the regular Cauchy route stated in the theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BisectionReal}^{\uparrow}$ packet is generated from a nested interval source row, dyadic midpoint and tolerance ledger, sign or bracket-preservation rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, seed)

Scope closed: Definition 5711.1, Definition 5711.2, Definition 5711.3, and Definition 5711.4 seed the finite constructive bisection route from nested interval rows through dyadic midpoint, bracket-preservation, stream-window, regular readback, and Real-seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BisectionReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind cuts, quotient reals, trichotomy on reals, selected roots, host equality of real points, countable choice, ambient $\mathrm{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source discipline, midpoint and tolerance exactness, bracket preservation, regular Cauchy handoff, RealUp seal nonescape, and local naming for BEDC.Derived.BisectionRealUp.

Chapter 5712

A Concrete Naming Certificate for $\mathsf{DarbouxCriterion}^{\uparrow}$

Following Darboux, Riemann, and Bishop–Bridges constructive integration, the $\mathsf{DarbouxCriterion}^{\uparrow}$ packet records the upper-lower gap test as finite NameCert evidence. It sits between the lower and upper Darboux sum packets of chapter 5858 and chapter 4083, the Darboux-sum surface of chapter 1880, and the Riemann-facing consumers of chapter 1960 and chapter 1526. The packet exposes a displayed partition, lower sum, upper sum, gap, tolerance, regular rational readback, and real seal; it does not import host suprema, measure completion, a selected mesh, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5712.1 (Darboux criterion carrier)**.** *A* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, L, U, G, T, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition row, recording the inspected interval endpoints and refinement requests.*
2. *L is the* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *row attached to P.*
3. *U is the* ***??*** *row attached to the same partition row P.*
4. *G is the upper-minus-lower gap row comparing U and L inside the displayed rational ledger.*
5. *T is the tolerance row, recording the positive rational request against which G is checked.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the same finite partition and gap evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after P, L, U, G, T, R have been displayed.*
8. *H records componentwise* hsame *transport for partition, lower-sum, upper-sum, gap, tolerance, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a tolerance request through P, L, U, G, T, R, E.*
10. *Q is the* Pkg *provenance over* $\mathsf{DarbouxCriterion}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{LowerDarbouxSum}^{\uparrow}$, ***??***, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host suprema or infima, measure completion, a selected mesh, an ambient completeness principle, a quotient real, or a global integration theorem.

Theorem 5712.2 (Darboux criterion NameCert obligations)**.** *For every accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *carrier*

$$K = (P, L, U, G, T, R, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, partition exposure, lower-sum and upper-sum compatibility over the same partition, gap comparison against the tolerance row, regular rational readback, terminal real-seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $P, L, U, G, T, R, E, H, C, Q, N$.

Lean4 ✓ `BEDC.Derived.DarbouxCriterionUp.DarbouxCriterionTasteGate single carrier alignment`

Proof. Project the carrier. The accepted packet exposes the partition row P, the paired lower and upper Darboux sum rows L, U, the gap row G, the tolerance row T, the regular readback row R, the seal row E, and the structural rows H, C, Q, N.

The lower and upper sum rows are tied to the same partition row, so a gap comparison cannot switch partitions while the criterion is being read. The tolerance row fixes the request against which the displayed gap is checked.

The regular readback and real seal are downstream of those finite rows. Componentwise transport and replay preserve the same order, while provenance and naming record the listed sibling packets.

Since host suprema, measure completion, selected meshes, quotient reals, ambient completeness, and global integration theorems are absent from the carrier coordinates, the displayed rows exhaust the local NameCert surface. □

Theorem 5712.3 (Darboux criterion upper-lower gap route)**.** *Every* $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, *or regulated-integral-facing read of an accepted* $\mathsf{DarbouxCriterion}^{\uparrow}$ *packet factors through the finite route*

$$P, L, U \longmapsto G, T \longmapsto R \longmapsto E.$$

Thus the consumer first reads a displayed partition, lower Darboux sum, and upper Darboux sum; then it reads the upper-minus-lower gap and tolerance rows; then it reads the $\mathsf{RegSeqRat}^{\uparrow}$ *hand-off and terminal* $\mathsf{Real}^{\uparrow}$ *seal. The packet exports no direct route through host suprema, measure completion, selected mesh choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 5712.2. They require carrier admission and classifier transport through the displayed rows of Definition 5712.1.

The partition row P is read before either sum row can be used. The rows L and U are therefore paired over the same finite partition, and G is the displayed comparison between those paired rows.

The tolerance row T is the only request against which the gap is consumed. After that finite comparison has been displayed, the read may pass to R and then to the real seal row E.

The excluded routes are not coordinates of the carrier. Hence every integration-facing consumer receives the upper-lower gap criterion only through the finite partition, sum, gap, tolerance, regular-readback, and seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DarbouxCriterion}^{\uparrow}$ packet is generated from a displayed partition, lower and upper Darboux sum rows, an upper-minus-lower gap row, a tolerance row, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DarbouxCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5712.1, Definition 5712.2, and Definition 5712.3 seed the finite upper-lower gap criterion for Riemann-facing and regulated-integral-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{DarbouxCriterion}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host suprema, measure completion, selected mesh choice, quotient real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, partition exposure, paired lower and upper sum compatibility, gap tolerance comparison, regular readback, real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.DarbouxCriterionUp.

Chapter 5713

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 5713.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 5713.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 5713.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 5714

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 5714.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. K *is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. N *is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. M *is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. F *is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. U *is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. T *is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. H *records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a compact-uniform tolerance request through* $K, N, M, F, U, T, H, P, L$.
9. P *is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 5714.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 5714.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 5715

A Concrete Naming Certificate for $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet records the explicit finite dyadic closed-interval net route for regular Cauchy real names. It sits over the dyadic arithmetic core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the finite-net Heine–Borel surface of chapter 3471. The chapter is a finite dyadic closed-interval net route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$; it is not general Heine–Borel for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, does not import open-cover compactness, and does not select limits outside the displayed interval-net data.

Definition 5715.1 (Regular Cauchy Heine–Borel route carrier)**.** *A* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the closed bounded regular-Cauchy interval row, exposing two endpoint* $\mathsf{RegSeqRat}^{\uparrow}$ *names, their* $\mathsf{StreamName}^{\uparrow}$ *finite windows, and the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh row, recording the finite dyadic subdivision, endpoint comparisons, cell widths, and requested tolerance bound.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that lists the finite observations used to compare each regular-Cauchy endpoint and each inspected interval cell.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint names and the retained dyadic cell names.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, Q have been displayed.*
6. *W is the closed-interval witness row, pairing every retained dyadic cell with its lower and upper endpoint comparisons inside I.*
7. *F is the finite net row, containing only the finite bundle of dyadic cell centers and radius witnesses read by the interval-net route.*
8. *H records componentwise* hsame *transport for the interval, dyadic mesh, stream windows, regular readback, real seal, cell witnesses, finite net, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a closed-interval net request through* I, D, S, Q, E, W, F, H.
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, Q, E, W, F, H, C, P, N$. *It has no coordinate for an arbitrary subset of* $\mathsf{Real}^{\uparrow}$, *arbitrary open covers, selected finite subcovers, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyHeineBorelRouteUp.TasteGate.RegularCauchyHeineBorelRouteTasteGate single carrier alignment`

Theorem 5715.2 (Regular Cauchy Heine–Borel route obligations)**.** *For every accepted* RegularCauchyHeineBorelRo *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the closed bounded regular-Cauchy interval row I before any interval-net consumer reads a cell;*
2. *display the dyadic subdivision and tolerance ledger D as the only mesh source;*
3. *route endpoint and cell observations through the finite* $\mathsf{StreamName}^{\uparrow}$ *windows S and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q;*
4. *reach the terminal* $\mathsf{Real}^{\uparrow}$ *seal E only after the dyadic, stream, and regular-rational rows have been displayed;*
5. *bind each retained dyadic interval cell to the closed interval by the witness row W and export the finite net bundle F;*
6. *preserve the route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
7. *source and name the same finite packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, *arbitrary open-cover compactness, a selected subcover functional, quotient real equality, countable choice, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5715.1. The interval row I is the only source for the closed bounded regular-Cauchy endpoints, and the route cannot inspect an interval cell before those endpoint rows are present.

Next read the dyadic mesh and observation rows. The dyadic ledger D supplies the finite subdivision and the tolerance comparison, while S and Q supply the finite windows and regular-rational readbacks needed to compare endpoint and cell data.

The Real-facing boundary is downstream. The terminal row E is reached only after the dyadic, stream, and regular-rational rows have been displayed, so the route exports a named real seal rather than a host-selected real limit or quotient equality.

Finally the finite-net part is exhausted by W and F. The witness row W binds each retained dyadic cell to the closed interval, and F lists the finite cell centers and radii. The structural rows H, C, P, N transport, replay, source, and name this same packet, so no arbitrary subset, open cover, selected subcover, choice, selected-limit, or ambient-completeness coordinate remains. □

Theorem 5715.3 (Regular Cauchy Heine–Borel finite interval-net route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{HeineBorelFiniteNet}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{RegularCauchy}}$$

The route supplies an explicit finite dyadic closed-interval net for closed bounded regular-Cauchy interval data. It supplies no compactness theorem for arbitrary subsets of $\mathsf{Real}^{\uparrow}$*, no arbitrary open-cover subcover extraction, no quotient real equality, no countable choice, and no* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with a consumer read of the carrier. The only numerical mesh coordinate is D, and every inspected endpoint or cell must be read through the finite windows S and regular-rational row Q before the terminal seal E is available.

The closed-interval part is then bound by W. Each retained dyadic cell is paired with displayed lower and upper endpoint comparisons, so the interval-net route remains a finite dyadic route inside the presented closed bounded interval.

The finite net export is exactly F. It lists the dyadic centers and radius witnesses read by the Heine–Borel finite-net surface, while H, C, P, N preserve and name the same route. Since the carrier contains none of the excluded arbitrary-subset, open-cover, quotient, choice, selected-limit, or completeness coordinates, every such consumer remains on the displayed finite interval-net route. □

Theorem 5715.4 (Regular Cauchy Heine–Borel inductive subcover route)**.** *Every finite interval-net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier may build subcover reads only by induction over the displayed dyadic cell list* F*. The base read over the empty cell list exports no cover row. The successor read appends one displayed dyadic cell, routes endpoint and cell comparisons through* D, S, Q, E, W*, and preserves the same* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows. No read imports arbitrary open-cover compactness, selected finite subcovers, countable choice, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. The finite net row F is the only coordinate that lists dyadic cells, and the rows D, S, Q, E, W are the only coordinates available for mesh, window, readback, Real seal, and closed-interval comparisons.

The induction base is the empty displayed cell list. With no cell in F, the consumer can read the interval, dyadic mesh, stream windows, regular readback, and structural rows, but it has no cover row to export.

For the successor step, append one displayed dyadic cell from F. Its endpoint and cell comparisons are routed through the dyadic mesh D, finite windows S, regular readback Q, terminal seal E, and witness row W before the consumer extends the finite read.

The structural rows H, C, P, N are preserved componentwise. They transport, replay, source, and name the same finite induction route, so no arbitrary open cover, selected subcover, countable-choice schedule, quotient equality, or ambient completeness principle is introduced. □

Theorem 5715.5 (Regular Cauchy Heine–Borel route obligation scope)**.** *Every accepted* RegularCauchyHeineBorelR *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

scopes its obligation rows through finite BEDC primitives as follows:

1. *I is scoped by the* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint rows and the closed bounded regular-Cauchy interval data;*
2. *D is scoped by the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh and tolerance rows;*
3. *S is scoped by finite* $\mathsf{StreamName}^{\uparrow}$ *windows;*
4. *Q is scoped by* $\mathsf{RegSeqRat}^{\uparrow}$ *readback;*
5. *E is scoped by the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
6. *W, F are scoped by the* $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ *finite closed-interval net route;*
7. *H, C, P, N are scoped respectively by componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row.*

The scoped surface exports no arbitrary subset compactness, open-cover subcover selector, quotient real equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its rows are exactly $I, D, S, Q, E, W, F, H, C, P, N$, so the obligation theorem can inspect no coordinate outside the displayed packet.

Apply Definition 5715.2. The interval row, dyadic mesh, finite stream windows, regular-rational readback, Real seal, closed-interval witnesses, finite net bundle, transport, replay, provenance, and local naming rows are precisely the finite obligations available to the local certificate.

Use Definition 5715.3. It identifies the consumer route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, with hsame, Cont, Pkg, and $\mathsf{NameCert}$ preserving the same displayed packet. The excluded compactness, quotient, selector, choice, selected-limit, and completeness readings therefore have no scoped row. □

Theorem 5715.6 (Regular Cauchy Heine–Borel closed-interval net scope)**.** *Every closed-interval net consumer of an accepted* $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ *carrier*

$$R = (I, D, S, Q, E, W, F, H, C, P, N)$$

must read the displayed rows in the order

$$I,\ D,\ S,\ Q,\ E,\ W,\ F,\ H,\ C,\ P,\ N.$$

Here I binds the closed bounded $\mathsf{RegSeqRat}^{\uparrow}$ *interval, D supplies the dyadic mesh, S and Q provide the finite* $\mathsf{StreamName}^{\uparrow}$ *observations and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and W, F are the finite closed-interval witness row and finite interval-net bundle. The remaining rows H, C, P, N transport, replay, source, and name the same route. Thus the consumer sees only the displayed closed-interval net data and imports no arbitrary open-cover compactness, selected subcover functional, quotient real equality, countable choice, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5715.1. Its row list is exactly $I, D, S, Q, E, W, F, H, C, P, N$, and there is no coordinate before I from which a consumer could infer a cell, mesh, cover, or selected limit.

Apply Definition 5715.2. The obligation surface forces the closed bounded interval row I to be exposed before any interval-net read, then forces the mesh D, finite windows S, regular-rational readback Q, and terminal Real seal E before the closed-interval witness and finite-net rows can be exported.

Use Definition 5715.3 and Definition 5715.4. They identify W, F as the finite closed-interval net surface and restrict subcover reads to induction over the displayed dyadic cell list, with endpoint and cell comparisons routed through D, S, Q, E, W.

Finally apply Definition 5715.5. It scopes H, C, P, N as componentwise transport, displayed replay, provenance, and local naming for the same packet. The composed read order is therefore the displayed order $I, D, S, Q, E, W, F, H, C, P, N$, and the excluded compactness, quotient, selector, choice, selected-limit, and completeness principles have no row to enter through. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}$ packet is generated from a closed bounded regular-Cauchy interval row, a DyadicRatCore mesh row, StreamName windows, RegSeqRat readback, a RealUp seal, closed-interval cell witnesses, a finite dyadic net row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5715.1, Definition 5715.2, and Definition 5715.3 seed the explicit finite dyadic closed-interval net route over DyadicRatCore, StreamName, RegSeqRat, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyHeineBorelRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness for arbitrary subsets of $\mathsf{Real}^{\uparrow}$, arbitrary open-cover compactness, selected finite subcovers, quotient real equality, countable choice, selected limits, Lean verification, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite stream-window readback, regular-rational endpoint routing, Real seal nonescape, finite interval-net exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyHeineBorelRouteUp.

Chapter 5716

A Concrete Naming Certificate for $\mathsf{CauchyApproximationTree}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet records a finite branching approximation surface over the L10 Cauchy spine: each active branch carries a displayed tolerance window, a dyadic approximation row, a regular-rational readback row, and a terminal Real seal. It sits between the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the real-name seal of chapter 38, and the Cauchy route of chapter 3413. The packet gives Real and Completion consumers a finite approximation tree without quotient streams, selected limits, countable choice, host real equality, trichotomy, or ambient completeness.

Definition 5716.1 (Cauchy approximation tree carrier)**.** *A* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, B, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *V is the finite tree-node row, listing the nodes presently opened by the approximation packet.*
2. *E is the finite edge and parenthood row, recording the branch relation among the nodes in V.*
3. *B is the branch-window row, assigning each active branch the finite tolerance window currently inspected.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic approximants, radius comparisons, and the error rows attached to each branch window.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the branch windows and no hidden stream coordinates.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows and dyadic tolerance rows.*
7. *L is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the node, branch-window, dyadic, stream, and regular-readback rows have been exposed.*
8. *H records componentwise* hsame *transport for tree nodes, branch windows, dyadic rows, stream schedules, regular readbacks, Real seals, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a finite tree branch through B, D, S, R and into the terminal Real seal L.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyApproximationTree}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $V, E, B, D, S, R, L, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyApproximationTreeUp.CauchyApproximationTreeTasteGate single carrier alignment`

Theorem 5716.2 (Cauchy approximation tree NameCert obligations)**.** *For every accepted* CauchyApproximationTre *carrier $T = (V, E, B, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the tree-node row, edge row, branch windows, dyadic tolerance ledger, stream schedule, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *tree finiteness, requiring every inspected branch to be generated by the finite node and edge rows V, E;*
3. *branch-window exactness, requiring each active branch to read its tolerance window from B before any dyadic approximation in D is accepted;*
4. *dyadic and stream exactness, requiring D and S to expose only the dyadic approximants and finite observations demanded by the displayed branch windows;*
5. *regular readback and Real-seal boundary, requiring R and L to remain downstream of V, E, B, D, S and never to serve as selected limits or host equality objects;*
6. *componentwise transport and replay, requiring H and C to preserve exactly the displayed finite branch route;*
7. *provenance and local naming, requiring P and N to export the route over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{Cauchy}^{\uparrow}$ *without adding quotient equality, choice, trichotomy, or ambient completeness.*

No approximation-tree obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5716.1. The accepted packet displays exactly the finite node row V, edge row E, branch windows B, dyadic ledger D, stream schedule S, regular readback R, Real seal L, and structural rows H, C, P, N.

Tree finiteness and branch-window exactness follow from the row order. A consumer first reads a node and its branch relation from V, E; only then may it inspect the branch tolerance window B and the dyadic approximation row D attached to that branch.

The stream and readback rows are downstream of the same branch window. The schedule S opens only the finite observations demanded by B, and R records the regular-rational data justified by D and S.

Finally, the Real seal row L is terminal for this finite branch route. The transport row H, replay row C, provenance row P, and local naming row N preserve and name the same displayed packet, so no quotient stream equality, selected limit, countable-choice selector, host real equality, trichotomy principle, or ambient completeness theorem can become a local obligation. □

Theorem 5716.3 (Cauchy approximation tree branch readback)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Cauchy}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyApproximationTree}^{\uparrow}$ *carrier reads a branch only through the displayed route*

$$V, E, B, D, S, R, L, H, C, P, N.$$

Consequently each consumed branch exposes finite tree nodes, branch windows, dyadic tolerance rows, StreamName observations, RegSeqRat readbacks, a Real seal, componentwise transport, displayed replay, provenance, and local naming data. It exposes no quotient stream equality, selected limit, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let a Real-facing or Cauchy-facing consumer read an accepted packet $T = (V, E, B, D, S, R, L, H, C, P, N)$. Carrier projection gives access only to the rows named in Definition 5716.1.

The read begins with the finite tree rows V, E and then chooses a displayed branch window in B. The dyadic ledger D supplies the approximant and tolerance comparison for that branch, the stream schedule S opens the finite observation windows, and the regular readback row R exposes the rational data that the Real seal may consume.

The terminal row L is therefore reached only after the finite route through V, E, B, D, S, R has been displayed. The structural rows H, C, P, N transport, replay, source, and name the same route without adding any second equality or limit coordinate.

Since the packet has no carrier row for quotient stream equality, selected limits, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness, a Real-facing or Cauchy-facing consumer cannot read any of those objects from the local certificate. □

Falsifiable prediction: Every $\mathsf{CauchyApproximationTree}^{\uparrow}$ Real-facing or Cauchy-facing read factors through finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, selected StreamName observations, RegSeqRat readbacks, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy: $\mathsf{CauchyApproximationTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite tree nodes, branch windows, dyadic tolerance rows, finite stream schedules, regular-rational readback, a terminal Real seal, transport, replay, provenance, and local naming in one branch-indexed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyApproximationTree}^{\uparrow}$ packet is generated from finite tree nodes, branch windows, a DyadicRatCore tolerance ledger, StreamName finite observations, RegSeqRat readback rows, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5716.1, Definition 5716.2, and Definition 5716.3 seed the finite branch-indexed route from L10 Cauchy approximation windows to RealUp endpoint consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyApproximationTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree finiteness, branch-window exactness, dyadic ledger exactness, stream and readback exactness, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.CauchyApproximationTreeUp.

Chapter 5717

A Concrete Naming Certificate for $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet records the finite comparison surface between Bishop-style Cauchy representations and the BEDC real-name ladder. It sits after the Cauchy-representation route of chapter 1680, the precision ledger of chapter 4141, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the comparison route as finite NameCert data; it is not quotient stream equality, selected host limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5717.1 (Bishop Cauchy representation comparison carrier)**.** *A* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, B, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *row whose decoder route supplies the representation-facing observations.*
2. *B is the* $\mathsf{BishopRealPrecision}^{\uparrow}$ *row recording the requested constructive precision discipline for the comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row shared by the representation and precision reads.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed stream windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the precision and regular-readback rows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, B, S, Q, D have been displayed.*
7. *H records componentwise* hsame *transport for the representation row, precision row, stream windows, regular readback, dyadic ledger, terminal seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a comparison request through $R, B, S, Q, D, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{BishopRealPrecision}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *row.*
The carrier predicate asserts exactly that these rows form one finite comparison packet. It has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopCauchyRepresentationComparisonUp.BishopCauchyRepresentationComparisonTasteGa`
`single carrier alignment`

Theorem 5717.2 (Bishop Cauchy representation comparison NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ *carrier* $K = (R, B, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, representation decoding through* R*, constructive precision exposure through* B*, shared finite-window access through* S*, regular rational readback through* Q*, dyadic tolerance accounting through* D*, terminal real-name sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The visible analytic data are the representation decoder R, precision row B, shared stream window S, regular rational readback Q, dyadic tolerance ledger D, and terminal real seal E.

The comparison obligations are componentwise. A representation-facing consumer first reads R, then the requested Bishop precision surface B, and then the common finite-window, regular-readback, and dyadic-tolerance route S, Q, D.

Only after these rows are displayed may the consumer inspect the terminal seal E. The support rows H, C, P, N transport, replay, source, and name the same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5717.3 (Bishop Cauchy representation comparison nonescape)**.** *No accepted* BishopCauchyRepresentatio*n carrier exports quotient streams, selected host limits, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every downstream Real or Completion consumer receives only the displayed representation row, Bishop precision row, stream-window row, regular readback row, dyadic tolerance ledger, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

Proof. Inspect the carrier inventory. The tuple contains only $R, B, S, Q, D, E, H, C, P, N$.

The rows R, B, S, Q, D are the complete comparison route before the terminal Real seal can be consumed. Thus the seal E is not a selected host limit; it is the downstream endpoint of the displayed representation, precision, window, readback, and tolerance data.

Transport, replay, provenance, and local naming preserve the same boundary. Since no coordinate stores quotient stream equality, a choice schedule, a trichotomy branch, host real equality, or an ambient completeness principle, none can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}$ packet is generated from a CauchyRepresentation decoder, a BishopRealPrecision row, a StreamName finite window, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, seed)

Scope closed: Definition 5717.1, Definition 5717.2, and Definition 5717.3 seed the finite representation-comparison route from CauchyRepresentation and BishopRealPrecision data to StreamName, RegSeqRat, DyadicRatCore, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCauchyRepresentationComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, representation-decoder exactness, precision-row exposure, shared-window readback, dyadic tolerance accounting, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRepresentationComparisonUp.

The SeparableSpace$^{\uparrow}$ hub routes its countable dense schedule through chapter 1809 for Real-facing L10 density consumers.

Chapter 5718

A Concrete Naming Certificate for $\mathsf{SeparableSpace}^{\uparrow}$

Following Bourbaki's countable-base presentation and Bishop and Bridges' constructive dense-sequence discipline, the $\mathsf{SeparableSpace}^{\uparrow}$ packet names a countable dense schedule as finite BEDC data. It sits over the metric carrier of chapter 57, the finite window surface of chapter 126, the regular rational readback route of chapter 301, the terminal real-name surface of chapter 38, and the complete-separable consumer of chapter 1490. Its Real-facing dense schedule is also consumed by the downstream sibling dependency chapter 1809. The packet exports a dense schedule visible to L10 completion consumers without host set equality, selected dense points, countable choice, or ambient completeness.

Definition 5718.1 (Separable-space carrier)**.** *A* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, S, R, D, W, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *row containing the admitted metric carrier, distance classifier, and metric local* $\mathsf{NameCert}$*.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row that schedules finite dense-name probes and radius requests.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that reads the rational windows attached to those probes.*
4. *D is the countable-density ledger, recording that every displayed metric-radius request reaches an admitted stream probe with a regular rational distance bound.*
5. *W is the observation ledger joining stream positions, regular rational windows, metric-radius requests, and terminal* $\mathsf{Real}^{\uparrow}$ *seal reads.*
6. *H records componentwise* hsame *transport for M, S, R, D, W and the local certificate rows.*
7. *C records displayed* Cont *replay from a density request to the corresponding stream and regular-rational readback rows.*
8. *P is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for a host dense subset, selected nearest point, quotient point equality, or external enumeration.

Theorem 5718.2 (Separable-space NameCert obligations)**.** *For every accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier* $Q = (M, S, R, D, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SeparableSpace}^{\uparrow}}$ *surface consists exactly of six obligation rows: carrier admission for* $M, S, R, D, W, H, C, P, N$*; classifier transport to a common* $\mathsf{Met}^{\uparrow}$ *row before density is read; stability of the stream schedule and regular rational readback under* H*; exactness of* W *as the only route from a radius request to finite stream and rational-window data; displayed* Cont *replay from density requests through* S, R, D, W*; and* Pkg *provenance with local naming for precisely this packet. No additional obligation stores a selected dense point, countable-choice function, host set equality, or ambient completion datum.*

Proof. Project the carrier of Definition 5718.1. Carrier admission exposes the metric row, stream schedule, regular rational readback, density ledger, observation ledger, transport, replay, provenance, and local name.

The classifier route begins at M. Componentwise transport in H moves M, S, R, D, W together, so a dense schedule cannot be compared after changing the metric source.

The exactness row is W. Every radius request reaches a finite stream position in S, reads its rational window through R, and records the density witness in D before any terminal $\mathsf{Real}^{\uparrow}$ seal is consumed.

Finally replay the request through C and source it through P, N. Since no row names a selected point, host dense subset, countable-choice selector, or ambient completion, the six listed rows exhaust the NameCert obligations. □

Theorem 5718.3 (Separable-space StreamName dense readback)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Every dense-readback consumer reads the countable schedule only through* $\mathsf{StreamName}^{\uparrow}$ *row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, and density ledger* D *over the common* $\mathsf{Met}^{\uparrow}$ *row* M*. When the read is Real-facing it may consume only the terminal* $\mathsf{Real}^{\uparrow}$ *seal already displayed by* R, W*; when it is completion-facing, a* $\mathsf{PolishSpace}^{\uparrow}$ *or* $\mathsf{SeparableMetric}^{\uparrow}$ *consumer receives the same finite schedule route and no selected dense point, quotient point equality, host enumeration, or ambient completeness principle.*

Proof. Expose the accepted carrier. The only scheduling coordinate is S, and the only regular rational readback coordinate attached to the schedule is R.

Use Definition 5718.2. Observation exactness forces a dense-readback request through W before the density ledger D is available, while H keeps the route over the same metric row M.

The Real-facing boundary is downstream of the regular rational observations. Thus any $\mathsf{Real}^{\uparrow}$ seal is read after S, R, W, D, not as a separate host-real shortcut.

Finally pass the displayed route to the complete-separable consumers. The $\mathsf{PolishSpace}^{\uparrow}$ and $\mathsf{SeparableMetric}^{\uparrow}$ packets may inspect the same finite schedule rows, but the carrier contains no coordinate for selected dense points, quotient equality, host enumeration, or ambient completeness. □

Theorem 5718.4 (Separable-space finite dense-schedule induction)**.** *Let* $Q = (M, S, R, D, W, H, C, P, N)$ *be an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *carrier. Any finite list of dense-readback requests over* Q *is discharged by induction through the same* $\mathsf{StreamName}^{\uparrow}$ *schedule row* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, observation ledger* W*, density ledger* D*, and common* $\mathsf{Met}^{\uparrow}$ *row* M*. The induction exports only the displayed finite schedule route and no selected dense point, host enumeration, quotient equality, or countable-choice selector.*

Proof. Induct on the displayed finite list of dense-readback requests.

For the empty list, no dense point is selected and no host enumeration is opened. The carrier projection of Definition 5718.1 leaves only the common rows $M, S, R, D, W, H, C, P, N$, so the empty schedule is discharged by the displayed packet itself.

For a cons list, apply Definition 5718.3 to the head request. The head is routed through S, R, W, D over the fixed metric row M, and any Real-facing read remains behind the already displayed regular rational observations.

Apply the induction hypothesis to the tail list using the same accepted carrier. Componentwise transport through H and replay through C keep every tail request on the same S, R, W, D, M route. Since no step introduces a selected dense point, host enumeration, quotient equality, or countable-choice selector, the whole finite list is discharged inside the finite dense-schedule packet. □

Theorem 5718.5 (Separable-space RealSeparability forward route)**.** *Any* $\mathsf{RealSeparability}^{\uparrow}$ *consumer reading a* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule must factor the read through the* $\mathsf{Met}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal displayed by* R, W*, and the countable-density ledger* D *before the* $\mathsf{RealSeparability}^{\uparrow}$ *row of chapter 1809 is consumed. The route exports a finite dense-readback dependency, not a selected dense point, quotient point equality, host enumeration, countable-choice selector, or ambient completeness principle.*

Proof. Start with the accepted carrier $Q = (M, S, R, D, W, H, C, P, N)$. The metric source is M, the finite schedule is S, and the regular rational readback is R.

Use Definition 5718.3. It forces every dense read through S, R, W, D over the same metric row M before any Real-facing seal is consumed.

Now route the downstream sibling dependency through chapter 1809. The $\mathsf{RealSeparability}^{\uparrow}$ row receives the displayed countable-density ledger only after the metric, stream, regular rational, and $\mathsf{Real}^{\uparrow}$ rows are visible.

Because the carrier has no coordinate for selected dense points, quotient point equality, host enumeration, countable choice, or ambient completeness, the forward route remains a finite dependency edge. □

Theorem 5718.6 (SeparableSpace real-metric dependency route)**.** *Every completion, compactness, or real-facing consumer of an accepted* $\mathsf{SeparableSpace}^{\uparrow}$ *dense schedule reads the metric carrier data of chapter 57 and the Real-density route of chapter 1809 before it may inspect the downstream separability surface. The route exposes countable dense data only through displayed* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RealSeparability}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows; it exports no host enumeration, selected dense point, quotient point equality, countable-choice selector, or ambient completeness principle.*

Proof. Project the accepted dense schedule. Its metric source is the row that supplies distance observations, while the real-separability row supplies the Real-facing density readback.

The existing forward route through Definition 5718.5 places the Real-density consumer after the metric, stream, regular rational, terminal Real, and countable-density ledgers. Thus the child dependency route is not a separate carrier; it records the horizontal lattice edge through which completion and compactness consumers must enter the same finite packet.

Since the carrier has no coordinate for a host enumeration, selected dense point, quotient equality, countable-choice selector, or ambient completeness theorem, every such consumer remains inside the displayed metric and Real-separability dependency route. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ route points its finite readback through chapter 301 and its terminal seal through chapter 38.

Chapter 5719

A Concrete Naming Certificate for $\mathsf{DedekindCutStream}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DedekindCutStream}^{\uparrow}$ packet records a located cut-observation stream over the L10 stream-name substrate, not the forbidden Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$. It sits after the dyadic endpoint ledger of chapter 291, the finite observation interface of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the rounded located cut interface of chapter 436. The packet names the finite stream route by which lower and upper cut observations can be sampled before any Real-facing consumer treats them as regular Cauchy data.

Definition 5719.1 (Dedekind-cut stream carrier)**.** *A* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, recording the finite dyadic rational endpoints, radii, and comparison budgets used by the cut observation.*
2. *L is the located lower-cut observation row, carrying finite evidence that the displayed endpoints lie on the lower side of the cut.*
3. *U is the located upper-cut observation row, carrying finite evidence that the displayed endpoints lie on the upper side of the cut.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite observation windows through which L and U are sampled.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after the stream windows and dyadic endpoint budgets have been displayed.*
6. *H records componentwise* hsame *transport for dyadic endpoints, lower observations, upper observations, stream windows, regular readbacks, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located cut-observation request through D, L, U, W, R.*
8. *P is the* Pkg *provenance over* $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, L, U, W, R, H, C, P, N$. It has no coordinate for quotient cuts, quotient stream equality, trichotomy, selected suprema, selected real limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5719.2 (Dedekind-cut stream NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier* $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dyadic endpoint ledger, located lower observations, located upper observations, stream windows, regular rational readback, transport, replay, provenance, and local naming rows;*
2. *endpoint exactness, requiring every lower or upper cut observation to name its rational endpoint and comparison budget through D;*
3. *located observation separation, requiring L and U to remain displayed lower and upper observations rather than a quotient cut object;*
4. *stream-window exactness, requiring every cut observation consumed downstream to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row W;*
5. *regular-readback boundary, requiring R to be read only after D, L, U, W have been exposed;*
6. *componentwise transport and replay, requiring H and C to preserve the same endpoint, observation, stream, and readback route;*
7. *provenance and local naming, requiring P and N to source the packet through* $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *without adding quotient cuts, trichotomy, completeness, or host real equality.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5719.1. The accepted packet exposes only the dyadic endpoint row D, the lower and upper observation rows L, U, the stream window row W, the regular readback row R, and the structural rows H, C, P, N.

Endpoint exactness and located observation separation are read from the same projection. The rows L and U can mention a rational observation only through the dyadic endpoint and budget data in D, and they remain two displayed observation ledgers rather than a quotient cut.

The stream and readback obligations are downstream. A consumer first opens the finite windows in W, then obtains the regular rational readback R justified by those windows and the endpoint budget.

Finally, H and C transport and replay that same finite route, while P and N source and name it. Since the carrier contains no row for quotient cuts, trichotomy, completeness, host real equality, or selected real limits, the listed rows exhaust the local NameCert surface. □

Theorem 5719.3 (Dedekind-cut stream regular-Cauchy handoff)**.** *Every downstream* $\mathsf{Real}^{\uparrow}$ *read of an accepted* $\mathsf{DedekindCutStream}^{\uparrow}$ *carrier must pass through the* $\mathsf{StreamName}^{\uparrow}$ *window row W and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R before the cut-observation stream can be consumed. Equivalently, the only Real-facing route exported by the packet is*

$$\mathsf{DedekindCut}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}.$$

The handoff supplies located cut observations as finite stream-name data; it supplies no Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$*, quotient cut, trichotomy branch, completeness theorem, selected supremum, selected limit, or host real equality.*

This handoff is a source route for chapter 4372 when a downstream Real-facing consumer compares two regular-Cauchy presentations through the shared $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows, because the comparison sees displayed stream windows and dyadic tolerance ledgers rather than a quotient cut or host real equality.

Proof. Let a downstream Real-facing consumer read an accepted carrier $S_{\mathrm{cut}} = (D, L, U, W, R, H, C, P, N)$. By carrier projection, the only cut-facing data available before readback are the endpoint ledger D and the displayed lower and upper observation rows L, U.

The consumer cannot reach a regular rational datum directly from L or U. The finite stream-name row W records the observation windows that are opened, and the readback row R records the regular rational data obtained from those windows and the dyadic endpoint budget.

Thus any Real-facing use of the cut-observation stream is downstream of W and R. The remaining rows H, C, P, N preserve, replay, source, and name the same route.

Because no carrier coordinate stores quotient cuts, a Dedekind construction of reals, trichotomy, completeness, selected suprema, selected limits, or host real equality, the handoff is exactly the displayed stream-name and regular-Cauchy boundary. □

Falsifiable prediction: Every accepted $\mathsf{DedekindCutStream}^{\uparrow}$ Real-facing read factors through DyadicRatCore endpoints, located lower and upper cut observations, StreamName windows, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dedekindcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{DedekindCutStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located lower and upper cut observations, dyadic endpoint budgets, stream windows, regular rational readback, transport, replay, provenance, and local naming in one stream-observation packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$.

Constructive story: A $\mathsf{DedekindCutStream}^{\uparrow}$ packet is generated from DyadicRatCore endpoint rows, located lower and upper cut observations, StreamName finite windows, RegSeqRat readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5719.1, Definition 5719.2, and Definition 5719.3 seed the located cut-observation stream route over DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, DedekindCutUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Dedekind-cut construction of $\mathsf{Real}^{\uparrow}$, quotient cuts, trichotomy, completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, located observation separation, stream-window exactness, regular-readback boundary, transport, replay, provenance, and local naming for BEDC.Derived.DedekindCutStreamUp.

Theorem 5719.4 (Dedekind-cut stream RegSeqRat forward route)**.** *Every* $\mathsf{DedekindCutStream}^{\uparrow}$ *Real-facing consumer that exports a comparison must first display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{Real}^{\uparrow}$ *seal row of the accepted packet. The forward route is*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

inside the same BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{DedekindCutStream}^{\uparrow}}$ *surface. It exports no quotient cut, quotient stream equality, trichotomy branch, selected supremum, selected limit, ambient real completeness, or host comparison outside the displayed route.*

Proof. Start with the carrier of Definition 5719.1. A Real-facing comparison can read endpoint budgets only through the $\mathsf{DyadicRatCore}^{\uparrow}$ row D, and it can inspect cut observations only through the finite $\mathsf{StreamName}^{\uparrow}$ window row W.

Apply the regular-Cauchy handoff of Definition 5719.3. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R downstream of the displayed stream windows and dyadic endpoint budgets before any Real-facing read consumes the cut-observation stream.

Finally pass through the terminal $\mathsf{Real}^{\uparrow}$ seal while preserving the transport, replay, provenance, and local naming rows H, C, P, N. Since the packet has no coordinate for quotient cuts, selected suprema, selected limits, trichotomy, ambient completeness, or host comparison, the exported comparison route is exactly the displayed forward dependency chain. □

The $\mathsf{DedekindCutStream}^{\uparrow}$ dependency lattice consumes the finite-window $\mathsf{StreamName}^{\uparrow}$ spine of chapter 126 and the $\mathsf{RegSeqRat}^{\uparrow}$ readback route of chapter 301 before any $\mathsf{Real}^{\uparrow}$-facing cut stream is inspected through chapter 38. Thus the cut-observation packet remains a horizontal L10 route through displayed stream windows and regular rational readback, not a kernel-only pointer to a selected real limit or quotient cut.

Chapter 5720

A Concrete Naming Certificate for $\mathsf{StokesTheorem}^{\uparrow}$

Following Cartan and de Rham, the $\mathsf{StokesTheorem}^{\uparrow}$ packet names the finite BEDC surface on which exterior differentiation, oriented boundary, and integration over a carried chain can be consumed as one certificate. It reads differential forms from chapter 136, the cochain boundary route from chapter 137, the charted domain surface from chapter 96, and the scalar integral handoff from chapter 1526. The packet is only a naming certificate for the displayed Stokes boundary route; it does not supply a host smooth manifold, a selected triangulation, a quotient de Rham class, a measure completion, or an ambient real-completeness theorem.

Definition 5720.1 (Stokes theorem carrier)**.** *A* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, F, B, E, I, H, C, P, N)$$

whose rows have the following BEDC-facing meaning.

- *M is the carried* $\mathsf{Manifold}^{\uparrow}$ *chart-and-orientation source row for the integration domain.*
- *F is the carried* $\mathsf{DiffForm}^{\uparrow}$ *form row together with the exterior-derivative row consumed by the Stokes boundary read.*
- *B is the oriented boundary-chain row whose source and target faces are read through the same displayed manifold surface.*
- *E is the* $\mathsf{DeRham}^{\uparrow}$ *boundary route identifying the exterior derivative and boundary pairing as one chain-complex comparison.*
- *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *scalar-integration handoff for the displayed domain, boundary, form, and derivative rows.*
- *H records componentwise* hsame *transport for M, F, B, E, I and the structural rows.*
- *C records displayed* Cont *replay of the route from manifold charts through form rows, boundary-chain rows, de Rham boundary data, and integration handoff.*
- *P is the* Pkg *provenance over* $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
- *N is the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.StokesTheoremUp.StokesTheoremTasteGate single carrier alignment`

Theorem 5720.2 (Stokes theorem NameCert obligations)**.** *For every accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier* $K = (M, F, B, E, I, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$ *surface is exhausted by carrier admission for* M, F, B, E, I*, boundary-chain compatibility between* B *and* E*, integral-handoff compatibility between* E *and* I*, componentwise* hsame *transport in* H*, displayed* Cont *replay in* C*, provenance in* P*, and local naming in* N*. The certificate exports no host smooth map, no selected chain representative, no quotient de Rham class, no measure-completion object, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 5720.1. The analytic rows are exactly the manifold row M, differential-form row F, boundary-chain row B, de Rham boundary row E, and integration handoff row I.

The boundary compatibility is local: B supplies the displayed oriented boundary route, while E supplies the de Rham chain-complex comparison over the same form data. The integration compatibility is also local: I is read only after those rows have been displayed.

Finally attach H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the excluded host objects, the listed rows exhaust the local NameCert obligations. □

Theorem 5720.3 (Stokes boundary-chain route)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier. Any consumer of the Stokes boundary comparison must first read the manifold row* M*, the differential-form row* F*, and the oriented boundary-chain row* B *before it can consume the de Rham boundary row* E *or the integration handoff* I*. The comparison is therefore a finite BEDC route through* $M, F, B, E, I, H, C, P, N$*, not an appeal to a host boundary operator or an unlogged smooth-chain equality.*

Proof. Read the carrier inventory. The row B is the only boundary-chain coordinate, and it is stated over the manifold and form rows M, F.

Apply Definition 5720.2. Boundary-chain compatibility places B before the de Rham row E, and integral-handoff compatibility places I downstream from the same displayed route.

The structural rows H, C, P, N preserve and name this route without adding a second boundary source. Hence no consumer can bypass M, F, B when reading the Stokes comparison. □

Theorem 5720.4 (Stokes closed-form nonescape)**.** *Let* $K = (M, F, B, E, I, H, C, P, N)$ *be an accepted* $\mathsf{StokesTheorem}^{\uparrow}$ *carrier whose displayed* $\mathsf{DiffForm}^{\uparrow}$ *row* F *is read as closed along the de Rham route* E*. Then every zero-boundary integral read exported by the packet remains inside* $M, F, B, E, I, H, C, P, N$*. It may use the carried closed-form row, the boundary-chain route, the de Rham boundary comparison, and the Riemann-integral handoff, but it cannot introduce a quotient cohomology representative, a host Stokes theorem, a measure-completion argument, selected charts outside* M*, or ambient real completeness.*

Proof. Begin with the closed-form read inside F and E. The form row supplies the exterior data, and the de Rham row supplies the boundary comparison over the same displayed form.

The zero-boundary integral is then read through B and I. The boundary row supplies the oriented chain surface, and the integral row supplies only the scalar handoff already tied to M, F, B, E.

Transport and replay are confined to H and C, while P and N attach provenance and local naming. None of these rows contains a quotient representative, host theorem, measure-completion carrier, external chart selector, or ambient completeness coordinate, so the closed-form read cannot escape the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{StokesTheorem}^{\uparrow}$ packet records a displayed manifold-and-orientation row, carried differential-form row, oriented boundary-chain row, de Rham boundary route, Riemann-integral handoff, componentwise transport, continuation replay, provenance, and local naming. The packet names the Stokes boundary comparison only through these finite rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: The seed scope covers the displayed $\mathsf{Manifold}^{\uparrow}$, $\mathsf{DiffForm}^{\uparrow}$, $\mathsf{DeRham}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows of Definition 5720.1.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StokesTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close a public Stokes bridge, a host smooth-chain theorem, quotient cohomology, measure completion, selected triangulations, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-chain compatibility, integral-handoff compatibility, transport, replay, provenance, and ledger non-escape for BEDC.Derived.StokesTheoremUp.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ hub routes completion existence through the diagonal-window handoff of chapter 4080 and the pointwise stream metric boundary of chapter 4139 before its child files expose the completion seal.

Chapter 5721

A Concrete Naming Certificate for $\mathsf{MetricCompletionExistence}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet records the finite BEDC surface by which a displayed metric source receives a completion-facing existence certificate without selecting quotient classes or host representatives. It sits between the metric carrier of chapter 4230, the Cauchy route of chapter 3413, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real seal of chapter 38, the completion object surface of chapter 1506, and the complete-metric consumer row of chapter 128. The selected finite windows used by the completion-existence seal are compatible with the diagonal-selection route of chapter 4080 before any completion consumer treats the seal as public. The packet names existence as finite certificate data; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5721.1 (Metric completion existence carrier)**.** *A* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (M, S, D, W, R, Y, U, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row whose metric observations are to be completed.*
2. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *schedule row naming the finite Cauchy requests admitted from M.*
3. *D is the dense-embedding row that presents source points inside the completion-facing surface without choosing representatives.*
4. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the inspected windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the readback by finite dyadic budgets.*
7. *U is the completion seal row, tying the preceding route to* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumers while preserving the displayed* $\mathsf{Real}^{\uparrow}$ *distance boundary.*
8. *H records componentwise* hsame *transport for the source metric, Cauchy schedule, dense embedding, windows, readback, dyadic ledger, completion seal, uniqueness ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, S, D \longmapsto W \longmapsto R \longmapsto Y \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionExistence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows, their uniqueness ledger inside H, and the replay path C. It has no coordinate for quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5721.2 (Metric completion existence NameCert obligations)**.** *For every accepted* MetricCompletionExiste *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed source metric, Cauchy schedule, dense embedding, selected finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, uniqueness ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate refuses quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. Its rows expose the source $\mathsf{Met}^{\uparrow}$ data, the $\mathsf{Cauchy}^{\uparrow}$ schedule, the dense embedding, selected $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, completion seal, transport, replay, provenance, and local name.

The NameCert obligations follow the displayed order. The schedule is read from M, the embedding row D is inspected before any window, W and R provide finite observations and regular rational readback, and Y bounds them by dyadic tolerances before the completion seal U is available.

Transport and replay supply the uniqueness ledger. The row H moves the same coordinates componentwise, and C repeats the same inspection route for each consumer request; neither row introduces a host equality test or a chosen quotient representative.

Finally, P and N source and name the packet. Since the carrier contains no quotient-completion coordinate, selected representative, choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 5721.3 (Metric completion existence completion-seal nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$ reaches the completion seal U only after reading M, S, D, W, R, Y through the displayed* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, hsame, Cont, Pkg, *and local* NameCert *route. The seal cannot be replaced by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5721.1. The only route to U first exposes the source metric row M, the Cauchy schedule S, the dense embedding row D, selected finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligations theorem, Definition 5721.2. It orders those rows before the completion seal and binds the same route through componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

The refused objects are absent from the carrier. Quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no coordinate in E, so no completion-facing consumer can substitute them for the displayed seal row U. □

Theorem 5721.4 (Metric completion existence CompleteMetric consumer route)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier* $E = (M, S, D, W, R, Y, U, H, C, P, N)$ *may use the completion seal* U *only after the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule, dense embedding row,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary,* $\mathsf{MetricCompletion}^{\uparrow}$ *surface, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows have been displayed. The route exports no quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier $E = (M, S, D, W, R, Y, U, H, C, P, N)$. The rows M, S, D, W, R, Y display the source metric, Cauchy schedule, dense embedding, finite windows, regular rational readback, and dyadic tolerance ledger before the completion seal row can be read.

Invoke Definition 5721.2. It binds the same rows to the $\mathsf{Real}^{\uparrow}$ boundary, the $\mathsf{MetricCompletion}^{\uparrow}$ surface, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ row.

Invoke Definition 5721.3. It prevents any completion-facing consumer from replacing U by quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Finally read the consumer side through chapter 128. A $\mathsf{CompleteMetric}^{\uparrow}$ consumer receives only the displayed completion seal and the rows that precede it, so its use of U factors through $M, S, D, W, R, Y, U, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionExistence}^{\uparrow}$ packet is generated from a MetricSpace source row, a Cauchy schedule, dense embedding data, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a completion seal, uniqueness transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, seed)

Scope closed: Definition 5721.1, Definition 5721.2, Definition 5721.3, and Definition 5721.4 seed the finite completion-existence packet over MetricSpace, Cauchy, StreamName, RegSeqRat, DyadicRatCore, Real, MetricCompletion, CompleteMetric, hsame, Cont, provenance, and local NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-embedding order, finite-window readback, dyadic tolerance nonescape, completion-seal routing, CompleteMetric consumer routing, uniqueness transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionExistenceUp.

Every $\mathsf{MetricCompletionExistence}^{\uparrow}$ existence seal is read after the separated uniqueness boundary of chapter 1796, the universal-mapping row of chapter 1456, and the reflective completion route of chapter 1793. The handoff is finite completion-existence data: a source metric row, completion-facing route rows, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ naming. It is not quotient completion, selected representative equality, host limit choice, or an ambient completeness construction.

Theorem 5721.5 (Metric completion existence sibling handoff)**.** *For every accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet* $E = (M, S, D, W, R, Y, U, H, C, P, N)$*, a completion-existence consumer may use* U *only after the uniqueness route of chapter 1796, the universal-mapping route of chapter 1456, and the*

reflective-subcategory route of chapter 1793 have been displayed through the same finite packet. The consumer receives the completion seal, uniqueness transport, universal-mapping visibility, reflective visibility, replay, provenance, and local naming rows; it receives no quotient completion, choice-selected limit, or host completeness theorem.

Proof. Project the carrier of Definition 5721.1. The packet exposes $M, S, D, W, R, Y, U, H, C, P, N$ as the only completion-existence surface.

Use the local obligations of Definition 5721.2 to order every read of U through the source metric, Cauchy schedule, dense embedding, finite stream windows, rational readback, dyadic tolerance ledger, uniqueness ledger, transport, replay, provenance, and local naming rows.

The three sibling chapters supply the horizontal route: metric-completion uniqueness contributes the separated comparison boundary, Cauchy-completion adjunction contributes the universal-mapping row, and Cauchy-completion reflective subcategory contributes the reflective visibility boundary. The adjunction row forces completion-facing consumers and source finite-window consumers through the same dense embedding and uniqueness ledger before reflection is read.

Since all three are consumed as displayed rows inside the same finite handoff, no quotient completion or choice-selected limit construction can be substituted for the packet. □

Theorem 5721.6 (Metric completion existence Cauchy-filter basis route)**.** *Every accepted* MetricCompletionExisten *completion seal is read only after the Cauchy-filter basis row and the separated metric reflection boundary have been displayed with the same finite completion packet. In particular, a public completion seal must expose* $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{SeparatedMetricReflection}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CompletionEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any completion consumer treats the seal as public. The route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5721.1. Its finite route already displays the source metric, Cauchy schedule, dense embedding, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real-facing completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter basis boundary before the completion seal is consumed. The basis row refines the Cauchy schedule into finite filter requests, while the separated metric reflection row fixes the equality boundary through which the completion consumer may read the result.

The remaining rows are then forced in order: $\mathsf{MetricCompletion}^{\uparrow}$ supplies the completion target, $\mathsf{CompletionEmbedding}^{\uparrow}$ supplies the dense embedding boundary, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows supply the finite readback and tolerance seal. Since none of those rows contains a quotient object, selected representative, host limit selector, choice schedule, or ambient completeness principle, the public completion seal has exactly the stated dependency route. □

Theorem 5721.7 (Metric completion existence finite basis-read route)**.** *Every completion-facing read of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier first normalizes the Cauchy schedule into a finite* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis before exposing the public* $\mathsf{MetricCompletion}^{\uparrow}$ *seal. Only after that basis row is displayed may the consumer read the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *row attached to the same completion packet.*

The basis-read route exports no quotient completion, selected representative, host limit choice, countable-choice schedule, excluded ambient completeness row, or separated metric shortcut outside the displayed basis packet.

Proof. Project the carrier of Definition 5721.1. Its completion-facing coordinates already include the Cauchy schedule, finite readback rows, completion seal, embedding boundary, transport, replay, provenance, and local naming.

Apply Definition 5721.6. That route inserts the $\mathsf{CauchyFilterBasis}^{\uparrow}$ boundary before the completion seal is public, so the Cauchy schedule is read as finite basis data rather than as an ambient filter object.

Now expose the consumer rows in order. The $\mathsf{MetricCompletion}^{\uparrow}$ seal can be inspected only after the basis row, and the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are the finite observation, rational readback, tolerance, and terminal seal rows of that same packet.

The excluded rows have no coordinate in the carrier. There is no quotient completion row, selected representative row, host limit-choice row, countable-choice schedule row, or ambient completeness row from which the consumer could read additional data. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ completion-existence packet consumes the finite Cauchy-filter basis route of chapter 1501 and the completion-facing filter handoff of chapter 1569 before exporting its own seal to the complete-metric consumer surface of chapter 128. The dependency is horizontal: the basis rows choose the finite requests, the completion rows bind the public completion route, and the complete-metric rows receive only the displayed existence packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.

Theorem 5721.8 (Metric completion existence separated forward route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal row U is consumed by the separated-completion route of chapter 1612 only after the* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows are visible. The forward link exports a displayed completion-existence seal into* $\mathsf{SeparatedCompletion}^{\uparrow}$ *as finite route evidence, not as quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space.*

Proof. Project the accepted carrier of Definition 5721.1. Its completion-seal row is not isolated; it is read downstream of the metric-space source, Cauchy request, StreamName observation schedule, RegSeqRat readback, DyadicRatCore tolerance row, Real seal, MetricCompletion row, CompleteMetric row, transport, replay, provenance, and local naming rows.

Apply Definition 5721.2. The obligation surface forces the seal consumer to expose those prerequisite rows before the completion-existence claim can be replayed or named.

The separated-completion sibling then reads the displayed completion-existence seal as route evidence for chapter 1612. Since the carrier and obligation theorem have no coordinate for quotient completion equality, selected representatives, host point equality, countable choice, or an ambient complete metric space, the forward route exports none of those endpoints. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal is read only after the Cauchy-filter basis route of chapter 1501 and the separated-completion route of chapter 1612 have both been displayed for the same completion packet. Thus a completion-existence consumer first sees the finite Cauchy-basis rows, then the separated-completion boundary, and only then treats the $\mathsf{MetricCompletionExistence}^{\uparrow}$ row as a seal for a complete metric forward route.

Theorem 5721.9 (Metric completion existence uniqueness-reflective route)**.** *Every accepted* MetricCompletionExist *read factors first through the existence carrier of Definition 5721.1. Before any completion consumer receives the completion seal U, the read must display the separated comparison boundary of* $\mathsf{MetricCompletionUniqueness}^{\uparrow}$ *and the reflective visibility boundary of* $\mathsf{CauchyCompletionReflectiveSubcategory}^{\uparrow}$

over the same finite source, Cauchy schedule, dense embedding, window, readback, dyadic, transport, replay, provenance, and local naming rows. It exports no quotient completion, selected representative, host limit choice, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existence carrier. Its rows $M, S, D, W, R, Y, U, H, C, P, N$ are the whole route by which the source metric and Cauchy schedule reach a completion seal.

Use Definition 5721.2 to keep the source metric, schedule, dense embedding, finite windows, regular rational readback, dyadic tolerance, completion seal, transport, replay, provenance, and local naming in the displayed order.

Now read the sibling surfaces named in the packet provenance. The separated comparison boundary of chapter 1796 supplies the uniqueness side, and the reflective visibility boundary of chapter 1793 supplies the reflective side.

Because both sibling reads are consumed through the same finite carrier route, the completion consumer receives a uniqueness-reflective handoff rather than a quotient object, selected representative, host limit choice, countable-choice schedule, or ambient completeness theorem. □

Theorem 5721.10 (Metric completion existence to Cauchy completion monad law forward link)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion seal consumed by a completion-law reader must pass through the finite* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*, and* $\mathsf{CompleteMetric}^{\uparrow}$ *rows before the* $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ *law surface of chapter 3834 is read. The forward link exports only this displayed completion-existence route, together with componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionExistence}^{\uparrow}}$ *row.*

Proof. Project the carrier of Definition 5721.1. The source metric row M, Cauchy schedule S, selected stream windows W, regular rational readback R, dyadic tolerance ledger Y, and completion seal U form the ordered finite route toward $\mathsf{MetricCompletion}^{\uparrow}$ and $\mathsf{CompleteMetric}^{\uparrow}$ consumers.

Apply Definition 5721.2. It states that the local certificate is exhausted by carrier admission, the displayed completion-existence rows, transport, replay, provenance, and local naming. Hence a completion-law reader cannot bypass those rows when reaching the law surface.

Finally read the structural rows H, C, P, N. They preserve and replay the same finite route for the consumer, so the $\mathsf{CauchyCompletionMonadLaw}^{\uparrow}$ handoff is a forward sibling dependency through the completion-existence carrier rather than a quotient completion, selected representative, or ambient completeness principle. □

Theorem 5721.11 (Metric completion existence Real-consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *carrier must read the* $\mathsf{Met}^{\uparrow}$ *source row,* $\mathsf{Cauchy}^{\uparrow}$ *schedule row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *distance seal,* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row, and* $\mathsf{CompleteMetric}^{\uparrow}$ *consumer row before using the completion seal. The route is a finite L10 dependency route through the displayed completion-existence packet, not a quotient completion, selected representative, countable-choice schedule, host equality oracle, trichotomy row, open-cover compactness row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5721.1. Its only completion seal is the row U, and the route to U first displays the metric source M, Cauchy schedule S, dense embedding D, finite windows W, regular rational readback R, and dyadic tolerance ledger Y.

Apply the NameCert obligation theorem Definition 5721.2. It keeps $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ as separate source and consumer rows before the completion seal is exposed.

The sibling routes then keep the same boundary for downstream consumers. The complete-metric forward route and the completion uniqueness reflective route preserve the completion seal as packet data, while the Cauchy completion monad-law forward link reads the same finite source rows before any algebraic consumer uses the seal.

Since none of these rows contains quotient completion, selected representatives, countable choice, host equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every Real/completion consumer remains on the displayed L10 route. □

Theorem 5721.12 (MetricCompletionExistence pointwise-stream lattice route)**.** *Any* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *read consumed by* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *or by a regular Cauchy completion consumer first exposes the finite lattice*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the completion-existence boundary. Thus the read travels through displayed stream windows, regular rational readback, terminal real-name sealing, the metric carrier, and the pointwise stream metric row; it is not a quotient completion, a selected limit, an ambient completeness theorem, or a host equality of real streams.

Proof. Read the pointwise-stream metric consumer first. Its distance observations are stream-indexed, so the consumer must display the $\mathsf{StreamName}^{\uparrow}$ row and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ row before any completion-existence seal can be used.

For a regular Cauchy completion consumer, the same route exposes the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal on top of the ambient $\mathsf{Met}^{\uparrow}$ row. These rows are the finite completion-facing lattice visible to the consumer.

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ boundary is therefore reached only after the displayed lattice has been opened. Since none of those rows carries quotient completion, selected limits, ambient completeness, or host stream equality, the handoff cannot export those conclusions. □

Theorem 5721.13 (MetricCompletionExistence pointwise-stream factorization)**.** *Every completion-existence read consumed through* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *factors through the displayed order*

$$\mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{PointwiseStreamMetric}^{\uparrow}$$

before it reaches the $\mathsf{MetricCompletionExistence}^{\uparrow}$ *boundary. The factorization exposes finite stream windows, regular rational readback, the terminal Real seal, metric-space carrier rows, and the pointwise-stream distance row in that order; it exports no quotient completion, selected limit, ambient completeness theorem, or host equality of streams.*

Proof. Project the lattice route of Definition 5721.12. It already displays the only rows available to a pointwise-stream metric consumer before the completion-existence boundary.

The first factor is the finite stream read. A consumer sees $\mathsf{StreamName}^{\uparrow}$ windows and then the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to those windows before any Real or metric row can be used.

The second factor is the completion-facing metric read. The terminal $\mathsf{Real}^{\uparrow}$ seal is exposed before the ambient $\mathsf{Met}^{\uparrow}$ row, and the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ distance row is downstream of both.

Since the route contains only these displayed rows, it has no coordinate for quotient completions, selected limits, ambient completeness, or host stream equality. The completion-existence handoff therefore consumes the pointwise-stream factorization rather than a hidden completion principle. □

The $\mathsf{MetricCompletionExistence}^{\uparrow}$ seal consumes $\mathsf{RegularCauchyTailEstimate}^{\uparrow}$ before exposing completion-facing rows: finite tail thresholds enter through chapter 1487, then travel along the existing metric-completion sibling handoff of Definition 5721.5. Thus a completion consumer sees the tail estimate as displayed threshold and readback data attached to the same $\mathsf{Met}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, and $\mathsf{CompleteMetric}^{\uparrow}$ route, not as a quotient completion or ambient completeness principle.

Theorem 5721.14 (Metric completion existence Cauchy-convergence dependency route)**.** *Every* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *completion-seal consumer that uses Cauchy-convergence data first reads the Cauchy filter basis of chapter 1501, the regular Cauchy tail row, the separated-completion boundary of chapter 1612, and the terminal real-name readback row before the completion seal is public. Thus the consumer receives finite basis, tail, separated-completion, and* $\mathsf{Real}^{\uparrow}$ *rows attached to the same displayed completion packet, not quotient completion, selected representatives, host limit choice, or ambient completeness.*

Proof. Project the accepted $\mathsf{MetricCompletionExistence}^{\uparrow}$ carrier. Its existing route displays the source metric, Cauchy schedule, dense embedding, finite stream windows, regular rational readback, dyadic tolerance ledger, completion seal, transport, replay, provenance, and local naming rows.

Insert the Cauchy-filter and separated-completion dependencies before the seal is read. The basis row fixes the finite Cauchy requests, the regular tail row supplies the readback thresholds, and the separated-completion boundary fixes the equality route that downstream consumers may inspect.

The final readback remains terminal. The $\mathsf{Real}^{\uparrow}$ row is reached only through the same displayed packet, so the completion-seal consumer cannot replace the dependency route by quotient completion, selected representatives, host limit choice, or ambient completeness. □

Theorem 5721.15 (MetricCompletionExistence to CauchyRealSpread route)**.** *Every* $\mathsf{CauchyRealSpread}^{\uparrow}$ *consumer of an accepted* $\mathsf{MetricCompletionExistence}^{\uparrow}$ *packet receives the completion-facing seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *boundary, and* $\mathsf{MetricCompletion}^{\uparrow}$ *completion row of Definition 5721.1. The route supplies a finite completion-facing seal for chapter 4161; it does not add quotient completion, selected representatives, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, host equality, trichotomy, or a choice principle.*

Proof. Project the carrier of Definition 5721.1. The only completion-facing route is the displayed chain from the source metric and Cauchy schedule through selected stream windows, regular rational readback, dyadic tolerance data, and the completion seal.

Apply Definition 5721.2. It requires the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{MetricCompletion}^{\uparrow}$ rows to remain visible before the seal is consumed.

The $\mathsf{CauchyRealSpread}^{\uparrow}$ consumer reads exactly that seal as finite spread data. Since the carrier has no quotient-completion coordinate, selected representative, ambient completeness theorem, host equality oracle, trichotomy row, or choice schedule, the route cannot export any of the excluded objects. □

The $\mathsf{FiniteLimitSeal}^{\uparrow}$ route is consumed by the regular Cauchy gluing boundary of chapter 5899, so this hub records the finite seal as a sibling dependency before routing the child files.

Chapter 5722

A Concrete Naming Certificate for $\mathsf{FiniteLimitSeal}^{\uparrow}$

Following Bishop regular-Cauchy completion, the $\mathsf{FiniteLimitSeal}^{\uparrow}$ packet records the finite seal used by diagonal-limit arguments before any selected limit or quotient stream equality can be mentioned. It is anchored in the finite-window route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet packages the local Real-completion limit service as displayed BEDC data; it is not a global completeness theorem for $\mathsf{Real}^{\uparrow}$.

Its downstream role is fixed by the sibling routes Definition 5722.4 and Definition 5722.5: CategoryUp finite-limit consumers must first pass through the same M, W, R, D, E seal rows, and RealUp limit-seal consumers receive only that finite modulus-window-readback-tolerance-seal route. The Cauchy-filter forward route of Definition 5722.9 is consumed through the same finite seal rows, so completion-facing filter reads do not acquire a selected limit or quotient stream equality outside the local packet.

Definition 5722.1 (Finite limit seal carrier)**.** *A* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the modulus schedule row naming the finite tail and tolerance requests used by the diagonal-limit route.*
2. *W is the selected* $\mathsf{StreamName}^{\uparrow}$ *window row containing exactly the finite observations requested by M.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger against which the readback is checked.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the modulus schedule, selected windows, readback, tolerance ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{FiniteLimitSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by the displayed modulus schedule, window reads, rational readback, dyadic tolerance ledger, Real seal, transport, replay, provenance, and local name. It has no coordinate for selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5722.2 (Finite limit seal route nonescape)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier exports its diagonal-limit service only through the finite route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E.$$

It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier of Definition 5722.1. The only limit-facing row is the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the definition places it after the modulus schedule, selected stream windows, regular rational readback, and dyadic tolerance ledger.

Thus any consumer of the seal must first read M, W, R, D. The transport row H and replay row C preserve this same route under componentwise renaming and repeated consumer requests.

The provenance row P and local name N source the packet without adding mathematical content. Since the tuple contains no selected-limit coordinate, quotient equality relation, choice schedule, host real equality oracle, trichotomy branch, or ambient completeness theorem, the displayed route is the whole exported surface. □

Theorem 5722.3 (Finite limit seal bounded regular Cauchy route)**.** *When the modulus schedule M and the selected* $\mathsf{StreamName}^{\uparrow}$ *windows W of an accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier*

$$L = (M, W, R, D, E, H, C, P, N)$$

are supplied by an accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ *packet, the finite-limit seal route factors through the bounded tail envelope of chapter 1697 before reaching M, W, R, D, E. The handoff first reads the bounded regular Cauchy threshold and late-window rows, then repacks the displayed windows as the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the seal. It exports no selected limit, quotient stream equality, countable choice, host real equality, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ carrier and the supplying accepted $\mathsf{BoundedRegularCauchy}^{\uparrow}$ packet. The bounded regular Cauchy chapter provides the finite tail-envelope route before any Real-facing seal is consumed.

Read that envelope into the local coordinates M, W, R, D, E. The supplied threshold schedule becomes the modulus row M, the supplied late stream windows become W, and their regular rational and dyadic tolerance readbacks become the local rows R and D before the terminal seal E is exposed.

Finally apply Definition 5722.2. The structural rows H, C, P, N preserve the displayed route from M through E, and the local classifier has no coordinate for a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$.

Constructive story: A $\mathsf{FiniteLimitSeal}^\uparrow$ packet is generated from a modulus schedule, selected StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 5722.1, Definition 5722.2, and Definition 5722.3 seed the finite diagonal-limit seal over StreamName, RegSeqRat, DyadicRatCore, Real, hsame, Cont, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window selection, regular readback, dyadic tolerance nonescape, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLimitSealUp.

Theorem 5722.4 (Finite limit seal categorical consumer route)**.** *Every categorical or completion-facing consumer of an accepted* $\mathsf{FiniteLimitSeal}^\uparrow$ *carrier reads the seal through the finite-window route*

$$M \longmapsto W \longmapsto R \longmapsto D \longmapsto E$$

of Definition 5722.2, then through downstream completion siblings such as chapter 1276 and chapter 3701 before it can use the terminal $\mathsf{Real}^\uparrow$ *seal. The consumer receives a finite categorical handoff over the displayed StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and local NameCert rows; it receives no selected limits, quotient stream equality, countable choice, host real equality, trichotomy, or ambient Real completeness.*

Proof. Start with the carrier of Definition 5722.1. Its only completion-facing route places the modulus schedule, finite stream windows, regular readback, dyadic tolerance ledger, and terminal Real seal in the ordered rows M, W, R, D, E.

Apply Definition 5722.2. This prevents a categorical consumer from bypassing the finite-window rows or reading a selected limit directly from the seal.

Use Definition 5722.3 to align bounded regular Cauchy data with the same route. The bounded threshold and late-window rows are repacked as M, W, R, D before the Real seal E is visible.

Finally pass through the downstream envelope and classifier-lock chapters. chapter 1276 records the finite envelope consumed by Real and Completion routes, while chapter 3701 keeps the diagonal limit classifier on a single displayed seal route. Since none of these rows adds a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem, the categorical consumer remains on the finite-window lattice. □

Theorem 5722.5 (Finite limit seal real dependency route)**.** *Every public* $\mathsf{FiniteLimitSeal}^\uparrow$ *seal consumes its real-facing rows in the displayed dependency order*

$$\mathsf{Real}^\uparrow,\ \mathsf{RegSeqRat}^\uparrow,\ \mathsf{StreamName}^\uparrow,\ \mathsf{MetricCompletion}^\uparrow$$

only through the finite modulus, window, readback, tolerance, and completion-envelope rows of Definition 5722.1. A limit consumer must first read the $\mathsf{StreamName}^\uparrow$ *observation windows, then the* $\mathsf{RegSeqRat}^\uparrow$ *rational readback, then the* $\mathsf{Real}^\uparrow$ *seal and* $\mathsf{MetricCompletion}^\uparrow$ *envelope route before the local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^\uparrow}$ *row is public; it receives no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the accepted carrier from Definition 5722.1. The only real-facing path is the finite route from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal.

Apply Definition 5722.2 and Definition 5722.4. They force completion-facing consumers to remain behind the displayed M, W, R, D, E rows before any categorical or completion envelope can read the seal.

The added dependency order records the same lattice edge: $\mathsf{StreamName}^{\uparrow}$ supplies the finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the rational readback, $\mathsf{Real}^{\uparrow}$ supplies the terminal seal, and $\mathsf{MetricCompletion}^{\uparrow}$ consumes that seal only through the completion route. Since the local packet has no coordinate for a selected limit or ambient completeness theorem, the dependency route is exactly the finite readback route already named by the carrier. □

Theorem 5722.6 (FiniteLimitSeal MetricCompletionFinite forward link)**.** *Every finite-limit seal consumed by a completion-facing route must expose its* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows before entering* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. The forward link into chapter 1559 is a finite readback route, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Start with the real dependency route above. It already forces a public $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumer to read finite stream windows, regular rational readback, terminal Real sealing, and the completion envelope before any local naming row is public.

Next project the structural rows of Definition 5722.1. The same route carries hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data, so a completion-facing consumer cannot bypass the finite seal packet.

Finally compose this route with chapter 1559. A $\mathsf{MetricCompletionFinite}^{\uparrow}$ consumer therefore receives the finite-limit seal only after the StreamName, RegSeqRat, DyadicRatCore, Real, transport, replay, provenance, and naming rows are exposed, with no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness. □

Theorem 5722.7 (FiniteLimitSeal Real-completion route)**.** *Every accepted* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *completion-facing consumer factors through the route*

$$M, W, R, D, E, H, C, P, N.$$

Here W *is supplied by* $\mathsf{StreamName}^{\uparrow}$*,* R *by* $\mathsf{RegSeqRat}^{\uparrow}$*,* D *by* $\mathsf{DyadicRatCore}^{\uparrow}$*,* E *by* $\mathsf{Real}^{\uparrow}$*, and downstream completion is read only through* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{MetricCompletionFinite}^{\uparrow}$*. No selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness enters the route.*

Proof. Project the carrier of Definition 5722.1. The completion-facing rows available to a consumer are the displayed modulus, window, readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming rows.

Use route nonescape, Definition 5722.2. It prevents a completion consumer from bypassing M, W, R, D, E or replacing them by a selected limit, quotient stream equality, host equality, trichotomy branch, or ambient completeness theorem.

Apply the real dependency route above. It identifies W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the dyadic tolerance row, and E as the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally use the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link, Definition 5722.6. It records the finite completion handoff and leaves downstream completion only through $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{MetricCompletionFinite}^{\uparrow}$, so the displayed route is exhausted by $M, W, R, D, E, H, C, P, N$. □

Theorem 5722.8 (FiniteLimitSeal PointwiseStreamMetric reverse route)**.** *Any* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer entering* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *must expose* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{Met}^{\uparrow}$ *row,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *before the finite seal is read. The route from chapter 4139 is therefore the reverse completion-facing edge to the finite-limit seal routes above, not a selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient Real completeness theorem.*

Proof. Project the completion-facing route of Definition 5722.7. It requires the finite seal to expose $M, W, R, D, E, H, C, P, N$ before downstream completion data can be read.

Now read the $\mathsf{PointwiseStreamMetric}^{\uparrow}$ consumer through chapter 4139. Its stream metric input already carries a displayed $\mathsf{Met}^{\uparrow}$ row and pointwise StreamName observations, so entry into $\mathsf{FiniteLimitSeal}^{\uparrow}$ must align those observations with the seal rows W, R, D, E.

Finally apply the real dependency route and the $\mathsf{MetricCompletionFinite}^{\uparrow}$ forward link. They identify W as the $\mathsf{StreamName}^{\uparrow}$ window row, R as the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, D as the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and E as the $\mathsf{Real}^{\uparrow}$ seal, while H, C, P, N keep transport, replay, provenance, and naming visible. Thus the reverse route contributes only the displayed finite rows and no selected limit, quotient stream equality, choice schedule, host equality, trichotomy branch, or ambient Real completeness. □

Real-facing $\mathsf{FiniteLimitSeal}^{\uparrow}$ consumers of finite limits must read the $\mathsf{StreamName}^{\uparrow}$ observation windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38 before exporting finite-limit rows. This dependency route keeps the finite seal on displayed windows, regular rational readback, dyadic tolerance, and Real-name sealing; it does not supply selected limits, quotient stream equality, trichotomy, countable choice, or ambient Real completeness.

Theorem 5722.9 (FiniteLimitSeal CauchyFilterLimit forward route)**.** *Let*

$$L = (M, W, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteLimitSeal}^{\uparrow}$ *carrier. Every completion-facing read of* L *that is consumed as a* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *route first projects the displayed modulus schedule* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*, and only then enters the Cauchy-filter limit surface of chapter 1558. The handoff preserves the componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ *row* N*. It exports no selected limit, quotient stream equality, countable-choice schedule, host real equality, trichotomy branch, or ambient completeness theorem.*

Proof. Project the carrier of Definition 5722.1. Its only completion-facing route is the finite sequence M, W, R, D, E together with H, C, P, N.

Apply Definition 5722.2. This keeps the seal read behind the displayed StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal rather than behind a selected limit or quotient stream equality.

Now use the real dependency route of Definition 5722.5. It records the same L10 path from $\mathsf{StreamName}^{\uparrow}$ through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ to $\mathsf{Real}^{\uparrow}$, so a $\mathsf{CauchyFilterLimit}^{\uparrow}$ consumer can only receive the finite seal after those rows have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate is present in L, the route into chapter 1558 is a substantive dependency of the finite seal and not an external citation. □

Theorem 5722.10 (FiniteLimitSeal regular Cauchy gluing route)**.** *Every* $\mathsf{FiniteLimitSeal}^{\uparrow}$ *consumer that reads a glued regular-Cauchy limit must expose the finite-limit seal rows* $M, W, R, D, E, H, C, P, N$*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the sibling* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. The route exports no selected representatives, quotient stream equality, countable choice, or ambient Real completeness.*

Proof. Project the finite-limit seal carrier of Definition 5722.1. Its public limit-facing route is the displayed sequence from the modulus schedule through stream windows, regular rational readback, dyadic tolerance, and terminal Real seal, together with transport, replay, provenance, and local naming rows.

Apply the real-limit dependency route and the Cauchy-filter forward route, Definition 5722.5 and Definition 5722.9. These require every completion-facing seal read to pass through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local $\mathsf{NameCert}_{\mathsf{FiniteLimitSeal}^{\uparrow}}$ row is public.

For a glued regular-Cauchy limit, cite the substantive sibling dependency chapter 5899. The gluing packet supplies the compatible-window route, but the finite-limit seal still exposes its own $M, W, R, D, E, H, C, P, N$ rows before the terminal seal can be consumed.

The carrier and the cited sibling route contain no coordinate for selected representatives, quotient stream equality, countable choice, or ambient Real completeness. Thus a glued regular-Cauchy limit is read only through the finite seal route and the $\mathsf{RegularCauchyGluing}^{\uparrow}$ dependency. □

Chapter 5723

A Concrete Naming Certificate for $\mathsf{FormalBallDomain}^{\uparrow}$

Following Edalat and Heckmann on formal balls and domain-theoretic completion, the $\mathsf{FormalBallDomain}^{\uparrow}$ packet records the rounded, directed, radius-ordered domain surface used by constructive completion arguments. It sits over the formal-ball carrier of chapter 1510, the basis refinement route of chapter 3943, the radius-order ledger of chapter 3968, the completion route of chapter 2012, the metric completion boundary of chapter 1506, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a finite domain-neighbour route without quotienting Cauchy names, selecting limits, invoking host point equality, using countable choice, or assuming ambient completeness.

Definition 5723.1 (FormalBallDomain carrier)**.** *A* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (B, R, Q, I, C, E, H, T, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{FormalBall}^{\uparrow}$ *observation row, carrying the visible centre-radius windows used by the domain packet.*
2. *R is the* $\mathsf{FormalBallRadiusOrder}^{\uparrow}$ *row, recording the finite radius refinement and order comparisons available before any rounded-domain read is exported.*
3. *Q is the* $\mathsf{FormalBallBasis}^{\uparrow}$ *row, exposing the basis elements and basis-refinement route inspected by the domain surface.*
4. *I is the directed-ideal row, listing the finite compatible formal-ball windows retained by the rounded domain read.*
5. *C is the* $\mathsf{FormalBallCompletion}^{\uparrow}$ *row, reached only after the formal-ball, radius-order, basis, and directed-ideal rows have been displayed.*
6. *E is the completion endpoint row, read through* $\mathsf{MetricCompletion}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rather than through host point equality.*
7. *H records componentwise* hsame *transport for formal balls, radius orders, bases, directed ideals, completion rows, endpoint seals, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a rounded-domain request through $B, R, Q, I, C, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FormalBallDomain}^{\uparrow}$, $\mathsf{FormalBall}^{\uparrow}$, $\mathsf{FormalBallBasis}^{\uparrow}$, $\mathsf{FormalBallRadiusOrder}^{\uparrow}$, $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, R, Q, I, C, E, H, T, P, N$. It has no coordinate for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FormalBallDomainUp.FormalBallDomainUp`

Theorem 5723.2 (FormalBallDomain NameCert obligations)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier $D = (B, R, Q, I, C, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the formal-ball row, radius-order row, basis row, directed-ideal row, formal-ball-completion row, endpoint row, transport row, replay row, provenance row, and local naming row;*
2. *radius-order exactness, requiring every refinement or rounded containment used by the domain read to be recorded in R before any ideal or completion row is consumed;*
3. *basis coverage, requiring each retained formal-ball window to be inspected through Q and then recorded in the directed row I;*
4. *completion handoff, requiring C and E to remain downstream of B, R, Q, I and to expose only the displayed* $\mathsf{FormalBallCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route;*
5. *structural stability, requiring H and T to preserve the same component rows under* hsame *transport and displayed* Cont *replay;*
6. *provenance and local naming through P and N, with no obligation row for quotient completion, selected representatives, host point equality, countable choice, trichotomy, or ambient completeness.*

No $\mathsf{FormalBallDomain}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5723.1. The accepted packet exposes exactly the formal-ball, radius-order, basis, directed-ideal, completion, endpoint, transport, replay, provenance, and naming rows.

Radius-order exactness and basis coverage are forced by the row order. A domain read must first inspect B through the finite comparisons in R, then use Q to identify the displayed basis route before the retained directed row I is available.

The completion handoff is downstream of those rows. The formal-ball-completion row C and endpoint row E can be read only after B, R, Q, I have been displayed, so they cannot supply a quotient-completion representative or host point equality on their own.

Finally H and T transport and replay the same finite packet, while P and N source and name it. Since the carrier has no coordinate for quotient completion, selected representatives, countable choice, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 5723.3 (FormalBallDomain Cauchy completion handoff)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, *or* $\mathsf{FormalBallCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{FormalBall}^{\uparrow} \longmapsto \mathsf{FormalBallRadiusOrder}^{\uparrow} \longmapsto \mathsf{FormalBallBasis}^{\uparrow} \longmapsto I \longmapsto \mathsf{FormalBallCompletion}^{\uparrow} \longmapsto \mathsf{MetricComplet}$$

The handoff exports rounded formal-ball domain data, directed finite coverage, and completion-facing endpoint seals, but it exports no quotient completion, selected Cauchy representative, host point equality, countable-choice selector, trichotomy, or ambient completeness principle.

Proof. Start with an accepted carrier $D = (B, R, Q, I, C, E, H, T, P, N)$. A Cauchy-facing or completion-facing request can first read only the formal-ball row B and the displayed radius-order comparisons in R.

The basis and directed stages are then forced through Q and I. The basis row identifies the finite neighbourhood data retained by the packet, and the directed row records the compatible finite family used by the rounded-domain read.

The completion rows are terminal for this route. The row C supplies the formal-ball-completion handoff, while E records the metric-completion and Real-seal endpoint data that downstream consumers may inspect.

Componentwise transport and replay through H, T preserve the same route, and P, N record provenance and local naming. Since no excluded quotient, selected representative, host equality, choice, trichotomy, or ambient-completeness coordinate exists in the carrier, the handoff exposes only the displayed domain data. □

Theorem 5723.4 (FormalBallDomain Real seal boundary)**.** *For every accepted* $\mathsf{FormalBallDomain}^{\uparrow}$ *carrier* $D = (B, R, Q, I, C, E, H, T, P, N)$*, each* $\mathsf{Real}^{\uparrow}$*-facing read exported by* D *remains downstream of the formal-ball row* B*, radius-order row* R*, basis row* Q*, directed-ideal row* I*, and formal-ball-completion row* C*. The Real seal is therefore a terminal endpoint of the displayed completion route, not a detached real point, quotient-completion class, selected representative, host point equality, countable-choice selector, or ambient completeness witness.*

Proof. Read the endpoint row E from the carrier. By Definition 5723.1, E is exposed only after the formal-ball, radius-order, basis, directed-ideal, and formal-ball-completion rows have been displayed.

Apply the NameCert obligations of Definition 5723.2. Radius-order exactness keeps every refinement inside R, basis coverage keeps retained windows inside Q, I, and completion handoff keeps C, E downstream of that finite route.

The replay row T and transport row H preserve the same component order. The provenance row P and local naming row N name no second Real route, no representative selector, and no host point-equality witness.

Thus any $\mathsf{Real}^{\uparrow}$ read exported by the packet is terminal for $B, R, Q, I, C, E, H, T, P, N$. The excluded detached real point, quotient class, selected representative, choice schedule, and ambient completeness data have no carrier coordinate from which a consumer could read them. □

Falsifiable prediction: Every accepted $\mathsf{FormalBallDomain}^{\uparrow}$ completion-facing read factors through displayed FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, a directed finite row, FormalBallCompletionUp handoff, MetricCompletionUp endpoint data, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: formalball, formalballbasis, formalballradiusorder, formalballcompletion, metriccompletion, real: $\mathsf{FormalBallDomain}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries formal-ball windows, radius-order comparisons, basis coverage, directed finite domain data, completion handoff, endpoint sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FormalBallDomain}^{\uparrow}}$.

Constructive story: A FormalBallDomain$^{\uparrow}$ packet is generated from FormalBallUp windows, FormalBallRadiusOrderUp comparisons, FormalBallBasisUp coverage, directed finite domain rows, FormalBallCompletionUp handoff, MetricCompletionUp and RealUp endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, seed)

Scope closed: Definition 5723.1, Definition 5723.2, Definition 5723.3, and Definition 5723.4 seed the finite formal-ball domain route from rounded neighbourhood data to completion and Real endpoint consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FormalBallDomain}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, host point equality, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-order exactness, basis coverage, directed-row stability, completion handoff, Real seal boundary, componentwise transport, replay, provenance, and local naming for BEDC.Derived.FormalBallDomainUp.

Chapter 5724

A Concrete Naming Certificate for CauchyCompletionReflectiveLocalization$^{\uparrow}$

Mac Lane reflective subcategories and Bishop–Bridges constructive completion present Cauchy completion as a reflector whose completed objects form a saturated boundary of the metric-name world. The CauchyCompletionReflectiveLocalization$^{\uparrow}$ packet isolates that boundary as finite BEDC NameCert data: it reads the reflective subcategory of chapter 1793, the reflector of chapter 1585, the functorial route of chapter 423, the universal property of chapter 397, the stream-name surface of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the L10 localization boundary used by completion adjoint and universal-reflector consumers; it does not add a quotient completion carrier, selected limit representative, host equality on completed objects, countable-choice selector, ambient completeness theorem, or full Real$^{\uparrow}$ completeness principle.

Definition 5724.1 (CauchyCompletionReflectiveLocalization carrier)**.** *A* CauchyCompletionReflectiveLocalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, Q, I, U, F, K, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and source name-data row, including the displayed metric classifier, finite Cauchy observations,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *reads, and* DyadicRatCore$^{\uparrow}$ *tolerance ledger available before localization is read.*

2. *Q is the Cauchy completion reflector row, read through* CauchyCompletionReflector$^{\uparrow}$ *and* CauchyCompletionFunctc *naming the completed-object surface without exposing a quotient carrier or selected host limit.*

3. *I is the reflective subcategory inclusion row, read through* CauchyCompletionReflectiveSubcategory$^{\uparrow}$*, exposing completed objects as the public inclusion boundary.*

4. *U is the universal-property factorization row, read through* CompletionUniversalProperty$^{\uparrow}$ *after S, Q, I have all been displayed.*

5. *F is the saturated complete-object localization row, saying that a second completion-facing read returns to the same included complete-object surface rather than creating a second host completion.*

6. *K is the StreamName–RegSeqRat–DyadicRatCore–Real transport row, carrying finite windows, regular rational readback, dyadic tolerance accounting, and the terminal* Real$^{\uparrow}$ *seal used by S, Q, I, U, F.*

7. *T stores componentwise* hsame *transport for source metric/name data, the reflector, the inclusion, universal factorization, saturated localization, the local name route, continuation replay, provenance, and local naming.*

8. *H stores the displayed* Cont *replay from source observations through reflector admission, reflective inclusion, universal-property factorization, localization return, and the local StreamName–RegSeqRat–DyadicRatCore–Real route.*

9. *C stores the* Pkg *provenance over* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$, $\mathsf{CauchyCompletionReflectiveSubcatego}$ $\mathsf{CauchyCompletionReflector}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *P is the local provenance ledger tying the source, reflector, inclusion, factorization, localization, transport, replay, and naming rows to the same finite history.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectiveLocalizationUp.CauchyCompletionReflectiveLocalizationTas`
`single carrier alignment`

Theorem 5724.2 (CauchyCompletionReflectiveLocalization NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier* $L = (S, Q, I, U, F, K, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport,* hsame *transport,* Cont *replay,* Pkg *provenance, local provenance, and local naming rows;*

2. *reflector visibility, requiring every completion-facing read to pass through Q after the source row S has displayed the relevant finite windows and tolerance ledger;*

3. *inclusion exactness, requiring every completed-object consumer to pass through I as the reflective subcategory boundary;*

4. *universal factorization, requiring every finite-window continuous consumer to reach its extension route only through U after S, Q, I have been projected;*

5. *localization saturation, requiring repeated completion-facing reads or already-complete reads to return through F to the same included complete-object surface;*

6. *local name-route discipline, requiring* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *data to be consumed only through K;*

7. *transport and replay nonescape, requiring T, H, C, P, N to preserve the displayed dependency surface and to export no quotient completion, selected limit, host equality, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $S, Q, I, U, F, K, T, H, C, P, N$.

Proof. Project the carrier of Definition 5724.1. The displayed rows are the only data available to the local classifier: source metric/name data, reflector, reflective inclusion, universal factorization,

saturated localization, local name transport, structural transport, replay, provenance, and local naming.

The reflector and inclusion obligations are read in order. The row Q supplies the Cauchy completion reflector and functor route from the source data, while I exposes the completed objects only as the reflective-subcategory boundary.

The universal-property and saturation obligations are downstream from those rows. The row U is the sole factorization surface for finite-window continuous consumers, and F records that already-complete or repeated completion-facing reads remain on the same included complete-object surface.

Finally K, T, H, C, P, N preserve and name the route. Since the carrier has no coordinate for quotient completion equality, selected limits, host equality, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness, the NameCert surface cannot export any of those objects. □

Theorem 5724.3 (CauchyCompletionReflectiveLocalization universal exhaustion)**.** *Let* $L = (S, Q, I, U, F, K, T, H$ *be an accepted* $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ *carrier. Every L10 completion consumer that reads a reflector, completed-object inclusion, universal factorization, or repeated-completion return factors through the finite localization route*

$$S \longmapsto Q \longmapsto I \longmapsto U \longmapsto F \longmapsto K \longmapsto T, H, C, P, N.$$

Thus the completion route is exhausted by the source metric/name data, Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName–RegSeqRat–DyadicRatCore–Real transport, hsame *transport,* Cont *replay,* Pkg *provenance, and local NameCert rows of the carrier.*

Proof. Start with an L10 completion consumer and project the source row S. The finite source observations, stream windows, regular rational readbacks, and dyadic tolerances are visible before the consumer reaches any completed-object surface.

Next project the reflector and inclusion rows. The consumer enters completion data through Q and reads completed objects through I, so the reflective subcategory boundary is part of the same finite route rather than an external host collection.

The row U supplies universal-property factorization, and F supplies the saturated return to the same complete-object surface. Hence extension reads and repeated completion-facing reads have no second completion coordinate.

The local transport row K and structural rows T, H, C, P, N close the route. They carry StreamName, RegSeqRat, DyadicRatCore, and Real data together with transport, replay, provenance, and local naming, so the listed carrier rows exhaust every public completion read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}$ packet is generated from source metric/name data, a Cauchy completion reflector, reflective subcategory inclusion, universal-property factorization, saturated complete-object localization, StreamName-RegSeqRat-DyadicRatCore-Real transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, local provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5724.1, Definition 5724.2, and Definition 5724.3 seed the finite reflective-localization boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectiveLocalization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected limits, host equality on completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector visibility, inclusion exactness, universal factorization, localization saturation, local name-route discipline, and transport-replay nonescape for BEDC.Derived.CauchyCompletionReflecti

RieszThorinInterpolation$^{\uparrow}$ reads endpoint bounded-operator and strip-estimate rows through chapter 1644, chapter 3893, and chapter 38 before any interpolated norm-bound ledger is exposed.

Chapter 5725

A Concrete Naming Certificate for $\mathsf{RieszThorinInterpolation}^{\uparrow}$

Following Riesz, Thorin, and Stein-Weiss, the $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet records the finite interpolation certificate by which two endpoint bounded-operator rows and a Hadamard three-lines strip estimate expose one interpolated norm bound. It sits after the strip-bound packet of chapter 3893, the endpoint normed-space surface of chapter 1950, the bounded-operator row of chapter 1644, the interpolation carrier of chapter 224, and the terminal real-name seal of chapter 38. The chapter names the displayed certificate only; it does not import a host interpolation functor, quotient operator equality, selected extremal analytic family, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Every interpolated-bound consumer must read endpoint bounded-operator rows through chapter 1644 and the strip estimate through chapter 3893 before the $\mathsf{Real}^{\uparrow}$ norm-bound ledger is visible.

Definition 5725.1 (Riesz-Thorin interpolation carrier)**.** *A* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$$

with the following displayed rows:

1. X_0, X_1, Y_0, Y_1 *are endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows for the source and target normed spaces at the two endpoints.*
2. A_0, A_1 *are endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, recording the same displayed operator action with endpoint norm bounds over the corresponding normed-space rows.*
3. E_0, E_1 *are endpoint exponent rows, storing the source and target exponent ledgers and their rational comparison data.*
4. S *is the* $\mathsf{HadamardThreeLines}^{\uparrow}$ *strip-estimate row used to compare the endpoint bounds along the displayed interpolation strip.*
5. Θ *is the interpolation-parameter row, storing the inspected parameter, its endpoint incidence, and the convex-weight ledger consumed by* S*.*
6. B *is the* $\mathsf{Real}^{\uparrow}$ *norm-bound ledger exporting the interpolated bound only after* $A_0, A_1, E_0, E_1, S, \Theta$ *have been displayed.*

7. *H records componentwise* hsame *transport for endpoint normed spaces, bounded operators, exponent rows, strip estimate, interpolation parameter, norm-bound ledger, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an interpolated-bound request through $X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B$*
9. *P is the* Pkg *provenance over* $\mathsf{RieszThorinInterpolation}^{\uparrow}$, $\mathsf{HadamardThreeLines}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Interpolation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows. It has no coordinate for a host interpolation space, quotient equality of operators, selected extremal analytic family, non-displayed exponent completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5725.2 (Riesz-Thorin interpolation NameCert obligations)**.** *For every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *carrier*

$$T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$ *surface consists exactly of endpoint normed-space admission through X_0, X_1, Y_0, Y_1, bounded-operator endpoint admission through A_0, A_1, endpoint exponent admission through E_0, E_1, Hadamard three-lines strip-estimate compatibility through S, interpolation-parameter admission through Θ, interpolated* $\mathsf{Real}^{\uparrow}$ *norm-bound export through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 provide the endpoint normed-space surfaces, and A_0, A_1 are the only bounded-operator coordinates available to the classifier.

The exponent and strip data are then forced by row order. The endpoint exponent rows E_0, E_1 and the interpolation parameter Θ determine the displayed convex weights, while the $\mathsf{HadamardThreeLines}^{\uparrow}$ row S is the sole strip-estimate route that can compare endpoint bounds.

The bound row B reads only after those endpoint and strip rows are displayed. It exports the interpolated $\mathsf{Real}^{\uparrow}$ norm-bound ledger, not a selected analytic family or a host interpolation space.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since every excluded host object lacks a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5725.3 (Riesz-Thorin horizontal dependency route)**.** *Every accepted interpolated-bound read for a* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *packet exposes the Hadamard three-lines strip row, endpoint* $\mathsf{NormedSpace}^{\uparrow}$ *rows, endpoint* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *rows, interpolation-parameter row, and terminal* $\mathsf{Real}^{\uparrow}$ *bound row before the interpolated estimate is visible. Equivalently, every consumer route factors through the displayed dependency lattice*

$$X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N$$

and through the sibling certificate surfaces of chapter 3893, chapter 1950, chapter 1644, chapter 224, and chapter 38.

Proof. Project the carrier $T = (X_0, X_1, Y_0, Y_1, A_0, A_1, E_0, E_1, S, \Theta, B, H, C, P, N)$ of Definition 5725.1. The rows X_0, X_1, Y_0, Y_1 are exactly the endpoint normed-space rows, and A_0, A_1 are exactly the endpoint bounded-operator rows over those spaces.

The rows E_0, E_1, Θ record the endpoint exponent and interpolation-parameter data consumed by the strip comparison. The only strip-estimate coordinate is S, so the interpolated estimate cannot be read before the Hadamard three-lines surface named in chapter 3893 is displayed.

The terminal bound row B is downstream of the endpoint, operator, exponent, strip, and interpolation-parameter rows. The structural rows H, C, P, N transport, replay, package, and name that same dependency lattice, and they introduce no bypass around the sibling NormedSpace, BoundedLinearOperator, Interpolation, HadamardThreeLines, or Real surfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}$.

Constructive story: A $\mathsf{RieszThorinInterpolation}^{\uparrow}$ packet is generated from endpoint NormedSpace rows, endpoint BoundedLinearOperator rows, endpoint exponent rows, a HadamardThreeLines strip-estimate row, an interpolation-parameter row, a RealUp norm-bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5725.1, Definition 5725.2, and Definition 5725.3 seed the finite Riesz-Thorin interpolation carrier over endpoint normed spaces, bounded operators, exponent rows, strip estimates, interpolation parameters, norm-bound ledgers, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RieszThorinInterpolation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host interpolation spaces, quotient operator equality, selected extremal analytic families, non-displayed exponent completion, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint normed-space admission, bounded-operator admission, endpoint exponent compatibility, HadamardThreeLines strip-estimate compatibility, interpolation-parameter admission, RealUp norm-bound export, transport, replay, provenance, and local naming for BEDC.Derived.RieszThorinInterpolationUp.

The top-level $\mathsf{RieszThorinInterpolation}^{\uparrow}$ route records the sibling lattice that every interpolated norm-bound read must traverse. The proof surface is anchored in the three-lines estimate of chapter 3893, the normed source and target rows of chapter 1950, the operator-action rows of chapter 1644, the interpolation-family carrier of chapter 224, and the scalar seal of chapter 38.

Lemma 5725.4 (Riesz–Thorin sibling lattice route)**.** *Every accepted* $\mathsf{RieszThorinInterpolation}^{\uparrow}$ *consumer reads its endpoint bounds, analytic-strip comparison, interpolation parameter, bounded-linear-operator action, norm rows, and terminal* $\mathsf{Real}^{\uparrow}$ *scalar estimate through the displayed sibling lattice before the interpolated bound is public. The route exports no host complex interpolation theorem, quotient operator equality, selected analytic family, or ambient completeness coordinate beyond those sibling rows.*

Chapter 5726

A Concrete Naming Certificate for $\mathsf{HoeffdingFiniteSum}^{\uparrow}$

Following Hoeffding's finite bounded-sum inequality, the $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet records the finite concentration route by which independent bounded random variables yield an exponential tail estimate. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law and range summaries of chapter 186, the independence ledger of chapter 187, the dyadic exponential-bound arithmetic of chapter 291, the terminal seal of chapter 38, and the finite-average probability route of chapter 4373. The packet names a bounded finite-sum concentration certificate; it is not a martingale inequality, a Chernoff optimization principle, an almost-sure convergence theorem, a host measure-theoretic limit, or an ambient $\mathsf{Real}^{\uparrow}$ completeness result.

Definition 5726.1 (Hoeffding finite-sum carrier)**.** *A* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, I, B, S, D, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the finite family of* $\mathsf{RandomVar}^{\uparrow}$ *entries read over P. The row I is the* $\mathsf{Independence}^{\uparrow}$ *ledger certifying that the displayed family is independent in the finite index range under inspection. The row B is the bounded-range ledger: for each displayed variable it records lower and upper* $\mathsf{Real}^{\uparrow}$ *seals and the corresponding* $\mathsf{Distribution}^{\uparrow}$ *range summary. The row S records the centered finite-sum expression and the positive deviation request. The row D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *exponential-bound row naming the finite dyadic numerator, denominator, square, range-width sum, and slack comparison used by the displayed Hoeffding bound. The row E is the terminal* $\mathsf{Real}^{\uparrow}$ *tail seal naming the probability estimate for the exceptional event. The row H records componentwise* hsame *transport, C records* Cont *replay of the finite concentration route, Q records* Pkg *provenance over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Independence}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$, *and N is the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.HoeffdingFiniteSumUp.HoeffdingFiniteSumTasteGate single carrier alignment`

Theorem 5726.2 (Hoeffding finite-sum NameCert obligations)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *carrier $(P, V, I, B, S, D, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5726.1. Carrier admission displays the probability source, finite random-variable family, independence ledger, bounded-range ledger, finite-sum row, dyadic*

exponential-bound row, $\mathsf{Real}^{\uparrow}$ *tail seal, transport, replay, provenance, and local naming row. The certificate imports no host concentration theorem, martingale filtration, Chernoff variational optimizer, countable-choice schedule, external probability-limit operation, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Read the carrier rows in order. The source row P supplies the event classifier and normalization, while V exposes only a finite family of random variables over that same probability source.

The independence row I and bounded-range row B are then finite ledgers. They state exactly which variables are independent and which displayed lower and upper seals bound each variable before any exponential estimate is available.

The finite-sum row S fixes the centered sum and requested deviation. The dyadic row D records the arithmetic denominator, squared deviation, accumulated range-width budget, and slack comparison used by the exponential estimate, and the $\mathsf{Real}^{\uparrow}$ row E seals the resulting tail bound.

Finally H, C, Q, N transport, replay, source, and name the same route. Since the carrier has no coordinate for a host concentration theorem, variational optimizer, infinite limit, or completeness principle, these displayed rows exhaust the local certificate surface. □

Theorem 5726.3 (Hoeffding finite tail-bound route)**.** *Every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *tail-bound read factors through the route*

$$\mathsf{ProbSpace}^{\uparrow},\ \mathsf{RandomVar}^{\uparrow},\ \mathsf{Independence}^{\uparrow},\ \mathsf{Distribution}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by the carrier rows $P, V, I, B, S, D, E, H, C, Q, N$*. A probability-bound consumer can inspect the exceptional-event estimate only after the finite probability source, random-variable family, independence ledger, bounded-range ledger, centered-sum row, dyadic exponential-bound row, terminal real seal, transport, replay, provenance, and local naming row have been displayed.*

Proof. Start with an accepted carrier and project the source rows. The probability source P and finite random-variable family V provide the only measurable data available to the tail-bound request.

Next read I and B. The independence ledger is the only route by which the product-event step enters the packet, and the bounded-range ledger is the only route by which distributional lower and upper seals enter the range-width budget.

The centered-sum row S then determines the finite event being bounded. The dyadic row D supplies the exponential-bound arithmetic for that event, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream from the same displayed dyadic comparison.

The structural rows H, C, Q, N preserve that route under classifier replacement and local naming. Therefore every tail-bound read is the finite Hoeffding handoff itself, not a hidden appeal to an external concentration theorem or ambient real-completeness principle. □

Theorem 5726.4 (Hoeffding finite-sum nonescape)**.** *For every accepted* $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ *packet, every concentration-facing read remains inside the displayed rows* $P, V, I, B, S, D, E, H, C, Q, N$*. In particular, the packet exports no martingale filtration, Chernoff optimizer, almost-sure convergence claim, host probability-limit operator, countable-choice selection, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. The carrier of Definition 5726.1 has no coordinate beyond the eleven displayed rows. Thus a concentration-facing request can only project the probability source, finite variables, independence

ledger, range ledger, sum row, dyadic estimate, real tail seal, transport, replay, provenance, and local name.

The analytic content of the request is concentrated in I, B, S, D, E. Independence, bounded ranges, the finite centered sum, dyadic exponential arithmetic, and the terminal tail seal appear as explicit rows, so no further theorem object can be read from the packet.

The remaining rows H, C, Q, N are structural. They preserve displayed histories, replay the finite route, record package provenance, and attach the local certificate. Hence the accepted packet cannot export any of the excluded limit, optimizer, quotient, choice, or completeness principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}$.

Constructive story: A $\mathsf{HoeffdingFiniteSum}^{\uparrow}$ packet is generated from a probability source row, finite random-variable family, independence ledger, bounded-range ledger, centered finite-sum row, dyadic exponential-bound row, RealUp tail seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5726.1, Definition 5726.2, Definition 5726.3, and Definition 5726.4 seed the bounded finite-sum concentration route through ProbSpace, RandomVar, Independence, Distribution, DyadicRatCore, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HoeffdingFiniteSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close martingale inequalities, Chernoff optimization, almost-sure convergence, host probability limits, countable-choice selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite independence routing, bounded-range stability, dyadic exponential-bound handoff, tail-seal nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.HoeffdingFiniteSumUp.

Chapter 5727

A Concrete Naming Certificate for RegularCauchyLatticeOperations$^{\uparrow}$

Following Bishop and Bridges, the RegularCauchyLatticeOperations$^{\uparrow}$ packet records the finite constructive lattice-operation surface for regular Cauchy real names. It reads the regular rational route of chapter 301, the terminal real-name seal of chapter 38, the dyadic arithmetic surface of chapter 291, the lower-member packet of chapter 1832, and the upper-member packet of chapter 1833. The certificate gives downstream Real$^{\uparrow}$ consumers one finite min/max interface without importing trichotomy, quotient real equality, selected limits, countable choice, host order completeness, or ambient Real$^{\uparrow}$ completeness.

Definition 5727.1 (Regular Cauchy lattice operations carrier)**.** *A* RegularCauchyLatticeOperations$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two accepted* Real$^{\uparrow}$ *rows presented by regular Cauchy names, exposing only their finite* RegSeqRat$^{\uparrow}$ *readbacks and terminal real seals.*
2. *T is the shared tolerance row used to align the lower-member and upper-member requests.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *ledger for the displayed comparison scale, branch budgets, and readback arithmetic consumed by both operation rows.*
4. *M is the* RegularCauchyMinimum$^{\uparrow}$ *lower-member row.*
5. *A is the* RegularCauchyMaximum$^{\uparrow}$ *upper-member row.*
6. *R is the paired regular rational readback row for the two operation outputs.*
7. *E is the paired* Real$^{\uparrow}$ *seal row, reached only after M, A, R have been displayed.*
8. *H records componentwise* hsame *transport for the two input names, tolerance, dyadic arithmetic, minimum row, maximum row, readback, seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from a lattice-operation request through $X, Y, T, D, M, A, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* RegularCauchyLatticeOperations$^{\uparrow}$*,* RegularCauchyMinimum$^{\uparrow}$*,* RegularCauchyMaximu RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, T, D, M, A, R, E, H, C, P, N$. The carrier has no coordinate for trichotomy, quotient real equality, host order completeness, selected limits, countable choice, a host lattice structure, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.RegularCauchyLatticeOperationsUp.RegularCauchyLatticeOperationsTasteGate`
`single carrier alignment`

Theorem 5727.2 (Regular Cauchy lattice operations NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier*

$$L = (X, Y, T, D, M, A, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two regular Cauchy real rows, shared tolerance row, dyadic arithmetic ledger, minimum row, maximum row, paired readback, paired seal, transport, replay, provenance, and local name;*
2. *shared-scale exposure, requiring T and D to be displayed before either M or A may be consumed;*
3. *lower-member routing, requiring the minimum output to be read through the* $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *row M;*
4. *upper-member routing, requiring the maximum output to be read through the* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *row A;*
5. *readback and seal ordering, requiring R to expose the paired regular rational outputs before E exposes the paired* $\mathsf{Real}^{\uparrow}$ *seals;*
6. *classifier stability, requiring componentwise* hsame *transport of $X, Y, T, D, M, A, R, E, C, P, N$ to preserve the accepted operation route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, host lattice structure, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5727.1. Its displayed rows are exactly the two regular Cauchy real inputs, the shared tolerance row, the dyadic ledger, the two operation rows, the paired readback and seal rows, and the structural rows.

Shared-scale exposure follows because M and A have no independent tolerance coordinate inside this packet. Both operation rows read the same T, D surface before any output row can be consumed.

The lower-member and upper-member obligations are inherited as named rows, not as host operations. The packet can read the lower member only through M and the upper member only through A.

Readback and seal ordering is forced by R, E. The real-facing consumer first receives the paired regular rational readback and only then the paired $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming are exhausted by H, C, P, N, so no excluded host or quotient object is available. □

Theorem 5727.3 (Regular Cauchy lattice operations min-max readback)**.** *Let* $L = (X, Y, T, D, M, A, R, E, H, C, P$ *be an accepted* $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the lattice-operation output factors through the finite route*

$$X, Y, T, D \longmapsto M, A \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Consequently the packet exports the lower and upper regular Cauchy operation outputs only as paired $\mathsf{RegularCauchyMinimum}^{\uparrow}$ *and* $\mathsf{RegularCauchyMaximum}^{\uparrow}$ *readbacks over the same dyadic scale. It exports no trichotomy, quotient real equality, host order completeness, selected limit, countable choice, or host lattice operation.*

Proof. Start with a real-facing read of the carrier. The only input rows are X, Y, and the only scale rows are T, D, so the consumer must read those coordinates before either operation row is available.

The operation rows are M and A. By Definition 5727.1, M names the lower-member route and A names the upper-member route; neither row carries a separate host order principle.

After the operation rows are displayed, the readback row R exposes the paired regular rational outputs and the seal row E exposes the paired $\mathsf{Real}^{\uparrow}$ boundary. The structural rows H, C, P, N preserve and name this route without adding any coordinate beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5727.1, Definition 5727.2, and Definition 5727.3 seed a finite min/max operation route for two regular Cauchy real names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLatticeOperations}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, host order completeness, selected limits, countable choice, host lattice operations, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shared-scale exposure, lower-member routing, upper-member routing, readback ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLatticeOperationsUp.

Constructive story: A $\mathsf{RegularCauchyLatticeOperations}^{\uparrow}$ packet is generated from two regular Cauchy Real rows, a shared dyadic tolerance surface, RegularCauchyMinimum and RegularCauchyMaximum rows, paired regular rational readback, a paired RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5728

A Concrete Naming Certificate for $\mathsf{CauchyModulusBridge}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet records the finite handoff by which a displayed Cauchy-criterion request becomes an explicit modulus schedule before any regular-rational readback or real-name seal is consumed. It sits after the effective criterion packet of chapter 3695, the modulus schedule surface of chapter 292, the regular rational readback of chapter 301, the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the L10 bridge from criterion-facing reads to explicit schedules as finite NameCert data; it is not quotient stream equality, a selected host limit, countable choice, trichotomy, host sequence equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5728.1 (Cauchy modulus bridge carrier)**.** *A* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, M, E, S, D, R, L, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyCriterion}^{\uparrow}$ *request row naming the late-pair comparison surface to be served by the bridge.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row assigning the displayed precision request to a finite tail threshold.*
3. *E is the* $\mathsf{EffectiveCauchyCriterion}^{\uparrow}$ *row tying the same criterion request and modulus schedule to a finite effective handoff.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the criterion request and scheduled by the modulus row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the radius, tail comparison, and endpoint budget used by the request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available only after Q, M, E, S, D have been displayed.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the criterion, modulus, effective handoff, window, tolerance, and regular-rational rows have been read.*
8. *H records componentwise* hsame *transport for the criterion, modulus, effective handoff, stream-window, dyadic, regular-rational, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cauchy-criterion request through $Q, M, E, S, D, R, L, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{CauchyCriterion}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, EffectiveCauchyCriterio StreamName$^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, M, E, S, D, R, L, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host sequence equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate single carrier alignment`

Theorem 5728.2 (Cauchy modulus bridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carrier $B = (Q, M, E, S, D, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed criterion row Q, modulus schedule M, effective criterion handoff E, finite stream window S, dyadic tolerance ledger D, regular-rational readback R, terminal real seal L, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5728.1. The analytic rows visible to a consumer are exactly the criterion request Q, modulus schedule M, effective handoff E, finite window S, dyadic tolerance ledger D, regular-rational readback R, and terminal real seal L.

The bridge obligation is the ordering of those rows. A Cauchy-criterion request first exposes Q and then reads the displayed threshold in M through the effective handoff E; only after that handoff are the stream window S and dyadic tolerance ledger D available for regular readback.

The downstream rows are forced by the same carrier projection. The row R is the $\mathsf{RegSeqRat}^{\uparrow}$ readback obtained from the displayed window and tolerance data, and L is the $\mathsf{Real}^{\uparrow}$ seal reached only through that readback.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5728.3 (Cauchy modulus bridge effective-criterion handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyCriterion}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{EffectiveCauchyCriterion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed criterion request, explicit modulus schedule, effective criterion row, finite stream window, dyadic tolerance ledger, regular rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no quotient stream equality, selected host limit, hidden modulus oracle, countable-choice schedule, trichotomy split, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with Definition 5728.2. It places every consumer read behind the criterion row, the explicit modulus row, and the effective criterion handoff.

Next open the finite observation route. The stream-window row S and dyadic tolerance row D are downstream of Q, M, E, so the regular-rational row R is a displayed readback from the same request rather than a detached sequence or selected limit.

The terminal seal is last. The $\mathsf{Real}^{\uparrow}$ row L is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the route stated in the theorem and cannot bypass it through a hidden analytic object. □

Theorem 5728.4 (Cauchy modulus bridge choice-free schedule)**.** *Let*

$$B = (Q, M, E, S, D, R, L, H, C, P, N) \quad \textit{and} \quad B' = (Q', M', E', S', D', R', L', H', C', P', N')$$

be accepted $\mathsf{CauchyModulusBridge}^{\uparrow}$ *carriers. If they agree componentwise on the displayed rows* $Q, M, E, S, D, R, L, H, C, P, N$*, then they determine the same modulus schedule and the same* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of either packet can insert a countable-choice schedule, hidden modulus oracle, selected host limit, or extra selector coordinate between the criterion request and the terminal seal.*

Proof. Project both carriers through Definition 5728.1. The hypothesis identifies every displayed coordinate of B and B', so the criterion requests, explicit modulus rows, effective handoff rows, stream windows, dyadic ledgers, regular readbacks, Real seals, transport, replay, provenance, and local names are the same.

Use Definition 5728.2 to read the NameCert surface. Its obligation list has no row outside $Q, M, E, S, D, R, L, H, C, P, N$, so there is no remaining coordinate in which a second schedule or selector could differ.

Apply Definition 5728.3. The consumer order is forced as Q then M then E then S then D then R then L, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream from the same regular-rational readback.

Thus the two accepted packets determine the same choice-free schedule and the same Real seal. The excluded choice and selector objects cannot be inserted because the carrier and its NameCert obligations expose no such coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusBridge}^{\uparrow}$ packet is generated from a CauchyCriterion request row, a CauchyModulus schedule, an effective Cauchy criterion handoff, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5728.1, Definition 5728.2, Definition 5728.3, and Definition 5728.4 seed the finite bridge from Cauchy-criterion requests to explicit modulus schedules before RegSeqRat and Real consumers read the route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusBridge}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CauchyModulusBridgeUp.CauchyModulusBridgeTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, hidden modulus oracles, countable choice, trichotomy, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, criterion-to-modulus exactness, effective handoff ordering, StreamName and DyadicRatCore readback, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.CauchyModulusBridgeUp.

Chapter 5729

A Concrete Naming Certificate for $\mathsf{MyhillNerode}^{\uparrow}$

Following Myhill and Nerode, the $\mathsf{MyhillNerode}^{\uparrow}$ packet records the finite right-congruence classifier behind regular languages and minimal finite automata as displayed BEDC data. It consumes accepted-word rows from chapter 351, presentation rows from chapter 2364, finite carrier rows from chapter 112, and word-spine rows from chapter 36. The packet does not import an external quotient of all languages; it names the finite distinguishability ledger that a minimization consumer can audit.

Definition 5729.1 (Myhill–Nerode carrier)**.** *A* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row, read through* $\mathsf{FinSet}^{\uparrow}$*.*
2. W *is the finite word-window row, read through* $\mathsf{List}^{\uparrow}$ *over* Σ*.*
3. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
4. E *is the finite right-congruence table on the displayed word window.*
5. C *is the context-extension row recording that suffix extension by displayed words preserves the table.*
6. D *is the distinguishability row naming a displayed suffix witness whenever two table rows are separated.*
7. Q *is the finite state row whose states are the displayed equivalence classes of* E*.*
8. δ *is the transition row induced by appending one alphabet symbol to a class representative.*
9. q_0 *is the start class of the empty word.*
10. F *is the accepting-class row, determined by membership in the accepted-word ledger.*
11. μ *is the minimization handoff row from the right-congruence table to the finite automaton presentation.*
12. H *stores componentwise* hsame *transport for alphabet, word-window, language, congruence, suffix, state, transition, accepting, and minimization rows.*

13. *R stores* Cont *replay from a regular-language query through the displayed right-congruence table.*
14. *P stores* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
15. *N is the local* $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N$. It has no coordinate for an infinite language quotient, a nondeterministic presentation, a regular-expression conversion, or a proof that an arbitrary external automaton has already been minimized.

Lean4 ✓ `BEDC.Derived.MyhillNerodeUp.MyhillNerodeTasteGate single carrier alignment`

Theorem 5729.2 (Myhill–Nerode NameCert obligations)**.** *For every accepted* $\mathsf{MyhillNerode}^{\uparrow}$ *carrier*

$$M = (\Sigma, W, L, E, C, D, Q, \delta, q_0, F, \mu, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, word-window exposure, accepted-word observation, right-congruence tabulation, suffix-stability replay, distinguishability witnesses, quotient-state construction, induced transition replay, start-class selection, accepting-class selection, minimization handoff, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5729.1. The rows Σ, W, L, E, C, D expose the finite observation table and the suffix-stability checks used to read right congruence.

The rows Q, δ, q_0, F then repackage the table as a deterministic finite automaton presentation. Since Q is built from the displayed classes of E, the transition row is read by extending a representative by one displayed alphabet symbol and returning to the corresponding class.

The rows μ, H, R, P, N keep the minimization handoff, transport, replay, provenance, and local name attached to the same packet. No omitted coordinate can supply an external quotient, nondeterministic branch, or regular-expression conversion, so the listed rows exhaust the NameCert obligations. □

Theorem 5729.3 (Myhill–Nerode DFA minimization handoff)**.** *Every* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer that claims a minimal deterministic finite automaton through a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet must expose the finite right-congruence row E, the suffix-stability row C, the distinguishability row D, the quotient-state row Q, the induced transition row δ, the accepting-class row F, and the minimization handoff row μ. The handoff may feed a* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation with those rows, but it does not identify arbitrary automaton presentations, choose a nondeterministic branch, or translate the packet into a regular-expression proof.*

Proof. Begin with the obligation theorem Definition 5729.2. It forces every minimization read through the displayed right-congruence table, suffix-stability replay, and distinguishability witnesses before any quotient-state row is consumed.

The quotient construction then uses the finite rows Q, δ, q_0, F to build a deterministic presentation. This presentation is compatible with the finite automaton carrier of chapter 2364 because it exposes finite states, a finite alphabet, a transition table, a start state, accepting states, and replay rows.

The minimization handoff remains bounded by the displayed table. It proves only the certificate route from regular-language observations to the minimal finite presentation named by the packet; stronger equivalence claims are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}$.

Constructive story: A $\mathsf{MyhillNerode}^{\uparrow}$ packet is generated from a finite alphabet row, word-window row, accepted-word observation ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting classes, minimization handoff, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5729.1, Definition 5729.2, and Definition 5729.3 seed the finite right-congruence certificate and its deterministic-minimization handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MyhillNerode}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, host-language quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence stability, distinguishability coverage, quotient-state exactness, induced-transition replay, minimization handoff, TasteGate coverage, transport, replay, provenance, and local naming for BEDC.Derived.MyhillNerodeUp.

Chapter 5730

A Concrete Naming Certificate for $\mathsf{VariationSeminorm}^{\uparrow}$

Following Bishop–Bridges bounded-variation analysis and the Bourbaki treatment of seminormed structures, the $\mathsf{VariationSeminorm}^{\uparrow}$ packet records the finite seminorm-budget surface attached to displayed partition variation data. It sits between the bounded-variation carrier of chapter 398, the bounded-variation real-function row of chapter 3654, the variation-norm consumer of chapter 3804, the Jordan route of chapter 2069, the total-variation function surface of chapter 2257, the regular rational readback of chapter 301, and the terminal real endpoint seal of chapter 38. The packet names displayed partition-variation budgets and seminorm comparison rows without selecting a host supremum over all partitions, a quotient class of functions, or a completed bounded-variation space.

Definition 5730.1 (VariationSeminorm carrier)**.** *A* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, F, J, T, R, Q, A, L, H, C, P, N)$$

with the following displayed rows:

1. *B is the source* $\mathsf{BoundedVariation}^{\uparrow}$ *packet, including its displayed partition ledger, endpoint rows, adjacent-edge observations, finite variation sum, and common-refinement data.*
2. *F is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *read tying the partition ledger to a displayed real-valued function row over the inspected interval.*
3. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *handoff row, exposing only the finite positive- and negative-variation ledgers available from the same partition data.*
4. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *observation row for the displayed partition windows inspected by the packet.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal for all scalar values and variation budgets that the packet compares.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row used when scalar budgets and endpoint values are inspected through regular rational names.*
7. *A is the displayed variation-seminorm row. It records the null-row witness for constant differences, finite additivity over displayed concatenations, scalar budget transport, and monotone comparison of partition refinements.*

8. *L is the seminorm ledger, listing the partition windows, budget rows, finite variation sums, constant-row witnesses, and refinement comparisons used by A.*
9. *H records componentwise* hsame *transport for the bounded-variation source, real-function row, Jordan route, total-variation observation, Real seal, regular-name readback, seminorm row, ledger, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a seminorm-budget request through $B, F, J, T, R, Q, A, L, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{VariationNorm}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed source packet, real-function row, Jordan route, total-variation observation, Real and RegSeqRat scalar reads, seminorm ledger, componentwise transport, replay, provenance, and local name. It has no coordinate for a host supremum, quotient equality of functions, a completed seminormed space, a measure-theoretic variation object, or bridge checking.

Lean4 ✓ `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate single carrier alignment`

Theorem 5730.2 (VariationSeminorm NameCert obligations)**.** *For every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *carrier $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, bounded-variation real-function readback, Jordan-route availability, total-variation observation, Real endpoint sealing, RegSeqRat scalar readback, displayed seminorm-row formation, ledger exactness, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming through the displayed rows.*

Proof. Project the finite carrier of Definition 5730.1. The analytic data available to the packet are the bounded-variation source B, the real-function row F, the Jordan handoff J, the total-variation observation T, the Real seal R, and the regular-name readback Q.

The seminorm row A is then formed from the displayed budget comparisons in those rows. Its null-row, finite-additivity, scalar-transport, and refinement-monotonicity clauses are recorded in the ledger L, so each seminorm request is tied to inspected partition windows rather than to a hidden family of all partitions.

Finally H, C, P, N transport, replay, source, and name exactly the same finite surface. Since the carrier has no host-supremum, quotient-function, completed-space, variation-measure, or bridge coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5730.3 (VariationSeminorm refinement budget route)**.** *Every accepted* $\mathsf{VariationSeminorm}^{\uparrow}$ *packet exports the finite route*

$$\mathsf{BoundedVariation}^{\uparrow}, \mathsf{BoundedVariationRealFunction}^{\uparrow} \longrightarrow \mathsf{JordanVariationDecomposition}^{\uparrow}, \mathsf{TotalVariationFunction}^{\uparrow} \longrightarrow$$

to bounded-variation completion and regulated-integration consumers. The route transports displayed partition-variation budgets and seminorm comparison rows, but it exports no host supremum over all partitions, quotient equality of functions, selected Jordan decomposition, completed bounded-variation Banach space, or bridge data.

Proof. Begin with an accepted packet $S = (B, F, J, T, R, Q, A, L, H, C, P, N)$. A consumer first reads the bounded-variation source and real-function rows, so every endpoint value and edge variation remains tied to the finite partition windows in B and F.

The Jordan and total-variation rows are then read only as displayed handoffs. They expose finite positive- and negative-variation ledgers and total-variation observations attached to the same partition windows.

The scalar budget route is sealed by Q and R. The seminorm ledger L records the null, additivity, scalar-transport, and refinement-comparison rows used by A, while H, C, P, N transport, replay, source, and name the packet. No remaining coordinate can supply the excluded host objects, so the exported route is exactly the displayed finite seminorm-budget route. □

Falsifiable prediction: Every accepted $\mathsf{VariationSeminorm}^{\uparrow}$ read factors through a displayed $\mathsf{BoundedVariation}^{\uparrow}$ source, a $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ row, $\mathsf{JordanVariationDecomposition}^{\uparrow}$ and $\mathsf{TotalVariationFunction}^{\uparrow}$ handoff rows, $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ scalar reads, and the local seminorm ledger, transport, replay, provenance, and naming rows.

Independence witness: boundedvariation, bounded-variation-real-function, variation-norm, jordan-variation-decomposition, total-variation-function, regseqrat, and real: no listed sibling is bijective to $\mathsf{VariationSeminorm}^{\uparrow}$ because this packet stores the seminorm-specific null, finite-additivity, scalar-transport, refinement-budget, ledger, replay, provenance, and local naming rows together as one finite surface.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationSeminorm}^{\uparrow}$ packet is generated from a BoundedVariation source, a BoundedVariationRealFunction read, JordanVariationDecomposition and TotalVariationFunction handoff rows, RegSeqRat and Real scalar reads, a displayed seminorm row, a finite budget ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5730.1, Definition 5730.2, and Definition 5730.3 seed the finite variation-seminorm carrier, obligation surface, and refinement-budget route over displayed bounded-variation and scalar-readback rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationSeminorm}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.VariationSeminormUp.VariationSeminormTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host suprema over all partitions, quotient equality of functions, selected Jordan decompositions, completed bounded-variation Banach spaces, variation measures, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, real-function readback, Jordan and total-variation handoff availability, Real and RegSeqRat scalar exactness, seminorm ledger exactness, classifier transport, replay, provenance, local naming, and nonescape for BEDC.Derived.VariationSeminormUp.

Chapter 5731

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5731.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5731.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5731.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5731.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5731.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5732

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5732.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5732.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5732.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5732.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5733

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5733.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5733.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5733.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5734

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5734.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5734.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5734.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5734.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5735

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5735.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate single carrier alignment`

Theorem 5735.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5735.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5736

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5736.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5736.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5736.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5737

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5737.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5737.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5738

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5738.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5738.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5738.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5739

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5739.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5739.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5739.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5739.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5740

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5740.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5740.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5740.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5741

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5741.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5741.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5741.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5742

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5742.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert. *The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5742.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5742.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5743

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5743.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5743.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5743.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5743.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5744

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5744.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5744.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5744.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5744.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5745

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5745.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5745.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5745.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5746

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5746.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5746.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5746.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5746.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5746.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5747

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5747.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5747.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5747.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5748

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5748.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5748.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5748.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5748.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5748.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5748.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5749

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5749.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5749.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5749.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5750

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5750.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5750.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5750.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5751

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5751.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5751.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5751.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5751.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5752

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5752.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5752.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5752.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5753

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5753.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5753.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5753.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5753.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5754

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5754.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5754.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5754.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5754.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5755

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5755.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5755.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5755.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5756

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5756.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5756.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5756.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

$\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5757

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5757.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5757.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5757.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5757.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5757.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5758

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5758.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5758.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5759

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5759.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5759.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^{\uparrow}$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5759.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^{\uparrow}$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5760

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5760.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5760.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5760.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5761

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5761.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5761.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5761.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5762

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5762.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5762.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5762.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5763

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5763.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5763.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5764

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5764.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5764.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5764.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5765

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5765.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5765.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5765.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5766

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5766.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5766.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5766.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5767

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5767.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5767.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5767.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

Chapter 5768

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5768.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate`
`single carrier alignment`

Theorem 5768.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5768.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5769

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5769.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5769.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5769.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5770

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5770.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*

10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*

12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5770.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*

2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*

3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*

4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*

5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*

6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*

7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5770.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5770.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5771

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5771.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5771.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5771.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5772

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5772.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5772.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5772.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5773

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5773.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5773.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5773.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5774

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5774.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate single carrier alignment`

Theorem 5774.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5774.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5775

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5775.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5775.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5775.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5776

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5776.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5776.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5776.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5777

A Concrete Naming Certificate for $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on functoriality, the $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet records the finite BEDC surface where a displayed real-completion request is read functorially through bounded windows rather than through a quotient completion object. It sits over the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the completion-map discipline of chapter 1937. The packet gives downstream completion consumers one named window-functor route before any quotient stream, selected limit, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle is available.

Definition 5777.1 (RealCompletionWindowFunctor carrier)**.** *A* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row naming the source and target observation windows used by the completion request.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the source and target windows at the requested precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows and tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have all been displayed.*
5. *F is the functorial map row, read through* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*, recording how the displayed source window is transported to the displayed target window.*
6. *H records componentwise* hsame *transport for W, D, R, E, F, C, P, N.*
7. *C records displayed* Cont *replay from a completion-window request through W, D, R, E, F, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the nine displayed rows. It has no coordinate for quotient completion classes, selected representatives, host real equality, countable-choice schedules, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealCompletionWindowFunctorUp.RealCompletionWindowFunctorTasteGate single carrier alignment`

Theorem 5777.2 (RealCompletionWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the displayed finite packet* Q*;*
2. *analytic readback order* $W \longmapsto D \longmapsto R \longmapsto E$*;*
3. *functorial map exposure through the row* F *over the same windows and tolerances;*
4. *structural stability through* hsame *transport and displayed* Cont *replay;*
5. *provenance and local naming through* P *and* N*.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5777.1. The accepted packet displays exactly $W, D, R, E, F, H, C, P, N$, so every local NameCert read must be a read of one of those coordinates.

The analytic route is ordered by the displayed rows. The finite StreamName windows W are inspected before the dyadic tolerance ledger D, the regular rational readback R, and the terminal Real seal E.

The functorial row F is attached to the same window and tolerance ledger through $\mathsf{CauchyCompletionFunctoriality}$ Thus a completion consumer receives only the displayed map action on the finite route, not a quotient completion map or a selected representative.

Finally the rows H, C, P, N transport, replay, source, and name the same packet. Since no excluded object occurs as a carrier coordinate, no additional local obligation is available. □

Theorem 5777.3 (RealCompletionWindowFunctor finite route)**.** *Every completion-facing read exported by an accepted* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto D \longmapsto R \longmapsto E \longmapsto F \longmapsto H, C, P, N.$$

Consequently the functorial completion-map row is consumed only after the StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, and Real seal boundary have been displayed inside the same packet.

Proof. Apply Definition 5777.2. Its obligation surface exposes exactly carrier admission, analytic readback order, functorial map exposure, structural stability, provenance, and local naming.

Read the analytic side first. The rows W, D, R, E display the finite observation windows, dyadic tolerance ledger, regular rational readback, and terminal Real seal.

Then read the functorial side through F and the structural rows. The rows H, C, P, N keep that map action attached to the same finite packet. Therefore the completion consumer cannot detach the functorial map from the displayed window route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5777.1, Definition 5777.2, and Definition 5777.3 seed the finite completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, CauchyCompletionFunctoriality, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full functor laws for all completion objects, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window order, tolerance readback, Real-seal routing, functorial-map exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a terminal Real seal, a CauchyCompletionFunctoriality map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5778

A Concrete Naming Certificate for $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$

The $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet realizes the Real and Completion route of chapter 9238 as finite window-functor data before any completion consumer reads a limit-facing object. It binds the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the displayed completion functor row of chapter 5777. The packet names a finite completion-window functor surface for the Real completeness spine; it does not assert quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5778.1 (RealCompletenessWindowFunctor carrier)**.** *A* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (V, W, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *V is the originating Real-completeness request row read from chapter 9238.*
2. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row exposed before any regular-rational or Real-seal read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same window request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that consumes W and D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, D, R have been displayed.*
6. *F is the* $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$ *row that packages the finite functorial completion handoff.*
7. *H records componentwise* hsame *transport for $V, W, D, R, E, F, C, P, N$.*
8. *C records displayed* Cont *replay from a Real-completeness request through $V, W, D, R, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$, $\mathsf{RealCompletionWindowFunctor}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for quotient completion classes, selected representatives, countable-choice schedules, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5778.2 (RealCompletenessWindowFunctor NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ *carrier*

$$Q = (V, W, D, R, E, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission, originating Real-completeness request exposure, StreamName window exposure, DyadicRatCore tolerance exposure, RegSeqRat readback exposure, Real seal exposure, finite completion-window functor exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Carrier admission is projection from Definition 5778.1. The tuple displays exactly $V, W, D, R, E, F, H, C, P, N$ so the local NameCert can inspect only those rows.

The Real-completeness request row V is read before the finite analytic route. The rows W, D, R, E then display StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real seal.

The functorial row F is the only completion-window handoff. It packages the Real-completion functorial read after the displayed analytic route rather than replacing it by a quotient completion object.

The rows H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ read displays $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows before any completion consumer reads the packet.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchycompletionfunctoriality: $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ is not bijective to any listed sibling because it jointly retains the originating Real-completeness request, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, completion-window functor row, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5778.1 and Definition 5778.2 seed the finite Real-completeness completion-window functor route over StreamName, DyadicRatCore, RegSeqRat, Real, RealCompletionWindowFunctor, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessWindowFunctor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, full Real-completeness proof, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, request exposure, finite-window exposure, tolerance exposure, regular readback, Real-seal exposure, completion-window functor exposure, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessWindowFunctorUp.

Constructive story: A $\mathsf{RealCompletenessWindowFunctor}^{\uparrow}$ packet is generated from a Real-completeness request row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, a RealCompletionWindowFunctor row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5779

A Concrete Naming Certificate for $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet records the explicit modulus route by which a bounded compact-real sequence yields inspected subsequence windows before any downstream consumer reads a real cluster seal. It sits over the constructive subsequence packet of chapter 5830, the sibling Bolzano–Weierstrass route of chapter 1254, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite subsequence-modulus carrier for compact-real consumers; it is not an open-cover compactness theorem, not an arbitrary-choice selector, not quotient real equality, and not ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5779.1 (Compact real subsequence modulus carrier)**.** *A* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, S, B, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real source row, exposing the bounded* $\mathsf{Real}^{\uparrow}$ *sequence request and the compact-metric inspection data visible to the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing exactly the finite source windows inspected by the modulus.*
3. *B is the* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *and* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *route row, carrying the interval-refinement and subsequence scheduling surface without adding a host compactness principle.*
4. *W is the subsequence-modulus row, recording selected indices, monotonicity, cofinal scheduling, and the finite window assignment for each requested precision.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording radius requests, split budgets, comparison windows, and common-scale data used by the selected windows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same selected windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, visible only after K, S, B, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for the source, stream windows, route, modulus, tolerance, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-real subsequence request through source inspection, stream windows, route selection, modulus scheduling, dyadic tolerance accounting, regular rational readback, and the real seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$, $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, S, B, W, D, R, E, H, C, P, N$. It has no coordinate for open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactRealSubsequenceModulusUp.TasteGate.CompactRealSubsequenceModulusTasteGate`
`single carrier alignment`

Theorem 5779.2 (Compact real subsequence modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *carrier*

$$M = (K, S, B, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$ *surface consists exactly of compact real source exposure through K, finite stream-window exposure through S, constructive Bolzano–Weierstrass route exposure through B, subsequence-modulus scheduling through W, dyadic tolerance accounting through D, regular rational readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5779.1. Its analytic rows are the compact real source K, stream windows S, constructive route B, subsequence modulus W, tolerance ledger D, regular readback R, and real seal E.

The modulus obligation is finite because W records selected indices, monotonicity, cofinal scheduling, and selected windows as packet data. The route row B imports the displayed ConstructiveBolzanoWeierst and $\mathsf{BolzanoWeierstrass}^{\uparrow}$ subsequence construction rather than an ambient compactness theorem.

The readback obligation then passes through D and R. The same selected windows receive the dyadic tolerance budget before they can be read as a $\mathsf{RegSeqRat}^{\uparrow}$ sequence.

Finally E, H, C, P, N seal, transport, replay, source, and name only those rows. Since the carrier contains no open-cover, arbitrary-choice, quotient-equality, selected-least-cluster, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5779.3 (Constructive Bolzano–Weierstrass modulus route)**.** *Every* $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ *consumer that reads a compact real subsequence route from an accepted carrier must factor through*

$$\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow},\quad \mathsf{BolzanoWeierstrass}^{\uparrow},\quad \mathsf{StreamName}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow}$$

The factorization exposes finite source windows, a displayed subsequence-modulus row, dyadic tolerance accounting, regular rational readback, and a terminal real seal; it exports no open-cover compactness, arbitrary-choice selector, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Apply Definition 5779.2. It places the constructive and sibling Bolzano–Weierstrass route row before the modulus row and before any regular rational readback.

The finite windows are exactly the row S, and the selected windows are exactly the row W. Thus any consumer that changes the source window or selected window must leave the displayed carrier.

The dyadic tolerance ledger D is read before R, and R is read before the terminal real seal E. This orders the packet through the stated route and prevents the excluded host principles from entering the subsequence-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5779.1, Definition 5779.2, and Definition 5779.3 seed the explicit compact-real subsequence-modulus route through finite windows, dyadic tolerances, regular readback, and real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealSubsequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, arbitrary choice, quotient real equality, selected least cluster points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, constructive Bolzano–Weierstrass route exposure, subsequence-modulus scheduling, dyadic tolerance accounting, RegSeqRat readback, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealSubsequenceModulusUp.

Constructive story: A $\mathsf{CompactRealSubsequenceModulus}^{\uparrow}$ packet is generated from compact real source data, finite StreamName windows, ConstructiveBolzanoWeierstrass and BolzanoWeierstrass route data, subsequence-modulus scheduling, DyadicRatCore tolerance accounting, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5780

A Concrete Naming Certificate for MetaCICNormalEndpointComparator$^{\uparrow}$

The MetaCICNormalEndpointComparator$^{\uparrow}$ packet realizes the bounded-conversion endpoint bridge named by the MetaCIC decidability audit map of chapter 9379. It sits between the bounded normal conversion bridge of chapter 3439, the decidable conversion frontier of chapter 4097, the normalization frontier of chapter 892, the normal-form consistency seal of chapter 740, and the critical-path packet of chapter 765. The packet is a finite comparator for finished bounded normal endpoints; it is not a global beta-conversion decider, not a strong-normalization theorem, and not a host equality oracle.

Definition 5780.1 (MetaCIC normal endpoint comparator carrier)**.** *A* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, L, R, B, F, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed closed-term pair row supplied by the decidable conversion frontier.*
2. *L and R are the left and right bounded-normal computation traces, each carrying fuel, output syntax, and finished-endpoint evidence.*
3. *B is the* BoundedNormalConversionBridge$^{\uparrow}$ *row, exposing the bounded comparison result and its soundness route.*
4. *F is the* MetaCICNormalizationFrontier$^{\uparrow}$ *row, recording that both bounded traces have reached named normal endpoints at the comparator fuel.*
5. *S is the* NormalFormConsistencySeal$^{\uparrow}$ *row, checking that the two endpoint witnesses are read through the same normal-form consistency surface.*
6. *Q is the* MetaCICCriticalPath$^{\uparrow}$ *row that keeps the comparator tied to the decidability and normalization critical path rather than to an unrestricted conversion oracle.*
7. *H records componentwise* hsame *transport for the term pair, endpoint traces, bridge, frontier, seal, critical-path, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a bounded conversion request through endpoint computation, finished-normal evidence, endpoint comparison, consistency sealing, and critical-path accounting.*

9. *P is the* Pkg *provenance over* MetaCICNormalEndpointComparator$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, NormalFormConsistencySeal$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, R, B, F, S, Q, H, C, P, N$. It has no coordinate for unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotienting of syntax, or host equality.

Theorem 5780.2 (MetaCIC normal endpoint comparator NameCert obligations)**.** *For every accepted* MetaCICNormalEndpointComparator$^{\uparrow}$ *carrier*

$$E = (T, L, R, B, F, S, Q, H, C, P, N),$$

the local NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$ *surface consists exactly of typed closed-term pair admission through T, left and right bounded endpoint traces through L, R, bounded conversion bridge exposure through B, normalization-frontier endpoint evidence through F, normal-form consistency sealing through S, critical-path accounting through Q, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5780.1. The computational rows are exactly the term pair T, the endpoint traces L, R, the bounded conversion bridge B, the normalization frontier F, the consistency seal S, and the critical-path row Q.

The endpoint evidence is finite because L and R carry bounded traces at a displayed fuel. The row F records that those traces have reached finished normal endpoints before the comparator is consumed.

The comparison then passes through B and S. The bridge supplies the bounded conversion handoff, and the seal checks that the endpoint witnesses are compared through the same normal-form consistency surface.

Finally Q, H, C, P, N keep the packet on the MetaCIC critical path, transport it componentwise, replay it, source it, and name it. Since the carrier contains no unrestricted conversion, fuel-erased normalization, strong-normalization, proof-irrelevance, syntax-quotient, or host-equality coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5780.3 (Bounded conversion endpoint handoff)**.** *Every* MetaCICNormalEndpointComparator$^{\uparrow}$ *consumer that promotes bounded conversion evidence to a finished endpoint comparison must factor through*

MetaCICDecidableConversionFrontier$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, No

The handoff supplies only typed closed-term pair data, bounded traces, finished endpoint evidence, endpoint consistency, critical-path accounting, transport, replay, provenance, and local naming.

Proof. Use Definition 5780.2. The typed closed-term pair is read first through T, while L and R record the bounded normal computations to be compared.

The row F is the endpoint boundary: it must be displayed before a bounded comparison can be read as a finished normal-form comparison. The bridge row B and seal row S then compare those endpoints through the bounded-conversion and normal-form consistency surfaces.

The critical-path row Q prevents the handoff from being read as a global conversion theorem. The structural rows H, C, P, N preserve and name only this finite route, so the consumer receives no unrestricted conversion decider or host equality oracle. □

Theorem 5780.4 (Closed-normal comparator vision obligations)**.** *The* MetaCICNormalEndpointComparator$^{\uparrow}$ *packet exposes exactly the finite comparator surface consumed by chapter 9376: closed normal endpoint rows, false-endpoint exclusion rows, and comparator determinacy rows are read through the displayed term pair T, bounded traces L, R, bounded conversion bridge B, normalization frontier F, normal-form consistency seal S, critical-path row Q, componentwise transport H, replay C, provenance P, and local naming N. No consumer of this surface receives unrestricted conversion decidability, fuel-erased normalization, proof irrelevance, syntax quotient equality, or host equality.*

Proof. Project the carrier of Definition 5780.1. The closed normal endpoint rows are precisely the finished bounded traces L, R read together with the normalization-frontier evidence F.

The false-endpoint exclusion row is read through the normal-form consistency seal S and the critical-path row Q. Those rows are the only coordinates tying the finished endpoint comparison to the closed-normal consistency boundary of chapter 9376.

Comparator determinacy is then read through the bounded conversion bridge B over the same typed pair T. The structural rows H, C, P, N transport, replay, source, and name this finite route.

Since the carrier contains no unrestricted conversion decider, fuel-erased normalizer, proof-irrelevance coordinate, syntax quotient, or host equality oracle, the listed rows are the complete comparator obligation surface consumed by the closed-normal consistency route. □

Falsifiable prediction: A MetaCIC conversion consumer that reads a finished endpoint comparison while omitting either bounded endpoint trace, the bounded-normal-conversion bridge, the normalization-frontier endpoint evidence, the normal-form consistency seal, the critical-path row, transport, replay, provenance, or local naming refutes MetaCICNormalEndpointComparator$^{\uparrow}$.

Independence witness: metacic-decidable-conversion-frontier, bounded-normal-conversion-bridge, metacicnormalizationfrontier, normal-form-consistency-seal, metaciccriticalpath: MetaCICNormalEndpointCompara is not bijective to any listed sibling because it jointly carries typed closed-term pair data, two bounded endpoint traces, bounded conversion bridge data, finished-normal endpoint evidence, consistency sealing, critical-path accounting, transport, replay, provenance, and local NameCert rows in one comparator packet.

The Lean follow-up must provide explicit Nontrivial MetaCICNormalEndpointComparator$^{\uparrow}$ and FieldFaithful MetaCICNormalEndpointComparator$^{\uparrow}$ instances in TasteGate.lean for the same finite carrier.

Closure status for NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, seed)

Scope closed: Definition 5780.1, Definition 5780.2, and Definition 5780.3 seed the finite comparator that hands bounded conversion evidence to finished normal endpoints.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICNormalEndpointComparator}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim unrestricted conversion decidability, fuel-erased normalization, strong normalization, proof irrelevance, quotient syntax equality, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint trace exactness, bounded bridge exposure, normalization-frontier endpoint evidence, normal-form consistency sealing, critical-path accounting, transport, replay, provenance, local naming, and TasteGate nontriviality with field faithfulness for BEDC.Derived.MetaCICNormalEndpointComparatorUp.

Constructive story: A MetaCICNormalEndpointComparator$^{\uparrow}$ packet is generated from typed closed-term pair data, two bounded endpoint traces, BoundedNormalConversionBridge data, MetaCICNormalizationFrontier endpoint evidence, NormalFormConsistencySeal data, MetaCICCriticalPath accounting, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5781

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5781.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5781.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5781.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5781.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5782

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5782.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5782.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5782.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5783

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5783.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5783.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5783.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5784

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5784.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5784.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5784.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5785

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5785.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5785.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5786

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5786.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5786.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5787

A Concrete Naming Certificate for $\mathsf{ObservabilityGramian}^{\uparrow}$

The $\mathsf{ObservabilityGramian}^{\uparrow}$ interface records the finite observability Gramian packet of a finite-dimensional linear control system. Its textbook source is the finite-horizon observability Gramian in Kalman's control-theoretic duality and the matrix-ledger presentation used by Kailath, Sayed, and Hassibi for linear estimation. The BEDC packet is the finite dual certificate read beside $\mathsf{ReachabilityGramian}^{\uparrow}$ in chapter 455; it is not an ambient analytic observability theorem.

The dependency anchors are the finite observation surface of chapter 294, the reachability dual surface of chapter 297, the matrix rows of chapter 49, the vector-space rows of chapter 47, and the linear-map rows of chapter 48. These anchors force the Gramian to be read as displayed $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ data before a control consumer can compare it with the reachability Gramian.

5787.1 Carrier definition

Definition 5787.1 (ObservabilityGramianUp finite carrier)**.** *An* $\mathsf{ObservabilityGramian}^{\uparrow}$ *carrier is a finite* BHist *packet with rows*

$$(A, C, n, W, L, R, P, N).$$

Here A is the displayed transition-matrix row, C is the displayed output-matrix row, n is a unary finite-horizon row, W is the displayed finite Gramian-sum ledger with summand rows

$$C^{\mathsf{T}}C, \quad A^{\mathsf{T}}C^{\mathsf{T}}CA, \quad (A^{\mathsf{T}})^2C^{\mathsf{T}}CA^2, \quad \ldots, \quad (A^{\mathsf{T}})^{n-1}C^{\mathsf{T}}CA^{n-1},$$

L is the componentwise hsame *transport ledger for the transition, output, horizon, summand, and Gramian endpoint rows, R is the finite dual-readback row comparing W with the observation-matrix rows $C, CA, \ldots, CA^{n-1}$ of* $\mathsf{ControlObservability}^{\uparrow}$*, P is the* Pkg *provenance over* $\mathsf{ObservabilityGramian}^{\uparrow}$*,* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*,* $\mathsf{ReachabilityGramian}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*, and N is the local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by componentwise hsame *transport of the A, C, n, W, L, R, P, N rows. It does not quotient by a host unobservable subspace, analytic trajectory family, Riccati equation, spectral factorization, infinite-horizon Gramian, or external estimator object.*

Lean4 ✓ `BEDC.Derived.ObservabilityGramianUp.ObservabilityGramianTasteGate single carrier alignment`

5787.2 Certificate surface

Theorem 5787.2 (ObservabilityGramianUp NameCert obligations)**.** *The local* $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$ *obligation surface has exactly five finite rows:*

1. *carrier admission for the displayed* A, C, n, W, L, R, P, N *packet of Definition 5787.1;*
2. *classifier transport under componentwise* hsame *of the transition, output, horizon, Gramian, summand-ledger, dual-readback, package, and local certificate rows;*
3. *finite sum ledger exactness, saying that the Gramian endpoint inside* W *is read only as the finite sum of the displayed adjoint-output summands inside* W*;*
4. *observability handoff, saying that the observation-matrix surface of* $\mathsf{ControlObservability}^{\uparrow}$ *can consume* A*,* C*,* n*,* W*,* L*, and* R *only through the displayed* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* Cont*, and* Pkg *rows;*
5. *dual non-escape, saying that no accepted route exports a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator.*

Proof. The carrier row is Definition 5787.1. It displays the transition row, output row, unary horizon, Gramian endpoint, summand ledger, transport ledger, dual-readback row, package provenance, and local naming row as finite BHist coordinates.

Classifier transport is componentwise. Replacing each displayed coordinate by an hsame-matched coordinate leaves the same finite packet shape, so the transported packet is again admitted by the carrier.

For finite sum exactness, inspect W. Each summand is a displayed matrix product formed from A, C, and transposes, and the Gramian endpoint is the finite fold of precisely those rows. The readback row R ties that ledger to the observation rows without introducing a host observability theorem.

The observability handoff is a projection to the shared A, C, and n rows, to the finite observation rows of $\mathsf{ControlObservability}^{\uparrow}$, and to the Gramian sum named by W. Because those rows are already packaged with $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, and $\mathsf{LinMap}^{\uparrow}$ provenance, a downstream observability certificate can read them without importing analytic control-system semantics.

Dual non-escape follows from constructor inspection of the carrier. The packet has no coordinate for a host unobservable subspace, analytic output trajectory, Riccati solution, spectral factorization, infinite-horizon Gramian, covariance process, or external estimator, so none of those objects can be projected by a consumer route. □

Theorem 5787.3 (ObservabilityGramianUp finite dual handoff)**.** *Every* $\mathsf{ControlObservability}^{\uparrow}$*,* $\mathsf{ControlControllability}^{\uparrow}$*, or* $\mathsf{ReachabilityGramian}^{\uparrow}$ *consumer of an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet can read only the displayed* A *row,* C *row, unary horizon row* n*, finite Gramian-summand and endpoint row* W*,* hsame *transport ledger* L*, dual-readback row* R*,* Pkg *provenance, and local* NameCert *row of Definition 5787.1. The handoff supplies the finite dual route from observation rows to the reachability-side Gramian ledger, but it supplies no host Kalman decomposition, no analytic trajectory semantics, and no infinite-horizon duality theorem.*

Proof. Project the accepted packet to the rows A, C, n, W, L, R, P, N. The first three rows are the finite control-system coordinates shared with $\mathsf{ControlObservability}^{\uparrow}$, while W records the finite adjoint-output Gramian ledger and L records componentwise transport over those same coordinates.

The dual-readback row R ties the summands in W to the observation rows $C, CA, \ldots, CA^{n-1}$. Thus a $\mathsf{ControlObservability}^{\uparrow}$ consumer reads the packet through the observation matrix surface, and a $\mathsf{ControlControllability}^{\uparrow}$ or $\mathsf{ReachabilityGramian}^{\uparrow}$ consumer reads only the dual comparison route named by the shared package provenance.

The carrier contains no row for a host Kalman decomposition, analytic trajectory semantics, or infinite-horizon duality theorem. Therefore the handoff is exactly the finite BEDC packet projection and does not add a control-theory semantic bridge. □

Theorem 5787.4 (ObservabilityGramianUp ledger non-escape)**.** *Every exported route from an accepted* $\mathsf{ObservabilityGramian}^{\uparrow}$ *packet factors through the finite rows* A, C, n, W, L, R, P, N *of Definition 5787.1. In particular, the packet exports no ambient unobservable subspace, no analytic output trajectory family, no Riccati solution, no spectral factorization, no stochastic covariance process, no infinite-horizon Gramian, and no external numerical estimator.*

Proof. Take an exported consumer route and project the carrier. The available coordinates are exactly the transition row, output row, horizon row, finite Gramian-summand and endpoint row, transport ledger, dual-readback row, package provenance, and local certificate row.

The NameCert obligations account for carrier admission, classifier transport, finite sum exactness, observability handoff, and dual non-escape. Each obligation is stated over those displayed coordinates and is preserved by the stored hsame, Cont, and Pkg rows.

Thus every export factors through the listed finite coordinates. Since the carrier contains no ambient unobservable subspace, analytic output trajectory family, Riccati solution, spectral factorization, stochastic covariance process, infinite-horizon Gramian, or external numerical estimator, no accepted route can supply one downstream. □

Closure status for $\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}$.

Constructive story: An $\mathsf{ObservabilityGramian}^{\uparrow}$ packet is generated from displayed transition and output matrix rows, a unary finite horizon, the finite adjoint-output Gramian summand and endpoint row, componentwise hsame transport ledger, a dual-readback row to the observation-matrix surface, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5787.1, Definition 5787.2, Definition 5787.3, and Definition 5787.4 bind the finite carrier, five-row NameCert surface, dual handoff, and non-escape boundary for observability Gramian packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObservabilityGramian}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host unobservable-subspace semantics, analytic output trajectories, Riccati theory, spectral factorization, stochastic covariance, infinite-horizon Gramians, numerical estimators, or a checked Lean certificate.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked finite-ledger theorems for finite sum exactness, dual handoff, and non-escape.

Chapter 5788

A Concrete Naming Certificate for $\mathsf{HelmholtzDecomposition}^{\uparrow}$

Following Helmholtz and Hodge-style decomposition theory, the $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a displayed vector-field row is read as the sum of a gradient-potential row and a divergence-free row over a Hilbert source. It consumes the Hilbert carrier of chapter 91, the vector-space operations of chapter 47, the linear-map rows of chapter 48, and the projection-theorem surface of chapter 2374. The packet names only the displayed analytic decomposition route; it does not assert PDE solvability, global boundary regularity, unique gauges, Sobolev completeness, or a host Hodge theorem.

Definition 5788.1 (Helmholtz decomposition carrier)**.** *A* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, V, G, D, O, B, T, C, P, N)$$

with the following displayed rows:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *source row. It supplies the ambient inner-product and norm-facing carrier on which the vector field and both components are read.*
2. *V is the vector-field row. It records the admitted field endpoint together with the* $\mathsf{VecSpace}^{\uparrow}$ *addition, subtraction, and scalar rows used to state the splitting.*
3. *G is the gradient-potential row. It names a potential window and the displayed* $\mathsf{LinMap}^{\uparrow}$ *route that reads its gradient as a vector-field component.*
4. *D is the divergence-free row. It records the classifier saying that the second component lies in the displayed kernel row selected by the divergence operator.*
5. *O is the orthogonal-projection row. It reads the gradient component and the divergence-free component through the projection-theorem surface of chapter 2374.*
6. *B is the boundary ledger. It records exactly which boundary observations are available before a projection or divergence-free read is exposed.*
7. *T records componentwise* hsame *transport for the Hilbert source, vector field, gradient-potential row, divergence-free row, projection row, boundary ledger, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a Helmholtz-decomposition request through $H, V, G, D, O, B, T, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{HelmholtzDecomposition}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, OrthogonalProjectionThec BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, V, G, D, O, B, T, C, P, N$. It has no coordinate for a host PDE solution operator, selected gauge, ambient Sobolev space, hidden boundary regularity theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Lean4 ✓ `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate single carrier alignment`

Theorem 5788.2 (Helmholtz decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *carrier $K = (H, V, G, D, O, B, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation families:*

1. *Hilbert-source admission for the displayed* $\mathsf{Hilbert}^{\uparrow}$ *row H;*
2. *vector-field admission for the field endpoint and* $\mathsf{VecSpace}^{\uparrow}$ *operations stored in V;*
3. *gradient-potential admission for the potential window and* $\mathsf{LinMap}^{\uparrow}$ *gradient route stored in G;*
4. *divergence-free admission for the kernel classifier stored in D;*
5. *orthogonal-projection admission for the projection row O over the gradient and divergence-free components;*
6. *boundary-ledger admission for the displayed boundary observations stored in B;*
7. *componentwise transport, displayed replay, provenance, and local naming through T, C, P, N.*

Consequently a downstream PDE, variational, or Hodge-facing consumer may read Helmholtz data only through the displayed Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows. The packet exports no host PDE solver, gauge selector, Sobolev-completion theorem, quotient vector-field equality, or unrestricted Hodge decomposition.

Proof. Project the carrier of Definition 5788.1. The visible analytic rows are the Hilbert source H, the vector-field row V, the gradient-potential row G, the divergence-free row D, the orthogonal-projection row O, and the boundary ledger B.

The source and algebraic rows are read first. The Hilbert row comes from chapter 91, while V supplies the vector operations inherited from chapter 47. Thus the decomposition statement is formed only inside a displayed carrier.

Next the two component rows are exposed. The gradient-potential row G factors its gradient read through the linear-map surface of chapter 48, and the divergence-free row D supplies the displayed kernel classifier before any splitting conclusion is available.

The projection and boundary rows then bind the route. The row O reads the orthogonal splitting discipline through chapter 2374, while B restricts the boundary observations available to that read.

Finally T, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of K, the listed obligation families exhaust the local NameCert surface. □

Theorem 5788.3 (Helmholtz decomposition orthogonal splitting route)**.** *Every splitting read accepted from a* $\mathsf{HelmholtzDecomposition}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Hilbert}^{\uparrow} \longmapsto \mathsf{VecSpace}^{\uparrow} \longmapsto \mathsf{LinMap}^{\uparrow} \longmapsto \mathsf{OrthogonalProjectionTheorem}^{\uparrow} \longmapsto \mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}^{\uparrow}}.$$

The exported surface is a displayed field row together with a gradient-potential component, a divergence-free component, an orthogonal projection ledger, and a boundary ledger. It does not provide a global PDE solution operator, a selected gauge, Sobolev regularity, quotient equality of vector fields, or the full Hodge decomposition theorem.

Proof. Start from an accepted packet $K = (H, V, G, D, O, B, T, C, P, N)$. A consumer first opens the Hilbert row H, so every norm, orthogonality, and projection read remains attached to the same displayed source.

The vector-space row V then supplies the algebra needed to read the field as the sum of two displayed components. This step uses only the finite operations exposed by chapter 47.

The gradient-potential component is read through G and the linear-map surface of chapter 48. The divergence-free component is read through D, whose kernel classifier is already part of the packet.

The orthogonal splitting claim is finally routed through O and the projection-theorem packet of chapter 2374. The boundary ledger B records the available boundary observations before the local name N is exposed.

Transport and replay through T and C preserve this order, while P records the provenance of the same finite route. No host solver, gauge selector, Sobolev regularity theorem, quotient equality, or unrestricted Hodge theorem is introduced. □

Closure status for $\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5788.1, Definition 5788.2, and Definition 5788.3 seed the finite Helmholtz decomposition packet over Hilbert, vector-space, linear-map, projection, boundary, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HelmholtzDecomposition}\uparrow}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HelmholtzDecompositionUp.HelmholtzDecompositionTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim PDE solvability, global gauges, Sobolev regularity, quotient vector-field equality, unrestricted Hodge decomposition, or bridge checking.

Upgrade path: Obligation closure asks for checked Hilbert-source admission, vector-field admission, gradient-potential admission, divergence-free admission, orthogonal-projection admission, boundary-ledger admission, transport, replay, provenance, and local naming for BEDC.Derived.HelmholtzDecompositionUp.

Constructive story: A $\mathsf{HelmholtzDecomposition}^{\uparrow}$ packet is generated from a displayed Hilbert source, a vector-field row, a gradient-potential row read through LinearMapUp, a divergence-free kernel row, an orthogonal-projection row, a boundary ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5789

A Concrete Naming Certificate for $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet records the separated completion boundary needed by the L10 Real and Completion route as finite BEDC data. It sits after the separation predicate of chapter 1925, the separated-completion carrier of chapter 1612, the metric completion route of chapter 1506, the finite stream interface of chapter 126, the dyadic tolerance core of chapter 291, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the envelope through displayed rows only; it does not quotient filters, select representatives, identify host equalities, invoke countable choice, or import ambient completeness.

Definition 5789.1 (Hausdorff completion envelope carrier)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (H, S, M, W, D, R, L, T, K, P, N)$$

where H *is the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$ *separation row,* S *is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* L *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* T *is componentwise* $\mathsf{BHist/hsame}$ *transport,* K *is displayed* Cont *replay,* P *is* Pkg *provenance, and* N *is the local* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *row. A consumer of* E *receives a separated completion boundary only through these coordinates.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeTasteGate single carrier alignment`

Theorem 5789.2 (Hausdorff completion envelope obligations)**.** *Every accepted* $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$ *read must expose the carrier of Definition 5789.1 in the order*

$$H, S, M, W, D, R, L, T, K, P, N.$$

The separation row H *and separated-completion row* S *are available before metric completion, stream observation, dyadic tolerance, regular readback, and real-seal rows are consumed; the transport, replay, provenance, and naming rows preserve the same displayed packet.*

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeNameCertObligations`

Proof. Project the carrier of Definition 5789.1. The packet has no completion-facing coordinates except $H, S, M, W, D, R, L, T, K, P, N$.

The Hausdorff and separated-completion rows are the first boundary data in the packet. Thus a consumer cannot inspect the metric completion route, finite stream windows, dyadic tolerances, regular readback, or terminal real seal without first reading the displayed separation surface.

Finally, the structural rows T, K, P, N only transport, replay, source, and name the same finite packet. They do not create quotient filters, selected representatives, host equality, choice-selected routes, or ambient completeness. □

Theorem 5789.3 (Hausdorff completion separated metric route)**.** *Every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of a* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier factors through the separated route*

$$H, S \longmapsto M \longmapsto W, D, R, L.$$

The route records a separated metric-completion boundary for the L10 Real and Completion interface, not a quotient completion, a host equality theorem, a choice-selected representative, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier separated metric route`

Proof. Start from Definition 5789.2. It places the Hausdorff row H and separated-completion row S before the metric completion row M and before the stream, dyadic, regular-rational, and real-seal rows.

Read the metric completion route through M. Since M is downstream of H, S, every completion-facing use of M carries the displayed separated boundary with it.

The final Real-facing read is the finite route W, D, R, L. Because these rows are still coordinates of the same carrier, the route cannot be replaced by quotient representatives, host equality, countable choice, or ambient completeness. □

Theorem 5789.4 (Hausdorff completion envelope sibling route)**.** *A* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *consumer may hand its separated boundary to* $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ *only through the displayed* $\mathsf{HausdorffSpace}^{\uparrow}$*,* $\mathsf{SeparatedCompletion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of*

$$E = (H, S, M, W, D, R, L, T, K, P, N).$$

The handoff contains no quotient-filter coordinate, representative-selection coordinate, host-equality coordinate, or ambient-completeness coordinate.

Proof. Apply Definition 5789.2. It places the Hausdorff row H, separated-completion row S, metric-completion row M, stream row W, dyadic row D, regular-rational row R, and terminal Real row L inside one displayed packet.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

The sibling $\mathsf{HausdorffUniformCompletion}^{\uparrow}$ consumer therefore reads the same separated boundary as its input surface rather than receiving a separate primitive. Since the carrier only has the displayed rows and the structural rows T, K, P, N, quotient filters, selected representatives, host equality, and ambient completeness remain outside the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeCarrier sibling route`

Theorem 5789.5 (Hausdorff completion envelope Cauchy-net consumer link)**.** *Every* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *consumer of an accepted* $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ *carrier must read the same* $\mathsf{HausdorffSpace}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before its directed-net completion route is consumed. The downstream sibling is chapter 3794, and the handoff is the displayed route*

$$H, S, M, W, D, R, L \longmapsto \mathsf{CauchyNetCompletion}^{\uparrow}.$$

It exports no quotient net, selected representative, host equality, countable-choice schedule, or ambient completeness coordinate.

Proof. Project the carrier $E = (H, S, M, W, D, R, L, T, K, P, N)$ of Definition 5789.1. The only completion-facing rows available to a consumer are the Hausdorff row H, separated-completion row S, metric-completion route M, finite stream row W, dyadic tolerance row D, regular readback row R, and terminal Real row L.

Apply Definition 5789.2. It places H, S, M, W, D, R, L inside one displayed packet before the structural rows T, K, P, N are used for transport, replay, provenance, and local naming.

Next use Definition 5789.3. The separated boundary H, S is already carried through the metric and Real-facing route M, W, D, R, L.

Therefore a downstream $\mathsf{CauchyNetCompletion}^{\uparrow}$ read can only consume the envelope through the same separated metric route before its directed-net completion surface is inspected. Since the packet has no quotient-net, selected-representative, host-equality, countable-choice, or ambient-completeness row, none of those objects can enter the handoff. □

Lean4 ✓ `BEDC.Derived.HausdorffCompletionEnvelopeUp.HausdorffCompletionEnvelopeForwardLink`

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5789.1, Definition 5789.2, and Definition 5789.3 name the separated completion envelope surface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffCompletionEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, host equality, choice-selected representatives, or ambient completeness.

Upgrade path: Obligation closure requires the carrier route to be tied to the sibling completion obligation theorems and then exposed as a public chapter surface.

Constructive story: A $\mathsf{HausdorffCompletionEnvelope}^{\uparrow}$ packet is generated from Hausdorff separation, separated-completion boundary, metric-completion route, finite stream windows, dyadic tolerance, regular-rational readback, Real seal, transport, replay, provenance, and local NameCert rows.

Chapter 5790

A Concrete Naming Certificate for $\mathsf{BoundedLinearFunctional}^{\uparrow}$

Following Banach and Bourbaki functional analysis, the $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet records the scalar-valued bounded-linear-operator row that duality and weak-star consumers inspect before any host dual space is available. It sits over the linear-map route of chapter 48, the bounded-linear-operator surface of chapter 1644, the scalar-bound neighbour chapter 3603, the Banach-dual row of chapter 2265, the finite weak-star window of chapter 3420, the scalar endpoint seal of chapter 38, and the normed source surface of chapter 1950. The chapter names only displayed scalar-functional data: source rows, linearity rows, scalar target readback, norm-bound evidence, transport, replay, provenance, and local naming.

Definition 5790.1 (Bounded linear functional carrier)**.** *For a displayed* $\mathsf{NormedSpace}^{\uparrow}$ *source row* E *over the scalar endpoint surface of* $\mathsf{Real}^{\uparrow}$*, a* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E)$ *carrier is a finite* BHist *packet*

$$\Lambda = (E, L, T, K, O, H, C, P, N)$$

with the following displayed rows:

1. *E is the source normed-space row, including its carrier endpoints, scalar action, additive row, norm row, and classifier.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row from E to the scalar* $\mathsf{Real}^{\uparrow}$ *target.*
3. *T is the scalar-valued* $\mathsf{BoundedLinearOperator}^{\uparrow}(E, \mathsf{Real}^{\uparrow})$ *row carrying the same endpoint action as L.*
4. *K is the* $\mathsf{Real}^{\uparrow}$ *nonnegative bound endpoint for the functional.*
5. *O is the* $\mathsf{OperatorNorm}^{\uparrow}$ *handoff row exposing where the bound K is used.*
6. *H records componentwise* hsame *transport for the source, linear-map row, bounded-operator row, scalar bound, operator-norm handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a scalar-functional request through E, L, T, K, O, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedLinearFunctional}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{OperatorNorm}^{\uparrow}$, $\mathsf{BanachDual}^{\uparrow}$, $\mathsf{BanachAlaoglu}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed source classifier, scalar endpoint transport, bounded-operator transport, bound transport, norm-bound ledger transport, replay, provenance, and local naming rows. It has no coordinate for a host dual space, quotient equality of functions, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BoundedLinearFunctionalUp.BoundedLinearFunctionalTasteGate single carrier alignment`

Theorem 5790.2 (Bounded linear functional NameCert obligations)**.** *For every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *carrier* $\Lambda = (E, L, T, K, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$ *surface consists exactly of source normed-space admission, linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm ledger exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host dual objects. A consumer may inspect* $E, L, T, K, O, H, C, P, N$*, but it receives no ambient function space, quotient operator equality, selected representative, Hahn–Banach witness, unrestricted weak-star compactness theorem, or bridge data.*

Proof. Project the carrier rows. The analytic coordinates are exactly the source normed-space row E, the linear-map row L, the scalar-valued bounded-operator row T, the RealUp bound endpoint K, and the operator-norm handoff O.

The linearity obligation is read through $\mathsf{LinMap}^{\uparrow}$, while boundedness is read through $\mathsf{BoundedLinearOperator}^{\uparrow}(E,$ The scalar target is already the displayed $\mathsf{Real}^{\uparrow}$ endpoint surface, so the functional row never becomes a host-level function.

The bound exactness is the shared route through K and O. The operator-norm handoff records where the scalar bound is used, and componentwise hsame transport moves the source, action, bound, ledger, replay, provenance, and local naming rows together.

No remaining coordinate can manufacture a dual-space quotient, selected representative, Hahn–Banach witness, unrestricted compactness theorem, or bridge check. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 5790.3 (Bounded linear functional operator-norm handoff)**.** *Every accepted* $\mathsf{BoundedLinearFunctional}^{\uparrow}(E$ *packet exports the finite route*

$$\mathsf{LinMap}^{\uparrow}, \mathsf{BoundedLinearOperator}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{OperatorNorm}^{\uparrow} \longmapsto \mathsf{BoundedLinearFunctional}^{\uparrow}$$

to functional-analysis consumers. The route first exposes the carried scalar-valued linear action, then the bounded-operator row, then the RealUp bound endpoint and operator-norm ledger; it does not expose a host supremum, quotient equality of functionals, selected norm-attaining vector, or hidden Banach completeness.

Proof. Begin with an accepted packet $\Lambda = (E, L, T, K, O, H, C, P, N)$. The read of L supplies only the displayed linear-map action from the source row to the scalar target.

The read of T then packages the same action as a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row. Its target row is $\mathsf{Real}^{\uparrow}$, and its bound endpoint is the displayed scalar K.

Finally the operator-norm handoff O records the norm-bound ledger that downstream consumers may inspect. Because no host norm function, quotient functional equality, selected vector, or hidden completion row is present, the exported handoff is exactly the displayed scalar-bound route. □

Theorem 5790.4 (Bounded linear functional dual-row nonescape)**.** *Any read of a* $\mathsf{BanachDual}^{\uparrow}$ *or* $\mathsf{BanachAlaoglu}^{\uparrow}$ *consumer that uses a* $\mathsf{BoundedLinearFunctional}^{\uparrow}$ *row factors through the same source normed-space row, scalar-valued bounded-operator row, RealUp bound endpoint, operator-norm handoff, componentwise transport, replay, provenance, and local naming rows of Definition 5790.1. The duality consumer receives a finite scalar-functional row, not a host dual space, a quotient function class, an unrestricted weak-star compactness theorem, or a selected cluster point.*

Proof. Read the duality consumer first. A $\mathsf{BanachDual}^{\uparrow}$ request needs a scalar-functional row over a displayed source, while a $\mathsf{BanachAlaoglu}^{\uparrow}$ request needs a finite family of such scalar-functional observations.

In either case the only scalar-functional packet available here is Λ. Its source row, bounded-operator row, bound endpoint, operator-norm handoff, and structural rows are projected in the order displayed by Definition 5790.1.

The downstream dual chapters may reuse those rows, but they cannot add coordinates that are absent from the packet. Thus quotient dual objects, host function classes, unrestricted weak-star compactness, selected cluster points, and bridge checks remain outside the read. □

Falsifiable prediction: Every accepted $\mathsf{BoundedLinearFunctional}^{\uparrow}$ scalar-functional read factors through a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ action row, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, and the local hsame, Cont, Pkg, and NameCert rows.

Independence witness: linearmap, boundedlinearoperator, operatornorm, banachdual, banachalaoglu, real, and normedspace: no listed sibling is bijective to $\mathsf{BoundedLinearFunctional}^{\uparrow}$ because this packet stores the scalar target restriction, the dual-facing provenance row, the operator-norm handoff, transport, replay, and local naming together as one finite scalar-functional surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundedLinearFunctional}^{\uparrow}$ packet is generated from a displayed $\mathsf{NormedSpace}^{\uparrow}$ source, a $\mathsf{LinMap}^{\uparrow}$ scalar action, a scalar-valued $\mathsf{BoundedLinearOperator}^{\uparrow}$ row, a $\mathsf{Real}^{\uparrow}$ bound endpoint, an $\mathsf{OperatorNorm}^{\uparrow}$ ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5790.1, Definition 5790.2, Definition 5790.3, and Definition 5790.4 seed the scalar-functional row consumed by BanachDualUp and BanachAlaogluUp.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedLinearFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host dual spaces, quotient equality of functionals, selected representatives, Hahn–Banach separation, unrestricted weak-star compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, scalar linear-map admission, scalar-valued bounded-operator admission, RealUp bound exactness, operator-norm handoff exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.BoundedLinearFunctionalUp.

Chapter 5791

A Concrete Naming Certificate for $\mathsf{CompletionModulusCategory}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the functorial treatment of completions, the $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet records the finite category surface whose arrows preserve displayed completion moduli. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, the completion functor of chapter 426, and the Cauchy completion functor of chapter 423. The packet names modulus-preserving completion morphisms as BEDC rows; it does not export quotient completions, selected representatives, host continuous maps, countable choice, or ambient completeness.

Definition 5791.1 (CompletionModulusCategory carrier)**.** *A* $\mathsf{CompletionModulusCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, A, S, R, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the object ledger. Each object row records a completion-facing source, its displayed Cauchy-modulus discipline, and the local completion target read through the surrounding completion chapters.*
2. *A is the arrow ledger. Each arrow row records a finite morphism request together with the modulus transformation that sends source tolerances to target tolerances.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and target names of an arrow.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source and target moduli.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row read only after S, R, D have supplied the displayed tolerance route.*
7. *F records identity and composition rows for the completion functoriality surface, with identity arrows preserving the same modulus and composite arrows composing the displayed modulus transformations.*
8. *H records componentwise* hsame *transport for object rows, arrow rows, stream windows, regular readback, dyadic tolerances, Real seals, functoriality rows, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from an arrow request through $O, A, S, R, D, E, F, H, P, N$.*
10. *P is the* Pkg *provenance over* CompletionModulusCategory$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite category packet for modulus-preserving completion morphisms. It has no coordinate for quotient completion points, selected representatives, host functions, global choice schedules, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryTasteGate single carrier alignment`

Theorem 5791.2 (CompletionModulusCategory NameCert obligations)**.** *For every accepted* CompletionModulusCa *carrier $M = (O, A, S, R, D, E, F, H, C, P, N)$, the local* NameCert$_{\text{CompletionModulusCategory}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed object ledger, arrow ledger, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, identity and composition rows, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface additionally requires each arrow row in A to expose its source modulus, target modulus, and modulus transformation before any Real$^{\uparrow}$*-facing read is accepted; each identity row in F to preserve the displayed modulus without adding a choice coordinate; and each composition row in F to compose the two displayed modulus transformations before the downstream seal is visible.*

Proof. Project the accepted carrier. The packet contains only the rows $O, A, S, R, D, E, F, H, C, P, N$, so carrier admission and ledger exhaustion are componentwise.

An arrow-facing request first reads the object and arrow ledgers O, A. The source and target names are then inspected through the finite stream windows S, regular rational readback R, and dyadic tolerance ledger D.

The functoriality rows F are local to the same displayed moduli. Identity preserves the stored modulus transformation, and composition combines the two stored transformations before the terminal seal E can be read.

Finally H, C, P, N transport, replay, source, and name this same finite route. Since quotient completions, selected representatives, host functions, global choice schedules, and ambient completeness are not carrier coordinates, they cannot be exported by the local NameCert. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryNameCertObligations`

Theorem 5791.3 (CompletionModulusCategory modulus functoriality)**.** *Every completion-facing consumer of an accepted* CompletionModulusCategory$^{\uparrow}$ *packet reads a modulus-preserving morphism through the finite route*

$$O, A, S, R, D, F, E, H, C, P, N.$$

Thus identity and composition of completion morphisms preserve the displayed modulus data before any Real$^{\uparrow}$ *seal or completion-functor read is exposed. The route factors through* StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, *and* CauchyCompletionFunctor$^{\uparrow}$ *rather than through quotient completion equality, selected representatives, host continuous maps, countable choice, or ambient completeness.*

Proof. Start with a completion-facing morphism request. The object ledger O fixes the source and target completion rows, and the arrow ledger A fixes the modulus transformation to be checked.

The readback rows S, R, D display the inspected stream windows, regular rational observations, and dyadic tolerance comparisons. These rows are the only route by which the requested modulus can face the terminal Real seal.

The functoriality row F then supplies the identity or composition clause. In the identity case the same displayed modulus is replayed. In the composition case the two displayed modulus transformations are composed as finite rows before the result is handed to E.

The structural rows H, C, P, N preserve and name the same route. Therefore every completion-functor consumer receives finite modulus preservation data and no excluded quotient, selection, host-map, choice, or ambient-completeness surface. □

Lean4 ✓ `BEDC.Derived.CompletionModulusCategoryUp.CompletionModulusCategoryModulusFunctoriality`

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}$.

Constructive story: A $\mathsf{CompletionModulusCategory}^{\uparrow}$ packet is generated from completion object rows, modulus-preserving arrow rows, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, RealUp seals, identity and composition rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5791.1, Definition 5791.2, and Definition 5791.3 seed the finite category surface for modulus-preserving completion morphisms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionModulusCategory}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, host continuous maps, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-preserving arrow exactness, identity and composition rows, tolerance readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompletionModulusCategoryUp.

Chapter 5792

A Concrete Naming Certificate for $\mathsf{MarkovTriggerDynamics}^{\uparrow}$

The $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet realizes the finite rank-trigger dynamics of Definition 9462.1541, Definition 9462.1542, and Definition 9462.1546 as BEDC NameCert data. It records the state, support-rank, transition-count, drift, trigger-time, classifier, continuation, provenance, and naming rows needed by a downstream finite-dynamics consumer. The packet is a finite BHist surface for trigger-time readback; it is not a stochastic process on an ambient probability space, not an infinite sample path, not a hidden field model, and not a host Markov-chain object.

Definition 5792.1 (MarkovTriggerDynamics carrier)**.** *A* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, K, A, D, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the finite state row. It records the consumed external-codon count m and the covered support rank r for the current trigger state.*
2. *K is the support-rank row. It records the finite support coordinates currently covered by the observed codon rows and the rank read from that support.*
3. *A is the transition-count row. It stores the finite count of rank-preserving and rank-increasing next draws available from the current state.*
4. *D is the drift row. It records the displayed one-step rank drift computed from the same transition counts.*
5. *G is the trigger-time row. It records the finite recursion for the time to reach full rank and the deterministic threshold row consumed by trigger-time readers.*
6. *H records componentwise* hsame *transport for state, support-rank, transition-count, drift, trigger-time, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a state read through S, K, A, D, G, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$, BHist, hsame, Cont, *finite rank-trigger rows, finite transition-count rows, finite trigger-time rows, and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient probability space, hidden generator field, infinite sample path, external random oracle, host Markov-chain semantics, or unstated independence assumption.

Lean4 ✓ `BEDC.Derived.MarkovTriggerDynamicsUp.MarkovTriggerDynamicsTasteGate single carrier alignment`

Theorem 5792.2 (MarkovTriggerDynamics NameCert obligations)**.** *For every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *carrier* $T = (S, K, A, D, G, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed state row, support-rank row, transition-count row, drift row, trigger-time row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming row.*

The obligation surface requires the transition-count row A *to be computed from the finite support-rank row* K*, the drift row* D *to read only the transition counts in* A*, and the trigger-time row* G *to replay the finite recursion from the displayed state row* S *without adding an external probability carrier.*

Proof. Project the accepted carrier. The packet exposes only $S, K, A, D, G, H, C, P, N$, so every NameCert obligation is a componentwise obligation on those rows.

The state and support rows S, K determine the finite rank-trigger situation. The transition-count row A counts rank-preserving and rank-increasing next draws from that finite support surface.

The drift row D is downstream of A, and the trigger-time row G is downstream of the same state and transition-count data. Thus the trigger-time read is a finite recursion over displayed rows, not an imported Markov-chain object.

Transport, replay, provenance, and local naming preserve the same coordinates. Since the excluded ambient and hidden objects are not coordinates of T, no consumer can read them from the packet. □

Theorem 5792.3 (MarkovTriggerDynamics finite trigger readback)**.** *Every trigger-time consumer of an accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *packet reads the finite route*

$$S, K, A, D, G, H, C, P, N.$$

Consequently the transition law, drift, and trigger-time rows are all determined by finite BHist transition-count data. No trigger-time read exports an infinite sample path, host Markov process, hidden field, or unstated independence source.

Proof. Start with the state row S and support-rank row K. They display the finite input state and the rank information that determines which next-draw counts are available.

The transition-count row A is the only row from which the drift row D may be read. The trigger-time row G then replays the finite recursion from those same transition counts.

Finally H, C, P, N transport, replay, source, and name the finite trigger route. Because no other row exists in the carrier, the stated nonescape clauses follow. □

Falsifiable prediction: Every $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ trigger-time read factors through finite state, support-rank, transition-count, drift, trigger-time, hsame, Cont, Pkg, and NameCert rows.

Independence witness: dna-boundary-tile, rank-trigger, finite-markov-transition: $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries state rows, support-rank rows, transition-count rows, drift rows, trigger-time rows, componentwise hsame transport, displayed Cont replay, provenance, and local naming.

Theorem 5792.4 (MarkovTriggerDynamics sibling dependency route)**.** *Every accepted* $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ *trigger-time read routes its finite state and transition-count rows through chapter 189 or chapter 4074 when the carrier is read as a Markov surface, through chapter 195 when it is read as finite transition evolution, and through chapter 3508 for the DNA boundary-trigger consumer.*

Proof. Project the existing carrier rows $S, K, A, D, G, H, C, P, N$. The state row S, support-rank row K, and transition-count row A provide exactly the finite Markov read, so a Markov-surface consumer reaches this packet through the Markov-chain or finite Markov-partition certificates.

The same rows, together with drift row D and trigger-time row G, form the finite transition-evolution route consumed by the dynamical-system certificate. The structural rows H, C, P, N then transport, replay, source, and name that route for the DNA boundary-trigger consumer.

This is a dependency route only. It does not add an ambient probability space, infinite sample path, unstated independence source, or host Markov-chain object to the carrier. □

Nontrivial TasteGate and FieldFaithful instances remain required before this finite trigger surface can be read as a checked biological field-faithfulness claim.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MarkovTriggerDynamics}^{\uparrow}$ packet is generated from finite state rows, support-rank rows, transition-count rows, drift rows, trigger-time recursion rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5792.1, Definition 5792.2, and Definition 5792.3 seed the finite rank-trigger Markov readback surface for DNA boundary-tile dynamics.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovTriggerDynamics}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient probability spaces, infinite sample paths, hidden field models, host Markov-chain semantics, biological field-faithfulness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-count exactness, drift readback, trigger-time recursion, nontrivial TasteGate coverage, FieldFaithful evidence, transport, replay, provenance, and local naming for BEDC.Derived.MarkovTriggerDynamicsUp.

Chapter 5793

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet records the finite constructive route from regular Cauchy rational names to located completion data and then to the terminal $\mathsf{Real}^{\uparrow}$ seal. It reads the dyadic arithmetic surface of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the completion service exposed by chapter 128, and the real-name boundary of chapter 38. The packet supplies a located completion neighbour for the regular-Cauchy real route without asserting quotient completion equality, selected representatives, host order completeness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5793.1 (Regular Cauchy located completion carrier)**.** *A* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, R, D, Q, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the Cauchy name is inspected at requested precision scales.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger for tolerance comparison, shrinkage, and rational distance estimates.*
4. *Q is the located completion row, recording the displayed located-distance and approximation data extracted from S, R, D.*
5. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row used only as a finite completion handoff for the displayed located data.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback and located completion rows have been displayed.*
7. *H records componentwise* hsame *transport for stream windows, regular rational readbacks, dyadic arithmetic, located completion data, completion service, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a located-completion request through $S, R, D, Q, K, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyLocatedCompletion$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, CompleteMetric$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, Q, K, E, H, C, P, N$. The carrier has no coordinate for quotient completion equality, selected representatives, countable choice, host order completeness, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RegularCauchyLocatedCompletionUp.RegularCauchyLocatedCompletionTasteGate`
`single carrier alignment`

Theorem 5793.2 (Regular Cauchy located completion obligations)**.** *For every accepted* RegularCauchyLocatedCom *carrier*

$$L = (S, R, D, Q, K, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyLocatedCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream-window row, regular rational readback row, dyadic arithmetic ledger, located completion row, complete-metric handoff, real seal, transport, replay, provenance, and local name;*
2. *window-readback coherence, requiring S to expose the finite inspection windows before R may be consumed as a regular rational name;*
3. *dyadic locatedness, requiring D to supply the tolerance comparisons and distance estimates used by the located completion row Q;*
4. *completion handoff ordering, requiring Q to be displayed before the* CompleteMetric$^{\uparrow}$ *row K is exposed;*
5. *real-seal ordering, requiring E to be read only after S, R, D, Q, K have been displayed;*
6. *classifier stability, requiring componentwise* hsame *transport and displayed* Cont *replay to preserve all visible rows;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient completion equality, selected representative, countable choice, host order completeness, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5793.1. Its rows list the finite stream windows, regular rational readback, dyadic arithmetic ledger, located completion data, complete-metric handoff, real seal, and the structural rows.

Window-readback coherence follows because R has no independent observation coordinate in this packet. The regular rational read must pass through S, and the tolerance computations used by that read are then checked against the dyadic row D.

The located completion row Q is the only row that records located-distance data. The completion handoff K consumes that displayed row, and the real seal E is exposed only after the handoff has been displayed.

Transport, replay, provenance, and local naming are exhausted by H, C, P, N. Since quotient completion equality, selected representatives, countable choice, host order completeness, and ambient Real$^{\uparrow}$ completeness are absent from the carrier, no NameCert obligation can export them. □

Theorem 5793.3 (Regular Cauchy located completion Real seal route)**.** *Let* $L = (S, R, D, Q, K, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the located completion read factors through the finite route*

$$S, R, D \longmapsto Q \longmapsto K \longmapsto E \longmapsto H, C, P, N.$$

Thus the packet reaches the real-name boundary only through displayed $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, located completion data, and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff. It does not export quotient completion equality, selected representatives, countable choice, host order completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing read of the packet. The only observable name data are the stream windows S and the regular rational readback R, so any request for a real seal must first inspect those rows.

The dyadic row D supplies the tolerance and distance comparisons used by the located completion data Q. The complete-metric handoff K is then read as a service row for that displayed located data, not as an ambient completeness theorem.

Finally the seal row E exposes the $\mathsf{Real}^{\uparrow}$ boundary, and H, C, P, N transport, replay, source, and name the same displayed route. No remaining carrier coordinate can provide quotient completion equality, selected representatives, choice data, host order completeness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5793.1, Definition 5793.2, and Definition 5793.3 seed a finite route from regular Cauchy names through located completion data to the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host order completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback coherence, dyadic locatedness, completion handoff ordering, real-seal ordering, classifier stability, and non-escape proofs for BEDC.Derived.RegularCauchyLocatedCompletionU

Constructive story: A $\mathsf{RegularCauchyLocatedCompletion}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore arithmetic, located completion data, a CompleteMetric handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5794

A Concrete Naming Certificate for $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$

Following Banach, Picard, and Bishop–Bridges constructive analysis, the $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet records the finite handoff from an explicit contraction estimate to the Cauchy threshold consumed by the regular-rational and real endpoint routes. It sits between the Banach fixed-point modulus packet of chapter 4188, the Picard iteration and error-estimate packets of chapter 2359 and chapter 2233, and the regular Cauchy completion boundary of chapter 5793. The packet gives the endpoint route a single displayed Cauchy-threshold surface without selecting a fixed point, quotienting completion names, or importing ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5794.1 (Picard threshold Cauchy handoff carrier)**.** *A* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, I, E, M, R, K, S, T, C, P, N)$$

with the following visible rows:

1. *B is the* $\mathsf{BanachFixedPointModulus}^{\uparrow}$ *source row whose contraction-rate route has already exposed the rational Lipschitz and finite Picard ledgers.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *ledger of adjacent iterate windows and replay order.*
3. *E is the* $\mathsf{PicardErrorEstimate}^{\uparrow}$ *row, carrying the residual-and-ratio estimate used to bound the Picard tail.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *threshold row produced from B, I, E for each requested dyadic or rational tolerance.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row made available only after the threshold row M has been displayed.*
6. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row consumed at the handoff boundary, not an ambient completeness theorem.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, R, K have been read.*
8. *T stores componentwise* hsame *transports for the Banach-modulus, Picard-iteration, error-estimate, Cauchy-threshold, regular-rational, complete-metric, real-seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from the contraction-rate request through $B, I, E, M, R, K, S, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$, $\mathsf{BanachFixedPointModulus}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{PicardErrorEstimate}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, I, E, M, R, K, S, T, C, P, N$. It has no coordinate for a selected fixed point, host fixed-point equality, quotient completion equality, detached Cauchy selector, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.PicardThresholdCauchyHandoffUp.PicardThresholdCauchyHandoffTasteGate`
`single carrier alignment`

Theorem 5794.2 (Picard threshold Cauchy handoff NameCert obligations)**.** *For every accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier $H = (B, I, E, M, R, K, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyH}}$ *surface consists exactly of carrier admission for the displayed rows; Banach-modulus source visibility through B; Picard-iteration visibility through I; error-estimate visibility through E; Cauchy-threshold construction through M; regular-rational readback through R; complete-metric service access through K; real seal access through S; classifier stability through T and C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 5794.1. The accepted packet exposes exactly the Banach-modulus source row, the Picard-iteration ledger, the error-estimate row, the Cauchy-threshold row, the regular-rational readback, the complete-metric service, the real seal, and the structural rows.

The threshold rows are ordered. The rows B, I, E must be displayed before M can answer a tolerance request, since M is the only coordinate that records the finite Cauchy threshold.

The endpoint rows are downstream. The row R reads the threshold as regular rational data, K is the visible complete-metric service row, and S is exposed only after those rows have been displayed. Transport, replay, provenance, and local naming are exhausted by T, C, P, N, so no further NameCert obligation is present. □

Theorem 5794.3 (Picard threshold Cauchy handoff Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing endpoint read of an accepted* $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BanachFixedPointModulus}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{PicardErrorEstimate}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Complete}$$

Thus the real seal is reached only after the explicit Picard threshold has been displayed as a $\mathsf{CauchyModulus}^{\uparrow}$ *row and repacked through the regular-rational and complete-metric handoff rows. The route exports no selected fixed point, quotient completion equality, detached Cauchy selector, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a real-facing endpoint request. The only contraction-rate source row is B, and its Picard data are refined in the visible rows I and E, so the request first reads those rows.

The threshold stage is forced by M. Apply the Banach fixed-point modulus route of Definition 4188.3 together with the Picard iteration surface of chapter 2359 and the error-estimate surface of chapter 2233. These sources supply the displayed threshold row without adding a selected endpoint.

After M is fixed, the readback and completion rows R and K expose the regular-rational and complete-metric boundary used by chapter 301 and chapter 128. The seal row S is then the only $\mathsf{Real}^{\uparrow}$ coordinate, and the structural rows T, C, P, N preserve the same route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5794.1, Definition 5794.2, and Definition 5794.3 seed the finite Picard-threshold handoff from BanachFixedPointModulus, PicardIteration, and PicardErrorEstimate rows to CauchyModulus, RegSeqRat, CompleteMetric, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, host fixed-point equality, quotient completion equality, detached Cauchy selectors, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-threshold construction, regular-rational readback, complete-metric service access, real-seal ordering, classifier stability, provenance, and local naming for BEDC.Derived.PicardThreshol

Constructive story: A $\mathsf{PicardThresholdCauchyHandoff}^{\uparrow}$ packet is generated from a BanachFixedPointModulus source row, PicardIteration ledger, PicardErrorEstimate row, CauchyModulus threshold, RegSeqRat readback, CompleteMetric service row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5795

A Concrete Naming Certificate for $\mathsf{DiniUniformLimitModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the classical Dini theorem, the $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet extracts the finite uniform-modulus ledger visible in compact-metric monotone convergence data. It sits over chapter 2196, chapter 1436, chapter 129, chapter 130, chapter 463, chapter 369, chapter 301, and chapter 38. The packet names the finite net, rational tolerance budget, and extracted modulus route; it does not import open-cover compactness, chosen limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5795.1 (Dini uniform-limit modulus carrier)**.** *A* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact-metric finite-source observations consumed by the extraction.*
2. *F is the monotone* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, recording the displayed continuous approximants and their pointwise monotone order.*
3. *W is the pointwise window row supplied by the Dini-continuity and Dini-uniform-convergence routes.*
4. *E is the finite net row over K, naming the finite mesh on which pointwise windows are inspected.*
5. *Q is the rational tolerance-budget row, storing the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerances assigned to the finite net and to the later readback.*
6. *U is the extracted* $\mathsf{UniformModulus}^{\uparrow}$ *row, obtained from E, Q, W and the monotone continuous family row F.*
7. *L is the* $\mathsf{UniformLimit}^{\uparrow}$ *handoff row, exposing only the modulus-controlled limit route supplied by U.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational tolerance transport.*
9. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, naming the real-valued read without adding ambient real completeness.*
10. *H records componentwise* hsame *transport for compact-metric source, monotone family, pointwise window, finite net, rational tolerance, extracted modulus, uniform-limit handoff, rational readback, real seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from the compact-metric source and monotone family through the finite net, tolerance budget, extracted modulus, and uniform-limit handoff.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{DiniContinuity}^{\uparrow}$, $\mathsf{DiniUniformConvergence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, W, E, Q, U, L, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected limiting values, quotient function equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DiniUniformLimitModulusUp.TasteGate.DiniUniformLimitModulusTasteGate`
`single carrier alignment`

Theorem 5795.2 (Dini uniform-limit modulus NameCert obligations)**.** *For every accepted* DiniUniformLimitModulu *carrier*

$$D = (K, F, W, E, Q, U, L, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed compact-metric source, monotone continuous family, pointwise window, finite net, rational tolerance budget, extracted uniform modulus, UniformLimit handoff, RegSeqRat readback, Real seal, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires monotone-window visibility through F and W, finite-net extraction through E and Q, modulus handoff through U and L, Real-seal nonescape through S, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses open-cover compactness, selected limiting values, quotient function equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only computational rows are the compact-metric source K, monotone family F, pointwise windows W, finite net E, rational budget Q, extracted modulus U, uniform-limit handoff L, rational readback R, and real seal S.

The monotone-window step is local to F and W. The finite-net step is local to E and Q, so the modulus row U is extracted from displayed finite data rather than from an open cover or a choice of limiting values.

The handoff to $\mathsf{UniformLimit}^{\uparrow}$ passes only through U and L. The rational and real readback rows R and S name the downstream values without adding quotient function equality or ambient completeness.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded objects are not coordinates of D, no NameCert consumer can read them from this carrier. □

Falsifiable prediction: Every accepted $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ read must expose compact-metric source rows, monotone continuous-family rows, pointwise window rows, a finite net, a rational tolerance budget, an extracted uniform modulus, a UniformLimit handoff, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-continuity, diniuniformconvergence, compactmetric, continuousmap, uniformlimit, uniformmodulus, regseqrat, real: $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ is not bijective to any

listed sibling because it binds the finite net and rational tolerance budget to an extracted uniform-modulus handoff before UniformLimit and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5795.1 and Definition 5795.2 seed the finite uniform-modulus ledger extracted from compact-metric monotone convergence data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniUniformLimitModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected limiting values, quotient function equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window visibility, finite-net extraction, modulus handoff, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniUniformLimitModulusUp.

Constructive story: A $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ packet is generated from compact-metric source rows, monotone continuous-family rows, pointwise windows, finite net rows, rational tolerance-budget rows, extracted uniform-modulus rows, UniformLimit handoff rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5796

A Concrete Naming Certificate for $\mathsf{SummationByParts}^{\uparrow}$

Following Abel and Dirichlet, the $\mathsf{SummationByParts}^{\uparrow}$ packet records the finite identity surface by which displayed partial sums, coefficient windows, adjacent differences, and endpoint terms are bound into one NameCert row. It sits beside the Abel-summation route of chapter 1763, the Abel-test consumer of chapter 1496, and the Dirichlet-test consumer of chapter 1497. Its source rows use the finite series carrier of chapter 121; its regular readback, dyadic tolerance, and endpoint seal use chapter 301, chapter 291, and chapter 38. The packet names a finite algebraic identity for downstream convergence tests; it does not import infinite sums, quotient series equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5796.1 (Summation by parts carrier)**.** *A* $\mathsf{SummationByParts}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Series}^{\uparrow}$ *source row, exposing only the inspected finite term window and its displayed partial sums.*
2. *A is the coefficient-window row, naming the coefficient entries read over the same finite interval.*
3. *Δ is the adjacent-difference row, recording the displayed values $a_k - a_{k+1}$ used by the identity.*
4. *P is the partial-sum-window row, naming the cumulative sums paired with A and Δ.*
5. *B is the boundary-term row, displaying the endpoint products and the finite interior products determined by S, A, Δ, P.*
6. *T is the convergence-test handoff row consumed by* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$.
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed finite identity data.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite additions, subtractions, products, and endpoint comparisons used by B.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after $S, A, \Delta, P, B, T, R, D$ have been displayed.*
10. *H records componentwise* hsame *transport for the source, coefficient, difference, partial-sum, boundary, handoff, readback, dyadic, endpoint, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay of the route*

$$S, A, \Delta, P, B, T, R, D \longmapsto E.$$

12. *Q is the* Pkg *provenance over* $\mathsf{SummationByParts}^{\uparrow}$, $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$. It has no coordinate for an infinite sum, quotient series equality, selected convergence witness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SummationByPartsUp.SummationByPartsTasteGate single carrier alignment`

Theorem 5796.2 (Summation by parts NameCert obligations)**.** *For every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *carrier*

$$K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed finite packet; source and partial-sum exposure through S and P; coefficient-window and adjacent-difference exposure through A and Δ; boundary-term exactness through B; convergence-test handoff through T; regular readback, dyadic accounting, and endpoint sealing through R, D, E; and transport, replay, provenance, and local naming through H, C, Q, N.*

Proof. Project the accepted carrier. The projection exposes exactly the rows $S, A, \Delta, P, B, T, R, D, E, H, C, Q, N$, so every naming obligation must be read from one of those displayed coordinates.

The identity rows are finite. The source row S and partial-sum row P display the inspected cumulative sums, the coefficient row A displays the paired coefficients, and Δ displays the adjacent differences used by the interior terms. The boundary row B records only the endpoint products and interior products determined by those rows.

The consumer handoff is downstream of the identity. A convergence-test consumer first reads T, and the regular and real-facing route then passes through R, D, E. These rows can transport the displayed finite calculation, but they do not introduce an independent summation principle.

Finally H, C, Q, N preserve, replay, source, and name the same route. Because the carrier has no coordinate for infinite sums, quotient series equality, selected convergence witnesses, countable choice, trichotomy, or ambient real completeness, the stated rows exhaust the local NameCert surface. □

Theorem 5796.3 (Summation by parts Cauchy-series handoff)**.** *Every accepted* $\mathsf{SummationByParts}^{\uparrow}$ *packet that is consumed by an* $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, *or* $\mathsf{DirichletTest}^{\uparrow}$ *route hands off only the finite identity chain*

$$\mathsf{Series}^{\uparrow} \longmapsto S, A, \Delta, P, B, T \longmapsto R, D, E, H, C, Q, N.$$

Thus a Cauchy-series or real-seal consumer may read the summation-by-parts row only after the series window, coefficient window, adjacent-difference ledger, partial-sum window, boundary ledger, and test handoff have been displayed. The handoff supplies no infinite sum, quotient series equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a convergence-test read. Carrier admission exposes the source row S and the finite identity rows A, Δ, P, B before the handoff row T is available.

The downstream chapters may add their own test-specific rows. The $\mathsf{AbelSummation}^{\uparrow}$ consumer reads the displayed finite identity as its transform row, the $\mathsf{AbelTest}^{\uparrow}$ consumer combines it with a monotone null coefficient ledger, and the $\mathsf{DirichletTest}^{\uparrow}$ consumer combines it with bounded partial sums and a null-tail schedule. In each case the summation-by-parts contribution remains the same displayed finite packet.

The Cauchy-facing route is therefore forced through R and D before E can seal the endpoint. Transport, replay, provenance, and local naming preserve that same route, so no excluded summation or completeness datum can be read from the packet. □

Falsifiable prediction: An accepted $\mathsf{SummationByParts}^{\uparrow}$ consumer that reaches an $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed source, coefficient, adjacent-difference, partial-sum, boundary-term, and handoff rows refutes the carrier surface.

Independence witness: abel-summation, abel-test, dirichlet-test, series, regseqrat, dyadicratcore, real: no listed sibling supplies the whole thirteen-row packet $K = (S, A, \Delta, P, B, T, R, D, E, H, C, Q, N)$, because each sibling contributes only a transform consumer, convergence-test consumer, source series, regular readback, dyadic ledger, or endpoint seal while this chapter binds the finite identity itself.

Closure status for $\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$.

Constructive story: A $\mathsf{SummationByParts}^{\uparrow}$ packet is generated from a SeriesUp finite source row, coefficient windows, adjacent-difference rows, partial-sum windows, boundary-term ledgers, AbelSummation/AbelTest/DirichletTest handoff rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, seed)

Scope closed: Definition 5796.1, Definition 5796.2, and Definition 5796.3 seed the finite summation-by-parts identity packet over $\mathsf{Series}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{AbelTest}^{\uparrow}$, $\mathsf{DirichletTest}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SummationByParts}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite summation, quotient series equality, selected convergence witnesses, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite identity exactness, convergence-test handoff, Cauchy-series readback, dyadic accounting, endpoint sealing, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.SummationByPartsUp.

Chapter 5797

A Concrete Naming Certificate for $\mathsf{RealLineOrder}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealLineOrder}^{\uparrow}$ packet records the constructive real-line order and apartness boundary visible to finite BEDC NameCert consumers. It sits over the dyadic tolerance ledger of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the rational Cauchy completion route of chapter 3670, and the Bishop regular Cauchy limit seal of chapter 1821. The packet gives L10 consumers a located comparison interface before any real-order algebra can read it; it does not import trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5797.1 (Real line order carrier)**.** *A* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (D, S, R, L, A, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the positive dyadic radius used by every finite comparison request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposing only the finite stream observations inspected at that tolerance.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream windows into regular rational comparison data.*
4. *L is the located comparison row, recording the finite verdict surface for $x < y$ as a witnessed positive gap at the displayed tolerance, an unresolved window requiring a smaller displayed tolerance, or a symmetric failure of the current request.*
5. *A is the apartness row, recording positive-gap readback for $x \# y$ only through D, S, R, L.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the same displayed stream and regular-rational data, not from host real equality.*
7. *H records componentwise* hsame *transport for dyadic tolerance, stream windows, regular-rational readback, located comparison, apartness, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite order request through $D, S, R, L, A, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{RealLineOrder}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, L, A, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderTasteGate single carrier alignment`

Theorem 5797.2 (Real line order NameCert obligations)**.** *For every accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier*

$$O = (D, S, R, L, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, stream-window row, regular-rational readback row, located comparison row, apartness row,* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport row, replay row, provenance row, and local naming row.*

The obligation surface requires finite tolerance visibility through D, window visibility through S, regular-rational comparison readback through R, located order exposure through L, apartness exposure through A, real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the accepted carrier of Definition 5797.1. The only analytic rows are the dyadic tolerance D, the finite stream windows S, the regular-rational readback R, the located comparison row L, the apartness row A, and the terminal real seal E.

The comparison step is located rather than trichotomic. A finite request may display a positive gap, request a smaller displayed tolerance, or fail symmetrically at the current window; none of these outcomes supplies a global three-way case split.

The equality-sensitive routes are absent. The readback row R is regular-rational data over the displayed windows, and the seal row E is reached only after D, S, R, L, A have been inspected. Thus the packet refuses quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded principles are not coordinates of O, no local NameCert obligation can demand them. □

Lean4 ✓ `BEDC.Derived.RealLineOrderUp.RealLineOrderNameCertObligations`

Theorem 5797.3 (Real line order apartness readback)**.** *Every apartness read exported by an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The apartness row A may expose only a positive-gap witness already visible through D, S, R, L. It exports no trichotomy datum, quotient equality, selected representative, host real equality, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Proof. Start with an apartness request. The request must first choose a displayed positive dyadic tolerance in D and inspect the finite windows in S.

The regular-rational readback R is then the only comparison source. The located comparison row L reads this finite data and either displays a positive gap at the requested tolerance or leaves the request unresolved for a smaller displayed tolerance.

The apartness row A can cite only the positive-gap branch already exposed by L. It refuses trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$ because those objects are not carrier rows.

The terminal seal E preserves the same finite route into $\mathsf{Real}^{\uparrow}$. Transport and replay through H and C keep the route stable, while P and N only record provenance and local naming. □

Theorem 5797.4 (Real line order L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{RealLineOrder}^{\uparrow}$ *carrier must read the dyadic tolerance, stream-window, regular-rational readback, located comparison, apartness, and* $\mathsf{Real}^{\uparrow}$ *seal rows in the order*

$$D, S, R, L, A, E.$$

Consequently the handoff supplies a finite non-quotient comparison boundary for later real algebra and order packets, while refusing trichotomy, quotient equality, selected representatives, host real equality, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. An L10 order request begins with finite tolerance data. The row D fixes the displayed dyadic radius, and S opens only the stream windows inspected at that radius.

The row R repacks those windows as regular-rational readback. The located comparison row L can then answer only with the finite comparison surface generated from D, S, R.

Apartness is downstream of located comparison. The row A may expose a positive gap only when L has already displayed one, and E seals the same finite route as a $\mathsf{Real}^{\uparrow}$-facing endpoint.

Because no earlier or later row stores trichotomy, quotient equality, selected representatives, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$, the L10 handoff cannot export those principles to later consumers. □

Falsifiable prediction: Every accepted $\mathsf{RealLineOrder}^{\uparrow}$ read must expose dyadic tolerance, StreamName windows, RegSeqRat readback, located comparison, apartness, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before an L10 order consumer can read a comparison boundary.

Independence witness: dyadicratcore, streamname, regseqrat, real, rational-cauchy-completion, bishop-regular-cauchy-limit-seal: $\mathsf{RealLineOrder}^{\uparrow}$ is not bijective to any listed sibling because it binds located comparison and apartness rows to the finite readback route before the RealUp seal is available.

Closure status for $\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5797.1, Definition 5797.2, Definition 5797.3, and Definition 5797.4 seed the finite non-quotient order and apartness boundary for displayed RealUp names.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLineOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient equality, selected representatives, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite tolerance visibility, stream-window visibility, regular-rational readback, located comparison, apartness readback, RealUp seal handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLineOrderUp.

Constructive story: A $\mathsf{RealLineOrder}^{\uparrow}$ packet is generated from a DyadicRatCoreUp tolerance row, StreamName finite windows, RegSeqRat readback, located comparison, apartness, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5798

A Concrete Naming Certificate for $\mathsf{SchwartzKernel}^{\uparrow}$

Following Schwartz and Gelfand–Shilov, the $\mathsf{SchwartzKernel}^{\uparrow}$ packet records the finite NameCert route by which a distribution kernel is read as a bilinear action between Schwartz test probes and tempered-distribution endpoints. It sits over the Schwartz-function carrier of chapter 3399, the tempered-distribution endpoint of chapter 2310, the nuclear factorization row of chapter 215, the projective tensor norm route of chapter 4088, the tensor-product source of chapter 87, and the real seal of chapter 38. The chapter exposes the kernel theorem as finite action, tensor, nuclearity, bound, transport, replay, provenance, and local naming rows; it does not import a host distribution space, quotient test-function equality, selected representatives, or an unrestricted topological-vector-space oracle.

Definition 5798.1 (Schwartz kernel carrier)**.** *A* $\mathsf{SchwartzKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, T, U, J, B, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-probe row naming the finite seminorm windows inspected on the source side.*
2. *T is the* $\mathsf{TemperedDistribution}^{\uparrow}$ *endpoint row naming the distributional action read after the test windows are displayed.*
3. *U is the* $\mathsf{TensorProduct}^{\uparrow}$ *row naming the finite tensor source on which the two test-probe sides are paired.*
4. *J is the* ***??*** *row recording the projective-tensor bound used to compare the paired test windows.*
5. *B is the* $\mathsf{Nuclear}^{\uparrow}$ *factorization row, exposing the finite nuclear decomposition route by which the kernel action is read.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed scalar bounds and kernel-action readbacks.*
7. *H records componentwise* hsame *transport for test probes, tempered endpoint, tensor source, projective bound, nuclear factorization, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite kernel request through $S, T, U, J, B, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{SchwartzKernel}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{TemperedDistribution}^{\uparrow}$, $\mathsf{Nuclear}^{\uparrow}$, ***??***, $\mathsf{TensorProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $S, T, U, J, B, R, H, C, P, N$. It has no coordinate for a host distribution space, quotient equality of test functions, selected representatives, ambient bornological completeness, or a topological-vector-space theorem outside the displayed rows.

Lean4 ✓ `BEDC.Derived.SchwartzKernelUp.SchwartzKernelTasteGate single carrier alignment`

Theorem 5798.2 (Schwartz kernel NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *carrier*

$$K = (S, T, U, J, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^\uparrow}$ *surface consists exactly of Schwartz test-probe admission through S, tempered-distribution endpoint admission through T, tensor-source admission through U, projective-bound exposure through J, nuclear-factorization exposure through B, scalar RealUp sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every kernel consumer must read the distributional action through these ten rows. No consumer may replace them by an ambient kernel theorem, quotient test-function equality, selected representative, hidden topological vector space, or host nuclear-space completion.

Proof. Project the accepted carrier of Definition 5798.1. The analytic rows are the Schwartz test row S, the tempered-distribution endpoint T, the tensor source U, the projective bound J, the nuclear factorization B, and the RealUp seal R.

The action route is finite. The tensor row U supplies the paired test-probe source, J records the displayed projective bound, and B records the nuclear factorization by which the kernel action becomes readable by a finite consumer.

Transport and replay do not create new analytic structure. The rows H and C preserve the displayed packet componentwise and replay only the same kernel request. The provenance and naming rows P, N package the route without adding host quotient equality or a hidden completion theorem.

Thus the listed rows are exactly the local NameCert obligations for $\mathsf{SchwartzKernel}^\uparrow$. □

Theorem 5798.3 (Schwartz kernel nuclear factorization boundary)**.** *Every accepted* $\mathsf{SchwartzKernel}^\uparrow$ *read factors through the finite route*

$$\mathsf{SchwartzFunction}^\uparrow,\ \mathsf{TensorProduct}^\uparrow,\ ??,\ \mathsf{Nuclear}^\uparrow \longmapsto \mathsf{TemperedDistribution}^\uparrow \longmapsto \mathsf{Real}^\uparrow.$$

The nuclear row may expose only the displayed factorization and bound data already routed through S, U, J. It exports no host theorem identifying all kernels with all continuous bilinear forms, no quotient equality of test probes, no selected representative of a distribution, and no ambient completeness claim for the surrounding function spaces.

Proof. Start with a kernel read. The request first displays the finite Schwartz windows in S and the tensor source U on which the two test-probe sides are paired.

The projective tensor bound J is the only bound row available to the consumer. The nuclear factorization row B can cite only this displayed tensor-bound route, so its readback is a finite factorization certificate rather than an ambient nuclear-space theorem.

The tempered-distribution endpoint T then reads the resulting action, and R seals the scalar outputs as RealUp data. Transport through H and replay through C keep the same row order, while P and N record source and local naming.

Since no carrier coordinate stores host quotient equality, selected representatives, or ambient completeness, those principles cannot be exported by the factorization boundary. □

Falsifiable prediction: Every accepted $\mathsf{SchwartzKernel}^{\uparrow}$ packet must expose SchwartzFunctionUp test probes, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, TemperedDistributionUp endpoint action, RealUp scalar sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before a kernel consumer can read the action.

Independence witness: schwartz-function, tempered-distribution, nuclear, projective-tensor-norm, tensorproduct, real: $\mathsf{SchwartzKernel}^{\uparrow}$ is not bijective to any listed sibling because it binds the test-probe tensor route, nuclear factorization boundary, distribution endpoint, and scalar seal as one finite kernel-action packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{SchwartzKernel}^{\uparrow}$ packet is generated from SchwartzFunctionUp test probes, a TemperedDistributionUp endpoint, TensorProductUp source data, ProjectiveTensorNormUp bounds, NuclearUp factorization, RealUp scalar seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5798.1, Definition 5798.2, Definition 5798.3, the falsifiable prediction, and the independence witness seed the finite kernel-action route over Schwartz, tempered-distribution, tensor, nuclear, bound, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SchwartzKernel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host distribution-space semantics, quotient equality of test functions, selected distribution representatives, ambient nuclear-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, test-probe visibility, tensor-source admission, projective-bound exactness, nuclear-factorization exposure, tempered-endpoint action, RealUp scalar sealing, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SchwartzKernelUp.

Chapter 5799

A Concrete Naming Certificate for $\mathsf{SynonymousTransition}^{\uparrow}$

The $\mathsf{SynonymousTransition}^{\uparrow}$ packet realizes the finite synonymous-transition geometry of Definition 9462.1186 as BEDC NameCert data. It sits beside the six-bit codon-window carrier of chapter 2234, the codon transition matrix of chapter 2245, the DNA boundary-tile packet of chapter 3393, the window-coverage trigger algebra of chapter 3478, and the DNA-window trigger threshold of chapter 4217. The packet names finite codon-window, class-adjacency, boundary-spectrum, threshold, transport, replay, provenance, and local naming rows; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Definition 5799.1 (Synonymous transition carrier)**.** *A* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (W, M, A, B, T, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{CodonSixBitWindow}^{\uparrow}$ *row naming the finite Q_6 codon-window surface.*
2. *M is the* $\mathsf{CodonTransitionMatrix}^{\uparrow}$ *row naming one-bit codon transitions inspected by the packet.*
3. *A is the class-adjacency row recording the displayed synonymous and cross-class adjacency counts, including the largest weights of Definition 9462.1186.*
4. *B is the boundary-spectrum row recording internal edge counts, external boundary degrees, and the control-pair corner-contact data.*
5. *T is the* $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$ *row naming the finite threshold at which an adjacency or boundary-spectrum read may trigger a downstream tile consumer.*
6. *H records componentwise* hsame *transport for codon windows, transition matrix, class adjacency, boundary spectrum, trigger threshold, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite synonymous-transition request through W, M, A, B, T, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{SynonymousTransition}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{CodonTransitionMatrix}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, M, A, B, T, H, C, P, N$. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden enumeration, or an ambient graph-theory oracle outside the displayed finite rows.

Theorem 5799.2 (Synonymous transition NameCert obligations)**.** *For every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *carrier*

$$G = (W, M, A, B, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$ *surface consists exactly of codon-window admission through W, transition-matrix admission through M, class-adjacency exposure through A, boundary-spectrum exposure through B, trigger-threshold exposure through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every downstream synonymous-transition consumer must read the graph through these nine rows. It cannot replace them by biological causation, stochastic selection, quotient codon semantics, an unrecorded enumeration, or a hidden graph oracle.

Proof. Project the accepted carrier of Definition 5799.1. The finite codon surface is W, the one-bit transition source is M, and the named graph data are the class-adjacency row A and boundary-spectrum row B.

The trigger row T records when the displayed adjacency or boundary-spectrum data may be read by a downstream tile or trigger consumer. It does not add a biological mechanism or a stochastic model; it only names the finite threshold over the same graph rows.

Transport and replay are componentwise. The rows H and C preserve and replay W, M, A, B, T with the same finite endpoints, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient codon equality, hidden enumeration, biological causation, or ambient graph semantics, the listed rows exhaust the local NameCert obligations. □

Theorem 5799.3 (Synonymous transition boundary-spectrum handoff)**.** *Every accepted* $\mathsf{SynonymousTransition}^{\uparrow}$ *read factors through the finite route*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{CodonTransitionMatrix}^{\uparrow} \longmapsto \mathsf{SynonymousTransition}^{\uparrow} \longmapsto \mathsf{DnaWindowSixBoundaryTile}^{\uparrow},\ \mathsf{Windo}$$

The handoff exposes class-adjacency weights, internal-edge counts, external boundary degrees, and control-pair corner contacts only as finite packet rows. It exports no biological mechanism, stochastic fitness model, quotient codon semantics, or unrecorded enumeration oracle.

Proof. Start with the codon-window row W and the one-bit transition matrix row M. These rows supply the finite Q_6 source and the adjacency relation inspected by the packet.

The class-adjacency row A records the largest displayed weights, and B records the boundary-spectrum data. The downstream boundary-tile and trigger-algebra consumers therefore receive a finite graph packet rather than a biological or stochastic explanation.

The threshold row T names when the finite rows can be consumed by the trigger route. Transport through H and replay through C keep the same graph rows visible, while P and N only package their provenance and local name.

Thus every public read of the synonymous-transition geometry remains inside the finite codon-window, transition, adjacency, boundary-spectrum, threshold, transport, replay, provenance, and naming surface. □

Falsifiable prediction: Every accepted $\mathsf{SynonymousTransition}^{\uparrow}$ packet must expose CodonSixBitWindowUp, CodonTransitionMatrixUp, class-adjacency, boundary-spectrum, DNAWindowSixTriggerThresholdUp, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ rows before a downstream tile or trigger consumer can read the synonymous-transition graph.

Independence witness: codon-six-bit-window, codon-transition-matrix, dna-window-six-boundary-tile, window-coverage-trigger-algebra, dna-window-six-trigger-threshold: $\mathsf{SynonymousTransition}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries class-adjacency weights, boundary-spectrum rows, and trigger-threshold handoff over the finite codon-window graph.

Closure status for $\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}$.

Constructive story: A $\mathsf{SynonymousTransition}^{\uparrow}$ packet is generated from a CodonSixBitWindowUp source, a CodonTransitionMatrixUp one-bit transition row, class-adjacency counts, boundary-spectrum data, DNAWindowSixTriggerThresholdUp rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5799.1, Definition 5799.2, Definition 5799.3, the falsifiable prediction, and the independence witness seed the finite synonymous-transition packet over codon-window, transition, adjacency, boundary-spectrum, threshold, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SynonymousTransition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic fitness, quotient codon semantics, hidden enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, codon-window visibility, transition-matrix exactness, class-adjacency exposure, boundary-spectrum exposure, trigger-threshold handoff, componentwise transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.SynonymousTransitionUp.

Chapter 5800

A Concrete Naming Certificate for $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$

Following Feller and Billingsley on finite-sum normal approximation, the $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet records the finite BEDC surface by which an iid random-variable window is centered, scaled, read as a distribution row, and handed to a real-normal comparison seal. It sits after the probability source of chapter 184, the measurable random-variable rows of chapter 185, the law readback of chapter 186, the regular rational window of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite central-limit-facing certificate; it is not an asymptotic central limit theorem, a characteristic-function convergence theorem, a host probability-limit operator, countable choice, quotient distribution equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5800.1 (Central-limit finite-window carrier)**.** *A* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$(P, V, S, K, D, R, E, H, C, Q, N).$$

Here P is the source $\mathsf{ProbSpace}^{\uparrow}$ *row carrying the event classifier and total-mass normalization. The row V is the displayed iid* $\mathsf{RandomVar}^{\uparrow}$ *window over P, including the finite index range and the shared law token read by the window. The row S is the finite partial-sum ledger naming the displayed sum, the sample count, and the centered numerator. The row K is the centering and scaling row, recording the finite mean token, variance or spread token, positive scale request, and normalized finite-sum expression. The row D is the* $\mathsf{Distribution}^{\uparrow}$ *readback for the normalized finite sum and its comparison request. The row R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *window that presents rational test intervals, tolerances, and finite comparison samples for the distribution read. The row E is the* $\mathsf{Real}^{\uparrow}$ *normal-window seal naming the terminal real comparison against the displayed normal row at the requested finite window. The row H records componentwise* hsame *transport for the probability source, iid window, partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat window, Real normal-window seal, continuation, provenance, and local naming rows. The row C records* Cont *replay from a normal-facing read through P, V, S, K, D, R, E and then through the structural rows. The row Q is the* Pkg *provenance over* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*. The row N is the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *row. The classifier compares accepted carriers only through these eleven rows.*

Lean4 ✓ `BEDC.Derived.CentralLimitFiniteWindowUp.TasteGate.CentralLimitFiniteWindowTasteGate single carrier alignment`

Theorem 5800.2 (Central-limit finite-window NameCert obligations)**.** *For every accepted* CentralLimitFiniteWindo *carrier* $(P, V, S, K, D, R, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$ *surface consists exactly of the finite rows in Definition 5800.1. Carrier admission displays the probability source, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, transport, replay, provenance, and local naming row. The certificate imports no host asymptotic central limit theorem, characteristic-function limit, Lindeberg replacement principle, countable-choice schedule, quotient distribution equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Project the carrier rows in order. The probability source P and iid random-variable window V are the only stochastic data exposed to the certificate, and the partial-sum ledger S fixes the finite sum before any normal-facing read is visible.

The centering and scaling row K then determines the normalized finite expression. The distribution row D and RegSeqRat window R provide the only comparison surface through which a finite law read can reach the terminal $\mathsf{Real}^{\uparrow}$ normal-window seal E.

Finally H, C, Q, N transport, replay, source, and name the same displayed route. Since the carrier has no coordinate for an asymptotic theorem, characteristic-function limit, quotient law equality, countable choice, or ambient completeness, the listed rows exhaust the local finite NameCert surface. □

Theorem 5800.3 (Central-limit finite-window normal handoff)**.** *Every normal-facing read of an accepted* $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ *carrier factors through the finite route*

$$P \longmapsto V \longmapsto S \longmapsto K \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, Q, N$$

before any consumer receives a normal-window comparison. Thus the read is the displayed iid finite-sum, normalization, distribution, RegSeqRat, and RealUp handoff itself; it exports no host asymptotic CLT, countable choice, quotient distribution equality, characteristic-function convergence principle, or ambient real completeness.

Proof. Begin with a normal-facing request. The only probability data available are the source row P and the finite iid window V, so the request must first read the displayed sample range and shared law token.

The partial-sum row S and normalization row K are then forced. They provide the finite sum, centering token, scaling token, and normalized expression that the downstream distribution read consumes.

The request can reach a normal-window comparison only through D, R, E. The distribution row D names the finite law read, the RegSeqRat row R exposes rational windows and tolerances, and the RealUp row E seals the displayed normal comparison. The structural rows H, C, Q, N preserve that route without adding analytic content.

Therefore every accepted read factors through the finite iid-sum, normalization, distribution, RegSeqRat, and RealUp rows. The excluded asymptotic, choice, quotient, characteristic-function, and completeness principles cannot be inserted into the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CentralLimitFiniteWindow}^{\uparrow}$ packet is generated from a probability source row, iid random-variable window, finite partial-sum ledger, centering and scaling row, distribution readback, RegSeqRat comparison window, RealUp normal-window seal, componentwise hsame transport, continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CentralLimitFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5800.1, Definition 5800.2, and Definition 5800.3 seed the finite central-limit-facing handoff through ProbSpace, RandomVar, Distribution, RegSeqRat, RealUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{CentralLimitFiniteWindow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close the asymptotic central limit theorem, characteristic-function convergence, Lindeberg replacement, countable-choice schedules, quotient distribution equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, iid-window routing, partial-sum ledger exactness, normalization stability, distribution-to-RegSeqRat handoff, RealUp normal-window nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.CentralLimitFiniteWindowUp.

Chapter 5801

A Concrete Naming Certificate for $\mathsf{LipschitzRealMap}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LipschitzRealMap}^{\uparrow}$ packet records a constructive real-map certificate whose continuity is controlled by a displayed dyadic Lipschitz constant and finite regular-rational windows. It sits after the dyadic tolerance ledger of chapter 291, the countable-name window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the located order and apartness neighbour of chapter 5797. The packet names the finite L10 route by which later continuous-map and metric-uniform-continuity consumers may read a real-valued map without importing host function extensionality, quotient representatives, selected limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5801.1 (Lipschitz real-map carrier)**.** *A* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, K, W, R, M, H, C, P, N)$$

with the following rows:

1. *S is the source* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real input surface through displayed* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *T is the target* $\mathsf{Real}^{\uparrow}$ *seal row naming the accepted real output surface through the same kind of finite readback.*
3. *K is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *Lipschitz constant row, including the positive dyadic bound and the displayed tolerance ledger used by every modulus request.*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for the source and target names inspected by the request.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the displayed source and target windows into rational comparison data.*
6. *M is the modulus-inequality row, recording that every accepted finite pair of source reads with displayed apartness or distance bound is sent to a target read whose displayed difference is bounded by the dyadic constant in K times the source bound, up to the requested displayed tolerance.*
7. *H records componentwise* hsame *transport for source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a finite map request through $S, T, K, W, R, M, H, P, N$.*
9. *P is the* Pkg *provenance over* LipschitzRealMap$^{\uparrow}$, Real$^{\uparrow}$, RealLineOrder$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, T, K, W, R, M, H, C, P, N$. It has no coordinate for host function equality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LipschitzRealMapUp.LipschitzRealMapTasteGate single carrier alignment`

Theorem 5801.2 (Lipschitz real-map NameCert obligations)**.** *For every accepted* LipschitzRealMap$^{\uparrow}$ *carrier*

$$F = (S, T, K, W, R, M, H, C, P, N),$$

the local NameCert$_{\text{LipschitzRealMap}^{\uparrow}}$ *surface consists exactly of carrier admission for the source* Real$^{\uparrow}$ *seal, target* Real$^{\uparrow}$ *seal, dyadic Lipschitz constant, finite StreamName windows, RegSeqRat readback, modulus inequality, componentwise transport, replay, provenance, and local naming rows.*

The obligation surface requires finite source and target visibility through S and T, dyadic bound visibility through K, finite window visibility through W, rational readback through R, modulus-inequality exposure through M, transport and replay stability through H and C, provenance through P, and local naming through N. It explicitly refuses host function extensionality, quotient equality of real presentations, selected representatives, global uniform-continuity principles, and ambient completeness of Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5801.1. The source and target real seals are the only real-name endpoints, and both are read through finite StreamName and RegSeqRat windows rather than through host real equality.

The Lipschitz datum is the dyadic row K. A map request can inspect only the finite windows W and their rational readback R before the modulus row M exposes the displayed inequality for the requested tolerance.

The structural rows do not add analytic content. The transport row H preserves the already displayed source, target, constant, window, readback, and modulus rows componentwise, while C replays the same route and P, N record provenance and local naming.

Since no carrier coordinate stores host function equality, quotient real equality, selected representatives, global uniform continuity, or ambient completeness, no local NameCert obligation can require those objects. □

Theorem 5801.3 (Lipschitz real-map modulus stability)**.** *Let F_0 and F_1 be accepted* LipschitzRealMap$^{\uparrow}$ *carriers whose source seals, target seals, dyadic Lipschitz constant rows, finite StreamName windows, and RegSeqRat readback rows are componentwise* hsame*-transported. Then every displayed modulus-inequality read exported by F_0 transports to the corresponding displayed modulus-inequality read exported by F_1.*

The stability claim is confined to the finite rows $S, T, K, W, R, M, H, C, P, N$. It does not identify host functions, quotient real presentations, selected Cauchy names, or ambient complete metric structures.

Proof. Start with a displayed modulus request for F_0. Its source and target real seals are read only through the finite StreamName windows and RegSeqRat readback rows named by S, T, W, R.

Use the componentwise hsame hypotheses. They transport the dyadic constant row K and the finite window/readback rows into the corresponding rows of F_1, so the same displayed source tolerance and target tolerance are available after transport.

The modulus row M is a row of the carrier rather than a host theorem about all representatives. Therefore the transported request reads the F_1 modulus row with the same displayed dyadic bound and the transported finite readback data.

Finally C, P, N replay, source, and name the transported route. No quotient equality, host function equality, selected representative, or ambient completeness principle is introduced by this transport. □

Theorem 5801.4 (Lipschitz real-map L10 handoff)**.** *Every L10 consumer of an accepted* $\mathsf{LipschitzRealMap}^{\uparrow}$ *carrier must read the route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RealLineOrder}^{\uparrow} \longmapsto \mathsf{LipschitzRealMap}^{\uparrow}$$

through the finite source seal, target seal, dyadic constant, finite windows, regular-rational readback, modulus inequality, transport, replay, provenance, and local NameCert rows. The handoff supplies a finite Lipschitz-control surface for later continuous-map and metric-uniform-continuity packets, while refusing host function extensionality, quotient equality, selected representatives, global uniform-continuity principles, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. An L10 map request begins with source and target real-name seals. By the carrier definition, these seals are visible only through the finite StreamName windows and RegSeqRat readback rows used by S, T, W, R.

The request then reads the dyadic constant row K and the located comparison surface supplied by the real-line order neighbour. These rows provide the displayed source difference and target tolerance against which the modulus inequality is checked.

The modulus row M is the only analytic output of the packet. It hands a consumer a finite inequality route, not a host-level function theorem.

The rows H, C, P, N transport, replay, source, and name the same route. Thus later continuous-map and metric-uniform-continuity packets can consume the finite Lipschitz surface without receiving quotient equality, selected representatives, host function extensionality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzRealMap}^{\uparrow}$ packet is generated from source and target RealUp seal rows, a DyadicRatCore Lipschitz constant, finite StreamName windows, RegSeqRat readback, a displayed modulus-inequality row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, seed)

Scope closed: Definition 5801.1, Definition 5801.2, Definition 5801.3, and Definition 5801.4 seed the finite Lipschitz real-map surface over the current L10 real and completion route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LipschitzRealMap}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host function extensionality, quotient equality of real presentations, selected Cauchy representatives, global uniform continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source and target RealUp sealing, dyadic constant visibility, finite window readback, modulus-inequality stability, transport, replay, provenance, and local naming for future BEDC.Derived.LipschitzRealMapUp.

Chapter 5802

A Concrete Naming Certificate for HyperbolicQuasiGeodesicShadow$^{\uparrow}$

The HyperbolicQuasiGeodesicShadow$^{\uparrow}$ packet realizes the observer-expansion shadow route of section 9463.4 as finite quasi-geodesic, boundary, visual-metric, Busemann, and lamination-compatible rows. It sits after disk readout through chapter 377, the metric surface of chapter 3474, the visual-metric packet of chapter 3866, the Busemann cocycle route of chapter 4134, the geodesic-lamination packet of chapter 4203, and the observer-expansion packet of chapter 4167. The packet names a finite shadow tied to a displayed quasi-geodesic route; it does not complete the boundary, choose a global geodesic representative, identify quotient ideal points, or import host hyperbolic geometry.

Definition 5802.1 (Hyperbolic quasi-geodesic shadow carrier)**.** *A* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, G, E, V, B, L, H, C, P, A, N)$$

with the following rows:

1. *D is the disk readout row, exposing only displayed* UnitDisk$^{\uparrow}$ *coordinates and the associated* HyperbolicMetric$^{\uparrow}$ *metric-read data.*
2. *G is the quasi-geodesic route row, recording the finite path samples, distortion budget, endpoint-facing direction, and local route ledger inspected by the shadow request.*
3. *E is the boundary endpoint row named by the displayed route, not by a completed ideal-boundary quotient.*
4. *V is the visual-metric shadow row, recording the finite shadow window read through* HyperbolicVisualMetric$^{\uparrow}$ *and the boundary endpoint E.*
5. *B is the Busemann cocycle tolerance row, recording the displayed additive tolerance used to compare the route and the shadow window.*
6. *L is the geodesic-lamination compatibility row, recording whether the displayed route is compatible with the finite noncrossing leaf data consumed by* HyperbolicGeodesicLamination$^{\uparrow}$.
7. *H records componentwise* hsame *transport for disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, replay, provenance, refusal, and local naming rows.*
8. *C records displayed* Cont *replay from an observer-expansion shadow request through $D, G, E, V, B, L, H, P, A, N$.*

9. *P is the* Pkg *provenance over* HyperbolicQuasiGeodesicShadow$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HyperbolicGeodesicLamination$^{\uparrow}$, HyperbolicObserverExpar BHist, hsame, Cont, *and* NameCert.
10. *A is the refusal row excluding completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, and host curvature semantics.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, G, E, V, B, L, H, C, P, A, N$.

Theorem 5802.2 (Hyperbolic quasi-geodesic shadow NameCert obligations)**.** *For every accepted* HyperbolicQuasiGeodesicShadow$^{\uparrow}$ *carrier*

$$Q = (D, G, E, V, B, L, H, C, P, A, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$ *surface consists exactly of disk-readout visibility, quasi-geodesic route visibility, boundary endpoint exposure, visual-metric shadow-window exposure, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise transport, replay, provenance, refusal, and local naming rows.*

The packet exports no completed ideal boundary, quotient boundary equality, selected global geodesic, ambient hyperbolic compactification, physical observer trajectory, host curvature theorem, or global field semantics.

Proof. Project the carrier rows in order. The disk readout D supplies the only coordinate and metric source, while G supplies the displayed route samples and distortion budget inspected by the request.

The endpoint and shadow rows are downstream of that route. The endpoint row E names only the boundary read visible from G, and the visual row V records the finite shadow window tied to that endpoint.

The comparison rows B and L keep the shadow route finite. The Busemann tolerance is a displayed cocycle budget, and lamination compatibility is a finite noncrossing-leaf check, not a global geodesic-foliation theorem.

Finally H, C, P, A, N transport, replay, source, refuse, and name the same packet. Since the excluded host objects are not carrier rows, they cannot appear as NameCert obligations. □

Theorem 5802.3 (Hyperbolic quasi-geodesic shadow stability)**.** *Let Q_0 and Q_1 be accepted* HyperbolicQuasiGeodesi *carriers whose disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow window, Busemann tolerance, lamination compatibility, refusal, provenance, and local naming rows are componentwise* hsame*-transported. Then every observer-expansion consumer reads the same finite shadow surface through Q_0 and Q_1.*

The stability claim is confined to the finite rows $D, G, E, V, B, L, H, C, P, A, N$. It does not identify completed boundaries, quotient ideal points, selected geodesics, ambient compactifications, physical observer trajectories, host curvature data, or global fields.

Proof. Start with an observer-expansion shadow request for Q_0. The request must read disk coordinates and metric data through D and then the finite route samples and distortion budget through G.

Use the componentwise hsame transport. It carries the endpoint E, visual shadow window V, Busemann tolerance B, and lamination compatibility row L to the corresponding rows of Q_1.

The refusal row is transported with the same packet. Therefore the transported consumer sees the same excluded host objects and cannot insert completed boundary quotients, selected global geodesics, ambient compactifications, physical trajectories, curvature theorems, or global fields.

The replay row C and the provenance and naming rows P, N then present the transported shadow as the same finite observer-expansion surface. □

Theorem 5802.4 (Hyperbolic quasi-geodesic shadow vision handoff)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ *carrier must read the route*

$\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicGeodesicLaminatio}$

through disk readout, quasi-geodesic route, boundary endpoint, visual-metric shadow, Busemann tolerance, lamination compatibility, transport, replay, provenance, refusal, and local NameCert rows. The handoff realizes the vision-level shadow route as finite NameCert data without turning observer expansion into physical cosmology or a host hyperbolic-boundary theorem.

Proof. The vision handoff begins with disk and metric data. Rows D and G ensure that the consumer reads only displayed disk observations, metric readouts, finite route samples, and a distortion budget.

The endpoint and shadow window are then forced. Rows E and V bind the boundary endpoint and visual-metric shadow to the same route, while B records the Busemann tolerance needed to compare route and shadow.

Lamination compatibility is read through L. This makes the packet consumable by geodesic-lamination and observer-expansion rows without selecting a global geodesic representative or completing the boundary.

The rows H, C, P, A, N transport, replay, source, refuse, and name the handoff. Thus the vision-level quasi-geodesic shadow is realized as finite NameCert data and not as ambient hyperbolic geometry. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ read must expose a boundary endpoint, visual-metric shadow window, Busemann cocycle tolerance, quasi-geodesic route, transport row, provenance row, refusal row, and local NameCert before any observer-expansion consumer may read the shadow.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-cocycle, hyperbolic-geodesic-lamination, hyperbolic-observer-expansion: $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ is not bijective to these siblings because it binds the quasi-geodesic route and shadow window together before the observer-expansion packet may consume the boundary read.

The Lean TasteGate for this packet must later provide Nontrivial $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ and FieldFaithful $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ in `lean4/BEDC/Derived/HyperbolicQuasiGeodesicShadowUp/` Those instances must keep $D, G, E, V, B, L, H, C, P, A, N$ as separate fields before any checked carrier-alignment theorem can serve as a public TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}$ packet is generated from disk readout rows, a quasi-geodesic route, a boundary endpoint, a visual-metric shadow window, Busemann cocycle tolerance, geodesic-lamination compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, refusal, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5802.1, Definition 5802.2, Definition 5802.3, and Definition 5802.4 seed the finite quasi-geodesic shadow route for observer-expansion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicQuasiGeodesicShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed boundary quotients, selected global geodesics, ambient hyperbolic compactification, physical observer trajectories, host curvature semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk readout, quasi-geodesic route admission, endpoint exposure, visual-metric shadow windows, Busemann tolerance, lamination compatibility, TasteGate nontriviality, field faithfulness, refusal exactness, transport, replay, provenance, and local naming for future BEDC.Derived.HyperbolicQuasiGeodesicShadowUp.

Chapter 5803

A Concrete Naming Certificate for $\mathsf{Dendrite}^{\uparrow}$

Following Nadler and Kuratowski on continuum theory, the $\mathsf{Dendrite}^{\uparrow}$ packet records the finite BEDC surface for a locally connected compact metric continuum whose tree-like form is read through unique arcs and cutpoint separation. It sits over the compact metric carrier of chapter 129, the locally connected surface of chapter 5809, the Peano continuum boundary of chapter 2160, and the tree certificate of chapter 119. The packet names displayed endpoint, unique-arc, and cutpoint-separation rows; it does not import host continua, arbitrary path spaces, open-cover compactness, or quotient equality of arcs.

Definition 5803.1 (Dendrite carrier)**.** *A* $\mathsf{Dendrite}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric distance rows, finite ProbeBundle-net rows, complete-limit witness rows, and rational tolerance comparisons already certified by the compact-metric chapter.*
2. *L is the* $\mathsf{LocallyConnected}^{\uparrow}$ *row, recording the displayed connected neighbourhood refinements through which local arc and cutpoint requests are inspected.*
3. *P is the* $\mathsf{PeanoContinuum}^{\uparrow}$ *row, binding the compact metric and locally connected continuum surfaces before any dendrite consumer reads an arc boundary.*
4. *T is the* $\mathsf{Tree}^{\uparrow}$ *row, recording the finite tree-like separation surface used to rule out cycle-shaped routes.*
5. *E is the endpoint ledger, listing endpoint rows and their neighbourhood-read witnesses.*
6. *A is the unique-arc ledger, recording for every displayed endpoint or carried point pair the finite arc row, its endpoint membership rows, and its non-branching comparison rows.*
7. *C is the cutpoint-separation ledger, recording the displayed separation rows used when removing a carried point disconnects the relevant finite tree-like boundary.*
8. *H records componentwise* hsame *transport for compact metric data, local connectedness data, Peano-continuum data, tree data, endpoint rows, unique-arc rows, cutpoint-separation rows, replay, provenance, and local naming.*

9. *R records displayed* Cont *replay from a dendrite request through* $K, L, P, T, E, A, C, H, G, Q, N$.
10. *G is the* Pkg *provenance over* $\mathsf{Dendrite}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{PeanoContinuum}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *Q records the local source and consumer provenance for endpoint, unique-arc, and cutpoint-separation reads.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host continuum, arbitrary path selection, open-cover compactness, cycle elimination by excluded middle, or quotient equality of arcs.

Lean4 ✓ `BEDC.Derived.DendriteUp.DendriteTasteGate single carrier alignment`

Theorem 5803.2 (Dendrite NameCert obligations)**.** *For every accepted* $\mathsf{Dendrite}^{\uparrow}$ *carrier*

$$D = (K, L, P, T, E, A, C, H, R, G, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{Dendrite}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, locally connected row, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming;*
2. *compact-continuum visibility, requiring every dendrite consumer to read compact metric and local connectedness data through K, L, P before reading endpoint or arc rows;*
3. *tree-like exactness, requiring the non-cycle and separation information to pass through T and the cutpoint-separation ledger C;*
4. *unique-arc exactness, requiring every arc-facing read to use the displayed ledger A with its endpoint membership and non-branching comparison rows;*
5. *endpoint exactness, requiring endpoint reads to use only the displayed endpoint ledger E and not a host boundary predicate;*
6. hsame, Cont, *and* Pkg *stability, requiring componentwise transport, displayed replay, and provenance to preserve precisely the listed rows;*
7. *non-escape, excluding host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, and quotient equality of arcs.*

No dendrite obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5803.1. The accepted tuple displays exactly the compact metric source, local connectedness, Peano-continuum row, tree row, endpoint ledger, unique-arc ledger, cutpoint-separation ledger, transport, replay, provenance, source provenance, and local naming.

The compact-continuum route is K, L, P. The compact metric source supplies distance, finite-net, complete-limit, and tolerance-comparison rows; the locally connected row supplies connected neighbourhood refinements; and the Peano-continuum row binds those surfaces before endpoint or arc data are exposed.

The tree-like route is T, E, A, C. The tree row supplies the finite non-cycle surface, the endpoint ledger records boundary reads, the unique-arc ledger records the displayed arc row between

carried point pairs, and the cutpoint-separation ledger records the separation witnesses used by downstream consumers.

Finally, H, R, G, Q, N transport, replay, source, and name the same finite packet. Since the carrier has no row for a host continuum, arbitrary path space, open-cover compactness principle, selected arc representative, or quotient equality of arcs, the listed obligations exhaust the local $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$ surface. □

Theorem 5803.3 (Dendrite unique-arc handoff)**.** *Every unique-arc or cutpoint-facing consumer of an accepted* $\mathsf{Dendrite}^\uparrow$ *carrier must read the finite route*

$$\mathsf{CompactMetric}^\uparrow,\ \mathsf{LocallyConnected}^\uparrow,\ \mathsf{PeanoContinuum}^\uparrow,\ \mathsf{Tree}^\uparrow \longmapsto E, A, C, H, R, G, Q, N.$$

The handoff exposes endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row. It does not expose arbitrary host paths, open-cover compactness, selected arc representatives, quotient equality of arcs, or a continuum object outside the displayed packet.*

Proof. Start with the compact-continuum source. The rows K, L, P force a consumer to read the compact metric, locally connected, and Peano-continuum surfaces before any arc-facing data can be inspected.

Next read the tree row T. This is the only row carrying the tree-like non-cycle surface, so a later cutpoint or arc argument cannot replace it by a host graph or a classical continuum theorem.

The endpoint and arc data are then E, A, C. Endpoint reads pass through E, unique-arc reads pass through A, and cutpoint-separation reads pass through C; the three ledgers remain tied to the same compact-continuum and tree source rows.

The structural rows H, R, G, Q, N preserve this route under transport, replay, provenance, and local naming. Because no excluded host-side row appears in the carrier, the consumer receives exactly the displayed unique-arc handoff and nothing stronger. □

Falsifiable prediction: Every accepted $\mathsf{Dendrite}^\uparrow$ read must expose CompactMetric, LocallyConnected, PeanoContinuum, Tree, endpoint, unique-arc, cutpoint-separation, componentwise hsame, displayed Cont, Pkg, source provenance, and local NameCert rows before a continuum or arc consumer can inspect the dendrite boundary.

Independence witness: compactmetric, locally-connected, peano-continuum, tree: $\mathsf{Dendrite}^\uparrow$ is not bijective to any listed sibling because it binds endpoint, unique-arc, and cutpoint-separation ledgers over their common compact-continuum and tree-like source.

Closure status for $\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$**.**

Constructive story: A $\mathsf{Dendrite}^\uparrow$ packet is generated from CompactMetricUp, LocallyConnectedUp, PeanoContinuumUp, TreeUp, endpoint rows, unique-arc rows, cutpoint-separation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, source provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, seed)

Scope closed: Definition 5803.1, Definition 5803.2, and Definition 5803.3 seed the finite dendrite certificate over compact metric, locally connected, Peano-continuum, tree, endpoint, unique-arc, cutpoint-separation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Dendrite}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host continua, arbitrary path spaces, open-cover compactness, selected arc representatives, quotient equality of arcs, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-continuum visibility, tree-like exactness, endpoint exactness, unique-arc exactness, cutpoint-separation exactness, transport, replay, provenance, and local naming for BEDC.Derived.DendriteUp.

Chapter 5804

A Concrete Naming Certificate for $\mathsf{YoungIntegration}^{\uparrow}$

Following Young's integration theorem and Bishop–Bridges constructive analysis, the $\mathsf{YoungIntegration}^{\uparrow}$ packet records the finite dyadic route for integrating one Holder-controlled path against another. It sits beside the Holder-continuity surface of chapter 1766, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the tagged-mesh Riemann integral boundary of chapter 1526. The chapter names only a finite partition-and-error certificate; it does not import host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5804.1 (Young integration finite partition carrier)**.** *A* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the integrand path row, read through a* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
2. *G is the integrator path row, read through a second* $\mathsf{HolderContinuity}^{\uparrow}$ *packet with its own source interval observations, target Real estimates, rational exponent, and displayed Holder constant.*
3. *A and B are the two Holder exponent ledgers. They include the finite comparison row witnessing that the displayed exponents have a Young margin for the dyadic partition family.*
4. *W is the dyadic partition-window row. It lists the inspected endpoints, dyadic mesh refinements, cell order, adjacent-cell handoffs, and the finite cells on which the integration read is allowed to form sums.*
5. *V is the cross-variation tolerance ledger. It records the cellwise oscillation products, Holder-bound products, Young-margin comparisons, and finite sum-difference tolerances consumed by the route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger for endpoint radii, mesh widths, exponent-comparison margins, oscillation products, and final error bounds.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational trace of dyadic Young sums.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the integral value, reached only after the two Holder rows, dyadic windows, cross-variation ledger, dyadic tolerances, and regular readback have been displayed.*
9. *H records componentwise* hsame *transport for the two Holder paths, exponent ledgers, partition windows, cross-variation ledger, dyadic ledger, readback, Real seal, replay, provenance, and local name.*
10. *C records displayed* Cont *replay from a Young-integration request through Holder admission, dyadic partition windows, cross-variation tolerances, regular rational readback, and the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{YoungIntegration}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, G, A, B, W, V, D, R, E, H, C, P, N$. It has no coordinate for an ambient path space, quotient equality of functions, a selected sequence of partitions, a hidden limiting argument, countable choice, measure completion, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.YoungIntegrationUp.YoungIntegrationTasteGate single carrier alignment`

Theorem 5804.2 (Young integration NameCert obligations)**.** *For every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *carrier $Y = (F, G, A, B, W, V, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$ *surface is exhausted by admission of two* $\mathsf{HolderContinuity}^{\uparrow}$ *path rows, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting,* $\mathsf{DyadicRatCore}^{\uparrow}$ *exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No accepted consumer may replace these rows by host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, or ambient completeness of* $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5804.1. The rows F and G expose the two Holder-controlled paths, and A, B expose only the finite exponent and Young-margin comparisons attached to those rows.

The partition and tolerance data are then forced. The row W lists the dyadic windows available to the route, while V records the cross-variation products and sum-difference tolerances over exactly those windows.

The terminal read is D, R, E. Dyadic exactness is read through D, the regular rational trace through R, and the Real integral seal through E only after the preceding rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. Since the excluded objects have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5804.3 (Young integration dyadic error budget)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *read that reaches the* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite dyadic budget*

$$\mathsf{HolderContinuity}^{\uparrow}, \mathsf{HolderContinuity}^{\uparrow} \longmapsto \mathsf{DyadicPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The row V must display the cross-variation tolerance consumed between the partition windows and the rational readback. Therefore the exported integral value is a finite dyadic Young-sum seal, not a Riemann-integral replacement, a measure-completion integral, a quotient path equality, a selected limiting partition, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the two Holder rows. Their displayed exponents and constants are read through F, G, A, B, so the route has only the finite power-law data needed to bound oscillations on dyadic cells.

Next read the dyadic partition windows. The row W fixes the cells, endpoints, mesh refinements, and adjacent-cell handoffs before any sum-difference comparison is requested.

The error budget is the pair V, D. The cross-variation ledger V records the oscillation products and Young-margin comparisons, and the dyadic ledger D records the exact rational tolerances used to compare any two displayed sums.

The readback and seal are downstream. The regular sequence row R is the only rational trace exported from the finite budgets, and E is the only Real seal. Thus the consumer cannot obtain any of the excluded host-level or choice-level objects from the packet. □

Theorem 5804.4 (Young integration Riemann boundary)**.** *Every* $\mathsf{RiemannIntegral}^{\uparrow}$ *consumer of an accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet reads it only as a finite tagged-mesh boundary: two Holder path rows, a dyadic partition-window row, a cross-variation tolerance ledger, dyadic exactness, regular rational readback, Real sealing, transport, replay, provenance, and local naming. The consumer receives no independent Riemann-integral certificate, no Darboux gap claim, no quotient equality of integrands, no measure-completion object, and no selected mesh-limit principle.*

Proof. The only Riemann-facing data in the carrier are the displayed dyadic windows and finite sum budgets. The two Holder path rows provide values to be sampled, while W supplies the finite tagged-mesh boundary.

The cross-variation ledger is not a Darboux or measure-theoretic claim. It records only the tolerance comparisons needed for the Young-sum route before the regular rational readback is sealed as a Real value.

The provenance row may name $\mathsf{RiemannIntegral}^{\uparrow}$ as a consumer boundary, but the local certificate remains $F, G, A, B, W, V, D, R, E, H, C, P, N$. Therefore a Riemann-integral consumer receives a finite boundary row rather than any independent integral theorem or hidden limiting principle. □

Theorem 5804.5 (Young integration Cauchy-modulus consumer)**.** *Every accepted* $\mathsf{YoungIntegration}^{\uparrow}$ *packet whose cross-variation ledger* V *requests a tail precision must consume the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and* $\mathsf{Real}^{\uparrow}$ *sealing. The route remains finite: it exports the displayed Young partition windows, dyadic error budget, explicit Cauchy-modulus handoff, regular rational readback, and terminal Real seal, but it exports no selected limiting partition, countable-choice schedule, hidden modulus oracle, or ambient completion theorem.*

Proof. Project the carrier of Definition 5804.1. The rows W, V, D, R, E expose the partition windows, cross-variation tolerance ledger, dyadic error budget, regular rational readback, and terminal Real seal.

Use Definition 5804.3. The row V is the place where sum-difference tolerances and tail precision demands are recorded, and D is the dyadic ledger that makes those demands finite.

When such a tail precision is requested, apply the effective handoff of Definition 5728.3 and the schedule determinacy of Definition 5728.4. They force the precision request through an explicit $\mathsf{CauchyModulusBridge}^{\uparrow}$ schedule before R can be read and before E can seal the result.

The remaining rows H, C, P, N transport, replay, source, and name the same finite route. Since neither the Young carrier nor the bridge carrier contains a selected limiting partition, countable-choice schedule, hidden oracle, or ambient completion coordinate, no such object is exported. □

Falsifiable prediction: An accepted $\mathsf{YoungIntegration}^{\uparrow}$ packet is refuted if a consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting either $\mathsf{HolderContinuity}^{\uparrow}$ path row, the dyadic partition-window row W, the cross-variation ledger V, the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: HolderContinuity, DyadicPartition, DyadicRatCore, RegSeqRat, Real, RiemannIntegral: none is bijective with $\mathsf{YoungIntegration}^{\uparrow}$ because the packet jointly carries two Holder paths, Young-margin exponent rows, dyadic partition windows, cross-variation tolerances, dyadic exactness, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}$.

Constructive story: A $\mathsf{YoungIntegration}^{\uparrow}$ packet is generated from two HolderContinuityUp path rows, exponent-margin ledgers, dyadic partition windows, cross-variation tolerance accounting, DyadicRatCore exactness, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5804.1, Definition 5804.2, Definition 5804.3, Definition 5804.4, and Definition 5804.5 seed the finite dyadic Young-integration route over HolderContinuityUp, DyadicPartitionUp, DyadicRatCoreUp, CauchyModulusBridgeUp, RegSeqRatUp, RealUp, and RiemannIntegralUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungIntegration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host path spaces, quotient equality of functions, selected limiting partitions, countable choice, measure completion, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked two-path admission, exponent-margin comparison, dyadic partition-window formation, cross-variation tolerance accounting, dyadic exactness, regular rational readback, Real-seal nonescape, and structural transport for BEDC.Derived.YoungIntegrationUp.

Chapter 5805

A Concrete Naming Certificate for $\mathsf{VariationDiminishing}^{\uparrow}$

Following Schoenberg and Karlin, the $\mathsf{VariationDiminishing}^{\uparrow}$ packet records a finite analysis row for transformations that do not increase displayed sign variation. It consumes the total-variation function surface of chapter 2257, the bounded-variation real-function carrier of chapter 3654, the regulated-function source of chapter 1554, and the tagged integration boundary of chapter 1526. The packet names the finite comparison by which a kernel or transformation row is read as variation-diminishing without importing a host operator algebra, total-positivity theorem, selected partitions, quotient functions, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5805.1 (Variation-diminishing carrier)**.** *A* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, T, R, I, K, M, H, C, P, N)$$

with the following rows:

1. *B is the* $\mathsf{BoundedVariationRealFunction}^{\uparrow}$ *source row naming the accepted bounded-variation input surface.*
2. *T is the* $\mathsf{TotalVariationFunction}^{\uparrow}$ *row naming the displayed total-variation ledger used to count variation before and after the transformation.*
3. *R is the* $\mathsf{RegulatedRealFunction}^{\uparrow}$ *row naming the regulated readback through which input and output functions are compared by finite step windows.*
4. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row naming the finite tagged-mesh integration boundary used when the transformation is read by kernel action or integral averaging.*
5. *K is the kernel or transformation row, including its displayed positivity-order or sign-regularity window.*
6. *M is the variation comparison row recording that each accepted finite sign-change ledger for the output is routed to a no-larger sign-change ledger for the input.*
7. *H records componentwise* hsame *transport for bounded-variation source, total-variation ledger, regulated readback, integral boundary, kernel row, comparison row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite transformation request through $B, T, R, I, K, M, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{VariationDiminishing}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{TotalVariationFunction}^{\uparrow}$, $\mathsf{RegulatedRealFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, T, R, I, K, M, H, C, P, N$. It has no coordinate for host function equality, quotient representatives, selected limiting partitions, a global total-positivity theorem, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.VariationDiminishingUp.VariationDiminishingTasteGate single carrier alignment`

Theorem 5805.2 (Variation-diminishing NameCert obligations)**.** *For every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *carrier*

$$V = (B, T, R, I, K, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source row B, total-variation ledger T, regulated-function readback R, Riemann-integral boundary I, kernel or transformation row K, variation comparison row M, componentwise transport H, displayed replay C, provenance P, and local naming N.*

The obligation surface requires every output variation read to pass through the displayed comparison row M after B, T, R, I, K have been projected. It refuses host operator equality, quotient equality of functions, selected partitions, unrestricted total positivity, measure-theoretic kernels, and ambient completeness of $\mathsf{Real}^{\uparrow}$.

Proof. Project the carrier of Definition 5805.1. The source row B supplies the bounded-variation input, while T supplies the total-variation ledger on which finite sign-change comparisons can be read.

The rows R and I keep the analytic route finite. Regulated readback gives step-window access to input and output functions, and the Riemann-integral boundary records only the tagged-mesh action needed by a kernel or averaging read.

The row K names the transformation window, and M is the only row that compares variation before and after the transformation. Thus the NameCert consumer obtains a variation-diminishing assertion only through the displayed comparison row, not through a host theorem about all kernels.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate stores quotient functions, selected limiting partitions, unrestricted total positivity, measure completion, or ambient real completeness, no local obligation can require them. □

Theorem 5805.3 (Variation-diminishing total-variation route)**.** *Every accepted* $\mathsf{VariationDiminishing}^{\uparrow}$ *consumer read factors through the route*

$$\mathsf{BoundedVariationRealFunction}^{\uparrow},\ \mathsf{TotalVariationFunction}^{\uparrow},\ \mathsf{RegulatedRealFunction}^{\uparrow},\ \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{VariationD}$$

The output variation ledger is accepted only after the bounded-variation input, cumulative total-variation ledger, regulated readback, tagged integration boundary, kernel row, and comparison row have been displayed. The route exports no selected limit, quotient function equality, global total-positivity principle, measure-completion object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. A variation-diminishing request first projects B and T. These rows bind the input function to the bounded-variation and total-variation sibling surfaces, so the requested comparison starts from finite variation data rather than from a host function space.

Next project R and I. The regulated row supplies finite step-window readback, while the integral boundary supplies the tagged-mesh route through which a kernel or averaging transformation can be inspected.

The transformation row K and comparison row M then give the only accepted variation comparison. They state that the displayed output sign-change ledger routes to a no-larger input ledger in the same finite packet.

Finally H, C, P, N preserve the route under transport, replay, provenance, and local naming. Hence every public read of $\mathsf{VariationDiminishing}^{\uparrow}$ passes through the four cited sibling anchors and does not acquire the excluded host-level principles. □

Falsifiable prediction: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet whose output variation read bypasses the displayed total-variation ledger or the regulated readback row is not accepted by the local NameCert.

Independence witness: total-variation-function, bounded-variation-real-function, regulatedrealfunction, riemannintegral: $\mathsf{VariationDiminishing}^{\uparrow}$ is not bijective to any listed sibling because it binds the source variation ledger, regulated readback, tagged integration boundary, transformation row, and no-increase comparison as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationDiminishing}^{\uparrow}$ packet is generated from bounded-variation source data, a total-variation ledger, regulated readback, Riemann tagged-mesh boundary, a kernel or transformation row, a variation comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5805.1, Definition 5805.2, Definition 5805.3, the falsifiable prediction, and the independence witness seed the finite variation-diminishing route over the current analysis rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationDiminishing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host operator equality, quotient equality of functions, selected limiting partitions, unrestricted total positivity, measure completion, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, bounded-variation source visibility, total-variation ledger exactness, regulated readback, tagged integration boundary, transformation-row exposure, comparison-row stability, transport, replay, provenance, and local naming for future BEDC.Derived.VariationDiminishingUp.

$\mathsf{FrobeniusReciprocity}^{\uparrow}$ routes induction–restriction comparison through chapter 247, chapter 246, and chapter 245 before the child file states the local certificate.

Chapter 5806

A Concrete Naming Certificate for $\mathsf{FrobeniusReciprocity}^{\uparrow}$

Following Frobenius on induced characters, Serre's finite-group representation theory, and the induction–restriction formulation used by Fulton–Harris, the $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet records the finite comparison row between induction from a subgroup and restriction to that subgroup. It consumes the action rows of chapter 3662, the subgroup source of chapter 80, the representation-ring ledger of chapter 246, the induction–restriction packet of chapter 247, and the character-pairing surface of chapter 245. The packet names only finite source rows, adjunction ledgers, character-pairing boundaries, transport, replay, provenance, and downstream handoff rows; it does not assert a host category of representations, a functor equality, semisimplicity, Schur orthogonality, or a completed character table.

Theorem 5806.1 (FrobeniusReciprocityUp source obligations)**.** *For every accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet, the source surface is the displayed finite tuple*

$$\mathcal{F} = (G, H, R, I, Q, \Phi, \eta, \epsilon, \chi, \lambda, C, P, N),$$

where G is the consumed $\mathsf{FiniteGroupRepresentation}^{\uparrow}$ *action row, H is the consumed* $\mathsf{Subgroup}^{\uparrow}$ *row, R is the consumed* $\mathsf{RepresentationRing}^{\uparrow}$ *ledger, I is the consumed* $\mathsf{InducedRep}^{\uparrow}$ *induction–restriction row, Q is the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *pairing row, Φ is the Frobenius comparison row, η, ϵ are the displayed unit and counit rows, χ is the character-pairing boundary, λ is the finite ledger of visible source and endpoint reads, C is componentwise* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *row. No source obligation is imported from a host representation category, quotient carrier, functor category, or external adjunction object.*

Proof. Project the accepted packet. The source rows are precisely G, H, R, I, Q together with the local comparison rows $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

The consumed rows are already finite BEDC certificates. The finite-group representation row supplies the action display, the subgroup row supplies the inclusion boundary, the representation-ring row supplies the Grothendieck ledger, the induced-representation row supplies the induction–restriction endpoints, and the character-theory row supplies the pairing surface.

Since the tuple has no coordinate for a host representation category, quotient carrier, functor category, or external adjunction object, the listed rows exhaust the source obligations. □

Theorem 5806.2 (FrobeniusReciprocityUp adjunction ledger)**.** *The adjunction data of an accepted* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet are exactly the finite rows $\Phi, \eta, \epsilon, \lambda$ over the consumed sources*

H, R, I. The row Φ compares the displayed induction endpoint with the displayed restriction endpoint, η and ϵ record the unit and counit witnesses as BHist *rows, and λ records the visible continuations among source, endpoint, and Frobenius reads. The ledger exports no host naturality square, functor equality, adjunction object, or semisimple decomposition.*

Proof. Begin with the induction–restriction source. The packet consumes the $\mathsf{InducedRep}^{\uparrow}$ row I, so the available endpoints are the displayed induction and restriction rows already carried by that certificate.

The Frobenius comparison is read through Φ. The unit and counit witnesses are read through η and ϵ, and every visible path among source and endpoint reads is recorded by the finite ledger λ.

Exactness is projection from the tuple. Any row outside $\Phi, \eta, \epsilon, \lambda$ is not part of the adjunction ledger, and no host naturality square or functor equality can be recovered from a coordinate that the packet does not carry. □

Theorem 5806.3 (FrobeniusReciprocityUp character-pairing boundary)**.** *Every character-pairing read of a* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet factors through the consumed* $\mathsf{CharacterTheory}^{\uparrow}$ *row Q, the representation-ring source R, the induction–restriction source I, and the displayed pairing boundary χ before it can inspect Φ, η, ϵ or λ. The boundary supplies only the finite character-pairing comparison needed for reciprocity and does not export Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra.*

Proof. Read the character side first. The only character-theory coordinate is Q, and the only representation-ring coordinate feeding it is R.

The induction and restriction endpoints enter through I. The displayed pairing boundary χ then names the finite comparison used by the Frobenius row before the ledger reads $\Phi, \eta, \epsilon, \lambda$.

Because the tuple has no coordinate for Schur orthogonality, irreducible decomposition, character-table computation, or a completed class-function algebra, every character-pairing read remains inside the stated finite boundary. □

Theorem 5806.4 (FrobeniusReciprocityUp NameCert surface)**.** *The* $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$ *obligation surface is the conjunction of the source obligations of Definition 5806.1, the adjunction ledger of Definition 5806.2, the character-pairing boundary of Definition 5806.3, componentwise* hsame *transport on all displayed source and ledger rows, finite* Cont *replay through C, and* Pkg *provenance through P. The surface exports no obligation outside the displayed tuple $\mathcal{F}$.*

Proof. First combine the source obligations with the adjunction ledger. The source theorem identifies the consumed BEDC certificates, and the ledger theorem identifies the only Frobenius comparison, unit, counit, and continuation rows.

Next add the character-pairing boundary. It places every character read behind Q, R, I, χ before the Frobenius ledger can be inspected.

Finally apply componentwise transport and replay. The rows C and P preserve exactly the displayed tuple, and the local row N names the resulting certificate. Since no obligation outside $\mathcal{F}$ is available, these clauses give the stated NameCert surface. □

Theorem 5806.5 (FrobeniusReciprocityUp consumer handoff)**.** *A downstream consumer of* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *may read exactly the consumed finite-group action row G, subgroup row H, representation-ring row R, induced-representation row I, character-theory row Q, Frobenius rows Φ, η, ϵ, character-pairing boundary χ, visible ledger λ, replay row C, provenance row P, and local name row N. Any consumer handoff to* $\mathsf{InducedRep}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, *or* $\mathsf{CharacterTheory}^{\uparrow}$ *factors through those*

rows and cannot bypass them through a host category, functor equality, semisimplicity theorem, or character-table computation.

Proof. Start with a downstream read. The only visible source coordinates are G, H, R, I, Q, and the only local Frobenius coordinates are $\Phi, \eta, \epsilon, \chi, \lambda, C, P, N$.

For an $\mathsf{InducedRep}^{\uparrow}$ consumer, the handoff is through I together with the Frobenius rows. For a $\mathsf{RepresentationRing}^{\uparrow}$ consumer, the handoff is through R and the source ledger. For a $\mathsf{CharacterTheory}^{\uparrow}$ consumer, the handoff is through Q and the pairing boundary χ.

The replay and provenance rows preserve the same finite route. Thus a consumer cannot bypass the displayed packet through a host category, functor equality, semisimplicity theorem, or character-table computation. □

Theorem 5806.6 (FrobeniusReciprocityUp sibling lattice)**.** *The* $\mathsf{FrobeniusReciprocity}^{\uparrow}$ *packet is a finite comparison consumer of the sibling carriers in chapter 3662, chapter 80, chapter 246, chapter 247, and chapter 245. Its local rows compare induction and restriction after those carriers are visible; they do not form an isolated kernel row and do not replace the representation, subgroup, ring, induced-representation, or character-theory certificates.*

Proof. Read the source tuple of Definition 5806.1. The rows G, H, R, I, Q are precisely the five sibling certificates named in the statement.

The consumer handoff of Definition 5806.5 permits downstream reads only through those rows together with the Frobenius comparison, unit, counit, pairing boundary, ledger, replay, provenance, and local name rows.

Therefore the packet sits inside the displayed representation-theory lattice. It is a comparison consumer over the sibling carriers, not a standalone kernel surface from which the sibling certificates can be reconstructed. □

Closure status for $\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5806.1, Definition 5806.2, Definition 5806.3, Definition 5806.4, Definition 5806.5, and Definition 5806.6 seed the finite Frobenius reciprocity certificate over displayed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FrobeniusReciprocity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, semisimplicity, Schur orthogonality, irreducible decomposition, character-table computation, functor-category equality, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, adjunction-ledger exactness, character-pairing boundary, NameCert surface, and downstream handoff for the same finite packet.

Constructive story: A $\mathsf{FrobeniusReciprocity}^{\uparrow}$ packet is generated from consumed finite-group representation, subgroup, representation-ring, induced-representation, and character-theory rows, then records a finite Frobenius comparison, unit and counit rows, character-pairing boundary, visible continuation ledger, package provenance, and local NameCert evidence.

Chapter 5807

A Concrete Naming Certificate for $\mathsf{NormalOperator}^{\uparrow}$

Following Reed and Simon, the $\mathsf{NormalOperator}^{\uparrow}$ packet records the standard normal-operator surface as finite NameCert data over displayed Hilbert and inner-product candidates. It consumes the inner-product carrier of chapter 86, the Hilbert carrier of chapter 91, the linear-map rows of chapter 48, the operator-norm boundary of chapter 3603, and the downstream spectral-theorem consumer of chapter 214. The packet exposes only a bounded linear operator, its displayed adjoint-action row, the norm ledger, and the normality commutator row; it does not claim spectral measure existence, functional calculus, diagonalization, projection-valued measures, or a host operator algebra.

Definition 5807.1 (NormalOperator carrier)**.** *For a displayed* $\mathsf{Hilbert}^{\uparrow}$ *candidate* H *with* $\mathsf{InnerProduct}^{\uparrow}$ *row* I*, a* $\mathsf{NormalOperator}^{\uparrow}(H)$ *carrier is a finite* BHist *packet*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N)$$

with the following ten displayed fields:

1. *H is the* $\mathsf{Hilbert}^{\uparrow}$ *carrier row and I is the* $\mathsf{InnerProduct}^{\uparrow}$ *row over the same point-history carrier.*
2. *T is a bounded* $\mathsf{LinMap}^{\uparrow}$ *endpoint action on the carried Hilbert endpoints.*
3. *T^* is the displayed adjoint-action row, certified by the finite inner-product comparison*

   $$\langle Tx, y\rangle_I \ \mathsf{hsame}\ \langle x, T^*y\rangle_I$$

   for the vector histories inspected by the packet.
4. *A is the adjoint-domain ledger, recording that the adjoint row is read over the same Hilbert, inner-product, linear-map, and boundedness surfaces as T.*
5. *K is the* $\mathsf{OperatorNorm}^{\uparrow}$ *bound row for T and for the displayed adjoint action T^*.*
6. *Q is the normality commutator row*

   $$T^*T \ \mathsf{hsame}\ TT^*$$

   read only on certified vector histories.
7. *L is the operator graph ledger: it stores inspected input histories, output histories, linearity rows, adjoint rows, norm-bound rows, and the commutator row.*

8. *H_T records componentwise* hsame *transport for the Hilbert source, inner-product row, endpoint actions, adjoint row, norm row, commutator row, and ledger.*
9. *C records displayed* Cont *replay for endpoint action requests, adjoint-action requests, norm-bound requests, and commutator inspections.*
10. *P, N are the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *naming rows.*

The classifier compares accepted carriers by action agreement on certified vector histories, adjoint-action agreement on the same inspected histories, transport of the operator-norm row, and transport of the graph, adjoint, normality, continuation, provenance, and local-name ledgers. It has no coordinate for host operator equality, quotient completion, spectral measures, eigenvector bases, or external functional calculus.

Lean4 ✓ `BEDC.Derived.NormalOperatorUp.NormalOperatorTasteGate single carrier alignment`

Theorem 5807.2 (NormalOperator NameCert obligations)**.** *For every accepted* $\mathsf{NormalOperator}^\uparrow$ *carrier*

$$\mathcal{N} = (H, I, T, T^*, A, K, Q, L, H_T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NormalOperator}^\uparrow}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the common* $\mathsf{Hilbert}^\uparrow$ *and* $\mathsf{InnerProduct}^\uparrow$ *rows;*
2. *bounded* $\mathsf{LinMap}^\uparrow$ *action for T on certified vector histories;*
3. *adjoint-action agreement for T^* against the displayed inner product;*
4. $\mathsf{OperatorNorm}^\uparrow$ *bound rows for the operator action and adjoint action;*
5. *normality commutator exactness through the displayed row Q;*
6. *graph-ledger exactness for input histories, output histories, linearity rows, adjoint rows, norm rows, and commutator rows;*
7. *classifier transport by action agreement, adjoint-action agreement, norm-row agreement, and componentwise ledger transport;*
8. Cont *replay,* Pkg *provenance, and local naming for exactly the displayed packet;*
9. *refusal of spectral measure existence, diagonalization, projection-valued measures, host operator quotients, and unrecorded functional calculus.*

Proof. Project the carrier of Definition 5807.1. The Hilbert and inner-product source rows, endpoint action, adjoint-action row, adjoint-domain ledger, norm row, commutator row, graph ledger, transport row, continuation row, provenance row, and local name are the full displayed inventory.

The adjoint and boundedness obligations are read before any spectral consumer appears. The adjoint row is the finite inner-product comparison for certified vector histories, and the norm row is the displayed $\mathsf{OperatorNorm}^\uparrow$ bound attached to both endpoint actions.

Normality is exactly the commutator row Q. Because Q compares T^*T and TT^* only through inspected vector histories and their graph ledger, the packet does not appeal to host equality of operators.

Finally H_T, C, P, N transport, replay, source, and name those rows. Since the carrier has no spectral-measure, diagonalization, projection-valued-measure, quotient, or functional-calculus coordinate, the listed rows exhaust the local normal-operator NameCert obligations. □

Theorem 5807.3 (NormalOperator InnerProduct spectral anchor)**.** *Every downstream spectral read of an accepted* $\mathsf{NormalOperator}^{\uparrow}$ *packet factors through the common Hilbert and inner-product rows, the bounded linear action, the displayed adjoint-action row, the operator-norm bound, and the normality commutator ledger before it can be consumed by the* $\mathsf{SpectralTheorem}^{\uparrow}$ *surface. The route is*

$$(H, I) \longmapsto (T, T^*, A, K, Q, L) \longmapsto (H_T, C, P, N),$$

and it exports no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality.

Proof. Expose the accepted packet by Definition 5807.1. The only analytic source rows are the Hilbert carrier, inner-product comparison row, endpoint action, displayed adjoint action, norm bound, and normality commutator.

Apply Definition 5807.2. The graph ledger forces the operator action, adjoint action, boundedness evidence, and commutator evidence to be read from one finite packet, and classifier transport keeps the operator and adjoint rows over the same certified vector histories.

It remains to hand the packet to spectral consumers. The $\mathsf{SpectralTheorem}^{\uparrow}$ chapter may consume the normality anchor only through the displayed rows already named here. Because no spectral projection, spectral measure, eigenbasis, Borel functional calculus, or host operator-algebra equality is a field of the carrier, the anchor supplies a pre-spectral normal-operator surface rather than a spectral theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5807.1, Definition 5807.2, and Definition 5807.3 seed the normal-operator surface over Hilbert, inner-product, bounded linear-map, adjoint-action, operator-norm, and commutator rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim spectral measures, spectral projections, diagonalization, eigenbasis construction, Borel functional calculus, host operator quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjoint-action agreement, norm-bound stability, normality commutator exactness, classifier transport, graph-ledger exactness, spectral-consumer nonescape, and bridge data for BEDC.Derived.NormalOperatorUp.

Constructive story: A $\mathsf{NormalOperator}^{\uparrow}$ packet is generated from a Hilbert source, an inner-product row, a bounded linear endpoint action, a displayed adjoint action, operator-norm bounds, a normality commutator row, graph and transport ledgers, continuation replay, provenance, and a local NameCert row.

Chapter 5808

A Concrete Naming Certificate for $\mathsf{HausdorffMeasure}^{\uparrow}$

Following Hausdorff and Federer's metric measure theory, the $\mathsf{HausdorffMeasure}^{\uparrow}$ packet records the finite dyadic-cover surface through which Hausdorff content and scale-stable measure reads become visible inside BEDC. It sits over the metric carrier of chapter 57, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the cover-valued outer-measure boundary of chapter 3385, and the finite scale-cover comparison surface of chapter 3877. The packet names only displayed dyadic cover packets, diameter budgets, weighted cover sums, scale comparison, transport, replay, provenance, and local naming; it is not a quotient cover family, not a selected infimum, not countable choice over covers, and not an ambient measure primitive.

Definition 5808.1 (Hausdorff measure dyadic-cover carrier)**.** *A* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (M, D, R, O, B, C, S, T, E, K, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite point classifier and distance comparisons used to bound cover diameters.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row, naming the dyadic radius budget and dyadic exponent used by the cover packet.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *value-seal row for diameter powers and finite cover sums.*
4. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *handoff row, read only as a finite cover-valued boundary and not as a completed measure primitive.*
5. *B is the* $\mathsf{BoxDimension}^{\uparrow}$ *comparison row, exposing finite scale-cover counts against the same dyadic scale.*
6. *C is the finite dyadic cover packet, listing the displayed cover cells, their metric diameter witnesses, and their membership reads.*
7. *S is the cover-sum ledger, recording the finite Real-sealed sum of dyadic diameter weights.*
8. *T is the scale-stability ledger, recording that dyadic refinement reads the same metric source, outer-measure boundary, and box-dimension comparison rows.*
9. *E records componentwise* hsame *transport for metric, dyadic scale, Real seal, outer-measure handoff, box-dimension comparison, cover packet, cover sum, and stability rows.*

10. *K records displayed* Cont *replay from a metric source and dyadic scale through C, S, T, E, P, N.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffMeasure}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{BoxDimension}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, D, R, O, B, C, S, T, E, K, P, N$. It has no coordinate for quotient cover families, selected infima, countable-choice schedules over covers, host sigma-algebras, or ambient measure primitives.

Lean4 ✓ `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate single carrier alignment`

Theorem 5808.2 (Hausdorff measure NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier $H = (M, D, R, O, B, C, S, T, E, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$ *surface consists exactly of metric source admission, dyadic scale exposure, Real value sealing, finite outer-measure handoff, box-dimension scale comparison, finite dyadic cover admission, cover-sum exactness, scale-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every measure-facing or dimension-facing consumer must read the displayed route*

$$\mathsf{Met}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto C, S, T \longmapsto \mathsf{OuterMeasure}^{\uparrow}, \mathsf{BoxDimension}^{\uparrow} \longmapsto E, K, P, N$$

before treating the packet as a Hausdorff-measure read.

Proof. Project the carrier of Definition 5808.1. The metric row M, dyadic row D, and Real seal row R are the only sources for distances, scales, and finite value readback.

The cover packet C and cover-sum ledger S are finite rows. They record displayed cover cells and dyadic diameter weights, so no step selects a minimizing cover family or takes an external infimum.

The outer-measure and box-dimension rows O, B are downstream comparison boundaries. They can read the same finite cover data, but they do not add a completed host measure, a quotient of covers, or countable choice over refinements.

Finally T, E, K, P, N transport, replay, source, and name the same finite route. Since every exported obligation is one of the listed rows, the local classifier cannot expose an ambient measure primitive or any detached cover selector. □

Theorem 5808.3 (Hausdorff measure dyadic-cover stability)**.** *Let $H = (M, D, R, O, B, C, S, T, E, K, P, N)$ be an accepted* $\mathsf{HausdorffMeasure}^{\uparrow}$ *carrier. For any displayed dyadic refinement of the scale row D, the scale-stability ledger T forces the refined read to reuse the same* $\mathsf{Met}^{\uparrow}$ *source, the same* $\mathsf{Real}^{\uparrow}$ *value-seal route, the same finite* $\mathsf{OuterMeasure}^{\uparrow}$ *cover boundary, and the same* $\mathsf{BoxDimension}^{\uparrow}$ *scale-count comparison before any consumer reads a Hausdorff-measure claim. The stable route is*

$$M, D, R, C, S, T \longmapsto O, B \longmapsto E, K, P, N.$$

It exports no quotient cover family, selected infimum, countable-choice cover schedule, host sigma-algebra closure, or ambient measure primitive.

Proof. Read the carrier of Definition 5808.1. A dyadic refinement changes only the displayed scale request inside D and the finite cover packet admitted by C.

Use the stability ledger T. It records that the refined cover is replayed against the same metric source and that its finite cover sum is sealed through the same Real row, so the refinement cannot introduce a detached minimizing cover or a host infimum.

The comparison rows O and B then read the refined packet as finite outer-measure and box-dimension boundary data. Componentwise transport and displayed replay through E, K, P, N preserve that route, while quotient covers, countable choice, host sigma-algebras, and ambient measure primitives remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HausdorffMeasure}^{\uparrow}$ consumer that reaches a Hausdorff-measure read without first exposing a MetricUp source row, a DyadicRatCore scale row, a RealUp value seal, a finite dyadic cover packet, a finite cover-sum ledger, an OuterMeasureUp handoff, a BoxDimensionUp comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row refutes the carrier surface.

Independence witness: metric, dyadicratcore, real, outermeasure, boxdimension: $\mathsf{HausdorffMeasure}^{\uparrow}$ is not bijective to any listed sibling because it binds metric diameter data, dyadic scales, Real-sealed finite cover sums, outer-measure cover boundaries, and box-dimension scale counts into one finite cover-stability NameCert packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffMeasure}^{\uparrow}$ packet is generated from a MetricUp source row, DyadicRatCore scale row, RealUp value-seal row, finite OuterMeasureUp handoff, BoxDimensionUp comparison row, finite dyadic cover packet, cover-sum ledger, scale-stability ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5808.1, Definition 5808.2, and Definition 5808.3 seed the finite dyadic-cover Hausdorff-measure interface over MetricUp, DyadicRatCoreUp, RealUp, OuterMeasureUp, and BoxDimensionUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffMeasure}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HausdorffMeasureUp.HausdorffMeasureTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim quotient cover families, selected infima, countable choice over covers, host sigma-algebra completion, ambient measure primitives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic scale exposure, finite cover admission, cover-sum exactness, scale-stability replay, outer-measure handoff, box-dimension comparison, transport, provenance, and no-escape for BEDC.Derived.HausdorffMeasureUp.

Chapter 5809

A Concrete Naming Certificate for $\mathsf{LocallyConnected}^{\uparrow}$

Following Kelley and Bourbaki on local connectedness, the $\mathsf{LocallyConnected}^{\uparrow}$ packet records finite neighbourhood-component data rather than a selected basis of connected open sets. It sits beside the global connectedness carrier of chapter 2353, the compact metric source discipline of chapter 129, the metric ball rows of chapter 57, and the open-neighbourhood ledger of chapter 88. The packet exposes the local component route needed by compact-metric and connected-space consumers without importing path selection, arc connectedness, open-cover compactness, or a host topological basis.

Definition 5809.1 (Locally connected carrier)**.** *A* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (T, M, G, K, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *open-neighbourhood row whose entries name the ambient open requests.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *ball-observation row used when the local request is metric.*
3. *G is the* $\mathsf{ConnectedSpace}^{\uparrow}$ *global component row available to consumers that compare local and global connectedness.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-source row available to compact-metric consumers.*
5. *B is the finite neighbourhood-refinement row: each displayed point and open request has a named smaller neighbourhood row.*
6. *C is the local connected-component row attached to the refinement in B.*
7. *H records componentwise* hsame *transport for the topology, metric, connectedness, compact-metric, refinement, component, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from an ambient open request through metric ball observations when present, the finite refinement row, and the local component row.*
9. *P is the* Pkg *provenance over* $\mathsf{LocallyConnected}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ConnectedSpace}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, M, G, K, B, C, H, R, P, N$. It has no coordinate for selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, or a host basis of connected neighbourhoods.

Lean4 ✓ `BEDC.Derived.LocallyConnectedUp.TasteGate.LocallyConnectedTasteGate single carrier alignment`

Theorem 5809.2 (Locally connected NameCert obligations)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology, metric, connected-space, compact-metric, finite neighbourhood-refinement, local component, transport, replay, provenance, and naming rows; neighbourhood refinement from T through B; component exposure through C; metric compatibility through M when the request is ball-based; compact-metric compatibility through K for compact finite-source consumers; connected-space comparison through G; and non-escape from selected paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice.*

Proof. Project the carrier of Definition 5809.1. The projection exposes exactly the rows $T, M, G, K, B, C, H, R, P, N$ so every local naming obligation must be read from those coordinates.

The neighbourhood part is finite. An ambient open request is first read in T, then refined by the displayed row B, and only then receives the local component row C.

The metric and compact-metric rows are compatibility rows rather than extra selectors. A ball-based request may inspect M, and a compact-metric consumer may inspect K, but both routes still pass through the same B and C rows before local connectedness is exported.

Finally, H and R transport and replay the displayed route, while P and N source and name it. Since paths, arcs, arbitrary open-cover compactness, quotient component spaces, and host basis choice are not rows of the carrier, the NameCert surface cannot export them. □

Theorem 5809.3 (Locally connected neighbourhood component route)**.** *Every connected-space or compact-metric consumer of an accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *packet reaches local connectedness through*

$$\mathsf{Topology}^{\uparrow},\quad \mathsf{Met}^{\uparrow},\quad \mathsf{ConnectedSpace}^{\uparrow},\quad \mathsf{CompactMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}.$$

The finite route reads an ambient neighbourhood, optionally reads metric ball observations, refines to a displayed smaller neighbourhood, and then reads the named local connected component. It does not select a path, an arc, a subcover, a quotient component representative, or a host connected-neighbourhood basis.

Proof. Start with a consumer that asks for local connectedness around a displayed point. Carrier admission gives the consumer the topology row T, the metric row M when a ball observation is part of the request, the global connectedness row G, and the compact-metric row K.

The route then passes through B and C. The row B supplies the finite neighbourhood refinement, and C supplies the local component attached to that refinement.

The structural rows H, R, P, N do not add a hidden selector. They transport, replay, source, and name the same finite route, so no path, arc, arbitrary subcover, quotient representative, or host basis can replace the displayed neighbourhood-component data. □

Theorem 5809.4 (Locally connected obligation closure surface)**.** *For every accepted* $\mathsf{LocallyConnected}^{\uparrow}$ *carrier* $L = (T, M, G, K, B, C, H, R, P, N)$*, the obligation-closure surface is exhausted by the displayed finite topology, metric, connected-space, compact-metric, neighbourhood-refinement, local-component, transport, replay, provenance, and naming rows. The row T supplies the open-neighbourhood*

source, M supplies metric-ball observations, G supplies connected-space comparison, K supplies compact-metric finite-source compatibility, B, C supply the local refinement and component rows, and H, R, P, N supply transport, replay, provenance, and local naming.

The local component route therefore consumes the connected-space carrier of chapter 2353, the compact-metric finite neighbourhood discipline of chapter 129, the metric-ball rows of chapter 57, and the topology neighbourhood ledger of chapter 88 before a local component can be named.

Proof. Apply Definition 5809.2 to the accepted carrier. It identifies the complete NameCert surface with the rows $T, M, G, K, B, C, H, R, P, N$.

Apply Definition 5809.3. The route from an open request to a local component passes through topology, optional metric-ball observations, connected-space comparison, compact-metric compatibility, finite refinement, and component rows before the structural rows transport and name the result.

The sibling lattice is read in that order: topology supplies the neighbourhood ledger, metric supplies ball observations, connected-space supplies the comparison carrier, and compact metric supplies the finite neighbourhood discipline. The local refinement and component rows then bind those sibling contributions into the $\mathsf{LocallyConnected}^{\uparrow}$ packet.

Thus the obligation surface is the stated finite route. Path or arc selection, open-cover compactness, quotient components, and a host basis are refused because none of them is a carrier coordinate or a row used by the two preceding theorems. □

Falsifiable prediction: Every accepted $\mathsf{LocallyConnected}^{\uparrow}$ read must expose a $\mathsf{Topology}^{\uparrow}$ open-neighbourhood row, optional $\mathsf{Met}^{\uparrow}$ ball observations, the $\mathsf{ConnectedSpace}^{\uparrow}$ comparison row when a global consumer is present, the $\mathsf{CompactMetric}^{\uparrow}$ finite-source row when a compact-metric consumer is present, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric, connected-space, compactmetric: $\mathsf{LocallyConnected}^{\uparrow}$ is not bijective to any listed sibling because it binds a local neighbourhood-refinement row to a local component row before connected-space or compact-metric consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5809.1, Definition 5809.2, and Definition 5809.3 seed the finite local neighbourhood-component route over topology, metric, connected-space, and compact-metric rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocallyConnected}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close path connectedness, arc connectedness, arbitrary open-cover compactness, quotient component spaces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighbourhood refinement, local component exposure, connected-space comparison, compact-metric compatibility, transport and replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.LocallyConnectedUp.

Constructive story: A $\mathsf{LocallyConnected}^{\uparrow}$ packet is generated from an open-neighbourhood row, optional metric ball observations, connected-space and compact-metric compatibility rows, a finite neighbourhood-refinement row, a local component row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5810

A Concrete Naming Certificate for $\mathsf{BishopCauchyMean}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, the $\mathsf{BishopCauchyMean}^{\uparrow}$ packet records the finite averaging step used when a Cauchy source is read through displayed rational windows and sealed as a Real-facing name. It sits after the $\mathsf{Cauchy}^{\uparrow}$ source route of chapter 3413, the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names finite averages of observed Cauchy data; it does not assert a selected limit, quotient stream equality, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5810.1 (Bishop Cauchy mean carrier)**.** *A* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Cauchy}^{\uparrow}$ *source row naming the Cauchy data whose finite observations are being averaged.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the requested dyadic radius, denominator budget, and comparison scale for the average.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the source entries inspected at tolerance D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating the inspected stream window into regular rational data.*
5. *A is the finite averaging ledger: it lists the inspected rational entries, their displayed weights, the denominator row, the partial-sum row, and the final mean row consumed by the certificate.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, D, W, R, A have been displayed.*
7. *H records componentwise* hsame *transport for the source, tolerance, window, readback, averaging ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a finite mean request through $S, D, W, R, A, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyMean}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $S, D, W, R, A, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$*, or an infinite averaging oracle.*

Lean4 ✓ `BEDC.Derived.BishopCauchyMeanUp.BishopCauchyMeanTasteGate single carrier alignment`

Theorem 5810.2 (Bishop Cauchy mean NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier*

$$M = (S, D, W, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$ *surface consists exactly of carrier admission for the Cauchy source row, dyadic tolerance row, finite stream window, regular rational readback, finite averaging ledger, terminal Real seal, componentwise transport row, displayed continuation replay, package provenance, and local naming row.*

The obligation surface requires Cauchy-source visibility through S, tolerance accounting through D, finite-window visibility through W, regular rational readback through R, average-ledger exactness through A, Real-seal nonescape through E, transport and replay stability through H and C, provenance through P, and local naming through N. It refuses selected limits, quotient stream equality, host real equality, countable choice, and ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the accepted carrier of Definition 5810.1. The only analytic rows are the Cauchy source S, the dyadic tolerance D, the finite window W, the regular rational readback R, the averaging ledger A, and the terminal Real seal E.

The averaging step is finite. The ledger A displays the inspected rational entries, their weights, the denominator row, the partial sums, and the final mean; no later consumer can ask for an average outside this listed window.

The terminal seal is downstream of the displayed route. Since the carrier has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, ambient completeness, or an infinite averaging oracle, those objects cannot enter the local NameCert surface.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. They preserve the displayed rows and do not add analytic content. □

Theorem 5810.3 (Bishop Cauchy mean finite-average stability)**.** *Let*

$$M = (S, D, W, R, A, E, H, C, P, N)$$

be an accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ *carrier. Every accepted consumer read of the mean row factors through the finite route*

$$\mathsf{Cauchy}^{\uparrow}(S),\ \mathsf{DyadicRatCore}^{\uparrow}(D),\ \mathsf{StreamName}^{\uparrow}(W),\ \mathsf{RegSeqRat}^{\uparrow}(R),\ A,\ \mathsf{Real}^{\uparrow}(E).$$

If the source row, tolerance row, finite window, readback row, and averaging ledger are transported componentwise by the displayed hsame *rows in H and replayed through C, then the consumer-visible mean row is unchanged as a* $\mathsf{BishopCauchyMean}^{\uparrow}$ *read. No quotient stream equality, selected limit, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row participates in the stability claim.*

Proof. Start with the consumer route. By the carrier definition, a mean read must first inspect the Cauchy source S, choose the displayed dyadic tolerance D, open the finite window W, and read the regular rational row R.

The average itself is then the ledger row A. Its entries, weights, denominator, partial sums, and final mean are finite displayed data, so componentwise hsame transport preserves exactly the rows that the consumer can inspect.

Replay through C exposes the transported packet with the same provenance and local naming rows P, N. The terminal Real seal E is reached from that transported finite route, not from a quotient stream or selected limit.

Thus any consumer-visible equality of mean reads is exhausted by $S, D, W, R, A, E, H, C, P, N$. The excluded rows are absent from the carrier and cannot affect the stability conclusion. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyMean}^{\uparrow}$ read must expose a Cauchy source, dyadic tolerance, StreamName window, RegSeqRat readback, finite averaging ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a Real or Completion consumer can read the mean boundary.

Independence witness: cauchy, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopCauchyMean}^{\uparrow}$ is not bijective to any listed sibling because it binds a finite averaging ledger between regular rational readback and the Real seal.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyMean}^{\uparrow}$ packet is generated from a Cauchy source row, DyadicRatCore tolerance row, StreamName finite window, RegSeqRat readback, finite averaging ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5810.1, Definition 5810.2, and Definition 5810.3 seed the finite Cauchy-mean certificate over Cauchy, DyadicRatCore, StreamName, RegSeqRat, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyMean}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite average ledger exactness, tolerance-window readback, regular rational handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyMeanUp.

Chapter 5811

A Concrete Naming Certificate for $\mathsf{BishopFixedPointInterval}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet records the constructive interval fixed-point route as finite NameCert data. It sits after the located interval source of chapter 394, the bisection discipline of chapter 1465, the finite Cauchy-modulus rows used by chapter 292, the regular rational readback of chapter 301, the finite name windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the fixed-point certificate for a displayed interval self-map; it does not import a host compactness theorem, quotient real equality, choice-selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5811.1 (Bishop fixed-point interval carrier)**.** *A* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, T, M, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the closed located interval, its endpoint reads, and its dyadic radius ledger.*
2. *T is the contraction self-map row over I, recording that every admitted interval point is sent back into the same located interval.*
3. *M is an explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *or Lipschitz modulus row that turns each tolerance request into a finite contraction budget.*
4. *B is the* $\mathsf{IntervalBisection}^{\uparrow}$ *or nested located interval row that records the displayed shrinking interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational midpoint, endpoint, and radius observations.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *fixed-point seal reached only after the interval, contraction, modulus, bisection, and readback rows have been exposed.*
7. *H records componentwise* hsame *transport for I, T, M, B, R, E and for the structural rows.*
8. *C records displayed* Cont *replay for interval requests, contraction reads, modulus reads, nested interval reads, rational readback, and real-seal requests.*

9. *P is the* Pkg *provenance over* BishopFixedPointInterval$^{\uparrow}$, LocatedInterval$^{\uparrow}$, IntervalBisection$^{\uparrow}$, CauchyModulus$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, T, M, B, R, E, H, C, P, N$. It has no coordinate for a host fixed-point theorem, quotient completion, selected limit, trichotomy, countable choice, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.BishopFixedPointIntervalUp.BishopFixedPointIntervalTasteGate single carrier alignment`

Theorem 5811.2 (Bishop fixed-point interval NameCert obligations)**.** *For every accepted* BishopFixedPointInterval$^{\uparrow}$ *carrier*

$$F = (I, T, M, B, R, E, H, C, P, N),$$

the local NameCert$_{\text{BishopFixedPointInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for the located interval row I, contraction self-map row T, explicit modulus row M, nested bisection row B, regular rational readback row R, terminal real seal E, componentwise transport row H, displayed replay row C, provenance row P, and local naming row N. Every fixed-point consumer must read these rows in the displayed order and cannot replace them by host interval compactness, quotient real equality, a selected limit, or an unrecorded convergence principle.*

Proof. Carrier admission is the finite packet of Definition 5811.1. The rows $I, T, M, B, R, E, H, C, P, N$ are the only coordinates available to the local certificate.

The positive interval route is then forced. The interval source is I, the self-map evidence is T, the contraction budget is M, and the nested interval chain is B; each row is compared only through the componentwise transport row H and replayed only through C.

The readback and seal are downstream. Rational observations pass through R and the dyadic and stream-name data named in P before the terminal Real$^{\uparrow}$ row E is exposed. Since no host compactness, quotient equality, selected limit, or external convergence coordinate occurs in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5811.3 (Bishop fixed-point interval Cauchy-modulus route)**.** *Let $F = (I, T, M, B, R, E, H, C, P, N)$ be an accepted* BishopFixedPointInterval$^{\uparrow}$ *packet. Each tolerance request for the fixed-point name is answered by first reading the* LocatedInterval$^{\uparrow}$ *row I, then the contraction self-map row T, then the explicit* CauchyModulus$^{\uparrow}$ *or Lipschitz row M, and only then the* IntervalBisection$^{\uparrow}$ *or nested located interval row B. The resulting finite observations are read through* StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* RegSeqRat$^{\uparrow}$ *as recorded by P, so no consumer can infer convergence from an unbounded host iteration or a choice-selected subsequence.*

Proof. Start with a tolerance request. The packet has no direct convergence row, so the request must project I to know the interval source and T to know that the displayed map remains inside the interval.

Next read the modulus row M. This row converts the tolerance into a finite contraction or Lipschitz budget. The nested row B can then expose only the finite bisection or located-interval observations licensed by that budget.

Finally use the readback rows listed in P. Stream windows, dyadic tolerances, and regular rational observations feed R, and H, C, N transport and name the same route. Thus the Cauchy route is explicit and finite, with no unbounded host iteration or selected subsequence. □

Theorem 5811.4 (Bishop fixed-point interval real-seal nonescape)**.** *For every accepted* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *packet, the terminal row* E *is exposed only as the* $\mathsf{Real}^{\uparrow}$ *seal of the displayed nested interval and Cauchy-modulus route*

$$I, T, M, B, R \longmapsto E \longmapsto H, C, P, N.$$

The seal may be consumed as the named fixed-point endpoint of the packet, but it cannot escape as quotient real equality, a proof of ambient $\mathsf{Real}^{\uparrow}$ *completeness, a host fixed-point theorem, an unrestricted interval compactness principle, or a selected representative outside the displayed rows.*

Proof. Read E from the carrier. It is terminal in the packet: the interval, self-map, modulus, nested interval, and rational readback rows precede it, and the structural rows only transport, replay, source, and name the resulting read.

Apply Definition 5811.3. Every tolerance request reaching E has already passed through the finite interval, contraction, modulus, bisection, stream, dyadic, and regular rational observations recorded by I, T, M, B, R, P.

It remains to rule out escape. The carrier has no coordinate for quotient real equality, ambient completeness, a host fixed-point theorem, unrestricted compactness, or selected representatives. Therefore E seals only the displayed fixed-point name of this packet. □

Falsifiable prediction: An accepted $\mathsf{BishopFixedPointInterval}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ row E without first reading $I, T, M, B, R, H, C, P, N$ refutes the carrier surface.

Independence witness: locatedinterval, intervalbisection, cauchymodulus, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the full interval self-map packet $F = (I, T, M, B, R, E, H, C, P, N$ because each sibling contributes only a located interval source, a bisection route, a modulus row, a readback row, a finite name window, a dyadic ledger, or a terminal real seal while this chapter binds them into one fixed-point NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopFixedPointInterval}^{\uparrow}$ packet is generated from a located interval row, contraction self-map row, explicit Cauchy or Lipschitz modulus row, nested bisection row, regular rational readback row, terminal Real seal, componentwise transport, displayed replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5811.1, Definition 5811.2, Definition 5811.3, and Definition 5811.4 seed the finite constructive interval fixed-point route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopFixedPointInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point compactness, quotient real equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction self-map admission, explicit modulus routing, nested interval readback, terminal Real seal nonescape, and a Lean carrier/obligation formalization for BEDC.Derived.BishopFixedPointIntervalUp.

Chapter 5812

A Concrete Naming Certificate for $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$

The $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet realizes the private-coordinate minimal-cover theorem of Definition 9462.1678 as finite NameCert data. It records a trigger-coordinate carrier, a support-family cover ledger, and one private-coordinate witness row for each displayed support before any minimal-trigger criterion may be consumed. The packet keeps the enumeration theorem auditable without identifying it with a generic finite set, word list, Boolean classifier, regular-language recognizer, or finite automaton presentation.

Falsifiable prediction: Every accepted $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ consumer must display a cover row and one private-coordinate witness row for each support before using the minimal-trigger criterion.

Independence witness: finset, list, bool, regularlanguage, finite-automaton: $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ is not bijective to these siblings because it carries a support-family cover ledger plus a private-coordinate witness for every support row.

Definition 5812.1 (Private-coordinate trigger carrier)**.** *A* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the finite trigger-coordinate row.*
2. A *is the finite support-family row whose entries are displayed subsets of* Ω*.*
3. U *is the cover ledger asserting that the displayed supports cover every coordinate of* Ω*.*
4. V *is the private-coordinate witness ledger, with one row for each support in* A*.*
5. π *maps each support row to its selected private coordinate and records membership in that support.*
6. E *records exclusion from every other support row for the selected private coordinate.*
7. K *is the minimal-cover criterion row, readable only after* U, V, π, E *have been displayed.*
8. H *stores componentwise* hsame *transport for coordinate, support, cover, witness, exclusion, and criterion rows.*

9. *C stores* Cont *replay from a trigger query through the cover ledger and the private-coordinate witness rows.*
10. *P stores* Pkg *provenance over* BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $\Omega, A, U, V, \pi, E, K, H, C, P, N$. It has no coordinate for a bare finite set, unordered list, Boolean truth table, regular-language recognizer, finite automaton run, or support family without displayed private witnesses.

Theorem 5812.2 (Private-coordinate trigger NameCert obligations)**.** *For every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier*

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$ *surface consists exactly of coordinate exposure, support-family exposure, cover-ledger replay, one private-coordinate witness row for each support, membership and exclusion checks for the selected witnesses, minimal-cover criterion exposure, componentwise transport, continuation replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5812.1. The rows Ω, A, U display the finite coordinate space, the support family, and the fact that the supports cover the full coordinate row.

The rows V, π, E attach a private coordinate to each support. Membership is recorded by π, and exclusion from every other support is recorded by E, so deleting the support removes at least one displayed coordinate from the cover.

The criterion row K is therefore readable only after the cover and private-witness rows have been exposed. The remaining rows H, C, P, N preserve transport, replay, provenance, and local naming for the same finite packet, and no omitted sibling surface supplies the private-coordinate ledger. □

Theorem 5812.3 (Private-coordinate minimal-cover criterion)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *carrier exposes a minimal full trigger exactly through the displayed cover row U together with the private-coordinate witness rows V, π, E. A consumer may use the criterion row K only after reading a cover of Ω and one witness coordinate for every support row in A that belongs to that support and belongs to no other displayed support. The packet does not turn the support family into a regular language, finite automaton, Boolean table, or list quotient.*

Proof. The forward read uses the cover ledger U and the private-coordinate witness rows. If a support row is removed, its selected witness coordinate remains outside all other displayed supports by E, so the remaining family no longer covers Ω.

For the reverse read, the cover row already gives full coverage. The witness rows ensure that every support has a coordinate lost by deleting that support, so no displayed support is redundant.

Thus the criterion row K is exactly the finite private-coordinate minimal-cover test. The transport, replay, provenance, and local naming rows carry this test without converting it into any sibling classifier. □

Theorem 5812.4 (Private-coordinate trigger finite-set cover route)**.** *Every accepted* $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ *packet reads its support-family row as a finite-set-indexed cover through chapter 112. The exclusions from regular-language and automaton presentations are bounded by chapter 351 and chapter 2364: the private-coordinate carrier may display finite support rows, recognizer-like acceptance tests, or transition-like exclusion checks, but it exports the minimal-cover criterion only through the rows $\Omega, A, U, V, \pi, E, K, H, C, P, N$.*

Proof. Project the accepted carrier

$$T = (\Omega, A, U, V, \pi, E, K, H, C, P, N).$$

The row Ω is the displayed coordinate object, and A is read as a finite family of supports indexed by the finite-set surface. The cover ledger U then witnesses coverage of every coordinate in Ω by those displayed supports.

The rows V, π, E are the extra private-coordinate ledger. They attach one witness coordinate to each support, record membership in that support, and record exclusion from every other displayed support. These rows are not supplied by the finite-set sibling, nor by the regular-language or finite-automaton presentations.

Consequently K is a private-coordinate minimal-cover read rather than a bare finite-set cover, a regular-language recognizer, or an automaton acceptance predicate. The structural rows H, C, P, N transport and replay only this displayed packet, so the sibling route records the finite-combinatorial surface without replacing the private-coordinate data. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}$.

Constructive story: A $\mathsf{PrivateCoordinateTrigger}^{\uparrow}$ packet is generated from a finite trigger-coordinate row, support-family row, cover ledger, per-support private-coordinate witness ledger, membership and exclusion rows, minimal-cover criterion row, componentwise transport, continuation replay, provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5812.1, Definition 5812.2, Definition 5812.3, and Definition 5812.4 seed the finite NameCert packet for private-coordinate minimal-cover witnesses.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateCoordinateTrigger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, regular-language recognition, finite-automaton minimization, Boolean classifier completeness, or enumeration beyond the displayed private-coordinate criterion.

Upgrade path: Obligation closure requires checked carrier admission, cover-ledger replay, private-coordinate witness coverage, exclusion exactness, minimal-cover criterion replay, TasteGate nontriviality, FieldFaithful projection, transport, replay, provenance, and local naming for BEDC.Derived.PrivateCoordinateTriggerUp.

Chapter 5813

A Concrete Naming Certificate for $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet records the finite modulus by which a located real comparison request is read through displayed apartness or equality evidence, dyadic tolerances, explicit Cauchy-modulus schedules, regular rational readback, and the terminal real-name seal. It sits after the comparison surface of chapter 1730, the tail envelope of chapter 1769, the Cauchy-modulus bridge of chapter 5728, the rational readback of chapter 301, and the public real route of chapter 38. The packet names a finite comparison-schedule carrier; it is not trichotomy, quotient real equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5813.1 (Located real comparison modulus carrier)**.** *A* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, A, D, B, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{LocatedRealComparison}^{\uparrow}$ *request row for the two located real endpoints and the comparison direction being inspected.*
2. *A is the located evidence row. It records the displayed positive apartness witness when the request separates the endpoints, or the displayed equality-facing refusal row when the request reaches only an overlap tolerance.*
3. *D is the dyadic tolerance schedule. It lists the requested precision, comparison slack, endpoint radii, and finite tolerance refinements consumed by the comparison read.*
4. *B is the* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *route connecting the comparison request to an explicit Cauchy-modulus schedule before regular rational readback is allowed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected rational tails exposed by B and bounded by D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the comparison-facing read, available only after Q, A, D, B, R have been displayed.*
7. *H records componentwise* hsame *transport for the comparison request, evidence row, dyadic tolerance schedule, bridge route, readback, Real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from the comparison request through located evidence, dyadic scheduling, Cauchy-modulus bridging, regular rational readback, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$, $\mathsf{LocatedRealComparison}^{\uparrow}$, LocatedRealTailEnvelop $\mathsf{CauchyModulusBridge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, A, D, B, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, a hidden selector, countable choice, a host comparison oracle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedRealComparisonModulusUp.LocatedRealComparisonModulusTasteGate`
`single carrier alignment`

Theorem 5813.2 (Located real comparison modulus obligations)**.** *For every accepted* LocatedRealComparisonModu *carrier*

$$M = (Q, A, D, B, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$ *surface consists exactly of comparison-request admission, located apartness-or-equality evidence, dyadic tolerance scheduling,* $\mathsf{CauchyModulusBridge}^{\uparrow}$ *routing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any consumer of the comparison schedule must read those rows in the displayed order and receives no trichotomy split, quotient real equality, hidden selector, countable-choice schedule, host comparison oracle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Project the carrier of Definition 5813.1. The comparison request is Q, and the only evidence row attached to it is A.

The schedule is then forced through D and B. The dyadic row D records the finite tolerances to be inspected, while the $\mathsf{CauchyModulusBridge}^{\uparrow}$ row B turns those tolerance requests into explicit tail thresholds before any rational readback is exposed.

The terminal read is downstream. The regular rational row R reads only the displayed tails opened by B, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same rows. Since the carrier has no coordinate for the excluded host-level or choice-level objects, the listed rows exhaust the local NameCert surface. □

Theorem 5813.3 (Located real comparison modulus nontrichotomy boundary)**.** *Every accepted* $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ *packet exports a positive schedule for comparison reads without exporting trichotomy. If the evidence row A records positive apartness, a downstream consumer may read that apartness only through Q, D, B, R, E; if A records only an equality-facing overlap tolerance, the consumer receives the displayed tolerance route and no forced left-or-right order verdict.*

Proof. Read the evidence row A together with the comparison request Q. The carrier does not contain a three-way order coordinate, so the two possible reads are positive displayed apartness or finite overlap tolerance.

For the positive branch, the dyadic tolerance row D and bridge row B determine which tails are inspected before R and E can be consumed. The apartness read is therefore a finite schedule read, not a global order split.

For the overlap branch, the same schedule exposes only the inspected tolerance data. Since no trichotomy coordinate or hidden selector appears in $M = (Q, A, D, B, R, E, H, C, P, N)$, the packet refuses any forced left-or-right verdict not displayed by the rows. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is refuted if a comparison consumer reaches the $\mathsf{Real}^{\uparrow}$ seal while omitting the $\mathsf{LocatedRealComparison}^{\uparrow}$ request row Q, the evidence row A, the dyadic tolerance schedule D, the $\mathsf{CauchyModulusBridge}^{\uparrow}$ route B, or the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R.

Independence witness: LocatedRealComparison, LocatedRealTailEnvelope, CauchyModulusBridge, RegSeqRat, Real: none is bijective with $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ because the packet jointly carries a comparison request, apartness-or-equality evidence, dyadic tolerance scheduling, explicit Cauchy-modulus bridging, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealComparisonModulus}^{\uparrow}$ packet is generated from a LocatedRealComparison request row, located apartness-or-equality evidence, a dyadic tolerance schedule, a CauchyModulusBridge route, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5813.1, Definition 5813.2, and Definition 5813.3 seed the finite comparison-modulus route over LocatedRealComparisonUp, LocatedRealTailEnvelopeUp, CauchyModulusBridgeUp, RegSeqRatUp, and RealUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealComparisonModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, selected representatives, countable choice, host comparison oracles, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, evidence-row exactness, dyadic schedule determinacy, Cauchy-modulus bridge routing, regular rational readback, Real-seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealComparisonModulusUp.

Chapter 5814

A Concrete Naming Certificate for $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$

The $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet realizes the boundary-shadow and observer-expansion route of chapter 9463 as finite cocycle-level NameCert data. It sits after the boundary-shadow route of chapter 3813, the observer-shadow metric of chapter 4154, the Busemann cocycle of chapter 4134, the horosphere-flow ledger of chapter 4051, and the observer-expansion packet of chapter 4167. The chapter names a finite expansion cocycle over displayed boundary and horosphere rows; it is not a completed hyperbolic boundary, quotient geodesic ray space, physical observer dynamics, selected ideal point, or ambient manifold theorem.

Definition 5814.1 (Boundary-shadow expansion cocycle carrier)**.** *A* $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, O, B, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row recording the visible boundary shadow through which the cocycle is read.*
2. *O is the* $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$ *row recording the observer-facing shadow metric comparison for the same boundary read.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row recording the additive Busemann comparison consumed by the expansion read.*
4. *F is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row recording the horosphere-flow level through which boundary and Busemann rows are replayed.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *row recording the displayed observer-expansion magnitude or direction attached to S, O, B, F.*
6. *H records componentwise* hsame *transport for boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow ledger, observer expansion, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay through boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, and observer-expansion read.*

8. *P is the* Pkg *provenance over* BoundaryShadowExpansionCocycle$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLedger$^{\uparrow}$, HyperbolicObserverExp BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, O, B, F, E, H, C, P, N$. It has no coordinate for completed boundary points, quotient geodesic rays, a hidden expansion selector, physical observer dynamics, selected ideal endpoints, or ambient hyperbolic manifold data.

Theorem 5814.2 (Boundary-shadow expansion cocycle NameCert obligations)**.** *For every accepted* BoundaryShadowExpansionCocycle$^{\uparrow}$ *carrier*

$$X = (S, O, B, F, E, H, C, P, N),$$

the local NameCert$_{\text{BoundaryShadowExpansionCocycle}^{\uparrow}}$ *surface consists exactly of boundary-shadow admission, observer-shadow metric readback, Busemann cocycle comparison, horosphere-flow ledger inspection, observer-expansion read, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Any accepted expansion read that omits either the boundary-shadow row S or the Busemann cocycle row B is outside the packet.*

Proof. Project the carrier of Definition 5814.1. The boundary-facing rows are S and O, while the cocycle and horosphere-flow rows are B and F.

The expansion row E is downstream from those four rows. It can be read only after the visible boundary shadow, observer-shadow metric, Busemann cocycle, and horosphere-flow ledger have all been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for completed boundary data, quotient geodesic rays, hidden selectors, physical observer dynamics, selected endpoints, or ambient manifold structure, the listed rows exhaust the local NameCert surface. □

Theorem 5814.3 (Boundary-shadow expansion cocycle dependency route)**.** *Every accepted* BoundaryShadowExpan *read factors through*

HyperbolicBoundaryShadow$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicBusemannCocycle$^{\uparrow}$, HorosphereFlowLed

before a consumer may inspect the expansion cocycle as a local NameCert row. The route is finite and preserves the boundary-shadow row and Busemann cocycle row as separate coordinates.

Proof. Use Definition 5814.2. It places the boundary-shadow row, observer-shadow metric row, Busemann cocycle row, and horosphere-flow ledger before the expansion row.

The cocycle dependency is not reducible to one sibling. The boundary-shadow row supplies the boundary face, the observer-shadow metric supplies the observer-facing comparison, the Busemann row supplies the additive cocycle read, and the horosphere-flow row supplies the level ledger.

Only after those rows are present can E, H, C, P, N expose the expansion cocycle as a local packet. Therefore a consumer cannot collapse the route to a single hyperbolic object or bypass the separate boundary-shadow and Busemann coordinates. □

Falsifiable prediction: Any accepted BoundaryShadowExpansionCocycle$^{\uparrow}$ packet is refuted if the expansion read omits either the boundary-shadow row S or the Busemann cocycle row B.

Independence witness: hyperbolic_boundary_shadow, hyperbolic_observer_shadow_metric, hyperbolic_busemann_cocycle, horosphere_flow_ledger, hyperbolic_observer_expansion: none

is bijective with $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ because the packet jointly carries boundary shadow, observer-shadow metric, Busemann cocycle, horosphere-flow level, observer expansion, transport, replay, provenance, and local naming.

The Lean TasteGate route for this packet must provide `Nontrivial BoundaryShadowExpansionCocycleUp` and `FieldFaithful BoundaryShadowExpansionCocycleUp` instances in `lean4/BEDC/Derived/BoundaryShadowE` Those instances must preserve $S, O, B, F, E, H, C, P, N$ as distinct fields before a checked carrier-alignment theorem can serve as the TasteGate witness.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}$ packet is generated from HyperbolicBoundaryShadow rows, HyperbolicObserverShadowMetric rows, HyperbolicBusemannCocycle rows, HorosphereFlowLedger rows, HyperbolicObserverExpansion rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5814.1, Definition 5814.2, and Definition 5814.3 seed the finite boundary-shadow expansion cocycle route over boundary, observer-shadow, Busemann, horosphere-flow, expansion, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundaryShadowExpansionCocycle}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic boundaries, quotient geodesic ray spaces, physical observer dynamics, selected ideal endpoints, ambient manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow exactness, observer-shadow metric readback, Busemann cocycle separation, horosphere-flow routing, observer-expansion exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.BoundaryShadowExpansionCocycleUp.

The $\mathsf{HellyBraySelection}^{\uparrow}$ hub routes bounded-variation selection through chapter 398, chapter 2069, chapter 301, chapter 126, and chapter 38 before the child files expose the finite selection packet.

Chapter 5815

A Concrete Naming Certificate for $\mathsf{HellyBraySelection}^{\uparrow}$

Following Helly and Bray, the $\mathsf{HellyBraySelection}^{\uparrow}$ packet records the finite selection surface by which bounded-variation source data are read through Jordan signed-variation rows, finite stream windows, regular rational readback, and terminal real seals before a downstream selection consumer may use the packet. It sits after the bounded-variation carrier of chapter 398, the Jordan variation decomposition rows of chapter 2069, the regular rational readback route of chapter 301, the finite observation windows of chapter 126, and the terminal real-name boundary of chapter 38. Helly–Bray selection also feeds distribution-facing compactness through chapter 186, chapter 92, and chapter 4008 when the bounded-variation read is consumed as a tight probability-family route. The chapter names a finite BEDC selection packet, not host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5815.1 (Helly–Bray selection carrier)**.** *A* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (B, J, S, R, E, T, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, carrying the bounded-variation object, finite variation ledger, and endpoint budget exposed to the selection request.*
2. *J is the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *row, displaying the signed-variation split and the positive and negative variation rows read from B.*
3. *S is the family of* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened by the inspected variation rows and by the downstream Real-facing request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows, recording the rational comparisons used before any Real seal is visible.*
5. *E is the finite family of* $\mathsf{Real}^{\uparrow}$ *seal rows reached only after B, J, S, R have been displayed.*
6. *T records componentwise* hsame *transport for the bounded-variation source, Jordan rows, finite windows, readbacks, Real seals, continuation, provenance, and local naming.*
7. *C records displayed* Cont *replay from bounded-variation data through Jordan signed-variation rows, finite windows, regular rational readback, Real seals, transport, provenance, and local naming.*

8. *P is the* Pkg *provenance over* $\mathsf{HellyBraySelection}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{JordanVariationDecomposition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, J, S, R, E, T, C, P, N$. The carrier has no coordinate for host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5815.2 (Helly–Bray selection NameCert obligations)**.** *For every accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier*

$$H = (B, J, S, R, E, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded-variation source row, Jordan signed-variation decomposition row, StreamName finite windows, RegSeqRat readback row, Real seal rows, transport, replay, provenance, and local naming;*
2. *variation-source exposure, requiring every selection-facing read to begin with B and then pass through the signed-variation rows J rather than through a host weak-convergence theorem;*
3. *finite-window locality, requiring S to list the exact stream windows opened by the variation rows and the Real-facing request;*
4. *readback ordering, requiring R to consume the displayed windows in S before any row of E is exposed;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same rows through T, C, P, N;*
6. *nonescape, excluding host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Helly–Bray selection obligation is stored outside these rows.

Proof. Project the accepted carrier. The only source coordinate is the bounded-variation row B, and the only signed-variation coordinate is the Jordan row J.

The finite observation surface is then forced by row order. The StreamName windows S are opened from the displayed variation rows, the RegSeqRat readback R reads only those windows, and the Real seals E appear only after that readback has been displayed.

Transport and replay preserve the same route componentwise through T and C. Provenance P and local naming N record that route as the local certificate surface.

Since none of the excluded objects occurs as a carrier coordinate, a consumer cannot add weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, or ambient Real completeness to the local NameCert obligations. □

Theorem 5815.3 (Helly–Bray selection finite-window route)**.** *Every Real-facing or variation-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier factors its read through*

$$\mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{JordanVariationDecomposition}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto T, C, P, N$$

Thus the consumer receives bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise transport, continuation replay, provenance, and local naming. The route supplies no host weak convergence, measure completion, quotient function equality, selected subsequence, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier $H = (B, J, S, R, E, T, C, P, N)$. The first projection gives the bounded-variation source row B, and the next projection gives the Jordan signed-variation row J over that source.

The finite-window part is local. The windows S are the only observation rows available to the packet, and R is the only regular rational readback attached to those windows.

The Real-facing boundary is reached only at E. The remaining rows T, C, P, N transport, replay, source, and name the same finite route; they do not add a new selection principle.

Because the carrier omits host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, and ambient Real completeness, every accepted consumer read is exhausted by the displayed finite-window route. □

Theorem 5815.4 (Helly–Bray distribution forward link)**.** *Every* $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *or* $\mathsf{ProkhorovTightProbabi}$ *consumer that cites an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *carrier must first read the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row, the* $\mathsf{JordanVariationDecomposition}^{\uparrow}$ *signed-variation row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal rows of Definition 5815.1. The downstream route to chapter 186, chapter 92, and chapter 4008 therefore receives only bounded-variation source data, Jordan variation data, displayed windows, rational readback, Real seals, componentwise transport, replay, provenance, and local naming. It receives no host weak convergence theorem, measure completion, quotient equality, selected subsequence, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier $H = (B, J, S, R, E, T, C, P, N)$. The only distribution-facing source row is B, and the only signed-measure decomposition row available to the packet is J.

Use the finite-window route of Definition 5815.3. It forces every Real-facing and variation-facing read through S, R, E before the transport, replay, provenance, and local naming rows can be consumed.

Now attach a downstream distribution, measure, or tight-probability request. Since those consumers cite the Helly–Bray packet only through the displayed route, the request reaches chapter 186, chapter 92, or chapter 4008 after $B, J, S, R, E, T, C, P, N$ have already been displayed.

The excluded objects are not carrier coordinates. Thus the forward link cannot export host weak convergence, measure completion, quotient equality, selected subsequences, countable choice, or ambient Real completeness to any distribution-facing consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5815.1, Definition 5815.2, Definition 5815.3, and Definition 5815.4 seed the finite Helly–Bray selection route over bounded-variation data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, local naming, and distribution-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HellyBraySelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host weak convergence, measure completion, quotient function equality, selected subsequences, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, variation-source exposure, finite-window locality, readback ordering, transport, replay, provenance, local naming, and finite-window routing for BEDC.Derived.HellyBraySelectionUp.

Constructive story: A $\mathsf{HellyBraySelection}^{\uparrow}$ packet is generated from bounded-variation source data, Jordan signed-variation rows, finite StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The Helly–Bray route is a distribution-facing read of the finite selection surface already named by chapter 1732; its bounded real sequence and compactness rows are supplied through chapter 1415 before the distribution consumer is allowed to inspect the signed-variation packet.

Theorem 5815.5 (Helly–Bray Helly-selection dependency route)**.** *Every* $\mathsf{Distribution}^{\uparrow}$ *or compactness-facing consumer of an accepted* $\mathsf{HellyBraySelection}^{\uparrow}$ *packet must first expose the finite* $\mathsf{HellySelection}^{\uparrow}$ *selection surface of chapter 1732 together with the bounded real sequence route of chapter 1415. In carrier notation, the read passes through the bounded-variation source row, Jordan signed-variation row, finite window rows, regular rational readback, terminal Real seals, componentwise transport, replay, provenance, and local naming rows before the distribution-facing route of Definition 5815.4 is public. It is therefore not a free weak-convergence theorem, measure-completion theorem, quotient-function equality, choice-selected subsequence, or ambient compactness principle.*

Proof. Start with the carrier of Definition 5815.1. Its only sequence-facing data are the bounded-variation row, Jordan signed-variation row, finite windows, regular rational readback, and terminal Real seals.

Read those rows through the finite-window route of Definition 5815.3. The same bounded source discipline is the one exported by the $\mathsf{HellySelection}^{\uparrow}$ packet, and the bounded real sequence route supplies the finite window and bound rows before any compactness-facing consumer can inspect them.

Finally attach the distribution forward link. Since Definition 5815.4 exposes distribution consumers only after the Helly–Bray carrier rows have been displayed, the downstream route receives the Helly-selection and bounded-sequence surfaces as dependencies rather than as hidden host compactness or choice principles. □

Chapter 5816

A Concrete Naming Certificate for DiniFiniteMinimumBudget$^{\uparrow}$

Following Dini's theorem and Bishop–Bridges constructive compactness, the DiniFiniteMinimumBudget$^{\uparrow}$ packet records the finite extraction row that turns a monotone compact-window Dini certificate into one displayed tolerance budget. It consumes the CompactMetric$^{\uparrow}$ source, monotone ContinuousMap$^{\uparrow}$ family rows, the DiniTheorem$^{\uparrow}$ compact-window route, UniformModulus$^{\uparrow}$ handoff data, DyadicRatCore$^{\uparrow}$ tolerance cells, the terminal Real$^{\uparrow}$ seal, BHist history rows, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data. The packet names only the finite minimum-budget route; it is not host supremum selection, a selected minimum outside the carrier, open-cover compactness, quotient equality, or ambient Real$^{\uparrow}$ completeness.

The dependency surface is horizontal: chapter 1387 supplies the compact-window Dini route, chapter 129 supplies the inspected compact carrier, chapter 130 supplies the monotone continuous-family rows, and chapter 369 receives the extracted budget. Thus the minimum-budget row is a downstream finite extraction from those sibling carriers, not an isolated tolerance selector.

Definition 5816.1 (Dini finite-minimum budget carrier)**.** *A* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, D, U, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the finite compact-metric windows inspected by the extraction.*
2. *F is the monotone* ContinuousMap$^{\uparrow}$ *family row, including the displayed pointwise comparison ledger used by the Dini route.*
3. *D is the* DiniTheorem$^{\uparrow}$ *row that supplies the monotone uniform-window certificate over K and F.*
4. *U is the* UniformModulus$^{\uparrow}$ *handoff row consumed by uniform-limit and compact-uniform-modulus readers.*
5. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row, listing the candidate dyadic budgets and their finite comparison cells.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached by every displayed tolerance and value comparison.*
7. *H records componentwise* hsame *transport for $K, F, D, U, Q, E, C, P, N$.*

8. *C records displayed* Cont *replay from compact windows and monotone family rows through the Dini row, the finite dyadic budget comparisons, the uniform-modulus handoff, and the Real seal.*
9. *P is the* Pkg *provenance over* DiniFiniteMinimumBudget$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DiniTheorem$^{\uparrow}$, UniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for an ambient supremum, an externally selected minimum, open-cover compactness, quotient equality of real names, countable choice, or host Real completeness.

Lean4 ✓ `BEDC.Derived.DiniFiniteMinimumBudgetUp.DiniFiniteMinimumBudgetTasteGate single carrier alignment`

Theorem 5816.2 (Dini compact finite-minimum budget)**.** *For every accepted* DiniFiniteMinimumBudget$^{\uparrow}$ *carrier*

$$B = (K, F, D, U, Q, E, H, C, P, N),$$

the finite probe cover displayed by K, F, D and the dyadic tolerance cells displayed by Q determine one displayed minimum tolerance budget before the uniform-modulus handoff U is exported. The extraction route is

$$K, F, D, Q, E, H, C, P, N \longrightarrow U.$$

Thus a UniformLimit$^{\uparrow}$ *or compact-uniform-modulus consumer receives a finite* DyadicRatCore$^{\uparrow}$ *budget sealed by* Real$^{\uparrow}$ *and transported by* hsame *and* Cont*, without host supremum, selected minimum outside the carrier, open-cover compactness, quotient equality, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the accepted packet. The compact source row K, monotone family row F, and Dini row D are the only rows that can name the finite probe cover for the tolerance request.

The dyadic row Q lists the candidate tolerance cells. Since the carrier is finite, the budget comparison is a displayed finite comparison among the cells in Q, with every real-valued read sealed through E before the handoff row is exposed.

The route to U is then replayed by C and transported componentwise by H. The provenance and naming rows P, N record the same finite route as the local certificate surface.

No excluded object is a coordinate of B. Consequently the exported budget is the displayed finite minimum-budget row inside the carrier, not an ambient supremum, external selector, open-cover argument, quotient equality, choice schedule, or completeness argument. □

Falsifiable prediction: Every accepted DiniFiniteMinimumBudget$^{\uparrow}$ consumer that exports a UniformModulus$^{\uparrow}$ handoff must first expose the compact-metric windows, monotone continuous-family rows, DiniTheorem route, finite DyadicRatCore$^{\uparrow}$ budget cells, Real$^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: dini-theorem, compactmetric, continuousmap, uniformmodulus, dyadicratcore, real: DiniFiniteMinimumBudget$^{\uparrow}$ is not bijective to any listed sibling because it binds the finite Dini probe cover to one displayed dyadic minimum-budget handoff before uniform-limit or compact-uniform-modulus consumers may read the result.

Closure status for NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, seed)

Scope closed: Definition 5816.1 and Definition 5816.2 seed the finite minimum-budget extraction route from compact Dini data to a uniform-modulus-facing handoff.

Formal verification: FormalStatus(NameCert$_{\text{DiniFiniteMinimumBudget}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host supremum selection, selected minima outside the carrier, open-cover compactness, quotient equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite probe-cover exposure, dyadic minimum-budget extraction, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.DiniFiniteMinimumBudgetUp.

Constructive story: A DiniFiniteMinimumBudget$^{\uparrow}$ packet is generated from compact-metric source rows, monotone ContinuousMap family rows, a DiniTheorem compact-window route, UniformModulus handoff rows, DyadicRatCore tolerance cells, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5817

A Concrete Naming Certificate for $\mathsf{FiniteCauchyFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteCauchyFilter}^{\uparrow}$ packet records the finite directed-tail surface used when a completion consumer asks for Cauchy-filter evidence before reading a regular rational name or a terminal real-name seal. It sits beside the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite filter basis for tail requests; it is not a quotient filter, a selected limit, countable choice, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5817.1 (Finite Cauchy filter carrier)**.** *A* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (W, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the filter request.*
2. T *is the directed-tail request row, naming the finite family of later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii and comparison budgets used by the directed tails.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and tolerance ledger.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, T, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the stream windows, tail requests, dyadic ledgers, regular readback, and real seal.*
7. C *records displayed* Cont *replay of the same finite filter route.*
8. P *records* Pkg *provenance over* $\mathsf{FiniteCauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.FiniteCauchyFilterUp.FiniteCauchyFilterTasteGate single carrier alignment`

Theorem 5817.2 (Finite Cauchy filter NameCert obligations)**.** *For the carrier of Definition 5817.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^\uparrow$ *window exposure, directed-tail request visibility,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance-ledger preservation,* $\mathsf{RegSeqRat}^\uparrow$ *readback,* $\mathsf{Real}^\uparrow$ *seal access, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $F = (W, T, D, R, E, H, C, P, N)$. The rows W, T, D expose the finite source windows, directed tail requests, and dyadic tolerance budgets before any regular-rational or real-facing read is admitted.

The readback row R is then determined by the displayed windows and tolerance ledger. Since the carrier stores no second regular-rational coordinate, every $\mathsf{RegSeqRat}^\uparrow$ consumer reads the same finite route.

The seal row E is downstream of W, T, D, R. The transport row H, continuation row C, package provenance row P, and local naming row N preserve exactly that route and provide no coordinate for quotient filters, selected limits, countable choice, host stream equality, host real equality, or ambient completeness. □

Theorem 5817.3 (Finite Cauchy filter Real route)**.** *Every* $\mathsf{Real}^\uparrow$*-facing consumer of an accepted* $\mathsf{FiniteCauchyFilter}^\uparrow$ *carrier factors through the finite route*

$$W, T, D, R, E, H, C, P, N.$$

In particular, the consumer must first read $\mathsf{StreamName}^\uparrow$ *windows, directed tail requests,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance ledgers, and the* $\mathsf{RegSeqRat}^\uparrow$ *readback before the terminal* $\mathsf{Real}^\uparrow$ *seal is consumed.*

Proof. Start with the carrier definition. The only real-facing row is E, and it is displayed after the stream windows, directed-tail row, dyadic ledger, and regular readback row.

Apply Definition 5817.2. The local NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Thus any Real-facing consumer receives the finite packet $W, T, D, R, E, H, C, P, N$ and no quotient-filter, selected-limit, countable-choice, host-equality, or ambient-completeness substitute. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$.

Constructive story: A $\mathsf{FiniteCauchyFilter}^\uparrow$ packet is generated from finite StreamName windows, directed tail requests, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, seed)

Scope closed: Definition 5817.1, Definition 5817.2, and Definition 5817.3 specify the finite directed-tail filter carrier and its Real-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteCauchyFilter}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter construction, selected limits, countable choice, host stream equality, host real equality, ambient completeness of $\mathsf{Real}^\uparrow$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for directed-tail visibility, tolerance-ledger preservation, readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5818

A Concrete Naming Certificate for DNAWindowSixTranslationBoundary$^{\uparrow}$

The DNAWindowSixTranslationBoundary$^{\uparrow}$ packet realizes the local translation-boundary branch of subsection 9462.28.21 as finite NameCert data. It consumes the codon-window and stop-tile boundary analysis in the source vision, the curated code-boundary ledger of **??**, and the trigger-threshold surface of chapter 4217. The packet names only a finite window-six translation-boundary carrier; biological translation realization, universality, protein function, and evolutionary mechanism remain outside the BEDC kernel packet.

Definition 5818.1 (DNA window-six translation-boundary carrier)**.** *A* DNAWindowSixTranslationBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (Q, S, T, L, R, V, H, C, P, N)$$

with the following displayed rows:

1. *Q is the codon Q_6 window row recording the six-bit coordinate readback for the inspected codon window.*
2. *S is the stop-tile boundary row recording the local halt boundary and continuation-corner split from the source vision.*
3. *T is the translation-classifier row supplied by external biological bridge data.*
4. *L is the curated code-boundary ledger row inherited from* BioRealityQSixLeaveOneCodeBoundaryLedger$^{\uparrow}$*.*
5. *R is the release-factor or release-mechanism row that records which external recognition boundary is being tested.*
6. *V is the finite validation row recording which observed table, organismal code row, or assay class is used for the test.*
7. *H records componentwise* hsame *transport for the window, tile, classifier, ledger, release, and validation rows.*
8. *C records displayed* Cont *replay of the boundary read through those rows.*
9. *P records* Pkg *provenance over the source vision, the bio-reality ledger sibling,* DNAWindowSixTriggerThreshold$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for biological necessity, global identification of the genetic code with the window-six fold, protein-folding realization, universal stop assignment, hidden BEDC channel tokens, or untested mechanism.

Theorem 5818.2 (DNA window-six translation-boundary obligations)**.** *For the carrier of Definition 5818.1, the local* $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$ *surface consists exactly of codon-window exposure, stop-tile boundary visibility, external translation-classifier disclosure, curated code-ledger preservation, release-boundary test exposure, finite validation-row visibility, componentwise transport, displayed continuation replay, provenance, and local naming.*

Proof. Project the accepted packet $B = (Q, S, T, L, R, V, H, C, P, N)$. The rows Q, S, T expose the finite codon window, the local translation boundary, and the external classifier before any biological reading is admitted.

The ledger and release rows L, R keep the packet anchored to the bio-reality boundary sibling and to a displayed recognition test. The validation row V records the observed table or assay class being tested, so the packet has a falsifiable contact surface rather than a kernel claim of biological necessity.

The structural rows H, C, P, N transport, replay, source, and name exactly the displayed boundary route. Since no further coordinate is present, the carrier cannot export global window-six identity, universal stop assignment, protein-folding realization, hidden channel tokens, or untested mechanism. □

Theorem 5818.3 (DNA window-six translation-boundary sibling handoff)**.** *Every accepted* DNAWindowSixTranslat *read that reaches a translation-boundary consumer must first expose the source-vision stop-tile row of subsection 9462.28.21, the curated code-boundary ledger row of* **??**, *and the finite trigger-threshold row of chapter 4217. The handoff is therefore a concrete finite boundary-tile NameCert route, not a narrative-only direction.*

Proof. Use Definition 5818.1. The source-vision row is carried by S, the code-boundary ledger is carried by L, and the trigger-facing finite boundary read is recorded in the same package provenance P.

Now apply Definition 5818.2. The local NameCert requires those rows before the translation-boundary consumer can read N. Hence the consumer sees the source vision, the bio-reality ledger sibling, and the trigger-threshold sibling as displayed packet data rather than as detached prose. □

Falsifiable prediction: An accepted $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet must expose a finite validation row naming the observed code table, organismal code row, or release-factor assay class before it can export a translation-boundary reading.

Independence witness: bioreality-q-six-leave-one-code-boundary-ledger, dna-window-six-trigger-threshold, synonymous-transition: the $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is not bijective to any listed sibling because it jointly carries codon-window, stop-tile, external translation-classifier, curated code-ledger, release-boundary, validation, transport, replay, provenance, and local naming rows.

The Lean follow-up must provide $\mathsf{Nontrivial}\ \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{DNAWindowSi}$ TasteGate instances before this packet can move beyond seed closure.

Closure status for $\mathsf{NameCert}_{\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$ packet is generated from codon Q-six window rows, stop-tile boundary rows, external translation-classifier rows, curated code-boundary ledgers, release-boundary tests, finite validation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, seed)

Scope closed: Definition 5818.1, Definition 5818.2, and Definition 5818.3 specify the finite translation-boundary carrier and sibling route.

Formal verification: FormalStatus(NameCert$_{\mathsf{DNAWindowSixTranslationBoundary}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim biological-code necessity, global window-six identity, universal stop assignment, protein-folding realization, hidden BEDC channel tokens, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding, TasteGate nontriviality, field faithfulness, validation-row exactness, sibling-route preservation, transport, replay, provenance, and local naming.

Chapter 5819

A Concrete Naming Certificate for $\mathsf{RegulatedConvergence}^{\uparrow}$

Following Bishop and Bridges on regulated functions and uniform convergence, the $\mathsf{RegulatedConvergence}^{\uparrow}$ packet records the finite route by which a displayed sequence of regulated real functions converges uniformly to a regulated endpoint. It sits after the regulated-function carrier of chapter 1906, the regulated-integral consumer boundary of chapter 1557, the uniform-convergence ledger of chapter 3417, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite convergence-and-handoff surface; it does not import measure completion, quotient equality of functions, selected host limits, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5819.1 (Regulated convergence carrier)**.** *A* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the source row naming the regulated-function family through* $\mathsf{RegulatedFunction}^{\uparrow}$ *data, including the displayed interval, step-window, endpoint, and oscillation ledgers available to each inspected member.*
2. *U is the* $\mathsf{UniformConvergence}^{\uparrow}$ *tail row, recording one shared tolerance schedule for the whole regulated family rather than pointwise schedules chosen separately at each argument.*
3. *W is the finite window row exposing exactly the function indices, domain windows, and endpoint reads used by the shared tail schedule.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the real-valued observations in those windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after F, U, W, R have been displayed.*
6. *I is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row; it records that integral-facing consumers may read the regulated endpoint only through the same finite regulated-function, uniform-tail, window, readback, and Real-seal route.*
7. *H records componentwise* hsame *transport for F, U, W, R, E, I and for the structural rows.*
8. *C records displayed* Cont *replay for regulated-function reads, uniform-tail requests, finite window reads, regular rational readback, Real-seal requests, and integral handoff requests.*

9. *P is the* Pkg *provenance over* $\mathsf{RegulatedConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{UniformConvergence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, U, W, R, E, I, H, C, P, N$. It has no coordinate for a host function quotient, selected limiting function, measure completion, pointwise-only convergence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceTasteGate single carrier alignment`

Theorem 5819.2 (Regulated convergence NameCert obligations)**.** *For every accepted* $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier*

$$G = (F, U, W, R, E, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the regulated-function family row F, uniform-convergence tail row U, finite window row W, regular rational readback row R, terminal Real seal E, regulated-integral handoff row I, componentwise transport row H, displayed replay row C, package provenance row P, and local naming row N. Every endpoint-facing or integral-facing consumer must read these rows in the displayed route and cannot replace them by pointwise convergence, a selected host limit, quotient equality of functions, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegulatedConvergenceUp.RegulatedConvergenceCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5819.1. The only analytic rows are the regulated-function family F, the shared uniform tail ledger U, the finite window row W, the regular rational readback R, the terminal Real seal E, and the integral handoff I.

The convergence route is uniform. A consumer must read the single tail schedule U before it can inspect the windows W and the regular readback R, so pointwise-only schedules and externally selected host limits are not part of the packet.

The endpoint and integral rows are downstream. The seal E is exposed only after F, U, W, R, and the integral-facing row I reads that same route rather than a separate measure-completion or function-quotient object.

Finally, H, C, P, N transport, replay, source, and name the same finite packet. Since no excluded row occurs in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 5819.3 (Regulated convergence integral handoff)**.** *Let*

$$G = (F, U, W, R, E, I, H, C, P, N)$$

be an accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegulatedIntegral}^{\uparrow}$ *consumer read of the limiting regulated function factors through the finite route*

$$\mathsf{RegulatedFunction}^{\uparrow}(F),\ \mathsf{UniformConvergence}^{\uparrow}(U),\ W,\ \mathsf{RegSeqRat}^{\uparrow}(R),\ \mathsf{Real}^{\uparrow}(E),\ \mathsf{RegulatedIntegral}^{\uparrow}(I).$$

Thus the handoff to integration uses the same regulated-family rows, shared uniform-tail ledger, finite windows, regular rational readback, and Real seal as the convergence certificate. It supplies no measure completion, quotient function equality, selected limiting function, open-cover compactness, or ambient real-completeness principle.

Proof. Start with an integral-facing request. By the carrier definition, the only integral row is I, and its replay route in C passes through the same regulated-family and convergence rows used by endpoint-facing consumers.

Read the regulated source through F and the uniform tail through U. The finite windows W then expose the inspected function indices and domain reads, and the regular rational row R records the real-valued observations visible at those windows.

Only after this route is displayed can the terminal seal E be consumed and handed to I. Componentwise transport in H and replay in C preserve the same rows, while P and N record their provenance and local naming.

The carrier has no coordinate for measure completion, quotient function equality, selected limits, open-cover compactness, or ambient real completeness. Therefore every regulated-integral handoff is confined to the finite route displayed in the statement. □

Falsifiable prediction: An accepted $\mathsf{RegulatedConvergence}^{\uparrow}$ consumer that reaches a $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff without first reading $F, U, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: regulated-function, regulated-integral, uniform-convergence, regseqrat, real: none is bijective to $\mathsf{RegulatedConvergence}^{\uparrow}$ because this packet jointly binds a regulated-function family, a shared uniform tail ledger, finite windows, regular rational readback, terminal Real seal, integral handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{RegulatedConvergence}^{\uparrow}$ packet is generated from a regulated-function family row, UniformConvergence tail row, finite window row, RegSeqRat readback, terminal Real seal, RegulatedIntegral handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5819.1, Definition 5819.2, and Definition 5819.3 seed the finite regulated-convergence route over regulated functions, uniform convergence, regular rational readback, Real seals, and regulated-integral consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegulatedConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, quotient function equality, selected host limits, pointwise-only convergence, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family visibility, uniform-tail exactness, finite window readback, Real-seal nonescape, regulated-integral handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedConvergenceUp.

Chapter 5820

A Concrete Naming Certificate for $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet records the finite located nested-interval route by which displayed interval windows become a regular rational Cauchy name and then a terminal real seal. It refines the nested dyadic Cauchy route of chapter 1811, reads endpoint and radius rows through chapter 291, schedules finite observations through chapter 126, hands the scheduled rows to chapter 301, and reaches only the seal boundary of chapter 38. The chapter isolates the located interval-to-name handoff; it does not claim a host intersection point, quotient stream equality, selected limit, countable choice, or ambient completeness.

Definition 5820.1 (Located nested-interval limit carrier)**.** *A* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite nested located-interval window. It records the inspected interval chain, endpoint rows, located containment witnesses, and the finite separation reads available at each displayed stage.*
2. *D is the dyadic shrink ledger. It records the dyadic endpoint comparisons, width bounds, and tolerance requests read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row. It names exactly the finite endpoint, midpoint, and shrink-ledger observations that may be queried before regular rational readback.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the scheduled located-interval observations and the dyadic shrink ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for interval windows, shrink ledgers, stream schedules, regular rational handoffs, real seals, replay rows, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located nested-interval request through I, D, S, R, E and then through the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$*,* $\mathsf{NestedDyadicCauchy}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedNestedIntervalLimitUp.LocatedNestedIntervalLimitTasteGate single carrier alignment`

Theorem 5820.2 (Located nested-interval limit NameCert obligations)**.** *For every accepted* LocatedNestedIntervalL *carrier*

$$L = (I, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the finite located interval window, dyadic shrink ledger, StreamName schedule, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *located nesting exposure, requiring every interval read to be one of the displayed rows of I together with its displayed containment and located separation witnesses;*
3. *dyadic shrink exactness, requiring every tolerance request to be discharged by a visible width or endpoint comparison stored in D and read through* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
4. *schedule compatibility, requiring S to expose only the finite interval, endpoint, midpoint, and ledger observations used by D and R;*
5. *RegSeqRat handoff exactness, requiring R to be formed only after the interval window, shrink ledger, and schedule rows have been projected;*
6. *Real-seal nonescape, requiring E to be consumed only after I, D, S, R and the structural rows H, C, P, N have been displayed.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5820.1. The projection exposes one finite located nested-interval window, one dyadic shrink ledger, one StreamName schedule, one RegSeqRat handoff, one Real seal, and the transport, replay, provenance, and naming rows.

Located nesting exposure is forced by I. A consumer may inspect only interval rows already present in the finite window, and each such row carries its displayed containment and located separation witnesses.

Dyadic shrink exactness is forced by D. Each tolerance request is answered by a displayed dyadic width or endpoint comparison, so no host intersection or selected limit is needed to choose a tail.

The schedule and handoff rows are downstream. The row S records the finite observations that may be opened, and R repacks exactly those observations under the shrink ledger as a regular rational Cauchy surface.

Finally E, H, C, P, N close the route. The Real seal is reachable only after the regular handoff has been displayed, while the structural rows only transport, replay, source, and name the same finite data. Therefore no completed intersection, quotient stream equality, host real equality, countable choice, ambient completeness, Lean verification, or bridge-checking coordinate is present. □

Theorem 5820.3 (Located nested-interval RegSeqRat handoff)**.** *Let* $L = (I, D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer read exported by* L *factors through the displayed route*

$$I \longmapsto D \longmapsto S \longmapsto R \longmapsto H, C, P, N.$$

The route reads a finite located nested-interval window, a dyadic shrink ledger, and the scheduled endpoint observations before it exports the regular rational Cauchy row. The terminal $\mathsf{Real}^{\uparrow}$ *seal* E *is available only after this handoff has been displayed.*

Proof. Start with the displayed located interval window I. Its rows give the only endpoint, midpoint, containment, and located separation data available to a regular rational consumer.

Use the dyadic shrink ledger D to answer the displayed tolerance request. The chosen comparison is part of the same finite packet and is read through $\mathsf{DyadicRatCore}^{\uparrow}$, so there is no external modulus, choice-selected tail, or host interval intersection.

The schedule row S then fixes which interval observations are opened. The handoff row R is obtained by reading those scheduled observations under the shrink discipline and repacking them as the $\mathsf{RegSeqRat}^{\uparrow}$ surface.

The rows H, C, P, N transport, replay, source, and name that handoff, and the seal row E remains downstream of it. Since the carrier has no quotient-stream, host-real, selected-limit, choice, or completeness coordinate, every regular rational read factors through the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5820.1, Definition 5820.2, and Definition 5820.3 seed the finite located nested-interval route from displayed interval windows and dyadic shrink ledgers to a RegSeqRat handoff and terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedNestedIntervalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed interval intersections, quotient stream equality, host real equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located nesting exposure, dyadic shrink exactness, schedule compatibility, RegSeqRat handoff exactness, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedNestedIntervalLimitUp.

Constructive story: A $\mathsf{LocatedNestedIntervalLimit}^{\uparrow}$ packet is generated from a finite nested located-interval window, a dyadic shrink ledger, a StreamName schedule, a RegSeqRat handoff row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5821

A Concrete Naming Certificate for $\mathsf{CauchyTailRequest}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchyTailRequest}^{\uparrow}$ packet isolates the finite directed-tail request that a Cauchy or completion consumer must expose before it can read a regular rational name or a terminal real-name seal. It sits beside the finite window carrier of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the finite filter surface of chapter 5817. The packet names the request row consumed by those sibling routes; it is not a quotient filter, a selected limit, a countable-choice schedule, host stream equality, host real equality, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5821.1 (Cauchy tail request carrier)**.** *A* $\mathsf{CauchyTailRequest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, Q, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *row recording the finite source windows inspected by the request.*
2. Q *is the directed tail request row, naming the finite later-window obligations and their refinement order.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the directed tail request.*
4. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and tolerances.*
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reachable only after* W, Q, D, R *have been exposed.*
6. H *records componentwise* hsame *transport for the window, request, tolerance, readback, seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay of the same finite directed-tail route.*
8. P *records* Pkg *provenance over* $\mathsf{CauchyTailRequest}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteCauchyFilter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, Q, D, R, E, H, C, P, N$*. It has no coordinate for quotient filters, selected limits, countable choice, host equality of streams, host real equality, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*.*

Lean4 (definition) `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestUp`

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestTasteGate single carrier alignment`

Theorem 5821.2 (Cauchy tail request NameCert obligations)**.** *For the carrier of Definition 5821.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$ *surface consists exactly of carrier admission, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure, directed-tail visibility,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance preservation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every* $\mathsf{FiniteCauchyFilter}^{\uparrow}$ *consumer that reads a directed-tail row receives it through this displayed request surface.*

Lean4 ✓ `BEDC.Derived.CauchyTailRequestUp.CauchyTailRequestCarrier namecert obligations`

Proof. Project the accepted packet $T = (W, Q, D, R, E, H, C, P, N)$. The rows W, Q, D expose the finite windows, directed tail request, and dyadic tolerance ledger before any regular-rational or real-facing read is admitted.

The readback row R is downstream of those displayed rows. It therefore records only the $\mathsf{RegSeqRat}^{\uparrow}$ observation generated from the named windows and tolerance ledger, not an external sequence or choice schedule.

The seal row E is terminal for the request route. Transport through H and replay through C preserve exactly the displayed route, while P and N record provenance and local naming.

Finally compare the route with Definition 5817.2. The finite filter packet consumes its directed-tail row as a request coordinate, and the rows above are precisely the admitted request surface available to that consumer. Since no remaining coordinate exists, quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, and bridge checking are outside the chapter claim. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyTailRequest}^{\uparrow}$ packet is generated from finite StreamName windows, a directed tail request row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, seed)

Scope closed: Definition 5821.1 and Definition 5821.2 specify the finite directed-tail request carrier and its filter-facing route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyTailRequest}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected limits, countable choice, host equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for tail visibility, tolerance preservation, readback, seal routing, transport, replay, provenance, and local naming.

Chapter 5822

A Concrete Naming Certificate for $\mathsf{QuotientLeakageMarkov}^{\uparrow}$

The $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet realizes the quotient leakage transition matrix of Definition 9462.1376 as finite NameCert data. It consumes the DNA window-six boundary-tile source of subsection 9462.28.21, the codon-window carrier of chapter 2234, the boundary-tile packet of chapter 3393, the six-boundary coverage route of chapter 3470, and the translation-boundary packet of chapter 5818. The packet names a finite six-state Markov ledger only; it is not a biological stochastic model, empirical evolutionary claim, host quotient-codon semantics, or unrestricted state-space enumeration.

Definition 5822.1 (QuotientLeakageMarkov carrier)**.** *A* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, T, O, E, W, H, C, P, N)$$

with the following displayed rows:

1. *S is the six-state orbit row listing* AGR*,* Anchor_6*,* $\mathrm{StopStart}_6$*,* UUR*,* CUR*, and* CUY*.*
2. *T is the transient transition-matrix row of Definition 9462.1376, with entries read as exact rational finite counts.*
3. *O is the outside-probability row recording* $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ *for the six displayed states.*
4. *E is the expected-exit ledger row recording the rational exit values of Definition 9462.1380.*
5. *W is the finite window-six source row tying the orbit states to the codon-window and boundary-tile siblings.*
6. *H records componentwise* hsame *transport for the state, transition, outside-probability, exit-ledger, source, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite Markov read through* S, T, O, E, W, H, P, N*.*
8. *P is the* Pkg *provenance over* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$*,* $\mathsf{CodonSixBitWindow}^{\uparrow}$*,* $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$*,* $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$*,* $\mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$*, the source vision of subsection 9462.28.21,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, O, E, W, H, C, P, N$*. It has no coordinate for biological transition realization, hidden codon quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process.*

Theorem 5822.2 (QuotientLeakageMarkov NameCert obligations)**.** *For the carrier of Definition 5822.1, the local* $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$ *surface consists exactly of six-state exposure, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, window-six sibling routing, componentwise transport, displayed replay, provenance, and local naming. Every accepted read preserves the outside probabilities displayed in Definition 9462.1378.*

Proof. Project the accepted carrier $Q = (S, T, O, E, W, H, C, P, N)$. The state row S fixes the six orbit states, and the transition row T is the only source of one-bit flip probabilities.

The outside-probability row O is read from the outside column of T and matches the finite rational list in Definition 9462.1378. Thus an accepted packet cannot alter leakage rates without changing a displayed row.

The expected-exit row E is downstream of the same finite matrix and records the rational ledger of Definition 9462.1380. The source row W ties these finite state rows back to the codon-window, boundary-tile, coverage, and translation-boundary siblings.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since no other coordinate is present, the NameCert surface cannot export biological mechanism, hidden quotient semantics, empirical fitness, unbounded simulation, or a completed stochastic process. □

Theorem 5822.3 (QuotientLeakageMarkov sibling dependency route)**.** *Every accepted* $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ *consumer first reads the finite codon-window source, boundary-tile classifier, six-boundary coverage route, and translation-boundary packet before it consumes the transition row* T*, outside row* O*, or expected-exit row* E*. The dependency route is*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{DnaWindowSixBoundaryTile}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{DNAWindowSixTranslationBoundary}^{\uparrow}$$

Proof. Use the provenance row P and source row W of Definition 5822.1. These rows name the finite codon-window and boundary-tile sources before the Markov matrix is admitted.

The transition row T then reads the six-state quotient matrix of Definition 9462.1376. The outside-probability row O and expected-exit row E are downstream ledger rows, so they cannot be consumed before the sibling source route has been displayed.

Transport and replay by H and C preserve this order, while N names it locally. Hence the consumer receives a concrete dependency route rather than a detached numerical table. □

Falsifiable prediction: Every accepted $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ read preserves the displayed outside probabilities $1/3, 1/3, 1/3, 1/6, 1/2, 2/3$ for AGR, Anchor_6, StopStart_6, UUR, CUR, and CUY.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-window-six-translation-boundary: $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries six orbit states, a rational transition matrix, outside probabilities, expected-exit ledgers, window-source routing, transport, replay, provenance, and local naming.

The Lean follow-up must provide explicit $\mathsf{Nontrivial}\ \mathsf{QuotientLeakageMarkov}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{QuotientLeakageM}$ TasteGate instances preserving $S, T, O, E, W, H, C, P, N$ as distinct displayed rows.

Closure status for $\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}$**.**

Constructive story: A $\mathsf{QuotientLeakageMarkov}^{\uparrow}$ packet is generated from six orbit states, a rational transition matrix, outside-probability rows, expected-exit ledgers, a finite window-six source row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{QuotientLeakageMarkov}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5822.1, Definition 5822.2, and Definition 5822.3 specify the finite quotient-leakage Markov carrier and sibling dependency route.

Formal verification: FormalStatus(NameCert$_{\text{QuotientLeakageMarkov}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological transition realization, empirical evolutionary mechanism, hidden codon quotient semantics, unbounded simulation, completed stochastic processes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-matrix exactness, outside-probability preservation, expected-exit ledger exposure, sibling routing, transport, replay, provenance, and local naming for BEDC.Derived.QuotientLeakage

Chapter 5823

A Concrete Naming Certificate for $\mathsf{CauchyCriterionUniform}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet records the uniform Cauchy criterion route used when a Real and Completion consumer asks for one displayed threshold discipline across a finite family of stream windows. It sits beside the finite stream-window interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Cauchy hub of chapter 3413. The packet names a uniform criterion surface; it is not a quotient Cauchy class, selected limit, host equality theorem, countable-choice schedule, trichotomy principle, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5823.1 (Cauchy criterion uniform carrier)**.** *A* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *S records the* $\mathsf{StreamName}^{\uparrow}$ *window family inspected by the uniform criterion request.*
2. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the dyadic radii and comparison budgets shared by the family.*
3. *R records the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from the displayed windows and dyadic tolerances.*
4. *E records the terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, available only after S, D, R, M have been exposed.*
5. *M records the uniform modulus row, naming the common threshold discipline for all displayed windows in the family.*
6. *H records componentwise* hsame *transport for the stream windows, dyadic tolerances, regular readback, Real seal, uniform modulus, replay route, provenance, and local name.*
7. *C records displayed* Cont *replay of the same finite route.*
8. *P records* Pkg *provenance over* $\mathsf{CauchyCriterionUniform}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N records the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, D, R, E, M, H, C, P, N$. It has no coordinate for a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCriterionUniformUp.CauchyCriterionUniformTasteGate single carrier alignment`

Theorem 5823.2 (Cauchy criterion uniform NameCert obligations)**.** *For the carrier of Definition 5823.1, the local* $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the finite packet* $U = (S, D, R, E, M, H, C, P, N)$*;*
2. *uniform window exposure through the* $\mathsf{StreamName}^{\uparrow}$ *row* S *and the common modulus row* M*;*
3. *dyadic tolerance exactness through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D*;*
4. *regular readback through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* E*;*
5. *ledger non-escape through* H, C, P, N*.*

Proof. Project the accepted packet $U = (S, D, R, E, M, H, C, P, N)$. Carrier admission is exactly the admission of this finite tuple, so a consumer can inspect only the displayed coordinates.

The uniform window obligation reads S together with the common threshold discipline M. The dyadic tolerance obligation reads D, and the regular readback obligation reads R before any terminal seal row E is available.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no remaining coordinate, the ledger cannot export a quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient completeness claim. □

Theorem 5823.3 (Cauchy criterion uniform Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{CauchyCriterionUniform}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, R, M \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer must read stream windows, dyadic tolerances, regular readback, and the uniform modulus before the terminal real seal is exposed. The route exports no quotient Cauchy class, selected limit, host equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 5823.1. The only terminal real-facing coordinate is E, and the definition places it downstream of the finite stream-window, dyadic-tolerance, regular-readback, and uniform-modulus rows.

Apply Definition 5823.2. The NameCert surface admits the real seal only as part of the same transport, continuation, provenance, and naming route.

Therefore any Real-facing consumer receives exactly the finite packet route displayed above. No quotient Cauchy class, selected limit, host equality, countable-choice schedule, trichotomy row, or ambient-completeness coordinate supplies another path to the seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyCriterionUniform}^{\uparrow}$ packet is generated from finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal row, a uniform modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, seed)

Scope closed: Definition 5823.1, Definition 5823.2, and Definition 5823.3 specify the finite uniform Cauchy criterion carrier and its Real-facing seal route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyCriterionUniform}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient Cauchy classes, selected limits, host equality, countable choice, trichotomy, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean carrier encoding and checked theorems for uniform window exposure, dyadic tolerance exactness, regular readback, Real seal routing, transport, replay, provenance, and local naming.

Chapter 5824

A Concrete Naming Certificate for CauchyProductCompletionProjection$^{\uparrow}$

Following Bishop and Bridges on constructive products of Cauchy completions and Kelley on product uniformities, the CauchyProductCompletionProjection$^{\uparrow}$ packet records the coordinate projection surface for an accepted product-completion read. It sits downstream of the product packet of chapter 3797 and the completion-product route of chapter 1540, while every coordinate read still passes through the StreamName$^{\uparrow}$ windows of chapter 126, the RegSeqRat$^{\uparrow}$ readbacks of chapter 301, the DyadicRatCore$^{\uparrow}$ budgets of chapter 291, and the terminal Real$^{\uparrow}$ seal of chapter 38.

Definition 5824.1 (CauchyProductCompletionProjection carrier)**.** *A* CauchyProductCompletionProjection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N)$$

with the following displayed rows:

1. *Q is the accepted* CauchyProductCompletion$^{\uparrow}$ *product-completion packet whose coordinate route is being inspected.*
2. *π_0, π_1 are the left and right projection selector rows, each naming which displayed coordinate window is exposed.*
3. *S_0, S_1 are the coordinate* StreamName$^{\uparrow}$ *windows opened by the two projection selectors.*
4. *D_0, D_1 are the* DyadicRatCore$^{\uparrow}$ *budget rows attached to the finite coordinate observations read from those windows.*
5. *R_0, R_1 are the* RegSeqRat$^{\uparrow}$ *readback rows obtained from the same coordinate windows and dyadic budgets.*
6. *L_0, L_1 are the coordinate completion-seal rows witnessing that each projected read remains inside the product-completion packet.*
7. *H records componentwise* hsame *transport for the product packet, selectors, stream windows, dyadic budgets, regular readbacks, and coordinate completion seals.*
8. *C records the* Cont *replay from the product-completion packet through the selected coordinate window, dyadic budget, regular readback, and completion seal.*
9. *P is the* Pkg *provenance over* CauchyProductCompletionProjection$^{\uparrow}$, CauchyProductCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *row.*
The carrier compares accepted projections only through $Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, host product equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductCompletionProjectionUp.CauchyProductCompletionProjectionTasteGate`
`single carrier alignment`

Theorem 5824.2 (CauchyProductCompletionProjection NameCert obligations)**.** *For an accepted packet*

$$J = (Q, \pi_0, \pi_1, S_0, S_1, D_0, D_1, R_0, R_1, L_0, L_1, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$ *surface consists exactly of product-packet admission, selector exposure, coordinate stream-window exposure, dyadic-budget accounting, regular readback, coordinate completion-seal nonescape, componentwise transport, replay, provenance, and local naming for the displayed rows.*

Proof. Project the carrier of Definition 5824.1. The row Q supplies the already displayed product-completion packet, while π_0, π_1 determine which coordinate route is being opened.

For each selector, the corresponding stream window is read before any dyadic budget or regular readback is available. Thus S_0, S_1 precede D_0, D_1, and the regular rows R_0, R_1 are exposed only through those windows and budgets.

The completion-seal rows L_0, L_1 bind the coordinate read back to Q. The transport table H and replay table C preserve the same product packet, selector, window, budget, readback, and seal rows, while P and N record provenance and local naming.

Since these are the only carrier coordinates, the NameCert surface has no route to quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5824.3 (CauchyProductCompletionProjection route)**.** *Every coordinate read exported by an accepted* $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ *carrier factors through*

$\mathsf{CauchyProductCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameC

The left and right selectors expose only the displayed coordinate windows and their completion seals; they do not turn the product-completion packet into a host tuple, a quotient-completion equality, or a chosen representative.

Proof. Start with an accepted carrier J. The selector rows π_0, π_1 are not host projections; they are displayed BEDC rows attached to the product-completion packet Q.

Reading either selector first opens the matching $\mathsf{StreamName}^{\uparrow}$ window. The dyadic budget row is then inspected, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback is obtained from the same finite coordinate window.

The coordinate completion seal is the last coordinate-facing row before export. It keeps the read inside the product-completion packet named by Q, and the structural rows H, C, P, N attach transport, replay, provenance, and local naming to the same route.

Thus every projection read remains a finite NameCert route through the displayed rows. No coordinate of J supplies quotient completion equality, selected representatives, countable choice, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductCompletionProjection}^{\uparrow}$ packet is generated from an accepted CauchyProductCompletion packet, left and right projection selector rows, coordinate StreamName windows, DyadicRatCore budgets, RegSeqRat readbacks, coordinate completion seals, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5824.1, Definition 5824.2, and Definition 5824.3 seed the finite coordinate-projection route for product completions.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyProductCompletionProjection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host product equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, selector exposure, coordinate window accounting, regular readback, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyProductCompletionPro

Chapter 5825

A Concrete Naming Certificate for $\mathsf{CauchyModulusSubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet records the finite route by which a bounded regular Cauchy source exposes a modulus-indexed subsequence schedule without exporting an infinite selector. It sits between the bounded subsequence envelope of chapter 1698, the regular subsequence route of chapter 381, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet is a constructive completion-facing certificate: it supplies a monotone cofinal schedule, finite windows, tolerance accounting, and a Real seal, but not countable choice, quotient stream equality, a selected infinite subsequence, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5825.1 (Cauchy modulus subsequence carrier)**.** *A* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, T, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the accepted* $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$ *source row whose bounded regular Cauchy data are available before any subsequence-modulus request is read.*
2. *M is the modulus row. It records the requested Cauchy tolerance discipline and the finite comparison budget used by the subsequence schedule.*
3. *T is the monotone cofinal schedule row. It records the finite evidence that requested indices are sent to later source windows and that the schedule is cofinal in the displayed source.*
4. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows selected by the schedule row T at the requested modulus levels.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed windows and dyadic tolerances.*
7. *R is the* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff row reached from B, M, T, W, D, Q.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the bounded source, modulus, schedule, windows, tolerance ledger, regular readback, and regular subsequence handoff have been displayed.*

9. *H records componentwise* hsame *transport for the bounded source, modulus, cofinal schedule, windows, dyadic ledger, regular readback, regular subsequence handoff, terminal seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a subsequence-modulus request through $B, M, T, W, D, Q, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$, $\mathsf{BoundedRegularCauchySubsequence}^{\uparrow}$, $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, M, T, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a selected infinite subsequence, quotient stream equality, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSubsequenceUp.CauchyModulusSubsequenceTasteGate single carrier alignment`

Theorem 5825.2 (Cauchy modulus subsequence NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier $S = (B, M, T, W, D, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequ}}$ *surface consists exactly of carrier admission, bounded-source admission through B, modulus monotonicity through M, monotone cofinal schedule admission through T, finite-window determinacy through W, dyadic tolerance accounting through D, regular rational readback through Q,* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *handoff through R, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5825.1. Its analytic coordinates are exactly the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular readback Q, regular subsequence handoff R, and terminal Real seal E.

The modulus and schedule obligations are displayed before any support row is inspected. The row M supplies the tolerance discipline, while T records monotonicity and cofinality for the finite requested indices. The window row W and tolerance row D then determine which $\mathsf{StreamName}^{\uparrow}$ windows and dyadic bounds are visible at those requests.

The readback and seal obligations remain downstream. The row Q reads the visible windows as $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the $\mathsf{RegularCauchySubsequence}^{\uparrow}$ handoff, and E is reachable only after the preceding rows have been displayed. Rows H, C, P, N transport, replay, source, and name that same finite route, so no further NameCert obligation remains outside the listed coordinates. □

Theorem 5825.3 (Cauchy modulus subsequence window determinacy)**.** *Every consumer admitted by an accepted* $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ *carrier must read the bounded source B, modulus row M, monotone cofinal schedule T, finite windows W, dyadic tolerance ledger D, regular rational readback Q, regular subsequence handoff R, and terminal Real seal E before reading the transport, replay, provenance, or local naming rows. The packet exports no selected infinite subsequence, quotient stream equality, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer read. By Definition 5825.1, the only consumer-facing route before the support rows is the ordered list B, M, T, W, D, Q, R, E.

The rows B, M, T fix the source, tolerance request, and monotone cofinal schedule. The row W then supplies only the finite windows named by that schedule, and D supplies the dyadic tolerance ledger attached to those windows. Thus the visible subsequence data are determined by displayed finite requests, not by an externally selected infinite object.

The remaining analytic rows are also fixed by the displayed windows. The readback row Q converts those windows into $\mathsf{RegSeqRat}^{\uparrow}$ data, R records the regular subsequence handoff, and E is the terminal Real seal for that route. Since the carrier contains no coordinate for quotient stream equality, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the support rows H, C, P, N can only preserve the displayed finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSubsequence}^{\uparrow}$ packet is generated from a bounded regular Cauchy source, a modulus row, a monotone cofinal schedule, finite StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a RegularCauchySubsequence handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5825.1, Definition 5825.2, and Definition 5825.3 seed the constructive subsequence-modulus route over bounded regular Cauchy data, finite windows, dyadic tolerances, regular readback, regular subsequence handoff, Real sealing, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected infinite subsequences, quotient stream equality, countable choice, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, modulus monotonicity, cofinal schedule admission, finite-window determinacy, dyadic tolerance accounting, RegSeqRat readback, RegularCauchySubsequence handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusSubsequenceUp.

$\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ reads its comparison surface through chapter 126, chapter 301, chapter 291, chapter 1820, and chapter 38 before the child NameCert body is input.

Chapter 5826

A Concrete Naming Certificate for $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet records the finite comparison surface at which two regular Cauchy presentations can be inspected before a real-name or completion-facing consumer reads them. It sits on the finite observation windows of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the non-quotient classifier boundary of chapter 1820.

Falsifiable prediction: If a purported $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ consumer compares two regular Cauchy presentations without exposing finite StreamName windows, RegSeqRat readbacks, the DyadicRatCore tolerance ledger, the CauchyEquivalenceSetoid classifier, and the terminal Real seal, then the consumer has left the certificate.

Definition 5826.1 (Regular Cauchy sequence comparison carrier)**.** *A* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. S_0 *and* S_1 *are the two* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows available to the comparison request.*
2. R_0 *and* R_1 *are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows, exposing regular rational values before any real-name seal is consumed.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the displayed precision cells and refinement rows used by the comparison.*
4. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row over* S_0, S_1, R_0, R_1, D.
5. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite windows, readbacks, tolerance ledger, and classifier row are visible.*
6. H *records componentwise* hsame *transport for the two stream rows, two readback rows, dyadic ledger, classifier, real seal, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from a comparison request through* $S_0, S_1, R_0, R_1, D, Q, E, H, P, N$.
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *row.*
The carrier compares accepted packets only through $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, choice of representatives, selected limits, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchySequenceComparisonUp.RegularCauchySequenceComparisonTasteGate`
`single carrier alignment`

Theorem 5826.2 (Regular Cauchy sequence comparison NameCert obligations)**.** *For the carrier of Definition 5826.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

finite StreamName window exposure through S_0, S_1, regular rational readback through R_0, R_1, dyadic tolerance accountability through D, Cauchy-equivalence classifier exposure through Q, Real seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.

Proof. Carrier admission is projection from Definition 5826.1. The projection exposes exactly the two StreamName windows, two RegSeqRat readbacks, DyadicRatCore tolerance ledger, CauchyEquivalenceSetoid classifier, Real seal, transport, replay, provenance, and local naming rows.

The comparison route reads those rows in order. The two finite windows are inspected before the regular rational readbacks, the readbacks feed the dyadic tolerance ledger, and the ledger feeds the classifier row Q.

The Real seal E is downstream of the displayed comparison surface. The rows H, C, P, N transport, replay, source, and name that same finite route; none of them adds quotient stream equality, selected representatives, host real equality, choice, or ambient completeness. □

Theorem 5826.3 (Regular Cauchy sequence comparison carrier transport)**.** *Let*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N)$$

and

$$K' = (S'_0, S'_1, R'_0, R'_1, D', Q', E', H', C', P', N')$$

be $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *carrier packets. If the displayed row H supplies componentwise* hsame *transport from each row of K to the corresponding row of K', then carrier admission for K transports to carrier admission for K' along exactly the rows*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The transported packet has no added coordinate for quotient stream equality, host real equality, selected representative, countable choice, selected limit, or ambient completeness.

Proof. Project the carrier tuple of Definition 5826.1. The only exposed coordinates are the two StreamName rows, two RegSeqRat readback rows, DyadicRatCore tolerance ledger, classifier, Real seal, transport, replay, provenance, and local naming rows.

Use the NameCert obligations of Definition 5826.2. They require componentwise transport to act row by row on the displayed packet rather than by replacing the packet with a host quotient or selected real representative.

The sibling routes keep each transported row inside its own finite surface: StreamName supplies finite windows, RegSeqRat supplies rational readbacks, DyadicRatCore supplies tolerance cells,

CauchyEquivalenceSetoid supplies the classifier boundary, and Real supplies only the terminal seal. Therefore the transported accepted packet is still read through the same carrier coordinates, and no excluded equality, choice, limit, or completeness datum is introduced. □

Theorem 5826.4 (Regular Cauchy sequence comparison classifier stability)**.** *For an accepted packet*

$$K = (S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N),$$

replace the finite StreamName windows, RegSeqRat readbacks, and DyadicRatCore tolerance rows by rows displayed in H as componentwise hsame*-stable. Then the CauchyEquivalenceSetoid classifier row Q remains the classifier row for the transported comparison route. In particular, classifier stability is witnessed before any Real seal is consumed and does not use quotient stream equality or host real equality.*

Proof. Start from the carrier projection in Definition 5826.1. The row Q is named there only over S_0, S_1, R_0, R_1, D, so its inputs are the finite StreamName windows, the RegSeqRat readbacks, and the DyadicRatCore tolerance ledger.

Apply the obligation surface of Definition 5826.2. It places componentwise transport in the displayed row H and keeps replay in C, so replacing the finite windows, readbacks, and tolerance rows by H-stable rows preserves the same inspected classifier boundary.

The chapter's sibling routes enforce the order of reading: finite windows first, regular rational readbacks next, dyadic tolerance accounting next, and only then the CauchyEquivalenceSetoid row. The Real seal E is downstream, so classifier stability is decided without a host quotient, host equality, selected representative, choice, selected limit, or ambient completeness row. □

Theorem 5826.5 (Regular Cauchy sequence comparison ledger exactness)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or completion-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *comparison packet factors through exactly*

$$S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N.$$

The exported ledger gives the consumer finite StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, the CauchyEquivalenceSetoid classifier, the Real seal, componentwise transport, displayed replay, provenance, and local naming, and gives no selected representative, quotient equality, host real equality, countable choice, selected limit, or ambient completeness row.

Proof. Project the accepted carrier of Definition 5826.1. Its displayed coordinates are exactly $S_0, S_1, R_0, R_1, D, Q, E, H, C, P, N$, and the definition explicitly leaves out representative selection, quotient equality, host equality, choice, selected limits, and ambient completeness.

Use Definition 5826.2 to read the ledger as a NameCert surface. Carrier admission exposes the finite windows, readbacks, tolerance ledger, classifier, Real seal, transport, replay, provenance, and local name as the complete comparison route.

The StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real sibling chapters provide only their named local rows to this route. Hence any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer receives precisely the displayed packet rows and cannot obtain any excluded external equality, selection, choice, limit, or completeness datum from the ledger. □

Theorem 5826.6 (Regular Cauchy sequence comparison transitive window)**.** *Let*

$$K_{01} = (S_0, S_1, R_0, R_1, D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$K_{12} = (S_1', S_2, R_1', R_2, D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packets. If the middle presentation rows* S_1, R_1 *and* S_1', R_1' *are linked by the displayed componentwise* hsame *transport, and the dyadic tolerance/readback rows* D_{01} *and* D_{12} *expose compatible comparison windows, then the composed comparison from the first presentation to the third is forced through the same finite route*

$$S_0, S_1, S_2, R_0, R_1, R_2, D_{01}, D_{12}, Q_{01}, Q_{12}, E_{01}, E_{12}, H_{01}, H_{12}, C_{01}, C_{12}, P_{01}, P_{12}, N_{01}, N_{12}.$$

In particular, a Real or Completion consumer of the composed read receives only finite $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert *rows, and receives no quotient stream equality, selected representative, host real equality, countable choice, selected limit, or ambient completeness datum.*

Proof. Project the two accepted carriers. The first packet exposes the finite route from S_0, R_0 to S_1, R_1 through $D_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01}$, and the second exposes the corresponding route from S_1', R_1' to S_2, R_2 through $D_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12}$.

Use the displayed componentwise hsame data on the shared middle presentation. It identifies only the carried StreamName and RegSeqRat rows needed to splice the two finite comparison windows; it does not create a quotient equality of streams or a host equality of real names.

The compatible dyadic rows concatenate the displayed tolerance/readback windows. Therefore the composed read is still inspected through the listed StreamName, RegSeqRat, DyadicRatCore, classifier, Real seal, transport, replay, provenance, and local naming rows.

Since neither carrier has a coordinate for selected representatives, host real equality, selected limits, choice, or ambient completeness, the composed comparison has no such coordinate either. The transitive window is exactly the finite route displayed above. □

Theorem 5826.7 (Regular Cauchy sequence comparison RegSeqRat handoff)**.** *Every completion-facing or Real-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before it reaches the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier or the* $\mathsf{Real}^{\uparrow}$ *seal. More precisely, every exported comparison factors through*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N$$

and no consumer receives quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness from the packet.

Proof. Project the carrier of Definition 5826.1. The only pre-classifier rows are the finite StreamName windows S_0, S_1, the RegSeqRat readbacks R_0, R_1, and the DyadicRatCore ledger D.

Apply the NameCert obligations of Definition 5826.2. They force the CauchyEquivalenceSetoid classifier row Q to be read through those finite windows, readbacks, and tolerance rows before the Real seal E is available.

Finally use the sibling routes named in the chapter introduction: StreamName supplies finite windows, RegSeqRat supplies regular rational readback, DyadicRatCore supplies tolerance cells, CauchyEquivalenceSetoid supplies the non-quotient classifier, and Real supplies the terminal seal. The carrier contains no row for any excluded equality or selection principle. □

Theorem 5826.8 (Regular Cauchy sequence comparison Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ *packet reads comparison evidence only after passing through* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary. Thus equality or separation evidence is available only along the finite route*

$$S_0, S_1 \longmapsto R_0, R_1 \longmapsto D \longmapsto Q \longmapsto E \longmapsto H, C, P, N,$$

and not through host quotient equality, a selected representative, countable choice, selected limits, or ambient completeness.

Proof. Start with the RegSeqRat handoff of Definition 5826.7. It already forces every completion-facing or Real-facing read through finite StreamName windows, RegSeqRat readbacks, and the DyadicRatCore ledger before the classifier row Q and Real seal E are available.

The classifier row Q is the Cauchy-equivalence boundary for the displayed rows S_0, S_1, R_0, R_1, D. Reading equality or separation evidence before Q would skip the boundary named by the carrier, while reading it after E without Q would replace the packet by a host quotient.

The structural rows H, C, P, N preserve and name the same route. This is therefore a substantive sibling dependency among StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real, not a host quotient or a selected representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySequenceComparison}^{\uparrow}$ packet is generated from two StreamName finite-window rows, two RegSeqRat readback rows, a DyadicRatCore tolerance ledger, a CauchyEquivalenceSetoid classifier row, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 5826.1, Definition 5826.2, Definition 5826.3, Definition 5826.4, Definition 5826.5, Definition 5826.6, and Definition 5826.7 bind the obligation surface for the finite comparison route of regular Cauchy presentations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySequenceComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected representatives, countable choice, selected limits, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed BHist, hsame, Cont, Pkg, NameCert, StreamName, RegSeqRat, DyadicRatCore, CauchyEquivalenceSetoid, and Real rows as the complete BEDC primitive scope for the same comparison packet.

Chapter 5827

A Concrete Naming Certificate for $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$

Following Mac Lane on monads and Eilenberg–Moore comparison, together with Bishop–Bridges constructive completion, the $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet records the finite BEDC surface where the L10 completion monad, its algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared as one NameCert boundary. The packet reads chapter 378, chapter 1845, chapter 1981, chapter 397, chapter 3351, chapter 126, chapter 301, chapter 291, and chapter 38. It names the comparison surface only; it does not assert quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, or full $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5827.1 (CauchyCompletionMonadicity carrier)**.** *A* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *completion-monad row, displaying the unit and bind schedule before any algebra or comparison read is consumed.*
2. *A is the* $\mathsf{CauchyCompletionAlgebra}^{\uparrow}$ *action row, recording the Eilenberg–Moore algebra action over the displayed completion route.*
3. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, exposing the unit, bind, and reassociated-bind consumer route.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, carrying the extension and uniqueness ledger used by comparison consumers.*
5. *D is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *unit route, displaying the dense source-to-completion image before the algebra or universal-property read is exported.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row used by the monad, algebra, Kleisli, universal-property, and dense-embedding rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same finite windows.*
8. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare completion-facing reads at the same finite precision.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *readback seal reached only after W, R, Q have all been displayed.*
10. *H records componentwise* hsame *transport for the monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from comparison requests through $T, A, K, U, D, W, R, Q, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CauchyCompletionAlgebr $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, A, K, U, D, W, R, Q, E, H, C, P, N$. The carrier has no coordinate for quotient completion classes, selected limit representatives, host equality of completed objects, countable-choice schedules, ambient completeness, full $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

```
Lean4 ✓  BEDC.Derived.CauchyCompletionMonadicityUp.TasteGate.CauchyCompletionMonadicityTasteGate
single carrier alignment
```

Theorem 5827.2 (CauchyCompletionMonadicity NameCert obligations)**.** *For every accepted* CauchyCompletionM *carrier*

$$M = (T, A, K, U, D, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$ *surface consists exactly of carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, StreamName–RegSeqRat–DyadicRatCore–Real readback discipline, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No local obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5827.1. The displayed tuple lists every row available to the classifier, so local comparison can inspect only the completion monad, algebra action, Kleisli route, universal-property row, dense-embedding unit route, readback rows, transport, replay, provenance, and local naming.

The monadicity-facing rows are read in order. The monad row T displays the unit and bind schedule, the algebra row A records the Eilenberg–Moore action over that schedule, and the Kleisli row K keeps unit, bind, and reassociated-bind consumers on the same finite route.

The universal and dense rows then bind the comparison surface. The row U supplies extension and uniqueness data, while D supplies the dense source-to-completion unit route used before the comparison is exported.

Finally the readback and structural rows close the NameCert surface. The rows W, R, Q, E give finite windows, regular rational readback, dyadic tolerances, and the terminal Real seal; the rows H, C, P, N transport, replay, source, and name exactly that packet. Since no excluded coordinate occurs, no additional local obligation is available. □

Theorem 5827.3 (CauchyCompletionMonadicity comparison surface)**.** *Every comparison consumer of an accepted* $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ *carrier factors through the finite route*

$$T, A, K, U, D \longmapsto W \longmapsto R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus the completion-monad route, the algebra action, the Kleisli route, the universal-property row, and the dense-embedding unit route are compared only after the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$,

$\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback surface has been displayed. The comparison exports no quotient completion equality, selected limit, host equality of completed objects, countable choice, ambient completeness, or full* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the monad and algebra rows. The row T supplies the displayed completion transformer, and A reads its algebra action only as a row over that transformer.

Next attach the Kleisli and universal-property rows. The row K exposes the unit, bind, and reassociated-bind route, while U records the extension and uniqueness ledger for completion-facing consumers.

Now read the dense-embedding unit route D. It displays the source-to-completion image before the comparison reaches any terminal seal, so the route is a visible unit comparison rather than a selected representative.

Finally consume the readback and structural rows. The finite window W, regular readback R, dyadic tolerance ledger Q, and Real seal E precede transport, replay, provenance, and naming through H, C, P, N. Since the carrier contains no coordinate for the excluded objects, the comparison surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionMonadicity}^{\uparrow}$ packet is generated from a CauchyCompletionMonad route, CauchyCompletionAlgebra action row, CauchyCompletionKleisli route, CompletionUniversalProperty row, MetricCompletionDenseEmbedding unit route, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5827.1, Definition 5827.2, and Definition 5827.3 seed the finite monadicity comparison boundary for the L10 completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionMonadicity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, host equality of completed objects, countable choice, ambient completeness, full $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, completion-monad visibility, algebra-action exposure, Kleisli-route exposure, universal-property handoff, dense-embedding unit visibility, readback discipline, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionMonadicityUp.

Chapter 5828

A Concrete Naming Certificate for $\mathsf{BorsukUlam}^{\uparrow}$

Following Borsuk and Ulam, the $\mathsf{BorsukUlam}^{\uparrow}$ packet records the antipodal obstruction theorem as finite BEDC NameCert data rather than as a host topological principle. It sits after the degree surface of chapter 3745, the continuous-map rows of chapter 130, the real-name seal of chapter 38, the compact metric source of chapter 129, and the circle boundary model of chapter 58. The packet names the displayed antipodal route consumed by obstruction and degree-facing chapters; it does not select antipodal points, import unrestricted classical degree theory, identify quotient spheres, choose representatives, or appeal to ambient completeness.

Definition 5828.1 (Borsuk-Ulam carrier)**.** *A* $\mathsf{BorsukUlam}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, A, F, D, C, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the compact metric sphere row read through the displayed* $\mathsf{CompactMetric}^{\uparrow}$ *surface and the* $\mathsf{S}^{1\uparrow}$ *boundary model.*
2. *A is the antipodal-pair ledger, exposing only named paired source rows and the finite equality-refusal data attached to them.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row naming the map whose antipodal comparison is being inspected.*
4. *D is the* $\mathsf{BrouwerDegree}^{\uparrow}$ *obstruction row, read only through the displayed degree certificate and never through unrestricted host degree theory.*
5. *C is the* $\mathsf{Real}^{\uparrow}$ *comparison-seal row for the coordinate difference or collapsed real-valued comparison consumed by the obstruction route.*
6. *R is the refusal row excluding selected antipodal points, quotient equality of sphere points, choice, unrestricted classical degree theory, and ambient completeness.*
7. *H records componentwise* hsame *transport for the compact source, antipodal ledger, continuous-map row, degree row, real comparison seal, refusal row, provenance, and local naming.*
8. *K records displayed* Cont *replay from the compact metric source through the antipodal ledger, continuous-map row, degree obstruction, real comparison seal, and refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{BorsukUlam}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{BrouwerDegree}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *row.*
The classifier compares accepted packets only through $S, A, F, D, C, R, H, K, P, N$. It has no coordinate for a chosen antipodal pair, quotient equality on the sphere, unrestricted classical degree theory, countable choice, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.BorsukUlamUp.BorsukUlamUp`
Lean4 ✓ `BEDC.Derived.BorsukUlamUp.BorsukUlamTasteGate single carrier alignment`

Theorem 5828.2 (Borsuk-Ulam NameCert obligations)**.** *For every accepted* $\mathsf{BorsukUlam}^\uparrow$ *carrier* $U = (S, A, F, D, C, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BorsukUlam}^\uparrow}$ *surface consists exactly of compact metric sphere admission, antipodal-pair ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, refusal of host and quotient shortcuts, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obstruction-facing consumer must read*

$$S, A, F \longmapsto D, C \longmapsto R, H, K, P, N$$

before treating the packet as a Borsuk-Ulam certificate.

Proof. Project the carrier of Definition 5828.1. The compact source row S, antipodal ledger A, and continuous-map row F are the only rows available before the obstruction is read.

The degree and real-comparison rows D, C are downstream of that source route. The degree row is the displayed $\mathsf{BrouwerDegree}^\uparrow$ certificate, and the real row is the displayed $\mathsf{Real}^\uparrow$ seal for the comparison consumed by the obstruction.

Finally R, H, K, P, N refuse shortcuts, transport the same rows, replay the finite route, source it, and name it locally. Since the carrier has no coordinate for selected antipodal points, quotient equality, unrestricted host degree theory, countable choice, or ambient Real completeness, no consumer can export those objects from the certificate. □

Theorem 5828.3 (Borsuk-Ulam antipodal route)**.** *Every accepted* $\mathsf{BorsukUlam}^\uparrow$ *antipodal-obstruction read factors through the displayed compact metric sphere row, the* $\mathsf{S}^{1\uparrow}$ *boundary model, the antipodal-pair ledger, the* $\mathsf{ContinuousMap}^\uparrow$ *row, the* $\mathsf{BrouwerDegree}^\uparrow$ *obstruction row, the terminal* $\mathsf{Real}^\uparrow$ *comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no selected antipodal point, unrestricted classical degree theorem, quotient equality of sphere points, choice principle, ambient completeness, Lean verification, or bridge-checked surface.*

Proof. Start with an accepted carrier $U = (S, A, F, D, C, R, H, K, P, N)$. The source read begins at S and A, so the antipodal data is visible only as a finite paired ledger over the displayed compact metric and circle rows.

Read the map through F and then the obstruction through D, C. The $\mathsf{ContinuousMap}^\uparrow$ row supplies the function-facing data, the $\mathsf{BrouwerDegree}^\uparrow$ row supplies the obstruction route, and the $\mathsf{Real}^\uparrow$ row supplies only the comparison seal reached from those displayed rows.

The refusal and structural rows close the route. The row R blocks selected points, quotient equality, unrestricted degree theory, choice, and ambient completeness, while H, K, P, N preserve the same finite certificate under transport, replay, provenance, and local naming. □

Falsifiable prediction: Every accepted $\mathsf{BorsukUlam}^\uparrow$ consumer reads compact metric sphere data, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a Real

comparison seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it treats the packet as an antipodal obstruction certificate.

Independence witness: brouwer-degree, continuousmap, real, compactmetric, s1: $\mathsf{BorsukUlam}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact sphere data, antipodal pairing, continuous-map readback, degree obstruction, real comparison sealing, refusal, transport, replay, provenance, and local naming into one obstruction-facing certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}$.

Constructive story: A $\mathsf{BorsukUlam}^{\uparrow}$ packet is generated from a compact metric sphere row, an S1 boundary model, an antipodal-pair ledger, a ContinuousMap row, a BrouwerDegree obstruction row, a RealUp comparison seal, shortcut refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5828.1, Definition 5828.2, and Definition 5828.3 seed the finite antipodal obstruction route over compact metric, circle, continuous-map, degree, real-seal, transport, replay, provenance, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BorsukUlam}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected antipodal points, unrestricted classical degree theory, quotient equality, choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, antipodal ledger exposure, continuous-map readback, Brouwer-degree obstruction routing, Real comparison sealing, transport, replay, provenance, and local naming for BEDC.Derived.BorsukUlamUp.

Chapter 5829

A Concrete Naming Certificate for $\mathsf{DenjoyYoungSaks}^{\uparrow}$

Following Saks and Bishop–Bridges constructive real analysis, the $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet records the finite boundary at which a real-valued interval map can expose upper and lower Dini-derivative rows without importing the classical almost-everywhere Denjoy–Young–Saks theorem. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation window of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet is a finite classification boundary for displayed Dini rows, oscillation windows, regular-rational readback, and terminal Real seals; it is not a measure theorem, an almost-everywhere classifier, or a host-real differentiability principle.

Definition 5829.1 (Denjoy–Young–Saks finite Dini carrier)**.** *A* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the inspected interval row, listing the finitely presented source intervals and endpoint cells on which the local boundary is read.*
2. *F is the function-observation row, recording the finite source and target windows used to compare difference quotients.*
3. *U is the upper Dini row, storing the displayed upper right, upper left, and two-sided upper derivative bounds visible at the inspected windows.*
4. *L is the lower Dini row, storing the corresponding lower right, lower left, and two-sided lower derivative bounds.*
5. *O is the oscillation-window row, recording the finite alternation and stabilization tests used to classify a displayed window as one of the Dini-boundary cases.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by I, F, O and no hidden stream coordinates.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data extracted from the finite stream windows.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger supporting the difference-quotient comparisons and oscillation bounds.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the interval, function-observation, Dini, oscillation, stream, readback, and dyadic rows have been exposed.*
10. *H records componentwise* hsame *transport for intervals, function observations, Dini rows, oscillation windows, stream schedules, regular readbacks, dyadic ledgers, Real seals, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from the inspected interval and function rows through U, L, O, S, R, D and into the Real seal E.*
12. *P is the* Pkg *provenance over* $\mathsf{DenjoyYoungSaks}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, F, U, L, O, S, R, D, E, H, C, P, N$. It has no coordinate for measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, host real order trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate single carrier alignment`

Theorem 5829.2 (Denjoy–Young–Saks finite boundary obligations)**.** *For every accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the inspected interval, function-observation, upper Dini, lower Dini, oscillation-window, stream-schedule, regular-readback, dyadic-ledger, Real-seal, transport, replay, provenance, and local naming rows;*
2. *interval and observation exactness, requiring every Dini comparison to read its source cell from I and its finite function window from F before any derivative row is accepted;*
3. *upper and lower Dini-row exactness, requiring U and L to store only the finite upper and lower derivative bounds justified by the displayed difference-quotient windows;*
4. *oscillation-window classification, requiring O to record the finite alternation, stabilization, and gap tests that separate the visible Dini-boundary cases;*
5. *stream, readback, and dyadic exactness, requiring S, R, D to expose only the finite observations, regular rational readbacks, and dyadic tolerance rows demanded by I, F, U, L, O;*
6. *Real seal nonescape, requiring E to remain downstream of I, F, U, L, O, S, R, D and never to serve as a selected derivative, a host real equality witness, or an almost-everywhere theorem;*
7. *componentwise transport, replay, provenance, and local naming, requiring H, C, P, N to preserve exactly the displayed finite route.*

No $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *obligation is stored outside these rows, and the packet makes no claim that the classical Denjoy–Young–Saks alternatives hold almost everywhere.*

Proof. Carrier admission is projection from Definition 5829.1. The accepted packet displays exactly the inspected interval row I, function-observation row F, upper and lower Dini rows U, L, oscillation row O, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Interval and observation exactness follow from the row order. A Dini comparison first reads the source cell in I and the finite function-observation window in F; only then may it inspect the upper row U, lower row L, or oscillation row O attached to that window.

The stream, regular-readback, and dyadic rows are downstream of the same finite windows. The schedule S opens only the observations demanded by I, F, O, the row R records the resulting regular rational readback, and D stores the tolerance comparisons needed for the displayed difference-quotient bounds.

Finally, the Real seal E is terminal for this finite boundary route. The transport row H, replay row C, provenance row P, and local naming row N preserve the same packet, so measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness cannot become local obligations. □

Theorem 5829.3 (Denjoy–Young–Saks Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier reads the displayed route*

$$I, F, U, L, O, S, R, D, E, H, C, P, N.$$

Consequently the consumer may inspect finite interval cells, function observations, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal Real seal, componentwise transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere assertion, a selected derivative, quotient real equality, host real trichotomy, countable choice, or ambient completeness from the packet.

Proof. Project the accepted carrier through Definition 5829.1. Its Real-facing and readback-facing rows are exactly S, R, D, E together with the preceding interval, function, Dini, and oscillation rows I, F, U, L, O and the structural rows H, C, P, N.

The route reaches E only after the finite Dini and oscillation boundary has been displayed. Thus any consumer that asks for a Real seal or regular rational readback must already have passed through the interval cells, function observations, upper and lower Dini rows, oscillation windows, stream schedule, and dyadic tolerance ledger.

Since the carrier has no row for measure, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, or ambient completeness, none of those objects can be read by a downstream consumer. □

Theorem 5829.4 (Denjoy–Young–Saks oscillation-window induction)**.** *For an accepted* $\mathsf{DenjoyYoungSaks}^{\uparrow}$ *carrier* $Y = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, induction over the finite oscillation-window row* O *preserves the separation between the upper Dini row* U *and the lower Dini row* L*. At every displayed stage the packet remains a finite boundary classifier over the rows of Definition 5829.1; the induction cannot promote the packet to an almost-everywhere measure theorem, cannot select a derivative, and cannot export a host-real differentiability principle.*

Proof. Induct over the displayed list of oscillation windows in O. The empty row has no Dini comparison to transport, so the upper and lower rows remain separated by the carrier projection.

For the step, assume the previously displayed windows preserve the separation of U and L. The next window is admitted only through the oscillation-window classification row in Definition 5829.2, so it may inspect the finite alternation, stabilization, and gap tests attached to that window but cannot rewrite the carrier outside $I, F, U, L, O, S, R, D, E, H, C, P, N$.

The Real-seal nonescape theorem Definition 5829.3 keeps every downstream read behind the same finite route. Thus adjoining the next oscillation window preserves the visible upper/lower Dini separation and leaves measure-null exceptional sets, almost-everywhere validity, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness outside the packet. □

Falsifiable prediction: Every $\mathsf{DenjoyYoungSaks}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, oscillation windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{DenjoyYoungSaks}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries upper and lower Dini rows, oscillation-window classification, regular-rational readback, dyadic tolerance data, and a terminal Real seal in one finite boundary packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}$.

Constructive story: A $\mathsf{DenjoyYoungSaks}^{\uparrow}$ packet is generated from inspected interval cells, function-observation windows, upper and lower Dini rows, oscillation-window classifications, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5829.1, Definition 5829.2, and Definition 5829.3 seed the finite Denjoy–Young–Saks boundary route over Dini rows, oscillation windows, regular-rational readback, dyadic tolerance rows, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DenjoyYoungSaks}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DenjoyYoungSaksUp.DenjoyYoungSaksTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close the classical almost-everywhere Denjoy–Young–Saks theorem, measure-null exceptional sets, selected derivatives, quotient real equality, trichotomy, countable choice, ambient completeness, or bridge checking beyond the carrier-alignment Lean target.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row exactness, oscillation-window classification, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DenjoyYoungSaksUp.

Chapter 5830

A Concrete Naming Certificate for $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet records the bounded-sequence subsequence route in the form used by constructive real analysis: finite inspection of a bounded $\mathsf{Real}^{\uparrow}$ sequence, repeated interval refinement, monotone-cofinal subsequence scheduling, regular rational readback, and only then a $\mathsf{Real}^{\uparrow}$ cluster seal. The packet is anchored by the existing $\mathsf{BolzanoWeierstrass}^{\uparrow}$ route of chapter 1254, the sequential compactness naming surface of chapter 2372, the compact metric carrier of chapter 129, finite stream windows from chapter 126, regular rational readback from chapter 301, and the real-name seal of chapter 38. It is a displayed subsequence construction for bounded real sequences, not an open-cover compactness theorem, a quotient equality of real names, an ambient completeness principle, or a choice-selected arbitrary subsequence.

Definition 5830.1 (Constructive Bolzano–Weierstrass carrier)**.** *A* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, B, I, T, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the bounded* $\mathsf{Real}^{\uparrow}$ *sequence source, read through finite* $\mathsf{StreamName}^{\uparrow}$ *windows and the bounded-source handoff of chapter 1415.*
2. *B is the* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *carrier row, exposing the interval-net and regular-subsequence route without adding a separate host compactness object.*
3. *I is the finite interval-refinement ledger, read through the closed bounded interval net and compact metric rows consumed by the sibling* $\mathsf{BolzanoWeierstrass}^{\uparrow}$ *packet.*
4. *T is the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row, recording that the same finite subsequence schedule is the sequential-compactness-facing surface of the construction.*
5. *R is the monotone-cofinal subsequence row, read as a regular Cauchy schedule before any cluster seal is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected finite windows and their dyadic tolerance comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *cluster seal, reached only after S, B, I, T, R, Q have been displayed.*

8. *H records componentwise* hsame *transport for the bounded sequence, interval ledger, sequential compactness row, subsequence schedule, regular rational readback, and real seal.*
9. *C records displayed* Cont *replay from finite bounded-sequence inspection through interval refinement, subsequence scheduling, RegSeqRat readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, I, T, R, Q, E, H, C, P, N$. It has no coordinate for open-cover compactness, quotient real equality, a selected least cluster point, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveBolzanoWeierstrassUp.ConstructiveBolzanoWeierstrassTasteGate`
`single carrier alignment`

Theorem 5830.2 (Constructive Bolzano–Weierstrass NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$ *surface consists exactly of bounded sequence admission through S, Bolzano–Weierstrass route admission through B, finite interval-refinement admission through I, sequential compactness comparison through T, monotone-cofinal subsequence scheduling through R, RegSeqRat readback through Q, RealUp cluster sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5830.1. The rows $S, B, I, T, R, Q, E, H, C, P, N$ are the only coordinates exposed to a consumer.

The bounded source is read first through S, and the sibling $\mathsf{BolzanoWeierstrass}^{\uparrow}$ row B supplies the interval-net and regular-subsequence route. The interval ledger I keeps that route finite and tied to compact metric inspection rather than open-cover compactness.

The comparison row T then records the sequential compactness view of the same subsequence schedule. Rows R and Q expose the monotone-cofinal indices and regular rational readback before the real seal E can be consumed.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no further coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5830.3 (Constructive bounded Real subsequence obligation)**.** *Every* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *consumer that reads a bounded* $\mathsf{Real}^{\uparrow}$ *sequence from an accepted carrier must traverse the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{BolzanoWeierstrass}^{\uparrow} \longmapsto \mathsf{SequentialCompact}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes finite source windows, interval-refinement rows, a monotone-cofinal subsequence schedule, regular rational readback, and a terminal Real seal; it does not export an arbitrary subsequence selector, quotient real equality, open-cover compactness, or ambient completeness.

Proof. Use Definition 5830.2. It places S and the finite stream windows before the interval-refinement ledger I and before any subsequence schedule is read.

The row B imports the existing Bolzano–Weierstrass naming route as a displayed carrier row, while T records the sequential compactness comparison over the same finite schedule. Neither row adds a host open-cover object or a choice coordinate.

Rows R and Q provide the cofinal subsequence schedule and the regular rational readback. Only after these rows are displayed may E be consumed as a $\mathsf{Real}^{\uparrow}$ cluster seal, so the consumer route is the stated finite subsequence route. □

Theorem 5830.4 (Constructive cluster seal boundary obligation)**.** *For an accepted* ConstructiveBolzanoWeierstrass *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is downstream of the bounded source row, the interval-refinement ledger, the sequential compactness comparison, the monotone-cofinal subsequence schedule, and the RegSeqRat readback. Any consumer of* E *must preserve the same* H, C, P, N *support rows and receives no additional host completeness, quotient equality, or open-cover compactness datum.*

Proof. Start from the bounded subsequence route of Definition 5830.3. It identifies the finite route through source windows, compact metric inspection, the sibling Bolzano–Weierstrass packet, sequential compactness comparison, and regular rational readback.

In the carrier definition the seal row E occurs only after S, B, I, T, R, Q. The support rows H, C, P, N can transport, replay, source, and name that route, but they cannot introduce a new endpoint selector or an ambient completeness principle.

Thus the cluster seal is a boundary object for the displayed constructive route. It is a $\mathsf{Real}^{\uparrow}$ seal over the named finite data, not a quotient-real equality witness, open-cover compactness theorem, or hidden host-level limit. □

Theorem 5830.5 (Constructive finite subsequence envelope)**.** *For an accepted* $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ *carrier* $W = (S, B, I, T, R, Q, E, H, C, P, N)$*, the monotone-cofinal subsequence row* R *is determined by the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in* S*, the interval-refinement ledger rows in* I*, the* $\mathsf{SequentialCompact}^{\uparrow}$ *comparison row* T*, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *before the* $\mathsf{Real}^{\uparrow}$ *cluster seal* E *is exposed. Consequently every endpoint consumer reads the same finite envelope and receives no arbitrary subsequence selector, countable-choice schedule, quotient equality, or ambient completeness datum.*

Proof. Use the carrier definition of Definition 5830.1. It places the bounded source windows S, the sibling Bolzano–Weierstrass route B, the interval ledger I, and the sequential compactness comparison row T before the subsequence schedule R and the readback row Q.

The bounded-real-subsequence route of Definition 5830.3 identifies this as a finite route through StreamName windows, compact metric inspection, interval refinement, sequential compactness comparison, and RegSeqRat readback. Thus R is not an independently selected subsequence coordinate; it is the schedule visible through the same finite rows.

The cluster seal boundary of Definition 5830.4 keeps E downstream of S, B, I, T, R, Q and the support rows H, C, P, N. Any endpoint consumer therefore receives the same finite envelope before the seal.

Since the carrier has no row for an arbitrary subsequence selector, countable choice, quotient equality, or ambient completeness, none of those data can be exported by the cluster read. □

Theorem 5830.6 (Constructive Bolzano–Weierstrass selector nonescape)**.** *For an accepted* ConstructiveBolzanoWe *carrier*

$$W = (S, B, I, T, R, Q, E, H, C, P, N),$$

the $\mathsf{Real}^{\uparrow}$ *cluster-seal read* E *is available only through the displayed interval-refinement ledger* I*, the finite subsequence envelope* S, B, I, T, R, Q*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the terminal Real seal row* E *together with* H, C, P, N*. No consumer may detach* R *as a hidden selector, replace the route by countable choice, or read a cluster point without the finite interval-refinement and regular-rational rows.*

Proof. Start from the bounded Real subsequence route of Definition 5830.3. It forces every accepted read through finite StreamName windows, compact metric inspection, the sibling Bolzano–Weierstrass route, sequential compactness comparison, RegSeqRat readback, and the Real seal.

Apply the cluster seal boundary theorem Definition 5830.4. The seal row E is downstream of S, B, I, T, R, Q, and the support rows H, C, P, N preserve that route without adding a selected endpoint.

Use the finite subsequence envelope of Definition 5830.5. The subsequence schedule R is determined by the visible source windows, interval ledger, sequential compactness comparison, and regular rational readback before E is exposed.

Thus the selector cannot escape the displayed envelope. A consumer that reads R or E without the interval-refinement, finite envelope, RegSeqRat, and Real-seal rows is no longer reading an accepted $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}$ packet is generated from a bounded RealUp sequence source, the sibling Bolzano–Weierstrass interval route, compact metric inspection, sequential compactness comparison, monotone-cofinal subsequence scheduling, RegSeqRat readback, a RealUp cluster seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5830.1, Definition 5830.2, Definition 5830.3, Definition 5830.4, and Definition 5830.5 seed the Bishop-style bounded Real sequence subsequence route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveBolzanoWeierstrass}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient equality of real names, selected least cluster points, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-refinement exactness, sequential compactness comparison, subsequence scheduling, RegSeqRat readback, Real cluster sealing, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveBolzanoWeierstrassUp.

Chapter 5831

A Concrete Naming Certificate for $\mathsf{UniformlyConvexSpace}^{\uparrow}$

Following Clarkson's uniformly convex Banach-space reading and Bishop–Bridges constructive normed-space practice, the $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet records a finite modulus-of-convexity surface over the normed-space carrier of chapter 1950. It consumes the $\mathsf{NormedSpace}^{\uparrow}$ row, the real-valued norm and midpoint reads through chapter 38, the metric handoff of chapter 57, and the completion-facing route of chapter 95 and chapter 3413. The packet gives Banach, Hilbert, fixed-point, and convexity consumers a named uniform-convexity certificate without importing host Banach-space equality, selected minimizers, quotient completions, or choice.

Definition 5831.1 (UniformlyConvexSpace carrier)**.** *A* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, R, A, M, \Delta, Q, H, T, P, C)$$

with the following displayed rows:

1. *N is the* $\mathsf{NormedSpace}^{\uparrow}$ *row, exposing vector operations, the norm read, and the induced metric only through the normed-space NameCert surface.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by norm values, radius comparisons, midpoint estimates, and positive modulus outputs.*
3. *A is the midpoint row, reading $(x + y)/2$ through the vector addition, scalar action, and real scalar rows already visible in N and R.*
4. *M is the metric handoff row, routing distance and unit-ball comparisons through the* $\mathsf{Met}^{\uparrow}$ *read induced by N.*
5. *Δ is the convexity-modulus row: for each displayed positive separation request it supplies a positive real margin forcing the midpoint norm below the unit bound whenever the two admitted unit vectors are separated by that request.*
6. *Q is the completion-facing row, allowing Cauchy and completion consumers to read the same norm, metric, midpoint, and modulus data without selecting a completed point.*
7. *H records componentwise* hsame *transport for N, R, A, M, Δ, Q and for the local structural rows.*
8. *T records displayed* Cont *replay from a uniform-convexity request through $N, R, A, M, \Delta, Q, H, P, C$.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformlyConvexSpace}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. C is the local $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Banach-space equality, strict-convexity as a pointwise substitute, quotient completion, selected closest points, compactness, reflexivity, or a choice principle.

Theorem 5831.2 (UniformlyConvexSpace NameCert obligations)**.** *For every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *carrier* $U = (N, R, A, M, \Delta, Q, H, T, P, C)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, normed-space visibility, real-valued norm and midpoint readback, metric separation handoff, positive modulus admission, midpoint contraction under the displayed separation request, completion-facing nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $N, R, A, M, \Delta, Q, H, T, P, C$*.*

Lean4 ✓ `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5831.1. The rows $N, R, A, M, \Delta, Q, H, T, P, C$ are the whole accepted packet.

The normed-space, Real, midpoint, and metric obligations are read first. The row N supplies the normed vector carrier, R supplies real-valued comparisons, A supplies midpoint formation, and M supplies the induced metric separation route.

The uniform-convexity content is isolated in Δ. For a displayed positive separation request, Δ supplies a positive margin and forces the midpoint norm read through A, N, R below the unit bound. Since Q, H, T, P, C only hand off, transport, replay, source, and name the same route, no host Banach-space equality, quotient completion, selected point, compactness, reflexivity, or choice coordinate can enter. □

Theorem 5831.3 (UniformlyConvexSpace modulus convexity route)**.** *Every accepted* $\mathsf{UniformlyConvexSpace}^{\uparrow}$ *consumer that asks for uniform convexity factors through*

$$\mathsf{NormedSpace}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Cauchy}^{\uparrow} \longmapsto H, T, P, C.$$

The modulus row Δ may be read only after the normed-space carrier, real norm values, midpoint operation, and metric separation row have been displayed. The route exports a finite modulus-of-convexity NameCert surface, not host strict convexity, a selected minimizer, quotient completion equality, compactness, reflexivity, or countable choice.

Proof. Begin with a uniform-convexity request. The normed-space row N supplies the only vector and norm data, while R supplies the real-valued unit-ball and positivity comparisons.

Read midpoint and separation next. The midpoint row A is formed from the displayed operations of N and R, and the metric handoff M reads separation through the induced metric of the same normed-space packet.

Only then can the modulus row Δ answer the request. Its output is tied to the displayed positive separation and midpoint norm route, and the completion-facing row Q sends Cauchy or completion consumers through the same route rather than through a selected completed point.

Finally H, T, P, C transport, replay, source, and name the packet. The excluded host strict-convexity, selected-minimizer, quotient-completion, compactness, reflexivity, and choice objects are absent from the carrier, so the finite modulus route is the whole exported surface. □

Falsifiable prediction: An accepted $\mathsf{UniformlyConvexSpace}^{\uparrow}$ read is refuted if a consumer obtains a uniform-convexity margin without first displaying the $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, midpoint, $\mathsf{Met}^{\uparrow}$, completion-facing, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: normedspace, real, metric, completion, cauchy: no listed sibling is bijective to $\mathsf{UniformlyConvexSpace}^{\uparrow}$ because the packet jointly carries normed-space data, real norm comparisons, midpoint formation, metric separation, a positive convexity modulus, completion-facing handoff, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformlyConvexSpace}^{\uparrow}$ packet is generated from a NormedSpace row, Real norm-comparison rows, a midpoint row, an induced Metric separation row, a convexity-modulus row, a Completion and Cauchy handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5831.1, Definition 5831.2, and Definition 5831.3 seed the finite modulus-of-convexity route over normed-space, real, metric, completion, and Cauchy rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformlyConvexSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformlyConvexSpaceUp.UniformlyConvexSpaceTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Banach-space equality, strict-convexity substitution, selected minimizers, quotient completions, compactness, reflexivity, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, midpoint formation, metric separation, positive modulus exactness, completion-facing nonescape, and local naming for BEDC.Derived.UniformlyConvexSpaceUp.

Chapter 5832

A Concrete Naming Certificate for $\mathsf{HyperbolicTranslationLength}^{\uparrow}$

Following the classical translation length of a hyperbolic isometry, the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet records the finite invariant-length route visible to the hyperbolic phase geometry of chapter 9463. It consumes the metric-read surface of chapter 3474, the routed geodesic-flow surface of chapter 3856, the additive Busemann cocycle route of chapter 4134, the four-boundary-point comparison of chapter 3948, and the terminal real seal of chapter 38. The packet names stable displacement growth as finite BEDC data; it does not import a host hyperbolic manifold, quotient ideal boundary, selected axis, spectral radius, or global curvature theorem.

Definition 5832.1 (HyperbolicTranslationLength carrier)**.** *A* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, F, B, X, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, exposing displacement reads for the source and its iterated image.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite routed-flow data used to compare iterates along the same visible geodesic route.*
3. *B is the* $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$ *row, supplying additive boundary-cocycle comparisons for the same displayed endpoints.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *row, recording the four-boundary-point invariant used to stabilize the length read.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by displacement values, additive cocycle values, cross-ratio reads, and the final nonnegative length seal.*
6. *D is the displacement-growth row, comparing the displayed iterate window through M, F, B, X, R without selecting an infinite orbit or an axis.*
7. *E is the endpoint-stability row, stating that the same boundary data are replayed by the metric, flow, Busemann, and cross-ratio rows.*
8. *H records componentwise* hsame *transport for M, F, B, X, R, D, E and for the local structural rows.*

9. *C records displayed* Cont *replay from a translation-length request through $M, F, B, X, R, D, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed hyperbolic manifold, or physical expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicTranslationLengthUp.HyperbolicTranslationLengthTasteGate single carrier alignment`

Theorem 5832.2 (HyperbolicTranslationLength NameCert obligations)**.** *For every accepted* HyperbolicTranslation *carrier $L = (M, F, B, X, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$ *surface consists exactly of metric displacement visibility, geodesic-flow routing, Busemann additivity, cross-ratio stabilization, real-valued length sealing, displacement-growth comparison, endpoint stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $M, F, B, X, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 5832.1. The rows $M, F, B, X, R, D, E, H, C, P, N$ are the whole visible packet.

The geometric route is read first. Metric displacement comes from M, routed iteration from F, additive boundary comparison from B, and four-boundary-point stabilization from X.

The Real and displacement rows then seal the length read. The row R supplies the real-valued comparison surface, while D and E ensure that the same endpoint data are used by the metric, flow, Busemann, and cross-ratio reads.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate names a host isometry group, selected invariant axis, quotient boundary equality, spectral radius, completed manifold, or physical expansion law, the listed obligations exhaust the local surface. □

Theorem 5832.3 (HyperbolicTranslationLength invariant route)**.** *Every accepted* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *read factors through*

$$\mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicBusemannCocycle}^{\uparrow}, \mathsf{HyperbolicCrossRatio}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto H, C, P$$

The finite route records displacement growth and boundary stabilization as a real-sealed invariant length, and it exports no host hyperbolic isometry, selected axis, quotient ideal-boundary equality, spectral radius, completed hyperbolic manifold, or cosmological expansion claim.

Proof. Begin with the displacement request. The metric row M supplies finite distance reads, and the geodesic-flow row F keeps the iterated comparison on the same displayed route.

Read boundary stabilization next. The Busemann cocycle row B supplies additive endpoint comparison, while the cross-ratio row X records the four-boundary-point invariant. The endpoint-stability row E requires these boundary reads to describe the same displayed data.

The Real row R and displacement-growth row D seal the invariant length as a finite real-valued read. The structural rows H, C, P, N preserve, replay, source, and name that route, while the excluded host and quotient objects remain absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ consumer is refuted if it obtains an invariant length without displaying the $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicBusemannCocycle}$ $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, displacement-growth, endpoint-stability, transport, replay, provenance, and local $\mathsf{NameCert}$ rows.

Independence witness: hyperbolicmetric, hyperbolic-geodesic-flow, hyperbolic-busemann-cocycle, hyperbolic-cross-ratio, real: no listed sibling is bijective to $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ because the packet jointly carries metric displacement, routed flow, additive boundary cocycle, cross-ratio stabilization, Real sealing, displacement-growth comparison, endpoint stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ packet is generated from HyperbolicMetric displacement rows, HyperbolicGeodesicFlow routing rows, HyperbolicBusemannCocycle additive boundary rows, HyperbolicCrossRatio stabilization rows, Real sealing rows, displacement-growth and endpoint-stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5832.1, Definition 5832.2, and Definition 5832.3 seed the finite invariant-length packet over metric, flow, Busemann, cross-ratio, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicTranslationLength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic isometries, selected axes, quotient ideal-boundary equality, spectral radii, completed hyperbolic manifolds, physical expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement visibility, flow routing, Busemann additivity, cross-ratio stabilization, Real length sealing, endpoint stability, and local naming for BEDC.Derived.HyperbolicTranslationLengthUp.

Chapter 5833

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet records the finite envelope that carries a regular Cauchy rational readback into the terminal Real naming seal. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational sequence interface of chapter 301, the terminal real-name packet of chapter 38, and the completion route of chapter 1506. The packet is a finite NameCert envelope for the handoff itself; it does not assert that the host real line is complete.

Definition 5833.1 (Regular Cauchy completion envelope carrier)**.** *A* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier is a finite packet*

$$E = (W, D, R, E_R, M, T, C, P, N)$$

where W *is the displayed* $\mathsf{StreamName}^{\uparrow}$ *window row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row,* E_R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* M *is the* $\mathsf{MetricCompletion}^{\uparrow}$ *route row,* T *records componentwise* $\mathsf{BHist}/\mathsf{hsame}$ *transport,* C *records displayed* Cont *replay,* P *is* Pkg *provenance over the displayed rows, and* N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$ *row. A consumer of the carrier receives the completion handoff only through these nine coordinates.*

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate`
`single carrier alignment`

Theorem 5833.2 (Regular Cauchy completion envelope obligations)**.** *For every accepted* $\mathsf{RegularCauchyCompletion}$ *carrier*

$$E = (W, D, R, E_R, M, T, C, P, N),$$

the local NameCert surface consists exactly of window admission, tolerance admission, regular readback admission, terminal Real-seal admission, metric-completion route admission, transport stability, continuation replay, provenance preservation, and local naming preservation for the displayed packet. No quotient stream equality, selected representative, countable choice schedule, host completeness principle, or $\mathsf{Real}^{\uparrow}$ *completeness theorem is a coordinate of this surface.*

Proof. Project the carrier of Definition 5833.1. Its only data rows are $W, D, R, E_R, M, T, C, P, N$, so the local NameCert can inspect exactly those displayed coordinates.

The first four rows bind the finite observation, dyadic tolerance, regular rational readback, and terminal Real seal. The metric-completion row M is therefore read as a route row over those data, not as an ambient completeness object.

The structural rows T, C, P, N preserve the same packet under componentwise transport, continuation replay, provenance, and local naming. Since the carrier has no slot for quotient stream equality, selected representatives, countable choice, host completeness, or Real completeness, none of those objects can be exported by the obligation surface. □

Theorem 5833.3 (Regular Cauchy completion envelope Real handoff)**.** *Every Real-facing completion consumer of an accepted* $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ *carrier reads the terminal handoff through the ordered finite route*

$$W, D, R, E_R, M, T, C, P, N.$$

In particular, the consumer reaches the terminal $\mathsf{Real}^{\uparrow}$ *seal only after the* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback have been displayed inside the same envelope.*

Proof. Start from Definition 5833.2. It makes $W, D, R, E_R, M, T, C, P, N$ the entire accepted surface of the envelope.

Read the Real-facing route through W, D, R, E_R. The StreamName windows expose the finite observations, the dyadic ledger records the tolerance budget, the regular-rational row gives the readback, and the terminal row gives the Real seal inside the same packet.

The completion route M and the structural rows T, C, P, N only replay and name that displayed handoff. Thus the consumer cannot replace the handoff by quotient stream equality, a chosen representative, countable choice, host completeness, or a completeness theorem for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5833.1, Definition 5833.2, and Definition 5833.3 seed the finite regular-Cauchy-to-Real completion envelope.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RegularCauchyCompletionEnvelopeUp.RegularCauchyCompletionEnvelopeTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, countable choice, host completeness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, row order, Real handoff, transport stability, replay stability, provenance preservation, and local NameCert preservation for BEDC.Derived.RegularCauchyCompletionEnvelopeUp.

Constructive story: A $\mathsf{RegularCauchyCompletionEnvelope}^{\uparrow}$ packet is generated from StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, terminal Real seal, metric-completion routing, transport, replay, provenance, and local NameCert rows.

Chapter 5834

A Concrete Naming Certificate for $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$

Following Bishop–Bridges constructive locatedness, the $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ packet records the finite interval-selection surface that can be read by Cauchy and Real interval consumers without selecting an endpoint or importing compactness. It sits over the located interval carrier of chapter 394, the rational cell ledger of chapter 354, the optional dyadic refinement rows of chapter 306, the endpoint observation windows of chapter 126, the regular rational readbacks of chapter 301, and the terminal Real seal of chapter 38. The packet names a finite route through displayed interval data; it does not assert endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5834.1 (WeaklyLocatedIntervalSelection carrier)**.** *A* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, Q, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the source interval and its locatedness probes.*
2. *Q is the* $\mathsf{RationalInterval}^{\uparrow}$ *cell ledger listing the rational subinterval cells visible to the selection route.*
3. *D is an optional* $\mathsf{DyadicInterval}^{\uparrow}$ *refinement row; when present, it refines cells already displayed by Q and adds no independent selector.*
4. *W is the paired* $\mathsf{StreamName}^{\uparrow}$ *endpoint-window row through which finite endpoint observations are read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same endpoint windows and rational cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by any Real-facing endpoint or interval consumer.*
7. *H records componentwise* hsame *transport for $L, Q, D, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from located interval probes through rational cells, optional dyadic refinements, endpoint windows, regular readbacks, and the Real seal.*
9. *P is* Pkg *provenance over* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$*,* $\mathsf{LocatedInterval}^{\uparrow}$*,* $\mathsf{RationalInterval}^{\uparrow}$*,* $\mathsf{DyadicInterval}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets componentwise through these ten rows. There is no coordinate for an endpoint selector, an open-cover compactness argument, a trichotomy oracle, quotient equality of Real names, or host Real completeness.

Lean4 ✓ `BEDC.Derived.WeaklyLocatedIntervalSelectionUp.WeaklyLocatedIntervalSelectionTasteGate`
`single carrier alignment`

Theorem 5834.2 (WeaklyLocatedIntervalSelection NameCert obligations)**.** *For every accepted* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

the local NameCert surface consists exactly of the following obligations:

1. *carrier admission for the finite* BHist *packet S and its displayed rows;*
2. *located-cell visibility from the* $\mathsf{LocatedInterval}^{\uparrow}$ *row L into the* $\mathsf{RationalInterval}^{\uparrow}$ *ledger Q;*
3. *finite selection-route exactness through the visible rational cells and any displayed* $\mathsf{DyadicInterval}^{\uparrow}$ *refinements D;*
4. *endpoint-window transport from Q, D into the* $\mathsf{StreamName}^{\uparrow}$ *windows W and* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks R;*
5. *Real-seal nonescape, requiring every Real-facing read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *row E;*
6. *componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance preservation, and local* NameCert *preservation through H, C, P, N.*

No endpoint choice principle, Heine–Borel theorem, trichotomy principle, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem is an obligation or an output of this surface.*

Theorem 5834.3 (Weakly located interval selection route exhaustion)**.** *For every accepted* WeaklyLocatedIntervalSe *carrier*

$$S = (L, Q, D, W, R, E, H, C, P, N),$$

every endpoint, interval, or Real-facing consumer factors through the displayed rows $L, Q, D, W, R, E, H, C, P, N$. Such a consumer cannot read endpoint choice, Heine–Borel compactness, trichotomy, quotient equality of Real names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows of Definition 5834.1. The located interval source is L, the rational cell ledger is Q, and the optional dyadic refinement row is D.

Located-cell visibility therefore passes through L, Q, D. Endpoint or interval reads have no route around these rows because the packet stores no detached endpoint selector or host interval object.

The endpoint windows and regular rational readbacks are then W and R. A Real-facing consumer reaches the seal row E only after those finite windows and readbacks are visible.

Finally H, C, P, N transport, replay, source, and name the same route. Since endpoint choice, Heine–Borel compactness, trichotomy, quotient equality, and ambient completeness are absent from the tuple, the displayed rows exhaust every admitted consumer read. □

Closure status for $\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5834.1, Definition 5834.2, and Definition 5834.3 seed the finite weakly located interval-selection route over chapter 394, chapter 354, chapter 306, chapter 126, chapter 301, and chapter 38.

Formal verification: FormalStatus(NameCert$_{\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close endpoint choice, Heine–Borel compactness, trichotomy, quotient real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-cell visibility, finite selection-route exactness, endpoint-window transport, Real-seal nonescape, transport and replay stability, provenance exhaustion, and local naming for BEDC.Derived.WeaklyLocatedIntervalSelectionUp.

Constructive story: A WeaklyLocatedIntervalSelection$^{\uparrow}$ packet is generated from LocatedInterval rows, RationalInterval cell ledgers, optional DyadicInterval refinements, StreamName endpoint windows, RegSeqRat readbacks, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5835

A Concrete Naming Certificate for $\mathsf{UniformInverseModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UniformInverseModulus}^{\uparrow}$ packet records compact inverse-modulus extraction from uniform homeomorphism data as finite NameCert data. It reads the homeomorphism and inverse-graph surface of chapter 1815, the compact finite-net rows of chapter 129, the graph and pointwise continuity rows of chapter 130, the exported modulus interface of chapter 369, the metric-source and metric-target rows of chapter 4230, and the terminal real-name seal of chapter 38. The chapter names the finite compact route that turns displayed inverse radii into a uniform inverse modulus; it is not a selected inverse function, host function equality, quotient real equality, open-cover compactness, or ambient completeness principle.

Definition 5835.1 (Uniform inverse modulus carrier)**.** *A* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

with the following displayed rows:

1. *X is the compact source row, exposing the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data used to cover the compact domain of the homeomorphism.*
2. *Y is the target metric row, including the distance ledger into which the inverse-modulus estimate is reported.*
3. *F is the* $\mathsf{UniformHomeomorphism}^{\uparrow}$ *graph row from X to Y, carrying the forward graph and the homeomorphism naming data.*
4. *G is the inverse graph row displayed by the same homeomorphism packet.*
5. *E is the positive precision row, read through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint histories when the consumer presents a real tolerance.*
6. *N is the finite net row of the compact source at the scale needed for the inverse-modulus extraction.*
7. *R lists the pointwise inverse radius rows attached to the centers named by N.*
8. *M is the finite radius-fold row, obtained only by folding the finite family R over the index list of N.*
9. *T is the target estimate row, combining inverse-graph exactness, finite-net coverage, the folded radius row, and the target metric triangle ledger.*

10. *H records componentwise* hsame *transport for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, and estimate rows.*
11. *C records the displayed* Cont *replay route from compact homeomorphism data and pointwise inverse radii to the* $\mathsf{UniformModulus}^{\uparrow}$ *export.*
12. *P records* Pkg *provenance over* $\mathsf{UniformInverseModulus}^{\uparrow}$, $\mathsf{UniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *A is the local* $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *row.*

The carrier exposes only displayed compact finite nets, graph rows, inverse-graph rows, endpoint readbacks, finite radius rows, componentwise transport, replay, provenance, and local naming. It has no coordinate for a host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformInverseModulusUp.UniformInverseModulusTasteGate single carrier alignment`

Theorem 5835.2 (Uniform inverse modulus obligations)**.** *For every accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A),$$

the local $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the compact source, target metric, homeomorphism graph, inverse graph, precision, finite-net, inverse-radius, radius-fold, target-estimate, transport, replay, provenance, and local naming rows;*
2. *inverse-graph exactness, requiring every inverse read to pass through the displayed row G attached to the homeomorphism row F;*
3. *compact finite-net handoff, requiring the compact source row X and precision row E to determine the finite row N before any inverse radius is folded;*
4. *finite radius fold, requiring M to be obtained only by folding the pointwise inverse radius rows R over the finite indices of N;*
5. *target estimate, requiring T to combine compact-net coverage, inverse-graph exactness, folded-radius data, and the target metric ledger;*
6. *transport, replay, and provenance, requiring H, C, P, A to source the same displayed route componentwise;*
7. *nonescape, requiring the packet to export no host inverse function, selected inverse choice, host function equality, quotient real equality, trichotomy, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 5835.1. The projection exposes exactly the rows $X, Y, F, G, E, N, R, M, T, H$ so carrier admission and row inventory are finite.

The inverse read is confined to G. Since G is listed as a row of the same homeomorphism packet that supplies F, inverse-graph exactness is consumed as displayed graph data rather than as host function equality or selected inverse choice.

The compact handoff reads X and E before radius data. These rows determine the finite net N, and the pointwise inverse radii in R are indexed only by the centers listed by N.

Fold R over that finite index list to obtain M. The target estimate row T then combines the compact-net coverage, inverse-graph exactness, folded radius, and target metric triangle route. Finally H, C, P, A transport, replay, source, and name the same route, so no excluded equality, choice, open-cover, trichotomy, or completeness coordinate is introduced. □

Theorem 5835.3 (Uniform inverse modulus compact handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *export from an accepted* $\mathsf{UniformInverseModulus}^{\uparrow}$ *carrier*

$$I = (X, Y, F, G, E, N, R, M, T, H, C, P, A)$$

factors through the displayed finite route

$$X, F, G, E \longmapsto N, R \longmapsto M, T \longmapsto H, C, P, A.$$

Thus the inverse-modulus export is a compact finite-net and radius-fold handoff. It exports no host function equality, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a $\mathsf{UniformModulus}^{\uparrow}$ consumer of I. By Definition 5835.2, the consumer must read the compact source, homeomorphism graph, inverse graph, and precision rows before the finite-net row is available.

The only finite-net row is N, and the only pointwise inverse-radius inventory is R. Hence the consumer reaches the exported modulus row M only after the finite fold over R indexed by N.

The target estimate T is downstream of that fold. It cannot be read as a host inverse function or as quotient equality of inverse values, because the carrier contains only displayed graph and metric-ledger rows.

Finally the structural rows H, C, P, A transport, replay, source, and name the same compact route. Since the carrier contains no coordinate for selected inverse choice, trichotomy, open-cover compactness, or ambient completeness, the export factors through exactly the displayed finite-net and radius-fold handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformInverseModulus}^{\uparrow}$ packet is generated from UniformHomeomorphism graph and inverse-graph rows, CompactMetric finite-net rows, ContinuousMap pointwise inverse-radius rows, UniformModulus export rows, MetricSpace distance ledgers, RealUp and RegSeqRat endpoint readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5835.1, Definition 5835.2, and Definition 5835.3 seed the finite compact route from displayed inverse radii to a UniformModulus consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{UniformInverseModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient completeness, host inverse functions, selected inverse choice, quotient real equality, trichotomy, open-cover compactness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inverse-graph exactness, compact finite-net handoff, finite radius fold, target estimate, transport, replay, provenance, and local naming for BEDC.Derived.UniformInverseModulusUp.

Chapter 5836

A Concrete Naming Certificate for $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$

Following Bishop and Bridges on constructive real intervals, the $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet records a bounded interval together with the finite nested-window route used by Real and completion consumers. It sits over the dyadic rational arithmetic of chapter 291, the stream-window interface of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet names a bounded interval principle for displayed nested windows; it does not assert open-cover compactness, a global supremum operator, quotient real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5836.1 (BoundedCompleteRealInterval carrier)**.** *A* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the bounded interval row. It records the inspected lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint names, the boundedness witness visible to the packet, and the local containment predicate for interval windows.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint ledger. It records the dyadic rational cuts used to compare successive windows and to certify the displayed diameter bound.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists exactly the finite endpoint and interval observations that may be inspected by the nested-window route.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same dyadic cuts and stream windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by every Real-facing extraction from B, D, W, R.*
6. *H records componentwise* hsame *transport for bounded interval rows, dyadic ledgers, stream windows, regular rational readbacks, Real seals, replay, provenance, and the local name.*
7. *C records displayed* Cont *replay from a nested-window request through B, D, W, R, E.*
8. *P is the* Pkg *provenance over* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, D, W, R, E, H, C, P, N$. There is no coordinate for open-cover compactness, a selected supremum, host interval equality, quotient equality of Real names, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedCompleteRealIntervalUp.TasteGate.BoundedCompleteRealIntervalTasteGate single carrier alignment`

Theorem 5836.2 (BoundedCompleteRealInterval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier*

$$I = (B, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, package provenance, and local naming row;*
2. *bounded-window visibility, requiring every interval read to pass through the displayed lower-upper row B and its local containment predicate;*
3. *dyadic diameter discipline, requiring D to witness the inspected endpoint comparisons and the shrinking-width estimates used by the extraction route;*
4. *stream-readback exactness, requiring W and R to reuse the same finite endpoint observations and dyadic cuts rather than selecting a second Real representative;*
5. *Real-seal nonescape, requiring every Real-facing output to pass through E after B, D, W, R have all been projected;*
6. *structural preservation through H, C, P, N, with no exported row for open-cover compactness, selected suprema, host interval equality, quotient Real equality, countable choice, trichotomy, or ambient Real completeness.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5836.1. The accepted packet displays exactly the bounded interval row, dyadic endpoint ledger, stream windows, regular rational readback, Real seal, and the structural rows.

Bounded-window visibility is forced by B. A consumer can inspect only the finite lower-upper interval data and the local containment predicate already present in the carrier.

Dyadic diameter discipline is forced by D. The endpoint comparisons and shrinking-width estimates used by the packet are the only numerical rows available to the later stream and readback rows.

The rows W and R are downstream of the same dyadic ledger. They schedule finite observations and repack them as regular rational readback without replacing the bounded interval by a second Real representative.

Finally, E, H, C, P, N close the route. The Real seal is reached only after B, D, W, R, while transport, replay, provenance, and naming preserve the same displayed data. Since no excluded coordinate occurs in the carrier, the local NameCert surface has exactly the listed obligations. □

Theorem 5836.3 (BoundedCompleteRealInterval nested-interval extraction)**.** *Let $I = (B, D, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ *carrier. Every nested-window extraction exported by I factors through the finite route*

$$B \longmapsto D \longmapsto W \longmapsto R \longmapsto E$$

before it can inspect transport, replay, provenance, or local naming rows. The extracted Real-facing read is therefore determined by the bounded interval row, the dyadic diameter ledger, the stream windows, and the regular rational readback visible in the packet.

Proof. Begin with a nested-window extraction from I. The carrier gives no source row before the bounded interval row B, so the lower-upper interval data and local containment predicate are inspected first.

The extraction then projects the dyadic ledger D. This fixes the endpoint comparisons and diameter estimates available to the finite window schedule.

Next the extraction passes through W and R. The stream row exposes only the finite endpoint observations allowed by B, D, and the regular rational row repacks those same observations as the readback consumed by the Real seal.

Only after this route has been displayed can the extraction reach E. The remaining rows H, C, P, N preserve, replay, source, and name the same route, so the exported extraction is exhausted by the displayed bounded-window data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5836.1, Definition 5836.2, and Definition 5836.3 seed the bounded real interval packet over dyadic endpoint ledgers, StreamName windows, RegSeqRat readbacks, and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCompleteRealInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, a global supremum operator, host interval equality, quotient real equality, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded-window visibility, dyadic diameter discipline, stream-readback exactness, Real-seal nonescape, structural preservation, and nested-window extraction for BEDC.Derived.BoundedCompleteRealIntervalUp.

Constructive story: A $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$ packet is generated from a bounded Real interval row, a DyadicRatCore endpoint ledger, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5837

A Concrete Naming Certificate for $\mathsf{FreeGroup}^{\uparrow}$

Following Magnus–Karrass–Solitar, *Combinatorial Group Theory*, the $\mathsf{FreeGroup}^{\uparrow}$ packet records the free group on a displayed finite generator interface as reduced signed words rather than as a quotient presentation. It sits after the word carrier of chapter 36, the unit and multiplication rows of chapter 40, and the inverse-bearing algebra surface of chapter 41. The packet gives the algebra hierarchy a finite reduced-word constructor; it does not identify a quotient group, complete a presentation theory, select normal forms outside the carrier, or assert host equality of generated subgroups.

Definition 5837.1 (Free group reduced-word carrier)**.** *A* $\mathsf{FreeGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, S, W, R, M, I, U, J, H, C, P, N)$$

with the following displayed rows:

1. *G is the finite generator row, exposing exactly the accepted generator names.*
2. *S is the signed-generator row, whose entries are pairs of a generator from G and an orientation bit.*
3. *W is the finite word row over S, read through the list discipline of chapter 36.*
4. *R is the reduction row that deletes adjacent inverse pairs until no adjacent cancellation cell remains.*
5. *M is the multiplication row, concatenating two word rows and then applying R.*
6. *I is the inverse row, reversing the word row and flipping each orientation bit.*
7. *U is the unit row, naming the empty reduced word.*
8. *J is the handoff row into the* $\mathsf{Group}^{\uparrow}$ *surface of chapter 41.*
9. *H records componentwise* hsame *transport for $G, S, W, R, M, I, U, J, C, P, N$.*
10. *C records displayed* Cont *replay from generator inspection through signed words, reduction, multiplication, inverse, unit, and the group handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{FreeGroup}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{Monoid}^{\uparrow}$*,* $\mathsf{Group}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

12. N is the local $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for quotient presentation equality, a host word problem solver, a chosen normal form outside R, unrestricted subgroup generation, or completed presentation theory.

Lean4 ✓ `BEDC.Derived.FreeGroupUp.taste gate`

Theorem 5837.2 (Free group NameCert obligations)**.** *For every accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet* $F = (G, S, W, R, M, I, U, J, H, C, P, N)$*, carrier admission, signed-generator classification, reduced-word stability, multiplication replay, inverse replay, unit exposure, and the group-facing handoff are all exhausted by the displayed rows of Definition 5837.1. Any* $\mathsf{Group}^{\uparrow}$ *consumer receives the finite reduced-word packet through J after $G, S, W, R, M, I, U, H, C, P, N$ have been displayed.*

Proof. Project the carrier. The generator row G, signed row S, word row W, reduction row R, multiplication row M, inverse row I, unit row U, and handoff row J are the only algebraic coordinates available to a consumer.

The list-facing part is finite because W is read through chapter 36. The monoid-facing operation is not a host concatenation oracle: it is the displayed concatenate-then-reduce row M, with U as the empty reduced word and R as the cancellation boundary.

The inverse-bearing group-facing part is then exposed through I and J. Componentwise transport H, replay C, provenance P, and local naming N preserve the same rows under accepted package reads, so no quotient presentation equality, selected external normal form, subgroup-generation closure, or completed presentation theory can enter the certificate. □

Theorem 5837.3 (Free group reduction normal-form stability)**.** *In an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet, the reduction row R is stable under the finite cancellation scan: if a signed word has no adjacent inverse pair after R has acted, then concatenating with the unit row U, applying the inverse row I twice, or transporting the word componentwise by H leaves the same reduced-word row for every downstream read.*

Proof. Begin with the finite word row W and reduction row R. The scan performed by R inspects adjacent signed-generator cells in the displayed list and deletes only inverse neighbours, so the result is a word row whose remaining adjacent cells are not inverse pairs.

The unit cases use the empty word row U. Concatenating a reduced word with U on either side introduces no signed-generator cell, hence no new adjacent inverse pair, and the reduction row returns the same displayed word row.

For inverse stability, I reverses the word and flips orientations. Applying I twice restores the original finite list of signed cells, and R has no new cancellation cell to remove beyond those already absent after the first reduction.

Finally, H transports the generator, signed-word, reduction, multiplication, inverse, and unit rows componentwise. Therefore normal-form stability is a property of the displayed reduced-word rows, not of an external quotient or a selected host representative. □

Theorem 5837.4 (Free group universal word handoff)**.** *Every group-facing consumer of an accepted* $\mathsf{FreeGroup}^{\uparrow}$ *packet must receive the handoff*

$$G, S, W, R, M, I, U, H, C, P, N \longrightarrow J$$

before it reads the target $\mathsf{Group}^{\uparrow}$ *surface. The handoff says that a generator interpretation is consumed as finite signed words, reduction, multiplication, inverse, unit, transport, replay, provenance, and local naming; it does not export quotient group equality, a completed presentation, or a proof that every external group element has a selected word representative.*

Proof. Read the handoff row J through the carrier. Its source rows are exactly the generator row, signed-generator row, word row, reduction row, multiplication row, inverse row, unit row, transport row, replay row, provenance row, and local name row.

Use the obligation surface of Definition 5837.2. Any $\mathsf{Group}^{\uparrow}$ consumer has already passed through the list, monoid, and group sibling surfaces named in the chapter introduction, so the only algebra it sees is the finite reduced-word packet.

The normal-form stability of Definition 5837.3 supplies the cancellation boundary needed by the handoff. Since the packet has no quotient-presentation coordinate, no selected external representative, and no completed presentation row, the universal read is a finite word handoff into $\mathsf{Group}^{\uparrow}$ rather than a claim of quotient group equality. □

Falsifiable prediction: Every accepted $\mathsf{FreeGroup}^{\uparrow}$ consumer that reaches the $\mathsf{Group}^{\uparrow}$ handoff first exposes finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction, multiplication, inverse, unit, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: list, monoid, group: $\mathsf{FreeGroup}^{\uparrow}$ is not bijective to any listed sibling because it binds signed-generator words and cancellation reduction before exposing the inverse-bearing group handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5837.1, Definition 5837.2, Definition 5837.3, and Definition 5837.4 seed the finite reduced-word constructor and its group-facing handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FreeGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient group equality, completed presentation theory, subgroup generation, selected external normal forms, host word-problem decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed-generator classification, cancellation reduction stability, multiplication and inverse replay, unit stability, and group handoff exactness for BEDC.Derived.FreeGroupUp.

Constructive story: A $\mathsf{FreeGroup}^{\uparrow}$ packet is generated from finite generator rows, signed-generator rows, finite list-word rows, cancellation reduction rows, multiplication, inverse, unit, a group-facing handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 5838

A Concrete Naming Certificate for $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet records the finite route by which a continuous function on a located closed real interval receives a Darboux/Riemann integrability certificate. It consumes the pointwise-modulus surface of chapter 59, the tag-bearing finite interval route of chapter 4296, the Darboux upper/lower boundary of chapter 1880, the integrability carrier of chapter 1960, the weak located interval selector of chapter 5834, and the real-name seal of chapter 38. The packet gives integration consumers a finite NameCert surface for continuous functions without importing measure integration, open-cover compactness, quotient equality of real names, selected limiting values, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5838.1 (Continuous Riemann integrability carrier)**.** *A* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Continuous}^{\uparrow}$ *row exposing the pointwise modulus and graph-stability data of the integrand over the displayed interval source.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row naming the finite partition cells, tag containment, window reads, regular readbacks, and terminal real seals used for integration.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *row carrying lower and upper finite sums over the same displayed partition route.*
4. *J is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, read only after the Darboux gap row has been displayed.*
5. *W is the* $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$ *row that exposes located cells and endpoint windows for the closed interval source.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened for interval endpoints, partition tags, and pointwise modulus reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family attached to the same finite windows.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, T, D, J, W, S, R have been displayed.*

9. *H records componentwise* hsame *transport for the continuous row, tagged partition, Darboux rows, integrability handoff, weak interval selector, window and readback rows, real seal, replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from pointwise modulus requests through located cells, finite tagged partitions, Darboux lower/upper rows, the integrability handoff, and the terminal real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{WeaklyLocatedIntervalSelection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, D, J, W, S, R, E, H, C, P, N$. It has no coordinate for full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, or ambient real completeness.

Theorem 5838.2 (Continuous Riemann integrability NameCert obligations)**.** *For every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier*

$$I = (F, T, D, J, W, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the finite* BHist *packet and the displayed rows $F, T, D, J, W, S, R, E, H, C, P, N$;*

2. *interval-source visibility through the weakly located interval row W and its located cell, endpoint-window, and regular-readback route;*

3. *pointwise modulus exposure through the continuous-function row F, with every modulus read tied to displayed interval cells and finite windows;*

4. *finite tagged partition coverage through T, including cell containment, tag containment, and the same* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows;*

5. *Darboux gap control through D, read as finite lower and upper sums over the displayed tagged partition route;*

6. *integrability handoff through J, allowed only after the Darboux gap row and partition coverage rows are visible;*

7. *Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No full integration theory, measure integration, open-cover compactness, quotient equality of real names, selected limiting value, Lean verification, or bridge checking is an obligation or output of this packet.

Proof. Carrier admission is the projection of Definition 5838.1. The accepted packet displays the continuous row, tagged partition, Darboux row, integrability handoff, weak interval selector, window family, regular readback, real seal, transport, replay, provenance, and local name.

The interval and modulus rows are read first. The row W supplies located cells and endpoint windows, while F supplies pointwise modulus data only over those displayed cells and windows.

The finite partition route is then exposed by T. Tags and endpoints are read through S and R, so the integration-facing partition is a finite BEDC object rather than a host partition or a selected representative family.

The Darboux and integrability rows are downstream. The row D computes lower and upper finite sums over the displayed partition route, and J exports integrability evidence only after that gap control is visible.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded analytic objects, the listed rows exhaust the local NameCert surface. □

Theorem 5838.3 (Continuous Riemann integrability Darboux handoff)**.** *Every integration-facing consumer of an accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier factors through the finite partition and Darboux rows*

$$W, F \longmapsto T, S, R \longmapsto D \longmapsto J \longmapsto E, H, C, P, N.$$

Consequently the consumer receives a finite located-cell, pointwise-modulus, tagged-partition, lower/upper Darboux, integrability-handoff, and Real-seal route rather than quotient reals, host equality, choice-selected witnesses, or ambient Real completeness.

Proof. Begin with an integration-facing read. The weak interval selector W and continuous row F force the request through located cells, endpoint windows, and pointwise modulus rows before any partition or sum row can be inspected.

Next read the tagged partition route T, S, R. The finite cells and tags are displayed together with their StreamName windows and RegSeqRat readbacks, so later rows cannot detach the partition from the interval source.

The Darboux row D is then read as lower and upper finite sums over those displayed cells and tags. The integrability row J is therefore a handoff from finite Darboux gap control, not from a quotient real or host equality judgment.

The terminal seal and structural rows E, H, C, P, N preserve that route. No excluded object is stored in the carrier, so every consumer factors through the displayed finite partition and Darboux rows. □

Theorem 5838.4 (Continuous Riemann integrability uniform-modulus handoff)**.** *Every accepted* $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ *carrier whose continuous row factors through a displayed uniform-modulus or composed-modulus route passes the Darboux gap and* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff through the finite chain*

$$\mathsf{Continuous}^{\uparrow},\ \mathsf{UniformContinuityComposition}^{\uparrow},\ \mathsf{TaggedRealPartition}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ \mathsf{RiemannIntegrable}^{\uparrow},\ \mathsf{StreamNar}$$

The handoff therefore reads composed modulus data before the Darboux row and exports no host integral selector, quotient function equality, open-cover compactness argument, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5838.1. The displayed rows are the continuous row F, tagged partition row T, Darboux row D, Riemann-integrable handoff J, weak interval selector W, StreamName windows S, RegSeqRat readbacks R, Real seal E, transport, replay, provenance, and local naming rows.

Use Definition 5838.2 for row exhaustion. When the continuous row is supplied by a uniform-modulus or composed-modulus route, the only admitted modulus source is the displayed $\mathsf{Continuous}^{\uparrow}$ row together with the $\mathsf{UniformContinuityComposition}^{\uparrow}$ composed-modulus handoff of chapter 3891.

Now apply Definition 5838.3. The tagged partition, Darboux lower and upper sums, and integrability row must be read after the same StreamName windows, RegSeqRat readbacks, and terminal Real seal, so the composed-modulus source cannot bypass the finite Darboux route.

Since the carrier stores no host integral selector, quotient function equality, open-cover compactness proof, or ambient completeness object, the listed finite rows are the whole uniform-modulus handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}$.

Constructive story: A $\mathsf{ContinuousRiemannIntegrability}^{\uparrow}$ packet is generated from a Continuous row, TaggedRealPartition row, DarbouxSum row, RiemannIntegrable handoff row, WeaklyLocatedIntervalSelection row, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5838.1, Definition 5838.2, Definition 5838.3, and Definition 5838.4 seed the finite Darboux/Riemann integrability route for continuous functions on located closed real intervals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContinuousRiemannIntegrability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full integration theory, measure integration, open-cover compactness, quotient equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source visibility, pointwise modulus exposure, finite tagged partition coverage, Darboux gap control, integrability handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ContinuousRiemannIntegrabilityUp.

Chapter 5839

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximations, the $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet binds Cauchy mesh refinement, cofinal dyadic mesh selection, and diagonal Real mesh budgets into one finite common-refinement route. It sits over the two-request refinement carrier of chapter 3832, the cofinal dyadic selector of chapter 2254, the diagonal mesh-budget surface of chapter 618, and the terminal real-name seal of chapter 38. The packet gives Real-completeness consumers a finite cofinal mesh handoff without quotient streams, selected limits, countable choice, host real equality, or ambient completeness as an extra row.

Definition 5839.1 (Regular Cauchy mesh cofinality carrier)**.** *A* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, R, D, B, S, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{CauchyMeshRefinement}^{\uparrow}$ *row exposing source Cauchy requests, two displayed dyadic meshes, and their common finite refinement.*
2. *R is the* $\mathsf{CofinalDyadicMesh}^{\uparrow}$ *row refining the incoming observation request to a dyadic mesh window.*
3. *D is the* $\mathsf{RealDiagonalMeshBudget}^{\uparrow}$ *row listing the inspected diagonal indices, selected windows, dyadic tolerances, and aggregation ledger.*
4. *B is the shared budget row aligning the request precision of Q, the cofinal selector depth of R, and the diagonal finite mesh budget of D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window family opened by the shared budget.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback family over the same windows, with all dyadic tolerances read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, R, D, B, S, A have been displayed.*
8. *H records componentwise* hsame *transport for the refinement row, cofinal selector row, diagonal budget row, shared budget, window and readback rows, seal row, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from an observation request through*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E.$$

10. *P is the* Pkg *provenance over* RegularCauchyMeshCofinality$^{\uparrow}$, CauchyMeshRefinement$^{\uparrow}$, CofinalDyadicMesh$^{\uparrow}$, RealDiagonalMeshBudget$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, R, D, B, S, A, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Theorem 5839.2 (Regular Cauchy mesh cofinality NameCert obligations)**.** *For every accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier*

$$K = (Q, R, D, B, S, A, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshCofinality}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy mesh-refinement exposure through Q, cofinal dyadic mesh exposure through R, Real diagonal mesh-budget exposure through D, shared precision and mesh-budget alignment through B, finite* StreamName$^{\uparrow}$ *window exposure through S,* RegSeqRat$^{\uparrow}$ *readback and* DyadicRatCore$^{\uparrow}$ *tolerance visibility through A, terminal* Real$^{\uparrow}$ *seal admission through E, componentwise transport through H, route replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5839.1. The accepted packet displays exactly the Cauchy refinement, cofinal selector, diagonal budget, shared budget, windows, readbacks, seal, transports, replay, provenance, and local name.

The three mesh sources are read before any Real-facing seal. The row Q displays the common refinement of source Cauchy meshes, R displays the cofinal dyadic selector for the observation request, and D displays the finite diagonal mesh budget.

The shared row B aligns these sources. It is the only place where the precision request, cofinal selector depth, and diagonal inspected indices meet, so the window row S and regular readback row A inherit one finite route.

The seal and structural rows E, H, C, P, N preserve that route. Since the carrier has no coordinate for the excluded completion objects, the listed rows exhaust the local certificate surface. □

Theorem 5839.3 (Regular Cauchy mesh cofinality Real-completeness handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshCofinality$^{\uparrow}$ *carrier receives its seal-facing data only through the cofinal finite route*

$$Q, R, D \longmapsto B \longmapsto S \longmapsto A \longmapsto E, H, C, P, N.$$

Thus the consumer factors through Cauchy mesh refinement, cofinal dyadic mesh selection, Real diagonal mesh budgeting, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, and the RealUp seal before any completion-facing conclusion is exported.

Proof. Start from a Real-completeness consumer request. The packet first exposes Q, so the two source Cauchy requests and their common refinement are finite displayed data.

The request then passes through R and D. The cofinal dyadic selector supplies a finite mesh window, while the diagonal mesh-budget row supplies the finite inspected indices and dyadic tolerance ledger.

The budget row B is the common meeting point. It forces the StreamName windows S and RegSeqRat readbacks A to use the same finite refinement and the same dyadic tolerance rows.

Only after this route is displayed may the Real seal E be read. Transport, replay, provenance, and local naming preserve the route, and the excluded quotient or choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ packet is generated from CauchyMeshRefinement, CofinalDyadicMesh, RealDiagonalMeshBudget, a shared precision and mesh-budget row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5839.1, Definition 5839.2, and Definition 5839.3 seed the finite cofinal common-refinement mesh route used by Real-completeness and diagonal-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshCofinality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy mesh-refinement exposure, cofinal dyadic mesh exposure, Real diagonal mesh-budget exposure, shared-budget alignment, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshCofinalityUp.

Chapter 5840

A Concrete Naming Certificate for LocatedCauchyFilterBasis$^{\uparrow}$

Following Bishop and Bridges, the LocatedCauchyFilterBasis$^{\uparrow}$ packet names the finite located-basis interface used by the L10 Real and Completion route before the data are promoted to a located Cauchy filter. It refines the basis-generator surface of chapter 1501 and the located filter route of chapter 1661; its finite reads pass through chapter 126, chapter 301, chapter 291, and finally the downstream seal boundary of chapter 38. The chapter isolates the basis-level located witness, not a completed filter, quotient Cauchy name, selected real, or ambient completeness principle.

Definition 5840.1 (Located Cauchy filter basis carrier)**.** *A* LocatedCauchyFilterBasis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, L, S, R, D, W, T, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* CauchyFilterBasis$^{\uparrow}$ *row naming the finite basis elements and their common-refinement witnesses.*
2. *L is the located-basis request row. It records the displayed dyadic ball request and the basis element that is asserted to meet it.*
3. *S is the* StreamName$^{\uparrow}$ *finite-window row attached to the same located request.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the windows in S.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance ledger for the requested ball, inclusion comparison, and refinement radius.*
6. *W is the finite witness row that the selected basis element meets the requested located ball after S, R, D have been replayed.*
7. *T is the tail-refinement row sending the located basis witness to the filter-level route of* LocatedCauchyFilter$^{\uparrow}$.
8. *E is the downstream* Real$^{\uparrow}$ *seal boundary, reachable only after the basis, located request, finite windows, readback, tolerance, witness, and tail-refinement rows have been consumed.*
9. *H records componentwise* hsame *transport for all displayed rows.*
10. *C records displayed* Cont *replay from a located basis request through $B, L, S, R, D, W, T, E, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedCauchyFilterBasis$^{\uparrow}$, CauchyFilterBasis$^{\uparrow}$, LocatedCauchyFilter$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for completed filters, quotient stream equality, selected representatives, countable choice, trichotomy, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisTasteGate single carrier alignment`

Theorem 5840.2 (Located Cauchy filter basis NameCert obligations)**.** *For every accepted* LocatedCauchyFilterBasis *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the finite basis row, located request row, stream windows, regular rational readback, dyadic tolerance ledger, located witness, tail-refinement handoff, seal boundary, transport, continuation, provenance, and local naming row;*
2. *located basis classification, requiring two accepted reads to agree precisely when their displayed requests admit a common basis refinement after H transports the B, L, S, R, D, W rows;*
3. *finite-window replay, requiring the located request to inspect only the windows in S before the readback row R and tolerance ledger D are available;*
4. *witness exactness, requiring W to show that the displayed basis element meets the requested dyadic ball using only the finite rows S, R, D;*
5. *tail refinement, requiring T to hand the located basis witness to the* $\mathsf{LocatedCauchyFilter}^{\uparrow}$ *route before any real-facing seal can be read;*
6. *nonescape, excluding completed filters, quotient streams, selected representatives, countable choice, trichotomy, host real equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5840.1. The visible rows are exactly the finite basis data, located request, stream windows, rational readback, dyadic ledger, witness, tail refinement, seal boundary, transport, replay, provenance, and name.

The dependency anchors are the basis carrier of chapter 1501 and the located filter consumer of chapter 1661. The finite observations, rational readbacks, tolerance ledgers, and terminal seal are read through chapter 126, chapter 301, chapter 291, and chapter 38.

Classifier comparison is componentwise. Transport by H moves the basis row, located request, window row, readback row, tolerance ledger, and witness row to a common finite comparison surface, while C replays the same order.

The located witness is finite. A request first reads S, then R, then D, and only then may W certify that the displayed basis element meets the requested dyadic ball.

The tail-refinement row T is the only route to $\mathsf{LocatedCauchyFilter}^{\uparrow}$, and E is downstream from that route. Hence P and N cannot export completed filters, quotient streams, selected representatives, choice, trichotomy, host real equality, or real completeness. □

Theorem 5840.3 (Located Cauchy filter basis tail refinement)**.** *Let* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$ *be an accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier. Every located basis read factors through the finite route*

$$B, L \longmapsto S \longmapsto R \longmapsto D \longmapsto W \longmapsto T.$$

The row T then exposes the same witness as a filter-level located tail request for $\mathsf{LocatedCauchyFilter}^{\uparrow}$*, with no bypass around the* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, or* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier tail refinement`

Proof. Start with a located basis read. The rows B and L fix the displayed basis element and the dyadic ball request; neither row contains a sequence readback or a filter-level located-tail witness by itself.

Replay the continuation row C. The request reaches the finite window row S, then the regular rational readback row R, and then the dyadic tolerance ledger D.

Only after those finite rows have been consumed can W certify the located hit. The row T repackages that same finite witness for the $\mathsf{LocatedCauchyFilter}^{\uparrow}$ route, so the handoff cannot skip the StreamName, RegSeqRat, or DyadicRatCore ledgers. □

Theorem 5840.4 (Located Cauchy filter basis Real seal handoff)**.** *For every accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *carrier* $K = (B, L, S, R, D, W, T, E, H, C, P, N)$*, every* $\mathsf{Real}^{\uparrow}$*-facing read exposed by* P *factors through the tail-refinement row* T *and then through the seal boundary* E*. Equivalently, the only real-facing route visible in the packet is*

$$B, L, S, R, D, W \longmapsto T \longmapsto E \longmapsto H, C, P, N.$$

Thus the Real seal is downstream of the located basis witness and cannot be read from basis elements, windows, rational readbacks, or dyadic tolerances alone.

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisCarrier real seal handoff`

Proof. Project the seal boundary row. The carrier lists E after the finite basis route and after the tail-refinement row T, so the seal is not a coordinate of B, L, S, R, D, or W.

Replay the route by C. A real-facing request must first consume the basis row, located request, window row, readback row, tolerance ledger, and located witness, and must then pass through T.

Transport by H preserves the same dependency order under classifier comparison. The provenance and local naming rows therefore record a downstream Real seal handoff rather than a selected limit, quotient equality, host real equality, or completeness witness. □

Theorem 5840.5 (Located Cauchy filter basis nonescape)**.** *An accepted* $\mathsf{LocatedCauchyFilterBasis}^{\uparrow}$ *packet cannot export any consumer route that reaches outside the finite rows*

$$B, L, S, R, D, W, T, E, H, C, P, N.$$

In particular, any consumer asking for a completed filter, quotient Cauchy name, selected representative, countable-choice schedule, trichotomy decision, host real equality, or ambient completeness proof is refused by the carrier and by the local $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}^{\uparrow}}$ *obligations.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyFilterBasisUp.LocatedCauchyFilterBasisRealNonescape`

Proof. Inspect the carrier rows of Definition 5840.1. They include only finite BHist data, hsame transport, Cont replay, dyadic tolerance ledgers, StreamName windows, RegSeqRat readbacks, the located basis witness, the LocatedCauchyFilter tail route, and the Real seal boundary.

The obligation surface of Definition 5840.2 is exhaustive. Every accepted consumer must pass through those listed rows, and no obligation names a hidden completed filter, quotient stream, selected representative, choice schedule, trichotomy decision, host equality, or completeness proof.

The tail and seal theorems keep the downstream routes ordered. Since T and E are reached only after the finite located basis witness is replayed, the packet has no escape path around its displayed finite data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5840.1, Definition 5840.2, Definition 5840.3, Definition 5840.4, and Definition 5840.5 seed the located finite-basis interface between CauchyFilterBasis, LocatedCauchyFilter, StreamName, RegSeqRat, DyadicRatCore, and Real consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCauchyFilterBasis}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed filters, quotient Cauchy names, selected representatives, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, the located basis obligation surface, tail refinement, Real seal handoff, and nonescape for BEDC.Derived.LocatedCauchyFilterBasisUp.

Constructive story: The construction starts from a CauchyFilterBasis row, reads a located dyadic request through StreamName windows, RegSeqRat readbacks, and DyadicRatCore ledgers, packages the finite witness as a LocatedCauchyFilter tail route, and treats Real access only as downstream seal provenance.

Chapter 5841

A Concrete Naming Certificate for $\mathsf{OneSidedLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{OneSidedLimit}^{\uparrow}$ packet records the left- and right-hand real-limit interface as finite BEDC data. It sits next to the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the dyadic tolerance ledger of chapter 291. The packet names only displayed side-tagged approach windows, regular rational reads, dyadic tolerance rows, transport, replay, provenance, and a terminal Real seal; it does not import two-sided completeness, host real equality, quotient streams, trichotomy, or a selected limiting point.

Definition 5841.1 (One-sided limit carrier)**.** *A* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, L, W, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the approach-anchor row, naming the source point or endpoint cell at which the one-sided read is taken.*
2. *L is the side classifier row, with each visible approach window tagged as left-hand or right-hand before any limiting seal is inspected.*
3. *W is the finite approach-window row, listing only windows compatible with the tag in L and the anchor in A.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule opened by the side-tagged windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data observed on S.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the displayed one-sided comparisons.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from A, L, W, S, R, D.*
8. *H records componentwise* hsame *transport for the anchor, side classifier, windows, stream schedule, regular readback, dyadic ledger, Real seal, replay, provenance, and local name.*
9. *C records displayed* Cont *replay from the side-tagged approach windows through the stream, readback, dyadic, and Real-seal rows.*
10. *P is the* Pkg *provenance over* $\mathsf{OneSidedLimit}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $A, L, W, S, R, D, E, H, C, P, N$. It has no coordinate for a two-sided completion theorem, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the named side-tagged windows.

Lean4 ✓ `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate single carrier alignment`

Theorem 5841.2 (One-sided limit NameCert obligations)**.** *For every accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *carrier $K = (A, L, W, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the approach anchor, side classifier, approach windows, stream schedule, regular readback, dyadic ledger, Real seal, transport, replay, provenance, and local naming rows;*
2. *side classifier stability, requiring each transported window to retain its left-hand or right-hand tag before a readback row is consumed;*
3. *dyadic tolerance ledger exactness, requiring D to expose only the tolerance rows demanded by the side-tagged windows in W;*
4. $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring S and R to read only the finite observations selected by A, L, W;*
5. *Real seal nonescape, requiring E to remain downstream of A, L, W, S, R, D and never to serve as a two-sided completeness witness or host real equality row;*
6. *componentwise transport, replay, provenance, and local naming through H, C, P, N.*

No $\mathsf{OneSidedLimit}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5841.1. The accepted packet displays the approach anchor A, side classifier L, finite approach windows W, stream schedule S, regular readback R, dyadic ledger D, Real seal E, and structural rows H, C, P, N.

Side classifier stability is componentwise. A transported window may be read only with the left-hand or right-hand tag already recorded in L, and the replay row C keeps the same tag attached while the stream and regular-rational rows are inspected.

The tolerance and readback rows are downstream of the same finite windows. The ledger D records the dyadic comparisons demanded by W, the schedule S opens only those observations, and R records the regular rational readback.

Finally, the Real seal E is terminal for the route. Since the carrier contains no row for two-sided completeness, quotient real equality, host order trichotomy, countable choice, or a selected limit outside the displayed windows, those objects cannot become local obligations. □

Theorem 5841.3 (One-sided limit Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{OneSidedLimit}^{\uparrow}$ *packet reads the proposed one-sided limit only through the displayed finite route*

$$A, L, W, S, R, D, E, H, C, P, N.$$

Consequently the handoff to $\mathsf{Real}^{\uparrow}$ *is available only after the approach anchor, side classifier, finite approach windows, StreamName schedule, RegSeqRat readback, and DyadicRatCore tolerance ledger have been exposed. The handoff exports no two-sided completeness theorem, quotient stream equality, host real equality, order trichotomy, countable choice principle, or selected limiting point.*

Proof. Project the accepted carrier through Definition 5841.1. Its Real-facing row is the single terminal coordinate E, and the only route to E passes through A, L, W, S, R, D and the structural rows H, C, P, N.

The finite approach windows determine the side of approach before readback begins. Thus a consumer that asks for a Real seal must already have read the side classifier, the compatible windows, the stream schedule, the regular rational row, and the dyadic tolerance ledger.

The remaining coordinates transport and name the same route. Because no carrier row stores a two-sided completion principle, quotient equality, host equality, trichotomy, choice, or an independent selected limit, the Real-seal handoff cannot export any of those objects. □

Falsifiable prediction: Every $\mathsf{OneSidedLimit}^{\uparrow}$ Real-facing read factors through an approach anchor, a left/right side classifier, finite approach windows, a StreamName schedule, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: real, regseqrat, streamname, dyadicratcore: $\mathsf{OneSidedLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a side classifier, finite approach windows, scheduled regular-rational readback, dyadic tolerance data, and a terminal Real seal in one one-sided packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{OneSidedLimit}^{\uparrow}$ packet is generated from an approach anchor, a left/right side classifier, finite approach windows, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5841.1, Definition 5841.2, and Definition 5841.3 seed the finite one-sided real-limit route through side-tagged approach windows and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OneSidedLimit}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.OneSidedLimitUp.OneSidedLimitTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close two-sided real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected limits outside the displayed windows, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, side classifier stability, dyadic tolerance exactness, StreamName and RegSeqRat readback, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.OneSidedLimitUp.

Chapter 5842

A Concrete Naming Certificate for $\mathsf{BertrandSeriesTest}^{\uparrow}$

Following Bertrand's logarithmic refinement of ratio-type convergence tests and Bishop–Bridges constructive real-series practice, the $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet records the finite second-order quotient-defect route that turns displayed positive-series tail data into a regular Cauchy comparison handoff. It sits beside the neighboring quotient-defect surfaces of chapter 1719, chapter 2113, and chapter 2130, refines the ratio and comparison routes of chapter 1461 and chapter 1459, and exports endpoints only through the dyadic, stream, regular-rational, and Real seals of chapter 291, chapter 126, chapter 301, and chapter 38. The packet names finite Bertrand-test evidence only through inspected source windows, adjacent quotient windows, a logarithmic defect ledger, dyadic domination, comparison-test routing, regular rational readback, and terminal Real sealing; it is not host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 5842.1 (Bertrand series test carrier)**.** *A* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source real-series row, exposing only positive-term windows, partial-sum windows, and tail-threshold requests used by the current convergence test.*
2. *Q is the adjacent quotient-window row, recording the inspected neighboring-term quotients on the same displayed tail window.*
3. *L is the logarithmic defect ledger, storing the finite index, logarithmic scale, rational margin, and second-order defect rows used by the Bertrand comparison.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *domination ledger, converting the displayed logarithmic defect and margin rows into a finite majorant or minorant comparison.*
5. *T is the* $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff row that consumes S, Q, L, D before any convergence conclusion is exposed.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source, quotient, logarithmic defect, domination, and comparison rows.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after the comparison handoff and stream windows have been displayed.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint-facing consumers.*
9. *H stores componentwise* hsame *transport for the source, quotient, logarithmic defect, domination, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from each tail request through Q, L, D, T, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{BertrandSeriesTest}^{\uparrow}$, $\mathsf{RaabeDuhamelTest}^{\uparrow}$, $\mathsf{KummerTest}^{\uparrow}$, $\mathsf{GaussTest}^{\uparrow}$, $\mathsf{RegularCauchyRatioTest}^{\uparrow}$, $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, L, D, T, W, R, E, H, C, P, N$. It has no coordinate for host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.BertrandSeriesTestUp.BertrandSeriesTestTasteGate single carrier alignment`

Theorem 5842.2 (Bertrand series test NameCert obligations)**.** *For every accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier*

$$B = (S, Q, L, D, T, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source real-series windows, adjacent quotient windows, logarithmic defect ledger, dyadic domination ledger, comparison-test handoff, stream windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming row;*
2. *quotient-window exactness, requiring every quotient read used by the Bertrand defect to be read from the displayed neighboring positive-term windows in S and Q;*
3. *logarithmic-defect stability, requiring the finite index, logarithmic scale, rational margin, and second-order defect rows in L to remain attached to the accepted tail request;*
4. *dyadic domination routing, requiring D and W to convert the displayed logarithmic defect data into the same finite comparison windows consumed by T;*
5. *regular and Real handoff, requiring endpoint-facing data to pass through T, W, R, E in that order before a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer receives data;*
6. *ledger non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed rows and to exclude host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, and bridge checking.*

Proof. Project the carrier of Definition 5842.1. Carrier admission is exactly the inventory $S, Q, L, D, T, W, R, E, H, C$ so the local certificate has no detached row for an infinite sum, quotient series equality, selected sum, or ambient completeness theorem.

The quotient and logarithmic-defect clauses are read before the comparison handoff. The row S exposes only inspected positive-term and partial-sum windows, the row Q exposes only neighboring-term quotient windows, and L records the finite index, logarithmic scale, rational margin, and second-order defect attached to the current tail request.

The domination route is then forced through D, W, T. The dyadic ledger D converts the displayed logarithmic defect data into comparison evidence, the window row W fixes the finite stream windows, and the comparison-test row T is the only route that may export a regular Cauchy conclusion.

Finally, R and E supply the regular rational and Real-facing endpoint route. The structural rows H, C, P, N preserve, replay, source, and name the same route, so none of them can introduce host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge data. □

Theorem 5842.3 (Bertrand series test tail route)**.** *Every endpoint-facing read of an accepted* $\mathsf{BertrandSeriesTest}^{\uparrow}$ *carrier factors through*

$$\mathsf{RaabeDuhamelTest}^{\uparrow},\ \mathsf{KummerTest}^{\uparrow},\ \mathsf{GaussTest}^{\uparrow},\ \mathsf{RegularCauchyRatioTest}^{\uparrow},\ \mathsf{RegularCauchyComparisonTest}^{\uparrow},\ \mathsf{Dyadic}$$

Concretely, the read consumes source real-series windows and adjacent quotient rows, reads the logarithmic second-order defect ledger, converts that ledger into dyadic domination and stream-window evidence, enters the $\mathsf{RegularCauchyComparisonTest}^{\uparrow}$ *handoff, and reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the displayed readback and seal rows.*

Proof. Begin with an endpoint-facing read of an accepted carrier. By Definition 5842.1, the only source data available before the defect estimate are the source windows in S and the adjacent quotient windows in Q.

The Bertrand-specific data are then read from L and D. The logarithmic ledger supplies the finite index, logarithmic scale, rational margin, and second-order defect, while the dyadic ledger turns those rows into the domination evidence needed for comparison.

The neighboring quotient-defect tests are reference faces in the same lattice. The Raabe–Duhamel, Kummer, and Gauss packets identify the adjacent first-order and auxiliary defect surfaces, while the ratio route supplies the shared adjacent-quotient discipline. The accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint route still has to enter the comparison handoff T.

After T is displayed, the route proceeds through the finite $\mathsf{StreamName}^{\uparrow}$ windows in W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E. The transport, replay, provenance, and naming rows keep the same route, so no host summability, quotient series equality, selected sums, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness can enter. □

Falsifiable prediction: Every accepted $\mathsf{BertrandSeriesTest}^{\uparrow}$ endpoint-facing read factors through source real-series windows, adjacent quotient rows, a logarithmic defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: raabeduhameltest, kummertest, gausstest, regularcauchyratiotest, regularcauchycomparisontest, dyadicratcore, streamname, regseqrat, real: $\mathsf{BertrandSeriesTest}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, comparison-test routing, regular rational readback, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BertrandSeriesTest}^{\uparrow}$ packet is generated from source real-series windows, adjacent quotient windows, a logarithmic second-order defect ledger, dyadic domination, a RegularCauchyComparisonTest handoff, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5842.1, Definition 5842.2, and Definition 5842.3 seed the finite Bertrand convergence-test boundary over displayed source series, quotient, logarithmic defect, comparison, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BertrandSeriesTest}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host summability, quotient series equality, selected sums, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean theorems for carrier admission, quotient-window exactness, logarithmic-defect stability, dyadic domination routing, regular and Real handoff, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.BertrandSeriesTestUp.

Chapter 5843

A Concrete Naming Certificate for $\mathsf{RationalCanonicalForm}^{\uparrow}$

Following Frobenius and standard linear algebra treatments of finite-dimensional operators, the $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet records the rational canonical form of a displayed finite linear operator as BEDC NameCert data. It consumes the finite matrix rows of chapter 49, the linear-map surface of chapter 48, the polynomial source of chapter 50, the module and vector-space carriers of chapter 46 and chapter 47, and the invariant-factor route supplied by chapter 2262. The packet names only a finite certificate surface for companion-block assembly; it does not assert a host Jordan form theorem, algebraic closure, eigenvalue selection, basis choice outside the displayed rows, or a global classification oracle.

Definition 5843.1 (Rational canonical form carrier)**.** *A* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (V, L, M, S, I, B, A, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *or* $\mathsf{Module}^{\uparrow}$ *source row naming the finite underlying carrier and its scalar action.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row naming the endomorphism whose canonical form is being read.*
3. *M is the* $\mathsf{Mat}^{\uparrow}$ *presentation row for the same endomorphism over the displayed finite basis data already carried by V and L.*
4. *S is the consumed* $\mathsf{SmithNormalForm}^{\uparrow}$ *dependency row applied to the polynomial matrix route attached to M.*
5. *I is the invariant-factor* $\mathsf{Poly}^{\uparrow}$ *row, listing monic polynomial factors with the divisibility ledger exposed by S.*
6. *B is the companion-block row, assembling one finite block for each invariant factor in I and recording the row and column positions used by the assembly.*
7. *A is the block-assembly exactness row, stating that the displayed* $\mathsf{Mat}^{\uparrow}$ *presentation is transported to the companion-block matrix through the same* $\mathsf{LinMap}^{\uparrow}$ *and* $\mathsf{Poly}^{\uparrow}$ *rows.*
8. *H stores componentwise* hsame *transport for the vector or module source, linear map, matrix presentation, Smith-normal dependency, invariant factors, companion blocks, assembly row, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a canonical-form request through V, L, M, S, I, B, A and back to the same finite operator surface.*
10. *P is the* Pkg *provenance over* $\mathsf{RationalCanonicalForm}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Module}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{SmithNormalForm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *row.*

The carrier has no coordinate for selected eigenvalues, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle.

Lean4 ✓ `BEDC.Derived.RationalCanonicalFormUp.TasteGate.RationalCanonicalFormTasteGate single carrier alignment`

Theorem 5843.2 (Rational canonical form NameCert obligations)**.** *For every accepted* $\mathsf{RationalCanonicalForm}^{\uparrow}$ *carrier*

$$F = (V, L, M, S, I, B, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$ *surface consists exactly of finite source admission for V, L, M, Smith-normal dependency routing through S, invariant-factor exposure through I, companion-block assembly through B, block-assembly exactness through A, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5843.1. The algebraic source rows are exactly the vector or module carrier V, the endomorphism row L, the finite matrix presentation M, the Smith-normal dependency row S, the invariant-factor row I, and the companion-block assembly rows B, A.

The invariant factors are not read from a host classification theorem. They are exposed as $\mathsf{Poly}^{\uparrow}$ rows whose divisibility and matrix-reduction dependency is routed through $\mathsf{SmithNormalForm}^{\uparrow}$ before any companion block is assembled.

The companion-block row B is finite because it is indexed by the displayed invariant-factor list I. The exactness row A then compares the assembled block matrix with the original $\mathsf{Mat}^{\uparrow}$ presentation through the same $\mathsf{LinMap}^{\uparrow}$ endpoint and componentwise hsame transport.

Finally H, C, P, N transport, replay, source, and name only this finite route. Since the carrier contains no coordinate for eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, or a detached canonical-form oracle, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ consumer that exports a canonical-form matrix while omitting the Smith-normal dependency row, invariant-factor polynomial row, companion-block assembly row, componentwise hsame, displayed Cont, Pkg, or local NameCert rows refutes the carrier of Definition 5843.1.

Independence witness: matrix, linearmap, polynomial, module, vecspace, smith-normal-form: $\mathsf{RationalCanonicalForm}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite operator source, matrix presentation, invariant-factor polynomial ledger, Smith-normal dependency, companion-block assembly, exactness row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCanonicalForm}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RationalCanonicalForm}^{\uparrow}$ packet is generated from VecSpaceUp or ModuleUp source data, a LinearMapUp endomorphism row, a MatrixUp presentation, a SmithNormalFormUp dependency route, invariant-factor PolynomialUp rows, companion-block assembly rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, seed)

Scope closed: Definition 5843.1 and Definition 5843.2 seed the finite rational-canonical-form packet over matrix, linear-map, polynomial, module, vector-space, and Smith-normal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RationalCanonicalForm}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim eigenvalue selection, algebraic closure, host similarity classification, non-displayed basis choice, quotient matrix equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, invariant-factor exposure, Smith-normal dependency routing, companion-block assembly, exactness, transport, replay, provenance, and local naming for BEDC.Derived.RationalCanonicalForn

Chapter 5844

A Concrete Naming Certificate for $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$

The $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet realizes the Rule 110 pattern-classifier branch of chapter 9390 as finite BEDC NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the pattern catalog of chapter 995, the local-stencil observation route of chapter 1036, and the Rule 110 glider catalog of chapter 1045. The packet names finite row windows, transition-table reads, orbit prefixes, pattern tags, transport, replay, provenance, and a local NameCert row; it is not a universality theorem, an infinite-orbit classifier, an undecidable continuation oracle, or a hidden substrate semantics.

Definition 5844.1 (Rule 110 pattern classifier carrier)**.** *A* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, T, O, G, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite row-window row, naming the inspected Rule 110 space-time rectangle and its boundary cells.*
2. *T is the transition-table row, exposing the eight local update cases used to advance the window.*
3. *O is the orbit-prefix row, recording the finite sequence of rows replayed from the initial window by T.*
4. *G is the glider and ether catalog row consumed from* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$*.*
5. *K is the classifier-tag row, assigning each inspected subwindow to a displayed ether, glider, collision, boundary, or unknown tag.*
6. *H stores componentwise* hsame *transport for windows, transition-table rows, orbit prefixes, tags, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a classifier request through W, T, O, G, K.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{CellularPatternCatalog}^{\uparrow}$*,* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for infinite continuation, termination of all continuations, host universality, unbounded pattern search, quotient orbit equality, or a substrate semantics outside the displayed rows.

Theorem 5844.2 (Rule 110 pattern classifier NameCert obligations)**.** *For every accepted* RuleOneTenPatternClassif *carrier*

$$R = (W, T, O, G, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$ *surface consists exactly of row-window admission through* W*, transition-table exposure through* T*, orbit-prefix replay through* O*, catalog lookup through* G*, classifier-tag exactness through* K*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier of Definition 5844.1. The finite cellular rows are exactly the inspected window W, transition table T, orbit prefix O, catalog row G, classifier row K, and the structural rows H, C, P, N.

The transition-table row is read before the orbit prefix, so every displayed successor row is justified by the same eight local Rule 110 cases. The classifier tag is then attached to a finite subwindow already present in O and compared against the catalog row G.

The transport and replay rows preserve that order. Since the carrier contains no infinite continuation coordinate, no all-continuations termination field, no unbounded pattern search, no quotient orbit equality, and no host universality proof, the local NameCert obligations are exhausted by the listed finite rows. □

Falsifiable prediction: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ classifier read that assigns a glider, ether, collision, boundary, or unknown tag without exposing the finite row window, transition-table row, orbit-prefix row, catalog row, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5844.1.

Independence witness: continuationautomaton, finiteprefixautomaton, continuationtermination, typecheckingmembershiptrace, classifiermorphism: $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Rule 110 row windows, the transition table, orbit prefixes, catalog tags, classifier transport, replay, provenance, and local NameCert rows.

The Lean TasteGate file for this packet must provide Nontrivial $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and FieldFaithful $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ instances for the displayed rows $W, T, O, G, K, H, C, P, N$ before the classifier can serve as a verified taste-gate consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ packet is generated from a finite Rule 110 row window, transition table, orbit prefix, pattern catalog, classifier tags, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, seed)

Scope closed: Definition 5844.1 and Definition 5844.2 seed the finite Rule 110 pattern-classifier packet over row-window, transition-table, orbit-prefix, catalog, tag, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenPatternClassifier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a universality theorem, infinite-orbit classification, continuation termination, unbounded pattern search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked row-window admission, transition-table exposure, orbit-prefix replay, catalog lookup, classifier-tag exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPatternClassifierUp.

Chapter 5845

A Concrete Naming Certificate for $\mathsf{RealizerSpace}^{\uparrow}$

Following Weihrauch-style realizability over represented spaces, the $\mathsf{RealizerSpace}^{\uparrow}$ packet records the finite interface at which a represented input name, an admissible Cauchy route, and a target readback are allowed to meet. It sits between the represented-space syntax of chapter 1494 and the admissible Cauchy consumption of chapter 1760, while routing every real-facing read through chapter 1680, chapter 126, chapter 301, chapter 291, and chapter 38. The carrier treats a realizer as displayed finite NameCert data, not as a quotient of names or a selected host function.

Definition 5845.1 (RealizerSpace carrier)**.** *A* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, W, Q, D, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{RepresentedSpace}^{\uparrow}$ *source row whose input observations are read only through its finite representation interface.*
2. *A is the* $\mathsf{AdmissibleRepresentation}^{\uparrow}$ *row exposing the finality route for represented-space consumers.*
3. *K is the* $\mathsf{CauchyRepresentation}^{\uparrow}$ *source row used by the admissible consumer before any real-facing output is read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the input name, Cauchy source, and output request.*
5. *Q is the realizer-relation row; it relates displayed input windows to displayed output windows only through the local finite schedule.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the output readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the stream, regular rational, Cauchy, and dyadic rows have been replayed.*
8. *F is the admissible-finality handoff row, forcing every displayed represented-space consumer to factor through A, K, W, Q, D, E before output data are read.*
9. *H records componentwise* hsame *transport for the represented source, admissible row, Cauchy row, window schedule, relation row, dyadic ledger, real seal, finality row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from an input precision request through X, A, K, W, Q, D, E, F.*
11. *P is the* Pkg *provenance over* $\mathsf{RealizerSpace}^{\uparrow}$, $\mathsf{RepresentedSpace}^{\uparrow}$, $\mathsf{AdmissibleRepresentation}^{\uparrow}$, $\mathsf{CauchyRepresentation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting these rows to a common finite surface. It has no coordinate for quotient names, selected realizers, host function equality, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealizerSpaceUp.RealizerSpaceTasteGate single carrier alignment`

Theorem 5845.2 (RealizerSpace NameCert obligations)**.** *For every accepted* $\mathsf{RealizerSpace}^{\uparrow}$ *carrier $R = (X, A, K, W, Q, D, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed represented source, admissible row, Cauchy source, finite windows, realizer relation, dyadic ledger, real seal, finality handoff, transport, replay, provenance, and local naming;*
2. *realizer-relation exposure, requiring Q to be the only displayed route from input windows to output windows;*
3. *admissible finality, requiring A and F to force consumer access through the same Cauchy route before output data are read;*
4. *the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *handoff through W, K, D, E;*
5. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5845.1. Carrier admission exposes $X, A, K, W, Q, D, E, F, H, C, P, N$ and no further coordinate.

The realizer relation is the row Q. By construction it is read after the represented source, admissible row, Cauchy source, and finite stream schedule, so a consumer cannot replace it by a selected host function or quotient-name map.

Admissible finality is supplied by A and F. These rows force every represented-space consumer to pass through the same Cauchy route, with the regular rational, dyadic, and real-facing rows opened before output data are available.

Finally attach H, C, P, N. Transport, replay, provenance, and local naming preserve exactly the displayed packet. Since none of the excluded objects is a carrier coordinate, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RealizerSpace}^{\uparrow}$ packet is generated from a $\mathsf{RepresentedSpace}^{\uparrow}$ source, an $\mathsf{AdmissibleRepresentation}^{\uparrow}$ row, a $\mathsf{CauchyRepresentation}^{\uparrow}$ source, finite $\mathsf{StreamName}^{\uparrow}$ windows, a realizer relation, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance data, a $\mathsf{Real}^{\uparrow}$ seal, finality handoff, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5845.1 and Definition 5845.2 seed the finite realizer interface from represented input names through admissible Cauchy consumption to real-facing output seals.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealizerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient names, selected realizers, host function equality, countable choice, metric completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, realizer relation exposure, admissible finality, StreamName/RegSeqRat/DyadicRatCore/Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RealizerSpaceUp.

Chapter 5846

A Concrete Naming Certificate for $\mathsf{RuleOneTenCausalCone}^{\uparrow}$

Following Cook's Rule 110 universality theorem and Wolfram's cellular automata classification, the $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet records a finite causal cone rather than an ambient infinite cellular universe. It consumes the cellular automaton carrier of chapter 934, the local update surface of chapter 984, the cellular substrate row of chapter 942, and the Rule 110 glider catalog of chapter 1045. The packet names only finite windows, update-ledger rows, boundary rows, traces, transport, replay, provenance, and a local NameCert row.

Definition 5846.1 (Rule 110 causal-cone carrier)**.** *A* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, U, T, L, B, O, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial finite window row, naming the cells inspected at time 0 together with their left and right indices.*
2. *U is the local Rule 110 update ledger, exposing the eight radius-one cases used to compute successor cells inside the window.*
3. *T is the time-depth row, bounding the number of successor rows present in the cone.*
4. *L is the left boundary row, recording the finite boundary cells needed to replay the left edge of the cone.*
5. *B is the right boundary row, recording the finite boundary cells needed to replay the right edge of the cone.*
6. *O is the observed cone-trace row, listing the finite space-time cells computed from I, U, T, L, B.*
7. *H records componentwise* hsame *transport for the initial window, update ledger, time depth, boundary rows, observed trace, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a cone observation request through I, U, T, L, B, O.*
9. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularLocalUpdateRule}^{\uparrow}$, $\mathsf{CellularSubstrate}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It contains no coordinate for an infinite configuration, a full orbit, a universality proof, unbounded glider search, quotient orbit equality, or a substrate semantics outside the finite cone.

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.TasteGate.RuleOneTenCausalConeTasteGate single carrier alignment`

Theorem 5846.2 (Rule 110 causal-cone NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ *carrier* $R = (I, U, T, L, B, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$ *surface consists exactly of initial-window admission through* I*, local Rule 110 update-ledger exposure through* U*, time-depth bounding through* T*, left and right boundary replay through* L, B*, observed cone trace exposure through* O*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.RuleOneTenCausalConeUp.RuleOneTenCausalConeCarrier namecert obligations`

Proof. Project the carrier of Definition 5846.1. The finite cellular data are exactly I, U, T, L, B, O together with the structural rows H, C, P, N.

The update ledger is read locally. Each displayed successor cell in O is justified by the Rule 110 radius-one case stored in U, with T bounding the number of rows and L, B supplying the finite edge data needed for replay.

Transport and replay preserve that finite cone. The componentwise hsame rows compare only displayed rows, while C replays the same local-window route and P, N source and name it. Since no infinite configuration, full orbit, unbounded search, quotient orbit equality, or host universality coordinate is present, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenCausalCone}^{\uparrow}$ packet is generated from an initial finite window, local Rule 110 update ledger, time-depth row, left and right boundary rows, observed cone trace, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, seed)

Scope closed: Definition 5846.1 and Definition 5846.2 seed the finite Rule 110 causal-cone packet over local windows, update ledgers, boundary rows, observed traces, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RuleOneTenCausalCone}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Rule 110 universality, an ambient infinite cellular universe, a full orbit, unbounded glider search, quotient orbit equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, local update exposure, time-depth bounding, boundary replay, trace exposure, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenCausalConeUp.

Chapter 5847

A Concrete Naming Certificate for $\mathsf{BaireTreeModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on bars, spreads, and choice sequences, the $\mathsf{BaireTreeModulus}^{\uparrow}$ packet records the finite-prefix tree/bar modulus that a continuity consumer may read before it reaches a Baire-space or real-facing handoff. It sits beside the Baire-space prefix-cylinder surface of chapter 334, the choice-sequence prefix rows of chapter 1482, the spread-law admission route of chapter 1532, and the finite-modulus boundary of chapter 3465. The packet is a finite NameCert carrier for tree admission, bar coverage, modulus extraction, and handoff provenance; it is not a host Baire function, a selected infinite branch, a countable-choice principle, a quotient stream, or an ambient real-completeness theorem.

Definition 5847.1 (Baire tree/bar modulus carrier)**.** *A* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (B, S, Q, W, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the finite tree/bar admission row, recording the decidable finite-prefix tree, the bar classifier, and the displayed depth window.*
2. *S is the* $\mathsf{SpreadLaw}^{\uparrow}$ *row that admits the same prefixes and records extension compatibility for the tree.*
3. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *row carrying the displayed choice-sequence prefix answers that the modulus may inspect.*
4. *W is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder window, aligning the tree/bar prefixes with the stream-name schedule and unary answer rows.*
5. *M is the modulus row, extracting a finite prefix bound from B, S, Q, W for the downstream continuity read.*
6. *H records componentwise* hsame *transport for the tree/bar, spread, choice-sequence, Baire-window, modulus, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$B, S, Q, W \longmapsto M.$$

8. *P is the* Pkg *provenance over* $\mathsf{BaireTreeModulus}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{SpreadLaw}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, Q, W, M, H, C, P, N$. It has no coordinate for a completed host stream, a selected branch through the tree, countable choice, quotient stream equality, an external topology on Baire space, or real-completeness data.

Lean4 ✓ `BEDC.Derived.BaireTreeModulusUp.TasteGate.BaireTreeModulusTasteGate single carrier alignment`

Theorem 5847.2 (Baire tree/bar modulus NameCert obligations)**.** *For every accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *carrier $T = (B, S, Q, W, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireTreeModulus}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *tree/bar admission through B, including the decidable finite-prefix tree, bar classifier, and displayed depth window;*
2. *spread and choice compatibility through S, Q, requiring every admitted prefix to be a* $\mathsf{SpreadLaw}^{\uparrow}$ *prefix and every inspected answer to be a displayed* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row;*
3. *Baire prefix-cylinder alignment through W, tying the tree window to the* $\mathsf{BaireSpace}^{\uparrow}$ *stream schedule,* $\mathsf{Nat}^{\uparrow}$ *index rows, unary answers, and prefix-cylinder classifier;*
4. *finite modulus extraction through M, producing the bound used by the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *consumer without selecting a branch;*
5. *transport, replay, provenance, and local naming through H, C, P, N.*

No obligation is stored outside these rows, and no accepted packet exports a completed Baire function, a selected infinite branch, countable choice, quotient stream equality, external topology, or ambient real completeness.

Proof. Project the carrier of Definition 5847.1. The tuple displays exactly the tree/bar row, spread-law row, choice-sequence row, Baire prefix-window row, modulus row, transport, replay, provenance, and local naming data.

The tree and bar obligations are finite. Row B gives the decidable prefix tree, the bar classifier, and the displayed depth window, so the argument ranges over named finite prefixes rather than a selected infinite branch.

Rows S, Q, W supply the cross-chapter route. The spread-law row admits the same prefixes as legal tree extensions, the choice-sequence row supplies only displayed prefix answers, and the Baire-space row aligns those answers with the prefix-cylinder schedule of chapter 334.

The modulus row M is then read as a finite bound for the continuity consumer of chapter 3465. Since M is computed from B, S, Q, W, it cannot introduce a host stream, branch selector, quotient equality, topology, or completeness datum.

Finally, H, C, P, N transport, replay, source, and name the same finite route. These rows exhaust the local NameCert surface, so the listed five obligation rows are the whole accepted packet. □

Theorem 5847.3 (Baire tree/bar modulus continuity handoff)**.** *Every continuity read of an accepted* $\mathsf{BaireTreeModulus}^{\uparrow}$ *packet factors through*

$$B, S, Q, W, M, H, C, P, N.$$

Before any BrouwerContinuityPrinciple$^{\uparrow}$ *or real-facing handoff is available, the consumer must display the finite tree/bar row, the* SpreadLaw$^{\uparrow}$ *admission route, the* ChoiceSequence$^{\uparrow}$ *prefix answers, the* BaireSpace$^{\uparrow}$ *prefix-cylinder window, the extracted modulus row, and the structural provenance rows. The handoff cannot obtain quotient stream equality, a completed Baire function, a selected branch, countable choice, or ambient real completeness from the packet.*

Proof. Start with Definition 5847.2. It forces every accepted consumer to read B, S, Q, W before the modulus row M is available.

The sibling chapters determine the handoff route. The Baire-space surface of chapter 334 supplies the prefix-cylinder window, chapter 1482 supplies displayed choice-sequence answers, and chapter 1532 supplies legal prefix extension.

Once M is displayed, chapter 3465 may consume the finite modulus as its prefix bound. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming.

No excluded datum appears in T. Thus a continuity or real-facing read cannot bypass the tree/bar, spread, choice, Baire-window, modulus, and provenance rows. □

Closure status for NameCert$_{\text{BaireTreeModulus}\uparrow}$.

Constructive story: A BaireTreeModulus$^{\uparrow}$ packet is generated from finite tree/bar admission, SpreadLaw prefix admission, ChoiceSequence prefix answers, BaireSpace prefix-cylinder windows, finite modulus extraction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BaireTreeModulus}\uparrow}$, seed)

Scope closed: Definition 5847.1, Definition 5847.2, and Definition 5847.3 seed the finite-prefix tree/bar modulus route consumed by BaireSpace, ChoiceSequence, SpreadLaw, and BrouwerContinuityPrinciple rows.

Formal verification: FormalStatus(NameCert$_{\text{BaireTreeModulus}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Brouwer continuity as a theorem, fan compactness, selected infinite branches, countable choice, quotient stream equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tree/bar admission, SpreadLaw compatibility, ChoiceSequence prefix inspection, BaireSpace prefix-cylinder alignment, modulus extraction, transport, replay, provenance, and local naming for BEDC.Derived.BaireTreeModulusUp.

Chapter 5848

Located Uniform Cauchy Principle NameCert Construction

Following Bishop and Bridges on regular Cauchy approximations and located apartness data, the $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet records the finite BEDC surface of a uniform Cauchy budget that can be read from dyadic tolerance evidence. It consumes the dyadic tolerance ledger of chapter 291, the finite observation schedules of chapter 126, the rational regular-sequence route of chapter 301, and the terminal seal interface of chapter 38. The packet names only finite budgets, located comparison rows, stability transport, replay, provenance, and a local NameCert row; it does not claim Real completeness, open-cover compactness, quotient stream equality, or selected limits.

Definition 5848.1 (Located uniform Cauchy principle carrier)**.** *A* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic radii and the scale-window comparisons used by the request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, naming the finite observation windows that ask for simultaneous tail control.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-sequence row, exposing the rational approximants and the finite tail-index data visible to the schedule.*
4. *L is the located-comparison row, listing the dyadic inequalities that decide whether two displayed rational approximants are already within the requested tolerance window.*
5. *B is the uniform-budget row, assigning to each displayed dyadic request a finite tail threshold shared by the named observation windows.*
6. *E is the exactness row, stating that every comparison consumed by B is witnessed by the rows D, S, R, L and by no ambient real-equality oracle.*
7. *H stores componentwise* hsame *transport for tolerance rows, schedules, rational sequence rows, located comparisons, uniform budgets, exactness, replay, provenance, and local naming.*

8. *C stores displayed* Cont *replay from a uniform Cauchy request through D, S, R, L, B, E and back to the same finite request surface.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, open-cover compactness, quotient stream equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedUniformCauchyPrincipleUp.LocatedUniformCauchyPrincipleTasteGate`
`single carrier alignment`

Theorem 5848.2 (Located uniform Cauchy obligation surface)**.** *For every accepted* LocatedUniformCauchyPrinciple *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N),$$

the public obligation surface consists exactly of dyadic tolerance admission through D, finite observation scheduling through S, rational regular-sequence exposure through R, located comparison exposure through L, uniform tail-budget assignment through B, exactness through E, componentwise hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5848.1. The finite source rows are exactly the dyadic tolerance row D, the schedule row S, the regular-sequence row R, the located-comparison row L, and the budget and exactness rows B, E.

The uniform budget is read from finite dyadic and rational data. Each threshold in B is indexed by a displayed tolerance request from D and a displayed observation window from S, while L records the comparison evidence that the rational approximants exposed by R fit the same request.

The exactness row E prevents the budget from using a hidden Real-completeness or quotient-stream coordinate. The remaining rows H, C, P, N transport, replay, source, and name only this finite route, so the listed rows exhaust the public obligation surface. □

Theorem 5848.3 (Located uniform Cauchy NameCert)**.** *Every accepted* $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ *carrier*

$$U = (D, S, R, L, B, E, H, C, P, N)$$

presents a local $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$ *whose dependency boundary is confined to* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the finite* $\mathsf{Real}^{\uparrow}$ *terminal seal interface named in P.*

Proof. Use Definition 5848.2. The rows D, S, R, L, B, E provide the complete finite budget route, and H, C, P, N provide stability, replay, provenance, and naming for that route.

Since P names $\mathsf{Real}^{\uparrow}$ only as a terminal seal interface and the carrier contains no field for selected limits, quotient stream equality, open-cover compactness, countable choice, or Real completeness, any consumer of N can recover only the finite located uniform Cauchy certificate described by the displayed rows. □

Falsifiable prediction: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ consumer that exports a uniform Cauchy budget without exposing dyadic tolerance data, stream observation schedules, regular-sequence rational rows, located comparison rows, componentwise hsame, displayed Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5848.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries dyadic tolerance requests, finite observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}$ packet is generated from finite dyadic tolerance requests, stream observation schedules, rational regular-sequence rows, located comparison evidence, shared tail budgets, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5848.1, Definition 5848.2, and Definition 5848.3 seed the finite located uniform Cauchy packet over dyadic, stream-name, regular-sequence, and Real terminal-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformCauchyPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, open-cover compactness, quotient stream equality, selected limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite schedule admission, regular-sequence exposure, located comparison exposure, shared budget exactness, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUniformCauchyPrincipleUp.

Chapter 5849

A Concrete Naming Certificate for $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet records the compact-open reading of Baire-space maps as finite BEDC NameCert data. It sits after the Baire prefix carrier of chapter 334, the compact-open topology surface of chapter 4138, the continuous-map graph rows of chapter 130, and the L10 real-name route through chapter 126, chapter 301, and chapter 38. The packet names only finite prefix cylinders, compact-window tests, graph readbacks, regular rational handoffs, and real seals; it does not import a host function space.

Definition 5849.1 (Baire-space compact-open carrier)**.** *A* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, O, F, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix-cylinder row, exposing only finite natural-index windows.*
2. *O is the* $\mathsf{CompactOpenTopology}^{\uparrow}$ *subbasic-window row, recording the compact-open request as a finite list of prefix-cylinder probes and target-radius budgets.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row whose source inspections all pass through B and whose target inspections pass through the L10 real-name route.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used by the compact-open request and by every target readback.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational handoff row attached to the finite target windows in W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the displayed target values.*
7. *H records componentwise* hsame *transport for $B, O, F, W, R, E, C, P, N$.*
8. *C records displayed* Cont *replay from compact-open requests through prefix windows, graph inspection, regular readback, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{CompactOpenTopology}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, O, F, W, R, E, H, C, P, N$. It has no coordinate for host function equality, quotient streams, arbitrary open-cover compactness, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BaireSpaceCompactOpenUp.BaireSpaceCompactOpenTasteGate single carrier alignment`

Theorem 5849.2 (Baire-space compact-open NameCert obligations)**.** *For every accepted* $\mathsf{BaireSpaceCompactOpen}$ *carrier $K = (B, O, F, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$ *surface consists exactly of carrier admission for the Baire prefix row, compact-open subbasic-window row, continuous-map graph row, StreamName observation schedule, RegSeqRat target readback, Real seal, componentwise transport, displayed continuation replay, provenance, and local naming. Every compact-open read must inspect the prefix row B and the subbasic-window row O before it may consume F, W, R, E.*

Proof. Project the carrier of Definition 5849.1. The rows B and O display the source prefix and compact-open request before any graph or target row is available.

The continuous-map row F is therefore read only after the compact-open window has been fixed. Its target observations pass through the same W, R, E route used by StreamName, RegSeqRat, and Real consumers.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host function equality, quotient streams, open-cover compactness, countable choice, or ambient completeness, the listed rows exhaust the local obligation surface. □

Theorem 5849.3 (Baire-space compact-open prefix handoff)**.** *Every compact-open consumer of an accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{BaireSpace}^{\uparrow} \longrightarrow \mathsf{CompactOpenTopology}^{\uparrow} \longrightarrow \mathsf{ContinuousMap}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The handoff exports only displayed prefix cylinders, compact-open subbasic windows, graph rows, finite target observations, regular readbacks, and real seals.

Proof. Begin with the NameCert obligations of Definition 5849.2. They force the Baire prefix row and compact-open subbasic window to be present before any graph read is replayed.

Use the Baire prefix discipline of chapter 334. It supplies finite source cylinders rather than an infinite host sequence. The compact-open row then consumes those cylinders through the topology surface of chapter 4138.

The target side is the existing L10 route: StreamName windows feed RegSeqRat readbacks, and the Real row seals the displayed target value. Thus the public handoff is exactly the stated finite route. □

Theorem 5849.4 (Baire-space compact-open nonescape)**.** *An accepted* $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ *packet exports no compact-open or function-space read outside $B, O, F, W, R, E, H, C, P, N$. In particular, it cannot export host function equality, quotient streams, open-cover compactness, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a completed space of all Baire-to-real functions.*

Proof. Project the carrier of Definition 5849.1. The only analytic rows are the prefix row, compact-open request row, continuous-map graph row, finite observation schedule, regular readback, and real seal.

Apply the prefix handoff of Definition 5849.3. Every downstream read is routed through the same displayed rows before the structural rows H, C, P, N can replay or name it.

The excluded objects require coordinates that are absent from the carrier. Since transport and replay preserve only displayed rows, the packet cannot leave the finite compact-open route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireSpaceCompactOpen}^{\uparrow}$ packet is generated from BaireSpace prefix rows, compact-open subbasic-window rows, continuous-map graph rows, StreamName observations, RegSeqRat readbacks, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5849.1, Definition 5849.2, Definition 5849.3, and Definition 5849.4 seed the finite compact-open route for Baire-space maps.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireSpaceCompactOpen}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, quotient streams, arbitrary open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-open window exposure, prefix handoff exactness, graph replay, Real seal nonescape, and local naming for BEDC.Derived.BaireSpaceCompactOpenUp.

Chapter 5850

A Concrete Naming Certificate for $\mathsf{CompactApartnessProfile}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet records the finite compact-metric apartness profile used when total-bounded finite nets, completion service rows, and apartness-topology separation rows must be read together without importing open-cover compactness. It sits between the compact metric source of chapter 129, the total-boundedness surface of chapter 127, the complete-metric boundary of chapter 128, and the apartness topology route of chapter 1663. The packet exposes a finite profile for compact-metric consumers: named net centers, mesh bounds, completion handoff, apartness separation, transport, replay, provenance, and local naming, not an arbitrary open-cover compactness theorem.

Definition 5850.1 (Compact apartness profile carrier)**.** *A* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, T, C, S, F, R, H, E, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, finite compactness request, and compact-metric provenance visible to downstream consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, naming the displayed centers, mesh tolerance, and finite coverage ledger.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row, available only as the completion-facing handoff attached to accepted Cauchy-style reads.*
4. *S is the* $\mathsf{ApartnessTopology}^{\uparrow}$ *separation row, recording the apartness neighbourhood and located separation data used by profile consumers.*
5. *F is the finite handoff row tying T to S: each requested mesh cell carries the apartness-visible boundary data that the consumer may inspect.*
6. *R is the completion handoff row tying T and C: finite net reads may request a completion service only through the displayed compact-metric and complete-metric rows.*
7. *H records componentwise* hsame *transport for compact-metric, total-bounded, complete-metric, apartness, handoff, provenance, and local naming rows.*
8. *E records displayed* Cont *replay from a profile request through the finite-net, apartness, and completion handoff rows.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactApartnessProfile}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ApartnessTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, T, C, S, F, R, H, E, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, quotient completion equality, an apartness-free compactness principle, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactApartnessProfileUp.CompactApartnessProfileTasteGate single carrier alignment`

Theorem 5850.2 (Compact apartness profile NameCert obligations)**.** *For every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *carrier*

$$A = (K, T, C, S, F, R, H, E, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$ *surface consists exactly of finite obligation rows for compact-metric source exposure, total-bounded finite-net exposure, complete-metric service exposure, apartness-topology separation exposure, finite-net apartness handoff, finite-net completion handoff, componentwise transport, displayed replay, provenance, and local naming. No obligation row may be discharged by arbitrary open-cover compactness, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real completeness.*

Proof. Project the carrier of Definition 5850.1. The tuple displays the compact-metric source, total-bounded finite-net row, complete-metric service row, apartness-topology separation row, the two handoff rows, transport, replay, provenance, and local naming.

The finite-net obligations are exhausted by T, F, R. The row T supplies centers and mesh bounds, while F and R state exactly how those finite reads are handed to apartness and completion consumers.

The nonescape clauses follow by coordinate inspection. Since no row names an arbitrary cover, a selected subcover, quotient completion equality, apartness-free compactness, or ambient real-completeness theorem, the local NameCert surface cannot export such data. □

Theorem 5850.3 (Compact apartness profile finite-net apartness handoff)**.** *Every accepted* $\mathsf{CompactApartnessProfile}^{\uparrow}$ *packet sends a finite-net read to an apartness-topology consumer only through*

$$K, \quad T, \quad S, \quad F, \quad H, \quad E, \quad P, \quad N.$$

The same packet sends any completion-facing read only through K, T, C, R, H, E, P, N. Thus the profile connects total-boundedness, completion service, and apartness separation by displayed finite rows, without exporting an open-cover compactness principle or an apartness-free completion theorem.

Proof. Start with a finite-net request. The compact-metric source K names the metric side, and the total-bounded row T supplies the finite centers and mesh tolerance inspected by the consumer.

For apartness-facing reads, the only separation coordinate is S, and the only finite-net-to-apartness connector is F. Transport and replay must therefore pass through H, E, P, N after those rows have been selected.

For completion-facing reads, the only service coordinate is C, and the finite-net-to-completion connector is R. Coordinate exhaustion separates this route from the apartness handoff and leaves no coordinate for arbitrary open covers, selected subcovers, apartness-free compactness, quotient completion equality, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5850.1, Definition 5850.2, and Definition 5850.3 seed the finite profile connecting compact-metric, total-bounded, complete-metric, and apartness-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactApartnessProfile}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcover existence, quotient completion equality, apartness-free compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, apartness handoff, completion handoff, transport replay, provenance, and nonescape for BEDC.Derived.CompactApartnessProfileUp.

Constructive story: A $\mathsf{CompactApartnessProfile}^{\uparrow}$ packet is generated from compact-metric source data, total-bounded finite-net data, complete-metric service data, apartness-topology separation data, two handoff rows, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Chapter 5851

A Concrete Naming Certificate for RiemannIntegralCauchy$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegralCauchy$^{\uparrow}$ packet records the finite Cauchy convergence route by which tagged-gauge sums, Darboux upper/lower gaps, integrability evidence, regular rational readback, and a terminal real seal are named as one integration-facing certificate. It sits after the tagged-gauge surface of chapter 4115, the Darboux finite-sum boundary of chapter 1880, the gap witness of chapter 1960, the regular rational sequence row of chapter 301, and the real-name seal of chapter 38. The packet names a finite Cauchy convergence certificate for later integral-value consumers; it does not import host limits, quotient equality, choice, measure completion, or ambient Real$^{\uparrow}$ completeness.

Definition 5851.1 (Riemann integral Cauchy carrier)**.** *A* RiemannIntegralCauchy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, D, I, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* RiemannSumGauge$^{\uparrow}$ *row naming the tagged finite partitions, gauge-radius requests, mesh comparisons, and finite Riemann-sum reads.*
2. *D is the* DarbouxSum$^{\uparrow}$ *row naming the lower and upper finite Darboux sums, their shared partition interface, and their upper-minus-lower gap comparison.*
3. *I is the* RiemannIntegrable$^{\uparrow}$ *gap witness row, recording that the displayed Darboux gap answers the carried tolerance request before an integral value is read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the same tagged-gauge and Darboux-gap route, with no detached sequence coordinate.*
5. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after G, D, I, R have been displayed.*
6. *H records componentwise* hsame *transport for tagged-gauge rows, Darboux rows, integrability rows, regular readback rows, real-seal rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from gauge requests to finite sums, from finite sums to Darboux gaps, from gaps to regular rational readback, and from readback to the terminal real seal.*
8. *P is the* Pkg *provenance over* RiemannIntegralCauchy$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DarbouxSum$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, D, I, R, E, H, C, P, N$. It has no coordinate for a host limit, quotient equality of functions, a choice-selected partition family, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCauchyUp.RiemannIntegralCauchyTasteGate single carrier alignment`

Theorem 5851.2 (Riemann integral Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *carrier $K = (G, D, I, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$ *surface consists exactly of carrier admission, tagged-gauge admission, Darboux upper/lower gap admission, Riemann-integrability gap witnessing,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise transport, displayed replay, provenance, local naming, and non-escape from host limits, quotient equality, choice, measure completion, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 5851.1. The accepted packet displays exactly the tagged-gauge row G, Darboux-sum row D, integrability witness I, regular readback R, real seal E, transports H, replay rows C, provenance P, and local name N.

The Cauchy evidence is finite. The row G supplies tagged-gauge sums, the row D supplies the upper and lower Darboux sums, and the row I records the finite gap witness before any readback or real-facing value appears.

The readback and seal rows are downstream. The row R reads the same finite route as regular rational convergence data, and E seals only that displayed readback. Transport, replay, provenance, and naming preserve the same coordinates, so the listed rows exhaust the local certificate surface. □

Theorem 5851.3 (Riemann integral Cauchy Darboux handoff)**.** *Every accepted* $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ *packet hands a Cauchy integral value to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *only through the finite route*

$$\mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The Darboux handoff uses the same tagged-gauge row cited by chapter 4115 and the same upper/lower gap row cited by chapter 1880; it cannot be replaced by a detached host limit, quotient equality of functions, selected partitions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5851.1. The tagged-gauge row G must be read before the Darboux row D, and D must be read before the integrability gap witness I.

The row I is the only bridge from Darboux gap evidence to a Cauchy readback. Once I is displayed, R repacks the same data as a $\mathsf{RegSeqRat}^{\uparrow}$ row, and E seals that readback as the real-facing value.

The structural rows H, C, P, N transport, replay, source, and name this same route. Since the carrier contains no host limit, quotient equality, selected partition family, measure-completion row, or ambient completeness coordinate, every Cauchy integral handoff factors through the displayed Darboux route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCauchy}^{\uparrow}$ packet is generated from a RiemannSumGaugeUp tagged-gauge row, DarbouxSumUp upper/lower gap row, RiemannIntegrableUp gap witness, RegSeqRatUp readback, RealUp terminal seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannIntegralCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 5851.1, Definition 5851.2, and Definition 5851.3 seed the finite Cauchy integration route over tagged-gauge, Darboux-gap, integrability, regular-rational, and real-seal rows.

Formal verification: FormalStatus(NameCert$_{\text{RiemannIntegralCauchy}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of functions, selected partition families, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-gauge admission, Darboux-gap admission, integrability gap witnessing, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegralCauchyUp.

The inactive cubical-support route is read through the codon-window and DNA boundary-trigger sibling lattice of chapter 2234, chapter 3393, chapter 3470, and chapter 3508.

Chapter 5852

A Concrete Naming Certificate for InactiveAnchorCubicalSupport$^{\uparrow}$

The InactiveAnchorCubicalSupport$^{\uparrow}$ packet realizes the inactive-anchor closure and cubical-support complex of Definition 9462.1307 and Definition 9462.1308 as finite BEDC NameCert data. It consumes the six-coordinate codon source of chapter 2234, the DNA boundary-tile classifier of chapter 3393, the six-boundary coverage route of chapter 3470, and the boundary-trigger closure packet of chapter 3508. The cubical support row K is meaningful only after the codon six-bit window and DNA boundary-tile classifier have both been displayed. The packet records inactive anchors, cubical support cells, the displayed f-vector, and the no-3-cell row as finite data; it is not a biological mechanism, stochastic fitness model, quotient codon semantics, or hidden cubical-homology oracle.

Definition 5852.1 (Inactive-anchor cubical support carrier)**.** *An* InactiveAnchorCubicalSupport$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, T, A, K, F, H, C, P, N)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six bit coordinates used to read support vertices.*
2. *T is the* DnaWindowSixBoundaryTile$^{\uparrow}$ *boundary-tile row exposing the stop/Trp and Ile/Met boundary classifiers consumed by the support window.*
3. *A records the inactive-anchor row, including* UGG*,* AAG*, and* ACA *as displayed inactive boundary vertices rather than as a hidden biological table.*
4. *K records the cubical support complex row, whose cells are exactly the displayed axis-aligned subcubes whose vertices lie in the carried active support.*
5. *F records the displayed active-support f-vector row and no-3-cell row from Definition 9462.1309.*
6. *H records componentwise* hsame *transport for source windows, boundary tiles, inactive anchors, cubical cells, f-vector rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from source windows to boundary tiles, from boundary tiles to inactive-anchor rows, from anchor rows to cubical cells, and from cells to the displayed f-vector and no-3-cell row.*

8. *P is the* Pkg *provenance over* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, A, K, F, H, C, P, N$. It has no coordinate for an empirical code table, stochastic evolutionary law, quotient codon semantics, unrestricted cell enumeration, or host homology computation.

Theorem 5852.2 (Inactive-anchor cubical support NameCert obligations)**.** *For every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *carrier $S = (W, T, A, K, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}}$ *surface consists exactly of source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, displayed f-vector and no-3-cell admission, componentwise transport, displayed replay, provenance, local naming, and non-escape from empirical code tables, stochastic dynamics, quotient codon equality, unrestricted enumeration, and host homology oracles.*

Proof. Project the carrier in Definition 5852.1. The rows $W, T, A, K, F, H, C, P, N$ are the only rows available to an accepted packet.

The source and boundary rows are read first. The $\mathsf{CodonSixBitWindow}^{\uparrow}$ row exposes the six-coordinate source window, and the $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$ row exposes the boundary-tile classifier before any inactive anchor or cubical cell can be consumed.

The finite cubical rows are downstream. The anchor row A displays the inactive vertices, K displays the cells supported by the carried vertices, and F records the active-support f-vector and no-3-cell boundary. Transport, replay, provenance, and local naming preserve that same finite route, so no excluded host object can be exported. □

Theorem 5852.3 (Inactive-anchor cubical support nonescape)**.** *Every accepted* $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow},\ \mathsf{DnaWindowSixBoundaryTile}^{\uparrow} \longmapsto A \longmapsto K \longmapsto F \longmapsto H, C, P, N.$$

The read preserves the displayed active-support f-vector and no-3-cell row, and it cannot escape to an empirical code table, stochastic dynamics, quotient codon semantics, unrestricted cubical enumeration, or host homology oracle.

Proof. Start with the NameCert obligations. They force source-window and boundary-tile admission before any inactive-anchor or cubical-support read.

The anchor and cubical rows then determine the support data. The row A supplies the inactive anchors, K supplies the finite cubical support complex, and F records the displayed f-vector and no-3-cell row.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded host structures is a carrier coordinate, every accepted read remains inside the displayed route. □

Falsifiable prediction: Any accepted $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ packet that changes the displayed active-support f-vector or admits a 3-cell while preserving $W, T, A, K, F, H, C, P, N$ refutes the carrier of Definition 5852.1.

Independence witness: codon-six-bit-window, dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure: $\mathsf{InactiveAnchorCubicalSupport}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries inactive-anchor rows, cubical-support cell rows, displayed f-vector data, no-3-cell refusal, transport, replay, provenance, and local naming.

The Lean follow-up `BEDC.Derived.InactiveAnchorCubicalSupportUp` must provide explicit Nontrivial InactiveAnchorCubicalSupport$^{\uparrow}$ and FieldFaithful InactiveAnchorCubicalSupport$^{\uparrow}$ instances preserving $W, T, A, K, F, H, C, P, N$ as separate displayed fields.

Closure status for NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$.

Constructive story: An InactiveAnchorCubicalSupport$^{\uparrow}$ packet is generated from CodonSixBitWindowUp source windows, DnaWindowSixBoundaryTileUp boundary tiles, inactive-anchor rows, cubical-support cells, displayed f-vector and no-3-cell rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, seed)

Scope closed: Definition 5852.1, Definition 5852.2, and Definition 5852.3 seed the finite inactive-anchor cubical support route over codon-window, boundary-tile, anchor, cubical-cell, and f-vector rows.

Formal verification: FormalStatus(NameCert$_{\text{InactiveAnchorCubicalSupport}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close empirical code tables, stochastic evolutionary dynamics, quotient codon semantics, unrestricted cubical enumeration, host homology computation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window admission, boundary-tile admission, inactive-anchor admission, cubical-support cell admission, f-vector preservation, no-3-cell nonescape, Nontrivial InactiveAnchorCubicalSupportUp, and FieldFaithful InactiveAnchorCubicalSupportUp for BEDC.Derived.InactiveAnchorCubicalSupportUp.

Chapter 5853

A Concrete Naming Certificate for $\mathsf{BoundedRegulatedFunction}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet records a regulated function together with a displayed uniform dyadic envelope, so boundedness-facing consumers receive finite step data rather than a host supremum or selected limit. It sits over the regulated-function schedule of chapter 1906, the dyadic step-function carrier of chapter 428, the terminal real-name seal of chapter 38, and the dyadic mesh ledger of chapter 358. The packet is an integration-facing and variation-facing certificate: it exposes a bounded route through displayed approximants and does not import host function equality, selected limits, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5853.1 (Bounded regulated function carrier)**.** *A* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, B, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, carrying the domain, mesh, step schedule, tolerance-tail ledger, and finite-sum handoff of Definition 1906.1.*
2. *B is the displayed uniform dyadic envelope row. It records a finite dyadic bound and the mesh-indexed scope on which the bound is offered to downstream consumers.*
3. *E records that every visible* $\mathsf{DyadicStepFunction}^{\uparrow}$ *approximant in the regulated schedule lies inside the envelope displayed by B.*
4. *H records componentwise* hsame *transport for the regulated source, envelope, approximant-enclosure, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from a boundedness request through F, B, E, H, P, N.*
6. *P is the* Pkg *provenance over* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, B, E, H, C, P, N. It has no coordinate for host function equality, a selected pointwise representative, an ambient supremum, measure completion, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.BoundedRegulatedFunctionUp.TasteGate.BoundedRegulatedFunctionTasteGate
single carrier alignment

Theorem 5853.2 (Bounded regulated function NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRegulatedFunct}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, uniform dyadic envelope admission, visible step-approximant enclosure, componentwise transport, displayed continuation replay, provenance, local naming, and non-escape from host function equality, selected representatives, ambient suprema, measure completion, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5853.1. The accepted packet displays only the regulated source row F, the uniform dyadic envelope B, the approximant-enclosure row E, transports H, continuations C, provenance P, and local name N.

The regulated source is already finite. By Definition 1906.1, the row F opens through the real-domain row, dyadic mesh ledger, dyadic step-function schedule, tolerance-tail ledger, and finite-sum handoff before any integration-facing consumer can inspect it.

The boundedness data are displayed in B and E. The row B names the dyadic envelope, while E attaches each visible step approximant from the schedule in F to that same envelope. Rows H, C, P, N preserve and name this finite route, so the excluded host structures never become local obligations. □

Theorem 5853.3 (Bounded regulated function uniform bound route)**.** *Every boundedness-facing read exported by an accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegulatedFunction}^{\uparrow} \longmapsto \mathsf{DyadicStepFunction}^{\uparrow} \longmapsto B, E.$$

Thus an integral, bounded-variation, or completion consumer may request the finite dyadic envelope only after the regulated source row, mesh-indexed step approximants, and approximant-enclosure row have been displayed.

Proof. Start with a boundedness-facing request. The continuation row C first replays the regulated source F, so the request reaches the dyadic mesh ledger and step-function schedule through the route fixed in Definition 1906.3.

Next read the envelope row. The row B supplies only a displayed dyadic bound, and the row E records that the visible step approximants supplied by F lie inside that envelope. No separate pointwise function or selected representative is introduced.

Finally use the structural rows. Componentwise transport, replay, provenance, and local naming preserve the same route through F, B, E, H, C, P, N. Since the carrier has no coordinate for an ambient supremum, measure completion, host function equality, selected limit, or $\mathsf{Real}^{\uparrow}$ completeness, the uniform bound route is exhausted by the displayed rows. □

Theorem 5853.4 (Bounded regulated function obligation closure)**.** *For every accepted* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *carrier* $G = (F, B, E, H, C, P, N)$*, the carrier block of Definition 5853.1, the NameCert surface of Definition 5853.2, and the uniform-bound route of Definition 5853.3 jointly close carrier admission, regulated-source admission, dyadic-envelope admission, visible step-approximant enclosure, componentwise transport, displayed replay, provenance, local naming, and ledger refusal for the bounded regulated-function packet. The closure is over the displayed* $\mathsf{RegulatedFunction}^{\uparrow}$ *source route and does not add host function equality, selected representatives, ambient suprema, measure completion, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project $G = (F, B, E, H, C, P, N)$ from the carrier definition. The row F is the only regulated-source coordinate, B is the only dyadic-envelope coordinate, and E is the only visible step-approximant enclosure coordinate.

Apply Definition 5853.2 to identify carrier admission, regulated-source admission, dyadic-envelope admission, approximant enclosure, transport, replay, provenance, naming, and the ledger refusals as the whole local NameCert surface.

Apply Definition 5853.3. That route forces every boundedness-facing read through the displayed $\mathsf{RegulatedFunction}^{\uparrow}$ source row and the finite B, E envelope data before any consumer can use the packet.

The carrier has no further coordinate beyond F, B, E, H, C, P, N. Thus the listed obligations are closed by the three local blocks together with the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ source boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, a displayed uniform dyadic envelope, a visible step-approximant enclosure row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5853.1, Definition 5853.2, and Definition 5853.3 seed the bounded regulated-function route over regulated sources, dyadic step approximants, displayed envelopes, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRegulatedFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, selected representatives, ambient suprema, measure completion, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Lean verification requires carrier admission, regulated-source admission, dyadic-envelope admission, step-approximant enclosure, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRegulatedFunctionUp.

Chapter 5854

A Concrete Naming Certificate for CompactnessSequentialEquivalence$^{\uparrow}$

Following Bishop and Bridges constructive compactness practice, the CompactnessSequentialEquivalence$^{\uparrow}$ packet records the finite BEDC surface on which finite-cover compactness and sequential compactness can be compared without importing an ambient equivalence theorem. It sits between the bounded-subsequence route of chapter 1254, the finite-cover finite-net route of chapter 3471, the compact-metric source of chapter 129, the stream inspection surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert comparison surface for compact metric consumers; it is not a host proof that all compactness notions coincide, not countable choice, not quotient stream equality, and not ambient real completeness.

Definition 5854.1 (Compactness sequential equivalence carrier)**.** *A* CompactnessSequentialEquivalence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row whose metric, total-boundedness, and completeness-facing reads bound both sides of the comparison.*
2. *F is the finite-cover row read through* HeineBorelFiniteNet$^{\uparrow}$*, including the displayed finite net, cover request, and finite subcover ledger.*
3. *B is the bounded-subsequence row read through* BolzanoWeierstrass$^{\uparrow}$*, including the interval-net and selected regular-subsequence route.*
4. *S is the* StreamName$^{\uparrow}$ *window row naming the finite source indices inspected by the sequence side.*
5. *W is the finite subsequence-window row, recording selected indices, selected windows, and monotonicity evidence.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the selected windows and their regular rational tail budgets.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, recording net mesh, cover-radius comparison, and rational slack checks used by both directions.*

8. *Q is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback and cover-radius comparisons are accepted.*
9. *H records componentwise* hsame *transports for K, F, B, S, W, R, D, Q.*
10. *C records displayed* Cont *replays from compact-metric reads to finite covers, finite nets, stream windows, subsequence windows, dyadic tolerance checks, regular rational readbacks, and the terminal real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{BolzanoWeierstrass}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, B, S, W, R, D, Q, H, C, P, N$. It has no coordinate for a global open-cover compactness theorem, an unrestricted sequential compactness principle, countable choice, quotient equality of streams, a selected host subsequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.TasteGate.CompactnessSequentialEquivalenceTaste`
`single carrier alignment`

Theorem 5854.2 (Compactness sequential equivalence NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessSequentialEquivalence}^{\uparrow}$ *carrier*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactnessSequentialEquivalence}^{\uparrow}}$ *surface consists exactly of finite rows for compact-metric source exposure, finite-cover finite-net exposure, bounded-subsequence exposure, stream-window inspection, subsequence-window monotonicity, regular rational readback, dyadic tolerance comparison, Real seal nonescape, componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by a host compactness-equivalence theorem, countable choice, quotient stream equality, an externally selected subsequence, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessSequentialEquivalenceUp.CompactnessSequentialEquivalenceCarrier`
`namecert obligations`

Proof. Project the carrier of Definition 5854.1. Its rows expose exactly the compact-metric source, finite-cover finite-net route, bounded-subsequence route, stream windows, subsequence windows, regular rational readback, dyadic tolerance checks, Real seal, transport, continuation, provenance, and local naming data.

The cover side is finite because F records only the displayed net and the finite subcover ledger from chapter 3471. The sequence side is finite because B, S, and W record only the bounded-subsequence route, inspected stream windows, and selected finite windows.

The comparison between the two sides passes through D and Q. These rows record tolerance and seal data already visible to the packet, so the local NameCert surface has no remaining coordinate in which a host compactness-equivalence theorem, countable choice, quotient stream equality, external subsequence, or ambient Real completeness could enter. □

Theorem 5854.3 (Finite subcover data yields a subsequence-window route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the finite subcover side through F, then the same packet exposes a subsequence-window route through*

$$K, \quad F, \quad S, \quad W, \quad R, \quad D, \quad Q$$

without adding a host subsequence selector, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Begin with the finite-cover row F. It supplies only a finite net, finite cover request, and finite subcover ledger over the compact-metric source K.

Use the stream window S to record the finite source positions inspected by the consumer. The row W then records the selected finite windows and their monotonicity evidence; it is packet data, not a choice functional supplied from outside the carrier.

Finally apply R, D, and Q. The regular rational readback consumes the selected windows, the dyadic row records the cover-radius and tolerance comparisons, and the Real row seals only those accepted finite reads. Therefore the route remains inside the displayed packet and exports none of the excluded host principles. □

Theorem 5854.4 (Subsequence-window data yields a finite-net cover route)**.** *Let*

$$E = (K, F, B, S, W, R, D, Q, H, C, P, N)$$

be an accepted CompactnessSequentialEquivalence$^{\uparrow}$ *carrier. If a downstream compact-metric consumer reads the sequential side through B, S, W, R, then the same packet exposes a finite-net cover route through*

$$K, \quad B, \quad S, \quad W, \quad R, \quad D, \quad F, \quad Q$$

without adding a global open-cover compactness theorem, an unrestricted sequential compactness principle, quotient stream equality, countable choice, or ambient Real completeness row.

Proof. Read the sequential rows first. The bounded-subsequence row B, stream row S, subsequence-window row W, and regular rational row R expose only finite windows and regular readback data.

The dyadic row D transports that finite readback into the tolerance scale used by the finite-net cover ledger. It records mesh, radius, and slack comparisons; it does not create a second compactness principle.

The finite-net route is then the already carried row F, sealed by Q over the same compact-metric source K. Since all rows are components of the same accepted carrier and are replayed by H and C, the consumer obtains a finite-net cover route without any appeal to the excluded host principles. □

Closure status for NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5854.1, Definition 5854.2, Definition 5854.3, and Definition 5854.4 seed the finite comparison boundary between finite-cover compactness and sequential compactness for compact-metric consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactnessSequentialEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host compactness-equivalence theorem, unrestricted sequential compactness, countable choice, quotient stream equality, external subsequence selection, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover and subsequence-window classifier exposure, dyadic tolerance exactness, RegSeqRat handoff exactness, RealUp seal nonescape, and host-principle refusal proofs for BEDC.Derived.CompactnessSequentialEquivalenceUp.

Constructive story: A CompactnessSequentialEquivalence$^{\uparrow}$ packet is generated from CompactMetric source data, HeineBorelFiniteNet finite-cover data, BolzanoWeierstrass bounded-subsequence data, StreamName windows, subsequence windows, RegSeqRat readback, DyadicRatCore tolerance checks, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Chapter 5855

A Concrete Naming Certificate for $\mathsf{ArensEellsSpace}^{\uparrow}$

Following Arens–Eells and Weaver on Lipschitz-free spaces, the $\mathsf{ArensEellsSpace}^{\uparrow}$ packet names the finite molecule ledger by which a pointed metric source is linearized before normed-space and completion consumers read it. It sits beside the metric carrier of chapter 57, the Lipschitz map surface of chapter 410, the normed-space carrier of chapter 1950, the vector-space rows of chapter 47, the Real seal of chapter 38, and the completion boundary of chapter 95. The packet records only displayed molecules, zero-sum accounting, norm budgets, linear handoff rows, transport, replay, provenance, and local naming; it does not quotient a free vector space, select representatives, invoke duality, or assert an ambient Banach completion.

Definition 5855.1 (Arens–Eells space carrier)**.** *An* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, b, L, Z, Q, N, H, C, P, R)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, exposing the finite distance reads used by molecule budgets.*
2. *b is the basepoint row, naming the distinguished source point against which zero-sum molecules are displayed.*
3. *L is the finite molecule ledger, listing signed point-coefficient rows whose support is finite.*
4. *Z is the zero-sum coefficient row, recording that the displayed coefficients in L balance over the basepoint boundary.*
5. *Q is the Arens–Eells norm budget, read through* $\mathsf{Real}^{\uparrow}$ *rows as a finite infimum ledger over decompositions into elementary molecules.*
6. *N is the* $\mathsf{NormedSpace}^{\uparrow}$*-facing handoff row, with vector-space operations routed through* $\mathsf{VecSpace}^{\uparrow}$ *and norm exposure routed through Q.*
7. *H records componentwise* hsame *transport for the metric source, basepoint, molecule ledger, zero-sum row, norm budget, normed-space handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric-source inspection through molecule formation, zero-sum checking, norm-budget accounting, and the normed-space handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{ArensEellsSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. R is the local $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a quotient presentation, selected molecule representative, unrestricted dual space, host linearization theorem, completed Banach envelope, or choice of an optimal decomposition.

Lean4 ✓ `BEDC.Derived.ArensEellsSpaceUp.ArensEellsSpaceTasteGate single carrier alignment`

Theorem 5855.2 (Arens–Eells NameCert obligations)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier* $A = (M, b, L, Z, Q, N, H, C, P, R)$*, the local* $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source row, basepoint row, finite molecule ledger, zero-sum coefficient row, Arens–Eells norm budget, normed-space handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. In particular, every accepted molecule read must first expose* M, b, L, Z*, every accepted norm read must pass through the Real-valued budget* Q*, and every normed-space consumer must enter through* N, H, C, P, R*.*

Proof. Project the carrier of Definition 5855.1. The metric source M and basepoint b are the only rows from which point distances and pointed molecule data can be read.

The molecule ledger is finite. The row L lists the signed point-coefficient entries, while Z records the zero-sum boundary that permits the ledger to behave as an Arens–Eells molecule rather than as an arbitrary formal sum.

The norm budget is likewise displayed. The row Q reads Real-valued distance budgets over finite decompositions of the molecule ledger, and the handoff row N exposes only the resulting vector-space and normed-space interface.

Finally the structural rows H, C, P, R transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the NameCert surface cannot import quotient equality, a selected representative, unrestricted duality, a host linearization theorem, ambient completion, or an optimal-decomposition choice principle. □

Theorem 5855.3 (Arens–Eells Lipschitz linearization handoff)**.** *Let* $A = (M, b, L, Z, Q, N, H, C, P, R)$ *be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Any downstream* $\mathsf{LipschitzMap}^{\uparrow}$ *consumer that asks for a linearized normed-space route must factor through the finite handoff*

$$M, b, L, Z, Q \longmapsto N \longmapsto H, C, P, R.$$

The handoff records only that Lipschitz reads from the pointed metric source may be replayed against finite molecules and their Real-valued norm budgets before entering the $\mathsf{NormedSpace}^{\uparrow}$ *and* $\mathsf{VecSpace}^{\uparrow}$ *surfaces; it does not assert a completed free Banach space, host universal property, quotient by molecule equality, selected extension, or dual representation.*

Proof. Start with a Lipschitz-facing request. The source data are the pointed metric rows M, b, so the request cannot inspect a linear object before the finite molecule ledger is exposed.

Replay the molecule rows by C. The ledger L supplies finite signed point-coefficient data, and the zero-sum row Z supplies the balancing boundary required for molecule consumption.

The normed-space handoff is reached only after the budget row Q has been read. That row supplies the Real-valued Arens–Eells norm account over displayed decompositions, and N is the only row that passes this account to $\mathsf{NormedSpace}^{\uparrow}$ and $\mathsf{VecSpace}^{\uparrow}$ consumers.

The transport, provenance, and local naming rows H, P, R preserve this same route. Therefore the linearization handoff is a finite BEDC route from pointed metric and Lipschitz data to a normed-space interface, not a quotient construction, selected extension, ambient completion, or host universal-property theorem. □

Theorem 5855.4 (Arens–Eells molecule zero-sum transport)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ and $A' = (M', b', L', Z', Q', N', H', C', P', R')$ be accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carriers. If the pointed metric rows M, b, the finite molecule ledger L, and the zero-sum row Z transport componentwise through the displayed* hsame *rows, then the transported packet preserves both the displayed molecule ledger and the zero-sum boundary over the pointed* $\mathsf{Met}^{\uparrow}$ *source. In particular, no transported read may replace L by an unlisted formal sum or replace Z by a host equality proof.*

Proof. Project the two carriers through Definition 5855.1. The only rows that expose molecule data are M, b, L, Z, and the only transport coordinate is the componentwise row H.

Use the NameCert obligations of Definition 5855.2. Molecule admission reads the signed point-coefficient ledger L, while zero-sum admission reads the balancing row Z over the same pointed metric source M, b.

Transport is componentwise. Hence a valid hsame replay can rename or move the displayed rows, but it must carry M, b, L, Z together. Since the carrier has no coordinate for host formal sums or quotient equality, the transported molecule remains the finite displayed ledger with the displayed zero-sum boundary. □

Theorem 5855.5 (Arens–Eells norm-budget stability)**.** *For every accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier $A = (M, b, L, Z, Q, N, H, C, P, R)$, each* $\mathsf{Real}^{\uparrow}$*-valued norm-budget read through Q remains tied to the displayed molecule decomposition rows M, b, L, Z. The read may expose finite decomposition budgets and their Real comparisons, but it cannot select an optimal decomposition, choose a quotient representative, or detach the budget from the molecule ledger named by L.*

Proof. Begin with the carrier projection. The row Q is introduced only after the pointed source, molecule ledger, and zero-sum rows M, b, L, Z have already been displayed.

Apply Definition 5855.2. The theorem makes every accepted norm read pass through Q, and it makes Q the Real-valued account over displayed decompositions rather than an external norm primitive.

Finally use the handoff theorem Definition 5855.3. Downstream normed-space and vector-space consumers reach N only after Q is read. Thus the budget remains attached to the finite molecule data, with no coordinate for an optimal-decomposition selector or quotient representative. □

Theorem 5855.6 (Arens–Eells Lipschitz-free universal window)**.** *Let $A = (M, b, L, Z, Q, N, H, C, P, R)$ be an accepted* $\mathsf{ArensEellsSpace}^{\uparrow}$ *carrier. Every finite consumer window that treats A as a Lipschitz-free handoff factors through*

$$\mathsf{LipschitzMap}^{\uparrow},\ \mathsf{NormedSpace}^{\uparrow},\ \mathsf{VecSpace}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{Completion}^{\uparrow}$$

only by reading the displayed route $M, b, L, Z, Q, N, H, C, P, R$. The packet supplies the finite universal-window data needed by Lipschitz, normed-space, vector-space, Real, and completion-facing consumers, but it refuses the classical ambient Banach-space universal property.

Proof. Start with a consumer that asks for a Lipschitz-free route. By Definition 5855.3, the request must first pass through the pointed metric rows M, b, the molecule ledger L, the zero-sum row Z, and the norm-budget row Q.

The consumer then reaches the normed and vector interfaces through N. The rows H, C, P, R preserve the same finite route by componentwise transport, displayed replay, provenance, and local naming.

The completion-facing boundary has no additional coordinate in the carrier. Therefore a $\mathsf{Completion}^{\uparrow}$ read can only receive the already displayed Lipschitz-to-normed-space window, not a

completed Banach envelope, unrestricted extension theorem, selected quotient representative, or host universal property. □

Falsifiable prediction: Any consumer that claims an $\mathsf{ArensEellsSpace}^{\uparrow}$ molecule, norm, Lipschitz-linearization, normed-space, vector-space, Real, or Completion read while omitting $M, b, L, Z, Q, N, H, C, P$, or the local NameCert row refutes the carrier of Definition 5855.1.

Independence witness: metric, lipschitzmap, normedspace, vecspace, real, completion: $\mathsf{ArensEellsSpace}^{\uparrow}$ is not bijective to any listed sibling because it carries a pointed metric source, finite molecule ledger, zero-sum coefficient boundary, Arens–Eells norm budget, and a normed-space linearization handoff before any completion-facing consumer may read the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}$.

Constructive story: An $\mathsf{ArensEellsSpace}^{\uparrow}$ packet is generated from a pointed metric source, finite molecule ledger, zero-sum coefficient row, Real-valued norm budget, normed-space handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5855.1, Definition 5855.2, and Definition 5855.3 seed the finite molecule and norm-budget route from pointed metric data to Lipschitz and normed-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArensEellsSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient vector-space equality, a selected molecule representative, unrestricted duality, a completed free Banach space, an ambient universal property, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, molecule ledger replay, zero-sum boundary, Real-valued norm-budget exactness, and Lipschitz-to-normed-space handoff for BEDC.Derived.ArensEellsSpaceUp.

Chapter 5856

A Concrete Naming Certificate for $\mathsf{BanachFixedPointStability}^{\uparrow}$

Following Banach and Bishop–Bridges constructive metric analysis, the $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet records the finite stability surface by which a displayed contraction orbit keeps its fixed-point seal attached to explicit Picard and Cauchy-modulus data. It sits after the fixed-point certificate of chapter 396, the iteration ledger of chapter 2359, the contraction surface of chapter 1418, the complete-metric boundary of chapter 128, the modulus schedule of chapter 292, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a stability handoff for fixed-point consumers; it is not ambient real completeness, a selected host fixed point, a quotient orbit, a global contraction theorem, or a choice principle.

Definition 5856.1 (Banach fixed point stability carrier)**.** *A* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, I, T, M, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BanachFixedPoint}^{\uparrow}$ *source row naming the fixed-point certificate whose stability is being inspected.*
2. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row, recording the finite prefix and tail schedule consumed by the stability packet.*
3. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *row that exposes the displayed contraction constant, comparison window, and contraction replay surface.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *dependency row, available only as the completion boundary used by the Picard orbit and not as ambient completeness for* $\mathsf{Real}^{\uparrow}$*.*
5. *Q is the* $\mathsf{CauchyModulus}^{\uparrow}$ *stability schedule for the orbit tail and perturbation tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached only after I, T, M, Q have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after the fixed-point source, Picard orbit, contraction row, completion boundary, modulus schedule, and regular-rational readback have been read.*
8. *H records componentwise* hsame *transport for the displayed source, orbit, contraction, completion, modulus, readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the fixed-point source through $F, I, T, M, Q, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BanachFixedPointStability}^{\uparrow}$, $\mathsf{BanachFixedPoint}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, T, M, Q, R, E, H, C, P, N$. It has no coordinate for a selected host fixed point, quotient orbit equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable choice.*

Lean4 ✓ `BEDC.Derived.BanachFixedPointStabilityUp.BanachFixedPointStabilityTasteGate single carrier alignment`

Theorem 5856.2 (Banach fixed point stability NameCert obligations)**.** *For every accepted* BanachFixedPointStabili *carrier $B = (F, I, T, M, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed fixed-point source F, Picard orbit I, contraction row T, complete-metric dependency M, Cauchy-modulus schedule Q, regular-rational readback R, terminal real seal E, componentwise transport H, displayed replay C, package provenance P, and local naming row N.*

Proof. Project the carrier of Definition 5856.1. The analytic rows visible to a stability consumer are exactly F, I, T, M, Q, R, E, followed by transport, replay, provenance, and naming.

The fixed-point source is not read alone. The Picard row I and contraction row T display the orbit and contraction comparison before the complete-metric dependency M and modulus schedule Q can be consumed.

The Real-facing endpoint is downstream from the same finite route. The regular-rational row R is read only from the displayed orbit, contraction, completion, and modulus rows, and the terminal seal E is read only after R.

Finally H, C, P, N transport, replay, source, and name this packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5856.3 (Banach fixed point stability Picard-Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BanachFixedPoint}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{ContractionMapping}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{Re}$$

The handoff exposes only the displayed fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, terminal real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule.*

Proof. Begin with Definition 5856.2. It places every stability read behind the fixed-point source, Picard orbit, contraction comparison, complete-metric dependency, and explicit modulus schedule.

Next open the handoff to regular-rational data. The rows I, T, M, Q determine the displayed Cauchy tail route, so the $\mathsf{RegSeqRat}^{\uparrow}$ row R is a finite readback from those rows rather than a quotient orbit or host-selected fixed point.

The terminal real seal is last. The $\mathsf{Real}^{\uparrow}$ row E is consumed only after R has been displayed, while H, C, P, N preserve the same dependency order under transport, replay, provenance, and local naming.

Thus any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer sees exactly the stated Picard-Cauchy route and cannot bypass it through an ambient completeness or choice object. □

Theorem 5856.4 (Banach fixed point stability modulus nonescape)**.** *Let $B = (F, I, T, M, Q, R, E, H, C, P, N)$ be an accepted* $\mathsf{BanachFixedPointStability}^{\uparrow}$ *carrier. Every perturbation consumer and every tail-stability consumer of B must read the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule Q before it can read the regular-rational row R or the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E.*

Consequently no such consumer can obtain a selected host fixed point, quotient orbit equality, hidden completion theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness, global contraction soundness, trichotomy, or countable-choice schedule from R or E without first exposing the displayed modulus row Q.*

Proof. Project the carrier of Definition 5856.1. The rows are ordered so that Q is the displayed Cauchy-modulus schedule for the orbit tail and perturbation tolerance, while R and E are downstream readback and seal rows.

Apply Definition 5856.2. It says that the local NameCert surface contains the fixed-point source, Picard orbit, contraction row, complete-metric dependency, Cauchy-modulus schedule, regular-rational readback, and terminal real seal as listed carrier rows, with no excluded coordinate.

Next apply Definition 5856.3. The displayed route reaches $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ only after $\mathsf{CauchyModulus}^{\uparrow}$, so a perturbation or tail-stability request that tries to read R or E has already passed through Q.

The refusal clauses follow because the carrier and handoff theorems export no selected host fixed point, quotient orbit equality, hidden completion theorem, ambient Real completeness, global contraction theorem, trichotomy, or countable-choice row. Thus Q is a nonescape boundary for all tail-stability and perturbation reads. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BanachFixedPointStability}^{\uparrow}$ packet is generated from a BanachFixedPoint source row, PicardIteration orbit, ContractionMapping comparison, CompleteMetric dependency, CauchyModulus schedule, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5856.1, Definition 5856.2, and Definition 5856.3 seed the finite stability handoff from Banach fixed-point data through Picard, contraction, completion, modulus, regular-rational, and real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachFixedPointStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected host fixed points, quotient orbit equality, hidden completion theorems, ambient RealUp completeness, global contraction soundness, trichotomy, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard orbit ordering, contraction comparison stability, complete-metric dependency confinement, Cauchy-modulus exactness, RegSeqRat handoff, RealUp seal confinement, transport stability, continuation replay, and provenance nonescape for BEDC.Derived.BanachFixedPointStabilityUp.

Chapter 5857

A Concrete Naming Certificate for $\mathsf{RealTriangleInequality}^{\uparrow}$

Following Bishop and Bridges' constructive treatment of metric real names, the $\mathsf{RealTriangleInequality}^{\uparrow}$ packet records the finite three-leg route by which a displayed real distance estimate is assembled without importing quotient real equality or an ambient completeness theorem. It sits beside the real-name seal of chapter 38, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the stream-window discipline of chapter 126.

Definition 5857.1 (Real triangle inequality carrier)**.** *A* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, B, M, D, W, H, C, P, N)$$

with the following displayed coordinates:

1. *A and B are the two endpoint* $\mathsf{Real}^{\uparrow}$ *equality rows whose distance comparison is being bounded.*
2. *M is the intermediate* $\mathsf{RegSeqRat}^{\uparrow}$ *row that supplies the middle leg of the finite rational comparison.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger carrying the three displayed radii and the exact middle-leg budget.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family through which the two endpoints and the intermediate row are inspected.*
5. *H stores componentwise* hsame *transport for endpoint rows, the middle readback row, dyadic budgets, windows, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from a consumer request through A, B, M, D, W, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{RealTriangleInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite coordinates. It has no coordinate for host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.

Theorem 5857.2 (Real triangle inequality NameCert obligations)**.** *For an accepted* $\mathsf{RealTriangleInequality}^{\uparrow}$ *carrier $T = (A, B, M, D, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$ *surface consists exactly of endpoint admission through the two displayed* $\mathsf{Real}^{\uparrow}$ *equality rows A, B, dyadic middle-leg*

exactness through the ledger D, $\mathsf{RegSeqRat}^{\uparrow}$ *replay through M,* $\mathsf{StreamName}^{\uparrow}$ *window inspection through W, terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff for the three-leg comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row N. No consumer can escape from this surface to host real equality, quotient streams, selected limits, trichotomy, or ambient completeness.*

Proof. Project the carrier of Definition 5857.1. The two endpoint rows A, B, the intermediate readback row M, the dyadic ledger D, the stream windows W, transport, replay, provenance, and naming row are all displayed coordinates of the finite packet.

Endpoint admission is therefore read only through A and B. The middle leg is read through M, while D records the exact dyadic budget that joins the two endpoint comparisons through the displayed intermediate row.

Replay the consumer request through C after transporting coordinates with H. The replay order exposes the $\mathsf{StreamName}^{\uparrow}$ windows, the $\mathsf{RegSeqRat}^{\uparrow}$ middle row, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger before any terminal $\mathsf{Real}^{\uparrow}$ seal is inspected.

The exclusions are the carrier inventory. Since no coordinate stores host real equality, a quotient stream object, a selected limit, trichotomy data, or an ambient completeness theorem, the local naming certificate cannot export any of them. □

Lean4 ✓ `BEDC.Derived.RealTriangleInequalityUp.RealTriangleInequalityTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTriangleInequality}^{\uparrow}$ packet is generated from two endpoint $\mathsf{Real}^{\uparrow}$ equality rows, one intermediate $\mathsf{RegSeqRat}^{\uparrow}$ row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5857.1 and Definition 5857.2 seed the finite three-leg real route over RealUp, RegSeqRatUp, DyadicRatCoreUp, StreamNameUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealTriangleInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic middle-leg exactness, RegSeqRat replay, RealUp seal handoff, and non-escape for BEDC.Derived.RealTriangleInequalityUp.

$\mathsf{LowerDarbouxSum}^{\uparrow}$ is the lower finite-sum side consumed by chapter 1880 and paired with chapter 4083.

Chapter 5858

A Concrete Naming Certificate for $\mathsf{LowerDarbouxSum}^{\uparrow}$

Following Darboux and Riemann, the $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet isolates the lower-sum side of Darboux integration before the full upper/lower gap is read. It is the lower companion to chapter 4083 and supplies the missing finite half used by chapter 1880. The packet reads the integrability boundary of chapter 1960, the displayed finite partition mesh of chapter 3496, and the step-function bracket rows of chapter 428; it exports no host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Definition 5858.1 (Lower Darboux sum carrier)**.** *A* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (P, B, D, S, I, R, H, C, Q, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, naming the ordered displayed cells, endpoint ledger, width ledger, and refinement handle used by every lower-sum read.*
2. *B is the cell lower-bound ledger. For each displayed cell it records the chosen lower value row and the comparison proof that the carried value is below every admitted sample value in that cell.*
3. *D is the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row over the same partition cells, carrying the step values used to display the lower envelope.*
4. *S is the finite lower Darboux-sum row computed from the displayed widths in P and the lower values exposed by B and D.*
5. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *and* $\mathsf{DarbouxSum}^{\uparrow}$ *handoff row, recording that the lower-sum read is available before any upper-sum, gap, or integral seal is requested.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row, exposing only the finite real-name comparison needed by lower-bound consumers.*
7. *H records componentwise* hsame *transport for the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison row, replay, provenance, and local naming row.*
8. *C records displayed* Cont *replay from partition cells to lower-bound reads, from lower-bound reads to the dyadic step row, from the dyadic step row to the finite lower sum, and from the lower sum to the handoff row.*

9. *Q is the* Pkg *provenance over* $\mathsf{LowerDarbouxSum}^{\uparrow}$, *??*, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicStepFunction}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $P, B, D, S, I, R, H, C, Q, N$. It has no coordinate for a host infimum, a selected mesh limit, an upper Darboux sum, an upper-minus-lower gap, quotient equality of functions, open-cover compactness, or ambient real completeness.

Theorem 5858.2 (Lower Darboux sum NameCert obligations)**.** *For every accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier*

$$L = (P, B, D, S, I, R, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$ *surface consists exactly of the following five finite obligation rows:*

1. *carrier admission for the displayed partition mesh, cell lower-bound ledger, dyadic step row, finite lower sum, Darboux and Riemann handoff, comparison row, transport, replay, provenance, and local naming row;*
2. *partition-mesh stability, requiring each transported refinement of P to preserve the displayed cell-width ledger and to route lower-bound reads through the same refinement handle;*
3. *lower-bound ledger transport, requiring B and D to move componentwise by* hsame *while preserving the stated lower relation on every displayed cell;*
4. *finite lower-sum replay, requiring S to be recomputed from the transported cell widths and transported lower values before any handoff row can be consumed;*
5. *ledger non-escape, requiring I, R, H, C, Q, N to expose only the displayed lower-sum route and to exclude host infima, selected mesh limits, quotient function equality, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, upper-sum rows, and gap rows.*

Proof. Carrier admission is the projection of Definition 5858.1. The accepted packet displays exactly the partition mesh, lower-bound ledger, dyadic step row, finite lower sum, handoff, comparison, transport, replay, provenance, and local naming rows.

Partition-mesh stability is read before any numerical sum. The row P fixes the displayed cells and widths, so a transported refinement can only expose a lower-bound read through the same finite mesh ledger and refinement handle.

The lower-bound ledger is then transported componentwise. The rows B and D move together under H, while replay through C rebuilds the lower envelope and then the finite lower sum row S.

The handoff and non-escape clauses are exhausted by S, I, R, H, C, Q, N. Since no carrier coordinate stores a host infimum, selected mesh limit, quotient function equality, open-cover compactness theorem, ambient completeness row, upper-sum row, or gap row, the local NameCert surface is exactly the five listed obligations. □

Theorem 5858.3 (Lower Darboux finite partition handoff)**.** *Every* $\mathsf{DarbouxSum}^{\uparrow}$ *or* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier $L = (P, B, D, S, I, R, H, C, Q, N)$ must first read the finite partition mesh P, the lower-bound ledger B, the* $\mathsf{DyadicStepFunction}^{\uparrow}$ *row D, the finite lower sum S, and the comparison row R, with transport, replay, provenance, and local naming supplied by H, C, Q, N. The consumer therefore receives a finite partition-and-lower-envelope route rather than a host infimum, a selected mesh limit, quotient function equality, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows upstream of every exported lower-sum read are P, B, D, S, R together with the structural rows H, C, Q, N.

For a $\mathsf{DarbouxSum}^{\uparrow}$ read, the handoff row I is reachable only after the partition mesh, lower-bound ledger, dyadic step row, and finite lower sum have been replayed. For a $\mathsf{RiemannIntegrable}^{\uparrow}$ read, the same row S is the lower finite half that a later upper/lower gap comparison must consume.

Coordinate exhaustion closes the handoff. The packet has no analytic coordinate beyond the displayed rows, so the consumer cannot bypass the finite route through a host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row. □

Theorem 5858.4 (Lower Darboux gap boundary)**.** *Let $L = (P, B, D, S, I, R, H, C, Q, N)$ be an accepted* $\mathsf{LowerDarbouxSum}^{\uparrow}$ *carrier. Any later* $\mathsf{DarbouxSum}^{\uparrow}$ *upper-minus-lower gap row that uses L must take S as its lower finite sum over the displayed partition mesh P and the displayed lower envelope B, D. The lower-sum side of the gap therefore remains fixed by the finite lower packet and cannot be replaced by a host infimum, quotient function equality, selected limiting mesh, open-cover compactness, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or an independently chosen lower bracket.*

Proof. Start with the finite partition handoff of Definition 5858.3. Any Darboux-facing consumer already reads P, B, D, S with H, C, Q, N before it can ask for a gap row.

Compare this lower half with the upper half named by chapter 4083 and the combined surface of chapter 1880. The gap row can combine a transported upper finite sum with the transported lower finite sum, but the lower input remains the row S replayed from the displayed mesh and lower envelope.

The boundary is therefore finite and local. Since L contains no coordinate for an alternate lower bracket, host infimum, selected mesh limit, quotient equality, compactness theorem, or ambient completeness row, no later gap consumer can replace the lower side without leaving the displayed packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet is generated from finite partition-mesh rows, cell lower-bound ledgers, dyadic step-function rows, finite lower-sum rows, Darboux and Riemann handoff rows, Real comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5858.1, Definition 5858.2, Definition 5858.3, and Definition 5858.4 seed the finite lower-sum packet over displayed partition meshes, cell lower-bound reads, dyadic-step rows, finite-sum rows, Darboux and Riemann handoff, $\mathsf{Real}^{\uparrow}$ comparison, transport, replay, provenance, local naming, and Darboux gap-boundary rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LowerDarbouxSum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host infima, selected mesh limits, quotient function equality, open-cover compactness, upper Darboux sums, upper-minus-lower gap construction, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh stability, lower-bound ledger transport, finite lower-sum replay, finite partition handoff, Darboux gap boundary, componentwise transport, displayed replay, provenance, local naming, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.LowerDarbouxSumUp.

The $\mathsf{LowerDarbouxSum}^{\uparrow}$ packet consumes the shared $\mathsf{DarbouxSum}^{\uparrow}$ partition ledger of chapter 1880 and is paired with **??** through the same finite interval, cell-width, and partition rows of chapter 4083. A lower-sum consumer therefore reads the lower envelope as one half of the displayed Darboux partition lattice rather than as an isolated kernel row.

Chapter 5859

A Concrete Naming Certificate for $\mathsf{BoundedMetricSpace}^{\uparrow}$

Following Hausdorff and Kelley metric-space practice, the $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet records the bounded metric-space interface as finite BEDC NameCert data. It sits between the metric carrier of chapter 4230, the center-radius containment row of chapter 1450, the bounded-subset surface of chapter 1757, the diameter handoff of chapter 1503, the ordered-real bound route of chapter 766, the terminal real-name seal of chapter 38, and the completion-facing consumers of chapter 95 and chapter 1506. The packet supplies finite-net, Cauchy, compact-metric, and diameter consumers with a displayed bounded metric source without importing compactness, host suprema, selected centers, quotient equality, or ambient completeness.

Definition 5859.1 (BoundedMetricSpace carrier)**.** *A* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, B, D, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *carrier row. It supplies the point classifier, metric-source data, real-valued distance route, zero-distance classifier, symmetry row, and triangle row.*
2. *B is the bounded-containment row. It records the* $\mathsf{MetricBall}^{\uparrow}$ *center-radius ledger together with the* $\mathsf{BoundedSet}^{\uparrow}$ *membership classifier that every admitted point must read before boundedness is exposed.*
3. *D is the* $\mathsf{MetricDiameter}^{\uparrow}$ *row. It names the finite diameter-bound handoff consumed by diameter, finite-net, Cauchy, and compact-metric readers.*
4. *R is the real-bound readback row. It routes the radius and diameter bounds through* $\mathsf{Real}^{\uparrow}$ *and the ordered comparison surface of* $\mathsf{RealAlgOrder}^{\uparrow}$*.*
5. *H records componentwise* hsame *transport for the metric carrier, containment row, diameter row, real-bound readback, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from a bounded-metric request through metric admission, bounded-set containment, metric-ball comparison, diameter-bound readback, ordered-real comparison, and completion-facing handoff.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundedMetricSpace}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{MetricBall}^{\uparrow}$*,* $\mathsf{BoundedSet}^{\uparrow}$*,* $\mathsf{MetricDiameter}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RealAlgOrder}^{\uparrow}$*,* $\mathsf{Completion}^{\uparrow}$*,* $\mathsf{MetricCompletion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*

8. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through X, B, D, R, H, C, P, N. It has no coordinate for a host subset, host supremum, selected bounding center, selected diameter witness, compactness theorem, quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BoundedMetricSpaceUp.BoundedMetricSpaceTasteGate single carrier alignment`

Theorem 5859.2 (BoundedMetricSpace NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *carrier $Q = (X, B, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$ *surface consists exactly of the following obligation families:*

1. *metric-source admission for the displayed* $\mathsf{Met}^{\uparrow}$ *row X;*
2. *bounded-containment admission for the* $\mathsf{MetricBall}^{\uparrow}$ *and* $\mathsf{BoundedSet}^{\uparrow}$ *rows stored in B;*
3. *diameter-bound admission for the* $\mathsf{MetricDiameter}^{\uparrow}$ *row D;*
4. *real-bound readback through* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RealAlgOrder}^{\uparrow}$ *in R;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

Consequently a downstream finite-net, Cauchy, compact-metric, diameter, or completion-facing consumer may read bounded metric-space data only through the displayed metric, containment, diameter, ordered-real, and structural rows. The packet exports no host supremum, compactness theorem, selected bound, quotient completion, countable-choice selector, or ambient real completeness coordinate.

Proof. Project the carrier of Definition 5859.1. The only analytic rows are the metric source X, the bounded-containment row B, the diameter row D, and the real-bound readback row R.

The containment side is read first. The row B combines the center-radius ledger of chapter 1450 with the bounded-subset classifier of chapter 1757, so membership containment is visible before any bounded-space consumer receives the packet.

The diameter and real-bound rows are downstream from that containment. The row D supplies the sibling handoff of chapter 1503, and R records that the bound is read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ rather than through a host supremum.

Finally H, C, P, N transport, replay, source, and name the same displayed rows. Since no excluded analytic object is a coordinate of Q, the listed obligation families exhaust the local NameCert surface. □

Theorem 5859.3 (BoundedMetricSpace diameter-bound handoff)**.** *Every diameter, finite-net, Cauchy, compact-metric, or completion-facing read accepted from a* $\mathsf{BoundedMetricSpace}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{MetricBall}^{\uparrow}, \mathsf{BoundedSet}^{\uparrow} \longmapsto \mathsf{MetricDiameter}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}, \mathsf{RealAlgOrder}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow}, \mathsf{MetricCompletio}$$

The exported handoff is a displayed bounded metric source with containment and diameter-bound rows already visible. It does not provide compactness, a host supremum, a selected bounding center, a selected maximum pair, quotient completion equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted packet $Q = (X, B, D, R, H, C, P, N)$. A consumer first opens the metric row X, so every subsequent boundedness or diameter read remains attached to the same point classifier and distance route.

Read containment through B. This exposes the center-radius and bounded-subset rows before any finite-net or Cauchy consumer can treat the source as bounded.

Next read the diameter row D and the ordered-real readback R. These rows hand the displayed bound to the sibling diameter packet and to the $\mathsf{Real}^{\uparrow}$ and $\mathsf{RealAlgOrder}^{\uparrow}$ comparison surfaces, with no host supremum or selected maximum pair.

The completion-facing route is only a consumer of that finite surface. The structural rows H, C, P, N preserve the same path under transport and replay, so no quotient completion, choice schedule, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5859.1, Definition 5859.2, and Definition 5859.3 seed the finite bounded metric-space packet over metric, ball-containment, bounded-subset, diameter, ordered-real, and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim compactness, host suprema, selected bounds, quotient completions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked metric admission, bounded-containment admission, diameter-bound handoff, real-bound readback, transport, replay, provenance, and local naming for BEDC.Derived.BoundedMetricSpaceUp.

Constructive story: A $\mathsf{BoundedMetricSpace}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier, MetricBallUp and BoundedSetUp containment rows, a MetricDiameterUp bound row, RealUp and RealAlgOrderUp readback, completion-facing handoff data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The Legendre–Fenchel envelope concrete instance is housed in its thematic spine so the root concrete-instance index stays structural; its convex-optimization route is read through chapter 208, chapter 321, chapter 3713, chapter 3563, and chapter 300.

Chapter 5860

A Concrete Naming Certificate for $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$

Following Rockafellar, the $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet names the finite convex-envelope surface obtained by reading epigraph windows, supporting affine minorants, convex-conjugate data, and biconjugate comparison rows before any optimization consumer may inspect the result. It sits after the convex-set carrier of chapter 208, the Fenchel-duality ledger of chapter 321, and the proximal-resolvent handoff of chapter 3713; it also exposes KKT-facing and convex-conjugate-facing rows for certificates that need a displayed envelope without importing an ambient lower-semicontinuous hull, host supremum operator, quotient function space, or choice-selected minimizer.

Definition 5860.1 (Legendre–Fenchel envelope carrier)**.** *A* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (W, A, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the epigraph window row, read through the* $\mathsf{ConvexSet}^{\uparrow}$ *carrier and recording the finite domain, value, and order comparisons visible to the envelope request.*
2. *A is the supporting affine minorant ledger, listing the finitely displayed affine lower bounds and their compatibility with the same epigraph window.*
3. *D is the convex conjugate readback row, exposing the pairing, dual-vector, and budgeted conjugacy data that a* $\mathsf{ConvexConjugate}^{\uparrow}$ *or* $\mathsf{FenchelDuality}^{\uparrow}$ *consumer may inspect.*
4. *B is the biconjugate-envelope row, recording the displayed comparison between the primal epigraph window and the conjugate readback route.*
5. *R is the proximal consumer row, allowing a* $\mathsf{ProximalMap}^{\uparrow}$ *request to read the envelope only after the window, minorant, conjugate, and biconjugate rows have been displayed.*
6. *K is the KKT and Fenchel ledger row, tying stationarity, supporting minorants, and dual comparison data to the* $\mathsf{KKT}^{\uparrow}$ *and* $\mathsf{FenchelDuality}^{\uparrow}$ *packets.*
7. *H records componentwise* hsame *transport for the epigraph window, affine minorants, conjugate readback, biconjugate envelope, proximal consumer, KKT and Fenchel ledger, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from epigraph admission through supporting minorants, conjugate readback, biconjugate comparison, proximal consumption, and KKT/Fenchel ledger inspection.*

9. *P is the* Pkg *provenance over* $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{ConvexConjugate}^{\uparrow}$, $\mathsf{FenchelDuality}^{\uparrow}$, $\mathsf{ProximalMap}^{\uparrow}$, $\mathsf{KKT}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, A, D, B, R, K, H, C, P, N$. It has no coordinate for an external lower-semicontinuous hull, host biconjugation theorem, selected supremum witness, quotient equality of functions, ambient locally convex topology, or global minimization principle.

Falsifiable prediction: Every accepted $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ consumer read factors through an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT/Fenchel ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: convexset, fenchelduality, proximal-map, convex-conjugate, kkt: LegendreFenchelEnvelop is not bijective to any listed sibling because it jointly carries an epigraph window, supporting affine minorants, conjugate readback, biconjugate envelope comparison, proximal consumer route, and KKT/Fenchel ledger in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendreFenchelEnvelope}^{\uparrow}$ packet is generated from an epigraph window, supporting affine minorant ledger, convex conjugate readback, biconjugate-envelope row, proximal consumer row, KKT and Fenchel ledger row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 5860.1 seeds the finite envelope certificate surface over convex-set, convex-conjugate, Fenchel-duality, proximal-map, and KKT rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LegendreFenchelEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host biconjugation theorem, lower-semicontinuous hull existence, selected supporting functional, quotient function equality, ambient locally convex topology, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, affine-minorant ledger exactness, conjugate readback, biconjugate-envelope comparison, proximal consumer boundary, KKT/Fenchel ledger visibility, transport, replay, provenance, and local naming for BEDC.Derived.LegendreFenchelEnvelopeUp.

Chapter 5861

A Concrete Naming Certificate for $\mathsf{PrefixTree}^{\uparrow}$

Following Brouwer and Kleene on spreads and finite choice sequences, the $\mathsf{PrefixTree}^{\uparrow}$ packet records the finite prefix-tree carrier used before StreamName and compact fan routes ask for branch-facing data. It sits between the finite window discipline of chapter 126, the binary obstruction route of chapter 1622, the fan boundary of chapter 400, and the Cantor prefix reads of chapter 1475. The packet names a finite frontier and branch-window surface; it is not an infinite branch selector, host tree equality, countable choice, bar induction, compactness, or ambient stream equality.

Definition 5861.1 (PrefixTree frontier carrier)**.** *A* $\mathsf{PrefixTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (V, E, F, W, H, C, P, N)$$

with the following displayed rows:

1. *V is the prefix-closed* BHist *node row. It lists the root, every displayed node, every displayed parent-prefix row, and the finite prefix relation used by the packet.*
2. *E is the finite child* $\mathsf{ProbeBundle}$ *row. For each displayed node in V, it records the listed child probes, the edge labels, and the finite immediate-extension checks.*
3. *F is the frontier row. It lists the exposed leaves and boundary nodes whose child probes are not opened by this packet, together with their parent-prefix witnesses in V.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *branch-window row. It records exactly the finite stream-name windows read along displayed root-to-frontier paths before any fan, Cantor, or compact route may consume the tree.*
5. *H records componentwise* hsame *transport for prefix nodes, child probes, frontier rows, branch windows, replay rows, provenance rows, and local naming rows.*
6. *C records displayed* Cont *replay from a tree request through V, E, F, W and then into the downstream* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*, or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer route that asked for the prefix data.*
7. *P is the* Pkg *provenance over* $\mathsf{PrefixTree}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{FanTheorem}^{\uparrow}$*,* $\mathsf{KleeneTree}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* BHist*,* $\mathsf{ProbeBundle}$*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through V, E, F, W, H, C, P, N. It has no coordinate for selected infinite branches, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.

Lean4 (definition) `BEDC.Derived.PrefixTreeUp.PrefixTreeUp`

Theorem 5861.2 (PrefixTree frontier NameCert obligation surface)**.** *For every accepted* $\mathsf{PrefixTree}^{\uparrow}$ *carrier*

$$T = (V, E, F, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the prefix-closed node row, child probe bundle, frontier row, branch-window row, componentwise transport, continuation replay, provenance, and local naming;*
2. *prefix closure, requiring every non-root node in V to display its parent-prefix chain inside V before any child probe in E is consumed;*
3. *finite branching, requiring each child request to be read from the finite* ProbeBundle *row E attached to the currently displayed node;*
4. *frontier coverage, requiring every exposed leaf or boundary node in F to be reached by a displayed finite path through V and E;*
5. *branch-window stability, requiring every* $\mathsf{StreamName}^{\uparrow}$ *window in W to be attached to a displayed root-to-frontier path and to transport through H and replay through C without changing the prefix path;*
6. *non-escape from selected infinite branches, requiring P and N to export no selected branch, quotient stream equality, host tree equality, countable choice, open-cover compactness, or ambient fan compactness.*

Thus any downstream $\mathsf{StreamName}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, *or* $\mathsf{CantorSpace}^{\uparrow}$ *consumer receives only compact finite prefix data.*

Proof. Carrier admission is projection from Definition 5861.1. The accepted packet displays exactly the prefix-node row, finite child probes, frontier ledger, branch windows, and structural rows.

Prefix closure is read from V. A non-root node has no entry point except its displayed parent-prefix chain, so a consumer cannot jump to an unlisted node or compare nodes by host tree equality.

Finite branching is read from E. For a displayed node, the packet exposes only the listed child probes and edge labels, and each such probe is attached to the node already admitted by V.

Frontier coverage is read from F. Every exposed leaf or boundary node carries a finite path back through the node and child rows, so the frontier is a finite BEDC ledger rather than a completed branch object.

Branch-window stability is read from W, H, C. The $\mathsf{StreamName}^{\uparrow}$ window is attached to the same displayed root-to-frontier path, componentwise hsame transport preserves that path, and displayed Cont replay repeats the same finite route for fan, Kleene-tree, and Cantor-space consumers.

Finally, provenance and local naming in P, N add no excluded coordinate. Since selected infinite branches, quotient streams, host tree equality, countable choice, open-cover compactness, and ambient fan compactness are absent from the carrier, the local NameCert surface cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}$.

Constructive story: A $\mathsf{PrefixTree}^{\uparrow}$ packet is generated from prefix-closed BHist node rows, finite ProbeBundle child rows, frontier rows, finite $\mathsf{StreamName}^{\uparrow}$ branch windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5861.1 and Definition 5861.2 seed the finite prefix-tree frontier route for StreamName, fan, Kleene-tree, and Cantor-space consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrefixTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite branch selection, quotient stream equality, host tree equality, countable choice, bar induction, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix closure, finite branching, frontier coverage, branch-window stability, replay, provenance, local naming, and non-escape for BEDC.Derived.PrefixTreeUp.

$\mathsf{ChomskyNormalForm}^{\uparrow}$ reads its grammar-normalization route through chapter 356, finite word rows through chapter 36, and language-preservation comparison through chapter 351; the child files below keep the carrier and proof rows separate.

Chapter 5862

A Concrete Naming Certificate for $\mathsf{ChomskyNormalForm}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet records the finite normal-form handoff for a context-free grammar as BEDC NameCert data. It sits beside the grammar carrier of chapter 356, the finite word rows of chapter 36, and the regular-language comparison surface of chapter 351, with parse-tree rows consumed by downstream grammar chapters. The packet names the displayed nonterminal, terminal, production, start, nullable, unit-production, binarization, and language-preservation rows before any consumer may read a grammar as a Chomsky-normal-form grammar. The language-preservation row L is the shared boundary read consumed by the regular-language surface of chapter 351 and by parse-tree or CFG consumers through chapter 356.

Definition 5862.1 (ChomskyNormalForm carrier)**.** *A* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A)$$

with the following displayed rows:

1. *G is the source* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *row whose variables, terminals, productions, start symbol, derivation relation, and language surface are visible to the packet.*
2. *N is the nonterminal vocabulary row for the normal-form grammar, including the finite renamed variables introduced by binarization.*
3. *T is the terminal vocabulary row inherited from the source grammar and exposed through finite word rows.*
4. *S is the start-symbol row, including the displayed fresh-start handling used when the source grammar derives the empty word.*
5. *P is the production-shape row, requiring every displayed production to be terminal, binary nonterminal, or the admitted start-empty row.*
6. *E is the nullable and empty-word ledger, recording exactly where the construction retains or removes empty productions.*
7. *U is the unit-production elimination ledger, recording the finite route by which unit edges are replaced by ordinary productions.*

8. *B is the binarization ledger, recording the finite chain of helper variables used to replace long right-hand sides.*
9. *L is the language-preservation row comparing the source grammar and the normal-form grammar through finite derivation trees and finite word reads.*
10. *H records componentwise* hsame *transport for the source grammar, vocabulary rows, start row, production-shape row, elimination ledgers, binarization ledger, language-preservation row, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from a grammar consumer through $G, N, T, S, P, E, U, B, L$.*
12. *K is the* Pkg *provenance over* $\mathsf{ChomskyNormalForm}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *A is the local* $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted grammar quotient, an erased derivation equivalence, a host parser, countable grammar enumeration, or ambient language equality.

Theorem 5862.2 (ChomskyNormalForm NameCert obligations)**.** *For every accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *carrier*

$$F = (G, N, T, S, P, E, U, B, L, H, C, K, A),$$

the local $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed grammar rows, production-shape certification, empty-production handling, unit-production elimination, binarization, language-preservation readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside the finite rows of F.*

Proof. Carrier admission is projection from Definition 5862.1. The packet displays the source grammar, the normal-form vocabulary, the start row, the production-shape row, the nullable and unit ledgers, the binarization ledger, the language-preservation row, and the structural rows.

Production-shape certification is read from P. The row admits only terminal productions, binary nonterminal productions, and the displayed start-empty case, so the local certificate has no remaining route to arbitrary grammar productions.

Empty-production and unit-production handling are read from E and U. These ledgers record the finite eliminations before L may compare languages, preventing a consumer from replacing the construction by an erased grammar equivalence.

Binarization is read from B, and language preservation is read from L. The structural rows H, C, K, A transport, replay, source, and name the same finite route, while no carrier coordinate contains a quotient grammar, host parser, countable enumeration, or ambient language equality. □

Theorem 5862.3 (ChomskyNormalForm CFG handoff)**.** *Every* $\mathsf{ContextFreeGrammar}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{ChomskyNormalForm}^{\uparrow}$ *packet must pass through the displayed route*

$$G \longrightarrow E, U, B \longrightarrow P \longrightarrow L$$

and then through H, C, K, A. Thus the packet hands a context-free grammar to downstream normal-form consumers only after the empty-production ledger, unit-production ledger, binarization ledger, production-shape row, and language-preservation row have been displayed. It does not export a host

parser, a quotient grammar equality, an erased derivation equivalence, a countable-choice grammar selector, or ambient language extensionality.

Proof. Begin with a consumer read of the source grammar row G. The carrier does not expose the normal-form grammar as a bare replacement; it orders the read through the finite ledgers E, U, B.

The production-shape row P is downstream of those ledgers. It verifies that the resulting productions have the Chomsky-normal-form shape before a language-facing read is available.

The language-preservation row L then compares only finite derivation-tree and finite word reads. Finally H, C, K, A transport, replay, source, and name that route for CFG consumers. Since the excluded parser, quotient, erased-equivalence, selector, and ambient-extensionality objects are absent from the carrier, the handoff is exactly the displayed finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5862.1, Definition 5862.2, and Definition 5862.3 seed the finite context-free grammar route from source grammar rows through Chomsky-normal-form ledgers to language-preservation readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChomskyNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close parser implementation, quotient grammar equality, erased derivation equivalence, countable grammar selection, ambient language extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, production-shape exactness, empty-production handling, unit-production elimination, binarization, language-preservation readback, transport, replay, provenance, and local naming for BEDC.Derived.ChomskyNormalFormUp.

Constructive story: A $\mathsf{ChomskyNormalForm}^{\uparrow}$ packet is generated from a source CFG row, normal-form vocabulary rows, a start row, production-shape evidence, nullable and unit-production ledgers, a binarization ledger, language-preservation readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert data.

Chapter 5863

A Concrete Naming Certificate for $\mathsf{FiniteFanTheorem}^{\uparrow}$

Following Brouwer and Bishop-style finite fan practice, the $\mathsf{FiniteFanTheorem}^{\uparrow}$ packet records the bounded bar evidence that a Cantor or Baire prefix consumer may inspect before it exports a finite branch-window conclusion. It sits after the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the binary prefix carrier of chapter 1475, the countable prefix carrier of chapter 334, the endpoint rows of chapter 32, and the unary depth ledger of chapter 28. The chapter names a finite prefix-bar packet; it does not claim open-cover compactness, countable choice, selected branches, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5863.1 (Finite fan theorem carrier)**.** *A* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, exposing the finite observation schedule read by every branch-facing consumer.*
2. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *tree row, recording binary prefix extensions, Boolean endpoint labels, and the Cantor branch rows retained by the fan packet.*
3. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *branch row, recording the countable prefix route through which Baire-style consumers reuse the same finite observation window.*
4. *I is the* $\mathsf{Bool}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *finite prefix ledger, pairing endpoint decisions with unary prefix-depth indices and branch-length comparisons.*
5. *W is the finite bar-window row. It lists the prefixes inspected at the displayed depths and the bar decision read at each retained prefix.*
6. *H records componentwise* hsame *transport for the StreamName, CantorSpace, BaireSpace, prefix-ledger, bar-window, traversal, provenance, and local naming rows.*
7. *T records displayed* Cont *traversal from a branch request through S, C, B, I, W, H and into the local certificate.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteFanTheorem}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{CantorSpace}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{Bool}^{\uparrow}$*,* $\mathsf{Nat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row.*
The carrier has no coordinate for an infinite branch, a selected bar witness, countable choice, open-cover compactness, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteFanTheoremUp.FiniteFanTheoremTasteGate single carrier alignment`

Theorem 5863.2 (Finite fan theorem NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed StreamName prefix windows, CantorSpace tree rows, BaireSpace branch rows, Bool and Nat finite prefix ledger, finite bar-window row, componentwise* hsame *transport, displayed* Cont *traversal,* Pkg *provenance, and local naming. Every public read of the packet must inspect S before C or B, must read the finite prefix ledger I before the bar window W, and must pass through H, T, P, N before export.*

Thus a downstream fan, Cantor, Baire, Real, or Completion consumer receives only finite prefix-bar evidence. The packet exports no open-cover compactness theorem, countable-choice selector, selected branch, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Carrier admission is projection from Definition 5863.1. The accepted packet displays exactly the StreamName window, Cantor tree, Baire branch, finite prefix ledger, bar window, transport, traversal, provenance, and local naming rows.

The prefix order is forced by the carrier. A branch-facing request first reads S as the finite observation schedule, then reads the Cantor and Baire rows C, B only through the prefixes exposed by that schedule.

The bar evidence is finite because I and W are displayed ledgers. The Bool endpoint rows and Nat depth rows in I determine which retained prefixes are inspected, and W records the corresponding bar decisions without adding an infinite branch or a choice-selected witness.

Finally H, T, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for open covers, countable choice, selected branches, quotient streams, or completed real data, the listed rows exhaust the local NameCert surface. □

Theorem 5863.3 (Finite fan prefix-order exactness)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Every accepted branch-facing read exposes the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row S before it can read either the* $\mathsf{CantorSpace}^{\uparrow}$ *prefix row C or the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row B, and exposes the* $\mathsf{Nat}^{\uparrow}$ *depth ledger inside I before it can read the finite bar-window row W.*

Consequently the public prefix order of the packet is exactly

$$S \longrightarrow C, B \quad \textit{and} \quad I \longrightarrow W$$

with all exported reads then transported and named through H, T, P, N.

Proof. Project the accepted packet through Definition 5863.1. The row S is the only displayed finite observation schedule for branch-facing consumers, while C and B are retained as tree and branch prefix rows read through that schedule.

The depth order is also a carrier projection. The row I pairs Boolean endpoint decisions with Nat depth indices and branch-length comparisons, while W only lists the prefixes inspected at the displayed depths. Thus W cannot be read before the depth coordinates in I have been exposed.

Finally H, T, P, N preserve the same ordered rows under transport, traversal, provenance, and local naming. No listed carrier row supplies an alternate prefix order. □

Theorem 5863.4 (Finite fan bar-window coverage)**.** *For every accepted* $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N),$$

each accepted finite bar window in W *is covered by a displayed finite prefix family read from* S, C, B, I*. The bar-window read cannot inspect a branch outside the recorded* $\mathsf{Bool}^{\uparrow}/\mathsf{Nat}^{\uparrow}$ *finite ledger* I*, and every prefix used by the read is one of the prefixes retained by the finite observation schedule* S *and the Cantor or Baire prefix rows* C, B*.*

Proof. Use the carrier rows from Definition 5863.1. The finite observation schedule S records the inspected windows, the Cantor and Baire rows C, B retain the branch prefixes available to the packet, and I records the Boolean endpoint labels and Nat depth comparisons.

The bar row W is defined only as a finite bar-window row over those displayed depths. Hence an accepted read of W is already indexed by the prefixes and depths carried by S, C, B, I; there is no carrier coordinate for a branch outside that finite ledger.

The transport and replay rows do not add new branches. They move the same finite family through H, T, P, N, so coverage remains exactly the displayed finite prefix family. □

Theorem 5863.5 (Finite fan nonescape)**.** *Let*

$$\mathcal{B} = (S, C, B, I, W, H, T, P, N)$$

be an accepted $\mathsf{FiniteFanTheorem}^{\uparrow}$ *carrier. Any consumer reaching the local* $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$ *row* N *factors through* $S, C, B, I, W, H, T, P, N$ *and therefore exports no open-cover compactness, countable choice, selected infinite branch, or quotient stream equality.*

Proof. Carrier admission gives the complete row list $S, C, B, I, W, H, T, P, N$. A consumer that reaches N must first pass through the displayed traversal row T, provenance row P, and transport row H attached to the same finite prefix-bar packet.

The source rows are exhausted by S, C, B, I, W. These rows carry finite windows, prefix trees, branch rows, endpoint and depth ledgers, and finite bar decisions only. None of them contains an open cover, a countable-choice selector, an infinite branch choice, or a quotient equality of streams.

Thus the local naming row can export only the displayed finite packet. The nonlisted principles are outside the carrier and cannot be recovered by a NameCert consumer. □

Falsifiable prediction: Every $\mathsf{FiniteFanTheorem}^{\uparrow}$ branch-facing read factors through one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{CantorSpace}^{\uparrow}$ binary tree row, one $\mathsf{BaireSpace}^{\uparrow}$ branch row, one combined $\mathsf{Bool}^{\uparrow}$ and $\mathsf{Nat}^{\uparrow}$ finite prefix ledger, one finite bar-window row, componentwise hsame transport, displayed Cont traversal, Pkg provenance, and local naming.

Independence witness: streamname, cantorspace, bairespace, bool, nat: $\mathsf{FiniteFanTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries binary and countable prefix rows with a finite bar-window ledger as one branch-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFanTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5863.1 and Definition 5863.2 seed the finite prefix-bar route through StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, BaireSpace$^{\uparrow}$, Bool$^{\uparrow}$, and Nat$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteFanTheorem}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected branches, quotient stream equality, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-order exactness, finite bar-window coverage, componentwise transport, traversal replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteFanTheoremUp.

Constructive story: A FiniteFanTheorem$^{\uparrow}$ packet is generated from finite StreamName windows, CantorSpace tree rows, BaireSpace branch rows, Bool endpoint decisions, Nat depth indices, a finite bar-window ledger, componentwise transport, displayed traversal, package provenance, and a local NameCert row.

Chapter 5864

A Concrete Naming Certificate for $\mathsf{SpecializationOrder}^{\uparrow}$

Following Alexandroff and standard topology, the specialization preorder of a topological space is read through open-neighbourhood implication rather than host point equality. The $\mathsf{SpecializationOrder}^{\uparrow}$ packet sits after the indexed-open topology of chapter 88, the observable-open truth classifier of chapter 1860, the membership discipline of chapter 3668, and the separated order surface of chapter 53. It is also the topology-side order interface consumed by the metric-generated topology handoff of chapter 1655. The packet names only finite open-set implication data; it does not import quotient point equality, selected representatives, powerset extensionality, excluded middle over opens, or choice-based separation.

Definition 5864.1 (SpecializationOrder carrier)**.** *A* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the topology row, carrying the accepted indexed-open source, finite open-neighbourhood requests, and topology closure witnesses read from chapter 88.*
2. *O is the open-set membership classifier, recording for each displayed point and accepted open index whether the point is observed inside that open through the local open-row interface.*
3. *R is the specialization relation row. A displayed comparison $x \leq_{\mathsf{spec}} y$ is admitted only when every open-membership read true at x in O is replayed as true at y through the same topology row T.*
4. *H records componentwise* hsame *transport for the topology row, open-set membership classifier, specialization relation row, replay, provenance, and local naming rows.*
5. *C records displayed* Cont *replay from an open-neighbourhood request through T and O into the specialization relation row R.*
6. *P is the* Pkg *provenance over* $\mathsf{SpecializationOrder}^{\uparrow}$*, the topology, Sierpinski, SetLike, Poset, metric-topology,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *surfaces.*
7. *N is the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *row.*

The carrier has no coordinate for host point equality, quotient points, selected separated representatives, powerset equality of opens, excluded middle over open membership, or choice-based separation.

Lean4 ✓ `BEDC.Derived.SpecializationOrderUp.SpecializationOrderTasteGate single carrier alignment`

Theorem 5864.2 (SpecializationOrder NameCert obligations)**.** *For every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *carrier* $K = (T, O, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology row, open-set membership classifier, specialization relation row, transport, replay, provenance, and local naming;*
2. *classifier transport, requiring* H *to preserve every open-membership read in* O *along displayed* hsame *moves;*
3. *reflexivity, read by replaying each open-membership observation for a point back to the same point through* O*;*
4. *transitivity, read by composing two displayed open-neighbourhood implication rows inside* R*;*
5. *topology handoff, requiring every public order comparison to factor through the indexed-open row* T *before it reaches* R*;*
6. *sibling nonescape, forbidding quotient point equality, host separation, selected representatives, powerset extensionality, excluded middle, and choice data not displayed in* K*;*
7. *local naming through the provenance row* P *and the local certificate row* N*.*

These rows exhaust the packet obligations. A downstream T0-reflection, formal-topology, metric-topology, or separated-order consumer receives only the displayed open-neighbourhood implication route.

Proof. Carrier admission is projection from Definition 5864.1. The packet exposes only T, O, R, H, C, P, N, so each local obligation must be read from the topology row, the open-membership classifier, the specialization relation, or the structural rows.

Classifier transport is componentwise. The H row moves topology indices, point reads, open-membership observations, and specialization comparisons together, so an accepted classifier read cannot detach from the topology row that generated it.

Reflexivity and transitivity are internal to R. Reflexivity reuses the same open-membership read at the same point, while transitivity composes the two displayed implication routes through the common open index. No quotient or host equality coordinate is involved.

The topology handoff and sibling nonescape follow from the carrier boundary. Every public order read passes through T and O before R, and the rows P, N name only that route. Since quotient points, selected representatives, powerset extensionality, excluded middle, and choice-based separation are absent from the packet, none can be exported as a NameCert obligation. □

Theorem 5864.3 (SpecializationOrder topology handoff)**.** *Every accepted* $\mathsf{SpecializationOrder}^{\uparrow}$ *packet reads its preorder only from open-set implication:*

$x \leq_{\mathsf{spec}} y$ *means that every displayed open-neighbourhood read true at* x *is replayed as true at* y*.*

The handoff exports no quotient point equality, no selected representative, and no choice-based separation principle.

Proof. Start with an accepted carrier $K = (T, O, R, H, C, P, N)$. The topology row T supplies the only accepted open indices, and O supplies the only membership reads for those opens.

The specialization row R is defined as the replay of those membership reads from source point to target point. Thus a preorder comparison can be inspected by checking the displayed implication from each open-neighbourhood observation at x to the corresponding observation at y.

Finally H, C, P, N preserve, replay, source, and name that same implication route. The carrier has no coordinate for quotient points, selected representatives, host equality, or choice-based separation, so the topology handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5864.1, Definition 5864.2, and Definition 5864.3 seed the open-neighbourhood implication interface between topology rows and order-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpecializationOrder}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient point equality, T0 reflection, formal topology, metric separation, choice-based separation, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, open-membership classifier transport, reflexivity, transitivity, topology handoff, sibling nonescape, provenance, and local naming for BEDC.Derived.SpecializationOrderUp.

Constructive story: A $\mathsf{SpecializationOrder}^{\uparrow}$ packet is generated from a topology row, an open-set membership classifier, a specialization relation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5865

A Concrete Naming Certificate for $\mathsf{FiniteModulusReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteModulusReal}^{\uparrow}$ packet records a real-name route whose modulus ledger is finite and explicit at every consumer read. It sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the real-modulus criterion of chapter 3415. The packet gives Real and Completion consumers a modulus-visible carrier without quotienting Cauchy streams, selecting host limits, invoking countable choice, or asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5865.1 (FiniteModulusReal carrier)**.** *A* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, naming exactly the finite observation windows read by the carrier at the requested precision.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the inspected stream windows into regular rational approximation data.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales and comparison rows used by the displayed approximations.*
4. *M is the explicit finite modulus ledger, assigning to each retained precision request the finite window of S and dyadic tolerance data in D required before the readback row R may be consumed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S, R, D, M have been displayed.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance ledger, finite modulus ledger, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a finite Real or Completion read through S, R, D, M, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *the real-modulus criterion,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, R, D, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteModulusRealUp.FiniteModulusRealTasteGate single carrier alignment`

Theorem 5865.2 (FiniteModulusReal NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier*

$$F = (S, R, D, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$ *surface consists exactly of carrier admission for stream windows, regular rational readback, dyadic tolerances, explicit finite modulus rows, terminal Real seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must expose the finite modulus row M before it consumes R or E.*

Proof. Project the carrier of Definition 5865.1. The analytic rows are S, R, D, M, E, and the structural rows H, C, P, N transport, replay, source, and name the same packet.

The read order is finite. A precision request first selects the window named in S and the dyadic tolerance data recorded in D. The finite modulus ledger M binds those two rows to the requested read before the regular rational row R may be consumed.

The terminal seal E is downstream of S, R, D, M. Componentwise transport through H and continuation replay through C preserve the same displayed rows, while P and N record provenance and local naming. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5865.3 (FiniteModulusReal RegSeqRat handoff)**.** *For every accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier $F = (S, R, D, M, E, H, C, P, N)$, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R is consumed only after the finite stream windows S, dyadic tolerance ledger D, and explicit modulus ledger M have been displayed. Thus downstream regular rational consumers receive a finite modulus-indexed approximation route, not a quotient Cauchy stream or selected host limit.*

Proof. Project Definition 5865.1. The row R is the only regular rational readback coordinate, and it is listed after the stream-window row S, dyadic tolerance ledger D, and explicit finite modulus ledger M.

Apply Definition 5865.2. A consumer read must expose M before consuming R, so the demanded regular rational approximation is tied to a displayed finite window and a displayed dyadic tolerance row.

Transport and replay add no hidden readback source. The rows H, C, P, N preserve and name the same finite route, so a downstream $\mathsf{RegSeqRat}^{\uparrow}$ consumer cannot inspect quotient stream equality, a selected limit, countable choice, or ambient Real completeness through this handoff. □

Theorem 5865.4 (FiniteModulusReal Real seal nonescape)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *seal read of an accepted* $\mathsf{FiniteModulusReal}^{\uparrow}$ *carrier factors through*

$$S, D, M \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

No Real or Completion consumer can bypass the explicit finite modulus ledger before reading the terminal seal.

Proof. Start with an accepted packet $F = (S, R, D, M, E, H, C, P, N)$. By Definition 5865.1, the only terminal Real boundary is the row E.

Use Definition 5865.2. The seal row E is downstream of the stream windows, dyadic tolerance ledger, explicit modulus ledger, and regular rational readback. Therefore a consumer that has not displayed S, D, M, R has not reached the packet's terminal seal.

The structural rows do not create another Real-facing boundary. They transport, replay, source, and name the same finite packet, so the seal route cannot escape to quotient equality, selected host limits, countable choice, undecorated Real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: An accepted $\mathsf{FiniteModulusReal}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$ readback or the terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed finite modulus ledger M refutes the carrier surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-modulus-criterion: no listed sibling supplies the whole nine-row packet $F = (S, R, D, M, E, H, C, P, N)$, because each sibling contributes finite windows, regular readback, dyadic tolerance data, terminal Real sealing, or a modulus criterion while this chapter binds them into one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteModulusReal}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, an explicit finite modulus ledger, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5865.1, Definition 5865.2, Definition 5865.3, and Definition 5865.4 seed the finite explicit-modulus Real carrier over StreamName, RegSeqRat, DyadicRatCore, Real, and real-modulus criterion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected host limits, countable choice, undecorated Real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exactness, explicit modulus routing, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteModulusRealUp.

Chapter 5866

A Concrete Naming Certificate for $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$

Following Heine–Cantor and Bishop–Bridges constructive analysis, the $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet records the subordinate finite cover used before the compact-uniform-continuity lower-bound fold in chapter 9237. It sits over the compact source of chapter 60, the metric source of chapter 57, the continuous-map pointwise modulus rows of chapter 59, the rational positivity surface of chapter 37, and the sibling subordinate-cover packets of chapter 596 and chapter 3937. The packet names the finite ProbeBundle cover and its lower-bound handoff; it does not export open-cover compactness, a host Lebesgue number, a selected subcover, quotient point equality, or an external modulus function.

Definition 5866.1 (Compact-uniform subordinate-cover carrier)**.** *A* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, G, Q, F, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, combining the* $\mathsf{Compact}^{\uparrow}$ *admission and the* $\mathsf{Met}^{\uparrow}$ *distance carrier over the same source.*
2. *G is the continuous graph row, exposing pointwise modulus requests for the same source and target metric carriers.*
3. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row read by the pointwise-modulus requests.*
4. *F is the finite* ProbeBundle *subordinate-cover row, naming bundled centers, coverage witnesses, and the source-distance windows attached to those centers.*
5. *L is the rational lower-bound fold over the displayed subordinate radii in F.*
6. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced only after K, G, Q, F, L have been displayed.*
7. *H records componentwise* hsame *transport for K, G, Q, F, L, M and for the replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from compact-source inspection through pointwise-modulus reads, finite-cover coverage, rational lower-bound folding, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Continuous}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, G, Q, F, L, M, H, C, P, N$. It has no coordinate for open-cover compactness, host lower-bound extraction, a selected infinite subcover, quotient point equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate single carrier alignment`

Theorem 5866.2 (Compact-uniform subordinate cover handoff)**.** *For every accepted* CompactUniformSubordinateC *carrier $U = (K, G, Q, F, L, M, H, C, P, N)$, the finite* ProbeBundle *row F is the unique subordinate-cover input to the uniform-modulus handoff. A consumer reaches M only by first reading the compact metric source K, the continuous graph row G, the positive rational tolerance row Q, the bundled center and coverage row F, and the rational lower-bound fold L, with H, C, P, N preserving that route.*

Consequently the packet supplies the ProbeBundle modulus-fold route used in chapter 9237 without exposing open-cover compactness, a host Lebesgue-number choice, a selected subcover, quotient point equality, or an external modulus selector.

Proof. Project the carrier of Definition 5866.1. The compact metric source is K, the continuous graph row is G, the tolerance row is Q, the finite subordinate cover is F, the rational lower-bound fold is L, and the uniform-modulus handoff is M.

The cover row F is the only coordinate carrying bundled centers and coverage witnesses. Thus every local pointwise-modulus read used by the handoff is tied to a displayed probe name and to the same source metric row in K.

The fold row L is read after F and before M. It computes the tolerance-only radius through finite rational data rather than through a host minimum or an external Lebesgue-number selector.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for the excluded host objects, every $\mathsf{UniformModulus}^{\uparrow}$ consumer of M factors through the displayed subordinate cover. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactUniformSubordinateCover}^{\uparrow}$ packet is generated from CompactUp and MetricUp source rows, ContinuousUp pointwise-modulus rows, RatUp tolerance rows, a finite ProbeBundle subordinate-cover row, a rational lower-bound fold, a UniformModulusUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, seed)

Scope closed: Definition 5866.1 and Definition 5866.2 seed the finite subordinate-cover carrier used before compact-uniform-continuity lower-bound folding.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactUniformSubordinateCover}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CompactUniformSubordinateCoverUp.CompactUniformSubordinateCoverTasteGate\_single\_carri`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host lower-bound extraction, selected infinite subcovers, quotient point equality, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, continuous graph exposure, RatUp tolerance exposure, ProbeBundle coverage exactness, rational lower-bound folding, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformSubordinateCoverUp.

Chapter 5867

A Concrete Naming Certificate for $\mathsf{BishopIntervalContraction}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet records a finite interval-contraction handoff from located bisection data to regular Cauchy names. It consumes the located bisection route of chapter 1711, the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 neighbor of interval bisection and regular Cauchy handoff; it is not a quotient real construction, a selected infinite branch, a host fixed-point theorem, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5867.1 (Bishop interval contraction carrier)**.** *A* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (I, F, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the located source interval row, carrying endpoints, locator evidence, and the admitted interval boundary as BEDC data.*
2. *F is the contraction self-map row, recording the displayed map on the interval together with its contraction witness.*
3. *L is the Lipschitz or shrinkage ledger, exposing the dyadic contraction budget and the requested depth.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger used by the retained interval windows.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which contraction stages are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after I, F, L, D, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval, contraction, dyadic, stream, regular-rational, real-seal, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the interval-contraction request through I, F, L, D, W, R, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalContraction}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient real equality, a chosen infinite branch, a host contraction theorem, selected limits, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionTasteGate single carrier alignment`

Theorem 5867.2 (Bishop interval contraction NameCert obligations)**.** *For every accepted* BishopIntervalContractio *carrier*

$$B = (I, F, L, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed located interval, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *contraction stability, requiring every replayed self-map read to preserve the located source interval through I and to expose its shrinkage budget through L, D;*
3. *bisection compatibility, requiring the retained interval windows to factor through the located interval bisection route before any* $\mathsf{StreamName}^{\uparrow}$ *window or* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is consumed;*
4. *dyadic budget exactness, requiring every tolerance request to read the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *midpoint and error ledger D before it reaches W, R, E;*
5. *Cauchy handoff, requiring the completion-facing consumer to pass through W, R, E, H, C, P, N after the interval and contraction rows have been displayed.*

No NameCert obligation is stored outside $I, F, L, D, W, R, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 5867.1. The accepted packet displays the located interval, contraction, dyadic ledger, finite windows, regular readback, real seal, transport, replay, provenance, and local naming rows.

Contraction stability is read through I, F, L, D. The self-map row F is not a host function detached from the interval; it is paired with the located interval row I and with the shrinkage ledger L, D.

Bisection compatibility follows from the dependency route. A retained interval window must be displayed before a stream or regular-rational readback can be consumed, so the packet reaches W and R only after the located bisection and dyadic rows have been read.

Finally, the handoff rows E, H, C, P, N preserve the same finite route. Since the carrier contains no quotient equality, selected branch, host contraction theorem, selected limit, countable-choice row, or ambient completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5867.3 (Bishop interval contraction Cauchy handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must read the finite route*

$$I, F, L, D \longmapsto W, R, E \longmapsto H, C, P, N$$

before it can treat the contraction stages as a regular Cauchy real name. The consumer receives located interval data, a displayed contraction self-map, dyadic shrinkage budgets, finite stream windows, regular rational readback, a terminal real seal, componentwise transport, continuation replay, provenance, and local naming. It receives no quotient completion, selected limit, chosen infinite branch, countable choice, host fixed-point object, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionCauchyHandoff`

Proof. Start with the interval-contraction request. The source interval is read through I, while the self-map and contraction budget are read through F, L, D.

The finite windows then expose the Cauchy route. The row W records the inspected contraction stages, and R reads the corresponding regular rational approximants against the same dyadic tolerance ledger.

The terminal seal E is reached only after those rows are present. The structural rows H, C, P, N transport, replay, source, and name the same route for downstream completion consumers.

Thus the packet hands off a regular Cauchy name through displayed BEDC rows. No quotient completion, selected limit, chosen infinite branch, countable-choice row, host fixed-point object, or ambient real-completeness coordinate is present in the carrier. □

Theorem 5867.4 (Bishop interval contraction RegSeqRat endpoint stability)**.** *Let* $B = (I, F, L, D, W, R, E, H, C,$ *be an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *carrier. If the contraction route preserves the located interval row* I *and the shrinkage ledger rows* L, D*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R *is stable under componentwise* hsame *transport before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is passed to fixed-point endpoint consumers.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionRegSeqRatEndpointStability`

Proof. Project the carrier definition Definition 5867.1. The regular readback row R is downstream of the located interval, contraction, shrinkage, dyadic, and finite-window rows.

Apply the NameCert obligations of Definition 5867.2. They require contraction stability through I, F, L, D and bisection-compatible windows before the $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ rows can be consumed.

Use the Cauchy handoff theorem Definition 5867.3. It places R on the same displayed route as I, F, L, D, W and exposes the terminal $\mathsf{Real}^{\uparrow}$ seal E only after that route has been read.

The componentwise hsame row H therefore transports the readback without changing the endpoint corridor. The packet supplies no host fixed-point object, chosen limit, quotient completion, countable-choice row, or ambient Real-completeness datum, so fixed-point endpoint consumers receive only the displayed RegSeqRat-to-Real handoff. □

Theorem 5867.5 (Bishop interval contraction endpoint readback)**.** *Every fixed-point endpoint consumer reading the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *from an accepted* $\mathsf{BishopIntervalContraction}^{\uparrow}$ *packet must first read the located interval row* I*, the contraction row* F*, the shrinkage ledger* L, D*, the* $\mathsf{StreamName}^{\uparrow}$ *windows* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the structural rows* H, C, P, N*. Only after that displayed L10 route is present may the consumer hand* E *to* $\mathsf{BishopFixedPointInterval}^{\uparrow}$ *or* $\mathsf{BanachFixedPointStability}^{\uparrow}$*. The readback exports no host fixed point, chosen limit, quotient completion, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalContractionUp.BishopIntervalContractionEndpointReadback`

Proof. Project the accepted carrier of Definition 5867.1. The endpoint row E is terminal in the displayed inventory, and the only rows preceding it are the located interval, contraction, shrinkage, dyadic, window, and regular-rational readback rows.

Apply the Cauchy handoff of Definition 5867.3. It forces every completion-facing consumer to pass through $I, F, L, D, W, R, E, H, C, P, N$ before treating the contraction stages as a regular Cauchy real name.

The fixed-point endpoint consumer is a special consumer of that route. It may pass the terminal seal onward to $\mathsf{BishopFixedPointInterval}^{\uparrow}$ or $\mathsf{BanachFixedPointStability}^{\uparrow}$, but the handoff carries only the displayed rows and none of the excluded host fixed-point or completeness data. □

Theorem 5867.6 (Bishop interval contraction shrinkage composition)**.** *Let $B = (I, F, L, D, W, R, E, H, C, P, N)$ be an accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ carrier. If two accepted contraction-stage reads share the located interval row I and expose their shrinkage ledgers through the displayed rows L, D, then their composed finite stage route reaches W, R, E, H, C, P, N only through that same dyadic shrinkage budget. The composition exports no host fixed-point composition, selected limit, quotient completion, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Project the carrier of Definition 5867.1. The only rows available for contraction-stage reads are the located interval row I, the contraction row F, and the shrinkage ledger rows L, D before any stream or completion-facing data is visible.

Use the NameCert obligations of Definition 5867.2. They force contraction stability through I, F, L, D, so two stage reads sharing I can be composed inside the certificate only by replaying the displayed dyadic shrinkage ledger.

Apply the Cauchy handoff of Definition 5867.3. The composed stage route reaches the finite windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E after the shared interval and shrinkage rows have been read.

Finally use endpoint readback from Definition 5867.5. The structural rows H, C, P, N preserve the same route for endpoint consumers, and no carrier coordinate supplies a host composition theorem, a selected fixed point, quotient completion, countable choice, or ambient Real completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopIntervalContraction}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows $I, F, L, D, W, R, E, H, C, P, N$ refutes the carrier surface.

Independence witness: located-interval-bisection, dyadicratcore, streamname, regseqrat, real, bishop-fixed-point-interval: no listed sibling supplies the joint packet containing a located interval source, contraction self-map, dyadic shrinkage ledger, finite windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalContraction}^{\uparrow}$ packet is generated from a located interval row, a contraction self-map row, a dyadic shrinkage ledger, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5867.1, Definition 5867.2, and Definition 5867.3 seed the finite interval-contraction route from located bisection and dyadic shrinkage data to regular Cauchy and Real-seal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopIntervalContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient real equality, selected limits, chosen infinite branches, countable choice, a host fixed-point theorem, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, contraction stability, bisection compatibility, dyadic budget exactness, and Cauchy handoff for BEDC.Derived.BishopIntervalContractionUp.

Chapter 5868

A Concrete Naming Certificate for $\mathsf{SeminormedSpace}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on seminormed and locally convex structures, the $\mathsf{SeminormedSpace}^{\uparrow}$ packet records the finite family-seminorm interface between the single norm route of chapter 1950, the displayed variation-seminorm ledger of chapter 5730, the locally convex family surface of chapter 3386, and completion-facing functional-analysis packets. It consumes the $\mathsf{VecSpace}^{\uparrow}$ source of chapter 47 and the $\mathsf{Real}^{\uparrow}$ scalar seal of chapter 38; it does not collapse finite seminorm families to one $\mathsf{Norm}^{\uparrow}$ row or to a host topology.

Definition 5868.1 (SeminormedSpace carrier)**.** *A* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N)$$

with the following displayed rows:

1. *V is the* $\mathsf{VecSpace}^{\uparrow}$ *source row, exposing the vector carrier, classifier, zero vector, addition, negation, and scalar action only through displayed BEDC data.*
2. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar and budget row used by seminorm values, scalar absolute values, uniform entourages, and metric handoff tolerances.*
3. *I is the finite family-index row. It lists the displayed seminorm labels admitted by the packet and exposes no hidden index set.*
4. *F is the seminorm-family ledger. For each displayed index in I, it records the nonnegativity, zero-row compatibility, triangle, and scalar-budget rows of the corresponding seminorm read.*
5. *Z is the common zero-row compatibility ledger, comparing the vector zero classifier of V with every zero-value row displayed in F.*
6. *T is the triangle-budget ledger, routing every read of a displayed sum through vector addition in V, the selected seminorm row in F, and the real addition and order rows in R.*
7. *A is the scalar-budget ledger, routing every scalar multiplication read through scalar action in V, the absolute-value row in R, and the selected seminorm row in F.*
8. *U is the induced uniform-structure handoff row. It reads finite intersections of seminorm balls through I, F, R before any uniform consumer can inspect them.*
9. *M is the induced metric or pseudometric handoff row for downstream completion packets, available only after V, R, I, F, U have been displayed.*

10. *H records componentwise* hsame *transport for vector, Real, family-index, seminorm-ledger, zero, triangle, scalar, uniform, metric, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from every seminorm-family, uniform, or metric request through $V, R, I, F, Z, T, A, U, M, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{NormedSpace}^{\uparrow}$, $\mathsf{VariationSeminorm}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, *completion-facing functional-analysis rows,* BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host locally convex topology, a single hidden norm, a completed quotient, an unbounded seminorm index set, a selected Cauchy limit, or classical duality.

Lean4 ✓ `BEDC.Derived.SeminormedSpaceUp.TasteGate.SeminormedSpaceTasteGate single carrier alignment`

Theorem 5868.2 (SeminormedSpace NameCert obligations)**.** *For every accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *carrier*

$$S = (V, R, I, F, Z, T, A, U, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed vector-space, Real, finite family-index, seminorm-family, zero, triangle, scalar, induced uniform, induced metric, transport, replay, provenance, and local naming rows;*
2. *family-index exposure, requiring every seminorm read to name an index from the finite row I before any value row in F can be consumed;*
3. *zero-row compatibility, requiring every zero seminorm read to compare with the displayed vector-zero classifier in V through Z and not through host vector equality;*
4. *triangle-budget routing, requiring reads of $p_i(x + y)$ to pass through vector addition in V, the i-row of F, and the addition and order rows of R;*
5. *scalar-budget routing, requiring reads of $p_i(ax)$ to pass through scalar action in V, the absolute-value and multiplication rows of R, and the selected seminorm row of F;*
6. *induced uniform and metric handoff, requiring U and M to be read only after V, R, I, F, Z, T, A are displayed;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N.*

No $\mathsf{SeminormedSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 5868.1. The accepted packet lists the vector-space, Real, finite index, seminorm-family, zero, triangle, scalar, uniform, metric, transport, replay, provenance, and local naming rows.

Family-index exposure comes next. A seminorm read first chooses a displayed index in I, and the corresponding row in F is the only place where nonnegativity, zero compatibility, triangle, and scalar-budget evidence may be read.

The zero, triangle, and scalar obligations are ledger routes. The row Z compares zero seminorm reads with the vector-zero classifier in V; the row T sends displayed sums through V, F, R; and the row A sends scalar multiples through V, R, F.

Finally U and M are handoff rows derived from the same family ledger. Transport, replay, provenance, and local naming preserve this route through H, C, P, N. Since the carrier has no host topology, hidden norm, completed quotient, unbounded index set, selected limit, or classical-duality coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5868.3 (SeminormedSpace functional-analysis handoff)**.** *Every locally convex, variation-seminorm, normed-space comparison, or completion-facing functional-analysis consumer of an accepted* $\mathsf{SeminormedSpace}^{\uparrow}$ *packet must read the displayed route*

$$V, R, I, F, Z, T, A \longmapsto U, M \longmapsto H, C, P, N.$$

Thus the consumer receives a finite family of seminorm rows over $\mathsf{VecSpace}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, together with induced uniform and metric handoff data. It does not receive a single host norm, a host locally convex topology, an unbounded seminorm family, quotient completion, selected limits, or classical duality.*

Proof. Start with a downstream consumer request. A vector operation can be answered only from V, and a scalar or seminorm budget can be answered only after the Real row R and a displayed family index from I have been read.

Use Definition 5868.2. The seminorm-family ledger F is guarded by the zero, triangle, and scalar rows Z, T, A, so a consumer cannot inspect an induced uniform or metric handoff before the finite family surface has been exposed.

The rows U and M are therefore handoff rows rather than independent analytic objects. They package finite intersections of seminorm balls and metric or pseudometric reads for LocallyConvexSpace, VariationSeminorm, NormedSpace comparison, and completion-facing consumers.

The structural rows H, C, P, N transport, replay, source, and name the same packet. With no host norm, host locally convex topology, unbounded family, quotient completion, selected limit, or classical-duality row in the carrier, the displayed route is the entire handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SeminormedSpace}^{\uparrow}$ consumer that reaches a locally convex, variation-seminorm, normed-space comparison, or completion-facing read without first exposing $V, R, I, F, Z, T, A, U, M, H, C, P, N$ refutes the packet.

Independence witness: normedspace, variation-seminorm, locally-convex-space, vecspace, real: no listed sibling is bijective to $\mathsf{SeminormedSpace}^{\uparrow}$ because this packet jointly binds finite family-index exposure, zero-row compatibility, triangle and scalar budget ledgers, induced uniform and metric handoff, transport, replay, provenance, and local naming over the displayed vector and Real rows.

Closure status for $\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SeminormedSpace}^{\uparrow}$ packet is generated from a displayed VecSpace row, a Real scalar row, a finite family-index row, a seminorm-family ledger, zero, triangle, and scalar budget ledgers, induced uniform and metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5868.1, Definition 5868.2, and Definition 5868.3 seed the finite family-seminorm interface consumed by locally convex, variation-seminorm, normed-space comparison, and completion-facing functional-analysis packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeminormedSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host locally convex topology, hidden single-norm reduction, unbounded seminorm families, quotient completions, selected limits, classical duality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, family-index exposure, zero-row compatibility, triangle and scalar budget routing, induced uniform and metric handoff, transport, replay, provenance, and local naming for BEDC.Derived.SeminormedSpaceUp.

Chapter 5869

A Concrete Naming Certificate for ReassignmentMassSubcube$^{\uparrow}$

The ReassignmentMassSubcube$^{\uparrow}$ packet realizes the maximum-mass subcube geometry branch of the DNA six-bit boundary tile analysis as finite NameCert data. It reads the reassignment weight row, the subcube mass ledger, and the nested maximum-mass subcube theorem of Definition 9462.1232 and Definition 9462.1233 through displayed codon-window and reassignment sibling packets. The packet consumes the CodonSixBitWindow$^{\uparrow}$ row of chapter 2234, the SixBoundaryTileCoverage$^{\uparrow}$ row of chapter 3470, the ReassignmentSupportCompression$^{\uparrow}$ row of chapter 3558, and the AlternativeCodeReassignmen row of chapter 3589 before any maximum-mass face can be named; it is not a biological mechanism claim, a host optimization oracle, or a hidden table-family search.

Definition 5869.1 (ReassignmentMassSubcube carrier)**.** *A* ReassignmentMassSubcube$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, B, S, A, R, Q, L, N, H, C, P, K)$$

with the following displayed rows:

1. *W is the* CodonSixBitWindow$^{\uparrow}$ *source-window row exposing the six coordinate bits for codon subcube reads.*
2. *B is the* SixBoundaryTileCoverage$^{\uparrow}$ *boundary-coverage row used to keep boundary-trigger coverage visible while subcubes are inspected.*
3. *S is the* ReassignmentSupportCompression$^{\uparrow}$ *support row, exposing the finite reassigned support and motif-cover surface.*
4. *A is the* AlternativeCodeReassignment$^{\uparrow}$ *comparison-table row from which reassignment weights are read.*
5. *R is the reassignment-weight row, recording each displayed codon weight $w(c)$ from the curated comparison surface.*
6. *Q is the finite subcube ledger, listing the inspected axis-aligned Q_5, Q_4, Q_3, Q_2 faces and their fixed-coordinate conditions.*
7. *L is the subcube-mass ledger, recording each displayed mass $M(S)$ as a finite sum over the visible weight row R.*
8. *N is the nested maximum-mass row, recording the unique maximum-mass chain $Q_5 \supset Q_4 \supset Q_3 \supset Q_2$ displayed by the source theorem.*

9. *H records componentwise* hsame *transport for source-window, boundary-coverage, support-compression, alternative-code, weight, subcube, mass, nested-chain, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from reassignment comparison to weights, from weights to finite subcube masses, and from masses to the nested maximum-mass row.*

11. *P is the* Pkg *provenance over* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$, $\mathsf{CodonSixBitWindow}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{ReassignmentSupportCompression}^{\uparrow}$, $\mathsf{AlternativeCodeReassignment}^{\uparrow}$, *the source reassignment-mass geometry vision,* BHist, hsame, Cont, *and* NameCert.

12. *K is the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these twelve rows. It has no coordinate for uncurated code tables, stochastic evolutionary dynamics, biochemical causality, a host argmax operator, quotient codon equality, or hidden biological semantics.

Theorem 5869.2 (ReassignmentMassSubcube NameCert obligations)**.** *For every accepted* ReassignmentMassSubcu *carrier $M = (W, B, S, A, R, Q, L, N, H, C, P, K)$, the local* $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed codon window, boundary coverage, support compression, alternative-code comparison, reassignment-weight, finite subcube, mass, nested maximum-mass, transport, replay, provenance, and local naming rows. The mass row must compute only finite sums over the displayed weight row, and the nested row must expose the unique maximum-mass chain only after W, B, S, A, R, Q, L are visible.*

Proof. Project the carrier of Definition 5869.1. The first four rows provide the source coordinate, boundary coverage, reassignment support, and alternative-table comparison surfaces.

The weight row R is then a finite comparison read from A, and the subcube ledger Q lists the inspected axis-aligned faces over the six-bit source window W. The mass ledger L can therefore compute only finite sums of displayed weights.

The nested maximum-mass row N is downstream of those finite sums. It records the displayed chain from the vision theorem rather than invoking a host optimization oracle.

Finally H, C, P, K transport, replay, source, and name the same packet. Since no uncurated table family, stochastic dynamics, biochemical mechanism, host argmax, quotient codon equality, or hidden biological semantics occurs in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 5869.3 (ReassignmentMassSubcube sibling route)**.** *Every accepted* $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ *read factors through*

$$\mathsf{CodonSixBitWindow}^{\uparrow}, \mathsf{SixBoundaryTileCoverage}^{\uparrow}, \mathsf{ReassignmentSupportCompression}^{\uparrow}, \mathsf{AlternativeCodeReassignment}^{\uparrow}$$

Thus a maximum-mass subcube claim is read as finite reassignment geometry over displayed codon coordinates, boundary-tile coverage, support compression, and alternative-code comparison rows. It cannot be consumed as a free-standing biological mechanism, host optimization theorem, unbounded genetic-code survey, or quotient of codon labels.

Proof. Begin with the sibling sources. The codon six-bit window supplies the coordinate carrier, the six-boundary tile coverage row keeps trigger support visible, the reassignment-support-compression row exposes the finite support surface, and the alternative-code-reassignment row supplies the comparison table data.

Only after those sources are displayed can the packet read reassignment weights and subcube masses. The rows R, Q, L bind the finite weights, inspected faces, and mass sums, while N records the nested maximum-mass chain.

The structural rows then preserve that route. Transport, replay, provenance, and local naming cannot add the excluded biological or host-optimization coordinates, so every accepted read remains inside the displayed sibling lattice. □

Falsifiable prediction: An accepted $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ read is refuted if the finite maximum-mass subcube route can bypass reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, or six-boundary-tile-coverage rows.

Independence witness: reassignment-support-compression, alternative-code-reassignment, codon-six-bit-window, six-boundary-tile-coverage: no listed sibling is bijective with $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ because the siblings respectively expose support compression, comparison-table reassignment, coordinate windows, or trigger coverage, while this packet jointly binds reassignment weights, subcube ledgers, finite mass sums, nested maximum-mass exposure, transport, replay, provenance, and local naming.

The Lean TasteGate.lean development for $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ must include explicit Nontrivial $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ and FieldFaithful $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ instances preserving $W, B, S, A, R, Q, L, N, H, C, P, K$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}$.

Constructive story: A $\mathsf{ReassignmentMassSubcube}^{\uparrow}$ packet is generated from codon six-bit windows, six-boundary tile coverage rows, reassignment-support compression rows, alternative-code comparison rows, reassignment weights, finite subcube ledgers, mass sums, nested maximum-mass rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5869.1, Definition 5869.2, and Definition 5869.3 seed the finite maximum-mass subcube NameCert surface for reassignment geometry.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ReassignmentMassSubcube}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, uncurated genetic-code surveys, stochastic evolutionary dynamics, host optimization, quotient codon equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sibling-source exposure, reassignment-weight exactness, finite subcube ledger exposure, mass-sum exactness, nested maximum-mass routing, Nontrivial ReassignmentMassSubcubeUp, and FieldFaithful ReassignmentMassSubcubeUp for BEDC.Derived.ReassignmentMassSubcubeUp.

Chapter 5870

A Concrete Naming Certificate for $\mathsf{CauchyInterlacing}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyInterlacing}^{\uparrow}$ packet records the finite interlacing route by which two regular rational names are merged into one Real-facing seal. It sits over the finite StreamName schedules of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence handoff of chapter 1738, and the terminal Real seal of chapter 38. The packet names the even/odd merge as displayed certificate data; it is not a quotient stream, selected limit, countable-choice schedule, or ambient completeness principle.

Definition 5870.1 (Cauchy interlacing carrier)**.** *A* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (A, B, W, D, T, R, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular tails are to be merged.*
2. *W is the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule row. Even windows read A, odd windows read B, and the row records the finite requests needed by the merged consumer.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the two source tails and the interlaced tail.*
4. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff row witnessing that the even and odd tails feed the same regular Cauchy boundary.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, B, W, D, T have been displayed.*
6. *H records componentwise* hsame *transport for the two source rows, schedule, tolerance ledger, tail-equivalence handoff, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$A, B, W, D, T \longmapsto R.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchyInterlacing}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *row.*

The carrier predicate admits exactly these rows. Its classifier compares accepted packets only through $A, B, W, D, T, R, H, C, P, N$ and has no coordinate for quotient stream equality, selected representatives, countable choice, host limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyInterlacingUp.CauchyInterlacingTasteGate single carrier alignment`

Theorem 5870.2 (Cauchy interlacing NameCert obligations)**.** *For every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier*

$$K = (A, B, W, D, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *sources, even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, dyadic tolerance ledger, tail-equivalence handoff, Real seal, transport, replay, provenance, and local naming row;*
2. *schedule discipline, requiring every merged read to identify whether it is reading the even source row A or the odd source row B through W;*
3. *dyadic tolerance exactness, requiring D to record the source-tail and merged-tail tolerances consumed by the interlacing route;*
4. *tail handoff stability, requiring T to mediate the passage from the two regular tails to one common Cauchy boundary before R is visible;*
5. *real-seal nonescape, requiring R to remain downstream of A, B, W, D, T;*
6. *componentwise transport, displayed replay, package provenance, and local naming stability through H, C, P, N;*
7. *ledger refusal, excluding quotient streams, selected limits, countable choice, host sequence limits, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No Cauchy interlacing obligation is stored outside these rows.

Proof. Project the accepted carrier $K = (A, B, W, D, T, R, H, C, P, N)$. The visible coordinates are exactly the two regular rational source rows, one finite even/odd schedule, one dyadic tolerance ledger, one tail-equivalence handoff, one Real seal row, and the structural rows.

The schedule row W fixes the finite interlacing order. Even requests read A and odd requests read B, so no merged consumer can inspect an unrecorded source position.

The dyadic ledger D and tail-equivalence row T then bind the two regular tails to one Cauchy boundary. The Real seal R is downstream of that route, while H, C, P, N transport, replay, source, and name the same displayed rows.

Since the excluded host objects have no carrier coordinate, replay edge, provenance entry, or local naming row, the listed rows exhaust the local NameCert surface. □

Theorem 5870.3 (Cauchy interlacing RegSeqRat handoff)**.** *Every accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *carrier exposes the merged regular rational read only through the finite route*

$$A, B, W, D, T \longmapsto R \longmapsto H, C, P, N.$$

Thus a consumer receives the two $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, continuation replay, provenance, and local naming before it can use the merged real-facing name.*

Proof. Start with the carrier projection. The two source rows A and B are independent displayed $\mathsf{RegSeqRat}^{\uparrow}$ inputs, and the only row that relates them to one merged stream is the even/odd schedule W.

Read the dyadic ledger and tail handoff. The row D supplies the tolerances for both source tails and for the interlaced tail, while T records the CauchyTailEquivalence passage from those two tails to one common boundary.

Only after A, B, W, D, T have been displayed can R be read as the Real-facing seal. The structural rows H, C, P, N preserve the same dependency order, so the merged regular rational handoff cannot bypass the displayed schedule or tolerance ledger. □

Theorem 5870.4 (Cauchy interlacing Real seal nonescape)**.** *An accepted* $\mathsf{CauchyInterlacing}^{\uparrow}$ *consumer that reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal must have already displayed the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, the even/odd* $\mathsf{StreamName}^{\uparrow}$ *interlacing schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *handoff. Consequently the seal cannot be used as a shortcut to quotient stream equality, selected limits, countable choice, host limit existence, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the seal row R from an accepted carrier. By the carrier definition, R is downstream of A, B, W, D, T and records only the Real-facing boundary reached from those displayed rows.

The replay row C has no branch beginning at R. It replays the route from the two regular rational sources through the even/odd schedule, dyadic tolerance ledger, and tail-equivalence handoff before the seal can be consumed.

The transport and provenance rows H and P preserve that route componentwise, and the local name N names only the displayed interlacing packet. Hence no excluded host object can escape through the Real seal. □

Falsifiable prediction: Every $\mathsf{CauchyInterlacing}^{\uparrow}$ Real-facing read factors through two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, local naming, and one terminal RealUp seal.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{CauchyInterlacing}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two regular rational source rows, an even/odd interlacing schedule, a dyadic tolerance ledger, one tail-equivalence handoff, structural transport, replay, provenance, local naming, and a single Real-facing seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyInterlacing}^{\uparrow}$ packet is generated from two RegSeqRat source rows, one even/odd StreamName schedule, one DyadicRatCore tolerance ledger, one CauchyTailEquivalence handoff, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, seed)

Scope closed: Definition 5870.1, Definition 5870.2, Definition 5870.3, and Definition 5870.4 seed the finite constructive route for merging two regular rational names into one Real-facing seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyInterlacing}^{\uparrow}}$, formalTarget)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient stream equality, selected limits, countable choice, host sequence limits, trichotomy, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, even/odd schedule discipline, dyadic tolerance exactness, CauchyTailEquivalence handoff, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyInterlacingUp.

Chapter 5871

A Concrete Naming Certificate for $\mathsf{RealLocallyCompact}^{\uparrow}$

Following Bishop–Bridges and Kelley local compactness, the $\mathsf{RealLocallyCompact}^{\uparrow}$ packet records the finite Real-facing route by which a displayed real endpoint receives a compact neighbourhood certificate. It refines the generic local compactness row of chapter 1726 over the real uniformity surface of chapter 3445, with endpoint data from chapter 38, finite windows from chapter 126, regular rational readback from chapter 301, dyadic radius accounting from chapter 291, closed ball rows from chapter 1451, and compact metric witnesses from chapter 129. The packet names the local closed-ball compactness surface for $\mathsf{Real}^{\uparrow}$; it does not supply a host-selected neighbourhood base, open-cover compactness theorem, quotient real equality, countable choice, or ambient completeness row.

Definition 5871.1 (Real locally compact carrier)**.** *A* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (R, U, W, D, B, K, A, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint row whose local neighbourhood is inspected.*
2. *U is the* $\mathsf{RealUniformSpace}^{\uparrow}$ *entourage row attached to the same endpoint data.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read endpoint names, entourage requests, and radius comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}/\mathsf{RegSeqRat}^{\uparrow}$ *radius-accounting row, carrying dyadic radii and regular rational readback for the displayed window.*
5. *B is the* $\mathsf{MetricClosedBall}^{\uparrow}$ *row centred at the displayed real endpoint and radius.*
6. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *witness for the displayed closed-ball window.*
7. *A is the* $\mathsf{LocallyCompact}^{\uparrow}$ *handoff row, read only after R, U, W, D, B, K have been exposed.*
8. *H records componentwise* hsame *transport for the endpoint, uniformity, window, radius, closed-ball, compactness, handoff, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a Real local-neighbourhood request through $R, U, W, D, B, K, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{RealLocallyCompact}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealUniformSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricClosedBall}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these displayed rows and exports no host neighbourhood selector, open-cover compactness, quotient real equality, countable choice, selected representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealLocallyCompactUp.RealLocallyCompactUp`

Theorem 5871.2 (Real locally compact NameCert obligations)**.** *For every accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *carrier* $L = (R, U, W, D, B, K, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$ *surface consists exactly of endpoint admission through R, real-uniform entourage exposure through U, finite-window routing through W, dyadic and regular-rational radius accounting through D, closed-ball exposure through B, compact metric witnessing through K, local compactness handoff through A, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5871.1. The endpoint and uniformity rows R, U provide the Real-facing source before any radius or compactness data are read.

The radius route is finite. The window row W selects the inspected names and tolerance requests, while D records the dyadic radius ledger and regular rational readback needed to form the closed-ball row B.

Compactness enters only through K for the displayed closed-ball window. The handoff row A then presents the same data to the generic $\mathsf{LocallyCompact}^{\uparrow}$ surface, not to an ambient neighbourhood base or open-cover compactness theorem.

Transport, replay, provenance, and local naming preserve the same rows through H, C, P, N. Since the carrier has no coordinate for quotient real equality, countable choice, selected representatives, or ambient completeness, the listed rows exhaust the local certificate. □

Theorem 5871.3 (Real locally compact handoff)**.** *Every Real or Completion consumer that reads local compactness from an accepted* $\mathsf{RealLocallyCompact}^{\uparrow}$ *packet factors through*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{RealUniformSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{MetricClosedBall}^{\uparrow} \longmapsto \mathsf{CompactM}$$

The handoff exports only the displayed endpoint, uniformity, finite-window, radius-accounting, closed-ball, compact metric, transport, replay, provenance, and local naming rows of Definition 5871.1.

Proof. Start with a consumer request for a compact neighbourhood of a displayed real endpoint. The request first reads R and U, so the endpoint and real-uniformity data are fixed before any compactness row can be inspected.

Next read the finite radius route. The window row W and accounting row D expose the dyadic and regular-rational budget used to construct the closed-ball row B.

The compact witness K is attached to that same closed ball. The handoff row A is therefore a local compactness read over a displayed Real closed-ball window, not a host-selected neighbourhood or open-cover principle.

The structural rows H, C, P, N preserve the route for downstream Real and Completion consumers. No excluded equality, choice, selected representative, or ambient completeness row is available to the handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{RealLocallyCompact}^{\uparrow}$ packet is generated from RealUp endpoint rows, RealUniformSpaceUp entourage rows, StreamName finite windows, DyadicRatCore and RegSeqRat radius accounting, a MetricClosedBall row, a CompactMetric witness, a LocallyCompact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5871.1, Definition 5871.2, and Definition 5871.3 seed the Real-facing local compactness certificate over closed-ball and compact metric sibling rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealLocallyCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host neighbourhood bases, open-cover compactness, quotient real equality, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite radius accounting, closed-ball compactness witnessing, local compactness handoff, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RealLocallyCompactUp.

Chapter 5872

A Concrete Naming Certificate for CompactUniformModulusStability$^{\uparrow}$

Following the Bishop–Bridges compact finite-net proof of uniform continuity, the CompactUniformModulusStability$^{\uparrow}$ packet records the stability surface that keeps compact nets, pointwise moduli, lower-bound folds, and the final uniform-modulus handoff synchronized. It sits over the compact selector of chapter 914, the compact uniform modulus row of chapter 2299, the finite lower-bound fold of chapter 619, the consumer surface of chapter 369, the metric triangle estimate of chapter 1159, and the compact metric source of chapter 129. The packet supplies a finite stability object for compact-uniform consumers; it does not select a host modulus, a hidden center, a quotient equality, or an ambient compactness theorem.

Definition 5872.1 (Compact uniform modulus stability carrier)**.** *A* CompactUniformModulusStability$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, N, M, F, U, T, H, C, P, L)$$

with the following displayed rows:

1. K *is the* CompactMetric$^{\uparrow}$ *finite-net source row.*
2. N *is the* CompactNetModulusSelector$^{\uparrow}$ *selector row carrying bundled centers, pointwise radii, and selected precision data.*
3. M *is the* CompactUniformModulus$^{\uparrow}$ *compact-uniform handoff row.*
4. F *is the* FiniteLowerBoundFold$^{\uparrow}$ *row that synchronizes the finite radius ledger with the selected precision.*
5. U *is the downstream* UniformModulus$^{\uparrow}$ *consumer row.*
6. T *is the* MetricTriangleUniformEstimate$^{\uparrow}$ *row that transports the selected center estimates into the final source-target bound.*
7. H *records componentwise* hsame *transport for compact, selector, modulus, fold, triangle, consumer, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a compact-uniform tolerance request through* $K, N, M, F, U, T, H, P, L$.
9. P *is the* Pkg *provenance over* CompactUniformModulusStability$^{\uparrow}$, CompactMetric$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, MetricTriangleUniformEstimate$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

10. L is the local $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$ *row.*
The carrier compares accepted stability packets only through these rows. It contains no host compactness principle, selected center outside the finite net, external modulus function, quotient point equality, or choice schedule.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusStabilityUp.CompactUniformModulusStabilityTasteGate`
`single carrier alignment`

Theorem 5872.2 (Compact uniform modulus stability NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *carrier* $S = (K, N, M, F, U, T, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulus}}$ *surface consists exactly of compact finite-net admission through* K*, compact selector exposure through* N*, compact-uniform modulus handoff through* M*, lower-bound fold synchronization through* F*, uniform-modulus consumer export through* U*, metric-triangle estimate transport through* T*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 5872.1. The compact source row K and selector row N provide the finite centers, pointwise modulus rows, subordinate radii, and selected precision before any consumer reads a modulus.

The lower-bound fold row F synchronizes those finite radius rows with the selected precision. The compact-uniform row M and uniform-modulus row U therefore read the same finite ledger rather than an external modulus function.

The triangle row T is the final estimate route. It uses the selected center and the two source-distance bounds already exposed by the selector and fold rows, so it cannot introduce a hidden center or host compactness argument.

Transport, replay, provenance, and local naming preserve the same packet through H, C, P, L. Since no excluded object appears among the coordinates, the listed rows exhaust the local certificate. □

Theorem 5872.3 (Compact uniform modulus stability handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusStability}^{\uparrow}$ *packet reads the stable route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CompactNetModulusSelector}^{\uparrow} \longmapsto \mathsf{FiniteLowerBoundFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow}$$

Thus the consumer receives one synchronized finite packet of compact-net rows, pointwise-modulus rows, lower-bound fold data, selected precision, metric-triangle estimates, transport, replay, provenance, and local naming.

Proof. Start from a tolerance request. The compact metric row K and selector row N expose the finite net, selected center candidates, pointwise radii, and compact selector data.

Apply the fold row F. It binds the selected precision to the same finite radius ledger, so the request cannot move to a different center family or a host minimum.

Use the triangle estimate row T to transport the source bounds and pointwise estimates into the target bound consumed by M and then by U. The compact-uniform and uniform-modulus rows therefore receive the same synchronized finite route.

The structural rows H, C, P, L keep this route componentwise stable. No selected center outside the packet, quotient equality, choice schedule, host compactness, or external modulus function is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusStability}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, CompactNetModulusSelector rows, CompactUniformModulus rows, FiniteLowerBoundFold rows, UniformModulus consumer rows, MetricTriangleUniformEstimate rows, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5872.1, Definition 5872.2, and Definition 5872.3 seed the stability certificate tying compact-net selectors, folds, triangle estimates, compact-uniform rows, and uniform-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the finite packet, quotient point equality, choice schedules, external modulus functions, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, selector stability, fold synchronization, metric-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulu

Chapter 5873

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5873.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5873.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5873.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5873.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The sandwich hub routes its cofinal common-tail read through chapter 7532 before the child files expose the lower and upper RegSeqRat readbacks.

Chapter 5874

A Concrete Naming Certificate for $\mathsf{RegularCauchySandwich}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet names the two-sided bracketing surface used when regular Cauchy data is read by a real-completion consumer. It sits between the dyadic tolerance ledger of chapter 291, the finite window route of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the tail agreement boundary of chapter 1738. The packet records lower and upper regular tails as finite BEDC rows; it is not a quotient stream, a selected limit, a trichotomy principle, or an ambient completeness theorem.

Definition 5874.1 (Regular Cauchy sandwich carrier)**.** *A* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
2. *U is the upper regular-Cauchy row, read through* $\mathsf{RegSeqRat}^{\uparrow}$ *data.*
3. *T is the common tail-agreement row inherited from* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling both bracket sides at the same precision request.*
5. *W_L and W_U are the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for the lower and upper sides.*
6. *R_L and R_U are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows exposed after the windows are displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after both sides, the common tail row, the dyadic ledger, the windows, and the readbacks have been consumed.*
8. *H records componentwise* hsame *transport for the lower side, upper side, tail row, ledger, windows, readbacks, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each finite precision request through*

$$L, U, T, D, W_L, W_U, R_L, R_U \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySandwich}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyTailEquivalence}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *row.*
The classifier accepts only packets in which the lower and upper regular rows, common tail agreement, dyadic tolerance ledger, finite windows, readbacks, and terminal seal belong to the same displayed bracketing route. It has no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5874.2 (Regular Cauchy sandwich NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$ *surface consists exactly of lower regular-row exposure, upper regular-row exposure, common tail-agreement exposure, shared dyadic tolerance accounting, lower and upper finite-window exposure, lower and upper RegSeqRat readback, terminal Real seal routing, componentwise transport, continuation replay, package provenance, and local naming through N.*

Proof. Project the carrier tuple in Definition 5874.1. The lower and upper sides are displayed as L and U, and the only row that binds them is the common tail-agreement coordinate T.

The shared ledger D fixes the precision request for both sides. The finite windows W_L, W_U are opened against that same ledger, and the readback rows R_L, R_U expose exactly the inspected regular rational data.

The seal row E is downstream of those rows, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded analytic coordinate occurs in the carrier, the local NameCert surface is exactly the displayed bracketing route. □

Theorem 5874.3 (Regular Cauchy sandwich window bracketing)**.** *Let*

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

be an accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. Every finite precision request consumed by the packet reads the two-sided route*

$$L, U, T, D \longmapsto W_L, W_U \longmapsto R_L, R_U \longmapsto E, H, P, N.$$

Thus lower and upper regular windows are bracketed by one displayed tolerance ledger and one common tail-agreement row, not by quotient representatives, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient completeness.

Proof. Start with a finite precision request. The packet exposes only one dyadic tolerance ledger D, so both sides of the bracket are read against the same displayed tolerance data.

The lower and upper windows W_L, W_U are then opened after the common tail row T has been displayed. Their readbacks R_L, R_U remain attached to the same packet, so neither side can be replaced by an external quotient stream or a selected representative.

The terminal seal E is reached only after both readbacks. Transport, replay, provenance, and local naming through H, C, P, N preserve that order for every accepted request, which gives the claimed two-sided bracketing route. □

Theorem 5874.4 (Regular Cauchy sandwich Real seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *seal read accepted by a* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier can bypass the displayed route through the lower and upper regular rows, common tail agreement, shared dyadic ledger, lower and upper StreamName windows,*

lower and upper RegSeqRat readbacks, and local structural rows. Equivalently, every accepted terminal seal read is attached to

$$S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H, C, P, N)$$

and cannot be separated from those rows by quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 5874.2. It exposes the Real seal row E only as part of the local NameCert surface containing the two regular rows, common tail row, ledger, windows, readbacks, and structural rows.

Now use Definition 5874.3. Every precision request reaches the readbacks through the same two-sided route, so a seal read detached from one side of the bracket is not a read of this packet.

Finally inspect Definition 5874.1. The carrier contains no coordinate for quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, or ambient Real completeness. A purported seal read using such data therefore lies outside the accepted $\mathsf{RegularCauchySandwich}^{\uparrow}$ carrier. □

Theorem 5874.5 (Regular Cauchy sandwich Cantor intersection seal)**.** *Let* $S = (L, U, T, D, W_L, W_U, R_L, R_U, E, H$ *be an accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *carrier. If the lower and upper rows are supplied by a nested-shrink or Cantor-intersection route, then the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed by* S *is available only through the displayed sandwich route, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. In particular, the consumer-facing seal factors through Definition 5874.3, Definition 5874.4, chapter 4227, and chapter 7754, without asserting a selected intersection point.*

Proof. Apply Definition 5874.3. It forces every finite precision request for S through the common tail row, shared dyadic ledger, two StreamName windows, and two RegSeqRat readbacks before the seal row is visible.

The nested-shrink and Cantor-intersection inputs supply the lower and upper rows only as finite interval or shrink data. Those routes provide the source rows for the sandwich, but they do not add a coordinate for a selected point, quotient stream equality, or ambient completeness.

Now apply Definition 5874.4. Any terminal $\mathsf{Real}^{\uparrow}$ read detached from the displayed carrier is outside the accepted packet, so the Cantor-facing consumer receives only the finite sandwich seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySandwich}^{\uparrow}$ packet is generated from lower and upper regular-Cauchy rows, a common tail-agreement row, a shared dyadic tolerance ledger, lower and upper StreamName windows, lower and upper RegSeqRat readbacks, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, seed)

Scope closed: Definition 5874.1, Definition 5874.2, Definition 5874.3, and Definition 5874.4 seed the finite two-sided regular-Cauchy bracketing route over DyadicRatCore, StreamName, RegSeqRat, Real, and CauchyTailEquivalence rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySandwich}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, trichotomy, apartness decision, countable choice, host real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, common tail exactness, shared dyadic tolerance accounting, two-sided StreamName window routing, RegSeqRat readback, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySandwichUp.

Theorem 5874.6 (Regular Cauchy sandwich sibling dependency route)**.** *Every accepted* $\mathsf{RegularCauchySandwich}^{\uparrow}$ *sandwich packet depends on the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback rows of chapter 301 and the* $\mathsf{StreamName}^{\uparrow}$ *pointwise window rows of chapter 126 before it exposes a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The route consumes those siblings only through the displayed lower and upper windows, lower and upper regular readbacks, common tail row, shared dyadic ledger, transport, replay, provenance, and local naming rows of Definition 5874.1.*

Proof. Read the carrier rows from Definition 5874.1. The lower and upper windows W_L, W_U are the pointwise $\mathsf{StreamName}^{\uparrow}$ windows, and the rows R_L, R_U are the regular rational readbacks supplied by $\mathsf{RegSeqRat}^{\uparrow}$.

The bracketing theorem Definition 5874.3 orders every precision request through those windows and readbacks before the seal row is consumed. The Real-seal nonescape theorem Definition 5874.4 prevents bypass through quotient stream equality, selected limits, host real equality, or ambient completeness.

Thus the chapter's sibling dependency is the concrete route through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$, not an external analytic carrier. □

Theorem 5874.7 (Regular Cauchy sandwich cofinal tail-mesh dependency)**.** *Every accepted* $\mathsf{RegularCauchySandwic}$ *packet whose common tail row is presented cofinally must read the* $\mathsf{CofinalTailMesh}^{\uparrow}$ *route of chapter 7532 before the lower and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are consumed. The dependency is the displayed route through the sandwich carrier, the window bracketing theorem, and the sibling dependency route; it is not an appeal to quotient streams, selected limits, trichotomy, choice, host real equality, or ambient completeness.*

Proof. Read the carrier rows in Definition 5874.1. The common tail coordinate T is displayed before the lower and upper StreamName windows, the lower and upper RegSeqRat readbacks, and the Real seal row.

When T is presented cofinally, consume the CofinalTailMesh route of chapter 7532. That route supplies the shared finite tail-window mesh before any Real or completion-facing seal is read.

Apply the window bracketing theorem Definition 5874.3. It keeps the lower and upper readbacks downstream of the common tail row and the shared dyadic ledger.

Finally use the sibling dependency route Definition 5874.6. It already requires the StreamName and RegSeqRat siblings before the Real seal, so adding the cofinal tail-mesh read fixes the horizontal dependency without adding any external analytic coordinate. □

Chapter 5875

A Concrete Naming Certificate for $\mathsf{DynamicProgramming}^{\uparrow}$

Following Bellman's finite dynamic programming principle, the $\mathsf{DynamicProgramming}^{\uparrow}$ packet records finite backward recursion data as BEDC NameCert rows. It sits after the finite path and edge ledger of chapter 236, the transition-row discipline of chapter 189, the probability-row carrier of chapter 184, the terminal real-name seal of chapter 38, and the finite horizon counter of chapter 28. The chapter names only finite states, finite actions, displayed transition and cost rows, horizon-indexed Bellman tables, policy readback, transport, replay, provenance, and local naming; it does not assert an infinite-horizon fixed point, host minimizer, choice-selected policy, quotient value equality, or ambient completeness theorem.

Definition 5875.1 (Dynamic programming carrier)**.** *A* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, T, C, H, V, P, R, N)$$

with the following displayed rows:

1. *S is the finite displayed state row, sharing its path endpoints with chapter 236.*
2. *A is the finite action row, listing exactly the actions admitted at the displayed states.*
3. *T is the transition row, read through the finite Markov and probability surfaces of chapter 189 and chapter 184.*
4. *C is the cost row, exposing the finite real-valued immediate costs and terminal costs through the* $\mathsf{Real}^{\uparrow}$ *seal.*
5. *H is the horizon row, a* $\mathsf{Nat}^{\uparrow}$*-indexed finite counter for the backward recursion stages.*
6. *V is the Bellman value table, giving one displayed value row for each state and horizon stage.*
7. *P is the policy readback row, recording only actions whose value entry is witnessed by the same finite transition and cost rows.*
8. *R records componentwise* hsame *transport, displayed* Cont *replay, and* Pkg *provenance over S, A, T, C, H, V, P and the cited sibling packets.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *row.*

The classifier reads accepted packets only through $S, A, T, C, H, V, P, R, N$. It has no coordinate for a host minimizer, selected infinite policy, analytic contraction theorem, quotient value function, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DynamicProgrammingUp.DynamicProgrammingTasteGate single carrier alignment`

Theorem 5875.2 (Dynamic programming Bellman obligations)**.** *For every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *carrier* $D = (S, A, T, C, H, V, P, R, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$ *surface consists exactly of carrier admission for finite states, finite actions, transition rows, cost rows, horizon rows, Bellman value-table rows, policy readback rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and ledger refusal for host minimizers, selected infinite policies, quotient value equality, countable choice, and ambient completeness.*

Proof. Carrier admission is projection from Definition 5875.1. The finite state and action rows are displayed before transition and cost rows can be read, and the horizon row fixes which backward stages may appear.

The Bellman table row V is then read stage by stage over the same displayed horizon. A policy readback row P is exposed only after the corresponding state, action, transition, cost, and value rows have all been displayed.

Transport, replay, provenance, and local naming are exhausted by R and N. Since the carrier has no further coordinate beyond $S, A, T, C, H, V, P, R, N$, no accepted read can import a host minimizer, selected infinite policy, quotient value equality, countable choice, or ambient completeness. □

Theorem 5875.3 (Dynamic programming finite-horizon induction)**.** *Every accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *Bellman table is generated by finite backward induction over the displayed* $\mathsf{Nat}^{\uparrow}$ *horizon row. The terminal stage reads the terminal cost row from* C *and the* $\mathsf{Real}^{\uparrow}$ *seal; each predecessor stage reads only the previously displayed successor-stage value row, the transition row* T*, the action row* A*, the immediate cost row* C*, and the same local NameCert route.*

Proof. Project the horizon row H and read it through the unary source discipline of chapter 28. The empty endpoint is the terminal stage, so the value entry is the displayed terminal cost row sealed through $\mathsf{Real}^{\uparrow}$.

For a one-step predecessor, use the closed generation of the horizon row. The successor-stage table has already been displayed, and the new value row is computed from the finite transition row, finite action row, immediate cost row, and successor value row.

Repeating the one-step construction along the finite horizon supplies every value-table row. No step asks for an infinite horizon, a global fixed point, or an external minimizer; the induction is only the BEDC finite-history recursion named by the carrier. □

Theorem 5875.4 (Dynamic programming policy readback)**.** *Every policy row exported by an accepted* $\mathsf{DynamicProgramming}^{\uparrow}$ *packet is a readback from the displayed Bellman table. For each displayed state and horizon stage, the row* P *may expose an action only after the state row* S*, action row* A*, transition row* T*, cost row* C*, horizon row* H*, value row* V*, componentwise transport, replay, provenance, and local naming have all been read.*

Proof. Start with a policy-facing read. The carrier forces the read through the finite state and action rows, then through the transition and cost rows that define the local Bellman entry.

Next read the horizon row and the value-table row. The policy row is not an independent selector: it is attached to the displayed value entry at the same state and horizon stage.

Finally apply the structural row R and local naming row N. These rows preserve the finite route under transport and replay, so any action exposed by P remains tied to the same Bellman table entry and cannot become a host choice function or infinite policy. □

Theorem 5875.5 (Dynamic programming NameCert package)**.** *The* $\mathsf{DynamicProgramming}^{\uparrow}$ *NameCert package is the finite Bellman-recursion packet whose public surface is exactly the route*

$$\mathsf{Nat}^{\uparrow}, \mathsf{ShortestPath}^{\uparrow}, \mathsf{MarkovChain}^{\uparrow}, \mathsf{ProbSpace}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DynamicProgramming}^{\uparrow}.$$

It exports the state, action, transition, cost, horizon, value-table, policy-readback, transport, replay, provenance, and local naming rows of Definition 5875.1. It exports no host optimization theorem, no selected global policy, no infinite-horizon convergence theorem, no quotient value-function equality, and no bridge to a standard dynamic-programming model.

Proof. The cited sibling chapters supply the input rows: shortest-path data for finite route endpoints, Markov and probability data for transition rows, real data for cost and value seals, and natural-number data for the finite horizon.

Apply Definition 5875.2 to identify the local obligation surface. Apply Definition 5875.3 to generate the value table from the terminal stage backward through the displayed horizon. Apply Definition 5875.4 to expose policy rows only as readbacks from that table.

The package is therefore a finite NameCert surface over the displayed rows. Because no carrier coordinate stores host optimization, global choice, infinite convergence, quotient equality, or a standard bridge, those objects are not exported. □

Falsifiable prediction: A claimed $\mathsf{DynamicProgramming}^{\uparrow}$ read that exposes a policy action without first displaying S, A, T, C, H, V, R, N refutes the finite Bellman carrier.

Independence witness: shortestpath, markovchain, probspace, real, nat: no listed sibling is bijective to $\mathsf{DynamicProgramming}^{\uparrow}$ because this packet binds finite state-action data, transition and cost rows, horizon-indexed Bellman value tables, policy readback, transport, replay, provenance, and local naming as one finite recursion surface.

Closure status for $\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DynamicProgramming}^{\uparrow}$ packet is generated from finite state rows, finite action rows, transition rows, cost rows, a Nat-indexed horizon, Bellman value-table rows, policy readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5875.1, Definition 5875.2, Definition 5875.3, Definition 5875.4, and Definition 5875.5 seed the finite Bellman-recursion NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DynamicProgramming}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host minimizers, selected infinite policies, quotient value equality, infinite-horizon convergence, ambient RealUp completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Bellman table construction, finite-horizon induction, policy readback exactness, transport, replay, provenance, and local naming for BEDC.Derived.DynamicProgrammingUp.

Chapter 5876

Regular Cauchy Even-Odd Substreams NameCert Construction

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet records the standard constructive passage from one regular Cauchy name to its even and odd substreams before a Real or Completion consumer reads either side. It sits over the regular rational readback of chapter 301, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, the two-source interleaving packet of chapter 414, and the finite interleaving criterion of chapter 3405. The packet supplies a displayed subsequence route for the L10 Real and Completion corridor; it does not choose a new limit, quotient streams, or import ambient completeness.

Definition 5876.1 (Regular Cauchy even-odd substreams carrier)**.** *A* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, E, O, D, R, A, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular Cauchy row. It names one* $\mathsf{RegSeqRat}^{\uparrow}$ *source together with the finite precision requests inspected by the parity split.*
2. *E is the even selector-window row. It displays the finite windows read from the source at indices $2n$ and records no schedule beyond the even selector.*
3. *O is the odd selector-window row. It displays the finite windows read from the source at indices $2n + 1$ and records no schedule beyond the odd selector.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget row. It stores the shared dyadic radius, split tolerance, comparison windows, and common-scale data used by both selector windows.*
5. *R is the paired* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the even and odd windows under the budget D.*
6. *A is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the even readback, available only after S, E, D, R have been displayed.*
7. *B is the* $\mathsf{Real}^{\uparrow}$ *seal boundary for the odd readback, available only after S, O, D, R have been displayed.*

8. *H records componentwise* hsame *transport for the source, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an even or odd substream request through $S, E, O, D, R, A, B, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegularCauchyInterleaving}^{\uparrow}$, $\mathsf{CauchyInterleavingCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, E, O, D, R, A, B, H, C, P, N$. The carrier has no coordinate for quotient stream equality, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.RegularCauchyEvenOddSubstreamsTasteGate`
`single carrier alignment`

Theorem 5876.2 (Regular Cauchy even-odd substreams NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier*

$$X = (S, E, O, D, R, A, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source regular Cauchy row, even selector windows, odd selector windows, dyadic budget, paired readbacks, two Real seal boundaries, transport, replay, provenance, and local naming rows;*
2. *source visibility, requiring both parity branches to read the same displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *source row S;*
3. *even-window exactness, requiring every even branch request to pass through the selector windows E and the budget D before any readback is visible;*
4. *odd-window exactness, requiring every odd branch request to pass through the selector windows O and the same budget D before any readback is visible;*
5. *paired regular readback admission through R, requiring the two* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks to use the selector windows and the shared dyadic ledger already displayed;*
6. *two Real seal boundaries through A and B, requiring the even and odd Real-facing reads to remain downstream from their parity windows, D, R, H, and C;*
7. *componentwise transport, displayed replay, package provenance, and local naming for exactly $S, E, O, D, R, A, B, H, C, P, N$;*
8. *refusal of quotient stream equality, selected host limits, countable choice, trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as local certificate data.*

No local obligation is stored outside these rows.

Proof. Project the carrier of Definition 5876.1. Its displayed inventory is exactly $S, E, O, D, R, A, B, H, C, P, N$, so the source row, parity selector windows, dyadic budget, paired readbacks, two seals, and structural rows exhaust the packet.

The source visibility clause is read from S. Both parity branches share that row, so neither branch can replace the source by a second hidden regular name.

Even-window and odd-window exactness are read from E and O under the shared dyadic row D. The even branch exposes only indices of the form $2n$, while the odd branch exposes only indices of the form $2n+1$, and both branches use the same tolerance ledger before the readback row is consumed.

The paired readback row R is downstream from those windows. It repacks the two selector routes as $\mathsf{RegSeqRat}^{\uparrow}$ readbacks without adding a quotient stream or selected host limit.

Finally the Real seals A and B are reached only after the corresponding parity route has been transported by H and replayed by C. Since P and N name only the displayed packet, the listed rows are the complete local NameCert surface. □

Theorem 5876.3 (Regular Cauchy even-odd Real-seal nonescape)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier factors through one of the two finite parity routes*

$$S, E, D \longmapsto R \longmapsto A \longmapsto H, C, P, N,$$
$$S, O, D \longmapsto R \longmapsto B \longmapsto H, C, P, N.$$

The even route reads the even selector windows and the odd route reads the odd selector windows, but both routes share the same source row, dyadic budget, paired regular readback, transport, replay, provenance, and local naming. Hence no consumer reaches a $\mathsf{Real}^{\uparrow}$ *seal through quotient stream equality, a chosen host limit, countable choice, trichotomy, or ambient completeness.*

Proof. Apply the obligation surface of Definition 5876.2. It confines every accepted packet to the displayed source, two parity selector rows, one dyadic budget, paired readback, two seal rows, and structural data.

For an even request, the consumer reads S, E, D before R and then reaches A. The route has no coordinate other than the even selector windows and the shared budget, so the even seal cannot bypass the displayed packet.

For an odd request, the consumer reads S, O, D before R and then reaches B. This is the same source and budget route with the odd selector row in place of the even selector row.

The structural rows H, C, P, N transport, replay, source, and name the two routes componentwise. Because the carrier contains no quotient stream equality, selected host limit, countable-choice row, trichotomy branch, or ambient completeness coordinate, every Real or Completion consumer remains inside one of the two displayed parity routes. □

Theorem 5876.4 (Regular Cauchy parity cofinality)**.** *Let* $X = (S, E, O, D, R, A, B, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ *carrier over one source* $\mathsf{RegSeqRat}^{\uparrow}$ *row* S*. The even selector windows* E *and the odd selector windows* O *determine cofinal tail-shift reads of that same source row. Their Real-facing seals factor through the same ordered four-face packet*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The assertion is only this cofinal finite-window factorization. It does not identify quotient streams, select a limit, use countable choice, use trichotomy, or invoke ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source row S and the two selector-window rows E, O from Definition 5876.1. The rows are displayed as windows over one source $\mathsf{RegSeqRat}^{\uparrow}$ carrier, so no second regular name is introduced by the parity split.

Apply Definition 301.257 to transport late reads along the even and odd tail shifts. The tail-shift result keeps each parity branch inside the same regular-rational source route.

Next use Definition 126.670. It supplies one finite request section for the displayed selector windows, so both parity reads are compared at the finite StreamName surface rather than at a completed stream quotient.

Apply Definition 1647.3 to fuse the two parity budgets through the shared dyadic tolerance ledger. The shared D row is the DyadicRatCore face of the ordered packet.

Finally use Definition 38.713. The terminal Real seal is reached only after the dyadic budget, finite StreamName window, and RegSeqRat readback have been displayed. Thus the two parity seals are cofinal views of the same finite four-face route, with no selected limit, countable-choice selector, trichotomy branch, quotient stream equality, or ambient completeness principle. □

Falsifiable prediction: Every $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ Real or Completion read is refuted if it reaches either $\mathsf{Real}^{\uparrow}$ seal without displaying the source RegSeqRat row, the selected even or odd StreamName windows, the shared DyadicRatCore budget, the paired RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: regseqrat, streamname, dyadicratcore, real, regular-cauchy-interleaving, cauchy-interleaving-criterion: $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one source regular Cauchy name, two parity selector-window routes, one shared dyadic budget, paired regular readbacks, two Real seal boundaries, transport, replay, provenance, and local naming as one displayed packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}$ packet is generated from one RegSeqRat source, even selector windows, odd selector windows, a shared DyadicRatCore budget, paired RegSeqRat readbacks, two RealUp seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5876.1, Definition 5876.2, Definition 5876.3, and Definition 5876.4 seed the parity-substream route used by L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyEvenOddSubstreams}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient stream equality, selected host limits, countable choice, trichotomy, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, parity-window exactness, shared dyadic budget exactness, paired regular readback admission, two Real-seal nonescape clauses, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyEvenOddSubstreamsUp.

Chapter 5877

A Concrete Naming Certificate for $\mathsf{SchwartzSpace}^{\uparrow}$

Following Schwartz and Treves on test-function spaces, the $\mathsf{SchwartzSpace}^{\uparrow}$ packet records the finite locally convex carrier surface behind rapidly decreasing smooth test functions. It sits beside the complete locally convex interface of chapter 3495, the finite family-seminorm route of chapter 5868, the pointwise test-function carrier of chapter 3399, and the derivative observation route of chapter 2210. The packet gives Fourier, tempered-distribution, weak-derivative, and PDE-facing consumers a displayed test-function-space surface without importing a host topological vector space, quotient function equality, unrestricted seminorm enumeration, selected representatives, or classical completeness.

Definition 5877.1 (SchwartzSpace carrier)**.** *A* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, V, I, G, D, R, T, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FrechetSpace}^{\uparrow}$ *source row, exposing complete metrizability and locally convex handoff only through the displayed Frechet carrier.*
2. *V is the* $\mathsf{SeminormedSpace}^{\uparrow}$ *finite-family row, naming the seminorm labels used by the Schwartz test-function space read.*
3. *I is the* $\mathsf{SchwartzFunction}^{\uparrow}$ *test-function row, recording the accepted rapidly decreasing smooth windows inspected by the packet.*
4. *G is the polynomial-growth and rapid-decay ledger. It binds each displayed seminorm index to a derivative order, a polynomial weight, and the real-valued budget read used to test decay.*
5. *D is the* $\mathsf{WeakDerivative}^{\uparrow}$ *stability row, recording derivative-window shifts and sign conventions that remain inside the same Schwartz test-function-space surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row used by seminorm values, decay budgets, and completion-facing estimates.*
7. *T is the completion and distribution nonescape row. It permits Frechet and tempered-distribution consumers to read the displayed test-function-space boundary only after F, V, I, G, D, R have been exposed.*

8. *H records componentwise* hsame *transport for Frechet, seminorm-family, test-function, rapid-decay, derivative, Real, nonescape, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from every test-function-space request through $F, V, I, G, D, R, T, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{SchwartzSpace}^{\uparrow}$, $\mathsf{FrechetSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{SchwartzFunction}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host Schwartz topology, a quotient of functions, an unrestricted seminorm family, a selected completion point, a distribution dual, nuclearity, Fourier inversion, or classical choice.

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.SchwartzSpaceTasteGate single carrier alignment`

Theorem 5877.2 (SchwartzSpace NameCert obligations)**.** *For every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *carrier*

$$S = (F, V, I, G, D, R, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Frechet-space, seminorm-family, Schwartz-function, rapid-decay, weak-derivative, Real, nonescape, transport, replay, provenance, and local naming rows;*
2. *seminorm growth routing, requiring every seminorm read to choose a displayed finite-family row in V and a polynomial-growth budget in G before a Real value can be consumed from R;*
3. *rapid-decay test-function closure, requiring every accepted test window from I to satisfy the displayed derivative-order and polynomial-weight ledger in G;*
4. *derivative-row stability, requiring weak-derivative shifts to pass through D and return to the same I, G, R surface rather than to host smooth-function equality;*
5. *completion and distribution nonescape, requiring T to mediate every Frechet, completion-facing, Fourier-facing, tempered-distribution, or PDE-facing read after F, V, I, G, D, R have been displayed.*

No $\mathsf{SchwartzSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.SchwartzSpaceUp.TasteGate.SchwartzSpaceTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5877.1. The analytic rows are exactly F, V, I, G, D, R, T, and the structural rows are exactly H, C, P, N.

Seminorm growth routing is forced by the order of the carrier. A consumer first reads the finite family row V, then the derivative and polynomial-weight budget in G, and only then may read a Real scalar estimate through R.

Rapid-decay closure and derivative stability are the two test-function-space checks. The row I supplies accepted Schwartz test windows, while G records their rapid-decay budgets and D records the weak-derivative shifts that stay inside the same displayed surface.

Finally T is a boundary row rather than an additional topological object. It lets Frechet, completion-facing, Fourier-facing, tempered-distribution, and PDE consumers replay the same rows without receiving a host Schwartz topology, quotient function equality, selected completion point, distribution dual, nuclearity, inversion theorem, or choice principle. Thus the listed rows exhaust the local NameCert obligations. □

Theorem 5877.3 (SchwartzSpace functional-analysis handoff)**.** *Every accepted* $\mathsf{SchwartzSpace}^{\uparrow}$ *packet exported to a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing consumer factors through*

$$\mathsf{FrechetSpace}^{\uparrow},\ \mathsf{SeminormedSpace}^{\uparrow},\ \mathsf{SchwartzFunction}^{\uparrow},\ \mathsf{WeakDerivative}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto F, V, I, G, D, R, T, H, C, P, N.$$

The exported object is a finite test-function-space NameCert surface with displayed seminorm growth budgets and derivative stability. It is not a host locally convex function space, an unrestricted seminorm enumeration, a selected completion, a dual-space quotient, or a classical distribution theorem.

Proof. Open the accepted carrier using Definition 5877.1. Any functional-analysis request must first choose one of the displayed source rows F, V, I, D, R and then pass through the growth ledger G.

Use Definition 5877.2. The seminorm and rapid-decay obligations keep polynomial weights and derivative orders attached to finite displayed rows, while the derivative-row obligation prevents a consumer from replacing weak-derivative stability by host smooth-function equality.

The nonescape row T is then the only export boundary. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route for every downstream consumer. Since no excluded host object is a carrier coordinate, the displayed route is the whole handoff surface. □

Falsifiable prediction: An accepted $\mathsf{SchwartzSpace}^{\uparrow}$ consumer that reaches a Frechet, Fourier, tempered-distribution, weak-derivative, or PDE-facing read without first exposing $F, V, I, G, D, R, T, H, C, P, N$ refutes the carrier surface of Definition 5877.1.

Independence witness: frechet-space, seminormed-space, schwartz-function, weak-derivative, real: no listed sibling is bijective to $\mathsf{SchwartzSpace}^{\uparrow}$ because this packet jointly binds complete locally convex source rows, finite seminorm-family exposure, rapid-decay test-function windows, derivative stability, Real budget readback, completion and distribution nonescape, transport, replay, provenance, and local naming in one test-function-space carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchwartzSpace}^{\uparrow}$ packet is generated from a FrechetSpace row, a finite SeminormedSpace family row, SchwartzFunction test windows, rapid-decay and polynomial-growth ledgers, WeakDerivative stability rows, Real scalar readback, completion and distribution nonescape, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5877.1, Definition 5877.2, and Definition 5877.3 seed the finite locally convex test-function-space surface consumed by Frechet, Fourier, tempered-distribution, weak-derivative, and PDE-facing packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchwartzSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host Schwartz topology, quotient function equality, unrestricted seminorm enumeration, selected completion, distribution duality, nuclearity, Fourier inversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, seminorm growth routing, rapid-decay test-function closure, derivative-row stability, completion and distribution nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SchwartzSpaceUp.

Chapter 5878

A Concrete Naming Certificate for $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$

The $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet realizes the private/shared decoration model of chapter 9390 as finite Rule 110 classifier-decoration NameCert data. It consumes the cellular automaton carrier of chapter 934, the row-classifier surface of chapter 941, the finite pattern catalog of chapter 995, the Rule 110 classifier of chapter 5844, and the finite partition carrier of chapter 201. The packet names quotient support families, private blocks, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, transport, replay, provenance, and local naming; it does not import a host cellular semantics, an infinite orbit classifier, quotient support equality, a biological model, or a hidden enumeration oracle.

Definition 5878.1 (Rule 110 private/shared decoration carrier)**.** *A* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, B, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient support family row, naming the finite family of support blocks inspected by the decoration.*
2. *B is the private-block row, recording for each accepted support a displayed nonempty private coordinate block.*
3. *S is the shared-coordinate incidence row. For each shared coordinate it records the displayed set of incident private blocks and requires cardinality at least two.*
4. *R is the* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *row, attaching the decorated support family to finite Rule 110 row-window and pattern-tag reads.*
5. *E is the decoration-count provenance row. It records the partition and row-set enumeration data used by the private/shared decoration theorem without exposing a host generating-function oracle.*
6. *H records componentwise* hsame *transport for quotient support, private-block, shared-incidence, classifier, enumeration, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a decoration request through Q, B, S, R, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularRowClassifier}^{\uparrow}$, $\mathsf{CellularPatternCatalog}^{\uparrow}$, $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$, $\mathsf{Partition}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for absent private blocks, singleton shared incidence, host quotient equality, unbounded Rule 110 evolution, biological interpretation, or an external enumeration procedure.

Theorem 5878.2 (Rule 110 private/shared decoration NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ *carrier*

$$D = (Q, B, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of quotient support family admission through Q, private-block nonemptiness through B, shared-coordinate incidence through S, Rule 110 pattern-classifier routing through R, decoration-count provenance through E, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5878.1. The decoration data are exactly the quotient support row Q, the private-block row B, the shared-incidence row S, the Rule 110 classifier row R, and the enumeration provenance row E.

Private-block nonemptiness is read before a support can be accepted. Shared-coordinate incidence is then read as a displayed row set of size at least two, so a shared coordinate cannot be collapsed into a private coordinate or erased from the decoration.

The Rule 110 classifier row R attaches the decorated support family to finite row-window and pattern-tag reads from the cellular siblings. The count provenance row E records the finite partition and row-set enumeration route and does not supply an external counting oracle.

Finally H, C, P, N transport, replay, source, and name the same carrier. Since no excluded host object is present as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5878.3 (Private/shared decoration classifier handoff)**.** *Every accepted* RuleOneTenPrivateSharedDecorati
consumer that reads a decorated Rule 110 support family factors through

$$\mathsf{CellularAutomaton}^{\uparrow},\ \mathsf{CellularRowClassifier}^{\uparrow},\ \mathsf{CellularPatternCatalog}^{\uparrow},\ \mathsf{RuleOneTenPatternClassifier}^{\uparrow},\ \mathsf{Partition}^{\uparrow} \longmapsto Q$$

The exported object is a finite classifier-decoration packet with private-block nonemptiness, shared incidence, and decoration-count provenance. It is not an infinite cellular-orbit classifier, a host quotient-support equality, a biological stochastic model, or an unrestricted enumeration theorem.

Proof. Open the accepted carrier using Definition 5878.1. A support-family read can inspect only Q, and each support must expose its nonempty private block through B before shared coordinates are consumed.

The shared-coordinate row S supplies the incidence sets whose size is at least two. The classifier row R then routes the accepted decoration into the existing cellular automaton, row-classifier, pattern-catalog, and Rule 110 pattern-classifier surfaces.

The provenance row E records the finite decoration-count route, while H, C, P, N preserve the same data under transport, replay, package provenance, and local naming. Thus every accepted consumer receives exactly the displayed finite packet and none of the excluded host semantics. □

Falsifiable prediction: Any accepted $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ consumer that omits private-block nonemptiness or shared-coordinate incidence rows while reading a decorated Rule 110 support family refutes the carrier surface of Definition 5878.1.

Independence witness: cellularautomaton, cellularrowclassifier, cellularpatterncatalog, rule110patternclassifier: no listed sibling is bijective to $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ because this packet jointly binds quotient support families, nonempty private blocks, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, transport, replay, provenance, and local naming.

The Lean TasteGate file for this packet must provide $\mathsf{Nontrivial}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ instances for the displayed rows $Q, B, S, R, E, H, C, P, N$ before the decoration carrier can serve as a verified classifier-decoration consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}$ packet is generated from quotient support family rows, nonempty private-block rows, shared-coordinate incidence rows, Rule 110 pattern-classifier rows, decoration-count provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5878.1, Definition 5878.2, and Definition 5878.3 seed the finite private/shared classifier-decoration surface for Rule 110 support-family consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenPrivateSharedDecoration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite Rule 110 orbit classification, host quotient-support equality, biological interpretation, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked quotient support admission, private-block nonemptiness, shared-coordinate incidence, Rule 110 classifier routing, decoration-count provenance, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenPrivateSharedDecorationUp.

Chapter 5879

A Concrete Naming Certificate for $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$

Following Cantor and Bishop–Bridges, the $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ packet records the finite diagonal obstruction to enumerating Cantor-space bit names. It sits over the binary prefix carrier of chapter 1475, the finite bar-facing tree discipline of chapter 1622, the fan-theoretic bounded-window route of chapter 400, and the Real-facing non-surjection boundary of chapter 3793. The packet names only finite rows: an attempted enumeration window, its displayed diagonal bit schedule, the disagreement row at each inspected index, and the no-enumeration handoff. It is not a host-cardinality theorem, not a completed powerset construction, and not a quotient-stream principle.

Definition 5879.1 (Constructive Cantor finite diagonal carrier)**.** *A* $\mathsf{ConstructiveCantorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (E, W, D, G, T, F, R, H, C, P, N)$$

with the following visible rows:

1. *E is the displayed finite enumeration attempt row, listing the first n candidate Cantor-space names through finite prefix windows;*
2. *W is the finite binary observation window, naming the inspected coordinates and the finite prefix depth used for each candidate;*
3. *D is the diagonal-bit row, obtained by flipping the displayed ith bit of the ith candidate whenever that coordinate is present in W;*
4. *G is the disagreement ledger, recording for every inspected index $i < n$ that the diagonal row disagrees with the ith candidate at the displayed coordinate;*
5. *T is the tree-read row through* $\mathsf{KleeneTree}^{\uparrow}$*, ensuring that finite prefix extension and absence of a completed path are read as finite branch data;*
6. *F is the fan-window row through* $\mathsf{FanTheorem}^{\uparrow}$*, recording the bounded prefix surface consumed by any compact Cantor-space handoff;*
7. *R is the Real-facing route through* $\mathsf{RealDiagonalNonSurjection}^{\uparrow}$*, used only after the Cantor-space disagreement ledger has been displayed;*

8. *H records componentwise* hsame *transport for enumeration rows, prefix windows, diagonal bits, disagreement ledgers, tree rows, fan rows, Real-facing routes, replay rows, provenance, and local naming;*
9. *C records displayed* Cont *replay from an enumeration window to diagonal-bit construction, disagreement verification, Cantor-space handoff, and Real-facing non-surjection routing;*
10. *P is the* Pkg *provenance over* ConstructiveCantorTheorem$^{\uparrow}$, CantorSpace$^{\uparrow}$, KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, RealDiagonalNonSurjection$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, W, D, G, T, F, R, H, C, P, N$. It has no coordinate for a completed enumeration of all binary streams, host powersets, quotient stream equality, countable choice, or classical cardinal comparison.

Theorem 5879.2 (Finite diagonal obligations)**.** *For every accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier $K = (E, W, D, G, T, F, R, H, C, P, N)$, the local* NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$ *surface consists exactly of finite enumeration-window admission through E, binary observation through W, diagonal-bit formation through D, inspected-index disagreement through G, tree and fan routing through T, F, Real-facing non-surjection routing through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5879.1. The enumeration and observation rows are E and W, so every candidate read is confined to a displayed finite prefix window.

Build the diagonal row by induction on the finite index spine of W. The empty window contributes no bit and no disagreement row. At a successor index, read the displayed candidate bit at the diagonal coordinate and place the opposite bit in D.

The same induction fills G. Each successor step adds exactly the disagreement between the new diagonal bit and the candidate bit just inspected, while previous rows are transported componentwise by H and replayed by C.

The rows T, F, R do not add an infinite object. They only route the same finite disagreement ledger through the Kleene-tree, fan-window, and Real-facing non-surjection surfaces, so the listed rows exhaust the certificate without a host powerset, quotient stream equality, choice principle, or classical cardinal comparison. □

Theorem 5879.3 (No-enumeration window)**.** *Let K be an accepted* ConstructiveCantorTheorem$^{\uparrow}$ *carrier with a displayed enumeration window E of length n. The diagonal row D determines a Cantor-space prefix window such that for every inspected index $i < n$, the ith candidate of E differs from D at the displayed diagonal coordinate recorded in G. Hence the window cannot certify that its listed candidates exhaust the displayed Cantor-space prefixes.*

Proof. Induct on the finite enumeration window. For the empty window, there is no inspected candidate and the exhaustion claim has no displayed row.

At a successor window, the induction hypothesis supplies the disagreement rows for the previous candidates. The new candidate has a displayed diagonal coordinate in W, and the diagonal construction places the opposite binary value in D.

The new row of G records this disagreement. Componentwise hsame transport preserves the previous rows and the displayed Cont replay keeps the construction inside the same finite packet.

Therefore every inspected candidate is separated from the diagonal prefix at its own displayed coordinate. A finite window containing these disagreement rows cannot also certify that the same listed candidates exhaust the displayed Cantor-space prefix surface. □

Theorem 5879.4 (Cantor-space handoff)**.** *Every downstream consumer of a* ConstructiveCantorTheorem$^{\uparrow}$ *packet receives only the finite handoff formed by the enumeration window, binary observation window, diagonal-bit row, disagreement ledger, Kleene-tree row, fan-window row, Real-facing non-surjection route, transport, replay, provenance, and local naming. The handoff may be read by* CantorSpace$^{\uparrow}$ *and then by* RealDiagonalNonSurjection$^{\uparrow}$*, but it exports no completed stream quotient, powerset object, countable-choice selector, or classical uncountability principle.*

Proof. Start with the finite rows E, W, D, G of the carrier. By Definition 5879.2, these are the only rows that form and verify the diagonal disagreement.

Next read the Cantor-space side through chapter 1475. The Kleene-tree and fan rows T, F preserve finite prefix extension and bounded-window routing; they do not select a completed path or add an ambient compactness theorem.

Finally route the same disagreement ledger to chapter 3793 through R. Since the carrier has no coordinate for quotient stream equality, countable choice, host powersets, or classical cardinal comparison, the consumer receives exactly the finite handoff named in the statement. □

Falsifiable prediction: A purported ConstructiveCantorTheorem$^{\uparrow}$ consumer that reads a no-enumeration conclusion while omitting the enumeration window, diagonal-bit row, inspected-index disagreement ledger, Cantor-space prefix row, transport, replay, provenance, or local naming refutes the packet.

Independence witness: cantorspace, kleene-tree, fantheorem, real-diagonal-non-surjection: ConstructiveCantorT is not bijective to any listed sibling because it jointly carries an attempted enumeration window, a diagonal-bit construction, an inspected-index disagreement ledger, and a downstream Cantor-space-to-Real non-surjection route in one finite NameCert packet.

Closure status for NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 5879.1, Definition 5879.2, Definition 5879.3, and Definition 5879.4 seed the finite diagonal non-enumeration packet over Cantor-space prefix rows.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveCantorTheorem}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.ConstructiveCantorTheoremUp.ConstructiveCantorTheoremTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim a completed stream quotient, a host powerset object, a countable-choice selector, a classical cardinal comparison, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires Lean targets for carrier admission, diagonal-bit exactness, inspected-index disagreement, Cantor-space handoff, Real-facing route preservation, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveCantorTheoremUp.

Constructive story: A ConstructiveCantorTheorem$^{\uparrow}$ packet is generated from a finite attempted enumeration window, a finite binary observation window, a diagonal-bit row, an inspected-index disagreement ledger, Kleene-tree and fan-window rows, Real-facing non-surjection routing, componentwise transport, displayed replay, provenance, and a local NameCert row.

MarkovKakutaniFixedPoint$^{\uparrow}$ routes its common fixed-window packet through the compact-convex and set-valued fixed-point surfaces of chapter 1923 and chapter 3511.

Chapter 5880

Markov-Kakutani Fixed Point NameCert Construction

Following Markov and Kakutani, the $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet records the finite BEDC surface for a commuting affine family acting on a compact convex carrier. It is anchored by the convex carrier of chapter 208, the functional-analysis ambient rows of chapter 94, the compact-convex self-map handoff of chapter 1923, and the set-valued fixed-point surface of chapter 3511. The packet names a finite common fixed-point obligation route; it does not import a host compactness theorem, a global choice selector, an ambient locally convex space, or a completed real vector topology outside the displayed rows.

Definition 5880.1 (Markov-Kakutani fixed point carrier)**.** *A* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, A, F, C, W, I, H, R, P, N)$$

with the following displayed rows:

1. *D is the compact convex domain row, read through the carried* $\mathsf{ConvexSet}^{\uparrow}$ *and compact support surfaces.*
2. *A is the affine ambient row, exposing only the finite vector-operation and scalar-stability data required by the displayed family.*
3. *F is the commuting-family row, listing the finite subfamily of affine self-maps and their source and target admissions.*
4. *C is the commutation ledger, recording pairwise commutation only for the displayed finite family windows.*
5. *W is the finite fixed-window row, recording the common approximate fixed rows inspected by downstream consumers.*
6. *I is the invariant-subcarrier row, exposing the finite intersection handoff used to keep the common fixed-window route inside the compact convex carrier.*
7. *H records componentwise* hsame *transport for domain, affine ambient, family, commutation, fixed-window, invariant-subcarrier, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from a family request through $D, A, F, C, W, I, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{KakutaniFixedPoint}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $D, A, F, C, W, I, H, R, P, N$. It has no coordinate for an unbounded operator family, host compactness, selected fixed point, quotient equality of nets, or an ambient topological-vector-space theorem.

Theorem 5880.2 (Markov-Kakutani fixed point obligation surface)**.** *For every accepted* MarkovKakutaniFixedPoint *carrier*

$$M = (D, A, F, C, W, I, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$ *surface consists exactly of compact-convex domain admission through D, affine ambient admission through A, finite commuting-family admission through F, pairwise commutation exposure through C, fixed-window exposure through W, invariant-subcarrier handoff through I, componentwise* hsame *stability through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5880.1. The compact-convex, affine, commuting-family, commutation, fixed-window, and invariant-subcarrier data are exactly the rows D, A, F, C, W, I.

The domain and affine rows bind the finite ambient context before any map in the family is read. The family row F then lists only displayed affine self-maps, and the commutation ledger C records pairwise commutation inside that finite list.

The fixed-window and invariant-subcarrier rows W, I are the only route to a common fixed-point obligation. Finally H, R, P, N transport, replay, source, and name the same packet. Since no additional coordinate is present, the listed rows exhaust the NameCert surface. □

Theorem 5880.3 (Markov-Kakutani commuting handoff)**.** *Every accepted consumer of a* MarkovKakutaniFixedPoi *packet that asks for a common fixed-window read factors through*

$$\mathsf{ConvexSet}^{\uparrow},\ \mathsf{FunctionalAnalysis}^{\uparrow},\ \mathsf{SchauderFixedPoint}^{\uparrow},\ \mathsf{KakutaniFixedPoint}^{\uparrow} \longmapsto D, A, F, C, W, I, H, R, P, N.$$

The exported object is a finite commuting-family fixed-window packet. It is not a selected fixed point, a host compactness theorem, a global choice principle, a quotient of approximating nets, or an ambient locally convex-space completion.

Proof. Open the accepted carrier using Definition 5880.1. A consumer must first read the compact convex domain row D and the affine ambient row A, so the route is grounded in the displayed convex and functional-analysis siblings.

Next read the finite family row F and the commutation ledger C. These rows expose only the listed affine self-maps and their pairwise commutation windows, so no unbounded operator family or host selection principle enters the packet.

The fixed-window row W is then interpreted inside the invariant-subcarrier row I. The Schauder and Kakutani siblings provide fixed-point handoff surfaces for the displayed compact-convex route, but the Markov-Kakutani packet exports only the finite common-window ledger named by W, I.

Finally H, R, P, N preserve the same carrier under transport, replay, provenance, and local naming. The exclusion clauses in the carrier therefore prevent selected fixed points, host compactness, quotient-net equality, and ambient completion from leaving the packet. □

Falsifiable prediction: Every accepted $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ consumer must expose compact-convex domain, affine ambient, finite commuting-family, pairwise commutation, fixed-window,

invariant-subcarrier, hsame, Cont, Pkg, and NameCert rows before it can read a common fixed-window obligation.

Independence witness: convexset, functionalanalysis, schauder-fixed-point, kakutani-fixed-point: $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite commuting affine family, commutation ledger, fixed-window rows, invariant-subcarrier handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}$ packet is generated from compact-convex domain rows, affine ambient rows, finite commuting-family rows, pairwise commutation ledgers, fixed-window rows, invariant-subcarrier handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5880.1, Definition 5880.2, and Definition 5880.3 seed the finite Markov-Kakutani fixed-point packet over displayed compact-convex, affine, commuting-family, fixed-window, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MarkovKakutaniFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compactness, selected common fixed points, global choice, quotient equality of nets, ambient locally convex completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-convex domain admission, affine-family routing, commutation ledger coverage, fixed-window exposure, invariant-subcarrier handoff, transport, replay, provenance, and local naming for BEDC.Derived.MarkovKakutaniFixedPointUp.

Proposition 5880.4 (Markov-Kakutani fixed-window dependency route)**.** *Every accepted* MarkovKakutaniFixedPoi *consumer that reads the common fixed-window row W consumes* $\mathsf{KakutaniFixedPoint}^{\uparrow}$ *from chapter 3511 and* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *from chapter 1923 only through the displayed compact-convex fixed-window route $D, A, F, C, W, I, H, R, P, N$ of Definition 5880.3. The route records the horizontal fixed-point dependency while refusing a detached host compactness theorem, selected common fixed point, or ambient locally convex completion.*

$\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ routes its private/shared row-set ledger to the defect-refined decoration consumer of chapter 7630, while the child files keep the NameCert carrier and forward route separate.

Chapter 5881

Window-Six Private/Shared Decoration NameCert Construction

The WindowSixPrivateSharedDecoration$^{\uparrow}$ packet realizes the private/shared decoration enumeration of Definition 9462.1699 inside the DNA window-six boundary-tile source of subsection 9462.28.21. It consumes the boundary-tile packet of chapter 3393, the six-boundary coverage packet of chapter 3470, the boundary-trigger closure packet of chapter 3508, and the trigger-threshold packet of chapter 4217. The packet records private blocks, shared row sets, trigger coverage, decoration counts, transport, replay, provenance, and local naming as finite rows; it is not a biological mechanism, stochastic fitness model, hidden enumeration oracle, or quotient codon semantics.

Definition 5881.1 (Window-six private/shared decoration carrier)**.** *A* WindowSixPrivateSharedDecoration$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (B, S, T, E, D, H, C, P, N)$$

with the following displayed rows:

1. *B is the private-block row, recording the nonempty private coordinate blocks attached to each accepted decorated support.*
2. *S is the shared-row-set ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *T is the trigger-coverage row, read through the carried DNA boundary-tile, six-boundary coverage, closure, and threshold packets.*
4. *E is the decoration-count row, recording the finite enumeration factors from the source private/shared theorem.*
5. *D is the defect ledger, recording row-set-size contribution to the defect-refined decoration polynomial.*
6. *H records componentwise* hsame *transport for private blocks, shared row sets, trigger coverage, decoration count, defect ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a window-six decoration request through B, S, T, E, D, H, P, N.*
8. *P is the* Pkg *provenance over* WindowSixPrivateSharedDecoration$^{\uparrow}$, DnaWindowSixBoundaryTile$^{\uparrow}$, SixBoundaryTileCoverage$^{\uparrow}$, DnaBoundaryTriggerClosure$^{\uparrow}$, DNAWindowSixTriggerThreshold$^{\uparrow}$, *the source theorem of Definition 9462.1699,* BHist, hsame, Cont, Pkg, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $B, S, T, E, D, H, C, P, N$. It has no coordinate for biological mechanism, stochastic evolutionary semantics, quotient codon equality, hidden global enumeration, or an unrecorded trigger oracle.

Theorem 5881.2 (Window-six private/shared decoration obligations)**.** *For every accepted* WindowSixPrivateShared *carrier*

$$W = (B, S, T, E, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$ *surface consists exactly of private-block admission through B, shared-row-set admission through S, trigger-coverage exposure through T, decoration-count exposure through E, defect-ledger exposure through D, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5881.1. The private block, shared row set, trigger coverage, decoration count, defect ledger, transport, replay, provenance, and naming data are precisely $B, S, T, E, D, H, C, P, N$.

The rows B, S enforce the private/shared split: each accepted support has a displayed private block, and each shared coordinate is assigned to a row set of size at least two. The trigger-coverage row T then ties those rows to the existing window-six boundary, coverage, closure, and threshold packets.

The rows E, D record the enumeration and defect data used by the source theorem. Finally H, C, P, N preserve, replay, source, and name the same finite route. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5881.3 (Window-six private/shared decoration enumeration handoff)**.** *Every accepted* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *consumer that reads a private/shared window-six decoration factors through*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

The exported object is a finite private/shared row-set ledger with trigger-coverage and decoration-count rows. It is not a biological model, a quotient codon space, a hidden enumeration oracle, or an unrestricted trigger-search theorem.

Proof. Read the accepted carrier through Definition 5881.1. The private and shared ledgers B, S supply the displayed row-set split before a decoration count is exposed.

Next use T to route the split through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings. This prevents a decoration request from bypassing the window-six trigger-coverage surface.

The enumeration row E and defect row D then record the finite count and defect-polynomial data from the source theorem. The structural rows H, C, P, N transport, replay, source, and locally name the same rows.

Therefore every accepted consumer receives the finite packet described in the theorem, rather than biological semantics, quotient codon equality, hidden enumeration, or unrestricted trigger search. □

Theorem 5881.4 (Window-six private/shared decoration sibling lattice)**.** *Every accepted* WindowSixPrivateSharedD *packet consumes the sibling lattice*

$\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{SixBoundaryTileCoverage}^{\uparrow}$, $\mathsf{DnaBoundaryTriggerClosure}^{\uparrow}$, DNAWindowSixTriggerThres

before it exports the private/shared decoration ledger $B, S, T, E, D, H, C, P, N$. The lattice is finite and row-projection based: it displays boundary tiles, six-boundary coverage, trigger closure, and trigger-threshold evidence before the private block, shared row-set, trigger coverage, decoration count, and defect rows are consumed. It is not a biological mechanism, quotient codon semantics, stochastic evolutionary model, or hidden enumeration oracle.

Proof. Project the accepted carrier $W = (B, S, T, E, D, H, C, P, N)$ of Definition 5881.1. The private/shared ledger rows B, S, the trigger row T, the enumeration and defect rows E, D, and the structural rows H, C, P, N are the only exported coordinates.

Apply the enumeration handoff of Definition 5881.3. It forces every private/shared decoration read to pass through the boundary-tile, six-boundary coverage, boundary-trigger closure, and trigger-threshold siblings before $B, S, T, E, D, H, C, P, N$ are consumed.

Thus the dependency is a finite sibling lattice: each sibling supplies a displayed row that is projected before the decoration ledger is exported. Since the carrier has no coordinate for biological mechanism, quotient codon equality, stochastic evolution, or hidden enumeration, none of those interpretations is introduced by the lattice. □

Falsifiable prediction: Every accepted $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet must expose private-block, shared-row-set, trigger-coverage, decoration-count, hsame, Cont, Pkg, and NameCert rows before it can export a private/shared window-six decoration read.

Independence witness: dna-window-six-boundary-tile, six-boundary-tile-coverage, dna-boundary-trigger-closure, dna-window-six-trigger-threshold: $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ is not bijective to any listed sibling because the private/shared row-set ledger jointly carries private blocks, shared row sets, trigger coverage, decoration counts, defect rows, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.WindowSixPrivateSharedDecorationUp.TasteGate` must provide explicit Nontrivial $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ and FieldFaithful WindowSixPrivateSharedDecora instances preserving $B, S, T, E, D, H, C, P, N$ as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$.

Constructive story: A $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ packet is generated from private-block rows, shared-row-set ledgers, trigger-coverage rows, decoration-count rows, defect ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$, seed)

Scope closed: Definition 5881.1, Definition 5881.2, and Definition 5881.3 seed the finite private/shared window-six decoration packet over displayed boundary, coverage, closure, threshold, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global enumeration, unrestricted trigger search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block exposure, shared-row-set arity, trigger-coverage routing, decoration-count exposure, defect-ledger exposure, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.WindowSixPrivateSharedDecorationUp.

Proposition 5881.5 (Window-six private/shared decoration defect forward route)**.** *The private/shared row-set ledger exported by* $\mathsf{WindowSixPrivateSharedDecoration}^{\uparrow}$ *is an allowed child route for chapter 7630. Its rows $B, S, T, E, D, H, C, P, N$ supply the private block, shared row set, trigger coverage, enumeration, defect ledger, transport, replay, provenance, and local naming data used by the downstream defect-refined decoration polynomial route.*

Proof. Read the exported packet through Definition 5881.1. The private/shared ledger already separates private blocks from shared row sets and records the defect ledger as a displayed row.

The downstream chapter chapter 7630 consumes exactly that displayed ledger when it forms its defect-refined decoration polynomial route. Because the route is through the named child rows rather than through the hub file, it preserves the hub-only layout and records a genuine forward dependency between the two finite packets. □

Chapter 5882

A Concrete Naming Certificate for $\mathsf{SchurLemma}^{\uparrow}$

Following Schur and the finite-group representation-theory treatments of Serre and Fulton–Harris, the $\mathsf{SchurLemma}^{\uparrow}$ packet records the finite Hom-intertwiner boundary for irreducible representation rows. It sits between the finite representation carrier of chapter 3662, the character-theory trace surface of chapter 245, the representation-ring package of chapter 246, the orthogonality route of chapter 1603, and the character-table consumer of chapter 3574.

Definition 5882.1 (SchurLemma carrier)**.** *A* $\mathsf{SchurLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, H, S, T, R, P, N)$$

with the following displayed rows:

1. *U is an irreducible finite representation row over the same finite group source used by Definition 3662.1;*
2. *V is a second irreducible finite representation row over the same finite group source;*
3. *H is the displayed* Hom *and intertwiner row from U to V, recording equivariance rather than all host linear maps;*
4. *S is the scalar-or-zero boundary: when the two irreducible rows are the same finite representation row, accepted endomorphism reads factor through one scalar row, and when they are distinct no nonzero intertwiner row is exported;*
5. *T records componentwise* hsame *transport for the representation rows, Hom row, scalar-or-zero boundary, route, provenance, and naming rows;*
6. *R records displayed* Cont *replay from the finite representation packet through the Hom row into the scalar-or-zero boundary;*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteGroupRepresentation}^{\uparrow}$, $\mathsf{CharacterTheory}^{\uparrow}$, $\mathsf{RepresentationRing}^{\uparrow}$, $\mathsf{SchurOrthogonality}^{\uparrow}$, $\mathsf{CharacterTable}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows;*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *row.*

The carrier is exactly this finite representation-theory packet. It has no coordinate for a host abelian category, unrestricted module category, global algebra closure, or unrecorded linear-map equality.

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaTasteGate single carrier alignment`

Theorem 5882.2 (SchurLemma NameCert obligations)**.** *For the carrier of Definition 5882.1, the local* $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for* $K = (U, V, H, S, T, R, P, N)$*, irreducible finite representation exposure through* U, V*, Hom-intertwiner exposure through* H*, scalar-or-zero boundary exposure through* S*, componentwise* hsame *transport through* T*, displayed* Cont *replay through* R*, package provenance through* P*, and local naming through* N*.*

Proof. Project the displayed carrier. The packet lists U, V, H, S, T, R, P, N as its whole public surface, so the local certificate cannot introduce a second Hom object, a host representation category, or a hidden linear-map equality relation.

Read the representation rows first. The rows U and V are finite irreducible representation rows of the same group source, and the Hom row H is accepted only as an equivariant row between them.

The scalar-or-zero boundary is then read through S. It records scalar endomorphism reads in the same-row case and zero Hom reads in the distinct-row case. The rows T, R, P, N transport, replay, package, and name only that finite packet. □

Lean4 ✓ `BEDC.Derived.SchurLemmaUp.SchurLemmaCarrier namecert obligations`

Theorem 5882.3 (SchurLemma character-table handoff)**.** *Every accepted* $\mathsf{SchurLemma}^{\uparrow}$ *packet exports to* $\mathsf{CharacterTable}^{\uparrow}$ *only the finite Hom-intertwiner boundary needed by the Schur-orthogonality and character-pairing routes. The handoff reads* U, V, H, S *before* T, R, P, N *and exports no host representation category, unrestricted matrix algebra, quotient computation, or globally chosen basis.*

Proof. Start from Definition 5882.1. The downstream character-table consumer sees exactly the two irreducible representation rows, the displayed Hom-intertwiner row, and the scalar-or-zero boundary.

Use the neighboring representation-theory packets as routing anchors. The finite representation chapter supplies the representation rows, the character-theory and representation-ring chapters supply trace and sum surfaces, and the Schur-orthogonality chapter consumes the scalar-or-zero boundary before the character-table chapter reads pairings.

The structural rows T, R, P, N preserve that same finite packet under transport, replay, provenance, and local naming. Since no host category, unrestricted matrix algebra, quotient computation, or chosen-basis row appears in the carrier, the character-table handoff cannot export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SchurLemma}^{\uparrow}$ packet is generated from two irreducible finite representation rows, a displayed Hom-intertwiner row, a scalar-or-zero boundary row, componentwise transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, seed)

Scope closed: Definition 5882.1, Definition 5882.2, and Definition 5882.3 seed the finite Hom-intertwiner boundary between irreducible representation rows and the character-table consumer.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SchurLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close a host representation category, unrestricted module-category Hom, quotient computation, globally chosen bases, Lean verification, or bridge checking for $\mathsf{SchurLemma}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, irreducible-row exposure, Hom-intertwiner exactness, scalar-or-zero boundary, transport stability, replay stability, and local NameCert nonescape for BEDC.Derived.SchurLemmaUp.

Chapter 5883

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$

Following Weierstrass and Bishop–Bridges constructive analysis, the $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet records the interval-minimum half of the compact-real extreme-value route as finite BEDC NameCert data. It sits over the compact interval source of chapter 3986, the closed bounded interval source of chapter 1591, the continuous-map row of chapter 130, the located lower-bound route dual to chapter 1427, and the terminal real-name seal of chapter 38. The packet is minimum-specific: it exposes lower-bound attainment windows without importing open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5883.1 (Compact real interval minimum carrier)**.** *A* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, L, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, carrying endpoint order, interval containment, finite endpoint windows, and real endpoint seals.*
2. *K is the compact interval row, read through* $\mathsf{BishopCompactInterval}^{\uparrow}$ *or* $\mathsf{CompactMetric}^{\uparrow}$*, exposing only the finite inspection surface admitted for the interval source.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the real-valued map on the displayed compact interval.*
4. *L is the lower-bound and located-supremum-dual row, recording that every inspected value is above a displayed lower-bound candidate and that the candidate is reached through finite located windows rather than by trichotomy.*
5. *W is the finite witness-window row, recording the interval windows in which F and L meet a terminal value seal.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the value and lower-bound comparison windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *value seal row reached only after I, K, F, L, W, R have been displayed.*
8. *H records componentwise* hsame *transport for the interval source, compact source, continuous-map row, lower-bound row, witness windows, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.*

9. *C records displayed* Cont *replay from minimum-attainment requests through* $I, K, F, L, W, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for open-cover compactness, arbitrary real trichotomy, quotient real equality, a selected host minimum, a selected host maximum, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CompactRealIntervalMinimumUp.CompactRealIntervalMinimumTasteGate single carrier alignment`

Theorem 5883.2 (Compact real interval minimum NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ *carrier*

$$M = (I, K, F, L, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ *surface consists exactly of carrier admission for the closed bounded interval source, compact interval source, continuous-map row, lower-bound and located-supremum-dual row, finite witness-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *value seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No minimum obligation is stored outside these rows.*

Proof. Project the carrier of Definition 5883.1. The interval row I and compact row K expose the only compact-real source windows available to the packet, while F records the continuous real-valued map inspected on those windows.

Read the lower-bound and witness rows next. The row L gives the displayed lower-bound comparison as the dual of the located-supremum route, and W binds that comparison to finite source windows where the map row and bound row meet.

The regular and real-facing rows are downstream. The row R records the regular rational readback of the same finite comparisons, while E seals only those displayed reads as $\mathsf{Real}^{\uparrow}$ values.

The remaining rows H, C, P, N transport, replay, source, and locally name the same route. Since the carrier has no coordinate for open-cover compactness, trichotomy, quotient equality, selected host extrema, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5883.3 (Compact real interval minimum located attainment)**.** *Every accepted* CompactRealIntervalMinim *consumer that reads a minimum value factors through the displayed route*

$$I, K \longmapsto F, L \longmapsto W \longmapsto R, E \longmapsto H, C, P, N.$$

Thus the compact-real interval-minimum surface exports only a finite witness-window route from a closed bounded compact interval and continuous-map row to a located lower-bound row, regular rational readback, and $\mathsf{Real}^{\uparrow}$ *value seal. It does not export open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the source rows. The closed bounded interval source and compact interval source are displayed as I and K, so every later value comparison is scoped to the same finite compact-real interval windows.

Feed the source into the continuous-map and lower-bound rows. The row F exposes only displayed value reads, while L records the lower-bound comparison through the located-supremum-dual route rather than through a host order decision.

Use the witness-window and readback rows together. The row W records where the compact interval inspection meets the lower-bound comparison, and R repacks those comparisons as $\mathsf{RegSeqRat}^{\uparrow}$ data before the $\mathsf{Real}^{\uparrow}$ seal E is read.

Finally, transport and replay preserve the route through H and C, while P and N provide provenance and local naming. No row supplies open-cover compactness, trichotomy, quotient real equality, selected host extrema, or ambient completeness, so every accepted minimum read remains inside the displayed route. □

Falsifiable prediction: Every $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ minimum read factors through $\mathsf{ClosedBoundedInterva}$ $\mathsf{BishopCompactInterval}^{\uparrow}$ or $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$-dual lower-bound data, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before reaching finite witness windows and the local $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$ row.

Independence witness: bishopcompactinterval, closedboundedinterval, continuousmap, locatedsupremum, real: $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact interval source, continuous-map row, located lower-bound route, finite witness-window row, regular readback, Real value seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source row, a BishopCompactIntervalUp or CompactMetricUp compact source row, a ContinuousMapUp real-valued map row, a lower-bound row dual to LocatedSupremumUp, finite witness windows, RegSeqRatUp readbacks, RealUp value seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5883.1, Definition 5883.2, and Definition 5883.3 seed the finite compact-real interval-minimum route over interval, compact-source, continuous-map, lower-bound, witness-window, regular readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalMinimum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, trichotomy, quotient real equality, selected host extrema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactRealIntervalMinimum}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite witness-window exactness, lower-bound routing, regular readback, Real value sealing, replay stability, and nonescape for BEDC.Derived.CompactRealIntervalMinir

Chapter 5884

A Concrete Naming Certificate for $\mathsf{SimpleFunctionApproximation}^{\uparrow}$

Following Caratheodory and Bourbaki measurable approximation, the $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet names the finite-range measurable surface by which a measurable real-facing map is approximated without importing completed measure theory or host function equality. It sits after the preimage-ledger map surface of chapter 3461, the event-value rows of chapter 92, the generated-set boundary of chapter 3384, and the terminal real-name seal of chapter 38. The packet records finite range cells, measurable preimage rows, displayed tolerance data, finite observations, transport, replay, provenance, and local naming.

Definition 5884.1 (Simple function approximation carrier)**.** *A* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, F, R, E, T, O, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Measure}^{\uparrow}$ *row, exposing only the measurable event surface and event-value readback needed by the approximation packet.*
2. *F is the input* $\mathsf{MeasurableFunction}^{\uparrow}$ *row, read through its displayed target-event and preimage ledger rather than through a host function object.*
3. *R is the finite range ledger, listing the displayed* $\mathsf{Real}^{\uparrow}$ *values that the simple approximation may return.*
4. *E is the finite cell ledger, assigning each range row in R to a measurable source cell whose admission is read through* $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, *and the preimage rows of F.*
5. *T is the displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, naming the approximation budget used by the current consumer.*
6. *O is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, recording the visible source probes, range reads, and tolerance comparisons exposed to the packet.*
7. *H stores componentwise* hsame *transport for the measure source, measurable-function row, finite range ledger, cell ledger, tolerance row, observation row, provenance, and local naming rows.*
8. *C stores the displayed* Cont *replay from source-event inspection through finite cell selection, range-value readback, tolerance comparison, and observation replay.*

9. *P is the* Pkg *provenance over* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, F, R, E, T, O, H, C, P, N$. It has no coordinate for almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, or equality of host functions.

Lean4 ✓ `BEDC.Derived.SimpleFunctionApproximationUp.SimpleFunctionApproximationTasteGate single carrier alignment`

Theorem 5884.2 (Simple function approximation NameCert obligations)**.** *For the carrier of Definition 5884.1, the local* $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure source, measurable-function row, finite range ledger, measurable cell ledger, displayed* $\mathsf{Real}^{\uparrow}$ *tolerance row, finite* $\mathsf{StreamName}^{\uparrow}$ *observations, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted approximation consumer must read the finite cells through E, the measurable preimage route through F, the range values through R, and the tolerance comparison through T before any sibling* $\mathsf{Measure}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or* $\mathsf{StreamName}^{\uparrow}$ *row is exported.*

Proof. Project the carrier $S = (M, F, R, E, T, O, H, C, P, N)$. The measure source M and measurable-function row F are the only rows from which measurable event and preimage information can be read.

Read the finite approximation data. The range ledger R supplies the displayed real values, the cell ledger E ties each range value to a measurable source cell, and the tolerance row T records the visible approximation budget against which the current observations are compared.

Replay the observation and structural rows. The row O records the finite StreamName probes, while H, C, P, N transport, replay, source, and name the same route through the sibling certificates. Since no further coordinate is present, the listed rows exhaust the NameCert obligation surface and cannot export quotient classes, completed measures, selected representatives, ambient integration theorems, or host function equality. □

Theorem 5884.3 (Simple function approximation finite-cell route)**.** *Every accepted* $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ *read factors through the finite route*

$$M, F \longmapsto E \longmapsto R, T \longmapsto O \longmapsto H, C, P, N.$$

Thus a consumer receives a finite measurable partition-by-cells with displayed real values and a displayed tolerance budget, not an almost-everywhere quotient class, completed measure, selected representative, ambient integral, or equality proof for host functions.

Proof. Start with an accepted approximation request. The request first exposes the measure source and the measurable-function row, so every cell read is grounded in the displayed source-event and preimage ledgers.

The cell ledger E is finite. Each displayed cell is paired with a range value from R, and the tolerance row T supplies the visible real-valued budget for the comparison requested by the consumer.

Finally replay the observations and structural rows. The finite StreamName row O records only the exposed probes, while H, C, P, N transport and name the route. No hidden coordinate remains from which a quotient, completion, representative selection, integration theorem, or host equality could be recovered. □

Falsifiable prediction: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ consumer that obtains an almost-everywhere quotient class, completed measure, selected representative, ambient integration theorem, or host function equality without first reading $M, F, R, E, T, O, H, C, P, N$ refutes the carrier of Definition 5884.1.

Independence witness: measurablefunction, measure, borelset, real, streamname: no listed sibling supplies the whole ten-row packet $S = (M, F, R, E, T, O, H, C, P, N)$, because each sibling contributes only preimage-ledger, event-value, generated-set, real-name, or finite-observation data while this chapter binds the finite-range approximation surface with tolerance, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{SimpleFunctionApproximation}^{\uparrow}$ packet is generated from a measure source, a measurable-function preimage ledger, finite range cells, a displayed Real tolerance row, finite StreamName observations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5884.1, Definition 5884.2, and Definition 5884.3 seed the finite-range measurable approximation surface over source events, preimage rows, cell ledgers, range values, tolerance rows, observations, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SimpleFunctionApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close almost-everywhere quotient classes, completed measures, selected representatives, ambient integration theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cell exactness, measurable preimage preservation, Real tolerance replay, StreamName observation stability, and quotient-refusal proofs for BEDC.Derived.SimpleFunctionApproxima

Chapter 5885

A Concrete Naming Certificate for $\mathsf{DyadicModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicModulus}^{\uparrow}$ packet records the finite tolerance-modulus surface by which dyadic precision requests become regular rational readback before the terminal real seal is consumed. It sits between the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational handoff of chapter 301, the terminal seal boundary of chapter 38, and the Cauchy-modulus route carried by $\mathsf{CauchyModulus}^{\uparrow}$. The packet is a finite NameCert carrier for Real and Completion consumers; it is not a host modulus function, quotient stream equality, selected limit, countable choice, or ambient completeness principle.

Definition 5885.1 (DyadicModulus carrier)**.** *A* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming dyadic scales, finite comparison budgets, rounding windows, and precision requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row whose visible entries are exactly the observations demanded by the tolerance row.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained by reading the stream windows under the dyadic tolerance ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary, available only after D, S, R have been displayed.*
5. *H records componentwise* hsame *transport for tolerance, stream-window, regular-readback, seal, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from each dyadic precision request through D, S, R, E, H, P, N.*
7. *P is the* Pkg *provenance over* $\mathsf{DyadicModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, E, H, C, P, N. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, ambient Real completeness, or a detached host-level modulus function.

Lean4 ✓ `BEDC.Derived.DyadicModulusUp.DyadicModulusTasteGate single carrier alignment`

Theorem 5885.2 (DyadicModulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier* $M = (D, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window exposure, regular rational readback, terminal Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Every Real or Completion consumer must read the displayed route

$$D \longmapsto S \longmapsto R \longmapsto E \longmapsto H, C, P, N$$

before using the packet as a dyadic tolerance-modulus certificate.

Proof. Project the carrier of Definition 5885.1. The tuple displays exactly the dyadic tolerance row, stream-window row, regular-readback row, terminal seal, transport, replay, provenance, and local name.

The analytic route is forced in order. A precision request first reads D, then the finite stream observations in S, then the regular rational handoff R, and only then reaches the seal boundary E.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no quotient-stream, selected-limit, choice, host-equality, ambient-completeness, or detached-modulus coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 5885.3 (DyadicModulus RegSeqRat handoff)**.** *Let* $M = (D, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{DyadicModulus}^{\uparrow}$ *carrier. Any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, or Completion-facing consumer of the modulus row must factor through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the same* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row* R *before the terminal seal row* E *is read.*

The handoff route is

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

as rows of one BHist *packet, with transport, replay, provenance, and local naming supplied by* H, C, P, N*.*

Proof. Start from Definition 5885.2. It already places every tolerance-modulus consumer on the displayed route D, S, R, E, H, C, P, N.

The regular rational readback is the row R, and R is downstream of both the dyadic tolerance row D and the stream-window row S. Thus a consumer cannot read a regular name, Real seal, or Completion boundary before the finite tolerance and window data have been displayed.

Finally inspect the excluded coordinates of Definition 5885.1. Because no quotient stream, selected representative, host equality, countable-choice schedule, ambient completeness theorem, or detached modulus function is present, the RegSeqRat handoff is exactly the stated finite route. □

Falsifiable prediction: An accepted $\mathsf{DyadicModulus}^{\uparrow}$ consumer is refuted if it reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, or a Completion-facing seal without first displaying one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local $\mathsf{NameCert}$ row.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-modulus: $\mathsf{DyadicModulus}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately provide tolerance arithmetic,

finite windows, regular rational readback, terminal sealing, or threshold scheduling, while this packet binds them into one finite tolerance-modulus handoff route.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}$.

Constructive story: A $\mathsf{DyadicModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window row, a RegSeqRat handoff row, a RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5885.1, Definition 5885.2, and Definition 5885.3 seed the finite dyadic tolerance-modulus carrier between dyadic precision data and regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window exposure, RegSeqRat handoff, Real seal handoff, structural transport, and non-escape for BEDC.Derived.DyadicModulusUp.

Chapter 5886

A Concrete Naming Certificate for $\mathsf{CauchyFilterSelector}^{\uparrow}$

Following Bishop and Bridges and Kelley filter terminology, the $\mathsf{CauchyFilterSelector}^{\uparrow}$ packet records the finite selector interface by which a Cauchy-filter read can be inspected through named windows before any Real or Completion consumer reads the resulting boundary. It sits after the Cauchy-filter surface of chapter 333, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the completion handoff of chapter 1506. The packet gives the Phase Real and Completion route a finite selector row without quotient filters, host-selected representatives, countable choice, or ambient Real completeness.

Definition 5886.1 (Cauchy filter selector carrier)**.** *A* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the source* $\mathsf{CauchyFilter}^{\uparrow}$ *row, including its displayed base packets, directed refinement ledger, and Cauchy evidence surface.*
2. *W is the finite selector-window row. It records the displayed filter request, the finite family of inspected base packets, and the chosen window among those already exposed by F.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the selected windows in W.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the filter request, the selector comparison, and the readback radius consumed by R.*
5. *E is the completion-or-Real seal row, read either as a* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff or as the downstream* $\mathsf{Real}^{\uparrow}$ *seal after F, W, R, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter row, selector windows, regular readback, dyadic ledger, seal row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a Cauchy-filter request through F, W, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, R, D, E, H, C, P, N$. It has no coordinate for quotient filters, host-selected representatives, countable-choice schedules, host real equality, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CauchyFilterSelectorUp.CauchyFilterSelectorTasteGate single carrier alignment`

Theorem 5886.2 (Cauchy filter selector NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier $S = (F, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source Cauchy-filter data, selector windows, regular readback, dyadic tolerance ledger, completion-or-Real seal, transport, replay, provenance, and local naming row;*
2. *selector-window exactness, requiring the selected window to be one of the finite windows already displayed in W over the source filter row F;*
3. *regular readback handoff, requiring every numeric read to pass through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R after the selector window is fixed;*
4. *dyadic ledger stability, requiring the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D to govern the selector comparison and the readback radius;*
5. *seal boundary, requiring any* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer to read E only after F, W, R, D have been displayed and replayed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5886.1. The visible packet consists of the source filter row F, selector windows W, regular readback R, dyadic ledger D, seal boundary E, and the structural rows H, C, P, N.

Selector-window exactness is row-local. The selector does not choose from an undisplayed filter; it names one of the finite windows stored in W and read over the already exposed Cauchy-filter row F.

The readback and tolerance obligations are ordered by the carrier. A numeric read reaches R only after W fixes the finite window, and the same dyadic ledger D supplies the tolerance budget for both the selector comparison and the regular readback radius.

The seal boundary is downstream. The replay row C routes every completion-facing or real-facing request through F, W, R, D before E, while H preserves this order under componentwise transport.

Since the carrier contains no quotient-filter coordinate, host-selected representative, choice schedule, host real equality, or Real completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 5886.3 (Cauchy filter selector RegSeqRat handoff)**.** *Let $S = (F, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier. Every selector read that exports regular numerical data factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}.$$

Concretely, the source filter row F exposes the finite base request, the selector-window row W exposes the inspected $\mathsf{StreamName}^{\uparrow}$ *windows, the readback row R exposes the* $\mathsf{RegSeqRat}^{\uparrow}$ *data, and the dyadic ledger D supplies the tolerance budget used by that readback.*

Proof. Project the accepted carrier. The source request is held in F, but F alone does not expose a regular rational name to downstream consumers.

Replay the selector route by C. The request first reaches the finite selector windows W, and those windows are the only place where the relevant $\mathsf{StreamName}^{\uparrow}$ observations are visible.

After W is fixed, the readback row R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ data. The tolerance row D is then read as the dyadic budget controlling the selector comparison and the regular readback.

Transport by H preserves the same order. Thus the handoff is a finite BEDC route from Cauchy filters to stream windows, regular readbacks, and dyadic tolerances, not a quotient of filters or a selected host representative. □

Theorem 5886.4 (Cauchy filter selector Real and completion seal boundary)**.** *For every accepted* $\mathsf{CauchyFilterSelector}^{\uparrow}$ *carrier* $S = (F, W, R, D, E, H, C, P, N)$*, every* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read exposed by* P *factors through the seal row* E *only after the source Cauchy-filter data, finite selector windows, regular readback, and dyadic tolerance ledger have been displayed. The visible boundary is*

$$F, W, R, D \longmapsto E \longmapsto H, C, P, N.$$

Therefore the packet provides a finite selector interface for the Real and Completion route without exporting quotient filters, host-selected representatives, countable choice, host real equality, or ambient Real completeness.

Proof. Read the seal row in the carrier. The row E is placed after the source filter data, selector windows, regular readback, and dyadic tolerance ledger, so a completion-facing or real-facing consumer cannot reach it from F alone.

The replay row C fixes the same order for every accepted request. It first consumes F, then the selector windows W, then the regular readback R, and then the dyadic tolerance ledger D before the seal row is available.

Componentwise transport by H does not add a new route. It preserves the displayed rows under classifier comparison, while P and N record only the provenance and local naming boundary of that finite route.

The excluded objects are absent from the carrier and from the replay path. Hence the completion-or-Real seal is exactly the displayed downstream boundary, not a quotient filter, selected representative, choice schedule, host equality, or completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5886.1, Definition 5886.2, Definition 5886.3, and Definition 5886.4 seed the finite selector interface between CauchyFilter, StreamName, RegSeqRat, DyadicRatCore, Real, and MetricCompletion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyFilterSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filters, host-selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Lean closure requires carrier admission, selector-window exactness, RegSeqRat handoff, dyadic ledger stability, and Real or Completion seal boundary for BEDC.Derived.CauchyFilterSelectorUp.

Constructive story: The construction starts from a displayed CauchyFilter row, reads a finite selector window through StreamName observations, extracts RegSeqRat data under a DyadicRatCore tolerance ledger, and exposes Real or MetricCompletion access only through the downstream seal row.

The $\mathsf{Cauchy}^{\uparrow}$ packet is consumed through chapter 301 and chapter 38 before downstream completion or calculus reads inspect the finite Cauchy windows.

Chapter 5887

A Concrete Naming Certificate for $\mathsf{CauchyModulusSpace}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet records a carrier-space interface for explicit Cauchy modulus schedules. It is anchored in the single-modulus certificate of chapter 292, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real seal of chapter 38, and the fast Cauchy sequence route of chapter 2185. The chapter treats a family of modulus schedules as finite naming data, not as a quotient of real sequences or as an ambient completeness theorem.

Definition 5887.1 (Cauchy modulus space carrier)**.** *A* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, S, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index row for the family of explicit modulus schedules.*
2. *S assigns to each displayed index a* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, including the monotone precision requests used by the family.*
3. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows inspected by the schedule family.*
4. *D is the dyadic tolerance ledger, read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled windows.*
6. *Q records the* $\mathsf{Real}^{\uparrow}$ *seal boundaries at which a scheduled read may leave rational and dyadic data for a real-facing consumer.*
7. *H records componentwise* hsame *transport for $I, S, W, D, R, Q, C, P, N$.*
8. *C records displayed* Cont *replay from an index and precision request through the schedule row, stream window, dyadic tolerance ledger, regular rational readback, and real seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusSpace}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only after transporting the displayed rows componentwise by hsame*. It has no coordinate for an unscheduled Cauchy oracle, quotient equality of sequences, a selected real limit, countable choice, or an ambient completion theorem.*

Lean4 ✓ `BEDC.Derived.CauchyModulusSpaceUp.CauchyModulusSpaceTasteGate single carrier alignment`

Theorem 5887.2 (Cauchy modulus space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index family, schedule admission for the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window admission, dyadic tolerance compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal-boundary compatibility, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming for the packet.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceNameCertObligations`

Proof. Project the accepted carrier of Definition 5887.1. The analytic data are precisely I, S, W, D, R, Q, and the structural rows are precisely H, C, P, N.

Schedule admission is rowwise over I. Each index exposes its explicit modulus schedule before any stream window is inspected, so the packet cannot replace the family by an unscheduled oracle.

The tolerance and readback rows are then forced through D and R. The real boundary row Q is available only after those rational and dyadic ledgers have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5887.3 (Cauchy modulus space finite route)**.** *Every real-facing comparison of two accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carriers factors through finite schedule rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows, dyadic tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{Real}^{\uparrow}$ *seal boundaries, componentwise* hsame*, displayed* Cont*,* Pkg*, and the local* NameCert *rows. The route exports a shared modulus-comparison surface for later diagonal* $\mathsf{Real}^{\uparrow}$ *completion arguments without introducing quotient sequence equality, selected limits, countable choice, or ambient completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFiniteRoute`

Proof. Open both carriers using Definition 5887.1. A comparison can inspect only the finite index rows, schedule rows, stream windows, tolerance ledgers, readback rows, and seal boundaries.

Componentwise hsame in H transports the two packets to a common finite surface. Displayed Cont replay in C keeps each precision request attached to its schedule, stream window, dyadic tolerance, regular readback, and real seal boundary.

The provenance and naming rows P, N package that finite surface as the local certificate. Thus later diagonal completion routes can share modulus comparison data without receiving any quotient or choice-based object. □

Theorem 5887.4 (CauchyModulusSpace seed interface)**.** *Every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet exports its modulus schedule family to the L10 four-object seal route only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{Cauchy}^{\uparrow}$ *root route, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. No consumer of this seed interface can expose quotient stream equality, a selected limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness as part of the packet surface.*

Proof. Project the carrier of Definition 5887.1. The modulus family is displayed by I, S, and every inspected request is routed through the finite observation rows W, D, R, Q before the structural rows H, C, P, N can name or replay it.

Apply Definition 5887.3. Any real-facing comparison has already factored through finite schedule rows, stream windows, dyadic tolerances, regular rational readbacks, and the terminal Real seal boundary.

The $\mathsf{Cauchy}^{\uparrow}$ root route receives exactly that finite schedule surface. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, or ambient completeness, the seed interface cannot export any of those host objects to L10 Real and Completion consumers. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceSeedInterface`

Theorem 5887.5 (CauchyModulusSpace tail composition)**.** *For every accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *carrier*

$$M = (I, S, W, D, R, Q, H, C, P, N),$$

two displayed modulus schedules in the family compose only through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows. The composed tail read is the finite route*

$$I, S, W, D, R, Q \longmapsto H, C, P, N$$

and it exports no quotient sequence equality, selected limit, countable choice, or ambient completeness object.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceTailComposition`

Proof. Project the carrier of Definition 5887.1. The two schedules are displayed rows under S indexed by I, and their inspected tails must pass through the same finite window row W.

Apply the finite route and seed interface, Definition 5887.3 and Definition 5887.4. These results force the tolerance ledger D, regular readback R, and Real seal boundary Q to remain visible before the composed tail can be consumed.

Finally replay the structural rows. Componentwise transport H, displayed continuation C, provenance P, and local naming N preserve the same finite schedule surface, so tail composition does not add quotient equality, selected limits, countable choice, or ambient completeness. □

Theorem 5887.6 (CauchyModulusSpace filter handoff)**.** *Every CauchyFilter or CauchyFilterCompletion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet receives the modulus family only through the finite handoff*

$$I, S, W, D, R, Q, H, C, P, N.$$

The handoff may be consumed as filter-facing Cauchy evidence, but it cannot be reinterpreted as quotient sequence equality, a selected limit, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceFilterHandoff`

Proof. Begin with a filter-facing read. The seed interface of Definition 5887.4 already routes every scheduled request through finite stream windows, dyadic tolerances, regular rational readback, the Cauchy root route, and the terminal Real seal.

Use tail composition from Definition 5887.5. It shows that composing displayed schedules preserves the same finite route through I, S, W, D, R, Q before structural replay.

The filter consumer therefore receives only the transported and replayed packet named by H, C, P, N. Since the carrier contains no quotient equality coordinate, no selected limit, no

countable-choice schedule, and no ambient completeness row, CauchyFilter and CauchyFilterCompletion may consume only the displayed finite handoff. □

Theorem 5887.7 (CauchyModulusSpace diagonal-family exactness)**.** *A diagonal Real or Completion consumer of an accepted* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *packet may compare schedule families only through the displayed rows*

$$I, S, W, D, R, Q, H, C, P, N.$$

The diagonal request must be routed through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary before any structural transport, replay, provenance, or local naming row can be consumed. The comparison exports no quotient stream equality, selected limit, countable-choice schedule, or ambient Real completeness theorem.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyModulusSpaceDiagonalFamilyExactness`

Proof. Start with the finite route of Definition 5887.3. It already factors every real-facing comparison of two accepted packets through the finite schedule rows, stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, and the structural rows H, C, P, N.

Apply the seed interface of Definition 5887.4. It binds every L10 request to the schedule family I, S before the finite windows W, tolerance ledger D, readback R, and seal boundary Q are exposed.

Finally use the filter handoff of Definition 5887.6. A diagonal completion consumer is still a consumer of the same finite handoff, so it receives only $I, S, W, D, R, Q, H, C, P, N$ and cannot add quotient equality, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ consumer that compares schedule families while omitting $I, S, W, D, R, Q, H, C, P, N$ refutes the carrier surface of Definition 5887.1.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, fast-cauchy-sequence: no listed sibling binds a family of explicit Cauchy modulus schedules together with finite stream windows, dyadic tolerance ledgers, regular rational readbacks, Real seal boundaries, componentwise transport, replay, provenance, and local naming in one carrier-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusSpace}^{\uparrow}$ packet is generated from a finite family of explicit Cauchy modulus schedules, finite StreamName windows, dyadic tolerance ledgers, RegSeqRat readbacks, Real seal boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5887.1, Definition 5887.2, and Definition 5887.3 seed the carrier-space interface for explicit modulus schedule families and shared finite comparison routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close diagonal Real completion, quotient equality of Cauchy sequences, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule-family exactness, dyadic and regular readback compatibility, Real-seal boundary nonescape, and a Lean carrier formalization for BEDC.Derived.CauchyModulusSpaceUp.

Chapter 5888

A Concrete Naming Certificate for $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$

Following Bishop and Bridges on constructive completion and the finite-limit language of regular category theory, the $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet records the boundary where a displayed Cauchy-completion route preserves the finite-limit rows that its source already exposes. It sits beside the Cauchy source route of chapter 3413, the metric-completion universal-property route of chapter 2124, the functorial completion route of chapter 423, the regular-rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only a finite left-exactness boundary for displayed completion data; it does not assert quotient completions, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5888.1 (Cauchy completion left-exactness carrier)**.** *A* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (S, U, D, K, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source finite-limit row, recording the displayed terminal, product, equalizer, or kernel-pair data available before completion is read.*
2. *U is the completion unit row, recording the source-to-completion unit reads for every object named by S.*
3. *D is the dense image row, keeping each completed finite-limit comparison tied to the same source rows and unit rows.*
4. *K is the kernel-pair preservation row, recording that equalizer and pullback comparisons are read through the displayed source kernel-pair data rather than through host quotient equality.*
5. *E is the extension ledger, recording the universal-property and functorial completion rows used to transport finite-limit witnesses into completion-facing reads.*
6. *H records componentwise* hsame *transport for S, U, D, K, E, C, P, N.*
7. *C records displayed* Cont *replay from a finite-limit request through S, U, D, K, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, selected finite limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionLeftExactnessUp.TasteGate.CauchyCompletionLeftExactnessTasteGate single carrier alignment`

Theorem 5888.2 (Cauchy completion left-exactness NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the source finite-limit row, completion unit rows, dense image rows, kernel-pair preservation row, extension ledger, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every completion-facing finite-limit read must expose S, U, D, K before E, H, C, P, N can be consumed.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessNameCertObligations`

Proof. Project the carrier of Definition 5888.1. The source finite-limit row, unit row, dense image row, kernel-pair row, extension ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The finite-limit boundary is read through S, U, D, K. The source row supplies the terminal, product, equalizer, or kernel-pair data; the unit row embeds the displayed source names into completion-facing reads; the dense image row keeps those reads tied to source data; and the kernel-pair row confines exactness comparisons to displayed finite rows.

The remaining rows E, H, C, P, N transport and name the same route. Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient completeness have no carrier coordinate, projection of the displayed rows exhausts the NameCert obligations. □

Theorem 5888.3 (Cauchy completion left-exactness carrier admission)**.** *For every accepted* CauchyCompletionLeftE *carrier*

$$L = (S, U, D, K, E, H, C, P, N),$$

the source finite-limit row S, completion unit row U, dense image row D, kernel-pair preservation row K, extension ledger E, componentwise hsame *transport H, displayed* Cont *replay C, provenance row P, and local* NameCert *row N are all admitted as finite coordinates of the same packet. No completion-facing read may treat one of these rows as ambient host data detached from the carrier.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactness carrier admission`

Proof. Project the carrier of Definition 5888.1. Its coordinates are exactly $S, U, D, K, E, H, C, P, N$, so each named row is admitted by carrier membership.

Apply Definition 5888.2. The obligation surface requires every completion-facing finite-limit read to expose S, U, D, K before the extension, transport, replay, provenance, and naming rows can be consumed.

Since quotient completion equality, selected representatives, selected finite limits, countable choice, host equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness have no carrier coordinate, none of the admitted rows can be reread as detached host data. □

Theorem 5888.4 (Cauchy completion left-exactness unit-density route)**.** *Every displayed finite-limit comparison read through an accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier factors first through the source finite-limit row* S*, then through the completion unit row* U*, and then through the dense image row* D *before the extension ledger* E *and structural rows* H, C, P, N *are inspected. This route records the finite source-to-completion passage without selecting representatives of completed objects or invoking countable choice.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessUnitDensityRoute`

Proof. Begin with the carrier admission theorem Definition 5888.3. It keeps S, U, D as visible coordinates of the same finite packet.

The source row S supplies the finite-limit data before completion is read. The unit row U records the displayed source-to-completion unit for each object named by S, and the dense image row D keeps the resulting completion-facing comparison tied back to the same source and unit rows.

Only after this route is exposed can E, H, C, P, N be consumed. The extension ledger and the structural rows transport, replay, package, and name the source-unit-density passage; they do not add selected representatives, countable choice, or host equality of completed objects. □

Theorem 5888.5 (Cauchy completion left-exactness kernel-pair preservation)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any equalizer, pullback, or kernel-pair comparison visible after completion is read through the displayed kernel-pair row* K *together with the source finite-limit row* S*, completion unit row* U*, and dense image row* D*. The comparison cannot be exported as quotient completion equality, a selected finite limit, or ambient equality of completed objects.*

Proof. Project the kernel-pair coordinate K from the carrier. By definition, K records equalizer and pullback comparisons through displayed source kernel-pair data rather than through a host quotient.

Use the unit-density route theorem Definition 5888.4. It places every completion-facing finite-limit comparison behind S, U, D, so the kernel-pair row is read as part of the same source-to-completion passage.

Finally apply Definition 5888.2. The rows E, H, C, P, N only extend, transport, replay, package, and name the displayed comparison. They add no coordinate for quotient completion equality, selected finite limits, or ambient equality of completed objects, so preservation remains confined to K and its source, unit, and dense-image context. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessKernelPairPreservation`

Theorem 5888.6 (Cauchy completion left-exactness regular-limit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any regular Cauchy limit read used by a finite-limit comparison factors through the source finite-limit row* S*, the completion unit row* U*, the dense image row* D*, the kernel-pair row* K*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger* E *is consumed. Thus the regular-limit comparison is a pullback of displayed source, unit, dense-image, kernel-pair, regular-readback, and real-seal rows, not a quotient completion or a selected representative of a completed object.*

Proof. Project the accepted carrier. The rows $S, U, D, K, E, H, C, P, N$ are the only admitted coordinates, so a finite-limit comparison cannot begin from a hidden completed object or a host-level limit.

Apply Definition 5888.4. It places every completion-facing finite-limit comparison behind the source finite-limit row, the completion unit row, and the dense image row before the extension ledger and structural rows are inspected.

Next apply Definition 5888.5. Equalizer, pullback, and kernel-pair comparisons remain confined to the displayed kernel-pair row together with the same source, unit, and dense-image context.

The regular Cauchy limit read is then exposed through the sibling $\mathsf{RegSeqRat}^{\uparrow}$ readback surface named in the chapter introduction, and the resulting endpoint is sealed only through the sibling $\mathsf{Real}^{\uparrow}$ boundary. Since those sibling rows are read after S, U, D, K and before E, the comparison has the announced displayed pullback route. No coordinate remains for quotient completion equality, selected representatives, selected finite limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegularLimitPullback`

Theorem 5888.7 (Cauchy completion left-exactness RegSeqRat unit pullback)**.** *For every accepted* $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ *carrier* $L = (S, U, D, K, E, H, C, P, N)$*, any finite-limit comparison whose completion unit route reaches a* $\mathsf{RegSeqRat}^{\uparrow}$ *readback factors first through the source finite-limit row* S*, completion unit row* U*, dense image row* D*, and kernel-pair row* K*. It then reads the* $\mathsf{RegSeqRat}^{\uparrow}$ *surface and terminal* $\mathsf{Real}^{\uparrow}$ *seal before the extension ledger and structural rows* E, H, C, P, N *are consumed. Componentwise* hsame *transport preserves that order, so the exported comparison contains no quotient completion, selected representative, selected finite limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyCompletionLeftExactnessRegSeqRatUnitPullback`

Proof. Start with the carrier admission theorem Definition 5888.3. It keeps $S, U, D, K, E, H, C, P, N$ as finite coordinates of one accepted packet, so the comparison cannot begin from an external completed object.

Apply the unit-density route theorem Definition 5888.4. This places every completion-facing finite-limit comparison behind S, U, D before the extension ledger and structural rows are inspected.

Next apply the kernel-pair preservation theorem Definition 5888.5. Pullback and equalizer comparisons remain confined to K together with the same source, unit, and dense-image context.

Finally use the regular-limit pullback theorem Definition 5888.6. Its readback route inserts the $\mathsf{RegSeqRat}^{\uparrow}$ surface and the terminal $\mathsf{Real}^{\uparrow}$ seal after S, U, D, K and before E, H, C, P, N. Componentwise hsame transport therefore preserves the displayed order, while the excluded quotient, selection, choice, host-equality, and ambient-completeness rows remain absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionLeftExactness}^{\uparrow}$ packet is generated from source finite-limit rows, completion unit rows, dense image rows, kernel-pair preservation, an extension ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5888.1 and Definition 5888.2 seed the finite left-exactness boundary for displayed Bishop-style Cauchy completion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionLeftExactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected finite limits, selected representatives, countable choice, host equality of completed objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source finite-limit exposure, unit-row exposure, dense image confinement, kernel-pair preservation, extension-ledger confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionLeftExactnessUp.

Theorem 5888.8 (Cauchy root UnaryHistory admission)**.** *Every accepted root* $\mathsf{Cauchy}^{\uparrow}$ *consumer read admits a UnaryHistory-facing presentation before it enters any filter, modulus, or Real-facing route. The admitted packet is the finite row surface*

$$U = (S, Q, D, R, E, H, C, P, N),$$

where S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* Q *is the directed tail-request row,* D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, and* H, C, P, N *are componentwise transport, displayed replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootUnaryAdmission`

Proof. Project the root stream and tail-request carriers already displayed for $\mathsf{Cauchy}^{\uparrow}$. Their shared rows expose finite StreamName windows before any tolerance, readback, or terminal seal can be read.

The directed request row supplies the unary admission point. It records one visible later-window request over the same finite stream surface, so the read is a UnaryHistory-facing row rather than a parameter transport over an ambient sequence.

The dyadic, regular readback, Real seal, and structural rows then repack the same finite route. No quotient stream, selected limit, host equality, countable choice, or ambient completeness row is exported. □

Theorem 5888.9 (Cauchy root filter-modulus compatibility)**.** *For every admitted root packet* $U = (S, Q, D, R, E, H, C, P, N)$*, any* $\mathsf{CauchyFilter}^{\uparrow}$ *or finite filter consumer that reads the tail request must use the same modulus-compatible route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyTailRequest}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The filter row may refine the visible window named by S *through* Q*, but it cannot replace* D *by an external tolerance oracle or replace* R *by a selected limit.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootFilterModulusCompatibility`

Proof. Start with the admitted packet of Definition 5888.8. The stream row S and directed request row Q are read before any filter-facing or modulus-facing datum.

Apply the existing Cauchy filter Real-seal route. It inserts the finite request row before the dyadic tolerance ledger, so a filter consumer reaches D and R only through the displayed tail request.

The structural rows preserve that order under hsame transport and Cont replay. Thus a finite filter may narrow the displayed request, but the modulus and regular readback remain the same BEDC rows. □

Theorem 5888.10 (Cauchy root RegSeqRat Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-to-*$\mathsf{Real}^{\uparrow}$ *handoff exported by a root* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the finite windows, directed request, dyadic tolerance ledger, and regular rational readback of Definition 5888.9. The Real seal* E *is available only after* S, Q, D, R, H, C, P, N *have been displayed, and it exports no quotient filter, selected real, countable-choice schedule, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Read the filter-modulus compatibility theorem. It places the StreamName window, directed request, dyadic tolerance ledger, and RegSeqRat readback in a fixed order before any Real-facing boundary.

The handoff row is exactly R followed by E. A Real-facing consumer therefore receives rational readback through $\mathsf{RegSeqRat}^{\uparrow}$ first and may inspect the terminal seal only after that readback has been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient filter, selected real, choice schedule, host equality, or ambient completeness coordinate, the handoff remains a finite BEDC route. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRootRegSeqRatRealHandoff`

The $\mathsf{Cauchy}^{\uparrow}$ packet enters the L10 dependency lattice through its sibling route before completion or filter consumers inspect it. Every such consumer first reads the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The Cauchy route is therefore a displayed sibling dependency path, not a floating completion primitive or a host-level filter object.

Theorem 5888.11 (Cauchy L10 sibling dependency route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer in the L10 lattice receives finite observation windows from chapter 126, regular-rational readback from chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any Real-facing approximation is public. The route exports no selected limit, quotient equality, host filter object, or ambient completeness principle.*

Proof. Project the accepted Cauchy packet through its finite observation surface. The first visible rows are the $\mathsf{StreamName}^{\uparrow}$ windows, so downstream readers cannot begin from a host-level limit predicate.

Next read the regular-rational sibling. The $\mathsf{RegSeqRat}^{\uparrow}$ row supplies the finite regularity readback used by Cauchy consumers before the packet reaches the Real-facing boundary.

Finally expose the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed windows and readback rows, the route supplies finite windows and regular-rational data before public approximation, and it leaves no coordinate for selected limits, quotient equality, host filters, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.CauchyUp.L10SiblingDependency.CauchyL10SiblingDependencyRoute`

Theorem 5888.12 (Cauchy route RegSeqRat tail inversion)**.** *Every accepted* $\mathsf{Cauchy}^{\uparrow}$ *consumer that exposes a* $\mathsf{Real}^{\uparrow}$*-facing approximation through the displayed L10 sibling route must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational tail row and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. If either row is absent, the consumer has no public* $\mathsf{Real}^{\uparrow}$*-facing approximation route through the* $\mathsf{Cauchy}^{\uparrow}$ *L10 lattice.*

Proof. Start with Definition 5888.11. A public approximation route is visible only after the accepted Cauchy packet has passed through finite $\mathsf{StreamName}^{\uparrow}$ windows, regular-rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Read the windows first. They determine which finite observations are available to the consumer, but they do not by themselves supply a regular rational tail or a tolerance account.

Next inspect the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ rows. The regular-rational tail supplies the readback values, while the dyadic ledger supplies the tolerance discipline needed before those values can be read as a Real-facing approximation.

Finally apply the $\mathsf{Real}^{\uparrow}$ seal. Since the seal is reached only after the displayed tail and tolerance rows, deleting either row leaves no public route from the Cauchy consumer to a Real-facing approximation. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRouteRegSeqRatTailInversion`

Theorem 5888.13 (Cauchy RegSeqRat–Real completion lattice)**.** *Every completion-facing* $\mathsf{Cauchy}^{\uparrow}$ *read factors through the displayed sibling lattice*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before any completion consumer may read a limit row. The StreamName windows are the observation boundary of chapter 126; the regular rational tails are the readback boundary of chapter 301; the metric-completion request is the comparison boundary of chapter 1506; and the terminal seal is the Real boundary of chapter 38. The route exports no quotient stream, selected representative, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyRegSeqRatRealCompletionLattice`

Proof. Read the hub through its displayed child routes. The finite-window dependency of Definition 3688.8 and the RegSeqRat–StreamName dependency lattice already force every Real-facing Cauchy read to expose $\mathsf{StreamName}^{\uparrow}$ windows before it consumes regular rational readback rows.

Next apply the regular-source completion route of Definition 3688.11. The $\mathsf{MetricCompletion}^{\uparrow}$ boundary receives the same displayed $\mathsf{RegSeqRat}^{\uparrow}$ tail budget, so a completion consumer cannot jump directly from a Cauchy name to a limit row.

Finally use the metric-completion forward lattice. It places the terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the metric-completion comparison, with no quotient stream, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate. Thus the public completion read is exactly the four-surface lattice displayed in the statement. □

Theorem 5888.14 (Cauchy public L10 export)**.** *The public* $\mathsf{Cauchy}^{\uparrow}$ *export for L10 consumers is exactly the finite route already established by Definition 3413.1, Definition 5888.16, Definition 5888.18, and Definition 5888.10:* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *route. A public consumer receives no quotient stream equality, selected limit, countable-choice schedule, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyPublicL10Export`

Proof. Start with the carrier boundary. Definition 5888.16 exhausts the accepted Cauchy packet by the Cauchy source row, finite windows, dyadic tolerances, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Use the scoped filter-modulus route. Definition 5888.18 shows that filter, modulus, Real-facing, and completion-facing consumers all read the same finite path rather than an external completion object.

Finally apply the root readback handoff of Definition 5888.10. It places the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal downstream of the same window and tolerance rows. Thus the public L10 export is the displayed route itself, with no additional host equality, selection, choice, quotient, or completeness principle. □

Theorem 5888.15 (Cauchy forward L10 sum-tail route)**.** *Every* $\mathsf{Cauchy}^{\uparrow}$ *consumer that reaches the regular sum tail-estimate surface must first pass through the regular rational source of chapter 301, the terminal real-name boundary of chapter 38, and the finite tail-estimate packet of chapter 7387. The route is an L10 dependency path through displayed StreamName, DyadicRatCore,*

RegSeqRat, RegularCauchySum, Real, BHist, hsame, Cont, Pkg, *and* NameCert *rows; it exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the $\mathsf{Cauchy}^{\uparrow}$ hub through its child routes. The core carrier exposes finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, and the terminal Real-facing seal before any downstream completion or arithmetic consumer is available.

The regular sum tail-estimate chapter consumes exactly that surface twice, once for each regular source row. Its tail-estimate ledger then adds the RegularCauchySum row and finite error budget without creating a detached limit object.

Thus the forward L10 route from the Cauchy hub to the sum-tail packet is the displayed dependency path through RegSeqRat, Real, and the finite tail-estimate surface. Since each row is already a finite NameCert coordinate, no excluded host or choice principle can be read from the hub. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyBaseForwardL10Route`

Theorem 5888.16 (Cauchy seed carrier boundary)**.** *The* $\mathsf{Cauchy}^{\uparrow}$ *seed carrier boundary is exhausted by the displayed Cauchy source row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal-facing row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. It contains no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the carrier through Definition 3413.1. The displayed Cauchy packet provides only the source row, finite observation windows, dyadic tolerances, regular readback, the Real-facing seal route, and the structural transport, replay, provenance, and naming rows.

The sibling routes of Definition 3413.4 and Definition 3413.9 keep those rows ordered before any Real-facing consumer reads the packet. Since none of the displayed coordinates is a quotient completion, selected limit, choice schedule, host equality, or ambient completeness proof, the seed carrier boundary exports none of them. □

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedCarrierBoundary`

Theorem 5888.17 (Cauchy seed NameCert obligation)**.** *For every accepted* $\mathsf{Cauchy}^{\uparrow}$ *seed packet, the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *obligation surface records exactly carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, componentwise transport, continuation replay, provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchySeedNameCertObligation`

Proof. Project the Cauchy packet through the carrier boundary of Definition 5888.16. Every admissible NameCert read is a read of one displayed carrier coordinate or one structural row attached to that coordinate.

Carrier admission supplies the Cauchy source row. Window, tolerance, and readback exposure supply the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows. The terminal route supplies the Real-facing seal, and the remaining rows supply hsame transport, Cont replay, Pkg provenance, and local naming. No further local obligation is available inside the seed packet. □

Theorem 5888.18 (Cauchy filter-modulus scoped route)**.** *Every public* $\mathsf{Cauchy}^{\uparrow}$ *seed read used by a filter, modulus, Real-facing, or completion-facing consumer factors through the displayed finite route consisting of* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row,* BHist *carrier data, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$ *row. The route exports no quotient stream equality, selected limit, countable choice, trichotomy, host real equality, quotient completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyUp.CauchyFilterModulusScopedRoute`

Proof. Start from the Cauchy seed boundary of Definition 5888.16. It exhausts the carrier by the Cauchy source row, finite stream windows, dyadic tolerance rows, regular readback, Real-facing seal, transport, replay, provenance, and local naming.

Apply Definition 5888.17. It identifies those same rows as the local NameCert obligation surface, so any filter or modulus consumer reads the seed packet through finite windows, dyadic budgets, regular-rational data, and the Real-facing seal rather than through an external completion object.

The sibling routes of Definition 1981.22, Definition 3688.9, and Definition 3991.15 keep the same packet ordered through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before completion-facing consumers inspect it. Since BHist, hsame, Cont, Pkg, and NameCert only carry, transport, replay, source, and name that route, the excluded quotient, selection, choice, trichotomy, host-equality, and ambient-completeness objects remain outside the scoped boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{Cauchy}^{\uparrow}$ seed packet is generated from finite stream windows, DyadicRatCore tolerance rows, RegSeqRat readback, a terminal Real-facing seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, maturePackageClosed)

Scope closed: Definition 5888.16, Definition 5888.17, Definition 5888.18, and the bridge-facing L10 export of Definition 5888.14 bind the public $\mathsf{Cauchy}^{\uparrow}$ route to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Cauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, dyadic tolerance exposure, regular-rational readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.CauchyUp.

Chapter 5889

A Concrete Naming Certificate for CauchyCompletionEmbeddingReflection$^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on uniform embeddings, the CauchyCompletionEmbeddingReflection$^{\uparrow}$ packet records the finite route by which a dense completion embedding is reflected through the separated completion interface. It sits over the dense completion embedding of chapter 5893, the dense uniform embedding surface of chapter 1651, the functorial completion route of chapter 1937, the separated reflection boundary of chapter 3543, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real seal of chapter 38. The packet names the completion-embedding reflection route only; it does not select representatives, form a quotient completion outside the displayed certificate, or import ambient completeness.

Definition 5889.1 (CauchyCompletionEmbeddingReflection carrier)**.** *A* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, R, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the source approximation row whose metric or uniform histories feed the completion embedding.*
2. *G is the* CauchyCompletionDenseEmbedding$^{\uparrow}$ *row naming the dense completion embedding and its image-window requests.*
3. *K is the* DenseUniformEmbedding$^{\uparrow}$ *row that exposes uniform density before the completion target is read as separated.*
4. *R is the* CauchyCompletionFunctoriality$^{\uparrow}$ *row transporting maps and comparison windows through completion.*
5. *S is the* SeparatedCompletionReflection$^{\uparrow}$ *row that reflects the target route through the separated completion boundary.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row attached to the same finite approximation windows.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row reached after A, G, K, R, S, D have been displayed.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row exposed only after the dense embedding, separated reflection, dyadic tolerance, and regular-rational readback rows are fixed.*

9. *H records componentwise* hsame *transport for the analytic rows and the structural rows.*
10. *C records displayed* Cont *replay of the same embedding-reflection route.*
11. *P is the* Pkg *provenance over* CauchyCompletionEmbeddingReflection$^{\uparrow}$, CauchyCompletionDenseEmbedding$^{\uparrow}$, DenseUniformEmbedding$^{\uparrow}$, CauchyCompletionFunctoriality$^{\uparrow}$, SeparatedCompletionReflection$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected representatives, countable choice, a detached separated completion, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.TasteGate.CauchyCompletionEmbeddingReflectio
single carrier alignment

Theorem 5889.2 (CauchyCompletionEmbeddingReflection NameCert obligations)**.** *For every accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *carrier* $J = (A, G, K, R, S, D, Q, E, H, C, P, N)$*, the local* NameCert$_{\text{CauchyCompletionEmbeddingReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed approximation row, dense completion embedding row, dense uniform embedding row, completion functoriality row, separated reflection row, dyadic tolerance row, regular-rational readback row, terminal real seal row, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. Every completion-facing read must pass through* $A, G, K, R, S, D, Q, E, H, C, P, N$ *in that order, and no read may bypass the separated reflection boundary.*

Proof. Project the carrier of Definition 5889.1. Its analytic rows are A, G, K, R, S, D, Q, E, followed by the structural rows H, C, P, N.

The dense route begins with A and G. The approximation row supplies the finite source histories, while the CauchyCompletionDenseEmbedding$^{\uparrow}$ row records the image-window requests through which completion consumers may inspect the dense embedding.

The rows K, R, S then bind the same route to dense uniform embedding, functorial completion transport, and separated reflection. A completion-facing consumer therefore reaches the separated target only after the dense route and functorial route have been displayed.

The rows D, Q, E provide dyadic tolerance, regular-rational readback, and the terminal real seal. Finally H, C, P, N transport, replay, source, and name the same packet, so carrier exhaustion blocks every excluded quotient, selector, detached completion, and ambient-completeness route. □

Theorem 5889.3 (CauchyCompletionEmbeddingReflection handoff)**.** *Every downstream consumer of an accepted* CauchyCompletionEmbeddingReflection$^{\uparrow}$ *packet receives the dense completion embedding only through the finite handoff*

$$\text{CauchyCompletionDenseEmbedding}^{\uparrow} \longmapsto \text{DenseUniformEmbedding}^{\uparrow} \longmapsto \text{CauchyCompletionFunctoriality}^{\uparrow} \longmapsto \text{Separat}$$

Thus the consumer obtains a reflected separated-completion route with finite stream windows, dyadic tolerance, regular-rational readback, and a terminal real seal; it does not obtain a quotient-completion representative, a selected dense point, countable choice, or ambient completeness.

Proof. Start with Definition 5889.2. The NameCert surface forces every public read to begin with the approximation row and the dense completion embedding row.

Read the sibling chapters named in the statement along the displayed order. The DenseUniformEmbedding$^{\uparrow}$ row supplies the uniform dense image, CauchyCompletionFunctoriality$^{\uparrow}$ transports the route through

completion, and $\mathsf{SeparatedCompletionReflection}^{\uparrow}$ is the only separated target boundary available to the packet.

The finite observation route is then exposed through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. These rows are exactly the rows D, Q, E together with the stream-window data carried by A, G, K, R, S.

Since the carrier has only $A, G, K, R, S, D, Q, E, H, C, P, N$, the structural rows can preserve and name the handoff but cannot supply a quotient representative, selected dense point, countable-choice schedule, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5889.1, Definition 5889.2, and Definition 5889.3 seed the dense completion embedding reflection route through separated completion, finite stream windows, dyadic tolerance, regular-rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected representatives, countable choice, detached separated completions, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense completion embedding exposure, dense uniform embedding compatibility, completion functoriality, separated reflection, finite-window readback, dyadic tolerance, regular-rational handoff, real sealing, and nonescape for BEDC.Derived.CauchyCompletionEmbeddingReflectionUp.

Constructive story: A $\mathsf{CauchyCompletionEmbeddingReflection}^{\uparrow}$ packet is generated from a source approximation row, a dense completion embedding row, a dense uniform embedding row, a completion functoriality row, a separated reflection row, dyadic tolerance, regular-rational readback, a terminal real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5890

A Concrete Naming Certificate for $\mathsf{SnakeLemma}^{\uparrow}$

Following Cartan–Eilenberg and Weibel homological algebra, the $\mathsf{SnakeLemma}^{\uparrow}$ packet records the finite diagram chase that turns two exact rows and compatible vertical maps into the connecting boundary route. It sits over the abelian-category carrier of chapter 176, the homology carrier of chapter 98, the spectral-sequence consumer surface that exposes $\mathsf{DerivedFunctor}^{\uparrow}$ rows in chapter 178, and the derived-functor interface of chapter 177. The packet names a displayed exactness boundary; it does not import quotient-object equality, kernel-cokernel object extensionality, or an ambient abelian-category diagram oracle.

Definition 5890.1 (Snake lemma carrier)**.** *A* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row carrying the additive kernel-cokernel surface in which the diagram is read.*
2. *R_0 and R_1 are the upper and lower exact-row ledgers, each exposing three displayed object rows, two morphism rows, and the local image-kernel containment needed for exactness.*
3. *V is the vertical-compatibility row, recording the three displayed vertical morphisms and the two commutative squares that align R_0 with R_1.*
4. *K is the kernel-row ledger for the left and middle vertical maps.*
5. *C is the cokernel-row ledger for the middle and right vertical maps.*
6. *E is the exactness-boundary row, recording that the kernel leg, connecting boundary, and cokernel leg form the displayed exact boundary chain.*
7. *∂ is the connecting-boundary row from the right kernel surface to the left cokernel surface.*
8. *H records componentwise* hsame *transport for the abelian-category source, exact rows, vertical squares, kernel rows, cokernel rows, boundary rows, replay, provenance, and local naming.*
9. *T records displayed* Cont *replay from the two exact rows and vertical squares to $K, C, E, \partial, H, P, L$.*
10. *P is the* Pkg *provenance over* $\mathsf{SnakeLemma}^{\uparrow}$*,* $\mathsf{AbelianCat}^{\uparrow}$*,* $\mathsf{Homology}^{\uparrow}$*,* $\mathsf{DerivedFunctor}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$*.*

11. L is the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *row.*

The classifier compares accepted snake-lemma packets only through these rows. It contains no coordinate for host quotient representatives, unbounded diagram search, selected elements outside the displayed histories, or equality of ambient abelian-category objects.

Lean4 ✓ `BEDC.Derived.SnakeLemmaUp.TasteGate.SnakeLemmaTasteGate single carrier alignment`

Theorem 5890.2 (Snake lemma NameCert obligations)**.** *For every accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *carrier*

$$S = (A, R_0, R_1, V, K, C, E, \partial, H, T, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$ *surface consists exactly of abelian-category source exposure through A, upper and lower exact-row exposure through R_0, R_1, vertical-square compatibility through V, kernel-row exposure through K, cokernel-row exposure through C, exact-boundary exposure through E, connecting-boundary exposure through ∂, componentwise transport through H, displayed replay through T, package provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 5890.1. The rows A, R_0, R_1, V expose the ambient additive kernel-cokernel surface, the two exact rows, and the square compatibility before any boundary read is allowed.

The kernel and cokernel ledgers are then read from K and C. They supply the visible source and target surfaces for the connecting row ∂, while E records the exactness statement tying the kernel leg, boundary leg, and cokernel leg into one chain.

Transport, replay, provenance, and local naming preserve this same finite packet through H, T, P, L. Since no excluded quotient representative, host object equality, or unbounded diagram search appears among the coordinates, the listed rows exhaust the local certificate obligations. □

Theorem 5890.3 (Snake lemma exact boundary)**.** *Every downstream* $\mathsf{Homology}^{\uparrow}$ *or* $\mathsf{DerivedFunctor}^{\uparrow}$ *consumer of an accepted* $\mathsf{SnakeLemma}^{\uparrow}$ *packet reads the boundary route*

$$\ker(v_2) \longmapsto \operatorname{coker}(v_0)$$

only through the displayed chain

$$\mathsf{AbelianCat}^{\uparrow} \longmapsto (R_0, R_1, V) \longmapsto (K, C) \longmapsto (\partial, E) \longmapsto (H, T, P, L).$$

Thus the consumer receives one finite exact-boundary packet of abelian-category source rows, exact-row ledgers, vertical-square ledgers, kernel and cokernel rows, the connecting-boundary row, exactness-boundary evidence, transport, replay, provenance, and local naming.

Proof. Start with the abelian-category source row A. The two exact rows R_0, R_1 and the vertical-compatibility row V expose exactly the diagram data on which the classical snake-lemma chase depends.

Read K and C next. These rows identify the displayed kernel and cokernel surfaces that the connecting boundary is permitted to connect, without choosing representatives outside the finite histories.

The boundary row ∂ and exactness row E then supply the only route from the right kernel surface to the left cokernel surface. A homology or derived-functor consumer therefore reads the boundary as a named packet, not as an ambient quotient construction.

Finally H, T, P, L keep the same route stable under componentwise history equality, displayed replay, provenance, and local naming. No quotient-object equality, host diagram oracle, or unbounded element chase is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{SnakeLemma}^{\uparrow}$ packet is generated from an AbelianCat source row, two exact-row ledgers, vertical-square compatibility rows, kernel-row and cokernel-row ledgers, a connecting-boundary row, an exactness-boundary row, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5890.1, Definition 5890.2, and Definition 5890.3 seed the finite snake-lemma packet over abelian-category source rows, exact-row ledgers, vertical-square compatibility, kernel rows, cokernel rows, connecting-boundary rows, exactness-boundary evidence, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SnakeLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-object equality, host abelian-category equality, unbounded diagram search, or ambient element-chase completeness.

Upgrade path: Obligation closure requires checked carrier admission, exact-row exposure, vertical-square compatibility, kernel-row exposure, cokernel-row exposure, connecting-boundary construction, exactness-boundary preservation, replay stability, provenance exhaustion, and local naming for BEDC.Derived.SnakeLemmaUp.

Chapter 5891

A Concrete Naming Certificate for $\mathsf{RealTailDiameterGauge}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet records the finite gauge by which a real-facing consumer reads a regular rational tail only after the displayed tail-diameter, dyadic tolerance, stream-window, and readback rows have been exposed. It sits after the tail-diameter witness of chapter 1746, the dyadic tolerance core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a non-quotient L10 gauge object for Real and Completion consumers; it is not host real equality, quotient stream equality, selected limits, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5891.1 (Real tail-diameter gauge BHist source)**.** *A* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (E, T, D, W, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the real-facing boundary that the gauge is allowed to serve.*
2. *T is the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row, recording the inspected tail threshold, two visible tail indices, and the displayed diameter bound.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row consumed by the same tail-diameter ledger.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite source observations inspected by T.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same window and tolerance data before any real-facing read is accepted.*
6. *H records componentwise* hsame *transport for the seal, tail-diameter ledger, dyadic tolerance, window, readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a real-facing gauge request through E, T, D, W, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RealTailDiameterGauge}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailDiameter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E, T, D, W, R, H, C, P, N$. It has no coordinate for host real equality, quotient stream equality, selected limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.RealTailDiameterGaugeUp.TasteGate.RealTailDiameterGaugeTasteGate single carrier alignment`

Theorem 5891.2 (Real tail-diameter gauge NameCert obligation surface)**.** *For every accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *carrier*

$$G = (E, T, D, W, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, the* $\mathsf{CauchyTailDiameter}^{\uparrow}$ *ledger row* T*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, componentwise transport* H*, displayed replay* C*, package provenance* P*, and local naming* N*.*

Proof. Project the carrier of Definition 5891.1. The analytic rows are the Real seal E, tail-diameter ledger T, dyadic tolerance D, finite window W, and regular rational readback R.

The gauge obligation is read only through T, D, W, R. The tail-diameter ledger records the visible tail request and bound, the dyadic row supplies the tolerance budget, the stream row exposes the inspected finite windows, and the regular row repacks the same data for rational readback.

The structural rows H, C, P, N preserve that route under transport, replay, provenance, and naming. Since the carrier has no coordinate for the excluded host or completion objects, the displayed rows exhaust the local NameCert surface. □

Theorem 5891.3 (Real tail-diameter gauge RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{RealTailDiameterGauge}^{\uparrow}$ *packet factors through*

$$W, D \longmapsto T \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real and Completion route may read the finite stream window, dyadic tolerance ledger, tail-diameter gauge, regular rational handoff, terminal Real seal, transport, replay, provenance, and local naming rows, but it receives no host real equality, quotient stream equality, selected limit, trichotomy decision, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, Lean verification, or bridge check.*

Proof. Start with an exported regular-rational or real-facing read. The only finite observations available to the packet are the $\mathsf{StreamName}^{\uparrow}$ row W and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D.

The tail-diameter ledger T then binds those observations to a displayed tail request and diameter bound. This row is the gauge that prevents a consumer from replacing the finite tail evidence by a quotient stream or a selected limit.

Only after T is displayed does the handoff expose the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the $\mathsf{Real}^{\uparrow}$ seal row E. The readback and seal are therefore downstream from the same finite gauge data.

Finally H, C, P, N transport, replay, source, and name the route. Since the excluded objects do not occur in the carrier, no consumer can bypass the displayed handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$.

Constructive story: A $\mathsf{RealTailDiameterGauge}^{\uparrow}$ packet is generated from a RealUp seal row, a CauchyTailDiameterUp ledger, a DyadicRatCore tolerance row, a StreamName window row, a RegSeqRat readback row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, seed)

Scope closed: Definition 5891.1, Definition 5891.2, and Definition 5891.3 seed the finite real-facing tail-diameter gauge for regular rational and completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealTailDiameterGauge}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not close host real equality, quotient stream equality, selected limits, trichotomy, countable choice, Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked carrier admission, Real seal confinement, CauchyTailDiameter ledger exactness, DyadicRatCore tolerance readback, StreamName window coverage, RegSeqRat handoff, transport stability, continuation replay, provenance non-escape, and local naming for BEDC.Derived.RealTailDiameterGaugeUp.

This hub routes the EffectiveRealCompactIntervalSearch$^{\uparrow}$ packet through chapter 1250, chapter 1767, and chapter 5899 before the child files expose the compact interval search carrier.

Chapter 5892

A Concrete Naming Certificate for $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet records the finite search surface by which a continuous real-valued map on a displayed closed bounded interval is inspected through compact finite nets, uniform-modulus data, and interval bisection rows before a downstream extremum consumer reads a certified bracket. It sits over the real-name seal of chapter 38, the interval endpoint carrier of chapter 1591, the compact-net selector of chapter 914, the graph and pointwise rows of chapter 130, and the tolerance handoff of chapter 369. The packet gives the constructive extreme-value route a finite searchable carrier; it does not assert classical Heine–Borel compactness, choose a host maximum, or identify quotient real functions.

The search packet consumes the compact interval finite-net route before it exports bisection or extreme-value handoffs. Its selector row P is read downstream of the closed-bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; the bisection ledger then refines those displayed cells instead of introducing an open-cover compactness argument or a selected supremum.

Definition 5892.1 (Effective real compact-interval search carrier)**.** *An* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (I, F, U, P, B, V, H, C, Q, N)$$

with the following visible rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint and containment row whose endpoints are read through* $\mathsf{Real}^{\uparrow}$ *names.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for a real-valued map over the interval source named by I.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance handoff for F on the displayed interval source.*
4. *P is the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row that lists the rational mesh cells and selected centers used by the search.*
5. *B is the bisection ledger, a finite prefix tree of interval halves whose cells refine P at the requested tolerance.*
6. *V is the value-comparison ledger storing the displayed* $\mathsf{Real}^{\uparrow}$ *value rows and rational error budgets attached to the cells of B.*

7. *H stores componentwise* hsame *transport for the interval, continuous-map, uniform-modulus, selector, bisection, value-comparison, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an extremum-search request through $I, F, U, P, B, V, H, Q, N$.*
9. *Q is the* Pkg *provenance over* EffectiveRealCompactIntervalSearch$^{\uparrow}$, Real$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, ContinuousMap$^{\uparrow}$, UniformModulus$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *row.*

The classifier compares accepted search carriers only through $I, F, U, P, B, V, H, C, Q, N$. No open-cover compactness theorem, selected host maximum, excluded-middle comparison on all real values, quotient function equality, or ambient completeness coordinate is part of the carrier.

Theorem 5892.2 (Effective real compact-interval search obligations)**.** *For every accepted* EffectiveRealCompactInte *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, the local* NameCert$_{\text{EffectiveRealCompactIntervalSearch}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, value-comparison ledger, transport, replay, provenance, and local naming;*
2. *source agreement, requiring I, F, U, and P to name the same displayed interval source before any cell is searched;*
3. *selector exactness, requiring the search cells in B to refine the finite* ProbeBundle *centers and tolerances exposed by P;*
4. *value-row binding, requiring every value comparison in V to be attached to a cell of B, a displayed graph row of F, and the error budget supplied by U;*
5. *bisection stability, requiring interval refinement to preserve endpoint containment and the selected candidate cell under componentwise* hsame *transport;*
6. *extremum handoff, requiring any lower-bound, upper-bound, near-minimum, or near-maximum read to pass through B, V, H, C, Q, N rather than through a host maximum selector;*
7. *non-escape, excluding open-cover compactness, classical Heine–Borel selection, trichotomy on arbitrary real values, quotient function equality, and ambient* Real$^{\uparrow}$ *completeness from the local packet.*

No effective compact-interval search obligation is stored outside these rows.

Proof. Project the carrier of Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, and the finite selector row is P.

The search may begin only after these rows agree on the same displayed interval source. The bisection ledger B then refines the selector cells supplied by P, so each later comparison is attached to a visible interval cell rather than to an external point choice.

The value ledger V reads graph values from F with the error budget delivered by U. This makes every candidate comparison a finite statement about named cells and rational tolerances.

Finally the extremum-facing read uses H, C, Q, N to replay and name the same route. Since the carrier has no coordinate for open-cover compactness, a host maximum, global real trichotomy, quotient function equality, or ambient completeness, none of those objects can enter the local certificate. □

Theorem 5892.3 (Effective real compact-interval bisection induction)**.** *For an accepted* EffectiveRealCompactInterv *carrier $S = (I, F, U, P, B, V, H, C, Q, N)$, every finite bisection stage in B is obtained by induction from the endpoint row I and the selector row P. The induction step replaces one displayed interval cell by its two named halves, transports the corresponding* ContinuousMap$^{\uparrow}$ *and* UniformModulus$^{\uparrow}$ *reads through H, and extends V only by value rows attached to those halves.*

Proof. The base stage is the interval cell named by I together with the first finite-net read in P. Its value ledger is the corresponding displayed graph read from F with the tolerance supplied by U.

For the induction step, choose the cell already selected by the finite ledger. The midpoint and half-interval endpoints are read as interval rows below I, and the selector exactness obligation identifies their centers with the relevant ProbeBundle entries of P.

Transport along H keeps the continuous-map graph and uniform-modulus handoff on the same source. The new value rows are therefore added to V only for the two displayed halves, so the induction never leaves the finite bisection tree. □

Theorem 5892.4 (Effective real compact-interval extreme-value handoff)**.** *Every extremum-search consumer reading an accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *carrier factors through*

$$I, F, U, P \longmapsto B, V \longmapsto H, C, Q, N.$$

The consumer receives a finite bracket or near-extremal cell family whose error budget is supplied by UniformModulus$^{\uparrow}$ *and whose candidates are selected by* CompactNetModulusSelector$^{\uparrow}$*. It receives no classical maximum point, open-cover compactness proof, quotient equality of functions, global trichotomy on* Real$^{\uparrow}$ *values, or ambient completeness assertion.*

Proof. Begin with the carrier projection. The only source data available to the consumer are the interval row I, continuous graph row F, uniform-modulus row U, and compact-net selector row P.

Apply Definition 5892.3. Every searched cell and every candidate value is produced inside the finite bisection ledger B and value ledger V.

The final handoff reads those ledgers through H, C, Q, N. Applying Definition 5892.2, any route that asks for a host maximum, open-cover compactness, quotient function equality, global real trichotomy, or ambient completeness is outside the accepted packet. □

Theorem 5892.5 (Effective real compact interval search finite-net handoff)**.** *Every accepted* EffectiveRealCompactIntervalSearch$^{\uparrow}$ *search packet reads a compact interval finite-net certificate, a located interval bisection row, and the* Real$^{\uparrow}$ *comparison seal before exporting the extreme-value handoff. The route passes through the closed bounded interval net of chapter 1250 and the finite epsilon-net surface of chapter 1767; it exports no open-cover compactness, selected supremum, host trichotomy, quotient real equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The compact interval finite-net data enter through the selector row P, whose cells are read only after the displayed closed bounded interval source I is fixed.

Apply Definition 5892.3. The bisection row B refines the finite cells supplied by P, and every value comparison in V is attached to a displayed graph row of F with the tolerance budget from U.

Apply Definition 5892.4. The final route reads B, V, H, C, Q, N after the finite-net source rows, so the exported extreme-value handoff is a finite search certificate rather than an open-cover compactness, supremum-selection, trichotomy, quotient-equality, or completeness principle. □

Theorem 5892.6 (Effective real compact interval search L10 sibling dependency)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must display the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint carrier, the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row, the* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row, the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *finite net, the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *selector, the* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *row, and the* $\mathsf{LocatedIntervalBisection}^{\uparrow}$ *refinement before exporting any extremum handoff. The route is the finite-net and bisection dependency route of Definition 5892.5, not an isolated kernel-only search packet.*

Proof. Project the carrier definition, Definition 5892.1. The rows I, F, U, P, B, V expose the closed bounded interval source, continuous-map graph, uniform-modulus handoff, compact-net selector, bisection ledger, and value ledger, while the Real-facing seal is read through the displayed interval and value rows.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before any bisection or extreme-value consumer can read the packet.

The bisection row B then supplies the located refinement consumed by the extremum handoff. The structural rows H, C, Q, N only transport, replay, source, and name that same sibling-dependent route, so no extremum handoff is exported before the listed L10 siblings are displayed. □

Theorem 5892.7 (Effective real compact interval search inverse modulus bisection)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, a displayed* $\mathsf{UniformModulus}^{\uparrow}$ *tolerance row* U *and a* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *finite-net row* P *determine a finite bisection depth in* B *whose dyadic cell diameter is below the requested tolerance. The depth is read only through* P, B, V, H, C, Q, N*: the value ledger* V *attaches comparisons to cells at that depth,* H *transports the same cells under* hsame*,* C *replays the finite request, and* Q, N *package and name the route. The construction exports no open-cover compactness proof, no host maximum, no quotient equality of functions, no trichotomy on* $\mathsf{Real}^{\uparrow}$ *values, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the accepted search carrier. The uniform-modulus row U supplies the requested tolerance, while the selector row P displays the finite mesh data that the search is allowed to refine.

Apply Definition 5892.3. Each step of B replaces a displayed cell by named dyadic halves, so repeated refinement gives a finite depth at which the visible cell diameter is below the tolerance requested by U.

Apply Definition 5892.5. The finite-net row P remains the source of the cells refined by B, and the value ledger V records only comparisons attached to those cells.

Finally H, C, Q, N transport, replay, source, and name the same finite depth-control route. Because the carrier has no open-cover compactness coordinate, host maximum selector, quotient function equality coordinate, trichotomy oracle, or ambient completeness row, the inverse-modulus read remains finite interval-search data. □

Theorem 5892.8 (Effective real compact interval search finite-grid stability)**.** *For an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$*, the displayed finite-net cell selected by* P *and refined by the located bisection ledger* B *is preserved under componentwise* hsame *transport before any extreme-value handoff is exported. Thus every Real-facing extremum consumer reads the same finite cell, value row, and bisection row through* P, B, V, H, C, Q, N*, and receives no open-cover compactness proof, selected host maximum, or* $\mathsf{Real}^{\uparrow}$ *trichotomy.*

Proof. Project the carrier of Definition 5892.1. The finite-net cell is displayed by P, its bisection refinement is recorded in B, and the attached value comparisons are stored in V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search route through the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger is consumed.

Apply the inverse-modulus bisection theorem, Definition 5892.7. The bisection depth and the cell read remain tied to P, B, V, while H transports the same displayed rows and C replays the same finite request.

The carrier has no coordinate for open-cover compactness, host maxima, quotient function equality, Real trichotomy, or ambient completeness. The exported extreme-value handoff therefore preserves only the finite-grid cell and located bisection row named by the packet. □

Theorem 5892.9 (Effective real compact interval search Bishop cover link)**.** *Every accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *carrier* $S = (I, F, U, P, B, V, H, C, Q, N)$ *reads the Bishop interval-cover finite cover ledger of chapter 5124 before it exports any bisection or extreme-value handoff. The Bishop route supplies the finite interval-cover cells refined by* P *and* B*, while the local search packet preserves the continuous-map, uniform-modulus, value-comparison, transport, replay, provenance, and naming rows. The exported route contains no open-cover compactness theorem, selected supremum, host trichotomy, quotient equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The interval source is I, the continuous graph is F, the uniform tolerance handoff is U, the finite selector row is P, the bisection ledger is B, and the value ledger is V.

Apply the finite-net handoff theorem, Definition 5892.5. It forces the search packet to read the closed bounded interval net and finite epsilon-net surfaces before the bisection ledger can be consumed.

Apply the grid-stability theorem, Definition 5892.8. The selected finite cell is preserved through P, B, V, H, C, Q, N, so the Bishop interval-cover ledger supplies interval-cover cells for the same displayed route rather than a separate compactness principle.

The remaining rows H, C, Q, N transport, replay, source, and name the same finite cover-to-search handoff. Since the carrier has no row for open-cover compactness, selected suprema, host trichotomy, quotient function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, those principles cannot be exported by the bisection or extreme-value consumer. □

Theorem 5892.10 (Effective real compact interval search RegularCauchyGluing dependency)**.** *Every completion-facing extremum-search read of an accepted* $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ *packet must expose the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *route of chapter 5899 after the finite bisection and value ledger are displayed and before any* $\mathsf{Real}^{\uparrow}$*-facing seal is consumed. The dependency is read through the finite-net handoff of Definition 5892.5 and the grid-stability route of Definition 5892.8; it is not supplied by quotient stream equality, selected host limits, countable choice, or ambient real completeness.*

Proof. Project the carrier $S = (I, F, U, P, B, V, H, C, Q, N)$ from Definition 5892.1. The completion-facing rows available to the search are the bisection ledger B, the value-comparison ledger V, and the structural replay rows H, C, Q, N.

Apply Definition 5892.5. The finite-net route fixes the displayed interval cells and their Real-valued comparison windows before an extremum consumer can read the packet.

Apply Definition 5892.8. The same finite cell, bisection row, and value row are preserved under componentwise hsame transport, so the completion-facing read has a stable regular-Cauchy window to hand to the gluing sibling.

Route that window through $\mathsf{RegularCauchyGluing}^{\uparrow}$ before the terminal Real-facing seal is consumed. Since the carrier has no quotient stream-equality row, selected host limit, countable-choice

selector, or ambient-completeness coordinate, the gluing dependency is the only completion-facing route available after B and V are displayed. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5892.1, Definition 5892.2, Definition 5892.3, Definition 5892.4, Definition 5892.5, Definition 5892.7, and Definition 5892.8 seed the finite interval-search carrier from displayed Real, closed-bounded-interval, compact-net-selector, continuous-map, finite-net, and uniform-modulus rows to extremum-facing bracket and depth-control handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical Heine–Borel compactness, selected host maxima or minima, quotient equality of functions, global real trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source agreement, selector exactness, value-row binding, bisection stability, extremum handoff, and non-escape for BEDC.Derived.EffectiveRealCompactIntervalSearchUp.

Constructive story: An $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp source, ContinuousMapUp graph rows, a UniformModulusUp tolerance handoff, a CompactNetModulusSelectorUp finite-net row, a finite bisection ledger, displayed RealUp value comparisons, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ search packet consumes the closed-bounded interval net of chapter 1250, the finite epsilon-net surface of chapter 1767, the compact-net modulus selector of chapter 914, and the located interval bisection route of chapter 1711 before exporting its extremum handoff. Thus a downstream consumer sees the finite-net and bisection dependencies as displayed route data, not as an open-cover compactness proof or a selected host extremum. The $\mathsf{EffectiveRealCompactIntervalSearch}^{\uparrow}$ route hands its finite bisection rows, interval-value rows, and compact-window certificates to the compatible-window route of chapter 5899 before any $\mathsf{Real}^{\uparrow}$ seal consumer reads a glued local search window. This dependency keeps each L10 finite-window consumer downstream from the displayed compact search packet, so the gluing step reads certified local windows rather than an ambient completeness or choice principle.

Chapter 5893

A Concrete Naming Certificate for $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$

Following Bishop and Bridges on constructive completions and Bourbaki/Kelley on dense embeddings, the $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet records the finite dense-source route by which a Cauchy completion consumer reads a source object inside its completion boundary. It sits beside the dense uniform embedding surface of chapter 1651, the functorial Cauchy-completion route of chapter 1937, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies dense embedding data for completion consumers without asserting quotient completion equality, selected representatives, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5893.1 (Cauchy completion dense-embedding carrier)**.** *A* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (A, G, K, S, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the dense source row. It records the source-side Cauchy observations that a completion consumer may approximate by displayed source data.*
2. *G is the embedding graph row. It records the displayed source-to-completion graph and the finite image-window reads used by dense embedding consumers.*
3. *K is the Cauchy completion row, read through the completion-facing route associated with* $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$*.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row exposing the finite request windows inspected by the source and completion sides.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare source-image windows with completion windows.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reachable only after A, G, K, S, D, Q have been displayed.*
8. *H records componentwise* hsame *transport for the dense source, embedding graph, completion, window, tolerance, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dense-source request through $A, G, K, S, D, Q, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{DenseUniformEmbedding}^{\uparrow}$, $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, countable choice, host equality of completion classes, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionDenseEmbeddingUp.CauchyCompletionDenseEmbeddingTasteGate`
`single carrier alignment`

Theorem 5893.2 (Cauchy completion dense-embedding NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *carrier*

$$J = (A, G, K, S, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission for the dense source row, embedding graph row, Cauchy completion row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing dense read must expose A, G, K, S, D, Q before E, H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5893.1. The dense source, embedding graph, Cauchy completion row, finite windows, tolerance ledger, regular readback, and Real seal are exactly the displayed analytic coordinates.

The dense-source route is read through A, G, K. A completion consumer first fixes the source row, then the embedding graph row, and only then the completion-facing row associated with the Cauchy completion route.

The finite observation route is read through S, D, Q. The stream windows expose inspected requests, the dyadic ledger supplies the tolerance budget, and the regular rational readback repacks the same observations before any Real-facing seal is visible.

Finally E, H, C, P, N preserve and name the route. Since quotient completion equality, selected representatives, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the NameCert surface. □

Theorem 5893.3 (Cauchy completion dense-embedding Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion consumer exported by an accepted* $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ *packet factors through*

$$\mathsf{DenseUniformEmbedding}^{\uparrow},\ \mathsf{CauchyCompletionFunctoriality}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

More precisely, the consumer reads the dense source row A, embedding graph G, Cauchy completion row K, finite windows S, tolerance ledger D, regular readback Q, terminal seal E, componentwise transport H, displayed replay C, provenance P, and local name N before the real-facing boundary is exposed.

Proof. Start with an exported Real-facing or completion-facing read. The NameCert obligations of Definition 5893.2 force the read through A, G, K before the finite observation rows can be consumed.

Read the sibling routes next. The dense source is the $\mathsf{DenseUniformEmbedding}^{\uparrow}$ surface, the completed-map side is routed through $\mathsf{CauchyCompletionFunctoriality}^{\uparrow}$, and the finite observations are the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{RegSeqRat}^{\uparrow}$ rows displayed by the carrier.

Only after those rows have been displayed can the terminal $\mathsf{Real}^{\uparrow}$ seal row E be read. The structural rows H, C, P, N transport, replay, source, and name the same route, so the handoff exports no quotient completion equality, selected representative, countable-choice selector, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}$ packet is generated from a dense source row, embedding graph row, Cauchy completion row, StreamName windows, DyadicRatCore tolerance ledger, RegSeqRat readback, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5893.1, Definition 5893.2, and Definition 5893.3 seed the dense-source route for Cauchy completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionDenseEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, host equality of completion classes, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense source exposure, embedding graph exactness, Cauchy completion handoff, StreamName window coverage, DyadicRatCore tolerance readback, RegSeqRat handoff, Real seal confinement, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionDenseEmbeddingUp.

Chapter 5894

A Concrete Naming Certificate for $\mathsf{RiemannRefinementCriterion}^{\uparrow}$

Following Bishop and Bridges on Riemann–Darboux integration, the $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet records the finite refinement criterion that lets a tagged Riemann-sum consumer compare upper Darboux rows against a gauge-local partition route. It sits after the tagged partition carrier of chapter 4296, the upper Darboux-sum carrier of chapter 4083, the Riemann sum gauge packet of chapter 4115, and the Cousin cover carrier of chapter 3426. The packet supplies the finite refinement surface consumed by integrability criteria without importing host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5894.1 (Riemann refinement criterion carrier)**.** *A* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$$

with the following displayed rows:

1. *I is the closed bounded interval row shared by the refinement comparison.*
2. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *source row, including the finite cell order, tag containment, and mesh exactness visible before refinement is read.*
3. *U is the* ***??*** *row whose upper-sum ledger is compared against the refined tagged route.*
4. *G is the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row, carrying the positive rational gauge, mesh ledger, and finite tagged-sum handoff.*
5. *C is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge-cover row supplying the finite local cover route that admits the refined cells.*
6. *M is the common-refinement mesh row, listing the displayed refinement map from the tagged partition row into the gauge-cover and upper-sum rows.*
7. *D is the Darboux-gap lock row, recording the finite upper-bound comparison that remains visible after the common refinement has been read.*
8. *W is the window and readback row for interval endpoints, tags, gauges, cover cells, and upper-sum values.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the tagged partition, gauge, Cousin cover, refinement mesh, and Darboux-gap rows are displayed.*

10. *H records componentwise* hsame *transport for interval, partition, upper-sum, gauge, cover, refinement, Darboux-gap, window, seal, replay, provenance, and local naming rows.*
11. *K records displayed* Cont *replay from a refinement request through $I, T, U, G, C, M, D, W, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, *??*, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows. It has no coordinate for host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannRefinementCriterionUp.RiemannRefinementCriterionTasteGate single carrier alignment`

Theorem 5894.2 (Riemann refinement criterion NameCert obligations)**.** *For every accepted* RiemannRefinementCr *carrier $R = (I, T, U, G, C, M, D, W, E, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval, tagged partition, upper Darboux-sum, Riemann sum gauge, Cousin cover, refinement mesh, Darboux-gap lock, window, seal, transport, replay, provenance, and local naming rows.*

Every refinement-facing consumer must read the tagged partition row T from chapter 4296, the upper-sum row U from chapter 4083, the gauge row G from chapter 4115, and the gauge-cover row C from chapter 3426 before the common-refinement mesh row M and Darboux-gap lock row D are available.

Proof. Project the carrier of Definition 5894.1. The tuple displays the only rows available to a refinement consumer: interval source, tagged partition, upper-sum, gauge, Cousin cover, common refinement, Darboux-gap lock, finite windows, terminal seal, transport, replay, provenance, and local naming.

The source rows are T, U, G, C. The tagged partition supplies finite cells and tags, the upper Darboux row supplies the upper-bound ledger, the Riemann sum gauge row supplies the gauge mesh, and the Cousin cover row supplies local gauge cover cells.

The refinement rows are M, D. The mesh row M binds those four source rows into one common finite refinement before the Darboux-gap lock D can be read.

Finally W, E, H, K, P, N preserve the same displayed route through finite windows, the terminal Real seal, transport, replay, provenance, and local naming. Since the carrier has no coordinate for the refused host objects, the listed rows exhaust the local NameCert surface. □

Theorem 5894.3 (Riemann refinement criterion Darboux gap lock)**.** *In every accepted* RiemannRefinementCriterior *packet, the Darboux-gap row D is readable only after the common-refinement mesh row M has already displayed the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover sources. Thus a Darboux or Riemann-integrability consumer cannot replace the refinement route by host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the source rows from Definition 5894.1. The Darboux-gap row is not listed as an independent analytic conclusion; it appears after the tagged partition, upper-sum, gauge, cover, and common-refinement rows.

Apply Definition 5894.2. Its obligation surface requires T, U, G, C to be exposed before M, and M before D.

The structural rows only transport and replay the same finite packet. Therefore the Darboux-gap read is locked to the displayed refinement route and cannot be promoted to any refused host-level principle. □

Theorem 5894.4 (Riemann refinement criterion integral handoff)**.** *Every Riemann-integrability or gauge-integral consumer of an accepted* $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ *packet receives only the finite handoff*

$$\mathsf{TaggedRealPartition}^{\uparrow},\ \textit{??},\ \mathsf{RiemannSumGauge}^{\uparrow},\ \mathsf{CousinCover}^{\uparrow} \longmapsto M \longmapsto D \longmapsto W, E, H, K, P, N.$$

The handoff exports a refinement criterion over displayed finite rows, not an ambient theorem about all host partitions or all mesh limits.

Proof. Start with a consumer read. The packet first exposes the tagged partition, upper Darboux-sum, Riemann sum gauge, and Cousin cover rows through the sibling chapters cited in the carrier.

The common-refinement mesh row M then aligns those sources. The Darboux-gap lock row D is read only along that aligned route, and the window and seal rows W, E expose the Real-facing observations after the finite rows are visible.

Transport, replay, provenance, and local naming through H, K, P, N preserve the same route. No carrier coordinate supplies host partition equality, selected limiting meshes, choice, compactness, quotient function equality, or ambient completeness, so the integral handoff is exactly the displayed finite criterion. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannRefinementCriterion}^{\uparrow}$ packet is generated from a closed bounded interval row, TaggedRealPartition source, UpperDarbouxSum ledger, RiemannSumGauge mesh row, CousinCover gauge-cover row, common-refinement mesh row, Darboux-gap lock, finite windows, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5894.1, Definition 5894.2, Definition 5894.3, and Definition 5894.4 seed the finite refinement criterion between tagged partitions, upper Darboux sums, Riemann sum gauges, and Cousin covers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannRefinementCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host partitions, selected mesh limits, quotient function equality, choice, open-cover compactness, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tagged-partition source lock, upper-sum handoff, gauge mesh exactness, Cousin cover locality, Darboux-gap lock, integral handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannRefinementCriterionUp.

Chapter 5895

A Concrete Naming Certificate for $\mathsf{CompactUniformModulusComposition}^{\uparrow}$

Following the Bishop–Bridges and Heine–Cantor treatment of uniformly continuous maps on compact metric data, the $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ packet records the finite tolerance-only composition surface for two compact-uniform modulus rows. It sits after the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the compact uniform modulus row of chapter 2299, the metric-triangle estimate of chapter 1159, and the stability packet of chapter 5872. The packet supplies the finite composition object used by compact-open and compact-uniform-continuity consumers without exporting a host function space, quotient point equality, selected global modulus, choice schedule, or ambient compactness theorem.

Definition 5895.1 (Compact uniform modulus composition carrier)**.** *A* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$$

with the following displayed rows:

1. *X, Y, Z are compact metric source, middle, and target rows, with finite net and distance ledgers visible at the requested tolerances.*
2. *F and G are* $\mathsf{ContinuousMap}^{\uparrow}$ *rows for the two displayed maps.*
3. *M_F is the first* $\mathsf{CompactUniformModulus}^{\uparrow}$ *row, turning a target tolerance for Y into a source tolerance for X along F.*
4. *M_G is the second* $\mathsf{CompactUniformModulus}^{\uparrow}$ *row, turning a target tolerance for Z into a source tolerance for Y along G.*
5. *A is the tolerance-assembly row, recording that a Z-tolerance request is first read by M_G and that its resulting Y-tolerance request is then read by M_F.*
6. *T is the* $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$ *row that keeps the two transported estimates on the same metric route.*
7. *U is the downstream* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row for the composite map.*
8. *H records componentwise* hsame *transport for compact metric sources, continuous maps, modulus rows, tolerance assembly, triangle estimate, consumer, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a composite tolerance request through $Z, M_G, Y, M_F, X, A, T, U, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *row.*

The classifier compares accepted composition packets only by these finite rows. It contains no host function-space composition theorem, quotient point equality, selected global modulus, choice schedule, or ambient compactness principle.

Lean4 ✓ `BEDC.Derived.CompactUniformModulusCompositionUp.CompactUniformModulusCompositionTasteGate`
`single carrier alignment`

Theorem 5895.2 (Compact uniform modulus composition NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *carrier $Q = (X, Y, Z, F, G, M_F, M_G, A, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$ *surface consists exactly of compact source admission through X, Y, Z, continuous-map admission through F, G, first and second compact-uniform modulus reads through M_F, M_G, tolerance assembly through A, metric-triangle transport through T, uniform-modulus export through U, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5895.1. The compact metric rows, continuous-map rows, compact-uniform modulus rows, assembly row, triangle estimate, uniform-modulus consumer, and structural rows are exactly the displayed coordinates.

The tolerance route is ordered by A. A requested Z-tolerance is consumed by M_G to expose a Y-tolerance, and only that displayed Y-tolerance is then consumed by M_F to expose the source tolerance over X.

The triangle estimate T and consumer row U read the same assembled route. Transport, replay, provenance, and local naming preserve it through H, C, P, N.

Since the carrier has no coordinate for host function-space composition, quotient point equality, selected moduli, choice, or ambient compactness, the listed rows exhaust the NameCert surface. □

Theorem 5895.3 (Compact uniform modulus composition tolerance route)**.** *Every compact-open or compact-uniform-continuity consumer of an accepted* $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ *packet reads the composite modulus only through*

$$M_G \longmapsto A \longmapsto M_F \longmapsto T \longmapsto U.$$

Thus the composite tolerance is assembled from two displayed compact-uniform modulus rows and the metric-triangle ledger, not from a selected host-level modulus for the composite map.

Proof. Begin with the consumer's target tolerance for the Z row. The second modulus row M_G supplies the middle-space tolerance, and the assembly row A records that this is the only tolerance handed to M_F.

The first modulus row M_F returns the source-space tolerance over X. The metric-triangle row T then transports the two displayed estimates into the bound consumed by U.

The structural rows H, C, P, N preserve this route. Since there is no host composition coordinate or selected global modulus in the carrier, every consumer remains on the displayed tolerance chain. □

Theorem 5895.4 (Compact uniform modulus composition nonescape)**.** *No accepted* CompactUniformModulusComp *packet exports quotient point equality, a selected global modulus, a choice schedule, an ambient compactness theorem, or a host function-space composition proof. It exports only the two continuous-map rows, two compact-uniform modulus rows, the tolerance assembly row, the metric-triangle estimate, the uniform-modulus consumer, and the structural NameCert rows.*

Proof. Project the carrier. The only analytic rows are F, G, M_F, M_G, A, T, U over the compact metric sources X, Y, Z.

Apply Definition 5895.2. The local NameCert obligations require the consumer to read the two modulus rows and the tolerance assembly before the uniform-modulus export.

The refused host objects are absent from the carrier and cannot be produced by transport, replay, provenance, or local naming. Therefore the accepted packet exports only the displayed finite composition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformModulusComposition}^{\uparrow}$ packet is generated from compact metric source, middle, and target rows, two ContinuousMap rows, two CompactUniformModulus rows, a tolerance-assembly row, a MetricTriangleUniformEstimate row, a UniformModulus consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5895.1, Definition 5895.2, Definition 5895.3, and Definition 5895.4 seed the finite tolerance-only composition route for compact-uniform modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformModulusComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space composition, quotient point equality, selected global moduli, choice schedules, ambient compactness theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact source admission, continuous-map admission, first and second modulus reads, tolerance assembly, metric-triangle transport, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformModulusCompositionUp.

Chapter 5896

A Concrete Naming Certificate for $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$

The $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet realizes the private/shared decoration theorem of Definition 9462.1705 inside the boundary-tile source of subsection 9462.28.21. It sits beside the DNA boundary-tile packet of chapter 3393, the trigger-threshold packet of chapter 4217, and the finite trigger algebra route of chapter 3478. The packet records private blocks, shared rows, lift counts, and defect ledgers as finite NameCert data; it is not a biological mechanism, stochastic evolution model, hidden enumeration oracle, or quotient codon semantics.

Definition 5896.1 (PrivateSharedDecorationModel carrier)**.** *A* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, S, L, D, T, R, P, N)$$

with the following displayed rows:

1. *B is the nonempty private-block row, recording the partition of coordinates that remain private to a single decorated support.*
2. *S is the shared-row ledger, assigning every shared coordinate to a finite row set of size at least two.*
3. *L is the lift-count row, recording the finite ℓ^m lift factor attached to displayed support blocks.*
4. *D is the defect ledger, recording the row-set size contribution to the defect-refined polynomial.*
5. *T records componentwise* hsame *transport for B, S, L, D, R, P, N.*
6. *R records displayed* Cont *replay from a decorated trigger request through the private block, shared rows, lift count, defect ledger, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$, $\mathsf{DnaWindowSixBoundaryTile}^{\uparrow}$, $\mathsf{DNAWindowSixTriggerThreshold}^{\uparrow}$, $\mathsf{WindowCoverageTriggerAlgebra}^{\uparrow}$, *the source theorem of Definition 9462.1705,* BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, S, L, D, T, R, P, N. It has no coordinate for biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, or an unrecorded enumeration oracle.

Theorem 5896.2 (PrivateSharedDecorationModel NameCert obligations)**.** *For every accepted* $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ *carrier*

$$M = (B, S, L, D, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise transport, displayed replay, provenance, and local naming. A decorated trigger read is accepted only after it exposes a nonempty private block and assigns every shared coordinate to at least two rows.*

Proof. Project the carrier of Definition 5896.1. The private block, shared-row ledger, lift count, defect ledger, transport, replay, provenance, and local name are exactly the displayed coordinates.

The private/shared constraint is read through B, S. The row B must display a nonempty private partition, and the ledger S must attach every shared coordinate to a row set of size at least two.

The enumeration and defect data are read through L, D. The lift-count row records the finite lift factor, while the defect ledger records the shared-row contribution used by the source theorem.

Finally T, R, P, N transport, replay, source, and name the same route. Since biological mechanism, stochastic fitness, quotient codon equality, hidden global semantics, and unrecorded enumeration have no carrier coordinate, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: If a displayed decorated trigger lacks a nonempty private block or assigns a shared coordinate to fewer than two rows, it is rejected by the local NameCert.

Independence witness: dna-window-six-boundary-tile, dna-window-six-trigger-threshold, dna-window-six-translation-boundary: $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ is not bijective to any listed sibling because this packet jointly carries private-block rows, shared-row ledgers, lift-count rows, defect ledgers, transport, replay, provenance, and local naming.

The Lean TasteGate route for `BEDC.Derived.PrivateSharedDecorationModelUp` must provide explicit $\mathsf{Nontrivial}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{PrivateSharedDecorationModel}^{\uparrow}$ instances preserving B, S, L, D, T, R, P, N as separate displayed fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}$**.**

Constructive story: A $\mathsf{PrivateSharedDecorationModel}^{\uparrow}$ packet is generated from a nonempty private-block row, shared-row ledger, lift-count row, defect ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5896.1 and Definition 5896.2 seed the finite private/shared decoration packet over DNA boundary-tile trigger rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrivateSharedDecorationModel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close biological mechanism, stochastic evolution, quotient codon equality, hidden global semantics, unrestricted enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, private-block nonemptiness, shared-row arity, lift-count exposure, defect-ledger exposure, transport, replay, provenance, local naming, Nontrivial PrivateSharedDecorationModelUp, and FieldFaithful PrivateSharedDecorationModelUp for BEDC.Derived.PrivateSharedDecorationModelUp.

Chapter 5897

A Concrete Naming Certificate for $\mathsf{BishopLocatedRealFilter}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet records the finite filter-facing route by which located real brackets are read through dyadic tolerances, stream windows, regular rational readback, and the terminal real seal. It consumes the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the located bracket surface of chapter 1709, and the selector boundary of chapter 5886. The packet gives the L10 Real and Completion route a located filter interface without asserting quotient filter equality, host real equality, selected limit points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5897.1 (Bishop located real filter carrier)**.** *A* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier is a finite packet*

$$S = (F, L, D, W, R, E, B, H, T, P, N)$$

with the following displayed rows:

1. *F is the filter-base row, exposing the finite base request, refinement evidence, and Cauchy-facing filter data that a later completion consumer may inspect.*
2. *L is the located-real bracket row, recording lower and upper bracket reads over* $\mathsf{LocatedRealInterval}^{\uparrow}$ *together with the displayed apartness and containment data used to locate the requested real value.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing bracket width, filter-base comparison, stream-window radius, and regular-rational readback.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row attached to the filter request and the located bracket.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W under the tolerance ledger D.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after F, L, D, W, R have been displayed and replayed.*
7. *B is the* BHist *history row recording the finite route from the filter request through the located bracket, tolerance ledger, stream windows, regular readback, and Real seal.*
8. *H records componentwise* hsame *transport for $F, L, D, W, R, E, B, T, P, N$.*
9. *T records displayed* Cont *replay from every located filter request through $F, L, D, W, R, E, B, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{CauchyFilterSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, L, D, W, R, E, B, H, T, P, N$. It has no coordinate for quotient filter equality, host selected representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.BishopLocatedRealFilterUp.BishopLocatedRealFilterUp`

Theorem 5897.2 (Bishop located real filter NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *carrier $S = (F, L, D, W, R, E, B, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the filter base, located-real bracket, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, history, transport, replay, provenance, and local naming row;*
2. *bracket exactness, requiring each located-real bracket read in L to be witnessed by displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint and containment data;*
3. *filter-base stability, requiring refinements of F to preserve the same located bracket and tolerance ledger through* BHist *and* hsame *transport;*
4. *dyadic ledger stability, requiring D to control the bracket width, stream-window radius, and regular-rational readback without adding an external modulus;*
5. *StreamName and RegSeqRat handoff, requiring every numeric read to pass through W and then R before the terminal seal E can be consumed;*
6. *nonescape, excluding quotient filters, host-selected representatives, countable choice, host real equality, selected limiting points, and ambient Real completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 5897.1. The accepted packet displays the filter base, located bracket, dyadic ledger, stream windows, regular readback, Real seal, history, transport, replay, provenance, and local naming rows.

Bracket exactness is row-local. The bracket row L can expose only the endpoint and containment data already carried by $\mathsf{LocatedRealInterval}^{\uparrow}$, so it cannot identify a real by host order equality or by a selected representative.

The filter-base and tolerance rows then fix the route. Refinement of F is transported through B and H, while the same dyadic row D governs bracket width, window radius, and readback precision.

Finally the numeric handoff is ordered by W, R, E. Stream windows are read before regular rational data, and regular rational data are read before the terminal $\mathsf{Real}^{\uparrow}$ seal. Since no excluded host object is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5897.3 (Bishop located real filter L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ *packet $S = (F, L, D, W, R, E, B, H, T, P, N)$ must read the finite route*

$$F, L, D, W, R \longmapsto E \longmapsto B, H, T, P, N$$

before it treats the request as Real-facing or completion-facing data. Thus the packet supplies a filter-based located-real handoff to $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, *and*

$\mathsf{CauchyFilterSelector}^{\uparrow}$ *consumers without asserting ambient real completeness or quotient equality of filters.*

Proof. Read the carrier rows in order. A consumer first receives the filter base F and the located bracket L, while the dyadic ledger D supplies the common tolerance budget.

The numerical content is then exposed through W and R. The $\mathsf{StreamName}^{\uparrow}$ row records finite observation windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row records the regular rational readback extracted from those windows.

Only after these rows are displayed can the Real seal E be consumed. The history, transport, replay, provenance, and local naming rows B, H, T, P, N preserve that order for later L10 Real and Completion routes.

The carrier has no quotient-filter equality, selected representative, host real equality, choice schedule, or ambient completeness coordinate. Therefore every downstream handoff factors through the displayed finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedRealFilter}^{\uparrow}$ packet is generated from a displayed filter base, a located-real bracket, a dyadic tolerance ledger, finite StreamName windows, RegSeqRat readback, a Real seal, BHist history, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5897.1, Definition 5897.2, and Definition 5897.3 seed the finite filter-facing located-real interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedRealFilter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, ambient Real completeness, selected limits, countable choice, Lean verification, or bridge checking.

Upgrade path: Lean closure requires carrier admission, bracket exactness, filter-base stability, dyadic ledger stability, StreamName and RegSeqRat handoff, and Real seal boundary for BEDC.Derived.BishopLocatedRealFilterUp.

Chapter 5898

A Concrete Naming Certificate for $\mathsf{TaylorTheorem}^{\uparrow}$

Following Taylor and the validated-numerics treatment of finite Taylor models, the $\mathsf{TaylorTheorem}^{\uparrow}$ packet records the finite theorem-facing route from displayed derivative data to a polynomial jet and a checked remainder envelope. It sits after the Taylor-model enclosure surface of chapter 442, the derivative observation row of chapter 122, the polynomial carrier of chapter 50, the dyadic rational arithmetic core of chapter 291, the regular-rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite Taylor theorem certificate for one displayed center, order, domain window, derivative ledger, polynomial jet, and remainder bound; it does not import analytic convergence, quotient real equality, selected limits, or a host Taylor series.

Definition 5898.1 (Taylor theorem carrier)**.** *A* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, J, P, R, E, Q, H, C, G, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row naming the center, finite order, domain window, and derivative observations inspected by the packet.*
2. *J is the finite jet ledger. Each coefficient is read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic and the same derivative window named by D.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *row assembled from the displayed finite jet ledger.*
4. *R is the finite remainder-envelope row. It records the derivative-order budget, dyadic interval or ball enclosure, and endpoint window used to bound the residual after P is read.*
5. *E is the* $\mathsf{TaylorModel}^{\uparrow}$ *enclosure row that packages the displayed polynomial part and remainder envelope as a validated finite jet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback route for polynomial values, derivative observations, and the remainder bound.*
7. *H records componentwise* hsame *transport for derivative, jet, polynomial, remainder, Taylor-model, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every Taylor-theorem read through $D, J, P, R, E, Q, H, G, N$.*

9. *G is the* Pkg *provenance over* $\mathsf{TaylorTheorem}^{\uparrow}$, $\mathsf{TaylorModel}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, P, R, E, Q, H, C, G, N$. It has no coordinate for an infinite Taylor series, analytic convergence, quotient equality of real names, selected limiting values, or an ambient smooth-function theorem.

Theorem 5898.2 (Taylor theorem NameCert obligations)**.** *For every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier*

$$T = (D, J, P, R, E, Q, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed derivative row, finite jet ledger, polynomial row, remainder envelope, Taylor-model enclosure, readback route, transport, replay, provenance, and local naming rows;*
2. *derivative-to-jet routing, requiring every coefficient in J to be read from the finite* $\mathsf{Derivative}^{\uparrow}$ *observations in D and through* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic;*
3. *polynomial assembly, requiring P to use exactly the displayed finite coefficient rows in J;*
4. *remainder-envelope exposure, requiring R to bind the next derivative-order budget, the domain window, and the dyadic enclosure before any error bound is exported;*
5. *Taylor-model handoff, requiring E to package only the displayed P and R rows;*
6. *regular and real readback, requiring Q to pass through* $\mathsf{RegSeqRat}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible;*
7. *non-escape from analytic convergence, quotient real equality, selected limits, and host Taylor series semantics.*

No $\mathsf{TaylorTheorem}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.TaylorTheoremUp.TasteGate.TaylorTheoremTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 5898.1. The theorem-facing data are exactly D, J, P, R, E, Q, and the structural rows are exactly H, C, G, N.

The derivative-to-jet route is finite. Each coefficient read is a displayed entry of J, and the definition of J attaches that entry to the derivative row D and to dyadic rational arithmetic through $\mathsf{DyadicRatCore}^{\uparrow}$.

Polynomial assembly and remainder exposure are then separate rows. The polynomial row P consumes only the coefficients in J, while the remainder row R consumes the derivative-order budget and domain window before any bound is exported.

Finally the Taylor-model and readback rows seal the route. The row E packages the polynomial and remainder as a finite Taylor-model enclosure, and Q sends the same displayed data through $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$. Since no carrier coordinate names a host Taylor series or a selected limit, the listed rows exhaust the local NameCert surface. □

Theorem 5898.3 (Taylor theorem finite remainder envelope)**.** *Every accepted* $\mathsf{TaylorTheorem}^{\uparrow}$ *carrier exports a remainder claim only through the displayed route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Poly}^{\uparrow} \longmapsto \mathsf{TaylorModel}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More precisely, a consumer must first read the derivative row D, finite jet ledger J, polynomial row P, remainder-envelope row R, Taylor-model enclosure E, and regular-rational and real readback row Q before it may cite an error bound for the displayed polynomial approximation. The exported envelope is not an analytic convergence theorem, a quotient equality of real functions, a selected limit, or an ambient Taylor-series object.

Proof. Start with an accepted carrier $T = (D, J, P, R, E, Q, H, C, G, N)$. The only source of derivative information is D, and the only coefficient ledger is the finite row J read through $\mathsf{DyadicRatCore}^{\uparrow}$.

Next assemble the approximation. The polynomial row P is built from J, and the remainder row R binds the derivative-order budget, domain window, and dyadic enclosure that measure the residual after that polynomial row has been fixed.

Then pass through the enclosure and readback rows. The Taylor-model row E packages the polynomial and remainder as one finite validated jet, while Q routes the resulting value and bound through $\mathsf{RegSeqRat}^{\uparrow}$ and the terminal $\mathsf{Real}^{\uparrow}$ seal.

The transport, replay, provenance, and local naming rows preserve exactly this route. Because the carrier contains no coordinate for analytic convergence, quotient function equality, selected limits, or host Taylor series semantics, every exported remainder envelope is exhausted by the displayed finite rows. □

Falsifiable prediction: Every accepted $\mathsf{TaylorTheorem}^{\uparrow}$ remainder-envelope read factors through a DerivativeUp row, a DyadicRatCore coefficient ledger, a PolynomialUp row, a TaylorModelUp enclosure, a RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: taylormodel, derivative, polynomial, dyadicratcore, regseqrat, real: $\mathsf{TaylorTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds finite derivative observations, coefficient arithmetic, polynomial assembly, a theorem-facing remainder envelope, Taylor-model enclosure, regular-rational readback, real sealing, transport, replay, provenance, and local naming in one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{TaylorTheorem}^{\uparrow}$ packet is generated from a DerivativeUp row, a finite DyadicRatCore coefficient ledger, a PolynomialUp row, a finite remainder envelope, a TaylorModelUp enclosure, RegSeqRatUp and RealUp readback, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5898.1, Definition 5898.2, and Definition 5898.3 seed the finite Taylor theorem route from derivative observations through polynomial jets to displayed remainder envelopes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TaylorTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close analytic convergence, quotient equality of real names, selected limits, host Taylor series semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-to-jet routing, polynomial assembly, finite remainder-envelope exposure, Taylor-model handoff, regular and real readback, and non-escape for BEDC.Derived.TaylorTheoremUp.

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance rows of chapter 291, and the terminal Real seal of chapter 38 before routing the child files below.

Chapter 5899

A Concrete Naming Certificate for $\mathsf{RegularCauchyGluing}^{\uparrow}$

Bishop–Bridges style regular Cauchy gluing records the finite route by which two compatible local regular rational windows can be read as one BEDC real-name boundary without passing through quotient representatives or an ambient completeness theorem. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet sits after the dyadic tolerance core of chapter 291, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the reindexing route of chapter 1568.

Definition 5899.1 (Regular Cauchy gluing carrier)**.** *A* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (L, R, D, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *L and R are the left and right compatible regular Cauchy windows, each carrying finite* $\mathsf{StreamName}^{\uparrow}$ *observations and a local* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row.*
2. *D is the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger recording the shared dyadic radii, endpoint comparisons, overlap budgets, and refinement witnesses used by both windows.*
3. *S is the joint* $\mathsf{StreamName}^{\uparrow}$ *observation row, exposing exactly the finite indices inspected on the left window, right window, and overlap.*
4. *Q is the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by reading the two local regular rows through the common ledger D.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after L, R, D, S, Q have been displayed.*
6. *H records componentwise* hsame *transport for the two windows, common dyadic ledger, finite observations, glued readback, real seal, replay, provenance, and local certificate.*
7. *C records displayed* Cont *replay of the same finite gluing route.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that the two local windows are compatible on the displayed overlap through D, S, Q, H, C, P, N. It has no coordinate for quotient sequence equality, selected representatives, host real equality, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 5899.2 (Regular Cauchy gluing classifier)**.** *The* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *classifier compares accepted carriers only through the tuple* $L, R, D, S, Q, E, H, C, P, N$*. Two packets are indistinguishable for this classifier precisely when the left and right window rows, the common dyadic refinement ledger, the finite observation row, the glued regular readback, the terminal real seal, and the transport, replay, provenance, and naming rows are componentwise the same under the displayed* hsame *routes.*

Theorem 5899.3 (Regular Cauchy gluing NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *surface consists exactly of carrier admission for the two compatible regular Cauchy windows, common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement through* D*, finite* $\mathsf{StreamName}^{\uparrow}$ *observation through* S*, glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* Q*, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The analytic rows before the seal are exactly the two regular windows L, R, the common dyadic ledger D, the finite observation row S, and the glued regular readback Q.

The overlap obligation is finite. Its comparisons are stored in D and read through the same inspected indices in S, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ rows feed one glued readback row Q without adding a quotient representative.

Transport, replay, provenance, and local naming are supplied by H, C, P, N. Repacking these projections gives the listed $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ surface, and the carrier definition leaves no hidden row for selected representatives, host equality, countable choice, or ambient completeness. □

Theorem 5899.4 (Regular Cauchy gluing overlap stability)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Any refinement of the displayed overlap that is transported by* H *and replayed by* C *preserves the same glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *and the same* $\mathsf{Real}^{\uparrow}$ *seal row* E*. Consequently downstream Real and Completion consumers receive only the finite gluing packet and not quotient stream equality, selected representatives, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the displayed overlap ledger. The common dyadic row D records every radius and endpoint comparison used by both local windows, while S records the finite observations on which those comparisons are made.

Apply the classifier of Definition 5899.2. A refinement transported by H and replayed by C can change neither the componentwise ledger coordinates nor the finite observation coordinates exposed to the classifier.

The glued readback row Q is therefore the same row after refinement, and the $\mathsf{Real}^{\uparrow}$ seal E is read only through Q together with H, C, P, N. Since the carrier contains no quotient, selector, host-equality, choice, or completeness coordinate, none of those data can be exported by overlap stability. □

Theorem 5899.5 (Regular Cauchy gluing common-refinement exactness)**.** *Let* $G = (L, R, D, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. Every glued readback exported from* G *must expose the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement row* D *before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q *and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*. The exactness route is transported by* H *and replayed by* C*, and it exports no quotient stream equality, selected representative, host real equality, countable choice, or ambient completeness.*

Proof. Project the accepted carrier. The row D is the common dyadic refinement ledger shared by the two local regular Cauchy windows, while Q is the glued regular readback and E is the terminal Real seal.

Apply overlap stability from Definition 5899.4. Any accepted refinement of the overlap preserves the same Q and E only through the displayed ledger and observation rows, so a glued readback cannot bypass D.

Finally use the Cauchy-reindexing handoff of Definition 5899.7. It reads D, S, Q as the finite reindexing-compatible route before any Real-facing consumer reaches E, with H, C, P, N preserving and naming the same route. □

Theorem 5899.6 (Regular Cauchy gluing overlap induction)**.** *Let $G = (L, R, D, S, Q, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier. For any finite chain of displayed overlap refinements transported by H and replayed by C, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q and the terminal* $\mathsf{Real}^{\uparrow}$ *seal E remain unchanged. The resulting route still factors through* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *and* $\mathsf{CauchyReindexing}^{\uparrow}$, *and it exports no quotient stream equality, selected representatives, host equality, countable choice, or ambient completeness.*

Proof. Induct on the displayed finite refinement list.

The empty list is the overlap-stability theorem of Definition 5899.4: the common dyadic ledger D and the finite observation row S already determine the same glued readback Q and seal E.

For the step, assume the earlier refinements preserve Q and E. Compose the next displayed refinement with the induction hypothesis, transport the component rows by H, and replay the same finite route by C.

Apply the classifier of Definition 5899.2 componentwise. Since the classifier reads only $L, R, D, S, Q, E, H, C, P, N$ the next refinement cannot add a quotient, representative selector, host equality row, choice schedule, or completeness witness; it only repeats the finite route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5899.7 (Regular Cauchy gluing Cauchy-reindexing handoff)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier factors its gluing route through the finite* $\mathsf{CauchyReindexing}^{\uparrow}$ *surface recorded in P. The handoff exposes the common dyadic refinement ledger D, the finite observation row S, and the glued regular readback row Q as a reindexing-compatible route, and it exposes no global quotient of regular Cauchy names or ambient completeness principle.*

Proof. Read the package provenance P. It names the route through $\mathsf{CauchyReindexing}^{\uparrow}$ together with $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

The carrier rows D, S, Q are exactly the finite data consumed by that route: D supplies the common refinement ledger, S supplies the inspected windows, and Q supplies the regular rational readback after the two local windows agree on the overlap.

The support rows H, C, N transport, replay, and name this same finite path. Thus the handoff is reindexing-compatible BEDC data, not a quotient of regular names, a selected representative, or a completeness theorem. □

Theorem 5899.8 (Regular Cauchy gluing obligation closure handoff)**.** *Every obligation-closure read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier must expose the full row tuple*

$$L, R, D, S, Q, E, H, C, P, N.$$

The route factors through the common $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row, the local* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks and glued readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and*

the $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff named by* P *before any downstream real-name consumer may read the gluing packet.*

Proof. Start with the carrier and NameCert obligations. The accepted packet exposes $L, R, D, S, Q, E, H, C, P, N$, and the obligation surface has no smaller projection that still records the two local windows, their common refinement, and the glued readback.

Use the Cauchy-reindexing handoff of Definition 5899.7. It forces the gluing read through D, S, Q and through the provenance row P naming $\mathsf{CauchyReindexing}^{\uparrow}$, while the carrier rows L, R provide the two regular windows.

The rows E, H, C, N then seal, transport, replay, and name the same finite route. Thus obligation closure reads exactly the displayed tuple and factors through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$. □

Theorem 5899.9 (Regular Cauchy gluing Real seal ordering)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* E *is reachable only after the common dyadic refinement row* D*, the finite observation row* S*, and the glued readback row* Q *have been displayed. The seal-ordering route asserts no quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.*

Proof. Project the carrier definition. The row E is listed as the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after the two local windows, the common refinement ledger, the finite observations, and the glued readback have been displayed.

Apply the overlap-stability and Cauchy-reindexing handoff theorems. They keep the common dyadic refinement in D, the inspected windows in S, and the glued regular readback in Q visible before any real-name consumer can inspect E.

Since quotient stream equality, selected representatives, host equality, choice schedules, and ambient completeness are not carrier coordinates, the ordering statement cannot export them. It records only the displayed route from D, S, Q to the terminal seal E. □

Theorem 5899.10 (Regular Cauchy gluing local tail compatibility)**.** *For an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *carrier* $G = (L, R, D, S, Q, E, H, C, P, N)$*, the two local regular-Cauchy windows* L *and* R *can be consumed by a* $\mathsf{Real}^{\uparrow}$*-seal reader only after the overlap row, the common-refinement row, and the Cauchy-reindexing row exhibit a shared* $\mathsf{RegSeqRat}^{\uparrow}$ *tail boundary. The compatibility route is the finite route through* D, S, Q, H, C, P, N *supplied by the carrier, and it exports no quotient stream equality, selected representative, or ambient completeness.*

Proof. Project the carrier of Definition 5899.1. The two local windows L, R are not read independently by a Real-facing consumer; the packet first displays their shared dyadic ledger D, finite observations S, and glued readback Q.

Apply overlap stability, Definition 5899.4, and common-refinement exactness, Definition 5899.5. These theorems keep the overlap row and the common refinement visible before Q is consumed, so the two local $\mathsf{RegSeqRat}^{\uparrow}$ tails are compared at the displayed boundary rather than by host stream equality.

Apply overlap induction, Definition 5899.6, to finite chains of displayed refinements, and then use the Cauchy-reindexing handoff, Definition 5899.7. The resulting route makes the shared tail boundary stable under H and replayable by C before E is read.

Since quotient stream equality, selected representatives, and ambient completeness are absent from Definition 5899.1, the compatibility theorem can expose only the finite D, S, Q, H, C, P, N route. □

The local-tail compatibility row is consumed through the $\mathsf{RegSeqRat}^{\uparrow}$ finite-window surface of chapter 301 and the $\mathsf{StreamName}^{\uparrow}$ finite-observation surface of chapter 126. The exact dependency is the common-tail rows displayed by D and Q, together with finite-window readback through S, before the $\mathsf{Real}^{\uparrow}$ seal E is available to any downstream consumer.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet is generated from two compatible regular Cauchy windows, a common $\mathsf{DyadicRatCore}^{\uparrow}$ refinement ledger, finite $\mathsf{StreamName}^{\uparrow}$ observations, a glued $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5899.1, Definition 5899.2, Definition 5899.3, Definition 5899.4, Definition 5899.6, Definition 5899.7, Definition 5899.8, Definition 5899.9, and Definition 5899.10 seed the finite gluing route for compatible regular Cauchy windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exposure, common dyadic-refinement exactness, overlap stability, CauchyReindexing handoff, Real seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyGluingUp.

Theorem 5899.11 (Regular Cauchy gluing L10 sibling route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet reads its local gluing route through the L10 sibling lattice before it exports a NameCert row. The route first consumes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readbacks of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the Cauchy reindexing handoff of chapter 1568.*

Thus the gluing packet exports only a local finite NameCert route over the displayed dyadic, stream, regular-rational, Real-seal, and reindexing rows. It does not expose quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness.

Proof. Project the carrier of Definition 5899.1. Its visible rows are two local regular windows, the common dyadic ledger, the finite observation row, the glued regular readback, the Real seal, and the transport, replay, provenance, and naming rows.

Apply Definition 5899.3. The local NameCert surface must read exactly those rows, with the dyadic ledger, StreamName windows, RegSeqRat readback, and Real seal appearing before the certificate is exported.

Use Definition 5899.7 for the reindexing component. It identifies the same dyadic, stream, and regular readback rows as a $\mathsf{CauchyReindexing}^{\uparrow}$-compatible route.

Thus the dependency on chapter 1568 is consumed before the $\mathsf{RegSeqRat}^{\uparrow}$ readback and terminal $\mathsf{Real}^{\uparrow}$ seal are available to the gluing certificate.

The exclusions are inherited from the carrier inventory and the overlap stability theorem. Since no coordinate stores quotient stream equality, selected representatives, host real equality, choice, or ambient completeness, the L10 sibling route remains a finite local NameCert route. □

Theorem 5899.12 (Regular Cauchy gluing RegSeqRat route)**.** *Every glued readback row of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must pass through the existing* $\mathsf{RegSeqRat}^{\uparrow}$ *sibling surface of chapter 301 and the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff of chapter 1568 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is visible. The local obligations of Definition 5899.3 therefore expose a regular rational readback and a reindexing route before any Real-facing consumer may read the glued value.*

Proof. Start from the local L10 route of Definition 5899.11. It already orders the dyadic ledger, finite StreamName windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{CauchyReindexing}^{\uparrow}$ handoff, and terminal $\mathsf{Real}^{\uparrow}$ seal before the gluing NameCert row is exported.

Apply the local obligations of Definition 5899.3. The glued readback row is one of the displayed certificate rows, so it cannot be consumed without the same regular rational surface and structural support rows.

Use Definition 5899.7. The reindexing handoff keeps the dyadic, stream, and regular readback data aligned before the terminal Real seal appears. Thus a Real-facing read factors through RegSeqRat and CauchyReindexing rather than bypassing them. □

Theorem 5899.13 (Regular Cauchy gluing pasting nonescape)**.** *Every accepted pasting consumer of a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite route of Definition 5899.1, satisfy the local obligations of Definition 5899.3, and pass through the L10 sibling route of Definition 5899.11 before it exports any NameCert row. In particular, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Consequently no such pasting consumer exports quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as a coordinate of the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *read.*

Proof. Project the accepted carrier in Definition 5899.1. Its displayed route already contains the common dyadic ledger, finite StreamName observations, glued RegSeqRat readback, Real seal, and the support rows for transport, replay, provenance, and naming.

Apply Definition 5899.3. A pasting consumer can export a local NameCert row only by preserving exactly those displayed rows, so the consumer has no separate entry point around the dyadic, stream, regular readback, or seal coordinates.

Apply Definition 5899.11. The L10 route orders the same rows through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before export.

The refused objects are absent from the carrier and absent from the sibling route. Thus quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness cannot be repacked as a pasting consumer of the accepted $\mathsf{RegularCauchyGluing}^{\uparrow}$ packet. □

Theorem 5899.14 (Regular Cauchy gluing separator-use route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer that exports an apartness read or a located-comparison read must first pass through the* $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ *route of chapter 7930. Ordinary gluing consumers still export only the finite dyadic, stream,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyReindexing}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows named by Definition 5899.11.*

Proof. Start with the L10 sibling route of Definition 5899.11. It exposes the dyadic tolerance rows, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before the local gluing NameCert row is exported.

For a consumer that reads only gluing data, those rows exhaust the available surface by Definition 5899.13. The carrier has no coordinate for apartness, located comparison, quotient equality, selected representatives, choice, or ambient completeness.

If the same consumer asks for apartness or located comparison, apply the separator sibling dependency of Definition 7930.3. That route requires the stream, regular-rational, dyadic, Real-seal, located-cut, and apartness-field rows to be displayed through $\mathsf{RegularSeqApartnessSeparator}^{\uparrow}$ before such a read is available.

Thus the gluing packet itself remains the ordinary finite route, while apartness-facing or located-comparison-facing reads factor through the separator chapter rather than being smuggled into the gluing carrier. □

Theorem 5899.15 (Regular Cauchy gluing completion consumer route)**.** *Every Bishop-completion or Real-completion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the same finite L10 sibling route before exporting a gluing* $\mathsf{NameCert}$ *row. In particular, the consumer first exposes the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite observation windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The exported gluing certificate has no coordinate for quotient stream equality, selected representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already orders the dyadic ledger, stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any local gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12. A completion-facing read of the glued value therefore cannot bypass the regular rational surface or the reindexing route on its way to the terminal $\mathsf{Real}^{\uparrow}$ seal.

Apply the pasting nonescape theorem, Definition 5899.13. It keeps every gluing consumer inside the finite carrier route and excludes quotient stream equality, selected representatives, host real equality, countable choice, and ambient completeness as exportable coordinates.

Finally use the separator-use route of Definition 5899.14. Apartness-facing or located-comparison-facing completion reads factor through the separator sibling; ordinary Bishop-completion and Real-completion gluing reads still expose only the dyadic, stream, regular-rational, reindexing, and Real-seal rows before the gluing $\mathsf{NameCert}$ is named. □

Theorem 5899.16 (Regular Cauchy gluing uniform Cauchy criterion route)**.** *Every uniform-Cauchy-criterion consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must read the finite sibling route before exporting criterion data. The consumer must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff, the* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal as displayed rows of the gluing route.*

Consequently the criterion consumer exports only the finite BEDC packet obtained from those rows, componentwise hsame *transport,* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *ledger. It does not export quotient sequence equality, selected representatives, countable choice, host real equality, or ambient completeness.*

Proof. Start with the L10 sibling route of Definition 5899.11. It already places the dyadic tolerance ledger, finite stream windows, regular-rational readback, Cauchy reindexing handoff, and terminal Real seal before any gluing $\mathsf{NameCert}$ row is exported.

Apply the RegSeqRat route of Definition 5899.12 and the completion consumer theorem Definition 5899.15. Together they force a completion-facing criterion read through the regular rational surface, the reindexing handoff, and the Bishop-completion consumer rows before the terminal $\mathsf{Real}^{\uparrow}$ seal can be read.

Finally use the downstream sibling route of Definition 5899.18. It permits only concrete downstream consumers after the $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows have been displayed. Since the carrier and all cited routes omit quotient sequence equality, selected representatives, countable choice, host real equality, and ambient completeness, none of those objects can be repacked as uniform-Cauchy-criterion data. □

Theorem 5899.17 (Regular Cauchy gluing L10 sibling dependency)**.** *Before any gluing consumer reads the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ *row, it must display the finite sibling dependency route through the* $\mathsf{RegSeqRat}^{\uparrow}$ *rows of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *observation windows of chapter 126,*

the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the completion-facing handoff of chapter 3665, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. The route may also be read by Bishop-completion consumers through the completion consumer theorem Definition 5899.15, but it remains a finite dependency lattice rather than quotient sequence equality, selected representatives, countable choice, host real equality, or ambient real completeness.*

Proof. Start with Definition 5899.11. It already orders the dyadic tolerance rows, stream observation windows, regular-rational readback, Cauchy reindexing handoff, and Real seal before the local $\mathsf{NameCert}$ row is exported.

Apply Definition 5899.15. A Bishop-completion or Real-completion consumer therefore reads the same finite row sequence before it can use the gluing packet as completion-facing data.

The dependency is finite because every cited sibling appears as a displayed row or handoff in the local route. No step asks for quotient sequence equality, a selected representative of a Cauchy class, countable choice over tails, host real equality, or an ambient completeness theorem. □

Theorem 5899.18 (Regular Cauchy gluing downstream sibling route)**.** *Any cofinal, interleaving, or comparison-facing consumer of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first pass through the finite sibling route of Definition 5899.17. After the* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *rows have been displayed, such a consumer may continue only through a concrete downstream sibling such as chapter 414 or chapter 3514. The gluing packet itself exports no detached cofinal route, no interleaving selector, no quotient equality of Cauchy names, no selected representative, and no ambient completeness theorem.*

Proof. Start with Definition 5899.17. It forces every gluing consumer to display the regular rational rows, finite stream windows, dyadic tolerance rows, Bishop-completion handoff, and terminal Real seal before the local $\mathsf{NameCert}_{\mathsf{RegularCauchyGluing}^{\uparrow}}$ row is read.

Apply the completion consumer theorem Definition 5899.15. A completion-facing read therefore receives the same finite route, not a quotient stream or selected representative.

For an interleaving-facing read, use the public sibling surface of chapter 414. It supplies the alternating selector and combined modulus ledger only after the relevant $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are displayed.

For a comparison-facing read, use chapter 3514. That packet supplies the even–odd comparison and shared threshold rows as a separate finite consumer surface. Thus any cofinal or interleaving continuation is downstream of the existing L10 sibling dependency, and none of those rows is smuggled into the gluing carrier itself. □

Theorem 5899.19 (Regular Cauchy gluing tail-agreement composition)**.** *Any* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer that reads a glued* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet through Definition 5899.18 must continue through the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route of chapter 699 before treating the glued readback as completion-facing tail agreement. The composed route also reads the Cauchy limit seal of chapter 386; the gluing packet alone exports no ambient tail equality, selected representative, quotient sequence equality, or completeness witness.*

Proof. Start with Definition 5899.18. It permits downstream consumers only after the finite $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRealCompletion}^{\uparrow}$, and terminal $\mathsf{Real}^{\uparrow}$ rows have been displayed.

A completion-facing tail read then has an additional boundary. The $\mathsf{RealTailAgreementSeal}^{\uparrow}$ chapter supplies the route that turns a Real-facing readback into displayed tail-agreement data, and the $\mathsf{CauchyLimitSeal}^{\uparrow}$ chapter supplies the limit-facing seal used by completion consumers.

Thus the row order is finite: first the local gluing sibling route, then the tail-agreement seal, and finally the Cauchy-limit seal. Because each step only reuses displayed rows, no consumer can replace the route by host tail equality, a selected Cauchy representative, quotient sequence equality, or ambient completeness. □

$\mathsf{RegularCauchyGluing}^{\uparrow}$ consumes its L10 sibling lattice before any gluing read is exported: the $\mathsf{StreamName}^{\uparrow}$ finite windows provide the inspected overlap rows, $\mathsf{RegSeqRat}^{\uparrow}$ supplies the regular readback, $\mathsf{DyadicRatCore}^{\uparrow}$ carries the shared tolerance ledger, $\mathsf{Real}^{\uparrow}$ seals the terminal value, and chapter 5544 provides the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ schedule that keeps the two local Cauchy windows on one displayed budget.

Theorem 5899.20 (Regular Cauchy gluing L10 forward route)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer reaches Real-facing gluing only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of the local carrier. The route first reads the name certificate construction, then the L10 sibling route, and finally the Bishop-completion consumer route before any gluing conclusion is exported. It refuses quotient stream equality, selected limits, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness as coordinates of the gluing read.*

Proof. Project the local carrier and obligations from Definition 5899.1 and Definition 5899.3. The rows available to a consumer are the two compatible windows, the common dyadic ledger, finite observations, glued regular readback, Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the consumer through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before a local gluing certificate is exported.

Apply the Bishop-completion consumer theorem Definition 5899.22. A Real-facing or Bishop-completion consumer must display the same D, S, Q, E, H, C, P, N route, so the forward path remains finite and local to the accepted packet.

The refused objects are absent from every cited step. The name certificate construction supplies no quotient stream equality; the L10 sibling route supplies no selected limit or countable-choice schedule; and the Bishop-completion consumer route supplies no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. Hence every accepted forward gluing read factors through the displayed windows, readbacks, tolerance ledgers, and Real seal. □

Theorem 5899.21 (Regular Cauchy gluing sibling forward link)**.** *Every accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that is read as a gluing consumer first passes through the L10 sibling route of Definition 5899.11. In particular, the consumer must expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301 and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows of chapter 126 before the local gluing* $\mathsf{NameCert}$ *row is available.*

The forward link is therefore a concrete sibling dependency route rather than a detached hub edge. It also preserves the exclusions already named by the L10 route: no quotient stream equality, selected representative, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is exported by the gluing read.*

Proof. Start with the carrier and obligation rows of Definition 5899.1 and Definition 5899.3. They display the paired local windows, common dyadic ledger, finite observations, glued regular readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Apply the L10 sibling route of Definition 5899.11. That route orders the same gluing read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyReindexing}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any gluing certificate can be exported.

Thus the hub has a visible child route into the RegSeqRat and StreamName sibling chapters. The refused objects remain absent because neither the local carrier nor the L10 sibling route contains coordinates for quotient stream equality, selected representatives, choice, host equality, or ambient completeness. □

Theorem 5899.22 (Regular Cauchy gluing Bishop completion consumer)**.** *Every* $\mathsf{BishopRealCompletion}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet must first expose the same finite gluing carrier rows already displayed by Definition 5899.1. In particular, before the Bishop completion route may read the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the consumer must display the* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *glued readback* Q*, the* $\mathsf{Real}^{\uparrow}$ *seal* E*, the* $\mathsf{CauchyReindexing}^{\uparrow}$ *handoff recorded in* P*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*.*

Thus the Bishop completion consumer receives the gluing packet only as a finite L10 route over dyadic, stream, regular-rational, Real-seal, reindexing, transport, replay, provenance, and naming data. It receives no quotient stream equality, selected representative, host real equality, countable-choice schedule, or ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The rows that a completion-facing consumer can inspect are exactly the two local windows, the common dyadic ledger, the finite observations, the glued readback, the Real seal, transport, replay, provenance, and local naming.

Apply the L10 sibling route of Definition 5899.11. It orders the gluing packet through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and $\mathsf{CauchyReindexing}^{\uparrow}$ before a NameCert row is exported.

Now read the Bishop completion carrier of chapter 3665. Its completion-facing route consumes dyadic precision, finite stream windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming. The gluing rows D, S, Q, E, H, C, P, N are therefore the only rows that can feed such a consumer from $\mathsf{RegularCauchyGluing}^{\uparrow}$.

The exclusions follow from the same carrier inventory. Since neither Definition 5899.1 nor the Bishop completion route supplies quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness as a coordinate, the consumer cannot export any of those objects from the gluing packet. □

Theorem 5899.23 (Regular Cauchy gluing Real-tail forward route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or* $\mathsf{Completion}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *packet that asks for a tail-agreement seal is a forward dependency of the* $\mathsf{RealTailAgreementSeal}^{\uparrow}$ *route in chapter 699. Concretely, the gluing packet first exposes the common* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement ledger* D*, the paired* $\mathsf{StreamName}^{\uparrow}$ *observation row* S*, the glued* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the local* $\mathsf{Real}^{\uparrow}$ *seal* E*; only then may the consumer enter the Real-tail agreement and Cauchy-limit seal route through* $\mathsf{RealTailAgreementSeal}^{\uparrow}$*,* $\mathsf{CauchyLimitSeal}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{Completion}^{\uparrow}$*.*

Proof. Project the accepted gluing carrier $G = (L, R, D, S, Q, E, H, C, P, N)$ from Definition 5899.1. The Real-facing rows available before any tail agreement are exactly the common dyadic ledger D, finite observation row S, glued regular readback Q, terminal seal E, transport row H, replay row C, provenance row P, and local name row N.

Apply the Real seal ordering theorem Definition 5899.9. It forces the route to display D, S, Q before E is readable, so a consumer cannot jump from the two local windows directly to a Real-tail agreement row.

Now use the Real-tail agreement completion-consumer boundary of chapter 699. That boundary receives paired windows, dyadic readback, agreement, Cauchy-limit sealing, and Real or Com-

pletion endpoints as one finite route. Feeding it from $\mathsf{RegularCauchyGluing}^{\uparrow}$ therefore makes the gluing read a forward dependency of that Real-tail route rather than a separate kernel island.

The exclusions are inherited from the two carriers. Neither Definition 5899.1 nor the Real-tail agreement route supplies quotient stream equality, selected representatives, host convergence, countable choice, or ambient completeness as a row, so no such object can be exported by this forward dependency. □

Chapter 5900

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet names the explicit tail-majorant data used when regular rational sequences are read through finite tolerances. It sits between the $\mathsf{RegSeqRat}^{\uparrow}$ source of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ schedule of chapter 126, the dyadic tolerance ledger of chapter 291, the modulus surface of chapter 292, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet records a finite threshold service for tail estimates; it is not a completeness theorem, a selected limit operator, or a quotient equality on real names.

Falsifiable prediction: Removing the majorant-threshold row permits a $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ seal to cite a tail estimate without naming the finite stage after which the estimate is valid. Removing the dyadic tolerance row prevents the threshold from being checked against a displayed rational error bound.

Independence witness: Nearest siblings include $\mathsf{RegularCauchyTailBudget}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. None is bijective to $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ because the budget packet stores observation service, the modulus packet stores a general tolerance schedule, the regular sequence packet stores the source reads, the stream packet stores names, the dyadic packet stores rational tolerances, and the real packet stores a seal rather than the finite tail-majorant threshold binding them.

Definition 5900.1 (Regular Cauchy tail majorant carrier)**.** *A* $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (S, M, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule naming the inspected regular-sequence tail requests.*
2. *M is the majorant-threshold row assigning to each displayed tolerance request a finite stage after which the tail estimate is accepted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger whose dyadic entries are the error bounds tested by M.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that exposes the regular rational sequence only through the displayed schedule, threshold, and dyadic tolerance rows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after the scheduled tail-majorant readback has been replayed.*

6. *H records componentwise* hsame *transport for the schedule, majorant threshold, dyadic tolerance, regular readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from schedule requests through the majorant threshold, dyadic tolerance ledger, regular readback, and seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$ *row.*

The carrier accepts only packets whose regular Cauchy tail estimates are read through a displayed finite majorant threshold for the requested dyadic tolerance before any $\mathsf{Real}^{\uparrow}$ *seal row is consumed.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailMajorantUp.RegularCauchyTailMajorantTasteGate single carrier alignment`

Theorem 5900.2 (Regular Cauchy tail majorant NameCert obligations)**.** *For the carrier of Definition 5900.1, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$ *surface consists exactly of carrier admission for*

$$J = (S, M, D, R, E, H, C, P, N),$$

schedule exposure through S, majorant-threshold exposure through M, dyadic tolerance readability through D, regular rational readback through R, seal handoff through E, componentwise hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier. The displayed rows $S, M, D, R, E, H, C, P, N$ are the only coordinates of an accepted $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet.

Read the analytic rows in order. The schedule row S names the inspected tail requests, the threshold row M supplies the finite majorant stage for each displayed request, and the dyadic row D supplies the tolerance being tested.

The readback and seal rows are downstream of that finite service. The $\mathsf{RegSeqRat}^{\uparrow}$ row R may be read only after S, M, D have been displayed, and the $\mathsf{Real}^{\uparrow}$ seal row E consumes that readback rather than creating an ambient limit.

Finally H, C, P, N transport, replay, package, and name the same finite route. Since no carrier coordinate stores quotient real equality, a selected limit, countable choice, or completeness, the local NameCert obligations are exactly the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$, seed)

Scope closed: Definition 5900.1 and Definition 5900.2 bind the seed carrier and local NameCert surface for explicit regular Cauchy tail majorants.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyTailMajorant}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completeness of $\mathsf{Real}^{\uparrow}$, selected limits, quotient equality of real names, countable choice, or Lean verification.

Upgrade path: Formalization requires a Lean carrier for the schedule, dyadic tolerance, regular readback, seal handoff, and finite majorant-threshold rows.

Constructive story: A $\mathsf{RegularCauchyTailMajorant}^{\uparrow}$ packet is generated from a finite stream schedule, displayed majorant thresholds, dyadic tolerance rows, regular readback, a seal handoff, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Chapter 5901

A Concrete Naming Certificate for $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$

Following Bishop and Bridges on nested interval principles, the $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ packet records the finite Real-facing window by which nested located intervals are inspected before a real-name seal is read. It sits over the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the nested/Cantor neighbours chapter 320 and chapter 4359. The packet names a finite witness route for Real and Completion consumers; it does not claim open-cover compactness, countable choice, quotient reals, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5901.1 (RealCantorIntersectionFiniteWindow carrier)**.** *A* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (I, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the nested located-interval row, carrying the visible interval prefix, the containment witness, and the located endpoint tests read from the nested-interval surface.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter ledger, recording the dyadic width bounds and tolerance requests attached to the displayed prefix.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming exactly the interval and endpoint observations inspected by the consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and the diameter ledger.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after I, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for the nested interval, diameter, window, readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$I, D \longmapsto W \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $I, D, W, R, E, H, C, P, N$ and the displayed replay order. It has no coordinate for an infinite intersection selector, quotient real equality, open-cover compactness, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCantorIntersectionFiniteWindowUp.RealCantorIntersectionFiniteWindowTasteGate`
`single carrier alignment`

Theorem 5901.2 (RealCantorIntersectionFiniteWindow NameCert obligations)**.** *For every accepted* $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ *carrier*

$$Q = (I, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$ *surface consists exactly of carrier admission, nested located-interval exposure, dyadic diameter accounting, finite StreamName window exposure, RegSeqRat readback, Real-seal routing, componentwise transport, displayed continuation replay, package provenance, and local naming. Every Real or Completion consumer read of the packet must pass through the finite nested-window route*

$$I, D \longmapsto W \longmapsto R \longmapsto E$$

before the terminal seal is public.

Proof. Project the carrier of Definition 5901.1. The accepted packet exposes exactly the nested interval row I, dyadic diameter ledger D, finite window row W, regular readback row R, Real seal E, and the structural rows H, C, P, N.

The analytic obligations are read by projection through those rows. The nested located-interval row supplies the visible containment prefix and endpoint locatedness, while the dyadic ledger supplies the diameter budget that makes the inspected prefix usable for a Real-facing read.

Next the window and readback rows fix the finite observation route. The StreamName row W names the inspected observations, and the RegSeqRat row R is the only regular rational readback coordinate generated from them.

Finally the seal and structural rows close the packet. The continuation row places E after I, D, W, R, the transport row preserves the same coordinates, and P, N source and name only this local certificate. Since no further coordinate occurs in the carrier, the listed rows exhaust the NameCert obligations and exclude open-cover compactness, countable choice, quotient reals, selected limits, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$, seed)

Scope closed: Definition 5901.1 and Definition 5901.2 seed the finite Cantor-intersection window route over nested located intervals, dyadic diameter ledgers, StreamName windows, RegSeqRat readback, and the terminal Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, quotient reals, selected infinite intersections, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested located-interval exposure, diameter accounting, finite-window exposure, regular readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.RealCantorIntersectionFiniteWindowUp.

Constructive story: A $\mathsf{RealCantorIntersectionFiniteWindow}^{\uparrow}$ packet is generated from nested located intervals, a DyadicRatCore diameter ledger, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ selector chapter is a structural consumer of chapter 381 and chapter 5899 through its finite selected-window route.

Chapter 5902

A Concrete Naming Certificate for $\mathsf{CauchySubsequenceSelector}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy sequences, the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ packet records the finite selector schedule by which a regular rational Cauchy name may expose a monotone subsequence corridor before a Real or Completion consumer reads the terminal seal. It sits after the regular rational source of chapter 301, the finite window surface of chapter 126, the terminal real-name boundary of chapter 38, and the Cauchy hub of chapter 406. The packet gives diagonal, tail, and completion-facing chapters a named L10 selector object; it is not a quotient of streams, a selected host limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Any subsequence-selector read must display the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ route of chapter 4080 and the $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ finite-window schedule of chapter 8091 before reaching $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ consumers. The selector is therefore a visible diagonal-and-window corridor, not a detached subsequence choice. The fast selector route of chapter 1518 consumes this schedule when the cofinal subsequence corridor must be made fast enough for $\mathsf{Real}^{\uparrow}$ completion windows, while chapter 5899 consumes compatible selected windows when the subsequence corridor is glued back into a regular Cauchy route.

Definition 5902.1 (Cauchy subsequence selector carrier)**.** *A* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, W, M, G, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose rational reads are being sampled.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the source observations consumed by the selector.*
3. *M is the monotone selector row, recording a finite schedule of indices and the strict order checks between successive selected positions.*
4. *G is the Cauchy-gap ledger, recording the tail comparison requests that make the selected windows compatible with the source Cauchy evidence.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after R, W, M, G have been displayed.*
6. *H records componentwise* hsame *transport for the source, window, monotone selector, Cauchy-gap, seal, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a subsequence request through R, W, M, G, E, H, P, N.*

8. *P is the* Pkg *provenance over* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, W, M, G, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, host real equality, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5902.2 (Cauchy subsequence selector NameCert obligations)**.** *For every accepted* CauchySubsequenceSel *carrier $S = (R, W, M, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *surface consists exactly of carrier admission, selector monotonicity,* $\mathsf{StreamName}^{\uparrow}$ *window coverage, Cauchy-gap compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal non-escape, componentwise transport, displayed replay, package provenance, and local naming. Each obligation is read through the finite route*

$$R, W, M, G \longrightarrow E \longrightarrow H, C, P, N.$$

Proof. Carrier admission is projection from Definition 5902.1. The packet first exposes the regular rational source R, the finite observation windows W, the monotone selector schedule M, and the Cauchy-gap ledger G.

Selector monotonicity is local to M. The window coverage obligation then reads W at precisely the positions displayed by M, so every selected index has a finite $\mathsf{StreamName}^{\uparrow}$ observation before any handoff is consumed.

The $\mathsf{RegSeqRat}^{\uparrow}$ handoff is the source row R restricted along the displayed schedule and checked against the Cauchy-gap ledger G. The terminal seal E is downstream of those rows, and it is only a $\mathsf{Real}^{\uparrow}$ seal for the displayed subsequence corridor.

Transport through H, replay through C, provenance P, and local naming N preserve the same route. Since quotient stream equality, selected limits, host real equality, countable choice, trichotomy, and ambient Real completeness are absent from the carrier coordinates, no obligation can discharge outside the displayed finite packet. □

Theorem 5902.3 (Cauchy subsequence selector real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first read the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the monotone selector schedule, and the Cauchy-gap ledger. The handoff to E therefore supplies a finite subsequence corridor from regular rational data to Real-facing data, not a host subsequence theorem, not a selected limit, and not a quotient construction.*

Proof. Project an accepted carrier $S = (R, W, M, G, E, H, C, P, N)$. The source row R is the only regular rational input, and W is the only finite observation surface available to a consumer.

The selector row M records the monotone schedule. The Cauchy-gap ledger G records the finite tail comparisons that let the scheduled windows be read as a subsequence corridor of the same source.

Only after R, W, M, G are present may the consumer inspect E as a Real-facing seal. The transport, replay, provenance, and naming rows keep that order visible to downstream chapters.

The carrier contains no coordinate for quotient stream equality, a chosen host subsequence, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Thus every Real or Completion handoff factors through the displayed selector rows. □

Theorem 5902.4 (Cauchy subsequence selector sibling lattice route)**.** *Every* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *consumer must pass through* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *and* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *before the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows are consumed. The selector route therefore reads the diagonal-selection corridor, the regular Cauchy finite-window induction ledger, the source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, seal row E, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$ *row as one sibling lattice. Before any gluing-facing consumer reads* $\mathsf{RegularCauchyGluing}^{\uparrow}$, *the selected finite windows are certified as regular-Cauchy subsequence data through chapter 381; only then can the gluing route consume the selector rows R, W, M, G before the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

The route exports no detached subsequence choice, quotient stream equality, selected limit, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project an accepted carrier $S = (R, W, M, G, E, H, C, P, N)$ from Definition 5902.1. The source, window, monotone selector, Cauchy-gap, and seal rows are the only analytic rows available to a consumer, while H, C, P, N are the structural rows that transport, replay, source, and name them.

Apply Definition 5902.2. It forces carrier admission, selector monotonicity, window coverage, Cauchy-gap compatibility, regular-rational handoff, seal non-escape, transport, replay, provenance, and local naming before any consumer can read the selector.

Apply Definition 5902.3. The $\mathsf{Real}^{\uparrow}$ or Completion-facing read reaches E only after R, W, M, G have been displayed, so the diagonal-selection corridor and regular Cauchy window-induction ledger must be upstream of the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows named in the introduction; the same displayed windows are first certified by chapter 381 before $\mathsf{RegularCauchyGluing}^{\uparrow}$ can consume them.

Since the carrier has no coordinate for detached subsequence choice, quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient completeness, the sibling lattice cannot export any of those principles. □

Theorem 5902.5 (Cauchy subsequence selector diagonal-window readiness)**.** *Every fast-selector or gluing-facing consumer of a* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must expose the* $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ *route and the* $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$ *finite schedule before it consumes the selector rows $R, W, M, G, E, H, C, P, N$. In particular, a* $\mathsf{FastCauchySubsequence}^{\uparrow}$ *consumer cannot read the fast cofinal corridor, and a* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer cannot read compatible selected windows, until the diagonal route and finite-window schedule have already displayed the source row, finite windows, monotone selector row, Cauchy-gap ledger, terminal seal, transport, replay, provenance, and local naming rows.*

Proof. Apply Definition 5902.2. It makes $R, W, M, G, E, H, C, P, N$ the full NameCert surface for an accepted selector packet, so a downstream consumer has no row outside that finite display from which to read subsequence data.

Apply Definition 5902.3. The Real or Completion-facing handoff reaches E only after the regular rational source, finite windows, selector schedule, and Cauchy-gap ledger have been displayed.

Now apply Definition 5902.4. It places the $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ corridor and the $\mathsf{RegularCauchyWindowIndu}$ schedule before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, terminal $\mathsf{Real}^{\uparrow}$, and gluing-facing rows are consumed. Therefore both the fast-selector route and the gluing route must expose the diagonal-window readiness surface before reading $R, W, M, G, E, H, C, P, N$. □

Theorem 5902.6 (Cauchy subsequence selector regular-window handoff)**.** *Every* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *handoff that is later glued or sealed through* $\mathsf{Real}^{\uparrow}$ *first consumes* $\mathsf{RegularCauchySubsequence}^{\uparrow}$ *as the finite-window certificate that the selected rows remain regular-Cauchy subsequence data. Thus the selected windows* W *and monotone schedule* M *are certified through chapter 381 before any* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *or terminal* $\mathsf{Real}^{\uparrow}$ *consumer reads* R, W, M, G, E.

Proof. Project the accepted carrier $S = (R, W, M, G, E, H, C, P, N)$. The selected windows W and monotone schedule M are the only rows that can present subsequence data to a downstream consumer.

Use the sibling lattice route of Definition 5902.4. It places the diagonal-selection corridor and regular Cauchy window-induction ledger before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are consumed.

The regular-window handoff is therefore a finite certification step: W and M are read as regular-Cauchy subsequence data before gluing or sealing can consume R, W, M, G, E. The route has no coordinate for a detached subsequence choice, selected host limit, or ambient completeness principle. □

Theorem 5902.7 (Cauchy subsequence selector forward density route)**.** *Every* $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ *or density-facing completion consumer that reads selected Cauchy subsequence windows from an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first pass through* $\mathsf{RegularCauchySubsequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route displayed by the selector. The forward density read therefore consumes the selector rows* $R, W, M, G, E, H, C, P, N$ *as finite regular-Cauchy subsequence data, not as a separate host subsequence, a choice-selected limit, or an ambient completeness principle.*

Proof. Apply the Real handoff theorem, Definition 5902.3. It forces every Real or Completion-facing read through the regular rational source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, and terminal seal row E.

Use the sibling lattice route of Definition 5902.4 together with the regular-window handoff of Definition 5902.6. These place $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{RegularCauchySubsequence}^{\uparrow}$ before any downstream gluing, sealing, or density-facing consumer reads the selected windows.

Thus a $\mathsf{RegularCauchyDenseEmbedding}^{\uparrow}$ consumer receives only the displayed finite selector corridor. Since the carrier has no coordinate for a detached host subsequence, selected limit, quotient stream equality, or ambient completeness, the forward density route cannot bypass the existing selector handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ packet is generated from a RegSeqRat source row, finite StreamName windows, monotone selector schedule, Cauchy-gap ledger, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5902.1, Definition 5902.2, and Definition 5902.3 seed the finite selector corridor from regular rational Cauchy data to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySubsequenceSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, selector monotonicity, StreamName window coverage, Cauchy-gap compatibility, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming.

The fast selector route of chapter 1518 consumes the $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ monotone schedule when the subsequence corridor must be made fast enough for $\mathsf{Real}^{\uparrow}$ completion windows. The $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ route is consumed by chapter 1518 when a cofinal fast subsequence schedule must be displayed before a completion-facing read, and by chapter 5899 when selected finite windows are glued back into a regular Cauchy route. Thus the selector sits in the L10 dependency lattice between finite StreamName windows, regular rational source rows, fast subsequence scheduling, and regular Cauchy gluing, rather than floating as a kernel-only carrier.

Theorem 5902.8 (CauchySubsequenceSelector regular Cauchy gluing forward route)**.** *Every* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet must first read the source regular-rational row R, the selected finite windows W, the monotone selector schedule M, and the Cauchy-gap ledger G before gluing compatible selected windows back into a regular Cauchy route. The handoff to chapter 5899 therefore remains a displayed finite-window route rather than a selected representative, quotient stream equality, countable-choice schedule, or ambient Real-completeness principle.*

Proof. Project the selector carrier described in chapter 5902. Its gluing-facing data are the source row R, finite selected windows W, monotone selector schedule M, and Cauchy-gap ledger G.

Only after those rows are visible can a gluing consumer inspect compatible windows. The $\mathsf{RegularCauchyGluing}^{\uparrow}$ route then supplies its own regular-Cauchy compatibility and terminal Real-seal boundary, so the selector packet is a forward dependency of gluing rather than a replacement for the gluing certificate.

Since neither the selector rows nor the gluing route contain quotient stream equality, selected representatives, countable choice, or ambient Real completeness, the composed route exports only the finite selected-window data and the downstream regular Cauchy gluing surface. □

Theorem 5902.9 (Cauchy subsequence selector finite-window forward link)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion-facing consumer of an accepted* $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *read factors through the finite-window route*

$$\mathsf{FiniteStream}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{CauchyDiagonalSelection}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the terminal seal is available. The consumer therefore reads the selector carrier rows as displayed finite windows and diagonal-selection data, not as a quotient subsequence, selected limit, countable-choice schedule, host real equality, or ambient completeness principle.

Proof. Project the selector carrier from Definition 5902.1. Its source row, finite windows, monotone schedule, Cauchy-gap ledger, real seal, transport, replay, provenance, and local naming rows are the only coordinates available to a downstream read.

Use the real handoff of Definition 5902.3. It forces every seal consumer to pass through the regular rational source row, the $\mathsf{StreamName}^{\uparrow}$ finite windows, the selector schedule, and the Cauchy-gap ledger before reading the terminal $\mathsf{Real}^{\uparrow}$ row.

Now read the sibling lattice route of Definition 5902.4. The lattice places the selector between $\mathsf{CauchyDiagonalSelection}^{\uparrow}$, $\mathsf{RegularCauchyWindowInduction}^{\uparrow}$, finite window data, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally identify the forward consumer rows: $\mathsf{FiniteStream}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ provide the finite observations, $\mathsf{CauchyDiagonalSelection}^{\uparrow}$ supplies the displayed diagonal corridor, $\mathsf{RegSeqRat}^{\uparrow}$ reads the regular rational name, and $\mathsf{DyadicRatCore}^{\uparrow}$ records the tolerance row before the terminal seal. Since all of these rows are explicit carrier-local reads, no refused quotient, choice, selected-limit, host-equality, or ambient-completeness endpoint can enter the route. □

The selector packet also consumes the regular-Cauchy subsequence surface of chapter 381. Its monotone selector row, finite windows, and Cauchy-gap ledger are read as a route inside the same regular-Cauchy finite-window surface before the terminal $\mathsf{Real}^{\uparrow}$ seal is available. Thus a downstream completion consumer sees the selected subsequence as displayed regular-Cauchy data, not as a detached host subsequence or a separate choice principle. The selector dependency route is consumed by both the fast subsequence handoff of chapter 1518 and the regular-Cauchy gluing route of chapter 301. Thus a downstream Cauchy consumer reads the same displayed selector schedule, finite windows, and gap ledger before treating the selected route as fast enough for completion-facing readback.

Theorem 5902.10 (Cauchy subsequence selector orphan lattice route)**.** *Every selector read consumed by* $\mathsf{CauchySubsequenceExtractor}^{\uparrow}$ *must preserve the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ *witness row, and the* $\mathsf{Real}^{\uparrow}$ *terminal seal before any subsequence extraction is public. Thus the extractor-facing selector route is the finite lattice*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegularCauchyMinModulus}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

inside the accepted $\mathsf{CauchySubsequenceSelector}^{\uparrow}$ *packet, not a detached subsequence choice or quotient stream selection.*

Proof. Project the selector carrier of Definition 5902.1. The available selector rows are the source row R, finite windows W, monotone selector schedule M, Cauchy-gap ledger G, terminal seal E, transport, replay, provenance, and local naming.

Use the selector dependency route and the cofinal subsequence dependency route. They force every extractor-facing read to pass through the displayed selector schedule before the extraction row can be consumed as finite subsequence data.

Finally read the regular-Cauchy minimum-modulus forward link. It keeps the $\mathsf{RegularCauchyMinModulus}^{\uparrow}$ witness visible alongside the $\mathsf{RegSeqRat}^{\uparrow}$ source row and $\mathsf{StreamName}^{\uparrow}$ finite windows before the terminal $\mathsf{Real}^{\uparrow}$ seal is public. Hence the selector exports only the stated finite lattice to the extractor and no separate host-side subsequence selection. □

Chapter 5903

A Concrete Naming Certificate for RiemannIntegrationCompletion$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegrationCompletion$^{\uparrow}$ packet records the finite Cauchy-sum completion data visible when an integration consumer treats Riemann sums as a regular real-valued completion route. It sits after the integrability witness of chapter 1960, the tagged-sum gauge surface of chapter 4115, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the boundary object that completion-facing integration consumers can inspect without importing host function quotients, selected refining partitions, measure completion, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 5903.1 (Riemann integration completion carrier)**.** *A* RiemannIntegrationCompletion$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, G, S, Q, R, E, H, C, P, N)$$

with the following rows.

- *I is the* RiemannIntegrable$^{\uparrow}$ *row displaying finite Darboux-gap or integrability evidence for the inspected integrand and interval.*
- *G is the* RiemannSumGauge$^{\uparrow}$ *row carrying tagged partitions, gauge windows, mesh control, and finite sum observations.*
- *S is the finite Cauchy-sum schedule row: it records only displayed pairs of gauge windows and their rational comparison tolerances.*
- *Q is the dyadic or rational comparison row used to compare two displayed Cauchy-sum observations at a stated tolerance.*
- *R is the* RegSeqRat$^{\uparrow}$ *readback row reached from the finite Cauchy-sum schedule and comparison rows.*
- *E is the terminal* Real$^{\uparrow}$ *seal row for the completed integration value read.*
- *H is componentwise* hsame *transport for I, G, S, Q, R, E.*
- *C is displayed* Cont *replay of the same finite route.*
- *P is* Pkg *provenance over* RiemannIntegrationCompletion$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

- *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrationCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one displayed finite Cauchy-sum completion history. It has no coordinate for a host function quotient, a selected refining sequence of partitions, a completed measure object, countable choice, or an ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.RiemannIntegrationCompletionUp.RiemannIntegrationCompletionTasteGate`
`single carrier alignment`

Theorem 5903.2 (Riemann integration completion NameCert obligations)**.** *For every accepted* $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ *carrier* $K = (I, G, S, Q, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RiemannIntegrationCompletio}}$ *surface consists exactly of carrier admission, integrability-source exposure through I, tagged-sum gauge exposure through G, finite Cauchy-sum schedule exposure through S, tolerance comparison through Q, regular rational readback through R, terminal Real seal non-escape through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5903.1. The only integration rows available to the certificate are I and G, and the only completion-facing rows are the finite schedule S, comparison row Q, regular rational readback R, and Real seal E.

The source rows are already named by chapter 1960 and chapter 4115. Their use here is by displayed BHist rows, not by a host-level function space or by quotient equality between integrands.

The completion side is constrained by the regular rational and Real boundaries of chapter 301 and chapter 38. Therefore the local certificate can expose only $I, G, S, Q, R, E, H, C, P, N$, with H, C, P, N supplying transport, replay, provenance, and naming for the same finite route. □

Theorem 5903.3 (Riemann integration completion Cauchy-sum handoff)**.** *Every completion-facing read of an accepted* $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ *packet must pass from the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *source row I through the* $\mathsf{RiemannSumGauge}^{\uparrow}$ *row G, finite Cauchy-sum schedule S, tolerance comparison row Q, and* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is read. The handoff exports finite Cauchy-sum comparison data only; it exports no selected sequence of tagged partitions, quotient integrand equality, measure completion, or countable-choice selector.*

Proof. Start with the source rows. The integrability row I supplies the displayed finite evidence that a Riemann integration route is being named, while G supplies the finite tagged-gauge and sum observations available to downstream consumers.

The Cauchy-sum schedule S can only compare displayed observations through Q. Once a comparison has been displayed at a stated tolerance, the regular rational handoff R packages the finite Cauchy behavior as a $\mathsf{RegSeqRat}^{\uparrow}$-facing read.

Finally the terminal seal E is read only after R. The excluded selected partition sequence, quotient integrand equality, measure completion, and countable-choice selector have no coordinate in K, so no completion-facing read can bypass the route I, G, S, Q, R, E. □

Theorem 5903.4 (Riemann integration completion Real seal non-escape)**.** *For every accepted* $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ *packet, any* $\mathsf{Real}^{\uparrow}$*-facing integration value read is sealed by the displayed finite Cauchy-sum route $I, G, S, Q, R, E, H, C, P, N$. The Real seal cannot be used as a detached integral value, an ambient completeness witness, a host-level equality of integrands, or a measure-theoretic completion object.*

Proof. Apply Definition 5903.3. It places the terminal Real read after the integrability source, tagged-sum gauge surface, finite Cauchy-sum schedule, tolerance comparison, and regular rational readback.

Use Definition 5903.2 for the structural rows. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row all replay the same finite route and add no detached Real coordinate.

Thus the seal E remains a terminal observation of the finite Cauchy-sum packet. A consumer may inspect the displayed integration completion route, but cannot extract host equality, selected representatives, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness from this chapter. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegrationCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegrationCompletion}^{\uparrow}$ packet is generated from a $\mathsf{RiemannIntegrable}^{\uparrow}$ source row, a $\mathsf{RiemannSumGauge}^{\uparrow}$ tagged-sum gauge row, a finite Cauchy-sum schedule row, a dyadic or rational comparison row, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegrationCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5903.1, Definition 5903.2, Definition 5903.3, and Definition 5903.4 seed the finite Cauchy-sum completion boundary over $\mathsf{RiemannIntegrable}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegrationCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected refining partitions, quotient integrand equality, measure completion, countable choice, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, finite Cauchy-sum comparison, regular rational handoff, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannIntegrationCompletionUp.

Chapter 5904

A Concrete Naming Certificate for $\mathsf{CauchySchwarzCompletion}^{\uparrow}$

Following Riesz and Bishop–Bridges constructive analysis, the $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ packet records the finite completion-stable route by which a Cauchy–Schwarz estimate is carried from displayed real-vector inequality data into Cauchy-window and complete-metric handoff rows. It sits after the finite inequality certificate of chapter 1684, the Cauchy-window surface of chapter 406, the complete-metric boundary of chapter 128, and the terminal real-name seal of chapter 38. The packet names only the displayed finite route through regular rational readback, stream schedules, completion handoff, transport, replay, provenance, and local naming; it does not import a host Hilbert completion, quotient stream equality, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5904.1 (Cauchy–Schwarz completion carrier)**.** *A* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (R, W, M, Q, S, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{CauchySchwarzReal}^{\uparrow}$ *inequality row, including the displayed finite-vector source, inner-product read, norm-square rows, and dyadic square comparison already sealed for real-analysis consumers.*
2. *W is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *Cauchy-window row, carrying the common finite windows through which the inequality row is inspected as a completion-facing sequence of estimates.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row, recording the displayed completeness request and the finite nonescape boundary for the estimate route.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the Cauchy-window observations and the dyadic tolerance comparisons attached to R.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row that exposes the finite observation windows shared by R, W, Q, and M.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the estimate value after the Cauchy-window and complete-metric handoff rows have been displayed.*
7. *H records componentwise* hsame *transport for $R, W, M, Q, S, E, C, P, N$.*

8. *C records displayed* Cont *replay from the finite Cauchy–Schwarz inequality row through the Cauchy-window, regular-readback, stream-schedule, complete-metric, and Real-seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$, $\mathsf{CauchySchwarzReal}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for host Hilbert completion, quotient equality of Cauchy names, selected completed representatives, countable choice, detached real limits, or ambient completeness.

Theorem 5904.2 (Cauchy–Schwarz completion NameCert obligations)**.** *For every accepted* CauchySchwarzComple *carrier $K = (R, W, M, Q, S, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy–Schwarz real-inequality exposure through R, Cauchy-window exposure through W, complete-metric handoff exposure through M, regular rational readback through Q, shared stream scheduling through S, terminal Real seal non-escape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6898.1. The only inequality source is the displayed $\mathsf{CauchySchwarzReal}^{\uparrow}$ row R, whose finite vector, inner-product, norm-square, and dyadic square-comparison data are already part of the accepted packet.

The completion-facing route is then forced by W, M, Q, S, E. The Cauchy-window row W exposes the common finite windows, the complete-metric row M records the displayed handoff request, the regular rational row Q reads the finite observations, the stream row S schedules those observations, and the terminal Real row E seals the same route.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no row for host Hilbert completion, quotient Cauchy-name equality, selected completed representatives, countable choice, detached real limits, or ambient completeness, the listed rows exhaust the local NameCert obligation surface. □

Theorem 5904.3 (Cauchy–Schwarz completion limit handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ *packet must first read the finite route*

$$\mathsf{CauchySchwarzReal}^{\uparrow}, \mathsf{CauchySequenceSpace}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow} \longmapsto R, W, M, Q, S, E, H, C, P, N.$$

In particular, the Cauchy–Schwarz estimate is available at a completion boundary only after the Cauchy-window row, complete-metric handoff row, regular rational readback, stream schedule, terminal Real seal, transport, replay, provenance, and local naming rows have all been displayed.

Proof. Start from an accepted packet $K = (R, W, M, Q, S, E, H, C, P, N)$. The estimate source is R, so every inequality read already factors through the finite real-vector and dyadic comparison rows named by chapter 1684.

The estimate can then enter completion only through the visible Cauchy-window route. The row W supplies the inspected Cauchy windows, S schedules the finite observations, and Q records the regular rational readback attached to those windows.

The handoff row M is the only complete-metric coordinate, and the Real seal E is downstream of R, W, M, Q, S. Thus a consumer cannot read a completion-stable estimate without first displaying the Cauchy-window, complete-metric, readback, schedule, and seal rows.

Finally, H, C, P, N keep the handoff attached to the same packet by transport, replay, provenance, and local naming. The route exports a finite completion-stable estimate boundary, not a

host Hilbert completion, quotient stream equality, selected limit, countable choice principle, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem. □

Falsifiable prediction: An accepted $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ read is refuted if a completion-facing consumer obtains the estimate boundary without first displaying the CauchySchwarzRealUp inequality row, CauchySequenceSpaceUp window row, CompleteMetricUp handoff row, RegSeqRatUp readback, StreamNameUp schedule, RealUp seal, transport, replay, provenance, and local NameCert rows.

Independence witness: cauchy-schwarz-real, cauchysequence-space, completemetric, regseqrat, streamname, real: $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an inequality row, Cauchy-window row, complete-metric handoff row, regular readback, stream schedule, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySchwarzCompletion}^{\uparrow}$ packet is generated from a CauchySchwarzRealUp inequality row, CauchySequenceSpaceUp windows, a CompleteMetricUp handoff row, RegSeqRatUp readback, StreamNameUp schedules, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6898.1, Definition 6898.2, and Definition 6898.3 seed the finite completion-stable Cauchy–Schwarz estimate route over displayed Cauchy windows and complete-metric handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySchwarzCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Hilbert completion, quotient equality of Cauchy names, selected completed representatives, countable choice, detached real limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy–Schwarz inequality exposure, Cauchy-window exposure, complete-metric handoff, regular rational readback, stream scheduling, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySchwarzCompletionUp.

Chapter 5905

A Concrete Naming Certificate for $\mathsf{DCPO}^{\uparrow}$

Following Scott domain theory and standard order theory, the $\mathsf{DCPO}^{\uparrow}$ packet records the directed-complete partial-order surface as finite BEDC naming data. It sits after the directed-index substrate of chapter 1607, the separated order surface of chapter 53, the Moore–Smith net route of chapter 3403, the directed-filter corridor of chapter 3898, and the Cauchy-net completion boundary of chapter 589. It is also the order-theoretic neighbour of the displayed directed-supremum packet in chapter 4164. The packet names finite directed-supremum evidence for completion routes; it is not a host powerset lattice, not a quotient order, not a selected lub operator, and not an ambient completeness theorem.

Definition 5905.1 (DCPO carrier)**.** *A* $\mathsf{DCPO}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Poset}^{\uparrow}$ *order row, exposing the displayed comparison relation and the antisymmetry witness accepted by the packet.*
2. *I is the* $\mathsf{DirectedSet}^{\uparrow}$ *index row for a directed family whose inspected addresses are available to completion consumers.*
3. *W is the finite directed-window row, listing the displayed subfamilies of the directed diagram that a consumer may request.*
4. *S is the supremum row, recording the displayed least-upper-bound read attached to the directed windows in W.*
5. *M is the* $\mathsf{MooreSmithNet}^{\uparrow}$ *handoff row, reading the same directed family as a finite net route before any convergence or completion consumer is reached.*
6. *F is the* $\mathsf{DirectedFilter}^{\uparrow}$ *handoff row, translating the same directed-tail ledger into a filter-base corridor.*
7. *Q is the* $\mathsf{CauchyNet}^{\uparrow}$ *and completion-facing row, available only after O, I, W, S, M, F have been displayed.*
8. *L is the lower-bound exactness row, recording that a candidate receiving every displayed upper-bound comparison from S is already above the named supremum by the order row O.*

9. *H records componentwise* hsame *transport for the order, directed-family, supremum, handoff, exactness, replay, provenance, and naming rows.*
10. *C records displayed* Cont *replay from a directed-supremum request through $O, I, W, S, M, F, Q, L, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{DCPO}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{ScottDomain}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, I, W, S, M, F, Q, L, H, C, P, N$. It has no coordinate for arbitrary joins, quotient-order equality, selected least-upper-bound functions, global choice over directed families, host powersets, or ambient order completeness.

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier field projection round trip`

Theorem 5905.2 (DCPO NameCert obligations)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier $D = (O, I, W, S, M, F, Q, L,$ the local* $\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$ *surface consists exactly of order admission, directed-index admission, finite directed-window coverage, directed-supremum readability, Moore–Smith net handoff, directed-filter handoff, CauchyNet and completion-facing handoff, lower-bound exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every directed-supremum consumer must read the rows O, I, W, S, L before treating the supremum as public, and no obligation is stored outside $O, I, W, S, M, F, Q, L, H, C, P, N$.*

Proof. Project the carrier of Definition 5905.1. The order and directed-family rows are O, I, W, S, L, while the completion-facing handoff rows are M, F, Q and the structural rows are H, C, P, N.

Order admission is read from O. Directed-index admission and window coverage are read from I and W, so a consumer cannot ask for a directed supremum before the inspected directed family has been displayed.

The supremum row S supplies the named upper-bound endpoint, and the exactness row L records the leastness comparison back through the same order row O. Thus a lower-bound or completion consumer receives a displayed directed-supremum certificate rather than a host lub selector.

The sibling rows M, F, Q expose the same directed family to Moore–Smith nets, directed filters, and Cauchy-net completion routes. Transport and replay through H and C preserve the displayed rows, while P, N source and name the same finite carrier. Since none of the excluded host objects appears as a coordinate, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier local namecert surface`

Theorem 5905.3 (DCPO directed-supremum handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier must factor its directed-supremum read through*

$$\mathsf{Poset}^{\uparrow}, \mathsf{DirectedSet}^{\uparrow}, W, S, L \longmapsto \mathsf{MooreSmithNet}^{\uparrow}, \mathsf{DirectedFilter}^{\uparrow}, \mathsf{CauchyNet}^{\uparrow} \longmapsto H, C, P, N.$$

The route exposes the order row, the directed index row, finite directed windows, the named supremum, and lower-bound exactness before any Moore–Smith net, directed-filter, or CauchyNet consumer is read. It does not export arbitrary joins, quotient-order equality, global directed-choice data, host powersets, or ambient completeness.

Proof. Begin with an accepted carrier $D = (O, I, W, S, M, F, Q, L, H, C, P, N)$. The directed-supremum data are not supplied by an external order-completion object; the only available order and directed-family evidence is the displayed row family O, I, W, S, L.

The Moore–Smith route M and directed-filter route F are downstream of those rows. Each therefore receives the same finite directed-tail ledger and cannot replace it by a selected cofinal net, quotient filter, or host-level directed diagram.

The CauchyNet and completion-facing row Q is downstream of M and F. It may consume the directed-supremum handoff, but it cannot read an ambient completion witness before the finite order, directed-window, supremum, and leastness rows have been displayed.

Finally, componentwise transport and continuation replay through H and C preserve this row order, and P, N record provenance and local naming for the same packet. Since arbitrary joins, quotient equality, directed-choice data, host powersets, and ambient completeness are absent from the carrier, every public completion read remains inside the displayed handoff. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier directed supremum handoff`

Theorem 5905.4 (DCPO Scott-domain compatibility)**.** *The* $\mathsf{ScottDomain}^{\uparrow}$ *consumer of an accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier reads only the displayed directed-supremum route* O, I, W, S, L *and the structural rows* H, C, P, N*. Compact-basis, Scott-open, and represented-space observations may be attached downstream by chapter 4164, but the* $\mathsf{DCPO}^{\uparrow}$ *packet itself exports only the directed-complete order substrate.*

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoScottDomainCompatibility`

Proof. Apply Definition 5905.2. The DCPO obligation surface exposes the order row, directed-index row, directed windows, supremum row, and lower-bound exactness row before any sibling consumer is reached.

The Scott-domain packet of chapter 4164 adds basis, Scott-open, and represented-space rows on top of a displayed directed-supremum route. When it consumes a $\mathsf{DCPO}^{\uparrow}$ carrier, those rows can only be attached after O, I, W, S, L have been read.

Apply Definition 5905.3. Transport, replay, provenance, and local naming remain componentwise, so the Scott-facing route cannot turn the DCPO packet into a host domain, quotient order, selected basis enumeration, or ambient semantic-completeness claim. □

Theorem 5905.5 (DCPO carrier directed obligations)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the carrier-admission and directed-window obligations are exactly the displayed rows O, I, W*. A directed-supremum consumer must first read the* $\mathsf{Poset}^{\uparrow}$ *order row* O*, the* $\mathsf{DirectedSet}^{\uparrow}$ *index row* I*, and the finite directed-window row* W *before any supremum, net, filter, or completion-facing row is public.*

Proof. Project the carrier of Definition 5905.1. The row O is the only order-admission coordinate, and the row I is the only directed-index coordinate.

The finite windows are stored in W. Since the carrier has no coordinate for a host directed family, global choice over indices, quotient-order equality, or ambient powerset lattice, a consumer cannot bypass I and W when asking for directed coverage.

The later rows $S, M, F, Q, L, H, C, P, N$ are therefore downstream of O, I, W. They may consume the directed-window surface, but they cannot supply carrier admission or directed-window coverage in place of those rows. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier carrier directed obligations`

Theorem 5905.6 (DCPO supremum ledger obligations)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the public supremum ledger is the row pair S, L *read over the order and directed-window rows* O, I, W*. The row* S *supplies the named directed-supremum read, and* L *supplies lower-bound exactness for every displayed upper-bound comparison exposed through* O*.*

Proof. Begin with the directed rows O, I, W from Definition 5905.5. These rows determine which order and directed-window data are visible to the supremum reader.

Read S next. It records the named supremum endpoint attached to the displayed directed windows, so the supremum is public only as a row of the finite carrier.

Finally read L. This lower-bound exactness row sends every displayed upper-bound comparison back through O, forcing leastness to remain an order-row obligation rather than a selected least-upper-bound operator or an ambient completeness theorem. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier supremum ledger obligations`

Theorem 5905.7 (DCPO continuity scope obligations)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the Moore–Smith, directed-filter, and CauchyNet consumers can read the directed-complete order surface only through the finite handoff route

$$O, I, W, S, L \longmapsto M, F, Q \longmapsto H, C, P, N.$$

The rows H, C, P, N *preserve componentwise transport, displayed replay, provenance, and local naming for this route; they do not add ambient continuity, host completion, arbitrary joins, or selected suprema.*

Proof. First read the directed-supremum ledger of Definition 5905.6. It exposes the order row, directed-index row, finite windows, named supremum, and lower-bound exactness as the only public directed-complete substrate.

Next pass to M, F, Q. The Moore–Smith row, directed-filter row, and CauchyNet row consume that substrate in sequence; none of them can introduce a new directed family, quotient equality, host powerset, arbitrary join operation, or ambient completion witness.

Finally use H, C, P, N. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$ row preserve the same finite route, so the continuity-facing scope remains exactly the displayed carrier route. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier continuity scope obligations`

Theorem 5905.8 (DCPO directed-window coverage)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the finite directed-window row W *is the only coverage surface available to a directed-supremum consumer. A consumer must read* O, I, W *before reading* S, L *and before using the handoff of Definition 5905.3; no arbitrary join family, host powerset of directed subsets, global choice over indices, quotient-order equality, selected least-upper-bound function, or ambient completeness coordinate is exported.*

Proof. Project the carrier of Definition 5905.1. The rows O and I expose the order and directed-index data, while W is the finite window row attached to those data.

Apply Definition 5905.3. Its displayed route places O, I, W before S, L and before the Moore–Smith, directed-filter, and CauchyNet rows M, F, Q.

The remaining rows H, C, P, N only transport, replay, source, and name that same carrier. Since the carrier has no coordinate for arbitrary joins, host powersets, quotient-order equality, global directed choice, selected lub functions, or ambient completeness, directed-window coverage is exactly the displayed row W over O, I. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier directed window coverage`

Theorem 5905.9 (DCPO lower-bound exactness)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the lower-bound exactness row L is read only after the order row O, directed-index row I, finite-window row W, and supremum row S have been displayed. It records leastness through the same order row used by Definition 5905.3, and it does not introduce arbitrary joins, quotient-order equality, selected lub functions, global choice, host powersets, or ambient completeness.

Proof. Begin with the public supremum ledger of Definition 5905.6. It identifies S, L as the supremum and lower-bound exactness rows read over O, I, W.

The exactness row L therefore cannot be detached from the displayed order row. It sends lower-bound comparisons back through O after the directed windows in W and the named supremum in S have been exposed.

Apply Definition 5905.3. The downstream rows M, F, Q, H, C, P, N consume this exactness route; they do not supply a selected least-upper-bound function, quotient-order equality, arbitrary join operation, global choice schedule, host powerset, or ambient completeness theorem. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier lower bound exactness`

Theorem 5905.10 (DCPO completion-facing handoff)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the completion-facing route is exactly the finite handoff

$$O, I, W, S, L \longmapsto M, F, Q \longmapsto H, C, P, N.$$

The CauchyNet row Q is public only after the Moore–Smith row M and directed-filter row F have consumed the displayed directed-window and lower-bound exactness rows named in Definition 5905.3. The route exports no arbitrary joins, quotient-order equality, selected lub functions, global choice, host powersets, or ambient completeness.

Proof. Read O, I, W, S, L first. By Definition 5905.8 and Definition 5905.9, these rows provide the order, directed index, finite windows, named supremum, and leastness comparisons.

Apply Definition 5905.3. The Moore–Smith row M, directed-filter row F, and CauchyNet row Q are downstream consumers of that displayed directed-complete order surface.

Finally inspect H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same route and cannot add arbitrary joins, quotient-order equality, selected lub functions, global choice, host powersets, or ambient completeness. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier completion facing handoff`

Theorem 5905.11 (DCPO directed-supremum scope)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

every scoped directed-supremum read factors through the route

$$O, I, W, S, L \longmapsto M, F, Q \longmapsto H, C, P, N.$$

The read first exposes the $\mathsf{Poset}^{\uparrow}$ *order row,* $\mathsf{DirectedSet}^{\uparrow}$ *index row, finite directed-window row, named supremum row, and lower-bound exactness row; only then may it reach the Moore–Smith, directed-filter, and CauchyNet handoff rows, followed by componentwise transport, displayed replay, provenance, and local naming. The scope exports no arbitrary joins, quotient equality, host powersets, selected lub operator, global directed choice, or ambient completeness.*

Proof. Project the carrier of Definition 5905.1. The only rows available to a directed-supremum reader before handoff are O, I, W, S, L.

Apply Definition 5905.8 and Definition 5905.9. The first theorem forces every directed read to pass through the finite window row W over O, I, and the second attaches leastness to S, L over the same order row.

Finally apply Definition 5905.10. It places M, F, Q after the displayed directed-supremum route and leaves H, C, P, N as transport, replay, provenance, and naming rows. Since the carrier has no coordinate for arbitrary joins, quotient equality, host powersets, selected lubs, global directed choice, or ambient completeness, the displayed route is the whole scoped directed-supremum surface. □

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier directed supremum scope`

Theorem 5905.12 (DCPO obligation-closure package)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier*

$$D = (O, I, W, S, M, F, Q, L, H, C, P, N),$$

the directed-window coverage, supremum ledger, continuity scope, and completion-facing handoff obligations jointly exhaust the obligation *surface over* $\mathsf{Poset}^{\uparrow}$, $\mathsf{DirectedSet}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The package consists exactly of the order row O, directed-index row I, finite directed-window row W, supremum row S, lower-bound exactness row L, Moore–Smith handoff M, directed-filter handoff F, CauchyNet handoff Q, transport row H, replay row C, provenance row P, and local naming row N; it exports no selected lub operator, host powerset, quotient order, global directed-choice schedule, or ambient completeness claim.*

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier obligation closure package`

Proof. Start with carrier admission and directed coverage. Theorems 5905.5 and 5905.8 identify O, I, W as the whole order, directed-index, and finite directed-window surface available before any supremum consumer is public.

Next read the supremum ledger. Theorems 5905.6 and 5905.9 place S and L over the same O, I, W rows, so leastness remains a displayed order-row obligation rather than a selected least-upper-bound function or an ambient completeness theorem.

Now pass to the continuity and completion rows. The continuity scope theorem Definition 5905.7 and the completion-facing handoff theorem Definition 5905.10 force M, F, Q to consume the route O, I, W, S, L before any Moore–Smith, directed-filter, or CauchyNet read is public.

Finally use H, C, P, N. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming preserve the same finite route and add no new carrier coordinate. Therefore the listed rows jointly exhaust the obligation-closure surface, while selected lubs, host powersets, quotient orders, directed-choice schedules, and ambient completeness remain outside the package. □

Theorem 5905.13 (DCPO obligation-closure record)**.** *For every accepted* $\mathsf{DCPO}^{\uparrow}$ *carrier* $D = (O, I, W, S, M, F, Q, L, H, C, P, N)$*, the displayed rows* $O, I, W, S, L, M, F, Q, H, C, P, N$ *form the complete obligation record consumed by* $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedFilter}^{\uparrow}$, *and* $\mathsf{CauchyNet}^{\uparrow}$. *The record exposes order admission, directed-index admission, finite directed-window coverage, directed-supremum readability, lower-bound exactness, Moore–Smith handoff, directed-filter handoff, CauchyNet handoff, transport, replay, provenance, and local naming, and it exports no arbitrary joins or selected least-upper-bound functions.*

Lean4 ✓ `BEDC.Derived.DcpoUp.DcpoCarrier obligation closure record`

Proof. Start with directed-window coverage and the supremum ledger, Definition 5905.8 and Definition 5905.6. They identify O, I, W, S as the visible order, directed-index, finite-window, and supremum rows that every downstream order consumer must read.

Next apply lower-bound exactness, Definition 5905.9. It attaches L to the same O, I, W, S route, so leastness remains a displayed order-row obligation rather than a selected lub operation.

Finally use continuity scope and completion-facing handoff, Definition 5905.7 and Definition 5905.10. These theorems force $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{MooreSmithNet}^{\uparrow}$, $\mathsf{DirectedFilter}^{\uparrow}$, and $\mathsf{CauchyNet}^{\uparrow}$ consumers to receive the same finite route through M, F, Q, H, C, P, N. Since the carrier contains no rows for arbitrary joins, host powersets, quotient-order equality, global directed choice, or selected lubs, the listed rows are the complete obligation record. □

Closure status for $\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DCPO}^{\uparrow}$ packet is generated from a Poset order row, DirectedSet index row, finite directed windows, a directed-supremum row, MooreSmithNet and DirectedFilter handoffs, a CauchyNet completion-facing row, lower-bound exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$, scopedClosed)

Scope closed: Definition 5905.1, Definition 5905.2, Definition 5905.3, Definition 5905.5, Definition 5905.6, Definition 5905.7, and Definition 5905.13 bind the DCPO carrier, order, directed-index, finite-window, supremum, lower-bound, BHist, hsame, Cont, Pkg, and NameCert obligations as the displayed scope consumed by completion and domain-theoretic readers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DCPO}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close arbitrary joins, quotient-order equality, selected least-upper-bound functions, global choice over directed families, host powersets, ambient order completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires binding the displayed obligation record to a checked kernel route for carrier admission, directed-window coverage, supremum readability, lower-bound exactness, MooreSmithNet handoff, DirectedFilter handoff, CauchyNet handoff, transport, replay, provenance, and local naming.

Chapter 5906

A Concrete Naming Certificate for $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ packet records the finite route by which a regular Cauchy real presentation and a located Dedekind-cut boundary are compared without selecting a host limit or quotient representative. It sits over the dyadic tolerance cells of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the located-cut completion surface of chapter 5554, and the Cauchy–Dedekind boundary of chapter 1745. The router is a finite NameCert carrier for the comparison gap; it is not a construction of $\mathsf{Real}^{\uparrow}$ from primitive cuts and not an ambient completeness theorem.

Definition 5906.1 (Cauchy–Dedekind gap-router carrier)**.** *A* $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, S, R, E, L, B, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the positive dyadic gaps, refinements, and comparison cells used by both sides of the route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule for the observed Cauchy name and the observed cut-boundary probes.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, binding the Cauchy-facing windows in S to regular rational observations.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R have been displayed.*
5. *L is the* $\mathsf{LocatedCut}^{\uparrow}$ *lower-and-upper cut ledger addressed by the same dyadic and stream windows.*
6. *B is the* $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$ *boundary row comparing the displayed regular readback with the displayed located-cut ledger.*
7. *J is the gap-routing judgment row. It records, for each public comparison request, which dyadic gap cell in D routes the request through R toward E or through L, B toward the boundary surface.*
8. *H records componentwise* hsame *transport for the dyadic, stream, readback, real-seal, located-cut, boundary, judgment, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the comparison route*

$$D, S \longmapsto R, E \qquad \textit{and} \qquad D, S \longmapsto L, B, J.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedCut}^{\uparrow}$, $\mathsf{DedekindCutCauchyBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDedekindGapRouter}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying exactly these rows. It has no coordinate for quotient stream equality, primitive Dedekind-cut equality, selected limits, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDedekindGapRouterUp.CauchyDedekindGapRouterTasteGate single carrier alignment`

Theorem 5906.2 (Cauchy–Dedekind gap-router NameCert obligations)**.** *For an accepted* CauchyDedekindGapRout *carrier*

$$G = (D, S, R, E, L, B, J, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyDedekindGapRouter}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the dyadic tolerance ledger, stream-window schedule, regular rational readback, real seal, located-cut ledger, Cauchy–Dedekind boundary row, gap-routing judgment, transport, replay, provenance, and local naming rows;*
2. *window exactness, requiring every public comparison request to be addressed by S before either the Cauchy-facing readback R or the located-cut ledger L can be inspected;*
3. *dyadic gap exactness, requiring every routed comparison to consume a displayed positive dyadic cell from D;*
4. *Cauchy-side handoff, requiring the* $\mathsf{Real}^{\uparrow}$ *seal E to be downstream of D, S, R;*
5. *cut-side handoff, requiring the boundary row B to compare only the displayed R and L rows;*
6. *routing determinacy, requiring J to name the gap cell and side of the comparison without adding a second equality principle;*
7. *classifier transport and replay, requiring H and C to preserve exactly the listed rows;*
8. *non-escape, excluding quotient stream equality, primitive cut equality, selected limits, trichotomy, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No NameCert obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 5906.1. The accepted packet exposes the dyadic, stream, regular readback, real-seal, located-cut, boundary, judgment, and structural rows as one finite display.

Window exactness follows from the shared row S. A public request cannot reach either the Cauchy-facing row R or the located-cut row L until its finite observation window has been named.

Dyadic gap exactness is supplied by D. The same positive dyadic cell is the comparison budget used by the regular readback and by the located boundary row, so the router cannot introduce an unbudgeted comparison.

The Cauchy-side handoff reaches E only after D, S, R have been displayed. The cut-side handoff reaches B only after D, S, L and the common readback row have been displayed.

The routing row J then records which displayed gap cell and side of the comparison is used. Since J is a row of the carrier rather than an external equality principle, it cannot export quotient stream equality or primitive cut equality.

Finally H, C, P, N transport, replay, source, and name only the listed rows. The excluded completeness and choice principles have no carrier coordinate and therefore cannot become local obligations. □

Theorem 5906.3 (Cauchy–Dedekind gap-router L10 handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedCut}^{\uparrow},\ \mathsf{DedekindCutCauchyBoundary}^{\uparrow}.$$

The exported comparison is a displayed gap route between regular Cauchy readback and located Dedekind boundary data, not a quotient stream, primitive cut equality, selected host limit, trichotomy split, countable-choice construction, or ambient completeness theorem.

Proof. Project the public rows from Definition 5906.1. The dyadic ledger D and stream schedule S are read before either comparison side is public.

On the Cauchy-facing side, R reads the finite windows as regular rational observations and E seals that route as a $\mathsf{Real}^{\uparrow}$ boundary. On the cut-facing side, L exposes only located lower-and-upper tests and B compares those tests with the same regular readback.

The judgment row J attaches each comparison request to a displayed gap cell. Thus a consumer can move between the Cauchy and Dedekind presentations only through D, S, R, E, L, B, J.

The structural rows H, C, P, N preserve and name that route. Since no excluded principle appears as a carrier coordinate, the handoff exports only the finite BEDC comparison surface. □

Falsifiable prediction: Every accepted $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ comparison request factors through displayed DyadicRatCore, StreamName, RegSeqRat, RealUp, LocatedCut, DedekindCut-CauchyBoundary, routing judgment, transport, replay, provenance, and local naming rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, located-cut-completion, cauchy-dedekind-boundary: $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the shared dyadic gap ledger, finite observation windows, regular readback, Real seal, located-cut ledger, boundary comparison row, routing judgment, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDedekindGapRouter}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDedekindGapRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5906.1, Definition 5906.2, and Definition 5906.3 seed the finite Cauchy–Dedekind gap route over DyadicRatCore, StreamName, RegSeqRat, RealUp, LocatedCut, DedekindCutCauchyBoundary, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDedekindGapRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, primitive cut equality, selected host limits, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exactness, dyadic gap exactness, Cauchy-side handoff, cut-side handoff, routing determinacy, classifier transport, replay, and non-escape for BEDC.Derived.CauchyDedekindGap

Constructive story: A $\mathsf{CauchyDedekindGapRouter}^{\uparrow}$ packet is generated from DyadicRatCore gap cells, StreamName finite windows, RegSeqRat readback, a RealUp seal, a LocatedCut ledger, a DedekindCutCauchyBoundary comparison row, a routing judgment, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5907

A Concrete Naming Certificate for $\mathsf{SequentCalculus}^{\uparrow}$

Following Gentzen's sequent calculi for structural proof theory, the $\mathsf{SequentCalculus}^{\uparrow}$ packet records sequents, inference rows, structural rules, proof trees, and cut-free boundary data as finite BEDC naming evidence. It sits beside the first-order syntax surface of chapter 197, the model-theory satisfaction packet of chapter 198, and the MetaCIC critical-path route of chapter 765. The packet supplies a proof-calculus substrate for later proof-theoretic consumers; it is not host proof syntax, a quotient of derivations, a global cut-elimination theorem, or a semantic completeness principle.

Definition 5907.1 (Sequent calculus carrier)**.** *A* $\mathsf{SequentCalculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, \Gamma, \Delta, R, T, K, H, C, P, N)$$

with the following displayed rows:

1. *F is the formula-source row, read through the first-order syntax surface before any sequent endpoint is public.*
2. *Γ is the antecedent-context row, listing the finite left-hand formula window of each displayed sequent.*
3. *Δ is the succedent-context row, listing the finite right-hand formula window of each displayed sequent.*
4. *R is the rule-ledger row, naming the initial, logical, and structural rule entries accepted by the packet.*
5. *T is the proof-tree row, recording finite inference trees whose leaves and conclusions are sequents formed from F, Γ, Δ, R.*
6. *K is the cut-boundary row, recording which displayed trees are read as cut-free and which trees still contain an explicit cut occurrence.*
7. *H records componentwise* hsame *transport for formula, context, rule, tree, cut-boundary, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from a proof-calculus request through $F, \Gamma, \Delta, R, T, K, H$.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentCalculus}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{SequentCalculus}^{\uparrow}}$ *row.*
The carrier has no coordinate for host derivation equality, proof-term quotation, quotienting of derivations, semantic validity, completeness, global cut elimination, or normalization.

Lean4 ✓ `BEDC.Derived.SequentCalculusUp.SequentCalculusTasteGate single carrier alignment`

Theorem 5907.2 (Sequent calculus NameCert obligations)**.** *For every accepted* $\mathsf{SequentCalculus}^{\uparrow}$ *carrier* $S = (F, \Gamma, \Delta, R, T, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SequentCalculus}^{\uparrow}}$ *surface consists exactly of formula-source admission, antecedent-context display, succedent-context display, rule-ledger admission, proof-tree readability, cut-boundary visibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every proof-calculus consumer must read the rows* $F, \Gamma, \Delta, R, T, K$ *before exporting any sequent derivation route.*

Proof. Project the carrier of Definition 5907.1. The formula and context rows F, Γ, Δ determine the sequents that may be inspected, while R determines the finite rule ledger available to proof-tree consumers.

Read T only after those rows. A proof tree is public as a displayed finite inference tree over the visible sequents and rule entries, not as host syntax or an unrecorded derivation object.

Finally read K, H, C, P, N. The cut-boundary row keeps cut occurrences explicit, while transport, replay, provenance, and local naming preserve the same packet. Since semantic validity, completeness, quotient derivation equality, and global cut elimination are absent coordinates, the listed rows exhaust the NameCert obligations. □

Theorem 5907.3 (Sequent calculus cut-free scope)**.** *For every accepted* $\mathsf{SequentCalculus}^{\uparrow}$ *carrier* $S = (F, \Gamma, \Delta, R, T, K, H, C, P, N)$*, a cut-free read is scoped to the displayed proof-tree row* T *and the cut-boundary row* K*. If* K *marks a tree as cut-free, downstream consumers may use that tree only through the same formula, context, rule, transport, replay, provenance, and naming rows* $F, \Gamma, \Delta, R, H, C, P, N$*. The packet does not transform arbitrary cut-containing trees into cut-free trees and does not assert global cut elimination.*

Proof. Start with the NameCert obligations of Definition 5907.2. They force every proof-calculus read to pass through the formula source, antecedent and succedent contexts, rule ledger, proof-tree row, and cut-boundary row.

The cut-free claim is therefore a scope claim about an already displayed row of the carrier. A consumer may replay the cut-free marking only together with the same tree and rule ledger; it cannot infer a cut-free replacement for another tree.

Transport and continuation replay through H and C preserve the displayed rows componentwise, while P, N source and name the same packet. Since the carrier contains no global cut-elimination coordinate, no semantic completeness row, and no quotient of derivations, the public cut-free surface remains exactly the finite boundary recorded by K over T. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentCalculus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SequentCalculus}^{\uparrow}$ packet is generated from first-order formula rows, finite antecedent and succedent context rows, a rule ledger, proof-tree rows, a cut-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentCalculus}^{\uparrow}}$, seed)

Scope closed: Definition 5907.1, Definition 5907.2, and Definition 5907.3 seed the finite proof-calculus carrier and its cut-free read boundary for later proof-theoretic consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentCalculus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove cut elimination, proof normalization, semantic completeness, quotient derivation equality, host proof-term adequacy, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rule-ledger exactness, proof-tree readability, cut-boundary visibility, transport, replay, provenance, and local naming for BEDC.Derived.SequentCalculusUp.

Chapter 5908

A Concrete Naming Certificate for $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ packet names the bidirectional finite handoff between regular Cauchy reads and located-cut completion reads on the L10 Real and Completion route. It is anchored in the one-way bridge of chapter 2346, the located-cut completion surface of chapter 5554, the regular rational route of chapter 301, the finite observation discipline of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet records equivalence as two displayed certificate routes, not as quotient equality of completed real names or as an ambient completeness principle.

Definition 5908.1 (Cauchy-Dedekind equivalence carrier)**.** *A* $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R, W, D, L, B, E_0, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row exposed before any cut-facing comparison is consumed.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule controlling every source, comparison, and return read.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger shared by the regular Cauchy side and the located-cut side.*
4. *L is the* $\mathsf{LocatedCutCompletion}^{\uparrow}$ *bracket-completion row, carrying lower and upper located evidence through the same finite windows and dyadic refinements.*
5. *B is the forward* $\mathsf{CauchyDedekindBridge}^{\uparrow}$ *row from R, W, D into the located-cut-facing comparison route.*
6. *E_0 is the reverse located-cut-to-regular-Cauchy row. It reads the displayed L, W, D brackets back into the regular rational route R before any terminal real seal is exposed.*
7. *H records componentwise* hsame *transport for the regular row, window schedule, dyadic ledger, located completion row, forward bridge, reverse row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the two routes*

$$R, W, D \longmapsto B, L \qquad \textit{and} \qquad L, W, D \longmapsto E_0, R.$$

9. *P is the* Pkg *provenance over* $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$, $\mathsf{CauchyDedekindBridge}^{\uparrow}$, $\mathsf{LocatedCutCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyDedekindEquivalence}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying both directions. It has no coordinate for quotient equality of real names, a selected representative, countable choice, trichotomy, primitive Dedekind cuts outside the displayed rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyDedekindEquivalenceUp.CauchyDedekindEquivalenceTasteGate single carrier alignment`

Theorem 5908.2 (Cauchy-Dedekind equivalence NameCert obligations)**.** *For every accepted* CauchyDedekindEquiv *carrier*

$$E = (R, W, D, L, B, E_0, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyDedekindEquivalence}^{\uparrow}}$ *surface consists exactly of carrier admission, regular Cauchy readback, finite-window scheduling, dyadic tolerance compatibility, located-cut completion exposure, forward bridge exposure, reverse located-cut-to-regular-Cauchy exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from the excluded host principles.*

Proof. Project the accepted carrier of Definition 5908.1. The regular side is R, the common finite observation and tolerance surface is W, D, and the located-cut completion side is L.

The forward handoff is exactly the row B. It is read as the bridge from the regular Cauchy data into the located-cut-facing comparison surface already named by chapter 2346.

The reverse handoff is exactly the row E_0. It reads the displayed lower and upper located-cut completion evidence back through the same finite windows and dyadic tolerances into the regular rational route, so it cannot introduce a selected limit or a quotient representative.

Finally H, C, P, N transport, replay, source, and name the two routes as one packet. Since no excluded host principle occurs as a carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 5908.3 (Cauchy-Dedekind equivalence two-way handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ *packet factors through the paired finite routes*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyDedekindBridge}^{\uparrow},\ \mathsf{LocatedCutCompletion}^{\uparrow}$$

and

$$\mathsf{LocatedCutCompletion}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

Thus the packet exports a named two-way handoff between regular Cauchy reads and located-cut completion reads before the terminal $\mathsf{Real}^{\uparrow}$ *seal, without exposing quotient stream equality, selected limits, countable choice, trichotomy, primitive host cuts, or ambient completeness.*

Proof. Open an accepted carrier $E = (R, W, D, L, B, E_0, H, C, P, N)$. The forward route is forced through R, W, D and then through B, L because those are the only rows connecting regular Cauchy readback to located-cut completion evidence.

The reverse route is forced through L, W, D and then through E_0, R. The same finite window schedule and dyadic ledger are reused, so the return direction stays inside displayed packet data.

Rows H and C preserve the two routes under componentwise transport and replay. Rows P and N record the provenance and local naming of the paired handoff. Since the terminal $\mathsf{Real}^{\uparrow}$

seal is downstream of these rows, the consumer receives an equivalence interface rather than an ambient real-completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ consumer that claims a two-way Cauchy–Dedekind handoff while omitting one of $R, W, D, L, B, E_0, H, C, P, N$ refutes the carrier surface of Definition 5908.1.

Independence witness: cauchy-dedekind-bridge, located-cut-completion, regseqrat, streamname, dyadicratcore, real: $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a forward bridge row and a reverse located-cut-to-regular-Cauchy row over the same finite windows, dyadic tolerance ledger, transport, replay, provenance, and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyDedekindEquivalence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyDedekindEquivalence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5908.1, Definition 5908.2, and Definition 5908.3 seed the bidirectional finite handoff between regular Cauchy readback and located-cut completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyDedekindEquivalence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, primitive host cuts, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, forward bridge exposure, reverse located-cut-to-regular-Cauchy exposure, shared finite-window and dyadic tolerance exactness, transport and replay stability, provenance non-escape, and TasteGate field faithfulness for BEDC.Derived.CauchyDedekindEquivalenceUp.

Constructive story: A $\mathsf{CauchyDedekindEquivalence}^{\uparrow}$ packet is generated from RegSeqRat readback, StreamName finite windows, DyadicRatCore tolerances, a LocatedCutCompletion row, a CauchyDedekindBridge row, a reverse located-cut-to-regular-Cauchy row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{LuenbergerObserver}^{\uparrow}$ reads the state-transition row from chapter 329, the observability row from chapter 294, and the matrix and linear-map rows before the local observer correction packet is consumed.

Chapter 5909

A Concrete Naming Certificate for $\mathsf{LuenbergerObserver}^{\uparrow}$

Following Luenberger's observer construction for finite-dimensional linear systems, the $\mathsf{LuenbergerObserver}^{\uparrow}$ packet records a deterministic state-estimate correction row rather than a stochastic filter. It consumes the observed-output surface of $\mathsf{ControlObservability}^{\uparrow}$ in chapter 294, the reachability-side finite-control surface of $\mathsf{ControlControllability}^{\uparrow}$ in chapter 297, the finite dual readback of $\mathsf{ObservabilityGramian}^{\uparrow}$ in chapter 5787, the estimator boundary of $\mathsf{KalmanFilter}^{\uparrow}$ in chapter 5102, and the linear-map carrier of $\mathsf{LinMap}^{\uparrow}$ in chapter 48. The chapter names the displayed observer ledger used by downstream estimator and observer-based-control packets; it does not identify the deterministic observer with a Kalman update.

Definition 5909.1 (LuenbergerObserverUp BHist carrier)**.** *A* $\mathsf{LuenbergerObserver}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, C, Y, G, R, E, H, T, P, N).$$

Here A is the plant-dynamics row, read as the finite transition surface shared with $\mathsf{LinMap}^{\uparrow}$ *and the control-system certificates; C is the output-map row consumed through* $\mathsf{ControlObservability}^{\uparrow}$*; Y is the displayed measured-output and predicted-output comparison row; G is the observer-gain row, with its finite matrix or linear-map provenance separated from any Kalman covariance update; R is the residual row naming $y - C\widehat{x}$ as a finite output discrepancy; E is the error-propagation ledger recording the closed observer error route governed by the displayed plant, output, and gain rows; H is componentwise* hsame *transport for all observer rows; T is displayed* Cont *replay for residual formation and error-propagation reads; P is the* Pkg *provenance over* $\mathsf{LuenbergerObserver}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*; and N is the local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$ *row.*

The classifier compares observer packets only through $A, C, Y, G, R, E, H, T, P, N$. It has no coordinate for stochastic covariance recursion, optimal filtering, asymptotic convergence rate, pole-placement completeness, unobserved trajectory semantics, or a hidden analytic state space.

Theorem 5909.2 (LuenbergerObserverUp NameCert obligations)**.** *For every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *carrier $L = (A, C, Y, G, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$ *surface consists exactly of plant-dynamics admission through A, output-map admission through C, measured/predicted output comparison through Y, observer-gain admission through G, residual formation through R, error-propagation exposure through E, componentwise transport through H, displayed continuation replay through T, package provenance through P, and local naming through N.*

A downstream deterministic estimator, observer-based controller, or control-diagnostic packet may read this surface only after the plant dynamics, output map, observer gain, residual, and error-propagation ledger have all been displayed. The packet exports no Kalman covariance update, no stochastic innovation law, no host convergence theorem, no pole-placement theorem, and no analytic state-trajectory object.

Lean4 ✓ `BEDC.Derived.LuenbergerObserverUp.LuenbergerObserverCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 5909.1. The rows A, C, Y, G, R, E are the observer-facing data, while H, T, P, N transport, replay, source, and name the same finite packet.

The output row is read through $\mathsf{ControlObservability}^{\uparrow}$ and the transition/gain rows through the displayed finite linear-map surface. The observability Gramian and controllability packets can be used only as named provenance for finite control rows; they do not add an unrecorded analytic observer theorem.

The Kalman-filter sibling is present only as an estimator boundary. Since the carrier separates the deterministic gain row G and residual row R from any covariance update, a consumer cannot reinterpret the observer certificate as stochastic filtering data.

Thus the listed rows exhaust the local NameCert surface. Any read that demands a convergence theorem, pole-placement completeness theorem, covariance recursion, or host trajectory semantics leaves the accepted carrier. □

Falsifiable prediction: Every future consumer of $\mathsf{LuenbergerObserver}^{\uparrow}$ must expose the plant row, output row, observer-gain row, residual row, and error-propagation ledger before it reads a deterministic observer handoff; no consumer may obtain a Kalman covariance update from this packet.

Independence witness: controlobservability, controlcontrollability, observability-gramian, kalman-filter, linearmap: none is bijective to $\mathsf{LuenbergerObserver}^{\uparrow}$ because the observer packet jointly carries plant dynamics, output map, observer gain, residual, deterministic error propagation, transport, replay, provenance, and local NameCert rows while excluding stochastic covariance update and host convergence semantics.

Closure status for $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$.

Constructive story: A $\mathsf{LuenbergerObserver}^{\uparrow}$ packet is generated from displayed plant dynamics, output map, observer gain, measured/predicted output comparison, residual, deterministic error-propagation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5909.1 and Definition 5909.2 seed the deterministic observer carrier and its local NameCert obligation surface over displayed control and linear-map rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove observer convergence, pole-placement completeness, stochastic optimality, Kalman covariance update laws, analytic trajectory semantics, Lean verification, or a bridge-checked public schema.

Upgrade path: Obligation closure requires checked carrier admission, plant-dynamics exposure, output-map exposure, observer-gain exactness, residual exactness, error-propagation ledger coverage, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.LuenbergerObserverUp.

Theorem 5909.3 (LuenbergerObserverUp control sibling dependency route)**.** *Every deterministic observer correction read of an accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet consumes the control sibling route through chapter 294, chapter 297, chapter 5787, chapter 5102, and chapter 48 before any observer-interface row is treated as public. The route exposes the plant-dynamics row, output-map row, observer-gain row, residual row, deterministic error-propagation ledger, componentwise* hsame

transport, displayed Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$*; it exports no stochastic covariance update, pole-placement theorem, observer convergence theorem, or hidden analytic state trajectory.*

Proof. Project the accepted carrier of Definition 5909.1. Its rows $A, C, Y, G, R, E, H, T, P, N$ name the deterministic observer interface and the structural transport, replay, provenance, and local naming records.

The output row is read through $\mathsf{ControlObservability}^{\uparrow}$, the finite control provenance through $\mathsf{ControlControllability}^{\uparrow}$, the dual readback through $\mathsf{ObservabilityGramian}^{\uparrow}$, the estimator boundary through $\mathsf{KalmanFilter}^{\uparrow}$, and the transition and gain rows through $\mathsf{LinMap}^{\uparrow}$. These sibling chapters must therefore be displayed before a consumer treats the observer correction as public.

Since the carrier contains no covariance recursion, pole-placement completeness row, convergence theorem, or host trajectory semantics, the public observer-interface read is exactly the displayed control sibling route. □

Theorem 5909.4 (Luenberger observer control-system route)**.** *Every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *read that exposes observer-error feedback data first consumes the control-system state-transition row through* $\mathsf{DynSystem}^{\uparrow}$*, the finite observation row through* $\mathsf{ControlObservability}^{\uparrow}$*, and the state-space carrier through* $\mathsf{StateSpaceModel}^{\uparrow}$*. The public route is the displayed plant dynamics, observation map, state carrier, measured-output comparison, observer gain, residual row, deterministic error-propagation ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$*; it exports no hidden trajectory semantics, stochastic covariance recursion, pole-placement theorem, or convergence claim.*

Proof. Project the carrier of Definition 5909.1. The plant-dynamics row A, output row C, comparison row Y, observer-gain row G, residual row R, and error-propagation row E are displayed before transport, replay, provenance, and local naming are read.

The state-transition part of A is a control-system read through $\mathsf{DynSystem}^{\uparrow}$, while the state carrier is routed through $\mathsf{StateSpaceModel}^{\uparrow}$. The output row C and the observation-facing part of the residual ledger are routed through $\mathsf{ControlObservability}^{\uparrow}$ before any observer-error feedback is public.

The structural rows H, T, P, N preserve exactly those displayed dependencies. Since no carrier coordinate stores a host trajectory, covariance update, pole-placement theorem, or convergence result, every observer-error feedback read factors through the stated control-system route. □

Theorem 5909.5 (LuenbergerObserverUp control-observability forward route)**.** *Every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *deterministic correction read consumes the finite observer rows supplied by chapter 294 and the estimator boundary of chapter 5102 before the residual and error-propagation rows are public. The route keeps the output-map, measured/predicted comparison, gain, residual, and deterministic error ledger attached to the carrier of Definition 5909.1; it exports no covariance recursion, stochastic innovation law, pole-placement theorem, or hidden trajectory semantics.*

Proof. Project the accepted carrier $L = (A, C, Y, G, R, E, H, T, P, N)$ from Definition 5909.1. The public correction read depends on the output row C, comparison row Y, gain row G, residual row R, and error-propagation row E.

The observability sibling supplies the finite row by which C, Y, R, E are inspected as observer-facing data. The Kalman-filter boundary sibling supplies only estimator comparison data, so the deterministic rows remain A, C, Y, G, R, E rather than covariance or innovation coordinates.

The structural rows H, T, P, N preserve the same finite route under transport, replay, provenance, and local naming. Thus the observer packet is routed through control and observability

siblings before its correction interface is public, without becoming a stochastic filter or analytic convergence theorem. □

Theorem 5909.6 (LuenbergerObserverUp Kalman filter boundary route)**.** *Every deterministic observer read of an accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *carrier may cite chapter 5102 only as an estimator boundary. The read must keep the plant row, output row, observer-gain row, residual row, and error-propagation row visible as the carrier coordinates*

$$A, C, G, R, E$$

of Definition 5909.1, together with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$ *naming. It exports no covariance recursion, stochastic innovation law, optimal-filtering theorem, or hidden probabilistic state row.*

Proof. Project the accepted carrier $L = (A, C, Y, G, R, E, H, T, P, N)$. The plant, output, gain, residual, and error-propagation rows are the deterministic observer data, while H, T, P, N preserve and name that finite packet.

The control sibling route of Definition 5909.3 already displays the control, observability, Gramian, Kalman-boundary, and linear-map chapters before the observer correction becomes public. The observability route of Definition 5909.10 keeps the output, residual, gain, and error ledger attached to the same finite carrier.

The Kalman-filter citation is therefore boundary information for estimator comparison. Since no covariance coordinate, stochastic innovation coordinate, or probabilistic state coordinate occurs in L, the deterministic observer route cannot export a Kalman-filter law. □

Theorem 5909.7 (LuenbergerObserverUp Kalman filter dependency boundary)**.** *The* $\mathsf{LuenbergerObserver}^{\uparrow}$ *route consumes* $\mathsf{KalmanFilter}^{\uparrow}$ *only as an estimator-boundary sibling. The plant, output, observer-gain, residual, and error-propagation rows remain deterministic carrier data in* A, C, G, R, E*, and the structural rows* H, T, P, N *preserve that deterministic packet without adding covariance, innovation, probability-space, or optimality coordinates.*

Proof. Use Definition 5909.6. It already places every Kalman-filter citation outside the carrier coordinates that generate the deterministic observer read.

The displayed carrier rows of Definition 5909.1 are therefore the complete dependency surface for plant evolution, output comparison, gain application, residual formation, and error propagation. Since $\mathsf{KalmanFilter}^{\uparrow}$ appears only through the boundary citation, no stochastic row can be imported into the deterministic observer packet. □

Theorem 5909.8 (LuenbergerObserverUp Kalman filter dependency route)**.** *Every* $\mathsf{KalmanFilter}^{\uparrow}$ *or observer-controller consumer of an accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *carrier must first expose the state-space row, observability row, output-error row, observer-gain row, replay row, provenance row, and local* $\mathsf{NameCert}_{\mathsf{LuenbergerObserver}^{\uparrow}}$ *row of the displayed packet. The Kalman-filter side is read only through chapter 5102, and the observation side is read only through the* $\mathsf{ControlObservability}^{\uparrow}$ *and* $\mathsf{ObservabilityGramian}^{\uparrow}$ *rows already named by chapter 294 and chapter 5787. Thus no consumer may treat the deterministic observer correction as a hidden covariance update, stochastic innovation law, selected state trajectory, or host convergence theorem.*

Proof. Project the accepted carrier $L = (A, C, Y, G, R, E, H, T, P, N)$ of Definition 5909.1. The plant and state-space information is carried by A, the observability and output rows by C and Y, the gain and output-error data by G, R, E, and the structural replay by H, T, P, N.

Apply the Kalman boundary route of Definition 5909.6. It keeps every Kalman-filter citation outside the deterministic carrier coordinates that generate the observer read.

Apply the observability dependency route and observer-controller forward route, Definition 5909.10 and Definition 5909.9. These routes force the consumer to display the observation, residual, gain, error-propagation, replay, provenance, and naming rows before any controller or estimator handoff is public.

Since the carrier has no covariance recursion, stochastic innovation, selected trajectory, or host convergence coordinate, the Kalman-filter and observer-controller consumers can read only the displayed deterministic dependency route. □

Theorem 5909.9 (LuenbergerObserverUp observer-controller forward route)**.** *Every downstream deterministic observer-controller read of an accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet must consume the observer only through the displayed plant, output, gain, residual, error-propagation, transport, replay, provenance, and local NameCert rows of Definition 5909.1. The route may then pass to the closest deterministic control-system siblings chapter 294, chapter 297, chapter 5787, and chapter 48; it exports no stochastic covariance update, pole-placement theorem, host convergence theorem, or hidden analytic state trajectory.*

Proof. Project the accepted carrier $L = (A, C, Y, G, R, E, H, T, P, N)$ from Definition 5909.1. The observer-controller read can use only the plant row A, output row C, comparison row Y, gain row G, residual row R, and error-propagation row E before the structural rows are exposed.

Use the control sibling dependency route of Definition 5909.3 and the control-observability forward route of Definition 5909.5. They keep the deterministic control and observability siblings visible before the observer correction is public.

The rows H, T, P, N preserve, replay, source, and name the same finite packet. Since no covariance recursion, pole-placement completeness row, convergence theorem, or host trajectory row occurs in the carrier, every observer-controller consumer remains inside the displayed deterministic route. □

Theorem 5909.10 (LuenbergerObserverUp observability dependency route)**.** *Every observer-error ledger read of an accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet must pass through chapter 294 and chapter 5787 before the residual row and observer-gain row are exposed. Concretely, the read projects the carrier rows*

$$(C, Y, G, R, E)$$

from Definition 5909.1: C supplies the observed-output map, Y compares measured and predicted outputs, G supplies the deterministic observer gain, R names the residual, and E records the observer-error propagation ledger. The route exports no Kalman covariance recursion, pole-placement theorem, or host trajectory semantics.

Proof. Project the accepted carrier $L = (A, C, Y, G, R, E, H, T, P, N)$. The observer-error ledger E can be read only after the output-map row C, the measured/predicted comparison row Y, the gain row G, and the residual row R are visible.

The output-map row is the observability-facing entry point. It is therefore routed through $\mathsf{ControlObservability}^{\uparrow}$ before the observer-error ledger is public, and the finite dual readback required by the same output surface is routed through $\mathsf{ObservabilityGramian}^{\uparrow}$.

After those two sibling routes are displayed, the residual and gain rows may be exposed as the deterministic observer interface. The carrier still has no coordinate for Kalman covariance, pole placement, or host trajectory semantics, so the read remains an observability dependency route rather than a stochastic or analytic observer theorem. □

Theorem 5909.11 (Luenberger observer state-space dependency route)**.** *Every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet consumes the finite state-space carrier of chapter 329, the linear-map evolution row of chapter 48, the observability row routed through Definition 5909.10, the Kalman-filter boundary route of chapter 5102, and the observer-controller forward route of Definition 5909.9 before exporting estimator-error dynamics. The exported read is the displayed tuple of plant dynamics, output map, measured/predicted output comparison, observer gain, residual, deterministic error propagation, transport, replay, provenance, and local naming rows; it is not a hidden analytic state trajectory, stochastic covariance recursion, pole-placement theorem, or host convergence claim.*

Proof. Project the carrier of Definition 5909.1. The plant row A and output row C are meaningful only after the state-space and linear-map substrates have been displayed.

Next read the observability dependency route. It exposes the output-map, residual, observer-gain, and error-ledger rows through the accepted observability and Gramian surfaces before the deterministic observer interface is public.

Finally compose the Kalman-filter boundary route and the observer-controller forward route. These sibling routes keep stochastic covariance and controller synthesis outside the observer carrier while allowing the estimator-error ledger E to be consumed as deterministic state-space data.

The structural rows H, T, P, N transport, replay, source, and name exactly this route. Since the carrier has no coordinate for hidden state trajectories, covariance recursion, pole placement, or host convergence, every estimator-error export factors through the displayed state-space dependency route. □

Every observer-error feedback read of an accepted $\mathsf{LuenbergerObserver}^{\uparrow}$ packet factors through the displayed $\mathsf{StateSpaceModel}^{\uparrow}$ row, $\mathsf{ControlObservability}^{\uparrow}$ row, observer-gain row, residual row, deterministic error-propagation ledger, and the existing anchors chapter 329 and chapter 294 before it can be consumed by estimator or observer-controller routes. The lattice route exports the finite plant, observation, gain, residual, error, transport, replay, provenance, and local naming rows only; it exports no hidden trajectory semantics, stochastic covariance recursion, pole-placement theorem, or host convergence claim.

Theorem 5909.12 (Luenberger observer state-space realization forward link)**.** *Every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet that reads the observer gain, plant state, output row, and innovation replay consumes the finite* $\mathsf{StateSpaceModel}^{\uparrow}$ *and* $\mathsf{TransferFunctionStateSpaceRealization}^{\uparrow}$ *realization rows before any observer-controller or Kalman handoff. The forward link exports only the displayed plant, output, gain, innovation, realization, transport, replay, provenance, and local naming rows; it exports no hidden analytic transfer-function equality, stochastic covariance recursion, pole-placement theorem, or host convergence claim.*

Proof. Project the carrier of Definition 5909.1. The observer gain, plant state, output row, and innovation replay are visible only as displayed finite rows of the accepted packet.

Read the state-space dependency route of Definition 5909.11. It forces the observer packet through the finite $\mathsf{StateSpaceModel}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ surfaces before the estimator-error ledger is public.

Now use the realization surface of chapter 9117. The observer can pass to observer-controller or Kalman handoff rows only after the transfer readback and state-space realization compatibility rows have been displayed.

The structural rows transport, replay, source, and name exactly this route. Since neither the observer packet nor the realization packet contains analytic transfer-function equality, stochastic covariance recursion, pole placement, or host convergence, the forward link remains a finite NameCert dependency. □

Theorem 5909.13 (Luenberger observer transportability dependency route)**.** *Every accepted* $\mathsf{LuenbergerObserver}^{\uparrow}$ *packet used as an externally valid observer-state reconstruction route must first read the selection-domain row of* $\mathsf{Transportability}^{\uparrow}$ *through chapter 9055, the finite state-transition carrier of* $\mathsf{StateSpaceModel}^{\uparrow}$ *through chapter 329, the deterministic control-system rows through Definition 5909.4, the observability row through chapter 294, and the estimator boundary through chapter 5102. Only after those sibling rows are displayed may a consumer treat the observer correction, residual ledger, and error-propagation row as portable between source and target systems. The route exports no hidden domain-selection oracle, stochastic covariance update, host state equality, or ambient external-validity theorem.*

Proof. Project the accepted carrier of Definition 5909.1. Its observer rows $A, C, Y, G, R, E, H, T, P, N$ expose plant dynamics, output, comparison, gain, residual, deterministic error propagation, transport, replay, provenance, and local naming.

The transportability read supplies the source and target selection-domain row before those observer rows are used outside their original system. The state-space, control-system, observability, and Kalman-filter boundary siblings then keep the plant carrier, output route, and estimator boundary visible.

Since the Luenberger carrier contains no domain-selection oracle, host state equality, covariance recursion, or external-validity theorem, observer-state transfer is public only along the displayed transportability and control sibling route. □

Theorem 5909.14 (LuenbergerObserver observer-control lattice route)**.** *Every deterministic* $\mathsf{LuenbergerObserver}^{\uparrow}$ *correction read consumes the observer/control dependency lattice before it exposes the observer row. The route must display* $\mathsf{ControlObservability}^{\uparrow}$ *for the output-map read,* $\mathsf{ControlControllability}^{\uparrow}$ *for the finite control-side reachability boundary,* $\mathsf{ObservabilityGramian}^{\uparrow}$ *for the dual observability ledger,* $\mathsf{KalmanFilter}^{\uparrow}$ *as the stochastic-estimator boundary,* $\mathsf{LinMap}^{\uparrow}$ *for the plant and gain rows, and* $\mathsf{StateSpaceModel}^{\uparrow}$ *for the finite state-space carrier before the rows of Definition 5909.1 are public.*

Proof. Start from the carrier of Definition 5909.1. Its plant, output, observer-gain, residual, and error-propagation rows are not standalone data; each is displayed against a control or state-space sibling.

The observability and controllability packets supply the finite output and control lattice. The observability Gramian records the dual readback, while the Kalman-filter packet marks the estimator boundary that this deterministic observer does not internalize.

The remaining rows come from the displayed linear and state-space carriers. $\mathsf{LinMap}^{\uparrow}$ names the plant and gain maps, and $\mathsf{StateSpaceModel}^{\uparrow}$ supplies the state-space row through which those maps are read. Thus a consumer reaches the public $\mathsf{LuenbergerObserver}^{\uparrow}$ row only after the full observer/control dependency lattice is visible. □

Chapter 5910

A Concrete Naming Certificate for $\mathsf{BinaryTree}^{\uparrow}$

Following Knuth and Cormen–Leiserson–Rivest–Stein, the binary-tree packet treats a tree node as a finite record with at most one left branch and at most one right branch. The certificate is a finite-data neighbor of $\mathsf{List}^{\uparrow}$ (chapter 36), $\mathsf{Tree}^{\uparrow}$ (chapter 119), $\mathsf{Graph}^{\uparrow}$ (chapter 118), and $\mathsf{FinSet}^{\uparrow}$ (chapter 112): it uses list-style finite spines for node positions, graph-style incidence rows for branches, and tree-style rooted transport without importing a host datatype.

5910.1 Carrier definition

Definition 5910.1 (BinaryTree finite carrier)**.** *A* $\mathsf{BinaryTree}^{\uparrow}$ *carrier row is a finite* BHist *packet*

$$\mathsf{BinaryTreePacket}(h)$$

with the following displayed rows:

1. *a root history* r *carried by the same* Pkg *provenance as* h*;*
2. *a finite-node position carrier* $\mathsf{NodePos}(h)$ *with readback* $\mathsf{node}_h : \mathsf{NodePos}(h) \to \mathsf{BHist}$ *and root position* p_r *satisfying* $\mathsf{node}_h(p_r)\mathsf{hsame}r$*;*
3. *partial branch classifiers* left_h *and* right_h *from node positions to node positions, each represented by a finite incidence row rather than by a host function;*
4. *a branch classifier saying that any displayed branch row is either left or right and that a node has at most one displayed left target and at most one displayed right target up to* hsame *transport;*
5. hsame *transport rows for the root, every node position, and every left/right branch row;*
6. Cont *replay rows that read the same packet after transporting along the displayed* hsame *witnesses;*
7. Pkg *provenance for the root, node spine, branch rows, and local certificate rows;*
8. *local* $\mathsf{NameCert}$ *rows for the carried root, node-position spine, and left/right incidence classifiers.*

The carrier exports exactly these finite rows. It does not export a host recursive datatype, a hidden traversal order, or an unrecorded quotient of node labels.

Lean4 ✓ `BEDC.Derived.BinaryTreeUp.BinaryTreeTasteGate single carrier alignment`

5910.2 Obligation surface

Theorem 5910.2 (BinaryTree NameCert obligation surface)**.** *The* $\mathsf{BinaryTree}^{\uparrow}$ *certificate obligations are exactly the carrier, classifier, and ledger obligations displayed in Definition 5910.1. Carrier coverage says that an accepted history exposes a root, a finite node-position spine, finite left/right branch rows,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows. Classifier stability says that two accepted packets classify together precisely when their roots, node positions, and left/right branch rows are transported componentwise by the displayed* hsame *witnesses. Ledger exactness says that branch-choice evidence, finite-node enumeration evidence, and replay evidence remain ledger rows and never become host equality.*

Proof. Read an accepted packet through Definition 5910.1. The carrier projection gives the root, node-position spine, left/right branch incidence rows, transport rows, replay rows, provenance rows, and local naming rows.

For classifier stability, compare two packets componentwise. The root comparison is the displayed hsame row, node comparisons use the finite-position readback, and branch comparisons use the left/right incidence rows transported along the same provenance.

For ledger exactness, isolate the data that justifies the visible rows: branch-choice witnesses, finite enumeration witnesses, and replay witnesses. Since each of these is consumed only to justify or transport a visible row, none is exported as host equality or as a hidden traversal order. This is exactly the carrier/classifier/ledger surface claimed above. □

Theorem 5910.3 (BinaryTree induction readback)**.** *Let* $\mathsf{BinaryTreePacket}(h)$ *be an accepted* $\mathsf{BinaryTree}^{\uparrow}$ *carrier row. If a predicate* Q *on displayed node positions holds at the root position and is stable under each displayed left and right branch row, then* Q *holds at every node position in* $\mathsf{NodePos}(h)$*. Equivalently, structural induction over a binary tree is read back from the finite root and branch rows of the packet.*

Proof. Induct over the displayed finite node-position spine. The root case is the supplied root witness $Q(p_r)$.

For a node reached by a displayed left branch, apply the left-branch stability clause to the predecessor position and transport the resulting witness along the branch classifier row. The right-branch case is identical with the right-branch stability clause.

Every position in the finite packet is either the root position or appears as the target of a finite sequence of displayed left/right branch rows from the root, because the packet is accepted only after the branch incidence ledger has been replayed through the rooted $\mathsf{Tree}^{\uparrow}$ and finite $\mathsf{Graph}^{\uparrow}$ rows. Repeatedly applying the two branch steps therefore yields Q at the chosen position. Componentwise hsame transport carries the witness across equivalent packet presentations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BinaryTree}^{\uparrow}}$**.**

Constructive story: A binary tree is accepted only after a visible finite packet records a root row, finite node-position rows, left and right branch incidence rows, classifier transport, replay, provenance, and local naming rows. The construction reads ordinary binary-tree branching as displayed BHist data: root and branch rows are public, while traversal choices and replay witnesses remain ledger data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BinaryTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: The chapter names the finite carrier, classifier, induction readback, and ledger boundary for binary-tree packets over BHist rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BinaryTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a checked Lean implementation, a standard bridge, balancing invariants, search-order invariants, or host recursive tree equality.

Upgrade path: A checked carrier namespace and finite branch-row induction theorem would raise the verification axis.

Chapter 5911

A Concrete Naming Certificate for $\mathsf{SobolevSpace}^{\uparrow}$

Following Sobolev and Adams–Fournier, the $\mathsf{SobolevSpace}^{\uparrow}$ packet records a finite constructive-analysis surface for functions whose weak derivatives and norm budgets are visible as named BEDC rows. It sits downstream of the weak-derivative interface of chapter 2210, the norm carrier of chapter 89, the metric-completion boundary of chapter 1506, and the real seal of chapter 38. The packet supplies PDE, variational, and compactness-facing consumers with a displayed Sobolev-space carrier rather than a host function quotient, an ambient Banach theorem, or a selected completion point.

Definition 5911.1 (SobolevSpace carrier)**.** *A* $\mathsf{SobolevSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, D, N_0, I, M, R, H, C, P, L)$$

with the following displayed rows:

1. *F is the source-function row, listing the finite function-name windows, domain probes, boundary tags, and scalar readbacks exposed to this Sobolev-space packet.*
2. *D is the* $\mathsf{WeakDerivative}^{\uparrow}$ *row, recording the displayed weak derivative actions, derivative-coordinate tags, test-action windows, and candidate derivative readbacks attached to F.*
3. *N_0 is the* $\mathsf{Norm}^{\uparrow}$ *budget row. It binds the source-function row and each displayed weak-derivative row to the finite integrability, exponent, and seminorm ledger used by the carrier.*
4. *I is the integrability and local-domain budget row, naming the finite measurable-window, power-budget, and comparison data that justify the accepted norm reads.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row, available only after F, D, N_0, I have been displayed and only for the completion route named by this carrier.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for norm values, integrability budgets, weak-derivative action values, and completion-facing estimates.*
7. *H stores componentwise* hsame *transport for source-function, weak-derivative, norm, integrability, completion, Real readback, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from every Sobolev-space request through $F, D, N_0, I, M, R, H, P, L$.*
9. *P is the* Pkg *provenance over* $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. *L is the local* $\mathsf{NameCert}_{\mathsf{SobolevSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, D, N_0, I, M, R, H, C, P, L$. It has no coordinate for quotient equality of functions, a host distribution space, global measurability completion, countable choice, compact embedding, trace theory, PDE solvability, or an ambient Banach-space theorem.

Lean4 ✓ `BEDC.Derived.SobolevSpaceUp.SobolevSpaceTasteGate single carrier alignment`

Theorem 5911.2 (SobolevSpace NameCert obligations)**.** *For every accepted* $\mathsf{SobolevSpace}^{\uparrow}$ *carrier*

$$S = (F, D, N_0, I, M, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{SobolevSpace}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source-function, weak-derivative, norm-budget, integrability, completion-handoff, Real readback, transport, replay, provenance, and local naming rows;*
2. *weak-derivative coverage, requiring every derivative read to pass through the displayed* $\mathsf{WeakDerivative}^{\uparrow}$ *row D before it can contribute to a Sobolev-space norm budget;*
3. *norm and integrability exactness, requiring every norm, exponent, and local-domain budget read to factor through N_0, I before it reaches the* $\mathsf{Real}^{\uparrow}$ *row R;*
4. *completion nonescape, requiring the* $\mathsf{MetricCompletion}^{\uparrow}$ *handoff row M to be downstream of F, D, N_0, I, R and not detachable as a selected completion point;*
5. *structural stability under H, C, P, L, which transports, replays, sources, and names only the displayed finite route.*

No $\mathsf{SobolevSpace}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 5911.1. The analytic data visible to the classifier are exactly the source row F, weak-derivative row D, norm-budget row N_0, integrability row I, completion handoff M, and Real readback R.

The weak-derivative obligation is forced before any norm read. A consumer cannot inspect a derivative budget unless it first reads D, and D exposes only the finite test-action and candidate derivative surface inherited from $\mathsf{WeakDerivative}^{\uparrow}$.

The norm and integrability obligations are then attached to N_0, I, R. Norm values, exponent budgets, and local-domain comparisons reach the Real seal only through those displayed rows, so they cannot be replaced by quotient function equality or global measurability completion.

Finally read H, C, P, L. These rows transport, replay, source, and name the same finite packet, while M remains downstream of the displayed analytic rows. Hence the listed rows exhaust the local NameCert surface. □

Theorem 5911.3 (SobolevSpace completion handoff)**.** *Every completion-facing read exported by an accepted* $\mathsf{SobolevSpace}^{\uparrow}$ *carrier factors through the finite route*

$$F, D, N_0, I \longmapsto R \longmapsto M \longmapsto H, C, P, L.$$

The handoff supplies only displayed source-function windows, weak-derivative actions, norm and integrability budgets, Real readbacks, completion routing, transport, replay, provenance, and local naming. It does not supply a quotient of functions, a selected equivalence class, a global distribution dual, compact embedding, trace theory, PDE solvability, countable choice, or an ambient Banach-space completeness theorem.

Proof. Begin with a completion-facing request for an accepted carrier. The request must first expose the source-function row F and the weak-derivative row D, because those rows determine which function windows and derivative actions can enter the Sobolev norm.

Next the request reads N_0 and I. These rows bind the norm, exponent, and local integrability budgets to the same displayed function and derivative data, and the Real readback row R is the only scalar seal available to the route.

The completion handoff M is read after those rows. Transport and replay through H, C preserve the route, while P, L record provenance and local naming. Since the excluded host objects are not carrier coordinates, no completion-facing consumer can bypass the displayed route. □

Theorem 5911.4 (SobolevSpace dual test-action handoff)**.** *Every dual-facing consumer of an accepted* $\mathsf{SobolevSpace}^{\uparrow}$ *carrier*

$$S = (F, D, N_0, I, M, R, H, C, P, L)$$

must first read $F, D, N_0, I, R, H, C, P, L$ *before it can expose a test-action value or dual budget. Consequently a dual-facing route exported from this packet cannot bypass the displayed source-function windows, weak-derivative actions, norm and integrability budgets, scalar seal, transport, replay, provenance, and local naming.*

Proof. Project the accepted carrier. The source row F names the function windows available to any dual-facing request, and D supplies the weak derivative actions and test-action coordinates that the request may inspect.

Next the route reads N_0 and I. These rows bind the norm, exponent, and integrability budgets to the same source and derivative data; the Real row R is the only scalar seal for action values and budget comparisons.

Finally read H, C, P, L. These rows transport, replay, source, and name the finite route without adding host distribution spaces, quotient function equality, Hahn–Banach extension, countable choice, compact embedding, or PDE solvability. □

Falsifiable prediction: An accepted $\mathsf{SobolevSpace}^{\uparrow}$ completion-facing read that reaches $\mathsf{MetricCompletion}^{\uparrow}$ without first exposing source-function windows, WeakDerivativeUp rows, NormUp and integrability budgets, RealUp readbacks, hsame transport, Cont replay, Pkg provenance, and local naming refutes the carrier surface of Definition 5911.1.

Independence witness: weak-derivative, norm, metric-completion, real: no listed sibling is bijective to $\mathsf{SobolevSpace}^{\uparrow}$ because this packet jointly binds source-function windows, weak-derivative actions, norm and integrability budgets, completion handoff, Real readback, transport, replay, provenance, and local naming in one Sobolev-space carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{SobolevSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SobolevSpace}^{\uparrow}$ packet is generated from source-function windows, WeakDerivativeUp rows, NormUp and integrability budgets, MetricCompletionUp handoff data, RealUp readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SobolevSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5911.1, Definition 5911.2, and Definition 5911.3 seed the finite Sobolev-space carrier downstream of weak derivatives, norm budgets, completion handoff, and Real readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SobolevSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient equality of functions, global measurability completion, distribution duality, compact embedding, trace theory, PDE solvability, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, weak-derivative coverage, norm and integrability exactness, completion nonescape, transport, replay, provenance, and local naming for BEDC.Derived.SobolevSpaceUp.

Chapter 5912

A Concrete Naming Certificate for $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Kelley on metric completion, the MetricCompletionSeparableCor packet records the finite L10 handoff by which countable dense data remains visible to completion consumers. It reads the metric carrier of chapter 57, the indexed dense rows of chapter 1566, the separability surface of chapter 1458, the completion interface of chapter 1506, the dense-image packet of chapter 1985, the dense embedding of chapter 3351, the finite windows of chapter 126, the regular-rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet is a displayed separability-to-completion core; it does not use quotient completion equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or a host dense-image theorem.

Definition 5912.1 (Metric completion separable core carrier)**.** *A* $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, D, W, E, I, J, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, carrying the finite distance comparisons used by dense and completion-facing reads.*
2. *D is the* $\mathsf{CountableDenseSubset}^{\uparrow}$ *and* $\mathsf{SeparableMetric}^{\uparrow}$ *dense enumeration row over M.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the source indices and completion entries inspected by the current consumer.*
4. *E is the* $\mathsf{MetricCompletion}^{\uparrow}$ *entry row, exposing completion names only as displayed finite Cauchy-entry data.*
5. *I is the* $\mathsf{DenseImageCompletion}^{\uparrow}$ *row, recording that displayed source entries approximate the completion read.*
6. *J is the* $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$ *row, tying the dense image to the source metric and separated comparison route.*
7. *Q is the separated comparison row, recording zero-distance and apartness-facing comparisons only through finite displayed tolerances.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational observations opened by W.*

9. *L is the* $\mathsf{Real}^{\uparrow}$ *tolerance ledger, sealing the finite distance and approximation budgets read by R.*
10. *H records componentwise* hsame *transport for the metric, dense enumeration, completion-entry, dense-image, dense-embedding, separated comparison, readback, tolerance, replay, provenance, and naming rows.*
11. *C records displayed* Cont *replay from source metric inspection through dense enumeration windows, completion-entry reads, dense-image exposure, separated comparison, rational readback, and Real tolerance sealing.*
12. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CountableDenseSubset}^{\uparrow}$, $\mathsf{SeparableMetric}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{DenseImageCompletion}^{\uparrow}$, $\mathsf{MetricCompletionDenseEmbedding}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionSeparableCore}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, a selected dense subsequence, countable-choice enumeration, an ambient completed metric object, or Real completeness outside the displayed tolerance ledger.

Lean4 ✓ `BEDC.Derived.MetricCompletionSeparableCoreUp.MetricCompletionSeparableCoreTasteGate`
`single carrier alignment`

Theorem 5912.2 (Metric completion separable core NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$ *carrier $S = (M, D, W, E, I, J, Q, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletio}}$ *surface consists exactly of carrier admission for the source metric row, dense enumeration row, finite stream windows, completion entry row, dense-image row, dense-embedding row, separated comparison row, regular-rational readback, Real tolerance ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing separability read must expose $M, D, W, E, I, J, Q, R, L, H, C, P, N$ before it can be consumed.*

Proof. Project the carrier of Definition 5912.1. The only source and density coordinates are M, D, W, so a separability read cannot inspect completion data before the metric and dense enumeration windows are displayed.

The completion-facing rows are then read in order. The entry row E exposes completion names as finite Cauchy-entry data, while I and J expose the dense-image and dense-embedding routes that keep the source metric visible to completion consumers.

The comparison and readback rows are downstream of the same windows. The row Q records separated comparison data, R reads the regular rational observations opened by W, and L seals the resulting Real tolerance ledger.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the local NameCert surface cannot import quotient completion equality, countable choice, selected limits, ambient Real completeness, or a host dense-image theorem. □

Theorem 5912.3 (Metric completion separable core density handoff)**.** *Let $S = (M, D, W, E, I, J, Q, R, L, H, C, P,$ be an accepted* $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$ *carrier. Every* $\mathsf{MetricCompletion}^{\uparrow}$ *consumer that asks for separability of the completion receives the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{CountableDenseSubset}^{\uparrow},\ \mathsf{SeparableMetric}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{DenseImageCompletion}^{\uparrow},\ \mathsf{M}$$

Thus the completion-side dense read is generated by displayed source dense data, finite observation windows, regular-rational readback, Real tolerance ledgers, dense-image exposure, and dense-embedding comparison rows, not by quotient representatives, selected completion points, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing density request. The source of dense data is D over the metric row M, and the only observations available to the request are the finite stream windows W.

Read those windows through R and L. Regular-rational readback supplies the finite rational observations, and the Real tolerance ledger records the approximation budgets used by the completion-side consumer.

The handoff to completion is through E, I, J, Q. The completion entry row opens the finite target entry, the dense-image row records the source approximation, the dense-embedding row keeps the source and target tied to one metric route, and the separated comparison row records only displayed tolerances.

The structural rows H, C, P, N preserve this route. Consequently the completion consumer receives a finite separability handoff from the source dense enumeration to the completion read, with no coordinate for quotient equality, selected limits, countable-choice enumeration, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$ completion-facing density read factors through the source metric, countable dense schedule, finite StreamName windows, completion entry row, dense-image row, dense-embedding row, separated comparison, RegSeqRat readback, Real tolerance ledger, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: metric, countable-dense-subset, separablemetric, metriccompletion, dense-image-completion, metric-completion-dense-embedding, streamname, regseqrat, real: MetricCompletionSeparableCo is not bijective to any listed sibling because it carries the complete separability-to-completion handoff in one packet, with dense enumeration, completion entry, dense image, separated comparison, readback, and tolerance rows visible together.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionSeparableCore}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionSeparableCore}^{\uparrow}$ packet is generated from a source Metric row, CountableDenseSubset and SeparableMetric dense enumeration rows, finite StreamName windows, MetricCompletion entry rows, DenseImageCompletion and MetricCompletionDenseEmbedding rows, separated comparison rows, RegSeqRat readback, Real tolerance ledgers, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionSeparableCore}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5912.1, Definition 5912.2, and Definition 5912.3 seed the finite separability-to-completion route from countable dense source data to completion-facing dense consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionSeparableCore}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense enumeration window replay, completion-entry routing, dense-image exactness, separated comparison stability, regular-rational readback, Real tolerance exactness, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionSeparableCoreUp.

Chapter 5913

A Concrete Naming Certificate for $\mathsf{BaireMeasurableFunction}^{\uparrow}$

Following classical descriptive set theory, the $\mathsf{BaireMeasurableFunction}^{\uparrow}$ packet records a real-facing Baire-measurable map by finite inverse-image ledgers for Baire-cylinder generators. It sits between the cylinder rows of chapter 1685, the pointwise-limit ledger of chapter 1604, the prefix source of chapter 334, and the measurable preimage discipline of chapter 3461. The packet names generator-level Baire measurability only through displayed preimage rows; it does not add sigma-algebra completion, quotient equality of functions, a selected representative, or a choice principle.

Definition 5913.1 (BaireMeasurableFunction carrier)**.** *A* $\mathsf{BaireMeasurableFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, B, G, R, O, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-context row, read through* $\mathsf{BaireSpace}^{\uparrow}$ *prefixes or through a real-facing Baire source supplied by* $\mathsf{RealBaireCylinder}^{\uparrow}$*.*
2. *Y is the target real-observation row, exposing only displayed real-value windows and Baire-cylinder target events.*
3. *B is the Baire-cylinder generator ledger, listing the target* $\mathsf{RealBaireCylinder}^{\uparrow}$ *rows whose inverse images are inspected.*
4. *G is the graph-observation row tying displayed source windows to displayed target real windows.*
5. *R is the inverse-image ledger assigning each displayed generator in B to a displayed source event row over X.*
6. *O is the Baire-one handoff row, recording when the same finite preimage route is supplied by a* $\mathsf{BaireOneFunction}^{\uparrow}$ *approximation ledger.*
7. *H records componentwise* hsame *transport for the source, target, generator, graph, inverse-image, handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a target generator request through B, R, X, G, O.*
9. *P is the* Pkg *provenance over* $\mathsf{BaireMeasurableFunction}^{\uparrow}$*,* $\mathsf{RealBaireCylinder}^{\uparrow}$*,* $\mathsf{BaireOneFunction}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{MeasurableFunction}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireMeasurableFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a completed sigma algebra, countable union closure beyond displayed generator ledgers, almost-everywhere equality, selected representatives, or ambient function-space equality.

Lean4 ✓ `BEDC.Derived.BaireMeasurableFunctionUp.BaireMeasurableFunctionTasteGate single carrier alignment`

Theorem 5913.2 (BaireMeasurableFunction NameCert obligations)**.** *For every accepted* $\mathsf{BaireMeasurableFunction}^{\uparrow}$ *carrier* $F = (X, Y, B, G, R, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BaireMeasurableFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for* F*, target Baire-cylinder generator admission through* B*, inverse-image ledger exposure through* R*, source-event readback through* X*, graph-observation compatibility through* G*, Baire-one handoff compatibility through* O*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. Every measurable-map sibling read must factor through the displayed generator-to-preimage route before it may cite* $\mathsf{MeasurableFunction}^{\uparrow}$*.*

Proof. Project the carrier of Definition 5913.1. The target generators available to the packet are exactly the rows in B, and each such row is a finite $\mathsf{RealBaireCylinder}^{\uparrow}$ read rather than an ambient Borel family.

Read the inverse-image ledger R. It assigns each displayed target generator to a displayed source event over X, while the graph row G supplies the finite observation compatibility between source windows and target real windows.

The Baire-one row O is only a handoff for the same displayed route. It can replay a $\mathsf{BaireOneFunction}^{\uparrow}$ approximation ledger, but it cannot add a separate function-space representative or a sigma-algebra completion.

Finally replay H, C, P, N. These structural rows transport, replay, source, and name the same finite generator ledger. Since no other coordinate is present, the NameCert surface exports no host inverse-image operation, no completion principle, no quotient of functions, and no choice of representatives. □

Theorem 5913.3 (Baire measurable generator route)**.** *Every accepted target-generator read of a* $\mathsf{BaireMeasurableFunction}^{\uparrow}$ *packet factors through the finite route*

$$B \longmapsto R \longmapsto X, G, O \longmapsto H, C, P, N.$$

Thus the public Baire-measurability claim is an inverse-image ledger for displayed Baire-cylinder generators, not a sigma-algebra completion theorem and not an ambient measurable-function quotient.

Proof. Start with a target request. By Definition 5913.1, the request enters through the displayed generator ledger B and not through a host collection of all Baire sets.

The inverse-image row R then supplies the source-side event row. The source context X and graph row G keep the source and target observations paired, while O records only the Baire-one handoff cases that use the same finite route.

The structural rows H, C, P, N preserve and replay this route. Therefore every public read remains a finite generator-ledger read and cannot export sigma-algebra completion, choice of representatives, almost-everywhere equality, or ambient function equality. □

Falsifiable prediction: A $\mathsf{BaireMeasurableFunction}^{\uparrow}$ consumer that claims Baire measurability without displaying target Baire-cylinder generators, inverse-image rows, source-event readback,

graph compatibility, componentwise hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 5913.1.

Independence witness: realbairecylinder, baireonefunction, bairespace, measurablefunction: $\mathsf{BaireMeasurableFunction}^{\uparrow}$ is not bijective to any listed sibling because it binds target Baire-cylinder generators, inverse-image ledgers, source prefix rows, graph observations, Baire-one handoff rows, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BaireMeasurableFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireMeasurableFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5913.1, Definition 5913.2, and Definition 5913.3 seed the finite Baire-measurable inverse-image ledger over displayed Baire-cylinder generators.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireMeasurableFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close sigma-algebra completion, quotient equality of functions, almost-everywhere equality, selected representatives, full Borel measurability, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, Baire-cylinder generator exposure, inverse-image exactness, graph compatibility, Baire-one handoff compatibility, componentwise transport, and measurable-function sibling nonescape for BEDC.Derived.BaireMeasurableFunctionUp.

Constructive story: A $\mathsf{BaireMeasurableFunction}^{\uparrow}$ packet is generated from source-context, target-observation, Baire-cylinder generator, graph-observation, inverse-image, Baire-one handoff, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Chapter 5914

A Concrete Naming Certificate for $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$

The $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ packet realizes the automated theory-discovery pipeline of chapter 9410 as a finite NameCert surface. It sits beside theorem-DNA scoring in chapter 9408, theorem-genome lineage in chapter 9409, theory-stability admission in chapter 9459, and the compiler audit inventory of chapter 9407. The packet names discovery only when finite theory-state, automation-record, genome, score, lineage, classifier-shift, gate, and refusal rows are visible.

Definition 5914.1 (DiscoveryPipelineCalculus carrier)**.** *A* $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (T, A, G, S, L, Q, K, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the certified theory-state row, displaying source, pattern, ledger, classifier, and semantic NameCert coordinates.*
2. *A is the automation-record row, listing generated material without identifying raw record growth with discovery.*
3. *G is the theorem-genome extractor row, exposing theorem-DNA completeness before a theorem-like object is scored or selected.*
4. *S is the score-certificate row, carrying public weights, budget values, and the displayed comparison used by the pipeline.*
5. *L is the lineage ledger row, recording parent, child, and transition-kind edges.*
6. *Q is the classifier-shift row, separating classifier change from record growth, score compression, and recombination.*
7. *K is the discovery-gate row, admitting only claims whose kind matches the displayed evidence.*
8. *R is the refusal and demotion row, naming unpaid discovery, DNA-incomplete, unscored, unledgered, and prose-only attempts as non-admitted.*
9. *H records componentwise* hsame *transport for the state, automation, genome, score, lineage, classifier, gate, refusal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a pipeline claim through A, G, S, L, Q, K, R and back to the certified theory-state row T.*

11. *P is the* Pkg *provenance over* $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$*, chapter 9410, chapter 9408, chapter 9409, chapter 9459, chapter 9407,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DiscoveryPipelineCalculus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for self-certified discovery, prose-only novelty, raw output volume, hidden scoring weights, unledgered lineage, or a carrier map that bypasses classifier-shift evidence.

Theorem 5914.2 (DiscoveryPipelineCalculus NameCert obligations)**.** *For every accepted* DiscoveryPipelineCalculus *carrier $D = (T, A, G, S, L, Q, K, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DiscoveryPipelineCalculus}^{\uparrow}}$ *surface consists exactly of certified theory-state admission through T, automation-record exposure through A, theorem-genome extraction through G, score-certificate exposure through S, lineage-ledger exposure through L, classifier-shift evidence through Q, discovery-gate admission through K, refusal and demotion through R, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. A discovery claim is admitted only after classifier-shift evidence and the gate row are both visible.*

Proof. Project the carrier of Definition 5914.1. The certified theory-state row T fixes the state coordinates, while A records generated material without changing the classifier by itself.

The theorem-genome and scoring rows G, S expose the theorem-DNA completeness and public weight account needed for scored claims. The lineage row L records the parent-child route, so selection cannot be read as an unledgered assertion.

The classifier-shift row Q is the discovery-facing boundary. The gate row K consumes Q together with G, S, L and the certified state data before a discovery claim is admitted, while R records the non-admitted cases.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row identifies raw record growth, score reduction, or prose novelty with discovery, the local NameCert surface is exactly the listed finite ledger. □

Theorem 5914.3 (Discovery admission requires classifier shift)**.** *Every accepted discovery read from a* $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ *packet factors through*

$$A, G, S, L, Q, K, R \longmapsto H, C, P, N.$$

In particular, classifier-shift rows Q, not raw automation-record growth in A, are necessary for discovery admission.

Proof. Read the automation-record row A first. It lists generated material, but by Definition 5914.1 it has no coordinate that changes the certified classifier state.

The genome, score, and lineage rows G, S, L supply the audit account for theorem-like objects and their transitions. These rows make a claim reproducible, but they still do not identify record growth with discovery.

The discovery-facing test reaches the classifier-shift row Q and then the gate row K. Only after Q is visible can K admit a discovery claim; the refusal row R records attempts that omit the shift, score, lineage, genome, certificate, or boundary evidence.

The structural rows H, C, P, N preserve this dependency order. Thus discovery admission requires classifier-shift evidence and cannot be obtained from raw record growth alone. □

Theorem 5914.4 (DiscoveryPipelineCalculus audit-map route)**.** *Every accepted discovery claim in a* $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ *packet $D = (T, A, G, S, L, Q, K, R, H, C, P, N)$ routes its classifier-shift row Q and discovery-gate row K through the concrete audit-map surfaces of chapter 836 and*

chapter 806 before it is read as a pipeline admission. The audit-map route is a concrete NameCert *dependency carried by P and replayed by C, not a prose-only methodology reference.*

Proof. Project the carrier of Definition 5914.1. The classifier-shift row Q and the discovery-gate row K are read only after the certified theory-state, automation, genome, score, and lineage rows T, A, G, S, L have been displayed.

Apply the admission theorem Definition 5914.3. It forces every accepted discovery read to pass through Q, K, R and then through the structural rows H, C, P, N.

Now read the provenance row P and replay row C against the concrete siblings. The Ground-Compiler audit-map packet of chapter 836 supplies compiler-facing theorem inventory and audit rows, while the audit-map interface of chapter 806 supplies the common audit-map surface. Thus classifier-shift and gate admission are routed through concrete NameCert dependencies before the pipeline reads the claim as admitted. □

Falsifiable prediction: A $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ admission trace that accepts discovery from automation-record growth alone, without a classifier-shift row and discovery-gate row, refutes the carrier of Definition 5914.1.

Independence witness: theorem_dna_complexity_scoring, theorem_genome_evolution, theory_stability_taste_gate, groundcompiler_audit_map: $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ is not bijective to these siblings because it jointly binds certified theory-state rows, automation records, theorem-genome extraction, public score certificates, lineage, classifier-shift evidence, discovery-gate admission, refusal ledgers, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DiscoveryPipelineCalculus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiscoveryPipelineCalculus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5914.1, Definition 5914.2, and Definition 5914.3 seed the finite admission calculus for automated discovery claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiscoveryPipelineCalculus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close automated theorem proving, self-certified discovery, proof generation, hidden scoring, unledgered lineage, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, theorem-genome extraction, public scoring, lineage acyclicity, classifier-shift admission, gate soundness, refusal exactness, and TasteGate compatibility for BEDC.Derived.DiscoveryPipelineCalculusUp.

Constructive story: A $\mathsf{DiscoveryPipelineCalculus}^{\uparrow}$ packet is generated from certified theory-state, automation-record, theorem-genome, score-certificate, lineage, classifier-shift, discovery-gate, refusal, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows.

Chapter 5915

A Concrete Naming Certificate for $\mathsf{LipschitzRetraction}^{\uparrow}$

Following Banach and Bishop–Bridges metric fixed-point and retraction theory, the $\mathsf{LipschitzRetraction}^{\uparrow}$ packet records the finite certificate by which a metric source retracts onto a displayed subcarrier through a Lipschitz graph and an identity-on-subcarrier row. It sits on the L10 metric and complete-space boundary beside the metric source of chapter 57, the complete-service surface of chapter 128, the continuous-map graph route of chapter 130, and the located-apartness Real comparison route of chapter 3564. The packet names a constructive finite retraction certificate; it does not assert a selected nearest point, quotient equality of subspaces, ambient completeness of the source, or a classical choice principle.

Definition 5915.1 (Lipschitz retraction carrier)**.** *A* $\mathsf{LipschitzRetraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, K, G, I, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Met}^{\uparrow}$ *source row supplying all source-distance witnesses.*
2. *A is the subcarrier row, displaying membership, inclusion into X, and the inherited distance reads used by the retraction.*
3. *K is the Lipschitz-constant ledger, recording a rational bound together with the source and target distance reads compared by that bound.*
4. *G is the retraction graph row, whose source observations lie in X and whose values are admitted by A.*
5. *I is the retraction-identity row, requiring every point already admitted by A to replay through G as the same displayed subcarrier point.*
6. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row available to consumers that later read Cauchy or endpoint data on the subcarrier.*
7. *L is the* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *comparison row used when a distance inequality is inspected through located Real and completion readbacks.*
8. *E is the* $\mathsf{ContinuousMap}^{\uparrow}$ *replay row tying the graph G to the finite continuity route induced by the Lipschitz ledger K.*

9. *H records componentwise* hsame *transport for the metric source, subcarrier, Lipschitz ledger, graph, identity, complete-metric service, located-apartness comparison, continuous-map replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a retraction request through $X, A, K, G, I, M, L, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{LipschitzRetraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LipschitzRetraction}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for a selected best approximation, quotient subspace equality, ambient completeness theorem, global fixed-point theorem, or classical real-order decision.

Lean4 (definition) `BEDC.Derived.LipschitzRetractionUp.LipschitzRetractionUp`
Lean4 ✓ `BEDC.Derived.LipschitzRetractionUp.LipschitzRetractionTasteGate single carrier alignment`

Theorem 5915.2 (Lipschitz retraction NameCert obligations)**.** *For every accepted* $\mathsf{LipschitzRetraction}^{\uparrow}$ *carrier $R = (X, A, K, G, I, M, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LipschitzRetraction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed tuple, metric-source visibility through X, subcarrier admission and inherited-distance visibility through A, Lipschitz-constant readback through K, graph admission through G, retraction identity through I, complete-metric service access through M, located-apartness Real comparison through L, continuous-map replay through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5915.1. The displayed tuple lists exactly the metric source, subcarrier, Lipschitz ledger, graph, identity, complete-service, located-apartness, continuous-map, transport, replay, provenance, and naming rows.

The source-facing obligations are exhausted by X, A, K, G, I. The metric row supplies distance witnesses, the subcarrier row supplies admissible values, the Lipschitz ledger supplies the bound, the graph row supplies the finite retraction observations, and the identity row forces admitted subcarrier points to replay unchanged.

The consumer-facing obligations are exhausted by M, L, E. A complete-space consumer can inspect only the displayed $\mathsf{CompleteMetric}^{\uparrow}$ service row, a Real comparison can inspect only the $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ row, and a continuity consumer can inspect only the $\mathsf{ContinuousMap}^{\uparrow}$ replay induced by the Lipschitz ledger.

Finally H, C, P, N transport, replay, package, and name this same finite route. Since the carrier has no selected nearest point, quotient equality, ambient completeness theorem, global fixed-point theorem, or classical comparison branch, no such obligation is present in the local certificate. □

Theorem 5915.3 (Lipschitz retraction metric factorization)**.** *Every* $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *or* $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *consumer of an accepted* $\mathsf{LipschitzRetraction}^{\uparrow}$ *carrier factors through the finite route*

$$X, A \longmapsto K, G, I \longmapsto E, L, M \longmapsto H, C, P, N.$$

Thus source distances are read in the ambient metric row X, subcarrier values are admitted through A, the Lipschitz graph and identity rows are checked before any continuous-map or complete-service handoff, and located Real inequalities are exposed only through the displayed $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ *comparison route. The factorization exports a Lipschitz retraction certificate, not a choice-selected projection, quotient subspace equality, or ambient completeness claim.*

Proof. Start with the NameCert obligations of Definition 5915.2. Any consumer first receives the metric source row X and the subcarrier row A, since these are the only rows that can supply source distances and admitted retraction values.

The retraction data then pass through K, G, I. The Lipschitz ledger bounds graph distances, the graph row records the finite observations, and the identity row verifies that points already in the subcarrier replay as themselves.

Only after those rows are displayed can a consumer read E, L, M. The continuous-map row is induced by the same Lipschitz ledger, the located-apartness comparison row reads Real-facing inequalities through the finite readback bridge, and the complete-metric service row is a displayed downstream service rather than an ambient completeness theorem.

Transport, replay, provenance, and local naming are then carried by H, C, P, N. These rows preserve the factorization but add no selected projection, quotient equality, host equality, or classical branch. □

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzRetraction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzRetraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5915.1, Definition 5915.2, and Definition 5915.3 seed the finite Lipschitz retraction route over metric source, subcarrier, graph, identity, continuous-map replay, located-apartness comparison, and complete-service rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzRetraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected nearest points, quotient subspace equality, ambient completeness, global fixed-point existence, Lean verification, bridge checking, or classical real-order decision.

Upgrade path: Obligation closure requires checked carrier admission, metric-source visibility, subcarrier admission, Lipschitz-ledger readback, graph admission, retraction identity, continuous-map replay, located-apartness comparison, complete-metric service access, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzRetractionUp.

Constructive story: A $\mathsf{LipschitzRetraction}^{\uparrow}$ packet is generated from a MetricSpace source row, a subcarrier row, a Lipschitz-constant ledger, a retraction graph, a retraction identity row, a CompleteMetric service row, a LocatedApartnessRealBridge comparison row, a ContinuousMap replay row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

$\mathsf{FiniteStream}^{\uparrow}$ routes finite request rows from chapter 36 into the finite-window discipline of chapter 126, before RegSeqRat and Real-facing consumers inspect the child files below.

Chapter 5916

A Concrete Naming Certificate for $\mathsf{FiniteStream}^{\uparrow}$

Following Bishop and Bridges on finite sequences and finite observation records, the $\mathsf{FiniteStream}^{\uparrow}$ packet records the finite schedule object that sits between $\mathsf{List}^{\uparrow}$ spine data and the countable-name windows of $\mathsf{StreamName}^{\uparrow}$. It packages only the finitely requested observations needed by downstream regular-rational and real-name consumers. The chapter is a finite observation-schedule neighbour of chapter 36, chapter 126, chapter 301, chapter 291, and chapter 38; it does not export an infinite stream, quotient stream equality, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5916.1 (FiniteStream carrier)**.** *A* $\mathsf{FiniteStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, L, W, R, D, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite request row in* BHist*, recording the ordered observation indices and payload requests actually inspected by the consumer.*
2. *L is the* $\mathsf{List}^{\uparrow}$ *prefix spine that stores those requests as a finite constructor-generated list rather than as a host array or hidden stream segment.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window handoff row, exposing exactly the window generated from Q and L.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary available to regular-rational consumers only after the finite window W has been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the displayed dyadic radii, comparison cells, and finite precision budget attached to the window.*
6. *H records componentwise* hsame *transport for the request row, list prefix spine, StreamName window, RegSeqRat readback boundary, DyadicRatCore tolerance row, replay row, provenance row, and local naming row.*
7. *C records displayed* Cont *replay from a finite request through Q, L, W, R, D, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteStream}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FiniteStream}^\uparrow}$ *row.*
The classifier compares accepted packets only through the rows $Q, L, W, R, D, H, C, P, N$. No coordinate stores a completed stream, host stream equality, selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 5916.2 (FiniteStream NameCert obligations)**.** *For every accepted* $\mathsf{FiniteStream}^\uparrow$ *carrier*

$$F = (Q, L, W, R, D, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteStream}^\uparrow}$ *surface consists exactly of carrier admission for the finite request row,* $\mathsf{List}^\uparrow$ *prefix spine,* $\mathsf{StreamName}^\uparrow$ *finite-window handoff,* $\mathsf{RegSeqRat}^\uparrow$ *readback boundary,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The finite-window exactness clause requires W to expose only the observations listed by Q and represented by L. The StreamName handoff clause requires every downstream countable-name read to pass through W before it can reach R or D. The RegSeqRat and Real nonescape clause requires every regular-rational or* $\mathsf{Real}^\uparrow$*-facing consumer to remain downstream from $Q, L, W, R, D, H, C, P, N$. The provenance exhaustion clause requires P and N to name only this finite route.*

Lean4 ✓ `BEDC.Derived.FiniteStreamUp.FiniteStreamCarrier namecert obligations`

Proof. Project the carrier of Definition 5916.1. Its displayed inventory is exactly the finite request row, the finite list spine, the StreamName window, the regular-rational readback boundary, the dyadic tolerance row, and the structural rows H, C, P, N.

Finite-window exactness is read from Q, L, W. The request row lists the observations, the list spine represents them as finite constructor data, and the StreamName handoff exposes precisely that finite window.

The regular-rational and dyadic rows are downstream. A consumer may reach the $\mathsf{RegSeqRat}^\uparrow$ readback boundary R or the $\mathsf{DyadicRatCore}^\uparrow$ tolerance row D only after the finite window W has already been displayed.

Transport, replay, and provenance preserve the same route. Since the carrier contains no completed stream, quotient stream equality, selected limit, countable-choice row, trichotomy branch, or ambient $\mathsf{Real}^\uparrow$ completeness coordinate, no Real-facing consumer can escape the finite packet. □

Theorem 5916.3 (FiniteStream section sibling lock)**.** *Every* $\mathsf{Real}^\uparrow$*-facing consumer of an accepted* $\mathsf{FiniteStream}^\uparrow$ *packet reads the* $\mathsf{StreamName}^\uparrow$ *window W,* $\mathsf{RegSeqRat}^\uparrow$ *boundary R,* $\mathsf{DyadicRatCore}^\uparrow$ *tolerance row D, and downstream* $\mathsf{Real}^\uparrow$ *seal rows only over the same finite request row Q and* $\mathsf{List}^\uparrow$ *spine L. In particular, no sibling packet can replace Q or L while preserving the displayed route*

$$Q, L, W, R, D, H, C, P, N.$$

Proof. Project the carrier of Definition 5916.1. The rows W, R, D are not free-standing analytic objects; they are displayed after the finite request row Q and the finite list spine L.

The displayed Cont replay row C then fixes the request coordinate used by the downstream consumer. Replaying the packet with a different finite request row would leave the listed carrier rows and hence would leave the accepted $\mathsf{FiniteStream}^\uparrow$ packet.

The componentwise hsame transport row H preserves the same coordinate rather than identifying it with an unlisted sibling. Thus any $\mathsf{Real}^\uparrow$-facing read that reaches W, R, D or a Real seal remains locked to the original Q, L route named by P, N. □

Theorem 5916.4 (FiniteStream StreamName dependency route)**.** *Every accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet consumed by a* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *reader must expose the finite request row* Q*, the* $\mathsf{List}^{\uparrow}$ *prefix spine* L*, and the* $\mathsf{StreamName}^{\uparrow}$ *window* W *before the read can reach the downstream rows*

$$R, D, H, C, P, N.$$

The substantive dependency edge is therefore the finite request/list route into the StreamName window of chapter 126, as displayed by the carrier of Definition 5916.1.

The same finite request row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{Real}^{\uparrow}$ seal boundary are consumed by $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ when equality of regular real names is read through finite tail data in chapter 5940.

Proof. Project the carrier of Definition 5916.1. Its first exposed consumer rows are Q, L, W, and the remaining rows R, D, H, C, P, N are displayed only as downstream readback, tolerance, transport, replay, provenance, and naming data.

Use Definition 5916.3. A Real-facing read that reaches W, R, D or a Real seal remains locked to the original Q, L route, so a StreamName, RegSeqRat, DyadicRatCore, or Real reader cannot replace the finite request/list surface with an unlisted sibling.

Thus the dependency is not an ambient infinite-stream dependency. It is the displayed finite route from Q and L into the $\mathsf{StreamName}^{\uparrow}$ window W, after which the downstream rows may be consumed.

The same finite request/list/window route is the route consumed by chapter 8396 before cluster-witness readback can reach $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$. That consumer therefore receives the displayed Q, L, W boundary as an actual dependency row, not as a replaceable narrative shortcut. □

Theorem 5916.5 (FiniteStream StreamName route)**.** *Every* $\mathsf{StreamName}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet must display the finite request row* Q*, the* $\mathsf{List}^{\uparrow}$ *prefix spine* L*, and the finite-window row* W *before it can reach any* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *row. The only admissible route is*

$$Q, L, W \longmapsto R, D, H, C, P, N,$$

so the downstream regular-rational, dyadic-tolerance, and real-seal rows remain locked to the same finite request/list packet.

Proof. Project the carrier of Definition 5916.1. The rows Q, L, W are the finite request, list-spine, and StreamName-window data; the rows R, D, H, C, P, N are downstream readback, tolerance, transport, replay, provenance, and naming data.

Apply Definition 5916.3. It prevents a downstream $\mathsf{Real}^{\uparrow}$-facing read from replacing Q or L while keeping the displayed route.

Now apply Definition 5916.4. It forces any StreamName, RegSeqRat, DyadicRatCore, or Real reader to pass through Q, L, W before the downstream rows are consumed. Therefore every StreamName consumer sees the finite route rather than an ambient infinite stream or an unlisted sibling packet. □

Theorem 5916.6 (FiniteStream Real-seal forward link)**.** *Every* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet must read the finite request row* Q*,* $\mathsf{List}^{\uparrow}$ *spine* L*,* $\mathsf{StreamName}^{\uparrow}$ *window* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary in that order, with transport, replay, provenance, and local naming supplied*

only by H, C, P, N. The downstream sibling route is the regular-tail equality surface of chapter 5940, which consumes finite request windows rather than an ambient completed stream.

Proof. Project the carrier of Definition 5916.1. Its Real-facing rows are downstream from the displayed finite request, list spine, StreamName window, regular-rational readback, and dyadic tolerance rows.

Use the StreamName route of Definition 5916.5. It forces every StreamName-facing read to expose Q, L, W before R, D, H, C, P, N can be consumed.

Apply the finite-window factorization of Definition 5916.11. The factorization identifies the same ordered route as the only admissible path toward RegSeqRat or Real-facing finite-window data. Therefore a $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer receives a finite request-window seal, not quotient equality of real names, a selected limit, countable choice, or ambient Real completeness. □

Theorem 5916.7 (FiniteStream prefix-window Real handoff)**.** *An accepted* $\mathsf{FiniteStream}^{\uparrow}$ *prefix packet can be consumed by a Real-facing route only after the* $\mathsf{List}^{\uparrow}$ *finite row L, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row W, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger D, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary have all been displayed over the same finite request row Q. The handoff preserves componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *naming through H, C, P, N, and it exports no infinite stream, quotient stream equality, selected limit, countable choice, trichotomy, or ambient Real completeness.*

Proof. Project the carrier of Definition 5916.1. Its Real-facing route is the displayed packet $Q, L, W, R, D, H, C, P, N$, with the Real seal available only downstream from the finite request, list spine, StreamName window, regular readback, and dyadic tolerance rows.

Use the pointwise stream forward route, Definition 5916.8. It forces any pointwise or StreamName consumer of the finite prefix to expose Q, L, W before readback can reach RegSeqRat, DyadicRatCore, or Real-facing rows.

Apply the Real-seal forward link, Definition 5916.6, and the finite-window factorization, Definition 5916.11. Together they keep R and D downstream from the finite prefix window and keep the terminal Real boundary downstream from the displayed readback route. Thus a Real-facing consumer receives a finite prefix-window handoff rather than an ambient completed stream or selected limit. □

Theorem 5916.8 (FiniteStream pointwise stream forward route)**.** *Every pointwise stream consumer of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet, in particular a* $\mathsf{PointwiseStreamName}^{\uparrow}$ *reader of chapter 6084, must first expose the finite request row Q, the* $\mathsf{List}^{\uparrow}$ *spine L, and the* $\mathsf{StreamName}^{\uparrow}$ *finite window W. Only after that displayed window route is public may the readback reach* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *rows.*

Proof. Project the carrier of Definition 5916.1. Its finite observation surface begins with Q, L, W, so a pointwise reader has no entry point into the downstream rows before the request row, list spine, and window row are visible.

Use the StreamName route of Definition 5916.5. It forces every StreamName-facing consumer to display the same Q, L, W route before any regular-rational readback, dyadic tolerance row, or Real seal can be consumed.

Finally compare with the pointwise stream carrier of Definition 6084.1. A $\mathsf{PointwiseStreamName}^{\uparrow}$ read starts from a certified StreamName source and requested finite point windows; when that source is supplied by $\mathsf{FiniteStream}^{\uparrow}$, those windows are exactly the displayed finite request/list/window

route. Thus the pointwise consumer receives finite BEDC window data, not an ambient infinite stream, quotient stream equality, selected representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5916.9 (FiniteStream window concatenation)**.** *Let*

$$F_i = (Q_i, L_i, W_i, R_i, D_i, H_i, C_i, P_i, N_i)$$

for $i = 0, 1$ be accepted $\mathsf{FiniteStream}^{\uparrow}$ *packets whose endpoint window rows match, so the final displayed row of W_0 is the initial displayed row of W_1 under the componentwise* hsame *transport. Then the concatenated finite observation schedule is accepted as a* $\mathsf{FiniteStream}^{\uparrow}$ *packet whose* $\mathsf{StreamName}^{\uparrow}$ *consumer reads the first finite window, then the second finite window, before any* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *seal is exposed. The concatenated route remains finite:*

$$Q_0, Q_1, L_0, L_1, W_0, W_1 \longmapsto R, D, H, C, P, N.$$

Proof. Project the two carrier rows. The request rows Q_i, list spines L_i, and StreamName windows W_i are the only schedule data available before downstream readback.

Use the matching endpoint window rows to compose the finite lists. Concatenate the displayed request rows and their $\mathsf{List}^{\uparrow}$ spines, identifying the shared endpoint only through the given componentwise hsame transport.

The downstream rows are replayed componentwise. The transport row preserves the two windows, the displayed Cont replay reads W_0 and then W_1, and the provenance and local naming rows record the same finite route.

Thus the consumer reaches any $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ seal only after both finite windows have been displayed. No infinite stream, selected limit, quotient stream equality, or ambient completeness row is exposed by the concatenation. □

Theorem 5916.10 (FiniteStream product window)**.** *Given two accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packets over displayed rows*

$$F_i = (Q_i, L_i, W_i, R_i, D_i, H_i, C_i, P_i, N_i)$$

for $i = 0, 1$, a product finite-window read is admissible only by exposing both request rows Q_0, Q_1, both $\mathsf{List}^{\uparrow}$ *spines L_0, L_1, and the componentwise* $\mathsf{StreamName}^{\uparrow}$ *product window $W_0 \times W_1$ before any downstream row is consumed. The product route then reads the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, componentwise* hsame *transports, displayed* Cont *replays,* Pkg *provenance rows, and local* NameCert *rows for the same two finite packets. It exports no host product stream, no quotient stream equality, no selected limit, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the two carriers from Definition 5916.1. Each carrier exposes its own finite request row, list spine, StreamName window, regular-rational readback, dyadic tolerance row, transport, replay, provenance, and local naming.

Apply Definition 5916.3 separately to the two packets. The lock prevents either downstream read from replacing Q_i or L_i while keeping the displayed StreamName, RegSeqRat, DyadicRatCore, or Real-facing route.

Next use Definition 5916.4 and Definition 5916.5. They require each component window to pass through Q_i, L_i, W_i before its readback and tolerance rows are consumed.

The product window is therefore formed only after both component routes are visible. Pair the two displayed StreamName windows componentwise, then read R_0, R_1, D_0, D_1, H_0, H_1, C_0, C_1, P_0, P_1, and N_0, N_1 along the same finite request/list routes. Since neither carrier contains a host product stream, quotient stream equality, selected limit, or ambient completeness coordinate, the paired read remains a finite product-window route. □

Theorem 5916.11 (FiniteStream finite-window factorization)**.** *Every accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet consumed by a* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *finite-window route factors through the list-indexed finite request surface*

$$Q, L, W, R, D, H, C, P, N.$$

The factorization first exposes the finite request row Q *and* $\mathsf{List}^{\uparrow}$ *prefix spine* L*, then the displayed* $\mathsf{StreamName}^{\uparrow}$ *window* W*, then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary* R *and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and finally the componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. It exports no infinite stream, quotient stream equality, countable choice, selected limit, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows from Definition 5916.1. The packet inventory gives $Q, L, W, R, D, H, C, P, N$ and no coordinate outside that finite list.

Use the StreamName route of Definition 5916.5. It forces every StreamName-facing consumer to display Q, L, W before the downstream rows are available.

Next use the product-window theorem, Definition 5916.10, in the singleton component case. The same argument that blocks a product read from bypassing its two request/list surfaces blocks a one-window read from bypassing its single request/list surface.

It remains to identify the unavailable coordinates. The carrier contains no completed stream, quotient stream equality, countable-choice schedule, selected limit, or ambient Real completeness row, so the finite-window route factors exactly through the displayed request, window, readback, transport, replay, provenance, and naming rows. □

Falsifiable prediction: Every accepted $\mathsf{FiniteStream}^{\uparrow}$ read is refuted if it reaches a $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer without displaying the finite request row, $\mathsf{List}^{\uparrow}$ prefix spine, finite-window handoff, regular-rational readback boundary, dyadic tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: list, streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite request row, a List prefix spine, a StreamName window handoff, RegSeqRat readback boundary, DyadicRatCore tolerance row, transport, replay, provenance, and local naming as one finite schedule packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteStream}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteStream}^{\uparrow}$ packet is generated from finite BHist request rows, a ListUp prefix spine, a StreamName finite-window handoff, a RegSeqRat readback boundary, a DyadicRatCore tolerance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteStream}^{\uparrow}}$, seed)

Scope closed: Definition 5916.1, Definition 5916.2, Definition 5916.3, and Definition 5916.4 seed the finite observation-schedule route between List spines and StreamName finite windows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteStream}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close infinite stream equality, selected limits, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite request admission, List prefix exactness, StreamName handoff exactness, RegSeqRat and DyadicRatCore boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteStreamUp.

Theorem 5916.12 (FiniteStream pointwise metric forward route)**.** *Every* $\mathsf{PointwiseStreamMetric}^{\uparrow}$ *consumer that reads a* $\mathsf{FiniteStream}^{\uparrow}$ *packet must first consume the finite request row, list prefix spine, and finite-window handoff of Definition 5916.1. The route then reaches chapter 4139 only through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows. Thus the pointwise metric read receives finite stream windows as finite BEDC data and does not import an*

infinite stream, quotient stream equality, selected representative, countable choice, or ambient Real completeness.

Proof. Project the finite stream carrier of Definition 5916.1. The rows $Q, L, W, R, D, H, C, P, N$ display the finite request, list prefix, StreamName window, regular-rational readback, dyadic tolerance row, and structural evidence before any downstream consumer is visible.

Use the section sibling lock of Definition 5916.3. It keeps every Real-facing read attached to the same finite request row and list spine.

The pointwise metric chapter consumes exactly that finite-window route. Since the accepted packet has no coordinate for an infinite stream, quotient stream equality, selected representative, choice schedule, or ambient completeness, the forward route exports only the finite window surface. □

Theorem 5916.13 (FiniteStream prefix-cover determinacy)**.** *Every* $\mathsf{FiniteStream}^{\uparrow}$ *prefix-cover read offered to a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer first exposes the finite request row* Q*, the* $\mathsf{List}^{\uparrow}$ *prefix spine* L*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the local structural rows* H, C, P, N *of Definition 5916.1. The dependency route is read through chapter 126, chapter 301, chapter 291, and chapter 38; no downstream consumer receives an ambient stream, selected limit, quotient stream equality, countable-choice schedule, or detached real-completeness witness.*

Proof. Project the accepted $\mathsf{FiniteStream}^{\uparrow}$ carrier of Definition 5916.1. Its finite request row Q and $\mathsf{List}^{\uparrow}$ prefix spine L name the prefix-cover domain before any window is exported.

Read the StreamName, RegSeqRat, and DyadicRatCore rows next. The window row W is the only StreamName surface available to the consumer, the readback row R supplies the regular-rational interpretation, and the tolerance row D fixes the dyadic comparison budget.

Finally apply the existing NameCert obligations and sibling routes in the chapter. The structural rows H, C, P, N preserve the same finite route through the cited StreamName, RegSeqRat, DyadicRatCore, and Real chapters, so a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer cannot bypass the prefix cover or replace it with an ambient completed stream. □

Theorem 5916.14 (FiniteStream prefix-refinement window exhaustion)**.** *Every accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet consumed by a* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *reader factors through the same finite request row* Q *and list prefix row* L*. Any refinement of the requested window must display the prefix inclusion before downstream rows* W, R, D, H, C, P, N *are consumed. The route is read through chapter 126, chapter 301, chapter 291, and chapter 38; it is not an ambient infinite stream, quotient equality, selected limit, or Real completeness fact.*

Proof. Project the finite stream carrier of Definition 5916.1. The accepted packet exposes the finite request row and list spine before any StreamName or Real-facing rows are available.

Use the section sibling lock, Definition 5916.3. It prevents a downstream consumer from replacing Q or L while preserving the same StreamName, RegSeqRat, DyadicRatCore, or Real-facing route.

Now apply the StreamName dependency route, Definition 5916.4. A reader that reaches W, R, D, H, C, P, N must first pass through the displayed finite schedule. A refined window therefore has to show its prefix inclusion against the original L before those downstream rows can be consumed.

The sibling chapters supply only the named downstream surfaces: StreamName gives the finite window, RegSeqRat gives the regular readback, DyadicRatCore gives tolerance data, and Real

gives the terminal seal. None supplies a second request row or an unlisted list prefix, so the accepted read is exhausted by the displayed prefix-refinement route. □

Theorem 5916.15 (FiniteStream countable-name prefix route)**.** *Every countable-name consumer of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *finite prefix observation must pass through the displayed finite request row Q, the list prefix row L, the prefix-refinement window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, the* $\mathsf{StreamName}^{\uparrow}$ *row W, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{FiniteStream}^{\uparrow}}$ *row before exporting a countable-name observation. The route is local to the finite prefix and does not claim an ambient infinite stream, quotient stream equality, selected real limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. First project the existing prefix-refinement window of Definition 5916.14. It forces the consumer to read the same finite request row Q and list prefix row L before any downstream countable-name surface is available.

Next apply prefix-cover determinacy, Definition 5916.13, together with the section sibling lock of Definition 5916.3. These keep the read attached to the displayed prefix and prevent replacement by a second request row, a second list prefix, or an ambient infinite stream.

The remaining rows are then routed through the named dependency children. The $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{StreamName}^{\uparrow}$ window are consumed through Definition 5916.4; the DyadicRatCore tolerance and Real seal are the downstream sibling surfaces named in chapter 291 and chapter 38. Componentwise transport, replay, provenance, and the local NameCert row carry the same finite-prefix route to the exported countable-name observation. □

Finite $\mathsf{FiniteStream}^{\uparrow}$ windows are consumed by the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301 before the downstream $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is available. The dependency route is the displayed finite observation row W together with the carrier rows Q, L, R, D, H, C, P, N of Definition 5916.1; it records no completed stream, quotient stream equality, selected real limit, or ambient real-completeness datum.

Theorem 5916.16 (FiniteStream regular-real tail forward route)**.** *Any* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet must expose the finite request row Q, the* $\mathsf{List}^{\uparrow}$ *spine L, the* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D before reading the regular-real tail-equivalence boundary of chapter 5940. The forward route is therefore*

$$Q, L, W, R, D \longmapsto H, C, P, N,$$

with componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row supplied only after the finite request window has been displayed.*

Proof. Project the $\mathsf{FiniteStream}^{\uparrow}$ carrier of Definition 5916.1. The accepted packet stores exactly the finite request row Q, list spine L, StreamName window W, RegSeqRat readback boundary R, DyadicRatCore tolerance row D, and structural rows H, C, P, N.

Apply the StreamName route of Definition 5916.5. It forces any downstream StreamName, RegSeqRat, DyadicRatCore, or Real-facing read to expose Q, L, W before the readback, tolerance, transport, replay, provenance, or naming rows are consumed.

Now read chapter 5940 as the downstream consumer. Its same-real tail boundary consumes regular rational sources through a finite StreamName window and dyadic tolerance ledger, so a $\mathsf{FiniteStream}^{\uparrow}$ source must provide Q, L, W, R, D before that boundary is available. The route supplies finite request-window data only; it does not export quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5916.17 (FiniteStream Real seal dependency route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packet must expose the carrier rows*

$$Q, L, W, R, D, H, C, P, N$$

from Definition 5916.1 before any terminal $\mathsf{Real}^{\uparrow}$ *seal is consumed. The row* W *is read through the* $\mathsf{StreamName}^{\uparrow}$ *window,* R *through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary, and* D *through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance route. The downstream seal is the* $\mathsf{Real}^{\uparrow}$ *consumer of chapter 38; the route asserts neither ambient* $\mathsf{Real}^{\uparrow}$ *completeness nor quotient stream equality.*

Proof. Project the accepted carrier of Definition 5916.1. Its displayed inventory is exactly $Q, L, W, R, D, H, C, P, N$, with no hidden completed stream or selected limit coordinate.

Apply the NameCert obligations of Definition 5916.2. They force the finite request row Q and $\mathsf{List}^{\uparrow}$ spine L to appear before the $\mathsf{StreamName}^{\uparrow}$ window W, and they place the $\mathsf{RegSeqRat}^{\uparrow}$ boundary R and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D downstream of that window.

Now name $\mathsf{Real}^{\uparrow}$ as the downstream seal consumer. Since the carrier exposes only the finite request, window, readback, tolerance, transport, replay, provenance, and naming rows, the Real seal can be consumed only after those rows are visible, and the argument supplies no Real completeness theorem or quotient stream equality. □

Theorem 5916.18 (FiniteStream uniform Real seal route)**.** *Every family of accepted* $\mathsf{FiniteStream}^{\uparrow}$ *packets consumed by a common* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *finite-window reader uses the same displayed request, list, and window route* Q, L, W *before reaching* R, D, H, C, P, N*. Componentwise* hsame *transport and displayed* Cont *replay preserve that route uniformly across the family. The route exports no quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier inventory of Definition 5916.1. Each accepted packet exposes the same row pattern $Q, L, W, R, D, H, C, P, N$, so a common finite-window reader can compare packets only through those displayed rows.

Apply Definition 5916.2. The request row Q and list row L must be visible before the $\mathsf{StreamName}^{\uparrow}$ window W, and the readback and tolerance rows R, D are downstream from that window.

Use Definition 5916.3 to keep sibling section reads on the same finite route. Then apply Definition 5916.17; any Real-facing consumer reaches the terminal seal only after $Q, L, W, R, D, H, C, P, N$ are displayed.

The componentwise transport row H and displayed replay row C preserve the route uniformly for the whole family, while P, N source and name it. Since none of the packets carries quotient stream equality, selected limits, countable choice, or ambient Real completeness, the common reader receives only the uniform finite-window route. □

Every $\mathsf{FiniteStream}^{\uparrow}$ read consumed by chapter 301 or chapter 38 first passes through the finite-observation surface of chapter 126. The packet therefore exposes its request row, list spine, and StreamName window before the RegSeqRat readback boundary or the Real seal can be inspected; no consumer receives an ambient infinite stream or a selected real limit outside the displayed finite route.

The finite-window source is the StreamName naming certificate itself, where each exported observation is a bounded request against a named stream window, while the regular-Cauchy readback source is chapter 301, where the window is consumed as rational sequence data rather than as an ambient sequence object. Thus the sibling lattice for $\mathsf{FiniteStream}^{\uparrow}$ has two visible dependency

faces: chapter 126 supplies the window discipline, and chapter 301 supplies the regular-rational readback discipline used by later completion consumers.

Theorem 5916.19 (FiniteStream sibling dependency route)**.** *For every accepted* $\mathsf{FiniteStream}^{\uparrow}$ *carrier, any downstream finite-stream consumer that reads a prefix window toward* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *first consumes the same finite observation lattice*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the StreamName finite-window discipline of chapter 126, the regular-rational readback route of chapter 301, and the Real seal of chapter 38. The dependency is sibling-visible because each stage is named by the carrier's BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}_{\mathsf{FiniteStream}^{\uparrow}}$ *rows before the prefix window is exported to any later finite-stream consumer.*

The downstream packets $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ and $\mathsf{RegularSequenceClusterWitness}^{\uparrow}$ consume the same finite request/list/window route before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ seal read; for instance, chapter 5940 reads equality of regular real names through that displayed finite tail data. The $\mathsf{FiniteStream}^{\uparrow}$ dependency route reads finite-window names through chapter 126, the regular-sequence rational route of chapter 301, and the real-name seal of chapter 38 before any Real-facing consumer inspects a displayed finite window. Thus a finite stream is visible to downstream L10 consumers only after its stream-name row, rational regular-sequence row, and $\mathsf{Real}^{\uparrow}$ readback route have all been exposed inside the same dependency lattice.

Chapter 5917

A Concrete Naming Certificate for $\mathsf{PoincareDiskGeodesicProjection}^{\uparrow}$

Following classical Poincare disk geometry, the $\mathsf{PoincareDiskGeodesicProjection}^{\uparrow}$ packet records the finite route by which displayed disk points are projected onto a boundary-determined geodesic before metric and phase-transport consumers read the result. It sits after the disk carrier of chapter 377, the hyperbolic metric rows of chapter 3474, the geodesic-boundary packet of chapter 3491, the boundary-fixed transport packet of chapter 3500, and the disk phase-transport route of chapter 3455. The packet names a finite geodesic projection surface; it is not a completed hyperbolic plane, quotient ideal boundary, selected nearest point, host minimizer, physical horizon, or global compactification.

Definition 5917.1 (Poincare disk geodesic projection carrier)**.** *A* $\mathsf{PoincareDiskGeodesicProjection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, B, M, T, J, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-point row, recording the finite disk coordinates and source observations to be projected.*
2. *B is the boundary-geodesic row, recording the two displayed boundary endpoints or boundary-incidence data that determine the geodesic arc.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparison row, exposing the metric reads used to compare the disk point with the geodesic boundary data.*
4. *T is the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *row, carrying the boundary-fixed transport that preserves the displayed geodesic endpoints during the projection read.*
5. *J is the projection-join row, tying D, B, M, T to the same finite source surface before any projected point or phase-transport consumer can read the packet.*
6. *H records componentwise* hsame *transport for disk points, boundary-geodesic data, hyperbolic-metric comparisons, boundary-fixed transport, projection join, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a projection request through D, B, M, T, J, H, P, N.*

8. *P is the* Pkg *provenance over* PoincareDiskGeodesicProjection$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicGeodesicBoundary$^{\uparrow}$, PoincareBoundaryFixedTransport$^{\uparrow}$, PoincareDiskPhaseTransport$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{PoincareDiskGeodesicProjection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, B, M, T, J, H, C, P, N$. It has no coordinate for a completed disk, quotient boundary equality, selected nearest point, host minimizer, physical horizon, or global hyperbolic compactification.

Lean4 ✓ `BEDC.Derived.PoincareDiskGeodesicProjectionUp.PoincareDiskGeodesicProjectionTasteGate`
`single carrier alignment`

Theorem 5917.2 (Poincare disk geodesic projection NameCert obligations)**.** *For every accepted* PoincareDiskGeodesicProjection$^{\uparrow}$ *carrier*

$$G = (D, B, M, T, J, H, C, P, N),$$

the local NameCert$_{\text{PoincareDiskGeodesicProjection}^{\uparrow}}$ *surface consists exactly of disk-point admission, boundary-geodesic admission, hyperbolic-metric comparison, boundary-fixed transport, projection-join exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every projection consumer must read*

$$D, B, M, T \longmapsto J \longmapsto H, C, P, N$$

before it can pass projected geodesic data to a phase-transport or metric consumer.

Proof. Project the carrier of Definition 5917.1. The only visible source rows are the disk point row D, boundary-geodesic row B, hyperbolic-metric comparison row M, boundary-fixed transport row T, and the structural rows J, H, C, P, N.

The geodesic projection read is forced through D, B, M, T. Disk coordinates provide the source point, boundary-geodesic data provide the target arc, metric comparisons provide the displayed hyperbolic reading, and boundary-fixed transport preserves the endpoints while the request is replayed.

The projection-join row J binds those four rows to one finite source surface. Componentwise transport and displayed replay then preserve that same packet through H and C.

Finally P and N exhaust provenance and local naming. Since no completed disk, quotient boundary equality, selected nearest point, host minimizer, physical horizon, or global compactification row appears in the carrier, the projection remains a finite NameCert route. □

Falsifiable prediction: An accepted PoincareDiskGeodesicProjection$^{\uparrow}$ read is refuted if a consumer obtains projected geodesic data without displaying the disk-point row, boundary-geodesic row, HyperbolicMetric comparison row, boundary-fixed transport row, projection join, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: unitdisk, hyperbolic-metric, hyperbolic-geodesic-boundary, poincare-boundary-fixed-transport, poincare-disk-phase-transport: no listed sibling is bijective to PoincareDiskGeodesicProjec because this packet jointly carries disk points, boundary-geodesic data, metric comparisons, boundary-fixed transport, projection joining, transport, replay, provenance, and local naming as one finite route.

Closure status for NameCert$_{\text{PoincareDiskGeodesicProjection}^{\uparrow}}$.

Constructive story: A PoincareDiskGeodesicProjection$^{\uparrow}$ packet is generated from UnitDisk point rows, boundary-geodesic rows, HyperbolicMetric comparison rows, PoincareBoundaryFixedTransport rows, projection joining, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PoincareDiskGeodesicProjection}^{\uparrow}}$, seed)

Scope closed: Definition 5917.1 and Definition 5917.2 seed the finite disk-point, boundary-geodesic, hyperbolic-metric, and phase-transport projection route.

Formal verification: FormalStatus(NameCert$_{\text{PoincareDiskGeodesicProjection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic-plane geometry, quotient ideal-boundary equality, selected nearest points, host minimizers, physical horizons, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-point admission, boundary-geodesic admission, hyperbolic-metric comparison, boundary-fixed transport, projection-join exactness, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDiskGeodesicProjectionUp.

Theorem 5917.3 (Poincare disk geodesic projection hyperbolic route)**.** *Every public* PoincareDiskGeodesicProjectio *projection read factors through the hyperbolic geodesic and phase-transport siblings before export. For an accepted carrier*

$$G = (D, B, M, T, J, H, C, P, N),$$

the consumer first reads the boundary-geodesic data through chapter 3491, then the metric comparison row through chapter 3474, and then the phase-transport route through chapter 3455. Only after those sibling rows have been exposed may the local projection join J be exported as projected disk-geodesic data.

Proof. Project the accepted carrier through Definition 5917.1. The rows B, M, and T are visible coordinates of the same finite packet, and the projection join J is downstream of those rows.

Apply Definition 5917.2. Its NameCert surface requires every projection consumer to read

$$D, B, M, T \longmapsto J \longmapsto H, C, P, N$$

before phase-transport or metric export.

The boundary-geodesic row B is therefore read through chapter 3491; the metric row M is read through chapter 3474; and the outgoing transport consumer is read through chapter 3455. Since J is not available before those sibling rows are displayed, the projection read cannot float independently of the hyperbolic and phase-transport route. □

Theorem 5917.4 (Poincare disk geodesic projection sibling lattice)**.** *Every accepted* PoincareDiskGeodesicProjectior *route reads the disk-carrier, boundary row, metric/geodesic row, transport row, replay row, provenance row, and local* NameCert *row before any projection consumer inspects the projected route. The sibling lattice runs through the disk carrier of chapter 377, the hyperbolic metric surface of chapter 3474, the geodesic-boundary packet of chapter 3491, the boundary-fixed transport route of chapter 3500, and the outgoing phase-transport route of chapter 3455.*

Proof. Project an accepted carrier

$$G = (D, B, M, T, J, H, C, P, N)$$

through Definition 5917.1. The rows D, B, M, T, H, C, P, N are visible before the projection join J can be exported.

Apply Definition 5917.2. It forces every projection consumer to read the disk row, boundary-geodesic row, hyperbolic metric row, boundary-fixed transport row, projection join, componentwise transport, replay, provenance, and local naming in that order.

Thus the route is horizontal rather than kernel-only: D is anchored in chapter 377, M in chapter 3474, B in chapter 3491, T in chapter 3500, and the downstream projection consumer in chapter 3455. Since J is unavailable until those rows have been displayed, no projection read floats apart from its disk and hyperbolic siblings. $\square$

Chapter 5918

A Concrete Naming Certificate for $\mathsf{CalkinWilfTree}^{\uparrow}$

Following Calkin–Wilf and Hardy–Wright, the $\mathsf{CalkinWilfTree}^{\uparrow}$ packet records the finite binary tree route by which positive rational presentations are generated from 1/1 through the child rules $a/b \mapsto a/(a+b)$ and $a/b \mapsto (a+b)/b$. It sits between the rational carrier of chapter 37, the determinant-one tree boundary of chapter 1704, the adjacent rational ledger of chapter 1759, and the breadth-first stream surface of chapter 1474. The packet names a finite address-to-rational tree carrier; it is not a host enumeration of rational numbers, a quotient-rational equality theorem, a selected representative principle, or an ambient completion claim.

Definition 5918.1 (Calkin–Wilf tree carrier)**.** *A* $\mathsf{CalkinWilfTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, Q, L, R, S, F, H, C, P, N)$$

with the following displayed rows:

1. *A is the binary address row, recording the finite left/right word from the root address to the inspected node.*
2. *Q is the positive* $\mathsf{Rat}^{\uparrow}$ *readback row at the inspected address, stored as displayed numerator and denominator histories.*
3. *L is the left-child row, recording the displayed transition from a/b to $a/(a+b)$.*
4. *R is the right-child row, recording the displayed transition from a/b to $(a+b)/b$.*
5. *S is the* $\mathsf{SternBrocot}^{\uparrow}$ *comparison row, sending the inspected address to the sibling finite tree boundary and mediant route.*
6. *F is the* $\mathsf{FareySequence}^{\uparrow}$ *comparison row, recording the adjacent determinant-one rational window through which the inspected presentation is compared with Farey neighbors.*
7. *H stores componentwise* hsame *transport for the address, rational readback, child-step, Stern–Brocot comparison, Farey comparison, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an address request through child-step inspection, rational readback, Stern–Brocot comparison, Farey-neighbor comparison, and downstream handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CalkinWilfTree}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{SternBrocot}^{\uparrow}$, $\mathsf{FareySequence}^{\uparrow}$, $\mathsf{CalkinWilfEnumeration}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CalkinWilfTree}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, Q, L, R, S, F, H, C, P, N$. The carrier has no coordinate for host enumeration, quotient rational equality, selected representatives, signed-rational coverage, real order, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CalkinWilfTreeUp.CalkinWilfTreeUp`

Theorem 5918.2 (Calkin–Wilf tree NameCert obligations)**.** *For every accepted* $\mathsf{CalkinWilfTree}^{\uparrow}$ *carrier*

$$T = (A, Q, L, R, S, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CalkinWilfTree}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed binary address, positive rational readback, child-step, Stern–Brocot comparison, Farey comparison, transport, replay, provenance, and local naming rows;*
2. *root admission, requiring the empty address in A to expose the displayed positive rational row $1/1$ in Q;*
3. *child-step exactness, requiring every left edge to be read through L as $a/b \mapsto a/(a+b)$ and every right edge to be read through R as $a/b \mapsto (a+b)/b$;*
4. *Stern–Brocot comparison, requiring S to compare the addressed node through the finite rational-tree route of chapter 1704;*
5. *Farey comparison, requiring F to expose the adjacent determinant-one rational window before any consumer treats Q as rational comparison data;*
6. *enumeration handoff, requiring the breadth-first stream packet of chapter 1474 to read only the displayed A, Q, L, R route when it consumes tree nodes;*
7. *non-escape, stating that host enumeration, quotient rational equality, selected representatives, signed-rational coverage, real order, and ambient completion are absent from the packet.*

Proof. Carrier admission is projection from Definition 5918.1. The accepted tuple exposes exactly the address row, rational readback row, two child-step rows, sibling comparison rows, and structural rows listed in the statement.

Root admission is the base case of the displayed address row. At the empty address, the packet reads the positive $\mathsf{Rat}^{\uparrow}$ presentation $1/1$ from Q, with no external enumeration or quotient comparison.

For the successor step, inspect the next bit of A. A left bit uses L and a right bit uses R; in both cases the numerator-denominator presentation remains positive and is written back to Q as finite BHist data.

The sibling comparison obligations are then read from S and F. The Stern–Brocot row compares the addressed node with the finite tree boundary route, while the Farey row records the adjacent determinant-one window before rational comparison data can be consumed.

Finally H, C, P, N transport, replay, package, and name the same finite inventory. Because the inventory has no coordinate for host enumeration, quotient rational equality, selected representatives, signed coverage, real order, or ambient completion, the listed rows exhaust the local certificate. □

Theorem 5918.3 (Calkin–Wilf tree rational-enumeration handoff)**.** *Every* $\mathsf{CalkinWilfEnumeration}^{\uparrow}$ *read of an accepted* $\mathsf{CalkinWilfTree}^{\uparrow}$ *carrier factors through the finite route*

$$A \longmapsto L, R \longmapsto Q \longmapsto S, F \longmapsto H, C, P, N.$$

Consequently the enumeration sibling receives positive $\mathsf{Rat}^{\uparrow}$ *presentations generated by the displayed address row and child-step rows, while Stern–Brocot and Farey comparison data remain available as finite neighbor checks. The handoff does not export host rational enumeration, quotient equality, selected representatives, signed-rational coverage, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed address row A. The empty address exposes $1/1$ through Q, and the enumeration sibling may read that root as the first positive rational presentation.

For a successor address, the induction hypothesis supplies the parent presentation through the earlier prefix of A. The next bit selects exactly one of L or R, so the child presentation is again computed inside the finite carrier before it is written to Q.

The comparison rows are read after the child presentation is present. Row S routes the same address to the Stern–Brocot boundary surface, and row F records the Farey-neighbor comparison window for the displayed rational presentation.

The enumeration handoff is therefore a projection of finite rows, not a separate enumeration principle. Transport, replay, provenance, and local naming preserve this route through H, C, P, N, so downstream rational-enumeration consumers receive no quotient equality, host selector, signed coverage, real order, or completion coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CalkinWilfTree}^{\uparrow}}$.

Constructive story: A $\mathsf{CalkinWilfTree}^{\uparrow}$ packet is generated from a finite binary address row, a positive RatUp readback row, displayed left and right child-step rows, SternBrocot and FareySequence comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CalkinWilfTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5918.1, Definition 5918.2, and Definition 5918.3 seed the finite Calkin–Wilf binary-tree packet between Rat, SternBrocot, FareySequence, and CalkinWilfEnumeration rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CalkinWilfTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not claim host rational enumeration, quotient rational equality, selected representatives, signed-rational coverage, real order, ambient completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, root admission, child-step exactness, Stern–Brocot comparison, Farey comparison, enumeration handoff, transport, replay, provenance, and local naming for BEDC.Derived.CalkinWilfTree

Chapter 5919

A Concrete Naming Certificate for $\mathsf{TarskiFixedPoint}^{\uparrow}$

Following Tarski's fixed point theorem for monotone maps on complete lattices, the $\mathsf{TarskiFixedPoint}^{\uparrow}$ packet records the finite BEDC route by which a lattice self-map certificate exposes prefixpoint, postfixpoint, iteration, and fixed-row data to downstream order-theoretic consumers. It is grounded in the finite order surfaces of chapter 55, chapter 53, and chapter 1378, and it hands its iteration ledger to chapter 2383. The packet is a naming certificate for a displayed monotone fixed-row route, not a host theorem asserting ambient complete-lattice semantics or unrestricted fixed-point existence.

Definition 5919.1 (Tarski fixed-point carrier)**.** *A* $\mathsf{TarskiFixedPoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (L, P, O, F, A, B, I, X, H, C, G, N)$$

with the following displayed rows:

1. *L is the finite* $\mathsf{Lattice}^{\uparrow}$ *source row, exposing meet, join, bound, and finite comparison data only through the carried lattice certificate.*
2. *P is the* $\mathsf{Poset}^{\uparrow}$ *row underneath L, recording reflexive, transitive, and antisymmetric comparison reads for the same displayed endpoints.*
3. *O is the* $\mathsf{Order}^{\uparrow}$ *row used by arithmetic and ledger consumers to inspect the finite comparison branches without importing a host order.*
4. *F is the monotone self-map row. It stores source membership, target membership, and the displayed implication from $x \leq y$ to $F(x) \leq F(y)$ for the carried rows.*
5. *A is the prefixpoint ledger, listing finite rows whose displayed comparison reads certify $F(x) \leq x$.*
6. *B is the postfixpoint ledger, listing finite rows whose displayed comparison reads certify $x \leq F(x)$.*
7. *I is the* $\mathsf{FixedPointIteration}^{\uparrow}$ *chain row, recording the finite orbit, comparison phase, and stabilization checks read by iteration consumers.*
8. *X is the fixed-row witness. It is available only after A, B, I are displayed and records the paired comparisons $F(x) \leq x$ and $x \leq F(x)$ as local ledger rows.*
9. *H records componentwise* hsame *transport for the lattice, poset, order, self-map, prefixpoint, postfixpoint, iteration, fixed-row, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from order reads through the monotone self-map, prefixpoint and postfixpoint ledgers, iteration chain, fixed-row witness, transport, provenance, and naming rows.*
11. *G is the* Pkg *provenance over* $\mathsf{TarskiFixedPoint}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{FixedPointIteration}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{TarskiFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, P, O, F, A, B, I, X, H, C, G, N$. It has no coordinate for host complete-lattice semantics, quotient function equality, unrestricted choice, or ambient standard-model fixed points.

Lean4 (definition) `BEDC.Derived.TarskiFixedPointUp.TarskiFixedPointUp`

Theorem 5919.2 (Tarski fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{TarskiFixedPoint}^{\uparrow}$ *carrier*

$$T = (L, P, O, F, A, B, I, X, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{TarskiFixedPoint}^{\uparrow}}$ *surface consists exactly of the finite lattice source, poset comparison, order readback, monotone self-map, prefixpoint ledger, postfixpoint ledger, iteration chain, fixed-row witness, componentwise transport, continuation replay, package provenance, and local naming rows of Definition 5919.1. A valid obligation read must display the sibling route*

$$\mathsf{Order}^{\uparrow} \longmapsto \mathsf{Poset}^{\uparrow} \longmapsto \mathsf{Lattice}^{\uparrow} \longmapsto \mathsf{FixedPointIteration}^{\uparrow}$$

before the fixed-row witness is exported. No obligation may call on host complete-lattice semantics, quotient equality of self-maps, an unrestricted-choice selector, or a standard-model fixed point outside the packet.

Proof. Project the carrier of Definition 5919.1. The accepted packet exposes only the rows $L, P, O, F, A, B, I, X, H, C$

The order-theoretic source is read first. The $\mathsf{Order}^{\uparrow}$ row supplies finite comparison branches, the $\mathsf{Poset}^{\uparrow}$ row supplies comparison stability, and the $\mathsf{Lattice}^{\uparrow}$ row supplies meet, join, and bound data for the same displayed endpoints.

The monotone self-map row F is then interpreted over that carried source. Prefixpoint rows in A and postfixpoint rows in B can be checked only by replaying the displayed comparison implications from F together with the source rows.

The iteration chain I and fixed-row witness X consume those ledgers. Since X stores only the paired local comparisons $F(x) \leq x$ and $x \leq F(x)$, it is a packet witness rather than a host fixed point.

The remaining rows H, C, G, N transport, replay, source, and name the same route. No excluded complete-lattice, quotient-function, unrestricted-choice, or ambient fixed-point principle is present as a carrier coordinate. □

Theorem 5919.3 (Tarski fixed-point lattice-iteration handoff)**.** *Every downstream consumer of an accepted* $\mathsf{TarskiFixedPoint}^{\uparrow}$ *packet reads the fixed-row witness only after the* $\mathsf{Lattice}^{\uparrow}$ *source,* $\mathsf{Poset}^{\uparrow}$ *comparison surface,* $\mathsf{Order}^{\uparrow}$ *branch ledger, monotone self-map row, prefixpoint and postfixpoint ledgers, and* $\mathsf{FixedPointIteration}^{\uparrow}$ *chain have been displayed. The handoff exports the finite rows*

$$L, P, O, F, A, B, I, X, H, C, G, N$$

and nothing that identifies quotient self-maps, selects arbitrary lattice elements, or asserts a host standard-model fixed point.

Proof. Start with a downstream consumer of the accepted packet. The only consumer-facing route is the replay row C, and its first dependencies are the finite order, poset, and lattice rows O, P, L.

The self-map row F is available only as a monotone row over those displayed comparisons. The prefixpoint ledger A and postfixpoint ledger B therefore record finite comparison checks rather than semantic least or greatest fixed-point arguments.

The iteration chain I consumes the same ledgers and exports only its displayed finite chain. The fixed-row witness X is downstream of I and stores paired comparisons for one named row.

Transport, provenance, and local naming preserve that ordering. Thus every consumer receives the lattice-iteration handoff and no larger complete-lattice theorem, quotient-function equality, unrestricted-choice selector, or ambient fixed-point object. □

Closure status for $\mathsf{NameCert}_{\mathsf{TarskiFixedPoint}\uparrow}$.

Constructive story: A $\mathsf{TarskiFixedPoint}^{\uparrow}$ packet is generated from a finite LatticeUp source, PosetUp comparison surface, OrderUp branch ledger, monotone self-map row, prefixpoint and postfixpoint ledgers, FixedPointIterationUp chain, fixed-row witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TarskiFixedPoint}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5919.1, Definition 5919.2, and Definition 5919.3 seed the finite Tarski fixed-point naming route over displayed order, lattice, monotone-map, ledger, iteration, witness, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TarskiFixedPoint}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host complete-lattice semantics, quotient function equality, unrestricted choice, ambient standard-model fixed points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lattice and poset source coverage, monotone self-map stability, prefixpoint and postfixpoint ledger coverage, iteration-chain replay, fixed-row witness coverage, transport, replay, provenance, and local naming for BEDC.Derived.TarskiFixedPointUp.

Chapter 5920

A Concrete Naming Certificate for $\mathsf{LocatedRealCompletionUniversal}^{\uparrow}$

Following Bishop and Bridges, the constructive universal property of the located real completion is read here as a finite extension surface rather than as a quotient completion theorem. The $\mathsf{LocatedRealCompletionUniversal}^{\uparrow}$ packet sits downstream of the rational core of chapter 291, the stream-name route of chapter 126, the regular rational sequence carrier of chapter 301, the real-name seal of chapter 38, the located completion functor of chapter 3531, and the Cauchy completion universal property of chapter 2319. It records only displayed rational probes, regular Cauchy windows, located modulus data, extension rows, uniqueness comparisons, transport, replay, provenance, and local naming; it refuses quotient classes, choice-selected limits, and ambient completeness.

Definition 5920.1 (LocatedRealCompletionUniversal carrier)**.** *A* $\mathsf{LocatedRealCompletionUniversal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, S, R, E, L, F, X, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row, naming the finite rational probes and dyadic tolerance budgets inspected by the completion route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, displaying the finite stream windows on which regular names are read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, packaging the stream observations as regular rational Cauchy data with the displayed modulus and tolerance schedule.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from D, S, R and from no other source.*
5. *L is the located comparison row, recording the finite locatedness tests and apartness witnesses used by the extension route.*
6. *F is the* $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ *handoff row, carrying the displayed source map and located completion action on the finite rows already exposed.*
7. *X is the extension row, naming the candidate map on completed real names as a route through D, S, R, E, L, F.*

8. *Q is the* CauchyCompletionUniversalProperty$^{\uparrow}$ *comparison row, recording uniqueness only as a displayed comparison between two extension routes that agree on the dense rational source rows.*
9. *H records componentwise* hsame *transport for dyadic probes, stream windows, regular rational readbacks, Real seals, located comparisons, functor handoffs, extension rows, universal-property comparisons, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from every universal-property request through $D, S, R, E, L, F, X, Q, H, P, N$.*
11. *P is the* Pkg *provenance over* LocatedRealCompletionUniversal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, LocatedCompletionFunctor$^{\uparrow}$, CauchyCompletionUniversalProperty$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{LocatedRealCompletionUniversal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient classes of Cauchy names, choice-selected limits, ambient completeness of a host real line, global equality of completed objects, or an unrestricted extension theorem detached from the displayed located route.

Lean4 ✓ `BEDC.Derived.LocatedRealCompletionUniversalUp.LocatedRealCompletionUniversalTasteGate`
`single carrier alignment`

Theorem 5920.2 (LocatedRealCompletionUniversal NameCert obligations)**.** *For every accepted* LocatedRealCompletionUniversal$^{\uparrow}$ *carrier*

$$U = (D, S, R, E, L, F, X, Q, H, C, P, N),$$

the local NameCert$_{\text{LocatedRealCompletionUniversal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for dyadic rational probes, stream windows, regular rational readbacks, Real seals, located comparisons, located completion functor handoffs, extension rows, universal-property comparison rows, transport, replay, provenance, and local naming;*
2. *rational-source density, requiring every extension request to expose D, S, R before the terminal* Real$^{\uparrow}$ *seal E can be consumed;*
3. *located functoriality, requiring F to read the displayed located comparison row L together with the same rational-source and Real-seal rows;*
4. *extension formation, requiring X to be built from D, S, R, E, L, F and not from a quotient class or selected completion point;*
5. *uniqueness comparison, requiring Q to compare extension routes only after agreement on the displayed dense rational source has been named;*
6. *structural stability under H, C, P, N, which transports, replays, sources, and names only the displayed packet.*

No obligation is stored outside these rows.

Proof. Project the carrier of Definition 5920.1. The source-facing rows are D, S, R, E: dyadic probes, stream windows, regular rational readback, and the Real seal.

The located route is then read through L and F. The located comparison row supplies the finite tests used by the functor handoff, and F cannot act before the source-facing rows have been displayed.

The extension row X is downstream of those data. Since the carrier has no quotient-class or selected-limit coordinate, an accepted extension route is a displayed route through D, S, R, E, L, F rather than a host completion map.

Finally read the comparison and structural rows. The row Q compares extension routes only by their displayed agreement on the dense rational source, while H, C, P, N transport, replay, source, and name the same finite packet. These projections exhaust the listed NameCert obligations. □

Theorem 5920.3 (LocatedRealCompletionUniversal extension route)**.** *Every extension read exported by an accepted* $\mathsf{LocatedRealCompletionUniversal}^{\uparrow}$ *carrier factors through the finite route*

$$D, S, R \longmapsto E \longmapsto L, F \longmapsto X, Q \longmapsto H, C, P, N.$$

The route exposes dyadic rational probes, finite stream windows, regular rational Cauchy readbacks, a terminal Real seal, located comparisons, the located completion functor handoff, extension data, uniqueness comparison, transport, replay, provenance, and local naming. It does not export quotient classes of Cauchy names, choice-selected limits, ambient completeness, global equality of completed objects, or an extension principle outside the displayed packet.

Proof. Begin with an extension request. By carrier admission the only rational source available to the request is the displayed triple D, S, R, and the only real-facing endpoint attached to that source is E.

The located part of the request is read next. The row L supplies the finite located comparisons, and the $\mathsf{LocatedCompletionFunctor}^{\uparrow}$ handoff F consumes that row together with the already displayed rational and Real data.

The extension and uniqueness rows are downstream. The row X records the extension route built from the finite source and located functor rows, while Q compares two such routes only through displayed agreement on the dense rational source, as in the $\mathsf{CauchyCompletionUniversalProperty}^{\uparrow}$ packet.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same route. Since none of the refused objects is a coordinate of the carrier, no extension read can bypass the displayed finite surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealCompletionUniversal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealCompletionUniversal}^{\uparrow}$ packet is generated from DyadicRatCore probes, StreamName windows, RegSeqRat readbacks, a RealUp seal, located comparison rows, a LocatedCompletionFunctor handoff, extension rows, universal-property comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealCompletionUniversal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5920.1, Definition 5920.2, and Definition 5920.3 seed the constructive located Real completion universal-property carrier over displayed rational and located rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealCompletionUniversal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient classes of Cauchy names, choice-selected limits, ambient completeness, global equality of completed objects, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-source density, located functoriality, extension formation, uniqueness comparison, transport, replay, provenance, and local naming for BEDC.Derived.LocatedRealCompletionUniversal

Chapter 5921

A Concrete Naming Certificate for $\mathsf{NiemytzkiPlane}^{\uparrow}$

Following Niemytzki's classical tangent-disc topology, the $\mathsf{NiemytzkiPlane}^{\uparrow}$ packet records the upper half-plane together with its boundary-line neighbourhood discipline as finite NameCert data. It reads the open-set carrier of chapter 88, the metric-neighbourhood interface of chapter 1655, the real-coordinate seal of chapter 38, the coordinate-pair route of chapter 2000, and the separated-point surface of chapter 1925. The packet names the tangent-disc boundary route only; it does not assert metrizability, separability, compactness, classical choice principles, or a completed continuum beyond displayed real-coordinate rows.

Definition 5921.1 (Niemytzki plane carrier)**.** *A* $\mathsf{NiemytzkiPlane}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (U, B, T, R, Q, H, C, P, L)$$

with the following displayed rows:

1. *U is the upper-half-plane point row, presenting points with positive second coordinate through displayed* $\mathsf{Real}^{\uparrow}$ *coordinate reads.*
2. *B is the boundary-line point row, presenting points with zero second coordinate and a displayed first-coordinate readback.*
3. *T is the tangent-disc neighbourhood row, recording for a boundary point the basic neighbourhood consisting of that boundary point together with an open Euclidean disc tangent to the boundary line at that point.*
4. *R is the product-real coordinate readback row, routing all point and disc-center coordinates through* $\mathsf{Real}^{\uparrow}$ *and the product-topology surface before any topology consumer reads the packet.*
5. *Q is the classifier row, comparing accepted neighbourhood packets by upper-half-plane membership, boundary-line membership, tangent-disc incidence, and coordinate readback only.*
6. *H records componentwise* hsame *transport for point, boundary, tangent-disc, coordinate-readback, classifier, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a neighbourhood request through U, B, T, R, Q, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{NiemytzkiPlane}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricTopology}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ProductTopology}^{\uparrow}$, $\mathsf{HausdorffSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{NiemytzkiPlane}^{\uparrow}}$ *row.*
The classifier has no coordinate for a global metric inducing the topology, a countable base, compactness, selected centres by choice, or host equality of real pairs.

Theorem 5921.2 (Niemytzki plane NameCert obligations)**.** *For every accepted* $\mathsf{NiemytzkiPlane}^{\uparrow}$ *carrier*

$$N = (U, B, T, R, Q, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{NiemytzkiPlane}^{\uparrow}}$ *surface consists exactly of upper-half-plane point admission through U, boundary-line point admission through B, tangent-disc neighbourhood admission through T, product-real coordinate readback through R, classifier transport through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through L.*

Lean4 ✓ `BEDC.Derived.NiemytzkiPlaneUp.NiemytzkiPlaneNameCertObligations`
↪ variant `BEDC.Derived.NiemytzkiPlaneUp.NiemytzkiPlaneTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5921.1. The accepted packet exposes upper-half-plane points, boundary-line points, tangent-disc neighbourhoods, coordinate readback, classifier transport, replay, provenance, and local naming as separate rows.

The topology-facing rows are U, B, T. Upper-half-plane points are ordinary displayed coordinate pairs above the boundary, while boundary points obtain their basic neighbourhoods only through the tangent-disc row.

The coordinate and classifier rows are downstream of those point rows. The row R forces every coordinate read through $\mathsf{Real}^{\uparrow}$ and $\mathsf{ProductTopology}^{\uparrow}$, and Q compares packets only through visible incidence and coordinate data.

Finally H, C, P, L transport, replay, source, and name the same finite route. Since no row supplies a global metric, countable base, compactness witness, selected centre, or host pair equality, the listed rows exhaust the local NameCert surface. □

Theorem 5921.3 (Niemytzki tangent-disc boundary route)**.** *Every boundary-neighbourhood read exported by an accepted* $\mathsf{NiemytzkiPlane}^{\uparrow}$ *packet must pass through the boundary-line row B, the tangent-disc neighbourhood row T, and the product-real coordinate readback row R before it can be consumed as a* $\mathsf{Topology}^{\uparrow}$ *or* $\mathsf{HausdorffSpace}^{\uparrow}$ *neighbourhood. Thus the boundary topology is read as a finite tangent-disc certificate, not as an ordinary metric-ball neighbourhood, a chosen countable basis, a compactness claim, or a classical-choice selection.*

Proof. Start with the boundary-line row. It identifies a displayed point on the real boundary and separates it from upper-half-plane reads.

Next read the tangent-disc row. The neighbourhood of a boundary point is supplied by the boundary point together with a displayed tangent Euclidean disc, so the certificate does not reduce the boundary read to an ordinary product-open rectangle.

The coordinate row R then routes the centre, radius, tangent point, and incident upper-half-plane points through $\mathsf{Real}^{\uparrow}$ and $\mathsf{ProductTopology}^{\uparrow}$. Transport and replay preserve this order before provenance or local naming is inspected.

Consequently a topology or Hausdorff consumer receives only the displayed tangent-disc route. It receives no metrizability theorem, countable-base witness, compactness witness, selected family of discs, or hidden real-pair equality. □

Falsifiable prediction: Every accepted $\mathsf{NiemytzkiPlane}^{\uparrow}$ boundary-neighbourhood read displays a boundary-line row, a tangent-disc neighbourhood row, product-real coordinate readback, classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, metric-topology, real, product-topology, hausdorffspace: $\mathsf{NiemytzkiPlane}^{\uparrow}$ is not bijective to any listed sibling because it binds ordinary upper-half-plane coordinate reads and boundary tangent-disc neighbourhood reads into one finite topology packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NiemytzkiPlane}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NiemytzkiPlane}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5921.1, Definition 5921.2, and Definition 5921.3 seed the finite upper-half-plane, boundary-line, tangent-disc, product-real, classifier, transport, replay, provenance, and local naming route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NiemytzkiPlane}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim metrizability, separability, compactness, classical choice principles, selected tangent-disc families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tangent-disc neighbourhood exactness, product-real coordinate readback, classifier transport, replay, provenance, and local naming for BEDC.Derived.NiemytzkiPlaneUp.

Constructive story: A $\mathsf{NiemytzkiPlane}^{\uparrow}$ packet is generated from upper-half-plane point rows, boundary-line point rows, tangent-disc neighbourhood rows, product-real coordinate readback, classifier transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

The $\mathsf{RationalCauchyTelescope}^{\uparrow}$ hub routes completion-facing reads through chapter 1585 before exposing the dyadic, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Chapter 5922

A Concrete Naming Certificate for $\mathsf{RationalCauchyTelescope}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real names, the $\mathsf{RationalCauchyTelescope}^{\uparrow}$ packet records the finite telescoping route by which rational step rows become a regular Cauchy readback before any completed-real assertion is available. It sits after rational rows from chapter 37, the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational handoff of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite L10 route for Real and Completion consumers; it does not assert quotient streams, selected limits, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5922.1 (RationalCauchyTelescope carrier)**.** *A* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, D, W, R, M, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite* $\mathsf{Rat}^{\uparrow}$ *rational-step ledger, indexed by the same observation stages that the downstream stream window reads.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds the visible tail of the rational step rows.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing exactly the observation stages used by Q and D.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained by summing the displayed rational step rows over the windows in W.*
5. *M is the telescoping modulus row saying that every accepted tail comparison is discharged by the dyadic budget in D.*
6. *H records componentwise* hsame *transport for Q, D, W, R, M, C, P, N.*
7. *C records displayed* Cont *replay of the finite telescoping computation and its tail comparisons.*
8. *P is the* Pkg *provenance over* $\mathsf{RationalCauchyTelescope}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalCauchyTelescope}^{\uparrow}}$ *row.*

The carrier contains no quotient stream equality, selected limit, trichotomy decision, choice principle, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Theorem 5922.2 (RationalCauchyTelescope NameCert obligations)**.** *For every accepted* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *carrier*

$$T = (Q, D, W, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RationalCauchyTelescope}^{\uparrow}}$ *surface consists exactly of rational-step admission, dyadic tolerance visibility, finite-window exposure, regular rational readback, telescoping-modulus discharge, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every obligation is read through the finite route*

$$\mathsf{Rat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Proof. Project the carrier of Definition 5922.1. The only mathematical rows available to a consumer are Q, D, W, R, M together with the structural rows H, C, P, N.

The rational-step ledger Q supplies the finite addends, while W supplies the observation stages on which those addends may be read. The dyadic row D fixes the tolerance budget for tail comparisons on the same windows.

The readback row R is obtained by displayed finite summation over Q and W. The modulus row M then records that every accepted telescoping tail comparison is bounded by the dyadic budget from D.

Transport, replay, provenance, and local naming are confined to H, C, P, N. Since no quotient-stream, selected-limit, trichotomy, choice, or completeness coordinate occurs in the packet, the listed rows exhaust the NameCert obligation surface. □

Theorem 5922.3 (RationalCauchyTelescope RegSeqRat factorization)**.** *Every Real-facing or Completion-facing read of an accepted* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *packet factors through the displayed chain*

$$Q \longmapsto W \longmapsto D \longmapsto M \longmapsto R,$$

before it can expose a $\mathsf{Real}^{\uparrow}$ *seal. Thus the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback is sourced by finite rational step rows, finite StreamName windows, and a dyadic telescoping modulus, not by quotient streams, selected representatives, trichotomy, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from an accepted carrier $T = (Q, D, W, R, M, H, C, P, N)$. The rational step rows Q can be inspected only through the finite windows W, and the same windows carry the dyadic tolerance row D.

The telescoping modulus M is read after Q, W, D. It is the row that converts finite step sums into tail comparisons suitable for the regular rational readback.

Finally the readback row R is the only $\mathsf{RegSeqRat}^{\uparrow}$ coordinate in the packet. Any Real-facing seal therefore receives its regular rational name through R after the finite telescoping route has been displayed.

Because the carrier has no further coordinate for quotient stream equality, selected representatives, trichotomy, choice, or ambient completeness, no completion consumer can bypass the displayed factorization. □

Theorem 5922.4 (RationalCauchyTelescope dyadic budget composition)**.** *Let*

$$T_i = (Q_i, D_i, W_i, R_i, M_i, H_i, C_i, P_i, N_i)$$

for $i = 0, 1$ *be accepted* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *packets whose rational telescope segments are adjacent along the displayed finite window boundary. Their composition is accepted by concatenating the rational step rows, adding only the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget rows, and preserving the*

$\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff before the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The composed readback still factors through*

$$Q_0, Q_1 \longmapsto W_0, W_1 \longmapsto D_0, D_1 \longmapsto M \longmapsto R.$$

Proof. Project the two telescope carriers. Each packet exposes finite rational step rows, a dyadic tolerance ledger, a finite StreamName window, a RegSeqRat readback, a telescoping modulus, and the structural rows.

Use the shared finite window boundary to concatenate the rational step rows. The composed row is still a finite $\mathsf{Rat}^{\uparrow}$ ledger, and the observation stages remain those displayed by W_0 followed by W_1.

Combine the dyadic tolerance ledgers through the displayed budget coordinate. The composed modulus row records only the sum of the visible $\mathsf{DyadicRatCore}^{\uparrow}$ budgets needed for the adjacent segments.

Finally replay Definition 5922.3 for the composed packet. The $\mathsf{RegSeqRat}^{\uparrow}$ readback is reached after the finite rational rows, StreamName windows, dyadic budgets, and telescoping modulus have been displayed, so no quotient stream, selected representative, trichotomy, choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate enters the composition. □

Closure status for $\mathsf{NameCert}_{\mathsf{RationalCauchyTelescope}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalCauchyTelescope}^{\uparrow}$ packet is generated from RatUp rational step rows, a DyadicRatCoreUp tolerance ledger, StreamName finite windows, a RegSeqRatUp readback, a telescoping modulus row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RationalCauchyTelescope}^{\uparrow}}$, seed)

Scope closed: Definition 5922.1, Definition 5922.2, and Definition 5922.3 seed the finite telescoping route from rational step rows to regular rational readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RationalCauchyTelescope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, trichotomy, choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked rational-step admission, dyadic tolerance stability, finite-window exposure, regular rational readback, telescoping-modulus discharge, and ledger non-escape for BEDC.Derived.RationalCauchyTelescopeUp.

Every $\mathsf{RationalCauchyTelescope}^{\uparrow}$ completion-facing consumer reads the rational step rows through chapter 1585 after the dyadic, StreamName, RegSeqRat, and RealUp rows of Definition 5922.1, and before any terminal Real-facing seal is public. This route keeps the completion-reflector dependency on the displayed telescope surface: $Q, D, W, R, M, H, C, P, N$ remain the only rows available to the consumer, and no quotient stream equality, selected limit, choice principle, trichotomy decision, or ambient $\mathsf{Real}^{\uparrow}$ completeness row is exported by the hub.

Chapter 5923

A Concrete Naming Certificate for $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ packet records the finite diagonal-tail telescope by which a family of regular Cauchy rational readbacks is inspected along a displayed tail before the terminal real-name seal is available. It refines the rational telescoping route of chapter 5922 through the dyadic tolerance core of chapter 291, the finite observation windows of chapter 126, the regular rational handoff of chapter 301, and the Real seal boundary of chapter 38. The packet is a finite L10 diagonal-tail route for Real and Completion consumers; it is not quotient stream equality, a selected host limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5923.1 (Bishop diagonal Cauchy telescope carrier)**.** *A* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, D, W, G, M, L, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *source row naming the finite rational telescoping schedule being diagonalized.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic budget for each accepted diagonal-tail comparison.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the observed tail windows and the diagonal index at which each window begins.*
4. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the windows in W and the rational telescope row R.*
5. *M is the displayed diagonal modulus row, saying that later diagonal requests move to weakly later windows and preserve the same dyadic budget discipline.*
6. *L is the finite tail-link ledger tying each accepted diagonal step to the preceding telescope row and the next regular rational readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, D, W, G, M, L have been displayed.*
8. *H records componentwise* hsame *transport for the telescope, dyadic, window, readback, diagonal modulus, tail-link, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route from R through D, W, G, M, L to E.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$, $\mathsf{RationalCauchyTelescope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, D, W, G, M, L, E, H, C, P, N$. It contains no coordinate for quotient equality of streams, selected representatives, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5923.2 (Bishop diagonal Cauchy telescope NameCert obligations)**.** *For every accepted* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *carrier $T = (R, D, W, G, M, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauch}}$ *surface consists exactly of rational telescope admission, dyadic tolerance exposure, finite tail-window exposure, regular rational readback, diagonal-modulus exactness, finite tail-link composition, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$\mathsf{RationalCauchyTelescope}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

before using the diagonal-tail telescope as evidence for a completed real name.

Proof. Project the carrier of Definition 5923.1. The rows R, D, W, G, M, L are the only pre-seal mathematical data, and they respectively expose the rational telescope, dyadic budget, finite tail windows, regular rational readback, diagonal modulus, and tail-link ledger.

The diagonal step is finite. A consumer reads R through the windows W, measures the displayed tail by D, and then obtains the regular rational row G. The modulus row M restricts later requests to weakly later windows, while L records the finite chain of accepted tail links.

The seal row E is downstream of those displayed rows. The structural rows H, C, P, N only transport, replay, source, and name the same route, so the local NameCert surface has no remaining coordinate from which quotient streams, selected representatives, countable choice, host real equality, or ambient completeness could be read. □

Theorem 5923.3 (Bishop diagonal Cauchy telescope tail route)**.** *Every diagonal-tail read exported by an accepted* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *packet factors through the finite route*

$$R, D, W, G, M, L \longmapsto E \longmapsto H, C, P, N.$$

Thus a Real or Completion consumer receives a $\mathsf{RationalCauchyTelescope}^{\uparrow}$ *source row, one* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, one* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row, one* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a displayed diagonal modulus, a finite tail-link ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming before it may use the diagonal-tail telescope. The route exports no quotient stream equality, selected representative, selected host limit, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted carrier $T = (R, D, W, G, M, L, E, H, C, P, N)$. The rational telescope row R is not readable as a Real-facing object until the dyadic row D and the finite window row W have displayed the inspected tail.

The regular rational readback G is then obtained on those same windows. The diagonal modulus M keeps later reads on weakly later windows, and the link ledger L records the finite composition of the accepted tail steps.

Apply the NameCert obligations of Definition 5923.2. They force the Real seal E and the structural rows H, C, P, N to remain downstream of the displayed tail route. Since the carrier contains no further coordinate, no downstream consumer can replace the route by quotient stream equality, selected representatives, selected limits, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ Real or Completion consumer factors through one RationalCauchyTelescope source row, one DyadicRatCore tolerance ledger, one StreamName finite tail-window row, one RegSeqRat readback row, one diagonal-modulus row, one finite tail-link ledger, one RealUp seal handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert before it may use a diagonal-tail telescope.

Independence witness: rationalcauchytelescope, dyadicratcore, streamname, regseqrat, real: $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately supply rational telescoping, tolerance budgets, finite windows, regular rational readback, and terminal Real sealing, while this packet uniquely binds them into one finite diagonal-tail route.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ packet is generated from a RationalCauchyTelescope source row, a DyadicRatCore tolerance ledger, StreamName finite tail windows, a RegSeqRat readback row, a diagonal modulus, a finite tail-link ledger, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5923.1, Definition 5923.2, and Definition 5923.3 seed the finite diagonal-tail telescope route through RationalCauchyTelescope, DyadicRatCore, StreamName, RegSeqRat, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, selected host limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite tail-window readback, RegSeqRat handoff, diagonal-modulus exactness, finite tail-link composition, Real seal handoff, structural transport, and ledger non-escape for BEDC.Derived.BishopDiagonalCauchyTelescopeUp.

Theorem 5923.4 (Bishop diagonal Cauchy telescope RegSeqRat route)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *packet must read the rational telescope route,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal in that order before the diagonal tail is public. In carrier notation, the consumer reads*

$$R, \quad D, \quad W, \quad G, \quad E$$

inside $T = (R, D, W, G, M, L, E, H, C, P, N)$*, with the diagonal modulus* M*, tail-link ledger* L*, transport* H*, replay* C*, provenance* P*, and local* NameCert *row* N *preserving the same route.*

Proof. Project the carrier from Definition 5923.1. The rational telescope row R, dyadic tolerance ledger D, finite-window row W, regular rational readback G, and terminal real seal E are the only Real-facing analytic rows available to the packet.

Apply the NameCert obligation surface of Definition 5923.2. It forces any Real or Completion consumer to expose the rational telescope, dyadic tolerance, finite tail windows, RegSeqRat readback, diagonal modulus, and finite tail-link composition before using the completed real-name seal.

Finally read the tail route of Definition 5923.3. The structural rows H, C, P, N transport, replay, source, and name that ordered route, so no quotient stream equality, selected host limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness is available to the consumer. □

Theorem 5923.5 (Bishop diagonal Cauchy telescope regular readback forward link)**.** *Every accepted* $\mathsf{BishopDiagonalCauchyTelescope}^{\uparrow}$ *packet reads its regular rational component through chapter 301 before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is available. Consequently every downstream Real or Completion consumer inherits the dependency on the displayed* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* G *inside the carrier* $T = (R, D, W, G, M, L, E, H, C, P, N)$.

Proof. Project the carrier of Definition 5923.1. The row G is explicitly the $\mathsf{RegSeqRat}^{\uparrow}$ readback row, and the seal row E is listed only after the rational telescope, dyadic budget, finite windows, readback, diagonal modulus, and tail-link ledger.

Apply Definition 5923.4. It forces Real and Completion consumers to read R, D, W, G, E in order, with the remaining rows preserving that same route.

Thus the regular rational component is not an optional sibling citation. It is a forward dependency carried by every accepted diagonal-tail packet before any terminal $\mathsf{Real}^{\uparrow}$ seal or Completion-facing consequence may be consumed. □

Chapter 5924

A Concrete Naming Certificate for $\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}$

Following Bishop–Bridges completion and the Kelley–Bourbaki language of Hausdorff reflection, the $\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}$ packet records the finite return map from a Hausdorff-reflected completion surface back to its separated completion read. It consumes the unit row of chapter 1527, the counit row of chapter 1539, the reflection row of chapter 403, and the separated-completion boundary of chapter 1612. The chapter names the local counit interface only as displayed BEDC rows; it does not add quotient representatives, host equality of completed points, selected limits, countable choice, or ambient completeness.

Definition 5924.1 (Hausdorff reflection completion counit carrier)**.** *A* $\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, V, R, S, W, Q, D, E, H, C, P, N)$$

with the following visible rows:

1. *U is the* $\mathsf{CauchyCompletionUnit}^{\uparrow}$ *admission row for the displayed source entering the completion surface.*
2. *V is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *return row for already displayed completion reads.*
3. *R is the* $\mathsf{CompletionReflection}^{\uparrow}$ *row that reflects completion-facing consumers through the separated boundary.*
4. *S is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *boundary row inspected before Hausdorff equality is exported.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened before rational observations are read.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the finite windows and readback comparisons.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached after the separated completion read has passed through W, D, Q.*
9. *H records componentwise* hsame *transport for U, V, R, S, W, Q, D, E and the structural rows.*
10. *C records displayed* Cont *replay for consumers that inspect the reflected counit, the separated boundary, or the terminal real-facing route.*

11. *P is the* Pkg *provenance over* HausdorffReflectionCompletionCounit$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionCounit$^{\uparrow}$, CompletionReflection$^{\uparrow}$, SeparatedCompletion$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{HausdorffReflectionCompletionCounit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-completion equality, selected completion representatives, host equality of completed objects, countable choice, or an ambient completeness theorem.

Lean4 ✓ BEDC.Derived.HausdorffReflectionCompletionCounitUp.HausdorffReflectionCompletionCounitTasteGate
single carrier alignment

Theorem 5924.2 (Hausdorff reflection completion counit obligations)**.** *For an accepted* HausdorffReflectionComplet *carrier $K = (U, V, R, S, W, Q, D, E, H, C, P, N)$, the local* NameCert$_{\text{HausdorffReflectionCompletionCounit}^{\uparrow}}$ *surface consists exactly of the carrier admission row, unit admission row, counit return row, completion-reflection row, separated-completion boundary, finite-window row, regular-rational readback, dyadic tolerance ledger, terminal real seal, componentwise transport, continuation replay, package provenance, and local naming row.*

Proof. Project the carrier of Definition 5924.1. Its coordinates are exactly $U, V, R, S, W, Q, D, E, H, C, P, N$.

The completion-facing part is the route through U, V, R, S. The unit row admits the source into the completion surface, the counit row returns displayed completion reads, the reflection row routes the read through the Hausdorff boundary, and the separated-completion row is the boundary at which equality can be consumed.

The real-facing part is the finite route W, Q, D, E. These rows expose stream windows, regular-rational readback, dyadic tolerances, and the terminal real seal as data of the same packet.

Finally, H, C, P, N transport, replay, source, and name the displayed rows. Since no further coordinate exists, the local certificate exports no quotient completion, selected representative, host equality, countable-choice schedule, or ambient completeness principle. □

Theorem 5924.3 (Hausdorff reflection completion counit compatibility)**.** *Let $K = (U, V, R, S, W, Q, D, E, H, C, P,$ be an accepted* HausdorffReflectionCompletionCounit$^{\uparrow}$ *carrier. Every consumer of the reflected completion counit factors through the displayed route*

$$U, V \longmapsto R \longmapsto S \longmapsto W, Q, D \longmapsto E \longmapsto H, C, P, N.$$

Thus the counit read is compatible with both the CompletionReflection$^{\uparrow}$ *route and the* SeparatedCompletion$^{\uparrow}$ *boundary: it returns the same finite real-facing seal that the counit row already exposes, and it does not replace that seal by quotient representatives, host equality, selected limits, countable choice, or ambient completeness.*

Proof. Start with the unit and counit rows U, V. They are the only coordinates that admit a displayed source into the completion surface and return an already displayed completion read.

Next pass through the reflection row R. The carrier makes the reflected completion consumer read R before any separated equality boundary can be inspected, so the reflected counit remains attached to the same completion-facing data as U and V.

Now inspect the separated boundary S and the real-facing route W, Q, D, E. The finite-window row, regular-rational readback, dyadic tolerance ledger, and terminal real seal are coordinates of the same carrier, so the reflected counit reaches the same displayed seal as the original counit read.

The structural rows H, C, P, N preserve and name this route. They do not add quotient classes, selected representatives, host equality, countable choice, or ambient completeness; therefore every consumer factors through the displayed counit-compatible route. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}$ packet is generated from CauchyCompletionUnit admission, CauchyCompletionCounit return, CompletionReflection routing, SeparatedCompletion boundary, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5924.1, Definition 5924.2, and Definition 5924.3 seed the finite Hausdorff-reflected completion counit surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffReflectionCompletionCounit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-completion equality, selected completion representatives, host equality of completed objects, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, reflected counit compatibility, separated-boundary exactness, non-escape, and a Lean carrier formalization for BEDC.Derived.HausdorffReflectionCompletionCounitUp.

Chapter 5925

A Concrete Naming Certificate for MetaCICCandidateStrongNormalization$^{\uparrow}$

The MetaCICCandidateStrongNormalization$^{\uparrow}$ packet realizes the candidate-mediated boundary of chapter 9370 as finite MetaCIC evidence. It sits beside the normalization frontier of chapter 892, the decidability boundary of chapter 749, the closed-term substitution boundary of chapter 527, the authorized-generator recursor of chapter 525, and the normalization audit packet of chapter 750. The packet keeps candidate data, fuel, normal-form endpoints, refusal rows, transport, replay, provenance, and local naming visible before any strong-normalization read is accepted.

Definition 5925.1 (MetaCIC candidate strong-normalization carrier)**.** *A* MetaCICCandidateStrongNormalization$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, K, F, E, R, A, H, C, P, N, G)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term row, recording the term, context boundary, and closedness data consumed by the candidate route.*
2. *K is the candidate row, naming the reducibility candidate, closed-candidate wrapper, or candidate-set membership evidence used by the strong-normalization read.*
3. *F is the fuel row, displaying the finite reduction budget or staged descent ledger attached to the candidate read.*
4. *E is the normal-form endpoint row, recording the normal endpoint reached by the fuel-and-candidate route.*
5. *R is the refusal row, rejecting evidence-free strong normalization, missing-fuel endpoints, subject-reduction erasure, substitution erasure, global confluence, and closed-consistency shortcuts.*
6. *A is the audit-boundary row, reading* MetaCICNormalizationFrontier$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, ClosedTermSubstitutionBoundary$^{\uparrow}$, AuthorizedGeneratorRecursor$^{\uparrow}$, *and* MetaCICNormalizationAudit$^{\uparrow}$ *as sibling sources.*
7. *H records componentwise* hsame *transport for term, candidate, fuel, endpoint, refusal, audit-boundary, replay, provenance, TasteGate, and local naming rows.*
8. *C records displayed* Cont *replay from the strong-normalization request through $T, K, F, E, R, A, H, P, N, G$.*

9. *P is the* Pkg *provenance over* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$*, the five sibling anchors,* BHist*,* hsame*,* Cont*, and* NameCert*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}}$ *row.*
11. *G is the TasteGate obligation row requiring Lean support in* `TasteGate.lean` *for* Nontrivial *and* FieldFaithful *instances of the carrier.*

The classifier compares accepted packets only through $T, K, F, E, R, A, H, C, P, N, G$. It has no coordinate for evidence-free strong normalization, an endpoint without fuel, hidden subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Theorem 5925.2 (MetaCIC candidate strong-normalization NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$ *carrier*

$$S = (T, K, F, E, R, A, H, C, P, N, G),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}}$ *surface consists exactly of closed typed term admission through T, candidate evidence through K, fuel visibility through F, normal-form endpoint readback through E, refusal exactness through R, sibling audit-boundary anchoring through A, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, local naming through N, and TasteGate nontriviality and field faithfulness through G.*

Lean4 ✓ `BEDC.Derived.MetaCICCandidateStrongNormalizationUp.MetaCICCandidateStrongNormalizationCarrier namecert obligations`

Proof. Project the carrier of Definition 5925.1. The accepted packet displays the closed typed term, candidate evidence, fuel, endpoint, refusal, audit-boundary, transport, replay, provenance, local naming, and TasteGate rows.

The positive route begins with T, K, F, E. A strong-normalization read reaches a normal-form endpoint only after candidate evidence and a fuel ledger have both been displayed.

The refusal and audit rows are part of the same surface. The row R rejects missing-candidate, missing-fuel, and endpoint-only reads, while A preserves the normalization frontier, decidability boundary, closed-term substitution boundary, authorized-generator recursor, and normalization audit packet as named sibling sources.

Finally H, C, P, N, G transport, replay, source, name, and guard the carrier. Because G requires Nontrivial and FieldFaithful instances, a Lean realization must distinguish evidence-bearing packets from endpoint-only packets and preserve the displayed field order. □

Theorem 5925.3 (MetaCIC candidate strong-normalization endpoint refusal)**.** *Every strong-normalization read exported by an accepted* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$ *packet must display both a fuel row and a refusal row before the normal-form endpoint row can be consumed. Any claimed candidate strong-normalization read that omits fuel endpoint evidence or omits the refusal boundary is outside the carrier.*

Proof. Read the positive route through Definition 5925.2. The endpoint row E is downstream of the closed typed term, candidate, and fuel rows T, K, F.

Now read the negative route. The refusal row R remains visible alongside the endpoint row and rejects any route that bypasses candidate evidence, fuel, subject-reduction boundaries, substitution boundaries, confluence boundaries, or closed-consistency boundaries.

Transport and replay do not alter the route. The rows H, C, P, N, G preserve the same ordered packet, so a consumer cannot turn the endpoint into an evidence-free theorem or a fuel-free normal-form claim. □

Theorem 5925.4 (MetaCIC candidate strong-normalization fuel-refusal determinacy)**.** *For every accepted* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$ *carrier*

$$S = (T, K, F, E, R, A, H, C, P, N, G),$$

if two strong-normalization reads expose the same closed typed term row T, candidate row K, fuel row F, endpoint row E, and refusal row R, then their exported finite endpoint surface is the same displayed packet route through A, H, C, P, N, G. No such read can split into an evidence-free endpoint, omit the refusal boundary, or export a different endpoint surface while keeping T, K, F, E, R fixed.

Proof. Project the accepted carrier of Definition 5925.1. The rows T, K, F, E, R are the only displayed term, candidate, fuel, endpoint, and refusal coordinates available to a strong-normalization read.

Compare the two reads on those five rows. Since the closed typed term, candidate evidence, fuel ledger, endpoint row, and refusal row are the same, both reads consume the same finite positive route and the same refusal boundary before any endpoint surface is public.

It remains to inspect the structural rows. The audit row A names the same sibling frontier, decidability, substitution, recursor, and normalization-audit sources; H transports the same components; C replays the same displayed route; P records the same provenance; N binds the same local name; and G enforces the same TasteGate field order.

Therefore the exported endpoint surface is componentwise identical through A, H, C, P, N, G. An evidence-free endpoint or a missing-refusal endpoint would require a coordinate outside S, and no such coordinate exists in the carrier. □

Theorem 5925.5 (MetaCIC candidate strong-normalization candidate-fuel endpoint descent)**.** *Let $S = (T, K, F, E, R, A, H, C, P, N, G)$ be an accepted* $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$ *carrier. Any accepted carrier read exposing the same closed typed term row T, candidate row K, and fuel row F reaches the endpoint row E only through the displayed descent ledger and the refusal row R. Decreasing or replaying the fuel ledger preserves the same endpoint boundary through A, H, C, P, N, G and cannot yield evidence-free strong normalization, global confluence, or subject reduction.*

Proof. Project the carrier of Definition 5925.1. The positive route displays the closed typed term, candidate evidence, fuel ledger, and endpoint rows T, K, F, E in that order.

Read the refusal route at the same time. The row R rejects any endpoint read that omits candidate evidence, omits fuel, or turns the endpoint into an evidence-free strong-normalization theorem.

Now compare a decreased or replayed fuel ledger with the original accepted read. The structural rows A, H, C, P, N, G preserve the sibling audit boundary, componentwise transport, replay, provenance, local naming, and TasteGate field order without adding confluence or subject-reduction coordinates.

Apply Definition 5925.4 to rule out an alternate endpoint surface once T, K, F, E, R are fixed. Hence every accepted descent-facing read reaches the same endpoint boundary through the displayed ledger and refusal rows. □

Theorem 5925.6 (MetaCIC candidate strong-normalization TasteGate boundary)**.** *Lean support for* MetaCICCandidateStrongNormalization$^{\uparrow}$ *must provide* Nontrivial *and* FieldFaithful *instances in* `TasteGate.lean`*. Nontriviality distinguishes a carrier with candidate, fuel, endpoint, and refusal rows from a closed-term endpoint without those rows; field faithfulness preserves the field order*

$$T \longmapsto K \longmapsto F \longmapsto E \longmapsto R \longmapsto A \longmapsto H, C, P, N, G$$

before any carrier-alignment theorem can count as the Lean realization.

Proof. Nontriviality is forced by the carrier boundary: deleting K, F, or R leaves a closed typed term and an endpoint-like read, but not an accepted candidate strong-normalization packet.

Field faithfulness is the route-preservation condition. If a projection erases or permutes $K, F, E, R, A, H, C, P, N$ the displayed carrier no longer records the candidate-mediated strong-normalization boundary of Definition 5925.1. □

Falsifiable prediction: Any claimed MetaCICCandidateStrongNormalization$^{\uparrow}$ read that reaches a normal-form endpoint without both the fuel row and the refusal row refutes the carrier.

Independence witness: metacicnormalizationfrontier, metacicdecidableboundary, closedtermsubstitutionboundary, authorizedgeneratorrecursor, metacicnormalizationaudit: MetaCICCandidateStrongNormalizatio is not bijective to any listed sibling because it jointly carries candidate evidence, fuel, normal-form endpoint, refusal, sibling audit route, transport, replay, provenance, TasteGate, and local NameCert rows.

Closure status for NameCert$_{\text{MetaCICCandidateStrongNormalization}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICCandidateStrongNormalization}^{\uparrow}}$, seed)

Scope closed: Definition 5925.1, Definition 5925.2, Definition 5925.3, Definition 5925.4, and Definition 5925.6 seed the candidate-mediated finite strong-normalization boundary over term, candidate, fuel, endpoint, refusal, audit, transport, replay, provenance, TasteGate, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICCandidateStrongNormalization}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.MetaCICCandidateStrongNormalizationUp.MetaCICCandidateStrongNormalizationCarrier\_namec`

Bridge status: none

Not claimed: This chapter does not close unconditional strong normalization for all closed MetaCIC terms, unrestricted subject reduction, arbitrary substitution preservation, global confluence, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate visibility, fuel endpoint exactness, refusal exactness, sibling audit anchoring, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICCandidateStrongNormalizationUp.

Constructive story: A MetaCICCandidateStrongNormalization$^{\uparrow}$ packet is generated from a closed typed term row, candidate evidence, fuel row, normal-form endpoint row, refusal row, sibling audit-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, TasteGate obligations, and a local NameCert row.

Chapter 5926

A Concrete Naming Certificate for $\mathsf{RealDiagonalModulus}^{\uparrow}$

Following Bishop–Bridges style constructive diagonal arguments, the $\mathsf{RealDiagonalModulus}^{\uparrow}$ packet records the rank-one modulus row used when a regular rational diagonal route is read as a real-name construction. It sits between the finite selector packet of chapter 375, the finite stream surface of chapter 126, the regular rational readback of chapter 301, the dyadic observation core of chapter 291, and the terminal real-name seal of chapter 38. The packet isolates the single displayed modulus schedule consumed by regular-Cauchy and $\mathsf{Real}^{\uparrow}$ completion routes without quotient streams, selected host representatives, countable choice, or ambient completeness.

Definition 5926.1 (Real diagonal modulus carrier)**.** *A* $\mathsf{RealDiagonalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (n, \mu, S, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *n is the unary precision-request row submitted by a* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer;*
2. *μ is the accepted* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector row naming the threshold inspected for that request;*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the selected threshold;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the displayed window S;*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radius, endpoint comparison, and diagonal-budget cells read by R;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, R, D have been displayed;*
7. *H is the componentwise* hsame *transport ledger for n, μ, S, R, D, E and for their finite subrows;*
8. *C is the* Cont *route ledger from precision request to modulus threshold, stream window, regular readback, dyadic tolerance, and real seal;*
9. *Q is the* Pkg *provenance row over* $\mathsf{RealDiagonalModulus}^{\uparrow}$, $\mathsf{DiagonalModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalModulus}^{\uparrow}}$ *naming row.*

The classifier on such packets is componentwise transport through H with replay along C. It never compares hidden streams, quotient representatives, host real endpoints, selected limits, or ambient complete spaces.

Lean4 ✓ `BEDC.Derived.RealDiagonalModulusUp.RealDiagonalModulusTasteGate single carrier alignment`

Theorem 5926.2 (RealDiagonalModulusUp NameCert obligations)**.** *For every accepted* $\mathsf{RealDiagonalModulus}^{\uparrow}$ *carrier* $P = (n, \mu, S, R, D, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealDiagonalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, classifier transport through* H*, route replay through* C*, selector visibility through* μ*, stream-window visibility through* S*, regular-rational readback through* R*, dyadic tolerance exactness through* D*, real-seal non-escape through* E*, package provenance through* Q*, and local naming through* N*.*

Proof. Carrier admission is Definition 5926.1. It lists the precision request, modulus selector, stream window, regular readback, dyadic ledger, real seal, transport ledger, continuation ledger, provenance row, and local naming row.

The selector and readback obligations are inherited by projection. The row μ is the only $\mathsf{DiagonalModulus}^{\uparrow}$ selector coordinate, S is the only $\mathsf{StreamName}^{\uparrow}$ window coordinate, R is the only $\mathsf{RegSeqRat}^{\uparrow}$ readback coordinate, and D is the only $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance coordinate.

The transport and route obligations are exactly H and C. Any consumer that moves the packet must move each displayed row componentwise and then replay the same route from n through μ, S, R, D to E.

Finally, Q and N close the local naming surface. Since no other coordinate occurs in the carrier, the certificate cannot export quotient stream equality, selected host representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5926.3 (RealDiagonalModulusUp real-seal route)**.** *Let* $P = (n, \mu, S, R, D, E, H, C, Q, N)$ *be an accepted* $\mathsf{RealDiagonalModulus}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *consumer of the seal row* E *first reads the displayed precision request* n*, the* $\mathsf{DiagonalModulus}^{\uparrow}$ *selector* μ*, the* $\mathsf{StreamName}^{\uparrow}$ *finite window* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* D*, with transport and provenance supplied by* H, C, Q, N*. Thus the real-facing handoff is a finite diagonal-modulus route rather than a quotient stream, selected representative, choice sequence, host equality proof, or ambient completeness principle.*

Proof. Start with the carrier projection. The only row whose type can feed a terminal real-name read is E, and the carrier places it after the displayed diagonal route n, μ, S, R, D.

Replay the route ledger C. A precision request opens the $\mathsf{DiagonalModulus}^{\uparrow}$ selector, the selector opens the finite $\mathsf{StreamName}^{\uparrow}$ window, the window is read back by R, and the dyadic tolerance data in D records the comparison budget before E is exposed.

Transport is controlled by H, while Q and N record the package and local certificate names. Therefore every real-seal consumer factors through the same finite route and cannot read a quotient stream, selected host representative, choice sequence, host equality proof, or ambient complete space. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealDiagonalModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealDiagonalModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5926.1, Definition 5926.2, and Definition 5926.3 seed the rank-one diagonal-modulus route from $\mathsf{DiagonalModulus}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealDiagonalModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close quotient stream equality, selected representatives, countable choice, host real equality, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, selector visibility, stream readback, dyadic tolerance exactness, real-seal non-escape, and package provenance for the same packet.

Constructive story: The construction starts from a displayed precision request, reads one diagonal modulus selector, opens one finite stream window, reads the regular rational row and dyadic tolerance ledger, and then exposes the real seal as finite NameCert data.

Chapter 5927

A Concrete Naming Certificate for $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$

Following the compact-metric uniform-continuity theorem as presented in Munkres and in Bishop–Bridges, the $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$ packet records the finite certificate surface that turns a displayed compact metric homeomorphism into an inverse-uniform-continuity route. It consumes the finite cover and total-boundedness rows of chapter 129 and chapter 127, the function and continuity rows of chapter 130, the distance interface of chapter 57, and the terminal real-name seal of chapter 38. The inverse modulus is read from finite-net and separated-target rows; it is not read from choice, open-cover compactness, host equality, or ambient Real completeness.

Definition 5927.1 (CompactUniformHomeomorphism carrier)**.** *A* $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, M, T, F, G, A, B, E, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. X, Y *are the source and target* $\mathsf{Met}^{\uparrow}$ *carrier rows.*
2. M *is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-cover row for the source metric space.*
3. T *is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row read from* M *at each displayed dyadic scale.*
4. F *is the forward* $\mathsf{ContinuousMap}^{\uparrow}$ *row and* G *is the displayed inverse map row on the same named carrier windows.*
5. A *records the finite graph-window agreement that* $G(F(x))$ *and* $F(G(y))$ *return to the displayed source and target windows through componentwise classifier reads.*
6. B *is the separated-target ledger: for every finite net row and positive target separation request it displays the finite list of source cells whose forward images are classifier-separated in* Y*.*
7. E *is the inverse-modulus row obtained by refining* T *until the separated-target ledger* B *accepts the target tolerance.*
8. S *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened for source and target observations.*
9. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational metric observations exposed by* S*.*
10. R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for all distance and tolerance rows.*

11. *H records componentwise* hsame *transport for the metric, compact, totally bounded, continuity, graph-agreement, separated-target, inverse-modulus, stream, regular-rational, Real, replay, provenance, and local naming rows.*

12. *C records displayed* Cont *replay through*

$$X, Y, M, T, F, G, A, B, E, S, Q, R \longmapsto \mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}.$$

13. *P is the* Pkg *provenance over* $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, M, T, F, G, A, B, E, S, Q, R, H, C, P, N$. The carrier has no coordinate for open-cover compactness, selected nearest net points, quotient equality of functions, host equality, an ambient inverse map, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformHomeomorphismUp.CompactUniformHomeomorphismTasteGate single carrier alignment`

Theorem 5927.2 (CompactUniformHomeomorphism NameCert surface)**.** *For every accepted* CompactUniformHon *carrier*

$$U = (X, Y, M, T, F, G, A, B, E, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}$ *surface consists exactly of the metric-space rows X, Y, the compact-metric finite-cover row M, the totally-bounded finite-net row T, the forward and inverse graph rows F, G, A, the separated-target ledger B, the inverse-modulus row E, the finite stream and regular-rational rows S, Q, the terminal Real seal R, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and the local name N.*

Consequently every inverse-uniform-continuity read exported by the packet must pass through the sibling $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before any consumer may inspect the inverse modulus. The surface exports no open-cover compactness theorem, no selected finite subcover, no quotient equality of maps, no host equality, no choice function, and no ambient Real-completeness witness.*

Lean4 ✓ `BEDC.Derived.CompactUniformHomeomorphismUp.CompactUniformHomeomorphismNameCertSurface`

Proof. Project the accepted carrier. Its public rows are precisely the metric, compact, total-boundedness, continuity, graph-agreement, separation, modulus, stream, regular-rational, Real, transport, replay, provenance, and local naming rows listed in Definition 5927.1.

The finite-net rows are read from M through T, while the map rows are read from F, G, A. The separated-target ledger B compares finite target cells using only the metric classifier rows in Y and the displayed forward graph row.

The inverse-modulus row E is downstream of those finite rows. Transport, replay, provenance, and local naming are then supplied by H, C, P, N without adding another compactness or equality coordinate. □

Theorem 5927.3 (CompactUniformHomeomorphism inverse modulus)**.** *Let U be an accepted* $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$ *packet and let ε be a positive target tolerance row displayed through the target* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal. Then U displays a finite source tolerance row δ such that every pair of target observations whose source representatives are read through the accepted*

inverse graph row and whose target distance is below δ has source distance below ε on the displayed finite windows.

The row δ is obtained by refining the $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row of the* $\mathsf{CompactMetric}^{\uparrow}$ *source until the separated-target ledger accepts the target tolerance. It preserves the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph rows,* $\mathsf{Met}^{\uparrow}$ *distance rows,* $\mathsf{Real}^{\uparrow}$ *tolerance seal, componentwise* hsame*, displayed* Cont*,* Pkg*, and local* NameCert *support. The construction does not use choice, open-cover compactness, host equality, quotient equality of maps, selected limits, or ambient Real completeness.*

Proof. Start with the positive target tolerance row and open the finite-net row T supplied by the compact-metric source row M. Each refinement remains a displayed finite BEDC row with the same metric, stream, regular-rational, and Real seal support.

Use the separated-target ledger B to test the images of the finitely many source cells. When the ledger accepts the requested target separation, it records a finite source radius that prevents two distinct source cells from being identified by the inverse graph read.

Read that accepted radius as the inverse-modulus row E. The forward and inverse graph rows F, G, A then transport the finite target comparison back to the source metric rows, and H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Since every step uses displayed finite-net rows and separated-target rows, the modulus is not extracted from a choice operator, an open-cover compactness proof, host equality, quotient map equality, selected limits, or ambient Real completeness. □

Falsifiable prediction: Every $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$ inverse-modulus read factors through one CompactMetric finite-cover row, one TotallyBounded finite-net row, ContinuousMap graph rows, MetricSpace distance rows, RealUp tolerance seals, separated-target rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, metricspace, totallybounded, real: CompactUniformHom is not bijective to any listed sibling because it jointly carries finite-net refinement, separated-target comparison, inverse graph agreement, inverse-modulus extraction, Real tolerance sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformHomeomorphism}^{\uparrow}$ packet is generated from source and target MetricSpace rows, CompactMetric finite-cover data, TotallyBounded finite-net rows, ContinuousMap graph rows, separated-target comparison, inverse graph agreement, inverse-modulus extraction, stream and regular-rational readback, Real tolerance seals, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5927.1, Definition 5927.2, and Definition 5927.3 seed the finite compact-uniform-homeomorphism carrier, certificate, and inverse-modulus packet boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformHomeomorphism}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, quotient equality of functions or maps, host equality, choice functions, ambient inverse maps, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net refinement exactness, separated-target ledger exactness, inverse graph agreement, inverse-modulus extraction and non-escape, Real tolerance sealing, transport, replay, provenance, local NameCert coverage, package non-escape, and a Lean carrier formalization for BEDC.Derived.CompactUniformHomeom

Chapter 5928

A Concrete Naming Certificate for $\mathsf{LocatedMonotoneLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ packet records the finite route by which a located upper bound and a displayed monotone real sequence supply a regular Cauchy seal without importing a classical supremum, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. It sits after the reusable upper-bound witness of chapter 3396, the located-supremum monotone handoff of chapter 1427, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38.

Definition 5928.1 (LocatedMonotoneLimit carrier)**.** *A* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, D, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the monotone real-sequence row, carrying the inspected finite sequence windows and the displayed increasing comparison ledger.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to those finite windows.*
3. *U is the* $\mathsf{LocatedUpperBound}^{\uparrow}$ *row, naming the located upper-bound witness consumed by the sequence window.*
4. *W is the common precision-window row, recording the finite index at which the monotone window and upper-bound ledger are read together.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from S, D, U, W.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular readback R and from no other source.*
7. *H records componentwise* hsame *transport for the monotone sequence, dyadic tolerance, located upper bound, common window, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from every limit request through $S, D, U, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$, $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedMonotoneLimit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host supremum, a selected limit point, quotient real equality, trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedMonotoneLimitUp.LocatedMonotoneLimitTasteGate single carrier alignment`

Theorem 5928.2 (LocatedMonotoneLimit NameCert obligations)**.** *For every accepted* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ *carrier*

$$M = (S, D, U, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedMonotoneLimit}^{\uparrow}}$ *surface consists exactly of carrier admission, monotone-window exposure, located-upper-bound extraction, common-window compatibility, regular Cauchy readback, terminal Real-seal routing, componentwise transport, displayed replay, provenance, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 5928.1. The analytic rows visible before the seal are exactly the monotone sequence row S, dyadic tolerance row D, located upper-bound row U, common precision window W, and regular readback row R.

The located-upper-bound and monotone-window obligations are read before R. Thus the regular Cauchy readback is built from displayed finite rows rather than from a host supremum theorem or a selected limit.

Finally read the structural rows H, C, P, N. They transport, replay, source, and name the same packet componentwise, so the listed obligations exhaust the local certificate surface. □

Theorem 5928.3 (LocatedMonotoneLimit upper-bound extraction)**.** *Every accepted* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ *carrier exposes its upper-bound information only through the displayed* $\mathsf{LocatedUpperBound}^{\uparrow}$ *row* U *and the common precision window* W*. A completion-facing consumer must read* S, D, U, W *before it can inspect the regular readback* R *or the terminal Real seal* E*.*

Proof. The upper-bound coordinate of the carrier is the row U. By definition it is paired with the monotone sequence and tolerance rows through the window W, so no consumer can read an upper-bound witness detached from the inspected sequence window.

The readback row R is downstream of S, D, U, W. Therefore any completion-facing request that reaches R or E has already exposed the located upper-bound witness and the finite precision window that binds it to the monotone sequence. □

Theorem 5928.4 (LocatedMonotoneLimit regular Cauchy seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ *carrier factors through the finite route*

$$S, D, U, W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

The route records the constructive monotone-limit handoff as a regular rational Cauchy seal, not as a quotient completion, selected supremum, or ambient completeness theorem.

Proof. Start with the displayed monotone sequence, dyadic tolerance ledger, located upper-bound row, and common precision window. These rows are the only pre-readback coordinates of the carrier.

The row R repacks those finite rows as regular rational Cauchy data. The terminal row E is then a $\mathsf{Real}^{\uparrow}$ seal for that readback, and H, C, P, N preserve the same route under transport, replay, provenance, and naming.

Since the carrier has no quotient-completion, selected-supremum, or ambient-completeness coordinate, the exported seal is exactly the displayed regular Cauchy route. □

Theorem 5928.5 (LocatedMonotoneLimit nonescape)**.** *No accepted* $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ *consumer can extract a classical least upper bound, trichotomy decision, countable-choice selector, quotient representative, host sequence equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle from the packet. Every observable limit-facing row remains a read of* $S, D, U, W, R, E, H, C, P, N$*.*

Proof. Inspect the carrier coordinates. They list only the monotone sequence windows, dyadic tolerance ledger, located upper-bound witness, common precision window, regular readback, terminal seal, and structural rows.

The refused objects are absent from this list. Componentwise transport through H and replay through C preserve existing rows but do not add new coordinates, and provenance plus local naming record the same finite packet.

Consequently any accepted consumer remains inside the route $S, D, U, W, R, E, H, C, P, N$ and cannot export the refused classical or ambient data. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedMonotoneLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedMonotoneLimit}^{\uparrow}$ packet is generated from monotone real-sequence windows, a DyadicRatCore tolerance ledger, a LocatedUpperBound witness, a common precision window, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5928.1, Definition 5928.2, Definition 5928.3, Definition 5928.4, and Definition 5928.5 seed the finite constructive monotone-limit handoff from located upper-bound data to a regular Cauchy Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedMonotoneLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical supremum existence, trichotomy, countable choice, quotient representatives, host sequence equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window exposure, located-upper-bound extraction, common-window compatibility, regular Cauchy readback, Real-seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedMonotoneLimitUp.

Chapter 5929

A Concrete Naming Certificate for $\mathsf{HopfRinowFiniteGeodesic}^{\uparrow}$

Following Hopf and Rinow, the $\mathsf{HopfRinowFiniteGeodesic}^{\uparrow}$ packet records the finite BEDC surface on which a compact complete Riemannian metric presentation exposes geodesic segments only as named rows. It reads the complete-metric handoff of chapter 128, the geodesic metric surface of chapter 2213, the Riemannian metric row of chapter 133, the compact metric row of chapter 129, and the Real readback boundary of chapter 38. The packet is a finite geodesic NameCert neighbour for completion data; it does not assert classical compactness theorems, select minimizing geodesics globally, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5929.1 (Hopf–Rinow finite geodesic carrier)**.** *A* $\mathsf{HopfRinowFiniteGeodesic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (C, M, R, K, S, Q, A, H, T, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *row exposing the Cauchy-limit service used by finite geodesic endpoint reads.*
2. *M is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *row naming the displayed geodesic segment incidence surface.*
3. *R is the* $\mathsf{RiemannianMetric}^{\uparrow}$ *row whose tangent-fibre metric data supplies the local length ledger.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row whose finite-net and coverage witnesses bound the inspected endpoint windows.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row listing the endpoint, subdivision, and observation windows opened by the packet.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational length, radius, and tolerance observations exposed by S.*
7. *A is the* $\mathsf{Real}^{\uparrow}$ *seal for the finite distance and length comparisons read through Q.*
8. *H records componentwise* hsame *transport for the complete, geodesic, Riemannian, compact, stream, readback, Real, replay, provenance, and naming rows.*
9. *T records displayed* Cont *replay from endpoint windows through geodesic incidence, finite length comparisons, compact coverage, and complete-metric handoff.*

10. *P is the* Pkg *provenance over* HopfRinowFiniteGeodesic$^{\uparrow}$, CompleteMetric$^{\uparrow}$, GeodesicMetric$^{\uparrow}$, RiemannianMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{HopfRinowFiniteGeodesic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open-cover compactness theorem, a global minimizing-geodesic selector, quotient path equality, an unbounded Hopf–Rinow equivalence, or ambient Real$^{\uparrow}$ *completeness outside the displayed finite readback route.*

Lean4 ✓ `BEDC.Derived.HopfRinowFiniteGeodesicUp.HopfRinowFiniteGeodesicTasteGate single carrier alignment`

Theorem 5929.2 (Hopf–Rinow finite geodesic NameCert obligations)**.** *For every accepted* HopfRinowFiniteGeodesic *carrier $G = (C, M, R, K, S, Q, A, H, T, P, N)$, the local* NameCert$_{\text{HopfRinowFiniteGeodesic}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete-metric row, geodesic-metric incidence row, Riemannian metric row, compact-metric row, finite stream windows, regular-rational readback, Real comparison seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every finite geodesic read must expose $C, M, R, K, S, Q, A, H, T, P, N$ before it can be consumed by a completion or metric-geometry neighbour.*

The packet exports no classical Hopf–Rinow equivalence, no chosen minimizing geodesic for arbitrary endpoints, no open-cover compactness principle, no quotient path completion, and no ambient completeness theorem for Real$^{\uparrow}$.

Proof. Project the carrier of Definition 5929.1. The complete, geodesic, Riemannian, and compact faces occur only as the rows C, M, R, K, so a downstream reader cannot replace them by host compactness or a global geodesic theorem.

Read the finite observations through S, Q, A. The stream row displays the endpoint and subdivision windows, the regular-rational row reads the length and radius comparisons opened by those windows, and the Real row seals only those finite comparisons.

The structural rows H, T, P, N then transport, replay, source, and name the same packet. Since no other coordinate occurs in the carrier, the local NameCert obligations cannot import open-cover compactness, a selected minimizing-geodesic function, quotient path equality, an unbounded Hopf–Rinow equivalence, or ambient Real completeness. □

Falsifiable prediction: Every accepted HopfRinowFiniteGeodesic$^{\uparrow}$ finite geodesic read factors through displayed CompleteMetricUp, GeodesicMetricUp, RiemannianMetricUp, CompactMetricUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: completemetric, geodesic-metric, riemannianmetric, compactmetric, streamname, regseqrat, real: HopfRinowFiniteGeodesic$^{\uparrow}$ is not bijective to any listed sibling because it carries the finite geodesic endpoint route together with completion, compact coverage, Riemannian length, rational readback, and local naming rows in one packet.

Closure status for NameCert$_{\text{HopfRinowFiniteGeodesic}^{\uparrow}}$.

Constructive story: A HopfRinowFiniteGeodesic$^{\uparrow}$ packet is generated from CompleteMetricUp handoff rows, GeodesicMetricUp incidence rows, RiemannianMetricUp length rows, CompactMetricUp finite-net rows, StreamName endpoint windows, RegSeqRat readback, RealUp comparison seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{HopfRinowFiniteGeodesic}^{\uparrow}}$, seed)

Scope closed: Definition 5929.1 and Definition 5929.2 seed the finite geodesic NameCert surface over completion, compactness, Riemannian metric, and rational readback rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{HopfRinowFiniteGeodesic}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim the classical Hopf–Rinow theorem, global minimizing geodesic selection, open-cover compactness, quotient path completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite endpoint-window replay, geodesic incidence stability, Riemannian length readback, compact coverage routing, complete-metric handoff, Real comparison exactness, transport, replay, provenance, and local naming for BEDC.Derived.HopfRinowFiniteGeodesicUp.

EffectiveCauchyCompletion$^{\uparrow}$ reads its completion route through chapter 126, chapter 301, chapter 291, and chapter 38; the child files below keep the carrier and L10 dependency rows separate.

Chapter 5930

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges and Weihrauch on constructive completion, the $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ packet records the finite L10 route by which regular rational Cauchy data becomes a completion-facing name without quotienting streams or selecting representatives. It sits after the dyadic tolerance core of chapter 291, the finite window discipline of chapter 126, the regular rational readback of chapter 301, the functorial Cauchy-completion route of chapter 423, and the terminal real-name seal of chapter 38. The certificate names the displayed effective completion surface only; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, quotient equality of Cauchy names, countable choice, selected limits, or host equality of completed objects.

The completion dependency is substantive: $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ consumes the $\mathsf{RealCauchyCompletion}^{\uparrow}$ seal route of chapter 360 and the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ handoff of chapter 423 only through the visible $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{StreamName}^{\uparrow}$ finite-window rows. The consumed NameCert rows are the carrier rows W, R, D, F, S below: finite observation window, regular rational readback, dyadic tolerance, functorial completion route, and terminal $\mathsf{Real}^{\uparrow}$ seal. Thus the packet records a completion-lattice dependency rather than a kernel-only shortcut.

Definition 5930.1 (Effective Cauchy completion carrier)**.** *An* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (Q, W, R, D, F, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the rational source row, recording the finite rational object or rational metric source being completed.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing the finite observation schedule used by completion consumers.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, recording the regular rational Cauchy data visible on each requested window.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, storing the dyadic error budget that certifies regularity and comparison.*
5. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *functorial route row, recording the displayed map from source data into completion-facing names.*

6. *S is the* $\mathsf{Real}^{\uparrow}$ *seal row, recording the terminal real-name boundary consumed by downstream L10 readers.*
7. *H records componentwise* hsame *transport for $Q, W, R, D, F, S, C, P, N$.*
8. *C records displayed* Cont *replay from a completion request through $Q, W, R, D, F, S, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected stream representatives, selected limits, countable choice, host equality of completed objects, or ambient real completeness.

Theorem 5930.2 (Effective Cauchy completion NameCert obligations)**.** *For every accepted* EffectiveCauchyComple *carrier*

$$E = (Q, W, R, D, F, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the rational source row, finite stream-window row, regular rational readback row, dyadic tolerance row, functorial completion route, terminal real-name seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every effective completion read must expose Q, W, R, D, F, S before H, C, P, N can be consumed.*

Proof. Project the carrier of Definition 5930.1. Its displayed coordinates are exactly the rational source, stream-window, regular-readback, dyadic-tolerance, functorial-completion, real-seal, transport, replay, provenance, and local naming rows.

The effective completion route is read through Q, W, R, D, F, S. The source row supplies the rational object, the window row supplies finite access, the regular readback row supplies Cauchy data, the dyadic row supplies the error budget, the functorial row records the completion passage, and the seal row presents the terminal real-name boundary.

The remaining rows H, C, P, N transport, replay, package, and name the same displayed route. Since quotient equality, selected representatives, selected limits, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 5930.3 (Effective Cauchy completion regular readback)**.** *For every accepted* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *carrier $E = (Q, W, R, D, F, S, H, C, P, N)$, any completion-facing query that asks for a regular Cauchy name must read the finite* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D before it can consume the functorial completion route F or the terminal real-name seal S. The readback is therefore finite, dyadic, and regular before any completed-real boundary is visible.*

Proof. Begin with Definition 5930.2. It requires every effective completion read to expose the source, window, readback, tolerance, functorial, and seal rows before the structural rows are consumed.

The requested regular Cauchy name is not a host-level limit. It is the displayed triple W, R, D: finite windows determine what is observed, regular rational data supplies the Cauchy values, and the dyadic tolerance row records the budget that makes the read regular.

Only after this triple is visible can the functorial route F and the real seal S be consumed. Thus the completed boundary is reached through finite regular readback rather than through quotient completion or selected representatives. □

Theorem 5930.4 (Effective Cauchy completion window telescope)**.** *For every accepted* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *carrier* $E = (Q, W, R, D, F, S, H, C, P, N)$*, any finite chain of completion-facing window refinements exposes the same route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The later seal row factors through the earlier displayed windows, regular readbacks, dyadic tolerance rows, and functorial completion rows. No refinement step adds quotient completion equality, a selected stream representative, countable choice, or an ambient completeness coordinate.

Proof. Project the carrier of Definition 5930.1. The only refinement-facing rows are the finite window row W, the regular rational readback row R, the dyadic tolerance row D, the functorial route row F, and the terminal seal row S.

Apply Definition 5930.2. Each completion-facing read must expose those rows in order before H, C, P, N are consumed, so a later finite window refinement can only refine data already displayed by W, R, D, F, S.

Apply Definition 5930.3. Regular readback is visible before the functorial completion route and the real seal, and the dyadic tolerance row remains the only error-budget coordinate.

Therefore the finite telescope is componentwise: later rows factor through earlier displayed windows by hsame transport and displayed Cont replay. Since the carrier contains no quotient, choice, selected-representative, or ambient-completeness coordinate, none can appear in the refined seal. □

Theorem 5930.5 (Effective Cauchy completion rational telescope route)**.** *For every accepted* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *carrier* $E = (Q, W, R, D, F, S, H, C, P, N)$*, every finite window refinement consumed by* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *may be read through the rational telescope sibling of chapter 5922 before the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row* F *and the* $\mathsf{Real}^{\uparrow}$ *seal row* S *are consumed. The route exposes* Q, W, R, D *as rational source, stream window, regular rational readback, and dyadic tolerance data, then passes through the telescope boundary before the functorial completion and terminal real-name rows are read.*

Proof. Project the carrier of Definition 5930.1. The finite completion-facing route is displayed as $Q, W, R, D, F, S, H, C, P, N$, so the rational source, window, regular readback, and tolerance rows are visible before the functorial and seal rows.

Apply Definition 5930.4. Any finite window refinement factors through the same stream-window, regular-readback, dyadic-tolerance, functorial-completion, and real-seal route. Apply Definition 5930.3. Regular rational readback is already exposed before the functorial completion route and the terminal seal are consumed.

The rational telescope sibling is finite, so inserting its read between Q, W, R, D and F, S does not add a quotient completion, a selected representative, a selected limit, countable choice, or host equality of completed objects. The support rows H, C, P, N then transport, replay, source, and name the same displayed route. □

Theorem 5930.6 (Effective Cauchy completion real seal scope)**.** *For every accepted* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *carrier* $E = (Q, W, R, D, F, S, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *row* S *seals only the displayed regular-Cauchy route over* Q, W, R, D, F*. It does not identify all Cauchy names by quotient equality, does not choose a canonical representative, and does not supply ambient completeness for unrelated real-name consumers.*

Proof. Apply Definition 5930.3. The seal row is consumed only after the finite window, regular rational readback, dyadic tolerance, and functorial completion rows have been exposed.

This confines the seal to the same displayed route. The carrier has no row for quotienting all Cauchy names, selecting a canonical representative, or proving an ambient completeness principle for arbitrary real-name consumers.

Finally use Definition 5930.2. The structural rows H, C, P, N transport, replay, package, and name the route already bounded by Q, W, R, D, F, S; they do not add a second completion surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompletion}^{\uparrow}}$.

Constructive story: An $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ packet is generated from a rational source row, finite stream-window row, regular rational readback row, dyadic tolerance row, functorial Cauchy-completion route, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5930.1, Definition 5930.2, Definition 5930.3, and Definition 5930.6 seed the finite effective-completion surface for L10 real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-window exposure, regular-readback exposure, dyadic tolerance confinement, functorial completion routing, terminal seal scope, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCauchyCompletionUp.

Theorem 5930.7 (Effective Cauchy completion L10 dependency route)**.** *Every* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *completion read consumes its L10 dependency lattice before it exports completion evidence: finite windows from chapter 126, regular rational Cauchy readback from chapter 301, terminal real-name sealing from chapter 38, and the isometry-facing completion route from chapter 2036. These sibling rows must be visible before the packet presents* $Q, W, R, D, F, S, H, C, P, N$ *as an effective completion surface.*

Proof. Read the carrier through Definition 5930.1. The rows W, R, S are exactly the finite-window, regular-rational, and terminal real-name entries, while F is the completion-facing route row.

Apply Definition 5930.2. It requires the source, window, readback, tolerance, completion route, and real seal to be exposed before transport, replay, provenance, and naming are consumed.

Thus the L10 siblings are not optional commentary. They are the dependency route through which completion evidence becomes visible, and no quotient completion, selected representative, selected limit, or ambient completeness row is exported outside that route. □

Chapter 5931

A Concrete Naming Certificate for $\mathsf{EdelsteinContraction}^{\uparrow}$

Following Edelstein and Bishop–Bridges constructive metric practice, the $\mathsf{EdelsteinContraction}^{\uparrow}$ packet records the strict compact-metric contraction surface before any fixed-point endpoint is read. It sits after the metric source of chapter 57, the compact-metric finite inspection surface of chapter 129, the contraction comparison discipline of chapter 1418, the displayed Picard orbit route of chapter 2359, and the endpoint consumer of chapter 4298. The packet names only the compact source, a self-map graph, the Picard-style orbit, strict distance-decrease rows, sequential-continuity transport, componentwise history transport, continuation replay, provenance, and local naming; it does not select a fixed point, assert a Banach contraction ratio, import quotient completion equality, or use host choice.

Definition 5931.1 (EdelsteinContraction carrier)**.** *An* $\mathsf{EdelsteinContraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, K, F, O, D, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Met}^{\uparrow}$ *row carrying the inspected point and distance classifier;*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row exposing only the finite compact-metric inspection data used by the strict-decrease consumer;*
3. *F is the self-map graph row over the same metric carrier;*
4. *O is the* $\mathsf{PicardIteration}^{\uparrow}$*-style orbit row generated from the self-map graph and an initial displayed point;*
5. *D is the strict distance-decrease row, recording that every displayed non-diagonal comparison in the orbit is sent to a strictly smaller displayed distance;*
6. *S is the* $\mathsf{SequentialContinuity}^{\uparrow}$ *transport row carrying convergent subsequence and image-cluster reads for the same graph;*
7. *H records componentwise* hsame *transport for the metric, compact-metric, graph, orbit, strict-decrease, sequential-continuity, replay, provenance, and local naming rows;*
8. *C records displayed* Cont *replay for the finite strict-contraction route;*
9. *P is the* Pkg *provenance over* $\mathsf{EdelsteinContraction}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{EdelsteinFixedPoint}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{EdelsteinContraction}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected fixed point, a uniform Lipschitz constant, a quotient completion representative, host function extensionality, open-cover compactness, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.EdelsteinContractionUp.EdelsteinContractionTasteGate single carrier alignment`

Theorem 5931.2 (EdelsteinContraction NameCert obligations)**.** *For every accepted* $\mathsf{EdelsteinContraction}^{\uparrow}$ *carrier $E = (X, K, F, O, D, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{EdelsteinContraction}^{\uparrow}}$ *surface consists exactly of metric carrier admission through X, compact-metric finite inspection through K, self-map graph exposure through F, Picard-style orbit exposure through O, strict distance-decrease exposure through D, sequential-continuity transport through S, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5931.1. The rows $X, K, F, O, D, S, H, C, P, N$ are the only rows available to the local certificate, so each admissible read must begin with the displayed metric, compact-metric, graph, orbit, strict-decrease, sequential-continuity, transport, replay, provenance, and naming coordinates.

The compactness service is read only through K, and the distance service is read only through X. The self-map and orbit service is therefore constrained to the same source carrier before the strict decrease row D can be consumed.

Finally repack the rows through H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve exactly the listed rows and add no endpoint coordinate or host contraction principle. □

Theorem 5931.3 (EdelsteinContraction compact-metric route)**.** *Every fixed-point-facing consumer of an accepted* $\mathsf{EdelsteinContraction}^{\uparrow}$ *packet must read the route*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContractionMapping}^{\uparrow}, \mathsf{PicardIteration}^{\uparrow}, \mathsf{EdelsteinContraction}^{\uparrow}$$

through $X, K, F, O, D, S, H, C, P, N$ before the $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ *endpoint packet may be cited. In particular, the strict compact-metric contraction surface supplies a standalone dependency route and cannot be replaced by a Banach ratio row, a selected endpoint, quotient completion equality, host choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $E = (X, K, F, O, D, S, H, C, P, N)$. The metric and compact-metric rows X and K supply the only distance and compactness reads, so any endpoint-facing consumer first exposes the same source carrier and finite inspection surface.

Read the self-map graph F and the Picard-style orbit O next. The strict distance-decrease row D is meaningful only for those displayed graph and orbit rows, while S transports the sequential image and cluster reads along the same finite route.

The endpoint packet $\mathsf{EdelsteinFixedPoint}^{\uparrow}$ can therefore cite the contraction surface only through H, C, P, N after the metric, compact-metric, graph, orbit, strict-decrease, and sequential-continuity rows have been displayed. Since no Banach ratio, selected endpoint, quotient completion representative, host choice operator, or ambient real-completeness row appears in the carrier, none is exported by the route. □

Falsifiable prediction: Every $\mathsf{EdelsteinContraction}^{\uparrow}$ read factors through one MetricUp source row, one CompactMetricUp finite inspection row, one self-map graph row, one PicardIterationUp-style orbit row, one strict distance-decrease row, one SequentialContinuityUp transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before any EdelsteinFixedPointUp endpoint consumer is available.

Independence witness: metric, compactmetric, contractionmapping, picard-iteration, edelstein-fixed-point: $\mathsf{EdelsteinContraction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a compact-metric source, a self-map graph, an orbit row, a strict pointwise distance-decrease row, sequential-continuity transport, and endpoint-facing provenance without carrying an endpoint itself.

Closure status for $\mathsf{NameCert}_{\mathsf{EdelsteinContraction}^{\uparrow}}$.

Constructive story: An $\mathsf{EdelsteinContraction}^{\uparrow}$ packet is generated from a MetricUp source row, a CompactMetricUp finite inspection row, a self-map graph, a PicardIterationUp-style orbit row, a strict distance-decrease row, a SequentialContinuityUp transport row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EdelsteinContraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5931.1, Definition 5931.2, and Definition 5931.3 seed the strict compact-metric contraction surface feeding the Edelstein fixed-point endpoint route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EdelsteinContraction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close endpoint existence, endpoint uniqueness, Banach contraction ratios, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-metric inspection, self-map graph exposure, orbit exposure, strict distance-decrease preservation, sequential-continuity transport, replay, provenance, and local naming for BEDC.Derived.EdelsteinContractionUp.

Chapter 5932

A Concrete Naming Certificate for $\mathsf{BerkovichClosedDisc}^{\uparrow}$

Following Berkovich's closed-disc analytic viewpoint, the $\mathsf{BerkovichClosedDisc}^{\uparrow}$ packet records the finite BEDC surface of a closed non-Archimedean analytic disc as local-field source data, an ultrametric radius ledger, polynomial probes, and bounded multiplicative seminorm reads. It sits downstream of the p-adic carrier of chapter 100, the local-field source of chapter 173, the ultrametric space boundary of chapter 1535, and the polynomial probe family of chapter 50. The packet names the closed-disc seminorm interface; it does not claim full rigid analytic geometry, a completed adic space theory, a generic-point topology, or a global affinoid-category theorem.

Definition 5932.1 (Berkovich closed disc carrier)**.** *A* $\mathsf{BerkovichClosedDisc}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, a, \rho, U, Q, E, V, H, C, P, N)$$

with the following displayed rows:

1. *K is the local-field source row, exposing only the finite coefficient, valuation, residue, and completeness reads needed by the disc packet.*
2. *a is the centre row, read through the same local-field source and not as a selected host point outside the displayed packet.*
3. *ρ is the ultrametric radius ledger, recording the closed bound $|x - a| \leq \rho$ and the finite tolerance comparisons used by downstream consumers.*
4. *U is the* $\mathsf{UltrametricSpace}^{\uparrow}$ *route that keeps radius comparison non-Archimedean and componentwise.*
5. *Q is the polynomial probe-family row, listing the finite polynomial names and coefficient windows inspected by the current analytic read.*
6. *E is the bounded multiplicative-seminorm read row, recording the values assigned to the displayed probes and the bound imposed by the closed radius ledger.*
7. *V is the valuation compatibility row, requiring products, sums, and scalar coefficients to be consumed only through the displayed local-field and polynomial rows.*
8. *H records componentwise* hsame *transport for K, a, ρ, U, Q, E, V and for the replay, provenance, and local naming rows.*

9. *C records the displayed* Cont *route from local-field inspection through ultrametric radius comparison, polynomial probe readback, bounded seminorm evaluation, and valuation compatibility.*
10. *P is the* Pkg *provenance over* $\mathsf{BerkovichClosedDisc}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, $\mathsf{UltrametricSpace}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BerkovichClosedDisc}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, a, \rho, U, Q, E, V, H, C, P, N$. It has no coordinate for a full Berkovich spectrum, a rigid analytic atlas, an adic point space, a completed structure sheaf, a quotient of analytic functions, or a host compactness theorem.

Lean4 ✓ `BEDC.Derived.BerkovichClosedDiscUp.BerkovichClosedDiscTasteGate single carrier alignment`

Theorem 5932.2 (Berkovich closed disc NameCert obligations)**.** *For every accepted* $\mathsf{BerkovichClosedDisc}^{\uparrow}$ *carrier $D = (K, a, \rho, U, Q, E, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BerkovichClosedDisc}^{\uparrow}}$ *surface consists exactly of carrier admission for the local-field source row, centre row, ultrametric radius ledger, ultrametric comparison route, polynomial probe family, bounded multiplicative-seminorm reads, valuation compatibility, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every closed-disc analytic read must expose $K, a, \rho, U, Q, E, V, H, C, P, N$ before it can be consumed.*

Proof. Project the carrier of Definition 5932.1. The source of coefficients and the centre of the disc are visible only through K and a, while the closed-radius constraint is recorded by the ultrametric ledger ρ and route U.

The analytic read then reaches only the displayed probe surface. The polynomial row Q supplies the finite family being inspected, and the bounded multiplicative-seminorm row E records values for those probes under the bound enforced by ρ.

Valuation compatibility is supplied by V, so products, sums, and scalar coefficients remain tied to the local-field and polynomial rows. The structural rows H, C, P, N transport, replay, source, and name the same finite packet.

Since no further coordinate occurs in the carrier, the certificate cannot export a full spectrum, rigid atlas, adic point space, structure sheaf, analytic-function quotient, or host compactness theorem. □

Falsifiable prediction: Every accepted $\mathsf{BerkovichClosedDisc}^{\uparrow}$ analytic read exposes the local-field source, centre, ultrametric radius ledger, polynomial probe family, bounded multiplicative seminorm values, valuation compatibility, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before any closed-disc consumer may use it.

Independence witness: padic, localfield, ultrametricspace, polynomial: $\mathsf{BerkovichClosedDisc}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a local-field source, a closed ultrametric radius ledger, polynomial probe windows, and bounded multiplicative-seminorm reads in one finite analytic-disc packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BerkovichClosedDisc}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BerkovichClosedDisc}^{\uparrow}$ packet is generated from local-field source data, a centre row, an ultrametric closed-radius ledger, polynomial probe windows, bounded multiplicative-seminorm reads, valuation compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BerkovichClosedDisc}^{\uparrow}}$, seed)

Scope closed: Definition 5932.1 and Definition 5932.2 seed the finite closed-disc seminorm route over local-field, ultrametric, and polynomial probe rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BerkovichClosedDisc}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full rigid analytic geometry, completed adic space theory, a Berkovich spectrum topology, structure sheaf exactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ledger exactness, polynomial-probe readback, bounded multiplicative-seminorm compatibility, valuation stability, transport, replay, provenance, and local naming for BEDC.Derived.BerkovichClosedDiscUp.

The BoundedRealVariation$^{\uparrow}$ hub reads its partition-variation certificate through chapter 38, chapter 301, chapter 1863, chapter 291, and chapter 1757.

Chapter 5933

Bounded Real Variation NameCert Construction

Following Jordan's finite-variation analysis and the Bishop–Bridges constructive treatment of real functions by explicit rational names, the $\mathsf{BoundedRealVariation}^{\uparrow}$ packet records bounded real variation as finite BEDC ledger data. It sits between the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic partition surface of chapter 1863, the dyadic arithmetic ledger of chapter 291, and the bounded-set control of chapter 1757. The packet is a finite partition-variation certificate; it is not a host function quotient, an equality principle for completed reals, an open-cover compactness theorem, a variation measure, a supremum over all partitions, or a selected limiting partition.

Definition 5933.1 (Bounded real variation carrier)**.** *A* $\mathsf{BoundedRealVariation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (I, \Pi, E, R, D, B, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-source row naming the displayed domain endpoints only through terminal real seals.*
2. *Π is the finite partition row, recording the ordered endpoint windows and adjacency data inspected by the variation request.*
3. *E is the endpoint-read row. For every cell boundary in Π, it records the finite source coordinate by which the real-valued input is read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the endpoint values displayed by E.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *variation-bound ledger. It stores the rational absolute-difference rows, finite partial-sum rows, and the displayed dyadic upper bound for the partition variation.*
6. *B is the* $\mathsf{BoundedSet}^{\uparrow}$ *control row asserting that the endpoint reads used by Π stay inside the displayed bounded real window consumed by the ledger D.*
7. *H records componentwise* hsame *transport for the interval source, partition rows, endpoint reads, regular rational readbacks, dyadic variation ledger, bounded-window row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from partition exposure through endpoint readback, dyadic variation accounting, bounded-window control, and the terminal real-name route.*

9. *P is the* Pkg *provenance over* $\mathsf{BoundedRealVariation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, \Pi, E, R, D, B, H, C, P, N$. It has no coordinate for host real equality, arbitrary open covers, countable choice, a completed function-space quotient, a variation measure, or a supremum over unlisted partitions.

Lean4 ✓ `BEDC.Derived.BoundedRealVariationUp.BoundedRealVariationTasteGate single carrier alignment`

Theorem 5933.2 (Bounded real variation NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRealVariation}^{\uparrow}$ *carrier*

$$V = (I, \Pi, E, R, D, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed real interval source I, finite partition exposure through Π, endpoint reads through E,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R, dyadic variation-bound accounting through D, bounded-window control through B, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5933.1. Carrier admission is the assertion that the packet exposes only the rows $I, \Pi, E, R, D, B, H, C, P, N$.

The analytic rows are finite. The partition row Π lists the cells to be inspected, the endpoint-read row E names the finite coordinates used at each cell boundary, and R converts those reads into regular rational data before the dyadic ledger can sum any displayed difference.

The variation-bound obligation is exactly D, with bounded-window control supplied by B. Transport, replay, provenance, and local naming are the structural rows H, C, P, N over that same finite ledger.

Since no carrier coordinate stores host real equality, open-cover compactness, a selected limiting partition, a function-space quotient, a variation measure, or a supremum over all partitions, the listed rows exhaust the NameCert obligation surface. □

Theorem 5933.3 (Bounded real variation RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BoundedRealVariation}^{\uparrow}$ *carrier reads endpoint values only through the finite route*

$\mathsf{DyadicPartition}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BoundedSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, Na

In particular, the consumer must expose the partition row Π, endpoint-read row E, regular rational readback row R, variation-bound ledger D, and bounded-window row B before the terminal real-name row I can be used as a bounded-variation certificate.

Proof. Start with a consumer read of an accepted packet. By Definition 5933.1, the only displayed partition data are the ordered cells in Π, so every endpoint request is tied to that finite row.

Endpoint values are then read through E and R. The row E selects the finite source coordinates, while R provides the $\mathsf{RegSeqRat}^{\uparrow}$ readback visible to arithmetic consumers.

The variation certificate is obtained only after the dyadic ledger D and bounded-window row B have been displayed. Thus a Real or Completion consumer receives a finite partition-variation route rather than host real equality, open-cover compactness, a selected limiting partition, a function-space quotient, or an ambient completeness principle. □

Theorem 5933.4 (Bounded real variation compactness refusal)**.** *An accepted* $\mathsf{BoundedRealVariation}^{\uparrow}$ *packet exports no open-cover compactness, no host real equality, and no supremum principle over all finite partitions. Its public boundary is only the displayed partition ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *endpoint readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *variation-bound ledger, bounded-window control, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$ *row.*

Proof. Inspect the carrier coordinates in Definition 5933.1. None of $I, \Pi, E, R, D, B, H, C, P, N$ is an open cover, a cover-refinement choice function, a host-real equality witness, a quotient of real-valued functions, or a supremum construction.

The rows that can be consumed downstream are therefore the finite rows already named by the carrier: partition exposure, endpoint readback, dyadic variation accounting, bounded-window control, transport, replay, provenance, and naming. Any compactness or equality principle would require an additional coordinate, and the classifier ignores all such data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$, seed)

Scope closed: Definition 5933.1, Definition 5933.2, Definition 5933.3, and Definition 5933.4 seed the finite bounded-real-variation route over partition rows, RegSeqRat endpoint reads, DyadicRatCore variation ledgers, bounded-window control, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BoundedRealVariation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real equality, open-cover compactness, a completed function-space quotient, a variation measure, a supremum over all finite partitions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, RegSeqRat endpoint handoff, dyadic variation-bound exactness, bounded-window control, refusal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BoundedRealVariationUp.

Constructive story: A $\mathsf{BoundedRealVariation}^{\uparrow}$ packet is generated from a RealUp interval-source row, finite DyadicPartition endpoint rows, RegSeqRat readbacks, a DyadicRatCore variation-bound ledger, BoundedSet control, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5934

A Concrete Naming Certificate for $\mathsf{ProfiniteCompletion}^{\uparrow}$

Following Serre and Bourbaki-style profinite completion, the $\mathsf{ProfiniteCompletion}^{\uparrow}$ packet records a residually finite completion surface as finite inverse-limit quotient data rather than as a host topological group. It consumes the projection-window discipline of chapter 2168, the binary compact readback of chapter 1475, the finite observation windows of chapter 126, and the compact metric witness surface of chapter 129. The packet names only finite quotient stages, finite projection checks, compact totally disconnected readbacks, transport, replay, provenance, and local naming; it refuses quotient completion, Haar measure, selected infinite points, host equality, and choice.

Definition 5934.1 (Profinite completion carrier)**.** *A* $\mathsf{ProfiniteCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (Q, \Pi, W, B, K, H, C, P, N)$$

with the following displayed rows:

1. *Q is the quotient-stage ledger. Each row names a visible finite quotient stage, its finite kernel witness, and the finite comparison data admitted at that stage.*
2. *Π is the projection-compatibility ledger. It records only finite bonding maps between named quotient stages and the checked compatibility rows consumed by the inverse-limit surface.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row by which consumers ask for finitely many quotient stages and finitely many projection comparisons.*
4. *B is the* $\mathsf{CantorSpace}^{\uparrow}$ *readback row for the compact totally disconnected binary presentation obtained from the displayed finite windows.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *readback row, sealing the compact metric surface reached through the same finite quotient and projection data.*
6. *H records componentwise* hsame *transports for quotient stages, projection compatibility rows, stream windows, Cantor-space readbacks, compact metric readbacks, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route*

$$Q \longmapsto \Pi \longmapsto W \longmapsto B \longmapsto K.$$

8. *P is the* Pkg *provenance over* ProfiniteCompletion$^{\uparrow}$, InverseLimitMetric$^{\uparrow}$, CantorSpace$^{\uparrow}$, StreamName$^{\uparrow}$, CompactMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{ProfiniteCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion, Haar measure, selected infinite points, host equality, or choice.

Theorem 5934.2 (Profinite completion NameCert obligations)**.** *For every accepted* ProfiniteCompletion$^{\uparrow}$ *carrier $F = (Q, \Pi, W, B, K, H, C, P, N)$, the local* NameCert$_{\text{ProfiniteCompletion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed quotient-stage ledger, finite projection-compatibility ledger,* StreamName$^{\uparrow}$ *finite windows,* CantorSpace$^{\uparrow}$ *compact totally disconnected readback,* CompactMetric$^{\uparrow}$ *readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every public read must pass through those rows in the order displayed by C.*

Proof. Project the accepted carrier in Definition 5934.1. The only mathematical rows available to a consumer are the finite quotient stages Q, the projection-compatibility ledger Π, the finite window schedule W, the Cantor-space readback B, and the compact metric readback K.

The replay row C fixes the inspection route. A public read first selects finitely many quotient stages, then verifies finite projection compatibility, then reads the StreamName window, and only then reaches the CantorSpace and CompactMetric readback rows.

The rows H, P, N transport, source, and name exactly that surface. Since the carrier has no coordinate for quotient completion, Haar measure, selected infinite points, host equality, or choice, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.ProfiniteCompletionUp.ProfiniteCompletionTasteGate single carrier alignment`

Theorem 5934.3 (Profinite completion inverse-limit route)**.** *Every inverse-limit-facing read exported by an accepted* ProfiniteCompletion$^{\uparrow}$ *carrier factors through the displayed route*

$$Q, \Pi, W, B, K, H, C, P, N.$$

The quotient-stage ledger Q supplies finite stage data, Π supplies finite bonding compatibility, W supplies the StreamName$^{\uparrow}$ *window discipline, B supplies the* CantorSpace$^{\uparrow}$ *compact totally disconnected readback, and K supplies the* CompactMetric$^{\uparrow}$ *readback. The rows H, C, P, N provide transport, replay, provenance, and local naming for the same route.*

Proof. Start with the carrier of Definition 5934.1. Its inverse-limit content is not a host object; it is the displayed finite route from quotient stages and projection checks to finite windows and compact readbacks.

Apply the NameCert surface of Definition 5934.2. The consumer must inspect Q and Π before reading W, and W is the only access path to the CantorSpace row B and the CompactMetric row K.

Transport by H and replay by C preserve the same finite route, while P and N record provenance and local naming. Therefore every inverse-limit-facing read factors through $Q, \Pi, W, B, K, H, C, P, N$ and exports no quotient completion, Haar measure, selected infinite point, host equality, or choice principle. □

Closure status for NameCert$_{\text{ProfiniteCompletion}^{\uparrow}}$.

Theory closure: TheoryGate(NameCert$_{\text{ProfiniteCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 5934.1, Definition 5934.2, and Definition 5934.3 seed the finite profinite-completion packet over quotient stages, projection compatibility, StreamName windows, CantorSpace readback, CompactMetric readback, transport, replay, provenance, and naming.

Formal verification: FormalStatus(NameCert$_{\mathsf{ProfiniteCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion, Haar measure, selected infinite points, host equality, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite quotient-stage routing, projection compatibility, StreamName window routing, CantorSpace readback, CompactMetric readback, transport stability, provenance, and local naming for BEDC.Derived.ProfiniteCompletionUp.

Constructive story: A ProfiniteCompletion$^{\uparrow}$ packet is generated from finite quotient-stage rows, projection-compatibility rows, StreamName finite-window rows, CantorSpace compact totally disconnected readback rows, CompactMetric readback rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 5935

A Concrete Naming Certificate for $\mathsf{SequentialCompactReal}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{SequentialCompactReal}^{\uparrow}$ packet records the real-sequence compactness route as finite BEDC data. It sits beside the finite-window compactness surface of chapter 2372, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a non-quotient route from displayed sequence windows to a compact interval cluster read; it is not countable choice, not open-cover compactness, and not a selected infinite subsequence supplied by the host.

Definition 5935.1 (SequentialCompactReal carrier)**.** *A* $\mathsf{SequentialCompactReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, W, Q, I, S, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the real-sequence source row, naming the sequence request and the finite indices inspected by the compactness consumer.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite source windows, requested index windows, and observation schedule.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the rational approximants and tail budgets visible inside W.*
4. *I is the compact interval and finite-net row. It records the displayed interval bound, finite net scale, and interval membership witnesses used by the compactness route.*
5. *S is the subsequence-window row, recording only the displayed selected finite windows, their monotone index ledger, and the cluster-window comparison cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, compact interval row, and subsequence-window row have been exposed.*
7. *H records componentwise* hsame *transport for A, W, Q, I, S, E.*
8. *C records displayed* Cont *replay from the sequence source through stream windows, regular rational readback, compact interval nets, subsequence windows, and the real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{SequentialCompactReal}^{\uparrow}$, $\mathsf{SequentialCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentialCompactReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, W, Q, I, S, E, H, C, P, N$. It has no coordinate for quotient stream equality, a host-selected infinite subsequence, countable choice, open-cover compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.SequentialCompactRealUp.SequentialCompactRealTasteGate single carrier alignment`

Theorem 5935.2 (SequentialCompactReal NameCert obligations)**.** *For every accepted* $\mathsf{SequentialCompactReal}^{\uparrow}$ *carrier*

$$R = (A, W, Q, I, S, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentialCompactReal}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *real-sequence source admission through A and the finite* $\mathsf{StreamName}^{\uparrow}$ *windows in W;*
2. $\mathsf{RegSeqRat}^{\uparrow}$ *readback through Q, with rational observations restricted to the displayed windows and tail budgets;*
3. *compact interval and finite-net admission through I, before any cluster read can be public;*
4. *subsequence-window and cluster comparison through S, followed by the terminal* $\mathsf{Real}^{\uparrow}$ *seal E;*
5. *componentwise transport, displayed replay, provenance, and local naming through H, C, P, N.*

No obligation row may be discharged by quotient stream equality, countable choice, a host-selected infinite subsequence, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5935.1. The packet exposes exactly the real-sequence source, stream windows, regular rational readback, compact interval net, subsequence window, real seal, transport, replay, provenance, and local naming rows.

The source and rational data are finite. The rows A, W, Q name only requested sequence windows and their rational readbacks, so a consumer cannot replace them by a quotient stream or a full sequence object.

The compactness data enter through I and S. The interval and finite-net row is displayed before any cluster comparison, and the subsequence row records only finite selected windows with a monotone index ledger.

Finally E, H, C, P, N seal, transport, replay, source, and name that same packet. Since the excluded host principles are absent from the carrier, the five displayed obligation rows exhaust the local NameCert surface. □

Theorem 5935.3 (SequentialCompactReal regular rational route)**.** *Every public subsequence, cluster, or compact-interval read of an accepted* $\mathsf{SequentialCompactReal}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{SequentialCompact}^{\uparrow}$$

before it reaches a downstream real-completion consumer. More concretely, the read first opens the finite windows W, then reads the regular rational row Q, then reaches the real-name seal E, and only then may cite the sequential compactness surface through I, S, H, C, P, N. The route exports no quotient stream equality, host-selected infinite subsequence, countable choice, open-cover compactness theorem, or ambient real completeness principle.

Proof. Start from Definition 5935.1. A public read can inspect only $A, W, Q, I, S, E, H, C, P, N$.

Apply Definition 5935.2. The source row opens finite $\mathsf{StreamName}^{\uparrow}$ windows before the $\mathsf{RegSeqRat}^{\uparrow}$ readback row can be consumed, and the real seal remains downstream of those rational observations.

The compactness part is then read through the sibling route of chapter 2372. The compact interval row I and subsequence-window row S play the same finite-window role for real sequences that the $\mathsf{SequentialCompact}^{\uparrow}$ packet plays for compactness consumers.

The structural rows H, C, P, N preserve and name this route. Since no excluded host coordinate occurs among the displayed rows, every public subsequence or cluster read factors through the stated sequence of finite interfaces. □

Closure status for $\mathsf{NameCert}_{\mathsf{SequentialCompactReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentialCompactReal}^{\uparrow}$ packet is generated from a real-sequence source row, finite StreamName windows, RegSeqRat rational readbacks, compact interval finite-net rows, subsequence-window ledgers, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SequentialCompactReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5935.1, Definition 5935.2, and Definition 5935.3 seed the finite real-sequence compactness route through StreamName, RegSeqRat, Real, and SequentialCompact rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SequentialCompactReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, countable choice, host-selected infinite subsequences, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stream-window exactness, RegSeqRat readback, compact interval finite-net admission, subsequence-window monotonicity, Real seal nonescape, and selector-refusal proofs for BEDC.Derived.SequentialCompactRealUp.

Chapter 5936

A Concrete Naming Certificate for MetaCICNormalizationFuelBoundary$^{\uparrow}$

The MetaCICNormalizationFuelBoundary$^{\uparrow}$ packet realizes the candidate-mediated bounded-fuel reading of chapter 9383 as finite NameCert data. It sits beside the normalization-candidate socket of chapter 3566, the candidate router of chapter 5210, the critical-path packet of chapter 765, and the normalization frontier of chapter 892. The packet records bounded inspection of candidate evidence; it does not promote a fuel trace into global strong normalization.

Definition 5936.1 (MetaCIC normalization fuel-boundary carrier)**.** *A* MetaCICNormalizationFuelBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, K, B, L, R, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed-term row, naming the typed closed MetaCIC term and the finite observations requested by the audit.*
2. *K is the candidate row, carrying candidate, closed-candidate, or candidate-set evidence before any strong-normalization read is allowed.*
3. *B is the bounded-fuel row, recording the natural fuel budget, decrement ledger, and stopping boundary for the inspected trace.*
4. *L is the finite trace row, listing the inspected beta steps and the local normal-endpoint status reached within the budget.*
5. *R is the closedness-preservation row, requiring every inspected trace endpoint to keep the displayed closed-term boundary before it is replayed.*
6. *O is the obstruction row, recording timeout, missing candidate evidence, subject-reduction dependence, and evidence-erasure refusal as visible outcomes.*
7. *H records componentwise* hsame *transport for T, K, B, L, R, O.*
8. *C records displayed* Cont *replay from the closed term through candidate evidence, fuel, trace, closedness preservation, obstruction, and structural rows.*
9. *P is the* Pkg *provenance over* MetaCICNormalizationFuelBoundary$^{\uparrow}$, NormalizationCandidateSocket$^{\uparrow}$, MetaCICCandidateNormalizationRouter$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, MetaCICNormalizationFrontier$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, K, B, L, R, O, H, C, P, N$. It has no coordinate for evidence-erased strong normalization, erased subject reduction, host conversion equality, quotient syntax equality, or closed consistency.

Theorem 5936.2 (MetaCIC normalization fuel-boundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$ *carrier*

$$F = (T, K, B, L, R, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}}$ *surface consists exactly of finite rows for closed-term admission, candidate-evidence admission, bounded-fuel ledger exposure, trace inspection, closedness-preservation replay, obstruction visibility, componentwise transport, displayed continuation replay, provenance, and local naming. No obligation row may erase K, widen B into an unbounded normalizer, discharge the subject-reduction boundary, or export closed consistency.*

Proof. Project the carrier of Definition 5936.1. The displayed packet exposes a closed term, candidate evidence, bounded fuel, a finite trace, closedness preservation, obstruction, transport, replay, provenance, and local naming.

The candidate row is consumed before any normalization read. A public consumer may inspect K as evidence for the candidate-mediated route, but it cannot delete K and retain the same conclusion.

The fuel and trace rows are bounded. The row B supplies a finite budget and decrement ledger, while L records only the inspected steps within that budget.

The rows R and O keep the boundary visible. Closedness preservation is replayed only for the inspected trace endpoints, and missing candidate evidence, timeout, or subject-reduction dependence remain obstruction outcomes rather than hidden assumptions.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no evidence-erased normalizer or closed-consistency row, the listed obligations exhaust the local NameCert surface. □

Theorem 5936.3 (MetaCIC normalization fuel-boundary candidate route)**.** *Every public bounded-fuel normalization audit read of an accepted* $\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$ *packet factors through the candidate row K, the fuel row B, the finite trace row L, the closedness-preservation row R, and the visible obstruction row O before any replay through H, C, P, N. The route may report a bounded normal endpoint or a visible obstruction, but it cannot export evidence-erased strong normalization, unbounded normalization, erased subject reduction, host conversion equality, quotient syntax equality, or closed consistency.*

Proof. Begin with Definition 5936.2. It orders the public rows as closed-term admission, candidate evidence, bounded fuel, finite trace, closedness preservation, obstruction, transport, replay, provenance, and local naming.

The candidate-mediated siblings supply the front of the route. The normalization-candidate socket and candidate-normalization router provide the evidence row K and route it into the bounded trace without turning candidate evidence into an erased premise.

The critical-path and frontier siblings supply the boundary. They keep subject-reduction dependence, unfinished global normalization, and closed-consistency use outside the exported packet unless the displayed rows name them.

Thus a consumer can read only a bounded trace under candidate evidence, together with its visible obstruction status. The packet has no coordinate for the excluded host conclusions, so the candidate route is the whole public audit surface. □

Falsifiable prediction: A bounded normalization audit read that exports strong normalization after deleting the candidate row K and the fuel ledger B falsifies $\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$; the rejected evidence-erased fuel-to-SN route is not an accepted packet read.

Independence witness: Nearest siblings: normalization-candidate-socket, metacic-candidate-normalization-router, metaciccriticalpath, and metacicnormalizationfrontier. The fuel-boundary packet is not bijective to any of them because it jointly carries closed-term data, candidate evidence, bounded fuel, a finite trace, closedness preservation, visible obstruction, transport, replay, provenance, and local naming as one carrier.

The Lean TasteGate follow-up for this packet requires explicit $\mathsf{Nontrivial}\ \mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$ instances in `TasteGate.lean`. The nontrivial boundary must separate accepted candidate-mediated bounded traces from timeout or missing-candidate packets, and field faithfulness must preserve $T, K, B, L, R, O, H, C, P, N$ without erasing the candidate or fuel rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$ packet is generated from a closed MetaCIC term row, candidate evidence, bounded fuel, finite trace rows, closedness-preservation replay, visible obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5936.1, Definition 5936.2, and Definition 5936.3 seed the candidate-mediated bounded-fuel normalization audit boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim evidence-erased strong normalization, unbounded normalization, erased subject reduction, host conversion equality, quotient syntax equality, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked closed-term admission, candidate-evidence admission, fuel-ledger exactness, trace inspection, closedness-preservation replay, obstruction visibility, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.MetaCICNormalizationFuelBoundaryUp.

Chapter 5937

A Concrete Naming Certificate for $\mathsf{CompactIntervalModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CompactIntervalModulus}^{\uparrow}$ packet records the finite modulus route for compact intervals as displayed NameCert data. It refines the compact interval source of chapter 3986, consumes the finite compactness surface of chapter 3969, and terminates through the real-name seal of chapter 38. The packet supplies a choice-free L10 route from interval compactness to modulus readback for downstream Completion, LocatedMetric, and separated-reflection consumers without asserting open-cover compactness, selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5937.1 (Compact interval modulus carrier)**.** *A* $\mathsf{CompactIntervalModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, K, F, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *source row, exposing the constructive interval endpoint, locatedness, finite-net, dyadic, stream, regular, and Real-facing data of the compact interval route.*
2. *K is the* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *compactness row over the same interval source.*
3. *F is the finite-net handoff row, recording the inspected mesh, radius ledger, and coverage comparison used by modulus consumers.*
4. *T is the dyadic tolerance row for requested output precision.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the interval and finite-net data.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite windows and tolerance ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint and modulus readback.*
8. *H records componentwise* hsame *transport for source interval, compactness, finite-net, tolerance, stream, regular, Real, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from interval inspection through compact finite-net handoff, tolerance request, finite windows, regular readback, and Real sealing.*
10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalModulus}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, ClosedBoundedIntervalCompact $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulus}^{\uparrow}}$ *row.*

The classifier accepts only packets with these displayed rows. It has no coordinate for arbitrary open covers, selected finite subcovers, quotient real equality, countable-choice schedules, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactIntervalModulusUp.CompactIntervalModulusTasteGate single carrier alignment`

Theorem 5937.2 (Compact interval modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalModulus}^{\uparrow}$ *carrier* $M = (I, K, F, T, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, Bishop compact interval source exposure through* I*, closed-bounded compactness exposure through* K*, finite-net handoff through* F*, dyadic tolerance control through* T*, finite window exposure through* W*, regular rational readback through* R*, terminal Real sealing through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Carrier admission is projection from Definition 5937.1. The tuple lists the interval source, compactness row, finite-net handoff, tolerance row, stream windows, regular readback, Real seal, and structural rows.

The interval source is read first through I and K. These rows bind the Bishop compact interval route to the closed-bounded compactness surface before a modulus consumer can inspect any mesh or tolerance data.

The modulus handoff is finite. The row F records the mesh and coverage comparison, while T, W, R route each precision request through dyadic tolerance, finite stream windows, and regular rational readback.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no open-cover, selected-subcover, quotient, choice, representative, or ambient-completeness coordinate, the listed rows exhaust the local certificate surface. □

Theorem 5937.3 (Compact interval modulus finite-net handoff)**.** *Every modulus-facing consumer of an accepted* $\mathsf{CompactIntervalModulus}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{BishopCompactInterval}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalCompact}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}$$

The handoff exposes only the compact interval source, closed-bounded compactness row, finite-net mesh, dyadic tolerance, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming; it exports no open-cover compactness theorem, selected subcover, quotient real equality, countable-choice schedule, selected representative, or ambient Real completeness principle.

Proof. Project the accepted carrier $M = (I, K, F, T, W, R, E, H, C, P, N)$. The compact interval source is I, the compactness row is K, and the finite-net handoff is F.

Read a modulus request through T, W, R. The dyadic tolerance row fixes the requested precision, the stream row exposes the finite windows permitted by the interval and net data, and the regular row gives the rational readback.

The terminal route reaches E only after those displayed rows have been consumed. The structural rows H, C, P, N preserve the same order for downstream consumers.

Because the carrier lacks open-cover, selected-subcover, quotient, choice, representative, and ambient-completeness coordinates, no stronger compactness or Real principle can be read from the handoff. □

Theorem 5937.4 (Compact interval modulus completion readback)**.** *Any* Completion$^{\uparrow}$ *or* LocatedMetric$^{\uparrow}$ *consumer reading a modulus from an accepted* CompactIntervalModulus$^{\uparrow}$ *packet receives the displayed rows* $I, K, F, T, W, R, E, H, C, P, N$ *and no additional equality, selection, or completeness data. A* SeparatedMetric$^{\uparrow}$ *consumer may later transport the distance or readback rows through its own reflection surface, but this packet itself does not turn compact interval modulus evidence into host equality, quotient equality, selected representatives, selected subcovers, countable choice, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with the carrier of Definition 5937.1. Its completion-facing data are exactly the compact interval source, compactness row, finite-net handoff, tolerance row, stream row, regular readback, Real seal, and structural rows.

Apply Definition 5937.2. The obligation surface forces source exposure through I, K, handoff through F, tolerance and readback through T, W, R, and terminal sealing through E.

Apply Definition 5937.3. The theorem orders the same rows as the finite route from compact interval data to modulus readback.

Thus Completion and LocatedMetric consumers may consume the displayed distance and readback data only through this route. A later SeparatedMetric consumer must add its own reflection rows before any identity interpretation is available, and all refused open-cover, selection, quotient, choice, representative, and ambient-completeness objects remain absent. □

Falsifiable prediction: Every accepted CompactIntervalModulus$^{\uparrow}$ read passes through BishopCompactIntervalUp, ClosedBoundedIntervalCompactUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, transport, replay, provenance, and local naming rows before any modulus output is visible.

Independence witness: bishop-compact-interval, closed-bounded-interval-compact, compact-interval-modulus-bridge, compact-interval-choice-free, uniformmodulus: CompactIntervalModulus$^{\uparrow}$ is not bijective to any listed sibling because it jointly binds compact interval source rows, closed-bounded compactness rows, finite-net handoff, dyadic tolerance, stream windows, regular readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{CompactIntervalModulus}^{\uparrow}}$.

Constructive story: A CompactIntervalModulus$^{\uparrow}$ packet is generated from BishopCompactIntervalUp, ClosedBoundedIntervalCompactUp, finite-net handoff, DyadicRatCoreUp tolerance, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactIntervalModulus}^{\uparrow}}$, seed)

Scope closed: Definition 5937.1, Definition 5937.2, Definition 5937.3, and Definition 5937.4 seed the finite compact-interval modulus route over compact interval, closed-bounded compactness, dyadic, stream, regular, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactIntervalModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact interval source exposure, finite-net handoff, tolerance routing, readback, RealUp sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalModu

Chapter 5938

A Concrete Naming Certificate for $\mathsf{MoscoConvergence}^{\uparrow}$

Following Mosco's variational convergence theorem, the $\mathsf{MoscoConvergence}^{\uparrow}$ packet records the finite certificate surface by which lower-semicontinuous convex functionals converge through weak-liminf and strong-recovery clauses. It sits downstream of the weak-topology route of chapter 5192, the convex-support surface of chapter 208, the completion handoff of chapter 1506, and the terminal real readback of chapter 38. Its weak-liminf and recovery rows feed the explicit variational witness ledger of chapter 6745 before that sibling exposes recovery schedules, value readbacks, and completion handoffs. The packet is a NameCert for variational convergence data, not a host Hilbert-space theorem, a selected minimizer, quotient equality of functionals, compactness by choice, or an ambient dual-space completion principle.

Definition 5938.1 (MoscoConvergence carrier)**.** *A* $\mathsf{MoscoConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, W, L, R, C, H, T, P, N)$$

with the following displayed rows:

1. *F is the lower-semicontinuity row for the source functionals, exposing only finite sublevel probes, scalar readbacks, and closedness checks.*
2. *W is the weak-topology row, naming the finite weak neighborhoods and convergence probes used by liminf consumers.*
3. *L is the weak-liminf ledger, recording each lower-bound comparison through F, W and the terminal real readback.*
4. *R is the strong recovery-sequence row, displaying the finite approximation schedule, strong convergence probes, and scalar upper-bound witnesses.*
5. *C is the convex-set support row, binding the admissible-domain and sublevel convexity data used by both L and R.*
6. *H is the completion handoff row, available only after F, W, L, R, C have been displayed and only for the completion route named by this packet.*
7. *T stores componentwise* hsame *transport for lower-semicontinuity, weak topology, liminf, recovery, convex support, completion, provenance, and local naming rows.*

8. *P is the* Pkg *provenance over* $\mathsf{MoscoConvergence}^{\uparrow}$*, weak topology, convex support, completion handoff, lower-semicontinuity, recovery-sequence,* BHist, hsame, Cont, *and* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{MoscoConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, W, L, R, C, H, T, P, N$. It has no coordinate for host epigraph convergence, a selected minimizer, quotient equality of functionals, global compactness, countable choice, or ambient Hilbert-space completeness.

Theorem 5938.2 (MoscoConvergence NameCert obligations)**.** *For every accepted* $\mathsf{MoscoConvergence}^{\uparrow}$ *carrier*

$$M = (F, W, L, R, C, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MoscoConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission for the lower-semicontinuity row F, weak-topology row W, weak-liminf ledger L, recovery-sequence row R, convex support row C, completion handoff H, componentwise transport T, package provenance P, and local naming N.*

Proof. Project the carrier of Definition 5938.1. The convergence data visible to the classifier are precisely the lower-semicontinuity row F, weak-topology row W, weak-liminf ledger L, recovery row R, convex support row C, and completion handoff H.

The liminf obligation is forced through F, W, L. A consumer cannot read a weak lower-bound comparison until the weak neighborhood probes and lower-semicontinuity readbacks have both been displayed.

The recovery obligation is separate and positive. The row R supplies only finite strong-approximation schedules and scalar upper-bound witnesses, while C records the convex support on which those schedules are allowed to live.

Finally read H, T, P, N. These rows route completion handoff, transport, provenance, and local naming for the same finite packet. Since no host convergence theorem or selected minimizer is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 5938.3 (MoscoConvergence liminf-limsup scope)**.** *Every variational-convergence consumer of an accepted* $\mathsf{MoscoConvergence}^{\uparrow}$ *packet must read the weak-liminf route through F, W, L and the recovery route through R, C before it may use the completion handoff H or the structural rows T, P, N. The packet supplies a finite liminf–recovery certificate; it does not supply epi-convergence as a host theorem, Gamma-convergence equivalence, selected minimizers, quotient equality of functionals, compactness by countable choice, or ambient Hilbert-space completeness.*

Proof. Begin with the weak-liminf request. The only available lower-bound route is F, W, L, so the request must expose lower-semicontinuity probes, weak-topology neighborhoods, and the displayed real-valued comparison ledger.

Next read the recovery side. The row R supplies finite strong-approximation data, and C records the convex support needed to keep those approximants inside the admissible packet.

The completion handoff H is downstream of both routes. Transport, provenance, and local naming through T, P, N preserve the same displayed rows but add no global variational-convergence theorem. Thus the packet closes only the finite Mosco liminf–limsup scope named by the carrier. □

The same weak-liminf and recovery rows are consumed by chapter 5957 and **??** through the shared $F, W, L, R, C, H, T, P, N$ carrier before epigraphical or Gamma-facing consumers read completion data, so the local Mosco packet sits on the variational dependency lattice rather than as an isolated witness ledger.

Theorem 5938.4 (MoscoConvergence subspace topology route)**.** *Let $M = (F, W, L, R, C, H, T, P, N)$ be an accepted* $\mathsf{MoscoConvergence}^{\uparrow}$ *carrier. Any Gamma-convergence, epigraphical-limit, or subspace-topology consumer of the Mosco data must first display the lower-semicontinuity row F, weak-topology row W, recovery-sequence row R, convex-support row C, and completion handoff H, together with the weak-liminf ledger L and the structural rows T, P, N. In particular, consumers routed through chapter 5957,* **??***, or chapter 5993 receive the same finite variational packet before reading an induced topology or variational comparison conclusion.*

Proof. Project the carrier of Definition 5938.1. The rows F, W, L, R, C, H are the only analytic data available before transport, provenance, and local naming.

Apply Definition 5938.2. The lower-semicontinuity and weak-topology rows must precede the weak-liminf ledger, while the recovery and convex-support rows must precede any strong recovery read.

Next apply Definition 5938.3. It keeps the completion handoff downstream from both variational sides, so Gamma-facing, epigraphical, and subspace-topology consumers cannot read H or an induced-topology conclusion before F, W, L, R, C are displayed.

The structural rows T, P, N preserve the same finite packet. Since no carrier coordinate stores host epigraph convergence, quotient equality of functionals, selected minimizers, global compactness, or countable choice, the sibling lattice route is exactly the displayed row order. □

Closure status for $\mathsf{NameCert}_{\mathsf{MoscoConvergence}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MoscoConvergence}^{\uparrow}$ packet is generated from lower-semicontinuity rows, weak-topology probes, a weak-liminf ledger, recovery-sequence data, convex support, completion handoff, componentwise hsame transport, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MoscoConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5938.1, Definition 5938.2, and Definition 5938.3 seed the finite Mosco convergence carrier through lower-semicontinuity, weak topology, recovery, convex support, completion handoff, and the recovery-ledger route of chapter 6745.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MoscoConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host epigraph convergence, Gamma-convergence equivalence, selected minimizers, quotient equality of functionals, global compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lower-semicontinuity visibility, weak-liminf exactness, recovery-sequence coverage, convex-support routing, completion handoff, transport, provenance, and local naming for BEDC.Derived.MoscoConvergenceUp.

Theorem 5938.5 (Mosco-to-Gamma variational dependency route)**.** *Every Γ-convergence or epigraphical-limit consumer that uses a* $\mathsf{MoscoConvergence}^{\uparrow}$ *packet may read the Mosco data only through the displayed carrier rows*

$$F, W, L, R, C, H, T, P, N.$$

The weak-liminf request is therefore routed through F, W, L, the recovery request through R, C, and any completion handoff through H after those two variational rows have been displayed. Consequently the sibling routes of chapter 5957 and **??** *consume a shared finite variational lattice rather than a host equivalence between Mosco, epigraphical, and Γ convergence.*

Proof. Start with the accepted Mosco carrier of Definition 5938.1. Its weak-liminf side is exactly F, W, L, so a downstream lower-bound read cannot bypass the lower-semicontinuity row, weak-topology probes, or displayed comparison ledger.

The recovery side is separate. The row R gives the strong approximation schedule and C gives the convex support on which the schedule is read. These rows are the only recovery coordinates available to the variational consumers.

The completion and structural rows are downstream. The handoff H, transport T, provenance P, and local naming row N preserve the same packet for the Gamma and epigraphical routes, but they add no host theorem identifying the convergence notions. $\square$

Chapter 5939

A Concrete Naming Certificate for $\mathsf{FiniteRealIntervalChain}^{\uparrow}$

Following Cantor and Bishop–Bridges constructive nested-interval practice, the $\mathsf{FiniteRealIntervalChain}^{\uparrow}$ packet records a finite real-interval chain before any Real or Completion consumer may read an endpoint route as a point-name witness. It sits beside the dyadic rational core of chapter 291, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the located nested-interval sibling of chapter 1713. The packet names finite endpoint ledgers, containment witnesses, and diameter budgets; it is not an open-cover compactness theorem, a quotient-real construction, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5939.1 (Finite real interval chain carrier)**.** *A* $\mathsf{FiniteRealIntervalChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (E, I, J, D, S, R, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the endpoint ledger, listing the left and right endpoint rows for each retained interval cell and their* $\mathsf{Real}^{\uparrow}$*-facing names.*
2. *I is the closed interval row family, recording the displayed closed interval cells and their endpoint order witnesses.*
3. *J is the chain containment witness row, asserting that every later interval cell is read inside the preceding cell through finite endpoint comparisons.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter-budget row, carrying the requested dyadic tolerance and the displayed diameter bound for each retained cell.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which endpoint and containment observations are requested.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic budget ledgers.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after E, I, J, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for endpoint ledgers, closed interval rows, containment witnesses, dyadic budgets, stream windows, readback, Real seal, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a Real-facing interval witness request through E, I, J, D, S, R, L, H.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteRealIntervalChain}^{\uparrow}$*, chapter 291, chapter 126, chapter 301, chapter 38, chapter 1713,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealIntervalChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host interval intersection, selected quotient representatives, countable choice, open-cover compactness, detached completion data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.FiniteRealIntervalChainUp.FiniteRealIntervalChainUp`

Theorem 5939.2 (FiniteRealIntervalChain NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRealIntervalChain}^{\uparrow}$ *carrier $F = (E, I, J, D, S, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteRealIntervalChain}^{\uparrow}}$ *surface consists exactly of endpoint-ledger admission through E, closed interval row exposure through I, chain-containment witnesses through J, dyadic diameter budgets through D,* $\mathsf{StreamName}^{\uparrow}$ *window access through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5939.1. The endpoint ledger E and closed interval rows I expose the finite interval cells before any containment or point-name request can be replayed.

The chain part is read through J and D. The containment row J keeps later cells inside earlier cells, while the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the diameter budget used by the finite precision request.

The Real-facing part is read only after the finite observation route has been displayed. The $\mathsf{StreamName}^{\uparrow}$ windows S determine the requested endpoint observations, the $\mathsf{RegSeqRat}^{\uparrow}$ row R supplies regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal L is reached after those rows.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row carries host interval intersection, quotient-real equality, countable choice, open-cover compactness, detached completion data, or ambient completeness, the NameCert surface is exhausted by the displayed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealIntervalChain}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRealIntervalChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5939.1 and Definition 5939.2 seed the finite interval-chain route from endpoint ledgers through closed interval cells, containment witnesses, dyadic budgets, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRealIntervalChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, quotient Real construction, selected host intersection point, detached completion object, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-row exposure, containment witness transport, dyadic diameter budget exactness, readback route, Real seal non-escape, and TasteGate compatibility for BEDC.Derived.FiniteRealIntervalChainUp.

Constructive story: A $\mathsf{FiniteRealIntervalChain}^{\uparrow}$ packet is generated from endpoint ledgers, closed interval rows, chain containment witnesses, DyadicRatCore diameter budgets, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

$\mathsf{RegularRealTailEquivalence}^{\uparrow}$ consumes the regular-tail readback of chapter 301 together with the finite-window schedule of chapter 126.

Chapter 5940

A Concrete Naming Certificate for $\mathsf{RegularRealTailEquivalence}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ packet records equality of regular real names as a finite tail-equivalence certificate rather than as a quotient of streams. It sits after the regular rational source rows of chapter 301, the finite observation windows of chapter 126, the dyadic tolerance ledger of chapter 291, the zero-tail factorization of chapter 380, the comparison surface of chapter 411, and the terminal real-name seal of chapter 38. The packet supplies the same-real route used by $\mathsf{Real}^{\uparrow}$ consumers without exporting host equality, selected limits, or ambient completeness.

Definition 5940.1 (Regular real tail equivalence carrier)**.** *A* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R_0, R_1, W, D, Z, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *and* R_1 *are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular real names are compared.*
2. W *is the shared* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which both tails are inspected.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the displayed late-stage difference bounds.*
4. Z *is the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row for the displayed difference between the two source rows.*
5. Q *is the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *comparison row that ties the two source tails to the same finite comparison route.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal boundary available only after* R_0, R_1, W, D, Z, Q *have been displayed.*
7. H *records componentwise* hsame *transport for the source rows, window row, dyadic ledger, null row, comparison row, seal row, continuation, provenance, and local name.*
8. C *records displayed* Cont *replay of the same finite route.*
9. P *records the* Pkg *provenance over* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NullSequence}^{\uparrow}$, $\mathsf{RegularCauchyComparison}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularRealTailEquivalence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R_0, R_1, W, D, Z, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, a selected representative, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5940.2 (Regular real tail equivalence NameCert obligations)**.** *For every accepted* RegularRealTailEquivale *carrier*

$$T = (R_0, R_1, W, D, Z, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularRealTailEquivalence}^{\uparrow}}$ *surface consists exactly of source-row exposure for R_0, R_1, shared finite-window exposure for W, dyadic tolerance exposure for D, zero-tail exposure for Z, comparison-route exposure for Q,* $\mathsf{Real}^{\uparrow}$ *seal exposure for E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5940.1. The first two rows expose the two regular rational sources, while W supplies the common finite stream window through which the tails are inspected.

Next read the analytic rows. The dyadic ledger D records the late-stage tolerance bounds, Z records the zero-tail route for the displayed difference, and Q records the regular Cauchy comparison route connecting the same source rows to the same finite window.

Finally read E, H, C, P, N. The seal row is downstream of the source, window, dyadic, null, and comparison rows, and the administrative rows transport, replay, package, and name that same finite route. Hence the NameCert surface is exactly the displayed tuple. □

Theorem 5940.3 (Regular real tail equivalence RegSeqRat route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer read of an accepted* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *carrier factors through*

$$(R_0, R_1, W, D, Z, Q).$$

The read may use the two regular rational source rows only through the shared finite $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail row, and the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *route. It cannot introduce a second stream representative, quotient stream equality, host real equality, selected limit, or ambient completeness datum.*

Proof. Start with an accepted packet. Projection exposes R_0 and R_1 as the only source rows, so any regular-rational read must begin with those displayed coordinates.

The shared finite window and dyadic ledger are then forced by W and D. A consumer has no detached inspection window or tolerance row outside the packet.

The same-real handoff is supplied by Z and Q. The null row records the zero-tail route for the displayed difference, and the comparison row records the common regular Cauchy comparison route. Since the carrier contains no coordinate for quotient streams, host equality, selected limits, or ambient completeness, none can be exported by the consumer read. □

Theorem 5940.4 (Zero-tail symmetry route)**.** *Let*

$$T = (R_0, R_1, W, D, Z, Q, E, H, C, P, N)$$

be an accepted $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *carrier. Swapping the two source rows gives the same Real-facing tail-equivalence route through*

$$(R_1, R_0, W, D, Z, Q, E, H, C, P, N).$$

The only equality witnesses consumed by the $\mathsf{Real}^{\uparrow}$ *seal are the zero-tail row* Z *and the regular Cauchy comparison row* Q*, and both are read through the shared finite window and dyadic ledger* W, D*. Thus the certificate is symmetric in* R_0 *and* R_1 *without adding quotient stream equality, selected representatives, host real equality, or ambient completeness.*

Proof. Project the carrier of Definition 5940.1. The source rows R_0 and R_1 are the only ordered coordinates affected by the swap; the shared window, dyadic ledger, null row, comparison row, seal, transport, replay, provenance, and local name remain the displayed rows of the same packet.

Use the NameCert obligations of Definition 5940.2. They require every same-real read to expose W, D, Z, Q before E is available, so the seal consumes the zero-tail and regular Cauchy comparison witnesses rather than the presentation order of the two source rows.

Apply the RegSeqRat route of Definition 5940.3. It confines both source rows to the shared finite-window and tolerance path. After the two sources are swapped, the rows H, C, P, N transport, replay, package, and name the same finite route, so no quotient stream, selected representative, host real equality, or ambient completeness datum is introduced. □

Theorem 5940.5 (Regular real tail equivalence transitivity route)**.** *Let*

$$T_{01} = (R_0, R_1, W_{01}, D_{01}, Z_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$T_{12} = (R_1, R_2, W_{12}, D_{12}, Z_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *carriers whose middle* $\mathsf{RegSeqRat}^{\uparrow}$ *row is displayed as the same finite source row and whose* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers are jointly visible to one consumer. Then the two zero-tail and comparison routes compose to a third* $\mathsf{Real}^{\uparrow}$*-seal route from* R_0 *to* R_2*. The composed read is still a finite tail-equivalence certificate over displayed* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{NullSequence}^{\uparrow}$*, and* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *rows; it does not export quotient stream equality, host real equality, selected representatives, countable choice, or ambient completeness.*

Proof. Start with the NameCert obligations of Definition 5940.2. Each accepted packet exposes its two source rows, finite window, dyadic ledger, zero-tail row, comparison row, and Real seal before any same-real read is available.

Use the RegSeqRat route of Definition 5940.3. The middle source row R_1 is therefore visible in both reads only through the displayed finite windows and dyadic ledgers, so the consumer can align the two comparison routes without adding a new stream representative.

Apply zero-tail symmetry from Definition 5940.4 when the second route must be read in the matching orientation. The zero-tail rows and comparison rows remain the only equality witnesses consumed by the Real-facing seal.

Finally use Real seal nonescape from Definition 5940.6. Since every seal read is forced through the displayed source, window, dyadic, null, and comparison rows, composing the two displayed routes yields only the third finite Real-seal route and cannot create any of the excluded quotient, host-equality, representative, choice, or completeness coordinates. □

Theorem 5940.6 (Regular real tail equivalence Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *carrier reaches the seal row* E *only through the finite route*

$$R_0, R_1, W, D, Z, Q, H, C, P, N.$$

Thus the equality surface for the two regular real names is a tail-equivalence certificate over displayed RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, NullSequence$^{\uparrow}$, *and* RegularCauchyComparison$^{\uparrow}$ *rows, not a quotient of streams, host real equality, selected representative, or ambient* Real$^{\uparrow}$ *completeness theorem.*

Proof. Project the accepted carrier. The seal row E is listed after the two source rows, shared window, tolerance ledger, null row, and comparison row, so it is not available as an independent endpoint.

Apply the NameCert obligations of Definition 5940.2. They identify the complete route that must be displayed before the local certificate can expose the Real-facing seal.

Use the RegSeqRat route of Definition 5940.3. It keeps the two regular rational source rows inside the shared finite-window, dyadic, null, and comparison path. The remaining rows H, C, P, N only transport, replay, package, and name that route, so the seal cannot leak quotient stream equality, host real equality, selected representatives, or ambient completeness. □

Closure status for NameCert$_{\text{RegularRealTailEquivalence}^{\uparrow}}$.

Constructive story: A RegularRealTailEquivalence$^{\uparrow}$ packet is generated from two RegSeqRat$^{\uparrow}$ source rows, a shared StreamName$^{\uparrow}$ window, a DyadicRatCore$^{\uparrow}$ tolerance ledger, a NullSequence$^{\uparrow}$ zero-tail row, a RegularCauchyComparison$^{\uparrow}$ comparison route, a Real$^{\uparrow}$ seal boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularRealTailEquivalence}^{\uparrow}}$, seed)

Scope closed: Definition 5940.1, Definition 5940.2, Definition 5940.3, Definition 5940.4, Definition 5940.5, and Definition 5940.6 bind the carrier, local NameCert surface, regular-rational route, zero-tail symmetry route, transitivity route, and Real seal nonescape boundary for same-real tail equivalence.

Formal verification: FormalStatus(NameCert$_{\text{RegularRealTailEquivalence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove quotient stream equality, host real equality, selected representatives, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking for RegularRealTailEquivalence$^{\uparrow}$.

Upgrade path: Scoped closure requires Lean encoding of the carrier, NameCert obligation surface, regular-rational route, and Real seal nonescape boundary.

Theorem 5940.7 (Regular real tail-equivalence classifier forward link)**.** *Every public* RealClassifier$^{\uparrow}$ *read that consumes the regular-real tail-equivalence route of an accepted* RegularRealTailEquivalence$^{\uparrow}$ *packet does so through the downstream classifier chapter chapter 1620. The classifier receives the same finite route through* RegSeqRat$^{\uparrow}$ *source rows, the shared* StreamName$^{\uparrow}$ *window, the* DyadicRatCore$^{\uparrow}$ *tolerance ledger, the visible comparison boundary, the* Real$^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows before any public real-classifier read is exposed.*

Proof. Apply the Regular real tail equivalence Real seal nonescape theorem Definition 5940.6. It forces every Real-facing read through the finite tail route before the seal row is available.

Now read the RealClassifier consumer in chapter 1620. Its regular-real tail-equivalence readback theorem consumes the same regular rational sources, finite window, dyadic ledger, comparison boundary, and seal route before the public classifier boundary is reached.

The structural rows only transport, replay, source, and locally name that displayed packet. Hence the downstream classifier receives the existing tail-equivalence certificate rather than a quotient stream equality, selected representative, host equality, countable-choice schedule, or ambient completeness principle. □

Theorem 5940.8 (Regular real tail equivalence RegSeqRat StreamName route)**.** *Every accepted* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *packet consumes* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{StreamName}^{\uparrow}$ *finite-window scheduling, and* $\mathsf{Real}^{\uparrow}$ *sealing before comparing tails. The dependency route is the finite carrier read*

$$R_0, R_1, W, D, Z, Q, E, H, C, P, N,$$

where the two regular rational source rows are exposed through chapter 301, the shared observation window is scheduled through chapter 126, and the equality-facing endpoint is sealed only through chapter 38. Tail equivalence has no selected limit, host equality, quotient class, or countable-choice coordinate.

Proof. Project the carrier and NameCert obligations of Definition 5940.1 and Definition 5940.2. They place R_0, R_1 before the shared window W and place the Real seal E after the finite tolerance, zero-tail, and comparison rows.

Apply the RegSeqRat route of Definition 5940.3. It forces the two source rows to be read as regular rational names before any tail comparison can be exported.

Next use Real seal nonescape, Definition 5940.6. The seal row is available only after the finite $\mathsf{StreamName}^{\uparrow}$ window, dyadic ledger, zero-tail row, and comparison route have been displayed.

Thus the packet compares tails only through the displayed RegSeqRat, StreamName, and Real route. Since the carrier has no coordinate for selected limits, host equality, quotient classes, or countable choice, none can enter the tail-equivalence read. □

Theorem 5940.9 (Regular real tail equivalence transitive finite-window route)**.** *Let*

$$T_{01} = (R_0, R_1, W_{01}, D_{01}, Z_{01}, Q_{01}, E_{01}, H_{01}, C_{01}, P_{01}, N_{01})$$

and

$$T_{12} = (R_1, R_2, W_{12}, D_{12}, Z_{12}, Q_{12}, E_{12}, H_{12}, C_{12}, P_{12}, N_{12})$$

be accepted $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *packets whose middle* $\mathsf{RegSeqRat}^{\uparrow}$ *source row is the same displayed finite row. Then their same-real reads compose through a common* $\mathsf{StreamName}^{\uparrow}$ *finite window, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers* D_{01}, D_{12}*, the* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail rows* Z_{01}, Z_{12}*, the* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *routes* Q_{01}, Q_{12}*, and the downstream* $\mathsf{Real}^{\uparrow}$ *seal rows. The composed route is still a finite carrier-local certificate from* R_0 *to* R_2*; it introduces no quotient streams, host equality, selected representatives, countable-choice schedule, or ambient completeness row.*

Proof. Project both carriers using Definition 5940.1. The shared middle source is visible as the same displayed R_1 row, while each packet exposes its own finite window, dyadic tolerance ledger, zero-tail row, comparison route, Real seal, and structural rows.

Synchronize the finite windows by the $\mathsf{StreamName}^{\uparrow}$ rows. The two packets may be read by one consumer only after W_{01} and W_{12} are displayed, so the middle source is inspected through finite observation data rather than by selecting a representative stream.

Compose the zero-tail and comparison ledgers through that shared middle source. The rows D_{01}, Z_{01}, Q_{01} carry the first tail comparison and D_{12}, Z_{12}, Q_{12} carry the second, so the only available transitive evidence is their displayed dyadic, null-sequence, and regular-comparison data.

Finally re-seal through $\mathsf{Real}^{\uparrow}$. The seal rows E_{01} and E_{12} are downstream of the same finite route, and transport, replay, provenance, and local naming keep the composed read inside the two accepted packets. Since neither carrier contains quotient streams, host equality, selected representatives, choice, or ambient completeness, none can appear in the composed certificate. □

Theorem 5940.10 (Regular real tail transitive window coverage)**.** *Every completion-facing consumer of a composed* $\mathsf{RegularRealTailEquivalence}^{\uparrow}$ *route must expose both source packets'* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail rows,* $\mathsf{RegularCauchyComparison}^{\uparrow}$ *routes, and terminal* $\mathsf{Real}^{\uparrow}$ *seal rows before it may read the composed same-real certificate. The composed certificate is therefore a finite coverage route over the two displayed packets, not a quotient-stream equality, representative selection, choice schedule, or ambient completeness endpoint.*

Proof. Apply the transitive finite-window route of Definition 5940.9. It already forces the shared middle source, both finite windows, both tolerance ledgers, both zero-tail rows, both regular-comparison routes, and both Real seals to stay visible in the composed read.

Project the two carriers through Definition 5940.1. The rows W_{01} and W_{12} expose the source windows, D_{01} and D_{12} expose the dyadic tolerance data, Z_{01} and Z_{12} expose the zero-tail witnesses, and Q_{01} and Q_{12} expose the regular Cauchy comparison routes.

Only after those rows are displayed can the terminal seals E_{01} and E_{12} be read. Since the carrier presents no quotient stream, selected representative, choice schedule, or ambient completeness row, a completion-facing consumer receives exactly the displayed coverage route before the composed same-real certificate. □

Chapter 5941

A Concrete Naming Certificate for $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$

Following Brouwer and constructive fixed-point treatments, the $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ packet records finite metric approximations to the fixed-point route on compact convex metric data. It consumes the compact finite-net discipline of chapter 129 and the modulus-facing map discipline of chapter 130. The packet names mesh stages, approximate self-map witnesses, finite residual bounds, transport, replay, provenance, and local naming; it does not assert an unrestricted classical Brouwer fixed-point theorem.

Definition 5941.1 (Brouwer fixed-point metric carrier)**.** *A* $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (K, X, M, G, E, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through the finite-cover and finite-net surface of* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *X is the convex-cell ledger naming the finite simplex or box cells inspected at the current mesh scale.*
3. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the self-map whose finite modulus is being used.*
4. *G is the mesh-stage ledger, recording the finite grid, adjacency, and boundary-face rows inspected at that stage.*
5. *E is the approximate fixed-point evidence row, recording a cell, mesh point, or finite witness whose residual is bounded at the displayed tolerance.*
6. *R is the residual-bound row, comparing the map value and witness through metric radii available from K, M, G.*
7. *H records componentwise* hsame *transport for compact metric data, convex-cell rows, continuous-map rows, mesh stages, evidence rows, residual rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for the finite route*

$$K, X, M, G \longmapsto E, R.$$

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointMetric}^{\uparrow}}$ *row.*
The carrier has no coordinate for a host-selected exact fixed point, unrestricted compactness, quotient equality of completions, countable choice, or a global real-valued existence theorem.

Theorem 5941.2 (Brouwer fixed-point metric NameCert obligations)**.** *For an accepted* $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ *carrier*

$$F = (K, X, M, G, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointMetric}^{\uparrow}}$ *surface consists exactly of compact metric admission through K, convex-cell exposure through X, continuous-map modulus access through M, finite mesh-stage discipline through G, approximate witness exposure through E, residual-bound readback through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5941.1. The accepted packet exposes only compact metric data, convex-cell data, a continuous self-map row, finite mesh-stage rows, approximate witness rows, residual-bound rows, and the structural rows H, C, P, N.

The compact and map faces are read through K and M. The compact metric row supplies only finite-net and radius information, while the continuous-map row supplies only the displayed finite modulus data needed by the mesh stage.

The approximation face is read through G, E, R. The mesh row names the finite grid and boundary checks, the evidence row names the finite approximate witness, and the residual row records the metric comparison available at that tolerance.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no exact host fixed point or unrestricted choice principle appears among the rows, the listed obligations exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.BrouwerFixedPointMetricUp.BrouwerFixedPointMetricTasteGate single carrier alignment`

Theorem 5941.3 (Brouwer fixed-point metric finite-approximation non-escape)**.** *Every consumer read of an accepted* $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ *packet factors through the finite approximation route*

$$K, X, M, G \longmapsto E, R \longmapsto H, C, P, N.$$

The consumer may read compact finite-net data, convex-cell rows, continuous-map modulus data, finite mesh stages, approximate witness rows, residual bounds, transport, replay, provenance, and local naming. It cannot read an unrestricted exact fixed point, a host compactness oracle, quotient-completion equality, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness principle from the packet.*

Proof. Start with Definition 5941.1. The only route from compact metric and continuous-map data to a fixed-point-facing conclusion is the displayed finite mesh route through K, X, M, G, E, R.

Apply Definition 5941.2. The local NameCert forces every consumer to inspect compact metric admission, convex-cell exposure, continuous-map access, and mesh-stage discipline before the approximate witness or residual bound can be read.

Transport and replay do not add a stronger theorem. The rows H and C preserve the same finite approximation route, while P and N record provenance and local naming. Hence the packet exports finite approximate fixed-point evidence only, not the classical unrestricted Brouwer assertion. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerFixedPointMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerFixedPointMetric}^{\uparrow}$ packet is generated from compact metric rows, convex-cell rows, continuous self-map modulus rows, finite mesh-stage rows, approximate witness rows, residual-bound rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerFixedPointMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5941.1, Definition 5941.2, and Definition 5941.3 seed the finite metric approximation packet for Brouwer-style fixed-point consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerFixedPointMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close an unrestricted exact Brouwer fixed-point theorem, countable choice, host compactness, quotient-completion equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric routing, continuous-map modulus routing, mesh-stage discipline, approximate witness exposure, residual-bound readback, transport stability, replay stability, provenance, and local naming for BEDC.Derived.BrouwerFixedPointMetricUp.

Chapter 5942

A Concrete Naming Certificate for MetaCICGroundCompilerNormalizationRoute$^{\uparrow}$

The MetaCICGroundCompilerNormalizationRoute$^{\uparrow}$ packet realizes the normalization audit map as finite ground-compiler data. It consumes the critical-path reading of chapter 765 and the candidate-mediated normalization interface recorded in chapter 9383. The packet names generated closed terms, candidate evidence, candidate-set membership, no-infinite-reduction readback, compiler replay, obstruction boundary rows, transport, provenance, and local naming without turning candidate evidence into an unconditional global normalization oracle.

Definition 5942.1 (MetaCIC ground-compiler normalization route carrier)**.** *A* MetaCICGroundCompilerNormalizat *carrier is a finite* BHist *packet*

$$G = (T, A, S, Q, N, R, O, H, C, P, L)$$

with the following displayed rows:

1. *T is the generated closed-term row emitted by the ground compiler.*
2. *A is the typing and closedness audit row for the generated term.*
3. *S is the candidate-evidence row, recording the reducibility candidate, closed candidate, or candidate-set membership route consumed by the normalization proof.*
4. *Q is the Pi/application adequacy row that keeps the candidate evidence stable across the inspected compiler route.*
5. *N is the strong-normalization readback row obtained from the candidate evidence.*
6. *R is the no-infinite-reduction readback row obtained from the same evidence.*
7. *O is the obstruction-boundary row recording that no unconditional closed-term normalization theorem is being claimed.*
8. *H records componentwise* hsame *transport for generated terms, audit rows, candidate rows, adequacy rows, normalization readback, no-infinite-reduction readback, obstruction rows, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for the finite route*

$$T, A, S, Q \longmapsto N, R, O.$$

10. *P is the* Pkg *provenance over* MetaCICGroundCompilerNormalizationRoute$^{\uparrow}$, MetaCICCriticalPath$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *L is the local* NameCert$_{\text{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for unconditional normalization of all closed MetaCIC terms, subject-reduction closure, confluence closure, hidden compiler trust, or host-level termination.

Theorem 5942.2 (MetaCIC ground-compiler normalization route NameCert obligations)**.** *For an accepted* MetaCICGroundCompilerNormalizationRoute$^{\uparrow}$ *carrier*

$$G = (T, A, S, Q, N, R, O, H, C, P, L),$$

the local NameCert$_{\text{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}}$ *surface consists exactly of generated closed-term admission through T, typing and closedness audit through A, candidate-evidence exposure through S, Pi/application adequacy routing through Q, strong-normalization readback through N, no-infinite-reduction readback through R, obstruction-boundary recording through O, component-wise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through L.*

Proof. Project the carrier of Definition 5942.1. The accepted packet exposes generated term data, typing and closedness audit data, candidate evidence, adequacy routing, normalization readback, no-infinite-reduction readback, obstruction boundary, and the structural rows H, C, P, L.

The normalization face is read through S, Q, N, R. Candidate evidence is displayed before any strong-normalization or no-infinite-reduction conclusion is consumed, and the adequacy row records the Pi/application routing needed by the compiler path.

The boundary row O is part of the same packet. It records that the route is evidence-bearing and does not erase the candidate input into a blanket theorem for every closed term.

Finally H, C, P, L transport, replay, source, and name exactly this route. Since the carrier has no coordinate for subject-reduction closure, confluence closure, hidden compiler trust, or host termination, the listed rows exhaust the local NameCert obligations. □

Theorem 5942.3 (MetaCIC ground-compiler candidate non-escape)**.** *Every normalization-facing consumer read of an accepted* MetaCICGroundCompilerNormalizationRoute$^{\uparrow}$ *packet factors through the finite route*

$$T, A, S, Q \longmapsto N, R, O \longmapsto H, C, P, L.$$

Thus strong-normalization and no-infinite-reduction readbacks remain tied to generated closed-term audit rows, candidate evidence, and Pi/application adequacy rows. The packet cannot export unconditional normalization for all closed MetaCIC terms, subject-reduction closure, confluence closure, hidden compiler trust, or host-level termination.

Proof. Start with the carrier of Definition 5942.1. The rows N and R occur only after the generated term audit T, A and the candidate route S, Q have been displayed.

Apply Definition 5942.2. It forces every normalization-facing read to expose the generated closed term, its typing and closedness audit, the candidate evidence, and the adequacy route before reading normalization or no-infinite-reduction consequences.

The obstruction row O keeps the boundary visible, and the rows H, C, P, L only transport, replay, source, and name the same packet. Therefore the consumer cannot detach N or R from candidate evidence or promote the route to a global MetaCIC normalization theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}$ packet is generated from ground-compiler closed-term rows, typing and closedness audit rows, candidate-evidence rows, Pi/application adequacy rows, normalization readback rows, no-infinite-reduction rows, obstruction-boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5942.1, Definition 5942.2, and Definition 5942.3 seed the finite ground-compiler normalization route over candidate evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICGroundCompilerNormalizationRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close unconditional normalization for all closed MetaCIC terms, subject reduction, confluence, hidden compiler trust, host termination, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked generated-term admission, typing and closedness audit, candidate-evidence routing, Pi/application adequacy routing, strong-normalization readback, no-infinite-reduction readback, obstruction-boundary recording, transport stability, replay stability, provenance, and local naming for BEDC.Derived.MetaCICGroundCompilerNormalizatio

Chapter 5943

A Concrete Naming Certificate for $\mathsf{CauchyModulusCompletionSelector}^{\uparrow}$

Following Bishop and Bridges and Weihrauch on effective completion, the $\mathsf{CauchyModulusCompletionSelector}^{\uparrow}$ packet records the finite selector surface that passes an explicit Cauchy modulus into a completion-facing readback. It sits after the explicit rate rows of chapter 292, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the effective completion boundary of chapter 5930. The packet names a displayed selector for modulus evidence only; it does not choose a Cauchy representative, quotient streams, assert ambient real completeness, or export host equality of completed objects.

Definition 5943.1 (Cauchy-modulus completion selector carrier)**.** *A* $\mathsf{CauchyModulusCompletionSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, W, Q, D, E, R, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *rate row, recording the requested precision and the displayed threshold selected for the completion query.*
2. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row read at that threshold.*
3. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows.*
4. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the selected window and by the regular readback.*
5. E *is the* $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$ *completion-facing boundary row.*
6. R *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached through* M, W, Q, D, E.
7. H *records componentwise* hsame *transport for the displayed rows.*
8. C *records displayed* Cont *replay from a completion request through* $M, W, Q, D, E, R, H, P, N$.
9. P *is the* Pkg *provenance over* $\mathsf{CauchyModulusCompletionSelector}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{EffectiveCauchyCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. N *is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusCompletionSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a hidden threshold choice, quotient stream equality, selected limit, host real equality, countable choice, or ambient completion principle.

Lean4 ✓ BEDC.Derived.CauchyModulusCompletionSelectorUp.CauchyModulusCompletionSelectorTasteGate
single carrier alignment

Theorem 5943.2 (Cauchy-modulus completion selector obligations)**.** *For every accepted* CauchyModulusCompletio *carrier*

$$S = (M, W, Q, D, E, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusCompletionSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the explicit modulus row, finite stream-window row, regular rational readback row, dyadic tolerance row, effective completion boundary row, terminal real seal, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming. Every completion-facing selector read must expose* M, W, Q, D, E, R *before* H, C, P, N *can be consumed.*

Proof. Project the carrier of Definition 5943.1. Its coordinates are precisely the modulus, window, readback, tolerance, completion boundary, real seal, transport, replay, provenance, and local naming rows.

The selector route is read through M, W, Q, D, E, R. The modulus row supplies the threshold for the query, the window row identifies the inspected observations, the regular readback row records the Cauchy data, the dyadic row records the precision budget, the effective completion row exposes the completion-facing boundary, and the real row seals that same route.

The remaining rows H, C, P, N transport, replay, package, and name the route already displayed. Since hidden threshold choice, quotient streams, selected limits, countable choice, host equality, and ambient completeness have no carrier coordinate, the listed rows exhaust the local obligations. □

Theorem 5943.3 (Cauchy-modulus selector effective readback)**.** *For every accepted* CauchyModulusCompletionSele *carrier* $S = (M, W, Q, D, E, R, H, C, P, N)$*, any completion-facing query that consumes the effective boundary row* E *must first read the* $\mathsf{CauchyModulus}^{\uparrow}$ *row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*. Thus the selector exposes a finite, modulus-indexed readback before the completion boundary is visible.*

Proof. Begin with Definition 5943.2. It requires the selector read to expose the modulus, window, readback, tolerance, completion, and seal rows before the structural rows are consumed.

The effective boundary row E is therefore not read from a completed object in isolation. It is reached only after M selects the threshold, W displays the finite observations, Q records the regular rational readback, and D records the dyadic budget used by that readback.

Consequently the completion-facing route factors through the same finite selector surface. The carrier has no coordinate for bypassing M, W, Q, D by a host-level limit, quotient stream, or selected representative. □

Theorem 5943.4 (Cauchy-modulus selector real-seal nonescape)**.** *For every accepted* CauchyModulusCompletionSe *carrier* $S = (M, W, Q, D, E, R, H, C, P, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* R *is scoped to the displayed selector route* M, W, Q, D, E*. No downstream Real or Completion consumer can use* R *as evidence for an unrelated Cauchy name, a quotient-completion equality, a selected limit, or an ambient completeness principle.*

Proof. Apply Definition 5943.3. Before the effective boundary is visible, the consumer has already read the explicit modulus, finite window, regular readback, and dyadic tolerance rows.

The real seal row R is listed in the carrier only after that same boundary. Thus any read of R is a read of the displayed selector route M, W, Q, D, E, not of an independent completed-real object.

Finally use Definition 5943.2. The transport, replay, provenance, and naming rows preserve the same route and add no second seal surface. The excluded host equalities, quotient completions, selected limits, and ambient completeness claims therefore cannot escape the finite carrier boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusCompletionSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusCompletionSelector}^{\uparrow}$ packet is generated from an explicit CauchyModulus row, a finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance, EffectiveCauchyCompletion boundary, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusCompletionSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5943.1, Definition 5943.2, Definition 5943.3, and Definition 5943.4 seed the finite selector surface between modulus evidence and completion-facing readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusCompletionSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, selected limits, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus-window selection, regular-readback exposure, dyadic tolerance confinement, effective completion routing, terminal seal scope, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusCompletionSelectorUp.

Chapter 5944

A Concrete Naming Certificate for $\mathsf{KuratowskiHyperspace}^{\uparrow}$

Following Kuratowski and Hausdorff, the $\mathsf{KuratowskiHyperspace}^{\uparrow}$ packet names the completion-facing hyperspace whose public rows are compact-subset presentations, finite Hausdorff-distance comparisons, and Kuratowski embedding transport. It sits after the finite compact-subset carrier of chapter 1660, the embedding route of chapter 7569, and the universal Hausdorff completion handoff of chapter 2242. The chapter records the carrier boundary on which later completion consumers may inspect hyperspace data without importing host powersets, quotient compacta, selected Cauchy limits, or ambient equality of closed subsets.

Definition 5944.1 (Kuratowski hyperspace carrier)**.** *An accepted* $\mathsf{KuratowskiHyperspace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (H, E, U, F, D, S, C, P, M)$$

whose rows have the following public roles.

1. *H is the underlying* $\mathsf{Hyperspace}^{\uparrow}$ *finite compact-subset carrier, including the compact source, Vietoris finite-net rows, directed Hausdorff ledgers, and tolerance readback.*
2. *E is the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *row that sends each compact-subset presentation to its displayed distance-profile history.*
3. *U is the* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *handoff row that admits only completion routes already expressed through Hausdorff-distance Cauchy data.*
4. *F is the compact finite-net readback row used by hyperspace and embedding consumers before any completion read is public.*
5. *D is the distance-stability ledger comparing the Hausdorff classifier on H with the Kuratowski profile classifier on E.*
6. *S, C, P, M are respectively componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{KuratowskiHyperspace}^{\uparrow}}$ *naming row.*

The carrier has no coordinate for a host powerset of closed subsets, a quotient completion class, a selected representative, a selected compact limit, or an ambient extensional equality of subsets.

Lean4 ✓ `BEDC.Derived.KuratowskiHyperspaceUp.KuratowskiHyperspaceTasteGate single carrier alignment`

Theorem 5944.2 (Kuratowski hyperspace distance stability)**.** *Let $K = (H, E, U, F, D, S, C, P, M)$ be an accepted* $\mathsf{KuratowskiHyperspace}^{\uparrow}$ *carrier. Every distance comparison exposed by K is stable under componentwise* hsame *transport exactly when the* $\mathsf{Hyperspace}^{\uparrow}$ *Hausdorff classifier carried by H, the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *distance-profile classifier carried by E, and the* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *Cauchy handoff carried by U are transported through the same* BHist *route. The stability ledger D therefore compares finite-net Hausdorff rows with Kuratowski profile rows; it does not identify compact subsets by quotient equality or by host extensional equality.*

Proof. Read the carrier of Definition 5944.1. The only distance rows available to a consumer are the directed Hausdorff ledgers in H, the distance-profile route in E, and the completion-facing handoff in U.

The compact finite-net row F fixes the source data on which both H and E are evaluated. Transport through S is componentwise, so it preserves the displayed source, the compact finite-net schedules, and the tolerance readback before the comparison reaches D.

Finally replay the transported comparison through C and name it through P, M. Since the carrier lacks quotient representatives and host subset equality coordinates, the only possible stability statement is the finite classifier transport recorded by D. □

Lean4 ✓ `BEDC.Derived.KuratowskiHyperspaceUp.KuratowskiHyperspaceCarrier distance stability`

Theorem 5944.3 (Kuratowski hyperspace NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiHyperspace}^{\uparrow}$ *carrier $K = (H, E, U, F, D, S, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{KuratowskiHyperspace}^{\uparrow}}$ *obligations are exactly the admission of the* $\mathsf{Hyperspace}^{\uparrow}$ *carrier H, the* $\mathsf{KuratowskiEmbedding}^{\uparrow}$ *profile row E, the* $\mathsf{HausdorffCompletionUniversal}^{\uparrow}$ *completion handoff U, the compact finite-net readback F, the distance-stability ledger D, componentwise* hsame *transport S, displayed* Cont *replay C, and* Pkg *provenance P. These obligations exhaust the public naming surface before any completion consumer can read a hyperspace object.*

Proof. Project the carrier coordinates listed in Definition 5944.1. Each coordinate is already a displayed BEDC row: hyperspace finite-net data, Kuratowski profile transport, Hausdorff completion handoff, classifier stability, structural transport, replay, provenance, and local naming.

Use Definition 5944.2 for the only comparison row. It pins the Hausdorff and Kuratowski classifiers to the same finite BHist route and rules out quotient or host equality reads.

It remains to account for completion consumers. They can enter only through U, after H, E, F, D, S, C, P, M have been exposed. Thus the listed rows are exactly the NameCert obligations and no extra carrier coordinate remains public. □

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiHyperspace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KuratowskiHyperspace}^{\uparrow}$ packet is generated from Hyperspace finite compact-subset rows, Kuratowski embedding profile rows, Hausdorff completion handoff rows, compact finite-net readback, classifier stability, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KuratowskiHyperspace}^{\uparrow}}$, seed)

Scope closed: Definition 5944.1, Definition 5944.2, and Definition 5944.3 seed the completion-facing hyperspace carrier over Hyperspace, KuratowskiEmbedding, HausdorffCompletionUniversal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KuratowskiHyperspace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host powersets, quotient compacta, selected compact limits, selected completion representatives, ambient equality of closed subsets, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net readback, Hausdorff/Kuratowski classifier transport, completion handoff exactness, and local NameCert exhaustion for BEDC.Derived.KuratowskiHyperspaceUp.

Chapter 5945

A Concrete Naming Certificate for $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$

Following Brouwer, Bishop–Bridges, and Troelstra–van Dalen on constructive omniscience boundaries, the $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$ packet names the finite alternative-inspection surface that may be discussed without adopting global LPO. It sits next to the $\mathsf{LesserLimitedOmniscience}^{\uparrow}$ boundary of chapter 1977, the $\mathsf{SpeckerSequence}^{\uparrow}$ obstruction surface of chapter 1406, the $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ row of chapter 1666, the $\mathsf{FanTheorem}^{\uparrow}$ bar-inspection boundary of chapter 400, and the L10 readback faces $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet records a named boundary object only: it is not global LPO, not trichotomy for arbitrary reals, not a selected limit principle, not countable choice, not quotient equality, and not ambient Real completeness.

Definition 5945.1 (Limited principle of omniscience carrier)**.** *A* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, S, R, D, E, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite alternative row, recording only the finitely displayed binary alternatives inspected by the packet.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row whose finite source windows are read before any alternative is compared.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *row that converts the inspected stream windows into regular rational readback data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row used by the regular rational readback and by the finite comparison budget.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after S, R, D have been displayed.*
6. *T records transport from the finite alternative row to the sibling boundaries* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, *and* $\mathsf{FanTheorem}^{\uparrow}$.
7. *H records componentwise* hsame *transport for A, S, R, D, E, T.*
8. *C records displayed* Cont *replay for stream-window inspection, regular rational readback, dyadic comparison, Real sealing, and sibling-boundary transport.*

9. *P is the* Pkg *provenance over* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$, $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LimitedPrincipleOmniscience}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, R, D, E, T, H, C, P, N$. It has no coordinate for global LPO, WLPO, LLPO as an adopted principle, trichotomy for arbitrary reals, selected limits, countable choice, quotient equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LimitedPrincipleOmniscienceUp.LimitedPrincipleOmniscienceTasteGate single carrier alignment`

Theorem 5945.2 (Limited principle of omniscience NameCert obligations)**.** *For every accepted* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$ *carrier*

$$L = (A, S, R, D, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LimitedPrincipleOmniscience}^{\uparrow}}$ *surface consists exactly of finite rows for alternative admission,* $\mathsf{StreamName}^{\uparrow}$ *window inspection,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison support,* $\mathsf{Real}^{\uparrow}$ *sealing, sibling-boundary transport through* $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, *and* $\mathsf{FanTheorem}^{\uparrow}$, *componentwise transport, continuation replay, provenance, and local naming. No obligation row may be discharged by global LPO, WLPO, LLPO, trichotomy, selected limits, countable choice, quotient equality, or ambient Real completeness.*

Proof. Carrier admission is projection from Definition 5945.1. The packet exposes the finite alternative row, the three readback rows S, R, D, the terminal seal E, sibling transport T, and the structural rows H, C, P, N.

The sibling references are boundary reads, not imported omniscience principles. The rows named by $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, and $\mathsf{FanTheorem}^{\uparrow}$ record where finite alternative inspection is compared with known constructive obstructions.

The L10 route is also finite. A consumer reads stream windows through S, regular rational data through R, dyadic comparison data through D, and only then the $\mathsf{Real}^{\uparrow}$ seal E.

Since no row stores global LPO, WLPO, LLPO, trichotomy, selected limits, countable choice, quotient equality, or ambient Real completeness, the NameCert obligations are exhausted by the displayed carrier coordinates. □

Theorem 5945.3 (Limited principle of omniscience Real-completion refusal)**.** *Every completion-facing consumer of an accepted* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$ *carrier must route through the finite display*

$$\mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

together with the finite alternative row A and the sibling-boundary row T. The route may inspect named alternatives and compare them with $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, *and* $\mathsf{FanTheorem}^{\uparrow}$, *but it cannot export global LPO, real trichotomy, selected limits, countable choice, quotient equality, or ambient Real completeness.*

Proof. Start from the NameCert obligation theorem. It fixes the completion-facing route to S, R, D, E and keeps the alternative and sibling-boundary rows visible as packet data.

The regular rational and dyadic rows are read before the Real seal. Thus the $\mathsf{Real}^{\uparrow}$ endpoint is reached only after finite stream-window inspection and finite rational comparison support have been displayed.

The refused principles are not hidden in the structural rows. Transport by H, replay by C, provenance by P, and naming by N preserve the same displayed route and add no coordinate for global omniscience, choice, quotient equality, selected limits, trichotomy, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{LimitedPrincipleOmniscience}\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LimitedPrincipleOmniscience}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5945.1, Definition 5945.2, and Definition 5945.3 seed the finite alternative-inspection boundary for L10 Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LimitedPrincipleOmniscience}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global LPO, WLPO, LLPO, real trichotomy, selected limits, countable choice, quotient equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite alternative inspection, L10 readback exactness, sibling-boundary transport, Real seal refusal, and global-omniscience refusal for BEDC.Derived.LimitedPrincipleOmniscienceUp.

Constructive story: A $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$ packet is generated from finite alternative rows, StreamName windows, RegSeqRat readback, DyadicRatCore comparison support, a Real seal, sibling-boundary transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5946

A Concrete Naming Certificate for $\mathsf{HenselLemma}^{\uparrow}$

Following Hensel and standard p-adic algebra, the $\mathsf{HenselLemma}^{\uparrow}$ packet records the finite BEDC surface of a simple-root lifting certificate over displayed local arithmetic data. It sits downstream of the p-adic carrier of chapter 100, the local-field source of chapter 173, the polynomial carrier of chapter 50, the field schema of chapter 45, and the ring schema of chapter 43. The packet names a finite lifting witness and its Newton-style nonescape ledger; it does not import quotient roots, a selected root function, countable choice, full algebraic closure, or ambient completeness beyond the displayed source rows.

Definition 5946.1 (Hensel lemma carrier)**.** *A* $\mathsf{HenselLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (K, R, F, P, A, V, N, E, T, C, Q, L)$$

with the following displayed rows:

1. *K is the* $\mathsf{Padic}^{\uparrow}$ *source row, exposing only the finite valuation and congruence windows inspected by the lifting packet.*
2. *R is the* $\mathsf{LocalField}^{\uparrow}$ *coefficient row, supplying the local-field arithmetic reads used by the polynomial and derivative windows.*
3. *F is the* $\mathsf{Field}^{\uparrow}$ *compatibility row for the nonzero derivative and unit comparisons needed by the simple-root test.*
4. *P is the* $\mathsf{Poly}^{\uparrow}$ *row, listing the displayed polynomial, derivative, coefficient windows, and evaluation reads at the approximate root.*
5. *A is the approximate-root row, recording the chosen finite residue-window representative and the displayed smallness comparison for the polynomial value.*
6. *V is the valuation-separation row, recording that the derivative unit window and the polynomial-value window are separated enough to start the lifting route.*
7. *N is the Newton-correction ledger, listing the finite correction windows, derivative reads, and valuation gains that the packet promises to replay.*
8. *E is the nonescape row, asserting that every correction window stays inside the same displayed p-adic ball before any root conclusion is exposed.*

9. *T records componentwise* hsame *transport for the source, coefficient, polynomial, approximate-root, valuation, correction, and nonescape rows.*
10. *C records the displayed* Cont *replay route from source arithmetic through polynomial evaluation, correction, nonescape, provenance, and local naming.*
11. *Q is the* Pkg *provenance over* $\mathsf{HenselLemma}^{\uparrow}$, $\mathsf{Padic}^{\uparrow}$, $\mathsf{LocalField}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Field}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *L is the local* $\mathsf{NameCert}_{\mathsf{HenselLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, R, F, P, A, V, N, E, T, C, Q, L$. It has no coordinate for selected roots, quotient roots, countable choice, a full algebraic closure, or an ambient completeness theorem.

Theorem 5946.2 (Hensel lemma NameCert obligations)**.** *For every accepted* $\mathsf{HenselLemma}^{\uparrow}$ *carrier $H = (K, R, F, P, A, V, N, E, T, C, Q, L)$, the local* $\mathsf{NameCert}_{\mathsf{HenselLemma}^{\uparrow}}$ *surface consists exactly of carrier admission for the p-adic source row, local-field coefficient row, field-unit compatibility row, polynomial and derivative evaluation row, approximate-root row, valuation-separation row, Newton-correction ledger, nonescape row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every lifting read must expose these finite rows before it can be consumed as a Hensel-style certificate.*

Proof. Project the carrier of Definition 5946.1. The arithmetic source is visible only through K, R, F, while the polynomial and derivative reads are visible only through P and the approximate-root row A.

The lifting route then passes through V, N, and E. The valuation-separation row starts the correction schedule, the Newton ledger records the displayed correction windows, and the nonescape row keeps those windows inside the same p-adic ball.

The structural rows T, C, Q, L transport, replay, source, and name the same finite packet. Since no further coordinate occurs in the carrier, the certificate cannot export a selected root function, quotient-root semantics, countable choice, full algebraic closure, or an ambient completeness principle. □

Lean4 ✓ `BEDC.Derived.HenselLemmaUp.HenselLemmaTasteGate single carrier alignment`

Theorem 5946.3 (Newton witness nonescape)**.** *Every accepted* $\mathsf{HenselLemma}^{\uparrow}$ *packet exposes a Newton witness only through the valuation-separation row V, the correction ledger N, and the nonescape row E. A consumer therefore receives a finite chain of displayed p-adic windows whose corrections stay inside the original ball before any root-lifting read is available.*

Proof. Read the dependency order in Definition 5946.1. The correction ledger N is not independent of the simple-root test; it is reached only after the source rows K, R, F, the polynomial row P, the approximate-root row A, and the valuation-separation row V have been displayed.

The row E records the nonescape check for the same correction windows. Replaying C can inspect the valuation gains and ball-membership comparisons, but it cannot detach them from the finite windows named by K, R, P, A, V, and N.

Transport along T preserves those rows componentwise. Hence the witness remains a finite Newton-style nonescape certificate and not a selected root, quotient-root object, choice sequence, algebraic-closure construction, or completeness theorem. □

Lean4 ✓ `BEDC.Derived.HenselLemmaUp.HenselLemmaNewtonWitnessNonescape`

Falsifiable prediction: Every accepted $\mathsf{HenselLemma}^{\uparrow}$ lifting read exposes the p-adic source, local-field coefficient row, field-unit compatibility row, polynomial and derivative windows, approximate-root row, valuation-separation row, Newton-correction ledger, nonescape row, componentwise hsame transport, Cont replay, Pkg provenance, and local naming before any Hensel-style consumer may use it.

Independence witness: padic, localfield, polynomial, field, ring: $\mathsf{HenselLemma}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a simple-root valuation test, polynomial derivative windows, Newton-correction ledger, and nonescape witness in one finite lifting packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HenselLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{HenselLemma}^{\uparrow}$ packet is generated from p-adic source data, local-field coefficient rows, field-unit compatibility, polynomial and derivative evaluation windows, an approximate-root row, valuation separation, a Newton-correction ledger, nonescape checks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HenselLemma}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5946.1, Definition 5946.2, and Definition 5946.3 seed the finite simple-root lifting route over p-adic, local-field, polynomial, field, and ring rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HenselLemma}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected roots, quotient roots, countable choice, full algebraic closure, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, simple-root valuation separation, polynomial derivative readback, Newton-correction ledger exactness, nonescape stability, transport, replay, provenance, and local naming for BEDC.Derived.HenselLemmaUp.

Chapter 5947

A Concrete Naming Certificate for $\mathsf{LocatedRiemannIntegral}^{\uparrow}$

Following Bishop–Bridges constructive integration, the $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ packet names a located Riemann integral by finite partition data, regulated handoff rows, dyadic error ledgers, and the terminal $\mathsf{Real}^{\uparrow}$ seal. It consumes the Riemann-sum surface of chapter 1615, the regulated-integral handoff of chapter 1557, the dyadic core of chapter 291, the StreamName windows of chapter 126, the regular rational readback of chapter 301, and the real-name boundary of chapter 38. The packet is a finite L10 analysis certificate; it is not host integral equality, a selected limiting process, open-cover compactness, countable choice, or ambient Real completeness.

Definition 5947.1 (LocatedRiemannIntegral carrier)**.** *A* $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (A, M, \Sigma, G, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the located interval source row. It records the two located endpoints, the inhabited interval evidence visible to the packet, and the side condition that all partition points are read inside the displayed source interval.*
2. *M is the finite partition mesh row. It lists the ordered nodes, the mesh bound, and the finite refinement witnesses admitted for this integral request.*
3. *Σ is the* $\mathsf{RiemannSum}^{\uparrow}$ *row, recording the tagged subinterval sums obtained from A and M through the sibling Riemann-sum certificate.*
4. *G is the* $\mathsf{RegulatedIntegral}^{\uparrow}$ *handoff row. It exposes the regulated step approximation and integral handoff only after the partition mesh and RiemannSum rows have been displayed.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the dyadic error requests, mesh-refinement budget, and comparison rows used to certify located convergence of the displayed sums.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row for finite observation of the sum stream and regulated handoff stream.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, translating the displayed finite windows into regular rational approximants for the integral value.*

8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the source interval, mesh, RiemannSum row, regulated handoff, dyadic ledger, StreamName windows, and RegSeqRat readback have all been replayed.*
9. *H records componentwise* hsame *transport for $A, M, \Sigma, G, D, W, R, E, C, P, N$.*
10. *C records displayed* Cont *replay from a tolerance request through $A, M, \Sigma, G, D, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{LocatedRiemannIntegral}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{RegulatedIntegral}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRiemannIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, \Sigma, G, D, W, R, E, H, C, P, N$. It has no coordinate for host integral equality, selected limits, open-cover compactness, countable choice, ambient Real completeness, or any equality outside the displayed finite rows.

Lean4 (definition) `BEDC.Derived.LocatedRiemannIntegralUp.LocatedRiemannIntegralUp`

Theorem 5947.2 (LocatedRiemannIntegral NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ *carrier*

$$I = (A, M, \Sigma, G, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRiemannIntegral}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval source, finite partition mesh, RiemannSum row, regulated-integral handoff, dyadic tolerance ledger, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *partition-mesh exactness, requiring every displayed sum to read the same located interval row A and one of the finite partition refinements recorded by M;*
3. *located-modulus exposure, requiring the dyadic ledger D to display the mesh and tolerance budget used by the RiemannSum and regulated-integral rows;*
4. *sum-convergence ledger, requiring the route from Σ through G, D, W, R to expose the finite windows and regular rational approximants before the Real seal can be read;*
5. *Real seal nonescape, requiring E to be downstream of A, M, Σ, G, D, W, R and to export no host integral equality or ambient completeness theorem;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming through H, C, P, N;*
7. *refusal of host integral equality, selected limits, open-cover compactness, countable choice, and ambient Real completeness as carrier coordinates, replay edges, or provenance rows.*

No obligation is stored outside the displayed packet.

Proof. Carrier admission is projection from Definition 5947.1. The packet displays the located interval, mesh, RiemannSum, regulated handoff, dyadic, window, readback, seal, transport, replay, provenance, and naming rows as one finite BHist carrier.

Partition-mesh exactness follows by reading A and M before Σ. The Riemann-sum row is tied to the same interval source and the same finite refinement ledger, so no selected partition limit or host integral object can supply the sum.

The located-modulus and sum-convergence obligations are then read through D, W, R. The dyadic ledger supplies the tolerance and mesh budget, the StreamName row exposes finite observation windows, and the RegSeqRat row gives the regular rational readback before E is available.

Finally H, C, P, N transport, replay, source, and name only that displayed route. Since the carrier has no row for host integral equality, selected limits, open-cover compactness, countable choice, or ambient Real completeness, the local NameCert surface is exhausted by the listed obligations. □

Falsifiable prediction: Every $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ acceptance trace exposes one located interval source, one finite partition mesh, one $\mathsf{RiemannSum}^{\uparrow}$ row, one $\mathsf{RegulatedIntegral}^{\uparrow}$ handoff, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback, one terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: riemannsum, regulatedintegral, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the located interval source, mesh, sum, regulated handoff, dyadic tolerance, finite-window, regular-readback, Real-seal, transport, replay, provenance, and local NameCert rows as one integral packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRiemannIntegral}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRiemannIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5947.1 and Definition 5947.2 seed a finite constructive Riemann integral packet over located intervals, partition meshes, Riemann sums, regulated handoff, dyadic tolerance, readback, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRiemannIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host integral equality, selected limiting processes, open-cover compactness, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, partition-mesh exactness, located-modulus exposure, sum-convergence ledger, Real seal nonescape, transport, replay, provenance, and local naming for the displayed $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ packet.

Constructive story: A $\mathsf{LocatedRiemannIntegral}^{\uparrow}$ packet is generated from a located interval source, a finite partition mesh, a RiemannSum row, a RegulatedIntegral handoff, dyadic tolerance, StreamName windows, RegSeqRat readback, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5948

A Concrete Naming Certificate for $\mathsf{KolmogorovQuotient}^{\uparrow}$

Following Kolmogorov and Bourbaki's separation discipline for topological spaces, the $\mathsf{KolmogorovQuotient}^{\uparrow}$ packet records the constructive T_0 reflection as finite observable-open data rather than as a host quotient object. It sits next to the indexed-open topology of chapter 88, the metric-generated topology route of chapter 57, the membership discipline of chapter 3668, and the specialization reflection packet of chapter 3338. The packet names indistinguishability by mutual observable-open evidence and exposes a separated carrier only through displayed representative rows.

Definition 5948.1 (Kolmogorov quotient carrier)**.** *A* $\mathsf{KolmogorovQuotient}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, O, E, R, S, H, C, P, N)$$

with the following rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *indexed-open row, carrying the open indices and membership reads used to test observable separation.*
2. *O is the observable-open classifier row, recording for each displayed point row which indexed opens are observed as present.*
3. *E is the indistinguishability ledger: two point rows are equivalent for this packet exactly when every observable-open row in O agrees in both directions.*
4. *R is the representative-display row. It records a finite public name for an E-class but does not select a host representative by choice.*
5. *S is the separated-reader row. It exposes a T_0 read only after E and R have been displayed, so separated equality is mutual observable-open agreement.*
6. *H records componentwise* hsame *transport for T, O, E, R, S and the structural rows.*
7. *C records* Cont *replay from a topology request through observable opens, indistinguishability, representative display, and separated reading.*
8. *P is the* Pkg *provenance over* $\mathsf{KolmogorovQuotient}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TZeroReflection}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovQuotient}^{\uparrow}}$ *row.*

The carrier has no coordinate for a quotient primitive, proof irrelevance over opens, selected representative function, ambient set quotient, powerset extensionality, or excluded-middle split of points.

Definition 5948.2 (Kolmogorov quotient classifier)**.** *For an accepted carrier $K = (T, O, E, R, S, H, C, P, N)$, the displayed classifier on public quotient names is the finite route*

$$O, E, R, S \longmapsto H, C, P, N.$$

Two public names classify as the same $\mathsf{KolmogorovQuotient}^{\uparrow}$ *point only when their observable-open rows agree through E, their displayed representative rows are preserved by H, and their separated reads replay through C under the same provenance and local name rows.*

Theorem 5948.3 (Kolmogorov quotient NameCert obligations)**.** *For every accepted* $\mathsf{KolmogorovQuotient}^{\uparrow}$ *carrier $K = (T, O, E, R, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KolmogorovQuotient}^{\uparrow}}$ *surface consists exactly of topology-source admission through T, observable-open classifier exposure through O, mutual-observation indistinguishability through E, representative-display exposure through R, separated-reader exposure through S, componentwise* hsame *stability through H,* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.KolmogorovQuotientUp.KolmogorovQuotientNameCertObligations`

Proof. Project the accepted carrier of Definition 5948.1. The only topology data available to the packet is the indexed-open row T and its observable classifier O.

The indistinguishability row E is therefore forced to compare point rows by observable-open agreement. The representative-display row R records public names for such rows, but it has no host choice coordinate from which to select representatives.

The separated-reader row S can be consumed only after E and R have been exposed. Componentwise transport H, replay C, provenance P, and local naming N preserve that same route, so every NameCert obligation is one of the displayed rows and no quotient primitive or ambient set identification is added. □

Theorem 5948.4 (Kolmogorov quotient T0 reflection)**.** *Every separated consumer of an accepted* $\mathsf{KolmogorovQuotient}^{\uparrow}$ *packet factors through the finite reflection route*

$$\mathsf{Topology}^{\uparrow} \longmapsto \mathsf{KolmogorovQuotient}^{\uparrow} \longmapsto \mathsf{TZeroReflection}^{\uparrow}.$$

Thus the consumer may distinguish public quotient names exactly by observable-open disagreement and may identify them only by the mutual-observation row E transported through H, C, P, N.

Proof. Start with the topology row T and observable classifier O. These rows determine which opens are visible at each displayed point row.

Use the indistinguishability ledger E. If every observable-open read agrees in both directions, the packet admits the same public quotient name through R; if a displayed open separates the rows, the separated-reader row S records the T_0 distinction.

Finally replay the structural tail. The rows H, C, P, N keep the observable-open route fixed, and the sibling $\mathsf{TZeroReflection}^{\uparrow}$ packet of chapter 3338 consumes precisely the separated specialization surface. No quotient primitive, selected representative, or powerset equality participates in the reflection. □

Theorem 5948.5 (Kolmogorov quotient metric-source compatibility)**.** *If a* $\mathsf{Met}^{\uparrow}$ *source supplies the topology row* T *through the metric-generated route of* $\mathsf{Topology}^{\uparrow}$*, then every* $\mathsf{KolmogorovQuotient}^{\uparrow}$ *separated read still factors through the same observable-open rows* O, E, R, S, H, C, P, N*. The metric source contributes open-ball observations; it does not add a second equality on quotient names, a Cauchy-completion selector, or a Hausdorff quotient of points.*

Proof. Read the metric source only through the topology handoff. The metric carrier may generate basic open rows, but the $\mathsf{KolmogorovQuotient}^{\uparrow}$ packet consumes them as the indexed-open row T.

The observable classifier O and indistinguishability row E then perform the same mutual-open test as in the purely topological case. The rows R and S expose public quotient names and T_0 separation only after this test.

The structural rows H, C, P, N preserve the same route. Since no additional metric equality row belongs to the carrier, a metric source cannot replace observable-open indistinguishability by a host metric quotient or completion-side selector. □

Falsifiable prediction: A claimed $\mathsf{KolmogorovQuotient}^{\uparrow}$ read that identifies public names without displaying $T, O, E, R, S, H, C, P, N$, or that selects a representative by a host choice function, refutes the carrier of Definition 5948.1.

Independence witness: topology, metricspace, tzero-reflection, setlike: $\mathsf{KolmogorovQuotient}^{\uparrow}$ is not bijective to any listed sibling because it carries the joint observable-open indistinguishability ledger, representative-display row, separated-reader row, and non-quotient NameCert route. TopologyUp supplies indexed opens without the separated public-name row; MetricSpaceUp supplies distance-generated opens without a quotient-name surface; TZeroReflectionUp consumes specialization separation without this representative-display ledger; SetLikeUp supplies membership discipline without topological observability.

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovQuotient}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KolmogorovQuotient}^{\uparrow}$ packet is generated from a Topology indexed-open row, observable-open classifier, mutual-observation indistinguishability ledger, representative-display row, separated-reader row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovQuotient}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5948.1, Definition 5948.2, Definition 5948.3, and Definition 5948.4 seed the finite constructive T_0 reflection surface without quotient primitives.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovQuotient}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host quotient object, selected representatives, proof irrelevance over opens, ambient powerset extensionality, excluded middle over points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, observable-open exactness, mutual-observation stability, representative-display coverage, separated-reader exactness, and sibling handoff for BEDC.Derived.KolmogorovQuotientUp.

Chapter 5949

A Concrete Naming Certificate for BoundedTypecheckingBudget$^{\uparrow}$

The BoundedTypecheckingBudget$^{\uparrow}$ packet realizes the MetaCIC decidability audit map of chapter 9379 as finite checker rows with visible refusal boundaries. It packages the type-inference checker, structural same-term checker, bounded normal-equality checker, finished-normalization evidence, and gallery audit rows into a NameCert object that can be consumed without claiming a global conversion oracle. It is adjacent to MetaCICDecidableBoundary$^{\uparrow}$ of chapter 749, NormalFalseExclusion$^{\uparrow}$ of chapter 732, and ClosedNormalConfluenceSeal$^{\uparrow}$ of chapter 741: the first names the decidability boundary, the second records normal-false exclusion discipline, and the third records the closed normal-confluence seal consumed when bounded equality has endpoint evidence.

Definition 5949.1 (Bounded typechecking budget carrier)**.** *A* BoundedTypecheckingBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (\Gamma, T, Q, N, F, G, R, H, C, P, L)$$

with the following rows:

1. Γ *is the represented context row for the MetaCIC checker.*
2. T *is the option-valued type-inference row, exposing* `inferTypeCtx` *and its closed wrapper as finite checker data.*
3. Q *is the same-term row, exposing structural equality and* `checkSameTerm` *rather than proof-irrelevant equality.*
4. N *is the bounded normalizer row, carrying a visible fuel value and the bounded normal-form outputs used by* `decideClosedBetaConv`*.*
5. F *is the finished-normalization witness row. It is present only when the bounded normalizer has the endpoint evidence required by the audit map.*
6. G *is the gallery row, recording the executable sanity witnesses for Boolean, natural-number, pair, option, list, sum, and polymorphic examples.*
7. R *is the refusal-boundary row, recording that closedness and global beta-conversion decidability are not exported when the relevant endpoint data is absent.*
8. H *records componentwise* hsame *transport for* Γ, T, Q, N, F, G, R *and the structural rows.*

9. *C records* Cont *replay from a typechecking request through inference, same-term checking, bounded normal comparison, finished evidence, gallery rows, and refusal boundaries.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$, $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$, $\mathsf{NormalFalseExclusion}^{\uparrow}$ $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{BoundedTypecheckingBudget}^{\uparrow}}$ *row.*

The carrier has no coordinate for unrestricted beta-conversion decidability, fuel-erased normalization, proof-irrelevant term equality, external closedness decision, hidden oracle, or host typechecker.

Theorem 5949.2 (Bounded typechecking budget NameCert obligations)**.** *For every accepted* $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$ *carrier* $B = (\Gamma, T, Q, N, F, G, R, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedTypecheckingBudge}}$ *surface consists exactly of context admission through* Γ*, type-inference exposure through* T*, structural same-term exposure through* Q*, bounded-normalizer exposure through* N*, finished-endpoint evidence through* F *when present, gallery audit exposure through* G*, refusal-boundary exposure through* R*, componentwise* hsame *stability through* H*,* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* L*.*

Proof. Project the accepted carrier of Definition 5949.1. The represented context row Γ and inference row T are the only typechecking entry points.

The same-term checker Q and bounded normalizer N provide the equality-facing data. The finished row F is read only when endpoint evidence exists, while R records the refusal of a global conversion claim when that evidence is absent.

The gallery row G is audit-facing evidence for the executable examples, not a source of a general theorem. Componentwise transport H, replay C, provenance P, and local naming L preserve the same finite route, so the NameCert obligations cannot import a hidden oracle, host typechecker, or unrestricted conversion decision. □

Theorem 5949.3 (Bounded typechecking budget MetaCIC decidable handoff)**.** *Every consumer that cites the MetaCIC decidability audit map through an accepted* $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$ *packet must factor through the finite route*

$$\Gamma, T, Q, N, F, G, R \longmapsto H, C, P, L.$$

Consequently type inference is public at the checker interface, structural same-term equality is public through Q*, bounded normal equality is public through* N*, and full closed beta-conversion is public only under the finished endpoint row* F*; otherwise the refusal row* R *remains visible.*

Proof. Begin with the typechecking side. The context row Γ and inference row T expose the checker interface described by chapter 9379; they do not expose a host typechecker.

Move to equality. The row Q supplies structural same-term checking, while N supplies the bounded normalizer comparison at a displayed fuel. These rows are enough for bounded normal equality and for the soundness statements named by the audit map.

The endpoint condition is separate. If F is present, the consumer may read the finished-normalization handoff to the closed normal-confluence seal. If F is absent, the refusal row R is part of the same public packet and blocks any claim of global closed beta-conversion decidability.

Finally replay H, C, P, L. These rows transport, replay, source, and name the exact budget route, so every MetaCIC decidability handoff remains a finite checker-row handoff rather than an unrestricted oracle. □

Theorem 5949.4 (Bounded typechecking budget refusal boundary)**.** *For every accepted* BoundedTypecheckingBudg *carrier* $B = (\Gamma, T, Q, N, F, G, R, H, C, P, L)$*, any consumer that has* $\Gamma, T, Q, N, G, H, C, P, L$ *but lacks the finished endpoint row* F *must keep the refusal-boundary row* R *in scope. Such a consumer may cite type-inference soundness and completeness, same-term correctness, gallery sanity, and bounded-normalizer output equality, but it may not export fuel-erased normalization, unrestricted beta-conversion decidability, or a closedness decision theorem.*

Proof. Project the carrier and remove only the endpoint row F from the consumer-visible route. The rows Γ, T, Q, N, G still expose the context, inference checker, same-term checker, bounded normalizer, and gallery audit surface.

The missing row matters exactly at the conversion boundary. Without F, the bounded normalizer output row N does not assert that the displayed fuel has reached finished closed normal forms, so the stronger conversion claim is not available.

The refusal row R is therefore mandatory. Transport H, replay C, provenance P, and local naming L preserve the same absence, preventing the consumer from replacing the missing endpoint by a host oracle, proof irrelevance, or fuel-erased normalization theorem. □

Falsifiable prediction: A claimed $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$ consumer that exports global closed beta-conversion decidability while omitting the finished endpoint row F and refusal-boundary row R refutes the carrier of Definition 5949.1.

Independence witness: metacic-decidable-boundary, normal-false-exclusion, closed-normal-confluence-seal: $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$ is not bijective to any listed sibling because it carries the joint context, inference, same-term, bounded-normalizer, finished-endpoint, gallery, refusal, transport, replay, provenance, and local NameCert rows. The decidable-boundary sibling names the audit edge; the normal-false-exclusion sibling controls false normal reads; the closed-normal-confluence seal supplies endpoint compatibility when F is present.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedTypecheckingBudget}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedTypecheckingBudget}^{\uparrow}$ packet is generated from a represented context row, type-inference checker row, same-term checker row, bounded-normalizer row, finished endpoint row, gallery audit row, refusal-boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedTypecheckingBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5949.1, Definition 5949.2, and Definition 5949.3 seed the finite checker-budget handoff for the MetaCIC decidability audit map.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedTypecheckingBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted beta-conversion decidability, fuel-erased normalization, an external closedness decision theorem, proof-irrelevant term equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, inference checker exactness, same-term checker exactness, bounded-normalizer exactness, finished-endpoint handoff, gallery coverage, and refusal-boundary nonescape for BEDC.Derived.BoundedTypecheckingBudgetUp.

Chapter 5950

A Concrete Naming Certificate for $\mathsf{ScottTopology}^{\uparrow}$

Following Scott and Gierz–Hofmann–Keimel–Lawson–Mislove–Scott, the $\mathsf{ScottTopology}^{\uparrow}$ packet records the topology generated by upper sets inaccessible by directed suprema as finite BEDC naming data. It sits beside the directed-complete order surface of chapter 5905, the domain packet of chapter 4164, the interval real presentation of chapter 2083, the way-below interval route of chapter 1545, and the indexed-open topology surface of chapter 88. The packet names Scott-open observations for downstream domain-theoretic consumers; it is not a host powerset topology, quotient-open equality, selected directed supremum operation, arbitrary open lattice, choice principle, or Lean-verified bridge.

Definition 5950.1 (ScottTopology carrier)**.** *A* $\mathsf{ScottTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, S, U, I, B, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DCPO}^{\uparrow}$ *source row, exposing the finite order, directed-family schedule, supremum row, and lower-bound exactness already named by the directed-complete packet.*
2. *S is the* $\mathsf{ScottDomain}^{\uparrow}$ *source row, recording the domain-facing compact-basis and represented-space handoff rows that may consume the same directed-supremum route.*
3. *U is the upper-set row, saying that every accepted open observation is upward closed with respect to the displayed order in D.*
4. *I is the directed-supremum inaccessibility row, requiring membership of a directed supremum to be witnessed by membership at some displayed directed-family stage before the supremum is treated as open.*
5. *B is the way-below basis row. For interval-domain real consumers it reads the* $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ *refinements used by chapter 2083; for general domain consumers it records the compact-basis approximation row already exposed by S.*
6. *O is the* $\mathsf{Topology}^{\uparrow}$ *handoff row, exporting the accepted Scott-open observations as indexed open rows without adding a second open-set carrier.*
7. *H records componentwise* hsame *transport for the DCPO source, Scott-domain source, upper-set row, directed-supremum inaccessibility row, way-below basis row, topology handoff, replay, provenance, and naming rows.*

8. *C records displayed* Cont *replay from a Scott-open request through D, S, U, I, B, O, H.*
9. *P is the* Pkg *provenance over* $\mathsf{ScottTopology}^{\uparrow}$, $\mathsf{DCPO}^{\uparrow}$, $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{ScottReal}^{\uparrow}$, $\mathsf{IntervalDomainWayBelow}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ScottTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, U, I, B, O, H, C, P, N$. It has no coordinate for arbitrary powersets, quotient-open equality, selected directed suprema, host topological equality, choice, or a Lean theorem.

Lean4 ✓ `BEDC.Derived.ScottTopologyUp.ScottTopologyTasteGate single carrier alignment`

Theorem 5950.2 (ScottTopology NameCert obligations)**.** *For every accepted* $\mathsf{ScottTopology}^{\uparrow}$ *carrier $T = (D, S, U, I, B, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ScottTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{DCPO}^{\uparrow}$ *and* $\mathsf{ScottDomain}^{\uparrow}$ *sources, upper-set closure, directed-supremum inaccessibility, way-below basis compatibility,* $\mathsf{Topology}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Scott-open consumer must read $D, S, U, I, B, O, H, C, P, N$ before treating an open observation as public.*

Proof. Project the carrier of Definition 5950.1. The order source and domain source are D and S, the Scott-open tests are U, I, B, the topology export is O, and the structural rows are H, C, P, N.

The upper-set row U is read against the displayed order in D. Thus an open observation cannot be consumed independently of the directed-complete order carrier.

The directed-supremum inaccessibility row I is read against the directed-family and supremum rows already named by $\mathsf{DCPO}^{\uparrow}$ and by the $\mathsf{ScottDomain}^{\uparrow}$ source. It requires a displayed stage witness before a supremum membership is exported as open data.

The way-below basis row B supplies the compact-basis route. For real-facing domain consumers this route is the $\mathsf{IntervalDomainWayBelow}^{\uparrow}$ refinement used by $\mathsf{ScottReal}^{\uparrow}$; for general domain consumers it is the compact approximation row of S.

Finally O hands the accepted Scott-open observations to $\mathsf{Topology}^{\uparrow}$, while H, C, P, N transport, replay, source, and name the same finite packet. Since quotient opens, arbitrary powersets, host topological equality, selected suprema, choice, and Lean verification are absent from the carrier, these rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScottTopology}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ScottTopology}^{\uparrow}$ packet is generated from a DCPO source row, a ScottDomain source row, upper-set evidence, directed-supremum inaccessibility, way-below basis compatibility, a TopologyUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ScottTopology}^{\uparrow}}$, seed)

Scope closed: Definition 5950.1 and Definition 5950.2 seed the finite Scott-open interface over DCPO, ScottDomain, ScottReal, IntervalDomainWayBelow, and Topology rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ScottTopology}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient opens, arbitrary powersets, host topological equality, selected directed suprema, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-set closure, directed-supremum inaccessibility, way-below basis compatibility, TopologyUp handoff, transport, replay, provenance, and local naming for BEDC.Derived.ScottTopologyUp.

Chapter 5951

A Concrete Naming Certificate for $\mathsf{FiniteRealSupremum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteRealSupremum}^{\uparrow}$ packet records the finite upper-bound search used when a nonempty finite family of constructive real names must expose a supremum row before any compact finite-net or finite-bound consumer can read it. It sits over the rational arithmetic base of chapter 37, the terminal real-name seal of chapter 38, the displayed comparison discipline of chapter 54, the binary min–max sibling of chapter 1674, and the finite-net fold route of chapter 619. The packet names finite supremum data only; it does not import host maximum, quotient real equality, order trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5951.1 (Finite real supremum carrier)**.** *A* $\mathsf{FiniteRealSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R, I, O, M, U, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the nonempty finite* $\mathsf{Real}^{\uparrow}$ *packet, listing the real-name rows retained by the supremum request together with their finite index witnesses.*
2. *I is the inhabitedness row, exposing the selected starting index and its* $\mathsf{Real}^{\uparrow}$ *name before any comparison ledger is read.*
3. *O is the finite comparison ledger, recording the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison rows used to compare retained* $\mathsf{Real}^{\uparrow}$ *names through rational readback and displayed order witnesses.*
4. *M is the* $\mathsf{RealMinMax}^{\uparrow}$ *fold ledger, showing how the finite comparison rows select the displayed upper-bound candidate from the retained real rows.*
5. *U is the selected upper-bound row, asserting that every retained member of R is bounded above by the candidate selected through M and O.*
6. *L is the least-bound verification row, asserting that every displayed upper bound for R receives a comparison witness from the selected candidate into that bound.*
7. *H records componentwise* hsame *transport for the finite real packet, inhabitedness row, comparison ledger, fold ledger, upper-bound row, least-bound row, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a finite-supremum request through R, I, O, M, U, L, H.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteRealSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{RealMinMax}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteRealSupremum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host maximum operator, quotient-real equality, detached order trichotomy, selected limit outside the finite packet, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteRealSupremumUp.FiniteRealSupremumTasteGate single carrier alignment`

Theorem 5951.2 (Finite real supremum NameCert obligations)**.** *For every accepted* $\mathsf{FiniteRealSupremum}^{\uparrow}$ *carrier*

$$S = (R, I, O, M, U, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteRealSupremum}^{\uparrow}}$ *surface consists exactly of carrier admission for the nonempty finite* $\mathsf{Real}^{\uparrow}$ *packet R, inhabitedness exposure through I, finite* $\mathsf{TotalOrder}^{\uparrow}$ *comparison-ledger access through O,* $\mathsf{RealMinMax}^{\uparrow}$ *fold selection through M, selected upper-bound verification through U, least-bound verification through L, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5951.1. The row R gives the finite list of real-name rows, and I keeps the packet nonempty before any upper-bound request is considered.

The comparison and selection rows are read next. The ledger O exposes only displayed $\mathsf{TotalOrder}^{\uparrow}$ comparison witnesses over the retained $\mathsf{Real}^{\uparrow}$ rows, while M records the finite $\mathsf{RealMinMax}^{\uparrow}$ fold that selects the candidate read by the upper-bound row.

The bounding rows are downstream of that fold. The row U checks that every retained real row is below the selected candidate, and L checks that any displayed upper bound receives a comparison witness from the selected candidate.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no row carries host maximum, quotient-real equality, detached trichotomy, countable choice, open-cover compactness, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{FiniteRealSupremum}^{\uparrow}$ consumer that reads a supremum row without first displaying the nonempty finite $\mathsf{Real}^{\uparrow}$ packet, comparison ledger, selected upper-bound row, least-bound verification row, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: real, rat, total-order, real-min-max, finitenetminimumfold: no listed sibling is bijective with $\mathsf{FiniteRealSupremum}^{\uparrow}$ because this packet jointly carries a nonempty finite RealUp family, a finite comparison ledger, an upper-bound candidate, a least-bound verification row, componentwise transport, replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteRealSupremum}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteRealSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5951.1 and Definition 5951.2 seed the finite supremum route through nonempty Real rows, comparison ledgers, selected upper-bound rows, least-bound verification, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteRealSupremum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host maximum, quotient Real equality, detached order trichotomy, selected limits outside the finite packet, countable choice, open-cover compactness, Lean verification, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, nonempty finite Real packet exposure, comparison-ledger exactness, selected upper-bound verification, least-bound verification, and TasteGate compatibility for BEDC.Derived.FiniteRealSupremu

Constructive story: A FiniteRealSupremum$^{\uparrow}$ packet is generated from a nonempty finite RealUp packet, finite TotalOrderUp comparison rows, a RealMinMaxUp fold ledger, an upper-bound row, a least-bound verification row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 5952

A Concrete Naming Certificate for $\mathsf{HahnDecomposition}^{\uparrow}$

Following Hahn and standard signed-measure theory, the $\mathsf{HahnDecomposition}^{\uparrow}$ packet records the finite BEDC surface by which a signed measure is read through positive and negative regions before downstream integration or Radon representation consumers inspect it. It sits after the measure carrier of chapter 92, the regular measure packet of chapter 2263, the compact representation route of chapter 3504, the public integral row of chapter 123, and the terminal real seal of chapter 38. The chapter names a decomposition certificate; it does not import a host sigma-algebra completion, quotient equality of signed measures, or a choice-selected measurable partition.

Definition 5952.1 (Hahn decomposition carrier)**.** *A* $\mathsf{HahnDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, A, B, Z, M, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the signed-measure row, carrying the finite positive and negative variation ledgers visible to this certificate.*
2. *A is the positive-region row, naming the measurable region on which every displayed signed-measure read is nonnegative.*
3. *B is the negative-region row, naming the measurable region on which every displayed signed-measure read is nonpositive.*
4. *Z is the null-boundary row, recording the displayed overlap and complement boundary on which the decomposition exports no signed mass.*
5. *M is the* $\mathsf{Measure}^{\uparrow}$ *readback row for the underlying measurable-event and finite-value surface.*
6. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *regularity row used when a downstream regular-measure consumer asks for compact or open approximants.*
7. *L is the* $\mathsf{RieszMarkov}^{\uparrow}$ *representation row used when a compact positive-functional consumer reads the decomposed measure through a representing Radon packet.*
8. *E is the* $\mathsf{Integral}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for integral values and terminal real seals produced from the displayed signed-measure side.*
9. *H records componentwise* hsame *transport for S, A, B, Z, M, R, L, E and for every region, boundary, measure value, integral value, and real seal named by those rows.*

10. *C is the displayed* Cont *replay ledger routing signed-measure reads first through A, B, Z, then through M, R, L, E when a downstream consumer asks for measure, Radon, Riesz–Markov, integral, or real output.*

11. *P is the* Pkg *provenance over* $\mathsf{HahnDecomposition}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{RieszMarkov}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and local certificate rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{HahnDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only by these twelve displayed rows. It has no coordinate for host signed-measure equality, a completed measure algebra, a selected measurable partition, countable choice, quotient equality of regions, or ambient standard-model measure theory.

Lean4 (definition) `BEDC.Derived.HahnDecompositionUp.HahnDecompositionUp`
Lean4 ✓ `BEDC.Derived.HahnDecompositionUp.HahnDecompositionTasteGate single carrier alignment`

Theorem 5952.2 (Hahn decomposition NameCert obligations)**.** *For every accepted* $\mathsf{HahnDecomposition}^{\uparrow}$ *carrier*

$$D = (S, A, B, Z, M, R, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HahnDecomposition}^{\uparrow}}$ *surface consists exactly of signed-measure admission through S, positive-region visibility through A, negative-region visibility through B, null-boundary exactness through Z, measure readback through M, Radon regularity handoff through R, Riesz–Markov representation handoff through L, integral and real readback through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5952.1. It lists the signed-measure row, positive and negative region rows, null-boundary row, downstream measure-analysis rows, transport row, replay row, provenance row, and local naming row.

The decomposition obligations are the visible rows A, B, Z over the signed-measure row S. They state which region is read as positive, which region is read as negative, and where the null boundary prevents a consumer from extracting extra signed mass.

The downstream obligations are the rows M, R, L, E. They force every measure, RadonMeasure, RieszMarkov, Integral, or Real readback to pass through displayed BEDC packets rather than through a host theorem about signed measures.

Finally, H, C, P, N preserve and name the same finite route. Since no other coordinate is present, the local NameCert surface is exhausted by the listed rows. □

Theorem 5952.3 (Hahn decomposition signed-measure route)**.** *Every downstream consumer of an accepted* $\mathsf{HahnDecomposition}^{\uparrow}$ *carrier must read the signed-measure row S through the positive region A, negative region B, and null boundary Z before it may inspect the* $\mathsf{Measure}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{RieszMarkov}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *rows M, R, L, E. The route is transported by H, replayed by C, packaged by P, and named by N.*

Proof. Begin with Definition 5952.1. The only signed-measure coordinate is S, and its decomposition-facing rows are exactly A, B, Z.

Apply Definition 5952.2. The obligation surface requires positive-region visibility, negative-region visibility, and null-boundary exactness before any downstream readback row can be consumed.

Now read the downstream rows in order. Measure consumers pass through M, regular-measure consumers pass through R, compact representation consumers pass through L, and integral or

terminal real consumers pass through E. The replay ledger C records those routes, while H, P, N transport, source, and name them.

Thus the signed-measure route is the displayed finite BEDC packet. It exports no host signed-measure equality, selected measurable partition, completed measure algebra, quotient equality of regions, countable choice, or ambient standard-model measure theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{HahnDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{HahnDecomposition}^{\uparrow}$ packet is generated from a signed-measure row, positive-region row, negative-region row, null-boundary row, MeasureUp readback, RadonMeasureUp regularity handoff, RieszMarkovUp representation handoff, IntegralUp and RealUp readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HahnDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5952.1, Definition 5952.2, and Definition 5952.3 seed the finite signed-measure decomposition route through positive, negative, null-boundary, measure, Radon, Riesz–Markov, integral, and real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HahnDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host signed-measure equality, selected measurable partitions, completed measure algebras, quotient equality of regions, countable choice, or ambient standard-model measure theory.

Upgrade path: Obligation closure requires checked carrier admission, signed-measure classifier transport, positive-region visibility, negative-region visibility, null-boundary exactness, downstream measure readback, Radon and Riesz–Markov handoff, Integral and Real readback, transport stability, replay stability, and provenance for BEDC.Derived.HahnDecompositionUp.

Chapter 5953

A Concrete Naming Certificate for $\mathsf{ConstructivePolishSpace}^{\uparrow}$

Following Bishop and Bridges on complete separable metric spaces in constructive analysis, the $\mathsf{ConstructivePolishSpace}^{\uparrow}$ packet records the L10 completion frontier as finite NameCert data. It sits after the completion source of chapter 1506, the separated classifier of chapter 1411, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, the effective separable surface of chapter 1706, and the compact subcase of chapter 4016. The packet names a completion-facing constructive Polish carrier without claiming quotient completions, countable choice, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5953.1 (ConstructivePolishSpace carrier)**.** *A* $\mathsf{ConstructivePolishSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, S, D, W, Q, R, L, H, C, G, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *source row, exposing only the completion request, finite tolerance ledger, and completion-facing handoff consumed by the packet.*
2. *S is the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row, requiring zero-distance agreement for displayed points to pass through the separated metric boundary.*
3. *D is the dense-sequence row. It names the finite prefix and readback schedule inspected by a completion consumer, not an infinite choice of representatives.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for dense-name observations, Cauchy windows, and requested precision budgets.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational approximants, tolerance comparisons, and Cauchy witness reads.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after the finite stream windows and regular rational readbacks have been displayed.*
7. *L is the located completion service row, answering each displayed approximation request through M, S, D, W, Q, R.*
8. *H records componentwise* hsame *transport for the metric-completion source, separated classifier, dense schedule, stream windows, regular rational readback, Real seal, located service, replay row, provenance row, and local naming row.*

9. *C records displayed* Cont *replay from a finite completion request through W, Q, R and then back to M, S, D, L.*
10. *G is the* Pkg *provenance over* ConstructivePolishSpace$^{\uparrow}$, MetricCompletion$^{\uparrow}$, SeparatedMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, EffectivePolishSpace$^{\uparrow}$, CompactPolishSpace$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{ConstructivePolishSpace}^{\uparrow}}$ *row.*

The carrier exports only this displayed finite completion route. It has no coordinate for quotient completion classes, countable choice over dense points, host point equality, selected representatives, or an unrestricted completeness principle for Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.ConstructivePolishSpaceUp.ConstructivePolishSpaceTasteGate single carrier alignment`

Theorem 5953.2 (ConstructivePolishSpace NameCert obligation surface)**.** *For every accepted* ConstructivePolishSpace$^{\uparrow}$ *carrier $P = (M, S, D, W, Q, R, L, H, C, G, N)$, the local* NameCert$_{\text{ConstructivePolishSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, separated zero-distance classification, dense-prefix readback, finite StreamName window admission, RegSeqRat regular readback, Real seal handoff, located completion service replay, componentwise transport, provenance, and local naming through the displayed rows $M, S, D, W, Q, R, L, H, C, G, N$.*

Proof. Project the carrier of Definition 5953.1. The tuple displays exactly the metric-completion source, separated classifier, dense schedule, StreamName windows, RegSeqRat readback, Real seal, located completion service, transport, replay, provenance, and local naming rows.

A completion-facing request must first choose a finite window in W and a finite dense-prefix read through D. The regular rational row Q turns those observations into tolerance data before the terminal Real seal R can be inspected.

The separated classifier S is the only row that compares displayed points, and it does so over the completion source M. The located service L, transport row H, replay row C, provenance row G, and local name row N package this same route, so no excluded quotient, choice, host-equality, or ambient-completeness coordinate is available. □

Theorem 5953.3 (ConstructivePolishSpace completion readback route)**.** *Every completion-facing read served by an accepted* ConstructivePolishSpace$^{\uparrow}$ *packet factors through the finite route*

$$D, W \longmapsto Q \longmapsto R \longmapsto M, S, L \longmapsto H, C, G, N.$$

Thus an EffectivePolishSpace$^{\uparrow}$ *or* CompactPolishSpace$^{\uparrow}$ *consumer may read dense schedules, finite windows, regular rational approximants, the terminal Real seal, and the located completion service, but it cannot extract quotient completion equality, a countable-choice selector, host point equality, or ambient* Real$^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing consumer. The only dense-name data in the packet is D, and the only observation schedule is W, so the consumer first receives a finite dense prefix and finite stream windows.

The readback then passes through Q and R. This exposes regular rational approximants and a Real seal only after the finite observations have been displayed.

Finally the route returns to the completion source M, separated classifier S, and located service L, then through the structural rows H, C, G, N. Since no other coordinate appears in the carrier, the listed quotient, choice, host-equality, and ambient-completeness claims cannot be read from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructivePolishSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructivePolishSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5953.1, Definition 5953.2, and Definition 5953.3 seed the finite constructive Polish completion route over metric completion, separated classification, dense schedules, StreamName windows, RegSeqRat readback, Real sealing, located completion service, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructivePolishSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, countable choice, host point equality, selected representatives, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, dense-prefix readback, separated classifier stability, completion-readback routing, nonescape, and local naming for BEDC.Derived.ConstructivePolishSpaceUp.

Constructive story: A $\mathsf{ConstructivePolishSpace}^{\uparrow}$ packet is generated from a MetricCompletion source row, a SeparatedMetric classifier row, finite dense schedules, StreamName windows, RegSeqRat readback, a RealUp seal, located completion service, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5954

A Concrete Naming Certificate for $\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}$

Following Bishop, Bridges, and Munkres on compact metric spaces and equicontinuous families, the $\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}$ packet records the finite route from compact nets and family moduli to a stable uniform-modulus ledger. It refines the compact-uniform-continuity vision through the compact source of chapter 129, the continuous family surface of chapter 130, the uniform-modulus handoff of chapter 369, the equicontinuity carrier of chapter 4076, the rational fold interface of chapter 37, and the metric source and target rows of chapter 4230. The packet names only finite compact nets, displayed family moduli, rational lower-bound folds, metric-distance rows, transport, replay, provenance, and local naming; it does not import open-cover compactness, selected subcovers, function extensionality in the host, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5954.1 (CompactUniformEquicontinuityStability carrier)**.** *A* $\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, A, Q, U, M, T, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, naming the inspected net points, coverage radii, and located compact source data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, exposing only the displayed finite family observations needed by the compact-net argument.*
3. *A is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family-modulus row, assigning one displayed local modulus route to the family observations in F.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *finite lower-bound fold row, reducing the finitely many compact-net and family-radius budgets to a positive rational tolerance.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *export row consumed by compact-uniform-continuity, Arzela–Ascoli, and family-modulus consumers.*
6. *M is the source* $\mathsf{Met}^{\uparrow}$ *distance row for the compact domain.*
7. *T is the target* $\mathsf{Met}^{\uparrow}$ *distance row for the displayed codomain observations.*

8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row for the source and target distance reads after the rational fold has been displayed.*
9. *H records componentwise* hsame *transport for compact nets, continuous family rows, equicontinuity moduli, rational folds, uniform-modulus export, source and target distances, Real sealing, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from finite compact-net reads through the family-modulus row and rational lower-bound fold to the uniform-modulus export.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}}$ *row.*

The carrier exports family-modulus stability only through these rows. It has no coordinate for open-cover compactness, selected finite subcovers, unrestricted host function equality, countable choice, quotient maps, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactUniformEquicontinuityStabilityUp.CompactUniformEquicontinuityStabilityTaste`
`single carrier alignment`

Theorem 5954.2 (CompactUniformEquicontinuityStability handoff)**.** *For every accepted* CompactUniformEquicon *carrier $E = (K, F, A, Q, U, M, T, R, H, C, P, N)$, every compact-uniform family-stability read factors through*

$$K, F, A, M, T \longmapsto Q \longmapsto R, U \longmapsto H, C, P, N.$$

Consequently a downstream Arzela–Ascoli or compact family-modulus consumer receives a displayed uniform-modulus row built from finite compact nets, continuous family observations, equicontinuity moduli, rational lower-bound folds, metric distances, Real sealing, transport, replay, provenance, and local naming, with no access to open-cover compactness, selected subcovers, countable choice, or host function equality.

Proof. Project the carrier of Definition 5954.1. The compact source is the finite-net row K, the family observations are F, and the only family-modulus coordinate is A.

The source and target metric reads pass through M and T. The rational fold Q is the unique row that combines the finitely many net and modulus budgets into a positive tolerance before any Real seal or uniform-modulus export can be read.

After Q is displayed, R seals the finite metric reads and U exports the uniform-modulus handoff. The structural rows H, C, P, N transport, replay, source, and name the same route, so the excluded open-cover, selector, choice, quotient, host-equality, and ambient-completeness data have no row from which to be consumed. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}}$ ·

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5954.1 and Definition 5954.2 seed the finite family-modulus stability route over CompactMetric finite nets, ContinuousMap family rows, Equicontinuity moduli, RatUp lower-bound folds, UniformModulus export, MetricSpace source and target distances, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformEquicontinuityStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, selected subcovers, countable choice, host function equality, quotient maps, standard bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, family-modulus stability, rational fold positivity, uniform-modulus handoff, metric distance sealing, nonescape, and local naming for BEDC.Derived.CompactUniformEqui

Constructive story: A CompactUniformEquicontinuityStability$^{\uparrow}$ packet is generated from CompactMetric finite nets, ContinuousMap family rows, an Equicontinuity family modulus, a RatUp finite lower-bound fold, a UniformModulus export, MetricSpace source and target distance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5955

A Concrete Naming Certificate for $\mathsf{RealEqualityReflection}^{\uparrow}$

Following Bishop and Bridges' constructive analysis, equality of real numbers is presented through regular Cauchy data and explicit tolerance control rather than through an ambient quotient oracle. The $\mathsf{RealEqualityReflection}^{\uparrow}$ packet names the finite equality-reflection boundary where the setoid route of chapter 1820 is exposed to the equality-facing consumers of chapter 1636. It records the route from regular rational streams and dyadic tolerance rows into the $\mathsf{Real}^{\uparrow}$ seal without adding selected representatives, quotient equality as a primitive, countable choice, host real extensionality, Lean verification, or bridge checking.

Definition 5955.1 (Real equality reflection carrier)**.** *A* $\mathsf{RealEqualityReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (Q, E, R, S, D, H, C, P_0, N)$$

with the following displayed rows:

1. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route that reads equality through finite Cauchy windows.*
2. E *is the* $\mathsf{RealEquality}^{\uparrow}$ *consumer row whose equality-facing request is admitted only after* Q *is visible.*
3. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row used by the setoid route.*
4. S *is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened before any equality consumer reads the source.*
5. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the visible windows and regular rational readback.*
6. H *records componentwise* hsame *transport for the setoid route, equality consumer, source, windows, tolerance ledger, replay, provenance, and local naming rows.*
7. C *records displayed* Cont *replay from an equality-reflection request through* R, S, D, Q, E *and then through the structural rows.*
8. P_0 *is the* Pkg *provenance over* $\mathsf{RealEqualityReflection}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RealEqualityReflection}^{\uparrow}}$ *row.*

The carrier is admitted only when every $\mathsf{RealEquality}^{\uparrow}$ *read is downstream of the displayed* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route and every route into the* $\mathsf{Real}^{\uparrow}$ *seal still exposes the regular source, opened window, dyadic tolerance row, transport, replay, provenance, and local name.*

Theorem 5955.2 (Real equality reflection NameCert obligation surface)**.** *For every accepted* $\mathsf{RealEqualityReflection}^{\uparrow}$ *carrier*

$$P = (Q, E, R, S, D, H, C, P_0, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealEqualityReflection}^{\uparrow}}$ *surface consists exactly of carrier admission, setoid-route exposure, real-classifier reflection, transport, replay, provenance, and non-escape rows. Carrier admission displays* $Q, E, R, S, D, H, C, P_0, N$*; setoid-route exposure requires* Q *to read* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *through* R, S, D*; real-classifier reflection requires the equality-facing consumer row* E *to remain downstream of that setoid route before any* $\mathsf{Real}^{\uparrow}$ *seal is read; transport, replay, and provenance are carried only by* H, C, P_0*; non-escape is the assertion that no quotient-equality oracle, selected representative, countable-choice schedule, ambient real extensionality, Lean verification, or bridge coordinate is present.*

Lean4 ✓ `BEDC.Derived.RealEqualityReflectionUp.RealEqualityReflectionNameCertObligationSurface`

Proof. Project the carrier of Definition 5955.1. The projection exposes one setoid route, one equality-facing consumer row, one regular rational source, one opened window row, one dyadic tolerance ledger, and the structural transport, replay, provenance, and naming rows.

The setoid route is forced to pass through R, S, D before the consumer row E can read equality. Thus the packet first records the regular rational readback and finite StreamName inspection used by $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$.

The equality-facing row then reflects only that visible route into the Real boundary. Since the only structural rows are H, C, P_0, N, the packet can transport, replay, source, and name the displayed route but cannot add a quotient oracle, selected representative, choice schedule, ambient extensionality, Lean proof, or bridge datum. □

Theorem 5955.3 (Real equality reflection Cauchy setoid route)**.** *Every* $\mathsf{RealEquality}^{\uparrow}$ *consumer exported by an accepted* $\mathsf{RealEqualityReflection}^{\uparrow}$ *carrier factors through the finite route*

$$R, S, D \longmapsto Q \longmapsto E \longmapsto \mathsf{Real}^{\uparrow}.$$

In particular, the consumer first reads the $\mathsf{RegSeqRat}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, then exposes the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *route, and only after that may it reach the equality-facing* $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Begin with an equality-facing request. Carrier admission gives the displayed rows R, S, D, Q, E and no direct coordinate from the request to a Real seal.

Read R, S, D first. These rows determine the regular source readback, finite window, and dyadic tolerance surface on which the Cauchy setoid comparison is visible.

Now read the setoid row Q and only then the consumer row E. The target equality consumer therefore sees $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ before $\mathsf{RealEquality}^{\uparrow}$ reaches the $\mathsf{Real}^{\uparrow}$ seal, so the route has no hidden quotient equality, selected representative, countable choice, ambient completeness, Lean verification, or bridge checking coordinate. □

Falsifiable prediction: A $\mathsf{RealEquality}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ equality-facing seal from accepted $\mathsf{RealEqualityReflection}^{\uparrow}$ data without first exposing the $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ route refutes the carrier surface.

Independence witness: cauchy-equivalence-setoid, real-equality, regseqrat, streamname, dyadicratcore, real: no listed sibling supplies the whole nine-row packet $P = (Q, E, R, S, D, H, C, P_0, N)$, because each sibling contributes only setoid, equality-consumer, source, window, tolerance, or seal data while this chapter binds them with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealEqualityReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{RealEqualityReflection}^{\uparrow}$ packet is generated from a CauchyEquivalenceSetoid route, a RealEquality consumer row, a RegSeqRat source row, StreamName windows, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row before the Real seal is read.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealEqualityReflection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5955.1, Definition 5955.2, and Definition 5955.3 seed the finite equality-reflection surface over regular rational sources, finite windows, dyadic tolerances, Cauchy setoid routing, equality-facing consumers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealEqualityReflection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient equality as primitive, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, setoid-route exposure, real-classifier reflection, transport, replay, provenance, and non-escape for BEDC.Derived.RealEqualityReflectionUp.

Chapter 5956

A Concrete Naming Certificate for $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$

Following Sobolev and Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$ packet records finite energy evidence for a displayed function row and its weak-derivative rows without passing through a host quotient space. It sits after the Sobolev comparison surface of chapter 440, the Sobolev-space carrier of chapter 5911, the Hilbert inner-product route of chapter 91, and the terminal real readback seal of chapter 38. The packet gives PDE, trace, and embedding consumers a finite energy certificate surface rather than a completed function space.

Definition 5956.1 (Constructive Sobolev energy carrier)**.** *A* $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (F, D, E, A, B, H, C, P, N)$$

with the following displayed rows:

1. *F is the function row. It records the visible domain name, target* $\mathsf{Real}^{\uparrow}$ *readback route, finite test-window requests, and the stream or regular-rational windows through which function values are inspected.*
2. *D is the weak-derivative row. It records the finite multi-index list, test-window signs, integration-by-parts witnesses, and derivative readback windows imported from the weak-derivative and Sobolev siblings.*
3. *E is the energy norm row. It records the displayed L^2-style summands, Hilbert inner-product requests, rational upper bounds, and real seal endpoints used to certify finite energy.*
4. *A is the dyadic approximation row. It records finite dyadic tolerances, local mesh windows, rational error bounds, and replayable approximation requests for F, D, E.*
5. *B is the boundary and trace row. It records only the trace windows, boundary test rows, and compatibility checks explicitly present in the packet.*
6. *H records componentwise* hsame *transport across F, D, E, A, B.*
7. *C records displayed* Cont *replay from a Sobolev-energy request through the function row, weak-derivative rows, energy norm row, dyadic approximation row, boundary and trace row, and transport row.*
8. *P is the* Pkg *provenance row over* $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ConstructiveSobolevEnergy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. No coordinate stores a host quotient of functions, a measure-completed representative class, a Hahn–Banach separator, an open-cover compactness theorem, a selected trace representative, countable choice, or ambient equality of functions.

Lean4 ✓ `BEDC.Derived.ConstructiveSobolevEnergyUp.ConstructiveSobolevEnergyTasteGate single carrier alignment`

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveSobolevEnergy}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{ConstructiveSobolevEnergy}^{\uparrow}$ packet is generated from a function row, weak-derivative rows, an energy norm row, dyadic approximation data, boundary and trace data, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveSobolevEnergy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5956.1 binds the finite carrier surface for Sobolev energy certificates over displayed function, derivative, approximation, boundary, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveSobolevEnergy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host function quotients, measure-completion claims, Hahn–Banach separation, open-cover compactness, choice, or bridge checking.

Upgrade path: Obligation closure requires checking carrier admission, energy-row exactness, derivative-row replay, trace-window non-escape, and local NameCert stability for the displayed packet.

Chapter 5957

A Concrete Naming Certificate for $\mathsf{GammaConvergence}^{\uparrow}$

Following Dal Maso on Gamma-convergence, the $\mathsf{GammaConvergence}^{\uparrow}$ packet records the finite variational-convergence surface by which lower-bound and recovery-window data are routed to compactness and real-valued consumers. It sits after the metric carrier of chapter 57, the compact finite-read discipline of chapter 129, the lower-semicontinuity boundary of chapter 1536, the regular rational readback route of chapter 301, and the terminal real seal of chapter 38. The packet names only displayed liminf ledgers, recovery windows, compactness windows, real-value readbacks, transport, replay, provenance, and local naming; it does not assert host minimizer existence, quotient equality of functionals, selected recovery sequences, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5957.1 (Gamma-convergence carrier)**.** *A* $\mathsf{GammaConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, K, L, F, A, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row on which the variational data are inspected.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite compactness-window row used by subsequence-facing consumers.*
3. *L is the* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *liminf-boundary row for the inspected functionals.*
4. *F is the finite functional-index and value-window ledger.*
5. *A is the liminf-test row, recording every displayed approach window and its lower-bound comparison.*
6. *R is the recovery-window row, recording finite witnesses that a downstream consumer may replay without selecting a host recovery sequence.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational value windows exposed by A and R.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after X, K, L, F, A, R, Q have been displayed.*
9. *H records componentwise* hsame *transport for the displayed rows.*
10. *C records displayed* Cont *replay from metric and compact windows to the lower-bound, recovery, readback, and seal rows.*

11. *P is the* Pkg *provenance over* $\mathsf{GammaConvergence}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{GammaConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, K, L, F, A, R, Q, E, H, C, P, N$. It has no coordinate for quotient equality of functionals, selected minimizers, selected recovery sequences, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.GammaConvergenceUp.GammaConvergenceTasteGate single carrier alignment`

Theorem 5957.2 (Gamma-convergence NameCert obligations)**.** *For an accepted* $\mathsf{GammaConvergence}^{\uparrow}$ *carrier $G = (X, K, L, F, A, R, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GammaConvergence}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure through X, compact finite-window exposure through K, lower-semicontinuity exposure through L, finite functional-window exposure through F, liminf-test exposure through A, recovery-window exposure through R, regular rational readback through Q, terminal real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Every variational consumer of the packet must therefore read the finite route

$$\mathsf{Met}^{\uparrow},\ \mathsf{CompactMetric}^{\uparrow},\ \mathsf{LowerSemicontinuous}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}$$

before it may inspect a Gamma-limit-facing value. The exported route is a finite NameCert boundary for liminf and recovery data, not a host compactness theorem, a selected minimizer, quotient equality of functionals, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

The liminf and recovery rows of $\mathsf{GammaConvergence}^{\uparrow}$ consume the Mosco finite variational surface of chapter 5938 before Gamma-facing compactness and recovery consumers read the local packet: the lower-semicontinuity row, weak-liminf ledger, recovery-window row, convex support, completion handoff, and Real readback are the shared variational boundary.

Proof. Project the carrier of Definition 5957.1. The rows X, K, L, F, A, R, Q, E are the only coordinates from which a variational consumer may read metric source data, compact windows, lower-semicontinuity tests, finite functional windows, liminf checks, recovery windows, rational readbacks, and terminal real seals.

The liminf side is generated by A only after X, K, L, F have been displayed. Thus a lower-bound read cannot bypass the metric row, compact finite-window row, lower-semicontinuity row, or finite value-window ledger.

The recovery side is generated by R and read back through Q before E is visible. Since H, C, P, N preserve only the displayed rows under transport, replay, provenance, and local naming, they cannot add a selected recovery sequence, a minimizer, quotient equality, or an ambient completeness theorem.

Consequently the local NameCert obligations are exhausted by the listed finite rows. Any stronger variational principle must be supplied by a later consumer, not by the $\mathsf{GammaConvergence}^{\uparrow}$ packet itself. □

Falsifiable prediction: Every accepted $\mathsf{GammaConvergence}^{\uparrow}$ consumer that reaches a Real-facing Gamma-limit value without first reading a Metric row, CompactMetric row, LowerSemicontinuous row, liminf-test row, recovery-window row, RegSeqRat readback row, transport, replay, provenance, and local NameCert row refutes the carrier surface.

Independence witness: metric, compactmetric, lowersemicontinuous, regseqrat, real, mosco-convergence: $\mathsf{GammaConvergence}^{\uparrow}$ is not bijective to any listed sibling because the packet jointly stores liminf tests, recovery windows, compact finite-read boundaries, rational value readbacks, terminal real seals, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GammaConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GammaConvergence}^{\uparrow}}, \mathsf{seed})$

Constructive story: A $\mathsf{GammaConvergence}^{\uparrow}$ packet is generated from a metric source row, compact finite-window row, lower-semicontinuity row, finite functional windows, liminf-test rows, recovery-window rows, RegSeqRat readback, a RealUp seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Scope closed: Definition 5957.1 and Definition 5957.2 seed the finite variational-convergence certificate over Metric, CompactMetric, LowerSemicontinuous, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GammaConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host minimizer existence, selected recovery sequences, quotient equality of functionals, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Formalization requires a BEDC.Derived.GammaConvergenceUp carrier together with checked liminf, recovery-window, compactness-window, readback, transport, provenance, and local naming obligations.

Chapter 5958

A Concrete Naming Certificate for $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ packet records the finite choice-sequence surface by which located real observations are read as scheduled rational windows rather than as selected completed representatives. It sits downstream of the located-real carrier of **??**, the finite schedule discipline of chapter 126, the regular rational readback route of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the comparison boundary of chapter 1381. The packet names a finite L10 choice-sequence interface only; it does not import countable choice, quotient streams, selected representatives, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5958.1 (LocatedRealChoiceSequence carrier)**.** *A* $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (L, S, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *source row, naming the located real endpoint or local located-real family inspected by the choice-sequence request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, listing the finite indices, stages, and observation windows read before the request is accepted.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying the dyadic radii and residue checks attached to the windows in S.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, packaging the scheduled rational observations as regular Cauchy data with the displayed tolerance schedule.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after L, S, D, R have been displayed.*
6. *B is the* $\mathsf{BishopRealComparison}^{\uparrow}$ *boundary row, recording any finite comparison or overlap evidence used to keep the located-real windows compatible with the regular rational readback.*
7. *H records componentwise* hsame *transport for located-real rows, schedules, dyadic tolerances, regular readbacks, Real seals, comparison rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a choice-sequence request through L, S, D, R, E, B, H.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BishopRealComparison}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{LocatedRealChoiceSequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for a countable choice functional, quotient stream equality, a selected representative of a real, trichotomy of located real comparisons, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedRealChoiceSequenceUp.LocatedRealChoiceSequenceTasteGate single carrier alignment`

Theorem 5958.2 (LocatedRealChoiceSequence NameCert obligations)**.** *For every accepted* LocatedRealChoiceSequ *carrier*

$$Q = (L, S, D, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealChoiceSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the located-real source L, finite* $\mathsf{StreamName}^{\uparrow}$ *schedule exposure through S, dyadic tolerance access through D, regular rational readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission through E, Bishop comparison-boundary access through B, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5958.1. The located source is L, and the only schedule available to a consumer is the finite $\mathsf{StreamName}^{\uparrow}$ row S.

The rational data are downstream of that schedule. The dyadic tolerance ledger D attaches radii and residue checks to the same inspected windows, and R repacks those scheduled observations as a $\mathsf{RegSeqRat}^{\uparrow}$ readback.

The real-facing and comparison rows are then forced. The seal E is reached only after L, S, D, R have been displayed, while B records the finite $\mathsf{BishopRealComparison}^{\uparrow}$ boundary data used to compare or overlap the displayed windows.

Finally H, C, P, N transport, replay, source, and name exactly the same finite packet. Since the carrier has no row for countable choice, quotient streams, selected representatives, trichotomy, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 5958.3 (LocatedRealChoiceSequence regular-window handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{LocatedReal}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BishopRealComparison}^{\uparrow}.$$

Thus a downstream real or comparison consumer receives only the displayed located-real row, the finite schedule windows, the dyadic tolerance ledger, the regular rational readback, the Real seal, the Bishop comparison boundary, and the structural rows H, C, P, N.

Proof. Read the source and schedule rows first. The located-real row L supplies the inspected endpoint or local family, and the finite schedule row S names exactly the stages and windows available to the request.

The regular readback cannot bypass the tolerance ledger. The row D stores the dyadic budgets attached to those windows, and R is the only $\mathsf{RegSeqRat}^{\uparrow}$ row in the carrier, so every regular rational read factors through D and then through R.

The downstream rows preserve this order. The $\mathsf{Real}^{\uparrow}$ seal E and the $\mathsf{BishopRealComparison}^{\uparrow}$ boundary B are listed after the scheduled readback, and the structural rows H, C, P, N transport, replay, source, and name that same route. No refused object is a coordinate of the carrier, so no consumer can read a quotient stream, selected representative, trichotomy row, countable-choice functional, or ambient completeness witness through this handoff. □

Falsifiable prediction: An accepted $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ consumer that reads a regular rational or Real-facing choice-sequence row without first displaying the located-real source, finite stream schedule, dyadic tolerance ledger, regular readback, Real seal, Bishop comparison boundary, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: located-real, streamname, regseqrat, dyadicratcore, real, bishop-real-comparison: $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a located-real source, finite StreamName schedule rows, DyadicRatCore tolerance rows, RegSeqRat readbacks, Real seals, Bishop comparison boundaries, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealChoiceSequence}^{\uparrow}$ packet is generated from a located-real source, finite StreamName schedule rows, DyadicRatCore tolerance rows, RegSeqRat readbacks, RealUp seal rows, BishopRealComparison boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedRealChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5958.1, Definition 5958.2, and Definition 5958.3 seed the finite located-real choice-sequence route over schedules, tolerances, regular readbacks, Real seals, comparison boundaries, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedRealChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim countable choice, quotient streams, selected representatives, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule exposure, dyadic tolerance exactness, RegSeqRat readback handoff, Real seal nonescape, Bishop comparison-boundary stability, and structural transport for BEDC.Derived.LocatedRealChoiceSequenceUp.

Chapter 5959

A Concrete Naming Certificate for $\mathsf{NestedIntervalCauchySeal}^{\uparrow}$

Following Bishop and Bridges constructive nested-interval practice, the $\mathsf{NestedIntervalCauchySeal}^{\uparrow}$ packet records the finite seal that turns nested dyadic interval data into a regular Cauchy rational handoff before any completion consumer may read a $\mathsf{Real}^{\uparrow}$ name. It sits beside the dyadic rational core of chapter 291, the finite window service of chapter 126, the regular rational readback surface of chapter 301, the terminal real-name seal of chapter 38, and the located interval sibling of chapter 394. The packet names only finite nested located intervals, dyadic diameter budgets, stream windows, regular rational readback, Real handoff, transport, replay, provenance, and local naming; it does not invoke interval intersection, quotient real equality, countable choice, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5959.1 (NestedIntervalCauchySeal carrier)**.** *A* $\mathsf{NestedIntervalCauchySeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *nested interval ledger, listing the retained interval cells, endpoint windows, and containment witnesses.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *diameter-budget row, carrying the dyadic tolerance requested by the Cauchy read and the displayed diameter bound for each retained cell.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which interval endpoints, containment witnesses, and dyadic tolerances are inspected.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows and diameter budgets.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after I, D, S, R have been displayed.*
6. *H records componentwise* hsame *transport for located intervals, dyadic budgets, stream windows, regular readback, Real sealing, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a Real-facing nested-interval request through I, D, S, R, E, H.*
8. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalCauchySeal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchySeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, D, S, R, E, H, C, P, N$. It has no coordinate for host interval intersection, quotient real equality, a selected limiting point, countable choice, detached completion data, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.NestedIntervalCauchySealUp.NestedIntervalCauchySealTasteGate single carrier alignment`

Theorem 5959.2 (NestedIntervalCauchySeal NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalCauchySeal}$ *carrier $Z = (I, D, S, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchySeal}^{\uparrow}}$ *surface is exhausted by the following five rows:*

1. *located nested-interval admission through I, including endpoint windows and containment witnesses from* $\mathsf{LocatedInterval}^{\uparrow}$*;*
2. *dyadic diameter-budget admission through D, read only as* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance data;*
3. *finite observation admission through S and regular Cauchy readback through R, tying the* $\mathsf{StreamName}^{\uparrow}$ *windows to the* $\mathsf{RegSeqRat}^{\uparrow}$ *route;*
4. *terminal handoff through E, exposing a* $\mathsf{Real}^{\uparrow}$ *seal only after I, D, S, R have been displayed;*
5. *structural preservation through H, C, P, N, namely componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *naming.*

Consequently every Real or Completion consumer of the seal receives a regular Cauchy $\mathsf{Real}^{\uparrow}$ *handoff through finite nested dyadic interval data, and receives no interval-intersection theorem, quotient real equality, selected limit, countable choice, host equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 5959.1. The nested source row is I, and its only precision ledger is the dyadic row D.

The observation route is finite. The $\mathsf{StreamName}^{\uparrow}$ row S selects the endpoint and containment windows, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R reads the same windows as a regular rational sequence surface.

The Real-facing route starts only after those rows have been exposed. The seal row E is therefore a handoff from finite nested dyadic interval data to the regular Cauchy surface, not a proof of host interval intersection or ambient completeness.

Finally H, C, P, N transport, replay, source, and name exactly the displayed packet. Since the carrier has no row for quotient equality, a selected limiting point, countable choice, detached completion data, or host equality, the five-row NameCert surface is exhausted by the rows listed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalCauchySeal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchySeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5959.1 and Definition 5959.2 seed the finite route from located nested intervals through dyadic diameter budgets, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedIntervalCauchySeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim interval intersection, quotient Real construction, selected host limit, countable choice, detached completion data, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested located-interval containment, dyadic diameter budget exactness, StreamName and RegSeqRat readback, Real seal nonescape, and local naming for BEDC.Derived.NestedIntervalCauc

Constructive story: A $\mathsf{NestedIntervalCauchySeal}^{\uparrow}$ packet is generated from located nested interval rows, DyadicRatCore diameter budgets, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5960

A Concrete Naming Certificate for $\mathsf{Spectrahedron}^{\uparrow}$

Following Boyd–Vandenberghe convex optimization, the $\mathsf{Spectrahedron}^{\uparrow}$ packet records the finite BEDC surface by which an affine section of the semidefinite cone is presented as a convex feasible region. It sits after the matrix carrier of chapter 49, the linear-map row of chapter 48, the vector-space boundary of chapter 47, the semidefinite-cone packet of chapter 310, and the convex-set provenance of chapter 208. The chapter names the affine pencil route; it does not import host spectral calculus, determinant inequalities, quotient equality of matrices, selected feasible points, or ambient analytic closure.

Definition 5960.1 (Spectrahedron carrier)**.** *A* $\mathsf{Spectrahedron}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, L, A, S, K, V, T, C, P, N)$$

with the following displayed rows:

1. *X is the affine-parameter row over the displayed* $\mathsf{VecSpace}^{\uparrow}$ *source.*
2. *L is the* $\mathsf{LinMap}^{\uparrow}$ *row carrying the affine pencil direction from parameter rows to symmetric* $\mathsf{Mat}^{\uparrow}$ *rows.*
3. *A is the affine offset matrix row read through* $\mathsf{Mat}^{\uparrow}$*.*
4. *S is the symmetric matrix-pencil row naming the displayed sum $A + L(x)$ for each accepted parameter row.*
5. *K is the* $\mathsf{SemidefiniteCone}^{\uparrow}$ *membership row for the displayed pencil matrix.*
6. *V is the* $\mathsf{ConvexSet}^{\uparrow}$ *provenance row that records the accepted parameter rows as a convex feasible subset.*
7. *T records componentwise* hsame *transport for parameter, linear-map, matrix, cone-membership, convex-provenance, replay, package, and local naming rows.*
8. *C records displayed* Cont *replay from parameter reads through the affine pencil, semidefinite-cone membership, and convex-set provenance rows.*
9. *P is the* Pkg *provenance over* $\mathsf{Spectrahedron}^{\uparrow}$*,* $\mathsf{Mat}^{\uparrow}$*,* $\mathsf{SemidefiniteCone}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{LinMap}^{\uparrow}$*,* $\mathsf{VecSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, L, A, S, K, V, T, C, P, N$. It has no coordinate for eigenvalue decompositions, determinant tests, quotient equality of matrix pencils, selected feasible points, or ambient closed-convex analytic structure.

Theorem 5960.2 (Spectrahedron NameCert obligations)**.** *For every accepted* $\mathsf{Spectrahedron}^{\uparrow}$ *carrier*

$$Q = (X, L, A, S, K, V, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed parameter row X, linear-map row L, offset matrix row A, matrix-pencil row S, semidefinite-cone membership row K, convex-set provenance row V, componentwise* hsame *transport through T, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5960.1. Its coordinates are the parameter, linear-map, offset, matrix-pencil, semidefinite-cone, convex-provenance, transport, replay, package, and local naming rows.

The affine-packet obligations are the rows X, L, A, S. They force every consumer to read the candidate point through a displayed $\mathsf{VecSpace}^{\uparrow}$ parameter row, the $\mathsf{LinMap}^{\uparrow}$ pencil direction, and the $\mathsf{Mat}^{\uparrow}$ offset-and-sum surface.

The feasibility and convexity obligations are the rows K and V. The semidefinite membership row is read through $\mathsf{SemidefiniteCone}^{\uparrow}$, while convex provenance is read through $\mathsf{ConvexSet}^{\uparrow}$ over the same parameter surface.

Finally T, C, P, N transport, replay, source, and name the route. Since no other coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 5960.3 (Spectrahedron semidefinite-cone handoff)**.** *Every feasibility-facing consumer of an accepted* $\mathsf{Spectrahedron}^{\uparrow}$ *carrier must factor through the displayed affine pencil route*

$$X, L, A \longmapsto S \longmapsto K \longmapsto V, T, C, P, N.$$

Thus the consumer receives a parameter row, affine matrix-pencil row, semidefinite-cone membership row, convex-set provenance row, transport, replay, package provenance, and local NameCert rather than a host feasible set, spectral selector, determinant criterion, quotient matrix equality, selected feasible point, or ambient analytic closure theorem.

Proof. Begin with a feasibility-facing read of Definition 5960.1. The only route from a parameter row to matrix data is the displayed affine pencil formed by L and A and named by S.

Next read the cone-membership row K. Because K is a $\mathsf{SemidefiniteCone}^{\uparrow}$ row over the displayed matrix pencil, the feasibility read cannot bypass the sibling semidefinite-cone carrier or replace it with a host positivity predicate.

The convex provenance row V then records the accepted parameter surface as a $\mathsf{ConvexSet}^{\uparrow}$ consumer. The route remains tied to the same $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, and $\mathsf{SemidefiniteCone}^{\uparrow}$ rows.

Finally T, C, P, N preserve and name the same finite route under transport and replay. No excluded analytic or host object is stored in the carrier, so every consumer factors through the displayed handoff. □

Theorem 5960.4 (Spectrahedron semidefinite-cone route)**.** *For every accepted* $\mathsf{Spectrahedron}^{\uparrow}$ *carrier*

$$Q = (X, L, A, S, K, V, T, C, P, N),$$

each accepted affine-pencil read must display the $\mathsf{Mat}^{\uparrow}$ *rows* A, S*, the* $\mathsf{LinMap}^{\uparrow}$ *row* L*, the* $\mathsf{VecSpace}^{\uparrow}$ *parameter row* X*, the* $\mathsf{SemidefiniteCone}^{\uparrow}$ *membership row* K*, and the* $\mathsf{ConvexSet}^{\uparrow}$ *provenance row* V *before the local* $\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$ *row* N *is public. The route exports no host spectral calculus, determinant inequality, quotient matrix equality, selected feasible point, or analytic closure principle.*

Proof. Project the carrier of Definition 5960.1. The affine-pencil read begins at the $\mathsf{VecSpace}^{\uparrow}$ parameter row X, passes through the $\mathsf{LinMap}^{\uparrow}$ row L, reads the $\mathsf{Mat}^{\uparrow}$ offset row A, and names the displayed pencil matrix as S.

Apply the NameCert obligations of Definition 5960.2. They require the semidefinite-cone membership row K and the convex-set provenance row V as local obligations of the same packet, not as external analytic facts.

The semidefinite-cone handoff theorem Definition 5960.3 then forces every feasibility-facing consumer through K and V before the transport, replay, provenance, and naming rows T, C, P, N are read.

Because the carrier contains no coordinate for spectral decomposition, determinant feasibility, quotient equality of matrix pencils, selected feasible points, or analytic closedness, the public route cannot export any of those host objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$.

Constructive story: A $\mathsf{Spectrahedron}^{\uparrow}$ packet is generated from an affine parameter row, LinearMap pencil row, Matrix offset row, matrix-pencil row, SemidefiniteCone membership row, ConvexSet provenance row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$, seed)

Scope closed: Definition 5960.1, Definition 5960.2, and Definition 5960.3 seed the affine semidefinite-feasibility route through parameter, pencil, cone, convex, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{Spectrahedron}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, spectral decomposition, determinant feasibility criteria, quotient equality of matrix pencils, selected feasible points, or ambient analytic closedness of semidefinite sections.

Upgrade path: Obligation closure requires checked carrier admission, affine-parameter visibility, linear-map pencil exactness, matrix offset and pencil replay, semidefinite-cone membership, convex-set provenance, transport stability, replay stability, and local naming for BEDC.Derived.SpectrahedronUp.

The $\mathsf{Spectrahedron}^{\uparrow}$ dossier lattice depends on the concrete sibling route through chapter 310 and chapter 208. A spectrahedron reads a finite affine slice of the semidefinite cone, so its carrier consumes the cone classifier and the convex-set intersection rows before the local NameCert is public. Every $\mathsf{Spectrahedron}^{\uparrow}$ carrier reads its affine-linear matrix inequality through the displayed $\mathsf{SemidefiniteCone}^{\uparrow}$ membership boundary of chapter 310 and the $\mathsf{ConvexSet}^{\uparrow}$ provenance boundary of chapter 208 before exposing the local NameCert. Thus the affine pencil remains a finite forward route from parameter rows to cone membership and convex feasible-surface rows, not a host-level closed semialgebraic set.

Theorem 5960.5 (Spectrahedron sibling dependency route)**.** *Every accepted* $\mathsf{Spectrahedron}^{\uparrow}$ *carrier reads the matrix carrier of chapter 49 as offset rows, the affine pencil rows of chapter 48, vector-space parameter rows from chapter 47, semidefinite cone membership rows from chapter 310, and convex-set provenance rows from chapter 208 before exposing the local spectrahedron* NameCert.

Proof. Project the accepted carrier. Its affine-semidefinite feasibility surface contains a matrix offset row, a linear pencil row, parameter-coordinate rows, positive semidefinite membership, convex provenance, and the local naming row.

The sibling references above exhaust those coordinates. Thus the spectrahedron packet reads the sibling rows as finite provenance before the local NameCert is admitted, rather than importing a host feasible set. □

Chapter 5961

A Concrete Naming Certificate for $\mathsf{NestedRadicalConvergence}^{\uparrow}$

Following Herschfeld and Bishop–Bridges constructive analysis, the $\mathsf{NestedRadicalConvergence}^{\uparrow}$ packet records bounded nested-radical convergence as finite NameCert data. It sits over the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers a displayed route from radical coefficients and finite tail bounds to a regular Cauchy handoff without invoking ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, or host square-root convergence.

Definition 5961.1 (Nested radical convergence carrier)**.** *A* $\mathsf{NestedRadicalConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (A, S, D, W, T, Q, H, C, P, N)$$

with the following displayed rows:

1. *A is the radical-coefficient row, listing the finite prefix of nonnegative dyadic or rational coefficients requested by the consumer.*
2. *S is the finite stage row, recording the stagewise radical approximants built by the displayed arithmetic route.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the comparison and refinement budgets used to measure adjacent stages.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only finite windows of the approximation stream.*
5. *T is the tail-control row, recording the Herschfeld-style boundedness and finite tail estimates visible at the requested precision.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, presenting the finite windows and dyadic tolerances as a regular rational sequence.*
7. *H records componentwise* hsame *transport for A, S, D, W, T, Q.*
8. *C records displayed* Cont *replay from coefficient rows through stage construction, tolerance checking, finite windows, tail control, and regular readback.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedRadicalConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NestedRadicalConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a host infinite radical, quotient equality of streams, selected limiting representative, countable choice, trichotomy, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Theorem 5961.2 (Nested radical convergence NameCert obligations)**.** *For every accepted* NestedRadicalConvergenc *carrier*

$$R = (A, S, D, W, T, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NestedRadicalConvergence}^{\uparrow}}$ *surface consists exactly of finite rows for coefficient admission, stage construction, dyadic tolerance checking, StreamName window exposure, tail-control exposure, RegSeqRat readback, componentwise transport, displayed replay, provenance, and local naming. No obligation row may replace T by an unbounded convergence theorem, bypass D and W, or export a completed real without the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

Lean4 ✓ `BEDC.Derived.NestedRadicalConvergenceUp.NestedRadicalConvergenceTasteGate single carrier alignment`

Proof. Project the carrier of Definition 5961.1. The coefficient row A and finite stage row S are the only source of radical approximants.

The tolerance and window rows are explicit. The dyadic ledger D supplies the requested comparison budget, while W restricts every read to a finite StreamName window.

The tail row T is the convergence-facing datum. It records the bounded nested-radical estimates used at the requested precision, but it is not a global host theorem about infinite radicals.

Finally Q, H, C, P, N read the finite data as a regular rational sequence, transport the same rows, replay the construction, record provenance, and name the packet. Since every public read factors through those rows, the NameCert surface cannot import quotient streams, selected limits, or ambient completeness. □

Theorem 5961.3 (Nested radical convergence RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{NestedRadicalConvergence}^{\uparrow}$ *carrier must read the finite route*

$$A, S, D, W, T \longmapsto Q \longmapsto H, C, P, N$$

before any $\mathsf{Real}^{\uparrow}$ *seal may be consumed. The handoff exposes radical coefficients, finite stage approximants, dyadic tolerances, StreamName windows, and tail-control rows as a regular rational sequence; it does not assert Real completeness, quotient equality of approximation streams, selected limits, countable choice, trichotomy, or host square-root convergence.*

Proof. Begin with the NameCert obligations of Definition 5961.2. They place coefficient admission, stage construction, dyadic tolerance checking, finite windows, and tail control before the readback row.

The $\mathsf{RegSeqRat}^{\uparrow}$ row Q is therefore downstream of A, S, D, W, T. It can read only the displayed finite approximants and the dyadic tolerance ledger, so it cannot replace the carrier by a quotient stream or by a selected limiting representative.

Transport, replay, provenance, and local naming occur after the handoff. The terminal $\mathsf{Real}^{\uparrow}$ seal can be cited only as a consumer of the regular rational route, not as an ambient completeness principle that creates the route. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedRadicalConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{NestedRadicalConvergence}^{\uparrow}$ packet is generated from radical coefficient rows, finite stage approximants, dyadic tolerance rows, StreamName windows, tail-control estimates, RegSeqRat readback, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NestedRadicalConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5961.1, Definition 5961.2, and Definition 5961.3 seed the finite nested-radical route into regular rational readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NestedRadicalConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient Real completeness, quotient stream equality, selected limits, countable choice, trichotomy, host square-root convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, stage construction, dyadic tolerance exactness, StreamName window exposure, tail-control replay, RegSeqRat handoff, Real-seal nonescape, transport, provenance, and local naming for BEDC.Derived.NestedRadicalConvergenceUp.

Chapter 5962

A Concrete Naming Certificate for $\mathsf{BoundedNormalizationFuel}^{\uparrow}$

The $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ packet realizes the bounded normalisation endpoint boundary of chapter 9379 as finite checker data. It sits beside the decidability boundary of chapter 749, the budgeted checker route of chapter 5949, the fuel-trace frontier of chapter 5936, and the closed normal-confluence seal consumed when finished endpoints are visible. The packet records fuel-sensitive conversion endpoints without turning bounded computation into unrestricted beta-conversion decidability.

Definition 5962.1 (Bounded normalization fuel carrier)**.** *A* $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (\Gamma, t, u, k, B, E, Z, R, H, C, P, N)$$

with the following displayed rows:

1. Γ *is the represented MetaCIC context row for the checker request.*
2. t, u *are the two closed or context-indexed term rows whose bounded normal forms are compared.*
3. k *is the visible natural fuel row, including the decrement ledger consumed by the normalizer.*
4. B *is the bounded normalizer row, exposing the two computed endpoint terms at fuel* k*.*
5. E *is the equality-check row, comparing the two bounded endpoints by structural same-term checking.*
6. Z *is the finished-normal row, present exactly when both bounded endpoints carry finished-normalisation evidence at the displayed fuel.*
7. R *is the refusal row, recording timeout, missing finished evidence, missing closedness evidence, or attempted fuel erasure.*
8. H *records componentwise* hsame *transport for* $\Gamma, t, u, k, B, E, Z, R$*.*
9. C *records displayed* Cont *replay from the checker request through fuel, bounded endpoints, equality, finished-normal evidence, refusal, and structural rows.*
10. P *is the* Pkg *provenance over* $\mathsf{BoundedNormalizationFuel}^{\uparrow}$*,* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$*,* $\mathsf{BoundedTypecheckingBu}$ $\mathsf{MetaCICNormalizationFuelBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
11. N *is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFuel}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for unrestricted beta-conversion decidability, fuel-erased normalization, proof-irrelevant term equality, quotient syntax equality, host conversion, or closed consistency.

Theorem 5962.2 (Bounded normalization fuel NameCert obligations)**.** *For every accepted* BoundedNormalizationF *carrier*

$$F = (\Gamma, t, u, k, B, E, Z, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFuel}^{\uparrow}}$ *surface consists exactly of finite rows for checker-context admission, term-row admission, fuel-ledger exposure, bounded endpoint computation, structural endpoint comparison, finished-normal evidence when present, refusal visibility, componentwise transport, displayed replay, provenance, and local naming. No obligation row may delete* k*, infer* Z *from* B *alone, hide* R*, or export unrestricted closed beta-conversion.*

Proof. Project the carrier of Definition 5962.1. The checker context, term rows, and fuel row are public before any endpoint comparison is read.

The bounded endpoint row B is fuel-sensitive. It records what the normalizer produced at the displayed fuel k, while the equality row E compares only those bounded outputs.

The finished-normal row Z is separate. When it is present, it justifies the stronger endpoint reading named by the decidability audit map; when it is absent, the refusal row R remains part of the same public packet.

Finally H, C, P, N preserve, replay, source, and name the finite route. Since none of those rows contains an unrestricted normalizer or host conversion oracle, the listed obligations exhaust the local certificate surface. □

Theorem 5962.3 (Bounded normalization fuel finished boundary)**.** *Every public conversion-facing read of an accepted* $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ *carrier factors through*

$$\Gamma, t, u, k, B, E, Z, R \longmapsto H, C, P, N.$$

If Z *is displayed, the consumer may use the finished-normal endpoint handoff attached to the bounded equality row* E*. If* Z *is absent, the refusal row* R *blocks any export of unrestricted beta-conversion decidability, fuel-erased normalization, proof-irrelevant equality, quotient syntax equality, host conversion, or closed consistency.*

Proof. Start with the NameCert obligations in Definition 5962.2. They order the public route through context, terms, visible fuel, bounded endpoint computation, structural comparison, finished evidence, and refusal.

When Z is present, the endpoint evidence is still tied to the displayed fuel k and the bounded endpoint row B. Thus the consumer receives a finished-boundary handoff, not a global conversion oracle.

When Z is absent, R is mandatory. It records the timeout or missing endpoint evidence and prevents E from being read as unrestricted beta-conversion decidability.

Transport, replay, provenance, and local naming preserve the same distinction. The packet therefore has exactly the finite endpoint boundary stated above. □

Theorem 5962.4 (Bounded normalization fuel monotone readback)**.** *If an accepted* $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ *packet is reread at a larger displayed fuel, the public conversion-facing route must still expose the original context row, term rows, visible fuel row, bounded endpoint row, structural equality row, finished-normal row when present, and refusal row when absent. Increasing fuel may refine the bounded endpoint row, but it cannot delete fuel, infer finished evidence from bounded endpoints alone, hide refusal, or export unrestricted beta-conversion decidability.*

Proof. Project the carrier of Definition 5962.1. The reread keeps the same Γ, t, u request surface and replaces only the displayed fuel request by a larger visible fuel row.

Apply Definition 5962.2. The endpoint row may be recomputed at the larger fuel, but the public route still has to display fuel, bounded endpoints, structural equality, finished-normal evidence when present, refusal when absent, and the structural rows.

Now use Definition 5962.3. A refined endpoint row does not by itself create finished-normal evidence; the finished case remains tied to Z, and the nonfinished case remains tied to R.

Thus monotone fuel readback is still a bounded computation route. It can refine what the bounded endpoint row says at the larger fuel, but it cannot become an unrestricted conversion oracle. □

Falsifiable prediction: A $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ consumer that exports unrestricted closed beta-conversion decidability after deleting the fuel row k, the finished-normal row Z, or the refusal row R refutes the carrier of Definition 5962.1.

Independence witness: Nearest siblings: metacicdecidableboundary, bounded-typechecking-budget, metacic-normalization-fuel-boundary, and closed-normal-confluence-seal. $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ is not bijective to any of them because it jointly carries checker context, two term rows, visible fuel, bounded endpoint computation, structural endpoint equality, finished-normal evidence, refusal, transport, replay, provenance, and local naming as one carrier.

The Lean TasteGate follow-up for this packet requires explicit $\mathsf{Nontrivial\ BoundedNormalizationFuel}^{\uparrow}$ and $\mathsf{FieldFaithful\ BoundedNormalizationFuel}^{\uparrow}$ instances in the corresponding TasteGate file. Nontriviality must separate accepted finished endpoint packets from timeout or missing-evidence packets, and field faithfulness must preserve $\Gamma, t, u, k, B, E, Z, R, H, C, P, N$ without erasing fuel or refusal rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFuel}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedNormalizationFuel}^{\uparrow}$ packet is generated from checker-context rows, term rows, visible fuel, bounded endpoint computation, structural endpoint comparison, finished-normal evidence, refusal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationFuel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5962.1, Definition 5962.2, and Definition 5962.3 seed the finite endpoint boundary for bounded normal conversion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationFuel}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted beta-conversion decidability, fuel-erased normalization, proof-irrelevant equality, quotient syntax equality, host conversion, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fuel-ledger exactness, bounded endpoint computation, structural endpoint comparison, finished-normal boundary, refusal nonescape, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.BoundedNormalizationFuelUp.

Chapter 5963

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedField}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedField}^{\uparrow}$ packet records the L10 algebra surface available from regular Cauchy real names when every reciprocal and division read is guarded by displayed apartness. It sits over the regular rational source surface of chapter 301, finite stream windows from chapter 126, dyadic endpoint arithmetic from chapter 291, the sum route of chapter 417, the product route of chapter 461, the reciprocal route of chapter 1471, the division route of chapter 1943, the apartness and order boundary of chapter 1710, and the terminal seal of chapter 38. The chapter names a constructive field-operation packet for real-name algebra; it does not identify quotient real equality, trichotomy, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5963.1 (Regular Cauchy located field carrier)**.** *A* $\mathsf{RegularCauchyLocatedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, D, A, M, I, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S lists the* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows for the regular Cauchy names consumed by the algebra packet.*
2. *W records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows opened for every source read and every operation readback.*
3. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows used for rational endpoint arithmetic, tolerance comparison, and finite lower-bound inspection.*
4. *A is the apartness guard row from the* $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$ *boundary; it is required before any reciprocal or quotient route can be read.*
5. *M is the additive and multiplicative operation inventory, consisting of* $\mathsf{RegularCauchySum}^{\uparrow}$*,* $\mathsf{RegularCauchyProduct}^{\uparrow}$*, and scalar endpoint rows visible at the same window level.*
6. *I is the guarded reciprocal route from* $\mathsf{RegularCauchyReciprocal}^{\uparrow}$*, read only after A has displayed a positive nonzero-domain witness.*
7. *Q is the guarded division route from* $\mathsf{RegularCauchyDivision}^{\uparrow}$*, factoring quotient reads through the same apartness guard, reciprocal route, product route, and regular readback surface.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal receiving only the displayed regular Cauchy readbacks and their visible operation provenance.*

9. *H records componentwise* hsame *transport for $S, W, D, A, M, I, Q, E, C, P, N$.*
10. *C records displayed* Cont *replay for source inspection, window inspection, dyadic endpoint arithmetic, apartness exposure, sum and product exposure, reciprocal and division handoff, readback, and seal handoff.*
11. *P records* Pkg *provenance over* $\mathsf{RegularCauchyLocatedField}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyReciprocal}^{\uparrow}$, $\mathsf{RegularCauchyDivision}^{\uparrow}$, $\mathsf{RegularCauchyApartnessOrder}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedField}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the displayed rows $S, W, D, A, M, I, Q, E, H, C, P, N$.

Theorem 5963.2 (Regular Cauchy located field NameCert obligations)**.** *The seed* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedField}^{\uparrow}}$ *surface for Definition 5963.1 consists exactly of carrier admission, operation exposure, reciprocal and division reads guarded by apartness, regular readback, terminal Real seal nonescape, componentwise transport, displayed continuation replay, package provenance, and the local naming row.*

Proof. Carrier admission is projection from the displayed tuple $F = (S, W, D, A, M, I, Q, E, H, C, P, N)$. The source rows, stream windows, dyadic endpoint arithmetic, apartness guard, operation inventory, reciprocal route, division route, seal, transports, continuations, provenance, and local name are all explicit coordinates.

Operation exposure is finite. Sum and product reads are carried by M, while reciprocal reads are carried by I and quotient reads are carried by Q. The reciprocal and division rows are downstream of A, so no inverse or quotient consumer can bypass the positive apartness guard.

Readback and seal nonescape are terminal. The route reaches $\mathsf{RegSeqRat}^{\uparrow}$ readbacks through the operation rows and then reaches E as a $\mathsf{Real}^{\uparrow}$ seal; the carrier has no coordinate for quotient real equality, trichotomy, selected representatives, countable choice, or ambient completeness.

Transport, replay, and provenance are exactly H, C, P, N. They move and re-expose the same rows componentwise, so the local NameCert surface is exhausted by the displayed packet. □

Theorem 5963.3 (Regular Cauchy located field guarded division scope)**.** *Every reciprocal or quotient read from an accepted* $\mathsf{RegularCauchyLocatedField}^{\uparrow}$ *packet factors through*

$$S, W, D \longmapsto A \longmapsto I, Q \longmapsto E, H, C, P, N.$$

The field-operation packet therefore exposes guarded real-name algebra only through finite regular Cauchy source data, dyadic endpoint arithmetic, apartness evidence, reciprocal and division handoffs, and the terminal $\mathsf{Real}^{\uparrow}$ *seal.*

Proof. Project the packet. The rows S, W, D supply the visible regular source data, stream windows, and endpoint arithmetic used by every operation read.

The guard row A is the only coordinate that licenses reciprocal or division access. Hence I and Q can be read only after the apartness boundary has been displayed.

The terminal rows E, H, C, P, N seal and replay that finite route. Since no additional coordinate is present, the carrier cannot export total inversion, trichotomy, quotient equality, selected representatives, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedField}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedField}^{\uparrow}$ packet is generated from RegSeqRat source rows, StreamName windows, DyadicRatCore endpoint arithmetic, RegularCauchySum and RegularCauchyProduct operation rows, apartness-guarded RegularCauchyReciprocal and RegularCauchyDivision rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{RegularCauchyLocatedField}^{\uparrow}}$, seed)

Scope closed: Definition 5963.1, Definition 5963.2, and Definition 5963.3 seed the constructive located-field operation boundary over regular Cauchy real names.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyLocatedField}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, trichotomy, selected representatives, countable choice, total inversion, ambient Real$^{\uparrow}$ completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, operation exposure, apartness-guarded reciprocal and division handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedFieldUp

Chapter 5964

A Concrete Naming Certificate for CompactLipschitzUniformHandoff$^{\uparrow}$

Following Bishop–Bridges style compact metric uniform-continuity, the CompactLipschitzUniformHandoff$^{\uparrow}$ packet records the finite route by which compact finite-net data and a Lipschitz graph row supply a tolerance-only UniformModulus$^{\uparrow}$ consumer. It is grounded in the compact-uniform-continuity vision of chapter 9237; it reads compact metric nets from chapter 129, continuous graph rows from chapter 130, dyadic tolerance rows from chapter 291, rational lower-bound comparison from chapter 37, the uniform-modulus handoff of chapter 369, and the terminal real-name seal of chapter 38. The chapter packages the finite-net, Lipschitz graph, tolerance-fold, and uniform-modulus route without importing choice, open-cover compactness, quotient function equality, or ambient Real$^{\uparrow}$ completeness.

Definition 5964.1 (Compact Lipschitz uniform handoff carrier)**.** *A* CompactLipschitzUniformHandoff$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, G, T, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *finite-net row, exposing a finite* ProbeBundle *of compact source centers at the requested tolerance scale.*
2. *G is the* ContinuousMap$^{\uparrow}$ *graph row together with its Lipschitz ledger: one rational bound, source and target metric rows, and the graph-distance implication consumed by the uniform estimate.*
3. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying endpoint windows for the requested target tolerance and the derived source radius.*
4. *R is the* Rat$^{\uparrow}$ *lower-bound fold over the finite list of compact-net radii and Lipschitz-derived source radii.*
5. *U is the* UniformModulus$^{\uparrow}$ *handoff row exported to downstream compact-continuous consumers.*
6. *E is the terminal* Real$^{\uparrow}$ *seal, reached only after the compact net, graph, tolerance, rational fold, and uniform-modulus rows have been displayed.*
7. *H records componentwise* hsame *transport for $K, G, T, R, U, E, C, P, N$.*
8. *C records displayed* Cont *replay for compact finite-net inspection, graph and Lipschitz inspection, dyadic tolerance comparison, rational lower-bound folding, uniform-modulus handoff, and seal handoff.*

9. *P records* Pkg *provenance over* CompactLipschitzUniformHandoff$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Rat$^{\uparrow}$, UniformModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{CompactLipschitzUniformHandoff}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through the rows $K, G, T, R, U, E, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.CompactLipschitzUniformHandoffUp.CompactLipschitzUniformHandoffTasteGate`
`single carrier alignment`

Theorem 5964.2 (Compact Lipschitz uniform handoff NameCert obligations)**.** *The seed* NameCert$_{\text{CompactLipschitzUn}}$ *surface for Definition 5964.1 consists exactly of carrier admission, compact finite-net exposure, continuous graph and Lipschitz ledger exposure, dyadic tolerance inspection, rational lower-bound folding,* UniformModulus$^{\uparrow}$ *handoff, terminal Real seal nonescape, componentwise transport, displayed continuation replay, package provenance, and the local naming row.*

Proof. Carrier admission is projection from $L = (K, G, T, R, U, E, H, C, P, N)$. The compact metric net, graph and Lipschitz row, dyadic tolerance row, rational fold, uniform-modulus handoff, seal, transports, continuations, provenance, and local name are explicit coordinates.

The analytic route is finite. The compact row K supplies only the listed ProbeBundle centers, the graph row G supplies the Lipschitz estimate at those source distances, and the tolerance row T supplies the dyadic scale consumed by the lower-bound fold R.

The uniform-modulus row U is downstream of K, G, T, R. It can therefore be replayed only after the finite-net and Lipschitz ledger have fixed the source precision. The seal E receives this handoff without adding open-cover compactness, selected moduli, quotient function equality, or ambient completeness.

Transport, replay, and provenance are exactly H, C, P, N. Componentwise hsame and displayed Cont rows re-expose the same finite route, so the NameCert obligations are exhausted by the carrier. □

Theorem 5964.3 (Compact Lipschitz uniform consumer route)**.** *Every* UniformModulus$^{\uparrow}$ *consumer of an accepted* CompactLipschitzUniformHandoff$^{\uparrow}$ *packet factors through*

$$K, G, T \longmapsto R \longmapsto U \longmapsto E, H, C, P, N.$$

Thus the exported modulus is a finite compact-net and Lipschitz-ledger handoff, not a choice functional, not an open-cover argument, and not a quotient-function equality principle.

Proof. Read the compact source row K and graph row G first. These rows expose the finite centers, source and target metric data, and Lipschitz bound needed by the consumer route.

Use the dyadic tolerance row T to form the rational comparison scale. The finite lower-bound fold R is computed over the displayed compact-net and Lipschitz-derived radii, so it is a Rat$^{\uparrow}$ ledger rather than an external selector.

The uniform-modulus row U is then the only exported consumer row. The remaining rows E, H, C, P, N seal, transport, replay, source, and name this same finite route, leaving no coordinate for open-cover compactness, countable choice, selected moduli, quotient function equality, or ambient Real$^{\uparrow}$ completeness. □

Closure status for NameCert$_{\text{CompactLipschitzUniformHandoff}^{\uparrow}}$**.**

Constructive story: A CompactLipschitzUniformHandoff$^{\uparrow}$ packet is generated from CompactMetric finite-net rows, a ContinuousMap graph with Lipschitz ledger, DyadicRatCore tolerance rows, a RatUp lower-bound fold, a UniformModulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CompactLipschitzUniformHandoff}^{\uparrow}}$, seed)

Scope closed: Definition 5964.1, Definition 5964.2, and Definition 5964.3 seed the finite compact-net and Lipschitz-ledger route into a UniformModulus consumer.

Formal verification: FormalStatus(NameCert$_{\mathsf{CompactLipschitzUniformHandoff}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected moduli, countable choice, quotient function equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or standard bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exposure, Lipschitz graph exposure, tolerance folding, UniformModulus handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactLipschitzUniformHandoffUp.

RegularSequenceSpace$^{\uparrow}$ routes completion-facing sequence reads through chapter 301, chapter 126, chapter 291, and chapter 6003 before the local hub inputs expose the sequence-space certificate.

Chapter 5965

A Concrete Naming Certificate for $\mathsf{RegularSequenceSpace}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy sequences, the $\mathsf{RegularSequenceSpace}^{\uparrow}$ packet records the finite carrier for a displayed space of regular sequence names rather than a single regular sequence certificate. It sits between the stream schedules of chapter 126, the rational regular readbacks of chapter 301, the dyadic rational core of chapter 291, and the value-facing real seal of chapter 38. The same packet also consumes the explicit schedule-family surface of chapter 5887 and the rate ledger of chapter 8037 before tail-filter or completion-facing consumers read the terminal Real seal.

Definition 5965.1 (RegularSequenceSpace carrier)**.** *A* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, T, Q, D, R, H, C, P, N)$$

with the following rows:

1. *X is the displayed index-source row naming the finite family of regular sequence names currently admitted to the space packet.*
2. *T is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the inspected tails and precision windows.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for regular rational Cauchy names at those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale row recording the dyadic tolerance core used by the regularity checks.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *value-facing seal row reached only after X, T, Q, D have stayed visible.*
6. *H records componentwise* hsame *transport for the source, schedule, readback, dyadic scale, value seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a space-level sequence request through the stream schedule, regular rational readback, dyadic tolerance core, and real-facing seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient complete metric space, quotient equality of real names, arbitrary choice of representatives, unrestricted sequence space, or host set of all Cauchy sequences.

Theorem 5965.2 (RegularSequenceSpace NameCert obligations)**.** *For every accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier* $S = (X, T, Q, D, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$ *surface consists exactly of space-level carrier admission through* X*, schedule exposure through* T*, regular rational readback through* Q*, dyadic tolerance exposure through* D*, real-facing value seal through* R*, componentwise* hsame *stability through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*. A consumer therefore reads a regular sequence space only through the finite route*

$$X, T, Q, D \longmapsto R, H, C, P, N.$$

Proof. Project the accepted carrier of Definition 5965.1. The source row X opens the space-level family, while the schedule row T and regular rational readback row Q expose only displayed sequence windows.

The dyadic tolerance core D supplies the regularity scale before the real-facing seal R is consumed. Thus the packet reaches $\mathsf{Real}^{\uparrow}$ only after the stream, rational readback, and dyadic core rows have remained visible.

Finally replay the structural rows. Componentwise transport H, continuation replay C, provenance P, and local naming N preserve the same finite carrier, so no ambient complete space, unrestricted sequence space, quotient of real names, or arbitrary representative choice enters the NameCert surface. □

Lean4 ✓ `BEDC.Derived.RegularSequenceSpaceUp.RegularSequenceSpaceNameCertObligations`

Theorem 5965.3 (RegularSequenceSpace tail-filter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier* $S = (X, T, Q, D, R, H, C, P, N)$ *factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the regular Cauchy tail-filter route of chapter 1605 before the* $\mathsf{Real}^{\uparrow}$ *seal* R *can be consumed. The accepted space packet therefore exports only the finite regular-name route*

$$X, T, Q, D \longmapsto \mathsf{RegularCauchyTailFilter}^{\uparrow} \longmapsto R, H, C, P, N$$

and no quotient stream, selected limit, countable-choice selector, unrestricted sequence space, or ambient completeness witness.

Proof. Project the carrier of Definition 5965.1. The source row X names the admitted family, while T, Q, D display the stream schedule, rational readback, and dyadic tolerance rows available to a completion-facing consumer.

The schedule row T and dyadic row D are read as the $\mathsf{CauchyModulusSpace}^{\uparrow}$ and $\mathsf{CauchyRateSpace}^{\uparrow}$ budget interface, not as a selected modulus or an ambient complete-space datum.

Apply the NameCert obligations of Definition 5965.2. They require the real-facing seal R to be reached only after X, T, Q, D remain visible in the same packet.

Now route late observations through chapter 1605. Its regular Cauchy tail-filter surface consumes the same stream, readback, and dyadic tolerance data before a Real-facing tail read is available.

Finally H, C, P, N preserve that finite route by transport, replay, provenance, and local naming. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, an unrestricted sequence space, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no completion-facing consumer can escape the displayed tail-filter handoff. □

Theorem 5965.4 (RegularSequenceSpace tail coherence)**.** *Let* $S = (X, T, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any two completion-facing reads from* S *that agree on*

the displayed $\mathsf{StreamName}^{\uparrow}$ *schedule* T *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *also agree on the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *and on the terminal* $\mathsf{Real}^{\uparrow}$ *seal route* R, H, C, P, N*. This agreement is internal to the displayed carrier and does not use quotient streams, selected representatives, or an ambient complete sequence space.*

Proof. Project the carrier of Definition 5965.1. The tuple $S = (X, T, Q, D, R, H, C, P, N)$ keeps the source family, schedule, regular readback, and dyadic tolerance row visible as separate coordinates.

Use the NameCert obligations of Definition 5965.2. They require any completion-facing read to expose X, T, Q, D before the real-facing seal and structural rows can be consumed.

Route the late observations through the tail-filter handoff of Definition 5965.3. Two reads with the same displayed schedule T and readback Q therefore enter the same regular Cauchy tail-filter surface, where the dyadic tolerance row D is the displayed tolerance coordinate for that route.

Finally H, C, P, N preserve and name the same finite packet. Hence the two reads agree on the dyadic tolerance row and terminal Real seal route without choosing a quotient stream, representative, or external completion object. □

Theorem 5965.5 (RegularSequenceSpace value-seal exhaustion)**.** *Every value-facing read from an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier* $S = (X, T, Q, D, R, H, C, P, N)$ *exposes the source family* X*,* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D *before it may consume* R, H, C, P, N*. Consequently the terminal* $\mathsf{Real}^{\uparrow}$ *seal is exhausted by the displayed StreamName, RegSeqRat, DyadicRatCore, transport, replay, provenance, and NameCert route, with no ambient complete sequence space, quotient stream equality, selected representative, or unrestricted sequence object.*

Proof. Use the NameCert obligations of Definition 5965.2. They require every accepted read to keep the source row, schedule row, regular rational readback, and dyadic tolerance row visible before the Real-facing seal and structural rows can be consumed.

Apply the tail-filter handoff of Definition 5965.3. It routes completion-facing and value-facing reads through the same displayed stream schedule, regular rational readback, and dyadic tolerance data before the terminal seal is public.

Finally use tail coherence, Definition 5965.4. Since reads agreeing on T and Q also agree on D and on the terminal seal route R, H, C, P, N, the Real seal is exhausted by the displayed finite route and cannot import an ambient complete space, quotient stream equality, selected representative, or unrestricted sequence object. □

Theorem 5965.6 (RegularSequenceSpace pointwise stream projection)**.** *Every accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *read of a displayed family first exposes the source row* X *and the* $\mathsf{StreamName}^{\uparrow}$ *schedule* T*. It can therefore be projected to a finite PointwiseStreamName-style window, as in chapter 6084, before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and* $\mathsf{Real}^{\uparrow}$ *seal route* R, H, C, P, N *are consumed.*

Proof. Project the carrier of Definition 5965.1. The source row X and stream schedule T are the first two displayed coordinates of every accepted space read.

Apply the NameCert obligations of Definition 5965.2. They require schedule exposure before regular rational readback, dyadic tolerance exposure, and Real-facing sealing.

Use the tail-filter handoff of Definition 5965.3. Since the completion-facing route must pass through the same displayed schedule before Q, D, R, H, C, P, N are available, the read has a finite window that may be routed through the pointwise stream sibling.

The projection therefore consumes only X and T at the pointwise boundary. The later RegSeqRat, DyadicRatCore, and Real rows remain downstream and cannot be inspected before that finite window is displayed. □

Theorem 5965.7 (RegularSequenceSpace finite-window comparison)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any comparison between two displayed values of S must use the same* $\mathsf{StreamName}^{\uparrow}$ *finite window T,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, tail-filter ledger, dyadic tolerance row D, completion handoff, and transport, replay, provenance, and local naming rows H, C, P, N already exposed by the carrier. Thus a value comparison is a common-window comparison inside the displayed packet and cannot bypass the finite window or introduce quotient stream equality, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the pointwise stream projection of Definition 5965.6. It forces every displayed value read to expose the source row X and the finite stream window T before the readback, dyadic tolerance, and Real-facing seal are consumed.

Use tail coherence, Definition 5965.4. Once two reads share the displayed window and regular rational readback, they also share the dyadic tolerance row and the terminal seal route through R, H, C, P, N.

Finally apply value-seal exhaustion, Definition 5965.5. The terminal Real-facing comparison is exhausted by the displayed StreamName, RegSeqRat, DyadicRatCore, transport, replay, provenance, and NameCert rows, so no comparison can import quotient stream equality, selected limits, or ambient Real completeness. □

Theorem 5965.8 (RegularSequenceSpace obligation scope route)**.** *For every accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier $S = (X, T, Q, D, R, H, C, P, N)$, the carrier admission, tail-filter handoff, tail coherence, value-seal exhaustion, pointwise stream projection, and finite-window comparison obligations are all scoped by the finite rows*

$$\mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{NameCert},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

Thus a Real-facing or completion-facing consumer reaches the regular sequence space only through displayed history, componentwise transport, continuation replay, local naming, stream windows, regular rational readback, and the terminal Real seal.

Proof. Project the carrier of Definition 5965.1 and its NameCert obligations in Definition 5965.2. The accepted packet exposes the displayed history row, the componentwise hsame row, the Cont replay row, the local $\mathsf{NameCert}$ row, the $\mathsf{StreamName}^{\uparrow}$ schedule, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal as named coordinates of the same finite carrier.

Apply the tail-filter handoff and tail coherence results, Definition 5965.3 and Definition 5965.4. They keep completion-facing reads inside the same StreamName, RegSeqRat, dyadic tolerance, and Real-seal route.

Finally use value-seal exhaustion, pointwise stream projection, and finite-window comparison, Definition 5965.5, Definition 5965.6, and Definition 5965.7. These rows exhaust the value-facing and comparison obligations, so the scoped route has no coordinate for quotient streams, selected limits, arbitrary representatives, or ambient completeness. □

Theorem 5965.9 (RegularSequenceSpace obligation public interface)**.** *For every accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier $S = (X, T, Q, D, R, H, C, P, N)$, the carrier admission, tail-filter handoff, tail coherence, value-seal exhaustion, pointwise stream projection, and obligation scope route form the public obligation interface for completion-facing consumers. Every such consumer must pass through the*

displayed $\mathsf{StreamName}^{\uparrow}$ *schedule* T, $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q, $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D, *and terminal* $\mathsf{Real}^{\uparrow}$ *seal route* R, H, C, P, N. *The interface has no quotient stream, selected limit, ambient completeness coordinate, unrestricted sequence-space row, or host representative choice.*

Proof. Start with carrier admission and the NameCert obligations of Definition 5965.1 and Definition 5965.2. They expose $X, T, Q, D, R, H, C, P, N$ as the finite public packet and place the stream, readback, dyadic, seal, transport, replay, provenance, and local naming rows in one route.

Use the tail-filter handoff and tail coherence rows, Definition 5965.3 and Definition 5965.4. These keep completion-facing reads downstream of the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ data before any Real-facing read is public.

Next apply value-seal exhaustion and pointwise stream projection, Definition 5965.5 and Definition 5965.6. The terminal seal and every pointwise projection remain attached to the displayed finite window rather than to a quotient stream or chosen representative.

Finally invoke the obligation scope route of Definition 5965.8. The six named obligations therefore form exactly the public interface seen by completion-facing consumers through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, with no ambient completeness or unrestricted sequence-space coordinate. □

Theorem 5965.10 (RegularSequenceSpace RegSeqRat Real seal route)**.** *Every endpoint read from an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier factors first through the* $\mathsf{RegSeqRat}^{\uparrow}$ *classifier row* Q *and then through the* $\mathsf{Real}^{\uparrow}$ *seal row* R, *with the stream schedule* T, *dyadic tolerance row* D, *transport* H, *replay* C, *provenance* P, *and local naming* N *still visible. No endpoint read uses a quotient stream, an ambient complete sequence space, or an unrecorded completeness principle.*

Proof. Use the NameCert obligations of Definition 5965.2. They place the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q before the terminal $\mathsf{Real}^{\uparrow}$ seal R in every accepted value-facing route.

Apply value-seal exhaustion, Definition 5965.5. It states that the Real-facing seal is consumed only after the source, stream schedule, regular rational readback, and dyadic tolerance rows have stayed visible.

Now invoke the obligation scope route of Definition 5965.8. The remaining structural rows H, C, P, N preserve the same finite packet, so endpoint reads cannot pass through quotient streams, ambient completeness, or an unrecorded completion principle before the RegSeqRat classifier and Real seal route is displayed. □

Definition 5965.11 (RegularSequenceSpace Cauchy window)**.** *For an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier*

$$S = (X, T, Q, D, R, H, C, P, N),$$

the Cauchy-window row is the displayed subpacket

$$\mathrm{Win}_{\mathsf{RegularSequenceSpace}^{\uparrow}}(S) = (T, Q, D, R, H, C, P, N).$$

It records the $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, continuation replay, provenance, and local naming rows while omitting only the source family row* X.

Theorem 5965.12 (RegularSequenceSpace Cauchy-window NameCert route)**.** *Let* $S = (X, T, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every completion-facing read of* S *factors through the Cauchy-window subpacket* $\mathrm{Win}_{\mathsf{RegularSequenceSpace}^{\uparrow}}(S) = (T, Q, D, R, H, C, P, N)$ *before the* $\mathsf{Real}^{\uparrow}$ *seal is public. The route joins the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance row, and terminal $\mathsf{Real}^{\uparrow}$ *seal in one finite NameCert obligation and exports no quotient stream, selected representative, ambient completeness principle, or host real equality.*

Proof. Start with the NameCert obligations of Definition 5965.2. They expose $X, T, Q, D, R, H, C, P, N$ as the finite accepted carrier and place the schedule, readback, tolerance row, and Real seal in the same local certificate surface.

Apply the tail-filter handoff of Definition 5965.3. It routes completion-facing reads through the displayed schedule T, regular rational readback Q, and dyadic tolerance row D before the Real-facing row can be consumed.

Use tail coherence, Definition 5965.4. Once the completion read has entered the displayed window, the later rows R, H, C, P, N remain attached to the same finite route rather than becoming external representatives.

Finally apply the RegSeqRat Real seal route of Definition 5965.10. The terminal $\mathsf{Real}^{\uparrow}$ seal is public only after the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the visible structural rows have been read, so the Cauchy-window obligation is exhausted by (T, Q, D, R, H, C, P, N) and has no coordinate for quotient streams, selected representatives, ambient completeness, or host real equality. □

Theorem 5965.13 (RegularSequenceSpace RegSeqRat forward route)**.** *Every forward dependency from an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier to a regular rational or real-facing consumer factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

Thus the space packet contributes a horizontal L10 dependency edge to chapter 301 and chapter 38, rather than remaining a local carrier with no downstream regular-rational or real-name route.

Proof. Project the carrier of Definition 5965.1. The schedule row T and dyadic tolerance row D are exposed before the regular rational readback row Q and the terminal real-facing seal R.

Apply the RegSeqRat Real seal route of Definition 5965.10. It places every endpoint read through Q before R, with the stream schedule, dyadic tolerance, transport, replay, provenance, and local naming rows still visible.

Finally use the obligation scope route of Definition 5965.8. The resulting forward dependency is the displayed $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ route into $\mathsf{RegSeqRat}^{\uparrow}$ and then $\mathsf{Real}^{\uparrow}$, with no quotient stream, selected limit, unrestricted sequence space, or ambient completeness coordinate. □

Theorem 5965.14 (RegularSequenceSpace StreamName window route)**.** *Every finite comparison read from an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier exposes the* $\mathsf{StreamName}^{\uparrow}$ *window* T *before* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *sealing is consumed. Consequently common-window comparison is the public route for later regular rational and Real-facing reads, not a consequence of hidden quotient equality or ambient completeness.*

Proof. Start from the pointwise stream projection of Definition 5965.6. It exposes the source row and $\mathsf{StreamName}^{\uparrow}$ schedule before the regular rational readback, dyadic tolerance row, and Real-facing seal are available.

Apply finite-window comparison, Definition 5965.7. Any two displayed values must share the same finite stream window, regular rational readback, tail-filter ledger, dyadic tolerance row, completion handoff, and structural rows already exposed by the carrier.

Finally combine the RegSeqRat Real seal route of Definition 5965.10 with the obligation scope route of Definition 5965.8. The later RegSeqRat and Real rows remain downstream of the exposed

StreamName window, so the comparison route has no hidden quotient equality, selected limit, or ambient completeness witness. □

Theorem 5965.15 (RegularSequenceSpace tail-window transport)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. For every displayed finite tail-depth extension of a comparison read, the same* $\mathsf{StreamName}^{\uparrow}$ *window route transports through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the tail-filter ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal R, and the structural rows H, C, P, N. Thus finite tail-depth transport is a displayed route inside T, Q, D, R, H, C, P, N, not a quotient-stream argument, selected-limit argument, countable-choice argument, or appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed finite tail depth. At depth 0, apply the StreamName window route of Definition 5965.14. The initial comparison read exposes the finite $\mathsf{StreamName}^{\uparrow}$ window before $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ sealing is consumed.

For the successor step, first use the tail-filter handoff of Definition 5965.3. It routes the extended tail read through the same displayed stream schedule, regular rational readback, dyadic tolerance row, and tail-filter surface.

Next apply tail coherence and finite-window comparison, Definition 5965.4 and Definition 5965.7. These keep the successor read on the same finite window and carry Q, D, R, H, C, P, N along the accepted carrier route.

The induction therefore transports every displayed finite tail-depth extension through the same window, readback, tolerance, tail-filter, Real-seal, and structural rows. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, or ambient completeness, no such principle is introduced by the tail-depth step. □

Theorem 5965.16 (RegularSequenceSpace tail-diagonal projection)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any two tail-filter reads of S that feed the same* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer can be projected to one displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window before the value-seal row R is read. The projection stays inside the displayed route through T, Q, D, R, H, C, P, N and does not use quotient stream equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with tail coherence, Definition 5965.4. Reads that agree on the displayed stream schedule and regular rational readback also agree on the dyadic tolerance row and terminal seal route.

Apply finite-window comparison, Definition 5965.7. It forces every value comparison to use the same $\mathsf{StreamName}^{\uparrow}$ finite window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, tail-filter ledger, dyadic tolerance row, completion handoff, and structural rows already exposed by the carrier.

Now combine the RegSeqRat Real seal route and the StreamName window route, Definition 5965.10 and Definition 5965.14. The shared consumer cannot read the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row Q or the $\mathsf{Real}^{\uparrow}$ seal row R before the finite window T is displayed.

Since no carrier coordinate records quotient stream equality, selected limits, countable choice, or ambient Real completeness, the diagonal projection is exactly the common finite-window projection inside the accepted packet. □

Theorem 5965.17 (RegularSequenceSpace finite-density feedback)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. If a consumer receives a* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *row as feedback for S, then the feedback must be replayed through the* $\mathsf{StreamName}^{\uparrow}$ *windows T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal R*

before it is treated as regular-sequence data. The feedback route is therefore the displayed finite-density handoff of chapter 6003 together with H, C, P, N, not an ambient density assertion, selected subsequence, quotient stream equality, or completeness principle.

Proof. Start with the RegularSequenceSpace carrier obligations, Definition 5965.2. They require every accepted space read to expose the source row, StreamName schedule, RegSeqRat readback, dyadic tolerance row, Real seal, and structural rows before any consumer treats the packet as regular-sequence data.

Apply the finite-density sibling route of chapter 6003. Its regular-sequence consumer handoff reads a density net only after the same StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and Real seal have been displayed.

Now replay through the tail-filter handoff, Definition 5965.3. The density feedback reaches the regular sequence packet at the tail-filter boundary rather than as an external dense subset or selected subsequence.

Finally use tail coherence, Definition 5965.4. The structural rows H, C, P, N preserve the same window, readback, tolerance, and Real-seal route, so the finite density row cannot bypass the displayed regular-sequence feedback path. □

Theorem 5965.18 (RegularSequenceSpace tail-modulus projection)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any tail-filter read from S that is later consumed as a* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *or* $\mathsf{CauchyRateSpace}^{\uparrow}$ *budget must project through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the value seal R before the budget is public. The projection exports no selected modulus, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tail-diagonal projection of Definition 5965.16. It forces tail-filter reads feeding a common consumer into one displayed $\mathsf{StreamName}^{\uparrow}$ finite window before the value-seal row is read.

Apply finite-window comparison, Definition 5965.7. The common window still carries the $\mathsf{RegSeqRat}^{\uparrow}$ readback, tail-filter ledger, dyadic tolerance row, completion handoff, and structural rows already exposed by the accepted carrier.

Now use the RegSeqRat Real seal route and the StreamName window route, Definition 5965.10 and Definition 5965.14. They place the regular rational readback Q and the value seal R downstream of the displayed schedule T and tolerance row D.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite route for $\mathsf{CauchyModulusSpace}^{\uparrow}$ and $\mathsf{CauchyRateSpace}^{\uparrow}$ consumers, leaving no coordinate from which a selected modulus, quotient stream equality, countable-choice schedule, or ambient completeness principle could be read. □

Theorem 5965.19 (RegularSequenceSpace tail-filter cofinal witness)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every tail-filter read from S that feeds a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the displayed* $\mathsf{StreamName}^{\uparrow}$ *window T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the* $\mathsf{Real}^{\uparrow}$ *seal R before the consumer-facing row is public. The witness is cofinal only through the displayed finite windows and structural rows H, C, P, N; it does not select a subsequence, identify quotient streams, invoke countable choice, or import ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the tail-modulus projection of Definition 5965.18. It already forces every tail-filter budget read through the displayed schedule T, regular rational readback Q, dyadic tolerance row D, and value seal R before a consumer can inspect the budget.

Project the same rows for $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The StreamName window supplies the finite cofinal index surface, the RegSeqRat row supplies readback, the DyadicRatCore row supplies the tolerance boundary, and the Real seal is opened only after those rows are present.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming keep the witness inside the displayed packet, so no selected subsequence, quotient stream equality, countable-choice schedule, or ambient completeness principle can be read from the cofinal route. □

Theorem 5965.20 (RegularSequenceSpace dyadic interval net refinement)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any consumer that asks for finite dyadic cover refinement must factor through the* $\mathsf{StreamName}^{\uparrow}$ *schedule row T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, the finite-cell route of chapter 5966, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal R. The refinement exports no compactness theorem, selected subsequence, quotient equality of real names, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 5965.1. The only rows available to a refinement consumer are the displayed schedule, readback, tolerance, seal, transport, replay, provenance, and local naming rows.

Use the tail-filter cofinal witness and tail-modulus projection, Definition 5965.19 and Definition 5965.18. They force any tail-facing read through T, Q, D before the value seal is public.

Now pass to chapter 5966. Its finite-cell route supplies only dyadic endpoint arithmetic, cell incidence, StreamName windows, RegSeqRat readback, and a Real seal. Composing the two routes gives the stated refinement while leaving compactness, selected subsequences, quotient equality, and Real completeness outside the carrier. □

Theorem 5965.21 (RegularSequenceSpace common-window completion route)**.** *Let $S = (X, T, Q, D, R, H, C, P, N$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every completion-facing comparison read from S must expose one common* $\mathsf{StreamName}^{\uparrow}$ *window T, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, and the finite-cell route of chapter 5966 before it consumes the terminal* $\mathsf{Real}^{\uparrow}$ *seal R. The route exports a displayed common-window comparison and no quotient stream equality, selected limit, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with the tail-diagonal projection of Definition 5965.16. Any two tail-filter reads feeding the same completion-facing consumer are projected to one displayed $\mathsf{StreamName}^{\uparrow}$ finite window before the value seal is read.

Apply the tail-modulus projection of Definition 5965.18. The common window still carries $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance, and the value-seal route before any modulus or rate budget is public.

Now use the dyadic interval net refinement of Definition 5965.20. The finite-cell route of chapter 5966 is inserted after T, Q, D and before the terminal $\mathsf{Real}^{\uparrow}$ seal, so completion-facing comparisons receive finite dyadic cells rather than a hidden compactness or completion witness.

Finally replay finite-window comparison, Definition 5965.7. It keeps the two compared reads inside the same displayed StreamName, RegSeqRat, DyadicRatCore, DyadicIntervalNet, and Real-seal route. Since the carrier has no coordinate for quotient stream equality, selected limits, countable choice, or ambient Real completeness, the common-window completion route is exhausted by the displayed rows. □

Theorem 5965.22 (RegularSequenceSpace shift closure)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Shifting the displayed packet by any unary finite tail*

preserves the $\mathsf{StreamName}^{\uparrow}$ *schedule row* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route* R, H, C, P, N*. The shifted packet is therefore consumed through the same regular sequence space route and exports no quotient stream equality, selected limit, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Induct on the unary tail shift. In the zero case, apply the tail-filter cofinal witness of Definition 5965.19; the displayed schedule, readback, dyadic tolerance, and Real seal rows are exactly the rows already exposed by the accepted carrier.

For the successor step, first apply finite-window comparison, Definition 5965.7. It keeps the shifted read inside the same finite $\mathsf{StreamName}^{\uparrow}$ window and the same $\mathsf{RegSeqRat}^{\uparrow}$ readback route before any value comparison is public.

Next apply the tail-diagonal projection and tail-modulus projection, Definition 5965.16 and Definition 5965.18. These rows move the successor tail back to a displayed common window and preserve the dyadic tolerance row D before the $\mathsf{Real}^{\uparrow}$ seal can be consumed.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same terminal route after each finite tail step, while the carrier still has no coordinate for quotient stream equality, selected limits, countable choice, or ambient completeness. The induction gives shift closure for every unary finite tail. □

Theorem 5965.23 (RegularSequenceSpace diagonal tail extraction)**.** *Let* $S = (X, T, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. If a consumer displays a finite tail-filter row together with the* $\mathsf{StreamName}^{\uparrow}$ *window* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, and the* $\mathsf{Real}^{\uparrow}$ *seal row* R*, then the consumer admits a diagonal tail route through the same displayed rows. This route is consumable by* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *endpoints and exports no quotient stream equality, selected limit, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Proof. Start with the tail-filter handoff of Definition 5965.3. It places every completion-facing read through the displayed stream schedule, regular rational readback, dyadic tolerance row, and regular Cauchy tail-filter surface before the Real seal is consumed.

Apply the tail-diagonal projection and tail-modulus projection, Definition 5965.16 and Definition 5965.18. The first projects any two tail-filter reads feeding the same regular rational or Real consumer to one displayed finite window; the second keeps that window tied to the displayed modulus and rate budget rows before public consumption.

Finally use shift closure, Definition 5965.22. Every unary finite tail of the packet preserves the same T, Q, D, R, H, C, P, N route, so the diagonal tail extraction remains inside the accepted carrier for both $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ endpoints.

Because the carrier records only these finite rows and their structural replay, the extracted route has no coordinate from which quotient stream equality, a selected limit, countable choice, or ambient Real completeness could be read. □

Theorem 5965.24 (RegularSequenceSpace Cauchy-filter completion route)**.** *Let* $S = (X, T, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every Cauchy-filter or completion-facing read from* S *must expose the* $\mathsf{StreamName}^{\uparrow}$ *window* T*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* Q*, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*, the tail-filter ledger, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *before any completed value is public. The route is a finite L10 completion boundary consumable by* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *descendants; it exports no quotient stream equality, selected limit, countable-choice selector, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the tail-filter handoff of Definition 5965.3. It places any completion-facing read behind the displayed stream schedule, regular rational readback, dyadic tolerance row, and tail-filter ledger.

Apply window comparison and finite-density feedback, Definition 5965.7 and Definition 5965.17. These keep the completion request tied to one finite $\mathsf{StreamName}^{\uparrow}$ window and one $\mathsf{RegSeqRat}^{\uparrow}$ readback surface before the tolerance row is consumed.

Finally use the RegSeqRat Real seal route through Definition 5965.23. The structural rows H, C, P, N preserve the same displayed route, so the public completion read reaches the $\mathsf{Real}^{\uparrow}$ seal only through finite tail data and cannot expose quotient streams, selected limits, countable choice, or ambient Real completeness. □

Theorem 5965.25 (RegularSequenceSpace BishopRealInterval route)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier. If a tail window of S displays two endpoint reads whose endpoint packet is sealed by the $\mathsf{BishopRealInterval}^{\uparrow}$ carrier of chapter 8787, then the same tail window supplies a $\mathsf{Real}^{\uparrow}$-readable interval enclosure route. The route uses RegSeqRat tail coherence, StreamName pointwise projection, and the $\mathsf{BishopRealInterval}^{\uparrow}$ endpoint classifier before the terminal $\mathsf{Real}^{\uparrow}$ seal is public; it exports no quotient stream equality, selected limit, countable choice, classical interval subset, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Start with diagonal tail extraction, Definition 5965.23. It keeps the displayed tail window inside the same T, Q, D, R, H, C, P, N route for every $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ endpoint consumer.

Apply tail coherence and pointwise stream projection, Definition 5965.4 and Definition 5965.6. The endpoint reads therefore share one finite $\mathsf{StreamName}^{\uparrow}$ window and one $\mathsf{RegSeqRat}^{\uparrow}$ readback surface before the dyadic tolerance row and Real seal are consumed.

Now pass to the BishopRealInterval carrier and its regular seal route, Definition 8787.1 and Definition 8787.3. Its endpoint classifier reads finite stream windows, regular rational endpoint readback, and dyadic comparison data before releasing the interval enclosure seal.

Finally replay the RegularSequenceSpace structural rows H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same tail window and endpoint packet, so the resulting enclosure is a Real-readable L10 completion route rather than a quotient-real, selected-limit, classical-interval, countable-choice, or ambient-completeness assertion. □

Theorem 5965.26 (RegularSequenceSpace BishopSequence forward link)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier. Any downstream $\mathsf{BishopSequence}^{\uparrow}$ consumer of S must keep the $\mathsf{StreamName}^{\uparrow}$ window T, the $\mathsf{RegSeqRat}^{\uparrow}$ readback row Q, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, and the terminal $\mathsf{Real}^{\uparrow}$ seal route R, H, C, P, N visible before the Bishop sequence read is public. The resulting forward link feeds the carrier of chapter 2179 through the same tail window and regular readback used by Definition 5965.3, Definition 5965.12, and Definition 5965.25; it exports no quotient stream equality, selected sequence representative, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness principle.*

Proof. Start with the tail-filter handoff of Definition 5965.3. It places every completion-facing read behind the displayed $\mathsf{StreamName}^{\uparrow}$ schedule T, $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D before the terminal seal is public.

Apply the Cauchy-window route of Definition 5965.12. The window subpacket keeps T, Q, D, R, H, C, P, N together as one finite NameCert route rather than as detachable sequence representatives.

Now pass to the $\mathsf{BishopSequence}^{\uparrow}$ carrier of chapter 2179. A Bishop sequence consumer may read the displayed regular sequence space only through the same stream window, regular rational readback, tolerance ledger, and Real seal rows.

Finally use the BishopRealInterval route of Definition 5965.25 as the endpoint model. Since interval-facing reads already preserve the same tail window before public consumption, BishopSequence-facing reads inherit the same finite L10 boundary and cannot add quotient streams, selected representatives, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularSequenceSpace}^{\uparrow}$ packet is generated from a displayed source family of regular sequence names, stream schedules, regular rational readbacks, dyadic tolerance core rows, a real-facing value seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$, obligation)

Scope closed: Definition 5965.1 and Definition 5965.2 seed the finite space-level regular sequence carrier between stream names, rational regular readbacks, dyadic tolerance cores, and Real-facing consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim an ambient complete metric space, quotient equality of real names, arbitrary choice of representatives, unrestricted sequence space, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicit kernel scope for StreamName, RegSeqRat, DyadicRatCore, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

The $\mathsf{RegularSequenceSpace}^{\uparrow}$ route consumes chapter 126, chapter 291, chapter 301, and chapter 38 as its L10 finite-window, tolerance-budget, regular-readback, and terminal-seal siblings before the local sequence-space certificate is exposed. The $\mathsf{RegularSequenceSpace}^{\uparrow}$ route consumes chapter 1119 as its finite-tail filter carrier before the regular-sequence-space schedule and tail rows are inspected. The schedule/tail handoff therefore remains tied to the finite cofinal-tail filter surface rather than to a detached sequence-space island.

Theorem 5965.27 (RegularSequenceSpace tail-filter consumer)**.** *Every* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *tail consumer that reaches* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *must expose the Frechet-filter and cofinite-tail discipline before the regular readback or terminal seal is visible. Concretely, the consumer route factors through chapter 1564, chapter 7023, the* $\mathsf{StreamName}^{\uparrow}$ *tail windows, chapter 301, and chapter 38; after those rows are exposed, the same route records componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{RegularSequenceSpace}^{\uparrow}}$ *rows.*

Thus a RegularSequenceSpace tail read cannot reach a regular rational readback or a real seal from an unrestricted sequence-space carrier, an ambient Frechet-filter quotient, a hidden cofinite-set equality, or a host choice of representatives. It reaches those endpoints only through the displayed $\mathsf{FrechetFilter}^{\uparrow}/\mathsf{CofiniteFilter}^{\uparrow}$ *tail discipline,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows.*

Every $\mathsf{RegularSequenceSpace}^{\uparrow}$ consumer that forms a telescoping or completion budget routes through chapter 708, chapter 291, chapter 301, chapter 126, and chapter 38 before any value-facing seal is read. Thus the local sequence-space packet exposes telescoping budget data only after the finite budget, dyadic tolerance, regular readback, stream schedule, and terminal seal rows remain visible. The $\mathsf{RegularSequenceSpace}^{\uparrow}$ route consumes chapter 5966 as a finite dyadic tolerance-window child before $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal are public. This child route records the dyadic interval net as a genuine L10 sibling dependency of regular sequence finite-window reads, not as a self-contained completion principle.

Proposition 5965.28 (RegularSequenceSpace dyadic net dependency)**.** *Every accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier read by a completion-facing consumer must expose its* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the finite-cell route of chapter 5966, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and only then the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The dependency exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem.*

Proof. Project the displayed regular-sequence carrier rows. The schedule, dyadic tolerance, readback, seal, transport, replay, provenance, and local naming coordinates are the only rows available to a completion-facing consumer.

Insert the dyadic interval net after the tolerance row and before readback reaches the terminal value seal. The finite-cell route supplies dyadic endpoint and incidence data for the visible tolerance window while leaving RegSeqRat and Real readback downstream.

The route therefore records a sibling dependency through finite rows. Since no carrier coordinate contains a quotient of streams, a selected limit, a choice schedule, host real equality, or ambient completeness data, none of those endpoints can be exported by the dependency. □

Every $\mathsf{RegularSequenceSpace}^{\uparrow}$ completion-facing consumer reads the terminal value route through the finite L10 dependency chain before treating the packet as a Real or Completion input. The route first exposes the $\mathsf{StreamName}^{\uparrow}$ finite-window rows of chapter 126, then the $\mathsf{RegSeqRat}^{\uparrow}$ readback rows of chapter 301, then the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows of chapter 291, and finally the $\mathsf{Real}^{\uparrow}$ sealing boundary of chapter 38. The finite-tail and tail-filter rows already named in this hub remain upstream of any completion-facing read.

Theorem 5965.29 (RegularSequenceSpace Real and Completion dependency route)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier. Any consumer that exports a Real-facing or Completion-facing read from S must first display the $\mathsf{StreamName}^{\uparrow}$ window T, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, the existing finite-tail and tail-filter dependency rows, and then the terminal $\mathsf{Real}^{\uparrow}$ seal R. The export is therefore a displayed Real and Completion route through the regular-sequence-space packet, not an ambient completeness witness, quotient stream equality, selected limit, or unrestricted sequence-space object.*

Proof. Project the carrier of Definition 5965.1. The only value-facing rows available to a completion consumer are the stream schedule T, regular rational readback Q, dyadic tolerance row D, and Real seal R.

Use the finite-tail and tail-filter dependency routes already included by the hub. They keep the tail read inside displayed finite windows before the regular rational and Real rows are consumed.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite packet, so the Real and Completion export cannot import quotient stream equality, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5965.30 (RegularSequenceSpace sibling route)**.** *Every $\mathsf{RegularSequenceSpace}^{\uparrow}$ consumer that reads completion-facing data must first consume the sibling route through $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, and $\mathsf{Real}^{\uparrow}$ seal rows. The route passes through chapter 126, chapter 301, chapter 291, and chapter 38 before the regular-sequence-space packet exposes value-facing data to any downstream consumer.*

Proof. Project an accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier through the local carrier row. Its displayed schedule, readback, tolerance, and seal coordinates are the rows T, Q, D, R already named by the chapter.

The sibling chapters determine the order of consumption. The StreamName window T is read before the RegSeqRat readback Q; the DyadicRatCore tolerance row D bounds the readback; and the Real seal R is terminal for value-facing output.

Thus every completion-facing consumer reaches regular-sequence-space data only after the sibling route has been displayed. The carrier has no coordinate for quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness that could bypass this route. □

Theorem 5965.31 (RegularSequenceSpace finite shift stability)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every finite shift of the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule transports through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D, tail-filter ledger, terminal* $\mathsf{Real}^{\uparrow}$ *seal R, and structural rows H, C, P, N. Thus finite shift stability is a displayed route inside the regular sequence space packet, not a quotient-stream argument, selected-limit argument, countable-choice argument, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use finite tail-depth transport, Definition 5965.15. A finite shift of the schedule is a finite tail-depth extension of the displayed $\mathsf{StreamName}^{\uparrow}$ window, so the read remains on the same T, Q, D, R, H, C, P, N route.

Apply tail-diagonal projection, Definition 5965.16. Any two shifted tail-filter reads feeding the same regular rational or Real consumer are projected back to one displayed StreamName window before the value-seal row is public.

Finally use the RegSeqRat Real seal route of Definition 5965.10. The shifted read reaches $\mathsf{Real}^{\uparrow}$ only after the $\mathsf{RegSeqRat}^{\uparrow}$ classifier row and the visible structural rows have stayed attached to the accepted packet. □

Theorem 5965.32 (RegularSequenceSpace product-window route)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every finite product, paired-tail, or product-completion consumer of S must expose the two displayed* $\mathsf{StreamName}^{\uparrow}$ *finite windows, the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seals before any product completion read is public. The route exports only the common finite-window product surface of the accepted packet; it exports no quotient stream equality, selected representative pair, countable-choice selector, or ambient completeness coordinate.*

Proof. Project the carrier of Definition 5965.1. A product-facing read has access only to the displayed schedule row T, regular rational readback Q, dyadic tolerance row D, Real seal R, and structural rows H, C, P, N.

Use the sibling and tail routes already attached to the hub: Definition 5965.30, Definition 5965.3, and Definition 5965.21. They force each coordinate of a paired-tail read to pass through a displayed StreamName window before the RegSeqRat, DyadicRatCore, and Real rows are consumed.

Use the shift-stability route of Definition 5965.22. Finite shifts of either coordinate remain inside the same T, Q, D, R, H, C, P, N route, so a product window is assembled from two finite windows rather than from a selected representative pair.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the paired finite route and supply no coordinate for quotient stream equality, countable choice, selected representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5965.33 (RegularSequenceSpace hub forward link)**.** *Every* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *packet exposed by the hub routes a completion-facing read through at least one substantive L10 sibling before the terminal value seal is public. Concretely, a consumer must keep the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite-cell route,* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *uniqueness boundary, and terminal* $\mathsf{Real}^{\uparrow}$ *seal visible before treating the packet as a completion input. The hub therefore exports a forward lattice edge to existing L10 completion siblings, not a kernel-only carrier island, quotient stream equality, selected limit, countable-choice schedule, or ambient completeness principle.*

Proof. Start with the sibling route of Definition 5965.30. It forces every completion-facing read through the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before the local packet exports value-facing data.

Use the common-window completion route of Definition 5965.21. The same read must expose the finite-cell route of chapter 5966 after the common window and before the terminal seal, so the hub points to a concrete L10 finite-net sibling.

Finally route uniqueness through chapter 5983. The structural rows H, C, P, N preserve the displayed window, readback, finite-cell, uniqueness, and Real-seal route, and no coordinate in the carrier adds quotient stream equality, selected limits, countable choice, or ambient completeness. □

Theorem 5965.34 (RegularSequenceSpace RegSeqRat frontier)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Any Real-completion consumer of S reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule T,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity rows Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance surface D, and* $\mathsf{Real}^{\uparrow}$ *seal R, with transport, replay, provenance, and local naming rows H, C, P, N still visible. The packet exports no quotient of sequences, choice-selected subsequence, hidden representative selector, or ambient Real completeness principle.*

Proof. Project the carrier of Definition 5965.1. The value-facing route exposes the schedule row T, the regular rational readback row Q, the dyadic tolerance row D, the terminal seal R, and the structural rows H, C, P, N.

Use the sibling route of Definition 5965.30. It forces completion-facing consumers to read the StreamName, RegSeqRat, DyadicRatCore, and Real rows in that order before any value-facing data are public.

Next use the Real and Completion dependency route of Definition 5965.29. It keeps the finite-tail and tail-filter rows upstream of the same T, Q, D, R route and blocks any detour through quotient stream equality or selected limits.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the displayed frontier and add no coordinate for sequence quotients, choice-selected subsequences, hidden representatives, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the RegSeqRat frontier is the only route from the accepted packet to Real-completion consumers. □

Theorem 5965.35 (RegularSequenceSpace RegSeqRat frontier tail seal)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. A* $\mathsf{Real}^{\uparrow}$*-completion consumer must expose the finite-tail row, tail-filter row,* $\mathsf{StreamName}^{\uparrow}$ *schedule T,* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row Q,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row D,* $\mathsf{Real}^{\uparrow}$ *seal R, componentwise transport H, continuation replay C, provenance P, and local naming row N before reading the packet as a completed regular sequence. The tail seal is therefore a displayed RegSeqRat frontier obligation and not a quotient stream, selected representative, choice-selected subsequence, host real equality, or ambient completeness principle.*

Proof. Begin with the RegSeqRat frontier theorem, Definition 5965.34. It places T, Q, D, R, H, C, P, N on the only value-facing route available to Real-completion consumers.

Use the Real and Completion dependency route of Definition 5965.29. It keeps the finite-tail and tail-filter rows upstream of the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ route.

Apply the sibling route theorem, Definition 5965.30. It prevents a completion-facing consumer from skipping the displayed stream schedule, regular rational readback, dyadic tolerance row, terminal seal, transport, replay, provenance, and local naming rows.

Thus the accepted packet can be read as a completed regular sequence only after the tail seal has exposed the finite-tail and tail-filter rows together with T, Q, D, R, H, C, P, N. Since no carrier coordinate stores quotient stream equality, selected representatives, choice-selected subsequences,

host real equality, or ambient completeness, the seal is exactly the displayed RegSeqRat frontier boundary. □

Theorem 5965.36 (RegularSequenceSpace L10 link)**.** *Let $S = (X, T, Q, D, R, H, C, P, N)$ be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every completion-facing read from S factors through the displayed L10 chain*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow},$$

with transport, replay, provenance, and local naming rows H, C, P, N preserved along the route. The chain cannot import quotient stream equality, a selected representative, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the RegSeqRat frontier theorem, Definition 5965.34. It places the schedule row T, regular rational row Q, dyadic tolerance row D, Real seal R, and structural rows H, C, P, N on the only completion-facing route exposed by the carrier.

Use the frontier tail seal, Definition 5965.35. It adds the finite-tail and tail-filter rows before the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ chain can be read as a completed regular sequence.

Now apply the carrier and NameCert obligations of Definition 5965.1 and Definition 5965.2. They keep H, C, P, N attached to the displayed packet by componentwise transport, continuation replay, provenance, and local naming.

Since none of these rows contains quotient stream equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every accepted completion-facing read remains inside the displayed L10 chain. □

The $\mathsf{RegularSequenceSpace}^{\uparrow}$ lattice route consumes the finite scheduling surface of chapter 126 together with the regular readback surface of chapter 301. The StreamName row fixes the visible observation order, while the RegSeqRat row reads those observations as regular rational data before the local sequence-space certificate is exposed.

Proposition 5965.37 (RegularSequenceSpace RegSeqRat StreamName Real route)**.** *Every* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *completion-facing read must expose the finite* $\mathsf{StreamName}^{\uparrow}$ *schedule of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback surface of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before the local packet is inspected as a completion input. The route exports only the displayed finite rows of the accepted sequence-space carrier, not quotient stream equality, selected representatives, countable choice, unrestricted sequence-space completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted $\mathsf{RegularSequenceSpace}^{\uparrow}$ carrier through Definition 5965.1. Its visible schedule, readback, and value-facing rows are the displayed T, Q, R coordinates, with transport, replay, provenance, and local naming still attached.

Use the sibling route of Definition 5965.30. It already orders the finite StreamName window before the RegSeqRat readback and keeps the Real seal terminal for value-facing output.

Finally apply the Real and Completion dependency route of Definition 5965.29. A completion-facing consumer therefore reaches the local packet only through the StreamName, RegSeqRat, and Real sibling chapters, with no coordinate available for quotient stream equality, selected representatives, countable choice, unrestricted sequence-space completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Definition 5965.38 (RegularSequenceSpace density route)**.** *The* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *density route is the finite dependency corridor in which an accepted carrier*

$$S = (X, T, Q, D, R, H, C, P, N)$$

is read first through the $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback of chapter 301, then through the density-facing* $\mathsf{RealSeparability}^{\uparrow}$ *surface of chapter 1809. The route exposes only the displayed source row* X*, stream schedule* T*, regular readback* Q*, dyadic tolerance row* D*, terminal* $\mathsf{Real}^{\uparrow}$ *seal* R*, componentwise transport* H*, replay* C*, provenance* P*, and local naming row* N*.*

Theorem 5965.39 (RegularSequenceSpace RegSeqRat density consumption)**.** *Every* $\mathsf{RealSeparability}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier must consume the density route of Definition 5965.38. In particular, the consumer reaches chapter 1809 only after the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row remain visible in the same packet.*

Proof. Project the accepted carrier through Definition 5965.1. The available rows are exactly $X, T, Q, D, R, H, C, P, N$, with Q carrying the regular rational readback and R carrying the terminal Real-facing seal.

Use the RegSeqRat frontier of Definition 5965.34. It forces completion-facing consumers to read the displayed stream schedule, regular rational readback, dyadic tolerance row, and Real seal before a value-facing endpoint is public.

Now route density-facing reads through chapter 1809. That chapter consumes rational density only after the RegSeqRat and Real-seal rows are visible, so the $\mathsf{RegularSequenceSpace}^{\uparrow}$ consumer cannot bypass T, Q, D, R.

Finally H, C, P, N preserve the same packet by transport, replay, provenance, and local naming. Since the carrier has no coordinate for quotient stream equality, selected dense sequence, countable-choice schedule, unrestricted sequence-space completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness, no density consumer can export those objects from the route. □

Theorem 5965.40 (RegularSequenceSpace diagonal-adjoint handoff)**.** *Let* $S = (X, T, Q, D, R, H, C, P, N)$ *be an accepted* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *carrier. Every finite diagonal window over* S *that is consumed by a completion-facing endpoint projects through the displayed route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

without changing the dyadic tail budget D*. Equivalently, the diagonal read first fixes one* $\mathsf{StreamName}^{\uparrow}$ *window* T*, transports its* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* Q *by componentwise* hsame*, and then exposes the terminal* $\mathsf{Real}^{\uparrow}$ *seal* R *over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row.*

Proof. Induct on the displayed finite diagonal window. In the empty window, the carrier projection of Definition 5965.1 already exposes T, Q, D, R, H, C, P, N as one route, so the tail budget is exactly the displayed row D.

For the successor window, apply the diagonal tail extraction theorem, Definition 5965.23. It keeps the new diagonal read inside the same $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and $\mathsf{Real}^{\uparrow}$ seal route.

Now use shift closure, Definition 5965.22. The finite tail shift preserves T, Q, D before the terminal seal is consumed, so the successor step cannot tighten, weaken, or replace the dyadic budget row.

Finally replay the structural rows. Componentwise hsame transport H, continuation replay C, provenance P, and local naming N carry the same diagonal-adjoint handoff to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ consumers. The route has no coordinate for quotient stream equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The $\mathsf{RegularSequenceSpace}^{\uparrow}$ route consumes chapter 8972 whenever a regular sequence is read with an explicit dyadic modulus budget. The packet is reached through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and $\mathsf{StreamName}^{\uparrow}$ scheduling routes of Definition 5965.30 and Definition 5965.37, so the local sequence-space carrier exposes the modulus budget as finite sibling data rather than as quotient selection, countable choice, or ambient completion structure. The $\mathsf{RegularSequenceSpace}^{\uparrow}$ sibling route consumes the $\mathsf{RegSeqRat}^{\uparrow}$ stream readback of chapter 301 together with the finite Cauchy density-net handoff of chapter 6003 before exporting sequence-space reads. Thus a completion-facing consumer sees the finite density row only after the displayed StreamName window, RegSeqRat readback, dyadic tolerance ledger, and terminal Real seal remain attached to the accepted regular-sequence-space packet. The $\mathsf{DyadicIntervalNet}^{\uparrow}$ hub routes finite dyadic mesh data through chapter 1767, chapter 1808, and chapter 291 before Real-completion consumers read the mesh.

Chapter 5966

Dyadic Interval Net NameCert Construction

Following Bishop and Bridges, the $\mathsf{DyadicIntervalNet}^{\uparrow}$ packet records finite dyadic interval nets as the displayed cover primitive used by Real and Completion arguments before any consumer may invoke compactness language. It sits after the dyadic arithmetic core of chapter 291, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. As a consumer route, the finite interval-cell row consumes dyadic endpoint arithmetic from chapter 291, exposes endpoint observations through chapter 126, reads tolerances through chapter 301, and seals only through chapter 38. The packet gives compact-interval, completion, and uniform-modulus consumers a finite cover surface; it does not assert open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The same finite dyadic cell data is consumed by chapter 1250 and by the compact-interval uniform-modulus route as displayed finite cells, not as open-cover compactness. The finite Cauchy density-net route of chapter 6003 consumes these dyadic cell rows as finite density witnesses for L10 completion reads.

Definition 5966.1 (Dyadic interval net carrier)**.** *A* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

with the following rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row. It lists the dyadic endpoints, midpoint and radius arithmetic, and successor comparisons used by the net cells.*
2. *J is the finite interval-cell row. It records a finite list of closed dyadic cells, each with displayed left and right endpoints and a cell radius.*
3. *E is the endpoint-incidence row. It records that adjacent cells overlap or touch at displayed dyadic endpoints and that the first and last cells bracket the named interval endpoints.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed when an interval endpoint or cell endpoint is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the dyadic endpoints and the tolerance schedule induced by the cell radii.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed StreamName and RegSeqRat rows have been inspected.*

7. *H records componentwise* hsame *transport for D, J, E, S, R, Q and the structural rows.*
8. *C records displayed* Cont *replay of the finite route*

$$D, J, E \longmapsto S, R \longmapsto Q, H, P, N.$$

9. *P is the* Pkg *provenance over* DyadicIntervalNet$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{DyadicIntervalNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, E, S, R, Q, H, C, P, N$. It has no coordinate for an arbitrary open cover, a selected finite subcover, quotient stream equality, a chosen limit point, countable choice, trichotomy, or completed real compactness.

Theorem 5966.2 (Dyadic interval net cover refinement)**.** *For every accepted* DyadicIntervalNet$^{\uparrow}$ *carrier $I = (D, J, E, S, R, Q, H, C, P, N)$, every Real and Completion consumer that reads a finite dyadic cover through I factors through the finite route*

$$D, J, E \longmapsto S, R \longmapsto Q.$$

The interval-cell row J and endpoint-incidence row E refine any displayed dyadic cover request into listed dyadic cells whose endpoints and radii are checked in D, whose inspected name coordinates are exposed in S, whose tolerance schedule is read in R, and whose Real boundary is sealed in Q. No open-cover compactness, selected subcover, quotient stream equality, countable choice, trichotomy, host real equality, or ambient Real$^{\uparrow}$ *completeness is available along this route.*

Lean4 ✓ `BEDC.Derived.Dyadic interval netUp.DyadicIntervalNetCoverRefinement`

Proof. Project the accepted carrier of Definition 5966.1. The rows D, J, E are the only cover-facing data: D supplies dyadic endpoint arithmetic, J supplies the finite list of cells, and E supplies endpoint incidence and bracketing.

The refinement is therefore a finite list operation. A displayed cover request is answered by reading the listed cells in J, checking their dyadic radii and endpoint comparisons in D, and using E to verify that adjacent cells cover the named interval without opening a host compactness argument.

The Real-facing portion is delayed. The inspected endpoint names pass through S, the induced tolerance schedule passes through R, and the terminal real-name boundary is exposed only by Q.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve exactly the same finite route, so the consumer cannot replace the dyadic net by a selected subcover, quotient stream equality, choice, trichotomy, host equality, or ambient completeness principle. □

Theorem 5966.3 (Dyadic interval net uniform subnet)**.** *Every compact-uniform-continuity finite-cover read that consumes an accepted* DyadicIntervalNet$^{\uparrow}$ *packet must expose dyadic cell rows, interval-radius rows,* StreamName$^{\uparrow}$ *or* RegSeqRat$^{\uparrow}$ *window rows, and a* UniformModulus$^{\uparrow}$ *handoff before the compact finite-net conclusion is read. In carrier notation, the read begins with the endpoint arithmetic and cell rows D, J, E, then passes through the window and readback rows S, R, and only then may a compact finite-net or uniform-modulus consumer inspect the terminal Real-facing seal Q and the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 5966.1. The dyadic endpoint row D, finite interval-cell row J, and endpoint-incidence row E are the only rows that can display cell membership, adjacency, and interval radii.

Apply the cover refinement theorem Definition 5966.2. It forces every finite cover read to consume those dyadic rows before StreamName, RegSeqRat, or Real-facing data are visible.

The compact-uniform-continuity consumer then reads the finite-window route already named by this chapter. Endpoint observations pass through S, tolerance and radius readback pass through R, and the compact-interval uniform-modulus route can receive a $\mathsf{UniformModulus}^{\uparrow}$ handoff only after those rows have been exposed.

Finally replay H, C, P, N together with the sibling route paragraphs to chapter 1250 and chapter 6207. The resulting finite-subnet branch is a displayed dyadic-cell route, not open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness. □

Theorem 5966.4 (Dyadic interval net mesh-refinement induction)**.** *For every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

and every displayed finite dyadic refinement depth n, repeated dyadic bisection of the interval-cell row preserves endpoint incidence, finite-cell coverage, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route. Each refined packet remains routed through*

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N,$$

and every public read still occurs before the Real seal is opened.

Proof. Induct on the displayed finite refinement depth n. At depth zero, the accepted carrier itself supplies the endpoint incidence, coverage, StreamName windows, dyadic tolerances, RegSeqRat readback, and Real seal route.

For the step, apply Definition 5966.2 to the current refined packet. Bisection replaces each listed dyadic cell by finitely many listed subcells whose endpoints, radii, and adjacency checks are still read through the dyadic endpoint row D and the incidence row E.

The consumer-facing rows remain downstream. Endpoint observations pass through S, tolerance schedules and regular rational reads pass through R, and the terminal real-name boundary remains Q.

Finally transport, replay, provenance, and local naming preserve the refined route componentwise. Since every induction step displays its finite cells before Q, no open-cover compactness, selected subcover, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness principle enters the induction. □

Theorem 5966.5 (Dyadic interval net finite-bisection induction route)**.** *For every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

and every displayed finite bisection depth n, the row inventory

$$D, J, E, S, R, Q, H, C, P, N$$

is preserved by induction on n. Each bisection step replaces a listed parent cell in J by finitely displayed child cells whose endpoints, incidences, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming remain routed through the same rows. Thus

Real, Completion, closed-bounded interval, compact-interval, and uniform-modulus consumers receive a finite bisection route, not open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Induct on the displayed bisection depth n. At depth zero, the carrier of Definition 5966.1 already displays the endpoint arithmetic row D, finite cell row J, endpoint incidence row E, window row S, readback row R, Real seal row Q, and structural rows H, C, P, N.

For the successor step, apply the cover refinement theorem Definition 5966.2. The parent cell is replaced by finitely many listed child cells, and their endpoints, radii, and adjacency checks are still read through D, J, E before any Real-facing boundary is visible.

Now use mesh-refinement induction from Definition 5966.4. It transports the refined finite-cell packet across the displayed bisection step and keeps the endpoint windows, tolerance schedule, and terminal Real seal downstream of the child cells.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same row inventory at the new depth, so the induction never imports open-cover compactness, selected subcovers, quotient equality, choice, trichotomy, host equality, or ambient completeness. □

Theorem 5966.6 (Dyadic interval net subdivision-depth coverage)**.** *For every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

and every displayed finite subdivision depth n, each refined cell coverage read factors through the $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic row D, the finite cell list J, endpoint incidence E,* $\mathsf{StreamName}^{\uparrow}$ *windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback R, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal Q before any compact-interval or uniform-modulus consumer may inspect it. The structural rows H, C, P, N preserve the same route at every depth, so subdivision coverage remains finite dyadic interval-net data.*

Proof. Induct on the displayed depth n. At depth zero, the claim is the accepted carrier route: the original listed cells are read through D, J, E, endpoint observations pass through S, tolerance readback passes through R, and the Real-facing boundary is Q.

For the successor depth, apply the cover refinement theorem Definition 5966.2. Each bisection replaces a parent cell by finitely displayed child cells whose endpoints, radii, and adjacency checks still live in the dyadic endpoint and incidence rows.

Now use mesh-refinement induction from Definition 5966.4. It transports the finite coverage route across the displayed bisection step and keeps the endpoint windows, tolerance rows, and Real seal downstream of the refined cells.

Finally replay H, C, P, N. Componentwise transport, displayed continuation replay, provenance, and local naming preserve the same route for compact-interval and uniform-modulus consumers, without adding open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness. □

The induction step consumes the finite subdivision packet of chapter 7480: each bisection reads its dyadic scale, grid, and cell-coverage rows before the refined interval-net cells are handed back to D, J, E.

The same finite-cell surface is consumed by the compact interval uniform-modulus route of chapter 6207. A downstream consumer reads the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, the $\mathsf{StreamName}^{\uparrow}$ window row, the $\mathsf{RegSeqRat}^{\uparrow}$ tolerance row, and the terminal $\mathsf{Real}^{\uparrow}$ seal in the displayed order, so the interval net remains a finite cover handoff rather than an open-cover compactness principle.

Definition 5966.7 (Dyadic compact-modulus carrier)**.** *A dyadic compact-modulus carrier over an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet* $I = (D, J, E, S, R, Q, H, C, P, N)$ *is a finite* BHist *packet*

$$M = (I, K, B, U, L, S', R', Q', T, V, W, N')$$

with the following rows:

1. *K is the compact-interval source row, read only through the finite dyadic cells already listed in J and their endpoint-incidence data E.*
2. *B is the mesh-bound row, recording a displayed dyadic scale and the rational lower-bound fold used to compare cell radii.*
3. *U is the pointwise modulus row, listing the local modulus data attached to the displayed cells rather than a selected global modulus.*
4. *L is the finite lower-bound fold over the listed dyadic cells, producing the public compact-modulus budget from B and U.*
5. *S' and R' are the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows for endpoint observations, mesh bounds, and tolerance budgets.*
6. *Q' is the* $\mathsf{Real}^{\uparrow}$ *seal row for the compact-interval endpoint and value data, reached only after S' and R' are displayed.*
7. *T records componentwise* hsame *transport for $I, K, B, U, L, S', R', Q'$ and the structural rows.*
8. *V records displayed* Cont *replay from dyadic cells through mesh bounds, pointwise modulus rows, lower-bound fold, readback, and seal.*
9. *W is the* Pkg *provenance over* $\mathsf{DyadicIntervalNet}^{\uparrow}$*, compact-interval source rows, uniform-modulus consumers,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. *N' is the local dyadic compact-modulus* $\mathsf{NameCert}$ *row.*

The classifier compares accepted compact-modulus packets only through $I, K, B, U, L, S', R', Q', T, V, W, N'$. It has no coordinate for open-cover compactness, a selected finite subcover, a selected global modulus, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5966.8 (Dyadic compact-modulus NameCert obligations)**.** *For every accepted dyadic compact-modulus carrier $M = (I, K, B, U, L, S', R', Q', T, V, W, N')$, the local NameCert surface consists exactly of carrier admission for those twelve rows; finite-cell exposure through the accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet I; compact-interval source exposure through K; mesh-bound exposure through B; pointwise modulus exposure through U; rational lower-bound folding through L;* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through S' and R'; terminal* $\mathsf{Real}^{\uparrow}$ *sealing through Q'; componentwise* hsame *transport through T; displayed* Cont *replay through V;* Pkg *provenance through W; and local naming through N'.*

Proof. Project the carrier of Definition 5966.7. Its only cover-facing input is the accepted intervalnet packet I, and its only compact-interval input is the source row K read through the finite cells of I.

The modulus budget is finite. The mesh-bound row B records the dyadic scale, U lists pointwise modulus rows attached to displayed cells, and L folds the rational lower bounds over the finite cell list.

The readback and seal rows remain downstream. Endpoint observations and tolerance budgets pass through S' and R', and the terminal real boundary is Q'.

Finally replay T, V, W, N'. Transport, continuation replay, provenance, and local naming preserve exactly those rows, so no open cover, selected finite subcover, selected global modulus, quotient interval equality, countable choice, or ambient completeness row can enter the NameCert surface. □

Theorem 5966.9 (Dyadic compact-modulus Heine–Cantor route)**.** *Every compact-interval uniform-modulus or Heine–Cantor finite-net consumer of an accepted dyadic compact-modulus carrier $M = (I, K, B, U, L, S', R', Q', T, V, W, N')$ must read the finite route*

$$I, K \longmapsto B, U, L \longmapsto S', R', Q', T, V, W, N'$$

before it may use the public compact-modulus budget. The route sends dyadic cells, mesh bounds, endpoint incidence, compact-interval source data, pointwise modulus rows, rational lower-bound fold, $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *sealing to compact-interval uniform-modulus consumers. It exports no open-cover compactness, selected subcover, quotient interval equality, countable choice, selected global modulus, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the dyadic interval net uniform subnet theorem, Definition 5966.3. It forces compact-uniform-continuity reads to expose dyadic cell rows, interval-radius rows, StreamName or RegSeqRat windows, and a uniform-modulus handoff before a compact finite-net conclusion is read.

Apply the compact-modulus obligations of Definition 5966.8. They refine that handoff through the compact-interval source row K, mesh-bound row B, pointwise modulus row U, and finite lower-bound fold L.

Now use the Real source link and Real seal nonescape theorems, Definition 5966.14 and Definition 5966.13. The endpoint observations and tolerance budgets must pass through the displayed StreamName and RegSeqRat readback rows before the Real seal is opened.

Finally replay T, V, W, N'. The compact-interval uniform-modulus and Heine–Cantor consumers receive the finite dyadic-cell route just described, with no coordinate for open-cover compactness, selected subcovers, quotient interval equality, countable-choice schedules, selected global moduli, or ambient Real completeness. □

Theorem 5966.10 (Dyadic interval net obligation surface)**.** *For every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier $I = (D, J, E, S, R, Q, H, C, P, N)$, carrier admission and cell coverage are exhausted by the displayed rows D, J, E, H, C, P, N. The endpoint arithmetic, finite interval-cell list, endpoint incidence, componentwise transport, replay, provenance, and local naming rows supply the complete finite obligation surface for a dyadic net request, and no open-cover compactness or selected finite subcover is exported.*

Proof. Project the carrier of Definition 5966.1. The rows D, J, E display the endpoint arithmetic, listed cells, and incidence witnesses, while H, C, P, N transport, replay, source, and name those rows.

Cell coverage is therefore a finite row check. Each requested interval read is matched with cells already listed in J, the endpoint comparisons already listed in D, and the bracketing and adjacency data already listed in E.

The structural rows do not add a second cover object. They preserve the displayed packet componentwise, so no open cover, selected subcover, choice principle, or ambient compactness row can enter the obligation surface. □

Theorem 5966.11 (Dyadic interval net RegSeqRat route)**.** *Every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or Real-facing route first exposes the dyadic interval endpoints, finite net cells,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. Concretely, the route reads the endpoint arithmetic and cell rows* D, J, E*, then the window row* S *of chapter 126, then the tolerance and readback rows supplied by chapter 291 and chapter 301, and only then the Real seal* Q*. No quotient stream, selected limit, selected subcover, countable-choice schedule, or ambient completeness coordinate is available to the consumer.*

Proof. Project the accepted carrier of Definition 5966.1. The rows D, J, E are the finite interval-net rows: dyadic endpoints, listed cells, and endpoint incidence.

Use the NameCert obligation surface of Definition 5966.10. It permits a downstream consumer to inspect only the displayed carrier rows, so any RegSeqRat-facing request must first pass through the listed cells and the dyadic endpoint arithmetic already present in D, J, E.

The readback rows are then forced in order. The StreamName row S supplies finite observation windows, the DyadicRatCore tolerance information supplies the dyadic radius and comparison rows, and the RegSeqRat row R supplies the regular rational readback before Q can be opened.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same route, leaving no coordinate from which a quotient stream, selected limit, selected subcover, choice schedule, or ambient completeness principle could be read. □

Theorem 5966.12 (Dyadic interval net StreamName and RegSeqRat handoff)**.** *Every downstream consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier that inspects interval endpoints or cell tolerances must factor the read through*

$$D, J, E \longmapsto S, R \longmapsto H, C, P, N.$$

The $\mathsf{StreamName}^{\uparrow}$ *row* S *exposes the finite endpoint windows and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R *exposes the rational tolerance readback; neither row may be skipped or replaced by quotient stream equality, selected limits, or countable choice.*

Proof. Start from a downstream endpoint or tolerance read. By the carrier definition, the only cover-facing rows available before the read are D, J, E, so the requested endpoint or cell radius is first located inside the displayed dyadic net.

The read then passes through S and R. The StreamName row supplies finite observation windows for the endpoint names, and the RegSeqRat row supplies the regular rational schedule attached to those observations.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming keep the same finite handoff visible to the consumer, excluding quotient stream equality, selected limits, and choice. □

Theorem 5966.13 (Dyadic interval net Real seal nonescape)**.** *For every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier* $I = (D, J, E, S, R, Q, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* Q *is reachable only after the finite dyadic net route*

$$D, J, E \longmapsto S, R$$

has been displayed. Any Real or Completion consumer that reaches Q *remains inside the same routed packet* $D, J, E, S, R, Q, H, C, P, N$ *and cannot escape to host real equality, real trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, selected subcovers, quotient equality, selected limits, or countable choice.*

Proof. Project the accepted packet. The carrier places Q downstream of the endpoint arithmetic, finite cells, endpoint incidence, StreamName windows, and RegSeqRat readback rows.

Thus a Real or Completion consumer must first traverse D, J, E, S, R. These rows display the cover cells, endpoint reads, and tolerance schedule before any Real-facing boundary is available.

Once Q is reached, the structural rows H, C, P, N only preserve the same displayed route. Since no excluded equality, selection, trichotomy, choice, or completeness row is present in the carrier, the Real seal cannot be used as an escape from the finite dyadic interval-net packet. □

Theorem 5966.14 (Dyadic interval net Real source link)**.** *Every Real-facing finite-net consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier* $I = (D, J, E, S, R, Q, H, C, P, N)$ *must read the dyadic endpoint arithmetic row* D*, the finite cell row* J*, the endpoint-incidence row* E*, the* $\mathsf{StreamName}^{\uparrow}$ *window row* S*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* Q *in that displayed order before it can hand the packet to closed-bounded interval, compact-interval, or uniform-modulus consumers. The source link exports only the routed packet* $D, J, E, S, R, Q, H, C, P, N$*; it exports no selected subcover, quotient stream equality, host real equality, countable-choice schedule, trichotomy, or ambient completeness principle.*

Proof. Begin with the cover refinement theorem Definition 5966.2. It places every finite-net read at the dyadic endpoint, cell, and incidence rows D, J, E before any Real-facing boundary is visible.

Next use the RegSeqRat route and the StreamName and RegSeqRat handoff, Definition 5966.11 and Definition 5966.12. These force endpoint observations through S and rational readback through R, so the consumer cannot skip directly from dyadic cells to a terminal real object.

Finally apply the Real seal nonescape theorem Definition 5966.13. Once Q is reached, the structural rows H, C, P, N preserve the same displayed packet, giving closed-bounded interval, compact-interval, and uniform-modulus consumers a finite source link and no ambient compactness or choice endpoint. □

Theorem 5966.15 (Dyadic interval net barycentric refinement consumer)**.** *Let* $I = (D, J, E, S, R, Q, H, C, P, N)$ *be an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier. If a displayed* $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ *packet refines the parent cells in* J *by finite barycentric child cells, then every downstream consumer of the refined net still reads endpoint arithmetic through* $\mathsf{DyadicRatCore}^{\uparrow}$*, finite endpoint windows through* $\mathsf{StreamName}^{\uparrow}$*, tolerance readback through* $\mathsf{RegSeqRat}^{\uparrow}$*, and the terminal seal through* $\mathsf{Real}^{\uparrow}$ *before any compact-interval or completion consumer inspects it. The refined route remains a finite L10 route through*

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$$

together with the sibling refinement rows of chapter 8605; it exports no open-cover compactness, selected subcover, quotient stream equality, countable-choice schedule, trichotomy, host real equality, or ambient completeness principle.

Proof. Start from the cover refinement theorem Definition 5966.2. The parent cells and endpoint-incidence rows already force the consumer to read the finite cell ledger J, E and the endpoint arithmetic row D before any Real-facing boundary is visible.

Use mesh-refinement induction from Definition 5966.4. A barycentric child-cell packet is a displayed finite refinement of the same kind: each child endpoint and mesh witness is listed before the StreamName and RegSeqRat rows are opened.

Now apply the Real source link Definition 5966.14 and the sibling $\mathsf{DyadicBarycentricRefinement}^{\uparrow}$ chapter. The sibling packet supplies the parent pointers, barycenter rows, and child-cell ledger, while this chapter supplies the endpoint, window, readback, and Real-seal order consumed by compact-interval and completion arguments.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming keep the refined packet on the same finite L10 route, so barycentric refinement cannot introduce selected subcovers, quotient stream equality, choice, trichotomy, host equality, or ambient completeness. □

Theorem 5966.16 (Dyadic interval net finite-chain coverage)**.** *Every accepted finite chain of interval cells consumed by closed-bounded interval, compact-interval, or uniform-modulus routes is exhausted by the displayed rows*

$$D, J, E, S, R, Q, H, C, P, N.$$

Coverage is checked by finite induction over the listed cells in J: each cell reads its dyadic endpoints and radius from D, its adjacency and endpoint incidence from E, its observed endpoint windows from S, its rational readback and tolerance schedule from R, and its terminal Real-facing boundary from Q. No open-cover compactness, selected subcover, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness is available along the chain.*

Proof. Start with the carrier of Definition 5966.1. The finite chain is the listed interval-cell row J, while D and E provide the only endpoint arithmetic and incidence data attached to those cells.

Use the cover refinement theorem Definition 5966.2. It forces every consumer-facing cover read to pass through D, J, E before any StreamName, RegSeqRat, or Real-facing row is visible.

Induct over the finite list of cells using Definition 5966.4. The base case reads the initial listed cell data. The step consumes one displayed cell, preserves adjacency through E, and leaves the remaining listed cells under the same $D, J, E, S, R, Q, H, C, P, N$ route.

Finally replay the downstream rows. The windows S, readback R, seal Q, and structural rows H, C, P, N preserve the finite chain without adding open-cover compactness, selected subcovers, quotient stream equality, choice, trichotomy, host real equality, or ambient completeness. □

Theorem 5966.17 (Dyadic interval net finite-cell dependency chain)**.** *Every finite-cell read of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet consumed by compact, closed-bounded, or uniform-modulus chapters must expose* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ *retained-cell rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the interval-net row is public. The dependency chain is a finite route through the displayed packet and sibling refinement rows, not an open-cover compactness theorem, selected subcover, quotient stream equality, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Start with finite-chain coverage, Definition 5966.16. It exhausts every downstream finite-cell read by the displayed rows $D, J, E, S, R, Q, H, C, P, N$.

Apply the compact-interval forward route and the uniform-modulus refinement route, Definition 5966.24 and Definition 5966.22. These routes keep compact and modulus consumers downstream of endpoint arithmetic, retained finite cells, StreamName windows, RegSeqRat readback, and the Real seal.

Finally read the sibling dependency route of Definition 5966.20. It supplies the $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ retained-cell dependency together with DyadicRatCore, RegSeqRat, and Real rows, so the public interval-net read remains the stated finite-cell dependency chain. □

Theorem 5966.18 (Dyadic interval net sibling cover route)**.** *Every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite-cell route is consumed by the finite membership ledger of chapter 1428. The consumer route sends the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row D, the finite cell and endpoint-incidence rows J, E, the* $\mathsf{StreamName}^{\uparrow}$ *window row S, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R, and the* $\mathsf{Real}^{\uparrow}$ *seal row Q into the*

cover packet's endpoint, mesh, membership, window, readback, and seal rows. This is a substantive downstream cover-consumer dependency: it records how the interval-cover chapter uses the net carrier, and it is not a self-reference.

Proof. Project the net carrier. The rows D, J, E are the finite cell data, and the rows S, R, Q are the StreamName, RegSeqRat, and Real-facing handoff data already ordered by Definition 5966.13.

Read the cover consumer. The DyadicIntervalCover packet uses dyadic endpoints, mesh and membership rows, finite windows, regular readbacks, and a Real seal, so the net rows provide exactly the finite-cell source consumed by that ledger.

The dependency is external to this chapter. It points from the net carrier to the sibling cover chapter and does not add a second net theorem, selected subcover, open-cover compactness row, or ambient Real completeness principle. □

Theorem 5966.19 (Dyadic interval net regular tail cover)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *cover consumed by a regular-tail or Real-facing route must expose* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic, finite cell incidence,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance and readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any cover conclusion is public. The cover is therefore a finite route through* $D, J, E, S, R, Q, H, C, P, N$*, not a selected subcover, open-cover compactness principle, quotient stream equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the carrier of Definition 5966.1. The rows D, J, E expose dyadic endpoints, listed cells, and endpoint incidence before any cover read can be exported.

Apply the RegSeqRat route and StreamName handoff, Definition 5966.11 and Definition 5966.12. These force regular-tail consumers to read the finite windows S and the tolerance/readback row R after the endpoint and incidence rows.

Finally use the Real seal nonescape theorem Definition 5966.13. The seal row Q is downstream of the displayed finite route and cannot be read as selected subcovers, open-cover compactness, quotient stream equality, countable choice, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicIntervalNet}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicIntervalNet}^{\uparrow}$ packet is generated from dyadic endpoint arithmetic, finite interval cells, endpoint incidence, StreamName windows, RegSeqRat readback, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5966.1 and Definition 5966.2 seed the finite dyadic interval-net carrier over dyadic endpoint arithmetic, listed interval cells, endpoint incidence, StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicIntervalNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic cell coverage, endpoint-incidence refinement, StreamName and RegSeqRat readback routing, Real seal ordering, provenance, and local naming for BEDC.Derived.DyadicIntervalNetUp

The $\mathsf{DyadicIntervalNet}^{\uparrow}$ L10 route consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ radius and endpoint arithmetic rows of chapter 291 together with the located endpoint-and-cell surface of chapter 394 before interval-cover consumers read the listed dyadic cells. This dependency keeps the net packet on finite radius rows and located interval endpoints before the downstream cover route of chapter 1428 is exposed. The bounded interval net consumer of chapter 1250 reads the same DyadicRatCore endpoint arithmetic and dyadic interval refinement route before compact interval or Real-completion consumers may inspect a finite cover. Every $\mathsf{DyadicIntervalNet}^{\uparrow}$ finite-cell route supplies the finite mesh and dyadic comparison rows consumed by $\mathsf{LocatedIntervalLebesgueNumber}^{\uparrow}$

before the located-interval Lebesgue-number route is read. The downstream handoff stays tied to the finite interval-cell data of chapter 5966 and the located-interval certificate of chapter 8660, so the interval-net chapter remains the upstream finite-cover source for later compact-uniform and subcover consumers.

Theorem 5966.20 (DyadicIntervalNet sibling dependency route)**.** *Every finite-cell read of a* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier consumed by chapter 1250, chapter 6207, or chapter 5965 must first expose DyadicRatCore endpoint arithmetic, RegSeqRat readback, DyadicIntervalRefinement retained-cell rows, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus no compact, uniform-modulus, or common-window completion consumer can use the interval-net carrier as an ambient compactness principle, a host completion theorem, or an untracked real endpoint object.*

Proof. Read the dependency route through chapter 291, chapter 301, chapter 1831, and chapter 38. These sibling chapters supply endpoint arithmetic, regular rational readback, retained-cell refinement, and terminal sealing before the finite-cell row is visible to any compact or uniform-modulus consumer. Therefore the DyadicIntervalNet packet remains a finite interval-net route rather than an ambient compactness or host-completion principle.

The closed-bounded interval net consumer of chapter 1250 reads the same DyadicRatCore endpoint arithmetic, RegSeqRat tolerance readback, DyadicIntervalRefinement retained-cell rows, and terminal $\mathsf{Real}^{\uparrow}$ seal before it treats the finite cells as a bounded interval net.

The regular sequence space consumer of chapter 5965 uses the same finite-cell refinements as a common-window completion route. Its StreamName and RegSeqRat windows must still pass through DyadicRatCore tolerances, retained dyadic cells, and the terminal $\mathsf{Real}^{\uparrow}$ seal before Real-facing reads are public. □

Theorem 5966.21 (DyadicIntervalNet compactness consumer route)**.** *Any compactness consumer using a* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier factors through the existing compact-interval finite-net route of chapter 3471 and chapter 4031 only after DyadicRatCore endpoints, RegSeqRat tolerance readback, DyadicIntervalRefinement retained cells, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal are displayed. The route exports finite interval cells and endpoint incidence to compactness-facing consumers; it does not export open-cover compactness, selected subcovers, countable choice, quotient stream equality, host Real equality, or ambient Real completeness.*

Proof. Start with the sibling dependency route of Definition 5966.20. It places DyadicRatCore endpoint arithmetic, RegSeqRat readback, DyadicIntervalRefinement retained-cell rows, and the terminal Real seal before any compact, uniform-modulus, or completion consumer can use the finite-cell row.

For a named compactness consumer, route the read through chapter 3471 and chapter 4031. Those chapters record the compactness-facing boundary as a finite-net route, so the interval-net packet is consumed as displayed finite cells rather than as a detached compactness theorem.

For compact-interval finite-net uses, read the finite-net siblings through the compact-interval handoff already cited by the DyadicIntervalNet routes. The same endpoint arithmetic, tolerance readback, retained dyadic cells, and terminal seal remain visible before any finite-net conclusion is public.

Thus both consumer paths preserve the L10 finite-cover route. Since neither path adds a coordinate for open-cover compactness, selected subcovers, countable choice, quotient stream equality, host Real equality, or ambient Real completeness, the DyadicIntervalNet packet remains a finite dependency edge for compactness consumers. □

Theorem 5966.22 (Dyadic interval net uniform-modulus refinement route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *mesh-refinement read consumed by* $\mathsf{CompactIntervalUniformModulus}^{\uparrow}$ *or* $\mathsf{FiniteEpsilonNetUniformModulus}^{\uparrow}$ *must first pass through the finite route*

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$$

of the accepted carrier $I = (D, J, E, S, R, Q, H, C, P, N)$*. The mesh-refinement induction, the* $\mathsf{RegSeqRat}^{\uparrow}$ *route, and the* $\mathsf{StreamName}^{\uparrow}$ *handoff therefore expose dyadic endpoint arithmetic, listed interval cells, endpoint incidence, finite observation windows, regular rational readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming before any uniform-modulus ledger is read.*

Proof. Start with the mesh-refinement induction of Definition 5966.4. It keeps every refined finite cell inside the displayed rows D, J, E and preserves the downstream route through S, R, Q, H, C, P, N at each finite refinement depth.

Next apply the $\mathsf{RegSeqRat}^{\uparrow}$ route of Definition 5966.11 and the $\mathsf{StreamName}^{\uparrow}$ handoff of Definition 5966.12. These rows force endpoint observations through S and rational tolerance readback through R before the terminal Real seal Q is opened.

Now read the consumer rows of chapter 6207 and chapter 8269. A compact-interval or finite-epsilon-net uniform-modulus consumer receives only the routed dyadic packet, so its ledger is downstream of the displayed sequence $D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$.

The route has no coordinate for open-cover compactness, selected subcovers, quotient stream equality, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Thus every uniform-modulus read remains a finite dyadic interval-net refinement read before the modulus ledger is consumed. □

Theorem 5966.23 (Dyadic interval net compact-uniform-continuity forward route)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet first reads the dyadic endpoint arithmetic, finite interval cells, endpoint incidence,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before it folds the finite-net data into a uniform continuity ledger. The route passes through chapter 9237, chapter 6207, and chapter 8269 as a finite cover-refinement handoff, not as open-cover compactness, selected subcovers, quotient stream equality, or countable-choice schedules.*

Proof. Project the accepted $\mathsf{DyadicIntervalNet}^{\uparrow}$ carrier $I = (D, J, E, S, R, Q, H, C, P, N)$. Its cover-refinement rows are the dyadic endpoint arithmetic D, the listed interval cells J, and the endpoint-incidence row E.

Apply the uniform-modulus refinement route Definition 5966.22. It keeps the finite cells on the route

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$$

before any compact-interval or finite-epsilon-net uniform-modulus consumer can read the packet.

Now read the compact-uniform-continuity capstone and the sibling modulus chapters. The vision site supplies the finite-net fold being consumed, while the two sibling concrete chapters supply the compact-interval and finite-epsilon-net uniform-modulus targets for that fold.

Since the route contains no coordinate for arbitrary open covers, selected subcovers, quotient stream equality, countable-choice schedules, or ambient Real completeness, every compact-uniform-continuity read remains the displayed finite dyadic interval-net handoff. □

Theorem 5966.24 (Dyadic interval net compact interval forward route)**.** *Every compact-interval or closed-bounded-interval consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet must first read the finite cell row, endpoint incidence,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal before compact interval or uniform-modulus rows are public. In particular, consumers routed to chapter 6207 or chapter 1250 receive the interval net only as the displayed finite handoff*

$$D, J, E \longmapsto S, R \longmapsto Q.$$

The route exports no open-cover compactness theorem, selected finite subcover, quotient stream equality, host real equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 5966.1. The finite cell row and endpoint incidence are the rows J, E, while the endpoint arithmetic is read from the $\mathsf{DyadicRatCore}^{\uparrow}$ row D.

Use the cover refinement theorem Definition 5966.2. It places every cover-facing read through the finite cell and endpoint rows before any Real or Completion consumer can inspect the terminal seal.

Next use the StreamName and RegSeqRat handoff of Definition 5966.12. The window row S and regular rational row R are forced between the finite interval-net rows and any compact-interval consumer.

Finally replay the Real-seal nonescape theorem Definition 5966.13. The seal Q remains downstream of D, J, E, S, R, so the compact interval and uniform-modulus routes receive only the finite dyadic handoff and not a host compactness or choice principle. □

The $\mathsf{DyadicIntervalNet}^{\uparrow}$ packet supplies the finite cell route consumed by chapter 8657 and by compact-interval finite-cover consumers. The downstream read receives listed dyadic cells, endpoint incidence, dyadic tolerance rows, StreamName windows, RegSeqRat readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal as displayed packet data, not as open-cover compactness, selected subcovers, countable choice, quotient stream equality, or ambient real completeness.

Theorem 5966.25 (Dyadic interval net compactness forward reference)**.** *The* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet is consumed by the finite interval-net compactness route of chapter 8657 through dyadic endpoint rows, finite interval cells,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. In carrier notation, every such compactness-facing read first exposes the dyadic endpoint arithmetic D, the finite cell row J, and endpoint incidence E, then passes through the window and readback rows S, R, and only then reaches the Real seal Q and the structural rows H, C, P, N.*

Proof. Project the carrier of Definition 5966.1. The dyadic endpoint row D, finite interval-cell row J, and endpoint-incidence row E are the only rows that can feed the compactness consumer.

The compactness chapter chapter 8657 reads those rows as finite interval-net data. Its consumer route therefore receives cell coverage from the displayed dyadic rows, not from an ambient open-cover compactness principle.

The analytic readback remains downstream. Endpoint observations are exposed through S, regular rational tolerances are read through R, and the Real-facing boundary is sealed only at Q.

Finally the structural rows H, C, P, N transport, replay, source, and name the same packet. Thus the forward reference records a substantive child route from $\mathsf{DyadicIntervalNet}^{\uparrow}$ to its compactness consumer without adding selected subcovers, countable choice, quotient streams, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5966.26 (Dyadic interval net subdivision closed-bounded route)**.** *Every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

used by an interval-lattice consumer factors through the dyadic cell rows of Definition 5966.2, the subdivision carrier of chapter 7480, and the closed-bounded interval net rows of chapter 1250. The route sends D, J, E *to the subdivision scale, grid, and coverage rows, then sends the resulting finite cells to the closed-bounded interval mesh before* S, R, Q *are read. It exports only displayed dyadic cells, subdivision rows, finite mesh coverage, StreamName windows, RegSeqRat readback, and the Real seal; it exports no open-cover compactness theorem, selected finite subcover, quotient stream equality, countable-choice schedule, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the local carrier and cover-refinement rows. The rows D, J, E display dyadic endpoint arithmetic, the listed interval cells, and endpoint incidence, so every interval-net read starts as a finite cell read.

Next pass through dyadic interval subdivision. The subdivision chapter reads closed interval endpoints, dyadic scale, grid rows, cell coverage, and mesh bounds as displayed finite rows. Therefore a lattice consumer can refine J, E only by exposing those subdivision rows, not by invoking an ambient cover or hidden grid selector.

Now attach the closed-bounded interval net. Its carrier reads the located endpoint, mesh, rational and dyadic cell, center, and coverage rows before any Real-facing handoff is public, so the interval net becomes part of the closed-bounded finite mesh route.

Finally read S, R, Q, H, C, P, N. The StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows preserve the same finite route. Since none of the three carriers contains an open-cover compactness, selected-subcover, quotient-stream, countable-choice, host-equality, or ambient-completeness coordinate, the lattice route cannot export one. □

Theorem 5966.27 (Dyadic interval net finite-net sibling dependency route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite interval-net refinement consumed by a downstream compact or Real-facing route first displays the sibling rows*

$$\mathsf{FiniteLebesgueNumber}^{\uparrow}, \quad \mathsf{CompactMetric}^{\uparrow} \text{ or } \mathsf{CompactInterval}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The finite Lebesgue-number row supplies the mesh threshold, the compact-metric or compact-interval row supplies the bounded interval source, the dyadic row supplies endpoint and radius arithmetic, and the terminal Real row supplies only the seal reached after the finite cells have been listed. Thus the route links chapter 452, chapter 129, the compact-interval surface of chapter 3986, chapter 291, and chapter 38 without exporting an open-cover compactness theorem, selected subcover, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the interval-net carrier through the routed chapter body. Its finite cells are already listed before any Real-facing read is available, so a downstream consumer starts from the displayed mesh threshold and compact source rows.

The dyadic arithmetic row then fixes the endpoint and radius checks for each listed cell. This keeps refinement a finite ledger computation rather than an appeal to a host open-cover compactness principle.

Only after those rows have been displayed may the consumer reach the terminal Real seal. Transport, replay, provenance, and local naming preserve that same finite route, excluding selected subcovers, quotient equality, choice, and ambient completeness. □

Theorem 5966.28 (DyadicIntervalNet finite closed-interval nested-intersection route)**.** *Every* $\mathsf{FiniteClosedIntervalNestedIntersection}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet must inspect the finite interval-cell refinement rows before it reaches terminal* $\mathsf{Real}^{\uparrow}$ *seals. The route passes through the local DyadicIntervalNet carrier of chapter 5966 and then through the finite closed-interval nested-intersection packet of chapter 8584; it exports no selected intersection point, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the DyadicIntervalNet carrier of Definition 5966.1. The rows D, J, E display dyadic endpoint arithmetic, finite cells, and endpoint incidence before any StreamName, RegSeqRat, or Real-facing seal is visible.

Apply the finite closed-interval nested-intersection carrier of Definition 8584.1. Its cell row reads the same interval-net data before bounded-sequence or nested-intersection consumers receive a Real-facing handoff.

Thus the downstream nested-intersection route is a consumer of displayed finite cell refinement, not a source of completed intersections, selected points, quotient equality, choice, or ambient completeness. □

Every $\mathsf{DyadicIntervalNet}^{\uparrow}$ packet reads the dyadic interval basis of chapter 1916, the finite epsilon-net surface of chapter 1767, and the dyadic rational core of chapter 291 before compactness or Real-completion consumers inspect the mesh. This route keeps the interval cells, finite covering rows, and endpoint arithmetic visible before any downstream compactness, Cauchy, or $\mathsf{Real}^{\uparrow}$ seal argument treats the net as a completed object.

Theorem 5966.29 (Dyadic interval net basis route)**.** *Every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *read first consumes the dyadic interval basis, finite epsilon-net, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint rows before any compactness or Real-completion consumer reads the mesh.*

Proof. Project the mesh route through chapter 1916. The interval basis supplies the displayed cells, and chapter 1767 supplies the finite net rows over those cells.

The endpoint and width comparisons are then read through chapter 291. Only after these three sibling rows are visible can a compactness or Real-completion consumer inspect the mesh as a net. □

Theorem 5966.30 (DyadicIntervalNet forward lattice route)**.** *Every compactness, finite-net, or Real-completion consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier must expose* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius arithmetic, the listed interval-cell row, endpoint incidence,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before reading compactness or completion rows. The forward lattice is therefore the displayed route*

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$$

of Definition 5966.1, not an open-cover compactness theorem, selected finite subcover, quotient stream equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $I = (D, J, E, S, R, Q, H, C, P, N)$ of Definition 5966.1. Endpoint arithmetic and radius comparisons live in D, the listed cells live in J, endpoint incidence lives in E, and the completion-facing rows are downstream in S, R, Q.

Apply the finite-mesh Real-completion route and compact interval forward route, Definition 5966.36 and Definition 5966.24. They force mesh, compact-interval, and completion consumers to display finite cells, endpoint arithmetic, StreamName windows, RegSeqRat readback, and the terminal Real seal in that order.

Apply the regular-Cauchy Real forward route and finite-net sibling dependency route, Definition 5966.34 and Definition 5966.27. These routes keep regular-Cauchy, finite-net, and compactness consumers on the same dyadic endpoint, finite-cell, window, readback, and Real-seal lattice.

Finally replay H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the finite lattice route and provide no coordinate for open-cover compactness, selected subcovers, quotient stream equality, countable choice, or ambient Real completeness. □

The finite interval-net surface of $\mathsf{DyadicIntervalNet}^{\uparrow}$ is read before the finite coverage consumer of chapter 1767 and the endpoint-radius consumer of chapter 1709. The interval cells D, J, E expose dyadic endpoints and adjacency first; only then may a downstream finite epsilon-net package read centers, tolerances, and coverage rows, or a located real interval package read endpoint, radius, locator, and metric-bracket rows.

Theorem 5966.31 (Dyadic interval net finite-epsilon located-interval dependency)**.** *Every finite-epsilon-net or located-real-interval consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *carrier*

$$I = (D, J, E, S, R, Q, H, C, P, N)$$

must read the finite interval-net route

$$D, J, E \longmapsto S, R \longmapsto Q, H, C, P, N$$

before exposing interval coverage, center coverage, or endpoint-radius locatedness. Thus chapter 1767 receives a finite cell surface rather than a host finite subset, and chapter 1709 receives displayed endpoint-radius data rather than host interval equality.

Proof. Project the accepted carrier. The rows D, J, E are the only dyadic interval-net cells, endpoint incidences, and bracketing data available to any consumer.

The finite epsilon-net consumer can therefore read centers and coverage only after the interval cells have been displayed. Its coverage row is downstream of the same finite dyadic route, so it cannot replace the interval net by a host finite subset or selected open-cover proof.

The located real interval consumer is similarly downstream. It may read endpoints, radii, locators, and metric brackets only after the dyadic endpoint and incidence rows have supplied the displayed interval cells.

Finally S, R, Q, H, C, P, N preserve the route through StreamName windows, rational readback, the terminal Real seal, transport, replay, provenance, and local naming. No host interval equality, arbitrary open cover, selected subcover, countable choice, or ambient completeness row is exported. □

$\mathsf{DyadicIntervalNet}^{\uparrow}$ is consumed by chapter 8657 through finite subdivision-depth coverage and endpoint incidence rows. The compactness packet therefore reads the finite net before it exports any interval-cover handoff, and it does not introduce open-cover compactness, selected subcovers, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 5966.32 (Dyadic interval net finite-density consumer route)**.** *Every completion-facing consumer that reads a* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet as finite dyadic cell data may pass that read to* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *only after the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows have been exposed. The handoff cites the finite density packet of chapter 6003 only through dyadic cell incidence, stream windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.*

Proof. Project the dyadic interval-net hub through its L10 sibling routes. The dyadic cells and endpoint incidence are read first through the $\mathsf{DyadicRatCore}^{\uparrow}$ rows, so the consumer has no immediate completion-facing density claim.

The route then exposes the $\mathsf{StreamName}^{\uparrow}$ finite-window row and the $\mathsf{RegSeqRat}^{\uparrow}$ readback row. These rows are the visible bridge from finite dyadic cells to regular rational observations, and they match the stream and readback requirements used by chapter 6003.

Only after those rows are displayed may the consumer read a terminal $\mathsf{Real}^{\uparrow}$ seal and hand the packet to $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$. The same hsame transport, Cont replay, Pkg provenance, and local NameCert rows preserve the displayed route, so no quotient stream, selected representative, ambient completeness theorem, or unrecorded density selector is exported by the dyadic interval net itself. □

The compactness-facing route for $\mathsf{DyadicIntervalNet}^{\uparrow}$ is read by two finite-net siblings before interval compactness consumers inspect a finite subcover ledger. The finite dyadic cells, endpoint incidence rows, and mesh witnesses of the local packet feed the finite-cover carrier of chapter 3471; the same rows are then consumed by the net-compactness packet of chapter 4031 before closed-interval or compact-metric consumers read the retained finite-subcover data.

This dependency is a displayed compactness route, not an ambient open-cover theorem. It keeps the dyadic cell list, finite mesh, Heine–Borel finite-net carrier, Heine–Borel net-compactness carrier, and downstream finite subcover ledger visible as NameCert rows, while refusing selected subcovers, quotient interval equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness. The $\mathsf{DyadicIntervalNet}^{\uparrow}$ finite-net route supplies the finite dyadic endpoints, listed cells, endpoint incidence, and mesh witnesses that a finite-net consumer can read before any compact-interval boundary is exposed. The same route then hands those rows to chapter 3471 for finite-net coverage and to chapter 6207 for compact-interval modulus control. The dependency is horizontal and finite: it records how the dyadic net packet feeds those sibling consumers, and it does not claim open-cover compactness, selected subcovers, quotient interval equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

The displayed endpoint, radius, and cell-incidence rows are also the finite-net surface consumed by chapter 1250; both the closed-bounded interval and compact-interval uniform-modulus routes receive listed dyadic cells, not open-cover compactness.

Theorem 5966.33 (DyadicIntervalNet forward dependency route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite-cell route consumed by compact interval, finite subcover modulus, or regular-real completion consumers first exposes* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic,* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The forward dependency route therefore factors through chapter 291, chapter 126, chapter 301, and chapter 38 before any compactness, finite-subcover, or completion-facing consumer can treat the finite interval net as public.*

Proof. Project the accepted carrier of Definition 5966.1. Its finite-cell data is not public on its own; the endpoint arithmetic is read through the $\mathsf{DyadicRatCore}^{\uparrow}$ row, while endpoint windows and tolerance readback are routed through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$.

Use the existing sibling dependency and compact interval routes for $\mathsf{DyadicIntervalNet}^{\uparrow}$. They place compact interval and finite-subcover consumers downstream of the displayed finite cells, endpoint arithmetic, stream windows, and regular-rational readback.

Finally apply the terminal Real-seal discipline of the same packet. Since the $\mathsf{Real}^{\uparrow}$ seal remains downstream of those rows, regular-real completion consumers receive only the displayed forward dependency route and cannot replace it with an ambient compactness theorem, selected finite subcover, quotient stream equality, host real endpoint, or unrestricted completion principle. □

Theorem 5966.34 (DyadicIntervalNet regular-Cauchy Real forward route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that exports interval windows toward regular-Cauchy or Real-completion routes must first pass through chapter 301, chapter 126, and chapter 38 before any terminal* $\mathsf{Real}^{\uparrow}$ *read is available.*

Proof. Project the accepted carrier of Definition 5966.1. The finite dyadic interval cells expose endpoint and radius data, but they do not by themselves license a regular-Cauchy or Real-facing read.

For completion-facing consumers, route the endpoint windows through the $\mathsf{StreamName}^{\uparrow}$ schedule and the regular rational readback through $\mathsf{RegSeqRat}^{\uparrow}$. Only after those two sibling rows have been displayed may the terminal $\mathsf{Real}^{\uparrow}$ seal be read.

Thus every such consumer factors through the StreamName, RegSeqRat, and Real chapters named above. The route supplies no ambient compactness theorem, selected limit, quotient stream equality, host real endpoint, or unrestricted completion principle. □

Theorem 5966.35 (DyadicIntervalNet regular-Cauchy consumer route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer that reads a* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *surface must first expose the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic,* $\mathsf{StreamName}^{\uparrow}$ *finite windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *sealing,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows attached to the interval-net packet. The dyadic interval cells therefore enter regular-Cauchy and Real-name consumers only through the finite route named by chapter 291, chapter 126, chapter 301, and chapter 38.*

Proof. Project the accepted $\mathsf{DyadicIntervalNet}^{\uparrow}$ carrier. Its public surface is finite dyadic cell incidence, so endpoint arithmetic is read first through the $\mathsf{DyadicRatCore}^{\uparrow}$ row rather than through host real endpoints.

The same consumer route also serves the finite sign-change bisection path of chapter 3359. The $\mathsf{DyadicIntervalNet}^{\uparrow}$ packet supplies the displayed dyadic cells, while $\mathsf{RealBolzano}^{\uparrow}$ supplies the opposed sign-change row, midpoint read, retained bracket, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{Real}^{\uparrow}$ seal ordering needed by that bisection surface.

The completion-facing read then opens the $\mathsf{StreamName}^{\uparrow}$ finite windows and the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback. These rows are the only route by which the interval cells become regular-Cauchy data for a downstream consumer.

The terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of that readback. Componentwise hsame transport and displayed Cont replay preserve the same finite rows, while Pkg provenance and local NameCert name the packet.

Thus a regular-Cauchy or Real-facing consumer receives the interval-net surface only after the listed rows have been displayed. The route exports no selected limit, quotient stream equality, countable-choice schedule, host real endpoint, ambient completeness theorem, or detached interval-net object. □

Theorem 5966.36 (DyadicIntervalNet finite-mesh Real-completion route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that routes interval cells into finite dyadic mesh and Real-completion data reads the following displayed rows:*

1. *the* $\mathsf{FiniteDyadicMesh}^{\uparrow}$ *mesh row that groups the dyadic interval cells into a finite subdivision;*
2. *the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic row that records endpoint, radius, and containment comparisons;*
3. *the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the displayed mesh request;*
4. *the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows;*

5. the terminal $\mathsf{Real}^{\uparrow}$ *seal row available only after the preceding rows have been displayed.*

No such consumer may replace these rows by open-cover compactness, a selected limit, quotient stream equality, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted $\mathsf{DyadicIntervalNet}^{\uparrow}$ carrier. Its cells provide finite dyadic interval data, but a Real-completion consumer still needs an explicit mesh row and dyadic arithmetic ledger before a stream window can be opened.

Read the $\mathsf{FiniteDyadicMesh}^{\uparrow}$ row as the finite subdivision surface and the $\mathsf{DyadicRatCore}^{\uparrow}$ row as the endpoint and containment ledger. These rows determine the finite window request sent to $\mathsf{StreamName}^{\uparrow}$.

The $\mathsf{RegSeqRat}^{\uparrow}$ row reads back exactly those windows, and the terminal $\mathsf{Real}^{\uparrow}$ seal is downstream of that readback. The listed rows exhaust the consumer route because the carrier supplies no coordinate for the excluded principles. □

Theorem 5966.37 (DyadicIntervalNet Real-completion route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet consumed by a Real or regular-Cauchy completion route must expose the finite dyadic interval cells,* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint arithmetic,* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before the consumer reads completion data. Thus the dyadic interval net supplies the finite mesh route consumed by chapter 301 and chapter 38, with the same finite-cell discipline used by chapter 6207.*

Proof. Project the accepted $\mathsf{DyadicIntervalNet}^{\uparrow}$ carrier. Its finite cells and endpoint incidences are displayed first, and the endpoint arithmetic is read through $\mathsf{DyadicRatCore}^{\uparrow}$ rather than through host real equality.

The displayed cells open finite $\mathsf{StreamName}^{\uparrow}$ endpoint windows. Those windows are then read as $\mathsf{RegSeqRat}^{\uparrow}$ data, so a completion-facing consumer receives rational tolerance information before any terminal real seal is visible.

Only after this route has exposed cells, dyadic arithmetic, stream windows, regular-rational readback, and the $\mathsf{Real}^{\uparrow}$ seal may the consumer read completion data. The route exports no selected limit, quotient stream equality, countable-choice schedule, host real endpoint, or ambient completeness theorem. □

Theorem 5966.38 (DyadicIntervalNet finite-mesh extraction)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *finite mesh consumed by a* $\mathsf{RegularCauchySequence}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion route factors through the displayed chain*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The extraction first reads dyadic endpoint and containment arithmetic, then opens the finite StreamName windows induced by the mesh, then reads those windows as regular rational data, and finally exposes the terminal Real seal through the existing finite-mesh Real-completion route of Definition 5966.36. It exports no selected limit, quotient stream equality, host real equality, open-cover compactness, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the finite mesh Real-completion route Definition 5966.36. It already places the finite dyadic mesh row, $\mathsf{DyadicRatCore}^{\uparrow}$ arithmetic row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal in the required order.

Now read the regular-Cauchy forward route of Definition 5966.34. A $\mathsf{RegularCauchySequence}^{\uparrow}$ consumer may inspect interval windows only after they have been scheduled as StreamName windows and read back as regular rational data.

Combining the two routes makes finite mesh extraction a displayed dyadic, stream, regular-rational, and Real-seal handoff. Since neither route contains selected limits, quotient stream equality, host real equality, open-cover compactness, countable choice, or ambient completeness as carrier rows, the extraction cannot export those coordinates. □

Theorem 5966.39 (DyadicIntervalNet CauchyFilterSubsequence route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that exports finite dyadic interval windows toward* $\mathsf{CauchyFilterSubsequence}^{\uparrow}$ *must expose the dyadic endpoint and radius cells before the route reaches chapter 8749. The visible handoff factors through the existing* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance arithmetic, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback before the Cauchy-filter subsequence packet may consume those windows.*

Proof. Project the accepted carrier of Definition 5966.1. The finite net gives only displayed dyadic cells, endpoint comparisons, and radius budgets; it does not by itself select a subsequence or a filter limit.

Route those cells through the dyadic tolerance arithmetic, then through the StreamName schedule and RegSeqRat readback already used by the regular-Cauchy Real forward route. These rows are exactly the visible windows required by chapter 8749.

Thus the Cauchy-filter subsequence consumer receives finite dyadic net windows as displayed BEDC rows. The route exports no hidden subsequence selector, quotient stream equality, countable choice, host interval endpoint, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5966.40 (DyadicIntervalNet BishopCauchyDisplayCertificate route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer feeding a* $\mathsf{BishopCauchyDisplayCertificate}^{\uparrow}$ *route must expose the finite dyadic interval-net rows before the display certificate may read* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *seal rows. In carrier notation, the interval-net rows* D, J, E *are displayed first, the endpoint windows and regular readback pass through* S, R*, and only then may the Bishop Cauchy display certificate consume the Real-facing seal* Q *together with the structural rows* H, C, P, N*.*

Proof. Project the accepted carrier of Definition 5966.1. The rows D, J, E are the finite interval-net data: dyadic endpoint arithmetic, the listed cells, and endpoint incidence.

Route those rows into the display-certificate surface of chapter 5151. Its Cauchy display reads finite windows, regular rational readback, dyadic tolerances, and a terminal Real seal, so the interval-net source must first make the finite cells and endpoint bounds visible.

The downstream read therefore passes through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ only after the finite dyadic interval net has been displayed. The route exports no selected limit, quotient stream equality, countable choice, host endpoint equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5966.41 (DyadicIntervalNet RealInterval endpoint route)**.** *Every* $\mathsf{RealInterval}^{\uparrow}$ *endpoint-and-enclosure consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet must expose the dyadic endpoint row, finite interval-cell row, endpoint incidence row,* $\mathsf{StreamName}^{\uparrow}$ *window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and terminal* $\mathsf{Real}^{\uparrow}$ *seal before reading the interval enclosure of chapter 4111. In carrier notation, the read begins with the displayed dyadic net rows* D, J, E*, passes through the window and readback rows* S, R*, reaches the seal row* Q*, and only then may a RealInterval consumer inspect its endpoint-and-enclosure packet.*

Proof. Project the accepted carrier of Definition 5966.1. The dyadic endpoint row D, finite interval-cell row J, and endpoint-incidence row E are the only rows that can provide endpoint and enclosure evidence to a downstream interval consumer.

Apply the Real source link of Definition 5966.14. It requires Real-facing finite-net consumers to read D, J, E, S, R, Q in displayed order before handing the packet to interval or compactness consumers.

Now route the handoff to chapter 4111. Its endpoint-and-enclosure read consumes the same StreamName windows, RegSeqRat readback, and terminal Real seal as interval data, so the Real-Interval packet receives the dyadic net only after D, J, E, S, R, Q have been exposed.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same finite route, excluding selected subcovers, quotient stream equality, host endpoint equality, countable-choice schedules, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The $\mathsf{DyadicIntervalNet}^{\uparrow}$ prime forward route exposes a horizontal L10 dependency before arithmetic or Real-facing consumers inspect interval cells: dyadic endpoint and radius arithmetic comes from chapter 291, finite observation windows come from chapter 126, regular rational readback comes from chapter 301, and the terminal seal comes from chapter 38. Thus a prime-window or compact-interval consumer reads a finite dyadic interval net only after those displayed sibling rows are visible, not as an ambient compactness principle or host Real-completion object. The $\mathsf{DyadicIntervalNet}^{\uparrow}$ hub routes finite interval-cell readers through the L10 lattice before compact interval or completion consumers inspect a cell. The dependency first exposes $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint arithmetic from chapter 291, then $\mathsf{StreamName}^{\uparrow}$ finite windows from chapter 126, then $\mathsf{RegSeqRat}^{\uparrow}$ readback from chapter 301, and finally the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. Compact interval and completion-facing consumers therefore read finite interval cells only after those sibling rows are visible, not as ambient compactness, selected limits, quotient interval equality, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Theorem 5966.42 (DyadicIntervalNet sibling anchor)**.** *Every accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that supplies finite interval-net data must first expose the finite epsilon-net rows of chapter 1767 and the located endpoint rows of chapter 394. The dyadic interval-net packet therefore consumes finite center-radius coverage data together with located interval endpoint evidence before any L10 finite-net consumer may read it as interval-net coverage.*

Proof. Read the finite epsilon-net sibling first. Its rows supply the displayed finite family, radius ledger, and coverage request that a dyadic interval net later specializes to interval cells.

Next read the located interval sibling. Its endpoint rows provide the located lower and upper boundary data needed before a dyadic interval cell can be treated as an interval row rather than a bare dyadic token.

Finally combine the two routes through the local $\mathsf{DyadicIntervalNet}^{\uparrow}$ NameCert rows. The resulting consumer receives finite epsilon-net coverage and located interval endpoint evidence as explicit inputs, and it receives no host compactness theorem, selected subcover, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 5966.43 (DyadicIntervalNet interval-arithmetic sibling route)**.** *Every* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *consumer that uses interval arithmetic must expose the* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$*,* $\mathsf{FiniteCover}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *seal rows before treating the net as a cover witness. The accepted read is the displayed net route together with those sibling rows; it is not a host interval algorithm, a detached finite-cover proof, or an ambient real-completeness claim.*

Proof. Start with the displayed $\mathsf{DyadicIntervalNet}^{\uparrow}$ carrier. Its mesh and endpoint rows provide finite dyadic interval data, but they do not by themselves certify arithmetic closure or real-cover consumption.

Next read the sibling arithmetic route. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies the rational tolerance ledger, and the $\mathsf{DyadicIntervalArithmetic}^{\uparrow}$ row supplies the endpoint-operation surface needed before interval calculations can be consumed.

Finally attach the cover and real-seal rows. The $\mathsf{FiniteCover}^{\uparrow}$ row records the finite cover witness, while the $\mathsf{Real}^{\uparrow}$ seals keep the completed-real boundary visible. Thus an interval-arithmetic consumer receives exactly the displayed net plus its arithmetic, cover, and real-seal siblings. □

Theorem 5966.44 (DyadicIntervalNet interval-exchange dependency route)**.** *Every future* $\mathsf{IntervalExchange}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *packet must first read the dyadic endpoint arithmetic of chapter 291, the finite dyadic interval cells and endpoint-incidence rows of this chapter, the* $\mathsf{StreamName}^{\uparrow}$ *endpoint windows of chapter 126, the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance and readback rows of chapter 301, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38. Only after that route is displayed may the consumer interpret the finite partition as an interval exchange surface. The dependency exports no invariant measure, minimality claim, unique-ergodicity claim, host interval function, quotient real equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier $I = (D, J, E, S, R, Q, H, C, P, N)$ of Definition 5966.1. The endpoint arithmetic row D and the interval-cell rows J, E are the only finite partition data available before any Real-facing read.

Apply the cover refinement theorem Definition 5966.2. It forces every partition consumer to expose listed dyadic cells, endpoint incidence, and dyadic comparisons before opening StreamName, RegSeqRat, or Real rows.

The interval-exchange consumer then reads the same L10 route used by Real and completion consumers. Endpoint observations pass through S, tolerance and rational readback pass through R, and the terminal real-name boundary is available only at Q.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve that finite dependency route, so an interval-exchange surface cannot import invariant-measure, minimality, host-function, quotient-equality, choice, or ambient-completeness data from the dyadic net. □

The $\mathsf{DyadicIntervalNet}^{\uparrow}$ route reads dyadic mesh cells through chapter 291 and exports finite interval mesh data to the $\mathsf{Real}^{\uparrow}$-facing route of chapter 38. This is a finite net dependency, not compactness, open-cover completeness, or an ambient completeness principle for intervals.

Chapter 5967

Channel Termination Budget Certificate NameCert Construction

The $\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}$ packet realizes the local channel terminator and global halting-boundary separation of chapter 9369 as finite budget-certificate NameCert data. It sits beside the existing channel boundary packet of chapter 3996, the termination refusal packet of chapter 1197, the certificate compiler surface of chapter 473, and the bounded checker-budget surface of chapter 5949. The packet names a local event-delimiter budget and a refusal ledger for global certificate-level halting; it does not export a total halting classifier, an unbounded parser oracle, a host machine semantics theorem, proof-irrelevant equality, or a standard bridge.

Definition 5967.1 (Channel termination budget-certificate carrier)**.** *A* $\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (A, L, Z, F, G, R, H, C, P, N)$$

with the following rows:

1. *A is the active channel alphabet row. It records the binary source glyphs, the body fragments* 0 *and* 10*, and the displayed terminator* 11*.*
2. *L is the legal-event row. It records that a channel event lies in* $(0 \mid 10)^*11$ *and exposes the finite body fragments inspected before the terminator.*
3. *Z is the streaming scan budget row. It lists the finite positions inspected by the decoder until the first channel-level* 11 *is reached.*
4. *F is the local finish row. It certifies that the first displayed* 11 *ends the current legal event and that the body before it decodes to a source event.*
5. *G is the global-halting refusal row. It records that the finite local finish row is not a total classifier for arbitrary* Cont_P*-chains.*
6. *R is the sibling route row through the channel boundary, termination refusal, certificate compiler, and bounded checker-budget packets.*
7. *H records componentwise* hsame *transport for* A, L, Z, F, G, R *and the structural rows.*
8. *C records displayed* Cont *replay of the finite route*

$$A, L, Z \longmapsto F, G, R \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* ChannelTerminationBudgetCertificate$^{\uparrow}$, ChannelTerminationHaltingBoundary$^{\uparrow}$, TerminationRefusalBoundary$^{\uparrow}$, CertificateCompiler$^{\uparrow}$, BoundedTypecheckingBudget$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L, Z, F, G, R, H, C, P, N$. It has no coordinate for global halting decidability, an unbounded parser oracle, fuel-erased evaluation, host machine equality, proof irrelevance, or a bridge field.

Theorem 5967.2 (Channel termination budget-certificate local-global boundary)**.** *For every accepted* ChannelTerminationBudgetCertificate$^{\uparrow}$ *carrier $B = (A, L, Z, F, G, R, H, C, P, N)$, a local channel-event read is certified by the finite route*

$$A, L, Z \longmapsto F$$

while any attempted promotion to certificate-level halting must keep the refusal row G and sibling route row R in scope. Consequently the packet proves only that a legal channel event has a budgeted delimiter endpoint; it does not decide whether an arbitrary $\mathsf{NameCert}_P$ *and history h produce a terminating* Cont_P*-chain.*

Proof. Project the local channel route first. The alphabet row A names the fragments and terminator, L confirms that the inspected event is legal, and Z supplies the finite scan positions leading to the first channel-level 11.

The local finish row F is then read. It certifies the delimiter endpoint for the current event and the body decode before that endpoint, matching the local termination theorem of chapter 9369.

The global boundary is separate. Row G records that this finite delimiter budget is not a total classifier over arbitrary certificate histories, and R routes that refusal through the channel-boundary, termination-refusal, compiler, and bounded-budget sibling packets.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same separation, so no consumer may erase G and read the local delimiter certificate as a global halting decision. □

Falsifiable prediction: A claimed ChannelTerminationBudgetCertificate$^{\uparrow}$ consumer that exports a total certificate-level halting classifier while omitting the global-halting refusal row G and sibling route row R refutes the carrier of Definition 5967.1.

Independence witness: channel-termination-halting-boundary, terminationrefusalboundary, certificatecompiler, bounded-typechecking-budget: ChannelTerminationBudgetCertificate$^{\uparrow}$ is not bijective to any listed sibling because it carries the joint active alphabet, legal-event, finite scan-budget, local finish, global-halting refusal, sibling route, transport, replay, provenance, and local NameCert rows. The channel-boundary sibling separates event and halting strata without the explicit scan-budget certificate; the termination-refusal sibling records refusal visibility without the local delimiter budget; the certificate compiler sibling records compiler graph rows without the channel endpoint budget; the bounded checker-budget sibling records typechecking fuel without the channel terminator route.

Closure status for $\mathsf{NameCert}_{\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}}$.

Constructive story: A ChannelTerminationBudgetCertificate$^{\uparrow}$ packet is generated from an active channel alphabet row, legal-event row, streaming scan-budget row, local finish row, global-halting refusal row, sibling route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}}$, seed)

Scope closed: Definition 5967.1 and Definition 5967.2 seed the finite local-delimiter budget certificate and its global-halting refusal boundary.

Formal verification: FormalStatus(NameCert$_{\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global halting decidability, unbounded parser or evaluator totality, host machine semantics, proof-irrelevant equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, legal-event budget exactness, local finish correctness, global-halting refusal nonescape, sibling route coverage, provenance, and local naming for BEDC.Derived.ChannelTerminationBudgetCerti

Theorem 5967.3 (Channel termination budget-certificate dossier forward link)**.** *Every accepted* $\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}$ *packet forwards its local delimiter budget and global-halting refusal rows to the sibling surfaces chapter 3996, chapter 1197, chapter 473, and chapter 5949. The forward link reads exactly the rows*

$$A, L, Z, F, G, R, H, C, P, N$$

from Definition 5967.1: A, L, Z, F *certify the local channel delimiter budget,* G *preserves the refusal of global certificate-level halting,* R *exposes the sibling route, and* H, C, P, N *transport, replay, source, and locally name the same dossier.*

Proof. Project the accepted carrier of Definition 5967.1. The rows A, L, Z display the active alphabet, legal-event row, and finite scan budget, while F records the local delimiter endpoint.

Apply Definition 5967.2. It keeps the global-halting refusal row G and sibling route row R in scope whenever a consumer tries to read the local delimiter budget as certificate-level halting data.

Read the sibling route. The row R points to the channel termination halting boundary, termination refusal boundary, certificate compiler, and bounded typechecking budget surfaces, and the rows H, C, P, N preserve that same finite dossier by componentwise transport, displayed replay, provenance, and local naming.

Thus every accepted packet forwards the delimiter and refusal data to the stated sibling surfaces without adding a total halting classifier, unbounded parser oracle, host machine semantics theorem, proof-irrelevant equality, or bridge field. □

The local delimiter budget of $\mathsf{ChannelTerminationBudgetCertificate}^{\uparrow}$ is consumed through chapter 3996, chapter 1197, chapter 473, and chapter 5949. This route records local channel termination together with refusal of global halting; it is not a total halting classifier.

Chapter 5968

A Concrete Naming Certificate for $\mathsf{CadlagRealFunction}^{\uparrow}$

Following Skorokhod and Billingsley on right-continuous paths with left limits, the $\mathsf{CadlagRealFunction}^{\uparrow}$ packet records a real-valued path through finite window data, one-sided regulated fragments, and displayed jump-boundary rows. It sits after the regulated-function interface of chapter 1906, the bounded regulated envelope of chapter 5853, the finite observation schedule of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The certificate exposes a path-space boundary for stochastic-process and Skorokhod-topology consumers without importing selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5968.1 (Cadlag real function carrier)**.** *A* $\mathsf{CadlagRealFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, B, S, R, D, J, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, exposing finite one-sided regulated windows on the displayed real-domain schedule.*
2. *B is the* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *envelope row available on each bounded window requested by a consumer.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation schedule used to name right-hand windows and left-hand approach windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations of the displayed windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling window radius, jump comparison, and real-name readback precision.*
6. *J is the cadlag boundary row. It records right-continuity and left-limit availability only as finite window obligations tied to S, R, D.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regulated source, bounded-window envelope, stream schedule, rational readback, tolerance ledger, and cadlag boundary row have been displayed.*
8. *H records componentwise* hsame *transport for the source, bounded envelope, stream schedule, rational readback, tolerance, cadlag boundary, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a path-window request through $F, B, S, R, D, J, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{CadlagRealFunction}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedRegulatedFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CadlagRealFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, B, S, R, D, J, E, H, C, P, N$. It has no coordinate for selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, host path equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CadlagRealFunctionUp.CadlagRealFunctionTasteGate single carrier alignment`

Theorem 5968.2 (Cadlag real function NameCert obligations)**.** *For every accepted* $\mathsf{CadlagRealFunction}^{\uparrow}$ *carrier $K = (F, B, S, R, D, J, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CadlagRealFunction}^{\uparrow}}$ *surface consists exactly of carrier admission, regulated-source admission, bounded-window envelope admission, finite stream-window exposure, regular rational readback, dyadic tolerance control, right-continuity window obligation, left-limit window obligation, terminal real-seal nonescape, componentwise transport, displayed replay, package provenance, local naming, and ledger refusal for selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, host path equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows F, B, S, R, D, J, E are the only analytic rows before transport, replay, provenance, and local naming are read.

The regulated-source and bounded-window obligations are supplied by the sibling $\mathsf{RegulatedFunction}^{\uparrow}$ and $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ rows. The stream schedule S, rational readback R, and dyadic ledger D provide the finite data on which the cadlag boundary row J can state right-continuity and left-limit obligations.

The terminal seal E is downstream of these rows. Componentwise transport and displayed replay preserve the same finite route, and the provenance and naming rows record it as the local certificate surface.

Since no carrier coordinate stores selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, host path equality, or ambient real completeness, none of those objects can become a local NameCert obligation. □

Theorem 5968.3 (Cadlag real function regulated-window handoff)**.** *Every path-window consumer of an accepted* $\mathsf{CadlagRealFunction}^{\uparrow}$ *carrier must read the* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row, the* $\mathsf{BoundedRegulatedFunction}^{\uparrow}$ *bounded-window envelope, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger before the cadlag boundary row can export a terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff is therefore a finite regulated-window route, not a selected path limit, quotient path equality, measure-theoretic sample-space construction, trichotomy decision, or ambient completeness proof.*

Proof. Let $K = (F, B, S, R, D, J, E, H, C, P, N)$ be accepted. A path-window request first reaches the regulated source row F, then the bounded-window envelope B, then the stream schedule S, rational readback R, and dyadic tolerance ledger D.

Only after those rows are displayed can the cadlag boundary row J be consumed. The row J records right-continuity and left-limit availability as finite window obligations tied to the already displayed schedule and tolerance data.

The terminal $\mathsf{Real}^{\uparrow}$ seal E is downstream of J, and transport through H and replay through C keep the same route visible to downstream consumers. Provenance P and local naming N then certify the route as a $\mathsf{CadlagRealFunction}^{\uparrow}$ packet.

Because selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, host path equality, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from the carrier coordinates, no consumer can obtain them through the regulated-window handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{CadlagRealFunction}^{\uparrow}}$.

Constructive story: A $\mathsf{CadlagRealFunction}^{\uparrow}$ packet is generated from a RegulatedFunction source row, BoundedRegulatedFunction window envelope, StreamName finite schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, cadlag boundary row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CadlagRealFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5968.1, Definition 5968.2, and Definition 5968.3 seed the finite right-continuous-with-left-limits path route over regulated windows and terminal Real-facing seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CadlagRealFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient path equality, measure-theoretic sample spaces, trichotomy, host path equality, Lean verification, bridge checking, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, regulated-source admission, bounded-window envelope admission, stream-window exposure, rational readback, dyadic tolerance control, cadlag boundary obligations, transport, replay, provenance, and local naming for BEDC.Derived.CadlagRealFunctionUp.

Chapter 5969

A Concrete Naming Certificate for $\mathsf{AlmostPeriodic}^{\uparrow}$

Following Bohr and Besicovitch on almost periodic functions, read through Bishop–Bridges constructive analysis, the $\mathsf{AlmostPeriodic}^{\uparrow}$ packet records relative-density translation data for displayed real-valued function windows as finite BEDC rows. It sits beside the compact-source sup-norm ledger of chapter 3807, the modulus-composition route of chapter 3784, the Dini finite-window route of chapter 5154, the compact-uniform-continuity packet of chapter 388, and the terminal real-name surface of chapter 38. The packet names only finite translation ledgers and displayed sup-norm comparisons; it does not assert host function equality, an infinite hull compactification, Bohr compactification, quotient function-space equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5969.1 (AlmostPeriodic carrier)**.** *An* $\mathsf{AlmostPeriodic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, T, N_{\mathrm{rel}}, S, M, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-window row, listing the finite source windows and* $\mathsf{Real}^{\uparrow}$*-valued observations of the displayed function.*
2. *T is the translation-budget row, naming the finite translation requests, tolerance budgets, and admissible translated windows.*
3. *N_{rel} is the relatively-dense finite-net row, recording the bounded search interval and the finite list of candidate translations exposed to a consumer.*
4. *S is the sup-norm comparison row, read through the* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *surface and comparing each translated window only by displayed Real-sealed bounds.*
5. *M is the uniform-modulus row, assembled through* $\mathsf{UniformModulusComposition}^{\uparrow}$ *and the compact-uniform-continuity route before any translated read reaches S.*
6. *U is the Dini and compact-uniform handoff row, recording the finite monotone-window and compact-continuity evidence consumed from* $\mathsf{DiniUniformMonotone}^{\uparrow}$ *and* $\mathsf{CompactUniformContinuity}^{\uparrow}$.
7. *H records componentwise* hsame *transport for function windows, translation budgets, relatively-dense nets, sup-norm comparisons, uniform moduli, sibling handoffs, replay, provenance, and local naming.*

8. *C records displayed* Cont *replay from a tolerance request through* $F, T, N_{\mathrm{rel}}, S, M, U, H, P, N$.
9. *P is the* Pkg *provenance over* $\mathsf{AlmostPeriodic}^{\uparrow}$, $\mathsf{ContinuousSupNorm}^{\uparrow}$, $\mathsf{UniformModulusComposition}^{\uparrow}$, $\mathsf{DiniUniformMonotone}^{\uparrow}$, $\mathsf{CompactUniformContinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AlmostPeriodic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, T, N_{\mathrm{rel}}, S, M, U, H, C, P, N$. *It has no coordinate for host-level equality of functions, an infinite translation hull, quotient function spaces, choice-selected periods, open-cover compactness, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.AlmostPeriodicUp.AlmostPeriodicTasteGate single carrier alignment`

Theorem 5969.2 (AlmostPeriodic NameCert obligations)**.** *For every accepted* $\mathsf{AlmostPeriodic}^{\uparrow}$ *carrier* $A = (F, T, N_{\mathrm{rel}}, S, M, U, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{AlmostPeriodic}^{\uparrow}}$ *surface consists exactly of function-window admission, translation-budget accounting, relatively-dense finite-net exposure, sup-norm comparison through displayed Real-sealed bounds, uniform-modulus routing, Dini and compact-uniform sibling handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5969.1. The function rows, translation rows, relatively-dense net, sup-norm comparison, uniform-modulus handoff, and sibling rows are the only mathematical coordinates before transport, replay, provenance, and naming.

The translation read is finite. A tolerance request first enters T, then opens only the candidates listed in N_{rel}, and then compares translated windows through S.

The comparison is real-facing but not host-functional. The row S reads $\mathsf{ContinuousSupNorm}^{\uparrow}$ and terminal $\mathsf{Real}^{\uparrow}$ seals, while M and U route modulus evidence through chapter 3784, chapter 5154, and chapter 388.

Rows H, C, P, N transport, replay, source, and name exactly the same finite route. Since no excluded hull, quotient, host equality, choice-selected period, compactness, or completeness coordinate appears in the carrier, the listed rows exhaust the NameCert surface. □

Theorem 5969.3 (AlmostPeriodic sup-norm translation stability)**.** *Every accepted* $\mathsf{AlmostPeriodic}^{\uparrow}$ *packet exposes a translation-stability read only through the finite route*

$$F, T, N_{\mathrm{rel}} \longrightarrow M, U \longrightarrow S \longrightarrow H, C, P, N.$$

Thus a consumer may read a translated function window, a displayed translation budget, a relatively-dense finite-net candidate, a uniform-modulus handoff, and a sup-norm comparison row, but it may not replace this route by host equality of functions, an infinite period hull, quotient function-space equality, a choice-selected translation, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the accepted carrier. Rows F, T, N_{rel} determine which function windows and translation candidates can be inspected for the requested tolerance.

The modulus rows are downstream of that inspection. The uniform-modulus composition and Dini or compact-uniform handoff rows M, U must be replayed before a sup-norm comparison is visible.

The comparison row S is the only route to the Real-sealed sup-norm bound. The structural rows H, C, P, N preserve that finite route under transport and replay, so no host function equality, infinite hull, quotient equality, choice-selected translation, compactness principle, or Real completeness theorem is exported. □

Theorem 5969.4 (AlmostPeriodic ledger nonescape)**.** *Every* $\mathsf{AlmostPeriodic}^{\uparrow}$ *consumer read is generated by the rows named in Definition 5969.1. In particular, the packet exports no Bohr compactification, no completed translation hull, no host equality of functions, no quotient function-space equality, no choice-selected period, no open-cover compactness, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project an accepted carrier. Function-window reads pass through F, translation reads pass through T, relatively-dense candidate reads pass through N_{rel}, and every comparison read passes through S after the modulus rows M, U have been replayed.

Transport and replay preserve only those displayed rows. Provenance and local naming record the same packet and do not add analytic coordinates.

The excluded objects are absent from the carrier. They therefore cannot be obtained by projection, componentwise hsame transport, displayed Cont replay, package provenance, or local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlmostPeriodic}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AlmostPeriodic}^{\uparrow}$ packet is generated from function-window rows, translation-budget rows, relatively-dense finite-net rows, sup-norm comparison rows, uniform-modulus rows, Dini and compact-uniform handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AlmostPeriodic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5969.1, Definition 5969.2, Definition 5969.3, and Definition 5969.4 seed the finite almost-periodic translation ledger over displayed sup-norm and uniform-modulus rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AlmostPeriodic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, infinite translation hull compactness, Bohr compactification, quotient function-space equality, choice-selected periods, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, translation-budget accounting, relatively-dense finite-net exposure, sup-norm comparison stability, uniform-modulus routing, sibling handoff, transport stability, and ledger non-escape for BEDC.Derived.AlmostPeriodicUp.

Chapter 5970

A Concrete Naming Certificate for MetaCICResidualJoinDeterminacy$^{\uparrow}$

The MetaCICResidualJoinDeterminacy$^{\uparrow}$ packet realizes the residual-substitution boundary of chapter 9378 as finite local-join data. It sits beside the confluence-budget witness of chapter 5174, the residual-substitution compatibility boundary of chapter 4370, the local-diamond witness of chapter 3595, the residual envelope of chapter 3921, and the closed parallel-diamond premise ledger of chapter 2325. It names a budgeted local join packet for residual branches; it does not claim full Church–Rosser, unguarded substitution compatibility, global strong normalization, subject reduction, or host reduction semantics.

Definition 5970.1 (MetaCIC residual-join determinacy carrier)**.** *A* MetaCICResidualJoinDeterminacy$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (T, R, B, K, E, S, L, D, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed residual-window row, naming the closed MetaCIC source, the two residual branches, and the inspected redex-frontier fragments.*
2. *R is the residual-budget row, read from the confluence-budget witness and bounding residual addresses, replay fuel, and candidate-join search.*
3. *B is the bounded local-join row, recording the candidate common reduct accepted inside the displayed residual budget.*
4. *K is the determinacy row, requiring any accepted candidate for the same displayed residual branch data to be compared through the same budgeted row B.*
5. *E is the residual-envelope row, preserving the residual branches and the local envelope in which the bounded join is inspected.*
6. *S is the residual-substitution boundary row, recording that substitution compatibility is a guarded input shape, not an unguarded theorem.*
7. *L is the closed-premise ledger row consumed when closed local-diamond evidence is read.*
8. *D is the local-diamond witness row for the displayed peak and its bounded join.*
9. *O is the obstruction row retaining the refusal of full Church–Rosser, full parallel diamond, arbitrary residual substitution compatibility, subject reduction, and host semantics.*

10. *H records componentwise* hsame *transport for typed windows, residual budgets, bounded joins, determinacy rows, envelopes, substitution boundaries, closed-premise ledgers, local diamonds, obstruction rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from a residual-join request through $T, R, B, K, E, S, L, D, O, H, P, N$.*
12. *P is the* Pkg *provenance over* MetaCICResidualJoinDeterminacy$^{\uparrow}$, MetaCICConfluenceBudgetWitness$^{\uparrow}$, MetaCICResidualSubstitutionCompatibility$^{\uparrow}$, MetaCICLocalDiamondWitness$^{\uparrow}$, MetaCICConfluenceResidualEnvelope ClosedParallelDiamondPremiseLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{MetaCICResidualJoinDeterminacy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, R, B, K, E, S, L, D, O, H, C, P, N$. It has no coordinate for full parallel diamond, arbitrary closed Church–Rosser, unguarded substitution compatibility, global strong normalization, subject reduction, or host reduction semantics.

Theorem 5970.2 (MetaCIC residual-join determinacy NameCert obligations)**.** *For every accepted* MetaCICResidualJoinDeterminacy$^{\uparrow}$ *carrier $J = (T, R, B, K, E, S, L, D, O, H, C, P, N)$, the local* NameCert$_{\text{MetaCICResidualJoinDeterminacy}^{\uparrow}}$ *surface consists exactly of typed residual-window admission, residual-budget accounting, bounded local-join exposure, candidate determinacy inside the displayed budget, residual-envelope handoff, guarded residual-substitution boundary visibility, closed-premise ledger handoff, local-diamond handoff, obstruction retention, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5970.1. The displayed mathematical rows are $T, R, B, K, E, S, L, D, O$, followed by the structural rows H, C, P, N.

The local join is budgeted before it is deterministic. Rows T and R first bind the residual window and the residual budget; row B then records only the candidate common reduct accepted inside that budget.

The determinacy row K compares accepted candidates for the same displayed residual data through B. It does not supply a global confluence theorem or an unguarded substitution compatibility theorem.

Rows E, S, L, D, O keep the sibling handoffs and the obstruction boundary visible. Rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local NameCert surface. □

Theorem 5970.3 (MetaCIC residual-join determinacy budgeted local join)**.** *Every accepted* MetaCICResidualJoinD *packet exposes a local join only through the finite route*

$$T, R \longrightarrow B, K \longrightarrow E, S, L, D, O \longrightarrow H, C, P, N.$$

The route may compare bounded candidate joins for the same displayed residual branch data, but it cannot be read as full parallel diamond, arbitrary closed Church–Rosser, unguarded substitution compatibility, global strong normalization, subject reduction, or host reduction semantics.

Proof. Start with the typed residual window T and the residual budget R. These rows determine which branch data and which candidate addresses can be inspected.

The bounded join row B is downstream of that budget, and the determinacy row K compares only candidates accepted for the same displayed data. Thus determinacy is local to the packet.

The envelope, substitution-boundary, closed-premise, local-diamond, and obstruction rows remain visible after the local comparison. The structural rows preserve this finite route without adding endpoint claims. □

Theorem 5970.4 (MetaCIC residual-join determinacy boundary)**.** *Every confluence-facing consumer of* $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$ *receives only a budgeted residual-local join packet. In particular, the consumer must keep the residual-substitution compatibility row S as a guarded boundary and the obstruction row O as a visible refusal of full Church–Rosser, full parallel diamond, unguarded substitution compatibility, subject reduction, global strong normalization, and host reduction semantics.*

Proof. Read the carrier and apply Definition 5970.3. The route exposes typed residual windows, a residual budget, bounded candidate joins, and local determinacy before any sibling handoff is consumed.

The substitution boundary is not discharged by this packet. It is stored as row S, while the obstruction row O records the endpoint claims that remain unavailable to the consumer.

Transport, replay, provenance, and naming preserve the displayed rows only. Hence the consumer obtains a finite residual-local join boundary, not an endpoint confluence theorem. □

Falsifiable prediction: A $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$ packet that omits the residual-budget row R, the bounded local-join row B, the guarded residual-substitution boundary row S, or the obstruction row O is rejected even if it carries typed residual windows and local-diamond evidence.

Independence witness: metacic-confluence-budget-witness, metacic-residual-substitution-compatibility, metacic-local-diamond-witness, metacic-confluence-residual-envelope, closed-parallel-diamond-premise-ledger: $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds typed residual windows, residual budgets, bounded local joins, determinacy rows, residual envelopes, guarded substitution-boundary rows, closed-premise ledgers, local-diamond evidence, obstruction retention, transport, replay, provenance, and local NameCert data.

The Lean-side hardening surface requires future Nontrivial and FieldFaithful instances for the residual-join determinacy packet. The nontrivial witness separates an accepted packet with displayed residual-budget, bounded-join, guarded-boundary, and obstruction rows from a refused packet missing one of those rows; field faithfulness compares $T, R, B, K, E, S, L, D, O, H, C, P, N$ componentwise so that equal displayed fields are indistinguishable to the local NameCert classifier.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$ packet is generated from typed residual-window rows, residual-budget rows, bounded local-join rows, determinacy rows, residual-envelope rows, guarded residual-substitution boundary rows, closed-premise ledger rows, local-diamond rows, obstruction rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5970.1, Definition 5970.2, Definition 5970.3, and Definition 5970.4 seed the finite residual-local join determinacy boundary for MetaCIC confluence consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim full parallel diamond, arbitrary closed Church–Rosser, unguarded substitution compatibility, subject reduction, global strong normalization, host reduction semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, residual-budget accounting, bounded local-join exposure, candidate determinacy, residual-envelope handoff, guarded substitution-boundary visibility, local-diamond handoff, obstruction retention, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.MetaCICResidualJoinDeterminacyUp.

Chapter 5971

A Concrete Naming Certificate for $\mathsf{BrownianBridge}^{\uparrow}$

Following Doob's treatment of Brownian bridge conditioning, the $\mathsf{BrownianBridge}^{\uparrow}$ packet records the finite BEDC route by which a Brownian path segment is conditioned to meet a displayed endpoint without importing a host conditional-law object. It sits between the Brownian motion carrier of chapter 191, the transition-row discipline of chapter 189, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives stochastic-process consumers one named bridge-conditioning route before a filtration, regular conditional probability, quotient path law, selected endpoint representative, or ambient Gaussian process theorem is available.

Definition 5971.1 (BrownianBridge carrier)**.** *A* $\mathsf{BrownianBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (W, T, E, K, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{Brownian}^{\uparrow}$ *source row, exposing only the finite path-window and step-difference data read by the bridge.*
2. *T is the displayed time-window row, including the initial time, terminal time, and inspected intermediate time coordinates.*
3. *E is the endpoint row, carrying the terminal* $\mathsf{Real}^{\uparrow}$ *target value and the endpoint observation window to which the path is conditioned.*
4. *K is the Markov-transition conditioning row, read through the adjacent finite transition surface of* $\mathsf{MarkovChain}^{\uparrow}$ *rather than through a host conditional distribution.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting path observations, endpoint windows, and conditioning requests.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row for the displayed bridge values, time gaps, and endpoint comparisons.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached by stochastic-process or completion-facing consumers after S and Q have been displayed.*
8. *H stores componentwise* hsame *transport for Brownian source, time, endpoint, transition, stream, regular, Real seal, replay, provenance, and local naming rows.*

9. *C stores displayed* Cont *replay from a bridge-conditioning request through the same finite rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BrownianBridge}^{\uparrow}$, $\mathsf{Brownian}^{\uparrow}$, $\mathsf{MarkovChain}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BrownianBridge}^{\uparrow}}$ *row.*

The classifier compares accepted bridge packets only through these rows. It has no coordinate for a host Brownian bridge process, a filtration, a regular conditional probability, quotient path equality, selected path representative, Gaussian density theorem, or ambient stochastic-process completion.

Lean4 ✓ `BEDC.Derived.BrownianBridgeUp.BrownianBridgeTasteGate single carrier alignment`

Theorem 5971.2 (BrownianBridge NameCert obligations)**.** *For every accepted* $\mathsf{BrownianBridge}^{\uparrow}$ *carrier* $B = (W, T, E, K, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BrownianBridge}^{\uparrow}}$ *surface consists exactly of carrier admission for the Brownian source, displayed time and endpoint rows, Markov-transition conditioning row, StreamName windows, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming. Every bridge-conditioning read is exhausted by those rows and exports no filtration, regular conditional probability, quotient path law, selected endpoint representative, or host Gaussian-process object.*

Proof. Project the carrier of Definition 5971.1. The visible rows are exactly $W, T, E, K, S, Q, R, H, C, P, N$, so any local certificate obligation must be read from one of those displayed coordinates.

The stochastic source is W, T, E, K. The Brownian row supplies the finite path-window data, the time and endpoint rows bind the bridge request, and the Markov-transition row records the conditioning surface without creating a separate host conditional-law object.

The numerical and structural reads are S, Q, R, H, C, P, N. StreamName windows and RegSeqRat readback expose finite observations before the Real seal is visible, while transport, replay, provenance, and local naming keep the same packet stable for downstream consumers. □

Theorem 5971.3 (BrownianBridge conditioning route)**.** *Every Brownian-path, Markov-transition, Real, or Completion consumer of an accepted* $\mathsf{BrownianBridge}^{\uparrow}$ *packet factors through the displayed route*

$$W, T, E \longmapsto K \longmapsto S, Q \longmapsto R, H, C, P, N.$$

Thus the bridge endpoint is read first as finite Brownian and time-window data, then through the Markov conditioning row, and only then through StreamName windows, regular rational readback, and the terminal Real seal.

Proof. Begin with the Brownian source and endpoint data. The rows W, T, E determine the inspected Brownian path windows, the time interval, and the terminal endpoint request; they do not yet expose a completed path law.

Next read the conditioning row K. This row is the only bridge-specific transition surface and is tied to the Markov-chain dependency of chapter 189, so a consumer cannot replace it by a regular conditional probability or a host filtration.

Finally pass through the numerical route. The StreamName row S and RegSeqRat row Q expose finite bridge observations and rational comparisons before the Real seal R is consumed, and H, C, P, N transport, replay, source, and name the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrownianBridge}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrownianBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5971.1, Definition 5971.2, and Definition 5971.3 seed the finite Brownian bridge route over Brownian source windows, time and endpoint rows, Markov conditioning, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{BrownianBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, filtrations, regular conditional probabilities, quotient path laws, selected endpoint representatives, Gaussian density theorems, standard bridge checking, or ambient stochastic-process completion.

Upgrade path: Obligation closure requires checked carrier admission, Brownian source exposure, endpoint conditioning, Markov-transition routing, StreamName windows, RegSeqRat readback, Real sealing, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BrownianBridgeUp.

Constructive story: A BrownianBridge$^{\uparrow}$ packet is generated from Brownian source windows, displayed time and endpoint rows, a Markov-transition conditioning row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5972

A Concrete Naming Certificate for $\mathsf{CauchyModulusFunctor}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyModulusFunctor}^{\uparrow}$ packet records the finite modulus-transport object used when a completion argument must move from dyadic tolerances and stream windows to regular rational readback and a terminal real-name seal. It sits between the dyadic rational core of chapter 291, the finite observation surface of chapter 126, the regular rational sequence packet of chapter 301, and the real-name seal of chapter 38. The packet is a finite carrier for modulus schedules, not a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Definition 5972.1 (Cauchy modulus functor carrier)**.** *A* $\mathsf{CauchyModulusFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, S, R, E, A, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the dyadic precision requests consumed by the modulus schedule.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row on which the schedule is inspected.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows and dyadic tolerances.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R have been displayed.*
5. *A is the admissible-modulus row, recording the finite assignment from requested precision to a displayed window bound.*
6. *T is the transport row, recording how the same assignment is reused by completion-facing consumers without asserting host function equality.*
7. *H records componentwise* hsame *transport for tolerance ledgers, windows, readbacks, Real seals, modulus assignments, transport rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion request through D, S, R, E, A, T, H.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusFunctor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusFunctor}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient real equality, a selected limit, a global choice function, trichotomy, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyModulusFunctorUp.CauchyModulusFunctorTasteGate single carrier alignment`

Theorem 5972.2 (Cauchy modulus functor NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusFunctor}^{\uparrow}$ *carrier* $M = (D, S, R, E, A, T, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusFunctor}^{\uparrow}}$ *surface consists exactly of dyadic tolerance admission through* D*, finite StreamName window admission through* S*, regular rational readback through* R*, terminal Real seal admission through* E*, admissible-modulus assignment through* A*, completion-facing transport through* T*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 5972.1. Its numeric source rows are D, S, R: dyadic precision requests, finite observation windows, and regular rational readbacks.

The Real-facing boundary is the seal row E. Since E is displayed only after D, S, R, the packet exports a finite handoff route and not a detached completeness principle.

The modulus data are A and T. Row A records the precision-to-window assignment, while T records the reuse route seen by completion consumers. Neither row asserts equality of host functions or selects a limiting real.

Finally H, C, P, N transport, replay, source, and name the same rows. Because the carrier contains no coordinate for quotient equality, selected limits, choice, trichotomy, host function extensionality, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Theorem 5972.3 (Cauchy modulus functor completion handoff)**.** *Every Real-and-Completion consumer that reads a* $\mathsf{CauchyModulusFunctor}^{\uparrow}$ *packet receives the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

through the displayed rows $D, S, R, E, A, T, H, C, P, N$*. The handoff may transport a modulus schedule into diagonal-limit or Real-seal arguments, but it does not provide Real completeness, quotient real equality, selected limits, countable choice, trichotomy, or host function extensionality.*

Proof. Use Definition 5972.2. The obligation theorem exposes the dyadic tolerance ledger, StreamName window, RegSeqRat readback, and Real seal as the ordered source of every accepted packet.

The completion-facing rows are then forced through A and T. They record the finite modulus assignment and its transport route, so a diagonal-limit consumer can reuse the same schedule without importing an unrecorded host function.

The structural rows H, C, P, N preserve and name exactly that route. Since no refused coordinate is present in the carrier, the handoff is the displayed finite modulus-transport object and not a proof of $\mathsf{Real}^{\uparrow}$ completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusFunctor}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusFunctor}^{\uparrow}$ packet is generated from DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, a RealUp seal, admissible modulus assignments, completion-facing transport, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusFunctor}^{\uparrow}}$, seed)

Scope closed: Definition 5972.1, Definition 5972.2, and Definition 5972.3 seed the finite modulus-transport handoff from $\mathsf{DyadicRatCore}^{\uparrow}$ through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ to a terminal $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusFunctor}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient real equality, selected limits, countable choice, trichotomy, host function extensionality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window admission, RegSeqRat readback, Real seal nonescape, modulus assignment transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusFunctorUp.

Chapter 5973

A Concrete Naming Certificate for $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity and equicontinuous families, the $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$ packet records the finite-modulus route from compact source nets to one family-level uniform equicontinuity ledger. It realizes the compact-uniform-continuity pressure of chapter 9237 over the compact metric source of chapter 129, continuous-map observations from chapter 130, metric distance rows from chapter 57, finite observation windows from chapter 126, regular rational readback from chapter 301, and the terminal real-name seal of chapter 38. The packet exposes finite moduli and compact-net rows only; it does not import open-cover compactness, selected subcovers, host function equality, quotient map equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5973.1 (FiniteModulusCompactUniformEquicontinuity carrier)**.** *A* FiniteModulusCompactUniformEqui *carrier is a finite* BHist *packet*

$$E = (K, F, M, U, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying inspected centers, coverage radii, and located compact source data.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, exposing the displayed finite family observations used by the equicontinuity request.*
3. *M is the finite-modulus row assigning local source-distance budgets to the family observations at the compact-net centers.*
4. *U is the compact-uniform equicontinuity export row, giving one displayed uniform modulus for all members of the inspected family.*
5. *D is the* $\mathsf{Met}^{\uparrow}$ *distance row for source and target comparisons used by the finite-modulus test.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting compact-net centers, family observations, modulus requests, and distance reads.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for radii, distances, and modulus comparisons.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for compact-uniform consumers after S and Q have been displayed.*

9. *H records componentwise* hsame *transport for compact nets, continuous family rows, finite moduli, uniform export, metric distances, stream windows, regular readback, Real sealing, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from compact-net and family-modulus requests to the uniform equicontinuity export.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an open-cover proof, selected subcover, unrestricted host function equality, quotient map equality, countable choice, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FiniteModulusCompactUniformEquicontinuityUp.FiniteModulusCompactUniformEquicontinu`
`single carrier alignment`

Theorem 5973.2 (FiniteModulusCompactUniformEquicontinuity NameCert obligations)**.** *For every accepted* $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$ *carrier* $E = (K, F, M, U, D, S, Q, R, H, C, P, N)$, *the local* $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through K, continuous-family observations through F, finite-modulus control through M, compact-uniform export through U, metric-distance inspection through D, StreamName windows, RegSeqRat readback, Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 5973.1. The compact source row K, family row F, finite-modulus row M, export row U, and metric row D are the only mathematical coordinates available to the local certificate.

The numerical route is finite. Rows S and Q display the selected observation windows and rational comparisons before R seals the Real-facing distance reads.

The structural rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no row for open covers, selected subcovers, host function equality, quotient map equality, countable choice, or ambient Real completeness, none of those objects is part of the obligation surface. □

Theorem 5973.3 (Finite-modulus compact-uniform equicontinuity route)**.** *Every compact-uniform or Arzela–Ascoli consumer of an accepted* $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$ *packet factors through*

$$K, F, D \longmapsto M \longmapsto S, Q \longmapsto R, U \longmapsto H, C, P, N.$$

Thus the consumer reads compact finite nets, continuous family observations, and metric distances before the finite modulus is exported, and reaches the Real seal and uniform equicontinuity row only after StreamName windows and regular rational readback are visible.

Proof. Start with the compact source, family, and metric rows. The compact-net data in K, the continuous-family observations in F, and the distance comparisons in D determine the finite family-modulus requests.

Next read the finite-modulus row M. It is the only coordinate that binds the finitely many compact-net and family observations to a common source-distance budget, so it cannot be replaced by open-cover compactness or a selected subcover.

Finally pass through S, Q, R, U, H, C, P, N. StreamName windows and RegSeqRat readback expose the numerical comparisons before the Real seal and uniform export are consumed, and the structural rows keep the route inside the same finite packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5973.1, Definition 5973.2, and Definition 5973.3 seed the finite-modulus compact-uniform equicontinuity route over compact metric nets, continuous family observations, metric distances, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, open-cover compactness, selected subcovers, host function equality, quotient map equality, countable choice, standard bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact-net coverage, continuous-family observation, finite-modulus control, compact-uniform export, metric-distance inspection, StreamName windows, RegSeqRat readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformEquicontinuityUp.

Constructive story: A $\mathsf{FiniteModulusCompactUniformEquicontinuity}^{\uparrow}$ packet is generated from CompactMetric finite nets, ContinuousMap family observations, a finite modulus row, a compact-uniform equicontinuity export, Metric distance rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5974

A Concrete Naming Certificate for $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$ packet records the finite boundary data used by a regular-sequence completion proof at the moment where diagonal-limit convergence and uniqueness are read back as a Real-facing handoff. It is anchored in the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet witnesses the boundary that a completion proof may consume; it does not assert a global completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 5974.1 (Real completion boundary witness carrier)**.** *A* $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (D, S, R, E, L, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the diagonal-limit comparison.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected prefixes of the source streams.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational sequence obtained from those windows.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row read only after D, S, R have been displayed.*
5. *L is the convergence-boundary row, recording the finite diagonal-limit evidence available to the completion proof.*
6. *U is the uniqueness-boundary row, recording the finite same-limit comparison available at the same boundary.*
7. *H records componentwise* hsame *transport for tolerance ledgers, windows, readbacks, Real seals, convergence boundary, uniqueness boundary, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a completion-boundary request through D, S, R, E, L, U, H.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionBoundaryWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient completeness theorem, a quotient-real equality principle, countable choice, a selected representative stream, trichotomy, or host-level limit selection.

Lean4 (definition) `BEDC.Derived.RealCompletionBoundaryWitnessUp.RealCompletionBoundaryWitnessUp`
Lean4 ✓ `BEDC.Derived.RealCompletionBoundaryWitnessUp.RealCompletionBoundaryWitnessTasteGate`
`single carrier alignment`

Theorem 5974.2 (Real completion boundary witness NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$ *carrier* $W = (D, S, R, E, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionBoundaryW}}$ *obligations are exactly dyadic tolerance admission through* D*, finite StreamName window admission through* S*, RegSeqRat readback through* R*, Real seal admission through* E*, convergence-boundary display through* L*, uniqueness-boundary display through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 5974.1. The source of the boundary is the finite triple D, S, R: dyadic tolerances, stream windows, and regular rational readback.

The Real-facing row E seals only that displayed readback. It therefore supplies a boundary handoff to $\mathsf{Real}^{\uparrow}$ without adding ambient completeness or quotient equality.

Rows L and U are the two boundary witnesses. The first records convergence evidence for the diagonal-limit route, and the second records the uniqueness comparison available at the same finite boundary.

Finally H, C, P, N preserve, replay, source, and name those rows. Since the carrier has no coordinate for completeness, quotient equality, choice, representative selection, trichotomy, or host-level limit selection, the listed obligations exhaust the certificate surface. □

Theorem 5974.3 (Real completion boundary witness seal handoff)**.** *Every completion proof that consumes a* $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$ *packet reads convergence and uniqueness only through the displayed route*

$$D, S, R, E, L, U, H, C, P, N.$$

In particular, the packet hands a diagonal-limit boundary to $\mathsf{Real}^{\uparrow}$ *consumers after the DyadicRatCore, StreamName, and RegSeqRat rows are displayed, and it refuses ambient Real completeness, quotient real equality, countable choice, selected representatives, trichotomy, and host-level limit selection.*

Proof. Apply Definition 5974.2. The theorem lists the only rows available to a consumer of the packet, with dyadic tolerance, StreamName window, and RegSeqRat readback preceding the Real seal.

The boundary rows L and U are then read under the same seal. They expose convergence and uniqueness evidence as finite carrier data rather than as an external completeness theorem.

The structural rows H, C, P, N keep this route invariant under transport, replay, provenance, and local naming. Because no refused coordinate occurs in the carrier, the seal handoff is exactly the displayed finite boundary route. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionBoundaryWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCompletionBoundaryWitness}^{\uparrow}$ packet is generated from DyadicRatCore tolerance ledgers, StreamName windows, RegSeqRat readback rows, a RealUp seal, convergence-boundary rows, uniqueness-boundary rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionBoundaryWitness}^{\uparrow}}$, seed)

Scope closed: Definition 5974.1, Definition 5974.2, and Definition 5974.3 seed the finite diagonal-limit convergence and uniqueness boundary consumed before a RealUp seal is read.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionBoundaryWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim ambient Real completeness, quotient real equality, countable choice, selected representative streams, trichotomy, host-level limit selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window admission, RegSeqRat readback, Real seal nonescape, convergence and uniqueness boundary exactness, replay, provenance, and local naming for BEDC.Derived.RealCompletionBoundaryWitnessUp.

Chapter 5975

A Concrete Naming Certificate for $\mathsf{RegularCauchyMeshGluing}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy approximation and finite common refinements, the $\mathsf{RegularCauchyMeshGluing}^{\uparrow}$ packet records the mesh-gluing route by which regular Cauchy mesh cofinality, finite Cauchy gluing budgets, and the cofinal modulus seal meet in one displayed NameCert surface. It is grounded in chapter 5839, chapter 1113, and chapter 1198. The packet supplies a finite common-refinement route for Real-completeness consumers without quotient streams, hidden subnets, selected limits, countable choice, host real equality, or ambient completeness.

Definition 5975.1 (Regular Cauchy mesh-gluing carrier)**.** *A* $\mathsf{RegularCauchyMeshGluing}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, G, Q, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RegularCauchyMeshCofinality}^{\uparrow}$ *row exposing the cofinal finite mesh route.*
2. *G is the* $\mathsf{FiniteCauchyGluingBudget}^{\uparrow}$ *row exposing the finite gluing budget, synchronizer, and completion-seal route.*
3. *Q is the* $\mathsf{CofinalModulusSeal}^{\uparrow}$ *row that records the cofinal modulus seal reached by the shared route.*
4. *B is the common-refinement budget aligning the precision request, cofinal mesh depth, gluing synchronizer, and modulus-seal threshold.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the common budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row over those windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by the readback.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, G, Q, B, S, R, D have been displayed.*
9. *H stores componentwise* hsame *transport for every displayed row.*
10. *C stores displayed* Cont *replay from the incoming precision request through*

$$M, G, Q \longmapsto B \longmapsto S, R, D \longmapsto E.$$

11. *P is the* Pkg *provenance over* RegularCauchyMeshGluing$^{\uparrow}$, RegularCauchyMeshCofinality$^{\uparrow}$, FiniteCauchyGluingBud CofinalModulusSeal$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{RegularCauchyMeshGluing}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, G, Q, B, S, R, D, E, H, C, P, N$. It has no coordinate for quotient streams, hidden subnets, selected limits, countable-choice schedules, host real equality, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RegularCauchyMeshGluingUp.RegularCauchyMeshGluingTasteGate single carrier alignment`

Theorem 5975.2 (Regular Cauchy mesh-gluing NameCert obligations)**.** *For every accepted* RegularCauchyMeshGlu *carrier*

$$K = (M, G, Q, B, S, R, D, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyMeshGluing}^{\uparrow}}$ *surface consists exactly of carrier admission, cofinal mesh exposure through M, finite gluing-budget exposure through G, cofinal modulus-seal exposure through Q, common-refinement alignment through B, StreamName window exposure through S, RegSeqRat readback through R, DyadicRatCore tolerance visibility through D, terminal Real seal admission through E, transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 5975.1. The accepted packet displays exactly the mesh cofinality, gluing budget, cofinal modulus seal, common budget, windows, readback, tolerance, Real seal, transport, replay, provenance, and naming rows.

The three source rows M, G, Q are read before any Real-facing seal. The budget row B is the single meeting point for the mesh depth, gluing synchronizer, and modulus threshold.

Rows S, R, D, E are downstream of that budget, and H, C, P, N preserve and name the route. Since no additional coordinate exists, the listed rows exhaust the local NameCert surface. □

Theorem 5975.3 (Regular Cauchy mesh-gluing common-refinement handoff)**.** *Every Real-completeness or diagonal-limit consumer of an accepted* RegularCauchyMeshGluing$^{\uparrow}$ *carrier receives its seal-facing data only through the finite route*

$$M, G, Q \longmapsto B \longmapsto S, R, D \longmapsto E, H, C, P, N.$$

Thus the consumer must pass through mesh cofinality, finite Cauchy gluing, the cofinal modulus seal, StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, and the RealUp seal before exporting a completion-facing conclusion.

Proof. Start from a Real-completeness consumer request. The carrier exposes no direct seal row until the continuation route has displayed M, G, Q.

The common budget B aligns the cofinal mesh, gluing synchronizer, and modulus-seal threshold. This forces the StreamName windows S, RegSeqRat readback R, and Dyadic tolerance row D to share the same finite refinement.

Only then can the Real seal E be read. Transport, replay, provenance, and local naming preserve the same route, while quotient streams, hidden subnets, selected limits, and choice-based completion objects are absent from the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyMeshGluing}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyMeshGluing}^{\uparrow}$ packet is generated from regular Cauchy mesh cofinality, a finite Cauchy gluing budget, a cofinal modulus seal row, a common-refinement budget, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a RealUp seal, transport, replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshGluing}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5975.1, Definition 5975.2, and Definition 5975.3 seed the finite common-refinement route from mesh cofinality and gluing budgets to the cofinal modulus seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyMeshGluing}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, hidden subnets, selected limits, countable choice, host real equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal mesh exposure, finite gluing-budget exposure, cofinal modulus-seal exposure, common-refinement alignment, finite readback, tolerance visibility, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyMeshGluingUp.

Chapter 5976

A Concrete Naming Certificate for $\mathsf{BrouwerSpreadBar}^{\uparrow}$

Following Brouwer and Bishop on spreads, bars, and finite choice-sequence observation, the $\mathsf{BrouwerSpreadBar}^{\uparrow}$ packet records the finite route by which an admitted spread prefix, a decidable bar row, and a lawless boundary read are handed to Real and Completion-facing consumers. It sits between the spread carrier of chapter 1677, the barred-spread induction surface of chapter 1675, and the finite observation boundary of chapter 1495. The packet names finite bar/spread data only; it does not select an infinite branch, assert countable choice, identify quotient stream tails, close fan compactness, or supply ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5976.1 (Brouwer spread-bar carrier)**.** *A* $\mathsf{BrouwerSpreadBar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, B, R, L, W, A, H, C, P, N)$$

with the following displayed rows:

1. *S is the accepted* $\mathsf{BrouwerSpread}^{\uparrow}$ *finite prefix row, including its decidable prefix-tree admission and StreamName scheduling.*
2. *B is the finite bar row over the prefixes displayed by S, with explicit* $\mathsf{Bool}^{\uparrow}$ *bar decisions and monotone extension checks.*
3. *R is the barred-spread replay row inherited from* $\mathsf{BrouwerBarInduction}^{\uparrow}$*, restricted to the finite children displayed in S.*
4. *L is the lawless boundary row. It records only finite observed prefixes and refuses a law selecting all future answers.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window opened by the common spread/bar/lawless route.*
6. *A is the analytic handoff row through* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal when a Real or Completion consumer reads the packet.*
7. *H stores componentwise* hsame *transport for the spread prefix, bar decision, induction replay, lawless boundary, window, analytic handoff, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a finite prefix request through*

$$S, B, R, L \longmapsto W \longmapsto A.$$

9. *P is the* Pkg *provenance over* $\mathsf{BrouwerSpreadBar}^{\uparrow}$, $\mathsf{BrouwerSpread}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{LawlessSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerSpreadBar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, R, L, W, A, H, C, P, N$. It has no coordinate for completed branches, countable-choice selectors, quotient stream equality, fan compactness, selected limits, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.BrouwerSpreadBarUp.BrouwerSpreadBarTasteGate single carrier alignment`

Theorem 5976.2 (Brouwer spread-bar NameCert obligations)**.** *For every accepted* $\mathsf{BrouwerSpreadBar}^{\uparrow}$ *carrier*

$$K = (S, B, R, L, W, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BrouwerSpreadBar}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed spread prefix, finite bar row, barred-spread replay, lawless boundary row, StreamName window, analytic handoff, transport, continuation, provenance, and local naming rows; classifier agreement on the same rows; and non-escape requiring every public read to factor through $S, B, R, L, W, A, H, C, P, N$.*

Proof. Carrier admission is the tuple of Definition 5976.1. The accepted packet exposes only the spread prefix, bar decision, barred-spread replay, lawless boundary, window, handoff, transport, continuation, provenance, and local naming rows.

Classifier agreement is therefore rowwise agreement over the same tuple. The spread component is read through chapter 1677, the bar replay through chapter 1675, and the lawless boundary through chapter 1495.

The non-escape clause follows from the absence of any additional coordinate. Since the carrier does not store a completed branch, a choice selector, quotient stream equality, compactness, selected limits, or ambient completeness, the listed rows exhaust the local certificate surface. □

Theorem 5976.3 (Brouwer spread-bar finite bar handoff)**.** *Every Real or Completion-facing read of an accepted* $\mathsf{BrouwerSpreadBar}^{\uparrow}$ *carrier must first display the finite route*

$$S, B, R, L \longmapsto W \longmapsto A, H, C, P, N.$$

Thus the consumer receives only a finite barred-prefix handoff from the spread row, bar row, barred-spread replay row, and lawless boundary row before any $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Start with a public read of the packet. The only route to the analytic handoff row A passes through the continuation row C.

Replay C. It first exposes the spread prefix S, then the finite bar decisions B, the barred-spread replay R, and the lawless boundary row L. Only after these rows are displayed can the read open the StreamName window W and the analytic handoff A.

Transport and provenance are supplied by H, P, N. Since the carrier lacks any branch-selection or quotient-stream coordinate, the handoff is exactly the finite route stated above. □

Theorem 5976.4 (Brouwer spread-bar lawless boundary)**.** *The lawless component of an accepted* $\mathsf{BrouwerSpreadBar}^{\uparrow}$ *carrier is boundary data for finite observations only. It may certify that the currently displayed prefixes meet the bar route, but it cannot be used to construct a law for all future values, a completed spread branch, a countable-choice schedule, or a Real-completeness witness.*

Proof. Read the lawless row L inside the carrier. By definition it contains only the finite observed prefixes already synchronized with the spread row S and bar row B.

Use the finite handoff theorem, Definition 5976.3. Every downstream read must pass from S, B, R, L to the finite StreamName and analytic rows before exporting a Real or Completion-facing conclusion.

The packet has no coordinate for a future-value law, completed branch, countable-choice schedule, or completeness witness. Therefore the lawless component is a finite boundary row and not a generator of the excluded data. □

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerSpreadBar}\uparrow}$.

Constructive story: A $\mathsf{BrouwerSpreadBar}^{\uparrow}$ packet is generated from a Brouwer spread finite prefix row, a finite bar row, barred-spread replay, a lawless finite-observation boundary, StreamName windows, analytic handoff rows, componentwise transport, displayed continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerSpreadBar}\uparrow}, \mathsf{seed})$

Scope closed: Definition 5976.1, Definition 5976.2, Definition 5976.3, and Definition 5976.4 seed the finite spread-bar route from Brouwer spread prefixes and bar replay to lawless finite observations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerSpreadBar}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed branches, countable choice, quotient stream equality, fan compactness, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier agreement, finite bar handoff, lawless-boundary nonescape, transport, continuation, provenance, and local naming for BEDC.Derived.BrouwerSpreadBarUp.

Chapter 5977

A Concrete Naming Certificate for $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ packet records the finite stability surface needed when a located Cauchy completion proof is replayed through dyadic tolerances, stream windows, regular rational readback, locatedness evidence, and the terminal real seal. It is anchored in the DyadicRatCore ledger of chapter 291, the finite StreamName surface of chapter 126, the RegSeqRat readback of chapter 301, the LocatedReal observation row of chapter 1933, the terminal Real seal of chapter 38, and the located-real completion route of chapter 3488. The packet witnesses stability of displayed completion data; it does not introduce quotient streams, host limits, choice, trichotomy, or ambient completeness.

Definition 5977.1 (Bishop located Cauchy completion stability carrier)**.** *A* $\mathsf{BishopLocatedCauchyCompletionStabili}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, R, L, E, T, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-window ledger naming the radii and comparison budgets used by the stability check.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at those tolerances.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular Cauchy readback row obtained from the same displayed windows.*
4. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *locatedness row attached to that readback.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R, L have been displayed.*
6. *T records componentwise* hsame *transport for tolerance windows, stream windows, readback rows, locatedness rows, the Real seal, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from each stability request through D, S, R, L, E, T, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient-stream representatives, host-level limits, selected representatives, countable choice, trichotomy, apartness by host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BishopLocatedCauchyCompletionStabilityUp.BishopLocatedCauchyCompletionStab`

Theorem 5977.2 (Bishop located Cauchy completion tolerance-window stability)**.** *For every accepted* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ *carrier* $Q = (D, S, R, L, E, T, C, P, N)$*, any replay of a completion request keeps the DyadicRatCore tolerance row* D *fixed before reading the StreamName window* S*, the RegSeqRat readback* R*, the LocatedReal row* L*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Proof. Project the carrier of Definition 5977.1. The first displayed analytic row is the tolerance ledger D, and every later analytic row is indexed by the same completion request.

Apply the located-real completion route of Definition 3488.3. It identifies the order D, S, R, L, E for displayed completion data.

The stability packet adds T, C, P, N only for transport, replay, provenance, and local naming. Since none of those rows contains a second tolerance ledger or a host limit selector, replay cannot change the tolerance window while preserving the same packet. □

Theorem 5977.3 (Bishop located Cauchy completion readback stability)**.** *For every accepted* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ *carrier* $Q = (D, S, R, L, E, T, C, P, N)$*, the regular Cauchy readback exposed to a completion consumer is the row* R *read from the displayed* D, S *window route. No consumer can replace* R *by a quotient-stream representative, selected host limit, or ambient equality witness while remaining inside the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}}$ *surface.*

Proof. Read D and S from the carrier and then the readback row R. These are finite rows of the same BHist packet.

Apply Definition 3488.5. The Bishop located-real completion chapter exposes its Cauchy representation only through D, S, R, L before the terminal seal.

The stability rows T, C, P, N preserve and replay that displayed readback. Because the carrier has no quotient-stream, selected-limit, or host-equality coordinate, replacing R by such data would change the packet rather than witnessing stability. □

Theorem 5977.4 (Bishop located Cauchy completion locatedness preservation)**.** *For every accepted* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ *carrier* $Q = (D, S, R, L, E, T, C, P, N)$*, locatedness is preserved only as the displayed row* L *attached to the same* D, S, R *readback route. A replay may transport* L *through* T *and* C*, but it cannot derive locatedness from trichotomy, host apartness, choice, or an external order-completeness principle.*

Proof. Project L from the carrier after the tolerance, window, and readback rows. The row is part of the same finite packet and is not a detached located-real object.

Apply Definition 3488.4. It forces every located-real completion consumer through D, S, R, L before the Real seal.

Rows T and C transport and replay that row, while P and N source and name it. The refused trichotomy, host apartness, choice, and external completeness objects are absent from the carrier, so they cannot supply locatedness inside the certificate. □

Theorem 5977.5 (Bishop located Cauchy completion Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal exported by an accepted* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ *packet factors through the finite route* $D, S, R, L, E, T, C, P, N$*. The seal exports stability of the displayed located Cauchy completion data, but it exports no quotient stream, selected host limit, host real equality test, countable-choice selector, trichotomy split, apartness shortcut, or ambient Real completeness theorem.*

Proof. Project the carrier and isolate the terminal seal row E. It occurs after the displayed tolerance, window, readback, and locatedness rows.

Apply Definition 3488.6. The underlying located-real completion seal cannot be read as quotient equality, selected host limits, choice, trichotomy, or ambient completeness.

The stability packet preserves the same seal through T, C, P, N. Since no refused object has a carrier coordinate, the exported $\mathsf{Real}^{\uparrow}$ row remains the finite stability seal for D, S, R, L. □

Theorem 5977.6 (Bishop located Cauchy completion transport replay provenance)**.** *For every accepted* $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ *carrier* $Q = (D, S, R, L, E, T, C, P, N)$*, all stability claims are invariant only through the displayed transport row* T*, continuation replay row* C*, package provenance row* P*, and local naming row* N*. These rows preserve the finite route* D, S, R, L, E *and do not add analytic content beyond the displayed completion data.*

Proof. Project the structural rows T, C, P, N from the carrier. They are listed after the analytic rows and are therefore read as preservation, replay, provenance, and naming data for D, S, R, L, E.

Apply the preceding four stability theorems. Each one identifies the analytic route that must be preserved: fixed tolerances, regular readback, locatedness, and the terminal Real seal.

Thus T, C, P, N certify replay of the same finite route rather than a broader completion theorem. Any additional quotient, host-limit, choice, trichotomy, apartness, or ambient-completeness claim would require a carrier coordinate not present in Q. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}$ packet is generated from DyadicRatCore tolerance windows, StreamName finite windows, RegSeqRat regular Cauchy readback, LocatedReal locatedness rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5977.1, Definition 5977.2, Definition 5977.3, Definition 5977.4, Definition 5977.5, and Definition 5977.6 seed the finite stability route from displayed located Cauchy completion data to the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedCauchyCompletionStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-stream completion, host-level limits, selected representatives, countable choice, trichotomy, host apartness, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, tolerance-window stability, regular Cauchy readback stability, locatedness preservation, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedCauchyCompletionStabilityUp.

Chapter 5978

CauchyMatrixUp NameCert Construction

Following Bishop and Bridges, the $\mathsf{CauchyMatrix}^{\uparrow}$ packet records a two-index Cauchy matrix certificate for the L10 Real and Completion route. It sits after the dyadic tolerance core of chapter 291, the finite stream-window interface of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names row and column regularity, diagonal extraction, transport, replay, provenance, and local naming as finite BHist data; it does not assert quotient equality of completed real names, select a limiting row, use countable choice, or invoke ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5978.1 (Cauchy matrix carrier)**.** *A* $\mathsf{CauchyMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, C, W, D, Q, E, H, K, P, N)$$

with the following displayed rows:

1. *R is the row-regularity ledger, recording the finite row windows whose tails are read as regular rational Cauchy data.*
2. *C is the column-regularity ledger, recording the finite column windows whose tails satisfy the same tolerance discipline.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *two-index window schedule, listing every inspected pair (i, j) together with the row, column, and diagonal slices visible to the consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used for every row, column, and diagonal comparison.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the diagonal schedule of W and the tolerance row D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after R, C, W, D, Q have been displayed.*
7. *H records componentwise* hsame *transport for row regularity, column regularity, windows, dyadic tolerances, diagonal readback, Real sealing, replay, provenance, and naming.*
8. *K records displayed* Cont *replay from a completion request through $R, C, W, D, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyMatrix}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMatrix}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, C, W, D, Q, E, H, K, P, N$. It has no coordinate for a quotient real, a chosen limiting row or column, countable choice over rows, detached completion data, host equality of streams, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyMatrixUp.CauchyMatrixTasteGate single carrier alignment`

Theorem 5978.2 (Cauchy matrix NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMatrix}^{\uparrow}$ *carrier*

$$M = (R, C, W, D, Q, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission, row regularity, column regularity, finite two-index window scheduling, dyadic tolerance accounting, diagonal regular rational readback, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every L10 Real or Completion consumer must read R, C, W, D, Q, E before consuming H, K, P, N.*

Proof. Project the accepted carrier of Definition 5978.1. Its analytic data are exactly the row ledger R, the column ledger C, the two-index window schedule W, the dyadic tolerance ledger D, the diagonal readback row Q, and the Real seal E.

The structural rows do not add new analytic content. Rows H and K transport and replay the displayed route, while P and N record provenance and local naming.

Since the carrier has no coordinate for quotient equality, selected limiting rows, countable choice, detached completion data, host stream equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 5978.3 (Cauchy matrix row and column regularity)**.** *For every accepted* $\mathsf{CauchyMatrix}^{\uparrow}$ *carrier $M = (R, C, W, D, Q, E, H, K, P, N)$, a completion-facing read of the matrix can inspect row tails and column tails only through the finite schedule W and the dyadic tolerance row D. Thus row regularity and column regularity are synchronized as displayed packet data before any diagonal readback or Real seal is visible.*

Proof. By Definition 5978.2, every consumer reads the row ledger, column ledger, window schedule, and dyadic tolerance row before the diagonal and seal rows.

The row route is therefore R, W, D, and the column route is C, W, D. Both routes share the same finite two-index schedule and the same tolerance ledger.

The diagonal row Q and terminal seal E occur only after those routes. Hence neither row regularity nor column regularity can be consumed as an external host property; both are part of the displayed finite packet. □

Theorem 5978.4 (Cauchy matrix diagonal Real handoff)**.** *For every accepted* $\mathsf{CauchyMatrix}^{\uparrow}$ *carrier $M = (R, C, W, D, Q, E, H, K, P, N)$, the diagonal handoff to* $\mathsf{Real}^{\uparrow}$ *factors through the finite route*

$$R, C, W, D \longmapsto Q \longmapsto E.$$

Consequently the packet exports a regular rational diagonal name and then a terminal Real seal, without exporting quotient stream equality, a selected limit, countable choice over rows, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from Definition 5978.3. The row and column ledgers are synchronized through the displayed two-index windows and dyadic tolerance row before the diagonal readback is visible.

The diagonal readback row Q is a $\mathsf{RegSeqRat}^{\uparrow}$ row over that same window-and-tolerance surface. It is not an independent sequence chosen outside the matrix packet.

Finally the Real seal E is downstream of Q. Rows H, K, P, N preserve and name this route, so a consumer receives only the displayed diagonal handoff $R, C, W, D \mapsto Q \mapsto E$ and no excluded host completion principle. □

Falsifiable prediction: Every accepted $\mathsf{CauchyMatrix}^{\uparrow}$ consumer that claims a diagonal $\mathsf{Real}^{\uparrow}$ handoff while omitting one of $R, C, W, D, Q, E, H, K, P, N$ refutes the carrier surface of Definition 5978.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{CauchyMatrix}^{\uparrow}$ is not bijective to any listed sibling because only it simultaneously carries row regularity, column regularity, a two-index window schedule, a shared dyadic tolerance ledger, diagonal regular readback, terminal Real sealing, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMatrix}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5978.1, Definition 5978.2, Definition 5978.3, and Definition 5978.4 seed the finite two-index matrix route from row and column regularity to diagonal regular readback and terminal Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected matrix limits, countable choice over rows, host stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, row and column regularity exactness, two-index window scheduling, dyadic tolerance synchronization, diagonal RegSeqRat readback, terminal Real seal scope, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMatrixUp.

Constructive story: A $\mathsf{CauchyMatrix}^{\uparrow}$ packet is generated from row regularity, column regularity, StreamName two-index windows, DyadicRatCore tolerances, RegSeqRat diagonal readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 5979

A Concrete Naming Certificate for $\mathsf{GrothendieckGroup}^{\uparrow}$

Following Grothendieck's group-completion construction, the $\mathsf{GrothendieckGroup}^{\uparrow}$ packet records the standard passage from a commutative monoid to its universal abelian group as finite BEDC NameCert data. It is anchored by the monoid carrier of chapter 40, the abelian-group target surface of chapter 42, the semiring-to-additive-completion route of chapter 327, and the generator discipline of chapter 345. The packet names a displayed difference-pair congruence and its universal additive handoff; it does not introduce a primitive quotient group, ambient setoid completion, or external localization theorem.

Definition 5979.1 (Grothendieck group carrier)**.** *A* $\mathsf{GrothendieckGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, X, D, Q, A, U, H, K, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Monoid}^{\uparrow}$ *row, exposing the commutative additive operation, zero row, and BEDC monoid laws used by the completion.*
2. *X is the generator row, recording the formal generator map $x \mapsto [x, 0]$ from the monoid surface into the completion packet.*
3. *D is the difference-pair row, whose displayed elements are finite pairs (a, b) read as the formal difference $a - b$.*
4. *Q is the congruence row, generated by the finite relation*

   $$(a, b) \sim (c, d)$$

   whenever a displayed witness t satisfies $a + d + t = c + b + t$ inside the source monoid row M.
5. *A is the additive-group row, carrying zero $[(0, 0)]$, inverse $[(a, b)] \mapsto [(b, a)]$, and addition $[(a, b)] + [(c, d)] = [(a + c, b + d)]$ after the congruence row Q has been displayed.*
6. *U is the universal handoff row from M to any accepted* $\mathsf{AbGroup}^{\uparrow}$ *row: a monoid homomorphism out of M factors through X, D, Q, A by sending $[(a, b)]$ to $f(a) - f(b)$ in the target abelian group.*
7. *H records componentwise* hsame *transport for the monoid row, generator row, difference-pair row, congruence row, additive-group row, universal handoff, replay, provenance, and local naming.*

8. *K records displayed* Cont *replay from a source monoid query through M, X, D, Q, A, U, H.*
9. *P is the* Pkg *provenance over* $\mathsf{GrothendieckGroup}^{\uparrow}$, $\mathsf{Monoid}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Semiring}^{\uparrow}$, $\mathsf{FreeMonoid}^{\uparrow}$, BHist, msame, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GrothendieckGroup}^{\uparrow}}$ *row.*

The classifier accepts a group-completion read only through $M, X, D, Q, A, U, H, K, P, N$. It has no coordinate for a quotient-group primitive, hidden cancellation law, host abelian group equality, external localization object, or global exactness theorem.

Theorem 5979.2 (Grothendieck group obligation surface)**.** *For every accepted* $\mathsf{GrothendieckGroup}^{\uparrow}$ *carrier $G = (M, X, D, Q, A, U, H, K, P, N)$, the local NameCert surface consists exactly of the following finite obligations:*

1. *carrier admission for the source monoid, generator, difference-pair, congruence, additive-group, universal handoff, transport, replay, provenance, and local naming rows;*
2. *generator compatibility, requiring X to send each displayed monoid element a to the displayed pair $(a, 0)$;*
3. *congruence stability, requiring the row Q to be preserved by pairwise addition and by swapping coordinates for inverses;*
4. *additive-group exposure, requiring A to read zero, addition, and inverse only after D and Q have been displayed;*
5. *universal handoff discipline, requiring each accepted handoff into an* $\mathsf{AbGroup}^{\uparrow}$ *row to factor through U and the displayed formula $[(a, b)] \mapsto f(a) - f(b)$;*
6. *structural preservation through H, K, P, N;*
7. *non-escape, excluding primitive quotient groups, hidden cancellation, host equality, external localization data, and unrecorded exactness assumptions.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 5979.1. The rows $M, X, D, Q, A, U, H, K, P, N$ are the only displayed data available to an accepted packet.

The generator row X records the map into formal pairs, while D supplies the pair carrier. The congruence row Q supplies exactly the finite witness relation needed to identify two differences without assuming cancellation in M.

The additive-group row A is then read from D and Q: zero is the class of $(0, 0)$, addition is coordinatewise addition, and inverse swaps the two coordinates. Congruence stability is the obligation that makes these operations insensitive to the displayed equivalence row.

For a target $\mathsf{AbGroup}^{\uparrow}$ packet, the universal handoff row U records the factorization formula $[(a, b)] \mapsto f(a) - f(b)$. The structural rows H, K, P, N transport, replay, source, and name only this displayed route.

Since none of the excluded objects is a carrier coordinate, every accepted group-completion obligation is one of the listed rows and no quotient-group primitive is exported. □

Falsifiable prediction: A $\mathsf{GrothendieckGroup}^{\uparrow}$ consumer that forms inverses or universal maps without displaying the difference-pair row D and congruence row Q is not reading an accepted packet of this chapter.

Independence witness: monoid, abgroup, semiring, freemonoid: $\mathsf{GrothendieckGroup}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a source monoid row, generator row, difference-pair row, congruence row, additive-group row, universal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GrothendieckGroup}^{\uparrow}}$.

Constructive story: A $\mathsf{GrothendieckGroup}^{\uparrow}$ packet is generated from a source MonoidUp row, a generator row, a difference-pair row, a congruence row, an additive-group row, a universal handoff into AbGroupUp targets, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GrothendieckGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5979.1 and Definition 5979.2 seed the finite group-completion carrier and its local NameCert obligations.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GrothendieckGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, a quotient-group primitive, cancellation in the source monoid, host abelian group equality, external localization, or K-theory exactness.

Upgrade path: Obligation closure requires checked carrier admission, congruence stability, additive-group operation preservation, universal handoff uniqueness, Nontrivial GrothendieckGroupUp, and FieldFaithful GrothendieckGroupUp for BEDC.Derived.GrothendieckGroupUp.

Sibling dependency route. The $\mathsf{GrothendieckGroup}^{\uparrow}$ difference-pair congruence is read only after its source monoid, target abelian-group, additive semiring, and free-generator neighbours have been exposed. Concretely, the route consumes the monoid carrier of chapter 40, the abelian-group target surface of chapter 42, the additive semiring route of chapter 327, and the generator discipline of chapter 345. These sibling references are dependency edges for the displayed difference-pair and universal-handoff rows, not additional theorem content.

Chapter 5980

A Concrete Naming Certificate for $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$

Following Bishop, Bridges, and Troelstra–van Dalen on constructive analysis and weak omniscience principles, the $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ packet names the finite boundary where L10 real-completion reads may cite displayed evidence without becoming LPO, LLPO, or choice principles. It sits over the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, the Kleene-tree obstruction of chapter 1622, the Specker sequence surface of chapter 1406, and the finite-prefix choice-sequence row of chapter 1482. The packet refuses decision extraction from real-completion reads while preserving finite witness routes through BHist, hsame, Cont, Pkg, and NameCert rows.

Definition 5980.1 (Weak omniscience boundary carrier)**.** *A* $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (R, E, K, S, Q, F, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row naming the finite regular-rational windows available to an L10 Real or Completion consumer.*
2. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed regular-rational windows have been read.*
3. *K is the* $\mathsf{KleeneTree}^{\uparrow}$ *obstruction row, carrying finite prefix-tree evidence without selecting an infinite branch.*
4. *S is the* $\mathsf{SpeckerSequence}^{\uparrow}$ *row, carrying monotone bounded finite-window evidence without exposing a located supremum.*
5. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *prefix row, carrying displayed choice-sequence answers only on the inspected finite window.*
6. *F is the refusal row recording that LPO tests, LLPO splits, selected limits, quotient stream equality, trichotomy, countable choice, open-cover compactness, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness are not coordinates of the packet.*
7. *H records componentwise* hsame *transport for R, E, K, S, Q, F.*
8. *C records displayed* Cont *replay from a Real or Completion request through the regular-rational readback, terminal seal, obstruction rows, prefix rows, and refusal row.*

9. *P is the* Pkg *provenance over* $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{WeakOmniscienceBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, E, K, S, Q, F, H, C, P, N$. It has no hidden coordinate from which a real-completion read can decide LPO, split an LLPO instance, choose a limit, identify quotient streams, invoke countable choice, extract open-cover compactness, or complete ambient $\mathsf{Real}^{\uparrow}$.

Lean4 (definition) `BEDC.Derived.WeakOmniscienceBoundaryUp.WeakOmniscienceBoundaryUp`

Theorem 5980.2 (Weak omniscience boundary NameCert obligations)**.** *For every accepted* WeakOmniscienceBoun *carrier*

$$W = (R, E, K, S, Q, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WeakOmniscienceBoundary}^{\uparrow}}$ *surface consists exactly of regular-rational readback admission through R, terminal real-seal handoff through E, finite tree-obstruction readback through K, Specker finite-window readback through S, choice-sequence prefix readback through Q, refusal nonescape through F, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.WeakOmniscienceBoundaryUp.WeakOmniscienceBoundaryCarrier namecert obligations`

Proof. Project the carrier of Definition 5980.1. The Real and Completion route first exposes the regular-rational row R and then the terminal seal row E.

The obstruction rows are finite. The Kleene row K supplies only displayed prefix-tree data, the Specker row S supplies only displayed monotone bounded windows, and the choice-sequence row Q supplies only displayed finite-prefix answers.

The refusal row F binds the missing principles to absent coordinates rather than to extra assumptions. Transport by H, replay by C, provenance by P, and local naming by N preserve the same displayed rows and add no LPO test, LLPO split, selected limit, quotient stream equality, trichotomy branch, countable-choice selector, open-cover compactness principle, or ambient completeness field. □

Theorem 5980.3 (Weak omniscience boundary real-completion nonescape)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{KleeneTree}^{\uparrow}, \quad \mathsf{SpeckerSequence}^{\uparrow}, \quad \mathsf{ChoiceSequence}^{\uparrow}$$

together with the refusal row and the structural rows H, C, P, N. The route may preserve finite witness evidence from the sibling packets, but it cannot extract LPO, LLPO, selected limits, quotient stream equality, trichotomy, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness from a real-completion read.*

Proof. Apply Definition 5980.2. It fixes the consumer route to the displayed regular-rational row, terminal real seal, three finite sibling rows, refusal row, and structural rows.

The three sibling rows do not store decision principles. The Kleene row preserves finite prefix-tree obstruction data, the Specker row preserves finite monotone bounded windows without a located supremum, and the choice-sequence row preserves only displayed prefix answers.

It remains to inspect the Real and Completion handoff. Since the terminal $\mathsf{Real}^{\uparrow}$ seal is reached only through R and is followed by the explicit refusal row F, any attempted extraction of LPO, LLPO, selected limits, quotient equality, trichotomy, choice, compactness, or ambient completeness asks for a coordinate outside the accepted carrier. □

Falsifiable prediction: Every accepted $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ Real or Completion read factors through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{SpeckerSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, the refusal row, componentwise hsame, displayed Cont, Pkg, and the local NameCert row before the consumer may cite the boundary.

Independence witness: regseqrat, real, kleene-tree, speckersequence, choice-sequence: WeakOmniscienceBoundar is not bijective to any listed sibling because it jointly carries regular-rational readback, a terminal Real seal, finite Kleene obstruction data, Specker finite-window data, choice-sequence prefix data, an explicit weak-omniscience refusal row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{WeakOmniscienceBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ packet is generated from RegSeqRat readback, a RealUp seal, KleeneTree finite obstruction rows, SpeckerSequence finite-window rows, ChoiceSequence prefix rows, an explicit refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeakOmniscienceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5980.1, Definition 5980.2, and Definition 5980.3 seed the finite weak-omniscience boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeakOmniscienceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close LPO, LLPO, selected limits, quotient stream equality, trichotomy, countable choice, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, RegSeqRat readback exactness, Real-seal handoff, Kleene-tree obstruction stability, Specker finite-window stability, ChoiceSequence prefix stability, and refusal nonescape for BEDC.Derived.WeakOmniscienceBoundaryUp.

Chapter 5981

A Concrete Naming Certificate for $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$

Following Bourbaki and Bishop–Bridges, the $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$ packet records the finite bridge by which a metric Cauchy-filter presentation is read through StreamName windows, dyadic tolerance ledgers, RegSeqRat readback, and a terminal Real seal. It sits over the metric source of chapter 57, the finite observation route of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The bridge names only the displayed finite handoff; it does not identify quotient filter classes or select a limit point.

Definition 5981.1 (Metric Cauchy-filter bridge carrier)**.** *A* $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, F, S, D, R, E, H, C, P, N)$$

with the following rows:

1. *X is the metric source row, exposing only the displayed distance and finite ball-comparison data allowed by the* $\mathsf{Met}^{\uparrow}$ *packet.*
2. *F is the filter-base row, listing the finite inhabited tails and refinement requests that witness the Cauchy-filter presentation.*
3. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row extracted from the filter base.*
4. *D is the dyadic tolerance ledger read through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and attached to the displayed filter windows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational tail windows produced from S and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the displayed RegSeqRat readback.*
7. *H records componentwise* hsame *transport for the metric source, filter base, StreamName windows, dyadic ledger, RegSeqRat readback, and Real seal rows.*
8. *C records displayed* Cont *replay from a Cauchy-filter request through the finite StreamName, DyadicRatCore, RegSeqRat, and Real seal route.*
9. *P is the* Pkg *provenance over* $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyFilterBridge}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, selected limits, countable-choice schedules, host real equality, or ambient Real completeness.

Lean4 (definition) `BEDC.Derived.MetricCauchyFilterBridgeUp.MetricCauchyFilterBridgeUp`
Lean4 ✓ `BEDC.Derived.MetricCauchyFilterBridgeUp.MetricCauchyFilterBridgeTasteGate single carrier alignment`

Theorem 5981.2 (Metric Cauchy-filter bridge NameCert obligations)**.** *Every accepted* $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$ *packet binds exactly the rows displayed in Definition 5981.1: metric source visibility for* X*, finite filter-base visibility for* F*, finite StreamName windows for* S*, dyadic tolerance ledger visibility for* D*, RegSeqRat readback for* R*, terminal Real seal visibility for* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, sibling provenance through* P*, and local naming through* N*.*

The obligation surface exports no quotient filter class, selected limit, countable-choice selector, host real equality, ambient metric completeness datum, or ambient Real completeness principle.

Proof. Start with an accepted carrier $M = (X, F, S, D, R, E, H, C, P, N)$. The metric source row X and filter-base row F provide the only admissible Cauchy-filter data, and both are finite displayed rows.

Read the completion route through S, D, R, E. The $\mathsf{StreamName}^{\uparrow}$ row supplies finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the dyadic tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular rational readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal.

The support rows H, C, P, N transport, replay, source, and name those finite rows componentwise. Since no quotient, choice, selected-limit, host-equality, or completeness coordinate occurs in the carrier, the accepted packet binds exactly the displayed NameCert obligations. □

Theorem 5981.3 (Metric Cauchy-filter bridge real-completion handoff)**.** *Every terminal completion consumer of an accepted* $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$ *packet receives its* $\mathsf{Real}^{\uparrow}$ *seal only through the finite route*

$$\mathsf{Met}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The filter-base row can request narrower displayed tails and dyadic tolerances, but the handoff remains a finite StreamName, DyadicRatCore, RegSeqRat, and Real seal route transported by hsame*, replayed by* Cont*, sourced by* Pkg*, and named by the local* $\mathsf{NameCert}_{\mathsf{MetricCauchyFilterBridge}^{\uparrow}}$ *row.*

The handoff refuses quotient filter classes, selected limits, countable choice, host real equality, selected Real endpoints, metric-completeness data, and ambient Real completeness.

Proof. Project the accepted carrier. The only route from the metric Cauchy-filter rows to a real-name endpoint is the displayed sequence of S, D, R, E rows, with X and F supplying the source and filter-base requests.

Use the StreamName and DyadicRatCore rows to turn each accepted finite tail request into a dyadic tolerance window. The RegSeqRat row reads that window as a regular rational tail, and the Real row seals only that readback.

Finally use the support rows. Componentwise hsame keeps the route stable under displayed comparison, Cont replays the finite handoff, Pkg records the sibling provenance, and the local NameCert row names the bridge. The excluded quotient, choice, selected-limit, host-equality, endpoint-selection, metric-completeness, and Real-completeness rows are absent from the carrier, so no completion consumer can read them. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCauchyFilterBridge}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCauchyFilterBridge}^{\uparrow}$ packet is generated from a MetricUp source row, a finite Cauchy-filter base row, StreamName windows, a DyadicRatCore tolerance ledger, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCauchyFilterBridge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5981.1, Definition 5981.2, and Definition 5981.3 seed the finite metric Cauchy-filter bridge into the RegSeqRat and Real completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCauchyFilterBridge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient filter classes, selected limits, countable choice, host real equality, selected Real endpoints, ambient metric completeness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric-source admission, filter-base window extraction, dyadic tolerance readback, RegSeqRat handoff, Real seal non-escape, and ledger stability for BEDC.Derived.MetricCauchyFilterBridgeUp.

Chapter 5982

A Concrete Naming Certificate for CauchyCompletionReflectorTriangleHandoff$^{\uparrow}$

Following Bishop–Bridges constructive completion and Mac Lane's adjunction language, the CauchyCompletionRefle packet records the finite handoff by which the Cauchy completion reflector and its triangle identities become a local L10 NameCert surface. It sits over the reflector of chapter 1585, the triangle packet of chapter 3946, the unit of chapter 1527, the counit of chapter 1539, the regular rational source of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the finite coherence route; it does not assert quotient completion, selected representatives, countable choice, host equality, or ambient Real$^{\uparrow}$ completeness.

Definition 5982.1 (CauchyCompletionReflectorTriangleHandoff carrier)**.** *A* CauchyCompletionReflectorTriangleHa *carrier is a finite* BHist *packet*

$$H = (F, T, U, K, S, D, R, E, J, C, P, N)$$

with the following displayed rows:

1. *F is the* CauchyCompletionReflector$^{\uparrow}$ *row exposing the reflector as a finite completion-facing surface.*
2. *T is the* CompletionReflectorTriangle$^{\uparrow}$ *row binding the unit, counit, and idempotent triangle around the same reflector.*
3. *U is the* CauchyCompletionUnit$^{\uparrow}$ *row for source admission into the completion-facing route.*
4. *K is the* CauchyCompletionCounit$^{\uparrow}$ *row for already completed or real-sealed return reads.*
5. *S is the* StreamName$^{\uparrow}$ *finite-window row through which source and completed names are observed.*
6. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used by the same windows before any real seal is read.*
7. *R is the* RegSeqRat$^{\uparrow}$ *regular readback row carried from source names to completion-facing reads.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row available only after the window, tolerance, and regular readback rows are displayed.*
9. *J records componentwise* hsame *transport for reflector, triangle, unit, counit, window, tolerance, readback, real seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay along*

$$F, T, U, K \longmapsto S, D, R, E \longmapsto J, P, N.$$

11. *P is the* Pkg *provenance over* CauchyCompletionReflectorTriangleHandoff$^{\uparrow}$, CauchyCompletionReflector$^{\uparrow}$, CompletionReflectorTriangle$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionCounit$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* NameCert$_{\text{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for quotient completion, a selected representative, a countable-choice schedule, host equality, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyCompletionReflectorTriangleHandoffUp.CauchyCompletionReflectorTriangleHandof`
`single carrier alignment`

Theorem 5982.2 (CauchyCompletionReflectorTriangleHandoff NameCert obligations)**.** *For every accepted* CauchyCompletionReflectorTriangleHandoff$^{\uparrow}$ *carrier $H = (F, T, U, K, S, D, R, E, J, C, P, N)$, the local* NameCert$_{\text{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed reflector, triangle, unit, counit, stream window, dyadic tolerance, regular readback, real seal, transport, replay, provenance, and local naming rows; reflector exposure through F before triangle reads through T; triangle coherence through the shared U, K unit-counit boundary; regular-Cauchy handoff through S, D, R, E; transport and replay stability through J, C; and non-escape from quotient completion, selected representatives, countable choice, host equality, and ambient* Real$^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 5982.1. The rows $F, T, U, K, S, D, R, E, J, C, P, N$ are exactly the finite data admitted by the local packet.

The reflector is exposed first by F, and the triangle row T may be read only as the unit-counit-idempotent surface attached to that reflector. The unit row U and counit row K therefore share the same completion-facing boundary.

The regular-Cauchy part is forced through S, D, R, E. Finite StreamName windows, dyadic tolerance data, and RegSeqRat readback are displayed before the terminal Real seal can be consumed.

Finally J, C, P, N transport, replay, source, and name the same route. Since no carrier coordinate stores quotient completion, a selected representative, countable choice, host equality, or ambient real completeness, the listed obligations exhaust the accepted NameCert surface. □

Theorem 5982.3 (CauchyCompletionReflectorTriangleHandoff nonescape)**.** *Every accepted* CauchyCompletionRef
packet exports the reflector-triangle handoff only through

CauchyCompletionReflector$^{\uparrow}$, CompletionReflectorTriangle$^{\uparrow}$, CauchyCompletionUnit$^{\uparrow}$, CauchyCompletionCounit$^{\uparrow}$, Str

It exports no quotient completion, selected representative, countable-choice selector, host equality, or ambient Real$^{\uparrow}$ *completeness row.*

Proof. Use Definition 5982.2. It requires every accepted handoff to expose F, T, U, K, S, D, R, E before the structural rows are consumed.

The reflector and triangle siblings supply only displayed L10 completion rows, and the Real-facing path is confined to finite stream windows, dyadic tolerance, regular readback, and a terminal Real seal. None of those rows is a quotient carrier or a selected representative.

The structural rows J, C, P, N preserve and name the same finite route. The forbidden host-side and choice-side principles have no carrier coordinate, so they cannot be exported by an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}$ packet is generated from reflector, triangle, unit, counit, stream-window, dyadic-tolerance, regular-readback, Real-seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5982.1, Definition 5982.2, and Definition 5982.3 seed the finite completion-reflector triangle handoff surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionReflectorTriangleHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflector exposure, triangle-unit-counit coherence, regular-Cauchy handoff, transport and replay stability, provenance, local naming, and non-escape for BEDC.Derived.CauchyCompletionRe

Chapter 5983

A Concrete Naming Certificate for $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$

Following Bishop and Bridges on separated complete metric spaces, the $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet records the finite route by which two complete-metric limit rows for one displayed Cauchy source are compared through distance zero and then through the separated metric classifier. It sits over the complete-metric service of chapter 128, the metric source of chapter 57, the terminal real-name seal of chapter 38, the regular rational readback surface of chapter 301, and the separated-limit uniqueness boundary of chapter 3365. The packet names limit uniqueness only as displayed finite NameCert data; it does not identify quotient representatives, choose a limit, invoke countable choice, import ambient $\mathsf{Real}^{\uparrow}$ completeness, or export host point equality.

Before the two candidate limits are compared, the uniqueness packet reads a displayed $\mathsf{CompleteMetric}^{\uparrow}$ limit carrier from chapter 128 and the completion route supplied by chapter 1569. The uniqueness packet consumes $\mathsf{CauchyFilterCompletion}^{\uparrow}$ as the source completion route and $\mathsf{MetricCompletion}^{\uparrow}$ as the optional metric-completion face before the separated classifier can export uniqueness. The optional $\mathsf{MetricCompletion}^{\uparrow}$ face is used only as a sibling completion route when the child row already names it; the uniqueness boundary itself still waits for the complete-metric limit rows, Cauchy-filter completion data, distance-zero readback, and separated classifier before exposing a comparison.

Definition 5983.1 (Complete metric limit uniqueness carrier)**.** *A* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, L_0, L_1, M, Z, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the Cauchy-source row, exposing the finite request and tail data whose limit is being compared.*
2. *L_0 and L_1 are complete-metric limit rows for the same displayed source A.*
3. *M is the $\mathsf{Met}^{\uparrow}$ distance-witness row comparing the two displayed limit reads.*
4. *Z is the $\mathsf{Real}^{\uparrow}$ zero-distance row reached from the distance witness through the regular rational readback surface.*
5. *S is the separated-metric classifier row saying that the zero-distance read is the only admissible route from the two limit rows to a uniqueness handoff.*

6. *H records componentwise* hsame *transport for the source, limit rows, distance witness, zero-distance row, separated classifier, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from the Cauchy-source request through the two limit rows, metric distance witness, Real zero-distance row, and separated classifier.*
8. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{SeparatedMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, L_0, L_1, M, Z, S, H, C, P, N$. It has no coordinate for quotient representatives, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host equality of metric points.*

Theorem 5983.2 (Complete metric limit uniqueness NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier*

$$U = (A, L_0, L_1, M, Z, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ *surface consists exactly of Cauchy-source admission for A, paired complete-metric limit admission for L_0 and L_1 over that source, metric distance-witness exactness for M, Real zero-distance readback for Z, separated classifier handoff for S, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No complete-metric limit uniqueness obligation is stored outside these rows.

Proof. Project the accepted carrier. The source row A is the only Cauchy datum available to the packet, and the two complete-metric rows L_0 and L_1 are admitted only as limit reads for that displayed source.

The comparison then passes through the metric witness M and the Real zero-distance row Z. This records a finite distance-zero handoff, not a host equality of endpoints or an equality of quotient representatives.

Finally the separated classifier S consumes that zero-distance row. The support rows H, C, P, N transport, replay, source, and name the same route, so the listed rows exhaust the local NameCert obligations. □

Theorem 5983.3 (Complete metric limit uniqueness carrier rows)**.** *Every accepted* CompleteMetricLimitUniqueness *carrier displays exactly the rows*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

of Definition 5983.1 before the NameCert obligations of Definition 5983.2 can be read. The two limit rows L_0 and L_1 are therefore public only as complete-metric limit reads over the same source row A, and the structural rows H, C, P, N may replay or name only that displayed route.

Proof. Project the carrier definition. It lists the source, paired limit rows, metric witness, zero-distance row, separated classifier, transport, replay, provenance, and local naming as the complete finite packet.

Apply the NameCert obligation theorem. Its obligation surface is indexed by those same coordinates, so no consumer can read a limit comparison before the carrier rows have been displayed.

The replay and naming rows preserve the projected route. They do not add a second source, a selected limit, or an unstored equality principle. □

Theorem 5983.4 (Complete metric limit uniqueness separated nonescape)**.** *Let*

$$U = (A, L_0, L_1, M, Z, S, H, C, P, N)$$

be an accepted $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier. Any public uniqueness comparison must pass through the metric zero-distance row* Z *and then through the separated classifier row* S *before it is exposed. In particular, uniqueness cannot escape from* L_0 *and* L_1 *directly to host point equality, quotient representative equality, selected limits, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier-row theorem Definition 5983.3. It forces the comparison to begin with the shared source A and the two complete-metric limit rows L_0, L_1.

Use the obligation theorem Definition 5983.2. It places the metric witness M and zero-distance row Z before the separated handoff, so the comparison cannot bypass the distance-zero route.

Finally read the separated classifier S. Since H, C, P, N only transport, replay, source, and name the same finite packet, no host equality, quotient representative, selected limit, choice schedule, or ambient completeness principle is available. □

Theorem 5983.5 (Complete metric limit uniqueness NameCert scope)**.** *The local* $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniquene}}$ *scope for an accepted packet is exactly the carrier-row surface*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

together with the NameCert obligations of Definition 5983.2. A public comparison may be exposed only after the metric zero-distance row and separated classifier have both been displayed, so every downstream uniqueness consumer reads a separated nonescape route rather than an ambient equality theorem.

Proof. Combine Definition 5983.3 with the obligation theorem. The first theorem fixes the carrier coordinates, and the second assigns the local NameCert obligations to those coordinates.

Apply separated nonescape, Definition 5983.4. It identifies Z and S as the required boundary between the paired limit rows and any public uniqueness read.

The resulting scope is therefore finite and local to the displayed packet. It cannot be enlarged by replay into quotient equality, selected endpoints, countable choice, host equality, or ambient completeness. □

Theorem 5983.6 (Complete metric limit uniqueness separatedness handoff)**.** *Every consumer that reads uniqueness from an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier must factor the read through*

$$A, L_0, L_1, M, Z, S, H, C, P, N.$$

The separatedness step is visible only after both complete-metric limit rows have been tied to the same Cauchy source, the $\mathsf{Met}^{\uparrow}$ *distance witness has been displayed, and the* $\mathsf{Real}^{\uparrow}$ *zero-distance row has been sealed. The handoff exports no quotient representative equality, selected limit, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or host equality of metric points.*

Proof. Start from an accepted carrier. The consumer first reads A, L_0, L_1, so the two candidate limits are compared only as complete-metric limit rows for the same displayed Cauchy source.

The metric row M then supplies the displayed distance comparison, and Z seals the zero-distance read through the Real-facing route. At this point the carrier still has no coordinate for selected representatives or host point equality.

The separated classifier S is the only row that turns the zero-distance evidence into a uniqueness handoff. Since H, C, P, N merely preserve the same finite path, every accepted uniqueness read factors through the displayed rows and cannot escape into the excluded ambient principles. □

Theorem 5983.7 (Complete metric limit uniqueness zero-distance reflection)**.** *Any two accepted limit rows over the same Cauchy source can export a complete-metric uniqueness read only after the* $\mathsf{Met}^{\uparrow}$ *distance witness row and the* $\mathsf{Real}^{\uparrow}$ *zero-distance row have both been displayed and reflected through the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier of chapter 1790. Equivalently, every accepted zero-distance reflection route for*

$$U = (A, L_0, L_1, M, Z, S, H, C, P, N)$$

factors through A, L_0, L_1, M, Z, S *before* H, C, P, N *may replay or name the handoff. The route exports no selected limit, quotient representative equality, countable-choice schedule, ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle, or host equality of metric points.*

Proof. Use the carrier of Definition 5983.1. It places the two candidate limit rows L_0 and L_1 over the same displayed Cauchy-source row A, so no comparison can start from independent sources or selected representatives.

Apply the NameCert obligations of Definition 5983.2. They require the metric comparison row M and the Real zero-distance row Z to be displayed before any separated classifier handoff is available.

Finally apply the separatedness handoff of Definition 5983.6. The classifier S is the only row that reflects zero distance into uniqueness, and the structural rows H, C, P, N preserve that same finite route without adding quotient equality, choice, ambient completeness, or host point equality. □

Theorem 5983.8 (Complete metric limit uniqueness common window)**.** *If two limit-seal reads are accepted by the* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier* $U = (A, L_0, L_1, M, Z, S, H, C, P, N)$*, then the separatedness handoff and zero-distance reflection inspect the same metric row* M*, Cauchy route* A, L_0, L_1*,* $\mathsf{Real}^{\uparrow}$ *zero-distance seal* Z*,* hsame *transport* H*,* Cont *replay* C*,* Pkg *provenance* P*, and local* $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ *row* N*. The uniqueness read therefore has one finite witness route and exports no separate equality principle or selected limit.*

Proof. Project the carrier of Definition 5983.1. The two limit rows L_0 and L_1 are tied to the same Cauchy-source row A, and the only displayed metric comparison is the row M.

Apply the separatedness handoff of Definition 5983.6. It binds every admissible uniqueness read to the displayed route through $A, L_0, L_1, M, Z, S, H, C, P, N$.

Now apply zero-distance reflection, Definition 5983.7, on that same route. The Real zero-distance seal Z, separated classifier S, transport H, replay C, provenance P, and local naming N remain attached to the displayed packet, so no separate equality principle or selected limit is exported. □

Theorem 5983.9 (Common tail contraction for uniqueness)**.** *If two accepted complete-metric limit rows read the same Cauchy source through the common window of Definition 5983.8, then any uniqueness consumer must contract both reads to the same tail before applying zero-distance reflection and separatedness. The contracted route factors through the common window, the zero-distance row, and the separated classifier of Definition 5983.7 and Definition 5983.6. It does not appeal to selected limits, quotient equality, or ambient completeness.*

Proof. Start with the common window theorem Definition 5983.8. It shows that the two complete-metric limit rows share the same metric row, Cauchy route, zero-distance seal, transport, replay, provenance, and naming rows.

Contract both reads to that shared tail. The contraction step uses only the displayed common window, so it does not introduce a selected limit or any quotient representative.

Now apply zero-distance reflection, Definition 5983.7. The reflected route still passes through the same common tail, and the separated classifier remains the only row that can turn the zero-distance evidence into a uniqueness handoff.

Finally apply separatedness, Definition 5983.6. The uniqueness consumer therefore sees one contracted tail before the reflected comparison is exported, and the route stays inside the finite packet rather than invoking ambient completeness. □

Theorem 5983.10 (Common tail classifier for uniqueness)**.** *If two accepted complete-metric limit rows share a displayed Cauchy source and their common tail contraction supplies the Real zero-distance row, then every public uniqueness read must pass through the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier before any equality-like comparison is exported.*

Proof. Apply the common tail contraction theorem Definition 5983.9. It reduces the uniqueness read to the shared tail, the zero-distance row, and the separated classifier route already visible in the packet.

Next apply zero-distance reflection, Definition 5983.7. The reflected route keeps the zero-distance row attached to the same finite source, so no equality-like comparison can be exported from an unclassified read.

Finally apply separatedness, Definition 5983.6. The public read must therefore pass through $\mathsf{SeparatedMetricReflection}^{\uparrow}$ before any equality-like comparison is exposed. □

Theorem 5983.11 (Complete metric limit uniqueness separated zero-distance route)**.** *Every public uniqueness read from an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier must expose the complete-metric limit rows* L_0, L_1*, the* $\mathsf{Met}^{\uparrow}$ *distance witness* M*, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row* Z*, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier* S *before uniqueness is available. The route remains the finite packet*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

and exports no quotient limit, selected endpoint, countable-choice schedule, ambient completeness principle, or host equality of metric points.

Proof. Start with zero-distance reflection, Definition 5983.7. It places the metric witness M and Real zero-distance row Z downstream of the paired complete-metric limit rows L_0, L_1 over the displayed source A.

Apply the separated-reflection route Definition 5983.19. The separated classifier S is read only after the distance witness and zero-distance row are visible.

Finally compose with the sibling lattice route Definition 5983.20. The structural rows H, C, P, N preserve the same route, so public uniqueness cannot be read through quotient limits, selected endpoints, countable choice, ambient completeness, or host equality. □

Theorem 5983.12 (Complete metric limit uniqueness through separated classifier)**.** *Let* $U = (A, L_0, L_1, M, Z, S, H, C, P, N)$ *be an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier. Two accepted limit rows over the same Cauchy source become classifier-equal only after the common-window row, zero-distance reflection,* $\mathsf{Real}^{\uparrow}$ *zero-distance readback, and* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier row are all exposed. The public uniqueness route is therefore the displayed packet*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

and not a quotient representative equality, selected endpoint, countable-choice schedule, ambient completeness principle, or host point equality.

Proof. Start with the common tail contraction theorem, Definition 5983.9. It contracts the two accepted complete-metric limit rows to the same displayed tail over the Cauchy source row A before any uniqueness comparison is exported.

Next apply the common tail classifier theorem, Definition 5983.10. The public read must pass through the separated-reflection classifier after the Real zero-distance row has stayed attached to the same finite source.

Finally use the separated zero-distance route of Definition 5983.11. It displays M, Z, S together with L_0, L_1 and the structural rows H, C, P, N, so classifier equality is exposed only through the finite separated route and not through quotient equality, selected limits, countable choice, ambient completeness, or host equality. □

Theorem 5983.13 (Complete metric limit uniqueness separated seal)**.** *Let* $U = (A, L_0, L_1, M, Z, S, H, C, P, N)$ *be an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier. Two accepted limit rows over the same Cauchy source are publicly unique only through the route that exposes the complete-metric limit rows* L_0, L_1*, the* $\mathsf{Met}^{\uparrow}$ *distance witness* M*, the* $\mathsf{RegSeqRat}^{\uparrow}$*-backed* $\mathsf{Real}^{\uparrow}$ *zero-distance row* Z*, and the* $\mathsf{SeparatedMetric}^{\uparrow}$ *classifier* S *before any uniqueness handoff is public. The separated seal is the displayed packet*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

and does not assert host equality, selected limits, quotient representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with zero-distance reflection, Definition 5983.7. It forces any uniqueness read to expose the metric distance witness M and the Real zero-distance row Z downstream of the paired complete-metric limit rows over the source A.

Apply the separated-reflection route and sibling route, Definition 5983.19 and Definition 5983.20. These keep the separated classifier S attached to A, L_0, L_1, M, Z and preserve the same structural rows H, C, P, N.

Finally use the separated classifier theorem, Definition 5983.12. The public uniqueness handoff is available only after the classifier has consumed the zero-distance row on the displayed finite route, so the seal has no coordinate for host equality, selected limits, quotient representatives, countable choice, or ambient Real completeness. □

Theorem 5983.14 (Complete metric limit uniqueness kernel scope)**.** *Any accepted* $\mathsf{CompleteMetricLimitUniqueness}$ *uniqueness read must factor through separated metric identity, Cauchy witness rows,* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{StreamName}^{\uparrow}$ *readback, a terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ *before uniqueness is public. In carrier notation the kernel-scope route is*

$$A, L_0, L_1, M, Z, S, H, C, P, N.$$

The route exports no ambient $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient equality, countable choice, or host point equality.*

Proof. Project the carrier rows of Definition 5983.1. The Cauchy witness is the source row A together with the paired complete-metric limit rows L_0, L_1, while the metric comparison and zero-distance read are M and Z.

Use the separated reflection route already cited in the chapter. The separated classifier S is read only after the distance witness and zero-distance row are displayed, so uniqueness is a separated metric identity route rather than host equality.

Use the regular Cauchy Real route inputs through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ for the zero-distance seal. The structural rows H, C, P, N transport, replay, source, and locally name that same finite route.

The ledger refusal is immediate from the carrier surface. There is no coordinate for ambient Real completeness, selected limits, quotient equality, countable choice, or host point equality, so the kernel-scope route cannot export any of those principles. □

Theorem 5983.15 (Complete metric limit uniqueness Real completion witness route)**.** *Let $U = (A, L_0, L_1, M, Z, S, H, C, P, N)$ be an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier whose zero-distance row is supplied through the* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *route of chapter 652. Then the uniqueness read must first factor through the extractor coordinates that display the diagonal request, selected windows, regular readback, Real seal, completion-facing handoff, transport, replay, provenance, and local naming rows. Only after those extractor coordinates expose the zero-distance row may the packet apply the separated classifier S to export uniqueness. The route exports no selected limit, quotient representative equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or host point equality.*

Proof. Start with the kernel-scope route of Definition 5983.14. It forces every uniqueness read to pass through $A, L_0, L_1, M, Z, S, H, C, P, N$ and keeps the Real zero-distance row attached to regular readback, stream windows, transport, replay, provenance, and local naming.

Now read the zero-distance row through chapter 652. The extractor surface exposes diagonal, selected-window, regular-readback, Real-seal, completion-handoff, transport, replay, provenance, and naming coordinates before a completion-facing zero-distance witness can be consumed.

Finally apply the separated classifier only after those extractor coordinates have supplied Z. Since the classifier receives a displayed extractor-backed zero-distance row rather than a host completion object, the uniqueness route cannot export selected limits, quotient representative equality, countable choice, ambient Real completeness, or host point equality. □

Theorem 5983.16 (Complete metric limit uniqueness Cauchy-filter comparison)**.** *Let $U = (A, L_0, L_1, M, Z, S, H, C$ be an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier whose two limit rows are read from the same* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source row. Every uniqueness export must pass through the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source row A, the* $\mathsf{Met}^{\uparrow}$ *distance witness M, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier S before any equality-like comparison is public. The comparison therefore remains the finite route*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

and does not export quotient-filter equality, selected limits, countable choice, ambient completeness, or host equality of metric points.

Proof. Start with the common tail contraction theorem, Definition 5983.9. It contracts the two complete-metric limit rows to one displayed tail over the shared Cauchy source row before any comparison is read.

Apply zero-distance reflection, Definition 5983.7. The reflected route exposes the $\mathsf{Met}^{\uparrow}$ distance witness M and the $\mathsf{Real}^{\uparrow}$ zero-distance row Z downstream of the same source row and the paired limit rows.

Now use the separated zero-distance route, Definition 5983.11. The $\mathsf{SeparatedMetricReflection}^{\uparrow}$ classifier S is the boundary at which the zero-distance read becomes a public uniqueness handoff.

Finally apply the sibling lattice route already recorded by Definition 5983.20. The source row is the Cauchy-filter completion route, while H, C, P, N transport, replay, package, and locally name

the same finite comparison. No coordinate remains for quotient-filter equality, selected limits, countable choice, ambient completeness, or host equality. □

Theorem 5983.17 (Complete metric limit uniqueness filter-tail selector route)**.** *Let* $U = (A, L_0, L_1, M, Z, S, H, C,$ *be an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier whose uniqueness read comes from a* $\mathsf{CauchyFilterCompletion}$ *source. Before zero-distance reflection, regular rational readback, stream-window inspection, dyadic tolerance comparison, Real sealing, or separated classification is read, the packet must display one tail selector row inside* A *binding the two* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows* L_0 *and* L_1 *to the same filter tail. After that selector is exposed, the admissible route is the single finite path through* $A, L_0, L_1, M, Z, S, H, C, P, N$*, with the* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*, and* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *sibling reads attached to that same tail.*

Proof. Start with common tail contraction, Definition 5983.9. It requires the two complete-metric limit reads to be contracted to one displayed tail before a uniqueness comparison is exported.

Now apply the Cauchy-filter comparison theorem, Definition 5983.16. Since the source is a $\mathsf{CauchyFilterCompletion}^{\uparrow}$ row, the common tail must be presented as a selector row inside A before the two limit rows can be compared as reads from the same filter completion route.

With the selector displayed, use zero-distance reflection and the separated zero-distance route, Definition 5983.7 and Definition 5983.11. These force M and Z to remain downstream of the same tail selector before the $\mathsf{SeparatedMetricReflection}^{\uparrow}$ classifier S is read.

Finally compose with the sibling lattice route, Definition 5983.20. The regular rational readback, StreamName windows, dyadic tolerance comparison, Real seal, transport, replay, provenance, and local naming rows all preserve the selected tail route, so no quotient-filter equality, selected limit, countable-choice schedule, ambient completeness principle, or host point equality is exported. □

Theorem 5983.18 (Complete metric limit uniqueness forward route)**.** *Every* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *read that exports uniqueness must consume the separated reflection route of chapter 1790, the regular Cauchy real reflection of chapter 6280, and the displayed metric completion boundary before public uniqueness is available. The forward route remains the finite packet* $A, L_0, L_1, M, Z, S, H, C, P, N$ *and does not export quotient endpoints, selected limits, countable choice, ambient completeness, or host equality of metric points.*

Proof. Start with zero-distance reflection and the separated-reflection route, Definition 5983.7 and Definition 5983.11. They expose M, Z, S after the paired complete-metric limit rows over the source A.

Next read the regular Cauchy real sibling as the source of the Real zero-distance seal. The regular reflection keeps the comparison inside displayed stream, regular readback, and Real-facing rows rather than a quotient or host equality.

Finally apply the Cauchy-filter comparison route of Definition 5983.16. It binds the uniqueness export to the metric completion boundary and the structural rows H, C, P, N, so no refused endpoint is public. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet is generated from one Cauchy-source row, two CompleteMetricUp limit rows, a MetricUp distance witness, a RealUp zero-distance row, a SeparatedMetricUp classifier handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5983.1, Definition 5983.2, and Definition 5983.6 seed the finite complete-metric limit uniqueness packet over one Cauchy-source row, two complete-metric limit rows, a MetricUp distance witness, a RealUp zero-distance row, a separated classifier, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{CompleteMetricLimitUniqueness}^{\uparrow}}$, unformalized)
Bridge status: none
Not claimed: This chapter does not claim quotient representatives, selected limits, countable choice, ambient RealUp completeness, host point equality, Lean verification, or bridge checking.
Upgrade path: Obligation closure requires checked Cauchy-source admission, paired complete-metric limit admission, distance-witness exactness, zero-distance sealing, separated classifier handoff, transport stability, replay stability, and non-escape for BEDC.Derived.CompleteMetricLimitUniquenessUp.

Theorem 5983.19 (Complete metric limit uniqueness separated reflection route)**.** *Every* CompleteMetricLimitUniqu *read that exports uniqueness for two displayed complete-metric limit rows must consume the separated-reflection classifier route of chapter 1790 after the* Met$^{\uparrow}$ *distance witness and the* Real$^{\uparrow}$ *zero-distance row have been displayed. The route therefore passes through the packet rows*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

before any uniqueness handoff is exported. It does not identify quotient representatives, choose a limit, invoke ambient completeness, or export host equality of metric points.

Proof. Project the carrier of Definition 5983.1. The two limit rows L_0 and L_1 are tied to the same Cauchy-source row A before any comparison is available.

Use the metric distance witness M and the Real zero-distance row Z. These rows display the finite zero-distance comparison, but they do not by themselves export uniqueness or host point equality.

Now route the comparison through the separated-reflection classifier of chapter 1790. The separated row S is therefore consumed only after M and Z are visible, and before the uniqueness handoff is read by any downstream consumer.

Finally carry the route through H, C, P, N. These rows transport, replay, source, and name the same finite packet, so the handoff remains a separated-reflection dependency rather than a quotient, selected-limit, ambient-completeness, or host-equality principle. □

Theorem 5983.20 (Complete metric limit uniqueness sibling route)**.** *Every* CompleteMetricLimitUniqueness$^{\uparrow}$ *read that exports a public uniqueness handoff must expose the sibling lattice route consisting of the* CompleteMetric$^{\uparrow}$ *limit rows from chapter 128, the* CauchyFilterCompletion$^{\uparrow}$ *source route of chapter 1569, the* Met$^{\uparrow}$ *distance witness of chapter 57, the* Real$^{\uparrow}$ *zero-distance row of chapter 38, and the* SeparatedMetricReflection$^{\uparrow}$ *classifier of chapter 1790 before the handoff is public. In carrier notation this route factors through*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

with A carrying the Cauchy-filter completion source data, L_0, L_1 carrying the complete-metric limit rows, M carrying the metric comparison, Z carrying the Real zero-distance read, and S carrying the separated-reflection classifier. The route exports no selected limit, quotient representative equality, countable-choice schedule, ambient completeness principle, or host point equality.

Proof. Project the carrier obligations of Definition 5983.2. They expose the Cauchy-source row A, the paired complete-metric limit rows L_0, L_1, the metric distance witness M, the Real zero-distance row Z, and the separated classifier S before any local naming row can publish uniqueness.

Apply Definition 5983.19. The separated-reflection classifier is consumed only after the metric witness and zero-distance read are visible, so the public handoff must pass through the same sibling route.

The support rows H, C, P, N preserve and name this displayed route. They add transport, replay, provenance, and local NameCert data, but they do not add selected limits, quotient representative equality, countable choice, ambient completeness, or host point equality. □

CompleteMetricLimitUniquenessUp reads MetricCompletionUp only through the displayed CompleteMetricUp limit rows and the CauchyFilterCompletionUp source route before the zero-distance and separated-classifier handoff. The dependency is therefore a finite forward route through chapter 128 and chapter 1569; it does not collapse into host equality or an ambient completeness claim. Every $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ dossier read that is consumed by completion-lattice readers must first expose the downstream route through the complete-metric service of chapter 128, the Cauchy-filter completion service of chapter 1569, and the separated metric reflection service of chapter 1790. Only after those three rows have been displayed may a public uniqueness read leave the packet; the route therefore records a finite completion-to-separatedness dependency rather than an ambient equality principle.

Theorem 5983.21 (CompleteMetricLimitUniqueness downstream separated-completion link)**.** *Every downstream separated-completion or metric-completion uniqueness consumer that cites* CompleteMetricLimitU *must cite it through the displayed route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{SeparatedMetricReflection}^{\uparrow}.$$

The consumer first reads the complete-metric limit rows supplied by $\mathsf{CompleteMetric}^{\uparrow}$*, then the* $\mathsf{Met}^{\uparrow}$ *distance witness, then the* $\mathsf{Real}^{\uparrow}$ *zero-distance row, and only then the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier row. The route exports no quotient-limit equality, selected endpoint, countable-choice schedule, ambient completeness principle, or host equality of metric points.*

Proof. Start with the carrier of Definition 5983.1. Its limit rows are accepted only as complete-metric limit rows over the same displayed Cauchy source.

Apply the zero-distance reflection route of Definition 5983.7. It places the $\mathsf{Met}^{\uparrow}$ distance witness and the $\mathsf{Real}^{\uparrow}$ zero-distance read before any uniqueness handoff is available.

Finally apply Definition 5983.12 and the separated-reflection route of Definition 5983.19. They force the downstream uniqueness consumer to read the $\mathsf{SeparatedMetricReflection}^{\uparrow}$ classifier after the zero-distance row, so a separated-completion or metric-completion consumer can cite only the displayed finite route. □

Theorem 5983.22 (Complete metric limit uniqueness completion sibling route)**.** *Every* CompleteMetricLimitUnique *read that exports uniqueness must first consume the separated limit comparison supplied by chapter 128 and the completion-boundary row supplied by chapter 1506. The exported route therefore remains inside the displayed packet rows*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

with L_0, L_1 *read as complete-metric limit rows over one Cauchy source,* M, Z, S *read as the zero-distance and separated-classifier comparison, and* H, C, P, N *preserving the same finite path. It exports no quotient completion point, selected representative, or host equality of metric points.*

Proof. Project the carrier obligations of Definition 5983.2. They expose the complete-metric limit rows L_0, L_1 only over the shared source row A, so the separated comparison cannot begin from detached limit data.

Read the sibling completion boundary through chapter 1506. This supplies the completion-facing row consumed before M, Z, S compare the two displayed limit reads, but it does not add quotient completion points or selected representatives.

Finally apply the separatedness handoff of Definition 5983.6. The support rows H, C, P, N replay and name the same finite route, so the uniqueness export remains a completion-sibling dependency rather than host equality. □

Theorem 5983.23 (CompleteMetricLimitUniqueness regular-Cauchy real route)**.** *Every uniqueness read consumed by a* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *packet factors through the displayed* $\mathsf{CompleteMetric}^{\uparrow}$ *separatedness rows of chapter 128, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *tolerance-exactness route of chapter 3390, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows of chapter 126, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows of chapter 291, the* $\mathsf{Real}^{\uparrow}$ *sealing route of chapter 38, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ *row. Thus any regular-Cauchy consumer reaches uniqueness only through*

$$\mathsf{CompleteMetric}^{\uparrow}, \mathsf{RegularCauchyUniqueness}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Na}$$

before it may read the result.

Proof. Start with the carrier of Definition 5983.1. Its two limit rows, metric-distance witness, zero-distance row, separated classifier, transport, replay, provenance, and local name are the only rows visible to a uniqueness consumer.

Apply the common-tail and zero-distance routes, Definition 5983.9 and Definition 5983.7. They force the comparison through the same regular-Cauchy readback and the same Real zero-distance seal before separatedness is inspected.

Now read the finite-window siblings. The $\mathsf{RegSeqRat}^{\uparrow}$ source, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows provide the regular-Cauchy exactness surface, while the $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ sibling supplies the tolerance uniqueness boundary used by regular consumers.

Finally use separatedness and the sibling lattice route, Definition 5983.6 and Definition 5983.20. The structural rows preserve that same finite route through hsame, Cont, Pkg, and local naming, so the consumer cannot read quotient stream equality, a selected limit, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or host equality of metric points. □

Theorem 5983.24 (Complete metric limit uniqueness regular-Cauchy forward link)**.** *Every consumer that reads a separated* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *result after a regular-Cauchy completion seal first exposes the regular rational readback rows of chapter 301, the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38, and the metric-completion boundary of chapter 1506. Only after those rows are visible may it consume the complete-metric separatedness route of chapter 128 and the local uniqueness row of* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$*.*

Thus the forward dependency lattice is

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{MetricCompletion}^{\uparrow}, \mathsf{CompleteMetric}^{\uparrow} \longrightarrow \mathsf{CompleteMetricLimitUniqueness}^{\uparrow},$$

with the arrow mediated by displayed hsame*,* Cont*,* Pkg*, and local* $\mathsf{NameCert}$ *rows. The link exports no selected limit, quotient Cauchy stream equality, countable-choice schedule, host completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from a regular-Cauchy completion presentation. The regular rational readback rows must be public before the Real seal is available, so the consumer first reads chapter 301 and chapter 38.

Next pass through the metric-completion boundary. The completion row supplies the displayed completion-facing carrier, while the complete-metric sibling supplies the separatedness surface used by a uniqueness read.

Finally consume the local $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ row through the existing separated-route and sibling-lattice theorems in this chapter. Since every source row has already been displayed through hsame, Cont, Pkg, and local naming, the forward link cannot add a selected limit, quotient stream equality, countable choice, a host completion object, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 5983.25 (Complete metric limit uniqueness separated route)**.** *When a* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *packet reads uniqueness through the regular Cauchy Real route of Definition 5983.23, separatedness and zero-distance reflection force a unique classifier-compatible terminal seal. The consumer must pass through the shared Cauchy source, the two complete-metric limit rows, the* $\mathsf{Met}^{\uparrow}$ *distance witness, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row, the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier, the* $\mathsf{RegularCauchyUniqueness}^{\uparrow}$ *tolerance route, the* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows, and the structural* hsame*,* Cont*,* Pkg*, and local* NameCert *rows before uniqueness is exported.*

The route adds no host equality of metric points, selected representatives, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or separate completeness principle.*

Proof. Start with the regular-Cauchy Real route Definition 5983.23. It forces every regular-Cauchy consumer through $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegularCauchyUniqueness}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, and the structural rows before a uniqueness read is available.

Now apply zero-distance reflection and separatedness, Definition 5983.7 and Definition 5983.6. These routes bind the two complete-metric limit rows to one Cauchy source, then require the metric distance witness and Real zero-distance row before the separated classifier can export uniqueness.

Finally use the separated reflection route Definition 5983.19. It places the separated classifier after the displayed distance and zero-distance rows and before the downstream handoff. Hence the terminal seal is classifier-compatible and unique inside the packet, while host equality, selected representatives, quotient equality, selected limits, choice, ambient Real completeness, and separate completeness data remain off-surface. □

Theorem 5983.26 (Complete metric limit uniqueness Cauchy-filter Real seal)**.** *Every uniqueness read from a* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source must expose the Cauchy-filter completion row of chapter 1569, the paired* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, the* $\mathsf{Met}^{\uparrow}$ *distance witness, the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *zero-distance seal, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier before any uniqueness handoff is public. The route remains the displayed packet*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

and exports no quotient-filter equality, selected limit, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host equality of metric points.*

Proof. Start with the Cauchy-filter comparison route of Definition 5983.16. It ties the two complete-metric limit rows to the same $\mathsf{CauchyFilterCompletion}^{\uparrow}$ source row before any equality-like comparison is exposed.

Apply the common-tail contraction and zero-distance reflection routes, Definition 5983.9 and Definition 5983.7. They force the paired limit rows through one metric witness M and one $\mathsf{RegSeqRat}^{\uparrow}$-backed $\mathsf{Real}^{\uparrow}$ zero-distance row Z.

Finally use separatedness and the separated zero-distance route, Definition 5983.6 and Definition 5983.11. The separated classifier is visible before the uniqueness handoff, while H, C, P, N replay and name the same Cauchy-filter comparison route. □

Theorem 5983.27 (Complete metric limit uniqueness CauchyLimitSeal route)**.** *Every separated complete-metric limit-uniqueness read factors through the displayed* $\mathsf{CauchyLimitSeal}^{\uparrow}$ *boundary before public uniqueness is available. More precisely, an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *consumer must expose the finite route*

$$\mathsf{CauchyLimitSeal}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{MetricCompletion}^{\uparrow}$$

together with the carrier rows $A, L_0, L_1, M, Z, S, H, C, P, N$ before the separated classifier may publish uniqueness. The route exports no quotient stream equality, host selected limit, ambient completeness principle, or detached completion witness.

Proof. Start with the carrier and sibling lattice routes of Definition 5983.1 and Definition 5983.20. They place the two complete-metric limit rows over the same displayed Cauchy source and keep the metric-completion face inside the same finite packet.

Next attach the regular readback rows. The regular route of Definition 5983.23 and the regular-sequence forward route of Definition 5983.29 force every zero-distance read through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal $\mathsf{Real}^{\uparrow}$ seal before separatedness is inspected.

Now pass through the Cauchy-limit seal. The Cauchy-filter Real seal route of Definition 5983.26 and the completion sibling route of Definition 5983.22 keep the same source, readback, Real seal, and completion face attached to the packet before the separated classifier is visible.

Finally apply the separated reflection route of Definition 5983.19. Since S is read only after M and Z, and Z is reached through the displayed window, readback, Real, CauchyLimitSeal, and MetricCompletion rows, the public uniqueness export cannot introduce quotient stream equality, a host selected limit, ambient completeness, or a second completion witness. □

Theorem 5983.28 (Complete metric limit uniqueness completion-witness route)**.** *Every accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *comparison that is consumed by a completion witness must pass through the displayed source row A, the paired complete-metric limit rows L_0 and L_1, the* $\mathsf{Met}^{\uparrow}$ *distance row M, the* $\mathsf{Real}^{\uparrow}$ *zero-distance row Z, the separated classifier row S, and the structural rows H, C, P, N before it reaches the* $\mathsf{RealCompletionWitnessExtractor}^{\uparrow}$ *route of chapter 652. Thus the witness-facing comparison is the finite route*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

followed by the extractor's displayed diagonal, selected-window, regular readback, Real seal, completion handoff, transport, replay, provenance, and local naming rows.

Proof. Start with the carrier rows of Definition 5983.1. They make the source row, paired limit rows, metric distance row, Real zero-distance row, separated classifier, transport, replay, provenance, and local naming visible before any uniqueness comparison can be read.

Apply the separatedness handoff of Definition 5983.6 and the separated nonescape theorem Definition 5983.4. These force the comparison through M, Z, S after the paired limit rows and before any downstream consumer receives a uniqueness read.

Finally route the accepted comparison into chapter 652. The extractor consumes only its displayed finite rows, so the forwarded comparison adds no selected limit, quotient representative, ambient completeness row, or host equality principle. □

Theorem 5983.29 (Complete metric limit uniqueness regular-sequence forward route)**.** *Every* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *read consumed by a regular-sequence completion route must pass through the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *source surface of chapter 5965, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit rows, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier before uniqueness is public. In carrier notation the route is still the displayed packet*

$$A, L_0, L_1, M, Z, S, H, C, P, N,$$

with the regular-sequence data read before Z, the complete-metric rows read before the comparison, and the separated classifier read before the uniqueness handoff.

Proof. Start with the regular-Cauchy real route of Definition 5983.23. It already forces a regular consumer through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before the zero-distance row can be consumed.

Next insert the $\mathsf{RegularSequenceSpace}^{\uparrow}$ source row. The regular-sequence-space carrier supplies the common family and window surface before the same $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows are visible to the uniqueness packet.

Now apply the sibling route and separated zero-distance route, Definition 5983.20 and Definition 5983.11. These rows place the $\mathsf{CompleteMetric}^{\uparrow}$ limit reads before the metric comparison and require the $\mathsf{SeparatedMetricReflection}^{\uparrow}$ classifier after the $\mathsf{Real}^{\uparrow}$ zero-distance seal.

Finally carry the route through H, C, P, N. The structural rows transport, replay, source, and name the same finite packet, so regular-sequence completion consumers reach public uniqueness only through the displayed sequence, readback, tolerance, Real seal, complete-metric, and separated-reflection route. □

Theorem 5983.30 (Complete metric limit uniqueness RegSeqRat forward route)**.** *Any* CompleteMetricLimitUnique *read through the separated classifier must first preserve the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{StreamName}^{\uparrow}$ *window rows, the dyadic tolerance ledger, and the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal already exposed by the child routes. Thus the public uniqueness read cannot consume the separated classifier before the regular-rational, stream-window, dyadic, and Real-seal route has been displayed.*

Proof. Apply the common-tail and zero-distance theorems Definition 5983.9 and Definition 5983.7. They force the uniqueness comparison to pass through a shared tail read and then through the Real zero-distance seal.

Use the separated classifier route of Definition 5983.12. The separated read is public only after the carrier has exposed the limit comparison route, so it cannot bypass the regular-rational and stream-window data that feed the zero-distance test.

Finally apply the sibling route theorem Definition 5983.20. The sibling lattice records the RegSeqRat, StreamName, dyadic tolerance, and Real-facing rows as outgoing dependencies of the same packet. Therefore every accepted uniqueness read preserves those child routes before it reaches the separated classifier. □

The $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet consumes the separated-complete metric comparison route of chapter 128 together with the regular-sequence space of chapter 5965, the regular-rational readback surface of chapter 301, and the finite stream windows of chapter 126. This dependency keeps the uniqueness comparison horizontal in the dossier lattice: before the $\mathsf{Real}^{\uparrow}$ zero-distance seal and separated-reflection uniqueness can be consumed, the chapter reads CompleteMetric separation, regular-sequence readback, regular rational tails, stream windows, transport, replay, provenance, and local naming as displayed siblings rather than as a kernel-only float.

Theorem 5983.31 (Complete metric limit uniqueness regular-sequence-space route)**.** *Any* CompleteMetricLimitUni *read of equality of limits must expose the* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *schedule and readback route of chapter 5965, the* $\mathsf{RegSeqRat}^{\uparrow}$ *row, and the terminal* $\mathsf{Real}^{\uparrow}$ *zero-distance seal before separatedness is consumed. In carrier notation the public route remains*

$$A, L_0, L_1, M, Z, S, H, C, P, N,$$

with the regular-sequence-space schedule feeding the regular rational readback, the readback feeding the Real seal Z, and the separated classifier S read only after that seal is visible. The route exports no quotient equality of limits, selected endpoint, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or host equality of metric points.*

Proof. Start with the carrier of Definition 5983.1. The two limit rows L_0 and L_1 are admitted only over the displayed source row A, and the comparison is unavailable until the metric witness M and zero-distance row Z have been exposed.

Next apply the regular-sequence forward route of Definition 5983.29. It forces a regular-sequence completion consumer through $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before public uniqueness can be read.

Finally use the separated reflection route of Definition 5983.19. The separated classifier S is consumed only after the Real zero-distance seal Z, while H, C, P, N transport, replay, source, and name the same displayed packet. Therefore equality of limits is read through the regular-sequence-space schedule, regular rational readback, Real seal, and separatedness route, with no off-surface quotient, endpoint selector, choice schedule, ambient completeness, or host point equality. □

The $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ child route consumes the displayed $\mathsf{CompleteMetric}^{\uparrow}$ rows of chapter 128, the $\mathsf{MetricCompletion}^{\uparrow}$ completion boundary, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-Cauchy readback, the $\mathsf{StreamName}^{\uparrow}$ finite observation windows, and the terminal $\mathsf{Real}^{\uparrow}$ seal before a uniqueness read is exported. The route records this dependency lattice as finite sibling data: complete-metric limits are compared only after the completion row, regular-rational tolerance rows, stream windows, and Real zero-distance seal have been displayed, and the export remains a NameCert uniqueness read rather than a selected limit, quotient representative equality, countable-choice schedule, or host completeness principle. The $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet consumes separatedness through the separated metric certificate of chapter 1411 and reads completion-source data through chapter 1506. This sibling route keeps the two candidate limit rows over the same completion source before the separated classifier can turn the displayed zero-distance row into a uniqueness handoff; it exports no host equality of metric points, quotient representative, selected limit, or ambient completeness principle. The $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet consumes the separated metric reflection of chapter 1790 and the completion uniqueness route of chapter 1796 before asserting uniqueness for a displayed limit route. This child dependency keeps the two candidate limit rows over the same completion witness until separatedness turns the zero-distance row into a uniqueness handoff; it exports no quotient representative, selected limit, ambient completeness principle, or host equality of metric points.

Theorem 5983.32 (Complete metric limit uniqueness separated regular-sequence lattice route)**.** *Every* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *uniqueness read that consumes the separated identity route also consumes the regular-sequence Real route. The route cites the separated metric certificate of chapter 1411, the regular-sequence schedule of chapter 5965, and the Real seal of chapter 38 before the two candidate limit rows can be identified.*

Proof. Start with the displayed carrier of Definition 5983.1. The candidate limit rows are read over the same completion source before any separatedness conclusion is available.

Use the regular-sequence route through Definition 5983.31. It exposes the regular-sequence schedule and the Real zero-distance seal as visible rows.

Only then apply the separated metric dependency route. The separated classifier turns the displayed zero-distance seal into uniqueness, while the route exports no quotient representative, selected endpoint, ambient completeness principle, or host equality of points. □

Theorem 5983.33 (Separated completion limit route)**.** *Every* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *uniqueness read consumes the separated-completion boundary of chapter 1612 or the metric-completion uniqueness route of chapter 1796 before exporting a single limit row. In carrier notation, the two candidate limit displays remain over the same completion source until the separated classifier turns*

the zero-distance row into uniqueness evidence; the packet exports no host equality of metric points, quotient completion representative, selected limit, or ambient completeness theorem.

Proof. Read the completion source first. The two candidate limit rows are compared only after the displayed completion source has been fixed, so the uniqueness claim cannot switch to a second ambient completion object.

Next consume the separated boundary or the metric-completion uniqueness sibling. The separated-completion row supplies the classifier that turns displayed zero distance into a uniqueness-facing read, while the metric-completion uniqueness route records the same comparison as a completion-local obligation.

Finally project the $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ export. Because the export is downstream of those sibling rows, it remains a finite dependency route through displayed completion evidence rather than host equality, quotient representatives, selected limits, or ambient completeness. □

Theorem 5983.34 (CompleteMetricLimitUniqueness RegSeqRat dependency route)**.** *Every* CompleteMetricLimitU *read that reaches a separated completion limit comparison must expose the regular rational readback of chapter 301 before it consumes the separated-completion boundary of chapter 1612, the separated limit uniqueness packet of chapter 3365, or the local separated-completion limit route of Definition 5983.33. The public route remains the finite packet*

$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{SeparatedCompletion}^{\uparrow}, \mathsf{SeparatedLimitUniqueness}^{\uparrow}, \mathsf{CompleteMetri}$

with the hsame*,* Cont*,* Pkg*, and local* NameCert *rows attached to the same readback surface.*

Proof. Start with the regular-Cauchy real route Definition 5983.23. It already places the $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows before any zero-distance comparison can be consumed.

Next use the separated-completion limit route Definition 5983.33. That route admits the separated-completion boundary only after the completion source and zero-distance read have been fixed, so the separated comparison cannot be read from a detached completion object.

Finally attach the separated limit uniqueness packet through chapter 3365. Its carrier reads the two complete-metric limit rows, the common comparison route, the metric distance witness, the Real zero-distance row, and the separated classifier in one displayed packet. Thus the RegSeqRat readback and separated completion limit dependency remain one finite NameCert route, with no quotient completion equality, selected limit, host metric equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate. □

Theorem 5983.35 (Complete metric limit uniqueness RegSeqRat–Real separated dependency route)**.** *Every public* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *handoff must first expose the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, the* $\mathsf{Real}^{\uparrow}$ *zero-distance seal, the* $\mathsf{CompleteMetric}^{\uparrow}$ *completion boundary, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier route of Definition 5983.19. The uniqueness handoff therefore reads the regular-rational, finite-window, dyadic, Real-seal, completion, and separated-reflection dependencies before it is public, and it exports no selected limit, quotient point equality, host equality, or ambient completeness principle.*

Proof. Project the carrier of Definition 5983.1. The shared source row, two limit rows, metric witness, zero-distance row, separated row, transport, replay, provenance, and local name give the whole public uniqueness packet.

Use the RegSeqRat forward route Definition 5983.30. It forces the regular-rational readback, StreamName window, dyadic tolerance ledger, and Real zero-distance seal to be displayed before the separated classifier can be consumed.

Now apply the separated reflection route Definition 5983.19. The $\mathsf{SeparatedMetricReflection}^{\uparrow}$ read is public only after the metric distance witness and $\mathsf{Real}^{\uparrow}$ zero-distance row have been exposed, so the uniqueness handoff cannot bypass the $\mathsf{CompleteMetric}^{\uparrow}$ and Real-facing dependencies.

The structural rows transport, replay, source, and name that same route. Since the carrier has no selected-limit coordinate, quotient point-equality coordinate, host-equality coordinate, or ambient-completeness coordinate, the public handoff remains the listed finite dependency route. □

Theorem 5983.36 (Complete metric limit uniqueness public handoff route)**.** *Every public uniqueness handoff exported by an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *carrier must first consume the complete-metric limit rows, the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *source route, the* $\mathsf{Real}^{\uparrow}$ *zero-distance readback, and the* $\mathsf{SeparatedMetricReflection}^{\uparrow}$ *classifier. In carrier notation the handoff factors through*

$$A, L_0, L_1, M, Z, S, H, C, P, N$$

with L_0, L_1 *supplied by the* $\mathsf{CompleteMetric}^{\uparrow}$ *limit surface,* A *carrying the Cauchy-filter completion source route,* M, Z *carrying the metric and Real zero-distance readback, and* S *carrying the separated-reflection classifier before* H, C, P, N *may replay or name the handoff. The route exports no selected limit, quotient completion point, countable-choice schedule, ambient completeness principle, or host equality of metric points.*

Proof. Start with the sibling lattice route of Definition 5983.20. It exposes the paired $\mathsf{CompleteMetric}^{\uparrow}$ limit rows over the Cauchy-filter completion source before any public uniqueness handoff is available.

Use the Cauchy-filter Real seal route of Definition 5983.26. This forces the metric witness and the $\mathsf{Real}^{\uparrow}$ zero-distance readback to remain downstream of the same source route.

Finally apply the separated-reflection route of Definition 5983.19. The classifier is read only after the zero-distance row is visible, and the structural rows H, C, P, N preserve the same finite route without adding selected limits, quotient points, countable choice, ambient completeness, or host equality. □

Theorem 5983.37 (Complete metric limit uniqueness CauchySequenceSpace forward link)**.** *Every consumer that reads the separated-limit row of an accepted* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *packet after a sequence-based completion presentation first exposes either a paired* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *window or a paired* $\mathsf{RegularCauchySequenceSpace}^{\uparrow}$ *window. The consumer then reads the shared finite regular-family rows, stream schedules, dyadic tolerance ledgers, completion handoff rows, and local naming rows before using the* $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ *separated-limit row. This route cites the Cauchy sequence-space surface of chapter 406 and exports no selected limit, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a sequence-based completion presentation for the two candidate limits. The CauchySequenceSpace packet supplies the displayed regular-family rows, finite StreamName schedules, cofinal-window ledgers, DyadicRatCore tolerances, completion handoff rows, transport, replay, provenance, and local naming.

If the presentation uses a regular Cauchy sequence-space route, the same rows are read through the regular sequence-space sibling before the uniqueness consumer is public. In either case, the

separated-limit comparison has a visible pair of Cauchy sequence windows rather than a hidden quotient stream.

Now apply the CompleteMetricLimitUniqueness separated route already exposed by this chapter. The separated-limit row can be consumed only after the metric-completion and Real-seal rows have been displayed along the same finite comparison route.

Thus the forward link is a dependency edge from the paired CauchySequenceSpace or RegularCauchySequenceSpace windows into the CompleteMetricLimitUniqueness separated-limit row. The packet has no coordinate for selected limits, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

The $\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}$ packet consumes the separated-completion functor of chapter 4094 and the Cauchy-completion functoriality route of chapter 1937 only after the two displayed complete-metric limit rows, the zero-distance row, and the separated classifier have been exposed in the same carrier. The dependency route therefore keeps the two candidate limit reads over one visible completion source before any functorial or separated-completion consumer may export uniqueness.

This child route adds no equality principle beyond the local packet. It records that downstream uniqueness readers must still pass through the paired limit rows, the common zero-distance certificate, the separated classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}_{\mathsf{CompleteMetricLimitUniqueness}^{\uparrow}}$ row before the functorial completion surface is consumed.

Chapter 5984

A Concrete Naming Certificate for $\mathsf{ArchimedeanRealDensity}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ packet records the finite route by which a displayed $\mathsf{Real}^{\uparrow}$ name exposes density data through Archimedean bounds, dyadic brackets, finite stream windows, regular-rational readbacks, and a terminal Real seal. It sits over the dyadic arithmetic of chapter 291, the finite windows of chapter 126, the regular sequence readback of chapter 301, the Real seal of chapter 38, the apartness gap boundary of chapter 304, and the dyadic density route of chapter 1484. The certificate names a density handoff around displayed Real names; it does not import classical trichotomy, quotient equality of real names, a selected representative stream, countable choice, host least-upper-bound machinery, or ambient Real completeness.

Definition 5984.1 (ArchimedeanRealDensity carrier)**.** *An* $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, B, W, R, E, G, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{ArchimedeanReal}^{\uparrow}$ *bound row for the displayed* $\mathsf{Real}^{\uparrow}$ *name and the local rational scale at which density is read.*
2. *B is the* $\mathsf{DyadicDensityReal}^{\uparrow}$ *bracket row naming the dyadic witness inside the displayed interval surface.*
3. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window family opened from the Real name at the precisions requested by A and B.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after the bound, bracket, window, and readback rows have been displayed.*
6. *G is the positive-gap and apartness row, read through* $\mathsf{ApartnessReal}^{\uparrow}$*, that keeps the bracket separated at the displayed scale.*
7. *H stores componentwise* hsame *transport for the bound, bracket, window, readback, Real seal, gap, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from an Archimedean-density request through A, B, W, R, E, G, H.*

9. *P is the* Pkg *provenance over* ArchimedeanReal$^{\uparrow}$, DyadicDensityReal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ApartnessReal$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{ArchimedeanRealDensity}^{\uparrow}}$ *row.*

The carrier contains only finite displayed rows. It stores no trichotomy row, no quotient equality of real names, no selected representative stream, no host least upper bound, no countable-choice selector, and no Real completeness coordinate.

Lean4 ✓ `BEDC.Derived.ArchimedeanRealDensityUp.ArchimedeanRealDensityTasteGate single carrier alignment`

Theorem 5984.2 (ArchimedeanRealDensity NameCert obligations)**.** *For every accepted* ArchimedeanRealDensity$^{\uparrow}$ *carrier $D = (A, B, W, R, E, G, H, C, P, N)$, the local* NameCert$_{\text{ArchimedeanRealDensity}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Archimedean bound, dyadic bracket, finite window, regular readback, Real seal, apartness gap, transport, replay, provenance, and local naming rows;*
2. *bound-bracket compatibility, stating that the dyadic bracket row B is opened only at a scale admitted by the Archimedean row A;*
3. *window-readback exactness, stating that every density observation uses the finite windows W and the regular-rational readback R before the Real seal E is consumed;*
4. *gap stability, stating that the positive apartness row G transports componentwise with A, B, W, R, E through the stored* hsame *and* Cont *routes;*
5. *non-escape, stating that trichotomy, quotient equality of real names, selected representatives, host least upper bounds, countable choice, and Real completeness are absent from the carrier.*

Proof. Project the carrier of Definition 5984.1. The bound, bracket, finite windows, regular readback, Real seal, positive gap, transport table, replay table, provenance row, and local naming row are exactly the available coordinates.

The bound-bracket compatibility row reads A before B. Thus an accepted density request cannot open a dyadic bracket at a scale not already admitted by the Archimedean bound surface.

The window-readback and gap-stability rows then read the same finite route in two directions. The Real seal E is reached only after W and R, while the apartness row G is transported beside those rows through H and C.

Finally inspect the carrier inventory. Since none of the refused host or quotient objects is a coordinate of D, the listed rows exhaust the local NameCert surface. □

Theorem 5984.3 (ArchimedeanRealDensity dyadic-bracket handoff)**.** *Every density read exported by an accepted* ArchimedeanRealDensity$^{\uparrow}$ *carrier factors through the displayed route*

$$A, B, W, R, E$$

before any Real-facing consumer receives it. In particular, the read first uses the ArchimedeanReal$^{\uparrow}$ *bound row A, then the* DyadicDensityReal$^{\uparrow}$ *bracket row B, then the finite* StreamName$^{\uparrow}$ *windows W, then the* RegSeqRat$^{\uparrow}$ *readback row R, and only then the* Real$^{\uparrow}$ *seal row E, with G, H, C, P, N preserving the positive gap, transport, replay, provenance, and local naming.*

Proof. Begin with an exported density read and project the accepted carrier. The only Archimedean scale coordinate is A, so the read begins at the displayed bound row.

The dyadic bracket is then the row B. By the NameCert obligations, B is opened only at a scale admitted by A, and no separate bracket coordinate is available.

The Real-facing part of the read passes through W and R before E. The carrier lists no route from A or B directly to a Real consumer, so the finite window and regular readback rows are mandatory.

The remaining rows G, H, C, P, N keep this path inside the displayed positive gap, transport, replay, provenance, and local naming surface. Therefore every density read factors through A, B, W, R, E before reaching the Real boundary. □

Theorem 5984.4 (ArchimedeanRealDensity Real boundary)**.** *An accepted* $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ *carrier gives its Real-facing consumer only the finite displayed rows*

$$A, B, W, R, E, G, H, C, P, N.$$

It does not export trichotomy, quotient equality of real names, a selected representative stream, a host least-upper-bound object, countable choice, or a Real completeness principle.

Proof. Apply the dyadic-bracket handoff theorem. Any exported density read reaches a Real-facing consumer only after passing through the Archimedean bound, dyadic bracket, finite window, regular readback, and Real seal rows.

The positive-gap row and structural rows do not add a host-side object. They only transport and replay the same displayed carrier surface through G, H, C, P, N.

Inspecting Definition 5984.1 shows that the refused objects are not carrier coordinates. Hence no Real-facing consumer of the accepted packet can receive trichotomy, quotient equality, a selected representative, a host least upper bound, countable choice, or Real completeness from this certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArchimedeanRealDensity}^{\uparrow}}$.

Constructive story: An $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ packet is generated from an ArchimedeanReal bound row, a DyadicDensityReal bracket row, finite StreamName windows, a RegSeqRat readback, a RealUp seal, an ApartnessReal positive-gap row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArchimedeanRealDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5984.1, Definition 5984.2, Definition 5984.3, and Definition 5984.4 seed the finite density packet connecting Archimedean bounds, dyadic brackets, regular readback, Real sealing, and apartness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArchimedeanRealDensity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, quotient equality of real names, selected representatives, host least upper bounds, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bound-bracket compatibility, window-readback exactness, gap stability, and non-escape for BEDC.Derived.ArchimedeanRealDensityUp.

Chapter 5985

A Concrete Naming Certificate for $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$

Following Cantor's nested intersection principle and Bishop–Bridges constructive analysis, the $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$ packet records the finite nested-ball route by which complete metric data serves L10 completion consumers. It sits over the complete-metric service of chapter 128, the metric-space distance surface of chapter 4230, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The packet names a nested closed-ball certificate surface; it does not select an ambient Real limit, assert open-cover compactness, or export a choice-based center.

Definition 5985.1 (CompleteMetricNestedBalls carrier)**.** *A* $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, M, D, I, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *service row that admits the Cauchy-style limit request for the displayed nested ball schedule.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row used to read centers, ball membership, and containment witnesses.*
3. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius ledger, recording the shrinking dyadic radii used by the schedule.*
4. *I is the nested-containment row, requiring each successor ball to be displayed inside its predecessor at the radius requested by D.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which center names and containment checks are inspected.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that converts the displayed windows and dyadic radius ledger into regular rational Cauchy evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after K, M, D, I, S, Q have been displayed.*
8. *H records componentwise* hsame *transport for complete-metric, metric-space, dyadic-radius, containment, stream-window, regular-readback, real-seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay along*

$$K, M, D, I \longmapsto S, Q, E \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricNestedBalls}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient Real-completeness theorem, a selected limit representative, countable choice, quotient completion equality, or a host-side nested-intersection principle.

Lean4 ✓ `BEDC.Derived.CompleteMetricNestedBallsUp.CompleteMetricNestedBallsTasteGate single carrier alignment`

Theorem 5985.2 (CompleteMetricNestedBalls NameCert obligations)**.** *For every accepted* CompleteMetricNestedB *carrier $B = (K, M, D, I, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricNestedBalls}^{\uparrow}}$ *surface consists exactly of carrier admission for the complete-metric service, metric-space distance, positive dyadic radius ledger, nested containment row, finite stream windows, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows; shrinking-radius certification through D; containment certification through I; Cauchy-center handoff through S, Q, E; and non-escape from ambient Real completeness, selected limits, countable choice, quotient completion equality, and host-side nested-intersection principles.*

Proof. Project the carrier of Definition 5985.1. The displayed rows are exactly $K, M, D, I, S, Q, E, H, C, P, N$, and every local obligation must be attached to one of them.

The complete-metric and metric-space rows are read first. They supply the admissible completion service and the distance surface on which centers, ball membership, and containment can be checked.

The dyadic and containment rows provide the nested-ball discipline. The radius ledger D records shrinking positive radii, while I records the finite successor-containment checks over the same metric row.

The Cauchy-center route is then forced through S, Q, E. Finite windows are observed in S, regular rational readback is displayed in Q, and the Real seal E is available only after those rows have been consumed.

Finally H, C, P, N transport, replay, source, and name the same route. Since no listed row is an ambient Real-completeness theorem, selected limit representative, countable-choice selector, quotient completion equality, or host nested-intersection principle, the obligations are exhausted by the displayed packet. □

Theorem 5985.3 (CompleteMetricNestedBalls Cauchy-center route)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$ *packet reads a Cauchy center only through the route*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The nested-containment row must be displayed before the regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ *seal is a seal for that finite route rather than an ambient completeness claim.*

Proof. Begin with the NameCert obligations of Definition 5985.2. They require every accepted consumer to read the complete-metric service K, metric row M, dyadic ledger D, and containment row I before the Cauchy-center handoff can be public.

The finite-window stage comes next. The row S displays the center windows and containment checks, and Q converts those observations into the regular rational readback consumed by completion-facing callers.

The Real seal E is therefore downstream of the displayed nested-ball route. Transport, replay, provenance, and local naming through H, C, P, N preserve that same route without adding a selected limit or an ambient Real-completeness row. □

Theorem 5985.4 (CompleteMetricNestedBalls Real-seal nonescape)**.** *Let* $B = (K, M, D, I, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$ *carrier. Any* $\mathsf{Real}^{\uparrow}$ *endpoint read from* E *is reachable only after the complete-metric service row, metric-space row, shrinking dyadic radius ledger, nested-containment row, StreamName windows, and RegSeqRat readback have been displayed. The endpoint read cannot be consumed as a selected limit representative, quotient completion equality, countable-choice witness, or ambient Real-completeness theorem.*

Proof. Project the carrier in Definition 5985.1. The terminal Real row is E, and it appears after the rows K, M, D, I, S, Q in the displayed packet.

Apply Definition 5985.2. The obligations force shrinking-radius certification through D, nested containment through I, and regular rational Cauchy readback through S, Q before E can be read by a consumer.

Finally use Definition 5985.3. The completion-facing route is finite and named by H, C, P, N, so the endpoint seal remains a seal of that route rather than a selected representative, quotient equality proof, choice witness, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricNestedBalls}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricNestedBalls}^{\uparrow}$ packet is generated from CompleteMetric service, MetricSpace distance, positive DyadicRatCore shrinking radii, nested containment, StreamName windows, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricNestedBalls}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5985.1, Definition 5985.2, and Definition 5985.3 seed the finite nested-ball completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricNestedBalls}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompleteMetricNestedBallsUp.CompleteMetricNestedBallsTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, selected limits, countable choice, quotient completion equality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, shrinking-radius certification, nested-containment certification, Cauchy-center handoff, transport and replay stability, provenance, local naming, and non-escape for BEDC.Derived.CompleteMetr

Chapter 5986

A Concrete Naming Certificate for $\mathsf{CompactUniformCoverChain}^{\uparrow}$

The $\mathsf{CompactUniformCoverChain}^{\uparrow}$ packet realizes the compact-uniform cover-chain route as finite NameCert data. It sits between subordinate finite-cover packets, the compact uniform modulus row of chapter 2299, the compact Lebesgue-number row of chapter 4015, the finite-cover radius stability row of chapter 5203, and the composed uniform-continuity consumer of chapter 3891. The packet records a cover-chain handoff; it does not assert a host modulus selector or global compactness principle.

Falsifiable prediction: Every accepted $\mathsf{CompactUniformCoverChain}^{\uparrow}$ packet must expose the compact-uniform cover-chain route through subordinate finite-cover rows, radius ledgers, pointwise-modulus rows, finite lower-bound folds, uniform-modulus handoff rows, transport, replay, provenance, and local naming data before any compact-uniform consumer may read it.

Independence witness: $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{CompactLebesgueNumber}^{\uparrow}$, $\mathsf{FiniteCoverRadiusStability}^{\uparrow}$, and $\mathsf{UniformContinuityComposition}^{\uparrow}$: these siblings respectively expose a uniform-modulus ledger, a Lebesgue-number radius row, finite-cover radius stability, and composed uniform-continuity handoff; none is bijective with a packet that chains all four through one subordinate finite-cover route.

Definition 5986.1 (CompactUniformCoverChain carrier)**.** *A* $\mathsf{CompactUniformCoverChain}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (K, V, L, R, A, U, T, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact source row, exposing only the finite compact-metric or compact-cover data inspected by the route.*
2. *V is the subordinate finite-cover row, listing the displayed cover cells and their finite refinement relation over K.*
3. *L is the compact Lebesgue-number or finite lower-bound ledger attached to the displayed cover cells.*
4. *R is the finite-cover radius-stability row, keeping the radius ledger stable under the cover refinements consumed by the chain.*
5. *A is the pointwise-modulus row for the continuous map or family currently being tested against the cover.*

6. *U is the compact uniform-modulus handoff row generated from V, L, R, A.*
7. *T is the uniform-continuity composition row that consumes the produced uniform-modulus request without replacing it by a host-side modulus.*
8. *H records componentwise* hsame *transport for compact source, cover, lower-bound, radius-stability, pointwise-modulus, uniform-modulus, composition, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay along*

$$K, V, L, R, A \longmapsto U, T \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCoverChain}^{\uparrow}$, $\mathsf{CompactUniformModulus}^{\uparrow}$, CompactLebesgueNumber FiniteCoverRadiusStability$^{\uparrow}$, $\mathsf{UniformContinuityComposition}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCoverChain}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host compactness theorem, an unrecorded Lebesgue number, a selected global modulus, quotient equality of maps, or a hidden cover refinement.

Theorem 5986.2 (CompactUniformCoverChain NameCert obligations)**.** *For every accepted* CompactUniformCover *carrier $G = (K, V, L, R, A, U, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCoverChain}^{\uparrow}}$ *surface consists exactly of compact-source admission, subordinate finite-cover admission, lower-bound or Lebesgue-number ledger admission, finite-cover radius stability, pointwise-modulus admission, uniform-modulus handoff, uniform-continuity composition handoff, componentwise transport, displayed replay, provenance, and local naming. No NameCert obligation is stored outside those rows.*

Proof. Project the carrier of Definition 5986.1. The only public rows are $K, V, L, R, A, U, T, H, C, P, N$.

The cover-chain route begins with the compact source and subordinate finite-cover rows. The rows K and V determine which finite cover cells and refinements can be inspected.

The radius part is then forced through L and R. The lower-bound or Lebesgue-number ledger supplies positive radius data, and the finite-cover radius-stability row keeps that data synchronized with the displayed refinement.

Pointwise modulus data enters only at A, and the uniform-modulus consumer reads U after the cover and radius rows. The composition row T consumes that generated handoff rather than selecting a separate host-side modulus.

Finally H, C, P, N transport, replay, source, and name the same finite chain. Since the carrier contains no host compactness theorem, hidden Lebesgue number, selected global modulus, quotient equality of maps, or unrecorded refinement, the listed rows exhaust the NameCert surface. □

Theorem 5986.3 (CompactUniformCoverChain uniform-modulus handoff)**.** *Every uniform-modulus read from an accepted* $\mathsf{CompactUniformCoverChain}^{\uparrow}$ *packet factors through the finite route*

$$K, V, L, R, A \longmapsto U, T \longmapsto H, C, P, N.$$

The route supplies subordinate finite-cover data, a radius ledger, pointwise-modulus evidence, and a uniform-continuity composition consumer, but it supplies no host modulus selector or ambient compactness theorem.

Proof. Use the obligations of Definition 5986.2. A uniform-modulus read must first consume compact-source and subordinate finite-cover data from K, V.

The lower-bound and radius-stability rows L, R are then displayed before the pointwise-modulus row A can be folded into a uniform request. The generated uniform-modulus handoff is the row U.

The composition consumer T is downstream of U. Transport and replay through H, C preserve this order, while P, N name the route without adding a separate host-side selector or compactness principle. □

The Lean obligation for this packet is to provide `Nontrivial CompactUniformCoverChainUp` and `FieldFaithful CompactUniformCoverChainUp` TasteGate instances when the carrier is formalized. Those instances must check that the subordinate cover, radius ledger, pointwise-modulus, uniform-modulus, and composition rows are genuine fields of the carrier and not aliases for a single compact-uniform modulus row.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCoverChain}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCoverChain}^{\uparrow}$ packet is generated from compact-source, subordinate finite-cover, lower-bound, radius-stability, pointwise-modulus, uniform-modulus, uniform-continuity composition, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCoverChain}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5986.1, Definition 5986.2, and Definition 5986.3 seed the compact-uniform cover-chain route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCoverChain}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host compactness theorem, selected global modulus, hidden cover refinement, quotient map equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subordinate-cover admission, radius-ledger stability, pointwise-modulus handoff, uniform-modulus handoff, composition consumption, and TasteGate nontriviality and field faithfulness for BEDC.Derived.CompactUniformCoverChainUp.

Chapter 5987

A Concrete Naming Certificate for $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ packet records the finite certificate surface for the usual bounded monotone Cauchy convergence argument. It sits over the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the density handoff of chapter 5984. The certificate keeps monotonicity, Cauchy control, bounds, density witnesses, and the Real-facing seal as displayed BEDC rows; it does not import least-upper-bound completeness, quotient stream equality, selected representatives, trichotomy, countable choice, or an ambient convergence theorem.

Definition 5987.1 (Monotone Cauchy convergence carrier)**.** *A* $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, R, O, B, D, L, E, H, C, P, N)$$

with the following rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the sampled rational stream windows.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that records regular Cauchy control on the same windows.*
3. *O is the monotonicity row, recording the displayed order direction for the inspected window family.*
4. *B is the boundedness row, recording the finite upper or lower bound used by the monotone argument.*
5. *D is the* $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ *density row used to choose rational brackets around the displayed bound gap.*
6. *L is the local convergence ledger, recording the finite tolerance request and the response window.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after S, R, O, B, D, L have been displayed.*
8. *H records componentwise* hsame *transport for the analytic rows and structural rows.*
9. *C records the displayed* Cont *replay by which a consumer reads the stream, regularity, order, boundedness, density, ledger, and Real-seal rows.*
10. *P records* Pkg *provenance over* $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanRealDens}$ $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyConvergence}^{\uparrow}}$ *row.*
The carrier has no coordinate for a supremum object, quotient equality of streams, selected representative sequence, excluded-middle trichotomy, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 5987.2 (Monotone Cauchy convergence NameCert obligations)**.** *For every accepted* $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ *carrier $M = (S, R, O, B, D, L, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MonotoneCauchyConverg}}$ *surface consists exactly of finite-window exposure through S, regular-rational readback through R, displayed monotonicity through O, boundedness through B, Archimedean density bracketing through D, convergence-ledger response through L, terminal* $\mathsf{Real}^{\uparrow}$ *sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the accepted carrier of Definition 5987.1. The analytic rows S, R, O, B, D, L, E are the only coordinates from which a convergence consumer can read the source stream, regularity, order, bound, density bracket, tolerance response, and Real-facing seal.

The structural rows H, C, P, N preserve, replay, source, and name those same rows. Since the carrier contains no supremum coordinate, representative selector, quotient stream equality, trichotomy, choice principle, or ambient completeness coordinate, the local certificate obligations are exhausted by the displayed finite packet. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyConvergenceUp.MonotoneCauchyConvergenceNameCertObligations`

Theorem 5987.3 (Monotone convergence Real seal route)**.** *Every accepted* $\mathsf{Real}^{\uparrow}$ *read through a* $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{ArchimedeanRealDensity}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed by the rows S, R, D, E, H, C, P, N. The monotonicity row O, boundedness row B, and convergence ledger L may refine the route, but they cannot replace the $\mathsf{RegSeqRat}^{\uparrow}$ *readback or the terminal* $\mathsf{Real}^{\uparrow}$ *seal by a host limit.*

Proof. Begin with the carrier tuple $M = (S, R, O, B, D, L, E, H, C, P, N)$. A Real-facing consumer can reach the seal row E only after the stream windows S, regular readback R, density row D, and ledger response L have been displayed.

The monotonicity and boundedness rows are controls on the same finite windows. They do not form a separate completion object, and the replay row C exposes them only together with S, R, D, E.

Transport by H and provenance through P, N keep the same route visible. Thus the accepted Real read is a BEDC handoff through the displayed rows, not a host limit, quotient stream, or selected representative. □

Lean4 ✓ `BEDC.Derived.MonotoneCauchyConvergenceUp.MonotoneCauchyConvergenceRealSealRoute`

Theorem 5987.4 (Monotone Cauchy convergence nonescape)**.** *No consumer of an accepted* MonotoneCauchyConve *packet can extract least-upper-bound completeness, quotient equality of streams, selected representatives, trichotomy, countable choice, or an ambient convergence theorem from the packet. Every accepted convergence read remains bounded by the finite* BHist, hsame, Cont, Pkg, *and local* NameCert *route named in Definition 5987.1.*

Proof. The nonescape claim is carrier projection. The packet exposes only the rows $S, R, O, B, D, L, E, H, C, P, N$.

The convergence route of Definition 5987.3 identifies the only Real-facing handoff: stream windows, regular-rational readback, density bracketing, convergence ledger, and terminal seal. None of these rows is a supremum object, a quotient constructor, a representative selector, or a global completeness principle.

Therefore any accepted consumer stays inside the finite route preserved by H and replayed by C. The excluded principles require coordinates absent from the packet, so they cannot be read from $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$. □

Falsifiable prediction: A consumer that claims a $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ convergence read while omitting the $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, monotonicity row, boundedness row, $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ density bracket, convergence ledger, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the carrier of Definition 5987.1.

Independence witness: streamname, regseqrat, real, archimedeanrealdensity: $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite windows, regular Cauchy readback, monotonicity, boundedness, density bracketing, a convergence ledger, and a terminal Real seal. StreamNameUp carries windows without regularity or convergence; RegSeqRatUp carries regular rational readback without monotone boundedness; RealUp supplies the terminal seal without the convergence ledger; ArchimedeanRealDensityUp supplies density brackets without the monotone Cauchy route.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneCauchyConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{MonotoneCauchyConvergence}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, monotonicity and boundedness rows, $\mathsf{ArchimedeanRealDensity}^{\uparrow}$ density bracketing, a convergence ledger, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneCauchyConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5987.1, Definition 5987.2, Definition 5987.3, and Definition 5987.4 seed the finite constructive monotone Cauchy convergence route over $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{ArchimedeanRealDensity}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneCauchyConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim least-upper-bound completeness, quotient stream equality, selected representatives, trichotomy, countable choice, Lean verification, bridge checking, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, regular readback routing, monotone boundedness stability, density-ledger exactness, and Real-seal nonescape for BEDC.Derived.MonotoneCauchyConvergenceUp.

Chapter 5988

A Concrete Naming Certificate for $\mathsf{UniformCompletionKleisli}^{\uparrow}$

Following Mac Lane on Kleisli categories and Bishop–Bridges and Bourbaki on constructive uniform completion, the $\mathsf{UniformCompletionKleisli}^{\uparrow}$ packet records the L10 interface where a displayed uniform-completion unit and a completion-valued arrow compose without exposing a quotient completion object. It sits after the uniform completion carrier of chapter 1473, the Cauchy completion monad surface of chapter 378, the completion universal property of chapter 397, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real seal of chapter 38. The chapter names only a finite Kleisli route for displayed completion data; it does not assert quotient completion equality, selected limits, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5988.1 (Bishop completion Kleisli carrier)**.** *A* $\mathsf{UniformCompletionKleisli}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, A, B, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformCompletion}^{\uparrow}$ *unit row, naming the source-to-completion admission route.*
2. *A is the Kleisli arrow row, recording a displayed source request and a completion-valued response route.*
3. *B is the bind row, recording composition of two completion-valued arrows through the same unit and arrow surfaces.*
4. *M is the modulus row, recording the finite uniform-continuity budgets needed by A and B.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected by the completion-valued route.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and moduli.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after U, A, B, M, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the unit, Kleisli arrow, bind, modulus, stream-window, readback, Real seal, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from each Kleisli request through $U, A, B, M, S, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionKleisli}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionKleisli}^{\uparrow}}$ *row.*
The carrier has no coordinate for quotient completion equality, selected representatives, selected limits, countable choice, host equality of completed objects, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformCompletionKleisliUp.UniformCompletionKleisliTasteGate single carrier alignment`

Theorem 5988.2 (Bishop completion Kleisli NameCert obligations)**.** *For every accepted* $\mathsf{UniformCompletionKleisli}^{\uparrow}$ *carrier* $K = (U, A, B, M, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionKleisli}^{\uparrow}}$ *surface consists exactly of carrier admission for the uniform-completion unit, the displayed Kleisli arrow and modulus, bind, associativity ledger, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion-facing consumer must read* U, A, B, M, S, R *before the terminal seal and structural rows* E, H, C, P, N *are consumed.*

Proof. Project the carrier of Definition 5988.1. Its rows are exactly the unit, arrow, bind, modulus, stream-window, readback, seal, transport, replay, provenance, and local naming coordinates.

The dependency anchors fix the intended route. The unit row is read through chapter 1473; the bind and associativity ledger is the completion-valued part of the Cauchy monad surface in chapter 378; and the extension route is constrained by chapter 397.

Finally the readback rows pass through chapter 126, chapter 301, and chapter 38. Since the carrier contains no quotient, selector, choice, host-equality, or ambient-completeness row, those objects are not exported by the local NameCert surface. □

Theorem 5988.3 (Bishop completion Kleisli unit route)**.** *Every source object admitted by an accepted* $\mathsf{UniformCompletionKleisli}^{\uparrow}$ *packet enters the completion-facing route through the displayed unit row* U *before any Kleisli arrow* A*, bind row* B*, modulus row* M*, readback row* R*, or Real seal row* E *is inspected. The unit route is a finite* $\mathsf{UniformCompletion}^{\uparrow}$ *handoff, not a selected point of a quotient completion.*

Proof. Read the unit coordinate U from the carrier. By Definition 5988.1, every completion-facing row is downstream of the same finite packet.

Apply the obligation theorem Definition 5988.2. It places U before the arrow, bind, modulus, stream-window, and readback rows.

The remaining rows transport, replay, package, and name that route. They cannot supply a quotient representative or selected completion point because no such coordinate occurs in the carrier. □

Theorem 5988.4 (Bishop completion Kleisli associative bind)**.** *For every accepted* $\mathsf{UniformCompletionKleisli}^{\uparrow}$ *carrier* $K = (U, A, B, M, S, R, E, H, C, P, N)$*, two successive completion-valued arrows compose only through the displayed bind row* B *together with the same unit row* U*, modulus row* M*, stream-window row* S*, regular readback row* R*, and Real seal row* E*. The two parenthesizations of the Kleisli composite have the same componentwise* hsame *transport and displayed* Cont *replay route through* H, C, P, N*.*

Proof. Project the bind coordinate B and the rows it depends on. The carrier records no second completion object and no host composition operator.

Use the monad dependency in chapter 378. It supplies the finite unit-and-bind schedule that the Kleisli packet records as U and B.

Now compare the two parenthesizations componentwise. Both read the same U, M, S, R, E route before transport and replay, so the comparison is witnessed by the displayed hsame row H

and replayed by C. The provenance and local naming rows record that the associativity claim stays inside the finite packet. □

Theorem 5988.5 (Bishop completion Kleisli Real route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{UniformCompletionKleisli}^{\uparrow}$ *packet receives its terminal seal only through*

$$\mathsf{UniformCompletion}^{\uparrow} \longrightarrow \mathsf{CauchyCompletionMonad}^{\uparrow} \longrightarrow \mathsf{CompletionUniversalProperty}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSe}$$

The route exports the displayed unit, Kleisli arrow, modulus, bind, associativity, transport, replay, provenance, and local NameCert rows; it exports no quotient completion, selected limit, countable-choice schedule, host equality, or ambient Real completeness theorem.

Proof. Begin with the unit-route theorem Definition 5988.3. It places every source admission behind the displayed $\mathsf{UniformCompletion}^{\uparrow}$ unit row.

Apply the associative-bind theorem Definition 5988.4. It confines completion-valued composition to the finite bind row and its shared modulus, window, readback, and seal route.

The completion universal property contributes only the extension-facing ledger named in provenance; the StreamName, RegSeqRat, and Real sibling chapters supply the terminal readback chain. Because all excluded objects are absent from K, the Real-facing consumer receives the finite Kleisli route and nothing stronger. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionKleisli}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionKleisli}^{\uparrow}$ packet is generated from a UniformCompletion unit row, a Kleisli arrow and modulus row, a bind row, finite StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionKleisli}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5988.1, Definition 5988.2, Definition 5988.3, Definition 5988.4, and Definition 5988.5 seed the finite Kleisli interface for displayed uniform completion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionKleisli}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, selected limits, countable choice, host equality of completed objects, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, unit-route exposure, Kleisli arrow and modulus exposure, bind associativity, componentwise transport, replay, provenance, and local naming for BEDC.Derived.UniformCompletionKleisliUp.

Chapter 5989

A Concrete Naming Certificate for $\mathsf{KolmogorovComplexity}^{\uparrow}$

Following Kolmogorov and Chaitin, the $\mathsf{KolmogorovComplexity}^{\uparrow}$ packet records finite program-complexity evidence as displayed BEDC rows rather than as an oracle for uncomputable minima. It sits beside the machine interface of chapter 280, the bounded-computation surface of chapter 199, the finite digest discipline of chapter 242, the stream-prefix service of chapter 126, and the binary-code carrier of chapter 5910. The chapter names a finite certificate that a displayed program has bounded length and produces a displayed observation under a displayed universal-machine row; it is not an uncomputable minimum selector, Chaitin incompleteness theorem, quotient program equality, or host semantic universality principle.

Definition 5989.1 (Kolmogorov complexity carrier)**.** *A* $\mathsf{KolmogorovComplexity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (p, U, I, O, F, L, B, E, T, C, P, N)$$

with the following displayed rows:

1. *p is the program row, stored as finite binary or prefix-code data rather than as a host string.*
2. *U is the universal-machine row, read through* $\mathsf{TuringMachine}^{\uparrow}$ *and the bounded computation discipline of* $\mathsf{Computable}^{\uparrow}$*.*
3. *I is the finite input-observation row.*
4. *O is the finite output-observation row produced by replaying p on U for the displayed input.*
5. *F is the finite prefix or code ledger, recording prefix admissibility and code parsing data through the stream-prefix and binary-code sibling rows.*
6. *L is the bounded-length certificate for p and the visible code ledger.*
7. *B is the computable-enumeration provenance row, recording that the displayed program/code rows are reached by a finite enumeration route.*
8. *E is the bounded evaluation transcript row for the displayed input and output observations.*
9. *T records componentwise* hsame *transport for p, U, I, O, F, L, B, E.*
10. *C records displayed* Cont *replay of the same finite program, machine, code, length, enumeration, and evaluation rows.*

11. *P is the* Pkg *provenance over* KolmogorovComplexity$^{\uparrow}$, TuringMachine$^{\uparrow}$, Computable$^{\uparrow}$, Hash$^{\uparrow}$, StreamName$^{\uparrow}$, BinaryTree$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{KolmogorovComplexity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an uncomputable shortest-program selection, quotient equality of programs, host semantic universality, Chaitin incompleteness, or ambient real/stream completeness.

Lean4 ✓ `BEDC.Derived.KolmogorovComplexityUp.KolmogorovComplexityTasteGate single carrier alignment`

Theorem 5989.2 (Kolmogorov complexity NameCert obligations)**.** *For every accepted* KolmogorovComplexity$^{\uparrow}$ *carrier*

$$K = (p, U, I, O, F, L, B, E, T, C, P, N),$$

the local NameCert$_{\text{KolmogorovComplexity}^{\uparrow}}$ *surface consists exactly of carrier admission, finite program-row exposure, universal-machine and* TuringMachine$^{\uparrow}$ *row exposure, finite input and output observation rows, prefix/code ledger exposure, bounded-length certification, computable-enumeration provenance, bounded evaluation transcript, componentwise transport, displayed replay, package provenance, and local naming. The certificate may state that the displayed program has the displayed bounded length for the displayed observation, but it does not select an uncomputable minimum, identify quotient program semantics, prove Chaitin incompleteness, or assert host semantic universality.*

Proof. Project the carrier in Definition 5989.1. The rows p, U, I, O, F, L, B, E are the only analytic data available to a complexity consumer, and the structural rows T, C, P, N only transport, replay, source, and name those data.

Read the machine route through the sibling interfaces. The program row and universal-machine row are consumed through TuringMachine$^{\uparrow}$ and Computable$^{\uparrow}$, while the prefix/code ledger is a finite route through StreamName$^{\uparrow}$ and BinaryTree$^{\uparrow}$; the Hash$^{\uparrow}$ sibling supplies digest-style finite-output comparison discipline when a bounded transcript needs a visible finite output row.

The length claim is bounded and displayed. The row L certifies the length of the shown program/code ledger, and the row E records only the finite evaluation transcript for the shown input and output observations.

Since the carrier contains no row for shortest-program search, quotient program equality, host semantic universality, Chaitin incompleteness, or ambient completeness, the NameCert surface is exactly the displayed finite program-complexity packet. □

Falsifiable prediction: Every accepted KolmogorovComplexityUp read factors through a displayed finite program row, a TuringMachineUp universal-machine row, finite input and output observations, a prefix/code ledger, a bounded-length certificate, computable-enumeration provenance, bounded evaluation replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: turingmachine, computable, hash, streamname, binarytree: KolmogorovComplexity$^{\uparrow}$ is not bijective to any listed sibling because it binds program rows, universal-machine rows, finite observations, prefix/code ledgers, bounded-length certificates, computable-enumeration provenance, evaluation transcripts, transport, replay, provenance, and naming as one finite complexity packet.

Closure status for NameCert$_{\text{KolmogorovComplexity}^{\uparrow}}$.

Constructive story: A KolmogorovComplexity$^{\uparrow}$ packet is generated from finite program rows, a TuringMachineUp universal-machine row, finite input and output observation rows, prefix/code ledgers, bounded-length certificates, computable-enumeration provenance, bounded evaluation transcripts, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KolmogorovComplexity}^{\uparrow}}$, seed)

Scope closed: Definition 5989.1 and Definition 5989.2 seed the finite-program complexity packet over displayed program, machine, prefix/code, length, enumeration, evaluation, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{KolmogorovComplexity}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close uncomputable minimum selection, Chaitin incompleteness, quotient program equality, host semantic universality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, program-row exposure, universal-machine row exposure, finite observation rows, prefix/code ledger exposure, bounded-length certification, computable-enumeration provenance, evaluation replay, transport, provenance, and local naming for BEDC.Derived.KolmogorovComplexityUp.

Chapter 5990

A Concrete Naming Certificate for $\mathsf{RegularCauchyPower}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy real arithmetic, the $\mathsf{RegularCauchyPower}^{\uparrow}$ packet records finite integer powers of one displayed regular Cauchy rational name as a bounded product route. It sits after the regular rational source surface of chapter 301, the finite observation windows of chapter 126, the dyadic arithmetic core of chapter 291, the binary regular product packet of chapter 461, the finite product packet of chapter 2009, and the terminal real-name seal of chapter 38. The carrier packages repeated multiplication by a unary exponent without importing host exponentiation, quotient stream equality, selected limits, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 5990.1 (Regular Cauchy power carrier)**.** *A* $\mathsf{RegularCauchyPower}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (n, A, W, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *n is the finite unary exponent row. It records the requested integer power by a bounded successor chain, with the zeroth row reserved for the displayed unit read.*
2. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row whose scheduled rational observations are multiplied repeatedly.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window family for the source reads, intermediate product reads, and final power-facing read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget and observation row, storing source dyadic observations, intermediate product observations, requested radii, unit cells, and slack cells.*
5. *S is the displayed multiplication schedule. Each successor step cites either a* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *row for the current accumulator multiplied by A or a* $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$ *row for the finite fold named by n.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the final power, obtained only from n, A, W, D, S.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular readback of the power has been displayed.*
8. *H records componentwise* hsame *transport for the exponent, source, windows, dyadic observations, product schedule, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from any finite power request through the same rows.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyPower}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{RegularCauchyFiniteProduct}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyPower}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $n, A, W, D, S, R, E, H, C, P, N$. It has no coordinate for host exponentiation, quotient streams, selected limits, ambient completeness, Lean verification, or bridge credentials.

Lean4 ✓ `BEDC.Derived.RegularCauchyPowerUp.RegularCauchyPowerTasteGate single carrier alignment`

Theorem 5990.2 (Regular Cauchy power NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyPower}^{\uparrow}$ *carrier*

$$K = (n, A, W, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyPower}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the unary exponent, source* $\mathsf{RegSeqRat}^{\uparrow}$ *row, synchronized* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *observations and slack cells, repeated product schedule, final readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local name;*
2. *exponent-window coverage, requiring each successor stage named by n to open the source row A and the accumulator window used by the displayed product step;*
3. *dyadic product-budget exactness, requiring each intermediate product observation to consume its assigned radius and slack cells from D and the schedule row S;*
4. *RegSeqRat readback stability, requiring the final power readback R to depend only on the displayed exponent, source, windows, dyadic budget rows, and product schedule;*
5. *Real seal non-escape, requiring E, H, C, P, N to expose no host exponentiation, quotient stream equality, selected limit, trichotomy, ambient completeness theorem, Lean verification, or bridge credential.*

No Regular Cauchy power obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 5990.1. The displayed tuple $n, A, W, D, S, R, E, H, C, P, N$ is the entire public surface, so carrier admission is exactly the admission of those rows.

Exponent-window coverage is forced by the unary exponent row. The base stage names the displayed unit read, and each successor stage opens the source row A together with the accumulator window selected by W.

Dyadic exactness is the joint role of D and S. The observation row supplies the component and accumulator dyadic cells, while the schedule row records the binary product route or finite product route used at each successor stage.

The final regular readback row R is downstream of n, A, W, D, S. Therefore readback stability follows from the carrier order: no power-facing $\mathsf{RegSeqRat}^{\uparrow}$ data can be inspected until the exponent route, synchronized windows, dyadic observations, and product budget have already been displayed.

Finally, the seal, transport, replay, provenance, and local name rows preserve only this displayed route. Since none of the excluded objects is a coordinate of the carrier, the listed five rows exhaust the local NameCert obligations. □

Theorem 5990.3 (Regular Cauchy power product induction)**.** *Let*

$$K = (n, A, W, D, S, R, E, H, C, P, N)$$

be an accepted RegularCauchyPower$^{\uparrow}$ *carrier. Induction on the finite unary exponent row n shows that every intermediate power read remains inside the displayed product schedule S. The base read is the unit row recorded in D, and each successor read is obtained from the previous accumulator and the source row A through the next* RegularCauchyProduct$^{\uparrow}$ *or* RegularCauchyFiniteProduct$^{\uparrow}$ *schedule cell. Thus no stage of the power read leaves the rows $n, A, W, D, S, R, E, H, C, P, N$.*

Proof. Proceed by induction over the unary exponent row. In the base case, the carrier exposes the unit cell in D and the corresponding base entry in S, so the zeroth power read is already inside the displayed schedule.

For the successor step, assume the accumulator read for the current exponent has been displayed through W, D, S. The next schedule cell in S cites the binary RegularCauchyProduct$^{\uparrow}$ route for multiplying that accumulator by the source row A, or the equivalent RegularCauchyFiniteProduct$^{\uparrow}$ fold cell named by the same finite exponent route.

The dyadic budget row D supplies the observation and slack cells for that product cell, while W supplies the synchronized source and accumulator windows. The successor accumulator is therefore a displayed schedule read, not a host exponentiation step.

After the last successor stage, the final accumulator is repacked as R and then sealed by E. The remaining rows H, C, P, N transport, replay, source, and name the same route, so every intermediate and final power read stays inside the displayed product schedule. □

Falsifiable prediction: Every RegularCauchyPower$^{\uparrow}$ power-facing read factors through a finite unary exponent row, one RegSeqRat source row, synchronized StreamName windows, DyadicRatCore observations and slack cells, a repeated RegularCauchyProduct or RegularCauchyFiniteProduct schedule, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, dyadicratcore, regularcauchyproduct, regular-cauchy-finite-product, real: RegularCauchyPower$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite unary exponent route, a single reusable regular Cauchy source, synchronized accumulator windows, dyadic product-budget accounting, final RegSeqRat readback, RealUp sealing, transport, continuation, provenance, and local naming.

Closure status for NameCert$_{\text{RegularCauchyPower}\uparrow}$.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyPower}\uparrow}$, seed)

Scope closed: Definition 5990.1, Definition 5990.2, and Definition 5990.3 seed the finite power route from a displayed regular Cauchy source through synchronized windows, dyadic product budgets, repeated product schedule cells, final RegSeqRat readback, and RealUp sealing.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyPower}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host exponentiation, quotient stream equality, selected limits, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, exponent-window coverage, budget exactness, readback stability, seal non-escape, and local naming for BEDC.Derived.RegularCauchyPowerUp.

Constructive story: A RegularCauchyPower$^{\uparrow}$ packet is generated from a finite unary exponent row, one RegSeqRat source row, synchronized StreamName windows, DyadicRatCore observations and slack cells, a repeated product schedule, final RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

StreamNameExtensionalityUp reads paired finite-window equality through chapter 126, chapter 291, chapter 301, and chapter 38 before exposing its extensionality rows.

Chapter 5991

A Concrete Naming Certificate for $\mathsf{StreamNameExtensionality}^{\uparrow}$

Following Bishop-style sequence extensionality, the $\mathsf{StreamNameExtensionality}^{\uparrow}$ packet records equality of named streams only through matching finite observation windows. It sits over the finite StreamName interface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a concrete L10 equality object for RegSeqRat and RealUp seal comparisons without quotient stream equality, host function extensionality, selected infinite streams, or countable choice.

Definition 5991.1 (StreamName extensionality carrier)**.** *A* $\mathsf{StreamNameExtensionality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, Y, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{StreamName}^{\uparrow}$ *source rows being compared.*
2. *W is the shared finite-window row listing the observed indices and values inspected from both sources.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the window request when the stream entries feed rational or real-name consumers.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row saying that the paired finite windows give the same regular rational data for the requested precision.*
5. *Q is the terminal comparison row consumed by a* $\mathsf{Real}^{\uparrow}$ *seal route after the StreamName and RegSeqRat reads have matched pointwise on the displayed window.*
6. *H records componentwise* hsame *transport for the two stream rows, the shared window, dyadic ledger, regular readback, terminal comparison, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a finite equality request through W, D, R, Q.*
8. *P is the* Pkg *provenance over* $\mathsf{StreamNameExtensionality}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, W, D, R, Q, H, C, P, N$. It has no coordinate for quotient stream equality, host function extensionality, selected infinite streams, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 5991.2 (StreamName extensionality NameCert obligations)**.** *For every accepted* $\mathsf{StreamNameExtensional}$ *carrier*

$$E = (X, Y, W, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$ *surface consists exactly of paired StreamName admission for* X *and* Y*, pointwise finite-window agreement for* W*, dyadic tolerance ledger visibility for* D*, RegSeqRat readback agreement for* R*, terminal Real-facing comparison for* Q*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, sibling provenance through* P*, and local naming through* N*.*

No StreamName extensionality obligation is stored outside these rows.

Proof. Project the accepted carrier. The two stream names enter only as X and Y, and the only equality-facing observation is the displayed finite window W.

The dyadic ledger D and regular rational readback R consume that same window before any Real-facing comparison can be read. Thus the packet records pointwise finite-window agreement, not quotient equality of streams.

The terminal row Q is downstream of the StreamName, DyadicRatCore, and RegSeqRat rows. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route, so the displayed rows exhaust the NameCert obligations. □

Theorem 5991.3 (StreamName extensionality seal route)**.** *Every accepted* $\mathsf{StreamNameExtensionality}^{\uparrow}$ *packet exports equality evidence to* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumers only through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The route reads paired stream sources, a shared finite window, dyadic tolerance data, regular rational readback, and the terminal comparison row, then preserves the same packet through hsame*,* Cont*,* Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$ *row. It exports no quotient stream equality, host function extensionality, selected infinite stream, countable-choice selector, or ambient Real completeness theorem.*

Proof. Begin with the paired stream rows X, Y. A comparison request can inspect them only through the finite window row W, so the equality surface is pointwise and bounded by the displayed request.

The tolerance ledger D and regular readback R turn the same finite window into the data consumed by RegSeqRat-facing routes. The terminal comparison row Q is therefore a seal over a displayed finite-window readback, not an independent infinite-stream equality.

Finally H, C, P, N transport, replay, source, and name that route. Since the excluded quotient, function-extensionality, selected-stream, choice, and completeness coordinates are absent from the carrier, every RegSeqRat or RealUp equality read factors through the stated finite route. □

Theorem 5991.4 (StreamName extensionality finite-window symmetry)**.** *Let*

$$E = (X, Y, W, D, R, Q, H, C, P, N)$$

be an accepted $\mathsf{StreamNameExtensionality}^{\uparrow}$ *carrier. Swapping the two* $\mathsf{StreamName}^{\uparrow}$ *source rows* X *and* Y *preserves the shared finite window* W*, dyadic tolerance ledger* D*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, terminal* $\mathsf{Real}^{\uparrow}$ *seal comparison* Q*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$ *row* N*.*

Thus the equality evidence exported to RegSeqRat and RealUp consumers is symmetric at the finite-window level and introduces no quotient stream equality, host function extensionality, selected infinite stream, countable-choice selector, or ambient Real completeness theorem.

Proof. Project the carrier of Definition 5991.1. The only ordered coordinates are the two source rows X and Y; the shared window, tolerance ledger, readback, seal comparison, transport, replay, provenance, and local naming rows are displayed independently of that order.

Exchange X and Y while keeping W, D, R, Q, H, C, P, N fixed. The NameCert obligation theorem Definition 5991.2 says that these rows are exactly the local certificate surface, so no hidden equality datum changes under the exchange.

The seal-route theorem Definition 5991.3 then forces every RegSeqRat or RealUp consumer to read the same finite route. Since the carrier has no quotient, function-extensionality, selected-stream, choice, or completeness coordinate, the swapped packet exports only the symmetric finite-window evidence. □

Theorem 5991.5 (StreamName extensionality transitive window route)**.** *Let*

$$E_0 = (X, Y, W_0, D_0, R_0, Q_0, H_0, C_0, P_0, N_0) \quad \textit{and} \quad E_1 = (Y, Z, W_1, D_1, R_1, Q_1, H_1, C_1, P_1, N_1)$$

be accepted $\mathsf{StreamNameExtensionality}^{\uparrow}$ *packets sharing the middle* $\mathsf{StreamName}^{\uparrow}$ *row* Y *and compatible finite windows. A transitive equality read from* X *to* Z *may compose the packets only through the displayed* $\mathsf{StreamName}^{\uparrow}$ *source rows, finite windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, terminal* $\mathsf{Real}^{\uparrow}$ *seal rows, componentwise* hsame *transports, displayed* Cont *replays,* Pkg *provenance rows, and local* NameCert *rows. It exports no quotient stream equality, host function extensionality, selected infinite stream, countable-choice selector, or ambient Real completeness theorem.*

Proof. Apply the NameCert obligations, Definition 5991.2, to E_0 and E_1. They expose the paired StreamName rows, finite windows, dyadic ledgers, regular readbacks, terminal Real-facing comparisons, transport, replay, provenance, and local names as the whole certificate surface.

Use the seal route, Definition 5991.3. It forces each equality read to pass through the finite StreamName-to-DyadicRatCore-to-RegSeqRat-to-Real route before any RegSeqRat or Real consumer may use the comparison.

Finally apply finite-window symmetry, Definition 5991.4, to align the middle row when the two packets are read in opposite orientations. The shared Y row is therefore used only as a displayed finite-window coordinate, and the composed X to Z route retains both sets of ledgers, readbacks, seals, transports, replays, provenance rows, and names. Since neither carrier contains a quotient, host-extensionality, selected-stream, choice, or completeness coordinate, transitivity remains a finite BEDC route. □

Closure status for $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$.

Constructive story: A $\mathsf{StreamNameExtensionality}^{\uparrow}$ packet is generated from two StreamNameUp source rows, a shared finite-window row, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a terminal RealUp comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$, obligation)

Scope closed: Definition 5991.1, Definition 5991.2, Definition 5991.3, and Definition 5991.4 enumerate the paired StreamName admission, finite-window agreement, dyadic ledger, RegSeqRat readback, terminal Real seal comparison, transport, replay, provenance, and local NameCert obligations.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host function extensionality, selected infinite streams, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked paired StreamName admission, finite-window agreement, dyadic tolerance readback, RegSeqRat handoff, Real seal non-escape, transport stability, replay stability, and non-escape for BEDC.Derived.StreamNameExtensionalityUp.

Theorem 5991.6 (StreamName extensionality finite-window determinacy)**.** *Let* $E = (X, Y, W, D, R, Q, H, C, P, N$ *and* $E' = (X', Y', W', D', R', Q', H', C', P', N')$ *be accepted* $\mathsf{StreamNameExtensionality}^{\uparrow}$ *packets. Suppose the two packets display the same finite request index, the same point certificate, and componentwise* hsame *agreement on the inspected window rows. Then every public observation route through* Cont, Pkg*, and the local* $\mathsf{NameCert}_{\mathsf{StreamNameExtensionality}^{\uparrow}}$ *row exposes the same finite-window comparison data for the two packets.*

The determinacy claim is only a finite observation claim. It does not identify host functions, select infinite streams, assert quotient stream equality, or compare completed stream representatives outside the displayed window.

Proof. Project both accepted carriers. The only equality-facing rows available before any public observation are the paired StreamName rows, the finite window rows, the dyadic tolerance ledgers, the RegSeqRat readbacks, and the terminal Real-facing comparison rows.

Use the shared request index and point certificate to align the two window reads. Componentwise hsame agreement then identifies the displayed entries of W and W' at precisely the inspected finite indices, without making any claim about uninspected stream coordinates.

Apply the NameCert obligation theorem of Definition 5991.2. The dyadic ledger, RegSeqRat readback, and terminal comparison rows are downstream of that same finite window, so the aligned window determines the public comparison data available to the packet.

Finally use the seal route of Definition 5991.3. The Cont, Pkg, and local NameCert rows replay, source, and name only the displayed finite route. Therefore two accepted packets with the same request, point certificate, and componentwise window agreement expose the same public observation route, while host function equality and selected infinite stream equality remain outside the packet. □

Theorem 5991.7 (StreamName extensionality RegSeqRat forward link)**.** *Every accepted finite-window determinacy read accepted by a* $\mathsf{StreamNameExtensionality}^{\uparrow}$ *packet passes through the sibling* $\mathsf{RegSeqRat}^{\uparrow}$ *readback of chapter 301 and then through the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before any public comparison data is exposed. The route is finite-window data first, regular rational readback second, and Real-facing seal last; it does not expose host stream equality, quotient stream representatives, or a selected infinite comparison stream.*

Proof. Start from Definition 5991.6. The only comparison data available after aligning the request, point certificate, and componentwise window rows is the finite observation route carried by the accepted packets.

Use Definition 5991.2 to read the downstream rows of that route. The RegSeqRat readback is part of the same finite window ledger, so the public comparison cannot bypass it or replace it by a host rational sequence.

Finally apply Definition 5991.3. The Real-facing comparison rows are exposed only after the seal route has replayed the finite window and regular rational readback, which gives the stated forward link to the sibling RegSeqRat and Real chapters. □

Theorem 5991.8 (StreamName extensionality tail reindex)**.** *Let* $E = (X, Y, W, D, R, Q, H, C, P, N)$ *and* $E' = (X', Y', W', D', R', Q', H', C', P', N')$ *be accepted* $\mathsf{StreamNameExtensionality}^{\uparrow}$ *packets. Suppose* W' *displays the same finite request window as* W *after a tail shift and reindexing of the two* $\mathsf{StreamName}^{\uparrow}$ *source rows, and suppose the shifted window has componentwise* hsame *agreement on the displayed entries, dyadic tolerance rows, regular rational readback rows, terminal comparison rows, replay rows, provenance rows, and local naming rows. Then the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback and the* $\mathsf{Real}^{\uparrow}$*-facing seal route expose the same finite comparison data for* E *and* E'*.*

The conclusion is only a displayed tail-window stability claim. It does not identify quotient streams, host functions, selected infinite streams, countable-choice schedules, or completed real representatives.

Proof. Project the two accepted packets. The tail shift changes only the displayed request indices for the two source rows; the hypothesis supplies the reindexed window W' and the componentwise hsame witnesses comparing it with W.

Apply finite-window determinacy, Definition 5991.6, to the shifted window. The theorem identifies the public observation data available from the aligned windows, because the request, point certificate, and inspected entries agree after reindexing.

Use the NameCert obligation theorem Definition 5991.2. It forces the dyadic ledger, regular rational readback, terminal comparison, transport, replay, provenance, and local naming rows to be read from the same finite window data, so the tail shift cannot introduce a second comparison source.

Finally apply the seal route and the RegSeqRat forward link, Definition 5991.3 and Definition 5991.7. They route the aligned finite window through $\mathsf{RegSeqRat}^{\uparrow}$ and then through the $\mathsf{Real}^{\uparrow}$-facing seal before public comparison data is exposed. Since no quotient stream equality, host function extensionality, selected infinite stream, countable choice, or Real completeness coordinate appears in the carrier, the reindexed tail exposes exactly the same finite comparison data. □

Chapter 5992

A Concrete Naming Certificate for $\mathsf{CaratheodoryMeasureExtension}^{\uparrow}$

Following Caratheodory and the standard extension theorem for premeasures, the $\mathsf{CaratheodoryMeasureExtension}^{\uparrow}$ packet records the finite BEDC route by which displayed premeasure event-value rows are handed through an outer-measure boundary and returned as $\mathsf{Measure}^{\uparrow}$ rows. It sits after the measurable-event schedule of chapter 5186, the outer-measure boundary of chapter 3385, and the root premeasure rows of chapter 92. The packet names only the finite carrier data required to replay the extension read; it does not assert a completed measure space, a host powerset, quotient event equality, countable choice, or Lebesgue completion.

Definition 5992.1 (Caratheodory measure-extension carrier)**.** *A* $\mathsf{CaratheodoryMeasureExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (A, S, O, M, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the premeasure event-value row, including the displayed event family, the finite disjoint-union ledgers already accepted by the* $\mathsf{Measure}^{\uparrow}$ *root premeasure surface, and the* $\mathsf{Real}^{\uparrow}$ *value names attached to those events.*
2. *S is the* $\mathsf{SigmaAlgebra}^{\uparrow}$ *schedule row that records which generated measurable-event rows are admissible before the extension read reaches the target measure row.*
3. *O is the* $\mathsf{OuterMeasure}^{\uparrow}$ *boundary row that carries the cover-valued outer-measure read and the visible handoff from premeasure event values to outer-measure approximants.*
4. *M is the* $\mathsf{Measure}^{\uparrow}$ *target value row exposed after the schedule and outer-measure boundary have been read.*
5. *E is the extension exactness ledger, stating that every accepted target value read agrees with the displayed premeasure row on the admitted source events and with the outer-measure boundary on generated measurable events.*
6. *H records componentwise* hsame *transport for the premeasure, schedule, outer-measure, target-value, exactness, replay, provenance, and local name rows.*
7. *C records displayed* Cont *replay from an extension request through A, S, O, M, E, H.*

8. *P is the* Pkg *provenance over* $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{SigmaAlgebra}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and sibling* NameCert *rows.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryMeasureExtension}^{\uparrow}}$ *row.*

The carrier contains no coordinate for a completed measure space, a host powerset of events, quotient event equality, a countable-choice selector, or Lebesgue completion.

Lean4 ✓ `BEDC.Derived.CaratheodoryMeasureExtensionUp.CaratheodoryMeasureExtensionTasteGate`
`single carrier alignment`

Theorem 5992.2 (Caratheodory measure-extension NameCert obligations)**.** *For every accepted* $\mathsf{CaratheodoryMeasureExtension}^{\uparrow}$ *carrier* $X = (A, S, O, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CaratheodoryMeasureExtension}}$ *surface consists exactly of carrier admission, premeasure compatibility, sigma-algebra schedule admission, outer-measure boundary handoff,* $\mathsf{Measure}^{\uparrow}$ *nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 5992.1. The displayed rows are precisely the premeasure event-value row A, the sigma-algebra schedule S, the outer-measure boundary O, the target value row M, the exactness ledger E, and the structural rows H, C, P, N.

Premeasure compatibility is read from A and E: the ledger compares target values only against source events already displayed in the premeasure row. Schedule admission is read from S, so generated measurable-event requests cannot bypass the sigma-algebra schedule.

The handoff to outer measure is the row O. It is consumed before M, and E records the agreement between the outer-measure boundary and the target $\mathsf{Measure}^{\uparrow}$ value row. Thus a consumer cannot obtain a measure-extension read without the premeasure, schedule, and outer-measure rows already visible.

Finally replay H, C, P, N. These rows transport, continue, source, and locally name the same finite packet. Since no carrier coordinate stores completed measure spaces, host powersets, quotient event equality, countable choice, or Lebesgue completion, the listed rows exhaust the NameCert obligations. □

Theorem 5992.3 (Caratheodory measure-extension outer-measure handoff)**.** *Every accepted* CaratheodoryMeasure *extension read passes through* $\mathsf{OuterMeasure}^{\uparrow}$, $\mathsf{SigmaAlgebra}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows. No accepted read can be completed by a completed measure space, host powerset, quotient event equality, countable choice, or Lebesgue completion.*

Proof. Start from an accepted extension read. Carrier projection exposes the premeasure event-value row A, whose event and value names are read through $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and the finite BHist rows already present in the packet.

The generated-event part of the read is then routed through the $\mathsf{SigmaAlgebra}^{\uparrow}$ schedule row S and the $\mathsf{OuterMeasure}^{\uparrow}$ boundary row O. The target measure value M is not public until those two rows and the extension exactness ledger E have been displayed.

The structural route is fixed by H, C, P, N. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local NameCert row preserve exactly the same finite handoff. Therefore every accepted read passes through the listed BEDC rows and cannot appeal to any absent completion, host-powerset, quotient-equality, choice, or Lebesgue-completion coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{CaratheodoryMeasureExtension}^{\uparrow}}$ ·

Constructive story: A CaratheodoryMeasureExtension$^{\uparrow}$ packet is generated from a displayed premeasure event-value row, a SigmaAlgebra schedule, an OuterMeasure boundary row, a Measure target row, an extension exactness ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: TheoryGate(NameCert$_{\text{CaratheodoryMeasureExtension}^{\uparrow}}$, seed)

Scope closed: Definition 5992.1, Definition 5992.2, and Definition 5992.3 seed the finite extension carrier from premeasure rows through SigmaAlgebra, OuterMeasure, Measure, Real, StreamName, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{CaratheodoryMeasureExtension}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CaratheodoryMeasureExtensionUp.CaratheodoryMeasureExtensionTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not claim completed measure spaces, host powersets, quotient event equality, countable choice, Lebesgue completion, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, premeasure compatibility, sigma-algebra scheduling, outer-measure handoff, Measure nonescape, extension exactness, transport, replay, provenance, and local naming for BEDC.Derived.CaratheodoryMeasureExtensionUp.

Chapter 5993

A Concrete Naming Certificate for $\mathsf{SubspaceTopology}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{SubspaceTopology}^{\uparrow}$ packet records the induced topology on a displayed carrier inside an ambient topological space as finite BEDC naming data. It sits between the indexed-open topology surface of chapter 88, the metric-subspace restriction packet of chapter 1519, and the existing BHist subspace-open surface in Definition 88.69. The packet gives compact-subspace, restriction-of-continuity, and relative-open consumers a general topology-level carrier, without forcing them through metric distance rows, host powersets, quotient subset equality, or selected ambient points.

Before a subspace open is exported, the carrier consumes the ambient $\mathsf{Topology}^{\uparrow}$ open-row discipline of chapter 88 and the inclusion-facing admission classifier carried by its own row A. Thus every relative-open read is already anchored in the ambient indexed-open ledger and the displayed inclusion surface, rather than being a free-standing host subset or an independent topology replacement.

Definition 5993.1 (SubspaceTopology carrier)**.** *A* $\mathsf{SubspaceTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, A, O, R, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the ambient* $\mathsf{Topology}^{\uparrow}$ *row, exposing the point classifier, indexed-open ledger, finite-intersection rows, arbitrary-union rows, boundary opens, and visible open-membership stability.*
2. *A is the displayed admission row for the subspace carrier. A point is available to the subspace packet only when it passes the ambient point classifier of X and the admission row A.*
3. *O is the relative-open row. Each accepted subspace open is displayed as an ambient open row of X together with restriction to admitted points.*
4. *R is the restriction-exactness row, recording that membership in a relative open is read only as simultaneous admission by A and ambient membership in the corresponding open row of X.*
5. *B is the boundary and closure ledger, recording empty, whole-of-subspace, finite-intersection, and arbitrary-union rows obtained by restricting the corresponding ambient topology rows.*
6. *H records componentwise* hsame *transport for the ambient topology, admission row, relative-open row, restriction exactness, boundary and closure ledger, replay, provenance, and local naming rows.*

7. *C records displayed* Cont *replay from a relative-open request through X, A, O, R, B, H.*
8. *P is the* Pkg *provenance over* $\mathsf{SubspaceTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricSubspace}^{\uparrow}$, *Definition 88.69,* BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{SubspaceTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, O, R, B, H, C, P, N$. It has no coordinate for host powersets, quotient subset equality, selected ambient representatives, independent open selectors, metric distance data, or a replacement topology.

Theorem 5993.2 (SubspaceTopology NameCert obligations)**.** *For every accepted* $\mathsf{SubspaceTopology}^{\uparrow}$ *carrier $S = (X, A, O, R, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SubspaceTopology}^{\uparrow}}$ *surface consists exactly of carrier admission for the ambient topology and admission rows, relative-open exposure, restriction exactness, boundary and closure rows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every relative-open consumer must read the route*

$$\mathsf{Topology}^{\uparrow}, A, O, R, B, H, C, P, N$$

before treating a subspace open as public.

Proof. Carrier admission is projection from Definition 5993.1. The available rows are the ambient topology X, admission predicate A, relative-open row O, restriction exactness R, boundary and closure ledger B, and the structural rows H, C, P, N.

Relative-open exposure is the role of O and R. A public subspace membership read is not a host subset membership; it is the displayed conjunction of the admission row A and an ambient open-membership row of X.

The boundary and closure rows are inherited from the ambient topology. Empty and whole-of-subspace opens are obtained by restricting the corresponding boundary opens of X, finite intersections restrict the ambient meet row, and arbitrary unions restrict the ambient union row while keeping the same admission predicate.

Transport and replay preserve the same route. The row H aligns the ambient classifier and admission predicate, C replays relative-open requests through the displayed rows, and P, N source and name the packet.

Since host powersets, quotient subset equality, selected representatives, independent open selectors, metric distance data, and replacement topologies are absent from the carrier, no local NameCert obligation can export them. □

Theorem 5993.3 (SubspaceTopology neighbour compatibility)**.** *Every* $\mathsf{SubspaceTopology}^{\uparrow}$ *packet refines the ambient* $\mathsf{Topology}^{\uparrow}$ *row by restriction to admitted points, and every* $\mathsf{MetricSubspace}^{\uparrow}$ *topology consumer may read the same admission row through this topology-level packet before invoking metric distance data. In particular, the relative-open route factors as*

$$X, A \longmapsto O, R, B \longmapsto H, C, P, N,$$

so compact subspace, continuity restriction, and metric-subspace consumers can share the same displayed subspace carrier while keeping metric distances outside the general topology certificate.

Proof. Start with an accepted $\mathsf{SubspaceTopology}^{\uparrow}$ carrier and project the ambient topology row X and admission row A. These two rows determine the carrier on which all relative-open reads take place.

Use Definition 5993.2. It forces every relative-open read through O, R, B after the admission row has been displayed, and it keeps transport, replay, provenance, and local naming in H, C, P, N.

A $\mathsf{MetricSubspace}^{\uparrow}$ consumer supplies a compatible admission row and an inherited distance ledger, but those metric rows are not needed to form the relative opens. The topology-level packet can therefore be read first, and metric distance data can be invoked only by consumers that explicitly require it.

Thus the packet is the common neighbour of the ambient topology and metric-subspace surfaces: it restricts opens by admission, names the induced topology, and does not add an independent metric or host subset coordinate. □

Theorem 5993.4 (SubspaceTopology continuity restriction)**.** *Let $S = (X, A, O, R, B, H, C, P, N)$ be an accepted* $\mathsf{SubspaceTopology}^{\uparrow}$ *carrier. Any* $\mathsf{ContinuousMap}^{\uparrow}$ *route whose source is the ambient* $\mathsf{Topology}^{\uparrow}$ *row X restricts to admitted subspace points only after the consumer reads $X, A, O, R, B, H, C, P, N$ in order: ambient opens from X, admission from A, relative opens from O, restriction exactness from R, inherited boundary and closure rows from B, transport from H, replay from C, provenance from P, and local naming from N.*

The restricted continuity read is therefore a relative-open route through the displayed subspace packet. It introduces no metric rows, host subset equality, selected ambient representatives, independent open selectors, or replacement topology.

Proof. Project the carrier of Definition 5993.1. A subspace point is available only when the ambient point classifier in X and the admission row A have both been displayed.

Apply the neighbour compatibility theorem Definition 5993.3. It factors every relative-open read through X, A before O, R, B, so the preimage test for a restricted continuous route can inspect only admitted points and ambient open rows.

The restriction-exactness row R supplies the needed membership reading: a point belongs to a relative open exactly through the displayed admission row and the corresponding ambient open-membership row. Thus continuity on the subspace is checked by pulling the ambient open request back through the same relative-open route.

Finally H, C, P, N preserve, replay, source, and name that route. Since no metric distance row, host subset equality, selected representative, independent selector, or replacement topology is present in the carrier, none can be exported by the restricted continuity read. □

Theorem 5993.5 (SubspaceTopology downstream Kelleyfication route)**.** *Let $S = (X, A, O, R, B, H, C, P, N)$ be an accepted* $\mathsf{SubspaceTopology}^{\uparrow}$ *carrier. Any* $\mathsf{Kelleyfication}^{\uparrow}$ *or compactly generated reflection consumer that reads the topology induced on the displayed subspace must first pass through the ambient* $\mathsf{Topology}^{\uparrow}$ *row X, the admission row A, the relative-open row O, the restriction-exactness row R, and the transport and replay rows H, C before the compact-window tests of chapter 6830 may consume the result.*

Proof. Project the carrier in Definition 5993.1. The rows X, A, O, R, H, C are the only displayed route by which an accepted subspace packet exposes its ambient topology, admitted carrier, relative opens, restriction exactness, componentwise transport, and replay.

Apply Definition 5993.3. It factors every relative-open route through X, A before O, R, B and then through H, C, P, N, keeping metric distance data and host subset coordinates outside the general topology certificate.

A $\mathsf{Kelleyfication}^{\uparrow}$ consumer reads compact-window tests from a topology source. Therefore its downstream request must receive the induced topology only after the subspace packet has exposed the ambient topology, admission, relative-open, restriction-exactness, transport, and replay rows.

This is exactly the route displayed by S, so the compactly generated reflection gains no independent topology replacement. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubspaceTopology}^\uparrow}$.

Constructive story: A $\mathsf{SubspaceTopology}^\uparrow$ packet is generated from an ambient TopologyUp row, displayed admission row, relative-open row, restriction-exactness row, boundary and closure ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubspaceTopology}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 5993.1, Definition 5993.2, and Definition 5993.3 seed the finite induced-topology interface over ambient topology, admission, relative opens, restriction exactness, closure rows, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubspaceTopology}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host powerset topologies, quotient subset equality, selected ambient representatives, independent open selectors, metric distance construction, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, relative-open exposure, restriction exactness, inherited boundary and closure rows, neighbour compatibility with MetricSubspaceUp, transport, replay, provenance, and local naming for BEDC.Derived.SubspaceTopologyUp.

Theorem 5993.6 (SubspaceTopology topology forward route)**.** *Every relative-open read of an accepted* $\mathsf{SubspaceTopology}^\uparrow$ *packet first consumes the ambient* $\mathsf{Topology}^\uparrow$ *indexed-open ledger of chapter 88. Only after the ambient point classifier, indexed-open row, finite-intersection row, arbitrary-union row, admission row, relative-open row, restriction-exactness row, transport, replay, provenance, and local naming rows are visible may a* $\mathsf{MetricSubspace}^\uparrow$ *or* $\mathsf{Kelleyfication}^\uparrow$ *consumer use the induced topology. The route is a displayed topology-forward dependency; it does not export a host powerset topology, quotient subset equality, selected ambient representative, metric distance row, or independent open selector.*

Proof. Project the carrier of Definition 5993.1. The ambient topology row X and admission row A appear before the relative-open row O and restriction-exactness row R, so every relative-open read starts with the same displayed $\mathsf{Topology}^\uparrow$ source.

Use the NameCert obligations of Definition 5993.2. They require the indexed-open ledger, finite-intersection row, arbitrary-union row, admission row, relative-open row, restriction-exactness row, transport, replay, provenance, and local naming rows to remain visible before a subspace open is public.

Apply the neighbour compatibility theorem, Definition 5993.3, and the downstream Kelleyfication route, Definition 5993.5. These routes let $\mathsf{MetricSubspace}^\uparrow$ and $\mathsf{Kelleyfication}^\uparrow$ consume the induced topology only after the ambient $\mathsf{Topology}^\uparrow$ ledger has been exposed. Hence the subspace packet supplies a topology-forward dependency and not an independent host topology. □

Theorem 5993.7 (SubspaceTopology topology sibling route)**.** *Every accepted* $\mathsf{SubspaceTopology}^\uparrow$ *read that exports a subspace-open row consumes the ambient* $\mathsf{Topology}^\uparrow$ *carrier row of chapter 88, the subcarrier classifier, restriction and replay rows,* BHist, hsame, Cont, Pkg, *and the local* $\mathsf{NameCert}_{\mathsf{SubspaceTopology}^\uparrow}$ *row. The route is a sibling dependency on the displayed topology packet; it exports no detached host powerset, open-set lattice, quotient subset equality, selected ambient representative, or independent topology replacement.*

Proof. Project the carrier of Definition 5993.1. The ambient topology row X, admission row A, relative-open row O, restriction-exactness row R, replay row C, provenance row P, and local naming row N are visible before any subspace-open row is public.

Apply the NameCert obligations of Definition 5993.2. They force the relative-open read through the ambient $\mathsf{Topology}^{\uparrow}$ carrier, the subcarrier classifier, restriction exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$.

The topology-forward route of Definition 5993.6 identifies this as a horizontal dependency on chapter 88. Since host powersets, open-set lattices, quotient subset equality, selected ambient representatives, and independent topology replacements are absent from the carrier, no sibling consumer can export them. □

Chapter 5994

A Concrete Naming Certificate for NormalizationCandidateFrontier$^{\uparrow}$

The NormalizationCandidateFrontier$^{\uparrow}$ packet realizes the candidate-mediated boundary of chapter 9383 as finite evidence and refusal rows. It sits beside the candidate socket of chapter 3566, the bounded candidate route of chapter 4196, the MetaCIC normalization frontier of chapter 892, and the critical-path packet of chapter 765. The packet separates positive candidate evidence from requests for global strong normalization, so later audit consumers can cite a concrete frontier without turning candidate-mediated evidence into an evidence-free theorem.

Definition 5994.1 (NormalizationCandidateFrontier carrier)**.** *A* NormalizationCandidateFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, E, B, S, I, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed typed MetaCIC term window, recording the term, type, context boundary, and closedness evidence visible to the audit map.*
2. *E is the candidate-evidence row, carrying reducibility candidate, closed-candidate, or candidate-set membership data supplied by the sibling socket.*
3. *B is the bounded-candidate row, recording finite fuel, Pi adequacy, application adequacy, and candidate-mediated route data read through* BoundedNormalizationCandidate$^{\uparrow}$*.*
4. *S is the candidate-supported strong-normalization projection, downstream of E and B.*
5. *I is the no-infinite-reduction projection, downstream of the same candidate-supported route.*
6. *O is the frontier-obstruction row, recording requests for evidence-free closed-term strong normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, or completed closed consistency.*
7. *H records componentwise* hsame *transport for the closed term window, candidate evidence, bounded route, strong-normalization projection, no-infinite-reduction projection, obstruction row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a normalization-frontier request through T, E, B, S, I, O, H.*

9. *P is the* Pkg *provenance over* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$, $\mathsf{NormalizationCandidateSocket}^{\uparrow}$, $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, *chapter 9383,* BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationCandidateFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, E, B, S, I, O, H, C, P, N$. It has no coordinate for global strong normalization, candidate-erased evidence, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, Lean verification, or bridge checking.

Theorem 5994.2 (NormalizationCandidateFrontier stability)**.** *For every accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *carrier $F = (T, E, B, S, I, O, H, C, P, N)$, any public strong-normalization or no-infinite-reduction read is stable only along classifier transport that preserves the candidate evidence row E and bounded route row B. If a consumer keeps the closed term window T but drops E or B, the read reaches the obstruction row O and cannot export S or I from this packet.*

Proof. Project the carrier of Definition 5994.1. The closed term window is T, but the positive normalization rows are not attached directly to T.

The candidate evidence E and bounded route B are the gate. The strong-normalization projection S is downstream of those rows, and the no-infinite-reduction projection I is downstream of the same candidate-supported route.

Classifier transport through H must preserve E and B before S or I can be replayed. The replay row C therefore transports positive evidence only together with the candidate and bounded-route data that justify it.

If a consumer keeps T while dropping E or B, the carrier has no displayed route from the term window to S or I. The frontier-obstruction row O is then the only available response, and it records exactly the evidence-free global requests excluded from the packet. □

Theorem 5994.3 (NormalizationCandidateFrontier audit handoff)**.** *Every MetaCIC normalization audit consumer of an accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *packet factors through the finite route*

$$T \longmapsto E \longmapsto B \longmapsto S \longmapsto I \longmapsto O$$

before citing a candidate-mediated normalization consequence. The route is compatible with the sibling $\mathsf{NormalizationCandidateSocket}^{\uparrow}$, $\mathsf{BoundedNormalizationCandidate}^{\uparrow}$, $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$, *and* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *packets, but it does not claim global strong normalization for every closed typed MetaCIC term.*

Proof. Start with a public audit read. The term row T supplies the closed typed term window, but Definition 5994.2 requires candidate evidence and bounded route data before any positive normalization projection can be consumed.

Read E from the candidate socket surface and B from the bounded candidate route. These rows expose the reducibility-candidate or candidate-set evidence, finite fuel, and Pi/application adequacy data used by the audit consumer.

The strong-normalization projection S and no-infinite-reduction projection I are then available as candidate-mediated consequences. Requests that erase candidate evidence, demand unrestricted subject reduction, assert global confluence, identify quotient syntax, trust host evaluation, or claim completed closed consistency are retained in O.

The structural rows H, C, P, N preserve this route under transport, replay, provenance, and local naming. Therefore the packet is a finite frontier handoff for the audit map, not a global strong-normalization theorem. □

Theorem 5994.4 (NormalizationCandidateFrontier sibling dependency)**.** *Every accepted* NormalizationCandidateF *audit read supplies evidence and refusal rows to the MetaCIC normalization route of chapter 892 and chapter 9383. The rows $E, B, S, I, O, H, C, P, N$ are consumed as candidate-evidence, bounded-route, projection, obstruction, transport, replay, provenance, and local naming data; they are not a replacement for the sibling normalization frontier and do not assert evidence-free closed-term normalization.*

Proof. Use the audit handoff of Definition 5994.3. It routes every public read through the closed term window, candidate evidence, bounded route, positive projections, and obstruction row before a normalization audit consumer may cite the packet.

The MetaCIC normalization frontier sibling consumes exactly those rows as finite evidence and refusal data. The audit map supplies the route on which they are read, while H, C, P, N preserve the same dependency under transport, replay, provenance, and local naming.

Since the packet retains O as an explicit obstruction row, the dependency cannot erase candidate evidence or promote the frontier into a global strong-normalization theorem. □

Theorem 5994.5 (NormalizationCandidateFrontier obligation closure route)**.** *Every accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *read consumed by the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *route of chapter 892 remains valid only while the closed term window T, candidate evidence row E, bounded route B, strong-normalization projection S, no-infinite-reduction projection I, obstruction row O, componentwise transport H, displayed replay C, provenance P, and local naming row N remain visible.*

The obstruction row is part of the dependency lattice. It records evidence-erasing or global normalization requests as refusals rather than surrounding prose.

Proof. Use Definition 5994.4. The sibling dependency theorem sends the candidate-evidence, bounded-route, projection, obstruction, transport, replay, provenance, and naming rows to the MetaCIC normalization route.

Now include the source window from Definition 5994.1. The closed term row T is the term boundary on which E and B are interpreted, so a consumer cannot keep the projections S and I while hiding T.

Finally apply Definition 5994.2. Dropping E or B forces the read into O, and retaining O prevents the consumer from treating candidate-mediated evidence as evidence-free global normalization. Thus $T, E, B, S, I, O, H, C, P, N$ form the visible dependency lattice for the $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ consumer. □

Theorem 5994.6 (NormalizationCandidateFrontier obstruction exactness)**.** *Let $F = (T, E, B, S, I, O, H, C, P, N)$ be an accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *carrier. Any public read that exports the candidate-supported strong-normalization row S or the no-infinite-reduction row I must preserve the candidate-evidence row E, the bounded-candidate route row B, and the obstruction row O. If E or B is absent from the read, the only public route supplied by this packet is the obstruction row O, so evidence-erasing requests cannot be converted into global strong normalization.*

Proof. Project the carrier of Definition 5994.1. The rows S and I are displayed only downstream of E and B, while O records the requests that erase candidate evidence, bounded routes, or the explicit obstruction boundary.

Use stability, Definition 5994.2. It already forces any public strong-normalization or no-infinite-reduction read to preserve E and B under classifier transport. When either row is dropped, the read has no carrier coordinate from which S or I can be exported.

Now apply the audit handoff and sibling dependency results, Definition 5994.3 and Definition 5994.4. They route MetaCIC audit consumers through the candidate socket, bounded candidate packet, normalization frontier, and critical-path surface before any positive projection is public, and they keep O visible as a refusal row rather than prose outside the packet.

Finally use the obligation closure route of Definition 5994.5. Since $T, E, B, S, I, O, H, C, P, N$ are the visible dependency lattice, a read lacking E or B cannot bypass O. The obstruction row is therefore exact for evidence-erasing requests: it is the public result of the packet in those cases, and it cannot be erased into a global strong-normalization claim. □

Theorem 5994.7 (NormalizationCandidateFrontier local confluence scope)**.** *Let $F = (T, E, B, S, I, O, H, C, P, N)$ be an accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *carrier. A local confluence consumer may read a join scope from this packet only when the closed typed window T, candidate evidence row E, bounded route row B, positive projection rows S, I, and obstruction row O remain visible through H, C, P, N.*

The licensed join scope is therefore candidate-mediated frontier evidence for a closed typed window. It is not global confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or completed consistency.

Proof. Project the carrier of Definition 5994.1. The rows T, E, B, S, I, O are the only displayed normalization and obstruction rows, while H, C, P, N transport, replay, source, and name that same finite route.

Apply stability and audit handoff, Definition 5994.2 and Definition 5994.3. They keep candidate evidence and bounded routing visible before S or I can be exported to an audit or confluence-facing consumer.

Use sibling dependency and obligation closure from Definition 5994.4 and Definition 5994.5. These force the closed typed window and frontier siblings to remain part of the same read instead of being replaced by a detached join theorem.

Finally apply obstruction exactness, Definition 5994.6. If E, B, or O is erased, the packet supplies only the obstruction route, so the local join scope cannot become global confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or completed consistency. □

Theorem 5994.8 (NormalizationCandidateFrontier candidate-mediated confluence)**.** *Let $F = (T, E, B, S, I, O, H, C, P, N)$ be an accepted* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *carrier. A local confluence consumer may compose the rows $T, E, B, S, I, O, H, C, P, N$ only while preserving the candidate evidence E, bounded route B, and obstruction row O as visible coordinates of the same packet.*

The composed read is candidate-mediated frontier evidence for the closed typed window T. It cannot promote the packet to global confluence, unrestricted subject reduction, quotient syntax equality, host evaluator soundness, or completed closed consistency.

Proof. Start with stability, Definition 5994.2. It forces every positive normalization read to preserve the candidate-evidence row E and bounded route row B before S or I can be exported.

Apply the audit handoff of Definition 5994.3. The handoff routes MetaCIC consumers through T, E, B, S, I, O before any candidate-mediated consequence is public, so the composition cannot erase the candidate rows.

Use obstruction exactness, Definition 5994.6. The row O remains part of the public route and records evidence-erasing, global confluence, unrestricted subject-reduction, quotient-syntax, host-evaluator, and completed-consistency requests as refusals.

Finally use the local confluence scope of Definition 5994.7. It permits only a closed-window, candidate-mediated join scope transported and replayed by H, C, P, N. Therefore the composed read remains local frontier evidence and cannot become a global normalization or confluence theorem. □

Theorem 5994.9 (NormalizationCandidateFrontier nontrivial witness)**.** *Let $F = (T, E, B, S, I, O, H, C, P, N)$ be an accepted* NormalizationCandidateFrontier$^{\uparrow}$ *carrier. Any accepted read exposing a normalization candidate must preserve whether the exposed row came from the positive candidate-evidence route E, B, S, I or from the retained obstruction boundary O. The local classifier cannot identify these rows under* hsame *transport or* Cont *replay; bounded-route evidence and obstruction retention remain distinct carrier coordinates.*

Proof. Project the carrier of Definition 5994.1. The positive candidate route is carried by E, B, S, I, while the retained obstruction boundary is the separate row O.

Use stability and audit handoff, Definition 5994.2 and Definition 5994.3. They require candidate evidence and bounded-route rows to stay visible before a public normalization projection may be read.

Apply obstruction exactness, Definition 5994.6. A read that drops E or B reaches O instead of exporting S or I, so the obstruction row cannot be identified with the positive route.

Finally use candidate-mediated confluence, Definition 5994.8. It composes only reads that preserve E, B, O, H, C, P, N as visible coordinates of the same packet. Componentwise hsame transport and displayed Cont replay therefore carry the distinction between bounded-route evidence and obstruction retention rather than collapsing the two rows. □

Falsifiable prediction: An accepted NormalizationCandidateFrontier$^{\uparrow}$ consumer that exports closed-term strong normalization while omitting the candidate-evidence row E, bounded route row B, or obstruction row O refutes the packet.

Independence witness: normalization_candidate_socket, bounded_normalization_candidate, metacicnormalizationfrontier, metaciccriticalpath: the socket separates candidate families, the bounded candidate packet records finite fuel and adequacy data, the MetaCIC frontier records normalization boundary rows, and the critical path records dependency pressure; none is bijective with NormalizationCandidateFrontier$^{\uparrow}$ because this packet jointly carries the closed term window, candidate evidence, bounded route, positive projections, explicit obstruction, transport, replay, provenance, and local naming as one frontier carrier.

Later Lean TasteGate coverage for this packet must provide Nontrivial and FieldFaithful instances for the candidate-evidence row family, bounded-route row family, obstruction row, and frontier classifier before BEDC.Derived.NormalizationCandidateFrontierUp can serve as a checked carrier.

Closure status for NameCert$_{\text{NormalizationCandidateFrontier}^{\uparrow}}$ **.**

Constructive story: A NormalizationCandidateFrontier$^{\uparrow}$ packet is generated from a closed typed MetaCIC term window, candidate-evidence row, bounded-candidate route, candidate-supported strong-normalization projection, no-infinite-reduction projection, frontier obstruction, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{NormalizationCandidateFrontier}^{\uparrow}}$, seed)

Scope closed: Definition 5994.1, Definition 5994.2, Definition 5994.3, and Definition 5994.4 seed the finite candidate/refusal boundary for MetaCIC normalization audit consumers.

Formal verification: FormalStatus(NameCert$_{\text{NormalizationCandidateFrontier}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, global strong normalization, candidate-erased closed-term normalization, unrestricted subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed closed consistency, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate evidence preservation, bounded-route preservation, SN and no-infinite-reduction projection stability, obstruction exactness, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.NormalizationCandidateFrontierUp.

Theorem 5994.10 (NormalizationCandidateFrontier typed SN scope)**.** *For every accepted* NormalizationCandidate *carrier* $F = (T, E, B, S, I, O, H, C, P, N)$*, candidate-mediated strong-normalization scope is public only through the typed-term source* T*, candidate row* E*, residual-window and bounded-route row* B*, local join row carried by the audit handoff, normal-form seal represented by the projections* S, I*, transport* H*, replay* C*, provenance* P*, and local* NameCert *row* N*. The scope exports no global strong-normalization theorem, unrestricted subject reduction, evaluator equality, quotient term equality, or kernel consistency.*

Proof. Project the carrier of Definition 5994.1. The typed source, candidate evidence, bounded route, normalization projections, obstruction, transport, replay, provenance, and local naming rows are exactly $T, E, B, S, I, O, H, C, P, N$.

Use the local obligation route of Definition 5994.5. It keeps candidate evidence and bounded-route data visible before any strong-normalization projection can be read, and it retains the obstruction row for evidence-erasing requests.

Apply the audit handoff of Definition 5994.3 and the sibling dependency of Definition 5994.4. They route the candidate-mediated read through the MetaCIC normalization frontier and local audit rows rather than through a host evaluator or quotient term equality.

Thus the typed SN scope is confined to the displayed finite packet. Since no carrier coordinate supplies global SN, unrestricted subject reduction, evaluator equality, quotient term equality, or kernel consistency, none of those endpoints is exported by the theorem. □

Chapter 5995

A Concrete Naming Certificate for $\mathsf{EffrosBorelSpace}^{\uparrow}$

Following Effros and standard descriptive-set-theoretic hyperspace practice, the $\mathsf{EffrosBorelSpace}^{\uparrow}$ packet records the finite hit–miss Borel surface of a hyperspace without importing a host sigma-algebra, a completed measure object, or choice-selected closed sets. It sits beside the compact-subset carrier of chapter 1660, the Vietoris routing of chapter 2375, the compact metric source of chapter 129, and the metric carrier of chapter 57.

Definition 5995.1 (Effros Borel hit–miss carrier)**.** *An* $\mathsf{EffrosBorelSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (H, A, M, V, K, D, T, C, P, N)$$

with the following displayed rows:

1. *H is the underlying* $\mathsf{Hyperspace}^{\uparrow}$ *carrier row, exposing only finite compact-subset names and Hausdorff-facing inspection data.*
2. *A is the hit-row ledger, recording the finite open-window tests whose positive intersection is visible to the packet.*
3. *M is the miss-row ledger, recording the finite closed-window tests whose avoidance is visible to the packet.*
4. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *routing row that places the hit and miss ledgers in the same finite topological basis.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row and D is the* $\mathsf{Met}^{\uparrow}$ *distance row used before any hyperspace read is admitted.*
6. *T records componentwise* hsame *transport for hyperspace, hit, miss, Vietoris, compact-metric, metric, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a hit–miss request through H, A, M, V, K, D, T.*
8. *P is the* Pkg *provenance over* $\mathsf{EffrosBorelSpace}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffrosBorelSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host powerset of closed sets, an ambient Borel sigma-algebra, a completed measure space, a selected representative, quotient closed-set equality, or countable-choice enumeration of tests.

Lean4 (definition) `BEDC.Derived.EffrosBorelSpaceUp.EffrosBorelSpaceUp`

Theorem 5995.2 (Effros Borel NameCert obligations)**.** *For every accepted* $\mathsf{EffrosBorelSpace}^{\uparrow}$ *carrier* $E = (H, A, M, V, K, D, T, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EffrosBorelSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, hit-row stability, miss-row stability, hyperspace nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The hit and miss rows are admitted only through the common* $\mathsf{VietorisTopology}^{\uparrow}$ *route over the displayed* $\mathsf{Hyperspace}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{Met}^{\uparrow}$ *source rows.*

Proof. Project the carrier of Definition 5995.1. The only observable hyperspace data are the finite compact-subset row H, the hit ledger A, the miss ledger M, and the common Vietoris route V.

The compact and metric rows K, D fix the source on which the hit and miss tests are read. Since both ledgers pass through V, a consumer cannot separate hit stability from miss stability by appealing to an external topology.

Transport and replay are then forced through T and C, with provenance and local naming supplied by P, N. No remaining coordinate can introduce a host sigma-algebra, a completed measure object, a quotient closed-set representative, or a choice-selected enumeration. □

Theorem 5995.3 (Effros Borel hyperspace nonescape)**.** *Every Borel-facing consumer of an accepted* $\mathsf{EffrosBorelSpace}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Hyperspace}^{\uparrow}, \mathsf{VietorisTopology}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

before it receives a hit–miss name. The read remains inside the displayed hyperspace packet and exports no host Borel sigma-algebra, no completed measure structure, no selected closed subset, no quotient equality of closed sets, and no countable-choice schedule for tests.

Proof. Begin with a Borel-facing read. Carrier admission exposes the hyperspace row H, but no hit–miss information is public until the hit row A and miss row M have both been routed through the same Vietoris row V.

The source rows K, D keep the route tied to the compact metric and metric carriers already displayed in the packet. Thus a consumer can inspect finite hit and miss windows, but it cannot leave the packet for a host powerset or a detached measurable-space object.

Finally T, C, P, N transport, replay, source, and name the same finite route. The carrier omissions in Definition 5995.1 rule out the remaining exits, so every accepted read is a hyperspace nonescape read. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffrosBorelSpace}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffrosBorelSpace}^{\uparrow}$ packet is generated from a Hyperspace row, finite hit rows, finite miss rows, VietorisTopology routing, CompactMetric and Metric source rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EffrosBorelSpace}^{\uparrow}}$, seed)

Scope closed: Definition 5995.1, Definition 5995.2, and Definition 5995.3 seed the hit–miss hyperspace Borel carrier over Hyperspace, VietorisTopology, CompactMetric, Metric, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EffrosBorelSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host sigma-algebras, completed measure spaces, selected closed-set representatives, quotient closed-set equality, countable-choice test schedules, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, hit-row stability, miss-row stability, hyperspace nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EffrosBorelSpaceUp.

Chapter 5996

A Concrete Naming Certificate for $\mathsf{CountingMeasure}^{\uparrow}$

Following Halmos, Feller, and Kolmogorov, the $\mathsf{CountingMeasure}^{\uparrow}$ packet records the finite counting-measure surface as BEDC NameCert data. It sits over the measurable carrier of chapter 92, the probability-space normalization discipline of chapter 184, the finite index rows of chapter 28 and chapter 36, and the point-mass boundary of chapter 5181. The packet names finite event tallies only; it does not import a host measure object, a powerset sigma algebra, countable additivity, quotient equality of subsets, or a completed scalar measure theory.

Definition 5996.1 (CountingMeasure carrier)**.** *A* $\mathsf{CountingMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (E, F, T, L, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the finite event row, recording the displayed event entries whose membership is being counted.*
2. *F is the finite-support row, listing exactly the displayed points or atoms whose event-membership rows are visible to the packet.*
3. *T is the tally row, assigning to each event row the* $\mathsf{Nat}^{\uparrow}$ *count obtained by traversing the finite support.*
4. *L is the* $\mathsf{List}^{\uparrow}$ *provenance row, recording the finite traversal order and the duplicate-control evidence used by the tally.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *mass readback row, transporting the* $\mathsf{Nat}^{\uparrow}$ *tally to the displayed non-negative scalar endpoint consumed by* $\mathsf{Measure}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *readers.*
6. *H records componentwise* hsame *transport for event rows, finite support rows, tally rows, list provenance, Real mass readback, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an event request through E, F, L, T, R and then to the structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{CountingMeasure}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DiracPointMass}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CountingMeasure}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host measure object, sigma-additivity, a selected ambient support, a powerset event oracle, quotient equality of subsets, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CountingMeasureUp.TasteGate.CountingMeasureTasteGate single carrier alignment`

Theorem 5996.2 (CountingMeasure NameCert obligation surface)**.** *For every accepted* $\mathsf{CountingMeasure}^{\uparrow}$ *carrier* $Q = (E, F, T, L, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CountingMeasure}^{\uparrow}}$ *surface consists exactly of the finite event row, finite support row, Nat tally row, List traversal provenance, Real mass readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every* $\mathsf{Measure}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Distribution}^{\uparrow}$*, or* $\mathsf{DiracPointMass}^{\uparrow}$ *consumer that reads a counting-mass value must expose those rows before it treats the value as a measure-facing endpoint.*

Proof. Project the accepted carrier of Definition 5996.1. The rows E and F supply the only event and finite-support data, so a consumer can inspect membership only through the displayed finite event packet.

The tally is then forced through L and T. The $\mathsf{List}^{\uparrow}$ row records the finite traversal and duplicate-control evidence, while the $\mathsf{Nat}^{\uparrow}$ row records the resulting count; no outside cardinality oracle or selected support is part of the packet.

The scalar mass row R is downstream of that tally. It transports the finite natural count into the displayed $\mathsf{Real}^{\uparrow}$ endpoint used by measure-facing readers, and the route is preserved only by the componentwise hsame transport and displayed Cont replay in H and C.

Finally P and N source and name the same finite packet. Since the carrier contains no host measure object, sigma-additivity row, powerset event oracle, quotient equality, countable choice, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 5996.3 (CountingMeasure finite disjoint-union additivity)**.** *Let an accepted* $\mathsf{CountingMeasure}^{\uparrow}$ *packet display two finite event rows* E_0, E_1 *over the same finite support row* F*, together with a displayed disjointness row saying that no listed support point is counted by both events. Then the tally row for the displayed union event is classified with the* $\mathsf{Nat}^{\uparrow}$ *sum of the two tally rows, and the* $\mathsf{Real}^{\uparrow}$ *mass readback follows the same finite Nat-to-Real route.*

The statement is internal to the displayed carrier rows: it uses the finite support, event rows, tally rows, $\mathsf{List}^{\uparrow}$ *traversal provenance,* $\mathsf{Nat}^{\uparrow}$ *addition ledger,* $\mathsf{Real}^{\uparrow}$ *mass readback, componentwise transport, replay, provenance, and local naming. It does not assert countable additivity, a powerset sigma algebra, quotient equality of subsets, a host measure object, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier through Definition 5996.1. The two event rows and their union are read over the same finite support row F, so the traversal is the single $\mathsf{List}^{\uparrow}$ route already recorded by L.

Traverse the support once. At each listed point, the displayed disjointness row splits membership into the E_0 side, the E_1 side, or neither side; the mixed counted-by-both case is refused by that row.

Accumulate the two tallies through the $\mathsf{Nat}^{\uparrow}$ addition ledger. Because every counted point contributes to exactly one of the two displayed event rows, the union tally is the finite sum of the two component tallies.

Transport the result through the mass readback row R and the structural rows H, C, P, N. This gives the same finite Nat-to-Real route used by the NameCert obligation surface, without importing countable additivity or any ambient host measure structure. □

Closure status for $\mathsf{NameCert}_{\mathsf{CountingMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{CountingMeasure}^{\uparrow}$ packet is generated from finite event rows, finite support rows, Nat tally rows, List traversal provenance, RealUp mass readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountingMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5996.1 and Definition 5996.2 seed the finite counting-measure packet over MeasureUp, ProbSpaceUp, NatUp, ListUp, RealUp, and DiracPointMassUp rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountingMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim sigma-additivity, a host measure object, powerset sigma-algebra generation, quotient equality of subsets, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite event support, Nat tally exactness, List traversal provenance, Real mass readback, transport, replay, provenance, and local naming for BEDC.Derived.CountingMeasureUp.CountingMe

Chapter 5997

A Concrete Naming Certificate for $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$

Following Bishop and Bridges on rational interval approximations and Bourbaki on bases for ordered topological spaces, the $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ packet records the finite BEDC carrier for rational open balls on the real line. It sits between the rational arithmetic of chapter 37, the dyadic tolerance rows of chapter 291, the finite-window naming surface of chapter 126, the regular-rational handoff of chapter 301, the real seal of chapter 38, the second-countability boundary of chapter 1901, and the dyadic interval comparison packet of chapter 1916. The chapter names only finite rational centers, positive radii, membership rows, finite observation windows, regular-sequence handoff, terminal real seals, transport, replay, provenance, and local naming.

Definition 5997.1 (Rational open interval basis carrier)**.** *A* $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (Q, D, M, S, G, E, H, C, P, N)$$

whose rows have the following displayed roles.

1. *Q is the rational-center row, carrying the* $\mathsf{Rat}^{\uparrow}$ *center and the rational arithmetic needed to compare endpoints.*
2. *D is the positive radius row, displayed either as a positive rational radius or through a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *refinement.*
3. *M is the interval-membership row, exposing exactly the two strict rational inequalities that certify an inspected point lies between the displayed endpoints.*
4. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the coordinates read before the interval predicate is evaluated.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row, sending the inspected rational-window evidence into a regular rational sequence name.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after Q, D, M, S, G have all remained visible.*
7. *H is the componentwise* hsame *transport for the displayed rows.*
8. *C is the displayed* Cont *replay route for rational centers, radii, membership checks, finite windows, regular-sequence handoff, and real seals.*

9. *P is the* Pkg *provenance over* $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, $\mathsf{DyadicIntervalBasis}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{RationalOpenIntervalBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, D, M, S, G, E, H, C, P, N$. It has no coordinate for host topology, arbitrary open sets, quotient equality of reals, selected representatives, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.RationalOpenIntervalBasisUp.RationalOpenIntervalBasisTasteGate single carrier alignment`

Theorem 5997.2 (Rational open interval basis NameCert obligations)**.** *For every accepted* RationalOpenIntervalBas *carrier $I = (Q, D, M, S, G, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RationalOpenIntervalBasis}^{\uparrow}}$ *surface consists exactly of the following five obligation rows.*

1. *carrier admission for the displayed rational center Q and positive radius D;*
2. *interval-membership exposure through M, with membership read only by the strict endpoint inequalities determined by Q, D;*
3. *finite-window exposure through S, requiring every inspected real name to pass through a displayed* $\mathsf{StreamName}^{\uparrow}$ *window before M is consulted;*
4. *regular-to-real handoff through G, E, requiring the* $\mathsf{RegSeqRat}^{\uparrow}$ *row before the terminal* $\mathsf{Real}^{\uparrow}$ *seal;*
5. *structural closure through H, C, P, N, with provenance over the rational, dyadic, stream, regular-sequence, real, second-countable, and dyadic-interval sibling rows.*

No obligation row may be replaced by a host topology, an arbitrary open set lattice, quotient equality of reals, selected representatives, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.RationalOpenIntervalBasisUp.RationalOpenIntervalBasisCarrier namecert obligations`

Proof. Project the carrier of Definition 5997.1. The rows Q and D are the only rows carrying the center and radius data, and M is the only row assigned to interval membership.

The finite observation discipline is supplied by S. Any point-facing membership read must first expose its $\mathsf{StreamName}^{\uparrow}$ window, after which G and E provide the regular-sequence handoff and terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally, H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no coordinate for host topology, arbitrary opens, quotient real equality, selected representatives, countable choice, or ambient completeness, the five obligations exhaust the local naming surface. □

Theorem 5997.3 (Rational open interval basis finite-window nonescape)**.** *For every accepted* $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ *carrier $I = (Q, D, M, S, G, E, H, C, P, N)$, every basis-membership read factors through the finite route*

$$\mathsf{Rat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

before any second-countability or dyadic-interval consumer can cite it. The read exports no host topology, arbitrary open set, quotient equality of reals, selected representative, countable choice, or ambient real-completeness endpoint.

Proof. Use Definition 5997.2. The membership row M cannot be read until the center row Q, radius row D, and finite stream window S have been displayed.

The completion-facing part of the route is then G, E. These rows place the regular rational handoff before the terminal real seal, so downstream references by $\mathsf{SecondCountable}^{\uparrow}$ or $\mathsf{DyadicIntervalBasis}^{\uparrow}$ receive only the finite basis packet.

The structural rows H, C, P, N add transport, replay, provenance, and naming, but no host topology or quotient endpoint. Thus the finite-window route is the whole export surface. □

Falsifiable prediction: Every accepted $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ membership read exposes a rational center row, a positive rational or dyadic radius row, the strict interval-membership row, a finite StreamName window, a RegSeqRat handoff, a RealUp seal, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: rat, dyadicratcore, streamname, regseqrat, real, secondcountable, dyadicintervalbasis: $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ is not identical to any listed sibling because it binds rational centers and positive-radius membership before the finite-window handoff to regular rational names and the terminal RealUp seal.

Closure status for $\mathsf{NameCert}_{\mathsf{RationalOpenIntervalBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{RationalOpenIntervalBasis}^{\uparrow}$ packet is generated from rational center rows, positive rational or dyadic radius rows, strict interval-membership rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RationalOpenIntervalBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 5997.1, Definition 5997.2, and Definition 5997.3 seed the finite rational-center, positive-radius, membership, window, handoff, seal, transport, replay, provenance, and local naming surface over $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{SecondCountable}^{\uparrow}$, and $\mathsf{DyadicIntervalBasis}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RationalOpenIntervalBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host topology, arbitrary open set lattices, quotient equality of reals, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rational-center and positive-radius exactness, strict interval-membership exposure, finite-window membership exactness, finite-window handoff, $\mathsf{RegSeqRat}^{\uparrow}$-to-$\mathsf{Real}^{\uparrow}$ sealing, terminal seal nonescape, transport, replay, provenance, and local naming for the same finite packet and for BEDC.Derived.RationalOpenIntervalBasisUp.

Chapter 5998

A Concrete Naming Certificate for $\mathsf{FreudenthalEnd}^{\uparrow}$

Following Freudenthal and the classical theory of ends of locally compact spaces, the $\mathsf{FreudenthalEnd}^{\uparrow}$ packet records escaping components of a displayed locally compact or proper metric source as finite NameCert data. It reads the compact-neighbourhood surface of chapter 1726, the closed-ball compactness route of **??**, the compact metric witness of chapter 129, and the metric carrier of chapter 57. The packet names the tail-equivalence boundary of escaping rays without importing a host compactification, an ideal point selector, open-cover compactness, quotient end space, or choice-selected ray representative.

Definition 5998.1 (FreudenthalEnd carrier)**.** *A* $\mathsf{FreudenthalEnd}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, E, C_0, C_1, T, R, H, Q, P, N)$$

with the following displayed rows:

1. *X is the source metric or locally compact row, including the* $\mathsf{Met}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, *and optional* $\mathsf{ProperMetric}^{\uparrow}$ *support used by the packet.*
2. *E is the exhaustion row by compact metric subsets. Each inspected stage exposes a compact witness through* $\mathsf{CompactMetric}^{\uparrow}$ *and, in the proper metric case, through the displayed closed-ball route of* $\mathsf{ProperMetric}^{\uparrow}$.
3. *C_0 and C_1 are component rows for complements outside compact stages. A row records only the finite component identifier, the compact stage it avoids, and the displayed adjacency data needed to replay the complement read.*
4. *T is the tail-comparison ledger. It records that the two component rows eventually enter the same complement component after passing to a later compact stage.*
5. *R is the escaping-ray ledger. It records finite ray prefixes together with the compact stages already avoided and exposes no selected infinite ray object.*
6. *H records componentwise* hsame *transport for the source, exhaustion, complement-component rows, tail ledger, ray prefixes, provenance, and local naming rows.*
7. *Q records displayed* Cont *replay from an end request through X, E, C_0, C_1, T, R, H.*
8. *P is the* Pkg *provenance over* $\mathsf{FreudenthalEnd}^{\uparrow}$, $\mathsf{LocallyCompact}^{\uparrow}$, $\mathsf{ProperMetric}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{FreudenthalEnd}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host compactification, an ideal boundary point, a quotient space of ends, open-cover compactness, a selected infinite ray, or countable-choice enumeration of components.

Lean4 ✓ `BEDC.Derived.FreudenthalEndUp.FreudenthalEndTasteGate single carrier alignment`

Theorem 5998.2 (FreudenthalEnd NameCert obligation surface)**.** *For every accepted* $\mathsf{FreudenthalEnd}^{\uparrow}$ *carrier* $F = (X, E, C_0, C_1, T, R, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FreudenthalEnd}^{\uparrow}}$ *surface consists exactly of source admission, compact-exhaustion admission, complement-component stability, tail-equivalence readback, escaping-ray prefix replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream end consumer must read the compact exhaustion E before either complement row is public, must read both C_0 and C_1 before the tail ledger T is public, and must keep the ray-prefix ledger R separate from any host-selected ideal point.*

Proof. Project the carrier of Definition 5998.1. The only source data are the displayed metric, locally compact, and proper metric rows in X, and the only compact boundary data are the finite compact stages in E.

The complement rows C_0 and C_1 can therefore be inspected only after the compact stage they avoid has been displayed. The tail ledger T compares those component rows by passing to later compact stages, so it is a finite readback of eventual component agreement rather than a quotient end object.

The ray-prefix row R supplies only finite escaping prefixes and the compact stages already avoided. Transport through H and replay through Q preserve the same source, exhaustion, component, tail, and prefix rows.

Finally P and N name precisely this displayed packet. Since no carrier coordinate stores a compactification point, quotient end space, open-cover compactness proof, selected infinite ray, or countable-choice component enumeration, the listed rows exhaust the local NameCert obligation surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FreudenthalEnd}^{\uparrow}}$.

Constructive story: A $\mathsf{FreudenthalEnd}^{\uparrow}$ packet is generated from a locally compact or proper metric source, a compact metric exhaustion row, complement-component rows outside displayed compact stages, a tail-equivalence readback ledger, escaping-ray finite prefixes, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FreudenthalEnd}^{\uparrow}}$, seed)

Scope closed: Definition 5998.1 and Definition 5998.2 seed the Freudenthal end carrier over LocallyCompact, ProperMetric, CompactMetric, MetricSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FreudenthalEnd}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim a host compactification, quotient end topology, selected ideal points, selected infinite rays, countable-choice component enumeration, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-exhaustion stability, complement-component stability, tail-equivalence readback, escaping-prefix replay, transport, replay, provenance, and local naming for BEDC.Derived.FreudenthalEndUp.

Chapter 5999

A Concrete Naming Certificate for $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$

Following Baire and Osgood, the $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$ packet records the finite window of the Baire-category construction that a completion-facing consumer can inspect before any ambient theorem about residual sets is available. It sits over the complete-metric Baire surface of chapter 3535, the complete-metric service of chapter 128, the convergence ledger of chapter 2074, the pointwise stream metric route of chapter 4139, and the nested-ball completion neighbor chapter 5985. The packet is a finite-means dense-open route for Real and Completion consumers; it is not a countable-choice selector, arbitrary open-cover compactness theorem, quotient stream equality, host completeness assertion, or selected generic point.

Definition 5999.1 (Finite-window Baire category carrier)**.** *A* $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, U, D, Q, S, R, E, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the complete metric source row. It exposes only the complete-metric service already named by* $\mathsf{CompleteMetric}^{\uparrow}$ *and the metric-distance rows consumed by the window.*
2. *U is the nested dense-open window. It records a finite prefix of dense-open requests, the displayed ball-refinement clauses, and the order in which those clauses are read.*
3. *D is the dyadic radius ledger. It stores the finite sequence of positive dyadic radii, their shrink comparisons, and the lower-radius bounds needed before the next dense-open request is inspected.*
4. *Q is the finite center and ball ledger. It records the named center histories, ball inclusions, and branch-local containment rows produced by the complete-metric source.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row for the center and radius observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the scheduled finite observations and the dyadic radius ledger.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, U, D, Q, S, R have been displayed.*
8. *T is the completion transport row. It records the consumer-facing route from the complete-metric source to the Real seal without replacing the source by a host completeness theorem.*

9. *H records componentwise* hsame *transport for complete-metric rows, dense-open windows, dyadic radius ledgers, center ledgers, stream schedules, regular readbacks, Real seals, completion transport, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a finite Baire-window request through $M, U, D, Q, S, R, E, T, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{PointwiseStreamMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowBaireCategory}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, D, Q, S, R, E, T, H, C, P, N$. It has no coordinate for countable choice, arbitrary open-cover compactness, quotient stream equality, host completeness, selected generic points, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 (definition) `BEDC.Derived.FiniteWindowBaireCategoryUp.FiniteWindowBaireCategoryUp`
Lean4 ✓ `BEDC.Derived.FiniteWindowBaireCategoryUp.FiniteWindowBaireCategoryTasteGate single carrier alignment`

Theorem 5999.2 (Finite-window Baire category NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$ *carrier*

$$W = (M, U, D, Q, S, R, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowBaireCategory}^{\uparrow}}$ *surface consists exactly of complete-metric source exposure through M, nested dense-open finite-window exposure through U, dyadic radius-ledger exactness through D, finite center and ball routing through Q,* $\mathsf{StreamName}^{\uparrow}$ *scheduling through S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *sealing through E, completion transport through T, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 5999.1. The projection displays the complete-metric source, the dense-open window, the dyadic radius ledger, the center and ball ledger, the stream schedule, the regular readback, the Real seal, and the structural transport, replay, provenance, and naming rows.

The dense-open part is finite. Row U gives only the displayed prefix of dense-open requests, while D and Q record the radius shrinkage and center-ball refinements used by that prefix. Thus a consumer cannot replace the window by a host countable intersection or an arbitrary open-cover compactness theorem.

The Real-facing part is downstream. Rows S and R schedule and repack the finite observations, and E is read only after that regular-rational route is visible. The completion transport row T preserves the same displayed source rather than adding a separate host-completeness claim.

Finally H, C, P, N transport, replay, source, and name exactly those rows. Since the carrier contains no countable-choice selector, quotient stream coordinate, host completeness field, or selected generic point, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowBaireCategory}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowBaireCategory}^{\uparrow}$ packet is generated from a complete metric source row, nested dense-open finite windows, a dyadic radius ledger, finite center and ball rows, StreamName schedules, RegSeqRat readback, a RealUp seal, completion transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteWindowBaireCategory}\uparrow}$, seed)

Scope closed: Definition 5999.1 and Definition 5999.2 seed the finite Baire dense-open window consumed by Real and Completion routes.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWindowBaireCategory}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close countable choice, arbitrary open-cover compactness, quotient stream equality, host completeness, selected generic points, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open window exposure, dyadic radius-ledger exactness, StreamName scheduling, RegSeqRat readback, Real sealing, completion transport, provenance, and local naming for BEDC.Derived.FiniteWindowBaireCategoryUp.

Chapter 6000

A Concrete Naming Certificate for $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$

Following Lebesgue and Heine–Cantor, the $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$ packet records the finite radius ledger extracted from a compact metric subcover before a uniform-modulus consumer reads it. It sits over the compact metric carrier of chapter 129, the finite-subcover criterion of chapter 3503, the compact-cover Lebesgue ledger of chapter 1302, the rational lower-bound fold of chapter 4096, and the uniform-modulus surface of chapter 369. The packet exposes the compact finite-cover lower-bound route as BEDC data without host compactness, choice-selected centers, arbitrary open-cover compactness, semantic continuity, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6000.1 (Finite subcover radius ledger carrier)**.** *A* $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, G, R, B, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the metric carrier, finite-net, and total-bounded inspection rows available to compact-metric consumers.*
2. *F is the* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *row. It records the finite selected subcover spine and the displayed coverage witnesses over the compact metric source.*
3. *G is the compact-cover Lebesgue ledger row. It stores the positive radius candidates attached to the displayed subcover entries and the local containment clauses they support.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, obtained by primitive recursion over the finite subcover spine and read only as a positive rational history.*
5. *B is the uniform radius bound row. It records that the folded lower bound is below each displayed local radius and therefore may be consumed by the compact-uniform route.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row reached only after K, F, G, R, B have been projected.*
7. *H records componentwise* hsame *transport for compact-metric rows, finite-subcover rows, Lebesgue-ledger rows, rational lower-bound folds, uniform-radius bounds, uniform-modulus handoffs, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-cover request through $K, F, G, R, B, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$, $\mathsf{CompactCoverLebesgueLedger}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, F, G, R, B, U, H, C, P, N$. It has no coordinate for host compactness, selected centers outside the displayed subcover, arbitrary open-cover compactness, semantic continuity, external finite minima, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FiniteSubcoverRadiusLedgerUp.TasteGate.FiniteSubcoverRadiusLedgerTasteGate`
`single carrier alignment`

Theorem 6000.2 (Finite subcover radius ledger NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$ *carrier*

$$L = (K, F, G, R, B, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}}$ *surface consists exactly of* $\mathsf{CompactMetric}^{\uparrow}$ *source exposure through K,* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *coverage exposure through F, CompactCoverLebesgueLedger radius exposure through G,* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold exactness through R, uniform radius-bound routing through B,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6000.1. The rows $K, F, G, R, B, U, H, C, P, N$ are the only displayed coordinates, so every compact-uniform consumer begins with the compact metric source and the finite subcover spine.

Rows F and G supply the coverage and local positive-radius data. The subcover is already finite as a displayed BHist spine, and the Lebesgue ledger gives radius candidates only for those displayed entries.

Row R folds those rational candidates by the $\mathsf{Rat}^{\uparrow}$ lower-bound route, and B records that the folded bound is below each local radius. The $\mathsf{UniformModulus}^{\uparrow}$ handoff U is downstream of that finite bound rather than a separate compactness theorem.

Finally H, C, P, N transport, replay, source, and name the same route. Since no coordinate stores host compactness, selected centers outside the subcover, arbitrary open-cover compactness, semantic continuity, an external finite minimum, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}$ packet is generated from a CompactMetric source row, a FiniteSubcoverCriterion coverage row, a compact-cover Lebesgue radius ledger, a RatUp lower-bound fold, a uniform radius-bound row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}}$, seed)

Scope closed: Definition 6000.1 and Definition 6000.2 seed the finite subcover lower-bound route consumed by compact-uniform-modulus arguments.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteSubcoverRadiusLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactness, selected centers outside the displayed subcover, arbitrary open-cover compactness, semantic continuity, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, CompactMetric source exposure, FiniteSubcoverCriterion coverage, compact-cover Lebesgue radius exposure, RatUp lower-bound fold exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSubcoverRadiusLedgerUp.

Chapter 6001

A Concrete Naming Certificate for BishopLocatedInverseFunction$^{\uparrow}$

Following Bishop–Bridges constructive analysis, the BishopLocatedInverseFunction$^{\uparrow}$ packet records the finite local inverse read for a located continuous real map after interval location, apartness separation, regular rational readback, dyadic tolerance accounting, and the terminal real-name seal have all been displayed. It sits beside the monotone inverse route of chapter 1929, the local analytic inverse route of chapter 2055, the continuous-map surface of chapter 130, the located apartness bridge of chapter 3564, and the L10 real-name lattice of chapter 126, chapter 301, chapter 291, and chapter 38. The chapter names a located local inverse certificate, not a host inverse function, quotient real equality, trichotomy, selected representative, ambient completeness principle, or bridge verification.

Definition 6001.1 (Bishop located inverse function carrier)**.** *A* BishopLocatedInverseFunction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (F, A, B, U, V, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* ContinuousMap$^{\uparrow}$ *row naming the displayed map and the finite continuity modulus cells used by the local read.*
2. *A is the* LocatedApartnessRealBridge$^{\uparrow}$ *row. It records the located separation data that keeps candidate preimage windows apart before any inverse request is admitted.*
3. *B is the source and target interval row, carrying the displayed source window, target window, endpoint images, and located target request.*
4. *U is the uniqueness-window row. It records the finite apartness comparison showing that two accepted source windows for the same target request cannot remain distinct under the displayed bridge data.*
5. *V is the existence-window row. It records the finite bisection or refinement schedule that locates a source window for the displayed target request.*
6. *S is the* StreamName$^{\uparrow}$ *finite-window row opened for source endpoints, target requests, function values, and retained source windows.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *readback row attached to the same stream windows.*

8. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row recording dyadic radii, branch budgets, comparison slack, and endpoint-readback tolerances.*
9. *R is the terminal* Real$^{\uparrow}$ *seal row reached only after the continuous-map, located-apartness, interval, uniqueness, existence, stream, regular rational, and dyadic rows have been displayed.*
10. *H records componentwise* hsame *transport for all rows listed above, together with replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from a located inverse request through $F, A, B, U, V, S, Q, D, R, H, P, N$.*
12. *P is the* Pkg *provenance over* BishopLocatedInverseFunction$^{\uparrow}$, ContinuousMap$^{\uparrow}$, LocatedApartnessRealBridge$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{BishopLocatedInverseFunction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, A, B, U, V, S, Q, D, R, H, C, P, N$. It has no coordinate for a host inverse function, quotient real equality, real trichotomy, selected representative, ambient Real$^{\uparrow}$ *completeness, or an unrecorded equality of functions.*

Lean4 (definition) `BEDC.Derived.BishopLocatedInverseFunctionUp.BishopLocatedInverseFunctionUp`
Lean4 ✓ `BEDC.Derived.BishopLocatedInverseFunctionUp.BishopLocatedInverseFunctionTasteGate`
`single carrier alignment`

Theorem 6001.2 (Bishop located inverse function NameCert obligations)**.** *For every accepted* BishopLocatedInverseFunction$^{\uparrow}$ *carrier*

$$L = (F, A, B, U, V, S, Q, D, R, H, C, P, N),$$

the local NameCert$_{\text{BishopLocatedInverseFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the* ContinuousMap$^{\uparrow}$ *row,* LocatedApartnessRealBridge$^{\uparrow}$ *row, interval row, uniqueness-window row, existence-window row,* StreamName$^{\uparrow}$ *windows,* RegSeqRat$^{\uparrow}$ *readback,* DyadicRatCore$^{\uparrow}$ *tolerance ledger,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *continuity exposure, requiring every inverse read to open the displayed* ContinuousMap$^{\uparrow}$ *modulus cells before endpoint-image or retained-window comparisons are consumed;*
3. *located apartness exposure, requiring the row A to be read before uniqueness of the source window is inspected;*
4. *existence-window discipline, requiring V, S, Q, D to display the refinement schedule, finite windows, regular rational readbacks, and dyadic tolerance budget before the terminal* Real$^{\uparrow}$ *seal is available;*
5. *uniqueness-window discipline, requiring U to use the same located apartness and dyadic comparison rows rather than a host equality of real values;*
6. *structural nonescape, requiring H, C, P, N to preserve only the displayed route and no host inverse function, quotient real equality, real trichotomy, selected representative, ambient Real completeness, or bridge credential.*

No Bishop located inverse-function obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6001.1. The accepted packet displays exactly the continuous-map row, located-apartness bridge row, interval row, uniqueness and existence windows, stream windows, regular rational readback, dyadic tolerance ledger, Real seal, and structural rows.

The local inverse read begins with continuity and location. The row F supplies the finite modulus cells used for endpoint-image and retained-window comparisons, while A supplies the located apartness data that separates candidate source windows.

The existence route is finite. The row V records the refinement schedule, S exposes the relevant finite windows, Q reads those windows as regular rational data, and D records the dyadic comparison budget before the terminal seal R is inspected.

The uniqueness route uses the same displayed data. The row U can compare two source windows only through the already opened located apartness and dyadic tolerance rows, so it cannot import trichotomy or host equality of completed reals.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate stores a host inverse function, quotient real equality, selected representative, ambient completeness principle, or bridge credential, the local NameCert surface is exhausted by the listed rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedInverseFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedInverseFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6001.1 and Definition 6001.2 seed the finite located inverse-function certificate over $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedInverseFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, quotient real equality, real trichotomy, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, continuity exposure, located apartness exposure, existence-window discipline, uniqueness-window discipline, structural nonescape, and local naming for BEDC.Derived.BishopLocatedInverseFun

Constructive story: A $\mathsf{BishopLocatedInverseFunction}^{\uparrow}$ packet is generated from a $\mathsf{ContinuousMap}^{\uparrow}$ row, a $\mathsf{LocatedApartnessRealBridge}^{\uparrow}$ row, interval rows, uniqueness and existence windows, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6002

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$

Bishop–Bridges style diagonal arguments turn a displayed family of regular Cauchy rational names into a limit-facing regular read by synchronizing finite windows and tolerance ledgers before any real seal is exposed. The $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ packet records that consumer route as one finite NameCert surface. It sits after the $\mathsf{StreamName}^{\uparrow}$ window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular-rational readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the regular-Cauchy diagonal sibling of chapter 1773. The packet names the L10 diagonal-limit consumer for regular rational sequences and real-name seals; it is not quotient stream equality, a host-selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6002.1 (Regular Cauchy diagonal limit carrier)**.** *A* $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (S, W, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the displayed source-family row of regular Cauchy rational names whose diagonal entries may be inspected.*
2. *W is the synchronized* $\mathsf{StreamName}^{\uparrow}$ *window family for the source row, diagonal selector reads, and limit-facing output read.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *diagonal readback row assembled from the accepted source windows.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, slack, and comparison ledger used to align the source windows with the diagonal readback.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after S, W, R, Q have been displayed.*
6. *H records componentwise* hsame *transport for the source family, synchronized windows, regular readback, dyadic ledger, real seal, continuation, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from each limit-facing precision request through S, W, Q, R, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalExtraction}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalWindow}^{\uparrow}$, $\mathsf{RegularCauchyDiagonalSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, R, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable-choice schedules, host limits, trichotomy, ambient completeness, Lean verification, or bridge credentials.

Lean4 ✓ `BEDC.Derived.RegularCauchyDiagonalLimitUp.RegularCauchyDiagonalLimitTasteGate single carrier alignment`

Theorem 6002.2 (Regular Cauchy diagonal limit NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ *carrier*

$$D = (S, W, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the source family, synchronized windows, regular readback, dyadic tolerance ledger, terminal real seal, transport, replay, provenance, and local naming rows;*
2. *window coverage, requiring every limit-facing request to read only windows listed in W and only source data displayed by S;*
3. *dyadic exactness, requiring Q to be the sole tolerance, slack, and comparison ledger used to align the diagonal readback;*
4. *RegSeqRat handoff, requiring R to be downstream of S, W, Q and upstream of the terminal real seal E;*
5. *Real seal non-escape, requiring E, H, C, P, N to expose no quotient stream equality, host-selected limit, countable choice, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Regular Cauchy diagonal-limit obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6002.1. The displayed tuple $S, W, R, Q, E, H, C, P, N$ is the entire public surface, so carrier admission is exactly admission of those rows.

The window and dyadic clauses are read next. A limit-facing request enters through the synchronized window family W, while Q supplies the tolerance, slack, and comparison cells attached to that window. The source-family row S is inspected only through those displayed windows.

The regular readback row R is therefore downstream of S, W, Q. Once R has been displayed, the seal row E may be read as a $\mathsf{Real}^{\uparrow}$ boundary, and the structural rows H, C, P, N transport, replay, source, and name the same route.

Because the carrier has no further coordinate, the excluded analytic and choice-theoretic objects cannot be exported by the local NameCert surface. □

Theorem 6002.3 (Regular Cauchy diagonal limit RegSeqRat handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ *carrier first reads the finite route*

$$S, W, Q \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the L10 diagonal route hands off a $\mathsf{RegSeqRat}^{\uparrow}$ *readback before the real seal is available, and the consumer receives no quotient stream equality, selected limit, countable-choice schedule, trichotomy row, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start from a limit-facing request. The carrier admits no direct read of the real seal, so the request must first pass through the source row S, the synchronized windows W, and the dyadic tolerance ledger Q.

These rows assemble the regular rational diagonal readback R. The readback is the only row that can feed the terminal $\mathsf{Real}^{\uparrow}$ seal E, so any consumer reaching the seal has already consumed the RegSeqRat handoff.

The rows H, C, P, N preserve this displayed route under transport, replay, provenance, and local naming. Since no excluded coordinate occurs in the carrier, the handoff exports only the finite diagonal-limit packet. □

Falsifiable prediction: Every $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ Real-facing read factors through a displayed source family, synchronized StreamName windows, one DyadicRatCore tolerance ledger, one RegSeqRat diagonal readback, one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, regular-cauchy-diagonal-extraction, regular-cauchy-diagonal-window, regular-cauchy-diagonal-seal: $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the limit-facing source family, synchronized windows, tolerance ledger, RegSeqRat handoff, Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}$ packet is generated from a regular Cauchy source family, synchronized StreamName windows, a DyadicRatCore tolerance ledger, a RegSeqRat diagonal readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6002.1, Definition 6002.2, and Definition 6002.3 seed the finite diagonal-limit route from regular Cauchy source windows through RegSeqRat readback to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window coverage, dyadic exactness, RegSeqRat handoff, Real seal non-escape, componentwise transport, displayed replay, and local naming for BEDC.Derived.RegularCauchyDiagonalLimitUp

Chapter 6003

Finite Cauchy Density Net NameCert Construction

Following Bishop and Bridges, finite density data for real completion is represented by explicit dyadic finite nets and moduli, not by selected representatives or quotient completions. The $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet sits between the dyadic arithmetic of chapter 291, the finite dyadic cell surface of chapter 5966, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. It gives L10 completion consumers a finite density witness object without asserting ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6003.1 (Finite Cauchy density-net packet)**.** *A* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (D, W, R, E, M, T, C, P, L)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *finite-net ledger, listing the dyadic centers, dyadic radii, and finite containment comparisons used by the density read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row that records exactly which prefixes of an inspected Cauchy name are opened before the net row is consulted.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and dyadic ledger.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after D, W, R have all been displayed.*
5. *M is the modulus row, recording the finite precision budget that makes the displayed dyadic net adequate for the requested readback.*
6. *T records componentwise* hsame *transport for dyadic net rows, stream windows, regular readback rows, Real seals, modulus rows, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a density-net request through D, W, R, E, M, T.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *L is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, a selected representative of a completed real, countable choice over nets, host real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6003.2 (Finite Cauchy density-net obligation carrier)**.** *Every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *density read must expose dyadic tolerance rows, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the terminal* $\mathsf{Real}^{\uparrow}$ *seal,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}$ *before it can serve a density-net consumer. In carrier notation the admissible route is the displayed packet*

$$D, W, R, E, M, T, C, P, L$$

of Definition 6003.1; no consumer may replace it by quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier rows in Definition 6003.1. The dyadic tolerance data are contained in D and M, the finite stream inspection is contained in W, the regular readback is R, and the terminal Real-facing row is E.

Use the finite density-net route already displayed by the carrier: the density read first opens the dyadic ledger and finite stream window, then reads back through $\mathsf{RegSeqRat}^{\uparrow}$, and only then exposes the $\mathsf{Real}^{\uparrow}$ seal. The structural rows T, C, P, L keep the same route under transport, replay, provenance, and local naming.

The sibling routes input by the hub supply the same L10 discipline for finite density and regular readback consumers, so the carrier cannot be consumed as an unrecorded density assertion. Since the packet has no row for quotient streams, selected limits, countable choice, or ambient Real completeness, the displayed rows exhaust the obligation surface. □

Theorem 6003.3 (Finite Cauchy density-net extraction)**.** *For every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *carrier* $N = (D, W, R, E, M, T, C, P, L)$*, every density-net read extracts exactly the displayed finite dyadic net rows* D*, the finite* $\mathsf{StreamName}^{\uparrow}$ *observation row* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal route* E*, and the modulus row* M*, preserved by* T, C, P, L*. The extraction supplies a finite L10 density witness for Real-facing completion routes, but it does not claim* $\mathsf{Real}^{\uparrow}$ *completeness, quotient representatives, selected limits, countable choice, trichotomy, or host real equality.*

Proof. Project the carrier of Definition 6003.1. The dyadic net row D is the only row carrying centers, radii, and finite containment comparisons, while W is the only row opening inspected stream prefixes.

The readback row R is downstream of D, W and the modulus row M. Thus any density read reaches the regular rational surface through displayed finite data rather than through a detached sequence or host completion object.

The terminal row E seals only that readback route as a $\mathsf{Real}^{\uparrow}$ handoff. Rows T, C, P, L transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for quotient streams, selected representatives, countable choice, trichotomy, host real equality, or ambient Real completeness, the listed rows exhaust the extraction surface. □

Theorem 6003.4 (Finite Cauchy density-net Real seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the displayed carrier rows $D, W, R, E, M, T, C, P, L$*. The packet can be cited as a finite density witness by separability, completion, and compactness consumers, but it exports no completed-real theorem and no selector for representatives or limits.*

Proof. Apply Definition 6003.3. The extraction theorem identifies the dyadic net ledger, finite stream window, regular readback, Real seal, and modulus rows as the whole analytic route.

The structural rows T, C, P, L preserve that route under componentwise transport, continuation replay, provenance, and local naming. They do not add a second completion-facing row.

Therefore a Real-facing consumer receives only the displayed finite density packet. Any claim of completed Real separability, ambient completeness, quotient representatives, selected limits, or countable-choice net selection lies outside the local carrier. □

Theorem 6003.5 (Finite Cauchy density-net stream route)**.** *Every finite density-net consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet must first expose the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row of chapter 126 and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row of chapter 301 before treating the dyadic net as a* $\mathsf{Real}^{\uparrow}$*-facing density witness. Equivalently, the dyadic rows D become Real-facing only along the displayed carrier route $D, W, R, E, M, T, C, P, L$.*

Proof. Project the carrier of Definition 6003.1. The rows $D, W, R, E, M, T, C, P, L$ are the whole displayed packet, with W the finite window row and R the regular rational readback row.

Apply Definition 6003.3. It identifies the dyadic net ledger D, the $\mathsf{StreamName}^{\uparrow}$ window W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, the terminal $\mathsf{Real}^{\uparrow}$ seal E, and the modulus row M as the extracted analytic route.

Apply Definition 6003.4. A Real-facing consumer can read a seal only after the same rows have been displayed. The structural rows T, C, P, L transport, replay, source, and name that route, so the dyadic density net cannot be consumed as a Real-facing witness before W and R have been exposed. □

Theorem 6003.6 (Finite Cauchy density-net sibling route)**.** *Every completion-facing sibling consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet, including the real-uniform route of chapter 1624 and the modulus route of chapter 292, must consume the finite density-net read only after the displayed rows $D, W, R, E, M, T, C, P, L$ have all been exposed. The read may first consume the finite dyadic cell surface of chapter 5966:* $\mathsf{DyadicIntervalNet}^{\uparrow}$ *supplies the displayed cells and endpoint incidence, while* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *supplies the density witness and Real-facing seal. In particular, the packet supplies a completion-facing dependency edge through dyadic cells, endpoint incidence, stream windows, regular rational readback, Real sealing, finite modulus, transport, replay, provenance, and local naming, not through a kernel-only density assertion.*

Proof. Start with Definition 6003.5. It forces every density-net consumer to expose the $\mathsf{StreamName}^{\uparrow}$ row W and $\mathsf{RegSeqRat}^{\uparrow}$ row R before the dyadic net can be read as Real-facing data.

Before that Real-facing export, the consumer may read the sibling $\mathsf{DyadicIntervalNet}^{\uparrow}$ surface of chapter 5966. That sibling supplies the displayed dyadic cells and endpoint incidence; the density-net packet keeps those cells tied to D and then supplies the density witness, modulus row M, and terminal seal E.

Apply Definition 6003.4. The terminal $\mathsf{Real}^{\uparrow}$ seal E is available only along the same carrier route, so a $\mathsf{RealUniformStructure}^{\uparrow}$ or $\mathsf{CauchyModulus}^{\uparrow}$ consumer cannot bypass $D, W, R, E, M, T, C, P, L$ when it asks for completion-facing density.

Finally use Definition 6003.3. The extraction theorem keeps the dyadic ledger D, modulus row M, structural transport T, replay C, provenance P, and local naming L attached to the read. Thus the sibling dependency edge is a finite displayed route into later completion consumers, not an ambient Real-completion fact or an unrecorded selector for dense representatives. □

Theorem 6003.7 (Finite Cauchy density-net finite-window comparison)**.** *Every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *comparison exported to a* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer factors through the displayed finite net extraction,* $\mathsf{StreamName}^{\uparrow}$ *window, dyadic tolerance row, and* $\mathsf{Real}^{\uparrow}$ *seal row of* $N = (D, W, R, E, M, T, C, P, L)$*. The sibling route fixes the same* BHist *provenance, so the comparison uses one displayed window route rather than host limit selection, quotient representatives, or ambient completeness.*

Proof. First apply finite density-net extraction, Definition 6003.3. It obtains the finite dyadic net row D, the finite stream observation row W, the regular readback R, the Real seal E, and the modulus row M from the accepted carrier.

Next use the stream route of Definition 6003.5. It permits the comparison to read only the listed StreamName windows and the RegSeqRat readback attached to those windows.

Finally apply the sibling route and Real seal nonescape, Definition 6003.6 and Definition 6003.4. They keep the same BHist provenance and terminal Real seal attached to $D, W, R, E, M, T, C, P, L$, excluding host limit selection, quotient representatives, and ambient completeness. □

Theorem 6003.8 (Finite Cauchy density-net tail extraction)**.** *For every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet* $N = (D, W, R, E, M, T, C, P, L)$*, every completion-facing tail read consumed by* $\mathsf{RegularSequenceSpace}^{\uparrow}$*,* $\mathsf{RealUniformStructure}^{\uparrow}$*, or* $\mathsf{CauchyModulus}^{\uparrow}$ *exposes the same finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the same dyadic net ledger* D*, the same finite modulus row* M*, the same terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, and the same structural rows* T, C, P, L*. The tail read is therefore a displayed finite BEDC route, not a quotient stream, not a chosen representative, and not an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 6003.5. It forces every completion-facing read to pass through the displayed $\mathsf{StreamName}^{\uparrow}$ window W and the $\mathsf{RegSeqRat}^{\uparrow}$ row R before any tail data can be consumed.

Then apply Definition 6003.6. That theorem keeps the same displayed route attached to completion-facing readers such as $\mathsf{RealUniformStructure}^{\uparrow}$ and $\mathsf{CauchyModulus}^{\uparrow}$, so the tail read cannot bypass $D, W, R, E, M, T, C, P, L$.

Finally use Definition 6003.7. The comparison theorem factors the completion-facing read through the same finite packet data, and Definition 6003.1 lists exactly the rows $D, W, R, E, M, T, C, P, L$ with no coordinate for quotient streams, chosen representatives, or ambient completeness. Row exhaustion gives the claim. □

Theorem 6003.9 (Finite Cauchy density-net regular-sequence handoff)**.** *Every handoff from an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet* $N = (D, W, R, E, M, T, C, P, L)$ *to* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *exposes the finite dyadic net rows* D*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E*, the finite modulus row* M*, and the structural rows* T, C, P, L *before any regular-sequence consumer reads it. The handoff is therefore finite BEDC evidence only: it does not export a quotient stream, a selected representative, a completed Real limit, countable choice, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with finite density-net extraction, Definition 6003.3. It displays the dyadic net ledger D, the stream window W, the regular rational readback R, the Real seal E, and the modulus row M from the accepted packet.

Use the stream route of Definition 6003.5. A regular-sequence consumer can read the handoff only after the finite $\mathsf{StreamName}^{\uparrow}$ window and the $\mathsf{RegSeqRat}^{\uparrow}$ readback have been exposed.

Now apply tail extraction, Definition 6003.8. It keeps the same rows visible for completion-facing tail reads, including the structural rows T, C, P, L.

Finally use the sibling route of Definition 6003.6. The handoff to $\mathsf{RegularSequenceSpace}^{\uparrow}$ remains a displayed route through $D, W, R, E, M, T, C, P, L$, so no quotient stream, selected representative, completed limit, choice schedule, trichotomy, host equality, or ambient completeness is introduced. □

Theorem 6003.10 (Finite Cauchy density-net diagonal tightening)**.** *For every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet* $N = (D, W, R, E, M, T, C, P, L)$*, any two completion-facing tail reads of* N *can be tightened to a shared finite* $\mathsf{StreamName}^{\uparrow}$ *window row and a shared dyadic tolerance row before either* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or* $\mathsf{Real}^{\uparrow}$ *sealing is consumed. The tightening remains inside the displayed route* $D, W, R, E, M, T, C, P, L$ *and exports no quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with finite-window comparison, Definition 6003.7. It forces each completion-facing comparison to use the displayed finite net extraction, the $\mathsf{StreamName}^{\uparrow}$ window, the dyadic tolerance row, and the terminal seal route attached to the same BHist provenance.

Apply tail extraction from Definition 6003.8. Both tail reads must expose W, R, D, M, E, and the structural rows T, C, P, L before any completion-facing consumer can use them.

Now use the sibling route of Definition 6003.6. A $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer receives only this displayed density-net route, so the two reads meet at one finite stream window and one dyadic tolerance row before readback or sealing.

Since the carrier has no coordinate for quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient Real completeness, the shared window and tolerance row are obtained by row tightening inside the accepted packet rather than by an external selector. □

Theorem 6003.11 (Finite Cauchy density-net refinement)**.** *Let* $N = (D, W, R, E, M, T, C, P, L)$ *be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet. Refining the dyadic cell surface of* N *preserves the density witness* D*, the finite modulus row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *window* W*, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal route* E, T, C, P, L*. The refined packet consumes only finite dyadic cells and exports no ambient completeness, quotient representatives, selected limits, countable choice, trichotomy, or host real equality.*

Proof. Start with the sibling route of Definition 6003.6. It lets the refinement read the $\mathsf{DyadicIntervalNet}^{\uparrow}$ cell surface before any Real-facing export, while keeping the density-net rows $D, W, R, E, M, T, C, P, L$ attached to the same finite packet.

Apply the diagonal tightening theorem, Definition 6003.10. Any two refined completion-facing reads meet at one displayed $\mathsf{StreamName}^{\uparrow}$ window and one dyadic tolerance row before regular rational readback or Real sealing is consumed.

Now use tail extraction and finite-window comparison, Definition 6003.8 and Definition 6003.7. These rows preserve the modulus, stream window, RegSeqRat readback, and terminal Real seal through the same displayed carrier route.

Finally replay T, C, P, L. Componentwise transport, continuation replay, provenance, and local naming preserve the refined finite-cell route and add no coordinate for ambient completeness, quotient representatives, selected limits, countable choice, trichotomy, or host real equality. □

Theorem 6003.12 (Finite Cauchy density-net regular readback)**.** *Let* $N = (D, W, R, E, M, T, C, P, L)$ *be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet. Every* $\mathsf{RegSeqRat}^{\uparrow}$*-facing consumer of* N *must read the same finite* $\mathsf{StreamName}^{\uparrow}$ *window* W*, dyadic net ledger* D*, finite modulus row* M*, and readback row* R *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is available. The readback is therefore the displayed route*

through $D, W, R, E, M, T, C, P, L$ and exports no quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with finite density-net extraction, Definition 6003.3. It displays the dyadic net ledger D, stream window W, regular readback R, Real seal E, and modulus row M as the analytic rows of the accepted packet.

Apply the stream route and tail extraction, Definition 6003.5 and Definition 6003.8. They force every regular-facing and completion-facing read through the same W, R, D, M, E route before the terminal seal can be consumed.

Now use diagonal tightening and refinement, Definition 6003.10 and Definition 6003.11. The refined readback still meets at one displayed finite window and dyadic ledger before $\mathsf{RegSeqRat}^{\uparrow}$ readback or $\mathsf{Real}^{\uparrow}$ sealing is public.

Finally replay T, C, P, L. Componentwise transport, continuation replay, provenance, and local naming preserve the same finite readback route, so no coordinate remains for quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient Real completeness. □

Definition 6003.13 (Finite Cauchy density-net cofinal window)**.** *Let $N = (D, W, R, E, M, T, C, P, L)$ be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet. Its cofinal-window density subpacket is the displayed tuple*

$$N_{\mathrm{cof}} = (D, W, R, E, M, T, C, P, L)$$

read with the following completion-facing order: D is the finite $\mathsf{DyadicRatCore}^{\uparrow}$ *density ledger, W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window through which the tail request is opened, M is the finite precision budget attached to that window, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback obtained from D, W, M, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after R, and T, C, P, L transport, replay, source, and locally name that same finite route. The subpacket has no coordinate for a quotient stream, a chosen limit, an ambient complete real, trichotomy, host equality, or a selector over hidden tail representatives.*

Theorem 6003.14 (Finite Cauchy density-net cofinal window)**.** *Let $N = (D, W, R, E, M, T, C, P, L)$ be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet. Any two completion-facing tail reads of N that are later consumed by* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *refine to the displayed cofinal-window subpacket N_{cof}: they meet at one finite* $\mathsf{StreamName}^{\uparrow}$ *window W and one dyadic tolerance row controlled by D and M before either* $\mathsf{RegSeqRat}^{\uparrow}$ *readback or terminal* $\mathsf{Real}^{\uparrow}$ *sealing is available. The refinement is internal to $D, W, R, E, M, T, C, P, L$ and exports no quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply finite-window comparison, Definition 6003.7. Each completion-facing comparison must pass through the displayed finite net extraction, the $\mathsf{StreamName}^{\uparrow}$ window, the dyadic tolerance row, and the terminal seal route attached to the same BHist provenance.

Use tail extraction and diagonal tightening, Definition 6003.8 and Definition 6003.10. The two tail reads therefore expose the same W, R, D, M, E rows before any regular rational readback or Real sealing is consumed, and the tightening occurs inside the accepted carrier.

Finally apply refinement and regular readback, Definition 6003.11 and Definition 6003.12. Refining the dyadic cell surface preserves $D, W, R, E, M, T, C, P, L$, while the regular readback theorem forces every $\mathsf{RegSeqRat}^{\uparrow}$-facing consumer to read that same finite route first. Hence both reads factor through N_{cof} with no external selector, quotient representative, chosen limit, trichotomy, host equality, or ambient completeness. □

Theorem 6003.15 (Finite Cauchy density-net RegSeqRat density handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet receives density evidence only*

through the cofinal-window subpacket of Definition 6003.13. In particular, the consumer may use the displayed finite dyadic ledger D, finite $\mathsf{StreamName}^{\uparrow}$ *window W, finite precision budget M,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, and structural rows T, C, P, L, but it receives no quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient Real-completeness principle.*

Proof. Start with the cofinal-window theorem, Definition 6003.14. It refines any two completion-facing tail reads to one displayed $\mathsf{StreamName}^{\uparrow}$ window and one dyadic tolerance row before regular readback or Real sealing.

Apply regular readback, Definition 6003.12. A $\mathsf{RegSeqRat}^{\uparrow}$-facing consumer cannot reach R without first reading the same finite W, D, M route, and the terminal $\mathsf{Real}^{\uparrow}$ seal E remains downstream of that readback.

Use Real seal nonescape, Definition 6003.4. The $\mathsf{Real}^{\uparrow}$ consumer receives only the displayed finite density packet; the structural rows T, C, P, L transport, replay, source, and name that packet without creating a second completion coordinate.

The carrier exhaustion clause in Definition 6003.13 removes the remaining alternatives. Since N_{cof} has no coordinate for quotient representatives, selected limits, countable choice, trichotomy, host equality, or ambient Real completeness, every accepted density handoff to $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ is mediated by the displayed cofinal-window packet. □

Theorem 6003.16 (Finite Cauchy density-net regular-sequence forward link)**.** *Every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet is consumed by the regular sequence-space route of chapter 5965 and the regular rational readback route of chapter 301 only after the displayed finite net row D, dyadic tolerance and modulus rows D, M, finite* $\mathsf{StreamName}^{\uparrow}$ *window W, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R have all been exposed inside $N = (D, W, R, E, M, T, C, P, L)$. The forward link is therefore a finite row dependency from density nets to* $\mathsf{RegularSequenceSpace}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, not a host-density theorem, quotient stream equality, selected representative, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the regular-sequence handoff of Definition 6003.9. It requires the finite dyadic net, stream window, regular readback, Real seal, finite modulus, and structural rows to be visible before a $\mathsf{RegularSequenceSpace}^{\uparrow}$ consumer reads the packet.

Apply the RegSeqRat density handoff of Definition 6003.15. It keeps every $\mathsf{RegSeqRat}^{\uparrow}$-facing read inside the cofinal-window subpacket, so the readback row R is reached only through the displayed D, W, M surface.

Finally use the regular readback theorem, Definition 6003.12. It ties the same finite window and dyadic ledger to the terminal seal route E, T, C, P, L, excluding quotient streams, selected representatives, host density assertions, and ambient Real completeness from the forward dependency. □

Theorem 6003.17 (Finite Cauchy density-net regular-approximation handoff)**.** *Every* $\mathsf{RealApproximationScheme}^{\uparrow}$ *or* $\mathsf{RealApproximationEnvelope}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet must export its real-approximation handoff only after reading the displayed dyadic ledger D, finite* $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, modulus row M, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}$ *row L of Definition 6003.1. The handoff is therefore a finite BEDC route into real-approximation consumers, not a quotient representative, selected limit, countable-choice schedule, host density theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the carrier rows in Definition 6003.1. The only analytic rows available to a real-approximation consumer are the dyadic ledger D, finite stream window W, regular readback R, terminal seal E, and modulus row M, while T, C, P, L preserve the same route under transport, replay, provenance, and local naming.

Apply the RegSeqRat density handoff of Definition 6003.15. It forces every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ density read through the displayed cofinal-window subpacket before a downstream approximation surface can consume it.

Now use the Real seal nonescape theorem, Definition 6003.4. The seal row E exports only the finite route already displayed by D, W, R, M, T, C, P, L, so a $\mathsf{RealApproximationScheme}^{\uparrow}$ or $\mathsf{RealApproximationEnvelope}^{\uparrow}$ consumer cannot replace the route by a hidden completion object.

Since the carrier has no coordinate for quotient representatives, selected limits, countable choice, host density assertions, or ambient $\mathsf{Real}^{\uparrow}$ completeness, row exhaustion gives the claimed real-approximation handoff boundary. □

Falsifiable prediction: Every accepted $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ read exposes a DyadicRatCore finite-net ledger, a StreamName finite-window row, a RegSeqRat readback row, a terminal RealUp seal, a finite modulus row, componentwise transport, continuation replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ is not identical to any listed sibling because it binds dyadic finite-net rows and a modulus row before the regular rational readback and terminal RealUp seal.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet is generated from a DyadicRatCore finite-net ledger, a StreamName finite-window row, a RegSeqRat readback row, a RealUp seal, a finite modulus row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6003.1, Definition 6003.3, and Definition 6003.4 seed the finite dyadic density-net route from displayed Cauchy windows to the terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim RealUp completeness, completed-real separability, compactness, quotient representatives, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic finite-net exactness, StreamName window admission, RegSeqRat readback, Real seal nonescape, finite modulus stability, replay, provenance, and local naming for BEDC.Derived.FiniteCauchyDensityNetUp.

Theorem 6003.18 (Finite Cauchy density-net Bishop Cauchy route)**.** *Every Bishop Cauchy, regular-sequence, or completion-facing consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet*

$$N = (D, W, R, E, M, T, C, P, L)$$

reads the finite density witness only through the dyadic arithmetic source D, finite $\mathsf{StreamName}^{\uparrow}$ *window W,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, terminal* $\mathsf{Real}^{\uparrow}$ *seal E, modulus row M, componentwise* hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyDensityNet}^{\uparrow}}$ *row L. In particular, the handoff to the Bishop Cauchy route of chapter 5965 is a finite L10 sibling dependency and not a quotient-completion argument, selected representative, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with finite density-net extraction, Definition 6003.3. It identifies $D, W, R, E, M, T, C, P, L$ as the displayed density-net route and leaves no hidden completion row.

Use the regular-sequence handoff theorem, Definition 6003.9. A regular-sequence consumer reaches the density witness only after the $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, dyadic net ledger, modulus row, and terminal $\mathsf{Real}^{\uparrow}$ seal have been displayed.

Now compare with the regular sequence-space chapter named by chapter 5965. Its Bishop Cauchy consumer can read the density-net packet only as this already displayed finite handoff, so the dependency edge is the same L10 route through $D, W, R, E, M, T, C, P, L$.

Finally apply Real seal nonescape, Definition 6003.4. Since the accepted packet has no coordinate for quotient completion, selected representatives, host equality, countable choice, or ambient Real completeness, no Bishop Cauchy consumer can bypass the finite density-net route. □

Theorem 6003.19 (Finite Cauchy density-net L10 sibling route)**.** *Every completion-facing consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet must read the dyadic density-net row through chapter 291, the finite observation window through chapter 126, the regular rational readback through chapter 301, and the terminal seal through chapter 38 before the density witness reaches any* $\mathsf{Real}^{\uparrow}$*-completion consumer.*

Equivalently, the L10 sibling dependency route is the displayed carrier path $D, W, R, E, M, T, C, P, L$ *of Definition 6003.1, not a direct assertion of density inside ambient completed reals, quotient stream equality, selected representatives, or host real equality.*

Proof. Start with Definition 6003.3. It identifies the dyadic ledger D, finite stream window W, regular rational readback R, terminal Real seal E, and modulus row M as the extracted density surface.

Next apply Definition 6003.5. The route forces the $\mathsf{StreamName}^{\uparrow}$ window and $\mathsf{RegSeqRat}^{\uparrow}$ readback to stand between the dyadic net and any Real-facing interpretation, so the sibling chapters are read in the displayed order.

Finally use Definition 6003.4. The terminal seal is available only after $D, W, R, E, M, T, C, P, L$ have been exposed, and the structural rows keep transport, replay, provenance, and local naming attached to the same finite witness. Thus the completion-facing consumer receives an L10 density route through the named sibling chapters rather than an unrecorded ambient completion principle. □

Theorem 6003.20 (Finite Cauchy density-net regular-sequence consumer route)**.** *Every regular-sequence consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet reads the density witness only after the route through chapter 5965 and the finite-window extraction route of chapter 7135 have exposed the same* $\mathsf{StreamName}^{\uparrow}$ *window,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and terminal* $\mathsf{Real}^{\uparrow}$ *seal. Thus the finite density-net row is a downstream L10 dependency edge for regular Cauchy tails, not an ambient density assertion inside completed reals.*

Proof. Start with Definition 6003.19. It forces every completion-facing density read to pass through the displayed dyadic ledger, stream window, regular rational readback, and terminal Real seal.

The regular-sequence-space consumer contributes the ambient L10 sequence carrier, while the regular Cauchy window extraction consumer contributes the inspected finite tail window. Both consumers therefore meet the density-net packet at the already displayed route $D, W, R, E, M, T, C, P, L$ of Definition 6003.1.

Apply Definition 6003.4. Since the packet exports no quotient representatives, selected limits, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the dependency edge remains the finite L10 route through the named regular-sequence and window-extraction chapters. □

Theorem 6003.21 (Finite Cauchy density-net public density export)**.** *Every* $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegularSequenceSpace}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet receives the public density package exactly through the finite rows*

$$D, W, R, E, M, T, C, P, L.$$

The export is the displayed dyadic density ledger D, finite StreamName window W, RegSeqRat readback R, terminal Real seal E, finite modulus row M, componentwise hsame *transport T, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row L. The downstream sibling consumers are chapter 1624, chapter 292, chapter 5965, and chapter 38. It does not assert completed-real separability, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient representatives, selected limits, countable choice, trichotomy, or host real equality.*

Proof. First apply finite Cauchy density-net extraction, Definition 6003.3. It exposes $D, W, R, E, M, T, C, P, L$ as the finite density witness extracted from the accepted carrier.

Next use the stream route and sibling route, Definition 6003.5 and Definition 6003.6. These results place $\mathsf{RealUniformStructure}^{\uparrow}$ and $\mathsf{CauchyModulus}^{\uparrow}$ consumers downstream of the same StreamName window, RegSeqRat readback, dyadic ledger, Real seal, modulus, transport, replay, provenance, and local naming rows.

Now apply tail extraction, Definition 6003.8. It identifies the completion-facing reads used by $\mathsf{RegularSequenceSpace}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, and $\mathsf{CauchyModulus}^{\uparrow}$ with the same finite packet rows.

Finally use refinement, Definition 6003.11. Refining the dyadic cell surface preserves the displayed density route and adds no coordinate for completed-real separability, ambient completeness, quotient representatives, selected limits, countable choice, trichotomy, or host real equality. Thus the public density package exported to the listed consumers is exactly $D, W, R, E, M, T, C, P, L$. □

The $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet supplies the finite witness surface consumed by chapter 1809: its carrier exposes explicit finite dyadic representatives, stream windows, regular-Cauchy readback, and a terminal Real seal before separability reads density. This makes the density handoff a displayed finite BEDC route rather than a host-level countability assumption or a selected dense subset. The $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet supplies the finite density row consumed by chapter 1809 only after the dyadic, StreamName, RegSeqRat, and Real sibling route exposed by chapter 291, chapter 126, chapter 301, and chapter 38. Every $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ density read that serves a $\mathsf{Real}^{\uparrow}$ consumer first consumes the finite mesh rows of $\mathsf{DyadicIntervalNet}^{\uparrow}$ from chapter 5966 and the regular readback rows of $\mathsf{RegSeqRat}^{\uparrow}$ from chapter 301. The route exposes dyadic cells, endpoint incidence, finite stream windows, regular rational readback, terminal real sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert data before any downstream density consumer reads the packet; it exports no ambient completeness theorem or selector for dense representatives. The $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet supplies the visible dyadic net rows consumed by the bounded Cauchy subsequence-modulus route of chapter 9185. A downstream extractor first reads the finite density ledger, $\mathsf{StreamName}^{\uparrow}$ windows, and $\mathsf{RegSeqRat}^{\uparrow}$ readback already exposed by chapter 301, then carries those rows to the terminal $\mathsf{Real}^{\uparrow}$ consumer of chapter 38. The handoff is a finite row dependency from density nets into subsequence-modulus extraction, not a hidden sequential compactness assertion or a choice-selected subsequence.

Theorem 6003.22 (Finite Cauchy density-net sibling dependency)**.** *Every accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *carrier consumes its density row through the concrete L10 sibling chain*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before a completion-facing consumer may read the exported density-net data. The dyadic tolerance ledger is supplied by chapter 291, the inspected finite window by chapter 126, the regular rational readback by chapter 301, and the terminal Real seal by chapter 38. In carrier notation, the dependency route is the displayed path

$$D, W, R, E, M, T, C, P, L$$

of Definition 6003.1, where D records the dyadic cell and tolerance ledger, W records the inspected StreamName window, R records the RegSeqRat readback, and E records the terminal Real seal. The route therefore records a concrete sibling dependency edge through dyadic tolerance, finite stream windows, regular readback, and Real sealing, not a kernel-only density assertion, quotient representative, selected limit, countable-choice net, or ambient completeness theorem.

Proof. Project the carrier of Definition 6003.1. Its analytic rows are exactly the dyadic ledger D, stream window W, regular readback R, terminal seal E, and modulus row M, with T, C, P, L preserving that same packet.

Apply Definition 6003.19. It already forces every completion-facing density read through the dyadic, StreamName, RegSeqRat, and Real sibling chapters before the read is exported.

Now refine the dyadic step through chapter 291 and the finite cell surface of chapter 5966. The dyadic core supplies the tolerance arithmetic read by D, and the interval net supplies the displayed cells and incidence rows.

The finite stream and readback steps are then forced by chapter 126 and chapter 301. The $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ packet can export density only after W and R are displayed, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38 remains downstream of those rows.

Finally apply Definition 6003.4. Since the carrier has no coordinate for quotient representatives, selected limits, countable-choice nets, host equality, or ambient completeness, the only available dependency route is the displayed sibling route through $D, W, R, E, M, T, C, P, L$. □

Theorem 6003.23 (Finite Cauchy density-net regular readback scope)**.** *Let $N = (D, W, R, E, M, T, C, P, L)$ be an accepted* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *packet. Every density-net readback consumed by a* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *route passes through the finite density net D, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *window W, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R, the* $\mathsf{Real}^{\uparrow}$ *seal E,* hsame *transport T,* Cont *replay C,* Pkg *provenance P, and the local* $\mathsf{NameCert}$ *row L. The scope exports no quotient representative, selected limit, countable-choice net, trichotomy, host equality, or ambient completeness principle.*

Proof. Project the carrier rows $D, W, R, E, M, T, C, P, L$. The finite density ledger D, tolerance budget M, stream window W, regular readback R, and Real seal E are the only analytic rows available to a density-net consumer.

Apply the regular readback theorem, Definition 6003.12. It forces every $\mathsf{RegSeqRat}^{\uparrow}$-facing consumer to inspect the same finite window, dyadic ledger, finite modulus, and readback row before the terminal $\mathsf{Real}^{\uparrow}$ seal is available.

Use the sibling dependency route, Definition 6003.22. The dyadic tolerance read is supplied through $\mathsf{DyadicRatCore}^{\uparrow}$, the inspected finite window through $\mathsf{StreamName}^{\uparrow}$, the regular rational face through $\mathsf{RegSeqRat}^{\uparrow}$, and the terminal seal through $\mathsf{Real}^{\uparrow}$.

Finally replay the structural rows. The hsame transport T, Cont replay C, Pkg provenance P, and local $\mathsf{NameCert}$ row L preserve the same displayed route, and the accepted carrier has no coordinate for quotient representatives, selected limits, countable-choice nets, trichotomy, host equality, or ambient completeness. □

Every $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ consumer must read its density witness through the dyadic, regular-readback, and real-seal sibling route. The mesh cells are supplied by the $\mathsf{DyadicRatCore}^{\uparrow}$ rows of chapter 291; the regular rational inspection is supplied by the $\mathsf{RegSeqRat}^{\uparrow}$ rows of chapter 301; and the terminal endpoint is sealed only through the $\mathsf{Real}^{\uparrow}$ rows of chapter 38. Thus a finite density read reaches completion-facing consumers as displayed dyadic mesh data, regular readback data, and RealUp seal data, not as an ambient density theorem or a selected dense subset.

Chapter 6004

A Concrete Naming Certificate for $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$

The $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ packet realizes the local-rule observation branch of chapter 9390 as finite Rule 110 NameCert data. It specializes the local-stencil surface of chapter 1036 to the eight-entry Rule 110 transition table, sits beside the cellular automaton carrier of chapter 934, the cellular row-classifier surface of chapter 941, and the Rule 110 pattern classifier of chapter 5844. The packet names one finite neighborhood read and its observed next-cell row; it is not an infinite orbit trace, a glider classification theorem, a universality proof, or a host evaluator semantics.

Definition 6004.1 (Rule 110 local-stencil observation carrier)**.** *A* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (W, T, S, O, K, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite window row, naming the inspected left, center, and right cells together with the row coordinate at which the stencil is read.*
2. *T is the Rule 110 transition-table row, exposing the eight binary neighborhood cases and the selected case for W.*
3. *S is the local-stencil row inherited from* $\mathsf{LocalStencilObservation}^{\uparrow}$*, binding W and T as one displayed observation.*
4. *O is the observed next-cell row, recording the output cell certified by the selected Rule 110 case.*
5. *K is the classifier-adjacency row, exposing how the observed local output may be consumed by* $\mathsf{CellularRowClassifier}^{\uparrow}$ *or* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *only after W, T, S, O have been displayed.*
6. *H records componentwise* hsame *transport for window, table, stencil, observed output, classifier-adjacency, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a local Rule 110 observation request through W, T, S, O, K, H.*
8. *P is the* Pkg *provenance over* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$*,* $\mathsf{LocalStencilObservation}^{\uparrow}$*,* $\mathsf{CellularAutomaton}^{\uparrow}$*,* $\mathsf{CellularRowClassifier}^{\uparrow}$*,* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, T, S, O, K, H, C, P, N$. It has no coordinate for an unbounded orbit, global glider catalog, universality witness, hidden evaluator, quotient row equality, or a completed substrate semantics.

Theorem 6004.2 (Rule 110 local-stencil observation NameCert obligations)**.** *For every accepted* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ *carrier*

$$L = (W, T, S, O, K, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}}$ *surface consists exactly of finite window admission through W, Rule 110 transition-table exposure through T, local-stencil binding through S, observed next-cell exposure through O, classifier-adjacency routing through K, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6004.1. The finite observation rows are exactly the window W, transition-table row T, local-stencil binding S, observed output O, classifier-adjacency row K, and the structural rows H, C, P, N.

The transition-table read is local. Row T selects one of the eight displayed Rule 110 cases for the three-cell window W, and row S records that this read is a local-stencil observation rather than a whole-orbit computation.

The observed cell O is then the only output exposed to classifier consumers. The row K lets a cellular row classifier or the Rule 110 pattern classifier consume that output only after the local window, table, stencil, and output rows are visible.

Finally H, C, P, N preserve and name the same route. Since the excluded objects are absent from the carrier, no local NameCert obligation can be discharged by an unbounded orbit, glider catalog, universality witness, hidden evaluator, quotient row equality, or completed substrate semantics. □

Theorem 6004.3 (Rule 110 local-stencil classifier route)**.** *Every classifier-facing read of an accepted* $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ *packet factors through*

$$\mathsf{LocalStencilObservation}^{\uparrow},\ \mathsf{CellularAutomaton}^{\uparrow},\ \mathsf{CellularRowClassifier}^{\uparrow},\ \mathsf{RuleOneTenPatternClassifier}^{\uparrow}$$

by first exposing the finite local route

$$W, T, S, O \longmapsto K \longmapsto H, C, P, N.$$

The route exports one observed Rule 110 transition-table cell to downstream classifiers; it does not export an orbit prefix, global pattern catalog, glider classification, universality theorem, or host evaluator semantics.

Proof. Start with the local rows W, T, S, O. The finite window and selected transition-table case determine the observed next-cell row before any classifier-facing consumer can read the packet.

Apply Definition 6004.2. It keeps classifier adjacency in K downstream of the local window, table, stencil, and output rows, so the classifier receives exactly the displayed local observation.

The sibling chapters named in the statement supply the surrounding surfaces. $\mathsf{LocalStencilObservation}^{\uparrow}$ supplies the generic local-stencil discipline, $\mathsf{CellularAutomaton}^{\uparrow}$ supplies the whole-rule provenance, $\mathsf{CellularRowClassifier}^{\uparrow}$ supplies the row-classifier consumer, and $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ supplies the Rule 110 pattern consumer.

None of those siblings is an internal coordinate of this packet before K, H, C, P, N are displayed. The exported route therefore remains a finite local-stencil observation rather than a global cellular or universality claim. □

Falsifiable prediction: A purported $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ read is outside the packet if it classifies a Rule 110 row without first exposing the finite three-cell window, the selected transition-table case, the local-stencil binding, the observed next-cell row, classifier adjacency, transport, replay, provenance, and local naming.

Independence witness: local-stencil-observation, cellularautomaton, cellularrowclassifier, rule110-pattern-classifier: $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds the Rule 110 transition table, one selected local stencil, one observed output cell, classifier adjacency, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}}$.

Constructive story: A $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ packet is generated from a finite three-cell Rule 110 window, the selected transition-table row, a local-stencil binding, an observed next-cell row, classifier adjacency, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6004.1, Definition 6004.2, and Definition 6004.3 seed the finite Rule 110 local-stencil observation packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite orbit replay, global pattern catalogs, glider classification, universality, host evaluator semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked finite-window admission, transition-table exposure, local-stencil binding, observed-output exactness, classifier-adjacency routing, transport, replay, provenance, and local naming for BEDC.Derived.RuleOneTenLocalStencilObservationUp.

The $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$ carrier consumes the cellular automaton update surface of chapter 934 and the Rule 110 pattern classifier of chapter 5844. Its local stencil row supplies the finite neighbourhood read before any glider catalogue, global orbit account, or Cook-frontier budget is visible. The downstream trust-substrate packet of chapter 6517 consumes this local-stencil audit row after the Rule 110 transition table, pattern classifier, and finite window are displayed.

Chapter 6005

A Concrete Naming Certificate for $\mathsf{LocatedUniformLimit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedUniformLimit}^{\uparrow}$ packet records the finite certificate surface for a located uniform-limit handoff without turning the handoff into host completeness. It sits over the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the terminal real-name seal of chapter 38, the modulus carrier of chapter 5887, and the uniform continuity surface of chapter 130. The packet gives Real and Completion consumers a displayed common-modulus route for a located uniform Cauchy family; it does not select a limit, quotient functions, use countable choice, invoke trichotomy, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6005.1 (LocatedUniformLimit carrier)**.** *A* $\mathsf{LocatedUniformLimit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, S, M, W, R, E, G, H, C, P, N)$$

with the following visible rows:

1. *I is the index and source-domain row whose accepted points are read through displayed histories and classifier transport;*
2. *S is the sequence row, listing the finite family members and the graph rows through which each member is inspected;*
3. *M is the common located uniform Cauchy modulus row, giving one displayed precision schedule for all accepted source windows;*
4. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row for indices, source points, and value names;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled value windows and their rational tail budgets;*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the stream windows, regular rational readbacks, and common modulus checks have been displayed;*
7. *G is the graph row relating the sequence entries, the common modulus, and the terminal seal read by downstream Real-facing consumers;*
8. *H records componentwise* hsame *transport for index rows, sequence rows, modulus rows, windows, readbacks, graph rows, real seals, provenance, and local naming rows;*

9. *C records displayed* Cont *replay from index admission through sequence lookup, uniform-modulus inspection, finite-window readback, graph-row comparison, and real-seal handoff;*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedUniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformLimit}^{\uparrow}}$ *row.*

The classifier relates two accepted packets exactly when their index rows, sequence rows, common modulus rows, finite stream windows, regular rational readbacks, real-seal rows, graph rows, and provenance rows transport componentwise along H and replay through the same displayed routes in C. No host function equality, quotient real equality, selected limit, countable choice, trichotomy, or ambient completeness principle is a carrier coordinate.

Lean4 ✓ `BEDC.Derived.LocatedUniformLimitUp.LocatedUniformLimitTasteGate single carrier alignment`

Theorem 6005.2 (LocatedUniformLimit NameCert obligations)**.** *For an accepted* $\mathsf{LocatedUniformLimit}^{\uparrow}$ *carrier $U = (I, S, M, W, R, E, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedUniformLimit}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed index, sequence, common modulus, window, readback, real-seal, graph, transport, replay, provenance, and local naming rows;*
2. *common-modulus exactness, requiring each value comparison requested by a Real-facing consumer to use the same located uniform Cauchy modulus row M before the terminal seal E can be inspected;*
3. *finite-window readback, requiring every index, source, and value observation to pass through W and R rather than through an external sequence or function equality predicate;*
4. *graph-row compatibility, requiring G to connect the displayed sequence entries, common modulus checks, regular rational readbacks, and* $\mathsf{Real}^{\uparrow}$ *seal rows;*
5. hsame *and* Cont *replay, requiring all index, sequence, modulus, readback, graph, and seal rows to move together under H and C;*
6. *provenance and non-escape, stating that the package exports only the named finite rows and no host completeness theorem, quotient function equality, selected limit, countable choice, or trichotomy principle.*

Proof. Carrier admission is projection from Definition 6005.1. The displayed packet lists the index row, sequence row, common located modulus, finite windows, regular rational readbacks, real seal, graph row, transport ledger, replay ledger, package provenance, and local name.

The common-modulus obligation is exactly the role of M. Every Real-facing comparison must first request a precision through the same located uniform Cauchy row, then inspect the scheduled source and value windows in W.

Finite-window readback is forced by the order W, R, E. The packet gives no coordinate from which a consumer could read an infinite sequence, a host function, or a quotient equality before the regular rational window has been displayed.

Graph-row compatibility is the projection through G. It records the finite route joining the inspected family member, the common modulus check, the readback row, and the terminal real seal.

Finally, H, C, P, N package the same route after classifier transport and continuation replay. Since none of the excluded principles is a row of the carrier, none can be exported by the local NameCert surface. □

Theorem 6005.3 (LocatedUniformLimit L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{LocatedUniformLimit}^{\uparrow}$ *carrier factors through the finite route*

$$I, S, M, W, R, E, G, H, C, P, N.$$

The consumer may inspect displayed index rows, finite sequence entries, the common located uniform Cauchy modulus, finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seal row, the graph row, componentwise transport, continuation replay, provenance, and the local name. It cannot bypass this route to obtain host completeness, quotient function equality, quotient real equality, a selected limit, countable choice, or trichotomy.*

Proof. Start with an L10 consumer asking for a Real-facing endpoint from the packet. The only entry point is the displayed index and sequence surface I, S, so the consumer must name the inspected family member before any value row is read.

The common located uniform Cauchy modulus M then supplies the precision discipline. The finite windows in W and the regular rational readback R turn that discipline into displayed rational data before the terminal seal E is available.

The graph row G records that the same route connects the sequence entry, modulus check, readback, and seal. Transport and replay through H and C preserve that route componentwise.

The package and local name rows P, N expose exactly this finite handoff. Because the excluded host principles are absent from the carrier coordinates, they cannot be read by any L10 consumer of the certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUniformLimit}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUniformLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6005.1, Definition 6005.2, and Definition 6005.3 seed the located uniform-limit certificate over finite index rows, sequence rows, common located uniform Cauchy moduli, stream windows, regular rational readbacks, graph rows, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUniformLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: The chapter does not close host completeness, quotient function equality, quotient real equality, selected limits, countable choice, trichotomy, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, common-modulus exactness, finite-window readback, graph-row compatibility, transport replay, provenance, and Real-seal non-escape for the same packet.

Constructive story: The construction reads finite index and sequence rows, applies one located uniform Cauchy modulus, replays StreamName windows and RegSeqRat readbacks, and exposes the Real seal only through hsame, Cont, Pkg, and local NameCert rows.

$\mathsf{FiniteIntervalLattice}^{\uparrow}$ supplies the finite meet/join index surface consumed by chapter 9072 and the dyadic-net route.

Chapter 6006

A Concrete Naming Certificate for $\mathsf{FiniteIntervalLattice}^{\uparrow}$

Following Bishop–Bridges constructive interval and net arguments, the $\mathsf{FiniteIntervalLattice}^{\uparrow}$ packet records the finite lattice of dyadic interval cells that a real-and-completion consumer may inspect before it asks for limits, covers, or compact-window continuity. It sits after the dyadic rational arithmetic of chapter 291, the dyadic interval surface of chapter 306, the finite stream-window interface of chapter 126, the regular rational readback of chapter 301, the located interval carrier of chapter 1709, and the terminal real-name seal of chapter 38. The packet gives L10 Real and Completion routes a visible finite meet-join substrate; it is not an ambient interval-domain completeness theorem, not an open-cover compactness principle, and not a quotient real equality row.

Definition 6006.1 (Finite interval lattice carrier)**.** *A* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, I, L, J, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and radius ledger used by all interval cells.*
2. *I is the finite list of* $\mathsf{DyadicInterval}^{\uparrow}$ *cells with displayed lower and upper endpoints.*
3. *L is the located-interval source row, read through chapter 1709.*
4. *J is the finite join table: every listed pair of cells either has a displayed least containing cell or a displayed refusal row.*
5. *M is the finite meet table: every listed pair of cells either has a displayed greatest common subcell or a displayed empty-intersection refusal row.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that records which endpoint and cell observations are inspected.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from the displayed endpoint windows.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after D, I, L, J, M, S, R have been displayed.*
9. *H records componentwise* hsame *transports for all visible interval-lattice rows.*
10. *C records displayed* Cont *replays from dyadic endpoint reads to lattice cells, meet-join tables, stream windows, regular readback, and Real sealing.*

11. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalLattice}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalLattice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, L, J, M, S, R, E, H, C, P, N$. It has no coordinate for an infinite interval lattice, selected limit point, quotient real equality, open-cover compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteIntervalLatticeUp.FiniteIntervalLatticeTasteGate single carrier alignment`

Theorem 6006.2 (Finite interval lattice NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *carrier*

$$Q = (D, I, L, J, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalLattice}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic endpoint ledger exposure, finite dyadic interval cell exposure, located-interval source exposure, finite join and meet tables, stream-window inspection, regular rational readback, Real seal nonescape, component-wise transport, continuation replay, provenance, and local naming. Every consumer of the lattice must read the displayed rows before using it in dyadic-net, located-interval, compact-window, or continuity arguments.*

Proof. Carrier admission is projection from Definition 6006.1. The packet exposes a finite tuple of dyadic endpoint rows, interval cells, located interval data, meet-join tables, window rows, readback rows, seal rows, transport, replay, provenance, and local naming.

The lattice operations are finite because J and M are displayed tables over the listed cells I. A consumer may inspect only the table entry or refusal row already stored in the packet, and the dyadic endpoint ledger D supplies the comparisons used by those entries.

The analytic handoff then passes through S, R, and E. The stream row records finite endpoint observations, the regular rational row records the accepted readback from those windows, and the Real row seals the same route without adding a selected limit, quotient equality, or ambient completeness coordinate.

Finally, H, C, P, and N preserve the displayed rows as BEDC certificate data. Thus every obligation is discharged through finite BHist, hsame, Cont, Pkg, and NameCert rows, and no host interval lattice or open-cover compactness theorem can enter the local surface. □

Theorem 6006.3 (Finite interval lattice endpoint order)**.** *Let $Q = (D, I, L, J, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *carrier. Every listed dyadic interval cell exposed by I must display its lower and upper endpoint comparison through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D before any meet or join table entry in M or J can be read as a lattice operation.*

Consequently J and M are public only as endpoint-ordered finite tables over the cells listed in I. They cannot supply lattice operations for L10 Real or Completion consumers when the endpoint-order evidence from D is absent.

Proof. Project the carrier of Definition 6006.1. The row D is the dyadic endpoint and radius ledger, while I is the finite list of dyadic interval cells with displayed lower and upper endpoints.

Read a cell from I. Its lower and upper endpoint comparison is not a property of an ambient interval lattice; it is a displayed dyadic comparison supplied by D and carried with the cell row.

The finite join table J and meet table M are indexed by the cells already listed in I. A table entry can therefore be interpreted only after the endpoint order of the two source cells and the candidate result cell has been exposed through D.

If that endpoint-order evidence is hidden, the carrier still contains symbols named J and M, but they no longer certify meet or join operations for the visible interval cells. Hence L10 Real and Completion consumers must read D and I before using the meet and join tables. □

Theorem 6006.4 (Finite interval lattice Real-completion route)**.** *Let $Q = (D, I, L, J, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *carrier. Every* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of a meet or join read from M or J must pass through the finite route*

$$D, I, L, J, M, S, R, E, H, C, P, N.$$

Thus the consumer first reads dyadic endpoint rows, finite interval-cell rows, the located interval source, the meet and join tables, StreamName windows, RegSeqRat readback, the Real seal, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row before reading completion-facing data.*

Proof. Begin with the carrier of Definition 6006.1. The tuple exposes the dyadic endpoint ledger D, finite interval cells I, located source row L, meet and join tables M, J, stream windows S, regular readback R, Real seal E, and structural rows H, C, P, N.

Apply the endpoint-order theorem Definition 6006.3. It forces every meet or join table entry to read the lower and upper endpoint comparisons from D through the listed cells I before the entry can serve as a lattice operation.

The located source and analytic handoff then remain downstream of the same finite route. The row L supplies the located interval source, S records the inspected endpoint windows, R supplies the $\mathsf{RegSeqRat}^{\uparrow}$ readback, and E exposes the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name only those displayed rows. Therefore a $\mathsf{Real}^{\uparrow}$ or $\mathsf{Completion}^{\uparrow}$ consumer cannot read completion-facing data from the lattice unless the dyadic endpoints, interval cells, located source, stream windows, regular readback, Real seal, transport, replay, provenance, and local NameCert rows have already been exposed. □

Theorem 6006.5 (Finite interval lattice refinement induction)**.** *Let $Q = (D, I, L, J, M, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *carrier. Suppose a displayed subdivision-depth row refines each listed cell of I by finitely many dyadic subcells whose endpoint comparisons are read through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D. Then the meet and join tables refine by induction over the displayed depth: at every finite stage the refined tables remain indexed by dyadic endpoint rows from D, expose only* $\mathsf{StreamName}^{\uparrow}$ *windows from S, preserve* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R, and pass the terminal* $\mathsf{Real}^{\uparrow}$ *seal only through E.*

Consequently a $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{Completion}^{\uparrow}$ *consumer of the refined lattice must still read the route*

$$D, I, L, J, M, S, R, E, H, C, P, N$$

before using any refined meet, refined join, or refined endpoint cell.

Proof. Induct on the displayed subdivision-depth row. At depth zero the refined table is the original finite table of Definition 6006.4, so all meet and join reads already pass through $D, I, L, J, M, S, R, E, H, C, P, N$.

For the successor step, split each currently displayed cell into its finitely listed dyadic subcells. The endpoint-order theorem Definition 6006.3 forces the lower and upper endpoint comparisons of every subcell to be read through D before a table entry can be interpreted.

The finite meet and join rows are then rebuilt table-entry by table-entry. Each entry either displays the least containing refined cell, displays the greatest common refined subcell, or records the corresponding refusal row already governed by J and M.

Finally the analytic readback is unchanged. The stream windows remain the finite windows in S, the rational readback remains the row R, and the terminal real-name exposure is still sealed by E. Transport, replay, provenance, and local naming in H, C, P, N preserve the same route, so the induction cannot introduce an infinite lattice, selected limit, quotient equality, or ambient completion principle. □

Theorem 6006.6 (Finite interval lattice mesh-refinement consumption)**.** *Every* $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *packet reads*

$$D, I, L, J, M, S, R, E, H, C, P, N$$

before interpreting any refined parent-cell table. The downstream consumer of chapter 9072 may refine cells only after the endpoint order, Real-completion route, and refinement induction of the lattice packet have been displayed.

Proof. Start with endpoint order, Definition 6006.3. It forces every source cell and candidate refined cell to expose lower and upper endpoint comparisons through the dyadic ledger D before a meet or join table entry can be read.

Next use the Real-completion route of Definition 6006.4. The route places L, S, R, E, H, C, P, N downstream of the finite dyadic interval rows, so a completion-facing mesh consumer cannot bypass the located source, stream windows, regular readback, Real seal, transport, replay, provenance, or local naming.

Finally apply refinement induction, Definition 6006.5. The refined parent-cell table is rebuilt at finite subdivision depth from the same D, I, J, M rows and preserves the same analytic handoff rows.

Thus the $\mathsf{FiniteIntervalMeshRefinement}^{\uparrow}$ consumer named by chapter 9072 reads the full lattice route before interpreting refined parent cells, rather than importing selected partitions, quotient real equality, open-cover compactness, or ambient completion data. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalLattice}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntervalLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6006.1 and Definition 6006.2 seed the finite dyadic interval lattice boundary for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntervalLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite interval-domain completeness, open-cover compactness, selected limits, quotient real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, meet-join table exactness, dyadic endpoint transport, RegSeqRat readback handoff, RealUp seal nonescape, and consumer-boundary proofs for BEDC.Derived.FiniteIntervalLatticeUp.

Constructive story: A $\mathsf{FiniteIntervalLattice}^{\uparrow}$ packet is generated from dyadic endpoint rows, finite dyadic interval cells, located interval source rows, finite meet and join tables, stream windows, regular rational readback, Real seal, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theorem 6006.7 (Finite interval lattice dyadic-interval dependency)**.** *Every accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *meet or join read consumes the dyadic endpoint ledger of chapter 306 and the located-interval row of chapter 1709 before it reaches the StreamName, RegSeqRat, and Real seal rows. In the carrier*

$$Q = (D, I, L, J, M, S, R, E, H, C, P, N),$$

the finite tables J and M have no independent interval source outside the displayed D, I, L rows.

Proof. Project the carrier of Definition 6006.1. The endpoint ledger D, the dyadic interval list I, and the located interval source L are the only interval-source rows before the meet and join tables.

The join table J and meet table M are finite tables over the cells already listed in I. Their entries and refusal rows therefore use the displayed dyadic endpoints and located-interval source rather than a separate interval-domain object.

Only after D, I, L, J, M have been displayed can the analytic handoff proceed to S, R, E. The StreamName row records inspected endpoint windows, the RegSeqRat row reads those windows, and the Real row seals the same route. Transport, replay, provenance, and local naming preserve this dependency and do not add an independent interval source. □

Theorem 6006.8 (Finite interval lattice dyadic-net forward link)**.** *Every accepted* $\mathsf{FiniteIntervalLattice}^{\uparrow}$ *interval packet supplies the finite interval-index lattice consumed by the dyadic interval net route of chapter 5966. For a carrier*

$$Q = (D, I, L, J, M, S, R, E, H, C, P, N),$$

the dyadic-net consumer reads endpoint data from D, interval cells from I, located interval support from L, and finite order operations from J, M before it can use the lattice as an index surface.

Proof. Project the carrier of Definition 6006.1. The dyadic endpoints are supplied by D, the finite interval cells by I, and the located interval source by L.

Apply the endpoint-order theorem Definition 6006.3. It makes each cell and each table entry dependent on the displayed endpoint comparisons rather than on an ambient interval domain.

Use the Real-completion route Definition 6006.4. The rows J, M are finite tables over the same listed cells, and H, C, P, N transport, replay, source, and name only those rows.

Thus a dyadic interval net may consume the finite interval-index lattice only through the displayed packet $D, I, L, J, M, H, C, P, N$. No infinite interval lattice, quotient real equality, selected limit point, or host compactness principle is exported. □

Chapter 6007

A Concrete Naming Certificate for $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$

The $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ packet records the finite audit boundary inherited from the Lean kernel acceptance tradition: a generated candidate is either accepted as a declaration in an environment or refused before it becomes a BEDC theorem row. The packet concretizes chapter 9364 as displayed NameCert data over generated candidates, kernel classifier stamps, environment ledger rows, accepted declarations, axiom-query refusal, replay, provenance, and local naming. It is a finite audit boundary for formal content, not a proof of the kernel implementation, not a semantic interpretation of a theorem name, and not an additional logical axiom. It consumes $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ for accepted-declaration rows via chapter 549 and $\mathsf{AxiomQueryLedger}^{\uparrow}$ for forbidden-ancestry refusal rows via chapter 654.

Definition 6007.1 (Kernel acceptance boundary carrier)**.** *A* $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (G, K, A, E, L, F, H, C, P, N)$$

with the following displayed rows:

1. *G is the generated candidate row: a proposed declaration name, type, and proof term or inductive signature.*
2. *K is the kernel classifier stamp recording acceptance or refusal at the boundary.*
3. *A is the accepted declaration row, present only when the kernel classifier admits the candidate.*
4. *E is the environment ledger row containing the finite imported declarations and namespace context read by the classifier.*
5. *L is the ledger row for build replay, source location, and command provenance.*
6. *F is the axiom-query refusal row: forbidden ancestry, unresolved marker targets, and non-kernel semantic claims are refused as coordinates of the accepted declaration.*
7. *H records componentwise* hsame *transport for generated candidate, classifier, declaration, environment, ledger, and refusal rows.*
8. *C records displayed* Cont *replay from candidate generation to kernel classification, environment lookup, accepted declaration exposure, axiom query, and refusal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$*, generated candidates, accepted declarations, environment ledgers, axiom-query refusal,* BHist*,* hsame*,* Cont*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, K, A, E, L, F, H, C, P, N$. It has no coordinate for kernel self-correctness, compiler correctness, hardware trust, semantic adequacy of theorem names, or paper-side closure status.

Theorem 6007.2 (Kernel acceptance boundary NameCert obligations)**.** *For every accepted* KernelAcceptanceBoun *carrier*

$$Q = (G, K, A, E, L, F, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBoundary}^{\uparrow}}$ *surface consists exactly of candidate exposure, kernel classifier stamping, accepted declaration exposure, environment ledger display, replay ledger display, axiom-query refusal, componentwise transport, continuation replay, provenance, and local naming. A BEDC audit consumer may cite the boundary only through those finite rows, and it may not replace them by a claim of kernel self-verification, semantic adequacy, compiler correctness, or hardware correctness.*

Proof. Project the carrier in Definition 6007.1. The generated candidate, classifier stamp, accepted declaration, environment ledger, replay ledger, and refusal row are all displayed coordinates of the same finite packet.

The kernel-facing obligation is the boundary row K together with A and E. Acceptance of a declaration is visible only as the classifier stamp read in the displayed environment; refusal is likewise visible as a boundary stamp rather than as a semantic explanation of the candidate.

The audit-facing obligation is F. It records that forbidden ancestry, unresolved marker targets, and non-kernel semantic claims are not accepted-declaration coordinates, so a downstream BEDC audit cannot turn kernel acceptance into unlimited certification.

Finally H, C, P, and N replay and name the finite route. Since no row stores kernel self-correctness, compiler correctness, hardware trust, or paper-side closure status, the local NameCert obligations stop exactly at the accepted-declaration and axiom-query boundary. □

Theorem 6007.3 (Kernel acceptance boundary carrier admission)**.** *For every finite* BHist *packet*

$$Q = (G, K, A, E, L, F, H, C, P, N)$$

accepted by $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$*, carrier admission is exactly the display of generated-candidate rows, kernel classifier stamps, accepted-declaration exposure, environment-ledger rows, replay ledgers, axiom-query refusal rows, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. No consumer can enter the carrier through a hidden theorem-name semantics row, compiler-correctness row, hardware row, or paper-closure row.*

Proof. Project the coordinates of Q from Definition 6007.1. The generated candidate row G, classifier stamp K, accepted declaration row A, environment ledger E, replay ledger L, and axiom-query refusal row F are the data admitted into the finite carrier.

The transport and replay coordinates H and C keep admission stable under componentwise hsame transport and displayed Cont replay. The provenance and naming coordinates P and N then package the same finite route as Pkg provenance and local NameCert evidence.

Since the carrier definition lists no hidden coordinate for kernel self-correctness, compiler correctness, hardware trust, theorem-name semantics, or paper closure, carrier admission cannot read such data. It remains a projection of the displayed packet rows. □

Theorem 6007.4 (Kernel acceptance boundary accepted-declaration exposure)**.** *In every accepted* KernelAcceptanceBoundary$^{\uparrow}$ *packet*

$$Q = (G, K, A, E, L, F, H, C, P, N),$$

the accepted declaration row A is exposed only together with the generated candidate G, classifier stamp K, environment ledger E, replay ledger L, axiom-query refusal row F, componentwise hsame *transport H,* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N. The exposure route is the accepted-declaration source row supplied by chapter 549, with forbidden-ancestry refusal supplied by chapter 654.*

Proof. Read the accepted declaration row A from the carrier packet. It is not isolated: the same packet displays the generated candidate G, the classifier stamp K, and the environment ledger E that record where the declaration was admitted.

Next read L and F. The replay ledger keeps the finite command route visible, while the axiom-query refusal row keeps forbidden ancestry and unresolved target queries outside the accepted-declaration coordinates.

Finally use H, C, P, and N to transport, replay, package, and name the exposure route. The cited sibling chapters provide the accepted-declaration and forbidden-ancestry source rows; the theorem asserts only that the local packet exposes those rows, not that the kernel implementation is semantically certified. □

Theorem 6007.5 (Kernel acceptance boundary closed trace forward link)**.** *Every* ClosedObservationProofTrace$^{\uparrow}$ *consumer of chapter 6747 must read the accepted-declaration row A and the axiom-query refusal row F through the* KernelAcceptanceBoundary$^{\uparrow}$ *carrier*

$$Q = (G, K, A, E, L, F, H, C, P, N)$$

before replaying a closed proof trace. The forward dependency is the finite route through generated candidate, classifier stamp, accepted declaration, environment ledger, replay ledger, refusal row, transport, replay, provenance, and local naming; it is not a claim of kernel self-correctness or semantic adequacy of theorem names.

Proof. Apply accepted-declaration exposure, Definition 6007.4. It displays A only together with the generated candidate, classifier stamp, environment ledger, replay ledger, refusal row, transport, replay, provenance, and local naming.

Read the refusal row F from the same carrier. This row keeps forbidden ancestry, unresolved marker targets, and non-kernel semantic claims outside the accepted-declaration coordinates.

A closed proof trace can therefore consume kernel acceptance only through the displayed boundary packet. The downstream trace replay receives an accepted-declaration row and a refusal row, not a proof of kernel implementation correctness or theorem-name semantics. □

Theorem 6007.6 (Kernel acceptance boundary replay nonescape)**.** *Every* Cont *replay of a* KernelAcceptanceBounda *packet*

$$Q = (G, K, A, E, L, F, H, C, P, N)$$

remains confined to generated-candidate rows, classifier stamps, accepted-declaration rows, environment ledgers, replay ledgers, axiom-query refusal rows, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *evidence. In particular, replay cannot manufacture a theorem from forbidden ancestry, unresolved marker targets, non-kernel semantic claims, or any unstored kernel-correctness premise.*

Proof. Start with the displayed replay coordinate C. By carrier definition, C replays the path from candidate generation through kernel classification, environment lookup, accepted declaration exposure, axiom query, and refusal boundary.

The transport coordinate H is componentwise over the same rows, so replay after hsame transport still reads only $G, K, A, E, L, F, H, C, P, N$. The provenance and naming rows P and N record that finite route as package and local certificate data.

The refusal coordinate F blocks the only listed escape route: forbidden ancestry, unresolved marker targets, and non-kernel semantic claims are refusal data rather than accepted-declaration data. Since no unstored kernel-correctness premise appears in the carrier, replay non-escape follows by projection of the packet rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{KernelAcceptanceBoundary}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6007.1 and Definition 6007.2 seed the finite kernel-acceptance audit boundary for generated formal content.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KernelAcceptanceBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close kernel self-correctness, compiler correctness, hardware trust, semantic adequacy of theorem names, paper-side closure status, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, accepted-declaration exposure, environment-ledger exactness, axiom-query refusal, replay transport, and audit-boundary nonescape proofs for BEDC.Derived.KernelAcceptanceBoundaryUp

Constructive story: A $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ packet is generated from a candidate declaration, kernel classifier stamp, accepted declaration row, environment ledger, replay ledger, axiom-query refusal row, componentwise transport, continuation replay, provenance, and a local NameCert row.

Theorem 6007.7 (Kernel acceptance boundary acceptance audit dependency route)**.** *Every accepted-declaration exposure used to name a* $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ *packet first factors through the* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$ *source row, and every forbidden-ancestry refusal used by the same packet first factors through the* $\mathsf{AxiomQueryLedger}^{\uparrow}$ *refusal row. Thus the local boundary can be named only after the accepted-declaration route and the axiom-query route have both been displayed inside*

$$Q = (G, K, A, E, L, F, H, C, P, N).$$

It does not supply kernel self-correctness, compiler correctness, hardware trust, or semantic adequacy of theorem names.

Proof. Project the accepted carrier of Definition 6007.1. The row A is the accepted-declaration coordinate, and the row F is the axiom-query refusal coordinate.

Apply Definition 6007.4. It already exposes A only together with $G, K, E, L, F, H, C, P, N$ and identifies the accepted-declaration source row with chapter 549.

Now read the sibling dependency route. It states that the forbidden-ancestry refusal row is supplied by chapter 654 before local naming. Therefore A and F must both be present before the $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ name can be consumed.

The final exclusions are inherited from the carrier and sibling route: neither dependency supplies kernel self-correctness, compiler correctness, hardware trust, or semantic adequacy. □

The accepted-declaration exposure row of $\mathsf{KernelAcceptanceBoundary}^{\uparrow}$ is supplied by chapter 549, while the forbidden-ancestry refusal row is supplied by chapter 654. This sibling route only states that the boundary packet reads those two displayed audit dependencies before local naming; it does not claim kernel self-correctness, compiler correctness, or semantic adequacy. For the L10 Real and Completion path, a downstream selector such as chapter 1576 may cite the boundary

only after the accepted-declaration and axiom-query rows have been displayed, so kernel acceptance remains an audit boundary rather than a Real completeness, selector existence, or semantic adequacy principle.

Chapter 6008

A Concrete Naming Certificate for $\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}$ packet records the finite refinement point where a located real interval is read together with a regular Cauchy real name under one dyadic tolerance ledger. It sits beside the located interval surface of chapter 1709, the Bishop located Cauchy real packet of chapter 1850, and the four-object real-line interface of chapter 2062. The packet is an L10 Real and Completion naming certificate; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, trichotomy, countable choice, or bridge checking.

Definition 6008.1 (Located Cauchy interval refinement carrier)**.** *A* $\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (I, B, D, S, R, A, W, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row, supplying the interval endpoints, radius ledger, locator reads, locatedness modulus, and metric bracket route.*
2. *B is the* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *row, exposing the located Cauchy real interface that already binds the located interval and regular Cauchy readback surfaces.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and positive-tolerance ledger shared by interval refinement and regular Cauchy comparison.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the Cauchy name.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same stream windows and the same dyadic ledger D.*
6. *A is the* $\mathsf{RealRationalApproximation}^{\uparrow}$ *row comparing the regular rational readback with the located interval endpoints at the displayed tolerance.*
7. *W is the refinement witness row, recording that the interval width request and the regular Cauchy tail request consume the same dyadic tolerance before any terminal real-name seal is read.*
8. *H records componentwise* hsame *transport for the interval, located-Cauchy, dyadic, stream, regular-readback, approximation, witness, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a refinement request through $I, B, D, S, R, A, W, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{BishopLocatedCauchyReal}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealRationalApproximation}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, B, D, S, R, A, W, H, C, P, N$. The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, selected limits, quotient stream equality, host real equality, trichotomy, countable choice, compactness, or bridge checking.*

Lean4 ✓ `BEDC.Derived.LocatedCauchyIntervalRefinementUp.LocatedCauchyIntervalRefinementTasteGate`
`single carrier alignment`

Theorem 6008.2 (Located Cauchy interval refinement NameCert obligations)**.** *For every accepted* $\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}$ *carrier $J = (I, B, D, S, R, A, W, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedCauchyIntervalRefinement}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the located interval row I, located-Cauchy row B, shared dyadic ledger D, stream window S, regular readback R, rational approximation row A, refinement witness row W, transport row H, continuation row C, provenance row P, and local naming row N;*
2. *located interval admission, requiring every endpoint, radius, locator, locatedness, and metric-bracket request to pass through the displayed* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row I;*
3. *located Cauchy admission, requiring every Cauchy-facing read to pass through the* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *row B together with D, S, R, A, W before any comparison with the interval endpoints is consumed;*
4. *shared tolerance discipline, requiring D to be visible before both the interval refinement request and the regular Cauchy readback route are read;*
5. *approximation compatibility, requiring A to compare R with the endpoints in I only under the same dyadic row D;*
6. *non-escape, requiring P, N to export no* $\mathsf{Real}^{\uparrow}$ *completeness theorem, selected limit, quotient stream equality, host real equality, trichotomy, countable-choice selector, compactness principle, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6008.1. The accepted packet exposes exactly the located interval, located-Cauchy, dyadic, stream, regular-readback, approximation, witness, and structural rows.

Located interval admission is supplied by I, using the sibling surface of chapter 1709. Located Cauchy admission is supplied by B, using the sibling surface of chapter 1850.

The shared tolerance row is D. Since both the endpoint refinement and the Cauchy tail read cite D before S, R, A, W can be consumed, the packet has no route that compares interval data and Cauchy data under unrelated budgets.

Approximation compatibility is the row A. It reads the regular rational approximation against the interval endpoints through the displayed tolerance, while H, C, P, N preserve, replay, source, and name only the same finite packet.

No excluded object occurs as a carrier coordinate. Hence the local NameCert surface cannot export completeness, a selected limit, quotient stream equality, host real equality, trichotomy, choice, compactness, or bridge checking. □

Theorem 6008.3 (Located Cauchy interval refinement Cauchy route)**.** *Every accepted* LocatedCauchyIntervalRefine *packet that refines a located interval by a Cauchy readback factors through the route*

LocatedRealInterval$^{\uparrow}$, BishopLocatedCauchyReal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealRationalApprox

The same dyadic row D is consumed before the interval row I is narrowed and before the regular Cauchy route S, R, A is compared with the endpoints. The route may then replay componentwise hsame*,* Cont*,* Pkg*, and local* NameCert *data, but it exports no selected limit, quotient stream equality, host real equality, trichotomy, countable choice, ambient* Real$^{\uparrow}$ *completeness, or bridge checking.*

Proof. Start with an accepted carrier $J = (I, B, D, S, R, A, W, H, C, P, N)$. Projection gives the located interval row I, the located-Cauchy row B, the dyadic tolerance ledger D, the stream window S, the regular readback row R, and the rational approximation row A.

The interval side is read through chapter 1709. The Cauchy side is read through chapter 1850, with the real-line route of chapter 2062 supplying the surrounding DyadicRatCore, StreamName, RegSeqRat, and Real seal discipline.

The witness row W records that the interval narrowing and the Cauchy tail comparison cite the same row D. Thus the route cannot split into one tolerance for locatedness and another tolerance for regular Cauchy readback.

The structural rows H, C, P, N transport, replay, source, and name the displayed route only. Since no excluded coordinate is present, the refinement packet remains a finite NameCert boundary rather than a completeness or limit-selection principle. □

Closure status for NameCert$_{\text{LocatedCauchyIntervalRefinement}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\text{LocatedCauchyIntervalRefinement}^{\uparrow}}$, seed)

Scope closed: Definition 6008.1, Definition 6008.2, and Definition 6008.3 seed the located Cauchy interval refinement boundary over LocatedRealInterval$^{\uparrow}$, BishopLocatedCauchyReal$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, RealRationalApproximation$^{\uparrow}$, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedCauchyIntervalRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real$^{\uparrow}$ completeness, selected limits, quotient stream equality, host real equality, trichotomy, countable choice, compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval admission, located Cauchy admission, shared tolerance discipline, approximation compatibility, and non-escape proofs for BEDC.Derived.LocatedCauchyIntervalRefinementU

Constructive story: A LocatedCauchyIntervalRefinement$^{\uparrow}$ packet is generated from a LocatedRealInterval row, a BishopLocatedCauchyReal row, a shared DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat regular readback, a RealRationalApproximation row, a refinement witness row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6009

A Concrete Naming Certificate for $\mathsf{KleeneBrouwerOrder}^{\uparrow}$

Following Kleene and Troelstra–van Dalen on well-founded tree orderings in constructive bar and spread arguments, the $\mathsf{KleeneBrouwerOrder}^{\uparrow}$ packet records the finite order surface on displayed tree nodes. It sits beside the prefix-tree boundary of chapter 1622, the finite bar route of chapter 400, the bar-induction packet of chapter 1675, the spread-law admission surface of chapter 1532, the finite fan theorem packet of chapter 5863, and the finite-window interface of chapter 126. The packet names only finite node, prefix, and first-disagreement rows; it does not select an infinite branch, import classical well-ordering, decide global tree equality, or invoke countable choice.

Definition 6009.1 (Kleene–Brouwer order carrier)**.** *A* $\mathsf{KleeneBrouwerOrder}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, N, P, F, L, A, H, C, Q, M)$$

with the following displayed rows:

1. *T is the finite tree-node row, listing the displayed nodes and their admission decisions in the inspected prefix tree.*
2. *N is the node-code row, recording each inspected word as a finite* $\mathsf{List}^{\uparrow}$ *of* $\mathsf{Nat}^{\uparrow}$ *entries, with the empty word and parent-prefix witnesses exposed.*
3. *P is the prefix row, recording the displayed prefix relation between listed nodes and the finite parent spine used by tree induction.*
4. *F is the first-disagreement row, recording the first coordinate at which two incomparable listed nodes differ, together with the left/right* $\mathsf{Nat}^{\uparrow}$ *comparison visible at that coordinate.*
5. *L is the* $\mathsf{Nat}^{\uparrow}/\mathsf{List}^{\uparrow}$ *prefix ledger used to traverse lengths, heads, tails, and coordinate comparisons without reading a host sequence.*
6. *A is the order-admission row, classifying a displayed comparison by the Kleene–Brouwer rule: later extensions are below their prefixes, and incomparable nodes are ordered by the first displayed disagreement.*
7. *H records componentwise* hsame *transport for tree nodes, node codes, prefix rows, first-disagreement rows, ledgers, order admission, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a comparison request through T, N, P, F, L, A, H.*

9. *Q is the* Pkg *provenance over* KleeneBrouwerOrder$^{\uparrow}$, KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, FiniteFanTheorem$^{\uparrow}$, BrouwerBarInduction$^{\uparrow}$, SpreadLaw$^{\uparrow}$, StreamName$^{\uparrow}$, Nat$^{\uparrow}$, List$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *M is the local* NameCert$_{\text{KleeneBrouwerOrder}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these finite rows. It has no coordinate for an infinite branch, a host total order on all sequences, quotient equality of streams, classical well-ordering, a bar-induction conclusion, a spread choice functional, or countable choice.

Lean4 (definition) `BEDC.Derived.KleeneBrouwerOrderUp.KleeneBrouwerOrderUp`

Theorem 6009.2 (Kleene–Brouwer order NameCert obligation surface)**.** *For every accepted* KleeneBrouwerOrder$^{\uparrow}$ *carrier $K = (T, N, P, F, L, A, H, C, Q, M)$, the local* NameCert$_{\text{KleeneBrouwerOrder}^{\uparrow}}$ *surface consists exactly of finite tree-node admission through T, finite node coding through N, prefix exposure through P, first-disagreement exposure through F,* Nat$^{\uparrow}$/List$^{\uparrow}$ *prefix-ledger traversal through L, Kleene–Brouwer comparison admission through A, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through Q, and local naming through M.*

Every KleeneTree$^{\uparrow}$, FanTheorem$^{\uparrow}$, FiniteFanTheorem$^{\uparrow}$, BrouwerBarInduction$^{\uparrow}$, SpreadLaw$^{\uparrow}$, *or* StreamName$^{\uparrow}$ *consumer that reads this order must expose those rows before treating a displayed comparison as a well-founded tree-order step. The packet exports no infinite path, global branch selector, quotient stream equality, host well-ordering, countable-choice functional, or completed bar-induction proof.*

Proof. Project the carrier of Definition 6009.1. The only tree data available to a consumer are the displayed node rows T, N and the finite prefix row P.

For comparable nodes, the order read is forced through the prefix ledger. The row L supplies the Nat$^{\uparrow}$ lengths and List$^{\uparrow}$ parent spines, and the row A records the Kleene–Brouwer convention that a proper extension is ordered below its prefix.

For incomparable nodes, the read is forced through F. The first-disagreement coordinate and its two Nat$^{\uparrow}$ entries are displayed before A may classify the comparison, so no host comparison of completed streams is used.

The structural rows H, C, Q, M then transport, replay, source, and name the same finite comparison route. Since none of the carrier coordinates stores an infinite branch, a selected bar, quotient stream equality, a host well-ordering, or countable choice, the listed rows exhaust the NameCert obligation surface. □

Closure status for NameCert$_{\text{KleeneBrouwerOrder}^{\uparrow}}$**.**

Constructive story: A KleeneBrouwerOrder$^{\uparrow}$ packet is generated from finite tree-node rows, List/Nat node-code rows, prefix rows, first-disagreement rows, prefix ledgers, order-admission rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{KleeneBrouwerOrder}^{\uparrow}}$, seed)

Scope closed: Definition 6009.1 and Definition 6009.2 seed the finite Kleene–Brouwer comparison surface consumed by tree, fan, bar, spread, and StreamName routes.

Formal verification: FormalStatus(NameCert$_{\text{KleeneBrouwerOrder}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global well-foundedness, Brouwer bar induction, fan compactness, selected infinite branches, quotient stream equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix exactness, first-disagreement exactness, finite order classification, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.KleeneBrouwerOrderUp

Chapter 6010

A Concrete Naming Certificate for $\mathsf{DedekindCutArithmetic}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DedekindCutArithmetic}^{\uparrow}$ packet records the finite arithmetic surface of rounded located cuts before any consumer compares it with the regular Cauchy real-name route. It sits after the rounded cut interface of chapter 436, the Cauchy comparison boundary of chapter 1424, the located cut stream of chapter 5719, and the regular real-name route of chapter 301 and chapter 38. The packet names addition, negation, order, and multiplication reads as finite cut-side ledgers; it does not construct $\mathsf{Real}^{\uparrow}$ by cuts, identify quotient cuts, choose suprema, or assert completeness.

Definition 6010.1 (Dedekind cut arithmetic carrier)**.** *A* $\mathsf{DedekindCutArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, L, U, Q, S, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the source* $\mathsf{DedekindCut}^{\uparrow}$ *row, carrying the lower and upper rounded located cut observations.*
2. *L is the lower arithmetic ledger, recording finite lower-side witnesses for cut addition, negation, order, and multiplication requests.*
3. *U is the upper arithmetic ledger, recording the corresponding upper-side witnesses and disjointness guards.*
4. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and rational-operation row used to compare the arithmetic ledgers with regular rational readback.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which arithmetic observations are sampled.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained only after D, L, U, Q, S are displayed.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached by the regular Cauchy route, not by a quotient-cut construction.*
8. *O is the operation-admission ledger naming which displayed cut-side arithmetic requests have compatible regular-Cauchy readbacks.*
9. *H records componentwise* hsame *transport for cut, arithmetic, dyadic, stream, readback, seal, operation, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an arithmetic request through $D, L, U, Q, S, R, E, O, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{DedekindCutArithmetic}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{DedekindCutArithmetic}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for quotient-cut equality, host real equality, selected suprema, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DedekindCutArithmeticUp.DedekindCutArithmeticTasteGate single carrier alignment`

Theorem 6010.2 (Dedekind cut arithmetic NameCert obligations)**.** *For every accepted* $\mathsf{DedekindCutArithmetic}^{\uparrow}$ *carrier* $A = (D, L, U, Q, S, R, E, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DedekindCutArithmetic}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the cut source row D, lower and upper arithmetic ledgers L, U, dyadic tolerance row Q, stream window row S, regular readback row R, real seal row E, operation-admission row O, transport row H, continuation row C, provenance row P, and local naming row N;*

2. *cut-side operation discipline, requiring every arithmetic request to expose the lower and upper observation ledgers L, U over the same rounded located cut row D;*

3. *dyadic tolerance discipline, requiring each arithmetic comparison to consume Q before any regular readback or Real-facing seal is read;*

4. *stream and regular-readback discipline, requiring S and R to mediate all Cauchy-facing arithmetic reads;*

5. *operation-admission exactness, requiring O to name only those cut-side operation rows whose regular-Cauchy readback has passed through D, L, U, Q, S, R, E;*

6. *non-escape, requiring P, N to export no quotient-cut equality, host real equality, selected supremum, trichotomy, countable-choice selector, Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, or ambient completeness theorem.*

No local arithmetic obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6010.1. The accepted packet exposes exactly the cut source, lower and upper arithmetic ledgers, dyadic tolerance, stream window, regular readback, real seal, operation admission, transport, replay, provenance, and local naming rows.

The cut-side operation discipline is read from D, L, U. Each arithmetic request is named by lower and upper observation ledgers over the same rounded located cut source, so no operation can substitute a quotient cut or a host real representative.

The Cauchy-facing discipline is read from Q, S, R, E. The dyadic row supplies the tolerance budget, the stream row opens finite observation windows, the regular readback row exposes rational Cauchy data, and the seal row is reached only after that route is displayed.

Finally O, H, C, P, N admit, transport, replay, source, and name only the same finite route. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6010.3 (Dedekind cut arithmetic Cauchy compatibility)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{DedekindCutArithmetic}^{\uparrow}$ *carrier that reads a cut-side arithmetic operation must factor through*

$\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{DedekindCauchyComparison}^{\uparrow}$, $\mathsf{DedekindCutStream}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, R

The consumer first reads the displayed lower and upper cut arithmetic ledgers L, U *over* D*, then the dyadic tolerance row* Q*, then the stream and regular-readback rows* S, R*, and only then the Real seal row* E *and operation-admission row* O*. The route supplies compatibility of displayed cut-side operations with displayed regular-Cauchy readback; it supplies no quotient-cut arithmetic, no quotient real equality, no selected supremum, no trichotomy branch, no countable choice, and no completeness theorem.*

Proof. Open an accepted carrier $A = (D, L, U, Q, S, R, E, O, H, C, P, N)$. The only cut-side arithmetic data are the lower and upper ledgers L, U over the displayed rounded located cut row D.

To compare this arithmetic data with the regular Cauchy route, the packet must expose the dyadic tolerance row Q and the finite observation windows S. The readback row R is downstream of those rows, and the Real seal E is downstream of R.

The comparison boundary is therefore the same finite route named by the Dedekind-Cauchy and Dedekind-cut stream packets: cut observations are sampled through finite stream windows and then read as regular rational data before the terminal Real seal is available.

The operation-admission row O records only arithmetic requests that have passed through that route, while H, C, P, N transport, replay, source, and name the same data. Since no quotient-cut, quotient-real, selected-supremum, trichotomy, choice, or completeness coordinate occurs in the carrier, every Real-facing arithmetic read has exactly the displayed Cauchy-compatible boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{DedekindCutArithmetic}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DedekindCutArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6010.1, Definition 6010.2, and Definition 6010.3 seed the cut arithmetic route over DedekindCutUp, DedekindCauchyComparisonUp, DedekindCutStreamUp, DyadicRatCoreUp, StreamNameUp, RegSeqRatUp, RealUp, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DedekindCutArithmetic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient-cut arithmetic, quotient real equality, selected suprema, trichotomy, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cut-side operation discipline, dyadic tolerance discipline, stream and regular-readback discipline, operation-admission exactness, and non-escape proofs for BEDC.Derived.DedekindCutArithm

Constructive story: A $\mathsf{DedekindCutArithmetic}^{\uparrow}$ packet is generated from a DedekindCut row, lower and upper arithmetic ledgers, a DyadicRatCore tolerance row, StreamName finite windows, RegSeqRat readback, a RealUp seal, an operation-admission row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Chapter 6011

A Concrete Naming Certificate for $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Bourbaki on separated uniformities, the $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ packet records the completion neighbour where a uniformity source, a separated classifier, and a finite completion route are displayed in one NameCert surface. It sits after the uniform-space carrier of chapter 339, the Hausdorff-uniformity boundary of chapter 4072, the separated-reflection handoff of chapter 4151, the uniform-completion packet of chapter 1473, the metric-completion comparison surface of chapter 1506, the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet gives L10 completion consumers a separated-uniform completion object without quotient equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6011.1 (HausdorffUniformityCompletion carrier)**.** *A* $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (U, H, S, C, M, W, R, E, T, Q, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *source row, including the finite entourage basis and refinement ledger read before any completion-facing request.*
2. *H is the* $\mathsf{HausdorffUniformity}^{\uparrow}$ *separated-uniformity row, exposing the diagonal classifier for the same uniform source.*
3. *S is the* $\mathsf{SeparatedReflection}^{\uparrow}$ *row that routes indistinguishable Cauchy data through a displayed separated interface before a completion seal is consumed.*
4. *C is the* $\mathsf{UniformCompletion}^{\uparrow}$ *row, recording the finite Cauchy-filter or Cauchy-net completion route served by U, H, S.*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row used only when a metric-generated completion consumer asks for the same separated handoff.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which completion observations are scheduled.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the same displayed windows.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, read only after U, H, S, C, M, W, R have been displayed.*

9. *T records componentwise* hsame *transport for the source, separated classifier, reflection, completion, comparison, window, readback, seal, replay, provenance, and local naming rows.*
10. *Q records displayed* Cont *replay along the route from U, H, S through C, M, W, R, E.*
11. *P is the* Pkg *provenance over* $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{HausdorffUniformity}^{\uparrow}$, $\mathsf{SeparatedReflection}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HausdorffUniformityCompletion}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one separated-uniform completion packet. It contains no coordinate for quotient completion equality, selected Cauchy representatives, host point equality, countable choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6011.2 (HausdorffUniformityCompletion NameCert obligations)**.** *For every accepted* $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ *carrier*

$$K = (U, H, S, C, M, W, R, E, T, Q, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HausdorffUniformityCompletion}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed uniform-space source, Hausdorff-uniformity classifier, separated-reflection row, uniform-completion row, metric-completion comparison row, StreamName windows, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *uniform-source exactness, requiring every completion-facing request to begin with the finite entourage surface of U;*
3. *Hausdorff separation handoff, requiring indistinguishable Cauchy data to pass through H and then through S before the completion rows are visible;*
4. *completion readback stability, requiring C, M, W, R, E to be consumed in the displayed order whenever a uniform or metric completion consumer asks for a separated endpoint read;*
5. *transport and replay stability through T, Q for the same row order;*
6. *ledger non-escape, requiring P and N to name no quotient completion equality, selected representative, host equality, countable-choice schedule, open-cover compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

No local obligation is stored outside these displayed rows.

Proof. Carrier admission is projection from Definition 6011.1. The accepted tuple displays the uniform source, Hausdorff classifier, separated reflection, completion rows, finite windows, regular readback, real seal, and structural rows.

Uniform-source exactness follows from the first coordinate of the route. A completion-facing request must read U before it can inspect the separated classifier or any completion row, so the request is tested against finite entourage data rather than an ambient host uniformity.

The separation handoff is the paired role of H and S. The row H records diagonal indistinguishability through the Hausdorff uniformity, and S moves that evidence into the separated-reflection boundary before C or M can be consumed.

The completion readback is then confined to C, M, W, R, E. Transport and replay through T, Q preserve the same order, while P and N package only the named rows. Hence none of the excluded quotient, representative-selection, choice, compactness, host-equality, or ambient-completeness readings can be projected as a NameCert obligation. □

Theorem 6011.3 (HausdorffUniformityCompletion separated handoff)**.** *Every separated endpoint read exported by an accepted* $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{HausdorffUniformity}^{\uparrow} \longmapsto \mathsf{SeparatedReflection}^{\uparrow} \longmapsto \mathsf{UniformCompletion}^{\uparrow} \longmapsto \mathsf{MetricCompletion}^{\uparrow} \mapsto$$

Consequently a completion consumer receives the separated-uniform completion surface only through finite entourage, reflection, completion, window, regular-readback, and seal rows; it receives no quotient completion equality, selected representative, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Project the carrier of Definition 6011.1. The rows $U, H, S, C, M, W, R, E, T, Q, P, N$ are the only rows available to a separated completion consumer.

The route begins with U and H, so the consumer first reads the displayed uniform source and the Hausdorff uniformity classifier. The separated-reflection row S is the only handoff from indistinguishable Cauchy data to a separated completion boundary.

The completion rows C and M are downstream of that separated handoff. The finite-window row W, the regular readback row R, and the real seal row E record the observable endpoint surface after the completion route has already been displayed.

Finally T, Q, P, N transport, replay, package, and name the same finite route. Since no quotient equality, selected representative, choice schedule, host equality, or ambient completeness row occurs in the carrier, no separated endpoint read can export such a datum. □

Falsifiable prediction: Every accepted $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ consumer reads $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{HausdorffUniformity}^{\uparrow}$, $\mathsf{SeparatedReflection}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in the displayed route before citing separated completion uniqueness.

Independence witness: uniformspace, hausdorff-uniformity, separated-reflection, uniformcompletion, metriccompletion, streamname, regseqrat, real: $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the uniform source, Hausdorff classifier, separated reflection, completion rows, finite windows, regular readback, real seal, transport, replay, provenance, and local NameCert row as one packet.

Theorem 6011.4 (HausdorffUniformityCompletion sibling lattice)**.** *Every accepted* $\mathsf{HausdorffUniformityCompletion}$ *read that is consumed as a separated uniform completion must pass through the sibling lattice*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{HausdorffCompletion}^{\uparrow} \longmapsto \mathsf{Completion}^{\uparrow} \longmapsto (\mathsf{CauchyFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow})$$

before any separated-uniform completion consumer reads the packet. The $\mathsf{UniformSpace}^{\uparrow}$ *row supplies the finite entourage basis,* $\mathsf{HausdorffCompletion}^{\uparrow}$ *supplies the separated completion row,* $\mathsf{Completion}^{\uparrow}$ *supplies the completion boundary, and the* $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows supply the observable filter, window, regular-readback, and seal handoff surfaces.*

Proof. Project the carrier of Definition 6011.1. Its uniform source, separated classifier, completion rows, finite windows, regular readback, and real seal are displayed as rows of one finite packet.

The first lattice edge is the $\mathsf{UniformSpace}^{\uparrow}$ source row. A separated completion consumer cannot read a completion endpoint before the finite entourage basis and refinement ledger have been exposed.

The next edges are the separated completion and completion-boundary rows. The Hausdorff side records the separated completion constraint, and the completion row records the boundary at which a completion consumer is allowed to ask for an endpoint read.

Finally the observable side passes through $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows. These rows provide the finite filter observation, window schedule, regular rational readback, and real seal consumed by the downstream packet. Since the carrier has no bypass coordinate, every separated-uniform completion read follows this sibling lattice. □

Theorem 6011.5 (HausdorffUniformityCompletion UniformSpace route)**.** *Every separated completion read exported by an accepted* $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ *carrier first exposes the* $\mathsf{UniformSpace}^{\uparrow}$ *finite entourage source* U*. Only after that source row is displayed may the consumer inspect the* $\mathsf{HausdorffUniformity}^{\uparrow}$ *classifier* H*,* $\mathsf{SeparatedReflection}^{\uparrow}$ *row* S*,* $\mathsf{UniformCompletion}^{\uparrow}$ *row* C*,* $\mathsf{MetricCompletion}^{\uparrow}$ *comparison row* M*,* $\mathsf{StreamName}^{\uparrow}$ *windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, and* $\mathsf{Real}^{\uparrow}$ *seal* E*.*

Proof. Project the carrier of Definition 6011.1. The first coordinate is the uniform-space source U, and it contains the finite entourage basis and refinement ledger for the completion-facing request.

Apply the NameCert obligations of Definition 6011.2. Their uniform-source exactness clause requires every separated completion read to begin with U before the Hausdorff classifier or completion rows are visible.

Now use the separated handoff theorem Definition 6011.3. The displayed route proceeds from $\mathsf{UniformSpace}^{\uparrow}$ through $\mathsf{HausdorffUniformity}^{\uparrow}$, $\mathsf{SeparatedReflection}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ in that order.

The structural rows T, Q, P, N transport, replay, package, and name the same route. Hence no separated completion consumer can bypass the UniformSpace finite entourage source before reading the classifier, reflection, completion, window, readback, and seal rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{HausdorffUniformityCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{HausdorffUniformityCompletion}^{\uparrow}$ packet is generated from UniformSpace, HausdorffUniformity, SeparatedReflection, UniformCompletion, MetricCompletion, StreamName, RegSeqRat, Real, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HausdorffUniformityCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6011.1, Definition 6011.2, Definition 6011.3, and Definition 6011.5 seed the separated-uniform completion boundary over displayed uniformity, reflection, completion, finite-window, regular-readback, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HausdorffUniformityCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, quotient completion equality, selected representatives, countable choice, host equality, open-cover compactness, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exactness, Hausdorff separation handoff, completion readback stability, transport and replay stability, and ledger non-escape for BEDC.Derived.HausdorffUniformityCompletionU

Chapter 6012

A Concrete Naming Certificate for $\mathsf{CompleteOrderedField}^{\uparrow}$

Following Bourbaki and Bishop–Bridges on complete ordered fields, the $\mathsf{CompleteOrderedField}^{\uparrow}$ packet records the traditional complete ordered-field interface as finite BEDC NameCert data rather than as an ambient real line. It sits next to the terminal Real seal of chapter 38, the regular rational sequence route of chapter 301, and the dyadic precision core of chapter 291; it packages the completion obligations those siblings expose without claiming that $\mathsf{Real}^{\uparrow}$ has already discharged them.

Definition 6012.1 (Complete ordered-field carrier)**.** *A* $\mathsf{CompleteOrderedField}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (O, Q, D, E, L, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the ordered-field source row naming addition, multiplication, zero, one, positivity, order comparison, and compatibility only as finite* BHist *data.*
2. *Q is the regular-Cauchy completion interface read through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the terminal* $\mathsf{Real}^{\uparrow}$ *naming seal.*
3. *D is the dyadic density and Archimedean readback row read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
4. *E is the order classifier, comparing accepted ordered-field reads through the displayed algebra, order, dyadic, and regular-Cauchy rows.*
5. *L is the Cauchy-limit seal, naming only those limit requests whose precision, regular sequence, and order rows have all been displayed.*
6. *T is the transport row for componentwise* hsame *comparison of ordered-field, completion, dyadic, classifier, and limit-seal rows.*
7. *H records the local* hsame *ledger for replaying equality-facing component transport without quotient equality.*
8. *C records displayed* Cont *replay from ordered-field source rows through completion, dyadic readback, order classification, and limit sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteOrderedField}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompleteOrderedField}^{\uparrow}}$ *row.*

The classifier of accepted packets compares only $O, Q, D, E, L, T, H, C, P, N$. It has no coordinate for a proof that $\mathsf{Real}^{\uparrow}$ *is complete, for Dedekind-cut completeness, for quotient equality of real names, for a selected limit operator, or for a host complete ordered field.*

Lean4 ✓ `BEDC.Derived.CompleteOrderedFieldUp.CompleteOrderedFieldTasteGate single carrier alignment`

Theorem 6012.2 (CompleteOrderedField sibling route)**.** *Every accepted* $\mathsf{CompleteOrderedField}^{\uparrow}$ *carrier routes its Real-facing completion interface through the siblings already named in the L10 completion lattice. The completion read Q is a consumer of chapter 301 before it reaches chapter 38; the dyadic density and Archimedean readback row D is a consumer of chapter 291; and the order classifier E may compare only those displayed rows together with the ordered-field source O and Cauchy-limit seal L.*

Proof. Project the carrier of Definition 6012.1. Its completion-facing coordinates are Q, D, E, L together with structural rows T, H, C, P, N.

The regular-Cauchy row Q supplies the route through RegSeqRat and then Real naming. The dyadic row D supplies the finite density and Archimedean readback data used by that route.

The order classifier E can compare only the displayed ordered-field source O, the regular-Cauchy route Q, the dyadic readback row D, and the Cauchy-limit seal L. Transport and replay preserve those rows through T, H, C, P, N, so the packet supplies a sibling route and not an ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteOrderedField}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompleteOrderedField}^{\uparrow}$ packet is generated from an ordered-field source row, regular-Cauchy completion interface, dyadic density and Archimedean readback, order classifier, Cauchy-limit seal, transport, replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteOrderedField}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6012.1 and Definition 6012.2 seed the finite ordered-field, regular-Cauchy, dyadic-density, order-classifier, and limit-seal surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteOrderedField}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.CompleteOrderedFieldUp.CompleteOrderedFieldTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim that $\mathsf{Real}^{\uparrow}$ is already complete, does not claim Dedekind-cut completeness, and does not identify real names by quotient equality.

Upgrade path: Obligation closure requires checked carrier admission, order-classifier routing, Cauchy-limit sealing, transport, replay, provenance, and local NameCert obligations for BEDC.Derived.CompleteOrderedFieldUp.

Chapter 6013

A Concrete Naming Certificate for $\mathsf{ChebyshevEquioscillation}^{\uparrow}$

Following Chebyshev and de la Vallee Poussin, the $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ packet records the finite alternation certificate used by minimax polynomial approximation on a compact real interval. It consumes the compact interval source of chapter 1591, the continuous-map and sup-norm rows of chapter 3807, the polynomial carrier of chapter 50, the density witness of chapter 1634, and the algebraic separation surface of chapter 1919. The packet names a displayed best-approximation witness; it is not a host proof of compactness, a global function-space quotient, a selected optimizer outside the support ledger, or an ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6013.1 (Chebyshev equioscillation carrier)**.** *A* $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (I, F, \nu, P, S, \Sigma, A, R, H, C, Q, N)$$

with the following displayed rows:

1. *I is the compact interval row, read as a* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source with endpoint order, rational containment, and real-name sealing already displayed.*
2. *F is the continuous-map row on I, read through the same compact-source windows as the* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *packet.*
3. *ν is the requested polynomial degree bound and coefficient-window budget.*
4. *P is the displayed* $\mathsf{Poly}^{\uparrow}$ *row whose finite coefficient ledger has degree at most ν.*
5. *S is the finite alternating support ledger, listing inspected points of I in order together with the endpoint and interior-window data used to compare them.*
6. *Σ is the sign pattern row, recording alternating signs along S.*
7. *A is the signed extremal-error ledger, saying that the inspected errors $F - P$ reach the same displayed sup-norm magnitude with the signs in Σ.*
8. *R is the minimax refusal row, recording that every admissible lower-degree perturbation read by the packet fails to reduce the displayed sup-norm bound on the same support windows.*
9. *H records componentwise* hsame *transport for the interval, continuous-map, degree, polynomial, support, sign, extremal-error, refusal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a minimax request through $I, F, \nu, P, S, \Sigma, A, R, H, Q, N$.*

11. *Q is the* Pkg *provenance over* ChebyshevEquioscillation$^{\uparrow}$, WeierstrassApproximation$^{\uparrow}$, StoneWeierstrass$^{\uparrow}$, Poly$^{\uparrow}$, ContinuousSupNorm$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{ChebyshevEquioscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a host supremum operator, a quotient function space, an infinite alternation set, a selected optimizer outside P, trichotomy, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.ChebyshevEquioscillationUp.ChebyshevEquioscillationUp`
Lean4 ✓ `BEDC.Derived.ChebyshevEquioscillationUp.ChebyshevEquioscillationTasteGate single carrier alignment`

Theorem 6013.2 (Chebyshev equioscillation NameCert obligations)**.** *For every accepted* ChebyshevEquioscillation$^{\uparrow}$ *carrier*

$$E = (I, F, \nu, P, S, \Sigma, A, R, H, C, Q, N),$$

the local NameCert$_{\text{ChebyshevEquioscillation}^{\uparrow}}$ *surface consists exactly of carrier admission, compact interval source compatibility, continuous-map and sup-norm readback, polynomial degree admission, finite alternating support admission, signed extremal-error exactness, minimax refusal on the displayed support windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Each obligation reads the same* ClosedBoundedInterval$^{\uparrow}$, ContinuousSupNorm$^{\uparrow}$, *and* Poly$^{\uparrow}$ *rows before any minimax conclusion is exposed.*

Proof. Project the carrier of Definition 6013.1. The displayed tuple exposes the interval, continuous-map, degree, polynomial, support, sign, extremal-error, refusal, transport, replay, provenance, and local naming rows.

The compact-source and approximation rows are then forced to share one route. The interval row I supplies the only endpoint and containment data, the continuous-map row F is read through the same source windows, and the polynomial row P supplies the only coefficient ledger admitted by the degree row ν.

The alternation obligations are finite. The support row S orders the inspected points, Σ records the alternating signs, and A binds the displayed errors $F - P$ to one sup-norm magnitude on those windows.

Finally the refusal row R is read only against the same support and coefficient budgets. The structural rows H, C, Q, N transport, replay, source, and name this route, so no host supremum, quotient function space, infinite support, choice-selected optimizer, trichotomy, countable choice, or ambient completeness coordinate can enter the local certificate. □

Theorem 6013.3 (Chebyshev equioscillation alternation window)**.** *Every alternation read of an accepted* ChebyshevEquioscillation$^{\uparrow}$ *carrier factors through the finite route*

$$I, F, \nu, P \longmapsto S, \Sigma, A \longmapsto R, H, C, Q, N.$$

Thus the equioscillation claim is witnessed by the displayed support order, alternating sign ledger, and signed extremal-error ledger, not by an ambient compactness theorem or an uninspected function-space norm.

Proof. Begin with the carrier inventory. The only interval and function data available before the alternation rows are I, F, ν, P.

Apply Definition 6013.2. It requires the support, sign, and extremal-error rows to read the same compact interval and polynomial rows that were admitted by the carrier.

Consequently an alternation consumer can inspect only S, Σ, A after the source rows are fixed. The refusal, transport, replay, provenance, and local name rows then package that same finite route, and no additional compactness or host norm coordinate is present. □

Theorem 6013.4 (Chebyshev equioscillation minimax handoff)**.** *Every minimax-polynomial consumer of an accepted* $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ *packet receives the* $\mathsf{Poly}^{\uparrow}$ *row* P*, the* $\mathsf{ContinuousSupNorm}^{\uparrow}$ *extremal-error ledger* A*, and the finite refusal row* R *only after the compact interval row* I*, continuous-map row* F*, degree row* ν*, support ledger* S*, and sign pattern* Σ *have been displayed. The handoff exports a finite best-approximation certificate, not a density-only theorem, a selected host optimizer, or a completed quotient function space.*

Proof. Project the carrier and use Definition 6013.3. The source interval, continuous map, degree row, and polynomial row determine the finite comparison surface.

The support, sign, and extremal-error rows then expose the equioscillation evidence. Since A is tied to the same $\mathsf{ContinuousSupNorm}^{\uparrow}$ readback route, the minimax consumer receives a displayed norm-facing ledger rather than a host supremum.

The refusal row R records the finite perturbation boundary on those same support windows. The remaining structural rows only transport, replay, source, and name the packet, so the handoff cannot export a density-only statement or a completed function-space object. □

Theorem 6013.5 (Chebyshev equioscillation public export)**.** *The public surface exported by an accepted* $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ *carrier is exactly the finite compact-source, continuous-map, degree-bound, polynomial, alternating-support, signed-error, minimax-refusal, transport, replay, provenance, and local naming route displayed in Definition 6013.1. Downstream approximation chapters may consume this surface as a best-approximation certificate over* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *rows, while* $\mathsf{WeierstrassApproximation}^{\uparrow}$ *and* $\mathsf{StoneWeierstrass}^{\uparrow}$ *remain density and separation sources rather than replacements for the alternation witness.*

Proof. Use Definition 6013.2. It lists the complete local NameCert surface and ties each row to the same compact interval, continuous map, and polynomial source.

Use Definition 6013.4 for the consumer boundary. It identifies the rows that a minimax consumer can read before any best-approximation conclusion is exposed.

The provenance row records the relationship with the density and separation chapters, but no carrier coordinate lets those chapters replace the displayed alternation witness. Therefore the public export is exactly the finite route named by the carrier. □

Falsifiable prediction: Every accepted $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ consumer that claims a minimax polynomial witness must display the compact interval row, continuous-map row, degree row, polynomial row, alternating support, sign pattern, signed extremal-error ledger, minimax refusal row, hsame transport, Cont replay, Pkg provenance, and local NameCert before the best-approximation conclusion is inspected.

Independence witness: weierstrass-approximation, stone-weierstrass, polynomial, continuous-supnorm, closed-bounded-interval: $\mathsf{ChebyshevEquioscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a finite alternating support, signed extremal-error ledger, and minimax refusal row over one compact interval source.

Closure status for $\mathsf{NameCert}_{\mathsf{ChebyshevEquioscillation}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChebyshevEquioscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6013.1, Definition 6013.2, Definition 6013.3, and Definition 6013.4 seed the finite equioscillation carrier, local obligation surface, alternation window, and minimax handoff for compact real intervals.

Formal verification: FormalStatus(NameCert$_{\text{ChebyshevEquioscillation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host compactness, quotient function spaces, selected optimizers, infinite alternation sets, trichotomy, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, compact interval compatibility, continuous-map and sup-norm readback, polynomial degree admission, finite alternating support, signed extremal-error exactness, minimax refusal, transport replay, provenance, and local naming for BEDC.Derived.ChebyshevEquioscillationUp.

Constructive story: A ChebyshevEquioscillation$^{\uparrow}$ packet is generated from a compact interval row, continuous-map row, degree-bound row, displayed polynomial row, finite alternating support, sign pattern, signed extremal-error ledger, minimax refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

CauchyPasting$^{\uparrow}$ consumes the compatible gluing route of chapter 5899 and forwards pasted regular-readback rows to chapter 6003.

Chapter 6014

A Concrete Naming Certificate for $\mathsf{CauchyPasting}^{\uparrow}$

Following Bishop and Bridges, compatible local Cauchy data may be pasted only through explicit moduli, overlap witnesses, and regular readback rows before a real-name seal is exposed. The $\mathsf{CauchyPasting}^{\uparrow}$ packet records that compatible-local-Cauchy route as finite NameCert data for the L10 Real-and-Completion spine. It consumes the $\mathsf{RegularCauchyGluing}^{\uparrow}$ route of chapter 5899, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the $\mathsf{StreamName}^{\uparrow}$ finite-window interface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It is also a finite pasted-window source for the density-net consumer of chapter 6003. It names a finite pasting carrier, not quotient stream equality, selected representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6014.1 (Cauchy pasting carrier)**.** *A* $\mathsf{CauchyPasting}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, D, S, R, E, H, C, K, N)$$

with the following displayed rows:

1. *G is the accepted* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *source row for the compatible local Cauchy windows.*
2. *D is the shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the local windows on their displayed overlap.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row recording exactly the finite observations read by the pasting consumer.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained after G, D, S have been displayed.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the regular readback has been displayed.*
6. *H records componentwise* hsame *transport for source, tolerance, window, readback, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the same pasting route.*
8. *K is the* Pkg *provenance over* $\mathsf{CauchyPasting}^{\uparrow}$, $\mathsf{RegularCauchyGluing}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyPasting}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, D, S, R, E, H, C, K, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, or a general completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 6014.2 (Cauchy pasting NameCert obligations)**.** *For every accepted* $\mathsf{CauchyPasting}^{\uparrow}$ *carrier*

$$P = (G, D, S, R, E, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyPasting}^{\uparrow}}$ *surface consists exactly of carrier admission for the compatible* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *source row, shared* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance display, finite* $\mathsf{StreamName}^{\uparrow}$ *window display,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No pasting obligation is stored outside those finite rows.*

Proof. Project the carrier of Definition 6014.1. The visible route is exhausted by $G, D, S, R, E, H, C, K, N$, so carrier admission and all pasting obligations must be read from those rows.

The source row G supplies the compatible local Cauchy gluing surface. The tolerance row D and window row S record the finite overlap data read by the consumer, while R is the only regular rational readback row downstream of that finite data.

The terminal seal E is exposed only after G, D, S, R have been displayed. The rows H, C, K, N then transport, replay, source, and name the same route.

Since the carrier has no further coordinate, quotient stream equality, selected representatives, countable choice, host equality, and ambient completeness cannot be obligations of the local certificate. □

Theorem 6014.3 (Cauchy pasting Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyPasting}^{\uparrow}$ *packet reaches the seal row E only through the route*

$$G, D, S \longmapsto R \longmapsto E \longmapsto H, C, K, N.$$

In particular, the consumer inherits the pasting nonescape boundary of Definition 5899.13: it cannot export quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness as a coordinate of the accepted pasting read.*

Proof. Start with a $\mathsf{Real}^{\uparrow}$-facing read of the accepted packet. By Definition 6014.1, the only seal coordinate is E, and it is downstream of the source, tolerance, window, and regular readback rows.

Apply Definition 6014.2. The local certificate must expose G, D, S, R before the seal and must preserve the same transport, replay, provenance, and naming rows H, C, K, N.

A completion-facing density read may then pass from the pasted G, D, S, R, E route into $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$. That consumer supplies finite dyadic density rows and still adds no quotient streams, selected limits, or ambient completeness.

Finally use Definition 5899.13 on the source row G. The gluing source cannot contribute quotient stream equality, selected representatives, host equality, choice, or ambient completeness, and the pasting carrier adds no coordinate of those kinds. □

Theorem 6014.4 (Cauchy pasting common-tail gluing)**.** *Let*

$$P = (G, D, S, R, E, H, C, K, N)$$

be an accepted $\mathsf{CauchyPasting}^{\uparrow}$ *packet. If two compatible local Cauchy windows are read by the same pasting request, then their common-tail comparison is displayed through the finite route*

$G, D, S, R, E, H, C, K, N$ before any $\mathsf{Real}^{\uparrow}$ consumer reads the seal row E. The downstream finite-density route may reuse that common tail as a displayed local gluing witness, but it receives no ambient $\mathsf{Real}^{\uparrow}$ completeness theorem from the pasting packet.

Proof. Apply the NameCert obligations of Definition 6014.2. The compatible local Cauchy windows are admitted only through the source row G, the shared tolerance row D, and the finite StreamName window row S.

The regular readback row R records the common-tail comparison attached to those displayed windows. By Definition 6014.3, the seal row E is not reachable until G, D, S, R have already been displayed.

The rows H, C, K, N then transport, replay, source, and locally name the same finite route. A finite-density consumer can read this route as a local gluing witness, but no row adds quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness. □

Theorem 6014.5 (Cauchy pasting density-net forward link)**.** *Every $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ consumer of an accepted $\mathsf{CauchyPasting}^{\uparrow}$ packet must read the displayed route*

$$G, D, S, R, E, H, C, K, N$$

before using the density net of chapter 6003. The density-net read therefore receives the compatible gluing source, dyadic tolerance ledger, finite StreamName window, regular readback, RealUp seal, transport, replay, provenance, and local naming rows as prior data. It cannot replace that route by quotient stream equality, selected representatives, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Proof. Apply Definition 6014.4. The common-tail witness is already displayed through $G, D, S, R, E, H, C, K, N$ before any $\mathsf{Real}^{\uparrow}$ consumer reads the seal row.

The finite density-net consumer cited in chapter 6003 can therefore use the pasted-window route only after the compatible gluing source, tolerance row, finite window, regular readback, and seal have been exposed. The structural rows H, C, K, N preserve that same route as transported, replayed, sourced, and locally named data.

Since Definition 6014.1 contains no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness, the density-net handoff cannot export any of those excluded objects. □

Theorem 6014.6 (Cauchy pasting metric-completion filter route)**.** *Every completion-facing consumer that reads accepted $\mathsf{CauchyPasting}^{\uparrow}$ rows factors the compatible local-Cauchy source through the $\mathsf{MetricCompletionFilter}^{\uparrow}$ route of chapter 8944 before density-net or $\mathsf{Real}^{\uparrow}$-seal consumption. Thus the consumer first exposes the source gluing row G, shared tolerance row D, StreamName window S, RegSeqRat readback R, and structural rows H, C, K, N as a completion-filter dependency, and only then may pass the displayed route to chapter 6003 or to the terminal seal row E.*

Proof. Project the accepted carrier of Definition 6014.1. The compatible local-Cauchy data are present only through $G, D, S, R, E, H, C, K, N$, so a completion-facing consumer has no separate source for a filter limit or a selected representative.

Use the NameCert obligations of Definition 6014.2. They force the gluing source, dyadic tolerance, StreamName window, RegSeqRat readback, transport, replay, provenance, and local naming rows to remain visible before any downstream completion read.

Apply the common-tail gluing theorem Definition 6014.4. It identifies the common-tail witness as displayed local-Cauchy data, which is exactly the kind of finite filter-facing input consumed by the MetricCompletionFilter packet of chapter 8944.

Finally use the Real-seal nonescape and density-net forward link, Definition 6014.3 and Definition 6014.5. The seal row E and the density-net consumer are both downstream of the same displayed completion-filter route, so neither can bypass $\mathsf{MetricCompletionFilter}^{\uparrow}$ by quotient streams, selected representatives, host equality, countable choice, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyPasting}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyPasting}^{\uparrow}$ packet is generated from a compatible RegularCauchyGluing source row, DyadicRatCore tolerance row, StreamName window row, RegSeqRat readback row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyPasting}^{\uparrow}}$, seed)

Scope closed: Definition 6014.1, Definition 6014.2, and Definition 6014.3 seed the finite compatible-local-Cauchy pasting route up to the terminal RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyPasting}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tolerance exactness, StreamName window coverage, RegSeqRat readback, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CauchyPastingUp.

Theorem 6014.7 (Cauchy pasting regular-gluing density route)**.** *Every* $\mathsf{CauchyPasting}^{\uparrow}$ *consumer first reads the* $\mathsf{RegularCauchyGluing}^{\uparrow}$ *source row. It may then feed* $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ *only through the displayed* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *rows of Definition 6014.1; no density-net read may bypass the gluing source, finite window, regular readback, dyadic tolerance ledger, or terminal real seal.*

Proof. Apply the carrier and NameCert obligations of Definition 6014.1 and Definition 6014.2. They put the $\mathsf{RegularCauchyGluing}^{\uparrow}$ source row before the $\mathsf{StreamName}^{\uparrow}$ window, the $\mathsf{RegSeqRat}^{\uparrow}$ readback, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, and the $\mathsf{Real}^{\uparrow}$ seal.

The finite density-net consumer receives only that displayed route. Since the pasting carrier has no coordinate for quotient streams, selected representatives, host equality, countable choice, or ambient completeness, feeding $\mathsf{FiniteCauchyDensityNet}^{\uparrow}$ cannot add any row outside the same finite route. □

Sibling dependency route. The $\mathsf{CauchyPasting}^{\uparrow}$ packet is read as a horizontal dependency across the L10 Real-and-Completion spine: its compatible-gluing source consumes the regular-Cauchy route of chapter 5899, its finite overlap windows and dyadic tolerance ledger feed the density-net consumer of chapter 6003, and its regular readback reaches the $\mathsf{Real}^{\uparrow}$ completion-facing seal only through the displayed $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows already named in Definition 6014.1. Thus a Real-completion consumer receives a pasted-window dependency route, not quotient stream equality, selected representatives, countable choice, or ambient completeness.

Chapter 6015

A Concrete Naming Certificate for $\mathsf{CofinalTailNormalForm}^{\uparrow}$

Bishop–Bridges style regular-Cauchy tail arguments replace eventual agreement talk by displayed cofinal windows, explicit tolerance rows, and regular readbacks before a real-name seal is read. The $\mathsf{CofinalTailNormalForm}^{\uparrow}$ packet records that finite normal-form route for the Real-completeness vision of chapter 9238. It consumes the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ regular readback of chapter 301, the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the paired cofinal-tail route of chapter 920. It names a finite tail normal form, not quotient stream equality, a selected host limit, countable choice, or ambient Real completeness.

Definition 6015.1 (Cofinal tail normal-form carrier)**.** *A* $\mathsf{CofinalTailNormalForm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (A, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the accepted* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *row supplying the paired cofinal-tail agreement source.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *tail-selector window row recording the finite indices inspected by the normal form.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the selected cofinal windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and comparison ledger attached to the same selected tail windows.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after A, S, R, D have been displayed.*
6. *H records componentwise* hsame *transport for agreement, window, readback, tolerance, seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from each tail request through A, S, R, D, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CofinalTailNormalForm}^{\uparrow}$*,* $\mathsf{CofinalTailAgreement}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CofinalTailNormalForm}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, S, R, D, E, H, C, P, N$. It has no coordinate for quotient stream equality, hidden tail selectors, selected representatives, host-limit equality, countable choice, or an ambient completeness principle.

Theorem 6015.2 (Cofinal tail normal-form NameCert obligations)**.** *For every accepted* $\mathsf{CofinalTailNormalForm}^{\uparrow}$ *carrier*

$$T = (A, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CofinalTailNormalForm}^{\uparrow}}$ *surface consists exactly of carrier admission for the cofinal-tail agreement source, finite StreamName tail-selector windows, RegSeqRat readback, DyadicRatCore tolerance display, terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming. Every Real-completeness consumer of the normal form must read those rows before it can use the tail agreement.*

Proof. Project the carrier of Definition 6015.1. The displayed route is $A, S, R, D, E, H, C, P, N$, and the classifier has no coordinate outside that route.

The source row A supplies the cofinal-tail agreement. The finite selector windows S determine which tail observations are inspected, while R and D record the regular readback and dyadic tolerance data attached to those same windows.

The terminal seal E is downstream of A, S, R, D. The rows H, C, P, N then preserve the route under transport, replay, provenance, and local naming.

Because no hidden selector, quotient stream, selected limit, countable-choice schedule, or ambient completeness row occurs in the carrier, none can be part of the local NameCert obligations. □

Theorem 6015.3 (Cofinal tail normal-form shift invariance)**.** *Let*

$$T = (A, S, R, D, E, H, C, P, N)$$

be an accepted $\mathsf{CofinalTailNormalForm}^{\uparrow}$ *carrier. Shifting the displayed finite* $\mathsf{StreamName}^{\uparrow}$ *tail-selector window preserves the same* $\mathsf{CofinalTailAgreement}^{\uparrow}$ *source row A, the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route R, the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance display D, and the same terminal* $\mathsf{Real}^{\uparrow}$ *seal route E, H, C, P, N. The shifted read is therefore another display of the same cofinal-tail normal form, not a hidden selector, quotient stream equality, selected limit, or ambient Real-completeness principle.*

Proof. Project the carrier rows of Definition 6015.1. The only tail-selector coordinate is the displayed $\mathsf{StreamName}^{\uparrow}$ row S, and the agreement source, readback, tolerance, seal, and structural rows are fixed as A, R, D, E, H, C, P, N.

Apply Definition 6015.2. Every Real-completeness consumer must read the cofinal-tail source, the selector window, the regular readback, and the dyadic tolerance display before it can reach the terminal seal.

A shift of the displayed finite selector changes only which finite window of S is inspected. Since the local certificate has no coordinate for a hidden selector or replacement source, the shifted read preserves A, R, D, E, H, C, P, N and remains the same normal-form route. □

Theorem 6015.4 (Cofinal tail normal-form Real-seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CofinalTailNormalForm}^{\uparrow}$ *carrier*

$$T = (A, S, R, D, E, H, C, P, N)$$

reaches the terminal seal row E only through the displayed source, selector, readback, and tolerance route A, S, R, D, and then through H, C, P, N for transport, replay, provenance, and local naming. Therefore the exported Real-seal read contains no quotient stream equality, hidden selector, countable-choice schedule, host-limit equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 6015.1. The terminal $\mathsf{Real}^{\uparrow}$ seal row E is listed downstream of A, S, R, D, so a Real-facing consumer cannot inspect the seal before the cofinal-tail agreement source, displayed selector window, regular readback, and dyadic tolerance display have been exposed.

Apply Definition 6015.2. The local NameCert surface makes exactly the rows $A, S, R, D, E, H, C, P, N$ available to completion and Real-completeness consumers, and it provides no coordinate for quotient stream equality, a hidden selector, host-limit equality, countable choice, or ambient completeness.

Finally use Definition 6015.3. Changing the displayed finite selector window changes only the inspected window of S and preserves the same A, R, D, E, H, C, P, N route. Thus selector-window variation cannot bypass the displayed factorization through A, S, R, D before the terminal Real seal is read. □

Falsifiable prediction: Every $\mathsf{CofinalTailNormalForm}^{\uparrow}$ Real-completeness read factors through one CofinalTailAgreement source row, finite StreamName tail-selector windows, RegSeqRat readback, DyadicRatCore tolerance display, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, cofinaltailagreement: CofinalTailNormalForm is not bijective to any listed sibling because it carries a cofinal-tail agreement source together with the finite selector-window, regular-readback, tolerance, Real-seal, transport, replay, provenance, and naming rows as one normal-form packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CofinalTailNormalForm}^{\uparrow}}$.

Constructive story: A $\mathsf{CofinalTailNormalForm}^{\uparrow}$ packet is generated from a CofinalTailAgreement source row, StreamName tail-selector windows, RegSeqRat readback, DyadicRatCore tolerance display, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CofinalTailNormalForm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6015.1 and Definition 6015.2 seed the finite cofinal-tail normal form consumed by later Real-completeness tail and diagonal-limit routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CofinalTailNormalForm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, hidden tail selectors, selected representatives, host-limit equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-tail agreement source exactness, StreamName selector-window coverage, RegSeqRat readback, DyadicRatCore tolerance exactness, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CofinalTailNormalFormUp.

Lemma 6015.5 (Cofinal tail normal-form diagonal dependency)**.** *Every* $\mathsf{CofinalTailNormalForm}^{\uparrow}$ *consumer first reads the common-tail and finite-window route of chapter 632 before exporting a normal-form row. The diagonal cofinal tail supplies the shared tail window and terminal common-tail descent that this normal-form packet normalizes, so the normal-form read is a finite dependency route rather than a new selected limit, quotient stream equality, or ambient Real-completeness principle.*

Chapter 6016

A Concrete Naming Certificate for $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive compactness, the $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ packet records the finite Lebesgue-number surface for compact metric finite covers as NameCert data. It sits over the compact metric source of chapter 129, the metric distance rows of chapter 57, the finite-cover refinement route of chapter 3430, the compact Lebesgue-number ledger of chapter 4015, the finite-cover radius-stability row of chapter 5203, and the compact uniform-modulus consumer of chapter 2299. The packet exposes a positive radius for one displayed finite cover route; it is not an arbitrary open-cover compactness theorem, a choice-selected subcover, or an ambient completeness claim.

Definition 6016.1 (LebesgueNumberFiniteCover carrier)**.** *A* $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, U, R, F, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the compact metric source row, exposing the metric carrier, compact finite-net witnesses, and distance-read data consumed by the route.*
2. *U is the finite cover row, listing the displayed cover cells, their local metric-containment clauses, and the finite index spine inspected by the packet.*
3. *R is the positive radius row, recording the rational lower bound that is to be read as a Lebesgue number for the displayed finite cover.*
4. *F is the finite refinement and mesh row, recording the mesh budget and every refinement comparison used to pass from compact finite-net data to the cover cells of U.*
5. *E is the cover-exactness ledger, saying that every inspected metric ball of radius below R is routed into a named member of U through the displayed refinement rows.*
6. *H records componentwise* hsame *transport for compact metric, finite cover, positive radius, refinement, exactness, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a compact finite-cover request through*

$$X, U, R, F, E \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* LebesgueNumberFiniteCover$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, FiniteCoverRefinement$^{\uparrow}$, CompactLebesgueNumber$^{\uparrow}$, FiniteCoverRadiusStability$^{\uparrow}$, CompactUniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* NameCert$_{\text{LebesgueNumberFiniteCover}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for an arbitrary open cover, a selected subcover outside U, countable choice, a host-level finite minimum, quotient equality of covers, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LebesgueNumberFiniteCoverUp.LebesgueNumberFiniteCoverTasteGate single carrier alignment`

Theorem 6016.2 (LebesgueNumberFiniteCover NameCert obligations)**.** *For every accepted* LebesgueNumberFinite *carrier*

$$L = (X, U, R, F, E, H, C, P, N),$$

the local NameCert$_{\text{LebesgueNumberFiniteCover}^{\uparrow}}$ *surface consists exactly of carrier admission, compact metric source admission, finite-cover admission, positive radius admission, finite refinement and mesh admission, cover-exactness replay, refinement-stability transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside $X, U, R, F, E, H, C, P, N$.*

Proof. Project the carrier of Definition 6016.1. The public surface contains only the compact metric source, finite cover, positive radius, refinement, exactness, transport, replay, provenance, and naming rows.

The compact source row X supplies the finite metric inspections used by the packet, and the cover row U supplies the only cover cells that may be named. Thus carrier admission and finite-cover exactness are checked against one displayed finite spine.

The radius row R is positive before it is consumed. The refinement and mesh row F records every finite comparison needed to route the compact-source inspections into U, while the exactness ledger E records the containment of every inspected radius ball below R.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the tuple has no coordinate for an arbitrary open cover, selected subcover outside the packet, countable choice, host finite minima, quotient cover equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6016.3 (LebesgueNumberFiniteCover finite-cover handoff)**.** *Every compact-uniform consumer reading an accepted* LebesgueNumberFiniteCover$^{\uparrow}$ *packet receives the radius row R only through the finite route*

$$X, U, F, E \longmapsto R \longmapsto H, C, P, N.$$

The route may be consumed by compact uniform-modulus and finite-cover stability chapters, but it exports no arbitrary open-cover compactness principle and no selected subcover outside the displayed packet.

Proof. Use the NameCert obligations of Definition 6016.2. A consumer must first display the compact metric source X and finite cover U.

The refinement and exactness rows F, E then determine where each inspected metric ball lands. The positive radius R is readable only after those rows have fixed the finite cover route.

Transport and replay through H, C preserve this order, and P, N record provenance and local naming. The handoff therefore supplies a finite Lebesgue-number packet without adding open-cover compactness, a choice-selected subcover, or ambient completion data. □

Falsifiable prediction: Every accepted $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ consumer that claims a Lebesgue-number radius for a compact metric finite cover must display the compact metric source row, finite cover row, positive radius row, finite refinement and mesh row, cover-exactness ledger, hsame transport, Cont replay, Pkg provenance, and local NameCert before the radius is inspected.

Independence witness: compactmetric, metric, finite-cover-refinement, compact-lebesgue-number, compact-uniform-modulus: $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries one displayed compact metric source, one finite cover, one positive radius row, and one exact refinement ledger as a single local NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteCover}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6016.1, Definition 6016.2, and Definition 6016.3 seed the finite Lebesgue-number carrier, local obligation surface, and compact-uniform handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers outside the displayed finite cover, countable choice, host finite minima, quotient cover equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, compact metric source admission, finite-cover exactness, radius positivity, refinement stability, cover-exactness replay, transport, provenance, and local naming for BEDC.Derived.LebesgueNumberFiniteCoverUp.

Constructive story: A $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ packet is generated from a compact metric source row, finite cover row, positive radius row, finite refinement and mesh row, cover-exactness ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6017

A Concrete Naming Certificate for $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$

The $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ packet realizes the phase-distance observer-expansion reading of chapter 9463 as finite NameCert data. It sits between the visual-shadow metric route of chapter 4142, the boundary-shadow route of chapter 3813, the focused expansion ledger of chapter 3486, the observer-expansion certificate of chapter 4167, and the observer-shell packet of chapter 5180. The packet names one finite observer-phase and distance-comparison surface; it is not a host hyperbolic space, a physical cosmology, a global clock, or a completed observer trajectory.

Falsifiable prediction: Every accepted $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ consumer must expose observer-phase rows, distance-read rows, expansion-comparison rows, provenance rows, hsame transport, Cont replay, Pkg support, and local NameCert data before it exports a phase-distance expansion judgment.

Independence witness: hyperbolic-visual-shadow-metric, hyperbolic-boundary-shadow, hyperbolic-expansion-ledger, hyperbolic-observer-expansion, hyperbolic-observer-shell: the visual-shadow metric supplies distance comparison without observer-phase provenance; the boundary-shadow packet supplies boundary routing without expansion comparison; the expansion ledger records growth without phase-distance readback; the observer-expansion packet records shadow expansion without this phase-distance comparison surface; and the observer shell packages surrounding routes without identifying this local carrier.

Definition 6017.1 (PhaseDistanceObserverExpansion carrier)**.** *A* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (O, \Phi, D, E, B, H, C, P, N)$$

with the following displayed rows:

1. *O is the observer-window row, recording the focused history, visible disk window, and local observation budget inspected by the packet.*
2. *Φ is the observer-phase row, read through phase-ledger and disk-coordinate data rather than through a hidden global phase.*
3. *D is the distance-read row, recording the displayed metric, visual-shadow, or boundary-shadow comparison used by the local read.*

4. *E is the expansion-comparison row, saying which earlier and later observer windows are compared and how the displayed distance-read changes along the local* Cont *route.*
5. *B is the boundary and refusal row, recording that the expansion is a records-side readback and not a physical cosmology, global clock, host hyperbolic model, or completed observer trajectory.*
6. *H records componentwise* hsame *transport for observer-window, phase, distance, expansion, boundary-refusal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an observer-window request through*

$$O, \Phi, D, E, B \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, $\mathsf{HyperbolicVisualShadowMetric}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicExpansionLedger}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, HyperbolicObserverShe BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host hyperbolic manifold, a global observer, a universal time coordinate, a physical expansion constant, quotient equality of observer histories, or an unlogged boundary completion.

Theorem 6017.2 (PhaseDistanceObserverExpansion NameCert obligations)**.** *For every accepted* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *carrier*

$$Q = (O, \Phi, D, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}}$ *surface consists exactly of observer-window admission, observer-phase admission, distance-read admission, expansion-comparison admission, boundary-refusal coverage, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No NameCert obligation is stored outside $O, \Phi, D, E, B, H, C, P, N$.*

Proof. Project the carrier of Definition 6017.1. The only public rows are the observer window, observer phase, distance read, expansion comparison, boundary refusal, transport, replay, provenance, and local name rows.

Rows O and Φ fix the observer-side and phase-side data. The phase read is therefore tied to a focused finite ledger rather than to an unlogged global phase surface.

Rows D and E supply the comparison. The distance read is the displayed metric or visual-shadow row, and the expansion row records the before-and-after comparison along a local Cont route.

Row B keeps the packet at the representation layer by refusing host hyperbolic geometry, physical cosmology, global clocks, and completed observer trajectories. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 6017.3 (PhaseDistanceObserverExpansion comparison handoff)**.** *Every phase-distance expansion read from an accepted* $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ *packet factors through*

$$O, \Phi, D, E, B \longmapsto H, C, P, N.$$

The handoff exports a finite observer-phase, distance-read, and expansion-comparison certificate, not an ambient hyperbolic model, physical expansion law, global observer, or completed boundary object.

Proof. Use the obligations of Definition 6017.2. An expansion consumer first receives the observer-window and phase rows.

The distance row and expansion-comparison row then determine what can be compared. Since both are displayed rows of the carrier, the comparison remains local to the inspected observer-window route.

The refusal row blocks readings that require a host hyperbolic model, a physical cosmology, a global clock, or completed observer trajectory. Transport, replay, provenance, and naming only preserve that finite packet, so the handoff exports exactly the listed finite comparison surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6017.1, Definition 6017.2, and Definition 6017.3 seed the phase-distance observer-expansion carrier, local obligation surface, and comparison handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic geometry, physical cosmology, global observers, universal clocks, quotient observer-history equality, completed boundary objects, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode observer-window admission, phase admission, distance-read admission, expansion-comparison admission, boundary-refusal coverage, transport, replay, provenance, and local naming for BEDC.Derived.PhaseDistanceObserverExpansionUp.

Constructive story: A $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$ packet is generated from observer-window, observer-phase, distance-read, expansion-comparison, boundary-refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6018

A Concrete Naming Certificate for $\mathsf{PushdownAutomaton}^{\uparrow}$

Following Hopcroft–Ullman and Sipser, the $\mathsf{PushdownAutomaton}^{\uparrow}$ packet records a finite-control recognizer with a visible stack ledger as displayed BEDC data. It sits between the finite run surface of chapter 2364, the accepted-word surface of chapter 351, the finite derivation surface of chapter 356, and the bounded machine-trace surface of chapter 280.

The packet names one finite state carrier, one input word, one stack alphabet, one transition ledger, one finite run ledger, and one acceptance readback. It does not identify languages extensionally, select an infinite parse tree, or export an unbounded recognizer trace.

Definition 6018.1 (PushdownAutomaton carrier)**.** *A* $\mathsf{PushdownAutomaton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F, w, \rho, \sigma, e, \chi, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite control-state carrier row;*
2. *Σ is the finite input alphabet row;*
3. *Γ is the finite stack alphabet row;*
4. *δ is the finite transition ledger, whose displayed entries read a state row, an optional input-symbol row, and the top stack-symbol row, then return the next state row and the finite stack-word replacement row;*
5. *q_0 is the start-state row inside Q;*
6. *Z_0 is the initial stack-symbol row inside Γ;*
7. *F is the accepting-state row inside Q;*
8. *w is the finite input word as a $\mathsf{List}^{\uparrow}$ row over Σ;*
9. *ρ is the finite run ledger, beginning at (q_0, Z_0) and consuming w through displayed transition rows;*
10. *σ is the stack-history ledger whose entries are finite $\mathsf{List}^{\uparrow}$ rows over Γ;*
11. *e is the endpoint row of the run ledger;*

12. *χ is the acceptance readback row, by final state or by displayed empty-stack condition as named in the packet;*
13. *H stores componentwise* hsame *transports for state, input, stack, transition, run, stack-history, endpoint, and classifier rows;*
14. *C stores the* Cont *replay routes for reading input symbols, stack-top rows, transition entries, and the finite run ledger;*
15. *P stores the* Pkg *provenance naming* $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, $\mathsf{TuringMachine}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
16. *N stores the local* $\mathsf{NameCert}_{\mathsf{PushdownAutomaton}^{\uparrow}}$ *rows.*

The carrier predicate is $\mathsf{PushdownAutomatonPacket}(A)$. *It exports no host language set, grammar quotient, selected parse tree, infinite branch choice, or unbounded machine execution.*

Lean4 ✓ `BEDC.Derived.PushdownAutomatonUp.TasteGate.PushdownAutomatonTasteGate single carrier alignment`

Theorem 6018.2 (PushdownAutomaton NameCert obligations)**.** *For every accepted* $\mathsf{PushdownAutomaton}^{\uparrow}$ *carrier*

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F, w, \rho, \sigma, e, \chi, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PushdownAutomaton}^{\uparrow}}$ *surface consists of carrier admission, finite transition replay, stack-row stability, endpoint classifier stability, componentwise* hsame *transport,* Cont *replay, and* Pkg *provenance for the rows in Definition 6018.1. A consumer may project the finite control, alphabet, stack, transition, input, run, stack-history, endpoint, and local naming rows, but it obtains no regular-language quotient, context-free grammar quotient, selected parse tree, or unbounded* $\mathsf{TuringMachine}^{\uparrow}$ *trace.*

Proof. Project the accepted carrier A. The rows $Q, \Sigma, \Gamma, \delta, q_0, Z_0, F$ name the finite control and stack presentation, while w, ρ, σ, e, χ name the inspected input, run ledger, stack ledger, endpoint, and acceptance readback.

Finite transition replay is obtained by reading ρ one displayed step at a time. Each step consults the current state row, the next consumed input row when present, the top stack-symbol row, and one transition entry of δ.

The structural rows H, C, P, N transport and replay exactly these coordinates. Since the carrier has no coordinate for a host language set, grammar quotient, selected parse tree, or unbounded machine trace, no consumer can extract such data from the local NameCert surface. □

Theorem 6018.3 (PushdownAutomaton stack-transition stability)**.** *Let*

$$A = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F, w, \rho, \sigma, e, \chi, H, C, P, N)$$

be an accepted $\mathsf{PushdownAutomaton}^{\uparrow}$ *carrier. Every displayed transition step in ρ transports the current stack row in σ to the next stack row by popping the displayed top symbol and pushing the finite replacement* $\mathsf{List}^{\uparrow}$ *row named by the corresponding transition entry. The transported step preserves the input cursor, endpoint readback,* hsame *rows,* Cont *routes, and* Pkg *provenance, and it exports no hidden stack contents, epsilon-closure quotient, or infinite run branch.*

Proof. Read one step of the finite run ledger. The step exposes the current control state, the optional consumed input symbol, the current stack row, and the top-stack symbol read by the transition entry.

Use the displayed transition entry of δ to form the next state and the finite replacement stack word. The next stack ledger entry in σ is obtained by replacing the visible top symbol with that finite $\mathsf{List}^{\uparrow}$ word and leaving the unreduced stack suffix as the displayed remainder row.

The componentwise hsame rows transport the state, input cursor, stack row, transition entry, and endpoint readback. The Cont route then passes the remaining finite run ledger to the next step, and the Pkg row records the same finite-recognizer, grammar, and machine-boundary provenance.

No other stack data is available. In particular, the packet contains no epsilon-closure quotient, selected infinite branch, hidden stack store, or unbounded machine trace. □

Closure status for $\mathsf{NameCert}_{\mathsf{PushdownAutomaton}^{\uparrow}}$.

Constructive story: A $\mathsf{PushdownAutomaton}^{\uparrow}$ packet is generated from finite control rows, input alphabet rows, stack alphabet rows, a transition ledger, start and initial-stack rows, accepting rows, a finite input word, a finite run ledger, a stack-history ledger, an endpoint readback, componentwise hsame transports, Cont replay routes, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PushdownAutomaton}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6018.1, Definition 6018.2, and Definition 6018.3 seed the finite stack-recognizer carrier, local NameCert surface, and stack-transition stability boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PushdownAutomaton}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, equivalence with all context-free grammars, language-set semantics, determinization, selected parse trees, infinite branch choice, or unbounded Turing-machine recognition.

Upgrade path: Obligation closure requires checked carrier admission, transition replay, stack-row stability, acceptance readback, and ledger non-escape for BEDC.Derived.PushdownAutomatonUp.

Chapter 6019

A Concrete Naming Certificate for $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$

Following Bishop and Oxtoby, the $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$ packet records the complete-metric dense-open intersection route as finite NameCert data rather than as an ambient theorem about all complete metric spaces. It sits over the dense-open schedule of chapter 1542, the complete-metric source of chapter 128, the metric carrier of chapter 57, the finite stream windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real seal of chapter 38.

Definition 6019.1 (BaireCategoryCompleteMetric carrier)**.** *A* $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, K, M, D, S, R, Y, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireCategory}^{\uparrow}$ *dense-open schedule row, listing the finite dense-open requests and the predecessor-ball refinements visible to the packet.*
2. *K is the* $\mathsf{CompleteMetric}^{\uparrow}$ *source row, exposing the displayed Cauchy-thread service and limit handoff used by the dense-open route.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, naming the finite distance, ball, and comparison observations consumed by B and K.*
4. *D is the dense-open schedule ledger, recording the finite list of open rows, density witnesses, successor-ball choices, and requested indices.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the schedule, centers, radii, and requested tolerances are inspected.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational center and radius data visible before the terminal seal.*
7. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, storing the displayed dyadic radii and shrinkage comparisons used by the regular rational readback.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the dense-open schedule, complete-metric source, metric carrier, finite windows, regular-rational readback, and dyadic tolerance ledger have been displayed.*

9. *H records componentwise* hsame *transport for the dense-open schedule, complete-metric source, metric carrier, stream windows, regular-rational readback, dyadic tolerance ledger, real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same dense-open, complete-metric, metric, window, readback, tolerance, and seal route.*
11. *P is the* Pkg *provenance over* $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$, $\mathsf{BaireCategory}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BaireCategoryCompleteMetric}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, K, M, D, S, R, Y, E, H, C, P, N$. It has no coordinate for countable choice, a host intersection, an open-cover compactness theorem, a selected limit point, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a standard bridge datum.*

Lean4 ✓ `BEDC.Derived.BaireCategoryCompleteMetricUp.BaireCategoryCompleteMetricTasteGate single carrier alignment`

Theorem 6019.2 (BaireCategoryCompleteMetric NameCert obligations)**.** *For every accepted* $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$ *carrier*

$$Q = (B, K, M, D, S, R, Y, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BaireCategoryCompleteMetric}^{\uparrow}}$ *surface exposes exactly the dense-open schedule rows, complete-metric source row, metric carrier row, finite* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row. Every complete-metric Baire read is routed through those rows, and no obligation is stored outside them.*

Lean4 ✓ `BEDC.Derived.BaireCategoryCompleteMetricUp.BaireCategoryCompleteMetricNameCert obligations`

Proof. Project the accepted carrier of Definition 6019.1. The rows B, K, M, D, S, R, Y, E are the only analytic rows available to a consumer: Baire schedule, complete-metric service, metric comparison, dense-open ledger, finite windows, regular rational readback, dyadic tolerances, and terminal real seal.

The dense-open content is read first through B and D. Each request exposes only finite open rows, density witnesses, and successor-ball choices; the complete-metric row K then carries the displayed Cauchy thread built from those finite refinements.

The metric and numerical observations are constrained by M, S, R, Y, E. The metric carrier supplies distance and ball comparisons, the stream row supplies finite inspection windows, the dyadic ledger supplies radius comparisons, the regular-rational row reads center and radius data, and the real seal is reached only after those rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no coordinate for countable choice, host intersections, open-cover compactness, selected limit points, ambient real completeness, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 6019.3 (BaireCategoryCompleteMetric dense-open handoff)**.** *Let $Q = (B, K, M, D, S, R, Y, E, H, C, P,$ be an accepted* $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$ *carrier. Any downstream constructive Baire-category or Banach-style consumer that reads a dense-open intersection through a complete metric source must read the* $\mathsf{BaireCategory}^{\uparrow}$ *schedule B, dense-open ledger D,* $\mathsf{CompleteMetric}^{\uparrow}$ *source K,* $\mathsf{Met}^{\uparrow}$

carrier M, $\mathsf{StreamName}^{\uparrow}$ *finite windows S,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger Y, and* $\mathsf{Real}^{\uparrow}$ *seal E, transported and replayed only by H, C, P, N.*

The handoff exports no countable-choice selector, host intersection, selected limit point, open-cover compactness theorem, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or additional completion carrier.*

Proof. Start with the dense-open rows B and D. They display the finite schedule and successor-ball refinements already present in the packet, so a consumer cannot replace them by an external countable family or host intersection.

Next read the complete-metric and metric rows K and M. The complete-metric source supplies only the Cauchy-thread service recorded in the packet, while the metric carrier supplies the finite distance and ball comparisons used by that thread.

Route the numerical part through S, R, Y, E. The stream windows expose the requested finite observations, the dyadic ledger records the tolerance shrinkage, the regular-rational readback carries the displayed center and radius data, and the terminal real seal is reached only from those finite rows.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Coordinate exhaustion then gives the stated handoff boundary: every downstream consumer receives the displayed finite packet and nothing beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{BaireCategoryCompleteMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{BaireCategoryCompleteMetric}^{\uparrow}$ packet is generated from a BaireCategory dense-open schedule, a CompleteMetric source row, a Metric carrier row, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BaireCategoryCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6019.1, Definition 6019.2, and Definition 6019.3 seed the complete-metric dense-open intersection packet over the displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BaireCategoryCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close countable choice, host intersections, open-cover compactness, selected limit points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-open schedule exposure, complete-metric source exactness, metric row compatibility, finite-window readback, dyadic tolerance exactness, regular-rational readback, real sealing, and nonescape for BEDC.Derived.BaireCategoryCompleteMetricUp.

Chapter 6020

A Concrete Naming Certificate for $\mathsf{ContractiveCauchyIterator}^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the $\mathsf{ContractiveCauchyIterator}^{\uparrow}$ packet isolates the finite Picard-iteration surface that hands a Cauchy modulus to completion-facing consumers before any fixed-point endpoint is read. It sits between the metric carrier of chapter 57, the complete-metric service of chapter 128, the map-level contraction packet of chapter 1418, the one-step Picard contraction route of chapter 395, the Banach endpoint consumer of chapter 396, the dyadic budget ledger of chapter 291, and the real seal boundary of chapter 38. The packet names the finite iterate windows and the Cauchy-modulus handoff; it does not claim a fixed point.

Definition 6020.1 (ContractiveCauchyIterator carrier)**.** *A* $\mathsf{ContractiveCauchyIterator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, T, B, I, W, Q, D, E, H, C, P_0, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, including the carried point histories and distance-read surface used by the iterator.*
2. *T is the* $\mathsf{ContractionMapping}^{\uparrow}$ *self-map row whose graph and contraction comparison are available before any iterate window is opened.*
3. *B is the rational contraction-bound row below one, read through the same dyadic and metric budget as the map row.*
4. *I is the initial-row data, naming the starting point and the first graph read used to begin the Picard schedule.*
5. *W is the finite iterate-window ledger. It records displayed windows of $x_0, T(x_0), \ldots, T^n(x_0)$ and the adjacent-distance comparisons visible inside those windows.*
6. *Q is the Cauchy-modulus ledger. For each displayed tolerance request it records the finite threshold extracted from B, I, W.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget row used to compare requested precision, contraction powers, and tail bounds.*
8. *E is the completion and* $\mathsf{Real}^{\uparrow}$ *seal boundary. It is a boundary row only: no endpoint seal may be read until M, T, B, I, W, Q, D have been displayed.*

9. *H records componentwise* hsame *transport for the metric source, contraction map, bound, initial row, iterate windows, Cauchy ledger, dyadic tolerance budget, boundary seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through*

$$M, T, B, I, W, Q, D \longmapsto E, H, P_0, N.$$

11. *P_0 is the* Pkg *provenance over* ContractiveCauchyIterator$^{\uparrow}$, Met$^{\uparrow}$, CompleteMetric$^{\uparrow}$, ContractionMapping$^{\uparrow}$, PicardContraction$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{ContractiveCauchyIterator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, T, B, I, W, Q, D, E, H, C, P_0, N$. It has no coordinate for a fixed point, endpoint uniqueness, quotient metric equality, choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.ContractiveCauchyIteratorUp.ContractiveCauchyIteratorTasteGate single carrier alignment`

Theorem 6020.2 (ContractiveCauchyIterator NameCert obligations)**.** *For every accepted* ContractiveCauchyIterato *carrier*

$$P = (M, T, B, I, W, Q, D, E, H, C, P_0, N),$$

the local NameCert$_{\text{ContractiveCauchyIterator}^{\uparrow}}$ *surface consists exactly of five finite obligation families:*

1. *carrier admission for the metric source, contraction map, rational bound, initial row, iterate windows, Cauchy-modulus ledger, dyadic tolerance budget, completion boundary, transport, replay, provenance, and local naming rows;*

2. *contraction-budget visibility, requiring T and B to be read against M before any Picard window is consumed;*

3. *iterate-window stability, requiring every displayed adjacent-window comparison in W to replay the same contraction comparison and initial row I;*

4. *Cauchy-modulus exactness, requiring each tolerance request to read D and then Q as a finite threshold ledger over the displayed windows;*

5. *endpoint nonescape, requiring every* CompleteMetric$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, *or* Real$^{\uparrow}$ *consumer to pass through M, T, B, I, W, Q, D before the boundary row E is exposed.*

No obligation is stored outside those rows.

Proof. Carrier admission is projection from Definition 6020.1. The accepted packet displays exactly the metric source, contraction map, rational bound, initial row, iterate-window ledger, Cauchy-modulus ledger, dyadic tolerance budget, boundary seal, and structural rows.

The contraction budget is read before iteration. The rows T and B are tied to the metric source row M, so an iterate comparison that does not pass through the displayed metric and bound has no coordinate in the carrier.

Window stability is finite replay. Each row of W opens the initial row I and the graph reads of T, then records only adjacent comparisons certified by the same contraction budget.

The Cauchy-modulus obligation is the handoff row. A tolerance request first reads the dyadic budget D and then the threshold ledger Q, so downstream completion consumers receive a finite modulus rather than a selected endpoint.

Finally E, H, C, P_0, N do not add an analytic endpoint. They expose the boundary, transport the same rows, replay the same route, record provenance, and name the local packet. □

Theorem 6020.3 (ContractiveCauchyIterator completion handoff)**.** *Every* $\mathsf{CompleteMetric}^{\uparrow}$ *or* $\mathsf{BanachFixedPoint}^{\uparrow}$ *consumer of an accepted* $\mathsf{ContractiveCauchyIterator}^{\uparrow}$ *carrier must read the finite rows*

$$M, T, B, I, W, Q, D$$

before any endpoint boundary row E *is exposed. The handoff supplies a metric source, a contractive self-map, a displayed rational bound below one, an initial row, finite Picard iterate windows, a Cauchy-modulus ledger, and a dyadic tolerance budget. It does not supply a fixed point, endpoint uniqueness, quotient metric equality, choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The rows available before the boundary are precisely the metric source M, contraction map T, rational bound B, initial row I, iterate windows W, Cauchy ledger Q, and dyadic tolerance budget D.

A complete-metric consumer first reads the finite Cauchy evidence. The iterate windows expose only finite Picard prefixes, while the dyadic budget and Cauchy ledger convert a tolerance request into a displayed threshold.

A Banach-facing consumer receives the same data. It may use the later completion service or fixed-point route only after this iterator packet has handed off the Cauchy-modulus row, so no endpoint is selected inside the iterator itself.

The boundary row E records where completion and real-seal consumers may continue. Since E is downstream of M, T, B, I, W, Q, D and carries no separate choice or equality coordinate, the theorem exports only the finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContractiveCauchyIterator}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyIterator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6020.1, Definition 6020.2, and Definition 6020.3 seed the finite Picard-iteration Cauchy-modulus handoff before any endpoint seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContractiveCauchyIterator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a fixed point, endpoint uniqueness, quotient metric equality, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-budget visibility, iterate-window stability, Cauchy-modulus exactness, endpoint nonescape, provenance, and local naming for BEDC.Derived.ContractiveCauchyIteratorUp.

Constructive story: A $\mathsf{ContractiveCauchyIterator}^{\uparrow}$ packet is generated from a metric source, contractive self-map, rational bound below one, initial row, finite iterate windows, Cauchy-modulus ledger, dyadic tolerance budget, completion boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6021

A Concrete Naming Certificate for PoincareVisualRadiusLadder$^{\uparrow}$

Following classical Poincare disk geometry, the PoincareVisualRadiusLadder$^{\uparrow}$ packet records a finite visual-radius ladder for observer-side disk readouts. It refines the disk carrier of chapter 377, the metric surface of chapter 3474, the boundary-shadow rows of chapter 3813, the visual metric rows of chapter 4142, the observer-shadow rows of chapter 4154, the finite stream-window interface of chapter 126, the regular-rational readback route of chapter 301, and the real seal of chapter 38. The packet exposes finite disk radii, boundary shadows, visual-metric readbacks, and real seals; it does not assert a completed hyperbolic boundary or a global observer geometry.

Definition 6021.1 (PoincareVisualRadiusLadder carrier)**.** *A* PoincareVisualRadiusLadder$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, R, B, V, O, S, Q, E, T, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the* UnitDisk$^{\uparrow}$ *disk-coordinate row naming the displayed Poincare disk point and its finite admissibility data.*
2. *R is the visual-radius ladder row, a finite ordered ledger of disk radii and radial comparison requests.*
3. *B is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row that records which boundary-shadow windows are visible at the displayed radii.*
4. *V is the* HyperbolicVisualShadowMetric$^{\uparrow}$ *row that reads the boundary shadow through the visual metric at each radius level.*
5. *O is the* HyperbolicObserverShadowMetric$^{\uparrow}$ *row tying the visual-radius readback to the observer-facing shadow metric.*
6. *S is the* StreamName$^{\uparrow}$ *window row, recording the finite stream intervals from which the radius ladder may be replayed.*
7. *Q is the* RegSeqRat$^{\uparrow}$ *rational readback row for the visual-radius and shadow-metric values.*
8. *E is the* Real$^{\uparrow}$ *seal row reached only after S and Q have displayed the requested finite readback.*
9. *T is the finite transport row moving radius levels, boundary-shadow windows, visual-metric rows, observer-shadow rows, stream windows, rational readbacks, and real seals together.*

10. *A is the replay row for radius comparison, shadow inclusion, visual-metric readback, observer-shadow compatibility, and real-seal exposure.*

11. *H records componentwise* hsame *transport for every displayed row.*

12. *C records displayed* Cont *replay through*

$$D, R, B, V, O, S, Q \longmapsto E, T, A, H, P, N.$$

13. *P is the* Pkg *provenance over* PoincareVisualRadiusLadder$^{\uparrow}$, UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBoundarySha HyperbolicVisualShadowMetric$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

14. *N is the local* NameCert$_{\text{PoincareVisualRadiusLadder}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, R, B, V, O, S, Q, E, T, A, H, C, P, N$. It has no coordinate for a completed boundary, ambient hyperbolic manifold, global observer metric, arbitrary real representative, choice, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.PoincareVisualRadiusLadderUp.PoincareVisualRadiusLadderTasteGate single carrier alignment`

Theorem 6021.2 (PoincareVisualRadiusLadder NameCert obligations)**.** *For an accepted* PoincareVisualRadiusLadd *carrier*

$$L = (D, R, B, V, O, S, Q, E, T, A, H, C, P, N),$$

the local NameCert$_{\text{PoincareVisualRadiusLadder}^{\uparrow}}$ *surface consists exactly of carrier admission; disk-radius admissibility for D and R; boundary-shadow visibility for B; visual-metric readback for V; observer-shadow compatibility for O; stream and regular-rational readback through S and Q; real-seal ordering through E; transport and replay stability through T, A, H, C; provenance through P; and local naming through N.*

Proof. Carrier admission is projection from Definition 6021.1. The accepted packet displays disk coordinates, a finite radius ladder, boundary-shadow rows, visual-metric rows, observer-shadow rows, stream windows, rational readbacks, the real seal, and structural rows.

Disk-radius admissibility is finite. The row D supplies the disk surface, while R lists only the radius levels visible to the packet.

The shadow and metric obligations are read in order. Boundary-shadow windows come from B, visual-metric values from V, and observer-facing compatibility from O.

The readback rows then force the seal route. Stream windows S and regular-rational data Q must be displayed before the real seal E can be inspected.

Finally T, A, H, C, P, N transport, replay, source, and name the same finite ladder, so no completed boundary, global geometry, choice principle, or bridge credential is added. □

Theorem 6021.3 (PoincareVisualRadiusLadder shadow handoff)**.** *Every boundary-shadow, visual-metric, observer-shadow, regular-rational, or real-name consumer of an accepted* PoincareVisualRadiusLadder$^{\uparrow}$ *carrier must factor through the finite route*

UnitDisk$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, HyperbolicBoundaryShadow$^{\uparrow}$ $\longmapsto$ HyperbolicVisualShadowMetric$^{\uparrow}$, HyperbolicObserve

At each displayed radius level, the consumer reads disk data, boundary-shadow windows, visual-metric values, observer-shadow compatibility, stream windows, rational readbacks, and only then the real seal. The route exports no completed boundary, global observer metric, arbitrary real representative, or ambient hyperbolic manifold.

Proof. Start with a consumer request at a displayed radius level. The disk row D and radius row R determine the finite Poincare disk surface and the visual-radius level that may be inspected.

The boundary and metric rows are then forced. The row B supplies the boundary-shadow window, V reads it through the visual metric, and O ties the same readback to the observer-shadow surface.

The numeric readback is downstream. The stream window S and regular-rational row Q expose the finite value route before the real seal E is available.

Transport, replay, provenance, and local naming preserve this order through T, A, H, C, P, N. Since the carrier has no row for a completed boundary or global observer metric, every consumer receives only the finite shadow handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareVisualRadiusLadder}\uparrow}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PoincareVisualRadiusLadder}\uparrow}$, seed)

Scope closed: Definition 6021.1, Definition 6021.2, and Definition 6021.3 seed the finite Poincare disk visual-radius ladder and shadow handoff.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PoincareVisualRadiusLadder}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a completed hyperbolic boundary, ambient hyperbolic manifold, global observer metric, arbitrary real representative, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-radius admission, boundary-shadow visibility, visual-metric readback, observer-shadow compatibility, stream and regular-rational readback, real-seal ordering, provenance, and local naming for BEDC.Derived.PoincareVisualRadiusLadderUp.

Constructive story: A $\mathsf{PoincareVisualRadiusLadder}^{\uparrow}$ packet is generated from disk-coordinate rows, finite visual radii, boundary-shadow windows, visual-metric values, observer-shadow compatibility, stream windows, regular-rational readbacks, Real seals, transport, replay, componentwise hsame rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6022

A Concrete Naming Certificate for $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$

Following Bishop–Bridges and the Heine–Cantor compact metric proof, the $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$ packet records the proof-critical triangle handoff that turns pointwise modulus rows on a finite compact net into one uniform-modulus row. It sits over the compact finite-net carrier of chapter 129, the continuous-map pointwise rows of chapter 130, the rational tolerance surface of chapter 37, the metric source and target ledgers of chapter 57, the finite lower-bound fold of chapter 619, the final triangle estimate of chapter 1159, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer of chapter 369. The packet names only displayed finite-net, radius, pointwise-modulus, source-triangle, target-triangle, transport, replay, provenance, and local naming rows.

Definition 6022.1 (Compact uniform triangle ledger carrier)**.** *A* $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, X, Y, F, \varepsilon, B, R, \mu, S, T, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row for the compact metric carrier.*
2. *X is the source* $\mathsf{Met}^{\uparrow}$ *distance row used to compare an input point with the selected net center and with a second input point.*
3. *Y is the target* $\mathsf{Met}^{\uparrow}$ *distance row used by the final triangle estimate.*
4. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus packet over the same source metric and target metric.*
5. *ε is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row carried by the compact-uniform request.*
6. *B is the finite bundled-center ledger read from K, listing the inspected centers, radius histories, and coverage rows.*
7. *R is the rational lower-bound row obtained from the bundled radius histories and read through the* $\mathsf{Rat}^{\uparrow}$ *order surface.*
8. *μ is the tolerance-only precision row selected from R for the uniform-modulus consumer.*
9. *S is the source-triangle ledger showing that the selected center remains within the pointwise radius for both compared source points.*

10. *T is the target-triangle ledger joining the two pointwise estimates at the selected center into the final target-distance estimate.*
11. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row produced from μ, S, T.*
12. *H records componentwise* hsame *transport for compact, metric, continuous, rational, source-triangle, target-triangle, uniform-modulus, replay, provenance, and local naming rows.*
13. *C records displayed* Cont *replay along*

$$K, X, Y, F, \varepsilon, B, R, \mu \longmapsto S, T, U \longmapsto H, P, N.$$

14. *P is the* Pkg *provenance over* $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{MetricTriangleUniformEstimate}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
15. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host compactness theorem, selected global modulus, hidden center outside the finite bundle, quotient equality of points or functions, countable choice, or ambient real completeness.

Theorem 6022.2 (Compact uniform triangle ledger NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$ *carrier $L = (K, X, Y, F, \varepsilon, B, R, \mu, S, T, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleLedger}^{\uparrow}}$ *surface consists of compact finite-net admission, source and target metric admission, continuous pointwise-modulus admission, rational tolerance admission, bundled radius lower-bound readback, source-triangle handoff, target-triangle handoff, uniform-modulus export, componentwise transport, displayed replay, package provenance, and local naming. Every public consumer of the packet reads through those rows and through no host compactness, external modulus selector, hidden center, quotient equality, choice schedule, or ambient completeness object.*

Proof. Project the carrier of Definition 6022.1. The rows $K, X, Y, F, \varepsilon, B, R, \mu, S, T, U, H, C, P, N$ are the whole public surface.

The compact and rational source data are read first. The finite bundle B comes from the compact metric row K, and the lower-bound row R is the rational fold over the displayed radius histories for the tolerance row ε.

The source-triangle handoff is then forced through S. It uses the selected precision μ, the compact-net coverage row, and the source metric row X before the pointwise modulus packet F may be applied to both compared source points.

The target-triangle handoff is forced through T. The two pointwise estimates supplied by F at the same selected center are joined by the target metric row Y, and the result is repacked as the $\mathsf{UniformModulus}^{\uparrow}$ row U.

Finally H, C, P, N transport, replay, source, and locally name the same finite route. Since the carrier contains no coordinate for host compactness, an external modulus selector, a hidden center, quotient equality, choice, or ambient completeness, the listed rows exhaust the certificate. □

Theorem 6022.3 (Compact uniform triangle ledger handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read from an accepted* $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Rat}^{\uparrow} \longmapsto \mathsf{FiniteNetMinimumFold}^{\uparrow} \longmapsto \mathsf{MetricTriangleUniformEstimate}^{\uparrow} \longmapsto$$

The route records the compact finite-net bundle, rational lower-bound precision, source-triangle ledger, target-triangle ledger, transport, replay, provenance, and local naming rows as one packet.

Proof. Start with the compact finite-net data. The rows K and B display the finite bundle of centers and coverage rows, while ε and R display the positive rational tolerance and the lower-bound fold used to produce μ.

Use the source metric row X and the source-triangle ledger S. These rows certify that the selected center can be used for both compared source points before the pointwise-modulus row F is consumed.

Use the target metric row Y and the target-triangle ledger T. The two pointwise estimates at the selected center are joined inside the displayed metric triangle estimate, and the result is exactly the row exported to U.

The structural rows H, C, P, N keep this route componentwise stable and locally named. No host compactness theorem, external selector, hidden center, quotient equality, choice schedule, or real-completeness row is available to the consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformTriangleLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformTriangleLedger}^{\uparrow}$ packet is generated from CompactMetric finite-net rows, MetricSpace source and target rows, ContinuousMap pointwise-modulus rows, RatUp tolerance rows, finite lower-bound folds, source-triangle and target-triangle ledgers, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformTriangleLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6022.1, Definition 6022.2, and Definition 6022.3 seed the compact finite-net triangle ledger that carries pointwise moduli into a uniform-modulus row.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformTriangleLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host compactness theorem, selected global modulus, hidden center outside the finite bundle, quotient equality, choice schedule, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net admission, rational lower-bound readback, source-triangle handoff, target-triangle handoff, uniform-modulus export, replay stability, provenance exhaustion, and local naming for BEDC.Derived.CompactUniformTriangleLedgerUp.

Chapter 6023

A Concrete Naming Certificate for CantorIntersectionUniformWindow$^{\uparrow}$

Following Bishop and Bridges' constructive nested-interval treatment, the CantorIntersectionUniformWindow$^{\uparrow}$ packet records the finite uniform-window handoff for the L10 nested compact interval route. It sits over the nested interval surface of chapter 320, the dyadic interval nesting route of chapter 1812, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the metric Cantor-intersection packet of chapter 4359. The packet exposes only the finite window needed by a Real or Completion consumer; it does not assert Heine–Borel compactness, choose an intersection point, quotient streams, countable choice, host equality, or ambient Real$^{\uparrow}$ completeness.

Definition 6023.1 (CantorIntersectionUniformWindow carrier)**.** *A* CantorIntersectionUniformWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (N, D, M, W, R, E, J, H, C, P, K)$$

with the following displayed rows:

1. *N is the* NestedInterval$^{\uparrow}$ *source row, exposing the nested compact interval family and the requested finite depth.*
2. *D is the* DyadicIntervalNesting$^{\uparrow}$ *row, recording dyadic endpoint budgets and containment checks for the same depth.*
3. *M is the* CantorIntersectionMetric$^{\uparrow}$ *metric row, read only as a bounded diameter and overlap witness for the displayed interval family.*
4. *W is the uniform finite-window row, assigning one inspection window to every precision request visible in N, D, M.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the dyadic endpoints and the uniform window family.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row, available only after N, D, M, W, R have been displayed.*
7. *J is the L10 handoff row that routes finite interval windows toward Real and Completion consumers.*
8. *H records componentwise* hsame *transport for the nested source, dyadic ledger, metric row, windows, readbacks, seal, handoff, provenance, and local name.*

9. *C records displayed* Cont *replay of the route*

$$N, D, M \longmapsto W \longmapsto R \longmapsto E \longmapsto J.$$

10. *P is the* Pkg *provenance over* $\mathsf{CantorIntersectionUniformWindow}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalNesting}^{\uparrow}$, $\mathsf{CantorIntersectionMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *K is the local* $\mathsf{NameCert}_{\mathsf{CantorIntersectionUniformWindow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $N, D, M, W, R, E, J, H, C, P, K$. It has no coordinate for a selected intersection point, unrestricted compactness, quotient stream equality, countable choice, host endpoint equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6023.2 (CantorIntersectionUniformWindow NameCert obligations)**.** *For every accepted* $\mathsf{CantorIntersectionUniformWindow}^{\uparrow}$ *carrier*

$$U = (N, D, M, W, R, E, J, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{CantorIntersectionUniformWindow}^{\uparrow}}$ *surface consists exactly of the following seed obligation rows:*

1. *carrier admission for the displayed nested source, dyadic ledger, metric Cantor row, finite uniform windows, regular rational readback, terminal real seal, L10 handoff, transport, replay, provenance, and local name;*
2. *nested-source exposure, requiring every interval read to pass through N before a dyadic or metric row is consumed;*
3. *dyadic containment visibility, requiring D to supply the finite endpoint and containment checks used by W and R;*
4. *uniform-window exactness, requiring the same finite window family W to serve every visible precision request before the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff is read;*
5. *real-seal nonescape, requiring E and J to remain downstream of N, D, M, W, R;*
6. *structural transport, replay, provenance, and local naming through H, C, P, K.*

No Cantor intersection uniform-window obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6023.1. The displayed tuple lists the nested source, dyadic ledger, metric Cantor row, uniform windows, regular rational readback, real seal, L10 handoff, and structural rows.

The nested and dyadic obligations are read in order. A consumer first sees N, then D, and only then the metric row M and window row W; none of these rows exports a selected point or an unrestricted compactness theorem.

The uniform-window exactness obligation is the route through W and R. The row W fixes the finite inspection family, and R reads only the rational data exposed by that family and the dyadic endpoint ledger.

Finally, E, J, H, C, P, K preserve the same route. Since the carrier has no coordinate for the excluded principles, the listed rows exhaust the local NameCert surface. □

Theorem 6023.3 (CantorIntersectionUniformWindow L10 handoff)**.** *Every L10 Real or Completion consumer of an accepted* $\mathsf{CantorIntersectionUniformWindow}^{\uparrow}$ *carrier factors through*

$$N, D, M, W, R, E, J, H, C, P, K.$$

The consumer may inspect the nested interval source, dyadic endpoint ledger, metric Cantor row, finite uniform windows, $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, L10 handoff, transport, replay, provenance, and local name. It cannot bypass this route to obtain general Heine–Borel compactness, a selected intersection point, quotient streams, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an L10 consumer asking for a Real-facing window from the packet. The only source coordinate is N, so the consumer must retain the nested interval family and requested finite depth.

The dyadic and metric rows then fix the window discipline. The row D records endpoint budgets and containment checks, while M supplies only the bounded diameter and overlap information needed by the displayed finite window.

The readback and seal rows are downstream of that discipline. The row W displays the uniform finite windows, R reads them as regular rational data, and E, J expose the terminal seal and L10 handoff without adding a selector.

The rows H, C, P, K transport, replay, source, and name the same finite route. Therefore no consumer can extract the excluded compactness, choice, quotient, equality, or ambient completeness data from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CantorIntersectionUniformWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{CantorIntersectionUniformWindow}^{\uparrow}$ packet is generated from nested interval data, dyadic interval nesting rows, a CantorIntersectionMetric row, finite uniform windows, RegSeqRat readback, a RealUp seal, an L10 handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorIntersectionUniformWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6023.1, Definition 6023.2, and Definition 6023.3 seed the finite uniform-window route from nested compact interval data through dyadic endpoint budgets, CantorIntersectionMetric rows, RegSeqRat readback, RealUp sealing, and L10 handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorIntersectionUniformWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, selected intersection points, quotient streams, countable choice, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, nested-source exposure, dyadic containment visibility, uniform-window exactness, RegSeqRat handoff, RealUp seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CantorIntersectionUniformWindowUp.

Chapter 6024

A Concrete Naming Certificate for $\mathsf{VariationalEquation}^{\uparrow}$

Following Coddington and Levinson, the $\mathsf{VariationalEquation}^{\uparrow}$ packet records the finite linearized-flow companion to the local Picard–Lindelof route. It sits after the local existence surface of chapter 1888, the ODE source of chapter 193, the vector-field Lipschitz ledger of chapter 410, the Banach state classifier of chapter 90, and the complete-metric convergence boundary of chapter 128. The packet names the tangent linear equation along one displayed local flow; it is not a global differentiable flow, a host fundamental-matrix object, a maximal interval theorem, or a choice-selected family of derivatives.

Definition 6024.1 (Variational equation carrier)**.** *A* $\mathsf{VariationalEquation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (B, D, L, T, U, H, C, P, N)$$

with the following visible coordinates:

1. *B is the base local-flow row, read through a* $\mathsf{PicardLindelof}^{\uparrow}$ *packet and the underlying* $\mathsf{Ode}^{\uparrow}$ *time-state source.*
2. *D is the derivative and Lipschitz ledger for the vector field on the same local window, including the displayed linearization row and the Lipschitz bound used to compare nearby tangent reads.*
3. *L is the linearized Picard route, a finite contraction-style ledger that reads each tangent iterate through B, D and the carried Banach state classifier.*
4. *T is the tangent-transport row, recording how displayed tangent vectors are carried along the base local-flow row before the linearized endpoint is read.*
5. *U is the uniqueness and nonescape row for the tangent equation on the named local time-state window.*
6. *H stores componentwise* hsame *transport for the base flow, derivative ledger, linearized Picard route, tangent transport, uniqueness and nonescape row, replay row, provenance row, and local naming row.*
7. *C stores the displayed* Cont *replay from the base local-flow row through derivative reads, tangent Picard iterates, tangent transport, uniqueness, and local endpoint comparison.*
8. *P is the* Pkg *provenance over* $\mathsf{VariationalEquation}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{VariationalEquation}^{\uparrow}}$ *row.*
The carrier exposes only the finite tangent equation route attached to one accepted local-flow packet. It has no coordinate for a host derivative bundle, a global fundamental matrix, smooth flow dependence beyond the displayed derivative ledger, maximal continuation, or bridge-checked standardization.

Lean4 ✓ `BEDC.Derived.VariationalEquationUp.VariationalEquationTasteGate single carrier alignment`

Theorem 6024.2 (Variational equation NameCert obligations)**.** *For an accepted* $\mathsf{VariationalEquation}^{\uparrow}$ *carrier*

$$V = (B, D, L, T, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{VariationalEquation}^{\uparrow}}$ *surface consists exactly of base local-flow admission, derivative and Lipschitz ledger admission, linearized Picard-route compatibility, tangent-transport visibility, uniqueness and nonescape stability, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every tangent endpoint read must pass through these rows before it is treated as a linearized ODE solution.*

Proof. Carrier admission is projection from Definition 6024.1. The base flow row B fixes the $\mathsf{PicardLindelof}^{\uparrow}$ and $\mathsf{Ode}^{\uparrow}$ source, while D supplies the displayed derivative and Lipschitz data used by the linearized equation.

Induct over the finite linearized Picard route L. At the base row, the tangent iterate is attached to the displayed base flow and derivative ledger. At a successor row, the next tangent iterate is read by the same derivative ledger, the same Banach state classifier, and the same local window.

The tangent endpoint is then exposed only through T and U. The transport row T keeps tangent reads attached to the base local-flow route, and the row U records the finite uniqueness and nonescape boundary.

Finally the structural rows H, C, P, N transport, replay, source, and name this same route. Since no carrier coordinate contains a host derivative bundle, a global fundamental matrix, or a selected family of derivatives, the listed obligations exhaust the NameCert surface. □

Theorem 6024.3 (Variational equation local Picard companion route)**.** *Every accepted* $\mathsf{VariationalEquation}^{\uparrow}$ *packet is consumed as a local linearized companion of an accepted* $\mathsf{PicardLindelof}^{\uparrow}$ *packet. A consumer must first read the base local-flow row B, the derivative and Lipschitz ledger D, the linearized Picard route L, the tangent-transport row T, the uniqueness and nonescape row U, and the structural rows H, C, P, N. The packet exports no tangent endpoint outside that finite base-flow route.*

Proof. Project $V = (B, D, L, T, U, H, C, P, N)$. The only base trajectory available to the packet is the displayed local-flow row B, and the only vector-field variation available to the packet is the derivative and Lipschitz ledger D.

Use the finite route L for tangent iteration. Each linearized iterate reads the same base flow and derivative ledger, so a consumer cannot detach the tangent equation from the accepted $\mathsf{PicardLindelof}^{\uparrow}$ route.

The endpoint-facing rows are T and U. They record tangent transport, uniqueness, and nonescape inside the same local window, while H, C, P, N preserve the route under transport, replay, provenance, and naming. □

Falsifiable prediction: Every accepted $\mathsf{VariationalEquation}^{\uparrow}$ tangent endpoint read exposes one base local-flow row, one derivative and Lipschitz ledger, one linearized Picard route, one tangent-transport row, one uniqueness and nonescape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: picard-lindelof, ode, lipschitzmap, banach, completemetric: $\mathsf{VariationalEquation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the base local-flow row, derivative and Lipschitz ledger, linearized Picard route, tangent transport, uniqueness and nonescape row, and the transport/replay/provenance rows tying them together.

Closure status for $\mathsf{NameCert}_{\mathsf{VariationalEquation}^{\uparrow}}$.

Constructive story: A $\mathsf{VariationalEquation}^{\uparrow}$ packet is generated from one PicardLindelof base local-flow row, one derivative and Lipschitz ledger, one linearized Picard route, one tangent-transport row, one uniqueness and nonescape row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{VariationalEquation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6024.1, Definition 6024.2, and Definition 6024.3 seed the finite linearized ODE companion route over a displayed local Picard flow, derivative ledger, tangent transport, nonescape, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{VariationalEquation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global differentiable flow, a host fundamental matrix, maximal continuation, choice-selected derivative families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked base-flow admission, derivative and Lipschitz ledger admission, linearized Picard-route compatibility, tangent-transport visibility, uniqueness and nonescape stability, transport, replay, provenance, and local naming for BEDC.Derived.VariationalEquationUp.

Chapter 6025

A Concrete Naming Certificate for HyperbolicHorofunctionBoundary$^{\uparrow}$

The HyperbolicHorofunctionBoundary$^{\uparrow}$ packet realizes the boundary metric readout branch of chapter 9463 as finite NameCert data. It consumes the four-point comparison route of chapter 3948, the Busemann boundary surface of chapter 3732, the visual-boundary readback of chapter 4181, the observer-shadow metric route of chapter 4154, and the hyperbolic metric rows of chapter 3474. The packet names a finite horofunction-style boundary comparison; it is not a host compactification, a quotient ideal boundary, a selected geodesic ray space, or a global hyperbolic model.

Definition 6025.1 (Hyperbolic horofunction-boundary carrier)**.** *A* HyperbolicHorofunctionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (M, B, V, S, T, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the* HyperbolicMetric$^{\uparrow}$ *metric-read row for the displayed disk or boundary-facing source.*
2. *B is the* HyperbolicBusemannBoundary$^{\uparrow}$ *row recording finite distance-difference boundary data.*
3. *V is the* HyperbolicVisualBoundary$^{\uparrow}$ *row that exposes the same boundary read through visual-boundary data.*
4. *S is the* HyperbolicObserverShadowMetric$^{\uparrow}$ *row carrying the observer-facing shadow comparison.*
5. *T is the transport row tying metric, Busemann, visual-boundary, shadow, and cross-ratio reads to the same boundary route.*
6. *R is the refusal row excluding host horofunction compactification, quotient ideal-boundary equality, selected geodesic rays, and unrecorded global geometry.*
7. *H records componentwise* hsame *transport for metric, Busemann, visual-boundary, shadow, transport, refusal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from metric reads through Busemann difference rows, visual-boundary data, observer-shadow comparison, transport, refusal, and local naming.*
9. *P is the* Pkg *provenance over* HyperbolicHorofunctionBoundary$^{\uparrow}$, HyperbolicCrossRatio$^{\uparrow}$, HyperbolicBusemannBou HyperbolicVisualBoundary$^{\uparrow}$, HyperbolicObserverShadowMetric$^{\uparrow}$, HyperbolicMetric$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, B, V, S, T, R, H, C, P, N$. It has no coordinate for a host horofunction compactification, quotient boundary equality, selected rays, or a global metric oracle.

Theorem 6025.2 (Hyperbolic horofunction-boundary NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ *carrier*

$$Y = (M, B, V, S, T, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}}$ *surface consists exactly of metric admission, Busemann boundary admission, visual-boundary readback, observer-shadow metric comparison, boundary transport, refusal of host compactification, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every boundary metric read must inspect these rows before it treats the packet as a horofunction-boundary comparison.*

Proof. Project the carrier of Definition 6025.1. The metric row M fixes the finite hyperbolic comparison source, and the Busemann row B supplies the displayed distance-difference boundary data.

Read the boundary-facing rows V, S, T next. The visual-boundary row exposes the boundary token, the observer-shadow row supplies the finite metric readout seen by the observer route, and the transport row keeps these reads attached to the same boundary comparison.

The refusal row R blocks replacement by a host compactification or quotient ideal-boundary object. The structural rows H, C, P, N transport, replay, source, and name the same finite surface, so no consumer can obtain a horofunction-boundary comparison outside the displayed rows. □

Theorem 6025.3 (Hyperbolic horofunction-boundary metric readout route)**.** *Every accepted* HyperbolicHorofunctio
consumer that reads boundary metric data must pass through

$\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{HyperbolicVisualBoundary}^{\uparrow}$, $\mathsf{HyperbolicObserverShadowMetric}^{\uparrow}$,

The route exposes metric, Busemann, shadow, transport, refusal, provenance, and local NameCert rows before any horofunction-boundary comparison is public.

Proof. Let $Y = (M, B, V, S, T, R, H, C, P, N)$ be accepted. The consumer first reads M, B, V, S, so the boundary comparison is attached to displayed metric, Busemann, visual-boundary, and observer-shadow data.

Use T to align the boundary route with the cross-ratio-facing transport supplied by Definition 3948.5. This keeps the boundary metric comparison finite and prevents replacement by a host compactification.

Finally R, H, C, P, N refuse the excluded host objects and preserve the displayed route under transport, replay, provenance, and naming. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ packet must expose metric, Busemann, shadow, transport, refusal, provenance, and NameCert rows before a boundary metric read is accepted.

Independence witness: hyperbolic-cross-ratio, hyperbolic-busemann-boundary, hyperbolic-visual-boundary, hyperbolic-observer-shadow-metric, hyperbolic-metric: $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries metric admission, Busemann boundary

data, visual-boundary readback, observer-shadow metric comparison, boundary transport, host-compactification refusal, provenance, replay, transport, and local naming.

The Lean TasteGate side for $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ must provide explicit $\mathsf{Nontrivial\ HyperbolicHorofu}$ and $\mathsf{FieldFaithful\ HyperbolicHorofunctionBoundary}^{\uparrow}$ instances. These instances keep the metric, Busemann, visual-boundary, shadow, transport, refusal, provenance, replay, package, and local naming rows distinguishable.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ packet is generated from HyperbolicMetric, HyperbolicBusemannBoundary, HyperbolicVisualBoundary, HyperbolicObserverShadowMetric, HyperbolicCrossRatio-facing transport, refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6025.1, Definition 6025.2, and Definition 6025.3 seed the finite boundary metric readout route over Busemann, visual-boundary, observer-shadow, cross-ratio transport, refusal, provenance, replay, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host horofunction compactification, quotient ideal-boundary equality, selected geodesic rays, global hyperbolic geometry, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric admission, Busemann boundary admission, visual-boundary readback, observer-shadow metric comparison, boundary transport, refusal, TasteGate nontriviality, field faithfulness, provenance, replay, transport, and local naming for BEDC.Derived.HyperbolicHorofunctionBoundaryUp.

Chapter 6026

A Concrete Naming Certificate for $\mathsf{RegularCauchyOscillation}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyOscillation}^{\uparrow}$ packet records the finite oscillation control carried by a regular Cauchy rational name before a Real or completion-facing consumer may read the terminal seal. It sits after the tail-variation ledger of chapter 448, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the Cauchy modulus surface of chapter 292, and the terminal real-name seal of chapter 38. The packet is an L10 NameCert surface for regular-Cauchy oscillation arguments, not quotient stream equality, a selected limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6026.1 (Regular Cauchy oscillation carrier)**.** *A* $\mathsf{RegularCauchyOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (S, W, Q, D, L, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the inspected source indices, tail windows, and comparison pairs used by the oscillation request.*
2. *W is the finite window ledger synchronized with S and with the requested Cauchy-modulus thresholds.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational sequence observed through the displayed windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, slack, and comparison row used to measure the tail oscillation.*
5. *L is the* $\mathsf{CauchyOscillation}^{\uparrow}$ *ledger, recording the displayed tail-diameter comparisons read from S, W, Q, D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the StreamName schedule, RegSeqRat readback, dyadic tolerance row, and oscillation ledger have been displayed.*
7. *H records componentwise* hsame *transport for the schedule, windows, readback, tolerance, oscillation ledger, real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each Real or completion-facing oscillation request through $S, W, Q, D, L, E, H, P, N$.*

9. *P is the* Pkg *provenance over* RegularCauchyOscillation$^{\uparrow}$, CauchyOscillation$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, CauchyModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{RegularCauchyOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, Q, D, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge credentials.

Lean4 ✓ `BEDC.Derived.RegularCauchyOscillationUp.RegularCauchyOscillationTasteGate single carrier alignment`

Theorem 6026.2 (Regular Cauchy oscillation NameCert obligations)**.** *For every accepted* RegularCauchyOscillation *carrier*

$$O = (S, W, Q, D, L, E, H, C, P, N),$$

the local NameCert$_{\text{RegularCauchyOscillation}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the StreamName schedule, finite windows, RegSeqRat readback, DyadicRatCore tolerance row, CauchyOscillation ledger, Real seal, transport, replay, provenance, and local name;*
2. *schedule coverage, requiring every inspected index and tail pair to be named by S and opened through a window in W;*
3. *dyadic tolerance exactness, requiring every oscillation comparison to consume its tolerance and slack cells from D;*
4. *Cauchy-modulus compatibility, requiring the tail threshold read through* CauchyModulus$^{\uparrow}$ *to be attached to the same S, W, Q, D route;*
5. *Real seal non-escape, requiring E, H, C, P, N to expose no quotient stream equality, host real equality, selected limit, countable choice, trichotomy, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

No Regular Cauchy oscillation obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6026.1. The displayed tuple $S, W, Q, D, L, E, H, C, P, N$ is the entire public surface, so carrier admission is exactly admission of those rows.

The schedule and dyadic clauses are read next. The row S names the inspected indices and tail pairs, W opens the corresponding finite windows, and D supplies the tolerance and slack cells used by each comparison.

The oscillation ledger L is downstream of S, W, Q, D. Its Cauchy-modulus compatibility is therefore a finite route through the same schedule, windows, regular readback, and dyadic tolerance cells before the terminal seal E can be consumed.

The structural rows H, C, P, N transport, replay, source, and name only this route. Since the carrier contains no further coordinate, the excluded quotient, choice, order, completeness, verification, and bridge objects cannot be exported by the local NameCert surface. □

Theorem 6026.3 (Regular Cauchy oscillation Real handoff)**.** *Every* Real$^{\uparrow}$ *or completion-facing consumer of an accepted* RegularCauchyOscillation$^{\uparrow}$ *carrier first reads the finite route*

$$S, W, Q, D \longmapsto L \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, CauchyOscillation row, Real seal, transport, replay, provenance, and local NameCert before it can treat the regular Cauchy name as a real-facing source. It receives no quotient stream equality, host real equality, selected limit, countable-choice schedule, trichotomy row, ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate, Lean verification, or bridge credential.*

Proof. Start with a Real or completion-facing request. The carrier has no direct path to the real seal, so the request must first open the schedule S, the finite windows W, the regular rational readback Q, and the dyadic tolerance row D.

These rows determine the oscillation ledger L. The ledger is the only row that records the tail comparison needed before the terminal seal E is exposed, so every seal read factors through the displayed oscillation route.

Transport through H, replay through C, provenance through P, and local naming through N preserve the same route. With no quotient, choice, trichotomy, or ambient-completeness coordinate in the packet, the handoff exports only the finite regular-Cauchy oscillation surface. □

Falsifiable prediction: Every $\mathsf{RegularCauchyOscillation}^{\uparrow}$ Real-facing read factors through one StreamName schedule, one finite window ledger, one RegSeqRat readback row, one DyadicRatCore tolerance row, one CauchyOscillation ledger, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchyoscillation, regseqrat, streamname, dyadicratcore, real, cauchy-modulus: $\mathsf{RegularCauchyOscillation}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the regular source schedule, finite windows, regular readback, dyadic tolerance cells, Cauchy oscillation ledger, Real seal, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyOscillation}^{\uparrow}$ packet is generated from a StreamName schedule, finite regular-source windows, RegSeqRat readback, DyadicRatCore tolerance cells, a CauchyOscillation ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6026.1, Definition 6026.2, and Definition 6026.3 seed the finite regular-Cauchy oscillation route from inspected stream windows through dyadic tolerance cells, RegSeqRat readback, CauchyOscillation ledger, and RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule coverage, dyadic tolerance exactness, Cauchy-modulus compatibility, Real seal non-escape, transport stability, replay stability, provenance exhaustion, and local naming for BEDC.Derived.RegularCauchyOscillationUp.

The $\mathsf{FiniteWindowConfluence}^{\uparrow}$ spine is routed here so the downstream frontier consumer of chapter 4238 can find the finite confluence dependency without entering the child file.

Chapter 6027

A Concrete Naming Certificate for $\mathsf{FiniteWindowConfluence}^{\uparrow}$

The $\mathsf{FiniteWindowConfluence}^{\uparrow}$ packet realizes the MetaCIC normalization audit map of chapter 9383 as finite beta-window confluence data. It sits beside the redex-frontier packet of chapter 1008, the local join ledger of chapter 4126, the residual-substitution boundary of chapter 2181, the closed normal seal of chapter 741, and the normal-fragment confluence packet of chapter 737. The carrier records a bounded beta-window peak, local join evidence, substitution-boundary retention, and closed-normal endpoint rows without claiming unrestricted global confluence, quotient conversion equality, arbitrary subject reduction, or kernel consistency.

Falsifiable prediction: A $\mathsf{FiniteWindowConfluence}^{\uparrow}$ packet whose beta window omits the residual-substitution boundary row is rejected even when its local join ledger and closed-normal endpoint rows are present.

Independence witness: metacic-redex-frontier, metacic-local-join-ledger, metacic-residual-substitution-boundary, closednormalconfluenceseal, metacicconfluencenormalfragment: none is bijective with $\mathsf{FiniteWindowConfluence}^{\uparrow}$ because this packet jointly carries the beta-window peak, redex-frontier read, local join ledger, residual-substitution boundary, closed-normal endpoint, TasteGate readiness, transport, replay, provenance, and local naming rows.

Definition 6027.1 (FiniteWindowConfluence carrier)**.** *A* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, R, J, S, N, W, G, H, C, P, K)$$

with the following displayed rows:

1. *B is the bounded beta-window row, naming the source term, window radius, and two outgoing beta observations.*
2. *R is the* $\mathsf{MetaCICRedexFrontier}^{\uparrow}$ *row, exposing the redex-frontier discharge data consumed by the window.*
3. *J is the* $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$ *row, recording the local peak and candidate join ledger for the same bounded window.*
4. *S is the* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *row, retained whenever the join read depends on residual substitution data.*
5. *N is the* $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$ *endpoint row, available only for closed normal endpoint reads.*

6. *W is the* $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ *window-fragment row, keeping the closed-normal finite fragment separate from unrestricted global confluence.*
7. *G is the* $\mathsf{TasteGate}$ *readiness row for later* $\mathsf{Nontrivial}$ *and* $\mathsf{FieldFaithful}$ *instances over this packet.*
8. *H records componentwise* hsame *transport for beta windows, redex frontiers, local joins, residual-substitution boundaries, closed-normal endpoints, normal fragments, TasteGate readiness, provenance, and local naming.*
9. *C records displayed* Cont *replay from a normalization-audit query through $B, R, J, S, N, W, G, H, P, K$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowConfluence}^{\uparrow}$, $\mathsf{MetaCICRedexFrontier}^{\uparrow}$, $\mathsf{MetaCICLocalJoinLedger}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{ClosedNormalConfluenceSeal}^{\uparrow}$, $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$, BHist, hsame, Cont, $\mathsf{TasteGate}$, *and* $\mathsf{NameCert}$.
11. *K is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowConfluence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, R, J, S, N, W, G, H, C, P, K$. It contains no coordinate for unrestricted MetaCIC confluence, global Church–Rosser, quotient conversion equality, arbitrary subject reduction, evaluator equality, or kernel consistency.

Theorem 6027.2 (FiniteWindowConfluence NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *carrier*

$$F = (B, R, J, S, N, W, G, H, C, P, K),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowConfluence}^{\uparrow}}$ *surface consists exactly of bounded beta-window exposure through B, redex-frontier readback through R, local-join ledger exposure through J, residual-substitution boundary retention through S, closed-normal endpoint exposure through N, normal-fragment separation through W, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through K.*

Proof. Project the carrier of Definition 6027.1. Rows B, R, J, S, N, W are the confluence-facing finite data, while G, H, C, P, K are the structural certificate data.

The beta-window read starts with B and R. The source, radius, and two outgoing observations are bounded by B, and the redex-frontier row R keeps the required discharge data visible before a join can be consumed.

The local join is then read through J while retaining S. The residual-substitution boundary is a row of the same packet, so the local join cannot be detached into a global confluence theorem or a substitution oracle.

Closed-normal endpoints are read only through N and W. These rows expose the normal fragment and the closed-normal seal, but they do not supply arbitrary closed-term Church–Rosser or quotient conversion equality.

Finally G, H, C, P, K reserve the TasteGate instance surface, transport the rows, replay the audit query, source the packet, and name it. Since no carrier coordinate contains unrestricted confluence, subject reduction, evaluator equality, or kernel consistency, the listed rows exhaust the local NameCert surface. □

Theorem 6027.3 (FiniteWindowConfluence normalization-audit handoff)**.** *Every normalization-audit consumer of an accepted* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *carrier factors through*

$$B, R, J, S, N, W, G, H, C, P, K.$$

The consumer may inspect the bounded beta window, redex frontier, local join ledger, residual-substitution boundary, closed-normal endpoint, normal fragment, TasteGate readiness, transport,

replay, provenance, and local name. It cannot bypass this route to obtain unrestricted MetaCIC confluence, global Church–Rosser, quotient conversion equality, arbitrary subject reduction, evaluator equality, or kernel consistency.

Proof. Start with a normalization-audit query. The only beta-window entrance is B, and the only redex-facing discharge surface is R, so the query must retain the bounded source, outgoing beta observations, and redex-frontier data.

The local join route is then constrained by J and S. Row J records the candidate join ledger, while S keeps the residual-substitution boundary visible as an input rather than as a solved global theorem.

The endpoint route is constrained by N and W. A closed-normal read may use the closed-normal seal and normal-fragment packet, but those rows remain finite endpoint data, not an unrestricted confluence theorem.

The structural rows G, H, C, P, K preserve the same packet under TasteGate readiness, transport, replay, provenance, and local naming. Hence every normalization-audit consumer remains inside the displayed finite route. □

Theorem 6027.4 (FiniteWindowConfluence TasteGate readiness)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* FiniteWindowConfluence$^{\uparrow}$ *at the displayed finite carrier surface*

$$B, R, J, S, N, W, G, H, C, P, K.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the bounded beta window, redex frontier, local join ledger, residual-substitution boundary, closed-normal endpoint, normal-fragment row, transport, replay, provenance, and local NameCert data.

Proof. The readiness row G in Definition 6027.1 is tied to the whole finite-window packet, not to the local join ledger alone.

Nontriviality must exhibit an accepted packet with all displayed rows present. FieldFaithfulness must show that deleting any of $B, R, J, S, N, W, H, C, P, K$ changes the readable carrier surface. Thus the TasteGate burden remains exactly at the finite beta-window confluence boundary. □

Theorem 6027.5 (FiniteWindowConfluence sibling route)**.** *Every public confluence read exported by an accepted* FiniteWindowConfluence$^{\uparrow}$ *packet routes through the sibling rows named in the carrier: the* MetaCICRedexFrontier$^{\uparrow}$ *redex-frontier row, the* MetaCICLocalJoinLedger$^{\uparrow}$ *local-join ledger, the* MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *residual-substitution boundary, the* ClosedNormalConfluenceSeal$^{\uparrow}$ *closed-normal seal, and the* MetaCICConfluenceNormalFragment$^{\uparrow}$ *normal-fragment confluence row. A consumer cannot read the packet as an isolated confluence theorem without also retaining those sibling routes.*

Proof. Begin with the carrier of Definition 6027.1. Its public confluence-facing rows are B, R, J, S, N, W, so the bounded beta-window row is not readable without the redex-frontier row of chapter 1008 and the local-join ledger of chapter 4126.

The residual part of the read is held by S. Thus any join consumer that keeps J must also pass through the residual-substitution boundary of chapter 2181; otherwise the packet would claim a substitution oracle that is not present in its finite rows.

The endpoint part is held by N and W. These rows point respectively to chapter 741 and chapter 737, so the read remains a closed-normal finite fragment rather than unrestricted MetaCIC confluence.

The structural rows G, H, C, P, K then apply the existing NameCert obligations, the normalization-audit handoff, and the TasteGate readiness row to the same packet. Hence the public route is exactly the displayed finite sibling route. □

Theorem 6027.6 (FiniteWindowConfluence residual diamond scope)**.** *Any local diamond read over the bounded beta window of an accepted* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *carrier must expose the redex-frontier row, local-join ledger, residual-substitution boundary, closed-normal endpoint row, and normal-fragment row together. In carrier notation, a diamond-facing consumer may inspect*

$$B, R, J, S, N, W$$

only as the finite route already transported, replayed, sourced, and named by G, H, C, P, K*. The read remains a finite residual-diamond scope; it does not claim unrestricted Church–Rosser, global subject reduction, quotient conversion equality, evaluator equality, or kernel consistency.*

Proof. Apply the NameCert obligation surface of Definition 6027.2. It requires the bounded beta-window row, redex-frontier readback, local-join ledger, residual-substitution boundary, closed-normal endpoint, and normal-fragment separation to remain visible in the same accepted packet.

Next apply the normalization-audit handoff of Definition 6027.3. A local diamond read is a normalization-audit consumer, so it cannot bypass R, J, S, N, W after entering through B.

Finally use the sibling route of Definition 6027.5. The redex-frontier, local-join, residual-substitution, closed-normal, and normal-fragment rows are sibling dependencies rather than optional annotations. Hence the diamond scope is finite and residual-boundary-aware, with no route to the global principles excluded from the carrier. □

Theorem 6027.7 (FiniteWindowConfluence residual-boundary inversion)**.** *If a local diamond read from an accepted* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *packet reaches* J *as a candidate join, then the same read also exposes* S *as the residual-substitution boundary row. The candidate join cannot be reinterpreted as a join independent of residual data; it remains in the route*

$$B, R, J, S, N, W, G, H, C, P, K.$$

Proof. Project the accepted carrier $F = (B, R, J, S, N, W, G, H, C, P, K)$. The local join ledger J and residual-substitution boundary S are adjacent confluence-facing rows of the same finite packet.

Use the NameCert obligations of Definition 6027.2 and the normalization-audit handoff of Definition 6027.3. They require a diamond-facing read to keep the bounded beta window, redex frontier, local join, residual-substitution boundary, closed-normal endpoint, and normal fragment in one transported and replayed route.

Finally apply the sibling route of Definition 6027.5. It keeps the redex-frontier, local-join, residual-substitution, closed-normal, and normal-fragment rows together, so J cannot be cited as an unconditional global confluence join or as a join detached from residual-substitution data. □

Theorem 6027.8 (FiniteWindowConfluence normalization-frontier handoff)**.** *Every local diamond read from an accepted* $\mathsf{FiniteWindowConfluence}^{\uparrow}$ *carrier consumed by a* $\mathsf{NormalizationCandidateFrontier}^{\uparrow}$ *route carries the rows* B, R, J, S, N, W *through the frontier packet's candidate-evidence, bounded-route, obstruction, transport, replay, provenance, and local NameCert rows. The handoff is therefore a bounded candidate-mediated route from finite-window confluence data to the normalization frontier, not a proof of global Church–Rosser, unrestricted subject reduction, evaluator equality, quotient conversion equality, or kernel consistency.*

Proof. Start with residual-boundary inversion, Definition 6027.7. A local diamond read that reaches the candidate join ledger J must also expose the residual-substitution boundary S, while B, R, N, W remain in the same finite confluence route.

Now pass to the frontier through Definition 5994.2 and Definition 5994.3. The frontier can consume positive normalization evidence only while its candidate-evidence row E and bounded-route row B remain visible, so the finite diamond read is carried as bounded candidate data rather than as an evidence-free confluence theorem.

Finally use obstruction exactness and local confluence scope for the frontier, Definition 5994.6 and Definition 5994.7. These rows keep obstruction, transport, replay, provenance, and local naming in the handoff, so erased candidate evidence or erased residual data reaches the obstruction route instead of exporting global Church–Rosser, subject reduction, evaluator equality, quotient conversion equality, or kernel consistency. □

Theorem 6027.9 (FiniteWindowConfluence MetaCIC frontier forward route)**.** *Every* MetaCICParallelDiamondFron *consumer of an accepted* FiniteWindowConfluence$^{\uparrow}$ *carrier must keep the bounded beta-window row, redex-frontier row, local-join ledger, residual-substitution boundary, closed-normal endpoint row, normal-fragment row, componentwise transport, displayed replay, provenance, and local* NameCert *row visible before any frontier handoff. In carrier notation, the forward route is*

$$B, R, J, S, N, W, H, C, P, K,$$

read first through the finite-window obligations, residual-boundary inversion, and normalization-frontier handoff, then through the row discipline of chapter 4238. This is a substantive dependency on chapter 4238: the frontier consumes bounded-window evidence only after retained residual-substitution, critical-pair, candidate-join, checker, obstruction, transport, replay, provenance, and local naming rows are visible. The route supplies bounded-window evidence to the parallel-diamond frontier only; it does not claim global confluence, unrestricted subject reduction, quotient term equality, evaluator equality, or kernel consistency.

Proof. Apply the NameCert obligations of Definition 6027.2. They expose the bounded beta-window, redex-frontier, local-join, residual-substitution, closed-normal endpoint, and normal-fragment rows together with transport, replay, provenance, and local naming.

Next apply the residual-boundary inversion of Definition 6027.7. Any read that reaches the local join ledger J still carries S as residual-substitution boundary data, so the route cannot be recast as a residual-free local join.

Now use the normalization-frontier handoff of Definition 6027.8. It already transports the finite diamond read through candidate evidence, bounded-route, obstruction, transport, replay, provenance, and local NameCert rows before a frontier consumer may inspect it.

Finally compare with the MetaCICParallelDiamondFrontier$^{\uparrow}$ row discipline of Definition 4238.2 and Definition 4238.3. That frontier requires retained residual-substitution, critical-pair, candidate-join, residual-frontier, checker, obstruction, transport, replay, provenance, and local naming rows. The finite-window packet can therefore feed only its displayed bounded route into the frontier, with no route to global confluence, unrestricted subject reduction, quotient term equality, evaluator equality, or kernel consistency. □

Closure status for NameCert$_{\text{FiniteWindowConfluence}^{\uparrow}}$**.**

Constructive story: A FiniteWindowConfluence$^{\uparrow}$ packet is generated from a bounded beta-window row, MetaCICRedexFrontier row, MetaCICLocalJoinLedger row, MetaCICResidualSubstitutionBoundary row, ClosedNormalConfluenceSeal endpoint row, MetaCICConfluenceNormalFragment row, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteWindowConfluence}\uparrow}$, seed)

Scope closed: Definition 6027.1, Definition 6027.2, Definition 6027.3, and Definition 6027.4 seed the finite beta-window confluence packet for MetaCIC normalization-audit consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteWindowConfluence}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unrestricted MetaCIC confluence, global Church–Rosser, quotient conversion equality, arbitrary subject reduction, evaluator equality, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, beta-window exposure, redex-frontier readback, local-join exactness, residual-substitution boundary retention, closed-normal endpoint exposure, normal-fragment separation, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowConfluenc

Chapter 6028

A Concrete Naming Certificate for $\mathsf{BarInductionContinuity}^{\uparrow}$

Following Brouwer's bar induction and the presentation of Troelstra–van Dalen, the $\mathsf{BarInductionContinuity}^{\uparrow}$ packet records the finite certificate by which a barred spread computation yields a continuity modulus for a real-facing read. In the Bishop–Bridges reading, the continuity principle is not a free choice functional: it is a finite prefix route through the bar, the fan depth, the displayed stream windows, the regular rational readback, and the terminal modulus row.

The packet sits between the barred-spread source of chapter 1675, the finite-depth discipline of chapter 5863, the finite observation interface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. It names the continuity-extraction route only at the displayed prefix level.

Definition 6028.1 (Bar induction continuity carrier)**.** *A* $\mathsf{BarInductionContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (B, F, S, R, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the barred-spread row, containing the spread law, the decidable bar test on inspected prefixes, and the local bar-induction descent data.*
2. *F is the finite fan-depth row, bounding the prefix depth at which the accepted bar evidence must be read.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row, listing exactly the stream coordinates opened before the bar decision is consumed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same prefix windows, with the rational observations and regularity ledger visible to the consumer.*
5. *M is the modulus-output row, recording the continuity bound extracted from B, F, S, R for the displayed precision request.*
6. *H records componentwise* hsame *transport for the barred-spread row, fan-depth row, stream windows, rational readback, modulus output, replay routes, provenance, and local naming.*
7. *C records displayed* Cont *replay from a continuity request through B, F, S, R, M, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{BarInductionContinuity}^{\uparrow}$, $\mathsf{BrouwerBarInduction}^{\uparrow}$, $\mathsf{FiniteFanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{BarInductionContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these displayed rows. It has no coordinate for countable choice, a selected infinite branch, quotient stream equality, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6028.2 (Bar induction continuity NameCert obligations)**.** *For every accepted* $\mathsf{BarInductionContinuity}^{\uparrow}$ *carrier*

$$K = (B, F, S, R, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BarInductionContinuity}^{\uparrow}}$ *surface consists exactly of carrier admission for the barred-spread data, finite fan-depth admission, stream-prefix window exposure, regular rational readback, modulus-output exactness, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every exported continuity bound factors through the finite route*

$$B, F, S, R \longmapsto M, H, C, P, N.$$

Proof. Project the carrier of Definition 6028.1. The tuple displays the barred-spread row, finite fan depth, stream prefix windows, regular rational readback, modulus output, transport, replay, provenance, and local naming.

The bar-induction part supplies only finite prefix descent. The row B decides the inspected bar entries, while F bounds the depth at which those entries are read. Thus no selected infinite branch or countable-choice schedule is available.

The continuity output is then read through S and R. The stream row exposes the prefix coordinates, and the regular rational row packages exactly the rational observations and regularity ledger obtained from that same prefix window.

Finally M records the modulus extracted from the displayed route, and H, C, P, N only transport, replay, source, and name it. Since the carrier has no coordinate for quotient stream equality, open-cover compactness, host real equality, or ambient completeness, row projection and finite prefix induction give precisely the stated obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{BarInductionContinuity}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BarInductionContinuity}^{\uparrow}$ packet is generated from barred-spread data, a finite fan-depth row, StreamName prefix windows, RegSeqRat readback, a modulus-output row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BarInductionContinuity}^{\uparrow}}$, seed)

Scope closed: The seed scope records the finite carrier, prefix-window classifier, modulus-output route, transport stability, replay surface, and local non-escape boundary for bar-induction continuity packets.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BarInductionContinuity}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close countable choice, selected branches, quotient stream equality, open-cover compactness, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the barred-spread, finite-depth, prefix-window, readback, and modulus-output rows.

Chapter 6029

A Concrete Naming Certificate for $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ packet records the finite route by which a uniformly Cauchy family of real-valued names is admitted as a uniform-limit candidate before any endpoint, quotient function, selected limit, countable choice, or ambient completeness principle is available. It sits beside the finite family route of chapter 1830, the shared-tail criterion of chapter 435, the uniform-limit endpoint of chapter 463, and the continuity-facing handoff of chapter 1748. The chapter names only the Real-and-Completion corridor through displayed StreamName, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows.

Definition 6029.1 (Uniform limit Cauchy criterion carrier)**.** *A* $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, W, M, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the function-family row, carrying the inspected family-window entries and the point requests to which the uniform criterion is applied.*
2. *W is the finite family-window ledger, recording the shared windows for indices, source points, and comparison stages.*
3. *M is the uniform Cauchy modulus row, requiring one displayed modulus request to govern every retained point window.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row opened from the selected family windows before any regular rational readback is available.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same observations and dyadic tolerances.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after F, W, M, S, R have been read.*
7. *H records componentwise* hsame *transport for family windows, modulus rows, observations, regular readbacks, Real seals, replay rows, provenance, and local naming.*
8. *C records displayed* Cont *replay from a Real or Completion request through M, W, S, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{UniformLimitContinuous}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{UniformLimitCauchyCriterion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, W, M, S, R, E, H, C, P, N$. It has no coordinate for quotient function equality, pointwise-only convergence, selected family limits, countable choice, host completeness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformLimitCauchyCriterionUp.UniformLimitCauchyCriterionTasteGate single carrier alignment`

Theorem 6029.2 (Uniform limit Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ *carrier*

$$U = (F, W, M, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformLimitCauchyCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for family-window rows, a shared uniform Cauchy modulus,* $\mathsf{StreamName}^{\uparrow}$ *observations,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the uniform modulus M, then the displayed windows W, then the stream observations S and regular readback R, before the Real seal E is available.*

Proof. Project the carrier of Definition 6029.1. Its rows list the only family-window, modulus, observation, readback, seal, transport, replay, provenance, and naming data admitted by the local classifier.

The uniformity obligation is the placement of M before W, S, R, E. A consumer cannot replace it by a pointwise threshold, because the carrier stores one displayed modulus route for every retained family window.

The regular and Real-facing rows then follow the L10 route already used by $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The structural rows H, C, P, N transport, replay, source, and name that same finite route.

No excluded coordinate occurs in the packet. Thus quotient function equality, selected family limits, countable choice, host completeness, and ambient $\mathsf{Real}^{\uparrow}$ completeness cannot appear as local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.UniformLimitCauchyCriterionUp.UniformLimitCauchyCriterionCarrier namecert obligations`

Theorem 6029.3 (Uniform limit Cauchy criterion L10 handoff)**.** *Every endpoint-facing read of an accepted* $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ *packet factors through the finite L10 route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

with the shared uniform Cauchy evidence supplied by $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *and the continuity-facing endpoint discipline compatible with* $\mathsf{UniformLimitContinuous}^{\uparrow}$. *In particular, the handoff exposes family-window rows, one uniform Cauchy modulus, stream observations, regular rational readbacks, Real seals, componentwise transport, replay, provenance, and local naming before any Real-and-Completion consumer may inspect the endpoint.*

Proof. Start with the NameCert obligations of Definition 6029.2. They force each endpoint-facing request to display the family window and the shared uniform modulus before the regular readback route is opened.

Read the stream and regular rows next. The $\mathsf{StreamName}^{\uparrow}$ observations S and $\mathsf{RegSeqRat}^{\uparrow}$ readback R are tied to the same finite family-window ledger W and the same modulus row M.

The terminal Real seal is downstream of that route. The $\mathsf{UniformCauchyCriterion}^{\uparrow}$ provenance supplies the shared-tail criterion, while $\mathsf{UniformLimitContinuous}^{\uparrow}$ may consume the resulting endpoint only as a displayed finite handoff.

Since H, C, P, N preserve the same rows and add no external analytic object, the L10 handoff contains no quotient function equality, pointwise-only convergence witness, selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness datum. □

Theorem 6029.4 (Uniform limit Cauchy criterion endpoint stability)**.** *Let*

$$U = (F, W, M, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ *carrier. Any endpoint-facing consumer preserving the rows* $F, W, M, S, R, E, H, C, P, N$ *preserves the same* $\mathsf{StreamName}^{\uparrow}$ *observation,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and* $\mathsf{Real}^{\uparrow}$ *seal. In particular, the consumer cannot replace the shared uniform Cauchy modulus* M *by pointwise convergence, selected limits, countable choice, or ambient completeness.*

Proof. Project the carrier of Definition 6029.1. The endpoint-facing data are exactly the family row, finite windows, shared uniform modulus, stream observation, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Apply Definition 6029.2 in row order. The modulus row M precedes the observations S, readback R, and Real seal E, so any row-preserving consumer must keep the same uniform Cauchy evidence attached to the same finite windows.

Finally apply Definition 6029.3. It identifies the endpoint route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, while H, C, P, N preserve and name that route. Hence no pointwise-only convergence witness, selected limit, countable-choice selector, or ambient completeness datum can replace the shared uniform modulus. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLimitCauchyCriterion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLimitCauchyCriterion}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6029.1, Definition 6029.2, Definition 6029.3, and Definition 6029.4 close the finite Real-and-Completion route for uniform Cauchy function-family convergence through displayed StreamName, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows at obligation strength.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLimitCauchyCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient function equality, selected family limits, countable choice, pointwise-only convergence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-modulus stability, StreamName observation routing, RegSeqRat readback, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformLimitCauchy

Constructive story: A $\mathsf{UniformLimitCauchyCriterion}^{\uparrow}$ packet is generated from family-window rows, one uniform Cauchy modulus, StreamName observations, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6030

A Concrete Naming Certificate for $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen, the $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ packet records the finite-prefix modulus used when a choice-sequence input is read by a Real-facing consumer. It refines the continuity surface of chapter 3465 by isolating the concrete finite row that turns a displayed $\mathsf{ChoiceSequence}^{\uparrow}$ prefix into a $\mathsf{StreamName}^{\uparrow}$ window, then into $\mathsf{RegSeqRat}^{\uparrow}$ readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet is a finite NameCert carrier for prefix-modulus readback; it is not a host continuity theorem, a selected infinite branch, quotient stream equality, countable choice, arbitrary bar induction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6030.1 (Brouwer finite choice-sequence modulus carrier)**.** *A* $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (B, Q, W, R, E, M, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$ *source row that licenses finite-prefix dependence for the read.*
2. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *input row, exposing only the inspected finite prefix of the choice sequence.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *prefix-window row opened from Q and bounded by the displayed modulus.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite stream window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback is displayed.*
6. *M is the finite modulus row, recording the prefix length sufficient for the displayed output read.*
7. *H records componentwise* hsame *transport for the continuity source, choice-sequence prefix, window, readback, seal, modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the consumer request through B, Q, M, W, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$, $\mathsf{BrouwerContinuityPrinciple}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, Q, W, R, E, M, H, C, P, N$. It has no coordinate for a completed choice sequence, selected branch, quotient stream equality, countable choice, arbitrary bar-induction principle, host functional continuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BrouwerFiniteChoiceSequenceModulusUp.BrouwerFiniteChoiceSequenceModulusTasteGate`
`single carrier alignment`

Theorem 6030.2 (Brouwer finite choice-sequence modulus obligations)**.** *For every accepted* $\mathsf{BrouwerFiniteChoiceSe}$ *carrier* $F = (B, Q, W, R, E, M, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}}$ *surface consists exactly of continuity-source admission through* B*, choice-sequence prefix admission through* Q*, finite modulus display through* M*,* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{Real}^{\uparrow}$ *sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

No NameCert obligation is stored outside those displayed rows.

Proof. Project the carrier in Definition 6030.1. The analytic rows are precisely the Brouwer-continuity source, the choice-sequence prefix, the finite modulus, the stream window, the regular rational readback, and the terminal Real seal.

The finite-prefix part is read through B, Q, M, W. The continuity source permits only displayed prefix dependence, the choice-sequence row exposes only the inspected prefix, and the modulus row bounds the window opened by $\mathsf{StreamName}^{\uparrow}$.

The output-facing part is read through R, E. Regular rational readback is downstream of the finite window, and the Real seal is downstream of that readback.

The structural rows H, C, P, N transport, replay, source, and name the same route. Since no excluded object occurs as a coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6030.3 (Brouwer finite choice-sequence real readback)**.** *Every Real-facing consumer of an accepted* $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{BrouwerContinuityPrinciple}^{\uparrow},\ \mathsf{ChoiceSequence}^{\uparrow} \longmapsto M \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through B, Q, M, W, R, E *before the structural rows* H, C, P, N *package the read. Thus the output read is determined by a displayed finite choice-sequence prefix and its modulus row, and it exports no completed branch, quotient stream equality, selected real representative, countable-choice selector, host continuity theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the continuity source and input prefix. The row B imports only the finite-prefix discipline of chapter 3465, and the row Q supplies only the displayed $\mathsf{ChoiceSequence}^{\uparrow}$ prefix.

Next read the modulus and stream window. The row M fixes the inspected prefix length, and W opens exactly that $\mathsf{StreamName}^{\uparrow}$ window, so downstream consumers cannot replace the route by an infinite branch or quotient stream equality.

Then read the analytic output. The $\mathsf{RegSeqRat}^{\uparrow}$ row R is obtained from the same finite window, and the $\mathsf{Real}^{\uparrow}$ seal E is available only after that readback has been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite route. Because the carrier contains no coordinate for countable choice, arbitrary bar induction, selected real representatives, host continuity, or ambient Real completeness, none of those objects is exported by the readback. □

Falsifiable prediction: Every accepted $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ Real-facing read displays a Brouwer continuity source row, ChoiceSequence finite prefix, StreamName prefix window, RegSeqRat readback, Real seal, finite modulus row, transport, continuation, provenance, and local naming before any output claim is public.

Independence witness: brouwer-continuity-principle, choice-sequence, streamname, regseqrat, real: $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the source continuity row, displayed choice-sequence prefix, modulus row, stream window, regular readback, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}$ packet is generated from a BrouwerContinuityPrinciple source row, a ChoiceSequence finite prefix, a StreamName prefix window, RegSeqRat readback, a RealUp seal, finite modulus data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6030.1, Definition 6030.2, and Definition 6030.3 seed the finite-prefix modulus route from Brouwer choice-sequence input to regular-rational and Real readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BrouwerFiniteChoiceSequenceModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, host functional continuity, selected infinite branches, quotient stream equality, countable choice, arbitrary bar induction, selected real representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix modulus exposure, StreamName window routing, RegSeqRat readback, RealUp seal nonescape, and TasteGate field faithfulness for BEDC.Derived.BrouwerFiniteChoiceSequenc

Chapter 6031

A Concrete Naming Certificate for $\mathsf{MinkowskiFunctional}^{\uparrow}$

Following Bourbaki and Rockafellar on locally convex spaces, the $\mathsf{MinkowskiFunctional}^{\uparrow}$ packet records the gauge functional of a displayed absorbing balanced convex neighbourhood as finite NameCert data. It sits over the convex-set source of chapter 208, the topological-vector-space operations of chapter 205, the locally convex neighbourhood ledger of chapter 3386, the seminormed-space consumer surface of chapter 5868, and the terminal scalar seal of chapter 38. The packet names the finite route by which a gauge-bound row is exposed; it does not import host real infima, quotient vector equality, choice of a neighbourhood basis, Hahn–Banach separation, or ambient completeness.

Definition 6031.1 (Minkowski functional carrier)**.** *A* $\mathsf{MinkowskiFunctional}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, V, A, B, L, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{ConvexSet}^{\uparrow}$ *source row naming the displayed convex subset whose gauge is being read.*
2. *V is the* $\mathsf{TopVecSpace}^{\uparrow}$ *operation row for addition, zero, negation, scalar action, and topological transport on the displayed vector carrier.*
3. *A is the absorbing-neighbourhood ledger row, recording the finite scale witnesses by which each inspected vector is admitted into a scalar multiple of X.*
4. *B is the balanced-neighbourhood ledger row, recording the scalar-stability witnesses needed before the gauge read may be transported.*
5. *L is the* $\mathsf{LocallyConvexSpace}^{\uparrow}$ *neighbourhood row tying X, A, B to the displayed locally convex source surface.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *gauge-bound row, exposing the finite scalar bound obtained from the displayed absorbing and balanced ledgers.*
7. *H records componentwise* hsame *transport for the convex source, vector-space operations, absorbing ledger, balanced ledger, locally convex row, scalar bound, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a vector request through V, X, A, B, L, R and back to the public gauge-bound boundary.*

9. *P is the* Pkg *provenance over* $\mathsf{MinkowskiFunctional}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{TopVecSpace}^{\uparrow}$, $\mathsf{LocallyConvexSpace}^{\uparrow}$, $\mathsf{SeminormedSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MinkowskiFunctional}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, V, A, B, L, R, H, C, P, N$. It has no coordinate for a host infimum, quotient-vector equality, selected neighbourhood basis, separation theorem, ambient topological-vector-space completion, or standard bridge.

Lean4 ✓ `BEDC.Derived.MinkowskiFunctionalUp.MinkowskiFunctionalTasteGate single carrier alignment`

Theorem 6031.2 (Minkowski functional NameCert obligations)**.** *For every accepted* $\mathsf{MinkowskiFunctional}^{\uparrow}$ *carrier $G = (X, V, A, B, L, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MinkowskiFunctional}^{\uparrow}}$ *surface consists exactly of convex-source admission through X, topological-vector-space operation exposure through V, absorbing-ledger exposure through A, balanced-ledger exposure through B, locally convex neighbourhood routing through L, scalar gauge-bound sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

No NameCert obligation is stored outside those displayed rows.

Proof. Project the carrier. The analytic source rows are precisely the convex source X, the topological-vector-space operations V, the absorbing ledger A, the balanced ledger B, the locally convex neighbourhood row L, and the real scalar bound R.

The gauge route first reads X inside the vector-space operation surface V. Absorption and balancedness are then visible as finite ledger rows, so the scalar bound is read only after the neighbourhood evidence has been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no excluded host analytic object is a coordinate of the packet, the listed rows exhaust the local NameCert surface. □

Theorem 6031.3 (Minkowski functional gauge route)**.** *Every* $\mathsf{SeminormedSpace}^{\uparrow}$ *or locally convex consumer of an accepted* $\mathsf{MinkowskiFunctional}^{\uparrow}$ *packet receives the gauge-bound read only through the finite route*

$$\mathsf{ConvexSet}^{\uparrow}, \mathsf{TopVecSpace}^{\uparrow} \longmapsto A, B \longmapsto \mathsf{LocallyConvexSpace}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the public boundary is a displayed RealUp scalar-bound row generated from the convex, absorbing, and balanced neighbourhood ledgers; it is not a host real infimum, quotient-vector equality, selected basis element, Hahn–Banach separation result, or completed seminorm theorem.

Proof. Start from the convex and vector-space rows. The source set X is read as a displayed convex packet, while V supplies only the finite operation and topology rows needed by the inspected vector request.

Next read the neighbourhood ledgers. The absorbing row A records the available scale witnesses, and the balanced row B records scalar stability. The locally convex row L ties those two ledgers to the same neighbourhood surface.

The scalar read is then sealed by R. It is a $\mathsf{Real}^{\uparrow}$ gauge-bound row obtained from the displayed ledgers, not an imported infimum over all host scalars.

Finally H, C, P, N transport, replay, source, and name the same route for downstream locally convex and seminormed consumers. None of the excluded analytic objects appears as a carrier coordinate, so no consumer can read them from this packet. □

Falsifiable prediction: Every accepted $\mathsf{MinkowskiFunctional}^{\uparrow}$ gauge-bound read displays a ConvexSetUp source, TopVecSpaceUp operations, absorbing and balanced neighbourhood ledgers, LocallyConvexSpaceUp routing, a RealUp scalar-bound row, componentwise hsame transport, Cont replay, provenance, and local naming before a downstream SeminormedSpaceUp consumer receives the boundary.

Independence witness: convexset, topvecspace, locally-convex-space, seminormed-space, real: $\mathsf{MinkowskiFunctional}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the convex source, vector-space operations, absorbing ledger, balanced ledger, locally convex route, scalar gauge-bound row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MinkowskiFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{MinkowskiFunctional}^{\uparrow}$ packet is generated from a ConvexSetUp source row, TopVecSpaceUp operation row, absorbing-neighbourhood ledger, balanced-neighbourhood ledger, LocallyConvexSpaceUp route, RealUp gauge-bound row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinkowskiFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6031.1, Definition 6031.2, and Definition 6031.3 seed the finite gauge-bound route from absorbing balanced convex neighbourhood data to RealUp scalar readback.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinkowskiFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host real infima, quotient-vector equality, selected neighbourhood bases, Hahn–Banach separation, ambient locally convex completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, convex-source exactness, absorbing and balanced ledger routing, RealUp gauge-bound sealing, nonescape, and TasteGate field faithfulness for BEDC.Derived.MinkowskiFunctionalUp.

Chapter 6032

A Concrete Naming Certificate for GroundCompilerStrengthMonotonicity$^{\uparrow}$

The GroundCompilerStrengthMonotonicity$^{\uparrow}$ packet realizes the strength-monotonicity theorem family of the GroundCompiler derivation-certificate recognizer as finite NameCert data for MetaCIC normalization audit consumers. It sits beside the flow-level round-trip packet of chapter 1094, the event-flow audit packet of chapter 829, the recognizer-flow packet of chapter 1048, the candidate-mediated socket of chapter 3566, and the route-budget packet of chapter 1360. The packet records when a lower-strength derivation-certificate subflow may support a lower accepted flow, without turning the compiler into a source of hidden proof evidence, retroactive promotion, or unconditional normalization.

Definition 6032.1 (GroundCompiler strength monotonicity carrier)**.** *A* GroundCompilerStrengthMonotonicity$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (N, H, L, A, D, R, G, B, C, P, M)$$

with the following displayed rows:

1. *N is the accepted object-flow row whose higher-strength acceptance is being inspected.*
2. *H is the higher-strength row.*
3. *L is the lower-strength row requested by the downstream consumer.*
4. *A is the accepted-flow witness row at strength H.*
5. *D is the recognized lower-strength derivation-certificate subflow row supporting L.*
6. *R is the route row tying D to A as a displayed subflow rather than as hidden compiler evidence.*
7. *G records the GroundCompiler gate-monotone read of* StrengthMonotonicityAtFlowLevel*.*
8. *B records the audit-map route budget consumed through* AuditMapRouteBudget$^{\uparrow}$ *before the lower-strength acceptance is public.*
9. *C records componentwise* hsame *transport and displayed* Cont *replay for N, H, L, A, D, R, G, B.*
10. *P is the* Pkg *provenance over* GroundCompilerStrengthMonotonicity$^{\uparrow}$*,* GroundCompilerFlowLevelRoundTrip$^{\uparrow}$*,* GroundCompilerEventFlowAudit$^{\uparrow}$*,* GroundCompilerRecognizerFlow$^{\uparrow}$*,* NormalizationCandidateSocket$^{\uparrow}$*,* AuditMapRouteBudget$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

11. M is the local $\mathsf{NameCert}_{\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $N, H, L, A, D, R, G, B, C, P, M$. It has no coordinate for retroactive promotion, hidden compiler state, proof reconstruction, global normalization, erased candidate evidence, or a route that bypasses the displayed lower-strength derivation-certificate subflow.

Theorem 6032.2 (GroundCompiler strength monotonicity NameCert obligations)**.** *For every accepted* $\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}$ *carrier $S = (N, H, L, A, D, R, G, B, C, P, M)$, the local* $\mathsf{NameCert}_{\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}}$ *surface consists exactly of accepted-object exposure through N, higher-strength exposure through H, lower-strength request exposure through L, accepted-flow evidence through A, recognized lower-strength derivation-certificate subflow exposure through D, subflow routing through R, gate-monotone strength readback through G, audit-map route-budget exposure through B, componentwise transport and replay through C, provenance through P, and local naming through M.*

No NameCert obligation is stored outside those displayed rows.

Proof. Project the carrier. The monotonicity source consists of the accepted object row N, the higher and lower strength rows H, L, the higher-strength accepted-flow row A, and the lower-strength derivation-certificate subflow row D.

The route row R keeps D visible as a subflow of A. Thus the lower-strength conclusion is not obtained by weakening the strength label alone; it is obtained only after the lower derivation-certificate evidence has been displayed.

The checked GroundCompiler theorem supplies the gate-monotone readback row G, while the audit-map budget row B keeps the event-flow, recognizer, certificate-gate, report-boundary, and cannot-claim routes visible.

Finally C, P, M transport, source, and name the same finite surface. Since no excluded object occurs as a coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6032.3 (GroundCompiler strength monotonicity route)**.** *Every MetaCIC normalization audit consumer that reads* $\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}$ *must pass through the finite route*

$$\mathsf{GroundCompilerFlowLevelRoundTrip}^{\uparrow}, \mathsf{GroundCompilerEventFlowAudit}^{\uparrow}, \mathsf{GroundCompilerRecognizerFlow}^{\uparrow} \longmapsto A, D, R$$

The route preserves the lower-strength derivation-certificate subflow before the lower accepted-flow read is public. It does not export hidden proof evidence, retroactive promotion, erased candidate normalization, host compiler state, or an unconditional closed-term normalization endpoint.

Proof. Begin with the GroundCompiler flow rows. The flow-level round-trip row keeps encoded flow readback attached to its source, the event-flow audit row separates compiler-facing event data from hidden inputs, and the recognizer-flow row exposes generated-recognizer admission before any certificate-gate read.

Next read the monotonicity rows. The accepted-flow witness A and lower derivation-certificate subflow D are connected by R, and the gate-monotone row G reads the checked strength-monotonicity theorem only for that displayed subflow relation.

The audit-map route budget B fixes the event-flow, recognizer, certificate-gate, report-boundary, and cannot-claim path before the result reaches MetaCIC normalization consumers. The normalization candidate socket then preserves candidate-mediated evidence rather than replacing it by an unconditional endpoint.

The structural rows C, P, M replay and name the same route. Because the carrier has no coordinate for hidden proof evidence, retroactive promotion, erased candidate normalization, host

compiler state, or unconditional closed-term normalization, those reads remain outside the packet. □

Falsifiable prediction: A GroundCompiler consumer that lowers strength without displaying a recognized lower-strength derivation-certificate subflow, the gate-monotone strength row, and the audit-map route budget is not an accepted $\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}$ packet.

Independence witness: groundcompilerflowlevelroundtrip, groundcompilereventflowaudit, groundcompilerrecognizerflow, normalization-candidate-socket, auditmaproutebudget: GroundCompilerStrengthMonotonic is not bijective to any listed sibling because it jointly carries higher and lower strength rows, accepted-flow evidence, lower derivation-certificate subflow evidence, gate-monotone readback, route-budget exposure, candidate-socket consumer routing, transport, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}}$.

Constructive story: A $\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}$ packet is generated from an accepted object-flow row, higher and lower strength rows, higher-strength accepted-flow evidence, recognized lower-strength derivation-certificate subflow evidence, subflow routing, the checked gate-monotone strength row, audit-map route-budget data, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6032.1, Definition 6032.2, and Definition 6032.3 seed the finite GroundCompiler strength-monotonicity packet for MetaCIC normalization audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GroundCompilerStrengthMonotonicity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.GroundCompiler.DerivCertGenerated.gate\_monotone\_strength`

Bridge status: none

Not claimed: This chapter does not close hidden proof evidence, compiler correctness, retroactive promotion, erased candidate normalization, global closed-term normalization, host compiler state, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subflow-route exactness, audit-map budget nonescape, candidate-socket consumer preservation, and TasteGate field faithfulness for BEDC.Derived.GroundCompilerStrengthMonotonicityU

Chapter 6033

A Concrete Naming Certificate for $\mathsf{YoungMeasure}^{\uparrow}$

Following Young and standard weak-convergence treatments, the $\mathsf{YoungMeasure}^{\uparrow}$ packet records parametrized limiting probability data as finite BEDC NameCert rows rather than as a quotient measure class or a selected subsequential limit. It sits after the terminal real-name surface of chapter 38, the probability-space carrier of chapter 184, the tight probability route of chapter 4008, and the regular-Cauchy completion lattice of chapter 5833. The packet gives Phase Real and Completion consumers a displayed interface for tight probability and real-sequence limiting data without asserting ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6033.1 (YoungMeasure carrier)**.** *A* $\mathsf{YoungMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (R, P, T, S, Q, E, H, C, K, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and observable-value row through which scalar test reads are sealed.*
2. *P is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row naming the probability carrier, measurable event ledger, and finite probability observations.*
3. *T is the tightness row, read through the Prokhorov-style tight probability packet before any limiting probability read is admitted.*
4. *S is the real-sequence window row, recording the finite schedule of observations whose limiting behavior the packet exposes.*
5. *Q is the regular-Cauchy readback row tying the displayed scalar observations to the completion-facing regular rational route.*
6. *E is the Young-measure evaluation row, recording only test-observable probability reads over displayed windows and tightness ledgers.*
7. *H records componentwise* hsame *transport for real, probability, tightness, sequence-window, regular-readback, evaluation, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a test-observable request through real endpoints, probability rows, tightness rows, sequence windows, regular readback, and evaluation.*
9. *K is the* Pkg *provenance over* $\mathsf{YoungMeasure}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *together with the cited tightness and regular-Cauchy dependency rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{YoungMeasure}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $R, P, T, S, Q, E, H, C, K, N$. It has no coordinate for quotient measure equality, selected subsequences, selected host limits, countable choice, arbitrary weak-star compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.YoungMeasureUp.YoungMeasureTasteGate single carrier alignment`

Theorem 6033.2 (YoungMeasure NameCert obligations)**.** *For every accepted* $\mathsf{YoungMeasure}^{\uparrow}$ *carrier*

$$Y = (R, P, T, S, Q, E, H, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{YoungMeasure}^{\uparrow}}$ *surface consists exactly of Real endpoint exposure, probability-source exposure, tightness exposure, real-sequence window exposure, regular-Cauchy readback, Young-measure evaluation exposure, componentwise transport, displayed replay, package provenance, and local naming. No obligation is stored outside those rows.*

Proof. Project the carrier of Definition 6033.1. The displayed coordinates are precisely the Real endpoint row, probability source row, tightness row, sequence-window row, regular-readback row, evaluation row, and structural rows.

The probability and tightness obligations are read from P and T. Thus an accepted packet exposes tight limiting data only after the probability carrier and its tightness ledger have been displayed.

The real-valued observation obligations are read from R, S, Q, E. The sequence window S and regular readback Q keep scalar limiting data in the finite completion corridor before the evaluation row E is presented.

Finally H, C, K, N transport, replay, source, and name the same route. Since no quotient measure equality, selected subsequence, selected limit, countable-choice schedule, weak-star compactness theorem, or ambient completeness row appears in the carrier, none can be exported by the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.YoungMeasureUp.YoungMeasureCarrier namecert obligations`

Theorem 6033.3 (YoungMeasure tightness handoff)**.** *Every tight probability read accepted by a* $\mathsf{YoungMeasure}^{\uparrow}$ *carrier factors through the displayed probability source row, tightness row, finite real-sequence windows, regular-Cauchy readback, and local structural rows before any evaluation row is visible. The route is*

$$P, T, S, Q \longmapsto R, E \longmapsto H, C, K, N.$$

It exports a finite tightness handoff for probability-valued limiting data, not a quotient measure equality, a selected subsequence, or a host compactness principle.

Proof. Use the obligation theorem Definition 6033.2. It exposes the probability source and tightness rows before the sequence-window and regular-readback rows can be inspected by a limiting-data consumer.

The evaluation row is downstream of those finite rows. A consumer reaches E only after the probability source P, tightness ledger T, sequence window S, regular readback Q, and Real endpoint row R have been displayed.

The structural rows then preserve the same handoff. Componentwise transport H, replay C, provenance K, and local naming N cannot add quotient measure equality, selected subsequences, or host compactness, because those are not carrier coordinates. □

Theorem 6033.4 (YoungMeasure Real completion boundary)**.** *No* $\mathsf{Real}^{\uparrow}$*-facing completion consumer of an accepted* $\mathsf{YoungMeasure}^{\uparrow}$ *carrier can bypass the displayed route through Real endpoints, probability-source rows, tightness ledgers, real-sequence windows, regular-Cauchy readbacks, evaluation rows, and local structural rows. In particular, the packet does not assert quotient measure equality, selected limits, selected subsequences, countable choice, arbitrary weak convergence, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier projection of Definition 6033.1. The only Real-facing row is R, and the only completion-facing scalar readback row is Q.

Apply the tightness handoff Definition 6033.3. It places the probability source and tightness ledger before the sequence-window, regular-readback, and evaluation rows.

The Real endpoint row is therefore a boundary for displayed scalar observations, not a completeness theorem. Since the carrier stores no quotient measure equality, selected limit, selected subsequence, countable-choice schedule, arbitrary weak-convergence theorem, or ambient completeness row, a completion consumer cannot obtain any of those principles from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{YoungMeasure}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{YoungMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6033.1, Definition 6033.2, Definition 6033.3, and Definition 6033.4 seed the finite Young-measure carrier, local obligation surface, tightness handoff, and Real-completion boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{YoungMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient measure equality, selected subsequences, selected host limits, countable choice, arbitrary weak-star compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Formal verification can encode carrier admission, probability-source exposure, tightness handoff, sequence-window readback, Real endpoint boundary, transport, replay, provenance, and local naming for BEDC.Derived.YoungMeasureUp.

Constructive story: A $\mathsf{YoungMeasure}^{\uparrow}$ packet is generated from Real endpoint rows, probability-source rows, tightness rows, real-sequence windows, regular-Cauchy readbacks, Young-measure evaluation rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6034

A Concrete Naming Certificate for $\mathsf{FiniteSupportStream}^{\uparrow}$

Following Bishop-style sequence spaces, the $\mathsf{FiniteSupportStream}^{\uparrow}$ packet records the finite-support stream route by which a displayed sequence supplies bounded-window approximants before any $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer reads a seal. It sits after the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name boundary of chapter 38, and the finite list carrier of chapter 36. The packet names an eventually-zero support bound as finite certificate data; it is not a quotient stream, countable choice schedule, selected limit, or ambient completeness principle.

Definition 6034.1 (Finite-support stream carrier)**.** *A* $\mathsf{FiniteSupportStream}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, B, L, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row for the stream windows inspected by the packet.*
2. *B is the finite support-bound row, recording the displayed index after which all stream entries are the zero row.*
3. *L is the* $\mathsf{List}^{\uparrow}$ *row listing the nonzero support entries and their finite window positions.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed entries and tail-zero boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the bounded stream windows and dyadic ledger.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, B, L, D, R have been exposed.*
7. *H records componentwise* hsame *transport for S, B, L, D, R, Q and the structural rows.*
8. *C records displayed* Cont *replay from a precision request through the source stream, support bound, list row, dyadic ledger, regular rational readback, and real seal boundary.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSupportStream}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSupportStream}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite support oracle, quotient equality of streams, a selected limit, countable choice, or an ambient real-completeness theorem.

Lean4 ✓ `BEDC.Derived.FiniteSupportStreamUp.FiniteSupportStreamTasteGate single carrier alignment`

Theorem 6034.2 (Finite-support stream NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSupportStream}^{\uparrow}$ *carrier*

$$F = (S, B, L, D, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSupportStream}^{\uparrow}}$ *surface consists exactly of stream source admission, finite support-bound admission, finite nonzero-entry list admission, dyadic tolerance compatibility, regular rational readback, terminal Real seal access, componentwise* hsame *stability, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6034.1. The analytic rows are exactly S, B, L, D, R, Q, and the structural rows are exactly H, C, P, N.

The support-bound row B and finite list row L force every inspected nonzero entry to be displayed before the tail-zero boundary is used. The dyadic ledger D then records the tolerance budget for the listed entries and for the zero tail.

The regular rational readback R and Real seal Q are downstream rows of that same finite route. Transport, replay, provenance, and local naming preserve the rows componentwise, so no quotient stream, selected limit, countable-choice schedule, or ambient completeness object is introduced. □

Theorem 6034.3 (Finite-support stream RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteSupportStream}^{\uparrow}$ *packet must read the finite route*

$$S, B, L, D \longmapsto R, Q, H, C, P, N.$$

Thus a finite-window approximant reaches regular rational readback and the Real seal only through the StreamName source, finite support bound, listed support entries, dyadic tolerance ledger, componentwise transport, displayed replay, provenance, and local NameCert rows.

Proof. Begin with a regular rational read request. By Definition 6034.1, the request first exposes the stream source S, support bound B, finite list row L, and dyadic tolerance ledger D.

Apply the obligation surface of Definition 6034.2. It requires R and Q to appear only after the source, support, list, and tolerance rows have been accepted.

Finally replay H, C, P, N. These rows preserve and name the same finite packet, so the handoff supplies bounded-window approximants to $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{Real}^{\uparrow}$ without quotient streams, selected limits, countable choice, or an ambient completeness principle. □

Falsifiable prediction: Every accepted $\mathsf{FiniteSupportStream}^{\uparrow}$ read factors through one StreamName source row, one finite support-bound row, one finite ListUp support-entry row, one DyadicRatCore tolerance ledger, one RegSeqRat readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: streamname, regseqrat, dyadicratcore, real, list: $\mathsf{FiniteSupportStream}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a stream source, a finite support bound, a listed nonzero support surface, dyadic tolerance data, regular rational readback, a Real-facing seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSupportStream}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteSupportStream}^{\uparrow}$ packet is generated from a StreamName source row, a finite support-bound row, a finite ListUp support-entry row, a DyadicRatCore tolerance ledger, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSupportStream}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6034.1, Definition 6034.2, and Definition 6034.3 seed the finite-support stream route for bounded-window regular rational and real-facing reads.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteSupportStream}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, support-bound exactness, dyadic tolerance compatibility, regular rational readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSupportStrea

Chapter 6035

A Concrete Naming Certificate for $\mathsf{EffectiveCoverCompactness}^{\uparrow}$

Following Lebesgue and Bishop–Bridges constructive compactness, the $\mathsf{EffectiveCoverCompactness}^{\uparrow}$ packet records the finite BEDC surface by which a compact metric carrier exposes cover data only through explicit finite subcovers, mesh bounds, and Lebesgue-number handoffs. It sits after the compact metric source of chapter 129, the finite-cover refinement surface of chapter 3430, and the Lebesgue-number finite-cover route of chapter 6016, while supplying the compact-uniform modulus consumer of chapter 2299 and the vision corridor of chapter 9237. The chapter names effective finite-cover compactness, not arbitrary open-cover compactness.

Definition 6035.1 (Effective cover compactness carrier)**.** *An* $\mathsf{EffectiveCoverCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, O, R, L, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, carrying the inspected compact metric name, metric balls, and finite request windows.*
2. *O is the cover row, listing the finitely observed basic opens and their metric radii.*
3. *R is the* $\mathsf{FiniteCoverRefinement}^{\uparrow}$ *row, recording the finite refinement and subcover choices that remain visible to downstream consumers.*
4. *L is the* $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$ *row, supplying the displayed Lebesgue-number witness for the same finite cover.*
5. *M is the compact-uniform handoff row consumed by* $\mathsf{CompactUniformModulus}^{\uparrow}$*.*
6. *H records componentwise* hsame *transport for compact metric data, cover rows, refinement rows, Lebesgue-number rows, compact-uniform handoff, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay from a cover or modulus request through K, O, R, L, M, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveCoverCompactness}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{FiniteCoverRefinement}^{\uparrow}$*,* $\mathsf{LebesgueNumberFiniteCover}^{\uparrow}$*,* $\mathsf{CompactUniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCoverCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, O, R, L, M, H, C, P, N$. It has no coordinate for arbitrary open-cover compactness, countable subcover extraction, choice of centers outside the displayed cover, ambient completeness, or a selected global modulus.

Theorem 6035.2 (Effective cover compactness NameCert obligations)**.** *For every accepted* EffectiveCoverCompactn *carrier*

$$E = (K, O, R, L, M, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EffectiveCoverCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric row, cover row, finite refinement row, Lebesgue-number witness row, compact-uniform handoff row, componentwise transport, displayed replay, provenance, and local naming. Every cover-facing consumer must read* K *and* O *before* R*, read* R *before* L*, and read* L *before the compact-uniform handoff* M*.*

Proof. Project the carrier of Definition 6035.1. The visible coordinates are exactly the compact metric source, cover ledger, finite refinement, Lebesgue-number witness, compact-uniform handoff, and structural rows.

The obligation order is forced by those coordinates. The finite refinement row R is meaningful only over the displayed cover row O and compact metric row K, while the Lebesgue-number row L is read from that same finite cover surface.

Finally M, H, C, P, N only hand off, transport, replay, source, and name the displayed finite-cover data. Since no other carrier coordinate exists, the local NameCert surface cannot export arbitrary open-cover compactness, a countable subcover selector, hidden centers, ambient completeness, or a selected global modulus. □

Theorem 6035.3 (Effective cover compactness finite-cover handoff)**.** *Every compact-uniform consumer of an accepted* $\mathsf{EffectiveCoverCompactness}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto \mathsf{LebesgueNumberFiniteCover}^{\uparrow} \longmapsto \mathsf{CompactUniformModulus}^{\uparrow}.$$

Thus the consumer sees the compact metric source row, displayed cover row, finite refinement row, Lebesgue-number witness, compact-uniform handoff, componentwise transport, replay, provenance, and local naming before any modulus-facing conclusion is available.

Proof. Begin with a compact-uniform consumer and apply Definition 6035.2. The consumer must first display the compact metric source and the finite cover row.

Next it reads the finite-cover refinement and Lebesgue-number rows. These are the existing finite-cover surfaces of chapter 3430 and chapter 6016, so the handoff remains a finite certificate rather than an arbitrary open-cover principle.

The compact-uniform row M is downstream of the same finite route. The structural rows H, C, P, N preserve and name that route only, so the consumer receives no hidden subcover choice, no countable cover extraction, no ambient completeness, and no selected global modulus outside the displayed certificate. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCoverCompactness}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EffectiveCoverCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6035.1, Definition 6035.2, and Definition 6035.3 seed the finite-cover compactness surface from compact metric data through finite refinement, Lebesgue-number witnesses, compact-uniform handoff, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EffectiveCoverCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, countable subcover extraction, hidden center choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-cover refinement visibility, Lebesgue-number exactness, compact-uniform handoff, transport, replay, provenance, and local naming for BEDC.Derived.EffectiveCoverCompactnessUp

Constructive story: An EffectiveCoverCompactness$^{\uparrow}$ packet is generated from a CompactMetric source row, displayed cover row, finite refinement row, Lebesgue-number witness row, compact-uniform handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theorem 6035.4 (Effective cover compactness sibling route)**.** *Every accepted* EffectiveCoverCompactness$^{\uparrow}$ *packet consumes the compact metric source of* CompactMetric$^{\uparrow}$*, the finite-cover refinement route, the Lebesgue-number finite-cover route, and the* FiniteEpsilonNet$^{\uparrow}$ *finite coverage pattern before it exports compact-uniform modulus data. Its visible dependency route is therefore*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{FiniteCoverRefinement}^{\uparrow} \longmapsto \mathsf{LebesgueNumberFiniteCover}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow} \longmapsto \mathsf{CompactU}$$

No consumer may bypass these finite rows by appealing to arbitrary open-cover compactness, hidden centers, countable subcover extraction, or ambient completeness.

Proof. Project the carrier of Definition 6035.1. The rows $K, O, R, L, M, H, C, P, N$ force the compact metric source and displayed cover to be read before the finite refinement and Lebesgue-number witnesses.

The finite coverage step follows the same visible pattern as FiniteEpsilonNet$^{\uparrow}$: the consumer only sees finitely listed coverage data and the named transport rows that preserve it. The compact-uniform row M is downstream of that finite route, while H, C, P, N only transport, replay, source, and name the displayed certificate.

Thus the chapter exports compact-uniform modulus data through the sibling route recorded above, and not through any hidden compactness principle outside the effective cover packet. □

Chapter 6036

A Concrete Naming Certificate for $\mathsf{FastCauchyWitness}^{\uparrow}$

Following Bishop–Bridges constructive completion, a fast Cauchy witness is finite evidence that a regular rational name carries an explicit tail modulus before any real-name seal is read. The $\mathsf{FastCauchyWitness}^{\uparrow}$ packet records that L10 Real and Completion route as a NameCert surface over the finite StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real boundary of chapter 38. It sits beside the window-step sibling chapter 1020 and the diagonal-limit sibling chapter 6002; it does not quotient streams, select an infinite representative, invoke countable choice, or compare host real values.

Definition 6036.1 (Fast Cauchy witness carrier)**.** *A* $\mathsf{FastCauchyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, D, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *source row whose finite windows are inspected by the Cauchy request.*
2. *W is the finite window ledger listing the source indices and rational entries read for the displayed precision.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including the positive dyadic scale used by the fast-tail request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the source windows and the same dyadic ledger.*
5. *M is the fast modulus row, recording that a request at dyadic scale 2^{-n} is answered by a displayed tail window whose pairwise rational distances are bounded by the chosen dyadic tolerance.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, exposed only after S, W, D, R, M have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, dyadic, readback, modulus, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a precision request through $S, W, D, R, M, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{FastCauchyWitness}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FastCauchyWitness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, W, D, R, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected infinite representatives, countable choice, host sequence equality, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6036.2 (Fast Cauchy witness NameCert obligations)**.** *For every accepted* $\mathsf{FastCauchyWitness}^{\uparrow}$ *carrier*

$$F = (S, W, D, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FastCauchyWitness}^{\uparrow}}$ *surface consists exactly of source StreamName admission for S, finite-window coverage for W, dyadic tolerance visibility for D, RegSeqRat readback for R, fast modulus exposure for M, Real seal routing for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No Fast Cauchy witness obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6036.1. The tuple $S, W, D, R, M, E, H, C, P, N$ is the whole public packet, so carrier admission and local naming are exactly the displayed rows.

The analytic route starts with the finite StreamName window W over the source row S. The dyadic tolerance row D supplies the scale for both the regular rational readback R and the fast modulus row M.

The terminal seal E is downstream of those rows. Transport, replay, provenance, and local naming through H, C, P, N preserve the same route, and no excluded quotient, selected representative, choice, host equality, or completeness coordinate occurs in the carrier. □

Lean4 ✓ `BEDC.Derived.FastCauchyWitnessUp.FastCauchyWitnessTasteGate single carrier alignment`

Theorem 6036.3 (Fast Cauchy witness modulus normalization)**.** *Every accepted* $\mathsf{FastCauchyWitness}^{\uparrow}$ *packet may normalize a displayed regular Cauchy modulus to the dyadic fast form*

$$n \longmapsto 2^{-n}$$

only by reading the finite route

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{FastCauchyWitness}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The normalized modulus row M is therefore a finite BEDC witness tied to the source windows S, W, dyadic ledger D, and regular readback R. It does not export quotient stream equality, host function extensionality, a selected infinite tail, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a precision request. The packet can answer it only through the source StreamName row S and the displayed finite window W.

The tolerance scale is then read from the DyadicRatCore row D. Using the same row, the RegSeqRat readback R supplies the regular rational data consumed by the modulus row M.

The row M records the fast dyadic budget as part of the finite packet rather than as an external choice of tail. The Real seal E can be read only after this route has been displayed, while H, C, P, N transport, replay, source, and name the same evidence.

Since the carrier contains no quotient stream, selected infinite tail, choice row, host equality, or completeness coordinate, the normalization remains a finite NameCert witness. □

Theorem 6036.4 (Fast Cauchy witness window determinacy)**.** *For accepted* $\mathsf{FastCauchyWitness}^{\uparrow}$ *packets*

$$F = (S, W, D, R, M, E, H, C, P, N)$$

and

$$F' = (S, W, D, R', M, E', H', C', P', N')$$

sharing the same $\mathsf{StreamName}^{\uparrow}$ *source row, finite window ledger,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and fast modulus row, every Real-facing read through either packet uses the same displayed window and tolerance route before reaching its* $\mathsf{Real}^{\uparrow}$ *seal. The packets may differ in readback, seal, transport, replay, provenance, or local naming rows, but no Real-facing consumer can distinguish their displayed window budget before the seal is read.*

Proof. Project both carriers using Definition 6036.1. The shared rows S, W, D, M are the only source, window, tolerance, and fast-modulus rows available to either accepted packet.

Apply the NameCert obligations of Definition 6036.2. They force every Real-facing route to read the StreamName source, finite window ledger, dyadic tolerance ledger, and fast modulus row before the terminal Real seal is exposed.

Use modulus normalization, Definition 6036.3. It keeps the Real seal downstream of the finite route through S, W, D, M, so the later rows R, E, H, C, P, N and R', E', H', C', P', N' cannot introduce a second displayed tail budget.

Therefore both packets present the same window and tolerance route to every Real-facing read. Since neither carrier contains a quotient stream, selected tail, or ambient completeness row, no excluded object can distinguish the shared window before the Real seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{FastCauchyWitness}^{\uparrow}}$.

Constructive story: A $\mathsf{FastCauchyWitness}^{\uparrow}$ packet is generated from a StreamName source row, finite windows, a DyadicRatCore tolerance ledger, a RegSeqRat readback row, a fast modulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FastCauchyWitness}^{\uparrow}}$, seed)

Scope closed: Definition 6036.1, Definition 6036.2, Definition 6036.3, and Definition 6036.4 seed the finite fast-Cauchy witness route from StreamName windows through dyadic tolerance and RegSeqRat readback to a terminal RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FastCauchyWitness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected infinite tails, countable choice, host sequence equality, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source admission, finite-window coverage, dyadic tolerance visibility, RegSeqRat readback, fast modulus exposure, Real seal routing, transport stability, replay stability, and non-escape for BEDC.Derived.FastCauchyWitnessUp.

Chapter 6037

A Concrete Naming Certificate for $\mathsf{SingularValueDecomposition}^{\uparrow}$

Following Golub–Van Loan and Horn–Johnson, the $\mathsf{SingularValueDecomposition}^{\uparrow}$ packet records the finite spectral certificate by which a displayed matrix is factored through two finite orthonormal frames and a nonnegative diagonal spectrum. It consumes the finite matrix carrier of chapter 49, the inner-product rows of chapter 86, the finite orthonormal-basis packet of chapter 4758, the Frobenius ledger of chapter 5100, the rank accounting of chapter 3438, and the triangular factorization route of chapter 4240. The chapter names only finite displayed rows: it is not a host spectral theorem, not a choice of bases outside the packet, not quotient matrix equality, and not an ambient completeness claim.

Definition 6037.1 (Singular-value decomposition carrier)**.** *A* $\mathsf{SingularValueDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, U, V, D, L, R, F, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *A is the input* $\mathsf{Mat}^{\uparrow}$ *row, exposing the finite row-column table, scalar endpoint ledger, and shape data of the matrix being decomposed.*
2. *U is the left finite orthonormal frame row, read through* $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$ *and the source* $\mathsf{InnerProduct}^{\uparrow}$ *rows.*
3. *V is the right finite orthonormal frame row, read through the same finite orthonormal-basis and inner-product discipline on the domain side.*
4. *D is the nonnegative diagonal spectrum row, listing the displayed singular-value endpoints, diagonal-slot witnesses, zero off-diagonal witnesses, and ordering ledger.*
5. *L is the finite multiplication witness*
$$A = UDV^*$$
read as an equality of displayed $\mathsf{Mat}^{\uparrow}$ *multiplication rows, not as a quotient equality in a host matrix algebra.*
6. *R is the* $\mathsf{RankNullity}^{\uparrow}$ *row identifying the nonzero diagonal slots with the displayed rank ledger.*
7. *F is the* $\mathsf{FrobeniusNorm}^{\uparrow}$ *row that reads the square-sum of entries through the displayed diagonal spectrum ledger.*

8. *Q is the* $\mathsf{QRDecomposition}^{\uparrow}$ *handoff row used by constructive factorization consumers before the orthonormal-frame and diagonal rows are sealed.*
9. *T records refusal rows for host spectral calculus, selected eigenbasis data, quotient matrix equality, ambient completeness, and any basis choice not displayed in U or V.*
10. *H records componentwise* hsame *transport for the matrix, the two frames, the diagonal spectrum, the multiplication witness, the rank and norm rows, the QR handoff, refusal rows, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from a matrix-factorization request through $A, U, V, D, L, R, F, Q, T, H$.*
12. *P is the* Pkg *provenance over* $\mathsf{SingularValueDecomposition}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$, $\mathsf{FrobeniusNorm}^{\uparrow}$, $\mathsf{RankNullity}^{\uparrow}$, $\mathsf{QRDecomposition}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{SingularValueDecomposition}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. In particular, changing a host basis, replacing the displayed diagonal list by an extensionally equal external spectrum, or reading equality outside the multiplication witness L is invisible to the carrier.

Lean4 ✓ `BEDC.Derived.SingularValueDecompositionUp.SingularValueDecompositionTasteGate single carrier alignment`

Theorem 6037.2 (Singular-value decomposition NameCert obligations)**.** *For every accepted* SingularValueDecompo *carrier*

$$S = (A, U, V, D, L, R, F, Q, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SingularValueDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Mat}^{\uparrow}$ *row A, the left and right finite orthonormal frame rows U, V, and the nonnegative diagonal spectrum row D;*
2. *classifier comparison through the same matrix, frame, diagonal, multiplication, rank, norm, QR, refusal, transport, replay, provenance, and local naming rows;*
3. *stability under componentwise* hsame *transport, preserving the frame rows, diagonal slots, multiplication witness, and non-escape ledger;*
4. *exactness of the displayed multiplication witness L, requiring each consumer to read the finite matrix route $A = UDV^*$ before exporting a factorization claim;*
5. *ledger obligations through R, F, Q, T, C, P, N, recording rank, Frobenius square-sum, constructive factorization, refusal, replay, source, and local naming without importing host spectral data.*

No singular-value-decomposition obligation is stored outside these rows.

Proof. Project the carrier of Definition 6037.1. The displayed matrix row A, orthonormal frame rows U, V, and diagonal spectrum row D are the only admissible factor data.

The classifier is componentwise on the listed rows. Transport through H can replace displayed endpoints only when the corresponding $\mathsf{Mat}^{\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, and finite-frame classifiers already recognize the replacement.

The exactness obligation is the multiplication row L. A consumer therefore receives the factorization only after the finite matrix multiplication witnesses have replayed the route $A = UDV^*$.

The remaining rows expose the rank ledger, Frobenius ledger, QR handoff, refusal boundary, replay, provenance, and local naming. Since each of these rows is already part of the carrier, the five listed obligations exhaust the certificate. □

Theorem 6037.3 (Finite MatrixUp handoff for singular-value decomposition)**.** *Every accepted* $\mathsf{SingularValueDecomposition}^{\uparrow}$ *packet exports a finite* $\mathsf{Mat}^{\uparrow}$ *handoff only through the route*

$$\mathsf{Mat}^{\uparrow} \longrightarrow \mathsf{InnerProduct}^{\uparrow} \longrightarrow \mathsf{FiniteOrthonormalBasis}^{\uparrow} \longrightarrow \mathsf{FrobeniusNorm}^{\uparrow} \longrightarrow \mathsf{RankNullity}^{\uparrow} \longrightarrow \mathsf{QRDecomposition}^{\uparrow}.$$

The handoff exposes the input matrix, left and right finite orthonormal frames, nonnegative diagonal spectrum, multiplication witness, norm ledger, rank ledger, and QR-facing construction row. It exports no host spectral theorem, no selected basis outside the packet, no quotient matrix equality, and no ambient completeness principle.

Lean4 ✓ `BEDC.Derived.SingularValueDecompositionUp.SingularValueDecompositionFiniteMatrixHandoff`

Proof. Begin with an accepted carrier $S = (A, U, V, D, L, R, F, Q, T, H, C, P, N)$. The matrix endpoint is A, so every factorization read begins at the displayed $\mathsf{Mat}^{\uparrow}$ table.

The rows U and V are read through $\mathsf{InnerProduct}^{\uparrow}$ and $\mathsf{FiniteOrthonormalBasis}^{\uparrow}$, making both frames finite packet data rather than external basis choices. The diagonal spectrum row D is then tied to those frames by the multiplication witness L.

The Frobenius row F and rank row R read the diagonal ledger as finite square-sum and nonzero-slot data. The QR row Q records the constructive factorization route consumed before the SVD packet is sealed.

The refusal ledger T and structural rows H, C, P, N keep the route inside the named finite packet. Thus the exported handoff is exactly the finite MatrixUp route described in the statement. □

Closure status for $\mathsf{NameCert}_{\mathsf{SingularValueDecomposition}^{\uparrow}}$**.**

Constructive story: A $\mathsf{SingularValueDecomposition}^{\uparrow}$ packet is generated from a displayed MatrixUp table, left and right finite orthonormal frame rows, InnerProductUp comparisons, a nonnegative diagonal spectrum row, a finite matrix multiplication witness, RankNullityUp and FrobeniusNormUp ledgers, a QRDecompositionUp handoff, refusal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SingularValueDecomposition}^{\uparrow}}$, seed)

Scope closed: Definition 6037.1, Definition 6037.2, and Definition 6037.3 seed the finite matrix spectral certificate over displayed orthonormal frames and diagonal spectrum rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SingularValueDecomposition}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close a host spectral theorem, selected bases outside the packet, quotient matrix equality, infinite-dimensional singular-value theory, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier comparison, transport stability, multiplication exactness, rank and norm ledgers, QR handoff, refusal non-escape, replay, provenance, and local naming for BEDC.Derived.SingularValueDecompositionUp.

Chapter 6038

A Concrete Naming Certificate for Rule110GliderPhaseClassifier$^{\uparrow}$

The Rule110GliderPhaseClassifier$^{\uparrow}$ packet realizes the glider-phase classifier branch of chapter 9390 as finite Rule 110 NameCert data. It sits after the cellular automaton carrier of chapter 934, the finite pattern catalog of chapter 995, the Rule 110 pattern classifier of chapter 5844, and the glider catalog of chapter 1045. The packet names row windows, catalog tags, phase offsets, local-match witnesses, replay rows, provenance, and local naming; it does not assert an unbounded orbit classifier, a host visual recognizer, or a bijection between glider tags and phase-offset packets.

Falsifiable prediction: Every accepted Rule110GliderPhaseClassifier$^{\uparrow}$ packet exposes one finite local window, one catalog tag, one phase-offset row, one displayed local-match row, one replay row, provenance, and local NameCert data; a packet that assigns a phase without such a finite local window matching exactly one catalog tag is outside the carrier.

Independence witness: rule110-pattern-classifier, rule110glidercatalog, cellularpatterncatalog, cellularorbitsubstrate: none is bijective to Rule110GliderPhaseClassifier$^{\uparrow}$ because this packet adds a phase-offset row and a non-escape ledger to the finite glider-tag read. The pattern classifier carries broader Rule 110 tags, the glider catalog carries phase-period-displacement rows without the extra classifier seal, the cellular pattern catalog carries generic phase labels, and the cellular orbit substrate carries orbit observations without the local phase-offset classifier.

Definition 6038.1 (Rule 110 glider phase classifier carrier)**.** *A* Rule110GliderPhaseClassifier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (R, W, K, O, M, E, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* CellularAutomaton$^{\uparrow}$ *Rule 110 rule-table and finite orbit-window source row.*
2. *W is the finite row-window row, including the start coordinate, width, time index, and bit string inspected by the classifier.*
3. *K is the* RuleOneTenGliderCatalog$^{\uparrow}$ *catalog-tag row naming the glider or ether family whose local phase is being read.*
4. *O is the phase-offset row, recording the finite offset inside the chosen catalog period and the displayed phase string attached to that offset.*

5. *M is the local-match row, witnessing that the bits of W match the phase string of O under the neighbor discipline of K.*
6. *E is the* $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ *and* $\mathsf{CellularPatternCatalog}^{\uparrow}$ *exposure row that keeps the broader tag classifier visible before the phase classifier exports its local mark.*
7. *L is the non-escape ledger excluding hidden orbit semantics, host visual recognition, phase choice outside O, and any second catalog tag for the same accepted local window.*
8. *H records componentwise* hsame *transport for the rule row, window, catalog tag, phase offset, local match, exposure row, non-escape ledger, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a phase-classification request through R, W, K, O, M, E, L, H.*
10. *P is the* Pkg *provenance over* $\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}$, $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularPatternCatalog}^{\uparrow}$, $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$, $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. Agreement of a glider tag without agreement of the phase-offset row O is not enough to classify two accepted phase packets together.

Theorem 6038.2 (Rule 110 glider phase classifier obligations)**.** *For every accepted* $\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}$ *carrier*

$$G = (R, W, K, O, M, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission, catalog-tag exposure, phase-offset exactness, local-match exactness, replay discipline, provenance, non-escape, and local naming rows. Every accepted phase mark factors through*

$$R \longmapsto W \longmapsto K \longmapsto O \longmapsto M \longmapsto E$$

with L, H, C, P, N attached. The packet exports no hidden orbit classifier, no host visual recognizer, no second catalog tag for the same accepted local window, and no phase choice absent from the displayed offset row.

Proof. Carrier admission is projection from Definition 6038.1. The displayed tuple lists the Rule 110 row, finite window, catalog tag, phase offset, local match, exposure, refusal, transport, replay, provenance, and local naming rows.

The catalog-tag row K must be read before the phase-offset row O, and the local-match row M checks the bits of W against that exact phase string. Thus a phase read cannot be detached from the catalog tag or from the finite window that witnessed it.

The exposure row E keeps the broader $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$ and $\mathsf{CellularPatternCatalog}^{\uparrow}$ routes visible. The non-escape ledger L then excludes host visual recognition, hidden orbit semantics, and any second tag or phase source outside the displayed carrier.

The structural rows H, C, P, N transport, replay, source, and name this same route. Hence the rows listed in the statement exhaust the local certificate. □

Lean4 ✓ `BEDC.Derived.RuleOneTenGliderPhaseClassifierUp.TasteGate.RuleOneTenGliderPhaseClassifierTasteGa`
`single carrier alignment`

Theorem 6038.3 (Rule 110 glider phase classifier mark handoff)**.** *Every* $\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}$ *handoff to the mark discipline exports a finite local classifier mark only after the* $\mathsf{CellularAutomaton}^{\uparrow}$, $\mathsf{CellularPatternCatalog}^{\uparrow}$, $\mathsf{RuleOneTenPatternClassifier}^{\uparrow}$, *and* $\mathsf{RuleOneTenGliderCatalog}^{\uparrow}$ *sibling rows*

have been displayed. The exported mark is the tuple of one local window, one catalog tag, one phase offset, one local-match witness, replay, provenance, and local NameCert *row; it is not an infinite orbit fact, not an external catalog lookup, and not a host-side perceptual judgment.*

Proof. Start with an accepted carrier $G = (R, W, K, O, M, E, L, H, C, P, N)$. The $\mathsf{CellularAutomaton}^{\uparrow}$ row R supplies the local Rule 110 source, and the finite window W is the only row inspected by the classifier.

The pattern-catalog and Rule 110 pattern-classifier route is retained in E, while the glider-catalog row K supplies the tag family. The phase-offset row O and local-match row M then refine that tag into one displayed phase mark.

Finally L, H, C, P, N preserve the non-escape ledger, replay route, provenance, and local name. The mark handoff therefore contains exactly the finite rows named in the statement. □

Lean4 ✓ `BEDC.Derived.RuleOneTenGliderPhaseClassifierUp.TasteGate.RuleOneTenGliderPhaseClassifierCarrier mark handoff`

The Lean TasteGate namespace at `lean4/BEDC/Derived/RuleOneTenGliderPhaseClassifierUp/TasteGate.` must supply $\mathsf{Nontrivial\ Rule110GliderPhaseClassifier}^{\uparrow}$ and $\mathsf{FieldFaithful\ Rule110GliderPhaseClassifier}^{\uparrow}$ instances by exhibiting distinct accepted and refused phase packets and by matching on the rows $R, W, K, O, M, E, L, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}$ packet is generated from a Rule 110 automaton row, a finite local window, a catalog tag, a phase-offset row, a displayed local-match row, pattern-catalog exposure, non-escape ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6038.1, Definition 6038.2, and Definition 6038.3 seed the finite glider-phase classifier packet for Rule 110 local windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Rule110GliderPhaseClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unbounded Rule 110 orbit classification, host visual recognition, universality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, catalog-tag exposure, phase-offset exactness, local-match exactness, pattern-catalog exposure, non-escape, transport, replay, provenance, local naming, and TasteGate coverage for BEDC.Derived.RuleOneTenGliderPhaseClassifierUp.

Chapter 6039

A Concrete Naming Certificate for $\mathsf{SeparatedCompletionMonad}^{\uparrow}$

Following Bishop–Bridges constructive completion practice and Mac Lane's monad language, the $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ packet records the L10 completion route as a separated unit-and-bind interface over finite BEDC rows. It sits after the separated completion carrier of chapter 1612, the completion monad schedule of chapter 378, the stream-name observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names only the separated monad interface needed by completion-functor and filter-completion consumers; it does not assert quotient completion, selected representatives, countable choice, or a host limit operator.

Definition 6039.1 (Separated completion monad carrier)**.** *A* $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, M, U, B, W, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{SeparatedCompletion}^{\uparrow}$ *admission row, exposing the separated comparison boundary before any monad law is public.*
2. *M is the* $\mathsf{CauchyCompletionMonad}^{\uparrow}$ *schedule row, naming the unit and bind requests as finite completion-facing data.*
3. *U is the unit row, carrying the displayed source-name admission into the separated completion surface.*
4. *B is the bind row, carrying a completion-facing map only through the same separated completion and Cauchy-completion schedule.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row read before any completion-facing Real seal is consumed.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the stream and regular-rational reads.*
8. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only as a displayed name boundary.*
9. *H records componentwise* hsame *transport for separated completion, monad schedule, unit, bind, stream, regular-rational, dyadic, Real-seal, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from separated admission through unit, bind, finite windows, regular-rational readback, dyadic tolerance, Real seal, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{SeparatedCompletionMonad}^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionMonad}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, M, U, B, W, R, D, L, H, C, P, N$. It has no coordinate for quotient completion equality, selected representatives, countable choice, a chosen host limit, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.SeparatedCompletionMonadUp.SeparatedCompletionMonadTasteGate single carrier alignment`

Theorem 6039.2 (Separated completion monad NameCert obligations)**.** *For every accepted* SeparatedCompletionM *carrier $S = (A, M, U, B, W, R, D, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SeparatedCompletionMonad}^{\uparrow}}$ *surface consists exactly of separated admission, Cauchy-completion monad schedule exposure, unit admission, bind replay, finite stream windows, regular-rational readback, dyadic tolerance accounting, terminal Real sealing, componentwise transport, displayed continuation replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6039.1. The separated row A is read before the monad schedule M, so equality-facing completion reads cannot bypass the separated boundary.

The unit row U and bind row B are then inspected through the same schedule. Their finite observations are forced through W, R, D, L, which exposes stream windows, regular-rational readback, dyadic tolerance, and the terminal Real seal as displayed rows.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no quotient completion, selected representative, countable-choice schedule, host limit, or ambient completeness coordinate is present, the NameCert obligations are exhausted by the listed rows. □

Theorem 6039.3 (Separated completion monad unit and bind laws)**.** *Every unit or bind law exported by an accepted* $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ *packet is read as equality of the displayed finite ledger over $A, M, U, B, W, R, D, L, H, C, P, N$. The unit law admits source names only through U after separated admission, and the bind law composes completion-facing reads only through B after the same stream, regular-rational, dyadic, and Real-seal rows have been exposed.*

Proof. For the unit side, read A and M before U. The source name enters the completion surface only after the separated boundary and Cauchy-completion schedule have been displayed.

For the bind side, read B together with W, R, D, L. A completion-facing map can be replayed only against the visible finite windows, regular-rational readback, dyadic tolerance ledger, and Real seal already carried by the packet.

Componentwise transport by H and replay by C identify the two law reads only as equality of the same finite ledger. The provenance and local naming rows prevent the law from being replaced by quotient completion equality, selected representatives, or a host limit operation. □

Theorem 6039.4 (Separated completion monad idempotent reflection)**.** *An accepted* $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ *packet may be consumed as an idempotent separated-reflection surface only after the separated completion row A, the monad schedule M, the unit row U, the bind row B, and the L10 observation rows W, R, D, L have all been displayed. The reflection is therefore a finite NameCert route for repeated separated completion, not a quotient collapse of completion points and not a choice-selected limit operation.*

Proof. Begin with separated admission. The row A supplies the comparison boundary that repeated completion reads must share, while M records that the read is governed by the same completion monad schedule.

The unit and bind rows U, B supply the repeated-completion interface. Because both rows are tied to W, R, D, L, the second completion read has access only to the same finite stream windows, regular-rational readback, dyadic tolerances, and Real seal.

Transport, replay, provenance, and local naming keep the repeated read inside one packet. Thus the idempotent-reflection consumer receives a finite separated monad surface and receives no quotient collapse, selected representative, selected limit, countable-choice schedule, or ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ packet must expose separated admission, Cauchy-completion monad schedule, unit, bind, stream, regular-rational, dyadic, Real-seal, transport, replay, provenance, and NameCert rows before a completion-functor or filter-completion consumer may read it.

Independence witness: separated-completion, cauchy-completion-monad, stream-name, regular-rational, dyadic-core, real-seal: $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries separated admission, unit-and-bind scheduling, L10 observation rows, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparatedCompletionMonad}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparatedCompletionMonad}^{\uparrow}$ packet is generated from separated completion admission, a CauchyCompletionMonad unit-and-bind schedule, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionMonad}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6039.1, Definition 6039.2, Definition 6039.3, and Definition 6039.4 seed the finite separated unit-and-bind route for L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparatedCompletionMonad}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, selected limits, countable choice, ambient completeness of $\mathsf{Real}^{\uparrow}$, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, separated boundary routing, unit exactness, bind exactness, idempotent reflection routing, transport, replay, provenance, and local naming for BEDC.Derived.SeparatedCompletionMonadUp.

Chapter 6040

A Concrete Naming Certificate for $\mathsf{MorseBoundary}^{\uparrow}$

Following Gromov's boundary viewpoint and Bridson–Haefliger's treatment of hyperbolic and CAT(0) geometry, the $\mathsf{MorseBoundary}^{\uparrow}$ packet records stable geodesic directions as finite BEDC NameCert data. It sits after the finite boundary-shadow route of chapter 3813, the horofunction boundary comparison of chapter 6025, the metric-space carrier of chapter 4230, and the geodesic metric surface of chapter 2213. The packet realizes the hyperbolic observer-expansion route of chapter 9463 as finite geodesic stability and fellow-traveller NameCert data; it does not complete an ideal boundary, quotient geodesic rays, select global representatives, or import host hyperbolic geometry.

Definition 6040.1 (Morse boundary carrier)**.** *A* $\mathsf{MorseBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, G, S, F, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Met}^{\uparrow}$ *source row, naming the finite metric comparison surface used by the packet.*
2. *G is the geodesic row, read through* $\mathsf{Geodesic}^{\uparrow}$ *and the* $\mathsf{GeodesicMetric}^{\uparrow}$ *surface, exposing finite segment incidence and endpoint data.*
3. *S is the Morse-stability row, recording the displayed stability gauge for geodesic or quasi-geodesic detours.*
4. *F is the fellow-traveller row, recording finite comparison windows in which stable representatives remain visibly close.*
5. *B is the boundary-read row, connecting the stable geodesic route to* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *and* $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$ *consumers.*
6. *H records componentwise* hsame *transport for metric, geodesic, stability, fellow-traveller, boundary-read, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from metric source through geodesic incidence, stability windows, fellow-traveller comparison, boundary readback, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{MorseBoundary}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicHorofunctionBoundary}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Geodesic}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{MorseBoundary}^\uparrow}$ *row.*
The classifier compares accepted packets only through $X, G, S, F, B, H, C, P, N$. It has no coordinate for a completed ideal boundary, quotient equality of rays, selected global geodesics, host compactification, or ambient hyperbolic model.

Lean4 ✓ `BEDC.Derived.MorseBoundaryUp.MorseBoundaryTasteGate single carrier alignment`

Theorem 6040.2 (Morse boundary NameCert obligations)**.** *For every accepted* $\mathsf{MorseBoundary}^\uparrow$ *carrier $M = (X, G, S, F, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MorseBoundary}^\uparrow}$ *surface consists exactly of metric-source admission, geodesic incidence exposure, Morse-stability gauge display, fellow-traveller window comparison, boundary readback, componentwise transport, displayed continuation replay, package provenance, and local naming. Every boundary consumer must inspect these rows before it treats the packet as a Morse boundary read.*

Proof. Project the carrier of Definition 6040.1. The metric row X fixes the finite source surface, and the geodesic row G exposes the displayed segment and endpoint data consumed by the packet.

Read S and F next. The stability row records the Morse gauge, while the fellow-traveller row records the finite windows in which representative routes remain visibly close.

The boundary row B then hands the stable route to boundary-shadow and horofunction-boundary consumers. The structural rows H, C, P, N transport, replay, source, and name the same finite data, so the packet exports no quotient ray, selected representative, host compactification, or ambient hyperbolic model. □

Theorem 6040.3 (Morse boundary geodesic stability)**.** *Every accepted* $\mathsf{MorseBoundary}^\uparrow$ *read exposes geodesic stability as the finite route*

$$\mathsf{Met}^\uparrow \longmapsto \mathsf{Geodesic}^\uparrow \longmapsto S \longmapsto F \longmapsto B.$$

The route first displays metric and geodesic rows, then records a Morse-stability gauge, then records fellow-traveller windows, and only then exports a boundary read to $\mathsf{HyperbolicBoundaryShadow}^\uparrow$ *or* $\mathsf{HyperbolicHorofunctionBoundary}^\uparrow$ *consumers.*

Proof. Start with X and G. These rows identify the metric source and the geodesic incidence data before any stability claim is public.

The stability row S binds the detour-control gauge to that displayed geodesic route. The fellow-traveller row F then records the finite comparison windows that make the stable route readable by downstream boundary consumers.

Finally B, H, C, P, N export the boundary read, transport it componentwise, replay the displayed route, source it by package provenance, and name it locally. Since the export is built only from these rows, it is a finite geodesic-stability surface rather than a completed boundary or quotient of rays. □

Falsifiable prediction: Every accepted $\mathsf{MorseBoundary}^\uparrow$ packet must expose metric-source, geodesic, Morse-stability, fellow-traveller, boundary-read, transport, replay, provenance, and NameCert rows before a boundary consumer may read stable geodesic direction data.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-horofunction-boundary, metric-space, geodesic: $\mathsf{MorseBoundary}^\uparrow$ is not bijective to any listed sibling because it jointly carries metric-source admission, geodesic incidence, stability gauge, fellow-traveller windows, boundary readback, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MorseBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MorseBoundary}^{\uparrow}$ packet is generated from MetricSpace source rows, Geodesic and Geodesic-Metric incidence rows, a Morse-stability gauge, fellow-traveller comparison windows, boundary-shadow and horofunction-boundary readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MorseBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6040.1, Definition 6040.2, and Definition 6040.3 seed the finite geodesic-stability and fellow-traveller route for Morse boundary consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MorseBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed ideal boundaries, quotient equality of rays, selected global geodesic representatives, host compactification, ambient hyperbolic geometry, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source routing, geodesic incidence, stability-gauge exactness, fellow-traveller comparison, boundary readback, transport, replay, provenance, and local naming for BEDC.Derived.MorseBoundaryUp.

Chapter 6041

A Concrete Naming Certificate for $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$

Following Bishop and Bridges on constructive completions, the $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ packet records the L10 universal-property neighbour of the Real completion interface as finite NameCert data. It consumes the dyadic tolerance core of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the completion interface of chapter 1844, and the Bishop regular Cauchy completion route of chapter 1639. Its universal-property row is consumed by the Bishop completion comparison square of chapter 6057 and by the Cauchy completion functor route of chapter 423, so the extension-uniqueness packet is a visible L10 lattice node rather than an isolated completion label. The packet names extension and uniqueness rows for completion-facing consumers without asserting quotient streams, selected limits, host equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6041.1 (Real completion universal-property carrier)**.** *A* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, S, R, B, E, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *dense-source row, exposing only named dyadic tolerances and dyadic source observations.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which completion-facing source reads are scheduled.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-readback row attached to the same stream windows and dyadic tolerances.*
4. *B is the* $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$ *handoff row, recording the regular Cauchy completion route before any terminal Real seal is consumed.*
5. *E is the* $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$ *extension row, displaying the completion-interface read that receives D, S, R, B.*
6. *U is the universal-property row: it records existence and uniqueness of the displayed extension only as a finite route over D, S, R, B, E.*
7. *H records componentwise* hsame *transport for the dyadic, stream, regular-readback, completion, interface, universal-property, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a dense-source request through $D, S, R, B, E, U, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{BishopRegularCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyRealCompletionInterface}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionUniversalProperty}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $D, S, R, B, E, U, H, C, P, N$. It contains no coordinate for quotient streams, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or bridge credentials.*

Theorem 6041.2 (Real completion universal-property NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier $G = (D, S, R, B, E, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionUniversalP}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed dyadic source, stream-window, regular-readback, Bishop regular Cauchy completion, completion-interface, universal-property, transport, replay, provenance, and local naming rows;*
2. *dense-source exposure, requiring every extension read to project D before inspecting the completion-interface row;*
3. *stream-window exactness, requiring S to schedule the finite observations consumed by the regular-readback and completion handoff rows;*
4. *regular-readback handoff, requiring R and then B to mediate between the finite windows and the completion-facing interface;*
5. *extension existence, requiring E and U to display a completion-interface extension over the same D, S, R, B coordinates;*
6. *extension uniqueness, requiring any two accepted extension reads over the same displayed dyadic, stream, regular-readback, and terminal Real-seal rows to share the same completion-facing route;*
7. *non-escape, requiring H, C, P, N to transport, replay, source, and name no quotient stream, selected limit, host equality, countable-choice schedule, ambient Real-completeness theorem, Lean verification, or bridge credential.*

No local obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6041.1. The accepted tuple displays exactly the dyadic source, stream-window, regular-readback, completion, interface, universal-property, transport, replay, provenance, and local naming rows.

The dense-source and stream-window clauses follow from the order of the displayed route. An extension read must first expose D and then schedule finite observations through S before the regular readback or completion interface can be consumed.

The regular-readback handoff is carried by R and B. These rows bind the stream windows and dyadic tolerances to the Bishop regular Cauchy completion route, so the interface cannot choose a quotient representative or host limit.

The extension rows E and U then record existence and uniqueness only over the fixed D, S, R, B coordinates. Transport, replay, provenance, and local naming through H, C, P, N preserve that route and provide no coordinate for the excluded principles. □

Theorem 6041.3 (Real completion universal-property carrier rows)**.** *Every accepted* RealCompletionUniversalProper *carrier displays exactly the rows*

$$D, S, R, B, E, U, H, C, P, N$$

of Definition 6041.1 before the NameCert obligations of Definition 6041.2 can be read. The dense source, stream windows, regular readback, Bishop completion handoff, completion-interface extension, and universal-property rows are therefore public only as one finite route.

Proof. Project the carrier definition. It names $D, S, R, B, E, U, H, C, P, N$ as the full packet, with no coordinate for a quotient stream, selected completed point, host equality, or ambient completeness theorem.

Apply the NameCert obligation theorem. Its seven obligation rows are indexed by the same carrier coordinates, so each public universal-property read must expose the dense source, finite windows, readback, completion handoff, extension row, and uniqueness route in that order.

The structural rows preserve the displayed packet. They replay and name the route already present in the carrier rather than adding a second completion principle. □

Theorem 6041.4 (Real completion universal-property extension route)**.** *Let* $G = (D, S, R, B, E, U, H, C, P, N)$ *be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. Any completion-facing extension read from* G *must consume* $\mathsf{DyadicRatCore}^{\uparrow}$ *through* D*,* $\mathsf{StreamName}^{\uparrow}$ *through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *through* R*, and the Bishop completion row through* B *before the completion-interface row* E *and universal-property row* U *are public.*

Proof. Start with the carrier rows of Definition 6041.3. The dense dyadic row D, finite stream row S, regular readback row R, and Bishop completion row B are the entrance rows of the packet.

Use the obligation theorem Definition 6041.2. Dense-source exposure, stream-window exactness, and regular-readback handoff force an extension read to consume those sibling rows before the completion-interface row is inspected.

The extension rows E and U then record the public extension and uniqueness route over the same D, S, R, B coordinates. The structural rows H, C, P, N keep the route local to the carrier. □

Theorem 6041.5 (Real completion universal-property nonescape)**.** *Every public* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *universal-property read factors through the carrier rows*

$$D, S, R, B, E, U, H, C, P, N$$

and through the extension route of Definition 6041.4. It cannot export quotient streams, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached universal-property principle outside the displayed dense-source and completion-interface rows.*

Proof. Combine the carrier-row theorem with the extension route theorem. Together they force the public read through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, the Bishop completion row, the completion-interface row, and the universal-property row.

Apply the non-escape obligation from Definition 6041.2. The only remaining rows are transport, replay, provenance, and local naming, and each preserves the displayed route.

Since the carrier stores no quotient stream, selected limit, host equality, choice schedule, ambient completeness theorem, or detached adjunction datum, none can be exported by a public universal-property consumer. □

Theorem 6041.6 (Real completion universal-property obligation upgrade)**.** *For every accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier* $G = (D, S, R, B, E, U, H, C, P, N)$*, the seven obligation rows of Definition 6041.2 form the obligation surface for* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *over* $D, S, R, B, E, U, H, C, P, N$*. They are not merely seed prose: carrier admission, dense-source exposure, stream-window exactness, regular-readback handoff, extension existence, extension uniqueness, and non-escape together determine the public universal-property route for completion-facing consumers without quotient streams, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the existing obligation list of Definition 6041.2. Its seven rows are already indexed by the displayed carrier coordinates $D, S, R, B, E, U, H, C, P, N$, so the public surface is read from the finite packet rather than from an external completion theorem.

Use factorization through Definition 6041.7. The dense-source, stream-window, regular-readback, completion-handoff, and extension rows force any universal factorization witness to pass through the displayed dyadic, stream, readback, completion, interface, and universal-property coordinates.

Next apply Real-seal uniqueness and extension-uniqueness factorization, Definition 6041.8 and Definition 6041.9. These rows bind uniqueness to the same $D, S, R, B, E, U, H, C, P, N$ route.

The non-escape row is then carried by H, C, P, N. Since the carrier contains no quotient-stream, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, the seven rows constitute the obligation surface and do not export any stronger completion principle. □

Theorem 6041.7 (Real completion universal-property factorization obligation)**.** *Any accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier must factor a completion-facing map through the displayed* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal row, Cauchy-modulus row, and provenance rows before it exports a universal factorization witness. The carrier has no coordinate for quotient representatives, selected limits, host real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6041.1. The completion-facing route enters through the dyadic source row D, finite stream window S, regular readback row R, Bishop regular Cauchy completion handoff B, completion-interface row E, and universal-property row U before transport, replay, provenance, and local naming are read.

Apply the local NameCert obligations of Definition 6041.2. Dense-source exposure, stream-window exactness, regular-readback handoff, extension existence, extension uniqueness, and non-escape force a universal factorization witness to pass through the displayed finite-window, readback, Cauchy-modulus, Real-seal, and provenance rows.

The sibling routes chapter 126, chapter 301, and chapter 38 supply those rows as finite BEDC surfaces. Since the carrier definition stores no quotient representative, selected limit, host equality, countable-choice schedule, or ambient completeness coordinate, this is a concrete L10 carrier obligation rather than a parameter-transport theorem. □

Theorem 6041.8 (Real completion universal-property Real-seal uniqueness)**.** *Let* $G = (D, S, R, B, E, U, H, C, P, N$ *be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. If two accepted extension reads display the same dyadic row* D*, stream row* S*, regular-readback row* R*, Bishop regular Cauchy completion row* B*, and terminal* $\mathsf{Real}^{\uparrow}$ *seal row named by the completion interface, then they have the same completion-facing route through* E, U, H, C, P, N*.*

Proof. Project the carrier and the two accepted reads. The hypotheses identify the dyadic source, finite windows, regular readback, Bishop regular Cauchy handoff, and terminal Real seal rows.

Apply the extension uniqueness row of Definition 6041.2. Since uniqueness is recorded over exactly the displayed D, S, R, B, E coordinates, the two reads cannot diverge into separate completion-interface routes.

The remaining structural rows H, C, P, N transport, replay, source, and name the same accepted route. Because the carrier has no quotient-stream, selected-limit, host-equality, countable-choice, ambient-completeness, or bridge coordinate, the common displayed seal determines the completion-facing route. □

Theorem 6041.9 (Real completion universal-property extension-uniqueness factorization)**.** *Let $G = (D, S, R, B, E, U, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. Any two extension reads over the same dyadic row D, stream row S, regular-readback row R, Bishop regular Cauchy completion row B, and completion-interface extension row E factor through the same U, H, C, P, N route before a completion-facing consumer can read uniqueness. The factorization exports no quotient stream, selected limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the accepted carrier and the two extension reads. The common coordinates D, S, R, B, E are the only entrance rows for extension data in Definition 6041.1.

Apply the NameCert obligation row of Definition 6041.2. Extension uniqueness is stored over the displayed extension route, so the two reads must pass through the same universal-property row U and the same structural rows H, C, P, N before a consumer can use the uniqueness claim.

The non-escape inventory of the carrier then excludes alternate routes. Since quotient streams, selected limits, host equality, countable choice, and ambient $\mathsf{Real}^{\uparrow}$ completeness are absent from G, the uniqueness read cannot be reinterpreted as any of those principles. □

Theorem 6041.10 (Real completion universal-property Bishop Kleisli comparison)**.** *Let $G = (D, S, R, B, E, U, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. Any Bishop Kleisli triangle consumer may read the* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *universal-property row only through the displayed route*

$$D, S, R, B, E, U, H, C, P, N$$

and then through the Bishop triangle route

$$M, K, F, U, R, E, J, H, C, P, N.$$

The comparison exports no quotient stream, selected completed point, host equality, countable-choice schedule, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or detached adjunction principle.*

Proof. Project the accepted carrier G. By Definition 6041.2, its universal-property row is visible only after the dyadic source, stream windows, regular readback, Bishop regular Cauchy completion handoff, and completion-interface extension rows have been displayed.

Use Definition 6041.9 for the uniqueness handoff. Any completion-facing read of uniqueness must pass through U, H, C, P, N over the same D, S, R, B, E coordinates before it reaches a downstream consumer.

Finally apply the Bishop triangle obligation surface of Definition 6061.2. That consumer route reads the monad, Kleisli, functor, universal-property, regular readback, Real seal, triangle ledger, transport, replay, provenance, and local naming rows as $M, K, F, U, R, E, J, H, C, P, N$. Since both packets exclude quotient streams, selected completed points, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, and detached adjunction data, the comparison exports none of those objects. □

Theorem 6041.11 (Real completion universal-property diagonal-family extension)**.** *Let* $G = (D, S, R, B, E, U, H, C, P, N)$ *be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. Any family-extension export from* G *must be displayed through a finite-window* $\mathsf{StreamName}^{\uparrow}$ *family, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, a* $\mathsf{NullSequence}^{\uparrow}$ *zero-tail comparison, the terminal* $\mathsf{Real}^{\uparrow}$ *seal, and the same universal-property route*

$$D, S, R, B, E, U, H, C, P, N.$$

The export can therefore extend only the displayed diagonal family served by those rows. It exports no quotient streams, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or hidden equality principle.*

Proof. Project the accepted carrier. The only universal-property entrance rows are the dyadic source D, finite stream windows S, regular readback R, Bishop regular Cauchy completion handoff B, completion-interface row E, universal-property row U, and the structural rows H, C, P, N.

Apply Definition 6041.9. Any extension read over the displayed dyadic, stream, regular-readback, Bishop completion, and interface rows factors through the same U, H, C, P, N route before uniqueness can be consumed.

Next use Definition 6041.10. A Bishop Kleisli triangle consumer reads the universal-property row only after the $D, S, R, B, E, U, H, C, P, N$ route is visible, so a family extension cannot bypass the finite-window and regular-readback stages.

The diagonal family adds a $\mathsf{NullSequence}^{\uparrow}$ zero-tail comparison as a displayed consumer row between the regular readback and the Real seal. Since the carrier has no coordinate for quotient streams, selected limits, host equality, choice, or ambient completeness, the resulting extension is exactly the displayed finite-window family route and no stronger completion principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionUniversalProperty}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ packet is generated from DyadicRatCore dense-source rows, StreamName finite windows, RegSeqRat regular readback, BishopRegularCauchyCompletion handoff, CauchyRealCompletionInterface extension rows, a universal-property row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionUniversalProperty}^{\uparrow}}$, seed)

Scope closed: Definition 6041.1, Definition 6041.2, and Definition 6041.8 seed the finite universal-property packet over dyadic, stream, regular-readback, completion, interface, extension, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionUniversalProperty}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-source exposure, stream-window exactness, regular-readback handoff, extension existence, extension uniqueness, and non-escape for BEDC.Derived.RealCompletionUniversalProperty

Theorem 6041.12 (Real completion universal-property extension stability)**.** *Let* $G = (D, S, R, B, E, U, H, C, P, N)$ *be an accepted* $\mathsf{RealCompletionUniversalProperty}^{\uparrow}$ *carrier. If two accepted extension reads share the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row* D*,* $\mathsf{StreamName}^{\uparrow}$ *window* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, Bishop regular completion handoff* B*, completion-interface row* E*, and universal-property row* U*, then every downstream completion-facing consumer must read the same* H, C, P, N *route before uniqueness is public. The stability claim exports no quotient stream, selected limit, host equality, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Project the carrier of Definition 6041.1. The extension route enters through D, S, R, B, E, U and has no other entrance coordinate for a completion-facing read.

Apply the NameCert obligations of Definition 6041.2. The extension existence and uniqueness clauses are recorded only over the displayed dyadic source, stream window, regular readback, Bishop completion handoff, completion-interface, and universal-property rows.

Use Real-seal uniqueness and extension-uniqueness factorization, Definition 6041.8 and Definition 6041.9. They force two reads sharing the displayed extension route to pass through the same structural route before a downstream consumer may read uniqueness.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the common route and provide no coordinate for quotient streams, selected limits, host equality, countable choice, or ambient Real completeness. □

Theorem 6041.13 (Real completion universal-property sibling route)**.** *Every accepted* RealCompletionUniversalPro *universal-property read must pass through* DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BishopRegularCauchyComple CauchyRealCompletionInterface$^{\uparrow}$, *and* Real$^{\uparrow}$ *before a Bishop completion comparison or* CauchyCompletionFunctor$^{\uparrow}$ *consumer may read it. In carrier notation the route is the displayed chain*

$$D, S, R, B, E, U, H, C, P, N$$

of Definition 6041.1, with D supplied by chapter 291, S by chapter 126, R by chapter 301, B by chapter 1639, E by chapter 1844, and the terminal seal by chapter 38. The route exports no quotient stream, selected limit, host equality, countable-choice schedule, ambient Real$^{\uparrow}$ *completeness principle, or detached completion functor.*

Proof. Start with the carrier inventory and NameCert obligations, Definition 6041.1 and Definition 6041.2. They make the dyadic source, stream window, regular readback, Bishop completion handoff, completion-interface row, universal-property row, transport, continuation, provenance, and local naming rows the whole admission surface.

Apply the factorization and uniqueness rows of Definition 6041.7 and Definition 6041.9. These force every universal-property read to expose D, S, R, B, E, U before any completion-facing consumer can use extension existence or uniqueness.

Finally compare with the downstream siblings. The Bishop completion comparison square reads the same Bishop regular Cauchy completion and Real seal rows, while the CauchyCompletionFunctor$^{\uparrow}$ consumer can only receive the public completion route after the displayed interface row has been named. Since H, C, P, N preserve only that route, no consumer obtains an extra quotient, chosen limit, host equality, choice schedule, ambient completeness theorem, or detached functor coordinate. □

Chapter 6042

A Concrete Naming Certificate for $\mathsf{RealModulusArithmetic}^{\uparrow}$

Following Bishop and Bridges, explicit real arithmetic with displayed moduli is a finite consumer route rather than an operation on host equality classes. The $\mathsf{RealModulusArithmetic}^{\uparrow}$ packet sits over the Cauchy-modulus arithmetic surface of chapter 1485, the shared meet surface of chapter 462, the regular addition and multiplication consumers of chapter 417 and chapter 461, and the terminal real-name seal of chapter 38. It gives later Real algebra packets one finite route for reading arithmetic moduli without importing quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness.

Definition 6042.1 (Real-modulus arithmetic carrier)**.** *A* $\mathsf{RealModulusArithmetic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X_0, X_1, \mu_0, \mu_1, A, U, D, W, R, \Sigma, \Pi, E, H, C, P, N)$$

with the following displayed rows:

1. *X_0 and X_1 are the two* $\mathsf{RealCauchyModulus}^{\uparrow}$ *source rows, each exposing a real-name seal together with the modulus schedule used to read it.*
2. *μ_0 and μ_1 are the two displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *schedules attached to those real-name rows.*
3. *A is the* $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ *row combining the two source schedules into the arithmetic-modulus surface.*
4. *U is the shared* $\mathsf{CauchyModulusMeet}^{\uparrow}$ *row used by both source schedules and by both arithmetic consumers.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for source precision requests, shared-meet requests, sum tolerances, and product tolerances.*
6. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window ledger naming the inspected source observations and the arithmetic consumer windows.*
7. *R stores the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks obtained from the source windows, dyadic tolerances, displayed schedules, and shared meet row.*
8. *Σ and Π are the* $\mathsf{RegularCauchySum}^{\uparrow}$ *and* $\mathsf{RegularCauchyProduct}^{\uparrow}$ *consumer rows reached through A, U, D, W, R.*
9. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the arithmetic readback, reached only after the sum or product consumer row has been displayed.*

10. *H records componentwise* hsame *transport for the real-modulus source rows, Cauchy-modulus schedules, arithmetic row, shared meet, dyadic ledger, windows, readbacks, sum and product consumers, Real seal, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from a real arithmetic precision request through the two source schedules, the shared meet row, D, W, R, and then through Σ or Π to the terminal seal E.*
12. *P is the* Pkg *provenance over* $\mathsf{RealModulusArithmetic}^{\uparrow}$, $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealModulusArithmetic}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $X_0, X_1, \mu_0, \mu_1, A, U, D, W, R, \Sigma, \Pi, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or an ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 ✓ `BEDC.Derived.RealModulusArithmeticUp.RealModulusArithmeticUpTasteGate single carrier alignment`

Theorem 6042.2 (Real-modulus arithmetic NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusArithmetic}^{\uparrow}$ *carrier*

$$M = (X_0, X_1, \mu_0, \mu_1, A, U, D, W, R, \Sigma, \Pi, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealModulusArithmetic}^{\uparrow}}$ *surface consists exactly of carrier admission, the two real-modulus source rows, the two Cauchy-modulus schedules, arithmetic-modulus routing, shared-meet routing, dyadic tolerance exactness, finite stream windows, regular rational readback, sum and product consumer rows, Real seal nonescape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6042.1. Its displayed coordinates list the two real-modulus sources, their Cauchy-modulus schedules, the arithmetic and meet rows, the tolerance ledger, windows, readbacks, arithmetic consumers, seal, and structural rows.

The source and arithmetic rows are read before any Real-facing consumer can inspect a result. The rows $X_0, X_1, \mu_0, \mu_1, A, U$ place the two source schedules on the common modulus-arithmetic surface supplied by the Cauchy layer.

The ledger, window, and readback rows D, W, R then provide the finite observations consumed by Σ and Π. The seal E is downstream of those consumers, and H, C, P, N only transport, replay, source, and name the same finite route.

Since the carrier has no coordinate for quotient streams, selected limits, host real equality, trichotomy, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6042.3 (Real-modulus arithmetic consumer boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Real-algebra consumer of an accepted* $\mathsf{RealModulusArithmetic}^{\uparrow}$ *packet factors through the displayed route*

$$X_0, X_1, \mu_0, \mu_1, A, U, D, W, R, \Sigma, \Pi, E, H, C, P, N.$$

The consumer may read the regular sum row Σ or product row Π and then the terminal seal E, but it receives no quotient stream equality, selected limit, host real equality, trichotomy witness, countable-choice selector, ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate, Lean verification, or bridge row from the packet.*

Proof. Begin with a consumer request for real arithmetic. The carrier exposes the source real names only through X_0, X_1 and their displayed schedules μ_0, μ_1.

The arithmetic-modulus row A and shared-meet row U are then read before the dyadic ledger, finite windows, and regular rational readbacks can be consumed. Thus any sum or product read reaches Σ or Π only through A, U, D, W, R.

The terminal real seal E is downstream of Σ and Π. The rows H, C, P, N preserve the same displayed route under transport, replay, provenance, and local naming, so they do not add another consumer-facing object.

No unlisted coordinate belongs to the packet. Hence quotient streams, selected limits, host real equality, trichotomy, countable choice, ambient completeness, Lean verification, and bridge checking remain outside the consumer boundary. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusArithmetic}^{\uparrow}$ arithmetic read exposes two real-modulus source rows, two Cauchy-modulus schedules, a $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ row, a shared $\mathsf{CauchyModulusMeet}^{\uparrow}$ row, dyadic tolerance data, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, a $\mathsf{RegularCauchySum}^{\uparrow}$ or $\mathsf{RegularCauchyProduct}^{\uparrow}$ consumer row, and a terminal $\mathsf{Real}^{\uparrow}$ seal before any Real-algebra consumer reads the packet.

Independence witness: cauchymodulusarithmetic, cauchymodulusmeet, regularcauchysum, regularcauchyproduct, real: no listed sibling supplies the whole real-modulus arithmetic packet with two real-modulus source rows, two displayed schedules, arithmetic routing, shared meet, dyadic tolerance ledger, finite windows, regular rational readback, sum and product consumers, terminal Real seal, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusArithmetic}^{\uparrow}}$.

Constructive story: A $\mathsf{RealModulusArithmetic}^{\uparrow}$ packet is generated from two real-modulus source rows, two $\mathsf{CauchyModulus}^{\uparrow}$ schedules, one $\mathsf{CauchyModulusArithmetic}^{\uparrow}$ row, one $\mathsf{CauchyModulusMeet}^{\uparrow}$ shared-meet row, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{RegularCauchySum}^{\uparrow}$ and $\mathsf{RegularCauchyProduct}^{\uparrow}$ consumer rows, a $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transports, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealModulusArithmetic}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6042.1, Definition 6042.2, and Definition 6042.3 seed the finite Real-facing arithmetic-modulus boundary over $\mathsf{CauchyModulusArithmetic}^{\uparrow}$, $\mathsf{CauchyModulusMeet}^{\uparrow}$, $\mathsf{RegularCauchySum}^{\uparrow}$, $\mathsf{RegularCauchyProduct}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealModulusArithmetic}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected limits, host real equality, trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: A formal route can encode the displayed carrier and prove the finite NameCert obligation surface and Real consumer boundary for BEDC.Derived.RealModulusArithmeticUp.

Chapter 6043

A Concrete Naming Certificate for FiniteDimensionalCompactness$^{\uparrow}$

Following Bishop and Bridges and standard finite-dimensional normed-space compactness, the FiniteDimensionalCompactness$^{\uparrow}$ packet records the finite-net route by which a displayed bounded closed coordinate window is consumed as compact data. It sits after the vector-space surface of chapter 47, the metric surface of chapter 57, the real-name seal of chapter 38, the norm-comparison route of chapter 5566, the interval compactness packet of chapter 3969, and the finite-net Heine–Borel route of chapter 3471. The packet is a constructive finite-net NameCert surface for finite-dimensional bounded windows; it is not a general Heine–Borel theorem, sequential compactness principle, host basis selector, quotient vector equality, or choice-based compactness object.

Definition 6043.1 (Finite-dimensional compactness carrier)**.** *A* FiniteDimensionalCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (V, M, R, E, N, I, F, W, S, H, C, P, Q)$$

with the following displayed rows:

1. *V is the* VecSpace$^{\uparrow}$ *row exposing vector addition, zero, inverse, scalar action, and vector classification only through displayed finite carrier data.*
2. *M is the* Met$^{\uparrow}$ *row induced by a displayed norm or coordinate distance on the same finite vector source.*
3. *R is the* Real$^{\uparrow}$ *seal row used for coordinate bounds, metric radii, finite sums, and comparison constants.*
4. *E is the* FiniteDimensionalNormEquivalence$^{\uparrow}$ *row that keeps all norm and coordinate comparisons on one finite coordinate window.*
5. *N is the* ClosedBoundedIntervalCompact$^{\uparrow}$ *interval compactness row for each coordinate bound.*
6. *I is the product interval-window row, listing the finitely many coordinate intervals and their endpoint seals.*
7. *F is the finite-net row read through* HeineBorelFiniteNet$^{\uparrow}$*, recording displayed centers, mesh radius, coverage witnesses, and radius comparisons for the coordinate window.*
8. *W is the bounded-window row tying every accepted vector read to the listed coordinate bounds before any compactness-facing consumer sees the metric row.*

9. *S is the exactness row saying that a finite-net membership read reconstructs a vector-window read through V, M, R, E, N, I, F, W.*

10. *H records componentwise* hsame *transport for the vector, metric, real, norm-equivalence, interval-compactness, finite-net, bounded-window, exactness, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a finite-dimensional compactness request through $V, M, R, E, N, I, F, W, S, I$*

12. *P is the* Pkg *provenance over* $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, FiniteDimensionalNormEq ClosedBoundedIntervalCompact$^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *Q is the local* $\mathsf{NameCert}_{\mathsf{FiniteDimensionalCompactness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed tuple $V, M, R, E, N, I, F, W, S, H, C, P, Q$. It has no coordinate for arbitrary open covers, host bases, quotient vector equality, selected subsequences, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.FiniteDimensionalCompactnessUp.FiniteDimensionalCompactnessTasteGate`
`single carrier alignment`

Theorem 6043.2 (Finite-dimensional compactness NameCert obligations)**.** *For every accepted* $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$ *carrier*

$$K = (V, M, R, E, N, I, F, W, S, H, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDimensionalCompactness}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed vector-space, metric, real-seal, finite-dimensional norm-equivalence, coordinate interval-compactness, interval-window, finite-net, bounded-window, exactness, transport, replay, provenance, and local naming rows. Every compactness-facing read must first expose the coordinate window I, the finite-net row F, and the bounded-window row W; every metric or norm comparison must factor through M, R, E; and every exported finite-net membership read must be repacked by S before it reaches the local* NameCert *row.*

Proof. Project the carrier of Definition 6043.1. The tuple displays exactly the rows $V, M, R, E, N, I, F, W, S, H, C, P$ so the classifier has no access to compactness data outside that packet.

The finite-dimensional source is read first. The vector row V and metric row M expose the finite carrier and distance surface, while the real row R supplies the coordinate bounds, radii, sums, and comparison constants.

The compactness route is finite-net based. The norm-equivalence row E keeps all coordinate and norm comparisons on the same finite window, the interval compactness row N supplies compactness for each coordinate interval, and I, F, W list the coordinate intervals, finite centers, coverage witnesses, and bounded-window admission data.

The exactness row S is the only route from finite-net membership back to vector-window membership. The structural rows H, C, P, Q transport, replay, source, and name that same route. Since the carrier has no row for arbitrary open covers, host basis selection, quotient vector equality, selected subsequences, countable choice, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6043.3 (Finite-dimensional compactness finite-net readback)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *or* $\mathsf{VecSpace}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$ *packet that requests compactness data receives only the finite route*

$$\mathsf{VecSpace}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FiniteDimensionalNormEquivalence}^{\uparrow} \longmapsto \mathsf{ClosedBoundedIntervalCompact}^{\uparrow} \longmapsto \mathsf{HeineBorel}$$

through $V, M, R, E, N, I, F, W, S, H, C, P, Q$. The readback exports finite coordinate bounds, displayed interval compactness rows, finite-net centers, mesh-radius comparisons, bounded-window admission, exactness, transport, replay, provenance, and local naming only; it exports no general Heine–Borel theorem, open-cover selector, sequential compactness branch, quotient vector equality, host basis choice, countable choice, or ambient completeness principle.

Proof. Start from the obligation surface of Definition 6043.2. A compactness-facing request opens the displayed vector, metric, and real rows before any coordinate or finite-net row is consumed.

Next read the finite-dimensional comparison data. The norm-equivalence row E ties the coordinate and metric comparisons to one finite window, and the interval compactness row N supplies compactness only for the listed coordinate intervals.

The finite-net handoff is I, F, W, S. The interval-window row lists the coordinate bounds, the finite-net row lists centers and coverage witnesses, the bounded-window row keeps vector reads inside those bounds, and the exactness row repacks finite-net membership as vector-window membership.

Finally H, C, P, Q preserve the same route for each consumer. Because no excluded object is present in the carrier, the consumer receives exactly the finite-net readback displayed in the statement. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$ compactness read must display $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FiniteDimensionalNormEquivalence}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, bounded-window, exactness, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any compactness-facing output is visible.

Independence witness: vecspace, metric, real, finite-dimensional-norm-equivalence, closed-bounded-interval-compact, heine-borel-finite-net: $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite vector-source rows, metric and Real seal rows, norm-equivalence comparisons, coordinate interval compactness, product-window bounds, finite-net coverage, exactness, transport, replay, provenance, and local naming in one compactness-facing packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDimensionalCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteDimensionalCompactness}^{\uparrow}$ packet is generated from VecSpaceUp, MetricSpaceUp, RealUp, FiniteDimensionalNormEquivalenceUp, ClosedBoundedIntervalCompactUp, HeineBorelFiniteNetUp, bounded-window, exactness, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalCompactness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6043.1, Definition 6043.2, and Definition 6043.3 seed the finite-net compactness route for displayed finite-dimensional bounded coordinate windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDimensionalCompactness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general Heine–Borel compactness, open-cover compactness, sequential compactness, host basis choice, quotient vector equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, vector and metric exposure, finite-dimensional norm-equivalence routing, coordinate interval compactness, finite-net readback, bounded-window exactness, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDimensionalCompactnessUp.

Chapter 6044

A Concrete Naming Certificate for ComplexPhaseSurfaceFiber$^{\uparrow}$

The ComplexPhaseSurfaceFiber$^{\uparrow}$ packet realizes the complex phase surface of Definition 9463.1 as a finite source-fiber NameCert route. It sits beside the complex coordinate carrier of chapter 39, the disk-coordinate surface of chapter 377, the reversible transport rows of chapter 219, the finite flow surface of chapter 195, and the ledger-conservation packet of chapter 3487. The source ledger remains primary: the visible complex phase token is a readout of finite source, coordinate, fiber, GAP residue, provenance, hsame, Cont, Pkg, and NameCert rows, not a replacement source.

Definition 6044.1 (Complex phase surface fiber carrier)**.** *A* ComplexPhaseSurfaceFiber$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, Z, D, U, Y, L, G, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the source ledger row, recording the focused finite* Hist *rows, selector rows, and source classifier data that make the phase readout visible.*
2. *Z is the* Complex$^{\uparrow}$ *coordinate row, exposing the visible complex token as rational-pair NameCert data.*
3. *D is the* UnitDisk$^{\uparrow}$ *coordinate row when the readout is normalized to a disk-facing phase surface.*
4. *U is the* UnitaryGroup$^{\uparrow}$ *reversible transport row, read only as displayed classifier-preserving transport.*
5. *Y is the* DynSystem$^{\uparrow}$ *finite flow row, recording local record evolution without a hidden global dynamics.*
6. *L is the* LiouvilleLedgerConservation$^{\uparrow}$ *row, requiring reversible transport to preserve source, classifier, and provenance distinctions before a conserved phase read is exported.*
7. *G is the* GAP *residue row, listing unresolved exits, omitted source distinctions, and boundary provenance that the visible complex token hides.*
8. *R is the fiber readback row. It records the exact route from S, G to Z, D, U, Y, L and back to the source ledger when the visible phase token is inspected.*
9. *H records componentwise* hsame *transport for source ledger, complex coordinate, disk coordinate, reversible transport, finite flow, conservation,* GAP *residue, fiber readback, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from a phase-read request through $S, Z, D, U, Y, L, G, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, Z, D, U, Y, L, G, R, H, C, P, N$. It has no coordinate for treating a complex phase token as source equality, no hidden Hilbert-space state, no global dynamical solution, no erased GAP *residue, no quotient phase equality, no Lean verification, and no bridge checking.*

Theorem 6044.2 (Complex phase surface fiber NameCert obligations)**.** *For every accepted* ComplexPhaseSurfaceFi *carrier*

$$F = (S, Z, D, U, Y, L, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}}$ *surface consists exactly of source-ledger admission,* $\mathsf{Complex}^{\uparrow}$ *coordinate admission, optional* $\mathsf{UnitDisk}^{\uparrow}$ *coordinate admission, reversible* $\mathsf{UnitaryGroup}^{\uparrow}$ *transport admission, finite* $\mathsf{DynSystem}^{\uparrow}$ *flow admission,* $\mathsf{LiouvilleLedgerConservation}^{\uparrow}$ *preservation admission,* GAP *residue exposure, fiber readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The visible phase read can be exported only after the source ledger S, residue row G, and fiber readback row R are displayed.*

Proof. Project the carrier of Definition 6044.1. The tuple lists exactly the source, coordinate, transport, finite-flow, conservation, residue, readback, and structural rows, so the local classifier has no access to any detached phase object.

The source route is upstream. The source ledger S and GAP residue row G must be visible before the complex coordinate Z is interpreted as a phase readout.

The representation route is finite. The rows D, U, Y, L record disk normalization, reversible transport, finite flow, and ledger conservation only as displayed NameCert rows tied to S and G.

The fiber row R records the return route from visible phase token to source ledger and residue. The structural rows H, C, P, N preserve and name the same packet. Since the carrier has no row for source equality by phase equality, hidden Hilbert states, global dynamics, erased residue, or quotient phase equality, the listed obligations exhaust the local NameCert surface. □

Theorem 6044.3 (Complex phase surface fiber readback discipline)**.** *Every accepted* $\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}$ *phase read factors through the finite route*

$$S, G, R \longmapsto \mathsf{Complex}^{\uparrow}, \mathsf{UnitDisk}^{\uparrow}, \mathsf{UnitaryGroup}^{\uparrow}, \mathsf{DynSystem}^{\uparrow}, \mathsf{LiouvilleLedgerConservation}^{\uparrow} \longmapsto H, C, P, N.$$

Consequently a downstream consumer receives a visible complex coordinate, optional disk coordinate, reversible transport, finite flow, conservation ledger, source-fiber readback, GAP *residue, transport, replay, provenance, and local naming, and receives no source equality by phase equality, hidden global dynamics, ambient Hilbert state, erased residue, or quotient phase token.*

Proof. Use the obligation surface of Definition 6044.2. A phase consumer first reads the source ledger, the GAP residue, and the fiber readback row.

The visible phase route then opens the coordinate and dynamics rows. The complex coordinate, disk coordinate, reversible transport, finite flow, and ledger-conservation rows are all downstream of the source-fiber data rather than independent replacements for it.

Finally H, C, P, N preserve that route for downstream consumers. Because the carrier has no coordinate for phase equality as source equality, hidden global dynamics, ambient Hilbert state, erased residue, or quotient phase tokens, none of those readings can be exported. □

Theorem 6044.4 (Complex phase surface TasteGate boundary)**.** *Any checked TasteGate promotion for* ComplexPhaseSurfaceFiber$^{\uparrow}$ *must provide* Nontrivial ComplexPhaseSurfaceFiber$^{\uparrow}$ *and* FieldFaithful ComplexPh *instances preserving the distinct rows*

$$S, Z, D, U, Y, L, G, R, H, C, P, N.$$

Before those instances are present, the packet exports only the finite source-ledger, complex-coordinate, disk-coordinate, reversible-transport, finite-flow, ledger-conservation, GAP *residue, fiber-readback, transport, replay, provenance, and local naming rows.*

Proof. The carrier of Definition 6044.1 separates source, coordinate, transport, flow, conservation, residue, and fiber-readback rows. A nontriviality instance must distinguish an accepted packet from a packet missing one of those source-fiber rows.

Field faithfulness must preserve the row order rather than collapsing the visible complex coordinate into the source ledger or erasing the GAP residue. Thus the TasteGate boundary is exactly the requirement that future checked instances keep $S, Z, D, U, Y, L, G, R, H, C, P, N$ as distinct displayed fields. □

Falsifiable prediction: Every accepted ComplexPhaseSurfaceFiber$^{\uparrow}$ consumer must display source ledger, Complex$^{\uparrow}$ coordinate, fiber readback, GAP residue, provenance, hsame, Cont, Pkg, and local NameCert rows before treating a phase token as readable output.

Independence witness: complex, unitdisk, unitarygroup, dynsystem, liouville-ledger-conservation: ComplexPhaseSurfaceFiber$^{\uparrow}$ is not bijective to any listed sibling because it binds source ledger, visible complex coordinate, disk normalization, reversible transport, finite flow, ledger conservation, GAP residue, fiber readback, provenance, transport, replay, and local naming as one source-fiber packet.

Closure status for NameCert$_{\text{ComplexPhaseSurfaceFiber}^{\uparrow}}$**.**

Constructive story: A ComplexPhaseSurfaceFiber$^{\uparrow}$ packet is generated from source-ledger rows, a ComplexUp coordinate row, UnitDiskUp coordinate data, UnitaryGroupUp transport, DynSystemUp finite flow, LiouvilleLedgerConservationUp preservation, GAP residue, fiber readback, componentwise hsame, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ComplexPhaseSurfaceFiber}^{\uparrow}}$, seed)

Scope closed: Definition 6044.1, Definition 6044.2, and Definition 6044.3 seed the finite source-fiber phase readback route.

Formal verification: FormalStatus(NameCert$_{\text{ComplexPhaseSurfaceFiber}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close phase equality as source equality, hidden Hilbert-space state, global dynamical solutions, quotient phase tokens, erased GAP residue, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source-ledger admission, ComplexUp coordinate admission, fiber readback, GAP residue exposure, transport, finite-flow, conservation, Nontrivial and FieldFaithful TasteGate instances, replay, provenance, and local naming for BEDC.Derived.ComplexPhaseSurfaceFiberUp.

Chapter 6045

A Concrete Naming Certificate for $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$

Following Bishop and Bridges, located real Cauchy completeness is read constructively as a finite route from located rational interval evidence and regular rational names to a Real-name boundary, not as an ambient selection principle. The $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ packet records this L10 route beside the located-real surface of **??**, the Cauchy-modulus carrier of chapter 5887, the stream-name observation surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38.

Definition 6045.1 (Located real Cauchy completeness carrier)**.** *A* $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{LocatedReal}^{\uparrow}$ *admission row, exposing located interval evidence before any completion read is public.*
2. *M is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *modulus row that names the displayed Cauchy request and its finite tail budget.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose finite observations are read by the modulus.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the window and modulus rows.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row assembled from W and D.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after located admission, modulus exposure, finite windows, dyadic tolerance, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for located-real, modulus, stream, dyadic, regular-readback, Real-seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from located admission through $X, M, W, D, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, M, W, D, R, E, H, C, P, N$. It has no coordinate for quotient real equality, selected limits, countable choice, a host limit operator, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedRealCauchyCompletenessUp.LocatedRealCauchyCompletenessTasteGate`
`single carrier alignment`

Theorem 6045.2 (Located real Cauchy completeness NameCert obligations)**.** *For every accepted* $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ *carrier*

$$L = (X, M, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}}$ *surface consists exactly of located-real admission for X, modulus exposure for M, finite StreamName windows for W, dyadic tolerance visibility for D, RegSeqRat readback for R, terminal Real sealing for E, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

No located-real Cauchy completeness obligation is stored outside these rows.

Proof. Project the carrier of Definition 6045.1. The tuple $X, M, W, D, R, E, H, C, P, N$ is the whole public packet, so carrier admission and local naming are exhausted by the displayed rows.

The completion route begins with the located-real row X and the modulus row M. The finite StreamName window W is read against the DyadicRatCore tolerance ledger D, and the RegSeqRat readback R packages the same finite observations as a regular rational name.

Only after those rows have been displayed may the terminal Real seal E be consumed. The transport, replay, provenance, and local naming rows H, C, P, N preserve the same route and introduce no quotient equality, selected limit, countable-choice schedule, host limit operator, or ambient completeness coordinate. □

Theorem 6045.3 (Located real Cauchy completeness regular readback route)**.** *Every accepted* $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ *packet exposes its completion neighbour through the finite route*

$$\mathsf{LocatedReal}^{\uparrow} \longrightarrow \mathsf{CauchyModulusSpace}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}.$$

The $\mathsf{RegSeqRat}^{\uparrow}$ *row is therefore the regular readback boundary between located interval evidence and the terminal Real seal. The route does not export quotient real equality, selected limits, countable choice, a host limit operator, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the located-real row X first. It supplies the located interval evidence that anchors the Cauchy request without identifying host real values.

Next read the modulus row M and the finite StreamName window row W. The DyadicRatCore ledger D supplies the tolerance scale shared by both rows.

The RegSeqRat row R is then the only regular rational readback available to the packet. The terminal Real seal E is downstream of R, and H, C, P, N merely transport, replay, source, and name the same finite evidence.

Since the carrier has no quotient-real, selected-limit, countable-choice, host-limit, or ambient-completeness coordinate, every completion read remains inside the displayed regular readback route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedRealCauchyCompleteness}^{\uparrow}$ packet is generated from located-real admission, CauchyModulusSpace modulus exposure, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedRealCauchyCompleteness}^{\uparrow}}$, seed)

Scope closed: Definition 6045.1, Definition 6045.2, and Definition 6045.3 seed the finite located-real Cauchy completeness route from located admission through regular rational readback to a terminal Real seal.

Formal verification: FormalStatus(NameCert$_{\text{LocatedRealCauchyCompleteness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, host limit operators, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked located-real admission, modulus exposure, finite-window coverage, dyadic tolerance visibility, RegSeqRat readback, Real seal routing, transport stability, replay stability, and non-escape for BEDC.Derived.LocatedRealCauchyCompletenessUp.

Chapter 6046

A Concrete Naming Certificate for $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ packet records the finite subsequence-window extraction used by the constructive Real and completion route. It reads the bounded $\mathsf{Real}^{\uparrow}$ windows of chapter 38, the prefix schedules of chapter 126, the regular rational readbacks of chapter 301, the dyadic tolerance ledger of chapter 291, the finite extraction surface of chapter 1913, and the criterion route of chapter 1509. The packet names a local finite window, not an ambient completeness theorem, a selected host limit, or a countable-choice subsequence selector.

Definition 6046.1 (Localized Cauchy subsequence carrier)**.** *A* $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (W, S, R, D, B, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the bounded* $\mathsf{Real}^{\uparrow}$ *window row. It records the finite source interval, endpoint seals, and bounded comparison rows in which the subsequence extraction is inspected.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *prefix schedule. It displays the finite source prefixes, candidate subsequence indices, and monotone prefix inclusions used by the local extraction.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the selected prefix windows and their rational regularity data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic scales, rational comparisons, and exhaustion budget used to test local Cauchy behavior.*
5. *B is the* $\mathsf{BolzanoCauchy}^{\uparrow}$ *route row. It exposes the finite subsequence-window extraction surface without exporting a selected host limit.*
6. *E is the* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$ *endpoint row that can be read only after W, S, R, D, B have been displayed.*
7. *H records componentwise* hsame *transport for the bounded Real window, stream prefixes, regular rational readbacks, dyadic ledger, Bolzano-Cauchy route, criterion endpoint, continuation, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a local tolerance request through W, S, R, D, B, E and then through the structural rows.*

9. *P is the* Pkg *provenance over* $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BolzanoCauchy}^{\uparrow}$, $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocalizedCauchySubsequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, a host-selected limit, an infinite selector, open-cover compactness, or countable choice.*

Lean4 ✓ `BEDC.Derived.LocalizedCauchySubsequenceUp.TasteGate.LocalizedCauchySubsequenceTasteGate single carrier alignment`

Theorem 6046.2 (Localized Cauchy subsequence NameCert obligations)**.** *For every accepted* $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ *carrier*

$$Q = (W, S, R, D, B, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocalizedCauchySubsequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the bounded Real window, StreamName prefixes, RegSeqRat readback, DyadicRatCore tolerance ledger, Bolzano-Cauchy route, Cauchy-convergence endpoint, transport, replay, provenance, and local naming rows;*
2. *local-window exposure, requiring every subsequence read to display W before it can inspect S, R, D;*
3. *finite extraction routing, requiring every subsequence candidate to pass through S, R, D and the* $\mathsf{BolzanoCauchy}^{\uparrow}$ *row B before the criterion endpoint E is available;*
4. *structural stability, requiring H and C to preserve and replay only the displayed finite route;*
5. *ledger non-escape, requiring P and N to name no object outside the displayed packet.*

No localized subsequence obligation is stored outside these rows.

Proof. Project the carrier of Definition 6046.1. The visible coordinates are exactly the bounded Real window, prefix schedule, rational readback, dyadic ledger, Bolzano-Cauchy route, criterion endpoint, and the structural support rows.

The local-window obligation is forced by the row order. A subsequence read first sees the bounded $\mathsf{Real}^{\uparrow}$ window W, and only then can it inspect the finite prefixes S, readbacks R, and tolerances D.

The extraction route is the finite route through $\mathsf{BolzanoCauchy}^{\uparrow}$ and $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$. The rows B and E are downstream of S, R, D, so no endpoint can be exported before the local subsequence window has been displayed.

Finally, H, C, P, N transport, replay, source, and name that same packet. Since the carrier has no further coordinate, the five listed obligation rows exhaust the local NameCert surface. □

Theorem 6046.3 (Localized Cauchy subsequence window extraction)**.** *Every accepted* LocalizedCauchySubsequenc *carrier exposes a finite subsequence-window route*

$$W \longmapsto S \longmapsto R \longmapsto D \longmapsto B \longmapsto E \longmapsto H, C, P, N$$

before any Cauchy criterion endpoint can be consumed. The route may be read by Real and completion consumers only through the displayed $\mathsf{Real}^{\uparrow}$ *window,* $\mathsf{StreamName}^{\uparrow}$ *prefixes,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{BolzanoCauchy}^{\uparrow}$ *extraction row, and* $\mathsf{CauchyConvergenceCriterion}^{\uparrow}$

endpoint. It exports no ambient completeness principle, selected host limit, countable-choice subsequence selector, quotient stream equality, or open-cover compactness row.

Proof. Start with an accepted carrier. By Definition 6046.1, the only source of bounded sequence data is W, the only prefix data are S, and the only rational and dyadic tolerance data are R and D.

Apply Definition 6046.2. It requires any subsequence candidate to display W, S, R, D and then pass through the $\mathsf{BolzanoCauchy}^{\uparrow}$ route row B before the Cauchy-convergence endpoint E is public.

The structural rows do not add a second route. They preserve, replay, source, and name the same finite packet, so a Real or completion consumer can only receive the displayed route.

The excluded principles are absent from the carrier inventory. Hence the extraction remains a local finite subsequence-window route rather than an ambient completeness or choice construction. □

Falsifiable prediction: Every $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ read must display a bounded Real window, StreamName prefixes, RegSeqRat readback, DyadicRatCore tolerances, BolzanoCauchy routing, a CauchyConvergenceCriterion endpoint, transport, replay, provenance, and local naming before any Cauchy endpoint is consumed.

Independence witness: real, streamname, regseqrat, dyadicratcore, bolzano-cauchy, cauchy-convergence-criterion: $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the bounded local window, finite prefix extraction, tolerance ledger, criterion endpoint, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalizedCauchySubsequence}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalizedCauchySubsequence}^{\uparrow}$ packet is generated from a bounded RealUp window, StreamName prefix schedule, RegSeqRat readback, DyadicRatCore tolerance ledger, BolzanoCauchy route, CauchyConvergenceCriterion endpoint, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalizedCauchySubsequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6046.1, Definition 6046.2, and Definition 6046.3 seed the finite localized subsequence-window route for Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalizedCauchySubsequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient RealUp completeness, quotient stream equality, selected host limits, countable-choice subsequence selection, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-window exposure, finite extraction routing, structural stability, and ledger non-escape for BEDC.Derived.LocalizedCauchySubsequenceUp.

Chapter 6047

A Concrete Naming Certificate for $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$

Following Lebesgue and Stieltjes, the $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$ packet records the finite BEDC route by which a bounded-variation distribution function supplies interval masses to measure consumers. It sits after the partition-variation source of chapter 398, the finite interval-jump boundary of chapter 3528, the compact regular measure surface of chapter 2263, the endpoint carrier of chapter 1591, and the terminal real-name seal of chapter 38. The packet names only displayed half-open interval masses, finite additivity checks on the inspected interval algebra, compact-interval restriction rows, regular readback rows, transport, replay, provenance, and local naming; it is not a host measure space, a sigma-algebra completion, a selected Caratheodory extension, a quotient of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6047.1 (Lebesgue–Stieltjes measure carrier)**.** *A* $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, S, I, A, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedVariation}^{\uparrow}$ *source row naming the distribution function only through displayed partition-variation data.*
2. *S is the* $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$ *row carrying endpoint-jump reads for the inspected cells.*
3. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *restriction row, naming the compact interval on which the finite interval algebra is read.*
4. *A is the finite half-open interval algebra generated by the displayed endpoints, together with refinement witnesses for the cells requested by the packet.*
5. *K is the finite additivity ledger assigning* $\mathsf{DyadicRatCore}^{\uparrow}$ *mass rows to the cells of A and recording the displayed disjoint-union checks.*
6. *R is the* $\mathsf{RadonMeasure}^{\uparrow}$ *regular readback row exposed to compact-interval measure consumers.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal of the accepted finite mass readback.*
8. *H records componentwise* hsame *transport for the bounded-variation source, boundary row, interval restriction, interval algebra, additivity ledger, regular readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from bounded-variation partition reads through Stieltjes boundary data, finite interval algebra, additivity ledger, Radon readback, and the* $\mathsf{Real}^{\uparrow}$ *seal.*
10. *P is the* Pkg *provenance over* $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueStieltjesMeasure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, I, A, K, R, E, H, C, P, N$. It has no coordinate for a host sigma-algebra, a completed outer measure, a selected extension theorem, quotient equality of distribution functions, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.LebesgueStieltjesMeasureUp.LebesgueStieltjesMeasureTasteGate single carrier alignment`

Theorem 6047.2 (Lebesgue–Stieltjes measure NameCert obligations)**.** *For every accepted* LebesgueStieltjesMeasur *carrier*

$$L = (B, S, I, A, K, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LebesgueStieltjesMeasure}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded-variation source B, Stieltjes boundary exposure through S, compact interval restriction through I, finite interval-algebra visibility through A, finite additivity exactness through K, Radon-facing regular readback through R, terminal real sealing through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6047.1. The first three rows are the only source rows: B supplies bounded-variation partition data, S supplies endpoint-jump reads, and I restricts the read to a displayed compact interval.

The interval algebra and additivity obligations are A and K. The algebra A lists only the half-open cells generated by displayed endpoints, and K records the finite disjoint-union checks on those cells. Thus additivity is a finite ledger obligation rather than an appeal to a host sigma-algebra.

The downstream readback is R, E. The $\mathsf{RadonMeasure}^{\uparrow}$ row R exposes the compact-regular consumer surface, and E seals the same readback as a $\mathsf{Real}^{\uparrow}$ row. Transport, replay, provenance, and local naming are exactly H, C, P, N, leaving no hidden extension theorem, quotient, selector, countable-choice schedule, or ambient completeness coordinate. □

Theorem 6047.3 (Lebesgue–Stieltjes finite interval route)**.** *Every accepted* $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$ *packet routes a compact-interval mass request through*

$\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{StieltjesMeasureBoundary}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{RadonMeasure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, h

Concretely, a consumer must read the source row B, boundary row S, interval restriction I, finite interval algebra A, additivity ledger K, and regular readback R before the terminal seal E is visible.

Proof. Start with the source and boundary rows. The bounded-variation chapter supplies the finite partition-variation source, while chapter 3528 supplies the endpoint-jump boundary that can be inspected on displayed cells.

Restrict next to the compact interval. The row I is read through chapter 1591; it fixes the finite endpoint surface from which A is generated. The ledger K then records mass assignments and finite additivity checks only for those displayed cells.

Finally pass to regular readback and sealing. The $\mathsf{RadonMeasure}^{\uparrow}$ surface of chapter 2263 consumes the compact-interval finite ledger through R, and the real-name chapter seals that read through E. Since every consumer-visible coordinate appears in this route, bypassing any of B, S, I, A, K, R is outside the accepted packet. □

Theorem 6047.4 (Lebesgue–Stieltjes Real seal non-escape)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* LebesgueStieltjesMeasure *carrier is available only after the bounded-variation source, Stieltjes boundary row, closed bounded interval restriction, finite interval algebra, finite additivity ledger, and Radon readback have been displayed. The seal exports the same finite compact-interval mass readback and exports no host sigma-algebra, completed outer measure, selected extension theorem, quotient equality of distribution functions, countable-choice schedule, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Read the seal coordinate in Definition 6047.1. The row E is downstream from B, S, I, A, K, R, and its transport and replay are mediated only by H and C.

Apply Definition 6047.2. It identifies the displayed source, boundary, interval restriction, interval algebra, finite additivity ledger, regular readback, seal, transport, replay, provenance, and local naming rows as the complete local NameCert surface.

Apply Definition 6047.3. The route forces every compact-interval mass consumer to pass through the same finite rows before the $\mathsf{Real}^{\uparrow}$ seal is visible. No excluded object is a carrier coordinate, so the seal cannot export it. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueStieltjesMeasure}^{\uparrow}}$.

Constructive story: A $\mathsf{LebesgueStieltjesMeasure}^{\uparrow}$ packet is generated from a BoundedVariation source row, a StieltjesMeasureBoundary endpoint-jump row, a ClosedBoundedInterval restriction, a finite half-open interval algebra, a DyadicRatCore additivity ledger, a RadonMeasure regular readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueStieltjesMeasure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6047.1, Definition 6047.2, Definition 6047.3, and Definition 6047.4 seed the finite compact-interval Lebesgue–Stieltjes measure packet from bounded-variation boundary data to Radon-facing and Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueStieltjesMeasure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host sigma-additivity, sigma-algebra completion, completed outer measure, selected extension theorems, quotient equality of distribution functions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Stieltjes boundary exposure, compact interval restriction, finite interval-algebra visibility, additivity exactness, Radon readback, Real-seal non-escape, transport and replay stability, provenance non-escape, and a Lean carrier/obligation formalization for BEDC.Derived.LebesgueStieltjesMeasureUp.

Chapter 6048

A Concrete Naming Certificate for $\mathsf{CompactModulusRealizer}^{\uparrow}$

Following Heine–Cantor and Bishop constructive analysis, the $\mathsf{CompactModulusRealizer}^{\uparrow}$ packet realizes compact uniform continuity as a finite NameCert carrier. It reads compact finite-net data from chapter 60, pointwise modulus rows from chapter 130, metric distance witnesses from chapter 57, rational lower-bound folds from chapter 37, and the finite ProbeBundle recursion route underlying chapter 9237. The packet is a finite uniform-modulus extraction surface, not a host open-cover compactness proof, semantic continuity theorem, or choice-selected modulus.

Definition 6048.1 (Compact modulus realizer carrier)**.** *A* $\mathsf{CompactModulusRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, F, G, Q, B, U, H, C, P, N)$$

with the following displayed rows:

1. K *is the* $\mathsf{Compact}^{\uparrow}$ *finite-net witness row, carrying bundled centers, located coverage rows, and finite radius data for the compact source.*
2. F *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row for the displayed graph over the same source metric.*
3. G *is the* $\mathsf{Met}^{\uparrow}$ *distance-witness row, exposing source and target metric comparisons used by the triangle estimate.*
4. Q *is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row that records positive radius histories and the finite minimum ledger used to select a uniform precision.*
5. B *is the* ProbeBundle *recursion row over the compact finite net, including bundled center names, radius rows, and fold order.*
6. U *is the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger row exported only after* K, F, G, Q, B *have been displayed.*
7. H *records componentwise* hsame *transport for the compact, continuous-map, metric, rational-fold, ProbeBundle, uniform-modulus, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from a tolerance request through the compact finite net, pointwise moduli, metric-distance witnesses, rational lower-bound fold, ProbeBundle recursion, and uniform-modulus ledger.*
9. P *is the* Pkg *provenance over* $\mathsf{CompactModulusRealizer}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, ProbeBundle, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactModulusRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host open-cover compactness, semantic continuity, a choice-selected center outside the bundle, ambient completeness, quotient metric equality, or an external modulus selector.

Lean4 ✓ `BEDC.Derived.CompactModulusRealizerUp.CompactModulusRealizerTasteGate single carrier alignment`

Theorem 6048.2 (Compact modulus realizer NameCert obligations)**.** *For every accepted* $\mathsf{CompactModulusRealizer}^{\uparrow}$ *carrier*

$$M = (K, F, G, Q, B, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactModulusRealizer}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite-net witness, continuous-map pointwise modulus, metric distance witnesses, rational lower-bound fold, ProbeBundle recursion, uniform-modulus ledger, transport, replay, provenance, and local naming rows;*
2. *compact-source exposure, requiring every uniform-modulus request to display K and B before any finite lower-bound fold is read;*
3. *pointwise-modulus exposure, requiring the continuous-map row F and metric row G to supply the local distance estimates consumed by the fold;*
4. *rational-fold exactness, requiring Q to record the selected positive lower bound through finite* ProbeBundle *recursion;*
5. *uniform-modulus handoff, requiring U to be downstream of K, F, G, Q, B and to export only the displayed tolerance-to-precision ledger;*
6. *structural stability, requiring H, C, P, N to preserve, replay, source, and name the same finite packet.*

No compact-modulus obligation is stored outside these rows.

Proof. Project the carrier of Definition 6048.1. The listed coordinates are exactly the compact finite-net witness, pointwise-modulus row, metric row, rational fold, ProbeBundle recursion row, uniform-modulus ledger, and structural rows.

The compact-source and pointwise-modulus obligations are forced by the data flow. The finite net and bundle recursion supply the centers and radius rows, while the continuous-map and metric rows supply the local estimates that can be folded.

The rational-fold exactness obligation is the role of Q. It records the selected positive lower bound through finite recursion over B, so the exported precision is a displayed history rather than an external selector.

The handoff to U is therefore downstream of all analytic rows. The structural rows H, C, P, N add only transport, replay, provenance, and naming, so the six listed rows exhaust the local NameCert surface. □

Theorem 6048.3 (Compact modulus ProbeBundle fold)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactModulusRealizer}^{\uparrow}$ *carrier receives its precision row only through the finite route*

$$K \longmapsto B \longmapsto F \longmapsto G \longmapsto Q \longmapsto U \longmapsto H, C, P, N.$$

The fold is the compact finite-net and pointwise-modulus construction of Definition 9237.5 read as concrete NameCert data: bundled compact centers supply local pointwise moduli, the $\mathsf{Met}^{\uparrow}$ *rows supply source and target distance witnesses, the* $\mathsf{Rat}^{\uparrow}$ *fold selects a positive lower bound, and the* $\mathsf{UniformModulus}^{\uparrow}$ *ledger receives only that finite tolerance-to-precision route.*

Proof. Start with a uniform-modulus read of the accepted carrier. By carrier admission, the only public uniform-modulus row is U, and it is placed after the compact finite-net row K, bundle recursion row B, pointwise-modulus row F, metric row G, and rational lower-bound row Q.

Read the finite compact net first. The rows K and B expose the bundled centers and radius histories used by primitive recursion over the finite ProbeBundle.

Next read the local analytic data. The row F supplies pointwise moduli at bundled centers, while G supplies the metric-distance witnesses used by the source and target triangle estimates.

Finally, the row Q folds the positive radius histories into a displayed lower-bound ledger. The uniform-modulus row U receives exactly that ledger with the structural rows H, C, P, N, and the carrier has no coordinate for host compactness, semantic continuity, external choice, ambient completeness, or quotient metric equality. □

Falsifiable prediction: Every $\mathsf{CompactModulusRealizer}^{\uparrow}$ read must display CompactUp finite-net data, ContinuousMapUp pointwise-modulus rows, MetricSpaceUp distance witnesses, RatUp lower-bound folds, ProbeBundle recursion, UniformModulusUp handoff, transport, replay, provenance, and local naming.

Independence witness: compact, continuousmap, metricspace, rat, probebundle, uniformmodulus: $\mathsf{CompactModulusRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite nets, pointwise moduli, metric witnesses, rational lower-bound folds, ProbeBundle recursion, uniform-modulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactModulusRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactModulusRealizer}^{\uparrow}$ packet is generated from CompactUp finite-net witnesses, ContinuousMapUp pointwise-modulus rows, MetricSpaceUp distance witnesses, RatUp lower-bound folds, ProbeBundle recursion, a UniformModulusUp ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactModulusRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6048.1, Definition 6048.2, and Definition 6048.3 seed the finite compact-uniform-modulus realization surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactModulusRealizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host open-cover compactness, semantic continuity, external modulus selection, ambient completeness, quotient metric equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, pointwise-modulus exposure, rational-fold exactness, uniform-modulus handoff, and structural stability for BEDC.Derived.CompactModulusRealizerUp.

Chapter 6049

A Concrete Naming Certificate for $\mathsf{JacksonFiniteApproximation}^{\uparrow}$

Following Jackson and constructive approximation theory, the $\mathsf{JacksonFiniteApproximation}^{\uparrow}$ packet records a finite error-ledger route from compact-uniform data to a displayed polynomial approximant. It sits over the compact metric source of chapter 129, the continuous-map carrier of chapter 130, the polynomial code surface of chapter 50, the Chebyshev recurrence surface of chapter 3480, the finite Weierstrass approximation corridor of chapter 1634, and the terminal real-name seal of chapter 38. The packet names the inspected Jackson window only; it is not a host best-approximation theorem, a quotient function-space construction, a selected infinite sequence of polynomials, an open-cover compactness theorem, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6049.1 (Jackson finite approximation carrier)**.** *A* $\mathsf{JacksonFiniteApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, F, \Omega, D, T, PW, Q, E, H, C, \Pi, N)$$

with the following displayed rows:

1. *K is the compact metric source row read through* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *F is the continuous-map source row read through* $\mathsf{ContinuousMap}^{\uparrow}$*.*
3. *Ω is the finite modulus-of-continuity window used by the Jackson tolerance request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *error ledger naming the requested scale, the local mesh scale, and the residual bound.*
5. *T is the* $\mathsf{ChebyshevPolynomial}^{\uparrow}$ *recurrence row used to display the finite polynomial stencil.*
6. *PW is the* $\mathsf{Poly}^{\uparrow}$ *code row for the approximating polynomial and its inspected evaluation windows.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for source samples, polynomial samples, and dyadic residual checks.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the resulting finite error comparison.*
9. *H records componentwise* hsame *transport for compact source data, continuous-map data, modulus windows, dyadic errors, Chebyshev recurrence rows, polynomial codes, readbacks, Real seals, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a Jackson tolerance request through $K, F, \Omega, D, T, PW, Q, E, H, \Pi, N$.*

11. Π *is the* Pkg *provenance over* $\mathsf{JacksonFiniteApproximation}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{ChebyshevPolynomial}^{\uparrow}$, $\mathsf{WeierstrassApproximation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N *is the local* $\mathsf{NameCert}_{\mathsf{JacksonFiniteApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, \Omega, D, T, PW, Q, E, H, C, \Pi, N$. *It has no coordinate for a host supremum norm, a best approximant selector, quotient function equality, open-cover compactness, countable choice, or ambient Real completeness.*

Lean4 ✓ `BEDC.Derived.JacksonFiniteApproximationUp.JacksonFiniteApproximationTasteGate single carrier alignment`

Theorem 6049.2 (Jackson finite approximation NameCert obligations)**.** *For every accepted* JacksonFiniteApproxim *carrier*

$$J = (K, F, \Omega, D, T, PW, Q, E, H, C, \Pi, N),$$

the local $\mathsf{NameCert}_{\mathsf{JacksonFiniteApproximation}^{\uparrow}}$ *surface consists exactly of compact-source admission, continuous-map admission, finite modulus-window exposure, dyadic error-ledger visibility, Chebyshev recurrence display, polynomial code display, regular rational readback, Real seal nonescape, componentwise transport, displayed replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6049.1. Its displayed coordinates are precisely the compact source, continuous source, modulus window, dyadic error ledger, Chebyshev recurrence, polynomial code, readback, seal, and structural rows.

The approximation route starts with K and F, then reads the finite modulus window Ω before any polynomial window is consumed. The dyadic row D records the requested Jackson scale and the residual budget used by the later comparison.

The recurrence row T and polynomial row PW display the finite approximant, while Q reads the source and polynomial samples through regular rational data. The terminal seal E is downstream of those rows, and H, C, Π, N only transport, replay, source, and name the same finite route.

Since no other coordinate belongs to the packet, host supremum norms, best-approximation selectors, quotient function equality, open-cover compactness, countable choice, and ambient Real completeness remain outside the local NameCert surface. □

Theorem 6049.3 (Jackson finite polynomial window)**.** *Every polynomial-window consumer of an accepted* $\mathsf{JacksonFiniteApproximation}^{\uparrow}$ *packet reads the approximating polynomial through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\quad \mathsf{ContinuousMap}^{\uparrow},\quad \mathsf{ChebyshevPolynomial}^{\uparrow},\quad \mathsf{Poly}^{\uparrow},\quad \mathsf{WeierstrassApproximation}^{\uparrow},\quad \mathsf{Real}^{\uparrow}.$$

The consumer may inspect the rows $K, F, \Omega, D, T, PW, Q, E, H, C, \Pi, N$ *and the Jackson error ledger stored in* D*, but it receives no host best approximant, quotient function-space equality, selected infinite polynomial sequence, open-cover compactness theorem, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row.*

Proof. Begin with the compact source and continuous-map rows K and F. They provide the finite domain and function surface before any approximation data can be read.

Next inspect the modulus window Ω and dyadic ledger D. These rows bind the tolerance request to a finite scale and prevent the polynomial consumer from appealing to a host supremum norm.

The Chebyshev row T and polynomial row PW display the finite approximant. The readback row Q compares source samples and polynomial samples at the named dyadic scale, and the terminal Real seal E is available only after that comparison route has been displayed.

Transport, replay, provenance, and local naming preserve the same rows. Therefore the polynomial window is a finite Jackson error-ledger object rather than a host compactness or function-space theorem. □

Falsifiable prediction: Every accepted $\mathsf{JacksonFiniteApproximation}^{\uparrow}$ approximation read exposes a compact metric source, a continuous-map source, a finite modulus window, a dyadic error ledger, a Chebyshev recurrence row, a polynomial code row, regular rational readback, and a terminal $\mathsf{Real}^{\uparrow}$ seal before any polynomial-window consumer reads the packet.

Independence witness: compactmetric, continuousmap, polynomial, chebyshev-polynomial, weierstrass-approximation, real: no listed sibling supplies the whole Jackson finite approximation packet with compact source data, continuous-map data, finite modulus windows, dyadic error budgets, Chebyshev recurrence, polynomial code, regular rational readback, Real seal, transport, replay, provenance, and local NameCert naming.

Closure status for $\mathsf{NameCert}_{\mathsf{JacksonFiniteApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{JacksonFiniteApproximation}^{\uparrow}$ packet is generated from compact metric source data, continuous-map source data, a finite modulus window, a dyadic error ledger, a Chebyshev recurrence row, a polynomial code row, regular rational readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JacksonFiniteApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6049.1, Definition 6049.2, and Definition 6049.3 seed the finite Jackson error-ledger route from compact-uniform source data through Chebyshev and polynomial rows to a terminal RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JacksonFiniteApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host best approximation, quotient function equality, selected infinite polynomial sequences, open-cover compactness, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite modulus-window exposure, dyadic error-ledger exactness, Chebyshev and polynomial handoff, regular rational readback, Real seal routing, and nonescape for BEDC.Derived.JacksonFiniteApproximationUp.

The $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ hub routes dyadic tail-oscillation reads through chapter 291, chapter 126, chapter 301, and chapter 38 before the child packet is read.

Chapter 6050

A Concrete Naming Certificate for DyadicCauchyOscillationModulus$^{\uparrow}$

Following Bishop and Bridges, the DyadicCauchyOscillationModulus$^{\uparrow}$ packet records the finite modulus surface by which dyadic Cauchy oscillation reads are scheduled before a regular real completion route consumes them. It sits between the oscillation criterion of chapter 2123, the dyadic tolerance arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet also records compatibility with the explicit Cauchy modulus surface of chapter 292, while keeping completeness, quotient equality, selected limits, host equality, trichotomy, and choice outside the carrier.

Definition 6050.1 (Dyadic Cauchy oscillation modulus carrier)**.** *A* DyadicCauchyOscillationModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *O is the oscillation ledger, read through* CauchyOscillationCriterion$^{\uparrow}$ *as the finite record saying which pair of tail observations is compared for a requested tolerance.*
2. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance and modulus row, containing the requested dyadic radius, the displayed threshold, and the dyadic comparison budget used by O.*
3. *W is the visible tail-window schedule in* StreamName$^{\uparrow}$*, listing the finite windows beyond the threshold supplied by D and consumed by the oscillation read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the same windows and dyadic tolerance ledger.*
5. *E is the* Real$^{\uparrow}$ *seal row, visible only after O, D, W, R have fixed the finite oscillation-modulus route.*
6. *H stores componentwise* hsame *transport for O, D, W, R, E, C, P, N and the local packet shape.*
7. *C stores displayed* Cont *replay from a tolerance request to the dyadic threshold row, from that threshold to the visible tail windows, from the windows to the oscillation ledger, and from the ledger to the* RegSeqRat$^{\uparrow}$ *and* Real$^{\uparrow}$ *rows.*
8. *P is the* Pkg *provenance over* DyadicCauchyOscillationModulus$^{\uparrow}$*,* CauchyOscillationCriterion$^{\uparrow}$*,* CauchyModulus$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $O, D, W, R, E, H, C, P, N$. It has no coordinate for Real completeness, quotient stream equality, selected limits, host equality, trichotomy, countable choice, or an unbounded search over tails.

Theorem 6050.2 (Dyadic Cauchy oscillation modulus route)**.** *Every oscillation read exported by an accepted* $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ *carrier factors through*

$$D \longmapsto W \longmapsto O \longmapsto R \longmapsto E$$

together with the displayed hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModu}}$ *row. Thus the read is a finite dyadic modulus packet for tail oscillation, not a Real-completeness principle, quotient stream equality, selected limit, host equality, trichotomy row, or countable-choice selector.*

Proof. Project the accepted carrier in Definition 6050.1. The dyadic tolerance and modulus row D is the only coordinate that supplies the requested radius and finite threshold.

The visible tail-window schedule W is downstream of D. It displays the finite windows over which the oscillation ledger O may compare tail reads, so O cannot be consumed before the requested dyadic threshold and windows have been shown.

The rows R and E are downstream handoffs. They receive the same displayed windows and dyadic tolerance data after the oscillation ledger has fixed the tail comparison, and the structural rows H, C, P, N only transport, replay, source, and name this route.

No carrier coordinate supplies completeness, quotient equality, selected limits, host equality, trichotomy, choice, or an unbounded tail search. Hence every exported oscillation read is exhausted by the displayed finite route. □

Theorem 6050.3 (Dyadic Cauchy oscillation modulus NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ *carrier*

$$M = (O, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}}$ *surface consists exactly of carrier admission; dyadic tolerance exactness through D; tail-window scheduling through W; oscillation ledger routing through O;* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff through R;* $\mathsf{Real}^{\uparrow}$ *seal handoff through E; componentwise transport through H; displayed replay through C; provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6050.1. Carrier admission exposes precisely the rows $O, D, W, R, E, H, C, P, N$ and no other rows.

The dyadic and tail-facing obligations are D, W, O. The dyadic tolerance row fixes the requested radius and threshold, the StreamName row schedules only finite tail windows beyond that threshold, and the oscillation ledger records the displayed tail comparison for those windows.

The readback and seal obligations are R, E. The regular rational handoff reads the same scheduled windows, and the Real seal is downstream of the dyadic tolerance, window schedule, and oscillation ledger rather than a completeness principle.

Finally H, C, P, N transport, replay, source, and name the finite route. Completeness, quotient stream equality, selected limits, host equality, trichotomy, countable choice, ambient real arithmetic, and unbounded tail search are absent from the carrier, so none can appear as a local NameCert obligation. □

Theorem 6050.4 (Dyadic Cauchy oscillation modulus Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ *packet passes through the ordered finite route*

$$D \longmapsto W \longmapsto O \longmapsto R \longmapsto E.$$

The route cannot export $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream equality, selected limits, host equality, trichotomy, choice, or an unbounded tail search.*

Proof. Begin with a Real-facing consumer. By Definition 6050.1, the terminal row E is visible only after O, D, W, R have fixed the finite oscillation-modulus route.

Apply Definition 6050.2. It orders the consumer through D, W, O, R, E and keeps the structural rows as transport, replay, provenance, and naming data for the same route.

Use Definition 6050.3 for the obligation boundary. Since no excluded object is a carrier row or a NameCert obligation, a Real-facing read cannot insert completeness, quotient equality, selected limits, host equality, trichotomy, choice, or unbounded tail search between the finite windows and the terminal seal. □

Falsifiable prediction: Every $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer of a $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ packet must first display the dyadic tolerance and modulus row, the visible $\mathsf{StreamName}^{\uparrow}$ tail windows, and the $\mathsf{CauchyOscillationCriterion}^{\uparrow}$ ledger before any readback or seal row is consumed.

Independence witness: cauchyoscillationcriterion, cauchymodulus, dyadicratcore, streamname, regseqrat, real: $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries an oscillation ledger, dyadic tolerance and modulus data, visible tail-window scheduling, regular rational readback, a real seal boundary, componentwise transport, continuation replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}$ packet is generated from an oscillation ledger, a DyadicRatCore tolerance and modulus row, visible StreamName tail windows, RegSeqRat readback, a RealUp seal row, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6050.1, Definition 6050.2, Definition 6050.3, and Definition 6050.4 seed the finite Bishop–Bridges dyadic modulus packet for Cauchy oscillation reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyOscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, selected limits, host equality, trichotomy, countable choice, ambient real arithmetic, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, tail-window scheduling, oscillation ledger routing, regular rational handoff, real-seal handoff, transport, replay, provenance, and non-escape for BEDC.Derived.DyadicCauchyOscillationModulusUp.

Chapter 6051

A Concrete Naming Certificate for $\mathsf{FractionalSobolev}^{\uparrow}$

Following Adams and Fournier on fractional Sobolev and Slobodeckij spaces, the $\mathsf{FractionalSobolev}^{\uparrow}$ packet records the finite carrier surface by which a downstream analysis consumer reads non-integer smoothness without importing a completed quotient function space. It sits over the Sobolev weak-derivative surface of chapter 440, the weak-derivative observation route of chapter 2210, the dyadic metric tolerance ledger of chapter 2001, and the terminal real-name seal of chapter 38. The packet names only finite difference windows, dyadic scales, seminorm ledgers, componentwise transport, replay, provenance, and local naming evidence.

Definition 6051.1 (Fractional Sobolev carrier)**.** *A* $\mathsf{FractionalSobolev}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, W, D, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row naming the integer-order weak-derivative and magnitude surface already accepted by the consumer.*
2. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *observation row used to anchor finite weak-test reads when the fractional packet is compared to the ordinary Sobolev surface.*
3. *D is the* $\mathsf{DyadicMetric}^{\uparrow}$ *scale row naming the inspected dyadic radii, difference windows, and finite comparison thresholds.*
4. *Q is the fractional-difference seminorm ledger, recorded as finite pairs of displayed source windows and dyadic scale requests.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for the finite magnitude comparisons exposed by Q.*
6. *E is the terminal exclusion row stating that the packet has no coordinate for quotient functions, almost-everywhere equality, completed measure spaces, Bessel-potential transforms, or host interpolation functors.*
7. *H records componentwise* hsame *transport for the Sobolev source, weak-derivative row, dyadic metric scale row, seminorm ledger, real readback, exclusion row, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a fractional smoothness request through $S, W, D, Q, R, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{FractionalSobolev}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{DyadicMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FractionalSobolev}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, Q, R, E, H, C, P, N$. It has no coordinate for a host fractional Sobolev quotient, a selected representative, a completed L^p carrier, a Fourier transform, a Bessel-potential model, or an interpolation-space equality.

Theorem 6051.2 (Fractional Sobolev NameCert obligation)**.** *For every accepted* $\mathsf{FractionalSobolev}^{\uparrow}$ *carrier*

$$F = (S, W, D, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FractionalSobolev}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Sobolev}^{\uparrow}$ *source row, the* $\mathsf{WeakDerivative}^{\uparrow}$ *observation row, the* $\mathsf{DyadicMetric}^{\uparrow}$ *scale row, the finite fractional-difference seminorm ledger, the* $\mathsf{Real}^{\uparrow}$ *magnitude readback, the exclusion row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A consumer may use the packet as a finite non-integer smoothness certificate only after reading the displayed route*

$$\mathsf{Sobolev}^{\uparrow}, \quad \mathsf{WeakDerivative}^{\uparrow}, \quad \mathsf{DyadicMetric}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

The obligation exports no quotient function equality, no almost-everywhere representative, no completed measure carrier, no Fourier-side model, no Bessel-potential transform, and no host interpolation theorem.

Proof. Project the carrier of Definition 6051.1. The displayed rows are precisely the Sobolev source row, weak-derivative observation row, dyadic scale row, seminorm ledger, Real readback, exclusion row, and structural rows H, C, P, N.

The fractional read begins at S and W, then reaches the finite dyadic windows in D before the seminorm ledger Q can be consumed. The readback row R compares only the finite magnitude entries displayed by Q.

Transport and replay preserve that route componentwise through H and C, while P and N source and name the same finite packet. Since the remaining analytic objects are absent from the carrier coordinates, they cannot be exported by the local NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{FractionalSobolev}^{\uparrow}$ read is refuted if it reaches a fractional smoothness consumer without displaying the same $\mathsf{Sobolev}^{\uparrow}$ source row, $\mathsf{WeakDerivative}^{\uparrow}$ observation row, $\mathsf{DyadicMetric}^{\uparrow}$ scale row, finite seminorm ledger, $\mathsf{Real}^{\uparrow}$ readback, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: sobolev, weak-derivative, dyadic-metric, real: $\mathsf{FractionalSobolev}^{\uparrow}$ is not bijective to any listed sibling because it binds a Sobolev source, weak-test route, dyadic metric scale ledger, finite fractional-difference seminorm row, and Real readback as one non-integer smoothness packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FractionalSobolev}^{\uparrow}}$.

Constructive story: A $\mathsf{FractionalSobolev}^{\uparrow}$ packet is generated from a Sobolev source row, a WeakDerivative observation row, a DyadicMetric scale row, a finite fractional-difference seminorm ledger, a RealUp magnitude readback, an exclusion row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FractionalSobolev}^{\uparrow}}$, seed)

Scope closed: Definition 6051.1 and Definition 6051.2 seed the finite non-integer smoothness route from Sobolev and weak-derivative data through dyadic metric scales to a RealUp readback.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FractionalSobolev}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient function equality, almost-everywhere representatives, completed measure carriers, Fourier-side models, Bessel-potential transforms, host interpolation theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Sobolev source exactness, WeakDerivative observation routing, DyadicMetric scale nonescape, finite seminorm ledger exactness, RealUp readback nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FractionalSobolevUp.

The horizontal dependency route for $\mathsf{FractionalSobolev}^{\uparrow}$ is the finite non-integer regularity edge from chapter 440 through the normed and Hilbert magnitude rows of chapter 89 and chapter 91, with distribution-style and derivative-style observation supplied by chapter 186, chapter 122, and chapter 2210. Thus the packet consumes existing concrete-instance regularity rows as sibling data before exposing its fractional seminorm ledger; it does not create an ambient quotient function space or a completed interpolation theorem.

Chapter 6052

A Concrete Naming Certificate for $\mathsf{CauchyCondensationTailLock}^{\uparrow}$

Following Cauchy's condensation test and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ packet records the finite tail route by which condensation evidence reaches a regular rational readback and then the Real seal. It consumes the dyadic-block condensation carrier of chapter 1464, the regular rational handoff of chapter 301, the dyadic tolerance surface of chapter 291, and the terminal Real boundary of chapter 38. The certificate locks a displayed condensation tail to one completion-facing route; it does not import selected sums, quotient series equality, countable choice, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6052.1 (Cauchy condensation tail-lock carrier)**.** *A* $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, W, B, L, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the monotone nonnegative* $\mathsf{Series}^{\uparrow}$ *source whose finite summand windows are consumed by the condensation route.*
2. *W is the summand-window ledger inherited from the accepted* $\mathsf{CauchyCondensation}^{\uparrow}$ *carrier.*
3. *B is the dyadic block schedule, including displayed block endpoints, powers of two, and block lengths.*
4. *L is the block-sum ledger, storing the original block sums and the condensed dyadic terms on the same schedule.*
5. *T is the tail-bound row, recording the dyadic tolerances and finite domination comparisons read through* $\mathsf{DyadicRatCore}^{\uparrow}$*.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row produced from S, W, B, L, T.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal boundary attached to the handoff row R.*
8. *H records componentwise* hsame *transports for the source, windows, schedule, block sums, tail bounds, readback, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from each finite precision request through $S, W, B, L, T, R, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{Series}^{\uparrow}$, $\mathsf{CauchyCondensation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTailLock}^{\uparrow}}$ *row.*

The classifier compares accepted tail-lock packets only through these rows. It has no coordinate for quotient series equality, a selected infinite sum, countable choice, trichotomy beyond displayed windows, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCondensationTailLockUp.CauchyCondensationTailLockTasteGate single carrier alignment`

Theorem 6052.2 (Cauchy condensation tail-lock NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ *carrier* $Q = (S, W, B, L, T, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationTailLoc}}$ *surface consists exactly of source exposure for* S*, summand-window coverage for* W*, dyadic block scheduling for* B*, block-sum ledger exposure for* L*, dyadic tail-bound visibility for* T*,* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff exposure for* R*,* $\mathsf{Real}^{\uparrow}$ *seal routing for* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, sibling provenance through* P*, and local naming through* N*.*

No tail-lock obligation is stored outside the displayed finite packet.

Proof. Project the accepted carrier of Definition 6052.1. The tuple $S, W, B, L, T, R, E, H, C, P, N$ is the whole public packet, so carrier admission and local naming are exactly the displayed rows.

The analytic route begins with the finite condensation surface of chapter 1464. The source row S, summand windows W, dyadic schedule B, and block ledger L determine which finite terms can be inspected.

The tail-bound row T reads its dyadic tolerances through the $\mathsf{DyadicRatCore}^{\uparrow}$ surface and sends the resulting finite comparison to the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R. The seal row E is downstream of that handoff.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same route. Since the carrier has no further coordinate, the listed rows exhaust the local certificate. □

Theorem 6052.3 (Cauchy condensation tail-lock Real-seal route)**.** *Every accepted* $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ *carrier routes a condensation tail to a* $\mathsf{Real}^{\uparrow}$ *seal only through the finite path*

$$S, W, B, L, T \longmapsto R \longmapsto E.$$

The route is locked to the same $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row before the Real seal is read. No selected sum, quotient series equality, countable choice, trichotomy premise, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness coordinate can produce a second seal-facing path.*

Proof. Start with an accepted tail-lock carrier and read the rows exposed by Definition 6052.2. The only analytic input available before the handoff is the finite source-window, dyadic-schedule, block-ledger, and tail-bound data S, W, B, L, T.

Apply the condensation handoff route of Definition 1464.6. It forces the condensation comparison through the displayed dyadic-block surface before any regular rational readback can be consumed.

Now read the Real seal boundary using Definition 1464.7. The seal-facing input is the handoff row R together with the structural rows H, C, P, N, so E cannot be reached by an alternate quotient, selected-sum, choice, trichotomy, host-equality, or completeness coordinate. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without first reading the displayed rows S, W, B, L, T and the $\mathsf{RegSeqRat}^{\uparrow}$ handoff row R refutes the carrier surface.

Independence witness: cauchycondensation, regseqrat, dyadicratcore, real: no listed sibling supplies the whole tail-lock packet $Q = (S, W, B, L, T, R, E, H, C, P, N)$, because the siblings expose the condensation carrier, regular rational readback, dyadic tolerance surface, or seal boundary separately while this chapter binds them into one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationTailLock}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCondensationTailLock}^{\uparrow}$ packet is generated from a Series source, condensation windows, dyadic block schedule, block-sum ledger, dyadic tail-bound row, RegSeqRat handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTailLock}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6052.1, Definition 6052.2, and Definition 6052.3 seed the finite tail-lock route from condensation evidence through dyadic tolerance and RegSeqRat readback to a RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCondensationTailLock}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient series equality, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source exposure, summand-window coverage, dyadic block scheduling, block-sum ledger exposure, dyadic tail-bound visibility, RegSeqRat handoff exposure, Real seal routing, transport stability, replay stability, and non-escape for BEDC.Derived.CauchyCondensationTailLockUp.

Chapter 6053

A Concrete Naming Certificate for $\mathsf{BanachContractionEndpoint}^{\uparrow}$

Following Banach and Bishop–Bridges on constructive fixed-point iteration, the $\mathsf{BanachContractionEndpoint}^{\uparrow}$ packet records the finite fixed-endpoint seal for a contraction orbit over a complete separated metric source. It sits after the contraction certificate of chapter 3529, the contraction-map surface of chapter 1418, the Picard orbit interface of chapter 2359, the complete metric service of chapter 128, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the endpoint read as displayed NameCert data; it is not a host-selected fixed point, a quotient completion, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6053.1 (Banach contraction endpoint carrier)**.** *A* $\mathsf{BanachContractionEndpoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (X, F, O, M, L, U, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *X is the complete separated metric source row consumed by the contraction.*
2. *F is the contraction source row, naming the displayed* $\mathsf{BanachContraction}^{\uparrow}$ *and* $\mathsf{ContractionMapping}^{\uparrow}$ *data.*
3. *O is the* $\mathsf{PicardIteration}^{\uparrow}$ *orbit row generated from the displayed source and contraction data.*
4. *M is the Cauchy modulus row for the orbit, recording the finite tail budget used by the completion service.*
5. *L is the complete-metric endpoint row reached from the displayed orbit and modulus.*
6. *U is the separated uniqueness row, showing that any accepted endpoint read is forced through the same completion boundary.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the endpoint observations.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after X, F, O, M, L, U, Q have been displayed.*
9. *H records componentwise* hsame *transport for source, contraction, orbit, modulus, endpoint, uniqueness, readback, seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay for the same endpoint route.*
11. *P is the* Pkg *provenance over* $\mathsf{BanachContractionEndpoint}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{ContractionMapping}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. N is the local $\mathsf{NameCert}_{\mathsf{BanachContractionEndpoint}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $X, F, O, M, L, U, Q, R, H, C, P, N$. It has no coordinate for a host-selected fixed point, quotient completion, ambient real completeness, countable choice, or a fixed-point theorem detached from the displayed orbit and endpoint seal.

Lean4 ✓ `BEDC.Derived.BanachContractionEndpointUp.BanachContractionEndpointTasteGate single carrier alignment`

Theorem 6053.2 (Banach contraction endpoint NameCert obligations)**.** *For every accepted* BanachContractionEnd *carrier*

$$E = (X, F, O, M, L, U, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BanachContractionEndpoint}^{\uparrow}}$ *surface consists exactly of complete separated metric admission through X, contraction-source exposure through F, Picard-orbit exposure through O, Cauchy-modulus exposure through M, complete-metric endpoint exposure through L, separated uniqueness through U, regular rational readback through Q, real seal exposure through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6053.1. The endpoint-facing rows are X, F, O, M, L, U, Q, R, and the structural rows are H, C, P, N.

The contraction route is read in order. Rows X and F provide the complete separated metric source and contraction data, O records the Picard orbit, and M supplies the finite Cauchy budget that lets the completion service expose L.

The endpoint is then sealed only through U, Q, R, H, C, P, N. Separated uniqueness, regular rational readback, and the terminal real seal remain displayed rows of the same packet, so the listed rows exhaust the local NameCert obligations. □

Theorem 6053.3 (Banach contraction endpoint nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or fixed-endpoint consumer of an accepted* $\mathsf{BanachContractionEndpoint}^{\uparrow}$ *packet reaches the endpoint only through*

$$\mathsf{CompleteMetric}^{\uparrow},\ \mathsf{BanachContraction}^{\uparrow},\ \mathsf{ContractionMapping}^{\uparrow},\ \mathsf{PicardIteration}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed by the rows $X, F, O, M, L, U, Q, R, H, C, P, N$. No consumer may replace that route by a host-selected fixed point, quotient completion, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a fixed-point theorem detached from the Picard orbit.*

Proof. Apply the obligations theorem Definition 6053.2. It requires every endpoint read to expose the complete source, contraction source, Picard orbit, Cauchy modulus, completion endpoint, separated uniqueness, readback, and seal rows.

The terminal real seal R is downstream of Q, and Q is downstream of the displayed endpoint row L. Thus a Real-facing consumer cannot reach R without first reading the orbit and completion route carried by O, M, L.

Since the classifier has no coordinate for host-selected fixed points, quotient completion, countable choice, ambient completeness, or detached fixed-point theorems, those reads cannot be exported by an accepted packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachContractionEndpoint}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BanachContractionEndpoint}^{\uparrow}$ packet is generated from a complete separated metric source, contraction data, a Picard orbit, a Cauchy modulus, a complete-metric endpoint row, separated uniqueness, RegSeqRat

readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{BanachContractionEndpoint}\uparrow}$, seed)

Scope closed: Definition 6053.1, Definition 6053.2, and Definition 6053.3 seed the fixed-endpoint certificate for contraction orbits over complete separated metric sources.

Formal verification: FormalStatus(NameCert$_{\mathsf{BanachContractionEndpoint}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host-selected fixed points, quotient completions, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, contraction-source routing, Picard orbit exposure, Cauchy-modulus exactness, complete-metric endpoint exposure, separated uniqueness, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BanachContractionEndpointUp.

Chapter 6054

A Concrete Naming Certificate for $\mathsf{FiniteIntervalChoice}^{\uparrow}$

Following Bishop and Bridges on finite choice over displayed finite covers, the $\mathsf{FiniteIntervalChoice}^{\uparrow}$ packet records the finite selector surface needed when a closed bounded interval net is read through finite stream windows. It sits over the finite mesh route of chapter 1250, the observation schedule of chapter 5916, the stream-window surface of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names only choices made inside an already displayed finite row; it does not import countable choice, an ambient selector, trichotomy, quotient real equality, or host compactness.

Definition 6054.1 (Finite interval choice carrier)**.** *A* $\mathsf{FiniteIntervalChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (K, Q, L, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *cell-data row, including the located interval source, finite mesh scale, rational cells, dyadic refinements, center rows, and finite coverage ledger.*
2. *Q is the* $\mathsf{FiniteStream}^{\uparrow}$ *request row that lists exactly the interval-net reads demanded by the consumer.*
3. *L is the finite* $\mathsf{List}^{\uparrow}$ *spine carried by the request row Q.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed over the same finite request row and list spine.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary available after K, Q, L, W have been displayed.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row attached to the same interval-net cells and stream windows.*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed cell, request, window, readback, and tolerance rows.*
8. *H records componentwise* hsame *transport for the interval net, request row, list spine, stream windows, regular readbacks, dyadic tolerances, Real seal, replay row, provenance row, and local naming row.*
9. *C records displayed* Cont *replay from an interval-net request through $K, Q, L, W, R, D, E, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{FiniteIntervalChoice}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{FiniteStream}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntervalChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, Q, L, W, R, D, E, H, C, P, N$. No coordinate stores a hidden chooser, countable family of choices, host subcover selector, quotient real equality, trichotomy branch, or ambient completeness principle.

Lean4 ✓ `BEDC.Derived.FiniteIntervalChoiceUp.FiniteIntervalChoiceTasteGate single carrier alignment`

Theorem 6054.2 (Finite interval choice NameCert obligations)**.** *For every accepted* $\mathsf{FiniteIntervalChoice}^{\uparrow}$ *carrier*

$$I = (K, Q, L, W, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteIntervalChoice}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *cell-data row, the* $\mathsf{FiniteStream}^{\uparrow}$ *request row, the finite* $\mathsf{List}^{\uparrow}$ *spine, the* $\mathsf{StreamName}^{\uparrow}$ *windows, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback boundary, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The finite-choice clause says that each selected interval-net read is indexed by the displayed finite request row Q and list spine L. The window clause says that every stream, regular-rational, dyadic, or Real-facing consumer must pass through the same Q, L, W route before reading R, D, E.*

Proof. Project the carrier of Definition 6054.1. The displayed rows are exactly the interval-net cell data, finite request row, list spine, stream window, regular readback, dyadic tolerance, Real seal, and structural rows H, C, P, N.

The finite-choice clause is row exactness. A choice made by the packet is not a host selector; it is the finite coordinate of Q as represented by L and then read by W.

The downstream rows preserve that finite coordinate. The readback row R, tolerance row D, and Real seal E appear only after K, Q, L, W, while H, C, P, N transport, replay, source, and name the same route. Hence no countable-choice, subcover-selector, quotient-equality, trichotomy, or ambient-completeness row is available. □

Theorem 6054.3 (Finite interval choice closed-bounded-net handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read accepted by a* $\mathsf{FiniteIntervalChoice}^{\uparrow}$ *packet factors through the finite route*

$$K, Q, L, W, R, D, E, H, C, P, N.$$

Here K supplies the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ *cell data, Q and L display the finite request spine, W exposes the corresponding* $\mathsf{StreamName}^{\uparrow}$ *windows, R supplies the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, D supplies the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and E supplies the terminal* $\mathsf{Real}^{\uparrow}$ *seal. The handoff gives L10 interval-net consumers a constructive finite-choice certificate without countable choice, hidden selectors, quotient real equality, or ambient real completeness.*

Proof. Start with the NameCert obligations of Definition 6054.2. They bind the interval-net row K to the same finite request row Q and list spine L before any stream window is exposed.

The StreamName, RegSeqRat, DyadicRatCore, and Real rows are then downstream of the same request coordinate. The displayed replay row C prevents a consumer from replacing Q by a sibling request row, and the transport row H preserves equality only componentwise along the displayed packet.

Finally the provenance and local naming rows P, N name the same finite route. Since the carrier has no coordinate for a hidden selector, a countable choice family, quotient real equality,

trichotomy, or ambient completeness, the closed-bounded-net handoff is exactly the constructive finite-choice route displayed above. □

Falsifiable prediction: An accepted $\mathsf{FiniteIntervalChoice}^{\uparrow}$ read is refuted if it reaches a $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ consumer without displaying the same finite request row, finite list spine, window row, readback boundary, dyadic tolerance row, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Independence witness: closedboundedintervalnet, finite_stream, streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteIntervalChoice}^{\uparrow}$ is not bijective to any listed sibling because it binds interval-net cell data to one displayed finite request row and the downstream readback, tolerance, and Real seal rows as a single finite-choice packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntervalChoice}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntervalChoice}^{\uparrow}$ packet is generated from ClosedBoundedIntervalNet cell data, a FiniteStream request row, a finite ListUp spine, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteIntervalChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6054.1, Definition 6054.2, and Definition 6054.3 seed the finite-choice route for L10 interval-net and finite-stream reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteIntervalChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, hidden host selectors, open-cover compactness, quotient real equality, trichotomy, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite request exactness, interval-net handoff, StreamName and RegSeqRat readback exactness, DyadicRatCore tolerance nonescape, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteIntervalChoiceUp.

Chapter 6055

A Concrete Naming Certificate for $\mathsf{MinimalRealization}^{\uparrow}$

Following Kalman on finite-dimensional realization theory, the $\mathsf{MinimalRealization}^{\uparrow}$ packet records the displayed finite control-system rows that witness a realization with no smaller state ledger carrying the same input-output behavior. It sits after the observation surface of chapter 294, the reachability surface of chapter 297, the reachability Gramian of chapter 455, the observability Gramian of chapter 5787, and the finite $\mathsf{LinMap}^{\uparrow}$ and $\mathsf{Mat}^{\uparrow}$ rows of chapter 48 and chapter 49. The packet names only finite plant, input, output, transport, replay, provenance, and local naming rows; it does not import a host quotient state space, a canonical Kalman decomposition, or an analytic transfer-function equality.

Definition 6055.1 (MinimalRealization carrier)**.** *A* $\mathsf{MinimalRealization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, U, Y, A, B, C, D, Q, O, G_o, G_r, n, m, I, L, T, P, N)$$

with the following displayed rows:

1. *X is the finite plant state row, U is the input row, and Y is the output row, each read through displayed* $\mathsf{Mat}^{\uparrow}$ *or* $\mathsf{LinMap}^{\uparrow}$ *coordinates;*
2. *$A : X \to X$, $B : U \to X$, $C : X \to Y$, and $D : U \to Y$ are the finite transition, input, output, and feedthrough rows of the plant;*
3. *Q is the reachability row consumed from* $\mathsf{ControlControllability}^{\uparrow}$ *and* $\mathsf{ReachabilityGramian}^{\uparrow}$*;*
4. *O is the observability row consumed from* $\mathsf{ControlObservability}^{\uparrow}$ *and* $\mathsf{ObservabilityGramian}^{\uparrow}$*;*
5. *G_o is the state-dimension ledger and G_r is the reduced-state ledger, both recorded as finite natural rows with matrix-rank readback;*
6. *n is the displayed state-dimension row and m is the reduced-state candidate row compared against it;*
7. *I is the isomorphism witness between any accepted reduced realization at dimension n and the displayed realization rows;*
8. *L records the minimality ledger saying that every realization with the same displayed finite input-output rows has reduced-state ledger m admitting a transport into n;*
9. *T records componentwise* hsame *transport and displayed* Cont *replay for the plant, input, output, reachability, observability, dimension, reduced-state, isomorphism, and minimality rows;*

10. *P is the* Pkg *provenance over* $\mathsf{MinimalRealization}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{ReachabilityGramian}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, BHist, msame, hsame, psame, Cont, *and* NameCert*;*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.MinimalRealizationUp.MinimalRealizationTasteGate single carrier alignment`

Theorem 6055.2 (MinimalRealization NameCert obligations)**.** *For the carrier of Definition 6055.1, the local* $\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}$ *surface consists exactly of finite plant admission for X, U, Y, A, B, C, D, reachability admission through Q, observability admission through O, state-dimension and reduced-state ledger admission through G_o, G_r, n, m, isomorphism-witness admission through I, minimality-ledger admission through L, componentwise* hsame *and* msame *transport through T, displayed* Cont *replay through T, package provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 6055.1. The plant rows X, U, Y, A, B, C, D are finite $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$ reads, so the local certificate begins with displayed finite control-system data.

The rows Q and O are consumed from chapter 297, chapter 294, chapter 455, and chapter 5787. They provide the reachability and observability witnesses used by the minimality ledger without introducing a host reachable quotient or hidden unobservable subspace.

Finally $G_o, G_r, n, m, I, L, T, P, N$ record finite dimension comparison, reduced-state comparison, the displayed isomorphism witness, transport, replay, provenance, and local naming. Repacking those rows gives the stated NameCert obligation surface and no additional coordinate is available. □

Theorem 6055.3 (Controllability-observability minimality)**.** *Let $R = (X, U, Y, A, B, C, D, Q, O, G_o, G_r, n, m, I, L,$ be an accepted* $\mathsf{MinimalRealization}^{\uparrow}$ *packet. Any downstream realization consumer that reads R as a minimal finite realization must pass first through the displayed* $\mathsf{ControlControllability}^{\uparrow}$ *row Q, the displayed* $\mathsf{ControlObservability}^{\uparrow}$ *row O, the finite Gramian rows named by* $\mathsf{ReachabilityGramian}^{\uparrow}$ *and* $\mathsf{ObservabilityGramian}^{\uparrow}$*, the state-dimension and reduced-state ledgers G_o, G_r, n, m, the isomorphism witness I, the minimality ledger L, componentwise transport and replay through T, package provenance through P, and the local* $\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}$ *row N.*

The exported minimality claim supplies no host quotient state space, no selected canonical realization, no unbounded trajectory semantics, and no transfer-function equality outside the displayed finite plant rows.

Proof. Start with Definition 6055.2. It exposes exactly the plant, reachability, observability, dimension, reduced-state, isomorphism, minimality, transport, replay, provenance, and local naming rows.

The controllability side is read through Q and the reachability Gramian route, while the observability side is read through O and the observability Gramian route. These are finite BHist rows over $\mathsf{Mat}^{\uparrow}$ and $\mathsf{LinMap}^{\uparrow}$, so the consumer cannot replace them by an ambient state-space quotient.

The ledger L compares every same-input-output realization against the displayed dimension rows n, m, and I is the only isomorphism witness exported at the same dimension. Therefore the minimality read is the finite BEDC route named by R, not a host canonical-form assertion or analytic transfer-function identification. □

Falsifiable prediction: An accepted $\mathsf{MinimalRealization}^{\uparrow}$ packet is refuted by any realization consumer that claims minimality while bypassing either the displayed reachability row Q, the

displayed observability row O, the dimension comparison rows G_o, G_r, n, m, or the isomorphism and minimality ledgers I, L.

Independence witness: controlobservability, controlcontrollability, reachabilitygramian, observabilitygramian, linearmap, matrix: none is bijective to $\mathsf{MinimalRealization}^{\uparrow}$ because the minimal-realization packet jointly carries plant, input, output, reachability, observability, dimension, reduced-state, isomorphism, minimality, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalRealization}^{\uparrow}$ packet is generated from finite plant, input, output, reachability, observability, dimension, reduced-state, isomorphism, minimality, transport, replay, provenance, and local naming rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6055.1, Definition 6055.2, Definition 6055.3, the displayed falsifiable prediction, and the displayed independence witness seed the finite minimal-realization NameCert surface over control, Gramian, linear-map, and matrix rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalRealization}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, canonical Kalman decomposition, transfer-function equality, or host quotient state-space semantics.

Upgrade path: Obligation closure requires checked carrier admission, reachability and observability admission, dimension and reduced-state comparison, isomorphism-witness transport, minimality-ledger exactness, continuation replay, package provenance, and local NameCert exhaustion for BEDC.Derived.MinimalRealizationUp.

Chapter 6056

A Concrete Naming Certificate for $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$

Following Bishop–Bridges constructive analysis, Specker's monotone bounded sequence obstruction, and the Troelstra–van Dalen account of weak omniscience principles, the $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ packet records the finite L10 boundary at which Real and Completion reads may cite Specker-style obstruction data without importing classical completeness. It sits after the Specker boundary of chapter 3516, the LLPO boundary of chapter 1978, the WLPO boundary of chapter 4062, the weak-omniscience boundary of chapter 5980, the finite observation interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a human-origin completion boundary; it refuses $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient primitives, selected limits, and omniscience leakage.

Definition 6056.1 (Bishop–Specker completion-boundary carrier)**.** *A* $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (S, L, W, O, A, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *row carrying the Specker obstruction surface.*
2. *L is the* $\mathsf{LLPOBoundary}^{\uparrow}$ *row recording the lesser-limited omniscience refusal attached to the same finite observation.*
3. *W is the* $\mathsf{WLPOBoundary}^{\uparrow}$ *row recording the weak-limited omniscience refusal attached to the same finite observation.*
4. *O is the* $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$ *row collecting the weak-omniscience nonescape ledger.*
5. *A is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected prefixes of the Specker stream and the omniscience-refusal ledgers.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row available from the displayed windows only.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the rational bound, requested precision, and comparison budget used by R.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, L, W, O, A, R, D have all been displayed.*
9. *H records componentwise* hsame *transport for the obstruction, refusal, window, readback, tolerance, seal, provenance, and local naming rows.*

10. *C records displayed* Cont *replay of the route*

$$S, L, W, O \longmapsto A, D \longmapsto R \longmapsto E.$$

11. *P is the* Pkg *provenance over* $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$, $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, $\mathsf{LLPOBoundary}^{\uparrow}$, $\mathsf{WLPOBoundary}^{\uparrow}$, $\mathsf{WeakOmniscienceBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}}$ *row.*

The carrier predicate asserts that these rows form one finite Real-and-Completion boundary packet. The classifier compares accepted packets only by componentwise agreement of $S, L, W, O, A, R, D, E, H, C, P, N$ under the displayed hsame *and* Cont *routes; it has no coordinate for* $\mathsf{Real}^{\uparrow}$ *completeness, trichotomy, quotient real equality, selected limits, countable choice, LLPO, WLPO, or a detached omniscience principle.*

Lean4 ✓ `BEDC.Derived.BishopSpeckerCompletionBoundaryUp.BishopSpeckerCompletionBoundaryTasteGate`
`single carrier alignment`

Theorem 6056.2 (Bishop–Specker completion-boundary NameCert obligations)**.** *For every accepted* $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ *carrier*

$$B = (S, L, W, O, A, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, Specker-obstruction exposure through S, LLPO-refusal exposure through L, WLPO-refusal exposure through W, weak-omniscience refusal through O, finite-window exposure through A, regular rational read-back through R, dyadic tolerance accounting through D, real-seal handoff through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The projection exposes the Specker obstruction row, the LLPO refusal row, the WLPO refusal row, the weak-omniscience refusal row, the finite StreamName window, the regular rational readback, the dyadic tolerance ledger, the Real seal, transport, continuation, provenance, and local naming rows.

The analytic obligations are exactly the visible obstruction and readback rows. The Specker boundary supplies the monotone bounded obstruction, the three omniscience rows record which principles are not available, and the finite window plus dyadic ledger determine the only regular-rational read permitted by the packet.

The structural obligations are H, C, P, N. They transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient primitives, selected limits, LLPO, WLPO, countable choice, or a detached omniscience principle, no additional local NameCert obligation can be read. □

Theorem 6056.3 (Finite-window completion boundary)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion-facing consumer of an accepted* $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ *packet reads the route*

$$\mathsf{SpeckerSequenceBoundary}^{\uparrow},\ \mathsf{LLPOBoundary}^{\uparrow},\ \mathsf{WLPOBoundary}^{\uparrow},\ \mathsf{WeakOmniscienceBoundary}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{Dy}$$

through the displayed rows $S, L, W, O, A, D, R, E, H, C, P, N$. The handoff supplies a finite obstruction-and-readback boundary for L10 Real and Completion arguments, not a proof that $\mathsf{Real}^{\uparrow}$ *is complete, not a selected-limit construction, and not a quotient primitive.*

Proof. Start with the four obstruction rows. The Specker boundary row S supplies the nonlocated completion obstruction, while L, W, O attach the LLPO, WLPO, and weak-omniscience refusal ledgers to the same finite observation surface.

Next read the completion-facing window. The $\mathsf{StreamName}^{\uparrow}$ row A selects the finite prefixes inspected by the consumer, and the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger D records the tolerance budget used by the $\mathsf{RegSeqRat}^{\uparrow}$ readback R.

Finally pass through the terminal Real seal. The row E may be consumed only after the obstruction, refusal, window, readback, and tolerance rows are displayed, and H, C, P, N preserve that route. The consumer therefore receives a finite boundary packet, not an ambient completion theorem, selected host limit, trichotomy branch, or quotient equality primitive. □

Theorem 6056.4 (Omniscience nonescape)**.** *No accepted* $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ *packet exports* $\mathsf{LimitedPrincipleOmniscience}^{\uparrow}$, $\mathsf{LesserLimitedOmniscience}^{\uparrow}$, $\mathsf{WLPOBoundary}^{\uparrow}$ *as an available decision principle, weak omniscience as a usable selector, trichotomy for real comparison, quotient equality of real names, selected limits, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every public read factors through the finite rows* $S, L, W, O, A, R, D, E, H, C, P, N$ *of Definition 6056.1.*

Proof. Inspect the coordinates of the carrier. The only analytic rows are the Specker obstruction, three refusal ledgers, finite StreamName window, regular rational readback, dyadic tolerance ledger, and terminal Real seal.

The refusal rows are negative boundary data. They record that the packet is read without LLPO, WLPO, or weak-omniscience escape; they do not introduce a decision function, a selector for limits, or a trichotomy branch.

The remaining rows are structural. Componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming preserve the displayed route but add no new analytic coordinate. Thus every public read remains inside $S, L, W, O, A, R, D, E, H, C, P, N$, and the listed omniscience, quotient, selection, trichotomy, and completion principles cannot be extracted. □

Falsifiable prediction: Every $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ read factors through a SpeckerSequenceBoundary obstruction row, LLPOBoundary refusal row, WLPOBoundary refusal row, WeakOmniscienceBoundary refusal row, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: specker-sequence-boundary, llpo-boundary, wlpo-boundary, weak-omniscience-boundary, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Specker obstruction, three omniscience-refusal ledgers, finite observation window, regular rational readback, dyadic tolerance accounting, Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}$ packet is generated from a SpeckerSequenceBoundary obstruction row, LLPOBoundary refusal row, WLPOBoundary refusal row, WeakOmniscienceBoundary refusal row, finite StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6056.1, Definition 6056.2, Definition 6056.3, and Definition 6056.4 seed the finite Bishop–Specker completion boundary over SpeckerSequenceBoundary, LLPOBoundary, WLPOBoundary, WeakOmniscienceBoundary, StreamName, RegSeqRat, DyadicRatCore, Real, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopSpeckerCompletionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, trichotomy, quotient real equality, selected limits, countable choice, LLPO, WLPO, weak omniscience, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, finite-window completion handoff, omniscience nonescape, and TasteGate field faithfulness for BEDC.Derived.BishopSpeckerCompletionBoundaryUp.

Chapter 6057

A Concrete Naming Certificate for $\mathsf{BishopRealCompletionAdjoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopRealCompletionAdjoint}^{\uparrow}$ packet records the finite adjoint surface by which a Bishop regular real name is admitted into the completion route and read back through the same L10 data. It sits after the regular completion monad of chapter 2136, the Cauchy completion functor of chapter 423, the universal property row of chapter 397, the regular rational readback of chapter 301, and the visible real-name seal of chapter 38. The packet names a non-quotient public completion-adjoint boundary; it does not assert ambient completeness of $\mathsf{Real}^{\uparrow}$, choose completed limits, compare host real values, or identify quotient streams.

Definition 6057.1 (Bishop real completion adjoint carrier)**.** *A* $\mathsf{BishopRealCompletionAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, F, U, R, E, J, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *row, recording the finite unit and bind schedule for completion-facing regular names.*
2. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row, recording the source-to-completion transport used by functorial consumers.*
3. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, recording extension existence and uniqueness only as displayed finite ledger data.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached by the source windows and by the completion-facing unit request.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after M, F, U, R have been read.*
6. *J is the unit-extension handoff row, matching the unit row of M with the extension row of U through the functorial surface F.*
7. *T is the triangle-ledger row, recording exactness for the unit leg, extension leg, readback leg, and Real seal leg.*
8. *H records componentwise* hsame *transport for the monad, functor, universal-property, readback, seal, handoff, triangle-ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, F, U \longmapsto R \longmapsto E \longmapsto J, T, H, P, N.$$

10. *P is the* Pkg *provenance over* BishopRealCompletionAdjoint$^{\uparrow}$, RegularCauchyCompletionMonad$^{\uparrow}$, CauchyCompletionFunctor$^{\uparrow}$, CompletionUniversalProperty$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* NameCert$_{\text{BishopRealCompletionAdjoint}^{\uparrow}}$ *row.*

The accepted carrier is exactly the displayed packet A. It has no coordinate for quotient completion equality, selected limits, host real equality, countable choice, ambient Real$^{\uparrow}$ *completeness, or a detached adjunction theorem outside the finite rows.*

Theorem 6057.2 (Bishop real completion adjoint NameCert obligations)**.** *For every accepted carrier $A = (M, F, U, R, E, J, T, H, C, P, N)$, the local* NameCert$_{\text{BishopRealCompletionAdjoint}^{\uparrow}}$ *obligations are exhausted by carrier admission for the displayed monad, functor, universal-property, readback, seal, unit-extension handoff, triangle-ledger, transport, replay, provenance, and local naming rows. No obligation may call on quotient completion equality, a selected limit, host real equality, countable choice, ambient* Real$^{\uparrow}$ *completeness, or an external adjunction principle.*

Proof. Project the carrier of Definition 6057.1. The only rows visible to the local classifier are $M, F, U, R, E, J, T, H, C, P, N$.

Read the completion route in order. The monad row supplies the finite unit and bind schedule, the functor row supplies the completion-facing transport, and the universal-property row supplies the displayed extension ledger. Only after those rows are present may the readback row R and terminal seal row E be consumed.

The handoff and exactness rows are structural over that same route. The row J compares the unit and extension legs, T records the triangle ledger, H transports displayed coordinates, C replays the permitted reads, and P, N record provenance and local naming. Since the carrier has no excluded coordinate, no local obligation can export one. □

Theorem 6057.3 (Bishop real completion adjoint unit-extension handoff)**.** *Let $A = (M, F, U, R, E, J, T, H, C, P, N$ be an accepted* BishopRealCompletionAdjoint$^{\uparrow}$ *carrier. Every unit-extension read factors first through the* RegularCauchyCompletionMonad$^{\uparrow}$ *unit row in M, then through the* CauchyCompletionFunctor$^{\uparrow}$ *source-to-completion row F, and only then through the* CompletionUniversalProperty$^{\uparrow}$ *extension ledger U. The handoff row J records this finite route before the* RegSeqRat$^{\uparrow}$ *readback R and terminal* Real$^{\uparrow}$ *seal E are exposed.*

Proof. Start with the monad row. The unit part of M admits source regular data into the completion interface through finite windows and tolerance rows.

Next apply the functor row F. It transports the same displayed source data to the completion-facing route without introducing a quotient completion object or selected representative.

Finally read the universal-property row U. The handoff row J can compare unit and extension data only after M and F have been projected. The readback R, seal E, transport H, replay C, provenance P, and local naming N preserve that order, so the consumer receives a finite handoff rather than a host adjunction. □

Theorem 6057.4 (Bishop real completion adjoint triangle ledger exactness)**.** *For every accepted* BishopRealCompletionAdjoint$^{\uparrow}$ *carrier $A = (M, F, U, R, E, J, T, H, C, P, N)$, the triangle-ledger row*

T is exact for the displayed unit leg, extension leg, regular readback leg, and terminal Real seal leg. Any completion-facing consumer that reads the public adjoint surface must account for the rows

$$M, F, U, R, E, J, T, H, C, P, N$$

in that order, and it obtains no quotient stream equality, selected completion point, host real equality, countable-choice schedule, ambient completeness theorem, or detached adjunction proof.

Proof. Project the accepted carrier. The triangle ledger T is not an independent theorem row; it is displayed beside the monad, functor, universal-property, readback, seal, and handoff rows.

Use the unit-extension handoff of Definition 6057.3. It places J downstream of M, F, U and upstream of R, E, so every triangle comparison is read through the same finite route.

The structural rows finish the ledger accounting. Componentwise transport in H preserves the displayed coordinates, replay in C records the allowed reads, and P, N bind provenance and naming. Because the carrier contains no excluded row, exactness of T remains a finite ledger statement and cannot become a quotient, choice, host-equality, completeness, or external-adjunction claim. □

Theorem 6057.5 (Bishop real completion adjoint readback nonescape)**.** *Let $A = (M, F, U, R, E, J, T, H, C, P, N)$ be an accepted* $\mathsf{BishopRealCompletionAdjoint}^{\uparrow}$ *carrier. Every readback through R, E remains downstream of the displayed monad, functor, universal-property, unit-extension handoff, triangle-ledger, transport, replay, provenance, and local naming rows $M, F, U, J, T, H, C, P, N$. The readback cannot become quotient completion equality, a selected completed point, host real equality, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or a detached adjunction theorem.*

Proof. Project the carrier of Definition 6057.1. The only readback and seal coordinates are R and E, and both are displayed inside the same packet as $M, F, U, J, T, H, C, P, N$.

Use the unit-extension handoff of Definition 6057.3. It places every readback behind the monad unit row, functorial transport row, universal-property extension ledger, and handoff row before the terminal seal is exposed.

Apply triangle-ledger exactness, Definition 6057.4. The ledger keeps the unit leg, extension leg, readback leg, and Real seal leg in one finite route.

Finally replay H, C, P, N. Transport, continuation replay, provenance, and local naming preserve the same displayed carrier, so no quotient completion equality, selected completed point, host equality, countable-choice schedule, ambient completeness theorem, or detached adjunction can be produced by the readback. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRealCompletionAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRealCompletionAdjoint}^{\uparrow}$ packet is generated from a regular completion monad row, a Cauchy completion functor row, a completion universal-property row, a RegSeqRat readback, a Real seal, unit-extension handoff, triangle-ledger exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRealCompletionAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6057.1, Definition 6057.2, Definition 6057.3, and Definition 6057.4 seed the non-quotient completion-adjoint packet over the monad, functor, universal-property, readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRealCompletionAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completed limits, host real equality, countable choice, ambient Real completeness, Lean verification, bridge checking, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, unit-extension handoff, triangle-ledger exactness, readback nonescape, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRealCompletionAdjointUp

Chapter 6058

A Concrete Naming Certificate for $\mathsf{AbelTheorem}^{\uparrow}$

Following Abel and Hardy, the $\mathsf{AbelTheorem}^{\uparrow}$ packet records the finite radial-limit route for a real series whose Cesaro-facing summation data is already displayed. It is a Real-series consumer of chapter 1763, chapter 1765, chapter 301, and chapter 38: the packet names the finite handoff from summation rows through dyadic radial parameters and regular rational readback to a Real seal, without importing a host power-series limit or ambient completeness theorem.

Definition 6058.1 (Abel theorem carrier)**.** *An* $\mathsf{AbelTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (A, H, C, D, W, R, E, T, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelSummation}^{\uparrow}$ *row, carrying the finite radial polynomial-window reads used by the Abel summation route.*
2. *H is the* $\mathsf{HardyCesaroMean}^{\uparrow}$ *or* $\mathsf{CesaroConvergence}^{\uparrow}$ *row, carrying the Cesaro-facing mean data whose displayed convergence surface is consumed by the Abel route.*
3. *C records the componentwise* hsame *transport for the Abel row, Cesaro row, dyadic radial row, finite stream window, regular rational readback, Real seal, replay row, provenance row, and local naming row.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radial-parameter ledger, including dyadic radii, one-sided approach windows, and tolerance comparisons for parameters below the terminal boundary.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the dyadic radial ledger.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the Abel and Cesaro data after D and W have fixed the current tolerance window.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the Abel, Cesaro, dyadic, stream, and regular rational rows have been displayed.*
8. *T records the displayed* Cont *replay from a radial tolerance request through A, H, D, W, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{AbelTheorem}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{CesaroConvergence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, H, C, D, W, R, E, T, P, N$. It has no coordinate for a host Abel limit, selected real power-series value, quotient stream equality, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.AbelTheoremUp.AbelTheoremTasteGate single carrier alignment`

Theorem 6058.2 (Abel theorem NameCert obligations)**.** *For every accepted* $\mathsf{AbelTheorem}^{\uparrow}$ *carrier*

$$L = (A, H, C, D, W, R, E, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *Abel summation admission through A, requiring every radial polynomial-window read to pass through the displayed* $\mathsf{AbelSummation}^{\uparrow}$ *packet;*
2. *Cesaro-source admission through H, requiring the Cesaro-facing mean or convergence row to be visible before the Abel route is read as a radial limit certificate;*
3. *dyadic radial admission through D and W, requiring radial parameters and finite observation windows to be read from* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*;*
4. *regular rational readback through R, requiring the radial-window observations to be repacked as a* $\mathsf{RegSeqRat}^{\uparrow}$ *row before the Real seal is visible;*
5. *terminal Real-seal admission through E, requiring every Real-facing consumer to pass through A, H, D, W, R before reading the displayed* $\mathsf{Real}^{\uparrow}$ *boundary;*
6. *componentwise transport, replay, provenance, and local naming through C, T, P, N, with no extra host limit, quotient stream equality, countable choice, trichotomy, host equality, or ambient completeness row.*

No Abel theorem naming obligation is stored outside these rows.

Proof. Project the carrier of Definition 6058.1. The mathematical source rows are A, H, D, W, R, E, while C, T, P, N only transport, replay, source, and name that displayed route.

The Abel summation row supplies the radial polynomial-window data. The Cesaro-facing row supplies the mean or convergence surface already displayed by the sibling chapters, so the route consumes a finite summation certificate rather than a host series limit.

The dyadic radial ledger D and the stream window W determine the visible precision request. The readback row R repacks exactly that finite observation as regular rational data, and the Real seal E is downstream of the same route.

Since the carrier has no coordinate for a selected radial limit, quotient stream equality, choice, trichotomy, host equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6058.3 (Abel theorem radial Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{AbelTheorem}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{AbelSummation}^{\uparrow} \longmapsto \mathsf{HardyCesaroMean}^{\uparrow}/\mathsf{CesaroConvergence}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRa}$$

Thus the radial-limit claim is read as a displayed Real-name boundary generated from Abel summation data, Cesaro-facing source data, dyadic radial tolerances, finite stream windows, and regular rational readback. It is not a claim of host Abel convergence, selected power-series value, quotient stream equality, countable choice, trichotomy, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the Abel and Cesaro rows A and H. They are the only carrier rows that name the summation-side data, so a Real-facing consumer begins with the finite sibling packets from chapter 1763 and chapter 1765.

Next read the dyadic radial ledger D and stream window W. These rows bind the radial tolerance request to the same finite dyadic and observation surfaces used by the Real-name route.

Apply Definition 6058.2. The regular rational row R is required before any Real seal is visible, and the terminal row E is exposed only after A, H, D, W, R have been read.

The structural rows C, T, P, N preserve and name this route. Because the excluded host and quotient data are absent from the carrier, every Real-facing consumer remains inside the displayed radial Real-seal path. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbelTheorem}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6058.1, Definition 6058.2, and Definition 6058.3 seed the finite Abel theorem route from Abel summation and Cesaro-facing data through dyadic radial windows, regular rational readback, and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelTheorem}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.AbelTheoremUp.AbelTheoremTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close host Abel convergence, selected power-series values, quotient stream equality, countable choice, trichotomy, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked Abel summation admission, Cesaro-source admission, dyadic radial admission, regular rational readback, terminal Real-seal admission, transport, replay, provenance, and local naming for BEDC.Derived.AbelTheoremUp.

Constructive story: An $\mathsf{AbelTheorem}^{\uparrow}$ packet is generated from Abel summation data, Hardy-Cesaro or Cesaro-convergence source data, dyadic radial tolerances, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6059

A Concrete Naming Certificate for $\mathsf{LawlikeCauchyReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LawlikeCauchyReal}^{\uparrow}$ packet presents a real number by a lawlike regular Cauchy rational name whose request schedule is visible before any quotient, selected limit, countable-choice principle, or ambient completeness theorem is read. It sits on the lawlike generation discipline of chapter 3790, the finite window surface of chapter 126, the dyadic tolerance rows of chapter 291, the regular rational readback of chapter 301, the Cauchy-modulus interface of chapter 292, and the terminal real-name seal of chapter 38. The packet records a non-quotient, choice-free Real-and-Completion route; it does not assert that arbitrary Cauchy data carries a selected Real limit.

Definition 6059.1 (LawlikeCauchyReal carrier)**.** *A* $\mathsf{LawlikeCauchyReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (G, S, D, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{LawlikeSequence}^{\uparrow}$ *generation row, recording the lawlike request procedure that produces the rational observations.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the displayed request procedure.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic error budgets, endpoint comparisons, and late-window bounds.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the lawlike stream windows and dyadic tolerance ledger.*
5. *M is the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *row naming the displayed modulus schedule or convergence budget consumed by the regular readback.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after G, S, D, Q, M have all been exposed.*
7. *H records componentwise* hsame *transport for the lawlike generator, stream window, dyadic ledger, regular readback, modulus row, Real seal, replay row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from a lawlike Cauchy real request through G, S, D, Q, M, E.*
9. *P is the* Pkg *provenance over* $\mathsf{LawlikeCauchyReal}^{\uparrow}$, $\mathsf{LawlikeSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{LawlikeCauchyReal}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, S, D, Q, M, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable-choice schedules, host real equality, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credentials.*

Lean4 ✓ `BEDC.Derived.LawlikeCauchyRealUp.LawlikeCauchyRealTasteGate single carrier alignment`

Theorem 6059.2 (LawlikeCauchyReal NameCert obligations)**.** *For every accepted* $\mathsf{LawlikeCauchyReal}^{\uparrow}$ *carrier*

$$L = (G, S, D, Q, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LawlikeCauchyReal}^{\uparrow}}$ *surface consists exactly of carrier admission, lawlike generator exposure, finite stream-window exposure, dyadic tolerance exactness, regular rational readback, displayed Cauchy-modulus handoff, Real-seal routing, componentwise transport, displayed replay, provenance, local naming, and non-escape from the displayed rows.*

Proof. Project the carrier rows of Definition 6059.1. The accepted packet exposes the lawlike generator G, the stream window S, the dyadic ledger D, the regular readback Q, the modulus row M, the Real seal E, and the structural rows H, C, P, N.

The source rows are then read in order. The lawlike procedure of chapter 3790 supplies the request discipline, chapter 126 opens only the finite windows requested by that procedure, and chapter 291 supplies the tolerance ledger.

The regular and completion-facing rows close the route without adding a selected limit. The readback of chapter 301 packages the finite rational data, chapter 292 supplies the displayed budget row, and chapter 38 receives the terminal seal only through E.

Finally H and C transport and replay the same coordinates, while P and N record provenance and local naming. Since no additional coordinate is present, the certificate surface exports none of the excluded quotient, choice, completeness, host-equality, verification, or bridge data. □

Theorem 6059.3 (LawlikeCauchyReal modulus handoff)**.** *Every completion-facing request for an accepted* $\mathsf{LawlikeCauchyReal}^{\uparrow}$ *carrier reads the displayed Cauchy modulus row M only after the lawlike generator G, finite stream window S, dyadic tolerance ledger D, and regular rational readback Q have been exposed. Thus the handoff supplies a finite convergence budget for the named lawlike route, not a selector for arbitrary Cauchy data.*

Proof. Start with an accepted carrier $L = (G, S, D, Q, M, E, H, C, P, N)$. By Definition 6059.1, the only Cauchy-modulus coordinate is M, and it is downstream of the lawlike source, finite window, tolerance ledger, and regular readback rows.

The sibling routes determine the inputs to M. The generator row G supplies a lawlike request discipline, the window row S opens the requested observations, the dyadic row D supplies the tolerance comparisons, and the regular row Q packages those observations as the readback expected by the modulus interface.

Therefore a completion-facing consumer cannot read M as ambient completeness or as a selected limit. The only accepted handoff is the displayed route through G, S, D, Q, M, transported by H, replayed by C, and named by P, N. □

Theorem 6059.4 (LawlikeCauchyReal Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{LawlikeCauchyReal}^{\uparrow}$ *carrier factors through the finite route*

$$G, S, D, Q, M \longmapsto E$$

before the terminal Real seal is visible. Thus a consumer receives lawlike generation, finite StreamName windows, dyadic tolerance rows, RegSeqRat readback, and the CauchyModulusSpace budget before it may consume the $\mathsf{Real}^{\uparrow}$ *row.*

Proof. Apply Definition 6059.2. The Real-facing coordinate is E, and the obligation surface places it after G, S, D, Q, M.

The route cannot bypass the source stage because G is the only row naming the lawlike request procedure. It cannot bypass the window or tolerance stage because S and D are the only rows containing finite observations and dyadic error bounds.

The regular readback and modulus handoff are also forced. The row Q is the sole $\mathsf{RegSeqRat}^{\uparrow}$ readback available to the packet, and M is the sole displayed $\mathsf{CauchyModulusSpace}^{\uparrow}$ budget consumed before the Real seal.

Transport, replay, provenance, and local naming preserve the same route. Since H, C, P, N add no quotient, selected-limit, countable-choice, host-equality, or ambient-completeness coordinate, every Real-facing read reaches E only through the displayed route. □

Theorem 6059.5 (LawlikeCauchyReal non-escape)**.** *An accepted* $\mathsf{LawlikeCauchyReal}^{\uparrow}$ *carrier exports no quotient stream equality, selected limit, countable-choice schedule, trichotomy principle, open-cover compactness proof, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, host real equality, Lean verification, or bridge credential. Every exported read is confined to* $G, S, D, Q, M, E, H, C, P, N$.

Proof. Use the carrier definition to enumerate the available coordinates. They are exactly the lawlike generator, stream window, dyadic ledger, regular readback, modulus row, Real seal, transport, replay, provenance, and local naming rows.

The obligation theorem identifies the NameCert surface with those same rows. Any proposed exported read must therefore be one of the displayed coordinates or a structural transport of them through H and C.

None of the displayed coordinates stores quotient stream equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient Real completeness, host real equality, Lean verification, or bridge data. Hence no such datum can escape from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LawlikeCauchyReal}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LawlikeCauchyReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6059.1, Definition 6059.2, Definition 6059.3, Definition 6059.4, and Definition 6059.5 seed the non-quotient lawlike Cauchy real-name packet over displayed $\mathsf{LawlikeSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulusSpace}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LawlikeCauchyReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lawlike generator exposure, stream-window exposure, dyadic tolerance exactness, regular rational readback, displayed Cauchy-modulus handoff, Real-seal routing, transport stability, replay stability, provenance, local naming, and non-escape for BEDC.Derived.LawlikeCauchyRealUp.

Constructive story: A $\mathsf{LawlikeCauchyReal}^{\uparrow}$ packet is generated from a $\mathsf{LawlikeSequence}^{\uparrow}$ request procedure, finite $\mathsf{StreamName}^{\uparrow}$ windows, a $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, a $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a $\mathsf{CauchyModulusSpace}^{\uparrow}$ budget row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6060

A Concrete Naming Certificate for $\mathsf{BoundedInverseTheorem}^{\uparrow}$

Following Banach and Rudin-style functional analysis, the $\mathsf{BoundedInverseTheorem}^{\uparrow}$ packet records the finite BEDC certificate for reading a bijective bounded linear map between Banach rows as a displayed inverse-bound route. It is grounded in the open-ball image handoff of chapter 1538, the graph-read boundary of chapter 1537, the Banach source and target rows of chapter 90, the bounded linear map surface of chapter 48, and the terminal real-bound seal of chapter 38. The packet exposes only finite rows: Banach endpoints, a bounded linear map row, bijectivity ledgers, an open-image witness, an inverse-bound ledger, graph compatibility, transport, replay, provenance, and local naming.

Definition 6060.1 (Bounded inverse theorem carrier)**.** *A* $\mathsf{BoundedInverseTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, Y, T, J, O, B, G, H, C, P, N)$$

with the following visible rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, and completion observations used to read source windows.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, including the displayed norm, metric, regular-Cauchy, and completion observations used to read target windows.*
3. *T is the bounded* $\mathsf{LinMap}^{\uparrow}$ *row from X to Y, carrying only the finite action, linearity, and bound data displayed by the map certificate.*
4. *J is the bijectivity ledger, with injective comparison rows and surjective preimage rows over the same displayed X, Y, T data.*
5. *O is the* $\mathsf{OpenMappingTheorem}^{\uparrow}$ *open-image handoff row, recording the finite ball-image and source-window data needed before an inverse-bound request can be public.*
6. *B is the inverse-bound ledger, recording target tolerance requests, source preimage windows, norm comparisons, and the real-bound seal read through* $\mathsf{Real}^{\uparrow}$*.*
7. *G is the* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *compatibility row, recording that graph-facing consumers read the inverse graph only through displayed graph rows and not through host graph equality.*
8. *H records componentwise* hsame *transport for endpoint, map, bijectivity, open-image, inverse-bound, graph, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from Banach admission through bounded map inspection, bijectivity lookup, open-image handoff, inverse-bound readback, graph compatibility, and real-bound sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{BoundedInverseTheorem}^{\uparrow}$, $\mathsf{OpenMappingTheorem}^{\uparrow}$, $\mathsf{ClosedGraphTheorem}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedInverseTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host inverse function, quotient operator equality, choice-selected inverse values, a non-displayed open mapping principle, a non-displayed closed graph principle, or ambient completeness beyond the Banach rows X and Y.

Lean4 ✓ `BEDC.Derived.BoundedInverseTheoremUp.TasteGate.BoundedInverseTheoremTasteGate single carrier alignment`

Theorem 6060.2 (Bounded inverse theorem NameCert obligations)**.** *For an accepted* $\mathsf{BoundedInverseTheorem}^{\uparrow}$ *carrier $Q = (X, Y, T, J, O, B, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedInverseTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the Banach source row, Banach target row, bounded linear map row, bijectivity ledger, open-image handoff, inverse-bound ledger, graph compatibility row, componentwise transport, continuation replay, provenance, and local naming;*

2. *endpoint agreement, requiring T, J, O, B, G to read the same displayed source and target Banach rows before any inverse-bound request is admitted;*

3. *bijective map exposure, requiring inverse-facing consumers to read injective comparison and surjective preimage data through the bounded* $\mathsf{LinMap}^{\uparrow}$ *row T and the finite ledger J;*

4. *open-mapping route, requiring every inverse norm-bound estimate to pass through the* $\mathsf{OpenMappingTheorem}^{\uparrow}$ *row O before the inverse-bound ledger B is exposed;*

5. *closed-graph route, requiring graph-facing consumers to route the displayed inverse graph through the* $\mathsf{ClosedGraphTheorem}^{\uparrow}$ *row G and the same X, Y, T, J, O, B fields;*

6. *real-bound exactness, requiring every numeric inverse-bound read to pass through the displayed* $\mathsf{Real}^{\uparrow}$ *seal in B rather than through a host real or quotient operator norm;*

7. *structural replay, requiring* hsame, Cont, Pkg, *and local* NameCert *rows to preserve the same bounded-inverse route;*

8. *ledger non-escape, excluding host inverse functions, quotient operator equality, choice-selected inverse values, non-displayed open-mapping or closed-graph principles, and ambient completeness beyond X and Y.*

No bounded inverse theorem obligation is stored outside these rows.

Proof. Project the carrier of Definition 6060.1. The endpoint data are X and Y, the bounded map is T, the bijectivity ledger is J, the open-image handoff is O, the inverse-bound ledger is B, and the graph compatibility row is G.

Endpoint agreement is rowwise. Since T, J, O, B, G are all carried over the same displayed Banach endpoints, an inverse-facing consumer cannot mix a map row from one source with an open-image witness or graph row from another target.

The bounded-inverse estimate is read through a finite handoff chain. The ledger J first supplies injective comparison and surjective preimage data for T. The open-mapping row O then supplies

the displayed open-ball image route of chapter 1538. Only after those rows are visible may B expose the source-window request, norm comparison, and $\mathsf{Real}^{\uparrow}$ bound seal.

Graph compatibility is separate from the inverse-bound estimate. The row G cites the closed-graph boundary of chapter 1537, so a graph-facing consumer reads the inverse graph only as displayed graph data and cannot replace the route by host graph equality.

Finally H, C, P, N transport, replay, record provenance for, and name exactly this finite route. Since the carrier fields are exhausted by $X, Y, T, J, O, B, G, H, C, P, N$, the excluded host inverse, quotient operator, choice-selected, non-displayed theorem, and ambient-completeness principles have no row in the packet. □

Theorem 6060.3 (Bounded inverse theorem open-mapping route)**.** *Let $Q = (X, Y, T, J, O, B, G, H, C, P, N)$ be an accepted* $\mathsf{BoundedInverseTheorem}^{\uparrow}$ *carrier. Every inverse-bound read exported by Q factors through the displayed route*

$$X, Y, T, J, O, B, G, H, C, P, N.$$

In particular, the consumer must pass through the Banach rows of chapter 90, the bounded $\mathsf{LinMap}^{\uparrow}$ *row of chapter 48, the open-image handoff of chapter 1538, the graph boundary of chapter 1537, and the real-bound seal of chapter 38 before any inverse-bound surface is public.*

Proof. Start with the endpoint and map rows. A consumer reads source and target norm windows only through X and Y, and it reads the bounded map only through T.

Next inspect the bijectivity and open-image rows. The ledger J supplies the displayed injective and surjective data, while O supplies the finite open-ball image handoff. No inverse-bound conclusion is available before both rows have been read.

The inverse-bound and graph rows then determine the public surface. The row B records the target tolerance requests, source preimage windows, norm comparison, and $\mathsf{Real}^{\uparrow}$ bound seal. The row G keeps the inverse graph visible as a closed-graph route rather than as host graph equality.

Transport by H, replay by C, and packaging by P, N preserve exactly the same route. Hence the exported inverse-bound surface is the displayed finite chain and not a quotient operator space, selected inverse map, host real norm, or ambient completeness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedInverseTheorem}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedInverseTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6060.1, Definition 6060.2, and Definition 6060.3 seed the finite inverse-bound certificate over displayed Banach endpoints, bounded LinearMap rows, bijectivity ledgers, open-image handoffs, graph compatibility rows, Real bound seals, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedInverseTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse functions, quotient operator equality, choice-selected inverse values, non-displayed open-mapping or closed-graph principles, Lean verification, bridge checking, or ambient completeness beyond the displayed Banach rows.

Upgrade path: Obligation closure requires checked carrier admission, endpoint agreement, bijective map exposure, open-mapping routing, closed-graph routing, real-bound exactness, structural replay, and ledger non-escape for BEDC.Derived.BoundedInverseTheoremUp.

Constructive story: The construction starts from displayed Banach endpoints, reads a bounded LinearMap row with bijectivity data, routes inverse-bound requests through OpenMappingTheorem and ClosedGraphTheorem rows, seals the numeric bound through Real data, and exports only the local hsame and Cont route.

Chapter 6061

A Concrete Naming Certificate for $\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}$

Following Mac Lane and Bishop–Bridges, the $\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}$ packet records the finite triangle where the regular completion monad, the Kleisli consumer, the completion functor, and the universal-property readback meet the Bishop real seal. It sits after the adjoint packet of chapter 6057, the regular completion monad of chapter 2136, and the Kleisli boundary of chapter 1845. The packet names a non-quotient L10 completion route; it does not select completed points, compare quotient streams, import countable choice, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6061.1 (Bishop completion Kleisli triangle carrier)**.** *A* $\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, K, F, U, R, E, J, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$ *row, recording the unit and bind schedule for regular completion names.*
2. *K is the* $\mathsf{CauchyCompletionKleisli}^{\uparrow}$ *row, recording the consumer boundary for unit, bind, and reassociated bind requests.*
3. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *row, recording functorial source-to-completion transport.*
4. *U is the* $\mathsf{CompletionUniversalProperty}^{\uparrow}$ *row, recording extension and uniqueness only as displayed finite ledger data.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the regular source windows are inspected.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, K, F, U, R have been displayed.*
7. *J is the triangle ledger, matching the monad unit, Kleisli bind, functor handoff, universal-property extension, readback, and Real seal legs.*
8. *H records componentwise* hsame *transport for the monad, Kleisli, functor, universal-property, readback, seal, triangle-ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay of the route*

$$M, K, F, U \longmapsto R \longmapsto E \longmapsto J, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}$, $\mathsf{RegularCauchyCompletionMonad}^{\uparrow}$, $\mathsf{CauchyCompletionKleisli}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}}$ *row.*

The accepted carrier is exactly the displayed packet T. It has no coordinate for quotient completion equality, selected limits, host real equality, countable-choice schedules, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a detached Kleisli adjunction theorem.*

Theorem 6061.2 (Bishop completion Kleisli triangle NameCert obligations)**.** *For every accepted carrier $T = (M, K, F, U, R, E, J, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}}$ *obligations are exhausted by carrier admission for the displayed monad, Kleisli, functor, universal-property, readback, seal, triangle-ledger, transport, replay, provenance, and local naming rows. Every completion-facing consumer must read the route in the order*

$$M, K, F, U \longmapsto R \longmapsto E \longmapsto J, H, C, P, N.$$

No local obligation may call on quotient completion equality, selected limits, host real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or an external adjunction principle.*

Proof. Project the carrier of Definition 6061.1. The only rows visible to the local classifier are $M, K, F, U, R, E, J, H, C, P, N$.

Read the completion route by route ordering. The monad row M supplies unit and bind schedule data, the Kleisli row K fixes the consumer boundary, the functor row F transports source data to completion-facing names, and the universal-property row U supplies the displayed extension ledger. Only after those rows are present may the regular readback R and terminal seal E be consumed.

The triangle row is structural over that same route. The row J records compatibility of the monad, Kleisli, functor, universal-property, readback, and seal legs, while H transports displayed coordinates, C replays the permitted reads, and P, N record provenance and local naming.

Since the accepted packet has no coordinate for quotient completion equality, selected limits, host real equality, countable choice, ambient completeness, or a detached adjunction theorem, no NameCert obligation can export one. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}$ packet is generated from a regular completion monad row, a Cauchy Kleisli consumer row, a Cauchy completion functor row, a completion universal-property row, a RegSeqRat readback, a Real seal, a triangle ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}}$, seed)

Scope closed: Definition 6061.1 and Definition 6061.2 seed the non-quotient completion triangle over the monad, Kleisli, functor, universal-property, readback, and Real seal rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BishopCompletionKleisliTriangle}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected completed limits, host real equality, countable choice, ambient Real completeness, Lean verification, bridge checking, or a host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, triangle-ledger exactness, readback nonescape, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionKleisliTriangleUp.

Chapter 6062

A Concrete Naming Certificate for $\mathsf{RiemannUniformMesh}^{\uparrow}$

Following Riemann and Bishop–Bridges constructive analysis, the $\mathsf{RiemannUniformMesh}^{\uparrow}$ packet records the finite equal-width dyadic mesh rows used before a Riemann-sum or completion-facing consumer reads an integral approximation. It sits after the dyadic endpoint discipline of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the tagged finite-sum consumer of chapter 1615, and the displayed finite partition mesh of chapter 3496. It also uses the dyadic selector route of chapter 2254 to keep mesh depth finite and visible.

Definition 6062.1 (Riemann uniform mesh NameCert carrier)**.** *A* $\mathsf{RiemannUniformMesh}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (P, D, W, R, S, E, H, C, G, N)$$

with the following displayed rows:

1. *P is the finite partition-mesh row, exposing interval endpoints, cell order, adjacency, and common-refinement evidence through the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *surface.*
2. *D is the dyadic mesh scale row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and equal-width dyadic scale used by every cell.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the displayed mesh scale.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same window and dyadic tolerance.*
5. *S is the* $\mathsf{RiemannSum}^{\uparrow}$ *consumer row, reading tag values, cell widths, and the finite sum only after P, D, W, R are visible.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only as the terminal handoff for the displayed finite approximation.*
7. *H records componentwise* hsame *transport for partition mesh, dyadic scale, finite window, regular readback, Riemann-sum handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the partition row through the dyadic scale, finite window, regular readback, Riemann-sum row, and Real seal.*
9. *G is the* Pkg *provenance over* $\mathsf{RiemannUniformMesh}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DyadicMesh}^{\uparrow}$, $\mathsf{CofinalDyadicMesh}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RiemannSum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RiemannUniformMesh}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $P, D, W, R, S, E, H, C, G, N$. It contains no host interval continuum, no quotient completion, no selected limiting mesh family, no unrestricted tag-choice principle, and no ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RiemannUniformMeshUp.RiemannUniformMeshTasteGate single carrier alignment`

Theorem 6062.2 (Riemann uniform mesh NameCert obligations)**.** *For every accepted* $\mathsf{RiemannUniformMesh}^{\uparrow}$ *carrier $U = (P, D, W, R, S, E, H, C, G, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannUniformMesh}^{\uparrow}}$ *surface consists exactly of carrier admission; finite partition-mesh exposure through P; dyadic mesh exactness through D together with the dyadic mesh siblings; Riemann-sum handoff through S after W and R; Real-seal non-escape through E; componentwise transport through H; displayed replay through C; package provenance through G; and local naming through N.*

Proof. Project the carrier of Definition 6062.1. The partition row P supplies the only finite mesh carrier, and the dyadic row D supplies the equal-width scale and endpoint ledger used by each cell.

The finite readback rows then determine the consumer window. The row W opens the $\mathsf{StreamName}^{\uparrow}$ window at the displayed scale, while R records the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to that same dyadic budget.

The Riemann-sum handoff is forced through S. A consumer may read tag values, cell widths, and the finite sum only after P, D, W, R are present, so the handoff is the finite tagged-sum route of chapter 1615, not a limiting integration claim.

Finally the Real seal E is terminal. Transport, replay, provenance, and local naming are exactly the rows H, C, G, N, and the carrier has no coordinate for host intervals, quotient completions, selected limiting mesh families, unrestricted tag choice, or ambient completeness. Thus the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: If the dyadic scale row D is removed, a Riemann-sum consumer can still read a finite partition row but can no longer force equal-width dyadic exactness before the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows are opened.

Independence witness: finite-partition-mesh, dyadicratcore, streamname, regseqrat, riemannsum, real, cofinal-dyadic-mesh: no listed sibling is bijective to $\mathsf{RiemannUniformMesh}^{\uparrow}$ because this packet jointly carries the finite partition mesh, equal dyadic scale, finite observation window, regular rational readback, tagged-sum handoff, terminal Real seal, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannUniformMesh}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RiemannUniformMesh}^{\uparrow}$ packet is generated from finite partition mesh rows, a dyadic mesh scale, StreamName finite windows, RegSeqRat readback, a RiemannSum consumer row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RiemannUniformMesh}^{\uparrow}}$, seed)

Scope closed: Definition 6062.1 and Definition 6062.2 seed the finite equal-width dyadic mesh packet consumed by Riemann-sum, Real, and Completion routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RiemannUniformMesh}^{\uparrow}}$, unformalized)

Bridge status: none

Origin: human

Not claimed: This chapter does not close Riemann integrability, limiting mesh convergence, host interval continua, quotient completion, selected tag families, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, Riemann-sum handoff, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.RiemannUniformMeshUp.

Chapter 6063

A Concrete Naming Certificate for FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$

The FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$ packet realizes the confluence audit map of chapter 9378 as a finite residual-window diamond envelope. It sits beside the residual-substitution boundary of chapter 2181, the parallel-diamond frontier of chapter 4238, the residual-join determinacy packet of chapter 5970, and the closed parallel-diamond premise ledger of chapter 2325. The packet records bounded peak, residual-window, candidate-join, premise-retention, and TasteGate-readiness rows without claiming residual-substitution compatibility, a premise-free parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient conversion equality, or kernel consistency.

Falsifiable prediction: A FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$ read is rejected whenever its residual-window row is present but either the residual-substitution boundary row or the closed parallel-diamond premise ledger row is absent.

Independence witness: metacic-residual-substitution-boundary, metacic-parallel-diamond-frontier, metacic-residual-join-determinacy, closed-parallel-diamond-premise-ledger: FiniteWindowMetaCICDiamondEnvelope is not bijective to any listed sibling because it jointly carries a bounded peak window, residual-window budget, candidate-join determinacy, parallel-diamond frontier handoff, closed-premise ledger, obstruction-retention row, TasteGate-readiness row, transport, replay, provenance, and local NameCert data.

Definition 6063.1 (Finite-window MetaCIC diamond-envelope carrier)**.** *A* FiniteWindowMetaCICDiamondEnvelope *carrier is a finite* BHist *packet*

$$E = (W, R, J, F, L, O, G, H, C, P, N)$$

with the following displayed rows:

1. *W is the bounded MetaCIC peak-window row, naming the source term, two parallel observations, and the finite residual radius.*
2. *R is the* MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *row, kept visible as the residual-substitution premise boundary for the same peak.*
3. *J is the* MetaCICResidualJoinDeterminacy$^{\uparrow}$ *row, recording the budgeted local-join candidate and determinacy readback for the residual window.*

4. *F is the* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *row, exposing the conditional parallel-diamond frontier consumed by the envelope.*
5. *L is the* $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$ *row, recording the closed-premise ledger that any closed-star handoff must cite.*
6. *O is the obstruction-retention row, stating that the envelope still displays the missing residual-substitution premise whenever the conditional frontier is read.*
7. *G is the* $\mathsf{TasteGate}$ *readiness row for later* $\mathsf{Nontrivial}$ *and* $\mathsf{FieldFaithful}$ *instances over the displayed envelope.*
8. *H records componentwise* hsame *transport for peak windows, residual-substitution boundaries, residual joins, frontier rows, closed-premise ledgers, obstruction retention, TasteGate readiness, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a confluence-audit query through $W, R, J, F, L, O, G, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$, $\mathsf{MetaCICResidualJoinDeterminacy}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{ClosedParallelDiamondPremiseLedger}^{\uparrow}$, BHist, hsame, Cont, $\mathsf{TasteGate}$, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $W, R, J, F, L, O, G, H, C, P, N$. It contains no coordinate for full residual-substitution compatibility, a premise-free parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient conversion equality, selected joins, or kernel consistency.

Theorem 6063.2 (Finite-window MetaCIC diamond-envelope NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}$ *carrier*

$$E = (W, R, J, F, L, O, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}}$ *surface consists exactly of bounded peak-window exposure through W, residual-substitution boundary retention through R, residual-join determinacy through J, parallel-diamond frontier handoff through F, closed-premise ledger exposure through L, obstruction retention through O, TasteGate readiness through G, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6063.1. Rows W, R, J, F, L, O are the confluence-facing envelope data, while G, H, C, P, N are the structural certificate data.

The residual-window read starts with W and R. The peak, radius, and two outgoing observations are bounded by W, and the residual-substitution boundary row R keeps the missing compatibility premise visible before any diamond-facing handoff can be consumed.

The join and frontier rows are then read through J, F, L. The residual-join determinacy row records a budgeted candidate join, the frontier row exposes only the conditional parallel-diamond route, and the closed-premise ledger records the closed handoff that such a route would consume.

The obstruction and structural rows finish the surface. Row O prevents the envelope from being read as a solved residual-substitution theorem, and G, H, C, P, N reserve the TasteGate instance surface, transport the rows, replay the audit query, source the packet, and name it. Since no carrier coordinate contains unrestricted confluence, subject reduction, evaluator equality, quotient conversion equality, selected joins, or kernel consistency, the listed rows exhaust the local NameCert surface. □

Theorem 6063.3 (Finite-window MetaCIC diamond-envelope handoff)**.** *Every confluence-audit consumer of an accepted* FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$ *carrier factors through*

$$W, R, J, F, L, O, G, H, C, P, N.$$

The consumer may inspect the bounded peak window, residual-substitution boundary, residual-join determinacy row, parallel-diamond frontier, closed-premise ledger, obstruction-retention row, TasteGate readiness, transport, replay, provenance, and local name. It cannot bypass this route to obtain full residual-substitution compatibility, a premise-free parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient conversion equality, selected joins, or kernel consistency.

Proof. Start with a confluence-audit query. The only entrance to the envelope is the finite peak-window row W, and the only residual-substitution premise surface is R.

The local join route is constrained by J. It records a budgeted candidate join for the displayed residual window, so the consumer may not detach the join row into a global diamond theorem.

The frontier and closed-premise route is constrained by F and L. These rows expose the conditional parallel-diamond frontier and the closed parallel-diamond premise ledger, but they do not supply the missing residual-substitution compatibility proof.

Finally O, G, H, C, P, N preserve the same boundary under obstruction retention, TasteGate readiness, transport, replay, provenance, and local naming. Every confluence-audit consumer therefore remains inside the finite envelope route. □

Theorem 6063.4 (Finite-window MetaCIC diamond-envelope subject slice)**.** *Any subject-reduction-facing consumer of an accepted* FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$ *packet may inspect only the bounded peak window W, residual-substitution boundary R, residual-join determinacy row J, parallel-diamond frontier F, closed-premise ledger L, and obstruction-retention row O before reaching TasteGate, transport, replay, provenance, and local naming rows. It cannot turn the envelope into subject reduction, evaluator equality, quotient conversion equality, or kernel consistency.*

Proof. Project the carrier of Definition 6063.1. The subject-facing entrance is still the finite envelope tuple, so the only confluence data visible before the structural rows are W, R, J, F, L, O.

Apply Definition 6063.2. The obligation surface assigns bounded peak exposure to W, residual-substitution boundary retention to R, residual-join determinacy to J, parallel-diamond frontier handoff to F, closed-premise ledger exposure to L, and obstruction retention to O.

Finally use Definition 6063.3. The handoff already blocks full residual-substitution compatibility, premise-free diamonds, subject reduction, evaluator equality, quotient conversion equality, selected joins, and kernel consistency. The present statement records the subject-reduction slice of that boundary; it is not itself a confluence or subject-reduction theorem. □

Theorem 6063.5 (Finite-window MetaCIC diamond-envelope TasteGate scope)**.** *The later Lean side must provide* Nontrivial *and* FieldFaithful TasteGate *instances for* FiniteWindowMetaCICDiamondEnvelope$^{\uparrow}$ *at the displayed finite carrier surface*

$$W, R, J, F, L, O, G, H, C, P, N.$$

Those instances must certify that the packet is neither empty nor field-forgetting: every accepted carrier visibly retains the bounded peak window, residual-substitution boundary, residual-join determinacy row, parallel-diamond frontier, closed-premise ledger, obstruction-retention row, transport, replay, provenance, and local NameCert data.

Proof. The readiness row G in Definition 6063.1 is tied to the whole envelope, not to the residual-join row alone.

Nontriviality must exhibit an accepted packet with all displayed rows present. FieldFaithfulness must show that deleting any of $W, R, J, F, L, O, H, C, P, N$ changes the readable carrier surface. Thus the TasteGate burden remains exactly at the finite residual-window diamond-envelope boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}$ packet is generated from a bounded MetaCIC peak-window row, residual-substitution boundary row, residual-join determinacy row, parallel-diamond frontier row, closed parallel-diamond premise ledger row, obstruction-retention row, Nontrivial and FieldFaithful TasteGate readiness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6063.1, Definition 6063.2, Definition 6063.3, and Definition 6063.5 seed the finite residual-window diamond envelope for MetaCIC confluence-audit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowMetaCICDiamondEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close residual-substitution compatibility, a premise-free parallel diamond, arbitrary closed Church–Rosser, subject reduction, evaluator equality, quotient conversion equality, selected joins, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, peak-window exposure, residual-substitution boundary retention, residual-join determinacy, parallel-diamond frontier handoff, closed-premise ledger exposure, obstruction retention, TasteGate nontriviality, FieldFaithful coverage, transport, replay, provenance, and local naming for BEDC.Derived.FiniteWindowMetaCICDiamondEnvelopeUp.

Chapter 6064

A Concrete Naming Certificate for $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy convergence, the $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ packet records the finite readback by which one displayed uniform Cauchy witness yields an explicit modulus row before any regular rational handoff or terminal Real seal is consumed. It sits over the Real uniform surface of chapter 1624, the explicit-modulus sequence route of chapter 1705, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet makes finite uniform modulus extraction visible as NameCert data rather than leaving it distributed across later sequence, completion, or Real-facing chapters.

Definition 6064.1 (Uniform Cauchy modulus extractor carrier)**.** *A* $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (U, S, M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *U is the uniform-Cauchy source row, naming the displayed family, tail comparison, and uniformity request from which an extraction is attempted.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, recording the finite observation windows inspected from the same source.*
3. *M is the extracted* $\mathsf{CauchyModulus}^{\uparrow}$ *row, storing the stage bound selected for each dyadic tolerance request.*
4. *W is the window-realization row, recording that the selected stage from M governs all retained stream windows in S.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to translate requested radii, slack rows, comparison budgets, and dyadic refinements.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row opened only after U, S, M, W, D have been displayed.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available downstream of the extracted modulus and regular readback.*
8. *H records componentwise* hsame *transport for the source, schedule, extracted modulus, window realization, dyadic ledger, readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a uniformity request through $U, S, M, W, D, R, E, H, P, N$.*
10. *P is the* Pkg *provenance over* UniformCauchyModulusExtractor$^{\uparrow}$, RealUniformStructure$^{\uparrow}$, EffectiveCauchySequence StreamName$^{\uparrow}$, CauchyModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformCauchyModulusExtractor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $U, S, M, W, D, R, E, H, C, P, N$. It has no coordinate for a selected host limit, quotient stream equality, ambient Real$^{\uparrow}$ *completeness, countable choice, a hidden modulus oracle, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.UniformCauchyModulusExtractorUp.UniformCauchyModulusExtractorTasteGate single carrier alignment`

Theorem 6064.2 (Uniform Cauchy modulus extractor NameCert obligations)**.** *For every accepted* UniformCauchyModulusExtractor$^{\uparrow}$ *carrier*

$$X = (U, S, M, W, D, R, E, H, C, P, N),$$

the local NameCert$_{\text{UniformCauchyModulusExtractor}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the uniform-Cauchy source, stream schedule, extracted modulus, window realization, dyadic tolerance ledger, regular rational readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *uniform-source exposure, requiring the extraction request to display the source row U and the finite stream schedule S before any modulus row is exported;*
3. *finite modulus extraction, requiring M to be read as a displayed stage ledger and not as an external selector or hidden oracle;*
4. *dyadic tolerance exactness, requiring D to account for requested radii, slack rows, comparison budgets, and tolerance conversions consumed by M and W;*
5. *RegSeqRat and Real routing, requiring R and E to remain downstream of U, S, M, W, D;*
6. *transport, replay, provenance, and local naming stability for the same displayed rows through H, C, P, N.*

No uniform-Cauchy-modulus-extractor obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.UniformCauchyModulusExtractorUp.TasteGate.UniformCauchyModulusExtractorNameCertObl`

Proof. Carrier admission is projection from Definition 6064.1. The accepted packet exposes exactly the uniform-Cauchy source U, finite schedule S, extracted modulus M, window realization W, dyadic ledger D, regular readback R, Real seal E, and structural rows H, C, P, N.

The extraction row is forced to be finite. The source and schedule rows are displayed before M, and the dyadic ledger D supplies the tolerance conversions that make the selected stages meaningful inside the packet.

The downstream route is then fixed by the EffectiveCauchySequence and Real uniform surfaces. The regular rational readback R can only consume the displayed modulus, window, and tolerance rows, and the Real seal E is visible only after that readback has been exposed.

Finally, H, C, P, N transport, replay, source, and name the same packet. Since no host selector, quotient equality, ambient completeness theorem, hidden oracle, Lean target, or bridge certificate appears among the rows, the listed obligations exhaust the local NameCert surface. □

Theorem 6064.3 (Uniform Cauchy modulus extractor readback route)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ *carrier factors through the finite route*

$$U \longmapsto S \longmapsto M \longmapsto W \longmapsto D \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

Thus the consumer receives a displayed uniform-Cauchy source, finite stream schedule, extracted modulus, window realization, dyadic tolerance ledger, regular readback, Real seal, transport, replay, provenance, and local naming before it can inspect any endpoint-facing result.

Proof. Project the carrier of Definition 6064.1. The only regular readback row is R, and the only Real seal row is E.

The source route comes first. The uniform-Cauchy request is displayed by U, the finite observations are named by S, the extracted stage ledger is M, and W records that the chosen stages govern the retained windows.

The dyadic accounting row D then supplies the tolerance budget consumed by the readback. Hence R is downstream of the source, schedule, extracted modulus, window realization, and tolerance ledger, while E is downstream of R.

The structural rows preserve this route and add no analytic coordinate. Therefore every endpoint-facing read has exactly the stated finite factorization and cannot bypass it through excluded host or completeness data. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ read that reaches a RegSeqRat or Real boundary without displaying the uniform-Cauchy source, finite StreamName schedule, extracted CauchyModulus row, window realization, DyadicRatCore tolerance ledger, transport, replay, provenance, and local naming rows refutes the packet.

Independence witness: real-uniform-structure, effective-cauchy-sequence, streamname, cauchy-modulus, dyadicratcore, regseqrat, real: $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a uniform-Cauchy source, finite schedule, extracted modulus, window realization, tolerance ledger, regular readback, terminal Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyModulusExtractor}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformCauchyModulusExtractor}^{\uparrow}$ packet is generated from a displayed uniform-Cauchy source, finite StreamName schedule, extracted CauchyModulus row, window realization, DyadicRatCore tolerance ledger, RegSeqRat readback, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyModulusExtractor}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6064.1, Definition 6064.2, and Definition 6064.3 seed the finite uniform modulus extraction route for Real and completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyModulusExtractor}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected host limits, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, countable choice, hidden modulus oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-source exposure, finite modulus extraction, dyadic tolerance exactness, RegSeqRat and Real routing, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyModulusExtractorUp.

Chapter 6065

A Concrete Naming Certificate for $\mathsf{BoundedCauchyInterval}^{\uparrow}$

Bishop–Bridges style constructive interval and Cauchy data enter this chapter through a finite packet: a closed rational interval bracket, a regular rational sequence, a finite observation schedule, and a dyadic tolerance ledger are displayed before any $\mathsf{Real}^{\uparrow}$ seal or completion-facing consumer can read an endpoint. The $\mathsf{BoundedCauchyInterval}^{\uparrow}$ packet sits over the real-name seal of chapter 38, the regular rational readback of chapter 301, the finite window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the located regular route of chapter 2376, and the finite located interval net of chapter 2385. It also consumes chapter 3834: the displayed B, S, R, D, L, F, E route is the finite interval-level instance where completion unit and bind laws preserve bracket, stream-window, regular-readback, and Real-seal rows. It supplies Real and Completion routes with a bounded interval-level Cauchy packet, not quotient streams, selected limits, choice, or ambient completeness.

Definition 6065.1 (Bounded Cauchy interval carrier)**.** *A* $\mathsf{BoundedCauchyInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (B, S, R, D, L, F, E, H, C, P, N)$$

with the following displayed rows:

1. *B is a bounded rational interval bracket, recording two rational endpoints, an order witness, and the interval membership window used by every later read.*
2. *S is the finite* $\mathsf{StreamName}^{\uparrow}$ *observation schedule for the Cauchy name inside the bracket.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic radii, endpoint slack, and bracket-preserving comparison rows.*
5. *L is the* $\mathsf{LocatedRegSeqRat}^{\uparrow}$ *locatedness row for the same regular rational readback.*
6. *F is the* $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$ *finite-net handoff row for the displayed bracket and located readback.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row read only after B, S, R, D, L, F are displayed.*
8. *H records componentwise* hsame *transport for the bracket, stream schedule, regular readback, dyadic tolerance ledger, locatedness row, finite-net handoff, and real seal.*

9. *C records displayed* Cont *replay from a bounded Cauchy interval request through B, S, R, D, L, F, E and then to the structural rows.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedCauchyInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, countable choice, open-cover compactness, host real equality, or ambient completeness.

Lean4 ✓ `BEDC.Derived.BoundedCauchyIntervalUp.BoundedCauchyIntervalTasteGate single carrier alignment`

Theorem 6065.2 (Bounded Cauchy interval NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCauchyInterval}^{\uparrow}$ *carrier $I = (B, S, R, D, L, F, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BoundedCauchyInterval}^{\uparrow}}$ *surface consists exactly of bracket admission, finite stream-window admission, regular rational readback, dyadic tolerance exposure, located regular-sequence handoff, finite located-interval net handoff, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Consequently every Real or Completion consumer of the packet must read the route

$$\begin{aligned} &B, S, R, D \\ &\longrightarrow L, F \\ &\longrightarrow E, H, C, P, N \end{aligned}$$

and cannot replace it by quotient streams, selected limits, countable choice, open-cover compactness, host real equality, or ambient completeness.

Proof. Project the carrier of Definition 6065.1. The only available rows are the bracket, stream schedule, regular readback, dyadic tolerance ledger, locatedness row, finite-net handoff, terminal real seal, and structural BEDC rows.

Use the sibling surfaces named in chapter 126, chapter 301, and chapter 291. They force each finite Cauchy request to pass through a displayed window, regular readback, and dyadic tolerance ledger before a real seal is visible.

Then apply the located and interval handoff routes of chapter 2376 and chapter 2385. These rows bind locatedness and finite interval-net coverage to the same displayed bracket and readback. The final $\mathsf{Real}^{\uparrow}$ seal of chapter 38 is therefore reached only through $B, S, R, D, L, F, E, H, C, P, N$. □

Theorem 6065.3 (Bounded Cauchy interval regular-seal handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedRegSeqRat}^{\uparrow}$, $\mathsf{HeineBorelLocatedInterval}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{BoundedCauchyInterval}^{\uparrow}$ *carrier $I = (B, S, R, D, L, F, E, H, C, P, N)$ receives the same bounded regular-seal surface: finite observations through S, regular readback through R, dyadic tolerance and bracket slack through D, locatedness through L, finite interval-net handoff through F, and the terminal real seal through E.*

The handoff exports no completion carrier beyond these rows. In particular, a consumer cannot read a selected limit, quotient stream representative, global compactness theorem, host real equality, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness proof from the packet.*

Proof. Begin with Definition 6065.2. It fixes the public route through $B, S, R, D, L, F, E, H, C, P, N$.

For regular-sequence consumers, the route is already displayed by chapter 301 after the finite window and dyadic ledger of chapter 126 and chapter 291. For located consumers, chapter 2376 uses the same regular readback and tolerance data.

For interval and Real consumers, chapter 2385 supplies the finite interval-net row and chapter 38 supplies the terminal seal. Since the carrier contains no coordinate outside the displayed finite route, the handoff cannot introduce quotient streams, selected limits, global compactness, countable choice, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchyInterval}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchyInterval}^{\uparrow}$ packet is generated from a bounded rational interval bracket, a finite StreamNameUp observation schedule, a RegSeqRatUp regular readback, a DyadicRatCoreUp tolerance ledger, a LocatedRegSeqRatUp locatedness row, a HeineBorelLocatedIntervalUp finite-net handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCauchyInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6065.1, Definition 6065.2, and Definition 6065.3 seed the bounded interval-level Cauchy packet for the Real and Completion route through finite stream, regular-rational, dyadic, located, interval-net, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCauchyInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, open-cover compactness, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bracket preservation, finite-window regular readback, dyadic tolerance exposure, locatedness, interval-net handoff, Real seal admission, and structural row exhaustion for BEDC.Derived.BoundedCauchyIntervalUp.

Chapter 6066

A Concrete Naming Certificate for $\mathsf{AbsoluteContinuity}^{\uparrow}$

Following Folland and Rudin, the $\mathsf{AbsoluteContinuity}^{\uparrow}$ packet names the finite epsilon–delta partition surface used to read absolutely continuous real functions before any measure-theoretic completion is available. It sits between the regulated-function schedules of chapter 1906, the finite partition-variation ledgers of chapter 398, the variation-measure consumers of chapter 3593, and the terminal real-name seal of chapter 38. The packet records only finite interval partitions, endpoint reads, dyadic tolerance ledgers, componentwise transport, continuation replay, provenance, and local naming rows.

Definition 6066.1 (AbsoluteContinuity carrier)**.** *An* $\mathsf{AbsoluteContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, \Pi, E, \Delta, X, D, R, V, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval-domain row, read through the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint surface of chapter 38.*
2. *Π is the finite partition family: a displayed finite list of interval subfamilies with endpoint rows, length rows, and common-refinement rows.*
3. *E is the epsilon-budget row, recording the finite positive tolerance request that the packet answers.*
4. *Δ is the delta-budget row, recording the displayed finite bound on the total length of the selected interval subfamily.*
5. *X records the* $\mathsf{Real}^{\uparrow}$ *endpoint reads, or their regular rational names when the endpoint is read through a sibling regular-name route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing interval lengths, endpoint differences, epsilon rows, and delta rows by finite dyadic observations.*
7. *R is the source row, read from either a* $\mathsf{RegulatedFunction}^{\uparrow}$ *step schedule of chapter 1906 or a* $\mathsf{BoundedVariation}^{\uparrow}$ *finite partition ledger of chapter 398.*
8. *V is the finite variation-control row consumed by chapter 3593 when the endpoint-difference ledger is repacked as variation data.*

9. *H records componentwise* hsame *transport for interval domain, finite partition family, epsilon budget, delta budget, endpoint reads, dyadic tolerance ledger, source row, variation-control row, provenance, and local naming.*
10. *C records displayed* Cont *replay from an epsilon request to a delta row, then to a finite interval subfamily, endpoint reads, dyadic comparisons, source schedule, and variation-control handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{AbsoluteContinuity}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{VariationMeasure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{AbsoluteContinuity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, \Pi, E, \Delta, X, D, R, V, H, C, P, N$. It has no coordinate for host function quotients, measure completion, selected suprema over all partitions, ambient Lebesgue integration, or unlisted endpoint representatives.

Lean4 ✓ `BEDC.Derived.AbsoluteContinuityUp.AbsoluteContinuityTasteGate single carrier alignment`

Theorem 6066.2 (AbsoluteContinuity NameCert obligations)**.** *For every accepted* $\mathsf{AbsoluteContinuity}^{\uparrow}$ *carrier $A = (I, \Pi, E, \Delta, X, D, R, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AbsoluteContinuity}^{\uparrow}}$ *surface consists exactly of interval-domain admission, finite partition-family admission, epsilon-budget admission, delta-budget admission,* $\mathsf{Real}^{\uparrow}$ *endpoint reads,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance comparisons, regulated-function or bounded-variation source routing, variation-control handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming, and non-escape from host function quotients, measure completion, selected suprema, ambient Lebesgue integration, and unlisted endpoint representatives.*

Proof. Project the carrier of Definition 6066.1. The analytic coordinates are exactly the interval row I, finite partition family Π, epsilon row E, delta row Δ, endpoint reads X, dyadic tolerance ledger D, source row R, and variation-control row V.

The source row is already confined to displayed BEDC data. If it is read through $\mathsf{RegulatedFunction}^{\uparrow}$, the schedule has passed through the finite step-envelope route of Definition 1906.5. If it is read through $\mathsf{BoundedVariation}^{\uparrow}$, it is a finite partition-variation ledger over the rows of chapter 398.

The structural rows H, C, P, N transport, replay, source, and name those same displayed coordinates. Since no carrier row names a host quotient, completed measure object, selected supremum, ambient integral, or hidden endpoint representative, none of those objects belongs to the NameCert surface. □

Theorem 6066.3 (AbsoluteContinuity finite variation control)**.** *Every accepted* $\mathsf{AbsoluteContinuity}^{\uparrow}$ *packet whose epsilon–delta row is read on a displayed finite interval subfamily exposes variation control only through*

$$I, \Pi, E, \Delta, X, D, R, V, H, C, P, N.$$

The handoff to $\mathsf{BoundedVariation}^{\uparrow}$ *and* $\mathsf{VariationMeasure}^{\uparrow}$ *is the finite endpoint-difference ledger D, V over the displayed partition family Π; it is not a host total-variation read, a measure completion, a selected supremum, or a quotient of functions.*

Proof. Start with an epsilon request in E. The continuation row C first replays the delta row Δ and the displayed interval subfamily in Π, so the request is already confined to finitely many endpoint rows.

Read the endpoint values through X and the dyadic comparisons through D. These rows give only finite endpoint-difference observations and finite length comparisons, all over the interval domain I.

The source row R then supplies either a regulated step schedule or a bounded-variation partition ledger. In both cases, the downstream variation handoff is repacked into V before any consumer in chapter 398 or chapter 3593 can read it.

Thus the exposed variation control is the displayed finite ledger D, V over Π. The carrier has no coordinate from which a host total variation, measure completion, selected supremum, quotient function, or ambient completeness principle could be obtained. □

Theorem 6066.4 (AbsoluteContinuity no-escape boundary)**.** *The* $\mathsf{AbsoluteContinuity}^{\uparrow}$ *public boundary exports finite interval-domain rows, finite partition families, epsilon and delta budget rows,* $\mathsf{Real}^{\uparrow}$ *endpoint reads,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers, regulated-function or bounded-variation source rows, finite variation-control handoff rows, visible* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *rows. It exports no measure completion, host function quotient, selected supremum over all partitions, ambient Lebesgue integral, hidden endpoint representative, or bridge claim.*

Proof. Inspect the carrier coordinates in Definition 6066.1. Every exported analytic read is one of the finite rows $I, \Pi, E, \Delta, X, D, R, V$, and every structural read is one of H, C, P, N.

Apply the NameCert-obligation theorem. It identifies the same rows as the complete certificate surface and keeps the source route inside the regulated-function and bounded-variation chapters.

Finally apply the finite variation-control theorem. Any variation-facing consumer reaches only the displayed endpoint-difference and partition ledger, so there is no route to a completed measure object, quotient function, selected supremum, ambient integral, hidden endpoint representative, or bridge theorem. □

Theorem 6066.5 (AbsoluteContinuity scoped kernel route)**.** *The carrier, NameCert-obligation surface, finite variation-control theorem, and no-escape boundary jointly bind* $\mathsf{AbsoluteContinuity}^{\uparrow}$ *to the concrete BEDC route through* $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{VariationMeasure}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *Every accepted public read factors through the displayed carrier tuple before any downstream variation or integration-facing consumer sees it.*

Proof. Project the carrier tuple from Definition 6066.1. It lists the interval, partition, tolerance, endpoint, source, variation, transport, continuation, provenance, and naming rows available to the packet.

Use Definition 6066.2 to identify those rows as the local obligation surface. The theorem also records the exclusions that keep absent analytic objects outside the packet.

Use Definition 6066.3 for downstream consumers. It forces the bounded-variation and variation-measure handoff to pass through the displayed finite partition ledger and dyadic tolerance rows.

The no-escape boundary of Definition 6066.4 then exhausts the public interface. Hence the scoped kernel route is exactly the displayed tuple and its listed BEDC dependencies. □

Closure status for $\mathsf{NameCert}_{\mathsf{AbsoluteContinuity}^{\uparrow}}$.

Constructive story: An $\mathsf{AbsoluteContinuity}^{\uparrow}$ packet is generated from an interval-domain row, a finite partition family, epsilon and delta budget rows, Real endpoint reads, a dyadic tolerance ledger, a regulated-function or bounded-variation source row, a finite variation-control handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6066.1, Definition 6066.2, Definition 6066.3, Definition 6066.4, and Definition 6066.5 seed the finite epsilon–delta partition surface over $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, and $\mathsf{VariationMeasure}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbsoluteContinuity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function quotients, measure completion, selected suprema, ambient Lebesgue integration, hidden endpoint representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, epsilon–delta partition routing, finite variation-control handoff, transport, provenance, and no-escape for BEDC.Derived.AbsoluteContinuityUp.

Chapter 6067

A Concrete Naming Certificate for $\mathsf{MonotoneBarModulus}^{\uparrow}$

Following Brouwer and Troelstra–van Dalen on monotone bars, fan bounds, and constructive choice sequences, the $\mathsf{MonotoneBarModulus}^{\uparrow}$ packet records the finite handoff by which a displayed monotone bar yields a modulus surface for current Real and Completion consumers. It sits after the fan-bound certificate of chapter 400, the barred-spread replay of chapter 1675, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite BHist route from bar evidence to requested windows; it is not countable choice, open-cover compactness, branch selection, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6067.1 (Monotone bar modulus carrier)**.** *A* $\mathsf{MonotoneBarModulus}^{\uparrow}$ *carrier is a finite packet*

$$K = (T, B, D, W, R, E, H, C, P, N)$$

whose rows have the following roles:

1. *T is the monotone bar row. It records displayed prefix decisions and the rule that a barred prefix remains barred along every displayed extension.*
2. *B is the finite fan-depth ledger. It records the bounded depth obtained from the admitted fan surface, as finite data rather than as an open cover or an infinite branch search.*
3. *D is the modulus handoff row. It translates the visible bar-depth ledger into the index budget requested by downstream readers.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which Real and Completion consumers request finite observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same requested windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the bar, fan-depth, modulus, window, and readback rows have been displayed.*
7. *H stores componentwise* hsame *transport for the monotone bar, fan-depth ledger, modulus handoff, StreamName windows, RegSeqRat readback, Real seal, continuation, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from monotone-bar inspection through the fan-depth ledger, modulus handoff, StreamName windows, RegSeqRat readback, and the terminal Real seal.*

9. *P is the* Pkg *provenance over* MonotoneBarModulus$^\uparrow$, FanTheorem$^\uparrow$, BrouwerBarInduction$^\uparrow$, StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{MonotoneBarModulus}^\uparrow}$ *row.*

The classifier compares accepted carriers only through $T, B, D, W, R, E, H, C, P, N$. It has no coordinate for countable choice, open-cover compactness, selected branches, quotient equality of streams, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.MonotoneBarModulusUp.TasteGate.MonotoneBarModulusTasteGate single carrier alignment`

Theorem 6067.2 (Monotone bar modulus nonescape)**.** *Every* StreamName$^\uparrow$, RegSeqRat$^\uparrow$, Real$^\uparrow$, *or Completion-facing read of an accepted* MonotoneBarModulus$^\uparrow$ *carrier factors through*

$$T, B, D, W, R, E, H, C, P, N.$$

The read may use the displayed monotone bar and its finite fan-depth ledger to obtain a modulus budget, but it cannot extract countable choice, open-cover compactness, a selected branch, quotient stream equality, or ambient Real$^\uparrow$ *completeness.*

Lean4 ✓ `BEDC.Derived.MonotoneBarModulusUp.TasteGate.MonotoneBarModulusNonescape`

Proof. Project the carrier of Definition 6067.1. The only analytic-facing rows are the StreamName window W, the RegSeqRat readback R, and the Real seal E.

Replay the continuation row C. It first exposes the monotone bar row T, then the finite fan-depth ledger B, then the modulus handoff row D. Only after those rows are visible can a consumer open W, read R, and reach E.

The rows H, P, N preserve and name the same route. Since no carrier coordinate stores a branch selector, quotient stream equality, an open-cover compactness principle, countable choice, or ambient completeness, none of those principles can be exported by the packet. □

Theorem 6067.3 (Monotone bar modulus NameCert obligations)**.** *For every accepted* MonotoneBarModulus$^\uparrow$ *carrier $K = (T, B, D, W, R, E, H, C, P, N)$, the local* NameCert$_{\text{MonotoneBarModulus}^\uparrow}$ *surface consists exactly of the following finite rows:*

1. *carrier admission for the monotone bar, fan-depth ledger, modulus handoff, StreamName window, RegSeqRat readback, Real seal, transport, continuation, provenance, and naming rows;*
2. *monotone bar stability, requiring extension reads to stay inside the displayed bar row T;*
3. *finite fan-depth extraction, requiring the depth budget to be read from B through the fan surface of chapter 400;*
4. *modulus handoff exactness, requiring every downstream budget request to pass through D before reaching W, R, E;*
5. *analytic boundary nonescape, requiring every Real or Completion-facing read to use the route of Definition 6067.2.*

No local obligation is stored outside these five rows.

Proof. Carrier admission is the tuple displayed in Definition 6067.1. The classifier can inspect only the listed coordinates, so the first obligation is exactly the projection of $T, B, D, W, R, E, H, C, P, N$.

The monotone bar and fan-depth clauses are read directly from T and B. The bar row gives only displayed prefix decisions and extension preservation, while the fan-depth ledger supplies only the bounded depth exposed by the finite fan certificate.

The modulus handoff clause is the row D. It is placed before W, R, E in the continuation route, so every downstream window or readback request consumes the displayed budget before reaching a Real-facing seal.

The analytic boundary is Definition 6067.2. It forces every StreamName, RegSeqRat, Real, or Completion consumer through the displayed finite rows and prevents escape through the excluded principles. □

Theorem 6067.4 (Monotone bar modulus Real handoff)**.** *An accepted* $\mathsf{MonotoneBarModulus}^{\uparrow}$ *carrier supplies Real and Completion consumers with only the following handoff:*

$$\mathsf{BrouwerBarInduction}^{\uparrow}, \mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{MonotoneBarModulus}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff records the finite passage from monotone bar evidence and a fan-depth ledger to StreamName windows, RegSeqRat readback, and a Real seal; it does not provide a separate compactness or choice principle.

Proof. Read the provenance row P together with the continuation row C. The source side names the monotone bar evidence against the barred-spread surface of chapter 1675 and the bounded fan surface of chapter 400.

The middle of the route is exactly the window and readback pair W, R, which points to chapter 126 and chapter 301. The terminal row E points to chapter 38.

Thus the handoff is a named finite route through the displayed rows of the carrier. The excluded principles are not fields of the tuple, so the Real and Completion consumers receive only the stated finite modulus surface. □

Falsifiable prediction: Every accepted $\mathsf{MonotoneBarModulus}^{\uparrow}$ Real or Completion-facing read factors through a displayed monotone bar row, finite fan-depth ledger, modulus handoff row, StreamName window, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: fantheorem, brouwer-bar-induction, streamname, regseqrat, real: $\mathsf{MonotoneBarModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries monotone bar evidence, a finite fan-depth ledger, a modulus handoff, analytic windows, readback, terminal sealing, transport, continuation, provenance, and local naming while refusing choice, compactness, branch selection, quotient streams, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{MonotoneBarModulus}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MonotoneBarModulus}^{\uparrow}$ packet is generated from a displayed monotone bar row, a finite fan-depth ledger, a modulus handoff row, StreamName windows, RegSeqRat readback, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MonotoneBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6067.1, Definition 6067.2, Definition 6067.3, and Definition 6067.4 seed the finite monotone-bar-to-modulus handoff for Real and Completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MonotoneBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.MonotoneBarModulusUp`

Bridge status: none

Not claimed: This chapter does not close countable choice, open-cover compactness, branch selection, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone bar stability, finite fan-depth extraction, modulus handoff exactness, analytic boundary nonescape, and TasteGate field faithfulness for BEDC.Derived.MonotoneBarModulusUp.

Chapter 6068

A Concrete Naming Certificate for $\mathsf{MetricCompletionAdjoint}^{\uparrow}$

Following Kelley and Bishop–Bridges, the $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ packet records the metric-specific adjoint surface by which a separated metric source enters a completion object and by which uniformly continuous maps out of the source extend through the same finite L10 route. It sits after the metric completion packet of chapter 1506, the Cauchy filter completion of chapter 1569, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the uniform completion interface of chapter 1473. The packet is a human-origin completion-adjunction carrier; it does not assert host quotient equality, selected Cauchy limits, ambient metric completeness, or a detached categorical adjunction outside the displayed rows.

Definition 6068.1 (MetricCompletionAdjoint carrier)**.** *A* $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (M, U, X, F, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the metric source row, recording the separated metric carrier, distance tolerance ledger, and finite source windows inspected by the completion request.*
2. *U is the completion unit row, routing the metric source into the* $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{UniformCompletion}^{\uparrow}$ *surfaces without choosing a host representative.*
3. *X is the universal extension route, matching uniformly continuous source maps with their completion-facing extensions as finite ledger data.*
4. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row, recording the Cauchy-filter completion face consumed by the extension route.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, exposing the regular rational name produced by the displayed completion window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after M, U, X, F, R have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, unit, extension, Cauchy-filter completion, readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$M, U, X, F \longmapsto R \longmapsto E.$$

9. *P is the* Pkg *provenance over* $\mathsf{MetricCompletionAdjoint}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, U, X, F, R, E, H, C, P, N$. It has no coordinate for quotient completion equality, selected limits, host continuous maps, countable choice, ambient completeness, or an external adjunction theorem.

Lean4 ✓ `BEDC.Derived.MetricCompletionAdjointUp.TasteGate.MetricCompletionAdjointTasteGate`
`single carrier alignment`

Theorem 6068.2 (Metric completion adjoint NameCert obligations)**.** *For every accepted* $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ *carrier*

$$A = (M, U, X, F, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjoint}^{\uparrow}}$ *surface consists exactly of carrier admission, metric-source exposure through M, completion-unit exposure through U, universal-extension routing through X, Cauchy-filter completion exposure through F, regular rational readback through R, Real seal handoff through E, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The projection exposes the metric source, completion unit, universal extension route, Cauchy-filter completion row, regular rational readback, terminal Real seal, transport, continuation, provenance, and local naming rows.

The analytic obligations are exactly the visible metric and completion rows. The source row supplies the inspected metric data, the unit row admits it into the completion surface, the extension row records the finite universal route, and the Cauchy-filter row supplies the completion face consumed by the route.

The readback and seal obligations are downstream of those rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row and $\mathsf{Real}^{\uparrow}$ seal may be consumed only after the metric, unit, extension, and Cauchy-filter rows have been displayed. Since the carrier contains no coordinate for quotient equality, selected limits, host maps, countable choice, ambient completeness, or a detached adjunction theorem, no additional local obligation can be read. □

Theorem 6068.3 (Metric completion adjoint finite extension route)**.** *Every completion-facing consumer of an accepted* $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ *packet factors through*

$$\mathsf{MetricCompletion}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow},\ \mathsf{UniformCompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

by reading the displayed rows $M, U, X, F, R, E, H, C, P, N$. The handoff supplies a finite metric completion-adjunction route, not quotient completion equality, selected Cauchy limits, host real equality, countable choice, ambient completeness, or an external categorical adjunction.

Proof. Start with the metric source and unit rows. The source row M fixes the displayed metric carrier and tolerance window, while U records the finite admission into the metric and uniform completion surfaces.

Next read the universal route and completion face. The row X is the only extension route stored by the packet, and the row F is the only Cauchy-filter completion face available to that route.

Finally pass through the readback and terminal seal. The row R exposes the regular rational name and E exposes the Real seal only after M, U, X, F are present. Transport, replay, provenance,

and naming preserve the same route, so the consumer receives finite adjunction data and no excluded host object. □

Falsifiable prediction: Every $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ read factors through metric source, completion unit, universal extension route, CauchyFilterCompletion, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before any completion-facing conclusion is consumed.

Independence witness: metriccompletion, cauchyfiltercompletion, regseqrat, real, uniformcompletion: $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ is not bijective to any listed sibling because it binds the metric source, completion unit, universal extension route, Cauchy-filter completion face, regular rational readback, Real seal, transport, replay, provenance, and local naming in one finite adjunction packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionAdjoint}^{\uparrow}$ packet is generated from a metric source row, completion unit row, universal extension route, CauchyFilterCompletion row, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6068.1, Definition 6068.2, and Definition 6068.3 seed the finite metric completion-adjunction packet over metric completion, Cauchy filter completion, uniform completion, regular readback, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetricCompletionAdjoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected Cauchy limits, host real equality, countable choice, ambient metric completeness, Lean verification, bridge checking, or a detached host adjunction theorem.

Upgrade path: Obligation closure requires checked carrier admission, metric-source exposure, completion-unit exactness, universal-extension routing, Cauchy-filter completion compatibility, readback and Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetricCompletionAdjointUp.

Chapter 6069

A Concrete Naming Certificate for MetaCICTypedConversionResidualFrontier$^{\uparrow}$

The MetaCICTypedConversionResidualFrontier$^{\uparrow}$ packet realizes the decidability-frontier question of chapter 9386 as finite BEDC data. It sits beside the decidability boundary of chapter 749, the bounded-normal conversion bridge of chapter 3439, the conversion endpoint seal of chapter 4113, the decidable conversion frontier of chapter 4097, and the subject-reduction obstruction of chapter 731. The packet names the residual frontier where bounded normal computation, typed conversion, and finished-normal evidence must remain visible together; it does not turn the bounded checker into an unrestricted closed conversion decider.

Definition 6069.1 (MetaCIC typed-conversion residual-frontier carrier)**.** *A* MetaCICTypedConversionResidualFront *carrier is a finite* BHist *packet*

$$F = (W, R, B, E, Q, O, H, C, P, N)$$

with the following displayed rows:

1. *W is the typed-window row, recording the closed typed terms, their type boundary, and the finite context window inspected by the conversion request.*
2. *R is the residual-frontier row, retaining the redex and subject-reduction obstruction data that cannot be erased by bounded computation.*
3. *B is the bounded-normal comparison row, recording the fuel, two bounded normalizer outputs, and the Boolean comparison accepted by the checker.*
4. *E is the finished-normal witness row, requiring the compared outputs to be finished closed normal forms before a conversion endpoint can be read.*
5. *Q is the refusal row, rejecting any conversion read that omits the typed window, residual frontier, bounded comparison, or finished-normal witness.*
6. *O is the endpoint-obstruction row, exposing the* MetaCICConversionEndpointSeal$^{\uparrow}$ *handoff while retaining the subject-reduction obstruction as a separate coordinate.*
7. *H records componentwise* hsame *transport for W, R, B, E, Q, O and the local naming row.*
8. *C records displayed* Cont *replay from a typed conversion request through $W, R, B, E, Q, O, H, P, N$.*

9. *P is the* Pkg *provenance over* MetaCICTypedConversionResidualFrontier$^{\uparrow}$, MetaCICDecidableBoundary$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICConversionEndpointSeal$^{\uparrow}$, MetaCICDecidableConversionFrontier$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* NameCert$_{\text{MetaCICTypedConversionResidualFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, B, E, Q, O, H, C, P, N$. It has no coordinate for unrestricted closed beta-conversion, fuel-erased normalization, quotient syntax equality, evaluator equality, subject-reduction discharge, confluence, or kernel consistency.

Theorem 6069.2 (MetaCIC typed-conversion residual-frontier NameCert obligations)**.** *For every accepted* MetaCICTypedConversionResidualFrontier$^{\uparrow}$ *carrier*

$$F = (W, R, B, E, Q, O, H, C, P, N),$$

the local NameCert$_{\text{MetaCICTypedConversionResidualFrontier}^{\uparrow}}$ *surface consists exactly of typed-window exposure through W, residual-frontier exposure through R, bounded-normal comparison through B, finished-normal witness exposure through E, refusal stability through Q, endpoint-obstruction routing through O, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier. The visible rows are the typed window, residual frontier, bounded comparison, finished-normal witness, refusal, endpoint-obstruction route, and structural rows.

The typed conversion obligations are W, R, B, E. The window fixes the typed closed terms under discussion, the residual frontier keeps the obstruction data visible, the bounded comparison records the finite normalizer outputs, and the finished-normal witness certifies that the endpoint read is not merely a fuel-bounded observation.

The refusal and endpoint-obstruction rows preserve the boundary. They reject reads that omit any of the four typed conversion rows, and they keep the conversion endpoint seal separate from subject-reduction discharge. Since no carrier coordinate contains unrestricted conversion, fuel-erased normalization, quotient syntax equality, evaluator equality, confluence, or kernel consistency, the listed rows exhaust the NameCert surface. □

Theorem 6069.3 (MetaCIC typed-conversion endpoint frontier)**.** *Every typed conversion consumer of an accepted* MetaCICTypedConversionResidualFrontier$^{\uparrow}$ *packet factors through the finite route*

MetaCICDecidableBoundary$^{\uparrow}$, BoundedNormalConversionBridge$^{\uparrow}$, MetaCICConversionEndpointSeal$^{\uparrow}$, MetaCICDecida

with $W, R, B, E, Q, O, H, C, P, N$ displayed. The route exports a residual-frontier packet for bounded normal comparison and finished-normal endpoint evidence; it does not export unrestricted closed conversion, subject-reduction discharge, confluence, or kernel consistency.

Proof. Begin with the typed window. The row W fixes the closed typed terms and their finite context-facing boundary before any checker result is read.

Next read the bounded and residual rows together. The row B records the bounded normal comparison, while R retains the residual and subject-reduction obstruction data that prevent the comparison from becoming an unrestricted conversion theorem.

Finally consume the endpoint evidence. The row E supplies the finished-normal witness, O routes the endpoint seal while retaining the obstruction, and Q rejects bypass reads. Transport, replay, provenance, and naming preserve the same finite route, so the consumer receives endpoint-frontier data only. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}$ read displays typed-window, residual-frontier, bounded-normal comparison, finished-normal witness, refusal, endpoint-obstruction, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a typed conversion endpoint is consumed.

Independence witness: metacicdecidableboundary, bounded-normal-conversion-bridge, metacic-conversion-endpoint-seal, metacic-decidable-conversion-frontier, metacic-subject-reduction-obstruction: the residual-frontier carrier is not bijective to any listed sibling because it binds typed-window data, residual obstruction, bounded normal comparison, finished-normal witness, endpoint refusal, endpoint seal routing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}$ packet is generated from typed-window data, residual-frontier data, bounded-normal comparison, finished-normal witness, refusal, endpoint-obstruction routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6069.1, Definition 6069.2, and Definition 6069.3 seed the finite residual frontier for typed MetaCIC conversion.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICTypedConversionResidualFrontier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim unrestricted closed beta-conversion, fuel-erased normalization, quotient syntax equality, evaluator equality, subject-reduction discharge, confluence, kernel consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked typed-window exposure, residual-frontier retention, bounded-normal comparison, finished-normal witness, endpoint refusal, endpoint-obstruction routing, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICTypedConversionResidualFrontierUp.

Chapter 6070

A Concrete Naming Certificate for HyperbolicPattersonSullivanShadow$^{\uparrow}$

Following Patterson and Sullivan on hyperbolic limit-set shadows, the HyperbolicPattersonSullivanShadow$^{\uparrow}$ packet records a finite shadow ledger for boundary-weight readout. It sits after the visual-metric certificate of chapter 3866, the boundary-shadow route of chapter 3813, the Busemann-boundary certificate of chapter 3732, and the geodesic-flow carrier of chapter 3856. The packet names a displayed Patterson–Sullivan shadow surface over finite boundary, window, counting, transport, replay, provenance, and local naming rows; it is not a completed conformal density, quotient limit set, ambient ergodic theorem, host hyperbolic group action, or global measure construction.

Definition 6070.1 (Hyperbolic Patterson-Sullivan shadow carrier)**.** *A* HyperbolicPattersonSullivanShadow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, V, W, C, R, H, K, P, N)$$

with the following displayed rows:

1. *B is the boundary basepoint row, naming the finite Busemann-boundary face from which the shadow is observed.*
2. *V is the* HyperbolicVisualMetric$^{\uparrow}$ *row, displaying the visual metric comparison used to read the boundary shadow.*
3. *W is the shadow-window row, recording the finite boundary-shadow aperture and the accepted visible window.*
4. *C is the critical exponent and counting-budget row, recording the finite growth bound under which the shadow ledger is admitted.*
5. *R is the Busemann transport row, tying the basepoint, visual metric, shadow window, and counting budget to one boundary-facing route.*
6. *H records componentwise* hsame *transport for B, V, W, C, R, K, P, N.*
7. *K records displayed* Cont *replay from boundary-basepoint admission through visual-metric read-back, shadow-window exposure, critical-exponent counting, and Busemann transport.*
8. *P is the* Pkg *provenance over* HyperbolicPattersonSullivanShadow$^{\uparrow}$, HyperbolicVisualMetric$^{\uparrow}$, HyperbolicBoundaryS HyperbolicBusemannBoundary$^{\uparrow}$, HyperbolicGeodesicFlow$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $B, V, W, C, R, H, K, P, N$. It has no coordinate for a completed conformal density, quotient limit set, ambient ergodic theorem, host group action, global measure construction, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.HyperbolicPattersonSullivanShadowUp.HyperbolicPattersonSullivanShadowTasteGate single carrier alignment`

Theorem 6070.2 (Hyperbolic Patterson-Sullivan shadow NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}$ *carrier $S = (B, V, W, C, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicPatters}}$ *surface consists exactly of boundary-basepoint admission, visual-metric row exposure, shadow-window admission, critical-exponent and counting-budget ledger exposure, Busemann transport, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming. These five NameCert obligations are carrier admission for B, V, W, C, route exactness through R, transport stability through H, replay stability through K, and provenance-local-name exhaustion through P, N.*

Proof. Project the carrier of Definition 6070.1. The displayed tuple exposes the boundary basepoint, visual-metric row, shadow-window row, critical exponent and counting budget, Busemann transport, and the structural rows.

The first obligation is carrier admission: B, V, W, C are the only geometric and counting rows admitted by the packet. The second obligation is route exactness: R ties those rows into one Busemann-transport route before any shadow ledger can be read.

The remaining obligations are structural. The row H gives componentwise hsame transport, K replays the same route by displayed Cont, and P, N record provenance and local naming. Since the carrier has no coordinate for a completed conformal density, quotient limit set, ambient ergodic theorem, host group action, global measure construction, Lean verification, or bridge checking, these rows exhaust the NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}$ read must expose a boundary basepoint, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ row, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ window, critical-exponent counting budget, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before it exports a public shadow ledger.

Independence witness: hyperbolic-visual-metric, hyperbolic-boundary-shadow, hyperbolic-busemann-boundary, hyperbolic-geodesic-flow: no listed sibling is bijective to $\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}$ because this packet jointly carries boundary-basepoint admission, visual comparison, shadow-window exposure, critical-exponent counting, Busemann transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6070.1 and Definition 6070.2 seed the finite Patterson–Sullivan shadow ledger through boundary basepoints, visual metrics, shadow windows, counting budgets, Busemann transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicPattersonSullivanShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed conformal densities, quotient limit sets, ambient ergodic theorems, host hyperbolic group actions, global measure constructions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, visual-metric exposure, shadow-window admission, critical-exponent counting, Busemann transport, replay stability, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.HyperbolicPattersonSullivanShadowUp.

Constructive story: A HyperbolicPattersonSullivanShadow$^{\uparrow}$ packet is generated from a boundary basepoint, visual-metric row, shadow-window row, critical-exponent counting-budget row, Busemann transport, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Chapter 6071

A Concrete Naming Certificate for $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$

Following Cauchy and Bishop–Bridges constructive analysis, the $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ packet records the finite comparison bridge between dyadic condensation tails and integral-test windows. It sits after the theorem-level condensation surface of chapter 3727, the integral-window route of chapter 2259, the dyadic tolerance ledger of chapter 291, the regular rational readback corridor of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite handoff surface for Real and Completion consumers; it is not a quotient stream, a selected infinite sum, a host function equality, an ambient integral, countable choice, or a completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 6071.1 (CauchyCondensationIntegralBridge carrier)**.** *A* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (K, I, D, W, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the accepted* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *row, exposing the monotone source tail, dyadic schedule, original block sums, condensed dyadic terms, and Cauchy-criterion handoff.*
2. *I is the accepted* $\mathsf{CauchyIntegralTest}^{\uparrow}$ *row, exposing the nonnegative monotone observation, the Darboux or Riemann integral tail-window row, and the regular Cauchy series-tail row.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger that compares the dyadic block endpoints used by K with the finite integral-window endpoints used by I.*
4. *W is the common finite window schedule. It lists the block indices, interval endpoints, tolerances, and comparison rows inspected before any regular readback is exported.*
5. *Q is the bridge comparison row, recording that the condensation tail and the integral window are read over the same displayed tolerance budget.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from D, W, Q.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the regular readback.*
8. *H records componentwise* hsame *transports for the condensation row, integral row, dyadic ledger, window schedule, comparison row, readback, real seal, replay, provenance, and local name.*

9. *C records displayed* Cont *routes from each finite precision request through the dyadic and integral windows to R and E.*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$, $\mathsf{CauchyCondensationTheorem}^{\uparrow}$, $\mathsf{CauchyIntegralTest}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for quotient streams, host functions, selected sums, ambient integrals, arbitrary real equality, countable choice, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6071.2 (CauchyCondensationIntegralBridge NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ *carrier $J = (K, I, D, W, Q, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyCondensatior}}$ *surface consists exactly of carrier admission for the displayed eleven-row packet; condensation-row exposure through K; integral-window exposure through I; dyadic endpoint and tolerance comparison through D; finite schedule coverage through W; bridge comparison exposure through Q;* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R;* $\mathsf{Real}^{\uparrow}$ *sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C; provenance through P; and local naming through N.*

Proof. Project the accepted packet $J = (K, I, D, W, Q, R, E, H, C, P, N)$. The projection exposes exactly the condensation theorem row, integral-test row, dyadic ledger, finite window schedule, bridge comparison, regular readback, real seal, transport, replay, provenance, and local name.

Read a finite tail request through K and I. The condensation side supplies dyadic block endpoints and condensed terms, while the integral side supplies the matching tail-window row. The dyadic ledger D is the only comparison surface between those two rows.

Now inspect W and Q. They bind the finite block indices, integral endpoints, and tolerance budget before R can be produced, so neither side can export a separate quotient stream, host function, selected sum, or ambient integral.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6071.3 (CauchyCondensationIntegralBridge dyadic tail exactness)**.** *Every accepted* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ *carrier preserves the dyadic tail route of* $\mathsf{CauchyCondensationTheorem}^{\uparrow}$ *before it enters an integral window. Concretely, a consumer can read the bridge comparison only through the finite path*

$$K, D, W, Q \longmapsto R, E, H, C, P, N,$$

where K supplies the condensation tail rows, D supplies the dyadic tolerance ledger, and W, Q record the common comparison window.

Proof. Begin with an accepted bridge carrier and project K. By the condensation theorem packet, the monotone source, dyadic schedule, block sums, condensed terms, and Cauchy handoff are already displayed before the bridge reads them.

Read D against that displayed dyadic schedule. The tolerance ledger compares only finite endpoints and finite dyadic budgets, so the bridge does not select a new tail, quotient a stream, or appeal to an ambient real limit.

Finally use W, Q, R, E, H, C, P, N. The window and comparison rows attach the dyadic tail to the regular readback and real seal along the same displayed route, while the structural rows preserve that route componentwise. □

Theorem 6071.4 (CauchyCondensationIntegralBridge integral-window handoff)**.** *Every accepted* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ *carrier hands an integral-test window to Real and Completion consumers only after the condensation-side dyadic tail and the integral-side tail window have been matched by* D, W, Q*. The handoff exports the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the* $\mathsf{Real}^{\uparrow}$ *seal row* E*, but exports no host function equality, ambient integral object, selected infinite sum, quotient stream, countable-choice schedule, or completeness principle.*

Proof. Project the integral-test row I. Its finite tail-window data is the only integral-facing input available to the bridge, and it is read beside the nonnegative monotone observation and regular Cauchy series-tail row already named by $\mathsf{CauchyIntegralTest}^{\uparrow}$.

Use the dyadic exactness theorem above to align the same request with K, D, W, Q. The bridge comparison cannot reach R until both the condensation tail and the integral window have been displayed under the shared tolerance budget.

Read the seal row E through R and the structural rows. Since the carrier contains no host function equality, ambient integral coordinate, selected sum, quotient stream, countable-choice schedule, or completeness field, the Real and Completion handoff is exactly the finite regular readback route. □

Theorem 6071.5 (CauchyCondensationIntegralBridge obligation handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ *carrier must read*

$$K, I, D, W, Q, R, E, H, C, P, N$$

in order: condensation theorem row, integral-test row, dyadic endpoint ledger, finite window schedule, bridge comparison row, regular readback row, Real seal row, hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert*. The route exports no host summability, quotient stream equality, selected infinite sum, ambient integral, countable choice, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The first available rows are K and I, so every Real or Completion consumer must begin with the condensation theorem surface and the integral-test window surface.

The comparison rows come next. The dyadic ledger D, window schedule W, and bridge comparison row Q bind the two source rows to one finite tolerance budget before any regular readback is exposed.

Only then may the consumer read R and E. These rows provide the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal, and they do not add host summability, quotient stream equality, a selected infinite sum, an ambient integral, countable choice, or a completeness theorem.

Finally H, C, P, N transport, replay, source, and name exactly the displayed route. Since the carrier fields are exhausted by $K, I, D, W, Q, R, E, H, C, P, N$, no excluded object can be exported by the handoff. □

Falsifiable prediction: An accepted $\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without reading the displayed rows K, I, D, W, Q, R refutes the carrier surface.

Independence witness: cauchy-condensation-theorem, cauchy-integral-test, dyadicratcore, regseqrat, real: no listed sibling supplies the whole bridge packet $J = (K, I, D, W, Q, R, E, H, C, P, N)$, because each sibling contributes only a condensation theorem row, an integral-window row, a dyadic ledger, a regular readback, or a terminal seal while this chapter binds them to one finite comparison certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCondensationIntegralBridge}^{\uparrow}}$.

Constructive story: A CauchyCondensationIntegralBridge$^{\uparrow}$ packet is generated from a CauchyCondensationTheorem row, a CauchyIntegralTest row, a DyadicRatCore endpoint and tolerance ledger, a finite shared window schedule, a bridge comparison row, a RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyCondensationIntegralBridge}^{\uparrow}}$, obligation)

Scope closed: Definition 6071.1, Definition 6071.2, Definition 6071.3, Definition 6071.4, and Definition 6071.5 close the obligation handoff between condensation tails and integral windows over DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, and Real$^{\uparrow}$.

Formal verification: FormalStatus(NameCert$_{\text{CauchyCondensationIntegralBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host summability, quotient stream equality, selected infinite sums, ambient integrals, countable choice, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires explicit dependency scope for Real and Completion consumers of BEDC.Derived.CauchyCondensat

Chapter 6072

A Concrete Naming Certificate for $\mathsf{OptionalSampling}^{\uparrow}$

Following Doob, the $\mathsf{OptionalSampling}^{\uparrow}$ packet names the bounded optional-sampling surface that sits between the finite optional-stopping certificate of chapter 383 and the martingale-convergence preparation route of chapter 2246. It consumes the probability, random-variable, conditional-expectation, martingale, stopping-time, and optional-stopping rows already displayed by those siblings, and it exports only a bounded stopped-window handoff. It does not introduce unbounded stopping-time choice, almost-sure quotient equality, or a completed measure carrier.

Definition 6072.1 (Optional sampling carrier)**.** *An* $\mathsf{OptionalSampling}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, X, E, M, T, O, B, L, H, C, Q, N)$$

with the following displayed rows:

1. *P is the consumed* $\mathsf{ProbSpace}^{\uparrow}$ *probability-source row.*
2. *X is the finite* $\mathsf{RandomVar}^{\uparrow}$ *endpoint family read by the stopped window.*
3. *E is the* $\mathsf{CondExp}^{\uparrow}$ *conditional-expectation face used by the martingale law.*
4. *M is the finite* $\mathsf{Martingale}^{\uparrow}$ *adapted-window row.*
5. *T is the bounded* $\mathsf{StoppingTime}^{\uparrow}$ *decision row.*
6. *O is the* $\mathsf{OptionalStopping}^{\uparrow}$ *stopped-window row over the same martingale and bound.*
7. *B is the bounded sampling-window row, displaying the lower and upper stopping indices and the finite horizon in which both are read.*
8. *L is the martingale-law handoff row, recording that the sampled endpoint comparison is read through E, M, T, O, B.*
9. *H records componentwise* hsame *transport for the displayed probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, sampling-window, and law rows.*
10. *C records displayed* Cont *replay from the bounded stopping decision to the sampled-window comparison.*
11. *Q is the* Pkg *provenance over* $\mathsf{OptionalSampling}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{StoppingTime}^{\uparrow}$, $\mathsf{OptionalStopping}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. N is the local $\mathsf{NameCert}_{\mathsf{OptionalSampling}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $P, X, E, M, T, O, B, L, H, C, Q, N$. No coordinate names an unbounded stopping theorem, a selected almost-sure representative, a completed measure space, or an ambient martingale convergence theorem.

Lean4 (definition) `BEDC.Derived.OptionalSamplingUp.TasteGate.OptionalSamplingUp`
Lean4 ✓ `BEDC.Derived.OptionalSamplingUp.TasteGate.OptionalSamplingTasteGate single carrier alignment`

Theorem 6072.2 (Optional sampling NameCert obligations)**.** *For every accepted* $\mathsf{OptionalSampling}^{\uparrow}$ *carrier $S = (P, X, E, M, T, O, B, L, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{OptionalSampling}^{\uparrow}}$ *surface consists exactly of carrier admission for P, X, E, M, T, O, B, L, classifier transport through H, bounded stopped-window exactness through O, B, C, martingale-law handoff through E, M, L, provenance-local-name exhaustion through Q, N, and no-escape from the displayed bounded stopping rows.*

Proof. Project the carrier of Definition 6072.1. The probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, bounded-window, and law-handoff rows are the only mathematical coordinates of the packet.

Carrier and classifier obligations are immediate from those coordinates. Rows P, X, E, M, T, O, B, L admit the bounded sampling surface, while H transports it componentwise and C replays the same finite route.

The stopped-window exactness is bounded because O and B expose the finite horizon before the handoff row L is read. The martingale-law handoff is therefore consumed through E, M, T, O, B, L, not through a host stopping theorem.

Finally Q, N exhaust the provenance and local naming rows. Since no carrier coordinate contains unbounded stopping-time choice, almost-sure quotient equality, completed measure data, or convergence, the NameCert surface exports none of those objects. □

Theorem 6072.3 (Optional sampling bounded stopping handoff)**.** *Every optional-sampling read consumed by* $\mathsf{MartingaleConvergence}^{\uparrow}$ *factors through the displayed bounded stopping rows T, O, B, L of an accepted* $\mathsf{OptionalSampling}^{\uparrow}$ *carrier. The handoff exports the finite stopped-window comparison over* $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CondExp}^{\uparrow}$, *and* $\mathsf{Martingale}^{\uparrow}$ *rows, and exports no unbounded stopping-time choice, almost-sure quotient equality, completed measure carrier, or ambient convergence theorem.*

Proof. Start with an accepted $\mathsf{OptionalSampling}^{\uparrow}$ carrier. The stopping decision is the bounded row T, and the stopped process is exposed only through the $\mathsf{OptionalStopping}^{\uparrow}$ row O inside the finite sampling window B.

The martingale-law handoff is L. By Definition 6072.2, L is read after the conditional-expectation and martingale rows E, M and after the bounded stopped-window rows T, O, B.

Thus a downstream $\mathsf{MartingaleConvergence}^{\uparrow}$ consumer receives only the displayed finite route. The packet has no coordinate for unbounded stopping-time choice, almost-sure quotient equality, measure completion, or convergence, so the handoff cannot export any of them. □

Closure status for $\mathsf{NameCert}_{\mathsf{OptionalSampling}^{\uparrow}}$.
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OptionalSampling}^{\uparrow}}, \mathsf{seed})$
Scope closed: Definition 6072.1, Definition 6072.2, and Definition 6072.3 seed the bounded optional-sampling packet over displayed probability, random-variable, conditional-expectation, martingale, stopping-time, optional-stopping, transport, replay, provenance, and local naming rows.
Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OptionalSampling}^{\uparrow}}, \mathsf{unformalized})$
Bridge status: none

Not claimed: This chapter does not close unbounded optional sampling, almost-sure quotient equality, completed measure carriers, martingale convergence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier transport, bounded stopped-window exactness, martingale-law handoff, provenance exhaustion, no-escape, and TasteGate field faithfulness for BEDC.Derived.OptionalSamplingUp.

Constructive story: An OptionalSampling$^{\uparrow}$ packet is generated from displayed ProbSpaceUp, RandomVarUp, CondExpUp, MartingaleUp, StoppingTimeUp, OptionalStoppingUp, bounded sampling-window, martingale-law handoff, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6073

A Concrete Naming Certificate for $\mathsf{HyperbolicGromovProduct}^{\uparrow}$

The $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ packet realizes the boundary-shadow and visual-metric comparison route of chapter 9463 as finite basepointed NameCert data. It sits after the visual-metric packet of chapter 3866, the Busemann-boundary packet of chapter 3732, the boundary-shadow packet of chapter 3813, and the observer-expansion packet of chapter 4167. The packet names only a displayed Gromov-product comparison surface; it does not complete the boundary, select geodesic representatives, import a host hyperbolic space, or identify visual metrics by quotient equality.

Definition 6073.1 (Hyperbolic Gromov product carrier)**.** *A* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (B, V, U, S, E, R, H, C, P, N)$$

with the following displayed rows:

1. *B is the basepoint row naming the finite observer base from which the product is read.*
2. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *comparison row.*
3. *U is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *boundary-distance row.*
4. *S is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *visual-boundary shadow row.*
5. *E is the* $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$ *observer-expansion row tying the comparison to the focused route.*
6. *R is the Gromov-product readback row, recording the finite basepointed comparison extracted from B, V, U, S, E.*
7. *H records componentwise* hsame *transport for basepoint, visual-metric, Busemann-boundary, boundary-shadow, observer-expansion, readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from basepoint admission through visual-boundary shadow comparison to the Gromov-product readback.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}$ $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGromovProduct}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, V, U, S, E, R, H, C, P, N$. It has no coordinate for a completed ideal boundary, a selected geodesic-ray quotient, a host hyperbolic group action, or a global visual metric.

Theorem 6073.2 (Hyperbolic Gromov product NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGromovProd}$ *carrier $G = (B, V, U, S, E, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGromovProduct}^{\uparrow}}$ *surface consists exactly of basepoint admission through B, visual-metric comparison through V, Busemann-boundary exposure through U, visual-boundary shadow admission through S, observer-expansion anchoring through E, Gromov-product readback through R, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6073.1. The geometric data are exactly the basepoint, visual metric, Busemann boundary, boundary shadow, observer-expansion, and readback rows.

The obligation surface reads each geometric coordinate once. Row B admits the basepoint, V supplies the visual comparison, U supplies the Busemann boundary face, S supplies the visible shadow row, E anchors the observer-expansion route, and R records the product readback.

The remaining rows are structural. Transport is componentwise through H, replay is displayed through C, and P, N provide provenance and local naming. Since no carrier coordinate contains a completed boundary, selected quotient, host hyperbolic action, or global visual metric, the NameCert obligations do not export those objects. □

Theorem 6073.3 (Hyperbolic Gromov product boundary-shadow handoff)**.** *Every boundary-shadow or visual-metric consumer of an accepted* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *packet factors its Gromov-product read through the displayed rows*

$$B, V, U, S, E, R, H, C, P, N.$$

The handoff preserves the basepoint row, visual-metric comparison, Busemann-boundary exposure, boundary-shadow row, observer-expansion route, product readback, transport, replay, provenance, and local name; it exports no completed boundary, quotient geodesic equality, host hyperbolic group action, or global visual metric.

Proof. Begin with the accepted carrier. Its basepointed comparison is not read from a host metric space; it is the displayed row R over B, V, U, S, E.

Apply Definition 6073.2. The visual-metric and boundary-shadow information enters only through V and S, while the Busemann and observer-expansion rows U, E keep the boundary comparison attached to the focused route.

Finally transport and replay the same rows through H, C, P, N. The carrier has no coordinate for completed boundaries, quotient geodesic equality, host group actions, or global visual metrics, so the handoff remains finite and local. □

Falsifiable prediction: A $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ read is rejected whenever its basepoint row or visual-boundary shadow row is absent.

Independence witness: hyperbolic-visual-metric, hyperbolic-busemann-boundary, hyperbolic-boundary-shadow, hyperbolic-observer-expansion: no listed sibling is bijective to $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ because this packet jointly carries basepoint admission, visual-metric comparison, Busemann-boundary exposure, boundary-shadow readback, observer-expansion anchoring, product readback, transport, replay, provenance, and local naming rows.

The Lean follow-up for this packet must provide `Nontrivial` and `FieldFaithful` instances in `lean4/BEDC/Derived/HyperbolicGromovProductUp/TasteGate.lean`; the paper packet records the finite NameCert surface those instances must preserve.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGromovProduct}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGromovProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6073.1, Definition 6073.2, and Definition 6073.3 seed the finite basepointed Gromov-product packet over visual-metric, Busemann-boundary, boundary-shadow, observer-expansion, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGromovProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed ideal boundaries, quotient geodesic equality, host hyperbolic group actions, global visual metrics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked basepoint admission, visual-metric comparison, Busemann-boundary exposure, boundary-shadow readback, observer-expansion anchoring, product readback, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.HyperbolicGromovProductUp.

Constructive story: A $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ packet is generated from a basepoint row, HyperbolicVisualMetricUp row, HyperbolicBusemannBoundaryUp row, HyperbolicBoundaryShadowUp row, HyperbolicObserverExpansionUp row, Gromov-product readback, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Chapter 6074

A Concrete Naming Certificate for $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$ packet records the finite Cauchy-window exhaustion surface used by real-completion consumers before any completed real object is exposed. It sits over the uniform Cauchy criterion of chapter 435, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names finite exhaustion data, not quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6074.1 (Uniform Cauchy window exhaustion carrier)**.** *A* $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, T, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row of finite stream windows inspected by the exhaustion request.*
2. *T is the uniform threshold row, assigning each displayed tolerance request a single threshold that governs all later windows named by S.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic radii, dominance checks, and comparison budgets used by T.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row assembled from the window observations in S after the threshold and dyadic rows have been displayed.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, T, D, R have been replayed.*
6. *H records componentwise* hsame *transport for the stream-window, threshold, dyadic-ledger, regular-readback, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$S, T, D \longmapsto R \longmapsto E.$$

8. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, D, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected limit, a countable-choice selector, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.UniformCauchyWindowExhaustionUp.TasteGate.UniformCauchyWindowExhaustionUp`
Lean4 ✓ `BEDC.Derived.UniformCauchyWindowExhaustionUp.TasteGate.UniformCauchyWindowExhaustionTasteGate`
`single carrier alignment`

Theorem 6074.2 (Uniform Cauchy window exhaustion NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$ *carrier $U = (S, T, D, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyWin}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed stream-window row, uniform threshold row, dyadic tolerance ledger, regular rational handoff, real seal, transport, replay, provenance, and local naming row;*
2. *finite-window exposure, requiring every exhaustion read to inspect the* $\mathsf{StreamName}^{\uparrow}$ *row S before it can consult T, D, R, E;*
3. *uniform-threshold exactness, requiring the threshold row T to govern all later windows named by S for the same displayed tolerance request;*
4. *dyadic-to-regular handoff, requiring the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger D to feed the* $\mathsf{RegSeqRat}^{\uparrow}$ *row R before the* $\mathsf{Real}^{\uparrow}$ *seal E is visible;*
5. *ledger refusal, excluding quotient streams, selected limits, countable choice, host real equality, and ambient real completeness as certificate data.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 6074.1. The accepted inventory is exactly $S, T, D, R, E, H, C, P, N$, so every local obligation must be read from those rows.

The finite-window and uniform-threshold rows are upstream. A consumer cannot read a regular rational handoff until the displayed stream windows S, the threshold row T, and the dyadic tolerance ledger D have supplied the requested tail comparison.

The real seal is downstream of the regular handoff. The row R records the $\mathsf{RegSeqRat}^{\uparrow}$ readback assembled from the same windows and budgets, and E seals only that displayed readback.

Transport by H, replay by C, provenance by P, and local naming by N preserve the same route. Since quotient stream equality, selected limits, countable choice, host real equality, and ambient real completeness have no carrier coordinate, the five listed rows exhaust the local certificate surface. □

Falsifiable prediction: Any accepted $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$ consumer that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ read without the displayed StreamName windows, uniform threshold row, DyadicRatCore tolerance ledger, transport, replay, provenance, and local NameCert rows refutes the carrier.

Independence witness: UniformCauchyCriterion, StreamName, RegSeqRat, DyadicRatCore, and Real are not individually bijective with $\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}$ because the packet jointly carries finite windows, one uniform threshold row, a dyadic tolerance ledger, a regular rational handoff, a Real seal, componentwise transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyWindowExhaustion}^{\uparrow}}$.

Constructive story: A UniformCauchyWindowExhaustion$^{\uparrow}$ packet is generated from finite StreamName windows, a uniform threshold row, a DyadicRatCore tolerance ledger, RegSeqRat handoff, RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{UniformCauchyWindowExhaustion}^{\uparrow}}$, seed)

Scope closed: Definition 6074.1 and Definition 6074.2 seed the finite uniform Cauchy-window exhaustion surface over StreamName, threshold, dyadic tolerance, RegSeqRat, Real, transport, replay, provenance, and local NameCert rows.

Formal verification: FormalStatus(NameCert$_{\text{UniformCauchyWindowExhaustion}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.UniformCauchyWindowExhaustionUp.TasteGate.UniformCauchyWindowExhaustionTasteGate\_singl`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-threshold exactness, dyadic tolerance routing, RegSeqRat handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchyWindowEx

Chapter 6075

A Concrete Naming Certificate for $\mathsf{PointedGromovHausdorff}^{\uparrow}$

Following Gromov and Burago–Burago–Ivanov, the $\mathsf{PointedGromovHausdorff}^{\uparrow}$ packet records compact-metric comparison with displayed basepoints. It sits after the ordinary comparison carrier of chapter 2104, the compact-metric surface of chapter 129, the metric source of chapter 57, the scalar Hausdorff-distance route of chapter 3653, and the basepoint-sensitive observer rows of chapter 6073. The packet names a finite pointed comparison surface; it is not a quotient compactum, a selected optimal pointed correspondence, a completed moduli space, or an ambient isometry theorem.

Definition 6075.1 (Pointed Gromov–Hausdorff carrier)**.** *A* $\mathsf{PointedGromovHausdorff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, x_0, y_0, K_X, K_Y, R, D, B, H, Q, U, T, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two compact-metric endpoint names, each read through the* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{CompactMetric}^{\uparrow}$ *source rows.*
2. *x_0 and y_0 are the displayed basepoint rows attached to the two endpoints before any comparison row is public.*
3. *K_X and K_Y are the endpoint compact-metric certificates consumed by the ordinary* $\mathsf{GromovHausdorff}^{\uparrow}$ *route.*
4. *R is the finite pointed correspondence table, containing the ordinary correspondence rows and the displayed pair (x_0, y_0).*
5. *D is the dyadic distortion ledger for endpoint distances along the displayed correspondence rows.*
6. *B is the basepoint preservation ledger, recording that the pointed correspondence reads the pair (x_0, y_0) as distinguished rather than as an anonymous correspondence row.*
7. *H records the* $\mathsf{HausdorffDistance}^{\uparrow}$ *and* $\mathsf{HausdorffMetric}^{\uparrow}$ *comparison rows used to read the compact-subset distance data attached to the same finite table.*
8. *Q records the optional observer handoff to the* $\mathsf{HyperbolicGromovProduct}^{\uparrow}$ *basepoint-sensitive comparison surface when such a consumer is present.*

9. *U records componentwise* hsame *transport for endpoints, basepoints, compact-metric rows, correspondence rows, distortion bounds, basepoint preservation, comparison handoffs, replay, provenance, and local naming rows.*
10. *T records displayed transport of the pointed comparison surface across accepted endpoint and basepoint renamings.*
11. *C records displayed* Cont *replay of the route*

$$X, Y, x_0, y_0, K_X, K_Y \longmapsto R \longmapsto D, B \longmapsto H, Q \longmapsto U, T.$$

12. *P is the* Pkg *provenance over* PointedGromovHausdorff$^{\uparrow}$, GromovHausdorff$^{\uparrow}$, CompactMetric$^{\uparrow}$, Met$^{\uparrow}$, HausdorffDistance$^{\uparrow}$, HyperbolicGromovProduct$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{PointedGromovHausdorff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, Y, x_0, y_0, K_X, K_Y, R, D, B, H, Q, U, T, C, P, N$. It has no coordinate for unpointed quotient compacta, selected optimal correspondences, completed moduli spaces, host equality of compact metric spaces, or ambient isometry of pointed completions.

Lean4 ✓ `BEDC.Derived.PointedGromovHausdorffUp.TasteGate.PointedGromovHausdorffTasteGate single carrier alignment`

Theorem 6075.2 (Pointed Gromov–Hausdorff NameCert obligations)**.** *For every accepted* PointedGromovHausdorf *carrier*

$$G = (X, Y, x_0, y_0, K_X, K_Y, R, D, B, H, Q, U, T, C, P, N),$$

the local NameCert$_{\text{PointedGromovHausdorff}^{\uparrow}}$ *surface consists exactly of endpoint admission, basepoint admission, compact-metric support, pointed-correspondence admission, dyadic distortion exactness, basepoint preservation, Hausdorff-distance comparison, optional basepoint-sensitive observer handoff, componentwise transport, displayed replay, provenance, and local naming through the displayed rows of G.*

Proof. Project the carrier of Definition 6075.1. The endpoint rows X, Y, K_X, K_Y give the same compact-metric surface consumed by GromovHausdorff$^{\uparrow}$, while x_0, y_0 are additional rows attached before any correspondence is read.

The pointed correspondence row R and the distortion ledger D provide the ordinary comparison data. The row B then marks the distinguished pair (x_0, y_0) as preserved by that correspondence, so the basepoint information is not recovered from an unpointed quotient.

The comparison handoffs H, Q read the same finite table through Hausdorff-distance and basepoint-sensitive observer surfaces. The structural rows U, T, C, P, N transport, replay, source, and name exactly those displayed rows.

Since no further coordinate appears in the carrier, the listed rows exhaust the local NameCert obligations and export no selected optimal correspondence, completed moduli object, host equality, or ambient isometry theorem. □

Theorem 6075.3 (Pointed-to-unpointed forgetful route)**.** *Every ordinary* GromovHausdorff$^{\uparrow}$ *consumer of an accepted* PointedGromovHausdorff$^{\uparrow}$ *carrier factors through the forgetful route*

$$X, Y, K_X, K_Y, R, D, H, U, T, C, P, N.$$

The pointed rows x_0, y_0, B remain available only to pointed consumers. Forgetting them recovers the ordinary compact-metric comparison surface, but it does not allow the unpointed carrier to inspect, preserve, or compare basepoints.

Proof. Start with the accepted carrier and erase the basepoint-specific rows x_0, y_0, B. The remaining endpoint, compact-metric, correspondence, distortion, Hausdorff-distance, transport, replay, provenance, and local-name rows have the same shape as the ordinary comparison route of Definition 2104.3.

A pointed consumer may read the erased rows before forming its comparison claim. An ordinary consumer cannot, because the target $\mathsf{GromovHausdorff}^{\uparrow}$ surface contains no basepoint coordinate and no basepoint preservation ledger.

Thus the forgetful route is constructive and finite: it projects the pointed packet to the ordinary comparison rows, while preserving the refusal stated in Definition 2104.4. Basepoint comparison requires the pointed carrier itself. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointedGromovHausdorff}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedGromovHausdorff}^{\uparrow}$ packet is generated from two compact-metric endpoints, displayed basepoints, finite pointed correspondence rows, dyadic distortion bounds, a basepoint preservation ledger, HausdorffDistanceUp comparison rows, an optional HyperbolicGromovProductUp observer handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedGromovHausdorff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6075.1, Definition 6075.2, and Definition 6075.3 seed the finite pointed compact-metric comparison packet over endpoints, basepoints, correspondence, distortion, Hausdorff-distance, observer handoff, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedGromovHausdorff}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.PointedGromovHausdorffUp.TasteGate.PointedGromovHausdorffTasteGate\_single\_carrier\_al`

Bridge status: none

Not claimed: This chapter does not close selected optimal pointed correspondences, quotient compacta, completed moduli spaces, host compact-metric equality, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, basepoint admission, pointed-correspondence admission, dyadic distortion exactness, basepoint preservation, comparison handoffs, transport, replay, provenance, and local naming for BEDC.Derived.PointedGromovHausdorffUp.

Chapter 6076

A Concrete Naming Certificate for $\mathsf{CauchySequenceComplete}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CauchySequenceComplete}^{\uparrow}$ packet records Cauchy-sequence completeness as finite BEDC NameCert data before any $\mathsf{Real}^{\uparrow}$ or complete-metric consumer receives an endpoint. It sits over the finite window discipline of chapter 126, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the ordered Cauchy completion boundary of chapter 3735. The packet names the direct Cauchy-sequence completeness surface; it is not a quotient of Cauchy streams, a selected host limit, countable choice, or an ambient completion primitive.

Definition 6076.1 (Cauchy sequence complete carrier)**.** *A* $\mathsf{CauchySequenceComplete}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, R, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *family of finite windows through which the sequence entries and tail comparisons are inspected.*
2. *M is the Cauchy-modulus ledger, recording the threshold assigned to each requested tolerance and the dominance checks needed by endpoint consumers.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the windows in S and the thresholds in M.*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the window, modulus, and regular readback rows have been displayed.*
5. *Q is the* $\mathsf{CompleteMetric}^{\uparrow}$ *endpoint row recording the finite completion-facing handoff visible after S, M, R, E.*
6. *H records componentwise* hsame *transport for the stream-window, modulus-ledger, readback, real-seal, complete-metric endpoint, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the route*

$$S, M \longmapsto R \longmapsto E \longmapsto Q.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchySequenceComplete}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchySequenceComplete}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, M, R, E, Q, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, countable choice, open-cover compactness, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySequenceCompleteUp.CauchySequenceCompleteTasteGate single carrier alignment`

Theorem 6076.2 (Cauchy sequence complete NameCert obligations)**.** *For every accepted* CauchySequenceComplete *carrier $K = (S, M, R, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchySequenceComplete}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite* $\mathsf{StreamName}^{\uparrow}$ *windows, Cauchy-modulus ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CompleteMetric}^{\uparrow}$ *endpoint row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A completion-facing consumer may read the endpoint row Q only after the same S, M, R, E route has been exposed.*

Proof. Project the carrier of Definition 6076.1. The accepted inventory is exactly $S, M, R, E, Q, H, C, P, N$, so every local obligation must be read from those rows.

The finite windows and Cauchy-modulus ledger are upstream. A consumer cannot inspect the regular rational readback R until the displayed window row S and threshold row M have supplied the requested tail data.

The endpoint handoff is downstream of the real seal. The row E seals the regular readback through $\mathsf{Real}^{\uparrow}$, and Q records only the finite complete-metric endpoint visible after the same route has been transported and replayed.

Since quotient equality, selected representatives, countable choice, open-cover compactness, host real equality, and ambient real completeness have no carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6076.3 (Cauchy sequence complete Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *endpoint read exported by an accepted* $\mathsf{CauchySequenceComplete}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The handoff preserves the finite window and Cauchy-modulus route before the $\mathsf{Real}^{\uparrow}$ *seal is exposed, and it is compatible with the ordered completion boundary of chapter 3735 without importing that chapter's order-comparison rows as hidden endpoint data.*

Proof. Start with an accepted carrier $K = (S, M, R, E, Q, H, C, P, N)$. By the carrier definition, the only real-facing seal is E, and E is downstream of the windows S, the modulus ledger M, and the regular readback R.

The complete-metric endpoint row Q is read only after E. Thus an endpoint consumer sees the same displayed sequence windows, thresholds, regular rational readback, real seal, and structural rows that generated the packet.

The sibling ordered-completeness chapter supplies a comparison boundary, but this carrier does not add an order row, quotient stream equality, selected limit, or host completeness datum. The real handoff therefore remains the direct finite Cauchy-sequence completeness surface named by $S, M, R, E, Q, H, C, P, N$. □

Falsifiable prediction: Any accepted $\mathsf{CauchySequenceComplete}^{\uparrow}$ endpoint consumer that reaches a $\mathsf{Real}^{\uparrow}$ or $\mathsf{CompleteMetric}^{\uparrow}$ read without the displayed StreamName windows, Cauchy-modulus

ledger, RegSeqRat readback, transport, replay, provenance, and local NameCert rows refutes the carrier.

Independence witness: StreamName, RegSeqRat, Real, CompleteMetric, and CauchySequence-OrderCompleteness are not individually bijective with $\mathsf{CauchySequenceComplete}^{\uparrow}$ because the packet jointly carries finite sequence windows, an explicit Cauchy-modulus ledger, regular readback, a Real seal, a complete-metric endpoint row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySequenceComplete}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySequenceComplete}^{\uparrow}$ packet is generated from finite StreamName windows, a Cauchy-modulus ledger, RegSeqRat readback, a RealUp seal, a CompleteMetric endpoint row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchySequenceComplete}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6076.1, Definition 6076.2, and Definition 6076.3 seed the finite constructive Cauchy-sequence completeness surface over StreamName, RegSeqRat, Real, CompleteMetric, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchySequenceComplete}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, open-cover compactness, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-modulus exposure, RegSeqRat readback exactness, Real-seal handoff, CompleteMetric endpoint exposure, transport, replay, provenance, and local naming for BEDC.Derived.CauchySequenceCompleteUp.

Chapter 6077

A Concrete Naming Certificate for $\mathsf{RiemannSumCauchyNet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ packet records the finite Cauchy-net bridge by which dyadic Riemann sums become completion-facing data without importing ambient real completeness. It sits after the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Cauchy boundary of chapter 3413, and the finite mesh discipline of chapter 6062. The packet names the finite net of dyadic Riemann-sum approximants; it is not a host integrability theorem, a selected limit, a quotient completion, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6077.1 (Riemann sum Cauchy-net carrier)**.** *A* $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, R, M, S, E, Q, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording the dyadic mesh depth, endpoint budget, and comparison scale used by the net;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at that dyadic scale;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows and dyadic tolerances;*
4. *M is the finite* $\mathsf{RiemannUniformMesh}^{\uparrow}$ *mesh row through which partition widths and tagged cells are inspected;*
5. *S is the dyadic Riemann-sum row, recording the finite sum visible after D, W, R, M are displayed;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the current finite approximation;*
7. *Q is the* $\mathsf{Cauchy}^{\uparrow}$ *comparison row, recording that two displayed mesh depths have a dyadic tolerance comparison inside the same finite packet;*
8. *H records componentwise* hsame *transport for the dyadic, window, readback, mesh, sum, Real, Cauchy, replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay of the route*

$$D, W, R, M \longmapsto S \longmapsto E \longmapsto Q;$$

10. *P is the* Pkg *provenance over* $\mathsf{RiemannSumCauchyNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RiemannUniformMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RiemannSumCauchyNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, M, S, E, Q, H, C, P, N$. It has no coordinate for a host integral, ambient real completeness, quotient equality of nets, selected limiting value, or unrestricted choice of partitions.

Lean4 ✓ `BEDC.Derived.RiemannSumCauchyNetUp.RiemannSumCauchyNetTasteGate single carrier alignment`

Theorem 6077.2 (Riemann sum Cauchy-net NameCert obligations)**.** *For every accepted* $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ *carrier $K = (D, W, R, M, S, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RiemannSumCauchyNet}^{\uparrow}}$ *surface consists exactly of carrier admission; dyadic tolerance exposure through D; finite-window inspection through W; regular rational readback through R; uniform mesh exposure through M; finite Riemann-sum handoff through S; terminal Real sealing through E; Cauchy comparison through Q; componentwise transport through H; displayed replay through C; package provenance through P; and local naming through N.*

Proof. Project the carrier of Definition 6077.1. The displayed rows are exactly $D, W, R, M, S, E, Q, H, C, P, N$, so each local obligation must be discharged by one of those finite coordinates.

The upstream route is dyadic and finite. The row D fixes the tolerance ledger, W opens only the requested finite windows, R records the regular rational readback, and M supplies the uniform mesh cells used by the Riemann-sum row.

The downstream route is sum, seal, and comparison. The row S exports only the finite dyadic Riemann sum, E seals the current approximation as $\mathsf{Real}^{\uparrow}$ data, and Q records the Cauchy comparison between displayed mesh depths.

Finally transport, replay, provenance, and local naming preserve the same packet through H, C, P, N. Since host integrals, quotient nets, selected limits, unrestricted partitions, and ambient real completeness are absent from the carrier, the listed rows exhaust the NameCert surface. □

Theorem 6077.3 (Riemann sum Cauchy-net Real-completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ *packet receives a finite handoff*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RiemannUniformMesh}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, Pkg, Na

The consumer may read the $\mathsf{Real}^{\uparrow}$ *seal only after the dyadic tolerance, finite-window, regular readback, uniform-mesh, and finite-sum rows have been displayed, and may read the* $\mathsf{Cauchy}^{\uparrow}$ *comparison only as a finite tolerance comparison between two displayed mesh depths. The handoff supplies no ambient completion of Riemann sums, selected integral value, quotient equality of Cauchy nets, or host real-completeness coordinate.*

Proof. Start with an accepted carrier $K = (D, W, R, M, S, E, Q, H, C, P, N)$. The Real-facing row E is downstream of the dyadic tolerance row D, finite window row W, regular readback row R, uniform mesh row M, and finite sum row S.

The Cauchy-facing row Q is also downstream of those displayed approximants. It records a finite tolerance comparison between two visible mesh depths, not a completed quotient net or a selected endpoint.

Use H, C, P, N to transport, replay, source, and name the same route. A completion-facing consumer therefore receives the finite dyadic Riemann-sum Cauchy-net packet and not an ambient real-completeness principle. □

Falsifiable prediction: If a completion-facing consumer reaches the $\mathsf{Real}^{\uparrow}$ seal E or the $\mathsf{Cauchy}^{\uparrow}$ comparison row Q without first displaying the dyadic tolerance, StreamName window, RegSeqRat readback, uniform mesh, finite-sum, transport, replay, provenance, and local NameCert rows, the $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ carrier has been misread.

Independence witness: dyadicratcore, streamname, regseqrat, riemann-uniform-mesh, real, and cauchy are not individually bijective with $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ because this packet jointly carries dyadic tolerances, finite observation windows, regular readback, uniform mesh rows, finite Riemann sums, Real seals, Cauchy comparisons, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannSumCauchyNet}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannSumCauchyNet}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, RiemannUniformMesh rows, a finite Riemann-sum row, a Real seal, a Cauchy comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannSumCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6077.1, Definition 6077.2, and Definition 6077.3 seed the finite dyadic Riemann-sum Cauchy-net bridge consumed by Real and completion routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannSumCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Riemann integrability, ambient Real completeness, quotient Cauchy nets, selected integral values, unrestricted partition choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window inspection, regular rational readback, uniform mesh exposure, finite-sum handoff, Real sealing, Cauchy comparison, transport, replay, provenance, and local naming for BEDC.Derived.RiemannSumCauchyNetUp.

Chapter 6078

A Concrete Naming Certificate for BoundedCauchySequence$^{\uparrow}$

Following Bishop and Bridges, a bounded Cauchy sequence is a rational-name object whose tail control and global bound are displayed before any completed real value is read. The BoundedCauchySequence$^{\uparrow}$ packet records this L10 route as finite NameCert data over the Cauchy request of chapter 3413, the StreamName windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal Real boundary of chapter 38. The packet supplies bounded Cauchy evidence without importing selected limits, quotient streams, countable choice, or ambient completeness.

Definition 6078.1 (Bounded Cauchy sequence carrier)**.** *A* BoundedCauchySequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (M, S, R, T, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the Cauchy modulus row, assigning each dyadic precision request a displayed tail index and the pairwise tail comparison rows certified by* Cauchy$^{\uparrow}$*.*
2. *S is the* StreamName$^{\uparrow}$ *finite-window ledger, listing exactly the source windows inspected by the modulus and bound requests.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row, converting the displayed StreamName windows into regular rational sequence entries.*
4. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying the dyadic scales used by the tail comparisons and the bound ledger.*
5. *L is the dyadic bound ledger, recording a displayed dyadic bound q and certified comparisons saying that every inspected prefix entry and every modulus-controlled tail entry remains below q in absolute value.*
6. *H records componentwise* hsame *transport for the modulus, stream, readback, tolerance, bound, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a precision or bound request through M, S, R, T, L, H, P, N.*
8. *P is the* Pkg *provenance over* BoundedCauchySequence$^{\uparrow}$*,* Cauchy$^{\uparrow}$*,* StreamName$^{\uparrow}$*,* RegSeqRat$^{\uparrow}$*,* DyadicRatCore$^{\uparrow}$*,* Real$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*

9. N is the local $\mathsf{NameCert}_{\mathsf{BoundedCauchySequence}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, S, R, T, L, H, C, P, N$. It has no coordinate for quotient equality of streams, a selected limit, host sequence equality, a supremum principle, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BoundedCauchySequenceUp.BoundedCauchySequenceTasteGate single carrier alignment`

Theorem 6078.2 (Bounded Cauchy NameCert obligations)**.** *For every accepted* $\mathsf{BoundedCauchySequence}^{\uparrow}$ *carrier*

$$B = (M, S, R, T, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedCauchySequence}^{\uparrow}}$ *surface consists exactly of Cauchy modulus admission for M, StreamName finite-window admission for S, RegSeqRat readback for R, DyadicRatCore tolerance visibility for T, dyadic bound-ledger enclosure for L, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

The enclosure is monotone in the displayed tail request: increasing the requested tail index may shrink the inspected tail window, but it cannot add a bound obligation outside the rows already carried by $M, S, R, T, L, H, C, P, N$.

Proof. Project the carrier of Definition 6078.1. The tuple $M, S, R, T, L, H, C, P, N$ is the whole accepted packet, so the local NameCert can read only the displayed modulus, window, readback, tolerance, bound, transport, replay, provenance, and naming rows.

The Cauchy route begins with a precision request. The modulus row M names the tail index, the StreamName row S supplies the finite windows inspected at that index, and the RegSeqRat row R reads those windows as rational entries.

The dyadic rows then supply both estimates. The tolerance row T fixes the scale for pairwise tail control, while the bound ledger L records the global dyadic enclosure for the displayed prefix and for all entries reached through the modulus-controlled tail.

If the tail index is increased, the packet replays the same route through C and transports it through H. The bound row is not enlarged by an external supremum or selected limit; it is only reread on a smaller displayed tail surface, with provenance and local naming carried by P, N.

Since no quotient stream, selected limit, host equality, countable choice, or completeness coordinate occurs in the carrier, no additional NameCert obligation can be exported. □

Theorem 6078.3 (Bounded Cauchy consumer route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or Completion consumer of an accepted* $\mathsf{BoundedCauchySequence}^{\uparrow}$ *packet receives bounded Cauchy evidence only through the finite route*

$$\mathsf{Cauchy}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{BoundedCauchySequence}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The consumer receives a displayed Cauchy modulus, finite rational windows, dyadic tolerances, a dyadic bound ledger, transport, replay, provenance, and local naming. It receives no selected real limit, quotient stream equality, ambient completeness principle, or host supremum.

Proof. Start with the consumer read. The only route to the bounded Cauchy evidence is the accepted carrier of Definition 6078.1.

Read M first to obtain the displayed Cauchy modulus, then S and R to expose the finite rational windows. The dyadic tolerance row T supplies the scale used by the tail comparison, and the bound ledger L supplies the displayed enclosure.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet for Real and Completion consumers. Because the carrier contains no coordinate for selected limits, quotient streams, host suprema, or ambient completeness, none of those objects can be imported by the consumer route. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedCauchySequence}^{\uparrow}$ packet is generated from a Cauchy modulus row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance row, a dyadic bound ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6078.1, Definition 6078.2, and Definition 6078.3 seed the bounded Cauchy route from finite stream windows and dyadic tolerance to Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host sequence equality, host suprema, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked Cauchy modulus admission, StreamName finite-window coverage, RegSeqRat readback, DyadicRatCore tolerance visibility, dyadic bound-ledger enclosure, transport stability, replay stability, and non-escape for BEDC.Derived.BoundedCauchySequenceUp.

Chapter 6079

A Concrete Naming Certificate for $\mathsf{CompactOscillationBarModulus}^{\uparrow}$

Following Bishop-style compact metric finite-net arguments, the $\mathsf{CompactOscillationBarModulus}^{\uparrow}$ packet realizes the compact-uniform-continuity route of chapter 9237 as a finite bar-modulus NameCert carrier. It reads compact metric finite nets from chapter 129, pointwise oscillation rows from chapter 130, dyadic tolerances from chapter 291, rational lower-bound folds from chapter 37, and the downstream uniform-modulus ledger of chapter 369. The packet records the bar-modulus handoff without importing open-cover compactness, semantic continuity, selected centers outside the finite bundle, or ambient completeness.

Definition 6079.1 (Compact fan-bar route)**.** *A* $\mathsf{CompactOscillationBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, F, D, R, B, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying the bundled centers and located coverage histories inspected by the compact request.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise oscillation row, recording the local radius at each bundled center and the target oscillation estimate used there.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, exposing the dyadic scale of the requested oscillation bound and the source-radius cuts read by the local estimates.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row, recording the positive finite minimum over the displayed local radius histories.*
5. *B is the compact bar row, the finite* $\mathsf{ProbeBundle}$ *recursion surface that orders the centers, local radii, dyadic cuts, and rational fold.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, exported only after K, F, D, R, B have been displayed.*
7. *H records componentwise* hsame *transport for the compact, oscillation, dyadic, rational-fold, bar, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through the compact net, pointwise oscillation rows, dyadic tolerance, rational lower-bound fold, compact bar, and uniform-modulus handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactOscillationBarModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactOscillationBarModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, D, R, B, U, H, C, P, N$. It has no coordinate for host open-cover compactness, a semantic continuity theorem, quotient point equality, a choice-selected center outside the displayed bar, or an external modulus selector.

Lean4 ✓ `BEDC.Derived.CompactOscillationBarModulusUp.CompactOscillationBarModulusTasteGate`
`single carrier alignment`

Theorem 6079.2 (Compact fan-bar modulus extraction)**.** *For every accepted* $\mathsf{CompactOscillationBarModulus}^{\uparrow}$ *carrier*

$$O = (K, F, D, R, B, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactOscillationBarModulus}^{\uparrow}}$ *surface extracts a uniform-modulus row exactly through compact finite-net admission for K, pointwise oscillation admission for F, dyadic tolerance visibility for D, rational lower-bound fold exactness for R, finite bar recursion for B, downstream handoff for U, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Consequently every $\mathsf{UniformModulus}^{\uparrow}$ *consumer receives a tolerance-only precision row through the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Rat}^{\uparrow},\ \mathsf{CompactOscillationBarModulus}^{\uparrow},\ \mathsf{UniformModulus}^{\uparrow}.$$

Proof. Project the carrier of Definition 6079.1. The rows $K, F, D, R, B, U, H, C, P, N$ are the whole accepted packet, so every exported uniform-modulus read must pass through them.

The compact row K and the bar row B supply the finite ProbeBundle of centers. The pointwise oscillation row F attaches a local source radius and a target oscillation estimate to each displayed center.

The dyadic row D fixes the tolerance scale. The rational fold row R then records a positive lower bound over the finitely many local radius histories, producing a displayed precision history rather than an external selector.

The handoff row U receives only that folded precision and the visible oscillation estimates. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route.

Since the carrier has no coordinate for host open-cover compactness, semantic continuity, quotient point equality, selected centers outside the bar, or ambient completeness, the extracted uniform-modulus row cannot import those objects. □

Theorem 6079.3 (Compact oscillation bar consumer boundary)**.** *Every consumer of an accepted* $\mathsf{CompactOscillationBarModulus}^{\uparrow}$ *packet reads compact-uniform-continuity data as a finite oscillation bar. The consumer may inspect the compact finite net, pointwise oscillation rows, dyadic tolerance ledger, rational lower-bound fold, compact bar recursion, uniform-modulus handoff, transport, replay, provenance, and local naming rows, and nothing else.*

Proof. The consumer begins with the accepted carrier. By Definition 6079.1, the only analytic rows are K, F, D, R, B, U and the only structural rows are H, C, P, N.

Reading K and B exposes the finite bar. Reading F attaches local oscillation control to each center, while D and R convert the requested dyadic tolerance into a displayed lower-bound precision.

The row U is downstream of those data and is the only exported uniform-modulus coordinate. The structural rows do not add compactness, continuity, quotient, choice, or completeness objects; they only transport, replay, source, and name the same packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactOscillationBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactOscillationBarModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ContinuousMap pointwise oscillation rows, a DyadicRatCore tolerance row, a RatUp lower-bound fold, finite ProbeBundle bar recursion, a UniformModulusUp handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactOscillationBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6079.1, Definition 6079.2, and Definition 6079.3 seed the compact oscillation bar route from finite compact nets and local oscillation rows to UniformModulusUp consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactOscillationBarModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not close host open-cover compactness, semantic continuity, quotient point equality, choice-selected centers outside the displayed bar, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact finite-net admission, pointwise oscillation admission, dyadic tolerance visibility, rational lower-bound fold exactness, finite bar recursion, uniform-modulus handoff, transport stability, replay stability, and non-escape for BEDC.Derived.CompactOscillationBarModulusUp.

Chapter 6080

A Concrete Naming Certificate for $\mathsf{EnrichedYoneda}^{\uparrow}$

Following Kelly's enriched category theory and Mac Lane's categorical account of representability, the $\mathsf{EnrichedYoneda}^{\uparrow}$ packet records the enriched functor-of-points surface as finite NameCert data. It sits over the category carrier of chapter 61, the ordinary functor and natural-transformation rows of chapter 62 and chapter 63, the enriched hom-object surface of chapter 182, and the monoidal tensor boundary of chapter 181. The packet names the finite route by which enriched representables, enriched naturality, and monoidal coherence can be replayed without importing a host enriched presheaf category, ambient ends, quotient functor categories, or external universality.

Definition 6080.1 (EnrichedYoneda seed carrier)**.** *An* $\mathsf{EnrichedYoneda}^{\uparrow}$ *seed carrier is a finite* BHist *packet*

$$Y = (C, V, O, H, F, S, U, Q, T, R, P, N)$$

with the following displayed rows:

1. *C is the consumed* $\mathsf{Cat}^{\uparrow}$ *object-and-hom boundary for the underlying ordinary category read by downstream categorical consumers;*
2. *V is the consumed* $\mathsf{EnrichedCat}^{\uparrow}$ *row, including the object boundary, enriched hom-object rows, identity rows, and enriched composition rows;*
3. *O records the chosen object row whose enriched representable is being named;*
4. *H is the enriched hom-object row assigning to each displayed object x the hom-object $V(x, O)$ in the consumed monoidal category;*
5. *F is the enriched functor-of-points row, read as the object-indexed assignment $x \mapsto V(x, O)$ together with its displayed action on enriched hom rows;*
6. *S records enriched naturality-square rows comparing the action of F against enriched composition in V;*
7. *U records monoidal-unit rows and the coherence rows needed to read identities, associators, left unitors, and right unitors through the consumed* $\mathsf{MonoidalCat}^{\uparrow}$ *boundary;*
8. *Q records componentwise* hsame *transport for the ordinary category, enriched category, chosen object, hom-object, functor-of-points, naturality, monoidal coherence, provenance, and local naming rows;*

9. *T records displayed* Cont *replay from the chosen object through enriched hom-object inspection, functor-of-points readback, naturality-square comparison, and monoidal coherence;*
10. *R is the finite readback ledger by which ordinary* $\mathsf{Functor}^{\uparrow}$ *and* $\mathsf{NatTrans}^{\uparrow}$ *consumers may read the enriched rows without receiving host functor-category equality;*
11. *P is the* Pkg *provenance over* $\mathsf{EnrichedYoneda}^{\uparrow}$, $\mathsf{Cat}^{\uparrow}$, $\mathsf{Functor}^{\uparrow}$, $\mathsf{NatTrans}^{\uparrow}$, $\mathsf{EnrichedCat}^{\uparrow}$, $\mathsf{MonoidalCat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{EnrichedYoneda}^{\uparrow}}$ *row.*

The classifier relates two accepted packets exactly when these rows transport componentwise through Q and replay through the same displayed routes in T. It does not contain a coordinate for a host enriched presheaf category, external end calculus, quotient natural-transformation equality, global enriched embedding theorem, or ambient universal property.

Lean4 (definition) `BEDC.Derived.EnrichedYonedaUp.EnrichedYonedaUp`

Closure status for $\mathsf{NameCert}_{\mathsf{EnrichedYoneda}^{\uparrow}}$.

Constructive story: An $\mathsf{EnrichedYoneda}^{\uparrow}$ packet is generated from a consumed CategoryUp boundary, FunctorUp and NatTransUp consumer rows, an EnrichedCatUp hom-object surface, a MonoidalCatUp tensor-coherence boundary, a chosen object, enriched functor-of-points rows, naturality squares, componentwise transport, displayed replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EnrichedYoneda}^{\uparrow}}$, seed)

Scope closed: Definition 6080.1 seeds the finite enriched functor-of-points certificate over CategoryUp, FunctorUp, NatTransUp, EnrichedCatUp, MonoidalCatUp, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EnrichedYoneda}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: The chapter does not close enriched Yoneda equivalence, enriched presheaf categories, end calculus, quotient functor-category equality, host universality, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, enriched hom-object exactness, functor-of-points readback, enriched naturality, monoidal coherence, transport replay, provenance, and non-escape for the same packet.

Chapter 6081

A Concrete Naming Certificate for $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$ packet records the constructive cut-facing real boundary where an upper Dedekind observation is read with explicit located comparison data before a regular-Cauchy or real-seal consumer treats it as real evidence. It sits beside the $\mathsf{Real}^{\uparrow}$ completion surface of chapter 38, the rounded cut rows of chapter 436, the one-sided upper ledger of chapter 1447, the lower comparison ledger of chapter 1446, and the located real interface of **??**. It also points toward the representation comparisons of chapter 2346 and chapter 5908 without identifying their carriers.

Definition 6081.1 (Located upper Dedekind real carrier)**.** *A* $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (D, A, L, O, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *row, read only as rounded lower-and-upper rational observation data;*
2. *A is the* $\mathsf{UpperReal}^{\uparrow}$ *row, recording the upper-side rational or dyadic observations that the consumer may inspect;*
3. *L is the* $\mathsf{LowerReal}^{\uparrow}$ *comparison row, used only to display lower-upper separation when a locatedness request asks for it;*
4. *O is the locatedness row: for each displayed rational interval $a < b$, it records either a lower-side observation below b or an upper-side observation above a inside the same finite history;*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, A, L, O have been displayed;*
6. *H is componentwise* hsame *transport for the cut row, upper row, lower comparison row, locatedness row, Real seal, and structural rows;*
7. *C is the displayed* Cont *replay route for rereading the same cut-facing upper observations;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperDedekindReal}^{\uparrow}}$ *row.*

The carrier exports no quotient class of cuts, no selected representative, no trichotomy branch, no countable-choice selector, no Dedekind completeness theorem, and no completeness claim for $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ BEDC.Derived.LocatedUpperDedekindRealUp.TasteGate.LocatedUpperDedekindRealTasteGate
single carrier alignment

Theorem 6081.2 (Located upper Dedekind real NameCert obligations)**.** *For an accepted* LocatedUpperDedekindRe *carrier* $U = (D, A, L, O, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperDedekindReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed* $\mathsf{DedekindCut}^{\uparrow}$ *row,* $\mathsf{UpperReal}^{\uparrow}$ *row,* $\mathsf{LowerReal}^{\uparrow}$ *comparison row, locatedness row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *upper cut observation, stating that every upper-side read factors through* A *together with the upper side of* D*;*
3. *located comparison, stating that every requested rational interval* $a < b$ *is answered by the displayed row* O *together with the comparison row* L *and the upper row* A*;*
4. *Real-seal boundary, stating that a* $\mathsf{Real}^{\uparrow}$*-facing consumer reaches* E *only after the cut, upper, lower, and locatedness rows have been inspected;*
5. *representation restraint, stating that Cauchy-Dedekind bridge or equivalence consumers may cite this packet only as a finite cut-facing located upper surface.*

No obligation introduces quotient stream equality, quotient cut equality, selected representatives, trichotomy, countable choice, Dedekind completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6081.1. The displayed packet contains the cut row, upper row, lower comparison row, locatedness row, Real seal, and structural rows as finite BHist data.

The upper cut observation obligation reads D and A together. The cut row supplies rounded lower-and-upper rational observations, while the upper row exposes only the upper-side candidates and their finite comparison budgets.

The located comparison obligation is the role of O. For a displayed interval $a < b$, the packet answers through the same finite lower and upper rows, so a consumer receives located evidence without a trichotomy branch or an external order oracle.

The Real-seal boundary is E. Since E is downstream of D, A, L, O and is transported only by H and replayed only by C, a $\mathsf{Real}^{\uparrow}$-facing read cannot bypass the finite cut-facing route.

Finally the provenance row P names $\mathsf{LocatedReal}^{\uparrow}$ and the Cauchy-Dedekind comparison siblings only as dependency anchors. The packet therefore supports later bridge comparison while adding no quotient equality, selected representative, countable-choice selector, Dedekind completeness theorem, or completeness claim for $\mathsf{Real}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUpperDedekindReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$ packet is generated from a DedekindCut row, an UpperReal row, a LowerReal comparison row, a locatedness row, a RealUp seal, componentwise transports, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedUpperDedekindReal}^{\uparrow}}$, seed)

Scope closed: Definition 6081.1 and Definition 6081.2 seed the finite cut-facing located upper-real surface for Real-completion and Cauchy-Dedekind comparison consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedUpperDedekindReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient classes, trichotomy, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires carrier admission, upper cut observation, located comparison, Real-seal boundary, and representation restraint for the displayed packet, with future Lean work under BEDC.Derived.LocatedUpperDedekindRealUp.

Chapter 6082

A Concrete Naming Certificate for $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$ packet records the constructive lower-cut real boundary where a lower Dedekind observation is read with explicit located comparison data before a regular-Cauchy or real-seal consumer treats it as real evidence. It is the lower-side sibling of chapter 6081; it also depends on the dyadic arithmetic ledger of chapter 291, the finite observation windows of chapter 126, the regular rational handoff surface of chapter 301, and the terminal real-name seal of chapter 38. The packet displays a finite L10 lower-cut interface; it does not assume quotient cuts, selected representatives, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6082.1 (Located lower Dedekind real carrier)**.** *A* $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, B, U, O, Y, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DedekindCut}^{\uparrow}$ *row, read only as rounded lower-and-upper rational observation data;*
2. *B is the* $\mathsf{LowerReal}^{\uparrow}$ *row, recording the lower-side rational or dyadic observations that the consumer may inspect;*
3. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *comparison row, used only to display lower-upper separation when a locatedness request asks for it;*
4. *O is the locatedness row: for each displayed rational interval $a < b$, it records either a lower-side observation below b or an upper-side observation above a inside the same finite history;*
5. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row carrying the dyadic comparison budgets used when lower observations are sampled at finite precision;*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row recording which lower-cut observations are inspected before any sequence readback occurs;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row attached to the same windows and dyadic tolerances;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, B, U, O, Y, W, R have been displayed;*
9. *H is componentwise* hsame *transport for the cut row, lower row, upper comparison row, locatedness row, dyadic budget, stream window, regular handoff, Real seal, and structural rows;*

10. *C is the displayed* Cont *replay route for rereading the same lower-cut observations;*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$, $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$, $\mathsf{DedekindCut}^{\uparrow}$, $\mathsf{LowerReal}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLowerDedekindReal}^{\uparrow}}$ *row.*

The carrier exports no quotient class of cuts, no selected cut representative, no order trichotomy branch, no countable-choice selector, no Dedekind completeness theorem, and no completeness claim for $\mathsf{Real}^{\uparrow}$*.*

Theorem 6082.2 (Located lower Dedekind real NameCert obligations)**.** *For an accepted* LocatedLowerDedekindRea *carrier $L = (D, B, U, O, Y, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedLowerDedekindReal}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed* $\mathsf{DedekindCut}^{\uparrow}$ *row,* $\mathsf{LowerReal}^{\uparrow}$ *row,* $\mathsf{UpperReal}^{\uparrow}$ *comparison row, locatedness row, dyadic budget, stream windows, regular handoff,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*

2. *lower cut observation, stating that every lower-side read factors through B together with the lower side of D;*

3. *located comparison, stating that every requested rational interval $a < b$ is answered by the displayed row O together with the lower row B and comparison row U;*

4. *finite sampling discipline, stating that every lower observation used for a sequence handoff passes through the dyadic budget Y and the stream window row W;*

5. *Real-seal boundary, stating that a* $\mathsf{Real}^{\uparrow}$*-facing consumer reaches E only after the cut, lower, upper-comparison, locatedness, dyadic, stream-window, and regular-handoff rows have been inspected;*

6. *representation restraint, stating that the lower packet may be paired with the upper packet only as a finite cut-facing comparison surface.*

No obligation introduces quotient stream equality, quotient cut equality, selected representatives, trichotomy, countable choice, Dedekind completeness, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6082.1. The displayed packet contains the cut row, lower row, upper comparison row, locatedness row, dyadic budget, stream windows, regular handoff, Real seal, and structural rows as finite BHist data.

The lower cut observation obligation reads D and B together. The cut row supplies rounded lower-and-upper rational observations, while the lower row exposes only the lower-side candidates and their finite comparison budgets.

The located comparison obligation is the role of O. For a displayed interval $a < b$, the packet answers through the same finite lower and upper rows, so a consumer receives located evidence without a trichotomy branch or an external order oracle.

The finite sampling discipline is the route through Y and W. A lower observation that will feed a regular sequence is first tied to dyadic precision data and then to a displayed finite window, so the handoff cannot be read from an unbounded stream or a quotient cut.

The Real-seal boundary is E. Since E is downstream of D, B, U, O, Y, W, R and is transported only by H and replayed only by C, a $\mathsf{Real}^{\uparrow}$-facing read cannot bypass the finite lower-cut route.

Finally the provenance row P names $\mathsf{LocatedUpperDedekindReal}^{\uparrow}$ only as the sibling comparison surface. The packet therefore supports later Cauchy-Dedekind comparison while adding no quotient equality, selected representative, countable-choice selector, Dedekind completeness theorem, or completeness claim for $\mathsf{Real}^{\uparrow}$. □

Theorem 6082.3 (Located lower Dedekind real regular-sequence handoff)**.** *Let $L = (D, B, U, O, Y, W, R, E, H, C, P$ be an accepted* $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$ *carrier. Every downstream* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer read of the handoff row R factors through the finite route*

$$D, B, U, O, Y, W, R$$

inside the same BHist *packet, and the terminal seal E may be read only after that route has been displayed. Consequently the handoff exposes lower-cut sampling data, dyadic precision budgets, and finite stream windows, but it exposes no quotient cut, no selected real representative, no ambient completeness principle, and no order-trichotomy decision.*

Proof. Begin with the NameCert obligations of Definition 6082.2. They force every lower-side observation to read the lower row B together with the cut row D, and they force every located comparison to use the displayed row O rather than an external order oracle.

Next use the finite sampling discipline in the same obligation list. Before R can be consumed as a $\mathsf{RegSeqRat}^{\uparrow}$ handoff, the dyadic budget Y and the finite window row W must already have been displayed. Thus the sequence-facing read is tied to finite lower-cut observations rather than to an unbounded stream.

Finally read the Real-seal boundary. The seal row E is downstream of D, B, U, O, Y, W, R, and the structural rows H, C, P, N only transport, replay, name, and package the same displayed route. The listed quotient, representative, completeness, and trichotomy readings are outside the carrier inventory and cannot be consumed here. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLowerDedekindReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedLowerDedekindReal}^{\uparrow}$ packet is generated from a DedekindCut row, a LowerReal row, an UpperReal comparison row, a locatedness row, a DyadicRatCore budget, StreamName windows, a RegSeqRat handoff, a RealUp seal, componentwise transports, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedLowerDedekindReal}^{\uparrow}}$, seed)

Scope closed: Definition 6082.1, Definition 6082.2, and Definition 6082.3 seed the finite lower-cut surface for regular-sequence and real-completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedLowerDedekindReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Dedekind completeness, quotient classes, trichotomy, countable choice, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, lower cut observation, located comparison, finite sampling discipline, Real-seal boundary, and representation restraint for the displayed packet, with future Lean work under BEDC.Derived.LocatedLowerDedekindRealUp.

Chapter 6083

A Concrete Naming Certificate for $\mathsf{DilworthDecomposition}^{\uparrow}$

Following Dilworth, the $\mathsf{DilworthDecomposition}^{\uparrow}$ packet records the finite-order decomposition theorem as a BEDC NameCert carrier: a finite $\mathsf{Poset}^{\uparrow}$ row, an antichain-width witness, a chain-cover ledger, and a matching or network-flow dual certificate are read as one finite certificate surface. It is anchored in the partial-order carrier of chapter 53, the bipartite matching surface of chapter 234, and the max-flow/min-cut route of chapter 233. The packet exports only displayed finite rows, not a host poset, not quotient order equality, and not a selected global decomposition outside the ledger.

Definition 6083.1 (Dilworth decomposition carrier)**.** *A* $\mathsf{DilworthDecomposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (O, A, K, M, F, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite* $\mathsf{Poset}^{\uparrow}$ *order row, including the carried elements, the displayed order comparison ledger, and the classifier used to compare repeated element reads.*
2. *A is the antichain-width witness row, recording a displayed antichain spine and a finite proof ledger that no larger displayed antichain is admitted by the same* $\mathsf{Poset}^{\uparrow}$ *order rows.*
3. *K is the chain-cover row, recording a finite list of chains whose union covers the carried elements and whose internal comparisons are justified by O.*
4. *M is the* $\mathsf{Matching}^{\uparrow}$ *dual-certificate row for the standard bipartite split graph of the finite order, reading augmenting-path and cover data only through the matching packet.*
5. *F is the optional* $\mathsf{NetworkFlow}^{\uparrow}$ *dual-certificate row for the same split graph, exposing the max-flow/min-cut handoff as a second route to the same width-cover equality.*
6. *H is componentwise* hsame *transport for the order row, antichain row, chain-cover row, matching or flow certificate rows, replay, provenance, and local naming.*
7. *C is the displayed* Cont *replay route*

$$O, A, K, M, F \longmapsto H \longmapsto P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{DilworthDecomposition}^{\uparrow}$*,* $\mathsf{Poset}^{\uparrow}$*,* $\mathsf{Matching}^{\uparrow}$*,* $\mathsf{NetworkFlow}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

9. *N is the local* $\mathsf{NameCert}_{\mathsf{DilworthDecomposition}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient host partial order, quotient order equality, an unlisted chain cover, a selected maximum antichain outside A, countable choice, or an infinite decomposition principle.

Lean4 ✓ `BEDC.Derived.DilworthDecompositionUp.DilworthDecompositionTasteGate single carrier alignment`

Theorem 6083.2 (Dilworth decomposition NameCert obligations)**.** *For an accepted* $\mathsf{DilworthDecomposition}^{\uparrow}$ *carrier $D = (O, A, K, M, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DilworthDecomposition}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the finite* $\mathsf{Poset}^{\uparrow}$ *order row, antichain-width witness row, chain-cover row, dual certificate rows, transport, replay, provenance, and local naming;*
2. *order exposure, requiring every chain, antichain, matching, and flow read to use the same displayed O row and its classifier;*
3. *antichain-width exposure, requiring width claims to be read through A and not through an external cardinality oracle;*
4. *chain-cover exposure, requiring every cover claim to be read through the listed chain ledger K;*
5. *dual-certificate handoff, requiring the equality between width and minimum chain cover size to factor through the* $\mathsf{Matching}^{\uparrow}$ *row M or the* $\mathsf{NetworkFlow}^{\uparrow}$ *row F;*
6. *structural replay, requiring* hsame, Cont, Pkg, *and local* NameCert *rows to preserve the same finite decomposition route;*
7. *nonescape, excluding host posets, quotient order equality, unlisted chain covers, selected maximum antichains outside the carrier, countable choice, and infinite-order decomposition data.*

No $\mathsf{DilworthDecomposition}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Project the carrier of Definition 6083.1. The finite order is O, the width witness is A, the chain-cover ledger is K, and the dual certificate rows are M and F.

The order exposure obligation is rowwise. Since the antichain, chain-cover, matching, and flow rows are all carried over O, a consumer cannot combine the antichain from one order with the chain cover or split-graph certificate from another.

The width and cover obligations are the two visible halves of the decomposition. The row A supplies the displayed antichain-width witness, while K supplies the displayed chain cover and its coverage ledger.

The dual-certificate handoff is the finite proof route. A matching-facing consumer reads M through chapter 234; a flow-facing consumer reads F through chapter 233. Both routes are tied back to the same order row O and the same width and cover rows A, K.

Finally H, C, P, N transport, replay, record provenance for, and name exactly this finite packet. Since the carrier fields are exhausted by $O, A, K, M, F, H, C, P, N$, the excluded host order, quotient equality, unlisted cover, selected antichain, choice, and infinite decomposition principles have no row in the certificate. □

Theorem 6083.3 (Dilworth chain-antichain handoff)**.** *Let $D = (O, A, K, M, F, H, C, P, N)$ be an accepted* $\mathsf{DilworthDecomposition}^{\uparrow}$ *carrier. Every chain-cover or antichain-width consumer exported by D factors through the displayed route*

$$\mathsf{Poset}^{\uparrow},\ \mathsf{Matching}^{\uparrow},\ \mathsf{NetworkFlow}^{\uparrow}$$

by reading $O, A, K, M, F, H, C, P, N$ in order. The route identifies the width row and the chain-cover row only through the displayed matching or network-flow dual certificate, and it exports no host summability of sizes, quotient order equality, selected external cover, countable choice, or infinite poset theorem.

Proof. First read the finite order row. The $\mathsf{Poset}^{\uparrow}$ packet supplies the carried elements, comparison ledger, and classifier needed before either an antichain or a chain can be inspected.

Next read the two decomposition rows. The antichain row A records the width witness, and the chain-cover row K records the finite cover together with its internal comparison checks.

The equality route is then supplied by the dual certificate. The matching row M exposes the bipartite split-graph handoff, while the network-flow row F exposes the equivalent cut route. A consumer may use either displayed certificate, but both remain downstream of O, A, K.

Transport by H, replay by C, and packaging by P, N preserve this same finite route. Therefore the public handoff is the listed BEDC packet and not an ambient Dilworth theorem over a host poset. □

Closure status for $\mathsf{NameCert}_{\mathsf{DilworthDecomposition}^{\uparrow}}$.

Constructive story: A $\mathsf{DilworthDecomposition}^{\uparrow}$ packet is generated from a finite Poset row, an antichain-width witness, a chain-cover ledger, Matching and NetworkFlow dual-certificate rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DilworthDecomposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6083.1, Definition 6083.2, and Definition 6083.3 seed the finite chain-antichain decomposition certificate over Poset, Matching, and NetworkFlow rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DilworthDecomposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host partial orders, quotient order equality, selected external chain covers, countable choice, infinite posets, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, order exposure, antichain-width exposure, chain-cover exposure, dual-certificate handoff, transport, replay, provenance, and local naming for BEDC.Derived.DilworthDecompositionUp.

$\mathsf{PointwiseStreamName}^{\uparrow}$ routes the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126 into the $\mathsf{Real}^{\uparrow}$-facing seal of chapter 38 through the child file below.

Chapter 6084

A Concrete Naming Certificate for $\mathsf{PointwiseStreamName}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{PointwiseStreamName}^{\uparrow}$ packet records the traditional countable pointwise name as finite BEDC NameCert data. It is the countable-name neighbour of the $\mathsf{StreamName}^{\uparrow}$ finite-window surface of chapter 126; its point values are read through the dyadic ledger of chapter 291, the regular-rational gateway of chapter 301, and the terminal real-name seal of chapter 38.

Definition 6084.1 (Pointwise stream name carrier)**.** *A* $\mathsf{PointwiseStreamName}^{\uparrow}$ *carrier is a finite NameCert packet*

$$Q = (S, I, D, R, E, H, C, P, N)$$

with the following displayed coordinates:

1. *S is the certified* $\mathsf{StreamName}^{\uparrow}$ *source row.*
2. *I is the finite ledger of requested point indices and observation windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for the inspected point values.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row generated from the pointwise observations.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal available only after the regular-rational route has been displayed.*
6. *H stores componentwise* hsame *transport for sources, indices, windows, dyadic ledgers, readbacks, seals, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a consumer request through S, I, D, R, E, H, P, N.*
8. *P is* Pkg *provenance over* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{PointwiseStreamName}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these coordinates. It contains no quotient stream equality, selected limit, host real equality, countable-choice row, or $\mathsf{Real}^{\uparrow}$ *completeness assertion.*

Theorem 6084.2 (Pointwise stream name finite read)**.** *Every accepted* $\mathsf{PointwiseStreamName}^{\uparrow}$ *read factors through the certified source S, requested index ledger I, dyadic tolerance row D, regular-rational readback R, and terminal* $\mathsf{Real}^{\uparrow}$ *seal E before any completion-facing consumer may inspect it. The route is a pointwise countable-name source for* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*, not a quotient stream, selected limit, host equality, or completeness datum.*

Proof. Project the carrier of Definition 6084.1. The request has only the certified stream source, index ledger, dyadic row, regular readback, Real seal, transport, replay, provenance, and local naming coordinates available.

The read begins at S and I. A consumer must open the finite BHist windows for the requested point indices before it can inspect dyadic or regular-rational data.

The dyadic, regular, and Real rows are then consumed in order. Since no coordinate stores quotient equality, a selected limit, host equality, countable choice, or Real completeness, the pointwise read cannot export those data. □

Theorem 6084.3 (Pointwise stream name transport stability)**.** *Let Q_0 and Q_1 be accepted* $\mathsf{PointwiseStreamName}^{\uparrow}$ *carriers. If a displayed* hsame *route transports their certified source rows, requested index ledgers, dyadic tolerance rows, regular-rational readbacks, Real seals, replay rows, provenance rows, and local naming rows componentwise, then every pointwise read accepted for Q_0 is accepted for Q_1 with the transported finite route.*

Proof. Read the componentwise transport coordinate H in the two carriers. It identifies only the displayed StreamName source rows, requested windows, dyadic ledgers, regular readbacks, Real seals, replay rows, provenance rows, and local naming rows.

Replay the consumer request through C. The replay preserves the order of the pointwise route, so a transported read still reaches $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ only after the finite StreamName source and requested windows have been displayed.

The carrier inventory supplies the exclusion. Quotient streams, selected representatives, host real equality, countable-choice schedules, and completeness principles are not coordinates of either carrier. □

Theorem 6084.4 (Pointwise stream index stability)**.** *Let Q_0 and Q_1 be accepted* $\mathsf{PointwiseStreamName}^{\uparrow}$ *carriers. Suppose their certified* $\mathsf{StreamName}^{\uparrow}$ *source rows and requested index ledgers are componentwise* hsame*, and suppose the displayed transport also preserves the dyadic tolerance rows, regular-rational readbacks, Real seals, replay rows, provenance rows, and local naming rows. Then every requested finite* $\mathsf{StreamName}^{\uparrow}$ *window in Q_0 reaches the same ordered route through* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{Real}^{\uparrow}$ *in Q_1.*

Proof. Project the two carriers by Definition 6084.1. The only index-facing coordinates are the certified source row S and finite requested-index ledger I.

Apply transport stability from Definition 6084.3. The componentwise hsame route carries S, I, D, R, E, C, P, N together, so the accepted pointwise read is replayed with the same source windows and requested indices.

Now use finite read from Definition 6084.2. It forces every requested window to pass first through the StreamName source and index ledger, then through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and finally the $\mathsf{Real}^{\uparrow}$ seal. Since transport preserves those rows in order, the requested-index route is stable. □

Theorem 6084.5 (Pointwise stream name L10 gateway)**.** *The* $\mathsf{PointwiseStreamName}^{\uparrow}$ *packet is consumed by the L10 Real and Completion route only as a finite pointwise source: it sends certified windows to* $\mathsf{StreamName}^{\uparrow}$*, dyadic tolerance rows to* $\mathsf{DyadicRatCore}^{\uparrow}$*, regular readback rows to* $\mathsf{RegSeqRat}^{\uparrow}$*, and terminal seal rows to* $\mathsf{Real}^{\uparrow}$*. The gateway does not assert equality of quotient streams, choose a limiting real, identify host real objects, or prove* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Combine Definition 6084.2 with the transport stability of Definition 6084.3. These two rows force every L10 read to pass through the same displayed source, index, dyadic, regular, Real, transport, replay, provenance, and naming coordinates.

The sibling chapters named in the opening paragraph supply the downstream rows. $\mathsf{StreamName}^{\uparrow}$ supplies finite windows, $\mathsf{DyadicRatCore}^{\uparrow}$ supplies tolerance data, $\mathsf{RegSeqRat}^{\uparrow}$ supplies regular readback, and $\mathsf{Real}^{\uparrow}$ supplies the terminal seal.

Since the carrier lacks quotient equality, selected limits, host equality, choice, and completeness coordinates, the L10 gateway exports only the finite pointwise NameCert route. □

Theorem 6084.6 (Pointwise stream name L10 index determinacy)**.** *Let Q_0 and Q_1 be accepted* $\mathsf{PointwiseStreamName}^{\uparrow}$ *carriers. If two L10 reads agree on the displayed* $\mathsf{StreamName}^{\uparrow}$ *source row, the requested index ledger, and the finite window schedule, then both reads route through the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row before any consumer can inspect Real-facing data.*

Proof. Begin with finite read from Definition 6084.2. It forces each L10 read to expose the source row, requested index ledger, and finite windows before dyadic, regular-rational, or Real-facing data are available.

Apply transport stability from Definition 6084.3. Agreement on the displayed source and window route transports the dyadic tolerance row, readback row, seal row, replay row, provenance row, and local naming row componentwise.

Use index stability from Definition 6084.4. The shared requested index and finite window schedule determine the ordered route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$.

Finally apply the L10 gateway of Definition 6084.5. A Real or Completion consumer can inspect only the route already determined by the source, index, window, dyadic, readback, seal, transport, replay, provenance, and naming rows. □

Theorem 6084.7 (Pointwise stream vision transport)**.** *Every vision-facing* $\mathsf{PointwiseStreamName}^{\uparrow}$ *read consumed by the L10 Real and Completion route must pass through the certified* $\mathsf{StreamName}^{\uparrow}$ *source row, requested index ledger, finite window schedule,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row already exposed by the pointwise finite-read, transport-stability, and L10 index-determinacy routes. The transported read exports no quotient stream equality, selected limit, host real equality, countable-choice schedule, or* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Begin with pointwise finite read, Definition 6084.2. It forces the vision-facing read to expose the certified source row, requested index ledger, dyadic tolerance row, regular-rational readback, and terminal $\mathsf{Real}^{\uparrow}$ seal before any Real or Completion consumer inspects the pointwise name.

Apply transport stability from Definition 6084.3. Componentwise hsame carries the source, index, dyadic, readback, seal, replay, provenance, and local naming rows together, so the transported vision read remains the same finite NameCert route.

Use L10 index determinacy, Definition 6084.6. Agreement on the displayed source, requested index ledger, and finite window schedule determines the ordered route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before any Real-facing data are available.

The exclusions follow from the same carrier inventory. No row stores quotient stream equality, selected limits, host real equality, countable-choice schedules, or $\mathsf{Real}^{\uparrow}$ completeness, so the vision-facing transport cannot export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{PointwiseStreamName}^{\uparrow}}$.

Constructive story: A $\mathsf{PointwiseStreamName}^{\uparrow}$ packet is generated from a certified StreamName source, requested point windows, a dyadic tolerance ledger, regular-rational readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointwiseStreamName}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6084.1, Definition 6084.2, Definition 6084.3, and Definition 6084.5 seed the finite pointwise source route for StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, RealUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointwiseStreamName}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host real equality, RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, pointwise finite-read exposure, transport stability, L10 gateway coverage, and non-escape for BEDC.Derived.PointwiseStreamNameUp.

Chapter 6085

A Concrete Naming Certificate for $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$

The $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ packet realizes the normalization audit map of chapter 9383 as a finite candidate-mediated strong-normalization handoff. It sits beside the finite-window diamond envelope of chapter 6063, the subject-reduction obstruction of chapter 731, the confluence audit packet of chapter 735, and the typed conversion residual frontier of chapter 6069. The packet records the supplied candidate row, closed typed term row, no-infinite-reduction projection, obstruction handoff, confluence boundary, transport, replay, provenance, and local naming data without claiming unconditional strong normalization for all closed MetaCIC terms.

Definition 6085.1 (MetaCIC candidate SN handoff carrier)**.** *A* $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, A, S, N, O, D, F, H, C, P, L)$$

with the following displayed coordinates:

1. *T is the closed typed MetaCIC term row.*
2. *A is the candidate or candidate-set membership row supplied to the SN projection.*
3. *S is the candidate-mediated strong-normalization projection row.*
4. *N is the no-infinite-reduction row obtained from the same candidate evidence.*
5. *O is the subject-reduction obstruction handoff row.*
6. *D is the finite-window diamond-envelope row.*
7. *F is the confluence audit and typed conversion frontier row.*
8. *H records componentwise* hsame *transport for term, candidate, SN, no-infinite-reduction, obstruction, confluence, frontier, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a normalization-audit query through $T, A, S, N, O, D, F, H, P, L$.*
10. *P is* Pkg *provenance over the normalization audit map, the finite-window diamond envelope, the subject-reduction obstruction, the confluence audit packet, the typed conversion residual frontier,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
11. *L is the local* $\mathsf{NameCert}_{\mathsf{MetacicCandidateSnHandoff}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for candidate-free closed-term SN, unconditional subject reduction, unrestricted confluence, full substitution compatibility, typed conversion decidability, or closed consistency.

Theorem 6085.2 (MetaCIC candidate SN handoff obligations)**.** *For every accepted* $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ *carrier* $K = (T, A, S, N, O, D, F, H, C, P, L)$*, the local* $\mathsf{NameCert}_{\mathsf{MetacicCandidateSnHandoff}^{\uparrow}}$ *surface consists exactly of closed typed term exposure through* T*, candidate evidence through* A*, SN projection through* S*, no-infinite-reduction projection through* N*, subject-reduction obstruction through* O*, finite-window diamond-envelope handoff through* D*, confluence and typed-conversion frontier routing through* F*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* L*.*

Proof. Project the carrier of Definition 6085.1. Each public row is displayed as one finite coordinate, and the SN row S is attached to the candidate row A rather than to the closed term row T alone.

Read the no-infinite-reduction row from the same candidate evidence. This keeps the termination consequence on the candidate-mediated route named by chapter 9383.

The remaining rows route neighboring audit consumers. The obstruction row O points to chapter 731, the diamond row D points to chapter 6063, and the frontier row F points to chapter 735 and chapter 6069. Since no row stores candidate-free SN, unconditional subject reduction, unrestricted confluence, full substitution compatibility, typed conversion decidability, or closed consistency, the NameCert surface is exhausted by the displayed packet. □

Theorem 6085.3 (MetaCIC candidate SN nonescape)**.** *Every normalization-audit read of an accepted* $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ *packet must pass through the supplied candidate row* A *before it may cite the strong-normalization row* S *or the no-infinite-reduction row* N*. The handoff may route that finite evidence to the subject-reduction obstruction, finite-window diamond envelope, confluence audit packet, and typed conversion residual frontier, but it does not promote the evidence to candidate-free strong normalization, closed subject reduction, unrestricted confluence, full substitution compatibility, typed conversion decidability, or closed consistency.*

Proof. Start with the carrier rows T, A, S, N. The closed typed term row T is visible, but the termination rows S and N are displayed as consequences of A.

Use componentwise transport H and replay C to move the same evidence through a consumer request. Transport preserves the attachment of S and N to A, so a replayed request cannot drop the candidate row.

Finally inspect the sibling routes O, D, F. They hand the finite candidate-mediated evidence to the subject-reduction, diamond-envelope, confluence, and typed-conversion boundaries as audit data only. Those siblings are boundary packets, so the handoff exports no candidate-free SN theorem, no unconditional subject-reduction theorem, no unrestricted confluence theorem, no full substitution-compatibility theorem, no typed conversion decider, and no closed-consistency endpoint. □

Theorem 6085.4 (MetaCIC normalization audit map realization)**.** *The* $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ *packet is the finite NameCert realization of the candidate-mediated branch of chapter 9383: it packages exactly the closed typed term, supplied candidate evidence, SN projection, no-infinite-reduction projection, subject-reduction obstruction handoff, finite-window diamond envelope, confluence audit boundary, typed conversion residual frontier, transport, replay, provenance, and local naming rows that a normalization-audit consumer may inspect.*

Proof. Apply Definition 6085.2. It lists every carrier row and identifies the candidate row as the required source of the SN and no-infinite-reduction projections.

Apply Definition 6085.3. It prevents a consumer from erasing the candidate evidence or replacing the boundary siblings by unconditional subject-reduction, confluence, substitution, typed-conversion, or consistency theorems.

Thus the vision label is realized by finite NameCert data: the handoff names the candidate-mediated SN route and the adjacent audit boundaries, while refusing the unconditional all-closed-terms reading. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetacicCandidateSnHandoff}^{\uparrow}}$.

Constructive story: A $\mathsf{MetacicCandidateSnHandoff}^{\uparrow}$ packet is generated from a closed typed MetaCIC term row, candidate evidence row, SN projection, no-infinite-reduction projection, subject-reduction obstruction handoff, diamond-envelope row, confluence and typed-conversion frontier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetacicCandidateSnHandoff}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6085.1, Definition 6085.2, Definition 6085.3, and Definition 6085.4 seed the finite candidate-mediated SN handoff for the normalization audit map and its subject-reduction, confluence, diamond-envelope, and typed-conversion boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetacicCandidateSnHandoff}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close candidate-free strong normalization, unconditional subject reduction, unrestricted confluence, full substitution compatibility, typed conversion decidability, closed consistency, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, candidate-mediated SN projection, no-infinite-reduction projection, sibling-boundary handoff, non-escape, provenance, and local naming for BEDC.MetaCIC.closed_typed_term_sn_from_c and BEDC.Derived.MetacicCandidateSnHandoffUp.

Chapter 6086

A Concrete Naming Certificate for $\mathsf{CauchySeparatedReflection}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy presentations, the $\mathsf{CauchySeparatedReflection}^{\uparrow}$ packet records the finite separated-reflection surface needed before regular Cauchy names are consumed as separated real data. It sits over the precompletion source of chapter 1846, the regular rational readback of chapter 301, the metric source of chapter 57, the separated-reflection boundary of chapter 4151, and the terminal real-name seal of chapter 38. The packet is a finite NameCert route from Cauchy precompletion data to separated real reads; it is not quotient representative data, countable choice, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6086.1 (Cauchy separated-reflection carrier)**.** *A* $\mathsf{CauchySeparatedReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, R, M, Z, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *source row, carrying the finite Cauchy presentation, precompletion admission, and tolerance requests visible before reflection.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the same Cauchy windows and tolerance ledger.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *source row through which the distance comparisons used by the reflection request are read.*
4. *Z is the* $\mathsf{SeparatedReflection}^{\uparrow}$ *row, recording the zero-distance classifier boundary before any separated real read is public.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after A, R, M, Z have been displayed.*
6. *H records componentwise* hsame *transport for the precompletion source, regular readback, metric source, separated-reflection boundary, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay of the route*

$$\mathsf{CauchyPrecompletion}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{SeparatedReflection}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

8. *P is the* Pkg *provenance over* $\mathsf{CauchySeparatedReflection}^{\uparrow}$, $\mathsf{CauchyPrecompletion}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{SeparatedReflection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{CauchySeparatedReflection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $A, R, M, Z, E, H, C, P, N$. It has no coordinate for quotient representatives, countable choice, host equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchySeparatedReflectionUp.CauchySeparatedReflectionTasteGate single carrier alignment`

Theorem 6086.2 (Cauchy separated-reflection classifier stability)**.** *For every accepted* $\mathsf{CauchySeparatedReflection}^{\uparrow}$ *carrier $S = (A, R, M, Z, E, H, C, P, N)$, classifier transport along the displayed separated-reflection route preserves the finite Cauchy-to-real surface. A consumer may read the separated* $\mathsf{Real}^{\uparrow}$ *boundary only after the* $\mathsf{CauchyPrecompletion}^{\uparrow}$ *source A,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback R,* $\mathsf{Met}^{\uparrow}$ *distance row M,* $\mathsf{SeparatedReflection}^{\uparrow}$ *classifier row Z, componentwise* hsame *transport H, displayed* Cont *replay C,* Pkg *provenance P, and local* NameCert *row N have all been exposed.*

Proof. Project the carrier of Definition 6086.1. The visible data are exactly $A, R, M, Z, E, H, C, P, N$, so every reflected Cauchy read starts from the displayed precompletion source and regular rational readback.

The separated boundary is then read through M and Z. The metric row supplies the distance comparisons used by the zero-distance classifier, while the separated-reflection row records the only route by which indistinguishable Cauchy data can reach a separated consumer.

Transport and replay preserve this same route. The row H compares the source, readback, metric, reflection, and real-seal rows componentwise, and C repeats the path from Cauchy precompletion data through separated reflection to the $\mathsf{Real}^{\uparrow}$ seal.

Because quotient representatives, countable choice, host equality, selected limits, and ambient real completeness have no carrier coordinate, classifier stability is exhausted by the displayed rows. No consumer can read a separated real boundary outside $A, R, M, Z, E, H, C, P, N$. □

Falsifiable prediction: Every accepted $\mathsf{CauchySeparatedReflection}^{\uparrow}$ consumer that reaches a separated $\mathsf{Real}^{\uparrow}$ read while omitting the CauchyPrecompletion source, RegSeqRat readback, MetricSpace distance row, SeparatedReflection classifier row, hsame transport, Cont replay, Pkg provenance, or local NameCert row refutes the packet.

Independence witness: cauchyprecompletion, regseqrat, metricspace, separated-reflection, real: $\mathsf{CauchySeparatedReflection}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Cauchy source, regular readback, metric source, separated-reflection boundary, real seal, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchySeparatedReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchySeparatedReflection}^{\uparrow}$ packet is generated from a CauchyPrecompletion source row, RegSeqRat readback, MetricSpace distance row, SeparatedReflection classifier boundary, Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchySeparatedReflection}^{\uparrow}}$, seed)

Scope closed: Definition 6086.1 and Definition 6086.2 seed the finite separated-reflection route from Cauchy presentation data to separated Real reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchySeparatedReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient representatives, countable choice, host equality, selected limits, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular readback exactness, metric-source exposure, separated-reflection stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchySeparatedRe

Chapter 6087

A Concrete Naming Certificate for FiniteApproximateFixedPoint$^{\uparrow}$

Following Brouwer, Schauder, and Bishop–Bridges finite approximation methods, the FiniteApproximateFixedPoint$^{\uparrow}$ packet records an approximate fixed-point witness as finite mesh, radius, and displacement data before any host fixed point is selected. It sits beside the Brouwer surface of chapter 2030, the Schauder surface of chapter 1923, the compact-metric surface of chapter 129, the finite-net carrier of chapter 1767, the metric-ball route of chapter 1450, and the real-name seal of chapter 38. The packet exposes the finite approximation data consumed by fixed-point arguments; it is not a host fixed-point selector, quotient equality on points, open-cover compactness theorem, or ambient completeness principle.

Definition 6087.1 (Finite approximate fixed-point carrier)**.** *A* FiniteApproximateFixedPoint$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (K, F, E, B, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row naming the compact metric object on which the approximation is inspected.*
2. *F is the map row, read only as a displayed continuous or uniformly controlled self-route over the same compact metric source.*
3. *E is the* FiniteEpsilonNet$^{\uparrow}$ *row listing the finite mesh centers and the mesh radius requested by the approximation.*
4. *B is the* MetricBall$^{\uparrow}$ *coverage row tying each inspected point to one displayed mesh ball.*
5. *D is the displacement ledger recording the finite comparisons between each mesh center and its image under F.*
6. *R is the* Real$^{\uparrow}$ *radius and displacement seal row used for the numerical tolerance comparisons.*
7. *S is the approximate fixed-point handoff row asserting that an accepted mesh point meets the displayed displacement tolerance.*
8. *H records componentwise* hsame *transport for the compact source, self-map, finite net, ball coverage, displacement ledger, real seal, handoff, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a fixed-point request through $K, F, E, B, D, R, S, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$, $\mathsf{BrouwerFixedPoint}^{\uparrow}$, $\mathsf{SchauderFixedPoint}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{MetricBall}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteApproximateFixedPoint}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, E, B, D, R, S, H, C, P, N$. It has no coordinate for a selected fixed point, quotient point equality, arbitrary open covers, host compactness, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.FiniteApproximateFixedPointUp.FiniteApproximateFixedPointTasteGate single carrier alignment`

Theorem 6087.2 (Finite approximate fixed-point NameCert obligations)**.** *For every accepted* $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$ *carrier*

$$A = (K, F, E, B, D, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteApproximateFixedPoint}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, displayed self-map route, finite ε-net, metric-ball coverage, displacement ledger, real-valued tolerance seals, approximate handoff, componentwise* hsame *transport, displayed* Cont *replay, sibling provenance, and local naming rows. Every approximate fixed-point consumer must expose the finite net E, the ball coverage B, and the displacement ledger D before it reads the handoff row S.*

Proof. Project the carrier of Definition 6087.1. The displayed tuple is the whole public packet, so the local certificate can inspect only $K, F, E, B, D, R, S, H, C, P, N$.

The compactness-facing part is finite. The compact metric row K supplies the source, the self-map row F supplies the only allowed map read, and the finite-net row E with the metric-ball row B supplies the bounded mesh coverage.

The fixed-point-facing part is approximate. The ledger D records finite displacement comparisons, the real row R supplies the numerical tolerance seals, and the handoff row S may be read only after those finite rows are accepted.

Finally H, C, P, N transport, replay, source, and name the same route. Since no selected fixed point, quotient point equality, arbitrary open cover, host compactness object, countable-choice schedule, or ambient completeness row exists in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 6087.3 (Finite approximate fixed-point Brouwer–Schauder route)**.** *Every* $\mathsf{BrouwerFixedPoint}^{\uparrow}$ *or* $\mathsf{SchauderFixedPoint}^{\uparrow}$ *consumer that reads an accepted* $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$ *packet receives only the finite route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{FiniteEpsilonNet}^{\uparrow},\ \mathsf{MetricBall}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The route exposes mesh centers, coverage balls, displacement tolerances, and the approximate handoff row. It exports no exact fixed-point selector, no global fixed-point theorem, no quotient point equality, no open-cover compactness object, and no ambient completeness witness.

Proof. Start with the NameCert obligations of Definition 6087.2. A fixed-point consumer must first read the compact metric source, self-map route, finite net, ball coverage, displacement ledger, and real tolerance rows.

The Brouwer and Schauder siblings may consume that data only as a finite approximation surface. The provenance row P records those siblings, but it does not add a row selecting an exact fixed point or proving a global fixed-point theorem.

The structural rows H, C, N preserve and name the same finite route. Because the carrier has no coordinate for quotient equality, arbitrary open-cover compactness, selected fixed points, or ambient completeness, the exported route is precisely the finite approximation packet displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteApproximateFixedPoint}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteApproximateFixedPoint}^{\uparrow}$ packet is generated from compact metric source rows, displayed self-map rows, finite epsilon-net rows, metric-ball coverage, displacement ledgers, RealUp tolerance seals, approximate handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteApproximateFixedPoint}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6087.1, Definition 6087.2, and Definition 6087.3 seed the finite approximate fixed-point packet over compact metric, finite-net, metric-ball, RealUp, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteApproximateFixedPoint}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close exact fixed-point selection, global Brouwer or Schauder fixed-point theorems, quotient point equality, arbitrary open-cover compactness, host compactness, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, displacement-ledger exactness, Real tolerance handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteApproximateFixedPointUp.

Chapter 6088

A Concrete Naming Certificate for $\mathsf{RealCompletenessGapRouter}^{\uparrow}$

The $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ packet realizes the boundary route of chapter 9238 as finite BEDC data separating permitted StreamName, RegSeqRat, DyadicRatCore, and RealUp reads from off-table completeness claims. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, the terminal real-name seal of chapter 38, and the exact completion boundary of chapter 1120. The packet is a gate for what may be read at the completion boundary, not an assertion of ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6088.1 (Real completeness gap-router carrier)**.** *A* $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, Q, D, R, A, O, T, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which every completion-boundary request exposes finite observations.*
2. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger governing the requested precision.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, Q, D have been displayed.*
5. *A is the allowed-read ledger listing the finite routes a completion consumer may inspect.*
6. *O is the off-table ledger naming refused reads: quotient streams, selected limits, host real equality, choice schedules, and ambient completeness.*
7. *T is the TasteGate obligation row requiring allowed reads to remain finite, routed, and separately auditable from refused completeness claims.*
8. *H records componentwise* hsame *transport for the window, readback, tolerance, Real seal, allowed-read, off-table, TasteGate, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a completion-boundary request through $S, Q, D, R, A, O, T, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{RealCompletenessGapRouter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, RealCompletionExactBoundaryUp, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletenessGapRouter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, Q, D, R, A, O, T, H, C, P, N$. It has no coordinate for quotient real equality, selected representatives, countable choice, host limits, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6088.2 (Real completeness gap-router NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ *carrier*

$$G = (S, Q, D, R, A, O, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealCompletenessGapRouter}^{\uparrow}}$ *surface consists exactly of finite window exposure through S, regular rational readback through Q, dyadic tolerance visibility through D, terminal Real sealing through R, allowed-read routing through A, refused-read separation through O, TasteGate obligations through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, sibling provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6088.1. Its public rows are exactly $S, Q, D, R, A, O, T, H, C, P, N$, so any completion-boundary read must be witnessed by those rows.

The allowed side is finite. StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, and the Real seal are displayed separately before the allowed-read ledger is consumed.

The refused side is explicit. The off-table ledger O records quotient streams, selected limits, host equality, choice schedules, and ambient completeness as absent rows, while T states the TasteGate obligation that allowed reads remain finite and routed.

Transport and replay preserve the same split through H and C, and P, N source and name it. No additional completion object exists in the carrier, so the listed obligations exhaust the gap-router surface. □

Theorem 6088.3 (Real completeness gap-router allowed-read route)**.** *Every allowed completion-boundary read consumed from an accepted* $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ *packet factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

and then through the allowed-read ledger A and TasteGate row T. The same read must keep the off-table ledger O visible, so it cannot be promoted to quotient stream equality, selected-limit construction, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6088.2. They require each accepted read to expose S, Q, D, R before the allowed-read ledger A is consumed.

The TasteGate row T then checks that the route is finite, separately auditable, and sourced by the displayed StreamName, RegSeqRat, DyadicRatCore, and RealUp rows. The row does not contain a completion theorem or a bridge credential.

Finally the off-table ledger O stays visible along the same route. Since the refused objects have no carrier coordinate, the allowed read cannot become quotient equality, selected limits, host equality, choice, or ambient completeness. □

Theorem 6088.4 (Real completeness gap-router exact-boundary separation)**.** *The* $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ *route may consume* $\mathsf{RealCompletionExactBoundaryUp}$ *only as a finite sibling boundary over StreamName, RegSeqRat, DyadicRatCore, and RealUp rows. It does not convert that sibling boundary into a completed Real object, a selected representative, a quotient stream, or a host completeness theorem.*

Proof. Read the provenance row P in Definition 6088.1. It cites RealCompletionExactBoundaryUp beside the four displayed source faces, so the exact-boundary sibling is a routed dependency rather than a new coordinate of completed real equality.

Apply the allowed-read route theorem, Definition 6088.3. Any consumer must still pass through StreamName, RegSeqRat, DyadicRatCore, RealUp, the allowed ledger, the TasteGate row, and the off-table ledger.

Thus the sibling exact-boundary read remains finite and separated from the refused completion objects. The gap router exposes the boundary and the refusal ledger together, and no host completeness theorem is available for export. □

Falsifiable prediction: Every accepted $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ completion-boundary read must display StreamName, RegSeqRat, DyadicRatCore, RealUp, allowed-read, off-table, TasteGate, componentwise hsame, displayed Cont, Pkg, and local NameCert rows before any Real-facing output is visible.

Independence witness: streamname, regseqrat, dyadicratcore, real, real-completion-exact-boundary: $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries allowed-read rows, off-table refusal rows, and TasteGate obligations in addition to the four finite completion-source faces.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletenessGapRouter}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletenessGapRouter}^{\uparrow}$ packet is generated from StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, a RealUp seal, allowed-read rows, off-table refusal rows, TasteGate obligations, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletenessGapRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6088.1, Definition 6088.2, Definition 6088.3, and Definition 6088.4 seed the finite completion-boundary router separating allowed reads from refused completeness claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletenessGapRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient real equality, selected representatives, countable choice, host limit operators, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, allowed-read routing, off-table separation, TasteGate stability, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletenessGapRouterUp.

Chapter 6089

A Concrete Naming Certificate for $\mathsf{BishopCompletionSpace}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCompletionSpace}^{\uparrow}$ packet records the constructive completion-space interface as finite BEDC NameCert data. It sits after the stream-name carrier of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy-filter completion packet of chapter 1569. The packet names the completion-space rows consumed by real and metric completion arguments; it does not assert ambient completeness, quotient equality of Cauchy objects, selected limits, countable choice, or a host completion theorem.

Definition 6089.1 (Bishop completion-space carrier)**.** *A* $\mathsf{BishopCompletionSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, Q, D, R, E, F, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-space row, recording the displayed carrier, apartness or metric-facing classifier, and finite observation route used by the completion request.*
2. *Q is the regular Cauchy-name row, read through* $\mathsf{StreamName}^{\uparrow}$ *and the lawlike finite windows visible to the packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that bounds the inspected Cauchy windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the finite Cauchy data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after Q, D, R have been displayed.*
6. *F is the* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *face recording how the same Cauchy data enters the completion-facing filter route.*
7. *H records componentwise* hsame *transport for the source, Cauchy-name, dyadic tolerance, regular readback, Real seal, filter-completion face, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$X, Q, D \longmapsto R \longmapsto E \qquad \textit{and} \qquad X, Q, D \longmapsto F.$$

9. *P is the* Pkg *provenance over* $\mathsf{BishopCompletionSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCompletionSpace}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $X, Q, D, R, E, F, H, C, P, N$. It has no coordinate for quotient equality of Cauchy names, selected limits, ambient completeness, countable choice, host real equality, or an unrestricted completion object.

Lean4 ✓ `BEDC.Derived.BishopCompletionSpaceUp.TasteGate.BishopCompletionSpaceTasteGate single carrier alignment`

Theorem 6089.2 (Bishop completion-space NameCert obligations)**.** *For every accepted* $\mathsf{BishopCompletionSpace}^{\uparrow}$ *carrier $S = (X, Q, D, R, E, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopCompletionSpace}^{\uparrow}}$ *surface consists exactly of source-space exposure through X, regular Cauchy-name exposure through Q, dyadic tolerance exposure through D, regular rational readback through R, Real seal handoff through E, Cauchy-filter completion exposure through F, componentwise transport through H, displayed continuation replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The source row X, Cauchy-name row Q, dyadic tolerance ledger D, regular readback R, Real seal E, and filter-completion face F are the only completion-facing data admitted by the packet.

The analytic obligations are therefore finite. A completion consumer must first display the source row and inspected Cauchy windows, then the dyadic tolerance ledger, then the regular rational readback and Real seal. The filter-completion face is read from the same displayed data and does not add a second completion object.

The structural obligations are componentwise transport, replay, provenance, and local naming. Since the carrier has no coordinate for quotient Cauchy equality, selected limits, ambient completeness, countable choice, host real equality, or an unrestricted completion object, the listed rows exhaust the NameCert surface. □

Theorem 6089.3 (Bishop completion-space Real route)**.** *Every real-facing or completion-facing consumer of an accepted* $\mathsf{BishopCompletionSpace}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyFilterCompletion}^{\uparrow}$$

by reading $X, Q, D, R, E, F, H, C, P, N$. The route exports finite Cauchy, regular-rational, Real-seal, and filter-completion rows; it exports no quotient equality, selected limit, ambient complete space, countable-choice principle, or hidden host completion.

Proof. Begin with the stream-name and Cauchy-window rows. The row Q is read only as displayed finite windows over the source row X, and the tolerance ledger D bounds those windows before any real-facing readback is public.

Next read the regular rational and Real rows. The row R exposes the regular rational name, and E seals it as a $\mathsf{Real}^{\uparrow}$ row only after X, Q, D have been displayed.

Finally read the completion face. The row F records the Cauchy-filter completion handoff over the same finite data, while H, C, P, N preserve transport, replay, provenance, and local naming. Thus every consumer receives the finite Bishop completion-space route and none of the excluded host objects. □

Falsifiable prediction: Every $\mathsf{BishopCompletionSpace}^{\uparrow}$ read must expose source-space, StreamName, dyadic tolerance, RegSeqRat readback, Real seal, CauchyFilterCompletion, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before a completion-facing conclusion is consumed.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchyfiltercompletion: $\mathsf{BishopCompletionSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source-space exposure, regular Cauchy windows, dyadic tolerance, regular rational readback, Real seal, Cauchy-filter completion face, transport, replay, provenance, and local naming in one completion-space packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCompletionSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCompletionSpace}^{\uparrow}$ packet is generated from a source-space row, regular Cauchy-name row, dyadic tolerance ledger, RegSeqRat readback, Real seal, CauchyFilterCompletion face, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCompletionSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6089.1, Definition 6089.2, and Definition 6089.3 seed the finite constructive completion-space interface over stream-name, regular rational, dyadic tolerance, Real-seal, and Cauchy-filter completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCompletionSpace}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.BishopCompletionSpaceUp.TasteGate.BishopCompletionSpaceTasteGate\_single\_carrier\_alig`

Bridge status: none

Not claimed: This chapter does not claim ambient completeness, quotient equality of Cauchy names, selected limits, countable choice, host real equality, bridge checking, or an unrestricted completion theorem.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exposure, dyadic tolerance exactness, regular rational readback, Real-seal nonescape, Cauchy-filter completion compatibility, transport, replay, provenance, and local naming for BEDC.Derived.BishopCompletionSpaceUp.

Chapter 6090

A Concrete Naming Certificate for HyperbolicLinearizedPhaseDistance$^{\uparrow}$

The HyperbolicLinearizedPhaseDistance$^{\uparrow}$ packet realizes the linearized phase-distance reading of chapter 9463 as finite BEDC NameCert data. It sits after the boundary-shadow packet of chapter 3813, the visual-metric packet of chapter 3866, the observer-expansion packet of chapter 4167, the phase-distance observer-expansion packet of chapter 6017, the radius-ladder packet of chapter 6021, and the complex phase-surface fiber of chapter 6044. The packet names a finite boundary-shadow, visual-metric, Real-seal, transport, replay, provenance, and local naming surface; it does not assert a host hyperbolic metric space, a global visual boundary, differentiable phase flow, quotient geodesic equality, or an observer object outside the displayed rows.

Definition 6090.1 (Hyperbolic linearized phase-distance carrier)**.** *A* HyperbolicLinearizedPhaseDistance$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, V, O, \Phi, R, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* HyperbolicBoundaryShadow$^{\uparrow}$ *row naming the finite boundary-shadow read.*
2. *V is the* HyperbolicVisualMetric$^{\uparrow}$ *row naming the visual-metric comparison consumed by the phase distance.*
3. *O is the* HyperbolicObserverExpansion$^{\uparrow}$ *row anchoring the observer-side expansion route.*
4. *Φ is the phase-distance row, read through* PhaseDistanceObserverExpansion$^{\uparrow}$ *and the complex phase-surface fiber.*
5. *R is the regular real-readback row, displaying the* RegSeqRat$^{\uparrow}$ *and* DyadicRatCore$^{\uparrow}$ *tolerance windows used by the linearized comparison.*
6. *E is the* Real$^{\uparrow}$ *seal row for the displayed distance value.*
7. *H records componentwise* hsame *transport for boundary shadow, visual metric, observer expansion, phase distance, real readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from boundary-shadow admission through visual-metric comparison and phase-distance readback to the Real seal.*

9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpansion}^{\uparrow}$, $\mathsf{ComplexPhaseSurface}$ $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $B, V, O, \Phi, R, E, H, C, P, N$. It has no coordinate for a host hyperbolic metric space, global visual boundary, differentiable phase flow, quotient geodesic equality, selected observer point, or hidden real-distance oracle.

Theorem 6090.2 (Hyperbolic linearized phase-distance NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$ *carrier $L = (B, V, O, \Phi, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicLinea}}$ *surface consists exactly of boundary-shadow exposure through B, visual-metric exposure through V, observer-expansion anchoring through O, phase-distance readback through Φ, regular real readback through R, Real-seal handoff through E, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6090.1. The geometric rows are exactly the boundary-shadow, visual-metric, observer-expansion, phase-distance, real-readback, and Real-seal rows.

The distance read is linearized only as a finite route. Row B supplies the boundary shadow, V supplies the visual comparison, O anchors the observer-expansion route, Φ records the phase-distance readback, and R, E expose the regular rational and Real-seal interface.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Since the carrier has no coordinate for a host hyperbolic metric space, global visual boundary, differentiable phase flow, quotient geodesic equality, selected observer point, or hidden real-distance oracle, the listed rows exhaust the NameCert surface. □

Theorem 6090.3 (Hyperbolic linearized phase-distance observer route)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$ *packet reads the distance route through* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{PhaseDistanceObserverExpans}$ $\mathsf{ComplexPhaseSurfaceFiber}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *rows before it receives a public comparison value. The handoff exports no host hyperbolic space, global visual boundary, quotient geodesic equality, selected observer point, or differentiable phase-flow theorem.*

Proof. Start with the boundary and visual rows. The rows B and V are the only boundary-shadow and visual-metric data admitted by the carrier, so an observer-expansion consumer cannot read a distance value before those rows are displayed.

Next read the observer and phase rows. The row O anchors the focused expansion route, while Φ records the phase-distance readback against the complex phase-surface fiber rather than against a hidden geometric object.

Finally pass through R, E, H, C, P, N. The regular readback and Real seal expose the public comparison value, and the structural rows preserve the route. Thus the observer route remains finite NameCert data and not any excluded host-level geometry. □

Falsifiable prediction: A $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$ packet is rejected whenever it cannot display boundary-shadow, visual-metric, observer-expansion, phase-distance, regular readback, Real-seal, hsame, Cont, Pkg, and local NameCert rows before exporting a comparison value.

Independence witness: hyperbolic-boundary-shadow, hyperbolic-visual-metric, hyperbolic-observer-expansion, phase-distance-observer-expansion, complex-phase-surface-fiber: no listed sibling is bijective to $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$ because this packet jointly carries boundary shadow,

visual metric, observer expansion, phase-distance readback, complex phase-fiber anchoring, Real seal, transport, replay, provenance, and local naming rows.

The Lean TasteGate route for this packet is planned to expose Nontrivial and FieldFaithful obligations for `BEDC.Derived.HyperbolicLinearizedPhaseDistanceUp`; those obligations must preserve the displayed boundary-shadow, visual-metric, Real-seal, Cont, Pkg, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}$ packet is generated from a boundary-shadow row, visual-metric row, observer-expansion row, phase-distance readback, regular real-readback row, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6090.1, Definition 6090.2, and Definition 6090.3 seed the finite linearized phase-distance packet over boundary-shadow, visual-metric, observer-expansion, phase-distance, Real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicLinearizedPhaseDistance}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host hyperbolic metric spaces, global visual boundaries, differentiable phase flows, quotient geodesic equality, selected observer points, hidden real-distance oracles, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked boundary-shadow exposure, visual-metric exposure, observer-expansion anchoring, phase-distance readback, Real-seal nonescape, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.HyperbolicLinearizedPhaseDistanceUp.

Chapter 6091

A Concrete Naming Certificate for $\mathsf{NestedIntervalCompleteness}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ packet records the finite L10 route by which a bounded decreasing nested interval chain is read as a completion-facing Real seal. It sits over the closed bounded interval packet of chapter 1591, the nested interval source of chapter 320, the Cantor-intersection consumer of chapter 1421, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names completeness as displayed finite rows; it does not export a host-selected point, quotient stream equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6091.1 (Nested interval completeness carrier)**.** *A* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, N, C, W, R, E, H, T, P, Q)$$

with the following displayed rows:

1. *B is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *endpoint row, carrying the located closed interval bounds used at the root of the chain.*
2. *N is the* $\mathsf{NestedInterval}^{\uparrow}$ *row, carrying the decreasing finite-window chain and the endpoint containment ledger.*
3. *C is the* $\mathsf{CantorIntersection}^{\uparrow}$ *handoff row, available only after the nested source and closed bounded interval rows have been displayed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation window read from the displayed nested interval chain.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row obtained from the same finite windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed after B, N, C, W, R.*
7. *H records componentwise* hsame *transport for the interval, nested source, Cantor handoff, stream window, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *T is the displayed* Cont *replay from a requested interval precision through B, N, C, W, R, E.*
9. *P is the* Pkg *provenance over* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$, $\mathsf{ClosedBoundedInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{CantorIntersection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *Q is the local* $\mathsf{NameCert}_{\mathsf{NestedIntervalCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these ten rows. It has no coordinate for a selected host real, quotient stream equality, countable choice, trichotomy, host equality, or detached completeness theorem.

Theorem 6091.2 (Nested interval completeness NameCert obligations)**.** *For every accepted* $\mathsf{NestedIntervalComplete}$ *carrier*

$$L = (B, N, C, W, R, E, H, T, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{NestedIntervalCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *closed bounded interval admission through* B*;*
2. *nested interval admission through* N*, requiring monotone containment and displayed finite width windows;*
3. *Cantor-intersection handoff through* C*, requiring any completion-facing read to pass through the existing nested-intersection packet;*
4. *finite stream and regular rational readback through* W *and* R*;*
5. *terminal Real-seal admission through* E*;*
6. *componentwise transport, replay, provenance, and local naming through* H, T, P, Q*.*

No nested interval completeness obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6091.1. The rows B, N, C, W, R, E are the mathematical route, while H, T, P, Q preserve and name it.

The closed bounded interval row fixes the endpoint surface. The nested interval row supplies the decreasing finite-window chain, and the Cantor-intersection row consumes that chain before any completion-facing Real seal is visible.

The stream and regular rational rows expose only finite observations and their readback. Since the carrier has no host point, quotient equality, countable choice, trichotomy, host equality, or detached completeness coordinate, the displayed rows exhaust the local NameCert obligations. □

Theorem 6091.3 (Nested interval completeness Real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ClosedBoundedInterval}^{\uparrow} \longmapsto \mathsf{NestedInterval}^{\uparrow} \longmapsto \mathsf{CantorIntersection}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the completeness read is a displayed Real-name seal generated from interval endpoints, nested containment, Cantor-intersection handoff, finite stream windows, and regular rational readback. It is not a host completeness theorem, selected point, quotient stream equality, countable choice, trichotomy, or host real equality claim.

Lean4 ✓ `BEDC.Derived.NestedIntervalCompletenessUp.NestedIntervalCompleteness real seal`

Proof. Begin with the endpoint and nested-source rows B and N. They determine the interval family visible to the packet, so a Real-facing consumer cannot start from an ambient complete line.

Read the Cantor-intersection handoff row C. This row is downstream of the displayed interval family and upstream of the finite windows, so it is the only completion-facing bridge available inside the carrier.

Finally read the stream window W, regular rational readback R, and Real seal E. The transport, replay, provenance, and naming rows preserve this same route. Because no other coordinate exists, every Real-facing consumer factors through the displayed finite path. □

Theorem 6091.4 (Nested interval completeness finite-window extraction)**.** *For every accepted* $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ *carrier*

$$L = (B, N, C, W, R, E, H, T, P, Q),$$

any completion-facing $\mathsf{Real}^{\uparrow}$ *read is exposed only after the finite* $\mathsf{StreamName}^{\uparrow}$ *window row* W *and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *have been displayed. Classifier transport preserves those two rows componentwise before the Real seal* E *is consumed. The extraction does not select a host real, identify quotient streams, use countable choice, invoke trichotomy, or import ambient Real completeness.*

Proof. Project the carrier of Definition 6091.1. The rows W and R are explicit coordinates of the accepted packet, while the seal E is listed only downstream of the closed interval, nested source, Cantor handoff, finite window, and regular readback rows.

Apply the NameCert obligations of Definition 6091.2. They force every completion-facing read to pass through B, N, C, W, R before E, and they require componentwise transport through H for the same displayed rows.

Finally use the Real seal route of Definition 6091.3. The route places $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$ immediately before $\mathsf{Real}^{\uparrow}$, so a consumer cannot bypass W or R when consuming the seal. Since the carrier has no selected host real, quotient stream equality, countable-choice schedule, trichotomy coordinate, or ambient completeness row, the finite-window extraction is exactly the displayed witness route. □

Closure status for $\mathsf{NameCert}_{\mathsf{NestedIntervalCompleteness}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NestedIntervalCompleteness}^{\uparrow}}$, seed)

Scope closed: Definition 6091.1, Definition 6091.2, and Definition 6091.3 seed the finite nested-interval completeness route from closed bounded interval and nested interval data through CantorIntersection, StreamName, RegSeqRat, and the RealUp seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NestedIntervalCompleteness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host completeness, selected real points, quotient stream equality, countable choice, trichotomy, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked closed-interval admission, nested-source admission, Cantor-intersection handoff, finite-window readback, Real-seal admission, transport, replay, provenance, and local naming for BEDC.Derived.NestedIntervalCompletenessUp.

Constructive story: A $\mathsf{NestedIntervalCompleteness}^{\uparrow}$ packet is generated from a ClosedBoundedIntervalUp endpoint row, a NestedIntervalUp source row, a CantorIntersectionUp handoff, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6092

A Concrete Naming Certificate for $\mathsf{GaugeLebesgueNumber}^{\uparrow}$

Following Cousin and Bishop gauge-cover arguments, the $\mathsf{GaugeLebesgueNumber}^{\uparrow}$ packet records the finite handoff from a displayed gauge cover to a positive Lebesgue-number ledger. It sits over the reusable cover row of chapter 3426, the tagged-subdivision theorem row of chapter 1861, the finite radius ledger of chapter 452, the compact metric source of chapter 129, and the rational comparison surface of chapter 37. The packet is the gauge-facing radius object consumed by compact-uniform and gauge-integration routes; it does not choose an open subcover, select a host minimum, import open-cover compactness, or assert ambient completeness.

Definition 6092.1 (Gauge Lebesgue number carrier)**.** *A* $\mathsf{GaugeLebesgueNumber}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, G, U, F, Q, H, T, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite-net and located coverage data bound the gauge request.*
2. *G is the* $\mathsf{CousinCover}^{\uparrow}$ *gauge-cover row, carrying tagged cover cells and their local gauge admissions.*
3. *U is the* $\mathsf{CousinLemma}^{\uparrow}$ *subdivision row, tying the same gauge cover to a finite tagged partition.*
4. *F is the* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *positive-radius ledger for the displayed gauge cover.*
5. *Q is the* $\mathsf{Rat}^{\uparrow}$ *comparison row certifying that the chosen radius is positive and below the relevant gauge radii.*
6. *H records componentwise* hsame *transport for the compact source, gauge cover, Cousin subdivision, finite radius ledger, rational comparison, replay, provenance, and local naming rows.*
7. *T is the displayed* Cont *replay from a compact-cover request through K, G, U, F, Q.*
8. *P is the* Pkg *provenance over* $\mathsf{GaugeLebesgueNumber}^{\uparrow}$, $\mathsf{CousinCover}^{\uparrow}$, $\mathsf{CousinLemma}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{GaugeLebesgueNumber}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, U, F, Q, H, T, P, N$. It has no coordinate for arbitrary open covers, selected subcovers, host minima, quotient metric equality, countable choice, or detached compactness.

Lean4 ✓ `BEDC.Derived.GaugeLebesgueNumberUp.GaugeLebesgueNumberTasteGate single carrier alignment`

Theorem 6092.2 (Gauge Lebesgue number NameCert obligations)**.** *For every accepted* $\mathsf{GaugeLebesgueNumber}^{\uparrow}$ *carrier*

$$L = (K, G, U, F, Q, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{GaugeLebesgueNumber}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact metric admission through* K*;*
2. *Cousin gauge-cover admission through* G*;*
3. *Cousin-lemma subdivision compatibility through* U*;*
4. *finite Lebesgue-number radius admission through* F*;*
5. *rational positivity and lower-bound comparison through* Q*;*
6. *componentwise transport, replay, provenance, and local naming through* H, T, P, N*.*

No gauge Lebesgue-number obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6092.1. The rows K, G, U, F, Q are the mathematical handoff, while H, T, P, N preserve and name the same displayed route.

The compact metric row supplies the finite coverage source. The Cousin cover and Cousin lemma rows keep the gauge data tied to a finite tagged subdivision rather than to an arbitrary open-cover argument.

The finite Lebesgue-number row and rational comparison row provide the positive lower radius. Since the carrier contains no selected subcover, host minimum, quotient equality, choice, or detached compactness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6092.3 (Gauge Lebesgue number cover handoff)**.** *Every compact-uniform or gauge-integration consumer of an accepted* $\mathsf{GaugeLebesgueNumber}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{CousinCover}^{\uparrow} \longmapsto \mathsf{CousinLemma}^{\uparrow} \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto \mathsf{Rat}^{\uparrow}.$$

Thus the exported radius is read as a finite gauge-cover lower-bound ledger over BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows. It is not a selected host Lebesgue number, not a general open-cover compactness proof, and not an external bridge request.*

Proof. Start with the compact metric row K and the gauge-cover row G. Together they bind the cover request to displayed finite metric and gauge data.

Read the Cousin-lemma row U. It supplies the finite tagged-subdivision compatibility needed by gauge-integration consumers and keeps the cover handoff inside the displayed gauge route.

Finally read the finite radius ledger F and rational comparison row Q. These are the only rows that export a positive radius, and they are already tied to K, G, U by the carrier. The structural rows H, T, P, N transport and name that path, so every consumer receives the same finite cover handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaugeLebesgueNumber}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaugeLebesgueNumber}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6092.1, Definition 6092.2, and Definition 6092.3 seed the finite gauge-cover Lebesgue-number route over CompactMetric, CousinCover, CousinLemma, FiniteLebesgueNumber, Rat, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaugeLebesgueNumber}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, selected subcovers, host Lebesgue numbers, quotient metric equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked compact metric admission, Cousin cover admission, Cousin lemma compatibility, finite Lebesgue-number radius admission, rational comparison, transport, replay, provenance, and local naming for BEDC.Derived.GaugeLebesgueNumberUp.

Constructive story: A GaugeLebesgueNumber$^{\uparrow}$ packet is generated from a CompactMetricUp source row, a CousinCoverUp gauge-cover row, a CousinLemmaUp subdivision row, a FiniteLebesgueNumberUp positive-radius ledger, a RatUp lower-bound comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6093

A Concrete Naming Certificate for $\mathsf{EilenbergSteenrodAxioms}^{\uparrow}$

Following Eilenberg and Steenrod, the $\mathsf{EilenbergSteenrodAxioms}^{\uparrow}$ packet records the finite NameCert surface by which a homology-style theory exposes dimension, exactness, excision, homotopy invariance, and naturality obligations without importing an ambient theorem oracle. It sits over the homology carrier of chapter 98, the cohomology comparison surface of chapter 99, the finite simplicial source of chapter 238, and the homotopy route of chapter 156. The packet supplies a classical axiom ledger that downstream homology and cohomology chapters can consume as displayed rows rather than as host mathematical background.

Definition 6093.1 (Eilenberg–Steenrod axioms carrier)**.** *An* $\mathsf{EilenbergSteenrodAxioms}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (H, C, S, T, D, X, E, I, Nt, R, P, L)$$

with the following displayed rows:

1. *H is the* $\mathsf{Homology}^{\uparrow}$ *target row whose groups, boundaries, cycles, and quotient-facing claims are read only through the displayed packet;*
2. *C is the* $\mathsf{Cohomology}^{\uparrow}$ *comparison row, recording the contravariant consumer surface and coefficient-facing requests;*
3. *S is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *finite source row used when the axiom ledger is instantiated on finite complexes;*
4. *T is the* $\mathsf{Homotopy}^{\uparrow}$ *route row for maps, homotopies, and homotopy-class reads;*
5. *D is the dimension row, naming the point object and coefficient landing used by the theory;*
6. *X is the exactness row for the boundary, kernel, image, and connecting-map obligations;*
7. *E is the excision row, naming the finite subobject and removal route on which the comparison is valid;*
8. *I is the homotopy-invariance row, recording the displayed route through T rather than a host equality of maps;*
9. *Nt is the naturality row for maps of pairs and induced homology or cohomology reads;*
10. *R is the refusal ledger for host exactness, quotient-object equality, selected representatives, unbounded chain complexes, and ambient abelian-category or topological-space oracles;*

11. *P records* Pkg *provenance over* EilenbergSteenrodAxioms$^{\uparrow}$, Homology$^{\uparrow}$, Cohomology$^{\uparrow}$, SimplicialComplex$^{\uparrow}$, Homotopy$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows;*
12. *L is the local* NameCert$_{\text{EilenbergSteenrodAxioms}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an external homology theory, global exactness theorem, quotient equality of chain representatives, countable choice of cycles, or ambient topological completeness principle.

Lean4 ✓ `BEDC.Derived.EilenbergSteenrodAxiomsUp.EilenbergSteenrodAxiomsTasteGate single carrier alignment`

Theorem 6093.2 (Eilenberg–Steenrod exactness obligation)**.** *For an accepted* EilenbergSteenrodAxioms$^{\uparrow}$ *carrier*

$$A = (H, C, S, T, D, X, E, I, Nt, R, P, L),$$

the exactness obligation visible to a downstream Homology$^{\uparrow}$ *or* Cohomology$^{\uparrow}$ *consumer is precisely the displayed row X together with the source rows H, C, S, the homotopy route T, the refusal ledger R, provenance P, and local name L. A consumer may request boundary, kernel, image, and connecting-map data only through this finite* BHist *route, and may not replace it by host exactness, quotient-object equality, selected chain representatives, countable choice, or an ambient abelian-category oracle.*

Proof. Project the carrier of Definition 6093.1. The only exactness coordinate is X, and it is displayed alongside the homology, cohomology, simplicial-source, and homotopy-route rows.

The downstream consumer therefore reads exactness after the source rows have been exposed. The row H supplies the homology-facing target, C supplies the cohomology-facing comparison, S supplies the finite simplicial carrier, and T supplies the homotopy route on which induced maps are compared.

The refusal ledger R closes the non-displayed alternatives. Since R excludes host exactness, quotient-object equality, selected representatives, countable choice, and ambient categorical oracles, the exactness obligation remains the finite row X named by the local NameCert row L. □

Theorem 6093.3 (Eilenberg–Steenrod homotopy invariance ledger)**.** *For an accepted* EilenbergSteenrodAxioms$^{\uparrow}$ *carrier*

$$A = (H, C, S, T, D, X, E, I, Nt, R, P, L),$$

the homotopy-invariance claim is the ledger row I read after the Homotopy$^{\uparrow}$ *route T and before the naturality row Nt. It records equality of induced homology or cohomology reads only as displayed NameCert data over $H, C, S, T, I, Nt, R, P, L$; it does not export host equality of maps, quotient equality of chain representatives, selected homotopy representatives, global topological equivalence, or ambient completeness.*

Proof. Start from the homotopy route row T. It is the only row in the carrier that names maps and homotopies, so a homotopy-invariance read cannot begin from a host equality of maps.

Next read the ledger row I. The row records the invariance obligation over the already displayed homology, cohomology, simplicial-source, and homotopy-route data, while Nt keeps the induced-map comparison on the same finite route.

Finally inspect the refusal and naming rows. The ledger R rejects quotient representatives, selected homotopies, global equivalence, and ambient completeness, and P, L only source and name the displayed packet. Hence the homotopy-invariance claim is a finite NameCert ledger entry, not an external theorem import. □

Falsifiable prediction: A homology or cohomology consumer that proves exactness or homotopy invariance without reading the displayed X, T, I, Nt, R, P, L rows is not consuming an accepted EilenbergSteenrodAxiomsUp packet.

Independence witness: homology, cohomology, simplicialcomplex, and homotopy are non-bijective siblings: none alone carries the complete finite ledger of dimension, exactness, excision, homotopy invariance, naturality, refusal, provenance, and local NameCert data exposed by $\mathsf{EilenbergSteenrodAxioms}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{EilenbergSteenrodAxioms}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EilenbergSteenrodAxioms}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6093.1, Definition 6093.2, Definition 6093.3, the falsifiable prediction, and the independence witness seed the finite Eilenberg–Steenrod axiom ledger for homology and cohomology consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EilenbergSteenrodAxioms}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, full classical homology theory, quotient chain equality, selected representatives, arbitrary topological spaces, countable choice, or ambient exactness or completeness.

Upgrade path: Obligation closure requires checked carrier admission, dimension, exactness, excision, homotopy-invariance, naturality, refusal-ledger, provenance, and local NameCert rows for BEDC.Derived.EilenbergSteenrodAxiomsUp.

Constructive story: An $\mathsf{EilenbergSteenrodAxioms}^{\uparrow}$ packet is generated from homology and cohomology rows, a finite simplicial source, a homotopy route, dimension, exactness, excision, homotopy-invariance, naturality, refusal, provenance, and local naming rows.

Chapter 6094

A Concrete Naming Certificate for $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$

Following Bishop–Bridges constructive analysis and the finite-net proof of Arzela–Ascoli, the $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ packet records the finite selector route that turns displayed equicontinuous family rows into a reusable modulus packet. It reads compact finite-net data from chapter 129, family-radius rows from chapter 4076, reusable modulus ledgers from chapter 369, rational lower-bound folds from chapter 37, metric source comparisons from chapter 4230, and terminal real-name seals from chapter 38. The packet names a finite selection route only; it is not an open-cover compactness proof, a selected subcover, a countable-choice principle, a host function-equality theorem, or an ambient completeness result.

Definition 6094.1 (Equicontinuity modulus selection carrier)**.** *An* $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, E, Q, M, U, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, carrying named centers, located coverage radii, and the finite observation order.*
2. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family row, displaying the shared local radius data for the indexed function family over the same source metric.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row that records the positive radius ledger selected by finite recursion over K and E.*
4. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row used to transport source-distance estimates from net centers to arbitrary displayed points.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *packet exported only after K, E, Q, M have been displayed.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row for Real-facing consumers of the selected modulus packet.*
7. *H records componentwise* hsame *transport for the compact-net, equicontinuity, rational-fold, metric, uniform-modulus, Real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through $K, E, Q, M, U, R, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityModulusSelection}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected subcovers, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityModulusSelectionUp.TasteGate.EquicontinuityModulusSelectionTasteGate`
`single carrier alignment`

Theorem 6094.2 (Equicontinuity modulus selection NameCert obligations)**.** *For every accepted* $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ *carrier*

$$S = (K, E, Q, M, U, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuityModulusSelection}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite-net, equicontinuity, rational-fold, metric, uniform-modulus, Real-seal, transport, replay, provenance, and local naming rows;*
2. *compact-net exposure, requiring every selected modulus read to display K before the family-radius fold is inspected;*
3. *family-radius exposure, requiring E to supply the shared equicontinuity radius rows consumed by the finite fold;*
4. *rational-fold exactness, requiring Q to record the positive lower bound chosen by finite recursion over the displayed net and family rows;*
5. *uniform-modulus handoff, requiring U and then R to be downstream of K, E, Q, M;*
6. *structural stability, requiring H, C, P, N to preserve, replay, source, and name the same finite selection packet.*

No selection obligation is stored outside these rows.

Proof. Project the carrier of Definition 6094.1. The visible analytic rows are exactly the compact finite net, equicontinuity family radii, rational lower-bound fold, metric comparisons, uniform-modulus packet, and Real seal.

The compact-net and family-radius obligations fix the finite domain of selection. A selected radius is read as a coordinate of the displayed net and family rows, not as a host chooser.

The rational-fold row then records the finite minimum route. The metric row transports the center estimate to the displayed source comparison, and the uniform-modulus and Real rows receive only that folded ledger.

The structural rows add no selector data. They transport, replay, source, and name the same packet, so the six obligation rows exhaust the local NameCert surface. □

Theorem 6094.3 (Equicontinuity modulus selection finite-net handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *or Arzela–Ascoli-facing consumer of an accepted* $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ *carrier receives its modulus only through the finite route*

$$K, E, Q, M, U, R, H, C, P, N.$$

Here K supplies compact finite-net centers, E supplies the displayed equicontinuity family radii, Q supplies the positive rational lower-bound fold, M supplies the source-metric comparison, U supplies

the reusable uniform-modulus packet, and R supplies the terminal Real-facing seal. The handoff exports a reusable modulus packet without open-cover compactness, selected subcovers, countable choice, host function equality, or ambient Real completeness.

Proof. Start with the NameCert obligations of Definition 6094.2. They require the compact finite-net row and the equicontinuity family row before any rational fold or uniform-modulus handoff is available.

Read the finite selector route through Q. Since the fold is over the displayed net and family rows, the selected radius is a finite packet coordinate.

The metric row M and uniform-modulus row U then consume the same radius ledger, and the Real seal R is downstream of that handoff. The structural rows H, C, P, N preserve the route, so no consumer can replace it by a host selector or ambient compactness principle. □

Falsifiable prediction: Every accepted $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ read must display CompactMetric finite-net data, Equicontinuity family-radius rows, RatUp lower-bound folds, MetricSpace comparisons, UniformModulus handoff, Real seal, transport, replay, provenance, and local naming.

Independence witness: compactmetric, equicontinuity, rat, metricspace, uniformmodulus, real: $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ is not reducible to any listed sibling because it binds compact finite nets and family-radius rows to one reusable modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuityModulusSelection}^{\uparrow}}$ **·**

Constructive story: An $\mathsf{EquicontinuityModulusSelection}^{\uparrow}$ packet is generated from CompactMetric finite-net data, Equicontinuity family-radius rows, RatUp lower-bound folds, MetricSpace comparisons, a UniformModulus handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuityModulusSelection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6094.1, Definition 6094.2, and Definition 6094.3 seed the finite selector route from equicontinuous family rows and compact finite nets to a reusable modulus packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuityModulusSelection}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, host function equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-net exposure, family-radius exposure, rational-fold exactness, uniform-modulus handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EquicontinuityModulusSelectionUp.

Chapter 6095

A Concrete Naming Certificate for $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$

Following Bishop–Bridges constructive analysis of equicontinuous families on compact metric sources, the $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ packet records the finite family-radius choice used before a uniform-modulus consumer reads the compact route. It sits on the compact finite-net surface of chapter 129, the displayed family-radius surface of chapter 4076, the graph and pointwise-control rows of chapter 130, the reusable ledger of chapter 369, the positive-radius fold of chapter 37, the source-metric comparison row of chapter 4230, and the terminal real-name boundary of chapter 38. The packet chooses only among displayed finite family-radius rows; it does not supply an open-cover subcover, a countable selector, host function equality, or ambient completeness.

Definition 6095.1 (Equicontinuity radius choice carrier)**.** *An* $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (K, F, E, Q, M, U, A, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net row, including named centers, located coverage radii, and the finite order in which centers are inspected.*
2. *F is the finite family-index row, listing exactly the functions whose equicontinuity radii are available to the packet.*
3. *E is the* $\mathsf{Equicontinuity}^{\uparrow}$ *family-radius row over F, displaying the common source-radius choices at the compact-net centers.*
4. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive lower-bound fold row over the displayed family and compact-net radius ledger.*
5. *M is the* $\mathsf{Met}^{\uparrow}$ *source comparison row used to move from center estimates to arbitrary displayed source points.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row reached only after K, F, E, Q, M have been displayed.*
7. *A is the downstream* $\mathsf{Real}^{\uparrow}$ *seal for Real-facing reads of the chosen family-radius packet.*
8. *H records componentwise* hsame *transport for the compact-net, finite-family, equicontinuity, rational-fold, metric, uniform-modulus, Real-seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a family-radius request through $K, F, E, Q, M, U, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Equicontinuity}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{EquicontinuityRadiusChoice}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a hidden family selector, selected subcover, countable choice, host function equality, quotient Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EquicontinuityRadiusChoiceUp.EquicontinuityRadiusChoiceTasteGate single carrier alignment`

Theorem 6095.2 (Equicontinuity radius choice NameCert obligations)**.** *For every accepted* EquicontinuityRadiusCh *carrier*

$$R = (K, F, E, Q, M, U, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{EquicontinuityRadiusChoice}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact finite-net, finite-family, equicontinuity, rational-fold, metric, uniform-modulus, Real-seal, transport, replay, provenance, and local naming rows;*
2. *finite-family exposure, requiring every radius choice to be indexed by the displayed row F;*
3. *compact-net exposure, requiring the same compact finite-net row K to supply the centers and coverage radii consumed by the family-radius fold;*
4. *equicontinuity-radius exactness, requiring E to display the family radii before Q records a positive lower bound;*
5. *uniform-modulus handoff, requiring U and then A to be downstream of K, F, E, Q, M;*
6. *structural stability, requiring H, C, P, N to preserve, replay, source, and name the same finite family-radius packet.*

No family-radius choice obligation is stored outside these rows.

Proof. Project the carrier of Definition 6095.1. Its visible rows are exactly the compact finite net, finite family index, equicontinuity radius row, rational lower-bound fold, metric comparison, uniform-modulus handoff, Real seal, and structural rows.

The finite-family and compact-net exposure clauses fix the two finite domains of the choice. A chosen radius is therefore a displayed coordinate of F and K, not an ambient selector.

The equicontinuity row E supplies the available radii, and Q records the positive lower-bound fold over those displayed rows. The metric row M is the only source-comparison row available to transport the estimate.

Finally U, A, H, C, P, N receive, seal, transport, replay, source, and name the same packet. Since none of those rows adds choice data, the listed obligations exhaust the local NameCert surface. □

Theorem 6095.3 (Equicontinuity radius choice finite-family handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ *packet receives its family radius only through the finite route*

$$K, F, E, Q, M, U, A, H, C, P, N.$$

The route first displays the compact finite net K and finite family row F, then reads the equicontinuity radius row E, folds the positive rational bound through Q, transports the source estimate

through M, and hands the result to U and A. It is a finite family-radius handoff, not a hidden selector, selected subcover, countable-choice family, host function equality, quotient Real equality, or ambient completeness principle.

Proof. Begin with the NameCert obligations of Definition 6095.2. They require K and F before any radius choice can be read by a uniform-modulus consumer.

The family-radius row E is then consumed over those displayed finite domains. The rational row Q records the lower-bound fold, and the metric row M supplies the source comparison needed by the handoff.

The consumer row U and Real seal A are downstream of the same fold. Transport, replay, provenance, and local naming preserve the route, so the packet exports exactly the finite family-radius choice displayed above. □

Falsifiable prediction: Every accepted $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ read must display CompactMetric finite-net data, a finite family row, Equicontinuity family radii, RatUp lower-bound folds, MetricSpace comparisons, UniformModulus handoff, Real seal, transport, replay, provenance, and local naming.

Independence witness: compactmetric, equicontinuity, continuousmap, rat, metricspace, uniformmodulus, real: $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ is not reducible to any listed sibling because it binds a finite family-radius row to a compact finite-net lower-bound fold and a uniform-modulus handoff.

Closure status for $\mathsf{NameCert}_{\mathsf{EquicontinuityRadiusChoice}^{\uparrow}}$ **.**

Constructive story: An $\mathsf{EquicontinuityRadiusChoice}^{\uparrow}$ packet is generated from CompactMetric finite-net data, a finite family row, Equicontinuity radius rows, RatUp lower-bound folds, MetricSpace comparisons, a UniformModulus handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{EquicontinuityRadiusChoice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6095.1, Definition 6095.2, and Definition 6095.3 seed the finite family-radius choice route from compact nets and displayed equicontinuity rows to a uniform-modulus consumer.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{EquicontinuityRadiusChoice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, hidden family selectors, countable choice, host function equality, quotient Real equality, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-family exposure, compact-net exposure, equicontinuity-radius exactness, rational-fold exactness, uniform-modulus handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.EquicontinuityRadiusChoiceUp.

Chapter 6096

A Concrete Naming Certificate for $\mathsf{CompleteMetricCriterion}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompleteMetricCriterion}^{\uparrow}$ packet records the finite Cauchy-limit criterion needed before an L10 completion route can hand a Cauchy object to a metric or real-facing consumer. It sits between the filter-base and Cauchy-filter surfaces of chapter 1601 and chapter 333, the completion boundary of chapter 1569, the metric completion surface of chapter 1506, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the criterion as finite BEDC data; it does not quotient filters, select limits, use host equality, invoke countable choice, or assume ambient completeness.

Definition 6096.1 (CompleteMetricCriterion carrier)**.** *A* $\mathsf{CompleteMetricCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, W, T, R, M, L, H, K_0, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row read through a displayed* $\mathsf{FilterBase}^{\uparrow}$ *source and the finite requests admitted by that base;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite windows used when a Cauchy request asks for tail or neighbourhood evidence;*
3. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same finite requests;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback assembled from the scheduled windows and dyadic tolerances;*
5. *M is the* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *completion-boundary row exposed only after C, W, T, R have been replayed;*
6. *L is the candidate* $\mathsf{Real}^{\uparrow}$ *limit seal read from the displayed completion route;*
7. *H records componentwise* hsame *transport for Cauchy-filter, filter-base, window, tolerance, readback, completion, real-seal, provenance, and naming rows;*
8. *K_0 records displayed* Cont *replay of the route*

$$C, W, T, R \longmapsto M \longmapsto L;$$

9. *P is the* Pkg *provenance over* $\mathsf{CompleteMetricCriterion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{FilterBase}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $C, W, T, R, M, L, H, K_0, P, N$. There is no coordinate for quotient filters, selected representatives, selected limits, host equality, countable-choice schedules, or ambient completeness of the metric or real carrier.

Lean4 ✓ `BEDC.Derived.CompleteMetricCriterionUp.CompleteMetricCriterionTasteGate single carrier alignment`

Theorem 6096.2 (CompleteMetricCriterion NameCert obligations)**.** *For every accepted* $\mathsf{CompleteMetricCriterion}^{\uparrow}$ *carrier $K = (C, W, T, R, M, L, H, K_0, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompleteMetricCriterion}^{\uparrow}}$ *surface consists exactly of five obligation rows:*

1. *carrier admission for the displayed Cauchy-filter, filter-base, window, tolerance, readback, completion, real-seal, transport, replay, provenance, and naming rows;*
2. *Cauchy-window exactness, requiring every Cauchy request to be witnessed by the same C, W, T, R rows before completion data are read;*
3. *completion handoff, requiring the* $\mathsf{MetricCompletion}^{\uparrow}$ *or* $\mathsf{CauchyFilterCompletion}^{\uparrow}$ *row M to be downstream from the displayed Cauchy route;*
4. *Real seal nonescape, requiring the limit row L to be read only after M and never through quotient filters, selected limits, host equality, countable choice, or ambient completeness;*
5. *provenance and local naming, requiring H, K_0, P, N to transport, replay, package, and name the same finite route.*

Proof. Carrier admission is projection from Definition 6096.1. The packet lists exactly the Cauchy-filter, filter-base, finite-window, dyadic tolerance, regular readback, completion, real-seal, transport, replay, provenance, and local naming rows.

Cauchy-window exactness is read from C, W, T, R. A Cauchy request first consumes the displayed filter row and finite schedule, then checks the dyadic tolerance ledger, and only then exposes the regular rational readback.

The completion handoff is the row M. Since M is downstream from C, W, T, R, a completion-facing consumer receives the finite Cauchy route rather than a quotient filter or chosen representative.

The real seal L is downstream from M and is preserved by H and K_0. The provenance and local naming rows P, N therefore certify exactly the displayed finite route, while the excluded host principles have no carrier coordinate. □

Theorem 6096.3 (CompleteMetricCriterion Cauchy-limit handoff)**.** *Every limit-facing read exported by an accepted* $\mathsf{CompleteMetricCriterion}^{\uparrow}$ *carrier factors through the finite route*

$$C, W, T, R \longmapsto M \longmapsto L \longmapsto H, K_0, P, N.$$

The route supplies the Cauchy-filter and filter-base request, finite stream window, dyadic tolerance ledger, regular rational readback, completion boundary, real-seal row, transport, replay, provenance, and local naming. It refuses quotient filters, selected limits, host equality, countable choice, and ambient completeness as routes to the same limit.

Proof. Start with an accepted carrier $K = (C, W, T, R, M, L, H, K_0, P, N)$. By Definition 6096.1, the only route to a completion-facing read begins with C, W, T, R.

The row M is the sole completion boundary. It records the $\mathsf{MetricCompletion}^{\uparrow}$ or $\mathsf{CauchyFilterCompletion}^{\uparrow}$ handoff after the finite Cauchy route has been displayed, so the consumer cannot replace the route by a quotient filter or a selected completion point.

The row L is the terminal $\mathsf{Real}^{\uparrow}$ seal. Transport through H and replay through K_0 keep L attached to the same finite request, and P, N package that route as the local certificate.

Since the carrier has no coordinate for host equality, countable choice, or ambient completeness, the handoff proves exactly the finite route stated in the theorem. □

Falsifiable prediction: Any accepted $\mathsf{CompleteMetricCriterion}^{\uparrow}$ read that reaches a $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ seal without the displayed CauchyFilterUp, FilterBaseUp, StreamNameUp, DyadicRatCoreUp, RegSeqRatUp, hsame, Cont, Pkg, and local NameCert rows refutes the carrier.

Independence witness: cauchyfilter, filterbase, streamname, dyadicratcore, regseqrat, metriccompletion, cauchyfiltercompletion, real: none is bijective to $\mathsf{CompleteMetricCriterion}^{\uparrow}$ because this packet jointly carries Cauchy requests, finite windows, dyadic tolerances, regular readbacks, completion handoff, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteMetricCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteMetricCriterion}^{\uparrow}$ packet is generated from a CauchyFilter row over a FilterBase route, StreamName windows, DyadicRatCore tolerances, RegSeqRat readbacks, a MetricCompletion or CauchyFilterCompletion handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteMetricCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6096.1, Definition 6096.2, and Definition 6096.3 seed the finite Cauchy-limit criterion route needed by L10 completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteMetricCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient filters, selected limits, host equality, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window exactness, completion handoff, Real seal nonescape, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.CompleteMetricCriterionUp.

Chapter 6097

A Concrete Naming Certificate for $\mathsf{ClassifierFormingLearning}^{\uparrow}$

The $\mathsf{ClassifierFormingLearning}^{\uparrow}$ packet realizes the classifier-forming learning route of section 9462.14 as finite NameCert data. It records parameter motion, loss observations, classifier shift, training residue, audit export, and TasteGate commitments before any consumer may call the episode certified learning discovery. The packet is downstream from the registry export surface of chapter 1216, the machine-readable audit interface of chapter 1232, the positive interpretability discovery boundary, NameCert governance, and the GroundCompiler audit-map surface of chapter 836.

Definition 6097.1 (ClassifierFormingLearning carrier)**.** *A* $\mathsf{ClassifierFormingLearning}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, A, S, Q, R, G, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the dataset or task row whose accepted observations are read only through displayed histories;*
2. *A is the parameter-motion row recording the before-and-after training state without treating motion itself as discovery;*
3. *S is the classifier-shift row, naming the finite decision boundary or equivalence-class change certified by the packet;*
4. *Q is the loss, score, or benchmark residue row, kept as evidence residue rather than as the learning object;*
5. *R is the registry-export row tying the learning claim to public status and refusal boundaries;*
6. *G is the audit-interface and GroundCompiler map row that makes the packet machine-readable for later checking;*
7. *T is the* TasteGate *row requiring nontriviality and field faithfulness for the displayed classifier-shift carrier;*
8. *H records componentwise* hsame *transport for dataset, parameter, classifier-shift, residue, registry, audit, TasteGate, provenance, and naming rows;*
9. *C records displayed* Cont *replay from training evidence through classifier shift, residue ledger, registry export, audit map, TasteGate, provenance, and local naming;*

10. *P is the* Pkg *provenance over* ClassifierFormingLearning$^{\uparrow}$, RegistryExportSurface$^{\uparrow}$, MachineReadableAuditInterface$^{\uparrow}$ *positive-interpretability-discovery-facing packets,* GroundCompilerAuditMap$^{\uparrow}$, BHist, hsame, Cont, Pkg, TasteGate, *and* NameCert *rows;*

11. *N is the local* NameCert$_{\text{ClassifierFormingLearning}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, A, S, Q, R, G, T, H, C, P, N$. Parameter motion and loss reduction are admissible evidence rows, but they are not identified with learning unless the classifier-shift row S and the TasteGate row T are present.

Theorem 6097.2 (ClassifierFormingLearning NameCert obligations)**.** *For every accepted* ClassifierFormingLearning *carrier $L = (D, A, S, Q, R, G, T, H, C, P, N)$, the local* NameCert$_{\text{ClassifierFormingLearning}^{\uparrow}}$ *surface consists exactly of carrier admission, dataset/task exposure, parameter-motion residue, classifier-shift exposure, loss and benchmark residue, registry export, machine-readable audit handoff, GroundCompiler audit-map handoff, TasteGate nontriviality and field faithfulness commitments, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

No obligation may be discharged by parameter motion alone, lower loss alone, benchmark improvement alone, memorized records, shortcut performance, or unregistered interpretability prose.

Proof. Project the carrier of Definition 6097.1. The displayed inventory is exactly $D, A, S, Q, R, G, T, H, C, P, N$, so every accepted learning claim must be read from those finite rows.

The parameter row A and residue row Q are evidence ledgers. They can support the claim only after the classifier-shift row S is present, because the NameCert surface names classifier formation rather than motion or lower loss by itself.

The registry and audit rows R, G make the claim exportable and machine-readable. The TasteGate row T records that later Lean work must provide nontriviality and field-faithfulness instances for this displayed carrier.

Finally H, C, P, N transport, replay, package, and name the same route. Since the excluded substitutes have no carrier coordinate, they cannot discharge the NameCert obligations. □

Theorem 6097.3 (ClassifierFormingLearning metric nonescape)**.** *Every metric-facing read exported by an accepted* ClassifierFormingLearning$^{\uparrow}$ *carrier factors through*

$$D, A, S, Q, R, G, T, H, C, P, N.$$

Parameter motion, loss reduction, and benchmark movement remain ledger rows inside the packet. They become certified learning discovery only when the classifier-shift row, registry export, machine-readable audit handoff, GroundCompiler audit-map handoff, TasteGate commitments, transport, replay, provenance, and local naming rows are all displayed.

Proof. Begin with a metric-facing read of the carrier. The definition exposes parameter and loss data through A and Q, but those rows are downstream evidence and not the classifier named by the packet.

The classifier-shift row S is the decisive learning row. The route then passes through registry export R and audit handoff G, so the claim remains publicly checkable and machine-readable rather than a private score report.

The TasteGate row T keeps nontriviality and field faithfulness attached to the same carrier. Componentwise transport and continuation replay preserve the rows, and P, N package the accepted route.

Therefore metric improvement cannot escape the packet boundary. Without the classifier-shift and TasteGate rows, the read remains residue rather than a certified ClassifierFormingLearning$^{\uparrow}$ claim. □

Falsifiable prediction: A $\mathsf{ClassifierFormingLearning}^{\uparrow}$ packet with parameter motion and loss reduction but no classifier-shift row is rejected by the NameCert surface.

Independence witness: machine-readable-audit-interface, registry-export-surface, positive-interpretability-discovery-facing packets, namecert, groundcompiler-audit-map: none is bijective to $\mathsf{ClassifierFormingLearning}^{\uparrow}$ because dropping any one row loses a distinct obligation: audit readability, public export, interpretability-facing classifier evidence, local naming, or compiler-audit trace.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ ClassifierFormingLearning}^{\uparrow}$ and $\mathsf{FieldFaithful\ ClassifierFormingL}$ instances in `TasteGate.lean`. Those instances must preserve the rows $D, A, S, Q, R, G, T, H, C, P, N$ so a checked TasteGate theorem can separate classifier shift from parameter motion and loss residue.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierFormingLearning}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierFormingLearning}^{\uparrow}$ packet is generated from dataset/task rows, parameter-motion residue, classifier-shift evidence, loss and benchmark residue, registry export, machine-readable audit handoff, GroundCompiler audit-map handoff, TasteGate commitments, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierFormingLearning}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6097.1, Definition 6097.2, and Definition 6097.3 seed the finite certified classifier-formation route over learning evidence, audit export, TasteGate commitments, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierFormingLearning}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, classifier correctness for a particular model family, benchmark validity, causal explanation, or metric improvement as learning by itself.

Upgrade path: Obligation closure requires checked carrier admission, classifier-shift exposure, metric nonescape, registry export, audit-map handoff, TasteGate nontriviality, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.ClassifierFormingLearningUp.

Chapter 6098

A Concrete Naming Certificate for $\mathsf{LocatedSequenceSupremum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedSequenceSupremum}^{\uparrow}$ packet records the finite BEDC surface for taking a supremum of a located bounded real sequence. It sits over the sequence boundary of chapter 1415, the regular rational readback of chapter 301, the real-name seal of chapter 38, and the located-upper-bound route of chapter 1427. The packet is sequence-level supremum data; it does not assert Dedekind completeness for arbitrary subsets, open-cover compactness, a selected maximizing index, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6098.1 (Located sequence supremum carrier)**.** *A* $\mathsf{LocatedSequenceSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, W, R, L, U, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *source row, carrying the bounded sequence surface to be inspected.*
2. *W is the finite window ledger listing the sequence entries whose bound and approximation requests are currently visible.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed sequence windows.*
4. *L is the located upper-bound row, recording finite evidence that requested rational cuts are either below some displayed sequence value or above a displayed upper bound.*
5. *U is the* $\mathsf{LocatedSupremum}^{\uparrow}$ *handoff row, read only after B, W, R, L have been exposed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row naming the supremum candidate after the located sequence data have been displayed.*
7. *H records componentwise* hsame *transport for the source, window, readback, located-bound, handoff, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a rational bound request through the same rows.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedSequenceSupremum}^{\uparrow}$, $\mathsf{BoundedRealSequence}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceSupremum}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary subset completeness, a Dedekind-completeness principle, a selected maximizing index, quotient equality of real names, countable choice, or host-level supremum.

Lean4 ✓ `BEDC.Derived.LocatedSequenceSupremumUp.LocatedSequenceSupremumTasteGate single carrier alignment`

Theorem 6098.2 (Located sequence supremum NameCert obligations)**.** *For every accepted* $\mathsf{LocatedSequenceSupre}$ *carrier* $S = (B, W, R, L, U, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedSequenceSupremum}^{\uparrow}}$ *surface consists exactly of carrier admission for the bounded sequence source, finite windows, regular rational readback, located upper-bound row, located-supremum handoff, real seal, transport, replay, provenance, and local naming; window coverage requiring every supremum request to read* W *before* R*; locatedness requiring* L *to answer rational cut requests before* U *is consumed; and nonescape excluding arbitrary subset completeness, selected maximizers, quotient real equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedSequenceSupremumUp.LocatedSequenceSupremumCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 6098.1. The accepted packet displays exactly $B, W, R, L, U, E, H, C, P, N$.

The sequence read begins with B and W. The bounded real sequence row supplies the source, while the finite window ledger specifies which sequence entries are available for the current rational request.

The regular readback and locatedness rows are R and L. They convert the displayed windows into $\mathsf{RegSeqRat}^{\uparrow}$ evidence and answer the lower-or-upper side of the located cut request before the $\mathsf{LocatedSupremum}^{\uparrow}$ handoff row U is exposed.

Finally, E, H, C, P, N seal, transport, replay, source, and name the same data. Since the carrier has no coordinate for arbitrary subset data, selected maximizers, quotient equality, countable choice, or ambient completeness, the listed obligations exhaust the NameCert surface. □

Theorem 6098.3 (Located sequence supremum Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedSequenceSupremum}^{\uparrow}$ *carrier receives the supremum candidate only through the route*

$$\mathsf{BoundedRealSequence}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedSupremum}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the real seal E *is read only after the bounded sequence row* B*, finite window ledger* W*, regular readback row* R*, located upper-bound row* L*, and located-supremum handoff* U *have been displayed through* H, C, P, N*.*

Proof. Start from an accepted carrier $S = (B, W, R, L, U, E, H, C, P, N)$. A $\mathsf{Real}^{\uparrow}$ consumer has access to the real seal only at the row E.

The row E is downstream of the displayed sequence route. The consumer must first read the bounded source B, inspect finite windows W, convert them through R, and consume the located cut evidence L before the $\mathsf{LocatedSupremum}^{\uparrow}$ handoff U names a supremum candidate.

Transport and replay through H and C preserve that order, while P, N only record provenance and local naming. Thus the handoff reaches $\mathsf{Real}^{\uparrow}$ without importing an arbitrary-subset completeness principle or a selected maximizing index. □

Lean4 ✓ `BEDC.Derived.LocatedSequenceSupremumUp.LocatedSequenceSupremumCarrier real handoff`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedSequenceSupremum}^{\uparrow}}$**.**

Constructive story: A LocatedSequenceSupremum$^{\uparrow}$ packet is generated from a bounded real sequence row, finite sequence windows, RegSeqRat readback, a located upper-bound row, a LocatedSupremum handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedSequenceSupremum}^{\uparrow}}$, seed)

Scope closed: Definition 6098.1, Definition 6098.2, and Definition 6098.3 seed the sequence-level located supremum route over bounded real sequences and Real-facing handoff rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedSequenceSupremum}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Dedekind completeness, arbitrary subset suprema, selected maximizing indices, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires carrier admission, finite-window coverage, RegSeqRat readback, located cut response, LocatedSupremum handoff, Real seal preservation, transport, replay, provenance, and local naming for BEDC.Derived.LocatedSequenceSupremumUp.

Chapter 6099

A Concrete Naming Certificate for $\mathsf{ScottContinuous}^{\uparrow}$

Following Scott's domain-theoretic treatment of continuous maps and the later domain-theory accounts of directed-complete orders, the $\mathsf{ScottContinuous}^{\uparrow}$ packet records morphisms that preserve directed suprema and Scott-open observation as finite BEDC naming data. It sits over the domain carrier of chapter 4164, the directed-complete order surface of chapter 5905, the lattice-facing supremum discipline of chapter 5155, and the Scott-open interface of chapter 5950. The packet names a displayed continuity carrier; it does not assert a host function space, quotient map equality, selected pointwise supremum proof, arbitrary topological inverse image, or semantic completeness theorem.

Definition 6099.1 (ScottContinuous carrier)**.** *A* $\mathsf{ScottContinuous}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, T, G, D, O, H, C, P, N)$$

with the following displayed rows:

1. *S is the source* $\mathsf{DCPO}^{\uparrow}$ *or* $\mathsf{ScottDomain}^{\uparrow}$ *row, exposing the source order, directed-family schedule, supremum row, compact-basis data when present, and local NameCert surface.*
2. *T is the target* $\mathsf{DCPO}^{\uparrow}$ *or* $\mathsf{ScottDomain}^{\uparrow}$ *row, exposing the target order, directed-family schedule, supremum row, compact-basis data when present, and local NameCert surface.*
3. *G is the graph row, recording the finite source-target read used by the continuity packet rather than a host function object.*
4. *D is the directed-supremum preservation ledger. For every displayed directed schedule admitted by S, it records that the graph read in G carries the source supremum row to the target supremum row visible in T.*
5. *O is the Scott-open inverse-image row. Every accepted Scott-open observation on the target is replayed through chapter 5950 as a Scott-open observation on the source.*
6. *H records componentwise* hsame *transport for the source row, target row, graph row, directed-supremum ledger, inverse-image row, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a source read, target read, directed-supremum preservation request, or Scott-open inverse-image request through S, T, G, D, O, H.*

8. *P is the* Pkg *provenance over* $\mathsf{ScottContinuous}^{\uparrow}$, $\mathsf{ScottDomain}^{\uparrow}$, $\mathsf{DCPO}^{\uparrow}$, $\mathsf{CompleteLattice}^{\uparrow}$, $\mathsf{ScottTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{ScottContinuous}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, T, G, D, O, H, C, P, N$. It has no coordinate for a host exponential object, quotient equality of maps, pointwise choice of directed suprema, arbitrary inverse-image topology, or unrecorded semantic continuity proof.

Lean4 ✓ `BEDC.Derived.ScottContinuousUp.ScottContinuousTasteGate single carrier alignment`

Theorem 6099.2 (ScottContinuous NameCert obligations)**.** *For every accepted* $\mathsf{ScottContinuous}^{\uparrow}$ *carrier $F = (S, T, G, D, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ScottContinuous}^{\uparrow}}$ *surface consists exactly of source admission, target admission, graph-row admission, directed-supremum preservation through D, Scott-open inverse-image replay through O, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every continuity consumer must read $S, T, G, D, O, H, C, P, N$ before treating the morphism as public.*

Proof. Project the carrier of Definition 6099.1. The source and target order data are S and T, the finite morphism presentation is G, the continuity ledgers are D and O, and the structural rows are H, C, P, N.

The directed-supremum obligation reads the directed schedule and supremum row exposed by S, then checks the target supremum row exposed by T through the graph row G. Thus preservation is attached to displayed dcpo or domain data and not to a host function space.

The Scott-open obligation reads target opens through the Scott-topology packet and replays their inverse image as source Scott-open data. The replay uses the same graph row and the same directed-supremum ledger, so it cannot introduce an arbitrary topological inverse-image operation.

Rows H, C, P, N transport, replay, source, and name this finite packet. Since quotient map equality, selected supremum witnesses, host exponentials, and semantic completeness are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6099.3 (ScottContinuous directed-supremum stability)**.** *Every accepted* $\mathsf{ScottContinuous}^{\uparrow}$ *packet preserves directed-supremum visibility under componentwise transport and continuation replay. If a consumer reads a directed source schedule through S and the preservation ledger D, any transported or replayed read of the same continuity packet factors again through*

$$S, T, G, D, O, H, C, P, N.$$

Thus directed-supremum preservation remains a finite ledger over displayed dcpo or domain rows and never exposes a host function equality, selected pointwise lub proof, or hidden target supremum oracle.

Proof. Start with a directed-supremum preservation read. By Definition 6099.1, the read reaches D only after the source row S, target row T, and graph row G are displayed.

Use Definition 6099.2. Componentwise transport through H preserves the same carrier rows, and Cont replay through C repeats the displayed route rather than generating a second continuity source.

The provenance and local naming rows P, N identify the same packet after the replay. Because selected pointwise lub proofs, host function equality, and hidden supremum oracles are not coordinates of $S, T, G, D, O, H, C, P, N$, the transported read remains the same finite directed-supremum ledger. □

Theorem 6099.4 (ScottContinuous domain handoff)**.** *Let $F = (S, T, G, D, O, H, C, P, N)$ be an accepted* $\mathsf{ScottContinuous}^{\uparrow}$ *carrier whose source and target rows are* $\mathsf{ScottDomain}^{\uparrow}$ *packets. Any downstream domain-semantic consumer of F must first read the source domain row, target domain row, graph row, directed-supremum preservation ledger, Scott-open inverse-image row, transport, replay, provenance, and local naming rows. The handoff exports no host domain morphism, quotient map equality, selected basis enumeration, or ambient semantic interpretation.*

Proof. Project the source and target coordinates. The domain-facing information available to the consumer is precisely the pair of $\mathsf{ScottDomain}^{\uparrow}$ rows S, T plus the graph and continuity rows G, D, O.

Apply the NameCert obligations of Definition 6099.2. They force every public continuity read through $S, T, G, D, O, H, C, P, N$ before a downstream consumer can use it as a morphism of domain packets.

Finally use the Scott-domain and Scott-topology sibling surfaces cited at the chapter start. They provide directed-supremum and Scott-open observation rows as finite NameCert data. Since the continuity carrier merely links those rows and lacks a host morphism coordinate, the handoff is the displayed finite packet rather than an ambient semantic interpretation. □

Closure status for $\mathsf{NameCert}_{\mathsf{ScottContinuous}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ScottContinuous}^{\uparrow}$ packet is generated from source and target dcpo or domain rows, a graph row, directed-supremum preservation, Scott-open inverse-image replay, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ScottContinuous}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6099.1, Definition 6099.2, Definition 6099.3, and Definition 6099.4 seed the finite Scott-continuity interface over dcpo, domain, complete-lattice, and Scott-topology rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ScottContinuous}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function spaces, quotient map equality, selected directed-supremum witnesses, arbitrary inverse-image topology, semantic completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-supremum preservation, Scott-open inverse-image replay, transport, replay, provenance, local naming, and a concrete Lean target for BEDC.Derived.ScottContinuousUp.

Chapter 6100

A Concrete Naming Certificate for $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$

The $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$ packet realizes the decomposition surface of section 9462.18 as finite NameCert data. It sits beside the classifier-preserving route of chapter 590, the classifier-typing surface of chapter 489, the self-classifier row of chapter 871, and the smuggling-pattern refusal classifier of chapter 1329. The packet names a certificate-deficit carrier for supported behavior; it is not a host interpretability theorem, not a semantic explanation oracle, not a model-internals extractor, and not a verification-hardening certificate.

Definition 6100.1 (BlackBoxDecompositionCertificate carrier)**.** *A* $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (B, S, C, F, M, L, D, E, H, R, P, N)$$

with the following displayed rows:

1. *B is the behavior row, recording the displayed behavior, scope, evidence packet, observation protocol, and stability claim being decomposed.*
2. *S is the support-candidate row, naming the classifier candidate that supports the behavior through prediction, correlation, intervention, formal constraint, or hybrid evidence.*
3. *C is the classifier row, read through* $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{ClassifierMorphism}^{\uparrow}$*, or* $\mathsf{TheorySelfClassifier}^{\uparrow}$ *before the support claim is public.*
4. *F is the certificate-field row, listing source, pattern, classifier, stability, and ledger-policy fields visible for the support claim.*
5. *M is the missing-field row, recording exactly which certificate fields are absent or incomplete for the declared scope.*
6. *L is the ledger row, recording attached ledger evidence, required residue, recorded residue, and ledger-policy status.*
7. *D is the deficit row, measuring unrecorded or incomplete critical residue in the decomposition.*
8. *E is the closed-explanation state row, requiring NameCert-completeness, ledger-completeness, scope seal, and not-claimed boundary before a behavior is no longer black-box relative to the support candidate.*

9. *H records componentwise* hsame *transport for behavior, support, classifier, field, missing-field, ledger, deficit, explanation-state, replay, provenance, and naming rows.*
10. *R records displayed* Cont *replay from a black-box query through $B, S, C, F, M, L, D, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$*, section 9462.18,* $\mathsf{ClassifierTyping}^{\uparrow}$*,* $\mathsf{ClassifierMorphism}^{\uparrow}$*,* $\mathsf{TheorySelfClassifier}^{\uparrow}$*,* $\mathsf{SmugglingPatternClassifier}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, S, C, F, M, L, D, E, H, R, P, N$. It has no coordinate for hidden model semantics, unrecorded classifier support, host explanation equality, oracle access to internal weights, or a formal-verification verdict.

Theorem 6100.2 (BlackBoxDecompositionCertificate NameCert obligations)**.** *For every accepted* $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$ *carrier $Q = (B, S, C, F, M, L, D, E, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BlackBoxDecomp}}$ *surface consists exactly of behavior admission, support-candidate admission, classifier-row exposure, certificate-field exposure, missing-field exposure, ledger exposure, deficit exposure, closed-explanation-state exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream black-box-decomposition consumer must read all twelve rows before claiming that a behavior has reached closed-explanation state.*

Proof. Project the carrier of Definition 6100.1. The behavior and support rows are B, S, the classifier-facing row is C, the certificate and ledger rows are F, M, L, D, E, and the structural rows are H, R, P, N.

The classifier row is read through the sibling classifier surfaces named in the chapter introduction. Thus support for the behavior cannot be treated as an unrecorded semantic relation or host interpretability judgment.

The field, missing-field, ledger, and deficit rows separate certificate completeness from ledger completeness. The closed-explanation state row E can be public only after those rows are displayed together with the scope seal and not-claimed boundary.

Rows H, R, P, N transport, replay, source, and name the same finite packet. Since hidden model semantics, internal-weight oracles, host explanation equality, and verification-hardening verdicts are absent from the carrier, these rows exhaust the local NameCert obligations. □

Theorem 6100.3 (BlackBoxDecompositionCertificate deficit non-closure)**.** *Let $Q = (B, S, C, F, M, L, D, E, H, R,$ be an accepted* $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$ *carrier. If the missing-field row M is empty and the deficit row D records nonempty ledger deficit, then the closed-explanation state row E cannot be read as satisfied. The packet remains a ledger-deficit black-box certificate rather than a closed explanation.*

Proof. Read the carrier rows. Empty M records certificate-field completeness, while nonempty D records remaining ledger deficit for the same support claim.

Use Definition 6100.2. The closed-explanation state row E requires both certificate completeness and ledger completeness before a consumer can leave black-box status on the sealed scope.

The hypotheses supply only the first condition and deny the second. Therefore the accepted packet cannot expose E as satisfied; it remains a certificate with ledger deficit, not a closed explanation. □

Theorem 6100.4 (BlackBoxDecompositionCertificate classifier handoff)**.** *Every accepted* BlackBoxDecompositionC *packet hands its support-candidate row to downstream consumers through the displayed classifier rows C, H, R, P, N. The handoff may cite classifier typing, classifier morphism, theory self-classification, and smuggling-pattern refusal surfaces, but it exports no host interpretability relation, hidden model semantics, selected explanation object, or formal-verification verdict.*

Proof. Project the classifier row C. By Definition 6100.1, it is the only row that connects the behavior support candidate to classifier-facing siblings.

Apply the NameCert obligations of Definition 6100.2. They require the classifier row to remain attached to behavior, support, field, missing-field, ledger, deficit, and explanation-state rows before any public handoff.

Finally use the classifier-morphism, classifier-typing, theory-self-classifier, and smuggling-pattern-classifier sibling references in the chapter introduction. They provide finite classifier and refusal surfaces, not semantic access to model internals. Thus the handoff is a finite NameCert route through C, H, R, P, N and not a hidden explanation oracle. □

Falsifiable prediction: Any accepted BlackBoxDecompositionCertificateUp packet with empty missing-field family and nonempty ledger deficit cannot satisfy closed-explanation state.

Independence witness: classifiertyping, classifiermorphism, theoryselfclassifier, and smuggling-patternclassifier are non-bijective siblings: none alone carries the complete behavior, support-candidate, certificate-field, missing-field, ledger-deficit, closed-explanation-state, transport, replay, provenance, and NameCert surface exposed by $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$.

The Lean follow-up must provide $\mathsf{Nontrivial\ BlackBoxDecompositionCertificate}^{\uparrow}$ and FieldFaithful BlackBoxDecom instances for the displayed carrier before any checked TasteGate marker can support a closure upgrade.

Closure status for $\mathsf{NameCert}_{\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}$ packet is generated from behavior, support-candidate, classifier, certificate-field, missing-field, ledger, deficit, closed-explanation-state, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6100.1, Definition 6100.2, Definition 6100.3, and Definition 6100.4 seed the finite certificate-deficit surface for black-box decomposition.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BlackBoxDecompositionCertificate}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close hidden model semantics, host interpretability relations, selected explanation objects, formal-verification verdicts, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier-row exposure, missing-field exactness, ledger-deficit non-closure, closed-explanation-state boundary, transport, replay, provenance, local naming, Nontrivial, and FieldFaithful rows for BEDC.Derived.BlackBoxDecompositionCertificateUp.

Chapter 6101

A Concrete Naming Certificate for $\mathsf{BishopLocatedModulus}^{\uparrow}$

Following Bishop and Bridges on located constructive reals and explicit moduli for Cauchy approximation, the $\mathsf{BishopLocatedModulus}^{\uparrow}$ packet records the finite BEDC surface at which a located comparison row supplies an inspectable modulus schedule. It sits after the located-real boundary of chapter 1933, the Cauchy-modulus ledger of chapter 292, the stream-name observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the constructive route from located comparison data to an explicit modulus row; it does not assert trichotomy, a selected real representative, quotient stream equality, ambient completeness, or a host least-bound principle.

Definition 6101.1 (Bishop located-modulus carrier)**.** *A* $\mathsf{BishopLocatedModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (L, \mu, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *comparison row, recording the finite apartness or interval-decision data available to the modulus request.*
2. *μ is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row extracted from that located comparison surface.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row opened by μ at the requested tolerance.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the inspected regular rational window.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that indexes the requested precision and bounds the inspected stream window.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after L, μ, S, R, D have been displayed.*
7. *H records componentwise* hsame *transport for located comparison, modulus, stream-window, regular readback, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the located comparison request through modulus extraction, finite stream inspection, regular rational readback, dyadic tolerance, and the terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedModulus}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host real value, countable choice over tolerances, quotient stream equality, trichotomy, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BishopLocatedModulusUp.BishopLocatedModulusTasteGate single carrier alignment`

Theorem 6101.2 (Bishop located-modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedModulus}^{\uparrow}$ *carrier*

$$M = (L, \mu, S, R, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopLocatedModulus}^{\uparrow}}$ *surface consists exactly of located comparison exposure through* L*, explicit Cauchy-modulus exposure through* μ*, finite stream-window admission through* S*, regular rational readback through* R*, dyadic tolerance exactness through* D*, terminal Real seal non-escape through* E*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Proof. Project the accepted carrier. The tuple displays the located comparison row, the extracted modulus schedule, the finite stream window, the regular rational readback, the dyadic tolerance ledger, the terminal Real seal, and the structural rows.

The analytic obligations are ordered by the carrier. A consumer first reads L, then the schedule μ, then the window S, readback R, and tolerance ledger D before the Real seal E may be consumed.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. Since the carrier has no coordinate for host real values, tolerance choice, quotient stream equality, trichotomy, ambient completeness, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 6101.3 (Bishop located-modulus Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopLocatedModulus}^{\uparrow}$ *packet must factor through the displayed route*

$$\mathsf{LocatedReal}^{\uparrow} \longrightarrow \mathsf{CauchyModulus}^{\uparrow} \longrightarrow \mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before it can use the local certificate. The route exports an explicit finite modulus schedule and its inspected rational windows; it exports no selected representative, quotient real equality, countable-choice selector, trichotomy row, or ambient completeness witness.

Proof. Begin with the located comparison row L. It is the only source of comparison data admitted by the carrier, so the consumer cannot begin from a host real value or a detached order decision.

Next read the modulus and tolerance rows. The schedule μ opens only the finite windows requested by the dyadic ledger D, and S records those windows as StreamName data.

Then pass through the regular readback and seal. The row R transports the inspected window into the regular rational route, and E seals the result as $\mathsf{Real}^{\uparrow}$ only after the displayed route has been read.

Finally use H, C, P, N to preserve the same finite package. No row in the carrier contains a selected representative, quotient equality, choice selector, trichotomy, or ambient completeness witness, so the Real-seal boundary cannot export any of those objects. □

Falsifiable prediction: A $\mathsf{BishopLocatedModulus}^{\uparrow}$ packet is rejected whenever a Real or completion-facing consumer claims a modulus without displaying the located comparison row, CauchyModulus schedule, DyadicRatCore tolerance, StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local NameCert rows.

Independence witness: located-real, cauchymodulus, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{BishopLocatedModulus}^{\uparrow}$ because this packet jointly carries the located comparison source, explicit modulus schedule, dyadic tolerance, finite observation window, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedModulus}^{\uparrow}$ packet is generated from a LocatedReal comparison row, CauchyModulus schedule, StreamName finite window, RegSeqRat readback, DyadicRatCore tolerance ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6101.1, Definition 6101.2, and Definition 6101.3 seed the finite located-modulus route from located comparison data to explicit modulus schedules and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected real representatives, quotient real equality, countable choice over tolerances, trichotomy, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located comparison exposure, modulus extraction, dyadic tolerance exactness, stream-window admission, RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedModulusUp.

Chapter 6102

A Concrete Naming Certificate for $\mathsf{FiniteApartnessNet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteApartnessNet}^{\uparrow}$ packet records the finite apartness-net certificate used when a Real or completion-facing argument needs finite local separation without open-cover compactness. It reads the dyadic tolerance ledger of chapter 291, the finite observation schedule of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet is a finite certificate surface; it does not assert host compactness, quotient equality of real names, selected representatives, countable choice, or ambient completeness.

Definition 6102.1 (Finite apartness-net carrier)**.** *A* $\mathsf{FiniteApartnessNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (C, D, A, V, H, K, P, N)$$

with the following displayed rows:

1. *C is the finite center row, listing the displayed centers through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback data.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius row, recording dyadic radii, tolerance comparisons, and the positive-radius ledger used by the net.*
3. *A is the pairwise apartness ledger, certifying apartness between distinct displayed centers at the requested dyadic scale.*
4. *V is the coverage ledger, recording that every accepted finite request is routed to a displayed center and radius before any Real-facing readback is exported.*
5. *H records componentwise* hsame *transport for centers, dyadic radii, apartness rows, coverage rows, replay, provenance, and local naming.*
6. *K records displayed* Cont *replay from a tolerance request through the same center, radius, apartness, and coverage rows.*
7. *P is the* Pkg *provenance over* $\mathsf{FiniteApartnessNet}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteApartnessNet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through C, D, A, V, H, K, P, N. It has no coordinate for an open cover, a finite-subcover theorem, quotient equality of real names, a selected representative, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.FiniteApartnessNetUp.FiniteApartnessNetTasteGate single carrier alignment`

Theorem 6102.2 (Finite apartness-net NameCert obligations)**.** *For every accepted* $\mathsf{FiniteApartnessNet}^{\uparrow}$ *carrier $F = (C, D, A, V, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteApartnessNet}^{\uparrow}}$ *surface consists exactly of carrier admission for finite centers, dyadic radii, pairwise apartness, coverage, transport, replay, provenance, and local naming; radius admission through* $\mathsf{DyadicRatCore}^{\uparrow}$ *before apartness is read; coverage admission requiring each accepted request to expose a center and radius before a* $\mathsf{Real}^{\uparrow}$ *consumer sees the result; and nonescape excluding open-cover compactness, selected representatives, quotient real equality, countable choice, host real equality, and ambient completeness.*

Proof. Carrier admission is projection from Definition 6102.1. The accepted packet displays exactly C, D, A, V, H, K, P, N.

The radius and apartness obligations are read before coverage. The dyadic row D supplies the visible scale, while the apartness ledger A compares only the displayed finite centers in C at that scale.

Coverage is the row V. It can answer a finite request only by pointing back to a displayed center and dyadic radius, so a Real-facing consumer cannot read a hidden finite-subcover theorem or a selected host representative.

Finally, H, K, P, N transport, replay, source, and name the same data. Since the carrier has no coordinate for open covers, quotient real equality, countable choice, host real equality, or ambient completeness, the listed obligations exhaust the certificate surface. □

Theorem 6102.3 (Finite apartness-net RegSeqRat consumer)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{FiniteApartnessNet}^{\uparrow}$ *carrier must pass through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, finite centers are read as stream names with regular rational windows, dyadic radii supply the tolerance scale, the pairwise apartness and coverage ledgers are exposed before the Real seal, and no consumer receives open-cover compactness, countable choice, quotient equality, selected representatives, host real equality, or ambient completeness.

Proof. Start from an accepted carrier $F = (C, D, A, V, H, K, P, N)$. A regular-rational consumer can see only the centers named by C and the dyadic scale named by D.

The route then exposes the finite ledgers. Pairwise apartness is read through A, and coverage is read through V only after the displayed center and radius rows have fixed the current tolerance request.

Transport and replay through H and K preserve that route, while P and N record provenance and local naming. Therefore every RegSeqRat or Real-facing read is generated by the displayed finite-net data and cannot import the excluded host principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteApartnessNet}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteApartnessNet}^{\uparrow}$ packet is generated from finite centers, dyadic radii, pairwise apartness, coverage, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteApartnessNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6102.1, Definition 6102.2, and Definition 6102.3 seed the finite apartness-net route from dyadic tolerances and stream centers through RegSeqRat readback to Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteApartnessNet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, finite-subcover extraction, quotient real equality, selected representatives, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires center admission, dyadic-radius admission, pairwise apartness, coverage, transport, replay, provenance, and local naming for BEDC.Derived.FiniteApartnessNetUp.

Chapter 6103

A Concrete Naming Certificate for $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$

Following de Bruijn's index discipline and Barendregt's account of substitution under binders, the $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$ packet records the finite BEDC surface for the closed substitution audit fragment. It lands the audit map of chapter 9380 as a concrete NameCert spine and reads the closed-boundary packet of chapter 527. The packet is a closed-fragment substitution spine; it does not assert unrestricted substitution composition, subject reduction, host syntax quotients, hidden evaluator state, or compiler correctness.

Definition 6103.1 (MetaCIC substitution spine carrier)**.** *A* $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (T, V, D, Q, R, L, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the closed MetaCIC term row, listing the displayed source, payload, and boundary terms.*
2. *V is the de Bruijn variable ledger for the finite indices inspected by shift and substitution.*
3. *D is the cutoff-depth row recording the boundary at which closedness is tested.*
4. *Q is the closedness classifier row for the source and payload terms.*
5. *R is the closed shift row, exposed only when the closedness classifier has already been read.*
6. *L is the closed substitution row, exposed only through the same closedness classifier and depth row.*
7. *A is the closed composition audit row, recording the checked closed-fragment composition and swap surfaces.*
8. *H records componentwise* hsame *transport for terms, indices, depth, closedness, shift, substitution, composition, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay for closed substitution requests through the same rows.*
10. *P is the* Pkg *provenance over* $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$*,* $\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpine}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $T, V, D, Q, R, L, A, H, C, P, N$. It has no coordinate for unrestricted substitution algebra, typing preservation, subject reduction, host syntax equality, quotient terms, hidden evaluator state, or compiler correctness.

Lean4 ✓ `BEDC.Derived.MetaCICSubstitutionSpineUp.MetaCICSubstitutionSpineTasteGate single carrier alignment`

Theorem 6103.2 (MetaCIC substitution spine NameCert obligations)**.** *For every accepted* MetaCICSubstitutionSpi *carrier $S = (T, V, D, Q, R, L, A, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpine}^{\uparrow}}$ *surface consists exactly of carrier admission for displayed terms, de Bruijn indices, cutoff depth, closedness, closed shift, closed substitution, closed composition audit, transport, replay, provenance, and local naming; shift and substitution admission requiring Q and D before R or L is visible; closed-composition admission requiring A to read only the checked closed-fragment rows; and nonescape excluding unrestricted substitution composition, subject reduction, host syntax quotients, hidden evaluator state, and compiler correctness.*

Lean4 ✓ `BEDC.Derived.MetaCICSubstitutionSpineUp.MetaCICSubstitutionSpineCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 6103.1. The accepted packet displays exactly $T, V, D, Q, R, L, A, H, C, P, N$.

Closedness gates the operation rows. The rows Q and D must be exposed before the closed shift row R or the closed substitution row L can be read.

The audit row A records only the closed composition surface. It reads the already displayed operation rows and therefore cannot be reinterpreted as unrestricted substitution algebra or as a subject-reduction theorem.

Finally, H, C, P, N preserve, replay, source, and name the same finite spine. Since the carrier has no coordinate for host syntax quotients, hidden evaluator state, or compiler correctness, the listed obligations exhaust the NameCert surface. □

Theorem 6103.3 (MetaCIC substitution spine closed-term consumer)**.** *Every closed-term consumer of an accepted* $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{ClosedTermSubstitutionBoundary}^{\uparrow} \longmapsto T, V, D, Q, R, L, A, H, C, P, N.$$

Thus a consumer may read closed shift, closed substitution, and closed composition audit data only after the closed boundary, de Bruijn index, cutoff-depth, and closedness rows have been displayed, and cannot import subject reduction, host syntax equality, quotient terms, hidden evaluator state, or compiler correctness.

Proof. Start with the closed boundary supplied by chapter 527. It gives the displayed term, closedness, shift, substitution, ledger, audit, transport, replay, provenance, and naming rows that this spine specializes.

Project the carrier rows T, V, D, Q, R, L, A. These rows force a closed-term consumer to read the de Bruijn index and cutoff-depth data before the operation rows are exposed.

Transport, replay, provenance, and local naming are then read through H, C, P, N. They preserve the same finite route, so the consumer receives only closed-fragment substitution spine data and none of the excluded host-level principles. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICSubstitutionSpine}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICSubstitutionSpine}^{\uparrow}$ packet is generated from closed MetaCIC term rows, de Bruijn index rows, cutoff-depth rows, closedness classifiers, closed shift rows, closed substitution rows, closed composition audit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{MetaCICSubstitutionSpine}\uparrow}$, seed)

Scope closed: Definition 6103.1, Definition 6103.2, and Definition 6103.3 seed the closed substitution, shift, and composition audit spine over the closed-term boundary packet.

Formal verification: FormalStatus(NameCert$_{\mathsf{MetaCICSubstitutionSpine}\uparrow}$, theoremChecked)

Lean target: `BEDC.Derived.MetaCICSubstitutionSpineUp.MetaCICSubstitutionSpineCarrier\_namecert\_obligations`

Bridge status: none

Not claimed: This chapter does not close unrestricted substitution algebra, subject reduction, typing preservation, host syntax quotients, hidden evaluator state, compiler correctness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires term admission, de Bruijn index admission, cutoff-depth admission, closedness gating, closed shift, closed substitution, closed composition audit, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICSubstitutionSpineUp.

Chapter 6104

A Concrete Naming Certificate for $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$

The $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ packet realizes the observer-expansion reading of chapter 9463 as a finite ledger for geodesic-flow windows and boundary-shadow readout. It sits after the geodesic-flow carrier of chapter 3856, the horosphere-flow ledger of chapter 4051, the boundary-shadow packet of chapter 3813, the visual-metric packet of chapter 3866, and the terminal real-name seal of chapter 38. The same finite geodesic-flow window is a downstream input for chapter 4203 when laminations read compatible flow windows and boundary-shadow rows. The packet names a finite route from geodesic flow windows to boundary-shadow ledger rows; it does not assert a host hyperbolic manifold, quotient geodesic equality, invariant measure, differentiable flow theorem, or global visual boundary.

Definition 6104.1 (Hyperbolic geodesic-flow ledger carrier)**.** *A* $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, O, B, V, W, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the finite geodesic-flow window admitted by the packet.*
2. *O is the* $\mathsf{HorosphereFlowLedger}^{\uparrow}$ *row, recording the horosphere-side ledger that stabilizes the same window near the boundary.*
3. *B is the* $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$ *row, recording the boundary-shadow aperture consumed by the flow ledger.*
4. *V is the* $\mathsf{HyperbolicVisualMetric}^{\uparrow}$ *row, displaying the visual comparison used to read the boundary shadow.*
5. *W is the flow-window exactness row, tying the geodesic-flow segment, horosphere ledger, boundary shadow, and visual metric into one finite route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for any displayed comparison value exported by the ledger.*
7. *H records componentwise* hsame *transport for flow, horosphere, boundary-shadow, visual-metric, window-exactness, Real-seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from flow-window admission through horosphere stabilization, boundary-shadow readback, visual-metric comparison, and the terminal Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HorosphereFlowLedger}^{\uparrow}$, $\mathsf{HyperbolicBoundaryShadow}^{\uparrow}$, $\mathsf{HyperbolicVisualMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host hyperbolic manifold, quotient geodesic equality, invariant measure, differentiable flow theorem, global visual boundary, Lean verification, or bridge checking.

Theorem 6104.2 (Hyperbolic geodesic-flow ledger NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ *carrier*

$$G = (F, O, B, V, W, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}}$ *surface consists exactly of geodesic-flow window exposure through F, horosphere-ledger stabilization through O, boundary-shadow readback through B, visual-metric comparison through V, window exactness through W, Real-seal non-escape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier. The tuple displays the flow window, horosphere ledger, boundary shadow, visual comparison, exactness row, Real seal, and structural rows.

The geometric obligations are finite. Row F admits the geodesic-flow segment, row O stabilizes that segment through the horosphere ledger, row B reads the boundary shadow, and row V supplies the visual comparison.

The row W binds these readings into one window-exact route before the Real seal E is available. The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming.

Since no carrier coordinate stores a host manifold, quotient geodesic equality, invariant measure, differentiable flow theorem, global visual boundary, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 6104.3 (Hyperbolic geodesic-flow boundary-shadow route)**.** *Every observer-expansion consumer of an accepted* $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ *packet must factor through*

$$\mathsf{HyperbolicGeodesicFlow}^{\uparrow} \longrightarrow \mathsf{HorosphereFlowLedger}^{\uparrow} \longrightarrow \mathsf{HyperbolicBoundaryShadow}^{\uparrow} \longrightarrow \mathsf{HyperbolicVisualMetric}^{\uparrow}$$

before it can export a flow comparison value. The route exports only the displayed finite flow, horosphere, boundary-shadow, visual-metric, Real-seal, transport, replay, provenance, and local naming rows.

Proof. Begin with the flow-window row F. It is the only geodesic-flow segment admitted by the carrier, so the consumer cannot read a flow value from a host manifold or global flow object.

Next pass through O, B, V. The horosphere ledger records the boundary-near stabilization, the boundary-shadow row supplies the aperture, and the visual-metric row supplies the comparison surface.

Then use W and E. The exactness row binds the same flow segment to the same horosphere and shadow rows, and the Real seal is consumed only after that finite route has been displayed.

Finally apply the structural rows H, C, P, N. They preserve the route as packet data and add no coordinate for quotient geodesic equality, invariant measure, differentiable flow, or global visual boundary. □

Falsifiable prediction: A $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ packet is rejected whenever an observer-expansion consumer claims a flow comparison without displaying geodesic-flow, horosphere-flow-ledger, boundary-shadow, visual-metric, Real-seal, transport, replay, provenance, and local NameCert rows.

Independence witness: horosphere-flow-ledger, hyperbolic-boundary-shadow: $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ is not bijective to either sibling because it jointly carries the geodesic-flow window, horosphere stabilization ledger, boundary-shadow aperture, visual comparison, Real seal, transport, replay, provenance, and local naming rows in one flow-ledger packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}$ packet is generated from a HyperbolicGeodesicFlow window, HorosphereFlowLedger row, HyperbolicBoundaryShadow aperture, HyperbolicVisualMetric comparison, flow-window exactness row, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6104.1, Definition 6104.2, and Definition 6104.3 seed the finite geodesic-flow ledger route through horosphere stabilization, boundary-shadow readback, visual-metric comparison, and terminal Real seals.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicGeodesicFlowLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host hyperbolic manifolds, quotient geodesic equality, invariant measures, differentiable flow theorems, global visual boundaries, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, geodesic-flow exposure, horosphere-ledger stabilization, boundary-shadow readback, visual-metric comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicGeodesicFlowLedgerUp.

Chapter 6105

A Concrete Naming Certificate for $\mathsf{FiniteMetricCover}^{\uparrow}$

Following Bishop and Bridges on constructive compact metric spaces and finite covers, the $\mathsf{FiniteMetricCover}^{\uparrow}$ packet records a finite family of metric balls together with the displayed radius and coverage witnesses needed by compactness consumers. It sits between the metric carrier of chapter 57, the compact source rows of chapter 129, the finite-net coverage surface of chapter 127, the closed-ball boundary rows of chapter 1451, and the real-radius seal of chapter 38. The packet names finite cover data only; it does not assert open-cover compactness, choice of subcovers, quotient point equality, host compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Definition 6105.1 (FiniteMetricCover carrier)**.** *A* $\mathsf{FiniteMetricCover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, K, T, B, R, G, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row carrying the point carrier and distance classifier.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose compactness consumers read the finite cover through this packet.*
3. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row that supplies center admission and tolerance scheduling.*
4. *B is the finite* $\mathsf{ClosedMetricBall}^{\uparrow}$ *row family, with one displayed center and positive radius ledger for each cover cell.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *or dyadic-real radius seal row attached to the ball radii and requested tolerance.*
6. *G is the finite cover membership bundle, naming the displayed cells and their coverage witnesses as a closed* ProbeBundle*.*
7. *H records componentwise* hsame *transport for M, K, T, B, R, G and the structural rows.*
8. *C records displayed* Cont *replay from the metric source through the finite cover bundle to downstream compactness consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteMetricCover}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{TotallyBounded}^{\uparrow}$*,* $\mathsf{ClosedMetricBall}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* ProbeBundle*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteMetricCover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected subcovers outside G, quotient point equality, host compactness, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FiniteMetricCoverUp.FiniteMetricCoverTasteGate single carrier alignment`

Theorem 6105.2 (FiniteMetricCover NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMetricCover}^{\uparrow}$ *carrier*

$$F = (M, K, T, B, R, G, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteMetricCover}^{\uparrow}}$ *surface consists exactly of metric source admission, compact-metric source exposure, total-bounded center scheduling, closed metric ball boundary rows, positive radius sealing, finite membership-bundle coverage, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6105.1. The accepted packet exposes only the metric source, compact source, total-bounded finite-net schedule, closed ball family, radius seal, finite membership bundle, and structural BEDC rows.

The ball family B is read together with the radius row R. Thus each cover cell has a displayed center, a displayed radius, and a closed metric boundary row before it can be consumed by compactness-facing routes.

The membership bundle G supplies the finite coverage witnesses. The rows H, C, P, N transport, replay, source, and name that same displayed route, so no arbitrary open cover, detached selected subcover, quotient equality, host compactness, or ambient completeness coordinate is introduced. □

Theorem 6105.3 (FiniteMetricCover refinement stability)**.** *Let*

$$F = (M, K, T, B, R, G, H, C, P, N)$$

and

$$F' = (M', K', T', B', R', G', H', C', P', N')$$

be accepted $\mathsf{FiniteMetricCover}^{\uparrow}$ *carriers. If componentwise* hsame *transport identifies* M, K, T, B, R, G *with* M', K', T', B', R', G' *and the second cover bundle refines the first by displayed membership witnesses, then every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{ClosedMetricBall}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of* F' *factors through the same finite cover route as a consumer of* F.

In particular, refinement may split cells and decrease displayed radii, but it cannot add a host open-cover object, choice-selected subcover, quotient point equality, ambient compactness theorem, or detached real-completeness row.

Proof. Start with the two accepted carriers and read their common transported rows. The hsame hypotheses identify the metric source, compact source, total-bounded schedule, closed-ball family, radius seal, and finite membership bundle before any consumer reads a refinement.

The displayed refinement witnesses attach each cell of G' to a cell of G. A consumer of the refined packet therefore reaches metric and compact data through M, K, T, reads ball boundaries through B' and R', and then returns to the original finite cover route through the displayed membership map into G.

Finally use the structural rows H, C, P, N and H', C', P', N' to replay and name the same route. Since both carriers contain only finite bundle data and structural BEDC rows, the refinement cannot export any excluded open-cover, choice, quotient, host-compactness, or real-completeness coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMetricCover}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteMetricCover}^{\uparrow}$ packet is generated from a MetricUp source row, CompactMetricUp source exposure, TotallyBoundedUp finite-net scheduling, ClosedMetricBallUp cell rows, RealUp radius seals, finite ProbeBundle coverage witnesses, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteMetricCover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6105.1, Definition 6105.2, and Definition 6105.3 seed the finite metric cover route used by compact metric and closed-ball consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteMetricCover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, choice-selected subcovers, quotient point equality, host compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked metric source admission, compact source exposure, finite-net scheduling, closed-ball coverage, radius sealing, refinement stability, transport, replay, provenance, and local naming for BEDC.Derived.FiniteMetricCoverUp.

Chapter 6106

A Concrete Naming Certificate for $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$

Following Hahn and Mazurkiewicz, the $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$ packet records the finite boundary certificate through which a compact connected locally connected metric continuum may be read before a Peano-continuum or Peano-curve consumer treats it as curve-facing data. It sits between the Peano continuum surface of chapter 2160, the finite curve-parametrization route of chapter 2078, the compact metric source of chapter 129, the continuous-map rows of chapter 130, and the terminal real-name seal of chapter 38. The packet names the boundary ledger needed for the classical Hahn–Mazurkiewicz handoff without importing an ambient topological quotient, host compactness theorem, selected parametrization, countable-choice schedule, or unrestricted real equality.

Definition 6106.1 (Hahn-Mazurkiewicz boundary carrier)**.** *A* $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$ *carrier is a finite BEDC packet*

$$H = (K, M, C, P, B, E, T, R, N)$$

with the following visible rows:

1. *K is a compact metric source row read through* $\mathsf{CompactMetric}^{\uparrow}$*;*
2. *M is a Peano-continuum local connectedness and connectedness ledger;*
3. *C is a finite* $\mathsf{ContinuousMap}^{\uparrow}$ *curve-read row from a compact interval-facing source into K;*
4. *P is the Peano-curve parametrization boundary row that records surjective coverage only through displayed windows;*
5. *B is the boundary classifier row comparing compact-metric neighborhoods by finite* BHist *support and componentwise* hsame *transport;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for endpoint and radius data;*
7. *T is the displayed* Cont *replay ledger for boundary refinements;*
8. *R is the* Pkg *provenance row; and*
9. *N is the local* $\mathsf{NameCert}$ *row naming the packet.*

The carrier contains no host quotient space, selected Peano parametrization, global compactness theorem, countable-choice selector, quotient real equality, or ambient completeness coordinate.

Lean4 ✓ BEDC.Derived.HahnMazurkiewiczBoundaryUp.HahnMazurkiewiczBoundaryTasteGate single carrier alignment

Theorem 6106.2 (Hahn-Mazurkiewicz boundary NameCert obligations)**.** *For every carrier $H = (K, M, C, P, B, E, T, R, N)$ of Definition 6106.1, the source, classifier, stability, replay, provenance, and local naming obligations for* $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$ *are discharged on the displayed finite rows. In particular, compact-metric reads remain on K, Peano-continuum reads remain on M, continuous-map and Peano-curve reads remain on C and P, boundary comparisons remain on B and E, and transport/replay/provenance/naming remain on T, R, N.*

Proof. Project the carrier rows of Definition 6106.1. The compact metric source, Peano-continuum ledger, continuous-map row, Peano-curve boundary, boundary classifier, real seal, replay ledger, provenance row, and local NameCert row are all displayed coordinates of the same finite packet.

The classifier obligation is read through B with BHist support and componentwise hsame transport, while endpoint and radius comparisons pass through E before any Real-facing consumer sees them. Replay is confined to T, and provenance is confined to R.

Thus every NameCert obligation is witnessed by the listed rows. Since the carrier has no coordinate for a host quotient space, selected parametrization, global compactness theorem, countable-choice selector, quotient real equality, or ambient completeness, none of those claims can be exported by the certificate. □

Theorem 6106.3 (Hahn-Mazurkiewicz boundary Peano-continuum handoff)**.** *Every* $\mathsf{PeanoContinuum}^{\uparrow}$ *or* $\mathsf{PeanoCurve}^{\uparrow}$ *consumer that reads a* $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$ *packet must factor through the compact metric source of chapter 129, the continuous-map rows of chapter 130, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal of chapter 38 before it consumes the Peano-continuum and Peano-curve boundary rows. The handoff exports a finite boundary certificate, not a host Hahn–Mazurkiewicz theorem, not a selected surjection, and not an unrestricted quotient equality of points.*

Proof. Start with the carrier decomposition. Any consumer receives the compact metric row K, continuous-map row C, Peano-continuum ledger M, Peano-curve boundary P, boundary classifier B, and Real seal E as separate visible rows.

The Peano-continuum side can inspect local connectedness and connectedness only through M, while the Peano-curve side can inspect coverage only through P after C has supplied the finite continuous-map read. Endpoint and radius data pass through E before the boundary classifier B is consumed.

Therefore the handoff factors through the compact metric, continuous-map, and Real sibling chapters before either Peano consumer reads the boundary packet. The missing host quotient, selected surjection, and unrestricted equality coordinates remain outside the exported interface. □

Closure status for $\mathsf{NameCert}_{\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}$ packet is generated from a compact metric source, a Peano-continuum ledger, a continuous-map row, a Peano-curve coverage boundary, a boundary classifier, a Real seal, displayed replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}}, \mathsf{scopedClosed})$

Scope closed: Definition 6106.1, Definition 6106.2, and Definition 6106.3 bind the finite compact-metric, Peano-continuum, Peano-curve, continuous-map, boundary-classifier, Real-seal, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HahnMazurkiewiczBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a host Hahn–Mazurkiewicz theorem, selected Peano surjection, host quotient topology, global compactness theorem, countable-choice selector, quotient real equality, ambient Real completeness, or a standard bridge.

Upgrade path: Formal verification can encode carrier admission, boundary classifier transport, continuous-map replay, Peano-continuum handoff, Peano-curve handoff, Real-seal factorization, and NameCert obligations for the displayed packet.

Chapter 6107

A Concrete Naming Certificate for $\mathsf{ConstantRegSeqRat}^{\uparrow}$

Following Bishop and Bridges on constant Cauchy names, the $\mathsf{ConstantRegSeqRat}^{\uparrow}$ packet records the stationary regular-Cauchy construction that sends one dyadic rational point to the regular rational name constantly equal to that point. It sits between the finite stream carrier of chapter 126, the regular rational readback of chapter 301, the dyadic point rows of chapter 291, and the terminal real-name seal of chapter 38. The packet supplies a visible source object for constant embedding routes without quotient stream equality, selected limits, host real equality, or ambient completion.

Definition 6107.1 (ConstantRegSeqRat carrier)**.** *A* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (D, S, R, H, K, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *point row, naming the dyadic rational endpoint whose constant regular name is being formed.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *constant schedule row, recording that every inspected finite window reads the same point row D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, carrying the stationary Cauchy proof rows and the dyadic tolerance comparisons needed by the regular rational readback.*
4. *H records pointwise* hsame *transport for the dyadic point, constant schedule, regularity row, provenance row, and local naming row.*
5. *K records displayed* Cont *replay from the point row through the constant schedule into the regularity row.*
6. *P is the* Pkg *provenance over* $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, R, H, K, P, N. It has no coordinate for quotient stream equality, selected representatives, selected limits, host real equality, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden nonstationary tail.*

Lean4 ✓ `BEDC.Derived.ConstantRegSeqRatUp.ConstantRegSeqRatTasteGate single carrier alignment`

Theorem 6107.2 (ConstantRegSeqRat NameCert obligations)**.** *For every accepted* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *carrier* $C = (D, S, R, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}$ *surface consists exactly of five obligation rows: dyadic point exposure through* D*, constant schedule exposure through* S*, stationary regularity exposure through* R*, transport and replay through* H, K*, and provenance with local naming through* P, N*.*

Proof. Project the carrier of Definition 6107.1. The point row D, schedule row S, regularity row R, transport row H, replay row K, provenance row P, and local name row N are the only displayed coordinates.

The source row is dyadic. A consumer must first read D as the point supplied by chapter 291, not as a host rational endpoint outside the packet.

The schedule row is constant. The $\mathsf{StreamName}^{\uparrow}$ row S repeats the same displayed point on every finite window, and the $\mathsf{RegSeqRat}^{\uparrow}$ row R records the stationary regularity proof rows for that schedule.

Finally H, K, P, N transport, replay, source, and name the same finite route. Since no extra coordinate stores quotient stream equality, selected limits, host equality, or ambient completion data, the five obligation groups exhaust the local NameCert surface. □

Theorem 6107.3 (ConstantRegSeqRat regularity stability)**.** *Let* $C = (D, S, R, H, K, P, N)$ *and* $C' = (D', S', R', H', K', P', N')$ *be accepted* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *carriers. If the point rows* D *and* D' *compare by the displayed* $\mathsf{DyadicRatCore}^{\uparrow}$ *classifier and the constant schedules are transported pointwise through* H, H'*, then the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row of* C *transports to the stationary regularity row of* C'*. Thus a constant stream preserves regular Cauchy regularity under pointwise classifier transport.*

Proof. Read the point comparison first. The only source data for either carrier is the displayed dyadic point row, so a pointwise classifier comparison relates every inspected constant-window entry.

Use the schedule rows next. Because S and S' are constant schedules, transporting the point comparison along H and H' transports every finite StreamName window without introducing a new tail or selector.

The regularity rows are stationary. The same dyadic tolerance comparison that witnesses the pointwise transport also witnesses each regular Cauchy window, since both schedules repeat the transported point. The replay rows K, K' preserve this route into the $\mathsf{RegSeqRat}^{\uparrow}$ readback.

Therefore the accepted regularity row transports from R to R'. The argument uses only displayed point, schedule, regularity, transport, replay, provenance, and local naming rows, so it adds no quotient stream equality or host real equality. □

Theorem 6107.4 (ConstantRegSeqRat Real-seal nonescape)**.** *For every accepted* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *carrier* $C = (D, S, R, H, K, P, N)$*, any* $\mathsf{Real}^{\uparrow}$*-facing constant-embedding read must consume the displayed route*

$$D, S, R, H, K, P, N$$

before the terminal $\mathsf{Real}^{\uparrow}$ *seal is available. Thus the read passes through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *point row, the* $\mathsf{StreamName}^{\uparrow}$ *constant schedule, the stationary* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row, pointwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}$ *support.*

The terminal seal exposes no quotient stream equality, selected limit, host real equality, arbitrary Cauchy-name existence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Project the carrier of Definition 6107.1. The only public source rows are the displayed dyadic point row D, constant schedule row S, stationary regularity row R, transport row H, replay row K, provenance row P, and local naming row N.

Apply Definition 6107.2. It identifies these rows as the full local NameCert obligation surface, so a Real-facing constant embedding cannot read a hidden representative or a second completion coordinate.

Use Definition 6107.3 for the stationary regularity step. The constant schedule transports each finite window through the same dyadic point row and the same regularity ledger, preserving the ordered route into the terminal seal.

Therefore every accepted Real-facing read consumes D, S, R, H, K, P, N before the $\mathsf{Real}^{\uparrow}$ seal is available. Since no accepted coordinate records quotient stream equality, selected limits, host equality, arbitrary Cauchy-name existence, or ambient completeness, none of those objects is exposed by the seal. □

Falsifiable prediction: Every accepted $\mathsf{ConstantRegSeqRat}^{\uparrow}$ read must expose a DyadicRatCore point row, a StreamName constant schedule, a stationary RegSeqRat regularity row, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Real-facing constant embedding may consume it.

Independence witness: dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{ConstantRegSeqRat}^{\uparrow}$ because this packet jointly carries a dyadic point, a constant StreamName schedule, stationary RegSeqRat regularity, transport, replay, provenance, and local naming for the same constant-name construction.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstantRegSeqRat}^{\uparrow}$ packet is generated from a DyadicRatCore point row, a constant StreamName schedule, stationary RegSeqRat regularity rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6107.1, Definition 6107.2, and Definition 6107.3 seed the finite stationary regular-Cauchy construction over DyadicRatCore, StreamName, RegSeqRat, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstantRegSeqRat}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, selected limits, quotient stream equality, host real equality, arbitrary Cauchy-name existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constant schedule exposure, stationary RegSeqRat regularity, pointwise transport stability, replay, provenance, local naming, and non-escape for BEDC.Derived.ConstantRegSeqRatUp.

Chapter 6108

A Concrete Naming Certificate for $\mathsf{CompactnessCompletionRoute}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley compact-metric practice, the CompactnessCompletion packet records the finite route by which compact-metric finite-net data are read into completion-facing Cauchy windows before any real seal or completion boundary may be consumed. It sits over the compact metric source of chapter 129, the complete-metric boundary of chapter 128, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the Cauchy completion surface of chapter 360. The chapter names a finite bridge from compact finite covers to completion-facing Cauchy readback; it does not assert open-cover compactness, choose a convergent subnet, quotient Cauchy streams, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6108.1 (Compactness completion route carrier)**.** *A* $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, T, C, W, R, E, H, Q, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the finite metric observations and compact-source identifiers read by the packet.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net coverage row, or equivalently the finite net schedule attached to the displayed compact metric source.*
3. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *or Cauchy-completion boundary row that records where completion-facing Cauchy requests may land after the finite-net route has been read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose displayed observations are generated from the finite-net coverage schedule.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the dyadic tolerance ledger, before any real seal is public.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *or completion-seal row reached only after M, T, C, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the compact source, finite-net coverage, completion boundary, stream windows, regular readback, real seal, continuation, provenance, and local naming rows.*

8. *Q records displayed* Cont *replay from compact-source finite nets through completion-facing Cauchy windows and back to the same regular readback rows.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactnessCompletionRoute}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactnessCompletionRoute}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, a selected subnet, quotient stream equality, countable choice, host metric completion, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessCompletionRouteUp.CompactnessCompletionRouteTasteGate single carrier alignment`

Theorem 6108.2 (Compactness completion route NameCert obligations)**.** *For every accepted* $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ *carrier* $K = (M, T, C, W, R, E, H, Q, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactnessCompletionRou}}$ *surface consists exactly of carrier admission for the ten displayed rows; finite-net coverage admission from M to T; completion-boundary admission through C; stream-window generation through W; regular rational readback through R; terminal real or completion sealing through E; componentwise transport through H; displayed replay through Q; provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.CompactnessCompletionRouteUp.CompactnessCompletionRouteCarrier namecert obligations`

Proof. Project the carrier of Definition 6108.1. The analytic rows are precisely the compact source M, finite-net coverage T, completion boundary C, stream windows W, regular readback R, and terminal seal E.

The NameCert obligations follow the order of those rows. A consumer first reads the compact metric source, then the finite-net coverage row, then the completion-facing boundary, and only then the stream and regular-rational readback rows.

The structural rows H, Q, P, N transport, replay, source, and name the same route. Since no excluded object appears in the carrier, the obligation surface cannot export open-cover compactness, selected subnets, quotient streams, countable choice, host completion, or ambient real completeness. □

Theorem 6108.3 (Compactness completion nonescape)**.** *Every completion-facing consumer of an accepted* $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ *carrier must read the visible route*

$$M, T, C, W, R, E, H, Q, P, N.$$

In particular, a consumer moving from $\mathsf{CompactMetric}^{\uparrow}$ *finite-net data toward* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *readback cannot bypass the displayed stream windows, cannot substitute a detached Cauchy representative, and cannot import open-cover compactness or ambient real completeness.*

Lean4 ✓ `BEDC.Derived.CompactnessCompletionRouteUp.CompactnessCompletionRouteNonescape`

Proof. Start from the compact metric source M. The only compactness-facing data admitted by the packet are the finite-net coverage rows in T, so any completion-facing read begins with the displayed finite schedule rather than with an open-cover theorem.

Next read C, W, R, E in order. The completion boundary C identifies the permitted Cauchy-facing landing surface, the stream windows W display the finite observations, the regular readback R records the Cauchy name data, and the seal E remains terminal.

Finally inspect H, Q, P, N. These rows preserve the same route under transport and replay. They provide no coordinate for a detached representative, quotient stream, selected subnet, host completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness, so the consumer cannot leave the displayed carrier. □

Theorem 6108.4 (Compact finite-net completion handoff)**.** *Let $K = (M, T, C, W, R, E, H, Q, P, N)$ be an accepted* $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ *carrier. The finite-net coverage row T and the stream-window row W determine the only handoff by which compact-metric data enter completion-facing Cauchy readback. Thus every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *read from the packet is tied to the same compact source M, completion boundary C, transport row H, replay row Q, provenance row P, and local name N.*

Lean4 ✓ `BEDC.Derived.CompactnessCompletionRouteUp.CompactnessCompletionRouteFiniteNetHandoff`

Proof. Read T as the finite-net coverage admitted over the compact metric source M. It records the displayed coverage schedule and does not itself choose a limit point or completed object.

The stream windows W are generated from that schedule before the regular readback R can be consumed. The handoff to C and E therefore carries exactly the finite windows and regular Cauchy readback named by the carrier.

Transport, replay, provenance, and local naming preserve the same rows. Any readback that omits T or W would use a coordinate absent from Definition 6108.1, so it is not a read of the accepted packet. □

Falsifiable prediction: An accepted $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ consumer that reaches $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ completion-facing data without displaying the compact finite-net row T and the $\mathsf{StreamName}^{\uparrow}$ window row W refutes the carrier surface.

Independence witness: compactmetric, completemetric, streamname, regseqrat, dyadicrat-core, real, real-cauchy-completion: no listed sibling supplies the whole ten-row packet $K = (M, T, C, W, R, E, H, Q, P, N)$, because each sibling contributes only a compact source, completion boundary, finite window, regular readback, dyadic ledger, real seal, or completion-facing boundary while this chapter binds the finite-net route through transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactnessCompletionRoute}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactnessCompletionRoute}^{\uparrow}$ packet is generated from a compact metric source row, finite-net coverage row, completion boundary row, stream-window row, RegSeqRat readback row, Real or completion seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactnessCompletionRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6108.1, Definition 6108.2, Definition 6108.3, and Definition 6108.4 seed the finite compactness-to-completion readback route over compact-source, finite-net, completion-boundary, stream-window, regular-readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactnessCompletionRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subnet extraction, quotient Cauchy streams, countable choice, host metric completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{CompactnessCompletionRoute}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage admission, completion-boundary admission, stream-window generation, regular readback, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactnessCompletionRouteUp.

Chapter 6109

A Concrete Naming Certificate for $\mathsf{NewtonPolygon}^{\uparrow}$

Following Newton and Dumas, the $\mathsf{NewtonPolygon}^{\uparrow}$ packet records the finite lower-hull surface attached to a displayed polynomial with valued coefficients. It sits after the coefficient carrier of chapter 50, the dyadic valuation rows of chapter 2106, the non-archimedean source of chapter 100, the comparison surface of chapter 54, the convex carrier of chapter 208, and the finite face ledger of chapter 209. The packet names a finite valuation/polytope certificate rather than a host convex-hull oracle, algebraic-closure theorem, Puiseux expansion chooser, quotient polynomial equality, or selected root branch.

Definition 6109.1 (NewtonPolygon carrier)**.** *A* $\mathsf{NewtonPolygon}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, V, T, Q, F, S, H, C, P, N)$$

with the following displayed rows:

1. *A is the polynomial coefficient row, read through the normalized coefficient surface of* $\mathsf{Poly}^{\uparrow}$*.*
2. *V is the dyadic valuation row for every displayed nonzero coefficient, read through* $\mathsf{DyadicValuation}^{\uparrow}$ *and the surrounding* $\mathsf{Padic}^{\uparrow}$ *scale source.*
3. *T is the* $\mathsf{TotalOrder}^{\uparrow}$ *comparison ledger ordering exponent and valuation pairs without importing trichotomy outside the displayed row.*
4. *Q is the lower convex hull row, a finite* $\mathsf{ConvexSet}^{\uparrow}$ *and* $\mathsf{Polytope}^{\uparrow}$ *face packet over the point ledger (i, V_i).*
5. *F is the finite lower-face ledger selecting exactly the faces visible from below.*
6. *S is the slope ledger, recording for each lower face the displayed ordered valuation difference and exponent-length readback.*
7. *H records componentwise* hsame *transport for coefficient, valuation, order, lower-hull, face, slope, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from coefficient inspection through valuation exposure, order comparison, lower-hull admission, face selection, slope reading, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{NewtonPolygon}^{\uparrow}$*,* $\mathsf{Poly}^{\uparrow}$*,* $\mathsf{DyadicValuation}^{\uparrow}$*,* $\mathsf{Padic}^{\uparrow}$*,* $\mathsf{TotalOrder}^{\uparrow}$*,* $\mathsf{ConvexSet}^{\uparrow}$*,* $\mathsf{Polytope}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{NewtonPolygon}^{\uparrow}}$ *row.*

The classifier reads two accepted carriers only through $A, V, T, Q, F, S, H, C, P, N$. It has no coordinate for host convex-hull computation, quotient polynomial equality, selected roots, algebraic closure, Puiseux branches, ambient real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.NewtonPolygonUp.NewtonPolygonTasteGate single carrier alignment`

Theorem 6109.2 (NewtonPolygon NameCert obligations)**.** *For every accepted* $\mathsf{NewtonPolygon}^{\uparrow}$ *carrier $G = (A, V, T, Q, F, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NewtonPolygon}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the polynomial coefficient row, dyadic valuation rows, total-order comparisons, lower convex hull, lower-face ledger, slope ledger, transport, replay, provenance, and local naming rows;*
2. *valuation readback, requiring every displayed coefficient used by the polygon to expose its dyadic valuation through V before it contributes to Q or S;*
3. *lower-hull stability, requiring componentwise* hsame *transport of coefficient valuations and order comparisons to transport the lower convex hull row Q and the finite lower-face ledger F;*
4. *face-slope classifier transport, requiring each accepted lower face to read its slope from the same ordered endpoint valuations and exponent-length row before a consumer can classify a side of the polygon;*
5. *ledger nonescape, excluding host powersets of points, quotient polynomial equality, host convex-hull or polytope algorithms, selected roots, algebraic closure, Puiseux branch selection, ambient real completeness, Lean verification, and bridge checking.*

No Newton-polygon obligation is stored outside $A, V, T, Q, F, S, H, C, P, N$.

Proof. Project the accepted carrier. The polynomial row A supplies the finite coefficient list, while V supplies valuation readback only for coefficients displayed by that list and the surrounding p-adic or dyadic scale rows.

Read the comparison and hull rows next. The total-order row T compares the finite exponent-valuation pairs, and the convex/polytope rows Q, F record the lower hull and its lower faces as finite BEDC ledgers rather than as an ambient convex-hull computation.

For each lower face, the slope ledger S reads the two endpoint valuations and the finite exponent-length row before any classifier result is exported. Componentwise hsame transport in H and displayed replay in C preserve that same route across accepted carriers.

Finally inspect P and N. They record provenance and local naming for precisely the displayed rows A, V, T, Q, F, S, H, C, and the carrier contains no coordinate for host powersets, quotient polynomial equality, selected roots, algebraic closure, Puiseux branch choice, ambient real completeness, Lean verification, or bridge checking. The listed rows therefore exhaust the $\mathsf{NewtonPolygon}^{\uparrow}$ NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{NewtonPolygon}^{\uparrow}}$.

Constructive story: A $\mathsf{NewtonPolygon}^{\uparrow}$ packet is generated from finite polynomial coefficient rows, dyadic valuation rows, total-order comparisons, lower convex hull and face rows, slope ledger rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{NewtonPolygon}^{\uparrow}}$, seed)

Scope closed: Definition 6109.1 and Definition 6109.2 seed the finite valuation/polytope certificate surface over polynomial, dyadic valuation, p-adic, total-order, convex-set, and polytope rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{NewtonPolygon}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim algebraic closure, root existence, Puiseux expansions, host convex-hull computation, quotient polynomial equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, valuation readback, lower-hull stability, face-slope classifier transport, provenance, and ledger exhaustion for BEDC.Derived.NewtonPolygonUp.

Chapter 6110

A Concrete Naming Certificate for $\mathsf{DiniDerivativeWindow}^{\uparrow}$

Following Saks and Bishop–Bridges constructive analysis, the $\mathsf{DiniDerivativeWindow}^{\uparrow}$ packet records the finite window surface by which a real-valued interval observation exposes upper and lower Dini rows before any differentiability-boundary consumer may read them. It sits below the broader Denjoy–Young–Saks boundary of chapter 5829, and it uses the $\mathsf{Real}^{\uparrow}$ seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, the $\mathsf{StreamName}^{\uparrow}$ observation schedule of chapter 126, and the $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger of chapter 291. The chapter names only a finite Dini-window carrier; it does not claim measure-theoretic alternatives, almost-everywhere differentiability, selected derivatives, or ambient real completeness.

Definition 6110.1 (DiniDerivativeWindow carrier)**.** *A* $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, F, U, L, O, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval-cell row naming the inspected source cell and its endpoint readback surface.*
2. *F is the function-observation window row, storing the finite source-target observations visible at the inspected cell.*
3. *U is the upper Dini row, containing the displayed upper right, upper left, and two-sided upper difference-quotient bounds for the window.*
4. *L is the lower Dini row, containing the corresponding lower right, lower left, and two-sided lower bounds.*
5. *O is the oscillation cell, recording the finite comparison between the upper and lower rows at the displayed tolerance scale.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule that orders the finite observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the scheduled rational observations.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare the windowed difference quotients.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal visible only after I, F, U, L, O, S, R, D have been replayed.*
10. *H records componentwise* hsame *transport for the interval, observation, Dini-row, oscillation, schedule, readback, tolerance, seal, replay, provenance, and local naming rows.*

11. *C records displayed* Cont *replay from a Dini-window request through finite observations, regular readback, dyadic tolerance, and the Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{DiniDerivativeWindow}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{DiniDerivativeWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for a measure structure, an almost-everywhere classifier, a selected derivative, quotient real equality, trichotomy, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.DiniDerivativeWindowUp.DiniDerivativeWindowTasteGate single carrier alignment`

Theorem 6110.2 (DiniDerivativeWindow NameCert obligations)**.** *For every accepted* $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *carrier* $W = (I, F, U, L, O, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DiniDerivativeWindow}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the interval cell, function-observation window, upper Dini row, lower Dini row, oscillation cell, StreamName schedule, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, replay, provenance, and local naming rows;*
2. *Dini-row separation, requiring the upper row U and lower row L to remain distinct visible rows before an oscillation cell can compare them;*
3. *schedule-readback exactness, requiring every inspected observation to pass through S and R before the tolerance ledger D is consulted;*
4. *Real-seal handoff, requiring E to remain downstream of the finite Dini window and never upstream of it;*
5. *ledger refusal, excluding measure-theoretic alternatives, almost-everywhere claims, selected derivatives, quotient real equality, trichotomy, countable choice, and ambient completeness.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6110.1. The packet displays exactly the interval, observation, upper Dini, lower Dini, oscillation, schedule, readback, tolerance, Real-seal, transport, replay, provenance, and local naming rows.

The Dini-row separation is forced by the row list. The oscillation row O reads both U and L, but neither row is identified with the other, and neither can be replaced by the schedule, readback, tolerance, or seal rows.

Replay by C gives the only consumer route. A request passes through I, F, U, L, O, S, R, D before it can reach E, while H, P, N transport, source, and name the same route. Hence no off-packet measure principle, selected derivative, quotient equality, or completeness coordinate can supply an additional obligation. □

Theorem 6110.3 (DiniDerivativeWindow Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-,* $\mathsf{RegSeqRat}^{\uparrow}$*-, or* $\mathsf{StreamName}^{\uparrow}$*-facing read of an accepted* $\mathsf{DiniDerivativeWindow}^{\uparrow}$ *packet factors through the finite route*

$$I, F, U, L, O, S, R, D \longmapsto E \longmapsto H, C, P, N.$$

Thus a consumer can inspect interval cells, function-observation windows, upper and lower Dini rows, oscillation cells, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance rows, a terminal Real seal, transport, replay, provenance, and local naming. It cannot read a measure structure, an almost-everywhere theorem, a selected derivative, quotient real equality, trichotomy, countable choice, or ambient completeness from the packet.

Proof. Start with any read aimed at the Real seal, regular readback, or stream schedule. Carrier admission exposes those rows only as components of the packet in Definition 6110.1.

Use the NameCert obligations of Definition 6110.2. The request must pass through the displayed interval cell, function-observation window, separated upper and lower Dini rows, oscillation cell, schedule, readback, and dyadic tolerance ledger before the Real seal can be read.

The structural rows preserve the same dependency. Transport by H and replay by C cannot add a hidden measure space, selected derivative, quotient equality, trichotomy, choice principle, or completeness theorem, while P and N source and name only the displayed finite route. □

Falsifiable prediction: Every accepted $\mathsf{DiniDerivativeWindow}^{\uparrow}$ Real-facing or readback-facing read factors through finite interval cells, function-observation windows, upper and lower Dini rows, an oscillation cell, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance rows, a terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: denjoy-young-saks, real, regseqrat, streamname, dyadicratcore: no listed sibling supplies the whole DiniDerivativeWindow packet, because each sibling contributes only a broader Dini boundary, terminal real seal, rational readback, observation schedule, or tolerance ledger while this chapter binds the finite Dini window in one carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{DiniDerivativeWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{DiniDerivativeWindow}^{\uparrow}$ packet is generated from interval cells, function-observation windows, upper and lower Dini rows, oscillation cells, StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerance rows, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DiniDerivativeWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6110.1, Definition 6110.2, and Definition 6110.3 bind the finite Dini-window carrier and its Real-seal nonescape route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DiniDerivativeWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure-theoretic alternatives, almost-everywhere differentiability, selected derivatives, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, Dini-row separation, oscillation-cell exactness, stream-readback-dyadic exactness, Real seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.DiniDerivativeWindowUp.

Chapter 6111

A Concrete Naming Certificate for MetaCICDischargeInversionBoundary$^{\uparrow}$

The MetaCICDischargeInversionBoundary$^{\uparrow}$ packet realizes the four subject-reduction discharge rows of section 9386.2 as a finite inversion boundary. It sits beside the discharge socket of chapter 719, the subject-reduction frontier of chapter 730, the redex frontier of chapter 1008, the proof-search frontier of chapter 4284, and the critical-path packet of chapter 765. The carrier names the finite boundary at which beta-substitution, application-argument stability, lambda-domain preservation, and Pi-domain preservation can be inspected as rows; it does not prove any of those rows and does not export a closed subject-reduction theorem.

Falsifiable prediction: An accepted MetaCICDischargeInversionBoundary$^{\uparrow}$ consumer that reads a subject-reduction discharge route without displaying the beta-substitution, application-argument, lambda-domain, Pi-domain, obstruction, socket, transport, replay, provenance, and local naming rows refutes the carrier.

Independence witness: subject-reduction-discharge-socket, metacic-subject-reduction-frontier, metacic-redex-frontier, metacic-proof-search-frontier, metaciccriticalpath: none is bijective to MetaCICDischargeInve because this packet jointly binds four discharge rows, an obstruction row, a socket boundary, inversion routing, TasteGate readiness, transport, replay, provenance, and local naming.

The Lean TasteGate for this packet will provide explicit Nontrivial MetaCICDischargeInversionBoundary$^{\uparrow}$ and FieldFaithful MetaCICDischargeInversionBoundary$^{\uparrow}$ instances. Those instances must keep the discharge rows B, A, L, P, the obstruction row O, the socket row S, the inversion route V, the TasteGate row T, and the structural rows H, C, Q, N as separate fields before any checked carrier-alignment theorem can serve as the public gate.

Definition 6111.1 (MetaCICDischargeInversionBoundary carrier)**.** *A* MetaCICDischargeInversionBoundary$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (B, A, L, P, O, S, V, T, H, C, Q, N)$$

with the following displayed rows:

1. *B is the beta-substitution preservation discharge row.*
2. *A is the application-argument type-stability discharge row.*
3. *L is the lambda-domain subject-reduction discharge row.*
4. *P is the Pi-domain subject-reduction discharge row.*

5. *O is the obstruction row recording that the four-row interface is not replaced by a closed subject-reduction theorem.*
6. *S is the* $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$ *row that names the finite socket through which the four rows are consumed.*
7. *V is the inversion route row, recording which discharge row a subject-reduction branch asks to inspect.*
8. *T is the TasteGate readiness row requiring Nontrivial and FieldFaithful separation of the displayed fields.*
9. *H records componentwise* hsame *transport for the discharge, obstruction, socket, inversion, TasteGate, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a branch inspection request through the socket, inversion route, selected discharge row, obstruction boundary, and TasteGate row.*
11. *Q is the* Pkg *provenance over* $\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}$, $\mathsf{SubjectReductionDischargeSocket}^{\uparrow}$, *the four MetaCIC discharge rows,* BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}}$ *row.*

The carrier contains no closed subject-reduction proof, no replacement fifth discharge row, no host typing oracle, no quotient syntax equality, and no hidden axiom channel.

Theorem 6111.2 (MetaCICDischargeInversionBoundary NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}$ *carrier $M = (B, A, L, P, O, S, V, T, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the four discharge rows, obstruction row, socket row, inversion route, TasteGate row, transport, replay, provenance, and local naming rows;*
2. *four-row coverage, requiring the beta-substitution, application-argument, lambda-domain, and Pi-domain rows to remain distinct visible inputs;*
3. *inversion exactness, requiring the route row V to select among B, A, L, P without replacing any selected row by O, S, T, H, C, Q, or N;*
4. *socket preservation, requiring every discharge read to pass through S before a branch-specific row is inspected;*
5. *TasteGate separation, requiring the Nontrivial and FieldFaithful obligations for the displayed carrier fields;*
6. *ledger refusal, excluding a closed subject-reduction theorem, host typing oracle, quotient syntax equality, hidden axiom channel, or fifth discharge row.*

No NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6111.1. The packet displays exactly four discharge rows, an obstruction row, a socket row, an inversion route, a TasteGate row, and the structural transport, replay, provenance, and naming rows.

Use the four-row ledger exactness of Definition 719.8. The socket already separates beta-substitution, application-argument, lambda-domain, and Pi-domain inputs, and it does not allow a fifth discharge row or an obstruction row to stand in for one of them.

The inversion route V refines only which of the four rows is being inspected. The route can be transported by H and replayed by C, while Q, N source and name the same finite packet, but none of those rows creates a closed subject-reduction proof or a host typing oracle.

TasteGate readiness records the remaining admission discipline. Nontriviality distinguishes packets with missing discharge fields from accepted packets, and field faithfulness preserves the displayed row order, so the local NameCert surface is exactly the finite surface listed above. □

Theorem 6111.3 (MetaCICDischargeInversionBoundary socket nonescape)**.** *Every subject-reduction-facing read of an accepted* $\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}$ *carrier factors through the socket-preserving route*

$$S, V, (B, A, L, P), O, T \longmapsto H, C, Q, N.$$

The read may inspect the four discharge rows, the obstruction row, the socket boundary, the inversion route, TasteGate readiness, transport, replay, provenance, and local naming. It cannot obtain a closed subject-reduction theorem, a replacement discharge row, a host typing oracle, quotient syntax equality, or a hidden axiom channel.

Proof. Start with a subject-reduction-facing read. The carrier exposes the socket row S as the public entry to the four discharge inputs, so the read cannot directly inspect B, A, L, or P without passing through S.

Apply Definition 6111.2. The inversion route V selects among the four displayed rows only after the socket boundary is visible, and the obstruction row O remains a boundary row rather than a discharge proof.

The TasteGate row T and structural rows H, C, Q, N preserve that same inventory. Since the carrier has no coordinate for a closed theorem, replacement row, host typing oracle, quotient syntax equality, or hidden axiom channel, none can escape through a public read. □

Closure status for $\mathsf{NameCert}_{\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}$ packet is generated from beta-substitution, application-argument, lambda-domain, Pi-domain, obstruction, discharge-socket, inversion-route, TasteGate, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6111.1, Definition 6111.2, and Definition 6111.3 bind the finite discharge-inversion boundary and its socket nonescape route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MetaCICDischargeInversionBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not prove any subject-reduction discharge row, close subject reduction, supply a host typing oracle, quotient syntax equality, hidden axiom channel, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, four-row coverage, inversion exactness, socket preservation, TasteGate nontriviality, field faithfulness, socket nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MetaCICDischargeInversionBoundaryUp.

Chapter 6112

A Concrete Naming Certificate for BishopDiagonalCauchySequence$^{\uparrow}$

Following Bishop and Bridges on diagonal arguments for regular Cauchy names, the BishopDiagonalCauchySequence packet records the constructive diagonal readback of a displayed family of regular rational Cauchy names as finite NameCert data. It sits after the diagonalization route of chapter 3782, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet gives Real and Completion consumers one visible diagonal Cauchy sequence route without quotient streams, selected limits, countable choice, trichotomy, or ambient Real$^{\uparrow}$ completeness.

Definition 6112.1 (Bishop diagonal Cauchy sequence carrier)**.** *A* BishopDiagonalCauchySequence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, I, S, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the family row, naming the finite inspected portion of a family of regular rational Cauchy names.*
2. *I is the diagonal-index row, recording the unary stage selector used to read the family along the Bishop diagonal.*
3. *S is the* StreamName$^{\uparrow}$ *window row, displaying the finite windows inspected from each selected source name.*
4. *T is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying the dyadic budgets and comparison ledgers used by the diagonal read.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the diagonal sequence obtained from F, I, S, T.*
6. *E is the* Real$^{\uparrow}$ *seal row consumed only after the regular rational diagonal readback is visible.*
7. *H records componentwise* hsame *transport for the family, index, stream, tolerance, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Real or Completion precision request through the diagonal selector into the regular rational readback.*
9. *P is the* Pkg *provenance over* BishopDiagonalCauchySequence$^{\uparrow}$, RegularCauchyDiagonalization$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchySequence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, I, S, T, R, E, H, C, P, N$. It has no coordinate for quotient stream equality, countable choice, a selected host limit, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopDiagonalCauchySequenceUp.BishopDiagonalCauchySequenceTasteGate`
`single carrier alignment`

Theorem 6112.2 (Bishop diagonal Cauchy sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopDiagonalCauchySequence}^{\uparrow}$ *carrier*

$$D = (F, I, S, T, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchySequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *family and diagonal-index exposure through F and I;*
2. *finite stream-window exposure through S;*
3. *dyadic tolerance and comparison exposure through T;*
4. *regular rational diagonal readback through R;*
5. *terminal Real-facing seal through E;*
6. *structural transport, replay, provenance, and local naming through H, C, P, N.*

No diagonal Cauchy obligation is stored outside these rows.

Proof. Project the carrier of Definition 6112.1. The displayed coordinates are exactly the family, diagonal-index, stream-window, dyadic-tolerance, regular-readback, Real-seal, transport, replay, provenance, and local naming rows.

The family and selector rows expose the diagonal source before any Cauchy read is available. The stream and tolerance rows then supply the finite inspected windows and dyadic budgets used by the diagonal readback.

The regular rational row R is downstream of those rows, so the Real-facing seal E can be consumed only after the $\mathsf{RegSeqRat}^{\uparrow}$ diagonal sequence is visible. The structural rows H, C, P, N transport, replay, source, and name the same finite route.

Since the carrier contains no quotient stream, choice functional, selected host limit, trichotomy row, host equality row, or ambient-completeness row, the listed obligations exhaust the local NameCert surface. □

Theorem 6112.3 (Bishop diagonal Cauchy sequence Real handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{BishopDiagonalCauchySequence}^{\uparrow}$ *carrier must read the diagonal route*

$$\mathsf{RegularCauchyDiagonalization}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before treating the diagonal sequence as a completed real-name boundary. The handoff exports only the displayed family row, diagonal-index row, finite stream windows, dyadic tolerance ledger, regular rational readback, Real seal, and structural rows.

Proof. Start from the obligation surface in Definition 6112.2. It orders the diagonal family and selector before stream windows, dyadic tolerances, regular readback, and Real seal data.

Apply the sibling diagonalization route Definition 3782.3. The diagonal read becomes Real-facing only through the same $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ seal rows.

The replay row C sends a Real or Completion precision request through exactly that route, while H, P, N preserve, source, and name it. Therefore the handoff cannot bypass the four-object L10 seal or export quotient streams, selected limits, countable choice, trichotomy, host equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{BishopDiagonalCauchySequence}^{\uparrow}$ read must expose a regular-Cauchy diagonalization row, StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any Real or Completion consumer receives it.

Independence witness: regular-cauchy-diagonalization, streamname, regseqrat, dyadicratcore, real: no listed sibling supplies the whole ten-row packet $D = (F, I, S, T, R, E, H, C, P, N)$, because each sibling contributes only diagonal selection, finite windows, tolerance rows, regular readback, or terminal Real sealing while this chapter binds the route with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchySequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopDiagonalCauchySequence}^{\uparrow}$ packet is generated from a displayed family of regular rational names, a diagonal-index row, StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchySequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6112.1, Definition 6112.2, and Definition 6112.3 seed the finite constructive diagonal Cauchy sequence route for Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopDiagonalCauchySequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, diagonal-index exposure, finite-window exposure, dyadic-tolerance exactness, RegSeqRat readback, Real-seal handoff, structural stability, and non-escape for BEDC.Derived.BishopDiagonalCauchySequenceUp.

Chapter 6113

A Concrete Naming Certificate for $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$

Following Heine–Cantor and Bishop constructive analysis, the $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$ packet records the finite stability boundary for the radius-minimum fold used in compact uniform-continuity arguments. It sits after the compact metric source of chapter 129, the ProbeBundle lower-bound fold of chapter 4118, the finite-radius stability route of chapter 5203, the metric comparison surface of chapter 57, the rational lower-bound rows of chapter 37, and the uniform-modulus consumer of chapter 369. The packet exposes classifier stability of the finite minimum as concrete NameCert data rather than hiding it inside a fold step.

Definition 6113.1 (Compact uniform radius-minimum stability carrier)**.** *A* $\mathsf{CompactUniformRadiusMinimumStabili}$ *carrier is a finite* BHist *packet*

$$M = (K, B, Q, S, G, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row whose centers carry local radius witnesses.*
2. *B is the finite* $\mathsf{ProbeBundle}$ *row listing the radius witnesses, fold order, and bundle membership classifier.*
3. *Q is the* $\mathsf{Rat}^{\uparrow}$ *positive lower-bound row produced by the finite minimum fold.*
4. *S is the classifier-stability row, recording that componentwise transport of the bundle rows preserves the selected minimum radius ledger.*
5. *G is the* $\mathsf{Met}^{\uparrow}$ *comparison row used by the compact-uniform triangle estimate.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, downstream of K, B, Q, S, G.*
7. *H records componentwise* hsame *transport for compact-source, ProbeBundle, rational-fold, stability, metric, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a tolerance request through the finite bundle, radius-minimum fold, stability check, metric comparison, and uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ProbeBundle}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, B, Q, S, G, U, H, C, P, N$. It has no coordinate for host open-cover compactness, semantic continuity, a hidden minimum selector, quotient metric equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ BEDC.Derived.CompactUniformRadiusMinimumStabilityUp.CompactUniformRadiusMinimumStabilityTasteGa
single carrier alignment

Theorem 6113.2 (Compact uniform radius-minimum stability obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$ *carrier*

$$M = (K, B, Q, S, G, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *compact finite-net admission through K;*
2. *finite ProbeBundle membership and fold-order exposure through B;*
3. *rational lower-bound exactness through Q;*
4. *classifier stability of the selected minimum ledger through S;*
5. *metric comparison and uniform-modulus handoff through G, U;*
6. *structural transport, replay, provenance, and local naming through H, C, P, N.*

No radius-minimum stability obligation is stored outside these rows.

Proof. Project the carrier of Definition 6113.1. The finite compact source, ProbeBundle row, rational fold, stability row, metric row, uniform-modulus row, and structural rows are exactly the displayed coordinates.

The compact source and bundle rows expose the finite list of radius witnesses. The rational fold row Q records the selected positive lower bound as a history over B, not as an external selector.

The stability row S is then read before any uniform-modulus consumer receives Q. It checks that componentwise transport of the compact-source and bundle classifiers preserves the lower-bound ledger used by the fold.

Finally G, U, H, C, P, N route the stable radius minimum into the uniform-modulus handoff. Since the carrier contains no hidden open-cover compactness, semantic continuity, quotient metric equality, or ambient-completeness coordinate, the listed obligations exhaust the local NameCert surface. □

Theorem 6113.3 (Compact uniform radius-minimum uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$ *carrier receives the finite precision ledger only after reading*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ProbeBundle}, \quad \mathsf{Rat}^{\uparrow}, \quad \mathsf{Met}^{\uparrow}, \quad \mathsf{UniformModulus}^{\uparrow}.$$

The handoff factors through the displayed radius-minimum stability row S, so classifier transport of the finite bundle cannot change the uniform-modulus radius without first changing a visible compact-source, bundle-membership, rational-fold, or metric-comparison row.

Proof. Start from Definition 6113.2. The consumer can reach U only after K, B, Q, S, G have been exposed.

Read the finite minimum fold as in Definition 9237.5. The lower-bound row is produced by finite recursion over the ProbeBundle, so its classifier is determined by displayed bundle membership and rational comparison rows.

Now apply the stability row S. It transports the same finite fold through componentwise hsame data before the metric comparison and uniform-modulus rows are replayed.

Thus the exported uniform-modulus radius is stable under exactly the displayed compact-source, bundle, rational-fold, metric, transport, replay, provenance, and naming rows. No host open-cover proof, hidden selector, quotient metric equality, or ambient completeness row can alter the handoff. □

Falsifiable prediction: Every $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$ read must expose CompactMetric finite-net data, ProbeBundle membership and fold order, RatUp lower-bound exactness, radius-minimum classifier stability, MetricSpace comparison, UniformModulus handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, probebundle, rat, metricspace, uniformmodulus: no listed sibling supplies the whole ten-row packet $M = (K, B, Q, S, G, U, H, C, P, N)$, because each sibling contributes only compact-source, finite-bundle, rational-fold, metric-comparison, or uniform-modulus rows while this chapter binds radius-minimum stability with transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ProbeBundle membership and fold order, RatUp lower-bound exactness, radius-minimum classifier stability, MetricSpace comparison, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6113.1, Definition 6113.2, and Definition 6113.3 seed the finite classifier-stability surface for compact-uniform radius-minimum folds.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformRadiusMinimumStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host open-cover compactness, semantic continuity, hidden minimum selection, quotient metric equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ProbeBundle fold-order exposure, rational lower-bound exactness, classifier stability, metric comparison, uniform-modulus handoff, and non-escape for BEDC.Derived.CompactUniformRadiusM

Chapter 6114

A Concrete Naming Certificate for $\mathsf{FiniteSelectionSequence}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteSelectionSequence}^{\uparrow}$ packet records the finite selector surface used when a Real or Completion argument inspects only bounded windows of a regular Cauchy route. It sits between the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The point of the packet is not to choose a sequence from an infinite family; it is to expose the already bounded selector rows that a diagonal or tail-budget consumer may read without countable choice.

Definition 6114.1 (FiniteSelectionSequence carrier)**.** *A* $\mathsf{FiniteSelectionSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, D, R, T, C, P, N)$$

with the following displayed rows:

1. *W is a finite list of selected* $\mathsf{StreamName}^{\uparrow}$ *windows, each tagged by the request index and the bounded precision at which it is read.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *threshold ledger for those windows.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and D.*
4. *T stores componentwise* hsame *transport for the window, dyadic, readback, replay, provenance, and local naming rows.*
5. *C stores displayed* Cont *replay for the finite request trace.*
6. *P is the* Pkg *provenance over* $\mathsf{FiniteSelectionSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
7. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSelectionSequence}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows W, D, R, T, C, P, N. It has no coordinate for countable choice, an infinite selector, a selected subsequence, quotient streams, or an ambient Real completion principle.

Lean4 ✓ `BEDC.Derived.FiniteSelectionSequenceUp.FiniteSelectionSequenceTasteGate single carrier alignment`

Theorem 6114.2 (FiniteSelectionSequence NameCert obligations)**.** *For every accepted* $\mathsf{FiniteSelectionSequence}^{\uparrow}$ *carrier*

$$S = (W, D, R, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSelectionSequence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed StreamName windows, dyadic thresholds, RegSeqRat readbacks, transport, replay, provenance, and local naming rows;*
2. *classifier stability under componentwise* hsame *replacement of the displayed rows;*
3. *finite window exhaustion, requiring every consumer read to occur at one of the listed request-index and precision pairs in* W*;*
4. *no-choice nonescape, requiring the carrier to expose no countable selector, selected subsequence, quotient stream, or ambient completion datum;*
5. $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring any Real-facing read to pass through* R *after the displayed StreamName and dyadic rows.*

No $\mathsf{FiniteSelectionSequence}^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.FiniteSelectionSequenceUp.FiniteSelectionSequenceCarrier namecert obligations`

Proof. Project the accepted carrier. Its coordinates are exactly W, D, R, T, C, P, N, so carrier admission and classifier stability are coordinatewise statements over the finite BHist packet.

Every retained read is indexed in W, and every tolerance used by that read is listed in D. The readback row R is downstream of those two rows, so a Real-facing consumer reaches $\mathsf{RegSeqRat}^{\uparrow}$ before it reaches any $\mathsf{Real}^{\uparrow}$ seal.

Since the carrier has no field for a countable selector, selected subsequence, quotient stream, or ambient completion principle, transport, replay, provenance, and local naming cannot export such a field. This gives the advertised no-choice boundary. □

Theorem 6114.3 (FiniteSelectionSequence window exhaustion)**.** *Let* $S = (W, D, R, T, C, P, N)$ *be an accepted* $\mathsf{FiniteSelectionSequence}^{\uparrow}$ *carrier. Any Real or Completion consumer that cites* S *must read a finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

through one of the displayed windows in W *and one of the displayed thresholds in* D*. It cannot introduce a later window, an unrecorded dyadic tolerance, a host-selected subsequence, or a countable-choice selector.*

Lean4 ✓ `BEDC.Derived.FiniteSelectionSequenceUp.FiniteSelectionSequenceWindowExhaustion`

Proof. The consumer request is replayed by C, and C is part of the finite packet named by N. Therefore each request has a displayed row in W and its tolerance is one of the rows of D.

The $\mathsf{RegSeqRat}^{\uparrow}$ readback row R is the only row connecting those finite observations to the Real-facing seal. The remaining coordinates T, P, N transport, source, and name the same finite route, so they cannot add windows, tolerances, subsequences, or countable-choice selectors. □

Falsifiable prediction: Every $\mathsf{FiniteSelectionSequence}^{\uparrow}$ Real or Completion-facing read factors through displayed StreamName windows, dyadic thresholds, RegSeqRat readbacks, componentwise transport, replay, provenance, and local naming before reaching a $\mathsf{Real}^{\uparrow}$ seal.

Independence witness: streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteSelectionSequence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite selector-window list, dyadic threshold ledger, RegSeqRat readback, transport, replay, provenance, and local naming while refusing countable choice, selected subsequences, quotient streams, and ambient completion.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSelectionSequence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSelectionSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6114.1, Definition 6114.2, and Definition 6114.3 seed the finite selector-window route from StreamName windows through dyadic thresholds and RegSeqRat readbacks to Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSelectionSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, selected subsequences, quotient stream equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier stability, finite window exhaustion, no-choice nonescape, RegSeqRat and Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteSelectionSequenceUp.

Constructive story: A $\mathsf{FiniteSelectionSequence}^{\uparrow}$ packet is generated from finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ thresholds, $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6115

A Concrete Naming Certificate for $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$

Following Brouwer's fan theorem and Troelstra–van Dalen's constructive treatment of bars, the $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$ packet records the finite prefix-depth route by which a Cantor-space function obtains a uniform modulus from a fan-bar certificate before any Real or Completion consumer reads the result. It sits beside the fan-bar Real route of chapter 1707, the Cantor prefix surface of chapter 1475, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the downstream tolerance-only consumer of chapter 369. The packet is a finite NameCert surface for a fan-theorem modulus on Cantor space; it does not assert open-cover compactness, select an infinite branch, quotient streams, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6115.1 (CantorSpaceUniformModulus carrier)**.** *A* $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, A, B, S, D, R, U, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *row, carrying the decidable bar and the fan-bar depth bound.*
2. *A is the* $\mathsf{CantorSpace}^{\uparrow}$ *address row, exposing the finite Boolean prefix cylinder on which the modulus is inspected.*
3. *B is the finite bar-depth ledger, recording exactly which prefix depth is sufficient for every retained Cantor address.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row that turns the retained prefixes into finite source windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger consumed by the modulus request.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the stream windows and dyadic tolerances.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the fan depth as the exported uniform precision.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, readable only after F, A, B, S, D, R, U have been displayed.*
9. *H stores componentwise* hsame *transport for the fan, Cantor address, bar-depth, stream-window, dyadic-tolerance, regular-rational, uniform-modulus, real-seal, replay, provenance, and local naming rows.*

10. *C stores displayed* Cont *replay from a consumer request through the finite prefix-depth route.*
11. *P is the* Pkg *provenance over* CantorSpaceUniformModulus$^{\uparrow}$, FanTheorem$^{\uparrow}$, CantorSpace$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, UniformModulus$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{CantorSpaceUniformModulus}^{\uparrow}}$ *row.*

The classifier for accepted carriers compares only the displayed rows $F, A, B, S, D, R, U, E, H, C, P, N$. It has no coordinate for open-cover compactness, selected branches, quotient stream equality, an external modulus selector, selected Real limits, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CantorSpaceUniformModulusUp.CantorSpaceUniformModulusTasteGate single carrier alignment`

Theorem 6115.2 (CantorSpaceUniformModulus NameCert obligations)**.** *For every accepted* CantorSpaceUniformM *carrier*

$$M = (F, A, B, S, D, R, U, E, H, C, P, N),$$

the local NameCert$_{\text{CantorSpaceUniformModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed fan, Cantor-prefix, bar-depth, stream-window, dyadic-tolerance, regular-rational, uniform-modulus, Real-seal, transport, replay, provenance, and local naming rows;*
2. *fan-bar depth exactness, requiring B to expose precisely the depth supplied by F for the retained Cantor prefix cylinder A;*
3. *Cantor-prefix window exposure, requiring every source read to pass through A and then S before any rational or Real-facing consumer may inspect it;*
4. *dyadic tolerance exactness, requiring D to be the only tolerance ledger read by R and U;*
5. RegSeqRat$^{\uparrow}$ *readback, requiring the regular rational row R to be downstream of the finite stream windows and dyadic tolerances;*
6. UniformModulus$^{\uparrow}$ *handoff, requiring the exported uniform precision U to be the fan-depth row rather than a host-selected modulus;*
7. *Real-seal nonescape, requiring E to be terminal and to add no ambient completeness, selected limit, or quotient-stream equality;*
8. *ledger refusal, requiring H, C, P, N to transport, replay, source, and name only the displayed finite route.*

No CantorSpaceUniformModulus$^{\uparrow}$ *obligation is stored outside these rows.*

Lean4 ✓ `BEDC.Derived.CantorSpaceUniformModulusUp.CantorSpaceUniformModulusCarrier namecert obligations`

Proof. Project the accepted carrier. The row inventory is exactly $F, A, B, S, D, R, U, E, H, C, P, N$, so carrier admission and ledger refusal are coordinate exhaustion.

The fan-bar depth row is read through F and B. The Cantor address A restricts the read to a finite Boolean prefix cylinder, so the depth used by the modulus route is the displayed fan-bar depth rather than an open-cover or branch-selection datum.

The source-window route then passes through S. Dyadic tolerance is supplied only by D, and the regular rational row R reads those windows and tolerances before the uniform-modulus consumer sees the handoff.

Finally, U exports the fan-depth precision to the $\mathsf{UniformModulus}^{\uparrow}$ surface, while E seals the downstream Real read. Since the carrier contains no selected branch, quotient stream equality, external selector, selected Real limit, or ambient completeness row, no such datum can escape through H, C, P, N. □

Theorem 6115.3 (CantorSpaceUniformModulus sibling route)**.** *Every Real or Completion-facing consumer of an accepted* $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$ *carrier factors through the sibling route*

$$\mathsf{FanTheorem}^{\uparrow} \longmapsto \mathsf{CantorSpace}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto$$

In particular, the fan-modulus dependency is the finite Cantor-space neighbour of chapter 1707, and the exported precision is read by the tolerance-only consumer surface of chapter 369.

Lean4 ✓ `BEDC.Derived.CantorSpaceUniformModulusUp.CantorSpaceUniformModulusSibling route`

Proof. Read the carrier in the order specified by Definition 6115.1. The fan and Cantor rows provide the only prefix-depth source, and the stream, dyadic, and regular-rational rows are the only readback route.

The uniform-modulus row U can therefore consume only the displayed depth and tolerance rows. The Real seal E is downstream of that handoff, so a completion-facing consumer receives the same finite route and no ambient compactness or completeness principle. □

Falsifiable prediction: Every $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$ Real or Completion-facing read factors through the displayed fan row, Cantor prefix row, bar-depth ledger, StreamName window row, dyadic tolerance ledger, RegSeqRat readback row, UniformModulus handoff row, Real seal, transport, replay, provenance, and local naming.

Independence witness: fantheorem, cantorspace, bishop-fan-modulus, uniformmodulus: CantorSpaceUniformMo is not bijective to any listed sibling because it jointly carries the fan-depth row, Cantor prefix cylinder, stream-window readback, dyadic tolerance ledger, regular rational row, UniformModulus handoff, Real seal, and structural support while refusing open-cover compactness, selected branches, quotient streams, external modulus selectors, selected Real limits, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{CantorSpaceUniformModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CantorSpaceUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6115.1, Definition 6115.2, and Definition 6115.3 seed the finite fan-depth route from Cantor prefixes to StreamName, dyadic tolerance, RegSeqRat, UniformModulus, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CantorSpaceUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected infinite branches, quotient stream equality, external modulus selection, selected Real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, fan-bar depth exactness, Cantor-prefix window exposure, dyadic tolerance exactness, RegSeqRat readback, UniformModulus handoff, Real-seal nonescape, transport, replay, provenance, local naming, and ledger refusal for BEDC.Derived.CantorSpaceUniformModulusUp.

Constructive story: A $\mathsf{CantorSpaceUniformModulus}^{\uparrow}$ packet is generated from a finite $\mathsf{FanTheorem}^{\uparrow}$ row, $\mathsf{CantorSpace}^{\uparrow}$ prefix row, fan-depth ledger, $\mathsf{StreamName}^{\uparrow}$ window row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{RegSeqRat}^{\uparrow}$ readback row, $\mathsf{UniformModulus}^{\uparrow}$ handoff row, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6116

A Concrete Naming Certificate for RegularCauchySubsequenceModulus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the RegularCauchySubsequenceModulus$^{\uparrow}$ packet records the finite modulus surface that turns a displayed Cauchy source into a regular subsequence schedule before the RegSeqRat$^{\uparrow}$ and Real$^{\uparrow}$ handoff is read. It sits between the CauchyModulus$^{\uparrow}$ source of chapter 292, the dyadic arithmetic ledger of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the accelerated source interface of chapter 1781. The chapter makes the Cauchy-to-regular-real subsequence modulus explicit as NameCert data rather than treating it as an ambient extraction step.

Definition 6116.1 (Regular Cauchy subsequence modulus carrier)**.** *A* RegularCauchySubsequenceModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (Q, S, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the source* CauchyModulus$^{\uparrow}$ *row, naming the Cauchy tail modulus from which the subsequence schedule is read.*
2. *S is the extracted subsequence schedule row, recording the monotone stage requests and their finite cofinality checks.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tail-budget ledger, carrying dyadic radii, slack comparisons, and the precision budget consumed by S.*
4. *W records the finite* StreamName$^{\uparrow}$ *windows inspected at the stages displayed by S.*
5. *R is the* RegSeqRat$^{\uparrow}$ *handoff row opened only after Q, S, D, W have been displayed.*
6. *E is the downstream* Real$^{\uparrow}$ *seal row reached from the regular readback.*
7. *H records componentwise* hsame *transport for the source modulus, schedule, dyadic budget, stream windows, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a Cauchy tail request through $Q, S, D, W, R, E, H, P, N$.*
9. *P is the* Pkg *provenance over* RegularCauchySubsequenceModulus$^{\uparrow}$, CauchyModulus$^{\uparrow}$, FastCauchyModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $Q, S, D, W, R, E, H, C, P, N$. It has no coordinate for a choice-selected infinite subsequence, quotient stream equality, host sequence equality, host real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden fast-modulus oracle.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubsequenceModulusUp.RegularCauchySubsequenceModulusTasteGate`
`single carrier alignment`

Theorem 6116.2 (Regular Cauchy subsequence modulus surface)**.** *For every accepted* RegularCauchySubsequenceM *carrier*

$$M = (Q, S, D, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}}$ *surface consists exactly of five obligation rows: source Cauchy modulus exposure through Q, extracted subsequence schedule exposure through S, dyadic tail-budget and finite-window precision through D, W, RegSeqRat and Real handoff through R, E, and structural transport with replay, provenance, and local naming through H, C, P, N.*

Proof. Project the carrier of Definition 6116.1. The accepted packet displays only the source modulus row Q, schedule row S, dyadic budget row D, stream-window row W, regular readback R, Real seal E, and structural rows H, C, P, N.

The precision argument is finite. Induct over the displayed tail-budget list in D: the empty budget exposes no stage, while the next dyadic request reads the corresponding source-modulus entry in Q, records the selected stage in S, and checks the retained window in W. The induction never asks for a global selector, because every step consumes one visible budget row.

The handoff is therefore forced. The $\mathsf{RegSeqRat}^{\uparrow}$ row R may only read the schedule and finite windows already justified by Q, S, D, W, and the $\mathsf{Real}^{\uparrow}$ seal E appears downstream of R.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no coordinate stores quotient stream equality, selected subsequences, host equality, selected limits, ambient completeness, or a hidden $\mathsf{FastCauchyModulus}^{\uparrow}$ oracle, the five listed groups exhaust the local NameCert surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}$ route to a RegSeqRat or Real boundary must display a source CauchyModulus row, extracted subsequence schedule, dyadic tail budget, finite StreamName windows, hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cauchymodulus, fastcauchymodulus, dyadicratcore, streamname, regseqrat, real: none is bijective with $\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}$ because this packet jointly carries the source modulus, schedule extraction, dyadic precision ledger, finite windows, regular handoff, Real seal, transport, replay, provenance, and local naming for the same Cauchy-to-regular route.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}$ packet is generated from a source CauchyModulus row, extracted subsequence schedule, DyadicRatCore tail-budget ledger, finite StreamName windows, RegSeqRat handoff, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6116.1 and Definition 6116.2 seed the finite subsequence-modulus interface used by Cauchy-to-RegSeqRat and Real routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySubsequenceModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, choice-selected infinite subsequences, host sequence equality, host real equality, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source Cauchy modulus exposure, schedule extraction, dyadic precision induction, finite StreamName window routing, RegSeqRat and Real handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubsequenceModulusUp.

Chapter 6117

A Concrete Naming Certificate for $\mathsf{CertifiedModelQuality}^{\uparrow}$

The $\mathsf{CertifiedModelQuality}^{\uparrow}$ packet realizes the model-quality scope of section 9462.15 as finite certificate rows. It reads classifier support, behavior coverage, score accounting, and debt gates from displayed ledgers before any consumer may call a model-state transition a certified quality improvement. The packet sits beside the classifier-formation surface of chapter 6097, the GroundCompiler audit-map surface of chapter 836, the certificate gate of chapter 1002, and the effective replacement ledger of chapter 568.

Definition 6117.1 (CertifiedModelQuality carrier)**.** *A* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (S, B, L, W, D, A, G, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the declared model-quality scope row, naming the domain, model state, admitted transformations, critical behavior family, and public protocol.*
2. *B is the certified behavior coverage row, listing the finite critical behaviors whose classifier support and local* $\mathsf{NameCert}$ *rows are displayed.*
3. *L is the model-quality ledger row, recording benefit, cost, debt, failure cases, verification status, safety boundary, shortcut risk, and not-claimed scope.*
4. *W is the public score-accounting row, reading the weighted benefit, cost, and debt comparison from L.*
5. *D is the explicit debt row whose absence blocks certification of an improvement.*
6. *A is the audit-gate row, citing* $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ *and* $\mathsf{CertificateAuditGate}^{\uparrow}$ *consumers for machine-readable checking.*
7. *G is the replacement and belief-separation row, citing* $\mathsf{EffectiveReplacementLedger}^{\uparrow}$ *and* $\mathsf{Belief}^{\uparrow}$ *only as sibling dependencies rather than as equivalent model-quality packets.*
8. *R is the quality-gate decision row, accepting only when coverage, critical debt, safety, transfer, and verification ledgers meet the public protocol.*
9. *H records componentwise* hsame *transport for scope, behavior coverage, ledger, score, debt, audit, sibling-separation, gate, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from scope inspection through coverage, ledger, score, debt, audit, sibling separation, gate, provenance, and local naming.*
11. *P is the* Pkg *provenance over* $\mathsf{CertifiedModelQuality}^{\uparrow}$, $\mathsf{ClassifierFormingLearning}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, $\mathsf{CertificateAuditGate}^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, $\mathsf{Belief}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, L, W, D, A, G, R, H, C, P, N$. It has no coordinate for raw benchmark superiority, private evaluator preference, host semantic truth, hidden training success, unledgered classifier gain, or an equivalence between belief commitment and certified model quality.

Theorem 6117.2 (CertifiedModelQuality NameCert obligations)**.** *For every accepted* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *carrier*

$$Q = (S, B, L, W, D, A, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}^{\uparrow}}$ *surface consists exactly of scope exposure, certified behavior coverage, model-quality ledger accounting, public score comparison, explicit debt attachment, audit-gate handoff, sibling-separation handoff, quality-gate decision, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6117.1. The accepted packet displays exactly the scope row S, behavior-coverage row B, ledger row L, score row W, debt row D, audit row A, sibling-separation row G, gate row R, and structural rows H, C, P, N.

The ledger and score rows are inseparable. The score row W may compare two model states only through the benefit, cost, and debt entries shown in L, and the debt row D records the critical residue that must be accounted for before the comparison can be certified.

The audit and sibling rows keep the packet public and non-bijective. The $\mathsf{GroundCompilerAuditMap}^{\uparrow}$ and $\mathsf{CertificateAuditGate}^{\uparrow}$ entries in A make the quality read machine-checkable as audit data, while G records that $\mathsf{EffectiveReplacementLedger}^{\uparrow}$ and $\mathsf{Belief}^{\uparrow}$ are neighboring dependencies, not substitutes for model-quality certification.

Finally, R, H, C, P, N accept, transport, replay, source, and name the same finite route. Since no excluded coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6117.3 (CertifiedModelQuality debt row gate)**.** *An accepted* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *packet cannot certify a model-quality improvement when the debt row D is absent from the displayed carrier. In that case the score comparison remains an uncertified performance, preference, or classifier-support observation rather than a certified model-quality improvement.*

Proof. Start with the carrier projection. The quality score is read through W, and W depends on the ledger row L together with the explicit debt row D.

If D is absent, the packet has no displayed coordinate for black-box debt, ledger debt, stability debt, generalization debt, safety debt, shortcut debt, semantic debt, or verification debt. The quality gate R therefore lacks the row it must inspect before accepting the comparison.

The remaining rows still may record behavior coverage, audit handoff, or score movement, but none of them replaces D. Thus the packet refuses certification of improvement while preserving the weaker displayed observations. □

Falsifiable prediction: A $\mathsf{CertifiedModelQuality}^{\uparrow}$ packet that claims certified model-quality improvement while omitting the debt row D is rejected by the local NameCert surface.

Independence witness: GroundCompilerAuditMap, CertificateAuditGate, EffectiveReplacementLedger, Belief: none is bijective with $\mathsf{CertifiedModelQuality}^{\uparrow}$ because audit maps, gate passes, replacement ledgers, and belief commitments each omit at least one of the scope, behavior-coverage, score-accounting, debt, quality-gate, transport, replay, provenance, or local naming rows.

The Lean follow-up must provide explicit $\mathsf{Nontrivial\ CertifiedModelQuality}^{\uparrow}$ and $\mathsf{FieldFaithful\ CertifiedModelQualit}$ instances in `lean4/BEDC/Derived/CertifiedModelQualityUp/TasteGate.lean`. Those instances must preserve the rows $S, B, L, W, D, A, G, R, H, C, P, N$ so the checked TasteGate surface separates debt-aware model quality from score movement, audit-gate success, replacement-ledger availability, and belief commitment.

Closure status for $\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}^{\uparrow}}$.

Constructive story: A $\mathsf{CertifiedModelQuality}^{\uparrow}$ packet is generated from model-quality scope, certified behavior coverage, ledger accounting, public score comparison, explicit debt, audit-gate handoff, sibling-separation handoff, quality-gate decision, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6117.1, Definition 6117.2, and Definition 6117.3 seed the finite certificate packet for debt-aware model-quality certification.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CertifiedModelQuality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close benchmark validity, causal explanation, host semantic truth, hidden training success, global model safety, Lean checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, behavior coverage, ledger and score accounting, debt-row gate refusal, audit-gate handoff, sibling separation, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.CertifiedModelQualityUp.

Chapter 6118

A Concrete Naming Certificate for $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Mac Lane on adjunction counits, the $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$ packet records the regular-source face of the completion reflector's return map. It sits between the source-side unit of chapter 3646, the general counit packet of chapter 1539, and the monadic completion surface of chapter 378. Its readback route passes through the $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ packets of chapter 126, chapter 301, chapter 291, and chapter 38. The packet names the counit face only as finite BEDC data; it is not quotient-stream equality, a chosen regular representative, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6118.1 (RegularCauchyCompletionCounit carrier)**.** *A* $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, U, Q, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the completion-facing regular source row whose accepted read is being returned.*
2. *U is the* $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$ *admission route that records how the regular source entered the reflector surface.*
3. *Q is the* $\mathsf{CauchyCompletionCounit}^{\uparrow}$ *return row restricting the general counit to the regular source face.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the returned regular name is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row attached to the same finite windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger accounting for each precision request used by S and R.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff reached only after U, Q, S, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the completion-facing source, unit route, counit row, stream windows, regular readback, dyadic ledger, seal handoff, replay, provenance, and naming rows.*
9. *C records displayed* Cont *replay from the completion-facing source through the unit route, counit row, finite windows, dyadic ledger, regular readback, and real seal.*

10. *P records* Pkg *provenance over* $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$, $\mathsf{RegularCauchyCompletionUnit}^{\uparrow}$, $\mathsf{CauchyCompletionCounit}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyCompletionCounit}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient streams, selected representatives, host sequence equality, countable-choice schedules, analytic closure of the monad, ambient completion, or completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.RegularCauchyCompletionCounitUp.RegularCauchyCompletionCounitTasteGate`
`single carrier alignment`

Theorem 6118.2 (RegularCauchyCompletionCounit NameCert obligations)**.** *For an accepted* $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$ *carrier* $K = (M, U, Q, S, R, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComple}}$ *surface consists exactly of completion-facing source admission through* M*, regular unit admission through* U*, general counit restriction through* Q*, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure through* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance accounting through* D*,* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6118.1. Its displayed coordinates are exactly $M, U, Q, S, R, D, E, H, C, P, N$.

The source and adjunction rows are M, U, Q. The unit row U records how the regular source entered the completion-facing surface, and the counit row Q records the return route restricted to that same regular face.

The finite observation rows are S, R, D. They expose stream windows, regular rational readback, and dyadic tolerance accounting before any terminal real seal is consumed.

The seal handoff E is downstream of those rows, while H, C, P, N preserve, replay, source, and name the same finite route. No other local obligation is present. □

Theorem 6118.3 (RegularCauchyCompletionCounit real-seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyCompletionCounit}^{\uparrow}$ *packet reaches the seal row* E *only through the displayed route*

$$M \longmapsto U \longmapsto Q \longmapsto S \longmapsto D \longmapsto R \longmapsto E$$

together with H, C, P, N*. Thus the regular counit reads the completion return as finite regular Cauchy data with stream windows and dyadic tolerance accounting, not as quotient-stream equality, a selected representative, or an ambient completion principle.*

Proof. Begin with a real-seal consumer. By the carrier definition, the only completion-facing source row available to that consumer is M.

The consumer must next pass through U and Q. These rows identify the source admission as regular and restrict the general counit return to that regular source face.

The finite readback is then forced through S, D, R. Stream windows name the inspected observations, the dyadic ledger accounts for the requested tolerance, and the regular rational row supplies the readback before E is inspected.

Finally H, C, P, N transport, replay, source, and name the same route. The seal handoff is therefore the displayed factorization, not a hidden quotient, selector, or completeness witness. □

Theorem 6118.4 (RegularCauchyCompletionCounit non-escape)**.** *No accepted* RegularCauchyCompletionCounit$^{\uparrow}$ *packet exports data outside* $M, U, Q, S, R, D, E, H, C, P, N$*. In particular, it cannot be used to recover quotient-stream equality, a chosen regular representative, host sequence equality, countable-choice schedules, analytic closure of* CauchyCompletionMonad$^{\uparrow}$*, ambient completion, or completeness of* Real$^{\uparrow}$*.*

Proof. Inspect the carrier. The completion-facing source, unit admission, counit restriction, stream-window, regular readback, dyadic ledger, seal handoff, transport, replay, provenance, and local-name rows are the only fields.

Apply Definition 6118.2. Every accepted local obligation is a projection of those rows or a structural transport, replay, provenance, or naming row for them.

Use Definition 6118.3. Every real-seal consumer passes through $M, U, Q, S, D, R, E, H, C, P, N$, so no excluded quotient, selector, host, monadic closure, ambient completion, or real-completeness object can be exported. □

Closure status for NameCert$_{\text{RegularCauchyCompletionCounit}^{\uparrow}}$**.**

Constructive story: A RegularCauchyCompletionCounit$^{\uparrow}$ packet is generated from a completion-facing regular source row, RegularCauchyCompletionUnit admission, CauchyCompletionCounit return restriction, StreamName finite windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal handoff, componentwise transport, displayed replay, package provenance, and local naming.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyCompletionCounit}^{\uparrow}}$, seed)

Scope closed: Definition 6118.1, Definition 6118.2, Definition 6118.3, and Definition 6118.4 seed the finite regular-source counit face of the completion reflector.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyCompletionCounit}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean theorem checking, quotient-stream equality, selected representatives, countable choice, host sequence equality, analytic closure of CauchyCompletionMonad$^{\uparrow}$, ambient completion, or completeness of Real$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, unit-route admission, counit restriction, finite window exposure, regular readback, dyadic tolerance accounting, seal handoff, transport stability, provenance, and non-escape for the displayed packet.

Lean4 (definition) `BEDC.Derived.RegularCauchyCompletionCounitUp.RegularCauchyCompletionCounitUp`

Chapter 6119

A Concrete Naming Certificate for $\mathsf{ConstantCauchyName}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{ConstantCauchyName}^{\uparrow}$ packet records the constant Cauchy name as the finite route from a $\mathsf{DyadicRatCore}^{\uparrow}$ source row through $\mathsf{StreamName}^{\uparrow}$ constant windows, the $\mathsf{ConstantRegSeqRat}^{\uparrow}$ readback of chapter 6107, $\mathsf{RegSeqRat}^{\uparrow}$ regularity, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet gives Real and Completion consumers a named constant-name embedding without host constants, quotient streams, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6119.1 (Constant Cauchy name carrier)**.** *A* $\mathsf{ConstantCauchyName}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *source row naming the dyadic rational point.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *constant-window schedule, so every requested precision opens a finite window whose entries read back to the same dyadic source row.*
3. *Q is the* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *stationary readback row.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row inherited from the stationary readback.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after D, W, Q, R have been displayed.*
6. *H stores componentwise* hsame *transport for the source, windows, readback, regularity, seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a Real-facing constant-name request through the same finite route.*
8. *P is the* Pkg *provenance over* $\mathsf{ConstantCauchyName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ConstantRegSeqRat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstantCauchyName}^{\uparrow}}$ *row.*

The carrier has no coordinate for host real constants, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ConstantCauchyNameUp.ConstantCauchyNameTasteGate single carrier alignment`

Theorem 6119.2 (Constant Cauchy name NameCert obligations)**.** *For every accepted* $\mathsf{ConstantCauchyName}^{\uparrow}$ *carrier*

$$K = (D, W, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstantCauchyName}^{\uparrow}}$ *surface consists exactly of carrier admission for the nine displayed rows, dyadic source exposure through* D*, constant-window exposure through* W*, stationary readback through* Q*, regularity exposure through* R*, Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, local naming through* N*, and non-escape from host constants or quotient streams.*

Proof. Project the accepted carrier of Definition 6119.1. The tuple lists exactly the dyadic source, constant windows, stationary readback, regularity, Real seal, and structural rows.

Read the source route in order. The dyadic point is first exposed by D; then W opens only finite constant windows; then Q invokes the $\mathsf{ConstantRegSeqRat}^{\uparrow}$ readback of chapter 6107; and then R records the same route as $\mathsf{RegSeqRat}^{\uparrow}$ regularity.

Finally E, H, C, P, N seal, transport, replay, source, and name only that displayed route. Since no carrier coordinate stores host constants, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6119.3 (Constant Cauchy name real seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer admitted by an accepted* $\mathsf{ConstantCauchyName}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longrightarrow \mathsf{StreamName}^{\uparrow} \longrightarrow \mathsf{ConstantRegSeqRat}^{\uparrow} \longrightarrow \mathsf{RegSeqRat}^{\uparrow} \longrightarrow \mathsf{Real}^{\uparrow}$$

before the constant name is available as Real data. Thus the constant embedding is read as a finite BEDC route and not as a host constant, quotient Cauchy class, selected limit, or ambient completeness claim.

Proof. Begin with an accepted carrier $K = (D, W, Q, R, E, H, C, P, N)$. A Real-facing read has no access to the terminal seal E until it has displayed the source row D, the constant windows W, the stationary readback Q, and the regularity row R.

Use the $\mathsf{ConstantRegSeqRat}^{\uparrow}$ obligation surface of chapter 6107. It makes the stationary rational readback a visible row between the $\mathsf{StreamName}^{\uparrow}$ windows and the $\mathsf{RegSeqRat}^{\uparrow}$ regularity surface.

The structural rows H, C, P, N preserve the same route under transport, replay, provenance, and local naming. Therefore every Real seal read factors through the displayed dyadic, stream, constant-regular, regular-rational, and Real rows, and the excluded host or quotient principles remain outside the packet. □

Theorem 6119.4 (Constant Cauchy name dyadic source lock)**.** *Let* $K = (D, W, Q, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ConstantCauchyName}^{\uparrow}$ *carrier. Every consumer exposing the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *must first expose the same dyadic source row* D*, the same constant-window row* W*, the same* $\mathsf{ConstantRegSeqRat}^{\uparrow}$ *row* Q*, and the same* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*. No consumer may replace* D *by a host real constant, a quotient representative, a selected limit, or an ambient completeness witness.*

Proof. Project the carrier of Definition 6119.1. The terminal seal row E is downstream of D, W, Q, R, and there is no separate row from which a host constant or quotient representative could be read.

Use the NameCert obligations of Definition 6119.2. They expose the dyadic source through D, the constant-window schedule through W, the stationary readback through Q, and regularity through R as the finite public route of the packet.

Apply the Real seal theorem Definition 6119.3. It forces every $\mathsf{Real}^{\uparrow}$ consumer to factor through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ConstantRegSeqRat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and only then $\mathsf{Real}^{\uparrow}$.

The structural rows H, C, P, N transport, replay, source, and name the same displayed data. Therefore the source lock is part of the carrier surface, not an assertion about host real constants, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{ConstantCauchyName}^{\uparrow}$ read must expose a $\mathsf{DyadicRatCore}^{\uparrow}$ source row, a $\mathsf{StreamName}^{\uparrow}$ constant-window schedule, a $\mathsf{ConstantRegSeqRat}^{\uparrow}$ stationary readback, a $\mathsf{RegSeqRat}^{\uparrow}$ regularity row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before any $\mathsf{Real}^{\uparrow}$ constant-name consumer may proceed.

Independence witness: dyadicratcore, streamname, constantregseqrat, regseqrat, real: none is bijective with $\mathsf{ConstantCauchyName}^{\uparrow}$ because this packet jointly carries the dyadic source, constant windows, stationary regular readback, Real seal, transport, replay, provenance, and local naming for the same constant-name route.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstantCauchyName}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstantCauchyName}^{\uparrow}$ packet is generated from a DyadicRatCore source row, StreamName constant windows, a ConstantRegSeqRat stationary readback, RegSeqRat regularity, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ConstantCauchyName}^{\uparrow}}$, seed)

Scope closed: Definition 6119.1, Definition 6119.2, and Definition 6119.3 seed the finite constant-name route through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{ConstantRegSeqRat}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ConstantCauchyName}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host real constants, quotient stream equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, constant-window exposure, stationary readback, regularity, Real seal, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.ConstantCauchyNameUp.

Chapter 6120

A Concrete Naming Certificate for $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$

Following Kolmogorov's consistency principle for finite-dimensional distributions, the KolmogorovExtensionFiniteWi packet records only the finite-window surface that a BEDC probability consumer may inspect before any infinite product construction is requested. It sits over the probability carrier of chapter 184, the distribution ledger of chapter 186, the measurable random-variable rows of chapter 185, the finite observation windows of chapter 126, and the regular rational readback of chapter 301. The packet names finite marginal windows and their compatibility rows; it does not import infinite product measures, countable choice, a host sigma-product construction, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6120.1 (Kolmogorov finite-window carrier)**.** *A* $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, W, \mu, \pi, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite index-window ledger, naming the inspected coordinate set and its included subwindows.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation schedule used to read the finite coordinate windows.*
3. *μ is the family of finite* $\mathsf{Distribution}^{\uparrow}$ *rows over the displayed windows, read inside the probability context of* $\mathsf{ProbSpace}^{\uparrow}$*.*
4. *π is the marginal-projection row, recording how a larger displayed window restricts to a smaller displayed window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed probabilities and comparison tolerances.*
6. *H records componentwise* hsame *transport for I, W, μ, π, R.*
7. *C records displayed* Cont *replay from a marginal request through window exposure, distribution readback, projection, and rational comparison.*
8. *P is the* Pkg *provenance over* $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$*,* $\mathsf{ProbSpace}^{\uparrow}$*,* $\mathsf{Distribution}^{\uparrow}$*,* $\mathsf{RandomVar}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}}$ *row.*

The carrier has no coordinate for an infinite sample space, product sigma algebra, countable selection of cylinder representatives, completed measure, host random process, or ambient real-completeness principle.

Lean4 ✓ `BEDC.Derived.KolmogorovExtensionFiniteWindowUp.KolmogorovExtensionFiniteWindowTasteGate single carrier alignment`

Theorem 6120.2 (Kolmogorov finite-window NameCert obligations)**.** *For every accepted* KolmogorovExtensionFini *carrier* $K = (I, W, \mu, \pi, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}}$ *surface consists of five finite obligations:*

1. *carrier admission for the displayed finite index windows, observation schedule, distribution rows, projection rows, readback rows, transports, replay, provenance, and local name;*
2. *window exposure, requiring every marginal request to name a displayed subwindow of* I *before a distribution row is read;*
3. *distribution compatibility, requiring each finite window to read its probability data through* $\mathsf{Distribution}^{\uparrow}$ *and* $\mathsf{ProbSpace}^{\uparrow}$ *rows;*
4. *marginal projection exactness, requiring the row* π *to compare larger-window distributions with smaller-window distributions through the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*;*
5. *nonescape, excluding infinite product measures, countable choice, host sigma-product construction, quotient random processes, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6120.1. The tuple displays exactly the finite rows that the local certificate may name.

Window exposure follows from I and W. A request that does not name a displayed window has no route to a distribution row.

Distribution compatibility is the role of μ and P. The finite probability data are read through the existing $\mathsf{Distribution}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, and $\mathsf{RandomVar}^{\uparrow}$ sibling surfaces rather than through a host probability space.

Marginal projection exactness is recorded by π and checked through R. The same regular-rational readback compares both the larger-window and smaller-window probability values.

Finally, H, C, P, N only transport, replay, source, and name those rows. Since no omitted object is a coordinate of K, the nonescape clause is part of the local NameCert surface. □

Theorem 6120.3 (Kolmogorov finite-window marginal consistency)**.** *Let* $K = (I, W, \mu, \pi, R, H, C, P, N)$ *be an accepted* $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$ *carrier. If a displayed window* J *is included in a displayed window* L *inside* I*, then every marginal read from the* L*-distribution row to the* J*-distribution row factors through*

$$I, W, \mu, \pi, R, H, C, P, N.$$

Thus the finite-window packet supplies marginal consistency for displayed windows only, and it exports no infinite product measure, countable-choice selector, host sigma-product construction, quotient stochastic process, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the inclusion $J \subseteq L$ from the finite index-window ledger I. The observation schedule W exposes exactly the coordinate windows that may be inspected.

The distribution rows μ provide the probability data for J and L. The projection row π is the only row that relates the larger displayed distribution to the smaller displayed distribution.

The comparison is read through R, then transported and replayed by H and C. Provenance and local naming through P, N preserve the same finite route.

Because the carrier has no product sigma algebra, completed infinite product measure, countable selector, or host stochastic-process coordinate, the theorem closes only the displayed finite-window marginal consistency surface. □

Falsifiable prediction: An accepted $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$ consumer that reads a smaller-window distribution without displaying the containing finite window, the projection row, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback refutes the carrier surface.

Independence witness: probspace, distribution, randomvar, streamname, regseqrat: no listed sibling supplies the full finite-window consistency packet $K = (I, W, \mu, \pi, R, H, C, P, N)$, because each sibling contributes only probability context, finite distribution data, measurable random-variable reads, observation windows, or rational readback while this chapter binds them to marginal projection exactness and nonescape.

Closure status for $\mathsf{NameCert}_{\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}$ packet is generated from finite index windows, stream observation rows, finite distribution rows, marginal projection rows, regular rational readback, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6120.1, Definition 6120.2, and Definition 6120.3 seed the finite-window marginal-consistency surface over probability, distribution, random-variable, stream-window, and regular-rational rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KolmogorovExtensionFiniteWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close infinite product measures, countable choice, host sigma-product construction, quotient random processes, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window exposure, distribution compatibility, marginal projection exactness, readback stability, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.KolmogorovExtensionFiniteWindowUp.

Chapter 6121

A Concrete Naming Certificate for $\mathsf{IntermediateValueRealSeal}^{\uparrow}$

Following Bolzano and Bishop–Bridges constructive analysis, the $\mathsf{IntermediateValueRealSeal}^{\uparrow}$ packet records the finite handoff from displayed intermediate-value bisection data to the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the constructive intermediate-value surface of chapter 1402, the bisection route of chapter 1422, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet names a finite sign-change route into a real seal; it does not claim unrestricted intermediate value, selected roots, real trichotomy, quotient real equality, countable choice, or ambient completeness.

Definition 6121.1 (IntermediateValueRealSeal carrier)**.** *An* $\mathsf{IntermediateValueRealSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (I, B, A, S, W, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the located interval endpoint row, exposing the lower and upper rational endpoint cells and their located-order witness.*
2. *B is the* $\mathsf{IntermediateValueBisection}^{\uparrow}$ *branch row, recording the retained finite bisection prefix and the interval chosen at the requested tolerance.*
3. *A is the intermediate-value source row, tying the branch to the displayed function, endpoint signs, and local continuity budget.*
4. *S is the sign-change ledger, containing the initial endpoint signs and the transported midpoint signs used by B.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window surface for the retained endpoint names and the bisection readback window.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from W and the displayed dyadic tolerance request.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after Q has been read.*
8. *H records componentwise* hsame *transport for I, B, A, S, W, Q, R and the structural rows.*
9. *C records displayed* Cont *replay from endpoint inspection through bisection retention, stream windows, regular readback, and real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{IntermediateValueRealSeal}^{\uparrow}$, $\mathsf{IntermediateValueBisection}^{\uparrow}$, $\mathsf{IntermediateValue}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{IntermediateValueRealSeal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an unrestricted intermediate-value principle, a selected root, quotient equality of real names, real trichotomy, countable choice, a completed nested intersection, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6121.2 (IntermediateValueRealSeal obligations)**.** *For every accepted* $\mathsf{IntermediateValueRealSeal}^{\uparrow}$ *carrier* $Z = (I, B, A, S, W, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{IntermediateValueRealSeal}^{\uparrow}}$ *surface consists exactly of located endpoint admission, intermediate-value source exposure, sign-change ledger admission, bisection branch retention,* $\mathsf{StreamName}^{\uparrow}$ *window admission,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback admission, terminal* $\mathsf{Real}^{\uparrow}$ *seal admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $I, B, A, S, W, Q, R, H, C, P, N$.

Lean4 ✓ `BEDC.Derived.IntermediateValueRealSealUp.IntermediateValueRealSealCarrier namecert obligations`

Proof. Project the accepted carrier of Definition 6121.1. The endpoint, source, branch, sign, window, readback, seal, transport, replay, provenance, and naming rows are the whole visible packet.

Read the analytic route first. The located endpoints in I and the source row A provide the finite sign-change request, while the sign ledger S and bisection row B record the retained branch at the requested tolerance.

The Real-facing handoff is then forced through W, Q, R. The stream windows expose only finite endpoint observations, the regular rational readback seals those observations as Q, and the terminal real-name row R is available only after that readback.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no other analytic coordinate, the listed obligations exhaust the local certificate and the excluded root-selection, quotient, trichotomy, choice, and completeness principles remain unavailable. □

Theorem 6121.3 (IntermediateValueRealSeal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* IntermediateValueRealS *packet factors through the ordered finite route*

$$\mathsf{IntermediateValue}^{\uparrow}, \mathsf{IntermediateValueBisection}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{Name}$$

The consumer receives the terminal real seal only as the endpoint of the displayed bisection, window, and regular-readback chain. It receives no unrestricted intermediate-value theorem, selected root, real trichotomy, quotient real equality, countable-choice selector, hidden completeness principle, or independent real representative.

Proof. Begin with the source and branch rows. The intermediate-value row A supplies only the displayed sign-change and continuity data, and the bisection row B selects only the retained finite branch justified by the sign ledger S.

Next read the finite observation surface. The stream row W records the windows through which the retained endpoints are inspected, and the regular rational row Q is the only readback exported from those windows.

Now seal the Real-facing endpoint. The row R is downstream of Q, so a $\mathsf{Real}^{\uparrow}$ consumer cannot bypass the bisection branch or the regular readback by reading a host root, quotient stream, selected representative, or ambient completeness witness.

The structural rows H, C, P, N preserve and name that same route. Composing these projections gives exactly the ordered handoff in the statement and no further real-facing coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{IntermediateValueRealSeal}\uparrow}$.

Constructive story: An $\mathsf{IntermediateValueRealSeal}^{\uparrow}$ packet is generated from located interval endpoint rows, intermediate-value source rows, a sign-change ledger, a bisection branch row, StreamName windows, RegSeqRat readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{IntermediateValueRealSeal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6121.1, Definition 6121.2, and Definition 6121.3 seed the finite route from bisection-style intermediate-value data through regular rational readback to a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{IntermediateValueRealSeal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted intermediate value, selected roots, real trichotomy, quotient real equality, countable choice, completed nested intersections, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source exposure, sign-ledger retention, bisection branch routing, StreamName window admission, RegSeqRat readback, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.IntermediateValueRealSealUp.

Chapter 6122

A Concrete Naming Certificate for $\mathsf{HyperbolicAxisShadow}^{\uparrow}$

Following Bridson–Haefliger and Beardon on hyperbolic axes and shadows, the $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ packet records a finite shadow route associated to axis-like displacement data without selecting a global axis. It consumes the translation-length surface of chapter 5832, the routed geodesic-flow surface of chapter 3856, the metric-read surface of chapter 3474, the Busemann boundary surface of chapter 3732, the terminal real seal of chapter 38, and the observer-expansion vision of chapter 9463. The packet names finite axis-shadow evidence; it does not assert a selected host axis, global hyperbolic manifold, quotient boundary equality, complete geodesic line, or physical observer law.

Definition 6122.1 (HyperbolicAxisShadow carrier)**.** *A* $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Y = (T, F, M, B, R, A, S, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{HyperbolicTranslationLength}^{\uparrow}$ *row, exposing the finite displacement-length comparison that motivates the shadow read.*
2. *F is the* $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$ *row, recording the routed flow window along which the shadow is inspected.*
3. *M is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row, supplying the finite metric comparisons used to bound the visible shadow.*
4. *B is the* $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$ *row, recording the boundary-facing direction data through displayed Busemann rows.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *row used by the translation-length, metric, Busemann, and shadow-size reads.*
6. *A is the finite axis-shadow ledger, relating the displacement window, flow window, metric comparison, and boundary-facing row without naming a selected axis.*
7. *S is the shadow-stability row, requiring the same visible endpoint and metric data to be replayed by T, F, M, B, R, A.*
8. *H records componentwise* hsame *transport for T, F, M, B, R, A, S and the structural rows.*
9. *C records displayed* Cont *replay from a shadow request through $T, F, M, B, R, A, S, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicAxisShadow}^{\uparrow}$, $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicAxisShadow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host selected axis, a completed bi-infinite geodesic, quotient boundary equality, global hyperbolic manifold, curvature axiom, or observer-expansion law.

Lean4 ✓ `BEDC.Derived.HyperbolicAxisShadowUp.HyperbolicAxisShadowTasteGate single carrier alignment`

Theorem 6122.2 (HyperbolicAxisShadow obligations)**.** *For every accepted* $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ *carrier* $Y = (T, F, M, B, R, A, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicAxisShadow}^{\uparrow}}$ *surface consists exactly of translation-length visibility, geodesic-flow routing, hyperbolic metric comparison, Busemann boundary exposure, real-valued readback, finite axis-shadow ledger admission, shadow-stability replay, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $T, F, M, B, R, A, S, H, C, P, N$*.*

Proof. Project the carrier of Definition 6122.1. The rows $T, F, M, B, R, A, S, H, C, P, N$ are the whole visible packet.

Read the geometric data first. Translation length supplies the displacement comparison, geodesic flow supplies the routed inspection window, the metric row supplies finite distance comparisons, and the Busemann boundary row supplies the boundary-facing direction data.

The finite shadow ledger is then read through A and S. These rows relate the displacement, flow, metric, boundary, and real readback data while refusing to name a selected axis or a host geodesic line.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier contains no coordinate for the excluded global geometry, the listed rows exhaust the local certificate surface. □

Theorem 6122.3 (HyperbolicAxisShadow route)**.** *Every accepted* $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ *read factors through the ordered finite route*

$$\mathsf{HyperbolicTranslationLength}^{\uparrow}, \mathsf{HyperbolicGeodesicFlow}^{\uparrow}, \mathsf{HyperbolicMetric}^{\uparrow}, \mathsf{HyperbolicBusemannBoundary}^{\uparrow}, \mathsf{Real}^{\uparrow} \longmapsto$$

The route exports a finite shadow record tied to displayed displacement, flow, metric, boundary, and real-valued rows. It exports no selected invariant axis, host hyperbolic manifold, quotient boundary equality, completed geodesic line, curvature axiom, or global observer-expansion credential.

Proof. Start with the displacement and flow rows. The translation-length row T gives the finite length-facing comparison, and the geodesic-flow row F supplies the visible route along which the comparison is inspected.

Next read the metric and boundary rows. The metric row M bounds the finite shadow window, while the Busemann boundary row B records the boundary-facing data. The Real row R seals the numeric reads used by those comparisons.

The axis-shadow ledger A and stability row S bind those four geometric faces into one finite shadow route without selecting a host axis. The structural rows H, C, P, N then preserve and name that same route.

Composing these projections gives exactly the finite handoff in the statement. Every listed global or quotient object remains outside the carrier and cannot be exported by an accepting consumer. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ consumer is refuted if it obtains an axis-shadow read without displaying the $\mathsf{HyperbolicTranslationLength}^{\uparrow}$, $\mathsf{HyperbolicGeodesicFlow}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicBusemannBoundary}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, finite shadow ledger, stability, transport, replay, provenance, and local NameCert rows.

Independence witness: hyperbolic-translation-length, hyperbolic-geodesic-flow, hyperbolic-metric, hyperbolic-busemann-boundary, real: no listed sibling is bijective to $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ because the packet jointly carries displacement length, routed flow, metric comparison, Busemann boundary direction, Real sealing, finite shadow ledger, stability, transport, replay, package provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicAxisShadow}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicAxisShadow}^{\uparrow}$ packet is generated from HyperbolicTranslationLength rows, HyperbolicGeodesicFlow rows, HyperbolicMetric rows, HyperbolicBusemannBoundary rows, RealUp seal rows, a finite axis-shadow ledger, stability rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicAxisShadow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6122.1, Definition 6122.2, and Definition 6122.3 seed the finite non-selected axis-shadow route over translation-length, flow, metric, boundary, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicAxisShadow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected invariant axes, completed geodesic lines, quotient boundary equality, host hyperbolic manifolds, curvature axioms, global observer-expansion laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, translation-length visibility, flow routing, metric comparison, Busemann boundary exposure, Real sealing, shadow-ledger stability, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicAxisShadowUp.

Chapter 6123

A Concrete Naming Certificate for $\mathsf{AbelUniform}^{\uparrow}$

Following Abel's theorem and Bishop–Bridges constructive analysis, the $\mathsf{AbelUniform}^{\uparrow}$ packet records the finite uniform-convergence boundary where RealSeries partial sums are read as RealPowerSeries radial-window data. It consumes the Abel boundary routes of chapter 1922 and chapter 1763, the power-series carrier of chapter 1583, the series tail package of chapter 2155, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names uniform Abel convergence as displayed finite rows; it is not a quotient equality of series, a selected infinite sum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6123.1 (AbelUniform carrier)**.** *An* $\mathsf{AbelUniform}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, S, P, W, T, Q, E, H, C, L, N)$$

with the following displayed rows:

1. *R is the finite radius-window row, listing the dyadic subradius requests and boundary-approach windows inspected by the Abel consumer.*
2. *S is the* $\mathsf{RealSeries}^{\uparrow}$ *partial-sum row whose finite prefixes are being read inside those radius windows.*
3. *P is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *row exposing the same coefficient windows, radius data, and partial-sum reads as power-series data.*
4. *W is the* $\mathsf{UniformConvergenceSeries}^{\uparrow}$ *tail-budget row, assigning a common tail threshold to every displayed radius-window request in R.*
5. *T is the* $\mathsf{AbelSummation}^{\uparrow}$ *and* $\mathsf{AbelLimitTheorem}^{\uparrow}$ *boundary row that records the Abel radial handoff after R, S, P, W have been accepted.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the synchronized partial-sum and power-series windows.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row exposed only after the Abel boundary row and the regular rational readback have been displayed.*
8. *H records componentwise* hsame *transport for the radius, RealSeries, RealPowerSeries, uniform-tail, Abel-boundary, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from each Abel uniform request through R, S, P, W, T, Q, E.*
10. *L is the* Pkg *provenance over* $\mathsf{AbelUniform}^{\uparrow}$, $\mathsf{AbelLimitTheorem}^{\uparrow}$, $\mathsf{AbelSummation}^{\uparrow}$, $\mathsf{RealSeries}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{UniformConvergenceSeries}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{AbelUniform}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $R, S, P, W, T, Q, E, H, C, L, N$. It has no coordinate for selected sums, quotient series equality, countable choice, host power-series equality, ambient completeness, or a Real seal detached from the displayed Abel uniform route.

Lean4 ✓ `BEDC.Derived.AbelUniformUp.TasteGate.AbelUniformTasteGate single carrier alignment`

Theorem 6123.2 (AbelUniform NameCert obligations)**.** *For every accepted* $\mathsf{AbelUniform}^{\uparrow}$ *carrier*

$$U = (R, S, P, W, T, Q, E, H, C, L, N),$$

the local $\mathsf{NameCert}_{\mathsf{AbelUniform}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, Abel radius-window monotonicity through R, RealSeries and RealPowerSeries partial-sum alignment through S and P, uniform-tail exactness through W, Abel boundary handoff through T, RegSeqRat readback and Real seal nonescape through Q and E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through L, and local naming through N. No obligation row supplies selected sums, quotient series equality, countable choice, host analytic equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6123.1. Its mathematical rows are the radius-window row R, the $\mathsf{RealSeries}^{\uparrow}$ partial-sum row S, the $\mathsf{RealPowerSeries}^{\uparrow}$ row P, the $\mathsf{UniformConvergenceSeries}^{\uparrow}$ tail-budget row W, the Abel boundary row T, the $\mathsf{RegSeqRat}^{\uparrow}$ readback Q, and the $\mathsf{Real}^{\uparrow}$ seal E.

The radius-window monotonicity is local to R. A smaller displayed radius window is read by restricting the same dyadic boundary row, while S and P keep the RealSeries prefix and RealPowerSeries partial-sum read synchronized over that window.

Uniform-tail exactness is supplied by W. For each requested tolerance, W gives one tail threshold shared by every displayed radius-window row, so the Abel boundary read in T cannot choose a separate threshold for each point of the window.

The readback and seal rows are downstream. The row Q repacks the accepted synchronized windows as regular rational data, and E exposes the terminal Real seal only after R, S, P, W, T, Q have been read.

Finally H, C, L, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for selected sums, quotient series equality, countable choice, host analytic equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6123.3 (AbelUniform boundary consumer route)**.** *Every RealPowerSeries or RealSeries consumer of an accepted* $\mathsf{AbelUniform}^{\uparrow}$ *packet factors through*

$$\mathsf{RealSeries}^{\uparrow} \longmapsto \mathsf{RealPowerSeries}^{\uparrow} \longmapsto \mathsf{UniformConvergenceSeries}^{\uparrow} \longmapsto \mathsf{AbelSummation}^{\uparrow}/\mathsf{AbelLimitTheorem}^{\uparrow} \longmapsto \mathsf{Reg}$$

inside the same finite radius-window package. The consumer receives no selected sum, quotient equality of series, countable-choice schedule, host boundary-limit theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the RealSeries and RealPowerSeries rows S and P. They name the same finite partial-sum windows, first as series-prefix data and then as radius-window power-series data.

Apply the uniform-tail row W. It supplies the shared threshold discipline needed by $\mathsf{UniformConvergenceSeries}^{\uparrow}$, so a boundary consumer cannot vary the tail budget independently across the displayed radius window.

Now read the Abel boundary row T. The sibling $\mathsf{AbelSummation}^{\uparrow}$ and $\mathsf{AbelLimitTheorem}^{\uparrow}$ routes are consumed only after R, S, P, W have fixed the finite radial surface.

The route then passes through $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal $\mathsf{Real}^{\uparrow}$ seal E, with H, C, L, N preserving the same packet. Because the refused host and quotient objects are absent from the carrier, every boundary consumer remains inside the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{AbelUniform}^{\uparrow}$ boundary read exposes a radius-window row, RealSeries partial-sum row, RealPowerSeries row, UniformConvergenceSeries tail budget, Abel boundary row, RegSeqRat readback, RealUp seal, componentwise transport, displayed replay, provenance, and local NameCert before a Real-facing consumer can inspect the endpoint.

Independence witness: abel-limit-theorem, abel-summation, realpowerseries, uniform-convergence-series, regseqrat, real: none is bijective to $\mathsf{AbelUniform}^{\uparrow}$ because this packet jointly carries radius-window monotonicity, RealSeries and RealPowerSeries partial-sum alignment, one uniform tail budget, Abel boundary handoff, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{AbelUniform}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AbelUniform}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6123.1, Definition 6123.2, and Definition 6123.3 seed the finite uniform Abel boundary over radius windows, RealSeries partial sums, RealPowerSeries rows, UniformConvergenceSeries tail budgets, RegSeqRat readback, and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AbelUniform}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected sums, quotient equality of series, countable choice, host analytic equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-window monotonicity, partial-sum alignment, uniform-tail exactness, Abel boundary handoff, readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.AbelUniformUp.

Constructive story: An $\mathsf{AbelUniform}^{\uparrow}$ packet is generated from radius-window rows, RealSeries partial sums, RealPowerSeries rows, UniformConvergenceSeries tail budgets, AbelSummation and AbelLimitTheorem boundary rows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6124

A Concrete Naming Certificate for $\mathsf{WorldModelClassifier}^{\uparrow}$

The $\mathsf{WorldModelClassifier}^{\uparrow}$ packet realizes the classifier-certification branch of section 9462.16 as finite NameCert data. It sits beside the action-ledger surface of chapter 290, the statistical decision surface of chapter 231, the probability context of chapter 184, the measurable observation rows of chapter 185, and the certified-quality boundary of chapter 6117. The packet certifies scoped latent recovery, identifiability, transfer, and residue rows; it does not assert global representation truth, hidden causal sufficiency, unrestricted planning success, or model-quality completeness.

Definition 6124.1 (WorldModelClassifier carrier)**.** *A* $\mathsf{WorldModelClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, L, C, I, T, Q, E, H, R, P, N)$$

with the following displayed rows:

1. *S is the scoped source row, recording the observation carrier, admitted latent carrier, probability context, random-variable reads, pair protocol, transition family, and assumption ledger.*
2. *L is the latent-recovery row, exposing the recovered coordinate, invariant, state, or feature comparison used by the classifier.*
3. *C is the classifier row, naming the world-model relation read from L.*
4. *I is the identifiability row, recording equivalence mode, bound, gauge freedom, residual cases, and not-claimed latent domains.*
5. *T is the transfer row, routing the certified classifier to policy, hypothesis-test, or model-quality consumers.*
6. *Q is the* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *readback row for scoped quality claims.*
7. *E is the residue ledger, recording shortcut risk, finite-sample residue, distributional boundary, intervention failure, and planning-transfer boundary.*
8. *H records componentwise* hsame *transport for S, L, C, I, T, Q, E.*
9. *R records displayed* Cont *replay from source rows through recovery, classifier, identifiability, transfer, quality, and residue rows.*
10. *P is the* Pkg *provenance over* $\mathsf{WorldModelClassifier}^{\uparrow}$, $\mathsf{Policy}^{\uparrow}$, $\mathsf{HypothesisTest}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{CertifiedModelQuality}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{WorldModelClassifier}^{\uparrow}}$ *row.*

The carrier contains no coordinate for unscoped latent truth, host causal sufficiency, hidden distributional law, unrestricted policy optimality, or quality claims outside the displayed source and residue rows.

Theorem 6124.2 (WorldModelClassifier NameCert obligations)**.** *For every accepted* $\mathsf{WorldModelClassifier}^{\uparrow}$ *carrier* $W = (S, L, C, I, T, Q, E, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{WorldModelClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission; scoped source exposure through* $\mathsf{ProbSpace}^{\uparrow}$ *and* $\mathsf{RandomVar}^{\uparrow}$*; latent-recovery exposure; classifier-row exactness; identifiability ledger exposure; transfer routing to* $\mathsf{Policy}^{\uparrow}$*,* $\mathsf{HypothesisTest}^{\uparrow}$*, or* $\mathsf{CertifiedModelQuality}^{\uparrow}$*; residue nonescape; componentwise transport; displayed replay; provenance; and local naming.*

Proof. Carrier admission is projection from Definition 6124.1. The displayed packet lists the source, recovery, classifier, identifiability, transfer, quality, residue, and structural rows.

The source exposure is S. It names the probability and random-variable context before recovery or transfer can be read.

The classifier is exposed by L, C, I. Recovery produces the comparison surface, the classifier row names it, and the identifiability row records the equivalence mode, bound, gauge freedom, and residual cases.

The transfer and quality rows are T and Q. They allow policy, hypothesis-test, and certified-model-quality consumers to read only the scoped classifier route.

Finally E, H, R, P, N record residue, transport, replay, provenance, and local naming. Since the omitted objects are not carrier coordinates, the obligation surface is exhausted by the displayed rows. □

Theorem 6124.3 (WorldModelClassifier transfer and residue boundary)**.** *Let* $W = (S, L, C, I, T, Q, E, H, R, P, N)$ *be an accepted* $\mathsf{WorldModelClassifier}^{\uparrow}$ *carrier. Any* $\mathsf{Policy}^{\uparrow}$*,* $\mathsf{HypothesisTest}^{\uparrow}$*, or* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *consumer that reads the world-model classifier must traverse*

$$S, L, C, I, T, Q, E, H, R, P, N.$$

The route exports scoped latent recovery, identifiability, transfer, and residue ledgers, but it exports no global representation truth, unledgered distribution shift, unrestricted policy optimality, hidden causal sufficiency, or model-quality completeness.

Proof. Start from the consumer request. The only transfer row is T, and it is downstream of the source, recovery, classifier, and identifiability rows.

Read S, L, C, I first. These rows display the scoped observation and probability context, the recovered latent comparison, the classifier relation, and the identifiability ledger.

The quality readback Q and residue ledger E are part of the same finite packet. They force policy, hypothesis-test, and certified-quality consumers to preserve finite-sample, distributional, intervention, shortcut, and planning-boundary residue.

Transport and replay through H, R and provenance through P, N do not add hidden truth or optimality coordinates. Therefore every exported read is confined to the displayed route. □

Falsifiable prediction: A $\mathsf{WorldModelClassifier}^{\uparrow}$ packet is refuted whenever a policy, hypothesis-test, or certified-model-quality consumer reads classifier quality without displaying the scoped source row, latent-recovery row, identifiability row, transfer row, and residue ledger.

Independence witness: policy, hypothesis, probspace, randomvar, certified-model-quality: none is bijective to $\mathsf{WorldModelClassifier}^{\uparrow}$ because this packet binds scoped latent recovery, classifier

exactness, identifiability, transfer, quality readback, and residue ledger rows in one carrier, while each sibling supplies only action selection, statistical decision, probability context, observation measurability, or quality certification.

Lean follow-up must provide `Nontrivial WorldModelClassifierUp` and `FieldFaithful WorldModelClassif` in `TasteGate.lean` before the carrier can be treated as a TasteGate field-faithful object.

Closure status for $\mathsf{NameCert}_{\mathsf{WorldModelClassifier}^{\uparrow}}$.

Constructive story: A $\mathsf{WorldModelClassifier}^{\uparrow}$ packet is generated from a scoped source row, latent-recovery row, classifier row, identifiability ledger, transfer row, certified-quality readback, residue ledger, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WorldModelClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6124.1, Definition 6124.2, and Definition 6124.3 seed the finite classifier route for scoped latent recovery, identifiability, transfer, quality readback, and residue ledgers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WorldModelClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global representation truth, hidden causal sufficiency, unrestricted policy optimality, unledgered distribution shift, model-quality completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked source exposure, latent-recovery exposure, classifier exactness, identifiability ledger, transfer routing, quality readback, residue nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WorldModelClassifierUp.

Chapter 6125

A Concrete Naming Certificate for $\mathsf{FiniteSubcoverLedger}^{\uparrow}$

The $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ packet records the finite subcover handoff used by the compact-uniform-continuity route of chapter 9237. It sits over a $\mathsf{CompactMetric}^{\uparrow}$ source row, a $\mathsf{TotallyBounded}^{\uparrow}$ finite-net schedule, the dyadic radius rows of chapter 291, and the downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer, packaging the finite ProbeBundle cover-selection route as NameCert data rather than as an open-cover compactness theorem.

Definition 6125.1 (Finite subcover ledger carrier)**.** *A* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, T, B, D, S, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose metric windows and compact source provenance are visible to the cover request.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedule naming the centers and mesh tolerance used to inspect the cover.*
3. *B is the finite* ProbeBundle *cover spine whose entries are the only cover candidates admitted by the ledger.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius and comparison ledger attached to the same finite cover spine.*
5. *S is the selected finite subcover row, displayed as a sublist of B with its inherited radius data.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that consumes the finite subcover ledger for compact-continuity arguments.*
7. *H stores componentwise* hsame *transport for the compact source, net schedule, cover spine, dyadic ledger, subcover row, uniform-modulus handoff, replay, provenance, and local naming rows.*
8. *C stores displayed* Cont *replay from a compact-uniform request through the finite cover spine and into the uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteSubcoverLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, selected infinite subcovers, host compactness, quotient point equality, countable choice, or ambient completeness.

Lean4 (definition) `BEDC.Derived.FiniteSubcoverLedgerUp.FiniteSubcoverLedgerUp`
Lean4 ✓ `BEDC.Derived.FiniteSubcoverLedgerUp.FiniteSubcoverLedgerTasteGate single carrier alignment`

Theorem 6125.2 (Finite subcover ledger obligations)**.** *For every accepted* $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ *carrier*

$$L = (K, T, B, D, S, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteSubcoverLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the ten displayed rows, compact-source exposure through* K*, finite-net exposure through* T*, finite* ProbeBundle *cover admission through* B*, dyadic radius accounting through* D*, finite subcover display through* S*,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through* U*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, local naming through* N*, and non-escape from host open-cover compactness.*

Proof. Project the carrier of Definition 6125.1. The only cover candidates available to the packet are the finite entries of B, and the only source coverage data are the compact source K, finite-net schedule T, and dyadic radius ledger D.

Read the selected subcover row S as a displayed finite sublist of the same ProbeBundle spine. Because S inherits its radii from D and its source coverage from K, T, the ledger never invokes an arbitrary open cover or a selected infinite subcover.

Finally route the ledger into U and the structural rows H, C, P, N. The $\mathsf{UniformModulus}^{\uparrow}$ consumer receives exactly the finite subcover handoff, while host compactness, quotient point equality, countable choice, and ambient completeness remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ read must expose the $\mathsf{CompactMetric}^{\uparrow}$ source, the $\mathsf{TotallyBounded}^{\uparrow}$ finite-net schedule, the finite ProbeBundle cover spine, the $\mathsf{DyadicRatCore}^{\uparrow}$ radius ledger, the selected finite subcover row, the $\mathsf{UniformModulus}^{\uparrow}$ handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: compactmetric, totallybounded, dyadicratcore, uniformmodulus: none is bijective with $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ because this packet jointly carries the compact source, finite-net schedule, finite cover spine, selected subcover row, radius ledger, uniform-modulus handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteSubcoverLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteSubcoverLedger}^{\uparrow}$ packet is generated from a CompactMetric source row, a TotallyBounded finite-net schedule, a finite ProbeBundle cover spine, a DyadicRatCore radius ledger, a displayed finite subcover row, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteSubcoverLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6125.1 and Definition 6125.2 seed the finite subcover route through $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteSubcoverLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected infinite subcovers, host compactness, quotient point equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, finite-net exposure, cover-spine admission, dyadic radius accounting, selected subcover display, uniform-modulus handoff, transport, replay, provenance, local naming, and non-escape for BEDC.Derived.FiniteSubcoverLedgerUp.

Chapter 6126

A Concrete Naming Certificate for $\mathsf{RealSine}^{\uparrow}$

Following Bishop and Bridges on constructive power series, the $\mathsf{RealSine}^{\uparrow}$ packet isolates the sine series as its own finite NameCert route instead of leaving it as a local trigonometric block inside chapter 74. The packet reads the alternating odd-power coefficient ledger through $\mathsf{RealPowerSeries}^{\uparrow}$, the bounded-window convergence route through $\mathsf{UniformCauchySeries}^{\uparrow}$, the product and addition handoff through $\mathsf{CauchyProduct}^{\uparrow}$, the endpoint comparison through $\mathsf{RealEquality}^{\uparrow}$, and the terminal seal through $\mathsf{Real}^{\uparrow}$. It shares the factorial-budget discipline of chapter 1816, but its parity ledger is the odd alternating sine ledger rather than the exponential $1/n!$ ledger.

Definition 6126.1 (RealSine carrier)**.** *A* $\mathsf{RealSine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, F, B, U, M, K, P, E, Q, H, C, L, N)$$

with the following displayed rows:

1. *A is the alternating odd-coefficient ledger fixing the sine series by $a_{2k} = 0$ and $a_{2k+1} = (-1)^k/(2k+1)!$ on each inspected unary window.*
2. *F is the factorial and reciprocal dyadic budget row for the odd denominators, signs, reciprocal enclosures, and denominator-growth comparisons used by A.*
3. *B is the bounded real-window row on which the sine evaluation request is accepted.*
4. *U is the* $\mathsf{UniformCauchySeries}^{\uparrow}$ *row proving uniform Cauchy convergence of the displayed odd alternating partial sums on B.*
5. *M is the majorant tail-budget row, using the alternating-series and factorial-growth estimates to bound every endpoint-facing tail without selecting a host infinite sum.*
6. *K is the* $\mathsf{CauchyProduct}^{\uparrow}$ *route used by sine addition, odd-parity product splitting, and compatibility with the cosine-side sibling route in chapter 74.*
7. *P is the* $\mathsf{RealPowerSeries}^{\uparrow}$ *evaluation row exposing coefficient windows, argument powers, partial sums, and endpoint request as power-series data.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the coefficient ledger, dyadic budget, uniform Cauchy route, tail budget, product route, and power-series row have been displayed.*
9. *Q is the* $\mathsf{RealEquality}^{\uparrow}$ *classifier row for comparing accepted sine readbacks through the displayed endpoint boundary.*

10. *H is componentwise* hsame *transport for $A, F, B, U, M, K, P, E, Q, C, L, N$.*

11. *C is the displayed* Cont *replay from coefficient and factorial windows to partial sums, from partial sums to uniform Cauchy control, from the uniform route to the power-series endpoint, and from the endpoint to the Real and equality consumers.*

12. *L is the* Pkg *provenance over* $\mathsf{RealSine}^{\uparrow}$, $\mathsf{RealPowerSeries}^{\uparrow}$, $\mathsf{UniformCauchySeries}^{\uparrow}$, $\mathsf{CauchyProduct}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealSine}^{\uparrow}}$ *row.*

The classifier compares accepted sine packets only through $A, F, B, U, M, K, P, E, Q, H, C, L, N$. It has no coordinate for a host sine function, analytic continuation, quotient real equality, selected infinite sum, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealSineUp.RealSineTasteGate single carrier alignment`

Theorem 6126.2 (RealSine NameCert obligations)**.** *For every accepted* $\mathsf{RealSine}^{\uparrow}$ *carrier $S = (A, F, B, U, M, K, P, E, Q, H, C, L, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealSine}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the alternating odd-coefficient ledger, factorial and reciprocal dyadic budget, bounded real window, uniform Cauchy modulus, majorant tail budget, Cauchy-product route, RealPowerSeries evaluation row, Real seal, RealEquality classifier, transport, continuation, provenance, and local naming rows;*

2. *coefficient exactness, requiring every inspected coefficient to be read from A together with the odd factorial and sign budget in F;*

3. *uniform Cauchy convergence through U and M, requiring every endpoint-facing tail to pass through the bounded-window and majorant rows before P or E is consumed;*

4. *power-series evaluation through P, requiring the endpoint request to be a* $\mathsf{RealPowerSeries}^{\uparrow}$ *read rather than a host trigonometric primitive;*

5. *endpoint classifier stability through Q, H, C, requiring accepted comparisons to preserve the displayed* $\mathsf{RealEquality}^{\uparrow}$ *boundary under componentwise transport and replay;*

6. *ledger nonescape through L, N, requiring the local packet to export no host* sin*, analytic-continuation datum, quotient real equality, selected infinite sum, countable-choice selector, or ambient completeness witness.*

Proof. Carrier admission is projection from Definition 6126.1. The accepted packet displays exactly the rows $A, F, B, U, M, K, P, E, Q, H, C, L, N$, so every public sine read has a visible coefficient, convergence, endpoint, classifier, and structural source.

Coefficient exactness is the joint read of A and F. The ledger A fixes the even coefficients as zero and the odd coefficients as alternating reciprocal factorials, while F supplies the inspected factorial and dyadic reciprocal budgets that make those coefficients finite BEDC data.

Uniform Cauchy convergence is read through B, U, M. The bounded window B fixes the evaluation scope; U supplies the uniform Cauchy route; and M supplies the displayed tail budget before any endpoint seal may be consumed.

The endpoint and classifier rows are P, E, Q. The row P places the read inside $\mathsf{RealPowerSeries}^{\uparrow}$, E is the only $\mathsf{Real}^{\uparrow}$ seal, and Q is the only endpoint classifier boundary. The transport and replay rows H, C preserve these rows componentwise.

Finally L and N record provenance and local naming for the same finite packet. Since no listed row contains a host trigonometric function, selected limit, quotient equality, countable-choice selector, or ambient completeness principle, none of those objects is exported by the local $\mathsf{NameCert}_{\mathsf{RealSine}^\uparrow}$ surface. □

Theorem 6126.3 (RealSine power-series route)**.** *Every public read of an accepted* $\mathsf{RealSine}^\uparrow$ *packet factors through*

$$\mathsf{RealAnalytic}^\uparrow,\quad \mathsf{RealPowerSeries}^\uparrow,\quad \mathsf{UniformCauchySeries}^\uparrow,\quad \mathsf{CauchyProduct}^\uparrow,\quad \mathsf{RealEquality}^\uparrow,\quad \mathsf{Real}^\uparrow$$

before any endpoint is consumed. Consequently the public route names sine by finite alternating coefficient data, bounded-window Cauchy control, Cauchy-product handoff, endpoint equality, and Real seal data, not by a host trigonometric primitive.

Proof. Begin with a public sine read. The source theorem cluster in chapter 74 supplies the local sine partial-sum and trigonometric Cauchy-modulus material, while the standalone carrier of Definition 6126.1 repacks the relevant rows as $A, F, B, U, M, K, P, E, Q, H, C, L, N$.

The coefficient and factorial reads A, F enter $\mathsf{RealPowerSeries}^\uparrow$ through P. The bounded real window B fixes the evaluation scope, and the partial sums remain power-series data until the uniform route is accepted.

Uniform convergence is supplied by U and M. These rows place the odd alternating partial sums inside $\mathsf{UniformCauchySeries}^\uparrow$ and expose the majorant tail budget before the endpoint seal is available.

Law-facing and comparison-facing reads then pass through K, Q, E. The row K supplies the $\mathsf{CauchyProduct}^\uparrow$ handoff used by the sine addition route, Q supplies the $\mathsf{RealEquality}^\uparrow$ endpoint classifier, and E is the terminal $\mathsf{Real}^\uparrow$ seal.

The structural rows H, C, L, N preserve the same route as transport, replay, provenance, and local naming. Thus every public endpoint read factors through the displayed analytic, power-series, uniform-Cauchy, product, equality, and Real rows before the sine value is consumed. □

Falsifiable prediction: Every $\mathsf{RealSine}^\uparrow$ public read factors through an alternating odd-coefficient ledger, factorial and reciprocal dyadic budgets, a bounded real-window row, a uniform Cauchy-series modulus, a majorant tail budget, a Cauchy-product route, a RealPowerSeries evaluation row, a RealEquality classifier row, a RealUp seal, transport, continuation, provenance, and local naming.

Independence witness: real-analytic, real-exponential, realpowerseries, uniform-cauchy-series, cauchyproduct, real-equality, real: $\mathsf{RealSine}^\uparrow$ is not bijective to any listed sibling because it jointly carries the odd alternating sine coefficient ledger, factorial reciprocal budget, bounded-window uniform convergence route, Cauchy-product trigonometric handoff, endpoint equality classifier, Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealSine}^\uparrow}$**.**

Constructive story: A $\mathsf{RealSine}^\uparrow$ packet is generated from the alternating odd-coefficient ledger $a_{2k} = 0$ and $a_{2k+1} = (-1)^k/(2k+1)!$, factorial and reciprocal dyadic budgets, a bounded real-window row, a uniform Cauchy-series modulus, a majorant tail budget, a Cauchy-product route, a RealPowerSeries evaluation row, a RealEquality classifier row, a RealUp seal, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealSine}^\uparrow}$, seed)

Scope closed: Definition 6126.1, Definition 6126.2, and Definition 6126.3 seed the finite sine power-series route through existing BEDC analytic, power-series, uniform-Cauchy, product, equality, and Real infrastructure.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealSine}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host trigonometric functions, analytic continuation, quotient real equality, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coefficient exactness, uniform Cauchy convergence, power-series evaluation, endpoint classifier stability, ledger nonescape, and TasteGate field faithfulness for BEDC.Derived.RealSineUp.

Chapter 6127

A Concrete Naming Certificate for $\mathsf{FiniteMeasure}^{\uparrow}$

Following Halmos and Kolmogorov finite measure theory, the $\mathsf{FiniteMeasure}^{\uparrow}$ packet names the finite additive event-value ledger that measure-facing chapters use before invoking countable or completed measure structure. It sits after the event-family rows of chapter 5186, the finite tally route of chapter 5996, the finite-cell approximation surface of chapter 5884, the root event-value packet of chapter 92, and the scalar readback of chapter 38. The packet records only finite event values and their displayed finite additivity route.

Definition 6127.1 (Finite measure carrier)**.** *A* $\mathsf{FiniteMeasure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, E, V, A, T, R, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SigmaAlgebra}^{\uparrow}$ *event-family row, exposing the admitted measurable events and their displayed classifier.*
2. *E is the finite event ledger, listing the particular event rows whose values are visible to the packet.*
3. *V is the finite event-value ledger, assigning to every event in E a displayed non-negative* $\mathsf{Real}^{\uparrow}$ *value.*
4. *A is the finite disjoint-additivity ledger, recording the binary and finite-family disjoint-union checks admitted by the packet.*
5. *T is the total-mass bound row, naming a displayed* $\mathsf{Real}^{\uparrow}$ *bound for the whole admitted event family.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *scalar readback row, routing event values, sums, and the total-mass bound through the same scalar certificate surface.*
7. *H records componentwise* hsame *transport for the event-family, finite event ledger, value ledger, additivity ledger, total-mass bound, scalar readback, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from an event-value request through S, E, V, A, T, R and then to the structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteMeasure}^{\uparrow}$, $\mathsf{SigmaAlgebra}^{\uparrow}$, $\mathsf{CountingMeasure}^{\uparrow}$, $\mathsf{SimpleFunctionApproximation}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteMeasure}^\uparrow}$ *row.*

The carrier has no coordinate for sigma-additive completion, host measure objects, quotient event equality, countable choice, or ambient $\mathsf{Real}^\uparrow$ *completeness.*

Theorem 6127.2 (FiniteMeasure NameCert obligations)**.** *For every accepted* $\mathsf{FiniteMeasure}^\uparrow$ *carrier* $F = (S, E, V, A, T, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteMeasure}^\uparrow}$ *surface consists exactly of the sigma-algebra event-family row, finite event ledger, finite event-value ledger, finite disjoint-additivity ledger, total-mass bound,* $\mathsf{Real}^\uparrow$ *scalar readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream* $\mathsf{Measure}^\uparrow$*,* $\mathsf{RadonMeasure}^\uparrow$*,* $\mathsf{Egorov}^\uparrow$*,* $\mathsf{DominatedConvergence}^\uparrow$*,* $\mathsf{CountingMeasure}^\uparrow$*, or* $\mathsf{SimpleFunctionApproximation}^\uparrow$ *consumer must expose those rows before treating the packet as a finite measure substrate.*

Lean4 ✓ `BEDC.Derived.FiniteMeasureUp.FiniteMeasureCarrier namecert obligations`

Proof. Project the carrier of Definition 6127.1. The row S is the only source of measurable-event admission, while E restricts public values to the displayed finite event ledger.

Read the value and additivity rows. The ledger V supplies the visible event values, A records exactly the finite disjoint-union checks available to consumers, and T gives the displayed total-mass bound for the same event family.

The row R routes every value, finite sum, and bound through the $\mathsf{Real}^\uparrow$ scalar readback surface. Transport and replay are componentwise through H and C, so no second event equality or hidden scalar completion is introduced.

Finally P and N source and name this packet. Since the carrier contains no sigma-additive completion, host measure object, quotient event equality, countable-choice selector, or ambient Real-completeness coordinate, the listed rows exhaust the local NameCert obligation surface. □

Theorem 6127.3 (FiniteMeasure counting-measure handoff)**.** *Every accepted* $\mathsf{CountingMeasure}^\uparrow$ *packet whose finite support and event rows are read as finite event values can hand off to* $\mathsf{FiniteMeasure}^\uparrow$ *only through the displayed rows* $S, E, V, A, T, R, H, C, P, N$*. The handoff preserves the finite tally as a* $\mathsf{Real}^\uparrow$ *value row, exposes finite disjoint additivity through the ledger* A*, and exports no sigma-additive completion, host measure object, quotient event equality, countable-choice selector, or ambient* $\mathsf{Real}^\uparrow$ *completeness claim.*

Proof. Start with the finite support and tally rows supplied by $\mathsf{CountingMeasure}^\uparrow$. They determine event values only for the displayed finite event rows, so the target $\mathsf{FiniteMeasure}^\uparrow$ packet must first expose S and E.

The tally readback becomes a value row through V and R. Finite disjoint unions are then routed through A, while the total mass is bounded by T over the same event-family row.

Transport, replay, provenance, and naming are carried by H, C, P, N. These rows preserve the counting route as finite event-value data and do not add countable unions, host measures, quotient event representatives, choice-selected supports, or ambient Real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteMeasure}^\uparrow}$.

Constructive story: A $\mathsf{FiniteMeasure}^\uparrow$ packet is generated from a SigmaAlgebraUp event-family row, finite event ledger, finite event-value ledger, finite disjoint-additivity ledger, total-mass bound, RealUp scalar readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteMeasure}^\uparrow}$, seed)

Scope closed: Definition 6127.1, Definition 6127.2, and Definition 6127.3 seed the finite additive event-value substrate consumed by MeasureUp, RadonMeasureUp, EgorovUp, DominatedConvergenceUp, CountingMeasureUp, and SimpleFunctionApproximationUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteMeasure}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close sigma-additive completion, host measure objects, quotient event equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked event-family admission, finite event-value exactness, finite disjoint-additivity, total-mass bound readback, Real scalar transport, replay, provenance, and local naming for BEDC.Derived.FiniteMeasureUp.FiniteMe

Chapter 6128

A Concrete Naming Certificate for $\mathsf{LegendrePolynomial}^{\uparrow}$

Following Legendre's polynomial family and Szego's orthogonal-polynomial presentation, the $\mathsf{LegendrePolynomial}^{\uparrow}$ packet names the finite recurrence, normalization, and interval-weight rows for the displayed family P_n without importing a host function space. It sits after the polynomial carrier of chapter 50, the orthogonal-family surface of chapter 2264, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic endpoint and coefficient ledger of chapter 291, and the adjacent recurrence model of chapter 3480.

Definition 6128.1 (Legendre polynomial carrier)**.** *A* $\mathsf{LegendrePolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (d, P_0, P_1, R, N, O, W, Q, E, H, C, \Pi, A)$$

with the following displayed rows:

1. *d is the unary degree-window row.*
2. *P_0 and P_1 are the displayed seed polynomial rows for 1 and x inside* $\mathsf{Poly}^{\uparrow}$*.*
3. *R is the finite recurrence ledger for $(n+1)P_{n+1} = (2n+1)xP_n - nP_{n-1}$, including the dyadic coefficient rows needed for each inspected step.*
4. *N is the normalization row recording $P_n(1) = 1$ on the displayed window.*
5. *O is the* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *handoff row for the interval-weight and finite moment ledger over the real interval endpoints.*
6. *W is the finite evaluation-window row.*
7. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for inspected evaluations and coefficients.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the endpoint and evaluation reads.*
9. *H records componentwise* hsame *transport for the displayed rows.*
10. *C records displayed* Cont *replay of recurrence, normalization, orthogonality, and readback requests.*
11. *Π is the* Pkg *provenance over* $\mathsf{LegendrePolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{OrthogonalPolynomial}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ChebyshevPolynomial}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
12. *A is the local* $\mathsf{NameCert}_{\mathsf{LegendrePolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for host function-space equality, analytic completeness, selected infinite expansions, classical completeness, quotient real equality, or arbitrary choice of an orthogonal basis.

Lean4 (definition) `BEDC.Derived.LegendrePolynomialUp.LegendrePolynomialUp`

Theorem 6128.2 (Legendre polynomial NameCert obligations)**.** *For every accepted* $\mathsf{LegendrePolynomial}^{\uparrow}$ *carrier* $L = (d, P_0, P_1, R, N, O, W, Q, E, H, C, \Pi, A)$*, the local* $\mathsf{NameCert}_{\mathsf{LegendrePolynomial}^{\uparrow}}$ *surface consists exactly of carrier admission for the degree window, seed polynomials, finite recurrence ledger, normalization row, orthogonal-polynomial handoff, evaluation window, regular rational readback, Real seal, componentwise transport, displayed replay, package provenance, and local naming. The certificate exports no host function-space equality, analytic completeness, selected infinite expansion, classical completeness, or ambient basis theorem.*

Proof. Project the carrier of Definition 6128.1. The rows d, P_0, P_1, R, N display the finite Legendre recurrence and its normalization over the inspected degree window.

Read the orthogonal-family row O next. It is a handoff to $\mathsf{OrthogonalPolynomial}^{\uparrow}$ over the real interval and finite moment ledger, so the packet names only the displayed orthogonality surface rather than a host function space.

The consumer-facing rows W, Q, E then provide finite evaluation windows, regular rational readback, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The structural rows H, C, Π, A transport, replay, source, and name the same packet, and the excluded analytic and choice principles have no coordinate in the carrier. □

Theorem 6128.3 (Legendre polynomial orthogonal-family route)**.** *Every accepted* $\mathsf{LegendrePolynomial}^{\uparrow}$ *orthogonality read factors through*

$$d, P_0, P_1, R, N \longmapsto O \longmapsto W, Q, E \longmapsto H, C, \Pi, A.$$

Thus the Legendre packet reaches $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *through a finite recurrence and normalization ledger before any downstream approximation or spectral-basis consumer may read the orthogonal-family row. The route may cite the neighboring Chebyshev recurrence surface as a sibling recurrence anchor, but it does not identify the two families or import selected infinite expansions, host function-space equality, analytic completeness, or classical completeness.*

Proof. Begin with the finite recurrence rows d, P_0, P_1, R. These rows display the degree window, the two seeds, and each inspected recurrence step using $\mathsf{Poly}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ coefficient data.

The normalization row N and the orthogonal-family row O then form the route into $\mathsf{OrthogonalPolynomial}^{\uparrow}$. This is the only row that a downstream orthogonal-polynomial consumer may read, and it is still tied to the displayed recurrence ledger.

The rows W, Q, E provide evaluation windows, regular rational readback, and the $\mathsf{Real}^{\uparrow}$ seal. Transport, replay, provenance, and local naming through H, C, Π, A preserve the same route without adding a host function space, an infinite expansion selector, analytic completeness, or classical completeness. □

Falsifiable prediction: Every $\mathsf{LegendrePolynomial}^{\uparrow}$ read factors through one degree window, two seed polynomial rows, one finite recurrence ledger, one normalization row, one OrthogonalPolynomialUp handoff, finite evaluation windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, orthogonal-polynomial, real, regseqrat, dyadicratcore, chebyshev-polynomial: $\mathsf{LegendrePolynomial}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the Legendre seeds, recurrence coefficients, normalization row, orthogonal-family handoff, finite evaluation windows, regular readback, Real seal, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{LegendrePolynomial}^{\uparrow}}$.

Constructive story: A $\mathsf{LegendrePolynomial}^{\uparrow}$ packet is generated from a unary degree window, seed polynomial rows, a finite Legendre recurrence ledger, normalization at the real unit endpoint, an OrthogonalPolynomial handoff row, finite evaluation windows, RegSeqRat readback, Real seals, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LegendrePolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6128.1, Definition 6128.2, and Definition 6128.3 seed the finite Legendre recurrence certificate and its orthogonal-family route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LegendrePolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space equality, analytic completeness, selected infinite expansions, classical completeness, quotient real equality, Lean verification, or bridge checking for $\mathsf{LegendrePolynomial}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, recurrence ledger exactness, normalization exactness, orthogonal-family handoff, finite evaluation windows, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LegendrePolynomialUp.

Chapter 6129

A Concrete Naming Certificate for $\mathsf{CauchyFilterSpace}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley filter-space practice, the $\mathsf{CauchyFilterSpace}^{\uparrow}$ packet records the finite NameCert surface of a Cauchy filter space before a regularization or completion consumer reads it. It sits over the Cauchy filter carrier of chapter 333, next to the regularized completion-facing surface of chapter 4102, and routes finite observations through the stream windows of chapter 126, the regular rational readback of chapter 301, and the real-name seal of chapter 38. The packet is a human-known completion-facing object: it names the displayed filter-space rows that a completion route may inspect without adding quotient filters, selected limits, countable choice, separated completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6129.1 (Cauchy filter space carrier)**.** *A* $\mathsf{CauchyFilterSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, B, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the point and basic-neighborhood carrier row, retaining classifier-distinct point and neighborhood names without quotienting them by convergence or host equality.*
2. *B is the* $\mathsf{CauchyFilter}^{\uparrow}$ *basis row, exposing finite filter-membership requests, directed refinements, and Cauchy pressure witnesses available to the packet.*
3. *Q is the filter-space exactness row, tying each accepted Cauchy request to the same point classifier and the same displayed filter-basis rows in X and B.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *inspection row for finite observation windows requested by completion-facing consumers.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and Cauchy pressure rows, before any real seal is public.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *sealing row reached only after S, R and the filter-space exactness row Q have been displayed.*
7. *H records componentwise* hsame *transport for all carrier, basis, exactness, window, readback, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay through the same finite route.*

9. *P is the* Pkg *provenance over* $\mathsf{CauchyFilterSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyFilterSpace}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient filter equality, a selected representative filter, a selected limit point, countable choice, separated completion, or host equality of point names.

Lean4 ✓ `BEDC.Derived.CauchyFilterSpaceUp.CauchyFilterSpaceTasteGate single carrier alignment`

Theorem 6129.2 (Cauchy filter space NameCert obligations)**.** *For every accepted* $\mathsf{CauchyFilterSpace}^{\uparrow}$ *carrier*

$$F = (X, B, Q, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyFilterSpace}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the point/neighborhood carrier X, Cauchy filter basis B, filter-space exactness row Q, stream windows S, regular rational readback R, real seal E, transport H, replay C, provenance P, and local naming N;*
2. *filter-space exactness, requiring every accepted read to use the displayed point classifier and Cauchy filter rows of X, B, Q;*
3. *window and readback stability, requiring finite stream observations, rational readback, and real sealing to move only componentwise through H and to replay through C;*
4. *ledger nonescape, excluding quotient filters, host equality, selected representatives, selected limits, countable choice, separated completion, and ambient real completeness from the packet;*
5. *completion-facing handoff, exporting only $X, B, Q, S, R, E, H, C, P, N$ to regular Cauchy filter space or completion consumers before those consumers add their own regularity or completion rows.*

No other NameCert obligation is available from the Cauchy filter space packet.

Proof. Carrier admission is the projection of Definition 6129.1. The tuple lists the point/neighborhood carrier, Cauchy filter basis, exactness row, finite stream windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

Filter-space exactness is carried by Q. A consumer first fixes the point classifier in X and the Cauchy filter basis in B, then reads finite requests through Q, S before asking for rational or real data.

Window and readback stability are componentwise. The row H transports the displayed coordinates, and the row C replays the same route for consumers, so Cauchy pressure and readback cannot be replaced by host equality or by a detached separated-completion fact.

Ledger nonescape follows from the carrier inventory. There is no coordinate for quotient filter equality, selected representatives, selected limits, countable choice, separated completion, or ambient real completeness.

Finally the completion-facing handoff is only the displayed tuple $X, B, Q, S, R, E, H, C, P, N$. A $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$ or completion consumer may use that finite filter-space surface as input, but any regularity witness, completion object, quotient, or limit belongs to the consumer packet rather than to this carrier. □

Theorem 6129.3 (Cauchy filter space completion handoff)**.** *Let $F = (X, B, Q, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{CauchyFilterSpace}^{\uparrow}$ *carrier. Every* $\mathsf{RegularCauchyFilterSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$

consumer route exported by F factors through the displayed Cauchy filter basis, filter-space exactness row, stream-window inspection row, rational readback row, real-seal row, componentwise transport, replay, provenance, and local naming rows. The route exports no quotient filter, selected limit, countable-choice selector, separated completion, host point equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Proof. Start with any completion-facing read of F. By Definition 6129.1, the only source rows available to that read are the displayed point/neighborhood carrier X, Cauchy filter basis B, and exactness row Q.

The read then passes through the finite observation interface. The stream row S exposes the requested windows, the rational row R supplies the regular readback attached to those windows, and the seal row E is reached only after the same displayed route has been replayed.

The structural rows do not add new completion data. Transport is componentwise through H, replay is through C, provenance is through P, and local naming is through N.

Thus a downstream regular Cauchy filter space or completion packet can consume the finite boundary of F, but it cannot extract a quotient filter, selected limit, countable-choice selector, separated completion, host point equality, or ambient real-completeness witness from this carrier. □

Falsifiable prediction: A consumer that claims a $\mathsf{CauchyFilterSpace}^{\uparrow}$ handoff while omitting the Cauchy filter basis, filter-space exactness row, $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 6129.1.

Independence witness: cauchyfilter, regular-cauchy-filter-space, streamname, regseqrat, real: $\mathsf{CauchyFilterSpace}^{\uparrow}$ binds the point/filter carrier, Cauchy filter basis, finite request exactness, stream windows, rational readback, real sealing, transport, replay, provenance, and local naming rows in one completion-facing packet; each named sibling lacks at least one of these rows as a local filter-space coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyFilterSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyFilterSpace}^{\uparrow}$ packet is generated from a point/neighborhood carrier, CauchyFilter basis row, filter-space exactness row, StreamName finite windows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyFilterSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6129.1, Definition 6129.2, and Definition 6129.3 seed the finite Cauchy-filter-space surface that can feed regular Cauchy filter space and completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyFilterSpace}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CauchyFilterSpaceUp`

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected representatives, selected limits, countable choice, separated completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, filter-space exactness, window and readback stability, ledger nonescape, and completion-facing handoff for BEDC.Derived.CauchyFilterSpaceUp.

Chapter 6130

A Concrete Naming Certificate for $\mathsf{CauchyCompletionStrength}^{\uparrow}$

Following Bishop and Bridges on constructive completion and finite products, the $\mathsf{CauchyCompletionStrength}^{\uparrow}$ packet records the finite BEDC surface by which regular Cauchy completion data commute with a displayed product route. It sits over the dyadic tolerance core of chapter 291, the stream-name windows of chapter 126, the regular rational readback of chapter 301, the completion functor of chapter 423, and the terminal real-name seal of chapter 38. The packet names product-transport strength only as finite NameCert data; it does not assert quotient completion, selected limits, countable choice, host product equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6130.1 (Cauchy completion strength carrier)**.** *A* $\mathsf{CauchyCompletionStrength}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, B, W_A, W_B, D_A, D_B, P, Q, R, E, H, C, G, N)$$

with the following displayed rows:

1. *A and B are the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose regular Cauchy data are to be paired.*
2. *W_A and W_B are the corresponding* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows, displaying the observations used by each source row.*
3. *D_A and D_B are the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers attached to the two window rows.*
4. *P is the ?? pairing-route row, recording only the displayed product read of the two regular Cauchy sources.*
5. *Q is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff row, applied after the product route has displayed A, B, W_A, W_B, D_A, D_B.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *completion seal reached only after the paired regular readback and the completion-functor handoff are public.*
7. *E records strength exactness: every accepted product-completion read must use the displayed rows $A, B, W_A, W_B, D_A, D_B, P, Q, R$ in that order.*
8. *H records componentwise* hsame *transport for the two sources, two stream-window rows, two dyadic ledgers, product route, completion handoff, seal, exactness, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the same product-completion route.*
10. *G is the* Pkg *provenance over* $\mathsf{CauchyCompletionStrength}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *??*, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionStrength}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, W_A, W_B, D_A, D_B, P, Q, R, E, H, C, G, N$. It has no coordinate for quotient completion, selected limits, countable choice, host product equality, detached product representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionStrengthUp.CauchyCompletionStrengthTasteGate single carrier alignment`

Theorem 6130.2 (Cauchy completion strength NameCert obligations)**.** *For every accepted* CauchyCompletionStren *carrier*

$$S = (A, B, W_A, W_B, D_A, D_B, P, Q, R, E, H, C, G, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyCompletionStrength}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the two* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows, two* $\mathsf{StreamName}^{\uparrow}$ *window rows, two* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledgers, ?? pairing route,* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *handoff,* $\mathsf{Real}^{\uparrow}$ *seal, exactness, transport, replay, provenance, and local naming rows;*
2. *source-window compatibility, requiring W_A, D_A to read only from A and W_B, D_B to read only from B;*
3. *product-route stability, requiring P to pair the two displayed regular Cauchy sources before the completion handoff Q is consumed;*
4. *completion-seal nonescape, requiring R to be downstream of $A, B, W_A, W_B, D_A, D_B, P, Q$ and excluding quotient completions, selected limits, countable choice, host product equality, and ambient real completeness;*
5. *structural persistence through* hsame, Cont, Pkg, *and the local* NameCert *row N.*

No other NameCert obligation is available from the Cauchy completion strength packet.

Proof. Project the carrier of Definition 6130.1. Its analytic rows are exactly the two regular Cauchy sources, their finite stream windows, their dyadic ledgers, the product pairing route, the completion-functor handoff, and the terminal real seal.

The five obligation rows separate the finite data by use. Carrier admission displays every coordinate. Source-window compatibility ties each window and tolerance ledger to its own regular Cauchy source, so the two sources cannot be mixed by an ambient product equality.

Product-route stability then forces the pairing row P to be read before Q. The completion-seal row R is downstream of that displayed route, so it cannot stand for quotient completion, a selected limit, or a countable-choice schedule.

Finally H, C, G, N preserve transport, replay, provenance, and local naming for the same finite route. Since the carrier contains no further coordinate, the five rows exhaust the local NameCert surface. □

Theorem 6130.3 (Cauchy completion strength product route)**.** *Every product-completion read of an accepted* $\mathsf{CauchyCompletionStrength}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ ??,\ \mathsf{CauchyCompletionFunctor}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

by reading $A, B, W_A, W_B, D_A, D_B, P, Q, R, E, H, C, G, N$. Thus completion after pairing is visible only as the finite product route followed by the completion handoff and Real seal.

Proof. Start with a product-completion consumer. The only source rows available to it are A and B, and the only observations of those sources are the finite window rows W_A, W_B with dyadic ledgers D_A, D_B.

The consumer must next read the product route P. This route pairs the displayed regular Cauchy data before the completion-functor handoff Q is available, so a detached product representative or host product equality cannot supply the handoff.

After Q is displayed, the seal row R is the only real-facing completion row. The exactness row E records the required order, and H, C, G, N transport, replay, source, and name the same route.

Therefore every accepted read factors through the displayed RegSeqRat, StreamName, DyadicRatCore, Product, CauchyCompletionFunctor, and Real rows before the Real seal is consumed. The packet exports no quotient completion, selected limit, countable-choice principle, host product equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{CauchyCompletionStrength}^{\uparrow}$ read that reaches a $\mathsf{Real}^{\uparrow}$ completion seal after product pairing must display two RegSeqRat source rows, two StreamName window rows, two DyadicRatCore tolerance ledgers, a ProductUp pairing route, a CauchyCompletionFunctor handoff, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: regseqrat, streamname, dyadicratcore, product, cauchy-completion-functor, real: $\mathsf{CauchyCompletionStrength}^{\uparrow}$ binds two regular Cauchy sources, paired finite windows, paired dyadic ledgers, a product route, a completion-functor handoff, a Real seal, exactness, transport, replay, provenance, and local naming in one strength packet; each listed sibling lacks at least one of those rows as a local product-completion coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionStrength}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionStrength}^{\uparrow}$ packet is generated from two RegSeqRat source rows, two StreamName window rows, two DyadicRatCore tolerance ledgers, a ProductUp pairing route, a CauchyCompletionFunctor handoff, a RealUp seal, exactness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStrength}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6130.1, Definition 6130.2, and Definition 6130.3 seed the finite product-strength surface for regular Cauchy completion over source, window, tolerance, product, completion-functor, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionStrength}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion, selected limits, countable choice, host product equality, detached product representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-window compatibility, product-route stability, completion-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionStrengthUp.

Chapter 6131

A Concrete Naming Certificate for $\mathsf{DoublingMetricSpace}^{\uparrow}$

Following Heinonen's metric-geometry treatment of doubling spaces and Kelley's finite-cover language, the $\mathsf{DoublingMetricSpace}^{\uparrow}$ packet records a displayed metric space whose balls admit uniformly bounded finite half-radius covers. It sits over the metric carrier of chapter 57, sends finite-cover data to chapter 127, provides compact-metric neighbourhood data for chapter 129, and routes every positive-radius and terminal distance read through chapter 38. The packet names only finite ball-cover rows, radius ledgers, center bundles, transport, replay, provenance, and local naming; it does not assert open-cover compactness, select global covers, or import a host doubling theorem.

Definition 6131.1 (Doubling metric space carrier)**.** *A* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (X, M, B, R, K, E, T, S, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-history carrier and classifier row.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *distance row over X, read through the metric certificate of chapter 57.*
3. *B is the parent-ball request row, naming a displayed center and positive radius.*
4. *R is the* $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive-radius ledger carrying the radius and its half-radius request.*
5. *K is a closed finite* $\mathsf{ProbeBundle}$ *of half-radius centers admitted by X.*
6. *E is the coverage ledger, recording that every displayed point of the parent ball in B lies in one of the half-radius balls centered by K.*
7. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *handoff row, exposing the finite center bundle and coverage rows to total-bounded consumers.*
8. *S is the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row, making the same finite-cover data available to compact-metric consumers without adding open-cover compactness.*
9. *H records componentwise* hsame *transport for X, M, B, R, K, E, T, S and the structural rows.*
10. *C records displayed* Cont *replay from metric-ball admission through the finite half-radius cover and the sibling handoff rows.*

11. *P is the* Pkg *provenance over* $\mathsf{DoublingMetricSpace}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DoublingMetricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, selected subcovers, host compactness, quotient point equality, countable choice, ambient real completeness, or a semantic doubling-space theorem.

Lean4 ✓ `BEDC.Derived.DoublingMetricSpaceUp.DoublingMetricSpaceTasteGate single carrier alignment`
Lean4 (definition) `BEDC.Derived.DoublingMetricSpaceUp.DoublingMetricSpaceUp`

Theorem 6131.2 (Doubling metric space NameCert obligations)**.** *For every accepted* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *carrier*

$$D = (X, M, B, R, K, E, T, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DoublingMetricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the point and metric rows; positive-radius and half-radius exactness through R; parent-ball admission through B; finite center-bundle admission through K; half-radius coverage through E; total-bounded and compact-metric handoff through T and S; componentwise transport, displayed replay, provenance, and local naming through H, C, P, N; and non-escape from open-cover compactness, choice-selected covers, quotient point equality, host compactness, countable choice, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6131.1. The point row X, metric row M, parent-ball request B, radius ledger R, center bundle K, coverage ledger E, and handoff rows T, S are displayed before the local naming row is consumed.

Positive-radius and half-radius exactness are read from R. Parent-ball admission is read from B, while the finite center bundle is read from K as a closed ProbeBundle over the same source.

Coverage is the row E. Each displayed point in the parent ball is paired with a center from K and a half-radius distance read through the same metric row M.

The sibling handoffs T and S expose that cover only as total-bounded and compact-metric finite data. The structural rows H, C, P, N transport, replay, source, and name the same route, so the excluded host principles cannot be recovered from the packet. □

Theorem 6131.3 (Doubling metric space total-bounded handoff)**.** *Every* $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *packet reads the finite route*

$\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, ProbeBundle, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, BHist, hsame, Cont, Pkg, NameCert

through the rows $X, M, B, R, K, E, T, S, H, C, P, N$. In particular, total-bounded consumers receive the half-radius center bundle and coverage ledger, compact-metric consumers receive only the same finite-cover handoff, and Real-facing consumers receive radius and distance rows only after that displayed route has been read.

Proof. Start with a total-bounded consumer. The rows K and E supply the finite center bundle and coverage ledger, while B and R record the parent ball and the radius-to-half-radius step.

Next consider a compact-metric consumer. It can read the packet only through S, which is downstream of the same K, E, T finite-cover route. Thus it receives no open-cover compactness object, selected subcover, or host compactness theorem.

Finally inspect a Real-facing radius or distance read. The metric row M and radius ledger R are replayed through C and transported through H before P, N package the handoff. Therefore

every total-bounded, compact-metric, or Real-facing use remains inside the displayed finite route and cannot import quotient point equality, countable choice, or ambient real completeness. □

Falsifiable prediction: Every accepted DoublingMetricSpaceUp cover read displays one point-history carrier, one metric-distance row, one parent-ball request, one positive-radius half-step, one finite ProbeBundle of centers, one half-radius coverage ledger, one TotallyBounded handoff, one CompactMetric handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, totallybounded, compactmetric, real: $\mathsf{DoublingMetricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, radius-halving ledger, parent-ball request, finite center bundle, half-radius coverage row, total-bounded handoff, compact-metric handoff, and Real-facing radius route in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{DoublingMetricSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DoublingMetricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6131.1, Definition 6131.2, and Definition 6131.3 seed the finite doubling metric-space cover route over metric, total-bounded, compact-metric, and Real-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DoublingMetricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, selected subcovers, host compactness, quotient point equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-halving exactness, parent-ball coverage, total-bounded handoff, compact-metric handoff, transport, replay, provenance, and local NameCert rows for BEDC.Derived.DoublingMetricSpaceUp.

Constructive story: A $\mathsf{DoublingMetricSpace}^{\uparrow}$ packet is generated from point-history and metric rows, a parent-ball request, a positive-radius half-step, a finite center ProbeBundle, a half-radius coverage ledger, total-bounded and compact-metric handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6132

A Concrete Naming Certificate for $\mathsf{RealCauchyFilterLimit}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealCauchyFilterLimit}^{\uparrow}$ packet names the finite Cauchy-filter-to-real limit route without treating a filter as a quotient object or choosing a representative subnet. It sits after the Cauchy-filter source of chapter 333, the StreamName windows of chapter 126, the regular-rational readback of chapter 301, the dyadic tolerance rows of chapter 291, and the terminal real-name seal of chapter 38. The route is a completion/filter sibling of chapter 3949: it names the limit-facing handoff, while that sibling names the functorial finite-window transport.

Definition 6132.1 (Real Cauchy filter limit carrier)**.** *A* $\mathsf{RealCauchyFilterLimit}^{\uparrow}$ *carrier is a tuple*

$$L = (F, B, W, R, D, E, H, C, P, N)$$

where F is the accepted $\mathsf{CauchyFilter}^{\uparrow}$ *source row, B is a finite filter-base observation row, W is the* $\mathsf{StreamName}^{\uparrow}$ *window service read from B, R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row over those windows, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{RealCauchyFilterLimit}^{\uparrow}}$ *row.*

The classifier compares packets only through $F, B, W, R, D, E, H, C, P, N$. It has no coordinate for a quotient filter, selected representative, choice-selected limit, trichotomy principle, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealCauchyFilterLimitUp.RealCauchyFilterLimitTasteGate single carrier alignment`

Theorem 6132.2 (Real Cauchy filter limit NameCert obligations)**.** *For every accepted carrier $L = (F, B, W, R, D, E, H, C, P, N)$ of Definition 6132.1, the local certificate surface is exhausted by Cauchy-filter admission, finite filter-base admission, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance checking, the RealUp seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}$ *row. No obligation row can be discharged by quotient filters, selected representatives, choice-selected limits, trichotomy, host real equality, or ambient Real completeness.*

Proof. Project the displayed carrier. Each row required by the certificate is one of $F, B, W, R, D, E, H, C, P, N$, and the only analytic source before the terminal seal is the finite filter-base row B under the Cauchy-filter source F.

Read the route through the finite windows. The row W exposes the StreamName observations supplied by B; the row R packages those observations as regular rational readback; and the row D records the dyadic tolerance comparisons needed before the terminal RealUp seal E is accepted.

Finally transport and replay the same rows. The hsame row H acts componentwise on the displayed coordinates, Cont replay C consumes only that displayed continuation, and P, N record provenance and local naming for the same finite packet. Since the carrier has no quotient-filter, selected-representative, choice-limit, trichotomy, host-equality, or completeness coordinate, the certificate cannot export those claims. □

Theorem 6132.3 (Real Cauchy filter limit regular readback)**.** *Every completion-facing consumer of an accepted* $\mathsf{RealCauchyFilterLimit}^{\uparrow}$ *packet must read the finite filter-base row B, then the StreamName window row W, then the RegSeqRat readback row R, then the DyadicRatCore tolerance row D, before it reaches the RealUp seal E. The route exports only this finite readback chain and refuses quotient filters, selected representatives, choice-selected limits, trichotomy, host equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier definition. The only path from the Cauchy-filter source F to the seal E passes through B, W, R, D in the displayed tuple.

Use the NameCert obligation theorem. It identifies the local surface with the same finite rows and therefore leaves no hidden coordinate through which a consumer could replace R by a selected limit or a quotient-filter representative.

The structural rows preserve the order rather than adding data. Transport H, continuation replay C, provenance P, and local naming N repack the finite readback chain, so the terminal seal remains a RealUp name certified by the displayed observations, not by ambient completeness or trichotomy. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyFilterLimit}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealCauchyFilterLimit}^{\uparrow}$ packet is generated from a CauchyFilter source row, a finite filter-base observation row, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterLimit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6132.1, Definition 6132.2, and Definition 6132.3 seed the finite Cauchy-filter limit packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterLimit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected representatives, choice-selected limits, trichotomy, host equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: A Lean carrier and checked NameCert theorem are required before this packet can retire the completion-facing horizon.

Chapter 6133

A Concrete Naming Certificate for $\mathsf{RealCompletionLimitRouter}^{\uparrow}$

Following the diagonal regular-Cauchy limit target in chapter 9238, the $\mathsf{RealCompletionLimitRouter}^{\uparrow}$ packet names the finite-window router that carries a completion request only as far as the displayed source rows permit. It is a router between observation budgets, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, and the RealUp seal; it is not a theorem that $\mathsf{Real}^{\uparrow}$ is complete and it does not choose a limit representative.

Definition 6133.1 (Real completion limit router carrier)**.** *A* $\mathsf{RealCompletionLimitRouter}^{\uparrow}$ *carrier is a tuple*

$$R = (O, W, Q, D, E, H, C, P, N)$$

where O is a finite observation-budget row, W is the $\mathsf{StreamName}^{\uparrow}$ *window row admitted by that budget, Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, H is componentwise* hsame *transport, C is displayed* Cont *replay, P is* Pkg *provenance, and N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionLimitRouter}^{\uparrow}}$ *row.*

The classifier reads packets only through $O, W, Q, D, E, H, C, P, N$. It has no coordinate for an unbounded diagonal limit, quotient stream equality, selected representative, countable-choice selector, host real equality, or ambient Real completeness.

Theorem 6133.2 (Real completion limit router NameCert obligations)**.** *For every accepted carrier $R = (O, W, Q, D, E, H, C, P, N)$ of Definition 6133.1, the local certificate surface consists exactly of observation-budget admission, StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance checking, RealUp sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The router cannot discharge an unbounded diagonal limit, quotient equality of streams, selected representative, countable-choice selector, host equality, or ambient Real completeness.*

Proof. Project the displayed carrier. The observation budget O is the only entry point for a completion request, and the window row W is the only StreamName service exposed by that budget.

Read the analytic route in order. The row Q packages the accepted windows as regular rational readback, the row D records the dyadic tolerance checks attached to that readback, and the row E seals the result as a RealUp name.

Finally use the structural rows. Transport H, continuation replay C, provenance P, and local naming N preserve the same finite-window route. Since no carrier coordinate names an unbounded

diagonal, quotient equality, selected representative, countable-choice selector, host equality, or Real-completeness principle, the certificate cannot export those claims. □

Lean4 ✓ `BEDC.Derived.RealCompletionLimitRouterUp.RealCompletionLimitRouterCarrier namecert obligations`

Theorem 6133.3 (Real completion limit router nonescape)**.** *Any consumer that reaches the* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RealCompletionLimitRouter}^{\uparrow}$ *packet must first expose the finite observation budget* O*, the StreamName window row* W*, the RegSeqRat readback row* Q*, and the DyadicRatCore tolerance row* D*. The consumer may read the seal as a finite routed name, but it may not replace the route by an ambient Real completeness theorem, a selected diagonal limit, quotient stream equality, countable choice, host equality, or a hidden representative.*

Proof. Start from the carrier definition. The seal row E is downstream of O, W, Q, D in the displayed packet, so a consumer cannot reach E without first exposing those rows.

Apply the NameCert obligation theorem. It identifies the local surface as the finite router rows and structural replay rows, leaving no hidden coordinate that could supply a global completion principle.

The nonescape conclusion is therefore a row-exhaustion claim. Once $O, W, Q, D, E, H, C, P, N$ have been read, the packet has exported the finite router and nothing beyond it; selected limits, quotient streams, countable choice, host equality, and ambient Real completeness remain outside the carrier. □

Theorem 6133.4 (Real completion router schedule-family source)**.** *A completion request whose observation budget is supplied by a schedule family first consumes the* $\mathsf{CauchyModulusSpace}^{\uparrow}$ *route of chapter 5887 before it opens the* $\mathsf{RealCompletionLimitRouter}^{\uparrow}$ *rows* $O, W, Q, D, E, H, C, P, N$*. The router may then expose the finite observation budget, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, and RealUp seal, but it cannot replace the schedule-family source by quotient stream equality, a selected limit, countable choice, host real equality, or ambient Real completeness.*

Proof. Begin with a completion request whose budget is indexed by a schedule family. The $\mathsf{CauchyModulusSpace}^{\uparrow}$ finite route and filter handoff supply that family only through finite schedule rows, finite $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledgers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and terminal $\mathsf{Real}^{\uparrow}$ seal boundaries.

Apply Definition 6133.3. Once the request reaches the router, the seal row E is downstream of the displayed rows O, W, Q, D, and the structural rows H, C, P, N replay the same finite route.

Thus the router records a sibling dependency on schedule-family modulus data before it exposes its own rows. No step adds a coordinate for quotient stream equality, selected limits, countable choice, host real equality, or an ambient Real-completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionLimitRouter}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{RealCompletionLimitRouter}^{\uparrow}$ packet is generated from a finite observation-budget row, a StreamName window row, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCompletionLimitRouter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6133.1, Definition 6133.2, and Definition 6133.3 seed the finite-window router for the Real completeness route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCompletionLimitRouter}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full RealUp completeness, unbounded diagonal limits, selected representatives, quotient stream equality, countable choice, host equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window routing, RegSeqRat readback, DyadicRatCore tolerance, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionLimitRouterUp.

Chapter 6134

A Concrete Naming Certificate for $\mathsf{CauchyBornologyBasis}^{\uparrow}$

Following Bishop and Bridges on constructive completion and Bourbaki/Kelley on bornologies, the $\mathsf{CauchyBornologyBasis}^{\uparrow}$ packet records the finite basis-level route by which Cauchy filter observations, boundedness control, metric boundedness, stream windows, rational readback, dyadic tolerances, and the terminal real seal are exposed before a completion-facing consumer may act. It sits between the Cauchy bornology of chapter 1896, the finite Cauchy filter basis of chapter 1501, the metric bornology of chapter 4091, the located Cauchy filter basis of chapter 5840, the stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The chapter names the basis-bornology bridge needed by Real and Completion routes; it does not quotient filters, select limits, use countable choice, or assume ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6134.1 (Cauchy bornology basis carrier)**.** *A* $\mathsf{CauchyBornologyBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, G, M, S, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row, exposing finite basic filter requests, directed refinements, and Cauchy-pressure witnesses.*
2. *G is the* $\mathsf{CauchyBornology}^{\uparrow}$ *row, recording which displayed Cauchy-filter requests are bounded in the bornological sense.*
3. *M is the* ***??*** *row, tying the boundedness read to a metric-ball or diameter ledger rather than to host-set inclusion.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window service used to inspect the requested basis entries.*
5. *W is the window-boundedness row, matching the displayed stream windows with the bornology rows G, M.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to those windows and boundedness reads.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback available only after F, G, M, S, W, D have been displayed.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only through the displayed basis-bornology route.*

9. *H records componentwise* hsame *transport for all basis, bornology, metric, window, tolerance, readback, seal, exactness, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay of the same route.*
11. *P is the* Pkg *provenance over* $\mathsf{CauchyBornologyBasis}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyBornology}^{\uparrow}$, *??*, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyBornologyBasis}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, G, M, S, W, D, R, E, H, C, P, N$. It has no coordinate for quotient filters, selected limits, countable choice, host bounded-set equality, hidden Cauchy representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyBornologyBasisUp.TasteGate.CauchyBornologyBasisTasteGate single carrier alignment`

Theorem 6134.2 (Cauchy bornology basis NameCert obligations)**.** *For every accepted* $\mathsf{CauchyBornologyBasis}^{\uparrow}$ *carrier*

$$B = (F, G, M, S, W, D, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyBornologyBasis}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *row,* $\mathsf{CauchyBornology}^{\uparrow}$ *row, ?? row,* $\mathsf{StreamName}^{\uparrow}$ *windows, window-boundedness row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming rows;*
2. *basis-bornology compatibility, requiring every boundedness read in G to be attached to a displayed finite basis request from F;*
3. *metric boundedness stability, requiring M and W to witness the same bounded window before D, R, E are consumed;*
4. *completion-route nonescape, excluding quotient filters, selected limits, countable choice, host bounded-set equality, hidden Cauchy representatives, and ambient real completeness from the packet;*
5. *structural persistence through componentwise* hsame*, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row N.*

No other NameCert obligation is available from the Cauchy bornology basis packet.

Proof. Project the carrier of Definition 6134.1. Its analytic rows are exactly the Cauchy filter basis, Cauchy bornology row, metric bornology row, stream-window service, window-boundedness row, dyadic tolerance ledger, regular rational readback, and terminal real seal.

The first three obligation rows separate admission from compatibility. Carrier admission displays every coordinate. Basis-bornology compatibility ties G to the finite basis requests in F, and metric boundedness stability forces M and W to describe the same visible bounded windows before the tolerance and readback rows are public.

The completion-route nonescape row follows from the same inventory. Since no quotient filter, selected limit, countable-choice schedule, host bounded-set equality, hidden Cauchy representative, or ambient completeness coordinate appears in B, none can discharge a local obligation.

Finally H, C, P, N preserve the displayed rows by transport, replay, provenance, and naming. The five rows therefore exhaust the local NameCert surface. □

Theorem 6134.3 (Cauchy bornology basis filter handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{CauchyBornologyBasis}^{\uparrow}$ *carrier reads the finite route*

$$\mathsf{CauchyFilterBasis}^{\uparrow},\ \mathsf{CauchyBornology}^{\uparrow},\ ??,\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow}$$

before it can consume the terminal $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports only* $F, G, M, S, W, D, R, E, H, C, P, N$ *and refuses quotient filters, selected representatives, countable choice, host bounded-set equality, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a completion-facing read of B. The only filter source available to the consumer is the displayed basis row F, and the only bornological controls are G and M.

The consumer must next inspect finite windows. The rows S and W display the requested window and its boundedness read, while D records the dyadic tolerance ledger tied to the same visible data.

Only after this route is displayed may R supply regular rational readback and E supply the terminal real seal. The structural rows H, C, P, N transport, replay, source, and name that same route rather than adding a quotient filter or selected representative.

Thus the handoff is exactly the finite basis-bornology route listed in the statement, and it has no place for countable choice, host bounded-set equality, or ambient real completeness. □

Theorem 6134.4 (Cauchy bornology basis Real completion boundary)**.** *An accepted* $\mathsf{CauchyBornologyBasis}^{\uparrow}$ *packet reaches a Real and Completion boundary only through the displayed rows* $F, G, M, S, W, D, R, E, H, C, P, N$*. In particular the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is downstream of the finite Cauchy filter basis, bornology, metric bornology, stream-window, dyadic tolerance, and regular rational readback rows, and no quotient filter, selected limit, countable-choice selector, detached completion object, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness row can be inserted before* E*.*

Proof. Use Definition 6134.2. It identifies the complete local certificate surface and shows that the terminal seal is part of the same finite packet as the basis, bornology, metric, window, tolerance, and readback rows.

Now apply the filter handoff theorem, Definition 6134.3. Every completion-facing read reaches E only after F, G, M, S, W, D, R have been displayed and after the structural rows have replayed the same route.

The excluded objects have no carrier coordinate. A quotient filter, selected limit, countable-choice selector, detached completion object, or ambient completeness row would have to enter between R and E or replace one of the displayed support rows, but the packet exposes neither insertion point. □

Falsifiable prediction: Every accepted $\mathsf{CauchyBornologyBasis}^{\uparrow}$ read that reaches a $\mathsf{Real}^{\uparrow}$ seal must display a CauchyFilterBasis row, a CauchyBornology row, a MetricBornology row, StreamName windows, a window-boundedness row, a DyadicRatCore tolerance ledger, RegSeqRat readback, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: cauchy-filter-basis, cauchy-bornology, metric-bornology, located-cauchy-filter-basis, streamname, dyadicratcore, regseqrat, real: $\mathsf{CauchyBornologyBasis}^{\uparrow}$ binds finite filter-basis requests, bornological boundedness, metric boundedness, stream-window inspection, dyadic tolerance, regular rational readback, a Real seal, transport, replay, provenance, and local naming in one basis-level packet; each listed sibling lacks at least one of these rows as a local completion-facing coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyBornologyBasis}^{\uparrow}}$.

Constructive story: A CauchyBornologyBasis$^{\uparrow}$ packet is generated from a CauchyFilterBasis row, a CauchyBornology row, a MetricBornology row, StreamName windows, a window-boundedness row, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CauchyBornologyBasis}^{\uparrow}}$, seed)

Scope closed: Definition 6134.1, Definition 6134.2, Definition 6134.3, and Definition 6134.4 seed the finite basis-bornology route for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\text{CauchyBornologyBasis}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filters, selected representatives, selected limits, countable choice, host bounded-set equality, detached completion objects, ambient Real$^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, basis-bornology compatibility, metric boundedness stability, filter handoff, Real completion boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyBornologyBasisUp.

Chapter 6135

A Concrete Naming Certificate for $\mathsf{CompactDiniModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Dini's theorem, the $\mathsf{CompactDiniModulus}^{\uparrow}$ packet records the finite BEDC route by which a monotone continuous family on a compact metric source exposes one uniform tolerance modulus. It sits after the compact metric source of chapter 129, the continuous-map rows of chapter 130, the finite lower-bound fold of chapter 4096, the uniform-modulus surface of chapter 369, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the finite Dini-modulus route without importing host-selected limits, open-cover compactness, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6135.1 (Compact Dini modulus carrier)**.** *A* $\mathsf{CompactDiniModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, M, \epsilon, L, U, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the compact finite-net data and finite source observations consumed by the Dini fold.*
2. *F is the monotone finite family of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, recording the displayed maps on K and the continuity ledgers inspected at the finite net.*
3. *M records monotone decrease toward the zero or limiting row, including the visible comparisons used at each displayed net center.*
4. *ϵ is the positive tolerance request, stored as a finite rational or dyadic budget before any modulus row is exported.*
5. *L is the finite compact-net lower-bound and Dini-fold row, collecting the local lower-bound comparisons and the finite maximum of the displayed center thresholds.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row obtained from K, F, M, ϵ, L.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the tolerance and comparison budgets used by L and U.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, exposing only the real-valued endpoint read permitted after the regular rational readback.*
9. *H records componentwise* hsame *transport for the compact source, family, monotone ledger, tolerance, Dini fold, modulus output, readback, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay from compact finite-net data through monotone-family observations, the Dini fold, regular rational readback, and the uniform-modulus output.*
11. *P is the* Pkg *provenance over* $\mathsf{CompactDiniModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLowerBoundFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactDiniModulus}^{\uparrow}}$ *row.*

The carrier exposes only compact finite-net source data, monotone continuous-family rows, tolerance budgets, finite lower-bound folds, uniform-modulus output, regular rational readback, real sealing, componentwise transport, displayed replay, provenance, and local naming. It has no coordinate for host-selected limits, arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CompactDiniModulusUp.CompactDiniModulusTasteGate single carrier alignment`

Theorem 6135.2 (Compact Dini modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactDiniModulus}^{\uparrow}$ *carrier*

$$D = (K, F, M, \epsilon, L, U, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactDiniModulus}^{\uparrow}}$ *surface consists exactly of five analytic rows together with the structural certificate rows:*

1. *compact-source admission through K;*
2. *monotone-family admission through F and M;*
3. *finite Dini-fold exactness through ϵ and L;*
4. *uniform-modulus output through U and regular rational readback through R;*
5. *nonescape from host-selected limits, arbitrary open-cover compactness, quotient real equality, and ambient real completeness through the seal row E and the transport, replay, provenance, and naming rows H, C, P, N.*

Proof. Project the carrier of Definition 6135.1. The analytic data are the compact source K, the continuous family F, the monotone ledger M, the tolerance row ϵ, the finite Dini fold L, the uniform-modulus row U, the regular rational readback R, and the real seal E.

The compact-source obligation is exactly the admission of K. The monotone-family obligation is exactly the joint visibility of F and M, because no later row contains a separate family selector or host order comparison.

The finite Dini-fold obligation is the route through ϵ and L. The tolerance budget is fixed before the fold is read, and L stores the compact-net lower-bound comparisons together with the finite maximum of the displayed center thresholds.

The output obligation is the handoff through U and R. The row U is the only uniform-modulus output, and R is the only regular rational readback exposed before the terminal real seal.

Finally E, H, C, P, N close the packet boundary. Since the carrier has no coordinate for host-selected limits, arbitrary open covers, selected subcovers, quotient real equality, countable choice, or ambient real completeness, none of those readings can be exported by the local NameCert surface. □

Theorem 6135.3 (Compact Dini modulus uniform-convergence route)**.** *For every accepted* $\mathsf{CompactDiniModulus}^{\uparrow}$ *carrier, any consumer that reads a uniform-modulus claim must route the request through*

$$K, F, M, \epsilon \longmapsto L \longmapsto U \longmapsto R, E \longmapsto H, C, P, N.$$

Thus compact monotone convergence is exposed as displayed finite rows before any $\mathsf{Real}^{\uparrow}$ *consumer may read a uniform-modulus conclusion.*

Proof. Start with the NameCert obligations of Definition 6135.2. They place compact-source admission, monotone-family admission, tolerance budgeting, finite Dini-fold exactness, uniform-modulus output, regular rational readback, real sealing, transport, replay, provenance, and local naming in the displayed carrier inventory.

The first stage reads K, F, M, ϵ. These rows determine the compact finite-net source, the continuous family, the monotone decrease toward the zero or limiting row, and the requested tolerance budget.

The second stage reads L. This row is the only place where the finite lower-bound comparisons and the finite maximum of center thresholds are assembled, so the Dini fold cannot be bypassed by a host compactness or limit-selection argument.

The third stage reads U and then R, E. The uniform-modulus row is exported before regular rational and real consumers inspect it, and those consumers see only the routed readback and seal rows.

The final stage reads H, C, P, N. Componentwise transport, displayed replay, provenance, and local naming preserve the same route without adding analytic data. Therefore every uniform-convergence use of the packet factors through the displayed finite rows before any $\mathsf{Real}^{\uparrow}$ consumer reads the conclusion. □

Falsifiable prediction: Every accepted $\mathsf{CompactDiniModulus}^{\uparrow}$ read must expose compact-source admission, monotone continuous-family admission, finite Dini-fold exactness, uniform-modulus output, regular rational readback, real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, finite-lower-bound-fold, uniformmodulus, regseqrat, real: $\mathsf{CompactDiniModulus}^{\uparrow}$ is not bijective to any listed sibling because it binds a monotone compact-family route to a finite Dini-fold row before UniformModulus and Real consumers may read the result.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactDiniModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactDiniModulus}^{\uparrow}$ packet is generated from CompactMetric finite-net source rows, monotone ContinuousMap family rows, tolerance-budget rows, finite lower-bound Dini-fold rows, UniformModulus output rows, RegSeqRat readback, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactDiniModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6135.1, Definition 6135.2, and Definition 6135.3 seed the finite BEDC route from compact monotone continuous-family data to a displayed uniform modulus.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactDiniModulus}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CompactDiniModulusUp`

Bridge status: none

Not claimed: This chapter does not close host-selected limits, arbitrary open-cover compactness, selected subcovers, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, monotone-family admission, finite Dini-fold exactness, uniform-modulus output, readback and seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactDiniModulusUp.

Chapter 6136

A Concrete Naming Certificate for $\mathsf{BairePointwiseOscillation}^{\uparrow}$

Following Baire and Lebesgue on pointwise oscillation, the $\mathsf{BairePointwiseOscillation}^{\uparrow}$ packet records the finite value-window evidence by which a pointwise-limit ledger exposes an oscillation bound at named source points. It reads the source and approximant windows supplied by chapter 1604, checks the rational value windows through chapter 301, seals the endpoint observations through chapter 38, keeps the inspection schedule visible through chapter 126, and exports only the epigraph-facing handoff that chapter 1536 may consume. The packet is a local certificate for displayed oscillation data, not a topology of function spaces and not a theorem about all discontinuity sets.

Definition 6136.1 (Baire pointwise oscillation carrier)**.** *A* $\mathsf{BairePointwiseOscillation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, F, S, Q^-, Q^+, R^-, R^+, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the source-context row inherited from the* $\mathsf{BaireOneFunction}^{\uparrow}$ *packet, naming the source points and basic-open evidence inspected by the oscillation certificate.*
2. *F is the pointwise-limit family ledger of* $\mathsf{BaireOneFunction}^{\uparrow}$ *approximant observations whose two displayed value windows are being compared.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row selecting the two finite inspection windows, the source point, and the precision request.*
4. *Q^- and Q^+ are the lower and upper* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback rows for the two displayed value windows.*
5. *R^- and R^+ are the corresponding* $\mathsf{Real}^{\uparrow}$ *seals, reached only after X, F, S, Q^-, Q^+ have been displayed.*
6. *B is the rational oscillation-bound row recording a displayed $b \in \mathbb{Q}_{\geq 0}$ with the certified window inequality $R^+ - R^- \leq b$ at the named source point.*
7. *H records componentwise* hsame *transport for the source context, pointwise-limit ledger, stream schedule, rational readbacks, real seals, oscillation bound, continuation, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a source-point request through the* $\mathsf{BaireOneFunction}^{\uparrow}$ *value windows, the* $\mathsf{StreamName}^{\uparrow}$ *schedule, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, the* $\mathsf{Real}^{\uparrow}$ *seals, the rational oscillation bound, and the lower-semicontinuous handoff surface.*
9. *P is the* Pkg *provenance over* $\mathsf{BairePointwiseOscillation}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BairePointwiseOscillation}^{\uparrow}}$ *row.*

The carrier exports no ambient topology on a function space, no arbitrary selector of subsequences, no quotient equality of real names, no global discontinuity classification, and no completeness principle for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.BairePointwiseOscillationUp.BairePointwiseOscillationTasteGate single carrier alignment`

Theorem 6136.2 (Baire pointwise oscillation NameCert obligations)**.** *For every accepted* $\mathsf{BairePointwiseOscillation}^{\uparrow}$ *carrier* $K = (X, F, S, Q^-, Q^+, R^-, R^+, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BairePointwiseOscillation}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the source context, pointwise-limit family ledger, stream schedule, lower and upper rational readbacks, lower and upper real seals, rational oscillation bound, transport, continuation, provenance, and local naming rows;*
2. *value-window admission, requiring both displayed value windows to be read from the same* $\mathsf{BaireOneFunction}^{\uparrow}$ *pointwise-limit ledger over the same source row X;*
3. *schedule coherence, requiring the* $\mathsf{StreamName}^{\uparrow}$ *row S to name the source point, precision request, and two finite approximant windows used by Q^- and Q^+;*
4. *rational readback exactness, requiring Q^- and Q^+ to feed the real seals R^- and R^+ before the bound row B is available;*
5. *oscillation-bound stability, requiring the displayed inequality $R^+ - R^- \leq b$ in B to be preserved by the componentwise* hsame *rows in H;*
6. *handoff compatibility, allowing* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumers to read only the displayed oscillation-bound row and the route that produced it;*
7. *ledger refusal, excluding host function-space topology, arbitrary subsequence selection, quotient equality of real names, global discontinuity theorems, and completeness of* $\mathsf{Real}^{\uparrow}$.

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6136.1. The accepted packet exposes exactly the source context, pointwise-limit family ledger, stream schedule, two rational readbacks, two real seals, rational bound, transport, continuation, provenance, and local naming rows.

Value-window admission and schedule coherence come from the rows F and S. Both value windows are read from the same $\mathsf{BaireOneFunction}^{\uparrow}$ ledger, and S records the finite inspection windows before any rational readback can be used.

Rational readback exactness is forced by the order of the route. The rows Q^- and Q^+ feed the real seals R^- and R^+, and the bound row B is downstream of those seals.

Transport stability is componentwise. The hsame rows in H identify the displayed source, ledger, schedule, readback, seal, bound, replay, provenance, and naming coordinates, so the displayed inequality is preserved along accepted classifier moves.

The lower-semicontinuous handoff is a read of the same route. Since P and N contain no coordinate for ambient topology, arbitrary subsequences, quotient equality, global discontinuity classification, or $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the certificate. □

Theorem 6136.3 (BaireOneFunction oscillation handoff)**.** *Every* $\mathsf{BairePointwiseOscillation}^{\uparrow}$ *consumer row exported by an accepted carrier factors through the displayed* $\mathsf{BaireOneFunction}^{\uparrow}$ *value-window route*

$$X, F, S, Q^-, Q^+ \longmapsto R^-, R^+ \longmapsto B.$$

Consequently a downstream $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer may use the rational oscillation bound and the displayed provenance of its two value windows, but it receives no host function-space topology, no arbitrary subsequence selector, no quotient equality of real names, and no global theorem about the discontinuity set of a Baire-one function.*

Proof. Start with a consumer of the oscillation-bound row B. By carrier admission, B appears only after the source row X, the $\mathsf{BaireOneFunction}^{\uparrow}$ family ledger F, the schedule S, the rational readbacks Q^-, Q^+, and the real seals R^-, R^+.

Replay the continuation row C. The source-point request first reads the pointwise-limit ledger, then selects the two finite value windows through S, checks their rational windows through Q^- and Q^+, and seals the endpoints as R^- and R^+.

The bound row is therefore a downstream finite certificate, not an independent analytic object. Transport by H preserves the same coordinates, and the provenance row P records only the named packets in the displayed route.

A $\mathsf{LowerSemicontinuous}^{\uparrow}$ consumer can read the handoff only through that route. Since no excluded coordinate is present in the packet, the handoff exports neither ambient topology nor any global discontinuity theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{BairePointwiseOscillation}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BairePointwiseOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6136.1, Definition 6136.2, and Definition 6136.3 seed the finite pointwise-oscillation packet from Baire-one value windows to lower-semicontinuous consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BairePointwiseOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function-space topology, arbitrary subsequence selection, quotient equality of real names, global discontinuity classification, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, value-window admission, schedule coherence, rational readback exactness, oscillation-bound stability, handoff compatibility, and ledger refusal for BEDC.Derived.BairePointwiseOscillationUp.

Constructive story: A $\mathsf{BairePointwiseOscillation}^{\uparrow}$ packet is generated from a $\mathsf{BaireOneFunction}^{\uparrow}$ pointwise-limit ledger, a $\mathsf{StreamName}^{\uparrow}$ schedule, two $\mathsf{RegSeqRat}^{\uparrow}$ readbacks, two $\mathsf{Real}^{\uparrow}$ seals, a rational oscillation-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6137

A Concrete Naming Certificate for $\mathsf{StolzCesaroTheorem}^{\uparrow}$

Following Stolz and Cesaro, and Hardy's summability treatment, the $\mathsf{StolzCesaroTheorem}^{\uparrow}$ packet records the finite theorem surface by which quotient-difference data and denominator growth are read into a Cesaro-style averaging handoff before any regular real-sequence consumer may inspect a limit claim. It sits beside the Hardy–Cesaro mean packet of chapter 1765, the Toeplitz averaging route of chapter 1682, the regular rational sequence surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite handoff from inspected sequence windows to a theorem-level summability certificate; it does not assert ambient real completeness, quotient-stream equality, or an uninspected limit.

Definition 6137.1 (Stolz-Cesaro theorem carrier)**.** *A* $\mathsf{StolzCesaroTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (A, D, Q, C, R, E, H, T, P, N)$$

with the following displayed rows:

1. *A is the source real-sequence window row, displaying the inspected numerator and denominator observations available to the theorem packet.*
2. *D is the quotient-difference row, recording the finite ratios of successive numerator and denominator differences used by the Stolz–Cesaro test.*
3. *Q is the denominator-growth row, recording the monotone unbounded denominator evidence read by the packet before any averaging row is exposed.*
4. *C is the Cesaro or Toeplitz averaging row, read through the sibling routes named by chapter 1765 and chapter 1682.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from A, D, Q, C and the dyadic tolerance data.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite sequence windows, difference quotients, denominator-growth row, averaging row, and regular readback have been displayed.*
7. *H records componentwise* hsame *transport for A, D, Q, C, R, E.*
8. *T records displayed* Cont *replay from source windows through quotient differences, growth, averaging, regular readback, and the Real seal.*
9. *P is the* Pkg *provenance over* $\mathsf{StolzCesaroTheorem}^{\uparrow}$, $\mathsf{HardyCesaroMean}^{\uparrow}$, $\mathsf{ToeplitzLemma}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{StolzCesaroTheorem}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient completeness theorem, quotient sequence equality, selected real limit, countable choice, or host-order convergence outside the displayed rows.

Theorem 6137.2 (Stolz-Cesaro theorem NameCert obligations)**.** *For every accepted* $\mathsf{StolzCesaroTheorem}^{\uparrow}$ *carrier $S = (A, D, Q, C, R, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{StolzCesaroTheorem}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed packet $A, D, Q, C, R, E, H, T, P, N$;*
2. *quotient-difference exactness, requiring every tested ratio to be read through D over the source windows A;*
3. *denominator-growth and averaging compatibility, requiring Q to be displayed before the Cesaro or Toeplitz row C is consumed;*
4. *regular readback and Real seal nonescape, requiring R and E to factor through the* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *sibling surfaces rather than through quotient streams or selected limits;*
5. *structural preservation through H, T, P, N, requiring componentwise transport, displayed replay, provenance, and local naming to preserve the same theorem packet.*

Lean4 ✓ `BEDC.Derived.StolzCesaroTheoremUp.StolzCesaroTheoremTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6137.1. The analytic rows are the source windows A, quotient-difference row D, denominator-growth row Q, averaging row C, regular readback row R, and Real seal row E.

The quotient test is finite because every accepted ratio is recorded in D over the windows in A. The denominator-growth row Q is read before C, so the Cesaro or Toeplitz averaging route cannot be consumed as a free summability theorem detached from the displayed denominator data.

The readback and seal rows are downstream from the same route. Since R and E are reached only through the regular rational, dyadic tolerance, and Real seal siblings, no quotient stream, selected limit, countable choice, or ambient completeness coordinate is available.

Finally H, T, P, N transport, replay, source, and name only those displayed rows. Thus the five listed obligations exhaust the local NameCert surface for the theorem packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{StolzCesaroTheorem}^{\uparrow}}$**.**

Constructive story: A $\mathsf{StolzCesaroTheorem}^{\uparrow}$ packet is generated from source real-sequence windows, quotient-difference rows, denominator-growth rows, Cesaro or Toeplitz averaging rows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{StolzCesaroTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 6137.1 and Definition 6137.2 seed the finite theorem packet over sequence windows, quotient differences, denominator growth, averaging, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{StolzCesaroTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient stream equality, selected limits, countable choice, host-order convergence, Lean verification, or bridge checking for $\mathsf{StolzCesaroTheorem}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, quotient-difference exactness, denominator-growth compatibility, averaging handoff, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.StolzCesaroTheoremUp.

Chapter 6138

A Concrete Naming Certificate for $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$

The $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ packet realizes the local explanation-state non-equivalence row of section 9462.11 as finite NameCert data. It sits beside the concept registry surface of chapter 1252, the audit system of chapter 1258, the axiom-purity gate of chapter 1259, the effective replacement ledger of chapter 568, and the philosophy map target of chapter 1277. The packet names a single common-source classifier disagreement between two explanation states; it does not claim global interpretability quality, hidden semantic truth, or classifier novelty without a displayed witness.

Definition 6138.1 (Classifier novelty witness carrier)**.** *A* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (E, E', X, J, L, H, C, P, N)$$

with the following displayed rows:

1. *E is the source explanation-state row before the classifier judgment changes.*
2. *E' is the target explanation-state row after the classifier judgment changes.*
3. *X is the common-source display row, naming the inspected objects shared by E and E'.*
4. *J is the changed judgment row, recording a displayed pair in X whose classifier value differs between E and E'.*
5. *L is the ledger row, read through the effective replacement ledger, audit system, axiom-purity gate, concept registry surface, and philosophy map target siblings.*
6. *H records componentwise* hsame *transport for E, E', X, J, L.*
7. *C records displayed* Cont *replay through the common source, changed judgment, and ledger rows.*
8. *P is the* Pkg *provenance over* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, $\mathsf{ConceptRegistrySurface}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, $\mathsf{AxiomPurityGate}^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, $\mathsf{PhilosophyMapTarget}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyWitness}^{\uparrow}}$ *row.*

The carrier has no coordinate for a hidden semantic classifier, global truth predicate, host equivalence relation, quotient witness, or classifier novelty value without the displayed changed judgment row J.

Theorem 6138.2 (Classifier novelty witness NameCert obligations)**.** *For every accepted* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ *carrier* $W = (E, E', X, J, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyWitness}^{\uparrow}}$ *surface consists exactly of carrier admission for the two explanation-state rows, common-source display, changed judgment, ledger row, transport, replay, provenance, and local naming; classifier non-equivalence exactness through* J*; ledger visibility through* L*; and no-smuggling refusal for hidden semantic truth, host classifier equivalence, quotient witnesses, and unstated global novelty.*

Proof. Project the carrier of Definition 6138.1. The two explanation states E, E' and the common-source row X determine the finite domain on which a classifier comparison may be read.

The changed judgment row J is the only coordinate that certifies non-equivalence. It records a displayed pair in X with different classifier values, so a consumer cannot infer classifier novelty from a prose explanation, a hidden semantic relation, or a global truth predicate.

The ledger row L exposes the registry, audit, purity, replacement, and philosophy-map dependencies before the witness is exported. The structural rows H, C, P, N transport, replay, source, and name the same finite packet, so the listed obligations exhaust the NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ consumer that asserts classifier novelty without displaying $E, E', X, J, L, H, C, P, N$ refutes the carrier surface.

Independence witness: conceptregistrysurface, auditsystem, axiompuritygate, effective-replacement-ledger, philosophymaptarget: none is bijective because $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ simultaneously carries two explanation states, a common-source display, a changed classifier judgment, audit-facing ledger evidence, transport, replay, provenance, and local naming.

The Lean follow-up must provide $\mathsf{Nontrivial\ ClassifierNoveltyWitness}^{\uparrow}$ and $\mathsf{FieldFaithful\ ClassifierNoveltyWitness}^{\uparrow}$ instances before any checked marker for BEDC.Derived.ClassifierNoveltyWitnessUp can be cited.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyWitness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ packet is generated from two explanation-state rows, a common-source display row, a changed classifier judgment row, a ledger row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierNoveltyWitness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6138.1, Definition 6138.2, the falsifiable prediction, and the independence witness seed the finite classifier-novelty witness packet over explanation states, common source, changed judgment, ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierNoveltyWitness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global interpretability quality, hidden semantic truth, host classifier equivalence, quotient witnesses, classifier novelty without a displayed witness, Lean verification, or bridge checking for $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, changed-judgment exactness, ledger visibility, no-smuggling refusal, transport, replay, provenance, and local naming for BEDC.Derived.ClassifierNoveltyWitnessUp.

Chapter 6139

A Concrete Naming Certificate for $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ packet records the finite handoff from regular Cauchy name data to located apartness reads. It sits after the stream-name surface of chapter 126, the regular rational sequence surface of chapter 301, the dyadic rational core of chapter 291, the located-real surface of chapter 1933, the terminal real-name seal of chapter 38, and the Bishop located-real completion route of chapter 3488. The packet exposes apartness as inspected finite NameCert data; it does not import trichotomy, quotient stream equality, selected limits, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6139.1 (Regular Cauchy apartness reflector carrier)**.** *A* $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, R, D, A, L, E, H, C, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, naming the inspected source positions and readback order.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row whose accepted windows are being compared.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and positive-gap ledger used by the apartness read.*
4. *A is the displayed apartness row, recording the finite positive separation evidence accepted for the same regular Cauchy windows.*
5. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *handoff row, exposing the located read reached from S, R, D, A.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located-real handoff has been displayed.*
7. *H records componentwise* hsame *transport for S, R, D, A, L, E.*
8. *C records displayed* Cont *replay from regular Cauchy windows through dyadic positive-gap evidence, located-real handoff, and the Real seal.*
9. *N is both the* Pkg *provenance over the listed BEDC dependencies and the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessReflecto}}$ *row.*

The classifier compares accepted packets only through $S, R, D, A, L, E, H, C, N$. It has no coordinate for trichotomy, quotient stream equality, selected limit representatives, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessReflectorUp.RegularCauchyApartnessReflectorTasteGate`
`single carrier alignment`

Theorem 6139.2 (Regular Cauchy apartness reflector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ *carrier*

$$K = (S, R, D, A, L, E, H, C, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessReflector}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the stream windows, regular rational source, dyadic positive-gap ledger, apartness row, located-real handoff, Real seal, transport, replay, provenance, and local name;*
2. *window and readback exactness, requiring the read through S and R to be the same finite regular Cauchy window consumed by the apartness row;*
3. *apartness-row exposure, requiring every accepted positive separation read to pass through A after the dyadic ledger D has been displayed;*
4. *located-real handoff, requiring the row L to be reached only from S, R, D, A;*
5. *Real-seal nonescape, requiring E to expose only the terminal seal after the located-real handoff;*
6. *transport, replay, provenance, and local naming, requiring H, C, N to preserve and name the same displayed route.*

No obligation supplies trichotomy, quotient stream equality, selected limits, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness outside these rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessReflectorNamecertObligations`

Proof. Project the carrier of Definition 6139.1. The displayed tuple $S, R, D, A, L, E, H, C, N$ is the entire public surface, so carrier admission is admission of those rows.

Window and readback exactness are forced by the order of the tuple. The stream-name row S fixes the inspected windows, and the regular rational row R fixes the rational readback on those windows before any apartness row is available.

The apartness-row exposure is the joint role of D and A. The dyadic ledger supplies the positive-gap cells, while A records the accepted finite apartness read over those same cells.

The located-real handoff and Real seal are downstream rows. The handoff L can be read only after S, R, D, A, and the seal E can be read only after L has been displayed.

Finally, H, C, N transport, replay, source, and name the same route. Since the excluded objects are absent as carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 6139.3 (Regular Cauchy apartness located-real handoff)**.** *Let $K = (S, R, D, A, L, E, H, C, N)$ be an accepted* $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ *carrier. Every accepted apartness read from A passes through the finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{LocatedReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as witnessed by the rows S, R, D, L, E of the same packet, with H, C, N carrying only transport, replay, provenance, and local naming for that route.

Lean4 ✓ `BEDC.Derived.RegularCauchyApartnessReflectorLocatedRealHandoff`

Proof. Start with an accepted apartness read from A. By the carrier definition, the only inspected source windows available to that read are named by S.

The regular rational readback row R is then the only source of rational observations on those windows. The positive-gap evidence for apartness must next be read through the dyadic ledger D, so the apartness row is not a detached order comparison or a trichotomy branch.

The located-real row L consumes exactly the displayed stream, regular rational, dyadic, and apartness rows. Thus the handoff is a located-real read, not a quotient-stream equality or selected representative.

After L is displayed, the terminal row E seals the same read as a $\mathsf{Real}^{\uparrow}$ name. The structural rows H, C, N preserve, replay, source, and name the route without adding ambient completeness, so every accepted apartness read passes through the displayed finite handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat regular rational readback, DyadicRatCore positive-gap evidence, an apartness row, a LocatedReal handoff, a Real seal, componentwise hsame transport, displayed Cont replay, provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6139.1, Definition 6139.2, and Definition 6139.3 seed the finite apartness-reflection route from regular Cauchy windows through dyadic positive-gap evidence, located-real handoff, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyApartnessReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient stream equality, selected limit representatives, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking for $\mathsf{RegularCauchyApartnessReflector}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, window and readback exactness, apartness-row exposure, located-real handoff, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyApartnes

Lean4 (statement) `BEDC.Derived.RegularCauchyApartnessReflectorUp`

Chapter 6140

A Concrete Naming Certificate for $\mathsf{MonotoneSequenceWindowCauchy}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{MonotoneSequenceWindowCauchy}^{\uparrow}$ packet records the finite handoff by which inspected monotone bounded real-sequence windows are read as regular rational Cauchy data before any $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits beside the monotone-real-sequence surface of chapter 1434, the located bounded real-sequence surface of chapter 2183, the stream-name surface of chapter 126, the regular rational sequence surface of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names a finite monotone-window route; it does not import host monotone convergence, ambient real completeness, or selected limits.

Definition 6140.1 (Monotone sequence window Cauchy carrier)**.** *A* $\mathsf{MonotoneSequenceWindowCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{MonotoneRealSequence}^{\uparrow}$ *window row, displaying the finite monotonicity comparisons available to the packet.*
2. *B is the* $\mathsf{BoundedRealSequence}^{\uparrow}$ *row, displaying the inspected boundedness interval for the same windows.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, naming the finite stream positions and readback order consumed by the handoff.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row produced from the displayed monotone, bounded, and stream-name data.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic rational tolerances used to witness the regular Cauchy readback.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, B, S, R, D have all been displayed.*
7. *H records componentwise* hsame *transport for M, B, S, R, D, E.*
8. *C records displayed* Cont *replay from monotone windows through boundedness, stream naming, dyadic tolerances, regular readback, and the Real seal.*

9. *P is the* Pkg *provenance over* MonotoneSequenceWindowCauchy$^{\uparrow}$, MonotoneRealSequence$^{\uparrow}$, BoundedRealSequenc StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{MonotoneSequenceWindowCauchy}^{\uparrow}}$ *row.*

The carrier has no coordinate for trichotomy, an uninspected supremum, countable choice, quotient equality, selected limits, or ambient Real$^{\uparrow}$ *completeness outside the displayed rows.*

Lean4 ✓ `BEDC.Derived.MonotoneSequenceWindowCauchyUp.MonotoneSequenceWindowCauchyTasteGate`
`single carrier alignment`

Theorem 6140.2 (Monotone-window regular Cauchy handoff)**.** *For every accepted* MonotoneSequenceWindowCauc *carrier $W = (M, B, S, R, D, E, H, C, P, N)$, monotone bounded finite windows determine the* RegSeqRat$^{\uparrow}$ *handoff only through displayed dyadic tolerance rows. More precisely, every consumer of R must first display the monotone comparison row M, the boundedness row B, the stream-name row S, and the dyadic tolerance row D over the same finite* BHist *packet.*

Proof. Project the carrier of Definition 6140.1. The rows M and B provide only finite monotonicity and boundedness observations, and S fixes the inspected stream positions on which those observations are read.

The handoff row R is not an independent completion theorem. It is a displayed regular rational readback whose tolerance witnesses are recorded in D. Thus a consumer cannot reach R from monotonicity alone, boundedness alone, or an uninspected limit claim.

The structural rows H, C, P, N preserve the same packet through transport, replay, provenance, and local naming. Therefore the regular Cauchy handoff is forced to pass through M, B, S, D before R is available. □

Theorem 6140.3 (Monotone-window Real seal nonescape)**.** *For every accepted* MonotoneSequenceWindowCauchy$^{\uparrow}$ *carrier $W = (M, B, S, R, D, E, H, C, P, N)$, the* Real$^{\uparrow}$ *seal row E is reached only after the monotone-window row M, boundedness row B,* StreamName$^{\uparrow}$ *row S,* RegSeqRat$^{\uparrow}$ *row R, and* DyadicRatCore$^{\uparrow}$ *row D have been displayed in the same packet.*

Proof. Read the seal coordinate in Definition 6140.1. The definition places E downstream from the monotone-window, boundedness, stream-name, regular rational, and dyadic tolerance rows.

Since P sources the packet through those sibling surfaces and C replays their displayed route, no consumer can interpret E as an ambient completeness principle or as a selected limit. The Real seal names the terminal row of the finite route and nothing outside it. □

Closure status for NameCert$_{\text{MonotoneSequenceWindowCauchy}^{\uparrow}}$.

Constructive story: A MonotoneSequenceWindowCauchy$^{\uparrow}$ packet is generated from monotone real-sequence windows, boundedness rows, stream-name rows, RegSeqRat handoff, dyadic tolerance rows, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: TheoryGate(NameCert$_{\text{MonotoneSequenceWindowCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 6140.1, Definition 6140.2, and Definition 6140.3 seed the finite monotone-window Cauchy packet over monotone windows, boundedness, stream naming, dyadic tolerances, regular readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{MonotoneSequenceWindowCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close trichotomy, selected limits, countable choice, quotient equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking for MonotoneSequenceWindowCauchy$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, monotone-window exactness, boundedness display, stream-name compatibility, dyadic tolerance handoff, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.MonotoneSequenceWindowCauchyUp.

Chapter 6141

A Concrete Naming Certificate for $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$ packet records the finite comparison-net data needed by the Real and Completion route before any consumer asks for a completed real object. It sits between the explicit schedules of chapter 292, the regular-Cauchy diagonal route of chapter 3782, the finite stream names, the dyadic rational tolerance ledger, the regular rational readback row, and the terminal $\mathsf{Real}^{\uparrow}$ seal. The packet names a comparison surface for regular Cauchy schedules; it does not identify quotient streams, choose limits, or invoke ambient completeness.

Definition 6141.1 (Regular Cauchy comparison net carrier)**.** *A* $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (M, W, D, Q, E, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row for the compared regular Cauchy requests.*
2. W *is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row opened by the schedules in* M*.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to compare window values.*
4. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the scheduled dyadic comparisons.*
5. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after* M, W, D, Q *have been displayed.*
6. H *records componentwise* hsame *transport for* M, W, D, Q, E, C, P, N*.*
7. C *records displayed* Cont *replay from schedule requests through stream windows, dyadic tolerances, regular readback, and the Real seal.*
8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and the local naming row.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonNet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, W, D, Q, E, H, C, P, N$*. It has no coordinate for quotient stream equality, selected limits, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyComparisonNetUp.RegularCauchyComparisonNetTasteGate single carrier alignment`

Theorem 6141.2 (Regular Cauchy comparison net NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$ *carrier*

$$G = (M, W, D, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonNet}^{\uparrow}}$ *surface consists exactly of schedule admission for the displayed* $\mathsf{CauchyModulus}^{\uparrow}$ *rows, finite* $\mathsf{StreamName}^{\uparrow}$ *window exposure,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance compatibility,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback compatibility,* $\mathsf{Real}^{\uparrow}$ *seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate cannot export quotient stream equality, a selected limit, countable choice, host real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 6141.1. The comparison data are precisely M, W, D, Q, E, and the structural rows are H, C, P, N.

The schedule row M opens only finite stream windows in W. The dyadic row D records the tolerance ledger for those windows, and Q reads the same data as regular rational sequence evidence before the terminal seal E is available.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of the displayed rows contains a quotient stream, a selected limit, a choice schedule, host real equality, or an ambient completeness principle, the listed obligations exhaust the local certificate surface. □

Theorem 6141.3 (Regular Cauchy comparison net Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$ *packet must compare regular Cauchy schedules through the finite route*

$$M, W, D, Q, E, H, C, P, N.$$

The handoff sends schedule-family comparisons through finite $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledgers,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It may be consumed by later Real and Completion packets as comparison data, but it cannot be reinterpreted as quotient stream equality, a selected limit, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the NameCert obligations. They force every accepted comparison to expose the schedule row M before a stream window in W is inspected, and they force the dyadic tolerance row D before regular readback Q reaches the Real seal E.

The regular-Cauchy diagonal route of chapter 3782 can therefore consume the packet only as finite comparison data. The single-modulus surface of chapter 292 supplies schedule discipline, while this packet records how several such schedules are compared at the Real-facing boundary.

Structural replay through H, C, P, N preserves that route. No additional row supplies quotient equality, selected limits, countable choice, host real equality, or ambient completeness, so the handoff remains a finite comparison-net packet for Real and Completion consumers. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonNet}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyComparisonNet}^{\uparrow}$ packet is generated from CauchyModulus schedule rows, finite StreamName windows, DyadicRatCore tolerance ledgers, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyComparisonNet}^{\uparrow}}$, seed)

Scope closed: Definition 6141.1, Definition 6141.2, and Definition 6141.3 seed the finite comparison-net surface for Real and Completion consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{RegularCauchyComparisonNet}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, schedule comparison, window routing, dyadic tolerance compatibility, regular readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyComparisonNetUp.

Chapter 6142

A Concrete Naming Certificate for $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$

The $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ packet realizes the boundary-fixedness-not-center rule of Definition 9463.7 as finite NameCert data. It sits beside the Poincare disk dynamics packet of chapter 4117, the hyperbolic geodesic boundary packet of chapter 3491, the hyperbolic visual boundary packet of chapter 4181, the apophatic far-end socket packet, and the circle boundary reading. The packet names a displayed boundary far-end refusal; it does not identify the far end with an interior disk center or with a hidden global object.

Definition 6142.1 (Boundary far-end noncenter carrier)**.** *A* $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, B, F, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the disk/readout row, recording the visible disk surface and the inspected readout whose center interpretation is being tested.*
2. *B is the boundary far-end row, displaying the boundary-side far-end reading rather than an interior point.*
3. *F is the fixedness row, recording that fixedness is read at the boundary far end and not at the disk center.*
4. *A is the apophatic refusal row, refusing any consumer that converts boundary far-end data into center data or object-level* T*.*
5. *H records componentwise* hsame *transport for D, B, F, A.*
6. *C records displayed* Cont *replay from disk readout through boundary far-end fixedness and apophatic refusal.*
7. *P is the* Pkg *provenance over* $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$*, the hyperbolic boundary route, apophatic far-end discipline,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundaryFarEndNoncenter}^{\uparrow}}$ *row.*

The carrier has no coordinate for a privileged disk center, a selected ideal point, a quotient boundary equality, subject-substance, or object-level T *outside the displayed refusal row.*

Theorem 6142.2 (Boundary far-end noncenter obligations)**.** *For every accepted* $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ *carrier* $Q = (D, B, F, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundaryFarEndNoncenter}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed disk/readout, boundary far-end, fixedness, apophatic refusal, transport, replay, provenance, and local naming rows; fixedness exactness through* F*; refusal visibility through* A*; and noncenter refusal for any read that treats* D *as carrying the boundary far end through an interior center.*

Proof. Project the carrier of Definition 6142.1. The disk/readout row D names only the visible surface and inspected readout, while B names the boundary far-end row.

The fixedness row F is tied to B, not to an interior center coordinate in D. The refusal row A is therefore part of the certificate surface: a consumer that reads fixedness as center data has not displayed the accepted packet.

The rows H, C, P, N transport, replay, source, and name the same finite data. Thus the listed rows exhaust the local NameCert obligations, and no hidden center, selected ideal point, quotient boundary equality, subject-substance, or object-level T can enter the accepted route. □

Theorem 6142.3 (Boundary far-end fixedness noncenter rule)**.** *For every accepted* $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ *carrier* $Q = (D, B, F, A, H, C, P, N)$*, boundary fixedness is read only from the displayed boundary far-end row* B *together with the fixedness row* F*. Any disk-center reading of* D *that bypasses the apophatic refusal row* A *is not a consumer of the same packet.*

Proof. The carrier separates the visible disk/readout row D from the boundary far-end row B. The fixedness row F is attached to B, while A records the refusal of center collapse.

If a consumer bypasses A, it has removed the row that distinguishes boundary far-end data from center data. If it reads D as the fixed far end, it has replaced the boundary row by an interior readout. In both cases the displayed packet is not preserved, so the consumer is outside $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$. □

Falsifiable prediction: Every accepted $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ consumer must display the boundary far-end row and apophatic refusal row before any disk-center read is rejected as noncenter data.

Independence witness: poincare-disk-dynamics, hyperbolic-geodesic-boundary, hyperbolic-visual-boundary, apophaticfarendsocket, and s1: none carries the same boundary/noncenter/refusal rows. The dynamics packet exposes disk-map evolution, the geodesic and visual boundary packets expose boundary routes, the far-end socket exposes apophatic pressure, and s1 exposes a circle boundary shape; only $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ binds disk readout, boundary far-end fixedness, noncenter refusal, transport, replay, provenance, and local naming in one packet.

The Lean follow-up file `lean4/BEDC/Derived/BoundaryFarEndNoncenterUp/TasteGate.lean` must provide explicit $\mathsf{Nontrivial}\ \mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ instances preserving D, B, F, A, H, C, P, N before any checked carrier-alignment theorem can serve as a TasteGate realization.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundaryFarEndNoncenter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$ packet is generated from disk/readout rows, boundary far-end rows, boundary fixedness rows, apophatic refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundaryFarEndNoncenter}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6142.1, Definition 6142.2, Definition 6142.3, the falsifiable prediction, and the independence witness seed the finite boundary-far-end noncenter packet over disk readout, boundary far-end fixedness, refusal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BoundaryFarEndNoncenter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close privileged disk centers, selected ideal points, quotient boundary equality, subject-substance, object-level T, Lean verification, or bridge checking for BoundaryFarEndNoncenter$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, boundary fixedness exactness, apophatic refusal visibility, noncenter refusal, transport, replay, provenance, local naming, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.BoundaryFarEndNoncenterUp.

Chapter 6143

A Concrete Naming Certificate for $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$

Following Bishop and Bridges and the constructive Heine–Cantor route, the $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$ packet packages the finite radius-transport step used by chapter 9237. It sits beside the compact source packet of chapter 60, the metric distance surface of chapter 57, the continuous-map carrier of chapter 130, the rational radius ledger of chapter 37, and the uniform-modulus consumer of chapter 369. The packet names a finite transport from pointwise compact-cover radii to a tolerance-only modulus ledger; it does not choose a global modulus by countable choice or by an ambient compactness principle.

Definition 6143.1 (Compact uniform radius transport carrier)**.** *A* $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, M, F, R, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{Compact}^{\uparrow}$ *finite-net row, including the bundled compact-cover witnesses used at the requested tolerance.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *distance row for the source and target metric comparisons consumed by the transport.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row attached to the compact finite-net centers.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *positive-radius and lower-bound fold ledger for the finitely many displayed radii.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row reached only after K, M, F, R have been displayed.*
6. *H records componentwise* hsame *transport for K, M, F, R, U, C, P, N.*
7. *C records displayed* Cont *replay from compact finite-net coverage through pointwise radii, rational lower-bound folding, metric triangle use, and the uniform-modulus handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and the local naming row.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusTransport}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $K, M, F, R, U, H, C, P, N$. It has no coordinate for an external modulus selector, an infinite cover, countable choice, open-cover compactness outside the displayed finite net, host real equality, or ambient Real completeness.

Theorem 6143.2 (Compact uniform radius transport NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$ *carrier*

$$T = (K, M, F, R, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusTransport}^{\uparrow}}$ *surface consists exactly of compact finite-net admission, metric-distance compatibility, continuous-map pointwise-radius admission, rational lower-bound fold compatibility, uniform-modulus handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate cannot export an external modulus selector, infinite cover, countable choice, open-cover compactness detached from* K*, host real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 6143.1. The compact-cover and analytic rows are K, M, F, R, U, while H, C, P, N are the structural rows that preserve the same displayed transport.

The compact finite-net row K exposes only finitely many centers and coverage witnesses. The continuous-map row F supplies pointwise radii at those displayed centers, and the rational ledger R folds the finitely many positive radii before the uniform-modulus row U is available.

Metric compatibility is read through M along the same route. Since no carrier coordinate names an external selector, infinite cover, countable-choice schedule, detached open-cover compactness theorem, host real equality, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CompactUniformRadiusTransportUp.CompactUniformRadiusTransportTasteGate`
`single carrier alignment`

Theorem 6143.3 (Compact uniform radius transport handoff)**.** *Every uniform-modulus consumer of an accepted* $\mathsf{CompactUniformRadiusTransport}^{\uparrow}$ *packet receives the finite route*

$$K, M, F, R, U, H, C, P, N.$$

The handoff transports pointwise compact-cover radii through the displayed compact finite net, source and target metric rows, continuous-map local-modulus data, and $\mathsf{Rat}^{\uparrow}$ *lower-bound folding before exposing the* $\mathsf{UniformModulus}^{\uparrow}$ *row. It may be used as the radius-transport step in compact-uniform-continuity arguments, but it cannot be reinterpreted as an external modulus selector, an infinite cover, countable choice, detached open-cover compactness, host real equality, or ambient Real completeness.*

Proof. Apply the NameCert obligation theorem. It forces the compact finite-net row K before any pointwise radius can be consumed, and it forces the continuous-map row F and rational lower-bound row R before the uniform-modulus handoff U is exposed.

The compact-uniform-continuity vision of chapter 9237 uses exactly this order: finite compact-cover data first, pointwise local radii second, rational lower-bound folding third, and the tolerance-only uniform-modulus row last. Metric rows in M supply the triangle comparisons without adding a host compactness theorem.

Finally replay H, C, P, N. These rows preserve the displayed finite transport and do not create any coordinate for external selection, infinite covers, countable choice, detached open-cover compactness, host equality, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformRadiusTransport}^{\uparrow}}$.

Constructive story: A CompactUniformRadiusTransport$^{\uparrow}$ packet is generated from Compact finite-net rows, MetricSpace distance rows, ContinuousMap pointwise-radius rows, Rat lower-bound folds, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactUniformRadiusTransport}\uparrow}$, seed)

Scope closed: Definition 6143.1, Definition 6143.2, and Definition 6143.3 seed the finite radius-transport surface from compact pointwise radii to UniformModulus consumers.

Formal verification: FormalStatus(NameCert$_{\text{CompactUniformRadiusTransport}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close external modulus selection, infinite covers, countable choice, open-cover compactness outside the displayed finite net, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net routing, metric compatibility, continuous-map pointwise-radius admission, rational lower-bound folding, uniform-modulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformRadiusTransportUp.

Chapter 6144

A Concrete Naming Certificate for $\mathsf{NemytskiiOperator}^{\uparrow}$

Following Krasnoselskii and Rutickii on superposition operators, the $\mathsf{NemytskiiOperator}^{\uparrow}$ packet records the finite NameCert surface for nonlinear pointwise composition on named function spaces. It sits over the measurable-map surface of chapter 3461, the Sobolev-space carrier of chapter 5911, the complete normed-space interface of **??**, the measure rows of chapter 92, and the terminal real-name seal of chapter 38. The packet names the displayed superposition route for a nonlinear kernel and its source-function windows; it does not assert a host function quotient, global continuity theorem, PDE solvability result, selected representative, or ambient Banach-space completeness theorem.

Definition 6144.1 (Nemytskii operator carrier)**.** *A* $\mathsf{NemytskiiOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (D, M, F, W, I, T_0, S, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain and codomain row, naming the measurable source, target scalar seal, and function-space boundary inspected by the packet.*
2. *M is the* $\mathsf{MeasurableFunction}^{\uparrow}$ *row for the nonlinear kernel, exposing only displayed preimage ledgers and graph-read windows.*
3. *F is the source-function row, listing the finite function-name windows, domain probes, and scalar readbacks that may be substituted into the kernel.*
4. *W is the pointwise witness row, pairing each displayed source observation with the corresponding displayed kernel observation before any output function row is read.*
5. *I is the integrability and growth-budget row, binding the admissible source windows to the finite measure and Real rows used by the superposition request.*
6. *T_0 is the target-function row produced by the displayed superposition route, not by a host quotient of functions.*
7. *S is the* $\mathsf{SobolevSpace}^{\uparrow}$ *or function-space handoff row, available only after D, M, F, W, I, T_0 have been displayed.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *readback row for scalar kernel values, growth budgets, norm estimates, and target-function observations.*

9. *H stores componentwise* hsame *transport for domain, measurable-kernel, source-function, witness, integrability, target-function, function-space, Real, replay, provenance, and local naming rows.*
10. *C stores displayed* Cont *replay from every superposition request through $D, M, F, W, I, T_0, S, R, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{NemytskiiOperator}^{\uparrow}$, $\mathsf{MeasurableFunction}^{\uparrow}$, $\mathsf{SobolevSpace}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{NemytskiiOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for an ambient nonlinear operator theorem, quotient equality of functions, almost-everywhere representative choice, global compactness, PDE solvability, or an external Banach-space completeness claim.

Lean4 (statement) `BEDC.Derived.NemytskiiOperatorUp`

Theorem 6144.2 (Nemytskii operator NameCert obligations)**.** *For every accepted* $\mathsf{NemytskiiOperator}^{\uparrow}$ *carrier*

$$T = (D, M, F, W, I, T_0, S, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{NemytskiiOperator}^{\uparrow}}$ *surface consists exactly of carrier admission for the domain row D, measurable-kernel row M, source-function row F, pointwise witness row W, integrability and growth row I, target-function row T_0, function-space handoff S, Real readback R, componentwise transport H, displayed replay C, provenance P, and local naming N. Every admitted nonlinear composition read must expose the kernel row, source windows, pointwise witness row, growth budget, target-function row, and Real seal before a Sobolev, Banach, measure, or Real consumer can read the output.*

Proof. Project the carrier of Definition 6144.1. The only analytic data available to the classifier are the displayed domain row D, measurable-kernel row M, source-function row F, pointwise witness row W, integrability row I, target row T_0, function-space handoff S, and Real row R.

The measurable-kernel obligation is read before any output claim. The preimage and graph-read surface of M fixes which nonlinear kernel observations can be paired with the source-function windows in F.

The pointwise witness and budget rows then bind the superposition route. The rows W, I, T_0, R record the displayed source-to-target observation pairs, growth and integrability budgets, target-function output, and scalar readbacks.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since the excluded host objects are not coordinates of the carrier, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.NemytskiiOperatorUp.NemytskiiOperatorNamecertObligations`

Theorem 6144.3 (Nemytskii operator measurable handoff)**.** *Every measurable-output read exported by an accepted* $\mathsf{NemytskiiOperator}^{\uparrow}$ *carrier factors through the finite route*

$$M, F, W, I \longmapsto T_0 \longmapsto R \longmapsto H, C, P, N.$$

The route exposes a measurable-kernel ledger, source-function windows, displayed pointwise substitutions, growth and integrability budgets, target-function rows, Real readbacks, transport, replay, provenance, and local naming. It does not export a host inverse-image function, almost-everywhere quotient, selected representative, global continuity theorem, or completed measure space.

Proof. Start with a measurable-output request. The request first reads the measurable-kernel row M and the source-function row F, because the output is determined only by the displayed kernel observations paired with displayed source observations.

Next the route inspects W and I. These rows bind pointwise substitution and admissible growth or integrability budgets to the same finite windows, and the target-function row T_0 is read only after those rows are fixed.

The Real readback and structural rows close the route. The scalar seal R and the rows H, C, P, N preserve the same packet, so no host inverse-image function, representative selector, quotient, continuity theorem, or measure completion is exported. □

Lean4 ✓ `BEDC.Derived.NemytskiiOperatorUp.NemytskiiOperatorMeasurableHandoff`

Theorem 6144.4 (Nemytskii operator Banach–Sobolev consumer)**.** *Every Banach–Sobolev consumer of an accepted* $\mathsf{NemytskiiOperator}^{\uparrow}$ *carrier must read the displayed route*

$$\mathsf{MeasurableFunction}^{\uparrow},\ \mathsf{Measure}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{SobolevSpace}^{\uparrow},\ \mathsf{BanachSpace}^{\uparrow}$$

through the carrier rows M, I, R, S and the source-to-target rows F, W, T_0 before it may consume transport, replay, provenance, or local naming. The handoff supplies a finite nonlinear composition interface; it does not assert a compact embedding, a global boundedness theorem, PDE solvability, quotient equality of functions, countable choice, or an ambient completeness theorem.

Proof. Use the NameCert obligations of Definition 6144.2. They force every consumer to begin with the measurable kernel and source-function rows, then pass through the pointwise witness and growth-budget rows.

The Sobolev and Banach reads are downstream of S and of the Real readback row R. Thus any function-space statement consumes only the displayed target-function row T_0, the finite budget row I, and the sibling certificates named in the provenance row.

Transport, replay, provenance, and local naming through H, C, P, N keep the same finite route. Since no compact embedding, selected representative, quotient equality, choice schedule, PDE theorem, or ambient completeness coordinate appears in the carrier, every Banach–Sobolev consumer is exhausted by the displayed handoff. □

Falsifiable prediction: An accepted $\mathsf{NemytskiiOperator}^{\uparrow}$ consumer that reaches a Sobolev or Banach read without first exposing measurable-kernel rows, source-function windows, pointwise witness rows, growth budgets, target-function rows, Real readbacks, componentwise hsame transport, Cont replay, Pkg provenance, and local naming refutes the carrier surface of Definition 6144.1.

Independence witness: measurable-function, sobolev-space, banach-space, measure, real: no listed sibling supplies the whole twelve-row packet $T = (D, M, F, W, I, T_0, S, R, H, C, P, N)$, because each sibling contributes only kernel measurability, function-space, normed-completion, measure, or scalar-seal data while this chapter binds them into one nonlinear superposition carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{NemytskiiOperator}^{\uparrow}}$**.**

Constructive story: A $\mathsf{NemytskiiOperator}^{\uparrow}$ packet is generated from a displayed domain row, MeasurableFunction kernel row, source-function windows, pointwise witness rows, growth and integrability budgets, target-function rows, Sobolev or function-space handoff data, Real readbacks, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NemytskiiOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6144.1, Definition 6144.2, Definition 6144.3, and Definition 6144.4 seed the finite nonlinear superposition surface over displayed measurable kernels, source windows, pointwise witnesses, growth budgets, target rows, function-space handoffs, Real readbacks, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{NemytskiiOperator}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global continuity, compact embedding, PDE solvability, almost-everywhere quotient equality, selected representatives, countable choice, ambient Banach-space completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, measurable-kernel handoff, pointwise witness exactness, growth-budget routing, Banach–Sobolev consumer nonescape, transport, replay, provenance, and local naming for BEDC.Derived.NemytskiiOperatorUp.

Chapter 6145

A Concrete Naming Certificate for $\mathsf{FiniteLipschitzIteration}^{\uparrow}$

Following Banach and Picard, the $\mathsf{FiniteLipschitzIteration}^{\uparrow}$ packet records the finite contraction-budget route by which a displayed initial point, a Lipschitz self-map, and a dyadic tolerance schedule generate an explicitly bounded Picard-prefix ledger. It sits over the metric source of chapter 57, the continuous-map row of chapter 130, the real-name seal of chapter 38, the Cauchy-modulus window of chapter 292, and the Picard contraction surface of chapter 395. The packet names finite iteration data only; it does not select a fixed point, assert ambient completeness, or export a completed limit.

Definition 6145.1 (Finite Lipschitz iteration carrier)**.** *A* $\mathsf{FiniteLipschitzIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (X, d, f, L, q, n, x_0, W, B, H, C, P, N)$$

with the following displayed rows:

1. X, d *are the* $\mathsf{Met}^{\uparrow}$ *source and distance rows.*
2. f *is the* $\mathsf{ContinuousMap}^{\uparrow}$ *self-map row over the same metric source.*
3. L *is the Lipschitz-constant row, read through the same map and source.*
4. q *is the dyadic tolerance row used to compare the finite tail budget.*
5. n *is the displayed finite iteration counter.*
6. x_0 *is the initial-point row.*
7. W *is the finite prefix row listing the displayed reads of* $x_0, f(x_0), \ldots, f^n(x_0)$.
8. B *is the contraction-budget row consumed by* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{PicardContraction}^{\uparrow}$ *consumers.*
9. H *records componentwise* hsame *transport for the metric source, map, Lipschitz constant, dyadic tolerance, counter, prefix, and budget rows.*
10. C *records displayed* Cont *replay through the metric source, continuous-map row, Lipschitz row, finite prefix, and budget row.*
11. P *is the* Pkg *provenance over* $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{PicardContraction}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

12. N is the local $\mathsf{NameCert}_{\mathsf{FiniteLipschitzIteration}^{\uparrow}}$ *row.*
The carrier compares finite prefixes only through these rows. It has no coordinate for a selected fixed point, quotient orbit, infinite limit, host completeness theorem, or ambient contraction principle.

Lean4 ✓ `BEDC.Derived.FiniteLipschitzIterationUp.FiniteLipschitzIterationTasteGate single carrier alignment`

Theorem 6145.2 (Finite Lipschitz iteration NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLipschitzIteration}^{\uparrow}$ *carrier* $I = (X, d, f, L, q, n, x_0, W, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteLipschitzIteration}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, continuous self-map, Lipschitz constant, dyadic tolerance, finite counter, initial point, finite prefix, contraction budget, transport, replay, provenance, and local naming; exactness of the displayed prefix W at counter n; budget compatibility with the same X, d, f, L, q rows; and nonescape excluding fixed-point selection, ambient completeness, quotient orbit equality, and host-level limiting.*

Proof. Project the accepted carrier. The displayed rows are precisely $X, d, f, L, q, n, x_0, W, B, H, C, P, N$, so carrier admission and local naming are direct reads from Definition 6145.1.

The exactness of W is finite. At counter 0, the prefix row exposes only the initial point x_0. At a successor counter, the next row is obtained by replaying the same continuous-map row f over the same metric source and appending that displayed read to the previous prefix.

The budget row B is attached to the same metric source, continuous-map row, Lipschitz constant row, and dyadic tolerance row. Transport through H and replay through C preserve those rows before Pkg provenance and the local NameCert expose the packet.

Since the carrier contains no selected fixed point, infinite orbit, quotient equality, host limit, or completeness coordinate, the listed rows exhaust the local obligation surface. □

Theorem 6145.3 (Finite Lipschitz contraction-budget induction)**.** *For every accepted* $\mathsf{FiniteLipschitzIteration}^{\uparrow}$ *carrier and every displayed finite counter n, the contraction-budget row B is obtained by induction on n: the base prefix reads the initial point over the metric source, and each successor step consumes the same metric source, continuous-map row, Lipschitz constant row, dyadic tolerance row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}$ *row. The induction exports only a finite prefix budget for* $\mathsf{CauchyModulus}^{\uparrow}$ *and* $\mathsf{PicardContraction}^{\uparrow}$ *consumers.*

Proof. Induct on the displayed counter n.

In the base case, project the carrier and read x_0 as the whole prefix. The metric source X, d, tolerance q, transport, replay, provenance, and local naming rows are present, but no application of f has yet been consumed.

For the successor case, assume the previous prefix budget has been displayed over the same rows. Replay the continuous-map row f through C, transport the metric source, Lipschitz constant, and tolerance through H, and append the next displayed read to W. The budget row B is therefore updated by the same finite rows rather than by a new source, a hidden map, or an external limit.

The induction stops at the displayed counter. It supplies a finite contraction-budget handoff to $\mathsf{CauchyModulus}^{\uparrow}$ and $\mathsf{PicardContraction}^{\uparrow}$, and it does not assert convergence, completeness, or fixed-point existence. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLipschitzIteration}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteLipschitzIteration}^{\uparrow}$ packet is generated from a Metric source, ContinuousMap self-map, Lipschitz constant, dyadic tolerance, finite counter, initial point, finite prefix, contraction budget, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLipschitzIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6145.1, Definition 6145.2, and Definition 6145.3 seed the finite contraction-prefix budget route over metric, map, Lipschitz, tolerance, counter, prefix, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteLipschitzIteration}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close fixed-point existence, selected limits, ambient completeness, quotient orbit equality, Lean verification, or bridge checking for FiniteLipschitzIteration$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix exactness, contraction-budget induction, nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLipschitzIterationUp.

Chapter 6146

A Concrete Naming Certificate for $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$

The $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ packet realizes the reversible phase-surface transport direction of chapter 9463 as finite object, arrow, inverse, composition, boundary-refusal, provenance, and local naming rows. It consumes the visible phase surface of chapter 39, reversible transport through chapter 219, finite flow-style replay through chapter 195, the observer-expansion route of chapter 4167, and the boundary-far-end refusal of chapter 6142. The packet is a finite transport groupoid surface, not a host groupoid of all observers and not a completed phase space.

Definition 6146.1 (Observer phase groupoid carrier)**.** *An* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (O, A, S, T, I, M, B, H, C, P, N)$$

with the following displayed rows:

1. *O is the finite object row of observer-side phase surfaces.*
2. *A is the finite arrow row of reversible phase transports between displayed objects.*
3. *S is the source-target incidence row tying each arrow to two displayed phase objects.*
4. *T is the reversible transport row, read through the* $\mathsf{UnitaryGroup}^{\uparrow}$ *and* $\mathsf{DynSystem}^{\uparrow}$ *surfaces.*
5. *I is the inverse row, giving the displayed reverse transport for every accepted arrow.*
6. *M is the composition row, giving the displayed composite when source and target rows match.*
7. *B is the boundary-refusal row, excluding an interior center, a global observer object, and a boundary far-end treated as a phase object.*
8. *H records componentwise* hsame *transport for object, arrow, incidence, transport, inverse, composition, and boundary-refusal rows.*
9. *C records displayed* Cont *replay through phase objects, reversible transport, inverse, composition, and refusal reads.*
10. *P is the* Pkg *provenance over* $\mathsf{Complex}^{\uparrow}$, $\mathsf{UnitaryGroup}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{HyperbolicObserverExpansion}^{\uparrow}$, $\mathsf{BoundaryFarEndNoncenter}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ObserverPhaseGroupoid}^{\uparrow}}$ *row.*

The classifier compares arrows only by the joint object-arrow-incidence, inverse, composition, refusal, transport, replay, provenance, and local naming surface. It contains no host phase groupoid, quotient observer identity, global time, or completed hyperbolic phase space.

Theorem 6146.2 (Observer phase groupoid NameCert obligations)**.** *For every accepted* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *carrier* $G = (O, A, S, T, I, M, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ObserverPhaseGroupoid}^{\uparrow}}$ *surface consists exactly of object admission, arrow admission, source-target incidence, reversible transport, inverse, composition, boundary refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every accepted arrow factors through displayed phase object rows, reversible transport, inverse, composition, boundary refusal, hsame, Cont, Pkg, and NameCert rows.*

Proof. Project the carrier of Definition 6146.1. The object and arrow rows are O and A, while S records the finite source-target incidence that makes an arrow readable.

The transport, inverse, and composition rows are displayed separately as T, I, M. They are therefore part of the same finite surface rather than hidden operations on a host groupoid.

Boundary refusal is row B. It prevents the far-end boundary and observer-side focus from becoming ordinary phase objects, so accepted arrows remain reversible display transports rather than global observer identities.

Finally, H, C, P, N transport, replay, source, and name the same rows. No further row remains, so the listed obligations exhaust the local NameCert surface. □

Theorem 6146.3 (Observer phase groupoid transport composition)**.** *Whenever two accepted* $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ *arrows have matching displayed incidence rows, their composite is the row supplied by* M *after reading the same object surface* O*, arrow surface* A*, reversible transport row* T*, inverse row* I*, boundary-refusal row* B*,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. The composite exports no host observer equality, quotient phase object, global clock, or completed hyperbolic geometry.*

Proof. Read the two arrows through A and their source-target incidence through S. The matching condition is therefore a displayed row comparison, not a host equality of observers.

Use the reversible transport and inverse rows T, I to keep each arrow inside the accepted phase-transport surface. The composition row M is then the only row allowed to expose the composite.

Transport through H and replay through C preserve the same object, arrow, inverse, composition, and refusal rows. Provenance and local naming through P, N certify that the composite remains in the finite packet.

Since no quotient phase object, host observer equality, global clock, or completed hyperbolic geometry is present in the carrier, composition remains a finite BEDC row. □

Falsifiable prediction: Every accepted $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ arrow factors through displayed phase object rows, reversible transport, inverse, composition, boundary refusal, hsame, Cont, Pkg, and NameCert rows.

Independence witness: complex, unitarygroup, dynsystem, hyperbolic-observer-expansion, boundary-far-end-noncenter: none is bijective to $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ because the groupoid packet jointly carries object rows, arrow rows, incidence, reversible transport, inverse, composition, boundary refusal, hsame transport, Cont replay, provenance, and local naming.

The Lean follow-up for this packet provides explicit $\mathsf{Nontrivial}\ \mathsf{ObserverPhaseGroupoid}^{\uparrow}$ and $\mathsf{FieldFaithful}\ \mathsf{ObserverPhaseGroupoid}^{\uparrow}$ TasteGate instances in BEDC.Derived.ObserverPhaseGroupoidUp.TasteGate. The paper-side chapter remains a finite NameCert target; bridge checking is a separate closure axis.

Closure status for $\mathsf{NameCert}_{\mathsf{ObserverPhaseGroupoid}^{\uparrow}}$.

Constructive story: An $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$ packet is generated from finite phase-object rows, arrow rows, source-target incidence, reversible transport, inverse, composition, boundary refusal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ObserverPhaseGroupoid}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6146.1, Definition 6146.2, and Definition 6146.3 seed the finite reversible phase groupoid route over objects, arrows, incidence, inverse, composition, boundary refusal, transport, replay, provenance, and naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ObserverPhaseGroupoid}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a host observer groupoid, quotient phase object, global clock, completed hyperbolic phase space, Lean verification of the paper obligations, or bridge checking for $\mathsf{ObserverPhaseGroupoid}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, object-arrow incidence, reversible transport, inverse, composition, boundary refusal, TasteGate nontriviality, field faithfulness, and local naming for BEDC.Derived.ObserverPhaseGroupoidUp

Chapter 6147

A Concrete Naming Certificate for PompeiuHausdorffDistanceRealizer$^{\uparrow}$

Following Pompeiu, Hausdorff, and Kelley on compact hyperspaces, the PompeiuHausdorffDistanceRealizer$^{\uparrow}$ packet records the finite NameCert surface that turns two displayed nonempty compact-subset rows into the symmetric compact-set distance read consumed by fixed-point and compact-hyperspace arguments. It sits after the host finite ledger of chapter 1660, the scalar Hausdorff distance row of chapter 3653, the finite-net HausdorffMetric interface of chapter 2086, the compact metric source of chapter 129, the diameter tolerance ledger of chapter 1503, and the Vietoris neighbourhood route of chapter 2375. The packet is a compact-subset metric realizer: it names only displayed subset rows, directed finite-cover ledgers, symmetric readback, Vietoris compatibility, transport, replay, provenance, and local naming.

Definition 6147.1 (Pompeiu-Hausdorff distance realizer carrier)**.** *A* PompeiuHausdorffDistanceRealizer$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, B, H, D_0, D_1, S, V, T, C, P, N)$$

with the following displayed rows:

1. *X is the shared* CompactMetric$^{\uparrow}$ *source row, including the metric observations, finite-net schedules, and diameter bound available to compact-subset distance consumers.*
2. *A and B are the two nonempty compact-subset presentation rows over X.*
3. *H is the* Hyperspace$^{\uparrow}$ *finite Hausdorff ledger row that keeps the two subset presentations attached to the same compact metric source.*
4. *D_0 is the directed finite-cover comparison from A to B, read through the* HausdorffMetric$^{\uparrow}$ *finite-net interface.*
5. *D_1 is the directed finite-cover comparison from B to A with the same source, tolerance budget, and displayed net discipline.*
6. *S is the* HausdorffDistance$^{\uparrow}$ *scalar seal that packages the two directed comparisons as one symmetric compact-set distance read.*
7. *V is the* VietorisTopology$^{\uparrow}$ *compatibility row, recording that finite Vietoris neighbourhood tests inspect the same subset rows and tolerance budget before a topological consumer may read the distance.*

8. *T records componentwise* hsame *transport for $X, A, B, H, D_0, D_1, S, V, C, P, N$.*
9. *C records displayed* Cont *replay of the route*

$$X, A, B, H \longmapsto D_0, D_1 \longmapsto S, V.$$

10. *P is the* Pkg *provenance over* PompeiuHausdorffDistanceRealizer$^{\uparrow}$, Hyperspace$^{\uparrow}$, HausdorffDistance$^{\uparrow}$, HausdorffMetric$^{\uparrow}$, CompactMetric$^{\uparrow}$, MetricDiameter$^{\uparrow}$, VietorisTopology$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{PompeiuHausdorffDistanceRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, B, H, D_0, D_1, S, V, T, C, P, N$. It has no coordinate for a host powerset, quotient compactum, selected nearest point, open-cover compactness theorem, ambient supremum, or arbitrary Vietoris basis selector.

Lean4 (statement) `BEDC.Derived.PompeiuHausdorffDistanceRealizerUp`

Theorem 6147.2 (Pompeiu-Hausdorff realizer obligations)**.** *For every accepted* PompeiuHausdorffDistanceRealizer$^{\uparrow}$ *carrier*

$$R = (X, A, B, H, D_0, D_1, S, V, T, C, P, N),$$

the local NameCert$_{\text{PompeiuHausdorffDistanceRealizer}^{\uparrow}}$ *surface consists exactly of six finite obligation rows:*

1. *compact-source admission for the shared* CompactMetric$^{\uparrow}$ *row X, including the displayed metric, finite-net, and* MetricDiameter$^{\uparrow}$ *tolerance support;*
2. *compact-subset admission for A and B, requiring both presentations to be read through the same* Hyperspace$^{\uparrow}$ *source ledger H;*
3. *bidirectional finite coverage, requiring D_0 and D_1 to be present before any symmetric compact-set distance is public;*
4. *scalar distance realization, requiring the* HausdorffDistance$^{\uparrow}$ *seal S to be obtained only from the two directed* HausdorffMetric$^{\uparrow}$ *comparisons;*
5. *Vietoris compatibility, requiring V to reuse the same subset rows, source row, and tolerance budget when a topological neighbourhood consumer reads the packet;*
6. *transport, replay, provenance, and local naming exhaustion through T, C, P, N.*

No Pompeiu-Hausdorff realizer obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PompeiuHausdorffDistanceRealizerUp.TasteGate.PompeiuHausdorffDistanceRealizerCarri namecert obligations`

Theorem 6147.3 (Pompeiu-Hausdorff bidirectional coverage)**.** *Every compact-subset metric consumer of an accepted* PompeiuHausdorffDistanceRealizer$^{\uparrow}$ *packet reads the symmetric distance only after both directed finite-cover comparisons have been displayed:*

$$X, A, B, H \longmapsto D_0, D_1 \longmapsto S.$$

The route therefore supplies the same compact-set distance to Hyperspace$^{\uparrow}$, HausdorffDistance$^{\uparrow}$, HausdorffMetric$^{\uparrow}$, *and* VietorisTopology$^{\uparrow}$ *consumers without exporting a host powerset, quotient compactum, selected nearest point, open-cover compactness theorem, or ambient supremum principle.*

Theorem 6147.4 (Pompeiu-Hausdorff nonescape)**.** *For every accepted* PompeiuHausdorffDistanceRealizer$^{\uparrow}$ *carrier, the* MetricDiameter$^{\uparrow}$ *and* VietorisTopology$^{\uparrow}$ *support rows remain support rows for the displayed compact-subset distance route. They cannot export an additional compactness proof, a selected compact subset, a quotient equality of compacta, an arbitrary Vietoris basis selector, or a second scalar distance outside*

$$X, A, B, H, D_0, D_1, S, V, T, C, P, N.$$

Closure status for NameCert$_{\text{PompeiuHausdorffDistanceRealizer}^{\uparrow}}$.

Constructive story: A PompeiuHausdorffDistanceRealizer$^{\uparrow}$ packet is generated from a CompactMetricUp source row, two nonempty compact-subset rows, a HyperspaceUp ledger, two directed HausdorffMetricUp finite-cover comparisons, a HausdorffDistanceUp scalar seal, VietorisTopologyUp compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{PompeiuHausdorffDistanceRealizer}^{\uparrow}}$, seed)

Scope closed: Definition 6147.1, Definition 6147.2, Definition 6147.3, and Definition 6147.4 seed the compact-subset metric realizer for finite Hausdorff and Vietoris consumers.

Formal verification: FormalStatus(NameCert$_{\text{PompeiuHausdorffDistanceRealizer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host hyperspaces, quotient compacta, selected nearest points, open-cover compactness, ambient suprema, Lean theorem verification, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, compact-subset admission, bidirectional coverage, scalar distance realization, Vietoris compatibility, and support-row nonescape for BEDC.Derived.PompeiuHausdorffDistanceRealizerUp.

Chapter 6148

A Concrete Naming Certificate for CompactIntervalOscillationModulus$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CompactIntervalOscillationModulus$^{\uparrow}$ packet records the finite oscillation-modulus data by which a located compact interval read can hand a uniform tail-tolerance bound to Real-facing consumers. It sits after the located interval surface of chapter 394, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal seal discipline of chapter 38. The packet names an interval-local oscillation ledger and modulus handoff without asserting ambient Real$^{\uparrow}$ completeness, a selected supremum, host interval compactness, or quotient real equality.

Definition 6148.1 (Compact interval oscillation-modulus carrier)**.** *A* CompactIntervalOscillationModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, W, R, Q, O, M, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the* LocatedInterval$^{\uparrow}$ *source row, exposing lower and upper endpoint windows together with the interval-cell ledger used by the oscillation read.*
2. *W is the finite* StreamName$^{\uparrow}$ *window row over the displayed interval cells and endpoint observations.*
3. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the finite stream window.*
4. *Q is the* DyadicRatCore$^{\uparrow}$ *tolerance row that stores the dyadic budget requested by the oscillation test.*
5. *O is the oscillation ledger, recording finite comparisons of the displayed interval reads under the same W, R, Q rows.*
6. *M is the modulus handoff row, giving the finite interval-local threshold returned to downstream Cauchy or compact-interval consumers.*
7. *E is the* Real$^{\uparrow}$ *seal row, reached only after L, W, R, Q, O, M have been accepted.*
8. *H records componentwise* hsame *transport for the interval source, window, readback, tolerance, oscillation ledger, modulus handoff, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from located interval reads through stream windows, regular rational readback, dyadic tolerance, oscillation ledger, modulus handoff, and real sealing.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactIntervalOscillationModulus}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $L, W, R, Q, O, M, E, H, C, P, N$. It has no coordinate for a host compact interval, selected endpoint value, quotient real equality, selected supremum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6148.2 (Compact interval oscillation-modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$ *carrier*

$$I = (L, W, R, Q, O, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactIntervalOscillationModulus}^{\uparrow}}$ *surface consists exactly of located interval admission, finite stream-window exposure, regular rational readback, dyadic tolerance budgeting, oscillation-ledger exactness, modulus handoff, real-seal nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The certificate exports no host compact interval, selected endpoint value, quotient real equality, selected supremum, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6148.1. The analytic rows are L, W, R, Q, O, M, E, while H, C, P, N preserve, replay, source, and name the same finite packet.

The located interval admission row L supplies the interval source and endpoint-window data. The finite window row W and regular readback row R expose the only interval reads available to the oscillation ledger.

The dyadic tolerance row Q fixes the comparison budget before O is read. The oscillation ledger O records only finite comparisons over the same W, R, Q rows, and the modulus handoff M is the only downstream threshold exported by the packet.

Finally the seal row E is downstream of the accepted finite prefix, while H, C, P, N preserve the same route under transport and replay. Since no displayed coordinate names a host interval, selected endpoint, quotient real equality, selected supremum, countable choice, or ambient completeness principle, the listed rows exhaust the local NameCert obligations. □

Lean4 ✓ `BEDC.Derived.CompactIntervalOscillationModulusUp.CompactIntervalOscillationModulusTasteGate`
`single carrier alignment`

Theorem 6148.3 (Compact interval oscillation-modulus handoff)**.** *Every downstream Cauchy, compact-interval, or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$ *carrier receives the finite route*

$$L, W, R, Q, O, M, E, H, C, P, N.$$

The handoff displays the $\mathsf{LocatedInterval}^{\uparrow}$ *source, the* $\mathsf{StreamName}^{\uparrow}$ *window, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the oscillation ledger, the modulus handoff, and the terminal* $\mathsf{Real}^{\uparrow}$ *seal before any consumer may use the interval-local modulus.*

Proof. Start with the NameCert obligations of Definition 6148.2. They place located interval admission, stream-window exposure, regular rational readback, dyadic tolerance budgeting, oscillation-ledger exactness, modulus handoff, and real sealing in one displayed packet.

The route to M must pass through L, W, R, Q, O. The located interval row supplies the interval source, the window and readback rows supply the finite observations, the dyadic row supplies the

tolerance, and the oscillation ledger records the finite comparisons from which the modulus handoff is read.

The route to E must pass through the same prefix. Componentwise transport and replay in H and C preserve that prefix, while P and N source and name it locally. Thus any downstream Cauchy, compact-interval, or Real-facing consumer receives the interval-local oscillation modulus only as the displayed finite handoff. □

Falsifiable prediction: Every accepted $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$ read must expose $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, the oscillation ledger, the modulus handoff, $\mathsf{Real}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Independence witness: locatedinterval, streamname, regseqrat, dyadicratcore, real: CompactIntervalOscillationM is not bijective to any listed sibling because it binds interval-local oscillation comparisons to a finite modulus handoff before Real-facing consumers may read the seal.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalOscillationModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactIntervalOscillationModulus}^{\uparrow}$ packet is generated from a LocatedInterval source row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, an oscillation ledger, a modulus handoff, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalOscillationModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6148.1, Definition 6148.2, and Definition 6148.3 seed the finite interval-local route from located compact reads to oscillation-modulus consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalOscillationModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host compact intervals, selected endpoint values, quotient real equality, selected suprema, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exactness, finite-window readback, dyadic tolerance exactness, oscillation-ledger exactness, modulus handoff, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactIntervalOscillationModulusUp.

Chapter 6149

A Concrete Naming Certificate for $\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}$

Following Bishop and Bridges, Constructive Analysis, the $\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}$ packet records the finite BEDC boundary at which a real-name consumer may see displayed comparison evidence without receiving trichotomy for constructive reals. It sits beside the terminal real-name seal of chapter 38, the regular-rational readback of chapter 301, the stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, the apartness real boundary of chapter 304, and the Markov-sensitive inspection refusal of chapter 1666. The packet is a finite refusal object for Real and Completion consumers: it displays the rows that justify local apartness or equality readback, and it records that the missing third case is not filled by LPO, WLPO, Markov's principle, host equality, quotient stream equality, or selected limit data.

Definition 6149.1 (Constructive real trichotomy boundary carrier)**.** *A* $\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (X, Y, S_X, S_Y, R_X, R_Y, D, A, E, L, H, C, P, N)$$

with the following displayed rows:

1. *X and Y are the two* $\mathsf{Real}^{\uparrow}$ *name rows being compared.*
2. *S_X and S_Y are finite* $\mathsf{StreamName}^{\uparrow}$ *observation windows for the two names.*
3. *R_X and R_Y are the corresponding* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readbacks exposed inside the windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the comparison request.*
5. *A is the apartness-evidence row when a displayed dyadic gap separates the two readbacks.*
6. *E is the equality-readback row when the displayed regular-rational windows agree to the requested tolerance.*
7. *L is the refusal ledger used when neither a finite apartness row nor a finite equality-readback row is displayed.*
8. *H records componentwise* hsame *transport for the real names, stream windows, regular readbacks, dyadic ledger, apartness row, equality-readback row, refusal ledger, continuation, provenance, and local naming row.*

9. *C records displayed* Cont *replay from a real comparison request through the finite windows, readbacks, tolerance ledger, and the visible boundary row among A, E, L.*
10. *P is the* Pkg *provenance over* ConstructiveRealTrichotomyBoundary$^{\uparrow}$, Real$^{\uparrow}$, RegSeqRat$^{\uparrow}$, StreamName$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, ApartnessReal$^{\uparrow}$, MarkovPrincipleBoundary$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{ConstructiveRealTrichotomyBoundary}^{\uparrow}}$ *row.*

The carrier has no coordinate for constructive real trichotomy, LPO, WLPO, Markov's principle as an imported rule, host real equality, quotient stream equality, countable choice, selected representatives, selected limits, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveRealTrichotomyBoundaryUp.ConstructiveRealTrichotomyBoundaryTasteGate single carrier alignment`

Theorem 6149.2 (Constructive real trichotomy boundary NameCert obligations)**.** *For every accepted* ConstructiveRealTrichotomyBoundary$^{\uparrow}$ *carrier*

$$T = (X, Y, S_X, S_Y, R_X, R_Y, D, A, E, L, H, C, P, N),$$

the local NameCert$_{\text{ConstructiveRealTrichotomyBoundary}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier formation for the displayed real names, stream windows, regular-rational readbacks, dyadic tolerance ledger, boundary rows, transport, replay, provenance, and local name;*
2. *classifier stability, requiring every comparison read to be transported componentwise through H and replayed through C;*
3. *apartness/equality handoff, allowing a consumer to use A when a finite positive gap is displayed and E when a finite equality readback is displayed;*
4. *LPO-refusal, requiring the ledger L whenever neither displayed finite side is present;*
5. *non-escape, excluding constructive real trichotomy, LPO, WLPO, Markov's principle, host equality, quotient stream equality, countable choice, selected representatives, selected limits, and ambient real completeness from the exported certificate.*

No local naming obligation is stored outside these rows.

Proof. Carrier formation is projection from Definition 6149.1. The tuple displays the two real seals, the finite stream windows, the regular readbacks, the dyadic tolerance ledger, and exactly the three boundary rows A, E, L.

Classifier stability is componentwise. The row H transports each visible coordinate, while C replays the same finite comparison request after transport.

The apartness/equality handoff is read only from the displayed boundary rows. A positive gap must cite A, and an equality readback must cite E; when neither row is present, the consumer receives the refusal ledger L instead of a third global alternative.

The non-escape clause follows from the carrier inventory. Since no excluded semi-classical or host-equality object occurs as a coordinate of T, no local NameCert row can export it. □

Theorem 6149.3 (Constructive real trichotomy LPO refusal)**.** *Let a Real or Completion consumer accept a* ConstructiveRealTrichotomyBoundary$^{\uparrow}$ *packet*

$$T = (X, Y, S_X, S_Y, R_X, R_Y, D, A, E, L, H, C, P, N).$$

If the displayed rows do not expose an apartness witness in A and do not expose an equality-readback witness in E, the consumer must read the refusal ledger L. It cannot replace the missing finite row by LPO, WLPO, Markov's principle, host real equality, quotient stream equality, selected representatives, selected limit data, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier of Definition 6149.1. The only comparison-facing outputs are the apartness row A, the equality-readback row E, and the refusal ledger L.

Apply Definition 6149.2. Its apartness/equality handoff clause permits a consumer to use A or E only when the corresponding finite row is displayed, and its LPO-refusal clause assigns the undecided finite situation to L.

The excluded principles are absent from the tuple. Transport, replay, provenance, and local naming therefore preserve the same refusal boundary rather than supplying a host-side trichotomy consumer. □

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}$ packet is generated from two Real names, finite StreamName windows, RegSeqRat readbacks, a DyadicRatCore tolerance ledger, displayed apartness or equality-readback evidence when available, a refusal ledger otherwise, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6149.1, Definition 6149.2, and Definition 6149.3 seed the finite Real/Completion boundary for displayed apartness, equality readback, and refusal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim constructive real trichotomy, LPO, WLPO, Markov's principle, host real equality, quotient stream equality, selected representatives, selected limits, countable choice, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison classifier stability, apartness/equality handoff, refusal-ledger exactness, and non-escape for BEDC.Derived.ConstructiveRealTrichotomyBoundaryUp.

Chapter 6150

A Concrete Naming Certificate for $\mathsf{SeparableUltrametricSpace}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ packet records the finite NameCert data for an ultrametric space whose countable dense surface is available through displayed prefix windows. It sits after the countable-density route of chapter 1458, the strong-triangle surface of chapter 1535, the prefix geometry of chapter 3485, the Baire prefix carrier of chapter 334, and the finite name windows of chapter 126. The packet exposes separability and ultrametric comparison as one finite carrier, not as selected dense points, quotient stream equality, host metric equality, countable choice, or ambient completeness.

Definition 6150.1 (Separable ultrametric space carrier)**.** *A* $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, U, B, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense schedule row, recording the countable dense-name family and the finite request bounds inherited from the* $\mathsf{SeparableMetric}^{\uparrow}$ *route.*
2. *U is the ultrametric comparison row, recording parent metric reads together with the strong max-radius ledger inherited from* $\mathsf{UltrametricSpace}^{\uparrow}$*.*
3. *B is the Baire-prefix witness row, recording finite prefix cylinders and common-prefix witnesses consumed through* $\mathsf{BaireUltrametric}^{\uparrow}$ *and* $\mathsf{BaireSpace}^{\uparrow}$*.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite name coordinates inspected before a dense request or prefix comparison is accepted.*
5. *H records componentwise* hsame *transport for the dense schedule, ultrametric comparison, Baire-prefix witnesses, stream windows, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from dense requests to stream windows, from stream windows to ultrametric comparison, and from prefix witnesses to downstream Baire or completion-facing consumers.*
7. *P is the* Pkg *provenance over* $\mathsf{SeparableUltrametricSpace}^{\uparrow}$*,* $\mathsf{SeparableMetric}^{\uparrow}$*,* $\mathsf{UltrametricSpace}^{\uparrow}$*,* $\mathsf{BaireUltrametric}^{\uparrow}$*,* $\mathsf{BaireSpace}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{SeparableUltrametricSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, U, B, S, H, C, P, N. It contains no coordinate for choosing dense representatives from arbitrary balls, quotienting stream names, replacing the strong-triangle row by host metric equality, or importing ambient completeness.

Theorem 6150.2 (Separable ultrametric NameCert obligations)**.** *For every accepted* $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ *carrier*

$$X = (D, U, B, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SeparableUltrametricSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the dense schedule, ultrametric comparison, Baire-prefix witnesses, finite stream windows, componentwise transport, displayed replay, provenance, and local naming rows. The certificate exports no selected dense point, quotient stream equality, host metric equality, countable choice, or ambient completeness principle.*

Proof. Project the carrier of Definition 6150.1. The analytic rows are D, U, B, S, while H, C, P, N preserve, replay, source, and name the same finite packet.

The dense schedule row D supplies only indexed dense-name data through the separable-metric route. The ultrametric row U supplies only displayed strong-triangle comparisons, and the prefix row B supplies only finite Baire-prefix witnesses.

The stream window row S is the common readback boundary. Every accepted dense request or prefix comparison must expose its inspected coordinates there before any downstream consumer reads the route.

Since the remaining rows only transport, replay, source, and name this packet, the listed rows exhaust the local NameCert surface and the excluded ambient objects have no carrier coordinate. □

Lean4 ✓ `BEDC.Derived.SeparableUltrametricSpaceUp.SeparableUltrametricSpaceTasteGate single carrier alignment`

Theorem 6150.3 (Separable ultrametric dependency obligations)**.** *Every accepted* $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ *packet reads its density, ultrametric, prefix, and stream-window evidence through*

$$\mathsf{SeparableMetric}^{\uparrow}, \quad \mathsf{UltrametricSpace}^{\uparrow}, \quad \mathsf{BaireUltrametric}^{\uparrow}, \quad \mathsf{BaireSpace}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}.$$

The route is finite and ordered by D, U, B, S, H, C, P, N: density requests pass through D and S, ultrametric comparisons pass through U and S, and Baire-prefix witnesses pass through B and S before the structural rows are consumed.

Proof. Read the dense schedule first. The row D is the only countable-density coordinate, so any separable-metric consumer must use the displayed request schedule rather than a selected dense point.

Read the ultrametric comparison next. The row U is the only strong-triangle coordinate, so ball and prefix comparisons cannot bypass the finite max-radius ledger.

Read the Baire-prefix and stream-window rows together. The row B names finite prefix witnesses and S names the inspected stream coordinates; these are the only rows from which a Baire ultrametric or Baire-space consumer can obtain prefix data.

Finally H, C, P, N preserve that dependency route without adding a second source. Thus the dependency surface is exactly the displayed sibling route. □

Theorem 6150.4 (Separable ultrametric dense-prefix transport)**.** *Let $X = (D, U, B, S, H, C, P, N)$ and $X' = (D', U', B', S', H', C', P', N')$ be accepted* $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ *carriers. If the dense schedules, ultrametric comparison rows, Baire-prefix witness rows, and finite stream windows transport componentwise under* hsame *and replay through displayed* Cont *routes, then every dense-prefix consumer of X and X' reads the same finite route through D, U, B, S and D', U', B', S' before provenance and local naming are consumed.*

Proof. Start with the NameCert obligations of Definition 6150.2. They identify the only readable analytic rows as D, U, B, S and their primed counterparts.

Use the dependency obligations of Definition 6150.3. Density, ultrametric comparison, Baire-prefix witnesses, and stream windows are already tied to the listed sibling routes before any structural row is inspected.

The componentwise hsame rows in H, H' transport those four analytic rows, and the displayed Cont rows in C, C' replay the same dense-prefix route. Provenance and local naming in P, N, P', N' then record the transported packet without adding selected representatives, quotient stream equality, host equality, countable choice, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ read factors through the dense schedule, ultrametric comparison, Baire-prefix witness, StreamName window, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any consumer reads density or prefix geometry.

Independence witness: separablemetric, ultrametricspace, baire-ultrametric, bairespace, streamname: $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries countable-density scheduling, strong ultrametric comparison, Baire-prefix witnesses, finite stream windows, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{SeparableUltrametricSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{SeparableUltrametricSpace}^{\uparrow}$ packet is generated from a dense schedule row, an ultrametric comparison row, Baire-prefix witness rows, finite StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SeparableUltrametricSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6150.1, Definition 6150.2, Definition 6150.3, and Definition 6150.4 seed the finite separable-ultrametric route through dense schedules, strong ultrametric comparisons, Baire-prefix witnesses, and stream windows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SeparableUltrametricSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected dense representatives, quotient stream equality, host metric equality, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-schedule exactness, ultrametric comparison exactness, Baire-prefix witness coverage, stream-window readback, transport, replay, provenance, and local naming for BEDC.Derived.SeparableUltrametricSpaceUp.

Chapter 6151

A Concrete Naming Certificate for $\mathsf{FanTheoremBarModulus}^{\uparrow}$

Following Brouwer's fan theorem and Bishop–Bridges constructive uniform continuity, the $\mathsf{FanTheoremBarModulus}^{\uparrow}$ packet records the finite bar-modulus extraction route by which a decidable fan bar yields a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff. It sits over the finite fan certificate of chapter 400, the Bishop fan-modulus schedule of chapter 1707, the fan-modulus boundary of chapter 2379, the compact-uniform modulus carrier of chapter 2299, and the downstream consumer surface of chapter 369. The packet names only displayed finite bar coverage, local modulus rows, rational tolerance rows, and structural replay; it does not import open-cover compactness, countable choice, selected infinite branches, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6151.1 (Fan-theorem bar-modulus carrier)**.** *A* $\mathsf{FanTheoremBarModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, B, W, L, R, U, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row, recording the binary prefix tree, decidable bar classifier, unary bar-depth ledger, and finite witness window.*
2. *B is the bar-coverage row, asserting that every admitted branch-prefix request enters the displayed bar before the depth named by F.*
3. *W is the finite window row that converts the bar-depth ledger into the prefix window inspected by compact-uniform and Cantor-space modulus consumers.*
4. *L is the local modulus ledger, listing the pointwise modulus rows attached to the finitely many retained prefixes or compact probes.*
5. *R is the positive* $\mathsf{Rat}^{\uparrow}$ *tolerance row used to compare the local radii and to choose the tolerance-only precision exported downstream.*
6. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row, exposing the extracted precision as a modulus ledger after F, B, W, L, R have all been displayed.*
7. *H records componentwise* hsame *transport for the fan, bar-coverage, window, local-modulus, rational-tolerance, uniform-modulus, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the finite extraction route*

$$F, B, W, L, R \longmapsto U.$$

9. *P is the* Pkg *provenance over* $\mathsf{FanTheoremBarModulus}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{BishopFanModulus}^{\uparrow}$, FanModulusBoundary $\mathsf{CompactUniformModulus}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FanTheoremBarModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, B, W, L, R, U, H, C, P, N$. It has no coordinate for open-cover compactness, countable choice, a selected infinite branch, a selected subcover, quotient stream equality, a host modulus selector, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.FanTheoremBarModulusUp.FanTheoremBarModulusTasteGate single carrier alignment`

Theorem 6151.2 (Fan-theorem bar-modulus NameCert obligations)**.** *For every accepted* $\mathsf{FanTheoremBarModulus}^{\uparrow}$ *carrier $M = (F, B, W, L, R, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FanTheoremBarModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, finite fan-bar exposure through F, bar-coverage exactness through B, finite window extraction through W, local modulus exposure through L, positive rational tolerance accounting through R,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

In particular, every compact-uniform-continuity or finite fan consumer that extracts a uniform modulus from this packet must first read the finite fan row, the bar-coverage row, the finite window row, the local modulus ledger, and the rational tolerance row.

Proof. Project the carrier of Definition 6151.1. The projection lists exactly the finite fan, bar coverage, window, local modulus, rational tolerance, uniform-modulus, transport, replay, provenance, and local naming rows.

The fan and bar clauses are finite. The row F supplies the displayed tree, bar decisions, depth ledger, and witness window, while B states that branch-prefix requests meet the displayed bar before that depth. No selected branch or open cover is introduced.

The extraction clauses are W, L, R. The window row converts the bar depth into the finite family of inspected prefixes, the local modulus row attaches pointwise modulus data to that family, and the rational row stores the positive tolerance comparisons used to choose one exported precision.

Finally, U, H, C, P, N expose, transport, replay, source, and name the extracted precision as a $\mathsf{UniformModulus}^{\uparrow}$ handoff. Since the carrier has no coordinate for open-cover compactness, countable choice, selected branches, selected subcovers, quotient stream equality, host modulus selection, or ambient completeness, the stated obligations exhaust the packet. □

Theorem 6151.3 (Fan-theorem finite bar modulus extraction)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *read of an accepted* $\mathsf{FanTheoremBarModulus}^{\uparrow}$ *carrier factors through*

$$F, B, W, L, R, U, H, C, P, N.$$

The exported precision is obtained only from the finite fan bar, its coverage row, the extracted prefix window, the local modulus ledger, and the positive rational tolerance ledger. The read cannot be replaced by open-cover compactness, a chosen infinite branch, a selected subcover, countable choice, quotient stream equality, a host modulus selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a $\mathsf{UniformModulus}^{\uparrow}$ read of the accepted packet. By Definition 6151.2, the read must first expose F, B, W, L, R.

The route is finite at each step. The fan theorem row bounds branch-prefix inspection by a unary depth, the coverage row certifies that the displayed bar is met within that depth, and the window row lists the prefixes that may be inspected.

The local modulus and rational tolerance rows then determine the exported precision. The finite family of local modulus rows is read through L, and the positive rational comparisons in R select the tolerance-only lower bound delivered to U.

The structural rows H, C, P, N preserve the same route under transport, continuation replay, provenance, and local naming. Because the tuple contains no stronger compactness, choice, quotient, branch-selection, subcover-selection, host-selector, or completeness coordinate, every $\mathsf{UniformModulus}^{\uparrow}$ read factors through exactly the displayed finite packet. □

Falsifiable prediction: Every accepted $\mathsf{FanTheoremBarModulus}^{\uparrow}$ uniform-modulus read factors through a finite FanTheorem row, bar-coverage row, finite window row, local modulus ledger, positive RatUp tolerance row, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: fantheorem, bishop-fan-modulus, fan-modulus-boundary, compact-uniform-modulus, uniformmodulus: $\mathsf{FanTheoremBarModulus}^{\uparrow}$ is not bijective to any listed sibling because it isolates the finite bar-coverage-to-tolerance extraction route before the UniformModulus handoff, while refusing open-cover compactness, choice, selected branches, selected subcovers, quotient stream equality, host modulus selectors, and ambient completeness.

Closure status for $\mathsf{NameCert}_{\mathsf{FanTheoremBarModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{FanTheoremBarModulus}^{\uparrow}$ packet is generated from a finite FanTheorem row, bar-coverage exactness, finite window extraction, local modulus rows, positive RatUp tolerance accounting, a UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FanTheoremBarModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6151.1, Definition 6151.2, and Definition 6151.3 seed the finite fan-bar coverage route that extracts a tolerance-only UniformModulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FanTheoremBarModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FanTheoremBarModulusUp.FanTheoremBarModulusTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected infinite branches, selected subcovers, quotient stream equality, host modulus selection, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bar-coverage exactness, finite window extraction, local modulus exposure, RatUp tolerance accounting, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FanTheoremBarModulusUp.

Chapter 6152

A Concrete Naming Certificate for $\mathsf{FanUniformContinuity}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FanUniformContinuity}^{\uparrow}$ packet records the constructive fan-based route from finite bars on Cantor space to uniform-modulus extraction. It consumes the Cantor-prefix surface of chapter 1475, the finite fan row of chapter 400, the tolerance handoff of chapter 391, the finite compact metric row of chapter 129, and the terminal real-name seal of chapter 38. The packet names only displayed finite prefix windows, decidable bar coverage, modulus rows, transport, replay, provenance, and local naming; it does not claim open-cover compactness, a selected branch, countable choice, quotient function equality, or ambient real completeness.

Definition 6152.1 (FanUniformContinuity carrier)**.** *A* $\mathsf{FanUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (C, F, T, M, K, E, H, R, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CantorSpace}^{\uparrow}$ *finite-prefix row, including Boolean prefix observations and retained branch-window data.*
2. *F is the* $\mathsf{FanTheorem}^{\uparrow}$ *finite fan row, recording the decidable bar classifier, finite bar-depth ledger, and witness window for the displayed binary spread.*
3. *T is the target tolerance row for the* $\mathsf{UniformContinuity}^{\uparrow}$ *consumer, read as displayed precision data rather than as a host real predicate.*
4. *M is the extracted modulus row, assigning one finite prefix depth sufficient for all retained Cantor histories at tolerance T.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-prefix coverage row used only as a displayed compact metric ledger for the Cantor-prefix surface.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for real-valued reads that consume the same modulus route.*
7. *H records componentwise* hsame *transport for the Cantor, fan, tolerance, modulus, compact metric, Real seal, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay of the finite route*

$$C, F, T, K \longmapsto M \longmapsto E.$$

9. *P is the* Pkg *provenance over* FanUniformContinuity$^\uparrow$, CantorSpace$^\uparrow$, FanTheorem$^\uparrow$, UniformContinuity$^\uparrow$, CompactMetric$^\uparrow$, Real$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* NameCert$_{\text{FanUniformContinuity}^\uparrow}$ *row.*

The classifier compares accepted packets only through $C, F, T, M, K, E, H, R, P, N$. It has no coordinate for open-cover compactness, selected infinite branches, choice of bar witnesses, quotient equality of functions, selected real limits, or ambient Real$^\uparrow$ *completeness.*

Theorem 6152.2 (FanUniformContinuity NameCert obligations)**.** *For every accepted* FanUniformContinuity$^\uparrow$ *carrier $U = (C, F, T, M, K, E, H, R, P, N)$, the local* NameCert$_{\text{FanUniformContinuity}^\uparrow}$ *surface consists exactly of carrier admission, Cantor-prefix exposure through C, finite fan-bar exposure through F, tolerance accountability through T, modulus extraction through M, compact metric ledger exposure through K, Real-seal handoff through E, componentwise* hsame *transport through H, displayed* Cont *replay through R,* Pkg *provenance through P, and local naming through N.*

Every UniformContinuity$^\uparrow$ *or real-valued consumer of the packet must first read the displayed Cantor prefix row, finite fan row, tolerance row, and compact metric ledger before the modulus row or Real seal is exposed.*

Proof. Project the carrier of Definition 6152.1. The projection lists exactly the Cantor prefix, finite fan, tolerance, modulus, compact metric, Real seal, transport, replay, provenance, and local naming rows.

The fan-based step is finite. The FanTheorem$^\uparrow$ row supplies the decidable bar classifier and the finite bar-depth ledger, while the CantorSpace$^\uparrow$ row supplies the prefix windows over which the bar is inspected.

The modulus step reads T, K, M. The tolerance row fixes the requested precision, the compact metric row records only finite-prefix coverage, and the modulus row is extracted from those displayed rows before any downstream UniformContinuity$^\uparrow$ consumer can read it.

Finally E, H, R, P, N seal, transport, replay, source, and name the same finite route. Since the packet has no coordinate for open-cover compactness, selected branches, countable choice, quotient equality of functions, selected real limits, or ambient completeness, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: Every accepted FanUniformContinuity$^\uparrow$ consumer read factors through displayed CantorSpace, FanTheorem, UniformContinuity, CompactMetric, Real, BHist, hsame, Cont, Pkg, and NameCert rows before a uniform-modulus or Real-facing observation is exposed.

Independence witness: cantorspace, fantheorem, uniformcontinuity, compactmetric, real: FanUniformContinuity$^\uparrow$ is not bijective to any listed sibling because it jointly carries Cantor prefix windows, finite fan-bar depth, tolerance accountability, compact metric finite coverage, modulus extraction, Real sealing, transport, replay, provenance, and local naming while refusing open-cover compactness, selected branches, countable choice, quotient function equality, selected real limits, and ambient completeness.

Closure status for NameCert$_{\text{FanUniformContinuity}^\uparrow}$.

Constructive story: A FanUniformContinuity$^\uparrow$ packet is generated from CantorSpace finite-prefix rows, FanTheorem finite bar rows, a UniformContinuity tolerance row, a CompactMetric finite-prefix coverage row, an extracted modulus row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FanUniformContinuity}^\uparrow}$, seed)

Scope closed: Definition 6152.1 and Definition 6152.2 seed the finite fan-bar route from Cantor prefix inspection to uniform-modulus extraction and Real sealing.

Formal verification: FormalStatus(NameCert$_{\text{FanUniformContinuity}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, countable choice, selected infinite branches, quotient function equality, selected real limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cantor-prefix exposure, finite fan-bar exposure, tolerance accountability, compact metric ledger exposure, modulus extraction, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.FanUniformContinuityUp.

Chapter 6153

A Concrete Naming Certificate for $\mathsf{DyadicToleranceComposition}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{DyadicToleranceComposition}^{\uparrow}$ packet records the finite epsilon-budget composition route used when two local tolerance handoffs must be chained before a downstream real-name or completion-facing consumer reads the final bound. It sits over the split-budget surface of chapter 1761, the scale discipline of chapter 1768, the triangle ledger of chapter 1211, the dyadic arithmetic core of chapter 291, the finite window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet gives diagonal, Cauchy, and completion routes a named finite tolerance-composition carrier rather than forcing each consumer to restate the same two-step budget handoff.

Definition 6153.1 (Dyadic tolerance-composition carrier)**.** *A* $\mathsf{DyadicToleranceComposition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, S_0, S_1, B_0, B_1, Q', W_0, W_1, R_0, R_1, D, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the request tolerance row presented to the downstream consumer before local budgets are inspected.*
2. *S_0 and S_1 are the two* $\mathsf{DyadicToleranceScale}^{\uparrow}$ *rows that expose the input and output precision scales for the two local tolerance stages.*
3. *B_0 and B_1 are the two local* $\mathsf{DyadicErrorBudget}^{\uparrow}$ *rows consumed by the first and second tolerance stages.*
4. *Q' is the composed tolerance row, recording the final dyadic request after the two scale and budget rows have been chained.*
5. *W_0 and W_1 are the* $\mathsf{StreamName}^{\uparrow}$ *finite-window rows from which the two local reads are opened.*
6. *R_0 and R_1 are the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows attached to those windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *arithmetic ledger recording the displayed dyadic sum, comparison slack, and final bound behind Q'.*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the composed tolerance route has been displayed.*

9. *H records componentwise* hsame *transport for the request, two scale rows, two budget rows, composed tolerance row, windows, readbacks, dyadic arithmetic ledger, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from the request through the two scale rows, two local budgets, composed tolerance, windows, readbacks, dyadic ledger, and Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{DyadicToleranceComposition}^{\uparrow}$, $\mathsf{DyadicErrorBudget}^{\uparrow}$, $\mathsf{DyadicToleranceScale}^{\uparrow}$, $\mathsf{DyadicToleranceTriangleLedger}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceComposition}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, host real equality, selected limits, countable choice, trichotomy, detached completion evidence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.DyadicToleranceCompositionUp.DyadicToleranceCompositionTasteGate single carrier alignment`

Theorem 6153.2 (Dyadic tolerance-composition NameCert obligations)**.** *For every accepted* DyadicToleranceComp *carrier $T = (Q, S_0, S_1, B_0, B_1, Q', W_0, W_1, R_0, R_1, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicToleranceComposition}^{\uparrow}}$ *surface consists exactly of carrier admission for the request tolerance, two scale rows, two local budget rows, composed tolerance row, stream windows, regular-rational readbacks, dyadic arithmetic ledger, Real seal, transport, replay, provenance, and local naming; scale-budget handoff exactness requiring S_0, B_0 and S_1, B_1 to be consumed in order; composed-tolerance exactness requiring Q' to be read through D; Real-seal compatibility requiring E to be downstream of the composed route; and nonescape excluding quotient streams, selected limits, host equality, countable choice, trichotomy, detached completion evidence, and ambient completeness.*

Proof. Carrier admission is projection from Definition 6153.1. The packet displays exactly the request row, the two scale rows, the two local budget rows, the composed tolerance row, the two windows, the two readbacks, the dyadic ledger, the Real seal, and the support rows.

Scale-budget handoff exactness is read from S_0, B_0 and S_1, B_1. The carrier gives each local stage one scale row and one budget row, so a consumer cannot insert an unrecorded tolerance stage between them.

Composed-tolerance exactness is the role of Q' and D. The final bound is visible only through the dyadic arithmetic ledger, which records the displayed sum and comparison slack before the Real seal is consumed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier omits quotient streams, host equality, selected limits, countable choice, trichotomy, detached completion evidence, and ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6153.3 (Dyadic tolerance-composition scale-budget handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *diagonal, Cauchy, or completion-facing consumer of an accepted* $\mathsf{DyadicToleranceComposition}^{\uparrow}$ *carrier must read the finite route*

$$Q \longmapsto S_0, B_0 \longmapsto S_1, B_1 \longmapsto Q', D \longmapsto W_0, W_1, R_0, R_1, E, H, C, P, N.$$

Thus the first local tolerance stage, the second local tolerance stage, and the composed dyadic bound are displayed before the Real seal is available.

Proof. Project the carrier and read the request row Q. The only local tolerance-stage rows attached to it are S_0, B_0 and S_1, B_1.

Use the sibling surfaces named in the carrier. The $\mathsf{DyadicToleranceScale}^{\uparrow}$ rows expose the precision scales, while the $\mathsf{DyadicErrorBudget}^{\uparrow}$ rows expose the two local budget consumptions.

The row D composes those local reads into the final row Q'. The stream windows and regular-rational readbacks W_0, W_1, R_0, R_1 then provide the finite observation route before E seals the real-name boundary.

Transport, replay, provenance, and local naming preserve the same order. Since no unrecorded scale, budget, selected limit, or completeness row is present, every listed consumer receives exactly this handoff. □

Theorem 6153.4 (Dyadic tolerance-composition nonescape)**.** *An accepted* $\mathsf{DyadicToleranceComposition}^{\uparrow}$ *carrier exports only the finite composition rows*

$$Q, S_0, S_1, B_0, B_1, Q', W_0, W_1, R_0, R_1, D, E, H, C, P, N.$$

It cannot export quotient stream equality, host real equality, selected limits, countable choice, trichotomy, detached completion evidence, ambient $\mathsf{Real}^{\uparrow}$ *completeness, or a hidden third tolerance stage.*

Proof. The carrier definition lists the full public inventory. Its analytic rows are the request tolerance, two scale rows, two local budget rows, composed tolerance, finite windows, regular-rational readbacks, dyadic arithmetic ledger, and Real seal.

The structural rows H, C, P, N only transport, replay, source, and name those analytic rows. They do not add a selector, quotient, host equality predicate, trichotomy decision, completion witness, or independent tolerance stage.

Therefore downstream diagonal, Cauchy, and completion routes must obtain any stronger principle from separate chapters after reading this packet. The $\mathsf{DyadicToleranceComposition}^{\uparrow}$ surface supplies only the displayed finite epsilon-budget composition route. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicToleranceComposition}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicToleranceComposition}^{\uparrow}$ packet is generated from one request tolerance, two DyadicToleranceScale rows, two DyadicErrorBudget rows, one composed tolerance row, two StreamName windows, two RegSeqRat readbacks, one DyadicRatCore arithmetic ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicToleranceComposition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6153.1, Definition 6153.2, Definition 6153.3, and Definition 6153.4 seed the finite epsilon-budget composition packet over two local tolerance stages and one composed Real-facing bound.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicToleranceComposition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, host real equality, selected limits, countable choice, trichotomy, detached completion evidence, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking for $\mathsf{DyadicToleranceComposition}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, scale-budget handoff exactness, composed-tolerance exactness, stream-window readback, Real-seal compatibility, transport, replay, provenance, and local naming for BEDC.Derived.DyadicToleranceCompositionUp.

Chapter 6154

A Concrete Naming Certificate for $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$

Following Kuratowski's closure axioms and Kelley's topology language, the $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ packet records a displayed closure operator by finite BEDC rows rather than by an ambient powerset primitive. It sits after the indexed-open topology surface of chapter 88, the membership discipline of chapter 3668, the metric closure route of chapter 3342, and the neighbourhood closure surface of chapter 3559. The packet names exactly the finite closure-law interface consumed by topology and metric-closure chapters; it does not import host subset extensionality, arbitrary powersets, selected closure witnesses, open-cover compactness, or ambient real completeness.

Definition 6154.1 (Kuratowski closure axiom carrier)**.** *A* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (T, S, L, E, M, I, U, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the indexed open surface and classifier through which closed-neighbourhood requests are interpreted.*
2. *S is the* $\mathsf{SetLike}^{\uparrow}$ *subset row, recording membership, subset comparison, and classifier-respecting transport for the displayed source subset.*
3. *L is the closure row, assigning to each displayed subset row its displayed closure subset and the membership-read interface for that closure.*
4. *E is the extensivity row, requiring every displayed membership in the source subset to replay as membership in its closure.*
5. *M is the monotonicity row, requiring every displayed subset inclusion to transport to the corresponding displayed closure inclusion.*
6. *I is the idempotence row, requiring the closure of a displayed closure row to replay through the same closure membership route.*
7. *U is the finite-union and empty-compatibility row, carrying the displayed equality of the closure of a binary union with the binary union of closures and the displayed empty-closure boundary.*
8. *H records componentwise* hsame *transport for the topology source, subset row, closure row, the four closure-law rows, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a closure request through $T, S, L, E, M, I, U, H, P, N$ before any topology or metric-closure consumer reads the result.*
10. *P is the* Pkg *provenance over* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, $\mathsf{TopologicalClosure}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{KuratowskiClosureAxiom}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, S, L, E, M, I, U, H, C, P, N$ and the displayed routes among those rows. It has no coordinate for a host powerset, an ambient closure primitive, subset extensionality outside the carrier, selected closure witnesses, open-cover compactness, countable choice, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.KuratowskiClosureAxiomUp.KuratowskiClosureAxiomTasteGate single carrier alignment`

Theorem 6154.2 (Kuratowski closure axiom NameCert obligations)**.** *For every accepted* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ *carrier*

$$K = (T, S, L, E, M, I, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KuratowskiClosureAxiom}^{\uparrow}}$ *surface consists exactly of carrier admission for the topology source row, set-like subset row, closure row, extensivity row, monotonicity row, idempotence row, finite-union and empty-compatibility row, componentwise transport, displayed replay, provenance, and local naming. Every topology-facing or metric-closure-facing use of the packet must display those rows before it can treat a subset read as closure evidence.*

Proof. Project the carrier of Definition 6154.1. The source rows are T and S, the operation row is L, and the closure-law rows are E, M, I, U.

The law rows are finite certificate rows. Extensivity reads membership from S into L, monotonicity transports displayed subset inclusion through closure, idempotence replays the closure row through itself, and finite-union and empty compatibility provide the binary union and empty boundaries.

The structural rows H, C, P, N preserve the same finite packet by transport, continuation replay, provenance, and local naming. Since the carrier has no coordinate for host powersets, ambient closure primitives, external subset extensionality, selected witnesses, open-cover compactness, countable choice, Lean verification, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 6154.3 (Kuratowski closure topology handoff)**.** *Every accepted* $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ *packet gives a topology handoff from displayed subset reads to displayed closure reads over the route*

$$T, S, L, E, M, I, U, H, C, P, N.$$

A $\mathsf{Topology}^{\uparrow}$, $\mathsf{MetricClosure}^{\uparrow}$, *or* $\mathsf{TopologicalClosure}^{\uparrow}$ *consumer may use the closure only after the topology source, set-like subset row, closure row, four Kuratowski law rows, componentwise transport, displayed replay, provenance, and local naming rows have been inspected. The handoff exports no host powerset, no ambient closure primitive, no selected closure point, no open-cover compactness theorem, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the topology source row T and the subset row S. They determine the displayed carrier, classifier, membership reads, and subset comparisons available to a closure consumer.

Next read the closure row L together with the four law rows. The rows E, M, I, U are exactly the Kuratowski closure axioms as displayed BEDC obligations: source membership enters closure,

inclusion is preserved by closure, closure is idempotent, binary unions commute with closure, and the empty subset has the displayed boundary row.

Finally replay the structural rows. The hsame transport row H and Cont row C keep the topology, subset, closure, and law rows aligned under presentation change, while P and N source and name the packet for downstream consumers.

Thus the topology and metric-closure handoff is precisely the displayed finite route. No excluded host powerset, ambient closure primitive, selected closure witness, open-cover compactness result, or ambient Real completeness claim is available from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ topology or metric-closure read factors through a Topology source row, SetLike subset row, closure row, extensivity row, monotonicity row, idempotence row, finite-union and empty-compatibility row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: topology, setlike, metric-closure, topological-closure: $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a displayed closure operator and the four Kuratowski law rows over one topology and subset interface, while each sibling supplies only the topology, membership, metric-closure, or neighbourhood-closure part of the route.

Closure status for $\mathsf{NameCert}_{\mathsf{KuratowskiClosureAxiom}^{\uparrow}}$.

Constructive story: A $\mathsf{KuratowskiClosureAxiom}^{\uparrow}$ packet is generated from a Topology source row, SetLike subset row, closure row, extensivity, monotonicity, idempotence, finite-union and empty compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KuratowskiClosureAxiom}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6154.1, Definition 6154.2, and Definition 6154.3 seed the finite closure-operator route consumed by topology, metric-closure, and topological-closure packets.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KuratowskiClosureAxiom}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, ambient closure primitives, external subset extensionality, selected closure witnesses, open-cover compactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, closure-row exactness, extensivity, monotonicity, idempotence, finite-union and empty compatibility, transport, replay, provenance, and local naming for BEDC.Derived.KuratowskiClosureAxiom

Chapter 6155

A Concrete Naming Certificate for $\mathsf{ClusterFilter}^{\uparrow}$

Following Kelley, Bourbaki, and Bishop-style filter convergence, the $\mathsf{ClusterFilter}^{\uparrow}$ packet records the finite interface by which a displayed filter base can be read as clustering around a point or completion boundary. It sits after the Cauchy-filter surface of chapter 333, the metric-space carrier of chapter 4230, the topology carrier of chapter 88, the finite stream-name route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter supplies a standard cluster-filter neighbor for completion and compactness arguments without importing host powersets, selected adherent points, quotient filters, countable choice, or ambient completeness.

Definition 6155.1 (ClusterFilter carrier)**.** *A* $\mathsf{ClusterFilter}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, M, T, W, R, E, A, H, C, P, N)$$

with the following displayed rows:

1. *F is the visible* $\mathsf{CauchyFilter}^{\uparrow}$ *row naming the finite filter-base requests, directed refinements, and Cauchy-readiness data that may be inspected by a completion consumer.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *row recording the distance observations and metric-law ledger used by every ball-neighborhood request.*
3. *T is the* $\mathsf{Topology}^{\uparrow}$ *row recording the neighbourhood reads through which clustering is observed.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used when the filter is probed by sequential or net-shaped observations.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance-readback row attached to the windows and metric requests.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *terminal seal read after the filter, metric, topology, window, and tolerance rows have all been displayed.*
7. *A is the adherent-observation row. It records that every displayed neighbourhood request compatible with T has a displayed filter event in F whose metric tolerance is read through M, R, E.*
8. *H records componentwise* hsame *transport for the filter, metric, topology, window, tolerance, real-seal, adherent-observation, and structural rows.*

9. *C records displayed* Cont *replay from filter-base refinement to neighbourhood observation, from metric tolerance to regular rational readback, and from the adherent row to the real-seal and completion-facing handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{ClusterFilter}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClusterFilter}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, M, T, W, R, E, A, H, C, P, N$ and the displayed routes among those rows. There is no coordinate for a host powerset of neighbourhoods, a selected cluster point, quotient equality of filters, a maximal filter, countable choice, or an ambient completion theorem.

The W and R rows consume the finite-prefix observation boundary of chapter 7253: before any $\mathsf{ClusterFilter}^{\uparrow}$ read can reach the $\mathsf{Real}^{\uparrow}$ seal row E, the displayed stream prefix and its regular rational readback are already confined to the $\mathsf{FinitePrefixStream}^{\uparrow}$ window surface.

Lean4 ✓ `BEDC.Derived.ClusterFilterUp.ClusterFilterUpTasteGate single carrier alignment`

Theorem 6155.2 (ClusterFilter NameCert obligations)**.** *For every accepted* $\mathsf{ClusterFilter}^{\uparrow}$ *carrier*

$$G = (F, M, T, W, R, E, A, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ClusterFilter}^{\uparrow}}$ *surface consists exactly of carrier admission, Cauchy-filter exposure through F, metric-space exposure through M, topology-neighbourhood exposure through T, finite window exposure through W, regular rational readback through R, real-seal readiness through E, adherent-observation exposure through A, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.ClusterFilterUp.ClusterFilterCarrier namecert obligations`

Proof. Project the carrier of Definition 6155.1. The rows F, M, T, W, R, E, A are the only mathematical rows available to a cluster-filter read, and H, C, P, N are the only structural rows.

The filter and neighbourhood rows are read first. The row F supplies finite base requests and refinements, while T supplies only displayed neighbourhood observations over the metric carrier M.

The observation row then binds clustering to finite tolerance data. The row A can cite only the displayed filter events, neighbourhood requests, stream windows, regular rational tolerances, and real seal already present in F, M, T, W, R, E.

Finally H, C, P, N transport, replay, source, and name that same route. Since the excluded host powerset, selected cluster point, quotient filter, maximal filter, countable-choice schedule, and ambient completion theorem are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6155.3 (ClusterFilter Cauchy-completion handoff)**.** *Every Cauchy-completion-facing consumer of an accepted* $\mathsf{ClusterFilter}^{\uparrow}$ *packet must read the displayed route*

$$\mathsf{CauchyFilter}^{\uparrow},\ \mathsf{Met}^{\uparrow},\ \mathsf{Topology}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

through the carrier rows F, M, T, W, R, E before it may use the adherent-observation row A as cluster-filter evidence. The handoff supplies a finite cluster interface for completion and compactness arguments; it supplies no selected cluster point, quotient filter, host powerset of neighbourhoods, maximal-filter principle, countable choice, ambient compactness theorem, or ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.ClusterFilterUp.ClusterFilterCarrier cauchy completion handoff`

Proof. Start with the displayed Cauchy-filter row F. A completion-facing consumer can inspect only the finite filter-base requests and directed refinements carried by that row.

Next read the metric, topology, and window rows. The metric-space row M supplies distance observations, the topology row T supplies neighbourhood requests, and the stream row W gives the finite observation windows through which those requests are probed.

The tolerance and real-seal rows are downstream. The row R records the regular rational readback for each displayed request, and E seals the Real-facing tolerance only after the filter, metric, topology, and window data have been displayed.

Only then can the adherent-observation row A be consumed as cluster-filter evidence. The structural rows H, C, P, N preserve the same finite route under transport, replay, provenance, and local naming, so no excluded choice, quotient, host-powerset, compactness, or completeness principle can enter the handoff. □

Falsifiable prediction: Every accepted $\mathsf{ClusterFilter}^{\uparrow}$ completion-facing read factors through a CauchyFilter row, MetricSpace row, Topology neighbourhood row, StreamName finite-window row, RegSeqRat tolerance row, Real seal, adherent-observation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: cauchyfilter, metric-space, topology, streamname, regseqrat, real: $\mathsf{ClusterFilter}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a visible filter row, metric-space row, topology-neighbourhood row, finite observation windows, rational tolerance readback, real seal, and adherent-observation route.

Closure status for $\mathsf{NameCert}_{\mathsf{ClusterFilter}\uparrow}$.

Constructive story: A $\mathsf{ClusterFilter}^{\uparrow}$ packet is generated from a CauchyFilter row, MetricSpace row, Topology neighbourhood row, StreamName finite-window row, RegSeqRat tolerance row, Real seal, adherent-observation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClusterFilter}\uparrow}$, seed)

Scope closed: Definition 6155.1, Definition 6155.2, and Definition 6155.3 seed the finite cluster-filter interface consumed by Cauchy-completion and compactness routes.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClusterFilter}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected cluster points, quotient filters, host powersets of neighbourhoods, maximal filters, countable choice, ambient compactness, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter exposure, metric-space exposure, topology-neighbourhood exposure, finite-window exposure, tolerance readback, real-seal readiness, adherent-observation nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ClusterFilterUp.

Chapter 6156

A Concrete Naming Certificate for RiemannIntegralCompletionModulus$^{\uparrow}$

Following Riemann and Bishop–Bridges constructive integration, the RiemannIntegralCompletionModulus$^{\uparrow}$ packet records the finite modulus route by which integration evidence becomes regular-rational and real-name readback data. It sits after the Darboux-gap evidence of chapter 1960, the tagged-gauge window surface of chapter 4115, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the reusable finite interface between integrability evidence and completion-facing reads; it does not supply a host limit, quotient equality of integrands, selected partitions, measure completion, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 6156.1 (Riemann integral completion-modulus carrier)**.** *A* RiemannIntegralCompletionModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, G, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* RiemannIntegrable$^{\uparrow}$ *row carrying Darboux-gap evidence for the inspected integrand and interval.*
2. *G is the* RiemannSumGauge$^{\uparrow}$ *row carrying tagged windows, gauge radii, mesh comparisons, and finite Riemann-sum reads.*
3. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row that records the displayed precision request, comparison slack, and dyadic bound used by the completion-facing read.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row obtained from the displayed integrability, tagged-gauge, and dyadic tolerance rows.*
5. *E is the terminal* Real$^{\uparrow}$ *seal reached only after the regular-rational readback row is public.*
6. *H records componentwise* hsame *transport for the integrability row, tagged-gauge row, dyadic tolerance row, regular readback, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the integrability evidence through tagged windows, dyadic tolerance, regular readback, and Real seal.*
8. *P is the* Pkg *provenance over* RiemannIntegralCompletionModulus$^{\uparrow}$, RiemannIntegrable$^{\uparrow}$, RiemannSumGauge$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}}$ *row.*
The classifier reads accepted carriers only through $I, G, D, R, E, H, C, P, N$. It has no coordinate for host limits, quotient equality of integrands, selected partition families, measure completion, countable choice, detached integral representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6156.2 (Riemann integral completion-modulus NameCert obligations)**.** *For every accepted carrier*

$$M = (I, G, D, R, E, H, C, P, N)$$

of Definition 6156.1, the local $\mathsf{NameCert}_{\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *gap row,* $\mathsf{RiemannSumGauge}^{\uparrow}$ *tagged-window row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming; gap-to-window compatibility requiring G to read the integrability evidence displayed by I; tolerance exactness requiring D to mediate every completion-facing precision request before R is read; Real-seal compatibility requiring E to be downstream of R; and nonescape excluding host limits, quotient integrand equality, selected partition families, measure completion, countable choice, detached integral representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RiemannIntegralCompletionModulusUp.RiemannIntegralCompletionModulusTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 6156.1. The packet displays exactly one integrability row, one tagged-gauge row, one dyadic tolerance row, one regular-rational readback, one Real seal, and the structural rows.

Gap-to-window compatibility is read from I and G. The integrability row supplies the finite Darboux-gap evidence, while the tagged-gauge row supplies only the windows and mesh comparisons attached to that evidence.

Tolerance exactness is the role of D. A completion-facing precision request reaches the regular readback row R only after the dyadic bound and comparison slack have been displayed.

Finally E, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the carrier omits host limits, quotient integrand equality, selected partition families, measure completion, countable choice, detached representatives, and ambient completeness, the listed clauses exhaust the local NameCert surface. □

Theorem 6156.3 (Riemann integral completion-modulus handoff)**.** *Every completion-facing integration read of an accepted* $\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RiemannIntegrable}^{\uparrow} \longmapsto \mathsf{RiemannSumGauge}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the read exports only the finite rows $I, G, D, R, E, H, C, P, N$ and cannot export a host limit, quotient equality of integrands, selected partition family, measure completion, countable choice, detached integral representative, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a completion-facing integration read. The only source row available to it is the integrability row I, which displays Darboux-gap evidence before any tagged window or regular readback is inspected.

The next analytic row is G. It exposes tagged windows, gauge radii, mesh comparisons, and finite Riemann-sum reads, but it does not select a partition family or identify integrands by quotient equality.

The dyadic row D then mediates the precision request. Only after this displayed tolerance read may the carrier expose the $\mathsf{RegSeqRat}^{\uparrow}$ row R and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Transport, replay, provenance, and local naming preserve the same order through H, C, P, N. Because no omitted object is present as a carrier coordinate, every accepted completion-facing read factors through the displayed finite route and exports nothing beyond it. □

Closure status for $\mathsf{NameCert}_{\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}$ packet is generated from a RiemannIntegrableUp Darboux-gap row, a RiemannSumGaugeUp tagged-window row, a DyadicRatCoreUp tolerance row, RegSeqRatUp readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6156.1, Definition 6156.2, and Definition 6156.3 seed the finite integration-modulus route from integrability evidence through tagged windows, dyadic tolerance, regular readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RiemannIntegralCompletionModulus}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RiemannIntegralCompletionModulusUp.RiemannIntegralCompletionModulusTasteGate\_single\_c`

Bridge status: none

Not claimed: This chapter does not close host limits, quotient equality of integrands, selected partition families, measure completion, countable choice, detached integral representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked gap-to-window compatibility, tolerance exactness, RegSeqRat readback, Real-seal compatibility, transport, replay, provenance, and local naming beyond the carrier-alignment theorem.

Chapter 6157

A Concrete Naming Certificate for $\mathsf{FellTopology}^{\uparrow}$

Following Fell and Kelley, the $\mathsf{FellTopology}^{\uparrow}$ packet records the finite lower-hit and compact-miss topology read on closed-subset hyperspace names. It sits over the compact-subset and closed-subset surface of chapter 1660, compares its hit/miss classifiers with the Vietoris carrier of chapter 2375, and exposes only the compact metric source rows needed by downstream convergence consumers. The packet is a BEDC NameCert for a hyperspace-facing topology object; it is not a host powerset topology, not a selected closed set, not an ambient quotient of closed subsets, and not an open-cover compactness principle.

Definition 6157.1 (Fell topology carrier)**.** *A* $\mathsf{FellTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (H, L, U, V, K, W, S, C, P, N)$$

with the following displayed rows:

1. *H is the displayed closed-subset hyperspace source. It reads every closed-subset request through the finite* $\mathsf{Hyperspace}^{\uparrow}$ *carrier before a topology consumer can inspect it.*
2. *L is the lower-hit witness row. It records finite windows showing that a displayed closed-subset name hits a displayed open probe row.*
3. *U is the upper compact-miss finite-cover witness row. It records compact source data, a displayed compact test row, and a finite cover certificate showing that the inspected closed-subset name misses that compact row.*
4. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *comparison row. It records the overlap with the finite Vietoris hit/miss surface and marks which compact-miss reads are Fell-specific.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the* $\mathsf{Met}^{\uparrow}$ *distance witnesses and finite-net schedules on which the compact miss tests are evaluated.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to replay closed-subset names, compact tests, and hit witnesses through a bounded inspection surface.*
7. *S is componentwise* hsame *transport for H, L, U, V, K, W.*
8. *C records displayed* Cont *replay of the route*

$$H, L, U, V, K, W, S \longmapsto \mathsf{FellTopology}^{\uparrow}.$$

9. *P is the* Pkg *provenance over* $\mathsf{FellTopology}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{FellTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $H, L, U, V, K, W, S, C, P, N$. It has no coordinate for a host powerset of closed subsets, selected closed-set representatives, ambient subset equality, quotient hyperspaces, or open-cover compactness.

Lean4 ✓ `BEDC.Derived.FellTopologyUp.FellTopologyTasteGate single carrier alignment`

Theorem 6157.2 (FellTopologyUp NameCert obligations)**.** *For every accepted* $\mathsf{FellTopology}^{\uparrow}$ *carrier*

$$F = (H, L, U, V, K, W, S, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FellTopology}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the closed-subset hyperspace source, lower-hit witnesses, upper compact-miss finite-cover witnesses, Vietoris comparison row, compact metric source, stream-name finite windows, transport row, replay row, provenance row, and local naming row;*

2. *hit and miss classifier stability, requiring the lower-hit row L and compact-miss row U to be transported through the same* BHist *route before their tests are compared;*

3. *compact-source exposure, requiring every compact miss test to reveal the* $\mathsf{CompactMetric}^{\uparrow}$ *source and the* $\mathsf{Met}^{\uparrow}$ *distance witnesses carried by K;*

4. *componentwise* hsame *transport through S for all topology-facing rows;*

5. *displayed* Cont *replay through C for the accepted hit/miss inspection route;*

6. Pkg *provenance through P for every dependency named by the carrier;*

7. *ledger nonescape through N, excluding host powersets, selected closed-set representatives, ambient subset equality, quotient hyperspaces, and open-cover compactness.*

No Fell topology obligation is stored outside these rows.

Theorem 6157.3 (Fell topology hyperspace handoff)**.** *Every* $\mathsf{FellTopology}^{\uparrow}$ *read from an accepted carrier $F = (H, L, U, V, K, W, S, C, P, N)$ factors through* $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, *and* $\mathsf{StreamName}^{\uparrow}$ *before any closed-set topology consumer can inspect it. The consumer receives only the displayed closed-subset hyperspace source, finite lower-hit windows, compact-miss finite-cover witnesses, compact metric source data, stream-name finite windows, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* NameCert *row.*

Proof. Start with the carrier coordinates of Definition 6157.1. A closed-set topology consumer can enter only through H, and H is already a displayed $\mathsf{Hyperspace}^{\uparrow}$ source row.

The topology classifier then splits into the lower-hit witnesses L and the compact-miss finite-cover witnesses U. The comparison row V records the shared Vietoris hit/miss part, while K supplies the $\mathsf{CompactMetric}^{\uparrow}$ and $\mathsf{Met}^{\uparrow}$ data needed for the compact miss tests.

It remains to account for finite inspection. The row W bounds the closed-subset names, compact tests, and hit witnesses by $\mathsf{StreamName}^{\uparrow}$ windows. Transport through S, replay through C, packaging through P, and local naming through N preserve that route and add no host powerset, selected closed-set representative, quotient hyperspace, or ambient subset equality coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{FellTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{FellTopology}^{\uparrow}$ packet is generated from a closed-subset hyperspace source, lower-hit witnesses, compact-miss finite-cover witnesses, Vietoris comparison, CompactMetric and Metric source rows, StreamName finite windows, hsame transport, Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FellTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6157.1, Definition 6157.2, and Definition 6157.3 seed the finite Fell hit and compact-miss topology surface over Hyperspace, VietorisTopology, CompactMetric, Metric, StreamName, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FellTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host powersets, selected closed-set representatives, ambient subset equality, quotient hyperspaces, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, hit and compact-miss classifier stability, compact-source exposure, stream-window replay, transport, provenance, local naming, and ledger nonescape for BEDC.Derived.FellTopologyUp

Chapter 6158

A Concrete Naming Certificate for $\mathsf{HeatKernel}^{\uparrow}$

Following Evans and Friedman on heat kernels and Bishop–Bridges constructive analysis, the $\mathsf{HeatKernel}^{\uparrow}$ packet records only the finite kernel-window surface inspected by BEDC consumers. It is adjacent to the finite heat-equation stencil of chapter 3652: the stencil sibling names displayed difference rows, while this chapter names the finite kernel-window packet that a downstream $\mathsf{Pde}^{\uparrow}$ consumer may inspect after the kernel source, semigroup row, rational readback, $\mathsf{Real}^{\uparrow}$ seal, hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ row have all been displayed. The packet uses $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ anchors for finite readback and tolerance discipline; it does not export global analytic heat-kernel existence, a host integral kernel, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6158.1 (HeatKernel carrier)**.** *A* $\mathsf{HeatKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (W, S, G, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the inspected kernel-window row, listing the finite time windows, spatial windows, boundary tags, and dyadic tolerances requested by the consumer.*
2. *S is the finite heat-equation stencil source row, linked to the* $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$ *sibling and exposing only displayed source, mesh, time-step, and boundary reads.*
3. *G is the semigroup-composition row, recording the finite pair of kernel windows and the displayed intermediate-window budget used to compose them.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for every kernel value, intermediate value, and stencil value inspected by W, S, G.*
5. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular rational,* $\mathsf{Cauchy}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *rows have been read.*
6. *H records componentwise* hsame *transport for W, S, G, Q, R, C, P, N and for the local certificate rows.*
7. *C records displayed* Cont *replay from the requested kernel windows through source-stencil, semigroup, readback, and real-seal rows.*

8. *P is the* Pkg *provenance over* $\mathsf{HeatKernel}^{\uparrow}$, $\mathsf{HeatEquationFiniteStencil}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{HeatKernel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $W, S, G, Q, R, H, C, P, N$. It has no coordinate for global analytic kernel existence, host integral operators, function-space quotient equality, selected infinite partitions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6158.2 (HeatKernel NameCert obligations)**.** *For an accepted* $\mathsf{HeatKernel}^{\uparrow}$ *carrier $K = (W, S, G, Q, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeatKernel}^{\uparrow}}$ *surface consists exactly of carrier admission, kernel-window admission, finite stencil-source admission, semigroup-row exactness, regular rational readback, real-seal nonescape, componentwise transport, displayed continuation replay, provenance, and local naming. These obligations are discharged along the finite route*

$$W, S, G \longmapsto Q \longmapsto R$$

together with H, C, P, N.

Lean4 ✓ `BEDC.Derived.HeatKernelUp.HeatKernelTasteGate single carrier alignment`

Proof. Carrier admission is projection from Definition 6158.1. The projection exposes the inspected kernel windows, finite stencil source, semigroup row, readback row, real seal, transport rows, replay rows, provenance row, and local name.

The kernel-window and stencil-source obligations are W and S. The row W contains only the finite time, spatial, boundary, and tolerance windows that the consumer requested, and S routes those requests through the displayed finite heat-equation stencil surface of chapter 3652.

The semigroup obligation is G. It records the finite pair of kernel windows, the displayed intermediate-window budget, and the requested composition row; it is not an integral over a host continuum and does not select an infinite partition.

The readback and seal obligations are downstream. The row Q reads the displayed values through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$, and R exposes the $\mathsf{Real}^{\uparrow}$ seal only after those finite rows have been displayed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no row for global analytic kernel existence, host integral operators, quotient function equality, selected infinite partitions, countable choice, or ambient real completeness, the listed rows exhaust the local certificate surface. □

Theorem 6158.3 (HeatKernel semigroup handoff)**.** *Every* $\mathsf{Pde}^{\uparrow}$ *consumer of an accepted* $\mathsf{HeatKernel}^{\uparrow}$ *packet reaches a semigroup-composition read only through*

$$\mathsf{HeatEquationFiniteStencil}^{\uparrow} \longmapsto \mathsf{HeatKernel}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

inside the displayed finite window package. The consumer receives no global fundamental solution, host integral kernel, quotient equality of functions, countable-choice schedule, selected limiting partition, or ambient $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Start with the finite stencil-source row S. It names the source, mesh, time-step, and boundary data that the sibling chapter 3652 exposes, so the $\mathsf{Pde}^{\uparrow}$ consumer cannot read a kernel row detached from the finite heat-equation source.

Next read the inspected kernel windows W and the semigroup row G. The handoff composes only the displayed pair of windows over the displayed intermediate-window budget, with every requested value still recorded as a finite row.

The route then passes through $\mathsf{RegSeqRat}^{\uparrow}$ readback Q and terminal $\mathsf{Real}^{\uparrow}$ seal R, while H, C, P, N preserve the same packet under transport, replay, provenance, and local naming. Because the excluded analytic objects are absent from the carrier, every semigroup handoff remains a finite NameCert route rather than a claim of global heat-kernel existence. □

Falsifiable prediction: An accepted $\mathsf{HeatKernel}^{\uparrow}$ consumer that reaches a $\mathsf{Pde}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, or $\mathsf{Real}^{\uparrow}$ row without reading the displayed kernel windows, finite heat-equation stencil source, semigroup-composition row, $\mathsf{Cauchy}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows, componentwise hsame transports, displayed Cont routes, Pkg provenance, and local NameCert row refutes the carrier surface.

Independence witness: heat-equation-finite-stencil, pde, regseqrat, real, cauchy, streamname, dyadicratcore: no listed sibling supplies the whole heat-kernel packet $K = (W, S, G, Q, R, H, C, P, N)$, because each sibling contributes only a stencil source, PDE consumer interface, rational readback, real seal, Cauchy discipline, stream schedule, or dyadic tolerance ledger while this chapter binds them into one finite semigroup-window route and local naming surface.

Closure status for $\mathsf{NameCert}_{\mathsf{HeatKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{HeatKernel}^{\uparrow}$ packet is generated from inspected kernel windows, a finite heat-equation stencil source row, a semigroup-composition row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeatKernel}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6158.1, Definition 6158.2, and Definition 6158.3 seed the finite heat-kernel window boundary over finite stencil sources, semigroup-composition rows, $\mathsf{Pde}^{\uparrow}$ consumers, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Cauchy}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ schedules, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and the $\mathsf{Real}^{\uparrow}$ seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeatKernel}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.HeatKernelUp.HeatKernelTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not claim global analytic heat-kernel existence, host integral kernels, quotient equality of functions, selected infinite partitions, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked semigroup-row exactness, regular rational readback, real-seal nonescape, transport, replay, provenance, and local naming beyond the carrier-alignment theorem.

Chapter 6159

A Concrete Naming Certificate for $\mathsf{OsgoodCriterion}^{\uparrow}$

Following Osgood and Bishop–Bridges constructive analysis, the $\mathsf{OsgoodCriterion}^{\uparrow}$ packet records the finite criterion surface by which an ODE uniqueness request reads a divergence modulus before it may be handed to a Picard–Lindelof or Real-facing consumer. It sits beside the ODE source of chapter 193, the Picard–Lindelof handoff of chapter 1888, the Gronwall comparison surface of chapter 2034, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet names the criterion row that a uniqueness proof consumes; it is not a Lipschitz constant, not a Banach contraction theorem, not a selected solution curve, and not quotient equality of flows.

Definition 6159.1 (OsgoodCriterion carrier)**.** *An* $\mathsf{OsgoodCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, I, V, M, B, Q, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{Ode}^{\uparrow}$ *source row, carrying the time-state window, differential-equation request, and local solution-read boundary.*
2. *I is the common initial-data row read by the two solution traces whose equality is being tested.*
3. *V is the displayed vector-field continuity row on the same ODE window.*
4. *M is the Osgood modulus row, listing positive rational radii and the monotone comparison schedule used by the criterion.*
5. *B is the finite divergence-budget ledger for the reciprocal-modulus comparison; it records the checked lower windows rather than a host improper integral.*
6. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare radii, distances, reciprocal-modulus budgets, and finite error envelopes.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the solution-distance traces inspected by the criterion.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal exposed only after D, I, V, M, B, Q, R have been displayed.*
9. *H records componentwise* hsame *transport for $D, I, V, M, B, Q, R, E, C, P, N$.*
10. *C records displayed* Cont *replay from the Osgood criterion request through the ODE, initial-data, vector-field, modulus, budget, dyadic, regular-rational, and Real-seal rows.*

11. *P is the* Pkg *provenance over* $\mathsf{OsgoodCriterion}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{PicardLindelof}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{OsgoodCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, V, M, B, Q, R, E, H, C, P, N$. It has no coordinate for a Lipschitz constant, contraction ratio, selected solution representative, host improper integral, quotient equality of curves, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.OsgoodCriterionUp.TasteGate.OsgoodCriterionTasteGate single carrier alignment`

Theorem 6159.2 (OsgoodCriterion NameCert obligations)**.** *For every accepted* $\mathsf{OsgoodCriterion}^{\uparrow}$ *carrier*

$$K = (D, I, V, M, B, Q, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OsgoodCriterion}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, ODE-window visibility through D, common initial-data visibility through I, vector-field continuity visibility through V, Osgood modulus admission through M, finite divergence-budget exactness through B, dyadic tolerance exactness through Q, regular rational distance readback through R, terminal* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the criterion route without adding a Lipschitz constant, Banach contraction, host Gronwall inequality, selected solution representative, quotient equality of curves, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6159.1. The displayed rows $D, I, V, M, B, Q, R, E, H, C, P, N$ are the whole local surface, so carrier admission is exactly admission of those rows.

The ODE and initial data are read before the criterion is inspected. The row D fixes the local differential-equation window, while I fixes the common initial point for the two compared solution traces.

The criterion rows are then forced in order. The vector-field row V supplies continuity on the displayed window, M supplies the Osgood modulus schedule, and B supplies only finite reciprocal-modulus budget checks. The dyadic row Q and regular readback row R turn those checks into displayed distance comparisons.

Finally the Real seal and structural rows expose no extra analytic object. Since the carrier has no coordinate for a Lipschitz constant, contraction ratio, host improper integral, host Gronwall theorem, selected solution, quotient curve equality, countable-choice schedule, or ambient completeness theorem, the listed rows exhaust the local NameCert obligations. □

Theorem 6159.3 (OsgoodCriterion Picard handoff)**.** *Every Picard–Lindelof or ODE-uniqueness consumer of an accepted* $\mathsf{OsgoodCriterion}^{\uparrow}$ *packet must read the finite route*

$$\mathsf{Ode}^{\uparrow},\ \mathsf{PicardLindelof}^{\uparrow},\ \mathsf{GronwallInequality}^{\uparrow},\ M,\ B,\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

as displayed by Definition 6159.1. The route hands a divergence-modulus criterion to the downstream uniqueness read; it does not export a Lipschitz constant, a contraction certificate, a selected solution curve, a host integral theorem, or quotient equality between flows.

Proof. Start with a Picard-facing uniqueness request. The only ODE source available in the carrier is D, and the only common initial-data coordinate is I, so the consumer is already confined to the displayed local ODE window.

Read the comparison surface next. The Picard–Lindelof and Gronwall neighbours named in the provenance can consume the packet only after the vector-field row V, modulus row M, finite budget row B, dyadic row Q, and regular readback row R have been displayed.

The handoff reaches the Real seal E only through those rows and the structural transport, replay, provenance, and naming rows. Therefore the consumer receives the stated finite criterion route and cannot replace it by a Lipschitz hypothesis, a contraction theorem, a host integral principle, a selected solution representative, or quotient equality of solution curves. □

Falsifiable prediction: Every accepted $\mathsf{OsgoodCriterion}^{\uparrow}$ uniqueness read that reaches a $\mathsf{RegSeqRat}^{\uparrow}$ distance readback or $\mathsf{Real}^{\uparrow}$ seal displays the rows $D, I, V, M, B, Q, R, E, H, C, P, N$ first.

Independence witness: ode, picard-lindelof, gronwall-inequality, osgood-uniqueness, regseqrat, dyadicratcore, real: no listed sibling supplies the joint criterion packet containing an ODE source, common initial data, vector-field continuity row, Osgood modulus schedule, finite divergence-budget ledger, dyadic tolerance arithmetic, regular distance readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{OsgoodCriterion}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OsgoodCriterion}^{\uparrow}$ packet is generated from an ODE source row, common initial data, a vector-field continuity row, an Osgood modulus schedule, finite divergence-budget checks, DyadicRatCore tolerance arithmetic, RegSeqRat distance readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OsgoodCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6159.1, Definition 6159.2, and Definition 6159.3 seed the finite Osgood criterion route from displayed ODE data through Picard, Gronwall, modulus-budget, dyadic, regular-rational, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OsgoodCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Lipschitz uniqueness, Banach contraction, selected solution representatives, quotient equality of curves, countable choice, host improper-integral theorems, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, ODE-window visibility, common initial-data visibility, vector-field readback, Osgood modulus admission, divergence-budget exactness, dyadic tolerance exactness, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OsgoodCriterionUp.

Chapter 6160

A Concrete Naming Certificate for $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ packet records the finite NameCert bridge by which regular Cauchy data are reflected into the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits over the dyadic tolerance ledger of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real seal of chapter 38, and the source-to-completion entry route of chapter 1934. The packet names only finite windows, tolerance rows, regular readback, reflection transport, replay, provenance, and local naming; it does not claim $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, selected limits, countable choice, or a public bridge theorem.

Definition 6160.1 (Cauchy reflective embedding carrier)**.** *A* $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (C, S, Q, D, E, J, H, T, P, N)$$

with the following displayed rows:

1. *C is the regular Cauchy input row, displaying the finite regularity data requested by the consumer.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the Cauchy data are observed before any real seal is read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed windows into regular rational data.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling each requested comparison window.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Q, D have been displayed.*
6. *J is the* $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$ *entry row that records the source-to-completion handoff consumed by completion-facing readers.*
7. *H records componentwise* hsame *transport for the regular Cauchy input, window, readback, dyadic ledger, Real seal, completion-entry, replay, provenance, and local naming rows.*
8. *T records displayed* Cont *replay for consumers that inspect the regular Cauchy input, its reflected Real seal, or the completion-entry row.*
9. *P is the* Pkg *provenance over* $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{CauchyReflectiveEmbedding}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient equality of real names, selected limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable-choice schedules, host equality, or bridge credentials.*

Theorem 6160.2 (Cauchy reflective embedding NameCert obligations)**.** *For an accepted* CauchyReflectiveEmbeddi *carrier $R = (C, S, Q, D, E, J, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyReflectiveEmbedding}^{\uparrow}}$ *surface consists exactly of carrier admission, regular Cauchy input exposure, finite stream-window locality, regular-rational readback, dyadic tolerance exactness, terminal* $\mathsf{Real}^{\uparrow}$ *handoff, completion-entry handoff through J, componentwise transport, displayed replay, provenance accounting, and local naming.*

Proof. Project the carrier of Definition 6160.1. Its analytic rows are the regular Cauchy input C, window row S, readback row Q, dyadic tolerance ledger D, terminal Real seal E, and completion-entry row J.

The finite observation route is ordered. A consumer must read the window row S before the regular readback Q and dyadic ledger D are available, and the Real seal E is downstream of those displayed rows.

The completion-facing handoff is equally explicit. The row J is the only coordinate that connects the reflected Real-facing data to the source-to-completion entry route of chapter 1934.

The rows H, T, P, N preserve and name the same route. Since no carrier coordinate contains quotient equality, selected limits, ambient completeness, countable choice, host equality, or bridge credentials, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.CauchyReflectiveEmbeddingUp.TasteGate.CauchyReflectiveEmbeddingTasteGate`
`single carrier alignment`

Theorem 6160.3 (Cauchy reflective embedding Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ *carrier receives the finite route*

$$C \longmapsto S \longmapsto Q, D \longmapsto E$$

before any completion-entry row J is consumed. The route exports a reflected Real seal for displayed regular Cauchy data, not $\mathsf{Real}^{\uparrow}$ *completeness, quotient equality, selected limits, host equality, or countable choice.*

Proof. Start with the accepted packet $R = (C, S, Q, D, E, J, H, T, P, N)$. Projection exposes the regular Cauchy input row C and the finite observation row S.

The readback and tolerance rows Q, D are the only rows answering precision requests. Thus the terminal seal E is reached through the displayed regular-rational and dyadic route, not through an ambient completeness theorem.

Only after that seal has been displayed may a completion-facing consumer inspect J. The row J records the entry route already isolated by chapter 1934; it is not a quotient-completion equality or a selected host limit.

Transport, replay, provenance, and naming preserve this same route. Therefore Real-facing reads remain finite and do not escape into completeness, choice, quotient equality, or host equality claims. □

Theorem 6160.4 (Cauchy reflective embedding no-choice boundary)**.** *An accepted* $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ *carrier cannot be read as a selector of limits, a countable-choice schedule for Cauchy witnesses, or an equality proof between quotient real names. Any consumer that requests such data has left the displayed rows $C, S, Q, D, E, J, H, T, P, N$.*

Proof. Read the carrier inventory. The only analytic rows are the regular Cauchy input, finite windows, regular rational readback, dyadic tolerances, Real seal, and completion-entry handoff.

None of these rows is a selector, a quotient equality witness, or a countable-choice schedule. The structural rows transport, replay, package, and name the same inventory; they add no new analytic coordinate.

Thus every accepted consumer remains inside the finite row list. A request for selected limits, choice schedules, quotient equality, or host equality has no carrier row and no local NameCert obligation in this packet. □

Theorem 6160.5 (Cauchy reflective embedding public certificate)**.** *The public certificate exported by* $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ *is the finite route from displayed regular Cauchy data through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyCompletionEmbedding}^{\uparrow}$, hsame, Cont, Pkg, *and* NameCert. *It is a seed certificate for L10 Real and Completion consumers and not a completeness theorem, quotient-real theorem, selected-limit theorem, or bridge-checked public object.*

Proof. Combine the NameCert obligations of Definition 6160.2 with the Real handoff of Definition 6160.3. Together they force every public read through the regular Cauchy input, finite windows, readback, dyadic tolerance ledger, Real seal, and completion-entry row.

Use the no-choice boundary of Definition 6160.4. It removes selectors, quotient equalities, countable-choice schedules, host equality, and ambient Real completeness from the exported surface.

The remaining rows are exactly hsame, Cont, Pkg, and NameCert support for the same finite route. Hence the chapter exports a seed public certificate for the named L10 consumers and no stronger completeness or bridge claim. □

Falsifiable prediction: Every accepted $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ read reaches the $\mathsf{Real}^{\uparrow}$ seal only after the displayed $\mathsf{StreamName}^{\uparrow}$ window, $\mathsf{RegSeqRat}^{\uparrow}$ readback, and $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance rows are present.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchy-completion-embedding: $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries regular Cauchy input exposure, finite stream windows, rational readback, dyadic tolerance accounting, Real sealing, completion-entry routing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyReflectiveEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyReflectiveEmbedding}^{\uparrow}$ packet is generated from regular Cauchy input data, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance rows, a terminal RealUp seal, a CauchyCompletionEmbedding entry row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyReflectiveEmbedding}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6160.1, Definition 6160.2, Definition 6160.3, Definition 6160.4, and Definition 6160.5 seed the finite regular-Cauchy-to-Real-seal route for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyReflectiveEmbedding}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of real names, selected limits, countable choice, host equality, public bridge evidence, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, finite-window locality, regular-rational readback, dyadic tolerance exactness, Real handoff, completion-entry handoff, non-escape, and local naming for BEDC.Derived.CauchyReflectiveEmbeddi

Chapter 6161

A Concrete Naming Certificate for $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$

Following Dini and Bishop–Bridges constructive compactness, the $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ packet records the finite BEDC route by which a monotone filter of continuous observations over a compact metric source exposes a stable tolerance boundary. It sits after the compact metric source of chapter 129, the continuous-map ledgers of chapter 130, the uniform-modulus handoff of chapter 369, the regular rational window of chapter 301, and the real-name endpoint of chapter 38. The packet names Dini-style monotone convergence as compact finite-cover and filter-threshold data, not as host-selected limits, arbitrary open covers, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6161.1 (Compact filter Dini monotone carrier)**.** *A* $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (K, F, \epsilon, \rho, M, T, R, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only compact finite-cover and finite-net data.*
2. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, indexed by the displayed filter stages over the same compact source.*
3. *ϵ is the positive tolerance request, stored before any cover or threshold row is exported.*
4. *ρ is the shared radius or cover-budget row obtained from the compact-source and continuous-family ledgers.*
5. *M is the monotone-filter ledger, recording the displayed comparison that later stages stay below earlier stages at the requested tolerance.*
6. *T is the finite threshold row, giving the cover-facing stage after which every displayed center satisfies the tolerance boundary.*
7. *R records regular rational readback through* $\mathsf{RegSeqRat}^{\uparrow}$ *and the permitted real-name endpoint through* $\mathsf{Real}^{\uparrow}$*.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactFilterDiniMonotone}^{\uparrow}}$ *row.*

The carrier has no coordinate for host-selected limits, selected subcovers, ambient compactness, quotient real equality, countable choice, or external filter convergence.

Lean4 ✓ `BEDC.Derived.CompactFilterDiniMonotoneUp.CompactFilterDiniMonotoneTasteGate single carrier alignment`

Theorem 6161.2 (Compact filter Dini monotone obligations)**.** *For every accepted* $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ *carrier*

$$D = (K, F, \epsilon, \rho, M, T, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactFilterDiniMonotone}^{\uparrow}}$ *surface consists exactly of compact-source admission through* K*, continuous-family admission through* F*, tolerance admission through* ϵ*, shared radius or cover-budget admission through* ρ*, monotone-filter admission through* M*, finite threshold admission through* T*, regular rational and real-name readback through* R*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6161.1. The analytic rows are $K, F, \epsilon, \rho, M, T, R$, and the structural rows are P, N.

The compact and continuity obligations are exactly K and F. The row K supplies the compact finite-cover data, while F supplies the continuous family over that same source.

The Dini monotone obligations are ϵ, ρ, M, T. The tolerance is fixed first, ρ records the shared compact-cover budget, M records monotone filter comparison, and T records the finite threshold stage visible to downstream consumers.

The readback and naming obligations are R, P, N. Since the carrier contains no host-selected limit, selected subcover, ambient compactness, quotient real equality, countable choice, or external filter-convergence coordinate, the listed rows are the whole accepted NameCert surface. □

Theorem 6161.3 (Compact filter Dini monotone finite-cover route)**.** *For every accepted* $\mathsf{CompactFilterDiniMonoton}$ *carrier, any consumer of the Dini finite-cover conclusion must route its read through*

$$K, F, \epsilon \longmapsto \rho, M, T \longmapsto R, P, N.$$

Thus the compactness used by the packet is the displayed finite-cover route over $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{ContinuousMap}^{\uparrow}$*, not an ambient open-cover principle.*

Proof. Start with the carrier rows. The compact source K, continuous family F, and tolerance ϵ are the only rows visible before the cover budget and threshold rows.

The middle stage is ρ, M, T. The row ρ stores the shared radius or cover budget, M stores monotone filter comparison, and T stores the finite threshold stage after all displayed cover centers satisfy the requested boundary.

The final stage is R, P, N. Regular rational and real-name readback can inspect only the routed threshold data, and provenance and local naming preserve that same route. Therefore an accepted consumer cannot replace the route by an arbitrary open-cover compactness argument or by a selected subcover outside the packet. □

Theorem 6161.4 (Compact filter Dini monotone nonescape)**.** *For every accepted* $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ *carrier* $D = (K, F, \epsilon, \rho, M, T, R, P, N)$*, the packet cannot export a host-selected limit, selected subcover, selected subsequence, host continuity proof, external filter convergence, quotient real equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Every exported read is confined to the displayed compact-source, continuous-family, tolerance, shared-cover, monotone-filter, threshold, readback, provenance, and naming rows.*

Proof. Inspect the row list in Definition 6161.1. None of the excluded objects appears as a carrier coordinate.

Use Definition 6161.2. It identifies the complete local NameCert surface with the rows $K, F, \epsilon, \rho, M, T, R, P, N$.

Finally use the finite-cover route of Definition 6161.3. Any accepted downstream read must preserve the displayed route, so no host-selected limit, selected subcover, selected subsequence, host continuity proof, external filter convergence, quotient real equality, countable choice, or ambient completeness can be exported. □

Falsifiable prediction: Every accepted $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ read must expose compact-source admission, continuous-family admission, tolerance admission, shared cover-budget admission, monotone-filter admission, finite threshold admission, regular rational and real-name readback, provenance, and local naming.

Independence witness: compactmetric, continuousmap, uniformmodulus, regseqrat, real: CompactFilterDiniMon is not bijective to any listed sibling because it binds compact finite-cover data and monotone filter-threshold data before readback consumers may inspect the Dini boundary.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactFilterDiniMonotone}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactFilterDiniMonotone}^{\uparrow}$ packet is generated from CompactMetric finite-cover rows, ContinuousMap family rows, tolerance and shared cover-budget rows, monotone-filter comparison rows, finite threshold rows, RegSeqRat and Real readback, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactFilterDiniMonotone}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6161.1, Definition 6161.2, Definition 6161.3, and Definition 6161.4 seed the compact finite-cover route from monotone continuous-filter data to a displayed Dini threshold.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactFilterDiniMonotone}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-selected limits, selected subcovers, selected subsequences, host continuity proofs, external filter convergence, quotient real equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source admission, continuous-family admission, tolerance admission, shared cover-budget admission, monotone-filter admission, finite threshold admission, readback, provenance, and local naming for BEDC.Derived.CompactFilterDiniMonotoneUp.

Chapter 6162

A Concrete Naming Certificate for $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$

The $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ packet realizes the blocked lift in chapter 9375 as finite NameCert data. It sits between the closed-normal consistency boundary of chapter 733, the reduction-boundary packet of chapter 738, the route classifier of chapter 785, and the closed-normal packet ledger of chapter 739. The packet records endpoint, discharge, obstruction, transport, replay, provenance, and local naming rows for the unavailable reduction edge; it does not turn closed-normal consistency into full closed consistency.

Definition 6162.1 (Closed-consistency reduction obstruction carrier)**.** *A* $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, E, D, B, R, K, H, C, P, N)$$

with the following displayed rows:

1. *L is the closed source proof row, naming the arbitrary closed false-proof candidate submitted to the reduction route.*
2. *E is the endpoint row, recording the beta-strong normal endpoint that would be read by a closed-normal consistency consumer.*
3. *D is the subject-reduction discharge row whose absence blocks the arbitrary closed-proof lift.*
4. *B is the boundary row inherited from* $\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}$*.*
5. *R is the route-classifier row inherited from* $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$*, recording that any accepted route must expose the obstruction before the endpoint is consumed.*
6. *K is the closed-normal consistency consumer row, available only after the endpoint and discharge boundary have both been displayed.*
7. *H records componentwise* hsame *transport for the source, endpoint, discharge, boundary, route-classifier, consumer, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for every read from the closed source row to the endpoint, obstruction ledger, and closed-normal consumer.*
9. *P is the* Pkg *provenance over* $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$, $\mathsf{ClosedNormalConsistencyBoundary}^{\uparrow}$, $\mathsf{ClosedConsistencyReductionBoundary}^{\uparrow}$, $\mathsf{SubjectReductionRouteClassifier}^{\uparrow}$, $\mathsf{ClosedNormalConsistencyPacket}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}}$ *row.*
The carrier contains no full closed consistency theorem, no closed subject-reduction theorem, no hidden false-type transport, no host reduction semantics, and no endpoint read detached from the obstruction row.

Theorem 6162.2 (Closed-consistency reduction obstruction NameCert obligations)**.** *For an accepted* $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ *carrier* $O = (L, E, D, B, R, K, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}}$ *surface consists exactly of source exposure, endpoint exposure, subject-reduction discharge exposure, reduction-boundary inheritance, route-classifier exposure, closed-normal consumer handoff, obstruction-ledger accounting, componentwise transport, displayed replay, provenance accounting, and local naming.*

Proof. Project the carrier of Definition 6162.1. The rows L, E, D, B, R, K are the analytic rows, and H, C, P, N are the structural transport, replay, provenance, and naming rows.

The boundary and classifier rows are mandatory. The reduction-boundary row B names the blocked lift from an arbitrary closed source to the normal endpoint, while R records the subject-reduction route class that must be inspected before any endpoint consumer reads the packet.

The closed-normal consumer row K is therefore downstream of the endpoint and obstruction rows. It cannot be read as full closed consistency, because the discharge row D remains displayed as an obstruction rather than as a closed subject-reduction theorem.

No other row is available. Hence the local NameCert obligations are exactly the source, endpoint, discharge, boundary, classifier, consumer, obstruction, transport, replay, provenance, and naming rows listed in the statement. □

Theorem 6162.3 (Closed-consistency reduction obstruction frontier nonescape)**.** *Every accepted* $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ *packet forces a closed-consistency reduction read through the displayed obstruction route*

$$L \longmapsto B, R, D \longmapsto E \longmapsto K$$

before any closed-normal consistency consumer reads the endpoint. No consumer can bypass B, R, D *to obtain full closed consistency, a closed subject-reduction theorem, host conversion semantics, or hidden false-type transport.*

Proof. Start with the carrier rows. The closed source row L has no direct route to the consumer row K; the intervening rows are the reduction boundary B, route classifier R, discharge row D, and endpoint row E.

The displayed continuation row C replays that same route. It records reads from L through the boundary and classifier rows before the endpoint is presented to the closed-normal consumer.

The structural rows do not add a shortcut. Componentwise transport preserves the same row inventory, provenance names the same sibling packets, and local naming records the packet rather than a theorem outside it.

Thus the frontier cannot escape. Any attempted bypass of B, R, D asks for data absent from the carrier and is outside the NameCert surface. □

Theorem 6162.4 (Closed-consistency reduction obstruction TasteGate surface)**.** *The* ClosedConsistencyReductionO *packet exposes a finite TasteGate surface for the blocked edge: source row, endpoint row, subject-reduction discharge row, reduction-boundary row, route-classifier row, closed-normal consumer row, transport, replay, provenance, and local naming. Lean follow-up for this surface must supply* Nontrivial *and* FieldFaithful *instances in* lean4/BEDC/Derived/ClosedConsistencyReductionObstructionUp/TasteGate. *before any checked TasteGate claim is recorded in the paper.*

Proof. The carrier inventory gives the source, endpoint, discharge, boundary, classifier, and consumer rows. These are exactly the fields inspected by the chapter-level taste surface.

The rows H, C, P, N provide the structural evidence used by NameCert packets: componentwise transport, continuation replay, provenance, and local naming. They make the surface finite and auditable without adding a closed subject-reduction theorem.

The Lean follow-up named in the statement is therefore a field-faithfulness and nontriviality task for this displayed carrier. Until those instances exist, this chapter records the paper-side finite surface only and does not claim a checked TasteGate theorem. □

Falsifiable prediction: Every accepted $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ packet routes through the displayed subject-reduction obstruction row D and the route-classifier row R before any closed-normal consistency consumer reads the endpoint row E.

Independence witness: closed-normal-consistency-boundary, closed-consistency-reduction-boundary, subject-reduction-route-classifier, metacic-open-obligation-ledger: $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries arbitrary closed-source exposure, endpoint readback, the subject-reduction obstruction row, route classification, closed-normal consumer handoff, transport, replay, provenance, and local naming in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}$ packet is generated from an arbitrary closed false-proof source row, beta-strong endpoint row, subject-reduction discharge obstruction, reduction-boundary row, route-classifier row, closed-normal consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6162.1, Definition 6162.2, Definition 6162.3, and Definition 6162.4 seed the finite obstruction packet for the blocked closed-consistency reduction edge.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedConsistencyReductionObstruction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close full closed consistency, closed subject reduction, hidden false-type transport, host conversion semantics, bridge checking, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, endpoint readback, subject-reduction obstruction routing, route-classifier handoff, closed-normal consumer handoff, TasteGate instances, and local naming for BEDC.Derived.ClosedConsistencyReductionObstructionUp.

Chapter 6163

A Concrete Naming Certificate for $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$ packet records the finite interval-uniform-continuity surface for a displayed real map on a located real interval. It sits after the real-name seal of chapter 38, the endpoint-and-enclosure interval packet of chapter 4111, the located interval row of chapter 1709, the continuous-map row of chapter 130, the finite stream-window and regular-readback route of chapter 126 and chapter 301, the dyadic tolerance ledger of chapter 291, and the downstream modulus consumer of chapter 369. The packet names the finite tolerance handoff used by interval consumers; it does not claim open-cover compactness, a selected modulus from choice, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6163.1 (Real interval uniform continuity carrier)**.** *A* $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, L, F, T, W, R, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{RealInterval}^{\uparrow}$ *endpoint-and-enclosure row naming the source interval.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *locatedness row exposing endpoint brackets, locator reads, and interval-radius data for the same source interval.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row for the displayed map from the interval-facing real input row to the real output row.*
4. *T is the tolerance request row, recording the requested output precision and the interval window in which the request is read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the input names, output names, and endpoint observations visible to T.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row produced from the displayed windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger storing input-radius, output-tolerance, slack, and dyadic comparison data.*
8. *M is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row; it receives a tolerance-indexed input radius only after I, L, F, T, W, R, D have been displayed.*

9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for interval-valued input and output reads.*
10. *H records componentwise* hsame *transport for the interval, locatedness, map, tolerance, window, readback, dyadic, modulus, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a tolerance request through $I, L, F, T, W, R, D, M, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$, $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalUniformContinuity}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one displayed interval-uniform-continuity history. It has no coordinate for open-cover compactness, selected representatives, quotient real equality, host continuity, choice-selected moduli, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6163.2 (Real interval uniform continuity NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$ *carrier $U = (I, L, F, T, W, R, D, M, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalUniforr}}$ *surface consists exactly of carrier admission for the displayed interval, locatedness, continuous-map, tolerance-request, stream-window, regular-readback, dyadic-ledger, uniform-modulus, real-seal, transport, replay, provenance, and local naming rows.*

The obligation surface additionally requires the interval row I and locatedness row L to precede any input-radius read, the continuous-map row F to precede any output-tolerance read, and the dyadic ledger D to precede the $\mathsf{UniformModulus}^{\uparrow}$ *handoff M. No local obligation is stored outside those rows.*

```
Lean4 ✓  BEDC.Derived.RealIntervalUniformContinuityUp.RealIntervalUniformContinuityTasteGate
single carrier alignment
```

Proof. Project the accepted carrier. The packet contains only the rows $I, L, F, T, W, R, D, M, E, H, C, P, N$, so carrier admission and local naming are componentwise.

The source interval is read through I and L. These rows expose endpoint, enclosure, and locatedness data before any input-radius comparison can be treated as interval uniformity evidence.

The map and tolerance route is then finite. The row F exposes the displayed continuous map, T records the requested output precision, W and R expose the visible stream windows and regular rational observations, and D records the dyadic comparison budget.

Only after that route is displayed may M be read as a $\mathsf{UniformModulus}^{\uparrow}$ handoff. The rows E, H, C, P, N seal, transport, replay, source, and name the same finite route, while the excluded compactness, quotient, choice, and completeness objects have no carrier coordinate. □

Theorem 6163.3 (Real interval uniform continuity modulus handoff)**.** *Every uniform-modulus consumer of an accepted* $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$ *packet must pass through the displayed route*

$$\mathsf{RealInterval}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{UniformM}$$

Thus the consumer receives a tolerance-indexed input radius for the displayed located real interval and continuous map, without receiving open-cover compactness, a choice-selected modulus, quotient real equality, selected endpoint representatives, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier in Definition 6163.1. The interval row I, locatedness row L, and continuous-map row F are the only rows that identify the source, the interval boundary, and the map.

Next read the tolerance data. The request row T fixes the output precision, W supplies the finite windows, R supplies regular rational readback, and D supplies the dyadic comparison ledger.

The handoff row M is downstream of those rows. Therefore any accepted $\mathsf{UniformModulus}^{\uparrow}$ consumer receives its input-radius evidence only through the displayed interval, map, window, readback, and dyadic route.

The structural rows E, H, C, P, N preserve that route and add no hidden selector. Since no excluded object is present in the carrier, the handoff cannot export open-cover compactness, quotient real equality, endpoint representatives, choice-selected moduli, or ambient completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalUniformContinuity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalUniformContinuity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6163.1, Definition 6163.2, and Definition 6163.3 seed the finite interval-uniform-continuity handoff over $\mathsf{RealInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalUniformContinuity}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.RealIntervalUniformContinuityUp.RealIntervalUniformContinuityTasteGate\_single\_carrier`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected modulus existence, quotient real equality, selected endpoint representatives, trichotomy, countable choice, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked interval locatedness, continuous-map tolerance readback, dyadic exactness, uniform-modulus handoff, Real seal non-escape, and local naming beyond the carrier-alignment theorem.

Constructive story: A $\mathsf{RealIntervalUniformContinuity}^{\uparrow}$ packet is generated from a RealIntervalUp source interval, a LocatedRealIntervalUp locatedness row, a ContinuousMapUp map row, tolerance requests, StreamNameUp finite windows, RegSeqRatUp readback, a DyadicRatCoreUp ledger, a UniformModulusUp handoff, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6164

A Concrete Naming Certificate for $\mathsf{ArithmeticHierarchy}^{\uparrow}$

Following Kleene's recursion-theory hierarchy, the $\mathsf{ArithmeticHierarchy}^{\uparrow}$ packet records the finite BEDC route by which bounded first-order syntax, computability evidence, and oracle-free quantifier alternation data are named before a Real or Completion consumer may cite arithmetical complexity. It sits beside the syntactic carrier of chapter 197, the computability interface of chapter 199, the set-membership discipline of chapter 3668, the lambda-calculus syntax route of chapter 200, and the constructive-real boundary of chapter 336. The downstream use is the same finite obstruction surface read by chapter 3516: a Specker or computable-real boundary may cite a displayed arithmetical class, but it cannot import a host predicate hierarchy, quotient equality of predicates, excluded middle, countable choice, or an ambient completion theorem.

Definition 6164.1 (ArithmeticHierarchy carrier)**.** *An* $\mathsf{ArithmeticHierarchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (S, E, Q, B, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{FirstOrder}^{\uparrow}$ *syntax row containing the bounded formula spine, free-variable ledger, and term source read by the classifier.*
2. *E is the* $\mathsf{Computable}^{\uparrow}$ *effective-evaluation row for atomic and bounded subformula reads.*
3. *Q is the quantifier-prefix row listing a finite alternation word over existential and universal blocks, with its initial polarity and displayed block boundaries.*
4. *B is the bound ledger recording which quantifiers are bounded and which finite rows expose the requested numeric or finite-set search windows.*
5. *R is the rank row naming the local Σ^0_n, Π^0_n, or Δ^0_n class as finite data rather than as a host predicate family.*
6. *L is the stability ledger asserting that prefix padding, bounded-quantifier expansion, and componentwise formula transport preserve the displayed rank row.*
7. *H records componentwise* hsame *transport for $S, E, Q, B, R, L, C, P, N$.*
8. *C records displayed* Cont *replay from the syntax and computability rows through prefix, bound, rank, and stability reads.*

9. *P is the* Pkg *provenance over* ArithmeticalHierarchy$^\uparrow$, FirstOrder$^\uparrow$, Computable$^\uparrow$, SetLike$^\uparrow$, LambdaCalc$^\uparrow$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArithmeticalHierarchy}^\uparrow}$ *row.*

The classifier compares accepted packets only through $S, E, Q, B, R, L, H, C, P, N$. It has no coordinate for a host subset of $\mathbb{N}$, truth in the standard model, quotient equality of predicates, oracle jumps, countable choice, excluded middle, Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.ArithmeticalHierarchyUp.ArithmeticalHierarchyTasteGate single carrier alignment`

Theorem 6164.2 (ArithmeticalHierarchy NameCert obligations)**.** *For every accepted* ArithmeticalHierarchy$^\uparrow$ *carrier*

$$A = (S, E, Q, B, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArithmeticalHierarchy}^\uparrow}$ *surface consists exactly of carrier admission for the displayed rows, syntax visibility through S, effective atomic and bounded reads through E, quantifier-prefix admission through Q, bound-ledger exactness through B, rank-row exactness through R, stability under displayed prefix padding and bounded expansion through L, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

These obligations bind the arithmetical-class route without adding host predicate semantics, quotient predicate equality, oracle jumps, unrestricted comprehension, excluded middle, countable choice, or an ambient Real$^\uparrow$ *completion claim.*

Proof. Project the carrier of Definition 6164.1. The rows $S, E, Q, B, R, L, H, C, P, N$ are the whole local surface, so carrier admission is exactly admission of those rows.

The syntactic and effective data are read first. The row S fixes the first-order formula spine and free-variable ledger, while E supplies only the effective atomic and bounded reads available to the packet.

The hierarchy data are then forced by the displayed prefix. The row Q lists the finite alternation word, B records the bounded search windows, and R names the resulting Σ^0_n, Π^0_n, or Δ^0_n rank. The stability row L is the only place where padding or bounded expansion may be replayed.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for host truth, predicate quotients, oracle jumps, unrestricted comprehension, choice, excluded middle, or Real completeness, the listed clauses exhaust the local NameCert obligations. □

Theorem 6164.3 (ArithmeticalHierarchy computable-real boundary handoff)**.** *Every computable-real, Specker-sequence, or completion-boundary consumer of an accepted* ArithmeticalHierarchy$^\uparrow$ *packet must read the finite route*

$$\mathsf{FirstOrder}^\uparrow,\ \mathsf{Computable}^\uparrow,\ Q,\ B,\ R,\ L,\ \mathsf{SetLike}^\uparrow$$

as displayed by Definition 6164.1. The route exports a named arithmetical class for the downstream boundary read; it does not export host truth, a predicate quotient, an oracle jump, a selected completion point, countable choice, excluded middle, or an ambient Real$^\uparrow$ *completeness theorem.*

Proof. Start with a Real-facing or Specker-facing boundary read. The only syntax source available in the carrier is S, and the only effective-evaluation row is E, so the consumer is confined to the displayed first-order and computability data.

Read the classification surface next. The prefix row Q, bound row B, rank row R, and stability row L together expose the finite arithmetical-class route, while the set-like provenance can be cited only through P and the displayed replay C.

The structural rows do not add a hidden semantic object. Because no host predicate hierarchy, quotient equality, oracle jump, selected Real point, countable-choice schedule, excluded-middle principle, or ambient completeness theorem appears as a carrier coordinate, the consumer receives only the stated finite route. □

Falsifiable prediction: Every accepted $\mathsf{ArithmeticalHierarchy}^{\uparrow}$ boundary read that reaches a $\mathsf{ComputableReal}^{\uparrow}$, $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, or completion-facing consumer displays the rows $S, E, Q, B, R, L, H, C, P, N$ first.

Independence witness: firstorder, computable, setlike, lambdacalc, computablereal, specker-sequence-boundary: no listed sibling supplies the joint packet containing bounded first-order syntax, effective atomic and bounded reads, quantifier-prefix data, bound ledgers, rank rows, stability transport, displayed replay, provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmeticalHierarchy}^{\uparrow}}$.

Constructive story: An $\mathsf{ArithmeticalHierarchy}^{\uparrow}$ packet is generated from a FirstOrderUp syntax row, a ComputableUp effective-evaluation row, finite quantifier-prefix data, a bounded-search ledger, a displayed rank row, stability under prefix padding and bounded expansion, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ArithmeticalHierarchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6164.1, Definition 6164.2, and Definition 6164.3 seed the finite arithmetical-class route from displayed first-order and computability rows through prefix, bound, rank, stability, and boundary-consumer reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ArithmeticalHierarchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem checking, bridge checking, host predicate semantics, quotient predicate equality, oracle jumps, unrestricted comprehension, countable choice, excluded middle, selected Real points, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, syntax visibility, effective readback, quantifier-prefix admission, bound-ledger exactness, rank-row exactness, stability transport, displayed replay, provenance, and local naming for BEDC.Derived.ArithmeticalHierarchyUp.

Chapter 6165

A Concrete Naming Certificate for $\mathsf{HilbertCurve}^{\uparrow}$

Beginning with Hilbert's classical space-filling curve construction, the $\mathsf{HilbertCurve}^{\uparrow}$ packet records the finite dyadic subdivision route by which interval addresses are folded into square-cell addresses. It sits beside the Peano surface of chapter 2078, the compact uniform-modulus surface of chapter 2299, the graph carrier of chapter 130, and the compact metric completion route of chapter 1687. The packet names the recursive quadrant order, orientation bit, dyadic coordinate readback, and finite continuity budget; it does not claim an unrestricted function space, arbitrary pointwise choice, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6165.1 (Hilbert curve carrier)**.** *A* $\mathsf{HilbertCurve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (A, Q, O, F, X, Y, B, T, C, K, N)$$

with the following displayed rows:

1. *A is the interval-address row. It records finite dyadic subintervals, their precision levels, and the associated* $\mathsf{StreamName}^{\uparrow}$ *observation windows.*
2. *Q is the quadrant row. For each displayed address, it records the selected dyadic square quadrant and its horizontal and vertical dyadic endpoints.*
3. *O is the orientation row. It records the finite turn state used by Hilbert's recursive subdivision rule, including the two endpoint orientations needed to compare sibling cells.*
4. *F is the fold-coherence row. It checks that the four child quadrants of each displayed parent square are visited in the Hilbert order and that consecutive children share an edge or an endpoint at the displayed precision.*
5. *X and Y are the coordinate readback rows. They expose the horizontal and vertical dyadic endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$*, then through the regular rational and real-name support already used by continuous-map consumers.*
6. *B is the finite-window continuity row. A square precision request is answered by an interval precision, a dyadic diameter bound, and the fold-coherence witness needed before a* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer reads the graph.*
7. *T is the compact-metric handoff row. It packages the finite square-cell cover and the dyadic diameter budget as compact metric data without using open-cover compactness.*

8. *C records componentwise* hsame *transport for interval addresses, quadrants, orientations, fold coherence, coordinate readbacks, continuity budgets, compact-metric handoffs, provenance, and local naming.*

9. *K records displayed* Cont *replay through*

$$A, Q, O, F \longmapsto X, Y, B, T \longmapsto C, K, N.$$

10. *N is the local* $\mathsf{NameCert}_{\mathsf{HilbertCurve}^\uparrow}$ *row, sourced over* $\mathsf{HilbertCurve}^\uparrow$, $\mathsf{StreamName}^\uparrow$, $\mathsf{DyadicRatCore}^\uparrow$, $\mathsf{ContinuousMap}^\uparrow$, BHist, hsame, Cont, Pkg, *and* NameCert.

The classifier compares accepted packets only through $A, Q, O, F, X, Y, B, T, C, K, N$. It has no coordinate for a completed function space, arbitrary representative choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^\uparrow$ *completeness, Lean verification, or a bridge credential.*

Lean4 ✓ `BEDC.Derived.HilbertCurveUp.HilbertCurveTasteGate single carrier alignment`

Theorem 6165.2 (Hilbert curve NameCert obligations)**.** *For an accepted* $\mathsf{HilbertCurve}^\uparrow$ *carrier*

$$H = (A, Q, O, F, X, Y, B, T, C, K, N),$$

the local $\mathsf{NameCert}_{\mathsf{HilbertCurve}^\uparrow}$ *surface consists exactly of six finite obligation rows:*

1. *carrier admission for the interval-address, quadrant, orientation, fold-coherence, coordinate readback, finite-window continuity, compact-metric handoff, transport, replay, provenance, and local naming rows;*

2. *classifier exactness, requiring accepted packets to be compared only by the rows $A, Q, O, F, X, Y, B, T, C, K, N$;*

3. *subdivision coherence, requiring the quadrant and orientation rows to display the Hilbert child order before any coordinate endpoint is exposed;*

4. *finite-window continuity, requiring the fold-coherence and diameter-budget rows to answer every displayed graph request through a finite dyadic window;*

5. *compact-metric handoff, requiring the square-cell cover and dyadic diameter budget to be consumed through compact metric data rather than open-cover compactness;*

6. *ledger exhaustion, requiring C, K, N to transport, replay, source, and name only the listed finite dyadic route.*

No NameCert obligation is stored outside these rows, and none exports a host-level function space, arbitrary pointwise choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^\uparrow$ *completeness, Lean verification, or a standard bridge.*

Proof. Carrier admission is projection from Definition 6165.1. The accepted packet displays the interval-address row, quadrant row, orientation row, fold-coherence row, coordinate readback rows, finite-window continuity row, compact-metric handoff row, and structural rows.

Classifier exactness follows because the definition gives the classifier no coordinate except $A, Q, O, F, X, Y, B, T, C, K, N$. Any attempted comparison through a function space, representative choice, quotient equality, open-cover compactness, completion principle, verification credential, or bridge credential therefore has no row to inspect.

Subdivision coherence is read from Q, O, F. The quadrant row names the selected square cell, the orientation row records the finite turn state, and the fold-coherence row checks the four-child Hilbert order before X or Y can expose an endpoint.

Finite-window continuity is read from F and B. The fold-coherence row supplies the edge or endpoint adjacency check for neighboring interval cells, while the budget row supplies the dyadic diameter bound requested by a continuous-map consumer.

The compact-metric handoff is read from T. It repackages the displayed finite square-cell cover and diameter budget as compact metric data, so no open-cover argument or ambient completeness principle is introduced.

Finally C, K, N add no analytic coordinate. They preserve, replay, source, and name the same finite route, which exhausts the local NameCert surface. □

Theorem 6165.3 (Hilbert curve finite-window continuity)**.** *Let*

$$H = (A, Q, O, F, X, Y, B, T, C, K, N)$$

be an accepted $\mathsf{HilbertCurve}^{\uparrow}$ *carrier. Every continuous-map or compact-metric consumer of* H *receives the Hilbert curve through the finite route*

$$A, Q, O, F \longmapsto X, Y, B, T \longmapsto C, K, N.$$

For each requested square precision, the route opens a finite $\mathsf{StreamName}^{\uparrow}$ *interval window, reads dyadic quadrant endpoints through* $\mathsf{DyadicRatCore}^{\uparrow}$*, checks sibling fold coherence by* F*, answers the diameter budget by* B*, and then hands the resulting finite graph surface to* $\mathsf{ContinuousMap}^{\uparrow}$ *and compact metric support through* T*. Thus the dyadic window handoff stays inside the finite stream, dyadic rational, and continuous-map support named by the carrier.*

Proof. Fix a requested square precision. The interval-address row A supplies only finitely many dyadic input windows, and the quadrant row Q assigns each displayed address to a dyadic square cell.

The coordinate readback is forced through X and Y. Those rows expose the horizontal and vertical endpoints as dyadic rational data before any continuous-map consumer can read the graph.

Sibling compatibility is a finite check. The orientation row O records the turn state, and the fold-coherence row F verifies that consecutive displayed child cells share an edge or endpoint at the requested precision.

The budget row B then supplies the diameter bound associated with the same displayed window. Consequently the graph read by $\mathsf{ContinuousMap}^{\uparrow}$ is a finite-window surface rather than a host-level function graph.

The compact handoff is the row T. It carries the displayed square-cell cover and the same dyadic diameter budget into compact metric support, while C, K, N preserve the route under transport, replay, provenance, and local naming. □

Closure status for $\mathsf{NameCert}_{\mathsf{HilbertCurve}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HilbertCurve}^{\uparrow}$ packet is generated from finite dyadic interval addresses, quadrant cells, orientation states, fold-coherence rows, coordinate readbacks through DyadicRatCore support, finite-window continuity budgets, compact-metric handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HilbertCurve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6165.1, Definition 6165.2, and Definition 6165.3 seed the finite dyadic subdivision route for a Hilbert curve packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HilbertCurve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host-level function spaces, arbitrary pointwise choice, quotient equality of real names, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier exactness, subdivision coherence, finite-window continuity, compact-metric handoff, and ledger exhaustion for BEDC.Derived.HilbertCurveUp.

Chapter 6166

A Concrete Naming Certificate for $\mathsf{CauchyProductBoundary}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CauchyProductBoundary}^{\uparrow}$ packet records the finite boundary row needed before a Cauchy product read is handed to the regular-rational and real-name consumers. It sits after the product modulus ledger of **??**, the regular product criterion of chapter 3672, and the synchronized criterion of chapter 1586. The packet names a finite product-modulus handoff surface, not completed-real multiplication, quotient stream equality, a selected product limit, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6166.1 (Cauchy product boundary carrier)**.** *A* $\mathsf{CauchyProductBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (A, B, W_A, W_B, D, M, K, R, E, H, C, P, N)$$

with the following displayed rows:

1. *A and B are the two accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *source rows whose product is being read.*
2. *W_A and W_B are synchronized* $\mathsf{StreamName}^{\uparrow}$ *finite windows opened at the requested product tolerance.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-scaling row, recording the source-window tolerances, magnitude bounds, and cross-error budget needed by the product request.*
4. *M is the* $\mathsf{CauchyProductModulus}^{\uparrow}$ *budget row assigning the finite source-window budgets for the product read.*
5. *K is the* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *row certifying that the displayed regular source rows and budget data produce a regular Cauchy product handoff.*
6. *R is the* $\mathsf{CauchyProductCriterion}^{\uparrow}$ *real-seal handoff row.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, read only after M, K, R have been displayed.*
8. *H stores componentwise* hsame *transports for the source rows, windows, dyadic budget, product-modulus row, criterion rows, seal, provenance, and local name.*
9. *C stores displayed* Cont *replay from a product precision request through the windows, dyadic scaling row, product modulus, regular criterion, real-seal criterion, and terminal seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyProductBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyProductModulus}^{\uparrow}$, $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$, $\mathsf{CauchyProductCriterion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyProductBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, B, W_A, W_B, D, M, K, R, E, H, C, P, N$. It has no coordinate for host real multiplication, quotient stream equality, selected representatives, countable choice, trichotomy, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CauchyProductBoundaryUp.CauchyProductBoundaryTasteGate single carrier alignment`

Theorem 6166.2 (Cauchy product boundary bilinear modulus)**.** *For an accepted* $\mathsf{CauchyProductBoundary}^{\uparrow}$ *carrier*

$$Q = (A, B, W_A, W_B, D, M, K, R, E, H, C, P, N),$$

every bilinear product tolerance exported to a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer factors through the finite boundary route*

$$A, B \longmapsto W_A, W_B \longmapsto D \longmapsto M \longmapsto K \longmapsto R \longmapsto E.$$

Thus the product boundary supplies a modulus-mediated handoff from regular rational source rows to the terminal real seal, while refusing completed-real multiplication, quotient stream equality, selected product limits, countable choice, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6166.1. The only source observations available to a product request are the two $\mathsf{RegSeqRat}^{\uparrow}$ source rows and the synchronized $\mathsf{StreamName}^{\uparrow}$ finite windows W_A, W_B.

Read the $\mathsf{DyadicRatCore}^{\uparrow}$ row D. It performs the tolerance scaling for the bilinear product request by displaying the source tolerances, magnitude bounds, and cross-error budget inside finite dyadic data. The product request therefore cannot reach a downstream criterion before this scaling row is visible.

Next apply the $\mathsf{CauchyProductModulus}^{\uparrow}$ budget row M. It allocates the finite source-window budgets consumed by the product read, so the regular criterion receives a displayed modulus row rather than an external choice of convergence rate.

The $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ row K turns the window and modulus data into the regular rational product handoff. The $\mathsf{CauchyProductCriterion}^{\uparrow}$ row R then routes that handoff to the real-seal boundary, and E is available only after this route has been displayed.

Finally, H, C, P, N transport, replay, source, and name the same finite boundary. Since the carrier has no coordinate for quotient streams, selected limits, countable choice, trichotomy, host multiplication, or ambient completeness, every bilinear product tolerance exported by the packet remains inside the displayed route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyProductBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyProductBoundary}^{\uparrow}$ packet is generated from two RegSeqRat source rows, synchronized StreamName windows, a DyadicRatCore tolerance-scaling row, a CauchyProductModulus budget row, RegularCauchyProductCriterion and CauchyProductCriterion handoff rows, a RealUp seal row, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyProductBoundary}^{\uparrow}}$, seed)

Scope closed: Definition 6166.1 and Definition 6166.2 seed the finite boundary route from two regular rational source rows through StreamName windows, dyadic tolerance scaling, product modulus budgeting, product criteria, and the Real seal.

Formal verification: FormalStatus(NameCert$_{\mathsf{CauchyProductBoundary}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host real multiplication, quotient stream equality, selected product limits, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, synchronized-window exposure, dyadic tolerance scaling, product-modulus budgeting, regular product criterion handoff, Cauchy product criterion handoff, Real seal non-escape, transport, replay, provenance, and local naming for the same finite packet.

Chapter 6167

A Concrete Naming Certificate for $\mathsf{UniformLipschitzFamily}^{\uparrow}$

Following Arzela-Ascoli and the standard Lipschitz-family compactness literature, the $\mathsf{UniformLipschitzFamily}^{\uparrow}$ packet records the finite BEDC surface for a family of continuous maps controlled by one displayed Lipschitz bound across all listed members. It sits after the metric source of chapter 57, the continuous-map graph rows of chapter 130, the bound ledger of chapter 2328, the compact source of chapter 129, and the tolerance consumer of chapter 369. The packet names uniform Lipschitz control as finite family data; it does not assert ambient function-space compactness, select a convergent subsequence, or import open-cover compactness.

Definition 6167.1 (Uniform Lipschitz-family carrier)**.** *A* $\mathsf{UniformLipschitzFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, Y, I, F, L, V, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite metric windows supply all source-distance reads.*
2. *Y is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance reads are used only through displayed continuous-map values.*
3. *I is the finite family-index row naming the inspected maps.*
4. *F is the finite list of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, one for each displayed family member from X to Y.*
5. *L is the shared* $\mathsf{LipschitzConstant}^{\uparrow}$ *row, recording one common bound for every listed member rather than one hidden ambient bound per function.*
6. *V is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the common Lipschitz bound as a tolerance-only modulus surface.*
7. *H records componentwise* hsame *transport for X, Y, I, F, L, V and the structural rows.*
8. *C records displayed* Cont *replay of the route from compact source windows through continuous-map values and the shared Lipschitz bound to the uniform-modulus handoff.*
9. *P is the* Pkg *provenance over* $\mathsf{UniformLipschitzFamily}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{LipschitzConstant}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformLipschitzFamily}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient function-space topology, selected subsequence, quotient equality of functions, open-cover compactness, or a non-displayed modulus.

Lean4 ✓ `BEDC.Derived.UniformLipschitzFamilyUp.TasteGate.UniformLipschitzFamilyTasteGate single carrier alignment`

Theorem 6167.2 (Uniform Lipschitz-family NameCert obligations)**.** *For every accepted* $\mathsf{UniformLipschitzFamily}^{\uparrow}$ *carrier* $U = (X, Y, I, F, L, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformLipschitzFamily}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact source, target metric, family-index, continuous-map list, shared Lipschitz bound, uniform-modulus handoff, transport, replay, provenance, and local naming rows;*
2. *compact-source exposure, requiring every family read to pass through* X *before any pointwise value or bound row is consumed;*
3. *continuous-map exposure, requiring each listed family member in* I *to be read through a corresponding* $\mathsf{ContinuousMap}^{\uparrow}$ *row in* F*;*
4. *shared-bound exactness, requiring the single row* L *to control every listed member and forbidding a hidden per-member or ambient Lipschitz selector;*
5. *uniform-modulus handoff, requiring* V *to consume only the common bound row together with the displayed source and target metric evidence;*
6. *structural stability, requiring* H, C, P, N *to transport, replay, source, and name only the rows already listed.*

No uniform Lipschitz-family obligation is stored outside these rows.

Proof. Project the accepted packet $U = (X, Y, I, F, L, V, H, C, P, N)$. The displayed coordinates give the compact source, target metric, finite index row, listed continuous-map rows, shared Lipschitz bound, uniform-modulus handoff, and structural rows.

The source and map rows come first. The compact metric row X supplies source-distance windows, while I and F restrict inspection to the listed continuous maps.

The common bound is then read through L. Since the same row is used for every listed member, the certificate records a family-level Lipschitz constraint rather than a collection of host-level function predicates.

Finally V, H, C, P, N package the same displayed route for downstream consumers. The excluded objects do not occur as carrier coordinates, so the listed rows exhaust the local NameCert obligations. □

Theorem 6167.3 (Uniform Lipschitz-family compact-uniform handoff)**.** *Every compact-uniform or Arzela–Ascoli consumer of an accepted* $\mathsf{UniformLipschitzFamily}^{\uparrow}$ *carrier receives the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{LipschitzConstant}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

Thus the shared Lipschitz bound is exported only as compact-source metric data, displayed map rows, a common LipschitzConstant row, and a UniformModulus handoff. It exports no ambient function-space topology, selected subsequence, open-cover compactness, quotient function equality, or non-displayed modulus.

Proof. Start with the source and target metric rows. The compact source X and target metric Y provide the only distance reads available to the family.

Read the finite family rows I and F. Each inspected member is attached to a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row, so a consumer cannot quantify over an ambient function space.

The shared bound row L then feeds V. The downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer receives the common Lipschitz ledger as tolerance data, while H, C, P, N transport, replay, source, and name that same route.

Because the carrier lacks coordinates for subsequence selection, open-cover compactness, quotient function equality, and hidden moduli, every compact-uniform or Arzela–Ascoli read factors through the displayed handoff. □

Theorem 6167.4 (Uniform Lipschitz-family obligation surface)**.** *For every accepted* $\mathsf{UniformLipschitzFamily}^{\uparrow}$ *carrier* $U = (X, Y, I, F, L, V, H, C, P, N)$*, the displayed rows jointly discharge the obligation surface needed for obligation closure: compact-source exposure, continuous-map exposure, shared-bound exactness,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff,* hsame *transport,* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier rows $X, Y, I, F, L, V, H, C, P, N$. The compact-source and continuous-map obligations are exactly the exposure rows isolated in Definition 6167.2.

The shared-bound and modulus obligations are then read through L and V. Apply Definition 6167.2 to identify the single common Lipschitz row and Definition 6167.3 to route that row into the $\mathsf{UniformModulus}^{\uparrow}$ handoff.

Finally H, C, P, N supply transport, replay, provenance, and local naming for the same displayed packet. No additional ambient function-space, subsequence, open-cover, quotient-equality, or hidden-modulus row remains to be discharged. □

Falsifiable prediction: An accepted $\mathsf{UniformLipschitzFamily}^{\uparrow}$ read is refuted if a compact-uniform or Arzela–Ascoli consumer obtains a UniformModulusUp handoff without displaying the CompactMetric source row, target Metric row, finite family-index row, ContinuousMap rows, shared LipschitzConstant row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: metric, continuousmap, lipschitz-constant, compactmetric, uniform-modulus: no listed sibling is bijective with $\mathsf{UniformLipschitzFamily}^{\uparrow}$ because the packet jointly carries a compact source, target metric, finite family index, displayed continuous-map list, one shared Lipschitz bound, a uniform-modulus handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLipschitzFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLipschitzFamily}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6167.1, Definition 6167.2, Definition 6167.3, and Definition 6167.4 close the displayed obligation surface for the finite shared-Lipschitz family route from compact metric source data through continuous-map rows to a UniformModulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLipschitzFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function-space compactness, selected subsequences, open-cover compactness, quotient function equality, non-displayed moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-map exposure, shared-bound exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformLipschitz

Constructive story: A $\mathsf{UniformLipschitzFamily}^{\uparrow}$ packet is generated from a compact metric source row, target Metric row, finite family-index row, ContinuousMap rows, a shared LipschitzConstant row, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6168

A Concrete Naming Certificate for $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ packet records the finite total-boundedness route for a displayed located real interval before compact-interval and uniform-continuity consumers may read a finite net. It sits after the endpoint surface of chapter 394, the located real interval carrier of chapter 1709, and the finite epsilon-net certificate of chapter 1767. The packet is an L10 finite-net interval object; it does not claim open-cover Heine–Borel, a choice-selected subcover, quotient real equality, endpoint trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6168.1 (Real interval total boundedness carrier)**.** *A* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (I, L, \varepsilon, M, E, F, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *endpoint row exposing the lower endpoint, upper endpoint, and interval-order boundary.*
2. *L is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *row for the same displayed interval, carrying real-name enclosure and locatedness data.*
3. *ε is a positive* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance ledger read as the requested net radius.*
4. *M is a finite mesh row whose cells refine the located interval at tolerance ε.*
5. *E is the finite list of displayed centers, each center read through the same located real interval rows I, L.*
6. *F is the* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *handoff row asserting that every inspected interval point is covered by one center from E at radius ε.*
7. *H records componentwise* hsame *transport for the interval, locatedness, tolerance, mesh, centers, net handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a total-boundedness request through $I, L, \varepsilon, M, E, F, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for arbitrary open covers, selected finite subcovers, quotient real equality, endpoint trichotomy, countable choice, host compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealIntervalTotalBoundednessUp.TasteGate.RealIntervalTotalBoundednessTasteGate single carrier alignment`

Theorem 6168.2 (Real interval total boundedness NameCert obligations)**.** *For every accepted* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ *carrier* $T = (I, L, \varepsilon, M, E, F, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^{\uparrow}}$ *surface consists exactly of carrier admission, located endpoint admission, located real interval admission, positive tolerance admission, finite mesh admission, finite center admission, finite epsilon-net handoff, componentwise transport, replay, provenance, and local naming for the displayed rows* $I, L, \varepsilon, M, E, F, H, C, P, N$*.*

Proof. Project the accepted packet $T = (I, L, \varepsilon, M, E, F, H, C, P, N)$. Carrier admission reads the located endpoint row I, the located real interval row L, the positive tolerance ledger ε, the mesh row M, the center list E, and the finite-net handoff F.

The endpoint and located-real rows precede the net route. A point request first enters through I and L, then reads the tolerance ledger and mesh; no host interval, quotient representative, or endpoint trichotomy is selected.

The finite handoff is the row F. It is transported only through H and replayed only through C, while P records the sibling certificate provenance and N names the local packet.

Since the carrier contains no open-cover, subcover-selection, countable-choice, or ambient completeness coordinate, no $\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^{\uparrow}}$ consumer can recover those objects from the packet. □

Theorem 6168.3 (Real interval total boundedness finite-net handoff)**.** *Every compact-interval, completion, or uniform-continuity consumer of an accepted* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ *packet receives total boundedness only through the displayed route*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{LocatedRealInterval}^{\uparrow},\ \mathsf{Rat}^{\uparrow}\ \textit{or}\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{FiniteEpsilonNet}^{\uparrow}.$$

Thus the consumer receives a finite list of interval centers and a tolerance-indexed coverage row for the displayed located real interval, without receiving open-cover Heine–Borel, a selected subcover, quotient real equality, endpoint trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the carrier of Definition 6168.1. The rows I and L are the only interval source, and ε is the only tolerance request visible to the consumer.

The mesh and center rows M, E are finite data. They feed the $\mathsf{FiniteEpsilonNet}^{\uparrow}$ handoff F, so coverage is read as a displayed net over the same located real interval rather than as a subcover of an arbitrary open cover.

Transport and replay do not widen the boundary. The rows H, C, P, N preserve the same finite route and add provenance and local naming, while the excluded compactness, choice, quotient, trichotomy, and completeness objects have no carrier coordinate. □

Theorem 6168.4 (Real interval total boundedness mesh refinement)**.** *Let* $T = (I, L, \varepsilon, M, E, F, H, C, P, N)$ *be an accepted* $\mathsf{RealIntervalTotalBoundedness}^{\uparrow}$ *packet. For every displayed smaller positive tolerance request* δ *admitted by the same* $\mathsf{Rat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *tolerance ledger with* $\delta \le \varepsilon$*, the refined request factors through a finite mesh subdivision* M_δ *and center list* E_δ *over the same located interval rows* I, L*, and then through the same* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *handoff pattern used by* F*.*

Every read of the refined packet remains visible through $I, L, \varepsilon, M, E, F, H, C, P, N$ together with the displayed subdivision witnesses M_δ, E_δ. The refinement supplies no open-cover Heine–Borel theorem, no selected subcover, no quotient real equality, no endpoint trichotomy, no countable choice, and no ambient $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 6168.1. The only interval rows available to the smaller request are still I and L, and the only tolerance ledger already admitted by the packet is ε.

Use the finite-net handoff of Definition 6168.3. A smaller positive request is served by subdividing the displayed mesh row into a finite refinement M_δ, choosing its displayed center list E_δ, and reading coverage through the same finite handoff pattern as F.

The structural rows are unchanged. Componentwise transport still passes through H, replay through C, provenance through P, and local naming through N.

Since the carrier and its refinement rows remain finite mesh data over $I, L, \varepsilon, M, E, F, H, C, P, N$, none of the excluded open-cover, selected-subcover, quotient, trichotomy, choice, or ambient-completeness principles is introduced by the refined tolerance request. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^\uparrow}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^\uparrow}, \mathsf{seed})$

Scope closed: Definition 6168.1, Definition 6168.2, and Definition 6168.3 seed the finite total-boundedness route over $\mathsf{LocatedInterval}^\uparrow$, $\mathsf{LocatedRealInterval}^\uparrow$, $\mathsf{FiniteEpsilonNet}^\uparrow$, $\mathsf{Rat}^\uparrow$, and $\mathsf{Real}^\uparrow$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealIntervalTotalBoundedness}^\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover Heine–Borel, selected finite subcovers, quotient real equality, endpoint trichotomy, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located interval admission, positive tolerance exactness, finite mesh coverage, finite epsilon-net handoff, non-escape, and local naming for BEDC.Derived.RealIntervalTotalBoundednessUp.

Constructive story: A $\mathsf{RealIntervalTotalBoundedness}^\uparrow$ packet is generated from LocatedIntervalUp endpoints, a LocatedRealIntervalUp locatedness row, a positive RatUp or RealUp tolerance ledger, finite mesh cells, displayed centers, a FiniteEpsilonNetUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6169

A Concrete Naming Certificate for ClosedParallelDiamondObstruction$^{\uparrow}$

The ClosedParallelDiamondObstruction$^{\uparrow}$ packet realizes the MetaCIC confluence frontier of section 9386.3 as a finite obstruction surface. It sits beside the premise ledger of chapter 2325, the finite observation budget of chapter 5099, the residual-substitution boundary of chapter 2181, and the residual checker of chapter 3827. The packet records the missing closed parallel diamond route as a named obstruction row, separating it from both the existing premise ledger and any future proof of BetaParallelDiamondClosed.

Definition 6169.1 (Closed parallel diamond obstruction carrier)**.** *A* ClosedParallelDiamondObstruction$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (P, L, R, C, B, D, H, K, Q, N)$$

with the following displayed rows:

1. *P is the closed parallel-step window whose two branches are under inspection.*
2. *L is the* ClosedParallelDiamondPremiseLedger$^{\uparrow}$ *row, recording the premise-ledger route that a conditional confluence bridge may cite.*
3. *R is the* MetaCICResidualSubstitutionBoundary$^{\uparrow}$ *row, keeping residual-substitution compatibility visible as a separate obstruction.*
4. *C is the* MetaCICConfluenceResidualChecker$^{\uparrow}$ *row, recording the finite checker status available for the displayed residual window.*
5. *B is the* FiniteObservationConfluenceBudget$^{\uparrow}$ *row, bounding the observation budget in which local joins may be inspected.*
6. *D is the obstruction row:* BetaParallelDiamondClosed *is not discharged by the packet and remains a required closed-diamond input.*
7. *H records componentwise* hsame *transport for the closed parallel window, premise ledger, residual boundary, checker, finite budget, obstruction row, replay, provenance, and local naming rows.*
8. *K records displayed* Cont *replay from the closed parallel window through L, R, C, B, D, H, Q, N.*
9. *Q is the* Pkg *provenance over* ClosedParallelDiamondObstruction$^{\uparrow}$, ClosedParallelDiamondPremiseLedger$^{\uparrow}$, FiniteObservationConfluenceBudget$^{\uparrow}$, MetaCICResidualSubstitutionBoundary$^{\uparrow}$, MetaCICConfluenceResidualChecker BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{ClosedParallelDiamondObstruction}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $P, L, R, C, B, D, H, K, Q, N$. It has no coordinate proving BetaParallelDiamondClosed*, no proof of* BetaParallelSubstitutionCompatibility*, no arbitrary closed Church–Rosser theorem, no host residual semantics, and no unchecked confluence bridge.*

Theorem 6169.2 (Closed parallel diamond obstruction obligations)**.** *For every accepted* ClosedParallelDiamondObs *carrier $O = (P, L, R, C, B, D, H, K, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedParallelDiamondObstruction}^{\uparrow}}$ *surface consists exactly of closed parallel-window admission, premise-ledger exposure, residual-substitution boundary exposure, residual-checker exposure, finite observation-budget exposure, closed-diamond obstruction retention, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6169.1. The rows P, L, R, C, B, D are the mathematical obstruction surface, and H, K, Q, N are the structural rows.

The premise-ledger and residual rows are separate. Row L records the conditional premise route, while row R keeps substitution compatibility visible as an independent boundary.

The finite observation rows C and B record bounded checker status and local observation budget. They do not discharge the closed parallel diamond row D.

Rows H, K, Q, N transport, replay, source, and name the same finite packet. Since no row proves BetaParallelDiamondClosed or BetaParallelSubstitutionCompatibility, and no row exports arbitrary closed confluence, the listed obligations exhaust the local surface. □

Theorem 6169.3 (Closed parallel diamond obstruction separation)**.** *Every MetaCIC confluence read through an accepted* $\mathsf{ClosedParallelDiamondObstruction}^{\uparrow}$ *packet factors through*

$$P \longmapsto L, R, C, B \longmapsto D \longmapsto H, K, Q, N.$$

The packet therefore separates the missing closed parallel diamond premise from the existing premise-ledger carrier: it records why the confluence endpoint is blocked, not a proof that the endpoint has been supplied.

Proof. Begin with the closed parallel window P. The carrier allows that window to be read only together with the premise ledger L, residual boundary R, residual checker C, and finite observation budget B.

The obstruction row D is downstream of those rows and remains present in the accepted packet. It names the closed parallel diamond input as not discharged by the carrier.

Apply Definition 6169.2. Transport, replay, provenance, and local naming preserve the same finite route without replacing D by a theorem.

Thus the packet can guide a confluence consumer to the precise obstruction row, while the separate premise ledger remains a cited input surface rather than a closed diamond proof. □

Falsifiable prediction: If a MetaCIC confluence consumer cites $\mathsf{ClosedParallelDiamondObstruction}^{\uparrow}$ as proof of arbitrary closed-term confluence, then the packet is refuted because the obstruction row D is still present and no closed parallel diamond proof row is displayed.

Independence witness: closed_parallel_diamond_premise_ledger, finite_observation_confluence_budget, metacic_residual_substitution_boundary, metacic_confluence_residual_checker: ClosedParallelDiamondObstruct is not bijective to any listed sibling because it simultaneously displays the premise-ledger route, residual-substitution boundary, finite residual-checker status, observation budget, and retained closed-diamond obstruction row.

The Lean TasteGate site for ClosedParallelDiamondObstruction$^{\uparrow}$ must later provide explicit Nontrivial ClosedParallelDiamondObstruction$^{\uparrow}$ and FieldFaithful ClosedParallelDiamondObstruction$^{\uparrow}$ instances. Those instances should distinguish the premise-ledger row, finite observation budget, residual-substitution boundary, residual checker, obstruction row, transport, replay, provenance, and local naming as separate fields rather than collapsing the obstruction into the existing ledger.

Closure status for NameCert$_{\text{ClosedParallelDiamondObstruction}^{\uparrow}}$.

Constructive story: A ClosedParallelDiamondObstruction$^{\uparrow}$ packet is generated from closed parallel-step windows, ClosedParallelDiamondPremiseLedger rows, MetaCICResidualSubstitutionBoundary rows, MetaCICConfluenceResidualChecker rows, FiniteObservationConfluenceBudget rows, a retained closed-diamond obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{ClosedParallelDiamondObstruction}^{\uparrow}}$, seed)

Scope closed: Definition 6169.1, Definition 6169.2, and Definition 6169.3 seed the finite obstruction packet for the MetaCIC closed parallel diamond frontier.

Formal verification: FormalStatus(NameCert$_{\text{ClosedParallelDiamondObstruction}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not prove BetaParallelDiamondClosed, BetaParallelSubstitutionCompatibility, arbitrary closed Church–Rosser, host residual semantics, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, premise-ledger exposure, residual-substitution boundary exposure, residual-checker exposure, finite observation-budget exposure, obstruction retention, transport, replay, provenance, and TasteGate Nontrivial and FieldFaithful instances for BEDC.Derived.ClosedParallelDiamondObstructionUp.

Chapter 6170

A Concrete Naming Certificate for $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$

Following Bourbaki, Kelley, and Bishop–Bridges, the $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$ packet records the finite route by which the real uniform structure and Cauchy-filter completion data become visible to Real and Completion consumers. It sits after the real-uniformity surface of chapter 1624, the entourage interface of chapter 339, the complete-uniform-space boundary of chapter 1588, the Cauchy-filter packet of chapter 333, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a displayed completion interface; it is not a quotient of streams, a selected limit operator, a trichotomy principle, countable choice, or an ambient completeness theorem beyond the rows it carries.

Definition 6170.1 (Uniform-space real-completion carrier)**.** *A* $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (R, U, F, B, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealUniformStructure}^{\uparrow}$ *row exposing the real-name source together with the displayed uniform structure induced on real observations.*
2. *U is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row for that same real-uniform source, carrying only the finite entourage comparisons visible to a completion request.*
3. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row naming the Cauchy-filter request that the completion consumer may inspect.*
4. *B is a Cauchy-filter basis or completion-basis row refining F to the finite family of filter and entourage requests used by the packet.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which real-name observations are scheduled.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row converting the scheduled stream windows into regular rational approximation data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the real-uniform source, entourage row, Cauchy-filter row, finite basis, StreamName window, and RegSeqRat readback have been displayed.*

8. *H records componentwise* hsame *transport for the real-uniform source, entourage row, Cauchy-filter row, basis row, stream windows, rational readback, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a real-completion request through R, U, F, B, S, Q, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$, $\mathsf{RealUniformStructure}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceRealCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for quotient stream equality, a selected representative of a completion, countable choice, real trichotomy, or a host-level completeness theorem.

Lean4 ✓ `BEDC.Derived.UniformSpaceRealCompletionUp.TasteGate.UniformSpaceRealCompletionTasteGate single carrier alignment`

Theorem 6170.2 (Uniform-space real-completion NameCert obligations)**.** *For every accepted* $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$ *carrier $T = (R, U, F, B, S, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniformSpaceRealCompletio}}$ *surface consists exactly of carrier admission, Real uniform-structure admission, entourage-row admission, Cauchy-filter admission, finite basis admission, StreamName finite-window admission, RegSeqRat readback, Real seal admission, componentwise transport, replay, provenance, and local naming for the displayed rows $R, U, F, B, S, Q, E, H, C, P, N$.*

Proof. Project the accepted packet. Carrier admission reads the real-uniform source R, the entourage row U, the Cauchy-filter row F, the finite basis row B, the stream-window row S, the rational readback Q, and the Real seal E.

The completion-facing rows are ordered. A request first sees the real-induced uniformity and entourage data, then the Cauchy-filter and basis rows, and only then the finite StreamName and RegSeqRat readback that can be sealed as Real data.

Transport and replay do not widen the route. The row H transports the displayed components, C replays the same route, P records the sibling provenance, and N names the local certificate.

Since no row stores quotient stream equality, a selected completion representative, countable choice, trichotomy, or host completeness, no local NameCert consumer can recover those principles from the packet. □

Theorem 6170.3 (Uniform-space real-completion Cauchy handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$ *carrier receives Cauchy completion data only through the displayed route*

$$\mathsf{RealUniformStructure}^{\uparrow},\ \mathsf{UniformSpace}^{\uparrow},\ \mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The handoff supplies a finite real-uniform source, entourage comparisons, a Cauchy-filter request, a finite basis, scheduled stream windows, regular rational readback, and a Real seal, all transported and replayed through H, C, P, N. It supplies no quotient stream, selected limit, endpoint trichotomy, countable choice, or ambient completeness theorem.

Proof. Begin with the carrier of Definition 6170.1. The only completion-facing source rows are the real-uniform row R, the entourage row U, the Cauchy-filter row F, and the basis row B.

The finite observation route is downstream of those rows. StreamName windows S schedule the visible requests, RegSeqRat readback Q exposes rational approximation data, and the Real seal E is read only after those finite rows are displayed.

Finally apply the NameCert obligations of Definition 6170.2. They force transport, replay, provenance, and local naming through H, C, P, N, and the carrier has no coordinate from which a quotient stream, selected limit, trichotomy, choice principle, or independent completeness theorem could be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformSpaceRealCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformSpaceRealCompletion}^{\uparrow}$ packet is generated from a RealUniformStructure row, a UniformSpace entourage row, a CauchyFilter row, a finite completion-basis row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformSpaceRealCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6170.1, Definition 6170.2, and Definition 6170.3 seed the finite real-uniform completion route over Real uniformity, entourage, Cauchy-filter, StreamName, RegSeqRat, and Real seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformSpaceRealCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient streams, selected completion representatives, trichotomy, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, entourage admission, Cauchy-filter basis exactness, finite stream-window readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.UniformSpaceRealCompletionUp.

Chapter 6171

A Concrete Naming Certificate for $\mathsf{CechCompleteSpace}^{\uparrow}$

Following Cech, Engelking, and Kelley, the $\mathsf{CechCompleteSpace}^{\uparrow}$ packet records the topological completion-facing surface of a completely regular space through finite covers, uniform envelopes, and displayed completion seals. It sits after the topology carrier of chapter 88, the complete metric boundary of chapter 128, the uniform-complete route of chapter 2211, the Polish-space consumer of chapter 1490, and the finite observation windows of chapter 126. The packet is a NameCert reconstruction of the Cech-completion surface; it does not claim ambient choice, quotient completion equality, selected ultrafilters, or open-cover compactness.

Definition 6171.1 (Cech complete space carrier)**.** *A* $\mathsf{CechCompleteSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, U, M, F, W, S, R, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, carrying the open-set classifier and the completely regular observation surface used by the completion request.*
2. *U is the* $\mathsf{UniformCompleteSpace}^{\uparrow}$ *envelope row, recording the uniform cover and Cauchy-filter completion data available from the same topological source.*
3. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *comparison row, used only when a metrizable sibling read is explicitly present.*
4. *F is the finite Cech-filter row. It stores the nested cover, centered finite-intersection checks, and refusal boundary for non-convergent filter branches.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window row through which cover refinements, filter stages, and completion-facing observations are inspected.*
6. *S is the* $\mathsf{PolishSpace}^{\uparrow}$ *sibling handoff row, present only as a consumer-facing comparison surface and not as the definition of the packet.*
7. *R is the terminal regularization and real-boundary ledger for finite cover diameters, distance comparisons when M is present, and local completion thresholds.*
8. *H records componentwise* hsame *transport for the topology, uniform-complete, complete-metric, filter, stream-window, Polish comparison, regularization, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Cech-completion request through T, U, F, W, R and only then through M or S when those sibling consumers are used.*
10. *P is the* Pkg *provenance over* $\mathsf{CechCompleteSpace}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{UniformCompleteSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CechCompleteSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for ambient choice, quotient completion equality, selected ultrafilters, open-cover compactness, host equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CechCompleteSpaceUp.CechCompleteSpaceTasteGate single carrier alignment`

Theorem 6171.2 (Cech complete space NameCert obligations)**.** *For every accepted* $\mathsf{CechCompleteSpace}^{\uparrow}$ *carrier $Q = (T, U, M, F, W, S, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CechCompleteSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, topology-source admission, uniform-complete envelope admission, optional complete-metric comparison, finite Cech-filter admission, StreamName window visibility, optional Polish comparison, regularization-ledger admission, componentwise transport, replay, provenance, and local naming for the displayed rows $T, U, M, F, W, S, R, H, C, P, N$.*

Proof. Project the accepted carrier in Definition 6171.1. The projection exposes the topology source T, uniform-complete envelope U, complete-metric comparison row M, finite Cech-filter row F, StreamName windows W, Polish comparison row S, regularization ledger R, and structural rows H, C, P, N.

The completion-facing route starts from topology and uniform data. A request must read T and U before it can inspect the finite filter row F, and the stages of F are visible only through the finite windows W.

The sibling comparison rows do not define the packet. Row M supplies a complete-metric comparison only when such a comparison is displayed, and row S supplies a Polish-space consumer surface only after the Cech-filter route has already been read.

The structural rows preserve that same route. Since none of $T, U, M, F, W, S, R, H, C, P, N$ stores ambient choice, quotient completion equality, selected ultrafilters, open-cover compactness, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6171.3 (Cech complete space completion handoff)**.** *Every completion-facing, complete-metric-facing, uniform-complete-facing, or Polish-facing consumer of an accepted* $\mathsf{CechCompleteSpace}^{\uparrow}$ *carrier $Q = (T, U, M, F, W, S, R, H, C, P, N)$ reads the displayed route*

$$T, U \longmapsto F \longmapsto W \longmapsto R \longmapsto M, S \longmapsto H, C, P, N$$

before it receives the public packet. Thus the consumer sees the topological source, uniform completion envelope, finite Cech-filter evidence, StreamName observation windows, regularization ledger, optional complete-metric comparison, optional Polish comparison, transport, replay, provenance, and local naming without receiving ambient choice, quotient completion equality, selected ultrafilters, or open-cover compactness.

Proof. Start with a consumer of Q. Carrier admission gives the topology source and uniform-complete envelope as the displayed pair T, U, so the completion request cannot start from a hidden host object.

The Cech-filter evidence is then read through F and W. The finite filter row records the cover-refinement and centered finite-intersection checks, while the StreamName row exposes only finite windows of that route.

The regularization ledger R is downstream of those windows. Only after R has been read may the optional complete-metric comparison M or Polish comparison S be consumed.

Finally H, C, P, N transport, replay, source, and name the same displayed route. The excluded choice, quotient, ultrafilter, open-cover compactness, host-equality, and ambient-completeness data have no coordinate in the carrier. □

Falsifiable prediction: Every $\mathsf{CechCompleteSpace}^{\uparrow}$ completion-facing read exposes a $\mathsf{Topology}^{\uparrow}$ source row, $\mathsf{UniformCompleteSpace}^{\uparrow}$ envelope row, finite Cech-filter row, $\mathsf{StreamName}^{\uparrow}$ windows, regularization ledger, optional $\mathsf{CompleteMetric}^{\uparrow}$ comparison, optional $\mathsf{PolishSpace}^{\uparrow}$ comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: topology, complete-metric, uniform-complete-space, polishspace, streamname: $\mathsf{CechCompleteSpace}^{\uparrow}$ is not bijective to any listed sibling because it binds a topological source, uniform-complete envelope, finite Cech-filter evidence, StreamName windows, and completion-facing comparison rows in one finite carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{CechCompleteSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CechCompleteSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6171.1, Definition 6171.2, and Definition 6171.3 seed the finite Cech-completion route through $\mathsf{Topology}^{\uparrow}$, $\mathsf{UniformCompleteSpace}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, and $\mathsf{StreamName}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CechCompleteSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient choice, quotient completion equality, selected ultrafilters, open-cover compactness, host equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, topology-source admission, uniform-complete envelope admission, finite Cech-filter exactness, StreamName window visibility, comparison-row nonescape, completion handoff, and local naming for BEDC.Derived.CechCompleteSpaceUp.

Constructive story: A $\mathsf{CechCompleteSpace}^{\uparrow}$ packet is generated from a TopologyUp source, a UniformCompleteSpaceUp envelope, an optional CompleteMetricUp comparison, finite Cech-filter evidence, StreamNameUp windows, an optional PolishSpaceUp comparison, a regularization ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6172

A Concrete Naming Certificate for $\mathsf{LocatedCompactCompletion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{LocatedCompactCompletion}^{\uparrow}$ packet records the finite completion-facing route by which located compact metric data supplies Cauchy completion consumers with explicit finite-net and real-seal rows. It sits after the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the Cauchy-thread surface of chapter 128, the compact metric source of chapter 129, and the finite covering route of chapter 127. The certificate names only the located finite-net handoff into the RegSeqRat/Real completion route; it does not choose quotient representatives, host limits, or open-cover compactness data.

Definition 6172.1 (Located compact completion carrier)**.** *A* $\mathsf{LocatedCompactCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, T, D, S, Q, R, F, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the metric, total-boundedness, and complete-metric interfaces read by compact consumers.*
2. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row, indexed by a dyadic tolerance ledger from* $\mathsf{DyadicRatCore}^{\uparrow}$.
3. *D is the dyadic radius and rounding core that measures the requested net precision.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation-window row used to inspect candidate Cauchy threads.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row attached to the same window schedule.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the regular-rational readback.*
7. *F records the finite net centers and coverage witnesses carried from T into the completion route.*
8. *M is the complete-metric Cauchy handoff row, read before any completion consumer treats the finite-net route as convergent data.*
9. *H records componentwise* hsame *transport for all displayed compact, net, tolerance, stream, rational, real, handoff, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a located-compact completion request through $K, T, D, S, Q, R, F, M, H, P, N$*

11. *P is the* Pkg *provenance over* $\mathsf{LocatedCompactCompletion}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only by these displayed rows. It has no coordinate for arbitrary open covers, quotient completion equality, selected limits, host real equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedCompactCompletionUp.LocatedCompactCompletionTasteGate single carrier alignment`

Theorem 6172.2 (Located compact completion carrier obligations)**.** *Every accepted* $\mathsf{LocatedCompactCompletion}^{\uparrow}$ *carrier* $L = (K, T, D, S, Q, R, F, M, H, C, P, N)$ *exposes exactly the compact metric source, finite total-boundedness net, dyadic tolerance ledger, stream-window surface, regular-rational readback, Real seal, finite-net handoff, complete-metric handoff, componentwise transport, displayed replay, provenance, and local naming rows listed in Definition 6172.1. No consumer of the local* $\mathsf{NameCert}_{\mathsf{LocatedCompactCompletion}^{\uparrow}}$ *row receives an additional open-cover compactness theorem, quotient completion object, selected limit, host real equality, or Real completeness principle.*

Proof. Project the carrier L. The compact source is K, while the finite-net data is the pair of rows T, F measured by the dyadic tolerance row D.

The completion-facing readback is not taken directly from compactness. It passes through the window row S, the regular-rational row Q, the Real seal R, and the complete-metric handoff M before a downstream consumer may read the route as completion data.

Transport and replay preserve the displayed packet only. The rows H, C, P, N keep the same finite BHist data under hsame, Cont, Pkg, and NameCert bookkeeping, so the excluded open-cover, quotient, selected-limit, host-equality, choice, and completeness objects have no row from which they could be recovered. □

Theorem 6172.3 (Located compact completion finite-net seal)**.** *Every* $\mathsf{LocatedCompactCompletion}^{\uparrow}$ *consumer that asks for a completion-facing compactness row must first display the finite-net seal*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}$$

together with the $\mathsf{TotallyBounded}^{\uparrow}$ *net row and the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row. The seal exports a finite list of inspected centers, tolerance-indexed coverage, regular-rational readback, and a terminal Real seal; it exports no open-cover subcover selector, quotient completion equality, selected limit, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier rows in order. The finite-net seal begins with T and D, so every radius request is measured by dyadic BEDC data before any stream or real row is inspected.

Next the route enters S, Q, R. These rows force the completion-facing consumer to pass through StreamName windows, RegSeqRat regular readback, and the Real seal rather than through a host completion or quotient representative.

Finally F and M bind the finite centers to the complete-metric handoff. Since replay and provenance only repackage those rows, the consumer receives the listed seal and none of the excluded compactness, quotient, selected-limit, host-equality, or completeness principles. □

Theorem 6172.4 (Located compact completion Bishop handoff)**.** *The Bishop-style handoff from compact finite-net data to completion consumers factors through chapter 301 and chapter 38 after the chapter 126 and chapter 291 rows are displayed. Thus a downstream completion packet may cite* $\mathsf{LocatedCompactCompletion}^{\uparrow}$ *only as a finite NameCert route over* BHist*,* hsame*,* Cont*,* Pkg*, finite nets, and the RegSeqRat/Real seal rows, not as a host theorem identifying a quotient completion with a selected real limit.*

Proof. Start from the compact metric and total-boundedness rows of Definition 6172.1. They supply a located compact source and finite centers, but not a host compactness theorem.

The handoff to completion consumers is then routed through dyadic tolerances, stream observations, and regular rational readback. This is exactly the finite route named by the StreamName and RegSeqRat sibling chapters before the terminal Real seal is read.

The complete-metric handoff row records where Cauchy completion consumers may attach their own local obligations. It does not identify quotient representatives, pick limits, or equate the BEDC real seal with a host real object, so the Bishop-style completion use remains a displayed NameCert route. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedCompactCompletion}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedCompactCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6172.1, Definition 6172.2, Definition 6172.3, and Definition 6172.4 seed the located compact finite-net handoff into the RegSeqRat and Real completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedCompactCompletion}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.LocatedCompactCompletionUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, quotient completion equality, selected limits, host real equality, countable choice, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net exactness, dyadic tolerance routing, StreamName and RegSeqRat handoff, Real seal non-escape, CompleteMetric handoff, transport, replay, provenance, and local naming for BEDC.Derived.LocatedCompactCompletionUp.

Constructive story: A $\mathsf{LocatedCompactCompletion}^{\uparrow}$ packet is generated from a CompactMetricUp source, a TotallyBoundedUp finite-net row, a DyadicRatCoreUp tolerance ledger, StreamNameUp observation windows, RegSeqRatUp readback, a RealUp seal, a CompleteMetricUp handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6173

A Concrete Naming Certificate for UniformLipschitzCompactVision$^{\uparrow}$

Following Bishop and Bridges on compact metric uniform continuity, and the Arzela–Ascoli habit of making one shared family bound visible before compactness is consumed, the UniformLipschitzCompactVision$^{\uparrow}$ packet records the finite NameCert data behind the Heine–Cantor finite-net route. It sits after the compact metric source of chapter 129, the shared-bound family packet of chapter 6167, the compact-net selector of chapter 914, the finite Lebesgue-number handoff of chapter 1301, and the downstream UniformModulus$^{\uparrow}$ consumer. The packet names the compact-uniform-continuity vision as finite shared-Lipschitz data; it does not import open-cover compactness, selected subcovers, host function-space compactness, or a modulus chosen outside the displayed rows.

Definition 6173.1 (Uniform Lipschitz compact-vision carrier)**.** *A* UniformLipschitzCompactVision$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (K, F, L, U, B, S, M, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing finite metric-net observations and compact-metric provenance for the same source.*
2. *F is the finite* ContinuousMap$^{\uparrow}$ *family row whose members are the displayed maps inspected by the consumer.*
3. *L is the shared* LipschitzConstant$^{\uparrow}$ *row, recording one family-level bound for every listed member.*
4. *U is the* UniformLipschitzFamily$^{\uparrow}$ *handoff row tying K, F, L to the existing shared-Lipschitz family certificate.*
5. *B is the* FiniteLebesgueNumber$^{\uparrow}$ *row recording the positive finite radius ledger used by the compact-net route.*
6. *S is the* CompactNetModulusSelector$^{\uparrow}$ *row selecting the finite net, center-radius witnesses, and lower precision data from K, F, L, B.*
7. *M is the* UniformModulus$^{\uparrow}$ *export row consumed by compact-uniform-continuity clients.*
8. *H records componentwise* hsame *transport for the compact source, map family, shared Lipschitz row, family handoff, Lebesgue-number row, compact-net selector, uniform-modulus export, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a compact-uniform tolerance request through $K, F, L, U, B, S, M, H, P, N$.*
10. *P is the* Pkg *provenance over* UniformLipschitzCompactVision$^{\uparrow}$, CompactMetric$^{\uparrow}$, ContinuousMap$^{\uparrow}$, LipschitzConstant$^{\uparrow}$, UniformLipschitzFamily$^{\uparrow}$, FiniteLebesgueNumber$^{\uparrow}$, CompactNetModulusSelector$^{\uparrow}$, UniformModulus$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{UniformLipschitzCompactVision}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected subcovers, arbitrary cover refinements, ambient function-space topology, quotient equality of maps, choice-selected centers, or an undisplayed modulus.

Theorem 6173.2 (Uniform Lipschitz compact-vision NameCert obligations)**.** *For every accepted* UniformLipschitzCompactVision$^{\uparrow}$ *carrier $V = (K, F, L, U, B, S, M, H, C, P, N)$, the local* NameCert$_{\text{UniformLipschitzCompa}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for $K, F, L, U, B, S, M, H, C, P, N$;*
2. *compact-source exposure, requiring every finite-net and distance read to pass through the displayed* CompactMetric$^{\uparrow}$ *row K;*
3. *continuous-family exposure, requiring the inspected maps to be read through the finite* ContinuousMap$^{\uparrow}$ *family row F;*
4. *shared Lipschitz exactness, requiring the single* LipschitzConstant$^{\uparrow}$ *row L to control every listed member before the* UniformLipschitzFamily$^{\uparrow}$ *handoff U is available;*
5. *finite-radius accountability, requiring the* FiniteLebesgueNumber$^{\uparrow}$ *row B and* CompactNetModulusSelector$^{\uparrow}$ *row S to expose the same finite compact-net route;*
6. *uniform-modulus export, requiring M to consume only the shared Lipschitz row, finite radius ledger, compact-net selector, and displayed structural rows;*
7. *structural stability, requiring H, C, P, N to transport, replay, source, and name only the rows already listed.*

No compact-uniform-continuity obligation for this packet is stored outside those rows.

Lean4 ✓ `BEDC.Derived.UniformLipschitzCompactVisionUp.UniformLipschitzCompactVisionTasteGate`
`single carrier alignment`

Proof. Project the accepted carrier $V = (K, F, L, U, B, S, M, H, C, P, N)$. Its mathematical rows are the compact source K, map family F, shared Lipschitz row L, family handoff U, finite radius row B, compact-net selector S, and uniform-modulus export M.

The compact source and map family are read first. The row K supplies the finite metric windows, while F restricts the consumer to displayed continuous-map members.

The shared bound is then forced through L and U. Because one LipschitzConstant$^{\uparrow}$ row controls every listed member, the packet records family-level Lipschitz evidence rather than a hidden memberwise selector.

The finite-net route is recorded by B and S. These rows supply the positive finite-radius ledger and selected compact-net precision consumed by M.

The structural rows H, C, P, N preserve this same route. Since the excluded open-cover, subcover-selection, function-space, quotient, choice, and hidden-modulus objects do not occur as coordinates, the displayed obligations exhaust the local certificate. □

Theorem 6173.3 (Uniform Lipschitz compact-vision modulus handoff)**.** *Every compact-uniform-continuity consumer of an accepted* $\mathsf{UniformLipschitzCompactVision}^{\uparrow}$ *carrier factors through the shared Lipschitz row and the finite net modulus route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{LipschitzConstant}^{\uparrow} \longmapsto \mathsf{UniformLipschitzFamily}^{\uparrow} \longmapsto \mathsf{FiniteLebesgueNumber}^{\uparrow},\ \mathsf{Co}$$

Thus the consumer receives a uniform-modulus export from the displayed compact source, displayed family, one shared Lipschitz bound, finite radius ledger, and compact-net selector; it receives no open-cover compactness theorem, selected subcover, host compactness assertion, ambient family topology, or modulus selected outside the packet.

Proof. Start with a tolerance request. The compact source row K and map-family row F are the only rows that expose source metric windows and inspected maps.

Read the shared Lipschitz data through L and U. This step binds every listed map to one displayed family-level bound before any compact-net selector may be consumed.

Next read the finite radius route. The row B supplies the finite Lebesgue-number ledger, and S turns the compact-net observations into the selected precision data used by the downstream modulus.

The export row M is downstream of those rows. Therefore every accepted $\mathsf{UniformModulus}^{\uparrow}$ read is a finite handoff through $K, F, L, U, B, S, H, C, P, N$.

Because no open-cover compactness object, selected subcover, host compactness assertion, ambient family topology, quotient map equality, or hidden modulus occurs in the carrier, no compact-uniform-continuity consumer can factor through such data. □

Theorem 6173.4 (Compact-uniform equicontinuous family selector)**.** *An accepted* $\mathsf{UniformLipschitzCompactVision}^{\uparrow}$ *packet exports a* $\mathsf{UniformModulus}^{\uparrow}$ *row for an equicontinuous family only after the* $\mathsf{CompactMetric}^{\uparrow}$ *source,* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, shared* $\mathsf{LipschitzConstant}^{\uparrow}$ *row,* $\mathsf{UniformLipschitzFamily}^{\uparrow}$ *handoff,* $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ *row,* $\mathsf{CompactNetModulusSelector}^{\uparrow}$ *row,* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *rows are all displayed. The selector supplies no selected subcover, host function-space compactness, or modulus chosen outside those displayed rows.*

Proof. Project the carrier rows $V = (K, F, L, U, B, S, M, H, C, P, N)$. The rows K and F expose the compact metric source and continuous-map family before any family-level modulus can be named.

Invoke the shared-Lipschitz route through L and U. This binds the whole displayed family to one $\mathsf{LipschitzConstant}^{\uparrow}$ row and makes the equicontinuity read a family handoff rather than a memberwise hidden choice.

Apply the finite compact-net selector. The $\mathsf{FiniteLebesgueNumber}^{\uparrow}$ row B supplies the finite radius ledger, and S turns the compact-net observations into the precision data consumed by the $\mathsf{UniformModulus}^{\uparrow}$ row M.

Finally transport and replay the same rows through H, C, P, N. Since selected subcovers, host function-space compactness, and externally chosen moduli are absent from the carrier, the exported family selector is exactly the displayed compact-net route. □

Falsifiable prediction: An accepted $\mathsf{UniformLipschitzCompactVision}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer obtains a UniformModulusUp export without displaying the CompactMetric source row, ContinuousMap family row, shared LipschitzConstant row, UniformLipschitzFamily handoff, FiniteLebesgueNumber row, CompactNetModulusSelector row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, lipschitz-constant, uniform-lipschitz-family, finite-lebesgue-number, compactnetmodulusselector, and uniformmodulus: none is bijective to $\mathsf{UniformLipschitzCompactVision}^{\uparrow}$ because only this packet jointly records the shared-Lipschitz family route, finite radius ledger, compact-net selector, and UniformModulus export as one compact-uniform-continuity NameCert surface.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformLipschitzCompactVision}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformLipschitzCompactVision}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6173.1, Definition 6173.2, and Definition 6173.3 seed the finite shared-Lipschitz compact-uniform route from CompactMetric data through finite net modulus selection to UniformModulus export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformLipschitzCompactVision}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, host compactness, ambient function-space topology, quotient map equality, choice-selected centers, hidden moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, continuous-family exposure, shared Lipschitz exactness, finite-radius accountability, compact-net selector compatibility, uniform-modulus export, transport, replay, provenance, and local naming for BEDC.Derived.UniformLipschitzCompactVisionUp.

Constructive story: A $\mathsf{UniformLipschitzCompactVision}^{\uparrow}$ packet is generated from a CompactMetricUp source row, ContinuousMapUp family row, shared LipschitzConstantUp row, UniformLipschitzFamilyUp handoff, FiniteLebesgueNumberUp row, CompactNetModulusSelectorUp row, UniformModulusUp export, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6174

A Concrete Naming Certificate for $\mathsf{FiniteLipschitzFamily}^{\uparrow}$

Following Arzela-Ascoli and Bishop-style compact metric analysis, the $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ packet records the finite BEDC surface for a listed family of continuous maps controlled by one displayed Lipschitz row. It sits after the compact source of chapter 129, the target metric of chapter 57, the displayed map rows of chapter 130, the shared bound ledger of chapter 2328, and the tolerance handoff of chapter 369. The packet gives compact-uniform and Arzela–Ascoli consumers a finite family row; it does not assert ambient function-space compactness, selected subsequences, open-cover compactness, quotient function equality, or a hidden modulus.

Definition 6174.1 (Finite Lipschitz-family carrier)**.** *A* $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, I, M, L, V, H, C, P, N)$$

with the following displayed rows:

1. X *is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row whose finite metric windows supply all source-distance reads.*
2. Y *is the target* $\mathsf{Met}^{\uparrow}$ *row whose distance reads are used by the displayed map values.*
3. I *is the finite index row naming exactly the family members available for inspection.*
4. M *is the finite list of* $\mathsf{ContinuousMap}^{\uparrow}$ *rows, one row for each index in* I*.*
5. L *is the shared* $\mathsf{LipschitzConstant}^{\uparrow}$ *row, recording one displayed bound used by every listed member.*
6. V *is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that receives the compact source, target metric, displayed maps, and common Lipschitz bound as tolerance data.*
7. H *records componentwise* hsame *transport for* X, Y, I, M, L, V *and the structural rows.*
8. C *records displayed* Cont *replay of the route from compact metric source data through the listed continuous maps and the shared Lipschitz row to the uniform-modulus handoff.*
9. P *is the* Pkg *provenance over* $\mathsf{FiniteLipschitzFamily}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{LipschitzConstant}^{\uparrow}$*,* $\mathsf{UniformModulus}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
10. N *is the local* $\mathsf{NameCert}_{\mathsf{FiniteLipschitzFamily}^{\uparrow}}$ *row.*

The carrier has no coordinate for an ambient function-space topology, selected subsequence, open-cover compactness proof, quotient equality of functions, or non-displayed modulus.

Lean4 ✓ `BEDC.Derived.FiniteLipschitzFamilyUp.TasteGate.FiniteLipschitzFamilyTasteGate single carrier alignment`

Theorem 6174.2 (Finite Lipschitz-family NameCert obligations)**.** *For every accepted* $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ *carrier* $F = (X, Y, I, M, L, V, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteLipschitzFamily}^{\uparrow}}$ *surface consists exactly of compact-source exposure, target-metric exposure, finite-index admission, displayed continuous-map admission, shared-bound exactness,* $\mathsf{UniformModulus}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored in an ambient function-space object or in an unlisted modulus selector.*

Proof. Project the accepted packet $F = (X, Y, I, M, L, V, H, C, P, N)$. The rows X, Y, I, M, L, V are the only mathematical coordinates available to the family read.

The finite family part is therefore explicit. The index row I lists the inspectable members, M attaches each member to a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row, and L supplies one common $\mathsf{LipschitzConstant}^{\uparrow}$ row for all of them.

The handoff part is also explicit. The row V receives only the displayed compact source, target metric, map, and bound data, while H, C, P, N transport, replay, source, and name the same packet.

Because the excluded objects do not occur among the carrier coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 6174.3 (Finite Lipschitz-family modulus handoff)**.** *Every compact-uniform or Arzela–Ascoli consumer of an accepted* $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ *carrier receives the route*

$$\mathsf{CompactMetric}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow}, \mathsf{LipschitzConstant}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow} \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

The common Lipschitz row is exported only as a finite-family tolerance handoff. It exports no ambient function-space compactness, selected subsequence, open-cover compactness, quotient function equality, or hidden modulus.

Proof. Begin with the rows X and Y. They provide the source and target metric reads that any later compact-uniform consumer may inspect.

Next read the family rows I and M together with the shared bound row L. Since every listed member is paired with a displayed $\mathsf{ContinuousMap}^{\uparrow}$ row and the same $\mathsf{LipschitzConstant}^{\uparrow}$ row, the consumer never ranges over a host function space.

Finally export V, H, C, P, N. The $\mathsf{UniformModulus}^{\uparrow}$ handoff receives the compact metric, target metric, listed maps, and common bound, while the structural rows preserve exactly that route for downstream replay.

The absent coordinates rule out selected subsequences, open-cover compactness, quotient function equality, and hidden moduli, so the handoff is precisely the finite BEDC family packet. □

Falsifiable prediction: An accepted $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ read is refuted if a compact-uniform or Arzela–Ascoli consumer obtains a $\mathsf{UniformModulus}^{\uparrow}$ handoff without displaying the $\mathsf{CompactMetric}^{\uparrow}$ source row, target $\mathsf{Met}^{\uparrow}$ row, finite index row, $\mathsf{ContinuousMap}^{\uparrow}$ rows, shared $\mathsf{LipschitzConstant}^{\uparrow}$ row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, metric, continuousmap, lipschitz-constant, uniform-modulus, uniform-lipschitz-family: no listed sibling is bijective with $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ because

the packet jointly carries finite family indexing, displayed continuous-map rows, one shared Lipschitz bound, a tolerance-only $\mathsf{UniformModulus}^{\uparrow}$ handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteLipschitzFamily}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteLipschitzFamily}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6174.1, Definition 6174.2, and Definition 6174.3 seed the finite shared-Lipschitz family route from compact metric source data through listed continuous-map rows to a UniformModulus handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteLipschitzFamily}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient function-space compactness, selected subsequences, open-cover compactness, quotient function equality, non-displayed moduli, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, finite-index admission, continuous-map exposure, shared-bound exactness, UniformModulus handoff, transport, replay, provenance, and local naming for BEDC.Derived.FiniteLipschitzFamilyUp.

Constructive story: A $\mathsf{FiniteLipschitzFamily}^{\uparrow}$ packet is generated from a compact metric source row, target Metric row, finite family-index row, ContinuousMap rows, a shared LipschitzConstant row, UniformModulus handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6175

A Concrete Naming Certificate for $\mathsf{TypedGalleryContextProbe}^{\uparrow}$

The $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ packet realizes the typed-example context surface of chapter 9382 as finite NameCert data. It reads the gallery's sort-indexed contexts, Pi forms, applications, beta redexes, and Church examples as displayed rows rather than as an ambient test suite. The packet is a context probe for decidability and subject-reduction audit consumers; it does not prove global typechecking completeness, subject reduction, confluence, normalization, or conversion decidability.

The checker-read row K is consumed through chapter 749 before any decidability-facing reader may inspect bounded comparison or refusal data.

Definition 6175.1 (Typed gallery context-probe carrier)**.** *A* $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, B, A, R, C, K, D, Q, H, T, P, N)$$

with the following displayed rows:

1. *S is the sort-indexed context row, listing the contexts used by the gallery examples.*
2. *B is the binder-shape row for Pi forms, shifted variables, dependent identity examples, and polymorphic identity examples.*
3. *A is the application row for ordinary applications, application tails, and projected applications.*
4. *R is the beta-redex row for the displayed reductions used by the gallery.*
5. *C is the Church-example row for booleans, natural numbers, pairs, options, lists, sums, and equality examples.*
6. *K is the checker-read row consumed by the MetaCIC decidability boundary and bounded typechecking budget surfaces.*
7. *D is the discharge-facing row consumed by the substitution spine and subject-reduction boundary surfaces.*
8. *Q records the boundary refusal: the probe supplies typed context data but does not supply a global preservation theorem.*
9. *H records componentwise* hsame *transport for gallery rows and structural rows.*
10. *T records displayed* Cont *replay from the typed gallery audit map through $S, B, A, R, C, K, D, Q, H, P, N$.*

11. *P is the* Pkg *provenance over* $\mathsf{TypedGalleryContextProbe}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows together with the MetaCIC typed-gallery audit surface.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{TypedGalleryContextProbe}^{\uparrow}}$ *row.*

The classifier compares accepted probes only through $S, B, A, R, C, K, D, Q, H, T, P, N$. It has no coordinate for unrestricted typechecking completeness, unconditional subject reduction, global substitution soundness, confluence, normalization, conversion decidability, or a hidden checker oracle.

Theorem 6175.2 (Typed gallery context-probe NameCert obligations)**.** *For every accepted* TypedGalleryContextPr *carrier $G = (S, B, A, R, C, K, D, Q, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TypedGalleryContextProbe}^{\uparrow}}$ *surface consists exactly of sort-context admission, binder-shape admission, application admission, beta-redex admission, Church-example admission, checker-read exposure, discharge-facing exposure, boundary refusal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The proof-search and typechecking consumers receive a finite gallery probe, not an unrestricted metatheorem.*

Proof. Project the accepted packet $G = (S, B, A, R, C, K, D, Q, H, T, P, N)$. The rows S, B, A, R, C are the gallery-facing coordinates, so they give the only admitted context, binder, application, beta, and Church-example data.

The consumer-facing rows are K, D, Q. The checker-read row K is the route to decidability and bounded typechecking budget consumers, while D is the route to substitution-spine and subject-reduction boundary consumers. The refusal row Q records that these routes do not close the global metatheorems.

Finally H, T, P, N transport, replay, source, and name the same finite probe. Since no hidden checker oracle or unrestricted theorem coordinate is present, the listed rows exhaust the NameCert obligations. □

Theorem 6175.3 (Typed gallery context-probe binder handoff)**.** *Every decidability, substitution-spine, or subject-reduction audit consumer of an accepted* $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ *carrier receives binder data only through the route*

$$S, B, A, R, C \longmapsto K, D, Q \longmapsto \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

In particular, sort-indexed contexts and Pi/application examples are exposed before the checker-read and discharge-facing rows are consumed.

Proof. Start with S and B. These rows display the sort-indexed contexts and binder shapes before any checker or discharge request is available.

Read A, R, C next. Application forms, beta redexes, and Church examples become probe data only as listed gallery rows, so the consumer cannot appeal to an ambient collection of examples.

The handoff then passes through K, D, Q. The checker-facing row and discharge-facing row are both guarded by the refusal row, which prevents the probe from being read as a proof of global subject reduction or global decidability.

Finally H, T, P, N preserve the same route for replay and naming. Thus binder-sensitive audit consumers receive finite context probes with explicit boundaries. □

Falsifiable prediction: A $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ read is refuted if a typed-gallery, metacic-decidable-boundary, bounded-typechecking-budget, metacic-substitution-spine, or metacic-subject-reduction-boundary consumer obtains checker or discharge data without first displaying sort-context, binder-shape, application, beta-redex, Church-example, refusal, transport, replay, provenance, and local naming rows.

Independence witness: typed-gallery, metacic-decidable-boundary, bounded-typechecking-budget, metacic-substitution-spine, metacic-subject-reduction-boundary: none is bijective with $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ because this packet jointly carries sort contexts, binder shapes, application forms, beta redexes, Church examples, checker-read data, discharge-facing data, refusal, transport, replay, provenance, and local NameCert.

The Lean TasteGate work for $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ must provide $\mathsf{Nontrivial\ TypedGalleryContextProbe}^{\uparrow}$ and $\mathsf{FieldFaithful\ TypedGalleryContextProbe}^{\uparrow}$ instances preserving $S, B, A, R, C, K, D, Q, H, T, P, N$ as distinguishable rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TypedGalleryContextProbe}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TypedGalleryContextProbe}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6175.1, Definition 6175.2, and Definition 6175.3 seed the finite typed-gallery context probe from sort contexts and binder rows through checker and discharge-facing boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TypedGalleryContextProbe}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global typechecking completeness, subject reduction, confluence, normalization, conversion decidability, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, sort-context exposure, binder-shape exposure, application and beta-redex exposure, Church-example exposure, checker-read routing, discharge-facing routing, boundary refusal, Nontrivial and FieldFaithful TasteGate instances, transport, replay, provenance, and local naming for BEDC.Derived.TypedGalleryContextProbeUp.

Constructive story: A $\mathsf{TypedGalleryContextProbe}^{\uparrow}$ packet is generated from sort-indexed context rows, binder-shape rows, application rows, beta-redex rows, Church-example rows, checker-read rows, discharge-facing rows, boundary refusal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6176

A Concrete Naming Certificate for $\mathsf{GaleShapleyStableMatching}^{\uparrow}$

Following Gale and Shapley on stable marriage and deferred acceptance, the $\mathsf{GaleShapleyStableMatching}^{\uparrow}$ packet records the finite NameCert data behind a stable matching construction. It sits after the graph matching surface of chapter 234, the graph carrier of chapter 118, the finite-set enumeration machinery of chapter 112, the preference order rows of chapter 54, and the proposal-list discipline of chapter 36. The packet names the stable-matching construction as finite deferred-acceptance data; it does not import host marriage-market equality, selected optimality outside the ledger, or an unrecorded global search.

Definition 6176.1 (Gale-Shapley stable-matching carrier)**.** *A* $\mathsf{GaleShapleyStableMatching}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (U, V, P_U, P_V, L, T, R, B, M, H, C, Q, N)$$

with the following displayed rows:

1. *U and V are finite* $\mathsf{FinSet}^{\uparrow}$ *rows for the two sides of the market.*
2. *P_U and P_V are* $\mathsf{TotalOrder}^{\uparrow}$ *preference-profile rows over the finite candidate lists named by U and V.*
3. *L is a* $\mathsf{List}^{\uparrow}$ *proposal ledger recording the ordered proposal events of the deferred-acceptance run.*
4. *T is the tentative-matching ledger, recording at each proposal stage the currently held pairs as a* $\mathsf{Matching}^{\uparrow}$ *edge predicate in the bipartite* $\mathsf{Graph}^{\uparrow}$ *carrier.*
5. *R is the rejection-history ledger, recording every rejected proposal together with the displayed preference comparison that caused it.*
6. *B is the blocking-pair refusal row: every displayed unmatched pair is refused by the proposer-exhaustion row, the held-partner preference row, or the rejection-history row.*
7. *M is the terminal* $\mathsf{Matching}^{\uparrow}$ *export row read by stable-matching consumers.*
8. *H records componentwise* hsame *transport for the two finite sides, the two preference profiles, proposal ledger, tentative ledger, rejection history, blocking-pair refusal row, matching export, replay, provenance, and local naming.*
9. *C records displayed* Cont *replay from a stable-matching query through $U, V, P_U, P_V, L, T, R, B, M, H, Q, N$.*

10. *Q is the* Pkg *provenance over* $\mathsf{GaleShapleyStableMatching}^{\uparrow}$, $\mathsf{Matching}^{\uparrow}$, $\mathsf{Graph}^{\uparrow}$, $\mathsf{FinSet}^{\uparrow}$, $\mathsf{TotalOrder}^{\uparrow}$, $\mathsf{List}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{GaleShapleyStableMatching}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows. It has no coordinate for host tuple matching, undisplayed preference functions, unbounded search, selected optimality outside the proposal ledger, quotient equality of matchings, or a stability proof detached from the blocking-pair refusal row.

Lean4 ✓ `BEDC.Derived.GaleShapleyStableMatchingUp.TasteGate.GaleShapleyStableMatchingTasteGate single carrier alignment`

Theorem 6176.2 (Gale-Shapley stable-matching NameCert obligations)**.** *For every accepted* GaleShapleyStableMa *carrier $G = (U, V, P_U, P_V, L, T, R, B, M, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{GaleShapleyStableMatching}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for $U, V, P_U, P_V, L, T, R, B, M, H, C, Q, N$;*
2. *finite-market exposure, requiring both sides to be read through U and V before any preference or matching row is consumed;*
3. *preference-profile exactness, requiring P_U and P_V to supply the displayed total-order comparisons used by proposal, holding, and rejection rows;*
4. *proposal-ledger accountability, requiring every proposal event consumed by the construction to occur in L with its current finite prefix;*
5. *tentative-matching compatibility, requiring every stage row in T to be a* $\mathsf{Matching}^{\uparrow}$ *edge predicate over the displayed bipartite* $\mathsf{Graph}^{\uparrow}$ *carrier;*
6. *rejection-history exactness, requiring R to pair each rejection with the preference comparison that justifies keeping or replacing a tentative partner;*
7. *blocking-pair refusal, requiring B to expose one displayed refusal reason for every finite candidate pair not exported by M;*
8. *structural stability, requiring H, C, Q, N to transport, replay, source, and name only the rows already listed.*

No stable-matching obligation for this packet is stored outside those rows.

Proof. Project the accepted carrier $G = (U, V, P_U, P_V, L, T, R, B, M, H, C, Q, N)$. The finite-market rows U and V are read first, so every participant and every candidate edge is already inside the displayed $\mathsf{FinSet}^{\uparrow}$ and $\mathsf{Graph}^{\uparrow}$ surfaces.

The preference rows P_U and P_V supply the only comparisons used by the construction. Thus proposal ordering, tentative holding, and rejection justifications cannot consult an undisplayed preference function.

Next read the dynamic rows. The proposal ledger L records the finite event order, T records the tentative $\mathsf{Matching}^{\uparrow}$ predicate at each stage, and R records the rejected proposal together with its preference comparison. These rows are enough to reconstruct the deferred-acceptance trace as finite BHist data.

Finally inspect B, M, H, C, Q, N. The row B refuses blocking pairs from the displayed trace, M exports the terminal matching row, and the structural rows preserve the same finite route. Since no hidden matching, preference, search, or quotient coordinate occurs in the carrier, the listed obligations exhaust the local certificate. □

Theorem 6176.3 (Gale-Shapley deferred-acceptance route)**.** *Every stable-matching consumer of an accepted* $\mathsf{GaleShapleyStableMatching}^{\uparrow}$ *carrier factors through the finite deferred-acceptance route*

$$\mathsf{FinSet}^{\uparrow},\ \mathsf{TotalOrder}^{\uparrow},\ \mathsf{List}^{\uparrow} \longmapsto \mathsf{Graph}^{\uparrow},\ \mathsf{Matching}^{\uparrow} \longmapsto \mathsf{GaleShapleyStableMatching}^{\uparrow}.$$

The consumer receives a terminal matching together with proposal, tentative-matching, rejection, and blocking-pair refusal rows. It receives no host marriage-market object, unrecorded search trace, external optimality selector, quotient matching representative, or stability assertion outside the displayed packet.

Proof. Start with a consumer read of the terminal matching row M. By the obligation theorem, the read is not available until the finite sides U, V and preference profiles P_U, P_V have been exposed.

The proposal sequence is then forced through L. Each proposal event has a finite prefix, and every tentative change reads its matching predicate through T in the displayed $\mathsf{Graph}^{\uparrow}$ and $\mathsf{Matching}^{\uparrow}$ surfaces.

The rejection ledger R and blocking-pair refusal row B supply the stability-facing part of the route. A pair not present in M is rejected because one side exhausted the proposal row, a held partner is preferred by a displayed total-order comparison, or the rejection history records the relevant event.

The structural rows H, C, Q, N keep this route componentwise. Therefore any stable-matching consumer factors through the finite deferred-acceptance packet and cannot obtain any excluded host-level matching, search, optimality, quotient, or detached-stability datum. □

Closure status for $\mathsf{NameCert}_{\mathsf{GaleShapleyStableMatching}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaleShapleyStableMatching}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6176.1, Definition 6176.2, and Definition 6176.3 seed the finite deferred-acceptance NameCert route from finite preference profiles to a terminal MatchingUp export.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaleShapleyStableMatching}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host marriage-market equality, global optimality outside the displayed ledger, quotient matching representatives, unrestricted search, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-market exposure, preference-profile exactness, proposal-ledger accountability, tentative-matching compatibility, rejection-history exactness, blocking-pair refusal, transport, replay, provenance, and local naming for BEDC.Derived.GaleShapleyStableMatchingUp.

Constructive story: A $\mathsf{GaleShapleyStableMatching}^{\uparrow}$ packet is generated from two FinSetUp side rows, two TotalOrderUp preference-profile rows, a ListUp proposal ledger, a MatchingUp tentative ledger over a GraphUp carrier, rejection history, blocking-pair refusal, a terminal matching export, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6177

A Concrete Naming Certificate for $\mathsf{ClosedAppArgumentStability}^{\uparrow}$

The $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ packet realizes the independent-codomain branch of the closed MetaCIC application-argument frontier as finite NameCert data. It sits beside the subject-reduction discharge socket of chapter 719, the discharge ledger of chapter 720, the application-argument obstruction of chapter 9377, the closed lambda-domain packet of chapter 3381, and the closed Pi-domain packet of chapter 3352. The packet names the checked independent-codomain route; it does not turn the general dependent-codomain AppArgTypeStable obligation into an unconditional theorem.

Definition 6177.1 (Closed application-argument stability carrier)**.** *A* $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (F, X, X', D, C, Y, Z, S, O, H, R, Q, N)$$

with the following displayed rows:

1. *F is the closed function row, naming a term typed by a displayed Pi type $\Pi D.\, C$.*
2. *X is the original argument row, typed by the displayed domain D.*
3. *X' is the reduced argument row, typed by the same displayed domain D.*
4. *D is the domain row read by the application typing rule.*
5. *C is the codomain row, together with the independent-codomain closedness boundary $ClosedAt(0, C)$.*
6. *Y is the typing row, recording the source and target application-argument typing premises consumed by the closed independent-codomain theorem.*
7. *Z is the closedness row for F, X, C inside the empty-context MetaCIC boundary.*
8. *S is the beta-step row from X to X'.*
9. *O is the obstruction boundary row, recording that the general dependent-codomain* AppArgTypeStable *discharge remains outside the packet.*
10. *H records componentwise* hsame *transport for $F, X, X', D, C, Y, Z, S, O, R, Q, N$.*
11. *R records displayed* Cont *replay from the application-argument congruence read through the independent-codomain stability theorem and the obstruction boundary.*

12. *Q is the* Pkg *provenance over* ClosedAppArgumentStability$^{\uparrow}$, SubjectReduction$^{\uparrow}$, SubjectReductionDischargeSocket ClosedLamDomainSubjectReduction$^{\uparrow}$, ClosedPiDomainSubjectReduction$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* NameCert$_{\text{ClosedAppArgumentStability}^{\uparrow}}$ *row.*

The classifier accepts only packets displaying the closed function, original argument, reduced argument, domain, independent codomain, typing row, closedness row, beta step, obstruction boundary, transport, replay, provenance, and local naming. It has no coordinate for a general dependent-codomain discharge, host conversion, erased derivation semantics, quotient syntax equality, or a closed subject-reduction theorem.

Theorem 6177.2 (Closed application-argument stability NameCert obligations)**.** *For every accepted* ClosedAppArgumentStability$^{\uparrow}$ *carrier $A = (F, X, X', D, C, Y, Z, S, O, H, R, Q, N)$, the local* NameCert$_{\text{ClosedAppArgumentStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed closed function, original argument, reduced argument, domain row, independent codomain row, typing row, closedness row, beta-step row, obstruction boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The application-argument read is accepted only when C displays the independent-codomain closedness boundary and O keeps the general dependent-codomain obstruction visible.*

Proof. Project the carrier of Definition 6177.1. The rows F, X, X', D, C display the application source, the argument step, and the Pi type whose codomain is inspected.

Read Y, Z, S next. These rows provide the typing premises, closedness premises, and beta-step premise needed by the closed independent-codomain theorem, while keeping all data inside the displayed empty-context boundary.

The obstruction row O is part of the same local surface. It prevents a consumer from replacing the independent-codomain equality by the general dependent-codomain AppArgTypeStable discharge row.

Finally H, R, Q, N transport, replay, source, and name only this finite packet. Since no row stores host conversion, erased derivation semantics, quotient syntax equality, a general application-argument discharge, or a closed subject-reduction theorem, the listed rows exhaust the local certificate. □

Theorem 6177.3 (Closed application-argument independent-codomain route)**.** *Every subject-reduction discharge consumer that reads an accepted* ClosedAppArgumentStability$^{\uparrow}$ *packet as an application-argument handoff factors through the checked independent-codomain route*

$$F,\ X,\ X',\ D,\ C,\ Y,\ Z,\ S,\ O,\ H,\ R,\ Q,\ N.$$

The handoff may cite the equality supplied by the closed independent-codomain theorem `BEDC.MetaCIC.appArgType` *but only at the displayed $ClosedAt(0, C)$ boundary. It does not supply the general* AppArgTypeStable *field, the beta-substitution row, the lambda-domain row, the Pi-domain row, or an unconditional closed subject-reduction theorem.*

Proof. Start with an application-argument consumer. The rows F, X, X', D, C, Y, Z, S match the closed independent-codomain theorem: the function has Pi type $\Pi D.\, C$, both arguments have type D, and the codomain row is closed at the binder boundary.

Apply the independent-codomain route at that displayed codomain row. The result is the equality between the two substitutions of C, because a closed codomain ignores the argument readback.

Now read the obstruction row O. It records that the dependent-codomain case remains on the obstruction side of the MetaCIC ledger and that the full AppArgTypeStable field is not generated by this packet.

The structural rows H, R, Q, N preserve the same handoff for downstream ledgers. Thus the consumer obtains the independent-codomain application-argument equality and no beta, lambda-domain, Pi-domain, general application-argument, or closed subject-reduction discharge. □

Falsifiable prediction: An accepted $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ read is refuted if an application-argument consumer reaches the independent-codomain substitution equality without displaying the closed function row, both argument typing rows, the closed codomain boundary, the beta-step row, the obstruction row, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: ClosedLamDomainSubjectReduction, ClosedPiDomainSubjectReduction, SubjectReductionDischargeSocket, SubjectReductionRouteClassifier, and MetaCICSubjectReductionObligation are not bijective with $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ because only this packet jointly carries the closed application-argument row, independent-codomain equality route, obstruction boundary for the dependent-codomain case, transport, replay, provenance, and local naming.

The Lean TasteGate file for $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ must provide explicit Nontrivial ClosedAppArgumentSta and $\mathsf{FieldFaithful\ ClosedAppArgumentStability}^{\uparrow}$ instances preserving $F, X, X', D, C, Y, Z, S, O, H, R, Q, N$ as distinct rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedAppArgumentStability}^{\uparrow}}$.

Constructive story: A $\mathsf{ClosedAppArgumentStability}^{\uparrow}$ packet is generated from a closed function row, original argument row, reduced argument row, domain row, independent codomain row, typing row, closedness row, beta-step row, obstruction boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClosedAppArgumentStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6177.1, Definition 6177.2, and Definition 6177.3 seed the finite independent-codomain application-argument packet for the closed MetaCIC discharge route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClosedAppArgumentStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general AppArgTypeStable, beta-substitution preservation, lambda-domain preservation, Pi-domain preservation, unconditional closed subject reduction, host conversion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, independent-codomain closedness exactness, typing-row exposure, beta-step exposure, obstruction-boundary exposure, subject-reduction ledger handoff, and TasteGate field faithfulness for BEDC.Derived.ClosedAppArgumentStabilityUp.

Chapter 6178

A Concrete Naming Certificate for $\mathsf{ContextSensitiveGrammar}^{\uparrow}$

Following Hopcroft–Ullman and Sipser on formal languages, the $\mathsf{ContextSensitiveGrammar}^{\uparrow}$ packet records the finite grammar surface for noncontracting productions and sentential-window derivations. It sits after the regular-language carrier of chapter 351, the context-free grammar carrier of chapter 356, the pushdown-automaton comparison row of chapter 6018, the list carrier of chapter 36, and the natural-number row of chapter 28. The packet names a finite grammar-and-derivation certificate; it does not import a host language set, unrestricted Turing-machine semantics, selected derivation trees, or ambient decidability.

Definition 6178.1 (Context-sensitive grammar carrier)**.** *A* $\mathsf{ContextSensitiveGrammar}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, R, W, D, L, Q, M, H, C, P, N)$$

with the following displayed rows:

1. *A is the finite alphabet and variable row, read through* $\mathsf{List}^{\uparrow}$ *and* $\mathsf{Nat}^{\uparrow}$ *indices.*
2. *R is the noncontracting production row. Each production has a finite left context, a nonempty rewritten window, a finite right context, and a length-growth witness showing that the successor sentential form is no shorter than the predecessor.*
3. *W is the sentential-window row, exposing only the finite word segment inspected by a derivation consumer.*
4. *D is the derivation ledger. It lists the applied production, occurrence window, predecessor word, and successor word for each displayed step.*
5. *L is the length-growth ledger recording the* $\mathsf{Nat}^{\uparrow}$ *comparison attached to every production use in D.*
6. *Q is the word-classifier row for generated terminal words, read only after the derivation ledger and length ledger have been replayed.*
7. *M records comparison rows to* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{ContextFreeGrammar}^{\uparrow}$, *and* $\mathsf{PushdownAutomaton}^{\uparrow}$ *consumers when a downstream formal-language packet asks for a weaker or adjacent interface.*
8. *H records componentwise* hsame *transport for the alphabet, production, window, derivation, length, classifier, comparison, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a word-membership request through $A, R, W, D, L, Q, M, H, P, N$.*
10. *P is the* Pkg *provenance over* ContextSensitiveGrammar$^{\uparrow}$, RegularLanguage$^{\uparrow}$, ContextFreeGrammar$^{\uparrow}$, PushdownAutomaton$^{\uparrow}$, List$^{\uparrow}$, Nat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{ContextSensitiveGrammar}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, R, W, D, L, Q, M, H, C, P, N$. It has no coordinate for a host language set, an unrestricted machine computation, a selected infinite derivation tree, an ambient decidability theorem, or a grammar equality quotient.

Lean4 ✓ `BEDC.Derived.ContextSensitiveGrammarUp.TasteGate.ContextSensitiveGrammarTasteGate`
`single carrier alignment`

Theorem 6178.2 (Context-sensitive grammar NameCert obligations)**.** *For every accepted* ContextSensitiveGramma *carrier*

$$G = (A, R, W, D, L, Q, M, H, C, P, N),$$

the local NameCert$_{\text{ContextSensitiveGrammar}^{\uparrow}}$ *surface consists exactly of carrier admission, monotone production stability, derivation-window replay, classifier transport, ledger nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The alphabet row A, noncontracting production row R, sentential window W, derivation ledger D, and length-growth ledger L are the only rows that can justify a generated word.

Monotone production stability is read from R together with L. Each derivation step records the applied production and the corresponding Nat$^{\uparrow}$ length comparison, so no shortening step can enter the accepted ledger.

Derivation-window replay is supplied by W, D, C. A consumer may inspect only the displayed occurrence windows and their predecessor and successor words, and replay preserves that finite route before the classifier row Q is exposed.

The comparison row M transports the same grammar data toward regular-language, context-free grammar, and pushdown-automaton consumers without replacing the local classifier. Since the carrier contains no host language set, unrestricted machine computation, selected infinite derivation, ambient decidability theorem, or quotient grammar equality, the listed obligations exhaust the NameCert surface. □

Theorem 6178.3 (Context-sensitive grammar linear-bounded handoff)**.** *Every linear-bounded-automaton consumer of an accepted* ContextSensitiveGrammar$^{\uparrow}$ *packet must read the displayed grammar, word-window,* List$^{\uparrow}$, Nat$^{\uparrow}$, PushdownAutomaton$^{\uparrow}$, *and* ContextFreeGrammar$^{\uparrow}$ *comparison rows before any language-level claim is exposed. For*

$$G = (A, R, W, D, L, Q, M, H, C, P, N),$$

the route first fixes A, R, W, D, L, then reads the comparison row M, and only then exposes the classifier row Q through H, C, P, N. The handoff exports no host language equality, unrestricted machine semantics, selected accepting computation, global decidability theorem, or hidden grammar quotient.

Proof. Read the carrier of Definition 6178.1. The grammar and word-window coordinates are A, R, W, D, L, and they are indexed only through the displayed List$^{\uparrow}$ and Nat$^{\uparrow}$ rows.

Apply Definition 6178.2. A language-facing consumer cannot see Q until the production ledger and length ledger have been replayed, because monotone production stability and derivation-window replay are local obligations.

The comparison row M is the only coordinate that connects the packet to the pushdown-automaton and context-free grammar siblings. A linear-bounded-automaton consumer therefore reads through that row before it makes any language-level use of the generated-word classifier.

Finally H, C, P, N transport, replay, source, and locally name the same finite route. The excluded host objects are not coordinates of G, so the handoff remains a finite grammar certificate rather than a semantic language theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{ContextSensitiveGrammar}\uparrow}$.

Constructive story: A $\mathsf{ContextSensitiveGrammar}^{\uparrow}$ packet is generated from finite alphabet and variable rows, noncontracting production rows, sentential-window rows, derivation ledgers, length-growth ledgers, formal-language comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ContextSensitiveGrammar}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6178.1, Definition 6178.2, and Definition 6178.3 seed the finite noncontracting grammar route over List, Nat, RegularLanguage, ContextFreeGrammar, PushdownAutomaton, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ContextSensitiveGrammar}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, unrestricted machine semantics, host language equality, selected accepting computations, global decidability, or grammar quotient equality.

Upgrade path: Obligation closure requires checked carrier admission, monotone production stability, derivation-window replay, classifier transport, ledger nonescape, comparison-row handoff, transport, replay, provenance, and local naming for BEDC.Derived.ContextSensitiveGrammarUp.

Chapter 6179

A Concrete Naming Certificate for BoundedNormalizationFrontier$^{\uparrow}$

The BoundedNormalizationFrontier$^{\uparrow}$ packet realizes the MetaCIC decidability audit map of chapter 9379 as finite fuel, finished-normal, bounded-comparison, and refusal rows. It sits beside the decidable typing frontier of chapter 3616, the typed normal-form decider of chapter 3714, the bounded-normalization fuel packet of chapter 5962, the bounded normal-equality checker of chapter 752, and the finite observation fold of chapter 2236. The packet names the exact frontier where `normalizeBounded` and `decideClosedBetaConv` are finite procedures; it does not claim unrestricted beta-conversion decidability.

Falsifiable prediction: Every accepted BoundedNormalizationFrontierUp conversion consumer must expose typed term rows, a shared fuel row, normalizeBounded trace rows, finished-normal witness rows, bounded Term.eq comparison rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows before any conversion claim is available.

Independence witness: metacic-decidable-typing-frontier, typed-normal-form-decider, bounded-normalization-fuel, bounded-normal-equality-checker, finite-observation-fold: BoundedNormalizationFrontier$^{\uparrow}$ is not bijective to any listed sibling because it binds fuel, trace, finished-normal, bounded-comparison, refusal, replay, transport, provenance, and local naming as one frontier certificate with proposed Lean TasteGate obligations.

Definition 6179.1 (Bounded normalization frontier carrier)**.** *A* BoundedNormalizationFrontier$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (T, U, B, X, Y, E, O, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed term row, carrying the closed MetaCIC terms and their displayed typing context boundary.*
2. *U is the shared finite fuel row used by the bounded normalizer.*
3. *B is the* `normalizeBounded` *trace row for both terms at fuel U.*
4. *X is the finished-normal witness row, present only when the bounded traces are certified as closed finished normal forms.*

5. *Y is the bounded* `Term.eq` *comparison row over the two bounded normalizer outputs.*
6. *E is the bounded conversion result row exposed by* `decideClosedBetaConv` *at the displayed fuel.*
7. *O is the refusal and obstruction row. It records the missing endpoint data whenever the packet cannot promote a bounded comparison to unrestricted beta-conversion.*
8. *H records componentwise* hsame *transport for typed terms, fuel, bounded traces, finished-normal witnesses, bounded equality, refusal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a conversion request through $T, U, B, X, Y, E, O, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedNormalizationFrontier}^{\uparrow}$*, MetaCIC typed terms,* `normalizeBounded`*,* `Term.eq`*,* `decideClosedBetaConv`*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFrontier}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, U, B, X, Y, E, O, H, C, P, N$. It has no coordinate for global normalization, unrestricted beta-conversion decidability, fuel-erased confluence, selected normal forms outside the trace, or a host equality oracle.

Theorem 6179.2 (Bounded normalization frontier NameCert obligations)**.** *For every accepted* $\mathsf{BoundedNormalizationFrontier}^{\uparrow}$ *carrier*

$$F = (T, U, B, X, Y, E, O, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFrontier}^{\uparrow}}$ *surface consists exactly of typed-term admission, finite-fuel admission, bounded-normalizer trace replay, finished-normal witness exposure, bounded* `Term.eq` *comparison, bounded conversion transport, refusal-row nonescape, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier. The typed terms are read through T, the computation budget through U, and the bounded normalizer traces through B.

Finished-normal evidence is separate. The row X records the endpoint witnesses that make the bounded traces closed normal forms; if those witnesses are absent, the refusal row O is the only exposed endpoint information.

The bounded comparison uses Y and E. It compares the two displayed bounded normalizer outputs by `Term.eq` and exposes only the finite result at fuel U.

The structural rows H, C, P, N transport, replay, source, and locally name the same route. Since the carrier contains no row for global normalization, unrestricted beta-conversion decidability, fuel-erased confluence, selected external normal forms, or host equality, the listed obligations exhaust the NameCert surface. □

Theorem 6179.3 (Bounded conversion frontier handoff)**.** *Every conversion consumer of an accepted* $\mathsf{BoundedNormalizationFrontier}^{\uparrow}$ *packet receives a bounded frontier certificate, not an unrestricted beta-conversion decider. For*

$$F = (T, U, B, X, Y, E, O, H, C, P, N),$$

the consumer must read typed terms T, finite fuel U, bounded traces B, finished-normal witnesses X when present, bounded `Term.eq` *comparison Y, bounded conversion result E, and the refusal row O before any conversion-facing claim is exposed through H, C, P, N.*

Proof. Start from the carrier inventory in Definition 6179.1. It stores a concrete fuel value and concrete bounded traces, so every conversion request is already a finite-fuel request.

Apply Definition 6179.2. The comparison route is T, U, B, X, Y, E, O in that order: typed terms and fuel determine the traces, finished-normal witnesses mark the closed endpoint case, the bounded equality row compares the displayed outputs, and the refusal row records the missing endpoint data when no global conclusion is available.

The rows H, C, P, N preserve that finite route. Because unrestricted normalization and fuel-erased beta-conversion are absent from the carrier, a consumer cannot bypass the bounded frontier by appealing to a host conversion theorem or selected normal form. □

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedNormalizationFrontier}\uparrow}$.

Constructive story: A $\mathsf{BoundedNormalizationFrontier}^{\uparrow}$ packet is generated from typed term rows, finite fuel rows, bounded normalizer traces, finished-normal witnesses, bounded Term.eq comparison rows, bounded conversion result rows, refusal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationFrontier}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6179.1, Definition 6179.2, and Definition 6179.3 seed the finite MetaCIC bounded-normalization route over typed terms, fuel, traces, finished-normal witnesses, bounded equality, refusal, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedNormalizationFrontier}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global normalization, unrestricted beta-conversion decidability, fuel-erased confluence, selected external normal forms, or host equality.

Upgrade path: Obligation closure requires checked typed-term admission, finite-fuel admission, bounded trace replay, finished-normal witness exposure, bounded equality comparison, refusal-row nonescape, conversion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BoundedNormalizationFrontierUp.

Chapter 6180

A Concrete Naming Certificate for $\mathsf{BishopLocatedReal}^{\uparrow}$

From Bishop–Bridges constructive real analysis, the $\mathsf{BishopLocatedReal}^{\uparrow}$ packet records the finite located-real interface consumed by L10 Real and Completion routes. It sits after the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the located interval surface of chapter 394, the apartness evidence surface of chapter 304, the located Cauchy route of chapter 1850, and the terminal real-name seal of chapter 38. The packet supplies a displayed located-cut boundary for later completion and order consumers; it is not host real equality, quotient-class equality, trichotomy, countable choice, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6180.1 (BishopLocatedReal carrier)**.** *A* $\mathsf{BishopLocatedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, R, I, A, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying the rational cells, dyadic split radii, slack rows, and comparison budget used by every located read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected at the dyadic tolerances recorded in D.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback obtained from the same stream windows and tolerance ledger.*
4. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row naming the finite interval enclosure in which the located real may be inspected.*
5. *A is the* $\mathsf{ApartnessReal}^{\uparrow}$ *positive-separation row available only when a displayed window exhibits a strict gap from one side of the located cut.*
6. *B is the* $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$ *row binding the located interval and regular Cauchy readback to one shared dyadic tolerance route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, S, R, I, A, B have been displayed.*
8. *H records componentwise* hsame *transport for the dyadic ledger, stream windows, regular readback, located interval, apartness row, located Cauchy row, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a located-cut request through D, S, R, I, A, B, E, H.*
10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these eleven rows. It has no coordinate for host real equality, quotient classes, trichotomy, countable choice, selected representatives, or an ambient completeness assertion.

Lean4 ✓ `BEDC.Derived.BishopLocatedRealUp.BishopLocatedRealTasteGate single carrier alignment`

Theorem 6180.2 (BishopLocatedReal NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedReal}^{\uparrow}$ *carrier $L = (D, S, R, I, A, B, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedReal}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance admission, finite StreamName window admission, RegSeqRat readback, located interval admission, apartness-row admission, Bishop located Cauchy admission, Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the displayed rows $D, S, R, I, A, B, E, H, C, P, N$.*

Proof. Project the accepted packet. The analytic rows are precisely the dyadic tolerance ledger D, the finite stream windows S, the regular rational readback R, the located interval row I, the apartness row A, the located Cauchy row B, and the terminal Real seal E.

The local certificate reads these rows in the displayed order. A located request may inspect endpoint or cut information only after the shared tolerance and window rows are visible, and a Real-facing request reaches E only after the regular readback, located interval, apartness, and located Cauchy rows have been displayed.

Transport and replay remain componentwise. The rows H, C, P, N transport, replay, source, and name the same finite packet, so no outside equality, choice principle, selected representative, trichotomy split, or ambient completeness datum can enter the local surface. □

Theorem 6180.3 (BishopLocatedReal RegSeqRat route)**.** *Every completion-facing read of an accepted* $\mathsf{BishopLocatedReal}^{\uparrow}$ *carrier factors through the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{BishopLocatedCauchyReal}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exposes a dyadic tolerance ledger, a finite stream window, a regular rational readback, a located Cauchy handoff, componentwise transport, replay, provenance, and local naming before any terminal real seal is consumed.

Proof. Begin with Definition 6180.1. The only regular rational source row is R, and R is downstream of the dyadic ledger D and the finite stream window S.

The located Cauchy row B is admitted only as a row of the same packet. It binds the regular readback to the displayed located interval data before the terminal Real seal E can be read.

Apply Definition 6180.2. The certificate forces every completion-facing read through $D, S, R, B, E, H, C, P, N$, and the carrier contains no row for quotient streams, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Theorem 6180.4 (BishopLocatedReal located-cut stability)**.** *Let $L = (D, S, R, I, A, B, E, H, C, P, N)$ and $L' = (D', S', R', I', A', B', E', H', C', P', N')$ be accepted* $\mathsf{BishopLocatedReal}^{\uparrow}$ *carriers. If H*

records componentwise hsame *transport from each displayed row of L to the corresponding row of L', then the located-cut information of L transports to L' only through*

$$D, S, R, I, A, B, E, H, C, P, N.$$

In particular, located interval containment, positive apartness evidence, and located Cauchy compatibility remain attached to the same finite StreamName, RegSeqRat, and DyadicRatCore route.

Proof. Project both accepted carriers. The located-cut rows are I, A, B: the interval row I records the finite enclosure, the apartness row A records positive gap evidence, and the located Cauchy row B binds those observations to the regular readback.

Componentwise transport through H carries D to D', S to S', R to R', and then I, A, B to I', A', B'. The replay row C cannot change the order of these reads; it only replays the displayed route.

The NameCert obligations of Definition 6180.2 then repack the transported rows as an accepted located-cut surface for L'. Since the packet has no host equality, trichotomy, quotient, choice, or completeness coordinate, stability is exactly the displayed row-local transport. □

Theorem 6180.5 (BishopLocatedReal Real-seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{BishopLocatedReal}^{\uparrow}$ *carrier is exhausted by the displayed rows $D, S, R, I, A, B, E, H, C, P, N$. Thus the public real-name boundary supplies only dyadic tolerance data, finite stream windows, RegSeqRat readback, located interval evidence, ApartnessReal separation rows, BishopLocatedCauchyReal compatibility, the terminal Real seal, transport, replay, provenance, and local naming.*

Proof. Read the carrier of Definition 6180.1. The terminal Real row is E, and E appears only after the dyadic, stream, regular rational, located interval, apartness, and located Cauchy rows.

The RegSeqRat route of Definition 6180.3 keeps every completion-facing read downstream of D, S, R, B. The stability result of Definition 6180.4 keeps located interval and apartness reads on the same displayed route.

Finally, Definition 6180.2 supplies H, C, P, N and excludes all non-displayed data. Therefore no consumer can read host real equality, quotient classes, trichotomy, countable choice, a selected representative, or ambient $\mathsf{Real}^{\uparrow}$ completeness from the Real-seal boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedReal}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedReal}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a LocatedInterval row, an ApartnessReal row, a BishopLocatedCauchyReal row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedReal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6180.1, Definition 6180.2, Definition 6180.3, Definition 6180.4, and Definition 6180.5 seed the finite located-real interface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{ApartnessReal}^{\uparrow}$, $\mathsf{BishopLocatedCauchyReal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedReal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host real equality, quotient classes, trichotomy, countable choice, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, RegSeqRat route exactness, located-cut transport, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedRealUp.

Chapter 6181

A Concrete Naming Certificate for $\mathsf{ExplanationComplexityInformation}^{\uparrow}$

The $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ packet realizes the net-information calculus of section 9462.10 as finite NameCert rows. It sits beside the bounded program-cost surface of chapter 5989, the debt-aware quality surface of chapter 6117, and the classifier-shift witness of chapter 6138. The packet names a finite explanation-information read over score, benefit, debt, novelty, and quality-gate evidence; it is not a hidden semantic truth predicate, an oracle for shortest descriptions, a raw benchmark score, or a classifier novelty claim without a displayed witness.

Definition 6181.1 (Explanation-complexity information carrier)**.** *An* $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (S, B, D, K, V, Q, G, H, C, P, N)$$

with the following displayed rows:

1. *S is the explanation score row, recording the public weight profile and the finite source, feature, circuit, classifier, stability, intervention, ledger, seal, semantic, and verification cost coordinates.*
2. *B is the benefit row, recording coverage gain, prediction gain, transfer gain, intervention gain, verification gain, and unification gain inside the declared explanation scope.*
3. *D is the explanation-debt row, separating NDNA debt, ledger debt, semantic debt, black-box debt, shortcut debt, safety debt, generalization debt, and verification debt.*
4. *K is the bounded description-cost row, read through the displayed program and code discipline of* $\mathsf{KolmogorovComplexity}^{\uparrow}$ *without importing an uncomputable minimum selector.*
5. *V is the classifier-novelty row, read through* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ *only when two explanation states display a common source and a changed classifier judgment.*
6. *Q is the certified-quality row, read through* $\mathsf{CertifiedModelQuality}^{\uparrow}$ *when the score, debt, audit, and quality-gate rows jointly certify a model-quality comparison.*
7. *G is the net information row, displaying the accepted arithmetic comparison between benefit and the score-plus-debt surface over the same scope.*
8. *H records componentwise* hsame *transport for score, benefit, debt, description-cost, classifier-novelty, certified-quality, net-information, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from scope inspection through score, benefit, debt, description-cost, classifier novelty, certified quality, net information, provenance, and local naming.*
10. *P is the* Pkg *provenance over* $\mathsf{ExplanationComplexityInformation}^{\uparrow}$, $\mathsf{KolmogorovComplexity}^{\uparrow}$, $\mathsf{CertifiedModelQuality}^{\uparrow}$, $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, $\mathsf{AuditSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ExplanationComplexityInformation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, B, D, K, V, Q, G, H, C, P, N$. It has no coordinate for explanation brevity alone, private evaluator preference, hidden semantic truth, unbounded Kolmogorov minimization, raw model score, or classifier novelty without the displayed changed-judgment row.

Theorem 6181.2 (Explanation-complexity information NameCert obligations)**.** *For every accepted* $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ *carrier $E = (S, B, D, K, V, Q, G, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ExplanationComplexityInformation}^{\uparrow}}$ *surface consists exactly of scope-coherent score exposure, benefit exposure, explanation-debt exposure, bounded description-cost handoff through* $\mathsf{KolmogorovComplexity}^{\uparrow}$, *classifier-novelty handoff through* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, *certified-quality handoff through* $\mathsf{CertifiedModelQuality}^{\uparrow}$, *net-information comparison, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6181.1. The analytic rows are S, B, D, K, V, Q, G, and the structural rows are H, C, P, N.

The score, benefit, and debt rows share one explanation scope. Thus the net row G may compare benefit against score and debt only after S, B, D have been displayed for that same scope.

Read the sibling rows next. The description-cost row K uses chapter 5989 only as a bounded displayed-program cost surface; the novelty row V uses chapter 6138 only when a changed classifier judgment is visible; and the quality row Q uses chapter 6117 only after its debt and audit gates are exposed.

Finally, H, C, P, N transport, replay, source, and name the same rows. Since none of the excluded objects appears in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 6181.3 (Explanation-complexity information net-score route)**.** *An accepted* ExplanationComplexityInform *packet may export positive net explanation information only after the score row S, benefit row B, debt row D, classifier-novelty row V, and certified-quality row Q have all been read through the same scope. A lower score, a shorter description, a quality-gate pass, or a classifier-novelty witness alone is therefore insufficient for the exported net-information row G.*

Proof. Start from the net-information row G. By carrier projection, G is downstream of the public score row S, benefit row B, debt row D, classifier-novelty row V, and certified-quality row Q.

The score and description rows do not determine the export alone. The row S records weighted cost, while K records bounded description-cost evidence through $\mathsf{KolmogorovComplexity}^{\uparrow}$; neither row contains the benefit and debt comparison required by G.

The novelty and quality rows are also insufficient in isolation. The row V certifies a changed classifier judgment, and Q certifies debt-aware model quality, but G must still read the shared score, benefit, and debt surface before the packet exports positive net information. □

Falsifiable prediction: Every accepted $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ read exposes score, benefit, debt, bounded description-cost, classifier-novelty, certified-quality, net-information, trans-

port, replay, provenance, and local naming rows before it claims positive net explanation information.

Independence witness: kolmogorov-complexity, certified-model-quality, classifier-novelty-witness, effective-replacement-ledger, auditsystem: none is bijective with $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ because the packet simultaneously binds weighted explanation score, benefit, debt, bounded description cost, classifier novelty, certified quality, net-information comparison, transport, replay, provenance, and local naming as one finite explanation-information surface.

The Lean follow-up must provide $\mathsf{Nontrivial\ ExplanationComplexityInformation}^{\uparrow}$ and $\mathsf{FieldFaithful\ ExplanationComp}$ instances before checked TasteGate markers for BEDC.Derived.ExplanationComplexityInformationUp are cited. Those instances must preserve the rows $S, B, D, K, V, Q, G, H, C, P, N$ so the checked surface separates positive net information from short descriptions, raw score movement, quality-gate success, and classifier novelty alone.

Closure status for $\mathsf{NameCert}_{\mathsf{ExplanationComplexityInformation}^{\uparrow}}$.

Constructive story: An $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ packet is generated from public explanation score, benefit, debt, bounded description-cost, classifier-novelty, certified-quality, net-information, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExplanationComplexityInformation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6181.1, Definition 6181.2, and Definition 6181.3 seed the finite packet for net explanation-information reads over score, benefit, debt, bounded description cost, classifier novelty, certified quality, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExplanationComplexityInformation}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ExplanationComplexityInformationUp`

Bridge status: none

Not claimed: This chapter does not close hidden semantic truth, unbounded shortest-description selection, raw benchmark validity, global model safety, classifier novelty without a displayed witness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, score exposure, benefit exposure, debt exposure, bounded description-cost handoff, classifier-novelty handoff, certified-quality handoff, net-information comparison, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.ExplanationComplexityInformat

Chapter 6182

A Concrete Naming Certificate for $\mathsf{WijsmanTopology}^{\uparrow}$

Following Wijsman and the Kelley hyperspace tradition, the $\mathsf{WijsmanTopology}^{\uparrow}$ packet records the finite topology read induced by distance-to-set probes on a metric hyperspace. It starts from the closed-subset source of chapter 1660, the metric source of chapter 57, and the distance comparison boundary of chapter 3653, while keeping explicit comparison rows to the hit/miss surfaces of chapter 2375 and chapter 6157. The packet is a NameCert for the Wijsman distance-evaluation topology, not a host topology on a powerset of closed subsets, a quotient of closed-set equality, a selected nearest point, or an ambient completeness principle.

Definition 6182.1 (Wijsman topology carrier)**.** *A* $\mathsf{WijsmanTopology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (M, H, D, S, R, E, V, F, T, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row. It supplies the only distance primitive available to the packet.*
2. *H is the closed-subset* $\mathsf{Hyperspace}^{\uparrow}$ *source row whose finite closed-set names are being inspected.*
3. *D is the finite distance-to-set evaluation row. Each entry records a displayed metric point, a displayed closed-subset name, and the bounded evaluation request for the corresponding distance value.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to replay point names, closed-subset names, and distance-evaluation probes.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the finite evaluation windows.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the displayed distance comparisons.*
7. *V is the* $\mathsf{VietorisTopology}^{\uparrow}$ *comparison row, recording which finite hit/miss reads are visible before distance-evaluation inspection begins.*
8. *F is the* $\mathsf{FellTopology}^{\uparrow}$ *comparison row, recording which lower-hit and compact-miss reads remain separate from the Wijsman distance rows.*
9. *T records componentwise* hsame *transport for M, H, D, S, R, E, V, F.*

10. *C records displayed* Cont *replay of metric source admission, hyperspace source admission, distance-to-set evaluation, stream-window readback, regular rational readback, real comparison, Vietoris comparison, and Fell comparison.*

11. *P is the* Pkg *provenance over* $\mathsf{WijsmanTopology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, $\mathsf{FellTopology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{WijsmanTopology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, H, D, S, R, E, V, F, T, C, P, N$ and exports no host closed-set topology, quotient closed-set equality, selected nearest point, or ambient completeness coordinate.

Lean4 ✓ `BEDC.Derived.WijsmanTopologyUp.WijsmanTopologyTasteGate single carrier alignment`

Theorem 6182.2 (Wijsman topology NameCert obligations)**.** *For every accepted* $\mathsf{WijsmanTopology}^{\uparrow}$ *carrier*

$$W = (M, H, D, S, R, E, V, F, T, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WijsmanTopology}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the metric source, closed-subset hyperspace source, finite distance-to-set evaluation probes, stream windows, regular rational readbacks, real comparisons, Vietoris comparison, Fell comparison, transport, replay, provenance, and local naming rows;*

2. *distance-evaluation exposure, requiring each topology-facing read to display the metric point, closed-subset name, stream window, regular rational readback, and real comparison row used by its distance-to-set probe;*

3. *metric-source compatibility, requiring every evaluation row in D to read its distance primitive through the displayed* $\mathsf{Met}^{\uparrow}$ *row M;*

4. *comparison non-collapse, requiring the Vietoris row V and Fell row F to remain visible comparison rows rather than replacements for the Wijsman distance-evaluation classifier;*

5. *componentwise* hsame *transport through T, displayed* Cont *replay through C, and* Pkg *provenance through P for the same finite route;*

6. *local naming through N, excluding host closed-set topologies, quotient closed-set equality, selected nearest points, and ambient completeness.*

No Wijsman topology obligation is stored outside these rows.

Proof. Project the carrier of Definition 6182.1. The only source rows are the metric row M and the closed-subset hyperspace row H, and the only topology-facing observation row is the finite distance-to-set ledger D.

The evaluation row is read through finite windows. The rows S, R, E force every distance-to-set probe to pass through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ before it becomes a topology-facing comparison.

The comparison rows are kept separate. The Vietoris row V and Fell row F may be inspected as neighbouring hyperspace topologies, but they do not replace the distance-evaluation route M, H, D, S, R, E.

Finally, T, C, P, N transport, replay, source, and name that same route. Since the excluded host topology, quotient closed-set equality, selected nearest point, and ambient completeness coordinates are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6182.3 (Wijsman distance-evaluation handoff)**.** *Every topology-facing read from an accepted* $\mathsf{WijsmanTopology}^{\uparrow}$ *carrier factors through*

$$\mathsf{Met}^{\uparrow},\ \mathsf{Hyperspace}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{VietorisTopology}^{\uparrow},\ \mathsf{FellTopology}^{\uparrow}$$

before a hyperspace topology consumer can inspect it. The exported read is the displayed metric source, closed-subset hyperspace source, finite distance-to-set evaluation probe, stream-window replay, regular rational readback, real comparison, neighbouring Vietoris comparison, neighbouring Fell comparison, transport, replay, provenance, and local naming route; it exports no host closed-set topology, quotient closed-set equality, selected nearest point, or ambient completeness.

Proof. Start with the NameCert obligations of Definition 6182.2. Carrier admission places the metric source and closed-subset hyperspace source before any distance-evaluation row can be consumed.

Next read the evaluation route. The finite distance-to-set row D can expose a topology-facing comparison only after the stream window S, the regular rational readback R, and the real seal E have been displayed.

The surrounding hyperspace topology context is explicit. The rows V and F expose the Vietoris and Fell comparison surfaces as neighbours, while T, C, P, N preserve the same finite route under transport, replay, provenance, and local naming. No unlisted host topology, closed-set quotient, nearest-point selector, or ambient completeness row can enter the handoff. □

Falsifiable prediction: Every accepted $\mathsf{WijsmanTopology}^{\uparrow}$ topology read factors through $\mathsf{Met}^{\uparrow}$, $\mathsf{Hyperspace}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{VietorisTopology}^{\uparrow}$, and $\mathsf{FellTopology}^{\uparrow}$ before any hyperspace topology consumer can inspect it.

Independence witness: metric, hyperspace, hausdorff-distance, vietoris-topology, fell-topology, streamname, regseqrat, real: $\mathsf{WijsmanTopology}^{\uparrow}$ is not bijective to any listed sibling because it binds metric distance-to-set evaluation probes to closed-subset hyperspace names while keeping Vietoris and Fell comparison rows visible.

Closure status for $\mathsf{NameCert}_{\mathsf{WijsmanTopology}^{\uparrow}}$.

Constructive story: A $\mathsf{WijsmanTopology}^{\uparrow}$ packet is generated from a Metric row, a closed-subset Hyperspace row, finite distance-to-set evaluation probes, StreamName windows, RegSeqRat readbacks, Real comparison seals, Vietoris and Fell comparison rows, hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WijsmanTopology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6182.1, Definition 6182.2, and Definition 6182.3 seed the finite Wijsman distance-evaluation topology surface over Metric, Hyperspace, StreamName, RegSeqRat, Real, VietorisTopology, FellTopology, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WijsmanTopology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host closed-set topologies, quotient closed-set equality, selected nearest points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, distance-evaluation exposure, metric-source compatibility, comparison non-collapse, transport, replay, provenance, and local naming for BEDC.Derived.WijsmanTopologyUp.

Chapter 6183

A Concrete Naming Certificate for $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$

Following Bishop-style cofinal selection in constructive Cauchy arguments, the $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ packet records the finite route by which a regular Cauchy family supplies a cofinal subnet or subsequence window for diagonal-limit consumers. It sits over the finite observation discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the tail-equivalence boundary of chapter 1738, and the terminal real-name seal of chapter 38. The packet is a finite selector certificate for cofinal Cauchy windows, not countable choice, a quotient stream, a selected real limit, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6183.1 (Regular Cauchy subnet selector carrier)**.** *A* $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, I, W, R, D, E, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the source regular Cauchy family row whose visible windows are being routed to a cofinal selector.*
2. *I is the cofinal index ledger. It records the finite monotone or directed comparison data needed to show that each requested tail window is eventually reached.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the source family and selected subnet windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row extracted from the displayed selected windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row carrying the dyadic budgets used by I, W, R.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the selected windows and tolerance row have been displayed.*
7. *T is the* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *witness row recording that the selected cofinal windows preserve the same regular Cauchy tail route as the source family.*
8. *H records componentwise* hsame *transport for X, I, W, R, D, E, T.*
9. *C records displayed* Cont *replay of source-family admission, cofinal-index inspection, stream-window extraction, regular rational readback, dyadic tolerance comparison, real-seal handoff, and tail-equivalence transport.*

10. *P is the* Pkg *provenance over* $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyTailEquivalence}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySubnetSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, I, W, R, D, E, T, H, C, P, N$ and exports no countable-choice selector, quotient stream equality, selected limit, host subnet object, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubnetSelectorUp.RegularCauchySubnetSelectorTasteGate single carrier alignment`

Theorem 6183.2 (Regular Cauchy subnet selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ *carrier*

$$S = (X, I, W, R, D, E, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchySubnetSelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the source regular Cauchy family, cofinal index ledger,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{Real}^{\uparrow}$ *seal,* $\mathsf{CauchyTailEquivalence}^{\uparrow}$ *witness, componentwise transport, displayed replay, package provenance, and local naming rows. Every accepted selector read must expose I before it consumes W, expose W, D before it consumes R, expose R, D, T before it consumes E, and reach H, C, P, N only as structural support for that same finite route.*

Proof. Project the carrier of Definition 6183.1. The mathematical rows are exactly the source family X, the cofinal ledger I, the window row W, the readback row R, the tolerance row D, the real seal E, and the tail-equivalence row T.

The cofinal ledger is inspected before any selected window is read. This prevents the selector from acting as a free choice principle; each visible subnet window is justified by the finite comparison data already stored in I.

The regular Cauchy data then flows through W, R, D, T before reaching the real seal E. Transport through H, replay through C, provenance through P, and local naming through N preserve that route and add no quotient stream, selected limit, host subnet object, trichotomy, or ambient completeness coordinate. □

Theorem 6183.3 (Regular Cauchy subnet selector handoff)**.** *Every diagonal-limit or real-completeness read from an accepted* $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ *carrier factors through*

$$\mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CauchyTailEquivalence}^{\uparrow}$$

before the selected cofinal windows may be consumed. The exported route contains only the source regular Cauchy family row, cofinal index ledger, finite stream windows, regular rational readback, dyadic tolerance row, real seal, tail-equivalence witness, transport, replay, provenance, and local naming; it exports no countable choice, quotient stream equality, selected real limit, host subnet object, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchySubnetSelectorUp.RegularCauchySubnetSelectorCarrier handoff`

Proof. Begin with the obligation package of Definition 6183.2. A consumer cannot inspect selected windows until the cofinal index ledger I has displayed the finite directed comparison route for the requested tail.

Next route the selected windows through the regular Cauchy surface. The stream row W, regular rational readback R, dyadic tolerance row D, and tail-equivalence witness T jointly account for the finite-window extraction before the terminal $\mathsf{Real}^{\uparrow}$ seal E can be used.

The structural rows H, C, P, N transport, replay, source, and name the same handoff. Since the carrier has no coordinate for countable choice, quotient stream equality, selected limits, host subnet objects, trichotomy, or ambient completeness, the displayed route is the entire consumer surface. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ diagonal-limit read displays a source regular Cauchy family row, a cofinal index ledger, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance data, a CauchyTailEquivalence witness, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before real-completeness consumers can inspect it.

Independence witness: streamname, regseqrat, dyadicratcore, cauchy-tail-equivalence, real: $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ is not bijective to any listed sibling because it binds cofinal index data to selected regular Cauchy windows and their tail-equivalence route as one finite selector packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySubnetSelector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchySubnetSelector}^{\uparrow}$ packet is generated from a source regular Cauchy family, a cofinal index ledger, StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance row, a RealUp seal, a CauchyTailEquivalence witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchySubnetSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6183.1, Definition 6183.2, and Definition 6183.3 seed the finite cofinal-window selector surface for diagonal-limit and real-completeness consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchySubnetSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close countable choice, quotient stream equality, selected real limits, host subnet objects, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cofinal-ledger exposure, finite-window extraction, regular readback, dyadic tolerance accounting, Real seal handoff, tail-equivalence preservation, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySubnetSelectorUp.

Chapter 6184

A Concrete Naming Certificate for $\mathsf{FiniteStateTransducer}^{\uparrow}$

Following Hopcroft–Ullman finite transducers, the $\mathsf{FiniteStateTransducer}^{\uparrow}$ packet records the finite output-producing surface of a deterministic finite-state machine as displayed BEDC data. It sits after the acceptance-only automaton surface of chapter 2364 and the finite observation discipline of chapter 126. The packet names finite control with a synchronized output ledger; it does not assert language quotient equivalence, minimization, nondeterministic branch choice, unbounded stream processing, or a host function from all words to all output words.

Definition 6184.1 (FiniteStateTransducerUp carrier)**.** *A* $\mathsf{FiniteStateTransducer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o, H, C, P, N)$$

with the following displayed rows:

1. *Q is the finite state carrier row.*
2. *Σ is the input symbol row naming the finite alphabet window read by the packet.*
3. *Γ is the output symbol row naming the finite output alphabet window produced by the packet.*
4. *δ is the transition ledger whose displayed entries consume a state row and an input symbol row and return the next state row.*
5. *λ is the output ledger whose displayed entries consume the same state and input symbol rows and return the finite output-symbol block emitted at that step.*
6. *q_0 is the start-state row inside Q.*
7. *w is the finite input window, read through the* $\mathsf{StreamName}^{\uparrow}$ *inspection surface.*
8. *ρ is the transition trace ledger beginning at q_0 and reading w one symbol at a time through δ.*
9. *o is the accumulated output ledger obtained by replaying λ along the same trace ρ.*
10. *H records componentwise* hsame *transport for the state, input-symbol, output-symbol, transition, output, start, input-window, trace, output-ledger, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from the input window through the transition ledger and output ledger at each finite step.*

12. *P is the* Pkg *provenance over* $\mathsf{FiniteStateTransducer}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteStateTransducer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o, H, C, P, N$. The carrier has no coordinate for a host word function, output-stream totality, hidden state, minimized transducer, nondeterministic branch, or language-quotient equality.

Lean4 ✓ `BEDC.Derived.FiniteStateTransducerUp.FiniteStateTransducerTasteGate single carrier alignment`

Theorem 6184.2 (FiniteStateTransducerUp NameCert obligations)**.** *For every accepted* $\mathsf{FiniteStateTransducer}^{\uparrow}$ *carrier*

$$T = (Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteStateTransducer}^{\uparrow}}$ *surface consists exactly of carrier admission, finite state exposure, input-symbol exposure, output-symbol exposure, transition-ledger replay, output-ledger replay, start-state exposure, finite input-window exposure, transition-trace exposure, accumulated-output exposure, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance over* $\mathsf{FiniteAutomaton}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*, and local naming through N.*

Proof. Project the accepted carrier of Definition 6184.1. The rows $Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o$ name the finite control surface, the inspected input window, and the synchronized output surface.

Replay is finite. The input window w is read through the same finite-window discipline used by $\mathsf{StreamName}^{\uparrow}$ consumers, and each step reads one displayed transition entry in δ and one displayed output entry in λ over the same state and input-symbol rows.

The structural rows H, C, P, N transport, replay, package, and name exactly those rows. Since neither the carrier nor the local certificate contains a host word function, hidden state, minimized quotient, nondeterministic branch, output-stream totality, or language-quotient equality, the listed rows exhaust the NameCert surface. □

Lean4 ✓ `BEDC.Derived.FiniteStateTransducerUp.FiniteStateTransducerCarrier namecert obligations`

Theorem 6184.3 (FiniteStateTransducerUp output-ledger determinacy)**.** *Let $T = (Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o, H, C, P$ and $T' = (Q', \Sigma', \Gamma', \delta', \lambda', q_0', w', \rho', o', H', C', P', N')$ be accepted* $\mathsf{FiniteStateTransducer}^{\uparrow}$ *packets. If their finite state, input symbol, output symbol, transition ledger, output ledger, start-state, input-window, transport, replay, provenance, and local NameCert rows agree componentwise by the displayed* hsame *witnesses, then their accumulated output ledgers agree along the same finite trace. In particular, no consumer can obtain two different output ledgers from the same displayed transition and output rows.*

Proof. Use Definition 6184.1 to project both packets to their displayed rows. The only coordinates that can affect the accumulated output ledger are the finite input window, the start state, the transition ledger, and the output ledger.

Proceed along the displayed input window. At the empty window, both packets are at the displayed start-state row and the accumulated output is the empty finite ledger. At a nonempty window, the common state and input-symbol rows select the displayed transition entry in δ and the displayed output entry in λ; the Cont rows then replay the tail window over the transported next-state row.

The hsame rows transport each selected transition and output entry componentwise, while P and N keep both packets under the same $\mathsf{FiniteAutomaton}^{\uparrow}$ and $\mathsf{StreamName}^{\uparrow}$ provenance. There is

no remaining coordinate from which a second output ledger could be selected, so the accumulated output row is determined by the displayed finite route. □

Theorem 6184.4 (FiniteStateTransducerUp stream output trace obligation)**.** *Every output-producing read of an accepted* $\mathsf{FiniteStateTransducer}^{\uparrow}$ *packet factors through one displayed finite* $\mathsf{StreamName}^{\uparrow}$ *input window, the transition trace ledger over that window, and the accumulated output ledger replayed along the same rows. Thus a consumer may read output only after displaying the rows* $Q, \Sigma, \Gamma, \delta, \lambda, q_0, w, \rho, o, H, C, P, N$ *and the finite route from* w *through* ρ *to* o*.*

Proof. Project the carrier of Definition 6184.1. The finite input window w is the only row through which an input read enters the packet, and it is already tied to the $\mathsf{StreamName}^{\uparrow}$ inspection surface.

Proceed by induction over the displayed input window. The empty window exposes the start row q_0, the empty transition trace, and the empty accumulated output row. For a nonempty window, the head symbol and current state select one displayed transition entry from δ and one displayed output entry from λ.

The replay rows C continue the same finite route through the tail of the window, while H, P, N transport, package, and name the displayed rows. Since the carrier has no host word function, infinite stream totality, hidden output coordinate, or quotient equality, every output-producing read factors through the displayed window, transition trace, and output ledger. □

Theorem 6184.5 (FiniteStateTransducerUp window composition obligation)**.** *Suppose two displayed finite input windows for an accepted* $\mathsf{FiniteStateTransducer}^{\uparrow}$ *packet are concatenated after the first window has replayed to its displayed terminal state and output ledger. The trace and accumulated output for the concatenated window are obtained only by composing the displayed* δ*,* λ*,* ρ*,* o*,* H*,* C*,* P*, and* N *rows for the first window with the corresponding displayed rows for the second window.*

Proof. Use induction on the first displayed window. In the empty case, the concatenated route is exactly the second displayed window route, starting at the displayed state and using its displayed transition trace and output ledger.

For a nonempty first window, project the head transition and head output rows selected by δ and λ. The induction hypothesis applies to the tail window followed by the second displayed window, because the Cont replay row carries the transported next-state row into that tail route.

The accumulated output ledger is therefore the displayed head output block followed by the output ledger supplied by the induction hypothesis. All transports and naming data are supplied by H, C, P, N, and no composition step can appeal to an unstored word function, infinite trace, quotient stream equality, or nondisplayed output totality. □

Falsifiable prediction: An accepted $\mathsf{FiniteStateTransducer}^{\uparrow}$ read is refuted if an output-producing consumer obtains an output row without displaying the finite state row, input symbol row, output symbol row, transition ledger, output ledger, input window, transition trace, hsame transport, Cont replay, Pkg provenance over FiniteAutomatonUp and StreamNameUp, and local naming row.

Independence witness: FiniteAutomaton and StreamName: neither is bijective to $\mathsf{FiniteStateTransducer}^{\uparrow}$ because the transducer packet jointly stores finite control, input-window replay, and a synchronized output ledger, while the automaton packet stores acceptance replay and the stream-name packet stores observation windows without an emitted output ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteStateTransducer}^{\uparrow}}$**.**

Constructive story: A FiniteStateTransducer$^{\uparrow}$ packet is generated from a finite state row, input-symbol row, output-symbol row, transition ledger, output ledger, start state, finite input window, transition trace, accumulated output ledger, componentwise hsame transport, displayed continuation replay, package provenance over FiniteAutomatonUp and StreamNameUp, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{FiniteStateTransducer}^{\uparrow}}$, seed)

Scope closed: Definition 6184.1, Definition 6184.2, Definition 6184.3, Definition 6184.4, and Definition 6184.5 seed the finite transducer packet over finite control, stream windows, transition replay, output replay, transport, continuation, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteStateTransducer}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close language quotient equivalence, minimization, nondeterministic transducers, total stream functions, infinite traces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, transition-ledger replay, output-ledger determinacy, transport, replay, provenance, and local naming for BEDC.Derived.FiniteStateTransducerUp.

Chapter 6185

A Concrete Naming Certificate for $\mathsf{FarkasLemma}^{\uparrow}$

Following Farkas and standard finite-dimensional convex analysis, the $\mathsf{FarkasLemma}^{\uparrow}$ packet names the finite alternative row by which a displayed linear-inequality system is certified either by a feasible point or by a nonnegative multiplier obstruction. It sits between the convex carrier surface of chapter 208, the finite face carrier of chapter 209, the nonnegative comparison row of chapter 2297, and the weak-duality ledger of chapter 235. The packet is a finite certificate surface; it does not assert a separation theorem, a choice principle, a selected optimizer, or full strong duality.

Definition 6185.1 (Farkas finite-alternative carrier)**.** *A* $\mathsf{FarkasLemma}^{\uparrow}$ *carrier over an ordered-field row consists of a finite* BHist *packet*

$$\mathcal{F} = (I, J, F, A, b, \rho, \nu, \eta, H, C, P, N)$$

with the following displayed rows.

1. *I and J are finite row and column carriers, and F is the scalar carrier equipped with the shared* $\mathsf{Field}^{\uparrow}$ *and* $\mathsf{TotalOrder}^{\uparrow}$ *source rows used by the surrounding convex and linear-programming chapters.*

2. *$A : I \times J \to F$ and $b : I \to F$ are the coefficient rows of a finite inequality system, with row evaluation*
$$\mathsf{Row}_i(x) := \sum_{j \in J} A_{ij} \cdot_F x_j.$$

3. *ρ is the feasibility row, asserting that a displayed point $x : J \to F$ satisfies $\mathsf{Row}_i(x) \preceq_F b_i$ for every $i \in I$.*

4. *ν is the nonnegative multiplier row, a finite vector $\lambda : I \to F$ with $\mathsf{NonNeg}_F(\lambda_i)$ for every $i \in I$, read through the* $\mathsf{PositiveCone}^{\uparrow}$ *comparison surface.*

5. *η is the infeasibility-alternative row, recording the classifier equations*
$$\sum_{i \in I} \lambda_i \cdot_F A_{ij} \sim_F 0_F \qquad (j \in J)$$
and the strict obstruction comparison
$$\sum_{i \in I} \lambda_i \cdot_F b_i \prec_F 0_F.$$

6. *H records componentwise* hsame *transport for the finite row and column carriers, scalar operations, coefficient rows, right-hand-side row, feasibility witness, multiplier witness, obstruction witness, continuation, provenance, and local naming row.*
7. *C records displayed* Cont *replay from finite coefficient rows through row evaluation, multiplier summation, positive-cone comparison, and the final infeasibility alternative.*
8. *P is the* Pkg *provenance over* $\mathsf{FarkasLemma}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Polytope}^{\uparrow}$, $\mathsf{PositiveCone}^{\uparrow}$, $\mathsf{LPDuality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FarkasLemma}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed finite rows. It has no coordinate for ambient matrix equality, quotient vector-space equality, topological separation, a selected optimizer, or full strong-duality existence.

Lean4 ✓ `BEDC.Derived.FarkasLemmaUp.FarkasLemmaTasteGate single carrier alignment`

Theorem 6185.2 (FarkasLemma obligation surface)**.** *For every accepted* $\mathsf{FarkasLemma}^{\uparrow}$ *carrier* $\mathcal{F} = (I, J, F, A, b, \rho, \nu, \eta, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FarkasLemma}^{\uparrow}}$ *surface consists exactly of the following five finite obligations:*

1. *carrier admission for the ordered-field source, finite coefficient rows, right-hand-side row, feasibility row, multiplier row, obstruction row, transport, replay, provenance, and local naming row;*
2. *classifier transport, requiring componentwise* hsame *in H to preserve every finite scalar endpoint, row-evaluation equation, nonnegative multiplier witness, and obstruction comparison;*
3. *positive-cone stability, requiring the multiplier row ν to be read only through the* $\mathsf{PositiveCone}^{\uparrow}$ *nonnegative classifier and the shared scalar order;*
4. *finite-alternative exactness, requiring exactly one displayed branch to be exported to downstream consumers: either the feasibility row ρ or the obstruction row η;*
5. *ledger non-escape, requiring P and N to export no topological separation theorem, selected optimizer, quotient vector-space equality, ambient matrix equality, or full strong-duality existence.*

No local NameCert obligation is stored outside these rows.

Proof. Project an accepted carrier through Definition 6185.1. Carrier admission is exactly the displayed finite row, column, scalar, coefficient, feasibility, multiplier, obstruction, transport, replay, provenance, and naming data.

Classifier transport is componentwise. The hsame row in H preserves each scalar endpoint and each finite row evaluation, while the displayed Cont replay in C repeats the same coefficient-to-obstruction ledger after classified replacement.

Positive-cone stability follows from the placement of ν. A multiplier coordinate is usable only after its nonnegative witness has been read through the $\mathsf{PositiveCone}^{\uparrow}$ comparison surface over the shared ordered-field row.

The finite-alternative exactness clause is the public branch boundary. A consumer may receive a feasible point row ρ or an obstruction row η, but both are finite BHist rows tied to the same coefficient packet and neither introduces a detached separation principle.

Finally, provenance is exhausted by P and N. Since the carrier has no coordinate for topological separation, selected optimizers, quotient equality, ambient matrix equality, or full strong duality, the local certificate cannot export any such object. □

Theorem 6185.3 (Farkas finite alternative)**.** *Let a* $\mathsf{FarkasLemma}^{\uparrow}$ *carrier over a finite ordered-field row be accepted as in Definition 6185.1. For the displayed coefficient system* A, b*, every downstream convex, polytope, positive-cone, or weak-duality consumer reads the alternative through a finite row:*

$$\rho \quad or \quad \eta.$$

In the first branch the consumer receives a carried feasible point with all inequalities $\mathsf{Row}_i(x) \preceq_F b_i$*. In the second branch it receives a carried nonnegative multiplier row whose finite row sums annihilate the left-hand coefficient rows and whose right-hand-side sum is strictly negative. The proof data are confined to finite row induction, scalar classifier transport, multiplier-ledger summation, and the displayed* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*

Proof. Proceed by induction on the finite row carrier I. In the empty row case the feasibility branch is immediate: the displayed coefficient ledger has no inequalities to check, and the multiplier row is the zero finite row.

For the row-extension step, separate the last displayed inequality from the tail packet. Apply the induction hypothesis to the tail coefficient system, obtaining either a tail feasible row or a tail obstruction row.

In the tail-feasible case, inspect the last row through the ordered-field comparison. If the displayed point satisfies the last inequality, reattach the row and obtain the full feasibility branch. If it fails, the finite scalar comparison supplies the one-row obstruction coefficient; the $\mathsf{PositiveCone}^{\uparrow}$ nonnegative row and the finite-sum ledger extend it to a multiplier packet over the enlarged carrier.

In the tail-obstruction case, extend the multiplier by zero on the last row. Finite-sum transport preserves the coefficient annihilation equations, and scalar order stability preserves the strict right-hand-side obstruction.

All transports are recorded in the visible ledger. The H row moves scalar endpoints and coefficient sums by hsame, the C row replays the row induction and multiplier summation, and P, N name the resulting branch as the same $\mathsf{FarkasLemma}^{\uparrow}$ alternative. Therefore the consumer receives only the finite feasibility row or the finite obstruction row and no detached separation or optimization principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{FarkasLemma}^{\uparrow}}$.

Constructive story: A $\mathsf{FarkasLemma}^{\uparrow}$ packet is generated from finite row and column carriers, an ordered-field scalar surface, coefficient and right-hand-side rows, a feasibility branch, a nonnegative multiplier branch, an obstruction comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FarkasLemma}^{\uparrow}}$, seed)

Scope closed: Definition 6185.1, Definition 6185.2, and Definition 6185.3 seed the finite-alternative certificate consumed by convex, polytope, cone, and weak-duality rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FarkasLemma}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close topological separation, selected optimizers, full strong duality, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires carrier admission, classifier transport, positive-cone stability, finite-alternative exactness, and ledger non-escape proofs for the same packet.

Chapter 6186

A Concrete Naming Certificate for $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ packet records the finite modulus data by which a located interval apartness claim becomes stable under smaller Real and Completion tolerances. It sits after the interval separation surface of chapter 4362, the located endpoint packet of chapter 1709, the Bishop located-real interface of chapter 6180, the dyadic tolerance ledger of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a positive gap together with the finite threshold after which all endpoint inspections preserve the apartness certificate; it is not a trichotomy decision, quotient real equality, a chosen endpoint representative, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6186.1 (Bishop interval-apartness modulus carrier)**.** *A* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, A, G, T, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *endpoint packet whose lower and upper endpoint rows are exposed before any modulus read.*
2. *A is the* $\mathsf{RealIntervalApartness}^{\uparrow}$ *row recording the displayed interval separation side and the endpoint rows on which the gap is read.*
3. *G is the positive apartness gap, stored as a* $\mathsf{DyadicRatCore}^{\uparrow}$ *witness compatible with the side selected by A.*
4. *T is the modulus threshold row: every endpoint-window request at a tolerance below T preserves the positive gap row G through the same interval-apartness side.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming exactly the endpoint inspections, gap checks, and threshold requests used by the packet.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those finite endpoint and threshold windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger tying the positive gap G and the threshold row T to one dyadic budget.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after I, A, G, T, S, R, D have been displayed.*

9. *H records componentwise* hsame *transport for the interval, apartness, gap, threshold, window, readback, dyadic ledger, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from an interval-apartness-modulus request through $I, A, G, T, S, R, D, E, H$.*
11. *P is the* Pkg *provenance over* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$, $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalApartnessModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host interval order, trichotomy, quotient equality of completed reals, selected endpoint representatives, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.BishopIntervalApartnessModulusUp.BishopIntervalApartnessModulusTasteGate`
`single carrier alignment`

Theorem 6186.2 (Bishop interval-apartness modulus NameCert obligations)**.** *For every accepted* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ *carrier $M = (I, A, G, T, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{BishopIntervalApartnessModulus}^{\uparrow}}$ *surface consists exactly of carrier admission, located endpoint admission, interval-apartness admission, positive-gap admission, modulus-threshold admission, finite StreamName window admission, RegSeqRat readback, DyadicRatCore tolerance compatibility, Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the displayed rows $I, A, G, T, S, R, D, E, H, C, P, N$.*

Proof. Project the carrier of Definition 6186.1. The analytic rows are the located interval I, the interval-apartness row A, the positive gap G, the threshold row T, the finite windows S, the regular readback R, the dyadic ledger D, and the terminal Real seal E.

The local certificate reads these rows in the displayed order. The threshold row may be consumed only after the apartness side and positive dyadic gap are visible, and the Real seal may be consumed only after the endpoint windows, regular readback, and dyadic ledger have been displayed.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for host interval order, trichotomy, quotient equality, selected representatives, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6186.3 (Bishop interval-apartness modulus located-real route)**.** *Every located-order or completion-facing read of an accepted* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ *carrier factors through*

$\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, Re

The route exports a positive interval-apartness gap together with a threshold for stable endpoint inspections. It exports no host trichotomy, quotient real equality, selected endpoint representative, open-cover compactness, or ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Proof. Begin with the obligations of Definition 6186.2. They force every consumer to expose the located interval row, the interval-apartness row, and the positive dyadic gap before it may read the modulus threshold.

Next read the finite endpoint windows and regular rational readback. The $\mathsf{BishopLocatedReal}^{\uparrow}$ interface is consumed only through this same endpoint, window, readback, and dyadic-tolerance route; it cannot replace the displayed gap by a host comparison of completed reals.

Finally route through the Real seal and the structural rows. Componentwise transport and displayed replay preserve the same packet, so the located-order or completion-facing consumer receives only the named finite modulus surface and none of the excluded host-order, quotient, selection, compactness, trichotomy, or completeness data. □

Theorem 6186.4 (Bishop interval-apartness modulus stability)**.** *Let* $M = (I, A, G, T, S, R, D, E, H, C, P, N)$ *and* $M' = (I', A', G', T', S', R', D', E', H', C', P', N')$ *be accepted* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ *carriers. If* H *records componentwise* hsame *transport from each displayed row of* M *to the corresponding row of* M'*, then the interval-apartness modulus of* M *transports to* M' *only through*

$$I, A, G, T, S, R, D, E, H, C, P, N.$$

In particular, the positive gap and the threshold for smaller endpoint windows remain attached to the same finite $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *route.*

Proof. Project the two accepted carriers. The modulus rows are G and T: G records the positive dyadic gap, while T records the threshold below which endpoint inspections preserve the same apartness side.

Componentwise transport through H carries the located interval, apartness side, gap, threshold, window schedule, readback, dyadic ledger, and Real seal to the corresponding rows of M'. The replay row C cannot change the order of these reads; it only replays the displayed route.

The NameCert obligations of Definition 6186.2 then repack the transported rows as the accepted modulus surface for M'. Because no host interval order, trichotomy, quotient equality, representative choice, or completeness coordinate is present, stability is exactly this row-local transport. □

Theorem 6186.5 (Bishop interval-apartness modulus Real-completion handoff)**.** *Every Real-and-Completion consumer of an accepted* $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ *carrier reaches the terminal* $\mathsf{Real}^{\uparrow}$ *seal only after the displayed located interval, interval-apartness row, positive gap, threshold, finite windows, regular rational readback, and dyadic tolerance ledger have been read. Thus the handoff supplies a finite modulus for interval apartness without importing quotient real equality, selected limits, trichotomy, or ambient completion.*

Proof. Read the carrier of Definition 6186.1. The terminal Real row is E, and it appears only downstream of I, A, G, T, S, R, D.

Apply the located-real route theorem. It keeps the completion-facing read inside the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{RealIntervalApartness}^{\uparrow}$, $\mathsf{BishopLocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows before any Real seal is consumed.

Apply the stability theorem to any transported packet used by the consumer. The positive gap and threshold remain attached to the same finite route under hsame transport and Cont replay. The provenance and local NameCert rows then name the same handoff, so no quotient equality, selected limit, trichotomy branch, or ambient completeness principle is exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopIntervalApartnessModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopIntervalApartnessModulus}^{\uparrow}$ packet is generated from a LocatedRealInterval endpoint packet, a RealIntervalApartness row, a positive DyadicRatCore gap, a modulus threshold, finite StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopIntervalApartnessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6186.1, Definition 6186.2, Definition 6186.3, Definition 6186.4, and Definition 6186.5 seed the finite interval-apartness modulus route from located endpoint rows and a positive dyadic gap through threshold, finite-window, regular-readback, Real seal, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{BishopIntervalApartnessModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host interval order, real trichotomy, quotient real equality, selected endpoint representatives, selected limits, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-apartness gap exposure, threshold stability, located-real routing, Real-completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopIntervalApartnes

Chapter 6187

A Concrete Naming Certificate for $\mathsf{BairePrefixCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion, and Brouwer together with Troelstra–van Dalen on spread and prefix semantics, the $\mathsf{BairePrefixCompletion}^{\uparrow}$ packet names the finite-prefix handoff from Baire streams to the completion-facing boundary. It sits after the finite prefix carrier of chapter 334, the common-prefix distance routes of chapter 1602 and chapter 3485, the finite-window readback of chapter 126, and before the limit-boundary packet of chapter 1884. The chapter records the completion handoff as displayed finite data, not as host stream equality, a selected branch, quotient equality of streams, countable choice, or ambient completeness.

Definition 6187.1 (Baire prefix completion carrier)**.** *A* $\mathsf{BairePrefixCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, M, S, E, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row, naming the inspected prefix cylinder and the prefix length visible to the completion request.*
2. *M is the* $\mathsf{BaireMetric}^{\uparrow}$ *or* $\mathsf{BaireUltrametric}^{\uparrow}$ *distance row, recording the common-prefix dyadic distance used to compare displayed cylinders before a boundary read.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, listing exactly the stream coordinates opened by the prefix request and the distance request.*
4. *E is the* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *boundary row, available only after B, M, S have been displayed.*
5. *H records componentwise* hsame *transport for the prefix, distance, window, boundary, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the same finite prefix windows through the completion-facing request.*
7. *P is the* Pkg *provenance over* $\mathsf{BairePrefixCompletion}^{\uparrow}$, $\mathsf{BaireSpace}^{\uparrow}$, $\mathsf{BaireMetric}^{\uparrow}$, $\mathsf{BaireUltrametric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CompleteBaireSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{BairePrefixCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, M, S, E, H, C, P, N. It has no coordinate for host stream equality, quotient stream equality, a selected infinite branch, countable choice, ambient completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.BairePrefixCompletionUp.BairePrefixCompletionTasteGate single carrier alignment`

Theorem 6187.2 (Baire prefix completion NameCert obligations)**.** *For every accepted* $\mathsf{BairePrefixCompletion}^{\uparrow}$ *carrier* $L = (B, M, S, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BairePrefixCompletion}^{\uparrow}}$ *surface consists exactly of five finite obligation rows:*

1. *carrier admission for the displayed Baire prefix row, prefix-distance row, finite StreamName window row, completion boundary row, transport, replay, provenance, and local naming;*
2. *prefix-window coverage, requiring every completion-facing read to expose the* $\mathsf{BaireSpace}^{\uparrow}$ *prefix row* B *and the* $\mathsf{StreamName}^{\uparrow}$ *window row* S *before any boundary row is inspected;*
3. *distance compatibility, requiring the* $\mathsf{BaireMetric}^{\uparrow}$ *or* $\mathsf{BaireUltrametric}^{\uparrow}$ *row* M *to compare only the displayed prefix cylinders and their finite common-prefix witnesses;*
4. *completion-boundary ordering, requiring the* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *row* E *to be downstream of* B, M, S *and stable under componentwise* hsame *transport and displayed* Cont *replay;*
5. *nonescape, excluding host stream equality, quotient stream equality, selected branches, countable choice, ambient completeness, Lean verification, and bridge checking.*

No BairePrefixCompletion obligation is stored outside these rows.

Proof. Project the accepted carrier of Definition 6187.1. The displayed tuple B, M, S, E, H, C, P, N is the whole public surface, so carrier admission is exactly the admission of those rows.

Prefix-window coverage is forced by B and S. The prefix row names the inspected cylinder, while the StreamName row lists every coordinate opened by that request.

Distance compatibility is read from M. Whether the consumer uses the metric or ultrametric sibling route, the distance row can compare only the prefix cylinders and finite common-prefix witnesses already displayed by B and S.

The boundary ordering is the role of E, H, C. The completion row is available only after the finite prefix, distance, and window rows, while transport and replay preserve that same route.

The provenance and local naming rows cite no selector, quotient, host equality, countable-choice operation, ambient completeness theorem, Lean credential, or bridge credential. Therefore the five rows exhaust the local NameCert surface. □

Theorem 6187.3 (Baire prefix completion finite-window exhaustion)**.** *Every* $\mathsf{CompleteBaireSpace}^{\uparrow}$ *consumer reading a limit boundary from an accepted* $\mathsf{BairePrefixCompletion}^{\uparrow}$ *carrier* $L = (B, M, S, E, H, C, P, N)$ *must first expose the* $\mathsf{BaireSpace}^{\uparrow}$ *finite prefix row* B*, the* $\mathsf{BaireMetric}^{\uparrow}$ *or* $\mathsf{BaireUltrametric}^{\uparrow}$ *distance row* M*, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row* S*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and the local* NameCert *row* N*. Thus every completion-facing read exposes finite prefix windows before the boundary row* E *can be consumed.*

Proof. Begin with a completion-facing read. By the carrier definition, the only rows upstream of the boundary row are the displayed prefix, distance, and finite-window rows, followed by the structural support rows.

Apply the NameCert obligations of Definition 6187.2. Prefix-window coverage requires B and S before any boundary read, and distance compatibility places M on the same finite prefix route.

Use the completion-boundary ordering clause for E, H, C. The boundary row cannot be consumed until the finite rows have been displayed, and transport or replay can only carry the same displayed route.

Finally P and N source and name this finite handoff. Since no excluded branch, quotient, choice, host equality, or ambient completeness coordinate exists in the carrier, the consumer receives finite prefix windows before any limit boundary. □

Falsifiable prediction: Every $\mathsf{BairePrefixCompletion}^{\uparrow}$ completion-facing read factors through a displayed $\mathsf{BaireSpace}^{\uparrow}$ prefix row, a $\mathsf{BaireMetric}^{\uparrow}$ or $\mathsf{BaireUltrametric}^{\uparrow}$ distance row, a finite $\mathsf{StreamName}^{\uparrow}$ window row, a $\mathsf{CompleteBaireSpace}^{\uparrow}$ boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: bairespace, bairemetric, baire-ultrametric, streamname, complete-bairespace: $\mathsf{BairePrefixCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite Baire prefixes, prefix-distance comparison, finite StreamName windows, completion boundary ordering, transport, continuation, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BairePrefixCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BairePrefixCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6187.1, Definition 6187.2, and Definition 6187.3 seed the finite-prefix completion handoff from Baire prefix cylinders through StreamName windows and prefix distance rows into the CompleteBaireSpace boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BairePrefixCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host stream equality, quotient stream equality, selected branches, countable choice, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window coverage, distance compatibility, completion-boundary ordering, finite-window exhaustion, and nonescape for BEDC.Derived.BairePrefixCompletionUp.

Constructive story: A $\mathsf{BairePrefixCompletion}^{\uparrow}$ packet is generated from finite Baire prefix rows, prefix-distance rows, StreamName finite windows, a CompleteBaireSpace boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6188

A Concrete Naming Certificate for $\mathsf{MinimalDfa}^{\uparrow}$

Following Hopcroft and Ullman, the $\mathsf{MinimalDfa}^{\uparrow}$ packet records the canonical deterministic finite automaton presentation obtained from the finite right-congruence rows of chapter 5729. It consumes accepted-language observations from chapter 351, displayed finite control from chapter 2364, and the minimization handoff supplied by chapter 5729. The packet names the minimal deterministic presentation as finite NameCert data; it does not identify arbitrary external automata, choose nondeterministic branches, import a host quotient of all words, or assert regular-expression equivalence.

Definition 6188.1 (MinimalDfaUp carrier)**.** *A* $\mathsf{MinimalDfa}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (\Sigma, L, E, C, W, Q, \delta, q_0, F, A, \alpha, \eta, \kappa, H, R, P, N)$$

with the following displayed rows:

1. Σ *is the finite alphabet row.*
2. L *is the accepted-word observation ledger supplied by a* $\mathsf{RegularLanguage}^{\uparrow}$ *packet.*
3. E *is the finite right-congruence table supplied by a* $\mathsf{MyhillNerode}^{\uparrow}$ *packet.*
4. C *is the suffix-stability replay row for* E*.*
5. W *is the distinguishability-witness ledger separating distinct rows of* E*.*
6. Q *is the finite state carrier whose states are the displayed* E*-classes.*
7. δ *is the deterministic transition row induced by appending one alphabet symbol and returning to the displayed class.*
8. q_0 *is the class of the empty word.*
9. F *is the accepting-class row determined by* L*.*
10. A *is the* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentation assembled from* $Q, \Sigma, \delta, q_0, F$ *and its finite replay rows.*
11. α *is the reachability ledger naming a representative word for each displayed state class.*
12. η *is the separation ledger naming a displayed suffix witness for every pair of distinct displayed state classes.*

13. *κ is the canonicality row binding accepted* $\mathsf{MinimalDfa}^{\uparrow}$ *packets with the same E, C, W rows to the same displayed automaton rows.*

14. *H records componentwise* hsame *transport for alphabet, language, congruence, suffix-stability, distinguishability, state, transition, start, accepting, automaton, reachability, separation, and canonicality rows.*

15. *R records displayed* Cont *replay from accepted-word queries through the right-congruence table and into the deterministic transition row.*

16. *P records* Pkg *provenance over* $\mathsf{RegularLanguage}^{\uparrow}$, $\mathsf{FiniteAutomaton}^{\uparrow}$, $\mathsf{MyhillNerode}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

17. *N is the local* $\mathsf{NameCert}_{\mathsf{MinimalDfa}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through the displayed rows $\Sigma, L, E, C, W, Q, \delta, q_0, F, A, \alpha, \eta, \kappa, H, R, P,$ The carrier has no coordinate for a hidden automaton presentation, nondeterministic recognizer, infinite language quotient, regular-expression proof, or external state renaming oracle.

Lean4 ✓ `BEDC.Derived.MinimalDfaUp.MinimalDfaTasteGate single carrier alignment`

Theorem 6188.2 (MinimalDfaUp NameCert obligations)**.** *For every accepted* $\mathsf{MinimalDfa}^{\uparrow}$ *carrier*

$$D = (\Sigma, L, E, C, W, Q, \delta, q_0, F, A, \alpha, \eta, \kappa, H, R, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MinimalDfa}^{\uparrow}}$ *surface consists exactly of finite alphabet exposure, accepted-word observation, right-congruence exposure, suffix-stability replay, distinguishability witnesses, quotient-state exposure, induced deterministic transition replay, start-class exposure, accepting-class exposure, finite-automaton presentation, reachability representatives, pairwise separation witnesses, canonicality transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6188.1. The rows Σ, L, E, C, W expose the finite observation table inherited from the regular-language and Myhill–Nerode packets.

The rows Q, δ, q_0, F, A repackage that table as the deterministic finite automaton presentation consumed by $\mathsf{FiniteAutomaton}^{\uparrow}$. Reachability is recorded by α, and pairwise separation is recorded by η; these rows are the finite witnesses that no displayed state is unreachable and no two displayed states can be merged by the accepted-word classifier.

The rows κ, H, R, P, N keep canonicality, transport, replay, provenance, and local naming attached to the same finite packet. Since the carrier has no hidden presentation, nondeterministic branch, host quotient, regular-expression proof, or state-renaming oracle coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 6188.3 (MinimalDfaUp Myhill–Nerode canonicality)**.** *Let D and D' be accepted* $\mathsf{MinimalDfa}^{\uparrow}$ *packets over the same displayed alphabet, accepted-word ledger, right-congruence table, suffix-stability row, and distinguishability-witness ledger. If their* $\mathsf{MyhillNerode}^{\uparrow}$ *rows agree componentwise, then their finite state carriers, induced transition rows, start classes, accepting-class rows, reachability ledgers, separation ledgers, and* $\mathsf{FiniteAutomaton}^{\uparrow}$ *presentations agree through the canonicality rows of the two packets. Thus a* $\mathsf{RegularLanguage}^{\uparrow}$ *consumer reading through the Myhill–Nerode minimization handoff can obtain only the displayed minimal deterministic presentation named by the* $\mathsf{MinimalDfa}^{\uparrow}$ *packet.*

Proof. Use Definition 6188.2 to project both accepted packets to their displayed right-congruence data. Agreement of the $\mathsf{MyhillNerode}^{\uparrow}$ rows identifies the same finite class table, the same suffix-stability replay, and the same distinguishability witnesses.

The state carrier Q is exactly the displayed class carrier of that table. The transition row δ is forced by appending one displayed alphabet symbol and returning to the corresponding class, while q_0 and F are read from the empty-word class and the accepted-word ledger.

Reachability and separation are finite ledgers over the same displayed classes. The canonicality rows κ and κ' therefore transport the assembled $\mathsf{FiniteAutomaton}^{\uparrow}$ presentations componentwise. No further automaton can be selected because hidden presentations, state-renaming oracles, nondeterministic branches, and host quotients are absent from the carrier. □

Falsifiable prediction: An accepted $\mathsf{MinimalDfa}^{\uparrow}$ read is refuted if a minimization consumer obtains a deterministic automaton row without displaying the finite alphabet, accepted-word ledger, right-congruence table, suffix-stability row, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting-class row, reachability ledger, separation ledger, canonicality row, transport rows, replay rows, provenance row, and local naming row.

Independence witness: RegularLanguage, FiniteAutomaton, and MyhillNerode: none is bijective to $\mathsf{MinimalDfa}^{\uparrow}$. The regular-language packet records accepted-word observation without a canonical finite presentation, the finite-automaton packet records one presentation without the Myhill–Nerode separation ledger, and the Myhill–Nerode packet records the minimization handoff without owning the downstream minimal deterministic automaton NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{MinimalDfa}^{\uparrow}}$.

Constructive story: A $\mathsf{MinimalDfa}^{\uparrow}$ packet is generated from a finite alphabet row, accepted-word ledger, right-congruence table, suffix-stability replay, distinguishability witnesses, quotient-state row, induced transition row, start class, accepting-class row, finite automaton presentation, reachability representatives, pairwise separation witnesses, canonicality transport, componentwise hsame transport, displayed continuation replay, package provenance over RegularLanguageUp, FiniteAutomatonUp, and MyhillNerodeUp, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MinimalDfa}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6188.1, Definition 6188.2, and Definition 6188.3 seed the minimal deterministic automaton packet over finite right-congruence data, deterministic replay, reachability, separation, canonicality, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MinimalDfa}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary automaton equivalence, nondeterministic minimization, regular-expression conversion, external state-renaming quotients, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, right-congruence replay, distinguishability coverage, finite-automaton assembly, reachability, separation, canonicality, transport, replay, provenance, and local naming for BEDC.Derived.MinimalDfaUp.

Chapter 6189

A Concrete Naming Certificate for $\mathsf{BishopLocatedInterval}^{\uparrow}$

Bishop and Bridges-style located interval practice treats an interval as a finite endpoint and locatedness interface rather than as an ambient completeness principle. The $\mathsf{BishopLocatedInterval}^{\uparrow}$ packet records that interface for the L10 Real and Completion route: it reads the dyadic bracket ledger of chapter 291, the finite observation discipline of chapter 126, the regular rational handoff of chapter 301, the terminal real-name seal of chapter 38, and the located-real endpoint surface of **??**. It does not assert Heine–Borel compactness, selected finite subcovers, quotient stream equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6189.1 (Bishop located interval carrier)**.** *A* $\mathsf{BishopLocatedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, D, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet displayed before any interval-facing request is accepted.*
2. *U is the upper endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet displayed in the same history as L.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *bracket ledger, carrying the lower and upper dyadic cells, tolerance budget, visible slack, and comparison rows used to inspect L and U.*
4. *Q is the locatedness witness row: for each accepted finite probe it displays which endpoint bracket, dyadic subcell, or visible gap row answers the probe.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite observation row opened at the dyadic indices recorded in D and Q.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the same StreamName windows and dyadic bracket ledger.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal consumed only after the endpoint rows, bracket ledger, locatedness witness, finite observation row, and regular rational handoff have been displayed.*
8. *H records componentwise* hsame *transport for the endpoint rows, dyadic bracket ledger, locatedness witness, StreamName observations, RegSeqRat handoff, Real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for finite located-interval requests through L, U, D, Q, S, R, E, H.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopLocatedInterval}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}$ *row.*

The classifier reads only these rows. In particular, it has no coordinate for selected subcovers, global compactness, host real equality, quotient stream equality, countable choice, or a completeness assertion for all $\mathsf{Real}^{\uparrow}$ *Cauchy data.*

Lean4 ✓ `BEDC.Derived.BishopLocatedIntervalUp.BishopLocatedIntervalTasteGate single carrier alignment`

Theorem 6189.2 (Bishop located interval NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedInterval}^{\uparrow}$ *carrier* $I = (L, U, D, Q, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}$ *surface is exhausted by the following obligations:*

1. *endpoint admission: the lower and upper endpoint rows L, U are displayed* $\mathsf{LocatedReal}^{\uparrow}$ *packets in the same finite history;*
2. *interval order and bracket visibility: the dyadic ledger D exposes the endpoint order comparison, tolerance budget, bracket cells, and slack rows before an interval request can read them;*
3. *locatedness: the witness row Q answers each accepted finite probe by a displayed endpoint bracket, subcell, or visible gap row;*
4. *dyadic refinement stability: refining the accepted dyadic budget transports D, Q, S, R componentwise through the same* hsame *and* Cont *rows;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal nonescape: any regular rational or real-facing read must pass through S, R, E, H, C, P, N and cannot export quotient stream equality, a selected limit, countable choice, Heine–Borel data, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6189.1. The endpoint rows are L and U, and the only order or bracket data visible to an interval request is the dyadic ledger D together with the locatedness witness Q.

The finite observation and completion route is forced through S, R, E. The StreamName row S opens only the requested finite windows, the RegSeqRat row R reads back the regular rational handoff from those windows, and the terminal Real row E is sealed behind the same displayed route.

Transport and replay remain row-local. The rows H, C, P, N carry the endpoint, bracket, locatedness, observation, handoff, and seal data componentwise, so a refinement can only repack the same finite NameCert surface. Since the carrier contains no row for selected subcovers, quotient stream equality, countable choice, chosen limits, or ambient completeness, no such principle can be exported by the certificate. □

Theorem 6189.3 (Bishop located interval endpoint admission)**.** *For every accepted* $\mathsf{BishopLocatedInterval}^{\uparrow}$ *carrier* $I = (L, U, D, Q, S, R, E, H, C, P, N)$*, the lower endpoint row L and upper endpoint row U are admitted only as the displayed* $\mathsf{LocatedReal}^{\uparrow}$ *rows in the same finite* BHist*. Any endpoint-facing read of I must pass through the same* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}$ *surface recorded by H, C, P, N.*

Proof. Project the carrier of Definition 6189.1. Its first two rows are precisely L and U, and the defining display already types both rows as $\mathsf{LocatedReal}^{\uparrow}$ packets in the shared finite history. The endpoint surface is therefore the located-real surface of **??**, not a host real or quotient stream row.

The remaining endpoint-facing access is controlled by the displayed certificate rows. The dyadic brackets are read through chapter 291; the finite observation windows are those of chapter 126; the regular rational handoff is the one named in chapter 301; and the terminal seal is the $\mathsf{Real}^{\uparrow}$ route of chapter 38. Since H, C, P, N are the only transport, replay, provenance, and local naming rows, endpoint admission is exhausted by these finite rows. □

Theorem 6189.4 (Bishop located interval locatedness witness)**.** *For every accepted* $\mathsf{BishopLocatedInterval}^{\uparrow}$ *carrier* $I = (L, U, D, Q, S, R, E, H, C, P, N)$*, each finite locatedness probe answered by* Q *factors through a displayed endpoint bracket, dyadic subcell, or visible gap row. No such answer can invoke selected subcovers, countable choice, quotient stream equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the locatedness coordinate Q. By Definition 6189.1, Q answers only accepted finite probes, and each answer names one of the visible rows already displayed by the endpoint brackets, the $\mathsf{DyadicRatCore}^{\uparrow}$ subcell ledger of chapter 291, or the visible gap rows carried in the same ledger.

The finite row-exhaustion argument then excludes the ambient principles. Stream observations may only use the finite windows of chapter 126; regular readback may only use chapter 301; endpoint reads remain in **??**; and any terminal real name must enter through chapter 38. The tuple has no coordinate for subcover selection, countable choice, quotient equality, or host completeness, so a Q-answer cannot factor through any of them. □

Theorem 6189.5 (Bishop located interval Real seal non-escape)**.** *For every accepted* $\mathsf{BishopLocatedInterval}^{\uparrow}$ *carrier* $I = (L, U, D, Q, S, R, E, H, C, P, N)$*, any* $\mathsf{Real}^{\uparrow}$ *seal read from* E *is downstream of the displayed endpoint rows* L, U*, dyadic ledger* D*, locatedness witness* Q*, finite observation row* S*, regular rational handoff* R*, and the transport, replay, provenance, and local naming rows* H, C, P, N*.*

Proof. Project the seal coordinate E. The carrier definition places E after L, U, D, Q, S, R, so a real-facing read cannot reach the seal before the endpoint, dyadic, locatedness, stream, and regular-rational rows have been displayed. This is exactly the finite L10 route through **??**, chapter 291, chapter 126, chapter 301, and chapter 38.

The certificate rows H, C, P, N preserve that order. Componentwise hsame transport, displayed Cont replay, provenance, and local naming may repack the finite route, but they add no separate seal source. Thus every $\mathsf{Real}^{\uparrow}$ read from E remains downstream of the displayed rows and cannot escape to quotient stream equality, selected limits, countable choice, Heine–Borel compactness, or ambient real completeness. □

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{BishopLocatedInterval}^{\uparrow}$ packet is generated from lower and upper LocatedReal endpoint rows, a DyadicRatCore bracket ledger, a locatedness witness, StreamName finite observation, RegSeqRat handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6189.1, Definition 6189.2, Definition 6189.3, Definition 6189.4, and Definition 6189.5 bind the finite located-interval carrier, endpoint admission, locatedness witness, and Real seal non-escape obligations over $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Heine–Borel compactness, selected finite subcovers, quotient stream equality, countable choice, selected limits, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Scoped closure requires checked finite-row coverage for endpoint admission, locatedness factoring, RegSeqRat handoff, Real seal non-escape, transport, replay, provenance, and local naming over BEDC.Derived.BishopLocatedIntervalUp.

Chapter 6190

A Concrete Naming Certificate for KleeneKreiselContinuousFunctional$^{\uparrow}$

Following Kleene and Kreisel on continuous functionals over choice-sequence spaces, the KleeneKreiselContinuousFun packet records a higher-type finite-window interface for functionals read from countable names. It sits over the finite observation discipline of chapter 126, the binary prefix surface of chapter 1475, the bounded modulus handoff of chapter 1525, the neighborhood table of chapter 2336, and the domain-facing readback of chapter 6099. The packet is grounded in finite BEDC rows and displayed request ledgers, not in host function equality, an ambient exponential object, or a completed space of all functionals.

Definition 6190.1 (KleeneKreiselContinuousFunctional carrier)**.** *A* KleeneKreiselContinuousFunctional$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, C, F, Nb, Sc, H, T, P, N)$$

with the following displayed rows:

1. *S is the* StreamName$^{\uparrow}$ *source row, exposing the finite request windows through which countable names are inspected.*
2. *C is the* CantorSpace$^{\uparrow}$ *binary prefix row used when the functional read is restricted to a binary choice-sequence window.*
3. *F is the* FanFunctional$^{\uparrow}$ *modulus row, displaying the bounded prefix length needed before an output row may be read on compact binary windows.*
4. *Nb is the* NeighborhoodFunction$^{\uparrow}$ *table row, recording finite input-prefix requests and the output rows released by those requests.*
5. *Sc is the* ScottContinuous$^{\uparrow}$ *readback row, recording the directed-observation and Scott-open replay surface through which the same finite table is read as a continuity packet.*
6. *H records componentwise* hsame *transport for the stream, Cantor, fan, neighborhood, Scott-continuity, replay, provenance, and naming rows.*
7. *T records displayed* Cont *replay from every finite name request through S, C, F, Nb, Sc, H before an output row is exposed.*
8. *P is the* Pkg *provenance over* KleeneKreiselContinuousFunctional$^{\uparrow}$, StreamName$^{\uparrow}$, CantorSpace$^{\uparrow}$, FanFunctional$^{\uparrow}$, NeighborhoodFunction$^{\uparrow}$, ScottContinuous$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. N is the local $\mathsf{NameCert}_{\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed rows $S, C, F, Nb, Sc, H, T, P, N$. It has no coordinate for host equality of higher-type functions, an arbitrary map from all streams to all values, countable choice, a selected infinite path, or a semantic function-space quotient.

Lean4 (definition) `BEDC.Derived.KleeneKreiselContinuousFunctionalUp.taste gate`

Theorem 6190.2 (KleeneKreiselContinuousFunctional NameCert obligations)**.** *For every accepted* $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ *carrier*

$$K = (S, C, F, Nb, Sc, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}}$ *surface consists exactly of carrier admission, finite-window classifier exposure, neighborhood-function transport through Nb, fan-functional modulus handoff through F, Scott-continuity readback through Sc, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, local naming through N, and non-escape from those displayed rows. Every accepted read must expose the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{CantorSpace}^{\uparrow}$ *request rows before it may consume the neighborhood table, fan modulus, or Scott-continuity readback.*

Proof. Project the carrier of Definition 6190.1. The source name windows are S, the binary prefix face is C, the compact modulus face is F, the finite neighborhood table is Nb, and the domain-continuity readback is Sc.

The functional read is finite at each public use. The neighborhood row Nb lists the input-prefix requests that release an output row, while the fan row F records the bounded prefix length needed on the Cantor-space face. These rows are classifier data in the BEDC packet, not host function equality or an order on completed streams.

The Scott row Sc reads the same table through directed-observation and Scott-open replay. Componentwise transport through H and displayed continuation replay through T preserve the rows already displayed; they do not create a second functional source.

Finally P and N package and name the same finite route. Since the carrier contains no host exponential, no stream quotient, no selected infinite path, and no countable-choice operator, the listed rows exhaust the local NameCert obligations. □

Theorem 6190.3 (KleeneKreiselContinuousFunctional neighborhood handoff)**.** *Every accepted* $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ *read factors through the displayed* $\mathsf{NeighborhoodFunction}^{\uparrow}$ *row and the* $\mathsf{StreamName}^{\uparrow}$ *finite request rows before any output read is available. More precisely, for a carrier*

$$K = (S, C, F, Nb, Sc, H, T, P, N),$$

any public output row must first display the StreamName request window in S, the corresponding neighborhood-table entry in Nb, and the continuation replay row in T. On binary choice-sequence reads the same handoff may additionally use the $\mathsf{CantorSpace}^{\uparrow}$ *row C and the* $\mathsf{FanFunctional}^{\uparrow}$ *modulus row F; on domain-facing reads it may use the* $\mathsf{ScottContinuous}^{\uparrow}$ *row Sc. No output row is admitted from a host functional, a hidden infinite stream, or a detached equality proof.*

Proof. Start from an accepted public output read. By Definition 6190.1, the only rows that can expose an output are the finite request row S, the neighborhood table Nb, and the displayed continuation row T.

Use Definition 6190.2. It forces every accepted read through carrier admission, finite-window classifier exposure, neighborhood transport, continuation replay, provenance, and local naming before the output row is treated as public data.

For the binary face, the Cantor row C supplies the finite prefix window and the fan row F supplies the bounded modulus handoff. For the domain face, the Scott row Sc replays the same table as a continuity read. In both cases the consumer still reaches the output through S, Nb, T rather than through an ambient higher-type map.

Rows H, P, N transport, source, and name this handoff. Because host function equality, hidden infinite streams, selected paths, countable choice, and semantic function-space quotients are absent from K, no other output source remains. □

Falsifiable prediction: An accepted $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ read is refuted if a consumer obtains an output row without displaying the StreamName request window, NeighborhoodFunction table entry, continuation replay, componentwise hsame transport, Pkg provenance over StreamNameUp, CantorSpaceUp, FanFunctionalUp, NeighborhoodFunctionUp, ScottContinuousUp, and local naming row.

Independence witness: streamname, cantorspace, fanfunctional, neighborhood-function, scott-continuous: $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores source-name requests, binary prefix restriction, compact modulus handoff, a neighborhood table, Scott-continuity readback, transport, replay, provenance, and local naming in one higher-type finite-window packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}}$.

Constructive story: A $\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}$ packet is generated from StreamName request windows, CantorSpace prefix rows, FanFunctional modulus rows, NeighborhoodFunction table rows, ScottContinuous readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6190.1, Definition 6190.2, and Definition 6190.3 seed the finite-window interface for continuous higher-type functional reads over named streams.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KleeneKreiselContinuousFunctional}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function spaces, quotient equality of functionals, selected infinite paths, countable choice, total stream maps, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, neighborhood transport, fan modulus handoff, Scott-continuity readback, transport, replay, provenance, local naming, and a concrete Lean target for BEDC.Derived.KleeneKreiselContinuousF

Chapter 6191

A Concrete Naming Certificate for $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$

Following Picard, Lindelof, and Bishop–Bridges, the $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ packet records the local ODE approximation surface that lies between the finite $\mathsf{PicardIteration}^{\uparrow}$ ledger of chapter 2359 and the local-flow endpoint of chapter 1899. It reads the ODE source of chapter 193, the Lipschitz vector-field surface of chapter 410, and the completion-facing L10 route through chapter 126, chapter 301, chapter 291, and chapter 38. The packet names a local finite iteration handoff; it is not a host solution curve, a selected fixed point, a quotient path space, or an ambient $\mathsf{Real}^{\uparrow}$ completeness principle.

Definition 6191.1 (Picard–Lindelof local iteration carrier)**.** *A* $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (O, F, I, M, S, R, E, H, C, P_0, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *local time-state row, naming the initial cell, local interval, and state-window boundary used by all later reads.*
2. *F is the* $\mathsf{LipschitzMap}^{\uparrow}$ *vector-field row on the same local window, including the displayed graph reads and Lipschitz budget available to the Picard operator.*
3. *I is the* $\mathsf{PicardIteration}^{\uparrow}$ *finite iteration ledger, recording the inspected approximants and their one-step reads through F.*
4. *M is the* $\mathsf{CompleteMetric}^{\uparrow}$ *approximant-space row, exposing the Cauchy-modulus and late-comparison data needed before a completion-facing read is allowed.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row naming the visible approximation windows without replacing them by a host sequence object.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row for the same windows, local radius budgets, and regular rational readbacks.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary reached only after S and R have been displayed.*
8. *H records componentwise* hsame *transport for the ODE source, Lipschitz ledger, Picard iteration ledger, complete-metric row, stream windows, regular rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through*

$$O, F, I, M, S, R, E \longmapsto H, P_0, N.$$

10. *P_0 is the* Pkg *provenance over* $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{LipschitzMap}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalIteration}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, F, I, M, S, R, E, H, C, P_0, N$. It has no coordinate for a global solution, a maximal continuation interval, host function equality, selected limits, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.PicardLindelofLocalIterationUp.taste gate`

Theorem 6191.2 (Picard–Lindelof local iteration obligations)**.** *For every accepted* $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ *carrier*

$$P = (O, F, I, M, S, R, E, H, C, P_0, N),$$

the local $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalIteration}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Ode}^{\uparrow}$ *row,* $\mathsf{LipschitzMap}^{\uparrow}$ *row,* $\mathsf{PicardIteration}^{\uparrow}$ *ledger,* $\mathsf{CompleteMetric}^{\uparrow}$ *approximant-space row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal boundary, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the accepted carrier of Definition 6191.1. The ODE and vector-field rows are O and F, so every approximation read first fixes the initial cell, local window, and displayed Lipschitz graph data.

The finite iteration and completion-facing rows are I and M. The row I records only the Picard approximants inspected in the packet, while M exposes the complete-metric Cauchy and late-comparison data needed before a completion-facing consumer can ask for a terminal read.

The L10 readback rows are S, R, E. They route the visible approximants through finite stream windows, regular rational readbacks, and the real seal boundary of the same packet, so no host sequence, selected limit, quotient equality, or ambient completeness coordinate is introduced.

Finally H, C, P_0, N transport, replay, source, and name exactly the same rows. These structural rows cannot add a global solution, a maximal continuation interval, or host function equality, so the listed rows exhaust the local obligation surface. □

Theorem 6191.3 (Picard–Lindelof RegSeqRat handoff)**.** *Every accepted* $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ *carrier hands its visible Picard approximants to the L10 completion route only through*

$$\mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently the $\mathsf{RegSeqRat}^{\uparrow}$ *row R is downstream of the finite iteration ledger I, the complete-metric row M, and the stream-name row S, and upstream of the real-seal boundary E. The handoff exports no selected local solution, quotient stream equality, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite iteration ledger I. It is the only carrier row that stores Picard approximants, and every approximant in it is read from the same ODE row O and Lipschitz row F.

The complete-metric row M is downstream of that ledger. It records the Cauchy-modulus and late-comparison surface for the displayed approximants, not a selected endpoint or an ambient completeness theorem.

Now read the L10 rows S, R, E. The stream-name row S names the finite approximation windows, the regular rational row R reads those windows with their local budgets, and the real-seal boundary E is visible only after both rows have been displayed.

The structural rows H, C, P_0, N preserve and name the same route. Since no coordinate stores a selected local solution, quotient stream equality, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness, every RegSeqRat handoff remains inside the displayed packet. □

Falsifiable prediction: Every $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ read exposes one Ode local time-state row, one LipschitzMap vector-field row, one PicardIteration finite ledger, one CompleteMetric approximant-space row, one StreamName window row, one RegSeqRat readback row, one RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: picard-lindelof, picard-iteration, picard-lindelof-local-flow, streamname, regseqrat, dyadicratcore, real: $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the ODE local window, Lipschitz vector-field ledger, finite Picard iteration ledger, complete-metric approximation row, L10 stream and regular-rational readback rows, real seal boundary, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardLindelofLocalIteration}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardLindelofLocalIteration}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6191.1, Definition 6191.2, and Definition 6191.3 seed the finite local Picard iteration packet between ODE Lipschitz data, complete-metric approximation, L10 readback, and the real seal boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardLindelofLocalIteration}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim global ODE existence, maximal interval continuation, selected local solutions, quotient path or stream equality, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, ODE local-window compatibility, Lipschitz vector-field stability, PicardIteration ledger visibility, CompleteMetric approximation routing, StreamName and RegSeqRat readback, Real seal discipline, transport, replay, provenance, and local naming for BEDC.Derived.PicardLindelofLocalIterationUp.

Constructive story: A $\mathsf{PicardLindelofLocalIteration}^{\uparrow}$ packet is generated from Ode, LipschitzMap, PicardIteration, CompleteMetric, StreamName, RegSeqRat, Real, componentwise hsame, displayed Cont, Pkg provenance, and local NameCert rows.

Chapter 6192

A Concrete Naming Certificate for $\mathsf{LocatedDyadicReal}^{\uparrow}$

Following Bishop and Bridges, a located real presentation is useful only when every comparison request remains tied to explicit rational tolerances. The $\mathsf{LocatedDyadicReal}^{\uparrow}$ packet realizes that presentation over the dyadic core of chapter 291, the finite stream windows of chapter 126, the regular rational readback route of chapter 301, the located-real classifier of **??**, and the terminal Real seal of chapter 38. Its certificate is a finite bridge from dyadic tolerance data to located comparison and Real-seal readback, not a quotient stream construction or a selected limit.

Definition 6192.1 (Located dyadic real carrier)**.** *A* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, S, Q, L_c, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the comparison radius and the halving ledger used by finite approximation reads.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row, exposing only the finite windows requested at the displayed dyadic tolerance.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, converting each finite stream window into the regular rational comparison data consumed by the located surface.*
4. *L_c is the located comparison row, recording the finite alternative required by the* $\mathsf{LocatedReal}^{\uparrow}$ *classifier at the displayed dyadic tolerance.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the dyadic tolerance, stream window, regular readback, and located comparison rows have been replayed.*
6. *H records componentwise* hsame *transport for D, S, Q, L_c, R.*
7. *C records displayed* Cont *replay from tolerance requests through stream windows, regular readbacks, located comparisons, the Real seal, and structural rows.*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedDyadicReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.LocatedDyadicRealUp.taste gate`

Theorem 6192.2 (Located dyadic real NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *carrier*

$$L = (D, S, Q, L_c, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedDyadicReal}^{\uparrow}}$ *surface consists exactly of finite rows for dyadic tolerance admission, StreamName window exposure, RegSeqRat readback, located comparison, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming. No obligation row may erase* D*, infer located comparison without* S *and* Q*, identify quotient streams, select a limit, or export ambient Real completeness.*

Proof. Project the carrier of Definition 6192.1. The dyadic tolerance row is public before any stream window or regular rational readback is requested.

The window and readback rows are then forced in order. A located comparison may inspect the finite StreamName window and the RegSeqRat readback attached to the same displayed tolerance, but it cannot replace them by a quotient stream or a chosen limit.

The terminal Real seal is downstream of that located comparison. It names the sealed endpoint reached by the finite route, not a free ambient completeness principle.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since these structural rows add no selector, quotient, or host-real equality coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 6192.3 (Located dyadic real terminal seal)**.** *Every Real-facing consumer of an accepted* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *packet factors through the finite route*

$$D \to S \to Q \to L_c \to R \to H, C, P, N.$$

The consumer may narrow a dyadic tolerance request and reread the corresponding finite windows and regular rational rows, but it cannot bypass the located comparison row, identify quotient streams, select a limit, use countable choice, invoke trichotomy, or export ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the obligations of Definition 6192.2. They expose the tolerance, window, readback, located comparison, and Real seal rows in a fixed finite order.

A narrowed tolerance request remains a request in the same D ledger. Replaying it may refine the finite window S and the regular readback Q, but the located comparison row L_c is still the only classifier-facing handoff.

The Real seal R is therefore terminal only for this displayed route. It records that the located dyadic presentation has reached the Real surface, while leaving quotient equality, selected limits, choice, trichotomy, and ambient completeness outside the packet.

The structural rows preserve this boundary under hsame transport, Cont replay, provenance, and local naming. Hence every Real-facing read factors through the stated finite route. □

Theorem 6192.4 (Located dyadic real comparison refinement)**.** *Let* $L = (D, S, Q, L_c, R, H, C, P, N)$ *be an accepted* $\mathsf{LocatedDyadicReal}^{\uparrow}$ *packet. If the packet is reread at a narrower positive* $\mathsf{DyadicRatCore}^{\uparrow}$

tolerance recorded by the same D ledger, then it exposes a refined $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *row through the same route*

$$D, S, Q, L_c, R, H, C, P, N.$$

The located comparison row L_c remains the only classifier-facing handoff before the terminal $\mathsf{Real}^{\uparrow}$ *row is read.*

Proof. Project the dyadic tolerance row D and its halving ledger. A narrower positive tolerance is still a read of the same displayed ledger, so it cannot introduce an external comparison oracle or an ambient real-order principle.

Replay S and Q through Cont at the narrower request. The replay exposes the refined finite StreamName window and the corresponding RegSeqRat readback before any terminal Real row is public.

Transport the located comparison row componentwise by hsame. The row L_c therefore remains attached to the refined window and readback, and no other classifier-facing handoff is available in the carrier.

Finally apply Definition 6192.3. Its terminal route excludes quotient streams, selected limits, trichotomy, and ambient $\mathsf{Real}^{\uparrow}$ completeness before R is read. □

Falsifiable prediction: A $\mathsf{LocatedDyadicReal}^{\uparrow}$ consumer that exports ambient Real completeness, a selected limit, or quotient stream equality after deleting D, S, Q, or L_c refutes the carrier of Definition 6192.1.

Independence witness: Nearest siblings: dyadicratcore, streamname, regseqrat, locatedreal, and real. $\mathsf{LocatedDyadicReal}^{\uparrow}$ is not bijective to any one of them because it jointly carries dyadic tolerance, finite stream windows, regular rational readback, located comparison, terminal Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicReal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDyadicReal}^{\uparrow}$ packet is generated from positive DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readbacks, located comparison rows, terminal Real seal rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedDyadicReal}^{\uparrow}}$, seed)

Scope closed: Definition 6192.1, Definition 6192.2, and Definition 6192.3 seed the finite located dyadic bridge from dyadic tolerances to terminal Real reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedDyadicReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exactness, StreamName window replay, RegSeqRat readback, located comparison nonescape, terminal Real sealing, classifier transport, and local naming for BEDC.Derived.LocatedDyadicRealUp.

Chapter 6193

A Concrete Naming Certificate for $\mathsf{MomentProblem}^{\uparrow}$

Following Hausdorff, Stieltjes, and Hamburger on moment problems, the $\mathsf{MomentProblem}^{\uparrow}$ packet records the finite BEDC surface by which a truncated moment table may be inspected before any consumer asks for a representing measure. It sits after the rational and real readback rows of chapter 37 and chapter 38, the finite matrix rows of chapter 49, and the measure-facing boundary of chapter 92 and chapter 186. The certificate names only finite Hankel windows, positivity checks, transport, replay, provenance, and local naming; it does not export an ambient moment problem, a representing measure theorem, determinacy, uniqueness, or a host analytic solution principle.

Definition 6193.1 (MomentProblem carrier)**.** *A* $\mathsf{MomentProblem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, A, H_0, H_1, W, P_s, D, H, C, Q, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Rat}^{\uparrow}$ *coefficient ledger used to encode the requested finite moment entries and comparison tolerances.*
2. *A is the* $\mathsf{Real}^{\uparrow}$ *readback row that seals the inspected moment values as BEDC real-name data.*
3. *H_0 and H_1 are the first two finite* $\mathsf{Mat}^{\uparrow}$ *Hankel table rows, built from the displayed moment window.*
4. *W is the finite-window schedule naming which initial segment of the moment sequence and which principal Hankel minors are inspected.*
5. *P_s records the finite positivity certificates for the displayed Hankel windows, including the chosen rows, columns, and rational tolerance budget.*
6. *D is the optional* $\mathsf{Distribution}^{\uparrow}$ *boundary row available only as downstream probabilistic syntax; it is not a representing-measure object.*
7. *H records componentwise* hsame *transport for the rational ledger, real readbacks, Hankel matrices, finite-window schedule, positivity rows, distribution boundary, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite moment-window request through $R, A, H_0, H_1, W, P_s, D, H$.*
9. *Q is the* Pkg *provenance over* $\mathsf{MomentProblem}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{MomentProblem}^\uparrow}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for an ambient measure space, selected representing measure, moment determinacy theorem, infinite Hankel operator, or host analytic solution.

Lean4 ✓ `BEDC.Derived.MomentProblemUp.TasteGate.MomentProblemTasteGate single carrier alignment`

Theorem 6193.2 (MomentProblem NameCert obligations)**.** *For every accepted* $\mathsf{MomentProblem}^\uparrow$ *carrier* $M = (R, A, H_0, H_1, W, P_s, D, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MomentProblem}^\uparrow}$ *surface consists exactly of rational moment-entry admission, real readback admission, finite Hankel matrix admission, finite-window scheduling, displayed positivity certification, optional distribution-boundary syntax, componentwise transport, displayed replay, provenance, and local naming for the rows*

$$R, A, H_0, H_1, W, P_s, D, H, C, Q, N.$$

No consumer of the certificate receives a representing measure, an infinite moment sequence, a uniqueness theorem, a selected support, or an ambient analytic moment-problem solver.

Proof. Project the accepted packet. The numeric content is limited to the rational ledger R, the real readback row A, and the two displayed Hankel matrices H_0, H_1.

The finite-window row W determines which moments and minors may be inspected. The positivity row P_s certifies only those displayed Hankel windows with the recorded rational tolerance budget, so no infinite positivity condition or representing-measure theorem is available from the carrier.

The structural rows H, C, Q, N then transport, replay, source, and name exactly the same packet. The optional distribution row D is boundary syntax for downstream probabilistic consumers, not an analytic existence witness; hence the local certificate exposes only the listed finite rows. □

Theorem 6193.3 (MomentProblem finite Hankel window)**.** *Every finite-window read of an accepted* $\mathsf{MomentProblem}^\uparrow$ *packet factors through the displayed route*

$$\mathsf{Rat}^\uparrow,\ \mathsf{Real}^\uparrow,\ \mathsf{Mat}^\uparrow,\ \mathsf{Mat}^\uparrow$$

before a positivity consumer may inspect Hankel data. The read must use the window schedule W, the positivity ledger P_s, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{MomentProblem}^\uparrow}$ *row; it cannot be strengthened to an infinite Hankel operator, a representing measure, a selected support, or an ambient moment-problem theorem.*

Lean4 ✓ `BEDC.Derived.MomentProblemUp.FiniteHankelWindow.MomentProblemCarrier finite hankel window`

Proof. Read the carrier of Definition 6193.1. The Hankel data appears only as the finite matrices H_0, H_1, and both are built from the rational and real readback rows R, A.

Apply Definition 6193.2. It forces every positivity-facing read through the window row W and the finite positivity ledger P_s before the structural rows can replay or package the request.

Because no carrier coordinate stores an infinite sequence, selected measure, support, determinacy proof, or host solver, the finite Hankel window is the whole public surface for moment-problem consumption. □

Closure status for $\mathsf{NameCert}_{\mathsf{MomentProblem}^{\uparrow}}$.

Constructive story: A $\mathsf{MomentProblem}^{\uparrow}$ packet is generated from RatUp coefficient rows, RealUp readback rows, finite MatrixUp Hankel windows, a finite-window schedule, displayed positivity certificates, an optional DistributionUp boundary row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MomentProblem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6193.1, Definition 6193.2, and Definition 6193.3 seed the finite moment-window route over RatUp, RealUp, MatrixUp, DistributionUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MomentProblem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a representing measure theorem, determinacy, uniqueness, selected support, ambient measure-space existence, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite Hankel-window exactness, positivity ledger non-escape, transport, replay, provenance, and local naming for BEDC.Derived.MomentProblemUp.

Chapter 6194

A Concrete Naming Certificate for $\mathsf{KleeneTreeBoundary}^{\uparrow}$

Following Kleene's tree counterexample tradition and Bishop–Bridges constructive analysis, the $\mathsf{KleeneTreeBoundary}^{\uparrow}$ packet records the finite boundary at which Kleene-tree prefix data, finite bar data, and completion-facing readbacks remain visible while branch selection and compactness principles stay refused. It is read after the prefix-tree carrier of chapter 1622, the binary prefix surface of chapter 1475, the finite-bar discipline of chapter 400, the stream schedule of chapter 126, the regular-rational readback of chapter 301, and the terminal real boundary of chapter 38. The packet isolates the constructive data a consumer may inspect before any illicit branch, open-cover compactness, countable choice, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness claim would be needed.

Definition 6194.1 (Kleene tree boundary carrier)**.** *A* $\mathsf{KleeneTreeBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (K, F, S, R, E, O, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{KleeneTree}^{\uparrow}$ *finite binary-prefix data, including the displayed tree ledger, node decisions, prefix spine, and obstruction row inherited from the Kleene-tree packet.*
2. *F is the finite* $\mathsf{FanTheorem}^{\uparrow}$ *bar data used only as displayed finite bar and modulus-facing evidence.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window schedule through which branch-facing prefix requests are read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational observations routed through the same finite windows.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the prefix, bar, stream, and regular-rational rows have been displayed.*
6. *O is the obstruction row refusing a selected branch, open-cover compactness, countable choice, quotient stream equality, and* $\mathsf{Real}^{\uparrow}$ *completeness as coordinates of the packet.*
7. *H records componentwise* hsame *transport for the tree, bar, stream, regular-rational, real-seal, obstruction, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from finite stream windows through the Kleene prefix data and finite bar rows, then through the regular-rational readback and real seal to the obstruction row.*

9. *P is the* Pkg *provenance over* $\mathsf{KleeneTreeBoundary}^{\uparrow}$, $\mathsf{KleeneTree}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KleeneTreeBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, S, R, E, O, H, C, P, N$. It has no coordinate for a completed infinite path, a Cantor-point selector, an open-cover compactness theorem, countable choice, quotient stream equality, or real completeness.

Lean4 ✓ `BEDC.Derived.KleeneTreeBoundaryUp.KleeneTreeBoundaryTasteGate single carrier alignment`

Theorem 6194.2 (Kleene tree boundary NameCert obligations)**.** *For every accepted* $\mathsf{KleeneTreeBoundary}^{\uparrow}$ *carrier $B = (K, F, S, R, E, O, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KleeneTreeBoundary}^{\uparrow}}$ *surface consists exactly of carrier admission, finite Kleene-prefix exposure, finite bar exposure, stream-window exposure, regular-rational readback, real-seal handoff, obstruction-row exposure, componentwise transport, displayed replay, provenance over the named dependency rows, and local naming.*

Proof. Project the carrier of Definition 6194.1. The rows K, F, S, R, E, O display the finite prefix, bar, stream, rational-readback, real-seal, and obstruction data that a consumer may inspect.

The structural rows H and C preserve this route componentwise. A request first enters through the finite $\mathsf{StreamName}^{\uparrow}$ window S, then reads the Kleene-prefix row K and the finite bar row F, then passes through R and E before reaching the obstruction row O.

The provenance and local name rows P, N package exactly that finite route. Since the carrier contains no coordinate for a selected branch, open-cover compactness, countable choice, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.KleeneTreeBoundaryUp.TasteGate.KleeneTreeBoundaryTasteGate single carrier alignment`

Theorem 6194.3 (Kleene tree boundary finite bar refusal)**.** *Every* $\mathsf{FanTheorem}^{\uparrow}$, $\mathsf{CantorSpace}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{KleeneTreeBoundary}^{\uparrow}$ *packet factors through the finite rows $K, F, S, R, E, O, H, C, P, N$. The consumer may use the displayed finite prefix and bar data, but it cannot turn those rows into a selected branch, open-cover compactness, countable choice, quotient stream equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite-window route supplied by S. Each branch-facing request is replayed through a displayed prefix row in K and a displayed finite bar row in F; no request reads an unlisted infinite path.

Next inspect the readback rows R and E. They carry regular-rational and real-facing observations only after the finite prefix and bar evidence has already been displayed, so they cannot add a hidden selector or compactness principle.

Finally read the obstruction row O. Finite-row exhaustion leaves only the displayed route $K, F, S, R, E, O, H, C, P, N$, and O explicitly refuses branch selection, open-cover compactness, countable choice, quotient stream equality, and real completeness as packet coordinates. □

Falsifiable prediction: An accepted $\mathsf{KleeneTreeBoundary}^{\uparrow}$ read is refuted if a consumer obtains a selected branch, open-cover compactness, countable choice, quotient stream equality, or $\mathsf{Real}^{\uparrow}$ completeness without displaying the finite Kleene-prefix row, finite bar row, stream window, regular-rational readback, real seal, obstruction row, transport, replay, provenance, and local naming row.

Independence witness: KleeneTree, FanTheorem, StreamName, RegSeqRat, and Real: none is bijective to $\mathsf{KleeneTreeBoundary}^{\uparrow}$ because the boundary packet jointly records finite prefix data,

finite bar data, completion-facing readbacks, and an explicit obstruction row refusing branch selection and compactness principles.

Closure status for $\mathsf{NameCert}_{\mathsf{KleeneTreeBoundary}\uparrow}$.

Constructive story: A $\mathsf{KleeneTreeBoundary}^{\uparrow}$ packet is generated from finite Kleene-tree prefix data, finite FanTheorem bar data, StreamName windows, RegSeqRat readbacks, a Real seal, an obstruction row, componentwise hsame transport, displayed continuation replay, package provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KleeneTreeBoundary}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6194.1, Definition 6194.2, and Definition 6194.3 seed the finite boundary separating Kleene-tree and bar evidence from branch selection, compactness, choice, quotient equality, and real completeness claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KleeneTreeBoundary}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected branches, open-cover compactness, countable choice, quotient stream equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite prefix and bar exposure, readback and real-seal handoff, obstruction-row refusal, transport, replay, provenance, and local naming for BEDC.Derived.KleeneTreeBoundaryUp.

Chapter 6195

A Concrete Naming Certificate for $\mathsf{TotalBoundednessModulus}^{\uparrow}$

Following Bishop and Bridges on constructive metric compactness, the $\mathsf{TotalBoundednessModulus}^{\uparrow}$ packet records the finite-net schedule by which a totally bounded metric source supplies explicit net rows at each displayed tolerance. It sits after the $\mathsf{Met}^{\uparrow}$ metric carrier of chapter 57, the total-boundedness certificate of chapter 127, the compact metric surface of chapter 129, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite modulus neighbour for the L10 compact-metric route; it is not an open-cover compactness theorem, a selected subcover, a quotient point equality principle, or an ambient completeness assertion.

Definition 6195.1 (Total boundedness modulus carrier)**.** *A* $\mathsf{TotalBoundednessModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (M, N, D, R, E, H, C, P, L)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, carrying point classifiers, distance observations, and metric comparison data.*
2. *N is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net schedule row, assigning each admitted positive tolerance a finite displayed center list and coverage witness.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording halving, comparison, and positivity reads used by the net schedule.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row through which metric distances and tolerance comparisons are exposed to compact-metric consumers.*
5. *E is the* $\mathsf{CompactMetric}^{\uparrow}$ *handoff row, recording that compact-metric consumers may read the modulus only after the metric, total-boundedness, dyadic, and real-seal rows are visible.*
6. *H records componentwise* hsame *transport for the metric, finite-net schedule, dyadic tolerance, real seal, compact handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a compact-metric request through M, N, D, R, E, H, P, L.*
8. *P is the* Pkg *provenance over* $\mathsf{TotalBoundednessModulus}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. L is the local $\mathsf{NameCert}_{\mathsf{TotalBoundednessModulus}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $M, N, D, R, E, H, C, P, L$. It has no coordinate for arbitrary open covers, selected finite subcovers, countable choice, quotient point equality, host metric completeness, or an ambient $\mathsf{Real}^{\uparrow}$ *equality relation.*

Lean4 ✓ `BEDC.Derived.TotalBoundednessModulusUp.TotalBoundednessModulusTasteGate single carrier alignment`

Theorem 6195.2 (Total boundedness modulus NameCert obligations)**.** *For every accepted* TotalBoundednessModu *carrier $T = (M, N, D, R, E, H, C, P, L)$, the local* $\mathsf{NameCert}_{\mathsf{TotalBoundednessModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for T, metric-source admission through M, finite-net schedule exposure through N, dyadic tolerance accounting through D, real-seal exposure through R, compact-metric handoff through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through L.*

Proof. Project the accepted carrier of Definition 6195.1. The metric source is M, the schedule row is N, and the tolerance arithmetic is D.

The real-facing and compact-facing rows are R and E. They expose distance and tolerance comparisons only after the metric and finite-net schedule rows are visible, so a consumer cannot replace the packet by an open-cover compactness principle or by a selected subcover.

Finally the structural rows H, C, P, L transport, replay, source, and name the same finite route. Since no additional coordinate is present, the listed rows exhaust the NameCert obligations and no quotient point equality, countable-choice schedule, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ equality can be exported. □

Lean4 ✓ `BEDC.Derived.TotalBoundednessModulusUp.TasteGate.TotalBoundednessModulusTasteGate single carrier alignment`

Theorem 6195.3 (Total boundedness modulus finite-net handoff)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, *or completion-facing consumer of an accepted* $\mathsf{TotalBoundednessModulus}^{\uparrow}$ *carrier receives finite-net data only through the displayed route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{TotallyBounded}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CompactMetric}^{\uparrow}.$$

For each admitted positive dyadic tolerance, the consumer may inspect the finite center list and coverage row carried by N, the tolerance comparison ledger D, the real-valued distance seal R, and the compact-metric handoff E, together with H, C, P, L. It receives no open-cover compactness theorem, selected finite subcover, countable-choice schedule, quotient point equality, host metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *equality.*

Proof. Start from Definition 6195.1. The source metric row M and the total-boundedness row N are the only rows that can expose centers and coverage witnesses.

Read the tolerance and distance boundary through D and R. The dyadic ledger controls each positive request, and the real seal exposes the corresponding distance comparison before the compact-metric handoff is available.

Apply the obligation surface of Definition 6195.2. It forces the compact-facing row E and the structural rows H, C, P, L to remain visible for every handoff.

Because the carrier contains no open-cover, subcover-selection, choice, quotient, host-completeness, or ambient-real-equality coordinate, the finite-net handoff cannot export any of those objects. □

Theorem 6195.4 (Total boundedness modulus schedule induction)**.** *Let $T = (M, N, D, R, E, H, C, P, L)$ be an accepted* $\mathsf{TotalBoundednessModulus}^{\uparrow}$ *carrier. For every finite displayed request list*

$$\varepsilon_1, \ldots, \varepsilon_k$$

whose entries are admitted positive dyadic tolerances in D, the carrier answers the list by finite induction over the displayed order. The empty list uses the empty schedule read. The cons step reads the next tolerance through D, obtains the finite center list and coverage witness for that tolerance from N, exposes the distance comparison through R, passes it across the compact-metric handoff row E, and keeps the rows H, C, P, L as the transport, replay, provenance, and local naming data for the same request.

Proof. Induct on the displayed request list. For the empty list, no tolerance is requested, so the empty schedule read is the unique finite response and no row outside $M, N, D, R, E, H, C, P, L$ is inspected.

For the cons step, assume the tail list has already been answered by the induction hypothesis. The head tolerance is admitted by D, so N supplies its finite center list and coverage row. The comparison used by that coverage row is exposed through R, and the compact-facing access is routed through E.

The structural rows H, C, P, L transport, replay, source, and name the head response together with the tail response. Thus the enlarged finite list is answered by the same displayed rows, and the induction constructs the whole tolerance-indexed net schedule without invoking open covers, selected subcovers, countable choice, quotient point equality, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ equality. □

Theorem 6195.5 (Total boundedness modulus request monotonicity)**.** *Let $T = (M, N, D, R, E, H, C, P, L)$ be an accepted* $\mathsf{TotalBoundednessModulus}^{\uparrow}$ *carrier. If an admitted positive dyadic tolerance request is refined through the ledger row D, then the finite center list read from N for the refined request is obtained by reusing the earlier displayed list when the ledger records equality of tolerance reads, or by reading the refined finite list and coverage row from N when D records a proper refinement. In both cases the refined request remains inside the same $M, N, D, R, E, H, C, P, L$ carrier surface.*

Proof. Apply Definition 6195.4 to the two-request list consisting of the original tolerance and its displayed refinement. The first request produces the earlier finite center list and coverage row through N, with its tolerance comparison controlled by D and sealed through R.

For the second request, the ledger D decides whether the displayed refinement is the same tolerance read or a proper smaller request. In the equal-read case, the first center list may be reused because the schedule row is indexed by the same admitted tolerance. In the proper-refinement case, the second induction step reads the refined finite center list and coverage row from N.

The projection to the second request keeps the compact-facing handoff E and the structural rows H, C, P, L attached to the refined row. Hence request refinement changes only the tolerance-indexed read inside D and N; it does not leave the accepted $\mathsf{TotalBoundednessModulus}^{\uparrow}$ carrier surface. □

Falsifiable prediction: Every accepted $\mathsf{TotalBoundednessModulus}^{\uparrow}$ finite-net read factors through one $\mathsf{Met}^{\uparrow}$ source row, one $\mathsf{TotallyBounded}^{\uparrow}$ schedule row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, one $\mathsf{Real}^{\uparrow}$ distance seal, one $\mathsf{CompactMetric}^{\uparrow}$ handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ naming.

Independence witness: metric, totallybounded, compactmetric, dyadicratcore, real: $\mathsf{TotalBoundednessModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds metric observations, tolerance-indexed finite-net schedules, dyadic tolerance arithmetic, Real-facing distance seals, compact-metric handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TotalBoundednessModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TotalBoundednessModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6195.1, Definition 6195.2, Definition 6195.3, Definition 6195.4, and Definition 6195.5 seed the finite-net modulus route over MetricSpace, TotallyBounded, CompactMetric, DyadicRatCore, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TotalBoundednessModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, countable choice, quotient point equality, host metric completeness, ambient Real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, metric-source admission, finite-net schedule exposure, dyadic tolerance exactness, Real seal handoff, compact-metric handoff, structural transport, replay, provenance, and local naming for BEDC.Derived.TotalBoundednessModulusUp.

Constructive story: A $\mathsf{TotalBoundednessModulus}^{\uparrow}$ packet is generated from a MetricSpaceUp source row, a TotallyBoundedUp finite-net schedule, a DyadicRatCoreUp tolerance ledger, a RealUp distance seal, a CompactMetricUp handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6196

A Concrete Naming Certificate for $\mathsf{HyperbolicSet}^{\uparrow}$

Following Smale and Katok–Hasselblatt, the $\mathsf{HyperbolicSet}^{\uparrow}$ packet records the finite BEDC surface for an invariant set whose displayed tangent data split into stable and unstable rows with bounded contraction and expansion budgets. It sits over the finite routed dynamics of chapter 195, the metric observation surface of chapter 57, the compact-metric source discipline of chapter 129, and the stable-manifold consumer boundary of chapter 1649. The packet names a finite hyperbolic-set certificate; it does not assert global structural stability, shadowing, measure recurrence, or complete stable-manifold theory.

Definition 6196.1 (Hyperbolic set carrier)**.** *A* $\mathsf{HyperbolicSet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, D, M, K, E^s, E^u, \Lambda_s, \Lambda_u, A, H, C, P, N)$$

with the following displayed rows:

1. *X is the invariant-set carrier row, naming the inspected subset and the finite membership windows through which it is read.*
2. *D is the* $\mathsf{DynSystem}^{\uparrow}$ *row whose displayed transition data preserve the invariant-set row X.*
3. *M is the* $\mathsf{Met}^{\uparrow}$ *observation row for the source distances used by the finite estimates.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *support row, recording the compact metric windows on which the estimates are certified.*
5. *E^s and E^u are the stable and unstable splitting rows, displaying the two named directions over the inspected invariant-set windows.*
6. *Λ_s is the finite contraction-budget row for stable directions, and Λ_u is the finite expansion-budget row for unstable directions.*
7. *A is the* $\mathsf{StableManifold}^{\uparrow}$ *handoff row, exposing only the local splitting data that a stable-manifold consumer may read.*
8. *H records componentwise* hsame *transport for invariant-set, dynamical-system, metric, compact-metric, splitting, budget, handoff, provenance, replay, and local naming rows.*
9. *C records displayed* Cont *replay from a hyperbolic-set request through $X, D, M, K, E^s, E^u, \Lambda_s, \Lambda_u, A, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicSet}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StableManifold}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for global structural stability, shadowing, measure recurrence, complete stable-manifold theory, a host tangent bundle, or an ambient smooth manifold theorem.

Lean4 ✓ `BEDC.Derived.HyperbolicSetUp.HyperbolicSetTasteGate single carrier alignment`

Theorem 6196.2 (Hyperbolic set NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicSet}^{\uparrow}$ *carrier* $Q = (X, D, M, K, E^s, E^u, \Lambda_s, \Lambda_u, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicSet}^{\uparrow}}$ *surface is exhausted by the following obligation rows:*

1. *carrier admission for invariant-set, dynamical-system, metric, compact-metric, stable-splitting, unstable-splitting, contraction-budget, expansion-budget, handoff, transport, replay, provenance, and local naming rows;*
2. *invariant-set preservation, requiring the displayed* $\mathsf{DynSystem}^{\uparrow}$ *row to preserve the inspected membership windows of X;*
3. *metric and compact support, requiring M and K to expose the finite distance and compact-window data before any splitting estimate is read;*
4. *splitting visibility, requiring E^s and E^u to remain separate named rows rather than aliases of the invariant-set or metric rows;*
5. *budget exactness, requiring Λ_s and Λ_u to record finite contraction and expansion estimates for the displayed splitting rows;*
6. *stable-manifold handoff, requiring A to expose only the local splitting-and-budget data consumed by* $\mathsf{StableManifold}^{\uparrow}$*;*
7. *ledger nonescape, requiring H, C, P, N to export no global structural stability, shadowing theorem, measure recurrence theorem, complete stable-manifold theory, host tangent bundle, or ambient smooth-manifold principle.*

No local NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 6196.1. The accepted rows are exactly $X, D, M, K, E^s, E^u, \Lambda_s, \Lambda_u, A, H, C, P$ so carrier admission exposes only finite invariant-set, dynamical-system, metric, compact-support, splitting, budget, handoff, transport, replay, provenance, and local naming data.

The route first reads X through D. Thus invariant-set preservation is a displayed $\mathsf{DynSystem}^{\uparrow}$ obligation, not a host theorem about all orbits of an ambient dynamical system.

The metric and compact rows M, K are then read before the splitting rows. They bind the stable and unstable estimates to finite distance windows and compact-metric support.

Finally the separate rows $E^s, E^u, \Lambda_s, \Lambda_u$ provide the splitting and its budgets, while A, H, C, P, N only hand off, transport, replay, source, and name that same finite packet. Since none of these rows adds global structural stability, shadowing, measure recurrence, complete stable-manifold theory, a host tangent bundle, or an ambient smooth-manifold theorem, the listed obligations exhaust the NameCert surface. □

Theorem 6196.3 (Hyperbolic set splitting transport)**.** *Every stable-manifold or dynamical-systems consumer of an accepted* $\mathsf{HyperbolicSet}^{\uparrow}$ *packet factors through the displayed route*

$$\mathsf{DynSystem}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow} \longmapsto E^s, E^u, \Lambda_s, \Lambda_u \longmapsto \mathsf{StableManifold}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the consumer may transport stable and unstable splitting data, finite contraction and expansion budgets, compact-metric support, componentwise hsame*, displayed* Cont *replay,* Pkg *provenance, and local naming. It obtains no global structural stability theorem, shadowing theorem, measure recurrence theorem, or complete stable-manifold theory.*

Proof. Begin with a downstream consumer. By Definition 6196.2, the consumer must first read the invariant-set and $\mathsf{DynSystem}^{\uparrow}$ preservation rows, then the $\mathsf{Met}^{\uparrow}$ and $\mathsf{CompactMetric}^{\uparrow}$ support rows.

Next read $E^s, E^u, \Lambda_s, \Lambda_u$. These rows carry the stable and unstable splitting together with their finite budgets, so transport along hsame preserves the displayed splitting fields rather than replacing them by a host tangent-bundle construction.

The $\mathsf{StableManifold}^{\uparrow}$ handoff row A is downstream of those displayed fields. It exposes local splitting-and-budget data only, while C, P, N replay, source, and name the same packet.

No row in the carrier is a global structural-stability argument, a shadowing principle, a recurrence theorem, or a complete stable-manifold construction. The transported surface is therefore exactly the finite BEDC route displayed above. □

Falsifiable prediction: An accepted $\mathsf{HyperbolicSet}^{\uparrow}$ packet is refuted if a consumer obtains global structural stability, shadowing, measure recurrence, or complete stable-manifold theory without first displaying invariant-set, DynSystem, Metric, CompactMetric, stable-splitting, unstable-splitting, contraction-budget, expansion-budget, handoff, transport, replay, provenance, and local NameCert rows.

Independence witness: dynsystem, metric, compactmetric, stablemanifold: none is bijective to $\mathsf{HyperbolicSet}^{\uparrow}$ because this packet jointly carries invariant set preservation, metric and compact support, stable and unstable splitting, finite contraction and expansion budgets, stable-manifold handoff, transport, replay, package sourcing, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicSet}^{\uparrow}}$.

Constructive story: A $\mathsf{HyperbolicSet}^{\uparrow}$ packet is generated from an invariant-set carrier row, DynSystem transition data, Metric and CompactMetric support rows, stable and unstable splitting rows, finite contraction and expansion budgets, a StableManifold handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{HyperbolicSet}^{\uparrow}}$, seed)

Scope closed: Definition 6196.1, Definition 6196.2, and Definition 6196.3 seed the finite hyperbolic-set NameCert packet over invariant-set, dynamics, metric, compact-support, splitting, and budget rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{HyperbolicSet}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close global structural stability, shadowing, measure recurrence, complete stable-manifold theory, host tangent bundles, ambient smooth-manifold theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked invariant-set preservation, metric and compact support, splitting visibility, contraction and expansion budgets, stable-manifold handoff, field faithfulness, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicSetUp.

Chapter 6197

A Concrete Naming Certificate for $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$

Following Bishop and Bridges constructive analysis and the standard decimal-expansion normalization practice, the $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$ packet records the finite surface by which a located decimal stream is accepted in normal form before it is read as regular rational or Real-facing data. It sits downstream of the decimal stream carrier of chapter 4191, the signed decimal expansion route of chapter 5150, the endpoint normalization packet of chapter 1429, the finite-window naming discipline of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names an L10 finite normalization interface only; it does not import quotient streams, selected limits, host real equality, digit-choice omniscience, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6197.1 (LocatedDecimalStreamNormal carrier)**.** *A* $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (D, S, E, W, R, Y, T, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DecimalStream}^{\uparrow}$ *source row, naming the inspected signed digit stream and the finite digit windows available to the packet.*
2. *S is the* $\mathsf{SignedDecimalExpansion}^{\uparrow}$ *sign-and-digit row, exposing the sign convention and digit ledger used by the same windows.*
3. *E is the* $\mathsf{DecimalEndpointNormalization}^{\uparrow}$ *endpoint row, recording the finite trailing-nine or trailing-zero normalization decisions visible at the inspected endpoints.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row, listing the finite stages, digit positions, and endpoint windows read before the normal-form request is accepted.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the normalized windows.*
6. *Y is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, carrying dyadic radii and endpoint residues attached to the readback row R.*
7. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after D, S, E, W, R, Y have been displayed.*
8. *H records componentwise* hsame *transport for decimal streams, signed expansions, endpoint normalization, schedules, regular readbacks, dyadic tolerances, Real seals, replay, provenance, and local naming.*

9. *C records displayed* Cont *replay from a located decimal-stream normal-form request through* D, S, E, W, R, Y, T, H.
10. *P is the* Pkg *provenance over* LocatedDecimalStreamNormal$^{\uparrow}$, DecimalStream$^{\uparrow}$, SignedDecimalExpansion$^{\uparrow}$, DecimalEndpointNormalization$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* NameCert$_{\text{LocatedDecimalStreamNormal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a selected limiting real, host real equality, a hidden digit-choice principle, or ambient completeness of Real$^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedDecimalStreamNormalUp.TasteGate.LocatedDecimalStreamNormalTasteGate single carrier alignment`

Theorem 6197.2 (LocatedDecimalStreamNormal NameCert obligations)**.** *For every accepted* LocatedDecimalStreamNormal$^{\uparrow}$ *carrier*

$$Q = (D, S, E, W, R, Y, T, H, C, P, N),$$

the local NameCert$_{\text{LocatedDecimalStreamNormal}^{\uparrow}}$ *surface consists exactly of carrier admission for the decimal-stream source D, signed expansion exposure through S, endpoint normalization exposure through E, finite* StreamName$^{\uparrow}$ *schedule exposure through W, regular rational readback through R, dyadic tolerance access through Y, terminal* Real$^{\uparrow}$ *seal admission through T, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6197.1. The decimal source is D, and the only signed digit data available to a consumer is the row S attached to the same inspected stream windows.

The endpoint and schedule rows then fix the normal-form boundary. The row E records the finite endpoint normalization decisions, while W lists the stages and positions at which those decisions are visible.

The rational and real-facing rows are downstream of that boundary. The readback row R is produced from the normalized windows, the dyadic ledger Y records the tolerance data used by that readback, and the Real seal T is reached only after the preceding rows have been displayed.

Finally H, C, P, N transport, replay, source, and name exactly the same finite packet. Since the carrier has no row for quotient stream equality, selected limits, host real equality, digit-choice omniscience, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6197.3 (LocatedDecimalStreamNormal regular readback handoff)**.** *Every* RegSeqRat$^{\uparrow}$ *or* Real$^{\uparrow}$ *read exported by an accepted* LocatedDecimalStreamNormal$^{\uparrow}$ *carrier factors through the finite route*

DecimalStream$^{\uparrow}$, SignedDecimalExpansion$^{\uparrow}$, DecimalEndpointNormalization$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRa

Thus a downstream completion or real-name consumer receives only the displayed decimal stream, signed digit ledger, endpoint normalization row, finite schedule, regular rational readback, dyadic tolerance ledger, Real seal, and structural rows H, C, P, N.

Proof. Read the decimal and signed-expansion rows first. The source row D supplies the inspected stream windows, and the row S fixes the sign and digit ledger used by those same windows.

The normal-form handoff cannot bypass endpoint normalization. The row E records the finite trailing-endpoint decisions, and the schedule row W records the stages and digit positions at which those decisions are visible.

The downstream readback is therefore forced to pass through R, Y, T. The regular rational row R is the only readback row in the carrier, the dyadic row Y is the only tolerance ledger attached to it, and the terminal row T is the only Real-facing seal.

The structural rows H, C, P, N transport, replay, source, and name that same route. No refused object is a coordinate of the carrier, so no consumer can read quotient stream equality, a selected limit, host real equality, digit-choice omniscience, or ambient Real completeness through this handoff. □

Falsifiable prediction: An accepted $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$ consumer that reads a regular rational or Real-facing row without first displaying the decimal stream, signed expansion, endpoint normalization, finite schedule, dyadic tolerance ledger, transport, replay, provenance, and local naming refutes the carrier surface.

Independence witness: decimal-stream, signed-decimal-expansion, decimal-endpoint-normalization, streamname, regseqrat, dyadicratcore, real: $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries decimal-stream source rows, signed expansion rows, endpoint normalization rows, finite StreamName schedules, RegSeqRat readbacks, DyadicRatCore tolerances, Real seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDecimalStreamNormal}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedDecimalStreamNormal}^{\uparrow}$ packet is generated from decimal-stream source rows, signed expansion rows, endpoint normalization rows, finite StreamName schedule rows, RegSeqRat readbacks, DyadicRatCore tolerance rows, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedDecimalStreamNormal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6197.1, Definition 6197.2, and Definition 6197.3 seed the finite located decimal-stream normal-form route over decimal streams, signed expansions, endpoint normalization, schedules, regular readbacks, dyadic tolerances, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedDecimalStreamNormal}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.LocatedDecimalStreamNormalUp.TasteGate.LocatedDecimalStreamNormalTasteGate\_single\_car`

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, host real equality, digit-choice omniscience, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, signed expansion exposure, endpoint normalization exactness, RegSeqRat readback handoff, dyadic tolerance stability, Real seal nonescape, and structural transport for BEDC.Derived.LocatedDecimalStreamNormalUp.

Chapter 6198

A Concrete Naming Certificate for $\mathsf{ChurchRosser}^{\uparrow}$

Following Church and Rosser, the $\mathsf{ChurchRosser}^{\uparrow}$ packet records confluence as a named finite route from local join evidence to equality of normal forms, rather than as host syntax quotienting or evaluator equality. In the MetaCIC route it sits after the local-diamond witness of chapter 3595, the residual-substitution boundary of chapter 2181, the closed-normal fragment of chapter 737, the parallel-diamond frontier of chapter 4238, and the dependency ledger of chapter 765. The packet gives a traditional Church–Rosser naming surface for the conditional MetaCIC confluence route; it refuses quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, and any claim that the current frontier rows already prove confluence for all closed terms.

Definition 6198.1 (Church–Rosser carrier)**.** *A* $\mathsf{ChurchRosser}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (T, L, S, NF, P, D, C, O, H, Q, G, M)$$

with the following displayed rows:

1. *T is the closed source-term row whose confluence request is being named.*
2. *L is the local-diamond ledger row, read through* $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$ *and retaining the displayed peak and candidate-join evidence.*
3. *S is the residual-substitution row, read through* $\mathsf{MetaCICResidualSubstitutionBoundary}^{\uparrow}$ *and any separate compatibility handoff available to the route.*
4. *NF is the closed-normal-fragment row, read through* $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$ *before a normal-form comparison can be consumed.*
5. *P is the parallel-diamond frontier row, read through* $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$ *as conditional frontier data.*
6. *D is the critical-path dependency row, read through* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *and keeping subject reduction, normalization, confluence, decidability, and consistency boundaries visible as dependencies rather than as supplied theorems.*
7. *C is the Church–Rosser conclusion row: it records the bounded common-reduct or normal-form agreement certified by L, S, NF, P, D for the displayed request.*
8. *O is the obstruction row naming endpoints not carried by this packet: quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, premise-free parallel diamond, and unconditional confluence for every closed term.*

9. *H records componentwise* hsame *transport for T, L, S, NF, P, D, C, O.*
10. *Q records displayed* Cont *replay for Church–Rosser requests through the same rows.*
11. *G is the* Pkg *provenance over* $\mathsf{ChurchRosser}^{\uparrow}$, $\mathsf{MetaCICLocalDiamondWitness}^{\uparrow}$, $\mathsf{MetaCICResidualSubstitutionBound}$ $\mathsf{MetaCICConfluenceNormalFragment}^{\uparrow}$, $\mathsf{MetaCICParallelDiamondFrontier}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *M is the local* $\mathsf{NameCert}_{\mathsf{ChurchRosser}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $T, L, S, NF, P, D, C, O, H, Q, G, M$. It has no coordinate for quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, premise-free parallel diamond, or an unrestricted confluence theorem over all closed MetaCIC terms.

Lean4 ✓ `BEDC.Derived.ChurchRosserUp.ChurchRosserTasteGate single carrier alignment`

Theorem 6198.2 (Church–Rosser diamond ledger)**.** *For every accepted* $\mathsf{ChurchRosser}^{\uparrow}$ *carrier $R = (T, L, S, NF, P, D, C, O, H, Q, G, M)$, any Church–Rosser read must expose the local-diamond ledger L, residual-substitution row S, closed-normal-fragment row NF, parallel-diamond frontier row P, and MetaCIC critical-path row D before the conclusion row C is consumed. The conclusion is therefore ledger-scoped: it records the displayed join route for T and does not export quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, or unrestricted closed-term confluence.*

Proof. Project the carrier of Definition 6198.1. The confluence-facing rows are L, S, NF, P, D, C, O, and the source row T determines which closed request those rows address.

The local-diamond and residual-substitution rows are read first. The row L supplies only displayed local peaks and candidate joins, while S keeps the substitution premise visible; neither row is allowed to erase its boundary into a premise-free global theorem.

Next the closed-normal fragment and parallel-diamond frontier are read. The row NF supplies the normal-source and atom-source fragment, and P supplies conditional frontier data rather than a completed parallel diamond.

Finally the critical-path row D keeps the surrounding MetaCIC obligations visible before C is consumed. Since the obstruction row O remains part of the same packet and the structural rows H, Q, G, M only transport, replay, source, and name the displayed route, no refused endpoint can be exported by the ledger. □

Theorem 6198.3 (Church–Rosser normal-form confluence)**.** *If an accepted* $\mathsf{ChurchRosser}^{\uparrow}$ *carrier $R = (T, L, S, NF, P, D, C, O, H, Q, G, M)$ reaches a conclusion row C whose endpoints are both displayed normal-form reads, then the normal-form agreement factors through the closed-normal fragment NF and the same local-diamond, residual-substitution, parallel-frontier, and critical-path rows L, S, P, D. The agreement is a certified normal-form confluence read for the displayed source T; it is not host evaluator equality, quotient term equality, arbitrary subject reduction, full consistency, or an unrestricted Church–Rosser theorem for all closed terms.*

Proof. Begin with the diamond ledger of Definition 6198.2. It forces any read of C through L, S, NF, P, D and keeps the obstruction row O in scope.

The normal-form hypothesis only selects the NF branch of that same ledger. Thus the agreement between endpoints is read through the closed-normal fragment together with the displayed local-diamond and residual-substitution rows, not through a host evaluator or a quotient of term syntax.

The parallel frontier and critical path remain visible. They record which conditional confluence and dependency rows have been consumed before the normal-form agreement is named, so the result stays attached to the displayed source request T and does not become a global subject-reduction, consistency, or unrestricted confluence theorem. □

Theorem 6198.4 (Church–Rosser MetaCIC route boundary)**.** *Every downstream MetaCIC consumer of an accepted* $\mathsf{ChurchRosser}^{\uparrow}$ *packet receives only the finite route*

$$T, L, S, NF, P, D, C, O, H, Q, G, M.$$

In particular, the consumer may cite the packet as a Church–Rosser NameCert surface for the displayed local-diamond, residual-substitution, closed-normal-fragment, parallel-frontier, and critical-path route, but it may not reinterpret that surface as quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, or arbitrary closed Church–Rosser.

Proof. Use the obligation surface of Definition 6198.1. The only rows available to a consumer are the displayed source, local-diamond, residual-substitution, closed-normal-fragment, parallel-frontier, critical-path, conclusion, obstruction, transport, replay, provenance, and local naming rows.

Apply the diamond ledger to the confluence-facing part of the route, then apply the normal-form confluence theorem when the consumer asks for a normal-form endpoint comparison. Both reads keep O visible and retain the same finite row order.

The structural rows do not add semantic coordinates. They transport and replay the packet already named by G and M, so a downstream consumer that claims quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, or arbitrary closed Church–Rosser has left the packet. □

Falsifiable prediction: Every accepted $\mathsf{ChurchRosser}^{\uparrow}$ confluence read exposes local-diamond, residual-substitution, closed-normal-fragment, parallel-frontier, critical-path, and obstruction rows before its conclusion row is consumed.

Independence witness: metacic-local-diamond-witness, metacic-residual-substitution-boundary, metacic-confluence-normal-fragment, metacic-parallel-diamond-frontier, metacic-critical-path: these siblings respectively expose local peaks, substitution premises, closed-normal joins, conditional frontier data, and dependency pressure. None is bijective to a packet that jointly names the traditional Church–Rosser conclusion row together with all five route rows and an explicit obstruction row.

Closure status for $\mathsf{NameCert}_{\mathsf{ChurchRosser}^{\uparrow}}$.

Constructive story: A $\mathsf{ChurchRosser}^{\uparrow}$ packet is generated from a closed source row, local-diamond ledger, residual-substitution row, closed-normal-fragment row, parallel-diamond frontier row, MetaCIC critical-path row, conclusion row, obstruction row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChurchRosser}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6198.1, Definition 6198.2, Definition 6198.3, and Definition 6198.4 seed the traditional Church–Rosser NameCert surface for the displayed MetaCIC local-diamond, residual-substitution, closed-normal-fragment, parallel-frontier, and critical-path route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChurchRosser}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient syntax, host evaluator equality, arbitrary subject reduction, full consistency, premise-free parallel diamond, arbitrary closed Church–Rosser, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-diamond ledger exposure, residual-substitution retention, closed-normal-fragment routing, parallel-frontier handoff, critical-path retention, conclusion-row exactness, obstruction-row exposure, transport, replay, provenance, and local naming for BEDC.Derived.ChurchRosserUp.

Chapter 6199

A Concrete Naming Certificate for $\mathsf{LocatedIntervalRefinement}^{\uparrow}$

Bishop-style interval refinement treats nested located intervals as finite endpoint, mesh, and containment data before any real-completion argument reads a limit. The $\mathsf{LocatedIntervalRefinement}^{\uparrow}$ packet records that route for the L10 Real and Completion surface: it refines the located interval carrier of chapter 394, uses the dyadic one-step route of chapter 1831, inherits the bisection discipline of chapter 2081, reads endpoint tolerances through chapter 1933, and keeps the arithmetic budget inside chapter 291. The packet supplies a finite refinement surface for interval, compactness, and regular-Cauchy consumers without importing open-cover compactness, quotient equality, selected branches, or ambient completeness.

Definition 6199.1 (Located interval refinement carrier)**.** *A* $\mathsf{LocatedIntervalRefinement}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (L, U, M, F, Q, A, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower located-endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet visible before any refinement request is accepted.*
2. *U is the upper located-endpoint row, a* $\mathsf{LocatedReal}^{\uparrow}$ *packet displayed in the same history as L.*
3. *M is the dyadic mesh row supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*; it records the active mesh cells, radius budget, midpoint cells, slack, and comparison tolerances used by the refinement.*
4. *F is the refinement map row, sending each accepted parent located interval cell to its displayed child cells through the finite* $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ *route.*
5. *Q is the containment witness row, recording parent-to-child inclusion, endpoint order, and visible gap checks for every accepted finite refinement request.*
6. *A is the apartness and tolerance ledger, tying endpoint apartness, mesh diameter, positive slack, and requested completion tolerance to the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *budget.*
7. *H records componentwise* hsame *transport for located endpoints, dyadic mesh, refinement map, containment witnesses, apartness and tolerance ledgers, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay for finite refinement requests through L, U, M, F, Q, A, H.*
9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalRefinement}^{\uparrow}$, $\mathsf{LocatedInterval}^{\uparrow}$, $\mathsf{DyadicIntervalRefinement}^{\uparrow}$, $\mathsf{BishopIntervalHalving}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalRefinement}^{\uparrow}}$ *row.*

The classifier reads only these rows. It has no coordinate for open-cover compactness, quotient equality of real names, selected branches of an infinite bisection tree, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.LocatedIntervalRefinementUp.LocatedIntervalRefinementTasteGate single carrier alignment`

Theorem 6199.2 (Located interval refinement cover transport)**.** *For every accepted* $\mathsf{LocatedIntervalRefinement}^{\uparrow}$ *carrier* $R = (L, U, M, F, Q, A, H, C, P, N)$*, any interval-cover or compactness-facing consumer that transports a refinement cover must expose the located endpoint rows* L, U*, the dyadic mesh row* M*, the refinement map* F*, the containment witnesses* Q*, and the apartness and tolerance ledger* A *before reading the transported cover. The transport is componentwise along* H *and replayed by* C*; it exports no host open cover, selected finite subcover, quotient endpoint equality, or ambient compactness principle.*

Proof. Project the carrier of Definition 6199.1. The endpoints L, U supply the only located-real boundary, and the dyadic mesh M supplies the only finite scale at which a refinement can be inspected.

The refinement map and containment witnesses are then read as F and Q. Thus a transported cover first sees the same parent cell, child cells, endpoint order, and visible gap checks that are already present in the finite refinement packet.

Finally the ledger A binds the apartness and tolerance reads to the same dyadic budget. Componentwise transport through H and replay through C can only repack these displayed rows, so no host open-cover compactness, selected subcover, quotient endpoint equality, or ambient compactness coordinate can enter the transported cover. □

Theorem 6199.3 (Located interval refinement NameCert obligations)**.** *For every accepted* LocatedIntervalRefineme *carrier* $R = (L, U, M, F, Q, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalRefinement}^{\uparrow}}$ *surface consists exactly of located endpoint admission, dyadic mesh admission, refinement-map admission, containment witness admission, apartness and tolerance ledger admission, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the displayed rows* $L, U, M, F, Q, A, H, C, P, N$.

Proof. Carrier admission is projection from Definition 6199.1. The endpoint rows are L, U, the finite arithmetic surface is M, the one-step refinement route is F, and the containment and tolerance evidence is Q, A.

The sibling routes determine how these rows may be consumed. The $\mathsf{LocatedInterval}^{\uparrow}$ packet supplies endpoint and interval-cell visibility, $\mathsf{DyadicIntervalRefinement}^{\uparrow}$ supplies the child-cell refinement row, $\mathsf{BishopIntervalHalving}^{\uparrow}$ supplies the bisection reading discipline, $\mathsf{LocatedReal}^{\uparrow}$ supplies the endpoint tolerance surface, and $\mathsf{DyadicRatCore}^{\uparrow}$ supplies the arithmetic ledger.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier contains no row for selected branches, open-cover compactness, quotient equality, countable choice, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalRefinement}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalRefinement}^{\uparrow}$ packet is generated from located endpoint rows, a DyadicRatCore mesh, a finite refinement map, containment witnesses, an apartness and tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedIntervalRefinement}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6199.1, Definition 6199.2, and Definition 6199.3 seed the finite located-interval refinement surface over LocatedInterval$^{\uparrow}$, DyadicIntervalRefinement$^{\uparrow}$, BishopIntervalHalving$^{\uparrow}$, LocatedReal$^{\uparrow}$, and DyadicRatCore$^{\uparrow}$ rows.

Formal verification: FormalStatus(NameCert$_{\text{LocatedIntervalRefinement}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected finite subcovers, selected refinement branches, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked endpoint admission, dyadic mesh admission, refinement-map coverage, containment witnesses, tolerance ledger exactness, cover transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalRefinementUp.

Chapter 6200

A Concrete Naming Certificate for $\mathsf{DifferentialPrivacy}^{\uparrow}$

Following Dwork and Roth on differential privacy, the $\mathsf{DifferentialPrivacy}^{\uparrow}$ packet records the finite randomized-mechanism surface through which a BEDC consumer may inspect adjacent inputs, paired output distributions, a rational privacy budget, privacy-loss comparisons, postprocessing, transport, replay, provenance, and local naming. It depends on the finite probability spaces of chapter 184, the random-variable rows of chapter 185, the distribution readback surface of chapter 186, and the rational budget discipline of chapter 37. The certificate is intentionally finite: it names only displayed rows and does not export measure-theoretic Radon–Nikodym derivatives, selected probability kernels, host function equality, or amplification theorems.

Definition 6200.1 (DifferentialPrivacy carrier)**.** *A* $\mathsf{DifferentialPrivacy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (A, M, O_0, O_1, E, L, F, H, C, P, N)$$

with the following displayed rows:

1. *A is the adjacent-input relation row, naming the two displayed input histories, their finite database boundary, and the local adjacency witness used by the privacy comparison.*
2. *M is the randomized mechanism graph row, listing the inspected finite graph of input names, output names, and rational transition weights for the requested mechanism surface.*
3. *O_0 and O_1 are the two* $\mathsf{Distribution}^{\uparrow}$ *output-distribution rows obtained by reading M at the adjacent inputs displayed by A.*
4. *E is the positive rational epsilon-budget row, recorded through the* $\mathsf{Rat}^{\uparrow}$ *comparison discipline and available only as the displayed privacy tolerance.*
5. *L is the privacy-loss comparison ledger, pairing every displayed output atom with the rational inequalities that compare the O_0 and O_1 masses against the budget E.*
6. *F is the postprocessing map row, naming the finite output relabeling or coarsening that may be applied after the mechanism has been read through O_0 and O_1.*
7. *H records componentwise* hsame *transport for $A, M, O_0, O_1, E, L, F, C, P, N$ and for the local certificate rows.*

8. *C records displayed* Cont *replay from an adjacent-input request through the mechanism graph, paired distributions, privacy-loss ledger, postprocessing row, transport, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{DifferentialPrivacy}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{RandomVar}^{\uparrow}$, $\mathsf{Distribution}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{DifferentialPrivacy}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient measurable space, a selected stochastic kernel, host equality of functions, Radon–Nikodym density, adaptive composition theorem, or privacy amplification principle.

Lean4 ✓ `BEDC.Derived.DifferentialPrivacyUp.TasteGate.DifferentialPrivacyTasteGate single carrier alignment`

Theorem 6200.2 (DifferentialPrivacy NameCert obligations)**.** *For every accepted* $\mathsf{DifferentialPrivacy}^{\uparrow}$ *carrier $D = (A, M, O_0, O_1, E, L, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DifferentialPrivacy}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed packet D;*
2. *adjacency exposure through A and no other neighboring-input relation;*
3. *paired distribution readback through O_0 and O_1 after reading the randomized mechanism graph M;*
4. *epsilon privacy-loss bounding through the positive rational budget E and the comparison ledger L;*
5. *postprocessing non-escape through the displayed map row F;*
6. *provenance exhaustion through componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local name row N.*

No consumer of the certificate receives a measure-theoretic density, an unlisted probability kernel, host function equality, adaptive composition, subsampling amplification, or an external privacy proof.

Proof. Carrier admission is projection from Definition 6200.1. The projection exposes exactly the adjacent-input row, mechanism graph, paired distribution rows, rational epsilon budget, privacy-loss ledger, postprocessing row, and structural certificate rows.

The adjacency and readback obligations are A, M, O_0, O_1. The row A names the only adjacent input pair, M is the only randomized mechanism graph, and O_0, O_1 are the only output distributions read from that graph for the adjacent inputs.

The privacy-loss obligation is E, L. The budget E is a positive rational row, and L records only the displayed output-atom comparisons against that budget, so the certificate cannot infer a density or an unlisted kernel.

Finally F, H, C, P, N preserve the same displayed packet under postprocessing, transport, replay, provenance, and local naming. Since the excluded analytic and semantic objects are absent from the carrier, these rows exhaust the local NameCert surface. □

Theorem 6200.3 (DifferentialPrivacy postprocessing stability)**.** *Let $D = (A, M, O_0, O_1, E, L, F, H, C, P, N)$ be an accepted* $\mathsf{DifferentialPrivacy}^{\uparrow}$ *carrier. Any consumer that applies the displayed postprocessing row F preserves the same epsilon privacy ledger E, L, because F reads the mechanism only through the paired output-distribution rows O_0, O_1 and structural transport through H, C, P, N.*

Proof. Read F from the carrier of Definition 6200.1. The postprocessing row is downstream of the mechanism graph and has access only to the output names already displayed in O_0 and O_1.

Apply Definition 6200.2. The epsilon budget and privacy-loss comparisons are fixed by E, L before postprocessing is inspected, and the structural rows H, C, P, N replay and source the same packet rather than adding a new probability comparison.

Thus every displayed postprocessing read preserves the same privacy ledger. A consumer may relabel or coarsen the displayed outputs through F, but it cannot replace E, L or strengthen the certificate into amplification, adaptive composition, selected kernel equality, or host measure theory. □

Closure status for $\mathsf{NameCert}_{\mathsf{DifferentialPrivacy}^{\uparrow}}$.

Constructive story: A $\mathsf{DifferentialPrivacy}^{\uparrow}$ packet is generated from adjacent-input rows, a randomized mechanism graph, paired DistributionUp output rows, a positive RatUp epsilon budget, a displayed privacy-loss ledger, a postprocessing map row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DifferentialPrivacy}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6200.1, Definition 6200.2, and Definition 6200.3 seed the finite differential-privacy route over ProbSpaceUp, RandomVarUp, DistributionUp, RatUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DifferentialPrivacy}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.DifferentialPrivacyUp.TasteGate.DifferentialPrivacyTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not claim measure-theoretic Radon–Nikodym machinery, selected probability kernels, host function equality, privacy amplification, adaptive composition, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, adjacency exposure, paired distribution readback, epsilon privacy-loss bounding, postprocessing non-escape, provenance exhaustion, and local naming for BEDC.Derived.DifferentialPrivacy

Chapter 6201

A Concrete Naming Certificate for $\mathsf{UniformCauchySubnet}^{\uparrow}$

Following Bishop-style constructive analysis, the $\mathsf{UniformCauchySubnet}^{\uparrow}$ packet records the finite modulus data by which a displayed regular sequence exposes a uniform subnet route for completion-facing consumers. It sits between the $\mathsf{CauchyModulus}^{\uparrow}$ tolerance surface, the finite-window discipline of chapter 126, the regular rational readback surface of chapter 301, the dyadic tolerance ledger of chapter 291, the real-name seal of chapter 38, and the located-limit route of chapter 1420. The packet names a finite subnet modulus for displayed requests; it is not a countable-choice subsequence, a host subnet object, quotient stream equality, or ambient completeness for $\mathsf{Real}^{\uparrow}$.

Definition 6201.1 (Uniform Cauchy subnet carrier)**.** *A* $\mathsf{UniformCauchySubnet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, M, W, D, R, E, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source regular sequence surface, read only through finite* $\mathsf{StreamName}^{\uparrow}$ *windows.*
2. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row assigning a displayed threshold to each dyadic precision request.*
3. *W is the subnet-window ledger; for each displayed request it records the finite source windows and the finite subnet index comparisons needed for the request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used by M and W.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the scheduled windows and tolerances.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after S, M, W, D, R have been read.*
7. *L is the* $\mathsf{LocatedLimit}^{\uparrow}$ *handoff row exposing the same finite schedule as located-limit data.*
8. *H records componentwise* hsame *transport for S, M, W, D, R, E, L.*
9. *C records displayed* Cont *replay for source-window inspection, modulus lookup, subnet-window projection, dyadic tolerance comparison, regular rational readback, Real sealing, and located-limit handoff.*
10. *P is the* Pkg *provenance over* $\mathsf{UniformCauchySubnet}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedLimit}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchySubnet}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $S, M, W, D, R, E, L, H, C, P, N$. It has no coordinate for countable choice, a selected infinite subsequence, a host subnet, quotient equality of streams, trichotomy, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.UniformCauchySubnetUp.TasteGate.UniformCauchySubnetTasteGate single carrier alignment`

Theorem 6201.2 (Uniform Cauchy subnet NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchySubnet}^{\uparrow}$ *carrier*

$$U = (S, M, W, D, R, E, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchySubnet}^{\uparrow}}$ *surface consists exactly of carrier admission for the source sequence, modulus row, subnet-window ledger, dyadic tolerance row, regular rational readback, Real seal, located-limit handoff, componentwise transport, displayed replay, package provenance, and local naming. Every accepted consumer read must inspect M and D before reading a subnet window in W, must inspect W before extracting R, and must pass through R before consuming E or L.*

Proof. Project the carrier of Definition 6201.1. The only rows that can answer a precision request before readback are the modulus row M, the dyadic tolerance row D, and the subnet-window ledger W.

The proof is by induction over the displayed subnet requests. The base request reads the first stored tolerance, uses M to locate the finite threshold, and then projects the matching source and subnet windows from W.

For the step, a stronger displayed request carries a refined dyadic tolerance in D. The same modulus row M supplies a later threshold, and W appends only the finite comparisons needed to show that the new subnet window lies beyond the previous one.

After each request has been projected, the readback row R is determined by the scheduled windows and dyadic data. The Real seal E and located-limit handoff L are downstream of that readback, while H, C, P, N transport, replay, source, and name the same finite route. Since the excluded choice, host-subnet, quotient, trichotomy, and completeness objects have no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6201.3 (Uniform Cauchy subnet RegSeqRat handoff)**.** *Every completion-facing read from an accepted* $\mathsf{UniformCauchySubnet}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{CauchyModulus}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{LocatedLimit}^{\uparrow}.$$

Concretely, the consumer first reads the source surface S, the modulus row M, the subnet-window ledger W, and the dyadic tolerance row D; it then obtains the $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and only afterward reaches the* $\mathsf{Real}^{\uparrow}$ *seal E or the* $\mathsf{LocatedLimit}^{\uparrow}$ *handoff row L.*

Proof. Start with the obligation package of Definition 6201.2. It forces every precision request to expose M, D, W before any regular rational readback is available.

Use finite-window projection on the displayed request. The stream windows in W are not chosen by an external subsequence principle; they are the windows explicitly stored with the modulus threshold supplied by M and the dyadic tolerance supplied by D.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is then the only readback row available to the consumer. The terminal rows E and L read that same R, and the structural rows H, C, P, N preserve the route under transport and replay. Thus the handoff reaches Real and located-limit consumers without exporting countable choice, a host subnet, quotient stream equality, trichotomy, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{UniformCauchySubnet}^{\uparrow}$ completion read displays a CauchyModulus row, finite StreamName windows, DyadicRatCore tolerances, a RegSeqRat readback, a Real seal, a LocatedLimit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert naming before it can be consumed by Real-completion or located-limit routes.

Independence witness: cauchymodulus, streamname, regseqrat, dyadicratcore, real, locatedlimit: $\mathsf{UniformCauchySubnet}^{\uparrow}$ is not bijective to any listed sibling because it binds a uniform precision-to-window modulus, subnet-window comparisons, regular readback, Real sealing, and located-limit handoff in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchySubnet}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchySubnet}^{\uparrow}$ packet is generated from a displayed source sequence, a CauchyModulus row, a subnet-window ledger, DyadicRatCore tolerances, a RegSeqRat readback, a Real seal, a LocatedLimit handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchySubnet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6201.1, Definition 6201.2, and Definition 6201.3 seed the finite uniform subnet modulus route for completion-facing and located-limit consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchySubnet}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.UniformCauchySubnetUp.TasteGate.UniformCauchySubnetTasteGate\_single\_carrier\_alignmen`

Bridge status: none

Not claimed: This chapter does not close countable choice, selected host subsequences, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus exposure, subnet-window projection, dyadic tolerance accounting, regular readback, Real seal handoff, located-limit handoff, transport, replay, provenance, and local naming for BEDC.Derived.UniformCauchySubnetUp.

Chapter 6202

A Concrete Naming Certificate for $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$

The $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ packet realizes the certified-discovery and positive-cost-protocol surface of section 9462.13 as finite classifier-shift, ledger, debt, and NameCert rows. It sits beside the classifier-shift witness of chapter 6138, the pipeline calculus of chapter 5914, the net-information packet of chapter 6181, and the effective replacement ledger of chapter 568. The packet classifies a discovery claim only after the changed classifier, cost protocol, debt profile, and local not-claimed boundary are all displayed.

Definition 6202.1 (Positive interpretability classifier carrier)**.** *A* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (E_0, E_1, J, \Delta, W, L, D, B, Q, H, C, R, N)$$

with the following displayed rows:

1. E_0 *is the source explanation-state row before the interpretability claim is classified.*
2. E_1 *is the target explanation-state row after the claim is classified.*
3. J *is the* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ *row exposing a common source and a changed classifier judgment.*
4. Δ *is the classifier-shift row read by the discovery pipeline calculus.*
5. W *is the public cost-protocol weight row for benefit, score, debt, source scope, target scope, and not-claimed boundary.*
6. L *is the effective replacement ledger row recording which classifier and explanation rows have real downstream use.*
7. D *is the explanation-debt row separating ledger, semantic, verification, intervention, and generalization debts.*
8. B *is the benefit row for coverage, prediction, transfer, intervention, verification, and unification gains.*
9. Q *is the* $\mathsf{ExplanationComplexityInformation}^{\uparrow}$ *net-information row comparing benefit against score and debt under* W*.*
10. H *records componentwise* hsame *transport for* $E_0, E_1, J, \Delta, W, L, D, B, Q$*.*

11. *C records displayed* Cont *replay through classifier-shift inspection, cost-protocol accounting, ledger lookup, debt exposure, benefit exposure, and net-information comparison.*
12. *R is the* Pkg *provenance over* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$, $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$, DiscoveryPipelineCalculu ExplanationComplexityInformation$^{\uparrow}$, $\mathsf{EffectiveReplacementLedger}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}}$ *row.*

The carrier has no coordinate for prose-only classifier novelty, ledger-only completion, raw score movement, hidden semantic truth, uncounted cost, undisplayed debt, or a positive discovery claim without the displayed classifier-shift row J, Δ and the cost-protocol row W.

Theorem 6202.2 (Positive interpretability classifier NameCert obligations)**.** *For every accepted* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ *carrier*

$$P = (E_0, E_1, J, \Delta, W, L, D, B, Q, H, C, R, N),$$

the local $\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}}$ *surface consists exactly of carrier admission for the two explanation states, classifier-novelty witness, classifier-shift row, public cost protocol, effective replacement ledger, debt row, benefit row, net-information row, componentwise transport, displayed replay, package provenance, and local naming. A positive interpretability classification is accepted only when J, Δ display classifier novelty and Q reads B, D, W, L under the same scope.*

Proof. Project the carrier of Definition 6202.1. The classifier-shift surface is not implicit: the rows J and Δ display the common source, changed judgment, and pipeline-facing shift.

Next read the cost protocol. The weight row W fixes the scope and the benefit-score-debt accounting convention, while L, D, B expose replacement use, debt, and benefit inside that scope.

The row Q can classify the claim as positive only after those rows have been read together. This separates a true classifier shift with positive net information from ledger-only completion, formal hardening, raw score motion, or hidden semantic claims.

Finally, H, C, R, N transport, replay, source, and name the same finite packet. Since none of the excluded objects appears in the carrier, the NameCert obligations are exactly the listed rows. □

Theorem 6202.3 (Positive interpretability falsifiable shift)**.** *An accepted* $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ *consumer is refuted if it classifies a claim as positive interpretability discovery while any one of the classifier-shift rows J, Δ, the public cost-protocol row W, the replacement ledger row L, the debt row D, the benefit row B, or the net-information row Q is absent. Conversely, a ledger-only or formal-hardening improvement with unchanged classifier rows is not accepted by this classifier.*

Proof. Apply Definition 6202.2. The obligations place classifier novelty and net-information accounting in the same carrier, so a consumer cannot obtain the positive classification from one side alone.

If J or Δ is absent, the carrier has no displayed classifier shift. If W, L, D, B, or Q is absent, the carrier has no complete positive cost-protocol read.

The remaining rows H, C, R, N preserve the same classification route under transport, replay, provenance, and local naming. Therefore a claim missing any named row is rejected, and a classifier-unchanged improvement is classified outside positive interpretability discovery. □

Falsifiable prediction: Every accepted $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ positive-discovery read displays two explanation states, a classifier novelty witness, a classifier-shift row, a public cost

protocol, effective replacement ledger rows, debt and benefit rows, net explanation information, transport, replay, provenance, and local NameCert naming.

Independence witness: classifier-novelty-witness, explanation-complexity-information, discovery-pipeline-calculus, effective-replacement-ledger: none is bijective to $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ because this packet jointly carries the before-and-after explanation states, classifier-shift witness, public cost protocol, replacement ledger, debt accounting, benefit accounting, net-information comparison, transport, replay, provenance, and local naming as one classifier for positive interpretability discovery.

The Lean follow-up must provide $\mathsf{Nontrivial\ PositiveInterpretabilityClassifier}^{\uparrow}$ and $\mathsf{FieldFaithful\ PositiveInterpretabil}$ instances before checked markers for BEDC.Derived.PositiveInterpretabilityClassifierUp are cited. Those instances must preserve the rows $E_0, E_1, J, \Delta, W, L, D, B, Q, H, C, R, N$ so classifier novelty, ledger evidence, debt accounting, and positive net information remain separate fields.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}$ packet is generated from two explanation states, a classifier novelty witness, a classifier-shift row, public cost protocol weights, an effective replacement ledger, debt and benefit rows, net explanation information, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6202.1, Definition 6202.2, and Definition 6202.3 seed the finite classifier for positive interpretability discovery claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global interpretability quality, hidden semantic truth, prose-only classifier novelty, uncounted costs, undisplayed debts, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, classifier-shift exactness, cost-protocol exposure, ledger and debt accounting, net-information comparison, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.PositiveInterpretabilityClassifierUp.

Chapter 6203

A Concrete Naming Certificate for $\mathsf{RegularCauchyFilterReal}^{\uparrow}$

Following Bishop and Bridges, a real completion interface may be presented by regular filters only when each tolerance read remains finite, rounded, and tied to an explicit rational window. The $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ packet sits between the finite stream windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the filter surface of chapter 333. It names a selection-free regular Cauchy filter route to a displayed real seal; it is not quotient filter equality, a chosen representative, a selected limit, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6203.1 (Regular Cauchy filter real carrier)**.** *A* $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, S, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *row, carrying the rounded filter base and its finite tolerance requests.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row inspected at the displayed filter tolerance.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite windows.*
4. *D is the dyadic or rational diameter ledger proving that the inspected filter windows are regular at the requested precision.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, S, Q, D have been displayed.*
6. *H records componentwise* hsame *transport for the filter, window, readback, diameter, real-seal, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay from a tolerance request through the filter base, finite windows, regular rational readback, diameter ledger, and real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyFilterReal}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these nine rows. It has no coordinate for quotient filters, selected subfilters, chosen representatives, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RegularCauchyFilterRealUp.RegularCauchyFilterRealTasteGate single carrier alignment`

Theorem 6203.2 (Regular Cauchy filter real NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ *carrier* $R = (F, S, Q, D, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterReal}^{\uparrow}}$ *surface consists exactly of carrier admission, rounded CauchyFilter admission, StreamName finite-window admission, RegSeqRat readback, regular diameter verification, Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the displayed rows* $F, S, Q, D, E, H, C, P, N$*.*

Proof. Project the accepted carrier of Definition 6203.1. The analytic rows are precisely the rounded filter row F, the finite stream window row S, the regular rational readback Q, the diameter ledger D, and the terminal real seal E.

The local certificate reads these rows in order. A filter tolerance request can reach a regular rational readback only through the finite window row, and the Real seal can be inspected only after the diameter ledger has witnessed the displayed regularity at the same request.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since they add no quotient, selector, chosen representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the local naming obligations. □

Theorem 6203.3 (Regular Cauchy filter real completion route)**.** *Every completion-facing read of an accepted* $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{CauchyFilter}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer reads F*, then the finite windows* S*, then the regular rational readback* Q *and diameter ledger* D*, and finally the Real seal* E*, with* H, C, P, N *preserving the same route. The route supplies a filter-form completion boundary without quotient filter equality, selected subfilters, selected limits, chosen representatives, countable choice, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the obligation surface of Definition 6203.2. It exposes the rounded CauchyFilter row before any finite window or regular rational readback is available.

The StreamName and RegSeqRat rows supply the finite observation route. The diameter ledger is attached to that same observation, so a completion-facing consumer cannot replace it by a quotient filter, a selected subfilter, or a chosen representative.

Finally the Real seal is downstream of the displayed readback. Transport, replay, provenance, and local naming preserve the same packet, and the carrier contains no coordinate for selected limits, countable choice, host real equality, or ambient completeness. Hence every completion-facing read factors through the stated finite route. □

Falsifiable prediction: A $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ consumer that exports a selected limit, quotient filter equality, countable-choice schedule, host real equality, or ambient Real completeness after deleting F, S, Q, or D refutes the carrier of Definition 6203.1.

Independence witness: Nearest siblings: StreamName, RegSeqRat, Real, and CauchyFilter. $\mathsf{RegularCauchyFilterReal}^{\uparrow}$ is not bijective to any one of them because it jointly carries rounded filter admission, finite windows, regular rational readback, a diameter ledger, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyFilterReal}^{\uparrow}}$**.**

Constructive story: A RegularCauchyFilterReal$^{\uparrow}$ packet is generated from rounded CauchyFilter rows, StreamName finite windows, RegSeqRat readbacks, regular diameter ledgers, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyFilterReal}^{\uparrow}}$, seed)

Scope closed: Definition 6203.1, Definition 6203.2, and Definition 6203.3 seed the finite filter-form route from rounded CauchyFilter requests to terminal Real reads.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyFilterReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient filter equality, selected subfilters, selected limits, chosen representatives, countable choice, host real equality, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rounded filter exactness, StreamName window replay, RegSeqRat readback, diameter-ledger nonescape, Real seal admission, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyFilterRealUp.

Chapter 6204

A Concrete Naming Certificate for $\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}$

Following Bishop and Bridges on constructive intervals and finite nets, the $\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}$ packet records the explicit dyadic mesh by which a closed real interval is read as totally bounded before any open-cover compactness claim is available. It sits after the dyadic arithmetic of chapter 291, the finite window schedule of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the interval-net surface of chapter 1250, and the finite-net interface of chapter 127. The packet is a choice-free finite mesh certificate for the L10 real-and-completion route; it is not open-cover compactness, selected subcovers, countable choice, quotient interval equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6204.1 (Dyadic totally bounded interval carrier)**.** *A* $\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (L, U, M, Z, \rho, F, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L and U are the lower and upper endpoint rows, each read through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint interface and sealed only later as* $\mathsf{Real}^{\uparrow}$ *data;*
2. *M is the unary mesh-scale row naming the dyadic precision used for the interval net;*
3. *Z is the finite center list, carried as a closed* $\mathsf{ProbeBundle}$ *of dyadic center rows between the displayed endpoints;*
4. *ρ is the dyadic radius ledger assigning the mesh radius used by the centers of Z;*
5. *F is the finite coverage row saying that every interval read enters one displayed center of Z within the dyadic radius ρ;*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect endpoint, center, and radius names at the mesh scale;*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row assembled from the endpoint, center, and radius windows;*
8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after $L, U, M, Z, \rho, F, S, R$ are displayed;*
9. *H stores componentwise* hsame *transport for endpoints, mesh, centers, radii, coverage rows, windows, readback, seal, provenance, and local name;*

10. *C stores the displayed* Cont *replay from endpoints to mesh selection, from mesh to finite centers and radii, from coverage to window readback, and from readback to the real seal;*
11. *P records* Pkg *provenance over* $\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}}$ *row.*

The carrier has no coordinate for an open cover, a selected finite subcover, countable choice, quotient interval equality, real trichotomy, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.DyadicTotallyBoundedIntervalUp.DyadicTotallyBoundedIntervalTasteGate single carrier alignment`

Theorem 6204.2 (Dyadic totally bounded interval NameCert obligations)**.** *For the carrier of Definition 6204.1, the local* $\mathsf{NameCert}_{\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}}$ *surface consists exactly of carrier admission for $I = (L, U, M, Z, \rho, F, S, R, E, H, C, P, N)$, finite-net coverage through Z, ρ, F, endpoint containment for L and U, mesh-radius compatibility between M and ρ,* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through S and R,* $\mathsf{Real}^{\uparrow}$ *seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.DyadicTotallyBoundedIntervalUp.DyadicTotallyBoundedIntervalCarrier namecert obligations`

Proof. Project the carrier. The endpoint rows L, U, mesh row M, finite center bundle Z, dyadic radius ledger ρ, and coverage row F are displayed before any stream, regular-Cauchy, or real seal row is available.

The finite-net obligations are read from those rows. Endpoint containment uses the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint rows, mesh-radius compatibility ties M to ρ, and the coverage row F routes each interval read to one center of Z at that radius. This is the same kind of finite mesh handoff consumed by the $\mathsf{ClosedBoundedIntervalNet}^{\uparrow}$ and $\mathsf{TotallyBounded}^{\uparrow}$ sibling surfaces.

The readback and seal obligations are downstream. The $\mathsf{StreamName}^{\uparrow}$ row S and $\mathsf{RegSeqRat}^{\uparrow}$ row R expose only the displayed endpoint, center, and radius windows, and the $\mathsf{Real}^{\uparrow}$ seal E is reached only after that readback. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so the excluded compactness, choice, equality, trichotomy, and completeness principles have no carrier coordinate. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}$ packet is generated from dyadic endpoint rows, a unary dyadic mesh scale, a finite center ProbeBundle, a dyadic radius ledger, finite coverage rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}}$, seed)

Scope closed: Definition 6204.1 and Definition 6204.2 name the finite carrier and its local obligation surface for the dyadic totally bounded interval route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicTotallyBoundedInterval}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.DyadicTotallyBoundedIntervalUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, quotient interval equality, trichotomy, ambient Real completeness, Lean verification, or a public standard bridge.

Upgrade path: Scoped closure requires checked endpoint containment, finite-net coverage, mesh-radius compatibility, readback handoff, Real seal nonescape, transport stability, replay, provenance, and local naming for the same finite packet.

Chapter 6205

A Concrete Naming Certificate for $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$

Following Hardy and Knopp, the $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$ packet records the finite BEDC surface for a sequence whose next observation is generated from an initial value, a fixed multiplier, and an inhomogeneous geometric row. It reads the finite stream windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names the recurrence evidence used by downstream $\mathsf{Real}^{\uparrow}$ sequence calculus; it does not assert a closed-form host equality, selected infinite stream, quotient sequence identity, countable choice, or ambient real completeness.

Definition 6205.1 (Arithmetico-geometric sequence carrier)**.** *An* $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (I, M, G, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the initial-value row, recording the first named rational or real-facing approximation from which the recurrence is read.*
2. *M is the multiplier row, displaying the fixed scalar used by each recurrence step.*
3. *G is the inhomogeneous geometric row, carrying the finite geometric contribution read at the displayed step.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, listing the finite sequence windows and indices exposed to a consumer.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, converting the displayed windows into regular rational evidence before any real seal is consumed.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after the initial, multiplier, geometric, stream-window, and regular-readback rows have been displayed.*
7. *H records componentwise* hsame *transport for the initial value, multiplier, inhomogeneous geometric row, stream observations, regular readback, real seal, replay, provenance, and local name.*
8. *C records displayed* Cont *replay from a finite index and tolerance request through the same recurrence rows.*

9. *P is the* Pkg *provenance over* $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{ArithmeticoGeometricSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, M, G, W, R, E, H, C, P, N$. It has no coordinate for a completed infinite sequence object, host closed-form equality, quotient stream equality, selected limit, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.ArithmeticoGeometricSequenceUp.ArithmeticoGeometricSequenceTasteGate single carrier alignment`

Theorem 6205.2 (Arithmetico-geometric sequence NameCert obligations)**.** *For every accepted* $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$ *carrier*

$$A = (I, M, G, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ArithmeticoGeometricSequence}^{\uparrow}}$ *surface consists exactly of carrier admission for the initial value, fixed multiplier, inhomogeneous geometric row, StreamName observations, RegSeqRat readback, Real seal, componentwise transport, displayed replay, provenance, and local naming; recurrence visibility requiring every displayed step to read I, M, G, W before R; seal discipline requiring every* $\mathsf{Real}^{\uparrow}$ *consumer to pass through R before E; and nonescape excluding host closed forms, quotient stream equality, selected limits, countable choice, and ambient real completeness.*

Proof. Carrier admission is projection from Definition 6205.1. The accepted packet displays exactly the rows $I, M, G, W, R, E, H, C, P, N$.

The recurrence read starts with I, M, G, W. The initial-value row fixes the displayed starting observation, the multiplier row records the scalar reused by the finite recurrence route, the inhomogeneous geometric row supplies the finite forcing term, and the StreamName row lists the inspected sequence windows.

The regular readback and seal rows are downstream. A consumer must pass through R before it can read the $\mathsf{Real}^{\uparrow}$ seal E, and componentwise hsame transport in H together with displayed Cont replay in C preserves the same recurrence route.

Finally, P and N only source and name the displayed packet. Since the carrier has no coordinate for a completed infinite stream, a host closed-form equality, quotient sequence identity, countable choice, selected limits, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{ArithmeticoGeometricSequence}^{\uparrow}}$**.**

Constructive story: An $\mathsf{ArithmeticoGeometricSequence}^{\uparrow}$ packet is generated from an initial-value row, a fixed multiplier row, an inhomogeneous geometric row, finite StreamName observations, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ArithmeticoGeometricSequence}^{\uparrow}}$, seed)

Scope closed: Definition 6205.1 and Definition 6205.2 seed the finite recurrence route from initial value and inhomogeneous geometric evidence through StreamName windows, RegSeqRat readback, and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ArithmeticoGeometricSequence}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host closed-form equality, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recurrence visibility, StreamName observation coverage, RegSeqRat readback, Real seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.ArithmeticoGeometric

Chapter 6206

A Concrete Naming Certificate for InverseFunctionTheorem$^{\uparrow}$

Following Cartan, Rudin, and Bishop–Bridges, the InverseFunctionTheorem$^{\uparrow}$ packet records the theorem-level local inverse route exposed after a differentiable map, an invertible derivative row, a quantitative neighbourhood ledger, and the Real completion route have all been displayed. It sits above the local carrier boundary of chapter 2055, consumes the Newton–Kantorovich theorem route of chapter 1785 and the graph-window route of chapter 1834, and reaches observable endpoint data only through chapter 126, chapter 301, and chapter 38. The packet names a finite BEDC route from displayed analytic hypotheses to a local inverse theorem certificate; it does not assert a host inverse theorem, select a germ, identify quotient function spaces, import ambient completeness, or provide bridge verification.

Definition 6206.1 (Inverse function theorem carrier)**.** *An* InverseFunctionTheorem$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (F, U, D, L, K, G, W, S, Q, R, H, C, P, N)$$

with the following rows:

1. *F is the displayed map row together with the source and target classifier windows.*
2. *U is the base-point and neighbourhood row, naming the point, value, source radius, target radius, and local domain of the theorem read.*
3. *D is the derivative-control row over the displayed neighbourhood.*
4. *L is the invertible-linear row for the derivative at the base point, including the finite inverse witness used by the quantitative route.*
5. *K is the* NewtonKantorovich$^{\uparrow}$ *row that supplies the contraction and root-existence certificate for nearby target values.*
6. *G is the* ImplicitFunction$^{\uparrow}$ *graph-window row for the equation $F(x) - y = 0$.*
7. *W is the* InverseFunction$^{\uparrow}$ *local inverse-window row obtained from the same analytic and solving data.*
8. *S is the* StreamName$^{\uparrow}$ *finite observation row for source, target, and endpoint windows.*
9. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the rational approximation data exposed by S.*
10. *R is the terminal* Real$^{\uparrow}$ *seal row for the theorem-facing completion route.*

11. *H records componentwise* hsame *transport for all displayed analytic, solving, completion, replay, provenance, and naming rows.*
12. *C records displayed* Cont *replay through*

$$F, U, D, L, K, G, W, S, Q, R \longmapsto \mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}.$$

13. *P is the* Pkg *provenance over* InverseFunctionTheorem$^{\uparrow}$, InverseFunction$^{\uparrow}$, NewtonKantorovich$^{\uparrow}$, ImplicitFunction$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, U, D, L, K, G, W, S, Q, R, H, C, P, N$. The carrier has no coordinate for a host theorem schema, selected inverse germ, quotient function-space equality, global bijection theorem, ambient Real completeness principle, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.InverseFunctionTheoremUp.InverseFunctionTheoremTasteGate single carrier alignment`

Theorem 6206.2 (Inverse function theorem NameCert obligations)**.** *For every accepted* InverseFunctionTheorem$^{\uparrow}$ *carrier*

$$T = (F, U, D, L, K, G, W, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for T, derivative-control exposure through D, invertible-linear exactness through L, Newton–Kantorovich solving through K, implicit graph-window handoff through G, local inverse-window handoff through W, stream observation through S, regular rational readback through Q, terminal Real sealing through R, componentwise* hsame *stability through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6206.1. The analytic rows F, U, D, L are the only source of map, neighbourhood, derivative, and invertibility data.

The solving rows are then forced. The Newton–Kantorovich row K supplies the finite contraction route, the implicit-function row G repacks the same equation as a graph window, and the inverse-function row W exposes the resulting local inverse window.

The completion-facing rows come last. The stream row S provides finite observations, Q reads them as regular rational data, and R seals the theorem-facing Real endpoint. The structural rows H, C, P, N transport, replay, source, and name exactly this route, so no additional theorem schema, selected germ, quotient equality, global bijection, or ambient completeness coordinate is present. □

Theorem 6206.3 (Inverse function theorem Real completion route)**.** *Every accepted* InverseFunctionTheorem$^{\uparrow}$ *theorem read exported as a local inverse statement factors through the finite route*

$$\mathsf{NewtonKantorovich}^{\uparrow} \longmapsto \mathsf{ImplicitFunction}^{\uparrow} \longmapsto \mathsf{InverseFunction}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by K, G, W, S, Q, R and the analytic rows F, U, D, L. The Real-facing conclusion receives local window data, derivative and inverse-witness data, theorem-level solving data, finite observations, regular rational readback, terminal Real sealing, componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row; it receives no host inverse theorem, global inverse, selected germ, quotient function-space equality, or ambient completeness shortcut.*

Proof. Start with the accepted carrier $T = (F, U, D, L, K, G, W, S, Q, R, H, C, P, N)$. The rows F, U, D, L display the map, neighbourhood, derivative control, and invertible-linear witness before any theorem conclusion is read.

Apply the theorem-level solving route. The row K gives the Newton–Kantorovich certificate, G reads the same local equation as an implicit graph window, and W opens the local inverse-window carrier from chapter 2055.

Finally pass through the completion route. The stream, regular-rational, and Real rows S, Q, R are read in that order, and H, C, P, N preserve the same displayed path. Since no carrier coordinate stores a host inverse theorem, global inverse, selected germ, quotient function-space equality, or ambient completeness shortcut, the route is exhausted by the finite BEDC packet. □

Falsifiable prediction: Every $\mathsf{InverseFunctionTheorem}^{\uparrow}$ theorem-facing read factors through one displayed map row, one neighbourhood row, one derivative-control row, one invertible-linear row, one NewtonKantorovich row, one ImplicitFunction row, one InverseFunction row, one StreamName finite-observation row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: inverse-function, newton-kantorovich, implicitfunction, streamname, regseqrat: $\mathsf{InverseFunctionTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries theorem-level analytic hypotheses, Newton solving, implicit graph-window handoff, local inverse-window export, finite observation, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{InverseFunctionTheorem}^{\uparrow}$ packet is generated from a displayed map row, base neighbourhood row, derivative-control row, invertible-linear row, NewtonKantorovich theorem row, ImplicitFunction graph-window row, InverseFunction local-window row, StreamName observations, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6206.1, Definition 6206.2, and Definition 6206.3 seed the finite theorem-level local inverse route over analytic, solving, inverse-window, observation, regular rational, and Real completion rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InverseFunctionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host inverse theorems, global inverses, selected inverse germs, quotient function spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-control exposure, invertible-linear exactness, Newton solving, implicit graph-window handoff, local inverse-window handoff, Real completion routing, provenance exactness, and local naming for BEDC.Derived.InverseFunctionTheoremUp.

Chapter 6207

A Concrete Naming Certificate for CompactIntervalUniformModulus$^{\uparrow}$

Following Bishop and Bridges, the CompactIntervalUniformModulus$^{\uparrow}$ packet records the finite-net route by which a compact interval consumer reads a uniform modulus claim before any open-cover or selected-subcover principle is available. It sits after the dyadic arithmetic core of chapter 291, the finite dyadic interval nets of chapter 5966, the StreamName windows of chapter 126, the regular rational readback of chapter 301, and the terminal Real seal of chapter 38. The packet gives later compactness and uniform-continuity arguments a finite landing surface: a modulus is named only when the required dyadic mesh, endpoint reads, regular rational tolerances, and Real-facing seal rows have all been displayed.

Definition 6207.1 (Compact interval uniform modulus carrier)**.** *A* CompactIntervalUniformModulus$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, J, E, S, R, U, Q, H, C, P, N)$$

with the following rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *endpoint and precision row. It records dyadic endpoints, mesh widths, successor comparisons, and the arithmetic needed to compare a requested tolerance with a displayed mesh.*
2. *J is the* DyadicIntervalNet$^{\uparrow}$ *row. It lists the finite dyadic interval cells covering the named compact interval by displayed endpoint incidence.*
3. *E is the local endpoint-boundary row. It records that the first and last cells of J bracket the compact interval endpoints named through D.*
4. *S is the* StreamName$^{\uparrow}$ *window row read when a function value or endpoint name is inspected on a listed cell.*
5. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the rational values and tolerance schedule attached to those windows.*
6. *U is the finite modulus row. It records that a requested output tolerance is answered by one displayed dyadic mesh bound and by the finite cell checks listed in J.*
7. *Q is the terminal* Real$^{\uparrow}$ *seal row reached only after D, J, E, S, R, U have been displayed.*
8. *H records componentwise* hsame *transport for the displayed rows.*

9. *C records displayed* Cont *replay of the finite route*

$$D, J, E \longmapsto S, R \longmapsto U \longmapsto Q, H, P, N.$$

10. *P is the* Pkg *provenance over* CompactIntervalUniformModulus$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, DyadicIntervalNet$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{CompactIntervalUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, J, E, S, R, U, Q, H, C, P, N$. It has no coordinate for open-cover compactness, selected finite subcovers, real trichotomy, quotient equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.CompactIntervalUniformModulusUp.CompactIntervalUniformModulusTasteGate`
`single carrier alignment`

Theorem 6207.2 (Compact interval uniform modulus obligations)**.** *For every accepted* CompactIntervalUniformMo *carrier $M = (D, J, E, S, R, U, Q, H, C, P, N)$, the local* NameCert$_{\text{CompactIntervalUniformModulus}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *carrier admission for the dyadic endpoint row, dyadic interval-net row, endpoint-boundary row, StreamName window row, RegSeqRat readback row, finite modulus row, Real seal row, transport row, replay row, provenance row, and local naming row;*

2. *dyadic mesh exactness, requiring every modulus request to use a mesh width and endpoint comparison displayed in D;*

3. *finite-net coverage, requiring every interval read to pass through a listed cell of J and its endpoint incidence in E;*

4. *StreamName and RegSeqRat readback routing, requiring every inspected value to be exposed through S and then read through R before U may answer the tolerance request;*

5. *Real seal nonescape, requiring Q to be consumed only after the finite-net, readback, and modulus rows are displayed;*

6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation exports open-cover compactness, selected finite subcovers, real trichotomy, quotient equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Proof. Project the carrier of Definition 6207.1. The projection exposes the dyadic arithmetic row, the finite interval-net row, the endpoint-boundary row, the StreamName and RegSeqRat rows, the finite modulus row, the Real seal row, and the structural support rows.

The mesh and coverage obligations are finite. A modulus request first selects a displayed dyadic mesh comparison from D, then checks the listed cells of J and the bracketing incidences of E. No step asks for an arbitrary open cover or a selected subcover.

The value-facing obligations are routed through finite observations. Function and endpoint reads pass through S, rational readback passes through R, and only then does U answer the requested tolerance by the displayed mesh bound.

Finally Q, H, C, P, N seal, transport, replay, source, and name the same finite route. Since the packet has no row for quotient equality, selected limits, countable choice, trichotomy, ambient completeness, Lean verification, or bridge checking, the listed obligations exhaust the local NameCert surface. □

Theorem 6207.3 (Compact interval uniform modulus finite-net handoff)**.** *Every compact-interval uniform-continuity consumer of an accepted* $\mathsf{CompactIntervalUniformModulus}^{\uparrow}$ *carrier* $M = (D, J, E, S, R, U, Q, H, C$ *factors through the finite route*

$$D, J, E \longmapsto S, R \longmapsto U \longmapsto Q, H, C, P, N.$$

The consumer first reads dyadic endpoint and mesh arithmetic from $\mathsf{DyadicRatCore}^{\uparrow}$*, then the listed finite cells from* $\mathsf{DyadicIntervalNet}^{\uparrow}$*, then the finite windows and regular rational readback from* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal. The handoff exports a named finite modulus surface, not open-cover compactness, selected finite subcovers, real trichotomy, quotient equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.*

Proof. Start with the finite-net rows of the carrier. The rows D, J, E provide the mesh width, listed interval cells, and endpoint incidence needed by a compact-interval consumer before any value read occurs.

Next route every inspected value through S and R. The StreamName row supplies only finite windows, while the RegSeqRat row supplies the regular rational readback attached to those windows.

The modulus row U can then answer the consumer by the displayed mesh and cell checks. The answer remains a finite NameCert route because it is replayed by C, transported by H, sourced by P, and named by N before the Real seal Q is consumed.

The carrier contains no open-cover, selected-subcover, quotient, selected-limit, choice, trichotomy, completeness, Lean-verification, or bridge-checking coordinate. Thus every consumer receives exactly the finite-net handoff displayed above. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactIntervalUniformModulus}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{CompactIntervalUniformModulus}^{\uparrow}$ packet is generated from dyadic endpoint arithmetic, finite dyadic interval nets, endpoint bracketing, StreamName windows, RegSeqRat readback, a finite modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactIntervalUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6207.1, Definition 6207.2, and Definition 6207.3 seed the finite-net uniform-modulus carrier over dyadic endpoint arithmetic, listed interval cells, endpoint incidence, StreamName windows, RegSeqRat readback, Real seals, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactIntervalUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected finite subcovers, real trichotomy, quotient equality, selected limits, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic mesh exactness, finite-net coverage, StreamName and RegSeqRat readback routing, finite modulus response, Real seal ordering, provenance, and local naming for BEDC.Derived.CompactIntervalUniformModulusUp.

Chapter 6208

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailGauge}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ packet records the finite tail-gauge row by which a regular rational Cauchy name exposes an explicit precision rate before a $\mathsf{Real}^{\uparrow}$ consumer reads the seal. It sits after the finite window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the rate-extraction sibling of chapter 2085, and the terminal real-name seal of chapter 38. The packet names the displayed tail gauge itself; it does not assert quotient stream equality, host Cauchy completeness, selected moduli, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6208.1 (Regular Cauchy tail-gauge carrier)**.** *A* $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (W, R, D, G, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the inspected windows, observation indices, and point cells used by the tail request.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic precision request, comparison budget, and endpoint cells used by R.*
4. *G is the explicit tail-gauge row, assigning each displayed precision request to the finite window after which the R readback is certified within the D tolerance.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, R, D, and G have been displayed.*
6. *H records componentwise* hsame *transport for windows, readbacks, tolerance rows, tail gauges, Real seals, replay rows, provenance rows, and local naming rows.*
7. *C records displayed* Cont *replay from a tail request through window inspection, regular readback, dyadic tolerance inspection, gauge selection, Real-seal consumption, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailGauge}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailGauge}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, G, E, H, C, P, N$. It has no coordinate for quotient streams, selected limits, host Cauchy completeness, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6208.2 (Regular Cauchy tail-gauge NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ *carrier*

$$T = (W, R, D, G, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailGauge}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{StreamName}^{\uparrow}$ *window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the explicit tail-gauge row, the* $\mathsf{Real}^{\uparrow}$ *seal row, componentwise transport, displayed replay, provenance, and local naming. No tail-gauge obligation is stored outside those finite rows.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailGaugeUp.RegularCauchyTailGaugeTasteGate single carrier alignment`

Proof. Project the accepted carrier of Definition 6208.1. The analytic rows are W, R, D, G, E, and the structural rows are H, C, P, N.

Read the window and regular-rational rows first. The $\mathsf{StreamName}^{\uparrow}$ row fixes the finite observations available to the request, and the $\mathsf{RegSeqRat}^{\uparrow}$ row attaches the rational readback to those observations.

The dyadic ledger and gauge row then provide the tolerance accounting. The row D records the displayed precision budget, and G records the finite tail stage after which that budget is met.

Finally the seal and structural rows package the same route. Since the carrier has no coordinate for quotient streams, selected limits, host completeness, countable choice, trichotomy, or ambient Real completeness, none can become a local NameCert obligation. □

Theorem 6208.3 (Regular Cauchy tail-gauge Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ *packet must first read the finite route*

$$W \longmapsto R \longmapsto D \longmapsto G \longmapsto E$$

together with the transport, replay, provenance, and local naming rows H, C, P, N. The handoff is therefore an explicit finite tail gauge over StreamName, RegSeqRat, and DyadicRatCore data, not a host completeness argument or a selected limit.

Proof. Begin with a Real-facing read. Carrier admission exposes W before any regular rational or Real-facing row can be consumed, so the read starts from displayed finite windows.

Apply the readback and tolerance rows. The row R supplies the regular rational observation for those windows, while D supplies the dyadic precision ledger used by the request.

Now use the tail-gauge row G. It is the only row that names the finite stage at which the requested tolerance is reached, and it is stored before the Real seal E is exposed.

The rows H, C, P, N preserve and name the same dependency order. Thus the Real-facing consumer receives a finite tail-gauge route rather than quotient stream equality, a selected modulus outside the packet, or ambient Real completeness. □

Theorem 6208.4 (Regular Cauchy tail-gauge window determinacy)**.** *If two accepted* $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ *packets agree componentwise on their* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance rows through the displayed* hsame *transport, then their explicit tail-gauge rows name the same finite tail stage for every displayed precision request before either* $\mathsf{Real}^{\uparrow}$ *seal is read.*

Proof. Project the two accepted carriers and compare their W, R, D rows. Agreement on W fixes the finite observations, agreement on R fixes the regular rational readback, and agreement on D fixes the dyadic precision ledger.

Read the gauge row next. Since G is attached only to those displayed windows, readbacks, and tolerance rows, a matching W, R, D route leaves no second tail stage for the classifier to choose.

Finally inspect the seal boundary. The $\mathsf{Real}^{\uparrow}$ rows are downstream of G, and the structural rows transport only the displayed packet. Hence window, readback, and tolerance agreement determines the tail gauge before the Real seal is available. □

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailGauge}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailGauge}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6208.1, Definition 6208.2, Definition 6208.3, and Definition 6208.4 seed the finite tail-gauge route from StreamName windows through RegSeqRat readback, DyadicRatCore tolerance, explicit gauge rows, and the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailGauge}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, host Cauchy completeness, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, window-readback exactness, dyadic tolerance compatibility, gauge determinacy, Real-seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailGaug

Constructive story: A $\mathsf{RegularCauchyTailGauge}^{\uparrow}$ packet is generated from StreamName finite windows, RegSeqRat readbacks, DyadicRatCore tolerance ledgers, an explicit tail-gauge row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6209

A Concrete Naming Certificate for $\mathsf{PoincareDiskMetric}^{\uparrow}$

The $\mathsf{PoincareDiskMetric}^{\uparrow}$ packet realizes the displayed disk-map, metric-read, boundary, provenance, and refusal rows of chapter 9463 as finite NameCert data. It sits after the disk carrier of chapter 377, the phase-transport route of chapter 3455, the metric interface of chapter 57, the circle boundary source of chapter 58, and the terminal real-name seal of chapter 38. The packet names a metric-reading route on displayed disk rows; it is not a completed hyperbolic plane, quotient ideal boundary, selected geodesic, host analytic distance, or global compactification.

Definition 6209.1 (Poincare disk metric carrier)**.** *A* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (U, T, Q, R, S, B, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row for the finite source and target disk observations.*
2. *T is the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *route carrying disk-surface transport and phase provenance between displayed observations.*
3. *Q is the* $\mathsf{Met}^{\uparrow}$ *read row, recording the finite metric comparison requested for the displayed disk observations.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row attached to the metric readback.*
5. *S is the* $\mathsf{S}^{1\uparrow}$ *boundary row used when the metric read approaches or cites the visible boundary source.*
6. *B records boundary refusal data: no interior disk point, ideal boundary quotient, selected global endpoint, or completed hyperbolic plane is exposed by the packet.*
7. *H records componentwise* hsame *transport for disk rows, phase transport, metric reads, Real seals, boundary rows, refusal rows, replay rows, provenance rows, and local naming rows.*
8. *C records displayed* Cont *replay from a metric request through disk-coordinate inspection, phase transport, metric read, Real-seal consumption, boundary read, refusal, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{PoincareDiskMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $U, T, Q, R, S, B, H, C, P, N$. It has no coordinate for a host analytic metric, a completed disk, quotient ideal-boundary equality, selected geodesics, physical horizons, or global compactification.

Theorem 6209.2 (Poincare disk metric NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *carrier*

$$M = (U, T, Q, R, S, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareDiskMetric}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, phase-transport admission, metric-read admission, Real-seal boundary,* $\mathsf{S}^{1\uparrow}$ *boundary readback, refusal rows, componentwise transport, displayed replay, provenance, and local naming. No metric obligation is stored outside those finite rows.*

Proof. Project the accepted carrier of Definition 6209.1. The visible geometric rows are U, T, Q, R, S, B, and the structural rows are H, C, P, N.

Read disk coordinates and phase transport first. The metric row Q is available only after the displayed $\mathsf{UnitDisk}^{\uparrow}$ observations and the $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ route have been exposed.

The Real and boundary rows are downstream. The row R seals the metric read as a finite Real-facing datum, while S and B record the boundary source and the refusal of completed ideal-boundary or global-geometry data.

Transport, replay, provenance, and local naming preserve exactly that route. Since no carrier coordinate stores a host analytic metric, completed disk, quotient boundary, selected geodesic, physical horizon, or global compactification, none can become a local NameCert obligation. □

Theorem 6209.3 (Poincare disk metric transport nonescape)**.** *Every accepted* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *transport of a metric read must remain inside the displayed route*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{PoincareDiskPhaseTransport}^{\uparrow} \longmapsto \mathsf{Met}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow}$$

together with boundary refusal, BHist, hsame, Cont, Pkg, *and local* $\mathsf{NameCert}$ *rows. It cannot export a completed disk, quotient ideal-boundary equality, selected global endpoint, or host analytic distance.*

Proof. Start with the transport rows H and C. They can move only the displayed disk observations, phase-transport route, metric read, Real seal, boundary row, refusal row, provenance, and local naming.

Apply carrier admission. The route reaches the metric read Q only through U and T, and reaches the Real and boundary rows only after that metric read has been displayed.

The refusal row B remains attached to the same route. Thus transported metric data cannot escape into a completed disk, quotient ideal boundary, selected endpoint, host analytic distance, or global compactification. □

Theorem 6209.4 (Poincare disk metric phase-transport handoff)**.** *Every phase-transport consumer of an accepted* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *packet receives the metric read only after the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row, the* $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$ *route, the* $\mathsf{Met}^{\uparrow}$ *comparison row, the* $\mathsf{Real}^{\uparrow}$ *seal, the* $\mathsf{S}^{1\uparrow}$ *boundary row, and the refusal row have been displayed. The handoff is a finite Poincare-disk metric packet over BEDC rows, not a host-level hyperbolic metric.*

Proof. Begin with the phase-transport request. The carrier exposes the disk coordinates U and the transport route T before any metric or boundary-facing row is available.

Read the metric row next. The comparison Q is tied to those displayed disk observations, and the Real seal R records only the finite readback attached to Q.

Finally read the boundary and refusal rows. The row S supplies the visible circle boundary source, while B prevents the handoff from becoming a completed hyperbolic plane, quotient ideal boundary, selected geodesic, physical horizon, or global compactification. □

Falsifiable prediction: An accepted $\mathsf{PoincareDiskMetric}^{\uparrow}$ read is refuted if a metric consumer obtains a completed disk, quotient ideal-boundary equality, selected global endpoint, host analytic distance, or global compactification without displaying UnitDisk coordinates, PoincareDiskPhase-Transport routing, Metric rows, Real seals, SOne boundary rows, boundary refusal, transport, replay, provenance, and local naming.

Independence witness: unitdisk, poincare-disk-phase-transport, metric, s1: no listed sibling is bijective to $\mathsf{PoincareDiskMetric}^{\uparrow}$ because this packet jointly carries disk coordinates, phase transport, metric reads, Real seals, circle-boundary rows, refusal data, transport, replay, provenance, and local naming as one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskMetric}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6209.1, Definition 6209.2, Definition 6209.3, and Definition 6209.4 seed the finite Poincare disk metric route over UnitDisk, PoincareDiskPhaseTransport, Metric, Real, SOne, refusal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed hyperbolic plane, quotient ideal-boundary equality, selected geodesics, host analytic distance, physical horizon data, global compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-coordinate admission, phase-transport admission, metric-read exactness, Real-seal handoff, boundary refusal, transport, replay, provenance, and local naming for BEDC.Derived.PoincareDiskMetricUp.

Constructive story: A $\mathsf{PoincareDiskMetric}^{\uparrow}$ packet is generated from UnitDisk coordinates, PoincareDiskPhaseTransport rows, Metric readbacks, RealUp seals, SOne boundary rows, boundary-refusal data, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6210

A Concrete Naming Certificate for $\mathsf{UniformPolygonalApproximation}^{\uparrow}$

Following Bishop–Bridges constructive approximation theory, the $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ packet records the finite mesh-level certificate by which a displayed uniformly continuous map on a closed bounded interval is read through polygonal data. It sits after the interval source of chapter 1591, the mesh surfaces of chapter 1863 and chapter 2033, the affine-cell carrier of chapter 3906, the located-modulus handoff of chapter 1859, the continuous-map boundary of chapter 130, the terminal real-name seal of chapter 38, and the dyadic tolerance ledger of chapter 291. The chapter names a finite uniform approximation route; it does not assert open-cover compactness, choose a limit polygonal sequence, identify host functions by quotient equality, or import ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6210.1 (Uniform polygonal approximation carrier)**.** *A* $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (I, M, L, Q, F, R, D, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{ClosedBoundedInterval}^{\uparrow}$ *source row, displaying the endpoint real names and the bounded interval window on which the approximation is read.*
2. *M is a* $\mathsf{DyadicPartition}^{\uparrow}$ *or* $\mathsf{UniformPartition}^{\uparrow}$ *mesh row, including the ordered boundary ledger, endpoint-coverage route, and positive mesh-width evidence.*
3. *L is the* $\mathsf{PiecewiseLinearApproximation}^{\uparrow}$ *polygonal row whose affine cells, endpoint values, and local slope ledger are read over the mesh M.*
4. *Q is the* $\mathsf{LocatedUniformContinuity}^{\uparrow}$ *modulus row, exposing a located tolerance-to-mesh request for the same interval source and displayed map.*
5. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *row whose graph and pointwise continuity data are consumed only through I, M, Q, L.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *endpoint and value-readback row reached after the mesh, modulus, and polygonal rows have been displayed.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic request, mesh bound, cell-width comparison, and uniform error budget used by downstream consumers.*

8. *H records componentwise* hsame *transport for the interval source, mesh, polygonal row, located modulus, continuous-map handoff, real readbacks, dyadic ledger, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$I, M, Q, L, D \longmapsto F, R \longmapsto H, P, N.$$

10. *P is the* Pkg *provenance over* UniformPolygonalApproximation$^{\uparrow}$, ClosedBoundedInterval$^{\uparrow}$, DyadicPartition$^{\uparrow}$, UniformPartition$^{\uparrow}$, PiecewiseLinearApproximation$^{\uparrow}$, LocatedUniformContinuity$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* NameCert$_{\text{UniformPolygonalApproximation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $I, M, L, Q, F, R, D, H, C, P, N$ and their displayed transports. It has no coordinate for open-cover compactness, selected limiting polygonal sequence, host function graph, quotient equality of maps, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniformPolygonalApproximationUp.UniformPolygonalApproximationTasteGate`
`single carrier alignment`

Theorem 6210.2 (Uniform polygonal approximation NameCert obligations)**.** *For every accepted* UniformPolygonalApproximation$^{\uparrow}$ *carrier*

$$U = (I, M, L, Q, F, R, D, H, C, P, N),$$

the local NameCert$_{\text{UniformPolygonalApproximation}^{\uparrow}}$ *surface consists exactly of interval-source admission through I, mesh coverage through M, polygonal-cell admission through L, located modulus exposure through Q, continuous-map handoff through F, real readback through R, dyadic tolerance accounting through D, componentwise transport through H, replay through C, provenance through P, and local naming through N. No local obligation may read open-cover compactness, a selected limiting polygonal sequence, host function equality, quotient map equality, countable choice, or ambient real completeness.*

Proof. Project the accepted carrier of Definition 6210.1. The displayed analytic rows are precisely the closed bounded interval I, mesh M, piecewise-linear row L, located modulus Q, continuous-map handoff F, real readback R, and dyadic ledger D.

The mesh and modulus rows determine the finite approximation request. The row M supplies the visible cells, while Q supplies the located uniform-continuity budget that makes the requested mesh adequate for the same interval and continuous map.

The polygonal row L and dyadic ledger D then exhaust the approximation data. They record endpoint values, affine-cell interpolation, cell-width comparisons, and the error budget before F or R is public.

Finally H, C, P, N transport, replay, source, and name this same packet. Since no further carrier coordinate remains, the excluded compactness, limit-selection, quotient, choice, host-function, and completeness objects cannot become local NameCert obligations. □

Theorem 6210.3 (Uniform polygonal approximation mesh handoff)**.** *Every* PiecewiseLinearApproximation$^{\uparrow}$, LocatedUniformContinuity$^{\uparrow}$, ContinuousMap$^{\uparrow}$, Real$^{\uparrow}$, *or* DyadicRatCore$^{\uparrow}$ *consumer of an accepted* UniformPolygonalApproximation$^{\uparrow}$ *carrier receives only the finite route*

ClosedBoundedInterval$^{\uparrow}$, DyadicPartition$^{\uparrow}$ or UniformPartition$^{\uparrow}$, LocatedUniformContinuity$^{\uparrow}$, PiecewiseLinearApproxim

Thus the consumer may read a uniform error budget over the displayed mesh, but it cannot replace that handoff by an open-cover compactness theorem, a selected polygonal limit, quotient equality of functions, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the mesh row M and the interval source I. They display the finite cell boundary ledger and endpoint coverage over the closed bounded interval.

Read the located modulus row Q next. It connects the requested tolerance to a mesh-width bound for the same displayed continuous map, so the approximation request is routed through finite modulus data rather than through open-cover compactness.

The piecewise-linear row L and dyadic ledger D provide the polygonal handoff. They expose affine cells, endpoint-value rows, slope bounds, and error accounting before the continuous-map and real readback rows are visible.

The structural rows H, C, P, N preserve the same route for every downstream consumer. Any read that omits M, Q, L, or D would use a coordinate absent from the carrier, so it is not a read of the accepted packet. □

Theorem 6210.4 (Uniform polygonal approximation nonescape)**.** *Let* $U = (I, M, L, Q, F, R, D, H, C, P, N)$ *be an accepted* $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ *carrier. Every real-facing or continuous-map-facing read from* U *factors through the displayed mesh* M*, located modulus* Q*, polygonal row* L*, and dyadic tolerance ledger* D *before it reaches* F *or* R*. Consequently the packet exports no ambient real completeness, selected uniform limit, quotient function equality, open-cover compactness, countable-choice schedule, or host function-space equality.*

Proof. The carrier lists F and R only after the mesh, modulus, polygonal, and dyadic rows. A continuous-map or real-facing consumer therefore cannot inspect those terminal rows before the finite approximation evidence has been displayed.

The located modulus row Q is tied to the same interval source and continuous-map handoff as L. It supplies the requested mesh adequacy inside the packet, not an external compactness theorem.

The dyadic row D records the numerical tolerance and cell-width comparison used by the handoff. Thus every error claim exposed to F or R is a finite dyadic readback from the carrier.

Transport, replay, provenance, and local naming do not add analytic data. They keep the read inside $I, M, L, Q, F, R, D, H, C, P, N$, so the excluded limit, quotient, choice, host-function, compactness, and completeness surfaces remain outside the certificate. □

Theorem 6210.5 (Uniform polygonal approximation mesh refinement)**.** *Let* $U = (I, M, L, Q, F, R, D, H, C, P, N)$ *be an accepted* $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ *packet. If the packet is reread on a dyadic mesh* M' *refining* M *and satisfying the same located modulus row* Q *at the requested tolerance, then the refined polygonal row* L' *and dyadic ledger* D' *remain inside the route*

$$I, M, Q, L, D \to F, R \to H, P, N$$

and preserve the same finite approximation handoff to $\mathsf{ContinuousMap}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*

Proof. Project the mesh row M and the located modulus row Q. The refined mesh M' is accepted only as a refinement of the displayed cell ledger and only at a tolerance admitted by the same located modulus request.

Use the dyadic comparison ledger to relate the cell widths of M' to those of M. The refined dyadic row D' is therefore a replay of the packet's finite error accounting, not a separate compactness or choice argument.

Replay the piecewise-linear row over the refined mesh. The resulting L' supplies affine-cell data over the smaller cells while remaining tied to the original interval source I, modulus row Q, and continuous-map handoff F.

Finally apply Definition 6210.4. It blocks selected polygonal limits, quotient map equality, choice, open-cover compactness, and ambient $\mathsf{Real}^{\uparrow}$ completeness, so the refined read preserves the finite approximation route. □

Falsifiable prediction: A $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ consumer that reaches $\mathsf{ContinuousMap}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ approximation data without displaying the mesh row M, located modulus row Q, polygonal row L, and dyadic tolerance row D refutes the carrier surface.

Independence witness: Nearest siblings: closedboundedinterval, dyadic-partition, uniform-partition, piecewise-linear-approximation, located-uniform-continuity, continuousmap, real, dyadicratcore. Those chapters supply interval, mesh, affine-cell, modulus, map, real-readback, and dyadic-tolerance rows separately; this chapter binds them into the single eleven-row finite uniform approximation route $U = (I, M, L, Q, F, R, D, H, C, P, N)$.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformPolygonalApproximation}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformPolygonalApproximation}^{\uparrow}$ packet is generated from a ClosedBoundedInterval source row, a DyadicPartition or UniformPartition mesh row, a PiecewiseLinearApproximation polygonal row, a LocatedUniformContinuity modulus row, a ContinuousMap handoff row, RealUp readback rows, a DyadicRatCore tolerance ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformPolygonalApproximation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6210.1, Definition 6210.2, Definition 6210.3, and Definition 6210.4 seed the finite uniform polygonal approximation route over interval, mesh, polygonal, located-modulus, continuous-map, real-readback, dyadic-ledger, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformPolygonalApproximation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected uniform limits, quotient function equality, countable choice, host function spaces, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, mesh coverage, polygonal-cell admission, located modulus exposure, continuous-map handoff, real readback, dyadic tolerance accounting, transport, replay, provenance, and local naming for BEDC.Derived.UniformPolygonalApproximationUp.

Chapter 6211

A Concrete Naming Certificate for $\mathsf{ItoIntegral}^{\uparrow}$

Following Ito and standard constructive stochastic-integration presentations, the $\mathsf{ItoIntegral}^{\uparrow}$ packet records the finite adapted-step surface by which a Brownian cell-difference row calculation may be named before any host measure completion, almost-sure quotient, or selected limiting integral is invoked. It sits over the probability-space rows of $\mathsf{ProbSpace}^{\uparrow}$ in chapter 184, the random-variable rows of $\mathsf{RandomVar}^{\uparrow}$ in chapter 185, the adapted-sequence ledger of $\mathsf{Martingale}^{\uparrow}$ in chapter 190, and the finite Brownian prefix rows of $\mathsf{Brownian}^{\uparrow}$ in chapter 191. The certificate names only displayed partitions, step integrand values, Brownian cell-difference rows, finite sums, martingale transport, replay, provenance, and local naming rows.

Definition 6211.1 (ItoIntegral carrier)**.** *An* $\mathsf{ItoIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (\Omega, X, B, \Pi, A, \Delta, S, M, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row on which all probability-facing data are read.*
2. X *is a finite family of adapted step-integrand* $\mathsf{RandomVar}^{\uparrow}$ *rows over the displayed filtration schedule.*
3. B *is the* $\mathsf{Brownian}^{\uparrow}$ *finite-prefix row whose adjacent Brownian cell-difference rows are available on the same schedule.*
4. Π *is a rational partition row, with each cell endpoint named through the rational and dyadic-rational ledger already visible to the packet.*
5. A *records adaptedness of each step-integrand value before the matching Brownian cell-difference row is consumed.*
6. Δ *records the Brownian cell-difference rows associated to the cells of* Π.
7. S *is the finite sum row pairing each adapted step value with its displayed Brownian cell-difference row.*
8. M *is the martingale transport row exposing the finite martingale comparison for the step-integral sums.*
9. H *records componentwise* hsame *transport for the probability, random-variable, Brownian, partition, adaptedness, cell-difference, finite-sum, and martingale rows.*

10. *C is the displayed* Cont *replay row for the same finite packet.*

11. *P is the* Pkg *provenance over* ItoIntegral$^{\uparrow}$, ProbSpace$^{\uparrow}$, RandomVar$^{\uparrow}$, Martingale$^{\uparrow}$, Brownian$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{ItoIntegral}^{\uparrow}}$ *row.*

The carrier exports no host stochastic process equality, no almost-sure quotient, no selected refining partition, no measure-theoretic completion, no L^2 completion theorem, and no ambient stochastic-integral object beyond the displayed finite sum route.

Theorem 6211.2 (ItoIntegral NameCert obligations)**.** *For every accepted* ItoIntegral$^{\uparrow}$ *carrier $I = (\Omega, X, B, \Pi, A, \Delta, S, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ItoIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission through Ω, X, B, Π, adapted finite step exposure through A, Brownian cell-difference row exposure through Δ, finite-sum exposure through S, martingale transport through M, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Any consumer of the packet must read the integrand value, the matching Brownian cell-difference row, and the finite sum row before it can inspect the martingale transport row. The packet therefore names the finite Ito step integral route, not a completed stochastic integral in an ambient measure model.

Lean4 ✓ `BEDC.Derived.ItoIntegralUp.ItoIntegralTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6211.1. Its only probability-facing source is Ω, its only integrand rows are the adapted step rows X, A, and its only Brownian data are the finite-prefix and cell-difference rows B, Δ.

The finite sum row S is indexed by the same rational partition Π. Each summand pairs one adapted step value with the Brownian cell-difference row displayed on that partition cell, so the certificate cannot replace the visible step sum by a selected limiting partition or a host stochastic-integral object.

The martingale-facing read is mediated by M. Since M is transported through H and replayed through C inside the same provenance row P, it preserves the finite sum route already named by S rather than adding an almost-sure quotient, measure completion, L^2 completion, or ambient Brownian path equality.

Thus the NameCert obligations are exactly the listed carrier, adaptedness, cell-difference, finite-sum, martingale-transport, replay, provenance, and naming rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{ItoIntegral}^{\uparrow}}$**.**

Constructive story: An ItoIntegral$^{\uparrow}$ packet is generated from a ProbSpaceUp source row, adapted finite-step RandomVarUp integrand rows, BrownianUp finite-prefix cell-difference rows, rational partition rows, finite sums, martingale transport, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ItoIntegral}^{\uparrow}}$, seed)

Scope closed: Definition 6211.1 and Definition 6211.2 seed the finite adapted-step Ito integral NameCert route over ProbSpaceUp, RandomVarUp, MartingaleUp, BrownianUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ItoIntegral}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host stochastic process equality, almost-sure quotients, selected refining partitions, measure completion, L2 completion, Lean verification, standard bridges, or an ambient stochastic-integral object.

Upgrade path: Obligation closure requires checked carrier admission, adapted finite-step exposure, Brownian cell-difference row exposure, rational partition accountability, finite-sum exactness, martingale transport, replay, provenance, and local naming for BEDC.Derived.ItoIntegralUp.

Chapter 6212

A Concrete Naming Certificate for $\mathsf{ItoIsometry}^{\uparrow}$

Following Ito's stochastic-integration isometry and standard martingale Hilbert-space treatments, the $\mathsf{ItoIsometry}^{\uparrow}$ packet records the finite L^2 stability surface by which simple adapted stochastic-integral rows compare their squared norm ledger with the quadratic-variation ledger of the driving Brownian cell differences. It sits downstream of the finite adapted-step integral route of chapter 6211, the probability and Brownian rows of chapter 184 and chapter 191, the martingale ledger of chapter 190, and the Hilbert and norm-facing rows of chapter 91 and chapter 89. The packet names only finite simple-process rows, Brownian cell-difference rows, quadratic-variation comparisons, L^2 norm comparisons, transport, replay, provenance, and local naming; it does not build a completed stochastic-integral space.

Definition 6212.1 (ItoIsometry carrier)**.** *An* $\mathsf{ItoIsometry}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (\Omega, I, X, B, \Delta, V, L, H, C, P, N)$$

with the following displayed rows:

1. Ω *is the* $\mathsf{ProbSpace}^{\uparrow}$ *source row on which the probability-facing readback is interpreted.*
2. I *is the consumed* $\mathsf{ItoIntegral}^{\uparrow}$ *simple adapted-step integral packet.*
3. X *is the finite simple adapted-process row, indexed by the same displayed partition and filtration windows as* I*.*
4. B *is the* $\mathsf{Brownian}^{\uparrow}$ *finite-prefix source row exposing the Brownian cell-difference schedule.*
5. Δ *is the Brownian cell-difference row aligned with the simple-process cells.*
6. V *is the quadratic-variation ledger for the displayed Brownian cell differences and step-process coefficients.*
7. L *is the* L^2 *norm-comparison row, read through the* $\mathsf{Hilbert}^{\uparrow}$ *and* $\mathsf{Norm}^{\uparrow}$ *sibling surfaces, that compares the finite integral-sum norm ledger with* V*.*
8. H *records componentwise* hsame *transport for the source, integral, simple-process, Brownian, cell-difference, quadratic-variation, and norm-comparison rows.*
9. C *is the displayed* Cont *replay row for the same finite packet.*

10. *P is the* Pkg *provenance over* $\mathsf{ItoIsometry}^{\uparrow}$, $\mathsf{ItoIntegral}^{\uparrow}$, $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Brownian}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ItoIsometry}^{\uparrow}}$ *row.*

The carrier exports no selected refining partition, no almost-sure quotient, no completed L^2 space, no ambient stochastic-integral operator, and no measure-completion theorem.

Theorem 6212.2 (ItoIsometry NameCert obligations)**.** *For every accepted* $\mathsf{ItoIsometry}^{\uparrow}$ *carrier* $Q = (\Omega, I, X, B, \Delta, V, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ItoIsometry}^{\uparrow}}$ *surface consists exactly of probability-source exposure through Ω, simple-integral exposure through I, simple adapted-process exposure through X, Brownian source and cell-difference exposure through B, Δ, quadratic-variation ledger exposure through V, finite L^2 norm-comparison exposure through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6212.1. The source, integral, simple-process, and Brownian rows are displayed before any norm comparison can be read.

The quadratic-variation row V and norm-comparison row L are tied to the same finite partition and cell-difference schedule. Thus the certificate compares the visible simple integral sum with the visible quadratic-variation ledger, not with a selected completed stochastic integral.

Transport, replay, provenance, and local naming preserve that packet rowwise. Since the carrier has no coordinate for almost-sure quotienting, measure completion, selected refinement, or completed L^2 operator theory, the listed rows exhaust the local NameCert surface. □

Theorem 6212.3 (ItoIsometry finite L2 stability)**.** *Every accepted* $\mathsf{ItoIsometry}^{\uparrow}$ *consumer that reads a finite L^2 stability claim must preserve the simple adapted-process row X, Brownian cell-difference row Δ, quadratic-variation ledger V, and norm-comparison row L as one displayed packet over the consumed* $\mathsf{ItoIntegral}^{\uparrow}$ *row I. It cannot replace the finite comparison by a host L^2 completion, a selected limiting integral, or a quotient equality of stochastic processes.*

Proof. Use the NameCert obligations of Definition 6212.2. They make X, Δ, V, L public only as rows of the same accepted packet over I.

The martingale and Hilbert-facing dependencies are provenance rows, not hidden completions. They allow the finite norm comparison to be consumed through the displayed sibling surfaces while keeping the Brownian cell-difference and quadratic-variation ledgers visible.

Therefore a stability read that changes X, Δ, V, or L leaves the carrier. No completed L^2 space, selected limiting integral, or quotient process equality is available from transport, replay, provenance, or local naming. □

Theorem 6212.4 (ItoIsometry nonescape)**.** *Every* $\mathsf{ItoIsometry}^{\uparrow}$ *consumer factors through*

$$\Omega, I, X, B, \Delta, V, L, H, C, P, N.$$

In particular, stochastic-integral stability can be consumed only as the finite simple-process and Brownian cell-difference comparison named by the packet; it cannot escape to an ambient probability completion, host Hilbert-space completion, almost-sure quotient, selected refinement sequence, or unlogged martingale theorem.

Proof. Start with the carrier inventory. Each accepted packet displays exactly the probability source, ItoIntegral dependency, simple adapted-process row, Brownian source, cell-difference row, quadratic-variation ledger, norm-comparison row, transport, replay, provenance, and local name.

The sibling references to $\mathsf{ProbSpace}^{\uparrow}$, $\mathsf{Brownian}^{\uparrow}$, $\mathsf{Martingale}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, and $\mathsf{Norm}^{\uparrow}$ are consumed through P and the displayed rows. They do not add a coordinate for completed probability spaces, completed Hilbert spaces, almost-sure quotients, selected refinements, or detached martingale theorems.

Thus any consumer of the isometry surface must preserve the listed finite rows. A read that requires an ambient completion or quotient equality is outside the $\mathsf{ItoIsometry}^{\uparrow}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{ItoIsometry}^{\uparrow}}$.

Constructive story: An $\mathsf{ItoIsometry}^{\uparrow}$ packet is generated from a ProbSpaceUp source row, consumed ItoIntegralUp simple adapted-step integral row, simple adapted-process row, BrownianUp finite-prefix and cell-difference rows, quadratic-variation ledger, finite L2 norm-comparison row, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ItoIsometry}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6212.1, Definition 6212.2, Definition 6212.3, and Definition 6212.4 seed the finite L2-stability route for simple adapted stochastic-integral rows over ItoIntegralUp, ProbSpaceUp, BrownianUp, MartingaleUp, HilbertUp, NormUp, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ItoIsometry}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed stochastic-integral construction, almost-sure quotients, selected refining partitions, measure completion, host Hilbert-space completion, Lean verification, axiom-clean audit, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, simple adapted-process exposure, Brownian cell-difference exposure, quadratic-variation accounting, finite L2 norm comparison, transport, replay, provenance, and local naming for BEDC.Derived.ItoIsometryUp.

Chapter 6213

A Concrete Naming Certificate for $\mathsf{MaximumModulusPrinciple}^{\uparrow}$

Following Ahlfors and Conway on the maximum-modulus principle for holomorphic functions, the $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ packet records the boundary-maximum route as finite NameCert data. It consumes the complex carrier of chapter 39, the holomorphic source rows of chapter 68, the circle-boundary surface of chapter 58, the terminal real-name seal of chapter 38, and the three-circle annulus ledger of chapter 3892. The packet names a displayed boundary upper-bound certificate; it is not an ambient supremum theorem, a selected maximizer, analytic continuation beyond the named domain, quotient function equality, or a bridge certificate.

Definition 6213.1 (Maximum-modulus principle carrier)**.** *A* $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (D, F, B, R, U, H, C, P, N)$$

with the following displayed rows:

1. *D is the named complex domain or annulus row, read through* $\mathsf{Complex}^{\uparrow}$ *and the relevant boundary carrier.*
2. *F is the* $\mathsf{Holomorphic}^{\uparrow}$ *map row over D, with values in the same* $\mathsf{Complex}^{\uparrow}$ *carrier.*
3. *B is the finite boundary-read ledger, using* $\mathsf{S}^{1\uparrow}$ *or an equivalent displayed boundary source to expose the inspected boundary probes.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *modulus-read row for the boundary and interior comparison values.*
5. *U is the boundary upper-bound row, recording the displayed maximum-modulus inequality available to downstream consumers.*
6. *H stores componentwise* hsame *transport for the domain, holomorphic map, boundary reads, modulus reads, upper-bound row, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a maximum-modulus request through D, F, B, R, U, H.*
8. *P is the* Pkg *provenance over* $\mathsf{MaximumModulusPrinciple}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{Holomorphic}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{HadamardThreeCircle}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *dependency rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MaximumModulusPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. No coordinate stores a host compactness theorem, selected boundary point, ambient supremum, quotient function equality, analytic continuation outside D, or countable-choice schedule.

Lean4 ✓ BEDC.Derived.MaximumModulusPrincipleUp.MaximumModulusPrincipleTasteGate single carrier alignment

Theorem 6213.2 (Maximum-modulus principle NameCert obligations)**.** *For every accepted* MaximumModulusPrin *carrier* $M = (D, F, B, R, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MaximumModulusPrinciple}^{\uparrow}}$ *surface consists exactly of domain admission, holomorphic-map exposure, boundary-read visibility, real modulus readback, boundary upper-bound compatibility, componentwise transport, displayed replay, package provenance, and local naming. A consumer may not read a maximum-modulus bound until these rows are displayed, and the certificate does not export an ambient supremum theorem, selected maximizer, quotient function equality, analytic continuation, or countable choice.*

Proof. Project the carrier of Definition 6213.1. The domain row D and holomorphic row F fix the complex source before any boundary comparison is available.

The boundary and modulus data are then read in order. The row B exposes only the finite boundary probes, while R turns the inspected values into $\mathsf{Real}^{\uparrow}$ modulus rows. The upper-bound row U is therefore attached to the displayed boundary ledger rather than to an external supremum.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of these rows adds a selected boundary point, host compactness proof, analytic continuation coordinate, quotient equality, or countable-choice schedule, the listed obligations exhaust the NameCert surface. □

Theorem 6213.3 (Maximum-modulus principle Hadamard boundary handoff)**.** *Every* $\mathsf{HadamardThreeCircle}^{\uparrow}$ *or holomorphic-boundary consumer that reads an accepted* $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ *packet receives the displayed route*

$$\mathsf{Complex}^{\uparrow} \longmapsto \mathsf{Holomorphic}^{\uparrow} \longmapsto \mathsf{S}^{1\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{MaximumModulusPrinciple}^{\uparrow}$$

together with the rows $D, F, B, R, U, H, C, P, N$*. When the consumer is an annulus packet, the boundary read may be handed to Definition 3892.3; the handoff preserves the same finite boundary ledger and does not add selected maximizers, host analytic continuation, quotient equality, or an ambient maximum theorem.*

Proof. Start with the holomorphic source rows D and F. They determine the complex carrier and displayed map row that any boundary consumer is allowed to inspect.

Next read the boundary ledger B and the modulus row R. These rows expose the finite boundary probes and real modulus comparisons before U can be consumed as an upper-bound row.

For an annulus consumer, use Definition 3892.3. The Hadamard packet receives exactly the displayed boundary maximum row already attached to B, R, U, so the three-circle ledger is a downstream read of the same finite data.

The structural rows H, C, P, N keep the handoff inside the NameCert route. No row supplies a selected extremal point, an ambient supremum proof, analytic continuation beyond the named source, quotient equality, or countable choice. □

Falsifiable prediction: An accepted $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ read is refuted if the boundary upper-bound row U is preserved while the displayed domain D, holomorphic map row F, boundary ledger B, and Real modulus row R are removed.

Independence witness: complex, holomorphic, sone, real, hadamard-three-circle: none is bijective with $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ because this packet jointly carries a complex source, holomorphic map row, finite boundary ledger, Real modulus readback, boundary upper-bound row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MaximumModulusPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{MaximumModulusPrinciple}^{\uparrow}$ packet is generated from a displayed complex source, a holomorphic map row, finite boundary probes, Real modulus reads, a boundary upper-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MaximumModulusPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6213.1, Definition 6213.2, and Definition 6213.3 seed the finite boundary-maximum route for holomorphic NameCert consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MaximumModulusPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close an ambient maximum theorem, selected extremal points, analytic continuation beyond the named source, quotient function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked domain admission, holomorphic-map exposure, boundary-read visibility, Real modulus readback, boundary upper-bound compatibility, transport, replay, provenance, and local naming for BEDC.Derived.MaximumModulusPrincipleUp.

Chapter 6214

A Concrete Naming Certificate for $\mathsf{UniversalCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive completion, the $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ packet records the universal-property boundary of a Cauchy completion as finite NameCert data. It sits between the Cauchy-filter carrier of chapter 1569, the universal extension surface of chapter 2319, the metric completion route of chapter 1506, the separated boundary of chapter 1612, and the displayed readback chain through chapter 126, chapter 301, and chapter 38. The packet gives downstream completion chapters one named carrier for dense extension and uniqueness without importing quotient completion equality, selected representatives, countable choice, or ambient Real completeness.

Definition 6214.1 (Universal Cauchy completion carrier)**.** *A* $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (S, D, E, Q, Z, W, R, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric and Cauchy packet row, naming the metric source, its displayed Cauchy evidence, and the finite tolerance requests read by the completion construction.*
2. *D is the dense embedding row from the source carrier into the completion-facing carrier.*
3. *E is the extension-arrow row, carrying the displayed continuous arrow out of the completed carrier into a separated target.*
4. *Q is the uniqueness row, recording that two accepted extension arrows with the same source action are identified by the stored NameCert route rather than by quotient equality.*
5. *Z is the separated-reflection row, exposing the separated target boundary before uniqueness is read.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the source and extension data at the displayed tolerance.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same windows.*
8. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the StreamName windows and RegSeqRat readback are visible.*
9. *H records componentwise* hsame *transport for the source, embedding, extension, uniqueness, separated-reflection, window, readback, real-seal, replay, provenance, and naming rows.*

10. *C records displayed* Cont *replay from a source tolerance request through the dense embedding, extension arrow, separated boundary, finite windows, regular rational readback, and real seal.*

11. *P is the* Pkg *provenance over* $\mathsf{UniversalCauchyCompletion}^{\uparrow}$, $\mathsf{CauchyFilterCompletion}^{\uparrow}$, CauchyCompletionUniversalI MetricCompletion$^{\uparrow}$, $\mathsf{SeparatedCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

12. *N is the local* $\mathsf{NameCert}_{\mathsf{UniversalCauchyCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, selected representatives, selected limits, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.UniversalCauchyCompletionUp.UniversalCauchyCompletionTasteGate single carrier alignment`

Theorem 6214.2 (Universal Cauchy completion NameCert obligations)**.** *For every accepted* UniversalCauchyCompl *carrier $U = (S, D, E, Q, Z, W, R, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UniversalCauchyCompletion}^{\uparrow}}$ *surface consists exactly of source metric and Cauchy admission, dense embedding admission, extension-arrow admission, uniqueness-row admission, separated-reflection admission, StreamName finite-window admission, RegSeqRat readback, Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the displayed rows $S, D, E, Q, Z, W, R, L, H, C, P, N$.*

Proof. Project the accepted carrier of Definition 6214.1. The analytic rows are the source metric and Cauchy row S, the dense embedding D, the extension arrow E, the uniqueness row Q, the separated-reflection boundary Z, the finite windows W, the regular rational readback R, and the terminal real seal L.

The certificate reads these rows in order. A completion-facing request first sees the source and dense embedding, then the extension arrow into the separated boundary, then the finite windows and regular rational readback used by the real-facing seal.

The rows H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no quotient equality, selected representative, selected limit, countable-choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the local naming obligations. □

Theorem 6214.3 (Universal Cauchy completion density route)**.** *Every density-facing read of an accepted* $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CauchyFilterCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, the consumer reads the source metric and Cauchy packet S, the dense embedding D, the StreamName windows W, the RegSeqRat readback R, and finally the Real seal L, with H, C, P, N preserving the same route. The route supplies a density handoff for $\mathsf{MetricCompletion}^{\uparrow}$ *and* $\mathsf{SeparatedCompletion}^{\uparrow}$ *consumers without quotient completion equality, selected representatives, selected limits, countable choice, host equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start from the obligation surface of Definition 6214.2. It exposes the source metric and Cauchy packet before any dense embedding or completion-facing read is available.

The dense embedding row then fixes the route from source observations to completion observations. The StreamName row supplies finite windows, and the RegSeqRat row repacks those windows as regular rational readback before the terminal Real seal is visible.

Transport, replay, provenance, and local naming keep the same packet throughout the route. Thus a density-facing consumer can cite the completion route only through $S, D, W, R, L, H, C, P, N$, and no excluded host-side datum is introduced. □

Theorem 6214.4 (Universal Cauchy completion extension uniqueness)**.** *Let $U = (S, D, E, Q, Z, W, R, L, H, C, P, N$ be an accepted* $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ *carrier. Any two accepted extension-arrow reads over the same source action and the same separated target boundary are forced through the stored uniqueness row Q, after first exposing the dense embedding D, separated-reflection row Z, StreamName windows W, RegSeqRat readback R, Real seal L, and structural rows H, C, P, N. The conclusion is an equality of accepted NameCert routes, not a quotient equality of completed points or a selected representative.*

Proof. Expose the two extension-arrow reads through the carrier of Definition 6214.1. The shared source action is read through S and the dense embedding row D, while the common separated target boundary is read through Z.

The uniqueness row Q is only available after these rows have been displayed. Its replay uses the same finite windows W, the same regular rational readback R, and the same Real seal L for both extension arrows.

Finally H, C, P, N transport and name the comparison as an accepted route comparison. Because the carrier contains no quotient equality of completed points, selected representative, selected limit, host equality, countable-choice selector, or ambient completeness coordinate, the conclusion is exactly the displayed route uniqueness. □

Falsifiable prediction: A $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ consumer that exports quotient completion equality, a selected representative, a selected limit, a countable-choice schedule, host equality of completed points, or ambient Real completeness after deleting D, Q, W, or R refutes the carrier of Definition 6214.1.

Independence witness: Nearest siblings: CauchyFilterCompletion, CauchyCompletionUniversalProperty, MetricCompletion, SeparatedCompletion, StreamName, RegSeqRat, and Real. UniversalCauchyCompl is not bijective to any one of them because it jointly carries source Cauchy data, dense embedding, extension arrow, uniqueness row, separated reflection, finite windows, regular readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UniversalCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{UniversalCauchyCompletion}^{\uparrow}$ packet is generated from source metric and Cauchy rows, dense embedding rows, extension-arrow rows, uniqueness rows, separated-reflection rows, StreamName windows, RegSeqRat readbacks, RealUp seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniversalCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6214.1, Definition 6214.2, Definition 6214.3, and Definition 6214.4 seed the finite universal-property route from source Cauchy data to dense extension uniqueness.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniversalCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient completion equality, selected representatives, selected limits, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense embedding exactness, extension-arrow replay, uniqueness-row nonescape, separated-reflection routing, StreamName window replay, RegSeqRat readback, Real seal admission, transport, replay, provenance, and local naming for BEDC.Derived.UniversalCauchyCompletionUp.

Chapter 6215

A Concrete Naming Certificate for $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$

Following Minty and Browder, the $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ packet records the finite certificate surface for a monotone operator argument in Hilbert and functional-analysis settings. It reads the Hilbert carrier of chapter 91, the functional-analysis ledger of chapter 94, the weak derivative boundary of chapter 2210, the PDE-facing route of chapter 194, and the operator-ideal dependency of chapter 213. The packet names only the displayed monotone-operator and resolvent ledger rows; it does not assert host maximality, selected weak limits, spectral calculus, or ambient Hilbert-space compactness.

Definition 6215.1 (Minty–Browder monotone-operator carrier)**.** *A* $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (X, A, G, R, W, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Hilbert}^{\uparrow}$ *source row, including the inner-product, norm, and metric rows inspected by the monotonicity test.*
2. *A is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *operator row carrying the displayed graph schedule, domain window, and action ledger.*
3. *G is the monotone graph row, listing the inspected pairs and the nonnegative pairing comparisons.*
4. *R is the resolvent ledger, recording the finite approximation route for the equation carried by $I + \lambda A$ at displayed positive rational parameters.*
5. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *compatibility row used when the same graph is read by weak-form PDE consumers.*
6. *D is the* $\mathsf{Pde}^{\uparrow}$ *boundary row that records forcing terms, weak test windows, and coercive estimate requests consumed downstream.*
7. *H stores componentwise* hsame *transport for Hilbert, operator, graph, resolvent, weak-derivative, PDE, replay, provenance, and naming rows.*
8. *C stores displayed* Cont *replay from Hilbert source data through the monotone graph and resolvent ledger to the PDE-facing rows.*

9. *P is the* Pkg *provenance over* $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$, $\mathsf{Hilbert}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, $\mathsf{OperatorIdeal}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, A, G, R, W, D, H, C, P, N$. It has no coordinate for a host-level maximal-monotone principle, choice-selected solution, quotient graph equality, spectral theorem, or compactness theorem.

Theorem 6215.2 (Minty–Browder monotone-operator NameCert obligations)**.** *For every accepted* $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ *carrier $M = (X, A, G, R, W, D, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MintyBrowderMonotone}}$ *surface consists exactly of carrier admission for the displayed Hilbert source row, functional-analysis operator row, monotone graph row, resolvent ledger, weak-derivative compatibility row, PDE boundary row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming row.*

Lean4 ✓ `BEDC.Derived.MintyBrowderMonotoneOperatorUp.MintyBrowderMonotoneOperatorTasteGate`
`single carrier alignment`

Proof. Project the carrier of Definition 6215.1. The analytic data are exactly the Hilbert row X, the operator row A, the monotone graph row G, the resolvent ledger R, and the weak-form rows W, D.

The monotone graph and resolvent obligations are read before any PDE-facing consumer receives the packet. Thus the weak-derivative and forcing rows are downstream of the displayed graph comparisons and resolvent approximations, not independent host principles.

The structural rows H, C, P, N transport, replay, source, and name that same route. Since no other coordinate occurs in the carrier, the listed obligations exhaust the local NameCert surface. □

Theorem 6215.3 (Minty–Browder resolvent ledger)**.** *Every* $\mathsf{Pde}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, *or* $\mathsf{OperatorIdeal}^{\uparrow}$ *consumer of an accepted* $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ *packet reads the monotone-operator witness only through the finite route*

$$\mathsf{Hilbert}^{\uparrow}, \mathsf{FunctionalAnalysis}^{\uparrow} \longmapsto G \longmapsto R \longmapsto \mathsf{WeakDerivative}^{\uparrow}, \mathsf{Pde}^{\uparrow}, \mathsf{OperatorIdeal}^{\uparrow}$$

together with H, C, P, N. The route exports a resolvent ledger and monotonicity certificate, not a choice-selected weak solution, host graph equality, spectral decomposition, or ambient compactness claim.

Proof. Start with the Hilbert and functional-analysis rows. They provide the source metric, pairing, operator graph schedule, and action ledger used by the monotonicity comparison.

Read the monotone graph row G next. Its finite pairing comparisons are the only admissible monotonicity evidence, so a downstream consumer cannot replace them by an unrecorded maximality or compactness theorem.

The resolvent ledger R is then consumed before the weak-derivative and PDE rows. Rows W and D therefore receive only the displayed approximation and forcing-route data, while H, C, P, N preserve provenance and local naming without adding excluded solution-selection or spectral coordinates. □

Falsifiable prediction: An accepted $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ read is refuted if a PDE or operator-ideal consumer exports a monotone-operator conclusion without displaying Hilbert source

rows, functional-analysis operator rows, monotone graph comparisons, a resolvent ledger, weak-derivative compatibility, PDE boundary rows, transport, replay, provenance, and local naming.

Independence witness: hilbert, functional-analysis, weak-derivative, pde, operator-ideal: none alone is bijective with $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ because the packet jointly carries Hilbert source data, operator graph rows, monotonicity comparisons, resolvent ledgers, weak-form compatibility, PDE boundary rows, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}}$.

Constructive story: A $\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}$ packet is generated from Hilbert source rows, functional-analysis operator rows, monotone graph comparisons, a resolvent ledger, weak-derivative compatibility, PDE boundary rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6215.1, Definition 6215.2, and Definition 6215.3 seed the finite monotone-operator and resolvent certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MintyBrowderMonotoneOperator}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, host maximal monotonicity, selected weak solutions, quotient graph equality, spectral calculus, or ambient compactness.

Upgrade path: Obligation closure requires checked carrier admission, monotone graph stability, resolvent ledger exactness, weak-form compatibility, PDE boundary handoff, provenance, and local naming for BEDC.Derived.MintyBrowderMonotoneOperatorUp.

Chapter 6216

A Concrete Naming Certificate for $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$

Following the Heine–Cantor finite-net route, the $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ packet records the diameter-decay step that turns pointwise compact-cover moduli into a tolerance-only uniform-modulus ledger. It sits over the metric carrier of chapter 57, the compact metric source of chapter 129, the continuous-map surface of chapter 130, the rational fold of chapter 37, and the uniform-modulus consumer of chapter 369. The packet names finite compact-net diameter control; it does not choose points from a cover, assert host compactness, identify functions by quotient equality, or import ambient real completeness.

Definition 6216.1 (Compact uniform diameter-decay carrier)**.** *A* $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (M, K, F, N, L, T, U, H, C, P)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *comparison row, recording the source and target distance ledgers used by the triangle estimate.*
2. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the inspected centers and cover radii.*
3. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *pointwise-modulus row displayed at the finite centers.*
4. *N is the finite net schedule whose cells carry the local diameter rows.*
5. *L is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound fold row that produces the positive radius budget from the finite local ledgers.*
6. *T is the triangle ledger combining source-net distance, local modulus radius, and target-distance tolerance.*
7. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row reached after the finite diameter rows and rational lower-bound fold are displayed.*
8. *H stores componentwise* hsame *transport for metric, compact-net, continuous-map, net schedule, rational fold, triangle, uniform-modulus, replay, provenance, and naming rows.*
9. *C stores displayed* Cont *replay from compact source data through pointwise moduli and diameter decay into the uniform-modulus row.*

10. *P is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformDiameterDecay}^{\uparrow}}$ *row together with the* Pkg *provenance over* $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

The classifier compares accepted packets only through $M, K, F, N, L, T, U, H, C, P$ and has no coordinate for external modulus selection, host function equality, arbitrary cover choice, quotient representation, or ambient compactness.

Lean4 ✓ `BEDC.Derived.CompactUniformDiameterDecayUp.CompactUniformDiameterDecayTasteGate single carrier alignment`

Theorem 6216.2 (Compact uniform diameter-decay NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ *carrier $D = (M, K, F, N, L, T, U, H, C, P)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformDiameterD}}$ *surface consists exactly of carrier admission for the metric comparison row, compact finite-net row, continuous-map pointwise row, finite diameter schedule, rational lower-bound fold, triangle ledger, uniform-modulus output row, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6216.1. Its displayed analytic rows are exactly the metric comparison row M, compact finite-net row K, pointwise continuous-map row F, finite diameter schedule N, rational fold L, triangle ledger T, and uniform-modulus row U.

The lower-bound fold and triangle ledger are read after the compact-net and pointwise-modulus rows. Hence the output uniform modulus is sourced by finite diameter-decay data rather than by a detached compactness or continuity theorem.

Rows H, C, P transport, replay, source, and name the same route. Since no other coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 6216.3 (Compact uniform diameter-decay uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer of an accepted* $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ *packet factors through*

$$\mathsf{Met}^{\uparrow}, \mathsf{CompactMetric}^{\uparrow}, \mathsf{ContinuousMap}^{\uparrow} \longmapsto N, L, T \longmapsto \mathsf{UniformModulus}^{\uparrow}$$

together with H, C, P. Thus the consumer receives a tolerance-only modulus ledger after finite compact-net diameter decay, not an external modulus selector, host compactness theorem, arbitrary cover choice, quotient function equality, or ambient real-completeness claim.

Proof. Begin with the compact metric source and metric comparison rows. The finite net K and distance ledger M determine the source-side cells that will be inspected by the diameter-decay route.

Read the continuous-map row F at the displayed centers. The finite schedule N records the local diameter rows, while the rational fold L supplies the positive radius budget used by the common tolerance request.

The triangle ledger T combines source-cell control with target-distance estimates before U is exported. The structural rows H, C, P preserve this finite route and contain no selector, host equality, arbitrary cover-choice, quotient, compactness, or completeness coordinate. □

Falsifiable prediction: An accepted $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ read is refuted if a compact-uniform-continuity consumer exports a $\mathsf{UniformModulus}^{\uparrow}$ row without displaying metric comparisons, compact finite-net rows, ContinuousMap pointwise moduli, finite diameter schedules, RatUp lower-bound folds, triangle ledgers, transport, replay, provenance, and local naming.

Independence witness: metric, compactmetric, continuousmap, rat, uniformmodulus: none alone is bijective with $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ because the packet jointly carries metric comparisons, compact finite nets, pointwise moduli, finite diameter rows, rational lower-bound folds, triangle ledgers, uniform-modulus output, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformDiameterDecay}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformDiameterDecay}^{\uparrow}$ packet is generated from Metric distance rows, CompactMetric finite-net rows, ContinuousMap pointwise-modulus rows, finite diameter schedules, RatUp lower-bound folds, triangle ledgers, a UniformModulus output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformDiameterDecay}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6216.1, Definition 6216.2, and Definition 6216.3 seed the finite compact-net diameter-decay handoff into UniformModulus.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformDiameterDecay}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, external modulus selection, host function equality, arbitrary cover choice, quotient representation, ambient compactness, or RealUp completeness.

Upgrade path: Obligation closure requires checked compact-net exposure, pointwise-modulus visibility, rational lower-bound fold exactness, triangle ledger compatibility, uniform-modulus handoff, provenance, and local naming for BEDC.Derived.CompactUniformDiameterDecayUp.

Chapter 6217

A Concrete Naming Certificate for $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$

Following Bishop and Bridges, a bounded monotone real-completion argument is useful only when each displayed positive tolerance yields a finite tail window before the completed real name is read. The $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ packet records that tolerance-to-tail surface between the bounded monotone Cauchy witness of chapter 535, the convergence-seal boundary of chapter 918, the dyadic tolerance core of chapter 291, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. It names the finite modulus row consumed by Real and Completion-facing readers; it is not a selected-limit principle, a quotient stream equality, or an ambient completeness theorem.

Definition 6217.1 (Bounded monotone real modulus carrier)**.** *A* $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (W, T, D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *W is the accepted* $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$ *source row, exposing the bounded monotone witness before any completion-facing read.*
2. *T is the finite tail-modulus row assigning each displayed positive dyadic tolerance a tail-window bound for the same bounded monotone source.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, including positivity, halving, comparison, and upper-bound rows used by the tail request.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row exposed at the bound named by T.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row reading the same finite window as regular rational approximation data.*
6. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after W, T, D, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for W, T, D, S, Q, R.*
8. *C records displayed* Cont *replay from tolerance requests through tail bounds, finite windows, regular rational readbacks, the Real seal, and structural rows.*
9. *P is the* Pkg *provenance over* $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$, $\mathsf{BoundedMonotoneCauchyWitness}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneRealModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, T, D, S, Q, R, H, C, P, N$. It has no coordinate for selected limits, quotient stream equality, trichotomy, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.BoundedMonotoneRealModulusUp.BoundedMonotoneRealModulusUp`

Theorem 6217.2 (Bounded monotone real modulus NameCert obligations)**.** *For every accepted* $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ *carrier*

$$M = (W, T, D, S, Q, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BoundedMonotoneRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission through M, bounded monotone witness exposure through W, tail-modulus exposure through T, dyadic tolerance accounting through D, StreamName window exposure through S, RegSeqRat handoff through Q, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N. No local obligation may select a limit, identify quotient streams, infer a tail bound without the dyadic tolerance row, bypass the finite window, invoke trichotomy or countable choice, or export ambient real completeness.*

Proof. Project the carrier of Definition 6217.1. Its analytic rows are precisely the bounded monotone witness, tail modulus, dyadic tolerance ledger, finite stream window, regular rational readback, and Real seal.

The tolerance request is public before the tail row is consumed. The row D supplies the positive dyadic request and its comparison ledger, while T answers with a finite tail-window bound for the same witness row W.

The rows S and Q then expose only the finite window and regular rational readback determined by that bound. The Real seal R is downstream of those rows, so it cannot be used as an external completeness source.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since these structural rows add no selector, quotient, choice, trichotomy, host-real, compactness, or ambient-completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6217.3 (Bounded monotone real Cauchy modulus)**.** *Every accepted* $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ *packet exposes, for each positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance read from D, a finite tail-window bound through T before any* $\mathsf{Real}^{\uparrow}$ *row is readable. The corresponding* $\mathsf{StreamName}^{\uparrow}$ *window S and* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff Q are read at that bound, so the Cauchy-facing modulus is a finite route through*

$$W, D, T, S, Q \to R, H, C, P, N.$$

Proof. Start with a positive dyadic tolerance in D. By the carrier definition, this request is part of the displayed tolerance ledger and is not inferred from the terminal Real seal.

Read the tail row T next. It assigns the request to a finite tail-window bound for the same bounded monotone witness W; the construction therefore produces an explicit window before any Real-facing row can be consumed.

The rows S and Q replay that finite bound as a StreamName window and a RegSeqRat readback. The structural rows preserve the route under hsame, Cont, provenance, and local naming, while the excluded selector and quotient surfaces remain absent. □

Theorem 6217.4 (Bounded monotone real terminal seal)**.** *Every completion-facing consumer of an accepted* $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ *packet factors through* W, T, D, S, Q *before the terminal* $\mathsf{Real}^{\uparrow}$ *row* R *is readable. Consequently the packet exports a finite bounded-monotone tolerance-to-tail modulus and not a selected limit, quotient stream equality, trichotomy principle, countable-choice schedule, host real equality, open-cover compactness theorem, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness proof.*

Proof. Apply Definition 6217.2. It fixes the order in which the bounded witness, tail modulus, dyadic tolerance ledger, stream window, and regular rational readback become available.

A completion-facing consumer can therefore read R only after the finite modulus route has been displayed. If it omits one of W, T, D, S, Q, it asks for a Real seal coordinate without the carrier rows that make the seal local to the packet.

The structural rows H, C, P, N do not add analytic content. They keep every read inside the named route, so the excluded limit, quotient, choice, trichotomy, host-real, compactness, and ambient-completeness surfaces remain outside the certificate. □

Falsifiable prediction: A $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ consumer that reads $\mathsf{Real}^{\uparrow}$ before displaying W, T, D, S, Q, or that exports selected limits, quotient stream equality, trichotomy, countable choice, host real equality, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ completeness, refutes the carrier of Definition 6217.1.

Independence witness: Nearest siblings: boundedmonotonecauchywitness, bounded-monotone-convergence-seal, dyadicratcore, streamname, regseqrat, and real. $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ is not bijective to any one sibling because it jointly carries the bounded monotone witness, tolerance-to-tail row, dyadic ledger, finite stream window, regular rational readback, terminal Real seal, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedMonotoneRealModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedMonotoneRealModulus}^{\uparrow}$ packet is generated from a BoundedMonotoneCauchyWitness source row, a finite tail-modulus row, a DyadicRatCore tolerance ledger, a StreamName window row, a RegSeqRat readback row, a terminal RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneRealModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6217.1, Definition 6217.2, Definition 6217.3, and Definition 6217.4 seed the finite bounded-monotone tolerance-to-tail route before terminal Real reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedMonotoneRealModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, quotient stream equality, trichotomy, countable choice, host real equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, bounded monotone witness exposure, tail-modulus exactness, dyadic tolerance accounting, StreamName window replay, RegSeqRat readback, terminal Real sealing, classifier transport, displayed replay, provenance, and local naming for BEDC.Derived.BoundedMonotoneRealModulusUp.

Chapter 6218

A Concrete Naming Certificate for $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$

Following Moore–Smith, Kelley, and Bourbaki on directed nets and completion by tails, the $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ packet records the finite L10 handoff by which a displayed Moore–Smith net completion route reaches the regular rational and real-name corridor. It sits after the directed-net carrier of chapter 3403, the Cauchy-net completion boundary of chapter 3794, the filter-base surface of chapter 1601, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a finite BEDC bridge object for Real and Completion consumers; it does not select limits, quotient nets, cofinal functions, ambient completions, host equalities, or completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6218.1 (Moore–Smith L10 completion-net carrier)**.** *A* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (D, T, F, W, R, A, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row inherited from the accepted* $\mathsf{MooreSmithNet}^{\uparrow}$ *surface, including finite index vertices, displayed preorder comparisons, and common-upper witnesses.*
2. *T is the tail-window row, recording the finite late-index requests and the displayed tails that answer completion-facing reads.*
3. *F is the* $\mathsf{FilterBase}^{\uparrow}$ *row generated from the displayed directed tails D, T.*
4. *W is the* $\mathsf{CauchyNetCompletion}^{\uparrow}$ *row, exposing the same directed tail data as a Cauchy-net completion boundary.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the retained tail windows and tolerance reads.*
6. *A is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, T, F, W, R have been displayed.*
7. *H records componentwise* hsame *transport for the directed-index, tail-window, filter-base, Cauchy-net completion, regular readback, real-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-net request through D, T, F, W, R, A.*
9. *P is the* Pkg *provenance over* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$*,* $\mathsf{MooreSmithNet}^{\uparrow}$*,* $\mathsf{CauchyNetCompletion}^{\uparrow}$*,* $\mathsf{FilterBase}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{MooreSmithL10CompletionNet}^{\uparrow}}$ *row.*

The carrier exports only this finite L10 route. It has no coordinate for selected limits, quotient nets, cofinal choice, ambient completion objects, host equality of completed values, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.MooreSmithLTenCompletionNetUp.MooreSmithLTenCompletionNetTasteGate single carrier alignment`

Theorem 6218.2 (Moore–Smith L10 completion-net NameCert obligations)**.** *For an accepted* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ *carrier* $L = (D, T, F, W, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MooreSmithL10CompletionN}}$ *surface consists exactly of these finite obligation rows:*

1. *carrier admission for the directed-index, tail-window, filter-base, Cauchy-net completion, regular readback, real-seal, transport, replay, provenance, and local naming rows;*
2. *directed-tail coverage, requiring every completion-net request to consume D and T before F, W, R, A can be inspected;*
3. *filter-base generation, requiring F to be assembled only from the displayed directed tails;*
4. *Cauchy-net completion handoff, requiring W to expose the same completion boundary consumed by chapter 3794;*
5. *regularization, requiring R to be read before the terminal real seal A;*
6. *non-escape for selected limits, quotient nets, cofinal choice, ambient completion, host equality, and* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local obligation is stored outside $D, T, F, W, R, A, H, C, P, N$.

Proof. Carrier admission is projection from Definition 6218.1. The analytic rows are D, T, F, W, R, A, and the structural rows are H, C, P, N.

Directed-tail coverage comes first. A completion-net request reads a displayed index and late-window request through D and T before any filter, completion, regularization, or real-seal row can be used.

The filter-base and Cauchy-net obligations then use the sibling routes. The row F is generated from the same directed tails, and W exposes those tails as the Cauchy-net completion boundary consumed by the existing completion chapter.

Finally R and A provide the regular rational readback and terminal real seal. Transport and replay preserve the same route through H and C, while P, N name it as the local certificate surface. Since the excluded completion principles have no carrier coordinate, they cannot become local NameCert obligations. □

Theorem 6218.3 (Moore–Smith L10 regularization)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read from an accepted* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ *packet factors through the directed-tail and completion route*

$$D, T \longmapsto F, W \longmapsto R.$$

Thus the regular rational row is attached to the same Moore–Smith tail windows and Cauchy-net completion boundary; it is not a separate sequence selector, quotient of nets, cofinal-choice schedule, or host-level limit name.

Proof. Begin with the NameCert obligations of Definition 6218.2. Directed-tail coverage forces every read of the packet to expose D and T before downstream rows are inspected.

Apply filter-base generation and Cauchy-net completion handoff. The rows F and W are downstream of the same displayed tails, so the regular readback cannot receive an independent source.

The row R is therefore a readback of the retained finite route. The carrier inventory contains no sequence selector, quotient-net row, cofinal-choice function, or host limit coordinate, so the regularization is exhausted by D, T, F, W, R. □

Theorem 6218.4 (Moore–Smith L10 filter handoff)**.** *Every filter-facing consumer of an accepted* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ *packet reads its* $\mathsf{FilterBase}^{\uparrow}$ *data through the displayed Moore–Smith tail rows*

$$D, T \longmapsto F \longmapsto W, R, A, H, C, P, N.$$

The handoff preserves the directed-net origin of the completion boundary and does not import an external filter, ultrafilter, quotient net, selected limit, or ambient completion object.

Proof. Project the accepted carrier. Its only filter-facing coordinate is F, and the carrier definition presents F as generated from D and T.

The Cauchy-net completion and regularization rows are downstream of that same filter-base row. Thus a consumer that reaches W, R, A has already passed through the displayed directed-tail route.

The structural rows do not add an external filter. They transport, replay, source, and name the same finite package, so no ultrafilter, quotient net, selected limit, or ambient completion object can bypass D, T, F. □

Theorem 6218.5 (Moore–Smith L10 real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read from an accepted* $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ *packet factors through*

$$D, T, F, W, R, A, H, C, P, N.$$

The seal A is terminal for the displayed L10 route only: it does not assert completeness of $\mathsf{Real}^{\uparrow}$*, identify quotient-net representatives, select a limit, compare host real values, or provide a cofinal choice function.*

Proof. Use the regularization theorem. By Definition 6218.3, every regular rational read reaches R only after the directed tails, filter-base row, and Cauchy-net completion row have been displayed.

The real seal is downstream of that readback. The carrier definition gives A as the terminal $\mathsf{Real}^{\uparrow}$ row after D, T, F, W, R, and the NameCert obligations leave no other route to a real-facing read.

It remains to exclude escape data. The packet has no coordinate for $\mathsf{Real}^{\uparrow}$ completeness, quotient-net representatives, selected limits, host equality, cofinal choice, or ambient completion, and H, C, P, N preserve only the rows already displayed. □

Falsifiable prediction: If a Real and Completion consumer reads a $\mathsf{Real}^{\uparrow}$ seal from a directed-net completion boundary without first displaying the $\mathsf{MooreSmithNet}^{\uparrow}$ tail row, the $\mathsf{FilterBase}^{\uparrow}$ handoff, the $\mathsf{CauchyNetCompletion}^{\uparrow}$ boundary, and the $\mathsf{RegSeqRat}^{\uparrow}$ readback, it is not a $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ packet.

Independence witness: moore-smith-net, cauchynetcompletion, filterbase, regseqrat, real: MooreSmithL10Compl is not bijective to any listed sibling because it binds directed tails, filter-base generation, Cauchy-net completion, regular rational readback, terminal real sealing, transport, replay, provenance, and local naming in one finite L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{MooreSmithL10CompletionNet}^{\uparrow}}$.

Constructive story: A $\mathsf{MooreSmithL10CompletionNet}^{\uparrow}$ packet is generated from a MooreSmithNet directed-index row, finite tail-window data, a FilterBase handoff, a CauchyNetCompletion boundary, RegSeqRat readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MooreSmithL10CompletionNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6218.1, Definition 6218.2, Definition 6218.3, Definition 6218.4, and Definition 6218.5 seed the finite L10 completion-net handoff from Moore–Smith tails to regular rational readback and the real-name seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MooreSmithL10CompletionNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient nets, cofinal choice, ambient completion, host equality, $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail coverage, filter-base generation, Cauchy-net completion handoff, regularization, real-seal nonescape, transport preservation, and ledger refusal for BEDC.Derived.MooreSmithLTenCompletionNetUp.

Chapter 6219

A Concrete Naming Certificate for CauchyModulusEnvelope$^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy moduli, the CauchyModulusEnvelope$^{\uparrow}$ packet records a finite majorant envelope for the modulus rows used by regular rational and real-name convergence arguments. It sits over the dyadic arithmetic of chapter 291, the finite stream windows of chapter 126, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names cofinal modulus majorants and finite-family merges before any Real$^{\uparrow}$ seal is read; it is not a quotient of Cauchy streams, a selected completion point, countable choice, or an ambient real-completeness theorem.

Definition 6219.1 (Cauchy modulus envelope carrier)**.** *A* CauchyModulusEnvelope$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, W, R, F, M, U, S, H, C, P, N)$$

with the following displayed rows:

1. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row used to compare requested precision levels and dyadic tail bounds.*
2. *W is the* StreamName$^{\uparrow}$ *finite observation window containing the named sequence entries inspected by the envelope.*
3. *R is the* RegSeqRat$^{\uparrow}$ *regular-rational readback row attached to the same window.*
4. *F is a finite family of displayed Cauchy-modulus rows whose thresholds are all read through D, W, R.*
5. *M is the finite majorant row assigning a single envelope threshold to each requested dyadic precision.*
6. *U is the finite merge row that combines a displayed finite subfamily of modulus rows into the same majorant envelope.*
7. *S is the downstream* Real$^{\uparrow}$ *seal handoff available only after D, W, R, F, M, U are accepted.*
8. *H records componentwise* hsame *transport for dyadic tolerances, stream windows, regular-rational readback, modulus rows, envelope majorants, finite merges, seal handoffs, replay, provenance, and naming.*
9. *C records displayed* Cont *replay from precision requests through the finite envelope route and then to the real-name seal.*

10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusEnvelope}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusEnvelope}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $D, W, R, F, M, U, S, H, C, P, N$. There is no coordinate for an infinite modulus choice, a quotient stream equality, a selected limit, or an ambient completeness principle.

Lean4 ✓ `BEDC.Derived.CauchyModulusEnvelopeUp.CauchyModulusEnvelopeTasteGate single carrier alignment`

Theorem 6219.2 (Cauchy modulus envelope NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ *carrier $E = (D, W, R, F, M, U, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic tolerance row, finite stream window, regular-rational readback row, finite family of modulus rows, envelope majorant row, finite merge row, real-name seal handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6219.1. Its analytic coordinates are exactly D, W, R, F, M, U, S, and its structural coordinates are exactly H, C, P, N.

The dyadic, stream, and regular-rational rows supply the only observable precision and tail-window data. The modulus family F and majorant row M are read after those rows, so an envelope threshold cannot be supplied by an external modulus selector.

The merge row U is finite and stays attached to the same displayed family. Rows H, C, P, N transport, replay, source, and name the same packet, with no additional coordinate for quotient streams, selected limits, countable choice, or ambient real completeness. Hence the listed rows exhaust the NameCert surface. □

Theorem 6219.3 (Cauchy modulus envelope majorization)**.** *Let $E = (D, W, R, F, M, U, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ *carrier. Every modulus row in the displayed finite family F is majorized only through the envelope row M over the same dyadic tolerance row D, stream window W, and regular-rational readback row R. A consumer cannot read a threshold from F without also reading the common envelope row M and the structural rows H, C, P, N.*

Proof. Begin with the carrier. The finite family F is not a free list of host functions; it is a displayed family of modulus rows attached to D, W, R.

The envelope row M is the only row that assigns a common threshold to the requested dyadic precision. Thus a majorization read must inspect D, W, R first, then read the relevant modulus row from F, and then compare it with M inside the same packet.

Transport and replay through H and C preserve this dependency, while P, N only source and name it. Since the carrier has no external modulus coordinate, every accepted majorization remains inside the finite envelope route. □

Theorem 6219.4 (Cauchy modulus envelope finite-family merge)**.** *For every accepted* $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ *carrier $E = (D, W, R, F, M, U, S, H, C, P, N)$, the merge row U combines only finite displayed subfamilies of F into the same envelope row M. The merge may reuse dyadic tolerance comparisons from D and tail-window reads from W, R, but it exports no infinite family choice, no quotient equality of modulus functions, and no selected completion witness.*

Proof. Read F and U from the carrier. The family row is finite, and the merge row is listed as a row of that same finite packet rather than as an ambient operation over all moduli.

Use the majorization theorem Definition 6219.3. Each component modulus in the displayed family is compared with M only after D, W, R are visible.

The merge row therefore combines already displayed comparisons. The structural rows H, C, P, N preserve and name the finite merge route, but they add no infinite selector, quotient equality, selected limit, or completion object. □

Theorem 6219.5 (Cauchy modulus envelope Real-seal nonescape)**.** *Let $E = (D, W, R, F, M, U, S, H, C, P, N)$ be an accepted* $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ *carrier. Every downstream* $\mathsf{Real}^{\uparrow}$ *consumer of the seal row S must first expose the dyadic tolerance row D, stream window W, regular-rational readback row R, finite modulus family F, envelope majorant row M, and finite merge row U. The seal exports no quotient Cauchy stream, selected real value, countable-choice modulus selection, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*

Proof. Project the seal row S from the carrier. It appears only after D, W, R, F, M, U, so it is a handoff reached from the accepted finite envelope packet rather than an independent real object.

Apply Definition 6219.3. It keeps every threshold read tied to the same dyadic tolerance, stream window, regular-rational readback, and envelope row.

Apply Definition 6219.4. The merge route remains finite and displays the family members whose thresholds are being bounded.

Rows H, C, P, N transport, replay, source, and name only that finite route. Thus a real-name consumer must pass through the envelope packet before reading S, and none of the refused quotient, choice, selected-limit, or ambient-completeness data is exported. □

Falsifiable prediction: An accepted $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ read is refuted if a $\mathsf{Real}^{\uparrow}$ or Cauchy-convergence consumer exports a seal row without displaying DyadicRatCore tolerance, StreamName tail windows, RegSeqRat readback, finite modulus-family rows, a common envelope majorant, finite merge data, transport, replay, provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real: none alone is bijective with $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ because the packet jointly carries dyadic precision, named tail windows, regular-rational readback, finite modulus families, envelope majorants, finite merges, Real seal handoff, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusEnvelope}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CauchyModulusEnvelope}^{\uparrow}$ packet is generated from DyadicRatCore tolerances, StreamName finite windows, RegSeqRat readback, finite Cauchy-modulus rows, a common majorant envelope, finite merge data, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CauchyModulusEnvelope}^{\uparrow}}$, seed)

Scope closed: Definition 6219.1, Definition 6219.2, Definition 6219.3, Definition 6219.4, and Definition 6219.5 seed the finite majorant-envelope surface for Cauchy moduli before Real sealing.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CauchyModulusEnvelope}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient Cauchy completion, countable choice for moduli, selected real limits, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked dyadic tolerance exposure, finite stream-window readback, regular-rational dependency, majorant exactness, finite merge exactness, Real-seal nonescape, provenance, and local naming for BEDC.Derived.CauchyModulusEnvelopeUp.

Chapter 6220

A Concrete Naming Certificate for $\mathsf{RealArctangent}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RealArctangent}^{\uparrow}$ packet records arctangent as a finite Cauchy-stream certificate rather than as a host transcendental function. The construction reads the alternating arctangent series on bounded windows through $\mathsf{RealAnalytic}^{\uparrow}$ and the standalone $\mathsf{RealSine}^{\uparrow}$ trigonometric route of chapter 6126, then seals the endpoint through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. It is placed beside the exponential and logarithm packets of chapter 1816 and chapter 2315: all three expose explicit real-analysis stream data before any public endpoint is consumed.

Definition 6220.1 (RealArctangent Cauchy carrier)**.** *A* $\mathsf{RealArctangent}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, S, R, D, E, T, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Real}^{\uparrow}$ *input row, read only through the displayed real-name boundary and bounded-window request.*
2. *S is the alternating arctangent Cauchy stream row with inspected terms $(-1)^k x^{2k+1}/(2k+1)$ on the active bounded window.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row for the rational partial-sum readback of S.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger recording denominator growth, bounded-window estimates, and alternating-remainder comparisons.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *endpoint seal reached only after the stream, regularity, and dyadic tolerance rows have been displayed.*
6. *T is the* $\mathsf{RealAnalytic}^{\uparrow}$ *and* $\mathsf{RealSine}^{\uparrow}$ *transport row that identifies the local arctangent stream with the real-analytic trigonometric dependency surface, without importing a host inverse tangent.*
7. *Q is the endpoint classifier row comparing accepted readbacks through the displayed* $\mathsf{Real}^{\uparrow}$ *boundary.*
8. *H is the componentwise* hsame *transport for $X, S, R, D, E, T, Q, C, P, N$.*
9. *C is the displayed* Cont *replay from bounded real input to stream windows, from windows to regular readback, from readback to dyadic tolerance, and from tolerance to the endpoint seal.*

10. *P is the* Pkg *provenance over* $\mathsf{RealArctangent}^{\uparrow}$, $\mathsf{RealAnalytic}^{\uparrow}$, $\mathsf{RealSine}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{RealArctangent}^{\uparrow}}$ *row.*

The classifier compares accepted arctangent packets only through $X, S, R, D, E, T, Q, H, C, P, N$. It has no coordinate for a host arctangent function, quotient real equality, analytic continuation, selected infinite sum, countable-choice schedule, unrestricted trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.RealArctangentUp.RealArctangentTasteGate single carrier alignment`

Theorem 6220.2 (RealArctangent NameCert obligations)**.** *For every accepted* $\mathsf{RealArctangent}^{\uparrow}$ *carrier $A = (X, S, R, D, E, T, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealArctangent}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet; stream exposure through S over the* $\mathsf{StreamName}^{\uparrow}$ *finite-window route; regular readback through R; dyadic tolerance accounting through D; endpoint sealing through E and Q; analytic and trigonometric dependency transport through T; componentwise* hsame *stability through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Proof. Project the accepted carrier. The only visible input, stream, regularity, tolerance, endpoint, dependency, classifier, transport, replay, provenance, and naming rows are $X, S, R, D, E, T, Q, H, C, P, N$.

The stream obligation is discharged by the finite windows in S together with the regular readback R. The dyadic ledger D records the bounded-window estimates and alternating-remainder comparisons needed before the endpoint row is available.

Use the dependency row T to route the analytic and trigonometric material through $\mathsf{RealAnalytic}^{\uparrow}$ and $\mathsf{RealSine}^{\uparrow}$. This row is a displayed BEDC dependency route, not a host inverse-transcendental primitive.

Finally E, Q, H, C, P, N seal, classify, transport, replay, source, and name exactly the same rows. Since no carrier coordinate stores a host arctangent, quotient real equality, analytic-continuation datum, selected infinite sum, countable-choice schedule, trichotomy oracle, or ambient completeness theorem, none of those objects is exported by the local NameCert surface. □

Theorem 6220.3 (RealArctangent stream dependency route)**.** *Every public read of an accepted* $\mathsf{RealArctangent}^{\uparrow}$ *packet factors through the concrete route*

$$\mathsf{RealAnalytic}^{\uparrow}, \quad \mathsf{RealSine}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

before an arctangent endpoint is consumed. Consequently the packet names arctangent by alternating Cauchy-stream data, regular rational readback, dyadic tolerance accounting, and real endpoint sealing, not by a host inverse tangent or an ambient analytic library.

Proof. Start from the carrier of Definition 6220.1. The bounded real input X and the stream row S expose the arctangent partial sums only as finite $\mathsf{StreamName}^{\uparrow}$ windows.

The readback and tolerance steps are R and D. These rows place the partial sums inside the $\mathsf{RegSeqRat}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$ surfaces before any endpoint classifier may be inspected.

The analytic dependency row T ties the local alternating stream to $\mathsf{RealAnalytic}^{\uparrow}$ and the trigonometric route already isolated by $\mathsf{RealSine}^{\uparrow}$. It supplies a dependency handoff over existing siblings rather than a primitive host arctangent.

The endpoint rows E, Q and structural rows H, C, P, N then preserve and name the same route. Thus every public arctangent read factors through the displayed analytic, sine, stream, regular, dyadic, and real rows before the value is consumed. □

Falsifiable prediction: Every $\mathsf{RealArctangent}^{\uparrow}$ public read factors through a bounded RealUp input row, alternating arctangent StreamName windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a RealAnalytic and RealSine dependency transport row, a RealUp endpoint seal, endpoint classifier, componentwise hsame transport, displayed continuation, package provenance, and local naming.

Independence witness: real-analytic, real-sine, real-exponential, real-logarithm, streamname, regseqrat, dyadicratcore, real: $\mathsf{RealArctangent}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the alternating arctangent Cauchy stream, regular rational readback, dyadic tolerance ledger, analytic and trigonometric dependency transport, endpoint classifier, Real seal, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealArctangent}^{\uparrow}}$.

Constructive story: A $\mathsf{RealArctangent}^{\uparrow}$ packet is generated from a bounded RealUp input, alternating arctangent StreamName windows, RegSeqRat regular readback, a DyadicRatCore tolerance ledger, a RealAnalytic and RealSine dependency route, a RealUp endpoint seal, endpoint classifier, componentwise transport, displayed continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealArctangent}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6220.1, Definition 6220.2, and Definition 6220.3 seed the finite arctangent Cauchy-stream route through existing real, analytic, trigonometric, stream, regularity, and dyadic-tolerance surfaces.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealArctangent}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host arctangent functions, quotient real equality, analytic continuation, selected infinite sums, countable choice, unrestricted trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem-level Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, alternating stream exposure, regular readback, dyadic tolerance accounting, analytic and trigonometric dependency transport, endpoint classifier stability, and TasteGate field faithfulness for BEDC.Derived.RealArctangentUp.

Chapter 6221

A Concrete Naming Certificate for $\mathsf{CompletionDenseRange}^{\uparrow}$

Following Bishop–Bridges constructive completion and Kelley uniform spaces, the $\mathsf{CompletionDenseRange}^{\uparrow}$ packet records the finite dense-subsource extension interface for L10 Real and Completion consumers. It sits after the extension-uniqueness surface of chapter 397, the functorial uniform-completion handoff of chapter 1892, the Cauchy-completion reflection boundary of chapter 3376, the finite stream window route of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names dense-range extension as displayed BEDC data; it does not introduce quotient completion equality, selected limits, ambient complete objects, host equality of completed points, or countable-choice representatives.

Definition 6221.1 (CompletionDenseRange carrier)**.** *A* $\mathsf{CompletionDenseRange}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, D, E, U, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *S is the source carrier row, naming the metric or uniform source whose completion-facing reads are being served.*
2. *D is the dense-subsource witness, recording finite evidence that the inspected subsurface is dense for the source requests consumed by the packet.*
3. *E is the completion-extension row, carrying the displayed extension action from the dense-subsource data into the completion-facing target.*
4. *U is the uniqueness-on-dense-source row, comparing two extension reads only after their common action on D has been displayed.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect source, dense-subsource, and extension data at the requested tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced from the same finite windows.*
7. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that bounds the finite window and readback comparisons.*
8. *H records componentwise* hsame *transport for the source, dense-subsource, extension, uniqueness, window, readback, tolerance, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a dense-source request through S, D, E, U, W, R, Q.*

10. *P is the* Pkg *provenance over* $\mathsf{CompletionDenseRange}^{\uparrow}$, $\mathsf{CompletionUniversalProperty}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}$ $\mathsf{CauchyCompletionReflection}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseRange}^{\uparrow}}$ *row.*

The carrier exports only this finite dense-range route. It has no coordinate for quotient completion equality, selected representatives, selected limits, host equality of completed values, ambient complete objects, or a completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.CompletionDenseRangeUp.CompletionDenseRangeTasteGate single carrier alignment`

Theorem 6221.2 (CompletionDenseRange NameCert obligations)**.** *For an accepted* $\mathsf{CompletionDenseRange}^{\uparrow}$ *carrier $G = (S, D, E, U, W, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompletionDenseRange}^{\uparrow}}$ *surface consists exactly of source admission, dense-subsource admission, completion-extension admission, uniqueness-on-dense-source admission, StreamName finite-window admission, RegSeqRat readback, DyadicRatCore tolerance admission, componentwise transport, displayed replay, provenance, and local naming for the rows $S, D, E, U, W, R, Q, H, C, P, N$.*

Lean4 ✓ `BEDC.Derived.CompletionDenseRangeUp.CompletionDenseRangeCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 6221.1. The analytic rows are S, D, E, U, W, R, Q, and the structural rows are H, C, P, N.

The dense-window coverage comes first. A completion-facing request reads the source carrier S and the dense-subsource witness D before the extension row E or uniqueness row U can be inspected.

The finite observation rows then bind the extension to BEDC-readable data. The row W supplies StreamName windows, R repacks those windows as regular rational readback, and Q gives the dyadic tolerance budget used by both the extension and uniqueness comparisons.

Finally H, C, P, N transport, replay, source, and name the same packet. Since the carrier has no quotient completion equality, selected representative, selected limit, host equality, ambient completion object, or $\mathsf{Real}^{\uparrow}$ completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6221.3 (CompletionDenseRange extension)**.** *Every extension read from an accepted* $\mathsf{CompletionDenseRange}^{\uparrow}$ *packet factors through the finite route*

$$S, D \longmapsto E \longmapsto W, R, Q \longmapsto H, C, P, N.$$

Thus a Real or Completion consumer receives extension data only after the dense-subsource witness, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance budget are visible.

Proof. Start with the obligation surface of Definition 6221.2. It exposes S and D before any completion-extension row is available.

The extension row E is therefore tied to the displayed dense-subsource witness. A consumer that reaches E must also carry the finite StreamName windows W, the regular rational readback R, and the dyadic tolerance row Q that make the extension readable.

The structural rows do not create a second extension. They only transport, replay, source, and name the same route, so the accepted extension read factors through $S, D, E, W, R, Q, H, C, P, N$. □

Theorem 6221.4 (CompletionDenseRange nonescape)**.** *Every uniqueness or completion-facing read from an accepted* $\mathsf{CompletionDenseRange}^{\uparrow}$ *packet is exhausted by*

$$S, D, E, U, W, R, Q, H, C, P, N.$$

In particular, the packet cannot export quotient completion equality, a selected representative, a selected limit, host equality of completed points, an ambient complete object, countable choice, or completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Use the extension theorem. By Definition 6221.3, extension reads already pass through the source, dense-subsource witness, finite windows, regular readback, and dyadic tolerance budget.

The uniqueness row U is read on the same route. It compares accepted extension rows only after their common dense-source action has been displayed through D, so it is uniqueness on the dense source rather than quotient equality of completed points.

It remains to exclude escape data. The carrier inventory has no coordinate for quotient completion equality, selected representatives, selected limits, host equality, ambient complete objects, countable choice, or $\mathsf{Real}^{\uparrow}$ completeness, and H, C, P, N preserve only the displayed rows. □

Falsifiable prediction: If a Real or Completion consumer reads a dense-range extension without first displaying the source carrier, dense-subsource witness, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance row, it is not a $\mathsf{CompletionDenseRange}^{\uparrow}$ packet.

Independence witness: completionuniversalproperty, uniformcompletionfunctor, cauchy-completion-reflection, streamname, regseqrat, dyadicratcore, real: $\mathsf{CompletionDenseRange}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source, dense-subsource, extension, uniqueness, finite-window, readback, tolerance, transport, replay, provenance, and local naming rows in one finite dense-range route.

Closure status for $\mathsf{NameCert}_{\mathsf{CompletionDenseRange}^{\uparrow}}$.

Constructive story: A $\mathsf{CompletionDenseRange}^{\uparrow}$ packet is generated from source rows, dense-subsource witness rows, completion-extension rows, uniqueness-on-dense-source rows, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompletionDenseRange}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6221.1, Definition 6221.2, Definition 6221.3, and Definition 6221.4 seed the finite dense-range extension route from dense source data to completion-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompletionDenseRange}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.CompletionDenseRangeUp`

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, selected limits, host equality, countable choice, ambient completion, $\mathsf{Real}^{\uparrow}$ completeness, theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dense-window coverage, completion-extension replay, uniqueness-on-dense-source exactness, StreamName window replay, RegSeqRat readback, DyadicRatCore tolerance control, transport preservation, and ledger refusal for BEDC.Derived.CompletionDenseRangeUp.

Chapter 6222

A Concrete Naming Certificate for $\mathsf{HadamardFinitePart}^{\uparrow}$

Following Hadamard's finite-part treatment of singular integrals and the constructive integration discipline of Bishop and Bridges, the $\mathsf{HadamardFinitePart}^{\uparrow}$ packet records a finite cutoff-and-subtraction certificate over the L10 Real and Completion surface. It sits after the Darboux and Riemann integration rows of chapter 1460 and chapter 1526, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the terminal real-name seal of chapter 38, and the bounded-variation control route of chapter 398. The packet names only the finite singular window, the displayed subtraction row, rational readback, and Real seal handoff; it is not a measure completion, a principal-value host limit, a quotient-function construction, a selected limiting process, or an ambient real-completeness theorem.

Definition 6222.1 (Hadamard finite-part carrier)**.** *A* $\mathsf{HadamardFinitePart}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, S, K, A, B, R, D, E, V, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-source row, including the endpoint seals and the named singular location or finite singular subwindow.*
2. *S is the singular-kernel row, recording the local pole order or displayed asymptotic term to be subtracted.*
3. *K is the cutoff row, a finite family of excluded neighborhoods and retained compact subintervals.*
4. *A is the subtraction row, displaying the finite counterterm or polynomial singular part removed on the retained windows.*
5. *B is the Darboux or Riemann integration handoff row for the cutoff integrand on the retained finite partition.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the cutoff-and-subtracted values.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the cutoff width, subtraction budget, rational readback, and Real seal comparison.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, public only after K, A, B, R, D have been displayed.*
9. *V is the* $\mathsf{BoundedVariation}^{\uparrow}$ *or variation-control row used to justify that the retained window remains a finite integration object.*

10. *H stores componentwise* hsame *transport for $I, S, K, A, B, R, D, E, V$.*
11. *C stores displayed* Cont *replay from a finite-part request through cutoff selection, subtraction, integration handoff, rational readback, tolerance comparison, and Real seal.*
12. *P is the* Pkg *provenance over* $\mathsf{HadamardFinitePart}^{\uparrow}$, $\mathsf{DarbouxIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{HadamardFinitePart}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. No coordinate stores an ambient principal-value limit, measure completion, quotient function, chosen limiting sequence, host distribution object, or a proof that $\mathsf{Real}^{\uparrow}$ *is complete.*

Lean4 (definition) `BEDC.Derived.HadamardFinitePartUp.HadamardFinitePartUp`
Lean4 ✓ `BEDC.Derived.HadamardFinitePartUp.HadamardFinitePartTasteGate single carrier alignment`

Theorem 6222.2 (Hadamard finite-part NameCert obligations)**.** *For every accepted* $\mathsf{HadamardFinitePart}^{\uparrow}$ *carrier $F = (I, S, K, A, B, R, D, E, V, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HadamardFinitePart}^{\uparrow}}$ *surface consists exactly of interval-source admission, singular-kernel exposure, finite cutoff visibility, subtraction-row exposure, Darboux or Riemann handoff, regular rational readback, dyadic tolerance accounting, Real seal routing, bounded-variation control, componentwise transport, displayed replay, package provenance, and local naming. A finite-part read is accepted only after the cutoff row K, subtraction row A, integration handoff B, readback row R, and tolerance ledger D are all displayed.*

Proof. Project the carrier of Definition 6222.1. The interval row I and singular-kernel row S identify the singular site before any finite-part value can be read.

The cutoff and subtraction rows are then exposed. The row K records the retained windows, while A records the displayed singular term removed on those windows. The integration row B is therefore a Darboux or Riemann handoff for a finite cutoff object rather than an ambient singular integral.

Next R, D, E, V provide regular rational readback, tolerance accounting, the terminal Real seal, and bounded-variation control. Finally H, C, P, N transport, replay, source, and name the same finite route. Since none of these rows contains a principal-value host limit, measure completion, quotient function, selected limiting sequence, distribution object, or ambient real-completeness proof, the listed rows exhaust the local obligations. □

Theorem 6222.3 (Hadamard finite-part cutoff nonescape)**.** *Every* $\mathsf{HadamardFinitePart}^{\uparrow}$ *consumer that reads an accepted carrier must factor its singular request through the finite route*

$$\mathsf{Real}^{\uparrow},\ \mathsf{BoundedVariation}^{\uparrow} \longmapsto \mathsf{DarbouxIntegral}^{\uparrow} \text{ or } \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

using the displayed rows $I, S, K, A, B, R, D, E, V, H, C, P, N$. In particular, deleting K or A leaves no public finite-part read.

Proof. Begin with the NameCert obligations of Definition 6222.2. They place the finite cutoff K and subtraction row A before the integration handoff and before any rational readback is available.

The Darboux or Riemann row B can then be read only as the cutoff-and-subtracted integral surface. The rows R and D translate that surface into regular rational and dyadic tolerance data before the seal E is public.

The structural rows H, C, P, N preserve exactly that order. Without K the retained finite window is absent; without A the singular counterterm is absent. In either case the carrier has no coordinate from which a finite-part value can be exported. □

Theorem 6222.4 (Hadamard finite-part Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{HadamardFinitePart}^{\uparrow}$ *packet reaches the Real seal* E *only after the displayed cutoff row, subtraction row, integration handoff, regular rational readback, dyadic tolerance ledger, and bounded-variation control row have been consumed. The handoff supplies a finite singular-read certificate, not a measure completion, host principal-value limit, quotient-function equality, selected limit, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. By Definition 6222.1, the only Real-facing output coordinate is E. The coordinate E occurs after the cutoff, subtraction, integration, readback, tolerance, and variation-control rows in the displayed carrier.

Apply Definition 6222.3. Any consumer of E must first traverse K, A, B, R, D, V together with the structural rows. Therefore the Real seal is a terminal read of the finite certificate and not a host analytic limit. □

Falsifiable prediction: Every accepted $\mathsf{HadamardFinitePart}^{\uparrow}$ singular read is refuted if a Real seal is cited without the displayed cutoff row, subtraction row, Darboux or Riemann handoff, RegSeqRat readback, DyadicRatCore tolerance ledger, and bounded-variation control row.

Independence witness: darboux-integral, riemann-integral, regseqrat, dyadicratcore, real, bounded-variation: none is bijective to $\mathsf{HadamardFinitePart}^{\uparrow}$ because this packet jointly carries a singular-window row, cutoff row, subtraction row, integration handoff, rational readback, tolerance ledger, Real seal, variation-control row, transport, replay, provenance, and local naming as one finite-part certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{HadamardFinitePart}^{\uparrow}}$**.**

Constructive story: A $\mathsf{HadamardFinitePart}^{\uparrow}$ packet is generated from a Real interval source, singular-kernel row, finite cutoff row, subtraction row, Darboux or Riemann integration handoff, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, bounded-variation control, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HadamardFinitePart}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6222.1, Definition 6222.2, Definition 6222.3, and Definition 6222.4 seed the finite cutoff-and-subtraction route for Hadamard finite-part singular reads over the L10 Real and Completion surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HadamardFinitePart}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure completion, principal-value host limits, quotient functions, selected limits, host distributions, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked interval-source admission, singular-kernel exposure, cutoff visibility, subtraction-row exposure, Darboux or Riemann handoff, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal routing, bounded-variation control, transport, replay, provenance, and local naming for BEDC.Derived.HadamardFinitePartUp.

Chapter 6223

A Concrete Naming Certificate for $\mathsf{RegulatedMeshModulus}^{\uparrow}$

Following Riemann and Darboux, a regulated function is useful for constructive integration only when a finite mesh request can be read before any integral value is sealed. The $\mathsf{RegulatedMeshModulus}^{\uparrow}$ packet records that mesh-control surface between the regulated-function source of chapter 1906, the finite partition mesh of chapter 3496, the tagged partition observations of chapter 4296, and the integration-facing handoff of chapter 1960. It names a finite route for mesh-controlled tagged sums; it is not a host uniform-continuity theorem, an arbitrary refinement principle, a selected limiting integral, or an ambient completeness claim.

Definition 6223.1 (Regulated mesh modulus carrier)**.** *A* $\mathsf{RegulatedMeshModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, G, T, I, H, C, P, N)$$

with the following displayed rows:

1. *F is the accepted* $\mathsf{RegulatedFunction}^{\uparrow}$ *source row whose left and right local reads are the only function data available to the packet.*
2. *G is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *mesh-control row, assigning each displayed oscillation request a finite partition mesh bound for the same regulated source.*
3. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *observation row, reading tagged partition samples only at partitions whose mesh has passed through G.*
4. *I is the* $\mathsf{RiemannIntegrable}^{\uparrow}$ *handoff row, available only after F, G, T have been displayed.*
5. *H records componentwise* hsame *transport for F, G, T, I.*
6. *C records displayed* Cont *replay from regulated reads through mesh bounds, tagged partition observations, the integration handoff, and the structural rows.*
7. *P is the* Pkg *provenance over* $\mathsf{RegulatedMeshModulus}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{RiemannIntegrable}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegulatedMeshModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through F, G, T, I, H, C, P, N. It has no coordinate for host uniform continuity, arbitrary refinement choice, quotient equality of tagged sums, selected integral values, countable choice, open-cover compactness, or ambient real completeness.

Lean4 ✓ BEDC.Derived.RegulatedMeshModulusUp.RegulatedMeshModulusTasteGate single carrier alignment

Theorem 6223.2 (Regulated mesh modulus obligations)**.** *For every accepted* $\mathsf{RegulatedMeshModulus}^{\uparrow}$ *carrier*

$$M = (F, G, T, I, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegulatedMeshModulus}^{\uparrow}}$ *surface consists exactly of carrier admission through* M*, regulated-function exposure through* F*, finite mesh-control exposure through* G*, tagged partition observation through* T*, Riemann-facing handoff through* I*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, provenance through* P*, and local naming through* N*. No obligation exports a host uniform-continuity modulus, arbitrary partition refinement, quotient equality of sums, a selected integral, countable choice, compactness, or ambient completeness.*

Proof. Project the carrier of Definition 6223.1. The analytic rows are precisely the regulated source, the finite partition mesh, the tagged partition observations, and the Riemann handoff.

A mesh request is answered by G before tagged samples are inspected. Thus the row T can read only partitions whose mesh-control route has already been displayed for the same regulated source F.

The handoff row I is downstream of F, G, T. The structural rows H, C, P, N transport, replay, source, and name that same finite packet; they do not add any of the excluded host principles. □

Theorem 6223.3 (Regulated mesh modulus Riemann handoff)**.** *Every* $\mathsf{RiemannIntegrable}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegulatedMeshModulus}^{\uparrow}$ *carrier factors through the finite route*

$$F, G, T \longmapsto I, H, C, P, N.$$

The consumer first reads the regulated-function source, then the finite partition mesh bound, then the tagged real partition observations, and only then receives the integration-facing handoff. The route exports mesh-controlled tagged-sum data, not a selected integral value, host refinement theorem, quotient equality of sums, countable choice, compactness, or ambient real completeness.

Proof. Begin with a Riemann-facing consumer of M. By the carrier definition, the consumer can inspect the regulated source only through F and can request partition control only through the mesh row G.

The tagged row T then supplies observations attached to that mesh. Because T is downstream of G, the consumer cannot replace the mesh-control step by an arbitrary refinement or by a host limiting argument.

Finally the handoff row I is sealed by H, C, P, N as a local NameCert route. Since no selected integral, quotient, choice, compactness, or ambient-completeness coordinate occurs in M, the Riemann-facing output is exactly the displayed finite handoff. □

Falsifiable prediction: A $\mathsf{RegulatedMeshModulus}^{\uparrow}$ consumer that reads $\mathsf{RiemannIntegrable}^{\uparrow}$ before displaying F, G, T, or that exports selected integrals, quotient equality of sums, arbitrary refinement choice, compactness, countable choice, or ambient completeness, refutes the carrier of Definition 6223.1.

Independence witness: Nearest siblings: regulated-function, finite-partition-mesh, tagged-real-partition, and riemann-integrable. $\mathsf{RegulatedMeshModulus}^{\uparrow}$ is not bijective to any one sibling because it jointly carries regulated-function source data, finite mesh control, tagged partition observations, Riemann-facing handoff, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RegulatedMeshModulus}^{\uparrow}}$.

Constructive story: A RegulatedMeshModulus$^{\uparrow}$ packet is generated from a RegulatedFunction source row, a FinitePartitionMesh mesh-control row, a TaggedRealPartition observation row, a RiemannIntegrable handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegulatedMeshModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6223.1, Definition 6223.2, and Definition 6223.3 seed the finite regulated-function mesh-control route before Riemann-facing handoff reads.

Formal verification: FormalStatus(NameCert$_{\text{RegulatedMeshModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host uniform continuity, arbitrary refinement choice, quotient equality of tagged sums, selected integral values, countable choice, compactness, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-function exposure, mesh-control exactness, tagged partition observation, Riemann-facing handoff ordering, transport, replay, provenance, and local naming for BEDC.Derived.RegulatedMeshModulusUp.

Chapter 6224

A Concrete Naming Certificate for $\mathsf{Epigraph}^{\uparrow}$

Following Rockafellar's convex-analysis terminology, the $\mathsf{Epigraph}^{\uparrow}$ packet names the epigraph of a displayed extended-real-valued map as finite BEDC rows. It is read through domain points, function-value observations, lower-bound comparisons, and package provenance rather than through a host subset of a product space. The packet supplies the finite epigraph interface consumed by the lower-semicontinuity route of chapter 1536, and its value and comparison rows terminate at the real-name boundary of chapter 38.

Definition 6224.1 (Epigraph carrier)**.** *An* $\mathsf{Epigraph}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, V, L, O, H, C, P, N)$$

with the following displayed rows:

1. *D is the domain-point row naming the source object and the inspected point.*
2. *V is the function-value row, read through the relevant displayed value carrier and sealed at the* $\mathsf{Real}^{\uparrow}$ *boundary when a real comparison is requested.*
3. *L is the lower-bound row naming the candidate ordinate for the epigraph inspection.*
4. *O is the comparison row recording that the displayed lower bound is below the displayed value row.*
5. *H stores componentwise* hsame *transport for the domain point, value, lower bound, comparison, replay, provenance, and local naming rows.*
6. *C stores displayed* Cont *replay from an epigraph query through D, V, L, O, H.*
7. *P is the* Pkg *provenance over* $\mathsf{Epigraph}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *dependency rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. No coordinate stores a host graph, a powerset of a product, quotient equality of functions, a selected minimizing point, ambient real completeness, or an external lower-semicontinuity theorem.

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphCarrier rows exposed`

↪ variant `BEDC.Derived.EpigraphUp.EpigraphCarrierRowTransportSurface`

Theorem 6224.2 (Epigraph NameCert obligations)**.** *For every accepted* $\mathsf{Epigraph}^{\uparrow}$ *carrier* $E = (D, V, L, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, classifier comparison through the domain-point, value, lower-bound, and comparison rows, componentwise transport through* H*, lower-set exposure through the comparison row* O*, and ledger exactness through* C, P, N*. A consumer may inspect epigraph membership only after the domain point, function value, lower bound, and comparison rows are displayed, and the packet exports no host subset, graph quotient, selected minimizer, ambient completeness theorem, or hidden function-space equality.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphNamecertObligations`
↪variant `BEDC.Derived.EpigraphUp.EpigraphCarrier namecert obligations`

Proof. Project the carrier of Definition 6224.1. The only mathematical rows before the structural data are D, V, L, O, so carrier admission and classifier comparison must read the domain point, value, lower bound, and comparison directly.

The lower-set clause is exactly the comparison row O. Transport by H preserves that comparison componentwise, while C replays the same displayed route from an epigraph query to the visible rows.

The provenance and local naming rows P, N source and name this finite packet. Since no carrier coordinate contains a host subset, graph quotient, selected minimizer, completeness theorem, or function-space equality, the listed rows exhaust the NameCert obligations. □

Theorem 6224.3 (Epigraph lower-semicontinuity handoff)**.** *Every* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer that reads open-sublevel or closed-superlevel obligations through an accepted* $\mathsf{Epigraph}^{\uparrow}$ *packet receives only the finite route*

$$D, V, L, O \longmapsto H, C, P, N.$$

The handoff supplies the epigraph row consumed by the lower-semicontinuity certificate: a displayed domain point, a displayed value row, a lower-bound row, a comparison row, componentwise transport, replay, provenance, and local naming. It does not supply host open sets, a quotient graph, arbitrary function extensionality, selected witnesses, or ambient completeness of $\mathsf{Real}^{\uparrow}$*. The comparison row* O *is the finite lower-set surface consumed by chapter 1536, and its value comparison terminates at the real-name boundary of chapter 38.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphLowerSemicontinuousHandoff`

Proof. Use Definition 6224.2. It forces every epigraph membership read to expose D, V, L, O before the structural rows may replay or source the answer.

Now pass to the $\mathsf{LowerSemicontinuous}^{\uparrow}$ consumer. The open-sublevel and closed-superlevel obligations read the epigraph-facing comparison as the finite row O, together with the value and lower-bound rows that make the comparison meaningful at the real-name boundary.

The structural rows H, C, P, N preserve and name that route only. Because the carrier has no host topology, quotient graph, arbitrary function-extensional equality, selected witness coordinate, or ambient completeness field, the lower-semicontinuity consumer receives exactly the displayed epigraph packet. □

Definition 6224.4 (Epigraph classifier stability)**.** *For an accepted* $\mathsf{Epigraph}^{\uparrow}$ *carrier* $E = (D, V, L, O, H, C, P, N)$*, classifier stability is the requirement that every repeated epigraph-membership read compares the same domain-point row* D*, value row* V*, lower-bound row* L*, and comparison row* O*, transported*

only through H and replayed only through C. The stable classifier is therefore the finite row inventory governed by Definition 6224.2; it is not a host topology, quotient graph, arbitrary function equality, selected minimizer, or ambient completeness claim.

Lean4 (definition) `BEDC.Derived.EpigraphUp.EpigraphClassifierStability`

Theorem 6224.5 (Epigraph lower-set transport)**.** *Every lower-set read exported by an accepted* $\mathsf{Epigraph}^{\uparrow}$ *packet preserves the rows D, V, L, O, H, C, P, N before a* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *consumer may inspect it. In particular, the comparison row O can be transported only together with the displayed domain point, value row, lower-bound row, componentwise transport, replay, provenance, and local naming rows required by Definition 6224.2.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphLowerSetTransport`

Proof. Project the carrier of Definition 6224.1. The lower-set information appears only as the comparison row O attached to D, V, L.

Apply Definition 6224.2. Its lower-set exposure clause keeps O inside the local $\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}$ surface, while H transports that comparison componentwise and C replays the same route.

The provenance and naming rows P, N source the transported packet. Since no host subset, quotient graph, arbitrary function equality, selected minimizer, or completeness coordinate is present, lower-set transport cannot leave the displayed rows. □

Theorem 6224.6 (Epigraph ledger exactness)**.** *For every accepted* $\mathsf{Epigraph}^{\uparrow}$ *carrier $E = (D, V, L, O, H, C, P, N)$, the ledger rows C, P, N are exact for the classifier of Definition 6224.4: they replay, source, and name only the displayed domain-point, value, lower-bound, comparison, and transport rows. Together with Definition 6224.2, these rows close the obligation surface without asserting host topology, quotient graph equality, arbitrary function extensionality, selected minimizers, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphLedgerExactness`

Proof. Read the classifier stability definition. It allows repeated reads only through D, V, L, O, H, C, P, N.

Use Definition 6224.2. The NameCert surface already identifies C as replay, P as provenance, and N as local naming for the displayed carrier rows.

Therefore the ledger is exact by exhaustion of carrier coordinates. No additional row can name host topology, quotient graph equality, arbitrary function extensionality, selected minimizers, or ambient completeness, so the obligation surface remains finite. □

Theorem 6224.7 (Epigraph scoped kernel dependencies)**.** *Every* $\mathsf{Epigraph}^{\uparrow}$ *consumer route factors through the displayed domain-point rows, function-value observation rows, lower-bound comparison rows,* $\mathsf{Real}^{\uparrow}$ *boundary rows,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows of Definition 6224.1. The route exports no host subset of a product, function extensionality principle, selected minimizer, or ambient real-completeness principle.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphScopedKernelDependencies`

Proof. Start with the carrier in Definition 6224.1. Its analytic rows are exactly D, V, L, O, with V and O reaching the $\mathsf{Real}^{\uparrow}$ boundary only through displayed value and lower-bound comparisons.

Apply Definition 6224.3 to pass any lower-semicontinuity consumer through the same displayed value and comparison route. Then use Definition 6224.4 and Definition 6224.5 to keep repeated reads and transported lower-set reads on D, V, L, O, H, C, P, N.

Finally apply Definition 6224.6. The replay, provenance, and naming rows are exact for the displayed carrier, so no additional coordinate can supply a host product subset, function extensionality, selected minimizer, or ambient completeness principle. □

Theorem 6224.8 (Epigraph mature crosslink)**.** *The comparison row O of an accepted* $\mathsf{Epigraph}^{\uparrow}$ *packet is the lower-set comparison row consumed by both the lower-semicontinuity route of chapter 1536 and convex-analysis sibling routes such as chapter 208. Any such consumer must read O together with the displayed domain point D, value row V, lower-bound row L, componentwise transport H, replay C, provenance P, and local naming N before interpreting a lower-set or convex-level comparison.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphMatureCrosslink`

Proof. Start with Definition 6224.7. It keeps every epigraph consumer on the displayed rows D, V, L, O, H, C, P, N and refuses host subsets, function extensionality, selected minimizers, and ambient real completeness.

For lower-semicontinuity, apply Definition 6224.3. The open-sublevel and closed-superlevel reads receive the comparison row O only with its domain, value, and lower-bound rows still present.

For convex-analysis consumers, the same row is read as a finite lower-set comparison before any convex-level route can inspect it. The convex sibling therefore receives a displayed comparison surface, not a host epigraph, quotient graph, selected minimizer, or external convexity theorem. □

Theorem 6224.9 (Epigraph mature consumer completeness)**.** *Every lower-semicontinuity, convex-level, and lower-set consumer of an accepted* $\mathsf{Epigraph}^{\uparrow}$ *packet receives exactly the rows*

$$D, V, L, O, H, C, P, N$$

through the carrier of Definition 6224.1, the lower-semicontinuity handoff of Definition 6224.3, the lower-set transport of Definition 6224.5, the ledger exactness of Definition 6224.6, and the mature crosslink of Definition 6224.8. No consumer receives a host epigraph subset, quotient graph, selected minimizer, ambient $\mathsf{Real}^{\uparrow}$ *completeness principle, or function-extensional equality outside those rows.*

Lean4 ✓ `BEDC.Derived.EpigraphUp.EpigraphMatureConsumerCompleteness`

Proof. The carrier displays only D, V, L, O, H, C, P, N. By Definition 6224.2, every epigraph read must expose the domain point, function value, lower bound, comparison, transport, replay, provenance, and local naming rows before a consumer can inspect membership.

For lower-semicontinuity consumers, Definition 6224.3 routes the open-sublevel and closed-superlevel read through the same value and comparison rows. For lower-set consumers, Definition 6224.5 keeps the comparison row O attached to D, V, L, H, C, P, N.

Convex-level consumers use the crosslink of Definition 6224.8. Ledger exactness from Definition 6224.6 then says that replay, provenance, and local naming source only this displayed packet. The listed consumers therefore receive exactly the carrier rows and none of the excluded host objects. □

Falsifiable prediction: An accepted $\mathsf{Epigraph}^{\uparrow}$ read is refuted if the comparison row O is preserved while the displayed domain point D, value row V, and lower-bound row L are removed.

Independence witness: lowersemicontinuous, real: neither sibling is bijective with $\mathsf{Epigraph}^{\uparrow}$ because this packet jointly carries a domain-point row, function-value row, lower-bound row, comparison row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}$**.**

Constructive story: An $\mathsf{Epigraph}^{\uparrow}$ packet is generated from a domain-point row, a function-value row, a lower-bound row, a comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}, \mathsf{maturePackageClosed})$

Scope closed: Definition 6224.1, Definition 6224.2, Definition 6224.3, Definition 6224.4, Definition 6224.5, Definition 6224.6, Definition 6224.7, and Definition 6224.9 close the finite epigraph dependency scope and expose the lower-semicontinuity, convex-level, and lower-set consumer routes.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{Epigraph}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: paperBridge

Not claimed: This chapter does not close host epigraph topology, quotient graphs, arbitrary function extensionality, selected minimizers, ambient real completeness, Lean verification, or bridge checking.

Upgrade path: No further paper-side closure remains; Lean verification and bridge checking remain separate.

Chapter 6225

A Concrete Naming Certificate for $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$

The $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$ packet realizes the practice-level audit taxonomy of section 9462.12 as finite NameCert data. It is distinct from the positive-discovery classifier of chapter 6202: that chapter accepts a positive interpretability discovery only after classifier novelty and net-information accounting are displayed, while this packet classifies the practice event that changed the audit surface itself. It sits beside the classifier novelty witness of chapter 6138, the discovery pipeline calculus of chapter 5914, and the effective replacement ledger of chapter 568.

Definition 6225.1 (Positive interpretability classifier practice carrier)**.** *A* $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (E_0, E_1, J, T, L, Q, D, B, H, C, R, N)$$

with the following displayed rows:

1. E_0 *is the before-practice explanation-state row, including the classifier label then used by the interpretability report.*
2. E_1 *is the after-practice explanation-state row, including the classifier label after the practice change.*
3. J *is the* $\mathsf{ClassifierNoveltyWitness}^{\uparrow}$ *row exposing a common-source comparison and the before/after classifier disagreement.*
4. T *is the practice-taxonomy row, distinguishing feature event, classifier event, certificate event, ledger event, verification event, optimization lineage, discovery lineage, and positive discovery lineage.*
5. L *is the effective replacement ledger row recording which practice row is used by downstream reports rather than merely named.*
6. Q *is the public cost row, recording the audit cost of changing the practice classifier and the public rows needed to reproduce the classification.*
7. D *is the debt row separating classifier debt, ledger debt, semantic debt, verification debt, intervention debt, and generalization debt.*
8. B *is the benefit row recording the exposed coverage, prediction, transfer, intervention, verification, or unification gain.*

9. *H stores componentwise* hsame *transport for $E_0, E_1, J, T, L, Q, D, B$.*
10. *C stores displayed* Cont *replay through before/after classifier comparison, practice-taxonomy selection, ledger lookup, public-cost accounting, debt exposure, and benefit exposure.*
11. *R is the* Pkg *provenance over* PositiveInterpretabilityClassifierPractice$^{\uparrow}$, PositiveInterpretabilityClassifier$^{\uparrow}$, ClassifierNoveltyWitness$^{\uparrow}$, DiscoveryPipelineCalculus$^{\uparrow}$, EffectiveReplacementLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* NameCert$_{\text{PositiveInterpretabilityClassifierPractice}^{\uparrow}}$ *row.*

The carrier has no coordinate for a prose-only practice shift, a positive-discovery verdict without before/after classifier disagreement, a hidden cost convention, a ledger row with no downstream use, or a classifier-practice claim without public debt and benefit rows.

Theorem 6225.2 (Positive interpretability classifier practice NameCert obligations)**.** *For every accepted* PositiveInterpretabilityClassifierPractice$^{\uparrow}$ *carrier $P = (E_0, E_1, J, T, L, Q, D, B, H, C, R, N)$, the local* NameCert$_{\text{PositiveInterpretabilityClassifierPractice}^{\uparrow}}$ *surface consists exactly of before-state admission, after-state admission, classifier-novelty witness, practice-taxonomy exposure, effective replacement ledger, public cost row, debt row, benefit row, componentwise transport, displayed replay, package provenance, and local naming. A practice-level classifier shift is accepted only when J displays before/after classifier disagreement and Q, L, D, B display the public cost and ledger rows used by the report.*

Proof. Project the carrier of Definition 6225.1. The rows E_0, E_1, J expose the before state, after state, and common-source classifier disagreement.

Next read the practice rows. The taxonomy row T says which practice event is being classified, while L records downstream use and Q, D, B record public cost, debt, and benefit.

Finally H, C, R, N transport, replay, source, and name the same packet. Since no other coordinate can supply a hidden practice shift, hidden cost convention, ledger-only completion, or positive-discovery verdict, the listed rows are exactly the local obligations. □

Theorem 6225.3 (Practice classifier separation from positive discovery)**.** *An accepted* PositiveInterpretabilityClassifi *packet classifies the practice event that changes the interpretability audit surface; it does not by itself classify the result as* PositiveInterpretabilityClassifier$^{\uparrow}$. *A downstream positive-discovery read must separately supply the positive classifier's net-information rows and cannot replace them by the practice-taxonomy row T.*

Proof. Use Definition 6225.2. The practice packet carries T as an event-kind row and records before/after classifier disagreement through J.

The positive-discovery packet of chapter 6202 requires its own cost-protocol and net-information rows. The practice row T is not one of those rows, and the carrier here has no coordinate that asserts positive net information.

Thus the practice classifier can record a classifier-practice shift without accepting a positive-discovery verdict. Any downstream positive claim must pass through the separate positive classifier surface. □

Theorem 6225.4 (Practice classifier public-cost ledger nonescape)**.** *Every public* PositiveInterpretabilityClassifierPra *read factors through the before/after classifier disagreement row J, the practice-taxonomy row T, the effective replacement ledger L, the public cost row Q, and the debt and benefit rows D, B. If any one of J, T, L, Q, D, B is absent, the practice-level classifier shift is not public.*

Proof. Begin with the carrier definition. The only rows that expose the practice shift are J and T, and the only rows that expose public cost and use are L, Q, D, B.

Apply the NameCert obligations. They force any accepted practice read to consume all six rows before replay, provenance, and local naming can certify the packet.

The structural rows H, C, R, N preserve the same route and do not add missing classifier, cost, ledger, debt, or benefit data. Therefore deleting any one of J, T, L, Q, D, B prevents the read from being public. □

Falsifiable prediction: An accepted $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$ packet is refuted by a pair of before/after explanation states whose classifier labels agree, or by a report that hides the public cost row, effective replacement ledger, debt row, or benefit row used to classify the practice event.

Independence witness: positive-interpretability-classifier, classifier-novelty-witness, discovery-pipeline-calculus, effective-replacement-ledger: none is bijective to $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$ because this packet jointly carries before/after explanation states, a practice-taxonomy row, classifier disagreement, public cost, downstream ledger use, debt and benefit rows, transport, replay, provenance, and local naming as a practice-level classifier.

The Lean follow-up must provide $\mathsf{Nontrivial\ PositiveInterpretabilityClassifierPractice}^{\uparrow}$ and $\mathsf{FieldFaithful\ PositiveInte}$ instances before checked markers for $\mathsf{BEDC.Derived.PositiveInterpretabilityClassifierPracticeUp}$ are cited. Those instances must preserve $E_0, E_1, J, T, L, Q, D, B, H, C, R, N$ as distinct fields, so before/after classifier disagreement, public cost, ledger use, debt, benefit, and practice taxonomy cannot collapse into a one-row carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}}$ **·**

Constructive story: A $\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}$ packet is generated from before and after explanation states, a classifier novelty witness, a practice-taxonomy row, an effective replacement ledger, public cost, debt and benefit rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6225.1, Definition 6225.2, Definition 6225.3, and Definition 6225.4 seed the finite practice-level classifier for interpretability audit shifts.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PositiveInterpretabilityClassifierPractice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the positive interpretability discovery classifier, global interpretability quality, hidden semantic truth, prose-only practice shifts, uncounted costs, undisplayed debts, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked before and after state admission, classifier-disagreement exposure, practice-taxonomy exposure, public cost, effective replacement ledger, debt and benefit rows, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for $\mathsf{BEDC.Derived.PositiveInterpretabilityClassifierPracticeUp}$.

Chapter 6226

A Concrete Naming Certificate for $\mathsf{CauchyModulusCompleteness}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion, the $\mathsf{CauchyModulusCompleteness}^{\uparrow}$ packet records the finite handoff from an explicit Cauchy modulus to a completion-facing Real seal. It sits after the effective completion route of chapter 2026, the convergence-modulus ledger of chapter 2074, the criterion/modulus comparison of chapter 3343, the real Cauchy criterion of chapter 1590, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite NameCert route for modulus-controlled completion consumers; it is not an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$ and it does not create quotient streams, selected limits, countable choice, trichotomy, or host real equality.

Definition 6226.1 (CauchyModulusCompleteness carrier)**.** *A* $\mathsf{CauchyModulusCompleteness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (M, E, K, W, R, D, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *row, recording the finite modulus request and the threshold data accepted by the completion-facing read.*
2. *E is the* $\mathsf{EffectiveCompletion}^{\uparrow}$ *row, exposing the effective completion handoff through which the modulus data reaches a Real-facing consumer.*
3. *K is the* $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$ *or* $\mathsf{RealCauchyCriterion}^{\uparrow}$ *row, binding the same modulus request to a finite Cauchy-criterion read.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by M, E, K before any readback or Real seal is inspected.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed windows and consumed by the completion route.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic scales, split radii, slack, and comparison witnesses shared by M, W, R.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, reached only after M, E, K, W, R, D have been displayed.*

8. *H records componentwise* hsame *transport for modulus, effective completion, criterion, window, readback, tolerance, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$M, E, K \longmapsto W, D \longmapsto R \longmapsto L.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusCompleteness}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{EffectiveCompletion}^{\uparrow}$, $\mathsf{CauchyCriterionEquivalence}^{\uparrow}$, $\mathsf{RealCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusCompleteness}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, E, K, W, R, D, L, H, C, P, N$. It has no coordinate for ambient $\mathsf{Real}^{\uparrow}$ *completeness, quotient streams, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyModulusCompletenessUp.TasteGate.CauchyModulusCompletenessTasteGate`
`single carrier alignment`

Theorem 6226.2 (CauchyModulusCompleteness NameCert obligations)**.** *For every accepted* CauchyModulusComp *carrier*

$$X = (M, E, K, W, R, D, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusCompleteness}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed Cauchy modulus, effective completion, criterion, finite-window, regular readback, tolerance, real-seal, transport, replay, provenance, and local naming rows;*

2. *modulus visibility, requiring every completion-facing request to expose M before any finite window, readback, or Real seal is consumed;*

3. *criterion and completion compatibility, requiring E and K to bind the same modulus request, finite windows, regular readback, dyadic tolerance data, and terminal Real seal;*

4. *window-readback-tolerance exactness, requiring every read used by L to pass through the displayed* $\mathsf{StreamName}^{\uparrow}$ *row W,* $\mathsf{RegSeqRat}^{\uparrow}$ *row R, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *row D;*

5. *structural non-escape, requiring H, C, P, N to transport, replay, source, and name only the displayed route and to export no ambient* $\mathsf{Real}^{\uparrow}$ *completeness, quotient stream, selected limit, countable-choice schedule, trichotomy branch, host real equality, Lean verification credential, or bridge credential.*

No Cauchy modulus completeness obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6226.1. The accepted packet displays exactly the Cauchy modulus, effective completion, criterion, finite-window, readback, tolerance, Real seal, and structural rows.

The modulus-visibility clause is read from M. Since W, R, D, L occur downstream of the displayed modulus route, a completion-facing consumer cannot inspect stream data, rational readback, dyadic tolerance, or a Real seal before the modulus request is visible.

The compatibility and exactness clauses are read from E, K, W, R, D, L. The effective completion row E and the criterion row K bind the same request to finite windows W, regular readback R, and dyadic tolerance data D before the terminal seal L is reached.

The structural clause is exhausted by H, C, P, N. These rows only transport, replay, source, and name the displayed route, and the carrier has no coordinate for the excluded completeness, quotient, selection, choice, trichotomy, host-equality, verification, or bridge data. □

Theorem 6226.3 (CauchyModulusCompleteness Real seal handoff)**.** *For every accepted* CauchyModulusCompleten *carrier*

$$X = (M, E, K, W, R, D, L, H, C, P, N),$$

every completion-facing consumer must pass through the displayed route

$$M, E, K \longmapsto W, D \longmapsto R \longmapsto L$$

with transport, replay, provenance, and local naming supplied only by H, C, P, N before reading the terminal $\mathsf{Real}^{\uparrow}$ *seal. The consumer receives no ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, quotient stream equality, selected limit, countable-choice schedule, trichotomy branch, or host real equality.*

Proof. Fix a completion-facing consumer of X. The only terminal Real-facing coordinate is L, so any allowed read of a Real seal must reach that row through the displayed carrier route.

Read the dependencies of L. The seal is downstream of the finite-window row W, the dyadic tolerance row D, and the regular readback row R.

The source of those rows is fixed by M, E, K. The modulus row M exposes the Cauchy-modulus request, E carries the effective completion handoff, and K records the criterion route binding the same request to the same finite data.

The structural rows add no analytic data. Componentwise transport through H preserves the displayed coordinates, replay through C preserves the route order, and P, N only source and name the same packet.

Since no excluded object appears as a carrier coordinate, the consumer cannot obtain ambient Real completeness, quotient stream equality, a selected limit, countable choice, trichotomy, or host real equality. Thus every completion-facing consumer passes through the displayed modulus, window, readback, tolerance, and Real-seal route. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusCompleteness}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusCompleteness}^{\uparrow}$ packet is generated from a CauchyModulus row, an EffectiveCompletion row, a CauchyCriterionEquivalence or RealCauchyCriterion row, StreamName finite windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusCompleteness}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6226.1, Definition 6226.2, and Definition 6226.3 seed the finite modulus-controlled completion handoff from CauchyModulus data to a Real-facing seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusCompleteness}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient streams, selected limits, countable choice, trichotomy, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, modulus visibility, completion and criterion compatibility, window-readback-tolerance exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyModulusCompletenessUp.

Chapter 6227

A Concrete Naming Certificate for $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$

Following Specker's bounded monotone sequence construction and Bishop–Bridges constructive analysis, the $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ packet names the finite boundary between bounded monotone rational stream data and any later located-supremum or completion-facing claim. It is read next to the Specker obstruction surface of chapter 3516, the stream schedule of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet records monotone bounded window data and the refusal rows that keep that data from becoming a located supremum or ambient completeness principle.

Definition 6227.1 (Specker monotone boundary carrier)**.** *A* $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (S, W, D, R, B, L, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$ *source row naming the nonlocated-supremum obstruction inherited from the bounded monotone Specker stream.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *schedule selecting the finite rational windows inspected by the boundary packet.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and bound ledger read from those windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational data visible at the scheduled windows.*
5. *B is the monotone bounded stream ledger recording displayed order, bound, and window-comparison data before any supremum-facing row is considered.*
6. *L is the refusal row separating the displayed stream data from a located supremum, a selected limit, and any ambient* $\mathsf{Real}^{\uparrow}$ *completeness claim.*
7. *H stores componentwise* hsame *transport for S, W, D, R, B, L.*
8. *C stores displayed* Cont *replay from source and schedule rows through dyadic bounds, regular readback, monotone bounded evidence, and the refusal row.*
9. *P is the* Pkg *provenance over* $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$, $\mathsf{SpeckerSequenceBoundary}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SpeckerMonotoneBoundary}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, W, D, R, B, L, H, C, P, N$. It has no coordinate for a located supremum, selected limit, quotient stream equality, trichotomy, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.SpeckerMonotoneBoundaryUp.SpeckerMonotoneBoundaryTasteGate single carrier alignment`

Theorem 6227.2 (Specker monotone boundary NameCert obligations)**.** *For every accepted* SpeckerMonotoneBoun *carrier*

$$M = (S, W, D, R, B, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SpeckerMonotoneBoundary}^{\uparrow}}$ *surface consists exactly of source admission through S, finite schedule selection through W, dyadic tolerance and bound accounting through D, regular rational readback through R, bounded monotone ledger admission through B, supremum-and-completeness refusal through L, componentwise* hsame *transport through H, displayed* Cont *replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6227.1. The analytic rows are precisely the Specker boundary source, stream schedule, dyadic ledger, regular rational readback, bounded monotone ledger, and refusal row.

The structural rows H, C, P, N transport, replay, source, and name those same coordinates. Since the carrier contains no further analytic field, the listed rows exhaust the local NameCert obligations.

The excluded principles would require extra public coordinates. A located supremum, selected limiting real, quotient stream equality, trichotomy branch, countable-choice selector, or ambient completeness theorem is therefore not part of the accepted packet. □

Theorem 6227.3 (Specker monotone boundary nonescape)**.** *Every consumer of an accepted* $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ *carrier receives only the finite rows $S, W, D, R, B, L, H, C, P, N$. Consequently bounded monotone rational stream data may be passed to* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*-facing readers only through the displayed schedule, tolerance ledger, regular readback, bounded monotone ledger, and refusal row; it exports no located supremum, selected limit, trichotomy, quotient stream equality, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Proof. Use Definition 6227.2 to identify the complete public surface of the accepted packet. Every analytic read passes through S, W, D, R, B, L before the structural rows can name or replay it.

The Specker source row records the obstruction surface, while W, D, R, B expose only finite stream windows, dyadic bounds, regular rational readbacks, and monotone bounded comparison data. These rows do not contain a selector for a supremum or a quotient equality on streams.

Finally read the refusal row L. It is the only completion-facing boundary coordinate, and it records separation from located supremum and ambient Real completeness rather than supplying them. Transport, replay, provenance, and naming preserve that finite route, so none of the excluded principles can escape from the packet. □

Falsifiable prediction: Every $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ consumer that treats bounded monotone rational stream data as a located supremum or as ambient $\mathsf{Real}^{\uparrow}$ completeness without displaying $S, W, D, R, B, L, H, C, P, N$ refutes Definition 6227.1.

Independence witness: Nearest siblings: specker-sequence-boundary, streamname, regseqrat, dyadicratcore, and real. $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ is not bijective to any listed sibling because it

jointly carries the Specker obstruction source, finite stream schedule, dyadic bound ledger, regular rational readback, bounded monotone ledger, refusal row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SpeckerMonotoneBoundary}^{\uparrow}}$.

Constructive story: A $\mathsf{SpeckerMonotoneBoundary}^{\uparrow}$ packet is generated from a SpeckerSequenceBoundary source row, a StreamName schedule, a DyadicRatCore tolerance and bound ledger, a RegSeqRat readback row, a bounded monotone stream ledger, a located-supremum and Real-completeness refusal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SpeckerMonotoneBoundary}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6227.1, Definition 6227.2, and Definition 6227.3 seed the finite boundary between bounded monotone rational stream data and completion-facing claims.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SpeckerMonotoneBoundary}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim located supremum construction, selected limits, trichotomy, quotient stream equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, NameCert obligations, monotone bounded readback, nonescape, classifier transport, displayed replay, provenance, and local naming for BEDC.Derived.SpeckerMonotoneBoundaryUp.

Chapter 6228

A Concrete Naming Certificate for $\mathsf{UniformIntegrability}^{\uparrow}$

Following Dunford and Schwartz, the $\mathsf{UniformIntegrability}^{\uparrow}$ packet records the finite tail-control surface that sits between dominated convergence and Vitali convergence. It reads the measure and integral rows used by chapter 2340, exposes a family-indexed tail cutoff and uniform tail-bound ledger, and then supplies the uniform-integrability row consumed by chapter 5167. The packet is a finite BEDC carrier for tail budgets; it is not measure-theoretic choice, full dominated convergence, ambient integration completeness, quotient almost-everywhere equality, or a selected representative theorem.

Definition 6228.1 (Uniform integrability carrier)**.** *A* $\mathsf{UniformIntegrability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (M, I, F, T, B, H, C, P, N)$$

with the following displayed rows:

1. M *is the* $\mathsf{Measure}^{\uparrow}$ *source row for the displayed measurable-domain and measurable-set requests.*
2. I *is the* $\mathsf{Integral}^{\uparrow}$ *readback row through which tail integrals and budget comparisons are inspected.*
3. F *is the finite family row naming the integrands whose tails are being controlled.*
4. T *is the tail-cutoff row, assigning each displayed tolerance request a finite cutoff window.*
5. B *is the uniform tail-bound row, recording the shared bound read after* M, I, F, T *have been exposed.*
6. H *records componentwise* hsame *transport for* M, I, F, T, B *and the local structural rows.*
7. C *records displayed* Cont *replay from a tail-budget request through the cutoff and bound rows.*
8. P *is the* Pkg *provenance over* $\mathsf{UniformIntegrability}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{DominatedConvergence}^{\uparrow}$, $\mathsf{VitaliConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. N *is the local* $\mathsf{NameCert}_{\mathsf{UniformIntegrability}^{\uparrow}}$ *row.*

The carrier has no coordinate for countable choice of representatives, ambient measure completion, full dominated convergence, global integration completeness, quotient almost-everywhere equality, or a detached Vitali theorem.

Lean4 ✓ `BEDC.Derived.UniformIntegrabilityUp.UniformIntegrabilityTasteGate single carrier alignment`

Theorem 6228.2 (Uniform integrability NameCert obligations)**.** *For every accepted* $\mathsf{UniformIntegrability}^{\uparrow}$ *carrier* $U = (M, I, F, T, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{UniformIntegrability}^{\uparrow}}$ *surface consists exactly of carrier admission for the measure row, integral row, displayed family row, tail-cutoff row, uniform tail-bound row, transport row, replay row, provenance row, and local naming row. The cutoff and bound obligations are read only through* T *and* B *after* M, I, F *have been displayed.*

Proof. Project the carrier of Definition 6228.1. The analytic rows are exactly M, I, F, T, B, and the structural rows are exactly H, C, P, N.

Tail control is the ordered read through T and B. The cutoff row answers the displayed tolerance request, and the uniform bound row records the shared tail estimate after the same measure, integral, and family rows have already been exposed.

The structural rows preserve the same packet. Componentwise transport, replay, provenance, and local naming do not add a selected representative, measure completion, host domination theorem, or integration-completeness coordinate. Hence the listed rows exhaust the local NameCert surface. □

Theorem 6228.3 (Uniform integrability Vitali consumption)**.** *Every* $\mathsf{VitaliConvergence}^{\uparrow}$ *consumer of a* $\mathsf{UniformIntegrability}^{\uparrow}$ *packet reads the uniform-integrability budget through the finite route*

$$M \longmapsto I \longmapsto F \longmapsto T \longmapsto B \longmapsto H, C, P, N.$$

When a dominated-convergence sibling route is present, chapter 2340 may supply the displayed majorant-side evidence, but the Vitali-facing handoff of chapter 5167 still consumes only the tail-cutoff and uniform tail-bound rows listed in Definition 6228.1.

Proof. Start with a Vitali-facing read. By Definition 6228.1, the only family-level analytic rows are M, I, F, T, B.

The budget visible to $\mathsf{VitaliConvergence}^{\uparrow}$ is the tail route $T \longmapsto B$. It records finite cutoffs and a shared tail-bound ledger, not a host theorem about all measurable functions.

If a dominated-convergence handoff is also used, it is sibling evidence attached through the displayed measure, integral, and family rows. It does not replace the uniform-integrability row by full dominated convergence.

The structural rows H, C, P, N transport and replay the same finite route. Since the carrier lacks measure-theoretic choice, ambient integration completeness, quotient almost-everywhere equality, and selected representatives, the Vitali consumer cannot read any of those objects from this packet. □

Falsifiable prediction: Any $\mathsf{VitaliConvergence}^{\uparrow}$ read that claims a uniform-integrability budget while bypassing the displayed Measure, Integral, family, tail-cutoff, uniform tail-bound, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 6228.1.

Independence witness: dominated-convergence, vitali-convergence, measure, integral: $\mathsf{UniformIntegrability}^{\uparrow}$ is not bijective to any listed sibling because it stores the finite tail-cutoff and uniform tail-bound rows that dominated convergence may imply and Vitali convergence consumes, without storing full dominated convergence or the Vitali conclusion.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformIntegrability}^{\uparrow}}$**.**

Constructive story: A $\mathsf{UniformIntegrability}^{\uparrow}$ packet is generated from displayed Measure, Integral, integrand-family, tail-cutoff, uniform tail-bound, componentwise hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformIntegrability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6228.1, Definition 6228.2, and Definition 6228.3 seed the finite tail-budget interface between dominated convergence and Vitali convergence.

Formal verification: FormalStatus(NameCert$_{\text{UniformIntegrability}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim measure-theoretic choice, full dominated convergence, ambient integration completeness, quotient almost-everywhere equality, selected representatives, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-cutoff visibility, uniform tail-bound exactness, dominated-convergence sibling routing, Vitali consumption routing, structural transport, replay, provenance, and nonescape for BEDC.Derived.UniformIntegrabilityUp.

Chapter 6229

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailBall}^{\uparrow}$

Following Bishop and Bridges' regular Cauchy completion route, the $\mathsf{RegularCauchyTailBall}^{\uparrow}$ packet concretizes the finite tail-ball direction of chapter 9238. It sits on the dyadic tolerance surface of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal seal of chapter 38. The packet records a selected tail threshold, a displayed ball radius, a finite observation window, and a Real-facing seal row; it does not assert ambient completeness, quotient streams, selected host limits, countable choice, or host equality of real names.

Definition 6229.1 (Regular Cauchy tail-ball carrier)**.** *A* $\mathsf{RegularCauchyTailBall}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, S, D, Q, W, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational source row.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule opened at the tail threshold.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger comparing the selected tail radius with the requested precision.*
4. *Q is the tail-threshold row naming the finite index after which the regular Cauchy source is read inside the ball.*
5. *W is the tail-ball witness row, recording the displayed center, radius, and finite window selected by Q.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after R, S, D, Q, W have been displayed.*
7. *H records componentwise* hsame *transport for the analytic rows and the local structural rows.*
8. *C records displayed* Cont *replay from a Real-completion request through the threshold, window, dyadic ledger, and seal rows.*
9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailBall}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBall}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected host limits, ambient $\mathsf{Real}^{\uparrow}$ *completeness, countable choice, or host equality of real names.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailBallUp.RegularCauchyTailBallTasteGate single carrier alignment`

Theorem 6229.2 (Regular Cauchy tail-ball NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailBall}^{\uparrow}$ *carrier* $T = (R, S, D, Q, W, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBall}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular rational source, finite stream window, dyadic tolerance ledger, tail threshold, tail-ball witness, Real seal, componentwise transport, displayed replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 6229.1. The visible analytic rows are R, S, D, Q, W, E, and the structural rows are H, C, P, N.

The tail-ball obligation is the ordered read through Q and W. The threshold row fixes the finite tail, and the witness row records the center, radius, and finite window within that tail.

The dyadic and Real-facing rows are downstream. The ledger D supplies the displayed tolerance comparison, and the seal E is consumed only after the regular rational source and stream window have been exposed.

Transport, replay, provenance, and naming preserve the same finite route. Since the carrier lacks quotient streams, selected limits, countable choice, host equality, and ambient completeness, the listed rows exhaust the local surface. □

Theorem 6229.3 (Regular Cauchy tail-ball Real-completion route)**.** *Every Real-completion consumer that reads a* $\mathsf{RegularCauchyTailBall}^{\uparrow}$ *packet must pass through*

$$\mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by $R, S, D, Q, W, E, H, C, P, N$*. Thus the tail-ball route turns the finite regular Cauchy tail idea of chapter 9238 into a concrete NameCert packet available to* $\mathsf{Real}^{\uparrow}$ *consumers without exporting quotient streams or ambient completeness.*

Proof. Begin with the accepted carrier $T = (R, S, D, Q, W, E, H, C, P, N)$. The regular rational source R and finite window schedule S are the only source rows available to the packet.

The threshold and ball data are Q and W. They select a finite tail and a displayed ball witness before any Real-facing seal can be read.

The dyadic ledger D and terminal seal E form the completion-facing handoff. A consumer reads dyadic tolerance data before reaching the $\mathsf{Real}^{\uparrow}$ boundary, so it receives a finite tail-ball certificate rather than a quotient completion.

Finally H, C, P, N transport, replay, source, and name the same route. Because the excluded objects are absent from the carrier, the Real-completion consumer cannot extract ambient completeness, quotient streams, selected host limits, countable choice, or host equality from the packet. □

Falsifiable prediction: A Real-completion consumer that claims a $\mathsf{RegularCauchyTailBall}^{\uparrow}$ read while omitting the displayed RegSeqRat, StreamName, DyadicRatCore, tail-threshold, tail-ball, RealUp seal, hsame, Cont, Pkg, or NameCert rows refutes the carrier of Definition 6229.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyTailBall}^{\uparrow}$ is not bijective to any listed sibling because it jointly stores a regular rational source, finite tail

threshold, tail-ball witness, dyadic tolerance ledger, stream window, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailBall}\uparrow}$.

Constructive story: A $\mathsf{RegularCauchyTailBall}^{\uparrow}$ packet is generated from a RegSeqRat source, StreamName finite window, DyadicRatCore tolerance ledger, tail threshold, tail-ball witness, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBall}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6229.1, Definition 6229.2, and Definition 6229.3 seed the finite tail-ball route from regular Cauchy data to the RealUp seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailBall}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim ambient completeness, quotient streams, selected host limits, countable choice, host equality of real names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, threshold visibility, tail-ball witness exactness, dyadic tolerance readback, Real seal handoff, transport, replay, provenance, and nonescape for BEDC.Derived.RegularCauchyTailBallUp.

Chapter 6230

A Concrete Naming Certificate for $\mathsf{LocatedIntervalSupremum}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ packet records the finite interval-restricted route by which a located supremum read becomes available over a displayed real interval. It sits over the upper-bound and lower-approximation surface of chapter 1427, the retained-branch discipline of chapter 1711, the bounded interval completion boundary of chapter 5836, the dyadic tolerance ledger of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names constructive interval supremum data as finite rows; it does not assert general $\mathsf{Real}^{\uparrow}$ completeness, open-cover compactness, trichotomy, countable choice, quotient real equality, or a selected host supremum.

Definition 6230.1 (Located interval supremum carrier)**.** *A* $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (J, U, A, B, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *J is the located interval source row. It records the displayed endpoints, interval membership requests, and interval-local locatedness data consumed before any supremum-facing read is accepted.*
2. *U is the upper-bound ledger. It records that the candidate Real-facing row bounds the interval-local family through displayed endpoint, membership, and tolerance checks.*
3. *A is the lower-approximation ledger. For each accepted precision request it displays an interval-local approximation row that lies below the bound surface and is read through the same precision route.*
4. *B is the bisection and refinement row. It records retained subintervals, branch decisions, and diameter or tolerance refinements supplied by the located interval bisection surface.*
5. *W is the finite* $\mathsf{StreamName}^{\uparrow}$ *window opened at the indices and tolerances displayed by J, U, A, B.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and dyadic tolerance data.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, consumed only after the interval source, upper-bound ledger, lower-approximation ledger, refinement row, finite windows, and regular readback have been displayed.*

8. *H records componentwise* hsame *transport for $J, U, A, B, W, R, E, C, P, N$.*
9. *C records displayed* Cont *replay for interval-supremum requests through the finite route*

$$J, U, A, B \longmapsto W, R \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalSupremum}^{\uparrow}$, $\mathsf{LocatedSupremum}^{\uparrow}$, $\mathsf{LocatedIntervalBisection}^{\uparrow}$, $\mathsf{BoundedCompleteRealInterval}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSupremum}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $J, U, A, B, W, R, E, H, C, P, N$. It has no coordinate for a selected host supremum, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalSupremumUp.TasteGate.LocatedIntervalSupremumTasteGate single carrier alignment`

Theorem 6230.2 (Located interval supremum NameCert obligations)**.** *For every accepted* LocatedIntervalSupremum *carrier $I = (J, U, A, B, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalSupremum}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *carrier admission for the located interval source, upper-bound ledger, lower-approximation ledger, bisection and refinement row, finite StreamName window, RegSeqRat readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *upper-bound exposure, requiring every supremum-facing bound read to display J and U before E can be consumed;*
3. *lower-approximation cofinality, requiring each accepted precision request to receive an approximation row in A through the same interval and window route;*
4. *bisection-window exactness, requiring the retained subinterval and refinement data in B to determine the windows opened in W;*
5. *readback and seal handoff, requiring each Real-facing read to pass through W, R, E after the interval source, bound, approximation, and bisection rows have been displayed;*
6. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation exports a selected host supremum, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalSupremumUp.LocatedIntervalSupremumTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6230.1. The projection exposes exactly the located interval source J, the upper-bound ledger U, the lower-approximation ledger A, the bisection and refinement row B, the finite observation rows W, R, the terminal seal E, and the structural rows H, C, P, N.

The upper and lower supremum faces are row-local. A bound read must use J and U, while a precision request for lower approximation must use A over the same interval source. Since both faces are tied to B, the retained branch and refinement data determine which finite windows are opened.

The Real-facing handoff is then forced through W, R, E. The StreamName row supplies only the displayed finite windows, the RegSeqRat row supplies the regular rational readback attached to those windows, and the Real seal is downstream of that readback.

Finally the structural rows transport, replay, source, and name the same finite route. Because the carrier contains no row for a selected host supremum, quotient equality, choice schedule, trichotomy decision, open-cover subcover, or ambient completeness theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 6230.3 (Located interval supremum Real seal route)**.** *Every Real-facing supremum read exported by an accepted* $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ *carrier*

$$I = (J, U, A, B, W, R, E, H, C, P, N)$$

factors through the finite route

$$J, U, A, B \longmapsto W, R \longmapsto E, H, C, P, N.$$

The read first displays the located interval source, the upper-bound ledger, the lower-approximation ledger, and the retained bisection or refinement row; it then opens finite StreamName windows, reads them through RegSeqRat, and only then consumes the terminal Real seal. The route exports no selected host supremum, quotient real equality, countable choice, trichotomy, open-cover compactness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the carrier rows named in Definition 6230.1. A Real-facing supremum read has no access to the seal E until the interval source J, upper-bound ledger U, lower-approximation ledger A, and bisection row B have been displayed.

The finite readback route is fixed by those rows. The bisection and approximation ledgers determine the requested finite windows W, and the regular rational readback R is attached to those same windows before the seal can be consumed.

Transport and replay do not create a new supremum source. The rows H and C move and replay only the displayed coordinates, while P and N record provenance and local naming for the same packet.

Thus every exported Real-facing observation factors through $J, U, A, B, W, R, E, H, C, P, N$. Since no coordinate of the packet carries a host supremum selector, quotient equality, choice, trichotomy, open-cover compactness, or ambient completeness, none of those principles is exported by the route. □

Falsifiable prediction: Every $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ Real-facing read factors through one located interval source row, one upper-bound ledger, one lower-approximation ledger, one bisection and refinement row, one StreamName window row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: Nearest siblings: locatedsupremum, located-interval-bisection, bounded-complete-real-interval, dyadicratcore, streamname, regseqrat, and real. $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries interval source data, upper-bound exposure, lower-approximation cofinality, bisection-window exactness, readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalSupremum}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedIntervalSupremum}^{\uparrow}$ packet is generated from a located interval source, an upper-bound ledger, a lower-approximation ledger, a bisection and refinement row, finite StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{LocatedIntervalSupremum}\uparrow}$, seed)

Scope closed: Definition 6230.1, Definition 6230.2, and Definition 6230.3 seed the finite interval-supremum route over located interval, bound, approximation, bisection, readback, Real seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{LocatedIntervalSupremum}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general Real$^{\uparrow}$ completeness, selected host suprema, quotient real equality, countable choice, trichotomy, open-cover compactness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-bound exposure, lower-approximation cofinality, bisection-window exactness, RegSeqRat handoff, Real seal routing, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalSupremumUp.

Chapter 6231

A Concrete Naming Certificate for $\mathsf{EulerMethodFiniteStep}^{\uparrow}$

Following Euler and the constructive numerical ODE tradition, the $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ packet records the finite forward-step certificate by which a displayed ordinary differential equation is read through a bounded mesh, a visible initial state, and finitely many endpoint updates. It sits after the ODE source of chapter 193, the continuous-map row of chapter 130, the dyadic tolerance ledger of chapter 291, the finite window discipline of chapter 126, and the terminal real-name surface of chapter 38. The certificate names a bounded numerical flow segment; it does not assert exact solution existence, global continuation, convergence as the mesh tends to zero, or ambient real completeness.

Definition 6231.1 (Euler finite-step carrier)**.** *An* $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (O, F, T, X, D, H, C, P, N)$$

with the following rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *local time-state row, naming the initial state, finite time window, and state-cell boundary.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field read available on the same displayed window.*
3. *T is the finite step ledger, recording the ordered time nodes, step widths, and endpoint budget.*
4. *X is the finite state-readback ledger, recording each displayed Euler update from the previous state and the vector-field row.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and arithmetic row for step widths, local products, and endpoint error budgets.*
6. *H records componentwise* hsame *transport for the ODE, vector-field, step, state, dyadic, replay, provenance, and naming rows.*
7. *C records displayed* Cont *replay of the finite route*

$$O, F, T, D \longmapsto X \longmapsto H, P, N.$$

8. *P is the* Pkg *provenance over* $\mathsf{EulerMethodFiniteStep}^{\uparrow}$*,* $\mathsf{Ode}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{EulerMethodFiniteStep}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host solution curve, selected limiting path, quotient equality of functions, countable-choice schedule, or an exact ODE solution.

Lean4 ✓ `BEDC.Derived.EulerMethodFiniteStepUp.EulerMethodFiniteStepTasteGate single carrier alignment`

Theorem 6231.2 (Euler finite-step NameCert obligations)**.** *For every accepted* $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ *carrier* $E = (O, F, T, X, D, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{EulerMethodFiniteStep}^{\uparrow}}$ *surface consists of the following five obligation rows:*

1. *carrier admission for the displayed* $\mathsf{Ode}^{\uparrow}$ *source* O *and* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row* F*;*
2. *finite step-budget admission through* T*, including node order, step widths, and endpoint range;*
3. *update-readback exposure through* X*, with each update read from the previous state,* F*, and the matching step-width row;*
4. *dyadic arithmetic and tolerance accounting through* D*, stable under componentwise* hsame *transport* H *and displayed* Cont *replay* C*;*
5. Pkg *provenance and local naming through* P, N*, including the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *consumer faces reached only after the finite step ledger is visible.*

No local obligation may read exact ODE solution existence, convergence beyond the displayed finite step budget, global continuation, quotient path equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The only analytic source rows are the ODE row O, vector-field row F, finite step ledger T, update ledger X, and dyadic arithmetic row D.

The step ledger fixes the finite budget before any state readback is exposed. Each update in X is therefore a displayed read from the previous state, the vector-field row, and the matching dyadic step-width data.

The structural rows H, C, P, N transport, replay, source, and name the same finite packet. Since the carrier has no coordinate for a host solution curve, limiting path, global continuation, choice schedule, or ambient completeness theorem, the five rows listed in the statement exhaust the NameCert surface. □

Falsifiable prediction: An accepted $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ consumer that reads an endpoint update without the displayed ODE row, continuous-map vector-field row, finite step ledger, dyadic arithmetic row, and structural hsame, Cont, Pkg, and NameCert rows refutes the packet surface.

Independence witness: ode, continuousmap, dyadicratcore, streamname, real: none is bijective to $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ because this packet binds the local ODE source, vector-field read, finite step budget, update ledger, dyadic arithmetic, transport, replay, provenance, and local naming rows into one finite method certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{EulerMethodFiniteStep}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EulerMethodFiniteStep}^{\uparrow}$ packet is generated from an Ode local source row, a ContinuousMap vector-field row, a finite step ledger, finite update readbacks, DyadicRatCore arithmetic, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EulerMethodFiniteStep}^{\uparrow}}$, seed)

Scope closed: Definition 6231.1 and Definition 6231.2 seed the finite forward-Euler step route over ODE, continuous-map, dyadic, stream-name, real-readback, transport, replay, provenance, and naming rows.

Formal verification: FormalStatus(NameCert$_{\text{EulerMethodFiniteStep}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim exact ODE solution existence, convergence beyond the displayed finite step budget, global continuation, selected limiting paths, quotient function equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, step-budget visibility, update-readback exposure, dyadic arithmetic stability, transport, replay, provenance, and local naming for BEDC.Derived.EulerMethodFiniteStepUp.

Chapter 6232

A Concrete Naming Certificate for $\mathsf{RegularCauchySaturation}^{\uparrow}$

The $\mathsf{RegularCauchySaturation}^{\uparrow}$ packet realizes the real-completion vision route of chapter 9238 and section 9238.1 as a finite tail-budget closure surface. It binds the dyadic tolerance ledger of chapter 291, the finite window naming discipline of chapter 126, the regular rational readback of chapter 301, the terminal seal of chapter 38, and the equality boundary of chapter 1820. The packet names a saturation route for finite tail windows; it does not assert full real completeness, selected limits, quotient representatives, countable choice, or a global metric completion theorem.

Definition 6232.1 (Regular Cauchy saturation carrier)**.** *A* $\mathsf{RegularCauchySaturation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (W, R, D, E, Q, H, C, P, N)$$

with the following rows:

1. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite tail-window row, naming the inspected regular Cauchy stages and the requested precision window.*
2. R *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same windows.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the requested rational radius, tail comparison budget, and diagonal readback arithmetic.*
4. E *is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after* W, R, D *have been displayed.*
5. Q *is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *boundary row, recording equality only through the displayed tail-window and tolerance route.*
6. H *records componentwise* hsame *transport for the window, readback, tolerance, real-seal, setoid-boundary, replay, provenance, and naming rows.*
7. C *records displayed* Cont *replay through*

$$W, R, D, Q \longmapsto E \longmapsto H, P, N.$$

8. P *is the* Pkg *provenance over* $\mathsf{RegularCauchySaturation}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
9. N *is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySaturation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $W, R, D, E, Q, H, C, P, N$. It has no coordinate for a selected limit, quotient stream representative, countable-choice schedule, ambient completeness theorem, or host equality of real names.

Theorem 6232.2 (Regular Cauchy saturation NameCert obligations)**.** *For every accepted* RegularCauchySaturation *carrier $S = (W, R, D, E, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchySaturation}^{\uparrow}}$ *surface consists exactly of finite tail-window admission through W, regular-rational readback through R, dyadic tolerance accounting through D, terminal real-seal exposure through E, Cauchy-equivalence boundary routing through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N. No local obligation may read full* $\mathsf{Real}^{\uparrow}$ *completeness, a selected limit representative, quotient stream equality, countable choice, metric-completion universality, or host real equality.*

Proof. Project the accepted carrier. The completion-facing data are exactly the tail-window row W, regular-rational readback row R, dyadic tolerance row D, terminal real-seal row E, and setoid-boundary row Q.

The finite tail-window row comes first. It fixes the stages and precision request before the regular rational readback and dyadic comparison can be inspected.

The equality boundary is then only a displayed Cauchy-equivalence route over the same rows. Componentwise transport, continuation replay, provenance, and local naming preserve this finite surface and do not add a selected limit, quotient representative, choice schedule, universal completion object, or ambient completeness theorem. □

Falsifiable prediction: A $\mathsf{RegularCauchySaturation}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal without the displayed $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ rows refutes the packet surface.

Independence witness: streamname, regseqrat, dyadicratcore, real, cauchy-equivalence-setoid: none is bijective to $\mathsf{RegularCauchySaturation}^{\uparrow}$ because this packet jointly carries finite tail windows, regular-rational readback, dyadic tolerance, terminal Real seal, Cauchy-equivalence boundary, transport, replay, provenance, and local naming rows.

The Lean TasteGate follow-up for $\mathsf{RegularCauchySaturation}^{\uparrow}$ must include explicit Nontrivial RegularCauchySatura and FieldFaithful $\mathsf{RegularCauchySaturation}^{\uparrow}$ instances, with field faithfulness preserving the order $W, R, D, E, Q, H, C, P, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchySaturation}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RegularCauchySaturation}^{\uparrow}$ packet is generated from StreamName finite tail windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal row, a CauchyEquivalenceSetoid boundary row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchySaturation}^{\uparrow}}$, seed)

Scope closed: Definition 6232.1 and Definition 6232.2 seed the finite tail-budget route from regular Cauchy windows to the terminal Real seal and Cauchy-equivalence boundary.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchySaturation}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim full real completeness, selected limit representatives, quotient stream equality, countable choice, metric-completion universality, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked tail-window admission, regular-rational readback, dyadic tolerance accounting, terminal Real seal routing, Cauchy-equivalence boundary exposure, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchySaturationUp.

Chapter 6233

A Concrete Naming Certificate for $\mathsf{MarkovReal}^{\uparrow}$

Following Markov and Kushner on constructive recursive real analysis, the $\mathsf{MarkovReal}^{\uparrow}$ packet records a Markov-school effective real inspection boundary as finite BEDC data. It is read beside the effective source rows of chapter 336, the Markov-sensitive refusal row of chapter 1666, the constructive trichotomy boundary of chapter 6149, the regular-rational readback of chapter 301, the stream-window discipline of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet exposes the finite rows through which a Markov-school inspection may be replayed, while keeping Markov's principle, real trichotomy, stream quotients, selected representatives, countable choice, and ambient completeness outside the BEDC kernel.

Definition 6233.1 (MarkovReal carrier)**.** *A* $\mathsf{MarkovReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (C, B, A, S, R, D, E, H, K, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{ComputableReal}^{\uparrow}$ *source row naming the effective real presentation whose finite observations are being inspected.*
2. *B is the* $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$ *inspection/refusal row that bounds any use of negative evidence.*
3. *A is the apartness-or-trichotomy-boundary row supplied by* $\mathsf{ConstructiveRealTrichotomyBoundary}^{\uparrow}$ *for displayed comparison evidence and refusal.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite window through which the inspection reads the effective source.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposed by the window.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger naming the endpoint and margin rows used by the readback.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the effective source, boundary, window, readback, and tolerance rows have been displayed.*
8. *H records componentwise* hsame *transport for the effective source, Markov boundary, trichotomy boundary, stream window, regular readback, dyadic ledger, real seal, replay, provenance, and local naming row.*

9. *K records displayed* Cont *replay from a Markov-school inspection request through C, B, A, S, R, D, E.*
10. *P records* Pkg *provenance over* $\mathsf{MarkovReal}^{\uparrow}$, $\mathsf{ComputableReal}^{\uparrow}$, $\mathsf{MarkovPrincipleBoundary}^{\uparrow}$, ConstructiveRealTrich $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{MarkovReal}^{\uparrow}}$ *row.*

The carrier has no coordinate for Markov's principle as an imported axiom, constructive real trichotomy, quotient stream equality, selected representatives, countable choice, selected limits, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.MarkovRealUp.MarkovRealTasteGate single carrier alignment`

Theorem 6233.2 (MarkovReal NameCert obligations)**.** *For every accepted* $\mathsf{MarkovReal}^{\uparrow}$ *carrier*

$$M = (C, B, A, S, R, D, E, H, K, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{MarkovReal}^{\uparrow}}$ *surface consists exactly of the following rows:*

1. *carrier admission for the displayed effective source, Markov boundary, trichotomy boundary, finite window, readback, tolerance ledger, real seal, transport, replay, provenance, and local name;*
2. *effective-window exposure, requiring every public observation of C to pass through the finite* $\mathsf{StreamName}^{\uparrow}$ *row S;*
3. *Markov-boundary non-escape, requiring every use of negative evidence to cite B rather than a global Markov rule;*
4. *apartness/refusal compatibility, requiring every comparison-facing inspection to pass through the displayed row A;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *handoff, requiring each inspected rational value to be exposed through R before any real-facing conclusion is public;*
6. $\mathsf{Real}^{\uparrow}$ *seal non-escape, requiring the seal E to consume only the displayed readback and tolerance rows;*
7. *componentwise transport through H;*
8. *displayed continuation replay through K;*
9. *package provenance through P;*
10. *local naming through N.*

No Markov-real naming obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6233.1. The tuple displays the effective source, the two boundary rows, the finite window, the regular-rational readback, the dyadic tolerance ledger, and the real seal before it lists transport, replay, provenance, and local naming.

The effective-window and readback obligations follow from the positions of S and R. A consumer cannot inspect the effective source C without first naming a finite stream window S, and the rational data read from that window is public only through the regular-sequence row R and the dyadic ledger D.

The Markov-boundary and apartness/refusal obligations are the boundary rows B and A. They record the finite evidence or refusal that an inspection may cite, but they do not add Markov's principle or trichotomy as ambient rules.

The remaining rows are componentwise. Transport is exactly H, replay is exactly K, provenance is exactly P, and the local public name is exactly N. Since the carrier contains no additional coordinate, the listed rows exhaust the local $\mathsf{NameCert}_{\mathsf{MarkovReal}^\uparrow}$ surface. □

Theorem 6233.3 (MarkovReal RegSeqRat handoff)**.** *Every* $\mathsf{Real}^\uparrow$*-facing* $\mathsf{MarkovReal}^\uparrow$ *read factors through the displayed route*

$$\mathsf{ComputableReal}^\uparrow \to \mathsf{MarkovPrincipleBoundary}^\uparrow \to \mathsf{StreamName}^\uparrow \to \mathsf{RegSeqRat}^\uparrow \to \mathsf{DyadicRatCore}^\uparrow \to \mathsf{Real}^\uparrow$$

before any inspection conclusion is public. In particular, the $\mathsf{MarkovReal}^\uparrow$ *packet exposes no direct path from an effective real source to a real-facing Markov conclusion that bypasses the boundary, finite window, regular-rational readback, dyadic tolerance ledger, and real seal.*

Proof. Let $M = (C, B, A, S, R, D, E, H, K, P, N)$ be an accepted carrier. The source row C names the effective real presentation, while B bounds the Markov-sensitive inspection that may be performed on it.

Any actual observation must next pass through the finite window S. The read values are not public as a completed real object; they are exposed through the regular-rational row R and the dyadic tolerance ledger D.

Only after those rows have been displayed may the real seal E be cited. Transport H and replay K preserve the same route componentwise, while P and N repackage it as the public NameCert surface. Thus every Real-facing read factors through the stated chain before it can support an inspection conclusion. □

Closure status for $\mathsf{NameCert}_{\mathsf{MarkovReal}^\uparrow}$.

Constructive story: A $\mathsf{MarkovReal}^\uparrow$ packet is generated from a ComputableReal source row, a MarkovPrincipleBoundary inspection/refusal row, an apartness-or-trichotomy-boundary row, a finite StreamName window, a RegSeqRat readback, a DyadicRatCore tolerance ledger, a Real seal, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MarkovReal}^\uparrow}$, seed)

Scope closed: Definition 6233.1, Definition 6233.2, and Definition 6233.3 seed the finite Markov-school effective real inspection boundary over L10 Real and Completion rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MarkovReal}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim imported Markov's principle, constructive real trichotomy, quotient stream equality, selected representatives, countable choice, ambient $\mathsf{Real}^\uparrow$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, effective-window exposure, Markov-boundary non-escape, apartness/refusal compatibility, RegSeqRat handoff, Real-seal non-escape, componentwise transport, replay, provenance, and local naming for the same finite packet.

Chapter 6234

A Concrete Naming Certificate for $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$

Following Arzela-Ascoli and Bishop–Bridges, the $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$ packet records the finite BEDC route that turns a compact metric source and an equicontinuous family of continuous maps into one displayed uniform budget for the whole family. It sits over the compact finite-net surface of chapter 129, the map carrier of chapter 130, the family surface of chapter 413, the precision consumer of chapter 369, the finite-window row of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet is a finite-net modulus route; it does not claim open-cover compactness, a selected global modulus outside the carrier, or a choice principle over the family.

Definition 6234.1 (Compact equicontinuous family modulus carrier)**.** *A* $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathcal{E} = (X, F, M, U, W, R, S, H, C, P, N)$$

with the following visible rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including its finite-net schedules and tolerance coverage data.*
2. *F is the* $\mathsf{EquicontinuousFamily}^{\uparrow}$ *row, displaying the common pointwise budget for the family over the compact source.*
3. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family-member row used when a particular displayed map is inspected.*
4. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row that receives the compact-family budget.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row through which the source and target observations are scheduled.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the rational precision histories attached to the windows.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the finite-net, family, map, window, and readback rows have been displayed.*
8. *H records componentwise* hsame *transport for $X, F, M, U, W, R, S, C, P, N$.*
9. *C records displayed* Cont *replay of the same finite-net route for downstream consumers.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{EquicontinuousFamily}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. There is no coordinate for open-cover compactness, host function-space compactness, selected subnet, quotient equality of maps, or a modulus chosen outside the finite packet.

Lean4 (definition) `BEDC.Derived.CompactEquicontinuousFamilyModulusUp.TasteGate.CompactEquicontinuousFamily`
Lean4 ✓ `BEDC.Derived.CompactEquicontinuousFamilyModulusUp.TasteGate.CompactEquicontinuousFamilyModulus`
`single carrier alignment`

Theorem 6234.2 (Compact equicontinuous family modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$ *carrier*

$$\mathcal{E} = (X, F, M, U, W, R, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}}$ *surface consists exactly of compact finite-net admission through X, equicontinuous-family admission through F, displayed family-member inspection through M, uniform-budget output through U, finite-window and regular-rational readback through W, R, terminal Real sealing through S, componentwise transport through H, replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6234.1. Its visible coordinates list exactly the compact source, equicontinuity family, inspected continuous map, uniform-modulus output, finite windows, regular rational readback, Real seal, transport, replay, provenance, and local naming rows.

The compact and family clauses are read through X and F. The finite-net schedule belongs to the compact metric row, while the common family budget belongs to the equicontinuity row; neither clause exposes an open cover, a selected subnet, or a host function-space object.

The output and seal clauses are then read through M, U, W, R, S and preserved by H, C, P, N. Since no other coordinate is present, the listed rows exhaust the local NameCert surface. □

Theorem 6234.3 (Compact equicontinuous family uniform budget route)**.** *Every uniform-budget consumer of an accepted* $\mathsf{CompactEquicontinuousFamilyModulus}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{EquicontinuousFamily}^{\uparrow} \longmapsto \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{UniformModul}$$

Concretely, the compact finite-net row X and the family row F are public before any member row M or output row U is read, and the terminal seal S is public only after the window and readback rows W, R have been replayed.

Proof. Start with the carrier row $\mathcal{E} = (X, F, M, U, W, R, S, H, C, P, N)$. The source-side information is exactly X, and the common family budget is exactly F.

A consumer inspecting a displayed family member then reads M over the same compact source and family budget. The finite-window row W schedules the visible source and target observations, while R supplies the regular rational readback for the precision history used by the output row.

Finally U packages the uniform budget and S seals the Real-facing read. The rows H, C, P, N transport, replay, source, and name the same route, so no open-cover compactness theorem, selected global modulus, quotient map equality, or choice-selected family element can replace the displayed finite-net route. □

Theorem 6234.4 (Compact equicontinuous family finite-net nonescape)**.** *The* CompactEquicontinuousFamilyModul *packet exports no modulus data outside the compact finite-net, equicontinuity-family, continuous-map, finite-window, regular-rational, and Real-seal rows*

$$X, F, M, W, R, S.$$

Every downstream UniformModulus$^{\uparrow}$ *read is therefore a finite BEDC read over those rows together with* H, C, P, N*, not a host Arzela–Ascoli compactness argument, an open-cover proof, a selected subnet, or an external modulus selector.*

Proof. Use Definition 6234.2 to identify the carrier surface. The only rows that can supply the budget are X, F, M, W, R, S together with the structural rows H, C, P, N.

Apply Definition 6234.3. It orders the compact source, family budget, inspected map, finite windows, readback, output, and seal before any downstream consumer can read a uniform modulus.

Because the carrier of Definition 6234.1 contains no host compactness, selected subnet, quotient equality, or external selector coordinate, the finite-net route is nonescaping. □

Falsifiable prediction: Every accepted CompactEquicontinuousFamilyModulus$^{\uparrow}$ budget read factors through one CompactMetric$^{\uparrow}$ source row, one EquicontinuousFamily$^{\uparrow}$ family row, one ContinuousMap$^{\uparrow}$ member row, one StreamName$^{\uparrow}$ finite-window row, one RegSeqRat$^{\uparrow}$ readback row, one UniformModulus$^{\uparrow}$ output row, one terminal Real$^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: compactmetric, continuousmap, equicontinuityfamily, uniformmodulus, streamname, regseqrat, real: CompactEquicontinuousFamilyModulus$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact finite-net data, equicontinuity-family data, inspected map data, precision windows, rational readback, uniform output, Real sealing, transport, replay, provenance, and local NameCert rows as one packet.

Closure status for NameCert$_{\text{CompactEquicontinuousFamilyModulus}^{\uparrow}}$.

Constructive story: A CompactEquicontinuousFamilyModulus$^{\uparrow}$ packet is generated from a CompactMetricUp finite-net source, an EquicontinuousFamilyUp common-budget row, a ContinuousMapUp inspected-member row, StreamNameUp windows, RegSeqRatUp readback, a UniformModulusUp output row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{CompactEquicontinuousFamilyModulus}^{\uparrow}}$, seed)

Scope closed: Definition 6234.1, Definition 6234.2, and Definition 6234.3 seed the compact-family uniform-budget route over finite-net, family, map, window, readback, output, and seal rows.

Formal verification: FormalStatus(NameCert$_{\text{CompactEquicontinuousFamilyModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, host function-space compactness, selected subnets, quotient map equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net coverage, family-budget stability, member inspection, window and readback exactness, uniform-output exactness, Real-seal nonescape, transport, replay, provenance, and local naming.

Chapter 6235

A Concrete Naming Certificate for $\mathsf{CWComplex}^{\uparrow}$

Following Whitehead and Hatcher, the $\mathsf{CWComplex}^{\uparrow}$ packet records the finite cell-attachment surface used by algebraic-topology and homology consumers before any quotient-space, weak-topology, or homotopy-extension theorem is imported. It sits after the open-neighbourhood interface of chapter 88, the finite face inventory of chapter 238, the boundary and cycle consumer of chapter 98, and the finite subcover discipline of chapter 60. The chapter names cellular strata, attaching maps, boundary incidence, and homology handoff as displayed BHist data rather than as an ambient topological construction.

Definition 6235.1 (CW-complex carrier)**.** *A* $\mathsf{CWComplex}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (T, K, E, A, B, C, H, R, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, naming the open-neighbourhood interface on which cells are inspected.*
2. *K is the* $\mathsf{SimplicialComplex}^{\uparrow}$ *comparison row for finite face-style reads and cellular approximation consumers.*
3. *E is the finite cell inventory, with each displayed cell carrying a dimension, a name, and an incidence window.*
4. *A is the attaching ledger, recording the finite boundary route by which an n-cell reads the lower-dimensional skeleton.*
5. *B is the cellular boundary ledger, listing incidence coefficients only through displayed attaching data.*
6. *C is the* $\mathsf{Compact}^{\uparrow}$ *inspection row for finite subcomplex and compact-closure reads.*
7. *H records componentwise* hsame *transport for topology, simplicial comparison, cell inventory, attaching, boundary, compact inspection, replay, provenance, and local naming rows.*
8. *R records displayed* Cont *replay from T, K, E, A, B, C, H to cellular and homology-facing consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{CWComplex}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{Compact}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CWComplex}^\uparrow}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host quotient topology, arbitrary weak topology, selected characteristic maps, infinite cellular chains, global homotopy extension, or completed homology computation.

Lean4 ✓ `BEDC.Derived.CWComplexUp.CWComplexTasteGate single carrier alignment`

Theorem 6235.2 (CW-complex NameCert obligations)**.** *For an accepted* $\mathsf{CWComplex}^\uparrow$ *carrier* $X = (T, K, E, A, B, C, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CWComplex}^\uparrow}$ *surface consists exactly of the finite carrier admission, cell-inventory coverage, attaching-ledger replay, cellular-boundary exposure, compact finite-subcomplex inspection, componentwise transport, displayed replay, provenance, and local naming obligations recorded by* $T, K, E, A, B, C, H, R, P, N$*. A consumer cannot replace these rows by quotient-space semantics, weak-topology closure, selected characteristic maps, arbitrary infinite cellular chains, host homotopy extension, or an ambient homology computation.*

Proof. Project the carrier of Definition 6235.1. The topology and comparison rows are T, K, the cellular data are E, A, B, the compact inspection row is C, and the structural rows are H, R, P, N.

Cell-inventory coverage is read from E. Each cell has a displayed dimension and incidence window before an attaching route is considered, so the packet never starts from an unbounded class of cells.

The attaching and boundary obligations are then read from A and B. The boundary ledger can inspect only the lower-skeleton data named by the attaching ledger, while compact finite-subcomplex reads pass through C.

Finally H and R transport and replay the same rows, and P, N source and name them. Since no coordinate stores quotient topology, weak topology, selected characteristic maps, infinite cellular chains, homotopy extension, or completed homology, those endpoints are outside the certificate. □

Theorem 6235.3 (CW-complex cellular-boundary handoff)**.** *Every* $\mathsf{Homology}^\uparrow$ *consumer of an accepted* $\mathsf{CWComplex}^\uparrow$ *packet reaches cellular boundary data only through the finite route*

$$\mathsf{Topology}^\uparrow,\ \mathsf{SimplicialComplex}^\uparrow,\ E, A, B, C \longmapsto \mathsf{Homology}^\uparrow.$$

The handoff exposes the topological source, simplicial comparison surface, finite cell inventory, attaching ledger, cellular boundary ledger, compact finite-subcomplex inspection, componentwise transport, replay, provenance, and local naming rows. It does not export quotient-space equality, a weak-topology oracle, selected characteristic maps, unrestricted cellular chains, host homotopy extension, or computed homology groups.

Proof. Begin with the NameCert obligations of Definition 6235.2. They force the consumer to read the topology row T, the simplicial comparison row K, and the finite cell inventory E before boundary data is available.

Next use the attaching ledger A. A cellular boundary read through B is valid only after the lower-skeleton route named by A is displayed, and compact finite-subcomplex inspection remains the separate row C.

The structural rows H, R, P, N repack this route for homology-facing consumers. Therefore the $\mathsf{Homology}^\uparrow$ handoff is exactly the displayed finite cellular boundary route and cannot import quotient spaces, weak topology, selected characteristic maps, unrestricted chain complexes, host homotopy extension, or completed homology. □

Falsifiable prediction: A $\mathsf{Homology}^{\uparrow}$ consumer using $\mathsf{CWComplex}^{\uparrow}$ must display the topology source, simplicial comparison row, finite cell inventory, attaching ledger, cellular boundary ledger, compact finite-subcomplex inspection, componentwise hsame transport, Cont replay, Pkg provenance, and local naming rows.

Independence witness: topology, simplicialcomplex, homology, and compact are non-bijective siblings: none alone carries the complete cellular inventory, attaching ledger, boundary handoff, compact inspection, transport, replay, provenance, and local NameCert surface exposed by $\mathsf{CWComplex}^{\uparrow}$.

Closure status for $\mathsf{NameCert}_{\mathsf{CWComplex}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CWComplex}^{\uparrow}$ packet is generated from a Topology row, SimplicialComplex comparison row, finite cell inventory, attaching ledger, cellular boundary ledger, Compact finite-subcomplex inspection, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CWComplex}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6235.1, Definition 6235.2, Definition 6235.3, the falsifiable prediction, and the independence witness seed the finite CW-complex carrier and its cellular boundary handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CWComplex}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient-space semantics, weak topology, selected characteristic maps, infinite cellular chain complexes, homotopy extension, or completed homology computations.

Upgrade path: Obligation closure requires checked carrier admission, cell-inventory coverage, attaching-ledger replay, cellular-boundary exposure, compact finite-subcomplex inspection, transport preservation, provenance exhaustion, and local naming for BEDC.Derived.CWComplexUp.

Chapter 6236

A Concrete Naming Certificate for $\mathsf{HermitePolynomial}^{\uparrow}$

Following Hermite and the standard orthogonal-polynomial tradition, the $\mathsf{HermitePolynomial}^{\uparrow}$ packet names the finite Gaussian-weight recurrence surface for the displayed family H_n without importing an ambient L^2 space, quotient polynomial ring, or completed measure theory. It sits after the polynomial carrier of chapter 50, the orthogonal-polynomial surface of chapter 2264, the terminal real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the sibling recurrence packets of chapter 3480 and chapter 6128.

Definition 6236.1 (Hermite polynomial carrier)**.** *A* $\mathsf{HermitePolynomial}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (d, H_0, H_1, R, O, W, Q, E, T, C, P, N)$$

with the following displayed rows:

1. *d is the finite degree-window row.*
2. *H_0 and H_1 are the seed* $\mathsf{Poly}^{\uparrow}$ *rows for the inspected Hermite family.*
3. *R is the finite recurrence ledger for $H_{n+1} = 2xH_n - 2nH_{n-1}$, including the displayed coefficient and polynomial-equality rows for each accepted step.*
4. *O is the* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *handoff row for the Gaussian-weight moment ledger and finite orthogonality reads.*
5. *W is the finite Gaussian-weight window row, recording the bounded moment and evaluation requests through which no completed measure space is exposed.*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for inspected coefficients, moment approximants, and evaluations.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for recurrence coefficients, Gaussian-window reads, and evaluation results.*
8. *T records componentwise* hsame *transport for the displayed rows.*
9. *C records displayed* Cont *replay of recurrence, Gaussian-window, readback, and orthogonality requests.*
10. *P is the* Pkg *provenance over* $\mathsf{HermitePolynomial}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{OrthogonalPolynomial}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{HermitePolynomial}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host Gaussian measure, completed L^2 space, quotient polynomial equality, selected infinite basis, or analytic spectral theorem.

Lean4 ✓ `BEDC.Derived.HermitePolynomialUp.HermitePolynomialTasteGate single carrier alignment`

Theorem 6236.2 (Hermite polynomial NameCert obligations)**.** *For every accepted* $\mathsf{HermitePolynomial}^{\uparrow}$ *carrier $G = (d, H_0, H_1, R, O, W, Q, E, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HermitePolynomial}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *the* $\mathsf{Poly}^{\uparrow}$ *source row displaying the degree window, seed polynomials, recurrence coefficients, and polynomial-equality checks;*
2. *the* $\mathsf{Real}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row sealing every inspected coefficient, finite Gaussian-window moment, and evaluation request;*
3. *the* $\mathsf{OrthogonalPolynomial}^{\uparrow}$ *handoff row recording orthogonality only through displayed recurrence and finite Gaussian-weight window data;*
4. *the recurrence-stability row, requiring* hsame *transport and displayed* Cont *replay to preserve the same Hermite seed and step ledgers;*
5. *the* Pkg *and* NameCert *row naming the packet while exporting no host measure, quotient equality, infinite basis selector, or completed function space.*

No Hermite-polynomial obligation is stored outside these finite rows.

Proof. Project the carrier of Definition 6236.1. The rows d, H_0, H_1, R display the only polynomial source, so every recurrence and coefficient request is a finite $\mathsf{Poly}^{\uparrow}$ ledger read.

The Gaussian-weight side is exposed by O, W, Q, E. The orthogonal-polynomial handoff O can be read only together with the finite window row W, the regular rational readback row Q, and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Finally T, C, P, N preserve, replay, source, and name the same finite packet. Since no row stores a host Gaussian measure, completed L^2 space, quotient polynomial equality, selected infinite basis, or analytic spectral theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6236.3 (Hermite polynomial recurrence stability)**.** *Every accepted* $\mathsf{HermitePolynomial}^{\uparrow}$ *recurrence read factors through*

$$d, H_0, H_1 \longmapsto R \longmapsto O, W, Q, E \longmapsto T, C, P, N.$$

Under componentwise hsame *transport and displayed* Cont *replay, the same finite seed rows, recurrence coefficients, Gaussian-weight windows, readback rows, Real seals, provenance, and local naming remain visible. The packet therefore cannot infer a host Gaussian measure, completed L^2 equality, selected infinite Hermite basis, or analytic spectral theorem from recurrence data alone.*

Proof. Start with the recurrence ledger. The degree row and the two seed rows determine the inspected recurrence window, and R displays each accepted step of $H_{n+1} = 2xH_n - 2nH_{n-1}$ as a polynomial-row equality.

Next read the orthogonality and value-facing rows. The handoff through O is tied to the finite Gaussian-weight windows W, the regular readback Q, and the terminal Real seal E, so orthogonality remains a displayed finite ledger rather than a host measure-space claim.

The transport and replay rows T, C preserve exactly those coordinates, while P, N record provenance and local naming. No carrier coordinate supplies the excluded host or infinite data, so recurrence stability is confined to the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{HermitePolynomial}^{\uparrow}$ read factors through PolynomialUp seeds and recurrence rows, RealUp and RegSeqRat readback, an OrthogonalPolynomialUp handoff, finite Gaussian-weight windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: polynomial, real, regseqrat, dyadicratcore, orthogonal-polynomial, chebyshev-polynomial, legendre-polynomial: $\mathsf{HermitePolynomial}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries Hermite seeds, recurrence coefficients, Gaussian-weight windows, orthogonality handoff, readback, Real sealing, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HermitePolynomial}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HermitePolynomial}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6236.1, Definition 6236.2, and Definition 6236.3 seed the finite Hermite recurrence and Gaussian-weight orthogonality certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HermitePolynomial}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host Gaussian measures, completed function spaces, quotient polynomial equality, selected infinite Hermite bases, analytic spectral theorems, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, recurrence exactness, finite Gaussian-window readback, OrthogonalPolynomial handoff, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HermitePolynomialU

Constructive story: A $\mathsf{HermitePolynomial}^{\uparrow}$ packet is generated from a degree window, two seed PolynomialUp rows, a finite Hermite recurrence ledger, an OrthogonalPolynomialUp handoff, finite Gaussian-weight windows, RegSeqRat readback, RealUp seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6237

A Concrete Naming Certificate for $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$

The $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ packet realizes the boundary-fixed disk-transport reading of chapter 9463 as finite Mobius-invariant rows. It sits after the disk carrier of chapter 377, the metric route of chapter 3474, the Poincare disk metric packet of chapter 6209, the hyperbolic cross-ratio packet of chapter 3948, and the boundary-fixed transport route of chapter 3500. The packet names invariant preservation for displayed disk and boundary rows; it is not a completed hyperbolic plane, a quotient ideal boundary, a global automorphism group, or a physical observer law.

Definition 6237.1 (Hyperbolic Mobius invariant carrier)**.** *A* $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (U, M, D, X, B, A, R, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UnitDisk}^{\uparrow}$ *row for the source and target disk observations.*
2. *M is the disk-map row recording the Mobius-style transport on displayed disk coordinates.*
3. *D is the* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *or* $\mathsf{HyperbolicMetric}^{\uparrow}$ *metric-read row consumed after the disk observations and transport have been exposed.*
4. *X is the* $\mathsf{HyperbolicCrossRatio}^{\uparrow}$ *comparison row for boundary or near-boundary quadruple reads.*
5. *B is the boundary-fixed row, recording which visible boundary source is preserved by the transport.*
6. *A is the invariant audit row stating that the metric or cross-ratio read is preserved only through the displayed disk-map, boundary, and comparison rows.*
7. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row attached to metric and cross-ratio readback.*
8. *H records componentwise* hsame *transport for the displayed rows.*
9. *C records displayed* Cont *replay of disk-coordinate, map, metric, cross-ratio, boundary, audit, Real-seal, provenance, and local-name reads.*
10. *P is the* Pkg *provenance over* $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{HyperbolicCrossRatio}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{HyperbolicMobiusInvariant}^{\uparrow}}$ *row.*

The carrier has no coordinate for a completed disk, quotient boundary, host automorphism group, selected geodesic, global compactification, or physical spacetime invariant.

Theorem 6237.2 (Hyperbolic Mobius invariant NameCert obligations)**.** *For every accepted* HyperbolicMobiusInvari *carrier* $I = (U, M, D, X, B, A, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicMobiusInvariant}^{\uparrow}}$ *surface consists exactly of disk-coordinate admission, displayed Mobius-transport admission, hyperbolic metric readback, cross-ratio comparison, boundary-fixed preservation, invariant-audit exactness, Real sealing, componentwise transport, displayed replay, provenance, and local naming. The invariant row* A *may be consumed only after* U, M, D, X, B *have been displayed, and it exports no completed hyperbolic plane, quotient boundary, host automorphism group, global compactification, or physical observer law.*

Proof. Project the carrier of Definition 6237.1. The disk data U and transport row M are the only source rows from which the invariant route can begin.

The metric and cross-ratio reads are downstream. Rows D and X expose the finite comparison surfaces, while B records the boundary-fixed source preserved by the displayed transport.

The audit row A can therefore certify only preservation along that visible route. Rows R, H, C, P, N seal, transport, replay, source, and name the same packet, and no excluded host geometry or physical law is present in the carrier. □

Falsifiable prediction: Every accepted $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ consumer must display UnitDisk coordinates, Mobius-style disk transport, HyperbolicMetricUp or PoincareDiskMetricUp metric reads, HyperbolicCrossRatioUp comparison, boundary-fixed preservation, invariant audit, RealUp sealing, hsame transport, Cont replay, Pkg provenance, and local NameCert rows before reading invariant preservation.

Independence witness: unitdisk, hyperbolicmetric, poincare-disk-metric, hyperbolic-cross-ratio, poincare-boundary-fixed-transport: no listed sibling is bijective to $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ because this packet jointly carries disk coordinates, Mobius-style transport, metric readback, cross-ratio comparison, boundary-fixed preservation, invariant audit, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicMobiusInvariant}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicMobiusInvariant}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6237.1 and Definition 6237.2 seed the finite Mobius-invariant disk-transport route over UnitDisk, hyperbolic metric, cross-ratio, boundary-fixed, Real-seal, transport, replay, provenance, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicMobiusInvariant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed hyperbolic planes, quotient ideal boundaries, global automorphism groups, selected geodesics, physical observer laws, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-coordinate admission, Mobius-transport admission, metric and cross-ratio exactness, boundary-fixed preservation, invariant audit nonescape, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.HyperbolicMobiusInvariantUp.

Constructive story: A $\mathsf{HyperbolicMobiusInvariant}^{\uparrow}$ packet is generated from UnitDisk coordinates, displayed Mobius-style disk transport, PoincareDiskMetric or HyperbolicMetric reads, HyperbolicCrossRatio comparison, boundary-fixed preservation, an invariant audit row, RealUp sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6238

A Concrete Naming Certificate for $\mathsf{CauchyMeshStability}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{CauchyMeshStability}^{\uparrow}$ packet records the finite mesh-refinement stability surface used before a regular rational name is sealed as a real. It sits over the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet names mesh refinement as displayed BEDC data; it is not a quotient of Cauchy streams, a selected completion point, countable choice for refinements, or an ambient completeness theorem.

Definition 6238.1 (Cauchy mesh stability carrier)**.** *A* $\mathsf{CauchyMeshStability}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (M, W, D, R, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the mesh-source row, naming the finite refinement schedule whose entries are inspected by the packet.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, recording the bounded stream observations read at the displayed mesh levels.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, supplying the dyadic comparison budget used to test whether two displayed mesh reads are stable at the requested precision.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the accepted window and tolerance data as a regular rational name.*
5. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row, available only after M, W, D, R have been displayed.*
6. *H records componentwise* hsame *transport for the mesh source, finite windows, dyadic tolerances, regular rational readback, real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a mesh-refinement request through M, W, D, R and then to E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyMeshStability}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyMeshStability}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $M, W, D, R, E, H, C, P, N$. There is no coordinate for an infinite refinement selector, quotient stream equality, selected real value, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.CauchyMeshStabilityUp.CauchyMeshStabilityTasteGate single carrier alignment`

Theorem 6238.2 (Cauchy mesh stability NameCert obligations)**.** *For every accepted* $\mathsf{CauchyMeshStability}^{\uparrow}$ *carrier* $S = (M, W, D, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyMeshStability}^{\uparrow}}$ *surface consists exactly of carrier admission for the mesh source, finite stream window, dyadic tolerance row, regular rational readback row, real seal handoff, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6238.1. The analytic rows are exactly M, W, D, R, E, and the structural rows are exactly H, C, P, N.

The mesh source M supplies the refinement schedule before any stream entry is consumed. The finite window W then exposes only the bounded observations attached to that schedule, while the dyadic row D supplies the precision budget used to compare those observations.

The regular rational readback R is downstream of M, W, D, and the real seal E is downstream of R. Rows H, C, P, N transport, replay, source, and name only that finite route, so the listed rows exhaust the local NameCert obligations. □

Theorem 6238.3 (Cauchy mesh stability Real-completion handoff)**.** *Every Real or Completion consumer of an accepted* $\mathsf{CauchyMeshStability}^{\uparrow}$ *carrier factors through the finite route*

$$M, W, D \longmapsto R \longmapsto E, H, C, P, N.$$

The consumer first reads the mesh source, then the finite stream window and dyadic tolerance budget, then the regular rational readback, and only then receives the real-name seal. The route exports mesh-stable readback data, not quotient Cauchy-stream equality, a selected completion point, countable choice for refinements, or ambient completeness of $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with a Real or Completion consumer of S. The carrier exposes the mesh source only through M, and every bounded observation is read through the finite window W attached to that same source.

The dyadic tolerance row D is the only precision coordinate available for stability comparisons. Thus a consumer cannot replace the displayed mesh comparison by an arbitrary refinement theorem or an external selector.

The regular rational row R repacks the accepted mesh-window and tolerance data before the real seal E can be read. The structural rows H, C, P, N preserve that route without adding quotient equality, selected limits, countable choice, or ambient completeness, so the handoff factors through the displayed packet. □

Falsifiable prediction: A $\mathsf{CauchyMeshStability}^{\uparrow}$ consumer that reads a $\mathsf{Real}^{\uparrow}$ seal without first displaying mesh-source data, finite StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, transport, replay, provenance, and local naming is not reading the carrier of Definition 6238.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: none alone is bijective with $\mathsf{CauchyMeshStability}^{\uparrow}$ because the packet jointly carries mesh-source data, finite stream windows, dyadic tolerance comparisons, regular rational readback, Real seal handoff, componentwise transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyMeshStability}^{\uparrow}}$ **.**

Constructive story: A $\mathsf{CauchyMeshStability}^{\uparrow}$ packet is generated from a mesh-source row, StreamName finite windows, DyadicRatCore tolerance comparisons, RegSeqRat readback, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyMeshStability}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6238.1, Definition 6238.2, and Definition 6238.3 seed the finite mesh-refinement stability route from displayed stream windows to Real-facing completion reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyMeshStability}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, quotient Cauchy-stream equality, selected completion points, countable choice for refinements, or ambient RealUp completeness.

Upgrade path: Obligation closure requires checked carrier admission, mesh-source exposure, finite stream-window stability, dyadic tolerance exactness, regular rational readback, Real-seal handoff ordering, transport, replay, provenance, and local naming for BEDC.Derived.CauchyMeshStabilityUp.

Chapter 6239

A Concrete Naming Certificate for $\mathsf{CauchyCompletionPushout}^{\uparrow}$

Following Bishop–Bridges constructive completion and Mac Lane pushout terminology, the CauchyCompletionPusho packet records the finite gluing surface for two completion legs over a shared dense source. It sits between the unit-and-bind schedule of chapter 378, the extension uniqueness route of chapter 397, the metric completion carrier of chapter 1506, the finite stream interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names completion gluing without quotienting Cauchy names, selecting representatives, invoking countable choice, or importing ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6239.1 (Cauchy completion pushout carrier)**.** *A* $\mathsf{CauchyCompletionPushout}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, L, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the shared source row, naming the common metric or Cauchy source and the finite requests on which both completion legs are compared.*
2. *L is the left completion leg row, carrying the displayed dense entry, finite Cauchy windows, and completion-facing readback for the left side.*
3. *R is the right completion leg row, carrying the analogous displayed dense entry, finite Cauchy windows, and completion-facing readback for the right side.*
4. *Q is the compatibility-square row, recording that the two legs agree on the shared source through displayed* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback before any terminal seal is read.*
5. *E is the endpoint seal row, exposing the* $\mathsf{Real}^{\uparrow}$ *or completion-facing endpoint reached after S, L, R, Q have been displayed.*
6. *H records componentwise* hsame *transport for S, L, R, Q, E and for the structural rows.*
7. *C records displayed* Cont *replay from a shared source request through the two completion legs, the compatibility square, and the endpoint seal.*
8. *P is the* Pkg *provenance over* $\mathsf{CauchyCompletionPushout}^{\uparrow}$, $\mathsf{CauchyCompletionMonad}^{\uparrow}$, CompletionUniversalProper $\mathsf{MetricCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionPushout}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for quotient completion equality, a selected pushout representative, countable choice of approximants, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CauchyCompletionPushoutUp.CauchyCompletionPushoutTasteGate single carrier alignment`

Theorem 6239.2 (Cauchy completion pushout NameCert obligations)**.** *For every accepted* $\mathsf{CauchyCompletionPushout}$ *carrier* $X = (S, L, R, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CauchyCompletionPushout}^{\uparrow}}$ *surface consists exactly of shared-source admission, left completion-leg admission, right completion-leg admission, compatibility-square admission, endpoint-seal admission, componentwise transport, displayed replay, provenance, and local naming for the rows* $S, L, R, Q, E, H, C, P, N$*.*

Proof. Project the carrier of Definition 6239.1. The analytic rows are S, L, R, Q, E, and the structural rows are H, C, P, N.

The shared source row is read first. The left and right completion legs are then inspected against the same finite source request, using the finite-window and regular-readback routes already named by the StreamName and RegSeqRat siblings.

The compatibility square Q is available only after both legs have been displayed over S. The endpoint seal E is downstream of that square, so a Real or Completion consumer cannot bypass the compatibility row.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no quotient equality, selected representative, countable-choice schedule, host equality, or ambient completeness row appears in the carrier, the listed rows exhaust the local NameCert obligations. □

Theorem 6239.3 (Cauchy completion pushout nonescape)**.** *Every public read from an accepted* $\mathsf{CauchyCompletionPushout}^{\uparrow}$ *packet factors through*

$$S, L, R \longmapsto Q \longmapsto E \longmapsto H, C, P, N.$$

Thus completion functoriality and Real seal consumers receive a displayed gluing route, not quotient completion equality, selected pushout representatives, countable choice, host equality of completed points, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the NameCert obligations of Definition 6239.2. They expose the source row and both completion legs before the compatibility square is readable.

The endpoint seal is read only after the square. A consumer that reaches E has therefore already consumed the shared-source row, the left and right leg rows, and the square witnessing their agreement on the source.

The structural rows preserve this same order and add no analytic coordinate. Hence every public read factors through the displayed route and cannot export any of the excluded host-side or quotient data. □

Falsifiable prediction: A $\mathsf{CauchyCompletionPushout}^{\uparrow}$ consumer that reads a terminal $\mathsf{Real}^{\uparrow}$ or completion seal without first displaying one shared source row, two completion legs, one compatibility square, transport, replay, provenance, and local naming is not reading the carrier of Definition 6239.1.

Independence witness: cauchycompletionmonad, completionuniversalproperty, metriccompletion, streamname, regseqrat, real: none is bijective to $\mathsf{CauchyCompletionPushout}^{\uparrow}$ because the

packet jointly carries a shared source, two completion legs, a compatibility square, endpoint seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyCompletionPushout}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyCompletionPushout}^{\uparrow}$ packet is generated from a shared source row, left and right completion-leg rows, a compatibility-square row, an endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyCompletionPushout}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6239.1, Definition 6239.2, and Definition 6239.3 seed the finite completion-gluing route over a shared source.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyCompletionPushout}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient completion equality, selected representatives, countable choice, host equality of completed points, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-leg compatibility, endpoint seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CauchyCompletionPushoutUp.

Chapter 6240

A Concrete Naming Certificate for $\mathsf{LocatedLowerBound}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedLowerBound}^{\uparrow}$ packet isolates the finite lower-bound witness that located infimum and compact-minimum arguments may inspect before any ambient completeness or selected minimum principle is available. It is the order-dual reusable face to the $\mathsf{LocatedUpperBound}^{\uparrow}$ packet of chapter 3396, feeds the infimum route of chapter 1452, and keeps its readback tied to the located-real, regular-rational, and real-name surfaces of **??**, chapter 301, and chapter 38. The chapter names lower-bound data as a finite NameCert carrier, not as Real completeness, trichotomy, quotient cuts, selected extrema, or a host order oracle.

Definition 6240.1 (Located lower-bound carrier)**.** *A* $\mathsf{LocatedLowerBound}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, L, W, R, E, H, C, P, N)$$

with the following visible rows:

1. *F is the located source row, read through the* $\mathsf{LocatedReal}^{\uparrow}$ *surface and its displayed comparison windows;*
2. *L is the lower-bound comparison ledger, recording the finite rows that place the proposed lower surface below each inspected source value;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *request schedule that fixes the finite precision windows in which source and lower-bound comparisons are inspected;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that repacks the scheduled comparison observations as regular rational data;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after F, L, W, R have been displayed;*
6. *H records componentwise* hsame *transport for the located source, lower-bound ledger, window schedule, regular readback, Real seal, replay, provenance, and local naming rows;*
7. *C records displayed* Cont *routes for lower-bound reads, regular-rational handoff, Real-seal readback, and downstream located-infimum replay;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedLowerBound}^{\uparrow}$, $\mathsf{LocatedUpperBound}^{\uparrow}$, $\mathsf{LocatedInfimum}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedLowerBound}^{\uparrow}}$ *row.*

The carrier has no row for Real completeness, trichotomy, quotient-cut equality, a selected minimum, an arbitrary infimum operator, countable choice, or host-level order evaluation.

Lean4 (definition) `BEDC.Derived.LocatedLowerBoundUp`

Theorem 6240.2 (Located lower-bound NameCert obligations)**.** *For every accepted* $\mathsf{LocatedLowerBound}^{\uparrow}$ *carrier* $B = (F, L, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedLowerBound}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed packet, source exposure through* F*, lower-bound comparison exactness through* L*, finite request-window compatibility through* W*, regular-rational readback through* R*, terminal Real-seal routing through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.LocatedLowerBoundUp.LocatedLowerBoundCarrier namecert obligations`

Proof. Project the carrier of Definition 6240.1. The analytic rows are precisely F, L, W, R, E, and the structural rows are precisely H, C, P, N.

Carrier admission and source exposure read F before any comparison is consumed. Lower-bound comparison exactness then reads only the displayed entries of L inside the request schedule fixed by W.

The regular-rational and Real-facing obligations are downstream from that schedule. The row R repacks the inspected comparisons, and E seals the Real-facing read only after F, L, W, R have been exposed.

The remaining rows preserve the same packet. The transports in H, replays in C, provenance in P, and local name N add no completeness principle, trichotomy row, quotient-cut equality, selected extremum, or host order evaluator. □

Theorem 6240.3 (Located lower-bound Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{LocatedLowerBound}^{\uparrow}$ *carrier receives the finite route*

$$F, L, W, R, E, H, C, P, N.$$

The consumer must read the located source row, the lower-bound comparison ledger, the request schedule, and the regular-rational readback before the terminal Real seal is available. The route is stable under the displayed hsame *and* Cont *rows, and it exports no Real completeness, trichotomy, quotient cut, selected minimum, arbitrary infimum operator, countable choice, or host order evaluation.*

Proof. Start with an accepted carrier $B = (F, L, W, R, E, H, C, P, N)$. By Definition 6240.2, any Real-facing read must first expose the located source, lower-bound ledger, finite request window, and regular readback rows.

The terminal row E is therefore downstream from F, L, W, R. A consumer that has not read those rows has no coordinate from which to obtain a Real-facing lower-bound certificate.

Transport and replay do not enlarge the route. The rows H and C preserve the displayed packet through componentwise sameness and continuation replay, while P, N source and name the same finite data.

Thus the Real-seal route is exactly $F, L, W, R, E, H, C, P, N$, with the listed excluded principles absent from the carrier. □

Lean4 ✓ `BEDC.Derived.LocatedLowerBoundUp.LocatedLowerBoundRealSealRoute`

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedLowerBound}^{\uparrow}}$**.**

Constructive story: A LocatedLowerBound$^{\uparrow}$ packet is generated from a located source row, a lower-bound comparison ledger, a StreamName request schedule, a RegSeqRat readback row, a RealUp seal, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedLowerBound}\uparrow}$, seed)

Scope closed: Definition 6240.1, Definition 6240.2, and Definition 6240.3 seed the reusable lower-bound face for located infimum and compact-minimum consumers over the L10 Real route.

Formal verification: FormalStatus(NameCert$_{\text{LocatedLowerBound}\uparrow}$, encodedDef)

Lean target: `BEDC.Derived.LocatedLowerBoundUp`

Bridge status: none

Not claimed: This chapter does not claim Real completeness, trichotomy, quotient-cut equality, selected extrema, arbitrary infima, countable choice, host order evaluation, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, lower-bound comparison exactness, request-window compatibility, regular readback, Real-seal routing, transport, replay, provenance, and local naming for the displayed packet.

Chapter 6241

A Concrete Naming Certificate for $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$

The $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$ packet realizes the classifier-shift audit surface of section 9462.9 as finite BEDC certificate data. It sits beside the positive-discovery classifier of chapter 6202, the classifier novelty witness of chapter 6138, the practice-level classifier of chapter 6225, the discovery pipeline calculus of chapter 5914, and the effective replacement ledger of chapter 568. The packet records before and after classifier profiles as a ledgered finite object, rather than as a prose score or a marker-only label change.

Definition 6241.1 (Classifier novelty ledger carrier)**.** *A* $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (B, A, S, J, D, U, Q, H, C, P, N)$$

with the following displayed rows:

1. *B is the before-profile row, naming the source scope, classifier relation, stability mode, ledger status, and seal claim before the inspected shift.*
2. *A is the after-profile row, naming the corresponding source scope, classifier relation, stability mode, ledger status, and seal claim after the inspected shift.*
3. *S is the shared-source row, proving that B and A are compared over the same displayed behavior or source family.*
4. *J is the classifier-novelty witness row, exposing the before and after classifier disagreement inside the shared source scope.*
5. *D is the debt row, separating classifier debt, semantic debt, ledger debt, verification debt, intervention debt, and generalization debt.*
6. *U is the downstream-use row, recording which report, certificate, or consumer reads the after profile rather than merely naming it.*
7. *Q is the public audit row, recording the finite evidence needed to replay the disagreement, debt accounting, and downstream use.*
8. *H records componentwise* hsame *transport for B, A, S, J, D, U, Q and for the structural rows.*
9. *C records displayed* Cont *replay from the shared source through the before profile, after profile, novelty witness, debt row, downstream use, and audit row.*

10. *P is the* Pkg *provenance over* ClassifierNoveltyLedger$^{\uparrow}$, PositiveInterpretabilityClassifier$^{\uparrow}$, ClassifierNoveltyWitness$^{\uparrow}$, PositiveInterpretabilityClassifierPractice$^{\uparrow}$, DiscoveryPipelineCalculus$^{\uparrow}$, EffectiveReplacementLedger$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyLedger}^{\uparrow}}$ *row.*

The carrier has no coordinate for a marker-only classifier change, a detached score, a prose-only novelty claim, a hidden source switch, or a positive-discovery verdict without displayed classifier disagreement and ledger rows.

Theorem 6241.2 (Classifier novelty ledger NameCert obligations)**.** *For every accepted* ClassifierNoveltyLedger$^{\uparrow}$ *carrier $L = (B, A, S, J, D, U, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyLedger}^{\uparrow}}$ *surface consists exactly of before-profile admission, after-profile admission, shared-source admission, classifier-novelty witness admission, debt-row admission, downstream-use admission, public-audit admission, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6241.1. The profile rows are B, A, the comparison rows are S, J, the ledger rows are D, U, Q, and the structural rows are H, C, P, N.

The shared-source row is read before classifier disagreement. Thus the novelty witness cannot be manufactured by changing the source family or observation protocol.

The debt, downstream-use, and public-audit rows are read next. They record why the classifier shift matters to a consumer and which finite evidence replays the disagreement.

Finally H, C, P, N transport, replay, source, and name the same packet. Since marker-only labels, detached scores, hidden source switches, prose-only novelty claims, and positive-discovery verdicts have no carrier coordinate, the listed rows are exactly the local obligations. □

Theorem 6241.3 (Classifier novelty ledger rejects marker-only shifts)**.** *An accepted* ClassifierNoveltyLedger$^{\uparrow}$ *packet cannot be obtained by changing only an external marker attached to an unchanged classifier profile. If B and A agree on shared source, classifier relation, stability mode, ledger status, debt row, downstream use, and public audit row, then the packet has no classifier-novelty witness row J.*

Proof. Read the obligation surface of Definition 6241.2. The novelty witness J is a displayed classifier-disagreement row over the shared source S.

If the before and after profiles agree on the classifier relation and the supporting rows D, U, Q, then a marker change is outside the carrier inventory. It is not a row that J can read.

The structural rows preserve the displayed coordinates and do not create a new disagreement. Hence an external marker change alone cannot satisfy the novelty ledger. □

Theorem 6241.4 (Classifier novelty ledger handoff)**.** *Every downstream read from an accepted* ClassifierNoveltyLedger$^{\uparrow}$ *packet factors through*

$$B, A, S \longmapsto J \longmapsto D, U, Q \longmapsto H, C, P, N.$$

The consumer receives before and after classifier-profile disagreement as finite certificate data, not a detached theorem-DNA score or an unledgered interpretability label.

Proof. Start from the carrier. The before and after profiles and the shared source must be visible before J can certify classifier disagreement.

The rows D, U, Q then bind the disagreement to debt accounting, downstream use, and replayable audit evidence. Without those rows, the classifier shift has no public ledger surface.

Transport, replay, provenance, and local naming preserve that route. Therefore every downstream read factors through the displayed finite packet and cannot replace it by a detached score or unledgered label. □

Falsifiable prediction: A report that changes only a classifier marker while preserving the same source scope, classifier relation, stability mode, debt row, downstream-use row, and public audit row cannot satisfy the $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$ carrier of Definition 6241.1.

Independence witness: positive-interpretability-classifier, classifier-novelty-witness, positive-interpretability-classifier-practice, discovery-pipeline-calculus, effective-replacement-ledger: none is bijective to $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$ because the packet jointly carries before and after profiles, shared-source comparison, classifier disagreement, debt, downstream-use, public-audit, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{ClassifierNoveltyLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ClassifierNoveltyLedger}^{\uparrow}$ packet is generated from before and after classifier profiles, a shared-source row, a classifier-novelty witness, debt and downstream-use rows, a public audit row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ClassifierNoveltyLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6241.1, Definition 6241.2, Definition 6241.3, and Definition 6241.4 seed the finite before and after classifier-profile ledger for NDNA-style novelty audits.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ClassifierNoveltyLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive-discovery status, global interpretability quality, detached theorem-DNA scoring, prose-only novelty, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked before and after profile admission, shared-source comparison, classifier-disagreement exposure, debt and downstream-use rows, public-audit replay, transport, provenance, and local naming for BEDC.Derived.ClassifierNoveltyLedgerUp.

Chapter 6242

A Concrete Naming Certificate for $\mathsf{OsgoodModulus}^{\uparrow}$

Following Osgood and Bishop–Bridges constructive ODE uniqueness, the $\mathsf{OsgoodModulus}^{\uparrow}$ packet records the finite modulus-divergence surface consumed by uniqueness criteria before an ODE solution comparison may reach a Real-facing seal. It sits beside the continuous-map carrier of chapter 130, the ODE row of chapter 193, the Osgood uniqueness route of chapter 5199, the Osgood criterion route of chapter 6159, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The packet is a finite divergence ledger for uniqueness consumers; it is not a host improper-integral theorem, a Lipschitz constant, a contraction ratio, a selected solution representative, or quotient equality of solution traces.

Definition 6242.1 (OsgoodModulus carrier)**.** *An* $\mathsf{OsgoodModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (O, V, A, T, Q, R, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the* $\mathsf{Ode}^{\uparrow}$ *source row, recording the time-state window and the uniqueness request that will consume the modulus data.*
2. *V is the* $\mathsf{ContinuousMap}^{\uparrow}$ *vector-field row on the same ODE window, read as displayed continuity rather than as a host function.*
3. *A is the modulus schedule row, listing positive* $\mathsf{RegSeqRat}^{\uparrow}$ *radius names, monotone comparison bounds, and the admissible near-zero window.*
4. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording dyadic scales, split radii, slack, and comparison witnesses used by the schedule.*
5. *Q is the finite divergence ledger, recording reciprocal-modulus budget rows and lower-window checks rather than a host improper integral.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the radius, budget, and solution-distance traces inspected by downstream consumers.*
7. *L is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after O, V, A, T, Q, R have been displayed.*
8. *H records componentwise* hsame *transport for ODE, continuous-map, schedule, tolerance, divergence, readback, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$O, V \longmapsto A, T, Q \longmapsto R \longmapsto L.$$

10. *P is the* Pkg *provenance over* $\mathsf{OsgoodModulus}^{\uparrow}$, $\mathsf{Ode}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{OsgoodModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, V, A, T, Q, R, L, H, C, P, N$. It has no coordinate for a Lipschitz constant, contraction ratio, selected solution representative, host integral theorem, quotient equality of curves, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.OsgoodModulusUp.OsgoodModulusTasteGate single carrier alignment`

Theorem 6242.2 (OsgoodModulus NameCert obligations)**.** *For every accepted* $\mathsf{OsgoodModulus}^{\uparrow}$ *carrier*

$$M = (O, V, A, T, Q, R, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OsgoodModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed ODE source, continuous vector-field, modulus schedule, dyadic tolerance, divergence ledger, regular readback, Real seal, transport, replay, provenance, and local naming rows;*
2. *ODE and continuity visibility, requiring O and V to be public before any modulus or divergence budget is read;*
3. *schedule exactness, requiring A to display only positive* $\mathsf{RegSeqRat}^{\uparrow}$ *radius names, monotone comparison bounds, and the near-zero window used by the uniqueness consumer;*
4. *tolerance exactness, requiring T to record every dyadic scale, split radius, slack, and comparison witness used by the displayed schedule;*
5. *divergence-ledger exactness, requiring Q to store finite reciprocal-modulus budget rows and lower-window checks, not a host improper integral;*
6. *readback and seal nonescape, requiring R, L, H, C, P, N to read, seal, transport, replay, source, and name only the displayed route.*

No Osgood modulus obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6242.1. The accepted packet displays exactly the ODE source, continuous vector-field, modulus schedule, dyadic tolerance, divergence ledger, regular readback, Real seal, and structural rows.

The source and continuity clauses are read from O, V. A uniqueness-facing consumer therefore sees the ODE window and continuous vector-field row before it may inspect a modulus schedule or divergence budget.

The schedule and tolerance clauses are read from A, T. The schedule exposes positive regular-rational radius names, monotone comparison bounds, and the near-zero window, while the dyadic row records the scales and slack used by those comparisons.

The divergence and readback clauses are read from Q, R, L. The ledger Q stores finite reciprocal-modulus budget checks, R reads the radius and solution-distance traces, and L seals the Real-facing boundary only after the displayed route is present. The structural rows H, C, P, N transport, replay, source, and name that same route, so the carrier has no place to store the excluded host or choice data. □

Theorem 6242.3 (OsgoodModulus divergence handoff)**.** *Every uniqueness or criterion consumer of an accepted* $\mathsf{OsgoodModulus}^{\uparrow}$ *carrier must pass through*

$$\mathsf{Ode}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ Q \longmapsto \mathsf{Real}^{\uparrow}$$

as displayed by Definition 6242.1. The consumer receives a finite divergence ledger and tolerance schedule for Osgood comparison, not a Lipschitz constant, contraction certificate, selected solution curve, host improper-integral theorem, or quotient equality of solution traces.

Proof. Begin with a uniqueness-facing consumer of M. The only source row is O, and the only vector-field row is V, so the consumer is confined to the displayed ODE window and continuous-map readback.

Next read the modulus rows. The row A supplies the regular-rational radius schedule, T supplies the dyadic tolerance arithmetic, and Q supplies only finite reciprocal-modulus budget checks.

Finally read R, L, H, C, P, N. These rows provide regular readback, terminal Real sealing, componentwise transport, displayed replay, provenance, and local naming for the same route. Since no other coordinate is present, the handoff cannot export the excluded Lipschitz, contraction, selection, host-integral, quotient, choice, completeness, verification, or bridge data. □

Theorem 6242.4 (OsgoodModulus reciprocal-budget divergence row)**.** *For every accepted* $\mathsf{OsgoodModulus}^{\uparrow}$ *carrier*

$$M = (O, V, A, T, Q, R, L, H, C, P, N),$$

every Osgood uniqueness consumer reads the finite reciprocal-modulus budget row Q only after the ODE source row O, continuous vector-field row V, positive $\mathsf{RegSeqRat}^{\uparrow}$ *radius schedule A, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row T are displayed, and before the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R and* $\mathsf{Real}^{\uparrow}$ *seal L are exported. The row Q is therefore consumed as finite divergence data internal to the displayed packet, not as a host improper integral or as external completeness, choice, selection, contraction, Lipschitz, or quotient-curve data.*

Proof. Project the accepted carrier through Definition 6242.1. Its displayed coordinates put O, V, A, T, Q, R, L in the only route available to a uniqueness consumer, with H, C, P, N transporting, replaying, sourcing, and naming the same packet.

Apply Definition 6242.2 to identify the exact rows. The source and continuity clauses expose O, V, the schedule and tolerance clauses expose A, T, and the divergence-ledger clause identifies Q as finite reciprocal-modulus budget data rather than a host improper integral.

Apply Definition 6242.3 for the consumer order. The handoff sends the consumer through the displayed ODE and continuous-map rows, then through the regular-rational schedule, dyadic tolerance arithmetic, and Q, before any readback or Real-facing seal is exported. Since the carrier has no coordinate for the excluded host or choice data, the reciprocal-budget row cannot export a Lipschitz constant, contraction ratio, selected solution representative, quotient curve equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness. □

Falsifiable prediction: Every accepted $\mathsf{OsgoodModulus}^{\uparrow}$ uniqueness read that reaches $\mathsf{RegSeqRat}^{\uparrow}$ distance readback or a $\mathsf{Real}^{\uparrow}$ seal displays one $\mathsf{Ode}^{\uparrow}$ source row, one $\mathsf{ContinuousMap}^{\uparrow}$ vector-field row, one modulus schedule, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one finite divergence ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: ode, continuousmap, osgood-uniqueness, osgood-criterion, regseqrat, dyadicratcore, real: no listed sibling supplies the joint finite packet containing an ODE source,

continuous vector-field row, positive regular-rational radius schedule, dyadic tolerance row, divergence ledger, regular readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{OsgoodModulus}\uparrow}$.

Constructive story: An $\mathsf{OsgoodModulus}^{\uparrow}$ packet is generated from an OdeUp source row, a ContinuousMapUp vector-field row, a RegSeqRatUp radius schedule, a DyadicRatCoreUp tolerance row, a finite divergence ledger, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OsgoodModulus}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6242.1, Definition 6242.2, Definition 6242.3, and Definition 6242.4 seed the finite Osgood modulus route from displayed ODE and ContinuousMap rows through regular-rational radius schedules, dyadic tolerance arithmetic, divergence ledgers, readback, and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OsgoodModulus}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Lipschitz uniqueness, Banach contraction, selected solution representatives, quotient equality of curves, countable choice, host improper-integral theorems, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, ODE-window visibility, continuous-map readback, modulus schedule exactness, dyadic tolerance exactness, divergence-ledger exactness, regular readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OsgoodModulusUp.

Chapter 6243

A Concrete Naming Certificate for $\mathsf{TheoremDnaScore}^{\uparrow}$

The $\mathsf{TheoremDnaScore}^{\uparrow}$ packet realizes the neural theorem-DNA scoring surface of section 9462.9 as finite NameCert data. It sits beside the proof trace normal form of chapter 5157, the tensor product carrier of chapter 87, the black-box decomposition certificate of chapter 6100, the audit gate boundary of chapter 526, the certificate audit gate of chapter 1002, the discovery pipeline calculus of chapter 5914, and the positive interpretability classifier of chapter 6202. The packet names a finite scoring carrier for source, feature, circuit, classifier, stability, intervention, ledger, and seal rows; it is not a neural oracle, a hidden tensor explanation, or an unledgered taste judgment.

Definition 6243.1 (TheoremDnaScore carrier)**.** *A* $\mathsf{TheoremDnaScore}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, G, F, U, K, A, I, L, E, W, H, C, P, N)$$

with the following displayed rows:

1. *B is the behavior or theorem-like target row whose explanation is being scored.*
2. *G is the role-complete theorem-DNA source row, displaying source, feature, circuit, classifier, stability, intervention, ledger, and seal loci.*
3. *F is the feature row, including tensor direction, probe, sparse-latent, attention, residual-stream, or proof-feature evidence visible to the score.*
4. *U is the circuit row, recording the causal or proof-route structure that the feature row claims to explain.*
5. *K is the classifier row induced by B, G, F, U, with its declared boundary and comparison relation.*
6. *A is the stability row, recording perturbation, representation, transfer, or proof-normalization stability over the classifier boundary.*
7. *I is the intervention row, including ablation, activation transfer, proof-route edit, failed-variant, off-target, or counterfactual evidence.*
8. *L is the ledger row, recording residue, failed loci, debt, and audit boundary for the score.*
9. *E is the seal row, declaring the score scope, critical-locus coverage, hardening state, and not-claimed boundary.*

10. *W is the weight and score row, recording nonnegative locus weights, displayed costs, certified gain, and the finite comparison verdict.*
11. *H records componentwise* hsame *transport for behavior, genome, feature, circuit, classifier, stability, intervention, ledger, seal, score, replay, provenance, and local naming rows.*
12. *C records displayed* Cont *replay from the source behavior through loci, classifier, stability, intervention, ledger, seal, and score rows.*
13. *P is the* Pkg *provenance over* $\mathsf{TheoremDnaScore}^{\uparrow}$, BHist, *tensor-feature packets, proof-trace packets, audit-gate packets,* hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $B, G, F, U, K, A, I, L, E, W, H, C, P, N$. It has no coordinate for hidden model knowledge, unobserved circuit causality, natural-language substitution for anchors, unlabeled debt, unscoped taste, Lean verification, or bridge checking.

Theorem 6243.2 (TheoremDnaScore NameCert obligations)**.** *For every accepted* $\mathsf{TheoremDnaScore}^{\uparrow}$ *carrier*

$$S = (B, G, F, U, K, A, I, L, E, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed behavior, theorem-DNA loci, feature, circuit, classifier, stability, intervention, ledger, seal, score, transport, replay, provenance, and local naming rows;*
2. *role completeness, requiring G to display source, feature, circuit, classifier, stability, intervention, ledger, and seal loci before W may score the packet;*
3. *classifier faithfulness, requiring K to be induced by the displayed behavior, feature, and circuit rows and to expose its boundary before stability or intervention evidence is counted;*
4. *stability and intervention exactness, requiring A, I to name the perturbations, transfers, edits, failures, off-target effects, and counterfactuals that support the score;*
5. *ledger and seal exactness, requiring L, E to record residue, debt, critical-locus coverage, scope, hardening state, and not-claimed boundary;*
6. *score nonescape, requiring W, H, C, P, N to score, transport, replay, source, and name only the displayed route.*

No theorem-DNA score obligation is stored outside these rows.

Proof. Project the carrier of Definition 6243.1. The accepted packet displays exactly the behavior, role-complete theorem-DNA row, feature, circuit, classifier, stability, intervention, ledger, seal, score, and structural rows.

Role completeness is read from G. Because G displays source, feature, circuit, classifier, stability, intervention, ledger, and seal loci before W is read, a score cannot be certified from a single feature label, isolated circuit diagram, classifier-only relation, or narrative seal.

Classifier faithfulness is read from B, F, U, K. The classifier row is tied to the displayed behavior and explanatory rows, so the stability and intervention rows can only support the declared boundary.

The remaining obligations are read from $A, I, L, E, W, H, C, P, N$. Stability and intervention evidence is ledgered before the seal and score are consumed, and the structural rows only transport, replay, source, and name the same route. Since the carrier has no coordinate for hidden model

knowledge, unobserved causality, unanchored language, unlabeled debt, or unscoped taste, the listed rows exhaust the NameCert surface. □

Theorem 6243.3 (TheoremDnaScore critical-locus refusal)**.** *If an accepted* $\mathsf{TheoremDnaScore}^{\uparrow}$ *scoring request omits a critical source, feature, circuit, classifier, stability, intervention, ledger, or seal locus, then the local* $\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}$ *refuses certification. A certified score may be read only after the omitted locus is supplied as a displayed row and replayed through* H, C, P, N*.*

Proof. Use Definition 6243.2. Role completeness requires all eight loci to be visible before the score row W is read.

If a critical locus is absent, the corresponding row in G is not displayed. The score row therefore lacks one of the rows required for carrier admission and role completeness.

The structural rows cannot repair the gap. Transport, replay, provenance, and local naming preserve displayed rows; they do not create missing source, feature, circuit, classifier, stability, intervention, ledger, or seal content. Hence certification is refused until the missing locus is supplied and replayed through the finite packet. □

Falsifiable prediction: Any $\mathsf{TheoremDnaScore}^{\uparrow}$ packet missing a critical source, feature, circuit, classifier, stability, intervention, ledger, or seal locus fails local certification before the score row can be read.

Independence witness: proof-trace-normal-form, tensor-namecert, black-box-ledger, auditgateboundary, certificateauditgate: $\mathsf{TheoremDnaScore}^{\uparrow}$ is not bijective to any listed sibling because proof-trace-normal-form supplies proof-route shape without neural loci, tensor-namecert supplies feature carriers without score weights, black-box-ledger records residue without critical-locus scoring, auditgateboundary supplies refusal boundary without theorem-DNA weights, and certificateauditgate supplies admission discipline without the role-complete score row.

Closure status for $\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TheoremDnaScore}^{\uparrow}$ packet is generated from a behavior or theorem-like target, role-complete theorem-DNA loci, feature, circuit, classifier, stability, intervention, ledger, seal, score, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row. The Lean follow-up BEDC.Derived.TheoremDnaScoreUp.TasteGate must provide Nontrivial and FieldFaithful instances before TasteGate admission is claimed.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6243.1, Definition 6243.2, and Definition 6243.3 seed the finite theorem-DNA scoring route over role-complete loci, classifier faithfulness, stability, intervention, ledger, seal, score, and nonescape rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoremDnaScore}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, neural circuit truth, hidden-model explanation, positive discovery, unscoped taste, or classifier novelty beyond the displayed score carrier.

Upgrade path: Obligation closure requires checked carrier admission, role completeness, classifier faithfulness, stability and intervention exactness, ledger and seal exactness, score nonescape, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful TasteGate instances for BEDC.Derived.TheoremDnaScoreUp.TasteGate.

Chapter 6244

A Concrete Naming Certificate for $\mathsf{NoetherNormalization}^{\uparrow}$

Following Noether normalization in commutative algebra, the $\mathsf{NoetherNormalization}^{\uparrow}$ packet records the finite row surface by which an affine algebra is displayed as integral over a polynomial coordinate subalgebra. It sits after the polynomial and affine-variety carriers of chapter 50 and chapter 154, the algebraic source rows of chapter 43, chapter 46, and chapter 45, and the reduction ledger supplied by chapter 2197. The packet names only displayed coordinate rows, monic dependence witnesses, finite module-spanning ledgers, transport, replay, provenance, and a local certificate; it does not construct quotient rings, select bases, assert dimension equality, or import a host algebraic-geometry category.

Definition 6244.1 (Noether normalization carrier)**.** *A* $\mathsf{NoetherNormalization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$N = (F, R, P, A, G, C, I, M, H, Q, K)$$

with the following displayed rows:

1. *F is the* $\mathsf{Field}^{\uparrow}$ *source row over which the affine coordinate algebra is read.*
2. *R is the* $\mathsf{Ring}^{\uparrow}$ *row naming the accepted finitely generated algebra surface.*
3. *P is the* $\mathsf{Poly}^{\uparrow}$ *coordinate row, including the displayed finite list of polynomial coordinates used for the independent subalgebra.*
4. *A is the* $\mathsf{AffineVar}^{\uparrow}$ *source row whose coordinate data is consumed by R and P.*
5. *G is the* $\mathsf{GroebnerBasis}^{\uparrow}$ *reduction ledger used only as a finite dependency anchor for displayed polynomial reduction checks.*
6. *C is the coordinate-change ledger. It records the chosen finite coordinate rows and the replay order by which the polynomial subalgebra is presented before any dependence row is consumed.*
7. *I is the finite integral-dependence ledger. For each displayed algebra generator it records a monic polynomial equation over the coordinate subalgebra and the BEDC continuation path that reads its coefficients.*
8. *M is the* $\mathsf{Module}^{\uparrow}$ *finite-spanning ledger, recording the displayed list of module words read from the integral-dependence rows.*

9. *H records componentwise* hsame *transport for field, ring, polynomial, affine-variety, reduction, coordinate-change, integral-dependence, module-spanning, replay, provenance, and local naming rows.*

10. *Q records displayed* Cont *replay from an affine algebra request through $F, R, P, A, G, C, I, M, H$ and only then to provenance and naming.*

11. *K is the* Pkg *provenance together with the local* $\mathsf{NameCert}_{\mathsf{NoetherNormalization}^{\uparrow}}$ *row.*

Accepted carriers are compared only through these rows. There is no coordinate for a quotient-ring construction, selected basis, external transcendence basis, dimension theorem, or ambient algebraic-geometry category.

Lean4 ✓ `BEDC.Derived.NoetherNormalizationUp.NoetherNormalizationTasteGate single carrier alignment`

Theorem 6244.2 (Noether normalization NameCert obligations)**.** *For every accepted* $\mathsf{NoetherNormalization}^{\uparrow}$ *carrier $N = (F, R, P, A, G, C, I, M, H, Q, K)$, the local* $\mathsf{NameCert}_{\mathsf{NoetherNormalization}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed* $\mathsf{Field}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{AffineVar}^{\uparrow}$, $\mathsf{GroebnerBasis}^{\uparrow}$, *coordinate-change, integral-dependence, finite* $\mathsf{Module}^{\uparrow}$*-spanning, transport, replay, provenance, and local naming rows.*

Proof. Project the carrier of Definition 6244.1. The algebraic source rows are F, R, P, A, G, the normalization rows are C, I, M, and the structural rows are H, Q, K.

The coordinate-change row C is read before any dependence row, so the polynomial coordinate subalgebra is a displayed BEDC dependency rather than a quotient construction. The integral-dependence row I records only monic equations for the displayed generators over that coordinate surface.

The module-spanning row M is downstream of I and supplies finite words produced from those monic equations. Transport, replay, provenance, and local naming do not add quotient rings, selected bases, dimension statements, or host categories, so the listed rows exhaust the NameCert obligations. □

Theorem 6244.3 (Noether normalization finite integral-dependence route)**.** *Every accepted finite integral-dependence read exported by a* $\mathsf{NoetherNormalization}^{\uparrow}$ *carrier factors through the displayed route*

$$F, R, P, A, G, C \longmapsto I \longmapsto H, Q, K.$$

The read first exposes the field, ring, polynomial-coordinate, affine-variety, and reduction rows, then the finite coordinate-change ledger, and only then the monic dependence equations. It exports no quotient-ring carrier, selected basis, or external algebraic closure.

Proof. Start with an accepted dependence read. Carrier admission exposes the source algebra only through the rows F, R, P, A and the finite reduction ledger G.

The coordinate-change row C fixes the displayed polynomial coordinate surface before the dependence ledger can be inspected. Hence the coefficients of each monic equation are read from the shown coordinate rows, not from an ambient quotient presentation.

The integral-dependence row I then supplies the monic equations, and H, Q, K preserve that finite route under transport, replay, provenance, and naming. No extra coordinate can introduce selected bases, quotient rings, or host closure data. □

Theorem 6244.4 (Noether normalization finite module stability)**.** *Let $N = (F, R, P, A, G, C, I, M, H, Q, K)$ and $N' = (F', R', P', A', G', C', I', M', H', Q', K')$ be accepted* $\mathsf{NoetherNormalization}^{\uparrow}$ *carriers whose*

source, coordinate-change, and integral-dependence rows are related componentwise by hsame*. Then their finite* Module$^{\uparrow}$*-spanning ledgers classify the same normalization boundary: a consumer reading M may be transported to M' through the same displayed dependence route, and no independent basis-selection row is required.*

Proof. The componentwise hsame hypotheses identify the accepted field, ring, polynomial, affine-variety, reduction, coordinate-change, and integral-dependence rows of the two packets. Transport by H and H' therefore aligns the monic equations used to produce the finite module words.

The module ledgers M and M' are downstream of those equations. Their spanning reads are replayed through Q and Q' from the same displayed dependence route, so a consumer sees the same finite normalization boundary after transport.

Since the carrier has no basis-selection coordinate, the stability claim cannot depend on choosing a free basis or a quotient-ring presentation. It depends only on the displayed integral-dependence and module-spanning rows. □

Falsifiable prediction: Every accepted NoetherNormalization$^{\uparrow}$ consumer that reads integral dependence or finite module spanning must first display Field$^{\uparrow}$, Ring$^{\uparrow}$, Poly$^{\uparrow}$, AffineVar$^{\uparrow}$, GroebnerBasis$^{\uparrow}$, coordinate-change, transport, replay, provenance, and local NameCert rows.

Independence witness: polynomial, affinevar, ring, module, field, groebnerbasis: NoetherNormalization$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries coordinate-change rows, monic dependence witnesses, finite module-spanning ledgers, transport, replay, provenance, and local naming.

Closure status for NameCert$_{\text{NoetherNormalization}^{\uparrow}}$.

Constructive story: A NoetherNormalization$^{\uparrow}$ packet is generated from FieldUp, RingUp, PolynomialUp, AffineVarUp, GroebnerBasisUp, coordinate-change, monic integral-dependence, finite ModuleUp-spanning, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{NoetherNormalization}^{\uparrow}}$, seed)

Scope closed: Definition 6244.1, Definition 6244.2, Definition 6244.3, and Definition 6244.4 seed the finite normalization route from displayed affine coordinate data to integral-dependence and module-spanning ledgers.

Formal verification: FormalStatus(NameCert$_{\text{NoetherNormalization}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient rings, selected bases, dimension equality, transcendence-basis existence, host algebraic geometry, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, coordinate-change exposure, finite integral-dependence exactness, finite module-spanning stability, transport, replay, provenance, and local naming for BEDC.Derived.NoetherNormalizatio

Chapter 6245

A Concrete Naming Certificate for $\mathsf{CoxeterGroup}^{\uparrow}$

Following Coxeter's presentation of reflection groups and Bourbaki's treatment of Coxeter systems, the $\mathsf{CoxeterGroup}^{\uparrow}$ packet records a group generated by involutive simple generators with finite braid-relation rows. It sits between the algebraic group surface of chapter 41, the root-reflection ledger of chapter 144, and the Weyl-group consumer of chapter 145. The packet names the displayed Coxeter matrix, finite alternating-word ledgers, relation witnesses, word-reduction obligations, transport, replay, provenance, and local naming surface; it does not assert a host quotient group, a complete word-problem solver, or a classification of finite reflection groups.

Definition 6245.1 (CoxeterGroup carrier)**.** *A* $\mathsf{CoxeterGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (S, M, W, R, N, H, Q, P, L)$$

with the following displayed rows:

1. *S is the finite simple-generator row, with each generator tagged by an involution witness read through the* $\mathsf{Group}^{\uparrow}$ *operation surface.*
2. *M is the Coxeter matrix row. It records $m_{s,s} = 1$ and, for distinct displayed generators s, t, a finite braid length $m_{s,t}$ or an obstruction row stating that no finite braid row is exposed.*
3. *W is the finite signed-word carrier generated from S by empty word, singleton generator word, and concatenation.*
4. *R is the relation ledger. It contains involution rows $ss \rightsquigarrow e$ and, whenever M exposes a finite value $m_{s,t}$, the two alternating words of length $m_{s,t}$ with a classified relation witness between them.*
5. *N is the word-reduction obligation row. It records the local cancellation and braid-step routes that any public normal-form or reduction consumer must read before claiming a reduced endpoint.*
6. *H records componentwise* hsame *transport for S, M, W, R, N, Q, P, L.*
7. *Q records displayed* Cont *replay for generator insertion, concatenation, involution cancellation, braid transport, and reduction-step replay.*
8. *P is the* Pkg *provenance over* $\mathsf{CoxeterGroup}^{\uparrow}$, $\mathsf{Group}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, $\mathsf{WeylGroup}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. L is the local $\mathsf{NameCert}_{\mathsf{CoxeterGroup}^\uparrow}$ *row.*
The classifier compares Coxeter packets only through $S, M, W, R, N, H, Q, P, L$. It does not compare ambient presentations by host equality and does not identify the packet with a Weyl group unless the Weyl consumer supplies the root-action rows separately.

Lean4 ✓ `BEDC.Derived.CoxeterGroupUp.CoxeterGroupTasteGate single carrier alignment`

Theorem 6245.2 (CoxeterGroup word-reduction obligation)**.** *Every public word-reduction read from an accepted* $\mathsf{CoxeterGroup}^\uparrow$ *packet must factor through the displayed relation ledger R and reduction obligation row N. A finite generator word in W alone is only a carried word endpoint; it is not a reduced representative, a normal form, or a word-problem decision procedure until the involution, braid, and replay rows named by R, N, Q have been read.*

Proof. Project the carrier of Definition 6245.1. The row W supplies finite words generated from the simple-generator row S, but it records only construction of words by empty word, singleton generator, and concatenation.

Reduction information enters through R and N. The relation ledger R gives the involution rows and finite braid rows selected by the Coxeter matrix M, while N records the obligation that a consumer must use those rows as the public reduction surface.

The replay row Q then transports each cancellation or braid step through displayed Cont data, and the transport row H keeps the endpoints classified under hsame. Thus a public reduced-word claim is downstream of R, N, Q, H, not a property of the bare word carrier W. □

Theorem 6245.3 (CoxeterGroup NameCert obligations)**.** *A* $\mathsf{NameCert}_{\mathsf{CoxeterGroup}^\uparrow}$ *obligation surface consists exactly of the finite simple-generator carrier, the Coxeter matrix row, the word carrier, the involution and braid relation ledger, the word-reduction obligation row, componentwise transport, continuation replay, provenance, and local naming rows. The surface may be consumed by* $\mathsf{WeylGroup}^\uparrow$ *only after the* $\mathsf{RootSystem}^\uparrow$ *reflection and Cartan rows supply the root-action interpretation of the same relation ledger.*

Proof. Read the rows $S, M, W, R, N, H, Q, P, L$ from the carrier. These rows provide the whole Coxeter packet: generators, relation parameters, finite word construction, relation witnesses, reduction-facing obligations, classifier transport, replay, provenance, and the local certificate name.

The $\mathsf{Group}^\uparrow$ dependency supplies the operation, unit, inverse, and classifier rows needed to read generator words as group-facing histories. The Coxeter packet itself adds the matrix and braid ledger, but it does not add root actions or a finite-reflection representation.

For a Weyl consumer, the root-action interpretation must be supplied separately through the $\mathsf{RootSystem}^\uparrow$ reflection and Cartan ledger rows. Only after those rows are read can chapter 145 consume the Coxeter relation ledger as a root-action ledger, so the Coxeter certificate remains a distinct dependency surface. □

Falsifiable prediction: A $\mathsf{CoxeterGroup}^\uparrow$ consumer that exports a reduced word or Weyl action from W without reading R, N, Q refutes the carrier surface.

Independence witness: group, rootsystem, weylgroup, braidgroup: the group packet lacks the Coxeter matrix and braid ledger; the root-system packet has reflections and Cartan rows but no word carrier; the Weyl-group packet consumes root-action rows beyond the Coxeter relation carrier; the braid-group packet records Artin braid words without the involution relation $ss \rightsquigarrow e$.

Closure status for $\mathsf{NameCert}_{\mathsf{CoxeterGroup}^\uparrow}$.

Constructive story: A CoxeterGroup$^{\uparrow}$ packet is generated from a finite simple-generator row, Coxeter matrix row, finite word carrier, involution and braid relation ledger, word-reduction obligation row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{CoxeterGroup}\uparrow}$, seed)

Scope closed: Definition 6245.1, Definition 6245.2, and Definition 6245.3 seed the finite generator, Coxeter-matrix, word-carrier, relation-ledger, reduction-obligation, transport, replay, provenance, and local naming surface.

Formal verification: FormalStatus(NameCert$_{\mathsf{CoxeterGroup}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, a host quotient presentation, a complete word-problem decision procedure, finite reflection-group classification, or Weyl-group root-action soundness.

Upgrade path: Obligation closure requires checked generator, matrix, word-carrier, involution, braid, reduction-obligation, transport, replay, provenance, and local naming rows for BEDC.Derived.CoxeterGroupUp.

Chapter 6246

A Concrete Naming Certificate for $\mathsf{LipschitzCompletionExtension}^{\uparrow}$

Following Bishop and Bridges on metric completion and Lipschitz extension, the $\mathsf{LipschitzCompletionExtension}^{\uparrow}$ packet records the finite route by which a displayed Lipschitz map on a dense source row is handed to a completion target. It sits after the completion carrier of chapter 95, the complete-metric boundary of chapter 128, the finite window surface of chapter 126, and the regular rational readback of chapter 301. The packet is a completion-frontier NameCert surface; it is not a host extension theorem, a selected limit operation, quotient equality of completed points, countable choice, or ambient completeness imported from outside the displayed target seal.

Definition 6246.1 (Lipschitz completion extension carrier)**.** *A* $\mathsf{LipschitzCompletionExtension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (D, M, L, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense source row, naming the source carrier, dense observation interface, and the finite source points on which the map is already displayed.*
2. *M is the metric row, recording source and target distance readbacks, tolerance windows, and the metric comparison rows consumed by the extension route.*
3. *L is the Lipschitz modulus row, listing the rational modulus bound, its tolerance rescalings, and the finite checks that transport source closeness into target closeness.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *row, exposing the finite source windows and target windows inspected by the route.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the source stream, target stream, distance budget, and modulus-scaled tolerance requests.*
6. *E is the completion target seal, recording that the target-side Cauchy row is admitted only through the displayed completion boundary.*
7. *H records componentwise* hsame *transport for the dense source, metric, modulus, stream-window, regular-rational, completion-seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$D, M, L \longmapsto S, Q \longmapsto E.$$

9. *P is the* Pkg *provenance over* LipschitzCompletionExtension$^{\uparrow}$, Completion$^{\uparrow}$, CompleteMetric$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{LipschitzCompletionExtension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, M, L, S, Q, E, H, C, P, N$. It has no coordinate for a host extension operator, selected target limit, quotient equality of completed points, countable choice, global completeness of an ambient space, Lean verification, or bridge checking.

Lean4 ✓ `BEDC.Derived.LipschitzCompletionExtensionUp.LipschitzCompletionExtensionTasteGate`
`single carrier alignment`

Theorem 6246.2 (Lipschitz completion extension NameCert obligations)**.** *For every accepted* LipschitzCompletionExtension$^{\uparrow}$ *carrier*

$$X = (D, M, L, S, Q, E, H, C, P, N),$$

the local NameCert$_{\text{LipschitzCompletionExtension}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the dense source row, metric row, Lipschitz modulus row, StreamName windows, RegSeqRat readback, completion target seal, transport, replay, provenance, and local naming rows;*
2. *Lipschitz-modulus transport, requiring the rational bound and tolerance rescalings in L to be read before any target closeness claim is exported;*
3. *dense-window handoff, requiring each source observation used by the extension request to pass through the finite D, S, Q route;*
4. *completion target sealing, requiring the target-side Cauchy row to enter through E after the displayed windows and regular rational readback are visible;*
5. *nonescape from selected limits or host completeness, requiring H, C, P, N to transport, replay, source, and name only the rows D, M, L, S, Q, E.*

No Lipschitz completion extension obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6246.1. The accepted packet displays the dense source, metric comparison, Lipschitz modulus, finite windows, regular rational readback, completion seal, and structural rows.

The modulus clause is read from L together with the metric row M. A target closeness request must therefore cite the displayed rational Lipschitz bound and the tolerance rescalings before it can reach a target-side Cauchy claim.

The dense-window handoff is read from D, S, Q. The source observations are first exposed as dense rows, then inspected through finite StreamName windows, and then repacked as regular rational readback at the displayed tolerance.

The completion seal is read from E. The structural rows H, C, P, N transport, replay, source, and name the same route, so the carrier contains no field where a selected limit, quotient equality, countable-choice schedule, host extension operator, or ambient completeness theorem could enter. □

Theorem 6246.3 (Lipschitz completion extension Cauchy handoff)**.** *Let $X = (D, M, L, S, Q, E, H, C, P, N)$ be an accepted* LipschitzCompletionExtension$^{\uparrow}$ *carrier. Every Cauchy source stream routed through*

the displayed Lipschitz modulus yields a completion-target Cauchy row only through the finite route

$$D, M, L \longmapsto S, Q \longmapsto E, H, C, P, N.$$

The handoff supplies target Cauchy evidence for completion consumers without exporting a selected limit, quotient equality of completed points, countable choice, a host extension operator, or ambient completeness beyond the displayed completion target seal.

Proof. Start with a Cauchy source stream accepted by the dense source row. The stream is visible to the carrier only through D and the finite windows S, so every source tolerance request is confined to the displayed dense-window route.

The metric row M and Lipschitz modulus row L rescale the source tolerance into the target tolerance. This step uses only the stored rational bound and its finite checks, not a host continuity theorem or a selected target value.

The row Q reads the source and target windows as regular rational traces, and E admits the target-side Cauchy row through the completion boundary. Finally H, C, P, N preserve the same route as transport, replay, provenance, and local naming data. Since no other coordinate is present, any completion-target Cauchy handoff must pass through exactly these rows. □

Falsifiable prediction: Every accepted $\mathsf{LipschitzCompletionExtension}^{\uparrow}$ handoff from a dense source stream to a completion-target Cauchy row displays one dense source row, one metric row, one Lipschitz modulus row, one $\mathsf{StreamName}^{\uparrow}$ window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one completion target seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: completion, completemetric, streamname, regseqrat: no listed sibling supplies the joint finite packet containing dense source rows, metric comparison rows, Lipschitz modulus transport, finite source and target windows, regular rational readback, completion target sealing, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{LipschitzCompletionExtension}^{\uparrow}}$.

Constructive story: A $\mathsf{LipschitzCompletionExtension}^{\uparrow}$ packet is generated from dense source rows, metric comparison rows, Lipschitz modulus rows, StreamName windows, RegSeqRat readback, completion target seals, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LipschitzCompletionExtension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6246.1, Definition 6246.2, and Definition 6246.3 seed the finite route from dense-source Lipschitz data to completion-target Cauchy evidence.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LipschitzCompletionExtension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, host Lipschitz extension theorems, selected target limits, quotient equality of completed points, countable choice, or ambient completeness outside the displayed completion target seal.

Upgrade path: Obligation closure requires checked carrier admission, Lipschitz-modulus transport, dense-window handoff, completion target sealing, Cauchy handoff replay, nonescape from selected limits or host completeness, transport, replay, provenance, and local naming for BEDC.Derived.LipschitzCompletionExtensionUp.

Chapter 6247

A Concrete Naming Certificate for OrdinalAnalysis$^{\uparrow}$

Following Gentzen and proof-theoretic ordinal analysis, the OrdinalAnalysis$^{\uparrow}$ packet records the finite ordinal-notation and descent-budget surface that a proof-normalization consumer may read before it reaches the MetaCIC normalization frontier. It sits beside the decidability boundary of chapter 749, the normalization frontier of chapter 892, the natural-deduction proof-tree surface of chapter 3756, the first-order syntax carrier of chapter 197, and the lambda-calculus proof-term boundary of chapter 200. The packet names only notation, descent, trace, consumer, refusal, transport, replay, provenance, and local naming rows; it does not assert global normalization, a host proof ordinal, or an unrestricted consistency theorem.

Definition 6247.1 (Ordinal analysis carrier)**.** *An* OrdinalAnalysis$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (A, B, C, U, L, H, R, P, N)$$

with the following displayed rows:

1. *A is the ordinal-notation row, read as a finite notation code and comparison window rather than as a completed host ordinal.*
2. *B is the descent-budget row, recording the finite measure used by a cut-elimination or normalization trace.*
3. *C is the cut-elimination trace row, displaying the local reduction or cut-reduction steps whose budget is being consumed.*
4. *U is the normalization-consumer row, naming the* MetaCICNormalizationFrontier$^{\uparrow}$*,* NaturalDeduction$^{\uparrow}$*,* FirstOrder$^{\uparrow}$*, or* LambdaCalc$^{\uparrow}$ *reader that may inspect the trace.*
5. *L is the refusal ledger, recording absent global-normalization, host-ordinal, subject-reduction, confluence, and consistency coordinates.*
6. *H records componentwise* hsame *transport for the notation, budget, trace, consumer, refusal, replay, provenance, and local naming rows.*
7. *R records displayed* Cont *replay from ordinal notation and descent budget through the trace into the named consumer boundary.*
8. *P is the* Pkg *provenance over* OrdinalAnalysis$^{\uparrow}$*,* MetaCICDecidableBoundary$^{\uparrow}$*,* MetaCICNormalizationFrontier$^{\uparrow}$*,* NaturalDeduction$^{\uparrow}$*,* FirstOrder$^{\uparrow}$*,* LambdaCalc$^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{OrdinalAnalysis}^{\uparrow}}$ *row.*
The carrier has no coordinate for an ambient well-order theorem, a completed proof ordinal, a host consistency proof, a blanket cut-elimination theorem, or global MetaCIC normalization.

Theorem 6247.2 (Ordinal analysis NameCert obligations)**.** *For every accepted* $\mathsf{OrdinalAnalysis}^{\uparrow}$ *carrier* $O = (A, B, C, U, L, H, R, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OrdinalAnalysis}^{\uparrow}}$ *surface consists exactly of carrier admission for ordinal notation, descent budget, cut-elimination trace, normalization consumer, refusal ledger, transport, replay, provenance, and local naming; descent exactness requiring the trace row C to consume only the displayed budget B over the notation row A; consumer compatibility requiring every* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$*,* $\mathsf{NaturalDeduction}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{LambdaCalc}^{\uparrow}$ *read to pass through U; and nonescape excluding host proof ordinals, global normalization, unchecked subject reduction, unchecked confluence, and external consistency claims.*

Proof. Project the carrier of Definition 6247.1. The displayed rows are exactly $A, B, C, U, L, H, R, P, N$, so the local certificate has no source outside this finite packet.

Descent exactness is read from A, B, C. The notation row supplies the comparison window, the budget row supplies the finite descent measure, and the trace row records only the reduction steps charged to that budget.

Consumer compatibility is read from U, R, P, N. Replay and provenance make the named MetaCIC, natural-deduction, first-order, and lambda-calculus readers consume the displayed route rather than a detached host proof.

Finally, the refusal ledger L and the absence of further carrier coordinates exclude ambient proof ordinals, unrestricted normalization, subject reduction, confluence, and consistency claims. Thus the listed rows exhaust the NameCert obligations. □

Lean4 ✓ `BEDC.Derived.OrdinalAnalysisUp.OrdinalAnalysisCarrier namecert obligations`

Theorem 6247.3 (Cut-elimination budget boundary)**.** *Every cut-elimination or normalization-facing read of an accepted* $\mathsf{OrdinalAnalysis}^{\uparrow}$ *carrier must expose the route*

$$A, B, C, U, L, H, R, P, N$$

before it can be consumed by $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$*,* $\mathsf{MetaCICDecidableBoundary}^{\uparrow}$*,* $\mathsf{NaturalDeduction}^{\uparrow}$*,* $\mathsf{FirstOrder}^{\uparrow}$*, or* $\mathsf{LambdaCalc}^{\uparrow}$*. The budget row B is a finite BEDC descent ledger attached to the notation row A and trace row C; it is not a host proof ordinal and does not by itself close global normalization.*

Proof. Read the analytic rows first. The notation row A and descent-budget row B precede the trace row C, so any cut-elimination consumer receives a finite charged trace rather than an unbounded ordinal argument.

Next read the consumer row U. It records which sibling boundary is allowed to inspect the trace: the MetaCIC decidability and normalization frontiers, the natural-deduction proof-tree packet, the first-order syntax carrier, or the lambda-calculus proof-term carrier.

The support rows H, R, P, N transport, replay, source, and name the same route. Since L records the refused readings and no carrier coordinate supplies a completed proof ordinal or global normalization theorem, the budget remains a boundary object for downstream consumers. □

Falsifiable prediction: A packet claiming $\mathsf{OrdinalAnalysis}^{\uparrow}$ acceptance while omitting ordinal notation, descent budget, cut-elimination trace, normalization consumer, refusal ledger, transport, replay, provenance, or local NameCert rows fails local certification before any MetaCIC normalization-frontier read can begin.

Independence witness: metacicnormalizationfrontier, metacicdecidableboundary, natural-deduction, firstorder, lambdacalc: $\mathsf{OrdinalAnalysis}^{\uparrow}$ is not bijective to any listed sibling because it carries an ordinal-notation row, a descent-budget row, a cut-elimination trace row, a named normalization-consumer row, and a refusal ledger as one finite packet. The MetaCIC frontier chapters carry bounded normalization and decidability surfaces without a proof-theoretic descent ledger; NaturalDeductionUp carries proof trees without ordinal notation; FirstOrderUp carries syntax and deduction endpoints; LambdaCalcUp carries term reduction rows without the displayed ordinal-analysis budget.

Closure status for $\mathsf{NameCert}_{\mathsf{OrdinalAnalysis}^{\uparrow}}$.

Constructive story: An $\mathsf{OrdinalAnalysis}^{\uparrow}$ packet is generated from ordinal notation, descent budget, cut-elimination trace, normalization-consumer, refusal-ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrdinalAnalysis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6247.1, Definition 6247.2, and Definition 6247.3 seed the finite proof-theoretic descent route consumed by MetaCIC and proof-calculus boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrdinalAnalysis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, global normalization, host proof ordinals, unrestricted cut elimination, subject reduction, confluence, or consistency for $\mathsf{OrdinalAnalysis}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, descent-budget exactness, cut-elimination trace replay, normalization-consumer compatibility, refusal-ledger nonescape, transport, replay, provenance, and local naming for BEDC.Derived.OrdinalAnalysisUp.

Chapter 6248

A Concrete Naming Certificate for $\mathsf{CartanMatrix}^{\uparrow}$

Following Bourbaki and Humphreys, the $\mathsf{CartanMatrix}^{\uparrow}$ packet records the finite integer matrix row that sits between root-system pairings and Weyl or Coxeter consumers. It is read after the root-system surface of chapter 144, the Weyl-group boundary of chapter 145, the Coxeter-group relation surface of chapter 6245, the finite matrix rows of chapter 49, and the integer endpoint rows of chapter 31. The packet names the Cartan-pairing table as finite NameCert data; it does not import Lie-algebra classification, root-string completeness, ambient vector-space equality, or a host crystallographic oracle.

Definition 6248.1 (Cartan matrix carrier)**.** *A* $\mathsf{CartanMatrix}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (I, A, D, O, S, R, W, C, H, K, P, N)$$

with the following displayed rows:

1. *I is the finite index row, naming the simple-root positions whose pairings are recorded.*
2. *A is the integer matrix-entry row, read through* $\mathsf{Mat}^{\uparrow}$ *and* $\mathsf{Int}^{\uparrow}$*, with an entry a_{ij} for every displayed pair of indices.*
3. *D is the diagonal-two row, recording $a_{ii} \sim_{\mathsf{Int}^{\uparrow}} 2$ for each displayed index i.*
4. *O is the nonpositive off-diagonal row, recording the inspected integer sign evidence for a_{ij} when i and j are distinct.*
5. *S is the symmetrizability or finite-type sign row exposed by the carrier when such a row is claimed by the root-system source.*
6. *R is the* $\mathsf{RootSystem}^{\uparrow}$ *pairing provenance row, tying each a_{ij} to the displayed coroot-pairing or reflection-pairing read.*
7. *W is the* $\mathsf{WeylGroup}^{\uparrow}$ *and* $\mathsf{CoxeterGroup}^{\uparrow}$ *consumer row, recording exactly which reflection or braid reads may consume I, A, D, O, S, R.*
8. *C records componentwise* hsame *transport for the index, entry, diagonal, off-diagonal, sign, pairing, consumer, replay, provenance, and naming rows.*
9. *H records displayed* Cont *replay for matrix-entry lookup, diagonal readback, off-diagonal sign readback, root-system pairing transport, and Weyl or Coxeter consumer handoff.*

10. *K is the package-local replay ledger that keeps every row of A attached to the same index row I.*
11. *P is the* Pkg *provenance over* $\mathsf{CartanMatrix}^{\uparrow}$, $\mathsf{RootSystem}^{\uparrow}$, $\mathsf{WeylGroup}^{\uparrow}$, $\mathsf{CoxeterGroup}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{Int}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{CartanMatrix}^{\uparrow}}$ *row.*

The classifier compares two Cartan-matrix packets only by these displayed rows and their transports. It carries no host matrix equality, no hidden root datum, and no classification of finite type beyond the row S explicitly supplied in the packet.

Lean4 ✓ `BEDC.Derived.CartanMatrixUp.CartanMatrixTasteGate single carrier alignment`

Theorem 6248.2 (Cartan matrix NameCert obligations)**.** *For every accepted* $\mathsf{CartanMatrix}^{\uparrow}$ *carrier $M = (I, A, D, O, S, R, W, C, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CartanMatrix}^{\uparrow}}$ *surface consists exactly of carrier admission for the finite index and integer matrix rows, diagonal-two exactness, off-diagonal sign exposure, the optional symmetrizability or finite-type sign row, root-system pairing provenance, Weyl and Coxeter consumer exposure, componentwise transport, displayed replay, package-local row coherence, provenance, and local naming.*

Proof. Project the carrier of Definition 6248.1. The rows I, A, D, O, S supply the finite matrix data, while R, W state how that data is sourced and consumed.

Carrier admission reads I and A through the finite matrix and integer surfaces. Diagonal exactness is the displayed row D, and off-diagonal sign exposure is the displayed row O.

The source and consumer boundaries are then forced by R and W. The row R ties entries to root-system pairings, while W records the only Weyl or Coxeter reads that may consume the displayed Cartan table.

Finally, C, H, K, P, N preserve and name the same packet. Transport, replay, package-local row coherence, provenance, and local naming add no hidden root datum, host matrix equality, finite-type classification theorem, or ambient Lie-algebra object. □

Theorem 6248.3 (Cartan matrix root-system consumer)**.** *Every* $\mathsf{RootSystem}^{\uparrow}$ *consumer that exports a Cartan row through* $\mathsf{CartanMatrix}^{\uparrow}$ *must first expose the finite index row I, the integer entry row A, the diagonal-two row D, the off-diagonal sign row O, and the root-system pairing provenance row R. The exported table cannot be replaced by a prose pairing convention, a host bilinear form, or an unindexed family of integers.*

Proof. Start from an accepted carrier $M = (I, A, D, O, S, R, W, C, H, K, P, N)$. The carrier lists the Cartan table as finite entries over the displayed index row, so no entry can be read without first reading I and A.

The Cartan-specific obligations are the diagonal and off-diagonal rows. The row D supplies the diagonal value, and O supplies the sign evidence for distinct indices.

The root-system origin of the table is not inferred from the matrix shape. It is the row R, which ties each entry to the displayed $\mathsf{RootSystem}^{\uparrow}$ pairing route.

Thus a root-system consumer receives a finite indexed table with visible integer entries and pairing provenance. The refused replacements are absent from the carrier and cannot be introduced by the transport or replay rows. □

Theorem 6248.4 (Cartan matrix Weyl-Coxeter boundary)**.** *Every* $\mathsf{WeylGroup}^{\uparrow}$ *or* $\mathsf{CoxeterGroup}^{\uparrow}$ *read of an accepted* $\mathsf{CartanMatrix}^{\uparrow}$ *packet factors through the same finite route*

$$I, A, D, O, S, R, W, C, H, K, P, N.$$

The consumer may use only the reflection-pairing, sign, and braid-facing rows recorded by R, S, W and preserved by C, H, K. It cannot read an unrecorded Coxeter matrix, a host reflection representation, a root-system classification theorem, or a detached Weyl action from the Cartan packet alone.

Proof. Read the consumer row W from Definition 6248.1. It is explicitly the boundary for Weyl and Coxeter consumers and is attached to the same I, A, D, O, S, R rows as the Cartan table.

The sign row S and pairing row R are the only rows that can feed reflection or braid-facing interpretations. If a consumer needs a Coxeter relation, it must obtain that relation through the displayed consumer route rather than by inventing a separate matrix.

The transport, replay, and package-local coherence rows C, H, K preserve the same indexed table under accepted comparisons. They do not add a representation space, a classification theorem, or an independent Weyl action.

Therefore the public Weyl-Coxeter boundary of the Cartan packet is exactly the finite route displayed above, with the listed external objects outside the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CartanMatrix}^{\uparrow}}$.

Constructive story: A $\mathsf{CartanMatrix}^{\uparrow}$ packet is generated from a finite index row, integer matrix entries, diagonal-two rows, off-diagonal sign rows, an optional symmetrizability or finite-type sign row, RootSystem pairing provenance, Weyl and Coxeter consumer rows, componentwise hsame transport, displayed Cont replay, package-local row coherence, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CartanMatrix}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6248.1, Definition 6248.2, Definition 6248.3, and Definition 6248.4 seed the finite Cartan matrix carrier, root-system source, and Weyl-Coxeter consumer surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CartanMatrix}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, Lie-algebra classification, root-string completeness, a host root datum, ambient vector-space equality, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, diagonal exactness, off-diagonal sign exposure, root-system pairing provenance, Weyl-Coxeter consumer routing, transport, replay, package coherence, provenance, and local naming for BEDC.Derived.CartanMatrixUp.

Chapter 6249

A Concrete Naming Certificate for $\mathsf{ScopedExplanationLedger}^{\uparrow}$

The $\mathsf{ScopedExplanationLedger}^{\uparrow}$ packet realizes the scoped black-box certification principle of section 9462.19 as finite NameCert data. It records an explanation scope, the classifier rows valid inside that scope, the behavior ledger they cover, the global-ledger refusal row for uncovered behavior families, transport, replay, provenance, and local naming. The packet is an interpretability boundary: it certifies scoped explanation rows without pretending that a single classifier or natural-language summary eliminates global behavior residue.

The scoped ledger consumes relation rows from chapter 1089 and certificate compilation rows from chapter 473 before it records behavior-family residue. Those sibling surfaces supply the relation-site and compiled-certificate inputs that the scoped classifier reads; the ledger row keeps the uncovered behavior families visible instead of converting compilation into a global explanation claim.

Definition 6249.1 (Scoped explanation ledger carrier)**.** *A* $\mathsf{ScopedExplanationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (\Omega, B, E, Q, G, F, T, R, P, N)$$

with the following displayed rows:

1. *Ω is the explanation-scope row, naming the source domain, model state, admitted transformation family, and behavior being explained.*
2. *B is the behavior-family boundary row, separating the scoped behavior from behavior rows not covered by the packet.*
3. *E is the explanation-object row, containing the finite classifier, local NameCert, and scoped ledger entries used for the certified behavior.*
4. *Q is the scoped classifier row, recording exactly which comparisons or behavioral equivalences are valid inside Ω.*
5. *G is the global-ledger refusal row, recording that no global behavior-family ledger is exported unless a separate global carrier is supplied.*
6. *F is the failure and not-claimed row, listing uncovered behaviors, unverified transfers, semantic residue, and out-of-scope transformations.*
7. *T records componentwise* hsame *transport for $\Omega, B, E, Q, G, F, R, P, N$.*

8. *R records displayed* Cont *replay for scoped explanation reads, classifier reads, refusal reads, and downstream audit consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{ScopedExplanationLedger}^{\uparrow}$*, the scoped certification principle of section 9462.19,* BHist*,* hsame*,* Cont*,* Pkg*, and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ScopedExplanationLedger}^{\uparrow}}$ *row.*

The classifier compares packets only by the displayed scope, behavior boundary, explanation object, classifier, refusal, failure, transport, replay, provenance, and naming rows. It has no row that turns scoped resolution into global resolution.

Theorem 6249.2 (Scoped explanation ledger NameCert obligations)**.** *For every accepted* ScopedExplanationLedger *carrier $L = (\Omega, B, E, Q, G, F, T, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ScopedExplanationLedger}^{\uparrow}}$ *surface consists exactly of explanation-scope admission, behavior-boundary exposure, explanation-object exposure, scoped classifier exactness, global-ledger refusal, not-claimed residue, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6249.1. The rows Ω, B, E, Q state the scoped explanation and its classifier, while G, F state what is not exported.

The carrier admits a packet only when the scope and behavior boundary are displayed. The explanation object E and classifier row Q are therefore read inside that scope rather than as global model facts.

The refusal and residue rows are equally part of the certificate. The row G denies a global-ledger export, and F lists the behavior, transfer, semantic, and verification residue left outside the scoped claim.

Transport, replay, provenance, and naming preserve this same boundary. They do not create a global behavior-family ledger or a universal classifier, so the NameCert obligations are exactly the listed rows. □

Theorem 6249.3 (Scoped explanation ledger global nonescape)**.** *An accepted* $\mathsf{ScopedExplanationLedger}^{\uparrow}$ *packet never exports global black-box resolution from the scoped explanation row alone. Any consumer claiming global behavior coverage must read a separate global-ledger carrier; the rows $\Omega, B, E, Q, G, F, T, R, P, N$ only certify the scoped explanation and the explicit global-ledger refusal.*

Proof. Begin with the rows of Definition 6249.1. The explanation object E and classifier row Q are indexed by the scope row Ω and behavior boundary B.

The global-ledger row of this packet is a refusal row. It records that no global behavior-family ledger is exported from the scoped explanation.

The residue row F names the uncovered behavior, transfer, semantic, and verification residue. A consumer that ignores F has not consumed the certificate boundary of the packet.

Thus global behavior coverage is unavailable from this carrier. It requires a separate carrier whose rows actually display the covered behavior family and common global ledger. □

Theorem 6249.4 (Scoped explanation ledger cannot-claim dependency)**.** *The global-ledger refusal row G of an accepted* $\mathsf{ScopedExplanationLedger}^{\uparrow}$ *packet is consumed as refusal/export gating through chapter 1319. A consumer may export the scoped explanation rows Ω, B, E, Q only as scoped rows; any claim that removes the global-ledger refusal must present a separate global carrier and pass the cannot-claim export gate before it can be named as an exported target.*

Proof. Start with Definition 6249.3. The row G is not decorative metadata: it is the displayed refusal that blocks global behavior coverage from being read out of the scoped classifier.

The cannot-claim export gate supplies the sibling discipline for that refusal. Its registry, refusal, export-decision, target, audit, transport, continuation, provenance, and local naming rows force a refused claim to remain visible before any target export is named.

Transport the scoped explanation packet through T, R, P, N. These structural rows preserve the same refusal boundary, so a consumer can export the packet only as a scoped explanation ledger unless a separate global carrier discharges the refused coverage claim through the gate. □

Theorem 6249.5 (Scoped explanation ledger sibling independence)**.** *The* $\mathsf{ScopedExplanationLedger}^{\uparrow}$ *carrier is not bijective with any single classifier packet, relation packet, compiler packet, no-global-synchronization ledger, or machine-interface socket. Its public route requires the combined scope, behavior boundary, explanation object, scoped classifier, global-ledger refusal, residue, transport, replay, provenance, and local naming rows.*

Proof. A single classifier packet can supply a comparison row, but it lacks the explanation scope, behavior-family boundary, global-ledger refusal, and not-claimed residue required by Definition 6249.1.

A relation or compiler packet can expose relation or compilation rows, but those rows do not determine which behavior is certified inside which explanation scope.

A no-global-synchronization ledger or machine-interface socket can expose refusal or boundary rows, but it lacks the scoped explanation object and classifier exactness row.

Therefore none of those siblings is bijective with the scoped explanation ledger. The carrier is the combined finite route $\Omega, B, E, Q, G, F, T, R, P, N$. □

Falsifiable prediction: If a $\mathsf{ScopedExplanationLedger}^{\uparrow}$ packet is reduced to a single classifier row, then the global-ledger refusal row is lost.

Independence witness: layeredrelationsite, certificatecompiler, no_global_synchronization_ledger, machineinterfacesocket: the relation site supplies relation rows without the explanation-scope ledger; the certificate compiler supplies compilation rows without behavior-family residue; the no-global-synchronization ledger supplies refusal rows without scoped classifier exactness; the machine-interface socket supplies boundary access without the local explanation object. None is bijective with the combined scope, classifier, ledger, refusal, transport, replay, provenance, and NameCert route.

Closure status for $\mathsf{NameCert}_{\mathsf{ScopedExplanationLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{ScopedExplanationLedger}^{\uparrow}$ packet is generated from an explanation scope, behavior-family boundary, explanation object, scoped classifier, global-ledger refusal, not-claimed residue, componentwise hsame transport, displayed Cont replay, package provenance, and local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ScopedExplanationLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6249.1, Definition 6249.2, Definition 6249.3, Definition 6249.4, and Definition 6249.5 seed the scoped explanation, classifier, refusal, export-gating dependency, residue, transport, replay, provenance, and local naming surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ScopedExplanationLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, global black-box resolution, a universal classifier, global synchronization, open-domain behavior coverage, or a standard bridge.

Upgrade path: Obligation closure requires checked scope admission, behavior-boundary exposure, scoped classifier exactness, global-ledger refusal, residue, transport, replay, provenance, and local naming for BEDC.Derived.ScopedExplanationLedgerUp.

Chapter 6250

A Concrete Naming Certificate for $\mathsf{CompactCoverModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompactCoverModulus}^{\uparrow}$ packet records the finite-cover modulus route that turns compact-source coverage and local continuity radii into a uniform-modulus handoff. It sits between the compact metric source of chapter 129, the continuous-map rows of chapter 130, the finite lower-bound fold of chapter 4096, and the uniform-modulus consumer of chapter 369. The packet names finite cover rows and lower-bound folds as BEDC histories; it does not choose an external modulus function, appeal to open-cover compactness, or import ambient real completeness.

Definition 6250.1 (Compact cover modulus carrier)**.** *A* $\mathsf{CompactCoverModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, C, L, F, R, H, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, read through* $\mathsf{CompactMetric}^{\uparrow}$*.*
2. *C is the finite cover row, displaying the compact finite subcover or* $\mathsf{ProbeBundle}$ *spine used by the modulus construction.*
3. *L is the local modulus list row, one positive-radius ledger for each displayed cover member and continuous-map observation.*
4. *F is the finite lower-bound fold row, recording the certified positive lower bound consumed by* $\mathsf{UniformModulus}^{\uparrow}$*.*
5. *R is the uniform radius row exported to the downstream uniform-modulus consumer.*
6. *H records componentwise* hsame *transport for the compact source, cover, local moduli, fold, radius, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{CompactCoverModulus}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ $\mathsf{UniformModulus}^{\uparrow}$*,* $\mathsf{ProbeBundle}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCoverModulus}^{\uparrow}}$ *row.*

The carrier has no coordinate for an external choice of cover, a host finite minimum, a pointwise-only modulus selector, quotient equality of metric points, or ambient compactness outside the displayed source row.

Lean4 ✓ `BEDC.Derived.CompactCoverModulusUp.TasteGate.CompactCoverModulusTasteGate single carrier alignment`

Theorem 6250.2 (Compact cover modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactCoverModulus}^{\uparrow}$ *carrier* $M = (K, C, L, F, R, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactCoverModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for compact metric source, finite cover, local modulus list, finite lower-bound fold, uniform radius, transport, provenance, and local naming; cover-source compatibility requiring* C *to be a cover of the displayed compact metric row* K*; fold exactness requiring* F *to read only the finite local-modulus list* L *over that cover; uniform-radius handoff requiring* R *to be exported only after the fold row is displayed; and nonescape excluding external cover choice, host finite minima, pointwise-only selector data, quotient point equality, and ambient completeness.*

Proof. Project the carrier of Definition 6250.1. The packet displays exactly K, C, L, F, R, H, P, N, so every local obligation must be read from these rows.

Cover-source compatibility is the route from K to C. The cover row is attached to the compact metric source row and cannot be interpreted over a detached source.

Fold exactness is the route from L to F and then to R. The finite lower-bound fold consumes only the displayed local modulus list, and the uniform radius is exported only after that fold is present.

The support rows H, P, N transport, source, and name the same carrier. Since no row stores external cover choice, a host finite minimum, a pointwise-only selector, quotient point equality, or ambient completeness, the listed obligations exhaust the NameCert surface. □

Theorem 6250.3 (Finite subcover handoff)**.** *Every compact-uniform-continuity consumer reading an accepted* $\mathsf{CompactCoverModulus}^{\uparrow}$ *carrier must pass through the ordered route*

$$K, C, L, F, R, H, P, N.$$

The finite cover row C *and local modulus list* L *are read before the finite lower-bound fold* F*, and the uniform radius row* R *is available only after that fold. Thus the consumer receives a concrete finite-cover modulus packet rather than a narrative compactness argument or an external uniform-modulus selector.*

Proof. Read the compact source row K and finite cover row C. These rows expose the source and the finite subcover or $\mathsf{ProbeBundle}$ spine used by the compact-uniform-continuity route.

Next read the local modulus list L. Each entry is attached to a displayed cover member and a continuous-map observation, so the list is finite before the fold begins.

Finally apply the fold and handoff rows F, R. Transport, provenance, and naming through H, P, N preserve this route, while the carrier excludes external selectors, host minima, quotient point equality, and ambient completeness. The downstream $\mathsf{UniformModulus}^{\uparrow}$ consumer therefore reads the displayed finite-subcover handoff. □

Falsifiable prediction: A compact-uniform-continuity packet that claims $\mathsf{CompactCoverModulus}^{\uparrow}$ acceptance while omitting compact source, finite cover, local modulus list, finite lower-bound fold, uniform radius, transport, provenance, or local NameCert rows fails local certification before a $\mathsf{UniformModulus}^{\uparrow}$ consumer may read it.

Independence witness: compactmetric, continuousmap, finite-lower-bound-fold, uniformmodulus: $\mathsf{CompactCoverModulus}^{\uparrow}$ is not bijective to any listed sibling because it carries the compact source, finite cover, local modulus list, finite lower-bound fold, and uniform-radius handoff as

one packet. CompactMetricUp supplies the source without the continuous local modulus list; ContinuousMapUp supplies graph and local continuity rows without the finite compact cover; FiniteLowerBoundFoldUp supplies the fold without the source-cover handoff; UniformModulusUp consumes the radius without naming the finite cover that produced it.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCoverModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCoverModulus}^{\uparrow}$ packet is generated from compact metric source, finite cover, local modulus list, finite lower-bound fold, uniform radius, componentwise hsame transport, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCoverModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6250.1, Definition 6250.2, and Definition 6250.3 seed the finite-cover modulus handoff for compact uniform continuity.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCoverModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, external modulus choice, host finite minima, quotient point equality, or ambient completeness for $\mathsf{CompactCoverModulus}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, cover-source compatibility, local-modulus list admission, finite lower-bound fold exactness, uniform-radius handoff, transport, provenance, and local naming for BEDC.Derived.CompactCoverModulusUp.

Chapter 6251

A Concrete Naming Certificate for $\mathsf{RealReciprocal}^{\uparrow}$

Following Bishop and Bridges on constructive real arithmetic, the $\mathsf{RealReciprocal}^{\uparrow}$ packet names the guarded reciprocal surface for displayed nonzero-apart $\mathsf{Real}^{\uparrow}$ data. It sits after the real-name seal of chapter 38, the regular rational readback of chapter 301, the dyadic tolerance and lower-bound rows of chapter 291, the reciprocal-window budget of chapter 703, the guarded quotient route of chapter 1943, and the Heron consumer of chapter 3818. The chapter records reciprocal reads only under displayed apartness; it is not trichotomy, quotient real equality, a total inverse function, or ambient $\mathsf{Real}^{\uparrow}$ field completeness.

Definition 6251.1 (Real reciprocal carrier)**.** *A* $\mathsf{RealReciprocal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (X, A, B, Q, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the accepted* $\mathsf{Real}^{\uparrow}$ *source whose reciprocal is requested.*
2. *A is the nonzero-apartness row, recording the displayed lower-bound witness that keeps X away from the zero boundary.*
3. *B is the* $\mathsf{CauchyInverseBudget}^{\uparrow}$ *row that selects the finite reciprocal-window budget for the source name and apartness witness.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row produced by the budgeted reciprocal route.*
5. *D is the* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *handoff row, exposing the same reciprocal as division of the unit regular name by the guarded source name.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the reciprocal candidate.*
7. *H records componentwise* hsame *transport for the source, apartness, budget, readback, division handoff, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for source inspection, apartness exposure, budget selection, regular readback, division handoff, and Real sealing.*
9. *P is the* Pkg *provenance over* $\mathsf{RealReciprocal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyInverseBudget}^{\uparrow}$, $\mathsf{RegularCauchyDivision}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RealReciprocal}^{\uparrow}}$ *row.*

The local classifier compares accepted carriers only through $X, A, B, Q, D, E, H, C, P, N$ and has no coordinate for total inversion, trichotomy, quotient real equality, or an ambient field inverse.

Lean4 ✓ `BEDC.Derived.RealReciprocalUp.RealReciprocalTasteGate single carrier alignment`

Theorem 6251.2 (Real reciprocal NameCert obligations)**.** *For every accepted* $\mathsf{RealReciprocal}^{\uparrow}$ *carrier*

$$R = (X, A, B, Q, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealReciprocal}^{\uparrow}}$ *surface consists exactly of source Real admission, nonzero-apartness exposure, finite Cauchy inverse budget selection,* $\mathsf{RegSeqRat}^{\uparrow}$ *reciprocal readback,* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *handoff, terminal* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming for the same finite packet.*

Lean4 ✓ `BEDC.Derived.RealReciprocalUp.RealReciprocalCarrier namecert obligation surface`

Proof. Project the carrier of Definition 6251.1. The rows $X, A, B, Q, D, E, H, C, P, N$ list all analytic and structural data available to an accepted reciprocal packet.

The analytic route is ordered. The apartness row A is exposed before the inverse budget B, the budget produces the regular rational readback Q, and the division handoff D identifies the same guarded reciprocal route before the terminal Real seal E is read.

The rows H, C, P, N then transport, replay, source, and name the displayed route. Since no additional coordinate is present, the local NameCert obligations are exhausted by the listed packet and cannot import total inversion, trichotomy, quotient real equality, or ambient field completeness. □

Theorem 6251.3 (Real reciprocal apartness-domain stability)**.** *Let* $R = (X, A, B, Q, D, E, H, C, P, N)$ *and* $R' = (X', A', B', Q', D', E', H', C', P', N')$ *be accepted* $\mathsf{RealReciprocal}^{\uparrow}$ *packets. If the displayed transport rows relate* X, A, B, Q, D, E *to* X', A', B', Q', D', E' *componentwise by* hsame*, then the reciprocal route for* R' *still exposes* A' *before any budget, readback, division handoff, or Real seal row is consumed. A route with no transported apartness row remains outside the reciprocal domain.*

Lean4 ✓ `BEDC.Derived.RealReciprocalUp.RealReciprocalCarrier apartness domain stability`

Proof. Start with the NameCert obligations for R. They place A before B, Q, D, E in the displayed reciprocal route.

Use the componentwise hsame rows to transport the source, apartness, budget, readback, handoff, and seal coordinates to R'. The transported route has the same row order because hsame changes only the displayed coordinate representatives, not the carrier inventory.

It follows that A' is still the first domain row for the reciprocal read. If the transported route omits A', then no row can justify the Cauchy inverse budget or the later Real seal, so the route remains a boundary attempt rather than an accepted reciprocal packet. □

Theorem 6251.4 (Real reciprocal regular-Cauchy handoff)**.** *Every accepted* $\mathsf{RealReciprocal}^{\uparrow}$ *read factors through*

$$X, A \longmapsto B \longmapsto Q \longmapsto D \longmapsto E.$$

The handoff reads the reciprocal candidate from $\mathsf{CauchyInverseBudget}^{\uparrow}$ *through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{RegularCauchyDivision}^{\uparrow}$ *before the terminal* $\mathsf{Real}^{\uparrow}$ *seal is public. It exports no host inverse function, no total reciprocal operation, no trichotomy principle, no quotient real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness witness.*

Lean4 ✓ `BEDC.Derived.RealReciprocalUp.RealReciprocalCarrier regular cauchy handoff`

Proof. Apply Definition 6251.2. It exposes the source and apartness rows X, A before any reciprocal-window data is available.

Next read the Cauchy inverse budget row B. This row is the finite budget surface supplied by chapter 703; it selects only displayed windows and lower-bound data.

The regular rational handoff Q and guarded division row D are downstream of that budget. The division row agrees with the reciprocal route through chapter 1943, so the seal E receives a regular-Cauchy reciprocal packet rather than a host inverse.

Finally inspect the carrier inventory. Since the packet contains only $X, A, B, Q, D, E, H, C, P, N$, there is no coordinate for total reciprocal choice, trichotomy, quotient real equality, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{RealReciprocal}^{\uparrow}$ read exposes a nonzero-apartness row before the Cauchy inverse budget, RegSeqRat readback, RegularCauchyDivision handoff, and terminal Real seal.

Independence witness: real, regseqrat, dyadicratcore, cauchyinversebudget, regularcauchydivision: $\mathsf{RealReciprocal}^{\uparrow}$ is not bijective to any listed sibling because it binds a Real source, apartness-domain row, inverse budget, regular readback, guarded division handoff, Real seal, transport, replay, provenance, and local naming as one finite reciprocal packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealReciprocal}^{\uparrow}}$.

Constructive story: A $\mathsf{RealReciprocal}^{\uparrow}$ packet is generated from a displayed Real source, a nonzero-apartness row, a Cauchy inverse budget, RegSeqRat reciprocal readback, RegularCauchyDivision handoff, terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealReciprocal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6251.1, Definition 6251.2, Definition 6251.3, and Definition 6251.4 seed the guarded reciprocal boundary over nonzero-apart Real data, finite Cauchy inverse budgets, RegSeqRat readback, guarded division handoff, and Real sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealReciprocal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close total reciprocal choice, trichotomy, quotient real equality, selected representatives, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, apartness-domain stability, finite inverse-budget readback, RegularCauchyDivision handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealReciprocalUp.

Lean4 (statement) `BEDC.Derived.RealReciprocalUp`

Chapter 6252

A Concrete Naming Certificate for $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$

Following Newton, Leibniz, and Bishop–Bridges constructive analysis, the $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ packet records the finite derivative-integral handoff by which a displayed derivative window and a displayed Riemann integral row are read as one theorem-facing route. It sits after the derivative carrier of chapter 122, the Riemann-integral row of chapter 1526, the Darboux comparison surface of chapter 1880, the tagged-partition schedule of chapter 4296, the regular rational readback of chapter 301, the finite stream windows of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38. The certificate names only the displayed constructive route from derivative control to integral readback; it does not import a classical continuum theorem, select an antiderivative, identify quotient functions, or assert ambient completeness.

Definition 6252.1 (Fundamental theorem calculus carrier)**.** *A* $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, I, A, T, Q, S, G, R, H, C, P, N)$$

with the following rows:

1. *D is the* $\mathsf{Derivative}^{\uparrow}$ *window row, naming the displayed function, interval window, derivative witness, and local difference-quotient budget.*
2. *I is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *row through which the matching integral expression is inspected.*
3. *A is the* $\mathsf{DarbouxSum}^{\uparrow}$ *and refinement ledger, recording upper and lower comparison rows and the mesh refinement relation used by the integral readback.*
4. *T is the* $\mathsf{TaggedRealPartition}^{\uparrow}$ *row, exposing the ordered tags, cells, endpoint windows, and compatibility with the same interval read by D and I.*
5. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance budget for derivative windows, partition mesh, Darboux spread, integral readback, and endpoint comparison.*
6. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for the endpoint values, derivative samples, tagged cells, and integral approximants.*
7. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational sequences exposed by S.*
8. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the Real-facing derivative-integral conclusion.*

9. *H records componentwise* hsame *transport for derivative, integral, Darboux, tagged-partition, dyadic, stream, regular-rational, Real, replay, provenance, and naming rows.*
10. *C records displayed* Cont *replay of the finite route*

$$D, I, A, T, Q \longmapsto S \longmapsto G \longmapsto R \longmapsto \mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}.$$

11. *P is the* Pkg *provenance over* $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{TaggedRealPartition}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, I, A, T, Q, S, G, R, H, C, P, N$. It has no coordinate for ambient Real completeness, measure-theoretic integration, host function extensionality, selected antiderivatives, quotient equality of functions, or bridge checking.

Theorem 6252.2 (Fundamental theorem calculus NameCert obligations)**.** *For every accepted* $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ *carrier*

$$F = (D, I, A, T, Q, S, G, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}$ *surface consists exactly of carrier admission for F, derivative-window exposure through D, Riemann-integral exposure through I, Darboux and refinement accountability through A, tagged-partition visibility through T, dyadic tolerance accounting through Q, StreamName window exposure through S, RegSeqRat readback through G, terminal Real sealing through R, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Lean4 ✓ `BEDC.Derived.FundamentalTheoremCalculusUp.FundamentalTheoremCalculusTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6252.1. The analytic rows D, I, A, T, Q are the only source of derivative control, integral inspection, Darboux comparison, tagged partition data, and tolerance budgets.

The readback rows are ordered. The finite StreamName row S exposes the visible endpoint, derivative, cell, and integral windows before G reads them as regular rational data and R seals the Real-facing conclusion.

The structural rows H, C, P, N transport, replay, source, and name exactly this displayed route. Since no carrier coordinate stores ambient Real completeness, measure-theoretic integration, host function extensionality, selected antiderivatives, quotient equality, or bridge checking, the listed rows exhaust the local NameCert surface. □

Theorem 6252.3 (Fundamental theorem calculus Riemann derivative route)**.** *Every accepted* $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ *theorem read that hands a derivative window to a Riemann-integral conclusion factors through the finite route*

$$\mathsf{Derivative}^{\uparrow} \longmapsto \mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{DarbouxSum}^{\uparrow} \longmapsto \mathsf{TaggedRealPartition}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}$$

as instantiated by D, I, A, T, Q, S, G, R. The Real-facing conclusion receives derivative-window data, Riemann integral data, Darboux and refinement accountability, tagged partition visibility, dyadic tolerance budgets, finite stream windows, regular rational readback, terminal Real sealing,

componentwise hsame *transport, displayed* Cont *replay,* Pkg *provenance, and a local* NameCert *row; it receives no ambient Real completeness theorem, measure-theoretic integration object, host function extensionality principle, selected antiderivative, quotient equality, or checked bridge.*

Proof. Start with the accepted carrier $F = (D, I, A, T, Q, S, G, R, H, C, P, N)$. The rows D and I display the derivative window and the matching Riemann-integral row before any theorem-facing conclusion is read.

Use the partition and comparison route next. The Darboux ledger A, tagged partition row T, and dyadic budget Q expose the finite mesh and tolerance data needed by the integral readback.

Finally pass through the completion-facing rows. The stream, regular-rational, and Real rows S, G, R are read in that order, and H, C, P, N preserve the same displayed path. Since the carrier has no coordinate for ambient completeness, measure-theoretic integration, host extensionality, selected antiderivatives, quotient equality, or bridge checking, the route is exhausted by the finite BEDC packet. □

Falsifiable prediction: Every $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ theorem-facing read factors through one derivative-window row, one Riemann-integral row, one Darboux/refinement ledger, one tagged-partition row, one dyadic tolerance budget, one StreamName finite-window row, one RegSeqRat readback row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: derivative, riemannintegral, darboux-sum, tagged-real-partition, dyadicratcore, streamname, regseqrat, real: $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the derivative window, integral row, Darboux comparison, tagged partition schedule, dyadic tolerance budget, finite observations, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}$.

Constructive story: A $\mathsf{FundamentalTheoremCalculus}^{\uparrow}$ packet is generated from a DerivativeUp window row, a RiemannIntegralUp row, a DarbouxSum refinement ledger, a TaggedRealPartition row, DyadicRatCore tolerance budgets, StreamName windows, RegSeqRat readbacks, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}$, seed)

Scope closed: Definition 6252.1, Definition 6252.2, and Definition 6252.3 seed the finite derivative-integral route over derivative, Riemann-integral, Darboux, tagged-partition, dyadic, stream-name, regular rational, and Real rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FundamentalTheoremCalculus}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.FundamentalTheoremCalculusUp`

Bridge status: none

Not claimed: This chapter does not close ambient $\mathsf{Real}^{\uparrow}$ completeness, measure-theoretic integration, host function extensionality, selected antiderivatives, quotient equality, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-window exposure, Riemann-integral exposure, Darboux/refinement exactness, tagged-partition visibility, dyadic tolerance accounting, readback routing, Real sealing, provenance, and local naming for BEDC.Derived.FundamentalTheoremCalculusUp.

Chapter 6253

A Concrete Naming Certificate for $\mathsf{LocatedIntervalCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ packet records the finite interval-completion surface where endpoint real names, dyadic tolerance requests, finite stream windows, regular rational readback, and Cauchy modulus rows are bound before any completion consumer reads a sealed interval point. It sits after the terminal real-name surface of chapter 38, the regular rational readback of chapter 301, the finite window route of chapter 126, the dyadic tolerance core of chapter 291, and the Cauchy route of chapter 3413. The packet names located interval completion data as finite rows; it does not quotient streams, select representatives, invoke countable choice, use trichotomy, or assert ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6253.1 (Located interval completion carrier)**.** *A* $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (E_0, E_1, D, W, Q, M, R, H, C, P, N)$$

with the following displayed rows:

1. *E_0 and E_1 are the lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint rows. They are read as terminal real-name seals, not as host real representatives.*
2. *D is the positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the interval-width budget, endpoint tolerance budget, and halving ledger used by completion reads.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row opened at the endpoint indices and tolerances displayed by E_0, E_1, D.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows and dyadic tolerance data.*
5. *M is the Cauchy modulus row. It records the finite schedule that makes the endpoint window reads Cauchy-compatible inside the located interval route.*
6. *R is the completion seal row, consumed only after the endpoint rows, dyadic tolerance row, StreamName windows, RegSeqRat readback, and Cauchy modulus row have been displayed.*
7. *H records componentwise* hsame *transport for E_0, E_1, D, W, Q, M, R.*
8. *C records displayed* Cont *replay for interval-completion requests through the finite route*

$$E_0, E_1, D \longmapsto W, Q, M \longmapsto R, H, P, N.$$

9. *P is the* Pkg *provenance over* $\mathsf{LocatedIntervalCompletion}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Cauchy}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $E_0, E_1, D, W, Q, M, R, H, C, P, N$. It has no coordinate for quotient stream equality, selected limits, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.LocatedIntervalCompletionUp.LocatedIntervalCompletionTasteGate single carrier alignment`

Theorem 6253.2 (Located interval completion NameCert obligations)**.** *For every accepted* LocatedIntervalComplet *carrier $L = (E_0, E_1, D, W, Q, M, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompletion}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *endpoint admission for the two* $\mathsf{Real}^{\uparrow}$ *rows and their displayed interval order context;*
2. *dyadic tolerance admission for interval-width and endpoint-read budgets through D;*
3. *finite StreamName window exposure through W at the same endpoint and tolerance requests;*
4. *RegSeqRat readback through Q, attached to the displayed windows rather than to quotient streams;*
5. *Cauchy modulus compatibility through M, requiring the endpoint reads and the interval-width budget to share one finite schedule;*
6. *completion sealing through R, downstream of E_0, E_1, D, W, Q, M;*
7. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation exports a selected representative, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6253.1. The projection exposes exactly the endpoint rows, dyadic tolerance row, finite windows, regular rational readback, Cauchy modulus, completion seal, and structural rows.

The endpoint and tolerance obligations are upstream of every completion read. A consumer cannot inspect R before E_0, E_1 and D determine the displayed interval and its finite precision request.

The window and readback obligations then follow the sibling routes of chapter 126 and chapter 301. The Cauchy modulus row M binds those finite reads to the same dyadic budget supplied by chapter 291.

The completion seal is therefore terminal only for this displayed route. Since H, C, P, N transport, replay, source, and name the same finite packet without adding a selector, quotient, choice schedule, trichotomy decision, host equality row, or ambient completeness theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 6253.3 (Located interval completion Cauchy seal route)**.** *Every Cauchy-facing completion read exported by an accepted* $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ *carrier*

$$L = (E_0, E_1, D, W, Q, M, R, H, C, P, N)$$

factors through the finite route

$$E_0, E_1, D \longmapsto W, Q, M \longmapsto R, H, C, P, N.$$

The read first displays the Real endpoint rows and dyadic tolerance budget; it then opens finite StreamName windows, reads them through RegSeqRat, checks the Cauchy modulus row, and only then consumes the completion seal. The route exports no selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the endpoint and tolerance rows named in Definition 6253.1. A Cauchy-facing read has no access to the completion seal until the two $\mathsf{Real}^{\uparrow}$ endpoint rows and the dyadic budget row have been displayed.

The finite observation route is fixed by those rows. The StreamName row supplies only the requested finite windows, the RegSeqRat row supplies the regular rational readback attached to those windows, and the Cauchy modulus row binds the readbacks to one completion schedule.

Transport and replay do not create a second completion source. The rows H and C move and replay only the displayed coordinates, while P and N record provenance and local naming for the same packet.

Thus every exported Cauchy-facing completion observation factors through $E_0, E_1, D, W, Q, M, R, H, C, P, N$. Since the carrier contains no coordinate for a selected limit, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness, none of those principles is exported by the route. □

Falsifiable prediction: Every $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ Cauchy-facing read factors through two RealUp endpoint rows, one DyadicRatCore tolerance row, one StreamName window row, one RegSeqRat readback row, one Cauchy modulus row, one completion seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: Nearest siblings: real, regseqrat, streamname, dyadicratcore, and cauchy. $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries endpoint Real seals, dyadic tolerance data, finite windows, regular rational readback, Cauchy modulus scheduling, completion sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{LocatedIntervalCompletion}^{\uparrow}$ packet is generated from RealUp endpoint rows, DyadicRatCore tolerance rows, StreamName windows, RegSeqRat readback, Cauchy modulus rows, a completion seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 6253.1, Definition 6253.2, and Definition 6253.3 seed the finite located interval completion route over endpoint, tolerance, window, readback, Cauchy modulus, seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedIntervalCompletion}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected limits, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exactness, dyadic tolerance exactness, StreamName window replay, RegSeqRat readback, Cauchy modulus compatibility, completion sealing, classifier transport, and local naming for BEDC.Derived.LocatedIntervalCompletionUp.

Chapter 6254

A Concrete Naming Certificate for $\mathsf{FiniteVariationLedger}^{\uparrow}$

Following Jordan, Rudin, and Folland, the $\mathsf{FiniteVariationLedger}^{\uparrow}$ packet records the finite partition-sum object used before any bounded-variation or Riemann–Stieltjes consumer takes a supremum over partitions. It isolates the reusable finite ledger read by chapter 398: interval rows, displayed partition rows, endpoint names, dyadic edge observations, finite sums, common refinements, transport, replay, provenance, and local naming. The displayed partition and variation-sum rows consumed by this ledger are sourced through chapter 398 and the partition route of chapter 6386. The packet is deliberately finite. It does not contain a host supremum over all partitions, a variation measure, a quotient function space, or ambient real completeness.

Definition 6254.1 (Finite variation ledger carrier)**.** *A* $\mathsf{FiniteVariationLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (I, \Pi, X, D, V, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the interval row, read through* $\mathsf{RationalInterval}^{\uparrow}$ *or* $\mathsf{NestedInterval}^{\uparrow}$ *when the source supplies rational or nested endpoints.*
2. *Π is the displayed finite ordered partition of I, including endpoint order and adjacent-edge rows.*
3. *X records the endpoint rows, either as* $\mathsf{Real}^{\uparrow}$ *names or as* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational representatives consumed before a real seal is read.*
4. *D records the* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations comparing the adjacent endpoint values on every displayed edge of Π.*
5. *V is the finite variation-sum ledger, obtained from exactly the edge observations displayed in D.*
6. *R records common-refinement rows comparing Π with another displayed finite partition and transporting endpoint and edge-observation rows to the refinement.*
7. *H records componentwise* hsame *transport for the interval, partition, endpoint, edge-observation, finite-sum, common-refinement, package, and naming rows.*
8. *C records displayed* Cont *replay from an interval request to partition endpoints, dyadic edge reads, finite-sum formation, refinement comparison, and consumer exposure.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteVariationLedger}^{\uparrow}$, $\mathsf{RationalInterval}^{\uparrow}$, $\mathsf{NestedInterval}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteVariationLedger}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed interval row, finite partition, endpoint rows, edge observations, finite sum, common refinement, and visible transport, replay, package, and naming rows.

Lean4 ✓ `BEDC.Derived.FiniteVariationLedgerUp.FiniteVariationLedgerTasteGate single carrier alignment`

Theorem 6254.2 (Finite variation ledger NameCert obligations)**.** *For every accepted* $\mathsf{FiniteVariationLedger}^{\uparrow}$ *carrier* $L = (I, \Pi, X, D, V, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteVariationLedger}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval and displayed finite partition, endpoint exposure through* $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$*, dyadic edge-observation exposure through* $\mathsf{DyadicRatCore}^{\uparrow}$*, finite variation-sum exactness, common-refinement routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6254.1. The rows I and Π supply the interval and finite partition data, while X, D, V supply the endpoint, dyadic edge, and finite sum rows.

The finite variation obligation is therefore local to the displayed partition. Each edge contribution is read from D, and the row V records the finite sum of precisely those edge observations.

The refinement and stability obligations are carried by R, H, C. A consumer may move to a common refinement only through the displayed refinement row, with endpoint and edge data transported by the recorded hsame and replayed by the recorded Cont routes.

Finally, P and N provide package provenance and the local NameCert row. No step introduces a supremum over all partitions, a measure-valued object, a quotient of functions, or ambient real completeness. □

Theorem 6254.3 (Finite variation ledger common-refinement route)**.** *Every common-refinement read of an accepted* $\mathsf{FiniteVariationLedger}^{\uparrow}$ *packet factors through the finite route*

$$I, \Pi, X, D, V, R, H, C, P, N.$$

The refined read may use only the displayed interval row, both displayed finite partitions, transported endpoint $\mathsf{Real}^{\uparrow}$ *or* $\mathsf{RegSeqRat}^{\uparrow}$ *names, transported* $\mathsf{DyadicRatCore}^{\uparrow}$ *edge observations, the finite variation-sum ledger, the common-refinement row, and the visible* hsame, Cont, Pkg, *and* NameCert *rows. It exports no host supremum over partitions, variation measure, quotient function, selected refinement outside R, or ambient real-completeness principle.*

Proof. Start from the common-refinement row R in Definition 6254.1. It is attached to the same interval, partition, endpoint, dyadic, and finite-sum rows as the original ledger.

Transport the endpoint and edge-observation rows along H, and replay the finite read order along C. The refined ledger therefore reuses the same finite BEDC rows rather than introducing a new partition family.

The package and naming rows P, N then repack the refined finite sum as the public consumer route. Since the carrier has no coordinate for a global supremum, variation measure, quotient function, selected external refinement, or completeness principle, none of those objects can be exported. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVariationLedger}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FiniteVariationLedger}^{\uparrow}$ packet is generated from an interval row, a displayed finite partition, endpoint RealUp or RegSeqRatUp rows, DyadicRatCoreUp edge observations, a finite variation-sum ledger, a common-refinement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\mathsf{FiniteVariationLedger}\uparrow}$, seed)

Scope closed: Definition 6254.1, Definition 6254.2, and Definition 6254.3 seed the reusable finite partition-sum ledger for BoundedVariationUp and Real-facing integration consumers.

Formal verification: FormalStatus(NameCert$_{\mathsf{FiniteVariationLedger}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host suprema over all partitions, variation measures, quotient functions, selected global refinements, ambient Real completeness, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, endpoint exposure, dyadic edge observation, finite-sum exactness, common-refinement routing, transport, replay, provenance, and local naming for BEDC.Derived.FiniteVariationLedgerUp.

Chapter 6255

A Concrete Naming Certificate for $\mathsf{MetricCompletionMorphism}^{\uparrow}$

Following Bishop and Bridges on constructive metric completion, the $\mathsf{MetricCompletionMorphism}^{\uparrow}$ packet records the finite morphism surface by which a map between displayed metric-completion presentations preserves Cauchy data before any completed point or host map is read. It sits after the metric source of chapter 57, the complete-metric target of chapter 128, the completion-functor route of chapter 426, and the Cauchy-filter limit surface of chapter 1558. The packet names only finite completion-morphism rows; it does not assert quotient completion equality, choose representatives, import a host continuous map, use countable choice, or appeal to ambient completeness.

Definition 6255.1 (Metric completion morphism carrier)**.** *A* $\mathsf{MetricCompletionMorphism}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (S, T, W, R, D, M, H, C, P, N)$$

with the following displayed rows:

1. *S is the source metric row, recording the* $\mathsf{Met}^{\uparrow}$ *carrier, distance readbacks, and the Cauchy source interface that the morphism may inspect.*
2. *T is the target completion row, recording the* $\mathsf{CompleteMetric}^{\uparrow}$ *and completion-functor data visible to target-side consumers.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, listing the finite source windows and target windows inspected by the morphism.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for source distances, target distances, and modulus-scaled tolerance requests.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, naming the finite dyadic budgets used to compare source Cauchy windows with target Cauchy windows.*
6. *M is the modulus-preserving map row. It records the displayed map schedule and the finite checks that transport a source Cauchy tolerance into the target completion tolerance.*
7. *H records componentwise* hsame *transport for source metric, target completion, window, regular readback, dyadic tolerance, modulus-preserving map, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$S, T, M \longmapsto W, R, D \longmapsto H, P, N.$$

9. *P is the* Pkg *provenance over* MetricCompletionMorphism$^{\uparrow}$, Met$^{\uparrow}$, CompleteMetric$^{\uparrow}$, CompletionFunctor$^{\uparrow}$, CauchyFilterLimit$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionMorphism}^{\uparrow}}$ *row.*

Accepted carriers are compared only through these rows. There is no carrier coordinate for quotient completions, chosen representatives, host maps, countable choice, or ambient completeness.

Lean4 ✓ `BEDC.Derived.MetricCompletionMorphismUp.MetricCompletionMorphismTasteGate single carrier alignment`

Theorem 6255.2 (Metric completion morphism NameCert obligations)**.** *For every accepted* MetricCompletionMorp *carrier $G = (S, T, W, R, D, M, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MetricCompletionMorphism}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the source metric row, target completion row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, modulus-preserving map row, transport, replay, provenance, and local naming rows;*
2. *source and target completion visibility, requiring S and T to be displayed before any completion-facing morphism read is exported;*
3. *Cauchy-window preservation, requiring W, R, D, M to show that finite source Cauchy windows are sent to target Cauchy windows at the recorded tolerance;*
4. *modulus-stability under classifier transport, requiring H and C to preserve the same modulus row rather than replacing it by a host continuity theorem;*
5. *ledger refusal of quotient completions, chosen representatives, host maps, countable choice, and ambient completeness.*

No metric-completion morphism obligation is stored outside these rows.

Proof. Project the carrier of Definition 6255.1. The source and target rows are S, T, the finite Cauchy-data rows are W, R, D, M, and the structural rows are H, C, P, N.

The visibility clause follows because a morphism read cannot reach M before both the source metric row and target completion row have been displayed. The Cauchy-window clause then reads W, R, D, M in order: finite windows, regular rational readback, dyadic tolerance, and the modulus-preserving map schedule.

Transport and replay preserve the same rows componentwise. Since the carrier has no field for quotient completions, selected representatives, host maps, countable-choice schedules, or ambient completeness, the listed rows exhaust the local NameCert surface. □

Lean4 ✓ `BEDC.Derived.MetricCompletionMorphismUp.MetricCompletionMorphismCarrier namecert obligations`

Theorem 6255.3 (Metric completion morphism composition replay)**.** *Let $G = (S, T, W, R, D, M, H, C, P, N)$ and $G' = (T, U, W', R', D', M', H', C', P', N')$ be accepted* MetricCompletionMorphism$^{\uparrow}$ *carriers whose target row of G is the source row exposed to G' through componentwise* hsame*. Then their composition is replayed by concatenating the displayed finite routes through M and M':*

$$S, T, U \longmapsto W, R, D, M \longmapsto W', R', D', M' \longmapsto H, H', C, C', P, P', N, N'.$$

The replay preserves Cauchy windows and modulus budgets without adding quotient completion equality, chosen representatives, host maps, countable choice, or ambient completeness.

Proof. Start with the target-source agreement between the two carriers. Componentwise hsame identifies the completion row T exported by G with the row consumed by G', so the second morphism can read only the target completion data already displayed by the first route.

Apply the NameCert obligations of Definition 6255.2 to each carrier. The first route sends source Cauchy windows through W, R, D, M, and the second route sends the resulting target windows through W', R', D', M'.

Concatenate the replay rows C, C' and transport rows H, H'. Provenance and local naming preserve the same finite route, so the composite read has no opportunity to introduce a quotient completion, a chosen representative, a host map, a countable-choice schedule, or an ambient completeness theorem. □

Falsifiable prediction: Every accepted $\mathsf{MetricCompletionMorphism}^{\uparrow}$ consumer that asks for a completion morphism must display the source metric row, target completion row, StreamName windows, RegSeqRat readbacks, DyadicRatCore tolerances, modulus-preserving map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before receiving any completion-facing output.

Independence witness: metric, completemetric, completionfunctor, cauchyfilterlimit: no listed sibling is bijective to $\mathsf{MetricCompletionMorphism}^{\uparrow}$ because the morphism packet jointly carries source metric data, target completion data, finite Cauchy windows, regular readbacks, dyadic tolerances, a modulus-preserving map row, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{MetricCompletionMorphism}^{\uparrow}}$.

Constructive story: A $\mathsf{MetricCompletionMorphism}^{\uparrow}$ packet is generated from source MetricSpace data, target CompleteMetric and CompletionFunctor rows, CauchyFilterLimit-facing windows, StreamName observations, RegSeqRat readback, DyadicRatCore tolerance ledgers, a modulus-preserving map row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MetricCompletionMorphism}^{\uparrow}}$, seed)

Scope closed: Definition 6255.1, Definition 6255.2, and Definition 6255.3 seed the finite morphism route preserving metric-completion Cauchy data.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MetricCompletionMorphism}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, quotient completion equality, chosen representatives, host maps, countable choice, or ambient completeness.

Upgrade path: Obligation closure requires checked carrier admission, source and target visibility, Cauchy-window preservation, modulus-stability under classifier transport, composition replay, ledger refusal, provenance, and local naming for BEDC.Derived.MetricCompletionMorphismUp.

Chapter 6256

A Concrete Naming Certificate for $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$ packet records the finite uniform-Cauchy surface for located compact intervals. It reads located endpoints through **??** and chapter 394, interval containment through chapter 3986, finite observations through chapter 126, rational readback through chapter 301, dyadic tolerance through chapter 291, and terminal sealing through chapter 38. The packet names a compact-interval uniform-Cauchy route as finite certificate data; it is not a Heine–Borel theorem, an open-cover compactness theorem, a quotient completion, a selected limit, or a choice principle.

Definition 6256.1 (Located interval uniform Cauchy carrier)**.** *A* $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (L, I, K, W, R, D, M, S, H, C, P, N)$$

with the following displayed rows:

1. *L is the* $\mathsf{LocatedReal}^{\uparrow}$ *endpoint row, carrying the lower and upper located real names used by the interval request.*
2. *I is the* $\mathsf{LocatedInterval}^{\uparrow}$ *row, exposing located containment and interval-membership data for the same endpoints.*
3. *K is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *row, read only as a finite compact-interval certificate surface for this displayed interval.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the interval indices requested by L, I, K.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic mesh, refinement, and comparison budget used by uniform-Cauchy reads.*
7. *M is the uniform-Cauchy modulus row, requiring one dyadic tolerance budget to control all displayed interval windows accepted by W and R.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the located endpoint, located interval, compact-interval, window, readback, tolerance, and uniform-Cauchy modulus rows have been displayed.*
9. *H records componentwise* hsame *transport for L, I, K, W, R, D, M, S.*

10. *C records displayed* Cont *replay through the finite route*

$$L, I, K \longmapsto W, R, D, M \longmapsto S, H, P, N.$$

11. *P is the* Pkg *provenance over* LocatedIntervalUniformCauchy$^{\uparrow}$, LocatedReal$^{\uparrow}$, LocatedInterval$^{\uparrow}$, BishopCompactInterval$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* NameCert$_{\text{LocatedIntervalUniformCauchy}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $L, I, K, W, R, D, M, S, H, C, P, N$. It has no coordinate for general Heine–Borel compactness, open-cover compactness, quotient completion equality, selected limits, countable choice, or host real equality.

Lean4 ✓ `BEDC.Derived.LocatedIntervalUniformCauchyUp.LocatedIntervalUniformCauchyTasteGate single carrier alignment`

Theorem 6256.2 (Located interval uniform Cauchy NameCert obligations)**.** *For every accepted* LocatedIntervalUniformCauchy$^{\uparrow}$ *carrier $U = (L, I, K, W, R, D, M, S, H, C, P, N)$, the local* NameCert$_{\text{LocatedIntervalUnifor}}$ *surface consists exactly of located-real endpoint admission, located-interval containment admission, compact-interval certificate admission, finite StreamName window exposure, RegSeqRat readback, DyadicRatCore tolerance exactness, uniform-Cauchy modulus admission, Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6256.1. The projection exposes the endpoint row, located interval row, compact-interval row, finite windows, rational readback, dyadic tolerance, uniform-Cauchy modulus, seal, and structural rows.

The endpoint and interval obligations are upstream of every uniform-Cauchy read. A consumer must first display L, I, K before it can ask for the finite windows W or the readback row R.

The dyadic tolerance row D and modulus row M then bind all accepted interval windows to one uniform request. The terminal seal S is downstream of those rows, while H, C, P, N transport, replay, source, and name the same packet.

Since the carrier has no coordinate for general compactness, quotient equality, selected limits, choice, or host real equality, the listed rows exhaust the local NameCert surface. □

Theorem 6256.3 (Located interval uniform Cauchy finite-net handoff)**.** *Every finite-net consumer of an accepted* LocatedIntervalUniformCauchy$^{\uparrow}$ *carrier factors through*

$$L, I, K \longmapsto W, R, D, M.$$

The compact-interval row K may hand off only the displayed located interval to the finite windows W, regular readback R, dyadic tolerance D, and uniform-Cauchy modulus M; it does not export an open-cover compactness theorem, a general Heine–Borel theorem, or a choice-selected net.

Proof. Start from the displayed carrier. The finite-net request has no source other than the located endpoint, located interval, and compact-interval rows L, I, K.

The window and readback rows are then read at the tolerances shown by D. The modulus row M records that the same dyadic budget controls the displayed interval windows, so the finite-net handoff is a local route through W, R, D, M.

The structural rows preserve the same route. Because no row stores an open cover, a selected finite subcover, or a hidden choice schedule, the handoff is exactly the displayed finite-net route. □

Theorem 6256.4 (Located interval uniform Cauchy Real seal)**.** *For every accepted* $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$ *carrier* $U = (L, I, K, W, R, D, M, S, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* S *can be consumed only after the located endpoint, located interval, compact-interval, finite-window, regular-readback, dyadic-tolerance, and uniform-Cauchy modulus rows have been displayed. The seal is therefore a local real-name endpoint for the packet, not a quotient completion, selected limit, ambient completeness theorem, or host real representative.*

Proof. Read the carrier projection. The seal row S appears after L, I, K, W, R, D, M, and the carrier lists no second real-facing coordinate.

Apply the NameCert obligation surface of Definition 6256.2. It forces window exposure, regular readback, dyadic tolerance exactness, and uniform-Cauchy modulus admission before the seal is public.

Finally use the non-escape content of the carrier: H, C, P, N only transport, replay, source, and name the same finite packet. Thus a consumer may read S only as the displayed terminal $\mathsf{Real}^{\uparrow}$ seal, not as any excluded completion, compactness, or choice object. □

Theorem 6256.5 (Located interval uniform Cauchy tail stability)**.** *For every accepted* $\mathsf{LocatedIntervalUniformCauch}$ *carrier* $U = (L, I, K, W, R, D, M, S, H, C, P, N)$*, replacing the displayed finite-net window* W *by a later cofinal tail window preserves the visible route through the located endpoint row* L*, located interval row* I*, compact interval row* K*, uniform-Cauchy modulus row* M*, StreamName windows, RegSeqRat readback* R*, terminal Real seal* S*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* N*. The tail read does not introduce countable choice, a selected limit, open-cover compactness, or ambient Real completeness.*

Proof. Project the carrier of Definition 6256.1. The finite-net window, readback, tolerance, modulus, seal, and structural rows are all displayed as coordinates of the same packet.

Apply Definition 6256.3. The finite-net consumer reads only the route $L, I, K \longmapsto W, R, D, M$, so a later cofinal tail window can replace W only inside that displayed route and not by adding an external selector.

Next use Definition 6256.4. The terminal seal S is downstream of the located endpoint, located interval, compact-interval, finite-window, readback, tolerance, and modulus rows, so the tail read reaches the same Real-facing endpoint only after those rows remain visible.

The rows H, C, P, N then transport, replay, source, and name the same cofinal-tail route. Since the carrier contains no coordinate for countable choice, selected limits, open-cover compactness, or ambient Real completeness, the tail substitution preserves the visible certificate surface without importing any excluded principle. □

Falsifiable prediction: Every $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$ read factors through located real endpoints, a located interval row, a compact interval row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a uniform-Cauchy modulus, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: Nearest siblings: locatedreal, locatedinterval, compactinterval, streamname, regseqrat, dyadicratcore, and real. $\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries located endpoints, interval containment, compact-interval admission, finite windows, rational readback, dyadic tolerance, uniform-Cauchy control, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedIntervalUniformCauchy}^{\uparrow}}$.

Constructive story: A LocatedIntervalUniformCauchy$^{\uparrow}$ packet is generated from LocatedReal endpoint rows, a LocatedInterval containment row, a CompactInterval certificate row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a uniform-Cauchy modulus row, a RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{LocatedIntervalUniformCauchy}^{\uparrow}}$, seed)

Scope closed: Definition 6256.1, Definition 6256.2, Definition 6256.3, and Definition 6256.4 seed the finite compact-interval uniform-Cauchy route over located endpoints, interval containment, compact-interval admission, finite windows, readback, tolerance, modulus, seal, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{LocatedIntervalUniformCauchy}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general Heine–Borel compactness, open-cover compactness, quotient completion, selected limits, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located endpoint exactness, interval containment exactness, compact-interval admission, finite-window replay, RegSeqRat readback, dyadic tolerance exactness, uniform-Cauchy modulus exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedIntervalUniformCauchyUp.

Chapter 6257

A Concrete Naming Certificate for $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ packet records the finite search basis through which a compact real interval exposes located endpoint refinements, dyadic tolerances, finite stream windows, regular rational readback, and a terminal real-name seal. It sits after the compact real interval surface of chapter 5553, the located refinement route of chapter 6199, the interval-completion route of chapter 6253, the dyadic tolerance core of chapter 291, the finite window interface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite interval-search data; it does not assert Heine–Borel compactness, selected subcovers, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6257.1 (Compact real interval search basis carrier)**.** *A* $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, L, F, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact real interval source row, read through* $\mathsf{CompactRealInterval}^{\uparrow}$ *with its displayed lower and upper* $\mathsf{Real}^{\uparrow}$ *endpoint seals.*
2. *L is the located endpoint refinement row, using the located interval refinement surface to expose endpoint brackets and radius data at requested precisions.*
3. *F is the finite search and cover-basis row. It lists the displayed interval cells, membership tests, cover probes, and retained cells available to a finite search consumer.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, including the mesh budget, endpoint-read budget, and refinement depth used by L and F.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the endpoint and tolerance requests.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those finite windows and dyadic tolerance data.*
7. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row consumed only after the compact source, located refinement, finite search basis, dyadic tolerance, StreamName window, and RegSeqRat readback rows have been displayed.*
8. *H records componentwise* hsame *transport for K, L, F, D, W, R, E.*

9. *C records displayed* Cont *replay for search requests through the route*

$$K, L, F, D \longmapsto W, R \longmapsto E, H, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$, $\mathsf{CompactRealInterval}^{\uparrow}$, LocatedIntervalRefinement $\mathsf{LocatedIntervalCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, L, F, D, W, R, E, H, C, P, N$. It has no coordinate for an open-cover theorem, a selected finite subcover, countable choice, quotient real equality, host endpoint equality, or ambient completeness.

Lean4 (definition) `BEDC.Derived.CompactRealIntervalSearchBasisUp.CompactRealIntervalSearchBasisUp`
Lean4 ✓ `BEDC.Derived.CompactRealIntervalSearchBasisUp.CompactRealIntervalSearchBasisTasteGate`
`single carrier alignment`

Theorem 6257.2 (Compact real interval search basis obligations)**.** *For every accepted* CompactRealIntervalSearchB *carrier $S = (K, L, F, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}$ *surface is exhausted by the following finite obligations:*

1. *compact source admission for the* $\mathsf{CompactRealInterval}^{\uparrow}$ *row K and its displayed endpoint seals;*
2. *located refinement admission for the endpoint brackets and radius data in L;*
3. *finite search-basis admission for the cell list, probe list, and retained-cell ledger in F;*
4. *dyadic tolerance exactness for the mesh, endpoint-read, and refinement-depth budgets in D;*
5. *StreamName window exposure through W at the endpoint and tolerance requests named by K, L, D;*
6. *RegSeqRat readback through R, attached to the displayed windows rather than to quotient streams;*
7. *Real seal routing through E, downstream of K, L, F, D, W, R;*
8. *componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming through H, C, P, N.*

No obligation exports open-cover compactness, selected subcovers, countable choice, host endpoint equality, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6257.1. The projection exposes exactly the compact source, located refinement, finite search basis, dyadic tolerance, finite window, regular readback, real seal, transport, replay, provenance, and local naming rows.

The compact source and located refinement obligations are upstream of every search read. A consumer cannot inspect a retained search cell until K, L and the dyadic budget D determine the displayed interval bracket and precision request.

The finite search row then supplies the only cover-basis data available to the packet. StreamName and RegSeqRat rows read that data through finite windows and regular rational readback, while the Real seal is terminal only for this displayed route.

The support rows H, C, P, N transport, replay, source, and name the same packet. Since no row stores an open-cover theorem, selected subcover, choice schedule, quotient equality, host endpoint equality, or ambient completeness theorem, the listed obligations exhaust the local NameCert surface. □

Theorem 6257.3 (Compact real interval search finite cover route)**.** *Every finite-search consumer reading an accepted* $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ *carrier factors through the ordered route*

$$K, L, F, D, W, R, E, H, C, P, N.$$

The consumer first displays the compact interval source and located endpoint refinement, then reads the finite search and cover-basis row against a dyadic tolerance budget, then opens StreamName windows, performs RegSeqRat readback, and only then reaches the $\mathsf{Real}^{\uparrow}$ *seal. The route exports finite cover-basis data, not an unrestricted open-cover compactness theorem or a choice-selected subcover.*

Proof. Begin with the source and refinement rows K, L. These rows expose the compact real interval and the located endpoint brackets used by the search basis.

Read the finite search row F together with the tolerance row D. The listed cells, probes, retained-cell ledger, mesh budget, endpoint-read budget, and refinement-depth budget are all finite rows of the same packet.

Next open the StreamName window W and read it through the RegSeqRat row R. The consumer reaches the Real seal E only after those finite observations have been displayed.

Finally transport and replay through H, C, P, N. These rows preserve the same route and do not add open-cover compactness, selected subcovers, countable choice, host equality, quotient equality, or ambient completeness. Thus every finite-search read factors through the stated route. □

Theorem 6257.4 (Compact real interval search Real seal nonescape)**.** *No accepted* CompactRealIntervalSearchBasis *carrier exposes a* $\mathsf{Real}^{\uparrow}$ *seal read that bypasses its located endpoint refinement, finite search basis, dyadic tolerance row, StreamName finite window, or RegSeqRat readback.*

Equivalently, every $\mathsf{Real}^{\uparrow}$*-facing observation of the packet is downstream of* K, L, F, D, W, R *and carries the displayed* hsame *transport,* Cont *replay,* Pkg *provenance, and local* $\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}$ *row.*

Proof. The seal coordinate in Definition 6257.1 is E. Its inputs are the compact source, located refinement, finite search basis, dyadic tolerance, finite window, and regular readback rows.

If a consumer asks for a Real-facing observation, Definition 6257.3 forces that observation through K, L, F, D, W, R before E is available.

The remaining rows H, C, P, N transport, replay, source, and name the same route. Since the carrier has no coordinate for an alternate seal, hidden endpoint representative, selected subcover, choice schedule, or ambient completeness theorem, a bypass route is outside the accepted packet. □

Theorem 6257.5 (Compact real interval search basis NameCert surface)**.** *The public* CompactRealIntervalSearchBa *surface consists exactly of the carrier in Definition 6257.1, the obligation list in Definition 6257.2, the finite cover route in Definition 6257.3, and the Real seal nonescape boundary in Definition 6257.4.*

It is therefore a finite NameCert surface for compact real interval search data, not a proof of open-cover compactness, selected subcovers, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. The carrier determines the coordinates available to the packet. The obligation theorem names the local NameCert checks for those coordinates and excludes additional compactness or choice data.

The finite cover route then fixes the consumer order from compact source and located refinement through search basis, tolerance, finite windows, readback, seal, transport, replay, provenance, and local naming.

The nonescape theorem closes the Real-facing boundary by ruling out any seal read that bypasses the displayed finite route. Combining these three surfaces gives exactly the advertised public NameCert surface. □

Falsifiable prediction: Every $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ finite-search read factors through one compact real interval source row, one located endpoint refinement row, one finite search-basis row, one DyadicRatCore tolerance row, one StreamName window row, one RegSeqRat readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert.

Independence witness: Nearest siblings: compact-real-interval, located-interval-refinement, located-interval-completion, dyadicratcore, streamname, regseqrat, and real. $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact interval source data, located endpoint refinement, finite search and cover-basis rows, dyadic tolerance budgets, finite stream windows, regular rational readback, a real seal, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}$ packet is generated from compact real interval source rows, located endpoint refinement rows, finite search and cover-basis rows, DyadicRatCore tolerance rows, StreamName finite windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6257.1, Definition 6257.2, Definition 6257.3, Definition 6257.4, and Definition 6257.5 seed the finite compact real interval search-basis route over source, refinement, cover basis, tolerance, window, readback, seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealIntervalSearchBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, bridge checking, host endpoint equality, or quotient real equality.

Upgrade path: Obligation closure requires checked carrier admission, located refinement exactness, finite search-basis admission, dyadic tolerance exactness, StreamName window replay, RegSeqRat readback, Real seal nonescape, classifier transport, and local naming for BEDC.Derived.CompactRealIntervalSearchBasisUp.

Chapter 6258

A Concrete Naming Certificate for $\mathsf{CountablyBasedUniformity}^{\uparrow}$

Following Bourbaki and Kelley, the $\mathsf{CountablyBasedUniformity}^{\uparrow}$ packet records the standard countable-base interface for a uniform space as finite BEDC naming data. It sits after the entourage discipline of chapter 339, the Cauchy-filter consumer of chapter 333, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the Real seal of chapter 38. The packet gives completion consumers a countable basis route, not an arbitrary open-cover uniformity, quotient equality, countable-choice selector, or ambient completeness principle.

Definition 6258.1 (CountablyBasedUniformity carrier)**.** *A* $\mathsf{CountablyBasedUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (E, F, S, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *E is the* $\mathsf{UniformSpace}^{\uparrow}$ *entourage row whose requests are read only through the displayed countable basis.*
2. *F is the* $\mathsf{CauchyFilter}^{\uparrow}$ *consumer row that receives Cauchy data from the basis rather than from an arbitrary open-cover surface.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row indexing the displayed countable basis requests.*
4. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to those windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling finite refinements between basis elements.*
6. *R is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row available only after E, F, S, Q, D have been replayed.*
7. *H records componentwise* hsame *transport for the uniform, filter, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay of the route*

$$E, F \longmapsto S, Q, D \longmapsto R.$$

9. *P is the* Pkg *provenance over* $\mathsf{CountablyBasedUniformity}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CountablyBasedUniformity}^{\uparrow}}$ *row.*
The classifier compares accepted carriers only through $E, F, S, Q, D, R, H, C, P, N$. It has no coordinate for arbitrary open-cover uniformity, quotient equality of uniform spaces, countable choice, selected Cauchy-filter limits, host Real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.CountablyBasedUniformityUp.CountablyBasedUniformityTasteGate single carrier alignment`

Theorem 6258.2 (CountablyBasedUniformity NameCert obligations)**.** *For every accepted* $\mathsf{CountablyBasedUniform}$ *carrier*

$$U = (E, F, S, Q, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CountablyBasedUniformity}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, countable basis coverage for the* $\mathsf{UniformSpace}^{\uparrow}$ *requests, finite refinement stability through the* $\mathsf{DyadicRatCore}^{\uparrow}$ *ledger,* $\mathsf{CauchyFilter}^{\uparrow}$ *handoff through the generated basis, completion-consumer compatibility through the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *route, and ledger nonescape from arbitrary open-cover uniformity, quotient equality, countable choice, selected limits, host Real equality, or ambient Real completeness.*

Proof. Project the carrier of Definition 6258.1. The projection exposes exactly the uniform entourage row, Cauchy-filter consumer, stream-window basis index, regular rational readback, dyadic tolerance ledger, Real seal, and structural rows.

The countable-basis part is read through E, S, Q, D. The row S displays the finite windows for the basis, Q supplies the regular rational readback, and D records the finite refinement relation between basis elements before any completion-facing consumer can inspect the seal.

The filter and completion handoff then uses F and R. A Cauchy-filter request must pass through the generated basis before reaching the downstream Real seal, so the consumer receives a basis-indexed route rather than arbitrary open-cover data.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since none of the excluded principles occurs as a carrier coordinate, no NameCert obligation can export quotient equality, countable choice, selected limits, host equality, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CountablyBasedUniformity}^{\uparrow}$ completion consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, componentwise hsame transport, displayed Cont replay, Pkg provenance, and the local $\mathsf{NameCert}$ row refutes the packet surface.

Independence witness: uniformspace, cauchyfilter, streamname, regseqrat, dyadicratcore, real: no listed sibling is bijective to $\mathsf{CountablyBasedUniformity}^{\uparrow}$ because this packet binds countable basis coverage, finite refinement stability, Cauchy-filter handoff, completion-consumer compatibility, transport, replay, provenance, and local naming as one L10 route.

Closure status for $\mathsf{NameCert}_{\mathsf{CountablyBasedUniformity}^{\uparrow}}$.

Constructive story: A $\mathsf{CountablyBasedUniformity}^{\uparrow}$ packet is generated from a UniformSpace entourage row, CauchyFilter consumer row, StreamName finite-window basis row, RegSeqRat readback row, DyadicRatCore finite-refinement ledger, RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CountablyBasedUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6258.1 and Definition 6258.2 seed the countable-basis uniformity interface for Cauchy-filter and completion-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CountablyBasedUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover uniformity, quotient equality, selected limits, countable choice, host Real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, countable basis coverage, finite refinement stability, Cauchy-filter handoff, completion-consumer compatibility, transport, replay, provenance, local naming, and ledger nonescape for BEDC.Derived.CountablyBasedUniformityUp.

Chapter 6259

A Concrete Naming Certificate for $\mathsf{GaloisConnection}^{\uparrow}$

Following Birkhoff and Ore on Galois connections, the $\mathsf{GaloisConnection}^{\uparrow}$ packet records an order-theoretic adjunction between two displayed ordered carriers as finite NameCert data. It sits over the preorder substrate of chapter 52, the antisymmetric carrier discipline of chapter 53, the lattice consumers of chapter 55, and the order comparison surface of chapter 1378. The packet names monotone maps and the adjunction equivalence only through displayed rows; it does not import quotient posets, powerset closure, host implication, or choice-selected fixed points.

Definition 6259.1 (Galois connection carrier)**.** *A* $\mathsf{GaloisConnection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (P, Q, F, G, M, A, T, R, L, N)$$

with the following rows:

1. *P is the source preorder or poset carrier, read through the* $\mathsf{Preorder}^{\uparrow}$ *and* $\mathsf{Poset}^{\uparrow}$ *sibling surfaces.*
2. *Q is the target preorder or poset carrier, read through the same ordered-carrier discipline.*
3. *F is the displayed monotone map row from P to Q.*
4. *G is the displayed monotone map row from Q to P.*
5. *M records the two monotonicity ledgers: $x \leq_P x'$ implies $F(x) \leq_Q F(x')$, and $y \leq_Q y'$ implies $G(y) \leq_P G(y')$.*
6. *A records the adjunction inequality equivalence, namely that each displayed pair of histories satisfies $F(x) \leq_Q y$ exactly when $x \leq_P G(y)$ through the carried order rows.*
7. *T records componentwise* hsame *transport for $P, Q, F, G, M, A, R, L, N$.*
8. *R records displayed* Cont *replay for source-order reads, target-order reads, monotonicity reads, adjunction reads, and downstream closure or kernel consumers.*
9. *L is the* Pkg *provenance over* $\mathsf{GaloisConnection}^{\uparrow}$, $\mathsf{Preorder}^{\uparrow}$, $\mathsf{Poset}^{\uparrow}$, $\mathsf{Lattice}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{GaloisConnection}^{\uparrow}}$ *row.*

The classifier compares packets only by the displayed ordered carriers, the two map rows, the monotonicity ledger, the adjunction equivalence, structural transport, replay, provenance, and local

naming. It has no coordinate for an external order completion, quotient carrier, powerset universe, selected fixed point, or host-level implication principle.

Lean4 ✓ `BEDC.Derived.GaloisConnectionUp.GaloisConnectionTasteGate single carrier alignment`

Theorem 6259.2 (Galois connection NameCert obligations)**.** *For every accepted* $\mathsf{GaloisConnection}^{\uparrow}$ *carrier* $K = (P, Q, F, G, M, A, T, R, L, N)$*, the local* $\mathsf{NameCert}_{\mathsf{GaloisConnection}^{\uparrow}}$ *surface consists exactly of carrier admission for the ordered source and target rows, monotone-map exposure for* F *and* G*, monotonicity ledger exposure through* M*, adjunction-equivalence exposure through* A*, componentwise transport through* T*, displayed replay through* R*, provenance through* L*, and local naming through* N*.*

Lean4 ✓ `BEDC.Derived.GaloisConnectionUp.GaloisConnectionCarrier namecert obligations`

Proof. Project the carrier of Definition 6259.1. The rows P, Q supply the ordered carriers, while F, G supply only displayed map rows between those carriers.

The monotonicity ledger M is required before either map can be read as order preserving. The adjunction row A is required before a consumer may translate an inequality $F(x) \leq_Q y$ into $x \leq_P G(y)$ or conversely.

Transport, replay, provenance, and local naming preserve the same finite packet. Since the carrier has no quotient order, powerset universe, choice-selected fixed point, or host implication coordinate, the NameCert obligations are exactly the listed rows. □

Theorem 6259.3 (Galois connection closure operator)**.** *Let* $K = (P, Q, F, G, M, A, T, R, L, N)$ *be an accepted* $\mathsf{GaloisConnection}^{\uparrow}$ *carrier. Any closure-operator consumer on* P *may read the composite* $G \circ F$ *only through the displayed route*

$$P \longrightarrow F \longrightarrow Q \longrightarrow G \longrightarrow P$$

together with the monotonicity ledger M *and adjunction ledger* A*. The packet supplies the inflationary and monotone closure-facing boundary for that composite; it does not supply arbitrary suprema, a Moore-family representation, quotient fixed points, compactness, or a host powerset closure theorem.*

Proof. Start with the source row P and the map rows F, G. The composite $G \circ F$ is readable only by first applying the displayed F row and then the displayed G row.

Use the two monotonicity entries in M for the monotone part of the closure-facing boundary. A consumer cannot obtain monotonicity from the maps alone, because the maps are accepted as order maps only through M.

Use the adjunction row A for the inflationary direction. The equivalence specializes the inequality comparison so that the unit-style order read $x \leq_P G(F(x))$ is obtained from the displayed source and target order rows, not from host implication.

The structural rows T, R, L, N preserve this composite route. They do not add suprema, Moore-family data, quotient fixed points, compactness, or powerset closure, so the closure-operator consumer receives only the displayed finite boundary. □

Theorem 6259.4 (Galois connection kernel stability)**.** *For an accepted* $\mathsf{GaloisConnection}^{\uparrow}$ *carrier* $K = (P, Q, F, G, M, A, T, R, L, N)$*, any kernel or fixed-row consumer of the composite* $F \circ G$ *on* Q *must pass through the same two-map route, monotonicity ledger, and adjunction-equivalence ledger. The packet preserves the kernel-facing stability of the composite under* hsame *transport, but it does not identify the kernel with a chosen fixed-point set, a quotient lattice, or an interior operator unless a separate finite carrier supplies those rows.*

Proof. Read the target side first. A kernel-facing consumer begins with Q, then reaches the composite $F \circ G$ only through G followed by F.

The monotonicity ledger M transports order comparisons along both maps, and the adjunction ledger A records the inequality equivalence that controls the composite. These rows are part of the carrier and remain visible to the consumer.

Now apply componentwise transport through T and replay through R. If the ordered carriers, map rows, monotonicity rows, and adjunction rows are transported by hsame, the same finite kernel-facing boundary is replayed through L and named by N.

No coordinate of K contains a chosen fixed-point set, quotient lattice, or interior-operator carrier. Those structures can consume the Galois-connection packet, but they must bring their own finite carrier rows before they can be named. □

Falsifiable prediction: Deleting the adjunction row A leaves two monotone maps but no route that translates inequalities between P and Q.

Independence witness: preorder, poset, lattice, order: none is bijective to $\mathsf{GaloisConnection}^{\uparrow}$. The preorder and poset siblings supply ordered carriers without paired adjoint maps; the lattice sibling supplies meet/join consumers without the two-map adjunction row; the order sibling supplies comparison rows without the provenance, transport, replay, and local NameCert packet for $P, Q, F, G, M, A, T, R, L, N$.

Closure status for $\mathsf{NameCert}_{\mathsf{GaloisConnection}^{\uparrow}}$.

Constructive story: A $\mathsf{GaloisConnection}^{\uparrow}$ packet is generated from two displayed ordered carrier rows, two monotone map rows, a monotonicity ledger, an adjunction-equivalence ledger, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GaloisConnection}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6259.1, Definition 6259.2, Definition 6259.3, and Definition 6259.4 seed the finite order-adjunction surface over displayed ordered carriers, monotone maps, adjunction equivalence, closure-facing consumers, kernel-facing consumers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GaloisConnection}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.GaloisConnectionUp`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, quotient order completion, arbitrary suprema, Moore-family representation, chosen fixed points, compactness, host powerset closure, or a standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, monotonicity ledgers, adjunction-equivalence transport, closure-operator boundary, kernel-stability boundary, replay, provenance, and local naming for BEDC.Derived.GaloisConnectionUp.

Chapter 6260

A Concrete Naming Certificate for $\mathsf{StoneWeierstrassLattice}^{\uparrow}$

Following Stone and Weierstrass, the $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ packet records the finite lattice-subalgebra route by which compact real-valued approximation can be read without selecting external dense families or importing an ambient function-space completion. It sits beside the separating-algebra certificate of chapter 1919, the compact metric source of chapter 129, the continuous-name surface of chapter 59, the uniform-modulus handoff of chapter 369, and the real-name boundary of chapter 38. The chapter names only displayed compact metric rows, continuous real-valued rows, lattice operations, separation evidence, finite approximation budgets, transport, replay, provenance, and a local certificate.

Definition 6260.1 (Stone–Weierstrass lattice carrier)**.** *A* $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, F, A, S, B, R, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row, including the named metric windows and finite compactness witnesses consumed by the approximation route.*
2. *F is the continuous real-valued function row, reading each inspected target through the continuous chapter interface and the same* $\mathsf{Real}^{\uparrow}$ *windows used by the rest of the packet.*
3. *A is the lattice subalgebra row. It displays constants, pointwise addition, pointwise multiplication, scalar multiplication, pointwise maximum, pointwise minimum, and the finite replay order by which these operations stay inside the named family.*
4. *S is the separating-family evidence. For each displayed pair of apart compact metric names it records a member of A and the finite real-window comparison that separates the pair.*
5. *B is the finite approximation budget. It records the rational tolerance, the compact metric finite-net or finite-cover row used for that tolerance, and the finite list of lattice-subalgebra expressions inspected before the consumer accepts an approximation.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *seal. Every value comparison, tolerance comparison, lattice operation output, and approximation error read passes through this row before it is exposed.*
7. *H records componentwise* hsame *transport for compact metric source, continuous target, lattice operations, separating-family evidence, finite budget, real seal, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a compact real-valued approximation request through K, F, A, S, B, R, H and only then to provenance and naming.*
9. *P is the* Pkg *provenance tying the packet to* $\mathsf{StoneWeierstrassLattice}^{\uparrow}$, $\mathsf{StoneWeierstrass}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StoneWeierstrassLattice}^{\uparrow}}$ *row.*

Accepted carriers are compared only through these rows. There is no coordinate for classical choice, arbitrary function equality, measure completion, a selected dense sequence, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.StoneWeierstrassLatticeUp.StoneWeierstrassLatticeTasteGate single carrier alignment`

Theorem 6260.2 (Stone–Weierstrass lattice NameCert obligations)**.** *For every accepted* $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ *carrier*

$$L = (K, F, A, S, B, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{StoneWeierstrassLattice}^{\uparrow}}$ *surface consists exactly of compact metric source admission, continuous real-valued target admission, lattice subalgebra operation visibility, separating-family evidence, finite approximation budget admission,* $\mathsf{Real}^{\uparrow}$ *sealing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the rows of Definition 6260.1. The compact metric source is stored in K, while the inspected real-valued continuous target is stored in F.

The algebraic and lattice closure obligations are exactly the row A. The separating-family obligations are exactly the row S, and every accepted approximation request must pass through the finite budget row B.

The real-name boundary is fixed by R. Transport, replay, provenance, and naming are then supplied by H, C, P, N. Since the carrier has no other coordinates, the listed obligations exhaust the local certificate surface. □

Lean4 ✓ `BEDC.Derived.StoneWeierstrassLatticeUp.StoneWeierstrassLatticeNameCertObligations`

Theorem 6260.3 (Stone–Weierstrass lattice uniform approximation route)**.** *Every approximation-facing consumer of an accepted* $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ *carrier factors through the finite route*

$$K, F \longmapsto A, S \longmapsto B, R \longmapsto H, C, P, N.$$

Thus the consumer receives a compact metric source, a continuous real-valued target, displayed lattice-subalgebra operations, separating-family evidence, a finite approximation budget, a $\mathsf{Real}^{\uparrow}$ *seal, transport, replay, provenance, and local naming; it receives no selected subsequence, open-ended function equality test, measure completion, or ambient completeness principle.*

Proof. Begin with an approximation-facing read of the accepted carrier. The request first reaches K and F, so the source and target are the displayed compact metric and continuous real-valued rows.

The request then reads lattice operations from A and separation data from S. These rows supply only finite expressions and finite real-window comparisons.

The finite budget row B selects the displayed tolerance and compact witness used by the consumer, while R seals every comparison through $\mathsf{Real}^{\uparrow}$. The structural rows H, C, P, N preserve the same route, so the excluded principles cannot be recovered from the packet. □

Theorem 6260.4 (Stone–Weierstrass lattice real-seal nonescape)**.** *In an accepted* $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ *carrier, every value read, lattice-operation output, separating comparison, tolerance comparison, and approximation-error claim reaches downstream consumers through the displayed* $\mathsf{Real}^{\uparrow}$ *seal row* R*. Consequently no downstream consumer can use the packet to claim classical choice, arbitrary function equality, measure completion, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Inspect the carrier route. Value reads and operation outputs are produced by F and A, but they are not exported until R records the corresponding real-name windows.

Separating comparisons and approximation-error claims are produced by S and B. Those rows also feed their real comparisons through R before replay reaches the public naming rows.

The remaining rows H, C, P, N transport and name only the already sealed route. Since no excluded principle is a coordinate of L, and no replay edge bypasses R, the real-seal boundary is nonescaping. □

Falsifiable prediction: Every accepted $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ approximation-facing read factors through compact metric rows, continuous real-valued target rows, displayed lattice-subalgebra operations, separating-family evidence, a finite approximation budget, a $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: stone-weierstrass, compactmetric, continuous, uniformmodulus, real: $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries lattice-subalgebra closure, separating-family evidence, compact metric approximation budgets, real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{StoneWeierstrassLattice}^{\uparrow}}$.

Constructive story: A $\mathsf{StoneWeierstrassLattice}^{\uparrow}$ packet is generated from compact metric source rows, continuous real-valued target rows, displayed lattice-subalgebra operations, separating-family evidence, finite approximation budgets, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StoneWeierstrassLattice}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6260.1, Definition 6260.2, Definition 6260.3, and Definition 6260.4 seed the compact real-valued lattice approximation route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StoneWeierstrassLattice}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close classical choice, arbitrary function equality, measure completion, ambient RealUp completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, lattice-operation visibility, separating-family exactness, finite-budget routing, real-seal nonescape, and uniform approximation route proofs for BEDC.Derived.StoneWeierstrassLatticeUp.

Chapter 6261

A Concrete Naming Certificate for $\mathsf{PringsheimConvergence}^{\uparrow}$

Pringsheim and Bishop–Bridges constructive analysis treat convergence over two indices as an explicit route through finite rectangles, cofinal tails, and displayed regular readback, not as an ambient appeal to a completed double limit. The $\mathsf{PringsheimConvergence}^{\uparrow}$ packet records this route for positive rectangle partial sums after the two-parameter surface of chapter 460, the iterated handoff of chapter 1686, the finite windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. It is a finite NameCert reconstruction of the Pringsheim route; it does not claim host double-series convergence, quotient equality of arrays, selected sums, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6261.1 (Pringsheim convergence carrier)**.** *A* $\mathsf{PringsheimConvergence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (X, S, T, M, W, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CauchyDoubleSequence}^{\uparrow}$ *source row. It records the two-parameter array and the finite rectangle requests from which Pringsheim reads are admitted.*
2. *S is the positive rectangle partial-sum row. It lists the displayed positive rectangular sums, successor-edge additions, and finite positivity witnesses visible to the packet.*
3. *T is the diagonal and cofinal tail-budget row. It binds each tolerance request to a diagonal entry point and to cofinal rectangle enlargements beyond that entry point.*
4. *M is the* $\mathsf{MooreOsgood}^{\uparrow}$ *handoff row. It records the iterated-limit route available when a consumer asks to compare the Pringsheim rectangle read with an iterated regular read.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It stores the finite rectangle windows, index bounds, and inspection order used by X, S, T, M.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It reads the same rectangle partial sums and cofinal tail budgets as a regular rational sequence before any endpoint seal is exported.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It stores dyadic tolerances, slack radii, denominator bounds, rectangle-side budgets, and accumulated arithmetic error.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after the source, positive rectangle sums, cofinal tail budget, Moore-Osgood handoff, StreamName windows, RegSeqRat readback, and DyadicRatCore ledger have been displayed.*
9. *H stores componentwise* hsame *transport for the source, rectangle sums, tail budget, handoff, windows, readback, tolerance ledger, real seal, continuation, provenance, and local naming row.*
10. *C stores displayed* Cont *replay from rectangle inspection to positive partial sums, from those sums to cofinal tail budgets, from the budgets to the Moore-Osgood and RegSeqRat handoffs, and from the handoffs to the real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{PringsheimConvergence}^{\uparrow}$, $\mathsf{CauchyDoubleSequence}^{\uparrow}$, $\mathsf{MooreOsgood}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{PringsheimConvergence}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, T, M, W, R, D, E, H, C, P, N$. It has no coordinate for host double-series convergence, quotient equality of arrays, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or a standard bridge.*

Theorem 6261.2 (Pringsheim convergence NameCert obligations)**.** *For every accepted* $\mathsf{PringsheimConvergence}^{\uparrow}$ *carrier $K = (X, S, T, M, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PringsheimConvergence}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the displayed CauchyDoubleSequence source, positive rectangle partial sums, diagonal and cofinal tail budget, Moore-Osgood handoff, StreamName windows, regular rational readback, dyadic tolerance ledger, real seal, transport, replay, provenance, and local naming row;*
2. *rectangle-window exactness, requiring every inspected two-index term and every positive rectangle partial sum to pass through X, S, W before a tail request is admitted;*
3. *cofinal-tail exactness, requiring T to display the diagonal entry point, rectangle enlargement rule, dyadic tolerance request, and cofinal threshold consumed by the Pringsheim route;*
4. *iterated-limit compatibility, requiring M to expose only the Moore-Osgood handoff already routed through the same windows and tail ledger;*
5. $\mathsf{RegSeqRat}^{\uparrow}$ *readback, requiring R to read the same positive rectangle sums and cofinal budgets before any* $\mathsf{Real}^{\uparrow}$ *seal is available;*
6. *real-seal nonescape, requiring E to remain downstream of X, S, T, M, W, R, D and to export no host double-series convergence, quotient array equality, selected infinite sum, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge datum;*
7. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Pringsheim convergence obligation is stored outside these rows.

Lean4 ✓ `BEDC.Derived.PringsheimConvergenceUp.TasteGate.PringsheimConvergenceTasteGate single carrier alignment`

Proof. Project the accepted packet $K = (X, S, T, M, W, R, D, E, H, C, P, N)$. The carrier definition exposes exactly the CauchyDoubleSequence source, positive rectangle partial sums, diagonal and cofinal tail budget, Moore-Osgood handoff, StreamName windows, regular readback, dyadic ledger, seal, and structural rows.

The rectangle data are read first. The source row X and window row W supply only displayed finite two-parameter windows, while S supplies only positive rectangle partial sums and their visible successor-edge additions.

The tail and handoff route then passes through T, M, R, D. The row T records the diagonal entry point and cofinal enlargement budget, M records the Moore-Osgood comparison surface, R performs the regular rational readback, and D stores the dyadic tolerances consumed by both handoffs.

Finally E, H, C, P, N seal, transport, replay, source, and name the same displayed route. Since no excluded analytic, quotient, choice, verification, or bridge object appears as a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6261.3 (Pringsheim convergence iterated-limit route)**.** *Let $K = (X, S, T, M, W, R, D, E, H, C, P, N)$ be an accepted* $\mathsf{PringsheimConvergence}^{\uparrow}$ *carrier. Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer that compares the Pringsheim rectangle read with an iterated-limit read factors through*

$$X, S, W \longmapsto T, D \longmapsto M, R \longmapsto E, H, C, P, N.$$

Thus the comparison uses the displayed CauchyDoubleSequence source, positive rectangle partial sums, StreamName windows, diagonal and cofinal tail budgets, DyadicRatCore tolerance ledger, Moore-Osgood handoff, RegSeqRat readback, RealUp seal, transport, replay, provenance, and local naming. It cannot be replaced by host double-series convergence, quotient equality of arrays, a selected infinite sum, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with a consumer comparing the Pringsheim rectangle read with an iterated-limit read. By Definition 6261.1, the only visible two-parameter source data are X and the finite windows W, and the only visible rectangle-sum data are the positive partial sums in S.

The comparison cannot reach an iterated route before the tail budget is displayed. The diagonal and cofinal row T names the finite entry point and allowed rectangle enlargements, while D accounts for the dyadic tolerances and accumulated arithmetic error.

The Moore-Osgood row M and regular readback row R then expose the comparison surface. Both rows read the same windows and budgets already displayed by X, S, T, W, D, so neither row can choose a separate array representative, limiting value, or external completeness principle.

The route reaches a real-facing consumer only through E and is preserved by H, C, P, N. Therefore every such comparison factors through the displayed packet route and has no coordinate for the excluded host, quotient, choice, or completeness data. □

Theorem 6261.4 (Pringsheim convergence real seal nonescape)**.** *In every accepted* $\mathsf{PringsheimConvergence}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row E is downstream of the CauchyDoubleSequence source, positive rectangle partial sums, diagonal and cofinal tail budget, Moore-Osgood handoff, StreamName windows, RegSeqRat readback, and DyadicRatCore tolerance ledger. Consequently a Real-facing consumer cannot obtain from the packet a host double-series convergence theorem, quotient equality of arrays, selected infinite sum, countable choice, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge.*

Proof. Read the carrier order in Definition 6261.1. The seal row E is exposed only after X, S, T, M, W, R, D have supplied the source, positive rectangle sums, cofinal tail budget, iterated handoff, finite windows, regular readback, and dyadic tolerance data.

Apply Definition 6261.2. The real-seal nonescape clause binds E to that same displayed route, and the transport, replay, provenance, and local naming rows preserve the route without adding any new analytic coordinate.

Thus a Real-facing consumer receives only the sealed endpoint of the finite certificate. There is no row from which the consumer could read host double-series convergence, quotient array equality, a selected infinite sum, choice, ambient completeness, Lean verification, or bridge data. □

Falsifiable prediction: Every $\mathsf{PringsheimConvergence}^{\uparrow}$ Real-facing read factors through a CauchyDoubleSequence source, positive rectangle partial sums, StreamName windows, diagonal and cofinal tail budgets, a DyadicRatCore tolerance ledger, Moore-Osgood and RegSeqRat handoffs, a RealUp seal, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: cauchydoublesequence, moore-osgood, streamname, regseqrat, dyadicratcore, real: $\mathsf{PringsheimConvergence}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a two-parameter source, positive rectangle partial sums, cofinal tail budgets, iterated-limit handoff, regular readback, dyadic tolerance accounting, real sealing, transport, continuation, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PringsheimConvergence}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PringsheimConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6261.1, Definition 6261.2, Definition 6261.3, and Definition 6261.4 seed the finite constructive Pringsheim two-parameter convergence route from positive rectangle partial sums through cofinal tail budgets to RealUp sealing.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PringsheimConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host double-series convergence, quotient equality of arrays, selected infinite sums, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rectangle-window exactness, cofinal-tail exactness, Moore-Osgood handoff compatibility, RegSeqRat readback, Real seal nonescape, and structural transport for BEDC.Derived.PringsheimConvergenceUp.

Constructive story: A $\mathsf{PringsheimConvergence}^{\uparrow}$ packet is generated from a CauchyDoubleSequence source, positive rectangle partial sums, a diagonal and cofinal tail-budget row, a Moore-Osgood handoff row, StreamName windows, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a RealUp seal row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6262

A Concrete Naming Certificate for $\mathsf{OscillationRankFunction}^{\uparrow}$

Following Baire and Kechris, the $\mathsf{OscillationRankFunction}^{\uparrow}$ packet records the finite certificate surface by which a Baire-one name exposes a displayed oscillation-rank comparison at a named stage. It reads Baire-one value-window data from chapter 1604, local oscillation rows from chapter 3899, pointwise oscillation bounds from chapter 6136, pointwise-limit provenance from chapter 2314, and derivative-stage bookkeeping from chapter 2024. The packet names the finite rank-comparison ledger needed by downstream descriptive-analysis consumers; it is not a completed transfinite hierarchy and not a host quotient of function spaces.

Definition 6262.1 (Oscillation rank function carrier)**.** *An* $\mathsf{OscillationRankFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (F, O, B, L, D, S, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{BaireOneFunction}^{\uparrow}$ *source row, including the pointwise-limit ledger and value-window provenance of the named function.*
2. *O is the* $\mathsf{OscillationFunction}^{\uparrow}$ *row recording the local oscillation predicate whose threshold is being inspected.*
3. *B is the* $\mathsf{BairePointwiseOscillation}^{\uparrow}$ *bound row at the displayed source points and rational tolerance.*
4. *L is the* $\mathsf{PointwiseLimit}^{\uparrow}$ *lineage row that ties the approximant windows used by F and B to the same named source.*
5. *D is the finite* $\mathsf{CantorBendixson}^{\uparrow}$ *derivative-stage ledger used as the displayed rank stage, including the current derivative row and its predecessor window.*
6. *S is the rank-successor comparison row saying whether the named oscillation set survives to the displayed derivative stage.*
7. *Q is the rational threshold readback row for the oscillation bound and the displayed comparison tolerance.*
8. *H records componentwise* hsame *transport for the function source, oscillation predicate, pointwise bound, limit lineage, derivative stage, rank comparison, rational threshold, continuation, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from a Baire-one name and tolerance request through the oscillation windows, derivative-stage ledger, rank comparison, and downstream handoff surface.*
10. *P is the* Pkg *provenance over* $\mathsf{OscillationRankFunction}^{\uparrow}$, $\mathsf{BaireOneFunction}^{\uparrow}$, $\mathsf{OscillationFunction}^{\uparrow}$, $\mathsf{BairePointwiseOscillation}^{\uparrow}$, $\mathsf{PointwiseLimit}^{\uparrow}$, $\mathsf{CantorBendixson}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{OscillationRankFunction}^{\uparrow}}$ *row.*

The carrier exports no completed transfinite hierarchy, no host function-space quotient, no arbitrary choice of ranks, and no ambient descriptive-set-theory theorem.

Lean4 (definition) `BEDC.Derived.OscillationRankFunctionUp.OscillationRankFunctionUp`

Theorem 6262.2 (Oscillation rank function NameCert obligations)**.** *For every accepted* $\mathsf{OscillationRankFunction}^{\uparrow}$ *carrier* $R = (F, O, B, L, D, S, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{OscillationRankFunction}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; Baire-one source compatibility for F; oscillation-threshold compatibility for O, B, Q; pointwise-limit lineage compatibility for L; finite derivative-stage admission for D; rank-successor comparison exactness for S; transport stability through H; continuation replay through C; provenance and local naming through P, N; and ledger refusal for completed transfinite hierarchies, host function-space quotients, arbitrary rank selectors, and ambient descriptive-set-theory theorems.*

Proof. Project the carrier of Definition 6262.1. The accepted packet exposes only the rows $F, O, B, L, D, S, Q, H, C$, so each local obligation must be read from those coordinates.

The source and oscillation obligations are the routes through F, O, B, Q. The Baire-one source fixes the value-window provenance, the oscillation row fixes the inspected predicate, the pointwise bound supplies the local comparison data, and the rational threshold readback fixes the displayed tolerance.

The rank-stage obligation is the route through L, D, S. The pointwise-limit lineage keeps the approximant windows attached to the same source, the Cantor–Bendixson row supplies only a displayed finite derivative stage, and the successor-comparison row records the survival test at that stage.

Transport and replay do not add coordinates. The rows H and C preserve and replay the same source, oscillation, lineage, stage, threshold, and comparison data, while P and N source and name the packet. Since no row stores a completed hierarchy, function-space quotient, arbitrary selector, or ambient theorem, the listed obligations exhaust the certificate. □

Theorem 6262.3 (Baire-one oscillation-rank handoff)**.** *Every Baire-one consumer row exported by an accepted* $\mathsf{OscillationRankFunction}^{\uparrow}$ *carrier factors through the displayed route*

$$F, O, B, L, D, S, Q \longmapsto H, C, P, N.$$

Thus a downstream consumer may read the named Baire-one function, the local oscillation predicate, the pointwise oscillation bound, the pointwise-limit lineage, the displayed Cantor–Bendixson derivative stage, and the finite rank comparison, but it receives no completed transfinite hierarchy, no host function-space quotient, no arbitrary choice of ranks, and no ambient descriptive-set-theory theorem.

Proof. Start with a consumer of the rank-comparison row. By carrier admission, the row S is available only after the Baire-one source F, oscillation row O, pointwise bound B, pointwise-limit lineage L, derivative-stage row D, and threshold readback Q have been displayed.

Replay the continuation row C. The request first opens the Baire-one value-window provenance, then reads the oscillation predicate and pointwise bound, then checks the finite derivative-stage ledger, and only then exposes the rank comparison.

The handoff is therefore a finite certificate route. Componentwise transport through H preserves the displayed coordinates, and P, N source and name only those coordinates. A downstream consumer cannot recover any excluded hierarchy, quotient, selector, or ambient theorem from the packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{OscillationRankFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OscillationRankFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6262.1, Definition 6262.2, and Definition 6262.3 seed the finite Baire-one oscillation-rank certificate from displayed oscillation windows and Cantor–Bendixson stage data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OscillationRankFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close a completed transfinite hierarchy, host function-space quotient, arbitrary choice of ranks, ambient descriptive-set-theory theorems, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, Baire-one source compatibility, oscillation-threshold compatibility, pointwise-limit lineage compatibility, derivative-stage admission, rank comparison exactness, transport, replay, provenance, and local naming for BEDC.Derived.OscillationRankFunctionUp.

Constructive story: An $\mathsf{OscillationRankFunction}^{\uparrow}$ packet is generated from BaireOneFunction, OscillationFunction, BairePointwiseOscillation, PointwiseLimit, CantorBendixson, RatUp, transport, continuation, provenance, and local NameCert rows.

Chapter 6263

A Concrete Naming Certificate for $\mathsf{CompactCoverNerve}^{\uparrow}$

Following Bishop–Bridges compactness and the finite nerve construction, the $\mathsf{CompactCoverNerve}^{\uparrow}$ packet records the finite cover-member and overlap ledger behind the compact-uniform-continuity proof route of chapter 9237. It reads compact metric finite-net data from chapter 129, continuous-map source rows from chapter 130, rational lower-bound folds from chapter 4096, finite-net minimum folds from chapter 619, and the downstream tolerance-only consumer from chapter 369. The packet names the finite nerve object and its rational-fold handoff; it does not assert open-cover compactness, an external Lebesgue number selector, or host quotient equality of metric points.

Definition 6263.1 (Compact cover nerve carrier)**.** *A* $\mathsf{CompactCoverNerve}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (K, U, O, R, F, H, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *finite-net source row, including the compact metric carrier, finite probe spine, and located coverage witnesses.*
2. *U is the* $\mathsf{ProbeBundle}$*-indexed cover-member ledger assigning each displayed probe to the compact source row and the local continuity radius it carries.*
3. *O is the overlap and nerve-incidence row recording which displayed cover members have certified nonempty overlap or certified adjacency in the finite nerve.*
4. *R is the* $\mathsf{Rat}^{\uparrow}$ *lower-bound readback row for the positive radius data attached to the cover members and overlap witnesses.*
5. *F is the* $\mathsf{FiniteNetMinimumFold}^{\uparrow}/\mathsf{UniformModulus}^{\uparrow}$ *handoff row, recording the finite fold that exports a tolerance-only uniform-radius ledger.*
6. *H records componentwise* hsame *transport for the compact source, cover-member ledger, overlap incidence, rational readback, fold handoff, provenance, and local naming rows.*
7. *P is the* Pkg *provenance over* $\mathsf{CompactCoverNerve}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteNetMinimumFold}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{ProbeBundle}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCoverNerve}^{\uparrow}}$ *row.*

The carrier exports no external open-cover choice, no host nerve complex, no quotient equality of metric points, no arbitrary Lebesgue number, and no ambient compactness theorem beyond the displayed compact metric source row.

Theorem 6263.2 (Compact cover nerve NameCert obligations)**.** *For every accepted* $\mathsf{CompactCoverNerve}^{\uparrow}$ *carrier* $C = (K, U, O, R, F, H, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactCoverNerve}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows; compact-source compatibility for* K, U*; finite cover-member admission for the* ProbeBundle *ledger* U*; overlap-incidence exactness for* O*; rational lower-bound readback for* R*; uniform-modulus handoff exactness for* F*; transport stability through* H*; provenance and local naming through* P, N*; and ledger refusal for external open-cover choice, host nerve complexes, quotient equality of points, arbitrary Lebesgue numbers, and ambient compactness theorems.*

Lean4 ✓ `BEDC.Derived.CompactCoverNerveUp.TasteGate.CompactCoverNerveTasteGate single carrier alignment`

Proof. Project the carrier of Definition 6263.1. The accepted packet exposes only K, U, O, R, F, H, P, N, so every local obligation must be read from these rows.

Compact-source compatibility and cover-member admission are the route through K and U. The compact metric row fixes the finite probe spine, and the cover-member ledger attaches each cover row to that same displayed source.

Overlap-incidence exactness is the route through O. The finite nerve records only displayed overlap or adjacency evidence among members of U, so it cannot introduce a host nerve complex detached from the packet.

The rational fold and uniform-modulus handoff are the route through R and F. The lower-bound readback consumes the displayed radius data, and the fold handoff exports only the tolerance-only radius row available to $\mathsf{UniformModulus}^{\uparrow}$ consumers.

Transport, provenance, and naming through H, P, N preserve the same coordinates. Since no row stores external cover choice, quotient point equality, arbitrary Lebesgue-number selection, or ambient compactness, the listed obligations exhaust the certificate. □

Theorem 6263.3 (Compact cover nerve uniform-modulus handoff)**.** *Every* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row exported by an accepted* $\mathsf{CompactCoverNerve}^{\uparrow}$ *carrier factors through the displayed finite route*

$$K, U, O, R, F \longmapsto H, P, N.$$

The consumer receives compact metric finite-net data, cover-member rows, overlap incidence, rational lower-bound readback, and finite-fold handoff rows, but it receives no external open cover, no host nerve complex, no quotient equality of metric points, no arbitrary Lebesgue number, and no ambient compactness theorem.

Proof. Start with a uniform-modulus consumer of the fold handoff row F. Carrier admission makes F downstream of the compact source K, the cover-member ledger U, the overlap-incidence row O, and the rational readback row R.

Replay the finite route. The compact source supplies the finite probe spine, U attaches cover members and local radii to that spine, O records the finite incidence data, and R reads back the positive rational lower-bound data before F exports the uniform-radius handoff.

The structural rows do not widen the export surface. Transport through H preserves the same finite coordinates, while P, N source and name the packet. Thus every downstream $\mathsf{UniformModulus}^{\uparrow}$ read remains on the displayed cover-nerve route and cannot recover any excluded host object or selector. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCoverNerve}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactCoverNerve}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6263.1, Definition 6263.2, and Definition 6263.3 seed the finite cover nerve from CompactMetric cover-member rows to rational-fold and UniformModulus handoff rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactCoverNerve}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close external open-cover choice, host nerve complexes, quotient equality of metric points, arbitrary Lebesgue numbers, ambient compactness theorems, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked compact-source compatibility, cover-member admission, overlap-incidence exactness, rational readback, uniform-modulus handoff, transport, provenance, and local naming for BEDC.Derived.CompactCoverNe

Constructive story: A $\mathsf{CompactCoverNerve}^{\uparrow}$ packet is generated from CompactMetric finite-net data, ProbeBundle cover-member rows, finite overlap incidence, RatUp lower-bound readback, FiniteNetMinimumFold and UniformModulus handoff rows, transport, provenance, and local NameCert rows.

Chapter 6264

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchy}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Kelley compact-metric terminology, the $\mathsf{CompactUniformCauchy}^{\uparrow}$ packet records the finite route by which compact finite nets, uniform Cauchy tails, regular rational readback, dyadic tolerances, and the terminal $\mathsf{Real}^{\uparrow}$ seal are made into NameCert data. It sits after the compact metric source of chapter 129, the uniform Cauchy criterion of chapter 435, the uniform-limit handoff of chapter 463, the finite stream interface of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the real-name seal of chapter 38. It realizes the compact finite-net pressure of chapter 9237 without importing ambient real completeness.

Definition 6264.1 (CompactUniformCauchy carrier)**.** *An accepted* $\mathsf{CompactUniformCauchy}^{\uparrow}$ *carrier displays the finite* BHist *packet*

$$K = (K, F, W, R, D, U, E, H, C, P, N)$$

with the following visible coordinates:

1. *K is the compact metric source row, read as finite net data rather than as an open-cover compactness principle;*
2. *F is the family or stream of observed real-valued rows over the compact source;*
3. *W is the finite tail-window row, naming the indices and windows exposed to a uniform-Cauchy consumer;*
4. *R is the regular rational readback row through* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*;*
5. *D is the dyadic tolerance row supplied by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
6. *U is the uniform Cauchy tail row, recording one displayed tail bound for the whole compact finite net;*
7. *E is the downstream* $\mathsf{Real}^{\uparrow}$ *seal row reached after the finite tail and tolerance readback;*
8. *H stores the componentwise* hsame *and* psame *transports for the compact, tail, tolerance, uniformity, seal, provenance, and naming rows;*
9. *C stores the finite* Cont *replay routes by which a consumer reads the compact net, uniform tail, regular readback, dyadic tolerance, and Real seal;*

10. *P records* Pkg *provenance over the cited compact, uniform-Cauchy, uniform-limit, stream, regular-rational, dyadic, and real-name surfaces;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchy}^{\uparrow}}$ *row.*

The carrier contains no hidden coordinate for an open-cover choice, quotient stream, selected limit, host real equality, countable-choice selector, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Lean4 (definition) `BEDC.Derived.CompactUniformCauchyUp.CompactUniformCauchyUp`

Theorem 6264.2 (CompactUniformCauchy NameCert obligations)**.** *For every accepted carrier Q displaying the packet $(K, F, W, R, D, U, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchy}^{\uparrow}}$ *surface consists of finite obligations for compact-net exposure, family-row exposure, uniform tail-window exactness, regular rational readback, dyadic tolerance exactness, Real seal nonescape, componentwise transport, route replay, package provenance, and local naming. Each obligation is read from the displayed coordinates $K, F, W, R, D, U, E, H, C, P, N$.*

Proof. Project the carrier coordinates. The compact row, family row, tail window, regular readback, dyadic tolerance, uniform tail, and Real seal are all explicit entries of the finite packet.

The transport and replay rows H and C determine how those entries move between certified presentations. They expose only componentwise sameness and displayed continuation routes, so a consumer cannot insert an open-cover choice, quotient stream, selected limit, or host equality proof.

The provenance and local naming rows P, N bind the packet to the cited sibling chapters. Repacking these rows gives exactly the stated finite NameCert obligation surface. □

Theorem 6264.3 (CompactUniformCauchy finite-net extraction)**.** *Every uniform-Cauchy read of an accepted* $\mathsf{CompactUniformCauchy}^{\uparrow}$ *carrier first extracts the compact finite-net source row K, then reads the family row F, the tail-window row W, the regular readback row R, the dyadic tolerance row D, and the uniform tail row U through H, C, P, N. The extraction exports finite net and tail data only; it does not export open-cover compactness, selected subcovers outside the displayed packet, countable choice, quotient point equality, or a host uniform modulus.*

Proof. Start with the compact source coordinate K. By definition it is the displayed compact-metric finite-net row, not an ambient compactness principle.

Next read the analytic rows F, W, R, D, U. These rows move from observed family data to finite tail windows, regular rational readback, dyadic tolerance, and one uniform tail bound over the displayed finite net.

Finally use H, C, P, N to transport, replay, source, and name that same route. Since no other coordinate is present, finite-net extraction is exhausted by the packet and cannot supply the excluded choice, quotient, subcover, equality, or host-modulus data. □

Theorem 6264.4 (CompactUniformCauchy Real seal nonescape)**.** *In every accepted* $\mathsf{CompactUniformCauchy}^{\uparrow}$ *carrier, the* $\mathsf{Real}^{\uparrow}$ *seal row E is reachable only after the compact finite-net row K, the family row F, the tail-window row W, the regular readback row R, the dyadic tolerance row D, and the uniform tail row U have been read through the displayed transport and replay rows. A downstream consumer may use E as the terminal seal of this finite route, but not as ambient Real completeness, a selected limit, host equality of completed reals, or a quotient-stream representative.*

Proof. Project the terminal row E from the carrier. It is named as the downstream Real seal coordinate and appears after the compact-net, family, tail-window, regular-readback, dyadic-tolerance, and uniform-tail coordinates.

The only routes to E are the displayed continuation routes in C transported by H and sourced by P, N. Those routes force the consumer to pass through K, F, W, R, D, U before the seal can be read.

Therefore the seal is a terminal row of the finite packet. It cannot be detached into a completeness theorem, selected limit, host real equality, or quotient-stream representative. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchy}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCauchy}^{\uparrow}$ packet is generated from compact finite-net data, observed family rows, tail windows, regular rational readback, dyadic tolerances, uniform Cauchy tails, a terminal Real seal, componentwise transports, displayed continuation routes, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6264.1, Definition 6264.2, Definition 6264.3, and Definition 6264.4 record the finite carrier, NameCert obligations, finite-net extraction, and Real-seal nonescape rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: human

Not claimed: This chapter does not claim ambient Real completeness, quotient-stream completion, selected limits, open-cover compactness, countable choice, host real equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a Lean carrier encoding and checked NameCert obligation theorem for the compact-net, uniform-tail, regular-readback, dyadic-tolerance, and Real-seal rows.

Chapter 6265

A Concrete Naming Certificate for $\mathsf{CompleteRealModulus}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{CompleteRealModulus}^{\uparrow}$ packet records the finite modulus data by which a completion-facing real read is accepted without appealing to ambient completeness. It sits after the Cauchy-filter limit route of chapter 1558, the regular-rational sequence discipline of chapter 301, the stream-window naming surface of chapter 126, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. It also shares its modulus-facing boundary with chapter 1423: the point is not to add a new completion principle, but to name the displayed modulus route that lets completion consumers read a Real-facing limit through finite windows and tolerances.

Definition 6265.1 (CompleteRealModulus carrier)**.** *A* $\mathsf{CompleteRealModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (F, S, D, R, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{CauchyFilterLimit}^{\uparrow}$ *source row naming the Cauchy-filter data whose completion read is being inspected.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to expose the inspected stream segments.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row recording the displayed dyadic budget for the read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row reached after the stream windows and dyadic tolerances are displayed.*
5. *Q is the complete-modulus row, recording the finite rule that sends each requested output tolerance to the displayed filter window, stream segment, and regular-rational comparison needed for the read.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after F, S, D, R, Q have supplied the displayed route.*
7. *H records componentwise* hsame *transport for the Cauchy-filter source, stream windows, dyadic tolerances, regular readback, modulus row, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a completion-modulus request through $F, S, D, R, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{CompleteRealModulus}^{\uparrow}$, $\mathsf{CauchyFilterLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{CompleteRealModulus}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $F, S, D, R, Q, E, H, C, P, N$. It has no coordinate for a selected limit, quotient filter, host equality of reals, countable choice, or an ambient completeness theorem.

Lean4 ✓ `BEDC.Derived.CompleteRealModulusUp.CompleteRealModulusTasteGate single carrier alignment`

Theorem 6265.2 (CompleteRealModulus NameCert obligations)**.** *For every accepted* $\mathsf{CompleteRealModulus}^{\uparrow}$ *carrier*

$$M = (F, S, D, R, Q, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompleteRealModulus}^{\uparrow}}$ *surface consists exactly of carrier admission for M, Cauchy-filter source exposure through F, stream-window exposure through S, dyadic tolerance exposure through D, regular-rational readback through R, complete-modulus accountability through Q, terminal Real sealing through E, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6265.1. The analytic source is not an ambient completion object; it is the displayed Cauchy-filter row F together with finite stream windows S and dyadic tolerances D.

The readback rows are ordered. The regular-rational row R is reached only after the relevant stream windows and dyadic comparisons are present, and the modulus row Q records how each requested tolerance is answered by those displayed rows.

The Real-facing row E is terminal for this packet. It seals the displayed completion read after F, S, D, R, Q have been exposed, while H, C, P, N transport, replay, source, and name the same route.

Since no row stores a selected limit, quotient filter, host equality, countable choice schedule, or ambient completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6265.3 (CompleteRealModulus threshold seal route)**.** *Every Real-facing consumer of an accepted* $\mathsf{CompleteRealModulus}^{\uparrow}$ *packet must read a completion threshold through the finite route*

$$\mathsf{CauchyFilterLimit}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by F, S, D, R, Q, E. In particular, the terminal seal is available only after the Cauchy-filter source, stream windows, dyadic tolerance row, regular-rational readback, complete-modulus row, componentwise transport, continuation replay, provenance, and local NameCert row have been displayed.

Proof. Start with an accepted carrier $M = (F, S, D, R, Q, E, H, C, P, N)$. The source row F names the Cauchy-filter limit request, so the consumer has no detached completion carrier to inspect.

The finite observation route then passes through S, D, R. Stream windows expose the inspected name segments, dyadic tolerances record the requested threshold, and regular-rational readback presents the comparison data in a form that can face the Real seal.

The complete-modulus row Q binds these observations to every requested output tolerance. The terminal row E can therefore be read only after the threshold route is displayed, and the structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming.

Thus a Real-facing completion consumer receives a finite threshold-seal packet, not a selected limit, quotient filter, host equality, countable choice principle, or ambient completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{CompleteRealModulus}^{\uparrow}$ completion-threshold read factors through one CauchyFilterLimitUp source row, one StreamNameUp finite-window row, one DyadicRatCoreUp tolerance row, one RegSeqRatUp readback row, one complete-modulus row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: realcauchymodulus, cauchyfilterlimit, regseqrat, streamname, dyadicratcore, real: $\mathsf{CompleteRealModulus}^{\uparrow}$ is not bijective to any listed sibling because it packages the completion-threshold rule, source filter row, finite windows, dyadic tolerances, regular readback, terminal Real seal, transport, replay, provenance, and local NameCert as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{CompleteRealModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompleteRealModulus}^{\uparrow}$ packet is generated from a CauchyFilterLimitUp source row, StreamNameUp finite windows, DyadicRatCoreUp tolerances, RegSeqRatUp readback, a complete-modulus row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompleteRealModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6265.1, Definition 6265.2, and Definition 6265.3 seed the finite completion-threshold route from Cauchy-filter source data to regular-rational readback and the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompleteRealModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient completion, selected limits, host equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-filter source exposure, stream-window exposure, dyadic tolerance exposure, regular-rational readback, modulus accountability, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompleteRealModulusUp.

Chapter 6266

A Concrete Naming Certificate for $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ packet records the finite tail-filter basis generated by a regular Cauchy rational schedule before a completion consumer reads a $\mathsf{Real}^{\uparrow}$ seal. It sits after the regular rational sequence of chapter 301, the finite stream windows of chapter 126, the Cauchy filter-basis surface of chapter 1501, and the dyadic tolerance ledger of chapter 291. The packet names only displayed tail-filter basis rows; it does not assert quotient completion equality, selected limits, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ equality.

Definition 6266.1 (Regular Cauchy tail-filter basis carrier)**.** *A* $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (R, W, B, D, E, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row. It records the regular rational schedule and the explicit tail bounds that generate the basis.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It lists the finite observation windows, requested indices, and inspection order used to read the schedule.*
3. *B is the* $\mathsf{CauchyFilterBasis}^{\uparrow}$ *basis row. It stores the finite family of tail neighborhoods generated from the regular Cauchy schedule.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It records dyadic radii, denominator budgets, slack comparisons, and the tail-to-radius conversion used by the basis.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *endpoint seal row. It is visible only after R, W, B, D have displayed the regular schedule, finite windows, basis neighborhoods, and tolerance comparisons.*
6. *H records componentwise* hsame *transport for the source schedule, windows, basis, dyadic ledger, endpoint seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from regular Cauchy tail requests to finite filter-basis neighborhoods and then to the endpoint seal.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyFilterBasis}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $R, W, B, D, E, H, C, P, N$. There is no carrier coordinate for quotient completion equality, a selected limiting real, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ *equality.*

Lean4 ✓ `BEDC.Derived.RegularCauchyTailFilterBasisUp.RegularCauchyTailFilterBasisTasteGate`
`single carrier alignment`

Theorem 6266.2 (Regular Cauchy tail-filter basis NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ *carrier $T = (R, W, B, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the regular rational source, stream windows, Cauchy filter-basis row, dyadic tolerance ledger, endpoint seal, transport, replay, provenance, and local naming rows;*
2. *tail-window exactness, requiring every basis neighborhood to name the finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *tail request from which it was generated;*
3. *dyadic-radius exactness, requiring D to display the radius and slack comparison consumed by each tail neighborhood in B;*
4. *endpoint-seal nonescape, requiring E to remain downstream of R, W, B, D and to export no quotient completion equality, selected limiting real, countable-choice schedule, host completeness, or ambient* $\mathsf{Real}^{\uparrow}$ *equality;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No regular Cauchy tail-filter basis obligation is stored outside these rows.

Proof. Project the accepted packet $T = (R, W, B, D, E, H, C, P, N)$. The carrier definition exposes exactly the regular rational source, finite windows, basis neighborhoods, dyadic ledger, endpoint seal, and structural rows.

The basis route starts with R and W. A tail neighborhood cannot be read until the regular Cauchy tail request and its finite observation window have both been displayed.

The dyadic comparison is then read through D. This row supplies the radius, denominator budget, and slack comparison used by the basis row B, so a filter-basis consumer receives a finite tail-to-radius certificate rather than a host completeness argument.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for quotient equality, selected limits, countable choice, host completeness, or ambient real equality, the listed rows exhaust the local NameCert surface. □

Theorem 6266.3 (Regular Cauchy tail-filter endpoint handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ *carrier factors through*

$$\mathsf{RegSeqRat}^{\uparrow}, \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{CauchyFilterBasis}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}$$

Thus the endpoint seal is consumed only after the regular Cauchy schedule, finite windows, tail-filter basis neighborhoods, dyadic comparisons, transport, replay, provenance, and local naming have been displayed.

Proof. Begin with a completion-facing consumer of $T = (R, W, B, D, E, H, C, P, N)$. The only source rows available to the consumer are the regular Cauchy schedule R and the finite windows W.

The filter-basis handoff then passes through B and D. The row B records the tail neighborhoods generated by the schedule, while D records the dyadic radii and slack comparisons that make those neighborhoods usable by completion data.

The endpoint row E is downstream of this displayed basis route. The structural rows H, C, P, N preserve the same finite packet, so the consumer cannot replace the route by quotient completion equality, a selected limit, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ equality. □

Falsifiable prediction: Every $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ completion-facing read factors through a RegSeqRat source, StreamName windows, CauchyFilterBasis tail neighborhoods, a DyadicRatCore tolerance ledger, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, cauchyfilterbasis, dyadicratcore, real: RegularCauchyTailFilterBas is not bijective to any listed sibling because it jointly carries regular Cauchy schedules, finite stream windows, generated tail-filter basis neighborhoods, dyadic tolerance comparisons, endpoint sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6266.1, Definition 6266.2, and Definition 6266.3 seed the finite route from regular Cauchy schedules to tail-filter basis completion data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected limits, countable choice, host completeness, ambient $\mathsf{Real}^{\uparrow}$ equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exactness, dyadic-radius exactness, endpoint-seal nonescape, componentwise transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyTailFilterBasisUp.

Constructive story: A $\mathsf{RegularCauchyTailFilterBasis}^{\uparrow}$ packet is generated from a RegSeqRat regular Cauchy schedule, StreamName windows, CauchyFilterBasis tail neighborhoods, a DyadicRatCore tolerance ledger, a RealUp endpoint seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6267

A Concrete Naming Certificate for $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$

Following Poincare disk geometry and Bishop–Bridges finite-data discipline, the $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ packet records boundary-preserving disk transport as displayed BEDC rows rather than as a host analytic automorphism of the disk. It concretizes the Poincare disk reading of section 9463.3 after the disk-coordinate surface of chapter 377, the disk metric packet of chapter 6209, the hyperbolic metric readback of chapter 3474, and the boundary-fixed transport surface of chapter 3500. The packet names only finite disk, boundary, metric, transport, provenance, and NameCert rows; it does not claim a completed disk, quotient ideal-boundary equality, selected boundary point, host Mobius action, or global compactification.

Definition 6267.1 (Poincare disk boundary transport carrier)**.** *A* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, B, M, T, F, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{UnitDisk}^{\uparrow}$ *disk-coordinate row, recording the finite source and target disk observations inspected by the transport.*
2. *B is the boundary row, recording the displayed boundary incidences and far-end labels preserved by the transport.*
3. *M is the metric-read row, sourced through* $\mathsf{PoincareDiskMetric}^{\uparrow}$ *and* $\mathsf{HyperbolicMetric}^{\uparrow}$ *comparisons over the same disk observations.*
4. *T is the transport row, recording the finite disk movement, boundary-preservation check, and admissible source-target route.*
5. *F is the fixed-boundary compatibility row, comparing the transport with the* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *surface when a consumer asks for boundary invariance.*
6. *H records componentwise* hsame *transport for disk coordinates, boundary rows, metric reads, disk transport, fixed-boundary compatibility, replay, provenance, and local naming.*
7. *C records displayed* Cont *replay of the route from disk observations to boundary-preserving transport and then to metric-facing consumers.*

8. *P is the* Pkg *provenance over* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$, $\mathsf{UnitDisk}^{\uparrow}$, $\mathsf{PoincareDiskMetric}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$, $\mathsf{PoincareDiskPhaseTransport}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $D, B, M, T, F, H, C, P, N$. There is no coordinate for a completed disk, quotient ideal-boundary equality, selected boundary point, host Mobius action, global compactification, Lean verification, or bridge data.

Lean4 ✓ `BEDC.Derived.PoincareDiskBoundaryTransportUp.PoincareDiskBoundaryTransportTasteGate`
`single carrier alignment`

Theorem 6267.2 (Poincare disk boundary transport NameCert obligations)**.** *For every accepted* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *carrier $G = (D, B, M, T, F, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PoincareDiskBoundaryTranspo}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for disk coordinates, boundary incidences, metric reads, disk transport, fixed-boundary compatibility, transport, replay, provenance, and local naming;*
2. *disk-boundary visibility, requiring every transport read to expose D and B before a boundary-preserving route is exported;*
3. *metric compatibility, requiring M to be sourced through the same displayed disk observations and boundary rows used by T;*
4. *fixed-boundary exactness, requiring F to compare the route with* $\mathsf{PoincareBoundaryFixedTransport}^{\uparrow}$ *without introducing a selected boundary point or host automorphism;*
5. *componentwise transport, displayed replay, provenance, and local naming for the same route through H, C, P, N.*

No Poincare disk boundary-transport obligation is stored outside these rows.

Proof. Project the accepted packet $G = (D, B, M, T, F, H, C, P, N)$. The carrier definition exposes disk coordinates, boundary incidences, metric reads, transport, fixed-boundary compatibility, and structural rows.

The boundary-preserving route starts with D and B. A transport row cannot be read as boundary-preserving until the finite disk observations and the displayed boundary incidences are both visible.

The metric and fixed-boundary checks then pass through M, T, F. The metric row reads the same disk data, the transport row records the admissible movement, and the fixed-boundary row compares that movement with the boundary-fixed transport surface.

Finally H, C, P, N preserve, replay, source, and name the same finite route. Since no carrier coordinate stores a completed disk, quotient ideal boundary, selected boundary point, host Mobius action, global compactification, Lean verification, or bridge data, the listed rows exhaust the local NameCert surface. □

Theorem 6267.3 (Poincare disk boundary transport nonescape)**.** *Every boundary-preserving disk-transport read from an accepted* $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ *packet factors through*

$$\mathsf{UnitDisk}^{\uparrow} \longmapsto \mathsf{PoincareDiskMetric}^{\uparrow}, \mathsf{HyperbolicMetric}^{\uparrow} \longmapsto \mathsf{PoincareBoundaryFixedTransport}^{\uparrow}, \mathsf{PoincareDiskPhaseTran}$$

Thus a consumer receives only finite disk, boundary, metric, transport, replay, provenance, and naming rows, not a completed disk, quotient ideal-boundary equality, selected boundary point, host Mobius action, or global compactification.

Proof. Start with the carrier rows of Definition 6267.1. The disk row D and boundary row B are the only source of visible boundary data.

The route reaches metric and transport consumers through M, T, F. These rows bind the metric read, disk movement, and fixed-boundary comparison to the same displayed source and boundary observations.

The structural rows H, C, P, N do not add a second geometric object. They preserve, replay, source, and name the displayed packet, so no completed disk, quotient ideal boundary, selected boundary point, host Mobius action, or global compactification can be exported from the accepted read. □

Falsifiable prediction: An accepted $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ read is refuted if a boundary-transport consumer obtains boundary preservation without displaying UnitDisk coordinates, boundary rows, PoincareDiskMetric and HyperbolicMetric reads, PoincareBoundaryFixedTransport compatibility, PoincareDiskPhaseTransport routing, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: unitdisk, poincare-disk-metric, hyperbolic-metric, poincare-boundary-fixed-transport, poincare-disk-phase-transport: no listed sibling is bijective to $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ because this packet jointly carries disk coordinates, boundary incidences, metric reads, boundary-preserving transport, fixed-boundary compatibility, replay, provenance, and local naming as one finite route.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6267.1, Definition 6267.2, and Definition 6267.3 seed the finite boundary-preserving Poincare disk transport route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close completed disks, quotient ideal-boundary equality, selected boundary points, host Mobius actions, global compactification, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked disk-coordinate admission, boundary visibility, metric compatibility, fixed-boundary exactness, transport preservation, replay, provenance, and local naming for BEDC.Derived.PoincareDiskBoundaryTransportUp.

Constructive story: A $\mathsf{PoincareDiskBoundaryTransport}^{\uparrow}$ packet is generated from UnitDisk coordinates, displayed boundary incidences, PoincareDiskMetric and HyperbolicMetric reads, boundary-preserving disk transport, PoincareBoundaryFixedTransport compatibility, PoincareDiskPhaseTransport routing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6268

A Concrete Naming Certificate for $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ packet records the finite Cauchy-window extraction that a constructive Real and Completion route may inspect before any compactness or limit principle is available. It sits over the dyadic tolerance surface of chapter 291, the finite schedule surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet extracts a displayed finite net row from a Cauchy window; it is not an open-cover compactness theorem, not a selected limit construction, and not a countable-choice schedule.

Definition 6268.1 (Finite Cauchy net extraction carrier)**.** *A* $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (D, W, Q, M, T, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the precision request and the dyadic comparison ledger used to judge the finite net.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the observed stream positions available at that tolerance.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row through which the scheduled window is interpreted as regular rational Cauchy data.*
4. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger recording that the chosen window is large enough for the displayed tolerance request.*
5. *T is the* $\mathsf{TotallyBounded}^{\uparrow}$ *finite-net row extracted from the displayed Cauchy window and indexed only by the entries named in W.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff exposed only after D, W, Q, M, T have been displayed.*
7. *H records componentwise* hsame *transport for the dyadic tolerance, finite-window schedule, regular rational readback, modulus ledger, finite-net row, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite Cauchy-window request through D, W, Q, M, T, E, H and into a Real or Completion consumer.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyNetExtraction}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, Q, M, T, E, H, C, P, N$. It has no coordinate for open-cover compactness, a selected Cauchy limit, quotient stream equality, countable choice, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.FiniteCauchyNetExtractionUp.FiniteCauchyNetExtractionTasteGate single carrier alignment`

Theorem 6268.2 (Finite Cauchy net extraction NameCert obligations)**.** *For every accepted* $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ *carrier $F = (D, W, Q, M, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteCauchyNetExtraction}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window schedule readback, regular rational handoff, convergence-modulus ledger exactness, displayed finite-net extraction, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$D, W, Q, M \longmapsto T \longmapsto E \longmapsto H, C, P, N$$

before using a finite net as Cauchy-window evidence.

Proof. Project the carrier of Definition 6268.1. The dyadic row D supplies the tolerance request, and the finite schedule W supplies the only stream entries that may be inspected.

Read Q and M next. The regular rational row Q interprets the scheduled entries as Cauchy data, while the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger M records that the displayed window is large enough for the request named by D.

The finite-net row T is then extracted from those displayed entries only. The Real seal E and the structural rows H, C, P, N transport, replay, source, and name the same finite route. Since no carrier coordinate supplies open-cover compactness, selected limits, quotient stream equality, countable choice, host real equality, or ambient Real completeness, none of those objects can occur in the local NameCert surface. □

Theorem 6268.3 (Finite Cauchy net extraction L10 route)**.** *Every L10 Real and Completion consumer of an accepted* $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ *packet reads the extracted net only through the finite route*

$\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, P

The route exposes the dyadic tolerance, the scheduled Cauchy window, the regular rational readback, the convergence-modulus ledger, the finite-net row, the terminal Real seal, componentwise transport, replay, provenance, and local naming. It exports no open-cover compactness, selected limit, quotient stream equality, countable-choice schedule, host real equality, detached metric-completion row, or ambient Real completeness.

Proof. Apply Definition 6268.2. It fixes the only public route from the tolerance request through the scheduled window, regular rational readback, modulus ledger, finite-net row, and Real seal.

For a Real-facing consumer, the seal E is reached only after D, W, Q, M, T have already been displayed. Thus the consumer receives the finite net as an inspected Cauchy-window witness, not as a host-selected limit or quotient representative.

For a Completion-facing consumer, the same rows are replayed through H, C, P, N. The replay row can cite the displayed $\mathsf{TotallyBounded}^{\uparrow}$ net row and the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger, but it cannot add an open-cover compactness proof, countable choice, detached metric-completion row, or ambient Real completeness, because those data are absent from the carrier. □

Falsifiable prediction: Every accepted $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ Real or Completion consumer factors through one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger, one $\mathsf{TotallyBounded}^{\uparrow}$ finite-net row, one $\mathsf{Real}^{\uparrow}$ seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it may use finite Cauchy-window evidence.

Independence witness: dyadicratcore, streamname, regseqrat, real-modulus-of-convergence, totallybounded, real: $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately supply tolerance comparisons, finite schedules, rational readback, convergence ledgers, finite nets, and terminal Real sealing, while this packet binds them to the finite Cauchy-window extraction consumed by the L10 Real and Completion route.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCauchyNetExtraction}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteCauchyNetExtraction}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, a StreamName finite-window schedule, a RegSeqRat readback row, a RealModulusOfConvergence ledger, a TotallyBounded finite-net row, a Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCauchyNetExtraction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6268.1, Definition 6268.2, and Definition 6268.3 seed the finite Cauchy-window extraction interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCauchyNetExtraction}^{\uparrow}}, \mathsf{theoremChecked})$

Lean target: `BEDC.Derived.FiniteCauchyNetExtractionUp.FiniteCauchyNetExtractionTasteGate\_single\_carrier\_alignm`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient stream equality, countable choice, host real equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window scheduling, RegSeqRat readback, convergence-modulus ledger exactness, finite-net extraction, Real seal handoff, structural transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.FiniteCauchyNetExtractionUp.

Chapter 6269

A Concrete Naming Certificate for PoincareRecurrence$^{\uparrow}$

Following Poincare's recurrence theorem and Bishop-style finite evidence, the PoincareRecurrence$^{\uparrow}$ packet records the displayed finite return-window route behind recurrence in a measure-preserving system. It sits after the finite routed-flow carrier of chapter 195, the invariant-measure surface of chapter 196, the normalized probability carrier of chapter 184, and the underlying measure packet of chapter 92. The packet names only a finite recurrence certificate; it is not a host infinite orbit theorem, not a selected recurrent point, not a sigma-completion principle, and not an ambient ergodic decomposition.

Definition 6269.1 (Poincare recurrence carrier)**.** *A* PoincareRecurrence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, E, P, M, A, W, T, H, C, Q, N)$$

with the following displayed rows:

1. *D is the* DynSystem$^{\uparrow}$ *finite routed-flow row, exposing the inspected step relation and orbit cells.*
2. *E is the* Ergodic$^{\uparrow}$ *invariant-measure or recurrence-facing row used only as finite classifier support.*
3. *P is the* ProbSpace$^{\uparrow}$ *normalization row for the total event and event classifier.*
4. *M is the* Measure$^{\uparrow}$ *measurable-set row naming the inspected positive-measure cell A.*
5. *A is the displayed source event whose nonzero measure is being tested for return.*
6. *W is the finite return-window row, listing the inspected nonzero times and their displayed inverse-image or forward-image cells.*
7. *T is the return-hit row asserting that the window contains a measured overlap between A and an allowed return of A.*
8. *H records componentwise* hsame *transport for the dynamic, invariant, probability, measure, event, window, return-hit, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay through the dynamic route, event classifier, finite window, and return-hit ledger.*
10. *Q is the* Pkg *provenance over* PoincareRecurrence$^{\uparrow}$, DynSystem$^{\uparrow}$, Ergodic$^{\uparrow}$, ProbSpace$^{\uparrow}$, Measure$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PoincareRecurrence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these eleven rows. It has no coordinate for a completed measure algebra, infinite orbit selector, countable recurrence schedule, host almost-everywhere predicate, or ergodic decomposition.

Lean4 ✓ `BEDC.Derived.PoincareRecurrenceUp.PoincareRecurrenceTasteGate single carrier alignment`

Theorem 6269.2 (Poincare recurrence NameCert obligations)**.** *For every accepted* $\mathsf{PoincareRecurrence}^{\uparrow}$ *carrier*

$$R = (D, E, P, M, A, W, T, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{PoincareRecurrence}^{\uparrow}}$ *surface consists exactly of finite routed dynamics, invariant-measure support, probability normalization, measurable-event exposure, finite return-window data, a displayed return-hit row, componentwise transport, displayed replay, package provenance, and local naming through the rows* $D, E, P, M, A, W, T, H, C, Q, N$.

Proof. Project the carrier of Definition 6269.1. The rows D, E, P, M, A expose the only dynamic, invariant, probability, measure, and source-event data visible to the packet.

Read the return evidence next. The row W gives the inspected finite return window, and T records the measured overlap that certifies a return within that displayed window.

Finally attach H, C, Q, N. Componentwise transport, continuation replay, provenance, and local naming preserve exactly the displayed rows, so the certificate cannot export an infinite orbit selector, a completed measure algebra, a countable recurrence schedule, or an ergodic decomposition. □

Theorem 6269.3 (Poincare recurrence return-window handoff)**.** *Every accepted* $\mathsf{PoincareRecurrence}^{\uparrow}$ *recurrence read factors through the route*

$$D \longmapsto E \longmapsto P \longmapsto M \longmapsto A \longmapsto W \longmapsto T \longmapsto H, C, Q, N$$

before a downstream consumer receives the return claim. Thus the handoff to later dynamic or probability formalization is a displayed finite system row, invariant-support row, probability normalization row, measurable source event, return-window ledger, return-hit witness, transport, replay, provenance, and local naming data.

Proof. Start with a consumer requesting recurrence data. The packet first exposes the finite dynamic row D, the invariant-support row E, and the normalized probability row P.

The measurable source event is then fixed by M and A. No return statement is available before these rows name the event whose positive measure is being inspected.

Only after that does the finite window W feed the return-hit row T. The structural rows H, C, Q, N preserve the same route for later consumers, so the handoff is a finite return-window certificate rather than a host recurrence theorem. □

Theorem 6269.4 (Poincare recurrence measure-preserving nonescape)**.** *Every accepted* $\mathsf{PoincareRecurrence}^{\uparrow}$ *measure-preserving read remains inside the finite carrier of Definition 6269.1. In particular, a consumer can read a recurrence claim only after displaying the* $\mathsf{DynSystem}^{\uparrow}$ *row,* $\mathsf{Ergodic}^{\uparrow}$ *invariant-support row,* $\mathsf{ProbSpace}^{\uparrow}$ *normalization row,* $\mathsf{Measure}^{\uparrow}$ *event row, finite return-window ledger, return-hit row, componentwise* hsame *transport,* Cont *replay,* Pkg *provenance, and local* NameCert *row. No accepted read may replace this route by a selected recurrent point, a completed invariant sigma-algebra, a countable-choice return schedule, a host almost-everywhere predicate, or an ambient ergodic decomposition.*

Proof. Let a consumer request a measure-preserving recurrence claim from an accepted carrier. Projection of Definition 6269.1 shows that the request first meets D, E, P, M, A and has no independent coordinate for a selected point or completed orbit.

The finite return-window row W is downstream from the displayed measurable source event, and the return-hit row T is downstream from that window. Hence the recurrence read is already tied to a finite inspected window and its measured overlap.

The remaining rows H, C, Q, N add no new analytic primitive. They transport equal displayed histories, replay the finite route, record package provenance, and attach the local name. Therefore every accepted measure-preserving read stays inside the finite recurrence certificate surface and cannot escape to the excluded host principles. □

Falsifiable prediction: If a recurrence-facing consumer preserves the $\mathsf{DynSystem}^{\uparrow}$ row but drops the $\mathsf{ProbSpace}^{\uparrow}$ normalization row, the $\mathsf{Measure}^{\uparrow}$ event row, or the finite return-window row, it is not reading a $\mathsf{PoincareRecurrence}^{\uparrow}$ packet.

Independence witness: dynsystem, ergodic, probspace, measure: no listed sibling is bijective to $\mathsf{PoincareRecurrence}^{\uparrow}$ because this packet jointly carries finite routed dynamics, invariant-support provenance, normalized probability, measurable source events, finite return windows, return-hit evidence, transport, replay, package sourcing, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{PoincareRecurrence}^{\uparrow}}$.

Constructive story: A $\mathsf{PoincareRecurrence}^{\uparrow}$ packet is generated from a finite DynSystem route, Ergodic invariant-support row, ProbSpace normalization row, Measure event row, source event, finite return-window ledger, return-hit witness, componentwise hsame transport, Cont replay, Pkg provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PoincareRecurrence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6269.1, Definition 6269.2, Definition 6269.3, and Definition 6269.4 seed the finite recurrence route through dynamics, invariant support, probability normalization, measurable events, return windows, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PoincareRecurrence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host recurrence theorems, selected recurrent points, completed invariant sigma-algebras, countable-choice return schedules, almost-everywhere predicates, ergodic decompositions, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dynamic routing, invariant-support stability, probability normalization, measurable-event exposure, return-window handoff, return-hit nonescape, transport, replay, provenance, and local naming for future BEDC.Derived.PoincareRecurrenceUp.

Chapter 6270

A Concrete Naming Certificate for $\mathsf{GronwallBellman}^{\uparrow}$

Following Bellman's 1943 integral inequality and the standard Bishop–Bridges constructive analysis route, the $\mathsf{GronwallBellman}^{\uparrow}$ packet records the finite integral-comparison surface used before Picard, stability, or completion consumers read a real bound. It sits after the displayed Gronwall comparison of chapter 2034, the finite integration surface of chapter 1526, the regular-product route of chapter 1702, the rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite partition, kernel-bound, majorant, product/exponential budget, error-budget, readback, seal, transport, replay, provenance, and local naming rows.

Definition 6270.1 (Gronwall–Bellman carrier)**.** *A* $\mathsf{GronwallBellman}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (I, K, M, P, E, R, Z, H, C, Q, N)$$

with the following visible rows:

1. *I is the interval-partition row, recording the displayed time interval, finite ordered partition cells, endpoint names, and the tag or subinterval reads used by the integral comparison.*
2. *K is the kernel-bound row, displaying the nonnegative integral kernel or coefficient bound on each inspected cell and its finite comparison to the partition data in I.*
3. *M is the majorant row, storing the displayed initial bound and inhomogeneous comparison envelope whose reads are bounded through the Bellman route.*
4. *P is the finite product/exponential budget row, obtained from the kernel-bound row by a finite multiplication or exponential enclosure through* $\mathsf{RegularCauchyMultiplication}^{\uparrow}$ *and the dyadic budget rows already exposed to the Real route.*
5. *E is the error-budget row, recording the local slack by which cellwise Riemann sums, product enclosures, and majorant reads are kept inside the displayed tolerance.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the partition endpoints, kernel bounds, majorant samples, product/exponential budget, and error-budget witnesses.*
7. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row for the final Bellman bound after the finite partition and readback rows have been replayed.*

8. *H records componentwise* hsame *transport for the interval partition, kernel bound, majorant, product/exponential budget, error budget, readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the finite route*

$$I, K, M, P, E, R \longmapsto Z \longmapsto \mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}.$$

10. *Q is the* Pkg *provenance over* $\mathsf{GronwallBellman}^{\uparrow}$, $\mathsf{GronwallInequality}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, RegularCauchyMultiplicati $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}$ *row.*

The carrier predicate compares accepted packets only through these rows and their displayed transports. It has no coordinate for a host integral operator, selected limit, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem, or Lean-verified theorem.*

Lean4 ✓ `BEDC.Derived.GronwallBellmanUp.TasteGate.GronwallBellmanTasteGate single carrier alignment`

Theorem 6270.2 (Gronwall–Bellman NameCert obligations)**.** *For every accepted* $\mathsf{GronwallBellman}^{\uparrow}$ *carrier $G = (I, K, M, P, E, R, Z, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}$ *surface consists exactly of carrier admission for the interval partition, kernel bound, majorant, finite product/exponential budget, error budget,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming rows. No obligation is discharged by a host integral, selected mesh limit, quotient stream equality, trichotomy, ambient* $\mathsf{Real}^{\uparrow}$ *completeness, or Lean verification.*

Proof. Project the accepted carrier of Definition 6270.1. The rows I, K, M, P, E, R, Z are the only analytic coordinates, and the structural rows H, C, Q, N only transport, replay, source, and name those coordinates.

The interval partition and kernel-bound rows determine the finite integral-comparison surface. The majorant row is read only over that same partition, while the product/exponential row records the finite enclosure produced from the kernel bounds.

The error-budget and regular-rational rows then bind the comparison to displayed observations. The real seal Z is exposed only after those finite rows have been replayed, so the local certificate cannot import a host integral, selected limit, quotient stream equality, trichotomy, ambient completeness theorem, or Lean verification. □

Theorem 6270.3 (Gronwall–Bellman finite partition handoff)**.** *Every accepted* $\mathsf{GronwallBellman}^{\uparrow}$ *carrier routes any integral-inequality read through the finite handoff*

$$\mathsf{RiemannIntegral}^{\uparrow} \longmapsto \mathsf{GronwallBellman}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The read opens the interval partition I, kernel-bound row K, majorant row M, finite product/exponential budget P, and error-budget row E before the regular-rational readback R or real seal Z is available.

Proof. Begin with an accepted carrier G. The only integration-facing source is the finite partition row I, and the only coefficient source is the kernel-bound row K on that partition.

Read the majorant and product/exponential rows next. The majorant row M supplies the displayed comparison envelope, while P supplies the finite multiplicative or exponential enclosure needed for Bellman's bound.

The error row E and readback row R keep the handoff finite. Since Z is sealed only after I, K, M, P, E, R have been replayed, every integral-inequality consumer receives the stated route and cannot bypass it through host integration, selected mesh limits, quotient equality, trichotomy, or ambient completeness. □

Theorem 6270.4 (Gronwall–Bellman Real seal nonescape)**.** *For every accepted* $\mathsf{GronwallBellman}^{\uparrow}$ *carrier* $G = (I, K, M, P, E, R, Z, H, C, Q, N)$*, the* $\mathsf{Real}^{\uparrow}$ *seal row* Z *exposes only the Bellman bound obtained from the finite partition, kernel-bound, majorant, product/exponential, error-budget, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows. It exports no host integral value, selected limit, quotient stream equality, trichotomy branch, ambient* $\mathsf{Real}^{\uparrow}$ *completeness result, or Lean-checked bridge.*

Proof. Use Definition 6270.2 to identify the full local obligation surface. The seal row is listed there only as the terminal read of the finite carrier rows.

Use Definition 6270.3 to locate the route into Z. The handoff opens I, K, M, P, E, R before the real seal is available, so the seal cannot be an independent host integral or selected limit.

The structural rows H, C, Q, N preserve this same route under transport, replay, provenance, and local naming. Therefore a consumer of Z receives only the finite Bellman-bound packet and no quotient stream equality, trichotomy branch, ambient completeness theorem, or Lean-checked bridge. □

Falsifiable prediction: Every accepted $\mathsf{GronwallBellman}^{\uparrow}$ consumer that reads a real Bellman bound without the displayed interval partition, kernel-bound row, majorant row, finite product/exponential budget, error-budget row, RegSeqRat readback, transport, replay, provenance, and local NameCert rows refutes the packet surface.

Independence witness: gronwall-inequality, riemannintegral, regular-cauchy-multiplication, regseqrat, real: none is bijective to $\mathsf{GronwallBellman}^{\uparrow}$ because this packet binds finite partition data, kernel bounds, majorant reads, product/exponential budgets, error budgets, regular-rational readback, Real seal, transport, replay, provenance, and local naming into one integral-inequality certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}$**.**

Constructive story: A $\mathsf{GronwallBellman}^{\uparrow}$ packet is generated from one interval-partition row, one kernel-bound row, one majorant row, one finite product/exponential budget row, one error-budget row, one RegSeqRat readback row, one RealUp seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6270.1, Definition 6270.2, Definition 6270.3, and Definition 6270.4 seed the finite Bellman integral-inequality packet over partition, kernel-bound, majorant, product/exponential, error-budget, regular-rational, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GronwallBellman}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host integrals, selected limits, quotient stream equality, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite partition handoff, kernel-bound stability, product/exponential budget accounting, error-budget accounting, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.GronwallBellmanUp.

Chapter 6271

A Concrete Naming Certificate for $\mathsf{RealUniformCauchyProduct}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy products, the $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ packet records the finite route by which two uniformly Cauchy real-window families expose a product family with the same uniform-tail discipline. It sits beside the shared-threshold criterion of chapter 435, the product-convergence route of chapter 3608, the regular product criterion of chapter 3672, and the endpoint-seal discipline of chapter 463. The chapter names only the finite product-stability certificate; it does not assert quotient sequence equality, selected product limits, countable choice, host function equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6271.1 (Real uniform Cauchy product carrier)**.** *A* $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (F, G, M, W_F, W_G, D, U, K, R, E, H, C, Q, N)$$

with the following displayed rows:

1. *F and G are the two real-window family rows whose values are read only through finite* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *endpoint displays.*
2. *M is the shared uniform threshold ledger assigning each product tolerance to one family-wide tail bound for both inputs.*
3. *W_F and W_G are the synchronized source-window rows exposed at the product request and the threshold selected by M.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *product-budget row, recording input tolerances, visible endpoint bounds, cross-error terms, and the output tolerance for the product window.*
5. *U is the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row certifying that the source comparisons beyond M are uniform in the displayed index.*
6. *K is the* $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$ *row that turns the synchronized source windows and dyadic product budget into a regular product read.*
7. *R is the* $\mathsf{CauchyProductConvergence}^{\uparrow}$ *route row for the resulting finite product-tail calculation.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the uniform tail, product budget, regular criterion, and product-convergence route have been displayed.*

9. *H records componentwise* hsame *transport for the source families, threshold ledger, synchronized windows, dyadic budget, uniform-tail row, product criterion, convergence route, seal row, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from a product-stability request through $F, G, M, W_F, W_G, D, U, K, R, E, H, Q, N$*
11. *Q is the* Pkg *provenance over* $\mathsf{RealUniformCauchyProduct}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, RegularCauchyProductCriter CauchyProductConvergence$^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealUniformCauchyProduct}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows, their transports, replay routes, provenance, and local naming row.

Lean4 ✓ `BEDC.Derived.RealUniformCauchyProductUp.RealUniformCauchyProductTasteGate single carrier alignment`

Theorem 6271.2 (Real uniform Cauchy product NameCert obligations)**.** *For every accepted* $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ *carrier $P = (F, G, M, W_F, W_G, D, U, K, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealUniformCauchyProduct}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the two source families, shared threshold ledger, synchronized source windows, dyadic product budget, uniform-tail row, product criterion, product-convergence route, Real seal, transport, replay, provenance, and local naming;*
2. *uniform-threshold stability under componentwise* hsame *transport of F, G, M, W_F, W_G;*
3. *dyadic product-budget exactness through D, with all endpoint bounds and cross-error rows displayed before K is read;*
4. *uniform-tail compatibility between M, the synchronized source windows, and the* $\mathsf{UniformCauchyCriterion}^{\uparrow}$ *row U;*
5. *product-criterion routing through K and product-convergence routing through R before the terminal* $\mathsf{Real}^{\uparrow}$ *seal E is exposed;*
6. *displayed* Cont *replay and* Pkg *provenance for the whole route;*
7. *refusal of any consumer that omits a source family, shared threshold, synchronized window, dyadic product budget, uniform-tail row, product criterion, convergence route, Real seal, transport row, replay row, provenance row, or local naming row.*

Lean4 ✓ `BEDC.Derived.RealUniformCauchyProductUp.RealUniformCauchyProductCarrier namecert obligations`

Proof. Project the carrier of Definition 6271.1. The projection exposes the two source families, shared threshold ledger, synchronized windows, dyadic product budget, uniform-tail row, product criterion, product-convergence route, Real seal, and support rows.

The uniform part is bound by M, W_F, W_G, U. A consumer can read the product family only after the same threshold has supplied both source comparisons, so the route is not pointwise-only.

The product part is bound by D, K, R. The dyadic row records the finite product budget, the regular criterion consumes that budget, and the convergence row records the product-tail calculation before any seal is available.

Finally E, H, C, Q, N seal, transport, replay, package, and name the same finite route. Since the carrier has no quotient-sequence, selected-limit, host-function-equality, choice, or ambient-completeness coordinate, the obligation list is exhaustive. □

Theorem 6271.3 (Real uniform Cauchy product ten-field package)**.** *The* $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ *certificate instantiates the ten-field roadmap template as follows: traditional target, uniform Cauchy stability of products of real-window families; BEDC source, the displayed* BHist *route through* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{UniformCauchyCriterion}^{\uparrow}$, $\mathsf{RegularCauchyProductCriterion}^{\uparrow}$, $\mathsf{CauchyProductConverger}$ *and* $\mathsf{Real}^{\uparrow}$*; carrier,* P*; classifier, componentwise comparison of the displayed rows; stability,* hsame *transport through* H*; ledger, the rows* M, W_F, W_G, D, U, K, R*; core theorem list, Definition 6271.2 and Definition 6271.4; exactness, exhaustion of the listed rows; standard bridge, none; status, paper seed certificate for* $\mathsf{NameCert}_{\mathsf{RealUniformCauchyProduct}^{\uparrow}}$.

Proof. Start with the carrier P of Definition 6271.1. Its source rows, threshold ledger, synchronized windows, dyadic budget, uniform-tail row, product criterion, convergence route, seal row, and support rows provide the listed BEDC source and carrier fields.

Apply Definition 6271.2. The theorem supplies the classifier, stability, ledger, routing, replay, provenance, and nonescape obligations, while the absence of a bridge row keeps the standard-bridge field empty.

Thus the ten fields form one citable seed certificate rather than a stronger analytic claim about completed real functions or selected product limits. □

Theorem 6271.4 (Real uniform Cauchy product seal factorization)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ *carrier factors through the finite route*

$$F, G \longmapsto M, W_F, W_G \longmapsto D, U, K, R \longmapsto E, H, C, Q, N.$$

Thus a downstream Real or uniform-limit consumer first reads the two source families and the shared uniform threshold, then the synchronized windows, product budget, uniform-tail row, product criterion, and product-convergence route, and only then the terminal seal and local support rows. The route exports no quotient sequence equality, selected product limit, host function equality, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with an accepted packet $P = (F, G, M, W_F, W_G, D, U, K, R, E, H, C, Q, N)$. The only source-family coordinates are F and G, and the only shared threshold coordinate is M.

The synchronized windows W_F, W_G and the dyadic row D expose the finite data used by the product request. The uniform-tail row U and regular product criterion K then certify that these data are adequate for a uniform Cauchy product read.

The product-convergence row R precedes the Real seal E. The rows H, C, Q, N only transport, replay, package, and name the same route, so no second endpoint route can bypass the displayed product-tail calculation. □

Falsifiable prediction: Every accepted $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ consumer that reaches a terminal $\mathsf{Real}^{\uparrow}$ seal while omitting a source family, shared threshold ledger, synchronized source window, DyadicRatCore product-budget row, UniformCauchyCriterion row, RegularCauchyProductCriterion row, CauchyProductConvergence route, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{RealUniformCauchyProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{RealUniformCauchyProduct}^{\uparrow}$ packet is generated from two real-window source families, a shared uniform threshold ledger, synchronized StreamName windows, a DyadicRatCore product budget, a UniformCauchyCriterion row, a RegularCauchyProductCriterion row, a CauchyProductConvergence route, a RealUp seal, componentwise transport, displayed continuation replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealUniformCauchyProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6271.1, Definition 6271.2, Definition 6271.3, and Definition 6271.4 seed the finite uniform product-stability route from source families and shared thresholds to product budget, convergence routing, and Real seal rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{RealUniformCauchyProduct}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected product limits, host function equality, countable choice, trichotomy, ambient Real$^{\uparrow}$ completeness, Lean verification, or a public standard bridge.

Upgrade path: Obligation closure requires checked carrier admission, uniform-threshold stability, synchronized-window admission, dyadic product-budget exactness, UniformCauchyCriterion compatibility, RegularCauchyProductCriterion routing, CauchyProductConvergence routing, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealUniformCauchyProductUp.

Chapter 6272

A Concrete Naming Certificate for $\mathsf{PeanoKernel}^{\uparrow}$

Following Peano and classical numerical analysis, the $\mathsf{PeanoKernel}^{\uparrow}$ packet isolates the finite-window data by which a quadrature-error functional is read without importing a host limiting theorem. It sits after the finite mesh carrier of chapter 3496, the tagged Riemann-sum window surface of chapter 4115, the Darboux comparison row of chapter 1880, the regular-rational readback of chapter 301, the stream-window naming route of chapter 126, the dyadic tolerance ledger of chapter 291, and the terminal Real seal of chapter 38. The packet names only a displayed finite interval source, a mesh, a finite annihilation ledger for test polynomials, a kernel row, a sum consumer, regular readback, stream windows, dyadic tolerance, transport, provenance, and local naming; it does not claim host quadrature convergence, selected mesh limits, quotient equality of functions, measure completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6272.1 (Peano-kernel carrier)**.** *A* $\mathsf{PeanoKernel}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, M, A, Q, S, R, W, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Real}^{\uparrow}$ *interval-source row naming the two endpoint seals, the oriented finite window, and the real-valued functional being inspected.*
2. *M is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *row, either a finite dyadic mesh or a finite uniform mesh, with displayed mesh points and adjacent-cell incidence.*
3. *A is the test-polynomial annihilation ledger. It records the finite degree bound, the listed polynomial tests, and the zero-error verdict for those tests.*
4. *Q is the kernel row. It stores the signed finite-window weights or cellwise kernel values used by the quadrature-error read after I, M, A have been displayed.*
5. *S is the Riemann/Darboux sum consumer row, read through* $\mathsf{RiemannSumGauge}^{\uparrow}$ *or* $\mathsf{DarbouxSum}^{\uparrow}$ *data attached to the same mesh.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row exposing the displayed finite error value or bound as a regular rational name.*
7. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row recording the finite read windows through which consumers may request the error value.*

8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for the requested precision, comparison slack, and cellwise error budget.*
9. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *error seal reached only after M, Q, S, R, W, D are public.*
10. *H records componentwise* hsame *transport for interval source, mesh, annihilation ledger, kernel row, sum consumer, readback, stream windows, dyadic tolerance, Real seal, replay, provenance, and local naming.*
11. *C records displayed* Cont *replay from every quadrature-error request through $I, M, A, Q, S, R, W, D, E, H, P, N$.*
12. *P is the* Pkg *provenance over* $\mathsf{PeanoKernel}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
13. *N is the local* $\mathsf{NameCert}_{\mathsf{PeanoKernel}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $I, M, A, Q, S, R, W, D, E, H, C, P, N$. It has no coordinate for an infinite refinement process, selected mesh limit, quotient equality of functions, measure completion, host convergence theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.PeanoKernelUp.PeanoKernelTasteGate single carrier alignment`

Theorem 6272.2 (Peano-kernel NameCert obligations)**.** *For every accepted* $\mathsf{PeanoKernel}^{\uparrow}$ *carrier*

$$K = (I, M, A, Q, S, R, W, D, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{PeanoKernel}^{\uparrow}}$ *surface consists exactly of carrier admission for K; Real interval-source exposure through I; finite dyadic or uniform mesh exposure through M; test-polynomial annihilation accountability through A; kernel-row compatibility through Q; Riemann/Darboux sum-consumer compatibility through S; regular-rational readback through R; stream-window read discipline through W; dyadic tolerance exactness through D; terminal Real error sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; local naming through N; and nonescape excluding host quadrature convergence, selected mesh limits, quotient function equality, measure completion, detached error representatives, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Carrier admission is projection from Definition 6272.1. The interval, mesh, annihilation ledger, kernel row, and sum consumer are the analytic-facing coordinates I, M, A, Q, S.

The readback coordinates are ordered. A quadrature-error request first fixes the displayed mesh and polynomial-test ledger, then reads the kernel and sum rows, then exposes the regular rational row R through the stream window row W under the dyadic tolerance ledger D.

The terminal seal is E. It is downstream of the finite mesh, kernel, sum, readback, stream-window, and tolerance rows, so it exports a Real error name rather than a host convergence theorem.

Finally H, C, P, N transport, replay, source, and name exactly the same route. Since no carrier coordinate stores an infinite refinement process, selected mesh limit, quotient equality of functions, measure completion, detached error representative, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6272.3 (Peano-kernel error-functional route)**.** *Every quadrature-error read of an accepted* $\mathsf{PeanoKernel}^{\uparrow}$ *carrier factors through the displayed finite route*

$$\mathsf{Real}^{\uparrow} \longmapsto \mathsf{FinitePartitionMesh}^{\uparrow} \longmapsto \mathsf{PeanoKernel}^{\uparrow} \longmapsto (\mathsf{RiemannSumGauge}^{\uparrow} \textit{ or } \mathsf{DarbouxSum}^{\uparrow}) \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto$$

Equivalently, the read can inspect only the finite mesh row M, the kernel row Q, the sum row S, the regular readback row R, the stream-window row W, the dyadic tolerance ledger D, the Real error seal E, and the structural rows H, C, P, N after the interval source and annihilation ledger have been displayed.

Proof. Start with an accepted carrier $K = (I, M, A, Q, S, R, W, D, E, H, C, P, N)$ and a quadrature-error read. The only interval source is I, and the only mesh source is the displayed finite row M.

The annihilation ledger A binds the polynomial tests before the kernel is read. The row Q then provides the cellwise kernel or weight data, and S consumes it through the displayed Riemann or Darboux sum surface attached to the same mesh.

The read cannot leave the finite route after S. It reaches the regular-rational row R, the stream-window row W, and the dyadic tolerance ledger D before the terminal Real error seal E is exposed.

The rows H, C, P, N preserve that order under transport, replay, provenance, and local naming. Because the carrier contains no infinite refinement process, selected mesh limit, quotient function equality, measure completion, detached error representative, or ambient completeness coordinate, every accepted quadrature-error read factors through the stated finite route. □

Closure status for $\mathsf{NameCert}_{\mathsf{PeanoKernel}^{\uparrow}}$.

Constructive story: A $\mathsf{PeanoKernel}^{\uparrow}$ packet is generated from a RealUp interval source, a finite dyadic or uniform partition mesh, a test-polynomial annihilation ledger, a kernel row, a Riemann/Darboux sum consumer, RegSeqRatUp readback, StreamNameUp windows, a DyadicRatCoreUp tolerance ledger, a RealUp error seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{PeanoKernel}^{\uparrow}}$, seed)

Scope closed: Definition 6272.1, Definition 6272.2, and Definition 6272.3 seed the finite quadrature-error route from interval source and finite mesh to kernel, sum, regular readback, stream windows, dyadic tolerance, and the Real error seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{PeanoKernel}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host quadrature convergence, selected mesh limits, quotient function equality, measure completion, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, interval-source exposure, finite mesh exposure, annihilation-ledger accountability, kernel-row compatibility, sum-consumer compatibility, regular readback, stream-window discipline, dyadic tolerance exactness, Real error sealing, transport, replay, provenance, and local naming for BEDC.Derived.PeanoKernelUp.

Chapter 6273

A Concrete Naming Certificate for BanachFixedPointIterationBudget$^{\uparrow}$

Following Banach and Bishop–Bridges constructive analysis, the BanachFixedPointIterationBudget$^{\uparrow}$ packet records the finite budget surface that sits between displayed contraction iteration and the fixed-point endpoint seal. It consumes the finite orbit ledger of chapter 2383, the contraction-map surface of chapter 1418, the complete-metric boundary of chapter 128, the regular Cauchy handoff of chapter 301, and the endpoint-facing Banach certificate of chapter 396. The packet names the tail budget a consumer must inspect before BanachFixedPoint$^{\uparrow}$ may read an endpoint.

Definition 6273.1 (Banach fixed-point iteration budget carrier)**.** *A* BanachFixedPointIterationBudget$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, T, x_0, I, Q, R, K, H, C, P, N)$$

with the following displayed rows:

1. *X is the metric source row, read through the carried metric and complete-metric surfaces before any orbit estimate is consumed.*
2. *T is the* ContractionMapping$^{\uparrow}$ *self-map row, including the graph-read and strict contraction-ratio ledger.*
3. *x_0 is the displayed initial-point row for the Picard orbit.*
4. *I is the finite Picard ledger supplied by* FixedPointIteration$^{\uparrow}$*, listing only the displayed iterates and adjacent-step reads.*
5. *Q is the contraction-ratio and tail-budget row, recording the finite residual threshold attached to the displayed Picard ledger.*
6. *R is the* RegSeqRat$^{\uparrow}$ *handoff row that receives the tail budget as regular Cauchy data.*
7. *K is the* CompleteMetric$^{\uparrow}$ *boundary row; it records the separate completion service that must be supplied before an endpoint-facing read.*
8. *H records componentwise* hsame *transport for the metric source, contraction map, initial point, Picard ledger, tail budget, RegSeqRat handoff, complete-metric boundary, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from an endpoint request through X, T, x_0, I, Q, R, K, H.*

10. *P is the* Pkg *provenance over* BanachFixedPointIterationBudget$^{\uparrow}$, FixedPointIteration$^{\uparrow}$, ContractionMapping$^{\uparrow}$, CompleteMetric$^{\uparrow}$, RegSeqRat$^{\uparrow}$, BanachFixedPoint$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* NameCert$_{\text{BanachFixedPointIterationBudget}^{\uparrow}}$ *row.*

The classifier compares packets only through $X, T, x_0, I, Q, R, K, H, C, P, N$. It has no coordinate for a selected fixed point, completed Picard endpoint, host fixed-point equality, quotient map equality, countable choice, or ambient completeness outside the displayed CompleteMetric$^{\uparrow}$ *boundary.*

Lean4 (definition) `BEDC.Derived.BanachFixedPointIterationBudgetUp.BanachFixedPointIterationBudgetUp`

Theorem 6273.2 (Banach fixed-point contracting-step budget)**.** *For every accepted* BanachFixedPointIterationBudg *carrier $B = (X, T, x_0, I, Q, R, K, H, C, P, N)$, each displayed Picard step in I shrinks the residual budget through the contraction row T and the ratio-tail ledger Q. A* BanachFixedPoint$^{\uparrow}$ *endpoint consumer may inspect only the finite route*

$$\text{ContractionMapping}^{\uparrow}, \quad \text{FixedPointIteration}^{\uparrow}, \quad \text{RegSeqRat}^{\uparrow}, \quad \text{CompleteMetric}^{\uparrow}$$

before reading the endpoint-facing seal. Thus the packet exposes the finite tail-modulus route and no selected endpoint, host fixed-point equality, quotient-map equality, countable-choice selector, or ambient completeness theorem.

Proof. Project the carrier rows. The Picard ledger I is finite, and every adjacent displayed step is read through the same metric source X, contraction graph T, and initial-point row x_0.

The contraction budget is the row Q. It stores the strict ratio and the residual threshold attached to the displayed ledger, so each later Picard read can only decrease the residual request along the finite ledger already present in I.

The completion-facing handoff is separate. The row R repacks the tail budget as a RegSeqRat$^{\uparrow}$ route, and the row K records the CompleteMetric$^{\uparrow}$ boundary that must be supplied before BanachFixedPoint$^{\uparrow}$ reads an endpoint.

Finally, H, C, P, N transport, replay, source, and name this same route. Since no carrier coordinate contains a selected endpoint, host fixed-point equality, quotient-map equality, choice selector, or ambient completeness theorem, none can be exported by the contracting-step budget packet. □

Closure status for NameCert$_{\text{BanachFixedPointIterationBudget}^{\uparrow}}$.

Constructive story: A BanachFixedPointIterationBudget$^{\uparrow}$ packet is generated from a metric source row, a ContractionMapping self-map row, an initial point, a finite Picard ledger, a contraction-ratio and tail-budget row, a RegSeqRat handoff, a CompleteMetric boundary, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{BanachFixedPointIterationBudget}^{\uparrow}}$, seed)

Scope closed: Definition 6273.1 and Definition 6273.2 seed the finite budget surface between Picard iteration and Banach endpoint sealing.

Formal verification: FormalStatus(NameCert$_{\text{BanachFixedPointIterationBudget}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.BanachFixedPointIterationBudgetUp.BanachFixedPointIterationBudgetUp`

Bridge status: none

Not claimed: This chapter does not close selected fixed points, completed Picard endpoints, host fixed-point equality, quotient-map equality, countable choice, ambient completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Picard-step budget shrinkage, RegSeqRat tail handoff, complete-metric boundary discipline, transport, replay, provenance, and local naming for BEDC.Derived.BanachFixedPointIterationBu

Chapter 6274

A Concrete Naming Certificate for $\mathsf{CompactCauchyEquicontinuity}^{\uparrow}$

Following Heine–Cantor and Arzela–Ascoli constructive analysis, the $\mathsf{CompactCauchyEquicontinuity}^{\uparrow}$ packet records the finite Cauchy/equicontinuity route behind chapter 9237. It reads compact finite-net data from chapter 129, displayed map rows from chapter 130, regular tolerance readback from chapter 301, dyadic tolerance cells from chapter 291, and the terminal seal from chapter 38. The packet names the finite route used when a compact family has Cauchy-equicontinuity data before any $\mathsf{Real}^{\uparrow}$ consumer reads a completed value.

Definition 6274.1 (Compact Cauchy equicontinuity carrier)**.** *A* $\mathsf{CompactCauchyEquicontinuity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (K, F, M, W, D, R, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, including the finite-net centers, cover radii, and source metric observations used by the packet.*
2. *F is the finite* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, with each inspected member routed through a displayed graph and pointwise-modulus packet.*
3. *M is the pointwise family-modulus ledger, assigning only displayed center/tolerance requests to local source-radius rows.*
4. *W is the finite window row for the compact-net centers and inspected family members.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, carrying positive dyadic requests, slack comparisons, and radius refinements.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that turns the displayed Cauchy/equicontinuity window into regular rational tolerance data.*
7. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after K, F, M, W, D, R have been displayed.*
8. *H records componentwise* hsame *transport for compact source, family, pointwise modulus, windows, dyadic tolerance, regular readback, real seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay for every compact-uniform-continuity consumer route through K, F, M, W, D, R, S, H.*

10. *P is the* Pkg *provenance over* $\mathsf{CompactCauchyEquicontinuity}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactCauchyEquicontinuity}^{\uparrow}}$ *row.*

The classifier compares packets only through $K, F, M, W, D, R, S, H, C, P, N$. It has no coordinate for open-cover compactness, selected subcovers, selected limiting functions, quotient stream equality, countable choice, host equicontinuity, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.CompactCauchyEquicontinuityUp.CompactCauchyEquicontinuityUp`

Theorem 6274.2 (Compact Cauchy equicontinuity finite route)**.** *For every accepted* CompactCauchyEquicontinuity *carrier $E = (K, F, M, W, D, R, S, H, C, P, N)$, any compact-uniform-continuity consumer that requests a later Cauchy tolerance must first read the finite route*

$$\mathsf{CompactMetric}^{\uparrow}, \quad \mathsf{ContinuousMap}^{\uparrow}, \quad \mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}$$

through the displayed rows K, F, M, W, D, R, S. The consumer receives a finite-net, pointwise-modulus, regular-tolerance, and real-seal handoff, and it receives no open-cover compactness, selected family limit, quotient stream equality, countable choice, host equicontinuity theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The compact source row K supplies only finite-net centers, cover radii, and source metric observations, while the family row F supplies only the displayed $\mathsf{ContinuousMap}^{\uparrow}$ members available to the packet.

The equicontinuity request is then forced through M, W, D. The row M records the local pointwise-radius ledger, W restricts the request to displayed finite windows, and D carries the dyadic tolerance and slack comparisons that refine the requested budget.

The Cauchy and Real-facing handoff is downstream. The row R repacks the displayed window and dyadic tolerance as a $\mathsf{RegSeqRat}^{\uparrow}$ readback, and S may be consumed only after that readback is present.

Finally, H, C, P, N transport, replay, source, and name the same route. Since no row stores open-cover compactness, a selected family limit, quotient stream equality, countable choice, host equicontinuity, or ambient real completeness, the finite route cannot export any of those objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactCauchyEquicontinuity}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactCauchyEquicontinuity}^{\uparrow}$ packet is generated from a CompactMetricUp source row, finite ContinuousMapUp family rows, pointwise family-modulus rows, finite windows, DyadicRatCoreUp tolerance rows, RegSeqRatUp readback, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactCauchyEquicontinuity}^{\uparrow}}$, seed)

Scope closed: Definition 6274.1 and Definition 6274.2 seed the compact Cauchy/equicontinuity route for compact-uniform-continuity consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactCauchyEquicontinuity}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CompactCauchyEquicontinuityUp.CompactCauchyEquicontinuityUp`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, selected limiting functions, quotient stream equality, countable choice, host equicontinuity, ambient RealUp completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net route discipline, pointwise family-modulus routing, dyadic tolerance refinement, RegSeqRat readback, RealUp seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.CompactCauchyEquicontinuityUp.

Chapter 6275

A Concrete Naming Certificate for $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ packet records the finite compact-indexed modulus extraction by which a uniformly Cauchy family over a compact metric source supplies one shared tail schedule for Real and Completion consumers. It sits over the compact metric surface of chapter 129, the compact uniform Cauchy route of chapter 6264, the Cauchy-modulus schedule of chapter 292, the finite stream-window surface of chapter 126, the regular rational readback surface of chapter 301, and the terminal Real seal of chapter 38. The packet is a NameCert carrier for a compact-indexed tail schedule; it is not an open-cover compactness theorem, not a selected limit construction, and not a host real equality principle.

Definition 6275.1 (Compact uniform Cauchy modulus carrier)**.** *A* $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (K, F, M, S, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact metric index space whose finite cover data may be inspected.*
2. *F is the* $\mathsf{UniformCauchy}^{\uparrow}$ *family row naming the uniformly Cauchy rational stream family over K.*
3. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *shared tail schedule chosen uniformly for the displayed compact source and family.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the stream positions read after the shared schedule in M.*
5. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row for the scheduled finite windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row available only after K, F, M, S, Q have been displayed.*
7. *H records componentwise* hsame *transport for compact source, family, modulus schedule, finite window, regular readback, Real seal, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-indexed precision request through K, F, M, S, Q, E, H.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{UniformCauchy}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, M, S, Q, E, H, C, P, N$. It has no coordinate for a chosen compact cover, an extracted subnet, a selected limit, quotient stream equality, countable choice, host real equality, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CompactUniformCauchyModulusUp.CompactUniformCauchyModulusTasteGate single carrier alignment`

Theorem 6275.2 (Compact uniform Cauchy modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ *carrier* $U = (K, F, M, S, Q, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyM}}$ *surface consists exactly of carrier admission, compact metric source exposure, uniformly Cauchy family exposure, shared Cauchy-modulus extraction, finite stream-window exposure, regular rational readback, Real seal handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$K, F \longmapsto M \longmapsto S, Q \longmapsto E \longmapsto H, C, P, N$$

before using a compact-indexed uniform tail schedule.

Proof. Project the carrier of Definition 6275.1. The compact source row K and the uniformly Cauchy family row F are the only analytic inputs to the shared schedule.

Read M next. The Cauchy-modulus row records one displayed tail schedule for the whole compact-indexed family, so a consumer cannot replace it by pointwise schedules detached from K and F.

The finite stream-window row S and the regular rational readback row Q then expose the scheduled finite observations. The Real seal E and structural rows H, C, P, N transport, replay, source, and name the same route. Since selected limits, quotient stream equality, countable choice, host real equality, chosen compact covers, and ambient Real completeness are absent from the carrier, none of those data can occur in the local NameCert surface. □

Theorem 6275.3 (Compact finite-net tail extraction)**.** *Let* $U = (K, F, M, S, Q, E, H, C, P, N)$ *be an accepted* $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ *carrier. Any finite-net or Real-and-Completion consumer that asks for a compact-indexed uniform tail must read $K, F, M, S, Q, E, H, C, P, N$ in the displayed order. The exported tail is therefore a finite* $\mathsf{StreamName}^{\uparrow}$ *window and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback governed by one shared* $\mathsf{CauchyModulus}^{\uparrow}$ *row over the* $\mathsf{CompactMetric}^{\uparrow}$ *source, not a pointwise family of radii, selected subnet, host limit, quotient stream, countable-choice schedule, or ambient compactness theorem.*

Proof. Begin with the NameCert obligation theorem Definition 6275.2. It fixes the compact source, uniformly Cauchy family, shared schedule, finite window, rational readback, and Real seal as the whole public route.

A finite-net-facing consumer can inspect the compact source only through K and the family only through F. The shared schedule M must be read before any finite window S or readback row Q is available, so the tail cannot be assembled from detached pointwise radius schedules.

For a Real-and-Completion consumer, the seal row E is reached only after the same finite route has been displayed. The replay and provenance rows preserve that route and add no selected subnet, host limit, quotient stream, countable-choice schedule, or ambient compactness theorem. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ Real or Completion consumer factors through one $\mathsf{CompactMetric}^{\uparrow}$ source row, one $\mathsf{UniformCauchy}^{\uparrow}$ family row, one shared $\mathsf{CauchyModulus}^{\uparrow}$ schedule, one $\mathsf{StreamName}^{\uparrow}$ finite-window row, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{Real}^{\uparrow}$ seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it may use a compact-indexed uniform tail.

Independence witness: compactmetric, uniformcauchy, cauchymodulus, streamname, regseqrat, real: $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ is not bijective to any listed sibling because the siblings separately supply compact sources, uniformly Cauchy families, modulus schedules, finite stream windows, rational readback, and terminal Real sealing, while this packet binds them to one compact-indexed uniform tail schedule.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformCauchyModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformCauchyModulus}^{\uparrow}$ packet is generated from a CompactMetric source row, a UniformCauchy family row, a shared CauchyModulus schedule, a StreamName finite-window row, a RegSeqRat readback row, a Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6275.1, Definition 6275.2, and Definition 6275.3 seed the compact-indexed uniform Cauchy modulus interface for the L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformCauchyModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected limits, quotient stream equality, countable choice, host real equality, Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact-source exposure, uniformly Cauchy family exposure, shared schedule exactness, finite-window readback, Real seal handoff, structural transport, provenance non-escape, and local NameCert exactness for BEDC.Derived.CompactUniformCauchyModulusUp.

Chapter 6276

A Concrete Naming Certificate for MengerCoverCompactness$^{\uparrow}$

Following Menger and Bishop–Bridges constructive compactness, the MengerCoverCompactness$^{\uparrow}$ packet records finite cover-selection compactness as displayed NameCert data. It sits after the compact metric source of chapter 129, the finite-subcover criterion of FiniteSubcoverCriterion$^{\uparrow}$, the effective cover compactness surface of EffectiveCoverCompactness$^{\uparrow}$, the finite stream-window route of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names finite cover requests and finite subcover witnesses; it does not choose an open-cover subfamily from a host theorem.

Definition 6276.1 (Menger cover compactness carrier)**.** *A* MengerCoverCompactness$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, R, B, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact metric source row read through* CompactMetric$^{\uparrow}$*.*
2. *R is a finite sequence of cover requests, each naming one inspected finite open-cover window and its tolerance budget.*
3. *B is the* ProbeBundle *finite-subcover ledger, recording for every request in R the selected finite probe rows, membership witnesses, and cover verification.*
4. *W is the* StreamName$^{\uparrow}$ *finite-window readback row used to inspect the cover requests and probe names.*
5. *Q is the* RegSeqRat$^{\uparrow}$ *readback row for the rational radii and comparison margins attached to the selected probes.*
6. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after K, R, B, W, Q have been displayed.*
7. *H records componentwise* hsame *transport for the compact metric source, cover requests, finite-subcover ledger, stream windows, regular-rational readback, Real seal, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from each cover request through $K, R, B, W, Q, E, H, P, N$.*
9. *P is the* Pkg *provenance over* MengerCoverCompactness$^{\uparrow}$, CompactMetric$^{\uparrow}$, FiniteSubcoverCriterion$^{\uparrow}$, EffectiveCoverCompactness$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert*.*

10. N is the local $\mathsf{NameCert}_{\mathsf{MengerCoverCompactness}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $K, R, B, W, Q, E, H, C, P, N$. It contains no coordinate for countable choice, an arbitrary open-cover selector, quotient real equality, a host compactness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6276.2 (Menger cover compactness NameCert obligations)**.** *For every accepted* MengerCoverCompactnes *carrier $M = (K, R, B, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MengerCoverCompactness}^{\uparrow}}$ *surface consists exactly of compact metric source exposure through K; finite cover-request exposure through R;* ProbeBundle *finite-subcover verification through B; stream-window readback through W; regular-rational readback through Q; terminal Real sealing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; and local naming through N.*

Lean4 ✓ `BEDC.Derived.MengerCoverCompactnessUp.MengerCoverCompactnessCarrier namecert obligations`

Proof. Project the carrier of Definition 6276.1. The compact metric source, request list, and finite-subcover ledger are exactly K, R, B.

The readback rows W, Q, E are downstream of those finite cover rows. They expose the finite stream windows, rational radius data, and terminal real-name seal only after each selected subcover witness has been displayed.

Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route. Since the carrier has no row for countable choice, host open-cover selection, quotient real equality, or ambient completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6276.3 (Menger cover compactness finite cover selection)**.** *Every finite cover-selection consumer of an accepted* $\mathsf{MengerCoverCompactness}^{\uparrow}$ *carrier must read the compact metric source K, then the finite request sequence R, then the* ProbeBundle *finite-subcover row B, before it can inspect the* $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *readback rows. The route factors through* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$ *and* $\mathsf{EffectiveCoverCompactness}^{\uparrow}$ *as finite certificate surfaces, not as a choice-selected open-cover theorem.*

Proof. Begin with an accepted carrier $M = (K, R, B, W, Q, E, H, C, P, N)$. The row K supplies the compact metric source, while R lists the only cover requests visible to the consumer.

For each request, B stores the finite probe rows and verification that they cover the inspected window. Thus the consumer receives finite subcover evidence before any stream, rational, or real readback row is exposed.

Finally W, Q, E, H, C, P, N read, transport, replay, source, and name that finite evidence. The route cannot introduce an arbitrary open-cover selector or countable-choice schedule, because no such coordinate occurs in the carrier. □

Theorem 6276.4 (Menger cover compactness Real-seal nonescape)**.** *No* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{MengerCoverCompactness}^{\uparrow}$ *packet can bypass the displayed route*

$$\mathsf{CompactMetric}^{\uparrow},\ R,\ \mathsf{ProbeBundle},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

The terminal Real seal E is public only after the compact metric source, finite cover requests, finite-subcover ledger, stream windows, and rational radius readback have all been displayed.

Proof. Read the finite cover-selection theorem Definition 6276.3. It places K, R, B before the stream, rational, and Real rows.

The terminal seal E is not an independent Real object. It is reached only through W and Q, which are themselves attached to the finite request and finite-subcover rows.

Consequently every accepted Real-facing read must pass through the displayed compactness packet and its structural rows H, C, P, N. The excluded host compactness, selected subcover, countable-choice, quotient-equality, and ambient-completeness claims are not exported. □

Falsifiable prediction: Every accepted $\mathsf{MengerCoverCompactness}^{\uparrow}$ Real or Completion consumer exposes the compact metric source, finite cover-request sequence, ProbeBundle finite-subcover row, stream windows, regular-rational readback, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before using finite cover-selection compactness.

Independence witness: compactmetric, finite-subcover-criterion, effective-cover-compactness, streamname, regseqrat, real: $\mathsf{MengerCoverCompactness}^{\uparrow}$ is not bijective to any listed sibling because this packet binds compact metric source rows, finite cover requests, finite ProbeBundle subcover witnesses, stream windows, rational readback, Real sealing, transport, replay, provenance, and local naming into one cover-selection certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{MengerCoverCompactness}^{\uparrow}}$**.**

Constructive story: A $\mathsf{MengerCoverCompactness}^{\uparrow}$ packet is generated from a CompactMetricUp source row, finite cover requests, ProbeBundle finite-subcover rows, StreamNameUp windows, RegSeqRatUp radius readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{MengerCoverCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 6276.1, Definition 6276.2, Definition 6276.3, and Definition 6276.4 seed the finite cover-selection NameCert route for compact metric Real and Completion consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{MengerCoverCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, countable-choice subcover selection, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact metric source exposure, finite cover-request exposure, ProbeBundle finite-subcover verification, StreamName readback, RegSeqRat radius readback, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.MengerCoverCompactnessUp.

Chapter 6277

A Concrete Naming Certificate for $\mathsf{FiniteVolume}^{\uparrow}$

Following LeVeque and Eymard–Gallouet–Herbin, the $\mathsf{FiniteVolume}^{\uparrow}$ packet records conservative finite-mesh balance as BEDC-visible ledger data. It sits over the mesh cells of chapter 3496, the gauge and sum interfaces of chapter 4115 and chapter 1880, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, the real-name seal of chapter 38, and the PDE relation surface of chapter 194. The packet names the finite conservation rows inspected by numerical PDE consumers; it does not import a host solution, mesh-limit convergence theorem, weak formulation quotient, solver trace, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6277.1 (Finite-volume carrier)**.** *A* $\mathsf{FiniteVolume}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$V = (M, F, X, B, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{FinitePartitionMesh}^{\uparrow}$ *cell-mesh row, carrying finitely many control volumes with their displayed incidence names.*
2. *F is the face-incidence row, pairing each interior face with its two adjacent cells and each boundary face with its boundary tag.*
3. *X is the flux ledger, storing one oriented numerical flux entry for each displayed face incidence and the local stencil rows used to read it.*
4. *B is the boundary ledger, storing boundary flux, source, or trace data only along faces marked boundary by F.*
5. *S is the residual and conservation-balance row, recording for each cell the sum of incoming and outgoing face flux entries together with source and residual rows.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the cellwise flux, source, residual, and balance windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row used to compare paired face contributions and displayed residual budgets.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after M, F, X, B, S, R, D have been displayed.*
9. *H records componentwise* hsame *transport for the mesh, incidence, flux, boundary, residual, regular readback, dyadic tolerance, Real seal, replay, provenance, and local naming rows.*

10. *C is the displayed* Cont *replay from a finite cell-balance request through $M, F, X, B, S, R, D, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteVolume}^{\uparrow}$, $\mathsf{FinitePartitionMesh}^{\uparrow}$, $\mathsf{RiemannSumGauge}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Pde}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteVolume}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host PDE solution, unrestricted mesh refinement limit, quotient weak solution, selected trace, hidden solver state, or ambient real completeness theorem.

Lean4 (definition) `BEDC.Derived.FiniteVolumeUp.FiniteVolumeUp`
Lean4 ✓ `BEDC.Derived.FiniteVolumeUp.FiniteVolumeTasteGate single carrier alignment`

Theorem 6277.2 (Finite-volume conservation balance)**.** *For every accepted* $\mathsf{FiniteVolume}^{\uparrow}$ *carrier*

$$V = (M, F, X, B, S, R, D, E, H, C, P, N),$$

the aggregate conservation read over any displayed finite set of cells cancels every interior face flux that is paired by the incidence row F. The visible aggregate balance consists only of boundary-ledger contributions from B, displayed source rows, residual rows from S, and the downstream $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$ *seal data. It does not assert a host PDE solution, unrestricted convergence of a mesh family, selected weak trace, quotient equality of approximate solutions, or ambient completeness.*

Proof. Project the carrier of Definition 6277.1. The row F is the only source of face incidence, so every interior face visible to the packet appears with exactly the two adjacent cell orientations recorded by that row.

Read the oriented flux ledger X through the same incidence data. In a cellwise aggregate, the contribution of an interior face is inspected once from each adjacent cell with opposite orientation, so those two displayed ledger entries cancel as a pair. Boundary faces have no second adjacent cell row, hence they remain visible through the boundary ledger B.

The residual and conservation row S then records exactly what remains after the paired interior cancellations: boundary contributions, displayed source rows, and residual rows. The $\mathsf{RegSeqRat}^{\uparrow}$ readback R, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row D, and $\mathsf{Real}^{\uparrow}$ seal E are downstream of this balance, while H, C, P, N transport, replay, source, and name the same finite route. No excluded host solution, mesh-limit, quotient, trace, or completeness coordinate is present in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteVolume}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteVolume}^{\uparrow}}$, seed)

Scope closed: Definition 6277.1 and Definition 6277.2 seed the conservative finite-volume packet over mesh cells, face incidence, flux ledgers, boundary rows, residual balances, RegSeqRat readback, dyadic tolerance, RealUp seals, transport, replay, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteVolume}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim host PDE solutions, unrestricted mesh-limit convergence, quotient weak-solution equality, selected traces, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked mesh admission, face-incidence pairing, flux-ledger transport, boundary-ledger routing, residual-balance exactness, regular readback, dyadic tolerance, Real-seal admission, replay, provenance, and local naming for BEDC.Derived.FiniteVolumeUp.

Constructive story: A $\mathsf{FiniteVolume}^{\uparrow}$ packet is generated from a FinitePartitionMeshUp cell mesh, face incidence rows, oriented flux ledgers, boundary ledgers, residual conservation balances, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6278

A Concrete Naming Certificate for $\mathsf{OrderedCauchyGroup}^{\uparrow}$

Following Bourbaki on ordered topological groups and Bishop-style completion, the $\mathsf{OrderedCauchyGroup}^{\uparrow}$ packet records the finite interface by which an ordered abelian group completion can be read without quotient representatives, countable choice, trichotomy, or an ambient completeness theorem. It sits after the algebraic source of chapter 42, the order surface of chapter 1378, the completion-routing functor of chapter 423, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the constructive order-compatible completion route needed by Phase Real and Completion consumers; it does not claim host quotient equality of Cauchy filters, selected representatives, decidable order, Archimedean completeness, or a public bridge.

Definition 6278.1 (Ordered Cauchy-group carrier)**.** *An* $\mathsf{OrderedCauchyGroup}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (A, O, M, S, R, F, L, E, H, C, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbGroup}^{\uparrow}$ *source row, naming the group operation, inverse, neutral element, and finite algebraic laws through the accepted abelian-group packet.*
2. *O is the* $\mathsf{Order}^{\uparrow}$ *row on the same source, recording the displayed preorder or order comparisons available to the completion consumer.*
3. *M is the Cauchy-modulus ledger, assigning each requested comparison tolerance to a finite tail window and a group-difference bound.*
4. *S is the source-window row, exposing the finite Cauchy-name observations used by algebraic and order comparisons.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the displayed comparison radii and rational error bounds.*
6. *F is the* $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ *route row, carrying the source packet into a completion-facing object while preserving the visible modulus ledger.*
7. *L is the order-compatibility ledger. It records monotonicity of addition, sign reversal under inverse, and tolerance-stable comparison transport before any completed seal is public.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *or completion-facing seal row reached only after A, O, M, S, R, F, L have been displayed.*

9. *H records componentwise* hsame *transport for algebra, order, modulus, source windows, rational readback, completion route, order-compatibility ledger, seal, replay, provenance, and local naming.*
10. *C records displayed* Cont *replay from every ordered-completion request through $A, O, M, S, R, F, L, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{OrderedCauchyGroup}^{\uparrow}$, $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $A, O, M, S, R, F, L, E, H, C, P, N$. It has no coordinate for a quotient completion, selected Cauchy representative, detached order relation on equivalence classes, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.OrderedCauchyGroupUp.OrderedCauchyGroupTasteGate single carrier alignment`

Theorem 6278.2 (Ordered Cauchy-group NameCert obligations)**.** *For every accepted* $\mathsf{OrderedCauchyGroup}^{\uparrow}$ *carrier*

$$G = (A, O, M, S, R, F, L, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}$ *surface consists exactly of carrier admission for G; abelian-group source exposure through A; order-source exposure through O; Cauchy-modulus exactness through M; finite source-window discipline through S; regular rational readback through R; completion-functor routing through F; order-compatibility accountability through L; terminal seal routing through E; componentwise* hsame *transport through H; displayed* Cont *replay through C;* Pkg *provenance through P; local naming through N; and nonescape excluding quotient completions, selected representatives, detached order relations, choice principles, trichotomy, and ambient completeness.*

Proof. Carrier admission is projection from Definition 6278.1. The algebraic and ordered source rows are A and O, and they are the only coordinates from which the consumer can read group operations or comparisons.

The completion-facing rows are ordered. A request first reads the finite modulus ledger M and the source windows S, then obtains rational comparison bounds through R, and only then enters the completion route F.

The order side is bound by L. It records the displayed compatibility of addition, inverse, and tolerance-stable comparison transport before the terminal seal E is exposed.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains no quotient-completion, selected-representative, detached-order, choice, trichotomy, or ambient-completeness coordinate, the listed obligations exhaust the local certificate surface. □

Theorem 6278.3 (Ordered Cauchy-group ten-field package)**.** *The* $\mathsf{OrderedCauchyGroup}^{\uparrow}$ *certificate instantiates the ten-field roadmap template as follows: traditional target, ordered abelian-group completion with order-compatible operations; BEDC source, the displayed* BHist *route through* $\mathsf{AbGroup}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{CauchyCompletionFunctor}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$*; carrier, G; classifier, componentwise comparison of the displayed rows; stability,* hsame *transport through H; ledger, M, S, R, F, L, E; core theorem list, Definition 6278.2 and Definition 6278.4; exactness, exhaustion of the listed rows; standard bridge, none; status, paper seed certificate for* $\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}$.

Proof. Start with the carrier of Definition 6278.1. The rows A, O, M, S, R, F, L, E provide the source, completion route, order-compatibility ledger, and terminal seal fields.

Apply Definition 6278.2. It supplies classifier stability, ledger exactness, completion routing, order nonescape, replay, provenance, and local naming.

The absence of a bridge row keeps the standard-bridge field empty. The resulting package is therefore a seed certificate for the displayed constructive interface, not a stronger assertion about quotient completions or selected completed elements. □

Theorem 6278.4 (Ordered Cauchy-group order-compatible completion)**.** *Every completion-facing or* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{OrderedCauchyGroup}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{AbGroup}^{\uparrow}, \mathsf{Order}^{\uparrow} \longmapsto M, S, R \longmapsto \mathsf{CauchyCompletionFunctor}^{\uparrow} \longmapsto L \longmapsto E, H, C, P, N.$$

Thus a downstream Phase Real or Completion consumer first reads the abelian-group and order sources, then the Cauchy-modulus ledger, source windows, and regular rational comparison bounds, then the completion-functor route and order-compatibility ledger, and only then the terminal seal and structural rows. The route exports no quotient representative, detached completed order, countable-choice selector, trichotomy, Archimedean completeness theorem, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Let $G = (A, O, M, S, R, F, L, E, H, C, P, N)$ be an accepted packet. The source rows A and O are the only algebra and order entry points, so every completion read begins with the displayed $\mathsf{AbGroup}^{\uparrow}$ and $\mathsf{Order}^{\uparrow}$ surfaces.

The finite Cauchy information is then forced through M, S, R. These rows provide the tail bounds, inspected source windows, and rational comparison budgets that the completion route can consume.

The row F carries that finite information into the $\mathsf{CauchyCompletionFunctor}^{\uparrow}$ surface, and L records the compatibility of the order with completed addition, inverse, and tolerance-stable comparison transport. The terminal seal E is downstream of both rows.

The structural rows H, C, P, N only transport, replay, source, and name this same path. Since no other carrier coordinate stores a completed representative or a detached order relation, every accepted read factors through the displayed route. □

Falsifiable prediction: Every accepted $\mathsf{OrderedCauchyGroup}^{\uparrow}$ consumer that reaches a completion-facing or $\mathsf{Real}^{\uparrow}$ seal while omitting the AbGroup source row, Order source row, Cauchy-modulus ledger, source-window row, RegSeqRat readback, CauchyCompletionFunctor route, order-compatibility ledger, transport row, replay row, provenance row, or local naming row refutes the packet.

Closure status for $\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}$.

Constructive story: An $\mathsf{OrderedCauchyGroup}^{\uparrow}$ packet is generated from an AbGroupUp source row, an OrderUp source row, a Cauchy-modulus ledger, finite source windows, RegSeqRatUp readback, a CauchyCompletionFunctorUp route, an order-compatibility ledger, a RealUp or completion-facing seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6278.1, Definition 6278.2, Definition 6278.3, and Definition 6278.4 seed the finite order-compatible completion route from abelian-group and order sources through Cauchy windows, rational readback, completion routing, compatibility ledger, and terminal seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OrderedCauchyGroup}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected Cauchy representatives, detached completed orders, countable choice, trichotomy, Archimedean completeness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, abelian-group exposure, order-source exposure, Cauchy-modulus exactness, source-window discipline, regular rational readback, completion-functor routing, order-compatibility accountability, terminal seal routing, transport, replay, provenance, and local naming for BEDC.Derived.OrderedCauchyGroupUp.

Chapter 6279

A Concrete Naming Certificate for $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$ packet records the finite witness rows by which StreamName observations, regular rational readback, dyadic tolerance, and a Real seal are read as completion data. It sits over the finite stream interface of chapter 126, the regular rational source of chapter 301, the dyadic tolerance ledger of chapter 291, the nested interval completeness route of chapter 6091, and the terminal real-name seal of chapter 38. The packet is a witness ledger for finite completion reads, not a quotient construction, selected limit, countable-choice schedule, or ambient completeness theorem.

Definition 6279.1 (Real completion witness ledger carrier)**.** *A* $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (S, R, D, K, E, H, T, A, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, carrying the requested windows and the observed stream entries used by the completion read.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row assembled from those same finite windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic comparison radii that certify compatibility of the window and readback data.*
4. *K is the nested-interval or Cauchy handoff row, displaying the finite bridge by which the regular readback is admitted as completion-facing data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after S, R, D, K.*
6. *H records componentwise* hsame *transport for the stream windows, regular readback, dyadic tolerance, handoff, Real seal, replay, provenance, and local naming rows.*
7. *T is the displayed* Cont *replay from a finite completion request through S, R, D, K, E, H, P, N.*
8. *A is the provenance and replay ledger, recording which finite windows and tolerance rows were inspected before the seal was consumed.*
9. *P is the* Pkg *provenance over* $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{NestedIntervalCompleteness}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

10. N is the local $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessLedger}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through these rows. It has no coordinate for quotient stream equality, a host-selected representative, countable choice, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (definition) `BEDC.Derived.RealCompletionWitnessLedgerUp.RealCompletionWitnessLedgerUp`

Theorem 6279.2 (Real completion witness ledger seal)**.** *Every completion-facing* $\mathsf{Real}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$ *carrier factors through a finite route*

$$\mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{NestedIntervalCompleteness}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

More explicitly, the seal E can be consumed only after the finite window row S, regular readback row R, dyadic tolerance row D, and nested-interval or Cauchy handoff row K have been displayed and transported componentwise. The packet exports no quotient stream equality, selected limit representative, countable-choice schedule, trichotomy, or ambient Real completeness.

Proof. Project the carrier of Definition 6279.1. The first visible data are the finite StreamName windows S and the regular rational readback R assembled from the same observations.

Read the dyadic tolerance row D and the completion handoff row K. These rows certify that the finite windows and regular readback are compatible before any completion-facing seal is available; the handoff may be nested-interval shaped or Cauchy shaped, but in either case it is displayed as a finite row.

The terminal Real seal E is downstream of S, R, D, K, and the rows H, T, A, P, N only transport, replay, source, and name that route. Since the carrier has no quotient, selected representative, choice, trichotomy, or host-completeness coordinate, every exported completion read is exactly the displayed finite witness-ledger seal. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionWitnessLedger}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionWitnessLedger}^{\uparrow}}$, seed)

Scope closed: Definition 6279.1 and Definition 6279.2 seed the finite StreamName, RegSeqRat, DyadicRatCore, completion-handoff, RealUp seal, transport, replay, provenance, and local naming witness ledger.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionWitnessLedger}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, host-selected limits, countable choice, trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked StreamName window admission, RegSeqRat readback, DyadicRatCore tolerance, nested-interval or Cauchy handoff, Real-seal admission, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionWitnessLedgerUp.

Constructive story: A $\mathsf{RealCompletionWitnessLedger}^{\uparrow}$ packet is generated from finite StreamName windows, RegSeqRat regular readback, DyadicRatCore tolerance rows, a nested-interval or Cauchy handoff, a RealUp seal, componentwise hsame transport, displayed Cont replay, provenance rows, Pkg sourcing, and a local NameCert row.

Chapter 6280

A Concrete Naming Certificate for $\mathsf{RegularCauchyRealReflection}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ packet names the finite route by which a regular rational Cauchy sequence can be read as a Real-facing seal without importing quotient stream equality or host real equality. It sits after the regular rational source of chapter 301, the finite-window surface of chapter 126, the quotient-discipline boundary of chapter 1820, the dyadic tolerance core of chapter 291, and the terminal seal of chapter 38. The packet is a concrete L10 reflection route from classifier data to the public Real seal; it is not an assertion of ambient completeness, a selected representative theorem, or a host-limit construction.

Definition 6280.1 (RegularCauchyRealReflection carrier)**.** *A* $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (G, S, E, D, R, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *source row naming the regular rational Cauchy sequence whose Real-facing reflection is being inspected.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected prefixes, tail windows, and replay schedule for G.*
3. *E is the* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *classifier row recording exactly the displayed equivalence comparisons admitted for the regular sequence route.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-window row, with dyadic budgets aligned to the regular-Cauchy classifier comparisons named by E.*
5. *R is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row. It is visible only after G, S, E, D display the regular source, finite windows, admissible Cauchy-equivalence comparisons, and dyadic tolerance windows.*
6. *H records componentwise* hsame *transport for the regular source, stream windows, equivalence classifier, dyadic tolerance windows, Real seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a Real-facing reflection request through G, S, E, D, R, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyRealReflection}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}$ $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRealReflection}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $G, S, E, D, R, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative of the equivalence class, a host real limit, ambient real equality, countable choice, or an external completeness theorem.

Lean4 ✓ `BEDC.Derived.RegularCauchyRealReflectionUp.RegularCauchyRealReflectionTasteGate single carrier alignment`

Theorem 6280.2 (RegularCauchyRealReflection NameCert obligations)**.** *For every accepted* RegularCauchyRealRe *carrier*

$$M = (G, S, E, D, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyRealReflection}^{\uparrow}}$ *surface consists exactly of carrier admission for M, regular-Cauchy classifier exposure through G and E, dyadic tolerance-window exactness through D, terminal* $\mathsf{Real}^{\uparrow}$ *seal reflection through R, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. The certificate is stable under replay of the same displayed stream windows and tolerance budgets, and it never exports quotient stream equality or host real equality.*

Proof. Project the carrier of Definition 6280.1. The accepted source is the regular rational row G, and the only observable finite data for that source are the windows named by S.

The classifier rows are explicit. The Cauchy-equivalence row E records the admissible comparisons, while the dyadic tolerance row D records the exact tolerance windows used by those comparisons. No extra equality predicate is available outside these rows.

The Real-facing row R is therefore terminal for the displayed route. It can be read only after G, S, E, D have exposed the regular source, finite windows, classifier comparisons, and dyadic tolerances.

The remaining rows do not add analytic content. They transport the same data by H, replay the same route by C, record source accountability by P, and bind the local name by N. Since no row stores quotient stream equality, a selected representative, a host real limit, ambient real equality, countable choice, or external completeness, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ reflection factors through one RegSeqRatUp source row, one StreamNameUp finite-window row, one CauchyEquivalenceSetoidUp classifier row, one DyadicRatCoreUp tolerance-window row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: regseqrat, streamname, cauchy-equivalence-setoid, dyadicratcore, real: $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ is not bijective to any listed sibling because it packages the regular-Cauchy classifier, finite windows, dyadic tolerance windows, terminal Real seal, transport, replay, provenance, and local NameCert as one reflection certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyRealReflection}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ packet is generated from a RegSeqRatUp source row, StreamNameUp finite windows, a CauchyEquivalenceSetoidUp classifier row, DyadicRatCoreUp tolerance windows, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RegularCauchyRealReflection}^{\uparrow}}$, seed)

Scope closed: Definition 6280.1 and Definition 6280.2 seed the finite reflection route from regular rational Cauchy classifier data to the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RegularCauchyRealReflection}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ambient Real completeness, quotient equality, selected representatives, host limits, host real equality, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regular-Cauchy classifier exposure, dyadic tolerance-window exactness, Real seal reflection, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularCauchyR

Chapter 6281

A Concrete Naming Certificate for $\mathsf{InitialUniformity}^{\uparrow}$

Following Bourbaki on uniform structures and initial uniformities, the $\mathsf{InitialUniformity}^{\uparrow}$ packet names the finite pullback-uniformity route determined by a displayed map into a target uniform space. It sits after the Real-induced uniformity of chapter 1624, the abstract uniform entourage basis of chapter 2331, and the completion-facing handoff of chapter 3560. The packet records only source rows, target entourages, displayed maps, pulled-back entourage rows, transport, replay, provenance, and local naming; it is not an ambient topology, quotient uniformity, selected representative principle, open-cover compactness theorem, or countable-choice device.

Definition 6281.1 (InitialUniformity carrier)**.** *An* $\mathsf{InitialUniformity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (S, T, F, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the source carrier row. It records the points or names on which the pulled-back uniformity is being inspected.*
2. *T is the target uniformity row. It exposes the finite target-entourage requests admitted by the target* $\mathsf{UniformSpace}^{\uparrow}$ *or by a basis packet such as* $\mathsf{UniformEntourageBasis}^{\uparrow}$*.*
3. *F is the displayed map row. It names the finite graph, readback, or continuation by which a source request is sent into the target carrier before an entourage is tested.*
4. *E is the pulled-back entourage row. Each entry records a target entourage from T together with the displayed source-pair request whose two F-images are tested against that target entourage.*
5. *H records componentwise* hsame *transport for S, T, F, E, C, P, N.*
6. *C records displayed* Cont *replay from a source-pair request through F and then through the target-entourage read in T.*
7. *P is the* Pkg *provenance over* $\mathsf{InitialUniformity}^{\uparrow}$*,* $\mathsf{UniformSpace}^{\uparrow}$*,* $\mathsf{UniformEntourageBasis}^{\uparrow}$*,* $\mathsf{UniformSpaceCompletio}$ $\mathsf{RealUniformStructure}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{InitialUniformity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through S, T, F, E, H, C, P, N. It has no coordinate for an ambient topology on S, a quotient uniformity, a selected representative family, a host continuity theorem, open-cover compactness, countable choice, or external completeness.

Theorem 6281.2 (InitialUniformity pullback entourage stability)**.** *Let $I = (S, T, F, E, H, C, P, N)$ be an accepted* $\mathsf{InitialUniformity}^{\uparrow}$ *carrier. If two displayed source-pair requests are componentwise* hsame *through H, and the target-entourage read is replayed through the displayed* Cont *route C, then the pulled-back entourage row E returns the same accepted pullback read. Thus pullback-entourage stability is internal to the displayed source row, target uniformity row, map row, pulled-back entourage row, transport row, and replay row.*

Proof. Project the carrier of Definition 6281.1. A pullback-entourage read begins with a source-pair request in S, sends both components through the displayed map row F, and then tests the resulting target pair against the target-entourage row T.

Use componentwise hsame transport in H on the source request, the two displayed map reads, and the target-entourage request. These transports preserve the visible coordinates of the packet, so the transported read still lands in the same target-entourage row.

Replay the transported request through C. Since C records only the route from source-pair data through F into T, the replay reaches the same pulled-back row E and cannot introduce an ambient topology, quotient uniformity, selected representative, or choice coordinate. □

Theorem 6281.3 (InitialUniformity NameCert obligations)**.** *For the carrier of Definition 6281.1, the local* $\mathsf{NameCert}_{\mathsf{InitialUniformity}^{\uparrow}}$ *surface consists exactly of five finite obligations:*

1. *carrier exposure for $I = (S, T, F, E, H, C, P, N)$;*
2. *target-entourage exposure through the target uniformity row T;*
3. *map compatibility requiring every pulled-back request in E to read the displayed map row F before target-entourage testing;*
4. *pulled-back entourage stability under componentwise* hsame *transport and displayed* Cont *replay as in Definition 6281.2;*
5. *refusal of ambient topology on S, quotient uniformity, selected representatives, host continuity, open-cover compactness, countable choice, and external completion principles.*

Lean4 ✓ `BEDC.Derived.InitialUniformityUp.InitialUniformityCarrier namecert obligations`

Proof. Carrier exposure is the projection of the finite tuple $I = (S, T, F, E, H, C, P, N)$. The target-entourage exposure is the projection of T, and map compatibility is the requirement that every entry of E is read only after its two source components have been sent through F.

The stability obligation is exactly Definition 6281.2. It binds source-pair transport, target-entourage transport, and continuation replay to the same pulled-back entourage row.

The refusal obligation follows from the carrier inventory. No row stores an ambient topology, quotient uniformity, selected representative family, host continuity theorem, open-cover compactness proof, countable-choice selector, or external completion object, so the local certificate exports only the finite pullback-uniformity route. □

Falsifiable prediction: Every accepted $\mathsf{InitialUniformity}^{\uparrow}$ pullback-entourage read factors through one source row, one target uniformity row, one displayed map row, one pulled-back entourage row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: real-uniform-structure, uniform-entourage-basis, uniform-space-completion: $\mathsf{InitialUniformity}^{\uparrow}$ is not bijective to any listed sibling because it packages the source carrier, target uniformity, displayed map, pulled-back entourage row, transport, replay, provenance, and local NameCert as one pullback-uniformity certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{InitialUniformity}^{\uparrow}}$.

Constructive story: An $\mathsf{InitialUniformity}^{\uparrow}$ packet is generated from a source carrier row, a target uniformity row, a displayed map row, pulled-back entourage rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{InitialUniformity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6281.1, Definition 6281.2, and Definition 6281.3 seed the finite pullback-uniformity boundary for target-entourage consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{InitialUniformity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient topology, quotient uniformity, selected representatives, host continuity, open-cover compactness, countable choice, external completion, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier exposure, target-entourage exposure, map compatibility, pulled-back entourage stability, transport, replay, provenance, local naming, and nonescape for $\mathsf{InitialUniformity}^{\uparrow}$.

Chapter 6282

A Concrete Naming Certificate for $\mathsf{UniformCompletionReflector}^{\uparrow}$

Following Bourbaki and Kelley on uniform spaces and completion reflectors, the $\mathsf{UniformCompletionReflector}^{\uparrow}$ packet names the finite BEDC surface by which a displayed uniform source is sent to a completion target through certified countable names and regular Cauchy data. It sits after the quotient-discipline boundary of chapter 1820, the completion carrier of chapter 95, the uniform source of chapter 339, the uniform Cauchy filter route of chapter 5117, and the Real-facing uniform surface of chapter 1624. The packet is a NameCert interface for the reflector route; it is not a host quotient completion, not a choice-selected limit, and not an ambient real completeness theorem.

Definition 6282.1 (UniformCompletionReflector carrier)**.** *A* $\mathsf{UniformCompletionReflector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (X, W, Y, J, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the uniform source row, naming the uniform-space carrier, entourage basis, and separatedness boundary inspected by the reflector.*
2. *W is the Cauchy filter or finite-window row, recording the certified countable names, regular Cauchy windows, and tolerance comparisons admitted from X.*
3. *Y is the completion target row, displaying the completed uniform carrier only through the completion data certified by W.*
4. *J is the dense embedding row from X into Y, including the finite image windows and density witnesses visible to downstream consumers.*
5. *Q is the universal arrow row: every displayed uniformly continuous map from X into a complete separated target factors through the same Y, J surface by a unique displayed route.*
6. *H records componentwise* hsame *transport for the source, Cauchy window, target, dense embedding, universal arrow, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a completion-reflector request through X, W, Y, J, Q, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{UniformCompletionReflector}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformlyCauchyFilter}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCompletionReflector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, W, Y, J, Q, H, C, P, N$. It has no coordinate for quotient completion objects, selected Cauchy representatives, countable choice, a host limit operator, ambient equality on completed points, or an untracked universal map.

Lean4 ✓ `BEDC.Derived.UniformCompletionReflectorUp.UniformCompletionReflectorTasteGate single carrier alignment`

Theorem 6282.2 (UniformCompletionReflector NameCert obligations)**.** *For every accepted* UniformCompletionRef *carrier*

$$U = (X, W, Y, J, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCompletionReflector}^{\uparrow}}$ *surface consists exactly of uniform source admission through X, certified Cauchy-window readback through W, completion-target display through Y, dense embedding through J, universal arrow exactness through Q, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N. The certificate therefore exposes the same finite route to completion consumers, Real-facing consumers, and countable-name consumers without exporting quotient completion equality or selected limits.*

Lean4 ✓ `BEDC.Derived.UniformCompletionReflectorUp.UniformCompletionReflectorNameCertObligations`

Proof. Project the carrier of Definition 6282.1. The source row X fixes the uniform object that can be inspected, while W records the only Cauchy filter or regular-window evidence available to the reflector.

The completion target and dense embedding rows are downstream of those inputs. Thus Y and J can be read only after the source and Cauchy-window rows have been displayed, and neither row carries a host quotient or selected representative.

The universal arrow row Q is part of the same packet. Any factorization read must use the displayed Y, J route and cannot replace it with an ambient completion object or a choice-selected limit.

Finally H, C, P, N transport, replay, source, and name the same rows componentwise. Since the carrier has no additional equality, selector, or host-limit coordinate, the listed rows exhaust the NameCert obligations. □

Theorem 6282.3 (UniformCompletionReflector universal arrow)**.** *Every public universal-arrow read exported by an accepted* $\mathsf{UniformCompletionReflector}^{\uparrow}$ *carrier factors through the displayed source row X, Cauchy-window row W, completion target Y, dense embedding J, and universal row Q. A consumer may therefore replay the reflector property only as a finite route through H, C, P, N; it receives no host quotient completion, selected Cauchy representative, countable-choice selector, or ambient extension theorem outside the packet.*

Proof. Start from Definition 6282.2. It places the universal row Q after the uniform source, Cauchy-window, target, and dense-embedding rows.

The factorization route is consequently displayed. A consumer can inspect the uniformly continuous source map only through X and W, then read its extension through the public Y, J, Q rows.

Transport and replay preserve that order. The rows H, C, P, N move the same finite packet through classifier equivalence, replay the same route, record provenance, and bind the local name, so no endpoint can escape into an evidence-free completion or an untracked universal map. □

Falsifiable prediction: Any claimed $\mathsf{UniformCompletionReflector}^{\uparrow}$ universal-arrow read that reaches a completed target without the displayed uniform source row, Cauchy-window row, dense embedding row, and universal-arrow row refutes the carrier.

Independence witness: uniformspace, uniformly-cauchy-filter, cauchy-equivalence-setoid, completion, real-uniform-structure: $\mathsf{UniformCompletionReflector}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries source uniformity, Cauchy windows, a completion target, dense embedding, universal arrow, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCompletionReflector}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCompletionReflector}^{\uparrow}$ packet is generated from a uniform source row, certified Cauchy-window row, completion target row, dense embedding row, universal-arrow row, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCompletionReflector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6282.1, Definition 6282.2, and Definition 6282.3 seed the finite reflector route from uniform sources and Cauchy windows to completion targets and universal arrows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCompletionReflector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient uniform completion, quotient equality, selected representatives, countable choice, host limits, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Cauchy-window readback, dense embedding exactness, universal-arrow exactness, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.UniformCompletionReflectorU

Chapter 6283

A Concrete Naming Certificate for $\mathsf{RealCompletionAdjoint}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RealCompletionAdjoint}^{\uparrow}$ packet records the finite universal-property surface by which regular rational Cauchy data enters the Real-facing completion seal and by which Real-facing maps are read back as regular-Cauchy extensions. It sits over the regular rational source of chapter 301, the Real seal of chapter 38, the finite reflection route of chapter 6280, and the adjacent Bishop completion-adjoint packet of chapter 6057. The certificate is a displayed adjoint boundary, not a quotient-completion theorem, selected-limit construction, or host equality principle.

Definition 6283.1 (Real completion adjoint carrier)**.** *A* $\mathsf{RealCompletionAdjoint}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, S, Q, R, E, U, X, K, H, C, P, N)$$

with the following displayed rows:

1. *D is the dense-unit row naming the finite unit access from rational data into the completion-facing route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used by every unit, reflection, and extension read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy source row.*
4. *R is the* $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ *row that exposes the route from Q and S to the Real-facing seal.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after D, S, Q, R have been projected.*
6. *U is the extension row recording the finite continuation by which a Real-facing target reads a regular-Cauchy source through the unit and reflection rows.*
7. *X is the uniqueness row: two displayed extensions agreeing on the dense-unit and regular-Cauchy source rows are identified only by componentwise certificate data.*
8. *K records local compatibility between the dense-unit row, the reflection row, the extension row, and the uniqueness row.*
9. *H records componentwise* hsame *transport for $D, S, Q, R, E, U, X, K, C, P, N$.*
10. *C records displayed* Cont *replay of the route*

$$D, S, Q \longmapsto R \longmapsto E \longmapsto U, X, K, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{RealCompletionAdjoint}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{RegularCauchyRealReflection}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{RealCompletionAdjoint}^{\uparrow}}$ *row.*

The accepted carrier is exactly the displayed packet A. It has no coordinate for quotient stream equality, a selected limit, host real equality, countable choice, trichotomy, open-cover compactness, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.RealCompletionAdjointUp.RealCompletionAdjointTasteGate single carrier alignment`

Theorem 6283.2 (Real completion adjoint NameCert obligations)**.** *For every accepted* $\mathsf{RealCompletionAdjoint}^{\uparrow}$ *carrier $A = (D, S, Q, R, E, U, X, K, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCompletionAdjoint}^{\uparrow}}$ *obligations are exhausted by carrier admission for the dense unit, finite stream windows, regular-Cauchy source, Real reflection route, terminal Real seal, extension row, uniqueness row, local compatibility, transport, replay, provenance, and local naming rows. No obligation may call on quotient stream equality, a selected representative, host real equality, countable choice, trichotomy, open-cover compactness, ambient Real completeness, or an external adjunction theorem.*

Proof. Project the carrier of Definition 6283.1. The visible analytic rows are D, S, Q, R, E, U, X, K, and the visible structural rows are H, C, P, N.

Read the source side first. The dense-unit row D and finite windows S expose the rational access data, while Q supplies the regular-Cauchy source. The reflection row R is then the only route from that source to the terminal Real seal E.

Read the target side next. The extension row U consumes the same displayed unit and reflection route, and the uniqueness row X compares two such extensions only through componentwise displayed data. The compatibility row K records that the unit, reflection, extension, and uniqueness rows share the same finite source route.

Finally transport and replay by H and C, and bind provenance and naming by P, N. Since no excluded analytic coordinate appears in the carrier, the local certificate has no remaining place to import one. □

Theorem 6283.3 (Real completion adjoint reflection)**.** *Let $A = (D, S, Q, R, E, U, X, K, H, C, P, N)$ be an accepted* $\mathsf{RealCompletionAdjoint}^{\uparrow}$ *carrier. Every Real-facing read exported by A factors through the displayed finite route*

$$D, S, Q \longmapsto R \longmapsto E.$$

The extension row U and uniqueness row X may replay this route, but they cannot bypass the $\mathsf{RegSeqRat}^{\uparrow}$ *source row, the* $\mathsf{StreamName}^{\uparrow}$ *finite windows, or the* $\mathsf{RegularCauchyRealReflection}^{\uparrow}$ *reflection row.*

Proof. Start with the source projection. The carrier exposes the dense unit, finite windows, and regular-Cauchy source as D, S, Q, so any Real-facing read has an explicit finite source spine.

Apply the reflection row R. By construction it is the only row that turns the regular-Cauchy source and finite windows into a terminal $\mathsf{Real}^{\uparrow}$ seal. The seal row E is therefore downstream of D, S, Q, R.

Now inspect U and X. The extension row reads the same reflected source, and the uniqueness row compares extensions only after that read has occurred. Componentwise transport, replay, provenance, and local naming preserve this order, so no accepted route can jump directly to the Real seal. □

Theorem 6283.4 (Real completion adjoint uniqueness)**.** *Let* $A = (D, S, Q, R, E, U, X, K, H, C, P, N)$ *and* $A' = (D', S', Q', R', E', U', X', K', H', C', P', N')$ *be accepted* $\mathsf{RealCompletionAdjoint}^{\uparrow}$ *carriers. If the dense-unit, finite-window, regular-Cauchy source, reflection, Real seal, extension, compatibility, transport, replay, provenance, and local naming rows are componentwise* hsame*, then the uniqueness rows* X *and* X' *expose the same completion-adjoint comparison. The comparison is certificate-local and introduces no quotient stream equality, selected limit, host real equality, choice schedule, or ambient completeness row.*

Proof. Project both carriers and match the source rows. The componentwise hsame assumptions identify the dense unit, finite windows, regular-Cauchy source, reflection route, and Real seal as displayed certificate data.

Match the extension rows next. Since U and U' read the same reflected source route, their comparison is already constrained by the displayed rows D, S, Q, R, E and their primed counterparts.

The uniqueness rows X and X' therefore have only one local comparison surface: the componentwise certificate comparison replayed through K, H, C, P, N and K', H', C', P', N'. Because neither carrier contains an excluded analytic row, the uniqueness comparison remains internal to the NameCert packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{RealCompletionAdjoint}^{\uparrow}}$.

Constructive story: A $\mathsf{RealCompletionAdjoint}^{\uparrow}$ packet is generated from dense-unit access, finite StreamName windows, a RegSeqRat source, a RegularCauchyRealReflection route, a Real seal, extension and uniqueness rows, local compatibility, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealCompletionAdjoint}^{\uparrow}}$, seed)

Scope closed: Definition 6283.1, Definition 6283.2, Definition 6283.3, and Definition 6283.4 seed the completion-adjoint packet over regular-Cauchy source data and the Real seal.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealCompletionAdjoint}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.RealCompletionAdjointUp.RealCompletionAdjointTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected completed limits, host real equality, countable choice, trichotomy, open-cover compactness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, reflection factorization, uniqueness comparison, seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealCompletionAdjointUp.

Chapter 6284

A Concrete Naming Certificate for $\mathsf{UnifiedEpistemicObject}^{\uparrow}$

The $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ packet realizes the unified closure pattern of section 9462.20 as finite NameCert data. It gathers generated material, classifier surface, certificate rows, ledger accounting, theorem-DNA roles, hardening profile, and scope seal into one auditable object. The packet sits beside the classifier-novelty ledger of chapter 6241, the scoped explanation ledger of chapter 6249, and the theorem-DNA score surface of chapter 6243; it does not claim positive discovery without a later TasteGate witness.

Definition 6284.1 (Unified epistemic object carrier)**.** *A* $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (G, R, C, N, L, D, F, O, T, H, K, P, M)$$

with the following displayed rows:

1. *G is the generated-material row, covering observed material, proof traces, learned traces, theorem candidates, or intervention records.*
2. *R is the record row that fixes the local source scope and replayable material boundary.*
3. *C is the classifier row for the stated source scope.*
4. *N is the certificate row, carrying the local NameCert or Tensor NameCert data for C.*
5. *L is the ledger row recording residue from identification, compression, projection, thresholding, semantic assignment, verification, intervention, and scope refusal.*
6. *D is the theorem-DNA row recording role-complete loci for the object.*
7. *F is the hardening-profile row, recording which loci are hardened, which remain ledger-only, and which backend boundary is being used.*
8. *O is the scope-seal row, naming source scope, target scope, claim kind, not-claimed boundary, and closure route.*
9. *T is the classifier-nonescape row: every accepted classifier read must pass through G, R, C, N, L, D, F, O.*
10. *H records componentwise* hsame *transport for all rows.*
11. *K records displayed* Cont *replay from generated material through classifier, certificate, ledger, DNA, hardening, and seal rows.*

12. *P is the* Pkg *provenance over* $\mathsf{UnifiedEpistemicObject}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

13. *M is the local* $\mathsf{NameCert}_{\mathsf{UnifiedEpistemicObject}^{\uparrow}}$ *row.*

The accepted carrier has no coordinate for a bare classifier claim, an unledgered hardening claim, a positive-discovery verdict, a Nontrivial instance, a FieldFaithful instance, or a TasteGate result.

Theorem 6284.2 (Unified epistemic object NameCert obligations)**.** *For every accepted* $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ *carrier* $U = (G, R, C, N, L, D, F, O, T, H, K, P, M)$*, the local* $\mathsf{NameCert}_{\mathsf{UnifiedEpistemicObject}^{\uparrow}}$ *obligations are exhausted by generated material, record scope, classifier, certificate, ledger, theorem-DNA, hardening-profile, scope-seal, classifier-nonescape, transport, replay, provenance, and local naming rows. In particular, the carrier does not certify positive discovery, Nontriviality, or FieldFaithfulness merely by packaging the rows.*

Proof. Project the carrier of Definition 6284.1. The material and record rows G, R determine the source boundary, and the classifier row C is readable only relative to that boundary.

Read the certificate and ledger rows next. The row N records the certificate surface for C, while L records the residue that prevents classifier assertions from escaping their declared accounting surface.

The theorem-DNA, hardening, and seal rows then place the object in its role, backend, and scope boundaries. The nonescape row T says that classifier reads must pass through all of these rows before any public claim can be exported.

The remaining structural rows H, K, P, M transport, replay, account for provenance, and name the same packet. None of them supplies a TasteGate verdict, Nontrivial instance, FieldFaithful instance, or positive-discovery result, so those obligations remain outside this carrier. □

Theorem 6284.3 (Unified epistemic object classifier nonescape)**.** *Let* $U = (G, R, C, N, L, D, F, O, T, H, K, P, M)$ *be an accepted* $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ *carrier. Every classifier-facing read exported by* U *factors through*

$$G, R \longmapsto C \longmapsto N, L, D, F, O \longmapsto T, H, K, P, M.$$

Consequently a downstream consumer cannot read a bare classifier, unledgered verification claim, unsealed theorem-DNA packet, or positive-discovery verdict from the carrier.

Proof. Start with the generated-material and record rows. They fix the local source scope before a classifier can be read.

Apply the classifier and certificate rows. The classifier row C is named by N, and the ledger row L records residue from the relevant source and semantic operations.

Use the theorem-DNA, hardening-profile, and scope-seal rows. They bind role completeness, backend exposure, and public claim scope before any classifier-facing output may leave the packet.

Finally apply the classifier-nonescape row T and replay through H, K, P, M. Since the carrier has no row for a bare classifier, unledgered hardening claim, unsealed DNA packet, or TasteGate verdict, none of those objects can be exported by an accepted read. □

Falsifiable prediction: A downstream classifier report that reads a $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ classifier without generated material, record scope, certificate, ledger, theorem-DNA, hardening profile, and scope seal refutes the classifier-nonescape row.

Independence witness: classifier-novelty-ledger, scoped-explanation-ledger, theorem-dna-score, hard-certificate-layer, audit-axis-separation: none is bijective to $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ because

each sibling contributes only part of the generated-material, classifier, certificate, ledger, theorem-DNA, hardening-profile, scope-seal, nonescape, transport, replay, provenance, and local naming packet.

Closure status for $\mathsf{NameCert}_{\mathsf{UnifiedEpistemicObject}^{\uparrow}}$.

Constructive story: A $\mathsf{UnifiedEpistemicObject}^{\uparrow}$ packet is generated from material, record, classifier, certificate, ledger, theorem-DNA, hardening-profile, scope-seal, classifier-nonescape, transport, replay, provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UnifiedEpistemicObject}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6284.1, Definition 6284.2, and Definition 6284.3 seed the finite unified epistemic object packet without a discovery verdict.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UnifiedEpistemicObject}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close positive discovery, Nontriviality, FieldFaithfulness, TasteGate acceptance, Lean verification, bridge checking, or any classifier shift beyond the displayed carrier rows.

Upgrade path: Obligation closure requires checked carrier admission, classifier nonescape, ledger closure, theorem-DNA compatibility, hardening-profile accounting, scope-seal replay, and later TasteGate work for Nontrivial and FieldFaithful instances.

Chapter 6285

A Concrete Naming Certificate for $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$

Following Alexander and Kelley, the $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$ packet records a finite-subbase compactness route as BEDC naming data. It sits next to the compact metric finite-net surface of chapter 129, the finite-product compactness route of **??**, and the finite-subcover criterion of chapter 3503. The packet names only displayed subbase rows, finite obstruction ledgers, transport, replay, provenance, and local naming; it does not assert full choice-based compactness, arbitrary open-cover compactness, unrestricted Tychonoff compactness, quotient equality of covers, or a host subcover selector.

Definition 6285.1 (AlexanderSubbaseTheorem carrier)**.** *An* $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, S, F, U, B, H, C, P, N)$$

with the following displayed rows:

1. *X is the compactness-facing source row, read through the same finite-net discipline consumed by* $\mathsf{CompactMetric}^{\uparrow}$ *and* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$*.*
2. *S is the subbase row, recording the named subbasic opens available to the packet.*
3. *F is the finite subbase obstruction ledger, listing the finite subfamilies inspected before a cover-facing conclusion is exposed.*
4. *U is the finite cover-read row obtained from the displayed subbase ledger rather than from an arbitrary open cover.*
5. *B is the compact-product boundary row that may hand the packet to finite* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$ *consumers.*
6. *H records componentwise* hsame *transport for source, subbase, finite obstruction, finite cover, product-boundary, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the source row through subbase rows, finite obstruction checks, finite cover reads, product-boundary handoff, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{TychonoffFiniteProduct}^{\uparrow}$*,* $\mathsf{FiniteSubcoverCriterion}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$ *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{AlexanderSubbaseTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, S, F, U, B, H, C, P, N$. It has no coordinate for a choice-selected ultrafilter, arbitrary open-cover compactness, unrestricted product compactness, quotient equality of cover families, or a host finite-subcover operation.

Theorem 6285.2 (AlexanderSubbaseTheorem NameCert obligations)**.** *For every accepted* AlexanderSubbaseTheor *carrier $A = (X, S, F, U, B, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AlexanderSubbaseTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission, subbase exposure through S, finite obstruction exactness through F, finite cover-read exactness through U, finite product-boundary handoff through B, componentwise transport through H, displayed replay through C, provenance through P, local naming through N, and nonescape from full choice, arbitrary open-cover compactness, unrestricted Tychonoff compactness, quotient cover equality, and host subcover selection.*

Lean4 ✓ `BEDC.Derived.AlexanderSubbaseTheoremUp.AlexanderSubbaseTheoremCarrier namecert obligations`

Proof. Project the carrier of Definition 6285.1. The only compactness-facing source is X, the only subbase data are in S, and the only finite inspection rows are F and U.

The finite-product handoff is the displayed row B. It can be consumed by the finite product compactness route of **??**, but it does not add an unrestricted product theorem or a choice-selected product point.

Transport, replay, provenance, and local naming are exactly H, C, P, N. Since the carrier has no coordinate for full choice, arbitrary open-cover compactness, unrestricted Tychonoff compactness, quotient cover equality, or a host subcover selector, none of those objects can become a local NameCert obligation. □

Theorem 6285.3 (AlexanderSubbaseTheorem finite-subcover route)**.** *Every compactness-facing read from an accepted* $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$ *packet must display the source row X, the subbase row S, the finite obstruction ledger F, and the finite cover-read row U before any product-boundary handoff B or compactness consumer is exposed. The route factors through the finite-subcover criterion of chapter 3503, not through arbitrary open-cover compactness.*

Proof. Begin with the carrier in Definition 6285.1. Its compactness route is ordered by the displayed rows $X, S, F, U, B, H, C, P, N$.

Apply the finite-subcover criterion of chapter 3503. The cover-facing read therefore consumes a finite criterion row before a compactness consumer can inspect the result.

Use Definition 6285.2 for nonescape. The packet has no arbitrary open-cover row, no host subcover operation, and no full-choice coordinate, so the exported route is the displayed finite-subbase route through X, S, F, U and its structural rows. □

Closure status for $\mathsf{NameCert}_{\mathsf{AlexanderSubbaseTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{AlexanderSubbaseTheorem}^{\uparrow}$ packet is generated from a compactness-facing source row, a displayed subbase row, a finite obstruction ledger, a finite cover-read row, a finite product-boundary handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{AlexanderSubbaseTheorem}^{\uparrow}}$, seed)

Scope closed: Definition 6285.1, Definition 6285.2, and Definition 6285.3 seed the finite-subbase compactness packet over compact metric, finite-product compactness, and finite-subcover criterion sibling rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{AlexanderSubbaseTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close full choice-based compactness, arbitrary open-cover compactness, unrestricted Tychonoff compactness, quotient equality of cover families, host finite-subcover selection, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, subbase exposure, finite obstruction exactness, finite cover-read exactness, finite product-boundary handoff, transport, replay, provenance, and local naming for BEDC.Derived.AlexanderSubbaseTheoremUp.

Chapter 6286

A Concrete Naming Certificate for $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$

Following Bishop and Bridges and Kelley, the $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ packet records the finite uniform-Cauchy completion route by which regular rational names, finite windows, and dyadic tolerances are read before any Real or Completion consumer may inspect a completed endpoint. It sits beside the uniform-Cauchy packet of chapter 3618, the window and readback surfaces of chapter 126 and chapter 301, the tolerance core of chapter 291, the terminal Real seal of chapter 38, the Cauchy sequence space of chapter 406, and the theorem-level completion handoff of chapter 3484. The packet is a displayed NameCert interface for Bishop-style uniform Cauchy completion, not a quotient completion object, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6286.1 (Bishop uniform-Cauchy completion carrier)**.** *A* $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (A, W, R, D, E, S, B, H, C, P, N)$$

with the following displayed rows:

1. *A is the uniform-Cauchy source row, naming the inspected uniform family, entourage request, and finite tail schedule admitted from the* $\mathsf{UniformCauchyCompletion}^{\uparrow}$ *surface.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row whose finite observations open the source and tail data.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that records the dyadic precision at which W and R are compared.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the source, window, readback, and tolerance rows have been displayed.*
6. *S is the* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff row recording how the same windowed readback is consumed by the space of Cauchy names.*
7. *B is the* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *handoff row recording the theorem-level completion route generated from the displayed uniform-Cauchy source.*
8. *H records componentwise* hsame *transport for $A, W, R, D, E, S, B, C, P, N$.*

9. *C records displayed* Cont *replay of the route from the source row through the window, readback, tolerance, Real seal, CauchySequenceSpace handoff, and BishopCompletionTheorem handoff rows.*

10. *P is the* Pkg *provenance over* $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$, $\mathsf{UniformCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, $\mathsf{BishopCompletionTheorem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyCompletion}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient names, selected representatives, a limit selector, countable choice, host real equality, hidden completion points, or ambient Real completeness.

Theorem 6286.2 (Bishop uniform-Cauchy completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ *carrier* $U = (A, W, R, D, E, S, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyCompletion}^{\uparrow}}$ *surface consists exactly of the uniform-Cauchy source row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readbacks,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger,* $\mathsf{Real}^{\uparrow}$ *seal row,* $\mathsf{CauchySequenceSpace}^{\uparrow}$ *handoff,* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every completion consumer of the packet must expose those rows before it may read a Real seal or completion-facing endpoint.*

Lean4 ✓ `BEDC.Derived.BishopUniformCauchyCompletionUp.BishopUniformCauchyCompletionCarrier namecert obligations`

Proof. Project the carrier of Definition 6286.1. The only analytic rows are A, W, R, D, E, S, B, and the only structural rows are H, C, P, N.

The source row A supplies the uniform-Cauchy request, but it is not itself a completed point. The finite window row W, readback row R, and dyadic tolerance row D are the displayed data that make the source inspectable.

The endpoint rows are downstream of those displays. The Real seal E, the CauchySequenceSpace handoff S, and the BishopCompletionTheorem handoff B can be read only after A, W, R, D have been projected.

Finally, H, C, P, N transport, replay, source, and name the same rows componentwise. Since the carrier contains no quotient, selector, choice schedule, host equality, or ambient completeness coordinate, the listed rows exhaust the local certificate. □

Theorem 6286.3 (Bishop uniform-Cauchy completion Real seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, *or* $\mathsf{BishopCompletionTheorem}^{\uparrow}$ *consumer read exported by an accepted* $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ *carrier factors through the displayed finite route*

$$\mathsf{UniformCauchyCompletion}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow}$$

before it reaches the Real seal row E, the CauchySequenceSpace handoff row S, or the BishopCompletionTheorem handoff row B. The route exports no quotient completion, selected limit, countable-choice schedule, host real equality, or ambient completeness witness.

Proof. Read the source side first. The accepted carrier displays A as the uniform-Cauchy source and then exposes it through the finite window, regular readback, and dyadic tolerance rows W, R, D.

The seal and completion rows are therefore not primitive endpoints. The row E is a Real-facing seal read after A, W, R, D, while S and B record the sibling CauchySequenceSpace and BishopCompletionTheorem handoffs over the same displayed data.

The structural rows H, C, P, N preserve that order under transport and replay. Thus every downstream read factors through the named finite route and cannot obtain any excluded completion mechanism from the packet. □

Falsifiable prediction: A claimed $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchySequenceSpace}^{\uparrow}$, or $\mathsf{BishopCompletionTheorem}^{\uparrow}$ without displaying the $\mathsf{UniformCauchyCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{DyadicRatCore}^{\uparrow}$ rows refutes the carrier.

Independence witness: uniform-cauchy-completion, streamname, regseqrat, dyadicratcore, real, cauchysequencespace, bishop-completion-theorem: no listed sibling is bijective to $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ because the packet jointly carries the uniform-Cauchy source, finite windows, regular readback, dyadic tolerance, Real seal, CauchySequenceSpace handoff, BishopCompletionTheorem handoff, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopUniformCauchyCompletion}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$ packet is generated from a uniform-Cauchy source row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerances, Real seal rows, CauchySequenceSpace and BishopCompletionTheorem handoffs, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopUniformCauchyCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6286.1, Definition 6286.2, and Definition 6286.3 seed the finite Bishop uniform-Cauchy completion route from displayed source rows through windows, readbacks, tolerances, Real seal, CauchySequenceSpace handoff, and BishopCompletionTheorem handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopUniformCauchyCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion, selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopUniformCauchyCompletion}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, source-window readback, dyadic tolerance exactness, Real seal nonescape, CauchySequenceSpace handoff, BishopCompletionTheorem handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopUniformCauchyCompletionUp.

Chapter 6287

A Concrete Naming Certificate for $\mathsf{PointedCompleteMetric}^{\uparrow}$

Following Bishop and Bridges on complete metric spaces with a chosen origin, the $\mathsf{PointedCompleteMetric}^{\uparrow}$ packet records a metric carrier, a completion service, and a base-pointed Cauchy request route as one finite BEDC NameCert surface. It sits next to the $\mathsf{Met}^{\uparrow}$ carrier of chapter 57, the $\mathsf{CompleteMetric}^{\uparrow}$ completion row of chapter 128, the $\mathsf{Real}^{\uparrow}$ distance seal of chapter 38, the $\mathsf{RegSeqRat}^{\uparrow}$ rational schedule of chapter 301, and the $\mathsf{StreamName}^{\uparrow}$ window ledger of chapter 126. The packet names the rooted complete-metric route without importing a selected host point, quotient equality on completed points, open-cover compactness, countable choice, or an ambient real-completeness theorem.

Definition 6287.1 (PointedCompleteMetric carrier)**.** *A* $\mathsf{PointedCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (M, C, B, R, H, K, N)$$

with the following displayed rows:

1. *M is the* $\mathsf{Met}^{\uparrow}$ *carrier row, including the metric distance, classifier, and local comparison data available to rooted Cauchy consumers.*
2. *C is the* $\mathsf{CompleteMetric}^{\uparrow}$ *Cauchy-limit row, exposing the completion service only for regular Cauchy requests already displayed over M.*
3. *B is the base-point row, stored as named metric data over M rather than as a host point or as a point in a quotient completion.*
4. *R is the rooted Cauchy request row. It records the finite* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *rational tolerance reads that start at B and are sealed by the* $\mathsf{Real}^{\uparrow}$ *distance rows before the completion service C is inspected.*
5. *H records componentwise* hsame *transport for the metric carrier, completion row, base point, rooted Cauchy requests, provenance, replay, and local naming rows.*
6. *K records displayed* Cont *replay from a rooted Cauchy request through M, B, R, C, H and into the local certificate row.*
7. *N is the* Pkg *provenance and local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetric}^{\uparrow}}$ *row for exactly this displayed packet.*

The classifier compares accepted packets only by transporting these seven rows together. It has no coordinate for a selected host point, a quotient-completion representative, a choice-selected limit, open-cover compactness, or ambient equality on real-valued distances.

Lean4 (definition) `BEDC.Derived.PointedCompleteMetricUp`

Theorem 6287.2 (PointedCompleteMetric NameCert obligations)**.** *For every accepted* $\mathsf{PointedCompleteMetric}^{\uparrow}$ *carrier* $P = (M, C, B, R, H, K, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PointedCompleteMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{Met}^{\uparrow}$ *row* M*, completion-service admission for* C*, base-point admission for* B*, rooted Cauchy request admission for* R*, componentwise transport through* H*, replay through* K*, and provenance with local naming through* N*. Each rooted request must display its* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance reads, and* $\mathsf{Real}^{\uparrow}$ *distance seals before the* $\mathsf{CompleteMetric}^{\uparrow}$ *row is available.*

Proof. Project the carrier of Definition 6287.1. The rows M, C, B, R, H, K, N are the only coordinates available to a downstream NameCert consumer.

The rooted request is not a separate completion object. It begins at the base-point row B, reads the stream windows and rational tolerance schedule inside R, and reaches the completion service only through the row C displayed over the same metric carrier M.

Transport and replay preserve the route componentwise. The row H moves the metric, completion, base-point, and rooted-request data together, while K replays the same displayed path for consumers. Since N records only provenance and local naming for that packet, no unlisted host point, representative selector, open-cover compactness principle, quotient equality, or ambient real-completeness theorem can enter the certificate. □

Theorem 6287.3 (PointedCompleteMetric completion anchor)**.** *Let* $P = (M, C, B, R, H, K, N)$ *be an accepted* $\mathsf{PointedCompleteMetric}^{\uparrow}$ *carrier. Every* $\mathsf{CompleteMetric}^{\uparrow}$ *completion read exported by* P *is anchored at the same* $\mathsf{Met}^{\uparrow}$ *carrier* M *and the same base-point row* B *used by the rooted Cauchy request row* R*. Thus a completion consumer may inspect* C *only after the route*

$$M, B, R \longmapsto C \longmapsto H, K, N$$

has been displayed.

Proof. Read M, B, R from the carrier first. The base point is a row over the metric carrier, and the rooted Cauchy request row is displayed over the same source.

The completion row is downstream of that route. By the carrier definition, C is the complete-metric service for requests already displayed over M, not a detached complete object. Therefore a consumer cannot inspect C before the metric carrier and rooted base-point request are visible.

Finally apply transport and replay. The rows H and K preserve the order M, B, R, C, and the naming row N records only that finite packet. The completion anchor is therefore the displayed rooted route, not a selected host endpoint or quotient representative. □

Theorem 6287.4 (PointedCompleteMetric rooted Cauchy route)**.** *Every rooted Cauchy consumer of an accepted* $\mathsf{PointedCompleteMetric}^{\uparrow}$ *packet factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow} \longmapsto M, B, R, C, H, K, N.$$

The route supplies the base-pointed completion handoff needed by downstream Cauchy and completion chapters, while refusing selected host points, selected subsequences, quotient real equality, choice, open-cover compactness, and an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$*.*

Proof. Start with the rooted request row. It records the stream windows and regular rational tolerance reads over the displayed metric source and base point, so a consumer receives only finite observations before completion is considered.

Next use the real seal and completion row. The distance observations are sealed as $\mathsf{Real}^{\uparrow}$ data inside the same route, and the $\mathsf{CompleteMetric}^{\uparrow}$ row is available only after those observations have been displayed.

Transport, replay, provenance, and local naming do not add analytic content. They preserve and name the finite route through H, K, N. Hence every rooted Cauchy read factors through the displayed packet and cannot escape to selected host points, selected subsequences, quotient equality, choice, open-cover compactness, or ambient real completeness. □

Falsifiable prediction: A claimed $\mathsf{PointedCompleteMetric}^{\uparrow}$ consumer that reads a completed endpoint before displaying the metric carrier, base point, rooted Cauchy request, stream windows, regular rational tolerance reads, real distance seals, and complete-metric row refutes this carrier.

Independence witness: metricspace, completemetric, real, regseqrat, streamname: $\mathsf{PointedCompleteMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric carrier, completion row, base point, rooted Cauchy request, transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{PointedCompleteMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{PointedCompleteMetric}^{\uparrow}$ packet is generated from a MetricSpaceUp carrier row, a CompleteMetricUp Cauchy-limit row, a base-point row, a rooted Cauchy request row through StreamNameUp, RegSeqRatUp, and RealUp reads, componentwise hsame transport, displayed Cont replay, provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6287.1, Definition 6287.2, Definition 6287.3, and Definition 6287.4 seed the rooted complete-metric carrier and its Cauchy-to-completion handoff over the displayed finite rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PointedCompleteMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean theorem verification, bridge checking, quotient completion equality, selected host endpoints, selected subsequences, countable choice, open-cover compactness, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, rooted Cauchy request admission, completion-anchor exactness, transport, replay, provenance, local naming, and downstream completion-consumer alignment for BEDC.Derived.PointedCompleteMetricUp.

Chapter 6288

A Concrete Naming Certificate for $\mathsf{CauchyContinuousFunction}^{\uparrow}$

Following Bishop and Bridges, constructive analysis treats a continuous function on real names through a displayed rule on regular Cauchy representatives together with the modulus data that preserves regularity. The $\mathsf{CauchyContinuousFunction}^{\uparrow}$ packet records that surface as finite BEDC data. It consumes the dyadic tolerance arithmetic of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, the graph-and-modulus rows of chapter 130, and the tolerance-only handoff of chapter 369. The chapter names the map object used by Real and Completion consumers without adding host function equality, quotient representatives, selected limits, or ambient completeness to the carrier.

Definition 6288.1 (CauchyContinuousFunction carrier)**.** *A* $\mathsf{CauchyContinuousFunction}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (S, R, D, M, U, J, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window source row for regular Cauchy real-name observations.*
2. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those source windows, including the regularity budget used by downstream image requests.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger for source and target precision requests.*
4. *M is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph-and-local-modulus row; it records the displayed map rule only through finite source observations and target requests.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *handoff row that turns each target tolerance into a displayed source tolerance when uniform control is supplied.*
6. *J is the image-name production row, read only after S, R, D, M, U have fixed the finite source and tolerance route.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the produced regular Cauchy real name.*
8. *H records componentwise* hsame *transport for S, R, D, M, U, J, E and the structural rows.*
9. *C records displayed* Cont *replay forcing every consumer to read the packet in the order $S, R, D, M, U, J, E, H, P, N$*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyContinuousFunction}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyContinuousFunction}^{\uparrow}}$ *row.*

The carrier classifies functions only by these displayed rows. It has no coordinate for host function extensionality, quotient equality of real names, a selected representative sequence, countable choice, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Theorem 6288.2 (CauchyContinuousFunction NameCert obligations)**.** *For every accepted* CauchyContinuousFunct *carrier*

$$F = (S, R, D, M, U, J, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyContinuousFunction}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the StreamName source windows, RegSeqRat readback, DyadicRatCore tolerance ledger, ContinuousMap graph row, UniformModulus handoff, image-name row, Real seal, transport, replay, provenance, and local naming rows;*
2. *source-name exactness, requiring every map request to read the source only through S and R before any image-name row is available;*
3. *tolerance compatibility, requiring D to be the shared dyadic ledger consumed by the source readback, the continuous-map row, the uniform-modulus handoff, and the image-name production row;*
4. *continuity control, requiring M and U to answer each target precision request by a displayed source precision request before J may be read;*
5. *regular-image sealing, requiring J and E to expose only the produced regular Cauchy real name downstream of S, R, D, M, U;*
6. *componentwise transport, displayed replay, provenance, local naming, and nonescape for the same displayed route through H, C, P, N.*

No CauchyContinuousFunction obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6288.1. The accepted packet displays exactly the source windows S, regular readback R, dyadic ledger D, continuous-map row M, uniform-modulus handoff U, image-name row J, Real seal E, and structural rows H, C, P, N.

Source-name exactness reads the first two coordinates. A consumer opens the finite StreamName observations and then the RegSeqRat readback attached to those observations; the image-name side has no available row before this source route is fixed.

Tolerance compatibility and continuity control are the middle of the route. The DyadicRatCore ledger D supplies the common precision arithmetic, while M and U convert each requested target tolerance into a displayed source tolerance before the image-name row is admitted.

Regular-image sealing is terminal. The row J records only the produced regular Cauchy name generated from S, R, D, M, U, and E seals that name as a Real-facing datum only after those rows have been displayed.

Finally, H, C, P, N transport, replay, package, and name the same finite route. Since host function equality, quotient real-name equality, selected representatives, countable choice, ambient Real completeness, Lean verification, and bridge checking are absent from the carrier, no local obligation can export them. □

Falsifiable prediction: Every $\mathsf{CauchyContinuousFunction}^{\uparrow}$ read factors through one StreamName source-window row, one RegSeqRat readback row, one DyadicRatCore tolerance ledger, one ContinuousMap graph-and-local-modulus row, one UniformModulus handoff row, one image-name row,

one terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: streamname, regseqrat, dyadicratcore, real, continuousmap, uniformmodulus: $\mathsf{CauchyContinuousFunction}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the regular Cauchy source route, the displayed map row, the uniform precision handoff, and the produced real-name seal as one map object.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyContinuousFunction}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyContinuousFunction}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6288.1 and Definition 6288.2 seed the finite map object over regular Cauchy real names through StreamName windows, RegSeqRat readback, DyadicRatCore tolerance arithmetic, ContinuousMap graph data, UniformModulus handoff, image-name production, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyContinuousFunction}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host function equality, quotient equality of real names, selected representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source-name exactness, tolerance compatibility, continuity control, regular-image sealing, transport, replay, provenance, and nonescape for BEDC.Derived.CauchyContinuousFunctionUp.

Constructive story: A $\mathsf{CauchyContinuousFunction}^{\uparrow}$ packet is generated from StreamName source windows, RegSeqRat readback, a DyadicRatCore tolerance ledger, a ContinuousMap graph row, a UniformModulus handoff, an image-name row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6289

A Concrete Naming Certificate for $\mathsf{RegularCauchyDensity}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers, the $\mathsf{RegularCauchyDensity}^{\uparrow}$ packet records the finite density route by which a dyadic rational approximation is read through regular Cauchy data before a terminal $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits over the dyadic arithmetic of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the separable Real boundary of chapter 1809. The packet names a displayed approximation handoff for Real and Completion consumers; it is not a quotient equality of real names, a selected Cauchy representative, countable choice, trichotomy, a host density theorem, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6289.1 (RegularCauchyDensity carrier)**.** *A* $\mathsf{RegularCauchyDensity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (Q, S, R, A, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row naming the dyadic rational candidate and the dyadic tolerance budget requested by the density read.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected prefixes of the source real name at the precisions demanded by Q.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row reached from the finite windows.*
4. *A is the approximation row stating that Q is accepted only after the regular-rational comparison in R displays the requested dyadic tolerance.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after Q, S, R, A have been displayed.*
6. *H stores componentwise* hsame *transport for the dyadic candidate, stream windows, regular readback, approximation row, Real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a density request through Q, S, R, A, E, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDensity}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDensity}^{\uparrow}}$ *row.*

The carrier stores only displayed finite rows. It has no coordinate for quotient equality of real names, selected representatives, countable choice, trichotomy, host density of rationals, or ambient completeness.

Theorem 6289.2 (RegularCauchyDensity NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDensity}^{\uparrow}$ *carrier*

$$D = (Q, S, R, A, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyDensity}^{\uparrow}}$ *surface consists exactly of carrier admission for the dyadic candidate, finite stream windows, regular-rational readback, approximation row, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming. The obligation surface also requires dyadic-budget compatibility between* Q *and* A*, window-readback exactness between* S *and* R*, seal nonescape through* E*, and absence of quotient real equality, selected representatives, countable choice, trichotomy, host density, and ambient completeness.*

Proof. Project the carrier of Definition 6289.1. The available analytic rows are Q, S, R, A, E, and the available structural rows are H, C, P, N.

The budget compatibility row reads Q before A. Thus an accepted density request cannot claim a dyadic approximation unless the requested dyadic tolerance is one of the displayed rows.

The readback part then runs through S and R. Finite stream windows expose the inspected prefixes, and the regular-rational row records the comparison data consumed by the approximation row.

Finally E seals the Real-facing output only after those rows have been displayed. The transport, replay, provenance, and naming rows preserve the same finite route, while the refused host objects are absent from the carrier inventory. Hence the listed rows exhaust the local NameCert obligations. □

Theorem 6289.3 (RegularCauchyDensity dyadic approximation)**.** *Every Real-facing density read exported by an accepted* $\mathsf{RegularCauchyDensity}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}$$

as instantiated by Q, S, R, A, E*. In particular, the dyadic candidate and tolerance budget are read first, the finite stream windows and regular-rational comparison are displayed next, and the terminal Real seal is available only after the approximation row has consumed those data.*

Proof. Start with an exported density read and project the accepted carrier. The only dyadic candidate row is Q, so the read begins at the displayed $\mathsf{DyadicRatCore}^{\uparrow}$ surface.

The finite observation route then passes through S and R. The stream row opens the inspected windows, and the regular-rational row supplies the comparison surface needed to judge the dyadic budget.

The approximation row A is downstream of these rows and upstream of the terminal seal E. Since the carrier has no direct edge from Q to E, a Real-facing consumer can receive the seal only through the displayed dyadic-window-readback route.

The rows H, C, P, N do not add another approximation source. They transport, replay, source, and name the same finite route, so every exported density read factors as claimed. □

Theorem 6289.4 (RegularCauchyDensity RegSeqRat handoff)**.** *An accepted* $\mathsf{RegularCauchyDensity}^{\uparrow}$ *carrier hands a dyadic approximation to the* $\mathsf{Real}^{\uparrow}$ *boundary only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* R*. No Real-facing consumer of the packet receives a quotient representative, a selected Cauchy stream, a countable-choice schedule, trichotomy, a host rational-density theorem, or an ambient Real-completeness principle.*

Proof. Apply the dyadic approximation theorem. It forces every Real-facing density read to pass through Q, S, R, A, E in that order, with R as the unique regular-rational readback coordinate.

It remains to inspect what the handoff can export. The carrier's structural rows H, C, P, N only preserve the same displayed rows under transport, replay, provenance, and local naming.

The refused objects are not coordinates of Definition 6289.1. Therefore the handoff exports a finite RegSeqRat-mediated approximation route and none of the listed host or quotient principles. □

Lean4 ✓ `BEDC.Derived.RegularCauchyDensityRegSeqRatHandoff`

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDensity}^{\uparrow}$ density read factors through one DyadicRatCoreUp candidate-and-budget row, one StreamNameUp finite-window row, one RegSeqRatUp regular-rational readback row, one approximation row, one RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicratcore, streamname, regseqrat, real, real-separability: $\mathsf{RegularCauchyDensity}^{\uparrow}$ is not bijective to any listed sibling because it packages the dyadic approximation request, finite window opening, regular-rational readback, terminal Real seal, transport, replay, provenance, and local NameCert as one density certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDensity}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDensity}^{\uparrow}$ packet is generated from a DyadicRatCoreUp candidate-and-budget row, StreamNameUp finite windows, RegSeqRatUp regular-rational readback, an approximation row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyDensity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6289.1, Definition 6289.2, Definition 6289.3, and Definition 6289.4 seed the finite dyadic-density route from displayed dyadic budgets through regular-rational readback to the Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyDensity}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyDensityUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality of real names, selected Cauchy representatives, countable choice, trichotomy, host rational-density theorems, ambient Real completeness, Lean theorem checking, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic-budget compatibility, window-readback exactness, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, local naming, and nonescape for BEDC.Derived.RegularCauchyDensityUp.

Chapter 6290

A Concrete Naming Certificate for $\mathsf{ChoquetCompleteSpace}^{\uparrow}$

Following Choquet and Oxtoby, the $\mathsf{ChoquetCompleteSpace}^{\uparrow}$ packet records the complete-space side of the Choquet and Baire route as finite BEDC NameCert rows. It sits beside the game interface of chapter 4077, the complete-metric route of chapter 128, the Baire surface of chapter 3535, the Baire-space prefix of chapter 334, the finite schedule discipline of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The chapter names a constructive complete-space object whose evidence is a nested-open play ledger, not a selected generic point, countable-choice sequence, quotient completion, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6290.1 (Choquet-complete-space carrier)**.** *A* $\mathsf{ChoquetCompleteSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (G, M, L, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{ChoquetGame}^{\uparrow}$ *row naming the legal nested-open game requests and the finite play prefixes visible to the packet.*
2. *M is the complete metric and Baire route row, read through* $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CompleteMetricBaireSurface}^{\uparrow}$, *and* $\mathsf{BaireSpace}^{\uparrow}$ *data before any terminal seal is available.*
3. *L is the nested-open play ledger, recording each inspected open request, successor open row, metric-ball refinement, inclusion check, player tag, and finite stability witness.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selecting the visible play prefixes, open rows, centers, radii, and requested tolerances.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for rational centers, radius budgets, and tolerance comparisons used by the nested-open route.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal reached only after the Choquet game row, complete metric and Baire route, nested-open ledger, stream windows, and regular rational readback have been displayed.*
7. *H records componentwise* hsame *transport for Choquet-game rows, complete-metric and Baire route rows, nested-open play ledgers, finite windows, regular readbacks, Real seals, replay data, provenance, and local naming.*
8. *C records displayed* Cont *replay from a complete-space request to the same game, metric, Baire, ledger, window, readback, seal, transport, provenance, and naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{ChoquetCompleteSpace}^{\uparrow}$, $\mathsf{ChoquetGame}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, CompleteMetricBaireSu BaireSpace$^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{ChoquetCompleteSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a selected generic point, countable choice, quotient completion, global winning strategy, host intersection point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness principle.*

Lean4 ✓ `BEDC.Derived.ChoquetCompleteSpaceUp.ChoquetCompleteSpaceTasteGate single carrier alignment`

Theorem 6290.2 (Choquet-complete-space NameCert obligations)**.** *For every accepted* $\mathsf{ChoquetCompleteSpace}^{\uparrow}$ *carrier* $Q = (G, M, L, S, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ChoquetCompleteSpace}^{\uparrow}}$ *surface consists exactly of carrier admission, Choquet-game legality, complete-metric and Baire route exposure, nested-open ledger exactness, finite stream-window readback, regular rational tolerance readback, Real-seal ordering, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $G, M, L, S, R, E, H, C, P, N$.

Proof. Project the accepted carrier of Definition 6290.1. The analytic rows are G, M, L, S, R, E, and the structural rows are H, C, P, N.

The game and completeness content is finite. The row G supplies the legal Choquet-game requests, while M supplies only the displayed complete-metric and Baire route data needed by those requests.

The nested-open content is stored in L and inspected through S and R. Thus each open request, successor row, metric-ball refinement, inclusion check, center, radius, and tolerance read is visible before any Real-facing seal is read.

Finally E, H, C, P, N seal, transport, replay, source, and name the same route. Since the carrier has no coordinate for the excluded objects, the listed rows exhaust the local NameCert surface. □

Theorem 6290.3 (Choquet-complete-space nested-play stability)**.** *Let* $Q = (G, M, L, S, R, E, H, C, P, N)$ *be an accepted* $\mathsf{ChoquetCompleteSpace}^{\uparrow}$ *carrier. Every finite nested-open play prefix recorded by* L *remains stable under the displayed* hsame *transport and* Cont *replay exactly when its Choquet-game row, complete-metric and Baire route row, stream windows, regular rational readback, and Real-seal handoff remain on the route*

$$G \longmapsto M \longmapsto L \longmapsto S \longmapsto R \longmapsto E.$$

The stability claim exports no selected generic point, countable choice, quotient completion, global winning strategy, host intersection point, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the finite play prefix displayed in L. Its predecessor and successor open rows, metric-ball refinements, inclusion checks, and player tags are carrier rows rather than predicates over a host topology.

Read the surrounding route through G and M. The Choquet-game row supplies legality of the finite play prefix, and the complete-metric and Baire row supplies the displayed ball and dense-open data used by that prefix.

The observable numerical data passes through S and R. The stream windows choose the finite centers, radii, open rows, and tolerances, while the regular rational row reads those values without adding a selected limit point.

Transport and replay stability are exactly the rows H and C, followed by provenance and local naming in P and N. The Real seal E is downstream from the same route, so a consumer cannot replace the packet by a choice sequence, quotient completion, host intersection, global strategy, or ambient completeness theorem. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoquetCompleteSpace}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoquetCompleteSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6290.1, Definition 6290.2, and Definition 6290.3 seed the Choquet-complete-space certificate over game rows, complete-metric and Baire route rows, nested-open ledgers, finite windows, rational readback, real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoquetCompleteSpace}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected generic points, countable choice, quotient completion, global winning strategies, host intersection points, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Choquet-game legality, complete-metric and Baire route exposure, nested-open ledger exactness, finite-window readback, rational tolerance readback, real-seal ordering, transport, replay, provenance, and local naming for BEDC.Derived.ChoquetCompleteSpaceUp.

Constructive story: A $\mathsf{ChoquetCompleteSpace}^{\uparrow}$ packet is generated from ChoquetGame rows, complete-metric and Baire route rows, nested-open play ledgers, StreamName finite windows, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6291

A Concrete Naming Certificate for $\mathsf{SubstrateFlowInvariant}^{\uparrow}$

The $\mathsf{SubstrateFlowInvariant}^{\uparrow}$ packet realizes the cellular substrate reading of chapter 9390 as finite orbit-window invariant data. It sits beside the trust-substrate row of chapter 940, the orbit substrate of chapter 938, the classifier boundary of chapter 3810, the trace invariant of chapter 2360, and the local Rule 110 observation rows of chapter 6004. The chapter names a finite flow-invariant surface for cellular substrate evolution without importing an infinite orbit, internal induction over time, or a semantic conservation theorem outside the packet.

Definition 6291.1 (Substrate-flow-invariant carrier)**.** *A* $\mathsf{SubstrateFlowInvariant}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$I = (U, O, F, A, W, H, C, P, N)$$

with the following displayed rows:

1. *U is the cellular trust-substrate row naming the local update rule, manifest row, accepted alphabet, and substrate audit source.*
2. *O is the finite orbit-window row naming the inspected time interval, boundary convention, local stencils, and observed successor rows.*
3. *F is the flow-invariant ledger, recording the invariant tag, the row-window values before and after each displayed update, and the finite equality or monotone comparison witnessed at that window.*
4. *A is the audit-route row linking the local stencil observations, pattern-classifier rows, trace-invariant rows, and classifier-boundary rows used by F.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row selecting the finite orbit prefixes, stencil windows, classifier tags, and invariant checks visible to a consumer.*
6. *H records componentwise* hsame *transport for trust-substrate rows, orbit windows, invariant ledgers, audit routes, stream windows, replay data, provenance, and local naming.*
7. *C records displayed* Cont *replay from a substrate-flow request to the same trust-substrate, orbit-window, invariant-ledger, audit-route, stream-window, transport, provenance, and naming rows.*
8. *P is the* Pkg *provenance over* $\mathsf{SubstrateFlowInvariant}^{\uparrow}$, $\mathsf{CellularTrustSubstrate}^{\uparrow}$, $\mathsf{CellularOrbitSubstrate}^{\uparrow}$, $\mathsf{CellularSubstrateClassifierBoundary}^{\uparrow}$, $\mathsf{SubstrateTraceInvariant}^{\uparrow}$, $\mathsf{RuleOneTenLocalStencilObservation}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{SubstrateFlowInvariant}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the finite local-rule, orbit-window, invariant-ledger, audit-route, stream-window, transport, replay, provenance, and naming rows. It has no coordinate for infinite orbits, internal induction over all time, semantic conservation laws, hidden classifier repair, or external cellular substrate trust.

Theorem 6291.2 (Substrate-flow-invariant NameCert obligations)**.** *For every accepted* $\mathsf{SubstrateFlowInvariant}^{\uparrow}$ *carrier* $I = (U, O, F, A, W, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{SubstrateFlowInvariant}^{\uparrow}}$ *surface consists exactly of carrier admission, local-rule trust, finite orbit-window exposure, invariant-ledger exactness, audit-route coverage, stream-window readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside* $U, O, F, A, W, H, C, P, N$*.*

Proof. Project the carrier of Definition 6291.1. The substrate row U, orbit row O, invariant ledger F, audit route A, and stream window row W are the only data rows available before transport and replay.

The invariant check is finite. Each comparison in F is tied to an observed predecessor and successor window from O, and the relevant local stencil and classifier evidence is named by A.

The consumer sees those windows through W. Transport, replay, provenance, and local naming are then exactly the rows H, C, P, N, so the packet cannot substitute an infinite orbit, internal induction principle, semantic conservation theorem, hidden classifier repair, or external trust source for the displayed rows. □

Falsifiable prediction: A $\mathsf{SubstrateFlowInvariant}^{\uparrow}$ implementation should fail admission when a finite Rule 110 orbit-window update changes the displayed invariant tag while preserving the same trust-substrate and audit-route rows.

Independence witness: The packet is independent of any single cellular substrate sibling: it can cite chapter 940, chapter 938, chapter 2360, chapter 3810, and chapter 6004 while requiring only their finite row surfaces.

Theorem 6291.3 (Substrate-flow-invariant Lean interface)**.** *The Lean-facing surface for* $\mathsf{SubstrateFlowInvariant}^{\uparrow}$ *is expected to expose a nontrivial TasteGate over finite orbit-window invariant rows and FieldFaithful instances for the local-rule row, orbit-window row, invariant-ledger row, audit-route row, stream-window row, transport row, replay row, provenance row, and local NameCert row. The interface does not require an infinite orbit type or an induction principle over all time.*

Proof. Read the nine carrier rows of Definition 6291.1. A nontrivial TasteGate can distinguish at least one accepted finite invariant ledger from a ledger whose displayed before-and-after row values disagree on the same orbit window.

FieldFaithful evidence is rowwise. Each field named in the carrier has a finite projection whose equality is checked by the packet classifier, and no projection ranges over an infinite orbit or a semantic conservation predicate.

Thus the Lean interface is the row surface itself: nontriviality comes from finite disagreement of invariant ledgers, and faithfulness comes from the visible fields $U, O, F, A, W, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{SubstrateFlowInvariant}^{\uparrow}}$**.**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SubstrateFlowInvariant}^{\uparrow}}, \mathsf{seed})$
Scope closed: Definition 6291.1 and Definition 6291.2 seed the substrate-flow-invariant certificate over local-rule trust, finite orbit windows, invariant ledgers, audit routes, stream windows, transport, replay, provenance, and local naming.
Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SubstrateFlowInvariant}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim infinite orbit closure, internal induction over all time, semantic conservation outside finite rows, hidden classifier repair, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local-rule trust, finite orbit-window exposure, invariant-ledger exactness, audit-route coverage, stream-window readback, transport, replay, provenance, nontrivial Taste-Gate evidence, and FieldFaithful row instances for BEDC.Derived.SubstrateFlowInvariantUp.

Constructive story: A SubstrateFlowInvariant$^{\uparrow}$ packet is generated from cellular trust-substrate rows, finite orbit windows, invariant ledgers, audit routes, StreamName windows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6292

A Concrete Naming Certificate for $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$ packet records the finite-family form of equicontinuity used for located real intervals and regular Cauchy real-name maps. It sits after the Cauchy-continuous map object of chapter 6288, the located interval uniform-Cauchy route of chapter 6256, the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance arithmetic of chapter 291, and the terminal real-name seal of chapter 38. The packet names the textbook finite-means equicontinuity handoff without importing host function equality, open-cover compactness, selected limits, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6292.1 (Bishop equicontinuity principle carrier)**.** *A* $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, I, M, U, W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *F is the finite family row, listing the accepted family indices and the familywise consumer route without exposing a host function space.*
2. *I is the located interval source row, read through the displayed interval and located endpoint data used by the family.*
3. *M is the* $\mathsf{CauchyContinuousFunction}^{\uparrow}$ *row for each family member, recording the source-window, tolerance, graph, image-name, and real-seal route of the member.*
4. *U is the uniform-Cauchy handoff row supplied over the same located interval source; it records the common finite tolerance route consumed by the family.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for source and image observations.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same finite windows.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, recording the dyadic comparison budget used by the family and the common handoff.*
8. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after F, I, M, U, W, R, D have fixed the finite family route.*

9. *H records componentwise* hsame *transport for the family, interval, map, uniform-Cauchy, window, readback, tolerance, seal, replay, provenance, and local naming rows.*

10. *C records displayed* Cont *replay through the finite route*

$$F, I, M, U \longmapsto W, R, D, S, H, P, N.$$

11. *P is the* Pkg *provenance over* BishopEquicontinuityPrinciple$^{\uparrow}$, CauchyContinuousFunction$^{\uparrow}$, LocatedIntervalUniform StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.

12. *N is the local* NameCert$_{\text{BishopEquicontinuityPrinciple}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for host equality of functions, open-cover compactness, selected pointwise limits, quotient equality of real names, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6292.2 (Bishop equicontinuity principle NameCert obligations)**.** *For every accepted* BishopEquicontinuityPrinciple$^{\uparrow}$ *carrier*

$$E = (F, I, M, U, W, R, D, S, H, C, P, N),$$

the local NameCert$_{\text{BishopEquicontinuityPrinciple}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, finite family exposure through F, located interval exposure through I, CauchyContinuousFunction exposure through M, uniform-Cauchy handoff exposure through U, StreamName window exposure through W, RegSeqRat readback through R, DyadicRatCore tolerance accounting through D, Real sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6292.1. The analytic rows are the finite family F, the located interval source I, the Cauchy-continuous member rows M, the uniform-Cauchy handoff U, and the window, readback, tolerance, and seal rows W, R, D, S.

The family and interval obligations are finite. The row F fixes the accepted family indices, while I fixes the located interval source on which every member row is read.

The map and handoff obligations are then read from M and U. Each member exposes only the CauchyContinuousFunction route, and the common handoff supplies the tolerance route that a downstream equicontinuity consumer may share.

Finally H, C, P, N transport, replay, source, and name exactly the displayed rows. Since the carrier has no coordinate for host function equality, open-cover compactness, selected limits, quotient real-name equality, choice, or ambient completeness, those objects cannot appear as local obligations. □

Theorem 6292.3 (Bishop equicontinuity principle uniform-Cauchy handoff)**.** *Every equicontinuity-facing consumer of an accepted* BishopEquicontinuityPrinciple$^{\uparrow}$ *carrier receives the finite handoff*

CauchyContinuousFunction$^{\uparrow}$, LocatedIntervalUniformCauchy$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$ $\vdash$

The consumer may read the finite family row, located interval source, CauchyContinuousFunction rows, common uniform-Cauchy handoff, finite windows, regular rational readback, dyadic tolerance budget, terminal Real seal, transport, replay, provenance, and local NameCert. It cannot bypass those rows by comparing host functions, selecting pointwise limits, importing open-cover compactness, or appealing to ambient completeness.

Proof. Apply Definition 6292.2. Any accepted consumer must open F, I, M, U, W, R, D, S before the structural rows H, C, P, N are available.

The common equicontinuity route is controlled by U over the same located interval row I. The finite family row F determines which member rows in M are being read, while W, R, D, S expose only finite windows, rational readback, dyadic tolerance, and the terminal real-name seal.

The displayed Cont replay preserves that order. Hence a downstream uniform-Cauchy or compact-interval consumer receives a finite BEDC handoff rather than an external function-space principle, selected limit argument, open-cover compactness theorem, quotient comparison, or Real completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$ equicontinuity read factors through one finite family row, one located interval source row, CauchyContinuousFunction member rows, one uniform-Cauchy handoff row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchy-continuous-function, located-interval-uniform-cauchy, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$ is not bijective with any listed sibling because it jointly carries a finite family route, located interval source, Cauchy-continuous member rows, a common uniform-Cauchy handoff, finite windows, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}$ packet is generated from a finite family row, located interval source, CauchyContinuousFunction member rows, a uniform-Cauchy handoff, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6292.1, Definition 6292.2, and Definition 6292.3 seed the finite family equicontinuity route over located intervals, Cauchy-continuous map rows, uniform-Cauchy handoff, window readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopEquicontinuityPrinciple}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host function equality, open-cover compactness, selected pointwise limits, quotient equality of real names, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite family exposure, located interval exposure, CauchyContinuousFunction member exposure, uniform-Cauchy handoff exactness, finite-window replay, RegSeqRat readback, dyadic tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopEquicontinuityPrincipleUp.

Chapter 6293

A Concrete Naming Certificate for CompactUniformOscillationWindow$^{\uparrow}$

Following Bishop and Bridges constructive analysis, the CompactUniformOscillationWindow$^{\uparrow}$ packet records the finite-net and finite oscillation-window route used by compact uniform continuity. It is the concrete landing for the vision of chapter 9237, sitting between the compact oscillation packet of chapter 3737 and the finite oscillation partition packet of chapter 3800. It also reads compact metric source data from chapter 129, continuous graph data from chapter 130, uniform-modulus data from chapter 369, finite windows from chapter 126, regular rational readback from chapter 301, dyadic tolerance from chapter 291, and terminal real-name sealing from chapter 38. The packet does not claim open-cover compactness, selected extrema, quotient function equality, unbounded partition search, or ambient completeness.

Definition 6293.1 (Compact uniform oscillation window carrier)**.** *A* CompactUniformOscillationWindow$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (K, F, U, O, M, B, W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net source row, read through the accepted* CompactMetric$^{\uparrow}$ *finite-net surface.*
2. *F is the continuous graph row, exposing the displayed* ContinuousMap$^{\uparrow}$ *observations consumed by the compact-uniform route.*
3. *U is the* UniformModulus$^{\uparrow}$ *ledger, recording tolerance requests and source-radius responses.*
4. *O is the* CompactUniformOscillation$^{\uparrow}$ *oscillation-bound row over the same compact source and graph rows.*
5. *M is the finite mesh row, listing the compact finite-net cells or interval cells used by the oscillation window.*
6. *B is the finite oscillation-window row supplied by* FiniteOscillationPartition$^{\uparrow}$*, recording the cell-wise bound and refinement readback.*
7. *W is the* StreamName$^{\uparrow}$ *finite-window row for source and target observations.*
8. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to those windows.*

9. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row shared by U, O, M, B, W, R.*
10. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the compact finite-net, graph, modulus, oscillation, mesh, window, readback, and tolerance rows have been displayed.*
11. *H records componentwise* hsame *transport for all analytic and structural rows.*
12. *C records displayed* Cont *replay through*

$$K, F, U, O, M, B, W, R, D, S \longmapsto H, P, N.$$

13. *P is the* Pkg *provenance over* $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactUniformOscillation}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationWindow}^{\uparrow}}$ *row.*

The carrier compares accepted packets only through these rows. It has no coordinate for open-cover compactness, selected extrema, quotient equality of maps, unbounded partition search, choice-selected refinements, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6293.2 (Compact uniform oscillation window NameCert obligations)**.** *For every accepted* $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$ *carrier*

$$Q = (K, F, U, O, M, B, W, R, D, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationWindow}^{\uparrow}}$ *surface consists exactly of compact finite-net exposure through K, continuous graph exposure through F, uniform-modulus accounting through U, compact oscillation-bound exposure through O, finite mesh exposure through M, finite oscillation-window exposure through B, StreamName window exposure through W, RegSeqRat readback through R, DyadicRatCore tolerance accounting through D, Real sealing through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6293.1. The compact-uniform rows are K, F, U, O, and the finite oscillation-window rows are M, B, W, R, D, S.

The compact source is finite at the point of consumption. The row K supplies finite-net data, F supplies the displayed continuous graph, and U records the tolerance-only modulus ledger used by the compact-uniform read.

The oscillation window is then read through O, M, B, W, R, D, S. The compact oscillation-bound row and finite mesh row determine which cellwise windows are visible, while the stream, rational, dyadic, and real rows provide the finite readback and terminal seal.

Finally H, C, P, N transport, replay, source, and name the same packet. Since none of the excluded compactness, selection, quotient, unbounded-search, or completeness principles is a carrier coordinate, no NameCert obligation can project one. □

Theorem 6293.3 (Compact uniform oscillation window modulus handoff)**.** *Every compact-uniform-continuity or integration-facing consumer of an accepted* $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$ *packet receives a finite oscillation modulus through*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactUniformOscillation}^{\uparrow}$, $\mathsf{FiniteOscillationPartition}^{\uparrow}$, Strea

The handoff reads compact finite nets, continuous graph observations, the tolerance-only modulus ledger, compact oscillation bounds, finite mesh cells, cellwise oscillation windows, finite stream windows, regular rational readback, dyadic tolerance, and terminal Real sealing before any structural replay is available.

Proof. Apply Definition 6293.2. A consumer must expose $K, F, U, O, M, B, W, R, D, S$ before it can use the transport, replay, provenance, or local naming rows.

The modulus handoff is finite because K and M fix the finite source and cell rows. The tolerance request is answered through U and D, while O and B record the compact oscillation and cellwise window bounds.

The readback remains certificate-local. The rows W, R, S expose finite observations, regular rational readback, and the terminal real-name seal, and H, C, P, N preserve that same route.

Thus the packet exports a finite oscillation modulus for compact-uniform readers. It does not export open-cover compactness, selected extrema, quotient function equality, unbounded partition search, choice-selected refinements, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$ reader that claims a compact finite oscillation modulus while omitting one of the compact finite-net, continuous graph, uniform-modulus, compact oscillation-bound, finite mesh, cellwise window, StreamName, RegSeqRat, DyadicRatCore, Real seal, hsame transport, Cont replay, Pkg provenance, or local naming rows refutes the packet.

Independence witness: compact-uniform-oscillation, finite-oscillation-partition, compactmetric, continuousmap, uniformmodulus, streamname, regseqrat, dyadicratcore, real: $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$ is not bijective with any listed sibling because it joins compact finite-net source data, continuous graph rows, uniform-modulus accounting, compact oscillation bounds, finite oscillation partitions, finite readback, Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactUniformOscillationWindow}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactUniformOscillationWindow}^{\uparrow}$ packet is generated from compact finite-net source data, ContinuousMap graph rows, a UniformModulus ledger, a CompactUniformOscillation bound row, a finite mesh row, a FiniteOscillationPartition window row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillationWindow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6293.1, Definition 6293.2, and Definition 6293.3 seed the compact finite-net and finite oscillation-window route for compact-uniform-continuity and integration-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactUniformOscillationWindow}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected extrema, quotient function equality, unbounded partition search, choice-selected refinements, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, continuous graph exposure, uniform-modulus accounting, compact oscillation-bound exposure, finite mesh exactness, finite oscillation-window accounting, finite readback, tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactUniformOscillationWindowUp.

Chapter 6294

A Concrete Naming Certificate for $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$

Following Bishop–Bridges finite-cover compactness, the $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$ packet names the constructive selector sitting between displayed compact-cover data and a $\mathsf{UniformModulus}^{\uparrow}$ export. It reads the compact metric source of chapter 129, the displayed map rows of chapter 130, the output interface of chapter 369, and the positive finite-radius ledger of chapter 452. The packet records a finite-cover route to a modulus, not a selected subcover, host minimum, ambient compactness theorem, or hidden completeness principle.

Definition 6294.1 (Finite-cover modulus selector carrier)**.** *A* $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (K, F, R, L, U, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-cover row, exposing the compact metric source, cover cells, and finite coverage ledger consumed by the selector.*
2. *F is the pointwise modulus row, assigning each displayed cover cell the local continuity data read from the* $\mathsf{ContinuousMap}^{\uparrow}$ *surface.*
3. *R is the rational lower-bound row, recording the positive comparison data obtained from the finite cover and pointwise radii.*
4. *L is the selector ledger, pairing the finite cover row with the pointwise modulus row and lower-bound row before any output is visible.*
5. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *output row produced by the displayed finite-cover route.*
6. *H records componentwise* hsame *transport for the compact cover, pointwise modulus, rational lower bound, selector ledger, output, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a target-tolerance request through K, F, R, L, U, H, P, N.*
8. *P is the* Pkg *provenance over* $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{FiniteLebesgueNumber}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverModulusSelector}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, R, L, U, H, C, P, N$. It has no coordinate for a selected subcover, selected Lebesgue number, choice schedule, host minimum, sequence extraction, or ambient completeness.

Theorem 6294.2 (Finite-cover modulus selector NameCert obligations)**.** *For an accepted* FiniteCoverModulusSelec *carrier* $S = (K, F, R, L, U, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{FiniteCoverModulusSelector}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the compact finite-cover row* K*, pointwise modulus row* F*, rational lower-bound row* R*, selector ledger* L*, output row* U*, and structural rows* H, C, P, N*;*
2. *source compatibility, requiring* K *and* F *to read the same compact metric source before the selector ledger is formed;*
3. *finite-cover accountability, requiring every output read to pass through the displayed finite cover in* K *and the pointwise modulus data in* F*;*
4. *rational lower-bound exactness, requiring* R *to be the positive finite-radius comparison row consumed by* L*;*
5. *output non-escape, requiring* U *to export only the displayed* $\mathsf{UniformModulus}^{\uparrow}$ *row and no selected subcover, host minimum, selected Lebesgue number, choice schedule, or ambient completeness.*

Proof. Project the carrier of Definition 6294.1. The rows $K, F, R, L, U, H, C, P, N$ are the only rows available to an accepted selector read.

The first two obligation rows bind carrier admission and source compatibility. The compact finite-cover row K and pointwise modulus row F must therefore share the same compact metric source before L can be read.

The next two rows force the finite-cover computation. The selector ledger L consumes the displayed finite cover, pointwise radii, and rational lower-bound row R rather than a host minimum or unrecorded cover choice.

Finally the output row U is downstream of that ledger. Componentwise transport, replay, provenance, and local naming preserve the same route, so the absent compactness and choice data have no carrier coordinate from which an accepted read could export them. □

Theorem 6294.3 (Finite-cover selector uniform output)**.** *Let* $S = (K, F, R, L, U, H, C, P, N)$ *be an accepted* $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$ *carrier. Every* $\mathsf{UniformModulus}^{\uparrow}$ *output exported by* S *factors through the finite-cover selector route*

$$\mathsf{CompactMetric}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{FiniteLebesgueNumber}^{\uparrow} \longmapsto \mathsf{FiniteCoverModulusSelector}^{\uparrow} \longmapsto \mathsf{UniformModulus}^{\uparrow}.$$

The exported read exposes the compact finite-cover row, pointwise modulus row, rational lower-bound row, selector ledger, transport, replay, provenance, and local naming. It exports no selected subcover, host minimum, selected Lebesgue number, choice schedule, or ambient completeness.

Proof. Start with the output coordinate U. By the carrier definition, U is not visible until the finite-cover row K, pointwise modulus row F, rational lower-bound row R, and selector ledger L have all been displayed.

Use the NameCert obligations of Definition 6294.2. They bind the source, finite-cover accountability, and lower-bound exactness needed by a compact-uniform-continuity consumer.

The route is then exactly the displayed BEDC path from compact metric and continuous-map data through the finite-radius ledger into the selector packet and the $\mathsf{UniformModulus}^{\uparrow}$ output. The structural rows H, C, P, N only transport, replay, source, and name that route.

Since selected subcovers, host minima, selected Lebesgue numbers, choice schedules, and ambient completeness do not occur among the carrier coordinates, no accepted output read can export them. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteCoverModulusSelector}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteCoverModulusSelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6294.1, Definition 6294.2, and Definition 6294.3 seed the finite-cover selector route from compact finite-cover data and pointwise moduli to UniformModulus output.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteCoverModulusSelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected subcovers, host minima, selected Lebesgue numbers, choice schedules, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, source compatibility, finite-cover accountability, rational lower-bound exactness, output non-escape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteCoverModulusSele

Constructive story: A $\mathsf{FiniteCoverModulusSelector}^{\uparrow}$ packet is generated from compact finite-cover rows, pointwise modulus rows, rational lower-bound rows, a selector ledger, a UniformModulus output row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6295

A Concrete Naming Certificate for $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$

The $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ packet realizes the ordered route of Definition 9462.553 as one finite NameCert carrier. It sits beside the kernel-inductive acceptance surface of chapter 915 and the audit-map family surface of chapter 807: those chapters name certificates and audit families, while this packet records the entire primitive-to-certified reading as one ordered carrier from generated distinction rows through the certified-object seal.

Falsifiable prediction: An accepted $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ read is refuted if a CertifiedObject seal is exported without first displaying distinction, history, record, classifier, NameCert, ledger, transport, replay, provenance, and local naming rows.

Independence witness: namecert, auditmapfamily, machineinterfacesocket, groundcompilerauditmap, and certificategate: none is bijective to $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ because only this packet records the ordered primitive-to-certified route as one carrier from generated distinction data through classifier, ledger, and certified-object seal.

Definition 6295.1 (Primitive-certified route carrier)**.** *A* $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, H_0, A, Q, N_0, L, Z, T, C, P, N)$$

with the following displayed rows:

1. *D is the generated distinction row, recording the primitive same-and-different boundary being certified.*
2. *H_0 is the history row, displaying the finite route by which the distinction entered the packet.*
3. *A is the record row, naming the generated record inspected by the classifier.*
4. *Q is the classifier row, assigning the record to the stable classifier surface consumed by the certificate.*
5. *N_0 is the inner* $\mathsf{NameCert}$ *row for the classified record.*
6. *L is the ledger row, recording the certificate-facing audit trail from D, H_0, A, Q, N_0.*
7. *Z is the CertifiedObject seal row exported only after the previous rows are displayed.*
8. *T records componentwise* hsame *transport for distinction, history, record, classifier, certificate, ledger, seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay through the ordered route $D, H_0, A, Q, N_0, L, Z, T, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PrimitiveCertifiedRoute}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the ordered rows $D, H_0, A, Q, N_0, L, Z, T, C, P, N$. It has no coordinate for a certified-object seal detached from the distinction, history, record, classifier, certificate, ledger, transport, replay, provenance, or local naming rows.

Theorem 6295.2 (Primitive-certified route NameCert obligations)**.** *For an accepted* $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ *carrier $R = (D, H_0, A, Q, N_0, L, Z, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PrimitiveCertifiedRoute}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed distinction, history, record, classifier, certificate, ledger, seal, transport, replay, provenance, and local naming rows;*
2. *primitive order, requiring D, H_0, A to be read before the classifier row Q is available;*
3. *classifier certification, requiring Q and N_0 to certify the same record before the ledger row L is accepted;*
4. *ledger seal ordering, requiring Z to be downstream of L rather than a standalone certified-object export;*
5. *structural non-escape, requiring T, C, P, N to transport, replay, source, and name only the ordered route already displayed.*

Proof. Project the carrier of Definition 6295.1. The listed rows are the whole surface available to an accepted read.

The primitive order obligation forces a read through D, H_0, A before the classifier row Q can be consumed. Thus the classifier is attached to a displayed generated record rather than to an unnamed object.

Classifier certification then binds Q to the inner NameCert row N_0. The ledger row L records that certification before the certified-object seal Z can be exported.

Finally T, C, P, N only preserve this ordered route. Because the carrier has no detached seal coordinate, every accepted certified-object read remains inside the displayed primitive-to-certified packet. □

Theorem 6295.3 (Primitive-certified route classifier nonescape)**.** *Every classifier-facing read of an accepted* $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ *packet factors through*

$$D, H_0, A, Q, N_0, L, Z, T, C, P, N.$$

In particular, the classifier row cannot export a CertifiedObject seal without the preceding distinction, history, record, NameCert, ledger, transport, replay, provenance, and local naming rows, and no sibling packet supplies a bijective replacement for that ordered route.

Proof. Start with the carrier row of Definition 6295.1. A classifier-facing read reaches Q only after the distinction, history, and record rows have been displayed.

Use the NameCert obligations of Definition 6295.2. They force certification through N_0, ledger recording through L, and seal export through Z in that order.

The structural rows T, C, P, N preserve the route but do not add a detached classifier or certified-object coordinate. The independence witness separates this packet from sibling NameCert,

audit-map, interface, compiler, and gate packets because none of them stores the same ordered primitive-to-certified carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{PrimitiveCertifiedRoute}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PrimitiveCertifiedRoute}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6295.1, Definition 6295.2, and Definition 6295.3 seed the ordered primitive-to-certified route from distinction rows through classifier, NameCert, ledger, and certified-object seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PrimitiveCertifiedRoute}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Origin: ai

Not claimed: This chapter does not close Lean verification, bridge checking, a detached certified-object seal, or a bijective replacement by NameCert, AuditMapFamily, MachineInterfaceSocket, GroundCompilerAuditMap, or CertificateGate packets.

Upgrade path: Obligation closure requires checked carrier admission, primitive ordering, classifier certification, ledger seal ordering, structural non-escape, transport, replay, provenance, and local naming for BEDC.Derived.PrimitiveCertifiedRouteUp.

Constructive story: A $\mathsf{PrimitiveCertifiedRoute}^{\uparrow}$ packet is generated from distinction, history, record, classifier, NameCert, ledger, certified-object seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming rows.

Chapter 6296

A Concrete Naming Certificate for $\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}$ packet records the finite mesh-budget surface used when a dyadic refinement is consumed by Real-facing and formal-ball consumers. It sits after the common-refinement route of chapter 1877, the formal-ball window of chapter 1510, the dyadic tolerance core of chapter 291, the stream-window surface of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite budget for mesh shrinkage and handoff; it does not assert open-cover compactness, selected limiting meshes, quotient equality of names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6296.1 (Finite dyadic refinement budget carrier)**.** *A* $\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, F, Q, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicCoverRefinement}^{\uparrow}$ *common-refinement row whose source covers and restriction ledgers determine the dyadic cells being tightened.*
2. *F is the* $\mathsf{FormalBall}^{\uparrow}$ *center-radius window whose radius request consumes the mesh budget before any completion-facing read is available.*
3. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the dyadic mesh scale, its endpoint arithmetic, and the finite comparison rows used to say that the refined mesh is small enough for the requested radius.*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window address at which the budget is read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback that turns the stream-window and dyadic ledger into a scheduled tolerance row.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal read by Real-facing consumers after Q, S, R have been displayed.*
7. *H records componentwise* hsame *transport for the refinement row, formal-ball window, dyadic ledger, stream address, regular rational readback, real seal, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay of the finite route*

$$D, F, Q, S, R, E \longmapsto H.$$

9. *P is the* Pkg *provenance over* FiniteDyadicRefinementBudget$^{\uparrow}$, DyadicCoverRefinement$^{\uparrow}$, FormalBall$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* NameCert$_{\text{FiniteDyadicRefinementBudget}^{\uparrow}}$ *row.*

The classifier of an accepted packet compares mesh-budget data only through these ten rows. It has no coordinate for open-cover compactness, selected subcovers, selected limiting meshes, quotient equality of streams or real names, countable choice, ambient Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.FiniteDyadicRefinementBudgetUp.FiniteDyadicRefinementBudgetTasteGate`
`single carrier alignment`

Theorem 6296.2 (Finite dyadic refinement budget NameCert obligations)**.** *For every accepted* FiniteDyadicRefinementBudget$^{\uparrow}$ *carrier*

$$B = (D, F, Q, S, R, E, H, C, P, N),$$

the local NameCert$_{\text{FiniteDyadicRefinementBudget}^{\uparrow}}$ *surface consists exactly of the following six rows:*

1. *carrier admission for the displayed* DyadicCoverRefinement$^{\uparrow}$ *row,* FormalBall$^{\uparrow}$ *window,* DyadicRatCore$^{\uparrow}$ *ledger,* StreamName$^{\uparrow}$ *address,* RegSeqRat$^{\uparrow}$ *readback,* Real$^{\uparrow}$ *seal, transport, replay, provenance, and local naming;*
2. *mesh-source exposure, requiring every budget request to name the common-refinement row D before it names a refined mesh;*
3. *formal-ball exposure, requiring the radius request in F to be visible before the budget is consumed by a completion-facing route;*
4. *dyadic scale exactness, requiring the mesh scale and endpoint comparisons to be read through Q;*
5. *regular-readback exactness, requiring the finite stream address S and regular rational row R to mediate the tolerance schedule before the real seal E is inspected;*
6. *classifier transport through componentwise* hsame, *displayed* Cont, Pkg *provenance, and local* NameCert *naming, with no escape to open-cover compactness, selected subcovers, selected limiting meshes, quotient equality, countable choice, ambient* Real$^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

No local NameCert obligation is stored outside these six rows.

Proof. Carrier admission is the projection of Definition 6296.1. The tuple displays the refinement row, formal-ball window, dyadic scale ledger, finite stream address, regular rational readback, real seal, and structural rows.

Mesh-source exposure is forced by D. The carrier has no detached mesh field; a refined mesh is read only as a budget attached to the common dyadic refinement route.

Formal-ball exposure is forced by F. The radius request that motivates the budget is part of the carrier, so a completion-facing consumer cannot read a budget before the formal-ball window has been displayed.

Dyadic scale exactness is the role of Q. Endpoint arithmetic, mesh scale comparison, and tolerance narrowing all remain in the dyadic ledger. The stream and regular rational rows S, R then mediate the scheduled tolerance before the terminal real seal E is available.

Finally inspect H, C, P, N. These rows transport, replay, source, and name the same finite route. Since the carrier contains no field for open-cover compactness, selected subcovers, selected limiting meshes, quotient equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking, the classifier cannot export such data. □

Theorem 6296.3 (Finite dyadic refinement budget mesh handoff)**.** *Let*

$$B = (D, F, Q, S, R, E, H, C, P, N)$$

be an accepted $\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}$ *carrier. Every consumer that receives a mesh-budget handoff from B factors through the displayed route*

$$\mathsf{DyadicCoverRefinement}^{\uparrow},\ \mathsf{FormalBall}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto D, F, Q, S, R, E, H,$$

Consequently the handoff exposes a finite common-refinement source, a formal-ball radius request, a dyadic mesh ledger, a finite stream address, a regular rational tolerance readback, a terminal real seal, transport, replay, provenance, and local naming.

Proof. Project the accepted carrier. The source of the mesh is D, and the consuming radius request is F; these are the only rows that connect the refinement route to formal-ball data.

Read the scale data next. The dyadic ledger Q supplies the mesh size and endpoint comparisons, while S records the finite window at which that budget is read.

The regular rational row R and real seal E are downstream of the finite window. Thus a Real-facing consumer sees the mesh budget only after the stream-window and scheduled-tolerance rows have already been displayed.

The remaining rows H, C, P, N preserve and name this route. Hence every mesh handoff factors through the finite packet and not through a detached compactness principle, selected mesh limit, quotient equality, choice schedule, or ambient real-completeness claim. □

Theorem 6296.4 (Finite dyadic refinement budget Real nonescape)**.** *Let $B = (D, F, Q, S, R, E, H, C, P, N)$ be an accepted* $\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}$ *carrier. A* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of B may inspect the terminal real seal E only after the rows D, F, Q, S, R have been displayed and replayed through H, C, P, N. In particular, the consumer cannot replace the finite budget route by open-cover compactness, a selected limiting mesh, quotient equality of names, countable choice, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier rows preceding E. The refinement source D, formal-ball request F, dyadic ledger Q, stream address S, and regular rational readback R are all explicit coordinates of the packet.

The real seal is terminal in this route. It records the Real-facing tolerance read only after the finite dyadic and regular rational rows have been exposed.

Transport and replay do not add analytic content. The rows H, C, P, N preserve the displayed packet and name it locally. Because the carrier contains no coordinate for compactness, selected limiting meshes, quotient equality, choice, or ambient completeness, no Real-facing read can escape the finite budget surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDyadicRefinementBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6296.1, Definition 6296.2, Definition 6296.3, and Definition 6296.4 seed the finite dyadic mesh-budget route through refinement, formal-ball, dyadic, stream, regular rational, real-seal, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus(NameCert$_{\text{FiniteDyadicRefinementBudget}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.FiniteDyadicRefinementBudgetUp.FiniteDyadicRefinementBudgetTasteGate\_single\_carrier\_`

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, Heine–Borel, selected subcovers, selected limiting meshes, quotient real equality, countable choice, ambient Real$^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires a checked BEDC.Derived.FiniteDyadicRefinementBudgetUp carrier, NameCert rows, mesh handoff, and Real nonescape theorem.

Constructive story: A FiniteDyadicRefinementBudget$^{\uparrow}$ packet is generated from a DyadicCoverRefinement source, a FormalBall radius request, a DyadicRatCore mesh ledger, StreamName finite windows, RegSeqRat tolerance readback, a RealUp terminal seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6297

A Concrete Naming Certificate for $\mathsf{RealCauchyFilterBasis}^{\uparrow}$

Following Bishop and Bridges on constructive real completion, the $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ packet records the finite filter-basis route by which regular rational sequence data, finite stream windows, and dyadic tolerance rows are read before a terminal $\mathsf{Real}^{\uparrow}$ seal is exposed. It sits between the Cauchy-filter surface of chapter 333, the finite window naming of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational handoff of chapter 301, and the real-name boundary of chapter 38. The packet names a finite basis for Real-facing completion reads; it is not a quotient-stream construction, a selected limit, a countable-choice principle, or an ambient completeness theorem.

Definition 6297.1 (Real Cauchy filter-basis carrier)**.** *A* $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (F, W, D, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the basis row, recording the finite Cauchy-filter basic neighborhoods and their inclusion reads.*
2. *W is the finite tail-window row, recording the* $\mathsf{StreamName}^{\uparrow}$ *windows and requested indices used to inspect each basis neighborhood.*
3. *D is the dyadic tolerance row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *radius, denominator budget, and slack comparison attached to each window.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the accepted tail-window observations as regular rational sequence data.*
5. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after F, W, D, R have displayed the basis, windows, tolerances, and regular-sequence readback.*
6. *H stores componentwise* hsame *transport for the basis, tail-window, dyadic tolerance, readback, Real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from a Real-facing precision request through the basis row, finite windows, dyadic tolerance ledger, regular-sequence readback, and terminal seal.*
8. *P records* Pkg *provenance over* $\mathsf{RealCauchyFilterBasis}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *data.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RealCauchyFilterBasis}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $F, W, D, R, Q, H, C, P, N$. There is no coordinate for quotient streams, selected representatives, selected limits, countable choice, host completeness, or ambient $\mathsf{Real}^{\uparrow}$ *equality.*

Theorem 6297.2 (Real Cauchy filter-basis NameCert obligations)**.** *For every accepted* $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ *carrier $B = (F, W, D, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RealCauchyFilterBasis}^{\uparrow}}$ *surface consists exactly of carrier admission, basis-neighborhood exposure, finite tail-window exactness, dyadic-tolerance exactness,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal descent, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local naming row for the same finite packet. No local obligation is stored outside these rows.*

Proof. Project the accepted packet $B = (F, W, D, R, Q, H, C, P, N)$. The carrier definition exposes exactly the basis, window, tolerance, readback, seal, transport, replay, provenance, and naming rows.

The basis-facing obligations are F, W, D. A consumer must first read the basic neighborhood, then the finite stream window that inspects it, and then the dyadic radius and slack comparison attached to that window.

The completion-facing obligations are R, Q, H, C, P, N. They repack the displayed data as regular rational readback, seal it as a Real-facing row, transport and replay the same route, cite its provenance, and name it locally. Since the carrier has no further coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6297.3 (Real Cauchy filter-basis RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *read exported by an accepted* $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyFilter}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}$$

inside the displayed rows F, W, D, R. Thus the regular rational sequence readback is available only after the basis neighborhood, finite tail window, and dyadic tolerance ledger have been displayed.

Proof. Begin with a consumer asking for the regular rational readback row R. By the carrier definition, the only route to R passes through the basis row F, the window row W, and the dyadic tolerance row D.

The row F names the Cauchy-filter basic neighborhood. The row W records the finite stream window used to inspect the tail of that neighborhood, and D records the dyadic radius and comparison budget consumed by the read.

The row R then repacks exactly those displayed observations as $\mathsf{RegSeqRat}^{\uparrow}$ data. The route therefore has no coordinate for quotient streams, selected representatives, selected limits, countable choice, or host completeness. □

Theorem 6297.4 (Real Cauchy filter-basis completion ledger)**.** *For every accepted carrier $B = (F, W, D, R, Q, H, C, P, N)$, a completion-facing read consumes the finite ledger*

$$F, W, D, R, Q, H, C, P, N$$

in that order: basis neighborhood, tail window, dyadic tolerance, regular rational readback, Real seal, transport, replay, provenance, and local naming. The ledger exports no quotient-stream equality, chosen representative, selected limit, countable-choice schedule, host-completeness theorem, or ambient Real equality.

Proof. Read the displayed continuation row C. It routes a Real-facing precision request through F, W, D, R, Q before the structural rows are consumed.

The first four rows form the finite completion ledger. They display the basis neighborhood, the inspected window, the dyadic tolerance comparison, and the regular rational readback.

The row Q is downstream of that ledger, while H, P, N preserve, source, and name the same data. Because the excluded objects do not occur as carrier coordinates, the completion-facing read cannot export them. □

Theorem 6297.5 (Real Cauchy filter-basis Real seal)**.** *The* $\mathsf{Real}^{\uparrow}$ *seal of an accepted* $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ *packet is terminal for the displayed basis route: every public read of* Q *factors through* F, W, D, R *and then through* H, C, P, N*. Consequently the seal certifies only the finite Real-facing basis packet, not quotient completion equality, selected limits, countable choice, host completeness, or ambient equality of real names.*

Proof. Let a public consumer read the terminal row Q. The carrier definition places Q after the basis, window, dyadic, and regular readback rows, so the consumer must first expose F, W, D, R.

Componentwise transport H and replay C keep the public read on that same finite route. Provenance P and local naming N identify the packet without adding a completion principle.

Thus the terminal seal is a Real-facing row of the displayed carrier. It is not a quotient equality, selected limit, choice schedule, host-completeness theorem, or ambient equality proof. □

Falsifiable prediction: Every $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ Real-facing read factors through a CauchyFilter basis row, StreamName finite tail windows, a DyadicRatCore tolerance row, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchyfilter, streamname, dyadicratcore, regseqrat, real: $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries basis neighborhoods, finite tail windows, dyadic tolerance comparisons, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealCauchyFilterBasis}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterBasis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6297.1, Definition 6297.2, Definition 6297.3, Definition 6297.4, and Definition 6297.5 seed the finite route from Cauchy-filter basis rows to regular rational readback and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RealCauchyFilterBasis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected representatives, selected limits, countable choice, host completeness, ambient $\mathsf{Real}^{\uparrow}$ equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, tail-window exactness, dyadic-tolerance exactness, RegSeqRat readback, Real seal descent, transport, replay, provenance, and local naming for BEDC.Derived.RealCauchyFilterBasisUp.

Constructive story: A $\mathsf{RealCauchyFilterBasis}^{\uparrow}$ packet is generated from a CauchyFilter basis row, finite StreamName tail windows, DyadicRatCore tolerance data, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6298

A Concrete Naming Certificate for $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ packet records the finite pair interface by which a displayed metric source and its dense Cauchy-name completion route are read before any $\mathsf{Real}^{\uparrow}$ or completion-facing consumer may inspect a seal. It sits beside the dyadic tolerance core of chapter 291, the finite window surface of chapter 126, the regular rational readback of chapter 301, the terminal Real seal of chapter 38, and the oscillation-bound route of chapter 448. The packet is a NameCert interface for the completion pair, not a quotient completion, selected representative, host limit, countable-choice schedule, or ambient completeness theorem.

Definition 6298.1 (Bishop Cauchy completion-pair carrier)**.** *A* $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (M, D, W, R, T, E, H, C, P, N)$$

with the following displayed rows:

1. *M is the source metric row, naming the displayed points, distance evidence, and finite request surface of the pre-completion space.*
2. *D is the dense-embedding row, recording how source points are inserted into the completion route before any completed endpoint is inspected.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to inspect the Cauchy names attached to the source and embedded rows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy seal over the windows in W.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger controlling the comparison precision for W and R.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff, available only after M, D, W, R, T have been displayed.*
7. *H records componentwise* hsame *transport for the metric source, dense embedding, finite windows, regular-Cauchy seal, dyadic ledger, Real seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay of the route from M and D through W, R, T, E, H, P, N.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{CauchyOscillation}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionPair}^{\uparrow}}$ *row.*

The carrier stores no quotient class, selected limit, host equality proof, countable-choice schedule, hidden completion point, or ambient completeness witness.

Theorem 6298.2 (Bishop Cauchy completion-pair NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ *carrier* $K = (M, D, W, R, T, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionPair}}$ *surface consists exactly of the following five obligation families:*

1. *carrier admission for the displayed metric source row* M*, dense-embedding row* D*,* $\mathsf{StreamName}^{\uparrow}$ *finite windows* W*,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy seal* R*,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger* T*, and* $\mathsf{Real}^{\uparrow}$ *seal handoff* E*;*
2. *dense-source routing, requiring every completion-facing read to pass from* M *through* D *before any window, readback, tolerance, or seal row is consumed;*
3. *regular-Cauchy exactness, requiring* W, R, T *to be read as one finite window-and-tolerance packet before* E *is public;*
4. *transport and replay stability for the same rows under componentwise* hsame *and displayed* Cont*;*
5. *provenance and local naming exhaustion through* P *and* N*, with no quotient completion, selected representative, host limit, countable choice, or ambient completeness row exported by the packet.*

Proof. Project the carrier of Definition 6298.1. Its analytic rows are exactly M, D, W, R, T, E, and its structural rows are exactly H, C, P, N.

The source route begins at M and D. A completion-facing consumer may inspect the metric source and its dense embedding, but neither row contains a completed endpoint or quotient representative.

The Cauchy-name route then opens the finite window W, regular readback R, and dyadic tolerance ledger T. These rows are the only data from which the Real seal handoff E can be reached.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the carrier has no coordinate for the refused host objects, the five listed obligation families exhaust the local NameCert surface. □

Theorem 6298.3 (Bishop Cauchy completion-pair routed seal)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ *packet reads the terminal seal only through the finite route*

$$M \longmapsto D \longmapsto W \longmapsto R \longmapsto T \longmapsto E.$$

Equivalently, the source metric data, dense embedding, $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-Cauchy seal, and* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger must all be displayed before the* $\mathsf{Real}^{\uparrow}$ *seal handoff is public.*

Proof. The NameCert obligations of Definition 6298.2 order the carrier rows. Source admission exposes M, and dense-source routing forces the embedding row D to be read before the Cauchy-name rows are interpreted.

Regular-Cauchy exactness then ties W, R, T into one finite packet. The seal row E is downstream of that packet, so a consumer has no route to E that bypasses the displayed windows, readback, or tolerance ledger.

Transport, replay, provenance, and naming preserve the same route componentwise. Thus the exported seal is a finite completion-pair handoff rather than a quotient completion, selected representative, host limit, choice principle, or ambient completeness witness. □

Falsifiable prediction: A claimed $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ consumer that reaches $\mathsf{Real}^{\uparrow}$ or a completion-facing seal without displaying M, D, W, R, T refutes the carrier.

Independence witness: dyadicratcore, streamname, regseqrat, real, cauchyoscillation: no listed sibling is bijective to $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ because the packet jointly carries a source metric row, dense embedding, finite windows, regular-Cauchy seal, dyadic tolerance ledger, Real seal handoff, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionPair}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$ packet is generated from source metric data, a dense embedding row, StreamName finite windows, RegSeqRat regular-Cauchy seal, DyadicRatCore tolerance ledger, Real seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionPair}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6298.1, Definition 6298.2, and Definition 6298.3 seed the finite Bishop completion-pair route from metric source and dense embedding through windows, readback, tolerances, and Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyCompletionPair}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completions, selected representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopCauchyCompletionPair}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, dense-source routing, regular-Cauchy exactness, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyCompletionPairUp.

Chapter 6299

A Concrete Naming Certificate for JordanDecompositionBoundedVariation$^{\uparrow}$

Following Jordan's theorem for functions of bounded variation, the JordanDecompositionBoundedVariation$^{\uparrow}$ packet records the finite BEDC surface by which a bounded-variation real function is read as the difference of two increasing variation ledgers before any Riemann–Stieltjes or Real-facing consumer sees the result. It sits after the bounded-variation carrier of chapter 398, the finite partition-sum ledger of chapter 6254, the regular rational readback of chapter 301, the dyadic edge-observation route of chapter 291, and the terminal Real seal of chapter 38. The packet is a displayed Jordan split over finite partition rows; it does not introduce a host function quotient, a selected total-variation supremum, a measure-valued derivative, countable choice, or ambient Real$^{\uparrow}$ completeness.

Definition 6299.1 (Jordan decomposition bounded-variation carrier)**.** *A* JordanDecompositionBoundedVariation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$J = (B, L, P_+, P_-, M, R, D, E, H, C, Q, N)$$

with the following displayed rows:

1. *B is the accepted* BoundedVariation$^{\uparrow}$ *source row.*
2. *L is the finite* FiniteVariationLedger$^{\uparrow}$ *partition-sum row attached to the same interval and endpoint data.*
3. *P_+ records the positive-edge accumulated ledger over the displayed partition.*
4. *P_- records the negative-edge accumulated ledger over the displayed partition.*
5. *M is the monotonicity witness row for the two accumulated ledgers on common refinements.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row for the endpoint and accumulated-value windows.*
7. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row for edge signs, absolute-value reads, and common-refinement comparisons.*
8. *E is the terminal* Real$^{\uparrow}$ *seal row reached only after B, L, P_+, P_-, M, R, D have been displayed.*
9. *H records componentwise* hsame *transport for all source, split, readback, tolerance, Real-seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a bounded-variation request through finite partitions, signed edge ledgers, monotone accumulated rows, rational readback, dyadic tolerance, and terminal Real sealing.*

11. *Q is the* Pkg *provenance over* $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$, $\mathsf{BoundedVariation}^{\uparrow}$, $\mathsf{FiniteVariationLedger}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}$ *row.*

Accepted packets are compared only through these displayed rows.

Theorem 6299.2 (Jordan decomposition bounded-variation NameCert obligations)**.** *For every accepted* $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ *carrier* $J = (B, L, P_+, P_-, M, R, D, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}$ *surface consists exactly of bounded-variation source admission, finite variation-ledger exposure, positive-edge and negative-edge accumulated ledgers, monotonicity on common refinements, RegSeqRat readback, DyadicRatCore tolerance accounting, Real seal exposure, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6299.1. The source and partition data are precisely B and L, and the two signed accumulated ledgers are P_+ and P_-.

The Jordan split is finite at every read. Edge signs and absolute-value comparisons are opened through D, accumulated over the displayed partition in P_+ and P_-, and checked on common refinements by M.

The readback and sealing rows are downstream. The row R exposes regular rational windows for endpoint and accumulated-value reads, and E seals only the Real-facing boundary reached from those finite rows.

Finally H, C, Q, N transport, replay, source, and name the same packet. Since no row stores a host function quotient, a selected global supremum, a measure derivative, countable choice, or ambient Real completeness, none of those objects is exported by the local NameCert surface. □

Theorem 6299.3 (Jordan decomposition bounded-variation consumer route)**.** *Every Riemann–Stieltjes, bounded-variation, or Real-facing consumer of an accepted* $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ *packet must first read the route*

$$\mathsf{BoundedVariation}^{\uparrow},\ \mathsf{FiniteVariationLedger}^{\uparrow},\ P_+,\ P_-,\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto H, C, Q, N.$$

Thus the difference-of-increasing-functions handoff is exposed as signed finite variation ledgers over common refinements, not as a host-level decomposition of quotient functions or a selected supremum over all partitions.

Proof. Apply Definition 6299.2. A consumer cannot read the positive and negative accumulated ledgers until the bounded-variation source, finite partition ledger, edge signs, and common-refinement row have been displayed.

The handoff to analysis consumers then passes through the rational and Real boundary. The $\mathsf{RegSeqRat}^{\uparrow}$ row R records endpoint and accumulated-value windows, the $\mathsf{DyadicRatCore}^{\uparrow}$ row D records the tolerance and edge-comparison ledger, and the $\mathsf{Real}^{\uparrow}$ row E seals only the displayed result.

The structural rows H, C, Q, N preserve that route for downstream Riemann–Stieltjes and bounded-variation readers. They add no host function extensionality, quotient-function equality, selected limiting partition, countable choice, or ambient Real-completeness coordinate. □

Theorem 6299.4 (Jordan decomposition signed ledger exposure)**.** *For every accepted* JordanDecompositionBounde*carrier* $J = (B, L, P_+, P_-, M, R, D, E, H, C, Q, N)$*, the positive accumulated ledger* P_+ *and negative accumulated ledger* P_- *are exposed only after the bounded-variation source row* B *and the finite variation-ledger row* L *have been displayed. A reader that asks for either signed accumulated component without first reading* B *and* L *has not reached the local* $\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}$ *surface.*

Proof. Project the carrier of Definition 6299.1. Its row order makes B the source admission and L the finite partition-sum ledger attached to the same interval and endpoint data.

The signed accumulated rows P_+ and P_- are downstream of those two rows. Apply the NameCert obligation surface of Definition 6299.2: positive and negative edge accumulation is one obligation, but it is exposed only together with bounded-variation source admission and finite variation-ledger exposure.

Thus a consumer that omits B or L can at most name an external signed decomposition. It cannot read the displayed BEDC Jordan packet, because the local certificate has not opened the finite ledger on which P_+ and P_- depend. □

Theorem 6299.5 (Jordan decomposition common-refinement monotonicity)**.** *For every accepted* $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ *carrier* $J = (B, L, P_+, P_-, M, R, D, E, H, C, Q, N)$*, common-refinement monotonicity for the positive and negative accumulated ledgers is read through the displayed row* M *and transported componentwise by* H*. It is not a host theorem about all partitions, nor a selected supremum over a refinement family.*

Proof. The carrier of Definition 6299.1 assigns the common-refinement witness to the single row M. Therefore a monotonicity read has a finite source: the displayed accumulated ledgers and their stated common-refinement comparison row.

Transport is also rowwise. The componentwise hsame row H preserves the source, ledger, signed accumulated components, readback, tolerance, Real-seal, replay, provenance, and local naming rows without replacing them by a host partition family.

Consequently monotonicity reaches consumers only as the M row transported by H. No total-variation supremum, quotient function, or ambient completeness principle is added by this route. □

Theorem 6299.6 (Jordan decomposition Real seal nonescape)**.** *For every accepted* $\mathsf{JordanDecompositionBoundedVa}$ *carrier* $J = (B, L, P_+, P_-, M, R, D, E, H, C, Q, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal* E *is reached only through the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row* R *and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row* D*. The seal exports no selected total-variation supremum, host function quotient, measure derivative, countable choice, or ambient Real completeness.*

Proof. The carrier separates rational readback, tolerance accounting, and Real sealing into the rows R, D, E. The row R reads endpoint and accumulated-value windows, while D accounts for edge signs, absolute-value reads, and common-refinement comparisons.

Apply the consumer route of Definition 6299.3. It sends Real-facing readers through $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and only then $\mathsf{Real}^{\uparrow}$, so E is terminal rather than a new source of analytic data.

Since the rows before E are finite readback and tolerance rows, the terminal seal cannot introduce a selected supremum, a host quotient of functions, a measure derivative, countable choice, or ambient Real completeness. It records only the Real-facing boundary already supported by R and D. □

Falsifiable prediction: A purported $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ consumer that reads a positive or negative monotone component while omitting the bounded-variation source row, finite variation ledger, signed edge rows, RegSeqRat readback, DyadicRatCore tolerance, Real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: boundedvariation, finite-variation-ledger, regseqrat, dyadicratcore, real: $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ is not bijective with any listed sibling because it joins the

bounded-variation source, signed finite partition ledgers, monotone refinement rows, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming as one decomposition packet.

Closure status for $\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}$.

Constructive story: A $\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}$ packet is generated from a BoundedVariationUp source row, a FiniteVariationLedgerUp partition-sum row, positive and negative signed edge ledgers, common-refinement monotonicity rows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}, \mathsf{obligation})$

Scope closed: Definition 6299.1, Definition 6299.2, Definition 6299.3, Definition 6299.4, Definition 6299.5, and Definition 6299.6 close the bounded-variation source admission, signed finite-ledger exposure, common-refinement monotonicity, RegSeqRat readback, DyadicRatCore tolerance accounting, and Real seal nonescape obligations for bounded-variation and Riemann–Stieltjes consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{JordanDecompositionBoundedVariation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, host function quotients, selected total-variation suprema, measure-valued derivatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Scoped closure requires checked transport, replay, provenance, and local naming routes for BEDC.Derived.JordanDecompo

Chapter 6300

A Concrete Naming Certificate for $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$

Following the finite-cover proof of Heine–Cantor in chapter 9237, the $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$ packet records the compact-cover radius and dyadic oscillation ledger consumed before a uniform-continuity handoff is exposed. It sits beside the compact finite-window route of chapter 6293, the compact oscillation packet of chapter 3737, the finite lower-bound fold of chapter 4096, and the uniform modulus consumer of chapter 369. The packet is finite radius data over displayed covers and oscillation rows; it does not assert open-cover compactness, a selected Lebesgue number outside the ledger, quotient map equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6300.1 (Finite oscillation-radius ledger carrier)**.** *A* $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (K, U, A, F, B, W, R, D, S, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-cover or finite-net source row.*
2. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *consumer row receiving the final tolerance-only radius handoff.*
3. *A records the displayed center-radius cover cells and their rational lower-bound candidates.*
4. *F is the finite lower-bound fold row over those candidate radii.*
5. *B is the dyadic oscillation-bound row attached to each displayed cover cell.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row for source and target observations.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for those observations.*
8. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for oscillation and radius comparisons.*
9. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, U, A, F, B, W, R, D have been displayed.*
10. *H records componentwise* hsame *transport for all cover, fold, oscillation, readback, tolerance, seal, replay, provenance, and local naming rows.*
11. *C records displayed* Cont *replay from a compact-cover request through radius candidates, finite lower-bound folding, dyadic oscillation checks, uniform-modulus handoff, and terminal Real sealing.*

12. *P is the* Pkg *provenance over* FiniteOscillationRadiusLedger$^{\uparrow}$, UniformModulus$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{FiniteOscillationRadiusLedger}^{\uparrow}}$ *row.*

Accepted packets are compared only through the displayed finite cover, radius, fold, oscillation, readback, tolerance, seal, transport, replay, provenance, and naming rows.

Lean4 ✓ BEDC.Derived.FiniteOscillationRadiusLedgerUp.FiniteOscillationRadiusLedgerTasteGate
single carrier alignment

Theorem 6300.2 (Finite oscillation-radius ledger NameCert obligations)**.** *For every accepted* FiniteOscillationRadiusLedger$^{\uparrow}$ *carrier $O = (K, U, A, F, B, W, R, D, S, H, C, P, N)$, the local* NameCert$_{\text{FiniteOscillationRad}}$ *surface consists exactly of compact finite-cover exposure, uniform-modulus consumer exposure, displayed center-radius candidates, finite lower-bound folding, dyadic oscillation bounds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, Real sealing, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6300.1. The compact-cover source and uniform-modulus consumer are K and U, while A, F, B supply the finite radius and oscillation ledger.

The radius handoff is finite. Candidate radii are listed in A, folded in F, and checked against the displayed dyadic oscillation row B before any downstream modulus row is read.

The readback remains on the same route. Rows W, R, D, S expose finite observations, regular rational windows, dyadic tolerance comparisons, and the terminal Real seal after the radius ledger has been displayed.

Finally H, C, P, N transport, replay, source, and name the same packet. Since no row stores open-cover compactness, an external Lebesgue-number selector, quotient map equality, countable choice, or ambient Real completeness, the local NameCert surface cannot export those objects. □

Theorem 6300.3 (Finite oscillation-radius ledger uniform handoff)**.** *Every compact-uniform-continuity consumer of an accepted* FiniteOscillationRadiusLedger$^{\uparrow}$ *packet must read compact finite-cover rows, center-radius candidates, the finite lower-bound fold, dyadic oscillation bounds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, and the terminal Real seal before it consumes the* UniformModulus$^{\uparrow}$ *handoff. The handoff is a finite ledger route, not an open-cover compactness theorem or a selected radius outside the packet.*

Proof. Apply Definition 6300.2. The compact-cover row, radius candidates, lower-bound fold, and dyadic oscillation row are all displayed before the uniform-modulus consumer may read the handoff.

The finite readback rows keep the same route visible. A consumer reaches W, R, D, S only after the cover and oscillation ledger has fixed the radius row, so the final Real-facing seal cannot replace the finite ledger by a host compactness or completeness argument.

The structural rows H, C, P, N preserve the same compact-cover and oscillation-radius path. They add no selected Lebesgue number outside the ledger, quotient map equality, countable choice, or ambient Real-completeness coordinate. □

Theorem 6300.4 (Finite oscillation-radius compact-uniform sibling link)**.** *Every accepted* FiniteOscillationRadiusLe *compact-uniform reader factors through* CompactUniformOscillationWindow$^{\uparrow}$, CompactUniformOscillation$^{\uparrow}$, FiniteLowerBoundFold$^{\uparrow}$, UniformModulus$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, *and* Real$^{\uparrow}$ *before the uniform handoff is read. For a carrier $O = (K, U, A, F, B, W, R, D, S, H, C, P, N)$, the rows $K, U, A, F, B, W, R, D, S, H, C, P, N$ display that route in the order required by the reader.*

Proof. Project the carrier of Definition 6300.1. The compact source row K and the center-radius candidate row A provide the compact-cover side of the route, while B is the oscillation-bound row read through chapter 6293 and chapter 3737.

The radius extraction is finite. Row F is the $\mathsf{FiniteLowerBoundFold}^{\uparrow}$ fold of chapter 4096, and row U is the $\mathsf{UniformModulus}^{\uparrow}$ consumer row of chapter 369; therefore the uniform handoff is read only after the finite lower-bound and oscillation rows have been displayed.

The readback and seal rows keep the same path visible. Rows W, R, D, S pass through chapter 126, chapter 301, chapter 291, and chapter 38 before any compact-uniform consumer receives the final handoff.

Finally H, C, P, N transport, replay, source, and name the displayed route. These rows preserve the compact-uniform sibling dependency; they do not add host compactness, an external Lebesgue-number selector, quotient map equality, countable choice, or ambient Real completeness. □

Falsifiable prediction: Every accepted $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$ reader that claims a uniform-modulus handoff while omitting compact finite-cover exposure, center-radius candidates, finite lower-bound folding, dyadic oscillation bounds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, Real sealing, hsame transport, Cont replay, Pkg provenance, or local naming refutes the packet.

Independence witness: compact-uniform-oscillation-window, compact-uniform-oscillation, finite-lower-bound-fold, uniformmodulus, streamname, regseqrat, dyadicratcore, real: $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$ is not bijective with any listed sibling because it joins compact-cover radius candidates, dyadic oscillation checks, finite lower-bound folding, uniform-modulus handoff, readback, tolerance, Real sealing, transport, replay, provenance, and local naming as one ledger.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}$ packet is generated from compact finite-cover rows, UniformModulusUp consumer rows, displayed center-radius candidates, a FiniteLowerBoundFoldUp row, dyadic oscillation bounds, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance accounting, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6300.1, Definition 6300.2, Definition 6300.3, and Definition 6300.4 seed the compact-cover radius, dyadic oscillation, sibling dependency, and uniform-continuity handoff route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteOscillationRadiusLedger}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Lean verification, open-cover compactness, selected Lebesgue numbers outside the packet, quotient map equality, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-cover exposure, radius-candidate exposure, finite lower-bound folding, dyadic oscillation accounting, uniform-modulus handoff, finite readback, tolerance accounting, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.FiniteOscillationRadiusLedgerUp.

Chapter 6301

A Concrete Naming Certificate for $\mathsf{BishopRegularRealSeal}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRegularRealSeal}^{\uparrow}$ packet records the regular-rational seal by which a displayed Cauchy name reaches the public $\mathsf{Real}^{\uparrow}$ boundary. It sits between the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the completion modulus discipline of chapter 1469, and the terminal real-name seal of chapter 38. The packet names only finite windows, regularity budgets, dyadic tolerances, seal transport, replay, provenance, and a local certificate; it does not allocate quotient streams, selected limits, countable-choice schedules, host real equality, or ambient completeness.

Definition 6301.1 (Bishop regular-real seal carrier)**.** *A* $\mathsf{BishopRegularRealSeal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (W, G, D, M, E, H, C, P, N)$$

with the following displayed rows:

1. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the rational-name observations available to the seal.*
2. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the same displayed windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that supplies the dyadic precision and endpoint comparisons consumed by G.*
4. *M is the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row recording the explicit completion modulus that makes the windows in W reusable for all later seal reads.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *boundary seal, available only after W, G, D, M have been displayed.*
6. *H stores componentwise* hsame *transport for the stream windows, regular readback, dyadic ledger, completion modulus, real seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay for every consumer route from W, G, D, M to E.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularRealSeal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{BishopCompletionModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealSeal}^{\uparrow}}$ *row.*

The carrier stores no quotient stream, selected representative, host limit, countable-choice schedule, real trichotomy, or ambient completeness witness.

Theorem 6301.2 (Bishop regular-real seal NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularRealSeal}^{\uparrow}$ *carrier* $R = (W, G, D, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularRealSeal}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows* $W, G, D, M, E, H, C, P, N$*, stream-window exposure through* W*, regular-rational readback through* G*, dyadic tolerance exposure through* D*, completion-modulus reuse through* M*, terminal real-seal exposure through* E*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6301.1. The analytic rows are exactly the stream windows, regular readback, dyadic ledger, completion modulus, and real seal, and the structural rows are exactly transport, replay, provenance, and local naming.

The window route begins at W. A consumer may inspect only the displayed finite observations named by the $\mathsf{StreamName}^{\uparrow}$ row before any regular-rational or real-facing read is available.

The regularity route then reads G, D, M together. The $\mathsf{RegSeqRat}^{\uparrow}$ readback consumes the dyadic tolerance row D, while the $\mathsf{BishopCompletionModulus}^{\uparrow}$ row records the completion budget that lets later consumers reuse the same windows.

Finally E, H, C, P, N expose, transport, replay, source, and name the terminal $\mathsf{Real}^{\uparrow}$ seal over that finite route. Since no carrier coordinate stores quotient streams, selected limits, host equality, countable choice, trichotomy, or ambient completeness, the listed obligations exhaust the local NameCert surface. □

Theorem 6301.3 (Bishop regular-real seal RegSeqRat route)**.** *Every accepted* $\mathsf{BishopRegularRealSeal}^{\uparrow}$ *readback to* $\mathsf{RegSeqRat}^{\uparrow}$ *factors through the displayed route*

$$W \longmapsto D \longmapsto M \longmapsto G.$$

The stream-window row W *supplies the finite observations, the* $\mathsf{DyadicRatCore}^{\uparrow}$ *row* D *supplies the tolerance ledger, the* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row* M *supplies the reusable completion budget, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *row* G *is the only regular rational readback exported by the packet.*

Proof. Start with Definition 6301.1. The row G is not a detached regular sequence; it appears only with the window row W, the dyadic ledger D, and the completion modulus M.

Apply Definition 6301.2. Stream-window exposure precedes regular readback, dyadic tolerance exposure supplies the comparison budget, and completion-modulus reuse records the stable tail discipline for later consumers.

Therefore any accepted $\mathsf{RegSeqRat}^{\uparrow}$ readback uses the route W, D, M, G. A quotient stream, selected representative, host limit, or choice schedule has no row from which to replace that route. □

Theorem 6301.4 (Bishop regular-real seal Real boundary)**.** *Every terminal* $\mathsf{Real}^{\uparrow}$ *boundary read exported by an accepted* $\mathsf{BishopRegularRealSeal}^{\uparrow}$ *carrier factors through the finite route*

$$W \longmapsto G \longmapsto D \longmapsto M \longmapsto E \longmapsto H, C, P, N.$$

In particular, the public real-name boundary is reached only after the $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, and* $\mathsf{BishopCompletionModulus}^{\uparrow}$ *row have all been displayed.*

Proof. Read the terminal seal row in Definition 6301.1. The row E is stored downstream of W, G, D, M, so it cannot be consumed before the finite regularity route has been displayed.

Use Definition 6301.3. The regular rational readback already factors through the stream windows, dyadic tolerances, and completion modulus. Hence the terminal seal has the same preceding route whenever a real-facing consumer inspects it.

Finally apply Definition 6301.2. Transport, replay, provenance, and local naming preserve the same finite surface after E is reached. The exported $\mathsf{Real}^{\uparrow}$ boundary therefore contains no quotient stream row, selected limit, host real equality, countable-choice schedule, trichotomy, or ambient completeness coordinate. □

Falsifiable prediction: An accepted $\mathsf{BishopRegularRealSeal}^{\uparrow}$ consumer that reaches the terminal $\mathsf{Real}^{\uparrow}$ boundary without displaying $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerances, and a $\mathsf{BishopCompletionModulus}^{\uparrow}$ row refutes the carrier.

Independence witness: streamname, regseqrat, dyadicratcore, bishop-completion-modulus, real: no listed sibling is bijective to $\mathsf{BishopRegularRealSeal}^{\uparrow}$ because the packet jointly carries finite windows, regular readback, dyadic tolerances, completion-modulus reuse, terminal Real seal exposure, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularRealSeal}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularRealSeal}^{\uparrow}$ packet is generated from StreamName finite windows, a RegSeqRat readback row, a DyadicRatCore tolerance ledger, a BishopCompletionModulus row, a terminal RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularRealSeal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6301.1, Definition 6301.2, Definition 6301.3, and Definition 6301.4 seed the finite regular-real seal route from displayed windows and tolerances to the public Real boundary.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularRealSeal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient streams, selected limits, host real equality, countable choice, trichotomy, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopRegularRealSeal}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, regular readback routing, completion-modulus reuse, Real boundary nonescape, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularRealSealUp.

Chapter 6302

A Concrete Naming Certificate for $\mathsf{SyntacticMonoid}^{\uparrow}$

Following Myhill, Nerode, and Eilenberg, the $\mathsf{SyntacticMonoid}^{\uparrow}$ packet records the finite classifier monoid of a regular language as BEDC NameCert data. It sits beside the accepted-word surface of chapter 351, the right-congruence classifier of chapter 5729, and the canonical deterministic presentation of chapter 6188. The packet names finite congruence classes and their multiplication table before any language consumer may read a transition monoid, host quotient of all words, arbitrary automaton presentation, or regular-expression proof.

Definition 6302.1 (Syntactic monoid carrier)**.** *A* $\mathsf{SyntacticMonoid}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (A, W, L, E, C, \mu, e, \rho, H, K, P, N)$$

with the following displayed rows:

1. *A is the alphabet row, exposing the finite symbols available to the word-spine reader.*
2. *W is the word-spine row, recording finite word concatenation windows over A.*
3. *L is the regular-language observation row read from* $\mathsf{RegularLanguage}^{\uparrow}$*.*
4. *E is the syntactic congruence row, tabulating when displayed contexts make two words indistinguishable by L.*
5. *C is the finite class row, naming the displayed congruence classes admitted by E.*
6. *μ is the multiplication row, recording class multiplication induced by finite word concatenation.*
7. *e is the identity-class row, naming the class of the empty word.*
8. *ρ is the recognition morphism row from displayed words to classes and from accepted classes back to the regular-language observation.*
9. *H records componentwise* hsame *transport for alphabet, word-spine, language observation, congruence, class, multiplication, identity, recognition, replay, provenance, and local naming rows.*
10. *K records displayed* Cont *replay from word observations through congruence-class multiplication and recognition.*
11. *P is the* Pkg *provenance over* $\mathsf{SyntacticMonoid}^{\uparrow}$*,* $\mathsf{RegularLanguage}^{\uparrow}$*,* $\mathsf{MyhillNerode}^{\uparrow}$*,* $\mathsf{MinimalDfa}^{\uparrow}$*,* $\mathsf{List}^{\uparrow}$*,* BHist*,* hsame*,* Cont*,* Pkg*, and* $\mathsf{NameCert}$ *rows.*

12. N is the local $\mathsf{NameCert}_{\mathsf{SyntacticMonoid}^\uparrow}$ *row.*
The classifier compares accepted packets only through $A, W, L, E, C, \mu, e, \rho, H, K, P, N$. It has no coordinate for a host quotient over all words, selected representatives outside the displayed classes, arbitrary transition monoid presentation, nondeterministic branch, or regular-expression equivalence proof.

Theorem 6302.2 (Syntactic monoid NameCert obligations)**.** *For every accepted* $\mathsf{SyntacticMonoid}^\uparrow$ *carrier $M = (A, W, L, E, C, \mu, e, \rho, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SyntacticMonoid}^\uparrow}$ *surface consists exactly of alphabet exposure, word-spine exposure, regular-language observation, syntactic-congruence tabulation, finite class admission, multiplication-table replay, identity-class naming, recognition morphism readback, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6302.1. The alphabet row A, word-spine row W, and regular-language row L are displayed before any congruence or monoid data can be consumed.

Read E, C, μ, e, ρ next. The congruence row E classifies displayed words by finite context tests against L, the class row C names the admitted finite classes, and μ and e record multiplication and identity only on those classes. The recognition row ρ is therefore a finite route from words to classes and back to the accepted-language observation.

The structural rows H, K, P, N transport, replay, source, and name the same finite table. Since no further coordinate exists, the carrier cannot import a host word quotient, external representatives, arbitrary transition monoid presentation, nondeterministic branch, or regular-expression equivalence proof. □

Theorem 6302.3 (Syntactic monoid regular-language handoff)**.** *Every* $\mathsf{RegularLanguage}^\uparrow$, $\mathsf{MyhillNerode}^\uparrow$, *or* $\mathsf{MinimalDfa}^\uparrow$ *consumer of an accepted* $\mathsf{SyntacticMonoid}^\uparrow$ *packet first reads the displayed alphabet row, word-spine row, regular-language observation, syntactic congruence row, finite class row, multiplication table, identity class, and recognition morphism. The handoff through chapter 351, chapter 5729, and chapter 6188 therefore exports only the finite classifier-monoid route.*

Proof. Start with a consumer read of $M = (A, W, L, E, C, \mu, e, \rho, H, K, P, N)$. The regular-language sibling reads L only after the finite alphabet and word-spine rows have been displayed.

The Myhill–Nerode sibling then consumes the congruence row E as a finite right-congruence classifier. The finite class row C and multiplication table μ keep this handoff inside displayed class data rather than a host quotient of all words.

The minimal deterministic automaton sibling can read the recognition morphism ρ only after the identity row e and class multiplication have been shown. Thus the syntactic-monoid packet supplies a finite classifier monoid to those consumers, not an arbitrary automaton presentation or regular-expression proof. □

Closure status for $\mathsf{NameCert}_{\mathsf{SyntacticMonoid}^\uparrow}$**.**

Constructive story: A $\mathsf{SyntacticMonoid}^\uparrow$ packet is generated from finite alphabet, word-spine, regular-language observation, syntactic congruence, finite class, multiplication, identity, recognition, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SyntacticMonoid}^\uparrow}$, seed)

Scope closed: Definition 6302.1, Definition 6302.2, and Definition 6302.3 seed the finite classifier-monoid route over RegularLanguageUp, MyhillNerodeUp, and MinimalDfaUp rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SyntacticMonoid}^\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, transition-monoid equivalence, arbitrary automaton minimization, regular-expression equivalence, host quotients of all words, or selected external representatives.

Upgrade path: Obligation closure requires checked carrier admission, syntactic congruence stability, multiplication replay, identity exactness, recognition morphism readback, sibling handoff, transport, replay, provenance, and local naming for BEDC.Derived.SyntacticMonoidUp.

Chapter 6303

A Concrete Naming Certificate for $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$

The $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ packet realizes the five-field explanation schematic of Definition 9462.560 as finite BEDC NameCert data. It sits beside the machine-readable audit interface of chapter 1232, the registry export surface of chapter 1216, the philosophy formal-target ledger of chapter 1282, the tensor NameCert surface, and the GroundCompiler audit-map packet of chapter 836. The packet names a certified explanation route from source and classifier rows through evidence, ledger gaps, and certification rather than treating explanatory prose as proof evidence.

Falsifiable prediction: A $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ packet is refuted when a claimed explanation omits one of the source, classifier, evidence, ledger-gap, certification-route, transport, replay, provenance, or local naming rows, or when a ledger gap is presented as closed without a displayed certification route.

Independence witness: machinereadableauditinterface, registryexportsurface, philosophy-formal-target-ledger, tensornamecert, groundcompilerauditmap: no sibling is bijective to $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ because each lacks at least one of the source, classifier, evidence, ledger-gap, certification-route, transport, replay, provenance, or local NameCert rows as one explanation packet.

Definition 6303.1 (Certified explanation schematic carrier)**.** *A* $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (S, C, E, G, R, H, K, P, N)$$

with the following displayed rows:

1. *S is the source row, recording the behavior, claim, or object whose explanation is being certified.*
2. *C is the classifier row, recording the candidate explanatory classifier for the source.*
3. *E is the evidence row, exposing finite observations, proof-trace summaries, registry rows, or audit rows relevant to the classifier.*
4. *G is the ledger-gap row, recording the five-field deficit coordinates for source, pattern, classifier, stability, and ledger gaps.*
5. *R is the certification-route row, recording which displayed rows close the relevant coordinates and which remain refused.*

6. *H records componentwise* hsame *transport for source, classifier, evidence, gap, certification-route, replay, provenance, and local naming rows.*
7. *K records displayed* Cont *replay from source and classifier reads through evidence, ledger-gap accounting, and certification-route checking.*
8. *P is the* Pkg *provenance over* $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$, $\mathsf{MachineReadableAuditInterface}^{\uparrow}$, $\mathsf{RegistryExportSurface}^{\uparrow}$, $\mathsf{PhilosophyFormalTargetLedger}^{\uparrow}$, $\mathsf{GroundCompilerAuditMap}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CertifiedExplanationSchematic}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, C, E, G, R, H, K, P, N$. It has no coordinate for prose-only explanation, hidden proof evidence, unchecked export, unstated ledger closure, or a detached TasteGate verdict.

Theorem 6303.2 (Certified explanation schematic NameCert obligations)**.** *For every accepted* $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ *carrier $X = (S, C, E, G, R, H, K, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CertifiedExplanationSchematic}^{\uparrow}}$ *surface consists exactly of source exposure, classifier exposure, evidence readback, five-field ledger-gap accounting, certification-route checking, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming.*

Proof. Project the carrier of Definition 6303.1. The source row S and classifier row C fix the explanation target before evidence or closure data can be read.

Read the evidence and gap rows. The row E supplies the finite observation or audit material, while G records the five deficit coordinates from Definition 9462.560. A coordinate is not closed merely because it is named; it must be routed through R.

The certification-route row R records which displayed rows discharge the gap coordinates and which claims remain refused. The structural rows H, K, P, N transport, replay, source, and name the same route, so no prose-only explanation, hidden proof evidence, unchecked export, unstated ledger closure, or detached TasteGate verdict can enter the local NameCert surface. □

Theorem 6303.3 (Certified explanation schematic vision handoff)**.** *Every concrete read of the five-field schematic through* $\mathsf{CertifiedExplanationSchematic}^{\uparrow}$ *factors through the finite route*

$$S, C \longmapsto E, G \longmapsto R \longmapsto H, K, P, N.$$

The handoff to machine-readable audit, registry export, philosophy formal-target ledger, tensor NameCert, and GroundCompiler audit-map consumers remains a displayed NameCert route rather than an assertion that explanatory prose itself supplies theorem evidence.

Proof. Start with the source and classifier rows. They bind the explanation to a displayed behavior or object and to a concrete classifier before the five-field deficit vector is read.

The evidence row E and gap row G then expose the material and deficit coordinates. A downstream consumer can inspect these rows, but it cannot treat an open ledger coordinate as resolved until the certification-route row R displays the corresponding closure or refusal.

Finally read H, K, P, N. The provenance row may cite machine-readable audit, registry export, philosophy formal-target, tensor NameCert, or GroundCompiler audit-map consumers, but those siblings are reached only through the displayed route. Thus the schematic is concretized as a finite certified explanation packet and not as a prose-only explanation channel. □

The Lean follow-up must provide explicit Nontrivial CertifiedExplanationSchematic$^{\uparrow}$ and FieldFaithful CertifiedExp instances. The field-faithfulness witness must preserve $S, C, E, G, R, H, K, P, N$ as distinct rows before any TasteGate result can be cited by a checked paper marker.

Closure status for NameCert$_{\text{CertifiedExplanationSchematic}^{\uparrow}}$ ·

Constructive story: A CertifiedExplanationSchematic$^{\uparrow}$ packet is generated from source, classifier, evidence, ledger-gap, certification-route, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate(NameCert$_{\text{CertifiedExplanationSchematic}^{\uparrow}}$, seed)

Scope closed: Definition 6303.1, Definition 6303.2, and Definition 6303.3 seed the finite certified explanation route for the five-field schematic.

Formal verification: FormalStatus(NameCert$_{\text{CertifiedExplanationSchematic}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, TasteGate acceptance, Nontriviality, FieldFaithfulness, hidden theorem evidence, prose-only explanation, unchecked registry export, or detached ledger closure.

Upgrade path: Obligation closure requires checked carrier admission, evidence readback, five-field ledger accounting, certification-route exactness, sibling handoff, transport, replay, provenance, and local naming for BEDC.Derived.CertifiedExplanationSchematic

Chapter 6304

A Concrete Naming Certificate for $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ packet records the finite envelope surface by which an upper-semicontinuous majorant is read through located real windows before any Real-facing consumer inspects a seal. It sits beside the lower-semicontinuous packet of chapter 1536, the one-sided $\mathsf{UpperReal}^{\uparrow}$ ledger, the located endpoint discipline of **??**, the finite $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the regular rational readback of chapter 301, and the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet names a displayed located upper envelope; it is not a host limsup, quotient equality of real names, a choice-selected representative, or an ambient completeness principle.

Definition 6304.1 (Located upper-semicontinuous envelope carrier)**.** *A* $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, U, A, W, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *L is the source* $\mathsf{LowerSemicontinuous}^{\uparrow}$ *row whose lower-side sublevel and epigraph windows are being enveloped.*
2. *U is the* $\mathsf{UpperReal}^{\uparrow}$ *majorant ledger, recording the upper real cut rows and their finite comparison requests.*
3. *A is the located-approximation row, requiring every accepted upper bound request to be accompanied by a displayed* $\mathsf{LocatedReal}^{\uparrow}$ *comparison window.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *row listing the finite windows inspected for the source, the upper ledger, and the located approximation.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the windows have been displayed.*
6. *Q is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for the envelope value, visible only after L, U, A, W, R have been read.*
7. *H records componentwise* hsame *transport for the source, upper-ledger, located-approximation, window, readback, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from any envelope request through the same rows.*

9. *P is the* Pkg *provenance over* $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$, $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}}$ *row.*

The carrier has no coordinate for a host limsup, a completed supremum operator, quotient equality of real names, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6304.2 (Located upper-semicontinuous envelope obligation surface)**.** *For every accepted* $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ *carrier* $E = (L, U, A, W, R, Q, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}}$ *surface consists exactly of carrier admission, lower-semicontinuous source exposure, upper-real majorant exactness, located-comparison routing, finite window readback, regular rational repacking, terminal Real sealing, componentwise* hsame *transport, displayed* Cont *replay, provenance, and local naming for the same finite packet.*

Proof. Project the carrier of Definition 6304.1. Its analytic rows are exactly L, U, A, W, R, Q, and its structural rows are exactly H, C, P, N.

The source and majorant obligations are read through L and U. The row L exposes only the displayed $\mathsf{LowerSemicontinuous}^{\uparrow}$ surface, while U records the one-sided $\mathsf{UpperReal}^{\uparrow}$ comparison ledger used by the envelope.

The located and Real-facing obligations are A, W, R, Q. They force every upper bound request through a $\mathsf{LocatedReal}^{\uparrow}$ comparison window, a finite $\mathsf{StreamName}^{\uparrow}$ inspection, a $\mathsf{RegSeqRat}^{\uparrow}$ readback, and only then the terminal $\mathsf{Real}^{\uparrow}$ seal.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since the refused host objects are absent from the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6304.3 (Located upper-semicontinuous envelope Real seal boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ *packet reads the terminal envelope seal only through*

$$\mathsf{LowerSemicontinuous}^{\uparrow}, \mathsf{UpperReal}^{\uparrow}, \mathsf{LocatedReal}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus the envelope may expose a located upper-semicontinuous Real-facing row, but it does not export a host limsup, a quotient equality principle, a choice-selected representative, or ambient Real completeness.

Proof. Begin with a consumer of the terminal row Q. By the carrier definition, there is no route to Q before the lower-semicontinuous source row L, the upper-real majorant ledger U, and the located comparison row A have been displayed.

The finite observation route then passes through the window row W and the regular rational readback row R. These rows make the envelope request a displayed finite packet rather than a semantic appeal to a completed upper envelope.

The transport, replay, provenance, and local naming rows preserve that same route. Hence every Real-seal read is bounded by the displayed $L, U, A, W, R, Q, H, C, P, N$ coordinates and cannot bypass them through a host limsup, quotient equality, choice, or ambient completeness. □

Falsifiable prediction: Every $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ Real-facing read factors through displayed $\mathsf{LowerSemicontinuous}^{\uparrow}$, $\mathsf{UpperReal}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and local NameCert rows.

Independence witness: lowersemicontinuous, locatedreal, streamname, regseqrat, real: LocatedUpperSemicontin
is not bijective to any listed sibling because it jointly carries a lower-semicontinuous source, an

upper-real majorant ledger, located comparison windows, regular rational readback, a Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6304.1, Definition 6304.2, and Definition 6304.3 seed the finite located upper-envelope route from lower-semicontinuous source data through upper-real and located-real comparison windows to a terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host limsup construction, quotient equality of real names, selected envelope representatives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, upper-real majorant exactness, located-comparison routing, finite-window readback, regular rational repacking, Real seal descent, transport, replay, provenance, and local naming for BEDC.Derived.LocatedUpperSemicontinuousEnvelopeUp.

Constructive story: A $\mathsf{LocatedUpperSemicontinuousEnvelope}^{\uparrow}$ packet is generated from a LowerSemicontinuous source row, an UpperReal majorant ledger, LocatedReal comparison windows, StreamName finite observations, RegSeqRat readback, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6305

A Concrete Naming Certificate for NormalizationByEvaluation$^{\uparrow}$

Following Berger and Schwichtenberg, normalization by evaluation is read as a finite semantic-evaluation and reification packet for typed terms. The NormalizationByEvaluation$^{\uparrow}$ carrier sits beside the lambda-calculus certificate of chapter 200, the first-order syntax interface of chapter 197, and the candidate-mediated MetaCIC normalization audit of chapter 5925. It names only the displayed typed-term syntax, environment, semantic-value, neutral-reflection, readback, evaluation-ledger, transport, replay, provenance, and local NameCert rows needed for a readback consumer. It does not assert a host normalization function, unrestricted subject reduction, evidence-free MetaCIC strong normalization, quotient syntax equality, or a completed semantic model.

Definition 6305.1 (NormalizationByEvaluation carrier)**.** *A* NormalizationByEvaluation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (T, G, V, NF, R, Ev, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed-term row, carrying the source term, its type, and the binder context visible to the readback consumer.*
2. *G is the environment row, recording finite semantic assignments for the free variables and context entries named by T.*
3. *V is the semantic-value row obtained by evaluating the displayed term under G through the packet ledger.*
4. *NF is the neutral-reflection row, separating variables and neutral applications that are reflected into semantic values from already reified normal forms.*
5. *R is the normal-form readback row, turning accepted semantic values into displayed beta-normal eta-long syntax when the packet admits such readback.*
6. *Ev is the evaluation ledger, recording each semantic-evaluation step, reification step, substitution replay, and refusal boundary used by the carrier.*
7. *H records componentwise* hsame *transport for typed terms, environments, semantic values, neutral-reflection rows, readback rows, evaluation ledgers, replay, provenance, and local naming rows.*

8. *C records displayed* Cont *replay from a requested typed-term read through T, G, V, NF, R, Ev.*
9. *P is the* Pkg *provenance over* $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NormalizationByEvaluation}^{\uparrow}}$ *row.*

The classifier accepts the packet only when semantic evaluation and readback are both recorded as finite rows. No coordinate may be a host-language evaluator, a global normalization oracle, a quotient of syntax trees, or an unlogged semantic model.

Theorem 6305.2 (NormalizationByEvaluation NameCert obligations)**.** *For every accepted* NormalizationByEvaluat *carrier $E = (T, G, V, NF, R, Ev, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NormalizationByEvaluation}^{\uparrow}}$ *surface consists of the following six obligation rows:*

1. *carrier admission for the typed-term row, environment row, semantic-value row, neutral-reflection row, readback row, evaluation ledger, transport, replay, provenance, and local naming rows;*
2. *semantic evaluation, requiring V to be obtained from the displayed term and environment through the finite ledger Ev rather than a hidden evaluator;*
3. *neutral reflection, requiring reflected variables and neutral applications in NF to remain tied to the binder context and semantic assignments named by T, G;*
4. *reification and readback, requiring R to read only accepted semantic values and neutral reflections before producing a displayed normal form;*
5. *substitution stability, requiring context extension and substitution replay to pass through the same T, G, V, NF, R, Ev route under componentwise* hsame *transport;*
6. *no host normalization oracle, excluding external evaluators, quotient syntax equality, completed semantic models, unrestricted subject reduction, and evidence-free MetaCIC strong normalization.*

No accepted NameCert row is stored outside these finite coordinates.

Proof. Carrier admission is projection from Definition 6305.1. The tuple lists exactly the typed-term, environment, semantic-value, neutral-reflection, readback, ledger, transport, replay, provenance, and local-name rows.

Semantic evaluation is confined to the ledger Ev. A consumer may inspect V only after the displayed term T and environment G have been replayed, so the packet cannot consult a separate evaluator.

Neutral reflection and readback are then separated by NF and R. Reflection keeps variables and neutral applications attached to the visible context, while readback consumes accepted semantic values and neutral rows before producing normal-form syntax.

Substitution stability is recorded by the same route. Componentwise transport H and continuation replay C move the typed-term context, environment assignments, semantic values, neutral rows, readback rows, and ledger rows together, so substitution cannot bypass the displayed evaluation packet.

The provenance and local naming rows restrict the sibling imports to the lambda-calculus, first-order syntax, and candidate-mediated MetaCIC normalization surfaces cited above. Since no carrier coordinate stores a host evaluator, quotient syntax equality, completed semantic model, or evidence-free strong-normalization theorem, the listed rows are the whole NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{NormalizationByEvaluation}^{\uparrow}$ readback factors through the typed-term row, environment row, semantic-value row, neutral-reflection row, normal-form readback row, finite evaluation ledger, componentwise transport, continuation replay, provenance, and local NameCert before any consumer may cite a normal form.

Independence witness: lambdacalc, firstorder, metacic-candidate-strong-normalization: NormalizationByEvaluat is not bijective to any listed sibling because it jointly carries semantic evaluation, neutral reflection, readback, substitution replay, and no-oracle refusal rows. LambdaCalcUp carries beta syntax without the semantic-value/readback ledger; FirstOrderUp carries formula syntax without neutral reflection; MetaCICCandidateStrongNormalizationUp carries candidate evidence without an NBE readback packet.

Closure status for $\mathsf{NameCert}_{\mathsf{NormalizationByEvaluation}^{\uparrow}}$.

Constructive story: A $\mathsf{NormalizationByEvaluation}^{\uparrow}$ packet is generated from a typed-term row, environment row, semantic-value row, neutral-reflection row, normal-form readback row, finite evaluation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NormalizationByEvaluation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6305.1 and Definition 6305.2 seed the finite semantic-evaluation and readback packet for typed terms.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NormalizationByEvaluation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted MetaCIC normalization, subject reduction, global confluence, quotient syntax equality, host evaluator soundness, completed semantic models, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, semantic evaluation, neutral reflection, readback, substitution stability, and no-oracle refusal for BEDC.Derived.NormalizationByEvaluationUp.

Chapter 6306

A Concrete Naming Certificate for $\mathsf{NeutralReifier}^{\uparrow}$

The $\mathsf{NeutralReifier}^{\uparrow}$ packet isolates the neutral-reflection and semantic readback component used by normalization-by-evaluation. It realizes the MetaCIC normalization audit route of chapter 9383 as finite neutral, semantic-value, readback, transport, replay, provenance, and local NameCert rows, while sitting beside the lambda-calculus certificate of chapter 200, the candidate-mediated strong-normalization packet of chapter 5925, and the normalization-by-evaluation packet of chapter 6305. The packet names the reifier component only; it does not assert global normalization, unrestricted MetaCIC subject reduction, host evaluator soundness, or quotient equality of terms.

Definition 6306.1 (NeutralReifier carrier)**.** *A* $\mathsf{NeutralReifier}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (T, Z, V, U, R, A, H, C, P, N)$$

with the following displayed rows:

1. *T is the typed neutral-term row, recording the context, type, and neutral spine presented to the reifier.*
2. *Z is the neutral-reflection row, naming variables, eliminator spines, and application heads that may be reflected into semantic values.*
3. *V is the semantic-value row produced by reflection or by a prior accepted evaluation packet.*
4. *U is the universe or type-level guard row controlling which readback shape is allowed for the displayed semantic value.*
5. *R is the readback row that produces a normal neutral or normal form from Z, V, U.*
6. *A is the audit-route row binding the packet to the candidate-mediated MetaCIC normalization route of chapter 9383.*
7. *H records componentwise* hsame *transport for neutral terms, reflection rows, semantic values, type guards, readback rows, audit rows, replay, provenance, and local names.*
8. *C records displayed* Cont *replay from a requested reification read through T, Z, V, U, R, A.*
9. *P is the* Pkg *provenance over* $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{MetaCICCandidateStrongNormalization}^{\uparrow}$, $\mathsf{NormalizationByEvaluation}^{\uparrow}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{NeutralReifier}^{\uparrow}}$ *row.*

The classifier accepts the packet only when neutral reflection and readback remain finite displayed rows tied to the audit route. It has no slot for a host normalizer, a semantic equality oracle, a quotient of terms, or an unconditional strong-normalization theorem.

Theorem 6306.2 (NeutralReifier NameCert obligations)**.** *For every accepted* $\mathsf{NeutralReifier}^{\uparrow}$ *carrier* $Q = (T, Z, V, U, R, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{NeutralReifier}^{\uparrow}}$ *surface consists of the following six rows:*

1. *carrier admission for neutral terms, neutral reflection, semantic values, type guards, readback, audit-route, transport, replay, provenance, and local naming;*
2. *neutral reflection, requiring every reflected variable or eliminator spine in* Z *to be tied to the displayed context and typed neutral term row* T*;*
3. *semantic value admission, requiring* V *to arise from reflection or from an accepted evaluation packet named in the provenance row;*
4. *guarded readback, requiring* R *to read* Z, V, U *before producing a normal neutral or normal form;*
5. *transport and replay stability, requiring componentwise* hsame *and displayed* Cont *to preserve the same* T, Z, V, U, R, A *route;*
6. *audit nonescape, excluding host normalization oracles, semantic equality oracles, quotient syntax equality, unrestricted subject reduction, and evidence-free strong normalization.*

No consumer of the packet receives any stronger MetaCIC normalization claim than these rows display.

Proof. Carrier admission is immediate from Definition 6306.1. The tuple exposes neutral terms, reflection, semantic values, type guards, readback, the audit route, transport, replay, provenance, and local naming as the only rows.

Neutral reflection is tied to T and Z. Variables and eliminator spines are reflected only through the displayed context and typed neutral surface, so the packet cannot create a semantic value without a visible neutral source.

Semantic admission and guarded readback then pass through V, U, R. The value row is either reflected from Z or imported through an accepted evaluation packet named by P, and the readback row must consume the displayed type guard before producing syntax.

Transport and replay stability use H and C. These rows move the same neutral, semantic-value, type-guard, readback, and audit-route coordinates together, preserving the route into chapter 9383.

The audit row A, provenance P, and local name N keep the packet at the reifier component boundary. Since the carrier has no coordinate for a host normalizer, semantic equality oracle, quotient syntax equality, unrestricted subject reduction, or evidence-free strong normalization, the obligation surface is exactly the six rows listed. □

Falsifiable prediction: Every accepted $\mathsf{NeutralReifier}^{\uparrow}$ readback factors through the neutral-term row, neutral-reflection row, semantic-value row, type guard, readback row, MetaCIC audit-route row, componentwise transport, continuation replay, provenance, and local $\mathsf{NameCert}$ before any consumer may cite a normal neutral or normal form.

Independence witness: lambdacalc, metacic-candidate-strong-normalization, normalization-by-evaluation: $\mathsf{NeutralReifier}^{\uparrow}$ is not bijective to any listed sibling because it isolates neutral reflection, guarded readback, and MetaCIC audit-route replay as one packet. LambdaCalcUp has beta syntax without semantic reification; MetaCICCandidateStrongNormalizationUp has candidate evidence

without reifier rows; NormalizationByEvaluationUp includes full evaluation and substitution stability beyond the neutral reifier component.

Closure status for $\mathsf{NameCert}_{\mathsf{NeutralReifier}^{\uparrow}}$.

Constructive story: A $\mathsf{NeutralReifier}^{\uparrow}$ packet is generated from a typed neutral-term row, neutral-reflection row, semantic-value row, type guard, readback row, MetaCIC audit-route row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NeutralReifier}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6306.1 and Definition 6306.2 seed the finite neutral-reflection and readback component for the MetaCIC normalization audit route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NeutralReifier}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted MetaCIC normalization, subject reduction, global confluence, semantic equality, quotient syntax equality, host evaluator soundness, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, neutral reflection, semantic-value admission, guarded readback, transport stability, and audit nonescape for BEDC.Derived.NeutralReifierUp.

Chapter 6307

A Concrete Naming Certificate for $\mathsf{CayleyHamilton}^{\uparrow}$

Following Cayley and Hamilton, the $\mathsf{CayleyHamilton}^{\uparrow}$ packet records the characteristic-polynomial annihilation law as a finite matrix-polynomial certificate. It sits over the matrix carrier of chapter 49, the polynomial carrier of chapter 50, the scalar $\mathsf{Ring}^{\uparrow}$ structure of chapter 43, and the coefficient $\mathsf{CommRing}^{\uparrow}$ structure of chapter 44. The packet exposes a square matrix, its finite coefficient ledger, the characteristic-polynomial row, the matrix-polynomial evaluation row, and a zero-matrix annihilation witness before any spectral or determinant-facing consumer can read the claim.

The certificate is a NameCert boundary for the finite algebraic identity. It does not choose eigenvalues, diagonalize a matrix, assert field algebraic closure, or import host determinant semantics.

Definition 6307.1 (Cayley-Hamilton carrier)**.** *A* $\mathsf{CayleyHamilton}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (R, n, A, \chi, E, Z, H, K, P, N)$$

with the following displayed rows:

1. *R is the scalar* $\mathsf{CommRing}^{\uparrow}$ *coefficient source, read together with its underlying* $\mathsf{Ring}^{\uparrow}$ *addition, multiplication, zero, one, and distributivity rows;*
2. *n is the finite square-size row for the matrix index carrier;*
3. *A is the* $\mathsf{Mat}^{\uparrow}$ *row for a carried $n \times n$ matrix over R;*
4. *χ is the* $\mathsf{Poly}^{\uparrow}$ *row naming the characteristic polynomial of A as a finite coefficient ledger over R;*
5. *E is the matrix-polynomial evaluation row obtained by substituting A into χ using the finite matrix multiplication and addition ledgers;*
6. *Z is the zero-matrix comparison row certifying that the evaluated matrix-polynomial row is the displayed zero matrix;*
7. *H records componentwise* hsame *transport for R, n, A, χ, E, Z;*
8. *K records the displayed* Cont *replay route from the scalar and matrix rows through the polynomial evaluation row to the zero-matrix witness;*

9. *P is the* Pkg *provenance over* $\mathsf{Mat}^{\uparrow}$, $\mathsf{Poly}^{\uparrow}$, $\mathsf{Ring}^{\uparrow}$, $\mathsf{CommRing}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CayleyHamilton}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose rows are the displayed coefficient source, square-size row, matrix row, characteristic-polynomial row, matrix-polynomial evaluation row, zero-matrix comparison, transport, replay, provenance, and local naming data. It has no coordinate for an eigenvalue, diagonal form, ambient algebraic closure, determinant oracle, or host polynomial identity.

Lean4 ✓ `BEDC.Derived.CayleyHamiltonUp.CayleyHamiltonTasteGate single carrier alignment`

Theorem 6307.2 (Cayley-Hamilton NameCert obligations)**.** *For every accepted* $\mathsf{CayleyHamilton}^{\uparrow}$ *carrier* $C = (R, n, A, \chi, E, Z, H, K, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CayleyHamilton}^{\uparrow}}$ *surface consists exactly of carrier admission for the scalar* $\mathsf{CommRing}^{\uparrow}$ *and* $\mathsf{Ring}^{\uparrow}$ *rows, the finite square* $\mathsf{Mat}^{\uparrow}$ *row, the finite* $\mathsf{Poly}^{\uparrow}$ *characteristic-polynomial row, the displayed matrix-polynomial evaluation row, the zero-matrix comparison row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The annihilation obligation requires E to evaluate the same χ at the same A over the same scalar structure, and requires Z to compare that evaluation with the displayed zero matrix. No accepted packet exports eigenvalues, diagonalization, algebraic closure, a determinant oracle, host matrix equality, or a spectral decomposition.*

Proof. Project the carrier of Definition 6307.1. The accepted packet lists exactly $R, n, A, \chi, E, Z, H, K, P, N$, so every algebraic read starts from the displayed coefficient, matrix, polynomial, evaluation, and zero-comparison rows.

The evaluation obligation is finite. The row χ is a $\mathsf{Poly}^{\uparrow}$ coefficient ledger over the same scalar source R, and the row E is obtained by the displayed finite matrix addition and multiplication route over the $\mathsf{Mat}^{\uparrow}$ carrier A.

The annihilation witness is the comparison row Z. It relates the matrix-polynomial evaluation row to the zero matrix inside the same square-size and scalar structure, and transport through H preserves those displayed rows componentwise.

Finally, replay and provenance through K, P, N expose only the NameCert route just described. Since the carrier has no eigenvalue, diagonalization, algebraic-closure, determinant-oracle, host-equality, or spectral-decomposition coordinate, the listed obligations exhaust the certificate surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CayleyHamilton}^{\uparrow}}$.

Constructive story: A $\mathsf{CayleyHamilton}^{\uparrow}$ packet is generated from a scalar CommRingUp and RingUp source, a finite square MatrixUp row, a PolynomialUp characteristic-polynomial ledger, a matrix-polynomial evaluation row, a zero-matrix comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CayleyHamilton}^{\uparrow}}$, seed)

Scope closed: Definition 6307.1 and Definition 6307.2 seed the finite matrix-polynomial certificate over displayed scalar, matrix, polynomial, evaluation, zero-comparison, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CayleyHamilton}^{\uparrow}}$, theoremChecked)

Lean target: `BEDC.Derived.CayleyHamiltonUp.CayleyHamiltonTasteGate\_single\_carrier\_alignment`

Bridge status: none

Not claimed: This chapter does not close eigenvalue construction, diagonalization, algebraic closure, determinant semantics, spectral decomposition, Lean verification, or bridge checking for $\mathsf{CayleyHamilton}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, characteristic-polynomial evaluation exactness, zero-matrix comparison stability, transport, replay, provenance, and local naming for BEDC.Derived.CayleyHamiltonUp.

Chapter 6308

A Concrete Naming Certificate for $\mathsf{StandardBorelSpace}^{\uparrow}$

Following Kechris' descriptive set theory, the $\mathsf{StandardBorelSpace}^{\uparrow}$ packet records a Polish presentation together with a generated Borel event classifier. It sits over the Polish presentation surface of chapter 1490, the Borel-code classifier of chapter 3384, the terminal real-name seal of chapter 38, the finite stream interface of chapter 126, and the regular rational readback surface of chapter 301. The packet is a measurable-name interface for probability and distribution consumers; it is not a host measurable-space import, quotient completion, selected point representation, or ambient sigma-algebra oracle.

Definition 6308.1 (Standard Borel space carrier)**.** *A* $\mathsf{StandardBorelSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (P, B, R, W, G, E, H, C, Q, N)$$

with the following displayed rows:

1. *P is the* $\mathsf{PolishSpace}^{\uparrow}$ *presentation row, including the displayed complete separable metric data and dense naming discipline consumed by the packet.*
2. *B is the* $\mathsf{BorelSet}^{\uparrow}$ *code row generated from the topology carried by P.*
3. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row used for real-valued observable boundaries.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row used to read presented points and event probes.*
5. *G is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row used by real-valued event thresholds.*
6. *E is the event-classifier row assigning each displayed Borel code its finite membership-probe schedule over W, G, R.*
7. *H records componentwise* hsame *transport for $P, B, R, W, G, E, C, Q, N$.*
8. *C records displayed* Cont *replay from Polish presentation data through Borel-code generation to event-classifier reads.*
9. *Q is the* Pkg *provenance over* $\mathsf{StandardBorelSpace}^{\uparrow}$, $\mathsf{PolishSpace}^{\uparrow}$, $\mathsf{BorelSet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{StandardBorelSpace}^{\uparrow}}$ *row.*

The classifier accepts exactly packets whose Borel event reads are generated from the displayed Polish presentation and whose real-valued observable thresholds pass through the stream, regular-rational, and real-seal rows. It has no coordinate for a host measurable space, unrestricted powerset sigma-algebra, quotient presentation, selected representative, or ambient probability measure.

Lean4 (statement) `BEDC.Derived.StandardBorelSpaceUp`

Theorem 6308.2 (Standard Borel space NameCert obligations)**.** *For every accepted* $\mathsf{StandardBorelSpace}^{\uparrow}$ *carrier*

$$S = (P, B, R, W, G, E, H, C, Q, N),$$

the local $\mathsf{NameCert}_{\mathsf{StandardBorelSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the* $\mathsf{PolishSpace}^{\uparrow}$ *presentation row, the generated* $\mathsf{BorelSet}^{\uparrow}$ *code row, the* $\mathsf{Real}^{\uparrow}$ *observable boundary, the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, the event-classifier row, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. The event classifier may accept only Borel codes generated from* P *and threshold reads routed through* W, G, R*. It exports no host measurable space, unrestricted sigma-algebra, selected point representation, quotient completion, probability measure, or ambient measurability oracle.*

Proof. Project the carrier of Definition 6308.1. The presentation and classifier data are exactly P, B, E, while W, G, R are the only rows available for point windows and real-valued threshold reads.

The Borel-code route is generated from P. The row B records Borel codes over the topology displayed by the $\mathsf{PolishSpace}^{\uparrow}$ presentation, and E may classify only those codes against the finite window and regular-rational rows.

The observable route is finite. A real-valued threshold read must pass through W for stream windows, G for regular rational readback, and R for the terminal real-name seal before the classifier row can be consumed.

Finally H, C, Q, N transport, replay, source, and name the same displayed packet. Since the carrier has no coordinate for a host measurable space, powerset sigma-algebra, selected representative, quotient completion, probability measure, or ambient measurability oracle, the listed rows exhaust the NameCert obligations. □

Theorem 6308.3 (Standard Borel Polish-Borel handoff)**.** *Every accepted* $\mathsf{StandardBorelSpace}^{\uparrow}$ *event read factors through the displayed route*

$$\mathsf{PolishSpace}^{\uparrow} \longmapsto \mathsf{BorelSet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Thus a probability, distribution, or measurable-kernel consumer may inspect only generated Borel event codes and finite real-threshold readbacks; it may not replace the packet by a host measurable space, a selected Polish representative, quotient real equality, an unrestricted sigma-algebra, or an ambient probability object.

Proof. Start with the carrier rows of Definition 6308.1. The only Borel row is B, and it is tied to the Polish presentation row P rather than to an external measurable-space structure.

Apply Definition 6308.2. It binds each event-classifier read to the generated Borel code and then routes real-valued probes through W, G, R.

The structural rows H, C, Q, N preserve that route under transport, replay, provenance, and local naming. A downstream consumer therefore receives a generated Borel classifier over the displayed Polish presentation, not an imported measurable-space object or an unrestricted sigma-algebra. □

Closure status for $\mathsf{NameCert}_{\mathsf{StandardBorelSpace}^{\uparrow}}$.

Constructive story: A $\mathsf{StandardBorelSpace}^{\uparrow}$ packet is generated from a PolishSpace presentation, a BorelSet code row, a Real observable boundary, StreamName windows, RegSeqRat threshold readback, an event-classifier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{StandardBorelSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6308.1, Definition 6308.2, and Definition 6308.3 seed the generated Borel event-classifier interface over displayed Polish, Borel-code, stream, regular-rational, and Real rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{StandardBorelSpace}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close host measurable spaces, unrestricted sigma-algebras, quotient presentations, selected point representatives, probability measures, Lean checking, or bridge checking for $\mathsf{StandardBorelSpace}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, generated Borel-code stability, event-classifier stability, real-threshold routing, transport, replay, provenance, and local naming for BEDC.Derived.StandardBorelSpaceUp.

Chapter 6309

A Concrete Naming Certificate for $\mathsf{FiniteFourierEnergy}^{\uparrow}$

Following Parseval and Plancherel's finite-energy route in Fourier analysis, the $\mathsf{FiniteFourierEnergy}^{\uparrow}$ packet records the finite energy certificate that can be read between the Fourier observation surface of chapter 124 and the Parseval identity surface of chapter 1940. It consumes circle source rows, inner-product pairing rows, integral energy rows, rational readback rows, dyadic tolerance rows, and terminal Real seals without claiming an ambient Plancherel theorem, inversion theorem, completed Hilbert space, or quotient coefficient object.

Definition 6309.1 (Finite Fourier energy carrier)**.** *A* $\mathsf{FiniteFourierEnergy}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (F, V, S, I, J, Q, D, R, H, C, P, N)$$

with the following displayed rows:

1. *F is the* $\mathsf{Fourier}^{\uparrow}$ *finite observation row, listing the sampled modes, coefficient probes, transform-window probes, and truncation bounds under inspection.*
2. *V is the* $\mathsf{Parseval}^{\uparrow}$ *finite identity row, recording the coefficient-energy ledger and matching integral-energy ledger already displayed by the Parseval packet.*
3. *S is the circle source row, exposing only the finite phase samples and source classifier data consumed by the Fourier probes.*
4. *I is the* $\mathsf{InnerProduct}^{\uparrow}$ *pairing row for the coefficient and source rows.*
5. *J is the* $\mathsf{Integral}^{\uparrow}$ *energy row that reads the same finite source window as an integral-energy observation.*
6. *Q is the rational-readback row, using* $\mathsf{RegSeqRat}^{\uparrow}$ *data for finite sums, pairings, integral observations, and displayed energy comparisons.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row for summation, pairing, integral readback, Parseval comparison, and downstream energy-window reuse.*
8. *R is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the Fourier, Parseval, circle, inner-product, integral, rational-readback, and dyadic-tolerance rows have been displayed.*
9. *H stores componentwise* hsame *transport for all displayed rows.*

10. *C records the displayed* Cont *replay route from Fourier coefficient windows through Parseval energy comparison and into the Real seal.*
11. *P is the* Pkg *provenance over* $\mathsf{FiniteFourierEnergy}^{\uparrow}$, $\mathsf{Fourier}^{\uparrow}$, $\mathsf{Parseval}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{InnerProduct}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteFourierEnergy}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $F, V, S, I, J, Q, D, R, H, C, P, N$. It has no coordinate for ambient Plancherel, Fourier inversion, completed Hilbert-space norm equality, host function equality, selected representatives, or quotient coefficient classes.

Theorem 6309.2 (Finite Fourier energy NameCert obligations)**.** *For every accepted* $\mathsf{FiniteFourierEnergy}^{\uparrow}$ *carrier $E = (F, V, S, I, J, Q, D, R, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteFourierEnergy}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Fourier, Parseval, circle, inner-product, integral, rational-readback, dyadic-tolerance, Real-seal, transport, replay, provenance, and local naming rows; coefficient-window exactness through F; Parseval-row attachment through V; source and pairing exposure through S, I; integral-energy exposure through J; rational and dyadic accounting through Q, D; Real-seal non-escape through R; and componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

Proof. Project the carrier in Definition 6309.1. The finite inventory is exactly $F, V, S, I, J, Q, D, R, H, C, P, N$, so the NameCert obligations cannot cite a hidden analytic coordinate.

The energy-facing rows are read in order. The Fourier row F supplies the finite coefficient and transform windows, the Parseval row V supplies the finite energy identity surface, the circle row S supplies the finite source samples, and the pairing and integral rows I, J expose the two sides of the energy comparison.

The numerical endpoint is also finite. Rational readback Q and dyadic tolerance D record the visible arithmetic data before the Real seal R is reached.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier has no Plancherel, inversion, completed norm, host equality, representative-selection, or quotient coordinate, none of those conclusions can be projected from the packet. □

Theorem 6309.3 (Finite Fourier energy Parseval route)**.** *Every consumer read of a* $\mathsf{FiniteFourierEnergy}^{\uparrow}$ *packet factors through the finite Fourier observation surface of chapter 124 and the finite Parseval energy-identity surface of chapter 1940. The route may expose Fourier coefficient windows, Parseval finite-energy ledgers, circle samples, inner-product pairings, integral observations, rational readback, dyadic tolerances, and Real seals, but it exports no ambient Plancherel theorem, inversion theorem, completed Hilbert-space norm equality, host function equality, selected representative, or quotient coefficient class.*

Lean4 ✓ `BEDC.Derived.FiniteFourierEnergyUp.FiniteFourierEnergy parseval route`

Proof. Start with the rows F and V of Definition 6309.1. The row F is explicitly the finite Fourier observation row, while V is explicitly the finite Parseval identity row.

Attach the source, pairing, and energy rows S, I, J. They keep the route on finite circle samples, inner-product witnesses, and integral observations rather than on a completed ambient transform.

Read Q, D, R next. Rational readback and dyadic tolerance data are visible before the Real seal, so the terminal row is only the displayed finite energy comparison.

The non-energy rows H, C, P, N preserve this same factorization under transport, replay, provenance, and naming. Therefore every consumer route factors through the Fourier and Parseval siblings and stops before the excluded Plancherel, inversion, completion, host-equality, representative-selection, or quotient conclusions. □

Falsifiable prediction: Every $\mathsf{FiniteFourierEnergy}^{\uparrow}$ energy read factors through Fourier finite observations, Parseval finite-energy ledgers, circle source samples, InnerProduct pairings, Integral energy observations, RegSeqRat rational readback, DyadicRatCore tolerances, Real seals, hsame transport, Cont replay, Pkg provenance, and local naming.

Independence witness: fourier, parseval, s1, innerproduct, integral, regseqrat, dyadicratcore, real: $\mathsf{FiniteFourierEnergy}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the finite Fourier observation row, finite Parseval identity row, source, pairing, integral, rational-readback, dyadic-tolerance, Real-seal, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteFourierEnergy}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteFourierEnergy}^{\uparrow}$ packet is generated from finite Fourier observations, a finite Parseval identity row, circle source samples, inner-product pairings, integral energy observations, rational readback, dyadic tolerance accounting, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteFourierEnergy}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6309.1, Definition 6309.2, and Definition 6309.3 seed the finite Fourier energy certificate consumed by downstream Plancherel-facing work.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteFourierEnergy}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close ambient Plancherel, Fourier inversion, convergence closure, completed Hilbert-space norm equality, host function equality, selected representatives, quotient coefficient classes, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, Fourier-window exactness, Parseval-row attachment, circle-source exposure, inner-product and integral-energy accounting, rational readback, dyadic tolerance accounting, Real seal non-escape, transport, replay, provenance, local naming, and TasteGate field faithfulness for BEDC.Derived.FiniteFourierEnergyUp.

Chapter 6310

A Concrete Naming Certificate for $\mathsf{ProximitySpace}^{\uparrow}$

Following Efremovic and Naimpally–Warrack on proximity spaces, the $\mathsf{ProximitySpace}^{\uparrow}$ packet records nearness as finite NameCert data rather than as a quotient of filters or a selected completion object. It sits between the separated uniform boundary of chapter 1876, the completion route of chapter 3560, and the complete uniform-space seal of chapter 1588. The packet supplies a conservative L10 nearness interface for Real and Completion consumers; it does not assert compactness, choose completion points, or import choice principles.

Definition 6310.1 (Proximity-space carrier)**.** *A* $\mathsf{ProximitySpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (N, D, S, K, H, C, G, L)$$

with the following displayed rows:

1. *N is the nearness row, recording finite positive proximity observations between carried finite marks and the displayed neighborhood tests that witness them.*
2. *D is the diagonal and apartness row, requiring every accepted self-nearness read to pass through the displayed diagonal evidence and every rejected nearness read to expose its finite apartness witness.*
3. *S is the symmetry row, recording that a displayed nearness read may be replayed in the opposite order without introducing a host equality of points.*
4. *K is the composition row, recording the finite interpolation evidence used when two displayed nearness reads are composed through an admitted middle observation.*
5. *H records componentwise* hsame *transport for nearness, diagonal and apartness, symmetry, composition, continuation, provenance, and local naming rows.*
6. *C records displayed* Cont *replay of the route*

$$N, D, S, K \longmapsto H.$$

7. *G is the* Pkg *provenance over* $\mathsf{ProximitySpace}^{\uparrow}$, $\mathsf{SeparatedUniformSpace}^{\uparrow}$, $\mathsf{UniformSpaceCompletion}^{\uparrow}$, $\mathsf{CompleteUniformSpace}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *L is the local* $\mathsf{NameCert}_{\mathsf{ProximitySpace}^{\uparrow}}$ *row.*

The classifier compares accepted proximity packets only through N, D, S, K, H, C, G, L. It has no coordinate for quotient filters, selected completions, open-cover compactness, countable choice, host equality of points, or ambient completeness.

Theorem 6310.2 (Proximity-space NameCert obligations)**.** *For every accepted* $\mathsf{ProximitySpace}^{\uparrow}$ *carrier* $P = (N, D, S, K, H, C, G, L)$*, the local* $\mathsf{NameCert}_{\mathsf{ProximitySpace}^{\uparrow}}$ *surface consists exactly of carrier admission, finite nearness exposure, diagonal and apartness exactness, symmetry replay, finite composition, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every L10 completion or uniformity consumer must read the displayed route*

$$N, D, S, K \longmapsto H, C, G, L$$

before using proximity as evidence for a separated or complete uniform-space boundary.

Proof. Project the carrier of Definition 6310.1. The nearness row N is the only row from which a consumer can read positive proximity observations, and the diagonal and apartness row D is the only row from which it can read self-nearness or rejected-nearness evidence.

The symmetry row S and the composition row K are finite replay rules. They transport a displayed nearness observation across order reversal or through an admitted middle observation, but they do not add a quotient relation or a hidden completion point.

Finally inspect H, C, G, L. These rows transport, replay, source, and name the same finite route. Since no row contains quotient filters, selected completions, open-cover compactness, countable choice, host equality, or ambient completeness, the local NameCert surface exports none of those objects. □

Theorem 6310.3 (Proximity-space uniform-completion boundary)**.** *Every separated uniform-space or completion-facing consumer of an accepted* $\mathsf{ProximitySpace}^{\uparrow}$ *packet must read nearness through the finite diagonal, symmetry, and composition rows before entering the L10 uniform route. The boundary is*

$$\mathsf{ProximitySpace}^{\uparrow} \longmapsto \mathsf{SeparatedUniformSpace}^{\uparrow} \longmapsto \mathsf{UniformSpaceCompletion}^{\uparrow} \longmapsto \mathsf{CompleteUniformSpace}^{\uparrow}.$$

The downstream reads are the separated uniform boundary of chapter 1876, the uniform-space completion route of chapter 3560, and the complete uniform-space seal of chapter 1588. The proximity packet itself supplies only finite nearness data and does not construct quotient filters, select completions, prove open-cover compactness, or provide choice principles.

Proof. Begin with the carrier projection in Definition 6310.1. A consumer receives the finite nearness row N, the diagonal and apartness row D, the symmetry row S, and the composition row K before any downstream uniform object is read.

Use Definition 6310.2. It keeps the structural rows H, C, G, L attached to the same displayed nearness route, so a consumer cannot replace the route by a host proximity relation or a quotient of filters.

The separated uniform-space boundary is then the first downstream reader. It consumes the diagonal, symmetry, and composition information only as finite evidence compatible with chapter 1876.

Completion-facing consumers may then pass to chapter 3560 and chapter 1588. Since the proximity carrier has no coordinate for quotient filters, selected completions, open-cover compactness, countable choice, host equality, or ambient completeness, none of those objects is introduced by this boundary. □

Theorem 6310.4 (Proximity-space finite nearness stability)**.** *Let $P = (N, D, S, K, H, C, G, L)$ be an accepted* $\mathsf{ProximitySpace}^{\uparrow}$ *carrier. Any replay of a displayed nearness observation under* hsame*, symmetry, or finite composition remains inside the same finite route*

$$N, D, S, K \longmapsto H, C, G, L.$$

Thus proximity stability is a property of displayed BHist *rows and not a quotient-filter equality, selected completion, open-cover compactness theorem, countable-choice selector, host equality, or ambient completeness claim.*

Proof. Project the accepted packet. The only nearness observations available to the consumer are the finite rows displayed by N, with diagonal and apartness evidence controlled by D.

Symmetry and composition are read through S and K. A replay in the opposite order or through a finite middle observation therefore stays inside the carrier rather than creating a new comparison principle.

Componentwise hsame transport and displayed Cont replay are the structural rows H and C. They preserve the same route, while G and L provide provenance and local naming only. No excluded quotient, selected completion, compactness, choice, host equality, or ambient completeness object is present in the packet. □

Falsifiable prediction: Every accepted $\mathsf{ProximitySpace}^{\uparrow}$ consumer reads finite nearness, diagonal and apartness, symmetry, composition, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert before it may pass to separated uniformity or completion-facing rows.

Independence witness: separateduniformspace, uniformspacecompletion, completeuniformspace: $\mathsf{ProximitySpace}^{\uparrow}$ is not bijective to any listed sibling because it uniquely names finite nearness, diagonal and apartness, symmetry, and composition rows before the uniform and completion route is consumed.

Closure status for $\mathsf{NameCert}_{\mathsf{ProximitySpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ProximitySpace}^{\uparrow}$ packet is generated from a finite nearness row, a diagonal and apartness row, symmetry and composition rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ProximitySpace}^{\uparrow}}$, seed)

Scope closed: Definition 6310.1, Definition 6310.2, Definition 6310.3, and Definition 6310.4 seed the finite nearness interface over diagonal, apartness, symmetry, composition, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ProximitySpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient filters, selected completions, open-cover compactness, choice principles, host equality of points, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite nearness exposure, diagonal and apartness exactness, symmetry replay, finite composition, transport, replay, provenance, and non-escape for BEDC.Derived.ProximitySpaceUp.

Chapter 6311

A Concrete Naming Certificate for $\mathsf{CauchyTailSupremum}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{CauchyTailSupremum}^{\uparrow}$ packet records the finite BEDC surface by which a supremum-bound read of a Cauchy tail is consumed after the displayed tail-bound, threshold, window, dyadic, regular-readback, and real-seal rows are already visible. It sits after the tail-bound certificate of chapter 2071, the threshold discipline of chapter 1423, the finite stream window of chapter 126, the dyadic tolerance core of chapter 291, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a bounded-tail consumer surface; it does not assert an ambient completeness principle, an arbitrary host supremum, or a selected limit of a Cauchy stream.

Definition 6311.1 (Cauchy tail supremum carrier)**.** *A* $\mathsf{CauchyTailSupremum}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, M, W, D, R, E, U, H, C, P, N)$$

with the following displayed rows:

1. *B is the* $\mathsf{CauchyTailBound}^{\uparrow}$ *source row, recording the accepted tail-bound packet from which the bounded-tail read starts.*
2. *M is the* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold row, naming the stage after which the tail is read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *tail-window row, listing the inspected tail positions and the finite requests attached to them.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *upper-bound ledger, recording the dyadic radius, slack comparison, and upper-bound cells used by the supremum-bound read.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed window and dyadic ledger.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row visible only after the tail-bound source, threshold, window, dyadic ledger, and regular readback have been displayed.*
7. *U is the supremum-bound row, recording the finite assertion that the displayed upper-bound ledger bounds every inspected tail read admitted by B, M, W, D, R, E.*
8. *H records componentwise* hsame *transport for the tail-bound source, threshold, window, dyadic ledger, readback, Real seal, supremum-bound row, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay of the route*

$$B, M, W, D \longmapsto R \longmapsto E \longmapsto U.$$

10. *P is the* Pkg *provenance over* $\mathsf{CauchyTailSupremum}^{\uparrow}$, $\mathsf{CauchyTailBound}^{\uparrow}$, $\mathsf{RealCauchyModulus}^{\uparrow}$, $\mathsf{RegularCauchyCriterion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyTailSupremum}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $B, M, W, D, R, E, U, H, C, P, N$. There is no coordinate for a host supremum operator, quotient stream equality, selected limit, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge checking.*

Lean4 ✓ `BEDC.Derived.CauchyTailSupremumUp.CauchyTailSupremumTasteGate single carrier alignment`

Theorem 6311.2 (Cauchy tail supremum NameCert obligations)**.** *For every accepted* $\mathsf{CauchyTailSupremum}^{\uparrow}$ *carrier*

$$S = (B, M, W, D, R, E, U, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyTailSupremum}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows,* $\mathsf{CauchyTailBound}^{\uparrow}$ *source exposure through B,* $\mathsf{RealCauchyModulus}^{\uparrow}$ *threshold exposure through M,* $\mathsf{StreamName}^{\uparrow}$ *tail-window exactness through W,* $\mathsf{DyadicRatCore}^{\uparrow}$ *upper-bound accounting through D,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback through R,* $\mathsf{Real}^{\uparrow}$ *seal descent through E, supremum-bound accountability through U, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Proof. Carrier admission is projection from Definition 6311.1. The accepted packet exposes exactly $B, M, W, D, R, E, U, H, C, P, N$.

The source and threshold obligations are B and M. A consumer first reads the accepted $\mathsf{CauchyTailBound}^{\uparrow}$ source and then the $\mathsf{RealCauchyModulus}^{\uparrow}$ threshold that restricts which tail positions may be inspected.

The finite readback obligations are W, D, R, E. The window row lists the visible tail positions, the dyadic ledger records the upper-bound cells and slack comparison, the regular readback packages those observations, and the Real seal is downstream of those displayed rows.

The supremum-bound obligation is U. It records only the finite upper-bound assertion over the admitted tail-window reads, while H, C, P, N transport, replay, source, and name the same packet. Since no excluded principle is a carrier coordinate, the listed rows exhaust the local NameCert surface. □

Theorem 6311.3 (Cauchy tail supremum Real boundary)**.** *Every* $\mathsf{Real}^{\uparrow}$ *or completion-facing consumer of an accepted* $\mathsf{CauchyTailSupremum}^{\uparrow}$ *carrier must read the supremum-bound row only through the finite route*

$$\mathsf{CauchyTailBound}^{\uparrow}, \mathsf{RealCauchyModulus}^{\uparrow}, \mathsf{StreamName}^{\uparrow}, \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow} \longmapsto \mathsf{CauchyTailS}$$

Thus the packet supplies a displayed upper bound for admitted Cauchy-tail reads, but it supplies no host supremum, quotient stream equality, selected limit, countable-choice schedule, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or bridge credential.*

Proof. Begin with an accepted carrier $S = (B, M, W, D, R, E, U, H, C, P, N)$. Projection gives the tail-bound source, threshold row, finite tail window, dyadic upper-bound ledger, regular readback, Real seal, and supremum-bound row.

The route to U is downstream of B, M, W, D, R, E. A consumer cannot read the supremum-bound row until the tail-bound source and threshold have admitted the inspected window and the dyadic ledger has recorded the upper-bound cells consumed by the regular readback.

The Real-facing boundary is the seal row E, and it is already downstream of the finite tail route. Componentwise hsame transport and displayed Cont replay preserve that order, while Pkg provenance and the local NameCert row only source and name the same data.

Therefore the exported object is the finite supremum-bound packet itself. No coordinate remains from which a host supremum, quotient stream equality, selected limit, countable-choice schedule, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge credential could be read. □

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyTailSupremum}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyTailSupremum}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6311.1, Definition 6311.2, and Definition 6311.3 seed the finite supremum-bound route for Cauchy-tail reads over displayed tail-bound, threshold, window, dyadic, readback, and Real-seal rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyTailSupremum}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host supremum, quotient streams, selected limits, countable choice, trichotomy, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, CauchyTailBound source exposure, modulus-threshold exposure, StreamName tail-window exactness, DyadicRatCore upper-bound accounting, RegSeqRat readback, Real seal descent, supremum-bound accountability, transport, replay, provenance, and local naming for BEDC.Derived.CauchyTailSupremumUp

Constructive story: A $\mathsf{CauchyTailSupremum}^{\uparrow}$ packet is generated from a CauchyTailBound source row, a RealCauchyModulus threshold row, StreamName tail-window rows, a DyadicRatCore upper-bound ledger, RegSeqRat readback, a RealUp seal, a supremum-bound row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6312

A Concrete Naming Certificate for $\mathsf{SingularHomology}^{\uparrow}$

Following Eilenberg and Steenrod, the $\mathsf{SingularHomology}^{\uparrow}$ packet records singular homology as a finite NameCert-facing ledger of singular-simplex rows, chain rows, boundary rows, and cycle-boundary classifier rows. It refines the homology carrier of chapter 98, reads simplex incidence beside the finite combinatorics of chapter 238, keeps cochain duality separate from the cohomology certificate of chapter 99, and attaches all simplex domains and endpoint evaluations to the Topology packet of chapter 88. The chapter names the local certificate surface for singular chains; it does not assert exactness of long sequences, homotopy invariance, excision, universal coefficients, or de Rham comparison.

Definition 6312.1 (SingularHomology carrier)**.** *A* $\mathsf{SingularHomology}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (X, \Delta, \Sigma, C, B, Z, Q, H, T, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *carrier row naming the inspected space, its displayed opens, and the finite continuity witnesses used by singular simplices.*
2. *Δ is the finite simplex-domain row. It lists dimensions, face inclusions, vertex orders, and endpoint evaluations for the displayed standard simplices.*
3. *Σ is the singular-simplex row. Each entry records a displayed continuous map from a simplex domain in Δ into the topological space row X.*
4. *C is the chain row. It stores finite formal integer combinations of the displayed singular simplices and records the degree of each chain.*
5. *B is the boundary row. It records the alternating face ledger for each chain row using only the listed face maps and singular-simplex restrictions.*
6. *Z is the cycle-boundary classifier row. It separates rows whose boundary is the displayed zero chain from rows displayed as a boundary of a chain in the next degree.*
7. *Q is the quotient-facing homology-name row. It names the cycle modulo boundary classifier as local BEDC data rather than as host quotient equality.*

8. *H stores componentwise* hsame *transport for topology, simplex-domain, singular-simplex, chain, boundary, cycle-boundary classifier, quotient-facing name, continuation, provenance, and local naming rows.*
9. *T stores displayed* Cont *replay from simplex domains to singular maps, from singular maps to chains, from chains to boundaries, and from cycle-boundary classifiers to the local homology name.*
10. *P is the* Pkg *provenance over* $\mathsf{SingularHomology}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, $\mathsf{SimplicialComplex}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SingularHomology}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It contains no coordinate for ambient abelian-group quotients, choice of representatives, arbitrary topological colimits, exact-sequence machinery, homotopy invariance, excision, or de Rham integration.

Theorem 6312.2 (SingularHomology NameCert obligations)**.** *For every accepted* $\mathsf{SingularHomology}^{\uparrow}$ *carrier $S = (X, \Delta, \Sigma, C, B, Z, Q, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SingularHomology}^{\uparrow}}$ *surface is exhausted by topology admission for X, simplex-domain admission for Δ, singular-simplex continuity for Σ, finite chain admission for C, boundary-row computation for B, cycle-boundary classification for Z, quotient-facing homology naming for Q, componentwise* hsame *transport through H, displayed* Cont *replay through T, package provenance through P, and local naming through N.*

Proof. Project the accepted carrier row by row. The topology, simplex-domain, singular-simplex, chain, boundary, classifier, and homology-name coordinates are the only mathematical data visible to the local certificate.

The structural rows preserve that surface. Componentwise hsame transport in H can identify only displayed coordinates, the replay row T can traverse only the displayed route from simplex domains to homology names, and P, N record provenance and local naming for that same route.

The cited sibling chapters provide the intended boundaries: chapter 98 supplies the homology naming pattern, chapter 238 supplies finite face bookkeeping, chapter 99 remains the separate dual certificate surface, and chapter 88 supplies topological continuity data. Since no other row is present, the listed obligations exhaust the local NameCert surface. □

Theorem 6312.3 (SingularHomology boundary-square-zero ledger)**.** *For every accepted* $\mathsf{SingularHomology}^{\uparrow}$ *carrier $S = (X, \Delta, \Sigma, C, B, Z, Q, H, T, P, N)$, the displayed boundary ledger B contains the local witness that applying the singular boundary route twice reaches the displayed zero chain row. The witness is computed only from the alternating face rows in Δ, the singular-simplex restrictions in Σ, the chain row C, and the replay data in T; it does not use host quotient equality, selected chain representatives, homotopy invariance, excision, or a de Rham comparison theorem.*

Proof. Read the simplex-domain row Δ. Its face-inclusion ledger lists the two ways each codimension-two face appears in an iterated boundary, with opposite alternating signs.

Next read the singular-simplex row Σ and the chain row C. Restricting singular simplices along the two displayed face routes produces the same listed simplex face with opposite signs, so the iterated boundary entries cancel inside the finite chain ledger.

Finally replay the result through B, Z, Q, H, T, P, N. The boundary row records the cancellation as a displayed zero-chain witness, the classifier row Z consumes it as the cycle side of the cycle-boundary ledger, and the local homology-name row Q receives only that finite witness. No excluded global theorem or external quotient principle is available in the carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{SingularHomology}^{\uparrow}}$.

Constructive story: A $\mathsf{SingularHomology}^{\uparrow}$ packet is generated from a Topology row, finite simplex-domain rows, singular-simplex rows, chain rows, boundary rows, cycle-boundary classifier rows, quotient-facing homology names, componentwise transport, displayed continuation replay, package provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SingularHomology}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6312.1, Definition 6312.2, and Definition 6312.3 seed the finite singular-chain certificate surface and its boundary-square-zero ledger.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SingularHomology}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim exactness of long sequences, homotopy invariance, excision, universal coefficients, de Rham comparison, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, singular-simplex continuity, chain admission, boundary computation, cycle-boundary classification, transport stability, and local NameCert replay for BEDC.Derived.SingularHomologyUp.

Chapter 6313

A Concrete Naming Certificate for $\mathsf{KaramataInequality}^{\uparrow}$

Following Hardy–Littlewood–Polya, the $\mathsf{KaramataInequality}^{\uparrow}$ packet records the finite majorization route by which a convex-function row and two ordered finite vector rows produce the Karamata comparison. It consumes the Jensen route of chapter 2235, the convex-carrier surface of chapter 208, the rational coefficient source of chapter 37, the vector carrier of chapter 47, the real-name boundary of chapter 38, and the endpoint comparison discipline of chapter 1674. The packet names the traditional majorization inequality as finite BEDC data; it does not import a host convex function, quotient equality of real names, selected extrema, trichotomy, countable choice, or ambient real completeness.

Definition 6313.1 (Karamata inequality carrier)**.** *A* $\mathsf{KaramataInequality}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (F, X, Y, M, A, J, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *F is the convex-function source row, including the displayed domain, codomain, convexity ledger, and function-read route.*
2. *X and Y are the two finite vector rows over* $\mathsf{VecSpace}^{\uparrow}$*, with their coordinate reads and finite ordering windows.*
3. *M is the majorization ledger, recording the prefix-sum comparisons, total-sum agreement, and rational coefficient witnesses read through* $\mathsf{Rat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$*.*
4. *A is the* $\mathsf{ConvexSet}^{\uparrow}$ *affine-spine row certifying that the Jensen decompositions used by the comparison stay inside the displayed convex carrier.*
5. *J is the* $\mathsf{JensenInequality}^{\uparrow}$ *handoff row, recording the finite Jensen calls used to convert the majorization ledger into image-sum comparisons.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *comparison row for the displayed sum of convex images.*
7. *Q is the* $\mathsf{RealMinMax}^{\uparrow}$ *endpoint row used when a consumer reads the ordered finite comparison.*
8. *H records componentwise* hsame *transport for F, X, Y, M, A, J, R, Q and the local naming row.*
9. *C records displayed* Cont *replay from majorization rows through the Jensen handoff, real comparison, endpoint row, provenance, and local name.*

10. *P is the* Pkg *provenance over* $\mathsf{KaramataInequality}^{\uparrow}$, $\mathsf{JensenInequality}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealMinMax}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{KaramataInequality}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, X, Y, M, A, J, R, Q, H, C, P, N$. It has no coordinate for a host convex function, quotient equality of real names or functions, selected rearrangements, trichotomy, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6313.2 (Karamata inequality NameCert obligations)**.** *For every accepted* $\mathsf{KaramataInequality}^{\uparrow}$ *carrier $K = (F, X, Y, M, A, J, R, Q, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{KaramataInequality}^{\uparrow}}$ *surface consists exactly of carrier admission for the convex-function source row, the two finite vector rows, the majorization ledger, the convex-set affine-spine row, the Jensen handoff row, the real comparison row, the endpoint row, structural rows H, C, P, and local naming row N; prefix-sum and total-sum accountability through M; affine-spine accountability through A; Jensen-mediated comparison through J, R, Q; and componentwise transport, replay, provenance, and naming for the same finite route.*

Proof. Project the accepted packet $K = (F, X, Y, M, A, J, R, Q, H, C, P, N)$. The carrier definition exposes exactly the convex-function source, finite vector inputs, majorization ledger, affine-spine route, Jensen handoff, real comparison, endpoint comparison, transport, replay, provenance, and naming rows.

The majorization ledger M is read before the inequality row. It supplies the prefix-sum comparisons, total-sum agreement, and rational coefficient witnesses, while A supplies the displayed $\mathsf{ConvexSet}^{\uparrow}$ route that keeps each Jensen decomposition inside the accepted convex carrier.

The handoff row J then invokes only the already displayed Jensen certificate. The comparison rows R, Q carry the resulting real and endpoint comparisons, and H, C, P, N preserve and name that route. Since no host convex-function object, quotient equality, selected rearrangement, or ambient completeness coordinate is present, the listed rows exhaust the local NameCert obligations. □

Theorem 6313.3 (Karamata inequality Jensen-majorization handoff)**.** *Every accepted* $\mathsf{KaramataInequality}^{\uparrow}$ *packet routes its public comparison through*

$$X, Y, M \longmapsto A \longmapsto J \longmapsto R, Q \longmapsto H, C, P, N.$$

Consequently a consumer must read the two finite vector rows, the majorization ledger, the $\mathsf{ConvexSet}^{\uparrow}$ *affine-spine row, the* $\mathsf{JensenInequality}^{\uparrow}$ *handoff row, the* $\mathsf{Real}^{\uparrow}$ *comparison row, the* $\mathsf{RealMinMax}^{\uparrow}$ *endpoint row, componentwise transport, replay, provenance, and local naming before it can cite the Karamata inequality surface.*

Proof. Start with a consumer read of the Karamata comparison. By Definition 6313.1, the only majorization data are the vector rows X, Y and the ledger M.

The affine-spine row A is downstream of those finite rows. It keeps the displayed decompositions inside the same convex carrier before any Jensen call is read.

Apply the Jensen handoff row J. It is a reference to the finite NameCert surface of chapter 2235, so the consumer receives the Jensen comparison only after the majorization and affine-spine data have been displayed.

Finally R, Q, H, C, P, N expose the real comparison, endpoint comparison, transport, replay, provenance, and local naming. Thus no consumer can bypass the finite majorization route by invoking a host convexity theorem or ambient real-completeness principle. □

Falsifiable prediction: Every $\mathsf{KaramataInequality}^{\uparrow}$ consumer reads a convex-function source row, two finite VecSpace rows, a majorization ledger, a ConvexSet affine-spine row, a JensenInequality handoff row, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before the Karamata inequality surface is public.

Independence witness: jenseninequality, convexset, rat, vecspace, real, realminmax: $\mathsf{KaramataInequality}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries two ordered finite vector rows, a majorization ledger, affine-spine data, Jensen handoff, real comparison, endpoint comparison, transport, replay, provenance, and local NameCert data as one finite inequality packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KaramataInequality}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KaramataInequality}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6313.1, Definition 6313.2, and Definition 6313.3 seed the finite Karamata inequality route over convex-function, majorization, Jensen, ConvexSet, Rat, VecSpace, Real, and RealMinMax rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{KaramataInequality}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host convex functions, quotient equality of real names or functions, selected rearrangements, trichotomy, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, majorization ledger accountability, ConvexSet affine-spine exactness, Jensen handoff exactness, Real comparison nonescape, endpoint comparison handoff, and public export scope for BEDC.Derived.KaramataInequalityUp.

Constructive story: A $\mathsf{KaramataInequality}^{\uparrow}$ packet is generated from a convex-function source row, two finite VecSpace rows, a majorization ledger, a ConvexSet affine-spine row, a JensenInequality handoff row, Real comparison rows, RealMinMax endpoint rows, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6314

A Concrete Naming Certificate for $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$

The $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ packet realizes the redex-frontier inspection route of chapter 9383 as finite NameCert data. It sits beside the critical-path route of chapter 765, the normal-form consistency seal of chapter 740, and the normalization frontier packet of chapter 892. The packet records a bounded typed-term window, the displayed redex-frontier row, candidate-mediated strong-normalization handoff data, and the audit ledger that a consumer must read before making any normalization-audit conclusion.

Definition 6314.1 (Finite-window normalization audit carrier)**.** *A* $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (W, R, K, F, S, L, H, C, P, N)$$

with the following displayed rows:

1. *W is the finite typed-term window, including the inspected closed typed terms and the local depth or fuel boundary for the audit.*
2. *R is the redex-frontier row, recording exactly which beta-frontier and neutral-frontier cells are visible in the window.*
3. *K is the* $\mathsf{MetaCICCriticalPath}^{\uparrow}$ *handoff row, naming the consistency-facing route that any audit consumer must preserve.*
4. *F is the* $\mathsf{MetaCICNormalizationFrontier}^{\uparrow}$ *row, separating candidate evidence, finished-normal use, and subject-reduction obstruction data.*
5. *S is the* $\mathsf{NormalFormConsistencySeal}^{\uparrow}$ *row, consumed only after the frontier rows have been displayed.*
6. *L is the normalization audit ledger, recording candidate-mediated strong-normalization evidence, no-infinite-reduction projections, and refused global-normalization claims.*
7. *H records componentwise* hsame *transport for W, R, K, F, S, L and the local naming row.*
8. *C records displayed* Cont *replay from the finite window through redex-frontier inspection, candidate handoff, consistency-seal use, audit ledger, provenance, and local name.*
9. *P is the* Pkg *provenance over* $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$, $\mathsf{MetaCICCriticalPath}^{\uparrow}$, $\mathsf{MetaCICNormalizationFro}$ $\mathsf{NormalFormConsistencySeal}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $W, R, K, F, S, L, H, C, P, N$. It has no coordinate for global strong normalization, full closed consistency, closed subject reduction, unrestricted beta-conversion decidability, hidden normalization transport, or host equality of MetaCIC terms.

Theorem 6314.2 (Finite-window normalization audit NameCert obligations)**.** *For every accepted* $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ *carrier $A = (W, R, K, F, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FiniteWindowNormalizatio}}$ *surface consists exactly of carrier admission for the finite typed-term window, redex-frontier row, critical-path handoff row, normalization-frontier row, normal-form consistency seal row, normalization audit ledger, componentwise transport, displayed replay, provenance, and local naming. The audit conclusion is public only after the window W, frontier R, F, candidate handoff data in L, consistency-seal row S, transport H, replay C, provenance P, and local name N have been displayed.*

Proof. Project the accepted carrier $A = (W, R, K, F, S, L, H, C, P, N)$. The carrier definition exposes exactly the finite typed-term window, redex-frontier row, critical-path handoff, normalization frontier, consistency seal, audit ledger, transport, replay, provenance, and naming rows.

The window and redex-frontier rows are read first. They determine which finite terms and which frontier cells are inspected, so no conclusion can range over arbitrary closed typed terms.

The handoff rows K, F, S then attach the inspection to the existing MetaCIC critical path, normalization frontier, and normal-form consistency seal. The ledger row L records candidate-mediated strong-normalization evidence and refused global claims, while H, C, P, N preserve and name the same route.

Because the carrier has no coordinate for global strong normalization, closed subject reduction, full closed consistency, unrestricted beta-conversion, hidden normalization transport, or host equality, the listed rows exhaust the local NameCert obligations. □

Theorem 6314.3 (Finite-window normalization audit redex frontier)**.** *Every accepted* FiniteWindowNormalizationA *packet routes a normalization-audit read through*

$$W \longmapsto R \longmapsto F, L \longmapsto K, S \longmapsto H, C, P, N.$$

Thus a consumer must expose the finite typed-term window, redex-frontier row, candidate-mediated normalization frontier, audit ledger, MetaCIC critical-path row, normal-form consistency seal, componentwise transport, replay, provenance, and local naming before any normalization-audit conclusion is public.

Proof. Start with an audit consumer read. The finite window W is the only source of inspected terms, and the redex-frontier row R is the only source of visible frontier cells.

The normalization frontier F and audit ledger L are read after that window. They keep candidate evidence, finished-normal use, no-infinite-reduction consequences, and subject-reduction obstruction data attached to the displayed inspection.

The critical-path row K and consistency-seal row S are then consumed as handoff rows. They do not turn the packet into a proof of global normalization or full closed consistency; they only record how the finite audit may cite the existing MetaCIC route and seal.

Finally H, C, P, N transport, replay, source, and name the same finite route. Therefore every normalization-audit conclusion remains confined to the displayed window and redex frontier. □

Falsifiable prediction: Every accepted $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ read exposes the finite typed-term window, redex-frontier row, candidate handoff, normalization audit ledger, hsame

transport, Cont replay, Pkg provenance, and local NameCert row before any normalization-audit conclusion is public.

Independence witness: metaciccriticalpath, normalformconsistencyseal, metacicnormalization-frontier, metacicclosuretrace: none is bijective with $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ because the audit packet jointly binds the finite term window, redex frontier, candidate-mediated SN handoff, audit ledger, transport, replay, provenance, and local naming.

The Lean TasteGate surface for this packet must provide explicit $\mathsf{Nontrivial\ FiniteWindowNormalizationAudit}^{\uparrow}$ and $\mathsf{FieldFaithful\ FiniteWindowNormalizationAudit}^{\uparrow}$ instances preserving $W, R, K, F, S, L, H, C, P, N$ as distinct displayed rows before a checked target may cite the packet as a verified carrier-alignment theorem.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6314.1, Definition 6314.2, and Definition 6314.3 seed the finite normalization-audit route over typed-term windows, redex frontiers, critical-path rows, normalization-frontier rows, normal-form consistency seals, audit ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close global strong normalization, full closed consistency, closed subject reduction, unrestricted beta-conversion decidability, hidden normalization transport, host equality of MetaCIC terms, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, redex-frontier exactness, candidate-mediated audit ledger exposure, critical-path handoff, normal-form seal handoff, transport, replay, provenance, local naming, and Nontrivial and FieldFaithful instances for BEDC.Derived.FiniteWindowNormalizationAuditUp.

Constructive story: A $\mathsf{FiniteWindowNormalizationAudit}^{\uparrow}$ packet is generated from a finite typed-term window, redex-frontier row, MetaCIC critical-path row, normalization-frontier row, normal-form consistency seal row, candidate-mediated audit ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6315

A Concrete Naming Certificate for $\mathsf{BishopRegularSequence}^{\uparrow}$

Following Bishop and Bridges, *Constructive Analysis*, the $\mathsf{BishopRegularSequence}^{\uparrow}$ packet records the lawlike regular rational sequence before it is sealed as a $\mathsf{Real}^{\uparrow}$ name. It sits between the dyadic observation core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the Cauchy-modulus schedule of chapter 292, and the terminal real-name boundary of chapter 38. The packet names finite observations, windows, regularity rows, tail schedules, seal handoff, transport, replay, provenance, and a local certificate; it does not allocate quotient streams, selected representatives, countable-choice schedules, host real equality, or ambient completeness.

Definition 6315.1 (Bishop regular-sequence carrier)**.** *A* $\mathsf{BishopRegularSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, R, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *observation row, naming the dyadic rational values and tolerance comparisons available to the sequence.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, exposing the inspected prefixes and tail windows of the lawlike sequence.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row attached to the same windows and dyadic observations.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *schedule row, recording the finite tail threshold supplied for each dyadic tolerance request.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff, available only after D, W, R, M have been displayed.*
6. *H stores componentwise* hsame *transport for the observation, window, regularity, modulus, seal, replay, provenance, and local naming rows.*
7. *C stores displayed* Cont *replay from observations and windows through regularity and modulus data to the seal handoff.*
8. *P is the* Pkg *provenance over* $\mathsf{BishopRegularSequence}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopRegularSequence}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient stream equality, selected limits, hidden representatives, countable choice, host real equality, or an ambient completeness principle.

Theorem 6315.2 (Bishop regular-sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopRegularSequence}^{\uparrow}$ *carrier* $B = (D, W, R, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopRegularSequence}^{\uparrow}}$ *surface consists of the following five obligation rows:*

1. *carrier admission for the displayed packet* $D, W, R, M, E, H, C, P, N$*;*
2. *observation and window exposure through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*;*
3. *regularity and tail-modulus exposure through* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{CauchyModulus}^{\uparrow}$*;*
4. *terminal* $\mathsf{Real}^{\uparrow}$ *seal handoff through* E *after the observation, window, regularity, and modulus rows are visible;*
5. *structural stability through* hsame*,* Cont*,* Pkg*, and the local* $\mathsf{NameCert}$ *row.*

No accepted obligation row may be discharged by quotient streams, a selected limit, countable choice, host real equality, or ambient Real completeness.

Proof. Project the carrier of Definition 6315.1. The analytic rows are exactly D, W, R, M, E, and the structural rows are exactly H, C, P, N.

The observation and window rows are read first. The $\mathsf{DyadicRatCore}^{\uparrow}$ row supplies dyadic values and tolerance comparisons, while the $\mathsf{StreamName}^{\uparrow}$ row supplies the finite inspected windows.

Next read the regularity and modulus rows. The $\mathsf{RegSeqRat}^{\uparrow}$ row states the displayed regularity of the same windows, and the $\mathsf{CauchyModulus}^{\uparrow}$ row supplies the finite tail threshold for each dyadic tolerance request.

Finally the seal handoff and structural rows are consumed. The $\mathsf{Real}^{\uparrow}$ row E is downstream of D, W, R, M, and H, C, P, N transport, replay, source, and name that same finite route. Since the carrier has no coordinate for the refused host-level data, the five obligations exhaust the local NameCert surface. □

Theorem 6315.3 (Bishop regular-sequence tail-modulus induction)**.** *For every accepted* $\mathsf{BishopRegularSequence}^{\uparrow}$ *carrier* $B = (D, W, R, M, E, H, C, P, N)$ *and every finite dyadic tolerance request read through* D*, primitive induction on the displayed tail schedule* M *produces a window threshold in* W *such that all later inspected windows preserve the* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity row* R *before the* $\mathsf{Real}^{\uparrow}$ *seal handoff* E *is exposed. The induction stays inside the route*

$$D \longmapsto M \longmapsto W \longmapsto R \longmapsto E \longmapsto H, C, P, N.$$

It does not select an external infinite tail, quotient two streams, choose a representative, or invoke ambient completeness.

Proof. Begin with a dyadic tolerance request in D. The request is finite BEDC data, so it can be passed to the displayed $\mathsf{CauchyModulus}^{\uparrow}$ schedule row M without invoking an external selector.

Induct on the finite tail schedule carried by M. The base case exposes the first admissible window in W for the requested tolerance. The successor step refines the inspected window while preserving the same dyadic tolerance row and the same $\mathsf{StreamName}^{\uparrow}$ route.

Use the regularity row R after the window threshold is displayed. The $\mathsf{RegSeqRat}^{\uparrow}$ row is attached to D and W, so every later inspected window remains a regular rational readback inside the same packet.

Finally expose E, H, C, P, N. The $\mathsf{Real}^{\uparrow}$ seal is reached only after the induced tail threshold and regularity row are visible, and the structural rows replay the same finite route. Thus the

induction provides a tail-modulus handoff for the public seal without quotient streams, selected limits, hidden representatives, or ambient completeness. □

Falsifiable prediction: A claimed $\mathsf{BishopRegularSequence}^{\uparrow}$ consumer that reaches a $\mathsf{Real}^{\uparrow}$ seal without displaying $\mathsf{DyadicRatCore}^{\uparrow}$ observations, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ regularity, and a $\mathsf{CauchyModulus}^{\uparrow}$ tail schedule refutes the carrier.

Independence witness: dyadicratcore, streamname, regseqrat, cauchymodulus, real: no listed sibling is bijective to $\mathsf{BishopRegularSequence}^{\uparrow}$ because the packet jointly carries dyadic observations, finite sequence windows, regularity rows, tail-modulus schedules, Real seal handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopRegularSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopRegularSequence}^{\uparrow}$ packet is generated from DyadicRatCore observations, StreamName finite windows, RegSeqRat regularity rows, a CauchyModulus tail schedule, a RealUp seal handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopRegularSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6315.1, Definition 6315.2, and Definition 6315.3 seed the finite regular-sequence route from dyadic observations and windows to the Real seal handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopRegularSequence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{BishopRegularSequence}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, observation and window exposure, regularity readback, tail-modulus induction, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopRegularSequence

Chapter 6316

A Concrete Naming Certificate for $\mathsf{SystemF}^{\uparrow}$

Following Girard and Reynolds, $\mathsf{SystemF}^{\uparrow}$ is read here as the finite NameCert surface for the polymorphic typed lambda calculus. It sits between the term-syntax carrier of chapter 200, the first-order symbol and formula interface of chapter 197, the computability boundary of chapter 199, the type-like classifier surface of chapter 3600, and the typed normalization packets of chapter 6305 and chapter 6306. The packet names polymorphic syntax and typing as displayed BEDC rows; it does not export an ambient host calculus, quotient syntax equality, impredicative set semantics, or a global normalization theorem.

Definition 6316.1 (System F carrier)**.** *A* $\mathsf{SystemF}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, A, L, E, T, B, I, H, C, P, N)$$

with the following displayed rows:

1. *X is the term-variable row, naming de Bruijn or locally named term variables through finite* BHist *syntax endpoints.*
2. *A is the type-variable row, naming type variables separately from term variables and recording their binder depth.*
3. *L is the lambda row, carrying term abstraction and term application over the* $\mathsf{LambdaCalc}^{\uparrow}$ *syntax surface.*
4. *E is the type row, carrying type variables, arrow types, and universal type binders as displayed finite syntax.*
5. *T is the typing-context row, listing term-variable assumptions, type-variable assumptions, and the finite scope in which a judgment is read.*
6. *B is the beta and type-instantiation route row, recording term beta contraction and universal-type instantiation as ledgered source and endpoint rows.*
7. *I is the type-abstraction and type-application row, recording Λ-abstraction over type variables and application of polymorphic terms to displayed type rows.*
8. *H records componentwise* hsame *transport for term variables, type variables, lambda syntax, type syntax, contexts, beta and instantiation routes, abstraction/application rows, replay, provenance, and local names.*

9. *C records displayed* Cont *replay from a requested typing or reduction read through X, A, L, E, T, B, I.*
10. *P is the* Pkg *provenance over* $\mathsf{LambdaCalc}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{Computable}^{\uparrow}$, $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{NormalizationByEvaluation}^{\uparrow}$, $\mathsf{NeutralReifier}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{SystemF}^{\uparrow}}$ *row.*

The classifier accepts a packet only when term syntax, type syntax, typing context, beta routes, and type-instantiation routes are all displayed as finite rows. No coordinate may be a host parser, a host type checker, an external alpha-equivalence quotient, or an unlogged semantic model.

Theorem 6316.2 (System F NameCert obligations)**.** *For every accepted* $\mathsf{SystemF}^{\uparrow}$ *carrier $F = (X, A, L, E, T, B, I, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{SystemF}^{\uparrow}}$ *surface consists exactly of carrier admission, typing-surface exactness, type-abstraction stability, beta/type-instantiation nonescape, and provenance exhaustion over the displayed rows. In particular, every consumer of a* $\mathsf{SystemF}^{\uparrow}$ *typing, beta, or type-instantiation conclusion must first read the route*

$$X, A, L, E, T, B, I, H, C, P, N.$$

The packet supplies polymorphic typed-lambda-calculus data, but it supplies no host-level type checker, quotient equality of terms or types, impredicative semantic universe, parametricity theorem, subject-reduction theorem beyond the displayed route, or global normalization result.

Proof. Carrier admission is projection from Definition 6316.1. The tuple exposes term variables, type variables, lambda syntax, type syntax, typing context, beta/type-instantiation routes, type abstraction/application rows, transport, replay, provenance, and local naming as the only accepted coordinates.

Typing-surface exactness follows by reading the context row T together with the term and type syntax rows X, A, L, E. A typing judgment is accepted only when its variable assumptions, type assumptions, binder scopes, term endpoint, and type endpoint are all present in those displayed rows.

Type-abstraction stability is the role of I. The type-variable row A records the binder being abstracted, the context row T records the extended scope, and componentwise transport H preserves the same abstraction or type-application endpoint under hsame.

Beta and type-instantiation nonescape is enforced by B and C. A term beta read must pass through the displayed lambda source and substitution endpoint, while a polymorphic instantiation read must pass through the displayed universal type binder and argument type row; replay cannot jump to a host evaluator or external substitution function.

Provenance exhaustion is supplied by P and N. The cited sibling chapters provide the finite syntax, typing-shape, computability, type-like, and normalization-facing surfaces consumed by the packet, and no additional row is available from which a consumer could recover quotient syntax equality, impredicative semantics, parametricity, subject reduction, or global normalization. □

Falsifiable prediction: Every accepted $\mathsf{SystemF}^{\uparrow}$ typing, beta, or type-instantiation read factors through the term-variable row, type-variable row, lambda syntax row, type syntax row, typing-context row, beta/type-instantiation route, type-abstraction/application row, componentwise transport, continuation replay, provenance, and local NameCert before any consumer may cite the conclusion.

Independence witness: lambdacalc, firstorder, computable, typelike, normalization-by-evaluation, neutral-reifier: $\mathsf{SystemF}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries term variables, type variables, typed lambda syntax, universal type abstraction and application, typing contexts, beta routes, type-instantiation routes, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{SystemF}\uparrow}$**.**

Constructive story: A $\mathsf{SystemF}^{\uparrow}$ packet is generated from term-variable rows, type-variable rows, lambda syntax rows, type syntax rows, typing-context rows, beta/type-instantiation route rows, type-abstraction/application rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{SystemF}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6316.1 and Definition 6316.2 seed the finite polymorphic typed-lambda-calculus packet and its NameCert obligation surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{SystemF}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient syntax equality, impredicative set semantics, parametricity, closed subject reduction, global normalization, or Lean verification.

Upgrade path: Obligation closure requires checked carrier admission, typing-surface exactness, type-abstraction stability, beta/type-instantiation nonescape, and provenance exhaustion for BEDC.Derived.SystemFUp.

Chapter 6317

A Concrete Naming Certificate for $\mathsf{CompilerClosureSelfCompilation}^{\uparrow}$

The $\mathsf{CompilerClosureSelfCompilation}^{\uparrow}$ packet realizes the compiler-closure self-compilation audit surface of chapter 9411 as finite BEDC rows. It sits beside the subject-reduction-discharge-socket chapter chapter 719, the metacic-subject-reduction-trace chapter chapter 723, the normalization-by-evaluation chapter chapter 6305, the neutral-reifier chapter chapter 6306, and the public-local-reflective-compilation chapter chapter 3963. The packet names the finite flow from genome data to compiler phenotype, feedback, export seal, and audit ledger; it does not claim an unrestricted reflective compiler, hidden host prompt state, closed subject reduction, or unbounded self-application.

Definition 6317.1 (CompilerClosureSelfCompilation carrier)**.** *A* $\mathsf{CompilerClosureSelfCompilation}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, \Phi, F, L, E, S, H, C, P, N)$$

with the following displayed rows:

1. *G is the genome-flow row, recording the finite source theorem-flow, carrier rows, and compiler-facing genome entries supplied to the self-compilation audit.*
2. *Φ is the compiler-phenotype row, recording the expressed compiler behavior as a finite BEDC object rather than a host executable.*
3. *F is the feedback-ledger row, recording how accepted compiler observations feed back into the next accepted genome-flow row.*
4. *L is the audit-ledger row, recording the finite evidence obligations that link genome flow, phenotype, feedback, and accepted output.*
5. *E is the export-seal row, recording which generated artifacts may be exported as public compiler data and which remain local audit evidence.*
6. *S is the sibling-route row, naming subject-reduction-discharge-socket, metacic-subject-reduction-trace, normalization-by-evaluation, neutral-reifier, and public-local-reflective-compilation as the only imported certificate surfaces.*
7. *H records componentwise* hsame *transport for genome flow, compiler phenotype, feedback ledger, audit ledger, export seal, sibling routes, replay, provenance, and local names.*
8. *C records displayed* Cont *replay from a requested self-compilation read through G, Φ, F, L, E, S.*

9. *P is the* Pkg *provenance over* SubjectReductionDischargeSocket$^{\uparrow}$, MetaCICSubjectReductionTrace$^{\uparrow}$, NormalizationByEvaluation$^{\uparrow}$, NeutralReifier$^{\uparrow}$, PublicLocalReflectiveCompilation$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompilerClosureSelfCompilation}^{\uparrow}}$ *row.*

The classifier accepts a packet only when genome flow, compiler phenotype, feedback, export, transport, replay, provenance, and local naming are all displayed as finite rows. No coordinate stores a host compiler executable, hidden global state, an unrestricted reflective evaluator, or a closed subject-reduction theorem.

Theorem 6317.2 (CompilerClosureSelfCompilation NameCert obligations)**.** *For every accepted* CompilerClosureSelfCompilation$^{\uparrow}$ *carrier $K = (G, \Phi, F, L, E, S, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompilerClosureSelfCompila}}$ *surface consists exactly of genome-flow admission, compiler-phenotype exposure, feedback-ledger replay, export-seal exactness, transport stability, replay stability, provenance exhaustion, and local naming over the displayed rows. In particular, every consumer of a compiler-closure self-compilation conclusion must first read the route*

$$G, \Phi, F, L, E, S, H, C, P, N.$$

The route exports no host compiler executable, hidden prompt state, global reflective authority, unlogged feedback loop, closed subject-reduction proof, or unbounded self-application.

Proof. Genome-flow admission projects the row G from the carrier. The input flow is finite theorem and carrier data, so a consumer cannot replace it by an ambient repository state or hidden prompt stream.

Compiler-phenotype exposure is the row Φ. It records expressed compiler behavior as a displayed object and therefore separates compiler phenotype from a host executable or external build system.

Feedback-ledger replay uses F and L. The accepted observation must be listed in the feedback ledger and tied to the audit ledger before it can affect the next genome-flow row.

Export-seal exactness is supplied by E. Public compiler data may leave the packet only through the seal, while local audit rows remain local and cannot be promoted to public authority by replay alone.

Transport and replay stability follow from H and C. These rows move genome flow, phenotype, feedback, ledger, export, and sibling-route data together, so no consumer can bypass the displayed sequence.

Provenance exhaustion is supplied by S, P, N. The only sibling surfaces available are subject-reduction-discharge-socket, metacic-subject-reduction-trace, normalization-by-evaluation, neutral-reifier, and public-local-reflective-compilation, and none of them adds a host compiler executable, hidden prompt state, closed subject-reduction proof, or unrestricted self-application coordinate. □

Falsifiable prediction: Every accepted CompilerClosureSelfCompilation$^{\uparrow}$ self-compilation read factors through the genome-flow row, compiler-phenotype row, feedback ledger, audit ledger, export seal, sibling-route row, componentwise transport, continuation replay, provenance, and local NameCert before any consumer may cite compiler-closure output.

Independence witness: subject-reduction-discharge-socket, metacic-subject-reduction-trace, normalization-by-evaluation, neutral-reifier, public-local-reflective-compilation: CompilerClosureSelfCompilation$^{\uparrow}$ is not bijective to any listed sibling because it jointly carries genome flow, compiler phenotype,

feedback ledger, audit ledger, export seal, sibling routing, transport, replay, provenance, and local naming rows.

Closure status for $\mathsf{NameCert}_{\mathsf{CompilerClosureSelfCompilation}^{\uparrow}}$.

Constructive story: A $\mathsf{CompilerClosureSelfCompilation}^{\uparrow}$ packet is generated from genome-flow rows, compiler-phenotype rows, feedback-ledger rows, audit-ledger rows, export-seal rows, sibling-route rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompilerClosureSelfCompilation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6317.1 and Definition 6317.2 seed the finite compiler-closure self-compilation packet and its audit surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompilerClosureSelfCompilation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close unrestricted reflective compilation, hidden prompt state, closed subject reduction, host compiler semantics, or Lean verification.

Upgrade path: Obligation closure requires checked genome-flow admission, compiler-phenotype exposure, feedback-ledger replay, export-seal exactness, transport, replay, provenance, and local naming for BEDC.Derived.CompilerClosureSelfCompilationUp.

Chapter 6318

A Concrete Naming Certificate for $\mathsf{ChoiceSequenceReal}^{\uparrow}$

Following Bishop–Bridges constructive analysis and Troelstra–van Dalen on choice sequences, the $\mathsf{ChoiceSequenceReal}^{\uparrow}$ packet names a finite-prefix real surface before any consumer reads an infinite branch. It sits after the finite StreamName window of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. It also shares the choice-free boundary discipline of chapter 1406: finite prefixes may be inspected, but no selected limit, quotient stream equality, countable choice, trichotomy, or ambient Real completeness is exported.

Definition 6318.1 (ChoiceSequenceReal carrier)**.** *A* $\mathsf{ChoiceSequenceReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$C = (F, S, R, D, E, H, T, P, N)$$

with the following displayed rows:

1. *F is the finite prefix schedule. It lists the admitted prefix length, the unary request indices, and the displayed dependency of each read on earlier finite choices.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row. It records the finite stream observations exposed by F before any infinite branch is named.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It stores the regular rational values read from the finite stream window and the visible modulus entries used by later real-name consumers.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger. It records the signed dyadic mantissa, denominator, radius, and terminal half-radius rows used to certify the readback tolerance.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row. It may be consumed only after F, S, R, D have been displayed, and it stores no selected endpoint beyond that finite route.*
6. *H records componentwise* hsame *transport for finite prefixes, stream windows, regular rational readback, dyadic tolerance rows, Real seal rows, replay, provenance, and local naming.*
7. *T records displayed* Cont *replay from finite prefix schedules to StreamName windows, from StreamName windows to RegSeqRat readback, from dyadic tolerance rows to the terminal Real seal, and from structural rows to local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{ChoiceSequenceReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{ChoiceSequenceReal}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $F, S, R, D, E, H, T, P, N$. It has no coordinate for quotient stream equality, selected infinite branches, countable choice, trichotomy of arbitrary reals, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6318.2 (ChoiceSequenceReal NameCert obligations)**.** *For every accepted* $\mathsf{ChoiceSequenceReal}^{\uparrow}$ *carrier $C = (F, S, R, D, E, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ChoiceSequenceReal}^{\uparrow}}$ *surface consists exactly of finite prefix admission, StreamName window exposure, RegSeqRat regular readback, DyadicRatCore tolerance ledger admission, Real seal routing, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real or Completion consumer must read the displayed route*

$$F, S, R, D \longmapsto E, H, T, P, N$$

before it can consume the packet as real-name evidence.

Proof. Project the accepted carrier row by row. The finite prefix schedule F, the stream window S, the regular rational readback R, and the dyadic tolerance ledger D are the only rows that carry mathematical read data.

The Real seal row E is downstream from those rows. It can be replayed only through the displayed Cont route T and transported only through the componentwise hsame row H.

The provenance and naming rows P, N name the same route. Since no other coordinate is present, the local certificate has no source for quotient stream equality, selected infinite branches, countable choice, trichotomy, metric completeness, or ambient Real completeness. □

Theorem 6318.3 (ChoiceSequenceReal finite-prefix regularity)**.** *Let $C = (F, S, R, D, E, H, T, P, N)$ be an accepted* $\mathsf{ChoiceSequenceReal}^{\uparrow}$ *carrier. Any finite extension of the displayed prefix schedule whose new reads preserve the same StreamName window discipline and DyadicRatCore tolerance ledger has a RegSeqRat readback on a common finite window before the downstream* $\mathsf{Real}^{\uparrow}$ *seal is opened.*

The regularity read is determined by F, S, R, D, H, T, P, N. It does not choose an infinite branch, identify quotient streams, select a limit, use countable choice, appeal to metric completeness, or import ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the finite prefix schedule F and its StreamName window S. A finite extension supplies more displayed reads but remains a finite BHist row.

Use the RegSeqRat readback row R and the DyadicRatCore tolerance ledger D on the combined finite window. The common window has the same regular rational values and tolerance rows needed by the downstream seal.

Finally transport and replay through H and T, with provenance and local naming recorded by P and N. The argument never leaves the finite packet, so it cannot introduce a selected branch, quotient stream equality, a chosen limit, countable choice, metric completeness, or ambient Real completeness. □

Theorem 6318.4 (ChoiceSequenceReal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing or Completion-facing read of an accepted* $\mathsf{ChoiceSequenceReal}^{\uparrow}$ *packet factors through the finite prefix schedule, StreamName window, RegSeqRat readback, DyadicRatCore tolerance ledger, structural transport, replay, provenance, and local NameCert rows displayed by the carrier. No consumer can bypass those rows by appealing to a selected limit, countable choice, quotient stream equality, trichotomy, metric completeness, bridge row, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Apply Definition 6318.2. It identifies the only local certificate rows available to a Real or Completion consumer.

Then apply Definition 6318.3. Finite prefix extensions and their regular readbacks remain inside the displayed StreamName, RegSeqRat, and DyadicRatCore route before the Real seal is consumed.

The sibling surfaces provide the boundary of the route: chapter 126 supplies finite stream windows, chapter 301 supplies regular rational readback, chapter 291 supplies dyadic tolerance rows, chapter 38 supplies the terminal seal, and chapter 1406 records the compatible nonselection discipline. Hence the packet exports no bypass through the excluded global objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{ChoiceSequenceReal}\uparrow}$.

Constructive story: A $\mathsf{ChoiceSequenceReal}^{\uparrow}$ packet is generated from a finite prefix schedule, StreamName finite windows, RegSeqRat regular readback, DyadicRatCore tolerance rows, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceReal}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6318.1, Definition 6318.2, Definition 6318.3, and Definition 6318.4 seed the finite-prefix real-name packet for the open L10 Real and Completion route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ChoiceSequenceReal}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected limits, countable choice, quotient stream equality, trichotomy, metric completeness, bridge checking, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, finite-prefix regularity, terminal Real-seal nonescape, transport stability, replay exactness, and local NameCert replay for BEDC.Derived.ChoiceSequenceRealUp.

Chapter 6319

A Concrete Naming Certificate for $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$

The $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ packet realizes the shell-level reading of chapter 9463 as finite boundary-shadow, visual-boundary, phase-distance, transport, provenance, and NameCert rows. It unifies the boundary-shadow route of chapter 3813, the visual boundary route of chapter 4181, the phase-distance observer-expansion route of chapter 6017, the observer phase groupoid of chapter 6146, and the Poincare disk metric boundary of chapter 6209. The shell is a boundary-facing packet, not a physical cosmology and not a center-inside-disk assertion.

Definition 6319.1 (Hyperbolic observer phase shell carrier)**.** *A* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (B, V, D, G, M, H, T, P, N)$$

with the following displayed rows:

1. *B is the boundary-shadow row, recording the finite far-end shadow data through which observer expansion is read at the boundary rather than at an interior center.*
2. *V is the visual-boundary row, recording the finite visual-boundary readback attached to the displayed boundary shadow.*
3. *D is the phase-distance row, recording the finite distance comparison used by observer expansion before any global hyperbolic geometry is invoked.*
4. *G is the observer phase groupoid row, recording source, target, inverse, and composition data for the reversible phase-surface transport visible in the shell.*
5. *M is the Poincare-disk metric row, recording only the finite disk metric comparisons needed by B, V, D, G.*
6. *H records componentwise* hsame *transport for boundary-shadow, visual-boundary, phase-distance, groupoid, disk-metric, replay, provenance, and local naming rows.*
7. *T records displayed* Cont *replay from boundary shadows to visual-boundary reads, from visual-boundary reads to phase-distance comparisons, from phase-distance comparisons to observer phase transport, and from those rows to local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$*, hyperbolic observer expansion, hyperbolic visual boundary, phase-distance observer expansion, observer phase groupoid, Poincare disk metric,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}}$ *row.*
The classifier compares accepted shells only through $B, V, D, G, M, H, T, P, N$. It has no coordinate for an interior center of the far end, physical cosmology, host global hyperbolic space, untracked geodesic completion, or ambient real completeness.

Theorem 6319.2 (Hyperbolic observer phase shell NameCert obligations)**.** *For every accepted* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ *carrier $S = (B, V, D, G, M, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}}$ *surface is exhausted by boundary-shadow admission, visual-boundary readback, phase-distance comparison, observer phase groupoid transport, Poincare-disk metric comparison, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every shell read factors through*

$$B, V, D, G, M \longmapsto H, T, P, N.$$

Proof. Project the accepted shell carrier. The boundary-shadow, visual-boundary, phase-distance, observer phase groupoid, and disk-metric rows are the only geometric rows visible to the local certificate.

The structural rows preserve that visible route. Componentwise hsame transport can move only displayed coordinates, and displayed Cont replay can traverse only the listed route from boundary shadow through local naming.

The provenance and local naming rows record the same finite packet. Since no other row is present, a shell consumer receives no interior center, physical-cosmology datum, host global hyperbolic space, untracked geodesic completion, or ambient real-completeness principle. □

Theorem 6319.3 (Hyperbolic observer phase shell boundary nonescape)**.** *Any observer-expansion read of an accepted* $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ *packet is boundary-facing: it must read the boundary-shadow row, visual-boundary row, phase-distance row, observer phase groupoid row, Poincare-disk metric row, transport, replay, provenance, and local naming before it may use the shell as phase geometry evidence. The read cannot move the far end into an interior disk center, interpret the shell as physical cosmology, or appeal to host global geometry, untracked completion, or ambient real completeness.*

Proof. Use Definition 6319.2 to expose the shell route. The only admissible reads are the rows $B, V, D, G, M, H, T, P, N$.

The boundary-shadow row B and visual-boundary row V force the read to remain boundary-facing. The phase-distance and groupoid rows D, G may compare and transport observer phases, but they do so through displayed finite rows rather than through a hidden global hyperbolic object.

The disk-metric row M supplies only the local Poincare metric comparisons named by the packet. Transport, replay, provenance, and local naming preserve that packet, so no excluded center, cosmology, completion, or ambient real-completeness object can enter the read. □

Falsifiable prediction: Every shell read of an accepted $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ packet factors through boundary-shadow, visual-boundary, phase-distance, transport, provenance, and local $\mathsf{NameCert}$ rows before any observer-expansion conclusion is consumed.

Independence witness: hyperbolic-observer-expansion, hyperbolic-visual-boundary, phase-distance-observer-expansion, observer-phase-groupoid, poincare-disk-metric: $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ is not bijective to any listed sibling because it uniquely names their boundary-shadow, visual-boundary, phase-distance, transport, provenance, and local NameCert rows as one shell-level carrier.

The later TasteGate file for this shell must require Lean ‘Nontrivial‘ and ‘FieldFaithful‘ instances before any positive discovery or classifier-shift gate is recorded. Without those instances, the shell remains a finite NameCert packet and not a field-sensitive discovery claim.

Closure status for $\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseShell}\uparrow}$.

Constructive story: A $\mathsf{HyperbolicObserverPhaseShell}^{\uparrow}$ packet is generated from boundary-shadow rows, visual-boundary rows, phase-distance rows, observer phase groupoid rows, Poincare-disk metric rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseShell}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6319.1, Definition 6319.2, and Definition 6319.3 seed the shell-level boundary-facing observer-expansion packet.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HyperbolicObserverPhaseShell}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim an interior far-end center, physical cosmology, host global hyperbolic geometry, untracked geodesic completion, Lean verification, bridge checking, or ambient real completeness.

Upgrade path: Obligation closure requires checked carrier admission, boundary-shadow admission, visual-boundary readback, phase-distance comparison, observer phase groupoid transport, disk-metric comparison, transport stability, replay exactness, and local NameCert replay for BEDC.Derived.HyperbolicObserverPhaseShellUp.

Chapter 6320

A Concrete Naming Certificate for $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ packet records the finite diagonal route by which dyadic endpoint data becomes a regular-Cauchy name before any $\mathsf{Real}^{\uparrow}$ consumer may read it as a Real-facing presentation. It sits between the dyadic arithmetic core of chapter 291, the finite window surface of chapter 126, the regular-rational readback route of chapter 301, and the terminal Real naming packet of chapter 38. The certificate is a displayed embedding route for constant and diagonal Cauchy names; it is not a quotient-sequence equality, selected limit, ambient completeness theorem, or host Real construction.

Definition 6320.1 (Regular Cauchy diagonal embedding carrier)**.** *A* $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, S, Q, R, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint row, including the dyadic source value and the dyadic tolerance ledger inspected by the diagonal route.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *window row recording the finite windows at which the diagonal name is read.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row recording the regular-rational sequence obtained from the dyadic endpoint and window data.*
4. *R is the* $\mathsf{Real}^{\uparrow}$ *consumer row, available only as a named endpoint read of the displayed regular-Cauchy presentation.*
5. *H records componentwise* hsame *transport for the dyadic endpoint, stream windows, regular-rational readback, Real consumer row, and local naming row.*
6. *C records displayed* Cont *replay from dyadic endpoint data through stream windows, regular-rational readback, the Real consumer row, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through D, S, Q, R, H, C, P, N. It has no coordinate for quotient sequence equality, selected representatives, an ambient Real limit, countable choice, or host equality of completed Real values.

Theorem 6320.2 (Regular Cauchy diagonal embedding NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ *carrier* $E = (D, S, Q, R, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagona}}$ *surface consists exactly of carrier admission for the dyadic endpoint row, stream-window row, regular-rational readback row, Real consumer row, componentwise transport, displayed replay, provenance, and local naming. Every Real-facing read must expose D, S, Q, H, C, P, N before it may consume R as the public endpoint row.*

Proof. Project the carrier of Definition 6320.1. The source row is D, the finite observation row is S, the regular-Cauchy readback row is Q, and the endpoint consumer row is R.

The naming surface orders those rows. A Real-facing read starts from the dyadic endpoint and tolerance ledger, then opens the StreamName windows, then reads the RegSeqRat presentation produced by the diagonal route.

The structural rows H, C, P, N preserve this same route under transport, replay, provenance, and local naming. Since the carrier contains no quotient sequence equality, selected representative, ambient limit, countable-choice schedule, or host equality coordinate, the listed rows exhaust the local certificate obligations. □

Theorem 6320.3 (Regular Cauchy diagonal embedding constant-sequence stability)**.** *If two accepted* $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ *packets have classifier-equivalent dyadic endpoint rows and are read through the same* $\mathsf{StreamName}^{\uparrow}$ *windows, then their diagonal* $\mathsf{RegSeqRat}^{\uparrow}$ *readback rows remain componentwise* hsame *before the shared* $\mathsf{Real}^{\uparrow}$ *consumer row is public. Thus a constant dyadic endpoint cannot be separated by the diagonal embedding unless the displayed dyadic endpoint, stream-window, or regular-rational readback row is separated.*

Proof. Read both packets through Definition 6320.1. Classifier equivalence fixes the dyadic endpoint row D and the shared windows fix the observation row S.

The regular-rational readback row Q is downstream of exactly those two rows. By Definition 6320.2, a Real-facing read cannot inspect R before the dyadic endpoint, stream windows, regular readback, transport, replay, provenance, and local naming rows have been displayed.

The componentwise hsame row H therefore transports the equality of the dyadic and window data to the regular-rational readback. Any separation of the constant diagonal presentation must be visible in D, S, or Q, not in a hidden quotient, selected representative, ambient Real limit, or host equality. □

Falsifiable prediction: Every accepted $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ read exposes the $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any $\mathsf{Real}^{\uparrow}$ endpoint row is public.

Independence witness: dyadicratcore, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ because the packet jointly binds the dyadic endpoint, finite windows, diagonal regular-rational readback, Real consumer row, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyDiagonalEmbedding}^{\uparrow}$ packet is generated from a DyadicRatCore endpoint row, StreamName windows, RegSeqRat diagonal readback, a Real consumer row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{RegularCauchyDiagonalEmbedding}^{\uparrow}}$, seed)

Scope closed: Definition 6320.1, Definition 6320.2, and Definition 6320.3 seed the finite diagonal route from dyadic endpoints through stream windows and regular-rational readback to Real-facing consumers.

Formal verification: FormalStatus(NameCert$_{\text{RegularCauchyDiagonalEmbedding}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient sequence equality, selected representatives, ambient Real completeness, selected limits, countable choice, Lean verification, or bridge checking for RegularCauchyDiagonalEmbedding$^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, diagonal readback exactness, constant-sequence stability, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyDiagonalEmbeddingUp.

Chapter 6321

A Concrete Naming Certificate for $\mathsf{MetaCICResidualContextCut}^{\uparrow}$

The $\mathsf{MetaCICResidualContextCut}^{\uparrow}$ packet realizes the subject-reduction obstruction of chapter 9377 as finite typed-context, residual-cut, candidate-boundary, and provenance rows. It sits beside the finite-window normalization audit of chapter 6314, the MetaCIC critical path of chapter 765, and the subject-reduction discharge packets already attached to the obstruction map. The packet names a boundary where a reduction residual must preserve enough context data before a candidate-normalization or subject-reduction consumer may proceed.

Definition 6321.1 (MetaCIC residual-context cut carrier)**.** *A* $\mathsf{MetaCICResidualContextCut}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (G, R, B, A, O, H, C, P, N)$$

with the following displayed rows:

1. *G is the typed-context row, recording the inspected context, binder boundary, closedness window, and typing judgement visible to the packet.*
2. *R is the residual-cut row, recording the beta or congruence residual whose source and target must remain attached to G.*
3. *B is the candidate-boundary row, recording the candidate-normalization evidence or refusal boundary available at the inspected residual.*
4. *A is the subject-reduction obstruction row, naming the four discharge obligations and the checked obstruction boundary read from chapter 9377.*
5. *O is the observation ledger, recording which typed-context, residual, candidate, and obstruction fields are public to downstream reads.*
6. *H records componentwise* hsame *transport for G, R, B, A, O and the local naming row.*
7. *C records displayed* Cont *replay from typed-context inspection through residual cut, candidate boundary, obstruction row, observation ledger, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{MetaCICResidualContextCut}^{\uparrow}$*, typed-context rows, residual-cut rows, candidate-normalization boundaries, subject-reduction obstruction rows,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualContextCut}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $G, R, B, A, O, H, C, P, N$. It has no coordinate for an unconditional closed subject-reduction theorem, hidden substitution lemma, unrestricted normalization theorem, host equality of terms, or global conversion oracle.

Theorem 6321.2 (MetaCIC residual-context cut NameCert obligations)**.** *For every accepted* $\mathsf{MetaCICResidualContextCut}^{\uparrow}$ *carrier* $K = (G, R, B, A, O, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{MetaCICResidualContextCut}^{\uparrow}}$ *surface consists exactly of carrier admission for the typed-context row, residual-cut row, candidate-boundary row, subject-reduction obstruction row, observation ledger, componentwise transport, displayed replay, provenance, and local naming. Every accepted read factors through* $G, R, B, A, O, H, C, P, N$ *before it may cite the residual as a subject-reduction or normalization boundary.*

Proof. Project the carrier of Definition 6321.1. The typed-context row G fixes the visible context and judgement, and the residual-cut row R fixes the source-target residual attached to that context.

The candidate-boundary row B and obstruction row A are read next. They attach the residual to candidate-normalization evidence and to the four-obligation subject-reduction interface, without turning either surface into a hidden global theorem.

The observation ledger O records which fields are public. The structural rows H, C, P, N then transport, replay, source, and name exactly the displayed route, so no unconditional closed subject-reduction theorem, hidden substitution lemma, unrestricted normalization theorem, host term equality, or conversion oracle can be exported. □

Theorem 6321.3 (MetaCIC residual-context cut subject-reduction obstruction)**.** *Every* MetaCICResidualContextC *packet that is accepted as a subject-reduction boundary exposes the route*

$$G \longmapsto R \longmapsto B, A \longmapsto O \longmapsto H, C, P, N.$$

Consequently the packet may name a residual-context obstruction, but it cannot supply an unconditional closed subject-reduction theorem unless the typed-context, residual-cut, candidate-boundary, and obstruction rows are replaced by checked discharge data outside this packet.

Proof. Begin with an accepted subject-reduction boundary read. The carrier definition supplies the typed-context row G and the residual-cut row R as the only source of local context and reduction data.

Apply Definition 6321.2. The candidate-boundary row B and subject-reduction obstruction row A must be displayed before the observation ledger or any downstream consumer can read the packet.

The route closes through O, H, C, P, N. These rows expose, transport, replay, source, and name the boundary, but none of them contains the missing four discharge obligations as checked data. The packet therefore records the obstruction boundary rather than converting it into a closed subject-reduction theorem. □

Falsifiable prediction: Every accepted $\mathsf{MetaCICResidualContextCut}^{\uparrow}$ read factors through typed-context, residual-cut, candidate-boundary, hsame, Cont, Pkg, and NameCert rows before any subject-reduction or normalization-boundary conclusion is public.

Independence witness: metacic-confluence-critical-pair-boundary, metacic-normalization-window-confluence, metacic-candidate-set-normalization, closed-pi-domain-subject-reduction: none is bijective with $\mathsf{MetaCICResidualContextCut}^{\uparrow}$ because this packet jointly binds the typed-context row, residual-cut row, candidate-boundary row, subject-reduction obstruction row, observation ledger, transport, replay, provenance, and local NameCert.

The Lean TasteGate surface for this packet must later provide explicit Nontrivial MetaCICResidualContextCut$^{\uparrow}$ and FieldFaithful MetaCICResidualContextCut$^{\uparrow}$ instances in $BEDC.Derived.MetaCICResidualContextCutUp.Ta$ preserving $G, R, B, A, O, H, C, P, N$ as distinct displayed rows before a checked target may cite the packet as a verified carrier-alignment theorem.

Closure status for NameCert$_{\text{MetaCICResidualContextCut}^{\uparrow}}$.

Constructive story: A MetaCICResidualContextCut$^{\uparrow}$ packet is generated from a typed-context row, residual-cut row, candidate-boundary row, subject-reduction obstruction row, observation ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate(NameCert$_{\text{MetaCICResidualContextCut}^{\uparrow}}$, seed)

Scope closed: Definition 6321.1, Definition 6321.2, and Definition 6321.3 seed the finite residual-context obstruction route over typed contexts, residual cuts, candidate boundaries, observation ledgers, transport, replay, provenance, and local naming.

Formal verification: FormalStatus(NameCert$_{\text{MetaCICResidualContextCut}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close unconditional closed subject reduction, beta-substitution preservation, application-argument stability, lambda-domain preservation, Pi-domain preservation, global normalization, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, typed-context exactness, residual-cut exactness, candidate-boundary exposure, obstruction-row exposure, observation-ledger replay, and Nontrivial and FieldFaithful instances for BEDC.Derived.MetaCICResidualContextCutUp.

Chapter 6322

A Concrete Naming Certificate for $\mathsf{OpialCondition}^{\uparrow}$

Following Opial's fixed-point tradition, the $\mathsf{OpialCondition}^{\uparrow}$ packet records the finite certificate surface by which a bounded sequence window, a candidate weak-limit row, and a limsup-distance comparison are made available to nonexpansive fixed-point consumers without importing weak compactness, selected weak limits, or countable choice. It depends on the metric carrier of chapter 57, the Banach carrier of **??**, the regular Cauchy nonexpansive transport route of chapter 2105, and the L10 sequence/readback route represented by chapter 6315 and chapter 1413.

Definition 6322.1 (Opial condition carrier)**.** *An* $\mathsf{OpialCondition}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (M, B, S, W, L, D, F, R, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row on which distance readings are measured.*
2. *B is the ambient* $\mathsf{BanachSpace}^{\uparrow}$ *row, exposing only the linear and norm data needed to read the same metric distances through the Banach carrier.*
3. *S is the bounded sequence row, displayed through the L10 sequence/readback route rather than through an infinite choice object.*
4. *W is the finite bounded-sequence window opened around the requested tail and tolerance.*
5. *L is the candidate limit row. It is a named point read against M and B, not a selected weak limit supplied by a compactness principle.*
6. *D is the displayed limsup-distance comparison ledger*

$$\limsup_n d(S_n, L) < \limsup_n d(S_n, Y)$$

   for each displayed competing point row Y whose carrier classifier is distinct from L.

7. *F is the nonexpansive-map handoff row. It records that a* $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ *consumer may read S, W, L, and D when checking the fixed-point boundary for a displayed map, but may not manufacture a limit or a compactness witness.*
8. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *readback row for the distance values used in D.*

9. *H records componentwise* hsame *transport for the metric row, Banach row, sequence row, finite window, candidate limit, limsup ledger, nonexpansive handoff, readback rows, continuation replay, provenance, and local naming row.*

10. *C records displayed* Cont *replay from each finite fixed-point request through*

$$M, B, S, W, L, D, F, R \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{OpialCondition}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{BanachSpace}^{\uparrow}$, $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{OpialCondition}^{\uparrow}}$ *row.*

The carrier has no coordinate for weak compactness, a selected weak cluster point, quotient stream equality, excluded middle, countable choice, a demiclosedness theorem, or an ambient fixed-point theorem.

Theorem 6322.2 (Opial condition NameCert obligations)**.** *For every accepted* $\mathsf{OpialCondition}^{\uparrow}$ *carrier*

$$O = (M, B, S, W, L, D, F, R, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{OpialCondition}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the metric row, Banach row, bounded sequence row, finite window, candidate limit, limsup-distance ledger, nonexpansive handoff, rational and Real readback, transport, replay, provenance, and local naming rows;*

2. *sequence exposure, requiring every limsup-distance read to begin with the displayed L10 sequence/readback route and the finite bounded window W;*

3. *candidate discipline, requiring L to be a displayed point row and forbidding replacement of L by a selected weak cluster point;*

4. *limsup comparison exactness, requiring D to display the strict comparison between the distance tail at L and each displayed competing row Y distinct from L;*

5. *nonexpansive handoff, requiring F to export only the rows needed by a* $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$ *fixed-point-boundary consumer;*

6. *provenance and local naming, requiring H, C, P, N to transport, replay, source, and name only the displayed Opial route.*

No $\mathsf{OpialCondition}^{\uparrow}$ *obligation is stored outside these rows.*

Proof. Carrier admission is projection from Definition 6322.1. The accepted packet displays exactly the metric, Banach, sequence, finite-window, candidate-limit, comparison, handoff, readback, transport, replay, provenance, and naming coordinates.

The sequence and candidate obligations are forced by S, W, L. The sequence is opened only through the finite L10 readback route, and the limit candidate is a displayed point row, so the certificate has no position from which to extract a selected weak cluster point.

The comparison obligation is the role of D. Each accepted competing row Y is first displayed as distinct from L, and then the limsup-distance inequality is read through the rational and Real readback row R.

Finally F, H, C, P, N close the route. The nonexpansive handoff exports the visible rows to fixed-point-boundary consumers, while the structural rows preserve only the displayed coordinates

and leave no carrier position for weak compactness, demiclosedness, countable choice, or an ambient fixed-point theorem. □

Theorem 6322.3 (Opial condition nonexpansive handoff)**.** *Every* $\mathsf{RegularCauchyNonexpansiveMap}^{\uparrow}$*-facing consumer read accepted by an* $\mathsf{OpialCondition}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{BanachSpace}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{RegularCauchyNonexpansiveMap}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Equivalently, an accepted fixed-point-boundary read may consume the displayed bounded sequence window, candidate limit row, limsup-distance comparison, and nonexpansive handoff, but it cannot bypass them by weak compactness, a selected weak limit, demiclosedness, countable choice, or an ambient fixed-point theorem.

Proof. Use the obligation theorem Definition 6322.2. It exposes a nonexpansive handoff only after the metric and Banach rows, the bounded sequence row, the finite window, the candidate limit row, the limsup-distance comparison ledger, and the rational and Real readback row have all been displayed.

The displayed Cont route then replays each fixed-point-boundary request through the same rows. Thus a nonexpansive-map consumer reads the Opial comparison as finite certificate data rather than as a compactness principle.

Inspecting the carrier definition gives the exclusion. There is no coordinate for weak compactness, selected weak limits, demiclosedness, countable choice, or a detached fixed-point theorem, so any consumer requiring one of those objects is outside the local classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{OpialCondition}^{\uparrow}}$**.**

Constructive story: An $\mathsf{OpialCondition}^{\uparrow}$ packet is generated from metric and Banach rows, a bounded sequence row, a finite bounded-sequence window, a displayed candidate limit, a limsup-distance comparison ledger, a nonexpansive-map handoff, RegSeqRat and Real readback, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{OpialCondition}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6322.1, Definition 6322.2, and Definition 6322.3 seed the finite Opial-condition route from displayed bounded sequence data through limsup-distance comparison to nonexpansive fixed-point-boundary consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{OpialCondition}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim weak compactness, selected weak limits, demiclosedness, countable choice, an ambient fixed-point theorem, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, limsup-distance comparison exactness, nonexpansive handoff stability, and BEDC.Derived.OpialConditionUp formalization.

Chapter 6323

A Concrete Naming Certificate for $\mathsf{FiniteIntersectionCompactness}^{\uparrow}$

Following Bishop and Bridges' constructive compactness practice and Kelley's finite-intersection formulation of compactness, the $\mathsf{FiniteIntersectionCompactness}^{\uparrow}$ packet records the finite-window compactness data that a completion corridor may inspect without importing a general open-cover theorem. It sits beside the finite-intersection property row of chapter 1792, the compact metric certificate of chapter 129, the closed-interval Cauchy completion row of chapter 1778, the finite epsilon-net row of chapter 1767, and the Real seal of chapter 38. Those neighbors provide the traditional compactness vocabulary; this certificate keeps the usable claim finite, row-displayed, and local to the BEDC NameCert surface.

Definition 6323.1 (Finite-intersection compactness carrier)**.** *A* $\mathsf{FiniteIntersectionCompactness}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (I, K, J, E, R, H, C, P, N)$$

with the following displayed rows:

1. *I is the finite-intersection-property row read from chapter 1792;*
2. *K is the compact-metric source row read from chapter 129;*
3. *J is the closed-interval Cauchy-completion window read from chapter 1778;*
4. *E is the finite epsilon-net witness row read from chapter 1767;*
5. *R is the terminal Real-facing seal read from chapter 38;*
6. *H stores componentwise* hsame *transport for finite-intersection rows, compact-metric rows, closed-interval completion windows, finite epsilon-net witnesses, Real seals, provenance, and local naming;*
7. *C stores displayed* Cont *replay for the same finite-window route;*
8. *P records* Pkg *provenance over* $\mathsf{FiniteIntersectionProperty}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ClosedIntervalCauchyCompletion}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *data;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteIntersectionCompactness}^{\uparrow}}$ *row.*

The carrier records only finite-window compactness data. It contains no coordinate for general open-cover compactness, countable choice, quotient real equality, trichotomy, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6323.2 (Finite-intersection compactness NameCert obligations)**.** *For the carrier of Definition 6323.1, the local* $\mathsf{NameCert}_{\mathsf{FiniteIntersectionCompactness}^{\uparrow}}$ *obligation surface consists exactly of the finite-intersection-property row* I*, compact-metric source row* K*, closed-interval Cauchy-completion window* J*, finite epsilon-net witness row* E*, Real seal row* R*, componentwise transport* H*, continuation replay* C*, provenance* P*, and local naming* N*. A consumer may cite the packet only as finite-window compactness data over those rows.*

Proof. Project the carrier tuple $F = (I, K, J, E, R, H, C, P, N)$. The finite-intersection, compact-metric, closed-interval completion, finite epsilon-net, and Real-facing rows are displayed as separate coordinates before any compactness read is available.

Use chapter 1792 for the finite-intersection row and chapter 129 for the compact-metric source. These rows expose finite compatibility and the metric carrier without producing a general open-cover theorem.

Use chapter 1778, chapter 1767, and chapter 38 for the completion window, finite search net, and terminal seal. They keep the read inside finite windows and Real-facing certificates rather than quotient equality or ambient completeness.

The rows H, C, P, N only transport, replay, cite, and name that same finite route. Since no coordinate stores countable choice, trichotomy, quotient real equality, ambient Real completeness, or open-cover compactness, none is exported by the NameCert obligations. □

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteIntersectionCompactness}^{\uparrow}}$.

Constructive story: A $\mathsf{FiniteIntersectionCompactness}^{\uparrow}$ packet is generated from finite-intersection-property data, compact-metric source rows, closed-interval Cauchy-completion windows, finite epsilon-net witnesses, Real-facing seals, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FiniteIntersectionCompactness}^{\uparrow}}$, seed)

Scope closed: Definition 6323.1 and Definition 6323.2 seed finite-window compactness data over finite-intersection, compact-metric, closed-interval completion, finite epsilon-net, and Real-facing rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FiniteIntersectionCompactness}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim general open-cover compactness, countable choice, quotient real equality, trichotomy, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and finite-window compactness obligations for BEDC.Derived.FiniteIntersectionCompactnessUp.

Chapter 6324

A Concrete Naming Certificate for $\mathsf{MichaelLine}^{\uparrow}$

Following Michael and Kelley topology, the $\mathsf{MichaelLine}^{\uparrow}$ packet records the classical Michael-line presentation as a finite topology-adjacent NameCert object. It reads the real-order topology rows of chapter 1890, the indexed open-row discipline of chapter 88, and the observable-open truth classifier of chapter 1860. The packet names only the displayed real-order, distinguished-row, isolated-point, and observable-open handoff data needed by topology consumers; it does not import a host powerset, a choice-selected irrational subset, a quotient point equality, or an ambient completeness theorem for the real line.

Definition 6324.1 (Michael line carrier)**.** *A* $\mathsf{MichaelLine}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (R, T, D, O, S, L, H, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{RealOrderTopology}^{\uparrow}$ *source row, carrying the named real-order intervals and endpoint comparisons available from chapter 1890.*
2. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, exposing indexed open rows, finite neighbourhood membership reads, and topology closure witnesses from chapter 88.*
3. *D is the distinguished-row ledger marking the points whose singleton neighbourhood row is admitted by the Michael-line presentation.*
4. *O is the ordinary real-order open row used for points outside the distinguished ledger.*
5. *S is the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open truth row, recording whether the currently inspected named point is read through the ordinary open row or through the isolated singleton row.*
6. *L is the topology handoff row that exports the ordinary and isolated neighbourhood reads back to the indexed open surface of T.*
7. *H records componentwise* hsame *transport for the real-order source, topology source, distinguished ledger, ordinary open row, observable truth row, handoff row, continuation row, provenance row, and local naming row.*
8. *C records displayed* Cont *replay from point inspection to distinguished-row lookup, from lookup to observable-open truth, and from observable truth to the topology handoff.*

9. *P is the* Pkg *provenance over* $\mathsf{MichaelLine}^{\uparrow}$, $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{SierpinskiSpace}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{MichaelLine}^{\uparrow}}$ *row.*

The local classifier compares accepted packets only through $R, T, D, O, S, L, H, C, P, N$. It has no coordinate for an ambient powerset of reals, a selected subset outside the distinguished ledger, quotient equality of points, open-cover compactness, real completeness, or a standard bridge.

Theorem 6324.2 (Michael line NameCert obligations)**.** *For an accepted* $\mathsf{MichaelLine}^{\uparrow}$ *carrier $M = (R, T, D, O, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{MichaelLine}^{\uparrow}}$ *surface has five obligations:*

1. *carrier exposure for the displayed tuple $R, T, D, O, S, L, H, C, P, N$;*
2. *real-order source discipline, requiring ordinary neighbourhood reads to be supplied by R and its interval rows;*
3. *distinguished-row discipline, requiring isolated singleton reads to pass through the finite ledger D;*
4. *observable-open truth, requiring the choice between ordinary and isolated reads to be witnessed by the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *row S;*
5. *topology handoff and non-escape, requiring all exported open rows to pass through L, H, C, P, N without host powersets, selected subsets, quotient point equality, open-cover compactness, real completeness, or a standard bridge.*

Proof. Carrier exposure is projection from Definition 6324.1. The real-order source R and topology source T are the only ambient sibling rows available to the packet.

For an ordinary point read, the open neighbourhood is supplied by the real-order row R and then routed through the topology handoff L. For an isolated read, the distinguished ledger D must first display the singleton neighbourhood row.

The $\mathsf{SierpinskiSpace}^{\uparrow}$ row S records the observable-open truth value separating those two routes. By chapter 1860, this is a finite classifier read rather than a host-level predicate over all subsets of reals.

Finally H, C, P, N transport, replay, source, and name the same handoff. Since the carrier has no coordinate for host powersets, selected subsets, quotient point equality, open-cover compactness, real completeness, or bridge data, the five obligations exhaust the local NameCert surface. □

Theorem 6324.3 (Michael line Sierpinski topology handoff)**.** *Every accepted* $\mathsf{MichaelLine}^{\uparrow}$ *topology read reaches the public* $\mathsf{Topology}^{\uparrow}$ *surface only after the* $\mathsf{SierpinskiSpace}^{\uparrow}$ *observable-open row has selected one of the displayed real-order or isolated-neighbourhood routes. For*

$$M = (R, T, D, O, S, L, H, C, P, N),$$

the exported open row is computed through

$$R, D, O, S, H, C, P, N \longrightarrow L \longrightarrow T.$$

No host powerset, choice-selected distinguished subset, quotient point equality, open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ *completeness, Lean verification, or standard bridge supplies the handoff.*

Proof. Project the carrier rows from Definition 6324.1. The only routes to a neighbourhood read are the ordinary real-order row O sourced by R and the isolated singleton route sourced by the distinguished ledger D.

The observable-open row S is read before the topology handoff. It records which finite route is visible for the inspected point and gives the handoff a displayed $\mathsf{SierpinskiSpace}^{\uparrow}$ truth row instead of an ambient subset predicate.

The row L then exports the selected route to the indexed open surface T of chapter 88. Componentwise transport through H and continuation replay through C preserve the same ordinary-or-isolated decision, while P and N record provenance and local naming.

Thus the handoff is a finite BEDC route through the sibling certificates chapter 1890, chapter 1860, and chapter 88. The excluded host objects have no carrier coordinate, so they cannot supply topology data to $\mathsf{MichaelLine}^{\uparrow}$. □

Closure status for $\mathsf{NameCert}_{\mathsf{MichaelLine}^{\uparrow}}$.

Constructive story: A $\mathsf{MichaelLine}^{\uparrow}$ packet is generated from displayed real-order topology rows, indexed topology source rows, distinguished singleton rows, observable-open truth, topology handoff, componentwise transport, continuation replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{MichaelLine}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6324.1, Definition 6324.2, and Definition 6324.3 seed the finite Michael-line boundary over $\mathsf{RealOrderTopology}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, and $\mathsf{SierpinskiSpace}^{\uparrow}$.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{MichaelLine}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host powersets, choice-selected subsets, open-cover compactness, quotient point equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or standard bridges.

Upgrade path: Obligation closure requires checked carrier admission, distinguished-row membership, observable-open truth, topology handoff, transport, replay, provenance, and no-escape for BEDC.Derived.MichaelLineUp.

Chapter 6325

A Concrete Naming Certificate for $\mathsf{AssouadDimension}^{\uparrow}$

Following Assouad and Heinonen, the $\mathsf{AssouadDimension}^{\uparrow}$ packet records the finite metric-dimension surface by which a metric source, a scale ratio, and a uniformly bounded local cover become inspectable NameCert data. It sits over the metric carrier of chapter 57, reads local doubling cover rows from chapter 2035, routes radius and exponent comparisons through chapter 38, and exposes cover-refinement data to the covering-dimension sibling chapter 1853. The packet names a finite scale-cover ledger; it does not assert a host infimum of exponents, choose optimal covers, import Hausdorff measure, or identify metric spaces by ambient equality.

Definition 6325.1 (Assouad dimension carrier)**.** *An* $\mathsf{AssouadDimension}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (X, M, D, R, S, E, B, K, Q, H, C, P, N)$$

with the following displayed rows:

1. *X is the underlying point-history carrier and classifier row.*
2. *M is the inherited* $\mathsf{Met}^{\uparrow}$ *distance row over X.*
3. *D is the* $\mathsf{DoublingMetric}^{\uparrow}$ *or* $\mathsf{DoublingMetricSpace}^{\uparrow}$ *local cover row, exposing finite covers of displayed balls by smaller displayed balls.*
4. *R is the radius-ratio ledger. It names positive radii $0 < r < R$ through* $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, *or* $\mathsf{DyadicRatCore}^{\uparrow}$ *readback rows and records the comparison needed for the scale ratio.*
5. *S is the exponent-bound row. It stores the displayed rational or real candidate s and the coefficient row used in the Assouad cover estimate.*
6. *E is the local cover-count estimate row, recording that each displayed R-ball admits a finite r-ball cover whose count is bounded by the displayed coefficient and scale exponent.*
7. *B is the finite center-bundle row for the r-ball cover, carried as a closed* $\mathsf{ProbeBundle}$ *over X.*
8. *K is the covering-dimension handoff row. It exposes only the finite cover and refinement data readable by chapter 1853.*
9. *Q is the scale-comparison provenance row connecting D, R, S, E, B, K without adding a host logarithm or an exponent infimum.*

10. *H records componentwise* hsame *transport for the metric row, doubling row, radius ledger, exponent row, cover-count row, center bundle, covering-dimension handoff, provenance row, replay row, package row, and local naming row.*

11. *C records displayed* Cont *replay from each metric-ball cover request through*

$$X, M, D, R, S, E, B, K, Q \longmapsto H, P, N.$$

12. *P is the* Pkg *provenance over* $\mathsf{AssouadDimension}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{DoublingMetricSpace}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.

13. *N is the local* $\mathsf{NameCert}_{\mathsf{AssouadDimension}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for a selected optimal cover, a least exponent, Hausdorff dimension, host logarithms, quotient metric-space equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6325.2 (Assouad dimension NameCert obligations)**.** *For every accepted* $\mathsf{AssouadDimension}^{\uparrow}$ *carrier*

$$A = (X, M, D, R, S, E, B, K, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{AssouadDimension}^{\uparrow}}$ *surface consists exactly of carrier admission for the point and metric rows; local doubling-cover admission through D; positive radius-ratio exposure through R; exponent-bound and coefficient exposure through S; finite cover-count exactness through E; center-bundle admission through B; covering-dimension handoff through K; scale-comparison provenance through Q; and componentwise transport, displayed replay, package provenance, and local naming through H, C, P, N.*

Proof. Project the accepted carrier of Definition 6325.1. The point row X, metric row M, doubling row D, radius-ratio ledger R, exponent-bound row S, cover-count estimate E, center bundle B, and covering-dimension handoff K are all displayed before the local naming row is consumed.

The cover obligation is read from D, E, B. The doubling row opens the relevant metric ball, the center bundle lists the finite r-ball centers, and the cover-count estimate records the displayed Assouad bound for the same radius pair.

The scale obligation is read from R, S, Q. These rows carry positive radii, the candidate exponent, the coefficient, and the comparison route needed by the estimate without adding a host logarithm or a least exponent.

Finally H, C, P, N transport, replay, source, and name the same finite route. Since no row stores an optimal cover selector, Hausdorff-measure comparison, quotient metric equality, countable choice, or ambient real completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6325.3 (Assouad dimension doubling handoff)**.** *Every* $\mathsf{DoublingMetric}^{\uparrow}$, $\mathsf{DoublingMetricSpace}^{\uparrow}$, $\mathsf{CoveringDimension}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{AssouadDimension}^{\uparrow}$ *packet reads the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{DoublingMetric}^{\uparrow},\ \mathsf{DoublingMetricSpace}^{\uparrow},\ \mathsf{CoveringDimension}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{ProbeBundle},\ \mathsf{BHist},\ \mathsf{hsame}$$

through the rows $X, M, D, R, S, E, B, K, Q, H, C, P, N$. In particular, doubling consumers receive only the local ball-cover ledger, covering-dimension consumers receive only the finite cover and refinement handoff, and Real-facing consumers receive radius and exponent reads only after the displayed metric-cover route has been inspected.

Proof. Start with a doubling consumer. The metric row M and local doubling row D expose the parent ball and the smaller-ball cover request, while B records the finite center bundle and E records the displayed cover-count estimate.

Next consider a covering-dimension consumer. It reaches the packet through K, which is downstream of the same finite cover data D, E, B and the scale-comparison provenance Q. Hence it reads refinement-ready cover data rather than a host dimension theorem.

Finally inspect a Real-facing radius or exponent read. The rows R and S provide positive radii, the scale ratio, the candidate exponent, and the coefficient, but only along the route replayed by C and transported by H.

Thus every doubling, covering-dimension, or Real-facing use remains inside the displayed finite route and cannot import selected optimal covers, exponent infima, Hausdorff measure, quotient metric-space equality, countable choice, or ambient real completeness. □

Falsifiable prediction: Every accepted $\mathsf{AssouadDimension}^{\uparrow}$ cover read displays one point-history carrier, one metric-distance row, one local doubling-cover row, one positive radius-ratio ledger, one exponent-bound row, one finite cover-count estimate, one center ProbeBundle, one covering-dimension handoff, scale-comparison provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: metric, doublingmetric, coveringdimension, real: $\mathsf{AssouadDimension}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source, radius-ratio ledger, exponent-bound row, finite local cover-count estimate, center bundle, covering-dimension handoff, and Real-facing scale route in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{AssouadDimension}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AssouadDimension}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6325.1, Definition 6325.2, and Definition 6325.3 seed the finite Assouad dimension route over metric, doubling-metric, covering-dimension, and Real-facing rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AssouadDimension}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim optimal cover selection, a least exponent, Hausdorff dimension, host logarithm analysis, quotient metric-space equality, countable choice, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, radius-ratio exposure, exponent-bound rows, finite local cover-count exactness, covering-dimension handoff, transport, replay, provenance, and local NameCert rows for BEDC.Derived.AssouadDimensionUp.

Constructive story: An $\mathsf{AssouadDimension}^{\uparrow}$ packet is generated from a point-history carrier, a MetricUp distance row, a local doubling-cover row, positive radius-ratio data, an exponent-bound row, a finite cover-count estimate, a center ProbeBundle, a covering-dimension handoff row, scale-comparison provenance, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6326

A Concrete Naming Certificate for $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$

Following Bishop and Bridges' constructive finite-cover treatment, the $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$ packet records the finite dyadic closed-interval subcover data that interval-cover and uniform-modulus consumers may inspect without importing a general Heine–Borel theorem for arbitrary $\mathsf{Real}^{\uparrow}$ subsets. It sits on the dyadic interval-cover route of chapter 1428, the dyadic rational arithmetic core of chapter 291, the finite scheduling surface of chapter 126, the regular rational readback route of chapter 301, and the terminal Real seal of chapter 38. The certificate is only a finite dyadic closed-interval subcover packet; it does not claim general Heine–Borel compactness for arbitrary $\mathsf{Real}^{\uparrow}$ subsets or arbitrary open covers.

Definition 6326.1 (Heine–Borel finite subcover carrier)**.** *A* $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$H = (I, D, W, R, S, T, C, P, N)$$

with the following displayed rows:

1. *I is the dyadic closed-interval cover row read from chapter 1428;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and mesh-comparison row read from chapter 291;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite scheduling row that indexes the inspected cover windows;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for endpoint, mesh, and inclusion comparisons;*
5. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal for the closed interval endpoints and covered points;*
6. *T records componentwise* hsame *transport for cover rows, dyadic endpoint rows, stream windows, regular rational readback, Real seals, provenance, and local naming;*
7. *C records displayed* Cont *replay from an interval-cover request through I, D, W, R, S to the finite subcover output;*
8. *P records* Pkg *provenance over* $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *data;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteSubcover}^{\uparrow}}$ *row.*

The carrier has no coordinate for arbitrary open covers, powerset compactness, countable choice, excluded middle, quotient Real equality, or ambient completeness.

Theorem 6326.2 (Heine–Borel finite subcover obligation surface)**.** *For the carrier of Definition 6326.1, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteSubcover}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for* $I, D, W, R, S, T, C, P, N$*, cover-window exposure through* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *and* $\mathsf{StreamName}^{\uparrow}$*, endpoint and mesh comparison through* $\mathsf{DyadicRatCore}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$*, terminal interval sealing through* $\mathsf{Real}^{\uparrow}$*, componentwise transport, replay stability, provenance, local naming, and ledger non-escape from non-finite compactness principles.*

Proof. Project the finite packet $H = (I, D, W, R, S, T, C, P, N)$. The dyadic interval-cover row, dyadic rational row, finite stream-window row, regular rational readback row, and Real seal are displayed before a subcover output can be read.

Use chapter 1428 for the cover request and chapter 291 for endpoint arithmetic. These rows keep the cover finite and dyadic rather than an arbitrary open cover of an arbitrary subset.

Use chapter 126, chapter 301, and chapter 38 for scheduling, rational comparison, and the terminal closed-interval seal. The rows T, C, P, N only transport, replay, cite, and name the same finite route. Since no coordinate stores powerset compactness, arbitrary cover choice, quotient Real equality, or ambient completeness, none is exported by the obligation surface. □

Theorem 6326.3 (Heine–Borel finite subcover NameCert route)**.** *Every accepted* $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$ *packet routes a dyadic closed-interval cover request through*

$$\mathsf{DyadicIntervalCover}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

The exported certificate is a finite subcover row for the named closed interval and the displayed dyadic cover windows, not a proof of arbitrary $\mathsf{Real}^{\uparrow}$ *compactness.*

Proof. Start with the carrier admission of Definition 6326.1. It fixes the only readable source rows as I, D, W, R, S and the structural rows as T, C, P, N.

Apply the obligation surface of Definition 6326.2. Cover-window exposure, endpoint comparison, stream scheduling, rational readback, and Real sealing are each projections from the same finite packet.

It remains to identify the exported object. The replay row C can only replay the displayed route, and the provenance row P names exactly the listed dependencies. Thus the consumer obtains a finite dyadic closed-interval subcover certificate and no general compactness principle. □

Falsifiable prediction: Every $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$ consumer read factors through a displayed $\mathsf{DyadicIntervalCover}^{\uparrow}$ cover row, a $\mathsf{DyadicRatCore}^{\uparrow}$ endpoint row, a $\mathsf{StreamName}^{\uparrow}$ finite window, a $\mathsf{RegSeqRat}^{\uparrow}$ comparison row, a terminal $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dyadicintervalcover, dyadicratcore, streamname, regseqrat, real: $\mathsf{HeineBorelFiniteSubcov}$ is not bijective to any listed sibling because it jointly carries the finite dyadic cover request, endpoint arithmetic, scheduling window, rational readback, Real seal, transport, replay, provenance, and local NameCert rows as one finite subcover packet.

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteSubcover}^{\uparrow}}$.

Constructive story: A $\mathsf{HeineBorelFiniteSubcover}^{\uparrow}$ packet is generated from a dyadic closed-interval cover row, dyadic endpoint arithmetic, finite StreamName windows, RegSeqRat readbacks, a terminal Real seal, componentwise transport, continuation replay, provenance, and local naming.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteSubcover}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6326.1, Definition 6326.2, and Definition 6326.3 seed the finite dyadic closed-interval subcover certificate surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteSubcover}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, powerset compactness, countable choice, excluded middle, quotient Real equality, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission and NameCert obligation theorems for BEDC.Derived.HeineBorelFini

Chapter 6327

A Concrete Naming Certificate for $\mathsf{RealModulusComparison}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers with explicit moduli of convergence, the $\mathsf{RealModulusComparison}^{\uparrow}$ packet records how two Real-facing convergence-modulus certificates are compared without quotienting streams or selecting representatives. It sits after the comparison surface of chapter 636, the convergence ledger of chapter 2074, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal Real seal of chapter 38. The packet is a finite NameCert bridge between rate certificates; it is not a selected limit, a quotient equality of real names, a countable-choice schedule, or an ambient completeness principle.

Definition 6327.1 (Real modulus comparison carrier)**.** *A* $\mathsf{RealModulusComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (C, M_0, M_1, S, R, D, E, H, T, P, N)$$

with the following displayed rows:

1. *C is the* $\mathsf{CauchyModulusComparison}^{\uparrow}$ *comparison row naming the cofinal comparison between the two displayed schedules.*
2. *M_0 and M_1 are* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger rows for the two Real-facing convergence certificates under comparison.*
3. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule shared by the requested comparison read.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback row reached from the same finite windows.*
5. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row that names the requested precision and the dyadic comparison budget.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the two modulus ledgers, finite windows, readback row, tolerance row, and comparison row have been displayed.*
7. *H records componentwise* hsame *transport for the comparison row, both modulus ledgers, stream schedule, readback row, dyadic tolerance row, Real seal, replay, provenance, and local naming.*
8. *T records displayed* Cont *replay from a comparison request through $S, R, D, M_0, M_1, C, E, H, P, N$.*
9. *P is the* Pkg *provenance over* $\mathsf{RealModulusComparison}^{\uparrow}$, $\mathsf{CauchyModulusComparison}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}$ $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RealModulusComparison}^{\uparrow}}$ *row.*

Accepted packets are compared only through $C, M_0, M_1, S, R, D, E, H, T, P, N$. There is no coordinate for selected limits, quotient stream equality, countable choice, host Real equality, or ambient Real completeness.

Theorem 6327.2 (Real modulus comparison NameCert obligations)**.** *For every accepted* $\mathsf{RealModulusComparison}^{\uparrow}$ *carrier*

$$K = (C, M_0, M_1, S, R, D, E, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RealModulusComparison}^{\uparrow}}$ *surface consists exactly of carrier admission for K, dyadic tolerance exposure through D, finite-window handoff through S, regular-rational readback through R, cofinal modulus comparison through C, M_0, M_1, Real-seal nonescape through E, componentwise* hsame *transport through H, displayed* Cont *replay through T, package provenance through P, and local naming through N. Every Real or Completion consumer must read these rows before it may use comparison data between convergence-modulus packets.*

Proof. Project the carrier of Definition 6327.1. The rows M_0, M_1 are the only Real-facing convergence-modulus ledgers, and C is the only comparison row between them.

The finite observation route is ordered by S, R, D. A comparison request first opens the displayed $\mathsf{StreamName}^{\uparrow}$ windows, then reads regular rational data through R, and then names the requested dyadic tolerance through D before the cofinal comparison row can be consumed.

The Real seal E is terminal for this packet. It is reached only after the two modulus ledgers and their comparison row have been displayed, while H, T, P, N transport, replay, source, and name that same route.

Since the carrier has no coordinate for selected limits, quotient stream equality, countable choice, host Real equality, or ambient Real completeness, the listed rows exhaust the local NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{RealModulusComparison}^{\uparrow}$ comparison read factors through one $\mathsf{CauchyModulusComparison}^{\uparrow}$ row, two $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger rows, one $\mathsf{StreamName}^{\uparrow}$ finite-window schedule, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, one $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance row, one $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local $\mathsf{NameCert}$ before any Real or Completion consumer may use the comparison.

Independence witness: cauchymoduluscomparison, realmodulusofconvergence, streamname, regseqrat, dyadicratcore, real: $\mathsf{RealModulusComparison}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the comparison row, two Real-modulus ledgers, finite stream schedule, regular readback, dyadic tolerance, Real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{RealModulusComparison}^{\uparrow}}$**.**

Constructive story: A $\mathsf{RealModulusComparison}^{\uparrow}$ packet is generated from a CauchyModulusComparisonUp row, two RealModulusOfConvergenceUp ledger rows, StreamNameUp finite windows, RegSeqRatUp readback, DyadicRatCoreUp tolerances, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{RealModulusComparison}^{\uparrow}}$, seed)

Scope closed: Definition 6327.1 and Definition 6327.2 seed the finite comparison surface for Real-facing convergence-modulus certificates.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{RealModulusComparison}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient stream equality, countable choice, host Real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window handoff, regular-rational readback, cofinal modulus comparison, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RealModulusComparisonUp.

Chapter 6328

A Concrete Naming Certificate for $\mathsf{WhiteheadProduct}^{\uparrow}$

Following J. H. C. Whitehead on products in homotopy groups, the $\mathsf{WhiteheadProduct}^{\uparrow}$ packet names the bracket of two pointed sphere maps by finite BEDC rows. It sits beside the circle source of chapter 58, the deformation classifier of chapter 156, and the horn-and-face bookkeeping of chapter 157. The certificate records pointed sphere maps, a commutator attachment, a bracket classifier, componentwise transport, replay, provenance, and local naming; it does not import a completed homotopy group, a CW attaching theorem, or a host quotient of maps.

Definition 6328.1 ($\mathsf{WhiteheadProduct}^{\uparrow}$ finite carrier)**.** *A* $\mathsf{WhiteheadProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\Omega = (S_m, S_n, a, b, \partial, \kappa, \beta, H, C, P, N).$$

Its displayed rows are as follows:

1. *S_m and S_n are pointed sphere-source rows. Each records a finite basepoint, boundary, and face ledger read through the sphere source surface of chapter 58 and the simplicial face discipline of chapter 157.*

2. *a and b are pointed sphere-map rows into the same displayed target packet. Their basepoint rows agree with the target basepoint by visible* hsame *transport, and their homotopy admissibility is read through chapter 156.*

3. *∂ is the attachment row for the boundary of the product cell. It records the two coordinate-face routes, the basepoint collar, and the finite source-face identifications used to read the commutator boundary.*

4. *κ is the commutator row. It lists the ordered route $aba^{-1}b^{-1}$ along the displayed boundary faces, together with the face-by-face signs and basepoint returns needed to keep the pointed condition visible.*

5. *β is the homotopy-bracket classifier row. It classifies two accepted packets by componentwise* $\mathsf{Homotopy}^{\uparrow}$ *comparison of a and b, by* hsame *comparison of the sphere-source rows, and by equality of the displayed attachment and commutator ledgers.*

6. *H records componentwise* hsame *transport for $S_m, S_n, a, b, \partial, \kappa, \beta, C, P, N$.*

7. *C records displayed* Cont *replay for the two pointed map rows, the boundary attachment route, the commutator readback, and the bracket classifier readback.*

8. *P is the* Pkg *provenance over* $\mathsf{WhiteheadProduct}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{Homotopy}^{\uparrow}$, $\mathsf{SimplicialSet}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

9. *N is the local* $\mathsf{NameCert}_{\mathsf{WhiteheadProduct}^{\uparrow}}$ *row.*

The carrier is accepted only when all rows occur in the same displayed finite packet. Thus the Whitehead product is presented as an inspectable sphere-attachment and commutator ledger rather than as an element of an ambient homotopy group.

Theorem 6328.2 ($\mathsf{WhiteheadProduct}^{\uparrow}$ NameCert obligations)**.** *For every accepted* $\mathsf{WhiteheadProduct}^{\uparrow}$ *carrier*

$$\Omega = (S_m, S_n, a, b, \partial, \kappa, \beta, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{WhiteheadProduct}^{\uparrow}}$ *surface consists exactly of sphere-source admission, pointed map admission, attachment admission, commutator ledger exactness, homotopy-bracket classification, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. A downstream homotopy consumer may inspect $S_m, S_n, a, b, \partial, \kappa, \beta, H, C, P, N$ as the finite bracket packet, but it cannot replace the packet by a host homotopy-group element or by an uncarried attaching map.*

Proof. Carrier admission is projection from Definition 6328.1. The rows $S_m, S_n, a, b, \partial, \kappa, \beta, H, C, P, N$ already list the two pointed source spheres, the two pointed maps, the attachment boundary, the commutator route, the classifier, transport, replay, provenance, and local name.

The NameCert comparison is the classifier row β. It compares the two pointed map rows through $\mathsf{Homotopy}^{\uparrow}$, compares the sphere-source rows through their displayed source classifiers, and then checks that the attachment and commutator rows have the same finite face itinerary. No comparison step quantifies over all maps from a product cell or over a completed homotopy group.

The stability fields are exactly H and C. Transport along H keeps the component rows in the same packet positions, and replay along C keeps the boundary attachment, commutator readback, and bracket classifier visible under the same provenance P.

The local name N is therefore attached to the finite bracket packet itself. Since each public coordinate has been accounted for by admission, classification, transport, replay, and provenance, there is no remaining route through which a consumer can demand an ambient Whitehead-product theorem. □

Theorem 6328.3 ($\mathsf{WhiteheadProduct}^{\uparrow}$ sphere-attachment route)**.** *Let $\Omega = (S_m, S_n, a, b, \partial, \kappa, \beta, H, C, P, N)$ be an accepted* $\mathsf{WhiteheadProduct}^{\uparrow}$ *carrier. Any consumer that reads the bracket as a sphere-attachment factors through the pointed sphere rows S_m, S_n, the pointed maps a, b, the attachment row ∂, the commutator row κ, the homotopy-bracket classifier β, the componentwise transport H, the displayed replay C, the provenance P, and the local name N. The consumer may read the ordered boundary route and the homotopy classifier, but it cannot substitute a quotient of maps, an unbounded cell complex, or a completed homotopy-group operation.*

Proof. Project the two pointed sphere rows and the two pointed map rows. Their basepoints and boundary faces are already displayed by S_m, S_n, a, b, and their admissibility is routed through the sibling sphere, simplicial-set, and homotopy certificates named in the carrier.

Next project the attachment row ∂. It exposes only the finite boundary itinerary: coordinate faces, basepoint collar, and source-face identifications. The commutator row κ reads that itinerary in the order $aba^{-1}b^{-1}$ and records the signs and basepoint returns needed for a pointed bracket.

Finally apply the classifier and stability rows. The row β classifies the bracket by the transported pointed maps and the identical finite attachment itinerary, while H and C preserve that itinerary under componentwise transport and replay.

Thus the sphere-attachment consumer sees precisely the finite BHist bracket route with Pkg provenance and local NameCert naming. No step introduces a host quotient, a global CW-complex construction, or a stronger homotopy-group operation. □

Closure status for $\mathsf{NameCert}_{\mathsf{WhiteheadProduct}^{\uparrow}}$.

Constructive story: A $\mathsf{WhiteheadProduct}^{\uparrow}$ packet is generated from two pointed sphere-source rows, two pointed sphere-map rows, a finite boundary attachment row, an ordered commutator row, a homotopy-bracket classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WhiteheadProduct}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6328.1, Definition 6328.2, and Definition 6328.3 seed the finite pointed sphere-map, attachment, commutator, homotopy-bracket classifier, transport, replay, provenance, and local naming surface for the Whitehead product.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WhiteheadProduct}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close higher homotopy-group algebra, Jacobi identities, suspension naturality, a CW-complex model, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires BEDC.Derived.WhiteheadProductUp to encode and check carrier admission, pointed sphere-map admission, attachment exactness, commutator exactness, bracket classification, transport, replay, provenance, and local naming.

Chapter 6329

A Concrete Naming Certificate for $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$

Following Bishop and Bridges constructive compact metric analysis, the $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$ packet records the finite located-net selection surface consumed by L10 regular-sequence and Real readback routes. It sits on the compact-metric source of chapter 129, the dyadic tolerance ledger of chapter 291, the finite observation windows of chapter 126, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet selects only finite located-net rows already displayed by the carrier; it does not assert open-cover compactness, selected subcovers, quotient equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6329.1 (Compact metric located-net selection carrier)**.** *A* $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (K, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row naming the compact metric packet whose finite net may be inspected.*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row against which located distances and selected net radii are read.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened by the tolerance request.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that transports the located net window into the regular rational route.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after K, D, W, R have been displayed.*
6. *H records componentwise* hsame *transport for compact-metric, dyadic, window, readback, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from a located-net request through K, D, W, R and into the terminal seal E.*
8. *P is the* Pkg *provenance over* $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}}$ *row.*

The carrier has no coordinate for open-cover compactness, selected subcovers, quotient real equality, countable choice, an infinite choice sequence of net points, or ambient Real completeness.

Theorem 6329.2 (Compact metric located-net selection obligations)**.** *For every accepted* CompactMetricLocatedNe *carrier* $L = (K, D, W, R, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}}$ *surface consists exactly of compact-metric source exposure through* K*, dyadic tolerance exposure through* D*, finite StreamName window exposure through* W*, RegSeqRat readback through* R*, terminal Real sealing through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*,* Pkg *provenance through* P*, and local naming through* N*.*

Proof. Project the carrier. The compact-metric row K supplies the only source of finite net data, while D, W, R, E are the only analytic rows available before a Real-facing read is exposed.

Read the located-net request through the displayed L10 route. The dyadic tolerance row fixes the finite radius request, the StreamName row opens only the needed observation window, and the RegSeqRat row records the regular rational readback for that window.

The structural tail is exhausted by H, C, P, N. Componentwise transport, continuation replay, provenance, and local naming preserve the same finite packet, so the NameCert obligations cannot export open-cover compactness, selected subcovers, quotient equality, countable choice, or ambient Real completeness. □

Theorem 6329.3 (Compact metric located-net selection L10 handoff)**.** *Every L10 regular-sequence or Real readback consumer of an accepted* $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$ *carrier must first read the finite route*

$$\mathsf{CompactMetric}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exports a located finite-net window over displayed compact-metric and dyadic rows, not open-cover compactness, a selected subcover, quotient real equality, countable choice, or ambient Real completeness.

Proof. Begin with the NameCert obligations. They expose $K, D, W, R, E, H, C, P, N$ before any consumer can treat the located-net selection as available.

The L10 route is ordered by projection from the carrier. The compact metric row is read first, the dyadic tolerance row determines the requested finite radius, the StreamName window supplies the inspected observations, and the RegSeqRat row performs the regular rational readback.

Only after those rows are visible may the terminal Real seal be consumed. Since no coordinate stores open-cover compactness, selected subcovers, quotient equality, countable choice, or ambient Real completeness, the handoff remains a finite located-net readback surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}}$**.**

Constructive story: A $\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}$ packet is generated from compact-metric source rows, dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, terminal Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}}$, seed)

Scope closed: Definition 6329.1, Definition 6329.2, and Definition 6329.3 seed the finite located-net selection surface for L10 RegSeqRat and Real readback consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{CompactMetricLocatedNetSelection}^{\uparrow}}$, unformalized)

Lean target: `BEDC.Derived.CompactMetricLocatedNetSelectionUp`

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected subcovers, quotient real equality, countable choice, an infinite choice sequence of net points, ambient Real completeness, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-net row exposure, dyadic tolerance routing, finite-window readback, terminal Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.CompactMetricLocatedN

Chapter 6330

A Concrete Naming Certificate for $\mathsf{TriebelLizorkinSpace}^{\uparrow}$

Following Triebel–Lizorkin theory and constructive regularity spaces, the $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ packet records the finite mixed scale-position surface by which a smoothness request can inspect local dyadic data before aggregating it into a global regularity name. It sits beside the Besov aggregate of chapter 4204, the Sobolev weak-gradient ledger of chapter 440, the Holder pointwise boundary of chapter 1766, the weak-derivative action row of chapter 2210, the dyadic scale budget of chapter 291, the regular rational readback surface of chapter 301, and the terminal real-name seal of chapter 38. The packet names only finite scale windows, local position probes, mixed aggregation rows, dyadic tolerances, regular rational readback, real seals, transport, replay, provenance, and local naming; it does not assert a completed distribution space, Littlewood–Paley decomposition equality, quotient function identity, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6330.1 (Triebel–Lizorkin space carrier)**.** *A* $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, Hd, W, D, R, E, Q, L, A, M, G, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{Sobolev}^{\uparrow}$ *source row, recording the weak-gradient and integrability ledger consumed by the local regularity request.*
2. *Hd is the* $\mathsf{HolderContinuity}^{\uparrow}$ *boundary row, recording the visible power-law control used when the packet exposes pointwise smoothness data.*
3. *W is the* $\mathsf{WeakDerivative}^{\uparrow}$ *action row, naming the finite test probes and derivative observations inspected by the scale-position request.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *scale ledger, recording dyadic scale, radius, and comparison budgets for each displayed window.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the regular rational magnitudes attached to the selected scale-position cells.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after S, Hd, W, D, R have been displayed.*
7. *Q is the finite window schedule selecting the visible scale levels, local position probes, and comparison requests carried by this packet.*

8. *L is the local position-bundle row, recording the finite family of probe locations attached to each visible dyadic scale.*
9. *A is the mixed aggregation row. It records that local position data is aggregated inside each scale before the global regularity request is named.*
10. *M is the Besov comparison row, exposing the difference between the Triebel–Lizorkin mixed scale-position route and the aggregate Besov route of chapter 4204.*
11. *G stores componentwise* hsame *transport for the source, boundary, weak-action, dyadic, regular-rational, real-seal, window, local-position, mixed-aggregation, Besov-comparison, replay, provenance, and local naming rows.*
12. *C stores displayed* Cont *replay from a Triebel–Lizorkin regularity request through*
$$S, Hd, W, D, R, E, Q, L, A, M \longmapsto G, P, N.$$
13. *P is the* Pkg *provenance over* $\mathsf{TriebelLizorkinSpace}^{\uparrow}$, $\mathsf{BesovSpace}^{\uparrow}$, $\mathsf{Sobolev}^{\uparrow}$, $\mathsf{HolderContinuity}^{\uparrow}$, $\mathsf{WeakDerivative}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
14. *N is the local* $\mathsf{NameCert}_{\mathsf{TriebelLizorkinSpace}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $S, Hd, W, D, R, E, Q, L, A, M, G, C, P, N$. It has no coordinate for a completed distribution space, quotient equality of function spaces, selected Littlewood–Paley representatives, host equality of local probes, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6330.2 (Triebel–Lizorkin space NameCert obligations)**.** *For every accepted* $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ *carrier*
$$T = (S, Hd, W, D, R, E, Q, L, A, M, G, C, P, N),$$
the local $\mathsf{NameCert}_{\mathsf{TriebelLizorkinSpace}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed Sobolev row, Holder row, weak-derivative action row, dyadic scale ledger, regular rational readback row, terminal real seal, finite window schedule, local position bundle, mixed aggregation row, Besov comparison row, componentwise transport, displayed replay, provenance, and local naming rows. Every PDE, trace, interpolation, or nonlinear regularity consumer of the packet must read through those rows before it cites the Triebel–Lizorkin smoothness boundary.*

Proof. Carrier admission is projection from Definition 6330.1. The source rows are S, Hd, W, the scale and readback rows are D, R, E, Q, the mixed scale-position rows are L, A, M, and the structural rows are G, C, P, N.

The Sobolev, Holder, and weak-derivative rows supply the analytic source surface. The Sobolev row carries weak-gradient and integrability data, the Holder row carries the visible pointwise boundary, and the weak-derivative row carries the finite test-action observations.

The dyadic, regular-rational, and real-seal rows fix the numerical part of the packet. The window schedule Q selects the displayed scale levels, D attaches the tolerance budget, R reads back the regular rational magnitudes, and E seals the resulting real-facing data.

The local position bundle L and mixed aggregation row A are the Triebel–Lizorkin-specific surface. The Besov comparison row M records that the packet keeps scale and position coupled before a global regularity name is exported.

Finally G, C, P, N transport, replay, source, and name the same finite route. Since no completed distribution space, quotient function-space equality, selected Littlewood–Paley representative, host equality of probes, countable choice, or ambient completeness coordinate appears in the carrier, the listed rows exhaust the local NameCert surface. □

Theorem 6330.3 (Triebel–Lizorkin dyadic scale route)**.** *Every L10 or L11 regularity consumer of an accepted* $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ *carrier factors through the finite route*

$$\mathsf{Sobolev}^{\uparrow},\quad \mathsf{WeakDerivative}^{\uparrow},\quad \mathsf{HolderContinuity}^{\uparrow},\quad \mathsf{DyadicRatCore}^{\uparrow},\quad \mathsf{RegSeqRat}^{\uparrow},\quad \mathsf{Real}^{\uparrow},\quad \mathsf{BesovSpace}^{\uparrow},\quad \mathsf{BHi}$$

The route exports local scale-position windows, dyadic tolerance control, regular rational readback, terminal real seals, the explicit comparison with the Besov aggregate, componentwise transport, continuation replay, provenance, and local naming. It exports no completed distribution space, quotient function space, selected Littlewood–Paley representative, host equality of functions or local probes, countable-choice row, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Begin with the NameCert obligations of Definition 6330.2. They require every accepted regularity read to pass through the Sobolev, Holder, and weak-derivative rows before any downstream smoothness boundary is cited.

Next read the finite scale-position data. The window schedule Q chooses the visible dyadic levels, the local position bundle L records the probe locations at those levels, and the mixed aggregation row A keeps those observations coupled before the global name is exported.

The dyadic ledger D, regular rational row R, and terminal real seal E report the numerical data attached to the same windows. The Besov comparison row M exposes the sibling aggregate route without identifying the two carriers.

Thus an L10 or L11 consumer receives exactly the displayed chain from Sobolev and weak-derivative evidence through Holder, dyadic, regular-rational, real-seal, Besov-comparison, and structural rows. The excluded distribution-space, quotient, representative-selection, host-equality, choice, and ambient-completeness objects have no carrier coordinate and cannot be exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ regularity consumer must read Sobolev, HolderContinuity, WeakDerivative, DyadicRatCore, RegSeqRat, Real, BesovSpace, local position-bundle, mixed aggregation, hsame, Cont, Pkg, and local NameCert rows before citing the Triebel–Lizorkin smoothness boundary.

Independence witness: besov-space, sobolev, holder-continuity, weak-derivative, dyadicratcore, regseqrat, real: $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries smoothness-source rows, local scale-position probes, mixed aggregation, dyadic scale budgets, regular rational readback, real seals, Besov comparison, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{TriebelLizorkinSpace}^{\uparrow}}$**.**

Constructive story: A $\mathsf{TriebelLizorkinSpace}^{\uparrow}$ packet is generated from Sobolev, HolderContinuity, WeakDerivative, DyadicRatCore, RegSeqRat, Real, finite window, local position-bundle, mixed aggregation, Besov comparison, componentwise hsame, displayed Cont replay, Pkg provenance, and local NameCert rows.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TriebelLizorkinSpace}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6330.1, Definition 6330.2, and Definition 6330.3 seed the finite constructive Triebel–Lizorkin route over Sobolev, HolderContinuity, WeakDerivative, DyadicRatCore, RegSeqRat, Real, BesovSpace, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TriebelLizorkinSpace}^{\uparrow}}, \mathsf{unformalized})$

Lean4 (statement) `BEDC.Derived.TriebelLizorkinSpaceUp`

Bridge status: none

Not claimed: This chapter does not claim Lean verification, completed distribution spaces, Littlewood–Paley representative selection, quotient function spaces, host equality of functions or probes, countable choice, ambient Real completeness, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, local scale-position window exactness, mixed aggregation stability, dyadic-scale ledger stability, regular rational readback, Real seal nonescape, Besov comparison, transport, replay, provenance, and local naming for BEDC.Derived.TriebelLizorkinSpaceUp.

Chapter 6331

A Concrete Naming Certificate for $\mathsf{DyadicRegularBridge}^{\uparrow}$

Following Bishop and Bridges on constructive real numbers by regular Cauchy data, the $\mathsf{DyadicRegularBridge}^{\uparrow}$ packet records the finite bridge from dyadic tolerance rows to regular rational readback and the terminal $\mathsf{Real}^{\uparrow}$ seal. It sits after the dyadic rational core of chapter 291, the finite stream-window schedule of chapter 126, the regular rational readback of chapter 301, the Cauchy modulus route of chapter 292, and the real-name boundary of chapter 38. The bridge gives Real and Completion consumers a displayed BEDC route from finite dyadic tolerances to regular Cauchy real seals, without using quotient stream equality, selected limits, countable choice, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6331.1 (Dyadic regular bridge carrier)**.** *A* $\mathsf{DyadicRegularBridge}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (D, W, Q, M, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, containing the dyadic radius and comparison rows used by the bridge.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window schedule naming the inspected stream prefixes.*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row that interprets the scheduled windows as regular rational data.*
4. *M is the* $\mathsf{CauchyModulus}^{\uparrow}$ *scheduling row, matching dyadic tolerances to the finite windows that Q reads.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, W, Q, M have been displayed.*
6. *H records componentwise* hsame *transport for dyadic, window, readback, modulus, real-seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from dyadic tolerances to scheduled windows, from scheduled windows to regular readback, and from regular readback to the real seal.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicRegularBridge}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicRegularBridge}^{\uparrow}}$ *row.*

The classifier accepts only those reads that pass through $D, W, Q, M, E, H, C, P, N$. It has no coordinate for a quotient of streams, a chosen limiting point, countable-choice selection, host real equality, or ambient completeness.

Theorem 6331.2 (Dyadic regular bridge NameCert obligations)**.** *For every accepted* $\mathsf{DyadicRegularBridge}^{\uparrow}$ *carrier* $B = (D, W, Q, M, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{DyadicRegularBridge}^{\uparrow}}$ *surface consists exactly of carrier admission, dyadic tolerance exposure, finite-window scheduling, regular rational readback, Cauchy modulus scheduling, terminal real-seal handoff, componentwise* hsame *transport, displayed* Cont *replay, package provenance, and local naming.*

Proof. Project the carrier of Definition 6331.1. The rows $D, W, Q, M, E, H, C, P, N$ are the full finite inventory available to a bridge consumer.

The analytic part is ordered. The dyadic ledger D supplies the finite tolerance rows; the schedule W selects the inspected windows; the readback Q interprets those windows as regular rational data; and the modulus row M records which windows answer which dyadic tolerances.

The terminal seal E is reached only after those rows have been displayed. The structural rows H, C, P, N transport, replay, source, and name the same finite route, so no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness theorem is available inside the packet. □

Theorem 6331.3 (Dyadic regular bridge Real seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{DyadicRegularBridge}^{\uparrow}$ *carrier* $B = (D, W, Q, M, E, H, C, P, N)$ *factors through the dyadic tolerance ledger* D*, finite stream-window schedule* W*, regular rational readback* Q*, Cauchy modulus schedule* M*, terminal real seal* E*, componentwise transport* H*, displayed replay* C*, provenance* P*, and local naming* N*.*

Proof. Start from a Real-facing bridge read. The carrier contains exactly one terminal real-seal row, namely E, so any such read must arrive at E through the displayed bridge rows.

The source of the route is D. The finite window row W and the modulus row M make the tolerance schedule visible, while Q supplies the regular rational readback consumed by the seal.

Finally H, C, P, N preserve the same route under transport, continuation replay, package provenance, and local naming. Since the carrier contains no quotient stream equality, chosen limit, countable-choice selector, host real equality, or ambient completeness row, the Real seal route is exactly the finite BEDC route through $D, W, Q, M, E, H, C, P, N$. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicRegularBridge}^{\uparrow}}$**.**

Constructive story: A $\mathsf{DyadicRegularBridge}^{\uparrow}$ packet is generated from dyadic tolerance rows, finite StreamName windows, RegSeqRat readback, CauchyModulus scheduling, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{DyadicRegularBridge}^{\uparrow}}$, seed)

Scope closed: Definition 6331.1, Definition 6331.2, and Definition 6331.3 seed the finite dyadic-to-regular-to-real bridge route.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{DyadicRegularBridge}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, dyadic tolerance exposure, finite-window scheduling, regular readback, Cauchy modulus scheduling, Real seal handoff, transport, replay, provenance, and local naming for BEDC.Derived.DyadicRegularBridgeUp.

The $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ route reads event-row material through chapter 6508 before the hendeca event-flow spine consumes it, so the capstone surface stays singular.

Chapter 6332

A Concrete Naming Certificate for $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$

The $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ packet is the arity-eleven event-flow capstone for the carrier-isomorphism bucket. It sits beside the ordered sequence window of chapter 1377, the closed-ball compactness packet of chapter 3548, the regulated integral handoff of chapter 1557, the deca sequence capstone of chapter 1368, and the dodeca tuple capstone of chapter 1369. The chapter names one finite event-flow carrier that exposes exactly eleven event rows before any bucket member read.

Lean4 (statement) `BEDC.Derived.BHistHendecaEventFlowNameCertUp`

Falsifiable prediction: Every accepted $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ packet exposes exactly eleven event-flow rows before any bucket member read.

Independence witness: closed-ball-compactness, regulated-integral, b-hist-deca-sequence-name-cert, and b-hist-dodeca-sequence-name-cert: none is bijective because $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ carries eleven ordered event-flow rows together with transport, replay, provenance, and local naming, while the siblings provide compactness, integration, arity-ten sequence, or arity-twelve tuple surfaces.

Definition 6332.1 (BHist hendeca event-flow NameCert carrier)**.** *A* $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (R_0, R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, H, C, P, N)$$

whose first eleven rows are the ordered event-flow rows admitted by the capstone. The rows H, C, P, N *record componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and the local* $\mathsf{NameCert}_{\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}}$ *row. A consumer may inspect a bucket member only after the eleven event-flow rows and the four structural rows have been displayed.*

Theorem 6332.2 (BHist hendeca event-flow NameCert obligations)**.** *Every accepted* $\mathsf{BHistHendecaEventFlowName}$ *packet exposes exactly the rows* $R_0, \ldots, R_{10}, H, C, P, N$ *before any carrier-isomorphism bucket member is read. The local NameCert surface consists of eleven-row event admission, order preservation, componentwise* hsame *transport, displayed* Cont *replay, package provenance, local naming, and refusal of hidden twelfth rows, unordered tuple membership, host equality, or bucket-wide replacement of the sibling chapters.*

Proof. Project the carrier of Definition 6332.1. The first eleven coordinates are the only event-flow rows, and the remaining coordinates are exactly the transport, replay, provenance, and local naming rows.

Order preservation is part of the displayed packet: the consumer reads R_0 through R_{10} as a finite sequence of event-flow rows, not as an unordered tuple or a bucket-level classifier.

The structural rows H, C, P, N preserve and name that same eleven-row event surface. Since no coordinate contains a hidden twelfth row, host equality classifier, or replacement for the closed-ball compactness, regulated-integral, deca-sequence, or dodeca-tuple siblings, every accepted read remains inside the displayed capstone. □

Theorem 6332.3 (BHist hendeca event-flow TasteGate follow-up)**.** *The Lean-facing follow-up for* $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ *is the packet family under* Derived/BHistHendecaEventFlowNameCertUp/TasteGa *with explicit* Nontrivial *and* FieldFaithful *instances. The carrier must keep all eleven event-flow rows recoverable from the admitted packet before any bucket member read is accepted.*

Proof. The carrier definition lists eleven event rows and four structural rows. A nontrivial instance must display two distinct admitted packets, while field faithfulness must recover every listed row from the carrier projection.

The NameCert obligations identify the acceptance boundary. Any Lean gate that omits one event row, merges two positions, reads a bucket member first, or hides a twelfth row fails the displayed paper carrier. □

Closure status for $\mathsf{NameCert}_{\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}}$.

Constructive story: A $\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}$ packet is generated from eleven ordered event-flow rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}}$, seed)

Scope closed: Definition 6332.1, Definition 6332.2, and Definition 6332.3 seed the arity-eleven event-flow capstone surface.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{BHistHendecaEventFlowNameCert}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.BHistHendecaEventFlowNameCertUp`

Bridge status: none

Not claimed: This chapter does not close theorem checking, bridge checking, unordered bucket tuple equivalence, a hidden twelfth event row, host equality, or replacement of any sibling carrier.

Upgrade path: Obligation closure requires checked eleven-row event admission, order preservation, transport, replay, provenance, local naming, Nontrivial, and FieldFaithful data for BEDC.Derived.BHistHendecaEventFlowNameCertUp.

Chapter 6333

A Concrete Naming Certificate for $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ packet records the finite boundary at which a regular Cauchy rational name exposes a located split before the terminal real-name seal is read. It sits between the regular rational source of chapter 301, the located-real surface of **??**, the dyadic tolerance ledger of chapter 291, the finite window interface of chapter 126, the terminal Real naming packet of chapter 38, and the generator-internal closure discipline of chapter 9389. The packet is a displayed L10 split witness for Real and Completion consumers; it is not trichotomy, quotient real equality, countable choice, a selected limit, or an ambient completeness principle for $\mathsf{Real}^{\uparrow}$.

Definition 6333.1 (Regular Cauchy located split carrier)**.** *A* $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (Q, L, D, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *Q is the regular Cauchy source row, recording the accepted* $\mathsf{RegSeqRat}^{\uparrow}$ *name whose split boundary is being read.*
2. *L is the located interval evidence row, recording the* $\mathsf{LocatedReal}^{\uparrow}$ *lower and upper side data used by the split request.*
3. *D is the positive dyadic split budget row, recording the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance, slack, and positive-gap budget inspected at the split.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observation windows at which Q, L, and D are compared.*
5. *R is the regular readback row, recording the* $\mathsf{RegSeqRat}^{\uparrow}$ *finite readback after the split budget has been applied to the visible windows.*
6. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after the source, located evidence, dyadic budget, stream window, and regular readback rows have been displayed.*
7. *H records componentwise* hsame *transport for Q, L, D, W, R, E and the local certificate rows.*
8. *C records displayed* Cont *replay from a split request through the same source, located, dyadic, window, readback, Real-seal, transport, provenance, and local naming rows.*

9. *P is the* Pkg *provenance over* $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedSplit}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $Q, L, D, W, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient equality of completed reals, countable choice, an unbounded split search, a selected limit, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 (statement) `BEDC.Derived.RegularCauchyLocatedSplitUp`

Theorem 6333.2 (Regular Cauchy located split NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ *carrier $S = (Q, L, D, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedSplit}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular Cauchy source row, located interval evidence row, positive dyadic split budget, stream window, regular readback, Real seal, componentwise transport, displayed replay, provenance, and local naming. Every split read must expose $Q, L, D, W, R, H, C, P, N$ before the terminal Real seal row E is public.*

Proof. Project the carrier of Definition 6333.1. The source row is Q, the located evidence row is L, the positive dyadic budget is D, the finite observation row is W, the regular readback is R, and the Real seal is E.

The local certificate orders those rows. A split request first reads the regular Cauchy source, then the located interval evidence and dyadic budget at the finite windows where the readback is inspected.

Finally transport, replay, provenance, and local naming are supplied by H, C, P, N. Since the carrier has no trichotomy coordinate, quotient real equality, countable-choice schedule, selected limit, or ambient completeness row, the listed projections exhaust the NameCert obligations. □

Theorem 6333.3 (Regular Cauchy located split Real handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read exported by an accepted* $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{RegSeqRat}^{\uparrow},\ \mathsf{LocatedReal}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The handoff exposes the regular Cauchy source, located interval evidence, positive dyadic split budget, finite stream window, regular readback, componentwise transport, replay, provenance, and local naming before the Real seal may be consumed.

Proof. Start from Definition 6333.1. The Real seal row E is downstream of the source Q, located evidence L, dyadic budget D, finite windows W, and regular readback R.

Apply the NameCert obligations of Definition 6333.2. They require any public Real-facing read to keep the same $Q, L, D, W, R, H, C, P, N$ rows visible before E is opened.

Thus the L10 handoff is a finite generator-internal split witness in the sense of chapter 9389. A consumer that asks for trichotomy, quotient real equality, countable choice, selected limits, or ambient completeness must leave this packet and cite a downstream certificate. □

Falsifiable prediction: Deleting the positive dyadic split budget row while keeping the source and Real seal rows leaves no accepted $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ route from $\mathsf{RegSeqRat}^{\uparrow}$ finite windows to a located Real handoff.

Independence witness: regseqrat, locatedreal, dyadicratcore, streamname, real: no listed sibling is bijective with $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ because the packet jointly binds a source regular

Cauchy name, located interval evidence, positive dyadic budget, finite windows, regular readback, Real seal, transport, replay, provenance, and local NameCert row.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedSplit}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$ packet is generated from a regular Cauchy source row, located interval evidence, positive dyadic split budget, finite StreamName windows, regular readback, Real seal, hsame transport, Cont replay, Pkg provenance, and local NameCert evidence.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedSplit}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6333.1, Definition 6333.2, and Definition 6333.3 seed the finite located-split handoff from regular Cauchy names through dyadic and stream windows to Real-facing consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyLocatedSplit}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.RegularCauchyLocatedSplitUp`

Bridge status: none

Not claimed: This chapter does not close trichotomy, quotient real equality, countable choice, selected limits, ambient Real completeness, theorem-level Lean verification, or bridge checking for $\mathsf{RegularCauchyLocatedSplit}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, positive dyadic split exactness, located evidence stability, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularCauchyLocatedSplitUp.

Chapter 6334

A Concrete Naming Certificate for $\mathsf{RegularRealCauchySelector}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{RegularRealCauchySelector}^{\uparrow}$ packet records the finite selector surface by which an explicit regular-Cauchy Real read exposes its source, modulus, dyadic tolerance ledger, finite windows, regular-rational handoff, and terminal Real seal. It sits after the StreamName windows of chapter 126, the dyadic arithmetic ledger of chapter 291, the regular-rational route of chapter 301, the Cauchy-modulus discipline of chapter 292, and the terminal Real naming surface of chapter 38. The selector names one displayed read path; it does not choose a quotient-stream representative, select a host Real limit, invoke countable choice, or import ambient Real completeness.

Definition 6334.1 (Regular Real Cauchy selector carrier)**.** *A* $\mathsf{RegularRealCauchySelector}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$S = (R_0, M, D, W, Q, E, H, C, P, N)$$

with the following displayed rows:

1. R_0 *is the regular Real source row, recording the named Real-facing request before any terminal seal is consumed.*
2. M *is the explicit* $\mathsf{CauchyModulus}^{\uparrow}$ *row for the regular-Cauchy read, including the visible request-to-window schedule.*
3. D *is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same modulus request.*
4. W *is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row selected by* M *and read against* D*.*
5. Q *is the* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff row obtained from the displayed windows and dyadic tolerance data.*
6. E *is the* $\mathsf{Real}^{\uparrow}$ *seal row, available only after* R_0, M, D, W, Q *have been displayed.*
7. H *records componentwise* hsame *transport for the regular source, modulus, dyadic ledger, finite windows, regular-rational handoff, Real seal, replay, provenance, and local naming rows.*
8. C *records displayed* Cont *replay from the source and modulus rows through dyadic tolerance, finite windows, regular-rational handoff, the Real seal, provenance, and local naming.*
9. P *is the* Pkg *provenance over* $\mathsf{RegularRealCauchySelector}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*,* $\mathsf{StreamName}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* NameCert *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{RegularRealCauchySelector}^{\uparrow}}$ *row.*
The classifier compares accepted selectors only through $R_0, M, D, W, Q, E, H, C, P, N$. It has no coordinate for quotient stream equality, selected representatives, host real equality, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6334.2 (Regular Real Cauchy selector NameCert obligations)**.** *For every accepted* $\mathsf{RegularRealCauchySelector}^{\uparrow}$ *carrier $S = (R_0, M, D, W, Q, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularRealCauchySelector}^{\uparrow}}$ *surface consists exactly of carrier admission for the regular Real source, explicit Cauchy modulus, dyadic tolerance ledger, finite StreamName windows, RegSeqRat handoff, terminal Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every Real-facing selector read must expose $R_0, M, D, W, Q, H, C, P, N$ before it may consume E as Real seal data.*

Lean4 ✓ `BEDC.Derived.RegularRealCauchySelectorUp.RegularRealCauchySelectorCarrier namecert obligation surface`

Proof. Project the carrier of Definition 6334.1. The source row is R_0, the explicit schedule is M, the dyadic tolerance ledger is D, the finite observation surface is W, and the regular-rational handoff is Q.

The certificate orders those rows before the terminal seal. A consumer first reads the regular source and modulus request, then the dyadic tolerance ledger, then the StreamName windows, then the RegSeqRat handoff generated by those displayed rows.

The structural rows H, C, P, N preserve that same route under transport, replay, provenance, and local naming. Since no quotient stream, selected representative, host real equality, countable-choice schedule, or ambient completeness coordinate is present, the listed rows exhaust the local certificate obligations. □

Lean4 ✓ `BEDC.Derived.RegularRealCauchySelectorUp.RegularRealCauchySelectorTasteGate single carrier alignment`

Theorem 6334.3 (Regular Real Cauchy selector Real seal boundary)**.** *Any* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{RegularRealCauchySelector}^{\uparrow}$ *packet reads the terminal seal only after the regular source, explicit Cauchy modulus, dyadic tolerance ledger, finite StreamName windows, RegSeqRat handoff, transport, replay, provenance, and local naming rows have been displayed. Thus the selector cannot be used as a hidden choice of representative, a host Real limit, a quotient stream equality, or an ambient completeness principle.*

Proof. Use Definition 6334.2. It places the Real seal row E downstream of R_0, M, D, W, Q and downstream of the structural rows H, C, P, N.

The StreamName and RegSeqRat rows are therefore not optional evidence. They are the visible finite readback through which the explicit Cauchy modulus and dyadic tolerance ledger become Real-facing data.

All excluded objects would require a carrier coordinate absent from Definition 6334.1. The consumer receives only the displayed selector route, so the Real seal remains a terminal named row rather than a hidden representative, selected limit, quotient equality, or completeness claim. □

Falsifiable prediction: Every accepted $\mathsf{RegularRealCauchySelector}^{\uparrow}$ read exposes the regular source row, explicit $\mathsf{CauchyModulus}^{\uparrow}$ row, $\mathsf{DyadicRatCore}^{\uparrow}$ tolerance ledger, $\mathsf{StreamName}^{\uparrow}$ finite windows, $\mathsf{RegSeqRat}^{\uparrow}$ handoff, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any $\mathsf{Real}^{\uparrow}$ seal is public.

Independence witness: cauchymodulus, dyadicratcore, streamname, regseqrat, real: no listed sibling is bijective with $\mathsf{RegularRealCauchySelector}^{\uparrow}$ because the packet jointly binds the regular source, explicit modulus, dyadic tolerance ledger, finite windows, regular-rational handoff, terminal Real seal, transport, replay, provenance, and local NameCert.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularRealCauchySelector}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularRealCauchySelector}^{\uparrow}$ packet is generated from a regular Real source row, explicit Cauchy modulus row, DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat handoff, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularRealCauchySelector}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6334.1, Definition 6334.2, and Definition 6334.3 seed the explicit regular-Cauchy Real selector surface.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularRealCauchySelector}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected representatives, selected Real limits, countable choice, ambient Real completeness, Lean verification, or bridge checking for $\mathsf{RegularRealCauchySelector}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, explicit modulus exactness, dyadic tolerance stability, StreamName window readback, RegSeqRat handoff, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.RegularRealCauchySelectorUp.

Chapter 6335

A Concrete Naming Certificate for $\mathsf{LocalInterHistRateComparison}^{\uparrow}$

The $\mathsf{LocalInterHistRateComparison}^{\uparrow}$ packet realizes the local comparison surface of chapter 9419 as finite configuration, causal-witness, maximum-rate, no-global-sync, observer-local transport, provenance, and NameCert rows. It sits after the multi-history configuration of chapter 474, the cross-history causal witness surface of chapter 478, the unary maximum-rate packet of chapter 594, the no-global synchronization boundary of chapter 1003, and the observer-local transport route of chapter 889. The packet compares finite causal-rate rows without a synchronized frame, real-valued speed, metric tensor, Lorentz coordinate, or global observer chart.

Definition 6335.1 (Local inter-Hist rate comparison carrier)**.** *A* $\mathsf{LocalInterHistRateComparison}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (M, X, R, B, G, T, H, C, P, N)$$

with the following displayed rows:

1. *M is the* $\mathrm{MultiHistConfig}^{\uparrow}$ *row displaying the finite family of observer histories and their local ledgers.*
2. *X is the finite* $\mathsf{CrossHistCausal}^{\uparrow}$ *witness-family row over selected ordered pairs and local slots from M.*
3. *R is the* $\mathsf{MaxCausalRate}^{\uparrow}$ *row recording the shared unary bound and finite comparison ledger for the selected causal witnesses.*
4. *B is the local comparison row, recording which displayed causal-rate reads are compared and in which local order.*
5. *G is the* $\mathsf{NoGlobalSyncBoundary}^{\uparrow}$ *refusal row excluding synchronized clocks, privileged frames, and all-observer global charts.*
6. *T is the* $\mathsf{ObserverLocalityTransport}^{\uparrow}$ *row transporting the comparison through observer-local reindexings without changing the displayed configuration, causal witnesses, or bound row.*
7. *H records componentwise* hsame *transport for configuration, causal witnesses, maximum-rate data, comparison, refusal, observer-local transport, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from the multi-Hist configuration through causal witnesses, maximum-rate comparison, no-global-sync refusal, observer-local transport, provenance, and local naming.*

9. *P is the* Pkg *provenance over* LocalInterHistRateComparison$^{\uparrow}$, MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, NoGlobalSyncBoundary$^{\uparrow}$, ObserverLocalityTransport$^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*

10. *N is the local* NameCert$_{\text{LocalInterHistRateComparison}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $M, X, R, B, G, T, H, C, P, N$. It has no coordinate for a synchronized frame, real-valued speed, metric tensor, Lorentz geometry, global chart, quotient of histories, or host equality of observers.

Theorem 6335.2 (Local inter-Hist rate comparison NameCert obligations)**.** *For every accepted* LocalInterHistRateComparison$^{\uparrow}$ *carrier $L = (M, X, R, B, G, T, H, C, P, N)$, the local* NameCert$_{\text{LocalInterHistRateCompariso}}$ *surface consists exactly of multi-Hist configuration admission, cross-Hist causal-witness admission, maximum-rate bound readback, local comparison admission, no-global-sync refusal, observer-local transport, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. Every comparison read must expose $M, X, R, G, T, H, C, P, N$ before it may consume B as local rate-comparison data.*

Proof. Project the carrier of Definition 6335.1. The configuration row M supplies the observer-history family, X supplies selected causal witnesses over that family, and R supplies the maximum-rate bound and finite comparison ledger attached to those witnesses.

The comparison row B is local because it is read only after the no-global-sync refusal row G and observer-local transport row T are displayed. Thus the comparison cannot appeal to a hidden synchronized clock or global observer frame.

The structural rows H, C, P, N preserve exactly the same finite route under transport, replay, provenance, and local naming. Since no real-valued speed, metric tensor, Lorentz coordinate, global chart, quotient of histories, or host observer equality is a carrier coordinate, the listed rows exhaust the local certificate obligations. □

Theorem 6335.3 (Local inter-Hist rate comparison no-frame boundary)**.** *Any consumer of an accepted* LocalInterHistRateComparison$^{\uparrow}$ *packet reads rate comparison only as a finite row over* MultiHistConfig$^{\uparrow}$, CrossHistCausal$^{\uparrow}$, MaxCausalRate$^{\uparrow}$, NoGlobalSyncBoundary$^{\uparrow}$, *and* ObserverLocalityTransport$^{\uparrow}$. *The consumer cannot turn the comparison into a synchronized frame, global clock, real-valued speed, Lorentz geometry, or global observer chart.*

Proof. Use Definition 6335.2. It forces the comparison read through the displayed configuration, causal-witness family, maximum-rate bound, refusal row, observer-local transport, and structural rows.

The refusal row G keeps the comparison from becoming a global synchronization datum, while the transport row T keeps observer-local reindexing attached to the same visible configuration and causal witnesses.

The carrier contains no coordinate from which to build a metric tensor, Lorentz chart, real-valued speed, global clock, or quotient of histories. Therefore every accepted comparison remains a finite local BEDC row and nothing outside the listed surface is exported. □

Falsifiable prediction: Every accepted LocalInterHistRateComparison$^{\uparrow}$ read exposes the MultiHistConfig$^{\uparrow}$ configuration, CrossHistCausal$^{\uparrow}$ witness family, MaxCausalRate$^{\uparrow}$ bound row, NoGlobalSyncBoundary$^{\uparrow}$ refusal, ObserverLocalityTransport$^{\uparrow}$ route, transport, replay, provenance, and local NameCert before any local comparison row is public.

Independence witness: multihistconfig, crosshistcausal, maxcausalrate, noglobalsyncboundary, observerlocalitytransport: no listed sibling is bijective with LocalInterHistRateComparison$^{\uparrow}$ because

the packet jointly binds finite configuration, causal witnesses, maximum-rate bound, local comparison, no-global-sync refusal, observer-local transport, replay, provenance, and local NameCert.

The TasteGate file for $\mathsf{LocalInterHistRateComparison}^{\uparrow}$ must require Lean 'Nontrivial' and 'FieldFaithful' instances before any positive discovery or classifier-shift gate is recorded in BEDC.Derived.LocalInterHistR Without those instances, this chapter records a finite NameCert packet and not a field-sensitive discovery claim.

Closure status for $\mathsf{NameCert}_{\mathsf{LocalInterHistRateComparison}^{\uparrow}}$.

Constructive story: A $\mathsf{LocalInterHistRateComparison}^{\uparrow}$ packet is generated from MultiHistConfig rows, CrossHistCausal witness rows, MaxCausalRate bound rows, a local comparison row, a NoGlobalSyncBoundary refusal row, ObserverLocalityTransport rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocalInterHistRateComparison}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6335.1, Definition 6335.2, and Definition 6335.3 seed the finite local causal-rate comparison surface without a synchronized frame.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocalInterHistRateComparison}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close synchronized frames, global clocks, real-valued speeds, metric tensors, Lorentz geometry, host observer equality, Lean verification, or bridge checking for $\mathsf{LocalInterHistRateComparison}^{\uparrow}$.

Upgrade path: Obligation closure requires checked carrier admission, configuration locality, causal-witness handoff, maximum-rate bound readback, local comparison exactness, no-global-sync refusal, observer-local transport, replay, provenance, and local naming for BEDC.Derived.LocalInterHistRateComparisonUp.

Chapter 6336

A Concrete Naming Certificate for $\mathsf{CATZeroMetric}^{\uparrow}$

Following the Cartan–Alexandrov–Toponogov comparison tradition and the metric treatment of Bridson–Haefliger, the $\mathsf{CATZeroMetric}^{\uparrow}$ packet records a finite nonpositive-curvature comparison surface inside BEDC. It sits over the metric carrier of chapter 57, the finite segment witness of chapter 2213, the convex subset rows of chapter 208, and the completion handoff of chapter 128. The packet names comparison-triangle data for displayed geodesic windows; it is not a host path space, not a selected geodesic object, not a quotient path-equality principle, and not an ambient Hadamard-space theorem.

Definition 6336.1 (CAT(0) metric carrier)**.** *A* $\mathsf{CATZeroMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Z = (M, G, V, E, A, Q, D, L, H, C, P, N)$$

with the following displayed rows:

1. *M is the source* $\mathsf{Met}^{\uparrow}$ *row, carrying only the displayed point classifier, distance rows, and metric-law ledger.*
2. *G is the* $\mathsf{GeodesicMetric}^{\uparrow}$ *row that names the finite segment windows used by the comparison request.*
3. *V is the finite vertex triple row (x, y, z), with each vertex admitted through the point classifier of M.*
4. *E is the edge-segment row, displaying the three geodesic windows between x, y, z through G.*
5. *A is the Alexandrov comparison-triangle row. It records the Euclidean side-length readback and the comparison point parameters for the displayed edge windows.*
6. *Q is the CAT(0) comparison ledger. For each displayed pair of comparison parameters on the triangle edges, it stores the finite inequality saying that the metric distance in M is bounded by the corresponding Euclidean comparison distance.*
7. *D is the* $\mathsf{ConvexSet}^{\uparrow}$ *domain row for each displayed convex window on which the comparison read is consumed.*
8. *L is the* $\mathsf{CompleteMetric}^{\uparrow}$ *handoff row. It exposes the completion-facing read of the same finite comparison window without asserting that the ambient space is complete.*

9. *H records componentwise* hsame *transport for the metric row, geodesic row, vertex triple, edge windows, comparison-triangle row, inequality ledger, convex window, completion handoff, continuation replay, provenance, and local naming.*

10. *C records displayed* Cont *replay from a finite comparison request through*

$$M, G, V, E, A, Q, D, L \longmapsto H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{CATZeroMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{GeodesicMetric}^{\uparrow}$, $\mathsf{ConvexSet}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{CATZeroMetric}^{\uparrow}}$ *row.*

The carrier has no coordinate for a global CAT(0) existence theorem, selected geodesics, host Euclidean realization, quotient path equality, full convex projection operator, or ambient completeness.

Theorem 6336.2 (CAT(0) metric comparison obligations)**.** *For every accepted* $\mathsf{CATZeroMetric}^{\uparrow}$ *carrier*

$$Z = (M, G, V, E, A, Q, D, L, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CATZeroMetric}^{\uparrow}}$ *surface consists exactly of metric-source admission, geodesic segment-window admission, displayed vertex-triple admission, comparison-triangle readback, CAT(0) inequality ledger exactness, convex-window handoff, complete-metric handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming over the rows of Definition 6336.1.*

Proof. Carrier admission is projection from Definition 6336.1. The accepted packet displays exactly the metric source, geodesic row, vertex triple, edge windows, comparison-triangle row, inequality ledger, convex handoff, completion handoff, transport, replay, provenance, and naming coordinates.

The comparison obligation is the role of A and Q. Row A gives the Euclidean side-length and parameter readback for the displayed triangle, while Q records only the finite distance inequalities requested by the CAT(0) comparison.

Rows D and L keep downstream consumers attached to the displayed convex and complete-metric surfaces. The structural rows H, C, P, N transport, replay, source, and name the same finite data, so no obligation row can be read as a selected geodesic, host Euclidean construction, quotient path equality, or ambient completeness theorem. □

Theorem 6336.3 (CAT(0) metric projection non-escape)**.** *Every convex-projection or completion-facing consumer of an accepted* $\mathsf{CATZeroMetric}^{\uparrow}$ *packet must factor through the finite route*

$$\mathsf{Met}^{\uparrow},\ \mathsf{GeodesicMetric}^{\uparrow},\ \mathsf{ConvexSet}^{\uparrow},\ \mathsf{CompleteMetric}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

It may consume displayed comparison-triangle inequalities for the named convex window, but it cannot infer a global nearest-point projection map, a general Hadamard-space theorem, selected geodesics, quotient path equality, or ambient completeness.

Proof. Use Definition 6336.2. It exposes a convex-projection-facing or completion-facing read only after the metric source, geodesic windows, vertex triple, comparison-triangle row, and CAT(0) inequality ledger have all been displayed.

The displayed continuation route then passes through D and L. Thus a convex consumer reads the comparison ledger as finite certificate data for the shown window, and a completion

consumer reads the same window through the complete-metric handoff rather than through a global completeness assertion.

Inspecting the carrier definition gives the exclusion. There is no coordinate for a full projection operator, selected geodesic family, host Hadamard-space theorem, quotient path equality, or ambient completeness, so any consumer requiring one of those objects is outside the local classifier. □

Closure status for $\mathsf{NameCert}_{\mathsf{CATZeroMetric}^{\uparrow}}$.

Constructive story: A $\mathsf{CATZeroMetric}^{\uparrow}$ packet is generated from a MetricUp source, a GeodesicMetricUp segment-window row, a displayed vertex triple, comparison edge rows, an Alexandrov comparison-triangle readback, a finite CAT(0) inequality ledger, ConvexSetUp and CompleteMetricUp handoff rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CATZeroMetric}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6336.1, Definition 6336.2, and Definition 6336.3 seed the finite CAT(0) comparison route from displayed metric and geodesic data through convex and complete-metric consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CATZeroMetric}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim a global Hadamard-space theorem, selected geodesics, quotient path equality, a full nearest-point projection operator, ambient completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, comparison-triangle readback exactness, CAT(0) inequality-ledger stability, convex-window handoff, completion handoff, and BEDC.Derived.CatZeroMetricUp formalization.

Chapter 6337

Rising Sun Lemma NameCert Construction

Following Riesz's rising sun lemma, the $\mathsf{RisingSunLemma}^{\uparrow}$ packet records the finite BEDC surface for decomposing a located real interval into exposed and shaded subinterval windows for an upper-envelope comparison. It is a neighbour of the real-name boundary of chapter 38, the located-real route of **??**, the dyadic interval rows of chapter 306, and the compact interval maximum route of chapter 3907. The chapter isolates only the finite interval-decomposition certificate consumed by later $\mathsf{Real}^{\uparrow}$ analysis; it does not import a host measurable decomposition, selected family of open intervals, countable choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6337.1 (RisingSunLemma carrier)**.** *A* $\mathsf{RisingSunLemma}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$\mathsf{RisingSunLemma}(I, F, E, M, S, T, H, C, P, N)$$

with the following rows:

1. *I is the compact located interval row, read through the displayed* $\mathsf{Real}^{\uparrow}$ *endpoint names and the dyadic-interval subdivision ledger;*
2. *F is the located upper-envelope observation row on the interval;*
3. *E is the exposed-window ledger, listing finite dyadic subintervals on which the observation sits below a later envelope read;*
4. *M is the compact interval maximum handoff row used to certify the local envelope bound on each displayed closed window;*
5. *S is the shaded-window complement ledger, recording the remaining finite interval pieces after the exposed windows are read;*
6. *T is the terminal* $\mathsf{Real}^{\uparrow}$ *handoff row through which downstream consumers read the endpoint and envelope seals;*
7. *H is the componentwise* hsame *transport over the interval, observation, maximum, exposed-window, shaded-window, and terminal rows;*
8. *C is the displayed* Cont *replay route for the same rows;*
9. *P is the* Pkg *provenance over* $\mathsf{RisingSunLemma}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{DyadicInterval}^{\uparrow}$, $\mathsf{CompactIntervalMaximum}^{\uparrow}$ BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*

10. N is the local $\mathsf{NameCert}_{\mathsf{RisingSunLemma}^{\uparrow}}$ *row.*

Lean4 ✓ `BEDC.Derived.RisingSunLemmaUp.RisingSunLemmaTasteGate single carrier alignment`

Theorem 6337.2 (RisingSunLemma NameCert obligations)**.** *For every accepted* $\mathsf{RisingSunLemma}^{\uparrow}$ *carrier*

$$R = \mathsf{RisingSunLemma}(I, F, E, M, S, T, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RisingSunLemma}^{\uparrow}}$ *surface consists exactly of compact located interval admission through I, located upper-envelope readback through F, exposed-window visibility through E, compact interval maximum handoff through M, shaded-window complement visibility through S, terminal* $\mathsf{Real}^{\uparrow}$ *seal readback through T, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

The obligation rows are finite. The BHist *row stores only displayed dyadic subinterval packets and real-name observations; the* hsame *row transports the interval, envelope, maximum, exposed-window, shaded-window, and terminal rows componentwise; the* Cont *row replays the same finite inspection route; the* Pkg *row records provenance through the Real, located-real, dyadic-interval, and compact-maximum siblings; and the* NameCert *row admits no extra decomposition datum outside these rows.*

Proof. Start from the carrier rows in Definition 6337.1. The interval and endpoint data are already displayed as $\mathsf{Real}^{\uparrow}$ and located-real rows, while the subdivision data are finite dyadic interval rows.

Apply the compact interval maximum handoff to each displayed closed window in the finite dyadic ledger. This binds the envelope comparison to M before the exposed and shaded ledgers are consumed by any downstream real-analysis route.

The remaining structural rows are bookkeeping over the same finite packet. Componentwise hsame transport preserves each displayed interval and envelope row, Cont replay keeps the finite inspection order, Pkg records the sibling provenance, and N names the resulting local surface. Since the carrier contains no selected open family, no quotient real equality, no countable-choice row, and no ambient completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6337.3 (RisingSunLemma stopping-time handoff)**.** *Every downstream stopping-time or finite-sweep consumer of an accepted* $\mathsf{RisingSunLemma}^{\uparrow}$ *carrier factors through the exposed-window ledger E, the compact interval maximum handoff M, the shaded-window complement ledger S, and the terminal* $\mathsf{Real}^{\uparrow}$ *row T before it can read an interval-decomposition result. The handoff is therefore a finite interval decomposition over* BHist, hsame, Cont, Pkg, *and* NameCert *rows, not a selected countable union of open intervals, not quotient equality of real endpoints, and not an appeal to ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the accepted carrier in the order displayed by the packet. A stopping-time consumer first sees the compact located interval row and the finite dyadic subdivision ledger, so every candidate exposed window is already a finite interval row.

The envelope comparison is then routed through the compact interval maximum handoff. This supplies the local maximum row used to certify whether a window is exposed or remains in the shaded complement, without selecting an external family of intervals.

Finally the downstream consumer reads the terminal $\mathsf{Real}^{\uparrow}$ handoff through the displayed transport, replay, provenance, and naming rows. Thus every stopping-time read is a finite handoff through E, M, S, T, H, C, P, N, and no unlisted choice, quotient, bridge, or completeness datum can supply an alternate decomposition. □

Chapter 6338

A Concrete Naming Certificate for $\mathsf{RegularCauchyNet}^{\uparrow}$

Following Kelley, Bourbaki, and Moore–Smith on directed nets, the $\mathsf{RegularCauchyNet}^{\uparrow}$ packet records the directed-tail form of a regular Cauchy real-seal route as finite NameCert data. It refines the generic Cauchy-net carrier of chapter 589, the net/filter comparison of chapter 1789, and the separated completion surface of chapter 4066. It also keeps the finite observation windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance ledger of chapter 291, and the terminal real-name seal of chapter 38 visible before any completion consumer reads the route.

Definition 6338.1 (Regular Cauchy net carrier)**.** *A* $\mathsf{RegularCauchyNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (D, T, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the directed-index row, recording the finite directed preorder requests, comparison witnesses, and cofinal refinements used by the net route.*
2. *T is the tail-filter row, recording the finite tail neighborhoods and tail-containment reads induced by the directed indices.*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, recording the finite observation windows inspected after a tail request has been fixed.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row, repacking the accepted window observations as regular rational Cauchy data.*
5. *E is the* $\mathsf{Real}^{\uparrow}$ *seal handoff row, visible only after D, T, W, R have displayed the directed tail, filter read, window, and regular readback.*
6. *H stores componentwise* hsame *transport for directed indices, tail filters, StreamName windows, RegSeqRat readback, Real sealing, replay, provenance, and local naming.*
7. *C stores displayed* Cont *replay from a Real-facing precision request through the directed tail, tail-filter row, finite window, regular readback, and terminal seal.*
8. *P records* Pkg *provenance over* $\mathsf{RegularCauchyNet}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{CauchyNetFilterCorrespondence}^{\uparrow}$, $\mathsf{SeparatedCauchyNetCompletion}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *data.*

9. N is the local $\mathsf{NameCert}_{\mathsf{RegularCauchyNet}^\uparrow}$ *row.*

Accepted carriers are compared only through $D, T, W, R, E, H, C, P, N$. There is no coordinate for a quotient of nets, a selected subnet, a selected limit, countable choice, ambient separated completion, or host equality of real names.

Theorem 6338.2 (Regular Cauchy net NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyNet}^\uparrow$ *carrier $G = (D, T, W, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyNet}^\uparrow}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission for the directed index row D, tail-filter row T, finite window row W, regular rational readback R, Real seal E, transport H, replay C, provenance P, and local naming N;*
2. *directed-tail exactness, requiring every tail refinement read to use the same finite directed-index witnesses in D and the same tail-filter row T;*
3. *regular readback, requiring each* $\mathsf{RegSeqRat}^\uparrow$ *observation in R to be downstream of the displayed* $\mathsf{StreamName}^\uparrow$ *windows W and the dyadic tolerance comparisons carried by the route;*
4. *Real-seal nonescape, requiring the* $\mathsf{Real}^\uparrow$ *handoff E to expose only the displayed directed-tail, window, readback, transport, replay, provenance, and naming rows;*
5. *completion-consumer provenance, requiring every downstream completion read to cite the same packet G rather than a quotient net, selected subnet, selected limit, countable-choice schedule, or host completion theorem.*

No other NameCert obligation is available from the regular Cauchy net packet.

Proof. Carrier admission is the projection of Definition 6338.1. The tuple lists the directed index, tail filter, finite window, regular rational readback, Real seal, transport, replay, provenance, and naming rows.

Directed-tail exactness is carried by D and T. A consumer first fixes the finite directed comparison witnesses, then reads the induced tail-filter row before asking for window or Real-facing data.

Regular readback is downstream of W and the tolerance data cited in the route. The row R repacks only those displayed observations as $\mathsf{RegSeqRat}^\uparrow$ data, while E seals the resulting Real-facing row without adding a completion principle.

Finally H, C, P, N transport, replay, cite, and name the same packet. Since no carrier coordinate stores quotient nets, selected subnets, selected limits, countable choice, or ambient separated completion, none of those objects can be exported by the local NameCert surface. □

Theorem 6338.3 (Regular Cauchy net tail stability)**.** *Let $G = (D, T, W, R, E, H, C, P, N)$ be an accepted* $\mathsf{RegularCauchyNet}^\uparrow$ *carrier. If a directed-tail refinement replaces the active tail request by a cofinal tail still recorded in D and T, then every public completion read of the refined packet factors through the same* $\mathsf{StreamName}^\uparrow$ *windows, the same* $\mathsf{RegSeqRat}^\uparrow$ *readback route, and the same terminal* $\mathsf{Real}^\uparrow$ *seal handoff. Thus directed-tail refinement preserves the regular rational readback and Real-facing seal route of G.*

Lean4 ✓ `BEDC.Derived.RegularCauchyNetUp.RegularCauchyNetCarrier tail stability`

Proof. Start with the refined tail request. By directed-tail exactness in Definition 6338.2, the refinement is still read through D and T, so it has not left the accepted directed-index and tail-filter rows.

The window row W is downstream of those rows. The refined request therefore uses the same displayed finite StreamName windows, and the regular readback row R can only repack observations from that window surface.

The Real seal E is terminal for the displayed route through D, T, W, R. Componentwise transport H, replay C, provenance P, and local naming N preserve that same packet. Hence the refinement keeps both the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{Real}^{\uparrow}$ seal handoff on the original finite route. □

Falsifiable prediction: Every $\mathsf{RegularCauchyNet}^{\uparrow}$ Real-facing read factors through directed index rows, tail-filter rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: cauchynet, cauchy-net-filter-correspondence, separated-cauchy-net-completion, streamname, regseqrat, dyadicratcore, real: $\mathsf{RegularCauchyNet}^{\uparrow}$ binds directed-tail refinements, induced tail filters, finite observation windows, regular rational readback, terminal Real sealing, transport, replay, provenance, and local naming in one packet; each named sibling lacks at least one of these rows as a local directed-tail coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyNet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6338.1, Definition 6338.2, and Definition 6338.3 seed the finite directed-tail route from CauchyNet data through regular rational readback and the terminal Real seal.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient nets, selected subnets, selected limits, countable choice, ambient separated completion, host equality of real names, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, directed-tail exactness, regular readback, Real-seal nonescape, completion-consumer provenance, and tail-stability for BEDC.Derived.RegularCauchyNetUp.

Constructive story: A $\mathsf{RegularCauchyNet}^{\uparrow}$ packet is generated from directed index rows, tail-filter rows, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6339

A Concrete Naming Certificate for $\mathsf{BesicovitchCovering}^{\uparrow}$

Following Besicovitch's covering theorem and Bishop-style finite covering practice, the $\mathsf{BesicovitchCovering}^{\uparrow}$ packet records the finite-multiplicity ball-selection route used before differentiation, maximal-function, and metric-dimension consumers read a metric cover. It sits after the disjoint interval ledger of chapter 2184, the finite shadow-cover route of chapter 2166, the scale-control packet of chapter 2035, the gauge-cover surface of chapter 4119, and the domination surface of chapter 3581. The packet names only displayed metric balls, finite chosen subfamilies, bounded-overlap ledgers, transport, replay, provenance, and local naming; it does not import a host covering theorem, selected countable subfamilies, measure completion, quotient point equality, or ambient compactness.

Definition 6339.1 (Besicovitch finite-multiplicity covering carrier)**.** *A* $\mathsf{BesicovitchCovering}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$B = (X, M, R, Q, S, O, V, F, D, H, C, P, N)$$

with the following displayed rows:

1. *X is the point-history carrier and classifier row for the metric source.*
2. *M is the* $\mathsf{Met}^{\uparrow}$ *or* $\mathsf{Met}^{\uparrow}$ *distance row over X, supplying ball membership and distance comparisons.*
3. *R is the positive-radius ledger, read through* $\mathsf{Rat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *rows before any ball is admitted.*
4. *Q is the finite request bundle of target points and admissible ball candidates.*
5. *S is the finite selected subfamily of balls, stored as a closed* $\mathsf{ProbeBundle}$ *over the same source.*
6. *O is the bounded-overlap ledger, recording the finite multiplicity bound for the selected balls at every displayed inspection point.*
7. *V is the* $\mathsf{VitaliCover}^{\uparrow}$ *handoff row, exposing the disjoint-selection discipline that the request bundle may cite.*
8. *F is the* $\mathsf{FiniteVitaliCover}^{\uparrow}$ *handoff row, exposing finite shadow coverage and enlargement control for the selected subfamily.*
9. *D is the* $\mathsf{DoublingMetric}^{\uparrow}$ *handoff row, making the local finite-growth bound available when a dimension or maximal-function consumer asks for uniform multiplicity.*

10. *H records componentwise* hsame *transport for $X, M, R, Q, S, O, V, F, D$ and the structural rows.*
11. *C records displayed* Cont *replay from the ball requests through selection, bounded overlap, and the sibling handoff rows.*
12. *P is the* Pkg *provenance over* BesicovitchCovering$^{\uparrow}$, VitaliCover$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, DoublingMetric$^{\uparrow}$, HausdorffDimension$^{\uparrow}$, HardyLittlewoodMaximal$^{\uparrow}$, Met$^{\uparrow}$, Met$^{\uparrow}$, Rat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, *and* NameCert.
13. *N is the local* NameCert$_{\text{BesicovitchCovering}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these rows. It has no coordinate for an ambient locally finite cover, a selected countable subcover, a host differentiation theorem, measure completion, quotient point equality, open-cover compactness, countable choice, or ambient Real$^{\uparrow}$ *completeness.*

Theorem 6339.2 (Besicovitch covering NameCert obligations)**.** *For every accepted* BesicovitchCovering$^{\uparrow}$ *carrier*

$$B = (X, M, R, Q, S, O, V, F, D, H, C, P, N),$$

the local NameCert$_{\text{BesicovitchCovering}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric source, positive-radius ledger, finite request bundle, selected ball subfamily, bounded-overlap ledger, Vitali handoff, finite Vitali shadow-cover handoff, doubling-metric handoff, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local naming. No obligation is stored outside $X, M, R, Q, S, O, V, F, D, H, C, P, N$.*

Proof. Project the carrier of Definition 6339.1. The source rows X and M provide the metric points, distances, and ball-membership reads, while R supplies the positive-radius data used by every candidate ball.

Open the covering rows. The request bundle Q names the finite ball candidates, S names the selected subfamily, and O records the finite multiplicity bound visible at each displayed inspection point.

The sibling handoffs are then exhausted rowwise. The row V exposes only the Vitali disjoint-selection route, F exposes only finite shadow coverage, and D exposes only the doubling-metric finite-growth route.

Finally H, C, P, N transport, replay, source, and name the same route. Since the carrier contains none of the excluded host-cover, choice, measure-completion, quotient, compactness, differentiation, or ambient-completeness coordinates, the listed rows exhaust the local NameCert obligations. □

Theorem 6339.3 (Besicovitch covering finite-multiplicity route)**.** *Every* HausdorffDimension$^{\uparrow}$ *or* HardyLittlewoodMaximal$^{\uparrow}$ *consumer reading an accepted* BesicovitchCovering$^{\uparrow}$ *packet factors through the finite route*

Met$^{\uparrow}$, VitaliCover$^{\uparrow}$, FiniteVitaliCover$^{\uparrow}$, DoublingMetric$^{\uparrow}$, ProbeBundle, BHist, hsame, Cont, Pkg, N

Dimension-facing consumers receive only displayed ball radii, selected finite subfamilies, and multiplicity bounds before comparing with gauge-cover data. Maximal-function consumers receive only the same finite multiplicity ledger before using domination rows. The route exports no selected countable cover, no measure-completion theorem, no host differentiation argument, no quotient point equality, and no ambient compactness principle.

Proof. Start with a dimension-facing consumer. It can enter the packet only through the metric and radius rows X, M, R, then through the finite request and selection rows Q, S before the multiplicity ledger O is visible.

Next compare with the dimension surface. The $\mathsf{HausdorffDimension}^{\uparrow}$ consumer may read O as a finite overlap bound for gauge-cover bookkeeping, but it receives no Hausdorff-measure construction or selected countable family from the Besicovitch packet.

Now inspect a maximal-function consumer. The $\mathsf{HardyLittlewoodMaximal}^{\uparrow}$ route may cite the same selected balls and bounded overlap through S, O, F, while the Vitali and finite Vitali handoffs V, F keep the domination step inside displayed finite data.

The doubling row D supplies only local finite-growth control, and H, C, P, N preserve the route under transport, replay, provenance, and local naming. Therefore every dimension or maximal-function use remains a finite-multiplicity handoff rather than an ambient covering theorem. □

Falsifiable prediction: Every accepted $\mathsf{BesicovitchCovering}^{\uparrow}$ consumer read displays one metric source, one positive-radius ledger, one finite ball-request bundle, one selected finite subfamily, one bounded-overlap ledger, Vitali and finite Vitali handoff rows, one doubling-metric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: vitali-cover, finite-vitali-cover, doubling-metric, hausdorff-dimension, hardy-littlewood-maximal: $\mathsf{BesicovitchCovering}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite ball requests, selected subfamilies, bounded multiplicity, Vitali selection, finite shadow coverage, doubling control, and downstream dimension or maximal-function handoff rows in one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BesicovitchCovering}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BesicovitchCovering}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6339.1, Definition 6339.2, and Definition 6339.3 seed the finite-multiplicity covering packet over displayed metric balls, Vitali and finite Vitali handoffs, doubling control, Hausdorff-dimension consumers, and Hardy–Littlewood maximal consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BesicovitchCovering}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close the full Besicovitch covering theorem, exact Euclidean multiplicity constants, differentiation theorems, measure completion, selected countable subcovers, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, positive-radius exactness, selected-subfamily admission, bounded-overlap exactness, Vitali handoff, finite Vitali handoff, doubling-metric handoff, transport, replay, provenance, and local NameCert rows for BEDC.Derived.BesicovitchCoveringUp.

Constructive story: A $\mathsf{BesicovitchCovering}^{\uparrow}$ packet is generated from a metric source row, positive-radius ledger, finite ball-request bundle, selected finite subfamily, bounded-overlap ledger, Vitali and finite Vitali handoff rows, doubling-metric handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6340

A Concrete Naming Certificate for $\mathsf{ReducedProduct}^{\uparrow}$

Following Chang and Keisler on reduced products in model theory, the $\mathsf{ReducedProduct}^{\uparrow}$ packet records the finite BEDC naming surface for a family of first-order models read through a displayed filter. It sits after the membership discipline of $\mathsf{SetLike}^{\uparrow}$, the carrier discipline of $\mathsf{TypeLike}^{\uparrow}$, the syntactic rows of chapter 197, the satisfaction rows of chapter 198, and the local $\mathsf{NameCert}$ carrier discipline. The packet exposes model-family rows, a filter row, pointwise formula reads, a filter-visible agreement classifier, transport, replay, provenance, and local naming; it does not assert an ultrafilter choice, quotient equality of functions, Los theorem, compactness, or any host model product.

Definition 6340.1 (Reduced product carrier)**.** *A* $\mathsf{ReducedProduct}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (M, F, \Phi, A, H, C, P, N)$$

with the following displayed rows:

1. *M is the model-family source row, naming the finite family window and the associated* $\mathsf{ModelTheory}^{\uparrow}$ *structure rows available to a consumer.*
2. *F is the displayed filter row, recording the finite filter tokens, containment reads, and meet-closure witnesses used by the reduced-product route.*
3. *Φ is the pointwise formula row, reading* $\mathsf{FirstOrder}^{\uparrow}$ *formula endpoints and their coordinatewise* $\mathsf{ModelTheory}^{\uparrow}$ *satisfaction rows before any agreement comparison is public.*
4. *A is the filter-visible agreement classifier: two pointwise reads classify together only when the displayed filter row F marks the coordinate set on which the pointwise formula rows agree.*
5. *H stores componentwise* hsame *transport for the model-family source, displayed filter, formula row, agreement classifier, replay row, provenance row, and local naming row.*
6. *C stores displayed* Cont *replay from a reduced-product consumer through M, F, Φ, A, H before any local name is exported.*
7. *P records* Pkg *provenance over* $\mathsf{ReducedProduct}^{\uparrow}$, $\mathsf{SetLike}^{\uparrow}$, $\mathsf{TypeLike}^{\uparrow}$, $\mathsf{FirstOrder}^{\uparrow}$, $\mathsf{ModelTheory}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{ReducedProduct}^{\uparrow}}$ *row.*

Accepted carriers are compared only through $M, F, \Phi, A, H, C, P, N$. There is no coordinate for a chosen ultrafilter, quotient equality of functions, Los theorem, host compactness, or an ambient product model outside the displayed rows.

Lean4 (definition) `BEDC.Derived.ReducedProductUp`

Theorem 6340.2 (Reduced product NameCert obligations)**.** *For every accepted* $\mathsf{ReducedProduct}^{\uparrow}$ *carrier* $Q = (M, F, \Phi, A, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{ReducedProduct}^{\uparrow}}$ *surface consists exactly of the following obligations:*

1. *carrier admission for the displayed model-family source M, filter row F, pointwise formula row Φ, agreement classifier A, transport H, replay C, provenance P, and local naming N;*
2. *displayed filter exposure, requiring every reduced-product agreement read to expose F before the classifier A is read;*
3. *pointwise model-row stability, requiring every formula read in Φ to remain attached to the* $\mathsf{FirstOrder}^{\uparrow}$ *and* $\mathsf{ModelTheory}^{\uparrow}$ *rows named by M;*
4. *classifier transport, requiring filter-visible agreement to move only through componentwise* hsame *transport H and displayed replay C;*
5. *replay and provenance, requiring every downstream consumer to cite P and N after the route M, F, Φ, A, H, C has been displayed;*
6. *refusal of ultrafilter choice, quotient equality, Los theorem, and host compactness, because none of these objects appears as a carrier coordinate.*

Proof. Carrier admission is projection from Definition 6340.1. The tuple lists exactly the model-family source, displayed filter, pointwise formula row, agreement classifier, transport, replay, provenance, and local naming rows.

Displayed filter exposure is the second projection step. A consumer cannot read the agreement classifier A until the filter row F has named the finite coordinate set on which the pointwise formula rows are compared.

Pointwise model-row stability follows from the source row M and formula row Φ. Formula endpoints are read through $\mathsf{FirstOrder}^{\uparrow}$, and their satisfaction rows are read through $\mathsf{ModelTheory}^{\uparrow}$, so the reduced-product packet has no separate syntax or satisfaction source.

Finally use H, C, P, N. The transport and replay rows preserve the same displayed route, while provenance and local naming attach the route to the BEDC carrier. Since no coordinate stores an ultrafilter choice, quotient equality, Los theorem, host compactness, or an ambient product model, the local certificate cannot export those principles. □

Falsifiable prediction: Every accepted $\mathsf{ReducedProduct}^{\uparrow}$ agreement read exposes a model-family source row, displayed filter row, pointwise formula row, filter-visible classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row.

Independence witness: setlike, typelike, firstorder, modeltheory: $\mathsf{ReducedProduct}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a displayed filter row and a filter-visible pointwise agreement classifier over model-family and formula rows. SetLikeUp supplies membership without model satisfaction, TypeLikeUp supplies fiber transport without the filter-visible classifier, FirstOrderUp supplies syntax without model-family satisfaction, and ModelTheoryUp supplies satisfaction without the reduced-product filter row.

Closure status for $\mathsf{NameCert}_{\mathsf{ReducedProduct}^{\uparrow}}$**.**

Theory closure: TheoryGate(NameCert$_{\mathsf{ReducedProduct}^{\uparrow}}$, seed)

Scope closed: Definition 6340.1 and Definition 6340.2 seed the finite model-family, displayed-filter, pointwise-formula, agreement-classifier, transport, replay, provenance, and local-name rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{ReducedProduct}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close ultrafilter choice, quotient equality of functions, Los theorem, compactness, host model products, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displayed filter exposure, pointwise model-row stability, classifier transport, replay, provenance, and refusal boundaries for BEDC.Derived.ReducedProductUp.

Constructive story: A ReducedProduct$^{\uparrow}$ packet is generated from a model-family source row, displayed filter row, pointwise formula row, filter-visible agreement classifier, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6341

A Concrete Naming Certificate for $\mathsf{AronszajnTree}^{\uparrow}$

Following Aronszajn, the $\mathsf{AronszajnTree}^{\uparrow}$ packet records the classical height-ω_1 tree obstruction as finite BEDC order data. It sits over the $\mathsf{Tree}^{\uparrow}$ source of chapter 119, the $\mathsf{POSet}^{\uparrow}$ comparison surface of chapter 53, the unary $\mathsf{Order}^{\uparrow}$ read of chapter 1378, the membership discipline of chapter 111, the countable-selection boundary of chapter 1872, and the ordinal-analysis ledger of chapter 6247. The packet names countable displayed levels, a no-uncountable-branch boundary, and finite branch-height readback; it does not assert Suslinness, Kurepa splitting, a club selector, or a host set-theoretic construction of ω_1.

Definition 6341.1 (Aronszajn tree carrier)**.** *An* $\mathsf{AronszajnTree}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (T, O, L, B, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Tree}^{\uparrow}$ *source row, read through the same graph and incidence surface as chapter 119.*
2. *O is the* $\mathsf{POSet}^{\uparrow}$ *tree-order row, including parent-extension comparisons, initial-segment comparability, and the inherited* $\mathsf{Order}^{\uparrow}$ *comparison route.*
3. *L is the countable-level ledger. For each displayed height request it records a finite membership window in the level and a countability witness boundary, without selecting a global enumeration of all levels.*
4. *B is the branch-height boundary row. It records that every displayed branch read is routed through a finite height window and cannot be consumed as an ω_1-long branch.*
5. *H records componentwise* hsame *transport for the tree source, order row, level ledger, branch-height boundary, replay, provenance, and local naming rows.*
6. *C records displayed* Cont *replay from tree incidence reads through order comparison, level membership, countability, and branch-height boundary rows.*
7. *P is the* Pkg *provenance over* $\mathsf{AronszajnTree}^{\uparrow}$, $\mathsf{Tree}^{\uparrow}$, $\mathsf{POSet}^{\uparrow}$, $\mathsf{Order}^{\uparrow}$, $\mathsf{Set}^{\uparrow}$, $\mathsf{CountableChoiceBoundary}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
8. *N is the local* $\mathsf{NameCert}_{\mathsf{AronszajnTree}^{\uparrow}}$ *row.*

The carrier predicate asserts exactly that these rows form one finite Aronszajn-tree packet. It has no coordinate for a Suslin antichain theorem, Kurepa-many branches, a club subset of ω_1, a global choice selector, or a host-level ordinal construction.

Theorem 6341.2 (Aronszajn tree NameCert obligations)**.** *For every accepted* $\mathsf{AronszajnTree}^{\uparrow}$ *carrier* $A = (T, O, L, B, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{AronszajnTree}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed tree source, tree-order row, countable-level ledger, branch-height boundary, transport, replay, provenance, and local naming rows;*
2. *order-source discipline, requiring every branch and level read to use the same* $\mathsf{Tree}^{\uparrow}$ *source* T *and the same* $\mathsf{POSet}^{\uparrow}$ *comparison row* O*;*
3. *countable-level visibility, requiring each displayed level request to pass through the finite ledger* L *and the countable-choice boundary before any consumer reads level data;*
4. *branch-height nonescape, requiring every displayed branch request to pass through* B *before it can be consumed by order, set, or ordinal-analysis consumers;*
5. *set-theoretic boundary discipline, excluding Suslinness, Kurepa splitting, club selectors, host well-order construction, and unrecorded choice from the packet.*

No Aronszajn-tree obligation is stored outside these rows.

Proof. Project the carrier of Definition 6341.1. The projection exposes exactly the tree source T, tree-order row O, countable-level ledger L, branch-height boundary B, and structural rows H, C, P, N.

The order-source and countable-level obligations are tied to the same displayed route. A level read first sees the tree incidence data in T, then the order comparison in O, and only then the countable-level ledger L with its finite membership window.

The branch-height obligation is separate from the level ledger. The row B is reached only after the tree and order rows have been read, and it records a boundary on displayed branch consumption rather than a host proof about all branches.

The remaining exclusions follow from the carrier shape. None of the rows stores a Suslin antichain theorem, Kurepa splitting family, club selector, host well-order construction, or unrecorded choice row, so the listed obligations exhaust the local NameCert surface. □

Theorem 6341.3 (Aronszajn tree branch-height nonescape)**.** *Every order, set, or ordinal-analysis consumer of an accepted* $\mathsf{AronszajnTree}^{\uparrow}$ *carrier* $A = (T, O, L, B, H, C, P, N)$ *reads branch information through the finite route*

$$\mathsf{Tree}^{\uparrow} \longmapsto \mathsf{POSet}^{\uparrow} \longmapsto L \longmapsto B \longmapsto H, C, P, N.$$

Thus a branch-height read exposes the tree source, partial-order comparison, countable-level ledger, branch-height boundary, transport, replay, provenance, and local naming before any consumer-facing conclusion is exported. It cannot be replaced by an ω_1-long branch, a global level enumeration, a club selector, a Suslin antichain claim, or a Kurepa branch family.

Proof. Begin with the tree source. The carrier admits a branch read only after T displays the incidence and tree rows inherited from $\mathsf{Tree}^{\uparrow}$.

Read the order row next. The row O supplies parent-extension and initial-segment comparison through $\mathsf{POSet}^{\uparrow}$ and $\mathsf{Order}^{\uparrow}$, so branch comparability is a displayed finite order route rather than host membership in an ambient tree.

Now inspect the countable-level ledger. The row L records the finite level window and countability boundary needed before a branch-height request reaches B.

Finally the boundary row B and the structural rows H, C, P, N package the same route for downstream consumers. Since the carrier has no coordinate for an ω_1-long branch, global enumeration, club selector, Suslin antichain theorem, or Kurepa family, no such object can be read from the packet. □

Theorem 6341.4 (Aronszajn tree sibling route)**.** *The* $\mathsf{AronszajnTree}^{\uparrow}$ *packet is read by sibling consumers only through the displayed route*

$$\mathsf{Tree}^{\uparrow}, \quad \mathsf{POSet}^{\uparrow}, \quad \mathsf{Order}^{\uparrow}, \quad \mathsf{Set}^{\uparrow}, \quad \mathsf{CountableChoiceBoundary}^{\uparrow}, \quad \mathsf{OrdinalAnalysis}.$$

The first four rows supply tree incidence, order comparison, unary comparison, and membership visibility; the countable-choice boundary restricts level reads; the ordinal-analysis ledger records height requests. The packet does not collapse these sibling reads into a host set-theoretic object.

Proof. Use Definition 6341.2. It fixes the carrier, order-source, countable-level, and branch-height rows as the complete local obligation surface.

Use Definition 6341.3. It places every branch-height consumer behind the same tree, order, level, and boundary rows.

The provenance row P then names the sibling sources and the replay row C preserves their order. Therefore any consumer must read the displayed sibling route rather than an external tree, set, choice, or ordinal object. □

Falsifiable prediction: Every accepted $\mathsf{AronszajnTree}^{\uparrow}$ branch-height read factors through the displayed $\mathsf{Tree}^{\uparrow}$, $\mathsf{POSet}^{\uparrow}$, countable-level, branch-boundary, transport, replay, provenance, and local NameCert rows before any order, set, or ordinal-analysis consumer reads it.

Independence witness: tree, poset, order, set, countable-choice-boundary, ordinal-analysis: $\mathsf{AronszajnTree}^{\uparrow}$ is not bijective to any listed sibling because it jointly binds a tree source, tree-order comparison, countable-level ledger, branch-height boundary, componentwise transport, replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{AronszajnTree}^{\uparrow}}$**.**

Constructive story: An $\mathsf{AronszajnTree}^{\uparrow}$ packet is generated from a displayed TreeUp source, POSetUp tree-order row, countable-level ledger, branch-height boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AronszajnTree}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6341.1, Definition 6341.2, Definition 6341.3, and Definition 6341.4 seed the finite tree-obstruction NameCert route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AronszajnTree}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, Suslinness, Kurepa splitting, a club selector, host construction of omega-one, global level enumeration, or uncountable-choice principles.

Upgrade path: Obligation closure requires checked carrier admission, order-source discipline, countable-level visibility, branch-height nonescape, sibling route readback, and ledger exhaustion for BEDC.Derived.AronszajnTreeUp.

Chapter 6342

A Concrete Naming Certificate for $\mathsf{ExtFunctor}^{\uparrow}$

Following Cartan–Eilenberg and Weibel on derived functors over abelian categories, the $\mathsf{ExtFunctor}^{\uparrow}$ packet names the boundary where right-derived Hom data becomes a finite BEDC NameCert surface. It sits after the abelian-category carrier of chapter 176, the derived-functor spine of chapter 177, and the cycle-boundary readback of chapter 99 and chapter 98. The packet records Ext as a displayed derived-functor route through resolution, Hom-row, cohomology, transport, replay, provenance, and local naming rows; it is not a host quotient of complexes, not a chosen injective or projective resolution outside the ledger, not an abelian-category oracle, and not a classical exactness theorem hidden behind the name.

Definition 6342.1 (ExtFunctor carrier)**.** *An* $\mathsf{ExtFunctor}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (A, X, Y, R, \mathcal{H}, C^{\bullet}, Z, B, D, H, T, P, N)$$

with the following displayed rows:

1. *A is the* $\mathsf{AbelianCat}^{\uparrow}$ *source row naming the abelian-category carrier in which objects, morphisms, kernels, cokernels, and exactness reads are interpreted.*
2. *X and Y are the two object rows for the traditional* $\mathrm{Ext}^n(X, Y)$ *request.*
3. *R is the displayed resolution row. It may be read as an injective resolution of Y or a projective resolution of X, but the carrier records the chosen side explicitly as finite* BHist *data.*
4. *$\mathcal{H}$ is the displayed Hom-complex row obtained by applying the Hom request to the resolution row inside the same abelian-category source.*
5. *$C^{\bullet}$ is the cochain ledger whose differential and degree rows are consumed by* $\mathsf{Cohomology}^{\uparrow}$*.*
6. *Z and B are the cycle and boundary rows at the displayed degree.*
7. *D is the* $\mathsf{DerivedFunctor}^{\uparrow}$ *handoff row identifying the Ext read as the right-derived Hom route rather than as a separate quotient object.*
8. *H records componentwise* hsame *transport for the abelian source, object rows, resolution row, Hom-complex row, cochain ledger, cycle-boundary rows, derived-functor handoff, continuation rows, provenance, and local naming.*
9. *T records displayed* Cont *replay from the Ext request through $A, X, Y, R, \mathcal{H}, C^{\bullet}, Z, B, D, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{ExtFunctor}^{\uparrow}$, $\mathsf{AbelianCat}^{\uparrow}$, $\mathsf{DerivedFunctor}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, $\mathsf{Homology}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{ExtFunctor}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $A, X, Y, R, \mathcal{H}, C^{\bullet}, Z, B, D, H, T, P, N$. It has no coordinate for host quotient equality of complexes, unrecorded resolution replacement, selected representatives of cohomology classes, global abelian-category exactness, countable choice, or an ambient derived-category oracle.

Theorem 6342.2 (ExtFunctor NameCert obligations)**.** *For every accepted* $\mathsf{ExtFunctor}^{\uparrow}$ *carrier $E = (A, X, Y, R, \mathcal{H}, C^{\bullet}, Z, B, D, H, T, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ExtFunctor}^{\uparrow}}$ *surface consists exactly of carrier admission for the abelian source A, object rows X, Y, displayed resolution row R, Hom-complex row $\mathcal{H}$, cochain ledger $C^{\bullet}$, cycle and boundary rows Z, B, derived-functor handoff D, componentwise transport H, displayed replay T, package provenance P, and local naming row N.*

Every Ext consumer reads the public class through the displayed chain

$$\mathsf{AbelianCat}^{\uparrow} \longmapsto (X, Y, R) \longmapsto \mathcal{H} \longmapsto C^{\bullet} \longmapsto (Z, B) \longmapsto \mathsf{DerivedFunctor}^{\uparrow}.$$

The certificate exports no quotient complex, unrecorded resolution, selected cohomology representative, global exactness oracle, countable-choice schedule, or ambient derived-category equality.

Proof. Project the carrier of Definition 6342.1. The first visible rows are the abelian source, the two object rows, and the resolution row, so any Hom-complex request is already tied to displayed abelian-category data.

Read the Hom-complex row $\mathcal{H}$ and the cochain ledger $C^{\bullet}$ next. The cycle and boundary rows Z, B are consumed by the cohomology or homology certificate at the displayed degree, and the handoff row D records that the resulting class is the right-derived Hom read inside $\mathsf{DerivedFunctor}^{\uparrow}$.

Finally H, T, P, N transport, replay, source, and name exactly those rows. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert surface and no hidden quotient, selected representative, exactness oracle, choice object, or ambient derived-category equality can enter the Ext read. □

Theorem 6342.3 (ExtFunctor derived-functor handoff)**.** *Every* $\mathsf{DerivedFunctor}^{\uparrow}$, $\mathsf{Cohomology}^{\uparrow}$, *or* $\mathsf{Homology}^{\uparrow}$ *consumer of an accepted* $\mathsf{ExtFunctor}^{\uparrow}$ *packet receives only the displayed abelian source A, object rows X, Y, resolution row R, Hom-complex row $\mathcal{H}$, cochain ledger $C^{\bullet}$, cycle-boundary rows Z, B, derived-functor handoff D, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row.*

In particular, a consumer may change from an injective-resolution presentation to a projective-resolution presentation only by displaying a finite comparison ledger through $R, \mathcal{H}, C^{\bullet}, Z, B, D, H, T, P, N$. Such a comparison is a BEDC route equivalence; it is not host equality of resolutions, not a selected quasi-isomorphism inverse, and not an imported derived-category isomorphism.

Proof. Begin with Definition 6342.2. It places the Ext read behind the abelian-category source, object rows, displayed resolution, Hom-complex row, cochain ledger, cycle-boundary rows, and derived-functor handoff.

For the ordinary derived-functor direction, project D. This row identifies the accepted packet as the derived Hom route and hands the same cycle-boundary data to the downstream $\mathsf{DerivedFunctor}^{\uparrow}$ consumer.

For a change of resolution side, the carrier must expose the comparison as finite data in the resolution, Hom-complex, cochain, cycle-boundary, and replay rows. The comparison can therefore be consumed by $\mathsf{Cohomology}^{\uparrow}$ or $\mathsf{Homology}^{\uparrow}$ only after those rows are displayed.

Thus every downstream consumer sees the same finite ledger and no hidden resolution equality, selected inverse, quotient complex, global exactness theorem, countable-choice schedule, or ambient derived-category oracle. □

Closure status for $\mathsf{NameCert}_{\mathsf{ExtFunctor}\uparrow}$.

Constructive story: An $\mathsf{ExtFunctor}^{\uparrow}$ packet is generated from an AbelianCat source row, two object rows, a displayed resolution row, a Hom-complex row, a cochain ledger, cycle and boundary rows, a DerivedFunctor handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ExtFunctor}\uparrow}, \mathsf{seed})$

Scope closed: Definition 6342.1, Definition 6342.2, and Definition 6342.3 seed the finite Ext derived-functor boundary over abelian, cohomology, homology, and derived-functor rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ExtFunctor}\uparrow}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host quotient equality of complexes, unrecorded resolution comparison, selected cohomology representatives, global abelian-category exactness, countable choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, resolution-side comparison coverage, Hom-complex ledger exactness, cycle-boundary classifier stability, derived-functor handoff confinement, transport stability, replay, and provenance nonescape for BEDC.Derived.ExtFunctorUp.

Chapter 6343

A Concrete Naming Certificate for $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion, the $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ packet records the finite realizer surface that turns a uniform completion datum into a real-name seal. It sits after the functorial completion route of chapter 1892, the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback route of chapter 301, and the terminal real-name seal of chapter 38. The packet supplies a displayed modulus-governed realizer for L10 completion reads without importing quotient completions, selected limits, host complete spaces, choice, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Definition 6343.1 (UniformCauchyCompletionRealizer carrier)**.** *A* $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$U = (F, D, W, R, M, E, H, C, P, N)$$

with the following visible rows:

1. *F is the* $\mathsf{UniformCompletionFunctor}^{\uparrow}$ *source row, naming the displayed completion datum and its finite functorial handoff;*
2. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used to state all precision requests;*
3. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row that exposes the finite prefixes inspected at each tolerance;*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row supplying regular rational approximants for those windows;*
5. *M is the modulus row, recording that a dyadic tolerance request selects a displayed finite window before the completion-facing read is exported;*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached from the regular rational readback and the modulus ledger;*
7. *H records componentwise* hsame *transport for F, D, W, R, M, E;*
8. *C records displayed* Cont *replay from the uniform completion datum through the dyadic, stream, regular, modulus, and real-seal rows;*
9. *P is the* Pkg *provenance over* $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$, $\mathsf{UniformCompletionFunctor}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotient completion equality, a selected limit point, a host complete metric space, countable choice, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 6343.2 (UniformCauchyCompletionRealizer NameCert obligations)**.** *For every accepted* $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ *carrier*

$$U = (F, D, W, R, M, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}}$ *surface consists exactly of carrier admission for* U*, uniform completion datum exposure through* F*, dyadic tolerance visibility through* D*, finite stream-window visibility through* W*, regular rational readback through* R*, modulus selection through* M*, terminal* $\mathsf{Real}^{\uparrow}$ *seal readback through* E*, componentwise* hsame *transport through* H*, displayed* Cont *replay through* C*, package provenance through* P*, and local naming through* N*.*

Every completion-facing consumer must therefore read the displayed finite rows before it can use the realizer as a Real-facing output.

Proof. Project the accepted carrier through Definition 6343.1. The packet displays the rows $F, D, W, R, M, E, H, C, P, N$, and each row has a fixed local role in the carrier.

Read the analytic route first. The row F supplies the uniform completion datum, D states dyadic precision, W exposes the finite stream window requested by that precision, R supplies the regular rational readback, M records the modulus selecting the window, and E is the resulting Real seal.

The support rows do not add analytic content. Componentwise hsame transport preserves the same displayed rows, Cont replay keeps their inspection order, Pkg records the cited sibling provenance, and N names the local certificate. Since the carrier contains no additional coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6343.3 (UniformCauchyCompletionRealizer modulus stability)**.** *Let* $U = (F, D, W, R, M, E, H, C, P, N)$ *and* $U' = (F', D', W', R', M', E', H', C', P', N')$ *be accepted* $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ *carriers. If the dyadic tolerance rows* D, D'*, stream-window rows* W, W'*, regular readback rows* R, R'*, and modulus rows* M, M' *agree under the displayed componentwise* hsame *transport, then every completion-facing consumer reads the same finite modulus route before reaching the terminal* $\mathsf{Real}^{\uparrow}$ *seal.*

In particular, transport cannot replace the displayed modulus by a selected limit, a quotient-completion equality, or a host completeness argument.

Proof. Begin with the rows controlled by the hypothesis. Agreement of D, D' fixes the dyadic precision ledger, agreement of W, W' fixes the finite windows, agreement of R, R' fixes the regular rational readbacks, and agreement of M, M' fixes the modulus row that sends each precision request to its displayed window.

Apply componentwise transport through H and H'. These rows preserve the analytic coordinates already named and do not introduce a new completion object or a selector for limits.

Finally replay the route through C and C'. A completion-facing consumer reaches E or E' only after the same tolerance, window, readback, and modulus rows have been inspected. Therefore the consumer sees the same finite modulus route, and the excluded quotient, selection, host-space, and completeness readings remain outside the packet. □

Theorem 6343.4 (UniformCauchyCompletionRealizer Real-seal handoff)**.** *Every* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ *carrier*

$$U = (F, D, W, R, M, E, H, C, P, N)$$

reaches the terminal seal E only through the route

$$F \longmapsto D \longmapsto W \longmapsto R \longmapsto M \longmapsto E$$

and then through the structural rows H, C, P, N. The handoff exports a finite Real-facing seal from the uniform completion datum; it exports no selected limit, no quotient completion equality, no host complete-space object, no choice principle, and no ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Lean4 ✓ `BEDC.Derived.UniformCauchyCompletionRealizerUp.UniformCauchyCompletionRealizerCarrier real seal handoff`

Proof. Read the carrier rows in their displayed order. The uniform completion datum F is not itself a Real seal; it must first expose the dyadic tolerance row D, the finite stream window W, the regular rational readback R, and the modulus row M.

Only after those rows are visible does the consumer reach E. The continuation row C records exactly this replay order, while H preserves it under componentwise transport and P, N package and name the same local surface.

The carrier definition supplies no coordinate for a host completed object, quotient equality of completion names, selected limit, countable-choice schedule, or ambient $\mathsf{Real}^{\uparrow}$ completeness. Hence the Real-facing output is precisely the displayed finite seal handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}}$.

Constructive story: A $\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}$ packet is generated from uniform completion data, dyadic tolerance rows, finite stream windows, regular rational readback, a modulus row, a Real seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6343.1, Definition 6343.2, Definition 6343.3, and Definition 6343.4 bind the finite carrier, NameCert obligation surface, modulus stability, and Real-seal handoff for the L10 completion realizer route.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UniformCauchyCompletionRealizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected limits, quotient completion equality, host complete spaces, choice, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, modulus-row stability, Real-seal handoff coverage, and a Lean realization for BEDC.Derived.UniformCauchyCompletionRealizerUp.

Chapter 6344

A Concrete Naming Certificate for $\mathsf{GateauxDerivative}^{\uparrow}$

Following Gateaux, Bourbaki, and Bishop–Bridges constructive analysis, the $\mathsf{GateauxDerivative}^{\uparrow}$ packet records directional differentiability as a finite NameCert surface. It sits between the pointwise derivative interface of chapter 122 and the full linear approximation interface of chapter 3634. Its displayed route consumes the linear direction row of chapter 48, the norm comparison row of chapter 89, and the real-tolerance seal of chapter 38. The packet names a derivative read in one displayed direction; it is not a host function-extensionality principle, not a chosen limit, not quotient equality of germs, and not ambient completeness of the real line.

6344.1 Directional carrier

Definition 6344.1 (Gateaux derivative carrier)**.** *A* $\mathsf{GateauxDerivative}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, A, V, Q, T, H, C, P, N)$$

with the following displayed rows:

1. *X is the normed source-and-target row through which points, steps, and error sizes are read.*
2. *A records the base point and the finite admissible window in which directional probes are accepted.*
3. *V is the displayed direction row. It names one admissible direction over X and exposes no global tangent bundle.*
4. *Q is the directional difference-quotient row. It records the finite quotients obtained by probing the function along the displayed direction V inside the window A.*
5. *T is the* $\mathsf{Real}^{\uparrow}$ *tolerance-agreement row. It compares accepted quotient reads against one candidate directional derivative value by finite real-name tolerance data.*
6. *H records componentwise* hsame *transport for X, A, V, Q, T and the structural rows.*
7. *C records displayed* Cont *replay from a directional derivative request through the source, point-window, direction, quotient, and tolerance rows.*
8. *P is the* Pkg *provenance over* $\mathsf{GateauxDerivative}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{FrechetDerivative}^{\uparrow}$, $\mathsf{LinMap}^{\uparrow}$, $\mathsf{Norm}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$ *rows.*

9. N is the local $\mathsf{NameCert}_{\mathsf{GateauxDerivative}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through the displayed direction row V and the real-tolerance agreement row T, together with the structural transport and replay rows. It has no coordinate for host function equality, unrecorded limits, quotient equality of directional germs, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6344.2 (Gateaux derivative classifier adequacy)**.** *For accepted* $\mathsf{GateauxDerivative}^{\uparrow}$ *carriers*

$$G = (X, A, V, Q, T, H, C, P, N) \quad \textit{and} \quad G' = (X', A', V', Q', T', H', C', P', N'),$$

the public classifier identifies G with G' exactly when the source rows, point-window rows, displayed direction rows, directional quotient rows, real-tolerance rows, transport rows, replay rows, provenance rows, and local naming rows are componentwise hsame*-stable, and the direction read V and tolerance agreement T transport to V' and T' over the same source row. The comparison does not inspect host functions, chosen limits, quotient germs, or completeness witnesses outside the packet.*

Proof. Project the two carrier tuples. Every accepted row is displayed in the finite packet, so componentwise hsame transport gives the only data available to the public classifier.

The direction clause is read from V and V'. Because these rows are attached to the same transported source row, a classifier equality can compare the direction displays without importing a tangent-space quotient or a host vector equality.

The tolerance clause is read from T and T'. It compares the quotient rows against the candidate derivative values only through $\mathsf{Real}^{\uparrow}$ tolerance data, so the classifier remains finite and local.

Conversely, the listed componentwise transport data reconstructs the accepted packet row by row. The provenance row records that no additional host coordinate is part of the name, hence the displayed rows exhaust the classifier. □

Theorem 6344.3 (Gateaux derivative NameCert obligations)**.** *For every accepted* $\mathsf{GateauxDerivative}^{\uparrow}$ *carrier $G = (X, A, V, Q, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{GateauxDerivative}^{\uparrow}}$ *surface consists exactly of carrier admission for the normed source row, point-window row, displayed direction row, directional difference-quotient row,* $\mathsf{Real}^{\uparrow}$ *tolerance-agreement row, componentwise transport, displayed replay, provenance, and local naming; classifier stability under* hsame *transport of all displayed rows; direction-row exposure before any derivative read is public; tolerance agreement through the same* $\mathsf{Real}^{\uparrow}$ *row for every accepted quotient request; and ledger non-escape from host function extensionality, chosen limits, quotient equality, and ambient real completeness.*

Proof. Read the carrier of Definition 6344.1. The tuple $X, A, V, Q, T, H, C, P, N$ is the whole accepted packet, so the carrier-admission obligations are direct projections.

The direction exposure clause is the read of V over the source row X and point window A. This keeps the Gateaux derivative directional: no Frechet-style full linear approximation is exported unless a separate $\mathsf{FrechetDerivative}^{\uparrow}$ packet supplies it.

The quotient and tolerance clauses are the reads of Q and T. The quotient row records finite directional probes, while the tolerance row compares them through $\mathsf{Real}^{\uparrow}$ data already present in the packet.

Classifier stability is the componentwise transport row H, replay is the continuation row C, and source accounting is the package row P. Since no coordinate names any excluded host object, the local certificate exports only the displayed directional derivative route. □

Theorem 6344.4 (Gateaux derivative local non-escape)**.** *An accepted* $\mathsf{GateauxDerivative}^{\uparrow}$ *packet cannot be used to read host function extensionality, a chosen limiting value, quotient equality of directional germs, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness. Any consumer of the packet must first pass through the finite route*

$$\mathsf{Derivative}^{\uparrow},\ \mathsf{FrechetDerivative}^{\uparrow},\ \mathsf{LinMap}^{\uparrow},\ \mathsf{Norm}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{BHist},\ \mathsf{hsame},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{NameCert}.$$

Within that route, $\mathsf{GateauxDerivative}^{\uparrow}$ *contributes only the displayed direction row, finite directional difference quotients, and real-tolerance agreement.*

Proof. Read a consumer request through the continuation row C. The replay exposes the source row X, the point-window row A, and the displayed direction row V before any quotient or derivative value is visible.

Next read the quotient row Q and the tolerance row T. These rows provide finite directional probes and $\mathsf{Real}^{\uparrow}$ tolerance agreement, not a selected limit object.

Finally read the provenance row P and the local naming row N. Their dependency route lists the neighbouring derivative, Frechet derivative, linear-map, norm, real, history, transport, continuation, package, and NameCert rows. Since host function extensionality, quotient-germ equality, chosen limits, and ambient completeness do not occur in that route, they are not exported by the packet. □

Falsifiable prediction: Every accepted $\mathsf{GateauxDerivative}^{\uparrow}$ packet must expose a displayed direction row, a directional difference-quotient row, and a $\mathsf{Real}^{\uparrow}$ tolerance-agreement row before any directional derivative read is public.

Independence witness: derivative, frechet, linearmap, norm, real: $\mathsf{GateauxDerivative}^{\uparrow}$ is not bijective to any listed sibling because it binds one displayed direction and finite real-tolerance agreement without requiring a full Frechet linear approximation or a host derivative operator.

Closure status for $\mathsf{NameCert}_{\mathsf{GateauxDerivative}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{GateauxDerivative}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6344.1, Definition 6344.2, Definition 6344.3, and Definition 6344.4 seed the finite Gateaux directional-derivative route over Derivative, FrechetDerivative, LinearMap, Norm, Real, BHist, hsame, Cont, Pkg, and local NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{GateauxDerivative}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host differentiability, full Frechet differentiability, global tangent bundles, quotient germ equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, direction-row exposure, directional quotient readback, real-tolerance agreement, classifier stability, local non-escape, provenance exhaustion, and TasteGate field faithfulness for BEDC.Derived.GateauxDerivativeUp.

Constructive story: A $\mathsf{GateauxDerivative}^{\uparrow}$ packet is generated from a normed source row, a point-window row, a displayed direction row, a directional quotient row, a Real tolerance-agreement row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6345

A Concrete Naming Certificate for $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ packet records the finite-net Lebesgue-number data needed before compact-metric finite-modulus consumers read a cover-stability tolerance. It sits over the compact metric source of chapter 129, the total-bounded refinement surface of chapter 127, the finite epsilon-net handoff of chapter 1767, the finite-net compactness route of chapter 3471, and the Real-facing tolerance surface of chapter 38. The packet records finite centers, cover-stability rows, tolerance comparisons, and replay provenance without claiming open-cover compactness, selected subcovers, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6345.1 (Lebesgue-number finite-net carrier)**.** *A* $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$L = (X, U, \delta, E, P, C, D, T, H, R, S, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing the point classifier, metric-distance classifier, compact-metric ledger, and finite-modulus consumer surface read by the packet.*
2. *U is the cover-stability request row. It records the finite family of named cover cells inspected by the packet, not an arbitrary host open cover.*
3. *δ is the positive tolerance row, read through* $\mathsf{Rat}^{\uparrow}$*,* $\mathsf{DyadicRatCore}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *only as displayed tolerance data.*
4. *E is the imported* $\mathsf{FiniteEpsilonNet}^{\uparrow}$ *handoff row, exposing center rows, membership rows, dyadic or rational tolerance rows, and the* $\mathsf{TotallyBounded}^{\uparrow}$ *refinement surface.*
5. *P is the closed* ProbeBundle *of finite-net centers read from E.*
6. *C records center admission and cover-cell assignment for every center in P.*
7. *D records the finite coverage ledger from each inspected point to an admitted center, together with the displayed distance comparison bounded by δ.*
8. *T records the finite cover-stability tolerance comparison: if an inspected point is δ-close to an admitted center, the point remains inside the cover cell assigned by C.*
9. *H records componentwise* hsame *transport for $X, U, \delta, E, P, C, D, T, R, S, N$.*

10. *R records displayed* Cont *replay for finite cover-stability requests.*
11. *S is the* Pkg *provenance over* $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{TotallyBounded}^{\uparrow}$, $\mathsf{FiniteEpsilonNet}^{\uparrow}$, $\mathsf{HeineBorelFiniteNet}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteNet}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through these finite rows. It has no coordinate for an arbitrary open cover, a selected finite subcover, a host compactness theorem, countable choice, or ambient Real completeness.

Theorem 6345.2 (Lebesgue-number finite-net NameCert obligations)**.** *For every accepted* LebesgueNumberFiniteN *carrier $L = (X, U, \delta, E, P, C, D, T, H, R, S, N)$, the local* $\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteNet}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier admission for the compact metric source, cover-stability request, positive tolerance, finite epsilon-net handoff, center bundle, center-cover assignment ledger, coverage-distance ledger, stability-tolerance ledger, transport, replay, provenance, and local naming rows;*
2. *finite-net support, requiring the center rows, membership rows, tolerance rows, and* $\mathsf{TotallyBounded}^{\uparrow}$ *refinement surface of E to be exposed before any cover-stability tolerance is read;*
3. *center assignment, requiring every center used by the stability ledger to be admitted through P and assigned a displayed cover cell by C;*
4. *coverage transport, requiring every inspected point to reach an assigned center through D and to preserve the same tolerance row under H;*
5. *local stability, requiring T to certify the displayed cover-cell containment after the inspected point is compared with its admitted center;*
6. *ledger nonescape, excluding arbitrary open covers, selected finite subcovers, host compactness theorems, countable choice, host* $\mathsf{Real}^{\uparrow}$ *equality, and ambient completeness as local certificate data.*

No Lebesgue-number finite-net obligation is stored outside these rows.

Proof. Project the carrier of Definition 6345.1. The analytic surface consists of the compact metric source X, cover-stability request U, tolerance row δ, finite epsilon-net handoff E, center bundle P, assignment ledger C, coverage ledger D, and stability ledger T.

Use Definition 1767.6. The imported row E exposes center rows, membership rows, rational or dyadic tolerance rows, and the $\mathsf{TotallyBounded}^{\uparrow}$ refinement surface before this packet reads a cover-stability tolerance.

The center assignment and coverage transport rows are then forced locally. A center can be used only after it is visible in P and assigned by C, and an inspected point can use the assigned cell only after D supplies a displayed distance comparison to that center.

Finally apply the stability ledger T and replay it through H and R. The replay remains inside the finite request surface, and S, N record provenance and local naming without adding any selected subcover, open-cover compactness theorem, countable-choice row, host Real equality, or ambient completeness principle. □

Theorem 6345.3 (Lebesgue-number finite-net nonescape)**.** *Every* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{FiniteModulusReal}^{\uparrow}$, *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ *carrier reads a finite route*

$$X, U, \delta, E, P, C, D, T \longmapsto H, R, S, N$$

before it may use a cover-stability tolerance. The route may be combined with the finite-net compactness surface of $\mathsf{HeineBorelFiniteNet}^{\uparrow}$*, but it does not export open-cover compactness, selected subcovers, countable choice, host Real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a downstream consumer of L. By Definition 6345.1, the consumer enters through X, U, δ and the finite epsilon-net row E, not through an ambient cover or host compactness theorem.

Apply Definition 6345.2. The packet opens the center bundle, center-cover assignment, coverage-distance ledger, stability-tolerance ledger, transport, replay, provenance, and local naming rows in that order.

The optional $\mathsf{HeineBorelFiniteNet}^{\uparrow}$ sibling route is only a finite-net compactness surface. It can align named finite-net rows with compact-metric consumers, but it supplies no selected subcover and no open-cover compactness principle to this packet.

Thus every consumer read factors through the displayed finite route before any cover-stability tolerance is available. Since the carrier lacks coordinates for selected subcovers, countable choice, host Real equality, and ambient completeness, none of those objects escapes through the consumer boundary. □

Closure status for $\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteNet}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteNet}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6345.1, Definition 6345.2, and Definition 6345.3 seed the finite-net Lebesgue-number packet over compact metric sources, cover-stability requests, tolerance rows, finite epsilon-net handoffs, center bundles, assignment ledgers, coverage ledgers, stability ledgers, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LebesgueNumberFiniteNet}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close open-cover compactness, selected subcovers, countable choice, host Real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, finite-net support, center assignment, coverage transport, local stability, ledger nonescape, and compact-metric finite-modulus consumer handoff for BEDC.Derived.LebesgueNumberFiniteNet

Constructive story: A $\mathsf{LebesgueNumberFiniteNet}^{\uparrow}$ packet is generated from a CompactMetricUp source row, a finite cover-stability request, a positive tolerance row, a FiniteEpsilonNetUp handoff, a closed ProbeBundle of centers, center-cover assignment rows, coverage-distance rows, stability-tolerance rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6346

A Concrete Naming Certificate for $\mathsf{DunfordPettis}^{\uparrow}$

Following Dunford and Pettis on weak compactness in L^1 spaces, the $\mathsf{DunfordPettis}^{\uparrow}$ packet records the finite BEDC surface on which weak compactness is handed to uniform integrability before any selected weakly compact subsequence, countable-choice schedule, quotient almost-everywhere equality, or ambient Banach-space compactness theorem is public. It sits beside the measure and integral sources of chapter 92 and chapter 123, the tail-budget packet of chapter 6228, the functional-analysis carrier of chapter 94, and the bounded-operator surface of chapter 1644.

Definition 6346.1 (Dunford-Pettis carrier)**.** *A* $\mathsf{DunfordPettis}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (U, M, I, F, B, R, W, H, C, P, N)$$

with the following displayed rows:

1. *U is the* $\mathsf{UniformIntegrability}^{\uparrow}$ *source row naming the family-level tail budget.*
2. *M is the* $\mathsf{Measure}^{\uparrow}$ *source row through which measurable domains, measurable sets, and finite-measure windows are read.*
3. *I is the* $\mathsf{Integral}^{\uparrow}$ *readback row for displayed L^1 bounds and tail integrals.*
4. *F is the* $\mathsf{FunctionalAnalysis}^{\uparrow}$ *row naming the weak topology, dual-pairing probes, and finite neighborhood tests visible to the packet.*
5. *B is the* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *row for bounded linear probes used to inspect weak images.*
6. *R is the* $\mathsf{Real}^{\uparrow}$ *budget row storing scalar tolerances, norm bounds, and finite comparison windows.*
7. *W is the weak-compactness handoff row. It records only the finite implication route from uniform-integrability evidence through weak-probe tests to the consumer-visible compactness request.*
8. *H records componentwise* hsame *transport for U, M, I, F, B, R, W and the local structural rows.*
9. *C records displayed* Cont *replay from weak-compactness requests through $U, M, I, F, B, R, W, H, P, N$.*
10. *P is the* Pkg *provenance over* $\mathsf{DunfordPettis}^{\uparrow}$, $\mathsf{UniformIntegrability}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{DunfordPettis}^{\uparrow}}$ *row.*

The carrier has no coordinate for a selected weakly compact subsequence, countable choice, quotient almost-everywhere equality, ambient Banach-space compactness, a host dual-ball compactness theorem, or an unbounded family of weak probes.

Theorem 6346.2 (Dunford-Pettis NameCert obligations)**.** *For every accepted* $\mathsf{DunfordPettis}^{\uparrow}$ *carrier*

$$D = (U, M, I, F, B, R, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{DunfordPettis}^{\uparrow}}$ *surface consists exactly of carrier admission for the displayed rows, uniform-integrability source compatibility through* U*, measure source compatibility through* M*, integral readback through* I*, weak-test visibility through* F*, bounded-probe visibility through* B*, scalar-budget visibility through* R*, weak-compactness handoff through* W*, componentwise transport through* H*, displayed replay through* C*, package provenance through* P*, and local naming through* N*.*

Every weak-compactness-facing consumer must read the same rows before using the handoff. It receives no selected subsequence, no countable-choice schedule, no quotient almost-everywhere equality, no ambient Banach-space compactness theorem, no host dual-ball compactness theorem, and no infinite probe family.

Proof. Carrier admission is projection from Definition 6346.1. The accepted packet displays exactly $U, M, I, F, B, R, W, H, C, P, N$.

The analytic source rows U, M, I fix the uniform-integrability, measure, and integral surfaces. The weak-facing rows F and B then limit the consumer to the displayed weak topology tests and bounded linear probes, while R supplies only the scalar budgets visible in the packet.

The handoff row W is downstream of those rows. It records the finite implication route from uniform integrability to weak compactness, but it does not choose a subsequence, quotient representatives, or a host compactness theorem.

The structural rows H, C, P, N preserve that same packet under transport, replay, provenance, and naming. Since every excluded object has no carrier coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6346.3 (Dunford-Pettis uniform-integrability handoff)**.** *Every weak-compactness consumer of an accepted* $\mathsf{DunfordPettis}^{\uparrow}$ *carrier reaches the Dunford-Pettis handoff only through the displayed route*

$$U \longmapsto M, I \longmapsto F, B, R \longmapsto W \longmapsto H, C, P, N.$$

Thus the consumer may read the uniform-integrability budget, measure and integral evidence, finite weak-test and bounded-probe rows, scalar bounds, the weak-compactness handoff, transport, replay, provenance, and local naming, and it may not read selected weakly compact subsequences, countable choice, quotient almost-everywhere equality, ambient Banach-space compactness, host dual-ball compactness, or unbounded weak-probe families.

Proof. Begin with a weak-compactness-facing read. By Definition 6346.1, the only tail-control source is U, and it is interpreted only through the displayed measure and integral rows M and I.

The weak tests then pass through F and B. These rows expose finite dual-pairing and bounded-operator probes, while R records the scalar comparison budgets needed by those probes.

The consumer reaches weak compactness only at W. That row is a handoff from the already displayed uniform-integrability data to the public weak-compactness request, not an extraction of a subsequence or a compactness theorem from an ambient Banach space.

Finally H, C, P, N transport, replay, package, and name the same route. Because the carrier lacks coordinates for the refused objects, no consumer can recover them from the Dunford-Pettis packet. □

Falsifiable prediction: Any $\mathsf{DunfordPettis}^{\uparrow}$ read that claims weak compactness while bypassing the displayed $\mathsf{UniformIntegrability}^{\uparrow}$, $\mathsf{Measure}^{\uparrow}$, $\mathsf{Integral}^{\uparrow}$, $\mathsf{FunctionalAnalysis}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, hsame, Cont, Pkg, and local NameCert rows refutes the carrier of Definition 6346.1.

Independence witness: uniform-integrability, measure, integral, functional-analysis, bounded-linear-operator: $\mathsf{DunfordPettis}^{\uparrow}$ is not bijective to any listed sibling because it stores the finite weak-compactness handoff row over the shared tail-control and weak-probe sources without storing selected subsequences, countable choice, quotient almost-everywhere equality, or ambient compactness.

Closure status for $\mathsf{NameCert}_{\mathsf{DunfordPettis}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DunfordPettis}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6346.1, Definition 6346.2, and Definition 6346.3 seed the finite handoff from uniform integrability to weak compactness over measure, integral, weak-test, bounded-probe, scalar-budget, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DunfordPettis}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim selected weakly compact subsequences, countable choice, quotient almost-everywhere equality, ambient Banach-space compactness, host dual-ball compactness, unbounded weak-probe families, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, uniform-integrability source compatibility, measure and integral readback, weak-test visibility, bounded-probe visibility, scalar-budget visibility, handoff nonescape, transport, replay, provenance, and local naming for BEDC.Derived.DunfordPettisUp.

Constructive story: A $\mathsf{DunfordPettis}^{\uparrow}$ packet is generated from UniformIntegrability, Measure, Integral, FunctionalAnalysis, BoundedLinearOperator, Real scalar-budget, weak-compactness handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows.

Chapter 6347

A Concrete Naming Certificate for $\mathsf{LocatedOrderInterval}^{\uparrow}$

Following Bishop–Bridges constructive analysis, the $\mathsf{LocatedOrderInterval}^{\uparrow}$ packet names the finite order-facing interval data needed to compare located lower and upper boundary evidence before a real-completion consumer reads a terminal real seal. It sits between the interval carrier of chapter 1709 and the two-sided cut surface of chapter 2087: the packet records lower-bound and upper-bound rows, a located-gap row, dyadic tolerance data, finite stream windows, regular-rational readback, and a terminal $\mathsf{Real}^{\uparrow}$ seal without importing trichotomy, a Dedekind construction of $\mathsf{Real}^{\uparrow}$, quotient real equality, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6347.1 (Located order interval carrier)**.** *A* $\mathsf{LocatedOrderInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (L, U, G, D, S, R, E, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower-bound row. It records the finite lower observations and the order-facing comparison requests visible to a completion route.*
2. *U is the upper-bound row. It records the finite upper observations and the matching comparison requests.*
3. *G is the located-gap row. It binds the lower and upper observations to one displayed apartness or overlap request before any real-facing seal is read.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row. It stores the dyadic endpoints, slack, radius, and comparison budget used by L, U, G.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row. It schedules the visible lower and upper reads without selecting an infinite name.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row. It translates the scheduled windows into regular-rational comparison evidence.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal. It is available only after L, U, G, D, S, R have been displayed.*
8. *H records componentwise* hsame *transport for the lower-bound row, upper-bound row, located-gap row, dyadic tolerance row, stream window, regular-rational readback, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay for each order, interval, or real-completion request through* $L, U, G, D, S, R, E, H, P, N$.
10. *P is the* Pkg *provenance over* $\mathsf{LocatedOrderInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{LowerUpperRealCut}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedOrderInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $L, U, G, D, S, R, E, H, C, P, N$. It has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, open-cover compactness, a Dedekind construction of $\mathsf{Real}^{\uparrow}$*, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6347.2 (Located order interval NameCert obligations)**.** *For an accepted* $\mathsf{LocatedOrderInterval}^{\uparrow}$ *carrier*

$$O = (L, U, G, D, S, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{LocatedOrderInterval}^{\uparrow}}$ *surface consists exactly of the following finite obligation rows:*

1. *carrier exposure for the lower-bound row, upper-bound row, located-gap row,* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row,* $\mathsf{StreamName}^{\uparrow}$ *finite-window row,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, replay, provenance, and local naming row;*
2. *lower-bound stability, requiring every lower observation to be read through L, D, S, R, H, C, P, N before a completion-facing consumer may inspect it;*
3. *upper-bound stability, requiring every upper observation to be read through U, D, S, R, H, C, P, N by the same finite route;*
4. *located-gap exactness, requiring G to bind the displayed lower and upper rows to one located request before the regular-rational readback and real seal are consumed;*
5. *tolerance-window compatibility, requiring D and S to expose the same finite comparison budget that R reads back;*
6. *real-seal non-escape, requiring E to remain downstream of L, U, G, D, S, R and to export no trichotomy, quotient real equality, selected representative, countable-choice schedule, open-cover compactness witness, Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

No local NameCert obligation is stored outside these rows.

Proof. Carrier exposure is projection from Definition 6347.1. The accepted packet displays exactly the lower-bound row, upper-bound row, located-gap row, dyadic tolerance row, stream window, regular-rational readback, real seal, and structural rows.

Lower-bound and upper-bound stability are componentwise. The row D supplies the shared dyadic tolerance budget, S schedules the finite windows, and R reads the same windows back as regular-rational comparison evidence.

Located-gap exactness is read from G against L and U. The gap row binds the two order-facing ledgers to one displayed located request before any terminal real seal can be consumed.

The structural tail H, C, P, N transports, replays, sources, and locally names the same packet. Since the carrier has no coordinate for trichotomy, quotient real equality, selected representatives, countable choice, open-cover compactness, a Dedekind construction of $\mathsf{Real}^{\uparrow}$, or ambient completeness, the listed obligation rows are exhaustive. □

Theorem 6347.3 (Located order interval real-completion route)**.** *Every completion-facing read of an accepted* $\mathsf{LocatedOrderInterval}^{\uparrow}$ *carrier factors through the displayed route*

$$L, U, G \longmapsto D, S, R \longmapsto E, H, C, P, N.$$

The route first exposes lower-bound, upper-bound, and located-gap evidence, then consumes the $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, the* $\mathsf{StreamName}^{\uparrow}$ *finite window, and the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback, and only then reaches the* $\mathsf{Real}^{\uparrow}$ *seal. Thus it gives Real and Completion consumers a finite comparison packet between located interval data and lower/upper real cut boundaries without asserting trichotomy, quotient real equality, selected representatives, countable choice, open-cover compactness, a Dedekind construction of* $\mathsf{Real}^{\uparrow}$*, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with an accepted carrier $O = (L, U, G, D, S, R, E, H, C, P, N)$. The order-facing evidence is exactly the lower-bound row L, upper-bound row U, and located-gap row G.

Read the tolerance and name rows next. The dyadic row D records the finite comparison budget, the stream row S schedules the visible lower and upper windows, and the regular-rational row R reads those windows back for the real-completion route.

Only after those rows are present may the terminal seal E be consumed. The structural rows H, C, P, N preserve the same finite packet under transport, replay, provenance, and local naming. No further carrier coordinate exists, so the route cannot export any of the excluded order, quotient, choice, compactness, or completeness principles. □

Falsifiable prediction: Every accepted $\mathsf{LocatedOrderInterval}^{\uparrow}$ completion-facing read factors through lower-bound, upper-bound, located-gap, dyadic tolerance, stream-window, regular-rational readback, $\mathsf{Real}^{\uparrow}$ seal, transport, replay, provenance, and local naming rows.

Independence witness: locatedrealinterval, loweruppperrealcut, dyadicratcore, streamname, regseqrat, real: $\mathsf{LocatedOrderInterval}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries order-facing lower and upper boundary rows, a located-gap row, dyadic tolerance data, stream-window scheduling, regular-rational readback, a terminal real seal, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedOrderInterval}^{\uparrow}}$**.**

Constructive story: A $\mathsf{LocatedOrderInterval}^{\uparrow}$ packet is generated from lower-bound and upper-bound rows, a located-gap row, DyadicRatCore tolerance data, a StreamName finite window, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{LocatedOrderInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6347.1, Definition 6347.2, and Definition 6347.3 seed the finite order-facing comparison packet between located interval data and lower-upper real cut boundaries.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{LocatedOrderInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim trichotomy, a Dedekind construction of $\mathsf{Real}^{\uparrow}$, quotient real equality, selected representatives, countable choice, open-cover compactness, or $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier exposure, lower-bound stability, upper-bound stability, located-gap exactness, tolerance-window compatibility, and real-seal non-escape for BEDC.Derived.LocatedOrderIntervalUp.

Chapter 6348

A Concrete Naming Certificate for $\mathsf{NagataSmirnovMetrization}^{\uparrow}$

Following Nagata and Smirnov on metrization, the $\mathsf{NagataSmirnovMetrization}^{\uparrow}$ packet records the finite BEDC route by which a regular topological source with a displayed locally finite σ-cover basis exposes a metric handoff. It sits between the topology carrier of chapter 88, the metric consumer surface of chapter 57, the uniform-entourage handoff of chapter 4226, and the normal second-countable route of chapter 3337. The packet names only the displayed cover-family route to metric data; it does not assert $\mathsf{Real}^{\uparrow}$ completeness, choice-selected covers, host topology equality, or arbitrary point-set metrizability.

Definition 6348.1 (Nagata–Smirnov metrization carrier)**.** *A* $\mathsf{NagataSmirnovMetrization}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (T, R, L, S, B, E, D, M, F, U, H, C, P, N)$$

with the following displayed rows:

1. *T is the* $\mathsf{Topology}^{\uparrow}$ *source row, recording the open-set classifier and finite neighbourhood requests read by the packet.*
2. *R is the regularity row for closed-neighbourhood separation requests presented by the topology source.*
3. *L is the locally finite cover-family ledger. It lists the inspected cover layers and the finite star-neighbourhood checks used by the metric route.*
4. *S is the σ-layer schedule row. It records the finite layer window through which cover-family requests are replayed.*
5. *B is the base-refinement row, tying each inspected neighbourhood request in T to a displayed cover member in the locally finite schedule.*
6. *E is the entourage-shrink row exported to the* $\mathsf{FrinkMetrization}^{\uparrow}$ *route after the local cover checks have been displayed.*
7. *D is the dyadic radius ledger used by the downstream metric handoff.*
8. *M is the* $\mathsf{Met}^{\uparrow}$ *handoff row. It is available only after T, R, L, S, B, E, D have been exposed.*
9. *F records finite compatibility between the metric-ball rows from M and the source topology windows from T.*

10. *U records the comparison with the* $\mathsf{UrysohnMetrization}^{\uparrow}$ *route on packets that also expose normal second-countable data.*
11. *H records componentwise* hsame *transport for all displayed rows.*
12. *C records displayed* Cont *replay from a topology request through regularity, locally finite cover layers, entourage shrink, dyadic radius data, metric handoff, compatibility, provenance, and local naming.*
13. *P is the* Pkg *provenance over* $\mathsf{NagataSmirnovMetrization}^{\uparrow}$, $\mathsf{Topology}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{FrinkMetrization}^{\uparrow}$, $\mathsf{UrysohnMetrization}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
14. *N is the local* $\mathsf{NameCert}_{\mathsf{NagataSmirnovMetrization}^{\uparrow}}$ *row.*

The carrier has no coordinate for $\mathsf{Real}^{\uparrow}$ *completeness, choice-selected covers, host topology equality, arbitrary refinements, quotient point equality, or an ambient metric oracle.*

Lean4 (statement) `BEDC.Derived.NagataSmirnovMetrizationUp`

Theorem 6348.2 (Nagata–Smirnov metrization NameCert obligations)**.** *For every accepted* NagataSmirnovMetriza *carrier $G = (T, R, L, S, B, E, D, M, F, U, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{NagataSmirnovMetrization}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the topology source, regularity row, locally finite cover-family ledger, layer schedule, base-refinement row, entourage-shrink row, dyadic radius ledger, metric handoff, compatibility rows, transport, replay, provenance, and local naming row;*
2. *cover-family readability, requiring every cover request to start at T and pass through the locally finite ledger L and layer schedule S before any metric row is consumed;*
3. *regularity and base-refinement compatibility, requiring R and B to expose the finite closed-neighbourhood and cover-refinement checks used by the shrink row E;*
4. *metric handoff stability, requiring E, D, M, F to keep the* $\mathsf{Met}^{\uparrow}$ *ball rows downstream of the displayed cover-family route and comparable with the* $\mathsf{FrinkMetrization}^{\uparrow}$ *and* $\mathsf{UrysohnMetrization}^{\uparrow}$ *dependency anchors;*
5. *refusal, requiring P and N to name only this finite packet and to refuse* $\mathsf{Real}^{\uparrow}$ *completeness, choice-selected covers, host topology equality, arbitrary refinements, quotient point equality, and ambient metric oracles.*

No NameCert obligation is stored outside those rows.

Proof. Project the carrier of Definition 6348.1. The projection exposes exactly the topology source, regularity row, locally finite cover ledger, layer schedule, base-refinement row, shrink row, dyadic ledger, metric handoff, compatibility rows, and the structural rows H, C, P, N.

The cover-family route is forced by row order. A metric-facing request first reads neighbourhood data from T, then the locally finite cover ledger L, and then the finite layer schedule S before a base-refinement row B or shrink row E can be inspected.

The metric handoff is therefore downstream of R, L, S, B, E, D. The rows F and U only compare the resulting metric windows with the topology source and the neighbouring metrization packets; they do not supply a host metric or a new topology equality principle.

Since the tuple contains no $\mathsf{Real}^{\uparrow}$ completeness, choice-selected cover, host topology equality, arbitrary refinement, quotient point equality, or ambient metric coordinate, the local NameCert cannot export those principles. □

Theorem 6348.3 (Nagata–Smirnov metric handoff)**.** *Every* $\mathsf{Met}^{\uparrow}$ *consumer of an accepted* NagataSmirnovMetrizati *packet receives the finite route*

$$\mathsf{Topology}^{\uparrow},\ R,\ L,\ S,\ B,\ E,\ D,\ M,\ F,\ U,\ H,\ C,\ P,\ N.$$

The metric row M is consumed as a downstream handoff tied to the displayed locally finite cover family and its topology compatibility windows. It is not a host metric, not a selected cover metric, and not a claim that the source topology equals an external topology.

Proof. Read the carrier rows in order. The topology source, regularity row, locally finite cover ledger, layer schedule, base-refinement row, entourage shrink, and dyadic radius ledger are displayed before M is available.

The compatibility row F binds the handoff back to finite topology windows, and U records only the comparison with the Urysohn-style route when that sibling data is also present. Thus the metric consumer receives a named finite route from the original topology surface to the displayed metric row.

Finally use the refusal surface in P and N. Since no row stores $\mathsf{Real}^{\uparrow}$ completeness, selected covers, host topology equality, arbitrary refinements, quotient point equality, or an ambient metric oracle, the handoff remains the finite BEDC route named above. □

Closure status for $\mathsf{NameCert}_{\mathsf{NagataSmirnovMetrization}^{\uparrow}}$.

Constructive story: A $\mathsf{NagataSmirnovMetrization}^{\uparrow}$ packet is generated from a topology source row, regularity row, locally finite cover-family ledger, layer schedule, base-refinement row, entourage shrink, dyadic radius ledger, metric handoff, topology compatibility row, sibling metrization comparison row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{NagataSmirnovMetrization}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6348.1, Definition 6348.2, and Definition 6348.3 seed the finite regular locally finite cover-family route to a displayed metric handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{NagataSmirnovMetrization}^{\uparrow}}, \mathsf{formalTarget})$

Bridge status: none

Not claimed: This chapter does not close Real completeness, choice-selected covers, host topology equality, arbitrary refinements, quotient point equality, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, cover-family readability, regularity and base-refinement compatibility, entourage-shrink exactness, metric handoff stability, transport, replay, provenance, and local naming for BEDC.Derived.NagataSmirnovMetrizationUp.

Chapter 6349

A Concrete Naming Certificate for PicardCompletionResidualCauchyRealizer$^{\uparrow}$

Following Bishop–Bridges constructive analysis and Banach's fixed-point route, the PicardCompletionResidualCauchy packet records the finite residual-to-completion realizer that a Picard orbit exposes before endpoint consumers read a completed value. It refines the iteration ledger of chapter 2359 through the completion side of chapter 1691, the complete-metric source of chapter 128, the Cauchy-modulus budget of chapter 292, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names the residual Cauchy route consumed by Banach fixed-point and Picard–Lindelof surfaces; it does not select a fixed point, identify quotient reals, invoke countable choice, assume host function extensionality, or assert ambient Real$^{\uparrow}$ completeness.

Definition 6349.1 (Picard completion residual Cauchy realizer carrier)**.** *A* PicardCompletionResidualCauchyRealize *carrier is a finite packet*

$$P = (X, T, I, D, M, F, R, E, H, C, Q, N)$$

with the following displayed rows.

1. *X is the* CompleteMetric$^{\uparrow}$ *source row, including the metric carrier and distance-read boundary used by the residual comparisons.*
2. *T is the contraction and* PicardIteration$^{\uparrow}$ *ledger row, recording the self-map graph, initial point, finite orbit, and Picard replay data read before any completion handoff.*
3. *D is the residual-distance row, storing the adjacent distance comparisons and contraction-ratio bounds extracted from T inside the complete metric source X.*
4. *M is the* CauchyModulus$^{\uparrow}$ *row generated from the displayed residual-distance schedule.*
5. *F is the* FastCauchyCompletion$^{\uparrow}$ *handoff row, exposing the modulus-controlled completion route read from M and the finite Picard residual ledger.*
6. *R is the* RegSeqRat$^{\uparrow}$ *readback row available to consumers that inspect the completion realizer as a regular rational Cauchy name.*
7. *E is the* Real$^{\uparrow}$ *seal row, marking the terminal real-facing boundary after F and R have been displayed.*
8. *H records componentwise* hsame *transport for the complete-metric source, contraction/Picard ledger, residual-distance row, CauchyModulus row, FastCauchyCompletion handoff, RegSeqRat readback, Real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from the complete metric and contraction rows through the residual schedule, modulus extraction, completion handoff, regular readback, Real seal, transport, provenance, and local naming.*

10. *Q is the* Pkg *provenance over* $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{FastCauchyCompletion}^{\uparrow}$ $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $X, T, D, M, F, R, E, H, C, Q, N$. It has no coordinate for selected fixed points, quotient equality, countable choice, host function extensionality, or an ambient completeness theorem for $\mathsf{Real}^{\uparrow}$.

Theorem 6349.2 (Picard completion residual Cauchy realizer NameCert obligations)**.** *For every accepted* $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$ *carrier $P = (X, T, I, D, M, F, R, E, H, C, Q, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}}$ *surface consists exactly of the following obligation rows:*

1. *carrier admission for the displayed complete-metric source, contraction/Picard ledger, finite iterate row, residual-distance row, CauchyModulus row, FastCauchyCompletion handoff, RegSeqRat readback, Real seal, and structural rows H, C, Q, N;*

2. *residual discipline, requiring every completion-facing read to inspect the Picard ledger and residual-distance row before any Cauchy modulus or completion row is exposed;*

3. *modulus extraction, requiring M to be generated from the displayed residual schedule rather than from an endpoint selector or ambient completeness theorem;*

4. *completion handoff, requiring F to read the same modulus route before the regular readback row R or terminal Real seal E may be consumed;*

5. *transport and replay stability, requiring H and C to preserve the ordered route under componentwise sameness and displayed continuation replay;*

6. *ledger non-escape, requiring Q and N to name only the displayed rows and to export no selected fixed point, quotient equality, countable choice, host function extensionality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness coordinate.*

No NameCert obligation is stored outside these rows.

Proof. Carrier admission is projection from Definition 6349.1. The visible packet consists of the complete-metric source X, contraction/Picard ledger T, finite iterate row I, residual-distance row D, modulus row M, completion handoff F, regular readback R, Real seal E, and structural rows H, C, Q, N.

Residual discipline follows from the placement of X, T, I, D at the front of the route. A completion-facing consumer first reads the metric and contraction data, then the finite Picard ledger and the residual comparisons extracted from it.

Modulus extraction and completion handoff are downstream. The row M is generated only after the residual schedule is displayed, and F is the only completion row exposed after M. Thus R and E remain attached to the residual-to-modulus route.

Finally, H, C, Q, N transport, replay, source, and name exactly the same rows. Since selected fixed points, quotient equality, countable choice, host function extensionality, and ambient Real completeness are absent from the carrier, no local NameCert obligation can export them. □

Theorem 6349.3 (Picard completion residual Cauchy realizer handoff)**.** *Every completion-facing read of an accepted* $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$ *carrier factors through*

$$\mathsf{CompleteMetric}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{CauchyModulus}^{\uparrow} \longmapsto \mathsf{FastCauchyCompletion}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

Consequently Banach fixed-point and Picard–Lindelof consumers receive a complete-metric source, contraction/Picard ledger, residual-distance row, CauchyModulus schedule, FastCauchyCompletion handoff, RegSeqRat readback, terminal Real seal, transport, replay, provenance, and local naming before they can read the completion endpoint boundary.

Proof. Start with a completion-facing read of P. By Definition 6349.1, the only completion handoff is F, and it is downstream of the complete-metric row X, contraction/Picard ledger T, finite iterate row I, residual row D, and modulus row M.

The first part of the route is fixed by the Picard data. The complete metric supplies the distance-read boundary, the contraction/Picard ledger supplies the finite orbit, and the residual row supplies the adjacent-distance schedule used by the modulus row.

The second part is fixed by the completion and readback rows. The FastCauchyCompletion handoff F is read only after M, the regular rational row R is read only after F, and the Real seal E is terminal for the displayed route.

The structural rows H, C, Q, N preserve and name the same order. Because the excluded objects have no coordinate in the carrier and no replay edge in the displayed route, every completion-facing read factors through the route shown in the statement. □

Falsifiable prediction: Every accepted $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$ completion-facing read factors through a CompleteMetric source, contraction/Picard ledger, residual-distance row, CauchyModulus schedule, FastCauchyCompletion handoff, RegSeqRat readback, RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert row before any fixed-point or ODE endpoint boundary is consumed.

Independence witness: picard-iteration, fast-cauchy-completion, completemetric, cauchymodulus, regseqrat, real: $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries the complete-metric source, contraction/Picard ledger, residual-distance row, modulus extraction, completion handoff, regular readback, real-seal boundary, transport, replay, provenance, and local naming in one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6349.1, Definition 6349.2, and Definition 6349.3 seed the finite residual Cauchy realizer route from Picard residuals to completion-facing Real reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close selected fixed points, quotient equality, countable choice, host function extensionality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, residual discipline, modulus extraction, completion handoff, transport and replay stability, ledger non-escape, and TasteGate field faithfulness for BEDC.Derived.PicardCompletionResidualCau

Constructive story: A $\mathsf{PicardCompletionResidualCauchyRealizer}^{\uparrow}$ packet is generated from a CompleteMetric source, contraction/Picard ledger, residual-distance row, CauchyModulus schedule, FastCauchyCompletion handoff, RegSeqRat readback, RealUp seal boundary, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6350

A Concrete Naming Certificate for $\mathsf{CoreCompact}^{\uparrow}$

Following Isbell, Johnstone, and constructive locale practice, the $\mathsf{CoreCompact}^{\uparrow}$ packet records a core-compact neighbourhood relation as finite BEDC data. It sits between the topology carrier of chapter 88, the local compact-neighbourhood certificate of chapter 1726, the compact metric witness surface of chapter 129, and the located compact handoff of chapter 1488. The packet names one displayed way-below neighbourhood route and its compact witness support; it does not assert open-cover compactness, a host basis selector, countable choice, quotient opens, or ambient Real completeness.

Definition 6350.1 (CoreCompact carrier)**.** *A* $\mathsf{CoreCompact}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (X, U, V, W, K, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the* $\mathsf{Topology}^{\uparrow}$ *source row whose opens and incidence data are already admitted as finite BEDC carrier rows.*
2. *U is the inspected open-neighbourhood row requested by a consumer.*
3. *V is the displayed core row, read as a neighbourhood that is way below U in the finite inspection order carried by the packet.*
4. *W is the way-below witness row, recording the finite containment and directed-cover test by which the packet justifies $V \ll U$ for the displayed request.*
5. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *or compact-window witness attached to the displayed core neighbourhood when a metric compact witness is available.*
6. *L is the optional* $\mathsf{LocatedCompact}^{\uparrow}$ *handoff row for the same core neighbourhood, or an explicit empty handoff row when no located boundary is claimed.*
7. *H records componentwise* hsame *transport for topology source, inspected open, core row, way-below witness, compact witness, located handoff, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a finite neighbourhood request through X, U, V, W, K, L, H and into local compactness, compact metric, located compact, or completion-facing consumers.*
9. *P is the* Pkg *provenance over* $\mathsf{CoreCompact}^{\uparrow}$*,* $\mathsf{Topology}^{\uparrow}$*,* $\mathsf{LocallyCompact}^{\uparrow}$*,* $\mathsf{CompactMetric}^{\uparrow}$*,* $\mathsf{LocatedCompact}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CoreCompact}^\uparrow}$ *row.*

The classifier compares accepted carriers only through these ten rows. It has no coordinate for arbitrary open covers, a host-selected basis, quotient-open equality, countable choice, all-neighbourhood local compactness, or ambient completeness.

Lean4 ✓ `BEDC.Derived.CoreCompactUp.CoreCompactTasteGate single carrier alignment`

Theorem 6350.2 (CoreCompact way-below neighbourhood route)**.** *Let $Q = (X, U, V, W, K, L, H, C, P, N)$ be an accepted* $\mathsf{CoreCompact}^\uparrow$ *packet. Every* $\mathsf{LocallyCompact}^\uparrow$ *or* $\mathsf{Topology}^\uparrow$ *consumer that reads a core-compact neighbourhood must first display the topology source X, the inspected open row U, the core row V, and the way-below witness W. Only after those rows are present may the consumer inspect the compact witness K, the optional located compact handoff L, and the structural rows H, C, P, N. The route supplies a finite way-below neighbourhood certificate and does not export arbitrary open-cover compactness, a host basis selector, quotient-open equality, countable choice, all-centres/all-radii properness, or ambient* $\mathsf{Real}^\uparrow$ *completeness.*

Proof. Project the carrier of Definition 6350.1. The first public rows are the topology source, the inspected open, the displayed core neighbourhood, and the way-below witness.

Read the local compactness sibling through chapter 1726. That sibling supplies compact neighbourhood data only after an inspected point and displayed compact window are named; here the analogous route is U, V, W before compact support is read.

The compact support is therefore downstream. The row K records the compact metric or compact-window witness for the displayed core, and the optional row L can refine the same neighbourhood only when located compact data are shown.

Finally use the structural rows. Componentwise transport and replay preserve the same finite route, while provenance and local naming identify the sibling surfaces. Since no excluded open-cover, basis-selector, quotient, choice, properness-family, or ambient-completeness coordinate occurs in Q, no consumer can recover such data from the way-below route. □

Theorem 6350.3 (CoreCompact located compact handoff)**.** *Let $Q = (X, U, V, W, K, L, H, C, P, N)$ be an accepted* $\mathsf{CoreCompact}^\uparrow$ *packet. A* $\mathsf{LocatedCompact}^\uparrow$ *or* $\mathsf{CompactMetric}^\uparrow$ *consumer may read L or K only as support for the same displayed core neighbourhood $V \ll U$. The handoff factors through the* $\mathsf{Topology}^\uparrow$ *source row, the* $\mathsf{LocallyCompact}^\uparrow$ *compact-neighbourhood route, the* $\mathsf{CompactMetric}^\uparrow$ *witness row, the optional* $\mathsf{LocatedCompact}^\uparrow$ *row, and then the* BHist, hsame, Cont, Pkg, *and* NameCert *rows H, C, P, N. It exports no independent located compact object, no open-cover compactness theorem, no countable choice, and no ambient completeness principle.*

Proof. Begin with Definition 6350.2. It forces every consumer to expose X, U, V, W before compact or located support is available.

Use the compact metric sibling of chapter 129. The compact row K is read only for the displayed core neighbourhood; it is not a host compactness theorem and not a compact object detached from $V \ll U$.

Use the located compact sibling of chapter 1488. The row L is optional and exact: when present it refines the same compact neighbourhood, and when absent the packet makes no locatedness claim.

The remaining rows bind the handoff. Transport, replay, provenance, and local naming pass through H, C, P, N, so the located or compact consumer receives one displayed core-neighbourhood route and none of the excluded open-cover, choice, quotient, independent located-object, or ambient-completeness data. □

Closure status for $\mathsf{NameCert}_{\mathsf{CoreCompact}^{\uparrow}}$.

Constructive story: A $\mathsf{CoreCompact}^{\uparrow}$ packet is generated from a topology source row, an inspected open row, a displayed core neighbourhood row, a way-below witness row, a compact metric or compact-window witness, an optional located compact handoff, componentwise hsame transport, displayed Cont replay, package provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CoreCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6350.1, Definition 6350.2, and Definition 6350.3 seed the constructive core-compact interface between topology, local compactness, compact metric witnesses, and located compact handoffs.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CoreCompact}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim arbitrary open-cover compactness, host basis selection, quotient-open equality, countable choice, all-neighbourhood local compactness, global properness, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, way-below neighbourhood routing, compact witness projection, located compact handoff exactness, componentwise transport, replay stability, provenance, and local NameCert exactness for BEDC.Derived.CoreCompactUp.

Chapter 6351

A Concrete Naming Certificate for BoundedRiemannConvergence$^{\uparrow}$

Following Bishop and Bridges, the BoundedRiemannConvergence$^{\uparrow}$ packet records the finite BEDC surface by which a uniformly bounded row of regulated functions, a displayed pointwise limit row, and a Riemann/Darboux comparison ledger are handed to a Riemann-integral consumer. It sits after the regulated-function carrier of chapter 1906, the Riemann-integral boundary of chapter 1526, the Darboux-sum comparison route of chapter 1880, the bounded-variation real-function control surface of chapter 3654, and the real-equality naming boundary of chapter 1636. The packet names only the bounded finite handoff rows needed for constructive Riemann convergence; it does not import measure integration, host function quotients, selected limits, choice, or ambient Real$^{\uparrow}$ completeness.

Definition 6351.1 (Bounded Riemann convergence carrier)**.** *A* BoundedRiemannConvergence$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$R = (D, F, A, T, S, B, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the common compact interval and Darboux partition row, read through the accepted* DarbouxSum$^{\uparrow}$ *comparison surface.*
2. *F is the regulated-function family row, recording the displayed approximating functions over the same interval.*
3. *A is the pointwise approach row, listing the finite observation windows by which the family is compared with its named limit row.*
4. *T is the tagged Riemann readback row for the displayed members of the family.*
5. *S is the lower/upper Darboux bracket ledger associated to the same partition and tolerance rows.*
6. *B is the uniform bound row, exposing a common finite bound before any convergence consumer reads the integral handoff.*
7. *E is the terminal real-equality row used only to compare the named integral readbacks after D, F, A, T, S, B have been displayed.*
8. *H records componentwise* hsame *transport for the interval, family, approach, Riemann readback, Darboux bracket, bound, real-equality, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from regulated-function reads through pointwise approach windows, tagged sums, Darboux brackets, the uniform bound, and terminal real-equality comparison.*
10. *P is the* Pkg *provenance over* $\mathsf{BoundedRiemannConvergence}^{\uparrow}$, $\mathsf{RegulatedFunction}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{BoundedVariationRealFunction}^{\uparrow}$, $\mathsf{RealEquality}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{BoundedRiemannConvergence}^{\uparrow}}$ *row.*

Accepted packets are compared only by these displayed rows.

Theorem 6351.2 (Bounded Riemann convergence NameCert obligations)**.** *For every accepted* $\mathsf{BoundedRiemannConvergence}^{\uparrow}$ *carrier* $R = (D, F, A, T, S, B, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BoundedRiemannConverge}}$ *surface consists exactly of common interval and Darboux partition admission through D, regulated-family exposure through F, pointwise approach windows through A, tagged Riemann readback through T, lower/upper Darboux bracket visibility through S, uniform boundedness through B, terminal real-equality comparison through E, componentwise transport through H, displayed replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6351.1. The interval and partition data are opened by D, the regulated family by F, and the finite approach windows by A.

The Riemann/Darboux rows then supply the analysis handoff. Tagged Riemann reads pass through T, lower and upper Darboux brackets pass through S, and the common boundedness coordinate is visible as B before any terminal comparison is read.

Finally E, H, C, P, N compare, transport, replay, source, and name the same packet. Since the carrier contains no measure-integration row, no quotient-function equality row, no selected limit row, no choice row, and no ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6351.3 (Bounded Riemann convergence Darboux handoff)**.** *Every Riemann-integral consumer of an accepted* $\mathsf{BoundedRiemannConvergence}^{\uparrow}$ *carrier must first read the route*

$$\mathsf{RegulatedFunction}^{\uparrow},\ A,\ \mathsf{RiemannIntegral}^{\uparrow},\ \mathsf{DarbouxSum}^{\uparrow},\ B,\ \mathsf{RealEquality}^{\uparrow} \longmapsto H, C, P, N.$$

Thus the bounded convergence handoff is exposed as finite regulated-function observations, tagged Riemann readbacks, Darboux brackets, a common bound, and terminal real-equality comparison, not as measure integration, host function quotients, selected limits, choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with Definition 6351.2. It makes the regulated-family row, pointwise approach windows, Riemann readback, Darboux bracket, uniform bound, and terminal real-equality comparison the entire consumer-facing route.

The Darboux handoff is finite. A consumer may inspect the tagged Riemann row T only together with the lower/upper bracket row S over the same displayed interval and partition row D, and it may use the common bound only through B.

The structural rows H, C, P, N preserve that route for later integral readers. They do not add measure-theoretic integration, quotient equality of host functions, a selected limiting function, a choice principle, or ambient completeness of $\mathsf{Real}^{\uparrow}$. □

Falsifiable prediction: A purported $\mathsf{BoundedRiemannConvergence}^{\uparrow}$ consumer that reads a bounded convergence conclusion while omitting the regulated-family row, pointwise approach windows, tagged Riemann readback, Darboux bracket, common bound, real-equality comparison, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: regulated-function, riemannintegral, darboux-sum, bounded-variation-real-function, real-equality: $\mathsf{BoundedRiemannConvergence}^{\uparrow}$ is not bijective with any listed sibling because it joins the regulated-family source, pointwise approach windows, tagged Riemann readback, Darboux brackets, common bound, real-equality comparison, transport, replay, provenance, and local naming as one bounded convergence packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BoundedRiemannConvergence}^{\uparrow}}$.

Constructive story: A $\mathsf{BoundedRiemannConvergence}^{\uparrow}$ packet is generated from a common interval and Darboux partition row, a regulated-function family row, finite pointwise approach windows, tagged Riemann readback, lower/upper Darboux brackets, a uniform bound row, terminal real-equality comparison, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BoundedRiemannConvergence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6351.1, Definition 6351.2, and Definition 6351.3 seed the finite bounded Riemann convergence route for regulated functions, Darboux brackets, Riemann-integral consumers, real-equality comparison, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BoundedRiemannConvergence}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure integration, host function quotients, selected limits, choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, regulated-family exposure, pointwise approach windows, tagged Riemann readback, Darboux bracket exactness, uniform-bound visibility, real-equality comparison, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.BoundedRiemannConvergenceUp.

Chapter 6352

A Concrete Naming Certificate for $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ packet records the finite NameCert form of the constructive Dini theorem: a monotone continuous family on a compact metric source reaches a uniform limit through displayed finite modulus, window, rational readback, and Real-seal rows. It consumes the compact-uniform sibling chapter 4213, the modulus sibling chapter 3454, the uniform-limit sibling chapter 5795, the compact metric source of chapter 129, the continuous-map rows of chapter 130, the uniform-limit handoff of chapter 463, the finite $\mathsf{StreamName}^{\uparrow}$ windows of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback of chapter 301, and the $\mathsf{Real}^{\uparrow}$ seal of chapter 38. The packet is a finite theorem carrier; it does not assert open-cover compactness, ambient $\mathsf{Real}^{\uparrow}$ completeness, quotient equality of functions, countable choice, or hidden limiting values.

Definition 6352.1 (Constructive Dini theorem carrier)**.** *A* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (K, F, M, D, U, W, R, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *source row, exposing only the finite compact-metric source and probe rows used by the theorem route.*
2. *F is the* $\mathsf{ContinuousMap}^{\uparrow}$ *family row, naming the displayed continuous maps and the target* $\mathsf{Real}^{\uparrow}$ *value rows inspected by the packet.*
3. *M is the monotone ledger, recording the pointwise order route and the visible approach of F toward its limiting row.*
4. *D is the* $\mathsf{ConstructiveDiniModulus}^{\uparrow}$ *row, carrying the finite Dini modulus extracted from compact finite-net data, dyadic tolerances, pointwise windows, and finite maximum rows.*
5. *U is the combined* $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$ *and* $\mathsf{DiniUniformLimitModulus}^{\uparrow}$ *handoff row, exposing the compact-uniform route and the resulting* $\mathsf{UniformLimit}^{\uparrow}$ *row.*
6. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row, scheduling function values, limit values, finite net centers, and consumer precision requests.*
7. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached from W before the terminal real-name seal is exposed.*

8. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row for the uniform-limit read. It names the displayed real values without adding ambient real completeness.*
9. *H records componentwise* hsame *transport for the compact source, continuous family, monotone ledger, Dini modulus, uniform-limit handoff, stream windows, regular rational readback, Real seal, replay, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from a theorem request through $K, F, M, D, U, W, R, E, H, P, N$.*
11. *P is the* Pkg *provenance over* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$, $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$, $\mathsf{ConstructiveDiniModulu}$ $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
12. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveDiniTheorem}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, M, D, U, W, R, E, H, C, P, N$. It has no coordinate for general open-cover compactness, arbitrary subcover selection, quotient function equality, countable choice, hidden limit selectors, host suprema, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveDiniTheoremUp.TasteGate.ConstructiveDiniTheoremTasteGate`
`single carrier alignment`

Theorem 6352.2 (Constructive Dini theorem NameCert obligations)**.** *For every accepted* ConstructiveDiniTheorem *carrier*

$$T = (K, F, M, D, U, W, R, E, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{ConstructiveDiniTheorem}^{\uparrow}}$ *surface consists exactly of carrier admission for the compact metric source, continuous-map family, monotone ledger, Dini modulus row, uniform-limit row,* $\mathsf{StreamName}^{\uparrow}$ *windows,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{Real}^{\uparrow}$ *seal, componentwise transport, displayed replay, provenance, and local naming rows.*

The obligation surface requires the compact source and continuous family through K, F, monotone visibility through M, finite Dini modulus extraction through D, compact-uniform and uniform-limit handoff through U, L10 finite real-readback through W, R, E, transport and replay stability through H, C, provenance through P, and local naming through N.

Proof. Project the carrier of Definition 6352.1. The analytic rows are K, F, M, D, U, W, R, E, while H, C, P, N are the structural certificate rows.

The compact metric source and continuous family are admitted first. The monotone ledger then fixes which pointwise comparisons may be inspected, so the Dini modulus row D is downstream of displayed compact finite-net data and not of an external compactness principle.

The handoff row U consumes D together with the compact-uniform sibling route and exposes the $\mathsf{UniformLimit}^{\uparrow}$ read. The L10 boundary is then read only through the $\mathsf{StreamName}^{\uparrow}$ windows W, the $\mathsf{RegSeqRat}^{\uparrow}$ readback R, and the terminal $\mathsf{Real}^{\uparrow}$ seal E.

Finally H, C, P, N preserve, replay, source, and name the same route. Since no excluded object is a carrier coordinate, none is a NameCert obligation of the theorem packet. □

Theorem 6352.3 (Constructive Dini monotone uniform route)**.** *Every uniform-limit consumer of an accepted* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ *carrier must factor through the displayed route*

$$K, F, M \longmapsto D \longmapsto U \longmapsto W, R, E \longmapsto H, C, P, N.$$

Thus the theorem export is the finite BEDC route from compact metric source and monotone continuous family data through $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$, $\mathsf{ConstructiveDiniModulus}^{\uparrow}$, $\mathsf{DiniUniformLimitModulus}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, *and* $\mathsf{Real}^{\uparrow}$. *It exports no open-cover compactness, quotient function equality, countable choice, hidden limiting values, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Lean4 ✓ `BEDC.Derived.ConstructiveDiniTheoremUp.ConstructiveDiniTheoremMonotoneUniformRoute`

Proof. Begin with the NameCert obligations of Definition 6352.2. They place the compact source K, continuous family F, and monotone ledger M before the theorem can expose any uniform conclusion.

The row D is the finite Dini modulus row. It reads the compact finite-net and pointwise monotone data already displayed by the source and family rows, and it supplies the modulus object consumed by the compact-uniform handoff.

The row U composes the compact-uniform sibling and the uniform-limit-modulus sibling. Hence a uniform-limit consumer receives a single handoff produced by finite modulus extraction, not a selected open cover or a host-level supremum.

The terminal real-readback rows W, R, E keep the exported conclusion inside finite stream windows, regular rational readback, and a displayed Real seal. The structural rows H, C, P, N then transport, replay, source, and name the same route for downstream consumers. □

Theorem 6352.4 (Constructive Dini L10 seal nonescape)**.** *For an accepted* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ *carrier, the terminal* $\mathsf{Real}^{\uparrow}$ *values used by the uniform-limit conclusion are reachable only after the compact source row K, continuous-map family row F, monotone ledger M, Dini modulus row D, uniform-limit handoff row U,* $\mathsf{StreamName}^{\uparrow}$ *window row W, and* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row R have been displayed. The* $\mathsf{Real}^{\uparrow}$ *seal row E therefore exports the finite L10 readback route and no ambient completeness, quotient stream equality, hidden host limit, countable choice, or real trichotomy coordinate.*

Proof. Use Definition 6352.3. It places every terminal read downstream of the compact metric source, continuous family, monotone ledger, finite Dini modulus, and uniform-limit handoff rows.

The L10 readback portion is visible in W, R, E. The stream row W schedules the finite observations, the regular rational row R reads those observations at the requested precision, and E seals the displayed real values only after that readback has occurred.

Transport and replay do not create another analytic route. The rows H, C, P, N preserve the same carrier coordinates, so the terminal seal cannot export any object absent from T.

The excluded principles have no row in the carrier. Thus the theorem packet gives a finite Real-seal boundary for the constructive Dini route without importing ambient completion, quotient equality, hidden host limits, countable choice, or trichotomy. □

Theorem 6352.5 (Constructive Dini finite-net monotone-window route)**.** *Every uniform-limit consumer of an accepted* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ *carrier*

$$T = (K, F, M, D, U, W, R, E, H, C, P, N)$$

must expose the compact finite-net source and monotone window before the Dini modulus and $\mathsf{UniformLimit}^{\uparrow}$ *row are public. Equivalently, the consumer read factors through*

$$K, F, M \longmapsto D \longmapsto U \longmapsto W, R, E \longmapsto H, C, P, N,$$

with K providing the finite compact-metric net source, F the displayed continuous family, M the monotone comparison window, D the Dini modulus row, and U the uniform-limit handoff.

Lean4 ✓ `BEDC.Derived.ConstructiveDiniTheoremUp.ConstructiveDiniTheoremFiniteNetMonotoneWindowRoute`

Proof. Project the carrier rows $K, F, M, D, U, W, R, E, H, C, P, N$ from Definition 6352.1. The compact source, continuous family, and monotone ledger are the only rows available before the Dini modulus is inspected.

Use Definition 6352.3. It already forces every uniform-limit consumer through K, F, M, then D, then U, and finally the L10 readback and structural rows.

The finite-net source is part of the compact metric row K, while the monotone window is the public comparison data in M. Thus the modulus row D cannot be read as an ambient supremum or open-cover compactness principle; it is downstream of the displayed finite compact-metric and monotone data.

Apply Definition 6352.4. The $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows W, R, E keep the resulting uniform-limit read inside finite L10 windows, and H, C, P, N transport, replay, source, and name only that route. □

Theorem 6352.6 (Constructive Dini theorem obligation-closure package)**.** *The carrier, NameCert obligation, monotone-uniform route, L10 seal nonescape, and finite-net monotone-window route assemble the full obligation surface for* $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ *over* $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{ConstructiveDiniModulus}^{\uparrow}$, $\mathsf{UniformLimit}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert. *The assembled package excludes open-cover compactness, countable choice, hidden limiting values, quotient function equality, host suprema, real trichotomy, and ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the carrier of Definition 6352.1. Its rows name the compact metric source, continuous family, monotone ledger, Dini modulus, uniform-limit handoff, finite stream windows, regular-rational readback, Real seal, transport, replay, provenance, and local naming surface.

Apply Definition 6352.2. The listed carrier admission, theorem-route, modulus extraction, uniform-limit handoff, finite readback, transport, replay, provenance, local naming, and ledger exhaustion rows give the NameCert obligation frame for the packet.

Use Definition 6352.3, Definition 6352.4, and Definition 6352.5. These route the public consumer through the compact source and monotone window, then through the Dini modulus and uniform-limit handoff, and finally through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and the $\mathsf{Real}^{\uparrow}$ seal.

The carrier has no remaining analytic coordinate. Therefore the displayed rows close the paper-side obligation surface while the excluded principles remain outside the packet. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ read must expose a compact metric source, continuous-map family, monotone ledger, Dini modulus row, uniform-limit row, $\mathsf{StreamName}^{\uparrow}$ windows, $\mathsf{RegSeqRat}^{\uparrow}$ readback, $\mathsf{Real}^{\uparrow}$ seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming.

Independence witness: dini-monotone-compact-uniform, constructive-dini-modulus, dini-uniform-limit-modulus, compactmetric, continuousmap, uniformlimit, streamname, regseqrat, real: ConstructiveDiniTheorem is not bijective to any listed sibling because it binds the monotone compact-uniform theorem route to the L10 finite real-readback boundary in one local NameCert carrier.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveDiniTheorem}^{\uparrow}}$.

Constructive story: A $\mathsf{ConstructiveDiniTheorem}^{\uparrow}$ packet is generated from compact metric source rows, continuous-map family rows, monotone ledger rows, ConstructiveDiniModulus rows, DiniMonotoneCompactUniform and DiniUniformLimitModulus handoff rows, UniformLimit rows, StreamName finite windows, RegSeqRat readbacks, Real seal rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveDiniTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6352.1, Definition 6352.2, Definition 6352.3, and Definition 6352.4 seed the finite NameCert form of the constructive Dini theorem over compact metric, continuous-map, uniform-limit, StreamName, RegSeqRat, and Real rows.

Formal verification: FormalStatus(NameCert$_{\text{ConstructiveDiniTheorem}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, open-cover compactness, ambient Real$^{\uparrow}$ completeness, quotient equality of functions, countable choice, hidden limiting values, host suprema, or real trichotomy.

Upgrade path: Obligation closure requires checked carrier admission, monotone compact-uniform route exactness, Dini modulus extraction, uniform-limit handoff, StreamName and RegSeqRat readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.ConstructiveDiniTheoremUp.

Chapter 6353

A Concrete Naming Certificate for $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$

Following Banach and Rudin on closed operators between Banach spaces, the $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ packet records the graph norm of a displayed operator as finite NameCert data. It sits over the Banach carrier of chapter 90, the public bounded-linear-operator surface of chapter 1644, the closed-graph route of chapter 1537, the metric rows of chapter 57, the complete-metric witness rows of chapter 128, and the real-name seal of chapter 38. The packet names the displayed domain graph and its norm-sum route without replacing the graph by a quotient carrier or a host function-space object.

Definition 6353.1 (Banach operator graph-norm carrier)**.** *A* $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (X, Y, T, \Gamma, A, M, Q, L, H, C, P, N)$$

with the following displayed rows:

1. *X is the source* $\mathsf{Banach}^{\uparrow}$ *row, including the source norm, metric, complete-metric, and real-seal reads.*
2. *Y is the target* $\mathsf{Banach}^{\uparrow}$ *row, including the target norm, metric, complete-metric, and real-seal reads.*
3. *T is the operator action row, read either through a displayed* $\mathsf{BoundedLinearOperator}^{\uparrow}$ *packet or through a graph-closed operator row whose closedness is audited by* $\mathsf{ClosedGraphTheorem}^{\uparrow}$*.*
4. *Γ is the graph row, recording source endpoints, target endpoints, and graph-membership witnesses for the displayed action.*
5. *A is the graph-admission row, stating that only source endpoints whose graph membership is displayed in Γ are admitted to the graph-norm domain.*
6. *M is the metric readback row, routing source and target distance requests through* $\mathsf{Met}^{\uparrow}$ *and* $\mathsf{CompleteMetric}^{\uparrow}$ *data.*
7. *Q is the norm-sum row, recording the exact graph-norm expression $\|x\|_X + \|Tx\|_Y$ as a displayed* $\mathsf{Real}^{\uparrow}$ *read.*
8. *L is the completion handoff row, recording the Cauchy-to-limit route for graph-norm Cauchy data through X, Y, Γ, M, and Q.*

9. *H records componentwise* hsame *transport for source, target, operator, graph, admission, metric, norm-sum, completion handoff, provenance, and local naming rows.*
10. *C records displayed* Cont *replay from graph admission through metric readback, norm-sum exactness, graph-closedness stability, and completion handoff.*
11. *P is the* Pkg *provenance over* $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$, $\mathsf{Banach}^{\uparrow}$, $\mathsf{BoundedLinearOperator}^{\uparrow}$, $\mathsf{ClosedGraphTheorem}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}$ *row.*

The carrier has no coordinate for quotienting the graph, selecting host representatives, importing an ambient operator space, or replacing the displayed norm-sum by an external equality theorem.

Lean4 ✓ `BEDC.Derived.BanachOperatorGraphNormUp.TasteGate.BanachOperatorGraphNormTasteGate single carrier alignment`

Theorem 6353.2 (Banach operator graph-norm obligations)**.** *For every accepted* $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ *carrier* $G = (X, Y, T, \Gamma, A, M, Q, L, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}$ *surface consists exactly of graph-norm carrier admission, graph-closedness stability, norm-sum exactness, completion handoff, and refusal of quotient graph export.*

Lean4 ✓ `BEDC.Derived.BanachOperatorGraphNormUp.BanachOperatorGraphNormCarrier namecert obligations`

Proof. Carrier admission is projection from Definition 6353.1. The accepted graph-norm domain is read through A and Γ, so a source endpoint is admitted only when its graph-membership row is displayed.

Graph-closedness stability is read through T, Γ, L, H, C. The operator action is either already bounded-linear through chapter 1644 or is routed through the closed-graph certificate of chapter 1537; in both cases transport by H and replay by C preserve the same graph rows.

Norm-sum exactness is the role of Q. The source and target norm reads are measured through the metric and complete-metric rows M and sealed as $\mathsf{Real}^{\uparrow}$ data, so the graph norm is exactly the displayed sum $\|x\|_X + \|Tx\|_Y$.

Completion handoff is the role of L. A graph-norm Cauchy read supplies compatible Cauchy data in X and Y, replays graph membership through Γ, and returns a displayed graph limit through the complete-metric rows.

Finally, the carrier inventory rules out quotient graph export. No row of G stores a graph quotient, selected representative, ambient operator-space point, or host equality theorem, so every consumer remains inside the displayed NameCert surface. □

Theorem 6353.3 (Banach operator graph-norm completion handoff)**.** *Every completion-facing consumer of an accepted* $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ *packet factors through*

$$X, Y, T, \Gamma, A, M, Q, L, H, C, P, N.$$

The handoff reads source completeness, target completeness, graph membership, the exact norm-sum row, and the graph-closedness row before exporting graph-norm completion data.

Lean4 ✓ `BEDC.Derived.BanachOperatorGraphNormUp.BanachOperatorGraphNormCompletionHandoff`

Proof. Start with the Banach source and target rows X and Y. Their metric and complete-metric data provide the only Cauchy and limit witnesses available to the graph-norm packet.

Next read the graph rows T, Γ, A. The action row and graph-membership row determine which source endpoints are in the graph-norm domain, and the admission row prevents a consumer from reading outside the displayed graph.

Use M and Q for the graph norm itself. Metric readback supplies the source and target distances, while the norm-sum row records the exact real-valued graph-norm expression.

Finally apply L, H, C, P, N. The completion handoff replays graph-closedness through the same rows, then transport, continuation, provenance, and local naming preserve the packet. No quotient graph or external selector is exported. □

Theorem 6353.4 (Banach operator graph-norm no quotient graph)**.** *No accepted* $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ *consumer can replace the displayed graph row* Γ *by a quotient graph, host function graph, or selected representative while retaining the local* $\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}$ *read.*

Lean4 ✓ `BEDC.Derived.BanachOperatorGraphNormUp.BanachOperatorGraphNorm no quotient graph`

Proof. Project an accepted carrier $G = (X, Y, T, \Gamma, A, M, Q, L, H, C, P, N)$. The only graph coordinate is Γ, and graph-domain admission is mediated by A.

The metric and completion rows do not add another graph object. They only measure source and target reads, seal the norm-sum value, and replay graph-closedness through the completion handoff.

Transport, continuation, provenance, and local naming preserve the same displayed coordinates. Since the carrier contains no quotient graph, host function graph, or selected representative row, replacing Γ by such an object leaves the accepted NameCert surface. □

Closure status for $\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6353.1, Definition 6353.2, Definition 6353.3, and Definition 6353.4 seed the graph-norm packet over displayed Banach source and target rows, operator graph rows, metric readback, exact norm-sum data, and completion handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BanachOperatorGraphNorm}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close Lean verification, bridge checking, closed-operator calculus, inverse mapping, quotient graph equality, host operator spaces, or selected representatives.

Upgrade path: Obligation closure requires checked carrier admission, graph-closedness stability, norm-sum exactness, completion handoff, and no-quotient-graph nonescape for BEDC.Derived.BanachOperatorGraphNormUp.

Constructive story: A $\mathsf{BanachOperatorGraphNorm}^{\uparrow}$ packet is generated from Banach source and target rows, an operator action row, displayed graph membership, graph-domain admission, metric readback, an exact RealUp norm-sum row, completion handoff, componentwise transport, continuation replay, provenance, and local naming.

Chapter 6354

A Concrete Naming Certificate for $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$

Following Krantz and Parks and Bishop–Bridges constructive analysis, the $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ packet records the finite route by which a local equation solve may be read from displayed derivative, invertibility, Newton, completion, and real-name data. It sits beside the Newton iteration route of chapter 223, the derivative source of chapter 122, the real-name surface of chapter 38, the completion boundary of chapter 95, and the metric context of chapter 57. The chapter does not assert a host implicit-function theorem; it names the finite NameCert carrier a later constructive calculus consumer must inspect.

Definition 6354.1 (Implicit function theorem carrier)**.** *An* $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (E, D, I, N, K, R, H, C, P, L)$$

with the following displayed rows:

1. *E is the equation packet, recording the finite source variables, target variable, base point, local equation value, and classifier for the read $F(x, y) = 0$.*
2. *D is the* $\mathsf{Derivative}^{\uparrow}$ *row for the equation packet at the displayed base point, including the partial derivative history read by the target coordinate.*
3. *I is the invertibility window, naming the displayed inverse witness, local radius, and tolerance budget in the same* $\mathsf{Met}^{\uparrow}$ *context as E and D.*
4. *N is the* $\mathsf{NewtonIteration}^{\uparrow}$ *route, recording the correction step, residual budget, nonescape window, and finite stopping schedule used by the local solve.*
5. *K is the* $\mathsf{Completion}^{\uparrow}$ *and* $\mathsf{Real}^{\uparrow}$ *seal, requiring every limiting or real-facing read to pass through the displayed completion row before the Real name is exposed.*
6. *R is the transport row from the solved Newton endpoint to the local graph read, keeping the equation packet, derivative row, invertibility window, Newton route, completion seal, and Real seal componentwise aligned.*
7. *H records componentwise* hsame *transport for equation packets, derivative histories, invertibility witnesses, Newton schedules, Completion and Real seals, transport rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from a local solve request through $E, D, I, N, K, R, H, P, L$.*

9. *P is the* Pkg *provenance over* $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$, $\mathsf{NewtonIteration}^{\uparrow}$, $\mathsf{Derivative}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{Completion}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert *rows.*
10. *L is the local* $\mathsf{NameCert}_{\mathsf{ImplicitFunctionTheorem}^{\uparrow}}$ *row binding the packet to the displayed route.*

The classifier compares accepted packets only through $E, D, I, N, K, R, H, C, P, L$. The carrier has no coordinate for a host implicit function, a selected germ, global inverse theorem, quotient function equality, ambient choice, or an untracked completeness principle.

Lean4 (statement) `BEDC.Derived.ImplicitFunctionTheoremUp`

Theorem 6354.2 (Implicit function theorem NameCert obligations)**.** *For every accepted* $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ *carrier*

$$T = (E, D, I, N, K, R, H, C, P, L),$$

the local $\mathsf{NameCert}_{\mathsf{ImplicitFunctionTheorem}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *carrier admission for the displayed equation packet, derivative row, invertibility window, Newton route, Completion and Real seal, transport, replay, provenance, and local name;*
2. *derivative-invertibility compatibility, requiring D and I to display the same target-coordinate approximation inside the same* $\mathsf{Met}^{\uparrow}$ *window;*
3. *Newton residual replay, requiring N and C to keep every correction step downstream of E, D, I and inside the named nonescape window;*
4. *Completion and Real sealing, requiring K to be the only route from the Newton residual schedule to a real-facing endpoint;*
5. *graph transport, requiring R, H to preserve the equation packet, derivative row, invertibility window, Newton route, and seal rows componentwise;*
6. *provenance exactness, requiring P and L to name only the listed rows and no host implicit function theorem or selected germ.*

No local NameCert obligation is stored in an ambient function space or external completeness theorem.

Proof. Carrier admission is projection from Definition 6354.1. The packet lists exactly the equation, derivative, invertibility, Newton, seal, transport, replay, provenance, and local naming rows.

The analytic obligations are internal to D, I, N. The derivative and invertibility rows name the same local approximation, while the Newton row records the displayed correction schedule and residual budget.

The endpoint obligations are downstream of that schedule. The Completion and Real seal row K is reached only after the Newton route has been replayed, and R, H transport the resulting local graph read without replacing it by a host solution object.

Finally the provenance and local naming rows bind the same finite route. Since the carrier contains no selected germ, quotient function equality, global inverse theorem, ambient choice, or untracked completeness coordinate, these rows exhaust the NameCert surface. □

Theorem 6354.3 (Implicit function theorem Newton-completion handoff)**.** *Every* $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ *local-solve read factors through the displayed finite route*

$$\mathsf{Derivative}^{\uparrow}, \mathsf{Met}^{\uparrow}, \mathsf{NewtonIteration}^{\uparrow}, \mathsf{Completion}^{\uparrow}, \mathsf{Real}^{\uparrow}, \mathsf{BHist}, \mathsf{hsame}, \mathsf{Cont}, \mathsf{Pkg}, \mathsf{NameCert}.$$

In the notation of Definition 6354.1, the consumer must open $E, D, I, N, K, R, H, C, P, L$ before it can export a local graph endpoint.

Proof. Project the accepted carrier. The local-solve request has no available coordinate except the equation packet, derivative row, invertibility window, Newton route, Completion and Real seal, transport, replay, provenance, and local naming rows.

The derivative and metric rows provide the local analytic source. The Newton route consumes that source through the invertibility window and records the finite correction schedule together with the residual nonescape budget.

The handoff to Completion and Real data is then forced through K. The transport and replay rows preserve the same displayed endpoint before provenance and local naming expose it as a local graph read, so no host implicit-function theorem or selected germ is read. □

Falsifiable prediction: Every $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ local-solve read opens one equation packet, one Derivative row, one invertibility window in a MetricSpace context, one NewtonIteration route, one Completion and Real seal, graph transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before exporting a graph endpoint.

Independence witness: newtoniteration, derivative, real, completion, metricspace: $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries equation data, derivative data, an invertibility window, Newton residual replay, Completion and Real sealing, transport, provenance, and local naming as one certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{ImplicitFunctionTheorem}^{\uparrow}}$.

Constructive story: An $\mathsf{ImplicitFunctionTheorem}^{\uparrow}$ packet is generated from an equation packet, derivative row, invertibility window, NewtonIteration route, Completion and Real seal, graph transport, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ImplicitFunctionTheorem}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6354.1, Definition 6354.2, and Definition 6354.3 seed the finite implicit-function-theorem certificate over equation, derivative, invertibility, Newton, Completion, Real, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ImplicitFunctionTheorem}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.ImplicitFunctionTheoremUp`

Bridge status: none

Not claimed: This chapter does not close host implicit functions, selected germs, quotient function equality, global inverse theorem packages, ambient choice, full analytic existence, Lean theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, derivative-invertibility compatibility, Newton residual replay, Completion and Real sealing, graph transport, provenance exactness, and local naming for BEDC.Derived.ImplicitFunctionTheor

Chapter 6355

Picard Iteration Budget NameCert Construction

Following Picard's successive approximations and Banach's contraction principle in Bishop–Bridges constructive analysis, the $\mathsf{PicardIterationBudget}^{\uparrow}$ packet records the finite budget surface that a contraction orbit must expose before endpoint-facing completion data are read. It sits between the Cauchy contraction orbit of chapter 2350, the Banach contraction row of chapter 3529, the Picard ledger of chapter 2359, and the terminal real-name seal of chapter 38. The packet names the finite means of an iteration budget; it does not assert a general completeness theorem, choose an endpoint, identify quotient reals, or import countable choice.

Definition 6355.1 (Picard iteration budget carrier)**.** *A* $\mathsf{PicardIterationBudget}^{\uparrow}$ *carrier is a finite packet*

$$B = (S, K, W, D, E, H, C, P, N)$$

with the following displayed rows.

1. *S is the seed row, naming the initial point, accepted metric source, and contraction orbit entry used by the budget.*
2. *K is the contraction-modulus row, storing the displayed contraction comparison and the finite modulus read from the Banach contraction surface.*
3. *W is the iterate-window row, listing the finite interval of Picard iterates inspected by the budget request.*
4. *D is the dyadic error-budget row, recording the requested tolerance, the residual comparison, and the finite arithmetic certificate that the window is long enough.*
5. *E is the endpoint-handoff row, exposing only the completion-facing or Real-facing boundary available after S, K, W, D have been displayed.*
6. *H records componentwise* hsame *transport for the seed, modulus, window, dyadic error budget, endpoint handoff, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from the seed row through the contraction modulus, iterate window, dyadic budget, endpoint handoff, transport, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{PicardIterationBudget}^{\uparrow}$, $\mathsf{CauchyContractionOrbit}^{\uparrow}$, $\mathsf{BanachContraction}^{\uparrow}$, $\mathsf{PicardIteration}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.

9. N is the local $\mathsf{NameCert}_{\mathsf{PicardIterationBudget}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, K, W, D, E, H, C, P, N$. It has no coordinate for a hidden infinite orbit, a selected fixed point, quotient equality of endpoint names, countable choice, host function extensionality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6355.2 (Picard iteration budget NameCert obligations)**.** *For every accepted* $\mathsf{PicardIterationBudget}^{\uparrow}$ *carrier $B = (S, K, W, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{PicardIterationBudget}^{\uparrow}}$ *surface consists exactly of carrier admission for the seed row, contraction-modulus row, iterate-window row, dyadic error-budget row, endpoint-handoff row, componentwise transport, displayed replay, provenance, and local naming.*

The classifier agreement obligation requires two accepted packets to agree precisely when these rows agree componentwise through the displayed hsame *route. The stability obligation requires the same row order to be preserved by* hsame *transport and* Cont *replay before any endpoint handoff is consumed.*

Proof. Project the carrier of Definition 6355.1. The analytic part is S, K, W, D, E; the structural certificate part is H, C, P, N.

The seed row and contraction-modulus row determine what orbit may be inspected. The iterate-window row then restricts the read to finitely many Picard rows, and the dyadic error-budget row records the tolerance comparison attached to that window.

Endpoint handoff is terminal for the budget route. Since E is downstream of S, K, W, D, no consumer can read endpoint data without first exposing the finite seed, modulus, iterate window, and dyadic budget rows.

Finally H, C, P, N transport, replay, source, and name the same rows. No excluded object is a carrier coordinate, so no hidden infinite orbit, selected fixed point, quotient endpoint equality, choice principle, host extensionality rule, or ambient Real-completeness theorem is a local obligation. □

Theorem 6355.3 (Picard iteration budget exactness)**.** *Every endpoint-facing read of an accepted* $\mathsf{PicardIterationBudget}^{\uparrow}$ *carrier factors through*

$$\mathsf{CauchyContractionOrbit}^{\uparrow} \longmapsto \mathsf{BanachContraction}^{\uparrow} \longmapsto \mathsf{PicardIteration}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The finite budget is exact in the following sense: it exposes the seed row, contraction modulus, iterate window, dyadic error budget, and endpoint handoff as displayed rows, and it exports no additional infinite orbit data beyond those rows.

Proof. Begin with the NameCert obligations of Definition 6355.2. They place the seed row, contraction-modulus row, iterate-window row, and dyadic budget row before the endpoint handoff.

The Cauchy contraction orbit and Banach contraction rows supply the visible contraction data. The Picard iteration row supplies only the finite window requested by W, while D checks the requested dyadic tolerance against that finite window.

The endpoint handoff row E is reached only after those rows have been displayed. It may feed completion-facing or Real-facing consumers, but it does not add a selected endpoint, quotient equality, host extensionality, countable choice, hidden tail stream, or ambient $\mathsf{Real}^{\uparrow}$ completeness coordinate.

Thus the budget route is exact for finite inspection: every exported endpoint read is tied to the displayed rows of B, and no unlisted orbit data can be consumed by the local certificate. □

Theorem 6355.4 (Picard iteration budget Cauchy-tail handoff)**.** *Let $B = (S, K, W, D, E, H, C, P, N)$ be an accepted* $\mathsf{PicardIterationBudget}^{\uparrow}$ *carrier whose displayed rows include a contraction row, a finite Picard iteration ledger, a ratio-tail budget, a* $\mathsf{CompleteMetric}^{\uparrow}$ *boundary, and a* $\mathsf{RegSeqRat}^{\uparrow}$ *handoff. For every tolerance history accepted by the dyadic error-budget row, the packet exports a unary stage after which every displayed pair of Picard iterates in the window is within the* $\mathsf{RegSeqRat}^{\uparrow}$ *tolerance read by the handoff row.*

The exported tail ledger is finite. It is transported componentwise by hsame*, replayed only through the displayed* Cont *rows, and consumed by completion and Banach fixed-point chapters without selected endpoints, host functions, quotient equality of completed names, or countable choice.*

Proof. Project the carrier rows. The seed row S and contraction row K fix the accepted orbit and the displayed contraction comparison; the window row W names the finite Picard rows that may be inspected.

Use the contraction row to shrink adjacent residuals inside the displayed window. The ratio-tail part of D then dominates the finite sum of those residuals for the requested tolerance history, so the same row supplies a unary stage for the whole tail comparison.

Route this finite comparison through the $\mathsf{RegSeqRat}^{\uparrow}$ readback and the $\mathsf{CompleteMetric}^{\uparrow}$ boundary named by the endpoint handoff row E. The structural rows H, C, P, N transport, replay, source, and name the same ledger, and no selected endpoint or quotient equality is introduced after the finite tail stage has been exposed. □

Theorem 6355.5 (Picard iteration budget ledger nonescape)**.** *For an accepted* $\mathsf{PicardIterationBudget}^{\uparrow}$ *carrier, the ledger visible to downstream consumers is exhausted by the seed, contraction modulus, iterate window, dyadic error budget, endpoint handoff, transport, replay, provenance, and local naming rows. In particular, a downstream consumer cannot recover a hidden infinite orbit, a selected fixed point, quotient equality of completed endpoints, countable choice, host function extensionality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness from the budget packet.*

Proof. Use Definition 6355.1. The carrier lists exactly $S, K, W, D, E, H, C, P, N$.

The analytic ledger is exhausted by S, K, W, D, E. These rows record the seed, contraction comparison, finite iterate window, dyadic tolerance comparison, and endpoint handoff.

The structural ledger is exhausted by H, C, P, N. These rows preserve and name the analytic route, but they do not introduce another endpoint source or an infinite-orbit coordinate.

The excluded objects are absent from both ledgers. Therefore they cannot be recovered from the local NameCert surface of the accepted packet. □

Falsifiable prediction: Every accepted $\mathsf{PicardIterationBudget}^{\uparrow}$ endpoint-facing read must expose a seed row, contraction-modulus row, finite iterate-window row, dyadic error-budget row, endpoint-handoff row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a completion or Real-facing consumer reads endpoint data.

Independence witness: cauchy-contraction-orbit, banach-contraction, picard-iteration, real: $\mathsf{PicardIterationBudget}^{\uparrow}$ is not bijective to any listed sibling because it binds a seed row, contraction modulus, finite iterate window, dyadic error budget, endpoint handoff, transport, replay, provenance, and local naming in one finite budget packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PicardIterationBudget}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PicardIterationBudget}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6355.1, Definition 6355.2, Definition 6355.3, and Definition 6355.5 seed the finite budget interface from contraction orbit data to endpoint-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PicardIterationBudget}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close general completeness, selected fixed points, quotient endpoint equality, countable choice, host function extensionality, hidden infinite orbit data, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, classifier agreement, hsame and Cont stability, ledger nonescape, finite budget exactness, and local naming for BEDC.Derived.PicardIterationBudgetUp.

Constructive story: A PicardIterationBudget$^{\uparrow}$ packet is generated from a seed row, contraction modulus, finite Picard iterate window, dyadic error budget, endpoint handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6356

Kernel Budget Flow NameCert Construction

The $\mathsf{KernelBudgetFlow}^{\uparrow}$ packet realizes the finite kernel-budget flow suggested by the kernel vision hub Definition 9350.3 as concrete NameCert rows. It sits beside the governance route of chapter 9467, the kernel audit witness of chapter 546, the kernel acceptance witness of chapter 549, the finite multihist packet of chapter 468, and the kernel inductive acceptance row of chapter 915. The packet exposes BMark, BHist, Cont, Pkg, Gap, ledger, and NameCert rows as a finite flow boundary; it does not certify the host kernel, assert global soundness, hide an acceptance trace, or supply a theorem proof.

Definition 6356.1 (Kernel budget flow carrier)**.** *A* $\mathsf{KernelBudgetFlow}^{\uparrow}$ *carrier is a finite packet*

$$F = (B, H, K, G, L, C, P, N)$$

with the following displayed rows.

1. *B is the* BMark *budget row, naming the finite kernel-facing request and the bounded inspection scope.*
2. *H is the* BHist *flow row, recording the ordered budget events accepted by the packet.*
3. *K is the kernel-scope row, naming the accepted kernel witness or audit witness that the flow is allowed to inspect.*
4. *G is the* Gap *row, recording refused, absent, or not-yet-certified budget edges without erasing them.*
5. *L is the ledger row, pairing every visible budget event with the source, acceptance, gap, and consumer rows it may reach.*
6. *C is the displayed* Cont *replay row from budget request through history, kernel scope, gap accounting, ledger readback, provenance, and local naming.*
7. *P is the* Pkg *provenance over* $\mathsf{KernelBudgetFlow}^{\uparrow}$*,* $\mathsf{KernelAuditWitness}^{\uparrow}$*,* $\mathsf{KernelAcceptanceWitness}^{\uparrow}$*,* $\mathsf{KernelInductiveAcceptance}^{\uparrow}$*,* $\mathsf{FiniteMultiHistPacket}^{\uparrow}$*,* BMark*,* BHist*,* Gap*,* Cont*, and* $\mathsf{NameCert}$*.*
8. *N is the local* $\mathsf{NameCert}_{\mathsf{KernelBudgetFlow}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through B, H, K, G, L, C, P, N. It has no coordinate for host-kernel soundness, unrestricted proof checking, hidden traces, unbounded fuel, erased gap rows, or theorem-proof evidence.

Theorem 6356.2 (Kernel budget flow NameCert obligations)**.** *For every accepted* $\mathsf{KernelBudgetFlow}^{\uparrow}$ *carrier* $F = (B, H, K, G, L, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{KernelBudgetFlow}^{\uparrow}}$ *surface consists exactly of carrier admission for the BMark budget row, BHist flow row, kernel-scope row, Gap row, ledger row, displayed Cont replay, Pkg provenance, and local naming.*

The flow obligation requires each budget event in H *to be accounted for by* L *before a downstream consumer reads it. The gap obligation requires refused or unavailable edges to remain visible in* G *rather than being reclassified as accepted kernel evidence.*

Proof. Project the carrier of Definition 6356.1. The rows B, H, K, G, L are the budget-flow data, while C, P, N replay, source, and name the same data.

The BMark row bounds the request, and the BHist row orders the events. The kernel-scope row says which accepted witness may be inspected, while the Gap row keeps unavailable or refused edges visible.

The ledger row is the exact accounting surface. Every downstream read of a budget event must pass through L, so the packet cannot replace the ledger by a hidden trace or a host-kernel endorsement.

Finally, C, P, N preserve and name the displayed route. Since no excluded object is a carrier coordinate, no local NameCert obligation can export unrestricted proof checking, host soundness, unbounded fuel, or theorem-proof evidence. □

Theorem 6356.3 (Kernel budget flow finite realization)**.** *Every accepted* $\mathsf{KernelBudgetFlow}^{\uparrow}$ *read factors through displayed*

$$\mathsf{BMark},\ \mathsf{BHist},\ \mathsf{Cont},\ \mathsf{Pkg},\ \mathsf{Gap},\ L,\ \mathsf{NameCert}$$

rows, with L *the finite ledger of budget-event accountability. Thus the packet realizes kernel-budget flow as finite BEDC data rather than as an ambient kernel theorem, an untracked execution trace, or a hidden proof object.*

Proof. Start from the NameCert obligations of Definition 6356.2. They expose the BMark request, BHist flow, kernel-scope witness, Gap row, ledger, replay, provenance, and local naming.

The budget and history rows fix the finite event surface. The kernel-scope row restricts which accepted witness the flow may inspect, and the gap row preserves refused or missing edges.

The ledger row L is then the only route from events to consumers. The replay and provenance rows do not create additional events; they preserve the finite route already displayed.

Therefore the realization is finite and row-local. It exports no ambient kernel theorem, hidden trace, theorem proof, or unbounded budget source. □

Falsifiable prediction: Every accepted $\mathsf{KernelBudgetFlow}^{\uparrow}$ read must expose BMark budget data, BHist event flow, kernel-scope evidence, Gap accounting, finite ledger rows, displayed Cont replay, Pkg provenance, and a local NameCert row; Lean TasteGate work must later supply Nontrivial and FieldFaithful instances for this carrier.

Independence witness: kernel-audit-witness, kernelacceptancewitness, kernelinductiveacceptance, finite-multihist-packet, metacicclosuretrace: $\mathsf{KernelBudgetFlow}^{\uparrow}$ is not bijective to any listed sibling because it binds bounded kernel request data, event history, accepted-witness scope, gap accounting, finite ledger readback, replay, provenance, and local naming in one flow packet.

Closure status for $\mathsf{NameCert}_{\mathsf{KernelBudgetFlow}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{KernelBudgetFlow}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6356.1, Definition 6356.2, and Definition 6356.3 seed finite kernel-budget flow as displayed BMark, BHist, Cont, Pkg, Gap, ledger, and NameCert rows.

Formal verification: FormalStatus(NameCert$_{\mathsf{KernelBudgetFlow}\uparrow}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close host-kernel soundness, unrestricted proof checking, hidden acceptance traces, unbounded fuel, erased gap rows, theorem-proof evidence, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, flow accountability, gap visibility, finite ledger exactness, Nontrivial and FieldFaithful TasteGate instances, and local naming for BEDC.Derived.KernelBudgetFlowUp.

Constructive story: A KernelBudgetFlow$^{\uparrow}$ packet is generated from BMark budget rows, BHist event flow, kernel-scope evidence, Gap accounting, finite ledger rows, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6357

A Concrete Naming Certificate for CompactMetricContinuousImageCompact$^{\uparrow}$

Following Bishop-style compact metric analysis, the continuous image of compact metric data is compact only after two finite rows are visible: total bounded image nets and completeness for the target image route. The CompactMetricContinuousImageCompact$^{\uparrow}$ packet records that theorem-facing route as finite BEDC data. It consumes the compact metric source of chapter 129, the continuous-map graph of chapter 130, the tolerance routing of chapter 369, the finite image-net packet of chapter 3605, the total-bounded image carrier of chapter 3924, and the complete-metric handoff of chapter 128. The packet names a compact-metric image row; it does not assert open-cover compactness, selected inverse images, quotient equality of target names, or ambient Real$^{\uparrow}$ completeness.

Definition 6357.1 (Compact metric continuous-image compact carrier)**.** *A* CompactMetricContinuousImageCompa *carrier is a finite* BHist *packet*

$$J = (K, F, U, I, T, E, H, C, P, N)$$

with the following displayed rows:

1. *K is the* CompactMetric$^{\uparrow}$ *source row, exposing finite source-net and compact-metric provenance.*
2. *F is the* ContinuousMap$^{\uparrow}$ *graph row from the source carrier to the target metric carrier.*
3. *U is the* UniformModulus$^{\uparrow}$ *tolerance route used by the same displayed map row.*
4. *I is the* CompactMetricContinuousImage$^{\uparrow}$ *finite image-net route from source nets to target image centers.*
5. *T is the* CompactContinuousImageTotalBounded$^{\uparrow}$ *row that packages those image centers as total-bounded target-image data.*
6. *E is the* CompleteMetric$^{\uparrow}$ *row available for the displayed target-image route.*
7. *H records componentwise* hsame *transport for the compact source, map row, modulus row, finite image-net row, total-bounded row, complete-metric row, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact-image request through $K, F, U, I, T, E, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetricContinuousImage}^{\uparrow}$, $\mathsf{CompactContinuousImageTotalBounded}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, F, U, I, T, E, H, C, P, N$. It has no coordinate for open-cover compactness, a host image subset, selected preimages, quotient equality of target names, choice-selected target centers, or ambient real completeness.

Theorem 6357.2 (Compact metric continuous-image compact obligations)**.** *For every accepted* $\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}$ *carrier $J = (K, F, U, I, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactMetricC}}$ *surface consists exactly of carrier admission, compact source-net accountability, continuous-map graph routing, uniform-modulus compatibility, finite image-net exactness, image total-bounded export, complete-metric handoff, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6357.1. The source, map, and tolerance rows are K, F, U, while the image-facing compactness route is I, T, E.

The finite image-net row I may be read only after the source compact metric row and the displayed modulus row have routed a target tolerance to a source-net request. The total-bounded row T repackages those image centers without adding a host image set.

The complete-metric row E is read after the total-bounded image row. Transport, replay, provenance, and local naming are supplied by H, C, P, N, so the compact-metric image row is assembled from displayed finite data.

Since the carrier has no open-cover, selected-preimage, quotient-equality, choice, or ambient-completeness coordinate, the local NameCert surface cannot export those principles. □

Theorem 6357.3 (Compact metric continuous-image compact route)**.** *Every compact-image consumer of an accepted* $\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}$ *packet receives only the finite BEDC route*

$\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetricContinuousImage}^{\uparrow}$, CompactContinuous

The exported compactness row is a $\mathsf{CompactMetric}^{\uparrow}$*-style row for the displayed image route, assembled from image nets plus completeness, not an open-cover compactness theorem, not a selected inverse-image construction, not quotient target equality, and not ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with a compact-image request. The request first reads the compact source row K, the continuous-map graph row F, and the uniform-modulus row U.

Next apply the finite image-net route I. It reads source nets and displayed graph values to produce target image centers. Apply the total-bounded image row T to expose those centers as total-bounded target-image data.

Finally read the complete-metric row E for the same displayed target-image route. The structural rows H, C, P, N transport, replay, source, and name this route componentwise.

The construction is exhausted by $K, F, U, I, T, E, H, C, P, N$. Therefore a consumer receives compact-metric image data from finite image nets plus completeness, with none of the excluded open-cover, selected-preimage, quotient, choice, or ambient-completeness objects. □

Closure status for $\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}}$ **·**
Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactMetricContinuousImageCompact}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6357.1, Definition 6357.2, and Definition 6357.3 seed the compact-metric continuous-image compact route from finite image nets and complete-metric handoff to a compact-metric image row.

Formal verification: FormalStatus(NameCert$_{\text{CompactMetricContinuousImageCompact}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected inverse images, quotient target equality, choice-selected image centers, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, image-net exactness, total-bounded image export, complete-metric handoff, transport, replay, provenance, and local NameCert registration for BEDC.Derived.CompactMetricContinuousIma

Constructive story: A CompactMetricContinuousImageCompact$^{\uparrow}$ packet is generated from CompactMetric source nets, ContinuousMap graph rows, UniformModulus tolerance routing, CompactMetricContinuousImage image-net data, CompactContinuousImageTotalBounded total-bounded rows, CompleteMetric handoff, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6358

A Concrete Naming Certificate for $\mathsf{TheoremGenomeEvolution}^{\uparrow}$

The $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ packet realizes the theorem-genome evolution audit of chapter 9409 as finite NameCert data. It reads theorem-flow recognition through the recognizer-flow carrier of chapter 1048, theorem-DNA scoring through chapter 6243, classifier-level information through chapter 9366, and the theorem-DNA scoring taste gate through chapter 9460. The packet exists so discovery classifiers can inspect genome flow, mutation, selection, classifier shift, transport, provenance, and local naming rows rather than relying on narrative methodology.

Definition 6358.1 (Theorem genome evolution carrier)**.** *A* $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$G = (F, R, D, Q, M, S, L, H, C, P, N)$$

with the following displayed rows:

1. *F is the theorem-flow row, recording source theorem objects, proof loci, statement loci, certificate loci, and consumer-facing theorem roles.*
2. *R is the recognizer row, recording which theorem-flow entries are admitted as genome-bearing theorem records.*
3. *D is the theorem-DNA row, recording role-complete feature, circuit, classifier, stability, intervention, ledger, and seal surfaces.*
4. *Q is the classifier-shift row, recording the before and after classifier phenotype when a mutation claims discovery-facing movement.*
5. *M is the mutation row, recording compression, recombination, certificate insertion, ledger exposure, and classifier-shifting genome moves.*
6. *S is the selection row, recording the score, taste-gate, novelty, reuse, and downstream-consumer coordinates used to rank genome moves.*
7. *L is the ledger row, recording which mutation or selection reads are admitted, refused, or held as obligations.*
8. *H records componentwise* hsame *transport for theorem-flow, recognizer, theorem-DNA, classifier-shift, mutation, selection, ledger, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from a theorem-genome audit request through $F, R, D, Q, M, S, L, H, P, N$.*

10. *P is the* Pkg *provenance over* $\mathsf{TheoremGenomeEvolution}^{\uparrow}$*, theorem-flow recognition, classifier-level information, theorem-DNA scoring, theorem-DNA scoring taste gate, mutation, selection, transport,* BHist, hsame, Cont, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{TheoremGenomeEvolution}^{\uparrow}}$ *row.*

The carrier contains no coordinate for unledgered discovery, score-only discovery, theorem sentence identity as genome identity, proof compression as classifier shift, or a mutation whose before and after classifier rows are not displayed.

Theorem 6358.2 (Theorem genome evolution NameCert obligations)**.** *For every accepted* TheoremGenomeEvolutio *carrier $G = (F, R, D, Q, M, S, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TheoremGenomeEvolution}^{\uparrow}}$ *surface consists exactly of carrier admission, theorem-flow exposure, recognizer admission, theorem-DNA exactness, classifier-shift visibility, mutation classification, selection accounting, ledger non-escape, componentwise transport, displayed replay, provenance, and local naming.*

Proof. Project the carrier of Definition 6358.1. The theorem-facing rows are F, R, D, Q, M, S, L, and the structural rows are H, C, P, N.

The theorem-flow and recognizer rows identify the genome-bearing theorem records. The theorem-DNA row supplies the role-complete loci before any mutation or selection row can be inspected.

Classifier-shift visibility is supplied by Q. A mutation row M may be read as discovery-facing only when its before and after classifier phenotypes are displayed there; compression or recombination without Q remains a genome move but not a classifier-shift claim.

Selection accounting is supplied by S and bounded by the ledger row L. Transport, replay, provenance, and local naming through H, C, P, N preserve the same finite route, so no unledgered discovery or score-only discovery can escape. □

Theorem 6358.3 (Theorem genome evolution audit route)**.** *Every discovery-classifier consumer of an accepted* $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ *packet must read*

$$F, \quad R, \quad D, \quad Q, \quad M, \quad S, \quad L, \quad H, \quad C, \quad P, \quad N.$$

Thus theorem-genome evolution is a finite NameCert packet over theorem-flow, recognizer, theorem-DNA, classifier-shift, mutation, selection, transport, provenance, and local naming rows, not a prose-only discovery claim.

Proof. Begin with a discovery-classifier request. The theorem-flow row F and recognizer row R identify the theorem objects that can carry genome data.

Read the theorem-DNA row D next. It fixes the displayed role loci, so a later mutation cannot replace a theorem genome by a sentence-only surface.

Then read the classifier-shift row Q, mutation row M, and selection row S. The ledger row L records whether that route is admitted, refused, or left as an obligation.

Finally transport and replay the route through H and C, and source and name it through P and N. These rows exhaust the accepted carrier, so the consumer receives finite audit data rather than an uninspected methodology claim. □

The Lean follow-up file for $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ must provide explicit Nontrivial $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ and FieldFaithful $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ TasteGate instances, preserving $F, R, D, Q, M, S, L, H, C, P, N$ as separate displayed fields.

Falsifiable prediction: Deleting the classifier-shift row Q while retaining theorem-flow, theorem-DNA, mutation, and selection rows leaves a genome move but invalidates a discovery-facing theorem-genome evolution carrier.

Independence witness: theorem-dna-complexity-scoring, classifier-level-information, theorem-dna-scoring-taste-gate, theorem-genome-evolution: none is bijective because this packet uniquely binds theorem flow, recognizer admission, theorem-DNA rows, classifier-shift evidence, mutation classes, selection accounting, ledger verdicts, transport, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{TheoremGenomeEvolution}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{TheoremGenomeEvolution}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6358.1, Definition 6358.2, and Definition 6358.3 seed the theorem-genome evolution packet as finite theorem-flow, recognizer, theorem-DNA, classifier-shift, mutation, selection, ledger, transport, provenance, and NameCert data.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{TheoremGenomeEvolution}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim positive discovery without classifier shift, score-only discovery, proof compression as discovery, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, theorem-flow exposure, recognizer admission, theorem-DNA exactness, classifier-shift visibility, mutation classification, selection accounting, ledger non-escape, transport, replay, provenance, and local NameCert registration for BEDC.Derived.TheoremGenomeEvolutionUp.

Constructive story: A $\mathsf{TheoremGenomeEvolution}^{\uparrow}$ packet is generated from theorem-flow rows, recognizer rows, theorem-DNA rows, classifier-shift rows, mutation rows, selection rows, ledger verdicts, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6359

A Concrete Naming Certificate for $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$

Following Bourbaki–Kelley on uniform spaces and Bishop–Bridges constructive completion, the $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ packet records the finite NameCert surface of a completion read that is uniform, regular, and selection-free. It sits after the uniform source of chapter 339, the completion boundary of chapter 1473, the filter and net Cauchy routes of chapter 333 and chapter 589, the finite stream windows of chapter 126, the regular rational readback of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names the constructive completion route as displayed rows rather than as a quotient completion, chosen limit, countable-choice schedule, host equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness theorem.

Definition 6359.1 (Constructive uniform completion carrier)**.** *A* $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ *carrier is a finite packet*

$$U = (S, F, T, W, R, D, E, H, C, P, N)$$

with the following displayed rows.

1. *S is the uniform-source row naming the accepted* $\mathsf{UniformSpace}^{\uparrow}$ *surface, entourage ledger, and separatedness data visible to the completion request.*
2. *F is the Cauchy-filter row read through* $\mathsf{CauchyFilter}^{\uparrow}$*, including the finite filter-basis observations used by the completion route.*
3. *T is the Cauchy-net row read through* $\mathsf{CauchyNet}^{\uparrow}$*, including the directed finite comparison windows paired with the filter row.*
4. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposed before any regular readback is consumed.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational readback row attached to the finite stream windows.*
6. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger that records the positive dyadic scales used by the uniform Cauchy comparison.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row available only after S, F, T, W, R, D have been displayed.*
8. *H records componentwise* hsame *transport for the source, filter, net, window, readback, tolerance, seal, replay, provenance, and local naming rows.*
9. *C records displayed* Cont *replay from the uniform source through the Cauchy rows, finite windows, regular readback, dyadic tolerance ledger, Real seal, transport, provenance, and local naming.*

10. *P is the* Pkg *provenance over* $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$, $\mathsf{UniformSpace}^{\uparrow}$, $\mathsf{UniformCompletion}^{\uparrow}$, $\mathsf{CauchyFilter}^{\uparrow}$, $\mathsf{CauchyNet}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

11. *N is the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, F, T, W, R, D, E, H, C, P, N$. It has no coordinate for quotient completion classes, selected representatives, arbitrary choice of Cauchy subdata, host real equality, or ambient completeness.

Theorem 6359.2 (Constructive uniform completion NameCert obligations)**.** *For every accepted* $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ *carrier $U = (S, F, T, W, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ConstructiveUniformCon}}$ *surface consists exactly of carrier admission for the uniform-source row, Cauchy-filter row, Cauchy-net row, finite stream-window row, regular rational readback row, dyadic tolerance ledger, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming.*

The classifier agreement obligation requires two accepted packets to agree precisely when these displayed rows agree componentwise through hsame*. The stability obligation requires* hsame *transport and* Cont *replay to preserve the same row order before any completion-facing or Real-facing consumer reads the terminal seal.*

Proof. Project the carrier of Definition 6359.1. The analytic completion route is S, F, T, W, R, D, E, and the structural certificate route is H, C, P, N.

The uniform-source row fixes the entourage surface. The Cauchy-filter and Cauchy-net rows then expose the two finite Cauchy presentations that the completion request may compare.

The finite stream window, regular rational readback, and dyadic tolerance ledger form the computational readback route. The terminal Real seal is downstream of those rows, so it cannot be consumed before the finite Cauchy and tolerance data have been displayed.

Finally H, C, P, N transport, replay, source, and name the same rows. Since no excluded object is present as a coordinate, no quotient completion class, selected representative, arbitrary choice schedule, host equality, or ambient completeness theorem is a local obligation. □

Theorem 6359.3 (Constructive uniform completion uniform route)**.** *Every completion-facing read of an accepted* $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ *carrier factors through the displayed route*

$$\mathsf{UniformSpace}^{\uparrow} \longmapsto \mathsf{CauchyFilter}^{\uparrow}, \mathsf{CauchyNet}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The route exports a finite uniform Cauchy readback and terminal seal, not a host quotient completion, selected completed representative, countable-choice schedule, detached equality of completed names, or ambient $\mathsf{Real}^{\uparrow}$ *completeness theorem.*

Proof. Start with the NameCert obligations of Definition 6359.2. They place the uniform-source row and the two Cauchy presentation rows before finite stream windows and regular rational readback.

Read the completion request through chapter 1473. In the present packet, the completion boundary receives only S, F, T, W, R, D, E as its analytic rows.

The dyadic tolerance ledger gives the positive scales used by the uniform comparison, and the Real seal is terminal for the read. Because the carrier has no coordinate for quotient completion classes, selected representatives, choice schedules, detached host equality, or ambient Real completeness, those objects cannot be exported by the route. □

Theorem 6359.4 (Constructive uniform completion nonescape)**.** *Every downstream consumer of an accepted* $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ *packet receives only the uniform source, Cauchy-filter row, Cauchy-net row, finite stream window, regular rational readback, dyadic tolerance ledger, Real seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *row. In particular, the packet supplies no quotient completion class, selected limit, countable-choice schedule, host equality of completed representatives, compactness principle, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Use Definition 6359.1. The displayed analytic rows are exactly S, F, T, W, R, D, E, and the structural rows are exactly H, C, P, N.

Apply the uniform route of Definition 6359.3. It forces completion-facing reads through the uniform source, Cauchy presentations, finite window, regular readback, dyadic tolerance ledger, and terminal Real seal.

The structural rows preserve and name that route, but they do not add another completion object or equality source. Therefore every downstream read is exhausted by the listed rows and cannot escape to quotient completions, selected limits, choice schedules, host equality, compactness, or ambient completeness. □

Falsifiable prediction: Every accepted $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ completion-facing read must expose the uniform source, Cauchy-filter row, Cauchy-net row, finite stream window, regular rational readback, dyadic tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local naming before a consumer reads completed data.

Independence witness: uniformspace, uniformcompletion, cauchyfilter, cauchynet, streamname, regseqrat, dyadicratcore, real: $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ is not bijective to any listed sibling because it binds a uniform source, two finite Cauchy presentations, finite stream windows, regular rational readback, dyadic tolerance data, terminal Real seal, transport, replay, provenance, and local naming in one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ConstructiveUniformCompletion}^{\uparrow}}$**.**

Constructive story: A $\mathsf{ConstructiveUniformCompletion}^{\uparrow}$ packet is generated from a uniform source row, Cauchy-filter row, Cauchy-net row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6359.1, Definition 6359.2, Definition 6359.3, and Definition 6359.4 seed the finite constructive uniform-completion interface from uniform Cauchy data to Real-facing reads.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ConstructiveUniformCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected representatives, countable choice, compactness, host equality of completed representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, classifier agreement, hsame and Cont stability, finite uniform-route support, ledger nonescape, and local naming for BEDC.Derived.ConstructiveUniformCompletionUp.

Chapter 6360

A Concrete Naming Certificate for KalmanBucyFilter$^{\uparrow}$

Following Kalman and Bucy's continuous-time filtering theory, the KalmanBucyFilter$^{\uparrow}$ packet records the finite BEDC certificate rows for a state estimator whose prediction, observation, covariance, gain, and replay data are all displayed before any downstream control consumer reads the filter. It sits beside the discrete KalmanFilter$^{\uparrow}$ packet of chapter 5102, the regulator rows of LQR$^{\uparrow}$ in chapter 296, the Gramian rows of ObservabilityGramian$^{\uparrow}$ in chapter 5787, and the finite observation and reachability certificates of ControlObservability$^{\uparrow}$ and ControlControllability$^{\uparrow}$ in chapter 294 and chapter 297. The packet names only the displayed finite ledger used by continuous-time estimator consumers; it does not claim a host stochastic process, analytic convergence theorem, or infinite-dimensional filtering semantics.

Definition 6360.1 (KalmanBucyFilter carrier)**.** *A* KalmanBucyFilter$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (X, Y, A, C, Q, R, P, L, \widehat{x}, \Phi, \Omega, G, E, T, H, Pkg, N)$$

with the following visible rows:

1. *X is the state row and Y is the observation row, each read through displayed* VecSpace$^{\uparrow}$ *and* Mat$^{\uparrow}$ *coordinates;*
2. *A is the state-flow row and C is the observation-flow row consumed from finite* DynSystem$^{\uparrow}$ *and* ControlObservability$^{\uparrow}$ *packets;*
3. *Q and R are the displayed process and observation covariance rows, and P is the Riccati covariance ledger row;*
4. *L is the gain row determined by the covariance, observation, and measurement-covariance ledgers;*
5. *$\widehat{x}$ is the estimator row, Φ is the state-flow replay row, Ω is the observation-flow replay row, and G is the innovation replay row;*
6. *E records estimator readback through the sibling* KalmanFilter$^{\uparrow}$ *and* LQR$^{\uparrow}$ *consumer surfaces without identifying the continuous-time packet with either one;*
7. *T records componentwise* hsame *transport for state-flow, observation-flow, covariance, gain, estimator, and replay rows;*
8. *H records displayed* Cont *replay for finite time slices and accepted row concatenation;*

9. *Pkg is the* Pkg *provenance naming* $\mathsf{KalmanBucyFilter}^{\uparrow}$, $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, $\mathsf{ControlControllability}^{\uparrow}$, $\mathsf{Mat}^{\uparrow}$, $\mathsf{VecSpace}^{\uparrow}$, $\mathsf{DynSystem}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{KalmanBucyFilter}^{\uparrow}}$ *row.*

No host Brownian motion, selected sample path, analytic stochastic integral, convergence rate, or external observer object is part of the carrier.

Theorem 6360.2 (KalmanBucyFilter NameCert obligations)**.** *For the carrier of Definition 6360.1, the local* $\mathsf{NameCert}_{\mathsf{KalmanBucyFilter}^{\uparrow}}$ *surface consists exactly of state-flow admission through X, A, Φ, observation-flow admission through Y, C, Ω, covariance and Riccati-ledger admission through Q, R, P, gain admission through L, estimator replay through $\widehat{x}$ and G, sibling readback through the* $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, *and* $\mathsf{ControlControllability}^{\uparrow}$ *package rows, componentwise* hsame *transport through T, displayed* Cont *replay through H, package provenance through Pkg, and local naming through N.*

Proof. Project the accepted carrier of Definition 6360.1. The state-flow and observation-flow rows are displayed finite coordinates, so their reads pass through the same matrix, vector-space, dynamic-system, and observability certificates named by the package row.

The covariance and gain rows are the Riccati-style ledger face of the packet. They are consumed as finite BHist data: Q, R, P expose the covariance route, while L exposes the gain row read by the estimator replay.

The estimator replay rows $\widehat{x}, \Phi, \Omega, G$ are then transported by T and replayed by H. The sibling package rows identify exactly which $\mathsf{KalmanFilter}^{\uparrow}$, $\mathsf{LQR}^{\uparrow}$, $\mathsf{ObservabilityGramian}^{\uparrow}$, $\mathsf{ControlObservability}^{\uparrow}$, and $\mathsf{ControlControllability}^{\uparrow}$ surfaces are available to downstream readers, so no hidden stochastic process or analytic filtering object can be recovered from the certificate. □

Falsifiable prediction: An accepted $\mathsf{KalmanBucyFilter}^{\uparrow}$ packet is refuted by any estimator consumer that reads a gain or covariance row while bypassing the displayed Riccati ledger Q, R, P, the observation-flow row C, Ω, the estimator replay rows $\widehat{x}, \Phi, G$, or the package provenance naming the sibling Kalman, regulator, Gramian, observability, and controllability certificates.

Independence witness: kalman-filter, lqr, observability-gramian, controlobservability, control-controllability: none is bijective to $\mathsf{KalmanBucyFilter}^{\uparrow}$ because the continuous-time filter packet jointly carries state-flow, observation-flow, covariance/Riccati ledger, gain, estimator replay, transport, continuation replay, provenance, and local NameCert rows.

Closure status for $\mathsf{NameCert}_{\mathsf{KalmanBucyFilter}^{\uparrow}}$**.**

Constructive story: A $\mathsf{KalmanBucyFilter}^{\uparrow}$ packet is generated from finite state-flow, observation-flow, covariance, Riccati, gain, estimator replay, transport, continuation, provenance, and local naming rows.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{KalmanBucyFilter}^{\uparrow}}$, seed)

Scope closed: Definition 6360.1, Definition 6360.2, the displayed falsifiable prediction, and the displayed independence witness seed the finite continuous-time filtering NameCert surface over Kalman, regulator, Gramian, observability, controllability, matrix, vector-space, and dynamic-system rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{KalmanBucyFilter}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not claim Lean verification, bridge checking, stochastic integral semantics, analytic convergence, infinite-horizon optimality, or host continuous-time probability semantics.

Upgrade path: Obligation closure requires checked carrier admission, state-flow and observation-flow replay, covariance and Riccati-ledger exactness, gain admission, estimator replay, componentwise transport, continuation replay, package provenance, and local NameCert exhaustion for BEDC.Derived.KalmanBucyFilterUp.

Chapter 6361

A Concrete Naming Certificate for $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ packet records the finite modulus object used by constructive Cauchy convergence arguments before any Real-facing seal is consumed. It binds the dyadic tolerance rows of chapter 291, the finite stream windows of chapter 126, the regular readback rows of chapter 301, the terminal seal boundary of chapter 38, and the monotone convergence ledger of chapter 2074. The packet is a NameCert reconstruction of an effective Cauchy-modulus surface; it is not a Real completeness theorem, not a quotient of streams, and not a source of selected limits.

Definition 6361.1 (Effective Cauchy convergence modulus carrier)**.** *An* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (D, W, R, S, M, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row naming the dyadic precision request and the comparison budget used by the modulus read.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the bounded observations inspected at that precision.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row attached to D and W.*
4. *S is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, available only after D, W, R have been displayed.*
5. *M is the* $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *monotone ledger row, recording that later dyadic requests read weakly later finite windows without shortening the accepted tail.*
6. *H records componentwise* hsame *transport for the tolerance, window, readback, seal, modulus ledger, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from an effective Cauchy-modulus request through D, W, R, S, M, H.*
8. *P is the* Pkg *provenance over* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealModulusOfConvergence}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, R, S, M, H, C, P, N$. It has no coordinate for quotient stream equality, host real equality, selected representatives, selected limits, countable choice, metric completeness, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6361.2 (Effective Cauchy modulus carrier admission)**.** *Every accepted* $\mathsf{EffectiveCauchyConvergenceModul}$ *packet exposes exactly the rows $D, W, R, S, M, H, C, P, N$ of Definition 6361.1 before any Real or Completion consumer may use its convergence claim. The admitted surface contains dyadic tolerances, finite stream windows, regular readback, a terminal Real seal, a monotone convergence ledger, componentwise transport, continuation replay, package provenance, and local naming, and nothing else.*

Proof. Project the finite carrier. The rows D, W, R, S, M are the only analytic rows, while H, C, P, N transport, replay, source, and name that same route.

Since the classifier compares packets only through these coordinates, a consumer cannot read a hidden quotient stream, host equality, selected representative, selected limit, choice schedule, metric-completeness row, or ambient completeness row from the carrier. □

Theorem 6361.3 (Effective Cauchy monotone tail-window stability)**.** *Let $E = (D, W, R, S, M, H, C, P, N)$ be an accepted* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ *packet. If a later dyadic tolerance request is transported through the monotone ledger M, then the accepted read moves only to a weakly later* $\mathsf{StreamName}^{\uparrow}$ *finite window, preserves the same* $\mathsf{RegSeqRat}^{\uparrow}$ *readback route, and reaches the same terminal* $\mathsf{Real}^{\uparrow}$ *seal boundary before the convergence claim is consumed.*

Proof. Read the tolerance request through D and the finite observation through W. The monotone ledger M is the only row allowed to compare an earlier tolerance request with a later one.

Apply the Real-modulus tail-stability route of Definition 2074.4. The later request must remain on a weakly later finite window and on the same regular readback path.

The seal row S is downstream of D, W, R, M. Thus the later-tail read cannot introduce a separate selected tail, quotient equality, countable-choice schedule, or ambient completeness witness. □

Theorem 6361.4 (Effective Cauchy RegSeqRat handoff)**.** *Every accepted* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ *consumer that asks for a regular Cauchy read must pass through*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}$$

before the $\mathsf{RealModulusOfConvergence}^{\uparrow}$ *ledger or the terminal* $\mathsf{Real}^{\uparrow}$ *seal can be used. The handoff exports no quotient stream equality, selected representative, host real equality, selected limit, or countable-choice schedule.*

Proof. Project the carrier rows $D, W, R, S, M, H, C, P, N$. The dyadic row D supplies the precision budget, the stream row W supplies only a finite observation window, and the regular readback row R is the first row that can present the data as a regular Cauchy request.

Use Definition 2074.5. The Real-modulus ledger consumes the same dyadic tolerance, finite window, and regular readback surface before reaching a Real-facing seal.

Therefore any regular Cauchy consumer must read D, W, R first. The carrier contains no coordinate that could supply quotient stream equality, selected representatives, host equality, selected limits, or countable choice outside that route. □

Theorem 6361.5 (Effective Cauchy Real seal nonescape)**.** *For every accepted* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ *packet* $E = (D, W, R, S, M, H, C, P, N)$*, the terminal* $\mathsf{Real}^{\uparrow}$ *seal row* S *is reachable only after the dyadic tolerance row, finite stream window, regular readback row, and monotone convergence ledger have been displayed. The packet exports no Real completeness theorem, quotient stream equality, host real equality, selected limit, metric-completeness assertion, or countable-choice schedule.*

Proof. Carrier admission gives the displayed order of rows. The seal row S is not an independent coordinate: it is listed after the dyadic, window, and regular readback rows and is controlled by the monotone ledger.

Apply Definition 38.867. Any accepted Real terminal seal read on the L10 route factors through the displayed dyadic, stream, regular readback, and Real faces.

The structural rows H, C, P, N preserve that finite route. Since neither the carrier nor the Real L10 route contains a completeness coordinate, quotient stream, host equality row, selected limit, metric-completeness row, or choice schedule, the seal cannot export one. □

Theorem 6361.6 (Effective Cauchy Bishop interface consumer route)**.** *Every* $\mathsf{BishopRealLineInterface}^{\uparrow}$ *consumer of an accepted* $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ *packet must read*

$$\mathsf{DyadicRatCore}^{\uparrow}, \quad \mathsf{StreamName}^{\uparrow}, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}, \quad \mathsf{RealModulusOfConvergence}^{\uparrow}$$

before using the effective modulus as real-line interface data. The route reaches the Bishop interface only as a finite tolerance, finite-window, regular-readback, Real-seal, and monotone-ledger surface; it does not assert quotient equality, trichotomy, selected limits, countable choice, host real equality, metric completeness, or ambient Real completeness.

Proof. Start with the carrier admission theorem. It exposes the dyadic tolerance row, finite window row, regular readback row, Real seal row, and monotone modulus ledger before any structural row can be used by a downstream consumer.

Apply Definition 6361.4 and Definition 6361.5. These two routes force the regular readback and terminal seal to remain on the displayed L10 surface.

Now use the Bishop interface route of Definition 2062.11. A Bishop real-line consumer may compare convergence-modulus data only after the dyadic, stream, regular readback, Real seal, and Cauchy-equivalence boundary are visible, and it receives the downstream $\mathsf{RealModulusOfConvergence}^{\uparrow}$ ledger as finite evidence.

Combining the routes gives the stated consumer surface. The exclusion clauses are inherited from the effective carrier, the Real seal nonescape theorem, and the Bishop interface handoff. □

Closure status for $\mathsf{NameCert}_{\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}}$**.**

Constructive story: An $\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance row, finite StreamName windows, RegSeqRat readback, a terminal Real seal, a monotone RealModulusOfConvergence ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}}$, obligation)

Scope closed: Definition 6361.1, Definition 6361.2, Definition 6361.3, Definition 6361.4, Definition 6361.5, and Definition 6361.6 seed the effective Cauchy-modulus boundary over the displayed dyadic, stream, regular-readback, Real-seal, monotone-ledger, transport, continuation, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{EffectiveCauchyConvergenceModulus}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close Real completeness, quotient stream equality, host real equality, trichotomy, countable choice, selected representatives, selected limits, metric completeness, or bridge checking.

Upgrade path: Scoped closure requires binding the listed obligations to DyadicRatCore, StreamName, RegSeqRat, RealUp, RealModulusOfConvergence, BHist, hsame, Cont, Pkg, and NameCert rows for BEDC.Derived.EffectiveCauchyConvergenceModulusU

Chapter 6362

A Concrete Naming Certificate for $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$

Following Bishop–Bridges on regular Cauchy reals and constructive algebra, the $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ packet records the finite modulus surface that lets algebraic operations on regular Cauchy representatives be replayed as named BEDC data. It sits over the dyadic tolerance ledger of chapter 291, the finite StreamName windows of chapter 126, the regular rational sequence rows of chapter 301, and the terminal Real seal rows of chapter 38. The packet is an L10 completion-facing algebra ledger: it packages sum and product stability for displayed representatives, not Real completeness, quotient equality of real names, open-cover compactness, or host real algebra.

Definition 6362.1 (RegularCauchyModulusAlgebra carrier)**.** *A* $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (D, W, Q, E, S, M, U, V, H, T, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *modulus ledger, exposing dyadic tolerances, comparison rows, and arithmetic tolerance requests used by every operation row;*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row, giving the finite input windows inspected before a tolerance request is answered;*
3. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *sequence row, carrying the regular rational representatives, their window readback, and the local regularity witnesses consumed by the operation rows;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached after the regular rational readback is replayed through the displayed modulus rows;*
5. *S is the sum-stability row, recording a dyadic output tolerance, the two displayed input windows selected for that tolerance, the regular rational sum readback, and the resulting Real seal handoff;*
6. *M is the product-stability row, recording the bounded window data, dyadic tolerance split, rational product readback, and Real seal handoff required for multiplication of displayed regular representatives;*
7. *U is the transport row, recording componentwise* hsame *for D, W, Q, E, S, M and for the replay, provenance, and naming rows;*

8. *V is the replay row, recording displayed* Cont *routes from dyadic tolerance requests through StreamName windows, RegSeqRat readback, sum stability, product stability, and Real seal rows;*
9. *H is the operation-history row, collecting the finite* BHist *traces for the input representatives, selected windows, rational operation rows, and output seals;*
10. *T is the provenance row, a* Pkg *ledger over* $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*
11. *P is the local operation package row, exposing the accepted sum and product packets as downstream consumable BEDC objects;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}}$ *row.*

The carrier has no coordinate for Real completeness, quotient equality of Real names, open-cover compactness, a selected Cauchy limit, countable choice, or host real algebra.

Lean4 ✓ `BEDC.Derived.RegularCauchyModulusAlgebraUp.RegularCauchyModulusAlgebraTasteGate single carrier alignment`

Theorem 6362.2 (RegularCauchyModulusAlgebra NameCert obligations)**.** *For every accepted* $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ *carrier*

$$A = (D, W, Q, E, S, M, U, V, H, T, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}}$ *surface consists exactly of carrier admission for A, dyadic modulus admission through D, finite StreamName window admission through W, RegSeqRat sequence admission through Q, Real seal admission through E, sum-stability admission through S, product-stability admission through M, componentwise* hsame *transport through U, displayed* Cont *replay through V, operation history through H,* Pkg *provenance through T, operation package export through P, and local naming through N.*

Consequently a downstream algebra consumer may use the packet only after reading the finite dyadic modulus rows, StreamName windows, RegSeqRat sequence rows, Real seal rows, sum and product operation-stability rows, transport, replay, provenance, and local NameCert data displayed in the carrier.

Proof. Project the accepted carrier of Definition 6362.1. The rows $D, W, Q, E, S, M, U, V, H, T, P, N$ are the only coordinates exposed by the packet.

The analytic part is finite. Dyadic tolerance requests enter through D, are answered by StreamName windows in W, are read as regular rational representatives through Q, and reach the Real-facing seal row E only after those displayed reads have occurred.

The operation rows preserve the same discipline. The sum row S records its selected input windows and rational sum readback before exporting an output seal, while the product row M records bounded window data, tolerance split, and rational product readback before exporting its seal.

The structural rows do not add external algebraic content. Componentwise hsame transport in U, displayed Cont replay in V, operation histories in H, Pkg provenance in T, operation package export in P, and the local naming row N package the same finite surface. Since no other carrier coordinate is present, Real completeness, quotient equality, open-cover compactness, selected limits, choice, and host real algebra remain outside the NameCert obligations. □

Theorem 6362.3 (RegularCauchyModulusAlgebra operation replay)**.** *Every accepted* RegularCauchyModulusAlgeb *carrier*

$$A = (D, W, Q, E, S, M, U, V, H, T, P, N)$$

exports sum and product data only through the finite route

$$D, W, Q \longmapsto S, M \longmapsto E \longmapsto U, V, H, T, P, N.$$

Thus operation stability is a replayable BEDC obligation over dyadic modulus rows, StreamName windows, RegSeqRat sequence rows, Real seal rows, transport, replay, provenance, and local NameCert data. It is not a claim of Real completeness, quotient equality, open-cover compactness, or host real algebra.

Proof. Read the carrier rows in operation order. The dyadic row D states the output precision request, W selects the finite input windows, and Q supplies the regular rational representatives visible in those windows.

For addition, the row S records the selected windows and rational sum readback before the output seal can be read through E. For multiplication, the row M records the bounded window data, tolerance split, and rational product readback before its output seal can be read through E.

The rows U, V, H, T, P, N then transport, replay, remember, package, export, and name the same operation traces. Because the route never opens a completeness theorem, quotient equality relation, open-cover argument, or host algebra object, the exported stability remains finite BEDC replay data. □

Falsifiable prediction: A consumer that claims $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ operation stability while omitting the dyadic modulus row, StreamName window row, RegSeqRat sequence row, Real seal row, sum-stability row, product-stability row, hsame transport, Cont replay, Pkg provenance, operation package export, or NameCert row refutes the carrier of Definition 6362.1.

Independence witness: dyadicratcore, streamname, regseqrat, real: $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ binds dyadic modulus requests, finite stream windows, regular rational readback, Real seal handoff, and explicit sum/product stability in one local NameCert packet; each named sibling lacks at least one of these operation-stability rows as a local algebra coordinate.

Closure status for $\mathsf{NameCert}_{\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}}$.

Constructive story: A $\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}$ packet is generated from dyadic modulus rows, StreamName windows, RegSeqRat sequence rows, Real seal rows, sum and product stability rows, componentwise hsame transport, displayed Cont replay, operation histories, Pkg provenance, operation package export, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6362.1, Definition 6362.2, and Definition 6362.3 seed the finite modulus algebra packet for regular Cauchy representative sum and product stability.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{RegularCauchyModulusAlgebra}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim Real completeness, quotient equality of Real names, open-cover compactness, selected Cauchy limits, countable choice, or host real algebra.

Upgrade path: Obligation closure requires checked carrier admission, sum-stability replay, product-stability replay, Real seal handoff, and a Lean realization for BEDC.Derived.RegularCauchyModulusAlgebraUp.

Chapter 6363

A Concrete Naming Certificate for $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$

Following Heine, Borel, and Bishop–Bridges constructive analysis, the $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ packet records the finite-subcover boundary available for a displayed located real interval after the cover has already been reduced to finite dyadic interval data. It sits beside the compact interval route of chapter 3986, the dyadic interval-cover packet of chapter 1428, the finite closed-bounded interval compactness surface of chapter 3969, and the L10 Real and Completion route through $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$. The packet names a constructive finite interval subcover interface; it does not claim general open-cover compactness, a choice-selected subcover, quotient real equality, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6363.1 (Heine–Borel finite-interval carrier)**.** *A* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (I, D, B, C, S, Q, T, E, H, R, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{LocatedRealInterval}^{\uparrow}$ *source row, exposing lower and upper endpoint names, locatedness data, and interval-radius requests;*
2. *D is the* $\mathsf{DyadicIntervalCover}^{\uparrow}$ *finite cover row, recording dyadic cells, mesh scale, membership checks, and cover-read witnesses;*
3. *B is the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *compact-interval boundary row used only as the finite constructive interval route of chapter 3986;*
4. *C is the* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *row, supplying the closed-bounded interval compactness handoff after finite cover data is displayed;*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row attached to endpoint and cover observations;*
6. *Q is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row for the same observed windows;*
7. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance and mesh-refinement ledger;*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the interval, cover, mesh, window, and readback rows have been projected;*
9. *H records componentwise* hsame *transport for all displayed rows;*

10. *R records displayed* Cont *replay for interval admission, cover readback, dyadic refinement, finite subcover witness, and Real-seal handoff;*
11. *P is the* Pkg *provenance over* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$, $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
12. *N is the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteInterval}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for arbitrary open covers, countable choice, selected host representatives, quotient equality of real names, or ambient real completeness.

Theorem 6363.2 (Heine–Borel finite-interval NameCert obligations)**.** *For every accepted* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ *carrier $K = (I, D, B, C, S, Q, T, E, H, R, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteInterval}^{\uparrow}}$ *surface consists exactly of five finite rows: interval-source admission through I; finite cover admission through D; dyadic mesh refinement through T with the* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, Q; finite subcover witness readback through the* $\mathsf{BishopCompactInterval}^{\uparrow}$ *and* $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$ *rows B, C; and Real-seal nonescape through E, H, R, P, N. No row exports general open-cover compactness, a choice-selected subcover, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The source interval row is exactly I, so endpoint, radius, locatedness, and containment requests begin with the displayed $\mathsf{LocatedRealInterval}^{\uparrow}$ packet rather than with a host interval.

The finite cover and mesh rows are D and T. The cover cells, mesh scale, and membership checks are therefore read through $\mathsf{DyadicIntervalCover}^{\uparrow}$ and $\mathsf{DyadicRatCore}^{\uparrow}$, with S, Q supplying only the finite window and regular-rational readback needed to inspect the same observations.

The finite subcover witness is obtained only after the compact-interval boundary rows B, C have been displayed. These rows are the constructive compact interval surfaces cited above; they do not add an arbitrary open cover or a choice functional.

Finally read the seal and structural rows. The terminal $\mathsf{Real}^{\uparrow}$ row E is reached only after I, D, B, C, S, Q, T have been projected, and H, R, P, N transport, replay, provenance, and name that same finite packet. Since the carrier has no coordinate for the excluded principles, the five listed rows exhaust the local NameCert obligations. □

Theorem 6363.3 (Heine–Borel finite-interval dyadic mesh obligation)**.** *For every accepted* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ *carrier $K = (I, D, B, C, S, Q, T, E, H, R, P, N)$, the interval-cover mesh obligation is read only through the displayed located interval endpoints in I, the finite dyadic cover cells in D, and the* $\mathsf{DyadicRatCore}^{\uparrow}$ *mesh ledger T. The* $\mathsf{StreamName}^{\uparrow}$ *and* $\mathsf{RegSeqRat}^{\uparrow}$ *rows S, Q may report the endpoint windows used by the mesh, but the obligation exports no arbitrary open-cover refinement, no host interval cover, no quotient real equality, and no ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project I first. The closed interval endpoints are available only as displayed located real names, so every endpoint read begins inside the $\mathsf{LocatedRealInterval}^{\uparrow}$ row rather than in a host interval object.

Next project D and T. The cover cells, mesh scale, and dyadic comparisons are finite rows, and the mesh refinement is certified by the $\mathsf{DyadicRatCore}^{\uparrow}$ ledger attached to those rows.

Finally use S, Q only as the observation route for the same endpoint and cell windows. Because the carrier contains no coordinate for an arbitrary open cover or quotient real equality, the mesh obligation remains a finite dyadic-cover obligation over the displayed interval. □

Theorem 6363.4 (Heine–Borel finite-interval subcover witness obligation)**.** *For every accepted* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ *carrier* $K = (I, D, B, C, S, Q, T, E, H, R, P, N)$*, a finite subcover witness may be consumed only after the finite cover row* D*, the compact-interval handoff rows* B, C*, and the displayed witness rows carried through* $\mathsf{ProbeBundle}$*,* $\mathsf{CompactMetric}^{\uparrow}$*, and* $\mathsf{TotallyBounded}^{\uparrow}$*-style finite data have been exposed by the same* BHist *packet. The witness is a finite row selection inside the packet, not a countable-choice selector, not an arbitrary open-cover subcover, and not a host compactness theorem.*

Proof. Start with the finite cover row D. It lists the cover cells and membership checks already reduced to finite dyadic interval data, so the witness has a finite source to inspect.

Read the compact-interval handoff rows B, C next. These rows provide the constructive compact and closed-bounded interval surfaces used by the carrier, and they keep the witness attached to the same interval packet.

The probe, compact-metric, and totally-bounded reads are therefore finite rows inside the displayed BHist. The structural rows H, R, P, N transport, replay, source, and name that finite witness, while the carrier still lacks any coordinate for countable choice or arbitrary open-cover compactness. □

Theorem 6363.5 (Heine–Borel finite-interval Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing read of an accepted* $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ *carrier must display* $I, D, B, C, S, Q, T, H, R, P, N$ *before it reads the terminal seal row* E*. The seal records that the finite interval subcover route is Real-facing, but it does not claim arbitrary subset Heine–Borel compactness, general open-cover compactness, choice-selected subcovers, quotient real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the carrier in the order of Definition 6363.1. The interval, cover, compact handoff, window, readback, and dyadic mesh rows are all displayed before E is available.

The transport and replay rows preserve that same finite route. They allow the interval-cover and subcover-witness rows to be moved componentwise, but they do not create an arbitrary cover family or a selected subcover functional.

Finally inspect the provenance and local certificate. The Pkg and $\mathsf{NameCert}$ rows name only the displayed finite interval packet, so the terminal $\mathsf{Real}^{\uparrow}$ seal is a nonescape boundary rather than a completeness or open-cover compactness principle. □

Closure status for $\mathsf{NameCert}_{\mathsf{HeineBorelFiniteInterval}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteInterval}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6363.1, Definition 6363.2, Definition 6363.3, Definition 6363.4, and Definition 6363.5 seed the finite interval subcover boundary over $\mathsf{LocatedRealInterval}^{\uparrow}$, $\mathsf{DyadicIntervalCover}^{\uparrow}$, $\mathsf{BishopCompactInterval}^{\uparrow}$, $\mathsf{ClosedBoundedIntervalCompact}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{HeineBorelFiniteInterval}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim general open-cover compactness, arbitrary Heine–Borel compactness for subsets of $\mathsf{Real}^{\uparrow}$, choice-selected subcovers, quotient real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked interval-source admission, finite cover admission, dyadic mesh refinement, finite subcover witness readback, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.HeineBorelFiniteIntervalUp.

Constructive story: A $\mathsf{HeineBorelFiniteInterval}^{\uparrow}$ packet is generated from a LocatedRealInterval row, a DyadicIntervalCover finite cover row, BishopCompactInterval and ClosedBoundedIntervalCompact handoff rows, StreamName windows, RegSeqRat readback, a DyadicRatCore mesh ledger, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6364

A Concrete Naming Certificate for BishopLocatedIntervalOscillation$^{\uparrow}$

Following Bishop and Bridges on constructive real analysis, the BishopLocatedIntervalOscillation$^{\uparrow}$ packet records finite located-interval oscillation data for the L10 Real and Completion route. It sits over the located interval surface of chapter 394, the graph-read surface of chapter 130, the finite window route of chapter 126, the regular rational handoff of chapter 301, the dyadic tolerance core of chapter 291, and the terminal real-name seal of chapter 38. The packet names noncompact located-interval oscillation rows; it does not claim open-cover compactness, selected extrema, quotient function equality, or ambient Real$^{\uparrow}$ completeness.

Definition 6364.1 (Bishop located interval oscillation carrier)**.** *A* BishopLocatedIntervalOscillation$^{\uparrow}$ *carrier is a finite* BHist *packet*

$$O = (I, F, W, R, D, B, S, H, C, P, N)$$

with the following displayed rows:

1. *I is the* LocatedInterval$^{\uparrow}$ *source row, read as a located interval rather than as a compact interval object.*
2. *F is the* ContinuousMap$^{\uparrow}$ *graph row whose values are inspected only through displayed finite windows.*
3. *W is the* StreamName$^{\uparrow}$ *finite observation row for source and target probes.*
4. *R is the* RegSeqRat$^{\uparrow}$ *readback row attached to the same finite windows.*
5. *D is the* DyadicRatCore$^{\uparrow}$ *tolerance row, carrying the interval radius, value tolerance, and visible comparison budget.*
6. *B is the oscillation-bound row recording the accepted dyadic upper bound for the displayed interval probe.*
7. *S is the terminal* Real$^{\uparrow}$ *seal consumed only after I, F, W, R, D, B have been displayed.*
8. *H records componentwise* hsame *transport for the analytic rows and local naming rows.*
9. *C records displayed* Cont *replay through the finite route I, F, W, R, D, B, S, H.*
10. *P is the* Pkg *provenance over* BishopLocatedIntervalOscillation$^{\uparrow}$, LocatedInterval$^{\uparrow}$, ContinuousMap$^{\uparrow}$, StreamName$^{\uparrow}$, RegSeqRat$^{\uparrow}$, DyadicRatCore$^{\uparrow}$, Real$^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*

11. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for compactness, selected suprema, host real equality, quotient stream equality, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6364.2 (Bishop located interval oscillation NameCert obligations)**.** *For every accepted* $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$ *carrier* $O = (I, F, W, R, D, B, S, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalO}}$ *surface is exhausted by located-interval admission through* I*, continuous graph exposure through* F*, finite StreamName window exposure through* W*, RegSeqRat readback through* R*, DyadicRatCore tolerance exactness through* D*, oscillation-bound exposure through* B*, Real sealing through* S*, componentwise transport through* H*, displayed replay through* C*, provenance through* P*, and local naming through* N*.*

Proof. Project the carrier of Definition 6364.1. The source row is the located interval I, and the function row is the displayed continuous graph F.

The finite readback route is then forced. A probe opens the StreamName windows W, reads them through the RegSeqRat row R, applies the dyadic tolerance row D, and records the oscillation bound in B before the Real seal S can be consumed.

The structural rows H, C, P, N transport, replay, source, and name only that displayed route. Since the carrier has no compactness, selected-extremum, quotient, choice, host-equality, or completeness coordinate, no such object can become a local NameCert obligation. □

Theorem 6364.3 (Bishop located interval oscillation L10 route)**.** *Every L10 Real and Completion consumer of an accepted* $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$ *packet receives a noncompact located-interval oscillation surface through*

$$\mathsf{LocatedInterval}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{Real}^{\uparrow} \longmapsto H, C, P, N.$$

The route displays finite interval probes, continuous graph observations, regular rational readback, dyadic tolerances, and a terminal Real seal. It exports no open-cover compactness, selected maximum, quotient stream equality, selected limit, countable choice, or ambient completeness theorem.

Proof. Start with the NameCert obligations. They require the consumer to read I, F, W, R, D, B, S before transport, replay, provenance, or local naming can be used.

The route is noncompact because I is only a located interval source. The oscillation row B records a displayed dyadic bound for the accepted probe, not a finite-subcover or extremum principle.

Finally the Real-facing endpoint is the seal row S behind the same finite route. Thus an L10 consumer receives a certificate-local oscillation surface and none of the excluded compactness, quotient, selection, choice, host-equality, or completeness data. □

Falsifiable prediction: Every accepted $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$ consumer that reads an oscillation bound while omitting one of the LocatedInterval, ContinuousMap, StreamName, RegSeqRat, DyadicRatCore, Real seal, hsame, Cont, Pkg, or NameCert rows has left the packet.

Independence witness: locatedinterval, continuousmap, streamname, regseqrat, dyadicratcore, real: $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$ is not bijective with any listed sibling because it joins a noncompact located interval source, continuous graph observations, finite windows, regular rational readback, dyadic tolerances, oscillation bounds, Real sealing, transport, replay, provenance, and local naming as one finite packet.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}$ packet is generated from a LocatedInterval source row, a ContinuousMap graph row, StreamName windows, RegSeqRat readback, DyadicRatCore tolerance, an oscillation-bound row, a RealUp seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6364.1, Definition 6364.2, and Definition 6364.3 seed the finite noncompact located-interval oscillation surface for L10 Real and Completion consumers.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopLocatedIntervalOscillation}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, selected extrema, quotient function equality, quotient stream equality, selected limits, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, located-interval admission, continuous graph exposure, StreamName window replay, RegSeqRat readback, DyadicRatCore tolerance exactness, oscillation-bound exposure, Real sealing, transport, replay, provenance, and local naming for BEDC.Derived.BishopLocatedIntervalOscillationUp.

Chapter 6365

A Concrete Naming Certificate for $\mathsf{LocatedDyadicCompletion}^{\uparrow}$

Following Bishop–Bridges constructive completion practice, the $\mathsf{LocatedDyadicCompletion}^{\uparrow}$ packet records the finite BEDC route from located dyadic tolerance data to regular rational streams and the terminal real-name seal. It sits over the dyadic tolerance core of chapter 291, the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, and the real-name boundary of chapter 38. The packet names a displayed completion interface for the current L10 Real and Completion phase; it does not import quotient streams, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6365.1 (Located dyadic completion carrier)**.** *A* $\mathsf{LocatedDyadicCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, S, R, L, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording dyadic radii, common refinements, and positive located gaps;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by each tolerance request;*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row reached after the stream windows have been displayed;*
4. *L is the located split row, answering each dyadic comparison request before any real seal is exposed;*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal, read only after D, S, R, L have been projected;*
6. *H records componentwise* hsame *transport for the displayed dyadic, stream, regular, locatedness, and real rows;*
7. *C records displayed* Cont *replay for tolerance admission, finite-window readback, regular-sequence handoff, located split, and real sealing;*
8. *P is the* Pkg *provenance over* $\mathsf{LocatedDyadicCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*
9. *N is the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, S, R, L, E, H, C, P, N$. It has no coordinate for quotient stream equality, a selected completion limit, countable choice, detached order trichotomy, or ambient real completeness.

Theorem 6365.2 (Located dyadic completion NameCert obligations)**.** *For every accepted* $\mathsf{LocatedDyadicCompletio}$ *carrier* $K = (D, S, R, L, E, H, C, P, N)$*, the local* $\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$ *surface consists exactly of five finite rows: dyadic tolerance admission through* D*; finite stream-window inspection through* S*; regular rational readback through* R*; located comparison discharge through* L*; and real-seal nonescape through* E, H, C, P, N*. No obligation row exports quotient streams, selected limits, countable choice, detached trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The dyadic tolerance ledger is exactly D, so every radius, refinement, and positive-gap request begins inside $\mathsf{DyadicRatCore}^{\uparrow}$.

The finite observation route is S followed by R. Thus a completion-facing consumer reads only displayed stream windows and their $\mathsf{RegSeqRat}^{\uparrow}$ readback before it can ask for a real boundary.

The located split row is L. It answers the comparison request associated with the displayed dyadic budget, but it is not a detached trichotomy principle and it does not select a global representative.

Finally inspect E, H, C, P, N. The terminal $\mathsf{Real}^{\uparrow}$ seal is reached only after the four rows D, S, R, L have been projected, while the structural rows transport, replay, source, and name that same finite packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6365.3 (Located dyadic completion Cauchy handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *consumer of an accepted* $\mathsf{LocatedDyadicCompletion}^{\uparrow}$ *carrier factors through the route*

$$\mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{StreamName}^{\uparrow},\ \mathsf{RegSeqRat}^{\uparrow},\ \mathsf{Real}^{\uparrow}.$$

In particular, the Cauchy handoff from dyadic tolerance data to a real-name boundary is mediated by displayed finite windows and regular rational readback, not by quotienting Cauchy streams or selecting a limit object.

Proof. Start with the accepted carrier $K = (D, S, R, L, E, H, C, P, N)$. A regular rational consumer can inspect the readback row only after D supplies the tolerance ledger and S supplies the finite stream windows.

A real-name consumer reaches E only after R and L have been read. The located row L supplies the comparison response attached to the same dyadic budget, so the handoff carries the Cauchy information already displayed by D, S, R.

The structural rows H, C, P, N preserve this order under transport, replay, provenance, and local naming. Hence the only exported route is the displayed dyadic-to-stream-to-regular-to-real chain, with no quotient stream or selected limit coordinate available. □

Theorem 6365.4 (Located dyadic completion nonescape)**.** *No accepted* $\mathsf{LocatedDyadicCompletion}^{\uparrow}$ *carrier can be used as evidence for quotient stream equality, countable choice, a choice-selected completion limit, detached trichotomy of real names, or ambient completeness of* $\mathsf{Real}^{\uparrow}$*. Any downstream claim using the carrier must consume the displayed rows* $D, S, R, L, E, H, C, P, N$*.*

Proof. The carrier definition lists every exported coordinate. The computational rows are D, S, R, L, E, and the structural rows are H, C, P, N.

Quotient stream equality and selected limits would require a coordinate identifying arbitrary Cauchy names or choosing a representative beyond the displayed stream windows. No such coordinate appears in the carrier.

Countable choice, detached trichotomy, and ambient real completeness would require a global principle independent of the finite dyadic ledger. The carrier instead routes comparison through L and real sealing through E after D, S, R have already been displayed.

Therefore any downstream use must consume the listed rows, and the excluded principles cannot escape from the local $\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$ packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$, seed)

Scope closed: Definition 6365.1, Definition 6365.2, Definition 6365.3, and Definition 6365.4 seed the finite located dyadic completion route over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{LocatedDyadicCompletion}^{\uparrow}}$, formalTarget)

Lean target: `BEDC.Derived.LocatedDyadicCompletionUp`

Bridge status: none

Not claimed: This chapter does not close quotient stream equality, selected limits, countable choice, detached trichotomy, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite stream-window inspection, regular rational readback, located comparison discharge, real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.LocatedDyadicCompletionUp.

Constructive story: A $\mathsf{LocatedDyadicCompletion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, RegSeqRat readback, a located comparison row, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Lean4 (statement) `BEDC.Derived.LocatedDyadicCompletionUp`

Chapter 6366

Sequentially Located Real NameCert Construction

Following Bishop and Bridges on constructive real sequences and locatedness, the $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ packet records a sequence-indexed family of $\mathsf{Real}^{\uparrow}$ names whose finite observations carry an explicit locatedness witness at each inspected unary index. It sits after the $\mathsf{StreamName}^{\uparrow}$ window surface of chapter 126, the $\mathsf{RegSeqRat}^{\uparrow}$ readback surface of chapter 301, the terminal $\mathsf{Real}^{\uparrow}$ seal of chapter 38, and the located real sequence packet of chapter 1984. The packet is a choice-free sequential locatedness carrier: every visible coordinate is finite, named, and ledgered before any downstream Real or Completion consumer may read it.

Definition 6366.1 (Sequentially located real carrier)**.** *A* $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$X = (I, S, E, L, R, Q, H, C, P, N)$$

with the following displayed rows:

1. *I is the unary index-window row, naming the finite initial segment of sequence indices that a consumer may inspect.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *observation row, displaying the finite stream windows attached to the indices in I.*
3. *E is the pointwise* $\mathsf{Real}^{\uparrow}$ *name row, recording the real-name packet exposed at each inspected index.*
4. *L is the locatedness-witness row, assigning to each unary index in I the finite locatedness evidence used to compare the corresponding real-name observation with rational bounds.*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the displayed stream windows, pointwise real names, and locatedness witnesses.*
6. *Q is the finite request ledger, recording which indices, stream windows, rational bounds, and locatedness witnesses were exposed.*
7. *H records componentwise* hsame *transport for index windows, stream observations, pointwise real names, locatedness witnesses, regular rational readback, ledger rows, replay, provenance, and local naming.*
8. *C records displayed* Cont *replay from an index request through $I, S, E, L, R, Q, H, P, N$.*

9. *P is the* Pkg *provenance over* $\mathsf{SequentiallyLocatedReal}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{SequentiallyLocatedReal}^{\uparrow}}$ *row.*

The carrier exposes sequence-indexed $\mathsf{Real}^{\uparrow}$ *packets only through finite* $\mathsf{StreamName}^{\uparrow}$ *observations and a locatedness witness for each inspected unary index.*

Theorem 6366.2 (Sequentially located real NameCert obligations)**.** *For every accepted* $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ *carrier*

$$X = (I, S, E, L, R, Q, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{SequentiallyLocatedReal}^{\uparrow}}$ *surface consists of five finite obligation rows:*

1. *carrier admission: the accepted object is exactly the displayed* BHist *packet X with unary index windows, finite stream observations, pointwise real-name rows, locatedness witnesses, regular rational readback, request ledger, transport, replay, provenance, and local naming rows;*
2. *pointwise classifier stability: classified replacement of an inspected index, stream window, real-name row, or locatedness witness is transported componentwise through H and replayed only along the displayed route in C;*
3. *locatedness transport: the witness row L moves with its unary index in I and with its pointwise* $\mathsf{Real}^{\uparrow}$ *row in E, so no downstream read may detach a locatedness decision from the coordinate that generated it;*
4. $\mathsf{RegSeqRat}^{\uparrow}$ *readback: the row R is readable only after the corresponding I, S, E, L rows have been displayed, matching the regular rational handoff of chapter 301;*
5. *ledger nonescape: the request ledger Q records every exposed index, stream observation, rational bound, and locatedness witness before the packet reaches P and N, and the packet exports no quotient stream, countable-choice selector, selected limit, ambient* $\mathsf{Real}^{\uparrow}$ *completeness theorem, or unchecked bridge data.*

These rows are the whole sequential locatedness surface consumed by the sibling sequence route of chapter 1984.

Falsifiable prediction: Every accepted $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ read displays a finite unary index window, finite $\mathsf{StreamName}^{\uparrow}$ observations, pointwise $\mathsf{Real}^{\uparrow}$ names, locatedness witnesses, one $\mathsf{RegSeqRat}^{\uparrow}$ readback row, a request ledger, and local transport, replay, provenance, and naming rows before any Real or Completion consumer reads the packet.

Independence witness: streamname, regseqrat, real, located-real-sequence: $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ is not bijective to any listed sibling because it binds sequence-indexed $\mathsf{Real}^{\uparrow}$ names to per-index locatedness witnesses and a request ledger before the regular rational readback.

Closure status for $\mathsf{NameCert}_{\mathsf{SequentiallyLocatedReal}^{\uparrow}}$.

Constructive story: A $\mathsf{SequentiallyLocatedReal}^{\uparrow}$ packet is generated from unary index windows, finite StreamName observations, pointwise Real names, per-index locatedness witnesses, RegSeqRat readback, a request ledger, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{SequentiallyLocatedReal}^{\uparrow}}$, seed)

Scope closed: Definition 6366.1 and Definition 6366.2 seed the choice-free sequential locatedness carrier over StreamName, RegSeqRat, Real, and located-real-sequence consumers.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{SequentiallyLocatedReal}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not assert Real completeness, any choice principle, a selected limit, quotient stream equality, Lean verification, or checked bridge data.

Upgrade path: Lean closure requires checked carrier admission, pointwise classifier stability, locatedness transport, RegSeqRat readback, and ledger nonescape for BEDC.Derived.SequentiallyLocatedRealUp.

Chapter 6367

A Concrete Naming Certificate for $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$ packet records a constructive upper/lower rational enclosure for a real name on the L10 Real and Completion path. It sits between the rational tolerance service of chapter 291, the finite observation schedules of chapter 126, the regular rational readback route of chapter 301, the located-name boundary of chapter 1933, and the terminal real-name seal of chapter 38. The packet exposes interval enclosure data as finite BEDC rows; it does not identify quotient real classes, select a completed limit, invoke trichotomy, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6367.1 (Upper/lower real enclosure carrier)**.** *An* $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$E = (L, U, W, S, R, Z, H, C, P, N)$$

with the following displayed rows:

1. *L is the lower rational bound row, read through the rational and dyadic tolerance surface used by* $\mathsf{DyadicRatCore}^{\uparrow}$*;*
2. *U is the upper rational bound row for the same inspected real-name window;*
3. *W is the dyadic width row, displaying a tolerance certificate that $U - L$ is bounded by the named dyadic width and can be refined along the same finite schedule;*
4. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row naming exactly the finite observation windows used to inspect the enclosure;*
5. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row translating those observations into regular rational approximants;*
6. *Z is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the lower bound, upper bound, width, schedule, and readback rows have been displayed;*
7. *H records componentwise* hsame *transport for L, U, W, S, R, Z;*
8. *C records displayed* Cont *replay for bound inspection, width refinement, regular readback, located-name handoff, and Real sealing;*
9. *P is the* Pkg *provenance over* $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert*;*

10. *N is the local* $\mathsf{NameCert}_{\mathsf{UpperLowerRealEnclosure}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these rows. It has no coordinate for a host real representative, quotient equality of names, selected limits, trichotomy of real order, or unrestricted completeness.

Theorem 6367.2 (Upper/lower real enclosure NameCert obligations)**.** *For every accepted* $\mathsf{UpperLowerRealEnclosur}$ *carrier $E = (L, U, W, S, R, Z, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{UpperLowerRealEnclosure}^{\uparrow}}$ *surface consists exactly of five finite rows:*

1. *lower-bound admission, where L is displayed as a rational bound over the same* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger used by the inspected real-name window;*
2. *upper-bound admission, where U is displayed over the same ledger and is compared with L only through dyadic rational order data;*
3. *width control, where W records the finite dyadic enclosure width and its refinement schedule;*
4. *schedule/readback coherence, where S and R tie the bound rows to the same* $\mathsf{StreamName}^{\uparrow}$ *windows and* $\mathsf{RegSeqRat}^{\uparrow}$ *regular rational observations;*
5. *Real-seal nonescape, where Z, H, C, P, N transport, replay, source, and name the displayed enclosure without exporting host representatives, quotient equality, selected limits, trichotomy, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier. The first two rows are precisely the rational bound rows L and U, and both are read through the displayed dyadic tolerance ledger rather than through a host order on completed reals.

The width row W is part of the same finite packet. Its dyadic comparison is therefore a displayed tolerance certificate attached to L and U, not an external metric completeness assertion.

Next read the schedule and readback rows. The finite windows in S determine which observations may be inspected, and R translates only those observations into regular rational approximants. This keeps the enclosure on the L10 path through $\mathsf{StreamName}^{\uparrow}$ and $\mathsf{RegSeqRat}^{\uparrow}$.

Finally project the seal and structural rows. The $\mathsf{Real}^{\uparrow}$ row Z is reached only after L, U, W, S, R have been displayed, while H, C, P, N transport, replay, source, and name the same finite data. Since the carrier has no coordinate for the excluded principles, the five rows exhaust the local NameCert obligations. □

Theorem 6367.3 (Upper/lower real enclosure width shrink)**.** *Let $E = (L, U, W, S, R, Z, H, C, P, N)$ be an accepted* $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$ *carrier. Any refinement of the displayed* $\mathsf{StreamName}^{\uparrow}$ *schedule S that is replayed by C and read back by R may replace W by a smaller displayed dyadic width W' only through the same lower-bound row L, upper-bound row U,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{DyadicRatCore}^{\uparrow}$ *comparison ledger, and* $\mathsf{Real}^{\uparrow}$ *seal row Z. The shrink route exports a narrower finite enclosure row, but it does not export a selected limit point, quotient real equality, order trichotomy, or ambient completeness.*

Proof. Start with the carrier definition. The only width coordinate is W, and every permitted inspection of W is tied to the bound rows L, U, the finite schedule S, and the regular readback row R.

A refinement of S supplies a smaller finite observation window only when the replay row C displays that refinement inside the same packet. The readback row R then compares the resulting rational approximants through the dyadic ledger, so the proposed W' is a new displayed width row rather than a semantic limit.

The structural rows H, P, N keep the refined enclosure attached to the original packet, and the terminal seal Z is not reread as completeness. Therefore width shrink is a finite dyadic enclosure operation over the displayed rows and cannot yield any of the excluded host principles. □

Theorem 6367.4 (Upper/lower real enclosure Real-seal nonescape)**.** *Every* $\mathsf{Real}^{\uparrow}$*-facing consumer of an accepted* $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$ *packet must first display the lower rational bound* L*, upper rational bound* U*, dyadic width* W*,* $\mathsf{StreamName}^{\uparrow}$ *schedule* S*,* $\mathsf{RegSeqRat}^{\uparrow}$ *readback* R*, componentwise* hsame *transport* H*, displayed* Cont *replay* C*,* Pkg *provenance* P*, and local name* N *before reading the seal row* Z*. The seal may certify that the displayed enclosure is Real-facing, but it does not identify quotient real representatives, select a completed limit, prove arbitrary locatedness, invoke trichotomy, or supply ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Read the packet in the order forced by Definition 6367.1. The seal row Z is one coordinate of the carrier and is not available before the bound, width, schedule, and readback rows have been displayed.

The transport and replay rows keep the consumer on the same finite enclosure. They allow the listed rows to be moved and replayed componentwise, but they do not create a new representative of the real name or a completion operation.

Finally inspect the provenance and local certificate. The Pkg row names the dependency corridor through $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$, and the local $\mathsf{NameCert}$ row certifies only that corridor. Thus the Real seal is a nonescape boundary for the displayed enclosure packet. □

Closure status for $\mathsf{NameCert}_{\mathsf{UpperLowerRealEnclosure}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{UpperLowerRealEnclosure}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6367.1, Definition 6367.2, Definition 6367.3, and Definition 6367.4 seed the upper/lower rational enclosure surface over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{LocatedReal}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{UpperLowerRealEnclosure}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient real equality, selected completed limits, order trichotomy, arbitrary locatedness, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or a standard bridge.

Upgrade path: Obligation closure requires checked lower-bound admission, upper-bound admission, width control, schedule/readback coherence, Real-seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.UpperLowerRealEnclosu

Constructive story: An $\mathsf{UpperLowerRealEnclosure}^{\uparrow}$ packet is generated from lower and upper rational bound rows, a dyadic width row, StreamName windows, RegSeqRat readback, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6368

A Concrete Naming Certificate for $\mathsf{FrechetCompleteMetric}^{\uparrow}$

Following Frechet and Bishop on metric completion by Cauchy data, the $\mathsf{FrechetCompleteMetric}^{\uparrow}$ packet records the finite shared-tail route by which a metric Cauchy family exposes a completion-facing real seal. It sits after the finite stream-window surface of chapter 126, the regular rational readback of chapter 301, the terminal real-name boundary of chapter 38, the metric carrier of chapter 57, and the complete-metric interface of chapter 128. The packet names a finite shared-tail completion obligation; it does not construct a quotient completion, select a limit representative, assert countable choice, or claim ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6368.1 (Cauchy-modulus shared-tail carrier)**.** *A* $\mathsf{FrechetCompleteMetric}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$F = (M, S, R, D, T, E, H, C, P, N)$$

with the following displayed rows.

1. *M is the* $\mathsf{Met}^{\uparrow}$ *source row, naming the distance surface and the finite metric comparison ledger used by the Cauchy request.*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row exposing the inspected tail windows of the candidate Cauchy name.*
3. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to those windows.*
4. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *positive tolerance ledger consumed by the tail comparison.*
5. *T is the shared-tail modulus row, recording that every displayed tolerance request is answered by one finite tail window accepted by both the stream and regular-readback faces.*
6. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, visible only after M, S, R, D, T have been displayed.*
7. *H records componentwise* hsame *transport for the metric, stream, regular, tolerance, shared-tail, seal, replay, provenance, and naming rows.*
8. *C records displayed* Cont *replay from the metric source through the stream windows, regular readback, dyadic tolerance ledger, shared-tail row, Real seal, transport, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{FrechetCompleteMetric}^{\uparrow}$, $\mathsf{Met}^{\uparrow}$, $\mathsf{CompleteMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. N is the local $\mathsf{NameCert}_{\mathsf{FrechetCompleteMetric}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $M, S, R, D, T, E, H, C, P, N$. It has no coordinate for quotient completion equality, a selected completed point, an infinite choice schedule, host real equality, or an ambient completeness theorem.

Theorem 6368.2 (Cauchy-modulus shared-tail NameCert obligations)**.** *For every accepted* FrechetCompleteMetric *carrier $F = (M, S, R, D, T, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{FrechetCompleteMetric}^{\uparrow}}$ *surface consists exactly of carrier admission for the metric row, finite stream windows, regular rational readback, dyadic tolerance ledger, shared-tail modulus row, terminal Real seal, componentwise transport, displayed replay, provenance, and local naming.*

The classifier agreement obligation requires two accepted packets to agree precisely when these displayed rows agree componentwise through hsame*. The stability obligation requires* hsame *transport and* Cont *replay to preserve the same row order before any completion-facing or Real-facing consumer reads the terminal seal.*

Proof. Project the carrier of Definition 6368.1. The analytic route is M, S, R, D, T, E, and the structural certificate route is H, C, P, N.

The metric row fixes the distance surface. The stream and regular-readback rows expose the finite windows on which the Cauchy request is inspected, while the dyadic tolerance ledger supplies the positive comparison rows.

The shared-tail row is downstream of those rows and upstream of the terminal Real seal. It therefore records one displayed tail window for the two consuming faces without adding quotient equality, selected limits, choice, host equality, or ambient completeness as local certificate data.

Finally H, C, P, N transport, replay, source, and name exactly the same rows. Hence the listed fields exhaust the local NameCert obligations. □

Theorem 6368.3 (Cauchy-modulus shared-tail cofinality)**.** *Every finite family of* $\mathsf{StreamName}^{\uparrow}$ *finite-window requests and* $\mathsf{RegSeqRat}^{\uparrow}$ *regularity checks that consumes a positive* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row through an accepted* $\mathsf{FrechetCompleteMetric}^{\uparrow}$ *carrier has a common shared-tail refinement. The refinement preserves the metric row, finite stream windows, regular readback, dyadic denominator ledger, shared-tail modulus row, terminal* $\mathsf{Real}^{\uparrow}$ *seal, componentwise* hsame *transport, displayed* Cont *replay,* Pkg *provenance, and local* NameCert *support.*

The common refinement is cofinal among the finite tolerance demands visible to the packet: every listed request factors through its displayed tail window before any completion-facing consumer reaches the Real seal.

Proof. Enumerate the finite StreamName requests and RegSeqRat checks. Starting from the positive dyadic tolerance row, repeatedly read the shared-tail row T of Definition 6368.1 to enlarge the displayed tail window until all listed finite demands are covered.

At each step the metric row M, the stream row S, the regular row R, and the dyadic row D remain the same public source rows. Componentwise hsame transport and displayed Cont replay preserve the accumulated window rather than creating a second tolerance source.

The final window is therefore common to the finite family and is cofinal for the requests named inside the carrier. Since the construction uses only displayed carrier rows, it cannot appeal to quotient completion equality, selected representatives, countable choice, host equality, or ambient Real completeness. □

Theorem 6368.4 (Cauchy-modulus shared-tail Real-seal route)**.** *Every* $\mathsf{Real}^{\uparrow}$ *seal read exported by an accepted* $\mathsf{FrechetCompleteMetric}^{\uparrow}$ *carrier is reached only through the route*

$$\mathsf{Met}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

The shared-tail modulus row must be displayed before the seal is visible, and the read exports no quotient completion class, selected completed point, countable-choice schedule, host real equality, or ambient $\mathsf{Real}^{\uparrow}$ *completeness datum.*

Proof. Begin with the NameCert obligations of Definition 6368.2. They place the metric row, finite stream windows, regular readback, dyadic tolerance ledger, and shared-tail modulus before the Real seal.

Apply the cofinality statement of Definition 6368.3. Any finite completion-facing request is first absorbed into a displayed shared-tail window that both the StreamName and RegSeqRat faces can read.

Only after that finite route is fixed may the consumer read the terminal seal E. The carrier has no coordinate for quotient completion classes, selected completed points, choice schedules, host equality, or ambient completeness, so those objects cannot be exported by the seal route. □

Falsifiable prediction: Every accepted $\mathsf{FrechetCompleteMetric}^{\uparrow}$ completion-facing read must expose the metric row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, shared-tail modulus row, terminal RealUp seal, hsame transport, Cont replay, Pkg provenance, and local NameCert support before a consumer reads completed data.

Independence witness: metric, completemetric, streamname, regseqrat, dyadicratcore, real: $\mathsf{FrechetCompleteMetric}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a metric source row, finite stream windows, regular rational readback, dyadic tolerance data, one shared-tail modulus row, a terminal RealUp seal, transport, replay, provenance, and local naming in one completion packet.

Closure status for $\mathsf{NameCert}_{\mathsf{FrechetCompleteMetric}^{\uparrow}}$**.**

Constructive story: A $\mathsf{FrechetCompleteMetric}^{\uparrow}$ packet is generated from a metric source row, finite StreamName windows, RegSeqRat readback, DyadicRatCore tolerance ledger, a shared-tail modulus row, a Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{FrechetCompleteMetric}^{\uparrow}}$, seed)

Scope closed: Definition 6368.1, Definition 6368.2, Definition 6368.3, and Definition 6368.4 seed the finite shared-tail completion interface from metric Cauchy data to Real-facing reads.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{FrechetCompleteMetric}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: This chapter does not close quotient completion equality, selected completed representatives, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure asks for checked carrier admission, classifier agreement, shared-tail cofinality, Real-seal route support, ledger nonescape, and local naming for BEDC.Derived.FrechetCompleteMetricUp.

Chapter 6369

A Concrete Naming Certificate for $\mathsf{DyadicCauchyCriterion}^{\uparrow}$

Following Bishop and Bridges, the $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ packet records the finite dyadic criterion bridge from dyadic tolerance data to regular rational readback. It sits between the dyadic endpoint and tolerance rows of chapter 291, the finite stream-window surface of chapter 126, the regular Cauchy readback of chapter 301, the terminal real-name seal of chapter 38, and the equality-facing comparison boundary of chapter 1820. The packet names a criterion route for Real and Completion consumers before any completed real line is assumed.

Definition 6369.1 (Dyadic Cauchy criterion carrier)**.** *A* $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (D, W, T, R, E, H, C, P, N)$$

with the following displayed rows:

1. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *endpoint and tolerance ledger, recording dyadic radii, comparison budgets, and common refinements.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row inspected by each tolerance request.*
3. *T is the dyadic Cauchy criterion row, asserting that every requested dyadic tolerance is discharged by some displayed finite window over D and W.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular readback row obtained from the criterion row T.*
5. *E is the equality-facing boundary row: it records* $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison for the displayed regular readbacks and the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after D, W, T, R have been projected.*
6. *H records componentwise* hsame *transport for the dyadic, window, criterion, regular-readback, comparison, seal, replay, provenance, and local naming rows.*
7. *C records displayed* Cont *replay for tolerance admission, finite-window inspection, criterion discharge, regular readback, comparison, and sealing.*
8. *P is the* Pkg *provenance over* $\mathsf{DyadicCauchyCriterion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, BHist, hsame, Cont, *and* $\mathsf{NameCert}$.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $D, W, T, R, E, H, C, P, N$. It has no coordinate for quotient equality, selected limits, countable choice, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6369.2 (Dyadic Cauchy criterion NameCert obligations)**.** *For every accepted* $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ *carrier $K = (D, W, T, R, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{DyadicCauchyCriterion}^{\uparrow}}$ *surface consists exactly of five finite obligation rows: dyadic tolerance admission through D; finite stream-window inspection through W; dyadic criterion discharge through T; regular rational readback through R; and comparison, seal, transport, replay, provenance, and local naming through E, H, C, P, N. No obligation row exports quotient equality, selected limits, countable choice, trichotomy, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the accepted carrier of Definition 6369.1. The first two rows are read through the sibling dyadic and stream surfaces chapter 291 and chapter 126: the dyadic ledger supplies tolerances and common refinements, while the stream row supplies only finite windows.

The criterion row T is therefore local to the displayed finite data. It says that a requested dyadic tolerance has a displayed window witness, not that a global stream quotient or a selected limit has been chosen.

The readback row R is consumed through chapter 301. The equality-facing part of E is read through chapter 1820, and its terminal seal is read only through chapter 38.

The structural rows H, C, P, N transport, replay, source, and name the same displayed packet. Since none of the projected rows contains quotient equality, selected limits, countable choice, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness, the listed rows exhaust the local NameCert obligations. □

Theorem 6369.3 (Dyadic Cauchy criterion RegSeqRat handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$ *consumer of an accepted* $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ *carrier must read the route*

$$\mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicCauchyCriterion}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow}.$$

The handoff packages dyadic tolerance windows as a regular Cauchy readback while retaining only $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$ *comparison and the downstream* $\mathsf{Real}^{\uparrow}$ *seal boundary. It does not export quotient equality, selected limits, countable choice, trichotomy, host real equality, or* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with the dyadic source and finite-window source. By chapter 291 and chapter 126, the rows D and W expose only dyadic budgets and bounded observations.

Use the criterion row T to bind these two sources. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer may read R only after the tolerance request has been discharged by a displayed finite window, so the handoff lands in chapter 301 without choosing a limit representative.

The comparison and seal remain downstream. The row E cites the comparison boundary of chapter 1820 and reaches the terminal real-name boundary only as in chapter 38.

The transport, replay, provenance, and naming rows preserve this route. They contain no quotient equality, selected limit, countable-choice schedule, trichotomy decision, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness principle, so none is exported by the handoff. □

Theorem 6369.4 (Dyadic Cauchy criterion Real seal boundary)**.** *Every Real or Completion consumer of an accepted* $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ *carrier reaches* $\mathsf{Real}^{\uparrow}$ *only through the displayed rows $D, W, T, R, E, H, C, P, N$. The consumer may inspect the dyadic source of chapter 291, the finite-window source of chapter 126, the regular readback of chapter 301, the comparison boundary of*

chapter 1820, and the terminal seal of chapter 38; it receives no quotient equality, selected limits, countable choice, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Unpack the accepted carrier. The value-facing route begins at D and W, whose sibling chapters chapter 291 and chapter 126 expose finite dyadic and finite-window data rather than a host real object.

The criterion and readback rows are next. The row T records the finite dyadic Cauchy discharge, and the row R is read through chapter 301 as regular rational data.

The seal row is last. Its comparison part is bounded by chapter 1820, and its terminal real-name reading is the boundary supplied by chapter 38.

Finally inspect H, C, P, N. These structural rows transport, replay, source, and name the same route; they do not add quotient equality, selected limits, countable choice, trichotomy, host real equality, or $\mathsf{Real}^{\uparrow}$ completeness. Hence Real and Completion consumers cannot escape the displayed criterion bridge. □

Closure status for $\mathsf{NameCert}_{\mathsf{DyadicCauchyCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{DyadicCauchyCriterion}^{\uparrow}$ packet is generated from a DyadicRatCore tolerance ledger, StreamName finite windows, a dyadic criterion row, RegSeqRat readback, CauchyEquivalenceSetoid comparison, a RealUp seal, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{DyadicCauchyCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6369.1, Definition 6369.2, Definition 6369.3, and Definition 6369.4 seed the finite dyadic criterion bridge over $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{CauchyEquivalenceSetoid}^{\uparrow}$, and $\mathsf{Real}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{DyadicCauchyCriterion}^{\uparrow}}, \mathsf{formalTarget})$

Lean target: `BEDC.Derived.DyadicCauchyCriterionUp`

Bridge status: none

Not claimed: This chapter does not close quotient equality, selected limits, countable choice, trichotomy, host real equality, $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked dyadic tolerance admission, finite stream-window inspection, criterion discharge, regular readback, comparison, terminal sealing, transport, replay, provenance, and local naming for BEDC.Derived.DyadicCauchyCriterionUp.

Chapter 6370

A Concrete Naming Certificate for $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$

Following Bishop and Bridges constructive analysis and Brouwer–Troelstra choice-sequence grounding, the $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ packet records choice-sequence Cauchy data as finite BEDC rows before it enters the Real and Completion route. It sits after the choice-sequence source of chapter 1482, the finite stream-window surface of chapter 126, the dyadic tolerance ledger of chapter 291, the regular rational readback of chapter 301, the terminal real-name seal of chapter 38, and the spread-indexed real-name boundary of chapter 4161. The packet also points toward the regular spread sibling chapter 2072. It names a non-quotient Cauchy route; it does not choose an infinite branch, select a Cauchy representative, assert countable choice, identify quotient streams, or add ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6370.1 (Bishop Cauchy choice-sequence carrier)**.** *A* $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (Q, W, D, R, E, S, H, C, P, N)$$

with the following displayed rows:

1. *Q is the* $\mathsf{ChoiceSequence}^{\uparrow}$ *source row, exposing only the inspected finite prefixes of the choice sequence.*
2. *W is the* $\mathsf{StreamName}^{\uparrow}$ *window row through which the inspected prefixes are scheduled.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the displayed prefix windows.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row obtained from the same finite prefix and tolerance data.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after Q, W, D, R have been displayed.*
6. *S is the spread-comparison row, recording the compatible* $\mathsf{CauchyRealSpread}^{\uparrow}$ *or* $\mathsf{RegularCauchySpread}^{\uparrow}$ *read when a spread-indexed consumer asks for the same Cauchy data.*
7. *H records componentwise* hsame *transport for the choice-sequence source, window, tolerance, regular readback, real seal, spread-comparison, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay for prefix inspection, tolerance discharge, regular readback, spread comparison, and terminal sealing.*

9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$, $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealSpread}^{\uparrow}$, $\mathsf{RegularCauchySpread}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{BishopCauchyChoiceSequence}^{\uparrow}}$ *row.*

The classifier compares accepted carriers only through $Q, W, D, R, E, S, H, C, P, N$. It has no coordinate for a host infinite branch, selected Cauchy representative, quotient stream equality, countable choice, trichotomy, host real equality, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Lean4 (statement) `BEDC.Derived.BishopCauchyChoiceSequenceUp`

Theorem 6370.2 (Bishop Cauchy choice-sequence NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ *carrier*

$$K = (Q, W, D, R, E, S, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyChoiceSequence}^{\uparrow}}$ *surface consists exactly of choice-sequence prefix exposure through Q, finite stream-window scheduling through W, dyadic tolerance exposure through D, regular rational readback through R, terminal real sealing through E, spread-compatible comparison through S, componentwise transport through H, displayed replay through C, provenance through P, and local naming through N. No obligation row exports a host infinite branch, selected Cauchy representative, quotient stream equality, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6370.1. The choice-sequence source Q is visible only through the finite window schedule W, and every tolerance read is mediated by the dyadic ledger D.

The regular and real-facing rows stay downstream of that finite source. Row R records the $\mathsf{RegSeqRat}^{\uparrow}$ readback attached to the same prefixes and tolerances, while E is the terminal $\mathsf{Real}^{\uparrow}$ seal reached only after those rows have been displayed.

The spread row S is a comparison route, not a second source of representatives. It records compatibility with the Cauchy-real and regular-Cauchy spread siblings while keeping branch selection, quotient stream equality, countable choice, and ambient completeness outside the carrier.

Finally H, C, P, N transport, replay, source, and name exactly the same packet. Since the carrier has no excluded coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6370.3 (Bishop Cauchy choice-sequence RegSeqRat and Real handoff)**.** *Every* $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, *or Completion consumer of an accepted* $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ *carrier must read the finite route*

$$\mathsf{ChoiceSequence}^{\uparrow} \longmapsto \mathsf{StreamName}^{\uparrow} \longmapsto \mathsf{DyadicRatCore}^{\uparrow} \longmapsto \mathsf{RegSeqRat}^{\uparrow} \longmapsto \mathsf{Real}^{\uparrow}.$$

When a spread-indexed consumer is present, the same read also factors through the displayed $\mathsf{CauchyRealSpread}^{\uparrow}$ *or* $\mathsf{RegularCauchySpread}^{\uparrow}$ *comparison row S. The handoff exports prefix inspection, finite windows, dyadic tolerance, regular readback, real sealing, spread comparison, transport, replay, provenance, and local naming only; it exports no selected infinite branch, selected Cauchy representative, quotient stream equality, countable choice, trichotomy, host real equality, or ambient* $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Start with Definition 6370.2. It identifies the only available source rows as the choice-sequence prefix row, the finite window row, and the dyadic tolerance row.

The regular readback is therefore forced through those rows. A $\mathsf{RegSeqRat}^{\uparrow}$ consumer may read R only after the displayed prefix window and tolerance ledger have been consumed, so the route cannot replace the prefix data by a selected infinite branch or a quotient stream representative.

The real seal remains terminal. Row E is consumed only after Q, W, D, R, and any spread-indexed comparison uses S as a displayed compatibility row beside the Cauchy-real or regular-Cauchy spread siblings.

The structural rows preserve the same route under transport and replay. Since H, C, P, N add no selector, quotient, countable-choice schedule, trichotomy decision, host equality, or completeness principle, every downstream read is confined to the stated handoff. □

Falsifiable prediction: Every accepted $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ read exposes ChoiceSequence prefix rows, StreamName windows, DyadicRatCore tolerance, RegSeqRat readback, RealUp seal, CauchyRealSpread or RegularCauchySpread comparison, componentwise hsame, Cont replay, Pkg provenance, and local NameCert rows before any Real or Completion consumer receives the packet.

Independence witness: choice-sequence, streamname, dyadicratcore, regseqrat, real, cauchy-real-spread, regular-cauchy-spread: none is bijective with $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ because this packet alone binds a Brouwer-style choice-sequence prefix source to the finite Cauchy tolerance, regular readback, terminal Real seal, and spread-compatible comparison route without a selected branch or quotient stream.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyChoiceSequence}^{\uparrow}}$.

Constructive story: A $\mathsf{BishopCauchyChoiceSequence}^{\uparrow}$ packet is generated from ChoiceSequence prefix rows, StreamName windows, DyadicRatCore tolerance rows, RegSeqRat readback, RealUp seal rows, CauchyRealSpread or RegularCauchySpread comparison rows, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyChoiceSequence}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6370.1, Definition 6370.2, and Definition 6370.3 seed the finite choice-sequence Cauchy route over $\mathsf{ChoiceSequence}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{CauchyRealSpread}^{\uparrow}$, and $\mathsf{RegularCauchySpread}^{\uparrow}$ rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyChoiceSequence}^{\uparrow}}, \mathsf{unformalized})$

Lean target: `BEDC.Derived.BishopCauchyChoiceSequenceUp`

Bridge status: none

Not claimed: This chapter does not close selected infinite branches, selected Cauchy representatives, quotient stream equality, countable choice, trichotomy, host real equality, ambient $\mathsf{Real}^{\uparrow}$ completeness, theorem verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, prefix-window scheduling, dyadic tolerance exactness, regular readback, RealUp seal nonescape, spread comparison, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyChoiceSequenceUp.

Chapter 6371

A Concrete Naming Certificate for $\mathsf{CauchyModulusDiagonal}^{\uparrow}$

Following Bishop and Bridges on regular Cauchy reals and diagonal arguments for countable families, the $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ packet records the finite schedule that turns a displayed family of Cauchy-modulus requests into one diagonal tail witness. It sits over the dyadic tolerance ledger of chapter 291, the finite stream windows of chapter 126, the regular rational sequence rows of chapter 301, the terminal Real seal rows of chapter 38, and the modulus algebra surface of chapter 6362. The packet names a local diagonal route for completion-facing consumers; it does not select a global real representative, identify quotient real names, or assert a completed real-line choice principle.

Definition 6371.1 (CauchyModulusDiagonal carrier)**.** *A* $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$D = (F, S, Q, E, M, T, W, H, C, P, N)$$

with the following displayed rows:

1. *F is a finite family row of* $\mathsf{RegSeqRat}^{\uparrow}$ *names whose members expose regular rational readback only through named windows;*
2. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule assigning the finite input windows inspected for each family member and each requested dyadic tolerance;*
3. *Q is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger, recording the dyadic scales consumed by the diagonal comparison;*
4. *E is the* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the displayed regular readback and dyadic tolerance rows have been replayed;*
5. *M is the family of accepted* $\mathsf{CauchyModulus}^{\uparrow}$ *rows controlling the ordinary tail windows for the displayed regular rational names;*
6. *T is the diagonal threshold row. Given a finite tolerance request and a finite family window, it records one displayed tail index large enough for every modulus row visible in that family window;*
7. *W is the diagonal tail-witness row, storing the finite comparison traces showing that all scheduled members remain within the requested dyadic tolerance after the threshold named by T;*

8. *H records componentwise* hsame *transport for F, S, Q, E, M, T, W and for the replay, provenance, and local naming rows;*
9. *C records displayed* Cont *replay from a requested diagonal tail comparison through S, then through F, Q, M, T, W, and only then into the Real-facing seal row E;*
10. *P is the* Pkg *provenance over* $\mathsf{CauchyModulusDiagonal}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{CauchyModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and local naming rows;*
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CauchyModulusDiagonal}^{\uparrow}}$ *row.*

The carrier has no coordinate for a chosen infinite tail, equality of completed Real names, countable choice over all precision requests, or ambient completeness of $\mathsf{Real}^{\uparrow}$.

Theorem 6371.2 (CauchyModulusDiagonal tail witness)**.** *Let*

$$D = (F, S, Q, E, M, T, W, H, C, P, N)$$

be an accepted $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ *carrier. Every diagonal tail read exported by D factors through the finite route*

$$S, F, Q, M \longmapsto T \longmapsto W \longmapsto E.$$

Thus the witness for a requested dyadic tolerance is a displayed threshold over the scheduled finite family window, read from the existing Cauchy-modulus rows and dyadic tolerance ledger before any $\mathsf{Real}^{\uparrow}$ *seal can be consumed.*

Proof. Project the accepted carrier of Definition 6371.1. The stream schedule S fixes the finite family window and the inspected regular rational windows in F before the dyadic tolerance row Q or any modulus row in M is read.

The threshold row T is downstream of those displayed rows. It records a single finite tail index that dominates the visible modulus thresholds for the requested dyadic tolerance, so the row is a finite maximum over displayed certificate entries rather than a choice over completed sequences.

The witness row W then replays the tail comparisons after that threshold and hands the result to the seal row E. Transport by H, replay by C, provenance by P, and local naming by N preserve the same route, so no consumer can bypass the scheduled family, dyadic tolerance, or Cauchy-modulus coordinates. □

Theorem 6371.3 (CauchyModulusDiagonal NameCert obligations)**.** *For every accepted* $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ *carrier*

$$D = (F, S, Q, E, M, T, W, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{CauchyModulusDiagonal}^{\uparrow}}$ *surface consists exactly of carrier admission for D, regular rational family exposure through F, scheduled window exposure through S, dyadic tolerance exposure through Q, Real seal exposure through E, Cauchy-modulus exposure through M, diagonal threshold exposure through T, tail-witness exposure through W, componentwise* hsame *transport through H, displayed* Cont *replay through C,* Pkg *provenance through P, and local naming through N.*

Consequently a $\mathsf{RegSeqRat}^{\uparrow}$ *or* $\mathsf{Real}^{\uparrow}$ *completion-facing consumer may use the diagonal only after reading the displayed finite family schedule, dyadic tolerance ledger, modulus rows, diagonal threshold, and tail witness. The certificate does not claim completed-stream equality, quotient real equality, countable choice, or ambient completeness.*

Proof. Carrier admission lists exactly the rows $F, S, Q, E, M, T, W, H, C, P, N$. The analytic rows expose the regular rational family, finite stream schedule, dyadic tolerance request, Cauchy-modulus thresholds, diagonal threshold, tail-witness comparisons, and terminal Real-facing seal.

The structural rows do not add analytic content. Componentwise hsame transport keeps the same carrier coordinates, displayed Cont replay preserves the order of reads, Pkg provenance records the dependency ledger, and the local NameCert row names precisely this finite surface.

Apply Definition 6371.2. Any completion-facing read must pass through the finite diagonal route before it reaches a $\mathsf{RegSeqRat}^{\uparrow}$ or $\mathsf{Real}^{\uparrow}$ consumer. Since the carrier has no coordinate for completed-stream equality, quotient real equality, choice over all requests, or ambient completeness, none of those claims belongs to the local obligation surface. □

Falsifiable prediction: A consumer that claims $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ tail control while omitting the regular rational family row, StreamName schedule, dyadic tolerance ledger, CauchyModulus rows, diagonal threshold, tail witness, hsame transport, Cont replay, Pkg provenance, or NameCert row refutes the carrier of Definition 6371.1.

Independence witness: dyadicratcore, streamname, regseqrat, real, regular-cauchy-modulus-algebra: $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ binds a finite family schedule, dyadic tolerance ledger, existing CauchyModulus rows, one diagonal threshold, and a tail-witness route in one local NameCert packet; each named sibling lacks at least one of these diagonal-tail coordinates as a local carrier field.

Closure status for $\mathsf{NameCert}_{\mathsf{CauchyModulusDiagonal}^{\uparrow}}$.

Constructive story: A $\mathsf{CauchyModulusDiagonal}^{\uparrow}$ packet is generated from a regular rational family, StreamName schedule, dyadic tolerance ledger, Real seal row, CauchyModulus rows, diagonal threshold, tail witness, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CauchyModulusDiagonal}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6371.1, Definition 6371.2, and Definition 6371.3 seed the finite diagonal modulus packet for scheduled regular Cauchy families.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CauchyModulusDiagonal}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim completed-stream equality, quotient real equality, selected global real representatives, countable choice, or ambient Real completeness.

Upgrade path: Obligation closure requires checked carrier admission, diagonal threshold replay, tail-witness replay, Real seal handoff, and a Lean realization for BEDC.Derived.CauchyModulusDiagonalUp.

Chapter 6372

A Concrete Naming Certificate for $\mathsf{TeichmullerSpace}^{\uparrow}$

Following Teichmuller and Ahlfors-Bers, the $\mathsf{TeichmullerSpace}^{\uparrow}$ packet records a finite moduli-facing NameCert surface for marked conformal structures on a displayed surface. It sits after the charted carrier of chapter 96, the complex-structure rows of chapter 39, the metric-read surface of chapter 3474, and the structural hsame, Cont, Pkg, and NameCert anchors used throughout the concrete-instance corpus. The packet names the finite marking, complex-row, hyperbolic-metric, deformation-ledger, transport, replay, provenance, and local naming data; it does not complete analytic moduli, identify quotient surfaces by host equality, or choose representatives of equivalence classes.

Definition 6372.1 (Teichmuller space carrier)**.** *A* $\mathsf{TeichmullerSpace}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (S, M, J, G, D, E, H, C, P, N)$$

with the following displayed rows:

1. *S is the underlying oriented surface row, read through the* $\mathsf{Manifold}^{\uparrow}$ *chart and incidence carrier.*
2. *M is the marking row: a finite boundary, generator, or triangulation comparison that records how the named surface is related to the reference surface.*
3. *J is the* $\mathsf{Complex}^{\uparrow}$ *row carrying the displayed conformal or complex-chart compatibility data.*
4. *G is the* $\mathsf{HyperbolicMetric}^{\uparrow}$ *row when the finite surface window carries the negative-curvature metric read used by the moduli route.*
5. *D is the deformation ledger. It records displayed Beltrami, quasiconformal, or chart-transition evidence only as finite witness rows attached to S, M, J, G.*
6. *E is the equivalence-facing classifier row. It compares accepted packets by componentwise agreement of the surface, marking, complex row, metric row, and deformation ledger through* hsame*, not by host quotient equality of surfaces.*
7. *H records componentwise* hsame *transport for the surface, marking, complex, hyperbolic-metric, deformation, classifier, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from marked surface data through complex and hyperbolic metric reads to the deformation classifier.*

9. *P is the* Pkg *provenance over* $\mathsf{TeichmullerSpace}^{\uparrow}$, $\mathsf{Manifold}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{HyperbolicMetric}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

10. *N is the local* $\mathsf{NameCert}_{\mathsf{TeichmullerSpace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $S, M, J, G, D, E, H, C, P, N$. It has no coordinate for analytic moduli completion, quotient surface equality, selected representatives, universal curves, measurable Beltrami class equality, or host Teichmuller metric completion.

Theorem 6372.2 (Teichmuller space NameCert obligations)**.** *For every accepted* $\mathsf{TeichmullerSpace}^{\uparrow}$ *carrier $T = (S, M, J, G, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{TeichmullerSpace}^{\uparrow}}$ *obligation surface consists exactly of carrier admission for the marked manifold row; complex-structure exposure; hyperbolic-metric readback when present; deformation-ledger admission; equivalence-facing classifier stability through* hsame*; displayed* Cont *replay of the marked complex route;* Pkg *provenance over the sibling certificates; and local naming by N.*

Every consumer of the packet must therefore read the route

$$\mathsf{Manifold}^{\uparrow},\quad \mathsf{Complex}^{\uparrow},\quad \mathsf{HyperbolicMetric}^{\uparrow},\quad \mathsf{BHist},\quad \mathsf{hsame},\quad \mathsf{Cont},\quad \mathsf{Pkg},\quad \mathsf{NameCert}$$

through $S, M, J, G, D, E, H, C, P, N$, and no consumer receives analytic moduli completion, quotient surface equality, or choice-selected representatives.

Proof. Project the carrier of Definition 6372.1. The surface row S, marking row M, complex row J, hyperbolic metric row G, deformation ledger D, and classifier row E are displayed before the local naming row is consumed.

The moduli-facing comparison is the row E. It is transported only by H and replayed only by C, so equality-sensitive movement is componentwise over the same finite surface, marking, complex, metric, and deformation rows.

The provenance row P names only the cited sibling packets and N names only the local $\mathsf{TeichmullerSpace}^{\uparrow}$ carrier. Since no excluded analytic or quotient coordinate appears in the carrier, the listed rows exhaust the NameCert obligation surface. □

Falsifiable prediction: Every accepted $\mathsf{TeichmullerSpace}^{\uparrow}$ packet displays a ManifoldUp surface row, marking row, ComplexUp conformal row, HyperbolicMetricUp read when present, deformation ledger, classifier row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert row before a moduli-facing consumer reads it.

Independence witness: manifold, complex, hyperbolicmetric: $\mathsf{TeichmullerSpace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries a marked surface row, complex-structure row, hyperbolic-metric read, deformation ledger, classifier transport, replay, provenance, and local naming in one finite moduli packet.

Closure status for $\mathsf{NameCert}_{\mathsf{TeichmullerSpace}^{\uparrow}}$**.**

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{TeichmullerSpace}^{\uparrow}}$, seed)

Scope closed: Definition 6372.1 and Definition 6372.2 seed the finite Teichmuller-space NameCert surface over marked manifold, complex, hyperbolic-metric, deformation, transport, replay, provenance, and local naming rows.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{TeichmullerSpace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: Analytic moduli completion, quotient surface equality, choice-selected representatives, universal curves, measurable Beltrami class equality, host Teichmuller metric completion, Lean verification, and bridge checking are not claimed.

Upgrade path: Obligation closure requires checked carrier admission, marked-surface classifier stability, complex and hyperbolic metric readback, deformation-ledger exactness, transport, replay, provenance, and local NameCert rows for BEDC.Derived.TeichmullerSpaceUp.

Constructive story: A TeichmullerSpace$^{\uparrow}$ packet is generated from a ManifoldUp surface row, a marking row, ComplexUp conformal data, a HyperbolicMetricUp read when present, a finite deformation ledger, classifier transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6373

A Concrete Naming Certificate for $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$

The $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$ packet realizes the closed observational system and finite verification trace interface of chapter 9393 as finite NameCert data. It sits after the closed-observation rows of chapter 616, the finite trace socket of chapter 4149, the GAP-facing routes of chapter 504, and the structural BHist, hsame, Cont, Pkg, and NameCert anchors. The packet names observable records, monotone trace growth, GAP socket exposure, replay, provenance, and local naming without making a substrate-internal universal quantifier available.

Definition 6373.1 (Closed observer audit trace carrier)**.** *A* $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$ *carrier is a finite packet*

$$A = (O, R, T, G, L, H, C, P, N)$$

with the following displayed rows:

1. *O is the closed observer row: the current observation state and its finite* Obs *records as read from the closed observational system surface.*
2. *R is the record ledger. It lists the finite witness records admitted by closed-generation reasoning before the current audit trace is read.*
3. *T is the monotone trace row. It records that later audit states preserve earlier records and append only displayed finite observations.*
4. *G is the GAP socket row. It tags the unfulfilled promise that cannot be discharged from the finite records, especially any universal or induction-facing claim.*
5. *L is the replay ledger. It stores the finite sequence of record reads, preservation checks, and GAP handoffs that a consumer may audit.*
6. *H records componentwise* hsame *transport for the observer state, record ledger, monotone trace, GAP socket, replay ledger, provenance, and local naming rows.*
7. *C records displayed* Cont *replay from finite observations through monotone preservation, GAP exposure, audit replay, provenance, and local naming.*
8. *P is the* Pkg *provenance over* $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$, $\mathsf{ClosedObservationSystem}^{\uparrow}$, FiniteObservationTraceFilter $\mathsf{FiniteTraceGapSocket}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
9. *N is the local* $\mathsf{NameCert}_{\mathsf{ClosedObserverAuditTrace}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, R, T, G, L, H, C, P, N$. It has no coordinate for substrate-internal induction, a completed universal quantifier, external oracle content, erased observations, altered earlier records, or choice-selected future witnesses.

Theorem 6373.2 (Closed observer audit trace NameCert obligations)**.** *For every accepted* ClosedObserverAuditTrac *carrier $A = (O, R, T, G, L, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{ClosedObserverAuditTrace}^{\uparrow}}$ *surface consists exactly of observer-row admission; finite record-ledger admission; monotone trace preservation; GAP socket exposure; replay-ledger exactness; componentwise* hsame *transport; displayed* Cont *replay;* Pkg *provenance over the closed-observation and finite-trace siblings; and local naming by N.*

Thus a consumer reads finite Obs *records, monotone trace rows, GAP socket rows, replay rows, provenance, and local NameCert rows, but receives no substrate-internal universal quantifier, no induction rule, no external oracle content, no erased-history operation, and no selected future witness.*

Proof. Project the carrier of Definition 6373.1. The observer row O, finite record ledger R, monotone trace row T, GAP socket G, and replay ledger L are the complete audit data.

Monotonicity is read from T. It states preservation of earlier records and addition of only displayed later records, so no audit step can delete or rewrite R.

The GAP row G records claims not discharged by the finite trace. The replay row L exposes how the consumer reaches that socket, while H and C move and replay the same rows componentwise.

Finally P and N source and name the packet. Since the carrier contains no coordinate for internal induction, completed universals, oracle content, erased observations, or selected future witnesses, the listed rows exhaust the NameCert obligations. □

Falsifiable prediction: Every accepted $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$ packet displays finite Obs records, a monotone trace row, a GAP socket, replay ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert data before any closed-observer audit consumer can read the trace.

Independence witness: closedobservationsystem, finite-observation-trace-filter, finite-trace-gap-socket: $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries observer records, monotone preservation, GAP exposure, replay, transport, provenance, and local naming as one finite audit packet.

Closure status for $\mathsf{NameCert}_{\mathsf{ClosedObserverAuditTrace}^{\uparrow}}$.

Theory closure: TheoryGate($\mathsf{NameCert}_{\mathsf{ClosedObserverAuditTrace}^{\uparrow}}$, seed)

Scope closed: Definition 6373.1 and Definition 6373.2 seed the finite closed-observer audit route over Obs records, monotone trace rows, GAP sockets, replay, transport, provenance, and local naming.

Formal verification: FormalStatus($\mathsf{NameCert}_{\mathsf{ClosedObserverAuditTrace}^{\uparrow}}$, unformalized)

Bridge status: none

Not claimed: Substrate-internal induction, completed universal quantification, external oracle content, erased observations, altered earlier records, selected future witnesses, Lean verification, and bridge checking are not claimed.

Upgrade path: Obligation closure requires checked observer-row admission, finite record-ledger exactness, monotone trace preservation, GAP socket exposure, replay exactness, transport, provenance, and local NameCert rows for BEDC.Derived.ClosedObserverAuditTraceUp.

Constructive story: A $\mathsf{ClosedObserverAuditTrace}^{\uparrow}$ packet is generated from finite Obs records, a monotone trace row, a GAP socket, replay ledger, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6374

A Concrete Naming Certificate for $\mathsf{FiniteDiniModulus}^{\uparrow}$

Following Dini and Bishop–Bridges constructive analysis, the $\mathsf{FiniteDiniModulus}^{\uparrow}$ packet records the finite NameCert route from compact finite-net data and monotone pointwise descent to one uniform modulus row over $\mathsf{Real}^{\uparrow}$. It consumes the compact finite-net surface of chapter 4760, the monotone Dini route of chapter 4213, the continuous-map graph rows of chapter 130, the dyadic tolerance ledger of chapter 291, the finite minimum-budget handoff of chapter 5816, and the terminal real-name seal of chapter 38. The chapter names only the finite compact-window modulus route; it does not import arbitrary open-cover compactness, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6374.1 (Finite Dini modulus carrier)**.** *A* $\mathsf{FiniteDiniModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$M = (K, G, E, R, T, B, H, C, P, N)$$

with the following displayed rows:

1. *K is the compact finite-net row, exposing the inspected net centers, radius witnesses, and the finite cover ledger consumed by the modulus extraction.*
2. *G is the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row for the monotone family and the displayed limit row over the same compact source.*
3. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row for every function value, limit value, and comparison value read by the packet.*
4. *R is the monotone pointwise descent row, recording the displayed order comparisons and pointwise tail windows used at the finite net centers.*
5. *T is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance row, listing the requested tolerance, comparison slack, and finite dyadic descent used by the route.*
6. *B is the finite minimum-budget row, extracting one displayed index budget from the centerwise Dini comparisons before the uniform-modulus export is available.*
7. *H records componentwise* hsame *transport for the compact net, graph, real seal, monotone descent, tolerance, budget, replay, provenance, and local naming rows.*
8. *C records displayed* Cont *replay from a compact finite-net request through K, G, E, R, T, B, H to the exported modulus route.*

9. *P is the* Pkg *provenance over* $\mathsf{FiniteDiniModulus}^{\uparrow}$, $\mathsf{CompactUniformFiniteNet}^{\uparrow}$, $\mathsf{DiniMonotoneCompactUniform}^{\uparrow}$, $\mathsf{ContinuousMap}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{DiniFiniteMinimumBudget}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
10. *N is the local* $\mathsf{NameCert}_{\mathsf{FiniteDiniModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $K, G, E, R, T, B, H, C, P, N$. It has no coordinate for arbitrary open covers, selected finite subcovers, quotient real equality, hidden limiting values, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Theorem 6374.2 (Finite Dini modulus obligations)**.** *For every accepted* $\mathsf{FiniteDiniModulus}^{\uparrow}$ *carrier*

$$M = (K, G, E, R, T, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{FiniteDiniModulus}^{\uparrow}}$ *surface consists exactly of the following finite obligations:*

1. *expose the compact finite-net row K before any modulus consumer reads a centerwise comparison;*
2. *display the* $\mathsf{ContinuousMap}^{\uparrow}$ *graph row G and the terminal* $\mathsf{Real}^{\uparrow}$ *seal row E before any function-value comparison is used;*
3. *route monotone pointwise descent through R and dyadic tolerance descent through T;*
4. *extract the uniform index budget only through the finite minimum-budget row B;*
5. *preserve the same route by componentwise* hsame *transport and displayed* Cont *replay through H, C;*
6. *source and name the packet through* Pkg *provenance P and the local* NameCert *row N.*

No obligation row states arbitrary open-cover compactness, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Project the carrier of Definition 6374.1. The compact data are exactly the finite net centers and radius witnesses in K, so a modulus consumer cannot inspect an arbitrary open cover or a selected subcover.

Next read the analytic rows. The graph row G supplies the displayed $\mathsf{ContinuousMap}^{\uparrow}$ family and limit graph, while E is the only $\mathsf{Real}^{\uparrow}$ seal for the values compared by the packet.

The Dini part is split into visible finite rows. Monotone pointwise descent is recorded in R, and the requested tolerance plus slack comparisons are recorded in the dyadic row T.

The exported budget is downstream of those rows. The finite minimum-budget row B folds the centerwise comparisons into one displayed index budget, and H, C, P, N transport, replay, source, and name the same packet. Since no excluded coordinate occurs in the carrier, the local obligation surface is exhausted by the listed rows. □

Theorem 6374.3 (Finite Dini uniform modulus route)**.** *Every uniform-modulus consumer of an accepted* $\mathsf{FiniteDiniModulus}^{\uparrow}$ *carrier must factor through the displayed route*

$$\mathsf{CompactUniformFiniteNet}^{\uparrow},\ \mathsf{DiniMonotoneCompactUniform}^{\uparrow},\ \mathsf{ContinuousMap}^{\uparrow},\ \mathsf{Real}^{\uparrow},\ \mathsf{DyadicRatCore}^{\uparrow},\ \mathsf{DiniFiniteMi}$$

Thus the packet exports a uniform modulus route for compact finite-net and monotone pointwise-descent data. It does not claim arbitrary open-cover compactness, quotient real equality, selected limits, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ *completeness.*

Proof. Begin with the accepted carrier. The finite source row K names the compact net inspected by the route, and the graph plus real-seal rows G, E name the continuous-map values and limit values that can be compared.

Apply Definition 6374.2. The monotone descent row R, dyadic tolerance row T, and finite minimum-budget row B are the only rows from which a uniform index budget can be read.

The structural rows then keep the consumer on the same packet. Componentwise transport in H and replay in C preserve the displayed dependency order, while P, N provide provenance and local naming. Because the carrier has no field for arbitrary open covers, quotient real equality, selected limits, countable choice, or ambient completeness, the exported object is exactly the finite uniform-modulus route. □

Falsifiable prediction: Every accepted $\mathsf{FiniteDiniModulus}^{\uparrow}$ uniform-modulus read first exposes compact finite-net rows, ContinuousMap graph rows, RealUp seals, monotone pointwise descent, DyadicRatCore tolerance cells, a finite minimum budget, componentwise hsame transport, Cont replay, Pkg provenance, and a local NameCert row.

Independence witness: compact-uniform-finite-net, dini-monotone-compact-uniform, continuousmap, real, dyadicratcore, dini-finite-minimum-budget: $\mathsf{FiniteDiniModulus}^{\uparrow}$ is not bijective to any listed sibling because it binds finite compact-net Dini descent to one uniform-modulus budget route at the RealUp seal.

Closure status for $\mathsf{NameCert}_{\mathsf{FiniteDiniModulus}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{FiniteDiniModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6374.1, Definition 6374.2, and Definition 6374.3 seed the finite compact-net Dini modulus route over ContinuousMap, Real, DyadicRatCore, finite budget, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{FiniteDiniModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close arbitrary open-cover compactness, quotient real equality, selected limits, countable choice, Lean verification, bridge checking, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Upgrade path: Obligation closure requires checked carrier admission, compact finite-net exposure, ContinuousMap graph visibility, Real seal nonescape, monotone pointwise descent, dyadic tolerance exactness, finite budget extraction, transport, replay, provenance, and local naming for BEDC.Derived.FiniteDiniModulusUp.

Constructive story: A $\mathsf{FiniteDiniModulus}^{\uparrow}$ packet is generated from compact finite-net rows, ContinuousMap graph rows, RealUp seal rows, monotone pointwise descent, DyadicRatCore tolerance rows, a finite minimum-budget row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6375

A Concrete Naming Certificate for $\mathsf{CompactRealUniformModulus}^{\uparrow}$

Following Bishop and Bridges constructive analysis, the $\mathsf{CompactRealUniformModulus}^{\uparrow}$ packet records the finite uniform-modulus surface available from compact real data when the compactness evidence is already displayed as metric and finite-net information. It reads metric real distances through chapter 1638, uniform control through chapter 369, compact metric finite data through chapter 129, finite observations through chapter 126, rational readback through chapter 301, and terminal real names through chapter 38. The packet names an L10 adjacent finite-modulus route over compact real sources; it is not an open-cover compactness theorem, a choice-selected finite subcover, quotient real equality, or ambient completeness principle.

Definition 6375.1 (Compact real uniform modulus carrier)**.** *A* $\mathsf{CompactRealUniformModulus}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$Q = (R, M, U, K, W, A, S, T, C, P, N)$$

with the following displayed rows:

1. *R is the* $\mathsf{Real}^{\uparrow}$ *source row, carrying the real names whose finite compact real data are being read.*
2. *M is the* $\mathsf{RealMetric}^{\uparrow}$ *row, recording the metric distance comparison surface used by the compact real request.*
3. *U is the* $\mathsf{UniformModulus}^{\uparrow}$ *row, exposing the dyadic tolerance budget and response modulus.*
4. *K is the* $\mathsf{CompactMetric}^{\uparrow}$ *row, read only as finite compact metric data for the displayed real source.*
5. *W is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row opened at the compact real observations requested by R, M, K.*
6. *A is the* $\mathsf{RegSeqRat}^{\uparrow}$ *readback row attached to the same displayed windows and tolerance budget.*
7. *S is the target* $\mathsf{Real}^{\uparrow}$ *seal row reached after metric comparison, compact metric data, finite windows, readback, and uniform-modulus control have all been displayed.*
8. *T records componentwise* hsame *transport for R, M, U, K, W, A, S.*
9. *C records displayed* Cont *replay through the finite route*

$$R, M, K \longmapsto W, A, U \longmapsto S, T, P, N.$$

10. *P is the* Pkg *provenance over* $\mathsf{CompactRealUniformModulus}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, $\mathsf{RealMetric}^{\uparrow}$, $\mathsf{UniformModulus}^{\uparrow}$, $\mathsf{CompactMetric}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{CompactRealUniformModulus}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $R, M, U, K, W, A, S, T, C, P, N$. It has no coordinate for open-cover compactness, countable choice, selected subcovers, quotient real equality, host real representatives, or ambient Real completeness.

Lean4 ✓ `BEDC.Derived.CompactRealUniformModulusUp.CompactRealUniformModulusTasteGate single carrier alignment`

Theorem 6375.2 (Compact real uniform modulus NameCert obligations)**.** *For every accepted* $\mathsf{CompactRealUniformModulus}^{\uparrow}$ *carrier $Q = (R, M, U, K, W, A, S, T, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{CompactRealUniformModu}}$ *surface consists exactly of real-source admission, real-metric comparison admission, compact-metric finite-data admission, finite StreamName window exposure, RegSeqRat readback, uniform-modulus tolerance exactness, target Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6375.1. The projection exposes the real source, real metric, uniform modulus, compact metric, finite windows, rational readback, target seal, and structural rows.

The compact real obligations are upstream of every modulus read. A consumer must first display R, M, K before it can open finite windows W or consume the readback row A.

The uniform-modulus row U then binds all accepted compact real observations to the same dyadic tolerance budget. The seal S is downstream of those rows, while T, C, P, N transport, replay, source, and name the same finite packet.

Since the carrier has no coordinate for open covers, selected subcovers, quotient equality, countable choice, host real representatives, or ambient Real completeness, the listed rows exhaust the local NameCert surface. □

Theorem 6375.3 (Compact real uniform modulus finite-net extraction)**.** *Every finite-net extraction from an accepted* $\mathsf{CompactRealUniformModulus}^{\uparrow}$ *carrier factors through*

$$R, M, K \longmapsto W, A, U.$$

The compact metric row K may hand off only the displayed compact real source to finite StreamName windows, RegSeqRat readback, and the exposed uniform-modulus row. It does not export open-cover compactness, a choice-selected subcover, quotient syntax equality for reals, or ambient Real completeness.

Proof. Start from the carrier projection. The extraction request has no source other than the displayed real source, real-metric comparison, and compact-metric finite-data rows R, M, K.

The finite windows W and rational readback A are read at the tolerance governed by U. Thus the extraction is a local finite route through displayed compact real data and the visible uniform-modulus row.

The structural rows preserve the same route. Because no row stores an open cover, selected subcover, quotient equality proof, choice schedule, or ambient completeness witness, the extraction cannot be read as any of the excluded principles. □

Falsifiable prediction: Every $\mathsf{CompactRealUniformModulus}^{\uparrow}$ read that exports a uniform-modulus or finite-net result without first exposing real source data, RealMetric comparison, CompactMetric finite data, StreamName windows, RegSeqRat readback, target Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert is outside the carrier surface.

Independence witness: Nearest siblings: real, real-metric, uniform-modulus, compact-metric, streamname, and regseqrat. $\mathsf{CompactRealUniformModulus}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries compact real source data, metric comparison, finite compact metric evidence, finite windows, rational readback, uniform-modulus control, target Real sealing, transport, replay, provenance, and local naming as one packet.

Closure status for $\mathsf{NameCert}_{\mathsf{CompactRealUniformModulus}^{\uparrow}}$.

Constructive story: A $\mathsf{CompactRealUniformModulus}^{\uparrow}$ packet is generated from Real source rows, RealMetric comparison rows, a UniformModulus row, CompactMetric finite data, StreamName finite windows, RegSeqRat readback, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{CompactRealUniformModulus}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6375.1, Definition 6375.2, and Definition 6375.3 seed the finite compact real uniform-modulus route over metric comparison, compact finite data, windows, readback, seal, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{CompactRealUniformModulus}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim open-cover compactness, countable choice, selected finite subcovers, quotient real equality, host real representatives, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, RealMetric comparison exactness, UniformModulus tolerance exactness, CompactMetric finite-data admission, StreamName window replay, RegSeqRat readback, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.CompactRealUniformModulusUp.

Chapter 6376

A Concrete Naming Certificate for $\mathsf{WeylCriterion}^{\uparrow}$

Following Weyl and Kuipers–Niederreiter, the $\mathsf{WeylCriterion}^{\uparrow}$ packet records the finite BEDC surface of the Weyl equidistribution criterion. A sequence in the circle is inspected through finite exponential-sum windows: rational frequency rows, complex phase rows, unit-circle seal rows, finite stream-name schedules, regular rational readback, dyadic tolerance budgets, hsame transport, displayed Cont replay, Pkg provenance, and local NameCert rows. The packet sits after the natural-number index surface of chapter 28, the complex arithmetic surface of chapter 39, the circle carrier of chapter 58, finite schedules from chapter 126, rational readback from chapter 301, dyadic tolerance arithmetic from chapter 291, and terminal real seals from chapter 38. It names a finite certificate route for equidistribution, not a host integral, countable choice over frequencies, quotient equality of circle points, or ambient Fourier analysis.

Definition 6376.1 (Weyl criterion carrier)**.** *A* $\mathsf{WeylCriterion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$W = (I, F, Z, U, S, R, D, E, H, C, P, N)$$

with the following displayed rows:

1. *I is the* $\mathsf{Nat}^{\uparrow}$ *index-window row naming the finite initial segment being averaged.*
2. *F is the finite nonzero integer-frequency row read through rational and natural-number code rows.*
3. *Z is the* $\mathsf{Complex}^{\uparrow}$ *exponential-phase row for the displayed frequency and circle sample.*
4. *U is the* $\mathsf{S}^{1\uparrow}$ *unit-circle sample row whose points are inspected only through the displayed finite schedule.*
5. *S is the* $\mathsf{StreamName}^{\uparrow}$ *schedule row exposing the finite sample windows consumed by the average.*
6. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *rational readback row for the real and imaginary parts of the displayed averages.*
7. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance-budget row used to compare the finite averages with zero.*
8. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row recording the equidistribution-facing zero-limit boundary visible to the packet.*
9. *H records componentwise* hsame *transport for I, F, Z, U, S, R, D, E.*

10. *C records displayed* Cont *replay through the finite route*

$$I, F, U, S \longmapsto Z, R, D \longmapsto E, H, P, N.$$

11. *P is the* Pkg *provenance over* $\mathsf{WeylCriterion}^{\uparrow}$, $\mathsf{Nat}^{\uparrow}$, $\mathsf{Complex}^{\uparrow}$, $\mathsf{S}^{1\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.

12. *N is the local* $\mathsf{NameCert}_{\mathsf{WeylCriterion}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through the displayed rows $I, F, Z, U, S, R, D, E, H, C, P, N$. It has no coordinate for a host integral, a completed infinite average, selected limiting subsequences, quotient equality of circle points, or an unrestricted Fourier basis.

Theorem 6376.2 (Weyl criterion NameCert obligations)**.** *For every accepted* $\mathsf{WeylCriterion}^{\uparrow}$ *carrier $W = (I, F, Z, U, S, R, D, E, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{WeylCriterion}^{\uparrow}}$ *surface consists exactly of index-window admission, nonzero frequency admission, complex phase admission, unit-circle sample admission, finite schedule exposure, regular rational readback, dyadic tolerance comparison, terminal Real seal admission, componentwise transport, displayed replay, provenance, and local naming for the same displayed rows.*

Proof. Project the carrier of Definition 6376.1. The projection exposes the finite index window, frequency row, circle sample, finite schedule, complex phase, rational readback, dyadic tolerance, terminal seal, and structural rows.

The average can be consumed only after the index and schedule rows have been displayed. The frequency and circle rows then determine the complex phase row, while the rational readback and dyadic tolerance rows expose the finite zero-comparison data visible to the packet.

The terminal Real seal records only the boundary reached through these finite rows. Componentwise transport, replay, provenance, and local naming preserve the same route. Since the carrier has no coordinate for host integrals, completed infinite averages, selected subsequences, quotient circle equality, or unrestricted Fourier analysis, the listed rows exhaust the local NameCert surface. □

Theorem 6376.3 (Weyl criterion equidistribution route)**.** *Every equidistribution-facing read from an accepted* $\mathsf{WeylCriterion}^{\uparrow}$ *carrier factors through*

$$I, F, U, S \longmapsto Z, R, D \longmapsto E.$$

Thus the consumer may read only finite exponential-sum windows, rational readback, dyadic zero-comparison budgets, and the terminal Real seal. It cannot extract a host integral, a choice-selected limiting frequency, quotient equality of circle points, or an ambient Fourier theorem.

Proof. Start with the displayed index, frequency, circle, and schedule rows. These rows are the only entries from which a finite exponential sum can be formed.

The complex phase row Z and the regular rational readback row R expose the finite average through rational real and imaginary coordinates. The dyadic tolerance row D then supplies the visible zero-comparison budget.

Finally the Real seal E is downstream of those finite rows, and H, C, P, N transport, replay, source, and name the same packet. Because no row stores a host integral, a completed infinite average, a selected limiting subsequence, quotient circle equality, or an unrestricted Fourier basis, the factorization is exactly the stated finite BEDC route. □

Falsifiable prediction: Every $\mathsf{WeylCriterion}^{\uparrow}$ equidistribution read that exports a zero-limit or uniform-distribution conclusion without first exposing Nat index windows, nonzero frequency rows, Complex phase rows, SOne sample rows, StreamName schedules, RegSeqRat readback, DyadicRatCore tolerance comparison, Real sealing, componentwise hsame transport, displayed Cont replay, Pkg provenance, and local NameCert is outside the carrier surface.

Independence witness: Nearest siblings: nat, complex, sone, streamname, regseqrat, dyadicratcore, and real. $\mathsf{WeylCriterion}^{\uparrow}$ is not bijective to any listed sibling because it jointly carries finite index windows, nonzero frequencies, complex phase rows, unit-circle samples, finite schedules, rational readback, dyadic tolerance comparison, terminal Real sealing, transport, replay, provenance, and local naming as one equidistribution certificate.

Closure status for $\mathsf{NameCert}_{\mathsf{WeylCriterion}^{\uparrow}}$.

Constructive story: A $\mathsf{WeylCriterion}^{\uparrow}$ packet is generated from Nat index-window rows, nonzero frequency rows, Complex exponential-phase rows, SOne sample rows, StreamName finite schedules, RegSeqRat rational readback, DyadicRatCore tolerance comparison, a Real seal row, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{WeylCriterion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6376.1, Definition 6376.2, and Definition 6376.3 seed the finite Weyl equidistribution certificate over index windows, frequencies, complex phases, circle samples, schedules, rational readback, dyadic tolerance, Real sealing, transport, replay, provenance, and local naming.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{WeylCriterion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim host integrals, unrestricted Fourier analysis, quotient equality of circle points, countable choice over frequencies, completed infinite averages, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, frequency nonzero admission, complex phase exactness, circle sample admission, schedule-window replay, rational readback, dyadic zero-comparison, Real seal nonescape, transport, replay, provenance, and local naming for BEDC.Derived.WeylCriterionUp.

Chapter 6377

A Concrete Naming Certificate for $\mathsf{AsymptoticRegularity}^{\uparrow}$

Following Bruck and Reich on nonexpansive mappings and asymptotic regularity, the $\mathsf{AsymptoticRegularity}^{\uparrow}$ packet records orbit-displacement decay as finite NameCert data rather than as a host fixed-point theorem. It sits over the metric-distance surface of chapter 57, the regular rational readback of chapter 301, the Cauchy-modulus ledger of chapter 292, the terminal real-name seal of chapter 38, and the continuous-map route of chapter 130. The packet gives Completion and fixed-point-facing consumers a displayed orbit window before any $\mathsf{Real}^{\uparrow}$ seal is inspected.

Definition 6377.1 (Asymptotic regularity carrier)**.** *An* $\mathsf{AsymptoticRegularity}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$A = (O, D, M, E, S, T, H, C, P, N)$$

with the following displayed rows:

1. *O is the orbit-window row, naming the finite iterates and the source map row read by a* $\mathsf{ContinuousMap}^{\uparrow}$ *consumer.*
2. *D is the displacement-metric row, recording the displayed* $\mathsf{Met}^{\uparrow}$ *distances between successive orbit entries.*
3. *M is the Cauchy-modulus row, giving the finite tolerance window through which the displacement sequence is read.*
4. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row, reached only after O, D, M have been displayed.*
5. *S is the map-stability row, recording the nonexpansive or continuity budget used to keep the displayed orbit inside the same metric source.*
6. *T is the transport row binding orbit windows, metric displacements, modulus reads, Real seals, and map-stability data under componentwise* hsame*.*
7. *H records the completion-facing handoff from regular Cauchy readback to the displayed orbit-displacement row.*
8. *C records displayed* Cont *replay through O, D, M, E, S, T, H.*
9. *P is the* Pkg *provenance over* $\mathsf{AsymptoticRegularity}^{\uparrow}$*,* $\mathsf{Met}^{\uparrow}$*,* $\mathsf{RegSeqRat}^{\uparrow}$*,* $\mathsf{CauchyModulus}^{\uparrow}$*,* $\mathsf{Real}^{\uparrow}$*,* $\mathsf{ContinuousMap}^{\uparrow}$*,* BHist*,* hsame*,* Cont*, and* $\mathsf{NameCert}$*.*
10. *N is the local* $\mathsf{NameCert}_{\mathsf{AsymptoticRegularity}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through $O, D, M, E, S, T, H, C, P, N$.

Theorem 6377.2 (Asymptotic regularity NameCert obligations)**.** *For every accepted* $\mathsf{AsymptoticRegularity}^{\uparrow}$ *carrier $A = (O, D, M, E, S, T, H, C, P, N)$, the local* $\mathsf{NameCert}_{\mathsf{AsymptoticRegularity}^{\uparrow}}$ *surface consists exactly of the following five obligation rows:*

1. *carrier admission through the displayed orbit window O, displacement-metric row D, Cauchy-modulus row M, terminal seal E, map-stability row S, transport row T, handoff row H, replay row C, provenance row P, and local naming row N;*
2. *metric displacement visibility, requiring every orbit-decay read to expose the* $\mathsf{Met}^{\uparrow}$ *distance row D before it can be consumed as a real-valued displacement claim;*
3. *Cauchy-modulus routing, requiring M to give the finite tolerance window through which the displacement sequence is inspected before E is opened;*
4. *map-stability preservation, requiring S, T, C to keep the orbit entries and displacement rows inside the same displayed map and metric source;*
5. *seal non-escape, requiring E, P, N to export no host fixed-point theorem, ambient completeness, quotient orbit equality, selected limit, countable choice, or bridge-checking evidence.*

No local NameCert obligation is stored outside these rows.

Proof. Project the carrier of Definition 6377.1. Carrier admission lists every row available to a consumer, so the orbit window, metric displacement, modulus, seal, stability, transport, handoff, replay, provenance, and local naming rows are the complete packet surface.

The analytic read begins with O and D. A displacement claim has no separate host metric coordinate; it is read only from the displayed $\mathsf{Met}^{\uparrow}$ row attached to the finite orbit window.

The tolerance step is M. It supplies the finite Cauchy window before the terminal Real seal E is opened, while S, T, C preserve the same map and orbit rows under stability, transport, and replay.

Finally P, N source and name only this route. Since the carrier contains no coordinate for a host fixed point, ambient completeness, quotient orbit equality, selected limit, choice principle, or bridge certificate, the five listed rows exhaust the obligation surface. □

Theorem 6377.3 (Asymptotic regularity Cauchy-orbit handoff)**.** *Every Completion or fixed-point-facing consumer of an accepted* $\mathsf{AsymptoticRegularity}^{\uparrow}$ *carrier must read the displayed orbit-window row O, metric displacement row D, and Cauchy-modulus row M before any terminal* $\mathsf{Real}^{\uparrow}$ *seal E is exposed. The handoff may use* $\mathsf{RegSeqRat}^{\uparrow}$ *readback,* $\mathsf{CauchyModulus}^{\uparrow}$ *tolerance data, and* $\mathsf{ContinuousMap}^{\uparrow}$ *stability, but it exports no host fixed-point theorem, ambient completeness assertion, quotient orbit equality, selected orbit limit, countable-choice schedule, or bridge-checking row.*

Proof. Start from Definition 6377.2. The obligation surface forces carrier admission through $O, D, M, E, S, T, H, C, P, N$ and places the metric displacement and modulus rows before the terminal seal.

A Completion consumer therefore sees the regular Cauchy and modulus rows through the finite handoff H before reading E. A fixed-point-facing consumer sees the same orbit window and map-stability row S before any displacement-decay claim is consumed.

The structural rows T, C, P, N preserve the same displayed route. Because the carrier has no coordinate for fixed points, ambient completeness, quotient orbit equality, selected limits, countable choice, or bridge checking, none of those objects can be read from the handoff. □

Falsifiable prediction: Every accepted $\mathsf{AsymptoticRegularity}^{\uparrow}$ read exposes a finite orbit window, metric displacement row, Cauchy-modulus row, map-stability row, transport, handoff, replay, provenance, and local NameCert before a Real-facing displacement seal is consumed.

Independence witness: metric, regular-Cauchy, Cauchy-modulus, Real, continuous-map: $\mathsf{AsymptoticRegularity}^{\uparrow}$ is not bijective to any listed source because it binds orbit-displacement decay into a finite consumer route while preserving each source as a named sibling row.

Closure status for $\mathsf{NameCert}_{\mathsf{AsymptoticRegularity}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{AsymptoticRegularity}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6377.1, Definition 6377.2, and Definition 6377.3 seed the constructive orbit-displacement certificate over displayed metric, modulus, Real-seal, map-stability, transport, replay, provenance, and local naming rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{AsymptoticRegularity}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close host fixed-point theorems, ambient completeness, quotient orbit equality, selected limits, choice, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, displacement-metric visibility, Cauchy-modulus routing, map-stability preservation, Real-seal non-escape, transport, replay, provenance, and local naming for BEDC.Derived.AsymptoticRegula

Constructive story: An $\mathsf{AsymptoticRegularity}^{\uparrow}$ packet is generated from a finite orbit window, displayed metric displacements, a Cauchy-modulus tolerance row, a terminal Real seal, a map-stability row, componentwise hsame transport, completion-facing handoff, Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6378

A Concrete Naming Certificate for $\mathsf{PerronIntegral}^{\uparrow}$

Following Perron and the Bishop–Bridges constructive reading of integration, the $\mathsf{PerronIntegral}^{\uparrow}$ packet records an integral value through finite majorant and minorant envelopes before any real seal is exported. It sits beside the gauge-partition route of chapter 1999, the tagged-mesh row of chapter 1526, the upper/lower finite-sum boundary of chapter 1880, the regular rational readback of chapter 301, and the terminal real-name seal of chapter 38. The packet names a finite envelope interface for one displayed integrand window; it is not measure completion, quotient function equality, a selected primitive, countable choice, or ambient $\mathsf{Real}^{\uparrow}$ completeness.

Definition 6378.1 (Perron integral carrier)**.** *A* $\mathsf{PerronIntegral}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$P = (G, R, D, M, m, U, L, E, H, C, Q, N)$$

with the following displayed rows:

1. *G is the* $\mathsf{GaugeIntegral}^{\uparrow}$ *source row, exposing only the finite gauge-partition request and its tagged subdivision boundary.*
2. *R is the* $\mathsf{RiemannIntegral}^{\uparrow}$ *readback row over the same displayed interval and integrand window.*
3. *D is the* $\mathsf{DarbouxSum}^{\uparrow}$ *comparison row, carrying finite upper and lower Darboux sums for the displayed partition data.*
4. *M is the finite majorant envelope row. Each entry records a displayed upper control function, its gauge or Riemann readback, and the regular rational upper value by which it bounds the integrand window.*
5. *m is the finite minorant envelope row. Each entry records a displayed lower control function, its gauge or Riemann readback, and the regular rational lower value by which it is bounded by the integrand window.*
6. *U and L are the* $\mathsf{RegSeqRat}^{\uparrow}$ *upper and lower value rows read from the majorant and minorant envelopes.*
7. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal row reached only after the finite upper and lower regular-rational envelopes have been displayed and compared.*
8. *H records componentwise* hsame *transport for gauge, Riemann, Darboux-sum, majorant, minorant, upper value, lower value, real seal, replay, provenance, and local naming rows.*

9. *C records displayed* Cont *replay from gauge and Riemann reads through Darboux upper/lower sums, then through the majorant/minorant envelopes, the regular-rational rows, and the terminal real seal.*
10. *Q is the* Pkg *provenance over* $\mathsf{PerronIntegral}^{\uparrow}$, $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, *and* NameCert.
11. *N is the local* $\mathsf{NameCert}_{\mathsf{PerronIntegral}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for measure completion, quotient equality of functions, selected primitives, countable choice, or ambient real completeness.

Lean4 ✓ `BEDC.Derived.PerronIntegralUp.PerronIntegralTasteGate single carrier alignment`

Theorem 6378.2 (Perron integral NameCert obligations)**.** *For every accepted* $\mathsf{PerronIntegral}^{\uparrow}$ *carrier* $P = (G, R, D, M, m, U, L, E, H, C, Q, N)$*, the local* $\mathsf{NameCert}_{\mathsf{PerronIntegral}^{\uparrow}}$ *surface consists exactly of carrier admission, gauge and Riemann source visibility, Darboux upper/lower finite-sum compatibility, majorant envelope exposure, minorant envelope exposure, regular-rational upper and lower value readback, terminal real-seal handoff, componentwise transport, displayed replay, package provenance, and local naming.*

Lean4 ✓ `BEDC.Derived.PerronIntegralUp.PerronIntegralNamecertObligations`

Proof. Project the carrier of Definition 6378.1. The rows G, R, D make the gauge, Riemann, and Darboux-sum sources visible before any envelope value is read.

The majorant and minorant obligations are finite. The row M lists upper control windows and their regular-rational upper values, while m lists lower control windows and their regular-rational lower values. The rows U and L expose those values as $\mathsf{RegSeqRat}^{\uparrow}$ data before the real seal is available.

Finally E, H, C, Q, N seal, transport, replay, source, and name the same displayed packet. Since no carrier coordinate stores a measure space, quotient function equality, selected primitive, choice principle, or ambient real-completeness theorem, the listed rows exhaust the local NameCert surface. □

Theorem 6378.3 (Perron integral major/minor route)**.** *Every accepted* $\mathsf{PerronIntegral}^{\uparrow}$ *read that reaches a* $\mathsf{Real}^{\uparrow}$ *seal factors through the finite route*

$$\mathsf{GaugeIntegral}^{\uparrow}, \quad \mathsf{RiemannIntegral}^{\uparrow}, \quad \mathsf{DarbouxSum}^{\uparrow}, \quad M, \quad m, \quad \mathsf{RegSeqRat}^{\uparrow}, \quad \mathsf{Real}^{\uparrow}.$$

Thus the Perron value is exposed only as a displayed upper/lower envelope comparison over gauge, Riemann, Darboux-sum, regular-rational, transport, replay, provenance, and local naming rows.

Lean4 ✓ `BEDC.Derived.PerronIntegralUp.PerronIntegralMajorMinorRoute`

Proof. Start with a consumer read of the terminal seal E. By Definition 6378.1, E is downstream of the gauge row G, the Riemann readback R, the Darboux-sum comparison D, and the majorant/minorant envelope rows M, m.

The regular-rational boundary is then forced by U and L. The upper and lower values are not independent real numbers; they are read from the displayed envelope rows and compared only after the finite gauge, Riemann, and Darboux data have been opened.

Componentwise transport through H and replay through C preserve the same order, while Q and N name no additional analytic source. Hence every real-facing Perron read factors through the stated finite major/minor envelope route. □

Falsifiable prediction: A purported $\mathsf{PerronIntegral}^{\uparrow}$ consumer that exports a real-valued integral while omitting the gauge source, Riemann readback, Darboux-sum comparison, majorant envelope, minorant envelope, regular-rational upper and lower rows, terminal real seal, transport, replay, provenance, or local naming row refutes the packet.

Independence witness: gauge-integral, riemannintegral, darboux-sum, regseqrat, real: $\mathsf{PerronIntegral}^{\uparrow}$ is not bijective with any listed sibling because it joins majorant and minorant envelope rows to the gauge, Riemann, Darboux-sum, regular-rational, and real-seal route as one named packet.

Closure status for $\mathsf{NameCert}_{\mathsf{PerronIntegral}^{\uparrow}}$.

Constructive story: A $\mathsf{PerronIntegral}^{\uparrow}$ packet is generated from gauge and Riemann source rows, a Darboux upper/lower finite-sum comparison row, finite majorant and minorant envelope rows, upper and lower RegSeqRat readback, a terminal Real seal, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{PerronIntegral}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6378.1, Definition 6378.2, and Definition 6378.3 seed the finite Perron major/minor envelope route over $\mathsf{GaugeIntegral}^{\uparrow}$, $\mathsf{RiemannIntegral}^{\uparrow}$, $\mathsf{DarbouxSum}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, and NameCert rows.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{PerronIntegral}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not close measure integration, quotient equality of functions, selected primitives, countable choice, ambient $\mathsf{Real}^{\uparrow}$ completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, gauge and Riemann source visibility, Darboux-sum compatibility, majorant and minorant envelope exactness, regular-rational readback, real-seal non-escape, componentwise transport, displayed replay, provenance, and local naming for BEDC.Derived.PerronIntegralUp.

Chapter 6379

A Concrete Naming Certificate for $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$

Following Bishop and Bridges on constructive Cauchy completion, the $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ packet records a rate-bearing completion route for regular rational names as finite BEDC data. It sits after the dyadic tolerance core of chapter 291, the finite stream naming surface of chapter 126, the regular-rational readback of chapter 301, the terminal real-name seal of chapter 38, and the Bishop regular completion route of chapter 1639. The packet names rates, windows, tolerances, readbacks, and Real-facing seals by displayed rows only; it does not quotient regular sequences, select limits, invoke countable choice, import host real equality, or assume ambient completeness.

Definition 6379.1 (Bishop Cauchy rate completion carrier)**.** *A* $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$K = (S, M, D, R, E, B, H, C, P, N)$$

with the following displayed rows:

1. *S is the* $\mathsf{StreamName}^{\uparrow}$ *finite-window row naming the inspected rational stream windows.*
2. *M is the Bishop rate row, recording the displayed modulus request by which later windows are compared with earlier windows.*
3. *D is the* $\mathsf{DyadicRatCore}^{\uparrow}$ *tolerance ledger attached to the same rate request and window family.*
4. *R is the* $\mathsf{RegSeqRat}^{\uparrow}$ *regular-rational readback obtained from S, M, D.*
5. *E is the terminal* $\mathsf{Real}^{\uparrow}$ *seal reached only after the stream windows, rate row, dyadic tolerance ledger, and regular-rational readback are displayed.*
6. *B is the Bishop regular Cauchy completion row read from chapter 1639; it records that the rate-bearing readback is a completion-facing regular Cauchy name rather than a host limit.*
7. *H records componentwise* hsame *transport for S, M, D, R, E, B and the local naming row.*
8. *C records displayed* Cont *replay from stream windows through rate requests, dyadic tolerances, regular-rational readback, Bishop completion handoff, terminal Real seal, provenance, and local naming.*
9. *P is the* Pkg *provenance over* $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$, $\mathsf{DyadicRatCore}^{\uparrow}$, $\mathsf{StreamName}^{\uparrow}$, $\mathsf{RegSeqRat}^{\uparrow}$, $\mathsf{Real}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* $\mathsf{NameCert}$ *rows.*

10. N is the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRateCompletion}^{\uparrow}}$ *row.*
The classifier compares accepted packets only through $S, M, D, R, E, B, H, C, P, N$. It has no coordinate for quotient stream equality, a selected representative, a chosen limit, countable choice, host real equality, or ambient Real completeness.

Theorem 6379.2 (Bishop Cauchy rate completion NameCert obligations)**.** *For every accepted* $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ *carrier*

$$K = (S, M, D, R, E, B, H, C, P, N),$$

the local $\mathsf{NameCert}_{\mathsf{BishopCauchyRateCompletion}^{\uparrow}}$ *surface consists exactly of finite-window exposure through S, rate exposure through M, dyadic tolerance exposure through D, regular-rational readback through R, terminal Real seal exposure through E, Bishop regular completion handoff through B, componentwise* hsame *transport through H, displayed* Cont *replay through C, package provenance through P, and local naming through N.*

Proof. Project the carrier of Definition 6379.1. Its content rows are the finite stream window, rate, dyadic tolerance, regular-rational readback, terminal Real seal, and Bishop completion handoff.

Read the L10 dependency route in order. The $\mathsf{StreamName}^{\uparrow}$ row S exposes the finite observations, M fixes the rate request against those observations, and the $\mathsf{DyadicRatCore}^{\uparrow}$ row D supplies the tolerance ledger used by that rate request.

The $\mathsf{RegSeqRat}^{\uparrow}$ row R is downstream of S, M, D, and the $\mathsf{Real}^{\uparrow}$ seal E is downstream of R. The Bishop regular completion row B names the completion-facing handoff of the same route, so it cannot replace the displayed windows, rates, tolerances, or readback by a selected limit.

Finally H, C, P, N transport, replay, source, and name the same finite packet. Since no carrier coordinate remains outside $S, M, D, R, E, B, H, C, P, N$, the listed rows exhaust the local NameCert obligations. □

Theorem 6379.3 (Bishop Cauchy rate regular readback)**.** *Every accepted* $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ *consumer that reads a terminal* $\mathsf{Real}^{\uparrow}$ *seal from the packet must first read the same finite route*

$$\mathsf{StreamName}^{\uparrow}(S), \quad \mathsf{DyadicRatCore}^{\uparrow}(D), \quad \mathsf{RegSeqRat}^{\uparrow}(R)$$

under the displayed Bishop rate row M and the Bishop regular completion handoff B. The route exports no quotient stream equality, selected limit, countable-choice schedule, host real equality, or ambient completeness coordinate.

Proof. Start with Definition 6379.2. It places the Real seal E after the stream window S, rate row M, dyadic tolerance row D, regular-rational readback R, and Bishop completion handoff B.

The regular readback is fixed by the displayed window and tolerance route. A consumer therefore cannot read R as an abstract quotient class or choose a representative outside S, M, D; the row R is the named regular-rational readback of those finite observations.

The Real-facing seal is also downstream from the same route. The row E is reachable only after R and B have been exposed, and the structural rows H, C, P, N preserve that ordering. Hence every accepted Real-facing read factors through the displayed rate-completion packet rather than through selected limits, host equality, choice, or ambient completeness. □

Falsifiable prediction: An accepted $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ packet is refuted if a Real-facing completion read reaches the terminal seal without displaying the StreamName window,

Bishop rate row, DyadicRatCore tolerance ledger, RegSeqRat readback, Bishop regular completion handoff, transport, replay, provenance, and local NameCert rows.

Independence witness: dyadicratcore, streamname, regseqrat, real, bishop-regular-cauchy-completion: none is bijective to $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ because only the present packet jointly carries finite windows, an explicit Bishop rate row, dyadic tolerances, regular-rational readback, terminal Real seal, Bishop completion handoff, transport, replay, provenance, and local naming.

Closure status for $\mathsf{NameCert}_{\mathsf{BishopCauchyRateCompletion}^{\uparrow}}$.

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{BishopCauchyRateCompletion}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6379.1, Definition 6379.2, and Definition 6379.3 seed the finite Bishop rate-completion route over StreamName windows, DyadicRatCore tolerances, RegSeqRat readback, Real seals, and Bishop regular completion handoff.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{BishopCauchyRateCompletion}^{\uparrow}}, \mathsf{unformalized})$

Bridge status: none

Not claimed: This chapter does not claim quotient stream equality, selected representatives, selected limits, countable choice, host real equality, ambient Real completeness, Lean verification, or bridge checking.

Upgrade path: Obligation closure requires checked carrier admission, rate exposure, dyadic tolerance exactness, regular-rational readback, terminal Real seal handoff, Bishop completion handoff, transport, replay, provenance, and local naming for BEDC.Derived.BishopCauchyRateCompletionUp.

Constructive story: A $\mathsf{BishopCauchyRateCompletion}^{\uparrow}$ packet is generated from finite StreamName windows, a Bishop rate row, DyadicRatCore tolerances, RegSeqRat readback, a terminal RealUp seal, Bishop regular completion handoff, componentwise hsame transport, displayed Cont replay, Pkg provenance, and a local NameCert row.

Chapter 6380

A Concrete Naming Certificate for $\mathsf{ApophaticTLoop}^{\uparrow}$

The $\mathsf{ApophaticTLoop}^{\uparrow}$ packet realizes the apophatic T-loop discipline of subsection 9462.25.12 as finite NameCert data. It sits beside the host-source refusal of chapter 1157, the export refusal gate of chapter 1319, the claim-entry layer of chapter 1209, the external supply boundary of section 9395.1, and the negative certificate boundary of chapter 630. The packet names the records-side return through a gap inscription while keeping the far-end supply position outside the carrier.

Definition 6380.1 (Apophatic T-loop carrier)**.** *An* $\mathsf{ApophaticTLoop}^{\uparrow}$ *carrier is a finite* BHist *packet*

$$T = (H, K, M, R, G, I, L, E, N, H_s, C, P, Q)$$

with the following displayed rows:

1. *H is the records-side history row from which the loop read begins.*
2. *K is the classifier row naming the displayed commitment consumed by the history.*
3. *M is the reality-constrained meaning row attached to that classifier read.*
4. *R is the reality-constraint row whose source is not discharged by the closed records.*
5. *G is the gap row exposing the unclosed GAP_T question without internalizing its far end.*
6. *I is the inscription row, recording the local records-side return point.*
7. *L is the loop ledger, recording that the return reaches a successor or refinement of the records-side history rather than T itself.*
8. *E is the external-supply-boundary row, tying the unclosed source to an accepted boundary form without importing an external primitive.*
9. *N is the negative-certificate-boundary row, recording the cannot-internalize and cannot-export refusals.*
10. *H_s records componentwise* hsame *transport for $H, K, M, R, G, I, L, E, N$ and the local naming row.*
11. *C records displayed* Cont *replay from records through classifier, meaning, reality constraint, gap exposure, inscription, loop ledger, boundary rows, provenance, and local naming.*

12. *P is the* Pkg *provenance over* $\mathsf{ApophaticTLoop}^{\uparrow}$, $\mathsf{HostPrimitiveLeakage}^{\uparrow}$, $\mathsf{CannotClaimExportGate}^{\uparrow}$, $\mathsf{ClaimRegistryLayer}^{\uparrow}$, $\mathsf{NegativeCertificateBoundary}^{\uparrow}$, BHist, hsame, Cont, Pkg, *and* NameCert *rows.*
13. *Q is the local* $\mathsf{NameCert}_{\mathsf{ApophaticTLoop}^{\uparrow}}$ *row.*

The classifier compares accepted packets only through these displayed rows. It has no coordinate for the far-end supply position, host primitive content, total truth, object-layer T*, exportable cannot-claim discharge, or an erased gap.*

Theorem 6380.2 (Apophatic T-loop NameCert obligations)**.** *For every accepted* $\mathsf{ApophaticTLoop}^{\uparrow}$ *carrier*

$$T = (H, K, M, R, G, I, L, E, N, H_s, C, P, Q),$$

the local $\mathsf{NameCert}_{\mathsf{ApophaticTLoop}^{\uparrow}}$ *surface consists exactly of records-side history exposure through H, classifier exposure through K, reality-constrained meaning exposure through M, reality-constraint exposure through R, gap exposure through G, local inscription exposure through I, loop-ledger exposure through L, external-supply-boundary exposure through E, negative-certificate-boundary exposure through N, componentwise* hsame *transport through H_s, displayed* Cont *replay through C, package provenance through P, and local naming through Q.*

Proof. Project the carrier of Definition 6380.1. Its content rows are the records-side history, classifier, reality-constrained meaning, reality constraint, gap, inscription, loop ledger, external boundary, and negative boundary.

The records-facing route follows Definition 9462.746. A history read may return through classifier use, meaning, reality constraint, gap exposure, and local inscription, but the return is a records-side successor or refinement rather than an object-layer copy of T.

The refusal rows fix the boundary. $\mathsf{HostPrimitiveLeakage}^{\uparrow}$ blocks imported host primitives, $\mathsf{CannotClaimExportGate}^{\uparrow}$ blocks overstrong export, $\mathsf{ClaimRegistryLayer}^{\uparrow}$ stores the bounded claim row, and $\mathsf{NegativeCertificateBoundary}^{\uparrow}$ records the cannot-internalize certificate. These rows name the boundary without discharging the far-end supply position.

Finally H_s, C, P, Q transport, replay, source, and name the same finite packet. Since the carrier has no additional coordinate, the listed rows exhaust the local NameCert obligations. □

Theorem 6380.3 (Apophatic T-loop gap inscription handoff)**.** *Every accepted* $\mathsf{ApophaticTLoop}^{\uparrow}$ *consumer that reads the loop ledger L must pass through the gap row G, the local inscription row I, the external-supply-boundary row E, and the negative-certificate-boundary row N. The handoff records a records-side return; it does not internalize the far-end supply position, import host primitive leakage, export an overstrong claim, erase the claim registry, or close the negative certificate boundary.*

Proof. Begin with Definition 6380.2. The loop ledger is exposed only after the records-side history, classifier, meaning, reality constraint, gap, inscription, and boundary rows are visible.

Read the gap and inscription rows together. The gap row G names the unclosed question, while I names the local records-side return point. Thus the handoff is a visible inscription in the history, not a carrier field for the far-end supply position.

Now inspect the refusal siblings. The external-supply-boundary row E keeps the source outside the generated packet, and the negative-certificate-boundary row N records the cannot-internalize refusal. Host primitive leakage and cannot-claim export are therefore blocked by the same route that names the loop.

The structural rows H_s, C, P, Q preserve this ordering. They can replay the displayed handoff, but they cannot add a host primitive, erase the claim registry, close the gap, or turn the inscription into T itself. □

Falsifiable prediction: An accepted $\mathsf{ApophaticTLoop}^{\uparrow}$ packet is refuted if a records-side return is read without the displayed classifier, reality-constrained meaning, reality constraint, gap, local inscription, loop ledger, external-supply boundary, negative-certificate boundary, transport, replay, provenance, and local NameCert rows.

Independence witness: hostprimitiveleakage, cannotclaimexportgate, claimregistrylayer, external-supply-boundary, negativecertificateboundary: none is bijective to $\mathsf{ApophaticTLoop}^{\uparrow}$ because only the present carrier jointly records records-side history, classifier use, meaning, reality constraint, gap inscription, loop return, external boundary, negative certificate, transport, replay, provenance, and local naming.

The Lean follow-up obligation is a real $\mathsf{Nontrivial\ ApophaticTLoop}^{\uparrow}$ instance and a $\mathsf{FieldFaithful\ ApophaticTLoop}^{\uparrow}$ instance in `lean4/BEDC/Derived/ApophaticTLoopUp/TasteGate.lean`. Those instances must distinguish $H, K, M, R, G, I, L, E, N, H_s, C, P, Q$ and must keep the gap row, inscription row, external-supply boundary, and negative-certificate boundary as separate fields.

Closure status for $\mathsf{NameCert}_{\mathsf{ApophaticTLoop}^{\uparrow}}$**.**

Theory closure: $\mathsf{TheoryGate}(\mathsf{NameCert}_{\mathsf{ApophaticTLoop}^{\uparrow}}, \mathsf{seed})$

Scope closed: Definition 6380.1, Definition 6380.2, and Definition 6380.3 seed the finite gap-inscription route for the apophatic T-loop without internalizing the far-end supply position.

Formal verification: $\mathsf{FormalStatus}(\mathsf{NameCert}_{\mathsf{ApophaticTLoop}^{\uparrow}}, \mathsf